

DVBCore

15.3.0

Generated by Doxygen 1.7.6.1

Wed Apr 1 2015 11:33:21

Contents

1	Data Structure Index	1
1.1	Data Structures	1
2	File Index	7
2.1	File List	7
3	Data Structure Documentation	12
3.1	ADB_ALARM_REC Struct Reference	12
3.2	adb_alt_serv_rec Struct Reference	12
3.3	adb_audio_stream_data Struct Reference	13
3.4	adb_bat_version_rec Struct Reference	13
3.5	adb_crid_record Struct Reference	13
3.6	ADB_EVENT_COMPONENT_INFO Struct Reference	14
3.6.1	Detailed Description	14
3.7	adb_event_desc Struct Reference	14
3.8	ADB_EVENT_ITEMIZED_INFO Struct Reference	14
3.9	adb_event_rec Struct Reference	15
3.10	adb_favlist_rec Struct Reference	15
3.11	adb_favserv_rec Struct Reference	15
3.12	adb_icon_image Struct Reference	16
3.13	adb_inb_rec Struct Reference	16
3.14	ADB_LNB_SETTINGS Struct Reference	17
3.15	adb_network_rec Struct Reference	17
3.16	adb_pmt_version_rec Struct Reference	18
3.17	ADB_PVR_RECORD_REC Struct Reference	18
3.18	adb_rct_link_info Struct Reference	19
3.19	adb_satellite_rec Struct Reference	19
3.20	adb_service_rec Struct Reference	20
3.21	adb_stream_data Union Reference	22
3.22	adb_stream_rec Struct Reference	22
3.23	adb_string Struct Reference	22
3.24	adb_subtitle_stream_data Struct Reference	23
3.25	ADB_TIMER_REC Struct Reference	23

3.26	adb_transport_rec Struct Reference	23
3.27	adb_ttext_stream_data Struct Reference	25
3.28	ana_rf_channel_data Struct Reference	25
3.29	APP_SW_VER_STRUCT Struct Reference	25
3.30	cab_rf_channel_data Struct Reference	26
3.31	ci_host_info Struct Reference	26
3.32	ci_licence_update Struct Reference	26
3.33	ci_operator_info Struct Reference	27
3.34	ci_operator_nit Struct Reference	27
3.35	ci_operator_search Struct Reference	27
3.36	ci_query_param Struct Reference	27
3.37	ci_tune_del_sys_desc Struct Reference	28
3.38	ci_tune_service_info Struct Reference	28
3.39	clut Struct Reference	29
3.40	display_set Struct Reference	29
3.41	DVB_VER_STRUCT Struct Reference	29
3.42	epoch_region Struct Reference	30
3.43	LINK_LIST_HEADER Struct Reference	30
3.44	LINK_LIST_PTR_BLK Struct Reference	30
3.45	object Struct Reference	31
3.46	page_composition Struct Reference	31
3.47	region Struct Reference	32
3.48	region_object Struct Reference	32
3.49	S_ACB_AD_PREF Struct Reference	32
3.50	S_ACB_SUBTITLE_PREF Struct Reference	32
3.51	S_ACB_UI_INFO Struct Reference	33
3.52	S_ACTL_OWNER_INFO Struct Reference	33
3.53	s_agc_time_range Struct Reference	33
3.54	S_ALARM_INFO Struct Reference	34
3.55	s_avrec_settings Struct Reference	34
3.56	S_CIP_RELEASE_REPLY Struct Reference	34
3.57	S_CIPPLUS_TUNE_DATA Struct Reference	34
3.58	s_ciplus_tune_service Struct Reference	35
3.59	s_ciplus_tune_transport Struct Reference	35

3.60	s_ebutt_font Struct Reference	35
3.61	s_event Struct Reference	36
3.62	S_EVENT_INFO Struct Reference	36
3.63	S_MANUAL_ANA_TUNING_PARAMS Struct Reference	36
3.64	S_MANUAL_CABLE_TUNING_PARAMS Struct Reference	36
3.65	S_MANUAL_SAT_TUNING_PARAMS Struct Reference	37
3.66	S_MANUAL_TERR_TUNING_PARAMS Struct Reference	37
3.67	S_MANUAL_TUNING_PARAMS Struct Reference	37
3.68	S_NW_ACCESS_POINT Struct Reference	38
3.69	S_NW_ADDR_INFO Struct Reference	38
3.70	S_NW_SOCKET Struct Reference	38
3.71	S_OTA_CAB_TUNING_PARAMS Struct Reference	38
3.72	S_OTA_SAT_TUNING_PARAMS Struct Reference	39
3.73	S_OTA_TER_TUNING_PARAMS Struct Reference	39
3.74	S_OTA_TUNING_PARAMS Struct Reference	39
3.75	s_pvr_pid_info Struct Reference	40
3.76	S_PVR_RECORD_INFO Struct Reference	40
3.77	S_RECTANGLE Struct Reference	40
3.78	S_STB_AV_COPY_PROTECTION Struct Reference	41
3.79	S_STB_AV_VIDEO_INFO Struct Reference	41
3.80	S_STB_MP_COMPONENT_DETAILS Struct Reference	42
3.81	S_STB_MP_START_PARAMS Struct Reference	42
3.82	s_timer Struct Reference	42
3.83	s_vt_options Struct Reference	43
3.84	si_aac_desc Struct Reference	43
3.85	si_ac3_desc Struct Reference	43
3.86	si_ad_desc Struct Reference	44
3.87	si_app_sig_desc Struct Reference	44
3.88	si_bat_freesat_group_name_entry Struct Reference	44
3.89	si_bat_freesat_iactive_storage_desc Struct Reference	44
3.90	si_bat_freesat_region Struct Reference	45
3.91	si_bat_freesat_region_lcn_entry Struct Reference	45
3.92	si_bat_freesat_serv_group_entry Struct Reference	45
3.93	si_bat_table Struct Reference	46

3.94 si_bat_transport_entry Struct Reference	46
3.95 si_ca_desc Struct Reference	47
3.96 si_cable_del_sys_desc Struct Reference	47
3.97 si_ciplus_service Struct Reference	47
3.98 si_component_desc Struct Reference	48
3.99 si_content_desc Struct Reference	48
3.100si_crid_desc Struct Reference	48
3.101si_delivery_sys_desc Union Reference	48
3.102si_dvb_subt_desc Struct Reference	49
3.103si_eit_event_entry Struct Reference	49
3.104si_eit_table Struct Reference	50
3.105si_extended_event_desc Struct Reference	50
3.106si_freesat_info_location Struct Reference	51
3.107si_freesat_lcn Struct Reference	51
3.108si_freesat_linkage_desc Struct Reference	51
3.109si_freesat_prefix_desc Struct Reference	52
3.110si_freesat_tunnelled_data_desc Struct Reference	52
3.111si_fta_content_desc Struct Reference	52
3.112si_guidance_desc Struct Reference	52
3.113si_image_icon_desc Struct Reference	53
3.114si_iso_lang_desc Struct Reference	53
3.115si_linkage_desc Struct Reference	53
3.116si_lto_desc Struct Reference	54
3.117si_multilang_group_name_desc Struct Reference	54
3.118si_multiling_component_desc Struct Reference	54
3.119si_multiling_net_name_desc Struct Reference	55
3.120si_multiling_serv_name_desc Struct Reference	55
3.121SI_MULTILING_SHORT_NAME_DESC Struct Reference	55
3.122si_nit_change_entry Struct Reference	55
3.123si_nit_change_notify_desc Struct Reference	56
3.124si_nit_message_entry Struct Reference	56
3.125si_nit_region_name Struct Reference	56
3.126si_nit_table Struct Reference	57
3.127si_nit_target_region_name_desc Struct Reference	57

3.128	si_nit_transport_entry Struct Reference	58
3.129	SI_NORDIG_LCN_DESC Struct Reference	58
3.130	SI_NORDIG_SERV_LCN Struct Reference	59
3.131	si_parental_rating_desc Struct Reference	59
3.132	si_pat_service_entry Struct Reference	59
3.133	si_pat_table Struct Reference	59
3.134	si_pmt_stream_entry Struct Reference	60
3.135	si_pmt_table Struct Reference	60
3.136	si_preferred_name_desc Struct Reference	61
3.137	si_rct_link_info Struct Reference	61
3.138	si_rct_promo_text Struct Reference	62
3.139	si_rct_subtable Struct Reference	62
3.140	si_rct_subtable_data Struct Reference	62
3.141	si_rct_table Struct Reference	62
3.142	si_sat_del_sys_desc Struct Reference	63
3.143	si_sdt_service_entry Struct Reference	63
3.144	si_sdt_table Struct Reference	64
3.145	si_section_record Struct Reference	64
3.146	si_serv_attribute_desc Struct Reference	64
3.147	si_serv_avail_desc Struct Reference	65
3.148	si_serv_list_desc Struct Reference	65
3.149	si_service_move_desc Struct Reference	65
3.150	si_short_event_desc Struct Reference	65
3.151	si_string Struct Reference	66
3.152	si_t2_del_sys_cell Struct Reference	66
3.153	si_t2_del_sys_desc Struct Reference	66
3.154	si_table_record Struct Reference	66
3.155	si_target_region Struct Reference	67
3.156	si_target_region_desc Struct Reference	67
3.157	si_teletext_desc Struct Reference	67
3.158	SI_TERR_DEL_SYS Struct Reference	68
3.159	si_terr_del_sys_desc Struct Reference	68
3.160	si_time_table Struct Reference	68
3.161	T_BOOT_SW_VER_NUMBER Struct Reference	69

3.162	ter_rf_channel_data Struct Reference	69
3.163	uk_dtt_lcn_desc Struct Reference	69
4	File Documentation	69
4.1	database/dba_nvm/src/dba_nvm.h File Reference	69
4.1.1	Detailed Description	70
4.2	database/dba_nvm/src/stbcsnm.h File Reference	70
4.2.1	Detailed Description	70
4.3	database/dba_nvm/src/stbdbnm.h File Reference	70
4.3.1	Detailed Description	71
4.4	database/dba_nvm/src/stbdbnm.h File Reference	71
4.4.1	Detailed Description	72
4.5	database/dba_nvm/src/stbnvm.h File Reference	72
4.5.1	Detailed Description	73
4.5.2	Function Documentation	73
4.6	database/inc/dba.h File Reference	74
4.6.1	Detailed Description	78
4.6.2	Function Documentation	79
4.7	dvb/inc/ap_cfg.h File Reference	86
4.7.1	Detailed Description	94
4.7.2	Function Documentation	94
4.8	dvb/inc/ap_ci.h File Reference	111
4.8.1	Detailed Description	112
4.8.2	Function Documentation	113
4.9	dvb/inc/ap_cntrl.h File Reference	118
4.9.1	Detailed Description	125
4.9.2	Typedef Documentation	125
4.9.3	Function Documentation	125
4.10	dvb/inc/ap_dbacc.h File Reference	144
4.10.1	Detailed Description	165
4.10.2	Function Documentation	165
4.11	dvb/inc/ap_epgsearch.h File Reference	245
4.11.1	Detailed Description	246
4.11.2	Function Documentation	246

4.12	dvb/inc/ap_ipadd.h File Reference	247
4.12.1	Detailed Description	247
4.12.2	Function Documentation	247
4.13	dvb/inc/ap_pvr.h File Reference	248
4.13.1	Detailed Description	252
4.13.2	Function Documentation	252
4.14	dvb/inc/ap_si.h File Reference	263
4.14.1	Detailed Description	266
4.14.2	Function Documentation	266
4.15	dvb/inc/ap_tmr.h File Reference	271
4.15.1	Detailed Description	275
4.15.2	Function Documentation	275
4.16	dvb/inc/app.h File Reference	289
4.16.1	Detailed Description	291
4.16.2	Typedef Documentation	292
4.16.3	Function Documentation	292
4.17	dvb/inc/app_nvm.h File Reference	293
4.17.1	Detailed Description	295
4.17.2	Function Documentation	295
4.18	dvb/inc/dvbver.h File Reference	296
4.18.1	Detailed Description	296
4.19	dvb/src/ap_ca.h File Reference	297
4.19.1	Detailed Description	297
4.20	dvb/src/ap_cfdat.h File Reference	297
4.20.1	Detailed Description	298
4.21	dvb/src/ap_dbdef.h File Reference	298
4.21.1	Detailed Description	308
4.21.2	Function Documentation	309
4.22	dvb/src/ap_events.h File Reference	334
4.22.1	Detailed Description	334
4.23	dvb/src/ap_state.h File Reference	334
4.23.1	Detailed Description	334
4.24	dvb/src/ap_uiinfo.h File Reference	335
4.24.1	Detailed Description	335

4.25	dvb/src/ciplus_support.h File Reference	335
4.25.1	Detailed Description	336
4.26	externals/HBBTV/src/common.h File Reference	336
4.26.1	Detailed Description	336
4.26.2	Function Documentation	337
4.27	externals/MHEG5/src/dvbmh_int.h File Reference	337
4.27.1	Detailed Description	337
4.28	inc/dbgfuncs.h File Reference	338
4.28.1	Detailed Description	338
4.28.2	Define Documentation	338
4.29	inc/dbgmemory.h File Reference	338
4.29.1	Detailed Description	339
4.30	inc/techtype.h File Reference	339
4.30.1	Detailed Description	339
4.31	middleware/CA/inc/ca_glue.h File Reference	340
4.31.1	Detailed Description	341
4.31.2	Function Documentation	341
4.32	middleware/CI/inc/ci_plus_control.h File Reference	346
4.32.1	Detailed Description	352
4.32.2	Function Documentation	352
4.33	middleware/media/inc/media_image.h File Reference	364
4.33.1	Detailed Description	364
4.34	middleware/ota/inc/stbota.h File Reference	365
4.34.1	Detailed Description	365
4.35	middleware/stb/inc/stbdpc.h File Reference	365
4.35.1	Detailed Description	372
4.35.2	Function Documentation	373
4.36	middleware/stb/inc/stbdsapi.h File Reference	373
4.36.1	Detailed Description	374
4.37	middleware/stb/inc/stbebutt.h File Reference	374
4.37.1	Detailed Description	376
4.37.2	Function Documentation	377
4.38	middleware/stb/inc/stberc.h File Reference	380
4.38.1	Detailed Description	385

4.39	middleware/stb/inc/stbgc.h File Reference	386
4.39.1	Detailed Description	388
4.39.2	Function Documentation	389
4.40	middleware/stb/inc/stbheap.h File Reference	390
4.40.1	Detailed Description	390
4.41	middleware/stb/inc/stbinit.h File Reference	390
4.41.1	Detailed Description	391
4.42	middleware/stb/inc/stbllist.h File Reference	391
4.42.1	Detailed Description	392
4.42.2	Function Documentation	392
4.43	middleware/stb/inc/stbpes.h File Reference	393
4.43.1	Detailed Description	393
4.44	middleware/stb/inc/stbpvr.h File Reference	393
4.44.1	Detailed Description	398
4.44.2	Function Documentation	398
4.45	middleware/stb/inc/stbpvrmsg.h File Reference	402
4.45.1	Detailed Description	402
4.46	middleware/stb/inc/stbsiflt.h File Reference	403
4.46.1	Detailed Description	404
4.46.2	Function Documentation	404
4.47	middleware/stb/inc/stbsipmt.h File Reference	405
4.47.1	Detailed Description	406
4.48	middleware/stb/inc/stbsitab.h File Reference	406
4.48.1	Detailed Description	415
4.48.2	Function Documentation	415
4.49	middleware/stb/inc/stbuni.h File Reference	417
4.49.1	Detailed Description	418
4.50	middleware/stb/inc/stbvtc.h File Reference	418
4.50.1	Detailed Description	419
4.50.2	Function Documentation	419
4.51	middleware/stb/inc/vtctype.h File Reference	424
4.51.1	Detailed Description	424
4.52	middleware/stb/src/ascimap.h File Reference	425
4.52.1	Detailed Description	443

4.53	middleware/stb/src/huffman_table1.h File Reference	444
4.53.1	Detailed Description	444
4.54	middleware/stb/src/huffman_table2.h File Reference	444
4.54.1	Detailed Description	444
4.55	middleware/stb/src/stbds.h File Reference	444
4.55.1	Detailed Description	446
4.56	middleware/stb/src/stbhuffman.h File Reference	446
4.56.1	Detailed Description	447
4.57	middleware/stb/src/stbresmgr.h File Reference	447
4.57.1	Detailed Description	448
4.57.2	Function Documentation	448
4.58	middleware/stb/src/stbsic.h File Reference	449
4.58.1	Detailed Description	449
4.59	middleware/stb/src/stbvi.h File Reference	450
4.59.1	Detailed Description	450
4.60	middleware/stb/src/version.h File Reference	450
4.60.1	Detailed Description	450
4.61	middleware/stb/src/vtc.h File Reference	450
4.61.1	Detailed Description	452
4.61.2	Function Documentation	453
4.62	platform/inc/osdtype.h File Reference	460
4.62.1	Detailed Description	460
4.63	platform/inc/stbhwav.h File Reference	461
4.63.1	Detailed Description	466
4.63.2	Function Documentation	466
4.64	platform/inc/stbhwc.h File Reference	479
4.64.1	Detailed Description	480
4.64.2	Function Documentation	480
4.65	platform/inc/stbhwci.h File Reference	483
4.65.1	Detailed Description	483
4.65.2	Function Documentation	483
4.66	platform/inc/stbhwdmx.h File Reference	484
4.66.1	Detailed Description	486
4.66.2	Function Documentation	486

4.67	platform/inc/stbhwdsd.h File Reference	493
4.67.1	Detailed Description	495
4.67.2	Function Documentation	495
4.68	platform/inc/stbhwfp.h File Reference	502
4.68.1	Detailed Description	503
4.68.2	Function Documentation	503
4.69	platform/inc/stbhwini.h File Reference	506
4.69.1	Detailed Description	506
4.69.2	Function Documentation	506
4.70	platform/inc/stbhwip.h File Reference	507
4.70.1	Detailed Description	508
4.70.2	Function Documentation	508
4.71	platform/inc/stbhwmediaplayer.h File Reference	510
4.71.1	Detailed Description	511
4.71.2	Function Documentation	512
4.72	platform/inc/stbhwmem.h File Reference	516
4.72.1	Detailed Description	518
4.72.2	Function Documentation	518
4.73	platform/inc/stbhwnet.h File Reference	522
4.73.1	Detailed Description	524
4.73.2	Function Documentation	525
4.74	platform/inc/stbhwos.h File Reference	533
4.74.1	Detailed Description	536
4.74.2	Function Documentation	536
4.75	platform/inc/stbhwosd.h File Reference	543
4.75.1	Detailed Description	546
4.75.2	Function Documentation	546
4.76	platform/inc/stbhwun.h File Reference	557
4.76.1	Detailed Description	561
4.76.2	Function Documentation	561
4.77	platform/inc/stbhwupg.h File Reference	570
4.77.1	Detailed Description	571
4.77.2	Function Documentation	571
4.78	platform/inc/stbpvrpr.h File Reference	573

4.78.1 Detailed Description	576
4.78.2 Function Documentation	576
4.79 platform/inc/stbver.h File Reference	587
4.79.1 Detailed Description	587

1 Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

ADB_ALARM_REC	12
adb_alt_serv_rec	12
adb_audio_stream_data	13
adb_bat_version_rec	13
adb_crid_record	13
ADB_EVENT_COMPONENT_INFO	
Structure representing the component information as found in the EIT component_descriptor	14
adb_event_desc	14
ADB_EVENT_ITEMIZED_INFO	14
adb_event_rec	15
adb_favlist_rec	15
adb_favserv_rec	15
adb_icon_image	16
adb_lnb_rec	16
ADB_LNB_SETTINGS	17
adb_network_rec	17
adb_pmt_version_rec	18
ADB_PVR_RECORD_REC	18
adb_rct_link_info	19

adb_satellite_rec	19
adb_service_rec	20
adb_stream_data	22
adb_stream_rec	22
adb_string	22
adb_subtitle_stream_data	23
ADB_TIMER_REC	23
adb_transport_rec	23
adb_ttext_stream_data	25
ana_rf_channel_data	25
APP_SW_VER_STRUCT	25
cab_rf_channel_data	26
ci_host_info	26
ci_licence_update	26
ci_operator_info	27
ci_operator_nit	27
ci_operator_search	27
ci_query_param	27
ci_tune_del_sys_desc	28
ci_tune_service_info	28
clut	29
display_set	29
DVB_VER_STRUCT	29
epoch_region	30
LINK_LIST_HEADER	30
LINK_LIST_PTR_BLK	30
object	31

page_composition	31
region	32
region_object	32
S_ACB_AD_PREF	32
S_ACB_SUBTITLE_PREF	32
S_ACB_UI_INFO	33
S_ACTL_OWNER_INFO	33
s_ags_time_range	33
S_ALARM_INFO	34
s_avrec_settings	34
S_CIP_RELEASE_REPLY	34
S_CIPLUS_TUNE_DATA	34
s_ciplus_tune_service	35
s_ciplus_tune_transport	35
s_ebutt_font	35
s_event	36
S_EVENT_INFO	36
S_MANUAL_ANA_TUNING_PARAMS	36
S_MANUAL_CABLE_TUNING_PARAMS	36
S_MANUAL_SAT_TUNING_PARAMS	37
S_MANUAL_TERR_TUNING_PARAMS	37
S_MANUAL_TUNING_PARAMS	37
S_NW_ACCESS_POINT	38
S_NW_ADDR_INFO	38
S_NW_SOCKET	38
S_OTA_CAB_TUNING_PARAMS	38
S_OTA_SAT_TUNING_PARAMS	39

S_OTA_TER_TUNING_PARAMS	39
S_OTA_TUNING_PARAMS	39
s_pvr_pid_info	40
S_PVR_RECORD_INFO	40
S_RECTANGLE	40
S_STB_AV_COPY_PROTECTION	41
S_STB_AV_VIDEO_INFO	41
S_STB_MP_COMPONENT_DETAILS	42
S_STB_MP_START_PARAMS	42
s_timer	42
s_vt_options	43
si_aac_desc	43
si_ac3_desc	43
si_ad_desc	44
si_app_sig_desc	44
si_bat_freesat_group_name_entry	44
si_bat_freesat_iactive_storage_desc	44
si_bat_freesat_region	45
si_bat_freesat_region_lcn_entry	45
si_bat_freesat_serv_group_entry	45
si_bat_table	46
si_bat_transport_entry	46
si_ca_desc	47
si_cable_del_sys_desc	47
si_ciplus_service	47
si_component_desc	48
si_content_desc	48

si_crid_desc	48
si_delivery_sys_desc	48
si_dvb_subt_desc	49
si_eit_event_entry	49
si_eit_table	50
si_extended_event_desc	50
si_freesat_info_location	51
si_freesat_lcn	51
si_freesat_linkage_desc	51
si_freesat_prefix_desc	52
si_freesat_tunnelled_data_desc	52
si_fta_content_desc	52
si_guidance_desc	52
si_image_icon_desc	53
si_iso_lang_desc	53
si_linkage_desc	53
si_lto_desc	54
si_multilang_group_name_desc	54
si_multiling_component_desc	54
si_multiling_net_name_desc	55
si_multiling_serv_name_desc	55
SI_MULTILING_SHORT_NAME_DESC	55
si_nit_change_entry	55
si_nit_change_notify_desc	56
si_nit_message_entry	56
si_nit_region_name	56
si_nit_table	57

si_nit_target_region_name_desc	57
si_nit_transport_entry	58
SI_NORDIG_LCN_DESC	58
SI_NORDIG_SERV_LCN	59
si_parental_rating_desc	59
si_pat_service_entry	59
si_pat_table	59
si_pmt_stream_entry	60
si_pmt_table	60
si_preferred_name_desc	61
si_rct_link_info	61
si_rct_promo_text	62
si_rct_subtable	62
si_rct_subtable_data	62
si_rct_table	62
si_sat_del_sys_desc	63
si_sdt_service_entry	63
si_sdt_table	64
si_section_record	64
si_serv_attribute_desc	64
si_serv_avail_desc	65
si_serv_list_desc	65
si_service_move_desc	65
si_short_event_desc	65
si_string	66
si_t2_del_sys_cell	66
si_t2_del_sys_desc	66

si_table_record	66
si_target_region	67
si_target_region_desc	67
si_teletext_desc	67
SI_TERR_DEL_SYS	68
si_terr_del_sys_desc	68
si_time_table	68
T_BOOT_SW_VER_NUMBER	69
ter_rf_channel_data	69
uk_dtt_lcn_desc	69

2 File Index

2.1 File List

Here is a list of all documented files with brief descriptions:

database/dba_nvm/src/dba_nvm.h NVM database access defines, structures and functions	69
database/dba_nvm/src/stbcsun.h Header file - Function prototypes for check sum calcs	70
database/dba_nvm/src/stbdbnvm.h Header file - Function prototypes for NVM database	70
database/dba_nvm/src/stbdbram.h Header file - Function prototypes for RAM database	71
database/dba_nvm/src/stbnvm.h Header file - Function prototypes for NVM control	72
database/inc/dba.h Database access defines, structures and public functions	74
dvb/inc/ap_cfg.h Application configuration	86
dvb/inc/ap_ci.h Application level CI control functions	111

dvb/inc/ap_cntrl.h	
Application stb layer control	118
dvb/inc/ap_dbacc.h	
Application database access functions	144
dvb/inc/ap_epgsearch.h	
Macros and function prototypes for public use	245
dvb/inc/ap_ipadd.h	
Contains the initialise functions for IP	247
dvb/inc/ap_pvr.h	
Macros and function prototypes for public use	248
dvb/inc/ap_si.h	
Application level SI task	263
dvb/inc/ap_tmr.h	
Application timer functions and defines	271
dvb/inc/app.h	
Application header file	289
dvb/inc/app_nvm.h	
Header file for NVM data handling functions	293
dvb/inc/dvbver.h	
Application Version support functions	296
dvb/src/ap_ca.h	
Application level CA control functions	297
dvb/src/ap_cfdat.h	
Application configuration data	297
dvb/src/ap_dbdef.h	
Application database control	298
dvb/src/ap_events.h	
	334
dvb/src/ap_state.h	
	334
dvb/src/ap_uiinfo.h	
Definition of the UI information callback function to be used within dvb	335
dvb/src/ciplus_support.h	
CIPLUS support functions	335
externals/HBBTV/inc/hbbtv_ext_control.h	??

externals/HBBTV/src/common.h	
Definition of functions common withing externals/HBBTV	336
externals/MHEG5/src/dvbmh_int.h	
Header internal to DVB MHEG interface files	337
inc/dbgfuncs.h	
Debug functions header file	338
inc/dbgmemory.h	
Macro definition for memory debug	338
inc/techtype.h	
System Wide Global Technical Data Type Definitions	339
middleware/CA/inc/ca_glue.h	
Glue layer between DVB and conditional access systems	340
middleware/CI/inc/ci_plus_control.h	
CIPLUS Glue: Control functions	346
middleware/media/inc/media_image.h	
Media image functions	364
middleware/ota/inc/stbota.h	
API interfacing the middleware with Intellibyte loader library	365
middleware/stb/inc/stbdpc.h	
Header file - macros and function prototypes for public use	365
middleware/stb/inc/stbdsapi.h	
Header file - Function prototypes for DVB subtitles api	373
middleware/stb/inc/stbebutt.h	
Header file - EBU Teletext driver	374
middleware/stb/inc/stberc.h	
Header file - Function prototypes for Event Reporting	380
middleware/stb/inc/stbgc.h	
Header file - macros and function prototypes for public use	386
middleware/stb/inc/stbheap.h	
Header file - Function prototypes for heap memory	390
middleware/stb/inc/stbinit.h	
Header file - macros and function prototypes for public use	390
middleware/stb/inc/stbllist.h	
Header file - Function prototypes for linked lists	391

middleware/stb/inc/ stbpes.h	
Header file - Function prototypes for PES collection task	393
middleware/stb/inc/ stbpvr.h	
Header file - macros and function prototypes for public use	393
middleware/stb/inc/ stbpvrmsg.h	
PVR messages database access functions header file	402
middleware/stb/inc/ stbsiflt.h	
Header file - macros and function prototypes for public use	403
middleware/stb/inc/ stbsipmt.h	
Header file - Function prototypes for operating system	405
middleware/stb/inc/ stbsitab.h	
Header file - macros and function prototypes for public use	406
middleware/stb/inc/ stbuni.h	
Header for STB unicode string handling routines	417
middleware/stb/inc/ stbvtc.h	
Header file - macros and function prototypes for public use	418
middleware/stb/inc/ vtctype.h	
Header file - Function prototypes for A/V control	424
middleware/stb/src/ asciimap.h	
Contains character map tables for converting single byte ascii codes above 0xa0 to unicode codes	425
middleware/stb/src/ huffman_table1.h	
The data for this file is subject to the signing of a license agreement that can be obtained by contacting the DTG (www.dtg.org.uk) who will then provide the tables in electronic form	444
middleware/stb/src/ huffman_table2.h	
The data for this file is subject to the signing of a license agreement that can be obtained by contacting the DTG (www.dtg.org.uk) who will then provide the tables in electronic form	444
middleware/stb/src/ stbds.h	
Header file - Function prototypes for DVB subtitles	444
middleware/stb/src/ stbhuffman.h	
STB middleware Huffman decompression routines defined by the BBC	446
middleware/stb/src/ stbresmgr.h	
STB middleware resource management module header file	447
middleware/stb/src/ stbsic.h	
Header file - macros and function prototypes for public use	449

middleware/stb/src/ stbvbi.h	
Header file for the function prototypes for registering callback function to process Teletext into the VBI	450
middleware/stb/src/ version.h	
Header file - library version number	450
middleware/stb/src/ vtc.h	
Blank description	450
platform/inc/ osdtype.h	
Header file - Function prototypes for A/V control	460
platform/inc/ stbhwav.h	
Header file - Function prototypes for A/V control	461
platform/inc/ stbhwc.h	
Function prototypes for HW control	479
platform/inc/ stbhwci.h	
Header File for DVB-CI low-level Physical layer	483
platform/inc/stbhwcrypt.h	??
platform/inc/ stbhwdmx.h	
Header file - Function prototypes for Demux control	484
platform/inc/ stbhwdsk.h	
Function prototypes for disk functions	493
platform/inc/ stbhwfp.h	
Header file - Function prototypes for front panel control	502
platform/inc/ stbhwini.h	
Header file - Function prototype for init function	506
platform/inc/ stbhwip.h	
Macros and function prototypes for public use	507
platform/inc/ stbhwmediaplayer.h	
Media player API	510
platform/inc/ stbhwmem.h	
Header file - Function prototypes for NVM and Heap	516
platform/inc/ stbhwnet.h	
Socket functions	522
platform/inc/ stbhwsos.h	
Header file - Function prototypes for operating system	533

platform/inc/stbhwozd.h	
Header file - Function prototypes for OSD control	543
platform/inc/stbhwtn.h	
Header file - Function prototypes for tuner control	557
platform/inc/stbhwupg.h	
Functions for writing upgrade modules to non volatile memory	570
platform/inc/stbpvrpr.h	
Header file - macros and function prototypes for public use	573
platform/inc/stbver.h	
Header file - Function prototypes for version numbers	587

3 Data Structure Documentation

3.1 ADB_ALARM_REC Struct Reference

Data Fields

- **BOOLEAN** `change_service`
- **U16BIT** `service_id`
- **U16BIT** `transport_id`
- **U16BIT** `orig_net_id`
- **BOOLEAN** `ramp_volume`

The documentation for this struct was generated from the following file:

- [dvb/src/ap_dbdef.h](#)

3.2 adb_alt_serv_rec Struct Reference

Data Fields

- struct [adb_alt_serv_rec](#) * `next`
- **U16BIT** `onet_id`
- **U16BIT** `tran_id`
- **U16BIT** `serv_id`
- **U8BIT** `link_type`

The documentation for this struct was generated from the following file:

- [dvb/src/ap_dbdef.h](#)

3.3 adb_audio_stream_data Struct Reference

Data Fields

- U32BIT **lang_code**
- ADB_AUDIO_TYPE **type**
- E_STB_DP_AUDIO_MODE **mode**

The documentation for this struct was generated from the following file:

- [dvb/src/ap_dbdef.h](#)

3.4 adb_bat_version_rec Struct Reference

Data Fields

- struct [adb_bat_version_rec](#) * **next**
- U16BIT **bouquet_id**
- U8BIT **version**

The documentation for this struct was generated from the following file:

- [dvb/src/ap_dbdef.h](#)

3.5 adb_crid_record Struct Reference

Data Fields

- [LINK_LIST_PTR_BLK](#) **ptrs**
- void * **dba_rec**
- U16BIT **eit_date**
- BOOLEAN **series_flag**
- BOOLEAN **recommended_flag**
- BOOLEAN **radio_service**
- U32DHMS **date_time**
- U16BIT **serv_id**
- [ADB_STRING](#) * **name_str**
- [ADB_STRING](#) * **crid_str**

The documentation for this struct was generated from the following file:

- [dvb/src/ap_dbdef.h](#)

3.6 ADB_EVENT_COMPONENT_INFO Struct Reference

Structure representing the component information as found in the EIT component_descriptor.

```
#include <ap_dbacc.h>
```

Data Fields

- U8BIT **stream_content**
- U8BIT **component_type**
- U8BIT **component_tag**
- U8BIT **language_code** [3]

3.6.1 Detailed Description

Structure representing the component information as found in the EIT component_descriptor.

The documentation for this struct was generated from the following file:

- [dvb/inc/ap_dbacc.h](#)

3.7 adb_event_desc Struct Reference

Data Fields

- U8BIT * **desc_data**
- struct [adb_event_desc](#) * **next**

The documentation for this struct was generated from the following file:

- [dvb/src/ap_dbdef.h](#)

3.8 ADB_EVENT_ITEMIZED_INFO Struct Reference

Data Fields

- U8BIT * **item_description**
- U8BIT * **item**

The documentation for this struct was generated from the following file:

- [dvb/inc/ap_dbacc.h](#)

3.9 adb_event_rec Struct Reference

Data Fields

- struct [adb_event_rec](#) * **next**
- void * **dba_rec**
- U16BIT **event_id**
- U32DHMS **start**
- U32DHMS **duration**
- U8BIT **running_status**
- BOOLEAN **free_to_air**
- U8BIT **version**
- BOOLEAN **has_content_management_desc**
- BOOLEAN **do_not_scramble**
- [ADB_EVENT_DESC](#) * **desc_list_head**
- [ADB_EVENT_DESC](#) * **desc_list_tail**

The documentation for this struct was generated from the following file:

- [dvb/src/ap_dbdef.h](#)

3.10 adb_favlist_rec Struct Reference

Data Fields

- [LINK_LIST_PTR_BLK](#) **ptrs**
- void * **dba_rec**
- U8BIT **list_id**
- U8BIT **index**
- U32BIT **user_data**
- [ADB_STRING](#) * **name**
- [LINK_LIST_HEADER](#) **serv_list**

The documentation for this struct was generated from the following file:

- [dvb/src/ap_dbdef.h](#)

3.11 adb_favserv_rec Struct Reference

Data Fields

- [LINK_LIST_PTR_BLK](#) **ptrs**
- void * **dba_rec**
- [ADB_SERVICE_REC](#) * **serv_ptr**
- U8BIT **list_id**
- U16BIT **index**

The documentation for this struct was generated from the following file:

- [dvb/src/ap_dbdef.h](#)

3.12 adb_icon_image Struct Reference

Data Fields

- struct [adb_icon_image](#) * **next**
- U8BIT **icon_id**
- BOOLEAN **position_defined**
- E_ICON_COORD_SYSTEM **coord_system**
- U16BIT **x_pos**
- U16BIT **y_pos**
- U16BIT **width**
- U16BIT **height**
- E_ICON_TYPE **icon_type**
- E_ICON_TRANSPORT_MODE **transport_mode**
- U8BIT * **icon_url**
- U32BIT **data_len**
- U8BIT * **icon_data**
- void(* **destroy_func**)(void *)
- void * **dsm_handle**

The documentation for this struct was generated from the following file:

- [dvb/src/ap_dbdef.h](#)

3.13 adb_lnb_rec Struct Reference

Data Fields

- [LINK_LIST_PTR_BLK](#) **ptrs**
- void * **dba_rec**
- E_STB_DP_LNB_TYPE **type**
- U16BIT **freq_low**
- U16BIT **freq_high**
- E_STB_DP_LNB_POWER **power**
- BOOLEAN **is_22k**
- BOOLEAN **is_12v**
- BOOLEAN **is_pulse_posn**
- BOOLEAN **is_diseqc_posn**
- E_STB_DP_DISEQC_TONE **diseqc_tone**
- E_STB_DP_DISEQC_CSWITCH **c_switch**
- U8BIT **u_switch**

- BOOLEAN **is_smatv**
- U8BIT **diseqc_repeats**
- U32BIT **unicable_if**
- U8BIT **unicable_chan**

The documentation for this struct was generated from the following file:

- [dvb/src/ap_dbdef.h](#)

3.14 ADB_LNB_SETTINGS Struct Reference

Data Fields

- E_STB_DP_LNB_TYPE **type**
- U16BIT **freq_low**
- U16BIT **freq_high**
- E_STB_DP_LNB_POWER **power**
- E_STB_DP_DISEQC_TONE **diseqc_tone**
- E_STB_DP_DISEQC_CSWITCH **c_switch**
- BOOLEAN **is_22k**
- BOOLEAN **is_12v**
- BOOLEAN **is_pulse_posn**
- BOOLEAN **is_diseqc_posn**
- BOOLEAN **is_smatv**
- U8BIT **diseqc_repeats**
- U8BIT **u_switch**
- U8BIT **unicable_chan**
- U32BIT **unicable_if**

The documentation for this struct was generated from the following file:

- [dvb/inc/ap_dbacc.h](#)

3.15 adb_network_rec Struct Reference

Data Fields

- [LINK_LIST_PTR_BLK](#) **ptrs**
- void * **dba_rec**
- U16BIT **net_id**
- U8BIT **nit_version**
- BOOLEAN **nit_version_changed**
- [ADB_STRING](#) * **name_str**
- [ADB_STRING](#) * **name_array** [ACFG_NUM_DB_LANGUAGES]
- BOOLEAN **has_fta_desc**

- BOOLEAN **do_not_scramble**
- [SI_NIT_TARGET_REGION_NAME_DESC](#) * **target_region_name_list**
- [SI_TARGET_REGION_DESC](#) * **target_region_list**
- [SI_LINKAGE_DESC_ENTRY](#) * **last_linkage_entry**
- [SI_LINKAGE_DESC_ENTRY](#) * **linkage_desc_list**
- U16BIT **num_linkage_entries**
- [ADB_STRING](#) * **def_authority**
- [ADB_SATELLITE_REC](#) * **satellite**
- ADB_PROFILE_TYPE **profile_type**

The documentation for this struct was generated from the following file:

- [dvb/src/ap_dbdef.h](#)

3.16 adb_pmt_version_rec Struct Reference

Data Fields

- struct [adb_pmt_version_rec](#) * **next**
- U16BIT **serv_id**
- U8BIT **version**

The documentation for this struct was generated from the following file:

- [dvb/src/ap_dbdef.h](#)

3.17 ADB_PVR_RECORD_REC Struct Reference

Data Fields

- U32DHMS **duration**
- BOOLEAN **event_triggered**
- U16BIT **event_id**
- U16BIT **service_id**
- U16BIT **transport_id**
- U16BIT **orig_net_id**
- U8BIT **prog_crid** [TMR_PVR_CRID_LEN_MAX]
- U8BIT **other_crid** [TMR_PVR_CRID_LEN_MAX]
- U16BIT **disk_id**
- BOOLEAN **recommendation**
- U16BIT **notify_time**
- U8BIT **path**
- U32BIT **recording_handle**
- U8BIT **additional_info** [TMR_PVR_ADDINFO_LEN_MAX]
- S32BIT **start_padding**

- S32BIT **end_padding**

The documentation for this struct was generated from the following file:

- [dvb/src/ap_dbdef.h](#)

3.18 adb_rct.link_info Struct Reference

Data Fields

- struct [adb_rct_link_info](#) * **next**
- BOOLEAN **is_group_trailer**
- U8BIT * **uri_string**
- BOOLEAN **can_use_default_icon**
- U8BIT **icon_id**
- [ADB_STRING](#) * **promo_text** [ACFG_NUM_DB_LANGUAGES]
- [ADB_STRING](#) * **event_name**

The documentation for this struct was generated from the following file:

- [dvb/src/ap_dbdef.h](#)

3.19 adb_satellite_rec Struct Reference

Data Fields

- [LINK_LIST_PTR_BLK](#) **ptrs**
- void * **dba_rec**
- [ADB_LNB_REC](#) * **lnb**
- [ADB_STRING](#) * **name**
- U16BIT **dish_pos**
- U16BIT **long_pos**
- BOOLEAN **east_west**
- BOOLEAN **has_fta_desc**
- BOOLEAN **do_not_scramble**
- [ADB_STRING](#) * **def_authority**
- [ADB_BAT_VERSION_REC](#) * **bat_version_list**

The documentation for this struct was generated from the following file:

- [dvb/src/ap_dbdef.h](#)

3.20 adb_service_rec Struct Reference

Data Fields

- [LINK_LIST_PTR_BLK](#) ptrs
- void * **dba_rec**
- [ADB_TRANSPORT_REC](#) * transport
- ADB_SERVICE_TYPE **serv_type**
- U16BIT **serv_id**
- U16BIT **serv_lcn**
- U16BIT **allocated_lcn**
- U16BIT **old_allocated_lcn**
- [ADB_STRING](#) * **name_str**
- [ADB_STRING](#) * **provider_str**
- [ADB_STRING](#) * **short_name_str**
- [ADB_STRING](#) * **short_name_array** [ACFG_NUM_DB_LANGUAGES]
- [ADB_STRING](#) * **name_array** [ACFG_NUM_DB_LANGUAGES][ADB_NUM_SERV_NAME_IDS]
- [ADB_STRING](#) * **provider_array** [ACFG_NUM_DB_LANGUAGES]
- [ADB_STRING](#) * **guidance** [ACFG_NUM_DB_LANGUAGES]
- [ADB_STREAM_REC](#) * **stream_list**
- [ADB_ALT_SERV_REC](#) * **alt_serv_list**
- U8BIT **fav_groups**
- BOOLEAN **sched_disabled**
- BOOLEAN **now_next_disabled**
- BOOLEAN **eit_now_next_avail**
- [ADB_EVENT_REC](#) * **now_event**
- [ADB_EVENT_REC](#) * **next_event**
- BOOLEAN **eit_sched_avail**
- U16BIT **num_events_in_schedule**
- [ADB_EVENT_REC](#) * **event_schedule**
- BOOLEAN **pmt_received**
- U16BIT **pcr_pid**
- U16BIT **video_pid**
- U16BIT **audio_pid**
- U16BIT **ad_pid**
- U16BIT **ttext_pid**
- U8BIT **ttext_mag**
- U8BIT **ttext_page**
- U16BIT **subtitle_pid**
- U16BIT **subtitle_cpage**
- U16BIT **subtitle_apage**
- ADB_STREAM_TYPE **video_type**
- ADB_STREAM_TYPE **audio_type**
- BOOLEAN **reqd_audio_valid**
- U32BIT **reqd_audio_lang_code**
- ADB_AUDIO_TYPE **reqd_audio_type**

- ADB_STREAM_TYPE reqd_stream_type
- BOOLEAN reqd_subtitle_valid
- U32BIT reqd_subtitle_lang_code
- ADB_SUBTITLE_TYPE reqd_subtitle_type
- BOOLEAN reqd_ttext_valid
- U32BIT reqd_ttext_lang_code
- ADB_TELETEXT_TYPE reqd_ttext_type
- BOOLEAN found
- U8BIT running_status
- BOOLEAN not_running
- BOOLEAN unavailable
- BOOLEAN new_service
- BOOLEAN locked
- BOOLEAN hidden
- BOOLEAN selectable
- BOOLEAN lcn_editable
- BOOLEAN scrambled
- BOOLEAN has_fta_desc
- BOOLEAN do_not_scramble
- [SI_TARGET_REGION_DESC](#) * target_region_list
- [ADB_STRING](#) * def_authority
- U16BIT rct_pid
- [ADB_RCT_LINK_INFO](#) * rct_link_list
- [ADB_IMAGE_ICON](#) * icon_list
- U8BIT * pmt_data
- U16BIT pmt_data_len
- U16BIT pmt_pid
- [SI_LINKAGE_DESC_ENTRY](#) * last_linkage_entry
- [SI_LINKAGE_DESC_ENTRY](#) * linkage_desc_list
- U16BIT num_linkage_entries
- [SI_LCN_DESC](#) * hd_lcn_desc
- BOOLEAN has_ca_descriptor
- U16BIT freesat_id
- U16BIT region_id
- U16BIT bat_pid
- U16BIT eit_pf_pid
- U16BIT eit_pf_plus_pid
- U16BIT eit_sched_pid
- U16BIT nit_pid
- U16BIT sdt_pid
- U16BIT tdt_pid
- U16BIT tot_pid

The documentation for this struct was generated from the following file:

- [dvb/src/ap_dbdef.h](#)

3.21 adb_stream_data Union Reference

Data Fields

- [ADB_AUDIO_STREAM_DATA](#) **audio**
- [ADB_SUBTITLE_STREAM_DATA](#) **subtitle**
- [ADB_TTEXT_STREAM_DATA](#) **ttext**

The documentation for this union was generated from the following file:

- [dvb/src/ap_dbdef.h](#)

3.22 adb_stream_rec Struct Reference

Data Fields

- struct [adb_stream_rec](#) * **next**
- ADB_STREAM_TYPE **type**
- U8BIT * **tag_array**
- U8BIT **num_tags**
- U16BIT **pid**
- BOOLEAN **in_use**
- [ADB_STREAM_DATA](#) **data**
- BOOLEAN **has_ca_descriptor**

The documentation for this struct was generated from the following file:

- [dvb/src/ap_dbdef.h](#)

3.23 adb_string Struct Reference

Data Fields

- U32BIT **lang_code**
- U16BIT **nbytes**
- U8BIT * **str_ptr**

The documentation for this struct was generated from the following file:

- [dvb/src/ap_dbdef.h](#)

3.24 adb_subtitle_stream_data Struct Reference

Data Fields

- U32BIT **lang_code**
- ADB_SUBTITLE_TYPE **type**
- U16BIT **composition_page**
- U16BIT **ancillary_page**

The documentation for this struct was generated from the following file:

- [dvb/src/ap_dbdef.h](#)

3.25 ADB_TIMER_REC Struct Reference

Data Fields

- [LINK_LIST_PTR_BLK](#) **ptrs**
- void * **dba_rec**
- U32BIT **handle**
- E_TIMER_TYPE **type**
- U32DHMS **start_time**
- U8BIT **name** [TMR_MAX_NAME_LENGTH]
- E_TIMER_FREQ **frequency**
- BOOLEAN **expired**
- BOOLEAN **starting**
- BOOLEAN **started**
- BOOLEAN **selected**
- BOOLEAN **missed**
- union {
 - [ADB_ALARM_REC](#) **alarm**
 - [ADB_PVR_RECORD_REC](#) **record**
- **u**

The documentation for this struct was generated from the following file:

- [dvb/src/ap_dbdef.h](#)

3.26 adb_transport_rec Struct Reference

Data Fields

- [LINK_LIST_PTR_BLK](#) **ptrs**
- void * **dba_rec**
- U16BIT **tran_id**

- U16BIT **orig_net_id**
 - [ADB_STRING](#) * **name_str**
 - U32BIT **frequency**
 - [ADB_NETWORK_REC](#) * **network**
 - E_STB_DP_SIGNAL_TYPE **sig_type**
 - U8BIT **signal_level_at_search**
 - union {
 - struct **_analog_trans_rec** {
 - S8BIT **freq_offset**
 - E_STB_DP_ANALOG_VIDEO_TYPE **vtype**
 - anal**
 - struct **_terr_trans_rec** {
 - E_STB_DP_TBWIDTH **bwidth**
 - E_STB_DP_TMODE **tmode**
 - E_STB_DP_TTYPE **terr_type**
 - U8BIT **plp_id**
 - S8BIT **freq_offset**
 - E_STB_TUNE_TCONST **constellation**
 - E_STB_TUNE_THIERARCHY **hierarchy**
 - E_STB_TUNE_TCODERATE **lp_code_rate**
 - E_STB_TUNE_TCODERATE **hp_code_rate**
 - E_STB_TUNE_TGUARDINT **guard_int**
 - [SI_TARGET_REGION_DESC](#) * **target_region_list**
 - terr**
 - struct **_cable_trans_rec** {
 - E_STB_DP_CMODE **cmode**
 - U16BIT **symbol_rate**
 - cab**
 - struct **_sat_trans_rec** {
 - U16BIT **symbol_rate**
 - E_STB_DP_POLARITY **polarity**
 - E_STB_DP_FEC **fec_code**
 - E_STB_DP_FEC_TYPE **fec_type**
 - BOOLEAN **dvb_s2**
 - E_STB_DP_MODULATION **modulation**
 - sat**
 - u**
- U8BIT **pat_version**
 - BOOLEAN **sdt_received**
 - BOOLEAN **sdt_version_changed**
 - U8BIT **sdt_version**
 - [ADB_PMT_VERSION_REC](#) * **pmt_version_list**
 - BOOLEAN **has_fta_desc**
 - BOOLEAN **do_not_scramble**
 - [ADB_STRING](#) * **def_authority**
 - BOOLEAN **searched**
 - BOOLEAN **available**

The documentation for this struct was generated from the following file:

- [dvb/src/ap_dbdef.h](#)

3.27 adb_ttext_stream_data Struct Reference

Data Fields

- U32BIT **lang_code**
- U8BIT **type**
- U8BIT **magazine**
- U8BIT **page**

The documentation for this struct was generated from the following file:

- [dvb/src/ap_dbdef.h](#)

3.28 ana_rf_channel_data Struct Reference

Data Fields

- U8BIT * **name**
- U32BIT **freq_hz**

The documentation for this struct was generated from the following file:

- [dvb/inc/ap_cfg.h](#)

3.29 APP_SW_VER_STRUCT Struct Reference

Data Fields

- U8BIT **product**
- U8BIT **major**
- U8BIT **minor**
- U8BIT **release**
- U32BIT **revision**

The documentation for this struct was generated from the following file:

- [platform/inc/stbver.h](#)

3.30 cab_rf_channel_data Struct Reference

Data Fields

- U8BIT * **name**
- U32BIT **freq_hz**
- E_STB_DP_CMODE **mode**
- U16BIT **symbol_rate**

The documentation for this struct was generated from the following file:

- [dvb/inc/ap_cfg.h](#)

3.31 ci_host_info Struct Reference

Data Fields

- BOOLEAN **standby**
- U8BIT **num_service_types**
- U32BIT **service_types**
- U16BIT **delivery_types**
- U16BIT **app_types**

The documentation for this struct was generated from the following file:

- [middleware/CI/inc/ci_plus_control.h](#)

3.32 ci_licence_update Struct Reference

Data Fields

- U8BIT **slot_id**
- U8BIT **cicam_id** [8]
- U8BIT **cicam_id_len**
- U8BIT **raw_uri** [CIP_URI_LEN]
- U8BIT * **licence**
- U16BIT **licence_len**

The documentation for this struct was generated from the following file:

- [middleware/CI/inc/ci_plus_control.h](#)

3.33 ci_operator_info Struct Reference

Data Fields

- U16BIT **cicam_onet_id**
- U32BIT **cicam_identifier**

The documentation for this struct was generated from the following file:

- [middleware/CI/inc/ci_plus_control.h](#)

3.34 ci_operator_nit Struct Reference

Data Fields

- U16BIT **cicam_onet_id**
- U32BIT **cicam_identifier**
- U32BIT **lang_code**
- U8BIT **profile_name_length**
- U8BIT * **profile_name**
- [SI_NIT_TABLE](#) * **table**

The documentation for this struct was generated from the following file:

- [middleware/CI/inc/ci_plus_control.h](#)

3.35 ci_operator_search Struct Reference

Data Fields

- E_CIP_OPERATOR_SEARCH_REQUEST_TYPE **refresh_request**
- U16BIT **refresh_date**
- U8BIT **refresh_hour**
- U8BIT **refresh_min**

The documentation for this struct was generated from the following file:

- [middleware/CI/inc/ci_plus_control.h](#)

3.36 ci_query_param Struct Reference

Data Fields

- E_CI_QUERY **query**
- U32BIT **module**

- union {
 - S_CIP_TUNE_SERVICE_INFO [tune_service](#)
 - S_CIP_TUNE_DEL_SYS_DESC [tune_del_sys](#)
 - S_CIP_OPERATOR_NIT [nit](#)
 - S_CIP_HOST_INFO [host_info](#)
 - S_CIP_OPERATOR_SEARCH [op_search](#)
 - S_CIP_OPERATOR_INFO [op_info](#)
 - S_CIP_LICENCE_UPDATE [licence_info](#)
- } **u**

The documentation for this struct was generated from the following file:

- [middleware/CI/inc/ci_plus_control.h](#)

3.37 ci_tune_del_sys_desc Struct Reference

Data Fields

- SI_DELIVERY_SYS_DESC_TYPE **type**
- [SI_DELIVERY_SYS_DESC](#) * **desc**
- U16BIT **service_id**
- U8BIT * **pmt**
- U8BIT **service_type**
- [SI_STRING_DESC](#) * **service_name**
- [SI_SHORT_EVENT_DESC](#) * **event_desc**

The documentation for this struct was generated from the following file:

- [middleware/CI/inc/ci_plus_control.h](#)

3.38 ci_tune_service_info Struct Reference

Data Fields

- U16BIT **nid**
- U16BIT **onid**
- U16BIT **tsid**
- U16BIT **sid**

The documentation for this struct was generated from the following file:

- [middleware/CI/inc/ci_plus_control.h](#)

3.39 clut Struct Reference

Data Fields

- struct [clut](#) * **next**
- U8BIT **clut_id**
- U8BIT **clut_version_number**
- U32BIT * **clut_2_bit**
- U32BIT * **clut_4_bit**
- U32BIT * **clut_8_bit**
- BOOLEAN **default_2_bit**
- BOOLEAN **default_4_bit**
- BOOLEAN **default_8_bit**

The documentation for this struct was generated from the following file:

- [middleware/stb/src/stbds.h](#)

3.40 display_set Struct Reference

Data Fields

- [S_PAGE_COMPOSITION](#) * **page_display_buffer**
- [S_PAGE_COMPOSITION](#) * **page_composition_buffer**
- [S_EPOCH_REGION](#) * **region_list**
- [S_OBJECT](#) * **object_list**
- [S_CLUT](#) * **clut_list**

The documentation for this struct was generated from the following file:

- [middleware/stb/src/stbds.h](#)

3.41 DVB_VER_STRUCT Struct Reference

Data Fields

- U8BIT **major**
- U8BIT **minor**
- U32BIT **revision**

The documentation for this struct was generated from the following file:

- [dvb/inc/dvbver.h](#)

3.42 epoch_region Struct Reference

Data Fields

- struct [epoch_region](#) * **next**
- U8BIT **region_id**
- U8BIT **region_version_number**
- U8BIT **region_fill_flag**
- U16BIT **region_width**
- U16BIT **region_height**
- U8BIT **region_level_of_compatibility**
- U8BIT **region_colour_depth**
- U8BIT **region_clut_id**
- U8BIT **region_8_bit_pixel_code**
- U8BIT **region_4_bit_pixel_code**
- U8BIT **region_2_bit_pixel_code**
- [S_REGION_OBJECT](#) * **region_object_list**

The documentation for this struct was generated from the following file:

- [middleware/stb/src/stbds.h](#)

3.43 LINK_LIST_HEADER Struct Reference

Data Fields

- void * **null_next**
- void * **first**
- void * **last**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbllist.h](#)

3.44 LINK_LIST_PTR_BLK Struct Reference

Data Fields

- void * **next**
- void * **prev**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbllist.h](#)

3.45 object Struct Reference

Data Fields

- struct [object](#) * next
- U16BIT **object_id**
- U8BIT **object_version_number**
- U8BIT **object_coding_method**
- U8BIT **non_modifying_colour_flag**
- U8BIT * **map_table_2_to_4_bit**
- U8BIT * **map_table_2_to_8_bit**
- U8BIT * **map_table_4_to_8_bit**
- U8BIT * **character_code**
- U16BIT **num_chars**
- U8BIT * **bitmap**
- U16BIT **width**
- U16BIT **height**

The documentation for this struct was generated from the following file:

- [middleware/stb/src/stbds.h](#)

3.46 page_composition Struct Reference

Data Fields

- U8BIT **page_time_out**
- U8BIT **page_version_number**
- U8BIT **page_state**
- [S_REGION](#) * **region_list**
- U8BIT **pts** [5]
- U32BIT **presentation_time_ms**
- U32BIT **timeout_start**
- BOOLEAN **display_set_shown**
- BOOLEAN **display_set_removed**
- BOOLEAN **dds_present**
- U8BIT **dds_version**
- U16BIT **display_width**
- U16BIT **display_height**
- BOOLEAN **display_window**
- U16BIT **window_x**
- U16BIT **window_y**
- U16BIT **window_width**
- U16BIT **window_height**

The documentation for this struct was generated from the following file:

- [middleware/stb/src/stbds.h](#)

3.47 region Struct Reference

Data Fields

- struct [region](#) * **next**
- U8BIT **region_id**
- U16BIT **region_horizontal_address**
- U16BIT **region_vertical_address**

The documentation for this struct was generated from the following file:

- [middleware/stb/src/stbds.h](#)

3.48 region_object Struct Reference

Data Fields

- struct [region_object](#) * **next**
- U16BIT **object_id**
- U8BIT **object_type**
- U8BIT **object_provider_flag**
- U16BIT **object_horizontal_position**
- U16BIT **object_vertical_position**
- U8BIT **object_foreground_pixel_code**
- U8BIT **object_background_pixel_code**

The documentation for this struct was generated from the following file:

- [middleware/stb/src/stbds.h](#)

3.49 S_ACB_AD_PREF Struct Reference

Data Fields

- BOOLEAN **pref**

The documentation for this struct was generated from the following file:

- [dvb/inc/app.h](#)

3.50 S_ACB_SUBTITLE_PREF Struct Reference

Data Fields

- BOOLEAN **pref**

The documentation for this struct was generated from the following file:

- [dvb/inc/app.h](#)

3.51 S_ACB_UI_INFO Struct Reference

Data Fields

- **E_ACB_INFO_TYPE** **type**
- union {
 - [S_ACB_SUBTITLE_PREF](#) **subtitle**
 - [S_ACB_AD_PREF](#) **ad**
 - U8BIT **ui_lang_id**
 - S32BIT **bannerTransparency**
 - S32BIT **bannerTimeout**
- **u**

The documentation for this struct was generated from the following file:

- [dvb/inc/app.h](#)

3.52 S_ACTL_OWNER_INFO Struct Reference

Data Fields

- **E_STB_DP_RES_OWNER** **owner**
- void * **data**
- U32BIT **data_size**

The documentation for this struct was generated from the following file:

- [dvb/inc/ap_cntrl.h](#)

3.53 s_ags_time_range Struct Reference

Data Fields

- U8BIT **min_hour**
- U8BIT **min_minutes**
- U8BIT **min_seconds**
- U8BIT **max_hour**
- U8BIT **max_minutes**
- U8BIT **max_seconds**

The documentation for this struct was generated from the following file:

- [dvb/inc/ap_epgsearch.h](#)

3.54 S_ALARM_INFO Struct Reference

Data Fields

- BOOLEAN **change_service**
- U16BIT **service_id**
- U16BIT **transport_id**
- U16BIT **orig_net_id**
- BOOLEAN **ramp_volume**

The documentation for this struct was generated from the following file:

- [dvb/inc/ap_tmr.h](#)

3.55 s_avrec_settings Struct Reference

Data Fields

- U16BIT **av_rec_lcn**
- BOOLEAN **standby_mode**

The documentation for this struct was generated from the following file:

- [dvb/inc/ap_tmr.h](#)

3.56 S_CIP_RELEASE_REPLY Struct Reference

Data Fields

- U32BIT **module**
- U8BIT **reply**

The documentation for this struct was generated from the following file:

- [middleware/CI/inc/ci_plus_control.h](#)

3.57 S_CIPLUS_TUNE_DATA Struct Reference

Data Fields

- U32BIT **module**
- E_CIPLUS_TUNE_TYPE **type**
- union {
 - struct [s_ciplus_tune_service](#) **service**
 - struct [s_ciplus_tune_transport](#) **transport**
- **u**

The documentation for this struct was generated from the following file:

- [dvb/src/ciplus_support.h](#)

3.58 s_ciplus_tune_service Struct Reference

Data Fields

- void * **s_ptr**
- E_ACTL_SI_SRCH_REQD **reqd_si**
- U8BIT * **pmt**

The documentation for this struct was generated from the following file:

- [dvb/src/ciplus_support.h](#)

3.59 s_ciplus_tune_transport Struct Reference

Data Fields

- void * **t_ptr**
- E_ACTL_SI_SRCH_REQD **reqd_si**

The documentation for this struct was generated from the following file:

- [dvb/src/ciplus_support.h](#)

3.60 s_ebutt_font Struct Reference

Data Fields

- U8BIT **character_width**
- U8BIT **character_height**
- U32BIT **seperator_row_mask**
- U32BIT **seperator_column_mask**
- U8BIT * **font_table_set_ptr** [13]
- U8BIT * **font_table_option_subset_ptr**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbebutt.h](#)

3.61 **s_event** Struct Reference

Data Fields

- void * **serv_ptr**
- U16BIT **event_id**

The documentation for this struct was generated from the following file:

- [dvb/src/ap_dbdef.h](#)

3.62 **S_EVENT_INFO** Struct Reference

Data Fields

- U32BIT **event_code**
- void * **data**
- U32BIT **data_size**

The documentation for this struct was generated from the following file:

- [dvb/src/ap_events.h](#)

3.63 **S_MANUAL_ANA_TUNING_PARAMS** Struct Reference

Data Fields

- S8BIT **offset**
- E_STB_DP_ANALOG_VIDEO_TYPE **vtype**

The documentation for this struct was generated from the following file:

- [dvb/inc/ap_cntrl.h](#)

3.64 **S_MANUAL_CABLE_TUNING_PARAMS** Struct Reference

Data Fields

- E_STB_DP_CMODE **mode**
- U16BIT **symbol_rate**

The documentation for this struct was generated from the following file:

- [dvb/inc/ap_cntrl.h](#)

3.65 S_MANUAL_SAT_TUNING_PARAMS Struct Reference

Data Fields

- void * **satellite**
- E_STB_DP_POLARITY **polarity**
- U16BIT **symbol_rate**
- E_STB_DP_FEC **fec**
- BOOLEAN **dvb_s2**
- E_STB_DP_MODULATION **modulation**

The documentation for this struct was generated from the following file:

- [dvb/inc/ap_cntrl.h](#)

3.66 S_MANUAL_TERR_TUNING_PARAMS Struct Reference

Data Fields

- E_STB_DP_TBWIDTH **bwidth**
- E_STB_DP_TMODE **mode**
- E_STB_DP_TTYPE **type**
- U8BIT **plp_id**

The documentation for this struct was generated from the following file:

- [dvb/inc/ap_cntrl.h](#)

3.67 S_MANUAL_TUNING_PARAMS Struct Reference

Data Fields

- U32BIT **freq**
- union {
 - [S_MANUAL_ANA_TUNING_PARAMS](#) **ana**
 - [S_MANUAL_TERR_TUNING_PARAMS](#) **terr**
 - [S_MANUAL_CABLE_TUNING_PARAMS](#) **cab**
 - [S_MANUAL_SAT_TUNING_PARAMS](#) **sat**
- **u**

The documentation for this struct was generated from the following file:

- [dvb/inc/ap_cntrl.h](#)

3.68 S_NW_ACCESS_POINT Struct Reference

Data Fields

- U8BIT * **essid**
- U8BIT **quality**
- BOOLEAN **encrypted**
- BOOLEAN **connected**

The documentation for this struct was generated from the following file:

- platform/inc/[stbhwnet.h](#)

3.69 S_NW_ADDR_INFO Struct Reference

Data Fields

- E_NW_AF **af**
- E_NW_TYPE **type**
- E_NW_PROTOCOL **protocol**
- U8BIT **addr** [48]

The documentation for this struct was generated from the following file:

- platform/inc/[stbhwnet.h](#)

3.70 S_NW_SOCKET Struct Reference

Data Fields

- U16BIT **sock_count**
- void * **sock_array** [SOCK_SETSIZE]

The documentation for this struct was generated from the following file:

- platform/inc/[stbhwnet.h](#)

3.71 S_OTA_CAB_TUNING_PARAMS Struct Reference

Data Fields

- E_STB_DP_CMODE **modulation**
- U32BIT **srate**

The documentation for this struct was generated from the following file:

- middleware/ota/inc/[stbota.h](#)

3.72 S_OTA_SAT_TUNING_PARAMS Struct Reference

Data Fields

- U16BIT **longitude**
- BOOLEAN **direction_east**
- E_STB_DP_POLARITY **polarity**
- U32BIT **srate**
- E_STB_DP_MODULATION **modulation**
- E_STB_DP_FEC **fec**

The documentation for this struct was generated from the following file:

- [middleware/ota/inc/stbota.h](#)

3.73 S_OTA_TER_TUNING_PARAMS Struct Reference

Data Fields

- E_STB_DP_TBWIDTH **bwidth**
- E_STB_DP_TTYPE **type**
- U8BIT **plp_id**

The documentation for this struct was generated from the following file:

- [middleware/ota/inc/stbota.h](#)

3.74 S_OTA_TUNING_PARAMS Struct Reference

Data Fields

- E_STB_DP_SIGNAL_TYPE **signal_type**
- U32BIT **freq**
- union {
 - [S_OTA_SAT_TUNING_PARAMS](#) **sat**
 - [S_OTA_TER_TUNING_PARAMS](#) **ter**
 - [S_OTA_CAB_TUNING_PARAMS](#) **cab**
- **u**

The documentation for this struct was generated from the following file:

- [middleware/ota/inc/stbota.h](#)

3.75 s_pvr_pid_info Struct Reference

Data Fields

- E_PVR_PID_TYPE **type**
- U16BIT **pid**
- union {
 - E_STB_AV_VIDEO_CODEC **video_codec**
 - E_STB_AV_AUDIO_CODEC **audio_codec**
- } **u**

The documentation for this struct was generated from the following file:

- platform/inc/[stbpvrpr.h](#)

3.76 S_PVR_RECORD_INFO Struct Reference

Data Fields

- U32DHMS **duration**
- BOOLEAN **event_triggered**
- U16BIT **event_id**
- U16BIT **service_id**
- U16BIT **transport_id**
- U16BIT **orig_net_id**
- U8BIT **prog_crid** [TMR_PVR_CRID_LEN_MAX]
- U8BIT **other_crid** [TMR_PVR_CRID_LEN_MAX]
- U16BIT **disk_id**
- BOOLEAN **recommendation**
- U16BIT **notify_time**

The documentation for this struct was generated from the following file:

- dvb/inc/[ap_tmr.h](#)

3.77 S_RECTANGLE Struct Reference

Data Fields

- S32BIT **left**
- S32BIT **top**
- U32BIT **width**
- U32BIT **height**

The documentation for this struct was generated from the following file:

- platform/inc/[osdtype.h](#)

3.78 S_STB_AV_COPY_PROTECTION Struct Reference

Data Fields

- BOOLEAN **macrovision_set**
- U8BIT **macrovision**
- BOOLEAN **aps_set**
- U8BIT **aps**
- BOOLEAN **cgms_a_set**
- U8BIT **cgms_a**
- BOOLEAN **ict_set**
- U8BIT **ict**
- BOOLEAN **hdcp_set**
- BOOLEAN **hdcp**
- BOOLEAN **scms_set**
- U8BIT **scms**
- BOOLEAN **dot_set**
- U8BIT **dot**

The documentation for this struct was generated from the following file:

- platform/inc/[stbhwav.h](#)

3.79 S_STB_AV_VIDEO_INFO Struct Reference

Data Fields

- E_STB_AV_VIDEO_INFO_TYPE **flags**
- U16BIT **video_width**
- U16BIT **video_height**
- U16BIT **screen_width**
- U16BIT **screen_height**
- E_STB_AV_ASPECT_RATIO **video_aspect_ratio**
- E_STB_AV_ASPECT_RATIO **display_aspect_ratio**
- U8BIT **afd**
- E_STB_AV_DECODER_STATUS **status**

The documentation for this struct was generated from the following file:

- platform/inc/[stbhwav.h](#)

3.80 S_STB_MP_COMPONENT_DETAILS Struct Reference

Data Fields

- U8BIT **component_tag**
 - U16BIT **pid**
 - E_STB_MP_COMPONENT_TYPE **type**
 - union {
 - struct {
 - E_STB_AV_AUDIO_CODEC **encoding**
 - U32BIT **lang_code**
 - U8BIT **num_channels**
 - BOOLEAN **audio_description**
 - audio**
 - struct {
 - E_STB_AV_VIDEO_CODEC **encoding**
 - BOOLEAN **hd**
 - U8BIT **frame_rate**
 - E_STB_AV_ASPECT_RATIO **aspect_ratio**
 - video**
 - struct {
 - U32BIT **lang_code**
 - BOOLEAN **hearing_impaired**
 - subtitle**
 - av**
- BOOLEAN **encrypted**
- BOOLEAN **active**

The documentation for this struct was generated from the following file:

- platform/inc/[stbhwmediaplayer.h](#)

3.81 S_STB_MP_START_PARAMS Struct Reference

Data Fields

- BOOLEAN **cache**
- S32BIT **loops**

The documentation for this struct was generated from the following file:

- platform/inc/[stbhwmediaplayer.h](#)

3.82 s_timer Struct Reference

Data Fields

- E_TIMER_TYPE **type**
- U8BIT **name** [TMR_MAX_NAME_LENGTH]
- E_TIMER_FREQ **frequency**
- U32DHMS **start_time**
- union {
 - [S_ALARM_INFO](#) **alarm**
 - [S_PVR_RECORD_INFO](#) **record**
- **u**

The documentation for this struct was generated from the following file:

- [dvb/inc/ap_tmr.h](#)

3.83 s_vt_options Struct Reference

Data Fields

- BOOLEAN **mheg_required**
- BOOLEAN **afd_required**
- BOOLEAN **hbbtv_required**

The documentation for this struct was generated from the following file:

- [middleware/stb/src/vtc.h](#)

3.84 si_aac_desc Struct Reference

Data Fields

- U8BIT **profile_level**
- BOOLEAN **type_present**
- U8BIT **aac_type**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.85 si_ac3_desc Struct Reference

Data Fields

- U8BIT **dtag**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.86 si_ad_desc Struct Reference

Data Fields

- U32BIT **lang_code**
- BOOLEAN **mix_type**
- U8BIT **edit_class**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.87 si_app_sig_desc Struct Reference

Data Fields

- U16BIT **app_type**
- U8BIT **ait_version**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.88 si_bat_freesat_group_name_entry Struct Reference

Data Fields

- U8BIT **group_type**
- U16BIT **group_id**
- [SI_MULTILANG_GROUP_NAME_DESC](#) * **string_array**
- U8BIT **num_group_names**
- struct [si_bat_freesat_group_name_entry](#) * **next_group**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.89 si_bat_freesat_iactive_storage_desc Struct Reference

Data Fields

- U8BIT * **data**
- U8BIT **length**
- struct [si_bat_freesat_iactive_storage_desc](#) * **next_desc**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.90 si_bat_freesat_region Struct Reference

Data Fields

- U16BIT **region_id**
- U32BIT **lang_code**
- [SI_STRING_DESC](#) * **region_name**
- struct [si_bat_freesat_region](#) * **next**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.91 si_bat_freesat_region_lcn_entry Struct Reference

Data Fields

- U16BIT **tran_id**
- U16BIT **orig_net_id**
- U16BIT **service_id**
- U16BIT **freesat_id**
- struct [si_bat_freesat_region_lcn_entry](#) * **next**
- [SI_FREESAT_LCN](#) * **freesat_lcn_list**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.92 si_bat_freesat_serv_group_entry Struct Reference

Data Fields

- U16BIT **group_id**
- BOOLEAN **non_destructive_flag**
- BOOLEAN **return_channel_access_flag**
- BOOLEAN **g2_extension_flag**
- U8BIT **g2_flags**
- U8BIT **num_services**
- U16BIT * **freesat_id**
- struct [si_bat_freesat_serv_group_entry](#) * **next_group**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.93 si_bat_table Struct Reference

Data Fields

- U8BIT **version**
- U16BIT **bouquet_id**
- [SI_STRING_DESC](#) * **bouquet_name**
- [SI_STRING_DESC](#) * **def_authority**
- [SI_BAT_FREESAT_REGION](#) * **region_list**
- U16BIT **num_linkage_entries**
- [SI_LINKAGE_DESC_ENTRY](#) * **linkage_desc_list**
- [SI_LINKAGE_DESC_ENTRY](#) * **last_linkage_entry**
- U16BIT **num_transports**
- [SI_BAT_TRANSPORT_ENTRY](#) * **transport_list**
- [SI_BAT_TRANSPORT_ENTRY](#) * **last_transport_entry**
- [SI_BAT_FREESAT_GROUP_NAME_ENTRY](#) * **group_name_list**
- U16BIT **num_serv_groups**
- [SI_BAT_FREESAT_SERV_GROUP_ENTRY](#) * **serv_group_array**
- [SI_FTA_CONTENT_DESC](#) * **fta_content_desc**
- [SI_BAT_FREESAT_IACTIVE_STORAGE_DESC](#) * **iactive_storage_desc_list**
- [SI_BAT_FREESAT_INFO_LOCATION](#) * **info_location_list**
- [SI_FREESAT_PREFIX_DESC](#) * **freesat_prefix_list**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.94 si_bat_transport_entry Struct Reference

Data Fields

- struct [si_bat_transport_entry](#) * **next**
- U16BIT **tran_id**
- U16BIT **orig_net_id**
- [SI_SERV_LIST_DESC](#) * **serv_list_desc_array**
- U16BIT **num_serv_list_entries**
- U16BIT **num_lcn_entries**
- [SI_LCN_DESC](#) * **lcn_desc_array**
- [SI_BAT_FREESAT_REGION_LCN_ENTRY](#) * **lcn_region_list**
- [SI_STRING_DESC](#) * **def_authority**
- [SI_FTA_CONTENT_DESC](#) * **fta_content_desc**
- U16BIT * **int_rest_serv_array**
- U8BIT **num_int_rest_serv**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.95 si_ca_desc Struct Reference

Data Fields

- U16BIT **ca_id**
- U16BIT **ca_pid**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.96 si_cable_del_sys_desc Struct Reference

Data Fields

- U32BIT **freq_hz**
- U8BIT **fec_outer**
- U8BIT **fec_inner**
- U8BIT **modulation**
- U32BIT **symbol_rate**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.97 si_ciplus_service Struct Reference

Data Fields

- struct [si_ciplus_service](#) * **next**
- U16BIT **id**
- U8BIT **type**
- BOOLEAN **visible**
- BOOLEAN **selectable**
- U16BIT **lcn**
- [SI_STRING_DESC](#) * **provider_str**
- [SI_STRING_DESC](#) * **name_str**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.98 `si_component_desc` Struct Reference

Data Fields

- U8BIT **tag**
- U8BIT **content**
- U8BIT **type**
- U32BIT **lang_code**
- [SI_STRING_DESC](#) * **desc_str**

The documentation for this struct was generated from the following file:

- `middleware/stb/inc/stbsitab.h`

3.99 `si_content_desc` Struct Reference

Data Fields

- U8BIT **level_1**
- U8BIT **level_2**
- U8BIT **user_1**
- U8BIT **user_2**

The documentation for this struct was generated from the following file:

- `middleware/stb/inc/stbsitab.h`

3.100 `si_crid_desc` Struct Reference

Data Fields

- struct [si_crid_desc](#) * **next**
- U8BIT **type**
- [SI_STRING_DESC](#) * **crid_str**

The documentation for this struct was generated from the following file:

- `middleware/stb/inc/stbsitab.h`

3.101 `si_delivery_sys_desc` Union Reference

Data Fields

- [SI_SAT_DEL_SYS_DESC](#) **sat**
- [SI_CABLE_DEL_SYS_DESC](#) **cable**

- [SI_TERR_DEL_SYS](#) terr

The documentation for this union was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.102 si_dvb_subt_desc Struct Reference

Data Fields

- U32BIT **lang_code**
- U8BIT **type**
- U16BIT **composition_page**
- U16BIT **ancillary_page**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.103 si_eit_event_entry Struct Reference

Data Fields

- struct [si_eit_event_entry](#) * **next**
- U8BIT **sect_num**
- U16BIT **event_id**
- U16BIT **start_date**
- U8BIT **start_hrs**
- U8BIT **start_mins**
- U8BIT **start_secs**
- U8BIT **duration_hrs**
- U8BIT **duration_mins**
- U8BIT **duration_secs**
- U8BIT **running_status**
- BOOLEAN **all_streams_free**
- U8BIT **preferred_name_id**
- U8BIT **num_ca_id_entries**
- U16BIT * **ca_id_desc_array**
- U8BIT **num_component_entries**
- [SI_COMPONENT_DESC](#) * **component_desc_array**
- U8BIT **num_content_entries**
- [SI_CONTENT_DESC](#) * **content_desc_array**
- U8BIT **num_multiling_component_entries**
- [SI_MULTILING_COMPONENT_DESC](#) * **multiling_component_desc_array**
- U8BIT **num_parental_rating_entries**

- [SI_PARENTAL_RATING_DESC](#) * `parental_rating_desc_array`
- U8BIT `num_short_event_entries`
- [SI_SHORT_EVENT_DESC](#) * `short_event_desc_array`
- U8BIT `num_extended_event_entries`
- [SI_EXTENDED_EVENT_DESC](#) * `extended_event_desc_array`
- U8BIT `num_crids`
- [SI_CRID_DESC](#) * `crid_list`
- [SI_CRID_DESC](#) * `last_crid_entry`
- [SI_GUIDANCE_DESC](#) * `guidance`
- [SI_FTA_CONTENT_DESC](#) * `fta_content_desc`

The documentation for this struct was generated from the following file:

- `middleware/stb/inc/stbsitab.h`

3.104 si_eit_table Struct Reference

Data Fields

- U8BIT `version`
- U8BIT `table_id`
- U16BIT `serv_id`
- U16BIT `tran_id`
- U16BIT `orig_net_id`
- U8BIT `last_table_id`
- U16BIT `num_events`
- [SI_EIT_EVENT_ENTRY](#) * `event_list`
- [SI_EIT_EVENT_ENTRY](#) * `last_event_entry`

The documentation for this struct was generated from the following file:

- `middleware/stb/inc/stbsitab.h`

3.105 si_extended_event_desc Struct Reference

Data Fields

- U8BIT `desc_number`
- U8BIT `last_desc_number`
- U32BIT `lang_code`
- U8BIT `num_items`
- [SI_STRING_DESC](#) ** `item_desc_array`
- [SI_STRING_DESC](#) ** `item_text_array`
- [SI_STRING_DESC](#) * `text_str`

The documentation for this struct was generated from the following file:

- `middleware/stb/inc/stbsitab.h`

3.106 si_freesat_info_location Struct Reference

Data Fields

- U16BIT **orig_net_id**
- U16BIT **transport_id**
- U16BIT **service_id**
- struct [si_freesat_info_location](#) * **next**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.107 si_freesat_lcn Struct Reference

Data Fields

- BOOLEAN **numeric_selection**
- BOOLEAN **visible_flag**
- BOOLEAN **user_cust**
- U16BIT **lcn**
- U16BIT **region_id**
- struct [si_freesat_lcn](#) * **next**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.108 si_freesat_linkage_desc Struct Reference

Data Fields

- U16BIT **onet_id**
- U16BIT **trans_id**
- U16BIT **serv_id**
- U8BIT **num_data_types**
- U8BIT * **data_types**
- struct [si_freesat_linkage_desc](#) * **next**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.109 si_freesat_prefix_desc Struct Reference

Data Fields

- U8BIT **prefix_index**
- SI_STRING_DESC * **uri_prefix**
- struct [si_freesat_prefix_desc](#) * **next**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.110 si_freesat_tunelled_data_desc Struct Reference

Data Fields

- U8BIT **data_type**
- U8BIT **component_tag**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.111 si_fta_content_desc Struct Reference

Data Fields

- BOOLEAN **do_not_scramble**
- U8BIT **access_over_internet**
- BOOLEAN **do_not_apply_revocation**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.112 si_guidance_desc Struct Reference

Data Fields

- U8BIT **guidance_type**
- BOOLEAN **guidance_mode**
- U8BIT **num_langs**
- U32BIT * **lang_codes**
- SI_STRING_DESC ** **strings**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.113 si_image_icon_desc Struct Reference

Data Fields

- U8BIT **desc_num**
- U8BIT **last_desc_num**
- U8BIT **icon_id**
- E_ICON_TRANSPORT_MODE **transport_mode**
- BOOLEAN **position_defined**
- E_ICON_COORD_SYSTEM **coord_system**
- U16BIT **x_pos**
- U16BIT **y_pos**
- U8BIT * **icon_type**
- U32BIT **data_len**
- U8BIT * **icon_data**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.114 si_iso_lang_desc Struct Reference

Data Fields

- U32BIT **lang_code**
- U8BIT **audio_type**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.115 si_linkage_desc Struct Reference

Data Fields

- struct [si_linkage_desc](#) * **next**
- U16BIT **orig_net_id**
- U16BIT **tran_id**
- U16BIT **serv_id**
- U8BIT **link_type**
- U8BIT **data_length**
- U8BIT **data**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.116 si_lto_desc Struct Reference

Data Fields

- U32BIT **country_code**
- U8BIT **region**
- BOOLEAN **offset_negative**
- U8BIT **offset_hrs**
- U8BIT **offset_mins**
- U16BIT **change_date**
- U8BIT **change_hrs**
- U8BIT **change_mins**
- U8BIT **change_secs**
- U8BIT **next_offset_hrs**
- U8BIT **next_offset_mins**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.117 si_multilang_group_name_desc Struct Reference

Data Fields

- U32BIT **lang_code**
- [SI_STRING_DESC](#) * **name_str**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.118 si_multiling_component_desc Struct Reference

Data Fields

- U8BIT **tag**
- U32BIT **lang_code**
- [SI_STRING_DESC](#) * **desc_str**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.119 si_multiling_net_name_desc Struct Reference

Data Fields

- U32BIT **lang_code**
- [SI_STRING_DESC](#) * **name_str**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.120 si_multiling_serv_name_desc Struct Reference

Data Fields

- U32BIT **lang_code**
- [SI_STRING_DESC](#) * **name_str**
- [SI_STRING_DESC](#) * **provider_str**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.121 SI_MULTILING_SHORT_NAME_DESC Struct Reference

Data Fields

- U32BIT **lang_code**
- [SI_STRING_DESC](#) * **name_str**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.122 si_nit_change_entry Struct Reference

Data Fields

- U8BIT **change_id**
- U8BIT **version**
- U16BIT **start_date**
- U8BIT **start_hours**
- U8BIT **start_mins**
- U8BIT **start_secs**
- U8BIT **dur_hours**

- U8BIT **dur_mins**
- U8BIT **dur_secs**
- SI_NIT_RECEIVER_CATEGORY **receiver_category**
- SI_NIT_NETWORK_CHANGE_TYPE **change_type**
- U8BIT **message_id**
- BOOLEAN **invariant_ts_present**
- U16BIT **invariant_ts_tsid**
- U16BIT **invariant_ts_onid**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.123 si_nit_change_notify_desc Struct Reference

Data Fields

- U16BIT **cell_id**
- U8BIT **num_changes**
- [SI_NIT_CHANGE_ENTRY](#) * **change_array**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.124 si_nit_message_entry Struct Reference

Data Fields

- U8BIT **message_id**
- U32BIT **lang_code**
- [SI_STRING_DESC](#) * **message**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.125 si_nit_region_name Struct Reference

Data Fields

- U8BIT **region_depth**
- [SI_STRING_DESC](#) * **region_name**
- U8BIT **primary_region_code**
- U8BIT **secondary_region_code**

- U16BIT **tertiary_region_code**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.126 si_nit_table Struct Reference

Data Fields

- U8BIT **version**
- U16BIT **net_id**
- [SI_STRING_DESC](#) * **name_str**
- U16BIT **num_multiling_net_names**
- [SI_MULTILING_NET_NAME_DESC](#) * **multiling_net_name_desc_array**
- U16BIT **num_linkage_entries**
- [SI_LINKAGE_DESC_ENTRY](#) * **linkage_desc_list**
- [SI_LINKAGE_DESC_ENTRY](#) * **last_linkage_entry**
- [SI_STRING_DESC](#) * **def_authority**
- U16BIT **num_transports**
- [SI_NIT_TRANSPORT_ENTRY](#) * **transport_list**
- [SI_NIT_TRANSPORT_ENTRY](#) * **last_transport_entry**
- U16BIT **num_change_notifies**
- [SI_NIT_CHANGE_NOTIFY_DESC](#) * **change_notify_array**
- U16BIT **num_messages**
- [SI_NIT_MESSAGE_ENTRY](#) * **message_array**
- [SI_FTA_CONTENT_DESC](#) * **fta_content_desc**
- [SI_NIT_TARGET_REGION_NAME_DESC](#) * **target_region_name_list**
- [SI_TARGET_REGION_DESC](#) * **target_region_list**
- [SI_FREESAT_LINKAGE_DESC](#) * **freesat_linkage_desc**
- [SI_FREESAT_PREFIX_DESC](#) * **freesat_prefix_list**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.127 si_nit_target_region_name_desc Struct Reference

Data Fields

- struct [si_nit_target_region_name_desc](#) * **next**
- U32BIT **country_code**
- U32BIT **lang_code**
- U8BIT **num_names**
- [SI_NIT_REGION_NAME](#) * **name_array**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.128 si_nit_transport_entry Struct Reference

Data Fields

- struct [si_nit_transport_entry](#) * next
- U16BIT tran_id
- U16BIT orig_net_id
- SI_DELIVERY_SYS_DESC_TYPE del_sys_desc_type
- [SI_DELIVERY_SYS_DESC](#) * del_sys_desc
- U16BIT num_freq_list_entries
- U32BIT * freq_list_desc_array
- U16BIT num_serv_list_entries
- [SI_SERV_LIST_DESC](#) * serv_list_desc_array
- U16BIT num_lcn_entries
- [SI_LCN_DESC](#) * lcn_desc_array
- [SI_STRING_DESC](#) * def_authority
- U16BIT num_serv_attr_entries
- [SI_SERV_ATTRIBUTE_DESC](#) * serv_attr_array
- U16BIT num_nordig_lcn_entries
- [SI_NORDIG_LCN_DESC](#) * nordig_lcn_desc_array
- U16BIT num_hd_lcn_entries
- [SI_LCN_DESC](#) * hd_lcn_desc_array
- [SI_FTA_CONTENT_DESC](#) * fta_content_desc
- [SI_TARGET_REGION_DESC](#) * target_region_list
- U16BIT num_ciplus_services
- [SI_CIPLUS_SERVICE](#) * ciplus_service_list
- [SI_CIPLUS_SERVICE](#) * last_ciplus_service

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.129 SI_NORDIG_LCN_DESC Struct Reference

Data Fields

- U8BIT chan_list_id
- [SI_STRING_DESC](#) * chan_list_name
- U32BIT country_code
- U8BIT num_services
- [SI_NORDIG_SERV_LCN](#) * serv_array

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.130 SI_NORDIG_SERV_LCN Struct Reference

Data Fields

- U16BIT **serv_id**
- U16BIT **serv_lcn**
- BOOLEAN **visible**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.131 si_parental_rating_desc Struct Reference

Data Fields

- U32BIT **country_code**
- U8BIT **rating**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.132 si_pat_service_entry Struct Reference

Data Fields

- struct [si_pat_service_entry](#) * **next**
- U16BIT **serv_id**
- U16BIT **pmt_pid**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.133 si_pat_table Struct Reference

Data Fields

- U8BIT **version**
- U16BIT **tran_id**
- U16BIT **num_services**
- [SI_PAT_SERVICE_ENTRY](#) * **service_list**
- [SI_PAT_SERVICE_ENTRY](#) * **last_service_entry**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.134 si_pmt_stream_entry Struct Reference

Data Fields

- struct [si_pmt_stream_entry](#) * next
- SI_STREAM_TYPE type
- U16BIT pid
- U8BIT * tag_array_ptr
- U8BIT num_tag_entries
- U16BIT num_iso_lang_entries
- [SI_ISO_LANG_DESC](#) * iso_lang_desc_array
- U16BIT num_ca_entries
- [SI_CA_DESC](#) * ca_desc_array
- U16BIT num_subtitle_entries
- [SI_SUBTITLE_DESC](#) * subtitle_desc_array
- U16BIT num_teletext_entries
- [SI_TELETEXT_DESC](#) * teletext_desc_array
- [SI_SERVICE_MOVE_DESC](#) * service_move
- U16BIT num_app_sig_entries
- [SI_APP_SIG_DESC](#) * app_sig_desc_array
- BOOLEAN has_ait
- BOOLEAN has_rct
- U8BIT num_tunnelled_entries
- [SI_FREESAT_TUNNELLED_DATA_DESC](#) * tunnelled_desc_array
- U32BIT carousel_id
- [SI_AC3_DESC](#) * ac3_descriptor
- [SI_AD_DESC](#) * audio_desc
- [SI_AAC_DESC](#) * aac_descriptor

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.135 si_pmt_table Struct Reference

Data Fields

- U8BIT version
- U16BIT serv_id
- U16BIT pcr_pid
- U16BIT num_ca_entries
- [SI_CA_DESC](#) * ca_desc_array
- U8BIT num_tunnelled_entries
- [SI_FREESAT_TUNNELLED_DATA_DESC](#) * tunnelled_desc_array
- U16BIT num_streams
- [SI_PMT_STREAM_ENTRY](#) * stream_list

- [SI_PMT_STREAM_ENTRY](#) * `last_stream_entry`

The documentation for this struct was generated from the following file:

- `middleware/stb/inc/stbsitab.h`

3.136 `si_preferred_name_desc` Struct Reference

Data Fields

- U32BIT `lang_code`
- U8BIT `name_id`
- [SI_STRING_DESC](#) * `name_str`

The documentation for this struct was generated from the following file:

- `middleware/stb/inc/stbsitab.h`

3.137 `si_rct_link_info` Struct Reference

Data Fields

- E_RCT_LINK_TYPE `link_type`
- E_RCT_HOW_RELATED `how_related`
- U16BIT `term_id`
- U8BIT `group_id`
- U8BIT `precedence`
- U8BIT * `uri_string`
- U8BIT `num_items`
- [SI_RCT_PROMO_TEXT](#) * `promo_text_array`
- BOOLEAN `can_use_default_icon`
- U8BIT `icon_id`
- [SI_SHORT_EVENT_DESC](#) * `event_desc`
- U8BIT `num_icons`
- [SI_IMAGE_ICON_DESC](#) * `icon_array`

The documentation for this struct was generated from the following file:

- `middleware/stb/inc/stbsitab.h`

3.138 si_rct_promo_text Struct Reference

Data Fields

- U32BIT **lang_code**
- [SI_STRING_DESC](#) * **string**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.139 si_rct_subtable Struct Reference

Data Fields

- U8BIT **version**
- U16BIT **service_id**
- [SI_RCT_SUBTABLE_DATA](#) * **data**
- struct [si_rct_subtable](#) * **next**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.140 si_rct_subtable_data Struct Reference

Data Fields

- U16BIT **year_offset**
- U8BIT **link_count**
- [SI_RCT_LINK_INFO](#) * **link_array**
- U8BIT **num_icons**
- [SI_IMAGE_ICON_DESC](#) * **icon_array**
- struct [si_rct_subtable_data](#) * **next**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.141 si_rct_table Struct Reference

Data Fields

- [SI_RCT_SUBTABLE](#) * **subtables**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.142 si_sat_del_sys_desc Struct Reference

Data Fields

- U32BIT **freq_hz**
- U16BIT **position**
- U16BIT **sym_rate**
- E_STB_DP_POLARITY **polarity**
- BOOLEAN **east_west**
- E_STB_DP_FEC **fec_code**
- BOOLEAN **dvb_s2**
- E_STB_DP_MODULATION **modulation**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.143 si_sdt_service_entry Struct Reference

Data Fields

- struct [si_sdt_service_entry](#) * **next**
- U16BIT **serv_id**
- BOOLEAN **eit_sched_avail**
- BOOLEAN **eit_now_next_avail**
- U8BIT **running_status**
- BOOLEAN **all_streams_free**
- U8BIT **serv_type**
- [SI_STRING_DESC](#) * **name_str**
- [SI_STRING_DESC](#) * **provider_str**
- U8BIT **num_ca_id_entries**
- U16BIT * **ca_id_desc_array**
- U16BIT **num_multiling_names**
- [SI_MULTILING_SERV_NAME_DESC](#) * **multiling_name_desc_array**
- U16BIT **num_preferred_names**
- [SI_PREFERRED_NAME_DESC](#) * **preferred_name_desc_array**
- U16BIT **num_linkage_entries**
- [SI_LINKAGE_DESC_ENTRY](#) * **linkage_desc_list**
- [SI_LINKAGE_DESC_ENTRY](#) * **last_linkage_entry**
- [SI_STRING_DESC](#) * **def_authority**
- [SI_STRING_DESC](#) * **short_name_str**
- U16BIT **num_multiling_short_names**
- [SI_MULTILING_SHORT_NAME_DESC](#) * **multiling_short_name_array**
- [SI_GUIDANCE_DESC](#) * **guidance**
- [SI_FTA_CONTENT_DESC](#) * **fta_content_desc**
- [SI_TARGET_REGION_DESC](#) * **target_region_list**

- [SI_SERV_AVAIL_DESC](#) * **serv_avail_desc_list**
- U8BIT * **ci_protection_desc**
- [SI_FREESAT_PREFIX_DESC](#) * **freesat_prefix_list**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.144 **si_sdt_table** Struct Reference

Data Fields

- U8BIT **version**
- U16BIT **tran_id**
- U16BIT **orig_net_id**
- U16BIT **num_services**
- [SI_SDT_SERVICE_ENTRY](#) * **service_list**
- [SI_SDT_SERVICE_ENTRY](#) * **last_service_entry**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.145 **si_section_record** Struct Reference

Data Fields

- struct [si_section_record](#) * **next**
- U8BIT **sect_num**
- U16BIT **data_len**
- U8BIT **data_start**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsiflt.h](#)

3.146 **si_serv_attribute_desc** Struct Reference

Data Fields

- U16BIT **serv_id**
- BOOLEAN **service_selectable**
- BOOLEAN **service_visible**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.147 **si_serv_avail_desc** Struct Reference

Data Fields

- struct [si_serv_avail_desc](#) * **next**
- U16BIT **num_of_cell_ids**
- U16BIT * **cell_ids**
- BOOLEAN **availability_flag**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.148 **si_serv_list_desc** Struct Reference

Data Fields

- U16BIT **serv_id**
- U8BIT **serv_type**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.149 **si_service_move_desc** Struct Reference

Data Fields

- U16BIT **onet_id**
- U16BIT **ts_id**
- U16BIT **service_id**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.150 **si_short_event_desc** Struct Reference

Data Fields

- U32BIT **lang_code**
- [SI_STRING_DESC](#) * **name_str**
- [SI_STRING_DESC](#) * **desc_str**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.151 si_string Struct Reference

Data Fields

- U8BIT **nbytes**
- U8BIT * **str_ptr**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.152 si_t2_del_sys_cell Struct Reference

Data Fields

- U16BIT **cell_id**
- U8BIT **num_freqs**
- U32BIT **freq_hz** [6]

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.153 si_t2_del_sys_desc Struct Reference

Data Fields

- U8BIT **plp_id**
- U16BIT **t2_system_id**
- E_STB_DP_TBWIDTH **bwidth**
- E_STB_DP_TMODE **mode**
- U8BIT **num_cells**
- [SI_T2_DEL_SYS_CELL](#) * **cell**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.154 si_table_record Struct Reference

Data Fields

- U8BIT **path**
- U8BIT **tid**
- U16BIT **xtid**

- U8BIT **version**
- U16BIT **num_sect**
- [SI_SECTION_RECORD](#) * **section_list**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsifft.h](#)

3.155 `si_target_region` Struct Reference

Data Fields

- struct [si_target_region](#) * **next**
- U8BIT **region_depth**
- U8BIT **primary_region_code**
- U8BIT **secondary_region_code**
- U8BIT **tertiary_region_code**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.156 `si_target_region_desc` Struct Reference

Data Fields

- struct [si_target_region_desc](#) * **next**
- U32BIT **country_code**
- [SI_TARGET_REGION](#) * **region_list**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.157 `si_teletext_desc` Struct Reference

Data Fields

- U32BIT **lang_code**
- U8BIT **type**
- U8BIT **magazine**
- U8BIT **page**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.158 SI_TERR_DEL_SYS Struct Reference

Data Fields

- **BOOLEAN** **is_t2**
- **union** {
 - [SI_TERR_DEL_SYS_DESC](#) **t1**
 - [SI_T2_DEL_SYS_DESC](#) **t2**
- **u**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.159 si_terr_del_sys_desc Struct Reference

Data Fields

- **U32BIT** **freq_hz**
- **E_STB_DP_TBWIDTH** **bwidth**
- **E_STB_DP_TMODE** **mode**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.160 si_time_table Struct Reference

Data Fields

- **U16BIT** **date**
- **U8BIT** **hrs**
- **U8BIT** **mins**
- **U8BIT** **secs**
- **U16BIT** **num_lto_entries**
- [SI_LTO_DESC](#) * **lto_desc_array**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

3.161 T_BOOT_SW_VER_NUMBER Struct Reference

Data Fields

- U8BIT **major**
- U8BIT **minor**

The documentation for this struct was generated from the following file:

- [platform/inc/stbver.h](#)

3.162 ter_rf_channel_data Struct Reference

Data Fields

- U8BIT * **name**
- U32BIT **freq_hz**
- E_STB_DP_TBWIDTH **bwidth**
- E_STB_DP_TMODE **mode**
- E_STB_DP_TTYPE **type**

The documentation for this struct was generated from the following file:

- [dvb/inc/ap_cfg.h](#)

3.163 uk_dtt_lcn_desc Struct Reference

Data Fields

- U16BIT **serv_id**
- U16BIT **serv_lcn**
- BOOLEAN **visible**

The documentation for this struct was generated from the following file:

- [middleware/stb/inc/stbsitab.h](#)

4 File Documentation

4.1 database/dba_nvm/src/dba_nvm.h File Reference

NVM database access defines, structures and functions.

Enumerations

- enum **E_DBA_FIELD_TYPE** { **UNUM_TYPE** = 0, **STR_TYPE** = 1, **PTR_TYPE** = 2 }

4.1.1 Detailed Description

NVM database access defines, structures and functions.

Date

13/11/2013

Author

Ocean Blue

4.2 database/dba_nvm/src/stbcsun.h File Reference

Header file - Function prototypes for check sum calcs.

Functions

- U8BIT **STB_CalcChecksum** (U8BIT *data_ptr, U32BIT data_size)
- BOOLEAN **STB_CheckChecksum** (U8BIT *data_ptr, U32BIT data_size)
- U16BIT **STB_GetBE16Bit** (U16BIT *addr)
- void **STB_SetBE16Bit** (U16BIT *addr, U16BIT value)

4.2.1 Detailed Description

Header file - Function prototypes for check sum calcs.

Date

14/10/2000

4.3 database/dba_nvm/src/stbdbnvm.h File Reference

Header file - Function prototypes for NVM database.

Defines

- #define **NVM_INVALID_BLOCK_ID** 0xffff

Functions

- void **STB_InitNVMAccess** (U16BIT offset)
- void **STB_InitNVMap** (void)
- BOOLEAN **STB_CheckNVMDatabaseIntegrity** (void)
- void **STB_SetNVMAccessRAM** (U8BIT *ram_ptr)
- U8BIT * **STB_GetNVMAccessRAM** (void)
- U16BIT **STB_GetNVMBlockCount** (void)
- U16BIT **STB_GetNVMBlocksUsed** (void)
- U16BIT **STB_GetNVMBlocksNeeded** (U16BIT size)
- U16BIT **STB_CreateNVMSRecord** (U8BIT rec_id, U16BIT size)
- void **STB_DestroyNVMSRecord** (U16BIT block_no)
- void **STB_SetNextNVMSBlock** (U16BIT block_no, U16BIT next_block)
- U16BIT **STB_GetNextNVMSBlock** (U16BIT block_no)
- U16BIT **STB_FindNVMSRecordFromId** (U8BIT rec_id, U16BIT last_blk)
- void **STB_NVMSChanged** (BOOLEAN state)
- void **STB_NVMSave** (void)
- void **STB_SetNVMSRecordField** (U16BIT block_no, U16BIT offset, U16BIT size, U32BIT value, E_DBA_FIELD_TYPE type)
- U32BIT **STB_GetNVMSRecordField** (U16BIT block_no, U16BIT offset, U16BIT size, E_DBA_FIELD_TYPE type)
- void **STB_WriteNVMSData** (U16BIT offset, U16BIT size, U8BIT *data_ptr)
- BOOLEAN **STB_ReadNVMSData** (U16BIT offset, U16BIT size, U8BIT *data_ptr)

4.3.1 Detailed Description

Header file - Function prototypes for NVM database.

Date

06/09/2000

4.4 database/dba_nvm/src/stbdbram.h File Reference

Header file - Function prototypes for RAM database.

Functions

- void **STB_InitRAMAccess** (void)
- void **STB_PurgeRAMRecords** (void)
- void * **STB_CreateRAMRecord** (U8BIT rec_id, U16BIT size, U16BIT nvm_block, void *parent)
- void **STB_DestroyRAMRecord** (void *rec_ptr)
- U8BIT **STB_GetRAMRecordId** (void *rec_ptr)
- U16BIT **STB_GetRAMRecordNVMSBlock** (void *rec_ptr)
- U16BIT **STB_GetRAMRecordPrevNVMSBlock** (void *rec_ptr)

- U16BIT **STB_GetRAMRecordNextNVMBlock** (void *rec_ptr)
- void * **STB_GetRAMRecordParent** (void *rec_ptr)
- void **STB_SetRAMRecordParent** (void *rec_ptr, void *parent)
- void **STB_MoveRAMRecordBefore** (void *rec_ptr, void *dst_ptr)
- void **STB_MoveRAMRecordAfter** (void *rec_ptr, void *dst_ptr)
- void * **STB_FindRAMRecordFromId** (U8BIT rec_id, void *parent, void *last_rec)
- void * **STB_FindRAMRecordFromNVMBlock** (U16BIT nvm_block)
- void **STB_SetRAMRecordField** (void *data_ptr, U16BIT offset, U16BIT size, - U32BIT value, E_DBA_FIELD_TYPE type)
- U32BIT **STB_GetRAMRecordField** (void *data_ptr, U16BIT offset, U16BIT size, E_DBA_FIELD_TYPE type)

4.4.1 Detailed Description

Header file - Function prototypes for RAM database.

Date

06/09/2000

4.5 database/dba_nvm/src/stbnvm.h File Reference

Header file - Function prototypes for NVM control.

Functions

- void **STB_NVMInitialise** (void)
- U32BIT **STB_NVMGetDataBlockSize** (U32BIT data_block_id)
Returns the number of bytes of data stored for the given data block.
- BOOLEAN **STB_NVMDataBlockRead** (U32BIT data_block_id, U32BIT bytes, - U8BIT *dest_addr)
Reads data bytes for the given data block from NVM.
- BOOLEAN **STB_NVMDataBlockWrite** (U32BIT data_block_id, U32BIT num_bytes, U8BIT *src_addr)
Writes data bytes for the given data block to NVM.
- U32BIT **STB_NVMGetSTBSize** (void)
- BOOLEAN **STB_NVMSTBRead** (U32BIT offset, U32BIT bytes, U8BIT *dest_addr)
- BOOLEAN **STB_NVMSTBWrite** (U32BIT offset, U32BIT bytes, U8BIT *src_addr)

4.5.1 Detailed Description

Header file - Function prototypes for NVM control.

Date

06/09/2000

4.5.2 Function Documentation

4.5.2.1 **BOOLEAN STB_NVMDataBlockRead** (U32BIT *data_block_id*, U32BIT *bytes*, U8BIT * *dest_addr*)

Reads data bytes for the given data block from NVM.

Parameters

<i>data_block_id</i>	data block from which the data is to be read
<i>num_bytes</i>	number of bytes to be read
<i>dest_addr</i>	buffer to read the data into

Returns

TRUE if the data is read, FALSE otherwise

4.5.2.2 **BOOLEAN STB_NVMDataBlockWrite** (U32BIT *data_block_id*, U32BIT *num_bytes*, U8BIT * *src_addr*)

Writes data bytes for the given data block to NVM.

Parameters

<i>data_block_id</i>	data block to be written
<i>num_bytes</i>	number of bytes to be written
<i>src_addr</i>	data to be written

Returns

TRUE if the data is written, FALSE otherwise

4.5.2.3 **U32BIT STB_NVMGetDataBlockSize** (U32BIT *data_block_id*)

Returns the number of bytes of data stored for the given data block.

Parameters

<i>data_block_id</i>	data block identifier
----------------------	-----------------------

Returns

size in bytes

4.6 database/inc/dba.h File Reference

Database access defines, structures and public functions.

Defines

- #define **DBA_FIELD_PARENT** 0x0100
- #define **DBA_FIELD_REC_NAME** 0x0101
- #define **DBA_FIELD_ORIG_NET_ID** 0x0102
- #define **DBA_FIELD_NET_ID** 0x0103
- #define **DBA_FIELD_TRANSPORT_ID** 0x0104
- #define **DBA_FIELD_SERVICE_ID** 0x0105
- #define **DBA_FIELD_VERSION** 0x0106
- #define **DBA_FIELD_LNB_LOFREQ** 0x0200
- #define **DBA_FIELD_LNB_HIFREQ** 0x0201
- #define **DBA_FIELD_LNB_TYPE** 0x0202
- #define **DBA_FIELD_LNB_22K** 0x0203
- #define **DBA_FIELD_LNB_12V** 0x0204
- #define **DBA_FIELD_LNB_PULSEPOSN** 0x0205
- #define **DBA_FIELD_LNB_DISPOSN** 0x0206
- #define **DBA_FIELD_LNB_DISTONE** 0x0207
- #define **DBA_FIELD_LNB_DISCSWITCH** 0x0208
- #define **DBA_FIELD_LNB_DISUSWITCH** 0x0209
- #define **DBA_FIELD_LNB_DISSMATV** 0x020a
- #define **DBA_FIELD_LNB_DISREPEAT** 0x020b
- #define **DBA_FIELD_LNB_UNICABLEFREQ** 0x020c
- #define **DBA_FIELD_LNB_UNICABLECHAN** 0x020d
- #define **DBA_FIELD_LNB_POWER** 0x020e
- #define **DBA_FIELD_SAT_DISH** 0x0300
- #define **DBA_FIELD_SAT_LONGWE** 0x0301
- #define **DBA_FIELD_SAT_LONGPOS** 0x0302
- #define **DBA_FIELD_PROFILE_TYPE** 0x0401
- #define **DBA_FIELD_PROFILE_CAM_ID** 0x0402
- #define **DBA_FIELD_PROFILE_NAME** 0x0403
- #define **DBA_FIELD_OPERATOR_SEARCH** 0x0404
- #define **DBA_FIELD_OP_SEARCH_DATE** 0x0405
- #define **DBA_FIELD_OP_SEARCH_TIME** 0x0406
- #define **DBA_FIELD_TRAN_FREQ** 0x0500
- #define **DBA_FIELD_TRAN_SRATE** 0x0501
- #define **DBA_FIELD_TRAN_SIGNAL_LEVEL** 0x0502
- #define **DBA_FIELD_STRAN_POL** 0x0600
- #define **DBA_FIELD_STRAN_FEC** 0x0601

- #define **DBA_FIELD_STRAN_DVBS2** 0x0602
- #define **DBA_FIELD_STRAN_MODULATION** 0x0603
- #define **DBA_FIELD_TTRAN_MODE** 0x0700
- #define **DBA_FIELD_TTRAN_TERR_TYPE** 0x0701
- #define **DBA_FIELD_TTRAN_PLP_ID** 0x0702
- #define **DBA_FIELD_TTRAN_BWIDTH** 0x0703
- #define **DBA_FIELD_CTRAN_MODE** 0x0800
- #define **DBA_FIELD_SERV_ID** 0x0900
- #define **DBA_FIELD_SERV_TYPE** 0x0901
- #define **DBA_FIELD_SERV_LCN** 0x0902
- #define **DBA_FIELD_SERV_REQ_LCN** 0x0903
- #define **DBA_FIELD_SERV_HIDDEN** 0x0904
- #define **DBA_FIELD_SERV_SELECTABLE** 0x0905
- #define **DBA_FIELD_SERV_LOCKED** 0x0906
- #define **DBA_FIELD_SERV_SCHED_DISABLED** 0x0907
- #define **DBA_FIELD_SERV_NOWNEXT_DISABLED** 0x0908
- #define **DBA_FIELD_SERV_FAV_GROUPS** 0x0909
- #define **DBA_FIELD_SERV_FREESAT_ID** 0x090a
- #define **DBA_FIELD_SERV_REGION_ID** 0x090b
- #define **DBA_FIELD_SERV_LCN_EDITABLE** 0x090c
- #define **DBA_FIELD_TIMER_HANDLE** 0x0a00
- #define **DBA_FIELD_TIMER_STARTTIME** 0x0a01
- #define **DBA_FIELD_TIMER_DURATION** 0x0a02
- #define **DBA_FIELD_TIMER_TYPE** 0x0a03
- #define **DBA_FIELD_TIMER_FREQUENCY** 0x0a04
- #define **DBA_FIELD_TIMER_RAMPVOLUME** 0x0a05
- #define **DBA_FIELD_TIMER_EVENTID** 0x0a06
- #define **DBA_FIELD_TIMER_NOTIFY_TIME** 0x0a07
- #define **DBA_FIELD_TIMER_CRID** 0x0a08
- #define **DBA_FIELD_TIMER_DISKID** 0x0a09
- #define **DBA_FIELD_TIMER_OTHERCRID** 0x0a0a
- #define **DBA_FIELD_TIMER_MISSED** 0x0a0b
- #define **DBA_FIELD_TIMER_EVENT_TRIGGERED** 0x0a0c
- #define **DBA_FIELD_TIMER_ADDITIONAL_INFO** 0x0a0d
- #define **DBA_FIELD_TIMER_START_PADDING** 0x0a0e
- #define **DBA_FIELD_TIMER_END_PADDING** 0x0a0f
- #define **DBA_FIELD_CRID_EIT_DATE** 0x0b00
- #define **DBA_FIELD_CRID_SERIES** 0x0b01
- #define **DBA_FIELD_CRID_RECOMMENDED** 0x0b02
- #define **DBA_FIELD_CRID_RADIO_SERVICE** 0x0b03
- #define **DBA_FIELD_FAVLIST_ID** 0x0c00
- #define **DBA_FIELD_FAVLIST_INDEX** 0x0c01
- #define **DBA_FIELD_FAVLIST_USER_DATA** 0x0c02
- #define **DBA_FIELD_EVENT_STARTTIME** 0x0d00
- #define **DBA_FIELD_EVENT_DURATION** 0x0d01
- #define **DBA_FIELD_EVENT_ID** 0x0d02

- `#define DBA_FIELD_EVENT_CONTENT 0x0d03`
- `#define DBA_FIELD_EVENT_AGE_RATING 0x0d04`
- `#define DBA_FIELD_EVENT_SCRAMBLE 0x0d05`
- `#define DBA_FIELD_EVENT_SUBTITLES 0x0d06`
- `#define DBA_FIELD_EVENT_AUDIO_DESC 0x0d07`
- `#define DBA_FIELD_EVENT_FREE_TO_AIR 0x0d08`
- `#define DBA_FIELD_EVENT_DO_NOT_SCRAMBLE 0x0d09`
- `#define DBA_FIELD_EVENT_NAME 0x0d0a`
- `#define DBA_FIELD_EVENT_DESCRIPTION 0x0d0b`
- `#define DBA_FIELD_EVENT_EXTENDED_DESC 0x0d0c`
- `#define DBA_FIELD_EVENT_GUIDANCE 0x0d0d`
- `#define DBA_FIELD_EVENT_CONTENT_DATA 0x0d0e`

Enumerations

- `enum E_DBA_RECORDS { DBA_RECORD_LNB = 0, DBA_RECORD_SATELLITE, DBA_RECORD_NETWORK, DBA_RECORD_SAT_TRANSPORT, DBA_RECORD_TERR_TRANSPORT, DBA_RECORD_CAB_TRANSPORT, DBA_RECORD_SERVICE, DBA_RECORD_TIMER, DBA_RECORD_CRID, DBA_RECORD_FAV_LIST, DBA_RECORD_FAV_SERV, DBA_NUM_RECORDS }`

Functions

- `BOOLEAN DBA_Initialise (void)`
Performs any initialisation required prior to the database being loaded.
- `void DBA_Terminate (void)`
Releases any resources acquired by initialisation and clears any currently loaded database, including a backup, if one has been created.
- `BOOLEAN DBA_LoadDatabase (U8BIT *pathname)`
Reads a database from non-volatile storage, creating any structures in memory that will be required to access it and the records it contains and makes this the working database. If the database is found to be invalid, or doesn't exist at all, then an empty database should be created.
- `BOOLEAN DBA_SaveDatabase (void)`
Saves any changes made to the working database to non-volatile storage. If saving to a file, the pathname used to open it will be used.
- `BOOLEAN DBA_ClearDatabase (void)`
Clears the working database of all records. Following this call, it should still be possible to access the database, but it will be empty, and new records can be added to it.
- `BOOLEAN DBA_BackupDatabase (U8BIT *pathname)`
Creates a backup copy of the working database. Whether the backup database is saved to non-volatile storage is implementation dependant and so the backup database may not survive a system reboot.
- `BOOLEAN DBA_RestoreDatabase (void)`
Restores the working database from a previously made backup copy.

- void [DBA_EraseBackupDatabase](#) (void)
Erases the backup copy of the database. If data was stored in non-volatile storage then this should be deleted too.
- BOOLEAN [DBA_ExportToXML](#) (U8BIT *xml_pathname)
Export the working database to an XML file.
- BOOLEAN [DBA_ImportFromXML](#) (U8BIT *xml_pathname)
Imports records from the given XML file into the working database. If a record already exists in the database then it will be updated, and if it doesn't then a new record will be created.
- void [DBA_LockDatabase](#) (void)
Locks the database to prevent access from other threads or processes.
- void [DBA_UnlockDatabase](#) (void)
Unlocks the database to allow other threads or processes to access it.
- U8BIT * [DBA_DatabaseVersion](#) (void)
Returns a version string representing the working database.
- U32BIT [DBA_DatabaseFileSize](#) (U32BIT *max_size)
Returns the size in bytes of the database as stored in non-volatile storage.
- void * [DBA_CreateRecord](#) (U32BIT record_id, void *parent)
Creates a new record of the given type, adding it to the database as a child of the given parent record, if provided.
- void [DBA_DestroyRecord](#) (void *record)
Destroys the given record, removing it from the database and freeing any memory associated with it.
- void * [DBA_FindRecord](#) (U32BIT record_id, void *parent, void *last_rec)
Finds the next record, of the given type, that comes after last_rec. last_rec must be the same type of record as the one being found. Parent is optional, but if provided, a record will only be returned if the parent of the one found is the same.
- void [DBA_SetRecordParent](#) (void *record, void *parent)
Set of change the parent of the given record.
- void * [DBA_GetRecordParent](#) (void *record)
Returns the handle to the parent of the given record.
- void [DBA_SaveRecord](#) (void *record)
Forces a record to be saved to non-volatile storage. Depending on the implementation, this function may not do anything if the data is updated to non-volatile storage as any records and/or fields are created or updated.
- BOOLEAN [DBA_SetFieldValue](#) (void *record, U32BIT field_id, U32BIT value)
Set the value of a record's field. The function will fail if the record doesn't exist, the record doesn't include the given field, or the field isn't a value field.
- BOOLEAN [DBA_SetFieldString](#) (void *record, U32BIT field_id, U8BIT *string, U16BIT num_bytes)
Set the string value of a record's field. The function will fail if the record doesn't exist, the record doesn't include the given field, or the field isn't a string field. The string data provided will be copied and no interpretation is made of the format of the string.
- BOOLEAN [DBA_SetFieldLangString](#) (void *record, U32BIT field_id, U32BIT lang_code, U8BIT *string, U16BIT num_bytes)

Set the string value of a record's field. The function will fail if the record doesn't exist, the record doesn't include the given field, or the field isn't a string field. The string data provided will be copied and no interpretation is made of the format of the string.

- **BOOLEAN DBA_SetFieldData** (void *record, U32BIT field_id, U8BIT *data, U16BIT num_bytes)

Set a variable amount of data of a record's field. The function will fail if the record doesn't exist, the record doesn't include the given field, or the field doesn't hold data. The data provided will be copied.

- **BOOLEAN DBA_GetFieldValue** (void *record, U32BIT field_id, U32BIT *value)

Gets the value of a record's field. The function will fail if the record doesn't exist, the record doesn't include the given field, or the field isn't a value field.

- **BOOLEAN DBA_GetFieldString** (void *record, U32BIT field_id, U8BIT **string, U16BIT *num_bytes)

Gets the string value of a record's field. The function will fail if the record doesn't exist, the record doesn't include the given field, or the field isn't a string field. The pointer to the string returned will be to the string data held by the database, so the data must not be changed.

- **BOOLEAN DBA_GetFieldLangString** (void *record, U32BIT field_id, U32BIT lang_code, U8BIT **string, U16BIT *num_bytes)

Gets the string value of a record's field. The function will fail if the record doesn't exist, the record doesn't include the given field, or the field isn't a string field. The pointer to the string returned will be to the string data held by the database, so the data must not be changed.

- **BOOLEAN DBA_GetFieldData** (void *record, U32BIT field_id, U8BIT **data, U16BIT *num_bytes)

Gets the data of a record's field. The function will fail if the record doesn't exist, the record doesn't include the given field, or the field isn't a data field. The pointer to the data returned will be to the data held by the database, so the data must not be changed.

- **U32BIT DBA_DataBlockSize** (U32BIT data_block_id)

Returns the number of bytes available for the given data block.

- **U32BIT DBA_DataBlockRead** (U32BIT data_block_id, U8BIT *data, U32BIT max_num_bytes)

Read a block of data from the database into the given buffer.

- **BOOLEAN DBA_DataBlockWrite** (U32BIT data_block_id, U8BIT *data, U32BIT num_bytes)

Writes a block of data into the database from the given buffer.

4.6.1 Detailed Description

Database access defines, structures and public functions.

Date

11/11/2013

Author

Ocean Blue

4.6.2 Function Documentation

4.6.2.1 BOOLEAN DBA_BackupDatabase (U8BIT * *pathname*)

Creates a backup copy of the working database. Whether the backup database is saved to non-volatile storage is implementation dependant and so the backup database may not survive a system reboot.

Parameters

<i>pathname</i>	full pathname of the file to save the backup to if this is supported by the database implementation.
-----------------	--

Returns

TRUE if a backup is created, FALSE otherwise

4.6.2.2 BOOLEAN DBA_ClearDatabase (void)

Clears the working database of all records. Following this call, it should still be possible to access the database, but it will be empty, and new records can be added to it.

Returns

TRUE if the database is cleared, FALSE otherwise

4.6.2.3 void* DBA_CreateRecord (U32BIT *record_id*, void * *parent*)

Creates a new record of the given type, adding it to the database as a child of the given parent record, if provided.

Parameters

<i>record_id</i>	type of record to be created
<i>parent</i>	create a record that is a child of this parent, can be passed as NULL

Returns

handle of new record, or NULL if creation fails

4.6.2.4 U32BIT DBA_DatabaseFileSize (U32BIT * *max_size*)

Returns the size in bytes of the database as stored in non-volatile storage.

Parameters

<i>max_size</i>	returns the maximum size in bytes that the database can grow to. This value may be returned as 0 if this can't be determined.
-----------------	---

Returns

current size of the working database in bytes

4.6.2.5 U8BIT* DBA_DatabaseVersion (void)

Returns a version string representing the working database.

Returns

'\0' terminated string in UTF-8 format

4.6.2.6 U32BIT DBA_DataBlockRead (U32BIT *data_block_id*, U8BIT * *data*, U32BIT *max_num_bytes*)

Read a block of data from the database into the given buffer.

Parameters

<i>data_block_id</i>	id of the data block to be read
<i>data</i>	pointer to the buffer into which the data will be read
<i>max_num_bytes</i>	the maximum number of bytes to be read

Returns

number of bytes read into the buffer

4.6.2.7 U32BIT DBA_DataBlockSize (U32BIT *data_block_id*)

Returns the number of bytes available for the given data block.

Parameters

<i>data_block_id</i>	the data block whose size is to be returned
----------------------	---

Returns

size of bytes of the data block, or 0 if block not found

4.6.2.8 BOOLEAN DBA_DataBlockWrite (U32BIT *data_block_id*, U8BIT * *data*, U32BIT *num_bytes*)

Writes a block of data into the database from the given buffer.

Parameters

<i>data_block_id</i>	id of the data block being written
<i>data</i>	pointer to the data to be written
<i>num_bytes</i>	number of bytes of data to be written

Returns

TRUE if the data is written, FALSE otherwise

4.6.2.9 void DBA_DestroyRecord (void * record)

Destroys the given record, removing it from the database and freeing any memory associated with it.

Parameters

<i>record</i>	database record to be deleted
---------------	-------------------------------

4.6.2.10 BOOLEAN DBA_ExportToXML (U8BIT * xml_pathname)

Export the working database to an XML file.

Parameters

<i>xml_pathname</i>	full pathname of the file to export to
---------------------	--

Returns

TRUE if the database is exported successfully, FALSE otherwise

4.6.2.11 void* DBA_FindRecord (U32BIT record_id, void * parent, void * last_rec)

Finds the next record, of the given type, that comes after last_rec. last_rec must be the same type of record as the one being found. Parent is optional, but if provided, a record will only be returned if the parent of the one found is the same.

Parameters

<i>record_id</i>	type of record to look for
<i>parent</i>	if not NULL, this must be the parent of the record found for it to be returned
<i>last_rec</i>	previous record of the type being looked for, if NULL then the first record will be returned.

Returns

handle of the record, or NULL if none found

4.6.2.12 **BOOLEAN DBA_GetFieldData** (void * *record*, U32BIT *field_id*, U8BIT ** *data*, U16BIT * *num_bytes*)

Gets the data of a record's field. The function will fail if the record doesn't exist, the record doesn't include the given field, or the field isn't a data field. The pointer to the data returned will be to the data held by the database, so the data must not be changed.

Parameters

<i>record</i>	handle of the record being queried
<i>field_id</i>	field within the record being queried
<i>data</i>	pointer to the returned data in the field
<i>num_bytes</i>	number of bytes of data

Returns

TRUE if data is returned, FALSE otherwise

4.6.2.13 **BOOLEAN DBA_GetFieldLangString** (void * *record*, U32BIT *field_id*, U32BIT *lang_code*, U8BIT ** *string*, U16BIT * *num_bytes*)

Gets the string value of a record's field. The function will fail if the record doesn't exist, the record doesn't include the given field, or the field isn't a string field. The pointer to the string returned will be to the string data held by the database, so the data must not be changed.

Parameters

<i>record</i>	handle of the record being queried
<i>field_id</i>	field within the record being queried
<i>lang_code</i>	language code of the string to be returned
<i>string</i>	pointer to the returned string value in the field
<i>num_bytes</i>	number of bytes in the string

Returns

TRUE if a string is returned, FALSE otherwise

4.6.2.14 **BOOLEAN DBA_GetFieldString** (void * *record*, U32BIT *field_id*, U8BIT ** *string*, U16BIT * *num_bytes*)

Gets the string value of a record's field. The function will fail if the record doesn't exist, the record doesn't include the given field, or the field isn't a string field. The pointer to the string returned will be to the string data held by the database, so the data must not be changed.

Parameters

<i>record</i>	handle of the record being queried
<i>field_id</i>	field within the record being queried
<i>string</i>	pointer to the returned string value in the field
<i>num_bytes</i>	number of bytes in the string

Returns

TRUE if a string is returned, FALSE otherwise

4.6.2.15 **BOOLEAN DBA_GetFieldValue (void * *record*, U32BIT *field_id*, U32BIT * *value*)**

Gets the value of a record's field. The function will fail if the record doesn't exist, the record doesn't include the given field, or the field isn't a value field.

Parameters

<i>record</i>	handle of the record being queried
<i>field_id</i>	field within the record being queried
<i>value</i>	pointer to the returned value if function returns TRUE

Returns

TRUE if a value is returned, FALSE otherwise

4.6.2.16 **void* DBA_GetRecordParent (void * *record*)**

Returns the handle to the parent of the given record.

Parameters

<i>record</i>	handle of record whose parent is to be returned
---------------	---

Returns

handle of the record's parent

4.6.2.17 **BOOLEAN DBA_ImportFromXML (U8BIT * *xml_pathname*)**

Imports records from the given XML file into the working database. If a record already exists in the database then it will be updated, and if it doesn't then a new record will be created.

Parameters

<i>xml_pathname</i>	full pathname of the file to be imported
---------------------	--

Returns

TRUE if the file is imported successfully, FALSE otherwise

4.6.2.18 BOOLEAN DBA_Initialise (void)

Performs any initialisation required prior to the database being loaded.

Returns

TRUE if initialisation is successful, FALSE otherwise

4.6.2.19 BOOLEAN DBA_LoadDatabase (U8BIT * *pathname*)

Reads a database from non-volatile storage, creating any structures in memory that will be required to access it and the records it contains and makes this the working database. If the database is found to be invalid, or doesn't exist at all, then an empty database should be created.

Parameters

<i>pathname</i>	full pathname of the file to be loaded containing the database. This argument is only relevant where the system being used is file based.
-----------------	---

Returns

TRUE if a database is loaded successfully, FALSE otherwise

4.6.2.20 BOOLEAN DBA_RestoreDatabase (void)

Restores the working database from a previously made backup copy.

Returns

TRUE if the working database is restored from the backup, FALSE otherwise

4.6.2.21 BOOLEAN DBA_SaveDatabase (void)

Saves any changes made to the working database to non-volatile storage. If saving to a file, the pathname used to open it will be used.

Returns

TRUE if the database is saved successfully, FALSE otherwise

4.6.2.22 void DBA_SaveRecord (void * *record*)

Forces a record to be saved to non-volatile storage. Depending on the implementation, this function may not do anything if the data is updated to non-volatile storage as any records and/or fields are created or updated.

Parameters

<i>record</i>	handle of record to be saved
---------------	------------------------------

4.6.2.23 **BOOLEAN DBA_SetFieldData (void * *record*, U32BIT *field_id*, U8BIT * *data*, U16BIT *num_bytes*)**

Set a variable amount of data of a record's field. The function will fail if the record doesn't exist, the record doesn't include the given field, or the field doesn't hold data. The data provided will be copied.

Parameters

<i>record</i>	handle of the record to be changed
<i>field_id</i>	field within the record to be changed
<i>data</i>	data to be stored in the field
<i>num_bytes</i>	number of bytes of data

Returns

TRUE if the data is set, FALSE otherwise

4.6.2.24 **BOOLEAN DBA_SetFieldLangString (void * *record*, U32BIT *field_id*, U32BIT *lang_code*, U8BIT * *string*, U16BIT *num_bytes*)**

Set the string value of a record's field. The function will fail if the record doesn't exist, the record doesn't include the given field, or the field isn't a string field. The string data provided will be copied and no interpretation is made of the format of the string.

Parameters

<i>record</i>	handle of the record to be changed
<i>field_id</i>	field within the record to be changed
<i>lang_code</i>	language code of the string
<i>string</i>	string value of the field
<i>num_bytes</i>	number of bytes in the string, should include null terminator if present

Returns

TRUE if the string is set, FALSE otherwise

4.6.2.25 **BOOLEAN DBA_SetFieldString (void * *record*, U32BIT *field_id*, U8BIT * *string*, U16BIT *num_bytes*)**

Set the string value of a record's field. The function will fail if the record doesn't exist, the record doesn't include the given field, or the field isn't a string field. The string data provided will be copied and no interpretation is made of the format of the string.

Parameters

<i>record</i>	handle of the record to be changed
<i>field_id</i>	field within the record to be changed
<i>string</i>	string value of the field
<i>num_bytes</i>	number of bytes in the string, should include null terminator if present

Returns

TRUE if the string is set, FALSE otherwise

4.6.2.26 **BOOLEAN DBA_SetFieldValue (void * *record*, U32BIT *field_id*, U32BIT *value*)**

Set the value of a record's field. The function will fail if the record doesn't exist, the record doesn't include the given field, or the field isn't a value field.

Parameters

<i>record</i>	handle of the record to be changed
<i>field_id</i>	field within the record to be changed
<i>value</i>	new value of the field

Returns

TRUE if the value is set, FALSE otherwise

4.6.2.27 **void DBA_SetRecordParent (void * *record*, void * *parent*)**

Set of change the parent of the given record.

Parameters

<i>record</i>	handle of the record whose parent is to be changed
<i>parent</i>	handle of the new parent record, can be NULL

4.7 dvb/inc/ap_cfg.h File Reference

Application configuration.

Data Structures

- struct [ana_rf_channel_data](#)
- struct [ter_rf_channel_data](#)
- struct [cab_rf_channel_data](#)

Defines

- #define **ACFG_INVALID_REGION** 255
- #define **ACFG_INVALID_LANG** 255
- #define **COUNTRY_CODE_ARGENTINA** (U32BIT)(('a' << 16) | ('r' << 8) | 'g')
- #define **COUNTRY_CODE_AUSTRALIA** (U32BIT)(('a' << 16) | ('u' << 8) | 's')
- #define **COUNTRY_CODE_AUSTRIA** (U32BIT)(('a' << 16) | ('u' << 8) | 't')
- #define **COUNTRY_CODE_BELGIUM** (U32BIT)(('b' << 16) | ('e' << 8) | 'l')
- #define **COUNTRY_CODE_BOLIVIA** (U32BIT)(('b' << 16) | ('o' << 8) | 'l')
- #define **COUNTRY_CODE_BRAZIL** (U32BIT)(('b' << 16) | ('r' << 8) | 'a')
- #define **COUNTRY_CODE_CHILE** (U32BIT)(('c' << 16) | ('h' << 8) | 'l')
- #define **COUNTRY_CODE_CHINA** (U32BIT)(('c' << 16) | ('h' << 8) | 'i')
- #define **COUNTRY_CODE_COLOMBIA** (U32BIT)(('c' << 16) | ('o' << 8) | 'l')
- #define **COUNTRY_CODE_COSTARICA** (U32BIT)(('c' << 16) | ('r' << 8) | 'i')
- #define **COUNTRY_CODE_CROATIA** (U32BIT)(('h' << 16) | ('r' << 8) | 'v')
- #define **COUNTRY_CODE_CZECHREP** (U32BIT)(('c' << 16) | ('z' << 8) | 'e')
- #define **COUNTRY_CODE_DOMINICANREP** (U32BIT)(('d' << 16) | ('m' << 8) | 'a')
- #define **COUNTRY_CODE_ECUADOR** (U32BIT)(('e' << 16) | ('c' << 8) | 'u')
- #define **COUNTRY_CODE_ELSALVADOR** (U32BIT)(('s' << 16) | ('l' << 8) | 'v')
- #define **COUNTRY_CODE_FINLAND** (U32BIT)(('f' << 16) | ('i' << 8) | 'n')
- #define **COUNTRY_CODE_FRANCE** (U32BIT)(('f' << 16) | ('r' << 8) | 'a')
- #define **COUNTRY_CODE_GERMANY** (U32BIT)(('d' << 16) | ('e' << 8) | 'u')
- #define **COUNTRY_CODE_GUATEMALA** (U32BIT)(('g' << 16) | ('t' << 8) | 'm')
- #define **COUNTRY_CODE_HONDURAS** (U32BIT)(('h' << 16) | ('n' << 8) | 'd')
- #define **COUNTRY_CODE_HUNGARY** (U32BIT)(('h' << 16) | ('u' << 8) | 'n')
- #define **COUNTRY_CODE_ITALY** (U32BIT)(('i' << 16) | ('t' << 8) | 'a')
- #define **COUNTRY_CODE_LATVIA** (U32BIT)(('l' << 16) | ('v' << 8) | 'a')
- #define **COUNTRY_CODE_LUXEMBOURG** (U32BIT)(('l' << 16) | ('u' << 8) | 'x')
- #define **COUNTRY_CODE_MEXICO** (U32BIT)(('m' << 16) | ('e' << 8) | 'x')
- #define **COUNTRY_CODE_NETHERLANDS** (U32BIT)(('n' << 16) | ('l' << 8) | 'd')
- #define **COUNTRY_CODE_NICARAGUA** (U32BIT)(('n' << 16) | ('i' << 8) | 'c')
- #define **COUNTRY_CODE_PANAMA** (U32BIT)(('p' << 16) | ('a' << 8) | 'n')
- #define **COUNTRY_CODE_PERU** (U32BIT)(('p' << 16) | ('e' << 8) | 'r')
- #define **COUNTRY_CODE_POLAND** (U32BIT)(('p' << 16) | ('o' << 8) | 'l')
- #define **COUNTRY_CODE_PORTUGAL** (U32BIT)(('p' << 16) | ('o' << 8) | 'r')
- #define **COUNTRY_CODE_RUSSIA** (U32BIT)(('r' << 16) | ('u' << 8) | 's')
- #define **COUNTRY_CODE_SERBIA** (U32BIT)(('s' << 16) | ('r' << 8) | 'b')
- #define **COUNTRY_CODE_SLOVAKIA** (U32BIT)(('s' << 16) | ('v' << 8) | 'k')
- #define **COUNTRY_CODE_SLOVENIA** (U32BIT)(('s' << 16) | ('v' << 8) | 'n')
- #define **COUNTRY_CODE_SOUTHAFRICA** (U32BIT)(('z' << 16) | ('a' << 8) | 'f')
- #define **COUNTRY_CODE_SPAIN** (U32BIT)(('e' << 16) | ('s' << 8) | 'p')

- #define **COUNTRY_CODE_SWEDEN** (U32BIT)(('s' << 16) | ('w' << 8) | 'e')
- #define **COUNTRY_CODE_SWITZERLAND** (U32BIT)(('c' << 16) | ('h' << 8) | 'e')
- #define **COUNTRY_CODE_UK** (U32BIT)(('g' << 16) | ('b' << 8) | 'r')
- #define **COUNTRY_CODE_UKRAINE** (U32BIT)(('u' << 16) | ('k' << 8) | 'r')
- #define **COUNTRY_CODE_VENEZUELA** (U32BIT)(('v' << 16) | ('e' << 8) | 'n')
- #define **ENGLISH_LANG_CODE** (('e' << 16) | ('n' << 8) | 'g')
- #define **WELSH1_LANG_CODE** (('w' << 16) | ('e' << 8) | 'l')
- #define **WELSH2_LANG_CODE** (('c' << 16) | ('y' << 8) | 'm')
- #define **GAELIC_LANG_CODE** (('g' << 16) | ('l' << 8) | 'a')
- #define **IRISH_LANG_CODE** (('g' << 16) | ('l' << 8) | 'e')
- #define **DUTCH1_LANG_CODE** (('n' << 16) | ('l' << 8) | 'd')
- #define **DUTCH2_LANG_CODE** (('d' << 16) | ('u' << 8) | 't')
- #define **GERMAN1_LANG_CODE** (('g' << 16) | ('e' << 8) | 'r')
- #define **GERMAN2_LANG_CODE** (('d' << 16) | ('e' << 8) | 'u')
- #define **FRENCH1_LANG_CODE** (('f' << 16) | ('r' << 8) | 'e')
- #define **FRENCH2_LANG_CODE** (('f' << 16) | ('r' << 8) | 'a')
- #define **RUSSIAN_LANG_CODE** (('r' << 16) | ('u' << 8) | 's')
- #define **SIMPLIFIED_CHINESE_LANG_CODE** (('c' << 16) | ('h' << 8) | 'i')
- #define **TRADITIONAL_CHINESE_LANG_CODE** (('c' << 16) | ('h' << 8) | 'i')
- #define **FINNISH_LANG_CODE** (('f' << 16) | ('i' << 8) | 'n')
- #define **SWEDISH_LANG_CODE** (('s' << 16) | ('w' << 8) | 'e')
- #define **NORWEGIAN_LANG_CODE** (('n' << 16) | ('o' << 8) | 'r')
- #define **DANISH_LANG_CODE** (('d' << 16) | ('a' << 8) | 'n')
- #define **MANDARIN_LANG_CODE** (('c' << 16) | ('m' << 8) | 'n')
- #define **CANTONESE_LANG_CODE** (('y' << 16) | ('u' << 8) | 'e')
- #define **MAORI1_LANG_CODE** (('m' << 16) | ('a' << 8) | 'o')
- #define **MAORI2_LANG_CODE** (('m' << 16) | ('r' << 8) | 'i')
- #define **JAPANESE_LANG_CODE** (('j' << 16) | ('p' << 8) | 'n')
- #define **ITALIAN_LANG_CODE** (('i' << 16) | ('t' << 8) | 'a')
- #define **SPANISH1_LANG_CODE** (('s' << 16) | ('p' << 8) | 'a')
- #define **SPANISH2_LANG_CODE** (('e' << 16) | ('s' << 8) | 'l')
- #define **KOREAN_LANG_CODE** (('k' << 16) | ('o' << 8) | 'r')
- #define **HINDI_LANG_CODE** (('h' << 16) | ('i' << 8) | 'n')
- #define **CZECH1_LANG_CODE** (('c' << 16) | ('z' << 8) | 'e')
- #define **CZECH2_LANG_CODE** (('c' << 16) | ('e' << 8) | 's')
- #define **SLOVAK1_LANG_CODE** (('s' << 16) | ('l' << 8) | 'k')
- #define **SLOVAK2_LANG_CODE** (('s' << 16) | ('l' << 8) | 'o')
- #define **AFRIKAANS_LANG_CODE** (('a' << 16) | ('f' << 8) | 'r')
- #define **NDEBELE_LANG_CODE** (('n' << 16) | ('b' << 8) | 'l')
- #define **NORTHERN_SOTHO_LANG_CODE** (('n' << 16) | ('s' << 8) | 'o')
- #define **SOUTHERN_SOTHO_LANG_CODE** (('s' << 16) | ('o' << 8) | 't')
- #define **SWATI_LANG_CODE** (('s' << 16) | ('s' << 8) | 'w')
- #define **TSONGA_LANG_CODE** (('t' << 16) | ('s' << 8) | 'o')
- #define **TSWANA_LANG_CODE** (('t' << 16) | ('s' << 8) | 'n')
- #define **VENDA_LANG_CODE** (('v' << 16) | ('e' << 8) | 'n')

- #define **XHOSA_LANG_CODE** (('x' << 16) | ('h' << 8) | 'o')
- #define **ZULU_LANG_CODE** (('z' << 16) | ('u' << 8) | 'l')
- #define **CROATIAN_LANG_CODE** (('h' << 16) | ('r' << 8) | 'v')
- #define **HUNGARIAN_LANG_CODE** (('h' << 16) | ('u' << 8) | 'n')
- #define **LATVIAN_LANG_CODE** (('l' << 16) | ('a' << 8) | 'v')
- #define **LUXEMBOURGISH_LANG_CODE** (('l' << 16) | ('t' << 8) | 'z')
- #define **POLISH_LANG_CODE** (('p' << 16) | ('o' << 8) | 'l')
- #define **PORTUGUESE_LANG_CODE** (('p' << 16) | ('o' << 8) | 'r')
- #define **SERBIAN_LANG_CODE** (('s' << 16) | ('r' << 8) | 'p')
- #define **ROMANIAN1_LANG_CODE** (('r' << 16) | ('o' << 8) | 'n')
- #define **ROMANIAN2_LANG_CODE** (('r' << 16) | ('u' << 8) | 'm')
- #define **ALBANIAN1_LANG_CODE** (('s' << 16) | ('q' << 8) | 'i')
- #define **ALBANIAN2_LANG_CODE** (('a' << 16) | ('l' << 8) | 'b')
- #define **SLOVENE_LANG_CODE** (('s' << 16) | ('l' << 8) | 'v')
- #define **CATALAN_LANG_CODE** (('c' << 16) | ('a' << 8) | 't')
- #define **GALICIAN_LANG_CODE** (('g' << 16) | ('l' << 8) | 'g')
- #define **BASQUE1_LANG_CODE** (('e' << 16) | ('u' << 8) | 's')
- #define **BASQUE2_LANG_CODE** (('b' << 16) | ('a' << 8) | 'q')
- #define **UKRAINIAN_LANG_CODE** (('u' << 16) | ('k' << 8) | 'r')
- #define **UND_LANG_CODE** (('u' << 16) | ('n' << 8) | 'd')

Typedefs

- typedef struct [ana_rf_channel_data](#) **ACFG_ANA_RF_CHANNEL_DATA**
- typedef struct [ter_rf_channel_data](#) **ACFG_TER_RF_CHANNEL_DATA**
- typedef struct [cab_rf_channel_data](#) **ACFG_CAB_RF_CHANNEL_DATA**
- typedef BOOLEAN(* **AllocLcnFunc**)(E_STB_DP_SIGNAL_TYPE tuner_type)
- typedef void(* **DBTidyFunc**)(E_STB_DP_SIGNAL_TYPE tuner_type, BOOLEAN search_completed, BOOLEAN manual)

Enumerations

- enum **E_DB_LANGS** { **ACFG_DB_LANG_ENGLISH**, **ACFG_DB_LANG_WELSH1**, **ACFG_DB_LANG_WELSH2**, **ACFG_DB_LANG_GAELIC**, **ACFG_DB_LANG_IRISH**, **ACFG_DB_LANG_DUTCH1**, **ACFG_DB_LANG_DUTCH2**, **ACFG_DB_LANG_GERMAN1**, **ACFG_DB_LANG_GERMAN2**, **ACFG_DB_LANG_FRENCH1**, **ACFG_DB_LANG_FRENCH2**, **ACFG_DB_LANG_RUSSIAN**, **ACFG_DB_LANG_SIMPLIFIED_CHINESE**, **ACFG_DB_LANG_TRADITIONAL_CHINESE**, **ACFG_DB_LANG_FINNISH**, **ACFG_DB_LANG_SWEDISH**, **ACFG_DB_LANG_NORWEGIAN**, **ACFG_DB_LANG_DANISH**, **ACFG_DB_LANG_MANDARIN**, **ACFG_DB_LANG_CANTONESE**, **ACFG_DB_LANG_MAORI1**, **ACFG_DB_LANG_MAORI2**, **ACFG_DB_LANG_JAPANESE**, **ACFG_DB_LANG_ITALIAN**, **ACFG_DB_LANG_SPANISH1**, **ACFG_DB_LANG_SPANISH2**, **ACFG_DB_LANG_KOREAN**, **ACFG_DB_LANG_HINDI**, **ACFG_DB_LANG_CZECH1**, **ACFG_DB_LANG_CZECH2**, **ACFG_DB_LANG_SLOVAK1**, **ACFG_DB_LANG_SLOVAK2**, **ACFG_DB_LANG_AFRIKAANS**, **ACFG_DB_LANG_NDEBELE**, **ACFG_DB_LANG_NORTHERN_SOTHO**, **ACFG_DB_LANG_S-**

```

OUTHERN_SOTHO, ACFG_DB_LANG_SWATI, ACFG_DB_LANG_TSONG-
A, ACFG_DB_LANG_TSWANA, ACFG_DB_LANG_VENDA, ACFG_DB_LA-
NG_XHOSA, ACFG_DB_LANG_ZULU, ACFG_DB_LANG_CROATIAN, ACF-
G_DB_LANG_HUNGARIAN, ACFG_DB_LANG_LATVIAN, ACFG_DB_LAN-
G_LUXEMBOURGISH, ACFG_DB_LANG_POLISH, ACFG_DB_LANG_POR-
TUGUESE, ACFG_DB_LANG_SERBIAN, ACFG_DB_LANG_ROMANIAN1, -
ACFG_DB_LANG_ROMANIAN2, ACFG_DB_LANG_ALBANIAN1, ACFG_D-
B_LANG_ALBANIAN2, ACFG_DB_LANG_SLOVENE, ACFG_DB_LANG_CA-
TALAN, ACFG_DB_LANG_GALICIAN, ACFG_DB_LANG_BASQUE1, ACFG-
_DB_LANG_BASQUE2, ACFG_DB_LANG_UKRAINIAN, ACFG_DB_LANG_-
UND, ACFG_NUM_DB_LANGUAGES }

```

Functions

- U8BIT [ACFG_GetNumCountries](#) (void)
Returns the number of country configurations included in the DVB stack.
- U32BIT [ACFG_GetCountry](#) (void)
Returns the country code the DVB is configured for.
- void [ACFG_GetCountryList](#) (U8BIT ***str_array_ptr, U8BIT *num_countries_ptr)
Returns an array containing the names of all the countries included in the DVB. The index into this array is referred to as the country_id. The returned array should be freed using ACFG_ReleaseCountryList.
- void [ACFG_ReleaseCountryList](#) (U8BIT **str_array, U8BIT num_countries)
Frees a country list previously acquired using ACFG_GetCountryList.
- U8BIT [ACFG_GetNumRegions](#) (U32BIT country_code)
Returns the number of regions in the given country.
- BOOLEAN [ACFG_GetRegionList](#) (U32BIT country_code, U8BIT ***str_array_ptr, U8BIT *num_regions_ptr)
Returns an array of pointers to strings containing the region names for the given country. The index into this array is the region_id. The array is allocated by this function and should be freed using ACFG_ReleaseRegionList.
- BOOLEAN [ACFG_GetRegionCode](#) (U32BIT country_code, U8BIT region_id, U8BIT *code_ptr)
Returns the region code that identifies the given region.
- void [ACFG_ReleaseRegionList](#) (U8BIT **str_array, U8BIT num_regions)
Frees the region array previously acquired using ACFG_GetRegionList.
- U8BIT [ACFG_GetCountryIndex](#) (U32BIT country_code)
Returns the index in the list of countries known by the DVB, of the given country.
- U8BIT [ACFG_GetRegionId](#) (void)
Returns the internal ID of the region the DVB is configured for.
- BOOLEAN [ACFG_SetCountry](#) (U32BIT country_code)
Sets the current country and sets default values for region and language.
- U8BIT [ACFG_GetNumDbLanguages](#) (U32BIT country_code)
Returns the number of languages defined for the given country that can be used for audio and subtitles/teletext.

- **BOOLEAN** [ACFG_GetDbLangList](#) (U32BIT country_code, U8BIT ***str_array_ptr, U8BIT *num_langs_ptr)
Returns an array of pointers to strings containing the available language names for the given country. The index into this array is the lang_id used when getting and setting audio and text language ids. The array is allocated by this function and should be freed using ACFG_ReleaseDbLangList.
- **U8BIT *** [ACFG_GetDbLangId](#) (U32BIT country_code, U8BIT lang_entry)
Returns an array of language ids for a given country and index into the language array. A language id is one of the ACFG_DB_LANG_XXXX values above and there might be more than one for a given language. For example german language has two ids associated to it: ACFG_DB_LANG_GERMAN1 and ACFG_DB_LANG_GERMAN2 because german language can have two different codes, 'deu' and 'ger'. See also functions ACFG_ConvertLangIdToCode and ACFG_ConvertLangCodeToId. The returned pointer is an internal array and must not be freed.
- **void** [ACFG_ReleaseDbLangList](#) (U8BIT **str_array, U8BIT num_langs)
Frees the language array previously acquired using ACFG_GetDbLangList.
- **U8BIT** [ACFG_ConvertLangCodeToId](#) (U32BIT lang_code)
Returns the language id for the given language code.
- **U32BIT** [ACFG_ConvertLangIdToCode](#) (U8BIT lang_id)
Returns the language code for the given language id.
- **void** [ACFG_SetCountryIds](#) (U32BIT country_code, U8BIT region_id, U8BIT audio_lang_id, U8BIT sub_lang_id)
Saves the configured country and changes settings related to the country.
- **U8BIT** [ACFG_GetPrimaryAudioLangId](#) (void)
Returns the primary audio language ID.
- **U8BIT** [ACFG_GetSecondaryAudioLangId](#) (void)
Returns the secondary audio language ID.
- **U8BIT** [ACFG_GetPrimaryTextLangId](#) (void)
Returns the primary DVB subtitle/teletext language ID.
- **U8BIT** [ACFG_GetSecondaryTextLangId](#) (void)
Returns the secondary DVB subtitle/teletext language ID.
- **void** [ACFG_SetPrimaryAudioLangId](#) (U8BIT lang_id)
Sets the primary language id to be used for audio. This is the language that will be chosen first, if available.
- **void** [ACFG_SetSecondaryAudioLangId](#) (U8BIT lang_id)
Sets the secondary language id to be used for audio.
- **void** [ACFG_SetPrimaryTextLangId](#) (U8BIT lang_id)
Sets the primary language id to be used for teletext/subtitles.
- **void** [ACFG_SetSecondaryTextLangId](#) (U8BIT lang_id)
Sets the secondary language id to be used for teletext/subtitles.
- **U8BIT** [ACFG_GetDefaultSecondaryLangId](#) (void)
Returns the default secondary language for audio and subtitles as defined for the selected country. This is used for countries that don't use the secondary language, but expect a defined language to be used if the primary one isn't available.
- **BOOLEAN** [ACFG_GetAnaRfChannelTable](#) (U32BIT country_code, [ACFG_ANA_RF_CHANNEL_DATA](#) **rf_chan_data_ptr, U16BIT *num_entries_ptr)

Returns a pointer to the channel table for an analog tuner for the given country.

- **BOOLEAN** [ACFG_GetTerRfChannelTable](#) (U32BIT country_code, [ACFG_TER_RF_CHANNEL_DATA](#) **rf_chan_data_ptr, U16BIT *num_entries_ptr)

Returns a pointer to the channel table for a terrestrial tuner for the given country.

- **BOOLEAN** [ACFG_GetCabRfChannelTable](#) (U32BIT country_code, [ACFG_CAB_RF_CHANNEL_DATA](#) **rf_chan_data_ptr, U16BIT *num_entries_ptr)

Returns a pointer to the channel table for a cable tuner for the given country.

- **void** [ACFG_SetCableChannelTable](#) ([ACFG_CAB_RF_CHANNEL_DATA](#) *cable_channel_data, U16BIT number_channels)

Replaces the cable tuning table for the current country with the one provided.

- U32BIT [ACFG_GetCountryCode](#) (U8BIT country_id)

Returns the country code for the given country index.

- U32BIT [ACFG_GetPrivateDataSpecifier](#) (U32BIT country_code)

Returns the private data specifier value for the given country.

- U16BIT [ACFG_GetFirstUnallocatedLcn](#) (U32BIT country_code)

Returns the first LCN that should be used when assigning LCNs to services that don't appear in an LCN descriptor, or can't be assigned their desired LCN.

- U16BIT [ACFG_GetLastUnallocatedLcn](#) (U32BIT country_code)

Returns the last LCN that should be used when assigning LCNs to services that don't appear in an LCN descriptor, or can't be assigned their desired LCN.

- **BOOLEAN** [ACFG_GetWatershedTimes](#) (U32BIT country_code, U16BIT *start_time, U16BIT *end_time)

Get the start and end times of the watershed hours for a country.

- U16BIT [ACFG_GetMinSatelliteScanFreq](#) (U32BIT country_code)

Returns the minimum frequency to be used when performing a satellite based frequency scan in the given country.

- U16BIT [ACFG_GetMaxSatelliteScanFreq](#) (U32BIT country_code)

Returns the maximum frequency to be used when performing a satellite based frequency scan in the given country.

- U16BIT [ACFG_GetSatelliteScanFreqInc](#) (U32BIT country_code)

Returns the increment frequency to be used when performing a satellite based frequency scan in the given country.

- U16BIT * [ACFG_GetSatelliteScanSymbolRates](#) (U32BIT country_code)

Returns a fixed array of symbol rates to be used when performing a satellite based frequency scan in the given country.

- U8BIT [ACFG_GetSatelliteScanNumSymbolRates](#) (U32BIT country_code)

Returns the number of items in the fixed array of symbol rates to be used when performing a satellite based frequency scan in the given country.

- **BOOLEAN** [ACFG_GetSatelliteScanDvbS2](#) (U32BIT country_code)

Returns whether DVB-S2 should be included when performing a satellite based frequency scan in the given country.

- **BOOLEAN** [ACFG_GetAerialPowerOptionReqd](#) (U32BIT country_code)

Returns whether the aerial power option is required for DVB-T/T2 for the given country.

- **BOOLEAN** [ACFG_GetDefaultAerialPower](#) (U32BIT country_code)

Returns the default aerial power option setting for DVB-T/T2 for the given country.

- **BOOLEAN** [ACFG_GetAerialTuningScreenReqd](#) (U32BIT country_code)
Returns whether an aerial tuning screen should be presented by the interface before performing a DVB-T/T2 scan for services for the given country.
- **U8BIT *** [ACFG_GetEventContentTypes](#) (U32BIT country_code)
Returns the event content types for the given country. This defines how the content type value broadcast as part of the EIT is to be interpreted.
- **void** [ACFG_SetBackgroundSearchTime](#) (U16BIT start_time, U16BIT end_time)
Sets the start and end times during which background searches are allowed run when in standby.
- **void** [ACFG_GetBackgroundSearchTime](#) (U16BIT *start_time, U16BIT *end_time)
Gets the background start and end search times.
- **void** [ACFG_SetBackgroundServiceSearch](#) (BOOLEAN enabled)
Enables or disables the background service search when in standby.
- **BOOLEAN** [ACFG_GetBackgroundServiceSearch](#) (void)
Returns whether the background service search is enabled or not.
- **void** [ACFG_SetBackgroundSSUSearch](#) (BOOLEAN enabled)
Enables or disables the background SSU search when in standby.
- **BOOLEAN** [ACFG_GetBackgroundSSUSearch](#) (void)
Returns whether the background SSU search is enabled or not.
- **BOOLEAN** [ACFG_IsNordigCountry](#) (void)
Returns whether the current country requires Nordig compliance for SI.
- **U8BIT** [ACFG_GetEitUpdateTime](#) (void)
Returns the number of minutes after which all the events should have been received during an EIT search. This is the EIT repetition time.
- **U8BIT** [ACFG_GetEitSearchesPerDay](#) (void)
Returns the number of EIT searches that should be performed per day when the box is in standby.
- **AllocLcnFunc** [ACFG_GetTerrestrialLcnFunction](#) (U32BIT country_code)
Returns a pointer to a function that's used to assign LCNs following a DVB-T/T2 scan for the given country.
- **void** [ACFG_SetTerrestrialLcnFunction](#) (U32BIT country_code, AllocLcnFunc func_ptr)
Overrides an existing DVB-T/T2 LCN allocation function for the given country.
- **AllocLcnFunc** [ACFG_GetCableLcnFunction](#) (U32BIT country_code)
Returns a pointer to a function that's used to assign LCNs following a DVB-C scan for the given country.
- **void** [ACFG_SetCableLcnFunction](#) (U32BIT country_code, AllocLcnFunc func_ptr)
Overrides an existing DVB-C LCN allocation function for the given country.
- **AllocLcnFunc** [ACFG_GetSatelliteLcnFunction](#) (U32BIT country_code)
Returns a pointer to a function that's used to assign LCNs following a DVB-S/S2 scan for the given country.
- **void** [ACFG_SetSatelliteLcnFunction](#) (U32BIT country_code, AllocLcnFunc func_ptr)

Overrides an existing DVB-S/S2 LCN allocation function for the given country.

- DBTidyFunc [ACFG_GetTerrestrialDBTidyFunction](#) (U32BIT country_code)

Returns a pointer to a function that's used to tidy up the database following a DVB-T/T2 scan for the given country.

- DBTidyFunc [ACFG_GetCableDBTidyFunction](#) (U32BIT country_code)

Returns a pointer to a function that's used to tidy up the database following a DVB-C scan for the given country.

- DBTidyFunc [ACFG_GetSatelliteDBTidyFunction](#) (U32BIT country_code)

Returns a pointer to a function that's used to tidy up the database following a DVB-S/-S2 scan for the given country.

4.7.1 Detailed Description

Application configuration.

Date

5/04/2004

Author

Ocean Blue

4.7.2 Function Documentation

4.7.2.1 U8BIT ACFG_ConvertLangCodeToId (U32BIT lang_code)

Returns the language id for the given language code.

Parameters

<i>lang_code</i>	language code
------------------	---------------

Returns

language id, or ACFG_INVALID_LANG if language code isn't found

4.7.2.2 U32BIT ACFG_ConvertLangIdToCode (U8BIT lang_id)

Returns the language code for the given language id.

Parameters

<i>lang_id</i>	language id
----------------	-------------

Returns

language code, or 0 if language id isn't valid

4.7.2.3 BOOLEAN ACFG_GetAerialPowerOptionReqd (U32BIT *country_code*)

Returns whether the aerial power option is required for DVB-T/T2 for the given country.

Parameters

<i>country_code</i>	country code
---------------------	--------------

Returns

TRUE if required, FALSE otherwise

4.7.2.4 BOOLEAN ACFG_GetAerialTuningScreenReqd (U32BIT *country_code*)

Returns whether an aerial tuning screen should be presented by the interface before performing a DVB-T/T2 scan for services for the given country.

Parameters

<i>country_code</i>	country code
---------------------	--------------

Returns

TRUE if required, FALSE otherwise

4.7.2.5 BOOLEAN ACFG_GetAnaRfChannelTable (U32BIT *country_code*, ACFG_ANA_RF_CHANNEL_DATA ** *rf_chan_data_ptr*, U16BIT * *num_entries_ptr*)

Returns a pointer to the channel table for an analog tuner for the given country.

Parameters

<i>country_code</i>	country whose table is to be returned
<i>rf_chan_data_ptr</i>	returned pointer to the channel table
<i>num_entries_ptr</i>	number of entries in the returned channel table

Returns

TRUE if the country is valid and data is returned, FALSE otherwise

4.7.2.6 void ACFG_GetBackgroundSearchTime (U16BIT * *start_time*, U16BIT * *end_time*)

Gets the background start and end search times.

Parameters

<i>start_time</i>	return start time in minutes since midnight
<i>end_time</i>	return end time in minutes

4.7.2.7 BOOLEAN ACFG_GetBackgroundServiceSearch (void)

Returns whether the background service search is enabled or not.

Returns

TRUE if enabled, FALSE otherwise

4.7.2.8 BOOLEAN ACFG_GetBackgroundSSUSearch (void)

Returns whether the background SSU search is enabled or not.

Returns

TRUE if enabled, FALSE otherwise

4.7.2.9 DBTidyFunc ACFG_GetCableDBTidyFunction (U32BIT *country_code*)

Returns a pointer to a function that's used to tidy up the database following a DVB-C scan for the given country.

Parameters

<i>country_ - code</i>	country code
----------------------------	--------------

Returns

pointer to function, or NULL

4.7.2.10 AllocLcnFunc ACFG_GetCableLcnFunction (U32BIT *country_code*)

Returns a pointer to a function that's used to assign LCNs following a DVB-C scan for the given country.

Parameters

<i>country_ - code</i>	country id
----------------------------	------------

Returns

pointer to function, or NULL

4.7.2.11 **BOOLEAN** **ACFG_GetCabRfChannelTable** (**U32BIT** *country_code*,
ACFG_CAB_RF_CHANNEL_DATA ** *rf_chan_data_ptr*, **U16BIT** *
num_entries_ptr)

Returns a pointer to the channel table for a cable tuner for the given country.

Parameters

<i>country_code</i>	country whose table is to be returned
<i>rf_chan_data_ptr</i>	returned pointer to the channel table
<i>num_entries_ptr</i>	number of entries in the returned channel table

Returns

TRUE if the country is valid and data is returned, FALSE otherwise

4.7.2.12 **U32BIT** **ACFG_GetCountry** (**void**)

Returns the country code the DVB is configured for.

Returns

country code

4.7.2.13 **U32BIT** **ACFG_GetCountryCode** (**U8BIT** *country_id*)

Returns the country code for the given country index.

Parameters

<i>country_id</i>	country index
-------------------	---------------

Returns

country code, or 0 if country id isn't valid

4.7.2.14 **U8BIT** **ACFG_GetCountryIndex** (**U32BIT** *country_code*)

Returns the index in the list of countries known by the DVB, of the given country.

Parameters

<i>country_ - code</i>	country code
----------------------------	--------------

Returns

country index, NUM_COUNTRIES if the country isn't found

4.7.2.15 void **ACFG_GetCountryList** (U8BIT *** *str_array_ptr*, U8BIT * *num_countries_ptr*)

Returns an array containing the names of all the countries included in the DVB. The index into this array is referred to as the *country_id*. The returned array should be freed using **ACFG_ReleaseCountryList**.

Parameters

<i>str_array_ptr</i>	pointer to an allocated array of static UTF-8 strings
<i>num_ - countries_ - ptr</i>	pointer to the number of countries in the returned array

4.7.2.16 U8BIT* **ACFG_GetDbLangId** (U32BIT *country_code*, U8BIT *lang_entry*)

Returns an array of language ids for a given country and index into the language array. A language id is one of the **ACFG_DB_LANG_XXXX** values above and there might be more than one for a given language. For example german language has two ids associated to it: **ACFG_DB_LANG_GERMAN1** and **ACFG_DB_LANG_GERMAN2** because german language can have two different codes, 'deu' and 'ger'. See also functions **ACFG_ConvertLangIdToCode** and **ACFG_ConvertLangCodeToId**. The returned pointer is an internal array and must not be freed.

Parameters

<i>country_ - code</i>	country code
<i>lang_entry</i>	index into the language array

Returns

pointer to the array of language ids, or NULL if either of the indices are invalid.

4.7.2.17 BOOLEAN **ACFG_GetDbLangList** (U32BIT *country_code*, U8BIT *** *str_array_ptr*, U8BIT * *num_langs_ptr*)

Returns an array of pointers to strings containing the available language names for the given country. The index into this array is the *lang_id* used when getting and setting audio and text language ids. The array is allocated by this function and should be freed using **ACFG_ReleaseDbLangList**.

Parameters

<i>country_ - code</i>	country code
<i>str_array_ptr</i>	pointer to an allocated array of static UTF-8 strings
<i>num_langs- _ptr</i>	pointer to the number of languages in the returned array

Returns

TRUE if the country_code is valid

4.7.2.18 BOOLEAN ACFG_GetDefaultAerialPower (U32BIT country_code)

Returns the default aerial power option setting for DVB-T/T2 for the given country.

Parameters

<i>country_ - code</i>	country code
----------------------------	--------------

Returns

TRUE or FALSE

4.7.2.19 U8BIT ACFG_GetDefaultSecondaryLangId (void)

Returns the default secondary language for audio and subtitles as defined for the selected country. This is used for countries that don't use the secondary language, but expect a defined language to be used if the primary one isn't available.

Returns

language id, or ACFG_INVALID_LANG when the value is not defined

4.7.2.20 U8BIT ACFG_GetEitSearchesPerDay (void)

Returns the number of EIT searches that should be performed per day when the box is in standby.

Returns

number of times

4.7.2.21 U8BIT ACFG_GetEitUpdateTime (void)

Returns the number of minutes after which all the events should have been received during an EIT search. This is the EIT repetition time.

Returns

time in minutes

4.7.2.22 U8BIT* ACFG_GetEventContentTypes (U32BIT *country_code*)

Returns the event content types for the given country. This defines how the content type value broadcast as part of the EIT is to be interpreted.

Parameters

<i>country_code</i>	country code
---------------------	--------------

Returns

pointer to the content type array. The index of this array is level_1 from the event content descriptors in the EIT and its elements are of type ADB_EVENT_CONTENT, defined in [ap_dbacc.h](#)

4.7.2.23 U16BIT ACFG_GetFirstUnallocatedLcn (U32BIT *country_code*)

Returns the first LCN that should be used when assigning LCNs to services that don't appear in an LCN descriptor, or can't be assigned their desired LCN.

Parameters

<i>country_code</i>	country code
---------------------	--------------

Returns

LCN; 0 if country isn't valid

4.7.2.24 U16BIT ACFG_GetLastUnallocatedLcn (U32BIT *country_code*)

Returns the last LCN that should be used when assigning LCNs to services that don't appear in an LCN descriptor, or can't be assigned their desired LCN.

Parameters

<i>country_code</i>	country code
---------------------	--------------

Returns

LCN; 0 if country isn't valid

4.7.2.25 U16BIT ACFG_GetMaxSatelliteScanFreq (U32BIT *country_code*)

Returns the maximum frequency to be used when performing a satellite based frequency scan in the given country.

Parameters

<i>country_code</i>	country code
---------------------	--------------

Returns

frequency in MHz; 0 if undefined or country is invalid

4.7.2.26 U16BIT ACFG_GetMinSatelliteScanFreq (U32BIT *country_code*)

Returns the minimum frequency to be used when performing a satellite based frequency scan in the given country.

Parameters

<i>country_code</i>	country code
---------------------	--------------

Returns

frequency in MHz; 0 if undefined or country is invalid

4.7.2.27 U8BIT ACFG_GetNumCountries (void)

Returns the number of country configurations included in the DVB stack.

Returns

number of countries

4.7.2.28 U8BIT ACFG_GetNumDbLanguages (U32BIT *country_code*)

Returns the number of languages defined for the given country that can be used for audio and subtitles/teletext.

Parameters

<i>country_code</i>	country code
---------------------	--------------

Returns

number of available languages

4.7.2.29 U8BIT ACFG_GetNumRegions (U32BIT *country_code*)

Returns the number of regions in the given country.

Parameters

<i>country_</i> - <i>code</i>	country code
----------------------------------	--------------

Returns

number of regions

4.7.2.30 U8BIT ACFG_GetPrimaryAudioLangId (void)

Returns the primary audio language ID.

Returns

language id

4.7.2.31 U8BIT ACFG_GetPrimaryTextLangId (void)

Returns the primary DVB subtitle/teletext language ID.

Returns

language id

4.7.2.32 U32BIT ACFG_GetPrivateDataSpecifier (U32BIT *country_code*)

Returns the private data specifier value for the given country.

Parameters

<i>country_</i> - <i>code</i>	country code
----------------------------------	--------------

Returns

private data specifier value; 0 if one isn't defined for the country or country code isn't valid

4.7.2.33 **BOOLEAN** **ACFG_GetRegionCode** (**U32BIT** *country_code*, **U8BIT** *region_id*, **U8BIT** * *code_ptr*)

Returns the region code that identifies the given region.

Parameters

<i>country_code</i>	country code
<i>region_id</i>	region index
<i>code_ptr</i>	pointer to value in which the code will be returned

Returns

TRUE if the country and region indices are valid, FALSE otherwise

4.7.2.34 **U8BIT** **ACFG_GetRegionId** (**void**)

Returns the internal ID of the region the DVB is configured for.

Returns

region id

4.7.2.35 **BOOLEAN** **ACFG_GetRegionList** (**U32BIT** *country_code*, **U8BIT** *** *str_array_ptr*, **U8BIT** * *num_regions_ptr*)

Returns an array of pointers to strings containing the region names for the given country. The index into this array is the *region_id*. The array is allocated by this function and should be freed using **ACFG_ReleaseRegionList**.

Parameters

<i>country_code</i>	country code
<i>str_array_ptr</i>	pointer to an allocated array of static UTF-8 strings
<i>num_regions_ptr</i>	pointer to the number of regions in the returned array

Returns

TRUE if the country_code is valid

4.7.2.36 DBTidyFunc ACFG_GetSatelliteDBTidyFunction (U32BIT country_code)

Returns a pointer to a function that's used to tidy up the database following a DVB-S/S2 scan for the given country.

Parameters

<i>country_ - code</i>	country code
----------------------------	--------------

Returns

pointer to function, or NULL

4.7.2.37 AllocLcnFunc ACFG_GetSatelliteLcnFunction (U32BIT country_code)

Returns a pointer to a function that's used to assign LCNs following a DVB-S/S2 scan for the given country.

Parameters

<i>country_ - code</i>	country id
----------------------------	------------

Returns

pointer to function, or NULL

4.7.2.38 BOOLEAN ACFG_GetSatelliteScanDvbS2 (U32BIT country_code)

Returns whether DVB-S2 should be included when performing a satellite based frequency scan in the given country.

Parameters

<i>country_ - code</i>	country code
----------------------------	--------------

Returns

TRUE if DVB-S2 should be used, FALSE otherwise

4.7.2.39 U16BIT ACFG_GetSatelliteScanFreqInc (U32BIT country_code)

Returns the increment frequency to be used when performing a satellite based frequency scan in the given country.

Parameters

<i>country_ - code</i>	country code
----------------------------	--------------

Returns

frequency in MHz; 0 if undefined or country is invalid

4.7.2.40 U8BIT ACFG_GetSatelliteScanNumSymbolRates (U32BIT *country_code*)

Returns the number of items in the fixed array of symbol rates to be used when performing a satellite based frequency scan in the given country.

Parameters

<i>country_ - code</i>	country code
----------------------------	--------------

Returns

number of symbol rates; 0 if undefined or country is invalid

4.7.2.41 U16BIT* ACFG_GetSatelliteScanSymbolRates (U32BIT *country_code*)

Returns a fixed array of symbol rates to be used when performing a satellite based frequency scan in the given country.

Parameters

<i>country_ - code</i>	country code
----------------------------	--------------

Returns

array of symbol rates in Kbps; NULL if undefined or country is invalid

4.7.2.42 U8BIT ACFG_GetSecondaryAudioLangId (void)

Returns the secondary audio language ID.

Returns

language id

4.7.2.43 U8BIT ACFG_GetSecondaryTextLangId (void)

Returns the secondary DVB subtitle/teletext language ID.

Returns

language id

4.7.2.44 DBTidyFunc ACFG_GetTerrestrialDBTidyFunction (U32BIT *country_code*)

Returns a pointer to a function that's used to tidy up the database following a DVB-T/T2 scan for the given country.

Parameters

<i>country_code</i>	country code
---------------------	--------------

Returns

pointer to function, or NULL

4.7.2.45 AllocLcnFunc ACFG_GetTerrestrialLcnFunction (U32BIT *country_code*)

Returns a pointer to a function that's used to assign LCNs following a DVB-T/T2 scan for the given country.

Parameters

<i>country_code</i>	country id
---------------------	------------

Returns

pointer to function, or NULL

4.7.2.46 BOOLEAN ACFG_GetTerRfChannelTable (U32BIT *country_code*, ACFG_TER_RF_CHANNEL_DATA ** *rf_chan_data_ptr*, U16BIT * *num_entries_ptr*)

Returns a pointer to the channel table for a terrestrial tuner for the given country.

Parameters

<i>country_code</i>	country whose table is to be returned
<i>rf_chan_data_ptr</i>	returned pointer to the channel table
<i>num_entries_ptr</i>	number of entries in the returned channel table

Returns

TRUE if the country is valid and data is returned, FALSE otherwise

4.7.2.47 **BOOLEAN ACFG_GetWatershedTimes** (U32BIT *country_code*, U16BIT * *start_time*, U16BIT * *end_time*)

Get the start and end times of the watershed hours for a country.

Parameters

<i>country_code</i>	country code
<i>start_time</i>	returns the start time in minutes since midnight
<i>end_time</i>	returns the end time in minutes since midnight

Returns

TRUE if the start and end times are valid and have been returned, FALSE otherwise

4.7.2.48 **BOOLEAN ACFG_IsNordigCountry** (void)

Returns whether the current country requires Nordig compliance for SI.

Returns

TRUE if Nordig, FALSE otherwise

4.7.2.49 **void ACFG_ReleaseCountryList** (U8BIT ** *str_array*, U8BIT *num_countries*)

Frees a country list previously acquired using ACFG_GetCountryList.

Parameters

<i>str_array</i>	array to be freed
<i>num_countries</i>	number of items in the array

4.7.2.50 **void ACFG_ReleaseDbLangList** (U8BIT ** *str_array*, U8BIT *num_langs*)

Frees the language array previously acquired using ACFG_GetDbLangList.

Parameters

<i>str_array</i>	array of lang names to be freed
<i>num_langs</i>	number of names in the array

4.7.2.51 void ACFG_ReleaseRegionList (U8BIT ** *str_array*, U8BIT *num_regions*)

Frees the region array previously acquired using ACFG_GetRegionList.

Parameters

<i>str_array</i>	array to be freed
<i>num_regions</i>	number of items in the array

4.7.2.52 void ACFG_SetBackgroundSearchTime (U16BIT *start_time*, U16BIT *end_time*)

Sets the start and end times during which background searches are allowed run when in standby.

Parameters

<i>start_time</i>	start time in minutes since midnight
<i>end_time</i>	end time in minutes

4.7.2.53 void ACFG_SetBackgroundServiceSearch (BOOLEAN *enabled*)

Enables or disables the background service search when in standby.

Parameters

<i>enabled</i>	TRUE to enable, FALSE to disable
----------------	----------------------------------

4.7.2.54 void ACFG_SetBackgroundSSUSearch (BOOLEAN *enabled*)

Enables or disables the background SSU search when in standby.

Parameters

<i>enabled</i>	TRUE to enable, FALSE to disable
----------------	----------------------------------

4.7.2.55 void ACFG_SetCableChannelTable (ACFG_CAB_RF_CHANNEL_DATA * *cable_channel_data*, U16BIT *number_channels*)

Replaces the cable tuning table for the current country with the one provided.

Parameters

<i>cable_channel_data</i>	pointer to the new table to point
<i>number_channels</i>	number of entries in the channel table

4.7.2.56 void **ACFG_SetCableLcnFunction** (U32BIT *country_code*, AllocLcnFunc *func_ptr*)

Overrides an existing DVB-C LCN allocation function for the given country.

Parameters

<i>country_code</i>	country id
<i>pointer</i>	to function, can be NULL if no function is to be called

4.7.2.57 **BOOLEAN ACFG_SetCountry** (U32BIT *country_code*)

Sets the current country and sets default values for region and language.

Parameters

<i>country_code</i>	country code, as defined above (see COUNTRY_CODE_* defines)
---------------------	---

Returns

TRUE if country is known, FALSE otherwise

4.7.2.58 void **ACFG_SetCountryIds** (U32BIT *country_code*, U8BIT *region_id*, U8BIT *audio_lang_id*, U8BIT *sub_lang_id*)

Saves the configured country and changes settings related to the country.

Parameters

<i>country_code</i>	country code
<i>region_id</i>	ID of region in country, if appropriate
<i>audio_lang_id</i>	ID of language to be used for audio
<i>sub_lang_id</i>	ID of language to be used for subtitles/teletext

4.7.2.59 void **ACFG_SetPrimaryAudioLangId** (U8BIT *lang_id*)

Sets the primary language id to be used for audio. This is the language that will be chosen first, if available.

Parameters

<i>lang_id</i>	language id
----------------	-------------

4.7.2.60 void ACFG_SetPrimaryTextLangId (U8BIT *lang_id*)

Sets the primary language id to be used for teletext/subtitles.

Parameters

<i>lang_id</i>	language id
----------------	-------------

4.7.2.61 void ACFG_SetSatelliteLcnFunction (U32BIT *country_code*, AllocLcnFunc *func_ptr*)

Overrides an existing DVB-S/S2 LCN allocation function for the given country.

Parameters

<i>country_code</i>	country id
<i>pointer</i>	to function, can be NULL if no function is to be called

4.7.2.62 void ACFG_SetSecondaryAudioLangId (U8BIT *lang_id*)

Sets the secondary language id to be used for audio.

Parameters

<i>lang_id</i>	language id
----------------	-------------

4.7.2.63 void ACFG_SetSecondaryTextLangId (U8BIT *lang_id*)

Sets the secondary language id to be used for teletext/subtitles.

Parameters

<i>lang_id</i>	language id
----------------	-------------

4.7.2.64 void ACFG_SetTerrestrialLcnFunction (U32BIT *country_code*, AllocLcnFunc *func_ptr*)

Overrides an existing DVB-T/T2 LCN allocation function for the given country.

Parameters

<i>country_code</i>	country id
<i>pointer</i>	to function, can be NULL if no function is to be called

4.8 dvb/inc/ap_ci.h File Reference

Application level CI control functions.

Functions

- U8BIT [ACI_GetCamUpgradeMode](#) (void)
Return CAM upgrade option (Yes/No/Ask)
- void [ACI_SetCamUpgradeMode](#) (U8BIT upgrade_mode)
Sets the CAM upgrade option (Yes/No/Ask)
- U8BIT [ACI_GetOperatorSearchMode](#) (void)
Return the option for starting an operator profile search (yes/no/ask)
- void [ACI_SetOperatorSearchMode](#) (U8BIT search_mode)
Sets the operator profile search mode (Yes/No/Ask). This should be used by the app to control whether a search can be started, or the user should be asked first, or can't run.
- void [ACI_OperatorSearchRequired](#) (U32BIT module, BOOLEAN required)
Sets whether an operator profile search needs to be run.
- void [ACI_ScheduleOperatorSearch](#) (U32BIT module, U16BIT date, U8BIT hour, U8BIT min)
Sets the date/time that an operator search should be started for the given module.
- BOOLEAN [ACI_IsOperatorSearchRequired](#) (void)
Returns whether an operator profile search has been requested.
- U32BIT [ACI_GetOperatorSearchModule](#) (void)
Returns the module performing or requiring and operator search.
- BOOLEAN [ACI_GetFirstOperatorSearchModule](#) (U32BIT *module)
Checks all the CI+ profiles to see if any have requested an update search.
- BOOLEAN [ACI_GetFirstScheduledOperatorSearch](#) (U32BIT *module, U16BIT *date, U8BIT *hours, U8BIT *mins)
Checks all the CI+ profiles to find the one with the earliest scheduled search. All returned values are only valid if the function returns TRUE.
- BOOLEAN [ACI_StartOperatorSearchForModule](#) (U32BIT module)
Called by the app to start an operator profile search for the given module.
- void [ACI_OperatorSearchFinished](#) (void)
Called when an operator search has finished. Returns the live path to the service that was being viewed before the search started.
- BOOLEAN [ACI_AcquireCISlot](#) (U8BIT path, void *s_ptr)
Acquires a CI slot for the given path on the given service after releasing any slot already being used by the path.
- void [ACI_AcquireCISlotForRecording](#) (U8BIT path, void *s_ptr)
Acquires a CI slot for the recording path. This function may need to "steal" the CI slot from the live path (if they are not the same path).
- U8BIT [ACI_FindCISlotForService](#) (void *serv_ptr)
Looks for a CAM that's able to descramble the given service and returns its slot id.
- S32BIT [ACI_ReadPinForSlot](#) (U8BIT slot_id)

Checks if a pin has been saved for the CAM in the given slot and returns it.

- **BOOLEAN [ACI_WritePinForSlot](#)** (U8BIT slot_id, S32BIT pin)
Saves the given pin associated with the CAM in the given slot. If the pin for this CAM has previously been saved then the saved value will be updated. If all entries are used then the oldest will be overwritten.
- **BOOLEAN [ACI_HandlePrivateTimer](#)** (U32BIT timer_handle)
Checks whether the given timer is associated with any of the CI+ profiles and starts the operator search if possible.
- **BOOLEAN [ACI_SetSecureRouting](#)** (U8BIT path)
Ensures the TS is routed securely for CI+, either by setting the TS to pass through if a CI slot contains a CI+ CAM or bypass it if it doesn't.
- **void [ACI_CISlotStatusChanged](#)** (U8BIT slot_id)
Re-evaluate current state following a change related to CI slots.
- **void [ACI_UsageRulesStatusChanged](#)** (U8BIT path)
Re-evaluate current state following a change related to usage rules.
- **void [ACI_ContentControlKeysChanged](#)** (void)
Apply new content control keys on all active paths.
- **void [ACI_ProgramMapTableChanged](#)** (U8BIT *pmt)
Handle PMT change.
- **void [ACI_ApplyCCKeys](#)** (U8BIT slot_id)
Handle PMT change.
- **BOOLEAN [ACI_IsTrustedPath](#)** (U8BIT path)
The given decode path is only trusted if it doesn't include a CI slot or the CI slot contains a CI+ CAM.
- **void [ACI_RecordLicenceChanged](#)** (U8BIT path, U8BIT slot_id)
Handles a CICAM licence update when recording, which results in the licence being saved with the recording for use during playback.
- **void [ACI_PlaybackLicenceChanged](#)** (U8BIT slot_id)
Handles a CICAM licence update during playback, which may result in the app being informed that the licence is no longer valid for the recording.
- **void [ACI_RecordPin](#)** (U8BIT path, U8BIT slot_id)
Handles a CICAM pin update when recording, which results in the pin info being saved with the recording for use during playback.

4.8.1 Detailed Description

Application level CI control functions.

Date

February 2011

Author

Ocean Blue

4.8.2 Function Documentation

4.8.2.1 BOOLEAN ACI_AcquireCISlot (U8BIT *path*, void * *s_ptr*)

Acquires a CI slot for the given path on the given service after releasing any slot already being used by the path.

Parameters

<i>path</i>	decode path
<i>s_ptr</i>	service to be used on the path

Returns

TRUE if a slot is acquired, FALSE otherwise

4.8.2.2 void ACI_AcquireCISlotForRecording (U8BIT *path*, void * *s_ptr*)

Acquires a CI slot for the recording path. This function may need to "steal" the CI slot from the live path (if they are not the same path).

Parameters

<i>path</i>	decode path for recording
<i>s_ptr</i>	service to be used on the path

4.8.2.3 void ACI_ApplyCCKeys (U8BIT *slot_id*)

Handle PMT change.

Parameters

<i>pmt</i>	the new PMT
------------	-------------

4.8.2.4 void ACI_CISlotStatusChanged (U8BIT *slot_id*)

Re-evaluate current state following a change related to CI slots.

Parameters

<i>slot_id</i>	slot on which the status has changed
----------------	--------------------------------------

4.8.2.5 U8BIT ACI_FindCISlotForService (void * *serv_ptr*)

Looks for a CAM that's able to descramble the given service and returns its slot id.

Parameters

<i>serv_ptr</i>	service
-----------------	---------

Returns

slot id, if CAM is found, or INVALID_RES_ID if not

4.8.2.6 U8BIT ACI_GetCamUpgradeMode (void)

Return CAM upgrade option (Yes/No/Ask)

Returns

U8BIT the cams current upgrade mode

4.8.2.7 BOOLEAN ACI_GetFirstOperatorSearchModule (U32BIT * module)

Checks all the CI+ profiles to see if any have requested an update search.

Parameters

<i>module</i>	pointer to return the handle of the first module requiring an update Only valid if the function returns TRUE
---------------	--

Returns

TRUE if a profile needs updating, FALSE otherwise

4.8.2.8 BOOLEAN ACI_GetFirstScheduledOperatorSearch (U32BIT * module, U16BIT * date, U8BIT * hours, U8BIT * mins)

Checks all the CI+ profiles to find the one with the earliest scheduled search. All returned values are only valid if the function returns TRUE.

Parameters

<i>module</i>	pointer to return the handle of the first module requiring an update
<i>date</i>	pointer to return the date code of the search
<i>hours</i>	pointer to return the hour of the search
<i>mins</i>	pointer to return the minute of the search

Returns

TRUE if a scheduled search is found for a profile, FALSE otherwise

4.8.2.9 U8BIT ACI_GetOperatorSearchMode (void)

Return the option for starting an operator profile search (yes/no/ask)

Returns

The current setting

4.8.2.10 U32BIT ACI_GetOperatorSearchModule (void)

Returns the module performing or requiring and operator search.

Returns

module

4.8.2.11 BOOLEAN ACI_HandlePrivateTimer (U32BIT *timer_handle*)

Checks whether the given timer is associated with any of the CI+ profiles and starts the operator search if possible.

Parameters

<i>timer_handle</i>	timer handle
---------------------	--------------

Returns

TRUE if the timer is associated with a profile, FALSE otherwise

4.8.2.12 BOOLEAN ACI_IsOperatorSearchRequired (void)

Returns whether an operator profile search has been requested.

Returns

TRUE if a search has been requested, FALSE otherwise

4.8.2.13 BOOLEAN ACI_IsTrustedPath (U8BIT *path*)

The given decode path is only trusted if it doesn't include a CI slot or the CI slot contains a CI+ CAM.

Parameters

<i>path</i>	decode path
-------------	-------------

Returns

TRUE if the decode path is trusted, FALSE otherwise

4.8.2.14 void ACI_OperatorSearchRequired (U32BIT *module*, BOOLEAN *required*)

Sets whether an operator profile search needs to be run.

Parameters

<i>module</i>	module requesting or cancelling an operator search
<i>required</i>	TRUE if a search is required, FALSE otherwise

4.8.2.15 void ACI_PlaybackLicenceChanged (U8BIT *slot_id*)

Handles a CICAM licence update during playback, which may result in the app being informed that the licence is no longer valid for the recording.

Parameters

<i>slot_id</i>	CI slot
----------------	---------

4.8.2.16 void ACI_ProgramMapTableChanged (U8BIT * *pmt*)

Handle PMT change.

Parameters

<i>pmt</i>	the new PMT
------------	-------------

4.8.2.17 S32BIT ACI_ReadPinForSlot (U8BIT *slot_id*)

Checks if a pin has been saved for the CAM in the given slot and returns it.

Parameters

<i>slot_id</i>	CI slot
----------------	---------

Returns

pin value, or -1 if there's no CAM in the slot or no pin has been saved

4.8.2.18 void ACI_RecordLicenceChanged (U8BIT *path*, U8BIT *slot_id*)

Handles a CICAM licence update when recording, which results in the licence being saved with the recording for use during playback.

Parameters

<i>path</i>	decode path
<i>slot_id</i>	CI slot

4.8.2.19 void ACI_RecordPin (U8BIT *path*, U8BIT *slot_id*)

Handles a CICAM pin update when recording, which results in the pin info being saved with the recording for use during playback.

Parameters

<i>path</i>	decode path
<i>slot_id</i>	CI slot

4.8.2.20 void **ACI_ScheduleOperatorSearch** (U32BIT *module*, U16BIT *date*, U8BIT *hour*, U8BIT *min*)

Sets the date/time that an operator search should be started for the given module.

Parameters

<i>module</i>	module requesting an operator search
<i>date</i>	UTC MJD date code
<i>hour</i>	UTC hour
<i>min</i>	UTC minute

4.8.2.21 void **ACI_SetCamUpgradeMode** (U8BIT *upgrade_mode*)

Sets the CAM upgrade option (Yes/No/Ask)

Parameters

<i>upgrade_mode</i>	cam upgrade mode
---------------------	------------------

4.8.2.22 void **ACI_SetOperatorSearchMode** (U8BIT *search_mode*)

Sets the operator profile search mode (Yes/No/Ask). This should be used by the app to control whether a search can be started, or the user should be asked first, or can't run.

Parameters

<i>search_mode</i>	search mode
--------------------	-------------

4.8.2.23 BOOLEAN **ACI_StartOperatorSearchForModule** (U32BIT *module*)

Called by the app to start an operator profile search for the given module.

Parameters

<i>module</i>	operator module
---------------	-----------------

Returns

TRUE if the search is started, FALSE otherwise

4.8.2.24 void ACI_UsageRulesStatusChanged (U8BIT *path*)

Re-evaluate current state following a change related to usage rules.

Parameters

<i>path</i>	decode path
-------------	-------------

4.8.2.25 BOOLEAN ACI_WritePinForSlot (U8BIT *slot_id*, S32BIT *pin*)

Saves the given pin associated with the CAM in the given slot. If the pin for this CAM has previously been saved then the saved value will be updated. If all entries are used then the oldest will be overwritten.

Parameters

<i>slot_id</i>	CI slot
<i>pin</i>	pin value to be saved

Returns

TRUE on success, FALSE otherwise

4.9 dvb/inc/ap_cntrl.h File Reference

Application stb layer control.

```
#include "stbhwav.h" #include "vtctype.h" #include "stbdpc.-
h" #include "ap_si.h" Include dependency graph for ap_cntrl.h:
```

Data Structures

- struct [S_MANUAL_ANA_TUNING_PARAMS](#)
- struct [S_MANUAL_TERR_TUNING_PARAMS](#)
- struct [S_MANUAL_CABLE_TUNING_PARAMS](#)
- struct [S_MANUAL_SAT_TUNING_PARAMS](#)
- struct [S_MANUAL_TUNING_PARAMS](#)
- struct [S_ACTL_OWNER_INFO](#)

Typedefs

- typedef void(* [DSM_FILE_CALLBACK](#))(void *user_data, U8BIT *file_data, U32-BIT data_size)

Prototype for function that's called when a file requested from a DSM-CC carousel is received.

Enumerations

- enum **E_ACTL_SI_SRCH_REQD** { **ACTL_SI_NO_SEARCH**, **ACTL_SI_CHANNEL_SEARCH**, **ACTL_SI_CHANNEL_SEARCH_NO_NIT**, **ACTL_SI_STARTUP_SEARCH**, **ACTL_SI_EVENT_SCHED_SEARCH**, **ACTL_SI_DVB_SSU_SEARCH**, **ACTL_SI_TOT_SEARCH**, **ACTL_SI_USER_DEFINED** }
- enum **E_ACTL_SEARCH_TYPE** { **ACTL_FREQ_SEARCH**, **ACTL_NETWORK_SEARCH** }
- enum **E_ACTL_AV_MODE** { **ACTL_STANDBY_MODE**, **ACTL_STANDBY_SCART_MODE**, **ACTL_TV_MODE**, **ACTL_SCART_MODE**, **ACTL_INT_DVD_MODE** }

Functions

- **BOOLEAN** [ACTL_StartServiceSearch](#) (**E_STB_DP_SIGNAL_TYPE** tuner_type, **E_ACTL_SEARCH_TYPE** type)
Entry point for starting a service search for a full retune or to update the existing service lineup.
- **BOOLEAN** [ACTL_IsTargetRegionRequired](#) (void)
When the search has completed, this function should be called to see whether the target region UI should be presented. This is only needed for DVB-T/T2 searches.
- **void** [ACTL_StopServiceSearch](#) (void)
Function to stop the service search before it completes.
- **void** [ACTL_CompleteServiceSearch](#) (void)
Function to complete a service search after an operation has completed that allows the LCNs to be assigned (e.g. after the target region has been selected).
- **BOOLEAN** [ACTL_StartStartupSearch](#) (void)
Function to start the startup search when booting from cold. This search checks the validity of the services contained in the database. It's only valid for DVB-T/T2 and DVB-C.
- **BOOLEAN** [ACTL_StartSSUSearch](#) (void)
Function to start a search to see if there's an SSU available.
- **BOOLEAN** [ACTL_StartTotSearch](#) (void)
Function to start a TOT search which is required to set the system clock when starting from power off if a real-time clock isn't available.
- **void** [ACTL_StopTotSearch](#) (void)
Function to stop the TOT search before it completes.
- **BOOLEAN** [ACTL_StartEitSearch](#) (void)
Entry point for starting an EIT search.
- **void** [ACTL_StopEitSearch](#) (void)
Function to stop the EIT search before it completes.
- **BOOLEAN** [ACTL_IsSearchComplete](#) (void)
Returns TRUE if current search has finished. This works for any of the available types of searches.
- **U8BIT** [ACTL_GetSearchProgress](#) (void)

Returns search progress as a percentage. This works for any of the available types of searches, but the scope for providing progress on anything other than the service searches is fairly limited.

- U8BIT [ACTL_GetServiceSearchPath](#) (void)

Returns path used by currently running search. This works for any of the available types of searches, but the scope for providing path on anything other than the service searches is fairly limited.
- BOOLEAN [ACTL_StartManualSearchById](#) (E_STB_DP_SIGNAL_TYPE tuner_type, U16BIT chan_id, BOOLEAN start_search)

Start a service search on, or just tune to, a transport, using chan_id as an index into the country's tuning table.
- BOOLEAN [ACTL_StartManualSearch](#) (E_STB_DP_SIGNAL_TYPE tuner_type, [S_MANUAL_TUNING_PARAMS](#) *tuning_params, E_ACTL_SEARCH_TYPE type)

Start a service search on, or just tune to, the transport defined by the given tuning parameters.
- void [ACTL_FinishManualSearch](#) (void)

Function to be called when a manual service search has completed, or is being stopped.
- BOOLEAN [ACTL_StartSSUUpdate](#) (void *t_ptr)

Starts an SSU update on the given transport. The update may still fail if the transport is later found not to contain an update for the system. If the update isn't mandatory then the process must be continued to actually perform the update.
- void [ACTL_FinishSSUUpdate](#) (BOOLEAN do_update)

Once an update has been found or not then this function must be called to continue or terminate the update.
- U8BIT [ACTL_AcquirePathForService](#) (void *s_ptr, BOOLEAN with_decoders, BOOLEAN for_recording, [S_ACTL_OWNER_INFO](#) *owner)

Acquires a decode path suitable for tuning to the given service.
- void [ACTL_ReleaseLivePathForService](#) (void *s_ptr, E_STB_DP_RES_OWNER owner)

Finds the path being used to view the given service and releases it.
- BOOLEAN [ACTL_AcquirePathOwnership](#) (U8BIT path, [S_ACTL_OWNER_INFO](#) *owner_info)

Attempts to take ownership of the given path (used by CI+)
- BOOLEAN [ACTL_ReleasePathOwnership](#) (U8BIT path, E_STB_DP_RES_OWNER owner)

Releases ownership of the path, and frees any associated data, if the given owner is the path's owner (used by CI+)
- BOOLEAN [ACTL_CanServiceBeViewed](#) (void *s_ptr)

Checks whether there's a tuner available to view the given service. This takes into account whether tuners are being used for recording or have been acquired by CI+.
- void * [ACTL_GetCurrentSatellite](#) (U8BIT path)

Returns the current satellite being used by the given decode path.
- void [ACTL_TuneToTransport](#) (U8BIT path, [S_ACTL_OWNER_INFO](#) *owner_info, void *t_ptr, E_ACTL_SI_SRCH_REQD reqd_si, BOOLEAN relock_on)

Tunes to the given transport and sets the type of SI monitoring that will be started when the tuning completes.

- U8BIT [ACTL_TuneToService](#) (U8BIT path, [S_ACTL_OWNER_INFO](#) *owner_info, void *s_ptr, BOOLEAN override_lock, BOOLEAN for_live)

Starts the process of tuning to the specified service. If the service is to be tuned on the live path, then path should be passed as INVALID_RES_ID to ensure that any CI+ resource handling is performed.

- BOOLEAN [ACTL_TuneUsingDSD](#) (U8BIT path, SI_DELIVERY_SYS_DESC_T-
TYPE dsd_type, [SI_DELIVERY_SYS_DESC](#) *dsd, U16BIT service_id, E_ACTL_
SI_SRCH_REQD reqd_si)

Starts the process of tuning to a given transport or service that's defined by the given delivery system descriptor. If the DSD defines a transport that doesn't currently exist, then one will be created, and if a service_id is given and a service with this ID doesn't exist on the transport, then one will be created, but will be marked as hidden and non-selectable so it won't appear to a user.

- BOOLEAN [ACTL_IsTuned](#) (U8BIT path)

Returns whether the given path is currently tuned.

- void [ACTL_TuneOff](#) (U8BIT path)

Stops tuning on the given path.

- void [ACTL_DecompileOff](#) (U8BIT path)

Stops decoding on the given path.

- void [ACTL_ReTuneAudio](#) (void)

Stops and restarts audio decoding on the live path. This may be required due to a change in language preferences, or some other audio setting.

- void [ACTL_ReTuneSubtitles](#) (void)

Stops and restarts subtitle decoding on the live path. This may be required due to a change in language preferences, or some other setting.

- BOOLEAN [ACTL_HasDecodingStarted](#) (U8BIT path)

Returns whether decoding has been started on the given path.

- BOOLEAN [ACTL_IsDecodingLocked](#) (U8BIT path)

Returns whether decoding is locked, due to parental locking, on the given path.

- void [ACTL_ReleaseChannelLock](#) (void)

Releases the lock on a channel after decoding has been blocked due to the service being parental locked, after which decoding will be started.

- BOOLEAN [ACTL_TuneToRfChanArrayEntry](#) (U8BIT path, U16BIT id, E_ACTL_
SI_SRCH_REQD reqd_si, BOOLEAN relock_on)

Tunes to the specified rf channel array entry for DVB-T and DVB-C.

- BOOLEAN [ACTL_TuneToRfChanArrayAnalogEntry](#) (U8BIT path, U16BIT id)

Tunes to the specified rf channel array entry in analogue mode.

- BOOLEAN [ACTL_TuneToUserDefinedParams](#) (U8BIT path, [S_MANUAL_TUNING_PARAMS](#) *tuning_params, E_ACTL_SI_SRCH_REQD reqd_si, BOOLEAN relock_on)

Tunes to the given set of tuning parameters.

- void [ACTL_EnterStandby](#) (void)

Puts DVB into standby mode. It will continue to monitor SI for recordings, SSU updates, etc., unless it goes into low power standby.

- void [ACTL_LeaveStandby](#) (void *s_ptr, BOOLEAN tune_to_service)
Brings the DVB out of standby mode.
- void [ACTL_SetStandbyState](#) (BOOLEAN state)
Reports the standby state to the A/V output controller.
- void [ACTL_SetStandbyVCRActive](#) (void)
Reports standby state to the a/v output control state machine.
- void [ACTL_SetVideoWindow](#) (S16BIT win_x, S16BIT win_y, U16BIT win_width, U16BIT win_height)
Sets the video window to the size specified. Coordinates are relative to the screen resolution.
- void [ACTL_SetTVAspectRatio](#) (E_STB_AV_ASPECT_RATIO aspect_ratio)
Used to set the TV aspect ratio.
- E_STB_AV_ASPECT_RATIO [ACTL_GetTVAspectRatio](#) (void)
Returns the current TV aspect ratio.
- void [ACTL_SetTVAspectMode](#) (E_STB_AV_ASPECT_MODE aspect_mode)
Used to set the TV aspect mode which defines how the video will be displayed based on the aspect ratio of the TV and video, along with some other factors.
- E_STB_AV_ASPECT_MODE [ACTL_GetTVAspectMode](#) (void)
Returns the current TV aspect mode.
- void [ACTL_SetTVAspectConversion](#) (E_STB_AV_ASPECT_RATIO aspect_ratio, E_STB_AV_ASPECT_MODE aspect_mode)
Used to set the aspect conversion applied to the video based on the TV aspect ratio and aspect mode to be applied. The given settings will also be saved.
- E_STB_AV_VIDEO_FORMAT [ACTL_GetActualVideoMode](#) (U16BIT *width, - U16BIT *height)
Reads the saved video format and returns the best mode available if it's set to AUTO or is invalid for the currently connected TV.
- void [ACTL_HDMIConnected](#) (void)
Checks that the selected HDMI resolution mode is supported and, if not, chooses the best one available. This function is called by the event task when the HDMI connected event is received.
- void [ACTL_HDMIDisconnected](#) (void)
Sets flag to indicate HDMI is now disconnected. This function is called by the event task when the HDMI disconnected event is received.
- BOOLEAN [ACTL_IsHDMIConnected](#) (void)
Returns whether the HDMI is connected or not.
- U16BIT [ACTL_GetHDMIResolutions](#) (E_STB_AV_VIDEO_FORMAT **video_formats, U16BIT *current_index)
Returns an array of valid HDMI resolutions and the index of the currently selected format. The final entry in each array will be the auto video format.
- void [ACTL_UpdateVideoMode](#) (E_STB_AV_ASPECT_RATIO aspect, BOOLEAN force)
Update video mode sets Voyager resolution as well as platform. Also, stops subtitles during the operation. If HDMI resolution mode is not supported, it chooses the best one available.
- void [ACTL_ApplyHDCP](#) (void *s_ptr)

Checks content protection requirements for the current event on the given service to determine whether HDCP has to be used. The HDMI output will be enabled/disabled accordingly.

- E_ACTL_AV_MODE [ACTL_GetAvModeStatus](#) (void)
Returns av_mode.
- U16BIT [ACTL_GetNumRfChanArrayEntries](#) (E_STB_DP_SIGNAL_TYPE tuner_type)
Returns the number of entries in the rf channel table.
- U8BIT * [ACTL_GetRfChanName](#) (E_STB_DP_SIGNAL_TYPE tuner_type, U16-BIT id)
Returns a pointer to the channel name.
- U8BIT * [ACTL_GetRfNameFromFreq](#) (E_STB_DP_SIGNAL_TYPE tuner_type, U32BIT freq_hz)
Returns the rf name appropriate to the frequency specified.
- U32BIT [ACTL_GetRfChanFreqHz](#) (E_STB_DP_SIGNAL_TYPE tuner_type, U16BIT id)
Returns a pointer to the channel name.
- U16BIT [ACTL_GetRfSymbolRate](#) (E_STB_DP_SIGNAL_TYPE tuner_type, U16-BIT id)
Returns the channel symbol rate.
- U8BIT [ACTL_GetRfModulation](#) (E_STB_DP_SIGNAL_TYPE tuner_type, U16BIT id)
Returns the modulation mode.
- E_STB_DP_TTYPE [ACTL_GetTerRfChanType](#) (U16BIT id)
Returns the terrestrial type (T or T2) for the given channel id.
- void [ACTL_EnableCiModule](#) (void)
Enables CI module.
- void [ACTL_DisableCiModule](#) (void)
Disables CI module.
- BOOLEAN [ACTL_IsCiUiRequired](#) (void)
Returns state of ci_ui_required flag.
- BOOLEAN [ACTL_GetDecodePausedState](#) (U8BIT path)
Gets user paused flag value.
- BOOLEAN [ACTL_StartSubtitles](#) (void)
Starts subtitle processing and display if the current service has valid subtitle data. If DVB subtitles aren't available, teletext will be used if available.
- void [ACTL_PauseSubtitles](#) (void)
Disables the display of subtitles but processing continues.
- void [ACTL_ResumeSubtitles](#) (void)
Resumes the display of subtitles after they've been paused.
- void [ACTL_StopSubtitles](#) (void)
Stops subtitles from being displayed and processed.
- BOOLEAN [ACTL_AreSubtitlesDisplayed](#) (void)
Returns whether subtitles are being displayed.
- BOOLEAN [ACTL_AreSubtitlesStarted](#) (void)

- Returns whether subtitles have been started, even if they aren't being displayed.*
- **BOOLEAN** [ACTL_IsAudioDescriptionOn](#) (void)
 - Returns whether audio description has been turned on.*
- **BOOLEAN** [ACTL_StartAudioDescription](#) (U8BIT path)
 - Starts decoding an audio description stream, if available, on the given path. If AD isn't currently available it will be marked as enabled and started when it becomes available.*
- **void** [ACTL_StopAudioDescription](#) (U8BIT path)
 - Stops AD decoding.*
- **void** [ACTL_SetADVolume](#) (U8BIT volume)
 - Sets the AD volume.*
- **void** [ACTL_SetParentalControl](#) (BOOLEAN enabled)
 - Enables or disables parental control. This enables or disables locking on a per channel basis. If an age has previously been set, this will override the setting, effectively turning off age related blockign of channels.*
- **BOOLEAN** [ACTL_ParentalControlEnabled](#) (void)
 - Returns whether parental control is enabled. This will also return TRUE if parental control has been set to a valid age.*
- **void** [ACTL_SetParentalControlAge](#) (U8BIT age)
 - Sets the age (valid values 4-18) at which parental control will be will be applied. If the age is invalid, no change will be made to the current setting.*
- **U8BIT** [ACTL_GetParentalControlAge](#) (void)
 - Returns the current age set for parental control. 0 will be returned if parental control is disabled or no age is set.*
- **void** [ACTL_ApplyParentalControl](#) (U8BIT path, void *s_ptr)
 - Checks the parental control for the current event on the given service to determine whether decoding should be locked.*
- **void** [ACTL_SetupPlayback](#) (void)
 - Starts the SI to acquire the PMT and fill the ip service in.*
- **void** [ACTL_StartPlayback](#) (void *s_ptr)
 - Start streaming the specified service.*
- **U8BIT** [ACTL_GetActivePath](#) (void)
 - Returns the active path (live, playback, etc...), i.e. the one using the decoders.*
- **void** [ACTL_StartDecoding](#) (U8BIT path, void *s_ptr)
 - Sets up and starts decoding for the given service. This is used for PVR playback, but could also be used to start decoding for other non-broadcast services.*
- **void** [ACTL_SetMhegAVControl](#) (BOOLEAN control)
 - Sets the MHEG5 audio volume adjust to be applied.*
- **void** [ACTL_SetVolumeAdjustment](#) (S8BIT scaling)
 - Sets the MHEG5 audio volume adjust to be applied.*
- **U8BIT** [ACTL_GetVolume](#) (void)
 - Returns the current audio volume.*
- **U8BIT** [ACTL_SetVolume](#) (U8BIT volume)
 - Sets the main audio volume and returns the new volume.*
- **U8BIT** [ACTL_ChangeVolume](#) (S8BIT volume_change)
 - Changes the main audio volume and returns the new volume.*

- void [ACTL_SetMute](#) (BOOLEAN mute)
Sets the audio mute state.
- BOOLEAN [ACTL_ToggleMute](#) (void)
Toggles the current mute state and returns the new mute setting.
- BOOLEAN [ACTL_IsMuted](#) (void)
Returns the muted state of the audio.
- BOOLEAN [ACTL_SetActiveProfile](#) (void *profile)
Sets the current profile. (CI+ only) If the profile is being set to CI+ profile and the necessary CAM isn't present, then setting the profile will fail.
- void [ACTL_InitialiseAppControl](#) (void)
Control system initialisation.
- void [ACTL_ActionEvent](#) (U32BIT event, void *event_data)
Actions external events.
- BOOLEAN [ACTL_CheckLiveServiceChange](#) (void)
Checks whether the service tuned to on the live path has changed and informs the UI if it has.
- BOOLEAN [ACTL_HandlePrivateTimerEvent](#) (U32BIT timer_handle)
Handles all the private timer events.
- void [ACTL_SetAnalogChanIdString](#) (U8BIT *str_ptr)
- void [ACTL_AllowAnalogVideo](#) (BOOLEAN allow_analog_video)
Enables or disables analog video display.

4.9.1 Detailed Description

Application stb layer control.

Date

18/03/2003

4.9.2 Typedef Documentation

4.9.2.1 typedef void(* [DSM_FILE_CALLBACK](#))(void *user_data, U8BIT *file_data, U32BIT data_size)

Prototype for function that's called when a file requested from a DSM-CC carousel is received.

Parameters

<i>user_data</i>	user defined data passed when the file was requested
<i>file_data</i>	data
<i>data_size</i>	number of bytes of data

4.9.3 Function Documentation

4.9.3.1 U8BIT ACTL_AcquirePathForService (void * *s_ptr*, BOOLEAN *with_decoders*, BOOLEAN *for_recording*, S_ACTL_OWNER_INFO * *owner*)

Acquires a decode path suitable for tuning to the given service.

Parameters

<i>s_ptr</i>	the service the decode path will be used for
<i>with_decoders</i>	TRUE if decoders are to be acquired with the path, FALSE otherwise
<i>for_recording</i>	TRUE if the path will be used for recording
<i>owner_info</i>	owner of the acquired path, should be NULL for live TV (used by CI+)

Returns

ID of path or INVALID_RES_ID on failure

4.9.3.2 BOOLEAN ACTL_AcquirePathOwnership (U8BIT *path*, S_ACTL_OWNER_INFO * *owner_info*)

Attempts to take ownership of the given path (used by CI+)

Parameters

<i>path</i>	decode path to be owned
<i>owner_info</i>	owner info for the path

Returns

TRUE if the ownership is set, FALSE otherwise

4.9.3.3 void ACTL_ActionEvent (U32BIT *event*, void * *event_data*)

Actions external events.

Parameters

<i>event</i>	the event to be handled
--------------	-------------------------

4.9.3.4 void ACTL_AllowAnalogVideo (BOOLEAN *allow_analog_video*)

Enables or disables analog video display.

Parameters

<i>allow_analog_video</i>	if TRUE analog video is enabled, otherwise disabled
---------------------------	---

4.9.3.5 void ACTL_ApplyHDCP (void * *s_ptr*)

Checks content protection requirements for the current event on the given service to determine whether HDCP has to be used. The HDMI output will be enabled/disabled accordingly.

Parameters

<i>s_ptr</i>	service
--------------	---------

4.9.3.6 void ACTL_ApplyParentalControl (U8BIT *path*, void * *s_ptr*)

Checks the parental control for the current event on the given service to determine whether decoding should be locked.

Parameters

<i>path</i>	decode path
<i>s_ptr</i>	service

4.9.3.7 BOOLEAN ACTL_CanServiceBeViewed (void * *s_ptr*)

Checks whether there's a tuner available to view the given service. This takes into account whether tuners are being used for recording or have been acquired by CI+.

Parameters

<i>s_ptr</i>	service being checked
--------------	-----------------------

Returns

TRUE if service can be viewed, FALSE otherwise

4.9.3.8 U8BIT ACTL_ChangeVolume (S8BIT *volume_change*)

Changes the main audio volume and returns the new volume.

Parameters

<i>volume_change</i>	signed value giving change in volume to be applied
----------------------	--

Returns

The new volume on a scale of 0-100.

4.9.3.9 BOOLEAN ACTL_CheckLiveServiceChange (void)

Checks whether the service tuned to on the live path has changed and informs the UI if it has.

Returns

TRUE if the service has changed, FALSE otherwise

4.9.3.10 void ACTL_DecodeOff (U8BIT *path*)

Stops decoding on the given path.

Parameters

<i>path</i>	decode path
-------------	-------------

4.9.3.11 void ACTL_FinishSSUUpdate (BOOLEAN *do_update*)

Once an update has been found or not then this function must be called to continue or terminate the update.

Parameters

<i>do_update</i>	TRUE if the update is to be downloaded and installed, FALSE otherwise
------------------	---

4.9.3.12 U8BIT ACTL_GetActivePath (void)

Returns the active path (live, playback, etc...), i.e. the one using the decoders.

Returns

active path

4.9.3.13 E_STB_AV_VIDEO_FORMAT ACTL_GetActualVideoMode (U16BIT * *width*, U16BIT * *height*)

Reads the saved video format and returns the best mode available if it's set to AUTO or is invalid for the currently connected TV.

Parameters

<i>width</i>	returns the new video width if saved mode isn't valid, can be NULL
<i>height</i>	returns the new video height if saved mode isn't valid, can be NULL

Returns

current video mode, or best video mode available

4.9.3.14 E_ACTL_AV_MODE ACTL_GetAvModeStatus (void)

Returns av_mode.

Returns

Scart AV status

4.9.3.15 void* ACTL_GetCurrentSatellite (U8BIT *path*)

Returns the current satellite being used by the given decode path.

Parameters

<i>path</i>	decode path
-------------	-------------

Returns

Pointer to satellite

4.9.3.16 BOOLEAN ACTL_GetDecodePausedState (U8BIT *path*)

Gets user paused flag value.

path decode path

Returns

TRUE if the subtitles are in paused mode, FALSE otherwise

4.9.3.17 U16BIT ACTL_GetHDMIResolutions (E_STB_AV_VIDEO_FORMAT **
video_formats, U16BIT * *current_index*)

Returns an array of valid HDMI resolutions and the index of the currently selected format. The final entry in each array will be the auto video format.

Parameters

<i>video_ - formats</i>	pointer to an array of video formats.
<i>current_ - index</i>	used to return index of currently selected video format

Returns

Size of returned arrays, will always be at least 1

4.9.3.18 U16BIT ACTL_GetNumRfChanArrayEntries (E_STB_DP_SIGNAL_TYPE
tuner.type)

Returns the number of entries in the rf channel table.

Parameters

<i>tuner_type</i>	tuner type
-------------------	------------

Returns

number of entries in the channel table for the present country

4.9.3.19 U8BIT ACTL_GetParentalControlAge (void)

Returns the current age set for parental control. 0 will be returned if parental control is disabled or no age is set.

Returns

age in the range 4-18, or 0

4.9.3.20 U32BIT ACTL_GetRfChanFreqHz (E_STB_DP_SIGNAL_TYPE *tuner_type*, U16BIT *id*)

Returns a pointer to the channel name.

Parameters

<i>tuner_type</i>	tuner type
<i>id</i>	index in the channel table for the current country

Returns

pointer to the const name string

4.9.3.21 U8BIT* ACTL_GetRfChanName (E_STB_DP_SIGNAL_TYPE *tuner_type*, U16BIT *id*)

Returns a pointer to the channel name.

Parameters

<i>tuner_type</i>	tuner type
<i>id</i>	index in the channel table for the current country

Returns

pointer to the const name string

4.9.3.22 U8BIT ACTL_GetRfModulation (E_STB_DP_SIGNAL_TYPE *tuner_type*, U16BIT *id*)

Returns the modulation mode.

Parameters

<i>tuner_type</i>	tuner type
<i>id</i>	index in the channel table for the current country

Returns

Modulation mode (E_STB_DP_TMODE or E_STB_DP_CMODE depending on the tuner type)

4.9.3.23 U8BIT* **ACTL_GetRfNameFromFreq** (E_STB_DP_SIGNAL_TYPE *tuner_type*, U32BIT *freq_hz*)

Returns the rf name appropriate to the frequency specified.

Parameters

<i>tuner_type</i>	tuner type
<i>freq_hz</i>	frequency in hz

Returns

pointer to the const name string, or NULL if not found

4.9.3.24 U16BIT **ACTL_GetRfSymbolRate** (E_STB_DP_SIGNAL_TYPE *tuner_type*, U16BIT *id*)

Returns the channel symbol rate.

Parameters

<i>tuner_type</i>	tuner type
<i>id</i>	index in the channel table for the current country

Returns

symbol rate of selected id

4.9.3.25 U8BIT **ACTL_GetSearchProgress** (void)

Returns search progress as a percentage. This works for any of the available types of searches, but the scope for providing progress on anything other than the service searches is fairly limited.

Returns

Percent complete, 0 if not started

4.9.3.26 U8BIT ACTL_GetServiceSearchPath (void)

Returns path used by currently running search. This works for any of the available types of searches, but the scope for providing path on anything other than the service searches is fairly limited.

Returns

Search path or INVALID_RES_ID if no search is currently running

4.9.3.27 E_STB_DP_TTYPE ACTL_GetTerRfChanType (U16BIT *id*)

Returns the terrestrial type (T or T2) for the given channel id.

Parameters

<i>id</i>	index in the terrestrial channel table for the current country
<i>terrestrial</i>	type for the channel

4.9.3.28 E_STB_AV_ASPECT_MODE ACTL_GetTVAspectMode (void)

Returns the current TV aspect mode.

Returns

aspect mode

4.9.3.29 E_STB_AV_ASPECT_RATIO ACTL_GetTVAspectRatio (void)

Returns the current TV aspect ratio.

Returns

aspect ratio

4.9.3.30 U8BIT ACTL_GetVolume (void)

Returns the current audio volume.

Returns

The current volume on a scale of 0-100.

4.9.3.31 BOOLEAN ACTL_HandlePrivateTimerEvent (U32BIT *timer_handle*)

Handles all the private timer events.

Parameters

<i>timer_handle</i>	timer handle
---------------------	--------------

Returns

TRUE if the timer event has been handled, FALSE otherwise

4.9.3.32 BOOLEAN ACTL_HasDecodingStarted (U8BIT *path*)

Returns whether decoding has been started on the given path.

Parameters

<i>path</i>	decode path
-------------	-------------

Returns

TRUE if decoding has started, FALSE otherwise

4.9.3.33 BOOLEAN ACTL_IsAudioDescriptionOn (void)

Returns whether audio description has been turned on.

Returns

TRUE if AD is on

4.9.3.34 BOOLEAN ACTL_IsCiUiRequired (void)

Returns state of ci_ui_required flag.

Returns

TRUE if CI OPEN event has been received

4.9.3.35 BOOLEAN ACTL_IsDecodingLocked (U8BIT *path*)

Returns whether decoding is locked, due to parental locking, on the given path.

Parameters

<i>path</i>	decode path
-------------	-------------

Returns

TRUE if decoding is locked, FALSE otherwise

4.9.3.36 BOOLEAN ACTL_IsHDMIConnected (void)

Returns whether the HDMI is connected or not.

Returns

TRUE if the HDMI is connected, FALSE otherwise

4.9.3.37 BOOLEAN ACTL_IsMuted (void)

Returns the muted state of the audio.

Returns

TRUE if muted, FALSE otherwise

4.9.3.38 BOOLEAN ACTL_IsSearchComplete (void)

Returns TRUE if current search has finished. This works for any of the available types of searches.

Returns

TRUE if search has finished, FALSE otherwise

4.9.3.39 BOOLEAN ACTL_IsTargetRegionRequired (void)

When the search has completed, this function should be called to see whether the target region UI should be presented. This is only needed for DVB-T/T2 searches.

Returns

TRUE if target region UI is required, FALSE otherwise

4.9.3.40 BOOLEAN ACTL_IsTuned (U8BIT *path*)

Returns whether the given path is currently tuned.

Parameters

<i>path</i>	decode path
-------------	-------------

Returns

TRUE if tuned, FALSE otherwise

4.9.3.41 void ACTL_LeaveStandby (void * *s_ptr*, BOOLEAN *tune_to_service*)

Brings the DVB out of standby mode.

Parameters

<i>s_ptr</i>	service to be tuned to for live viewing. If NULL then the last service viewed will be restored
--------------	--

<i>tune_to_service</i>	if <i>s_ptr</i> is NULL then this boolean is used to define whether to tune to the last service viewed
------------------------	--

4.9.3.42 BOOLEAN ACTL_ParentalControlEnabled (void)

Returns whether parental control is enabled. This will also return TRUE if parental control has been set to a valid age.

Returns

TRUE if parental control is enabled, FALSE otherwise

4.9.3.43 void ACTL_ReleaseLivePathForService (void * *s_ptr*, E_STB_DP_RES_OWNER *owner*)

Finds the path being used to view the given service and releases it.

Parameters

<i>s_ptr</i>	service pointer
<i>owner</i>	owner requesting the path to be released

4.9.3.44 BOOLEAN ACTL_ReleasePathOwnership (U8BIT *path*, E_STB_DP_RES_OWNER *owner*)

Releases ownership of the path, and frees any associated data, if the given owner is the path's owner (used by CI+)

Parameters

<i>path</i>	decode path
<i>owner</i>	owner releasing ownership

Returns

TRUE if the ownership is released or the path isn't owned, FALSE otherwise

4.9.3.45 BOOLEAN ACTL_SetActiveProfile (void * *profile*)

Sets the current profile. (CI+ only) If the profile is being set to CI+ profile and the necessary CAM isn't present, then setting the profile will fail.

Parameters

<i>profile</i>	profile to be set
----------------	-------------------

Returns

TRUE if the profile is set successfully, FALSE otherwise

4.9.3.46 void ACTL_SetADVolume (U8BIT *volume*)

Sets the AD volume.

Parameters

<i>volume</i>	AD volume as a percentage
---------------	---------------------------

4.9.3.47 void ACTL_SetMhegAVControl (BOOLEAN *control*)

Sets the MHEG5 audio volume adjust to be applied.

Parameters

<i>control</i>	TRUE if MHEG has control of Audio/Video
----------------	---

4.9.3.48 void ACTL_SetMute (BOOLEAN *mute*)

Sets the audio mute state.

Parameters

<i>mute</i>	TRUE to set mute, FALSE otherwise
-------------	-----------------------------------

4.9.3.49 void ACTL_SetParentalControl (BOOLEAN *enabled*)

Enables or disables parental control. This enables or disables locking on a per channel basis. If an age has previously been set, this will override the setting, effectively turning off age related blockign of channels.

Parameters

<i>enabled</i>	TRUE if parental control is to be enabled
----------------	---

4.9.3.50 void ACTL_SetParentalControlAge (U8BIT *age*)

Sets the age (valid values 4-18) at which parental control will be will be applied. If the age is invalid, no change will be made to the current setting.

Parameters

<i>age</i>	DVB SI age to be set
------------	----------------------

4.9.3.51 void ACTL_SetStandbyState (BOOLEAN *state*)

Reports the standby state to the A/V output controller.

Parameters

<i>state</i>	TRUE for standby is on, FALSE for standby is off
--------------	--

4.9.3.52 void ACTL_SetTVAspectConversion (E_STB_AV_ASPECT_RATIO *aspect_ratio*,
E_STB_AV_ASPECT_MODE *aspect_mode*)

Used to set the aspect conversion applied to the video based on the TV aspect ratio and aspect mode to be applied. The given settings will also be saved.

Parameters

<i>aspect_ratio</i>	TV aspect ratio, e.g. 4:3
<i>aspect_mode</i>	TV aspect mode, e.g. letterbox

4.9.3.53 void ACTL_SetTVAspectMode (E_STB_AV_ASPECT_MODE *aspect_mode*)

Used to set the TV aspect mode which defines how the video will be displayed based on the aspect ratio of the TV and video, along with some other factors.

Parameters

<i>aspect_mode</i>	TV aspect mode, e.g. letterbox
--------------------	--------------------------------

4.9.3.54 void ACTL_SetTVAspectRatio (E_STB_AV_ASPECT_RATIO *aspect_ratio*)

Used to set the TV aspect ratio.

Parameters

<i>aspect_ratio</i>	TV aspect ratio, e.g. 4:3
---------------------	---------------------------

4.9.3.55 void ACTL_SetVideoWindow (S16BIT *win_x*, S16BIT *win_y*, U16BIT *win_width*,
U16BIT *win_height*)

Sets the video window to the size specified. Coordinates are relative to the screen resolution.

Parameters

<i>win_x</i>	window X position
<i>win_y</i>	window Y position
<i>win_width</i>	window width (0 for full screen)
<i>win_height</i>	window height (0 for full screen)

4.9.3.56 U8BIT ACTL_SetVolume (U8BIT *volume*)

Sets the main audio volume and returns the new volume.

Parameters

<i>volume</i>	unsigned value giving volume to be applied
---------------	--

Returns

The new volume on a scale of 0-100.

4.9.3.57 void ACTL_SetVolumeAdjustment (S8BIT *scaling*)

Sets the MHEG5 audio volume adjust to be applied.

Parameters

<i>scaling</i>	the volume scaling from -100% to 100%
----------------	---------------------------------------

4.9.3.58 BOOLEAN ACTL_StartAudioDescription (U8BIT *path*)

Starts decoding an audio description stream, if available, on the given path. If AD isn't currently available it will be marked as enabled and started when it becomes available.

Parameters

<i>path</i>	decode path
-------------	-------------

Returns

TRUE if AD is turned on, FALSE otherwise

4.9.3.59 void ACTL_StartDecoding (U8BIT *path*, void * *s_ptr*)

Sets up and starts decoding for the given service. This is used for PVR playback, but could also be used to start decoding for other non-broadcast services.

Parameters

<i>path</i>	decode path
<i>s_ptr</i>	service

4.9.3.60 BOOLEAN ACTL_StartEitSearch (void)

Entry point for starting an EIT search.

Returns

TRUE if search starts successfully, FALSE otherwise

4.9.3.61 `BOOLEAN ACTL_StartManualSearch (E_STB_DP_SIGNAL_TYPE tuner_type,
S_MANUAL_TUNING_PARAMS * tuning_params, E_ACTL_SEARCH_TYPE type)`

Start a service search on, or just tune to, the transport defined by the given tuning parameters.

Parameters

<i>tuner_type</i>	tuner type to be used for the search
<i>tuning_params</i>	user defined tuning parameters
<i>type</i>	type of service search to be performed

Returns

TRUE if the search is started successfully, FALSE otherwise

4.9.3.62 `BOOLEAN ACTL_StartManualSearchById (E_STB_DP_SIGNAL_TYPE tuner_type,
U16BIT chan_id, BOOLEAN start_search)`

Start a service search on, or just tune to, a transport, using *chan_id* as an index into the country's tuning table.

Parameters

<i>tuner_type</i>	tuner type to be used for the search
<i>chan_id</i>	index into the country's channel array to tune to
<i>start_search</i>	TRUE if a service search is to be started, otherwise just tune

Returns

TRUE on success, FALSE otherwise

4.9.3.63 `void ACTL_StartPlayback (void * s_ptr)`

Start streaming the specified service.

Parameters

<i>s_ptr</i>	the pointer to the service structure containing all the information
--------------	---

4.9.3.64 `BOOLEAN ACTL_StartServiceSearch (E_STB_DP_SIGNAL_TYPE tuner_type,
E_ACTL_SEARCH_TYPE type)`

Entry point for starting a service search for a full retune or to update the existing service lineup.

Parameters

<i>tuner_type</i>	tuner type that defines the search to be performed
<i>type</i>	type of service search to be performed

Returns

TRUE if search starts successfully, FALSE otherwise

4.9.3.65 `BOOLEAN ACTL_StartSSUSearch (void)`

Function to start a search to see if there's an SSU available.

Returns

TRUE if the search is started, FALSE otherwise

4.9.3.66 `BOOLEAN ACTL_StartSSUUpdate (void * t_ptr)`

Starts an SSU update on the given transport. The update may still fail if the transport is later found not to contain an update for the system. If the update isn't mandatory then the process must be continued to actually perform the update.

Parameters

<i>t_ptr</i>	transport containing the SSU
--------------	------------------------------

Returns

TRUE if the SSU update process starts, FALSE otherwise

4.9.3.67 `BOOLEAN ACTL_StartStartupSearch (void)`

Function to start the startup search when booting from cold. This search checks the validity of the services contained in the database. It's only valid for DVB-T/T2 and DVB-C.

Returns

TRUE if the search is started, FALSE otherwise

4.9.3.68 `BOOLEAN ACTL_StartSubtitles (void)`

Starts subtitle processing and display if the current service has valid subtitle data. If DVB subtitles aren't available, teletext will be used if available.

Returns

TRUE if started, FALSE otherwise

4.9.3.69 **BOOLEAN ACTL_StartTotSearch (void)**

Function to start a TOT search which is required to set the system clock when starting from power off if a real-time clock isn't available.

Returns

TRUE if the search is started, FALSE otherwise

4.9.3.70 **void ACTL_StopAudioDescription (U8BIT *path*)**

Stops AD decoding.

Parameters

<i>path</i>	decode path
-------------	-------------

4.9.3.71 **BOOLEAN ACTL_ToggleMute (void)**

Toggles the current mute state and returns the new mute setting.

Returns

The new mute setting, TRUE = muted

4.9.3.72 **void ACTL_TuneOff (U8BIT *path*)**

Stops tuning on the given path.

Parameters

<i>path</i>	decode path
-------------	-------------

4.9.3.73 **BOOLEAN ACTL_TuneToRfChanArrayAnalogEntry (U8BIT *path*, U16BIT *id*)**

Tunes to the specified rf channel array entry in analogue mode.

Parameters

<i>path</i>	decode path id index in the analogue channel table for the current country
-------------	--

Returns

TRUE if id valid, FALSE otherwise

4.9.3.74 **BOOLEAN ACTL_TuneToRfChanArrayEntry** (U8BIT *path*, U16BIT *id*, E_ACTL_SI_SRCH_REQD *reqd_si*, BOOLEAN *relock_on*)

Tunes to the specified rf channel array entry for DVB-T and DVB-C.

Parameters

<i>path</i>	decode path
<i>id</i>	index in the current channel table (for the current signal type and country)
<i>reqd_si</i>	the type of SI handling required
<i>relock_on</i>	if FALSE auto relock will be turned off before tuning, otherwise it will be on

Returns

TRUE if id valid, FALSE otherwise

4.9.3.75 **U8BIT ACTL_TuneToService** (U8BIT *path*, S_ACTL_OWNER_INFO * *owner_info*, void * *s_ptr*, BOOLEAN *override_lock*, BOOLEAN *for_live*)

Starts the process of tuning to the specified service. If the service is to be tuned on the live path, then path should be passed as INVALID_RES_ID to ensure that any CI+ resource handling is performed.

Parameters

<i>path</i>	decode path, INVALID_RES_ID for the live path
<i>owner_info</i>	owner module requesting the tune, can be NULL (used by CI+)
<i>s_ptr</i>	required service
<i>override_lock</i>	TRUE if parental lock settings are to be ignored.
<i>for_live</i>	FALSE for recording, TRUE otherwise

Returns

decode path that is being used to tune to the service. Returning INVALID_RES_ID means the path couldn't be acquired immediately and doesn't mean that it has failed

4.9.3.76 **void ACTL_TuneToTransport** (U8BIT *path*, S_ACTL_OWNER_INFO * *owner_info*, void * *t_ptr*, E_ACTL_SI_SRCH_REQD *reqd_si*, BOOLEAN *relock_on*)

Tunes to the given transport and sets the type of SI monitoring that will be started when the tuning completes.

Parameters

<i>path</i>	decode path, use INVALID_RES_ID to allow the DVB to acquire an appropriate path
-------------	---

<i>owner_info</i>	owner info for the path, can be NULL (used by CI+)
<i>t_ptr</i>	transport to tune to
<i>reqd_si</i>	type of SI monitoring to be started when tuned
<i>relock_on</i>	TRUE if the tuner is to attempt to relock if the signal is lost

4.9.3.77 `BOOLEAN ACTL_TuneToUserDefinedParams (U8BIT path,
S_MANUAL_TUNING_PARAMS * tuning_params, E_ACTL_SI_SRCH_REQD
reqd_si, BOOLEAN relock_on)`

Tunes to the given set of tuning parameters.

Parameters

<i>path</i>	decode path
<i>tuning_params</i>	tuning parameters to be used
<i>reqd_si</i>	SI search mode to be used if tuning is successful
<i>relock_on</i>	defines whether the tuner is to attempt to relock if lock is lost

Returns

TRUE if tuning is started, FALSE otherwise

4.9.3.78 `BOOLEAN ACTL_TuneUsingDSD (U8BIT path, SI_DELIVERY_SYS_DESC_TYPE
dsd_type, SI_DELIVERY_SYS_DESC * dsd, U16BIT service_id,
E_ACTL_SI_SRCH_REQD reqd_si)`

Starts the process of tuning to a given transport or service that's defined by the given delivery system descriptor. If the DSD defines a transport that doesn't currently exist, then one will be created, and if a *service_id* is given and a service with this ID doesn't exist on the transport, then one will be created, but will be marked as hidden and non-selectable so it won't appear to a user.

Parameters

<i>path</i>	decode path to be tuned, must be valid
<i>dsd_type</i>	delivery system descriptor type (T, C or S)
<i>dsd</i>	delivery system descriptor
<i>service_id</i>	pass as 0 if just tuning to a transport
<i>reqd_si</i>	SI processing mode that should be selected when tuned. If tuning to a service the normal UPDATE mode will be used and this setting will be ignored.

Returns

TRUE if the tune operation is started, FALSE otherwise

4.9.3.79 void **ACTL_UpdateVideoMode** (E_STB_AV_ASPECT_RATIO *aspect*, BOOLEAN *force*)

Update video mode sets Voyager resolution as well as platform Also, stops subtitles during the operation. If HDMI resolution mode is not supported, it chooses the best one available.

Parameters

<i>aspect</i>	TV aspect ratio, e.g. 4:3 or 16:9
<i>force</i>	forces resolution update when TRUE

4.10 dvb/inc/ap_dbacc.h File Reference

Application database access functions.

```
#include "stbhwtun.h" #include "stbdpc.h" #include "stbsitab.-
h" #include "stbgc.h" Include dependency graph for ap_dbacc.h:
```

Data Structures

- struct [ADB_LNB_SETTINGS](#)
- struct [ADB_EVENT_COMPONENT_INFO](#)
Structure representing the component information as found in the EIT component - descriptor.
- struct [ADB_EVENT_ITEMIZED_INFO](#)

Defines

- #define **ADB_INVALID_DVB_ID** 0xffff
- #define **LINK_TYPE_CA_REPLACEMENT_SERVICE** 0x03
- #define **LINK_TYPE_SERVICE_REPLACEMENT** 0x05
- #define **ADB_NUM_SERV_NAME_IDS** 6
- #define **ADB_MAX_FAVGROUPS** 8
- #define **LNB_C_BAND_FREQUENCY** 5150
- #define **LNB_KU_BAND_FREQUENCY_MINIMUM** 9750
- #define **LNB_KU_BAND_FREQUENCY_MAXIMUM** 12000
- #define **LNB_KU_BAND_FREQUENCY_LOW** LNB_KU_BAND_FREQUENCY_MINIMUM
- #define **LNB_KU_BAND_FREQUENCY_HIGH** 10600
- #define **ADB_LIST_TYPE_FROM_FAVLIST**(id) (((id) << 16) | ADB_SERVICE_LIST_FAV_LIST)
- #define **ADB_FAVLIST_FROM_LIST_TYPE**(L) (U8BIT)((L) >> 16)

Enumerations

- enum **ADB_TRANSPORT_LIST_TYPE** { **ADB_TRANSPORT_LIST_ALL** = 0xff, **ADB_TRANSPORT_LIST_NOT_TUNED** = 0x01, **ADB_TRANSPORT_LIST_TUNED** = 0x02 }
- enum **ADB_SERVICE_LIST_TYPE** { **ADB_SERVICE_LIST_UNKNOWN** = 0x0000, **ADB_SERVICE_LIST_TV** = 0x0001, **ADB_SERVICE_LIST_RADIO** = 0x0002, **ADB_SERVICE_LIST_DATA** = 0x0004, **ADB_SERVICE_LIST_DIGITAL** = 0x0007, **ADB_SERVICE_LIST_ANALOG** = 0x0080, **ADB_SERVICE_LIST_TV_DATA** = 0x0085, **ADB_SERVICE_LIST_FAV_GROUP_A** = 0x0100, **ADB_SERVICE_LIST_FAV_GROUP_B** = 0x0200, **ADB_SERVICE_LIST_FAV_GROUP_C** = 0x0400, **ADB_SERVICE_LIST_FAV_GROUP_D** = 0x0800, **ADB_SERVICE_LIST_ALL** = (ADB_SERVICE_LIST_DIGITAL | ADB_SERVICE_LIST_ANALOG), **ADB_SERVICE_LIST_FAV_LIST** = 0x8000 }
- enum **ADB_SERVICE_TYPE** { **ADB_SERVICE_TYPE_TV** = 0x01, **ADB_SERVICE_TYPE_RADIO** = 0x02, **ADB_SERVICE_TYPE_TELETEXT** = 0x03, **ADB_SERVICE_TYPE_NVOD_REF** = 0x04, **ADB_SERVICE_TYPE_NVOD_TIMEShift** = 0x05, **ADB_SERVICE_TYPE_MOSAIC** = 0x06, **ADB_SERVICE_TYPE_AVC_RADIO** = 0x0a, **ADB_SERVICE_TYPE_AVC_MOSAIC** = 0x0b, **ADB_SERVICE_TYPE_DATA** = 0x0c, **ADB_SERVICE_TYPE_MPEG2_HD** = 0x11, **ADB_SERVICE_TYPE_AVC_SD_TV** = 0x16, **ADB_SERVICE_TYPE_AVC_SD_NVOD_TIMEShift** = 0x17, **ADB_SERVICE_TYPE_AVC_SD_NVOD_REF** = 0x18, **ADB_SERVICE_TYPE_HD_TV** = 0x19, **ADB_SERVICE_TYPE_AVC_HD_NVOD_TIMEShift** = 0x1a, **ADB_SERVICE_TYPE_AVC_HD_NVOD_REF** = 0x1b, **ADB_SERVICE_TYPE_UHD_TV** = 0x1f, **ADB_SERVICE_TYPE_ANALOG** = 0x100 }
- enum **ADB_STREAM_TYPE** { **ADB_VIDEO_STREAM**, **ADB_H264_VIDEO_STREAM**, **ADB_H265_VIDEO_STREAM**, **ADB_AUDIO_STREAM**, **ADB_AAC_AUDIO_STREAM**, **ADB_HEAAC_AUDIO_STREAM**, **ADB_AC3_AUDIO_STREAM**, **ADB_EAC3_AUDIO_STREAM**, **ADB_SUBTITLE_STREAM**, **ADB_DATA_STREAM**, **ADB_TTEXT_STREAM** }
- enum **ADB_STREAM_LIST_TYPE** { **ADB_STREAM_LIST_ALL** = 0xff, **ADB_VIDEO_LIST_STREAM** = 0x01, **ADB_AUDIO_LIST_STREAM** = 0x02, **ADB_SUBTITLE_LIST_STREAM** = 0x04, **ADB_DATA_LIST_STREAM** = 0x08, **ADB_TTEXT_LIST_STREAM** = 0x10, **ADB_TTEXT_SUBT_LIST_STREAM** = 0x20 }
- enum **ADB_AUDIO_TYPE** { **ADB_AUDIO_TYPE_UNDEFINED** = 0, **ADB_AUDIO_TYPE_NORMAL** = 1, **ADB_AUDIO_TYPE_FOR_HEARING_IMPAIRED** = 2, **ADB_AUDIO_TYPE_FOR_VISUALLY_IMPAIRED** = 3 }
- enum **ADB_SUBTITLE_TYPE** { **ADB_SUBTITLE_TYPE_DVB** = 0x10, **ADB_SUBTITLE_TYPE_DVB_4_3** = 0x11, **ADB_SUBTITLE_TYPE_DVB_16_9** = 0x12, **ADB_SUBTITLE_TYPE_DVB_221_1** = 0x13, **ADB_SUBTITLE_TYPE_DVB_HD** = 0x14, **ADB_SUBTITLE_TYPE_DVB_HARD_HEARING** = 0x20, **ADB_SUBTITLE_TYPE_DVB_HARD_HEARING_4_3** = 0x21, **ADB_SUBTITLE_TYPE_DVB_HARD_HEARING_16_9** = 0x22, **ADB_SUBTITLE_TYPE_DVB_HARD_HEARING_221_1** = 0x23, **ADB_SUBTITLE_TYPE_DVB_HARD_HEARING_HD** = 0x24 }
- enum **ADB_TELETEXT_TYPE** { **ADB_TELETEXT_TYPE_INITIAL** = 0x01, **ADB_TELETEXT_TYPE_SUBTITLE** = 0x02, **ADB_TELETEXT_TYPE_ADDITION-**

- AL_INFO** = 0x03, **ADB_TELETEXT_TYPE_SCHEDULE** = 0x04, **ADB_TELETEXT_TYPE_SUBTITLE_HARD_HEARING** = 0x05 }
- enum **ADB_EVENT_CONTENT** { **ADB_EVENT_CONTENT_UNCLASSIFIED** = 0x00, **ADB_EVENT_CONTENT_MOVIE** = 0x10, **ADB_EVENT_CONTENT_NEWS** = 0x20, **ADB_EVENT_CONTENT_ENTERTAINMENT** = 0x30, **ADB_EVENT_CONTENT_SPORT** = 0x40, **ADB_EVENT_CONTENT_CHILD** = 0x50, **ADB_EVENT_CONTENT_MUSIC** = 0x60, **ADB_EVENT_CONTENT_ARTS** = 0x70, **ADB_EVENT_CONTENT_SOCIAL** = 0x80, **ADB_EVENT_CONTENT_EDUCATION** = 0x90, **ADB_EVENT_CONTENT_LEISURE** = 0xA0, **ADB_EVENT_CONTENT_SPECIAL** = 0xB0, **ADB_EVENT_CONTENT_RESERVED1** = 0xC0, **ADB_EVENT_CONTENT_RESERVED2** = 0xD0, **ADB_EVENT_CONTENT_RESERVED3** = 0xE0, **ADB_EVENT_CONTENT_USERDEFINED** = 0xF0 }
 - enum **ADB_FAV_GROUP** { **FAV_GROUP_A** = 0x01, **FAV_GROUP_B** = 0x02, **FAV_GROUP_C** = 0x04, **FAV_GROUP_D** = 0x08, **FAV_GROUP_ALL** = 0x0f }
 - enum **ADB_PROFILE_TYPE** { **ADB_PROFILE_TYPE_BROADCAST**, **ADB_PROFILE_TYPE_CIPPLUS** }

Functions

- void **ADB_ResetDatabase** (void)
Clears the service database and resets everything to defaults values.
- void **ADB_SaveDatabase** (void)
Saves the database to non-volatile memory.
- void **ADB_SaveEventSchedule** (E_STB_DP_SIGNAL_TYPE tuner_type, void *satellite)
*Saves the event schedule for all services for the given type of tuner to the database.
Note: the database has to support this operation.*
- void **ADB_DeleteServices** (E_STB_DP_SIGNAL_TYPE tuner_type, void *satellite)
Deletes all networks, transports and services related to the given type of tuner.
- void **ADB_ReleaseNameList** (U8BIT **name_list, U16BIT num_names)
Frees the memory used by any of the name lists (e.g. networks, services, etc)
- U16BIT **ADB_GetNumSatellites** (void)
Returns the number of satellites in the database.
- void * **ADB_AddSatellite** (U8BIT *name_str, U16BIT dish_pos, U16BIT long_pos, BOOLEAN east_west, void *lnb_ptr)
Creates a new satellite in the database with the given settings.
- void * **ADB_GetNextSatellite** (void *sat_ptr)
Returns the next satellite from the database.
- U8BIT * **ADB_GetSatelliteName** (void *sat_ptr)
Returns the pointer to the name of the satellite.
- void * **ADB_GetSatelliteLNB** (void *sat_ptr)
Returns the LNB associated with the given satellite.
- U16BIT **ADB_GetSatelliteLongitude** (void *sat_ptr)
Returns the longitudinal position of the given satellite in 1/10ths degree.

- **BOOLEAN** [ADB_GetSatelliteDirection](#) (void *sat_ptr)
Returns the position direction of the given satellite.
- **U16BIT** [ADB_GetNumLNBS](#) (void)
Returns the number of LNBS in the database.
- **void *** [ADB_AddLNB](#) (**ADB_LNB_SETTINGS** *settings)
Creates a new LNB in the database with the given settings.
- **void *** [ADB_FindLNBWithSettings](#) (**ADB_LNB_SETTINGS** *settings)
Searches the database to find an existing LNB with the given settings.
- **BOOLEAN** [ADB_GetLNBSettings](#) (void *lnb_ptr, **ADB_LNB_SETTINGS** *settings)
Returns the current settings for the given LNB.
- **U16BIT** [ADB_GetNumNetworks](#) (void)
Returns the number of network records in the database.
- **void** [ADB_GetNetworkList](#) (void ***nlist_ptr, **U16BIT** *num_entries_ptr)
Allocates and returns a list of the network records in the database.
- **void** [ADB_ReleaseNetworkList](#) (void **nlist)
Frees a network list returned by ADB_GetNetworkList.
- **U8BIT **** [ADB_GetNetworkListNames](#) (void **nlist, **U16BIT** num_entries, **BOOLEAN** short_names)
Returns a list of names, in UTF-8 format, corresponding to the given network list. The returned list should be freed using ADB_ReleaseNameList.
- **U32BIT *** [ADB_GetNetworkListIds](#) (void **nlist, **U16BIT** num_entries)
Allocates and returns an array of network ids for the given networks. The returned array should be freed using STB_AppFreeMemory.
- **U8BIT *** [ADB_GetNetworkName](#) (void *n_ptr)
Returns the name, in UTF-8 format, of the given network. The returned name should be freed using STB_ReleaseUnicodeString.
- **U8BIT *** [ADB_GetNetworkNameByLang](#) (void *n_ptr, **U8BIT** lang)
Returns the name in the language defined by lang, in UTF-8 format, of the given network. The returned name should be freed using STB_ReleaseUnicodeString.
- **U16BIT** [ADB_GetNetworkId](#) (void *n_ptr)
Returns the network id of the given network.
- **U8BIT** [ADB_GetNetworkTargetCountries](#) (**U32BIT** **code_array)
Returns a combined array of country codes for all discovered networks.
- **U16BIT** [ADB_GetNetworkPrimaryTargetRegions](#) (**U32BIT** country_code, **U8BIT** **code_array, **U8BIT** ***name_array)
Returns an array of primary region codes and names for the given country.
- **U16BIT** [ADB_GetNetworkSecondaryTargetRegions](#) (**U32BIT** country_code, **U8BIT** primary_region, **U8BIT** **code_array, **U8BIT** ***name_array)
Returns an array of secondary region codes and names for the given country and primary region.
- **U16BIT** [ADB_GetNetworkTertiaryTargetRegions](#) (**U32BIT** country_code, **U8BIT** primary_region, **U8BIT** secondary_region, **U16BIT** **code_array, **U8BIT** ***name_array)

- Returns an array of tertiary region codes and names for the given country, primary region and secondary region.*
- void * [ADB_GetTunedNetwork](#) (U8BIT path)
Returns the network currently tuned to on the given decode path.
 - U16BIT [ADB_GetNumTransports](#) (void)
Returns the number of transports in the database.
 - void [ADB_GetTransportList](#) (void ***tlist_ptr, U16BIT *num_entries_ptr)
Allocates and returns a list of all transport records in the database.
 - void [ADB_GetNetworkTransportList](#) (void *n_ptr, void ***tlist_ptr, U16BIT *num_entries_ptr)
Allocates and returns a list of all transport records in the database for the given network.
 - void [ADB_ReleaseTransportList](#) (void **tlist, U16BIT num_transports)
Frees a transport list returned from one of the ADB_GetTransportList functions.
 - U8BIT ** [ADB_GetTransportListNames](#) (void **tlist, U16BIT num_entries)
Returns a list of names, in UTF-8 format, corresponding to the given transport list. The returned list should be freed using ADB_ReleaseNameList.
 - U32BIT * [ADB_GetTransportListTids](#) (void **tlist, U16BIT num_entries)
Allocates and returns an array of transport ids for the given transports. The returned array should be freed using STB_AppFreeMemory.
 - U32BIT * [ADB_GetTransportListTunedStrengths](#) (void **tlist, U16BIT num_entries)
Allocates and returns an array of percentage values representing the signal strength when each transport in the given list was tuned to. The returned array should be freed using STB_AppFreeMemory.
 - U32BIT * [ADB_GetTransportListOriginalNetworkIds](#) (void **tlist, U16BIT num_entries)
Allocates and returns an array of original network ids for each transport in the given list was tuned to. The returned array should be freed using STB_AppFreeMemory.
 - U32BIT * [ADB_GetTransportListFreqs](#) (void **tlist, U16BIT num_entries)
Allocates and returns an array of tuning frequencies for each transport in the given list. The returned array should be freed using STB_AppFreeMemory.
 - E_STB_DP_SIGNAL_TYPE [ADB_GetTransportSignalType](#) (void *t_ptr)
Returns the signal type for the given transport.
 - U8BIT * [ADB_GetTransportName](#) (void *t_ptr)
Returns the name, in UTF-8 format, of the given transport. The returned name should be freed using STB_ReleaseUnicodeString.
 - U16BIT [ADB_GetTransportTid](#) (void *t_ptr)
Returns the transport id of the given transport.
 - U16BIT [ADB_GetTransportOriginalNetworkId](#) (void *t_ptr)
Returns the original network id of the given transport.
 - void * [ADB_GetTransportNetworkPtr](#) (void *t_ptr)
Returns the network of the given transport.
 - U32BIT [ADB_GetTransportFreqHz](#) (void *t_ptr)
Returns the frequency, in Hz for DVB-T/T2 and DVB-C, and MHz for DVB-S/S2, of the given transport.

- U8BIT [ADB_GetTransportBwidth](#) (void *t_ptr)
Returns the bandwidth value of the given DVB-T/T2 transport.
- S8BIT [ADB_GetTransportOffset](#) (void *t_ptr)
Returns the tuning offset of the given DVB-T/T2 transport.
- E_STB_TUNE_TCONST [ADB_GetTransportConstellation](#) (void *t_ptr)
Returns the constellation of the given DVB-T/T2 transport.
- E_STB_DP_TMODE [ADB_GetTransportTerrMode](#) (void *t_ptr)
Returns the terrestrial mode of the given DVB-T/T2 transport.
- E_STB_TUNE_TCODERATE [ADB_GetTransportLpCodeRate](#) (void *t_ptr)
Returns the LP coderate of the given DVB-T/T2 transport.
- E_STB_TUNE_TCODERATE [ADB_GetTransportHpCodeRate](#) (void *t_ptr)
Returns the HP coderate of the given DVB-T/T2 transport.
- E_STB_TUNE_TGUARDINT [ADB_GetTransportGuardInt](#) (void *t_ptr)
Returns the guard interval of the given DVB-T/T2 transport.
- E_STB_TUNE_THIERARCHY [ADB_GetTransportHierarchy](#) (void *t_ptr)
Returns the hierarchy of the given DVB-T/T2 transport.
- E_STB_DP_CMODE [ADB_GetTransportCableMode](#) (void *t_ptr)
Returns the QAM mode of the given cable transport.
- U16BIT [ADB_GetTransportCableSymbolRate](#) (void *t_ptr)
Returns the symbol rate of the given cable transport.
- U16BIT [ADB_GetTransportSatelliteSymbolRate](#) (void *t_ptr)
Returns the symbol rate of the given satellite transponder.
- U8BIT [ADB_GetTransportTunedStrength](#) (void *t_ptr)
Returns the signal strength, as a percentage, of the tuned transport.
- U8BIT [ADB_GetTransportTunedQuality](#) (void *t_ptr)
Returns the signal quality, as a percentage, of the tuned transport.
- void * [ADB_GetFullSITransport](#) (void *n_ptr)
Returns a transport that is signalled as containing a complete set of SI data via an NIT linkage descriptor.
- void [ADB_GetTransportTerrestrialTuningParams](#) (void *t_ptr, E_STB_DP_TTYPE *terr_type, U32BIT *freq_hz, E_STB_DP_TMODE *mode, E_STB_DP_TBW_IDTH *bwidth, U8BIT *plp_id)
Returns the settings to tune to the given DVB-T/T2 transport.
- void [ADB_GetTransportAnalogTuningParams](#) (void *t_ptr, U32BIT *freq_hz, S8BIT *offset, E_STB_DP_ANALOG_VIDEO_TYPE *vtype)
Returns the settings to tune to the given terrestrial analog transport.
- void [ADB_GetTransportCableTuningParams](#) (void *t_ptr, U32BIT *freq_hz, E_STB_DP_CMODE *mode, U16BIT *symrate)
Returns the parameters needed to tune a cable tuner to a transport.
- void [ADB_GetTransportSatTuningParams](#) (void *t_ptr, U32BIT *freq_hz, E_STB_DP_POLARITY *polarity, U16BIT *symrate, E_STB_DP_FEC *fec, BOOLEAN *dvb_s2, E_STB_DP_MODULATION *modulation)
Returns the parameters needed to tune a satellite tuner to a transport.
- void * [ADB_GetTransportSatellite](#) (void *t_ptr)

- Returns the parent satellite for the given transport.*

 - void * [ADB_GetTransportFromIds](#) (U16BIT net_id, U16BIT onet_id, U16BIT tran_id)
- Finds the transport with the given ids.*

 - void [ADB_SetTunedTransport](#) (U8BIT path, void *t_ptr)

Sets the given transport as the one tuned to on the given decode path. The transport's network is also set as the tuned network, if valid.
- void * [ADB_GetTunedTransport](#) (U8BIT path)

Returns the transport that's tuned to on the given decode path.
- U16BIT [ADB_GetNumServices](#) (void)

Returns the total number of services in the database.
- U16BIT [ADB_GetNumServicesInList](#) (U32BIT list_type, BOOLEAN inc_hidden)

Returns the number of services in the database that would be returned with the given list type.
- void [ADB_GetServiceList](#) (U32BIT list_type, void ***slist_ptr, U16BIT *num_entries_ptr)

Allocates and returns a list of all services in the database for the given list type.
- void [ADB_GetServiceListIncludingHidden](#) (U32BIT list_type, void ***slist_ptr, U16BIT *num_entries_ptr, BOOLEAN ignore_selectable)

Allocates and returns a list of all services in the database for the given list type, including hidden services.
- void [ADB_GetTransportServiceList](#) (void *t_ptr, void ***slist_ptr, U16BIT *num_entries_ptr)

Allocates and returns a list of all services in the database on the given transport.
- void [ADB_GetNetworkServiceList](#) (void *n_ptr, void ***slist_ptr, U16BIT *num_entries_ptr)

Allocates and returns a list of all services in the database on all transports on the given network.
- void [ADB_GetUnavailableServiceList](#) (void ***slist_ptr, U16BIT *num_entries_ptr)

Allocates and returns a list of all services that are marked as being unavailable. A service becomes unavailable when it disappears from the SDT.
- void [ADB_GetLockedServiceList](#) (void ***slist_ptr, U16BIT *num_entries_ptr)

Allocates and returns a list of all services that are marked as locked for parental control purposes.
- void [ADB_ReleaseServiceList](#) (void **slist, U16BIT num_serve)

Frees a list of services returned from one of the functions that returns a service list, such as [ADB_GetServiceList](#).
- void [ADB_GetServiceListLockedSubset](#) (void **slist, U16BIT num_entries, void ***locked_slist_ptr, U16BIT *locked_num_entries_ptr)

Creates a subset of locked services from the supplied service list. Memory is allocated as TempMemory so will be released when the calling screen closes.
- void [ADB_SortServiceListAlphabetically](#) (void **slist, U16BIT num_entries, BOOLEAN short_name)

Sorts the given list of services into alphabetical order. Case is significant.

- **BOOLEAN** [ADB_IsValidService](#) (void *serv_ptr)
Checks whether the given service is in the current service list.
- void * [ADB_GetNextServiceInList](#) (U32BIT list_type, void *serv)
Returns the next service, in LCN order, for the given list type, starting from the given service. If serv is NULL then the first service is returned.
- void * [ADB_GetPrevServiceInList](#) (U32BIT list_type, void *serv)
Returns the previous service, in LCN order, for the given list type, starting from the given service. If serv is NULL then the last service is returned.
- void * [ADB_FindServiceByLcn](#) (U32BIT list_type, U16BIT lcn, BOOLEAN show_unselectable)
Returns the service matching the given LCN and list type.
- void * [ADB_FindServiceByNearestLcn](#) (U32BIT list_type, U16BIT lcn)
Returns the service matching the given, or preceding, LCN and list type.
- void * [ADB_FindServiceByIds](#) (U16BIT onet_id, U16BIT tid, U16BIT sid)
Returns a pointer to the service matching the given IDs.
- U16BIT [ADB_FindLcnInServiceList](#) (U16BIT lcn, void **slist_ptr, U16BIT num_entries)
Returns the index into the given list of services of the service with the given LCN.
- U16BIT [ADB_FindNearestLcnInServiceList](#) (U16BIT lcn, void **slist_ptr, U16BIT num_entries)
Returns the index in the given list of services of the service matching the given, or preceding, LCN.
- U16BIT [ADB_FindServiceInList](#) (void *s_ptr, void **slist_ptr, U16BIT num_entries)
Returns the index in the given list of services of the given service.
- U8BIT ** [ADB_GetServiceListFullNames](#) (void **slist, U16BIT num_entries, BOOLEAN use_pref_names)
Allocates and returns a list of the fullnames for all the services in the given service list. The returned list should be freed using ADB_ReleaseNameList.
- U8BIT ** [ADB_GetServiceListShortNames](#) (void **slist, U16BIT num_entries, BOOLEAN use_pref_names)
Allocates and returns a list of the short names for all the services in the given service list. The returned list should be freed using ADB_ReleaseNameList.
- U16BIT * [ADB_GetServiceListLcns](#) (void **slist, U16BIT num_entries)
Allocates and returns an array of the logical channel numbers that have been assigned to all the services in the given service list. The returned list should be freed using STB_AppFreeMemory.
- U16BIT * [ADB_GetServiceListOrigLcns](#) (void **slist, U16BIT num_entries)
Allocates and returns an array of the logical channel numbers that each of the services in the given service list originally requested. The returned list should be freed using STB_AppFreeMemory.
- U32BIT * [ADB_GetServiceListUnavailableFlag](#) (void **slist, U16BIT num_entries)
Allocates and returns an array of the unavailable flags for each of the services in the given service list. A value of TRUE means the service is unavailable, FALSE otherwise. The returned list should be freed using STB_AppFreeMemory.

- U32BIT * [ADB_GetServiceListNewFlag](#) (void **slist, U16BIT num_entries)
Allocates and returns an array of the 'new' flags for each of the services in the given service list. A service is marked as 'new' when it's added to the service database following an update service scan. A value of TRUE means the service is new, FALSE otherwise. The returned list should be freed using STB_AppFreeMemory.
- U32BIT * [ADB_GetServiceListLockedFlag](#) (void **slist, U16BIT num_entries)
Allocates and returns an array of the 'locked' flags for each of the services in the given service list. A service is marked as 'locked' due to setup of parental control. A value of TRUE means the service is locked, FALSE otherwise. The returned list should be freed using STB_AppFreeMemory.
- void [ADB_SetServiceListLockedFlag](#) (void **slist, U16BIT num_entries, BOOLEAN locked)
Sets the locked state of all services in the given service list.
- U32BIT * [ADB_GetServiceListScrambledFlag](#) (void **slist, U16BIT num_entries)
Allocates and returns an array of the scrambled flags for each of the services in the given service list. A value of TRUE means the service is signalled as scrambled and FALSE means it's free-to-air. The returned list should be freed using STB_AppFreeMemory.
- U32BIT * [ADB_GetServiceListHdFlag](#) (void **slist, U16BIT num_entries)
Allocates and returns an array indicating whether each service in the given service list is an HD service. A value of TRUE means the service is signalled as being HD and FALSE means it's SD. The returned list should be freed using STB_AppFreeMemory.
- U32BIT * [ADB_GetServiceListHiddenFlag](#) (void **slist, U16BIT num_entries)
Allocates and returns an array indicating whether each service in the given service list is signalled as being hidden or not. A value of TRUE means the service is hidden and FALSE means it's visible. The returned list should be freed using STB_AppFreeMemory.
- U8BIT ** [ADB_GetServiceListNetworkNames](#) (void **slist, U16BIT num_entries)
Allocates and returns a list of network names for all the services in the given service list. The returned list should be freed using ADB_ReleaseNameList.
- U8BIT ** [ADB_GetServiceListTransportNames](#) (void **slist, U16BIT num_entries)
Allocates and returns a list of transport names for all the services in the given service list. The returned list should be freed using ADB_ReleaseNameList.
- U32BIT * [ADB_GetServiceListTransportStrengths](#) (void **slist, U16BIT num_entries)
Allocates and returns an array of the signal strengths of the transports containing the given list of services. The returned list should be freed using STB_AppFreeMemory.
- void [ADB_SetServiceHiddenFlag](#) (void *s_ptr, BOOLEAN hidden)
Sets the hidden status of the given service, which means the service would not be presented to the user in any list of available services.
- BOOLEAN [ADB_GetServiceHiddenFlag](#) (void *s_ptr)
Returns the hidden status of the given service. A hidden service should not be present to the user in any list of available services.
- BOOLEAN [ADB_GetServiceLcnEditable](#) (void *s_ptr)
Returns whether the LCN assigned to the given service can be changed.

- U8BIT * [ADB_GetServiceFullName](#) (void *s_ptr, BOOLEAN use_pref_name)
Returns the full name of the given service as a UTF-8 string. The returned string should be freed using *STB_ReleaseUnicodeString*.
- U8BIT * [ADB_GetServiceShortName](#) (void *s_ptr, BOOLEAN use_pref_name)
Returns the short name of the given service as a UTF-8 string. The returned string should be freed using *STB_ReleaseUnicodeString*.
- U8BIT * [ADB_GetServiceFullNameByLangAndPrefId](#) (void *s_ptr, U8BIT lang, U8BIT pref_name_id)
Returns the full name of the given service as a UTF-8 string using the given language and preferred name id, rather than the defaults. The returned string should be freed using *STB_ReleaseUnicodeString*.
- U8BIT * [ADB_GetServiceShortNameByLangAndPrefId](#) (void *s_ptr, U8BIT lang, U8BIT pref_name_id)
Returns the short name of the given service as a UTF-8 string using the given language and preferred name id, rather than the defaults. The returned string should be freed using *STB_ReleaseUnicodeString*.
- U16BIT [ADB_GetServiceLcn](#) (void *s_ptr)
Returns the logical channel number assigned to the given service.
- void [ADB_SetServiceLcn](#) (void *s_ptr, U16BIT lcn)
Sets the assigned logical channel number of the given service. This will override any LCN assigned when performing a service scan.
- U16BIT [ADB_GetServiceId](#) (void *s_ptr)
Returns the signalled service id of the given service.
- void [ADB_GetServiceIds](#) (void *s_ptr, U16BIT *onet_id, U16BIT *trans_id, U16BIT *serv_id)
Returns the original network id, transport id and service id for the given service.
- void [ADB_SetServiceScheduleState](#) (void *s_ptr, BOOLEAN state)
Sets whether the EIT schedule events for this service are held in memory. All services hold schedule events in memory by default.
- BOOLEAN [ADB_GetServiceScheduleState](#) (void *s_ptr)
Returns whether the EIT schedule events are being held in memory for the service.
- void [ADB_SetServiceNowNextState](#) (void *s_ptr, BOOLEAN state)
Sets whether the now/next EIT events for this service are held in memory. All services save now/next events by default.
- BOOLEAN [ADB_GetServiceNowNextState](#) (void *s_ptr)
Returns whether the EIT now/next data is being held in memory for the service.
- void [ADB_SortServicesByLcn](#) (void)
Sort all services in the database according to their LCNs.
- U16BIT [ADB_GetServiceOrigLcn](#) (void *s_ptr)
Returns the logical channel number originally requested by the given service.
- U16BIT [ADB_GetServicePCRPid](#) (void *s_ptr)
Returns the PCR PID for the given service.
- U16BIT [ADB_GetServiceAudioPid](#) (void *s_ptr)
Returns the audio PID for the given service. This will be the one chosen based on the settings for language, etc, or the one explicitly set (e.g. due to user selection).

- ADB_STREAM_TYPE [ADB_GetServiceAudioType](#) (void *s_ptr)
Returns the audio type being used by the given service.
- U16BIT [ADB_GetServiceVideoPid](#) (void *s_ptr)
Returns the video PID for the given service.
- ADB_STREAM_TYPE [ADB_GetServiceVideoType](#) (void *s_ptr)
Returns the video type being used by the given service.
- U16BIT [ADB_GetServiceTextPid](#) (void *s_ptr)
Returns the PID being used for DVB subtitles and/or teletext for the given service. This will be the one chosen based on the settings for language, etc, or the one explicitly set (e.g. due to user selection).
- U16BIT [ADB_GetRequiredAudioPid](#) (void *s_ptr)
Returns the audio PID to be used for the given service based on language settings, output format, etc.
- U16BIT [ADB_GetRequiredADPid](#) (void *s_ptr, BOOLEAN *broadcast_mix)
Returns the PID to be used for audio description for the given service based on language settings, output format, etc.
- E_STB_DP_AUDIO_MODE [ADB_GetRequiredAudioMode](#) (void *s_ptr)
Returns the mode to be used for audio for the given service based on language settings, output format, etc.
- U16BIT [ADB_GetRequiredTtextPid](#) (void *s_ptr, BOOLEAN for_subtitles, U8BIT *magazine, U8BIT *page)
Returns the PID to be used for teletext for the given service based on language settings.
- U16BIT [ADB_GetRequiredSubtitleParams](#) (void *s_ptr, U16BIT *cpage_ptr, - U16BIT *apage_ptr)
Returns the PID to be used for DVB subtitles for the given service based on language settings.
- ADB_SERVICE_TYPE [ADB_GetServiceType](#) (void *s_ptr)
Returns the signalled type of the given service.
- U8BIT * [ADB_GetServiceProviderName](#) (void *s_ptr)
Returns the provider name, as a UTF-8 string, using the default language, of the given service. The returned string should be freed using STB_ReleaseUnicodeString.
- U8BIT * [ADB_GetServiceProviderNameByLang](#) (void *s_ptr, U8BIT lang)
Returns the provider name, as a UTF-8 string, using the given language, of the given service. The returned string should be freed using STB_ReleaseUnicodeString.
- BOOLEAN [ADB_GetServiceUnavailableFlag](#) (void *s_ptr)
Returns the status of the unavailable flag of the given service. This flag indicates whether a service is no longer signalled in the SDT.
- U8BIT [ADB_GetServiceRunningStatus](#) (void *s_ptr)
Returns the running status of the given service as reported by the SDT.
- BOOLEAN [ADB_GetServiceNotRunningFlag](#) (void *s_ptr)
Returns the status of the 'not running' flag of the given service. This flag indicates whether a service is signalled as not running in the SDT, or is removed from the PAT.
- BOOLEAN [ADB_GetServiceNewFlag](#) (void *s_ptr)
Returns the status of the 'new' flag of the given service. This flag indicates whether a service has been added to the service lineup following a service scan.

- **BOOLEAN ADB_GetServiceLockedFlag** (void *s_ptr)
Returns the status of the 'locked' flag of the given service, which is used by the parental control.
- **void ADB_ToggleServiceLockedFlag** (void *s_ptr)
Changes the current state of the given service from locked to unlocked, or vice versa.
- **void ADB_SetServiceLockedFlag** (void *s_ptr, BOOLEAN locked)
Locks or unlocks the given service.
- **BOOLEAN ADB_GetServiceScrambledFlag** (void *s_ptr)
Returns the status of the 'scrambled' flag of the given service. This flag is set depending on the scrambled state of all the streams that makeup the service being marked as free-to-air.
- **BOOLEAN ADB_GetServiceHasCaDesc** (void *s_ptr)
Used to query whether the given service has a CA descriptor.
- **BOOLEAN ADB_GetServiceDoNotScramble** (void *s_ptr)
Returns the do_not_scramble flag for the service which is based on the presence of an FTA descriptor. The value returned will be found by looking backwards from the service to the transport, etc.
- **BOOLEAN ADB_IsFreesatService** (void *s_ptr)
Returns a value indicating whether the given service is a Freesat service.
- **U16BIT ADB_GetFreesatServiceId** (void *s_ptr)
Returns Freesat service ID of the given service.
- **void * ADB_GetServiceTransportPtr** (void *s_ptr)
Returns a pointer to the service's parent transport record.
- **U8BIT * ADB_GetServiceDefaultAuthority** (void *serv_ptr)
Returns a copy of the default authority CRID string for the given service. This string will be returned as broadcast and should be freed using STB_AppFreMemory.
- **void ADB_ServiceAddRCTLink** (void *serv_ptr, void *link_ptr)
Adds the given RCT link info to the end of the list of existing RCT links already defined for the given service.
- **void ADB_ServiceReleaseRCTLinks** (void *serv_ptr)
Frees all RCT link info for the given service.
- **U8BIT ADB_GetServiceNumRCTLinks** (void *serv_ptr)
Returns the number of RCT links for the given service.
- **void * ADB_GetServiceRCTLinks** (void *serv_ptr, U8BIT *num_links)
Returns a copy of the RCT links for the given service. The returned list should be freed using ADB_ReleaseRCTLinks.
- **void * ADB_GetNextRCTLink** (void *link_ptr)
Returns the RCT link following the given link.
- **void ADB_ReleaseRCTLinks** (void *links)
Frees the given list of RCT links.
- **U8BIT * ADB_GetRCTLinkPromoText** (void *link_ptr)
Returns the RCT link's promotional text based on the default language.
- **BOOLEAN ADB_IsRCTLinkGroupTrailer** (void *link_ptr)
Returns whether the given link is for a group trailer.
- **U8BIT * ADB_GetRCTLinkName** (void *link_ptr)

- Returns the name of the given RCT link.*

 - U8BIT * [ADB_GetRCTLinkUriString](#) (void *link_ptr)
- Returns the uri string of the given RCT link.*

 - BOOLEAN [ADB_ServiceRCTCanUseDefaultIcon](#) (void *serv_ptr)

Checks all the RCT links for the given service to determine whether any of them specify that the default icon can be used.
- BOOLEAN [ADB_ServiceGetRCTIcon](#) (void *serv_ptr, U8BIT **icon_data, U32-BIT *data_size, BOOLEAN *pos_valid, U16BIT *x_pos, U16BIT *y_pos, U16BIT *width, U16BIT *height, E_ICON_COORD_SYSTEM *coord_system)

Searches all the RCT links for a service to see if any of them define an icon to be used, and if one is found then all the data required to display the icon is returned.
- BOOLEAN [ADB_ServiceAddImageIcon](#) (void *serv_ptr, void *icon_ptr)

Adds the given image icon to the end of the service's icon list. The icon id is checked and if it matches an icon already in the list then the new icon replaces the existing one.
- void * [ADB_GetAlternativeService](#) (void *s_ptr, U8BIT alt_serv_type)

Searches for a replacement service of the given type for the given service.
- BOOLEAN [ADB_ServiceHasSubtitles](#) (void *serv_ptr, BOOLEAN *dvd_subs)

Determines whether the given service has subtitles, DVB or teletext.
- void [ADB_SetTunedService](#) (U8BIT path, void *s_ptr)

Sets the tuned service for the given path, together with the associated tuned transport and network. The 'new' flag for the service is also cleared.
- void * [ADB_GetTunedService](#) (U8BIT path)

Returns the tuned service for the given decode path.
- void [ADB_DeleteServiceRec](#) (void *s_ptr)

Deletes the given service from the database.
- U16BIT [ADB_GetNumStreams](#) (void *serv_ptr, ADB_STREAM_LIST_TYPE stream_list_type)

Returns the number of streams of the given type for the given service.
- void [ADB_GetStreamList](#) (void *serv_ptr, ADB_STREAM_LIST_TYPE stream_list_type, void ***streamlist_ptr, U16BIT *num_entries_ptr)

Allocates and returns an array of the streams of the given type and for the given service. The returned array should be freed using [ADB_ReleaseStreamList](#).
- void [ADB_ReleaseStreamList](#) (void **streamlist_ptr, U16BIT num_entries)

Frees the memory allocated for a stream list using [ADB_GetStreamList](#).
- ADB_STREAM_TYPE [ADB_GetStreamType](#) (void *stream_ptr)

Returns the type of the given stream.
- U8BIT [ADB_GetStreamNumTags](#) (void *stream_ptr)

Returns the number of tag values defined for the given stream.
- U8BIT [ADB_GetStreamTag](#) (void *stream_ptr, U8BIT index)

Returns the requested tag value of the given stream.
- U16BIT [ADB_GetStreamPID](#) (void *stream_ptr)

Returns the PID for the given stream.
- BOOLEAN [ADB_GetStreamInUse](#) (void *stream_ptr)

Returns whether the given stream is marked as being 'in use', which means it will be the stream being decoded for video, audio, etc.

- **BOOLEAN** [ADB_GetStreamHasCaDesc](#) (void *stream_ptr)
Returns whether the given stream has a CA descriptor.
- **U8BIT** [ADB_GetStreamComponentType](#) (void *stream_ptr)
Returns the component type as defined in the EITp/f's component descriptor for the given stream.
- **U32BIT** [ADB_GetAudioStreamLangCode](#) (void *stream_ptr)
Returns the language code associated with the given audio stream.
- **ADB_AUDIO_TYPE** [ADB_GetAudioStreamType](#) (void *stream_ptr)
Returns the audio type of the given stream.
- **E_STB_DP_AUDIO_MODE** [ADB_GetAudioStreamMode](#) (void *stream_ptr)
Returns the audio mode of the given stream.
- **U32BIT** [ADB_GetTextStreamLangCode](#) (void *stream_ptr)
Returns the 3 char language code of the given teletext stream.
- **U8BIT** [ADB_GetTextStreamType](#) (void *stream_ptr)
Returns the type, as defined in the PMT, of the given teletext stream.
- **U8BIT** [ADB_GetTextStreamMagazine](#) (void *stream_ptr)
Returns the magazine value of the given teletext stream.
- **U8BIT** [ADB_GetTextStreamPage](#) (void *stream_ptr)
Returns the page value of the given teletext stream.
- **U32BIT** [ADB_GetSubtitleStreamLangCode](#) (void *stream_ptr)
Returns the 3 char language code of the given subtitle stream.
- **ADB_SUBTITLE_TYPE** [ADB_GetSubtitleStreamType](#) (void *stream_ptr)
Returns the type, as defined in the PMT, of the given subtitle stream.
- **U16BIT** [ADB_GetSubtitleStreamCompositionPage](#) (void *stream_ptr)
Returns the composition page value of the given subtitle stream.
- **U16BIT** [ADB_GetSubtitleStreamAncillaryPage](#) (void *stream_ptr)
Returns the ancillary page value of the given subtitle stream.
- **void** [ADB_SetAudioLang](#) (U32BIT country_code, U8BIT lang_id)
Sets the primary audio language to be used.
- **void** [ADB_SetSecondaryAudioLang](#) (U32BIT country_code, U8BIT lang_id)
Sets the secondary audio language to be used. This will be selected if the primary audio language isn't available.
- **void** [ADB_SetTextLang](#) (U32BIT country_code, U8BIT lang_id)
Sets the primary subtitle and teletext language to be used.
- **void** [ADB_SetSecondaryTextLang](#) (U32BIT country_code, U8BIT lang_id)
Sets the secondary subtitle and teletext language to be used. This will be used if the primary language isn't available.
- **void** [ADB_GetNowNextEvents](#) (void *serv_ptr, void **now_event, void **next_event)
Makes copies of the now and/or next events (EITp/f) for the given service. The returned events should be freed using ADB_ReleaseEventData.
- **void** [ADB_GetEventSchedule](#) (BOOLEAN include_old_events, void *serv_ptr, void ***elist_ptr, U16BIT *num_entries_ptr)

Allocates and returns an array containing copies of events for the given service from the service's EIT schedule. The returned array should be freed using ADB_ReleaseEventList.

- void [ADB_GetEventScheduleByGenre](#) (ADB_EVENT_CONTENT genre, BOOLEAN include_old_events, void *serv_ptr, void ***elist_ptr, U16BIT *num_entries_ptr)

Creates a copy of the event schedule for the given service, but only includes events of the given genre type.

- void [ADB_ReleaseEventData](#) (void *event_ptr)

Frees any memory allocated for the given event and the event itself.

- void [ADB_ReleaseEventList](#) (void **elist, U16BIT num_entries)

Frees all the events in the given list and all associated memory for those events.

- void [ADB_DeleteServiceEvents](#) (void *serv_ptr, BOOLEAN delete_now_next)

Deletes all events and event related data for the given service.

- void [ADB_DeleteAllServiceEvents](#) (BOOLEAN delete_now_next)

Deletes all events and event related data for all services.

- void * [ADB_GetEvent](#) (void *serv_ptr, U16BIT event_id)

Returns a copy of the event with the given event ID on the given service.

- U8BIT * [ADB_GetEventSIDDescriptorData](#) (void *serv_ptr, U16BIT event_id, U8BIT dtag_id, S16BIT ext_dtag_id, U16BIT *desc_len)

Returns a copy of the raw SI descriptor data with the given descriptor tag id and, optionally, extended descriptor tag id, for the event with the given event id. The data must be freed using ADB_ReleaseEventSIDDescriptorData.

- void [ADB_ReleaseEventSIDDescriptorData](#) (U8BIT *desc_data, U16BIT desc_len)

Frees the data returned by ADB_GetEventSIDDescriptorData.

- void * [ADB_EarlierEvent](#) (void *serv_ptr, U32DHMS time)

Returns a copy of the event preceeding the given date/time on the given service.

- void * [ADB_LaterEvent](#) (void *serv_ptr, U32DHMS time)

Returns a copy of the event following the given date/time on the given service.

- void * [ADB_GetEarlierEvent](#) (void *serv_ptr, U16BIT date, U8BIT hour, U8BIT min)

Returns a copy of the event preceeding the given date/time on the given service.

- void * [ADB_GetLaterEvent](#) (void *serv_ptr, U16BIT date, U8BIT hour, U8BIT min)

Returns a copy of the event following the given date/time on the given service.

- void * [ADB_FindEventFromTime](#) (U32DHMS time, void *serv_ptr)

Finds the event scheduled to be broadcast at the specified time on the specified service.

- void * [ADB_FindEITScheduleEventFromTime](#) (U32DHMS time, void *serv_ptr)

Finds the event in the event schedule to be broadcast at the specified time on the specified service.

- U32DHMS [ADB_GetEventStartDateTime](#) (void *event_ptr)

Returns a value representing the date and time of the start of the given event.

- U32DHMS [ADB_GetEventDuration](#) (void *event_ptr)

- Returns a value representing the duration of the given event.*

 - U32DHMS [ADB_GetEventEndDateTime](#) (void *event_ptr)
- Returns a value representing the date and time of the end of the given event.*

 - U16BIT [ADB_GetEventStartDate](#) (void *event_ptr)
- Returns the MJD date code for the start of the given event.*

 - U8BIT [ADB_GetEventStartHour](#) (void *event_ptr)
- Returns the hour (0-23) value for the start of the given event.*

 - U8BIT [ADB_GetEventStartMin](#) (void *event_ptr)
- Returns the minute (0-59) value for the start of the given event.*

 - U8BIT [ADB_GetEventStartSecs](#) (void *event_ptr)
- Returns the seconds (0-59) value for the start of the given event.*

 - U8BIT [ADB_GetEventDurationHour](#) (void *event_ptr)
- Returns the hour value for the duration of the given event.*

 - U8BIT [ADB_GetEventDurationMin](#) (void *event_ptr)
- Returns the minute value (0-59) for the duration of the given event.*

 - U8BIT [ADB_GetEventDurationSecs](#) (void *event_ptr)
- Returns the seconds value (0-59) for the duration of the given event.*

 - BOOLEAN [ADB_GetEventSubtitlesAvailFlag](#) (void *event_ptr)
- Returns whether DVB subtitles are signalled as being available for the given event.*

 - ADB_EVENT_CONTENT [ADB_GetEventContentDesc](#) (void *event_ptr)
- Returns the level 1 value of the first content identifier for the given event.*

 - U8BIT * [ADB_GetEventContentData](#) (void *event_ptr, U8BIT *p_len)
- Returns the level 1 and 2 values for all content identifiers for the given event.*

 - U8BIT [ADB_GetEventParentalAge](#) (void *event_ptr)
- Returns the parental age value for the given event.*

 - U8BIT * [ADB_GetEventName](#) (void *event_ptr)
- Returns the name of the event as a UTF-8 string. The returned string should be freed using STB_ReleaseUnicodeString.*

 - U8BIT * [ADB_GetEventDescription](#) (void *event_ptr)
- Returns the short event description text of the event as a UTF-8 string. The returned string should be freed using STB_ReleaseUnicodeString.*

 - BOOLEAN [ADB_GetEventHasExtendedDescription](#) (void *event_ptr)
- Checks whether the given event has any extended event descriptors.*

 - U8BIT * [ADB_GetEventExtendedDescription](#) (void *event_ptr)
- Returns the extended event description text of the event as a UTF-8 string. The returned string should be freed using STB_ReleaseUnicodeString.*

 - U8BIT * [ADB_GetEventGuidance](#) (void *event_ptr, void *serv_ptr)
- Returns the guidance text for an event, either from the event itself or the event's service, as a UTF-8 string. The returned string should be freed using STB_ReleaseUnicodeString.*

 - U16BIT [ADB_GetEventId](#) (void *event_ptr)
- Returns the event id for the given event.*

 - BOOLEAN [ADB_GetEventAudioDescriptionFlag](#) (void *event_ptr)
- Returns whether audio description is signalled as being available for the given event.*

- **BOOLEAN** [ADB_GetEventFreeToAir](#) (void *event_ptr)
Returns whether the event is signalled with none of its streams being CA scrambled.
- **BOOLEAN** [ADB_GetEventDoNotScramble](#) (void *event_ptr)
Returns the do_not_scramble flag for the event, which will have been found by looking backwards in the event, SDT, transport and NIT, until found or not.
- **BOOLEAN** [ADB_GetEventHDLinkageInfo](#) (void *event_ptr, **BOOLEAN** verify_event, **BOOLEAN** only_simulcast, void **hd_serv_ptr, void **hd_event_ptr)
Returns whether there's an HD event linked to the SD event and returns the info for it. The DVB info is checked to ensure it relates to a known service.
- **BOOLEAN** [ADB_IsSameEvent](#) (void *event1_ptr, void *event2_ptr)
Checks whether the two events are the same. Not all fields are checked, just date, time, duration and event ID.
- **U8BIT *** [ADB_GetEventFullProgrammeCrid](#) (void *serv_ptr, void *event_ptr)
Returns the full programme CRID of the given event (including IMI). The returned string should be freed using STB_AppFreeMemory.
- **U8BIT *** [ADB_GetEventProgrammeCrid](#) (void *serv_ptr, void *event_ptr)
Returns the programme CRID of the given event (excluding IMI) The returned string should be freed using STB_AppFreeMemory.
- **U8BIT *** [ADB_GetFullCrid](#) (void *serv_ptr, **U8BIT ***event_str)
Returns the full CRID for the given CRID string The returned string should be freed using STB_AppFreeMemory.
- **U8BIT** [ADB_GetEventNumSeriesCrids](#) (void *event_ptr)
Returns the number of series CRIDs for the given event.
- **U8BIT *** [ADB_GetEventSeriesCrid](#) (**U8BIT** index, void *serv_ptr, void *event_ptr)
Returns the full series CRID of the given event The returned string should be freed using STB_AppFreeMemory.
- **U8BIT** [ADB_GetEventNumRecommendationCrids](#) (void *event_ptr)
Returns the number of recommendation CRIDs for the given event.
- **U8BIT *** [ADB_GetEventRecommendationCrid](#) (**U8BIT** index, void *serv_ptr, void *event_ptr)
Returns the full recommendation CRID of the given event The returned string should be freed using STB_AppFreeMemory.
- **U8BIT** [ADB_GetEventComponentList](#) (void *event_ptr, [ADB_EVENT_COMPONENT_INFO](#) **component_list)
Retrieves a list of components associated with the specified event, as described by component descriptors in the EIT.
- **BOOLEAN** [ADB_IsSplitProgrammeCrid](#) (**U8BIT ***crid)
Returns TRUE if the given CRID represents a split event (i.e. it contains an Instance Metadata Identifier (IMI)).
- **U8BIT *** [ADB_GetAuthorityFromCrid](#) (**U8BIT ***crid)
Returns a copy of the authority part of the CRID string The returned string should be freed using STB_AppFreeMemory.
- **void *** [ADB_FindEventFromCrid](#) (**U8BIT ***prog_crid, void *original_event_ptr, **BOOLEAN** radio_service, void **serv_ptr)

Finds an alternative event for the given programme CRID and returns a pointer to a copy of the event and a pointer to the service.

- U8BIT [ADB_GetServiceFavGroups](#) (void *s_ptr)

Returns the favourite group value for the given service. This is an 8-bit value that can define whether the service is in any of upto 8 favourite groups. The value is a bit flag, so 1 means it's in group 1, 2 means it's in group 2, 4 means it's in group 3, 7 means it's in groups 1, 2 and 3, etc.

- void [ADB_SetServiceFavGroups](#) (void *s_ptr, U8BIT groups)

Sets the favourite group value for the given service. This is an 8-bit value that can define whether the service is in any of upto 8 favourite groups. The value is a bit flag, so 1 means it's in group 1, 2 means it's in group 2, 4 means it's in group 3, 7 means it's in groups 1, 2 and 3, etc.

- void [ADB_SetReqdAudioStreamSettings](#) (void *s_ptr, BOOLEAN valid, U32BIT lang_code, ADB_AUDIO_TYPE audio_type, ADB_STREAM_TYPE stream_type)

Explicitly sets or clears the stream that will be used for audio on the given service. If 'valid' is TRUE then the stream is set according to the given language and types, but if FALSE then the audio stream selected for the service will be based on the default settings.

- void [ADB_GetReqdAudioStreamSettings](#) (void *s_ptr, BOOLEAN *valid, U32BIT *lang_code, ADB_AUDIO_TYPE *audio_type, ADB_STREAM_TYPE *stream_type)

Returns the settings defined by ADB_SetReqdAudioStreamSettings that will be used for audio on the given service.

- void [ADB_SetReqdSubtitleStreamSettings](#) (void *s_ptr, BOOLEAN valid, U32BIT lang_code, ADB_SUBTITLE_TYPE subt_type)

Explicitly sets or clears the stream that will be used for subtitles on the given service. If 'valid' is TRUE then the stream is set according to the given language and type, but if FALSE then the stream selected for the service will be based on the default settings.

- void [ADB_GetReqdSubtitleStreamSettings](#) (void *s_ptr, BOOLEAN *valid, U32BIT *lang_code, ADB_SUBTITLE_TYPE *subt_type)

Returns the settings defined by ADB_SetReqdSubtitleStreamSettings that will be used for subtitles on the given service.

- void [ADB_SetReqdTeletextStreamSettings](#) (void *s_ptr, BOOLEAN valid, U32BIT lang_code, ADB_TELETEXT_TYPE ttext_type)

Explicitly sets or clears the stream that will be used for teletext on the given service. If 'valid' is TRUE then the stream is set according to the given language and type, but if FALSE then the stream selected for the service will be based on the default settings.

- void [ADB_GetReqdTeletextStreamSettings](#) (void *s_ptr, BOOLEAN *valid, U32BIT *lang_code, ADB_TELETEXT_TYPE *ttext_type)

Returns the settings defined by ADB_SetReqdTeletextStreamSettings that will be used for teletext on the given service.

- void * [ADB_AddCridRecord](#) (U8BIT *crid, BOOLEAN series, BOOLEAN recommended, BOOLEAN radio_service)

Adds a CRID record to the database; use ADB_SaveCridRecord to store it.

- void [ADB_SaveCridRecord](#) (void *c_ptr)

Saves a CRID record to non-volatile storage.

- void * [ADB_FindCridRecord](#) (U8BIT *crid_str)

- Searches for and returns the CRID record with the given CRID string.*

 - void [ADB_SetCridDateTime](#) (void *c_ptr, U32DHMS datetime)

Sets the date and time fields on the given CRID record.

 - void [ADB_SetCridService](#) (void *c_ptr, U16BIT serv_id)

Sets the service ID on the given CRID record.

 - void [ADB_SetCridProgrammeName](#) (void *c_ptr, U8BIT *prog_name)

Sets the programme name field of the given CRID record.

 - U8BIT * [ADB_GetCridProgrammeName](#) (void *c_ptr)

Returns the programme name field of the given CRID record. The returned string should be freed using STB_AppFreeMemory.

 - BOOLEAN [ADB_IsProgrammeCrid](#) (void *c_ptr)

Returns TRUE if the CRID record given represents a programme (split event)

 - BOOLEAN [ADB_IsSeriesCrid](#) (void *c_ptr)

Returns TRUE if the CRID record given represents a series.

 - BOOLEAN [ADB_IsRecommendationCrid](#) (void *c_ptr)

Returns TRUE if the CRID record given represents a recommendation.

 - void [ADB_UpdateCridEitDate](#) (void *c_ptr)

Updates the time the CRID was last seen in the EIT.

 - BOOLEAN [ADB_IsCridForRadioService](#) (void *c_ptr)

Returns whether the CRID record is for a radio service (TRUE)

 - U8BIT * [ADB_GetCridString](#) (void *c_ptr)

Returns the CRID string from the given CRID record. The returned value should not be freed.

 - U32DHMS [ADB_GetCridDateTime](#) (void *c_ptr)

Returns a value representing the date & time held in the given CRID record.

 - U16BIT [ADB_GetCridService](#) (void *c_ptr)

Returns the service ID held in the given CRID record.

 - U16BIT [ADB_GetCridEitDate](#) (void *c_ptr)

Returns the EIT update date contained in the given CRID record.

 - void [ADB_DeleteCridRecord](#) (void *c_ptr)

Deletes the given CRID record from the database.

 - void [ADB_GetCridRecordList](#) (void ***clist_ptr, U16BIT *num_entries_ptr, BOOLEAN inc_series_crids, BOOLEAN inc_rec_crids, BOOLEAN inc_prog_crids)

Creates an array of CRID record pointers to pass back to the caller with the number of entries in the array. Memory is allocated from temporary memory, so will be released when the calling screen closes.

 - void [ADB_ReleaseCridRecordList](#) (void **crid_list, U16BIT num_entries)

Frees memory allocated for a CRID list due to a call to ADB_GetCridRecordList.

 - U16BIT [ADB_GetProfileList](#) (void ***profile_list, U16BIT *active_profile)

Returns an array of available profiles. The array will be allocated within the function and should be released using ADB_ReleaseProfileList. This is a CI+ feature.

 - void [ADB_ReleaseProfileList](#) (void **profile_list, U16BIT num_profiles)

Frees a profile list acquired using ADB_GetProfileList.

 - ADB_PROFILE_TYPE [ADB_GetProfileType](#) (void *profile)

- Returns the type of the given profile.*

 - U8BIT * [ADB_GetServicePMTData](#) (void *s_ptr, U16BIT *data_len)

Returns the current PMT data for the given service.
- U16BIT [ADB_GetServicePmtPid](#) (void *s_ptr)

Returns the current PMT PID for the given service.
- BOOLEAN [ADB_ImportDB](#) (U8BIT *filename)

Imports the database from an XML file.
- BOOLEAN [ADB_ExportDB](#) (U8BIT *filename)

Exports the database as an XML file.
- BOOLEAN [ADB_ExportEPG](#) (U8BIT *filename, U16BIT utc_date_filter, U32BIT type, BOOLEAN use_dvb_uri)

Exports the current event schedule as an XML file.
- U16BIT [ADB_GetNumFavouriteLists](#) (void)

Returns the number of favourite lists.
- BOOLEAN [ADB_AddFavouriteList](#) (U8BIT *name, U32BIT user_data, S16BIT index, U8BIT *list_id)

Creates a new favourite list and adds it to the existing lists, if any.
- U8BIT [ADB_GetFavouriteListIdByIndex](#) (U16BIT index)

Returns the list id of the favourite list defined by the given index.
- U8BIT [ADB_GetFavouriteListByUserData](#) (U32BIT user_data)

Returns the list id of the favourite list with the given user data value.
- U8BIT * [ADB_GetFavouriteListName](#) (U8BIT list_id)

Returns the name of the given favourite list, as a Unicode string.
- BOOLEAN [ADB_SetFavouriteListName](#) (U8BIT list_id, U8BIT *name)

Set the name of the given favourite list.
- U32BIT [ADB_GetFavouriteListUserData](#) (U8BIT list_id)

Returns the user data value of the given favourite list.
- void [ADB_SetFavouriteListUserData](#) (U8BIT list_id, U32BIT user_data)

Set the user data of the given favourite list.
- BOOLEAN [ADB_MoveFavouriteListTo](#) (U8BIT list_id, S16BIT index)

Change the order of the favourite lists by moving the given list to the given position.
- void [ADB_DeleteFavouriteList](#) (U8BIT list_id)

Deletes the favourite list with the given list id.
- BOOLEAN [ADB_AddServiceToFavouriteList](#) (U8BIT list_id, void *serv_ptr, S16BIT index)

Adds the given service to the favourite list defined by list_id, with the service being optionally added at a given position.
- BOOLEAN [ADB_MoveFavouriteListServiceTo](#) (U8BIT list_id, void *serv_ptr, S16BIT index)

Change the order of the services in the given favourite lists by moving it to the given position.
- BOOLEAN [ADB_MoveFavouriteListServiceFromTo](#) (U8BIT list_id, S16BIT current_index, S16BIT new_index)

Change the order of the services in the given favourite lists by moving it to the given position.

- void [ADB_DeleteServiceFromFavouriteList](#) (U8BIT list_id, void *serv_ptr)
Removes the given service from the favourite list defined by the list id.
- void [ADB_DeleteAllServicesFromFavouriteList](#) (U8BIT list_id)
Removes all services from the favourite list defined by list_id.
- void [ADB_Initialise](#) (void)
Initialises database access.
- void [ADB_PrepareDatabaseForSearch](#) (E_STB_DP_SIGNAL_TYPE tuner_type, void *satellite, BOOLEAN retune, BOOLEAN manual_search)
Sets up the database for a search.
- void [ADB_FinaliseDatabaseAfterSearch](#) (BOOLEAN save_changes, E_STB_DP_SIGNAL_TYPE tuner_type, void *satellite, BOOLEAN search_completed, BOOLEAN clear_new_flags, BOOLEAN manual_search)
Completes the database setup after a search has completed or been stopped.
- BOOLEAN [ADB_AreLcnsDuplicated](#) (E_STB_DP_SIGNAL_TYPE tuner_type)
Checks through all services after a service search and determines whether any of them require the same LCN. This function assumes that ADB_FinaliseDatabaseAfterSearch has already been called.
- void [ADB_AllocateLcns](#) (E_STB_DP_SIGNAL_TYPE tuner_type)
Allocates LCNs to services after a service search. This function assumes that ADB_FinaliseDatabaseAfterSearch has already been called.
- void [ADB_ReleaseDatabaseSearchData](#) (void)
Frees all data that's only required for service search. This should be called after the search is complete and LCNs have been assigned, etc.
- void [ADB_ReportNoSignalDuringSearch](#) (void *t_ptr)
Set the signal level to 0 and searched flag to TRUE for the given transport.
- BOOLEAN [ADB_GetTransportSearchFlag](#) (void *t_ptr)
Returns whether a transport has been used during a service scan operation.
- void [ADB_SetTransportSearchFlag](#) (void *t_ptr, BOOLEAN state)
Sets whether a transport has been used during a service scan operation.
- E_STB_DP_AUDIO_CODEC [ADB_GetAudioCodecFromStream](#) (ADB_STREAM_TYPE audio_stream_type)
Derive audio codec from stream.
- E_STB_DP_VIDEO_CODEC [ADB_GetVideoCodecFromStream](#) (ADB_STREAM_TYPE video_stream_type)
Derive video codec from stream.
- void * [ADB_GetEventService](#) (void *event_ptr)
Returns the service pointer of event.
- [ADB_EVENT_ITEMIZED_INFO](#) * [ADB_GetEventItemizedDescription](#) (void *event_ptr, U16BIT *num_items_ptr)
Returns the items of extended event descriptor as item descriptor and item itself.
- void [ADB_ReleaseEventItemizedInfo](#) ([ADB_EVENT_ITEMIZED_INFO](#) *event_itemized_info, U16BIT num_of_items)
Frees the memory used by event itemized data.

4.10.1 Detailed Description

Application database access functions.

Date

24/03/2003

4.10.2 Function Documentation

4.10.2.1 void* **ADB_AddCridRecord** (U8BIT * *crid*, BOOLEAN *series*, BOOLEAN *recommended*, BOOLEAN *radio_service*)

Adds a CRID record to the database; use ADB_SaveCridRecord to store it.

Parameters

<i>crid</i>	CRID string to be added
<i>series</i>	TRUE if this is a series CRID
<i>recommended</i>	TRUE if this is a recommended programme CRID
<i>radio_service</i>	TRUE if this CRID is for a radio service, TV otherwise

Returns

CRID record handle, NULL if creation fails

4.10.2.2 **BOOLEAN ADB_AddFavouriteList** (U8BIT * *name*, U32BIT *user_data*, S16BIT *index*, U8BIT * *list_id*)

Creates a new favourite list and adds it to the existing lists, if any.

Parameters

<i>name</i>	name to be given to the list
<i>user_data</i>	user defined value to be stored in the list
<i>index</i>	0 based index where the new list should be inserted in any existing list. 0 = at start, negative = at end, any other value = position
<i>list_id</i>	used to get the unique id assigned to the list, can be NULL

Returns

TRUE if the list is created successfully, FALSE otherwise

4.10.2.3 void* **ADB_AddLNB** (ADB_LNB_SETTINGS * *settings*)

Creates a new LNB in the database with the given settings.

Parameters

<i>settings</i>	settings to be given to the new LNB
-----------------	-------------------------------------

Returns

pointer to the new LNB, NULL if creation fails

4.10.2.4 void* **ADB_AddSatellite** (U8BIT * *name_str*, U16BIT *dish_pos*, U16BIT *long_pos*, BOOLEAN *east_west*, void * *lnb_ptr*)

Creates a new satellite in the database with the given settings.

Parameters

<i>name_str</i>	name to be given to the satellite, must be a DVB format string
<i>dish_pos</i>	
<i>long_pos</i>	longitudinal position of the satellite in 1/10ths degree (e.g. 19.2 would be 192)
<i>east_west</i>	whether the position of defined from east (TRUE) or west (FALSE)
<i>lnb_ptr</i>	pointer to an LNB already present in the database

Returns

pointer to new satellite

4.10.2.5 BOOLEAN **ADB_AddServiceToFavouriteList** (U8BIT *list_id*, void * *serv_ptr*, S16BIT *index*)

Adds the given service to the favourite list defined by *list_id*, with the service being optionally added at a given position.

Parameters

<i>list_id</i>	favourite list id
<i>serv_ptr</i>	service to be added
<i>index</i>	position to add the service in the list, 0 = at start, negative = at end, any other value is the position

Returns

TRUE if the service is successfully added, FALSE otherwise

4.10.2.6 BOOLEAN **ADB_AreLcnsDuplicated** (E_STB_DP_SIGNAL_TYPE *tuner_type*)

Checks through all services after a service search and determines whether any of them require the same LCN. This function assumes that **ADB_FinaliseDatabaseAfterSearch** has already been called.

Returns

TRUE if duplicates are found, FALSE otherwise

4.10.2.7 void ADB_DeleteAllServiceEvents (BOOLEAN *delete_now_next*)

Deletes all events and event related data for all services.

Parameters

<i>delete_now- _next</i>	specifies whether the now/next events should also be deleted
------------------------------	--

4.10.2.8 void ADB_DeleteAllServicesFromFavouriteList (U8BIT *list_id*)

Removes all services from the favourite list defined by *list_id*.

Parameters

<i>list_id</i>	favourite list id
----------------	-------------------

4.10.2.9 void ADB_DeleteCridRecord (void * *c_ptr*)

Deletes the given CRID record from the database.

Parameters

<i>c_ptr</i>	pointer to CRID record
--------------	------------------------

4.10.2.10 void ADB_DeleteFavouriteList (U8BIT *list_id*)

Deletes the favourite list with the given list id.

Parameters

<i>list_id</i>	id of the favourite list to be deleted
----------------	--

4.10.2.11 void ADB_DeleteServiceEvents (void * *serv_ptr*, BOOLEAN *delete_now_next*)

Deletes all events and event related data for the given service.

Parameters

<i>serv_ptr</i>	service for the event
<i>delete_now- _next</i>	specifies whether the now/next events should also be deleted

4.10.2.12 void ADB_DeleteServiceFromFavouriteList (U8BIT *list_id*, void * *serv_ptr*)

Removes the given service from the favourite list defined by the list id.

Parameters

<i>list_id</i>	favourite list id
<i>serv_ptr</i>	service to be removed from the list

4.10.2.13 void ADB_DeleteServiceRec (void * *s_ptr*)

Deletes the given service from the database.

Parameters

<i>s_ptr</i>	service to be deleted
--------------	-----------------------

4.10.2.14 void ADB_DeleteServices (E_STB_DP_SIGNAL_TYPE *tuner_type*, void * *satellite*)

Deletes all networks, transports and services related to the given type of tuner.

Parameters

<i>tuner_type</i>	tuner type
<i>satellite</i>	if <i>tuner_type</i> is SIGNAL_QPSK then only services on the given satellite will be deleted. If NULL, services on all satellite will be deleted

4.10.2.15 void* ADB_EarlierEvent (void * *serv_ptr*, U32DHMS *time*)

Returns a copy of the event preceeding the given date/time on the given service.

Parameters

<i>serv_ptr</i>	service for the event
<i>time</i>	Date/time from which to start searching

Returns

pointer to a copy of the event, or NULL if an event isn't found

4.10.2.16 BOOLEAN ADB_ExportDB (U8BIT * *filename*)

Exports the database as an XML file.

Parameters

<i>filename</i>	full pathname of the file to export to
-----------------	--

Returns

TRUE if successful, otherwise FALSE

4.10.2.17 `BOOLEAN ADB_ExportEPG (U8BIT * filename, U16BIT utc_date_filter, U32BIT type, BOOLEAN use_dvb_uri)`

Exports the current event schedule as an XML file.

Parameters

<i>filename</i>	full pathname of the file to export to
<i>utc_date_filter</i>	If non-zero, only events matching date will be exported
<i>type</i>	The service type(s) to export
<i>use_dvb_uri</i>	TRUE to use dvb:// uri for id, otherwise allocated lcn

Returns

TRUE if successful, otherwise FALSE

4.10.2.18 `void ADB_FinaliseDatabaseAfterSearch (BOOLEAN save_changes, E_STB_DP_SIGNAL_TYPE tuner_type, void * satellite, BOOLEAN search_completed, BOOLEAN clear_new_flags, BOOLEAN manual_search)`

Completes the database setup after a search has completed or been stopped.

Parameters

<i>save_changes</i>	TRUE if the modified database is to be saved
<i>tuner_type</i>	type of tuner that was used when doing the search
<i>satellite</i>	satellite that the search was performed on, NULL if not relevant
<i>search_completed</i>	TRUE if the search was completed, FALSE if terminated before completion
<i>clear_new_flags</i>	TRUE if all new flags are to be cleared (e.g. after a full search rather than an update search)
<i>manual_search</i>	TRUE for manual searches, FALSE otherwise

4.10.2.19 `void* ADB_FindCridRecord (U8BIT * crid_str)`

Searches for and returns the CRID record with the given CRID string.

Parameters

<i>crid_str</i>	pointer to CRID string
-----------------	------------------------

Returns

pointer to the CRID record if one is found, or NULL

4.10.2.20 void* **ADB_FindEITScheduleEventFromTime** (U32DHMS *time*, void * *serv_ptr*)

Finds the event in the event schedule to be broadcast at the specified time on the specified service.

Parameters

<i>time</i>	date/time when the event is scheduled for broadcast
<i>serv_ptr</i>	handle of the service on which the event is scheduled for broadcast

Note

searches the complete schedule including old events

4.10.2.21 void* **ADB_FindEventFromCrid** (U8BIT * *prog_crid*, void * *original_event_ptr*, BOOLEAN *radio_service*, void ** *serv_ptr*)

Finds an alternative event for the given programme CRID and returns a pointer to a copy of the event and a pointer to the service.

Parameters

<i>prog_crid</i>	programme CRID to search for
<i>original_event_ptr</i>	pointer to the event for which an alternative is being sought, can be NULL
<i>radio_service</i>	TRUE if the search should be made on radio services, FALSE otherwise
<i>serv_ptr</i>	pointer to return the service record if an alternate event is found

Returns

pointer to a copy of the event, or NULL

4.10.2.22 void* **ADB_FindEventFromTime** (U32DHMS *time*, void * *serv_ptr*)

Finds the event scheduled to be broadcast at the specified time on the specified service.

Parameters

<i>time</i>	date/time when the event is scheduled for broadcast
<i>serv_ptr</i>	handle of the service on which the event is scheduled for broadcast

Returns

event handler, NULL if not found

Note

the first two events searched will be the now/next events, followed by the remaining schedule

4.10.2.23 U16BIT ADB_FindLcnInServiceList (U16BIT *lcn*, void ** *slist_ptr*, U16BIT *num_entries*)

Returns the index into the given list of services of the service with the given LCN.

Parameters

<i>lcn</i>	logical channel number to search for
<i>slist_ptr</i>	list of services to be searched
<i>num_entries</i>	number of services in the list

Returns

index of the service, or num_entries if service isn't found

4.10.2.24 void* ADB_FindLNBWithSettings (ADB_LNB_SETTINGS * *settings*)

Searches the database to find an existing LNB with the given settings.

Parameters

<i>settings</i>	LNB settings
-----------------	--------------

Returns

pointer to the LNB, or NULL if not found

4.10.2.25 U16BIT ADB_FindNearestLcnInServiceList (U16BIT *lcn*, void ** *slist_ptr*, U16BIT *num_entries*)

Returns the index in the given list of services of the service matching the given, or preceding, LCN.

Parameters

<i>lcn</i>	logical channel number
<i>slist_ptr</i>	list of services to be searched
<i>num_entries</i>	number of services in the list

Returns

index of the service, or num_entries if service isn't found

4.10.2.26 void* ADB_FindServiceByIds (U16BIT *onet_id*, U16BIT *tid*, U16BIT *sid*)

Returns a pointer to the service matching the given IDs.

Parameters

<i>onet_id</i>	original network id
<i>tid</i>	transport id
<i>sid</i>	service id

Returns

service pointer, or NULL if not found

4.10.2.27 void* ADB_FindServiceByLcn (U32BIT *list_type*, U16BIT *lcn*, BOOLEAN *show_unselectable*)

Returns the service matching the given LCN and list type.

Parameters

<i>list_type</i>	defines the types of services to be included, multiple types can be combined by ORing the values
<i>lcn</i>	logical channel number
<i>show_unselectable</i>	used by MHEG to allow it to find services that would normally not be selectable by a user

Returns

service, or NULL if not found

4.10.2.28 void* ADB_FindServiceByNearestLcn (U32BIT *list_type*, U16BIT *lcn*)

Returns the service matching the given, or preceding, LCN and list type.

Parameters

<i>list_type</i>	defines the types of services to be included, multiple types can be combined by ORing the values
<i>lcn</i>	logical channel number

Returns

service, or NULL if not found

4.10.2.29 U16BIT ADB_FindServiceInList (void * *s_ptr*, void ** *slist_ptr*, U16BIT *num_entries*)

Returns the index in the given list of services of the given service.

Parameters

<i>s_ptr</i>	service to search for
<i>slist_ptr</i>	list of services to be searched
<i>num_entries</i>	number of services in the list

Returns

index of the service, or num_entries if service isn't found

4.10.2.30 void* ADB_GetAlternativeService (void * *s_ptr*, U8BIT *alt_serv_type*)

Searches for a replacement service of the given type for the given service.

Parameters

<i>s_ptr</i>	service that a replacement is required for
<i>alt_serv_type</i>	type of replacement service to look for (e.g. CA replacement)

Returns

alternative service found, or NULL

4.10.2.31 E_STB_DP_AUDIO_CODEC ADB_GetAudioCodecFromStream (ADB_STREAM_TYPE *audio_stream_type*)

Derive audio codec from stream.

Parameters

<i>audio_stream_type</i>	from which to derive the audio codec
--------------------------	--------------------------------------

Returns

audio codec best corresponding to audio_stream_type

4.10.2.32 U32BIT ADB_GetAudioStreamLangCode (void * *stream_ptr*)

Returns the language code associated with the given audio stream.

Parameters

<i>stream_ptr</i>	stream
-------------------	--------

Returns

3 char language code encoded as a 32-bit number

4.10.2.33 E_STB_DP_AUDIO_MODE ADB_GetAudioStreamMode (void * *stream_ptr*)

Returns the audio mode of the given stream.

Parameters

<i>stream_ptr</i>	stream
-------------------	--------

Returns

audio mode

4.10.2.34 ADB_AUDIO_TYPE ADB_GetAudioStreamType (void * *stream_ptr*)

Returns the audio type of the given stream.

Parameters

<i>stream_ptr</i>	stream
-------------------	--------

Returns

audio type

4.10.2.35 U8BIT* ADB_GetAuthorityFromCrid (U8BIT * *crid*)

Returns a copy of the authority part of the CRID string The returned string should be freed using STB_AppFreeMemory.

Parameters

<i>crid</i>	CRID string
-------------	-------------

Returns

authority string

4.10.2.36 U32DHMS ADB_GetCridDateTime (void * *c_ptr*)

Returns a value representing the date & time held in the given CRID record.

Parameters

<i>c_ptr</i>	pointer to CRID record
--------------	------------------------

Returns

date/time value

4.10.2.37 U16BIT ADB_GetCridEitDate (void * *c_ptr*)

Returns the EIT update date contained in the given CRID record.

Parameters

<i>c_ptr</i>	pointer to CRID record
--------------	------------------------

Returns

GMT date from the record

4.10.2.38 U8BIT* ADB_GetCridProgrammeName (void * *c_ptr*)

Returns the programme name field of the given CRID record. The returned string should be freed using STB_AppFreeMemory.

Parameters

<i>c_ptr</i>	pointer to CRID record
--------------	------------------------

Returns

pointer to the programme name

4.10.2.39 void ADB_GetCridRecordList (void *** *clist_ptr*, U16BIT * *num_entries_ptr*, BOOLEAN *inc_series_crids*, BOOLEAN *inc_rec_crids*, BOOLEAN *inc_prog_crids*)

Creates an array of CRID record pointers to pass back to the caller with the number of entries in the array. Memory is allocated from temporary memory, so will be released when the calling screen closes.

Parameters

<i>clist_ptr</i>	address for the return of the CRID array
<i>num_entries_ptr</i>	address for the return of the number of entries
<i>inc_series_crids</i>	TRUE if returned list is to include any series CRIDs
<i>inc_rec_crids</i>	TRUE if returned list is to include any recommendation CRIDs
<i>inc_prog_crids</i>	TRUE if returned list is to include any programme CRIDs

4.10.2.40 U16BIT ADB_GetCridService (void * *c_ptr*)

Returns the service ID held in the given CRID record.

Parameters

<i>c_ptr</i>	pointer to CRID record
--------------	------------------------

Returns

service id, or ADB_INVALID_DVB_ID if crid not valid

4.10.2.41 U8BIT* ADB_GetCridString (void * *c_ptr*)

Returns the CRID string from the given CRID record. The returned value should not be freed.

Parameters

<i>c_ptr</i>	pointer to CRID record
--------------	------------------------

Returns

pointer to the CRID string

4.10.2.42 void* ADB_GetEarlierEvent (void * *serv_ptr*, U16BIT *date*, U8BIT *hour*, U8BIT *min*)

Returns a copy of the event preceeding the given date/time on the given service.

Parameters

<i>serv_ptr</i>	service for the event
<i>date</i>	MJD date code
<i>hour</i>	hour, 0-23
<i>min</i>	minute, 0-59

Returns

pointer to a copy of the event, or NULL if an event isn't found

4.10.2.43 void* ADB_GetEvent (void * *serv_ptr*, U16BIT *event_id*)

Returns a copy of the event with the given event ID on the given service.

Parameters

<i>serv_ptr</i>	service for the event
<i>event_id</i>	ID of the event to be returned

Returns

pointer to a copy of the event, or NULL if the event isn't found

4.10.2.44 **BOOLEAN** **ADB_GetEventAudioDescriptionFlag** (void * *event_ptr*)

Returns whether audio description is signalled as being available for the given event.

Parameters

<i>event_ptr</i>	event
------------------	-------

Returns

TRUE if available, FALSE otherwise

4.10.2.45 **U8BIT** **ADB_GetEventComponentList** (void * *event_ptr*,
ADB_EVENT_COMPONENT_INFO ** *component_list*)

Retrieves a list of components associated with the specified event, as described by component descriptors in the EIT.

Parameters

<i>event_ptr</i>	Pointer to the event
<i>component-list</i>	Pointer to the returned list of components. The list must be freed by calling STB_AppFreeMemory, provided the returned number of components is greater than 0.

Returns

Number of components associated with the specified event.

4.10.2.46 **U8BIT*** **ADB_GetEventContentData** (void * *event_ptr*, **U8BIT** * *p_len*)

Returns the level 1 and 2 values for all content identifiers for the given event.

Parameters

<i>event_ptr</i>	event
<i>p_len</i>	address to return the number of bytes in the returned data

Returns

pointer to the content data

4.10.2.47 **ADB_EVENT_CONTENT** **ADB_GetEventContentDesc** (void * *event_ptr*)

Returns the level 1 value of the first content identifier for the given event.

Parameters

<i>event_ptr</i>	event
------------------	-------

Returns

event content value, ADB_EVENT_CONTENT_UNCLASSIFIED if not available

4.10.2.48 U8BIT* ADB_GetEventDescription (void * *event_ptr*)

Returns the short event description text of the event as a UTF-8 string. The returned string should be freed using STB_ReleaseUnicodeString.

Parameters

<i>event_ptr</i>	event
------------------	-------

Returns

pointer to UTF-8 string, NULL on error

4.10.2.49 BOOLEAN ADB_GetEventDoNotScramble (void * *event_ptr*)

Returns the do_not_scramble flag for the event, which will have been found by looking backwards in the event, SDT, transport and NIT, until found or not.

Parameters

<i>event_ptr</i>	pointer to the event to check
------------------	-------------------------------

Returns

FALSE if the event is to be protected, TRUE otherwise

4.10.2.50 U32DHMS ADB_GetEventDuration (void * *event_ptr*)

Returns a value representing the duration of the given event.

Parameters

<i>event_ptr</i>	event
------------------	-------

Returns

duration

4.10.2.51 U8BIT ADB_GetEventDurationHour (void * *event_ptr*)

Returns the hour value for the duration of the given event.

Parameters

<i>event_ptr</i>	event
------------------	-------

Returns

hour value

4.10.2.52 U8BIT ADB_GetEventDurationMin (void * *event_ptr*)

Returns the minute value (0-59) for the duration of the given event.

Parameters

<i>event_ptr</i>	event
------------------	-------

Returns

minute value (0-59)

4.10.2.53 U8BIT ADB_GetEventDurationSecs (void * *event_ptr*)

Returns the seconds value (0-59) for the duration of the given event.

Parameters

<i>event_ptr</i>	event
------------------	-------

Returns

seconds value (0-59)

4.10.2.54 U32DHMS ADB_GetEventEndDateTime (void * *event_ptr*)

Returns a value representing the date and time of the end of the given event.

Parameters

<i>event_ptr</i>	event
------------------	-------

Returns

date/time

4.10.2.55 U8BIT* ADB_GetEventExtendedDescription (void * *event_ptr*)

Returns the extended event description text of the event as a UTF-8 string. The returned string should be freed using STB_ReleaseUnicodeString.

Parameters

<i>event_ptr</i>	event
------------------	-------

Returns

pointer to UTF-8 string, NULL on error

4.10.2.56 **BOOLEAN ADB_GetEventFreeToAir (void * *event_ptr*)**

Returns whether the event is signalled with none of its streams being CA scrambled.

Parameters

<i>event_ptr</i>	pointer to the event to check
------------------	-------------------------------

Returns

TRUE if the event all streams are free-to-air, FALSE otherwise

4.10.2.57 **U8BIT* ADB_GetEventFullProgrammeCrid (void * *serv_ptr*, void * *event_ptr*)**

Returns the full programme CRID of the given event (including IMI). The returned string should be freed using STB_AppFreeMemory.

Parameters

<i>serv_ptr</i>	service for the event
<i>event_ptr</i>	event for which the CRID is to be returned

Returns

pointer to a string containing the CRID name, or NULL if there isn't a programme CRID

4.10.2.58 **U8BIT* ADB_GetEventGuidance (void * *event_ptr*, void * *serv_ptr*)**

Returns the guidance text for an event, either from the event itself or the event's service, as a UTF-8 string. The returned string should be freed using STB_ReleaseUnicodeString.

Parameters

<i>event_ptr</i>	event, can be NULL, in which case any guidance defined by the service will be returned
<i>serv_ptr</i>	service, can be NULL to just get guidance text from the event

Returns

pointer to UTF-8 guidance string, NULL if there isn't any guidance text

4.10.2.59 **BOOLEAN ADB_GetEventHasExtendedDescription (void * *event_ptr*)**

Checks whether the given event has any extended event descriptors.

Parameters

<i>event_ptr</i>	event
------------------	-------

Returns

TRUE if the event has at least one extended event descriptor, FALSE otherwise

4.10.2.60 **BOOLEAN ADB_GetEventHDLinkageInfo (void * *event_ptr*, BOOLEAN *verify_event*, BOOLEAN *only_simulcast*, void ** *hd_serv_ptr*, void ** *hd_event_ptr*)**

Returns whether there's an HD event linked to the SD event and returns the info for it. The DVB info is checked to ensure it relates to a known service.

Parameters

<i>event_ptr</i>	pointer to the event to check
<i>verify_event</i>	TRUE, if the schedule should be checked for the target event
<i>only_simulcast</i>	TRUE, if only an HD simulcast event should be returned
<i>hd_serv_ptr</i>	pointer to return the service for the HD event, if found. This can be NULL if not required.
<i>hd_event_ptr</i>	pointer to return the HD event, if found. Note that this will only be filled in if <i>verify_event</i> is TRUE. This can be NULL if not required.

Returns

TRUE if an event is found, FALSE otherwise

4.10.2.61 **U16BIT ADB_GetEventId (void * *event_ptr*)**

Returns the event id for the given event.

Parameters

<i>event_ptr</i>	event
------------------	-------

Returns

event id

4.10.2.62 ADB_EVENT_ITEMIZED_INFO* ADB_GetEventItemizedDescription (void * *event_ptr*, U16BIT * *num_items_ptr*)

Returns the items of extended event descriptor as item descriptor and item itself.

Parameters

<i>event_ptr</i>	event
<i>num_items_ptr</i>	address in which to return the number of items

Returns

[ADB_EVENT_ITEMIZED_INFO](#) pointer to the list of items, NULL on error

4.10.2.63 U8BIT* ADB_GetEventName (void * *event_ptr*)

Returns the name of the event as a UTF-8 string. The returned string should be freed using STB_ReleaseUnicodeString.

Parameters

<i>event_ptr</i>	event
------------------	-------

Returns

pointer to UTF-8 string, NULL on error

4.10.2.64 U8BIT ADB_GetEventNumRecommendationCrids (void * *event_ptr*)

Returns the number of recommendation CRIDs for the given event.

Parameters

<i>event_ptr</i>	pointer to the event
------------------	----------------------

Returns

number of recommendation CRIDs

4.10.2.65 U8BIT ADB_GetEventNumSeriesCrids (void * *event_ptr*)

Returns the number of series CRIDs for the given event.

Parameters

<i>event_ptr</i>	pointer to the event
------------------	----------------------

Returns

number of series CRIDs

4.10.2.66 U8BIT ADB_GetEventParentalAge (void * *event_ptr*)

Returns the parental age value for the given event.

Parameters

<i>event_ptr</i>	event
------------------	-------

Returns

parental age value, 0 for invalid event

4.10.2.67 U8BIT* ADB_GetEventProgrammeCrid (void * *serv_ptr*, void * *event_ptr*)

Returns the programme CRID of the given event (excluding IMI) The returned string should be freed using STB_AppFreeMemory.

Parameters

<i>serv_ptr</i>	service for the event
<i>event_ptr</i>	event for which the CRID is to be returned

Returns

pointer to a string containing the CRID name, or NULL if there isn't a programme CRID

4.10.2.68 U8BIT* ADB_GetEventRecommendationCrid (U8BIT *index*, void * *serv_ptr*, void * *event_ptr*)

Returns the full recommendation CRID of the given event The returned string should be freed using STB_AppFreeMemory.

Parameters

<i>index</i>	0 based index of recommendation CRID to be returned
<i>serv_ptr</i>	service for the event
<i>event_ptr</i>	event for which the CRID is to be returned

Returns

pointer to a string containing the CRID name, or NULL if there isn't a CRID

4.10.2.69 void **ADB_GetEventSchedule** (*BOOLEAN include_old_events*, void * *serv_ptr*, void *** *elist_ptr*, U16BIT * *num_entries_ptr*)

Allocates and returns an array containing copies of events for the given service from the service's EIT schedule. The returned array should be freed using ADB_ReleaseEventList.

Parameters

<i>include_old_events</i>	if TRUE, all events in the services schedule will be returned, but if FALSE then the first two events in the returned list will be the now/next events, followed by the remaining schedule
<i>serv_ptr</i>	service to get events for
<i>elist_ptr</i>	address in which to return a pointer to the allocated array of events
<i>num_entries_ptr</i>	address in which to return the number of events in the list

4.10.2.70 void **ADB_GetEventScheduleByGenre** (*ADB_EVENT_CONTENT genre*, *BOOLEAN include_old_events*, void * *serv_ptr*, void *** *elist_ptr*, U16BIT * *num_entries_ptr*)

Creates a copy of the event schedule for the given service, but only includes events of the given genre type.

Parameters

<i>genre</i>	event genre to be matched
<i>include_old_events</i>	if FALSE, the first events returned will be the now/next events, if they match the genre, otherwise events that may have expired will also be included
<i>serv_ptr</i>	service
<i>elist_ptr</i>	returned with an allocated array of events
<i>num_entries_ptr</i>	returned with the number of entries in the returned array

4.10.2.71 U8BIT* **ADB_GetEventSeriesCrid** (*U8BIT index*, void * *serv_ptr*, void * *event_ptr*)

Returns the full series CRID of the given event The returned string should be freed using STB_AppFreeMemory.

Parameters

<i>index</i>	0 based index of series CRID to be returned
<i>serv_ptr</i>	service for the event
<i>event_ptr</i>	event for which the CRID is to be returned

Returns

pointer to a string containing the CRID name, or NULL if there isn't a series CRID

4.10.2.72 void* ADB_GetEventService (void * event_ptr)

Returns the service pointer of event.

Parameters

<i>event_ptr</i>	event pointer
------------------	---------------

Returns

service pointer of event

4.10.2.73 U8BIT* ADB_GetEventSIDDescriptorData (void * serv_ptr, U16BIT event_id, U8BIT dtag_id, S16BIT ext_dtag_id, U16BIT * desc_len)

Returns a copy of the raw SI descriptor data with the given descriptor tag id and, optionally, extended descriptor tag id, for the event with the given event id. The data must be freed using ADB_ReleaseEventSIDDescriptorData.

Parameters

<i>serv_ptr</i>	service for the event
<i>event_id</i>	id of the event on the service
<i>dtag_id</i>	descriptor tag id
<i>ext_dtag_id</i>	extended descriptor tag id, or negative if not to be used
<i>desc_len</i>	number of bytes in the returned descriptor data

Returns

pointer to the descriptor data, or NULL if event or descriptor isn't found

4.10.2.74 U16BIT ADB_GetEventStartDate (void * event_ptr)

Returns the MJD date code for the start of the given event.

Parameters

<i>event_ptr</i>	event
------------------	-------

Returns

date code

4.10.2.75 U32DHMS ADB_GetEventStartDateTime (void * event_ptr)

Returns a value representing the date and time of the start of the given event.

Parameters

<i>event_ptr</i>	event
------------------	-------

Returns

date/time

4.10.2.76 U8BIT ADB_GetEventStartHour (void * *event_ptr*)

Returns the hour (0-23) value for the start of the given event.

Parameters

<i>event_ptr</i>	event
------------------	-------

Returns

hour value (0-23)

4.10.2.77 U8BIT ADB_GetEventStartMin (void * *event_ptr*)

Returns the minute (0-59) value for the start of the given event.

Parameters

<i>event_ptr</i>	event
------------------	-------

Returns

minute value (0-59)

4.10.2.78 U8BIT ADB_GetEventStartSecs (void * *event_ptr*)

Returns the seconds (0-59) value for the start of the given event.

Parameters

<i>event_ptr</i>	event
------------------	-------

Returns

seconds value (0-59)

4.10.2.79 BOOLEAN ADB_GetEventSubtitlesAvailFlag (void * *event_ptr*)

Returns whether DVB subtitles are signalled as being available for the given event.

Parameters

<i>event_ptr</i>	event
------------------	-------

Returns

TRUE if subtitles are available, FALSE otherwise

4.10.2.80 U8BIT ADB_GetFavouriteListByUserData (U32BIT *user_data*)

Returns the list id of the favourite list with the given user data value.

Parameters

<i>user_data</i>	user data value
------------------	-----------------

Returns

list id, 0 if list isn't found

4.10.2.81 U8BIT ADB_GetFavouriteListIdByIndex (U16BIT *index*)

Returns the list id of the favourite list defined by the given index.

Parameters

<i>index</i>	list index
--------------	------------

Returns

list id, 0 if list isn't found

4.10.2.82 U8BIT* ADB_GetFavouriteListName (U8BIT *list_id*)

Returns the name of the given favourite list, as a Unicode string.

Parameters

<i>list_id</i>	list id
----------------	---------

Returns

name of the favourite list as a unicode string, which must be released using STB_-ReleaseUnicodeString

4.10.2.83 U32BIT ADB_GetFavouriteListUserData (U8BIT *list_id*)

Returns the user data value of the given favourite list.

Parameters

<i>list_id</i>	list id
----------------	---------

Returns

user data value, will be 0 if list not found

4.10.2.84 U16BIT ADB_GetFreesatServiceId (void * *s_ptr*)

Returns Freesat service ID of the given service.

Parameters

<i>s_ptr</i>	service handle
--------------	----------------

Returns

Freesat ID of the service

4.10.2.85 U8BIT* ADB_GetFullCrid (void * *serv_ptr*, U8BIT * *event_str*)

Returns the full CRID for the given CRID string The returned string should be freed using STB_AppFreeMemory.

Parameters

<i>serv_ptr</i>	service for the event
<i>event_str</i>	CRID string

Returns

pointer to a string containing the CRID name

4.10.2.86 void* ADB_GetFullSITransport (void * *n_ptr*)

Returns a transport that is signalled as containing a complete set of SI data via an NIT linkage descriptor.

Parameters

<i>n_ptr</i>	network pointer
--------------	-----------------

Returns

pointer to transport

4.10.2.87 void* **ADB_GetLaterEvent** (void * *serv_ptr*, U16BIT *date*, U8BIT *hour*, U8BIT *min*)

Returns a copy of the event following the given date/time on the given service.

Parameters

<i>serv_ptr</i>	service for the event
<i>date</i>	MJD date code
<i>hour</i>	hour, 0-23
<i>min</i>	minute, 0-59

Returns

pointer to a copy of the event, or NULL if an event isn't found

4.10.2.88 BOOLEAN **ADB_GetLNBSettings** (void * *lnb_ptr*, ADB_LNB_SETTINGS * *settings*)

Returns the current settings for the given LNB.

Parameters

<i>lnb_ptr</i>	LNB
<i>settings</i>	structure in which the values are returned

Returns

TRUE if the LNB is valid and values are returned, FALSE otherwise

4.10.2.89 void **ADB_GetLockedServiceList** (void *** *slist_ptr*, U16BIT * *num_entries_ptr*)

Allocates and returns a list of all services that are marked as locked for parental control purposes.

Parameters

<i>slist_ptr</i>	pointer to an array that will be allocated, value will be NULL on return if there are no services
<i>num_entries_ptr</i>	pointer to the number of items in the returned array, value will be 0 on return if there are no services

4.10.2.90 U16BIT ADB_GetNetworkId (void * *n_ptr*)

Returns the network id of the given network.

Parameters

<i>n_ptr</i>	network
--------------	---------

Returns

network id, 0 if network isn't found

4.10.2.91 void ADB_GetNetworkList (void * *nlist_ptr*, U16BIT * *num_entries_ptr*)**

Allocates and returns a list of the network records in the database.

Parameters

<i>nlist_ptr</i>	pointer to an array that will be allocated, value will be NULL on return if there are no networks
<i>num_entries_ptr</i>	pointer to the number of items in the returned array, value will be 0 on return if there are no networks

4.10.2.92 U32BIT* ADB_GetNetworkListIds (void ** *nlist*, U16BIT *num_entries*)

Allocates and returns an array of network ids for the given networks The returned array should be freed using STB_AppFreeMemory.

Parameters

<i>nlist</i>	the array of network records
<i>num_entries</i>	number of items in the list

Returns

pointer to the network id array, NULL if no networks are provided or memory allocation fails

4.10.2.93 U8BIT ADB_GetNetworkListNames (void ** *nlist*, U16BIT *num_entries*, BOOLEAN *short_names*)**

Returns a list of names, in UTF-8 format, corresponding to the given network list. The returned list should be freed using ADB_ReleaseNameList.

Parameters

<i>nlist</i>	the array of network records
<i>num_entries</i>	number of items in the list
<i>short_names</i>	TRUE if the short network names should be returned

Returns

pointer to the list of name strings

4.10.2.94 U8BIT* ADB_GetNetworkName (void * *n_ptr*)

Returns the name, in UTF-8 format, of the given network. The returned name should be freed using STB_ReleaseUnicodeString.

Parameters

<i>n_ptr</i>	network
--------------	---------

Returns

UTF-8 format string, NULL on failure

4.10.2.95 U8BIT* ADB_GetNetworkNameByLang (void * *n_ptr*, U8BIT *lang*)

Returns the name in the language defined by lang, in UTF-8 format, of the given network. The returned name should be freed using STB_ReleaseUnicodeString.

Parameters

<i>n_ptr</i>	network
<i>lang</i>	language id

Returns

UTF-8 format string, NULL on failure

4.10.2.96 U16BIT ADB_GetNetworkPrimaryTargetRegions (U32BIT *country_code*, U8BIT ** *code_array*, U8BIT *** *name_array*)

Returns an array of primary region codes and names for the given country.

Parameters

<i>country_code</i>	code of country
<i>code_array</i>	pointer to an array of primary region codes that will be allocated in UI temp memory.
<i>name_array</i>	pointer to an array of strings that represent the names of the primary regions.

Returns

Number of codes & names returned in the arrays

4.10.2.97 U16BIT ADB_GetNetworkSecondaryTargetRegions (U32BIT *country_code*, U8BIT *primary_region*, U8BIT ** *code_array*, U8BIT *** *name_array*)

Returns an array of secondary region codes and names for the given country and primary region.

Parameters

<i>country_code</i>	code of country
<i>primary_region</i>	primary region code
<i>code_array</i>	pointer to an array of secondary region codes that will be allocated in UI temp memory.
<i>name_array</i>	pointer to an array of strings that represent the names of the secondary regions.

Returns

Number of codes & names returned in the arrays

4.10.2.98 void ADB_GetNetworkServiceList (void * *n_ptr*, void *** *slist_ptr*, U16BIT * *num_entries_ptr*)

Allocates and returns a list of all services in the database on all transports on the given network.

Parameters

<i>n_ptr</i>	network to be queried
<i>slist_ptr</i>	pointer to an array that will be allocated, value will be NULL on return if there are no services
<i>num_entries_ptr</i>	pointer to the number of items in the returned array, value will be 0 on return if there are no services

4.10.2.99 U8BIT ADB_GetNetworkTargetCountries (U32BIT ** *code_array*)

Returns a combined array of country codes for all discovered networks.

Parameters

<i>code_array</i>	pointer to an array that will be allocated in UI temp memory
-------------------	--

Returns

Number of codes returned in the array

4.10.2.100 `U16BIT ADB_GetNetworkTertiaryTargetRegions (U32BIT country_code,
U8BIT primary_region, U8BIT secondary_region, U16BIT ** code_array, U8BIT ***
name_array)`

Returns an array of tertiary region codes and names for the given country, primary region and secondary region.

Parameters

<i>country_code</i>	code of country
<i>primary_region</i>	primary region code
<i>secondary_region</i>	secondary region code
<i>code_array</i>	pointer to an array of tertiary region codes that will be allocated in UI temp memory.
<i>name_array</i>	pointer to an array of strings that represent the names of the tertiary regions.

Returns

Number of codes & names returned in the arrays

4.10.2.101 `void ADB_GetNetworkTransportList (void * n_ptr, void *** tlist_ptr, U16BIT
* num_entries_ptr)`

Allocates and returns a list of all transport records in the database for the given network.

Parameters

<i>n_ptr</i>	network pointer
<i>tlist_ptr</i>	pointer to an array that will be allocated, value will be NULL on return if there are no transports
<i>num_entries_ptr</i>	pointer to the number of items in the returned array, value will be 0 on return if there are no transports

4.10.2.102 `void* ADB_GetNextRCTLink (void * link_ptr)`

Returns the RCT link following the given link.

Parameters

<i>link_ptr</i>	pointer to an RCT link
-----------------	------------------------

Returns

next RCT link, or NULL

4.10.2.103 void* ADB_GetNextSatellite (void * *sat_ptr*)

Returns the next satellite from the database.

Parameters

<i>sat_ptr</i>	current satellite pointer, NULL to get the first satellite
----------------	--

Returns

pointer to satellite

4.10.2.104 void* ADB_GetNextServiceInList (U32BIT *list_type*, void * *serv*)

Returns the next service, in LCN order, for the given list type, starting from the given service. If *serv* is NULL then the first service is returned.

Parameters

<i>list_type</i>	defines the types of services to be included, multiple types can be combined by ORing the values
<i>serv</i>	starting service, NULL to get the first service

Returns

next service, or NULL if no more services

4.10.2.105 void ADB_GetNowNextEvents (void * *serv_ptr*, void ** *now_event*, void ** *next_event*)

Makes copies of the now and/or next events (EITp/f) for the given service. The returned events should be freed using ADB_ReleaseEventData.

Parameters

<i>serv_ptr</i>	get the now/next events for this service
<i>now_event</i>	address in which to return a pointer to the now event This value can be NULL if the now event isn't to be returned
<i>next_event</i>	address in which to return a pointer to the next event This value can be NULL if the now event isn't to be returned

4.10.2.106 U16BIT ADB_GetNumFavouriteLists (void)

Returns the number of favourite lists.

Returns

number of favourite lists

4.10.2.107 U16BIT ADB_GetNumLNBS (void)

Returns the number of LNBS in the database.

Returns

Number of LNBS

4.10.2.108 U16BIT ADB_GetNumNetworks (void)

Returns the number of network records in the database.

Returns

Number of networks

4.10.2.109 U16BIT ADB_GetNumSatellites (void)

Returns the number of satellites in the database.

Returns

Number of satellites

4.10.2.110 U16BIT ADB_GetNumServices (void)

Returns the total number of services in the database.

Returns

number of services

4.10.2.111 U16BIT ADB_GetNumServicesInList (U32BIT *list_type*, BOOLEAN *inc_hidden*)

Returns the number of services in the database that would be returned with the given list type.

Parameters

<i>list_type</i>	defines the types of services to be included, multiple types can be combined by ORing the values
<i>inc_hidden</i>	TRUE if hidden services should be included in the returned count

Returns

number of services

4.10.2.112 U16BIT ADB_GetNumStreams (void * *serv_ptr*, ADB_STREAM_LIST_TYPE *stream_list_type*)

Returns the number of streams of the given type for the given service.

Parameters

<i>serv_ptr</i>	service
<i>stream_list_type</i>	type of stream

Returns

Number of streams

4.10.2.113 U16BIT ADB_GetNumTransports (void)

Returns the number of transports in the database.

Returns

Number of transports

4.10.2.114 void* ADB_GetPrevServiceInList (U32BIT *list_type*, void * *serv*)

Returns the previous service, in LCN order, for the given list type, starting from the given service. If *serv* is NULL then the last service is returned.

Parameters

<i>list_type</i>	defines the types of services to be included, multiple types can be combined by ORing the values
<i>serv</i>	starting service, NULL to get the last service

Returns

previous service, or NULL if no more services

4.10.2.115 U16BIT ADB_GetProfileList (void * *profile_list*, U16BIT * *active_profile*)**

Returns an array of available profiles. The array will be allocated within the function and should be released using ADB_ReleaseProfileList. This is a CI+ feature.

Parameters

<i>profile_list</i>	pointer to the returned array of profiles. Value isn't valid if 0 is returned
<i>active_profile</i>	pointer to return the index of the currently active profile

Returns

Number of profiles

4.10.2.116 ADB_PROFILE_TYPE ADB_GetProfileType (void * *profile*)

Returns the type of the given profile.

Parameters

<i>profile</i>	profile handle
----------------	----------------

Returns

profile type

4.10.2.117 U8BIT* ADB_GetRCTLinkName (void * *link_ptr*)

Returns the name of the given RCT link.

Parameters

<i>link_ptr</i>	pointer to an RCT link
-----------------	------------------------

Returns

pointer to the link's name - NOTE: this shouldn't be changed or freed!

4.10.2.118 U8BIT* ADB_GetRCTLinkPromoText (void * *link_ptr*)

Returns the RCT link's promotional text based on the default language.

Parameters

<i>link</i>	pointer to link
-------------	-----------------

Returns

Pointer to promotional link string

4.10.2.119 U8BIT* ADB_GetRCTLinkUriString (void * *link_ptr*)

Returns the uri string of the given RCT link.

Parameters

<i>link_ptr</i>	pointer to an RCT link
-----------------	------------------------

Returns

pointer to the link's uri - NOTE: this shouldn't be changed or freed!

4.10.2.120 void **ADB_GetReqdAudioStreamSettings** (void * *s_ptr*, BOOLEAN * *valid*, U32BIT * *lang_code*, ADB_AUDIO_TYPE * *audio_type*, ADB_STREAM_TYPE * *stream_type*)

Returns the settings defined by ADB_SetReqdAudioStreamSettings that will be used for audio on the given service.

Parameters

<i>s_ptr</i>	service
<i>valid</i>	return for the valid flag state
<i>lang_code</i>	returns the set audio language code
<i>audio_type</i>	returns the set audio type
<i>stream_type</i>	returns the set stream type

4.10.2.121 void **ADB_GetReqdSubtitleStreamSettings** (void * *s_ptr*, BOOLEAN * *valid*, U32BIT * *lang_code*, ADB_SUBTITLE_TYPE * *subt_type*)

Returns the settings defined by ADB_SetReqdSubtitleStreamSettings that will be used for subtitles on the given service.

Parameters

<i>s_ptr</i>	service
<i>valid</i>	return for the valid flag state
<i>lang_code</i>	returns the set language code
<i>subt_type</i>	returns the set subtitle type

4.10.2.122 void **ADB_GetReqdTeletextStreamSettings** (void * *s_ptr*, BOOLEAN * *valid*, U32BIT * *lang_code*, ADB_TELETEXT_TYPE * *ttext_type*)

Returns the settings defined by ADB_SetReqdTeletextStreamSettings that will be used for teletext on the given service.

Parameters

<i>s_ptr</i>	service
<i>valid</i>	return for the valid flag state
<i>lang_code</i>	returns the set language code
<i>ttext_type</i>	returns the set teletext type

4.10.2.123 U16BIT ADB_GetRequiredADPid (void * *s_ptr*, BOOLEAN * *broadcast_mix*)

Returns the PID to be used for audio description for the given service based on language settings, output format, etc.

Parameters

<i>s_ptr</i>	service
<i>broadcast_mix</i>	returned as TRUE if the selected AD is broadcast mix

Returns

PID value, or 0 if not known or invalid service

4.10.2.124 E_STB_DP_AUDIO_MODE ADB_GetRequiredAudioMode (void * *s_ptr*)

Returns the mode to be used for audio for the given service based on language settings, output format, etc.

Parameters

<i>s_ptr</i>	service
--------------	---------

Returns

mode, default is AUDIO_STEREO

4.10.2.125 U16BIT ADB_GetRequiredAudioPid (void * *s_ptr*)

Returns the audio PID to be used for the given service based on language settings, output format, etc.

Parameters

<i>s_ptr</i>	service
--------------	---------

Returns

PID value, or 0 if not known or invalid service

4.10.2.126 U16BIT ADB_GetRequiredSubtitleParams (void * *s_ptr*, U16BIT * *cpage_ptr*, U16BIT * *apage_ptr*)

Returns the PID to be used for DVB subtitles for the given service based on language settings.

Parameters

<i>s_ptr</i>	service
<i>cpage_ptr</i>	returns the composition page number
<i>apage_ptr</i>	returns the ancilliary page number

Returns

PID value, 0 if not found

4.10.2.127 U16BIT ADB_GetRequiredTtextPid (void * *s_ptr*, BOOLEAN *for_subtitles*, U8BIT * *magazine*, U8BIT * *page*)

Returns the PID to be used for teletext for the given service based on language settings.

Parameters

<i>s_ptr</i>	service
<i>for_subtitles</i>	find the teletext service marked as being subtitles
<i>magazine</i>	returns the magazine part of the teletext page number
<i>page</i>	returns the page number part of the teletext page number

Returns

PID value, or 0 if not known or invalid service

4.10.2.128 BOOLEAN ADB_GetSatelliteDirection (void * *sat_ptr*)

Returns the position direction of the given satellite.

Parameters

<i>sat_ptr</i>	satellite
----------------	-----------

Returns

TRUE=east, FALSE=west

4.10.2.129 void* ADB_GetSatelliteLNB (void * *sat_ptr*)

Returns the LNB associated with the given satellite.

Parameters

<i>sat_ptr</i>	satellite
----------------	-----------

Returns

pointer to LNB

4.10.2.130 U16BIT ADB_GetSatelliteLongitude (void * *sat_ptr*)

Returns the longitudinal position of the given satellite in 1/10ths degree.

Parameters

<i>sat_ptr</i>	satellite
----------------	-----------

Returns

satellite position in 1/10ths degree

4.10.2.131 U8BIT* ADB_GetSatelliteName (void * *sat_ptr*)

Returns the pointer to the name of the satellite.

Parameters

<i>sat_ptr</i>	satellite
----------------	-----------

Returns

pointer to satellite name

4.10.2.132 U16BIT ADB_GetServiceAudioPid (void * *s_ptr*)

Returns the audio PID for the given service. This will be the one chosen based on the settings for language, etc, or the one explicitly set (e.g. due to user selection).

Parameters

<i>s_ptr</i>	service
--------------	---------

Returns

PID value, or 0 if not known or invalid service

4.10.2.133 ADB_STREAM_TYPE ADB_GetServiceAudioType (void * *s_ptr*)

Returns the audio type being used by the given service.

Parameters

<i>s_ptr</i>	service
--------------	---------

Returns

Audio type, defaults to ADB_AUDIO_STREAM (MPEG1)

4.10.2.134 U8BIT* ADB_GetServiceDefaultAuthority (void * *serv_ptr*)

Returns a copy of the default authority CRID string for the given service. This string will be returned as broadcast and should be freed using STB_AppFreMemory.

Parameters

<i>c_ptr</i>	pointer to CRID record
--------------	------------------------

Returns

pointer to a copy of the CRID string, or NULL

4.10.2.135 BOOLEAN ADB_GetServiceDoNotScramble (void * *s_ptr*)

Returns the do_not_scramble flag for the service which is based on the presence of an FTA descriptor. The value returned will be found by looking backwards from the service to the transport, etc.

Parameters

<i>s_ptr</i>	pointer to the service
--------------	------------------------

Returns

FALSE if the service is to be protected, TRUE otherwise

4.10.2.136 U8BIT ADB_GetServiceFavGroups (void * *s_ptr*)

Returns the favourite group value for the given service. This is an 8-bit value that can define whether the service is in any of upto 8 favourite groups. The value is a bit flag, so 1 means it's in group 1, 2 means it's in group 2, 4 means it's in group 3, 7 means it's in groups 1, 2 and 3, etc.

Parameters

<i>s_ptr</i>	service
--------------	---------

Returns

favourite group bit value

4.10.2.137 U8BIT* ADB_GetServiceFullName (void * *s_ptr*, BOOLEAN *use_pref_name*)

Returns the full name of the given service as a UTF-8 string. The returned string should be freed using STB_ReleaseUnicodeString.

Parameters

<i>s_ptr</i>	service
<i>use_pref_name</i>	TRUE if the preferred name should be returned

Returns

pointer to UTF-8 string, or NULL

4.10.2.138 U8BIT* ADB_GetServiceFullNameByLangAndPrefId (void * *s_ptr*, U8BIT *lang*, U8BIT *pref_name_id*)

Returns the full name of the given service as a UTF-8 string using the given language and preferred name id, rather than the defaults. The returned string should be freed using STB_ReleaseUnicodeString.

Parameters

<i>s_ptr</i>	service
<i>lang</i>	language id
<i>pref_name_id</i>	preferred name id

Returns

pointer to UTF-8 string, or NULL

4.10.2.139 BOOLEAN ADB_GetServiceHasCaDesc (void * *s_ptr*)

Used to query whether the given service has a CA descriptor.

Parameters

<i>s_ptr</i>	service
--------------	---------

Returns

TRUE if the service has a CA descriptor, FALSE otherwise

4.10.2.140 BOOLEAN ADB_GetServiceHiddenFlag (void * *s_ptr*)

Returns the hidden status of the given service. A hidden service should not be present to the user in any list of available services.

Parameters

<i>s_ptr</i>	service
--------------	---------

Returns

TRUE if the service is hidden, FALSE if it's visible

4.10.2.141 U16BIT ADB_GetServiceId (void * *s_ptr*)

Returns the signalled service id of the given service.

Parameters

<i>s_ptr</i>	service
--------------	---------

Returns

service id

4.10.2.142 void ADB_GetServiceIds (void * *s_ptr*, U16BIT * *onet_id*, U16BIT * *trans_id*, U16BIT * *serv_id*)

Returns the original network id, transport id and service id for the given service.

Parameters

<i>s_ptr</i>	service handle
<i>onet_id</i>	pointer for the return of the original network id
<i>trans_id</i>	pointer for the return of the transport id
<i>serv_id</i>	pointer for the return of the service id

4.10.2.143 U16BIT ADB_GetServiceLcn (void * *s_ptr*)

Returns the logical channel number assigned to the given service.

Parameters

<i>s_ptr</i>	service
--------------	---------

Returns

logical channel number

4.10.2.144 BOOLEAN ADB_GetServiceLcnEditable (void * *s_ptr*)

Returns whether the LCN assigned to the given service can be changed.

Parameters

<i>s_ptr</i>	service
--------------	---------

Returns

TRUE if the service is valid and its LCN can be edited, FALSE otherwise

4.10.2.145 void **ADB_GetServiceList** (U32BIT *list_type*, void *** *slist_ptr*, U16BIT * *num_entries_ptr*)

Allocates and returns a list of all services in the database for the given list type.

Parameters

<i>list_type</i>	defines the types of services to be included, multiple types can be combined by ORing the values
<i>slist_ptr</i>	pointer to an array that will be allocated, value will be NULL on return if there are no services
<i>num_entries_ptr</i>	pointer to the number of items in the returned array, value will be 0 on return if there are no services

4.10.2.146 U8BIT** **ADB_GetServiceListFullNames** (void ** *slist*, U16BIT *num_entries*, BOOLEAN *use_pref_names*)

Allocates and returns a list of the fullnames for all the services in the given service list. The returned list should be freed using ADB_ReleaseNameList.

Parameters

<i>slist</i>	list of services
<i>num_entries</i>	number of services in the list
<i>use_pref_name</i>	TRUE if the preferred names are to be returned.

Returns

allocated array of service names as UTF-8 strings

4.10.2.147 U32BIT* **ADB_GetServiceListHdFlag** (void ** *slist*, U16BIT *num_entries*)

Allocates and returns an array indicating whether each service in the given service list is an HD service. A value of TRUE means the service is signalled as being HD and FALSE means it's SD. The returned list should be freed using STB_AppFreeMemory.

Parameters

<i>slist</i>	list of services
<i>num_entries</i>	number of services in the list

Returns

allocated array of flags, NULL if list is empty or memory allocation fails

4.10.2.148 U32BIT* **ADB_GetServiceListHiddenFlag** (void ** *slist*, U16BIT *num_entries*)

Allocates and returns an array indicating whether each service in the given service list is signalled as being hidden or not. A value of TRUE means the service is hidden and FALSE means it's visible. The returned list should be freed using STB_AppFreeMemory.

Parameters

<i>slist</i>	list of services
<i>num_entries</i>	number of services in the list

Returns

allocated array of flags, NULL if list is empty or memory allocation fails

4.10.2.149 void **ADB_GetServiceListIncludingHidden** (U32BIT *list_type*, void *** *slist_ptr*, U16BIT * *num_entries_ptr*, BOOLEAN *ignore_selectable*)

Allocates and returns a list of all services in the database for the given list type, including hidden services.

Parameters

<i>list_type</i>	defines the types of services to be included, multiple types can be combined by ORing the values
<i>slist_ptr</i>	pointer to an array that will be allocated, value will be NULL on return if there are no services
<i>num_entries_ptr</i>	pointer to the number of items in the returned array, value will be 0 on return if there are no services
<i>ignore_selectable</i>	defines whether the returned service list should include services that are signalled as being unselectable. Use TRUE to ignore a service's selectable flag

4.10.2.150 U16BIT* **ADB_GetServiceListLcns** (void ** *slist*, U16BIT *num_entries*)

Allocates and returns an array of the logical channel numbers that have been assigned to all the services in the given service list. The returned list should be freed using STB_AppFreeMemory.

Parameters

<i>slist</i>	list of services
<i>num_entries</i>	number of services in the list

Returns

allocated array of LCNs, NULL if list is empty or memory allocation fails

4.10.2.151 **U32BIT* ADB_GetServiceListLockedFlag** (void ** *slist*, U16BIT *num_entries*)

Allocates and returns an array of the 'locked' flags for each of the services in the given service list. A service is marked as 'locked' due to setup of parental control. A value of TRUE means the service is locked, FALSE otherwise. The returned list should be freed using STB_AppFreeMemory.

Parameters

<i>slist</i>	list of services
<i>num_entries</i>	number of services in the list

Returns

allocated array of flags, NULL if list is empty or memory allocation fails

4.10.2.152 **void ADB_GetServiceListLockedSubset** (void ** *slist*, U16BIT *num_entries*, void *** *locked_slist_ptr*, U16BIT * *locked_num_entries_ptr*)

Creates a subset of locked services from the supplied service list. Memory is allocated as TempMemory so will be released when the calling screen closes.

Parameters

<i>slist</i>	pointer to an array of service record pointers
<i>num_entries</i>	number of entries in the list
<i>locked_slist_ptr</i>	address for the return of the locked slist
<i>locked_num_entries_ptr</i>	address for the return of the number of entries in the locked list

4.10.2.153 **U8BIT** ADB_GetServiceListNetworkNames** (void ** *slist*, U16BIT *num_entries*)

Allocates and returns a list of network names for all the services in the given service list. The returned list should be freed using ADB_ReleaseNameList.

Parameters

<i>slist</i>	list of services
<i>num_entries</i>	number of services in the list

Returns

allocated array of service names as UTF-8 strings

4.10.2.154 U32BIT* ADB_GetServiceListNewFlag (void ** *slist*, U16BIT *num_entries*)

Allocates and returns an array of the 'new' flags for each of the services in the given service list. A service is marked as 'new' when it's added to the service database following an update service scan. A value of TRUE means the service is new, FALSE otherwise. The returned list should be freed using STB_AppFreeMemory.

Parameters

<i>slist</i>	list of services
<i>num_entries</i>	number of services in the list

Returns

allocated array of flags, NULL if list is empty or memory allocation fails

4.10.2.155 U16BIT* ADB_GetServiceListOrigLcns (void ** *slist*, U16BIT *num_entries*)

Allocates and returns an array of the logical channel numbers that each of the services in the given service list originally requested. The returned list should be freed using STB_AppFreeMemory.

Parameters

<i>slist</i>	list of services
<i>num_entries</i>	number of services in the list

Returns

allocated array of LCNs, NULL if list is empty or memory allocation fails

4.10.2.156 U32BIT* ADB_GetServiceListScrambledFlag (void ** *slist*, U16BIT *num_entries*)

Allocates and returns an array of the scrambled flags for each of the services in the given service list. A value of TRUE means the service is signalled as scrambled and FALSE means it's free-to-air. The returned list should be freed using STB_AppFreeMemory.

Parameters

<i>slist</i>	list of services
<i>num_entries</i>	number of services in the list

Returns

allocated array of flags, NULL if list is empty or memory allocation fails

4.10.2.157 `U8BIT** ADB_GetServiceListShortNames (void ** slist, U16BIT num_entries, BOOLEAN use_pref_names)`

Allocates and returns a list of the short names for all the services in the given service list. The returned list should be freed using ADB_ReleaseNameList.

Parameters

<i>slist</i>	list of services
<i>num_entries</i>	number of services in the list
<i>use_pref_name</i>	TRUE if the preferred names are to be returned.

Returns

allocated array of service names as UTF-8 strings

4.10.2.158 `U8BIT** ADB_GetServiceListTransportNames (void ** slist, U16BIT num_entries)`

Allocates and returns a list of transport names for all the services in the given service list. The returned list should be freed using ADB_ReleaseNameList.

Parameters

<i>slist</i>	list of services
<i>num_entries</i>	number of services in the list

Returns

allocated array of service names as UTF-8 strings

4.10.2.159 `U32BIT* ADB_GetServiceListTransportStrengths (void ** slist, U16BIT num_entries)`

Allocates and returns an array of the signal strengths of the transports containing the given list of services. The returned list should be freed using STB_AppFreeMemory.

Parameters

<i>slist</i>	list of services
<i>num_entries</i>	number of services in the list

Returns

allocated array of flags, NULL if list is empty or memory allocation fails

4.10.2.160 U32BIT* **ADB_GetServiceListUnavailableFlag** (void ** *slist*, U16BIT *num_entries*)

Allocates and returns an array of the unavailable flags for each of the services in the given service list. A value of TRUE means the service is unavailable, FALSE otherwise. The returned list should be freed using STB_AppFreeMemory.

Parameters

<i>slist</i>	list of services
<i>num_entries</i>	number of services in the list

Returns

allocated array of flags, NULL if list is empty or memory allocation fails

4.10.2.161 BOOLEAN **ADB_GetServiceLockedFlag** (void * *s_ptr*)

Returns the status of the 'locked' flag of the given service, which is used by the parental control.

Parameters

<i>s_ptr</i>	service
--------------	---------

Returns

TRUE if the service is locked, FALSE otherwise

4.10.2.162 BOOLEAN **ADB_GetServiceNewFlag** (void * *s_ptr*)

Returns the status of the 'new' flag of the given service. This flag indicates whether a service has been added to the service lineup following a service scan.

Parameters

<i>s_ptr</i>	service
--------------	---------

Returns

TRUE if the service is new, FALSE otherwise

4.10.2.163 BOOLEAN **ADB_GetServiceNotRunningFlag** (void * *s_ptr*)

Returns the status of the 'not running' flag of the given service. This flag indicates whether a service is signalled as not running in the SDT, or is removed from the PAT.

Parameters

<i>s_ptr</i>	service
--------------	---------

Returns

TRUE if the service is not running, FALSE otherwise

4.10.2.164 BOOLEAN ADB_GetServiceNowNextState (void * *s_ptr*)

Returns whether the EIT now/next data is being held in memory for the service.

Parameters

<i>s_ptr</i>	service
--------------	---------

Returns

TRUE if data is being held, FALSE otherwise

4.10.2.165 U8BIT ADB_GetServiceNumRCTLinks (void * *serv_ptr*)

Returns the number of RCT links for the given service.

Parameters

<i>serv_ptr</i>	service from which links are to be cleared
-----------------	--

Returns

Number of RCT links

4.10.2.166 U16BIT ADB_GetServiceOrigLcn (void * *s_ptr*)

Returns the logical channel number originally requested by the given service.

Parameters

<i>s_ptr</i>	service
--------------	---------

Returns

logical channel number

4.10.2.167 U16BIT ADB_GetServicePCRPid (void * *s_ptr*)

Returns the PCR PID for the given service.

Parameters

<i>s_ptr</i>	service
--------------	---------

Returns

PID value, or 0 if not known or invalid service

4.10.2.168 U8BIT* ADB_GetServicePMTData (void * *s_ptr*, U16BIT * *data_len*)

Returns the current PMT data for the given service.

Parameters

<i>s_ptr</i>	service pointer
<i>data_len</i>	pointer to return the size of the PMT data, only valid return value isn't NULL

Returns

PMT data or NULL if not available

4.10.2.169 U16BIT ADB_GetServicePmtPid (void * *s_ptr*)

Returns the current PMT PID for the given service.

Parameters

<i>s_ptr</i>	service pointer
--------------	-----------------

Returns

PMT PID

4.10.2.170 U8BIT* ADB_GetServiceProviderName (void * *s_ptr*)

Returns the provider name, as a UTF-8 string, using the default language, of the given service. The returned string should be freed using STB_ReleaseUnicodeString.

Parameters

<i>s_ptr</i>	service
--------------	---------

Returns

UTF-8 provider name, or NULL

4.10.2.171 U8BIT* ADB_GetServiceProviderNameByLang (void * *s_ptr*, U8BIT *lang*)

Returns the provider name, as a UTF-8 string, using the given language, of the given service. The returned string should be freed using STB_ReleaseUnicodeString.

Parameters

<i>s_ptr</i>	service
<i>lang</i>	language identifier, one of the ACFG_DB_LANG_ values defined in ap-_cfg.h

Returns

UTF-8 provider name, or NULL

4.10.2.172 void* ADB_GetServiceRCTLinks (void * *serv_ptr*, U8BIT * *num_links*)

Returns a copy of the RCT links for the given service. The returned list should be freed using ADB_ReleaseRCTLinks.

Parameters

<i>serv_ptr</i>	service from which links are to be copied
<i>num_links</i>	pointer to return the number of links

Returns

Copy of links, or NULL

4.10.2.173 U8BIT ADB_GetServiceRunningStatus (void * *s_ptr*)

Returns the running status of the given service as reported by the SDT.

Parameters

<i>s_ptr</i>	service
--------------	---------

Returns

running status will be 0 if service isn't valid or SDT hasn't been parsed yet

4.10.2.174 BOOLEAN ADB_GetServiceScheduleState (void * *s_ptr*)

Returns whether the EIT schedule events are being held in memory for the service.

Parameters

<i>s_ptr</i>	service
--------------	---------

Returns

TRUE if data is being held, FALSE otherwise

4.10.2.175 **BOOLEAN ADB_GetServiceScrambledFlag (void * *s_ptr*)**

Returns the status of the 'scrambled' flag of the given service. This flag is set depending on the scrambled state of all the streams that makeup the service being marked as free-to-air.

Parameters

<i>s_ptr</i>	service
--------------	---------

Returns

TRUE if the service is scrambled, FALSE otherwise

4.10.2.176 **U8BIT* ADB_GetServiceShortName (void * *s_ptr*, BOOLEAN *use_pref_name*)**

Returns the short name of the given service as a UTF-8 string. The returned string should be freed using STB_ReleaseUnicodeString.

Parameters

<i>s_ptr</i>	service
<i>use_pref_name</i>	TRUE if the preferred name should be returned

Returns

pointer to UTF-8 string, or NULL

4.10.2.177 **U8BIT* ADB_GetServiceShortNameByLangAndPrefId (void * *s_ptr*, U8BIT *lang*, U8BIT *pref_name_id*)**

Returns the short name of the given service as a UTF-8 string using the given language and preferred name id, rather than the defaults. The returned string should be freed using STB_ReleaseUnicodeString.

Parameters

<i>s_ptr</i>	service
<i>lang</i>	language id
<i>pref_name_id</i>	preferred name id

Returns

pointer to UTF-8 string, or NULL

4.10.2.178 U16BIT ADB_GetServiceTextPid (void * *s_ptr*)

Returns the PID being used for DVB subtitles and/or teletext for the given service. This will be the one chosen based on the settings for language, etc, or the one explicitly set (e.g. due to user selection).

Parameters

<i>s_ptr</i>	service
--------------	---------

Returns

PID value, or 0 if not known or invalid service

4.10.2.179 void* ADB_GetServiceTransportPtr (void * *s_ptr*)

Returns a pointer to the service's parent transport record.

Parameters

<i>s_ptr</i>	pointer to the service
--------------	------------------------

Returns

transport, or NULL

4.10.2.180 ADB_SERVICE_TYPE ADB_GetServiceType (void * *s_ptr*)

Returns the signalled type of the given service.

Parameters

<i>s_ptr</i>	service
--------------	---------

Returns

service type, returns 0 if unknown

4.10.2.181 BOOLEAN ADB_GetServiceUnavailableFlag (void * *s_ptr*)

Returns the status of the unavailable flag of the given service. This flag indicates whether a service is no longer signalled in the SDT.

Parameters

<i>s_ptr</i>	service
--------------	---------

Returns

TRUE if the service is unavailable, FALSE otherwise

4.10.2.182 U16BIT ADB_GetServiceVideoPid (void * *s_ptr*)

Returns the video PID for the given service.

Parameters

<i>s_ptr</i>	service
--------------	---------

Returns

PID value, or 0 if not known or invalid service

4.10.2.183 ADB_STREAM_TYPE ADB_GetServiceVideoType (void * *s_ptr*)

Returns the video type being used by the given service.

Parameters

<i>s_ptr</i>	service
--------------	---------

Returns

Video type, defaults to ADB_VIDEO_STREAM (MPEG2)

4.10.2.184 U8BIT ADB_GetStreamComponentType (void * *stream_ptr*)

Returns the component type as defined in the EITp/f's component descriptor for the given stream.

Parameters

<i>stream_ptr</i>	stream
-------------------	--------

Returns

component type

4.10.2.185 BOOLEAN ADB_GetStreamHasCaDesc (void * *stream_ptr*)

Returns whether the given stream has a CA descriptor.

Parameters

<i>stream_ptr</i>	stream
-------------------	--------

Returns

TRUE if stream has a CA descriptor, FALSE otherwise

4.10.2.186 BOOLEAN ADB_GetStreamInUse (void * *stream_ptr*)

Returns whether the given stream is marked as being 'in use', which means it will be the stream being decoded for video, audio, etc.

Parameters

<i>stream_ptr</i>	stream
-------------------	--------

Returns

TRUE if stream is in use, FALSE otherwise

4.10.2.187 void ADB_GetStreamList (void * *serv_ptr*, ADB_STREAM_LIST_TYPE *stream_list_type*, void *** *streamlist_ptr*, U16BIT * *num_entries_ptr*)

Allocates and returns an array of the streams of the given type and for the given service. The returned array should be freed using ADB_ReleaseStreamList.

Parameters

<i>serv_ptr</i>	service
<i>stream_list_type</i>	type of streams to be returned
<i>streamlist_ptr</i>	pointer to array that will be allocated
<i>num_entries_ptr</i>	pointer to number of items returned in the array

4.10.2.188 U8BIT ADB_GetStreamNumTags (void * *stream_ptr*)

Returns the number of tag values defined for the given stream.

Parameters

<i>stream_ptr</i>	stream
-------------------	--------

Returns

number of tag values

4.10.2.189 U16BIT ADB_GetStreamPID (void * *stream_ptr*)

Returns the PID for the given stream.

Parameters

<i>stream_ptr</i>	stream
-------------------	--------

Returns

PID

4.10.2.190 U8BIT ADB_GetStreamTag (void * *stream_ptr*, U8BIT *index*)

Returns the requested tag value of the given stream.

Parameters

<i>stream_ptr</i>	stream
<i>index</i>	tag value to return, must be < ADB_GetStreamNumTags

Returns

tag value

4.10.2.191 ADB_STREAM_TYPE ADB_GetStreamType (void * *stream_ptr*)

Returns the type of the given stream.

Parameters

<i>stream_ptr</i>	stream
-------------------	--------

Returns

stream type

4.10.2.192 U16BIT ADB_GetSubtitleStreamAncillaryPage (void * *stream_ptr*)

Returns the ancillary page value of the given subtitle stream.

Parameters

<i>stream_ptr</i>	stream
-------------------	--------

Returns

ancillary page value

4.10.2.193 U16BIT ADB_GetSubtitleStreamCompositionPage (void * *stream_ptr*)

Returns the composition page value of the given subtitle stream.

Parameters

<i>stream_ptr</i>	stream
-------------------	--------

Returns

composition page value

4.10.2.194 U32BIT ADB_GetSubtitleStreamLangCode (void * *stream_ptr*)

Returns the 3 char language code of the given subtitle stream.

Parameters

<i>stream_ptr</i>	stream
-------------------	--------

Returns

3 char language code encoded as a 32-bit number

4.10.2.195 ADB_SUBTITLE_TYPE ADB_GetSubtitleStreamType (void * *stream_ptr*)

Returns the type, as defined in the PMT, of the given subtitle stream.

Parameters

<i>stream_ptr</i>	stream
-------------------	--------

Returns

subtitle stream type

4.10.2.196 void ADB_GetTransportAnalogTuningParams (void * *t_ptr*, U32BIT * *freq_hz*, S8BIT * *offset*, E_STB_DP_ANALOG_VIDEO_TYPE * *vtype*)

Returns the settings to tune to the given terrestrial analog transport.

Parameters

<i>t_ptr</i>	transport to get the tuning parameters for
<i>freq_hz</i>	pointer to return the tuning frequency in Hz, 0 on error
<i>offset</i>	pointer to return offset value, 0 on error
<i>vtype</i>	pointer to return the video type

4.10.2.197 U8BIT ADB_GetTransportBwidth (void * *t_ptr*)

Returns the bandwidth value of the given DVB-T/T2 transport.

Parameters

<i>t_ptr</i>	transport
--------------	-----------

Returns

bandwidth, return value is undefined if transport isn't DVB-T/T2

4.10.2.198 E_STB_DP_CMODE ADB_GetTransportCableMode (void * *t_ptr*)

Returns the QAM mode of the given cable transport.

Parameters

<i>t_ptr</i>	cable transport
--------------	-----------------

Returns

QAM mode, returns MODE_QAM_AUTO if transport isn't DVB-C

4.10.2.199 U16BIT ADB_GetTransportCableSymbolRate (void * *t_ptr*)

Returns the symbol rate of the given cable transport.

Parameters

<i>t_ptr</i>	cable transport
--------------	-----------------

Returns

symbol rate, returns 0 if transport isn't DVB-C

4.10.2.200 void ADB_GetTransportCableTuningParams (void * *t_ptr*, U32BIT * *freq_hz*, E_STB_DP_CMODE * *mode*, U16BIT * *symrate*)

Returns the parameters needed to tune a cable tuner to a transport.

Parameters

<i>t_ptr</i>	transport to tune to
<i>freq_hz</i>	pointer to the returned freq of the transport
<i>mode</i>	pointer to the returned cable mode of the transport
<i>symrate</i>	pointer to the returned symbol rate of the transport

4.10.2.201 E_STB_TUNE_TCONST ADB_GetTransportConstellation (void * *t_ptr*)

Returns the constellation of the given DVB-T/T2 transport.

Parameters

<i>t_ptr</i>	transport
--------------	-----------

Returns

constellation, returns TUNE_TCONST_UNDEFINED if transport isn't DVB-T/T2

4.10.2.202 U32BIT ADB_GetTransportFreqHz (void * *t_ptr*)

Returns the frequency, in Hz for DVB-T/T2 and DVB-C, and MHz for DVB-S/S2, of the given transport.

Parameters

<i>t_ptr</i>	transport
--------------	-----------

Returns

frequency, 0 if transport not found

4.10.2.203 void* ADB_GetTransportFromIds (U16BIT *net_id*, U16BIT *onet_id*, U16BIT *tran_id*)

Finds the transport with the given ids.

Parameters

<i>net_id</i>	network id, can be ADB_INVALID_DVB_ID
<i>onet_id</i>	original network id
<i>tran_id</i>	transport id

Returns

transport, or NULL if not found

4.10.2.204 E_STB_TUNE_TGUARDINT ADB_GetTransportGuardInt (void * *t_ptr*)

Returns the guard interval of the given DVB-T/T2 transport.

Parameters

<i>t_ptr</i>	transport
--------------	-----------

Returns

guard interval, returns TUNE_TGUARDINT_UNDEFINED if transport isn't DVB-T/-T2

4.10.2.205 E_STB_TUNE_THIERARCHY ADB_GetTransportHierarchy (void * *t_ptr*)

Returns the hierarchy of the given DVB-T/T2 transport.

Parameters

<i>t_ptr</i>	transport
--------------	-----------

Returns

hierarchy, returns TUNE_THIERARCHY_UNDEFINED if transport isn't DVB-T/T2

4.10.2.206 E_STB_TUNE_TCODERATE ADB_GetTransportHpCodeRate (void * *t_ptr*)

Returns the HP coderate of the given DVB-T/T2 transport.

Parameters

<i>t_ptr</i>	transport
--------------	-----------

Returns

constellation, returns TUNE_TCODERATE_UNDEFINED if transport isn't DVB-T/-T2

4.10.2.207 void ADB_GetTransportList (void *** *tlist_ptr*, U16BIT * *num_entries_ptr*)

Allocates and returns a list of all transport records in the database.

Parameters

<i>tlist_ptr</i>	pointer to an array that will be allocated, value will be NULL on return if there are no transports
<i>num_entries_ptr</i>	pointer to the number of items in the returned array, value will be 0 on return if there are no transports

4.10.2.208 U32BIT* ADB_GetTransportListFreqs (void ** *tlist*, U16BIT *num_entries*)

Allocates and returns an array of tuning frequencies for each transport in the given list. The returned array should be freed using STB_AppFreeMemory.

Parameters

<i>tlist</i>	the array of transport records
<i>num_entries</i>	number of items in the list

Returns

pointer to the array of frequencies, NULL if no transports are provided or memory allocation fails

4.10.2.209 U8BIT ADB_GetTransportListNames (void ** *tlist*, U16BIT *num_entries*)**

Returns a list of names, in UTF-8 format, corresponding to the given transport list. The returned list should be freed using ADB_ReleaseNameList.

Parameters

<i>tlist</i>	the array of transport records
<i>num_entries</i>	number of items in the list

Returns

pointer to the list of name strings

4.10.2.210 U32BIT* ADB_GetTransportListOriginalNetworkIds (void ** *tlist*, U16BIT *num_entries*)

Allocates and returns an array of original network ids for each transport in the given list was tuned to. The returned array should be freed using STB_AppFreeMemory.

Parameters

<i>tlist</i>	the array of transport records
<i>num_entries</i>	number of items in the list

Returns

pointer to the array of original network ids, NULL if no transports are provided or memory allocation fails

4.10.2.211 U32BIT* ADB_GetTransportListTids (void ** *tlist*, U16BIT *num_entries*)

Allocates and returns an array of transport ids for the given transports. The returned array should be freed using STB_AppFreeMemory.

Parameters

<i>tlist</i>	the array of transport records
<i>num_entries</i>	number of items in the list

Returns

pointer to the transport id array, NULL if no transports are provided or memory allocation fails

4.10.2.212 U32BIT* ADB_GetTransportListTunedStrengths (void ** *tlist*, U16BIT *num_entries*)

Allocates and returns an array of percentage values representing the signal strength when each transport in the given list was tuned to. The returned array should be freed using STB_AppFreeMemory.

Parameters

<i>tlist</i>	the array of transport records
<i>num_entries</i>	number of items in the list

Returns

pointer to the signal strength array, NULL if no transports are provided or memory allocation fails

4.10.2.213 E_STB_TUNE_TCODERATE ADB_GetTransportLpCodeRate (void * *t_ptr*)

Returns the LP coderate of the given DVB-T/T2 transport.

Parameters

<i>t_ptr</i>	transport
--------------	-----------

Returns

constellation, returns TUNE_TCODERATE_UNDEFINED if transport isn't DVB-T/-T2

4.10.2.214 U8BIT* ADB_GetTransportName (void * *t_ptr*)

Returns the name, in UTF-8 format, of the given transport. The returned name should be freed using STB_ReleaseUnicodeString.

Parameters

<i>t_ptr</i>	transport
--------------	-----------

Returns

UTF-8 format string, NULL on failure

4.10.2.215 void* ADB_GetTransportNetworkPtr (void * *t_ptr*)

Returns the network of the given transport.

Parameters

<i>t_ptr</i>	transport
--------------	-----------

Returns

pointer to the network, possibly NULL no NIT was found for the transport

4.10.2.216 S8BIT ADB_GetTransportOffset (void * *t_ptr*)

Returns the tuning offset of the given DVB-T/T2 transport.

Parameters

<i>t_ptr</i>	transport
--------------	-----------

Returns

bandwidth, return value is undefined if transport isn't DVB-T/T2

4.10.2.217 U16BIT ADB_GetTransportOriginalNetworkId (void * *t_ptr*)

Returns the original network id of the given transport.

Parameters

<i>t_ptr</i>	transport
--------------	-----------

Returns

original network id, 0 if transport isn't found

4.10.2.218 void* ADB_GetTransportSatellite (void * *t_ptr*)

Returns the parent satellite for the given transport.

Parameters

<i>t_ptr</i>	transport
--------------	-----------

Returns

pointer to satellite

4.10.2.219 U16BIT ADB_GetTransportSatelliteSymbolRate (void * *t_ptr*)

Returns the symbol rate of the given satellite transponder.

Parameters

<i>t_ptr</i>	satellite transponder
--------------	-----------------------

Returns

symbol rate, 0 if transport isn't DVB-S

4.10.2.220 void **ADB_GetTransportSatTuningParams** (void * *t_ptr*, U32BIT * *freq_hz*, E_STB_DP_POLARITY * *polarity*, U16BIT * *symrate*, E_STB_DP_FEC * *fec*, BOOLEAN * *dvb_s2*, E_STB_DP_MODULATION * *modulation*)

Returns the parameters needed to tune a satellite tuner to a transport.

Parameters

<i>t_ptr</i>	transport to tune to
<i>freq_hz</i>	pointer to the returned freq of the transport
<i>polarity</i>	pointer to the returned polarity of the transport
<i>symrate</i>	pointer to the returned symbol rate of the transport
<i>fec</i>	pointer to the returned FEC of the transport
<i>dvb_s2</i>	returned as TRUE if a DVB_S2 transport
<i>modulation</i>	pointer to returned modulation used by the transport

4.10.2.221 BOOLEAN **ADB_GetTransportSearchFlag** (void * *t_ptr*)

Returns whether a transport has been used during a service scan operation.

Parameters

<i>t_ptr</i>	transport
--------------	-----------

Returns

TRUE if the transport has been used, FALSE otherwise

4.10.2.222 void **ADB_GetTransportServiceList** (void * *t_ptr*, void *** *slist_ptr*, U16BIT * *num_entries_ptr*)

Allocates and returns a list of all services in the database on the given transport.

Parameters

<i>t_ptr</i>	transport to be queried
<i>slist_ptr</i>	pointer to an array that will be allocated, value will be NULL on return if there are no services
<i>num_entries_ptr</i>	pointer to the number of items in the returned array, value will be 0 on return if there are no services

4.10.2.223 E_STB_DP_SIGNAL_TYPE ADB_GetTransportSignalType (void * *t_ptr*)

Returns the signal type for the given transport.

Parameters

<i>t_ptr</i>	transport
--------------	-----------

Returns

signal type

4.10.2.224 void ADB_GetTransportTerrestrialTuningParams (void * *t_ptr*,
E_STB_DP_TTYPE * *terr_type*, U32BIT * *freq_hz*, E_STB_DP_TMODE * *mode*,
E_STB_DP_TBWIDTH * *bwidth*, U8BIT * *plp_id*)

Returns the settings to tune to the given DVB-T/T2 transport.

Parameters

<i>t_ptr</i>	transport to get the tuning parameters for
<i>terr_type</i>	pointer to return the terrestrial type, TERR_TYPE_UNKNOWN on error
<i>freq_hz</i>	pointer to return the tuning frequency in Hz, 0 on error
<i>mode</i>	pointer to return the terrestrial mode, MODE_COFDM_UNDEFINED on error
<i>bwidth</i>	pointer to return the bandwidth
<i>plp_id</i>	pointer to return the PLP value for DVB-T2, will be 0 for DVB-T

4.10.2.225 E_STB_DP_TMODE ADB_GetTransportTerrMode (void * *t_ptr*)

Returns the terrestrial mode of the given DVB-T/T2 transport.

Parameters

<i>t_ptr</i>	transport
--------------	-----------

Returns

mode, returns MODE_COFDM_UNDEFINED if transport isn't DVB-T/T2

4.10.2.226 U16BIT ADB_GetTransportTid (void * *t_ptr*)

Returns the transport id of the given transport.

Parameters

<i>t_ptr</i>	transport
--------------	-----------

Returns

transport id, 0 if transport isn't found

4.10.2.227 U8BIT ADB_GetTransportTunedQuality (void * *t_ptr*)

Returns the signal quality, as a percentage, of the tuned transport.

Parameters

<i>t_ptr</i>	cable transport
--------------	-----------------

Returns

signal quality as a percentage

4.10.2.228 U8BIT ADB_GetTransportTunedStrength (void * *t_ptr*)

Returns the signal strength, as a percentage, of the tuned transport.

Parameters

<i>t_ptr</i>	cable transport
--------------	-----------------

Returns

signal strength as a percentage

4.10.2.229 U32BIT ADB_GetTtextStreamLangCode (void * *stream_ptr*)

Returns the 3 char language code of the given teletext stream.

Parameters

<i>stream_ptr</i>	stream
-------------------	--------

Returns

3 char language code encoded as a 32-bit number

4.10.2.230 U8BIT ADB_GetTtextStreamMagazine (void * *stream_ptr*)

Returns the magazine value of the given teletext stream.

Parameters

<i>stream_ptr</i>	stream
-------------------	--------

Returns

magazine value

4.10.2.231 U8BIT ADB_GetTtextStreamPage (void * *stream_ptr*)

Returns the page value of the given teletext stream.

Parameters

<i>stream_ptr</i>	stream
-------------------	--------

Returns

page value

4.10.2.232 U8BIT ADB_GetTtextStreamType (void * *stream_ptr*)

Returns the type, as defined in the PMT, of the given teletext stream.

Parameters

<i>stream_ptr</i>	stream
-------------------	--------

Returns

teletext stream type (ADB_TELETEXT_TYPE)

4.10.2.233 void* ADB_GetTunedNetwork (U8BIT *path*)

Returns the network currently tuned to on the given decode path.

Parameters

<i>path</i>	decode path
-------------	-------------

Returns

network pointer, or NULL if invalid path or not tuned

4.10.2.234 void* ADB_GetTunedService (U8BIT *path*)

Returns the tuned service for the given decode path.

Parameters

<i>path</i>	decode path
-------------	-------------

Returns

tuned service, or NULL

4.10.2.235 void* ADB_GetTunedTransport (U8BIT *path*)

Returns the transport that's tuned to on the given decode path.

Parameters

<i>path</i>	decode path
-------------	-------------

Returns

transport, NULL if not tuned

4.10.2.236 void ADB_GetUnavailableServiceList (void *** *slist_ptr*, U16BIT * *num_entries_ptr*)

Allocates and returns a list of all services that are marked as being unavailable. A service becomes unavailable when it disappears from the SDT.

Parameters

<i>slist_ptr</i>	pointer to an array that will be allocated, value will be NULL on return if there are no services
<i>num_entries_ptr</i>	pointer to the number of items in the returned array, value will be 0 on return if there are no services

4.10.2.237 E_STB_DP_VIDEO_CODEC ADB_GetVideoCodecFromStream (ADB_STREAM_TYPE *video_stream_type*)

Derive video codec from stream.

Parameters

<i>video_stream_type</i>	from which to derive the video codec
--------------------------	--------------------------------------

Returns

video codec best corresponding to *video_stream_type*

4.10.2.238 BOOLEAN ADB_ImportDB (U8BIT * *filename*)

Imports the database from an XML file.

Parameters

<i>filename</i>	full pathname of the file to import from
-----------------	--

Returns

TRUE if successful, otherwise FALSE

4.10.2.239 BOOLEAN ADB_IsCridForRadioService (void * *c_ptr*)

Returns whether the CRID record is for a radio service (TRUE)

Parameters

<i>c_ptr</i>	pointer to CRID record
--------------	------------------------

Returns

TRUE if the CRID is for a radio service

4.10.2.240 BOOLEAN ADB_IsFreesatService (void * *s_ptr*)

Returns a value indicating whether the given service is a Freesat service.

Parameters

<i>s_ptr</i>	service handle
--------------	----------------

Returns

TRUE if service is a Freesat service, FALSE otherwise

4.10.2.241 BOOLEAN ADB_IsProgrammeCrid (void * *c_ptr*)

Returns TRUE if the CRID record given represents a programme (split event)

Parameters

<i>c_ptr</i>	pointer to CRID record
--------------	------------------------

Returns

TRUE if the CRID record is for a programme

4.10.2.242 BOOLEAN ADB_IsRCTLinkGroupTrailer (void * *link_ptr*)

Returns whether the given link is for a group trailer.

Parameters

<i>link_ptr</i>	pointer to an RCT link
-----------------	------------------------

Returns

Link's is_group_trailer flag

4.10.2.243 BOOLEAN ADB_IsRecommendationCrid (void * *c_ptr*)

Returns TRUE if the CRID record given represents a recommendation.

Parameters

<i>c_ptr</i>	pointer to CRID record
--------------	------------------------

Returns

TRUE if the CRID record is for a recommended event/series

4.10.2.244 BOOLEAN ADB_IsSameEvent (void * *event1_ptr*, void * *event2_ptr*)

Checks whether the two events are the same. Not all fields are checked, just date, time, duration and event ID.

Parameters

<i>event1_ptr</i>	pointer to the first of the events to check
<i>event2_ptr</i>	pointer to the second of the events to check

Returns

TRUE if the two events are the same, FALSE otherwise

4.10.2.245 BOOLEAN ADB_IsSeriesCrid (void * *c_ptr*)

Returns TRUE if the CRID record given represents a series.

Parameters

<i>c_ptr</i>	pointer to CRID record
--------------	------------------------

Returns

TRUE if the CRID record is for a series

4.10.2.246 BOOLEAN ADB_IsSplitProgrammeCrid (U8BIT * *crid*)

Returns TRUE if the given CRID represents a split event (i.e. it contains an Instance Metadata Identifier (IMI)).

Parameters

<i>crid</i>	CRID string
-------------	-------------

Returns

TRUE if an IMI is found

4.10.2.247 **BOOLEAN ADB_IsValidService (void * *serv_ptr*)**

Checks whether the given service is in the current service list.

Parameters

<i>serv_ptr</i>	service to be checked
-----------------	-----------------------

Returns

TRUE if the service is found, FALSE otherwise

4.10.2.248 **void* ADB_LaterEvent (void * *serv_ptr*, U32DHMS *time*)**

Returns a copy of the event following the given date/time on the given service.

Parameters

<i>serv_ptr</i>	service for the event
<i>time</i>	Date/time from which to start searching

Returns

pointer to a copy of the event, or NULL if an event isn't found

4.10.2.249 **BOOLEAN ADB_MoveFavouriteListServiceFromTo (U8BIT *list_id*, S16BIT *current_index*, S16BIT *new_index*)**

Change the order of the services in the given favourite lists by moving it to the given position.

Parameters

<i>list_id</i>	favourite list to be moved
<i>current_index</i>	index of the service to be moved
<i>new_index</i>	position in the list to move the service to, negative value will move service to the end of the list

Returns

TRUE if the service is moved successfully, FALSE otherwise

4.10.2.250 **BOOLEAN** ADB_MoveFavouriteListServiceTo (U8BIT *list_id*, void * *serv_ptr*, S16BIT *index*)

Change the order of the services in the given favourite lists by moving it to the given position.

Parameters

<i>list_id</i>	favourite list to be moved
<i>serv_ptr</i>	service to be moved
<i>index</i>	position in the list to move the service to, negative value will move service to the end of the list

Returns

TRUE if the service is moved successfully, FALSE otherwise

4.10.2.251 **BOOLEAN** ADB_MoveFavouriteListTo (U8BIT *list_id*, S16BIT *index*)

Change the order of the favourite lists by moving the given list to the given position.

Parameters

<i>list_id</i>	favourite list to be moved
<i>index</i>	position to move the list to, negative value will move the list to the end

Returns

TRUE if the list is moved successfully, FALSE otherwise

4.10.2.252 **void** ADB_PrepareDatabaseForSearch (E_STB_DP_SIGNAL_TYPE *tuner_type*, void * *satellite*, **BOOLEAN** *retune*, **BOOLEAN** *manual_search*)

Sets up the database for a search.

Parameters

<i>tuner_type</i>	type of tuner that will be used when doing the search
<i>satellite</i>	satellite that the search will be performed on, NULL if not relevant
<i>retune</i>	TRUE if a retune will be performed
<i>manual_search</i>	TRUE for manual search, FALSE otherwise

4.10.2.253 void ADB_ReleaseCridRecordList (void ** *crid_list*, U16BIT *num_entries*)

Frees memory allocated for a CRID list due to a call to ADB_GetCridRecordList.

Parameters

<i>crid_list</i>	crid array
<i>num_entries</i>	number of items in the array

4.10.2.254 void ADB_ReleaseEventData (void * *event_ptr*)

Frees any memory allocated for the given event and the event itself.

Parameters

<i>event_ptr</i>	event to be freed
------------------	-------------------

4.10.2.255 void ADB_ReleaseEventItemizedInfo (ADB_EVENT_ITEMIZED_INFO * *event_itemized_info*, U16BIT *num_of_items*)

Frees the memory used by event itemized data.

Parameters

<i>event_itemized_info</i>	pointer of info
<i>num_of_items</i>	number of items to release

4.10.2.256 void ADB_ReleaseEventList (void ** *elist*, U16BIT *num_entries*)

Frees all the events in the given list and all associated memory for those events.

Parameters

<i>elist</i>	event list to be freed
<i>num_entries</i>	number of events in the list

4.10.2.257 void ADB_ReleaseEventSIDescriptorData (U8BIT * *desc_data*, U16BIT *desc_len*)

Frees the data returned by ADB_GetEventSIDescriptorData.

Parameters

<i>desc_data</i>	data to be freed
<i>desc_len</i>	number of bytes

4.10.2.258 void ADB_ReleaseNameList (U8BIT ** *name_list*, U16BIT *num_names*)

Frees the memory used by any of the name lists (e.g. networks, services, etc)

Parameters

<i>name_list</i>	name list to be freed
<i>num_names</i>	number of items in the name list

4.10.2.259 void ADB_ReleaseNetworkList (void ** *nlist*)

Frees a network list returned by ADB_GetNetworkList.

Parameters

<i>nlist</i>	network list to be freed
--------------	--------------------------

4.10.2.260 void ADB_ReleaseProfileList (void ** *profile_list*, U16BIT *num_profiles*)

Frees a profile list acquired using ADB_GetProfileList.

Parameters

<i>profile_list</i>	pointer to the array of profiles.
<i>num_profiles</i>	number of profiles in the array

4.10.2.261 void ADB_ReleaseRCTLinks (void * *links*)

Frees the given list of RCT links.

Parameters

<i>links</i>	links to be freed
--------------	-------------------

4.10.2.262 void ADB_ReleaseServiceList (void ** *slist*, U16BIT *num_servs*)

Frees a list of services returned from one of the functions that returns a service list, such as ADB_GetServiceList.

Parameters

<i>slist</i>	service list to be freed
<i>num_servs</i>	number of services in the list

4.10.2.263 void ADB_ReleaseStreamList (void ** *streamlist_ptr*, U16BIT *num_entries*)

Frees the memory allocated for a stream list using ADB_GetStreamList.

Parameters

<i>streamlist_ptr</i>	stream list to be freed
<i>num_entries</i>	number of items in the stream list

4.10.2.264 void ADB_ReleaseTransportList (void ** *tlist*, U16BIT *num_transports*)

Frees a transport list returned from one of the ADB_GetTransportList functions.

Parameters

<i>tlist</i>	transport list to be freed
<i>num_transports</i>	number of transports in the list

4.10.2.265 void ADB_ReportNoSignalDuringSearch (void * *t_ptr*)

Set the signal level to 0 and searched flag to TRUE for the given transport.

Parameters

<i>t_ptr</i>	pointer to the transport
--------------	--------------------------

4.10.2.266 void ADB_SaveCridRecord (void * *c_ptr*)

Saves a CRID record to non-volatile storage.

Parameters

<i>c_ptr</i>	pointer to CRID record
--------------	------------------------

4.10.2.267 void ADB_SaveEventSchedule (E_STB_DP_SIGNAL_TYPE *tuner_type*, void * *satellite*)

Saves the event schedule for all services for the given type of tuner to the database.
Note: the database has to support this operation.

Parameters

<i>tuner_type</i>	tuner type that the services must be on
<i>satellite</i>	if <i>tuner_type</i> is SIGNAL_QPSK then only events for services on the given satellite will be saved. If NULL, events for services on all satellites will be saved.

4.10.2.268 BOOLEAN ADB_ServiceAddImageIcon (void * *serv_ptr*, void * *icon_ptr*)

Adds the given image icon to the end of the service's icon list. The icon id is checked and if it matches an icon already in the list then the new icon replaces the existing one.

Parameters

<i>serv_ptr</i>	service to which link is to be added
<i>icon_ptr</i>	icon to be added

Returns

TRUE if the icon is added, FALSE otherwise

4.10.2.269 void ADB_ServiceAddRCTLink (void * *serv_ptr*, void * *link_ptr*)

Adds the given RCT link info to the end of the list of existing RCT links already defined for the given service.

Parameters

<i>serv_ptr</i>	service to which link is to be added
<i>link_ptr</i>	link to be added

4.10.2.270 BOOLEAN ADB_ServiceGetRCTIcon (void * *serv_ptr*, U8BIT ** *icon_data*, U32BIT * *data_size*, BOOLEAN * *pos_valid*, U16BIT * *x_pos*, U16BIT * *y_pos*, U16BIT * *width*, U16BIT * *height*, E_ICON_COORD_SYSTEM * *coord_system*)

Searches all the RCT links for a service to see if any of them define an icon to be used, and if one is found then all the data required to display the icon is returned.

Parameters

<i>serv_ptr</i>	service
<i>icon_data</i>	address of pointer to icon data. This data shouldn't be freed.
<i>data_size</i>	pointer to variable containing amount of data
<i>pos_valid</i>	pointer to boolean indicating whether returned x,y values are valid
<i>x_pos</i>	pointer to returned X position to display the icon
<i>y_pos</i>	pointer to returned Y position to display the icon
<i>width</i>	pointer to returned width of the icon
<i>height</i>	pointer to returned height of the icon
<i>coord_system</i>	pointer to coord system of returned pos and size values

Returns

TRUE if icon data is returned, FALSE otherwise

4.10.2.271 BOOLEAN ADB_ServiceHasSubtitles (void * *serv_ptr*, BOOLEAN * *dvb_subs*)

Determines whether the given service has subtitles, DVB or teletext.

Parameters

<i>serv_ptr</i>	service
<i>dvb_subs</i>	returned as TRUE if the subtitles are DVB, FALSE for teletext

Returns

TRUE if subtitles are available

4.10.2.272 **BOOLEAN ADB_ServiceRCTCanUseDefaultIcon** (void * *serv_ptr*)

Checks all the RCT links for the given service to determine whether any of them specify that the default icon can be used.

Parameters

<i>serv_ptr</i>	service containing the RCT links
-----------------	----------------------------------

Returns

TRUE if default icon can be used, FALSE otherwise

4.10.2.273 **void ADB_ServiceReleaseRCTLinks** (void * *serv_ptr*)

Frees all RCT link info for the given service.

Parameters

<i>serv_ptr</i>	service from which links are to be cleared
-----------------	--

4.10.2.274 **void ADB_SetAudioLang** (U32BIT *country_code*, U8BIT *lang_id*)

Sets the primary audio language to be used.

Parameters

<i>country_code</i>	current country code
<i>db_lang_id</i>	language id

4.10.2.275 **void ADB_SetCridDateTime** (void * *c_ptr*, U32DHMS *datetime*)

Sets the date and time fields on the given CRID record.

Parameters

<i>c_ptr</i>	pointer to CRID record
<i>datetime</i>	date and time to be saved in the record

4.10.2.276 void **ADB_SetCridProgrammeName** (void * *c_ptr*, U8BIT * *prog_name*)

Sets the programme name field of the given CRID record.

Parameters

<i>c_ptr</i>	pointer to CRID record
<i>prog_name</i>	pointer to name string, the name is copied

4.10.2.277 void **ADB_SetCridService** (void * *c_ptr*, U16BIT *serv_id*)

Sets the service ID on the given CRID record.

Parameters

<i>c_ptr</i>	pointer to CRID record
<i>serv_id</i>	service id

4.10.2.278 BOOLEAN **ADB_SetFavouriteListName** (U8BIT *list_id*, U8BIT * *name*)

Set the name of the given favourite list.

Parameters

<i>list_id</i>	list id
<i>name</i>	of the favourite list

Returns

TRUE if the name is set successfully, FALSE otherwise

4.10.2.279 void **ADB_SetFavouriteListUserData** (U8BIT *list_id*, U32BIT *user_data*)

Set the user data of the given favourite list.

Parameters

<i>list_id</i>	list id
<i>user_data</i>	value

4.10.2.280 void **ADB_SetReqdAudioStreamSettings** (void * *s_ptr*, BOOLEAN *valid*, U32BIT *lang_code*, ADB_AUDIO_TYPE *audio_type*, ADB_STREAM_TYPE *stream_type*)

Explicitly sets or clears the stream that will be used for audio on the given service. If 'valid' is TRUE then the stream is set according to the given language and types, but if FALSE then the audio stream selected for the service will be based on the default settings.

Parameters

<i>s_ptr</i>	service
<i>valid</i>	TRUE to set the stream using given values, FALSE to use defaults
<i>lang_code</i>	audio language to be used for the selected stream if 'valid' is TRUE
<i>audio_type</i>	audio type to be used for the selected stream if 'valid' is TRUE
<i>stream_type</i>	stream type to be used for the selected stream if 'valid' is TRUE

4.10.2.281 void ADB_SetReqdSubtitleStreamSettings (void * *s_ptr*, BOOLEAN *valid*, U32BIT *lang_code*, ADB_SUBTITLE_TYPE *subt_type*)

Explicitly sets or clears the stream that will be used for subtitles on the given service. If 'valid' is TRUE then the stream is set according to the given language and type, but if FALSE then the stream selected for the service will be based on the default settings.

Parameters

<i>s_ptr</i>	service
<i>valid</i>	TRUE to set the stream using given values, FALSE to use defaults
<i>lang_code</i>	language to be used for the selected stream if 'valid' is TRUE
<i>subt_type</i>	subtitle type to be used for the selected stream if 'valid' is TRUE

4.10.2.282 void ADB_SetReqdTeletextStreamSettings (void * *s_ptr*, BOOLEAN *valid*, U32BIT *lang_code*, ADB_TELETEXT_TYPE *ttext_type*)

Explicitly sets or clears the stream that will be used for teletext on the given service. If 'valid' is TRUE then the stream is set according to the given language and type, but if FALSE then the stream selected for the service will be based on the default settings.

Parameters

<i>s_ptr</i>	service
<i>valid</i>	TRUE to set the stream using given values, FALSE to use defaults
<i>lang_code</i>	language to be used for the selected stream if 'valid' is TRUE
<i>ttext_type</i>	teletext type to be used for the selected stream if 'valid' is TRUE

4.10.2.283 void ADB_SetSecondaryAudioLang (U32BIT *country_code*, U8BIT *lang_id*)

Sets the secondary audio language to be used. This will be selected if the primary audio language isn't available.

Parameters

<i>country_code</i>	current country code
<i>lang_id</i>	language id

4.10.2.284 void ADB_SetSecondaryTextLang (U32BIT *country_code*, U8BIT *lang_id*)

Sets the secondary subtitle and teletext language to be used. This will be used if the primary language isn't available.

Parameters

<i>country_code</i>	current country code
<i>lang_id</i>	language id

4.10.2.285 void ADB_SetServiceFavGroups (void * *s_ptr*, U8BIT *groups*)

Sets the favourite group value for the given service. This is an 8-bit value that can define whether the service is in any of upto 8 favourite groups. The value is a bit flag, so 1 means it's in group 1, 2 means it's in group 2, 4 means it's in group 3, 7 means it's in groups 1, 2 and 3, etc.

Parameters

<i>s_ptr</i>	service
<i>groups</i>	bit value, where a 1 means the service is in a group and 0 that it isn't

4.10.2.286 void ADB_SetServiceHiddenFlag (void * *s_ptr*, BOOLEAN *hidden*)

Sets the hidden status of the given service, which means the service would not be presented to the user in any list of available services.

Parameters

<i>s_ptr</i>	service
<i>hidden</i>	TRUE to hide the service, FALSE to make it visible

4.10.2.287 void ADB_SetServiceLcn (void * *s_ptr*, U16BIT *lcn*)

Sets the assigned logical channel number of the given service. This will override any LCN assigned when performing a service scan.

Parameters

<i>s_ptr</i>	service
<i>lcn</i>	logical channel number

4.10.2.288 void ADB_SetServiceListLockedFlag (void ** *slist*, U16BIT *num_entries*, BOOLEAN *locked*)

Sets the locked state of all services in the given service list.

Parameters

<i>slist</i>	list of services
<i>num_entries</i>	number of services in the list
<i>locked</i>	TRUE if the services are to be locked, or FALSE to unlock

4.10.2.289 void **ADB_SetServiceLockedFlag** (void * *s_ptr*, BOOLEAN *locked*)

Locks or unlocks the given service.

Parameters

<i>s_ptr</i>	service
<i>locked</i>	TRUE to lock the service, FALSE to unlock

4.10.2.290 void **ADB_SetServiceNowNextState** (void * *s_ptr*, BOOLEAN *state*)

Sets whether the now/next EIT events for this service are held in memory. All services save now/next events by default.

Parameters

<i>s_ptr</i>	service
<i>state</i>	FALSE to disable, TRUE to enable

4.10.2.291 void **ADB_SetServiceScheduleState** (void * *s_ptr*, BOOLEAN *state*)

Sets whether the EIT schedule events for this service are held in memory. All services hold schedule events in memory by default.

Parameters

<i>s_ptr</i>	service
<i>state</i>	FALSE to disable, TRUE to enable

4.10.2.292 void **ADB_SetTextLang** (U32BIT *country_code*, U8BIT *lang_id*)

Sets the primary subtitle and teletext language to be used.

Parameters

<i>country_code</i>	current country code
<i>lang_id</i>	language id

4.10.2.293 void **ADB_SetTransportSearchFlag** (void * *t_ptr*, BOOLEAN *state*)

Sets whether a transport has been used during a service scan operation.

Parameters

<i>t_ptr</i>	transport
<i>state</i>	set the flag to TRUE or FALSE

4.10.2.294 void ADB_SetTunedService (U8BIT *path*, void * *s_ptr*)

Sets the tuned service for the given path, together with the associated tuned transport and network. The 'new' flag for the service is also cleared.

Parameters

<i>path</i>	decode path
<i>s_ptr</i>	service

4.10.2.295 void ADB_SetTunedTransport (U8BIT *path*, void * *t_ptr*)

Sets the given transport as the one tuned to on the given decode path. The transport's network is also set as the tuned network, if valid.

Parameters

<i>path</i>	decode path
<i>t_ptr</i>	transport

4.10.2.296 void ADB_SortServiceListAlphabetically (void ** *slist*, U16BIT *num_entries*, BOOLEAN *short_name*)

Sorts the given list of services into alphabetical order. Case is significant.

Parameters

<i>slist</i>	service list to be sorted
<i>num_entries</i>	number of services in the list
<i>short_name</i>	TRUE if the short name for a service should be used

4.10.2.297 void ADB_ToggleServiceLockedFlag (void * *s_ptr*)

Changes the current state of the given service from locked to unlocked, or vice versa.

Parameters

<i>s_ptr</i>	service
--------------	---------

4.10.2.298 void ADB_UpdateCridEitDate (void * *c_ptr*)

Updates the time the CRID was last seen in the EIT.

Parameters

<code>c_ptr</code>	pointer to CRID record
--------------------	------------------------

4.11 dvb/inc/ap_epgsearch.h File Reference

macros and function prototypes for public use

Data Structures

- struct [s_ags_time_range](#)

Typedefs

- typedef enum e_ags_match_type **E_AGS_MATCH_TYPE**
- typedef enum e_ags_match_action **E_AGS_MATCH_ACTION**
- typedef enum e_ags_search_fields **E_AGS_SEARCH_FIELDS**
- typedef enum e_ags_search_days **E_AGS_SEARCH_DAYS**
- typedef struct [s_ags_time_range](#) **S_AGS_TIME_RANGE**

Enumerations

- enum **e_ags_match_type** { **AGS_MATCH_ANY_WORD** = 0, **AGS_MATCH_ALL_WORDS** = 1, **AGS_MATCH_EXACT_PHRASE** = 2, **AGS_MATCH_CONTAINS_PHRASE** = 3, **AGS_MATCH_NOT_EQUAL** = 4, **AGS_MATCH_IGNORE_CASE** = 0x80 }
- enum **e_ags_match_action** { **AGS_ACTION_NONE** = 0, **AGS_ACTION_SET_TIMER** = 1, **AGS_ACTION_SET_ALARM** = 2 }
- enum **e_ags_search_fields** { **AGS_SEARCH_ALL** = 0xff, **AGS_SEARCH_TITLE** = 0x01, **AGS_SEARCH_DESC** = 0x02, **AGS_SEARCH_EXT_DESC** = 0x04 }
- enum **e_ags_search_days** { **AGS_DAYS_ALL** = 0xff, **AGS_DAYS_WEEKDAYS** = 0x1f, **AGS_DAYS_WEEKENDS** = 0x60, **AGS_DAYS_MONDAY** = 0x01, **AGS_DAYS_TUESDAY** = 0x02, **AGS_DAYS_WEDNESDAY** = 0x04, **AGS_DAYS_THURSDAY** = 0x08, **AGS_DAYS_FRIDAY** = 0x10, **AGS_DAYS_SATURDAY** = 0x20, **AGS_DAYS_SUNDAY** = 0x40 }

Functions

- void * [AGS_OpenSearch](#) (U8BIT *search_str, E_AGS_SEARCH_FIELDS search_fields, E_AGS_MATCH_TYPE match_type, E_AGS_SEARCH_DAYS search_days, [S_AGS_TIME_RANGE](#) *search_times, E_AGS_MATCH_ACTION match_action)
Opens an EPG Search.
- BOOLEAN [AGS_GetNextResult](#) (void *ags_handle, U16BIT *event_id, void **serv_ptr)

Gets the next matching result for an open search.

- void [AGS_CloseSearch](#) (void *ags_handle)

Closes an open search and frees all associated resources.

4.11.1 Detailed Description

macros and function prototypes for public use

Date

13/08/2011

Author

Ocean Blue

4.11.2 Function Documentation

4.11.2.1 void AGS_CloseSearch (void * ags_handle)

Closes an open search and frees all associated resources.

Parameters

<i>ags_handle</i>	The handle of the open search
-------------------	-------------------------------

4.11.2.2 BOOLEAN AGS_GetNextResult (void * ags_handle, U16BIT * event_id, void ** serv_ptr)

Gets the next matching result for an open search.

Parameters

<i>ags_handle</i>	The handle of the open search
<i>event_id</i>	Caller's event id var to be set with matching result
<i>serv_ptr</i>	Caller's service pointer to be set with matching result

Returns

BOOLEAN, TRUE=Match Found, FALSE=No more matches

4.11.2.3 void* AGS_OpenSearch (U8BIT * search_str, E_AGS_SEARCH_FIELDS search_fields, E_AGS_MATCH_TYPE match_type, E_AGS_SEARCH_DAYS search_days, S_AGS_TIME_RANGE * search_times, E_AGS_MATCH_ACTION match_action)

Opens an EPG Search.

Parameters

<i>search_str</i>	The search string to compare
<i>search_ - fields</i>	The EPG parts to match
<i>match_type</i>	The type of comparison to make on the string
<i>search_days</i>	Which days of the week to search
<i>search_ - times</i>	Time range in which to search (NULL for any)
<i>match_ - action</i>	The action to perform on a search match

Returns

Handle for search, NULL if failed

4.12 dvb/inc/ap_ipadd.h File Reference

Contains the initialise functions for IP.

Enumerations

- enum **E_STB_IP_MODE** { **IP_DHCP**, **IP_STATIC** }

Functions

- void [AIP_ConnectToNetwork](#) (BOOLEAN wait_for_completion)
Connect to network based on IP_MODE from NVM e.g. restore from IP NVM and connect to network.

4.12.1 Detailed Description

Contains the initialise functions for IP.

Date

27/04/2010

Author

Ocean Blue

4.12.2 Function Documentation

4.12.2.1 void [AIP_ConnectToNetwork](#) (BOOLEAN *wait_for_completion*)

Connect to network based on IP_MODE from NVM e.g. restore from IP NVM and connect to network.

Parameters

<i>wait_for_completion</i>	if TRUE and the mode is DHCP this function blocks until the DHCP has returned a result
----------------------------	--

4.13 dvb/inc/ap_pvr.h File Reference

macros and function prototypes for public use

Typedefs

- typedef enum e_pvr_play_status **E_PVR_PLAY_STATUS**
- typedef enum e_pvr_stop_type **E_PVR_STOP_TYPE**

Enumerations

- enum **e_pvr_play_status** { **PVR_PLAY_STATUS_NULL** = 0, **PVR_PLAY_STATUS_STOP** = 1, **PVR_PLAY_STATUS_PAUSE** = 2, **PVR_PLAY_STATUS_SF** = 3, **PVR_PLAY_STATUS_SR** = 4, **PVR_PLAY_STATUS_NORMAL** = 5, **PVR_PLAY_STATUS_FF** = 6, **PVR_PLAY_STATUS_FR** = 7, **PVR_PLAY_STATUS_QR** = 8 }
- enum **e_pvr_stop_type** { **PVR_STOP_REC_ONLY** = 0, **PVR_STOP_PLAY** = 1, **PVR_STOP_PAUSE** = 2 }

Functions

- **BOOLEAN APVR_IsInitialised** (void)
Use to query whether the PVR function is available and ready to be used.
- **U16BIT APVR_GetPlayList** (U32BIT **handle_list, U32BIT **name_list, U32BIT **rec_status_list, U32BIT **locked_list, U32BIT **selected_list, U32BIT **split_list)
Returns a list of all the existing recordings that can be played. All the arrays returned will be allocated by this function and should be freed by calling APVR_ReleasePlayList.
- **void APVR_ReleasePlayList** (U32BIT *handle_list, U32BIT *name_list, U32BIT *rec_status_list, U32BIT *locked_list, U32BIT *selected_list, U32BIT *split_list)
Frees the arrays allocated by APVR_GetPlayList.
- **BOOLEAN APVR_PlayRecording** (U32BIT recording_handle, **BOOLEAN** resume_playback)
Starts playback of the given recording.
- **U32BIT APVR_GetPlaybackHandle** (void)
Returns the handle of the recording currently being played back.
- **BOOLEAN APVR_IsPlaying** (void)
Returns whether playback is in progress.

- **BOOLEAN** [APVR_GetPlaybackElapsedTime](#) (U32BIT handle, U8BIT *hours, - U8BIT *mins, U8BIT *secs, U8BIT *progress)
Returns the playback progress in time and as a percentage.
- **void** [APVR_StopPlay](#) (BOOLEAN return_to_live)
Stops playback and optionally returns to live TV.
- **void** [APVR_NormalPlay](#) (void)
If playback has been paused or is subject to some trick mode, calling this function will result in playback resuming at normal playback speed (i.e. 100%)
- **void** [APVR_PausePlay](#) (void)
Pause playback.
- **void** [APVR_FFPlay](#) (void)
If paused, playback will go into the first available slow motion speed in a forwards direction. If speed is already > 0% then the next fast forward speed will be selected. If playback is in the reverse direction then the next slowest reverse play speed will be selected.
- **void** [APVR_FRPlay](#) (void)
If paused, playback will go into the first available slow motion speed in a reverse direction. If speed is already < 0% then the next reverse speed will be selected. If playback is in the forwards direction then the next slowest forwards play speed will be selected.
- **E_PVR_PLAY_STATUS** [APVR_GetPlayMode](#) (void)
Returns the current play mode according to the current play speed.
- **S16BIT** [APVR_GetPlaySpeed](#) (void)
Returns the current play speed as a signed percentage value representing the speed, where 100% is normal playback speed.
- **U8BIT** [APVR_PrepareNewRecording](#) (U16BIT onet_id, U16BIT trans_id, U16BIT service_id, **BOOLEAN** *new_tuned_service)
Acquires a decode path for recording the given service and tunes to it.
- **BOOLEAN** [APVR_StartNewRecording](#) (U16BIT disk_id, U8BIT path, U8BIT *recording_name, U16BIT event_id, U8BIT *prog_crid, U8BIT *other_crid, U32BIT *rec_handle)
Starts a recording after any tuning has completed and sets the info to be stored with it.
- **BOOLEAN** [APVR_StopRecording](#) (U32BIT recording_handle)
Stops the given recording.
- **U8BIT** [APVR_GetActiveRecordingList](#) (U32BIT **handle_list)
Returns an array of recordings currently in progress. Use APVR_ReleaseActiveRecordingList to free the returned array.
- **void** [APVR_ReleaseActiveRecordingList](#) (U32BIT *handle_list)
Frees the arrays allocated by APVR_GetActiveRecordingList.
- **BOOLEAN** [APVR_GetPathRecordingName](#) (U8BIT path, U32BIT *name_ptr)
Get the name of the recording currently taking place on the given decode path.
- **BOOLEAN** [APVR_DeleteRecording](#) (U32BIT handle)
Delete the given recording, including all files associated with it and remove it from the database of available recordings that can be played back.
- **void** [APVR_SetTimeshiftBufferSize](#) (U16BIT time_in_mins)

Sets the size of the timeshift buffer so that it's able to store a recording of the given length before it has to loop round. Whether this value can be accurately achieved may vary according to the support provided by the platform.

- U8BIT [APVR_StartPauseRecord](#) (void)

Starts recording the live service ready for timeshift playback. The recording may not have started when this function returns, so an app should wait for the STB_EVENT_PVR_REC_START event before starting the paused playback using APVR_StartPausePlay.
- U8BIT [APVR_StartPausePlay](#) (void)

Starts playback of a live timeshift recording started with APVR_StartPauseRecord. - This function should not be called until STB_EVENT_PVR_REC_START has been received with the correct recording path as an argument (as returned by APVR_StartPauseRecord).
- void [APVR_StopPauseRecord](#) (BOOLEAN return_to_live)

Stops timeshifted playback and recording and optionally restarts decoding of the live signal.
- BOOLEAN [APVR_IsRecordingInProgress](#) (void)

Returns TRUE if there are any recordings currently in progress.
- BOOLEAN [APVR_IsRecordingHandle](#) (U8BIT path, U32BIT recording_handle)

Returns TRUE if given recording handle is still in progress on the given decode path.
- BOOLEAN [APVR_GetRecordingHandle](#) (U8BIT path, U32BIT *recording_handle_ptr)

Returns the handle of the recording taking place on the given decode path.
- U8BIT [APVR_GetTotalSpacePercentUsed](#) (U16BIT disk_id)

get total percentage of recording space used
- void [APVR_GetDiskTime](#) (U16BIT disk_id, U8BIT *used_hour_ptr, U8BIT *used_min_ptr, U8BIT *free_hour_ptr, U8BIT *free_min_ptr, U8BIT *size_hour_ptr, - U8BIT *size_min_ptr)

Gets an estimate of disk space in time (hour / min): used, free and size.
- void [APVR_GetDiskMbyte](#) (U16BIT disk_id, U32BIT *used_mbyte_ptr, U32BIT *free_mbyte_ptr, U32BIT *size_mbyte_ptr)

Get disk space in Mbytes: used, free and size.
- BOOLEAN [APVR_CheckSpaceForEvent](#) (U16BIT disk_id, void *event_ptr)

Checks if there is enough space for the recording of the given event to take place.
- BOOLEAN [APVR_CheckSpaceDuration](#) (U16BIT disk_id, U32BIT duration)

Check if there is enough space for the recording of the given duration to take place.
- BOOLEAN [APVR_CheckSpaceForDuration](#) (U16BIT disk_id, U8BIT hours, U8BIT mins)

Check if there is enough space for the recording of the given duration to take place.
- U32BIT [APVR_GetPauseProgress](#) (void)

Calculates and returns the number of seconds behind live TV.
- U8BIT [APVR_GetPlaybackPath](#) (void)

Returns the path currently acquired for playback.
- BOOLEAN [APVR_IsEventInRecordList](#) (U8BIT *prog_crid)

Determines whether the given programme CRID is already in the list of recorded events.

- void [APVR_RecordSeries](#) (void *crid_ptr, BOOLEAN radio_service)
Seaches the schedule for events that are part of the given series and that haven't already been recorded or aren't set to be recorded and creates a timer to record them.
- void [APVR_RecordRecommendations](#) (void *crid_ptr, BOOLEAN radio_service)
Searches the schedule for events that are part of the given recommendation and that haven't already been recorded or aren't set to be recorded and creates a timer to record them.
- BOOLEAN [APVR_FindNextSplitEvent](#) (U32BIT curr_handle, U32BIT *next_handle)
If the current playback recording is a split event then the next chronological part of the event is found and its handle returned.
- void [APVR_EitUpdated](#) (void)
Ensures that any EIT updates are processed to handle scenarios such as checking for new series recordings, etc.
- void [APVR_PidsUpdated](#) (U8BIT path)
This function should be called when there's an update to the PIDs for a service that's being recorded. The PIDs being recorded are checked and any that are no longer valid will no longer be recorded and any new PIDs will be added.
- void * [APVR_GetPlaybackService](#) (void)
A service instance is associated to the playback, this function returns its pointer. The handle can be used, for example, to call ADB_ServiceHasSubtitles as it's done for live channels.
- BOOLEAN [APVR_HandlePrivateTimer](#) (U32BIT timer_handle)
Deals with any private timers started by the PVR module.
- void [APVR_Initialise](#) (void)
Initialise all PVR resources: filters, recordings etc.
- void [APVR_Terminate](#) (void)
Destroy all PVR resources: filters, recordings etc.
- BOOLEAN [APVR_IsDecodingFile](#) (void)
Returns the PVR play status.
- void [APVR_SlowMoPlay](#) (void)
Play in slow mothing mode.
- void [APVR_QRPlay](#) (void)
- void [APVR_JTLPlay](#) (void)
- void [APVR_TrickModeMuteOn](#) (void)
Mute for trick mode.
- void [APVR_TrickModeMuteOff](#) (void)
Unmute for trick mode.
- BOOLEAN [APVR_DeleteAllSelectedRecordings](#) (U32BIT *handles, U16BIT num_handles)
Delete all the recordings marked as selected but not locked in the given list, including all files associated with it and remove it from the database of available recordings that can be played back.
- void [APVR_UnselectAllRecordings](#) (U32BIT *handles, U16BIT num_handles)
Un-select all recordings.

- BOOLEAN [APVR_AreSelectedRecordings](#) (U32BIT *handles, U16BIT num_handles)

Checks if any recording in a list is selected.

4.13.1 Detailed Description

macros and function prototypes for public use

Date

11/05/2004

Author

Ocean Blue

4.13.2 Function Documentation

4.13.2.1 BOOLEAN APVR_AreSelectedRecordings (U32BIT * handles, U16BIT num_handles)

Checks if any recording in a list is selected.

Parameters

<i>handles</i>	array of recording handles
<i>num_handles</i>	number of recording handles in the array

Returns

TRUE if at least one recording is selected, FALSE otherwise

4.13.2.2 BOOLEAN APVR_CheckSpaceDuration (U16BIT disk_id, U32BIT duration)

Check if there is enough space for the recording of the given duration to take place.

disk_id ID of disk to be queried

Parameters

<i>duration</i>	Recording duration expressed as U32DHMS
-----------------	---

Returns

TRUE if there is space available, FALSE otherwise

4.13.2.3 **BOOLEAN APVR_CheckSpaceForDuration** (U16BIT *disk_id*, U8BIT *hours*, U8BIT *mins*)

Check if there is enough space for the recording of the given duration to take place.

disk_id ID of disk to be queried

Parameters

<i>hours</i>	number of hours of the duration
<i>mins</i>	number of minutes of the duration

Returns

TRUE if there is space available, FALSE otherwise

4.13.2.4 **BOOLEAN APVR_CheckSpaceForEvent** (U16BIT *disk_id*, void * *event_ptr*)

Checks if there is enough space for the recording of the given event to take place.

disk_id ID of disk to be queried

Parameters

<i>event_ptr</i>	The event to check if there is any free space
------------------	---

Returns

TRUE if there is space available, FALSE otherwise

4.13.2.5 **BOOLEAN APVR_DeleteAllSelectedRecordings** (U32BIT * *handles*, U16BIT *num_handles*)

Delete all the recordings marked as selected but not locked in the given list, including all files associated with it and remove it from the database of available recordings that can be played back.

Parameters

<i>handles</i>	array of recording handles
<i>num_handles</i>	number of recordings in the array

Returns

TRUE if at least one recording is deleted, FALSE otherwise

4.13.2.6 **BOOLEAN APVR_DeleteRecording (U32BIT *handle*)**

Delete the given recording, including all files associated with it and remove it from the database of available recordings that can be played back.

Parameters

<i>handle</i>	recording to be deleted
---------------	-------------------------

Returns

TRUE if the recording is deleted, FALSE otherwise

4.13.2.7 **BOOLEAN APVR_FindNextSplitEvent (U32BIT *curr_handle*, U32BIT * *next_handle*)**

If the current playback recording is a split event then the next chronological part of the event is found and its handle returned.

Parameters

<i>curr_handle</i>	handle of split recording when searching for the next part
<i>next_handle</i>	handle of the next recording to be played, returned

Returns

TRUE if a recording is found, FALSE otherwise

4.13.2.8 **U8BIT APVR_GetActiveRecordingList (U32BIT ** *handle_list*)**

Returns an array of recordings currently in progress. Use APVR_ReleaseActiveRecordingList to free the returned array.

Parameters

<i>handle_list</i>	address of an array allocated by the function containing the recording handles
--------------------	--

Returns

number of recordings in progress

4.13.2.9 **void APVR_GetDiskMbyte (U16BIT *disk_id*, U32BIT * *used_mbyte_ptr*, U32BIT * *free_mbyte_ptr*, U32BIT * *size_mbyte_ptr*)**

Get disk space in Mbytes: used, free and size.

disk_id ID of disk to be queried *used_mbyte_ptr* amount of data stored on the disk
free_mbyte_ptr free space on the disk *size_mbyte_ptr* total size of the disk

4.13.2.10 void APVR_GetDiskTime (U16BIT *disk_id*, U8BIT * *used_hour_ptr*, U8BIT * *used_min_ptr*, U8BIT * *free_hour_ptr*, U8BIT * *free_min_ptr*, U8BIT * *size_hour_ptr*, U8BIT * *size_min_ptr*)

Gets an estimate of disk space in time (hour / min): used, free and size.

Parameters

<i>disk_id</i>	ID of disk to be queried
<i>used_hour_ptr</i>	pointer to the variable containing the number of hours that the stored data is worth
<i>used_min_ptr</i>	pointer to the variable containing the number of minutes that the stored data is worth
<i>free_hour_ptr</i>	pointer to the variable containing the number of hours that the available space is worth
<i>free_min_ptr</i>	pointer to the variable containing the number of minutes that the available space is worth
<i>size_hour_ptr</i>	pointer to the variable containing the number of hours that the total disk size is worth
<i>size_min_ptr</i>	pointer to the variable containing the number of minutes that the total disk size is worth

4.13.2.11 BOOLEAN APVR_GetPathRecordingName (U8BIT *path*, U32BIT * *name_ptr*)

Get the name of the recording currently taking place on the given decode path.

Parameters

<i>path</i>	decode path being used for a recording
<i>name_ptr</i>	address of variable into which the name reference will be returned. This is actually a U8BIT*, but the string shouldn't be freed.

Returns

TRUE if a recording is taking place on the given path and the name is returned,
FALSE otherwise

4.13.2.12 U32BIT APVR_GetPauseProgress (void)

Calculates and returns the number of seconds behind live TV.

Returns

Number of seconds behind live

4.13.2.13 BOOLEAN APVR_GetPlaybackElapsedTime (U32BIT *handle*, U8BIT * *hours*, U8BIT * *mins*, U8BIT * *secs*, U8BIT * *progress*)

Returns the playback progress in time and as a percentage.

Parameters

<i>handle</i>	recording handle being played
<i>hours</i>	pointer for return of playback hours
<i>mins</i>	pointer for return of playback minutes
<i>secs</i>	pointer for return of playback seconds
<i>progress</i>	pointer for return of position in playback as a percentage

Returns

TRUE if values are returned

4.13.2.14 U32BIT APVR_GetPlaybackHandle (void)

Returns the handle of the recording currently being played back.

Returns

playback handle, which may be invalid if playback isn't taking place

4.13.2.15 U8BIT APVR_GetPlaybackPath (void)

Returns the path currently acquired fro playback.

Returns

Decode path for playback

4.13.2.16 void* APVR_GetPlaybackService (void)

A service instance is associated to the playback, this function returns its pointer. The handle can be used, for example, to call ADB_ServiceHasSubtitles as it's done for live channels.

Returns

Playback service pointer

4.13.2.17 U16BIT APVR_GetPlayList (U32BIT ** *handle_list*, U32BIT ** *name_list*, U32BIT ** *rec_status_list*, U32BIT ** *locked_list*, U32BIT ** *selected_list*, U32BIT ** *split_list*)

Returns a list of all the existing recordings that can be played. All the arrays returned will be allocated by this function and should be freed by calling APVR_ReleasePlayList.

Parameters

<i>handle_list</i>	address to return an array of recording handles
<i>name_list</i>	address to return an array of names for each recording, the U32BIT value will actually be a U8BIT* of a UTF-8 string

<i>rec_status_list</i>	address to return an array indicating whether each recording is currently being recorded, 0=no, 1=yes
<i>locked_list</i>	address to return an array indicating whether each recording is parentally locked, 0=no, 1=yes
<i>selected_list</i>	address to return an array indicating whether each recording is marked as selected, 0=no, 1=yes
<i>split_list</i>	address to return an array indicating whether each recording is part of a split recording, 0=no, 1=yes

Returns

number of recordings

4.13.2.18 E_PVR_PLAY_STATUS APVR_GetPlayMode (void)

Returns the current play mode according to the current play speed.

Returns

play mode

4.13.2.19 S16BIT APVR_GetPlaySpeed (void)

Returns the current play speed as a signed percentage value representing the speed, where 100% is normal playback speed.

Returns

play speed

4.13.2.20 BOOLEAN APVR_GetRecordingHandle (U8BIT *path*, U32BIT * *recording_handle_ptr*)

Returns the handle of the recording taking place on the given decode path.

Parameters

<i>path</i>	decode path used for recording
<i>recording_handle_ptr</i>	address to return the recording handle, only valid on return if a recording is taking place

Returns

TRUE if a recording is taking place and handle is returned, FALSE otherwise

4.13.2.21 U8BIT APVR_GetTotalSpacePercentUsed (U16BIT *disk_id*)

get total percentage of recording space used

Parameters

<i>total_space_</i> <i>_used</i>	total percentage of recording space used
-------------------------------------	--

4.13.2.22 **BOOLEAN APVR_HandlePrivateTimer (U32BIT *timer_handle*)**

Deals with any private timers started by the PVR module.

Parameters

<i>timer_</i> - <i>handle</i>	handle of the timer that's been triggered
----------------------------------	---

Returns

TRUE if the timer is a PVR timer and has been dealt with, FALSE otherwise

4.13.2.23 **BOOLEAN APVR_IsDecodingFile (void)**

Returns the PVR play status.

Returns

TRUE if a recoding is being played, FALSE otherwise

4.13.2.24 **BOOLEAN APVR_IsEventInRecordList (U8BIT * *prog_crid*)**

Determines whether the given programme CRID is already in the list of recorded events.

Parameters

<i>prog_crid</i>	full CRID (i.e. including default authority) of event to be searched for
------------------	--

Returns

TRUE if a recording with the given CRID is found, FALSE otherwise

4.13.2.25 **BOOLEAN APVR_IsInitialised (void)**

Use to query whether the PVR function is available and ready to be used.

Returns

TRUE if available and ready, FALSE otherwise

4.13.2.26 **BOOLEAN APVR_IsPlaying (void)**

Returns whether playback is in progress.

Returns

TRUE if playback has been started, FALSE otherwise

4.13.2.27 **BOOLEAN APVR_IsRecordingHandle (U8BIT *path*, U32BIT *recording_handle*)**

Returns TRUE if given recording handle is still in progress on the given decode path.

Parameters

<i>path</i>	decode path used for recording
<i>recording_handle</i>	recording

Returns

TRUE if the recording is still in progress, FALSE otherwise

4.13.2.28 **BOOLEAN APVR_IsRecordingInProgress (void)**

Returns TRUE if there are any recordings currently in progress.

Returns

TRUE if recording in progress, FALSE otherwise

4.13.2.29 **void APVR_PidsUpdated (U8BIT *path*)**

This function should be called when there's an update to the PIDs for a service that's being recorded. The PIDs being recorded are checked and any that are no longer valid will no longer be recorded and any new PIDs will be added.

Parameters

<i>path</i>	recording path
-------------	----------------

4.13.2.30 **BOOLEAN APVR_PlayRecording (U32BIT *recording_handle*, BOOLEAN *resume_playback*)**

Starts playback of the given recording.

Parameters

<i>recording_handle</i>	handle of recorded programme to playback
<i>resume_playback</i>	TRUE if playback of the recording should resume from where it was last stopped, FALSE to start from the beginning

Returns

TRUE if playback starts, FALSE otherwise

4.13.2.31 U8BIT APVR_PrepareNewRecording (U16BIT *onet_id*, U16BIT *trans_id*, U16BIT *service_id*, BOOLEAN * *new_tuned_service*)

Acquires a decode path for recording the given service and tunes to it.

Parameters

<i>onet_id</i>	original network id for the service
<i>trans_id</i>	transport id for the service
<i>service_id</i>	service id for the service
<i>new_tuned_service</i>	returns TRUE if tuning is started. FALSE means the path is already tuned to the service

Returns

ID of path or INVALID_RES_ID on failure

4.13.2.32 void APVR_RecordRecommendations (void * *crid_ptr*, BOOLEAN *radio_service*)

Searches the schedule for events that are part of the given recommendation and that haven't already been recorded or aren't set to be recorded and creates a timer to record them.

Parameters

<i>crid_ptr</i>	pointer to the CRID record
<i>radio_service</i>	TRUE if the CRID is for a radio service

4.13.2.33 void APVR_RecordSeries (void * *crid_ptr*, BOOLEAN *radio_service*)

Searches the schedule for events that are part of the given series and that haven't already been recorded or aren't set to be recorded and creates a timer to record them.

Parameters

<i>crid_ptr</i>	pointer to the CRID record
<i>radio_service</i>	TRUE if the CRID is for a radio service

4.13.2.34 void APVR_ReleaseActiveRecordingList (U32BIT * *handle_list*)

Frees the arrays allocated by APVR_GetActiveRecordingList.

Parameters

<i>handle_list</i>	array of recording handles
--------------------	----------------------------

4.13.2.35 void **APVR_ReleasePlayList** (U32BIT * *handle_list*, U32BIT * *name_list*, U32BIT * *rec_status_list*, U32BIT * *locked_list*, U32BIT * *selected_list*, U32BIT * *split_list*)

Frees the arrays allocated by APVR_GetPlayList.

Parameters

<i>handle_list</i>	array of recording handles
<i>name_list</i>	array of recording names
<i>rec_status_list</i>	array of recording status'
<i>locked_list</i>	array of locked status'
<i>selected_list</i>	array of selected status'

4.13.2.36 void **APVR_SetTimeshiftBufferSize** (U16BIT *time_in_mins*)

Sets the size of the timeshift buffer so that it's able to store a recording of the given length before it has to loop round. Whether this value can be accurately achieved may vary according to the support provided by the platform.

Parameters

<i>time_in_mins</i>	timeshift buffer size in minutes
---------------------	----------------------------------

4.13.2.37 BOOLEAN **APVR_StartNewRecording** (U16BIT *disk_id*, U8BIT * *path*, U8BIT * *recording_name*, U16BIT *event_id*, U8BIT * *prog_crid*, U8BIT * *other_crid*, U32BIT * *rec_handle*)

Starts a recording after any tuning has completed and sets the info to be stored with it.

Parameters

<i>disk_id</i>	disk to record onto, default disk will be used if given disk id is invalid
<i>path</i>	decode path to use for recording
<i>recording_name</i>	name to be given to recording
<i>event_id</i>	event id which will be used to obtain info to save with the recording
<i>prog_crid</i>	programme crid (only relevant for systems that broadcast CRIDs)
<i>other_crid</i>	series or recommendation crid
<i>rec_handle</i>	pointer to returned recording handle

Returns

TRUE if the recording is started successfully, FALSE otherwise

4.13.2.38 U8BIT APVR_StartPausePlay (void)

Starts playback of a live timeshift recording started with APVR_StartPauseRecord. This function should not be called until STB_EVENT_PVR_REC_START has been received with the correct recording path as an argument (as returned by APVR_StartPauseRecord).

Returns

decode path used for the playback, INVALID_RES_ID on error

4.13.2.39 U8BIT APVR_StartPauseRecord (void)

Starts recording the live service ready for timeshift playback. The recording may not have started when this function returns, so an app should wait for the STB_EVENT_PVR_REC_START event before starting the paused playback using APVR_StartPausePlay.

Returns

decode path used for the pause recording, INVALID_RES_ID on error

4.13.2.40 void APVR_StopPauseRecord (BOOLEAN *return_to_live*)

Stops timeshifted playback and recording and optionally restarts decoding of the live signal.

Parameters

<i>return_to_live</i>	TRUE to start decoding the live service
-----------------------	---

4.13.2.41 void APVR_StopPlay (BOOLEAN *return_to_live*)

Stops playback and optionally returns to live TV.

Parameters

<i>return_to_live</i>	TRUE to return to live TV after stopping playback
-----------------------	---

4.13.2.42 BOOLEAN APVR_StopRecording (U32BIT *recording_handle*)

Stops the given recording.

Parameters

<i>recording_handle</i>	recording to be stopped
-------------------------	-------------------------

Returns

TRUE if recording stopped ok, FALSE otherwise

4.14 dvb/inc/ap_si.h File Reference

application level SI task

This graph shows which files directly or indirectly include this file:

Defines

- `#define GET_SIGNAL_STATUS(quality, strength) (((quality) & 0x0a) << 4 | ((strength) & 0x0a))`
- `#define BAD_SIGNAL_STATUS 0xff`
- `#define GET_SIGNAL_QUALITY(status) ((status) >> 4)`
- `#define GET_SIGNAL_STRENGTH(status) ((status) & 0x0f)`

Typedefs

- `typedef BOOLEAN(* F_SIManager)(U8BIT, U32BIT, SI_TABLE_RECORD *)`
- `typedef void(* F_EitParser)(U8BIT, U8BIT, U8BIT *, U32BIT, void *)`
- `typedef void(* F_BatTableUpdate)(U8BIT, SI_BAT_TABLE *, SI_TABLE_RECORD *)`
- `typedef void(* F_EitTableUpdate)(U8BIT, SI_TABLE_RECORD *)`
- `typedef void(* F_NitTableUpdate)(U8BIT, SI_NIT_TABLE *, SI_TABLE_RECORD *)`
- `typedef void(* F_PmtTableUpdate)(U8BIT, SI_PMT_TABLE *, SI_TABLE_RECORD *, void *)`
- `typedef void(* F_SdtTableUpdate)(U8BIT, SI_SDT_TABLE *, SI_TABLE_RECORD *)`

Enumerations

- `enum E_APP_SI_MODE { APP_SI_MODE_NO_SI, APP_SI_MODE_CHANNEL_SEARCH, APP_SI_MODE_CHANNEL_SEARCH_NO_NIT, APP_SI_MODE_STARTUP_SEARCH, APP_SI_MODE_EVENT_SCHED_SEARCH, APP_SI_MODE_DVB_SDU_SEARCH, APP_SI_MODE_UPDATE, APP_SI_MODE_CIPLUS_UPDATE, APP_SI_MODE_CIPLUS_NO_PAT_PMT, APP_SI_MODE_TOT_SEARCH, APP_SI_MODE_USER_DEFINED }`
- `enum E_SEARCH_SERVICE_TYPE { SEARCH_SERVICE_TYPE_ALL, SEARCH_SERVICE_TYPE_FTA, SEARCH_SERVICE_TYPE_SCRAMBLED }`
- `enum E_APP_SI_EVENT_CODE { APP_SI_START_MANAGER = 0x0001, APP_SI_STOP_MANAGER = 0x0002, APP_SI_CHANNEL_CHANGE = 0x0003, ... }`

```

APP_SI_PAT_RECEIVED = 0x0011, APP_SI_PMT_RECEIVED = 0x0012, AP-
P_SI_SDT_RECEIVED = 0x0013, APP_SI_NIT_RECEIVED = 0x0014, APP_S-
I_EIT_RECEIVED = 0x0015, APP_SI_TOT_RECEIVED = 0x0016, APP_SI_T-
DT_RECEIVED = 0x0017, APP_SI_CAT_RECEIVED = 0x0018, APP_SI_RCT-
_RECEIVED = 0x0019, APP_SI_AIT_RECEIVED = 0x001A, APP_SI_BAT_R-
ECEIVED = 0x0020, APP_SI_PAT_TIMEOUT = 0x0031, APP_SI_PMT_TIME-
OUT = 0x0032, APP_SI_SDT_TIMEOUT = 0x0033, APP_SI_NIT_TIMEOUT =
0x0034, APP_SI_EIT_TIMEOUT = 0x0035, APP_SI_TOT_TIMEOUT = 0x0036,
APP_SI_TDT_TIMEOUT = 0x0037, APP_SI_CAT_TIMEOUT = 0x0038, APP-
_SI_SCHED_TIMEOUT = 0x0039, APP_SI_BAT_TIMEOUT = 0x0040, APP-
_SI_PMT_UPDATE = 0x0051, APP_SI_UPDATE_DELAY_EXPIRED = 0x0052,
APP_SI_STOP_PMT_REPORTING = 0x0053, APP_SI_USER_DEFINED_EVE-
NT = 0x0100 }

```

Functions

- **BOOLEAN** [ASI_CheckForServiceChange](#) (void)
Checks whether the NIT or SDT version numbers have changed, which may indicate a change to the service lineup requiring an update service scan.
- **void** [ASI_SetSearchServiceType](#) (E_SEARCH_SERVICE_TYPE service_type)
Set the type for services that should be added during a service search.
- **void** [ASI_AddServiceToPmtList](#) (U16BIT service_id)
Add the given service id to the list of services whose PMT will be requested with a higher priority than others.
- **void** [ASI_RemoveServiceFromPmtList](#) (U16BIT service_id)
Removes the service id from the PMT priority list.
- **void** [ASI_ClearPmtList](#) (void)
Clears all service ids from the PMT priority list.
- **void** [ASI_ProcessPmt](#) (U8BIT path, void *s_ptr, U8BIT *pmt_data)
Takes data for a raw PMT for the given service and processes it as if it had been received from the demux, also passing it to anything monitoring PMTs and CI+.
- **void** [ASI_SetEITScheduleLimit](#) (U16BIT limit_hours)
Sets the number of hours of EIT data that's kept for each service that hasn't its had EIT schedule disabled.
- **U16BIT** [ASI_GetEITScheduleLimit](#) (void)
Returns the current setting for the number of hours of EIT data that's kept.
- **void** [ASI_EnableBatCollection](#) (BOOLEAN use_bats, U16BIT *bouquet_ids, - U16BIT num_ids)
Enables or disables the collection of BATs as part of the SI processing and allows the bouquet IDs of the BATs to be collected to be specified.
- **BOOLEAN** [ASI_GetDvbSsuStatus](#) (U16BIT *onid_ptr, U16BIT *tid_ptr, U16BIT *sid_ptr)
Returns whether an SSU over-the-air update is possibly available, together with the DVB triplet of the service where it can be found. The service ids are only valid if an update is available.
- **void** [ASI_RefuseSSU](#) (BOOLEAN refuse)

- Sets the flag indicating whether the SSU has been refused or not.*

 - **BOOLEAN** [ASI_SSURefused](#) (void)
- Returns the flag indicating whether the SSU was refused.*

 - void [ASI_SSUSetMandatory](#) (BOOLEAN mandatory)
- Sets the flag indicating whether the SSU can be refused by the user.*

 - **BOOLEAN** [ASI_SSUGetMandatory](#) (void)
- Returns the flag indicating whether the SSU is mandatory and so can't be refused.*

 - void [ASI_InitialiseAppSi](#) (void)
- Initialises application SI handling.*

 - void [ASI_SetAppSiMode](#) (U8BIT path, E_APP_SI_MODE si_mode)
- Sets application SI mode - used before STB_DPStartSI() is called.*

 - U16BIT [ASI_GetPmtPid](#) (U16BIT serv_id, U16BIT ts_id, U16BIT on_id)
- Returns the PID for the pmt of a given service (on live path)*

 - void [ASI_StopPmtReporting](#) (U8BIT path)
- Prevents the current pmt being reported (e.g. to MHEG).*

 - **BOOLEAN** [ASI_PmtReported](#) (U8BIT path)
- Returns TRUE if pmt has been reported to third parties.*

 - void [ASI_SetStandbyState](#) (BOOLEAN standby_state)
- Performs the neccessary actions for this module when entering/exiting standby according to the value passed in standby_state.*

 - void [ASI_RestartCatFilter](#) (U8BIT path)
- Forces the SI demux filter collecting the CAT tables to be reset, so a previously processed version of the table won't be ignored.*

 - void [ASI_RestartSITables](#) (U8BIT path, BOOLEAN use_standard_pids)
- Cancels any existing SI filters and starts a new one for the SDT, TDT, TOT, NIT, EIT (pf, pf++ and sched) and BAT SI tables.*

 - void [ASI_SetEITParserFunction](#) (F_EitParser parser_func)
- Sets a function that will be called when parsing an EIT table and a descriptor is found that the standard code doesn't know how to parse.*

 - void [ASI_SetUpdateBatFunction](#) (F_BatTableUpdate update_func)
- Sets a function that will be called when a BAT table is received.*

 - void [ASI_SetUpdateEitFunction](#) (F_EitTableUpdate update_func)
- Sets a function that will be called when an EIT table is received.*

 - void [ASI_SetUpdateNitFunction](#) (F_NitTableUpdate update_func)
- Sets a function that will be called when an NIT table is received.*

 - void [ASI_SetUpdatePmtFunction](#) (F_PmtTableUpdate update_func)
- Sets a function that will be called when a PMT table is received.*

 - void [ASI_SetUpdateSdtFunction](#) (F_SdtTableUpdate update_func)
- Sets a function that will be called when an SDT table is received.*

 - void [ASI_ProcessEitTable](#) (SI_TABLE_RECORD *table_rec, BOOLEAN playback)
- Processes an EIT table, partial or full, and updates the events of the service it is for.*

 - void [ASI_ProcessTotTable](#) (SI_TABLE_RECORD *table_rec)
- Processes the TOT table record to extract data for the database.*

 - void [ASI_ProcessTdtTable](#) (SI_TABLE_RECORD *table_rec)
- Processes the TDT table record to extract data for the database.*

4.14.1 Detailed Description

application level SI task

Date

01/10/2002

4.14.2 Function Documentation

4.14.2.1 void ASI_AddServiceToPmtList (U16BIT *service_id*)

Add the given service id to the list of services whose PMT will be requested with a higher priority than others.

Parameters

<i>service_id</i>	service id to be added to the list
-------------------	------------------------------------

4.14.2.2 BOOLEAN ASI_CheckForServiceChange (void)

Checks whether the NIT or SDT version numbers have changed, which may indicate a change to the service lineup requiring an update service scan.

Returns

TRUE if one of the version numbers has changed, FALSE otherwise

4.14.2.3 void ASI_EnableBatCollection (BOOLEAN *use_bats*, U16BIT * *bouquet_ids*, U16BIT *num_ids*)

Enables or disables the collection of BATs as part of the SI processing and allows the bouquet IDs of the BATs to be collected to be specified.

Parameters

<i>use_bats</i>	TRUE to enable the collection and processing of BATs, FALSE to disable
<i>bouquet_ids</i>	an array of the bouquet IDs that are to be used. If this is NULL then all BATs will be used.
<i>num_ids</i>	number of bouquet IDs in the array

4.14.2.4 BOOLEAN ASI_GetDvbSsuStatus (U16BIT * *onid_ptr*, U16BIT * *tid_ptr*, U16BIT * *sid_ptr*)

Returns whether an SSU over-the-air update is possibly available, together with the DVB triplet of the service where it can be found. The service ids are only valid if an update is available.

Parameters

<i>onid_ptr</i>	address to return the original network id
<i>tid_ptr</i>	address to return the transport id
<i>sid_ptr</i>	address to return the service id

Returns

TRUE if an SSU is available, FALSE otherwise

4.14.2.5 U16BIT ASI_GetEITScheduleLimit (void)

Returns the current setting for the number of hours of EIT data that's kept.

Returns

number of hours, 0 means all data is kept

4.14.2.6 U16BIT ASI_GetPmtPid (U16BIT *serv_id*, U16BIT *ts_id*, U16BIT *on_id*)

Returns the PID for the pmt of a given service (on live path)

Parameters

<i>service_id</i>	provides the required service ID
<i>ts_id</i>	provides the required transport stream ID
<i>on_id</i>	provides the required original network ID

Returns

the PID of the pmt for that service

4.14.2.7 BOOLEAN ASI_PmtReported (U8BIT *path*)

Returns TRUE if pmt has been reported to third parties.

Parameters

<i>path</i>	decode path
-------------	-------------

Returns

TRUE if reported, FALSE otherwise

4.14.2.8 void ASI_ProcessEitTable (SI_TABLE_RECORD * *table_rec*, BOOLEAN *playback*)

Processes an EIT table, partial or full, and updates the events of the service it is for.

Parameters

<i>table_rec</i>	SI table record containing the EIT data
<i>playback</i>	TRUE if the EIT is related to PVR playback

4.14.2.9 void ASI_ProcessPmt (U8BIT *path*, void * *s_ptr*, U8BIT * *pmt_data*)

Takes data for a raw PMT for the given service and processes it as if it had been received from the demux, also passing it to anything monitoring PMTs and CI+.

Parameters

<i>path</i>	decode path
<i>s_ptr</i>	service to be updated with info extracted from the PMT
<i>pmt_data</i>	raw PMT data

4.14.2.10 void ASI_ProcessTdtTable (SI_TABLE_RECORD * *table_rec*)

Processes the TDT table record to extract data for the database.

Parameters

<i>table_rec</i>	pointer to the TDT table record
------------------	---------------------------------

4.14.2.11 void ASI_ProcessTotTable (SI_TABLE_RECORD * *table_rec*)

Processes the TOT table record to extract data for the database.

Parameters

<i>table_rec</i>	pointer to the TOT table record
------------------	---------------------------------

4.14.2.12 void ASI_RefuseSSU (BOOLEAN *refuse*)

Sets the flag indicating whether the SSU has been refused or not.

Parameters

<i>refuse</i>	TRUE to refuse the SSU, FALSE to accept
---------------	---

4.14.2.13 void ASI_RemoveServiceFromPmtList (U16BIT *service_id*)

Removes the service id from the PMT priority list.

Parameters

<i>service_id</i>	service id to be removed
-------------------	--------------------------

4.14.2.14 void ASI_RestartCatFilter (U8BIT *path*)

Forces the SI demux filter collecting the CAT tables to be reset, so a previously processed version of the table won't be ignored.

Parameters

<i>path</i>	- decode path that will be affected
-------------	-------------------------------------

4.14.2.15 void ASI_RestartSITables (U8BIT *path*, BOOLEAN *use_standard_pids*)

Cancels any existing SI filters and starts a new one for the SDT, TDT, TOT, NIT, EIT (pf, pf++ and sched) and BAT SI tables.

Parameters

<i>path</i>	decode path the filters are using
<i>use_standard_pids</i>	TRUE if SI tables on the DVB standard PIDs are to be collected, FALSE will only start a filter if a PID is defined for that table on the current service.

4.14.2.16 void ASI_SetAppSiMode (U8BIT *path*, E_APP_SI_MODE *si_mode*)

Sets application SI mode - used before STB_DPStartSI() is called.

Parameters

<i>path</i>	decode path
<i>si_mode</i>	required mode

4.14.2.17 void ASI_SetEITParserFunction (F_EitParser *parser_func*)

Sets a function that will be called when parsing an EIT table and a descriptor is found that the standard code doesn't know how to parse.

Parameters

<i>parser_func</i>	function pointer
--------------------	------------------

4.14.2.18 void ASI_SetEITScheduleLimit (U16BIT *limit_hours*)

Sets the number of hours of EIT data that's kept for each service that hasn't its had EIT schedule disabled.

Parameters

<i>limit_hours</i>	number of hours to EIT data to be kept. 0 turns this feature off and all data will be kept.
--------------------	---

4.14.2.19 void ASI_SetSearchServiceType (E_SEARCH_SERVICE_TYPE *service_type*)

Set the type for services that should be added during a service search.

Parameters

<i>service_type</i>	all, free-to-air, or scrambled
---------------------	--------------------------------

4.14.2.20 void ASI_SetStandbyState (BOOLEAN *standby_state*)

Performs the necessary actions for this module when entering/exiting standby according to the value passed in *standby_state*.

Parameters

<i>standby_state</i>	if set TRUE indicates that the STB is entering standby mode and if set FALSE indicates that the STB is exiting from standby mode.
----------------------	---

4.14.2.21 void ASI_SetUpdateBatFunction (F_BatTableUpdate *update_func*)

Sets a function that will be called when a BAT table is received.

Parameters

<i>update_func</i>	function pointer
--------------------	------------------

4.14.2.22 void ASI_SetUpdateEitFunction (F_EitTableUpdate *update_func*)

Sets a function that will be called when an EIT table is received.

Parameters

<i>update_func</i>	function pointer
--------------------	------------------

4.14.2.23 void ASI_SetUpdateNitFunction (F_NitTableUpdate *update_func*)

Sets a function that will be called when an NIT table is received.

Parameters

<i>update_func</i>	function pointer
--------------------	------------------

4.14.2.24 void ASI_SetUpdatePmtFunction (F_PmtTableUpdate *update_func*)

Sets a function that will be called when a PMT table is received.

Parameters

<i>update_func</i>	function pointer
--------------------	------------------

4.14.2.25 void ASI_SetUpdateSdtFunction (F_SdtTableUpdate *update_func*)

Sets a function that will be called when an SDT table is received.

Parameters

<i>update_func</i>	function pointer
--------------------	------------------

4.14.2.26 BOOLEAN ASI_SSUGetMandatory (void)

Returns the flag indicating whether the SSU is mandatory and so can't be refused.

Returns

TRUE if the SSU is mandatory, FALSE otherwise

4.14.2.27 BOOLEAN ASI_SSURefused (void)

Returns the flag indicating whether the SSU was refused.

Returns

TRUE if the SSU was refused, FALSE otherwise

4.14.2.28 void ASI_SSUSetMandatory (BOOLEAN *mandatory*)

Sets the flag indicating whether the SSU can be refused by the user.

Parameters

<i>mandatory</i>	TRUE if the SSU can't be refused by the user, FALSE otherwise
------------------	---

4.15 dvb/inc/ap_tmr.h File Reference

Application timer functions and defines.

```
#include "stbgc.h" Include dependency graph for ap_tmr.h:
```

Data Structures

- struct [S_ALARM_INFO](#)
- struct [S_PVR_RECORD_INFO](#)
- struct [s_timer](#)
- struct [s_avrec_settings](#)

Defines

- #define **INVALID_TIMER_HANDLE** 0

- #define **TMR_MAX_NAME_LENGTH** 128
- #define **TMR_PVR_CRID_LEN_MAX** 65
- #define **TMR_PVR_ADDINFO_LEN_MAX** 255
- #define **EVENT_DURATION_OVERRUN_TIME** 2

Typedefs

- typedef struct [s_timer](#) **S_TIMER_INFO**
- typedef struct [s_avrec_settings](#) **S_AVREC_SETTINGS**

Enumerations

- enum **E_TIMER_TYPE** { **TIMER_TYPE_NONE** = 0x00, **TIMER_TYPE_ALARM** = 0x01, **TIMER_TYPE_SLEEP** = 0x02, **TIMER_TYPE_PVR_RECORD** = 0x04, **TIMER_TYPE_PRIVATE** = 0x80, **TIMER_TYPE_ALL** = 0xff }
- enum **E_TIMER_FREQ** { **TIMER_FREQ_ONCE**, **TIMER_FREQ_WEEKLY**, **TIMER_FREQ_WEEKENDDAYS**, **TIMER_FREQ_WEEKDAYS**, **TIMER_FREQ_DAILY**, **TIMER_FREQ_HOURLY** }

Functions

- void [ATMR_Initialise](#) (void)
Performs initialisation of the timers, reading existing entries from the database.
- U32BIT [ATMR_AddTimer](#) ([S_TIMER_INFO](#) *info)
Creates a new timer based on the information supplied.
- U32BIT [ATMR_AddTimerForEvent](#) (void *event_ptr, void *serv_ptr, BOOLEAN record, BOOLEAN event_triggered)
Creates a timer based on the given event. If a recording timer is created, it will be set to record on the default disk, which can be changed afterwards using [ATMR_SetDiskId](#).
- BOOLEAN [ATMR_UpdateTimerDuration](#) (U32BIT handle, U32DHMS duration)
Updates the duration for an existing PVR recording timer.
- BOOLEAN [ATMR_DeleteTimer](#) (U32BIT handle)
Deletes the timer with the given handle.
- BOOLEAN [ATMR_GetTimerList](#) (U32BIT **timer_list, U16BIT *list_size, E_TIMER_TYPE list_type, BOOLEAN date_time_order)
Returns a list of all the timer handles and the number of items in the list.
- void [ATMR_ReleaseTimerList](#) (U32BIT *timer_list, U16BIT list_size)
Release the given array of timer handles.
- BOOLEAN [ATMR_InitialiseTimer](#) ([S_TIMER_INFO](#) *timer_info, E_TIMER_TYPE timer_type, void *serv_ptr, void *event_ptr)
Sets up the given timer info structure with default values for the given timer type using the service and event depending on timer type.
- BOOLEAN [ATMR_GetTimerInfo](#) (U32BIT handle, [S_TIMER_INFO](#) *timer_info)
Copies timer data for the given timer info the info structure provided.
- BOOLEAN [ATMR_StartRecord](#) (U8BIT path)

Finds the timer using the given decode path and if it's a recording timer the recording will be started.

- void [ATMR_RecordingFailed](#) (U8BIT path)
Handles the timer when a recording fails to start for some reason. This may result in the timer being deleted.
- void * [ATMR_GetRecordService](#) (U8BIT path)
Returns the service for the recording timer on the given path.
- U8BIT * [ATMR_GetName](#) (U32BIT handle)
Get the name of the timer with the given handle.
- void [ATMR_SetName](#) (U32BIT handle, U8BIT *name)
Sets the name of the timer with the given handle.
- U32DHMS [ATMR_GetStartDateTime](#) (U32BIT handle)
Get the start date & time of the timer with the given handle. The date/time returned will be in UTC.
- U32DHMS [ATMR_GetDuration](#) (U32BIT handle)
Returns the duration of the timer with the given handle.
- U32DHMS [ATMR_GetEndDateTime](#) (U32BIT handle)
Get the end date & time of the timer with the given handle. The date/time returned will be in UTC.
- E_TIMER_TYPE [ATMR_GetType](#) (U32BIT handle)
Returns the type of the given timer.
- E_TIMER_FREQ [ATMR_GetFrequency](#) (U32BIT handle)
Returns the frequency setting for the given timer.
- BOOLEAN [ATMR_GetChangeService](#) (U32BIT handle)
Returns the change service setting for an alarm timer.
- BOOLEAN [ATMR_GetRampVolume](#) (U32BIT handle)
Returns the ramp volume setting for an alarm timer.
- U16BIT [ATMR_GetEventId](#) (U32BIT handle)
Returns the event ID for a PVR recording timer.
- U16BIT [ATMR_GetServiceId](#) (U32BIT handle)
Returns the service ID for an alarm or PVR recording timer.
- U16BIT [ATMR_GetTransportId](#) (U32BIT handle)
Returns the transport ID for an alarm or PVR recording timer.
- U16BIT [ATMR_GetOriginalNetworkId](#) (U32BIT handle)
Returns the original network ID for an alarm or PVR recording timer.
- BOOLEAN [ATMR_GetMissed](#) (U32BIT handle)
Gets the timer's missed flag.
- U8BIT * [ATMR_GetProgrammeCrid](#) (U32BIT handle)
Returns a pointer to the programme CRID string from a recording timer. The returned value shouldn't be changed or freed.
- BOOLEAN [ATMR_HasSeriesCrid](#) (U32BIT handle)
Does the timer have a series crid?
- BOOLEAN [ATMR_HasRecommendationCrid](#) (U32BIT handle)
Does the timer have a recommendation crid?

- U16BIT [ATMR_GetDiskId](#) (U32BIT handle)
Returns the disk id for the given timer if the timer is a PVR recording timer.
- void [ATMR_SetDiskId](#) (U32BIT handle, U16BIT disk_id)
Set the disk for the given timer if the timer is a recording timer.
- U32BIT [ATMR_GetRecordingHandle](#) (U32BIT handle)
Returns the recording handle associated with a PVR recording timer.
- void [ATMR_HandleTimerEvent](#) (U32BIT timer_handle)
Used by the DVB stack to handle an event for the given timer. If the timer requires the app to deal with it, such as for sleep or alarm timers, then another event will be sent to the app with an S_TIMER_INFO structure as the event data containing the timer's details to allow the app deal with it. The timer itself will have been updated or deleted.
- BOOLEAN [ATMR_CheckRecordStatus](#) (BOOLEAN recordings_can_start)
Checks all timers to see whether any recordings should be started or stopped as a result of the now event being changed.
- void [ATMR_EitUpdated](#) (void)
Checks each recording timer that's linked to an event to see whether the event is still in the schedule and whether the time has changed, and if it has changed then the timer is updated.
- void [ATMR_DeleteRecordingTimer](#) (U32BIT recording_handle)
Delete the PVR record timer with the given recording handle.
- U8BIT [ATMR_GetNumSimultaneousRecordings](#) (U8BIT max_recordings, U32DHMS start_date_time, U32DHMS end_date_time, U32BIT **conflicting_timers)
Counts the number of simultaneous recordings (EXT and PVR) between the given start and end dates/times.
- void * [ATMR_RecordEvent](#) (void *serv_ptr, void *event_ptr, U8BIT *prog_crid, U8BIT *series_crid, BOOLEAN is_series, BOOLEAN check_alternatives)
Adds a timer to perform a recording based on the given event and service. Conflicts are checked and a search is made for an alternative event if necessary.
- BOOLEAN [ATMR_RecordSplitEvent](#) (void *serv_ptr, U8BIT *prog_crid, U32DHMS start_date_time, BOOLEAN radio_service, BOOLEAN search_forward)
Searches for events within 3 hours of the given start date/time for an event with the given programme CRID and creates a timer to record any found. The search is performed forward or backward from the given start date/time as defined by the search direction argument.
- U32BIT [ATMR_GetFirstWakeupTime](#) (U32DHMS *rec_date_time, U16BIT *onet_id, U16BIT *trans_id, U16BIT *service_id)
Searches the timers for the first timer that will cause the DVB to wakeup that hasn't been missed. This will be an alarm or PVR recording timer.
- U32BIT [ATMR_FindTimerFromEvent](#) (U16BIT onet_id, U16BIT trans_id, U16BIT serv_id, U16BIT event_id)
Searches the timers for a recording timer with the given event and service IDs.
- U32BIT [ATMR_FindTimerFromCrid](#) (U8BIT *prog_crid)
Searches the timers for a recording timer with the given programme CRID.
- U32BIT [ATMR_FindTimerFromCridAndEvent](#) (U8BIT *prog_crid, U16BIT service_id, U16BIT event_id)

Searches the timers for a recording timer with the given programme CRID and event ID.

- void [ATMR_DeleteTimersForSeriesRecommendations](#) (U8BIT *crid, BOOLEAN is_recommendation)

Deletes any PVR record timers with the given series CRID.

- void [ATMR_SetAdditionalInfo](#) (U32BIT handle, U8BIT *info, U32BIT size)

Sets the additional information string for the specified timer and commits the change to the database.

- U8BIT * [ATMR_GetAdditionalInfo](#) (U32BIT handle, U32BIT *size)

Gets the additional information string associated with a timer. The name is allocated in UI temp memory.

- S32BIT [ATMR_GetStartPadding](#) (U32BIT handle)

Returns the value of start_padding associated with the specified timer The start padding is the number of seconds before (after, if negative) the actual timer start time the timer will be triggered.

- BOOLEAN [ATMR_SetStartPadding](#) (U32BIT handle, S32BIT start_padding)

Sets the value of start_padding associated with the specified timer The start padding is the number of seconds before (after, if negative) the actual timer start time the timer will be triggered.

- S32BIT [ATMR_GetEndPadding](#) (U32BIT handle)

Returns the value of end_padding associated with the specified timer The end padding is the number of seconds after (before, if negative) the actual timer end time the timer will expire.

- BOOLEAN [ATMR_SetEndPadding](#) (U32BIT handle, S32BIT end_padding)

Sets the value of end_padding associated with the specified timer The end padding is the number of seconds after (before, if negative) the actual timer end time the timer will expire.

- void [ATMR_DumpAllTimers](#) (void)

4.15.1 Detailed Description

Application timer functions and defines.

Date

27/5/2004

Author

Ocean Blue

4.15.2 Function Documentation

4.15.2.1 U32BIT ATMR_AddTimer (S_TIMER_INFO * info)

Creates a new timer based on the information supplied.

Parameters

<i>info</i>	timer info used to create the timer
-------------	-------------------------------------

Returns

timer handle, or INVALID_TIMER_HANDLE if creation fails

4.15.2.2 U32BIT ATMR_AddTimerForEvent (void * *event_ptr*, void * *serv_ptr*, BOOLEAN *record*, BOOLEAN *event_triggered*)

Creates a timer based on the given event. If a recording timer is created, it will be set to record on the default disk, which can be changed afterwards using ATMR_SetDiskId.

Parameters

<i>event_ptr</i>	- pointer to the event to be recorded
<i>serv_ptr</i>	service the event is on
<i>record</i>	TRUE if a recording timer is to be created, FALSE for an alarm timer
<i>event_triggered</i>	TRUE if the timer should be triggered by the change of event rather than the time

Returns

timer handle, or INVALID_TIMER_HANDLE if the timer couldn't be created

4.15.2.3 BOOLEAN ATMR_CheckRecordStatus (BOOLEAN *recordings_can_start*)

Checks all timers to see whether any recordings should be started or stopped as a result of the now event being changed.

Parameters

<i>recordings_can_start</i>	TRUE if the event should be checked to start a recording
-----------------------------	--

Returns

TRUE if a record can be/has been started

4.15.2.4 void ATMR_DeleteRecordingTimer (U32BIT *recording_handle*)

Delete the PVR record timer with the given recording handle.

Parameters

<i>recording_handle</i>	recording handle used by the timer to be deleted
-------------------------	--

4.15.2.5 **BOOLEAN ATMR_DeleteTimer (U32BIT *handle*)**

Deletes the timer with the given handle.

Parameters

<i>handle</i>	handle of timer to be deleted
---------------	-------------------------------

Returns

TRUE if the timer exists and is deleted, FALSE otherwise

4.15.2.6 **void ATMR_DeleteTimersForSeriesRecommendations (U8BIT * *crid*,
BOOLEAN *is_recommendation*)**

Deletes any PVR record timers with the given series CRID.

Parameters

<i>crid</i>	the series/recommendation CRID to be searched for
<i>is_recommendation</i>	TRUE if the given crid is for a recommendation, FALSE otherwise

4.15.2.7 **U32BIT ATMR_FindTimerFromCrid (U8BIT * *prog_crid*)**

Searches the timers for a recording timer with the given programme CRID.

Parameters

<i>prog_crid</i>	the programme CRID to be searched for
------------------	---------------------------------------

Returns

handle of timer, or INVALID_TIMER_HANDLE if no timer is found

4.15.2.8 **U32BIT ATMR_FindTimerFromCridAndEvent (U8BIT * *prog_crid*, U16BIT
service_id, U16BIT *event_id*)**

Searches the timers for a recording timer with the given programme CRID and event ID.

Parameters

<i>prog_crid</i>	the programme CRID to search for
<i>service_id</i>	service ID of the event
<i>event_id</i>	ID of the event

Returns

handle of timer, or INVALID_TIMER_HANDLE if no timer is found

4.15.2.9 U32BIT ATMR_FindTimerFromEvent (U16BIT *onet_id*, U16BIT *trans_id*, U16BIT *serv_id*, U16BIT *event_id*)

Searches the timers for a recording timer with the given event and service IDs.

Parameters

<i>onet_id</i>	original network ID or the service
<i>trans_id</i>	transport ID of the service
<i>serv_id</i>	service ID
<i>event_id</i>	event ID

Returns

handle of timer, or INVALID_TIMER_HANDLE if no timer is found

4.15.2.10 U8BIT* ATMR_GetAdditionalInfo (U32BIT *handle*, U32BIT * *size*)

Gets the additional information string associated with a timer. The name is allocated in UI temp memory.

Parameters

<i>handle</i>	Timer handle
<i>size</i>	Pointer to variable containing the number of bytes in the string

Returns

additional information string, or NULL

4.15.2.11 BOOLEAN ATMR_GetChangeService (U32BIT *handle*)

Returns the change service setting for an alarm timer.

Parameters

<i>handle</i>	timer handle
---------------	--------------

Returns

value of change service setting, or FALSE if timer is invalid or isn't an alarm timer

4.15.2.12 U16BIT ATMR_GetDiskId (U32BIT *handle*)

Returns the disk id for the given timer if the timer is a PVR recording timer.

Parameters

<i>handle</i>	timer handle
---------------	--------------

Returns

disk id if timer is valid and is a recording timer, INVALID_DISK_ID otherwise

4.15.2.13 U32DHMS ATMR_GetDuration (U32BIT *handle*)

Returns the duration of the timer with the given handle.

Parameters

<i>handle</i>	timer handle
---------------	--------------

Returns

timer duration, or 0 if the timer handle isn't valid

4.15.2.14 U32DHMS ATMR_GetEndTime (U32BIT *handle*)

Get the end date & time of the timer with the given handle. The date/time returned will be in UTC.

Parameters

<i>handle</i>	timer handle
---------------	--------------

Returns

UTC end date/time, or 0 if the timer handle isn't valid

4.15.2.15 S32BIT ATMR_GetEndPadding (U32BIT *handle*)

Returns the value of end_padding associated with the specified timer. The end padding is the number of seconds after (before, if negative) the actual timer end time the timer will expire.

Parameters

<i>handle</i>	Timer handle
---------------	--------------

Returns

start_padding

4.15.2.16 U16BIT ATMR_GetEventId (U32BIT *handle*)

Returns the event ID for a PVR recording timer.

Parameters

<i>handle</i>	timer handle
---------------	--------------

Returns

event ID, or 0 if the timer isn't valid or isn't a recording timer

4.15.2.17 U32BIT ATMR_GetFirstWakeupTime (U32DHMS * *rec_date_time*, U16BIT * *onet_id*, U16BIT * *trans_id*, U16BIT * *service_id*)

Searches the timers for the first timer that will cause the DVB to wakeup that hasn't been missed. This will be an alarm or PVR recording timer.

Parameters

<i>rec_date_time</i>	pointer to return the date/time of the first timer found
<i>onet_id</i>	returns the original network ID of the service associated with the timer
<i>trans_id</i>	returns the transport ID of the service associated with the timer
<i>service_id</i>	returns the service ID of the service associated with the timer

Returns

timer handle, or INVALID_TIMER_HANDLE if no timer is found

4.15.2.18 E_TIMER_FREQ ATMR_GetFrequency (U32BIT *handle*)

Returns the frequency setting for the given timer.

Parameters

<i>handle</i>	timer handle
---------------	--------------

Returns

frequency setting of the timer, or TIMER_FREQ_ONCE if handle isn't valid

4.15.2.19 BOOLEAN ATMR_GetMissed (U32BIT *handle*)

Gets the timer's missed flag.

Parameters

<i>handle</i>	timer handle
---------------	--------------

Returns

TRUE if the timer is marked as missed, FALSE otherwise

4.15.2.20 U8BIT* ATMR_GetName (U32BIT *handle*)

Get the name of the timer with the given handle.

Parameters

<i>handle</i>	timer handle
---------------	--------------

Returns

pointer to timer name, this value shouldn't be freed

4.15.2.21 U8BIT ATMR_GetNumSimultaneousRecordings (U8BIT *max_recordings*, U32DHMS *start_date_time*, U32DHMS *end_date_time*, U32BIT ** *conflicting_timers*)

Counts the number of simultaneous recordings (EXT and PVR) between the given start and end dates/times.

Parameters

<i>max_recordings</i>	the max recordings that can take place at the same time
<i>start_date_time</i>	start date and time of the period to be checked
<i>end_date_time</i>	end date and time of the period to be checked
<i>conflicting_timers</i>	pointer to the list of conflicting timers, the list must to be freed by the caller. If NULL it's ignored.

Returns

Number of simultaneous recordings during the given period

4.15.2.22 U16BIT ATMR_GetOriginalNetworkId (U32BIT *handle*)

Returns the original network ID for an alarm or PVR recording timer.

Parameters

<i>handle</i>	timer handle
---------------	--------------

Returns

original network ID, or ADB_INVALID_DVB_ID if the timer isn't valid or isn't an alarm or recording timer

4.15.2.23 U8BIT* ATMR_GetProgrammeCrid (U32BIT *handle*)

Returns a pointer to the programme CRID string from a recording timer. The returned value shouldn't be changed or freed.

Parameters

<i>handle</i>	timer handle
---------------	--------------

Returns

CRID string, NULL if there isn't a CRID or the timer isn't a recording timer

4.15.2.24 **BOOLEAN ATMR_GetRampVolume (U32BIT *handle*)**

Returns the ramp volume setting for an alarm timer.

Parameters

<i>handle</i>	timer handle
---------------	--------------

Returns

value of ramp volume setting, or FALSE if timer is invalid or isn't an alarm timer

4.15.2.25 **U32BIT ATMR_GetRecordingHandle (U32BIT *handle*)**

Returns the recording handle associated with a PVR recording timer.

Parameters

<i>handle</i>	timer handle
---------------	--------------

Returns

recording handle, or 0 if the timer isn't a PVR recording timer or no recording has been started

4.15.2.26 **void* ATMR_GetRecordService (U8BIT *path*)**

Returns the service for the recording timer on the given path.

Parameters

<i>path</i>	recording path
-------------	----------------

Returns

service pointer, or NULL if timer not found or service referenced by timer not found

4.15.2.27 **U16BIT ATMR_GetServiceId (U32BIT *handle*)**

Returns the service ID for an alarm or PVR recording timer.

Parameters

<i>handle</i>	timer handle
---------------	--------------

Returns

service ID, or ADB_INVALID_DVB_ID if the timer isn't valid or isn't an alarm or recording timer

4.15.2.28 U32DHMS ATMR_GetStartDateTime (U32BIT *handle*)

Get the start date & time of the timer with the given handle. The date/time returned will be in UTC.

Parameters

<i>handle</i>	timer handle
---------------	--------------

Returns

UTC start date/time, or 0 if the timer handle isn't valid

4.15.2.29 S32BIT ATMR_GetStartPadding (U32BIT *handle*)

Returns the value of start_padding associated with the specified timer. The start padding is the number of seconds before (after, if negative) the actual timer start time the timer will be triggered.

Parameters

<i>handle</i>	Timer handle
---------------	--------------

Returns

start_padding

4.15.2.30 BOOLEAN ATMR_GetTimerInfo (U32BIT *handle*, S_TIMER_INFO * *timer_info*)

Copies timer data for the given timer into the info structure provided.

Parameters

<i>handle</i>	timer handle the data is to be copied from
<i>timer_info</i>	will be filled with info from the timer

Returns

TRUE if the timer is valid and data is returned, FALSE otherwise

4.15.2.31 **BOOLEAN ATMR_GetTimerList** (U32BIT ** *timer_list*, U16BIT * *list_size*, E_TIMER_TYPE *list_type*, BOOLEAN *date_time_order*)

Returns a list of all the timer handles and the number of items in the list.

Parameters

<i>timer_list</i>	pointer to an array of timer handles, the array is allocated by this function
<i>list_size</i>	pointer to the number of items in the returned list
<i>list_type</i>	type of timers to put in the list
<i>date_time_order</i>	TRUE if the list is to be returned in date/time order, FALSE returns the list in alphabetical name order

Returns

TRUE if a list is returned, FALSE otherwise

4.15.2.32 **U16BIT ATMR_GetTransportId** (U32BIT *handle*)

Returns the transport ID for an alarm or PVR recording timer.

Parameters

<i>handle</i>	timer handle
---------------	--------------

Returns

transport ID, or ADB_INVALID_DVB_ID if the timer isn't valid or isn't an alarm or recording timer

4.15.2.33 **E_TIMER_TYPE ATMR_GetType** (U32BIT *handle*)

Returns the type of the given timer.

Parameters

<i>handle</i>	timer handle
---------------	--------------

Returns

type of the timer, or TIMER_TYPE_NONE if handle isn't valid

4.15.2.34 **void ATMR_HandleTimerEvent** (U32BIT *timer_handle*)

Used by the DVB stack to handle an event for the given timer. If the timer requires the app to deal with it, such as for sleep or alarm timers, then another event will be sent to the app with an S_TIMER_INFO structure as the event data containing the timer's details to allow the app deal with it. The timer itself will have been updated or deleted.

Parameters

<i>handle</i>	timer handle
---------------	--------------

4.15.2.35 **BOOLEAN ATMR_HasRecommendationCrid (U32BIT *handle*)**

Does the timer have a recommendation crid?

Parameters

<i>handle</i>	timer handle
---------------	--------------

Returns

TRUE if the timer is valid and is a recording timer and has a recommendation CRID,
FALSE otherwise

4.15.2.36 **BOOLEAN ATMR_HasSeriesCrid (U32BIT *handle*)**

Does the timer have a series crid?

Parameters

<i>handle</i>	timer handle
---------------	--------------

Returns

TRUE if the timer is valid and is a recording timer and has a series CRID, FALSE
otherwise

4.15.2.37 **BOOLEAN ATMR_InitialiseTimer (S_TIMER_INFO * *timer_info*,
E_TIMER.TYPE *timer_type*, void * *serv_ptr*, void * *event_ptr*)**

Sets up the given timer info structure with default values for the given timer type using the service and event depending on timer type.

Parameters

<i>timer_info</i>	pointer to timer info structure to be initialised
<i>timer_type</i>	type of timer to initialise the structure for
<i>serv_ptr</i>	service that will be used for alarm and PVR recording timers. It can be NULL, if not, it is used to initialise the service IDs in the alarm and PVR timers and to initialise the programme CRID field in PVR timers.
<i>event_ptr</i>	event that will be used for alarm and PVR recording timers. It can be NULL, if not, it is used to initialise start time and name in alarm and PVR timers and other event related fields in PVR timers. When <i>event_ptr</i> is specified for a PVR timer, it will be initialised as 'event triggered';

Returns

TRUE if the structure was setup, FALSE otherwise

4.15.2.38 void* **ATMR_RecordEvent** (void * *serv_ptr*, void * *event_ptr*, U8BIT * *prog_crid*, U8BIT * *series_crid*, BOOLEAN *is_series*, BOOLEAN *check_alternatives*)

Adds a timer to perform a recording based on the given event and service. Conflicts are checked and a search is made for an alternative event if necessary.

Parameters

<i>serv_ptr</i>	pointer to service on which recording is to take place
<i>event_ptr</i>	event to be recorded
<i>prog_crid</i>	programme CRID or NULL
<i>other_crid</i>	series or recommendation CRID, or NULL
<i>is_recommendation</i>	TRUE if the above crid is a recommendation crid, FALSE for series
<i>check_alternatives</i>	TRUE if alternative events are to be searched for if the specified event can't be recorded

Returns

pointer to the event actually recorded, which may be different from the one given, or NULL if the event can't be recorded

4.15.2.39 void **ATMR_RecordingFailed** (U8BIT *path*)

Handles the timer when a recording fails to start for some reason. This may result in the timer being deleted.

Parameters

<i>path</i>	- recording path
-------------	------------------

4.15.2.40 BOOLEAN **ATMR_RecordSplitEvent** (void * *serv_ptr*, U8BIT * *prog_crid*, U32DHMS *start_date_time*, BOOLEAN *radio_service*, BOOLEAN *search_forward*)

Searches for events within 3 hours of the given start date/time for an event with the given programme CRID and creates a timer to record any found. The search is performed forward or backward from the given start date/time as defined by the search direction argument.

Parameters

<i>serv_ptr</i>	pointer to service on which search is to take place
<i>prog_crid</i>	programme CRID to be searched for
<i>start_date_time</i>	date and time to start searching from

<i>radio_ - service</i>	TRUE if the search is on a radio service
<i>search_ - forward</i>	TRUE if the search is to be made for events following the given date/-time

Returns

TRUE if an event is found and is set for recording, FALSE otherwise

4.15.2.41 void ATMR_ReleaseTimerList (U32BIT * *timer_list*, U16BIT *list_size*)

Release the given array of timer handles.

Parameters

<i>timer_list</i>	- timer array to be freed
<i>list_size</i>	- number of items in the list

4.15.2.42 void ATMR_SetAdditionalInfo (U32BIT *handle*, U8BIT * *info*, U32BIT *size*)

Sets the additional information string for the specified timer and commits the change to the database.

Parameters

<i>handle</i>	Timer handle
<i>info</i>	pointer to the string
<i>size</i>	number of bytes in the string

4.15.2.43 void ATMR_SetDiskId (U32BIT *handle*, U16BIT *disk_id*)

Set the disk for the given timer if the timer is a recording timer.

Parameters

<i>handle</i>	timer handle
<i>disk_id</i>	disk id

4.15.2.44 BOOLEAN ATMR_SetEndPadding (U32BIT *handle*, S32BIT *end_padding*)

Sets the value of *end_padding* associated with the specified timer. The end padding is the number of seconds after (before, if negative) the actual timer end time the timer will expire.

Parameters

<i>handle</i>	Timer handle
<i>end_ - padding</i>	new value for the timer <i>end_padding</i>

Returns

TRUE if the start_padding could be changed, FALSE otherwise (e.g. due to a conflict)

4.15.2.45 void ATMR_SetName (U32BIT *handle*, U8BIT * *name*)

Sets the name of the timer with the given handle.

Parameters

<i>handle</i>	timer handle
---------------	--------------

4.15.2.46 BOOLEAN ATMR_SetStartPadding (U32BIT *handle*, S32BIT *start_padding*)

Sets the value of start_padding associated with the specified timer. The start padding is the number of seconds before (after, if negative) the actual timer start time the timer will be triggered.

Parameters

<i>handle</i>	Timer handle
<i>start_padding</i>	new value for the timer start_padding

Returns

TRUE if the start_padding could be changed, FALSE otherwise (e.g. due to a conflict)

4.15.2.47 BOOLEAN ATMR_StartRecord (U8BIT *path*)

Finds the timer using the given decode path and if it's a recording timer the recording will be started.

Parameters

<i>path</i>	decode path being used for the recording
-------------	--

Returns

TRUE if the recording is started, FALSE otherwise

4.15.2.48 BOOLEAN ATMR_UpdateTimerDuration (U32BIT *handle*, U32DHMS *duration*)

Updates the duration for an existing PVR recording timer.

Parameters

<i>handle</i>	handle of timer to be updated
<i>duration</i>	duration to be set in the timer

Returns

TRUE if the timer is updated, FALSE otherwise

4.16 dvb/inc/app.h File Reference

Application header file.

This graph shows which files directly or indirectly include this file:

Data Structures

- struct [S_ACB_SUBTITLE_PREF](#)
- struct [S_ACB_AD_PREF](#)
- struct [S_ACB_UI_INFO](#)

Defines

- #define **EV_SERVICE_NOT_RUNNING** 0x0001
- #define **EV_SERVICE_AUDIO_PID_UPDATE** 0x0002
- #define **EV_SERVICE_VIDEO_PID_UPDATE** 0x0003
- #define **EV_SERVICE_SUBTITLE_UPDATE** 0x0004
- #define **EV_SERVICE_SCRAMBLE_CHANGE** 0x0005
- #define **EV_SERVICE_ANALOG_LOCKED** 0x0006
- #define **EV_INDICATION_STATUS** 0x0007
- #define **EV_SERVICE_ANALOG_STARTED** 0x0008
- #define **EV_SERVICE_RUNNING** 0x0009
- #define **EV_SERVICE_EIT_NOW_UPDATE** 0x000a
- #define **EV_SERVICE_EIT_SCHED_UPDATE** 0x000b
- #define **EV_SERVICE_ECM_PID_UPDATE** 0x000c
- #define **EV_PVR_RECORDING_FAILED** 0x000d
- #define **EV_SERVICE_VIDEO_CODEC_CHANGED** 0x000e
- #define **EV_SERVICE_AUDIO_CODEC_CHANGED** 0x000f
- #define **EV_SERVICE_CHANGED** 0x0010
- #define **EV_SERVICE_STREAMS_CHANGED** 0x0011
- #define **EV_SERVICE_SI_PID_UPDATE** 0x0012
- #define **EV_TIME_CHANGED** 0x0013
- #define **EV_CIPLUS_TUNE_COMPLETED** 0x0014
- #define **EV_SERVICE_DELETED** 0x0015
- #define **EV_TIMER_TRIGGERED** 0x0016
- #define **EV_DECODE_PAUSED** 0x0017
- #define **EV_DECODE_RESUMED** 0x0018
- #define **EV_SWITCH_ALTERNATIVE_SERVICE** 0x0019
- #define **APP_EVENT_SERVICE_NOT_RUNNING** EVENT_CODE(EV_CLASS_APPLICATION, EV_SERVICE_NOT_RUNNING)
- #define **APP_EVENT_SERVICE_AUDIO_PID_UPDATE** EVENT_CODE(EV_CLASS_APPLICATION, EV_SERVICE_AUDIO_PID_UPDATE)

- `#define APP_EVENT_SERVICE_VIDEO_PID_UPDATE EVENT_CODE(EV_CLASS_APPLICATION, EV_SERVICE_VIDEO_PID_UPDATE)`
- `#define APP_EVENT_SERVICE_SUBTITLE_UPDATE EVENT_CODE(EV_CLASS_APPLICATION, EV_SERVICE_SUBTITLE_UPDATE)`
- `#define APP_EVENT_SERVICE_SCRAMBLE_CHANGE EVENT_CODE(EV_CLASS_APPLICATION, EV_SERVICE_SCRAMBLE_CHANGE)`
- `#define APP_EVENT_SERVICE_ANALOG_LOCKED EVENT_CODE(EV_CLASS_APPLICATION, EV_SERVICE_ANALOG_LOCKED)`
- `#define APP_EVENT_SERVICE_ANALOG_STARTED EVENT_CODE(EV_CLASS_APPLICATION, EV_SERVICE_ANALOG_STARTED)`
- `#define APP_EVENT_INDICATION_STATUS EVENT_CODE(EV_CLASS_APPLICATION, EV_INDICATION_STATUS)`
- `#define APP_EVENT_SERVICE_RUNNING EVENT_CODE(EV_CLASS_APPLICATION, EV_SERVICE_RUNNING)`
- `#define APP_EVENT_SERVICE_EIT_NOW_UPDATE EVENT_CODE(EV_CLASS_APPLICATION, EV_SERVICE_EIT_NOW_UPDATE)`
- `#define APP_EVENT_SERVICE_EIT_SCHED_UPDATE EVENT_CODE(EV_CLASS_APPLICATION, EV_SERVICE_EIT_SCHED_UPDATE)`
- `#define APP_EVENT_SERVICE_ECM_PID_UPDATE EVENT_CODE(EV_CLASS_APPLICATION, EV_SERVICE_ECM_PID_UPDATE)`
- `#define APP_EVENT_PVR_RECORDING_FAILED EVENT_CODE(EV_CLASS_APPLICATION, EV_PVR_RECORDING_FAILED)`
- `#define APP_EVENT_SERVICE_VIDEO_CODEC_CHANGED EVENT_CODE(EV_CLASS_APPLICATION, EV_SERVICE_VIDEO_CODEC_CHANGED)`
- `#define APP_EVENT_SERVICE_AUDIO_CODEC_CHANGED EVENT_CODE(EV_CLASS_APPLICATION, EV_SERVICE_AUDIO_CODEC_CHANGED)`
- `#define APP_EVENT_SERVICE_CHANGED EVENT_CODE(EV_CLASS_APPLICATION, EV_SERVICE_CHANGED)`
- `#define APP_EVENT_SERVICE_STREAMS_CHANGED EVENT_CODE(EV_CLASS_APPLICATION, EV_SERVICE_STREAMS_CHANGED)`
- `#define APP_EVENT_SERVICE_SI_PID_UPDATE EVENT_CODE(EV_CLASS_APPLICATION, EV_SERVICE_SI_PID_UPDATE)`
- `#define APP_EVENT_TIME_CHANGED EVENT_CODE(EV_CLASS_APPLICATION, EV_TIME_CHANGED)`
- `#define APP_EVENT_CPLUS_TUNE_COMPLETED EVENT_CODE(EV_CLASS_APPLICATION, EV_CPLUS_TUNE_COMPLETED)`
- `#define APP_EVENT_SERVICE_DELETED EVENT_CODE(EV_CLASS_APPLICATION, EV_SERVICE_DELETED)`
- `#define APP_EVENT_TIMER_TRIGGERED EVENT_CODE(EV_CLASS_APPLICATION, EV_TIMER_TRIGGERED)`
- `#define APP_EVENT_DECODE_PAUSED EVENT_CODE(EV_CLASS_APPLICATION, EV_DECODE_PAUSED)`
- `#define APP_EVENT_DECODE_RESUMED EVENT_CODE(EV_CLASS_APPLICATION, EV_DECODE_RESUMED)`
- `#define APP_EVENT_SWITCH_ALTERNATIVE_SERVICE EVENT_CODE(EV_CLASS_APPLICATION, EV_SWITCH_ALTERNATIVE_SERVICE)`
- `#define DISK_NAME_LEN 40 /* Max length for the name of a disk */`

Typedefs

- typedef BOOLEAN(* [ACB_INFO_CALLBACK](#))(S_ACB_UI_INFO *info)
Callback function implemented in the UI. This function is called by the DVB module to query values defined in the UI (e.g. user preferences)
- typedef void(* [DVB_EVENT_HANDLER](#))(U32BIT event, void *event_data, U32BIT data_size)
The event notification callback function.

Enumerations

- enum **E_ACB_INFO_TYPE** { **ACB_GET_SUBTITLE_PREF**, **ACB_GET_AD_PREF**, **ACB_GET_UI_LANG_PREF**, **ACB_GET_UI_BANNER_TRANSPARENCY**, **ACB_GET_UI_BANNER_TIMEOUT**, **ACB_NUM_INFO_TYPES** }

Functions

- BOOLEAN [APP_InitialiseDVB](#) ([DVB_EVENT_HANDLER](#) event_handler)
Main function to initialise the DVB.
- BOOLEAN [APP_RegisterDVBEventHandler](#) ([DVB_EVENT_HANDLER](#) event_handler)
Register for DVB event notifications.
- BOOLEAN [APP_UnregisterDVBEventHandler](#) ([DVB_EVENT_HANDLER](#) event_handler)
Unregister a previously registered event handler.
- U8BIT * **App_GetVersionString** (void)
- U32BIT **App_GetVersionNumber** (void)
- void **APP_RegisterUIInfoCallback** ([ACB_INFO_CALLBACK](#) ui_info_callback)

4.16.1 Detailed Description

Application header file. *****

Date

01/12/2004

Author

Ocean Blue

4.16.2 Typedef Documentation

4.16.2.1 typedef BOOLEAN(* ACB_INFO_CALLBACK)(S_ACB_UI_INFO *info)

Callback function implemented in the UI. This function is called by the DVB module to query values defined in the UI (e.g. user preferences)

Parameters

S_ACB_UI_INFO	(IN/OUT) pointer to the structure that contains the type of request and other fields to exchange information between the DVB and the UI. - These fields can be input or output values depending on the query type.
-------------------------------	--

Returns

TRUE if the information is returned, FALSE otherwise

4.16.2.2 typedef void(* DVB_EVENT_HANDLER)(U32BIT event, void *event_data, U32BIT data_size)

The event notification callback function.

Each registered function is called by the DVBCore to notify about DVB events.

Parameters

<i>event</i>	Event ID, as defined in stberc.h
<i>event_data</i>	Optional, event-specific data
<i>data_size</i>	size of the event_data

4.16.3 Function Documentation

4.16.3.1 BOOLEAN APP_InitialiseDVB (DVB_EVENT_HANDLER event_handler)

Main function to initialise the DVB.

Parameters

<i>event_handler</i>	Callback function used to pass events to the UI. If non-NULL value is passed-in the APP_RegisterDVBEventHandler() is called in order to register the 1st event handler.
----------------------	---

Returns

TRUE if initialisation succeeds, FALSE otherwise.

See also

[APP_RegisterDVBEventHandler](#)

4.16.3.2 **BOOLEAN APP_RegisterDVBEventHandler (DVB_EVENT_HANDLER event_handler)**

Register for DVB event notifications.

Multiple clients can register their own event handlers. Each new handler is added at the end of the list. For each event all registered handlers all called.

Parameters

<i>event_ handler</i>	A call-back function to receive event notifications.
---------------------------	--

Returns

TRUE on success, FALSE otherwise.

See also

[APP_RegisterDVBEventHandler](#)

4.16.3.3 **BOOLEAN APP_UnregisterDVBEventHandler (DVB_EVENT_HANDLER event_handler)**

Unregister a previously registered event handler.

Find and remove the function from the registered handlers list. If the same function was registered more than once, the 1st found occurrence will be removed.

Parameters

<i>event_ handler</i>	A function pointer previously passed into the APP_RegisterDVBEvent-Handler function.
---------------------------	--

Returns

TRUE on success, FALSE otherwise.

See also

[APP_RegisterDVBEventHandler](#)

4.17 dvb/inc/app_nvm.h File Reference

Header file for NVM data handling functions.

```
#include "stbhwc.h" Include dependency graph for app_nvm.h:
```

Typedefs

- typedef enum e_nvm_items **E_NVM_ITEMS**

Enumerations

- enum e_nvm_items { FIRST_BOOT_NVM, STANDBY_POWERSAVE_NVM, VOLUME_NVM, ASPECT_MODE_NVM, SCART_INPUT_TYPE_NVM, PRIMARY_AUDIO_LANG_NVM, SECONDARY_AUDIO_LANG_NVM, AUDIO_TYPE_NVM, STANDBY_STATE_NVM, SEARCHING_STATE_NVM, PARENTAL_LOCK_NVM, ICS_PARENTAL_LOCK_NVM, SYSTEM_PIN_NO_NVM, AERIAL_POWER_ON_NVM, COUNTRY_CODE_NVM, REGION_ID_NVM, RF_CHANNEL_NUM_NVM, ASPECT_RATIO_NVM, TV_SCART_TYPE_NVM, WAKEUP_VOLUME_NVM, HDMI_RESOLUTION_NVM, NET_IF_TYPE_NVM, IP_MODE_NVM, IP_ADDRESS_NVM, SUBNET_MASK_NVM, GATEWAY_IP_NVM, DNS_SERVER_IP_NVM, ESSID_NVM, ESSID_PASSWORD_NVM, AD_VOLUME_NVM, SPDIF_OUTPUT_NVM, LIP_SYNC_ADJUSTMENT_NVM, TARGET_REGION_DEPTH_NVM, TARGET_REGION_COUNTRY_NVM, TARGET_REGION_PRIMARY_NVM, TARGET_REGION_SECONDARY_NVM, TARGET_REGION_TERTIARY_NVM, OTA_TYPE_NVM, OTA_LAST_UPDATE_SRCH_NVM, LAST_CHAN_SRCH_NVM, LAST_EIT_UPDATE_NVM, WATCH_HD_PROGRAMMES_NVM, RECORD_HD_PROGRAMMES_NVM, TIME_SHIFT_BUFFER_SIZE_NVM, SUBTITLE_TYPE_NVM, BACKGROUND_SEARCH_START_NVM, BACKGROUND_SEARCH_END_NVM, SERVICE_SEARCH_ENABLED_NVM, SSU_SEARCH_ENABLED_NVM, LIVE_SERVICE_LCN_NVM, PRIMARY_TEXT_LANG_NVM, SECONDARY_TEXT_LANG_NVM, HDMI_AUDIO_OUTPUT_NVM, EIT_SCHED_LIMIT_NVM, NUM_ITEMS_NVM }
- enum E_STANDBY_STATE { STDBY_POWER_ON = 0, STDBY_STANDBY, STDBY_WAKE_FOR_SEARCH, STDBY_UPDATE_SEARCH, STDBY_WAKE_FOR_RECORDING, STDBY_RECORDING }
- enum E_OTA_TYPE { OTA_TYPE_AUTO = 0, OTA_TYPE_MANUAL }
- enum E_VIEW_REC_PREF { VIEW_REC_NEVER = 0, VIEW_REC_ASK, VIEW_REC_ALWAYS }
- enum E_SUBTITLE_TYPE { SUBTITLE_NORMAL = 0, SUBTITLE_HARD_OF_HEARING }
- enum E_PARENTAL_LOCK { PARENTAL_LOCK_OFF = 0, PARENTAL_LOCK_ON = 0xff }
- enum E_NET_IF_TYPE { NET_IF_NONE, NET_IF_WIRED, NET_IF_WIRELESS }

Functions

- void [APP_NvmRestoreDefaults](#) (void)
Resets the DVB's settings to factory defaults.
- U32BIT [APP_NvmReadDefault](#) (E_NVM_ITEMS nvm_item)
Returns the factory default value for the given DVB setting.

- U32BIT [APP_NvmRead](#) (E_NVM_ITEMS nvm_item)
Returns the current value for the given DVB setting.
- void [APP_NvmSave](#) (E_NVM_ITEMS nvm_item, U32BIT new_value, BOOLEAN write_to_flash_now)
Sets the current value for the given DVB setting.
- void [APP_NvmSaveAllNow](#) (void)
Saves DVB values immediately.
- U32BIT [APP_NvmGetDvbSize](#) (void)
Returns the size in bytes the DVB module uses to save its settings.
- void [APP_NvmInitialise](#) (void)
Initialises the DVB core's NVM data.

4.17.1 Detailed Description

Header file for NVM data handling functions.

Date

16/09/2003

4.17.2 Function Documentation

4.17.2.1 U32BIT APP_NvmGetDvbSize (void)

Returns the size in bytes the DVB module uses to save its settings.

Returns

size of DVB settings in bytes

4.17.2.2 U32BIT APP_NvmRead (E_NVM_ITEMS nvm_item)

Returns the current value for the given DVB setting.

Parameters

<i>nvm_item</i>	- value to be read
-----------------	--------------------

Returns

current value

4.17.2.3 U32BIT APP_NvmReadDefault (E_NVM_ITEMS nvm_item)

Returns the factory default value for the given DVB setting.

Parameters

<i>nvm_item</i>	- value to be read
-----------------	--------------------

Returns

factory default value

4.17.2.4 void APP_NvmSave (E_NVM.ITEMS *nvm_item*, U32BIT *new_value*, BOOLEAN *write_to_flash_now*)

Sets the current value for the given DVB setting.

Parameters

<i>nvm_item</i>	- item to be read
<i>new_value</i>	- value for the item
<i>write_to_flash_now</i>	- if TRUE then all the current values will be saved. When changing the values of several items, it will be more efficient to only this this TRUE for the last item

4.18 dvb/inc/dvbver.h File Reference

Application Version support functions.

```
#include <tectype.h> Include dependency graph for dvbver.h:
```

Data Structures

- struct [DVB_VER_STRUCT](#)

Functions

- U8BIT * **App_DvbVersionString** (void)
- U32BIT **App_DvbVersionNumber** (void)
- void **App_DvbVersionData** ([DVB_VER_STRUCT](#) *vptr)

4.18.1 Detailed Description

Application Version support functions.

Date

24th September 2012

Author

Adam Sturtridge

4.19 dvb/src/ap_ca.h File Reference

Application level CA control functions.

Functions

- **BOOLEAN ACA_AcquireCADescrambler** (U8BIT path, void *s_ptr)

4.19.1 Detailed Description

Application level CA control functions.

Date

October 2011

Author

Ocean Blue

4.20 dvb/src/ap_cfdat.h File Reference

Application configuration data.

Defines

- **#define ACFG_INVALID_DB_LANG** 255
- **#define ACFG_UNDEFINED_DB_LANG_CODE** 0
- **#define ACFG_MAX_DB_LANG_CODES** 3

Enumerations

- **enum E_LANGUAGE_STRINGS** { STR_LANG_ENGLISH, STR_LANG_WELSH, STR_LANG_GAELIC, STR_LANG_IRISH, STR_LANG_FRENCH, STR_LANG_GERMAN, STR_LANG_DUTCH, STR_LANG_RUSSIAN, STR_LANG_SIMPLIFIED_CHINESE, STR_LANG_TRADITIONAL_CHINESE, STR_LANG_FINNISH, STR_LANG_SWEDISH, STR_LANG_NORWEGIAN, STR_LANG_DANISH, STR_LANG_MANDARIN, STR_LANG_CANTONESE, STR_LANG_MAORI, STR_LANG_JAPANESE, STR_LANG_ITALIAN, STR_LANG_SPANISH, STR_LANG_KOREAN, STR_LANG_HINDI, STR_LANG_CZECH, STR_LANG_SLOVAK, STR_LANG_AFRIKAANS, STR_LANG_NDEBELE, STR_LANG_NORTHERN_SOTHO, STR_LANG_SOUTHERN_SOTHO, STR_LANG_SWATI, STR_LANG_TSONGA, STR_LANG_TSWANA, STR_LANG_VENDA, STR_LANG_XHOSA, STR_LANG_ZULU, STR_LANG_CROATIAN, STR_LANG_HUNGARIAN, STR_LANG_LATVIAN, STR_LANG_LUXEMBOURGISH, STR_LANG_POLISH, STR_LANG_PORTUGUESE, STR_LANG_SERBI-

```

    AN, STR_LANG_ROMANIAN, STR_LANG_ALBANIAN, STR_LANG_SLOVE-
    NE, STR_LANG_CATALAN, STR_LANG_GALICIAN, STR_LANG_BASQUE,
    STR_LANG_UKRAINIAN }
    • enum E_COUNTRY_STRINGS { STR_COUNTRY_UK, STR_COUNTRY_FRA-
    NCE, STR_COUNTRY_GERMANY, STR_COUNTRY_NETHERLANDS, STR-
    _COUNTRY_RUSSIA, STR_COUNTRY_AUSTRALIA, STR_COUNTRY_CHIN-
    A, STR_COUNTRY_FINLAND, STR_COUNTRY_AUSTRIA, STR_COUNTRY-
    _SWITZERLAND, STR_COUNTRY_CZECHREP, STR_COUNTRY_SLOVAKI-
    A, STR_COUNTRY_SOUTHAFRICA, STR_COUNTRY_BELGIUM, STR_COU-
    NTRY_CROATIA, STR_COUNTRY_HUNGARY, STR_COUNTRY_ITALY, ST-
    R_COUNTRY_LATVIA, STR_COUNTRY_LUXEMBOURG, STR_COUNTRY_-
    POLAND, STR_COUNTRY_PORTUGAL, STR_COUNTRY_SERBIA, STR_C-
    OUNTRY_SLOVENIA, STR_COUNTRY_SPAIN, STR_COUNTRY_SWEDEN,
    STR_COUNTRY_UKRAINE, STR_COUNTRY_ARGENTINA, STR_COUNTRY-
    _BOLIVIA, STR_COUNTRY_BRAZIL, STR_COUNTRY_CHILE, STR_COUN-
    TRY_COLOMBIA, STR_COUNTRY_COSTARICA, STR_COUNTRY_DOMINI-
    CANREP, STR_COUNTRY_ECUADOR, STR_COUNTRY_ELSALVADOR, ST-
    R_COUNTRY_GUATEMALA, STR_COUNTRY_HONDURAS, STR_COUNTR-
    Y_MEXICO, STR_COUNTRY_NICARAGUA, STR_COUNTRY_PANAMA, ST-
    R_COUNTRY_PERU, STR_COUNTRY_VENEZUELA }

```

4.20.1 Detailed Description

Application configuration data.

Date

8/04/2004

Author

Ocean Blue

4.21 dvb/src/ap_dbdef.h File Reference

Application database control.

```

#include "stbsiflt.h" #include "stbsitab.h" #include "stbllist.-
h" Include dependency graph for ap_dbdef.h:

```

Data Structures

- struct [adb_string](#)
- struct [adb_lnb_rec](#)
- struct [adb_bat_version_rec](#)
- struct [adb_satellite_rec](#)
- struct [adb_network_rec](#)
- struct [adb_pmt_version_rec](#)

- struct [adb_transport_rec](#)
- struct [adb_audio_stream_data](#)
- struct [adb_subtitle_stream_data](#)
- struct [adb_ttext_stream_data](#)
- union [adb_stream_data](#)
- struct [adb_stream_rec](#)
- struct [adb_event_desc](#)
- struct [adb_event_rec](#)
- struct [adb_alt_serv_rec](#)
- struct [adb_icon_image](#)
- struct [adb_rct_link_info](#)
- struct [adb_service_rec](#)
- struct [adb_crid_record](#)
- struct [adb_favserv_rec](#)
- struct [adb_favlist_rec](#)
- struct [ADB_ALARM_REC](#)
- struct [ADB_PVR_RECORD_REC](#)
- struct [ADB_TIMER_REC](#)
- struct [s_event](#)

Defines

- #define **ADB_LANG_CODE_UNDEF** 0x756e64
- #define **ADB_LANG_CODE_QAA** 0x716161
- #define **ADB_LANG_CODE_NAR** 0x6e6172
- #define **LINKAGE_DTAG** 0x4a
- #define **SHORT_EVENT_DTAG** 0x4d
- #define **EXTENDED_EVENT_DTAG** 0x4e
- #define **COMPONENT_DTAG** 0x50
- #define **CONTENT_DTAG** 0x54
- #define **PARENTAL_RATING_DTAG** 0x55
- #define **PRIVATE_DATA_SPEC_DTAG** 0x5f
- #define **CONT_ID_DESC_DTAG** 0x76
- #define **FTA_CONTENT_DESC_DTAG** 0x7e
- #define **EXTENSION_DTAG** 0x7f
- #define **USER_DEFINED_DTAG_0x85** 0x85
- #define **USER_DEFINED_DTAG_0x89** 0x89
- #define **LINK_TYPE_EVENT** 0x0d
- #define **LINK_TYPE_EXT_EVENT** 0x0e
- #define **INVALID_FREESAT_SERV_ID** 0x8000

Typedefs

- typedef struct [adb_string](#) ADB_STRING
- typedef struct [adb_lnb_rec](#) ADB_LNB_REC
- typedef struct [adb_bat_version_rec](#) ADB_BAT_VERSION_REC
- typedef struct [adb_satellite_rec](#) ADB_SATELLITE_REC
- typedef struct [adb_network_rec](#) ADB_NETWORK_REC
- typedef struct [adb_pmt_version_rec](#) ADB_PMT_VERSION_REC
- typedef struct [adb_transport_rec](#) ADB_TRANSPORT_REC
- typedef struct [adb_audio_stream_data](#) ADB_AUDIO_STREAM_DATA
- typedef struct [adb_subtitle_stream_data](#) ADB_SUBTITLE_STREAM_DATA
- typedef struct [adb_ttext_stream_data](#) ADB_TTEXT_STREAM_DATA
- typedef union [adb_stream_data](#) ADB_STREAM_DATA
- typedef struct [adb_stream_rec](#) ADB_STREAM_REC
- typedef struct [adb_event_desc](#) ADB_EVENT_DESC
- typedef struct [adb_event_rec](#) ADB_EVENT_REC
- typedef struct [adb_alt_serv_rec](#) ADB_ALT_SERV_REC
- typedef struct [adb_icon_image](#) ADB_IMAGE_ICON
- typedef struct [adb_rct_link_info](#) ADB_RCT_LINK_INFO
- typedef struct [adb_service_rec](#) ADB_SERVICE_REC
- typedef struct [adb_crid_record](#) ADB_CRID_REC
- typedef struct [adb_favserv_rec](#) ADB_FAVSERV_REC
- typedef struct [adb_favlist_rec](#) ADB_FAVLIST_REC
- typedef struct [s_event](#) S_EVENT

Enumerations

- enum **E_STREAM_MATCH_TYPE** { **STREAM_MATCH_EXACT**, **STREAM_MATCH_ASPECT**, **STREAM_MATCH_LANG**, **STREAM_MATCH_TYPE**, **STREAM_MATCH_NOT_ZERO**, **STREAM_MATCH_NONE** }
- enum **E_ICON_TYPE** { **ICON_TYPE_PNG**, **ICON_TYPE_JPEG** }

Functions

- void [DBDEF_Initialise](#) (void)
Initialises the database, preparing for it to be accessed.
- void [DBDEF_RequestAccess](#) (void)
Requests access to the app's database.
- void [DBDEF_ReleaseAccess](#) (void)
Releases access to the app's database.
- BOOLEAN [DBDEF_LoadDatabase](#) (U8BIT *db_pathname)
- void [DBDEF_DeleteAllRecords](#) (void)
- BOOLEAN [DBDEF_SaveDatabase](#) (void)
- void [DBDEF_DeleteRecordsForTunerType](#) (E_STB_DP_SIGNAL_TYPE tuner_type, void *satellite)
- void [DBDEF_DeleteServiceRec](#) ([ADB_SERVICE_REC](#) *s_ptr)

- U16BIT **DBDEF_NumStreamsInList** (**ADB_STREAM_REC** *slist)
- **ADB_STREAM_REC** * **DBDEF_CopyStreamList** (**ADB_STREAM_REC** *slist)
- void **DBDEF_DeleteStreamList** (**ADB_STREAM_REC** *slist)
- void **DBDEF_DeleteAltServList** (**ADB_ALT_SERV_REC** *aslist)
- void **DBDEF_DeleteRCTLinks** (**ADB_RCT_LINK_INFO** *links)
- void **DBDEF_DeletelmageIcons** (**ADB_IMAGE_ICON** *icon_list)
- void **DBDEF_SortServicesByLcn** (void)
Sort the full service list into ascending logical channel number order.
- BOOLEAN **DBDEF_AllocateLcns** (E_STB_DP_SIGNAL_TYPE tuner_type, BOOLEAN assign_lcns)
- BOOLEAN **DBDEF_AllocateLcnsDefault** (E_STB_DP_SIGNAL_TYPE tuner_type)
- U16BIT **DBDEF_GetNumLNBS** (void)
Returns the number of LNB records in the database.
- **ADB_LNB_REC** * **DBDEF_AddLNB** (E_STB_DP_LNB_TYPE type, U16BIT min_freq, U16BIT max_freq)
Add an LNB record to the database.
- BOOLEAN **DBDEF_SetLNBPower** (**ADB_LNB_REC** *lnb, E_STB_DP_LNB_POWER power)
Sets the LNB power setting.
- BOOLEAN **DBDEF_SetLNB22k** (**ADB_LNB_REC** *lnb, BOOLEAN is_22k)
Sets the LNB 22k setting.
- BOOLEAN **DBDEF_SetLNB12V** (**ADB_LNB_REC** *lnb, BOOLEAN is_12v)
Sets the LNB 12V setting.
- BOOLEAN **DBDEF_SetLNBpulsePosition** (**ADB_LNB_REC** *lnb, BOOLEAN is_pulse_posn)
Sets the LNB pulse position setting.
- BOOLEAN **DBDEF_SetLNBDiSeqCPosition** (**ADB_LNB_REC** *lnb, BOOLEAN is_diseqc_posn)
Sets the LNB DiSeqC position setting.
- BOOLEAN **DBDEF_SetLNBDiSeqCTone** (**ADB_LNB_REC** *lnb, E_STB_DP_DISEQC_TONE diseqc_tone)
Sets the LNB DiSeqC tone setting.
- BOOLEAN **DBDEF_SetLNBCSwitch** (**ADB_LNB_REC** *lnb, E_STB_DP_DISEQC_CSWITCH cswitch)
Sets the LNB committed switch setting.
- BOOLEAN **DBDEF_SetLNBUSwitch** (**ADB_LNB_REC** *lnb, U8BIT uswitch)
Sets the LNB uncommitted switch setting.
- BOOLEAN **DBDEF_SetLNBSmatv** (**ADB_LNB_REC** *lnb, BOOLEAN is_smatv)
Sets the LNB SMATV setting.
- BOOLEAN **DBDEF_SetLNBRepeats** (**ADB_LNB_REC** *lnb, U8BIT repeats)
Sets the LNB message repeat setting in the range 0 to 3.
- BOOLEAN **DBDEF_SetLNBUnicable** (**ADB_LNB_REC** *lnb, U32BIT inter_freq, U8BIT chan)

Sets the LNB Unicable settings.

- [ADB_LNB_REC](#) * [DBDEF_GetNextLNBBRec](#) ([ADB_LNB_REC](#) *lnb_ptr)
Returns the next LNB record after the one given. If the argument is NULL then the first record is returned.
- U16BIT [DBDEF_GetNumSatellites](#) (void)
Returns the number of satellite records in the database.
- [ADB_SATELLITE_REC](#) * [DBDEF_AddSatelliteRec](#) ([ADB_STRING](#) *name_str, U16BIT dish_pos, U16BIT long_pos, BOOLEAN east_west, [ADB_LNB_REC](#) *associated_lnb)
Add a satellite record to the database.
- [ADB_SATELLITE_REC](#) * [DBDEF_GetNextSatelliteRec](#) ([ADB_SATELLITE_REC](#) *sat_ptr)
Returns the next satellite record after the one given. If the argument is NULL then the first record is returned.
- U16BIT [DBDEF_GetNumNetworks](#) (void)
Returns the number of networks in the service database.
- [ADB_NETWORK_REC](#) * [DBDEF_AddNetworkRec](#) (U16BIT net_id, [ADB_SATELLITE_REC](#) *satellite)
Adds a new network record to the database with the given network ID.
- void [DBDEF_SetNetworkName](#) ([ADB_NETWORK_REC](#) *n_ptr, U8BIT *name)
Set or change the name of the given network.
- [ADB_NETWORK_REC](#) * [DBDEF_FindNetworkRec](#) (U16BIT net_id, [ADB_SATELLITE_REC](#) *satellite)
Finds the network with the given network ID.
- [ADB_NETWORK_REC](#) * [DBDEF_GetNextNetworkRec](#) ([ADB_NETWORK_REC](#) *n_ptr)
Returns the network following the one given. If the argument is NULL then the first network will be returned.
- U16BIT [DBDEF_GetNumTransports](#) (void)
Returns the number of transports in the service database.
- [ADB_TRANSPORT_REC](#) * [DBDEF_GetNextTransportRec](#) ([ADB_TRANSPORT_REC](#) *t_ptr)
Returns the transport following the one given. If the argument is NULL then the first transport will be returned.
- [ADB_TRANSPORT_REC](#) * [DBDEF_AddTerrestrialTransportRec](#) (U32BIT freq_hz, U8BIT plp_id, [ADB_NETWORK_REC](#) *network)
Adds a terrestrial transport record with the given frequency and PLP id.
- [ADB_TRANSPORT_REC](#) * [DBDEF_FindTerrestrialTransportRec](#) (U32BIT freq_hz, U8BIT plp_id)
- [ADB_TRANSPORT_REC](#) * [DBDEF_AddCableTransportRec](#) (U32BIT freq_hz, U32BIT symbol_rate, [ADB_NETWORK_REC](#) *network)
Adds a cable transport record with the given frequency and symbol rate.
- [ADB_TRANSPORT_REC](#) * [DBDEF_FindCableTransportRec](#) (U32BIT freq_hz, U32BIT symbol_rate)
- void [DBDEF_SetCableTransportSymbolRate](#) ([ADB_TRANSPORT_REC](#) *t_ptr, U16BIT symbol_rate)

- void **DBDEF_SetCableTransportMode** (ADB_TRANSPORT_REC *t_ptr, E_STB_DP_CMODE cmode)
Adds a satellite transport record with the given frequency, symbol rate and polarity.
- ADB_TRANSPORT_REC * **DBDEF_AddSatTransportRec** (U32BIT freq_hz, - U16BIT symbol_rate, E_STB_DP_POLARITY polarity, BOOLEAN dvb_s2, E_STB_DP_MODULATION modulation, ADB_NETWORK_REC *network)
Adds a satellite transport record with the given frequency, symbol rate and polarity.
- ADB_TRANSPORT_REC * **DBDEF_FindSatTransportRec** (U32BIT freq_hz, - U16BIT symbol_rate, E_STB_DP_POLARITY polarity, BOOLEAN dvb_s2, E_STB_DP_MODULATION modulation, void *satellite)
Find the satellite transport record in the database matching the given params.
- void **DBDEF_SetSatTransportTuningParams** (ADB_TRANSPORT_REC *t_ptr, - U32BIT freq_hz, U16BIT symbol_rate, E_STB_DP_POLARITY polarity, BOOLEAN dvb_s2, E_STB_DP_MODULATION modulation, ADB_NETWORK_REC *network)
Sets the tuning parameters for an existing satellite transport.
- ADB_TRANSPORT_REC * **DBDEF_FindTransportRecByIds** (ADB_TRANSPORT_REC *transp, U32BIT net_id, U32BIT onet_id, U32BIT tran_id)
Find a transport record matching the given set of IDs, starting from the given transport.
- void **DBDEF_SetTransportTransportId** (ADB_TRANSPORT_REC *t_ptr, U16BIT tran_id)
Sets the transport ID of the given transport.
- void **DBDEF_SetTransportOrigNetworkId** (ADB_TRANSPORT_REC *t_ptr, U16BIT orig_net_id)
Sets the original network ID of the given transport.
- void **DBDEF_DeleteTransportRec** (ADB_TRANSPORT_REC *t_ptr)
Deletes the given transport from the service database.
- void **DBDEF_ClearTableVersionHistory** (ADB_TRANSPORT_REC *t_ptr)
- U16BIT **DBDEF_GetNumServices** (void)
Returns the total number of services.
- ADB_SERVICE_REC * **DBDEF_GetNextServiceRec** (ADB_SERVICE_REC *s_ptr)
Returns the service after the one given. If NULL is passed then the first service in the list is returned.
- ADB_SERVICE_REC * **DBDEF_GetPrevServiceRec** (ADB_SERVICE_REC *s_ptr)
Returns the service before the one given. If NULL is passed then the last service in the list is returned.
- ADB_SERVICE_REC * **DBDEF_AddServiceRec** (U16BIT serv_id, ADB_TRANSPORT_REC *t_ptr)
Adds a new service record to the service database with the given service ID and parent transport.
- ADB_SERVICE_REC * **DBDEF_CopyServiceRec** (ADB_SERVICE_REC *orig_serv)
Creates a copy of the given service, copying the service's attributes, e.g. service name, scrambled, locked, hidde, etc, but not the runtime specific data such as the stream list or event schedule data.

- void **DBDEF_SetServiceName** (**ADB_SERVICE_REC** *s_ptr, U8BIT *name)
Set or change the name of a service.
- void **DBDEF_SetServiceType** (**ADB_SERVICE_REC** *s_ptr, **ADB_SERVICE_TYPE** serv_type)
Sets the service type for the given service record.
- void **DBDEF_SaveServiceEventSchedule** (**ADB_SERVICE_REC** *s_ptr)
Saves the event schedule of a service to the service database.
- **ADB_SERVICE_REC** * **DBDEF_GetNextServiceOnTransport** (**ADB_SERVICE_REC** *s_ptr, **ADB_TRANSPORT_REC** *t_ptr)
Find the next service following the given service that's on the given transport.
- **ADB_SERVICE_REC** * **DBDEF_FindServiceRecByIds** (**ADB_SERVICE_REC** *servp, U32BIT net_id, U32BIT onet_id, U32BIT tran_id, U32BIT serv_id)
Search for a service with the given IDs.
- **ADB_SERVICE_REC** * **DBDEF_FindServiceRecByLcn** (U16BIT lcn, **ADB_TRANSPORT_REC** *t_ptr, BOOLEAN allocated_lcn)
Find the service with the given LCN, and optionally on the given transport.
- **ADB_SERVICE_REC** * **DBDEF_FindServiceRecByFreesatId** (**ADB_SERVICE_REC** *servp, U16BIT freesat_id)
Search for a service with the given Freesat ID.
- U16BIT **DBDEF_GetReqdAudioPid** (**ADB_SERVICE_REC** *s_ptr, **E_STB_DP_AUDIO_MODE** *audio_mode, **ADB_STREAM_TYPE** *audio_type)
- U16BIT **DBDEF_GetReqdADPid** (**ADB_SERVICE_REC** *s_ptr, **E_STB_DP_AUDIO_MODE** *ad_mode, BOOLEAN *broadcast_mix)
- **E_STREAM_MATCH_TYPE** **DBDEF_GetReqdTtextPid** (**ADB_SERVICE_REC** *s_ptr, BOOLEAN for_subtitles, U16BIT *pid_ptr, U8BIT *magazine, U8BIT *page)
- **E_STREAM_MATCH_TYPE** **DBDEF_GetReqdSubtitleParams** (**ADB_SERVICE_REC** *s_ptr, U16BIT *pid_ptr, U16BIT *cpage_ptr, U16BIT *apage_ptr)
- U16BIT **DBDEF_GetReqdVideoPid** (**ADB_SERVICE_REC** *s_ptr, **ADB_STREAM_TYPE** *video_type)
- **ADB_STRING** * **DBDEF_GetServiceName** (**ADB_SERVICE_REC** *s_ptr, BOOLEAN short_name, BOOLEAN pref_name)
- **ADB_STRING** * **DBDEF_GetServiceProviderName** (**ADB_SERVICE_REC** *s_ptr)
- BOOLEAN **DBDEF_AddAnalogService** (void)
- void **DBDEF_SetAnalogServiceName** (**ADB_SERVICE_REC** *s_ptr, U8BIT *new_name, U8BIT new_len)
- void **DBDEF_SetServicePmtPid** (**ADB_SERVICE_REC** *s_ptr, U16BIT pmt_pid)
- U16BIT **DBDEF_GetServicePmtPid** (**ADB_SERVICE_REC** *s_ptr)
- void **DBDEF_SetServiceFavGroups** (**ADB_SERVICE_REC** *s_ptr, U8BIT groups)
Sets the favourite groups for a service.
- **ADB_EVENT_REC** * **DBDEF_FindScheduleEventById** (**ADB_SERVICE_REC** *s_ptr, U16BIT event_id)
Find an event for a service from its event_id.

- [ADB_EVENT_DESC](#) * [DBDEF_FindEventDescriptor](#) ([ADB_EVENT_DESC](#) *start_desc, U8BIT desc_tag, U32BIT private_data_specifier)
Searches a descriptor list for the first descriptor with the given descriptor tag.
- void [DBDEF_DeleteEventList](#) ([ADB_EVENT_REC](#) *elist)
Deletes all events in the given list.
- void [DBDEF_SetTunedNetwork](#) (U8BIT path, [ADB_NETWORK_REC](#) *n_ptr)
- [ADB_NETWORK_REC](#) * [DBDEF_GetTunedNetwork](#) (U8BIT path)
- void [DBDEF_SetTunedTransport](#) (U8BIT path, [ADB_TRANSPORT_REC](#) *t_ptr)
- [ADB_TRANSPORT_REC](#) * [DBDEF_GetTunedTransport](#) (U8BIT path)
- void [DBDEF_SetTunedService](#) (U8BIT path, [ADB_SERVICE_REC](#) *s_ptr)
- [ADB_SERVICE_REC](#) * [DBDEF_GetTunedService](#) (U8BIT path)
- void [DBDEF_SetTextLang](#) (U8BIT *lang_ids)
- U8BIT * [DBDEF_GetTextLang](#) (void)
- void [DBDEF_SetSecondaryTextLang](#) (U8BIT *lang_ids)
- U8BIT * [DBDEF_GetSecondaryTextLang](#) (void)
- void [DBDEF_SetAudioLang](#) (U8BIT *lang_ids)
- U8BIT * [DBDEF_GetAudioLang](#) (void)
- U8BIT * [DBDEF_GetSecondaryAudioLang](#) (void)
- void [DBDEF_SetSecondaryAudioLang](#) (U8BIT *lang_ids)
- void [DBDEF_TidyDatabaseAfterSearch](#) (E_STB_DP_SIGNAL_TYPE tuner_type, void *satellite, BOOLEAN search_completed, BOOLEAN manual)
Calls any country and tuner type specific function to tidy up the database following a search, such as removing duplicate services, etc.
- void [DBDEF_TidyDatabaseNordig](#) (E_STB_DP_SIGNAL_TYPE tuner_type, BOOLEAN search_completed, BOOLEAN manual)
- void [DBDEF_TidyDatabaseUK](#) (E_STB_DP_SIGNAL_TYPE tuner_type, BOOLEAN search_completed, BOOLEAN manual)
- void [DBDEF_TidyDatabaseSatUK](#) (E_STB_DP_SIGNAL_TYPE tuner_type, BOOLEAN search_completed, BOOLEAN manual)
- void [DBDEF_TidyDatabaseDefault](#) (E_STB_DP_SIGNAL_TYPE tuner_type, BOOLEAN search_completed, BOOLEAN manual)
- void [DBDEF_RemoveEmptyTransports](#) (E_STB_DP_SIGNAL_TYPE tuner_type, void *satellite)
Delete all transport records that don't have any services.
- void [DBDEF_DeleteLinkageDescriptorArray](#) ([SI_LINKAGE_DESC_ENTRY](#) *list_ptr)
- [ADB_CRID_REC](#) * [DBDEF_AddCridRecord](#) (U8BIT *crid, BOOLEAN series, BOOLEAN recommended, BOOLEAN radio_service)
Creates a CRID record and adds it to the database.
- void [DBDEF_SetCridDateTime](#) ([ADB_CRID_REC](#) *c_ptr, U32DHMS date_time)
Sets the date and time fields in the crid record.
- void [DBDEF_SetCridService](#) ([ADB_CRID_REC](#) *c_ptr, U16BIT serv_id)
Sets the service ID in the crid record.

- void [DBDEF_SetCridProgrammeName](#) ([ADB_CRID_REC](#) *c_ptr, U8BIT *prog_name)
Sets the programme name field of the given CRID record.
- void [DBDEF_UpdateCridEitDate](#) ([ADB_CRID_REC](#) *c_ptr)
Updates the stored EIT date of this CRID with the current GMT date.
- void [DBDEF_DeleteCridRecord](#) ([ADB_CRID_REC](#) *c_ptr)
Deletes the given CRID record from the database.
- U16BIT [DBDEF_GetNumCridRecords](#) (void)
Returns the number of CRID records in the database.
- [ADB_CRID_REC](#) * [DBDEF_GetNextCridRecord](#) ([ADB_CRID_REC](#) *c_ptr)
Returns the next CRID record after the one specified. If the record specified is NULL then the first record is returned.
- BOOLEAN [DBDEF_IsValidCridRecord](#) ([ADB_CRID_REC](#) *c_ptr)
Checks whether the given crid record is in the list of valid crid records.
- U16BIT [DBDEF_GetNumFavouriteLists](#) (void)
Returns the number of favourite lists.
- [ADB_FAVLIST_REC](#) * [DBDEF_AddFavouriteList](#) (U8BIT list_id, U8BIT *name, U32BIT user_data, S16BIT index)
Creates a new favourite list and adds it to the list of favourite lists. Creation of the new list will fail if a list_id can't be found for it, or memory allocation fails.
- [ADB_FAVLIST_REC](#) * [DBDEF_GetNextFavouriteList](#) ([ADB_FAVLIST_REC](#) *fav_list)
Returns the favourite list following the given item.
- [ADB_FAVLIST_REC](#) * [DBDEF_FindFavouriteList](#) (U8BIT list_id)
Return the favourite list with the given list id.
- void [DBDEF_SetFavouriteListUserData](#) ([ADB_FAVLIST_REC](#) *fav_list, U32BIT user_data)
Saves the given user data with a favourite list.
- void [DBDEF_MoveFavouriteListTo](#) ([ADB_FAVLIST_REC](#) *fav_list, S16BIT index)
Changes the order of the favourite lists by moving the given list to the given position.
- void [DBDEF_DeleteFavouriteList](#) ([ADB_FAVLIST_REC](#) *fav_list)
Deletes the given favourite list.
- U16BIT [DBDEF_GetNumServicesInFavouriteList](#) ([ADB_FAVLIST_REC](#) *fav_list)
Returns the number of services in the given favourite list.
- [ADB_FAVSERV_REC](#) * [DBDEF_AddServiceToFavouriteList](#) ([ADB_FAVLIST_REC](#) *fav_list, [ADB_SERVICE_REC](#) *serv_ptr, S16BIT index)
Adds a new service to the given favourite list at the given position.
- [ADB_FAVSERV_REC](#) * [DBDEF_FindServiceInFavouriteList](#) ([ADB_FAVLIST_REC](#) *fav_list, void *serv_ptr)
Returns the ADB_FAVSERV_REC from the given favourite list for the given service.
- [ADB_FAVSERV_REC](#) * [DBDEF_GetNextServiceFromFavouriteList](#) ([ADB_FAVLIST_REC](#) *fav_list, [ADB_FAVSERV_REC](#) *fav_serv)
Returns the next favourite list service record.

- [ADB_FAVSERV_REC](#) * [DBDEF_GetPrevServiceFromFavouriteList](#) ([ADB_FAVLIST_REC](#) *fav_list, [ADB_FAVSERV_REC](#) *fav_serv)
Returns the previous favourite list service record.
- void [DBDEF_MoveFavouriteListServiceTo](#) ([ADB_FAVLIST_REC](#) *fav_list, [ADB_FAVSERV_REC](#) *fav_serv, S16BIT index)
Changes the order of the services in the favourite list by moving the given service to the given position.
- void [DBDEF_DeleteServiceFromFavouriteList](#) ([ADB_FAVLIST_REC](#) *fav_list, [ADB_FAVSERV_REC](#) *fav_serv)
Delete the given service from the given favourite list.
- void [DBDEF_DeleteAllServicesFromFavouriteList](#) ([ADB_FAVLIST_REC](#) *fav_list)
Delete the all services from the given favourite list.
- void [DBDEF_SetFavouriteListName](#) ([ADB_FAVLIST_REC](#) *f_ptr, U8BIT *name)
Set or change the name of the given favourite list.
- BOOLEAN [DBDEF_ServiceForTunerType](#) ([ADB_SERVICE_REC](#) *s_ptr, E_STB_DP_SIGNAL_TYPE tuner_type, void *satellite)
Checks whether the given service is a terrestrial, cable or satellite service, and optionally check whether it's on a particular satellite.
- BOOLEAN [DBDEF_TransportForTunerType](#) ([ADB_TRANSPORT_REC](#) *t_ptr, E_STB_DP_SIGNAL_TYPE tuner_type, void *satellite)
Checks whether the given transport is a terrestrial, cable or satellite transport, and optionally check whether it's on a particular satellite.
- [ADB_TIMER_REC](#) * [DBDEF_AddTimerRec](#) (void)
Creates a new timer record in the database, assigning it a unique handle.
- [ADB_TIMER_REC](#) * [DBDEF_FindTimerRec](#) (U32BIT handle)
Returns the timer record with the given timer handle.
- [ADB_TIMER_REC](#) * [DBDEF_GetNextTimerRec](#) ([ADB_TIMER_REC](#) *timer_ptr)
Returns the next timer record after the one given. If the argument is NULL then the first record is returned.
- void [DBDEF_SortTimers](#) (BOOLEAN date_time_order)
Sorts timer list into date/time or alphabetical order.
- void [DBDEF_DeleteTimerRec](#) ([ADB_TIMER_REC](#) *timer)
Deletes the given timer from the database.
- U16BIT [DBDEF_GetNumProfiles](#) (void)
Returns the number of network profiles.
- U16BIT [DBDEF_GetProfileList](#) (void ***profile_list, U16BIT *active_profile)
Gets a list of the available network profiles.
- void [DBDEF_ReleaseProfileList](#) (void **profile_list, U16BIT num_profiles)
Frees a profile list returned by [DBDEF_GetProfileList](#).
- BOOLEAN [DBDEF_IsActiveProfile](#) ([ADB_NETWORK_REC](#) *profile)
Is the given profile the currently active profile?
- ADB_PROFILE_TYPE [DBDEF_GetCurrentProfileType](#) (void)

Returns the current profile type.

- void [DBDEF_SelectBroadcastProfile](#) (void)
Sets the broadcast profile type for for all network, transport and service record accesses.
- void [DBDEF_PushBroadcastProfile](#) (void)
Saves the current profile and any related data so that it can restored using [DBDEF_PopProfile\(\)](#), and sets the broadcast profile type for for all network, transport and service record accesses.
- void [DBDEF_PopProfile](#) (void)
Restores a previously pushed profile.
- BOOLEAN [DBDEF_ServiceInProfile](#) (ADB_SERVICE_REC *s_ptr)
Checks whether the given service is valid for the current profile. There may be multiple CI+ profiles and so data related to the CAM is also checked.
- BOOLEAN [DBDEF_TransportInProfile](#) (ADB_TRANSPORT_REC *t_ptr)
Checks whether the given transport is valid for the current profile. There may be multiple CI+ profiles and so data related to the CAM is also checked.
- BOOLEAN [DBDEF_NetworkInProfile](#) (ADB_NETWORK_REC *n_ptr)
Checks whether the given network is valid for the current profile. There may be multiple CI+ profiles and so data related to the CAM is also checked.
- ADB_NETWORK_REC * [DBDEF_FindOrAddPrivateNetwork](#) (void *satellite)
Find or add a private network, assigning an unused private network ID.
- ADB_STRING * [DBDEF_MakeString](#) (U32BIT lang_code, U8BIT *str_ptr, U16-BIT nbytes)
Creates an ADB_STRING, copying the given data into it. If the string passed in is NULL or the number of bytes is 0, then no string will be created.
- ADB_STRING * [DBDEF_CopyString](#) (ADB_STRING *src_str)
Creates a copy of the given ADB_STRING.
- ADB_STRING * [DBDEF_ConcatSIString](#) (ADB_STRING *str1, SI_STRING_DESC *str2)
Concatenates an SI_STRING_DESC string to the end of an ADB_STRING string, removing any terminating '\0' chars from the first string, and returns a new string.
- void [DBDEF_ReleaseString](#) (ADB_STRING *string)
Releases an ADB_STRING.

4.21.1 Detailed Description

Application database control.

Date

March 2003

4.21.2 Function Documentation

4.21.2.1 **ADB_TRANSPORT_REC*** DBDEF_AddCableTransportRec (U32BIT *freq_hz*, U32BIT *symbol_rate*, ADB_NETWORK_REC * *network*)

Adds a cable transport record with the given frequency and symbol rate.

Parameters

<i>freq_hz</i>	frequency in Hz
<i>symbol_rate</i>	symbol rate
<i>network</i>	network the transport is on

Returns

pointer to the new transport record or NULL if not created

4.21.2.2 **ADB_CRID_REC*** DBDEF_AddCridRecord (U8BIT * *crid*, BOOLEAN *series*, BOOLEAN *recommended*, BOOLEAN *radio_service*)

Creates a CRID record and adds it to the database.

Parameters

<i>crid</i>	CRID string
<i>series</i>	TRUE if this is a series CRID
<i>recommended</i>	TRUE if this is a recommended programme CRID
<i>radio_service</i>	TRUE if this CRID is for a radio service, TV otherwise

Returns

pointer to the created CRID record

4.21.2.3 **ADB_FAVLIST_REC*** DBDEF_AddFavouriteList (U8BIT *list_id*, U8BIT * *name*, U32BIT *user_data*, S16BIT *index*)

Creates a new favourite list and adds it to the list of favourite lists. Creation of the new list will fail if a *list_id* can't be found for it, or memory allocation fails.

Parameters

<i>list_id</i>	id to be given to the list, should be passed as 0 to have an id assigned. If an id is supplied and a list already exists with that id then no list will be created.
<i>name</i>	name to be given to the list, call be NULL
<i>user_data</i>	user defined value to be stored in the list,
<i>index</i>	where to add the new list into the existing lists, 0 = at start, negative = at end, else 0 based index

Returns

pointer to new favourite list, or NULL on failure

4.21.2.4 ADB_LNB_REC* DBDEF_AddLNB (E_STB_DP_LNB_TYPE *type*, U16BIT *min_freq*, U16BIT *max_freq*)

Add an LNB record to the database.

Parameters

<i>type</i>	single, universal or uncable
<i>min_freq</i>	min frequency
<i>max_freq</i>	max frequency

Returns

pointer to the new LNB record or NULL if not created

4.21.2.5 ADB_NETWORK_REC* DBDEF_AddNetworkRec (U16BIT *net_id*, ADB_SATELLITE_REC * *satellite*)

Adds a new network record to the database with the given network ID.

Parameters

<i>net_id</i>	network ID
<i>satellite</i>	satellite that contains the network, can be NULL

Returns

pointer to new network record

4.21.2.6 ADB_SATELLITE_REC* DBDEF_AddSatelliteRec (ADB_STRING * *name_str*, U16BIT *dish_pos*, U16BIT *long_pos*, BOOLEAN *east_west*, ADB_LNB_REC * *associated_lnb*)

Add a satellite record to the database.

Parameters

<i>name_str</i>	satellite name
<i>dish_pos</i>	
<i>long_pos</i>	position in 1/10ths of a degree
<i>east_west</i>	TRUE=east, FALSE=west
<i>associated_lnb</i>	LNB record the satellite record is associated with

Returns

pointer to the new satellite record or NULL if not created

4.21.2.7 ADB_TRANSPORT_REC* DBDEF_AddSatTransportRec (U32BIT *freq_hz*, U16BIT *symbol_rate*, E_STB_DP_POLARITY *polarity*, BOOLEAN *dvb_s2*, E_STB_DP_MODULATION *modulation*, ADB_NETWORK_REC * *network*)

Adds a satellite transport record with the given frequency, symbol rate and polarity.

Parameters

<i>freq_hz</i>	frequency in Hz
<i>symbol_rate</i>	symbol rate
<i>polarity</i>	polarity
<i>dvb_s2</i>	TRUE for DVB-S2
<i>modulation</i>	modulation type
<i>network</i>	network the transport is on

Returns

pointer to the new transport record or NULL if not created

4.21.2.8 ADB_SERVICE_REC* DBDEF_AddServiceRec (U16BIT *serv_id*, ADB_TRANSPORT_REC * *t_ptr*)

Adds a new service record to the service database with the given service ID and parent transport.

Parameters

<i>serv_id</i>	service ID for the new service
<i>t_ptr</i>	parent transport for the new service

Returns

pointer to the new service, or NULL if the creation fails

4.21.2.9 ADB_FAVSERV_REC* DBDEF_AddServiceToFavouriteList (ADB_FAVLIST_REC * *fav_list*, ADB_SERVICE_REC * *serv_ptr*, S16BIT *index*)

Adds a new service to the given favourite list at the given position.

Parameters

<i>fav_list</i>	list service is to be added to
<i>serv_ptr</i>	service to be added to the list
<i>index</i>	position to add the service in the list, 0 = at start, negative = at end, any other value is the position

Returns

new favourite service record, or NULL on failure

4.21.2.10 ADB_TRANSPORT_REC* DBDEF_AddTerrestrialTransportRec (U32BIT *freq_hz*, U8BIT *plp_id*, ADB_NETWORK_REC * *network*)

Adds a terrestrial transport record with the given frequency and PLP id.

Parameters

<i>freq_hz</i>	frequency in Hz
<i>plp_id</i>	PLP id for T2, pass 0 for T
<i>network</i>	network the transport is on

Returns

pointer to the new transport record or NULL if not created

4.21.2.11 ADB_TIMER_REC* DBDEF_AddTimerRec (void)

Creates a new timer record in the database, assigning it a unique handle.

Returns

pointer to the new timer, or NULL on failure

4.21.2.12 ADB_STRING* DBDEF_ConcatSIString (ADB_STRING * *str1*, SI_STRING_DESC * *str2*)

Concatenates an SI_STRING_DESC string to the end of an ADB_STRING string, removing any terminating '\0' chars from the first string, and returns a new string.

Parameters

<i>str1</i>	string 1
<i>str2</i>	string 2

Returns

new ADB_STRING containing the contents of both strings

4.21.2.13 ADB_SERVICE_REC* DBDEF_CopyServiceRec (ADB_SERVICE_REC * *orig_serv*)

Creates a copy of the given service, copying the service's attributes, e.g. service name, scrambled, locked, hidde, etc, but not the runtime specific data such as the stream list or event schedule data.

Parameters

<i>orig_serv</i>	service to be copied
------------------	----------------------

Returns

new copied service, NULL on failure

4.21.2.14 **ADB_STRING*** DBDEF_CopyString (**ADB_STRING *** *src_str*)

Creates a copy of the given ADB_STRING.

Parameters

<i>src_str</i>	string to be copied
----------------	---------------------

Returns

new ADB_STRING

4.21.2.15 **void** DBDEF_DeleteAllServicesFromFavouriteList (**ADB_FAVLIST_REC *** *fav_list*)

Delete the all services from the given favourite list.

Parameters

<i>fav_list</i>	favourite list
-----------------	----------------

4.21.2.16 **void** DBDEF_DeleteCridRecord (**ADB_CRID_REC *** *c_ptr*)

Deletes the given CRID record from the database.

Parameters

<i>c_ptr</i>	pointer to CRID record to be deleted
--------------	--------------------------------------

4.21.2.17 **void** DBDEF_DeleteEventList (**ADB_EVENT_REC *** *elist*)

Deletes all events in the given list.

Parameters

<i>elist</i>	list of events to be deleted
--------------	------------------------------

4.21.2.18 **void** DBDEF_DeleteFavouriteList (**ADB_FAVLIST_REC *** *fav_list*)

Deletes the given favourite list.

Parameters

<i>fav_list</i>	favourite list to be deleted
-----------------	------------------------------

4.21.2.19 void DBDEF_DeleteServiceFromFavouriteList (ADB_FAVLIST_REC * *fav_list*, ADB_FAVSERV_REC * *fav_serv*)

Delete the given service from the given favourite list.

Parameters

<i>fav_list</i>	favourite list to delete service from
<i>fav_serv</i>	service to be deleted from the favourite list

4.21.2.20 void DBDEF_DeleteTimerRec (ADB_TIMER_REC * *timer*)

Deletes the given timer from the database.

Parameters

<i>timer</i>	timer to be deleted
--------------	---------------------

4.21.2.21 void DBDEF_DeleteTransportRec (ADB_TRANSPORT_REC * *t_ptr*)

Deletes the given transport from the service database.

Parameters

<i>t_ptr</i>	pointer to transport to be deleted
--------------	------------------------------------

4.21.2.22 ADB_EVENT_DESC* DBDEF_FindEventDescriptor (ADB_EVENT_DESC * *start_desc*, U8BIT *desc_tag*, U32BIT *private_data_specifier*)

Searches a descriptor list for the first descriptor with the given descriptor tag.

Parameters

<i>start_desc</i>	descriptor where the search should start
<i>desc_tag</i>	descriptor tag to be found
<i>private_data_specifier</i>	the value of the private data specifier that must precede the descriptor being searched for. Use 0 if it isn't required.

Returns

pointer to the descriptor, or NULL if not found

4.21.2.23 ADB_FAVLIST_REC* DBDEF_FindFavouriteList (U8BIT *list_id*)

Return the favourite list with the given list id.

Parameters

<i>list_id</i>	favourite list id to find
----------------	---------------------------

Returns

pointer to favourite list record, or NULL if not found

4.21.2.24 ADB_NETWORK_REC* DBDEF_FindNetworkRec (U16BIT *net_id*, ADB_SATELLITE_REC * *satellite*)

Finds the network with the given network ID.

Parameters

<i>net_id</i>	network ID to be found
<i>satellite</i>	satellite the network is on, can be NULL

Returns

pointer to network record, or NULL if not found

4.21.2.25 ADB_NETWORK_REC* DBDEF_FindOrAddPrivateNetwork (void * *satellite*)

Find or add a private network, assigning an unused private network ID.

Parameters

<i>satellite</i>	optional satellite, can be NULL
------------------	---------------------------------

Returns

pointer to existing or new network record

4.21.2.26 ADB_TRANSPORT_REC* DBDEF_FindSatTransportRec (U32BIT *freq_hz*, U16BIT *symbol_rate*, E_STB_DP_POLARITY *polarity*, BOOLEAN *dvb_s2*, E_STB_DP_MODULATION *modulation*, void * *satellite*)

Find the satellite transport record in the database matching the given params.

Parameters

<i>freq_hz</i>	frequency in Hz
<i>symbol_rate</i>	symbol rate
<i>dvb_s2</i>	TRUE if DVB-S2
<i>modulation</i>	modulation type

Returns

pointer to the transport record or NULL if not found

4.21.2.27 **ADB_EVENT_REC*** DBDEF_FindScheduleEventById (
 ADB_SERVICE_REC * s_ptr, U16BIT event_id)

Find an event for a service from its event_id.

Parameters

<i>s_ptr</i>	service to be searched
<i>event_id</i>	ID of the event to be found

Returns

pointer to the ADB_EVENT_REC, or NULL if not found

4.21.2.28 **ADB_FAVSERV_REC*** DBDEF_FindServiceInFavouriteList (
 ADB_FAVLIST_REC * fav_list, void * serv_ptr)

Returns the ADB_FAVSERV_REC from the given favourite list for the given service.

Parameters

<i>fav_list</i>	favourite list to be searched
<i>serv_ptr</i>	service to be searched for

Returns

pointer to service record, or NULL if not found

4.21.2.29 **ADB_SERVICE_REC*** DBDEF_FindServiceRecByFreesatId (
 ADB_SERVICE_REC * servp, U16BIT freesat_id)

Search for a service with the given Freesat ID.

Parameters

<i>servp</i>	start searching from this service, NULL to start from first service
<i>freesat_id</i>	Freesat service ID

Returns

pointer to the next service found, or NULL if no service found

4.21.2.30 ADB_SERVICE_REC* DBDEF_FindServiceRecByIds (
ADB_SERVICE_REC * servp, U32BIT net_id, U32BIT onet_id, U32BIT tran_id,
U32BIT serv_id)

Search for a service with the given IDs.

Parameters

<i>servp</i>	start searching from this service, NULL to start from first service
<i>net_id</i>	network ID, use ADB_INVALID_DVB_ID if not to be used
<i>onet_id</i>	original network ID, use ADB_INVALID_DVB_ID if not to be used
<i>tran_id</i>	transport ID, use ADB_INVALID_DVB_ID if not to be used
<i>serv_id</i>	service ID, use ADB_INVALID_DVB_ID if not to be used

Returns

pointer to the next service found, or NULL if no service found

4.21.2.31 ADB_SERVICE_REC* DBDEF_FindServiceRecByLcn (U16BIT lcn,
ADB_TRANSPORT_REC * t_ptr, BOOLEAN allocated_lcn)

Find the service with the given LCN, and optionally on the given transport.

Parameters

<i>lcn</i>	LCN
<i>t_ptr</i>	transport the service must be on, use NULL for any transport
<i>allocated_lcn</i>	TRUE if the search should be based on the LCN a service has been allocated rather than the one it requested (its service LCN)

Returns

pointer to service if found, NULL otherwise

4.21.2.32 ADB_TIMER_REC* DBDEF_FindTimerRec (U32BIT handle)

Returns the timer record with the given timer handle.

Parameters

<i>handle</i>	timer handle
---------------	--------------

Returns

pointer to the timer record, or NULL if not found

4.21.2.33 **ADB_TRANSPORT_REC*** DBDEF_FindTransportRecByIds (
ADB_TRANSPORT_REC * *transp*, U32BIT *net_id*, U32BIT *onet_id*, U32BIT *tran_id*)

Find a transport record matching the given set of IDs, starting from the given transport.

Parameters

<i>transp</i>	start search from this transport, use NULL to start from the first transport
<i>net_id</i>	network ID, use ADB_INVALID_DVB_ID to ignore this value
<i>onet_id</i>	original network ID, use ADB_INVALID_DVB_ID to ignore this value
<i>tran_id</i>	transport ID, use ADB_INVALID_DVB_ID to ignore this value

Returns

pointer to transport record, NULL if none found

4.21.2.34 **ADB_PROFILE_TYPE** DBDEF_GetCurrentProfileType (void)

Returns the current profile type.

Returns

Current profile type in use

4.21.2.35 **ADB_CRID_REC*** DBDEF_GetNextCridRecord (**ADB_CRID_REC** * *c_ptr*)

Returns the next CRID record after the one specified. If the record specified is NULL then the first record is returned.

Parameters

<i>c_ptr</i>	CRID record pointer, NULL if first record is required
--------------	---

Returns

pointer to the next record after the one specified

4.21.2.36 **ADB_FAVLIST_REC*** DBDEF_GetNextFavouriteList (
ADB_FAVLIST_REC * *fav_list*)

Returns the favourite list following the given item.

Parameters

<i>fav_list</i>	return the item after this one, if NULL, returns the first item
-----------------	---

Returns

next favourite list, or NULL if no more

4.21.2.37 ADB_LNB_REC* DBDEF_GetNextLNBRec (ADB_LNB_REC * *lnb_ptr*)

Returns the next LNB record after the one given. If the argument is NULL then the first record is returned.

Parameters

<i>lnb_ptr</i>	LNB record pointer, NULL if first record is required
----------------	--

Returns

pointer to next LNB record

4.21.2.38 ADB_NETWORK_REC* DBDEF_GetNextNetworkRec (ADB_NETWORK_REC * *n_ptr*)

Returns the network following the one given. If the argument is NULL then the first network will be returned.

Parameters

<i>n_ptr</i>	network record, NULL if first network is required
--------------	---

Returns

pointer to next network, NULL if no more networks

4.21.2.39 ADB_SATELLITE_REC* DBDEF_GetNextSatelliteRec (ADB_SATELLITE_REC * *sat_ptr*)

Returns the next satellite record after the one given. If the argument is NULL then the first record is returned.

Parameters

<i>sat_ptr</i>	satellite record pointer, NULL if first record is required
----------------	--

Returns

pointer to next satellite record

4.21.2.40 ADB_FAVSERV_REC* DBDEF_GetNextServiceFromFavouriteList (ADB_FAVLIST_REC * *fav_list*, ADB_FAVSERV_REC * *fav_serv*)

Returns the next favourite list service record.

Parameters

<i>fav_list</i>	favourite list
<i>fav_serv</i>	current service, or NULL if first service to be returned

Returns

pointer to service record, or NULL if no more services

4.21.2.41 ADB_SERVICE_REC* DBDEF_GetNextServiceOnTransport (ADB_SERVICE_REC * *s_ptr*, ADB_TRANSPORT_REC * *t_ptr*)

Find the next service following the given service that's on the given transport.

Parameters

<i>s_ptr</i>	find the service after this one, use NULL to get the first service
<i>t_ptr</i>	transport the service is on

Returns

pointer to the service found, or NULL if not found

4.21.2.42 ADB_SERVICE_REC* DBDEF_GetNextServiceRec (ADB_SERVICE_REC * *s_ptr*)

Returns the service after the one given. If NULL is passed then the first service in the list is returned.

Parameters

<i>s_ptr</i>	pointer to service, NULL if first service is to be returned
--------------	---

Returns

pointer to service, or NULL if no more services

4.21.2.43 ADB_TIMER_REC* DBDEF_GetNextTimerRec (ADB_TIMER_REC * *timer_ptr*)

Returns the next timer record after the one given. If the argument is NULL then the first record is returned.

Parameters

<i>timer_ptr</i>	timer record pointer, NULL if first record is required
------------------	--

Returns

pointer to next timer record

4.21.2.44 ADB_TRANSPORT_REC* DBDEF_GetNextTransportRec (ADB_TRANSPORT_REC * *t_ptr*)

Returns the transport following the one given. If the argument is NULL then the first transport will be returned.

Parameters

<i>t_ptr</i>	transport record, NULL if first transport is required
--------------	---

Returns

pointer to next transport, NULL if no more transports

4.21.2.45 U16BIT DBDEF_GetNumCridRecords (void)

Returns the number of CRID records in the database.

Returns

number of records

4.21.2.46 U16BIT DBDEF_GetNumFavouriteLists (void)

Returns the number of favourite lists.

Returns

number of favourite lists

4.21.2.47 U16BIT DBDEF_GetNumLNBS (void)

Returns the number of LNB records in the database.

Returns

number of LNB records

4.21.2.48 U16BIT DBDEF_GetNumNetworks (void)

Returns the number of networks in their service database.

Returns

number of transports

4.21.2.49 U16BIT DBDEF_GetNumSatellites (void)

Returns the number of satellite records in the database.

Returns

number of satellite records

4.21.2.50 U16BIT DBDEF_GetNumServices (void)

Returns the total number of services.

Returns

total number of services

4.21.2.51 U16BIT DBDEF_GetNumServicesInFavouriteList (ADB_FAVLIST_REC * fav_list)

Returns the number of services in the given favourite list.

Parameters

<i>fav_list</i>	favourite list
-----------------	----------------

Returns

number of services

4.21.2.52 U16BIT DBDEF_GetNumTransports (void)

Returns the number of transports in ther service database.

Returns

number of transports

4.21.2.53 ADB_FAVSERV_REC* DBDEF_GetPrevServiceFromFavouriteList (ADB_FAVLIST_REC * fav_list, ADB_FAVSERV_REC * fav_serv)

Returns the previous favourite list service record.

Parameters

<i>fav_list</i>	favourite list
<i>fav_serv</i>	current service, or NULL if last service to be returned

Returns

pointer to service record, or NULL if no more services

4.21.2.54 ADB_SERVICE_REC* DBDEF_GetPrevServiceRec (
ADB_SERVICE_REC * *s_ptr*)

Returns the service before the one given. If NULL is passed then the last service in the list is returned.

Parameters

<i>s_ptr</i>	pointer to service, NULL if last service is to be returned
--------------	--

Returns

pointer to service, or NULL if no more services

4.21.2.55 U16BIT DBDEF_GetProfileList (void * *profile_list*, U16BIT * *active_profile*)**

Gets a list of the available network profiles.

Parameters

<i>profile_list</i>	address of pointer to an array that will be allocated by this function that will contain the profiles
<i>active_profile</i>	pointer to return the index of the currently active profile

Returns

number of profiles in the returned array

4.21.2.56 BOOLEAN DBDEF_IsActiveProfile (ADB_NETWORK_REC * *profile*)

Is the given profile the currently active profile?

Parameters

<i>profile</i>	profile
----------------	---------

Returns

TRUE if the given profile is the currently active profile, FALSE otherwise

4.21.2.57 BOOLEAN DBDEF_IsValidCridRecord (ADB_CRID_REC * *c_ptr*)

Checks whether the given crid record is in the list of valid crid records.

Parameters

<i>c_ptr</i>	crid record to look for
--------------	-------------------------

Returns

TRUE if crid record is valid, FALSE otherwise

4.21.2.58 ADB_STRING* DBDEF_MakeString (U32BIT lang_code, U8BIT * str_ptr, U16BIT nbytes)

Creates an ADB_STRING, copying the given data into it. If the string passed in is NULL or the number of bytes is 0, then no string will be created.

Parameters

<i>lang_code</i>	ISO 639-2 3 char language code
<i>str_ptr</i>	pointer to string data to be copied into the ADB_STRING
<i>nbytes</i>	number of bytes of string data

Returns

pointer to new ADB_STRING, or NULL

4.21.2.59 void DBDEF_MoveFavouriteListServiceTo (ADB_FAVLIST_REC * fav_list, ADB_FAVSERV_REC * fav_serv, S16BIT index)

Changes the order of the services in the favourite list by moving the given service to the given position.

Parameters

<i>fav_list</i>	favourite list
<i>fav_serv</i>	service to be moved
<i>index</i>	position to move the service to

Returns

TRUE if successful, FALSE otherwise

4.21.2.60 void DBDEF_MoveFavouriteListTo (ADB_FAVLIST_REC * fav_list, S16BIT index)

Changes the order of the favourite lists by moving the given list to the given position.

Parameters

<i>fav_list</i>	favourite list to be moved
<i>index</i>	position to move the list to

Returns

TRUE if successful, FALSE otherwise

4.21.2.61 **BOOLEAN DBDEF_NetworkInProfile (ADB_NETWORK_REC * *n_ptr*)**

Checks whether the given network is valid for the current profile. There may be multiple CI+ profiles and so data related to the CAM is also checked.

Parameters

<i>n_ptr</i>	network to be checked
--------------	-----------------------

Returns

TRUE if the network is in the current profile, FALSE otherwise

4.21.2.62 **void DBDEF_ReleaseProfileList (void ** *profile_list*, U16BIT *num_profiles*)**

Frees a profile list returned by DBDEF_GetProfileList.

Parameters

<i>profile_list</i>	profile list to be freed
<i>num_profiles</i>	number of profiles in the list

4.21.2.63 **void DBDEF_ReleaseString (ADB_STRING * *string*)**

Releases an ADB_STRING.

Parameters

<i>string</i>	string to be freed
---------------	--------------------

4.21.2.64 **void DBDEF_RemoveEmptyTransports (E_STB_DP_SIGNAL_TYPE *tuner_type*, void * *satellite*)**

Delete all transport records that don't have any services.

Parameters

<i>tuner_type</i>	- empty transports of this signal type will be deleted
<i>satellite</i>	- if the signal type is SIGNAL_QPSK then this parameter specifies the satellite the transports must be on. Ignored if tuner type isn't SIGNAL_QPSK, and can be NULL to indicate all satellites.

4.21.2.65 **void DBDEF_SaveServiceEventSchedule (ADB_SERVICE_REC * *s_ptr*)**

Saves the event schedule of a service to the service database.

Parameters

<i>s_ptr</i>	event data to be saved for this service
--------------	---

4.21.2.66 **BOOLEAN DBDEF_ServiceForTunerType** (**ADB_SERVICE_REC** * *s_ptr*, **E_STB_DP_SIGNAL_TYPE** *tuner_type*, void * *satellite*)

Checks whether the given service is a terrestrial, cable or satellite service, and optionally check whether it's on a particular satellite.

Parameters

<i>s_ptr</i>	service to be checked
<i>tuner_type</i>	type of tuner to check
<i>satellite</i>	if tuner type is SIGNAL_QPSK then check whether the service is from this satellite. This can be NULL if the satellite isn't important and is ignored for a non-satellite tuner.

Returns

TRUE if the service is of the given type, FALSE otherwise

4.21.2.67 **BOOLEAN DBDEF_ServiceInProfile** (**ADB_SERVICE_REC** * *s_ptr*)

Checks whether the given service is valid for the current profile. There may be multiple CI+ profiles and so data related to the CAM is also checked.

Parameters

<i>s_ptr</i>	service to be checked
--------------	-----------------------

Returns

TRUE if the service isn't part of a network or is in the current profile, FALSE otherwise

4.21.2.68 **void DBDEF_SetCridDateTime** (**ADB_CRID_REC** * *c_ptr*, **U32DHMS** *date_time*)

Sets the date and time fields in the crid record.

Parameters

<i>c_ptr</i>	pointer to CRID record
<i>date_time</i>	date & time to set in the crid

4.21.2.69 void DBDEF_SetCridProgrammeName (ADB_CRID_REC * *c_ptr*, U8BIT * *prog_name*)

Sets the programme name field of the given CRID record.

Parameters

<i>c_ptr</i>	pointer to CRID record
<i>prog_name</i>	name string to save in the CRID record

4.21.2.70 void DBDEF_SetCridService (ADB_CRID_REC * *c_ptr*, U16BIT *serv_id*)

Sets the service ID in the crid record.

Parameters

<i>c_ptr</i>	pointer to CRID record
<i>serv_id</i>	service ID

4.21.2.71 void DBDEF_SetFavouriteListName (ADB_FAVLIST_REC * *f_ptr*, U8BIT * *name*)

Set or change the name of the given favourite list.

Parameters

<i>f_ptr</i>	pointer to favourite record to be updated
<i>name</i>	of the favourite list

Returns

void

4.21.2.72 void DBDEF_SetFavouriteListUserData (ADB_FAVLIST_REC * *fav_list*, U32BIT *user_data*)

Saves the given user data with a favourite list.

Parameters

<i>fav_list</i>	favourite list
<i>user_data</i>	data to be stored with the list

4.21.2.73 BOOLEAN DBDEF_SetLNB12V (ADB_LNB_REC * *lnb*, BOOLEAN *is_12v*)

Sets the LNB 12V setting.

Parameters

<i>lnb</i>	pointer to LNB record
<i>is_12v</i>	12V setting

Returns

TRUE if value set successfully, FALSE otherwise

4.21.2.74 `BOOLEAN DBDEF_SetLNB22k (ADB_LNB_REC * lnb, BOOLEAN is_22k)`

Sets the LNB 22k setting.

Parameters

<i>lnb</i>	pointer to LNB record
<i>is_22k</i>	22k setting

Returns

TRUE if value set successfully, FALSE otherwise

4.21.2.75 `BOOLEAN DBDEF_SetLNBCSwitch (ADB_LNB_REC * lnb,
E_STB_DP_DISEQC_CSWITCH cswitch)`

Sets the LNB committed switch setting.

Parameters

<i>lnb</i>	pointer to LNB record
<i>cswitch</i>	switch setting

Returns

TRUE if value set successfully, FALSE otherwise

4.21.2.76 `BOOLEAN DBDEF_SetLNBDiSEqCPosition (ADB_LNB_REC * lnb,
BOOLEAN is_diseqc_posn)`

Sets the LNB DiSEqC position setting.

Parameters

<i>lnb</i>	pointer to LNB record
<i>is_diseqc_posn</i>	diseqc position setting

Returns

TRUE if value set successfully, FALSE otherwise

4.21.2.77 `BOOLEAN DBDEF_SetLNBDiSEqCTone (ADB_LNB_REC * lnb,
E_STB_DP_DISEQC_TONE diseqc_tone)`

Sets the LNB DiSEqC tone setting.

Parameters

<i>lnb</i>	pointer to LNB record
<i>diseqc_tone</i>	diseqc tone setting

Returns

TRUE if value set successfully, FALSE otherwise

4.21.2.78 `BOOLEAN DBDEF_SetLNBPowder (ADB_LNB_REC * lnb,
E_STB_DP_LNB_POWER power)`

Sets the LNB power setting.

Parameters

<i>lnb</i>	pointer to LNB record
<i>power</i>	power setting

Returns

TRUE if value set successfully, FALSE otherwise

4.21.2.79 `BOOLEAN DBDEF_SetLNBPulsePosition (ADB_LNB_REC * lnb, BOOLEAN
is_pulse_posn)`

Sets the LNB pulse position setting.

Parameters

<i>lnb</i>	pointer to LNB record
<i>is_pulse_posn</i>	pulse position setting

Returns

TRUE if value set successfully, FALSE otherwise

4.21.2.80 `BOOLEAN DBDEF_SetLNBBRepeats (ADB_LNB_REC * lnb, U8BIT repeats)`

Sets the LNB message repeat setting in the range 0 to 3.

Parameters

<i>lnb</i>	pointer to LNB record
<i>repeats</i>	repeat setting

Returns

TRUE if value set successfully, FALSE otherwise

4.21.2.81 **BOOLEAN DBDEF_SetLNBSmatv (ADB_LNB_REC * *lnb*, BOOLEAN *is_smatv*)**

Sets the LNB SMATV setting.

Parameters

<i>lnb</i>	pointer to LNB record
<i>is_smatv</i>	SMATV setting

Returns

TRUE if value set successfully, FALSE otherwise

4.21.2.82 **BOOLEAN DBDEF_SetLNBUnicable (ADB_LNB_REC * *lnb*, U32BIT *inter_freq*, U8BIT *chan*)**

Sets the LNB Unicable settings.

Parameters

<i>lnb</i>	pointer to LNB record
<i>inter_freq</i>	intermediate frequency setting
<i>chan</i>	channel setting

Returns

TRUE if value set successfully, FALSE otherwise

4.21.2.83 **BOOLEAN DBDEF_SetLNBUSwitch (ADB_LNB_REC * *lnb*, U8BIT *uswitch*)**

Sets the LNB uncommitted switch setting.

Parameters

<i>lnb</i>	pointer to LNB record
<i>uswitch</i>	switch setting

Returns

TRUE if value set successfully, FALSE otherwise

4.21.2.84 void DBDEF_SetNetworkName (ADB_NETWORK_REC * *n_ptr*, U8BIT * *name*)

Set or change the name of the given network.

Parameters

<i>n_ptr</i>	pointer to network record to be updated
<i>name</i>	name to be given to the network, can be any format, or NULL to clear the network name

4.21.2.85 void DBDEF_SetSatTransportTuningParams (ADB_TRANSPORT_REC * *t_ptr*, U32BIT *freq_hz*, U16BIT *symbol_rate*, E_STB_DP_POLARITY *polarity*, BOOLEAN *dvb_s2*, E_STB_DP_MODULATION *modulation*, ADB_NETWORK_REC * *network*)

Sets the tuning parameters for an existing satellite transport.

Parameters

<i>t_ptr</i>	satellite transport record
<i>freq_hz</i>	frequency in Hz
<i>symbol_rate</i>	symbol rate
<i>polarity</i>	polarity
<i>dvb_s2</i>	TRUE for DVB-S2
<i>modulation</i>	modulation type
<i>network</i>	network the transport is on, can be NULL

4.21.2.86 void DBDEF_SetServiceFavGroups (ADB_SERVICE_REC * *s_ptr*, U8BIT *groups*)

Sets the favourite groups for a service.

Parameters

<i>s_ptr</i>	service
<i>groups</i>	bitmask containing the favourite groups the service is to be in

4.21.2.87 void DBDEF_SetServiceName (ADB_SERVICE_REC * *s_ptr*, U8BIT * *name*)

Set or change the name of a service.

Parameters

<i>s_ptr</i>	service
<i>name</i>	name to be given to the service, can be any format, or NULL to clear the current name

4.21.2.88 void DBDEF_SetServiceType (ADB_SERVICE_REC * *s_ptr*,
ADB_SERVICE_TYPE *serv_type*)

Sets the service type for the given service record.

Parameters

<i>s_ptr</i>	service
<i>serv_type</i>	type of service

4.21.2.89 void DBDEF_SetTransportOrigNetworkId (ADB_TRANSPORT_REC *
t_ptr, U16BIT *orig_net_id*)

Sets the original network ID of the given transport.

Parameters

<i>t_ptr</i>	transport record the ID will be set for
<i>orig_net_id</i>	original network ID to be set

4.21.2.90 void DBDEF_SetTransportTransportId (ADB_TRANSPORT_REC * *t_ptr*,
U16BIT *tran_id*)

Sets the transport ID of the given transport.

Parameters

<i>t_ptr</i>	transport record the ID will be set for
<i>tran_id</i>	transport ID to be set

4.21.2.91 void DBDEF_SortTimers (BOOLEAN *date_time_order*)

Sorts timer list into date/time or alphabetical order.

Parameters

<i>date_time_order</i>	TRUE to sort into date/time order, FALSE for alphabetical
------------------------	---

4.21.2.92 void DBDEF_TidyDatabaseAfterSearch (E_STB_DP_SIGNAL_TYPE *tuner_type*, void * *satellite*, BOOLEAN *search_completed*, BOOLEAN *manual*)

Calls any country and tuner type specific function to tidy up the database following a search, such as removing duplicate services, etc.

Parameters

<i>tuner_type</i>	tuner type used for the search
<i>satellite</i>	satellite search was performed on for SIGNAL_QPSK, ignored otherwise
<i>search_completed</i>	TRUE if the search completed, FALSE if terminated prematurely
<i>manual</i>	TRUE if a manual search was performed, FALSE otherwise

4.21.2.93 BOOLEAN DBDEF_TransportForTunerType (ADB_TRANSPORT_REC * *t_ptr*, E_STB_DP_SIGNAL_TYPE *tuner_type*, void * *satellite*)

Checks whether the given transport is a terrestrial, cable or satellite transport, and optionally check whether it's on a particular satellite.

Parameters

<i>t_ptr</i>	transport to be checked
<i>tuner_type</i>	type of tuner to check
<i>satellite</i>	if tuner type is SIGNAL_QPSK then check whether the transport is from this satellite. This can be NULL if the satellite isn't important and is ignored for a non-satellite tuner.

Returns

TRUE if the transport is of the given type, FALSE otherwise

4.21.2.94 BOOLEAN DBDEF_TransportInProfile (ADB_TRANSPORT_REC * *t_ptr*)

Checks whether the given transport is valid for the current profile. There may be multiple CI+ profiles and so data related to the CAM is also checked.

Parameters

<i>t_ptr</i>	transport to be checked
--------------	-------------------------

Returns

TRUE if the transport isn't part of a network or is in the current profile, FALSE otherwise

4.21.2.95 void DBDEF_UpdateCridEitDate (ADB_CRID_REC * *c_ptr*)

Updates the stored EIT date of this CRID with the current GMT date.

Parameters

<code>c_ptr</code>	pointer to CRID record
--------------------	------------------------

4.22 dvb/src/ap_events.h File Reference

```
#include "app.h" Include dependency graph for ap_events.h:
```

Data Structures

- struct [S_EVENT_INFO](#)

Functions

- **BOOLEAN AEV_Initialise** ([DVB_EVENT_HANDLER](#) event_handler)
- **void AEV_Terminate** (void)

4.22.1 Detailed Description

Date

March 2011

Author

Steve Ford

4.23 dvb/src/ap_state.h File Reference

Functions

- **void ASTE_Initialise** (void)
- **void ASTE_EnterStandby** (BOOLEAN recording)
- **void ASTE_LeaveStandby** (void)
- **BOOLEAN ASTE_InStandby** (void)
- **void ASTE_StartRecording** (void)
- **void ASTE_StartServiceSearch** (void)
- **void ASTE_StartSSUSearch** (void)
- **void ASTE_SearchComplete** (void)

4.23.1 Detailed Description

Date

July 2011

Author

Steve Ford

4.24 dvb/src/ap_uiinfo.h File Reference

Definition of the UI information callback function to be used within dvb.

Functions

- **BOOLEAN ACB_GetUIInformation** ([S_ACB_UI_INFO](#) *data)

4.24.1 Detailed Description

Definition of the UI information callback function to be used within dvb.

Date

4/12/2012

Author

Sergio Panseri

4.25 dvb/src/ciplus_support.h File Reference

CIPLUS support functions.

```
#include "techtype.h" #include "ci_plus_control.h" Include dependency graph for ciplus_support.h:
```

Data Structures

- struct [s_ciplus_tune_transport](#)
- struct [s_ciplus_tune_service](#)
- struct [S_CIPLUS_TUNE_DATA](#)

Enumerations

- enum **E_CIPLUS_TUNE_TYPE** { **CIPLUS_TUNE_TYPE_SERVICE**, **CIPLUS_TUNE_TYPE_TRANSPORT**, **CIPLUS_TUNE_TYPE_CI_SERVICE** }

Functions

- **U32BIT CIPLUS_CICallback** ([S_CIP_QUERY_PARAM](#) *param)

4.25.1 Detailed Description

CIPLUS support functions.

Date

26th February 2010

Author

Willis Lau

4.26 externals/HBBTV/src/common.h File Reference

Definition of functions common withing externals/HBBTV.

Functions

- void [EXT_HbbtvGetEvent](#) (void *event, void *service, S_HBBTV_EVENT_DETAILS *details)
Fills in the event details structure given the event handle.
- U32BIT [EXT_HbbtvPVRGetRecID](#) (U32BIT timer, U32BIT recording)
Returns the recording ID associated with the specified timer handle and recording handle.
- void [EXT_HbbtvPVRInitialise](#) (void)
Initialises the PVR resources needed by the HbbTV external interface.
- void [EXT_HbbtvPVRTerminate](#) (void)
Releases the PVR resources allocated by the HbbTV external interface.
- void [EXT_HbbtvNWInitialise](#) (void)
Initialises the Network resources needed by the HbbTV external interface.
- void [EXT_HbbtvNWTerminate](#) (void)
Releases the Network resources allocated by the HbbTV external interface.
- void [EXT_HbbtvServiceInitialise](#) (void)
Initialises the Service resources needed by the HbbTV external interface.
- void [EXT_HbbtvServiceTerminate](#) (void)
Releases the Service resources allocated by the HbbTV external interface.

4.26.1 Detailed Description

Definition of functions common withing externals/HBBTV.

Date

February 2014

Author

Sergio Panseri

4.26.2 Function Documentation

4.26.2.1 void **EXT_HbbtvGetEvent** (void * *event*, void * *service*,
S_HBBTV_EVENT_DETAILS * *details*)

Fills in the event details structure given the event handle.

Parameters

<i>event</i>	Event handle
<i>service</i>	Service handle
<i>details</i>	Pointer to the structure to fill in. HBBTV_ReleaseEventDetails can be called passing the structure pointer to ensure any allocated data relative to the event is freed

4.26.2.2 U32BIT **EXT_HbbtvPVRGetRecID** (U32BIT *timer*, U32BIT *recording*)

Returns the recording ID associated with the specified timer handle and recording handle.

Parameters

<i>timer</i>	Timer handle
<i>recording</i>	Recording handle

Returns

Recording ID

4.27 externals/MHEG5/src/dvbmh_int.h File Reference

Header internal to DVB MHEG interface files.

Functions

- void **DVBMH_PlayAudio** (U8BIT path, U16BIT audio_pid, E_STB_DP_AUDIO_-CODEC codec, U16BIT pcr_pid, U16BIT ad_pid)
- void **DVBMH_PlayVideo** (U8BIT path, U16BIT video_pid, E_STB_DP_VIDEO_-CODEC codec, U16BIT pcr_pid)

4.27.1 Detailed Description

Header internal to DVB MHEG interface files.

Date

12/03/2014

Author

Adam Sturtridge

4.28 inc/dbgfuncs.h File Reference

Debug functions header file.

Defines

- #define **FUNCTION_START**(x)
- #define **FUNCTION_FINISH**(x)
- #define **ASSERT**(assertion)
- #define **DBGPRINT**(...)

4.28.1 Detailed Description

Debug functions header file.

Date

01/12/2004

Author

Ocean Blue

4.28.2 Define Documentation

4.28.2.1 #define **DBGPRINT**(...)

DEBUG_PRINTING_ENABLED

4.29 inc/dbgmemory.h File Reference

Macro definition for memory debug.

Defines

- #define **DBG_CHECK_APPHEAP**()
- #define **DBG_CHECK_STBHEAP**()

4.29.1 Detailed Description

Macro definition for memory debug.

Date

January 2014

Author

Sergio Panseri

4.30 inc/techtype.h File Reference

System Wide Global Technical Data Type Definitions.

This graph shows which files directly or indirectly include this file:

Defines

- `#define USE_UNWANTED_PARAM(param) (void)(param)`
- `#define FALSE 0`
- `#define TRUE 1`
- `#define NULL 0`
- `#define NULL_PTR ((void *)NULL)`

Typedefs

- `typedef unsigned char U8BIT`
- `typedef unsigned short U16BIT`
- `typedef signed char S8BIT`
- `typedef signed short S16BIT`
- `typedef unsigned long U32BIT`
- `typedef signed long S32BIT`
- `typedef U8BIT BOOLEAN`

4.30.1 Detailed Description

System Wide Global Technical Data Type Definitions.

Date

August 2006

Author

Paul Marshall

4.31 middleware/CA/inc/ca_glue.h File Reference

Glue layer between DVB and conditional access systems.

Enumerations

- enum **E_CA_DECODE_STATUS** { **CA_DECODE_STATUS_STARTING**, **CA_DECODE_STATUS_STARTED**, **CA_DECODE_STATUS_STOPPED** }

Functions

- **BOOLEAN STB_CAInitialise** (void)
Called once on system startup to allow initialisation of the CA systems.
- **BOOLEAN STB_CAAcquireDescrambler** (U8BIT demux, U16BIT *ca_ids, U16BIT num_ca_ids, U32BIT *handle)
This function is used by the resource manager to acquire a CA descrambler that's able to descramble a service that uses one of the CA systems defined by the array of CA system IDs (ca_ids). If a descrambler is available then a handle should be returned in 'handle' which will be used in all future calls related to this descrambler. If the CA software needs to set the demux descrambling keys, or create any filters to monitor SI data, the given demux handle should be used.
- **BOOLEAN STB_CAReleaseDescrambler** (U32BIT handle)
Will be called when a CA descrambler is no longer required.
- **void STB_CADescrambleServiceStart** (U32BIT handle)
This function will be called when decoding of a service is about to start and there's an associated descrambler.
- **void STB_CADescrambleServiceStop** (U32BIT handle)
This function will be called when decoding of a service is stopped.
- **void STB_CAReportPMT** (U32BIT handle, U8BIT *pmt_data, U16BIT data_len)
When there's an update to the PMT for a service, the updated PMT will be reported to the CA system using this function.
- **void STB_CAReportCAT** (U32BIT handle, U8BIT *cat_data, U16BIT data_len)
When there's an update to the CAT for a service, the updated CAT will be reported to the CA system using this function. The data is provided a section at a time, rather than as a complete table.
- **void STB_CAReportBAT** (U32BIT handle, U8BIT *bat_data, U16BIT data_len)
When there's an update to the BAT, the updated BAT will be reported to the CA system using this function. The data is provided a section at a time, rather than as a complete table.
- **void STB_CAReportNIT** (U32BIT handle, U8BIT *nit_data, U16BIT data_len)
When there's an update to the NIT, the updated NIT will be reported to the CA system using this function. The data is provided a section at a time, rather than as a complete table.
- **void STB_CADecodeVideoStatus** (U32BIT handle, E_CA_DECODE_STATUS decode_status)

Notifies the CA system of a change in the video decoding state.

- void [STB_CADecodeAudioStatus](#) (U32BIT handle, E_CA_DECODE_STATUS decode_status)

Notifies the CA system of a change in the audio decoding state.

- void [STB_CADecodeADStatus](#) (U32BIT handle, E_CA_DECODE_STATUS decode_status)

Notifies the CA system of a change in the AD decoding state.

- void [STB_CANotifyRunningStatus](#) (U32BIT handle, U8BIT status)

This function will be called when there's a change to the running status of a service being descrambled as indicated by the running_status field in the SDT.

- BOOLEAN [STB_CADescramblerRequiredForPlayback](#) (U16BIT *ca_ids, U16BIT num_ca_ids)

This function works out whether a CA descrambler is required to playback a recording with one of the given CA system IDs.

- BOOLEAN [STB_CADescramblerRequiredForRecording](#) (U16BIT *ca_ids, U16BIT num_ca_ids)

This function works out whether a CA descrambler is required to record a service with one of the given CA system IDs.

- U16BIT [STB_CAGetRecordingPids](#) (U8BIT *pmt_data, U16BIT **pid_array)

This function is called to get an array of PIDs that need to be recorded for the CA system required for the given PMT. The array must be allocated by this function, which also returns the number of items in the array.

- void [STB_CAReleaseRecordingPids](#) (U16BIT *pid_array, U16BIT num_pids)

Called to free the array of PIDs allocated by STB_CAGetRecordingPids.

- void [STB_CANotifyRecordingStatus](#) (U32BIT handle, BOOLEAN status)

This function is called when a recording starts and when it stops.

4.31.1 Detailed Description

Glue layer between DVB and conditional access systems.

Date

10/09/2013

Author

Steve Ford

4.31.2 Function Documentation

4.31.2.1 BOOLEAN STB_CAAcquireDescrambler (U8BIT demux, U16BIT * ca_ids, U16BIT num_ca_ids, U32BIT * handle)

This function is used by the resource manager to acquire a CA descrambler that's able to descramble a service that uses one of the CA systems defined by the array of CA

system IDs (*ca_ids*). If a descrambler is available then a handle should be returned in 'handle' which will be used in all future calls related to this descrambler. If the CA software needs to set the demux descrambling keys, or create any filters to monitor SI data, the given demux handle should be used.

Parameters

<i>demux</i>	- demux to be used if a descrambler is acquired
<i>ca_ids</i>	- array of CA system IDs for the service
<i>num_ca_ids</i>	- number of CA system IDs in the array
<i>handle</i>	- pointer to return a handle to identify the acquired CA descrambler

Returns

TRUE if a descrambler is acquired, FALSE otherwise

4.31.2.2 void STB_CADecodeADStatus (U32BIT *handle*, E_CA_DECODE_STATUS *decode_status*)

Notifies the CA system of a change in the AD decoding state.

Parameters

<i>handle</i>	- CA descrambler handle
<i>decode_status</i>	- decoding status

4.31.2.3 void STB_CADecodeAudioStatus (U32BIT *handle*, E_CA_DECODE_STATUS *decode_status*)

Notifies the CA system of a change in the audio decoding state.

Parameters

<i>handle</i>	- CA descrambler handle
<i>decode_status</i>	- decoding status

4.31.2.4 void STB_CADecodeVideoStatus (U32BIT *handle*, E_CA_DECODE_STATUS *decode_status*)

Notifies the CA system of a change in the video decoding state.

Parameters

<i>handle</i>	- CA descrambler handle
<i>decode_</i> <i>status</i>	- decoding status

4.31.2.5 **BOOLEAN STB_CADescramblerRequiredForPlayback** (U16BIT * *ca_ids*, U16BIT *num_ca_ids*)

This function works out whether a CA descrambler is required to playback a recording with one of the given CA system IDs.

Parameters

<i>ca_ids</i>	- array of CA system IDs
<i>num_ca_ids</i>	- number of CA system IDs in the array

Returns

TRUE if a CA descrambler is required, FALSE otherwise

4.31.2.6 **BOOLEAN STB_CADescramblerRequiredForRecording** (U16BIT * *ca_ids*, U16BIT *num_ca_ids*)

This function works out whether a CA descrambler is required to record a service with one of the given CA system IDs.

Parameters

<i>ca_ids</i>	- array of CA system IDs
<i>num_ca_ids</i>	- number of CA system IDs in the array

Returns

TRUE if a CA descrambler is required, FALSE otherwise

4.31.2.7 **void STB_CADescrambleServiceStart** (U32BIT *handle*)

This function will be called when decoding of a service is about to start and there's an associated descrambler.

Parameters

<i>handle</i>	- CA descrambler handle
---------------	-------------------------

4.31.2.8 void STB_CADescrambleServiceStop (U32BIT *handle*)

This function will be called when decoding of a service is stopped.

Parameters

<i>handle</i>	- CA descrambler handle
---------------	-------------------------

4.31.2.9 U16BIT STB_CAGetRecordingPids (U8BIT * *pmt_data*, U16BIT ** *pid_array*)

This function is called to get an array of PIDs that need to be recorded for the CA system required for the given PMT. The array must be allocated by this function, which also returns the number of items in the array.

Parameters

<i>pmt_data</i>	- raw PMT section data
<i>pid_array</i>	- pointer to an array allocated by this function on return, containing the PIDs to be recorded

Returns

the number of PIDs in the returned array

4.31.2.10 BOOLEAN STB_CAIinitialise (void)

Called once on system startup to allow initialisation of the CA systems.

Returns

TRUE if initialisation is successful, FALSE otherwise

4.31.2.11 void STB_CANotifyRecordingStatus (U32BIT *handle*, BOOLEAN *status*)

This function is called when a recording starts and when it stops.

Parameters

<i>handle</i>	- CA descrambler handle
<i>status</i>	- TRUE when a recording starts, FALSE when it stops

4.31.2.12 void STB_CANotifyRunningStatus (U32BIT *handle*, U8BIT *status*)

This function will be called when there's a change to the running status of a service being descrambled as indicated by the running_status field in the SDT.

Parameters

<i>handle</i>	- CA descrambler handle
<i>status</i>	- running status as defined in the SDT

4.31.2.13 BOOLEAN STB_CAReleaseDescrambler (U32BIT *handle*)

Will be called when a CA descrambler is no longer required.

Parameters

<i>handle</i>	- CA descrambler handle being released
---------------	--

Returns

TRUE if the descrambler is released, FALSE otherwise

4.31.2.14 void STB_CAReleaseRecordingPids (U16BIT * *pid_array*, U16BIT *num_pids*)

Called to free the array of PIDs allocated by STB_CAGetRecordingPids.

Parameters

<i>pid_array</i>	- array of PIDs to be freed
<i>num_pids</i>	- number of PIDs in the array

4.31.2.15 void STB_CAReportBAT (U32BIT *handle*, U8BIT * *bat_data*, U16BIT *data_len*)

When there's an update to the BAT, the updated BAT will be reported to the CA system using this function. The data is provided a section at a time, rather than as a complete table.

Parameters

<i>handle</i>	- CA descrambler handle
<i>bat_data</i>	- raw BAT section data
<i>data_len</i>	- number of bytes in the BAT section

4.31.2.16 void STB_CAReportCAT (U32BIT *handle*, U8BIT * *cat_data*, U16BIT *data_len*)

When there's an update to the CAT for a service, the updated CAT will be reported to the CA system using this function. The data is provided a section at a time, rather than as a complete table.

Parameters

<i>handle</i>	- CA descrambler handle
<i>cat_data</i>	- raw CAT section data
<i>data_len</i>	- number of bytes in the CAT section

4.31.2.17 void STB_CAReportNIT (U32BIT *handle*, U8BIT * *nit_data*, U16BIT *data_len*)

When there's an update to the NIT, the updated NIT will be reported to the CA system using this function. The data is provided a section at a time, rather than as a complete table.

Parameters

<i>handle</i>	- CA descrambler handle
<i>nit_data</i>	- raw NIT section data
<i>data_len</i>	- number of bytes in the NIT section

4.31.2.18 void STB_CAReportPMT (U32BIT *handle*, U8BIT * *pmt_data*, U16BIT *data_len*)

When there's an update to the PMT for a service, the updated PMT will be reported to the CA system using this function.

Parameters

<i>handle</i>	- CA descrambler handle
<i>pmt_data</i>	- raw PMT section data
<i>data_len</i>	- number of bytes in the PMT

4.32 middleware/CI/inc/ci_plus_control.h File Reference

CIPLUS Glue: Control functions.

```
#include "techtype.h" #include "stbhwc.h" #include "stbci.-
h" #include "stbsitab.h" Include dependency graph for ci_plus_control.h:
This graph shows which files directly or indirectly include this file:
```

Data Structures

- struct [ci_tune_service_info](#)
- struct [ci_tune_del_sys_desc](#)
- struct [ci_operator_nit](#)
- struct [ci_host_info](#)
- struct [ci_operator_info](#)
- struct [ci_operator_search](#)
- struct [ci_licence_update](#)
- struct [ci_query_param](#)
- struct [S_CIP_RELEASE_REPLY](#)

Defines

- #define **CIP_NO_CI_SCREEN** 0
- #define **CIP_CI_MENU_LIST_SCREEN** 1
- #define **CIP_CI_ENQUIRY_SCREEN** 2
- #define **CIP_URI_LEN** 8
- #define **CIP_CICAM_ID_LEN** 8
- #define **CIP_MAX_PIN_LENGTH** 8
- #define **CIP_PIN_PRIVATE_DATA_SIZE** 15
- #define **CIP_SERVICE_TYPE_TV** 0x00000001 /* MPEG2 TV */
- #define **CIP_SERVICE_TYPE_RADIO** 0x00000002 /* MPEG1 Layer-II radio */
- #define **CIP_SERVICE_TYPE_TELETEXT** 0x00000004 /* Teletext */
- #define **CIP_SERVICE_TYPE_AVC_RADIO** 0x00000008 /* AVC radio */
- #define **CIP_SERVICE_TYPE_DATA** 0x00000010 /* Data */
- #define **CIP_SERVICE_TYPE_HD_TV** 0x00000020 /* MPEG2 HD TV */
- #define **CIP_SERVICE_TYPE_AVC_SD_TV** 0x00000040 /* AVC SD TV */
- #define **CIP_SERVICE_TYPE_AVC_HD_TV** 0x00000080 /* AVC HD TV */
- #define **CIP_DELIVERY_TYPE_DVBT** 0x0001
- #define **CIP_DELIVERY_TYPE_DVBT2** 0x0002
- #define **CIP_DELIVERY_TYPE_DVBC** 0x0004
- #define **CIP_DELIVERY_TYPE_DVBC2** 0x0008
- #define **CIP_DELIVERY_TYPE_DVBS** 0x0010
- #define **CIP_DELIVERY_TYPE_DVBS2** 0x0020
- #define **CIP_APP_TYPE_MHEG** 0x0001
- #define **CIP_APP_TYPE_HBBTV** 0x0002
- #define **CIP_APP_TYPE_OIPF** 0x0004
- #define **CIP_START_OPERATOR_SEARCH_NO** 0x00
- #define **CIP_START_OPERATOR_SEARCH_YES** 0x01
- #define **CIP_START_OPERATOR_SEARCH_ASK** 0x02
- #define **CIP_RECORD_MODE_WATCH_AND_BUFFER** 0x00
- #define **CIP_RECORD_MODE_TIMESHIFT** 0x01
- #define **CIP_RECORD_MODE_UNATTENDED** 0x02
- #define **CIP_LICENCE_STATUS_OK** 0x00
- #define **CIP_LICENCE_STATUS_NO_ENTITLEMENT** 0x01

- `#define CIP_LICENCE_STATUS_UNDEFINED_ERROR 0x02`
- `#define CIP_LICENCE_STATUS_RIGHTS_EXPIRED 0x03`
- `#define CIP_LICENCE_STATUS_PLAY_COUNT_EXCEEDED 0x04`
- `#define CIP_LICENCE_STATUS_RETENTION_LIMIT_EXCEEDED 0x05`
- `#define CIP_LICENCE_STATUS_INVALID_LICENCE 0x06`
- `#define CIP_LICENCE_RECEIVED_OK 0x00`
- `#define CIP_LICENCE_RECEIVED_INVALID_DATA 0x01`
- `#define CIP_LICENCE_RECEIVED_HOST_ERROR 0x02`
- `#define CIP_CAM_PIN_CODE_WRONG 0`
- `#define CIP_CAM_PIN_CAM_BUSY 1`
- `#define CIP_CAM_PIN_CODE_CORRECT 2`
- `#define CIP_CAM_PIN_CODE_UNCONFIRMED 3`
- `#define CIP_CAM_PIN_NO_VIDEO_BLANK 4`
- `#define CIP_CAM_PIN_CONTENT_STILL_SCRAMBLED 5`

Typedefs

- `typedef struct ci_tune_service_info S_CIP_TUNE_SERVICE_INFO`
- `typedef struct ci_tune_del_sys_desc S_CIP_TUNE_DEL_SYS_DESC`
- `typedef struct ci_operator_nit S_CIP_OPERATOR_NIT`
- `typedef struct ci_host_info S_CIP_HOST_INFO`
- `typedef struct ci_operator_info S_CIP_OPERATOR_INFO`
- `typedef struct ci_operator_search S_CIP_OPERATOR_SEARCH`
- `typedef struct ci_licence_update S_CIP_LICENCE_UPDATE`
- `typedef struct ci_query_param S_CIP_QUERY_PARAM`
- `typedef U32BIT(* F_CIP_CALLBACK)(S_CIP_QUERY_PARAM *)`

Enumerations

- `enum E_CI_QUERY { CIPLUS_GET_COUNTRY_CODE, CIPLUS_GET_LANGUAGE_CODE, CIPLUS_TUNE_TO_SERVICE, CIPLUS_TUNE_DEL_SYS_DESC, CIPLUS_CAM_UPGRADE_OPTION, CIPLUS_REQUEST_HOST_CONTROL, CIPLUS_RELEASE_HOST_CONTROL, CIPLUS_PROCESS_NIT, × CIPLUS_REQUEST_HOST_INFO, CIPLUS_OPERATOR_TUNE, CIPLUS_REQUEST_OPERATOR_SEARCH, CIPLUS_OPERATOR_CAM_AVAILABLE, × CIPLUS_OPERATOR_CAM_REMOVED, CIPLUS_GET_SUBTITLES_START, CIPLUS_STOP_SUBTITLES, CIPLUS_START_SUBTITLES }`
- `enum E_CIP_SLOT_STATUS { CIP_SLOT_STATUS_UNKNOWN = 0x0000, CIP_SLOT_STATUS_MODULE_READY = 0x0001, CIP_SLOT_STATUS_CAS_USED = 0x0002, CIP_SLOT_STATUS_CAS_SUPPORTED = 0x0004, CIP_SLOT_STATUS_CI_PLUS_MODULE = 0x0008, CIP_SLOT_STATUS_SERVICE_ALLOWED = 0x0010 }`
- `enum E_CIP_TUNE_STATUS { CIP_TUNER_LOCKED, CIP_TUNER_NOTLOCKED, CIP_TUNER_BUSY, CIP_TUNER_UNSUPPORTED_SYSTEM, CIP_TUNER_BAD_PARAM, CIP_TUNER_SERVICE_NOT_FOUND, CIP_TUNER_UNDEFINED_ERROR }`

- enum **E_CIP_OPERATOR_SEARCH_REQUEST_TYPE** { CIP_OPERATOR_SEARCH_DEFERRED, CIP_OPERATOR_SEARCH_IMMEDIATE, CIP_OPERATOR_SEARCH_SCHEDULED }

Functions

- void **CIP_OpenEngine** (F_CIP_CALLBACK callback)
Open CIPLUS engine. Must be called before any other CIPLUS related function.
- void **CIP_EnterCIMenu** (U8BIT slot_id)
Enter CI menu.
- U8BIT * **CIP_GetCIModuleName** (U8BIT slot_id)
Returns a pointer to the string containing the name of the module.
- U8BIT **CIP_GetCIScreenType** (U8BIT slot_id)
Returns the screen type required, and also copies the required data into the current data for access by the application.
- U8BIT * **CIP_GetCIMenuScreenTitle** (U8BIT slot_id)
Return a pointer to the menu/list screen title string.
- U8BIT * **CIP_GetCIMenuScreenSubtitle** (U8BIT slot_id)
Returns a pointer to the menu/list screen subtitle string.
- U8BIT **CIP_GetCIMenuScreenNumItems** (U8BIT slot_id)
Return the number of items on the menu/list screen.
- U8BIT * **CIP_GetCIMenuScreenItemText** (U8BIT slot_id, U8BIT item)
Return a pointer to the string for the specified item on the menu/list screen.
- U8BIT * **CIP_GetCIMenuScreenBottomText** (U8BIT slot_id)
Returns a pointer to the menu/list screen bottom text string.
- void **CIP_SetCIMenuScreenResponse** (U8BIT slot_id, U8BIT response)
Send the response to the menu screen to the module.
- U8BIT * **CIP_GetCIEnquiryScreenText** (U8BIT slot_id)
Returns a pointer to the enquiry screen text string.
- U8BIT **CIP_GetCIEnqScreenHideResponse** (U8BIT slot_id)
Returns a value indicating whether the user response text should be hidden (e.g. pin number) or not.
- U8BIT **CIP_GetCIEnqScreenResponseLength** (U8BIT slot_id)
Return the length of the required user response text.
- void **CIP_SetCIEnquiryScreenResponse** (U8BIT slot_id, U8BIT ok_cancel, U8BIT *response_str_ptr)
sends a message to the CI+ stack to indicate the response to the enquiry screen
- U8BIT **CIP_GetCIScreenCloseDelay** (U8BIT slot_id)
Return the delay required before closing the CI screen.
- void **CIP_CloseCIScreen** (U8BIT slot_id)
Called by the application to close the MMI session.
- void **CIP_SetHostCountryCode** (U32BIT code)
Set the Host Country code.
- void **CIP_SetHostLanguageCode** (U32BIT code)

Set the Host Language code.

- void [CIP_SetDisplayCharacterTables](#) (U8BIT *tables, U16BIT len)
Sets the display character tables supported by the receiver. The tables are provided as a single string containing the prefix byte(s) of encoded text strings as defined in annex A.2 of ETSI EN 300 468. For example, if ISO/IEC 8859-7 is supported by the receiver, then text strings starting with 0x03 and with 0x10 0x00 0x07 are supported. In this case the string would be "\x03\x10\x00\x07".
- void [CIP_SetInputCharacterTables](#) (U8BIT *tables, U16BIT len)
Sets the input character tables supported by the receiver. The tables are provided as a single string containing the prefix byte(s) of encoded text strings as defined in annex A.2 of ETSI EN 300 468. For example, if ISO/IEC 8859-7 is supported by the receiver, then text strings starting with 0x03 and with 0x10 0x00 0x07 are supported. In this case the string would be "\x03\x10\x00\x07".
- U16BIT [CIP_GetSlotStatus](#) (U8BIT slot_id, U8BIT *pmt, U8BIT *ci_protection_descriptor)
Return status of CI slot.
- BOOLEAN [CIP_IsConditionalAccessUsed](#) (U8BIT *pmt)
Tell whether conditional access is used in the service.
- void [CIP_ReportPmt](#) (U8BIT slot_id, U8BIT *pmt)
Report PMT to CI stack.
- void [CIP_GetUsageRulesInfo](#) (U8BIT slot_id, U16BIT service_id, U8BIT uri[CIP_URI_LEN])
Return the current URI for the given service.
- void [CIP_GetDefaultUsageRulesInfo](#) (U8BIT raw_uri[CIP_URI_LEN])
Sets the default URI values into the given array. The URI requires HDCP.
- void [CIP_UpdateUsageRules](#) (U8BIT orig_uri[CIP_URI_LEN], U8BIT new_uri[CIP_URI_LEN])
Update packed URI given a new packed URI.
- U32BIT [CIP_GetRetentionLimit](#) (U8BIT uri[CIP_URI_LEN])
Return the retention limit given a packed URI.
- void [CIP_ApplyUsageRulesInfo](#) (U8BIT *uri)
Apply the given Usage Rules Information.
- void [CIP_ApplyUsageRulesInfoForPlayback](#) (U8BIT *uri)
Apply the given Usage Rules Information for playback.
- void [CIP_ApplyCCKeys](#) (U8BIT path)
Apply stored CC keys (if any) on given path.
- void [CIP_ClearCCKeys](#) (U8BIT path)
Clear CC keys (if any) from given path.
- BOOLEAN [CIP_IsHDCPRequired](#) (U16BIT service_id)
Tell whether the given service requires HDCP.
- void [CIP_AppStopped](#) (void)
Should be called to notify CI+ that the running MHEG app has been stopped.
- U8BIT [CIP_GetSlotIdForModule](#) (U32BIT module)
Returns the slot id for the given module handle.

- **BOOLEAN CIP_TuneReply** (U8BIT path, U32BIT module, E_CIP_TUNE_STAT-US status)
Called by the host to send the status of the tune operation to the module.
- **BOOLEAN CIP_AskRelease** (U32BIT module)
Called by the host to ask the module to restore replaced PIDs and close the host control session.
- **BOOLEAN CIP_RequestOperatorStatus** (U32BIT module)
Called by the host to request changing to an operator profile.
- **BOOLEAN CIP_OperatorExit** (void)
Request the current operator module to exit operator profile.
- **BOOLEAN CIP_StartOperatorSearch** (U32BIT module)
Called by the app to start an operator profile search that has been requested.
- **void CIP_SetRecordOperatingMode** (U8BIT slot_id, U8BIT mode, U16BIT service_id)
Sets the record operating mode for the given slot id.
- **BOOLEAN CIP_SendRecordStart** (U8BIT slot_id, U16BIT program_number, U8-BIT *pin_string)
Called by the app when a recording is to be started on a CA protected service.
- **BOOLEAN CIP_SendRecordStop** (U8BIT slot_id)
Called by the app when a recording is stopped or completes.
- **BOOLEAN CIP_SendPlaybackLicence** (U8BIT slot_id, U16BIT program_number, U8BIT *licence, U16BIT licence_len)
Sends a CICAM licence to a module during playback, which will result in a modified licence being notified through STB_CINotifyCicamLicence.
- **U8BIT * CIP_GetRecordingLicence** (U8BIT slot_id, U8BIT *licence_status, U16-BIT *licence_len, U8BIT raw_uri[CIP_URI_LEN])
Returns the last licence received from the CAM when recording.
- **U8BIT * CIP_GetPlaybackLicence** (U8BIT slot_id, U8BIT *licence_status, U16-BIT *licence_len, U8BIT raw_uri[CIP_URI_LEN])
Returns the last licence received from the CAM during playback.
- **void CIP_SendCicamLicenceStatus** (U8BIT slot_id, U32BIT status)
Must be called after a licence has been supplied during recording or playback to inform the CAM that the licence has been handled.
- **BOOLEAN CIP_GetCicamId** (U8BIT slot_id, U8BIT cicam_id[8])
Returns the CICAM identifier for the given slot.
- **U8BIT CIP_FindSlotForCicamId** (U8BIT cicam_id[8])
Checks CAMs in all slots to find the one with the given CAM id.
- **BOOLEAN CIP_CheckCicamPin** (U8BIT slot_id, U8BIT *pin_data)
Called by the host to check whether a CAM pin is valid. An STB_EVENT_CI_PIN_S-TATUS event will be sent to notify the host of the validity, or otherwise, of the pin.
- **BOOLEAN CIP_GetRecordingPinInfo** (U8BIT slot_id, U8BIT *status, U8BIT *age_rating, U8BIT **private_data, U16BIT *date_code, U8BIT *hour, U8BIT *min, U8BIT *sec)
Returns the information to be stored with a pin event when recording.
- **BOOLEAN CIP_SendPinPlayback** (U8BIT slot_id, U8BIT age_rating, U8BIT *private_data)
Sends a pin event to the CAM during playback.

4.32.1 Detailed Description

CIPLUS Glue: Control functions.

Date

9th March 2010

4.32.2 Function Documentation

4.32.2.1 void CIP_ApplyCCKeys (U8BIT *path*)

Apply stored CC keys (if any) on given path.

Parameters

<i>path</i>	- Path to apply keys on
-------------	-------------------------

4.32.2.2 void CIP_ApplyUsageRulesInfo (U8BIT * *uri*)

Apply the given Usage Rules Information.

Parameters

<i>uri</i>	- Usage Rules Information
------------	---------------------------

4.32.2.3 void CIP_ApplyUsageRulesInfoForPlayback (U8BIT * *uri*)

Apply the given Usage Rules Information for playback.

Parameters

<i>uri</i>	- Usage Rules Information
------------	---------------------------

4.32.2.4 BOOLEAN CIP_AskRelease (U32BIT *module*)

Called by the host to ask the module to restore replaced PIDs and close the host control session.

Parameters

<i>module</i>	- host control module
---------------	-----------------------

Returns

TRUE if request sent, FALSE otherwise

4.32.2.5 **BOOLEAN CIP_CheckCicamPin (U8BIT *slot_id*, U8BIT * *pin_data*)**

Called by the host to check whether a CAM pin is valid. An STB_EVENT_CI_PIN_ST-ATUS event will be sent to notify the host of the validity, or otherwise, of the pin.

Parameters

<i>slot_id</i>	- slot
<i>pin_data</i>	- ASCII encoded pin data, null terminated

Returns

TRUE if the pin was sent successfully, FALSE otherwise

4.32.2.6 **void CIP_ClearCCKeys (U8BIT *path*)**

Clear CC keys (if any) from given path.

Parameters

<i>path</i>	- Path to clear keys from
-------------	---------------------------

4.32.2.7 **void CIP_CloseCIScreen (U8BIT *slot_id*)**

Called by the application to close the MMI session.

Parameters

<i>slot_id</i>	- slot ID (0, 1, ...)
----------------	-----------------------

4.32.2.8 **void CIP_EnterCIMenu (U8BIT *slot_id*)**

Enter CI menu.

Parameters

<i>slot_id</i>	- slot ID (0, 1, ...)
----------------	-----------------------

4.32.2.9 **U8BIT CIP_FindSlotForCicamId (U8BIT *cicam_id*[8])**

Checks CAMs in all slots to find the one with the given CAM id.

Parameters

<i>cicam_id</i>	- CAM id to look for
-----------------	----------------------

Returns

slot id if found, INVALID_RES_ID if not found

4.32.2.10 BOOLEAN CIP_GetCicamId (U8BIT *slot_id*, U8BIT *cicam_id*[8])

Returns the CICAM identifier for the given slot.

Parameters

<i>slot_id</i>	- slot
<i>cicam_id</i>	- array of 8 bytes containing the returned CICAM id

Returns

TRUE if the id is returned, FALSE otherwise

4.32.2.11 U8BIT CIP_GetCIEnqScreenHideResponse (U8BIT *slot_id*)

Returns a value indicating whether the user response text should be hidden (e.g. pin number) or not.

Parameters

<i>slot_id</i>	- slot ID (0, 1, ...)
----------------	-----------------------

Returns

1 to indicate text should be hidden, 0 otherwise

4.32.2.12 U8BIT CIP_GetCIEnqScreenResponseLength (U8BIT *slot_id*)

Return the length of the required user response text.

Parameters

<i>slot_id</i>	- slot ID (0, 1, ...)
----------------	-----------------------

Returns

The length of the response text

4.32.2.13 U8BIT* CIP_GetCIEnquiryScreenText (U8BIT *slot_id*)

Returns a pointer to the enquiry screen text string.

Parameters

<i>slot_id</i>	- slot ID (0, 1, ...)
----------------	-----------------------

Returns

The enquiry text string

4.32.2.14 U8BIT* CIP_GetCIMenuScreenBottomText (U8BIT *slot_id*)

Returns a pointer to the menu/list screen bottom text string.

Parameters

<i>slot_id</i>	- slot ID (0, 1, ...)
----------------	-----------------------

Returns

The bottom text string

4.32.2.15 U8BIT* CIP_GetCIMenuScreenItemText (U8BIT *slot_id*, U8BIT *item*)

Return a pointer to the string for the specified item on the menu/list screen.

Parameters

<i>slot_id</i>	- slot ID (0, 1, ...)
----------------	-----------------------

Returns

The specified item on the screen

4.32.2.16 U8BIT CIP_GetCIMenuScreenNumItems (U8BIT *slot_id*)

Return the number of items on the menu/list screen.

Parameters

<i>slot_id</i>	- slot ID (0, 1, ...)
----------------	-----------------------

Returns

The number of items on the screen

4.32.2.17 U8BIT* CIP_GetCIMenuScreenSubtitle (U8BIT *slot_id*)

Returns a pointer to the menu/list screen subtitle string.

Parameters

<i>slot_id</i>	- slot ID (0, 1, ...)
----------------	-----------------------

Returns

The screen subtitle

4.32.2.18 U8BIT* CIP_GetCIMenuScreenTitle (U8BIT *slot_id*)

Return a pointer to the menu/list screen title string.

Parameters

<i>slot_id</i>	- slot ID (0, 1, ...)
----------------	-----------------------

Returns

The screen title

4.32.2.19 U8BIT* CIP_GetCIModuleName (U8BIT *slot_id*)

Returns a pointer to the string containing the name of the module.

Parameters

<i>slot_id</i>	- slot ID (0, 1, ...)
----------------	-----------------------

Returns

A pointer to the module name

4.32.2.20 U8BIT CIP_GetCIScreenCloseDelay (U8BIT *slot_id*)

Return the delay required before closing the CI screen.

Parameters

<i>slot_id</i>	- slot ID (0, 1, ...)
----------------	-----------------------

Returns

The delay in seconds before closing the screen

4.32.2.21 U8BIT CIP_GetCIScreenType (U8BIT *slot_id*)

Returns the screen type required, and also copies the required data into the current data for access by the application.

Parameters

<i>slot_id</i>	- slot ID (0, 1, ...)
----------------	-----------------------

Returns

The screen type

4.32.2.22 void CIP_GetDefaultUsageRulesInfo (U8BIT *raw_uri*[CIP_URI_LEN])

Sets the default URI values into the given array. The URI requires HDCP.

Parameters

<i>raw_uri</i>	- returns with the default URI settings
----------------	---

**4.32.2.23 U8BIT* CIP_GetPlaybackLicence (U8BIT *slot_id*, U8BIT * *licence_status*,
U16BIT * *licence_len*, U8BIT *raw_uri*[CIP_URI_LEN])**

Returns the last licence received from the CAM during playback.

Parameters

<i>slot_id</i>	- slot
<i>licence_status</i>	- status value associated with the licence
<i>licence_len</i>	- pointer to return the licence length in bytes
<i>raw_uri</i>	- returns with the packed URI data associated with the licence

Returns

pointer to the licence data, mustn't be freed. NULL if no licence

**4.32.2.24 U8BIT* CIP_GetRecordingLicence (U8BIT *slot_id*, U8BIT * *licence_status*,
U16BIT * *licence_len*, U8BIT *raw_uri*[CIP_URI_LEN])**

Returns the last licence received from the CAM when recording.

Parameters

<i>slot_id</i>	- slot
<i>licence_status</i>	- status value associated with the licence
<i>licence_len</i>	- pointer to return the licence length in bytes
<i>raw_uri</i>	- returns with the packed URI data associated with the licence

Returns

pointer to the licence data, mustn't be freed. NULL if no licence

4.32.2.25 `BOOLEAN CIP_GetRecordingPinInfo (U8BIT slot_id, U8BIT * status, U8BIT * age_rating, U8BIT ** private_data, U16BIT * date_code, U8BIT * hour, U8BIT * min, U8BIT * sec)`

Returns the information to be stored with a pin event when recording.

Parameters

<i>slot_id</i>	- slot being used for the recording
<i>status</i>	- pin status code
<i>age_rating</i>	- returns the age rating supplied with the pin event
<i>private_data</i>	- returns a pointer to the private data to be saved - don't free
<i>date_code</i>	- MJD UTC date code when the pin is to be applied
<i>hour</i>	- UTC hour when the pin is to be applied
<i>min</i>	- UTC minute when the pin is to be applied
<i>sec</i>	- UTC second when the pin is to be applied

Returns

TRUE if the info is returned, FALSE otherwise

4.32.2.26 `U32BIT CIP_GetRetentionLimit (U8BIT uri[CIP_URI_LEN])`

Return the retention limit given a packed URI.

Parameters

<i>uri</i>	- Usage Rules Information
------------	---------------------------

Returns

Retention limit in minutes (0 = no limit)

4.32.2.27 `U8BIT CIP_GetSlotIdForModule (U32BIT module)`

Returns the slot id for the given module handle.

Parameters

<i>module</i>	- module ID
---------------	-------------

Returns

slot id, INVALID_RES_ID if module isn't valid

4.32.2.28 U16BIT CIP_GetSlotStatus (U8BIT *slot_id*, U8BIT * *pmt*, U8BIT * *ci_protection_descriptor*)

Return status of CI slot.

Parameters

<i>slot_id</i>	- slot ID
<i>pmt</i>	- PMT for the service (section)
<i>ci_protection_descriptor</i>	- ci_protection_descriptor for the service

Returns

Slot status (bitmask)

4.32.2.29 void CIP_GetUsageRulesInfo (U8BIT *slot_id*, U16BIT *service_id*, U8BIT *uri*[CIP_URI_LEN])

Return the current URI for the given service.

Parameters

<i>slot_id</i>	- slot ID
<i>service_id</i>	- service ID
<i>uri</i>	- the URI for the service

Returns

URI for the service if available or "empty" URI otherwise

4.32.2.30 BOOLEAN CIP_IsConditionalAccessUsed (U8BIT * *pmt*)

Tell whether conditional access is used in the service.

Parameters

<i>pmt</i>	- PMT for the service (section)
------------	---------------------------------

Returns

TRUE if conditional access is used, FALSE otherwise

4.32.2.31 BOOLEAN CIP_IsHDCPRequired (U16BIT *service_id*)

Tell whether the given service requires HDCP.

Parameters

<i>service_id</i>	- Service ID
-------------------	--------------

4.32.2.32 **BOOLEAN CIP_OperatorExit (void)**

Request the current operator module to exit operator profile.

Returns

TRUE if request succeeded, FALSE otherwise

4.32.2.33 **void CIP_ReportPmt (U8BIT *slot_id*, U8BIT * *pmt*)**

Report PMT to CI stack.

Parameters

<i>slot_id</i>	- slot ID (0, 1, ...)
<i>pmt</i>	- PMT for the service (section)

4.32.2.34 **BOOLEAN CIP_RequestOperatorStatus (U32BIT *module*)**

Called by the host to request changing to an operator profile.

Parameters

<i>module</i>	- operator profile module
---------------	---------------------------

Returns

TRUE if request succeeded, FALSE otherwise

4.32.2.35 **void CIP_SendCicamLicenceStatus (U8BIT *slot_id*, U32BIT *status*)**

Must be called after a licence has been supplied during recording or playback to inform the CAM that the licence has been handled.

Parameters

<i>slot_id</i>	- slot
<i>status</i>	- licence status

4.32.2.36 **BOOLEAN CIP_SendPinPlayback (U8BIT *slot_id*, U8BIT *age_rating*, U8BIT * *private_data*)**

Sends a pin event to the CAM during playback.

Parameters

<i>slot_id</i>	- slot
<i>age_rating</i>	- rating as provided by the pin event during recording
<i>private_data</i>	- pin private data as provided by the pin event during recording

Returns

TRUE on success, FALSE otherwise

4.32.2.37 **BOOLEAN CIP_SendPlaybackLicence (U8BIT *slot_id*, U16BIT *program_number*, U8BIT * *licence*, U16BIT *licence_len*)**

Sends a CICAM licence to a module during playback, which will result in a modified licence being notified through STB_CINotifyCicamLicence.

Parameters

<i>slot_id</i>	- slot
<i>program_number</i>	- program number being played back
<i>licence</i>	- CICAM licence
<i>licence_len</i>	- licence length in bytes

Returns

TRUE if operation succeeded, FALSE otherwise

4.32.2.38 **BOOLEAN CIP_SendRecordStart (U8BIT *slot_id*, U16BIT *program_number*, U8BIT * *pin_string*)**

Called by the app when a recording is to be started on a CA protected service.

Parameters

<i>slot_id</i>	- slot being used for the recording
<i>program_number</i>	- service ID
<i>pin_string</i>	- pin as a null terminated ascii string

Returns

TRUE on success, but this doesn't mean the recording can start, FALSE otherwise

4.32.2.39 **BOOLEAN CIP_SendRecordStop (U8BIT *slot_id*)**

Called by the app when a recording is stopped or completes.

Parameters

<i>slot_id</i>	- slot to be, or being, used for recording
----------------	--

Returns

TRUE on success, FALSE otherwise

4.32.2.40 void CIP_SetCIEnquiryScreenResponse (U8BIT *slot_id*, U8BIT *ok_cancel*, U8BIT * *response_str_ptr*)

sends a message to the CI+ stack to indicate the response to the enquiry screen

Parameters

<i>slot_id</i>	- slot ID (0, 1, ...)
<i>ok_cancel</i>	- TRUE for OK, FALSE for cancel
<i>response_str_ptr</i>	- pointer to the response string

4.32.2.41 void CIP_SetCIMenuScreenResponse (U8BIT *slot_id*, U8BIT *response*)

Send the response to the menu screen to the module.

Parameters

<i>slot_id</i>	- slot ID (0, 1, ...)
<i>response</i>	- the index (1-based) of the response

4.32.2.42 void CIP_SetDisplayCharacterTables (U8BIT * *tables*, U16BIT *len*)

Sets the display character tables supported by the receiver. The tables are provided as a single string containing the prefix byte(s) of encoded text strings as defined in annex A.2 of ETSI EN 300 468. For example, if ISO/IEC 8859-7 is supported by the receiver, then text strings starting with 0x03 and with 0x10 0x00 0x07 are supported. In this case the string would be "\x03\x10\x00\x07".

Parameters

<i>tables</i>	- display character tables
<i>len</i>	- number of bytes in the tables string

4.32.2.43 void CIP_SetHostCountryCode (U32BIT *code*)

Set the Host Country code.

Parameters

<i>code</i>	- new country code (encoded as a U32BIT)
-------------	--

4.32.2.44 void CIP_SetHostLanguageCode (U32BIT *code*)

Set the Host Language code.

Parameters

<i>code</i>	- new language code (encoded in U32BIT)
-------------	---

4.32.2.45 void CIP_SetInputCharacterTables (U8BIT * *tables*, U16BIT *len*)

Sets the input character tables supported by the receiver. The tables are provided as a single string containing the prefix byte(s) of encoded text strings as defined in annex A.2 of ETSI EN 300 468. For example, if ISO/IEC 8859-7 is supported by the receiver, then text strings starting with 0x03 and with 0x10 0x00 0x07 are supported. In this case the string would be "\x03\x10\x00\x07".

Parameters

<i>tables</i>	- input character tables
<i>len</i>	- number of bytes in the tables string

4.32.2.46 void CIP_SetRecordOperatingMode (U8BIT *slot_id*, U8BIT *mode*, U16BIT *service_id*)

Sets the record operating mode for the given slot id.

Parameters

<i>slot_id</i>	- slot to be, or being, used for recording
<i>mode</i>	- operating mode of the host

4.32.2.47 BOOLEAN CIP_StartOperatorSearch (U32BIT *module*)

Called by the app to start an operator profile search that has been requested.

Parameters

<i>module</i>	- operator module
---------------	-------------------

Returns

TRUE if the search is started, FALSE otherwise

4.32.2.48 BOOLEAN CIP_TuneReply (U8BIT *path*, U32BIT *module*, E_CIP_TUNE_STATUS *status*)

Called by the host to send the status of the tune operation to the module.

Parameters

<i>path</i>	- decode path used for tuning
<i>module</i>	- host control module
<i>status</i>	- tune operation status

Returns

TRUE if reply sent, FALSE otherwise

4.32.2.49 void **CIP_UpdateUsageRules** (U8BIT *orig_uri*[CIP_URI_LEN], U8BIT *new_uri*[CIP_URI_LEN])

Update packed URI given a new packed URI.

Parameters

<i>orig_uri</i>	- Usage Rules Information to update
<i>new_uri</i>	- New Usage Rules Information

Note

Updated URI is at least as strict as original URI

4.33 middleware/media/inc/media_image.h File Reference

Media image functions.

#include "tectype.h" #include "stbhwc.h" Include dependency graph for media_image.h:

Functions

- BOOLEAN **STB_IMGConvertPNG** (U8BIT *image_data, U32BIT image_data_size, U8BIT **output_data, U32BIT *output_data_size, U16BIT *pixel_width, U16BIT *pixel_height)
- BOOLEAN **STB_IMGConvertJPEG** (U8BIT *image_data, U32BIT image_data_size, U8BIT **output_data, U32BIT *output_data_size, U16BIT *pixel_width, U16BIT *pixel_height)

4.33.1 Detailed Description

Media image functions.

Date

22 November 2010

Author

Steve Ford

4.34 middleware/ota/inc/stbota.h File Reference

API interfacing the middleware with Intellibyte loader library.

Data Structures

- struct [S_OTA_SAT_TUNING_PARAMS](#)
- struct [S_OTA_TER_TUNING_PARAMS](#)
- struct [S_OTA_CAB_TUNING_PARAMS](#)
- struct [S_OTA_TUNING_PARAMS](#)

Functions

- void **STB_OTASStartLoader** ([S_OTA_TUNING_PARAMS](#) *tuning_params, U8BIT ver_major, U8BIT ver_minor, U8BIT ver_release, U8BIT product_id)
- U8BIT **STB_OTAGetProgress** (void)
- void **STB_OTAContinueDownload** (BOOLEAN status)
- U16BIT **STB_OTAGetSSUCarouselPID** (void)

4.34.1 Detailed Description

API interfacing the middleware with Intellibyte loader library.

Date

06/10/2009

Author

Sergio Panseri

4.35 middleware/stb/inc/stbdpc.h File Reference

Header file - macros and function prototypes for public use.

This graph shows which files directly or indirectly include this file:

Defines

- #define **MAX_DISEQC_1_2_POSITION** 60
- #define **INVALID_POSITION_INDEX** 0x00
- #define **MAX_UNICABLE_BANKS** 8
- #define **INVALID_RES_ID** ((U8BIT)0xFF) /* ID used to represent an invalid resource */

Typedefs

- typedef enum e_stb_dp_tune_status **E_STB_DP_TUNE_STATUS**
- typedef enum e_stb_dp_decode_status **E_STB_DP_DECODE_STATUS**
- typedef enum e_stb_dp_demux_source **E_STB_DP_DEMUX_SOURCE**
- typedef enum e_stb_dp_decode_source **E_STB_DP_DECODE_SOURCE**
- typedef enum e_stb_dp_signal_type **E_STB_DP_SIGNAL_TYPE**
- typedef enum e_stb_dp_lnb_power **E_STB_DP_LNB_POWER**
- typedef enum e_stb_dp_lnb_type **E_STB_DP_LNB_TYPE**
- typedef enum e_stb_dp_diseqc_cswitch **E_STB_DP_DISEQC_CSWITCH**
- typedef enum e_stb_dp_diseqc_tone **E_STB_DP_DISEQC_TONE**
- typedef enum e_stb_dp_polarity **E_STB_DP_POLARITY**
- typedef enum e_stb_dp_fec **E_STB_DP_FEC**
- typedef enum e_stb_dp_fec_type **E_STB_DP_FEC_TYPE**
- typedef enum e_stb_dp_modulation **E_STB_DP_MODULATION**
- typedef enum e_stb_dp_tmode **E_STB_DP_TMODE**
- typedef enum e_stb_dp_tbwidht **E_STB_DP_TBWIDHT**
- typedef enum e_stb_dp_cmode **E_STB_DP_CMODE**
- typedef enum e_stb_dp_tune_terr_type **E_STB_DP_TTYPE**
- typedef enum e_stb_dp_audio_mode **E_STB_DP_AUDIO_MODE**
- typedef enum e_stb_ota_sw_upgrade_search_mode **E_STB_OTA_SW_UPGRADE_SEARCH_MODE**
- typedef enum e_stb_dp_analogue_video_type **E_STB_DP_ANALOG_VIDEO_TYPE**
- typedef enum e_stb_dp_video_codec **E_STB_DP_VIDEO_CODEC**
- typedef enum e_stb_dp_audio_codec **E_STB_DP_AUDIO_CODEC**
- typedef enum e_stb_dp_ad_audio **E_STB_DP_AD_AUDIO**

Enumerations

- enum **e_stb_dp_tune_status** { **TUNE_WAITING** = 0, **TUNE_NO_LOCK** = 1, **TUNE_LOCKED** = 2, **TUNE_STOPPED** = 3 }
- enum **e_stb_dp_decode_status** { **DECODE_STOPPING** = 0, **DECODE_STARTING** = 1, **DECODE_STOPPED** = 2, **DECODE_RUNNING** = 3, **DECODE_LOCKED** = 4 }
- enum **e_stb_dp_demux_source** { **DEMUX_SOURCE_TUNER** = 0, **DEMUX_SOURCE_FILE** = 1, **DEMUX_SOURCE_LINEIN** = 2 }
- enum **e_stb_dp_decode_source** { **DECODE_SOURCE_DEMUX** = 0, **DECODE_SOURCE_FILE** = 1, **DECODE_SOURCE_LINEIN** = 2 }
- enum **e_stb_dp_signal_type** { **SIGNAL_NONE** = 0, **SIGNAL_QPSK** = 1, **SIGNAL_COFDM** = 2, **SIGNAL_QAM** = 4, **SIGNAL_ANALOG** = 8 }
- enum **e_stb_dp_lnb_power** { **LNB_POWER_OFF** = 0, **LNB_POWER_ON** = 1, **LNB_POWER_AUTO** = 2 }
- enum **e_stb_dp_lnb_type** { **LNB_TYPE_SINGLE** = 0, **LNB_TYPE_UNIVERSAL** = 1, **LNB_TYPE_UNICABLE** = 2 }
- enum **e_stb_dp_diseqc_cswitch** { **DISEQC_CSWITCH_OFF** = 0, **DISEQC_CSWITCH_A** = 1, **DISEQC_CSWITCH_B** = 2, **DISEQC_CSWITCH_C** = 3, **DISEQC_CSWITCH_D** = 4 }

- enum **e_stb_dp_diseqc_tone** { DISEQC_TONE_OFF = 0, DISEQC_TONE_A = 1, DISEQC_TONE_B = 2 }
- enum **e_stb_dp_polarity** { POLARITY_HORIZONTAL = 0, POLARITY_VERTICAL = 1, POLARITY_LEFT = 2, POLARITY_RIGHT = 3 }
- enum **e_stb_dp_fec** { FEC_AUTOMATIC = 0, FEC_1_2 = 1, FEC_2_3 = 2, FEC_3_4 = 3, FEC_5_6 = 4, FEC_7_8 = 5, FEC_1_4 = 6, FEC_1_3 = 7, FEC_2_5 = 8, FEC_8_9 = 9, FEC_9_10 = 10 }
- enum **e_stb_dp_fec_type** { FEC_TYPE_AUTO, FEC_TYPE_DVBS1, FEC_TYPE_DVBS2 }
- enum **e_stb_dp_modulation** { MOD_AUTO, MOD_QPSK, MOD_8PSK, MOD_16QAM }
- enum **e_stb_dp_tmode** { MODE_COFDM_2K, MODE_COFDM_8K, MODE_COFDM_4K, MODE_COFDM_1K, MODE_COFDM_16K, MODE_COFDM_32K, MODE_COFDM_UNDEFINED }
- enum **e_stb_dp_tbwidth** { TBWIDTH_8MHZ, TBWIDTH_7MHZ, TBWIDTH_6MHZ, TBWIDTH_5MHZ, TBWIDTH_10MHZ, TBWIDTH_UNDEFINED }
- enum **e_stb_dp_cmode** { MODE_QAM_AUTO, MODE_QAM_4, MODE_QAM_8, MODE_QAM_16, MODE_QAM_32, MODE_QAM_64, MODE_QAM_128, MODE_QAM_256 }
- enum **e_stb_dp_tune_terr_type** { TERR_TYPE_UNKNOWN, TERR_TYPE_DVBT, TERR_TYPE_DVBT2 }
- enum **e_stb_dp_audio_mode** { AUDIO_STEREO = 0, AUDIO_LEFT = 1, AUDIO_RIGHT = 2, AUDIO_MONO = 3, AUDIO_MULTICHANNEL = 4, AUDIO_UNDEF = 5 }
- enum **e_stb_ota_sw_upgrade_search_mode** { OTA_SEARCH_OFF = 0, OTA_SEARCH_AUTO = 1, OTA_SEARCH_MANUAL = 2 }
- enum **e_stb_dp_analogue_video_type** { ANLG_VIDEO_PAL_I = 0, ANLG_VIDEO_PAL_B = 1, ANLG_VIDEO_PAL_G = 2, ANLG_VIDEO_PAL_D = 3, ANLG_VIDEO_PAL_K = 4, ANLG_VIDEO_PAL_L = 5, ANLG_VIDEO_PAL_L-DASH = 6 }
- enum **e_stb_dp_video_codec** { VIDEO_CODEC_AUTO = 0, VIDEO_CODEC_MPEG1 = 1, VIDEO_CODEC_MPEG2 = 2, VIDEO_CODEC_H264 = 3, VIDEO_CODEC_H265 = 4 }
- enum **e_stb_dp_audio_codec** { AUDIO_CODEC_AUTO = 0, AUDIO_CODEC_MP2 = 1, AUDIO_CODEC_MP3 = 2, AUDIO_CODEC_AC3 = 3, AUDIO_CODEC_EAC3 = 4, AUDIO_CODEC_AAC = 5, AUDIO_CODEC_HEAAC = 6, AUDIO_CODEC_AAC_ADTS = 7 }
- enum **e_stb_dp_ad_audio** { AD_AUDIO_OFF = 0, AD_AUDIO_ON = 1, AD_AUDIO_PAUSED = 2, AD_AUDIO_PLAYING = 3, AD_AUDIO_WAITING = 4 }
- enum **E_STB_DP_RES_OWNER** { RES_OWNER_NONE, RES_OWNER_DVB, RES_OWNER_CIPLUS }
- enum **E_STB_DP_PRIORITY** { DP_PRIORITY_LOW, DP_PRIORITY_HIGH }

Functions

- void **STB_DPInitialise** (void)
- U8BIT **STB_DPGetNumPaths** (void)

- U8BIT **STB_DPAcquireTunerPath** (E_STB_DP_SIGNAL_TYPE tuner_type, void *service, void *transport, E_STB_DP_RES_OWNER owner, E_STB_DP_PRIORITY priority, BOOLEAN with_decoders, BOOLEAN for_recording)
- U8BIT **STB_DPAcquirePlaybackPath** (void *service)
- void **STB_DPReleaseDecoders** (U8BIT path)
- BOOLEAN **STB_DPReleasePath** (U8BIT path, E_STB_DP_RES_OWNER owner)
- void **STB_DPReleaseAllPaths** (void)
- U8BIT **STB_DPAcquireCISlotForPath** (U8BIT path, U8BIT *pmt_data, U8BIT *ci_protection_desc)
- BOOLEAN **STB_DPUseCISlotWithPath** (U8BIT path, U8BIT slot_id)
- void **STB_DPReleaseCISlotFromPath** (U8BIT path)
- U8BIT **STB_DPGetPathCISlot** (U8BIT path)
- U8BIT **STB_DPIsCISlotInUse** (U8BIT start_path, U8BIT slot_id, U8BIT ignore_path)
- BOOLEAN **STB_DPAcquireCADescramblerForPath** (U8BIT path, U8BIT *pmt_data, U32BIT *ca_handle)
- void **STB_DPReleaseCADescramblerFromPath** (U8BIT path)
- BOOLEAN **STB_DPGetPathCADescrambler** (U8BIT path, U32BIT *handle)
- U8BIT **STB_DPGetLivePath** (void)
- BOOLEAN **STB_DPIsLivePath** (U8BIT path)
- BOOLEAN **STB_DPIsRecordingPath** (U8BIT path)
- BOOLEAN **STB_DPIsDecodingPath** (U8BIT path)
- U8BIT **STB_DPGetPlaybackPath** (void)
- U8BIT **STB_DPPathForTuner** (U8BIT start_path, U8BIT tuner_num)
- U8BIT **STB_DPPathForAudioDecoder** (U8BIT decoder_num)
- U8BIT **STB_DPPathForADDecoder** (U8BIT decoder_num)
- U8BIT **STB_DPPathForVideoDecoder** (U8BIT decoder_num)
- void **STB_DPSetTunedTransport** (U8BIT path, void *transport)
- void * **STB_DPGetTunedTransport** (U8BIT path)
- void **STB_DPSetTunedService** (U8BIT path, void *service)
- void * **STB_DPGetTunedService** (U8BIT path)
- U8BIT **STB_DPGetPathForService** (void *service)
- BOOLEAN **STB_DPCanTuneTo** (E_STB_DP_SIGNAL_TYPE tuner_type, void *service, void *transport)
Checks whether there's a tuner available to tune to the given service or transport.
- U8BIT **STB_DPGetPathTuner** (U8BIT path)
- E_STB_DP_SIGNAL_TYPE **STB_DPGetPathTunerType** (U8BIT path)
- void **STB_DPEnableAllTuners** (void)
Sets the 'disabled' state of all tuners to FALSE so they can all be used by the resource manager.
- void **STB_DPSetTunerDisabled** (U8BIT path, BOOLEAN disabled)
Disables/enables the tuner associated with this decode path.
- U8BIT **STB_DPGetNumEnabledTuners** (void)
Returns the number of enabled tuners (i.e. that haven't been disabled)
- U8BIT **STB_DPGetPathDemux** (U8BIT path)

- U8BIT **STB_DPGetPathAudioDecoder** (U8BIT path)
- U8BIT **STB_DPGetPathVideoDecoder** (U8BIT path)
- U8BIT **STB_DPGetMHEGPath** (void)
- void **STB_DPSetOwner** (U8BIT path, E_STB_DP_RES_OWNER owner)
- BOOLEAN **STB_DPSetOwnerData** (U8BIT path, void *data, U32BIT data_size)
- BOOLEAN **STB_DPIsOwnedBy** (U8BIT path, E_STB_DP_RES_OWNER owner)
- void * **STB_DPGetOwnerData** (U8BIT path, U32BIT *data_size)
- void **STB_DPSetDecodeSource** (U8BIT path, E_STB_DP_DECODE_SOURCE source, U32BIT param)
- void **STB_DPGetDecodeSource** (U8BIT path, E_STB_DP_DECODE_SOURCE *source, U32BIT *param)
- BOOLEAN **STB_DPStartRecording** (U8BIT path, U32BIT param)
- void **STB_DPStopRecording** (U8BIT path)
- BOOLEAN **STB_DPIsRecording** (U8BIT path, U32BIT *handle)
- void **STB_DPStartTune** (U8BIT path)
- void **STB_DPStartScan** (U8BIT path)
- void **STB_DPStopTune** (U8BIT path)
- void **STB_DPTuneOff** (U8BIT path)
- void **STB_DPStartVideoDecoding** (U8BIT path)
- void **STB_DPStartADDecoding** (U8BIT path)
- void **STB_DPStartAudioDecoding** (U8BIT path)
- void **STB_DPStartDecoding** (U8BIT path)
- void **STB_DPStopVideoDecoding** (U8BIT path)
- void **STB_DPStopADDecoding** (U8BIT path)
- void **STB_DPStopAudioDecoding** (U8BIT path)
- void **STB_DPStopDecoding** (U8BIT path)
- void **STB_DPStartSI** (U8BIT path)
- void **STB_DPStopSI** (U8BIT path)
- void **STB_DPRRequestSIExtendedEvent** (U8BIT path, U32BIT start_date, U32BIT start_hour, U32BIT start_min)
- U32BIT **STB_DPGetSIRequestParam1** (U8BIT path)
- U32BIT **STB_DPGetSIRequestParam2** (U8BIT path)
- U32BIT **STB_DPGetSIRequestParam3** (U8BIT path)
- void **STB_DPSetTuneStatus** (U8BIT path, E_STB_DP_TUNE_STATUS state)
- E_STB_DP_TUNE_STATUS **STB_DPGetTuneStatus** (U8BIT path)
- void **STB_DPSetADEnabled** (U8BIT path, BOOLEAN state)
- void **STB_DPSetADAudio** (U8BIT path, E_STB_DP_AD_AUDIO state)
- E_STB_DP_AD_AUDIO **STB_DPGetADAudio** (U8BIT path)
- void **STB_DPSetADStatus** (U8BIT path, E_STB_DP_DECODE_STATUS state)
- E_STB_DP_DECODE_STATUS **STB_DPGetADStatus** (U8BIT path)
- void **STB_DPSetAudioStatus** (U8BIT path, E_STB_DP_DECODE_STATUS state)
- E_STB_DP_DECODE_STATUS **STB_DPGetAudioStatus** (U8BIT path)
- void **STB_DPSetVideoStatus** (U8BIT path, E_STB_DP_DECODE_STATUS decoder)
- E_STB_DP_DECODE_STATUS **STB_DPGetVideoStatus** (U8BIT path)

- void **STB_DPSetSignalType** (U8BIT path, E_STB_DP_SIGNAL_TYPE sigtype)
- E_STB_DP_SIGNAL_TYPE **STB_DPGetSignalType** (U8BIT path)
- void **STB_DPSetTuneRelock** (U8BIT path, BOOLEAN state)
- BOOLEAN **STB_DPGetTuneRelock** (U8BIT path)
- void **STB_DPSetLockEnable** (U8BIT path, BOOLEAN state)
- BOOLEAN **STB_DPGetLockEnable** (U8BIT path)
- void **STB_DPSetSearchMode** (U8BIT path, BOOLEAN state)
- BOOLEAN **STB_DPGetSearchMode** (U8BIT path)
- void **STB_DPSetTVSearch** (U8BIT path, BOOLEAN state)
- BOOLEAN **STB_DPGetTVSearch** (U8BIT path)
- void **STB_DPSetRadioSearch** (U8BIT path, BOOLEAN state)
- BOOLEAN **STB_DPGetRadioSearch** (U8BIT path)
- void **STB_DPSetFTASearch** (U8BIT path, BOOLEAN state)
- BOOLEAN **STB_DPGetFTASearch** (U8BIT path)
- void **STB_DPSetScramSearch** (U8BIT path, BOOLEAN state)
- BOOLEAN **STB_DPGetScramSearch** (U8BIT path)
- void **STB_DPSetNetworkSearch** (U8BIT path, BOOLEAN state)
- BOOLEAN **STB_DPGetNetworkSearch** (U8BIT path)
- void **STB_DPSetLNBPower** (U8BIT path, E_STB_DP_LNB_POWER state)
- E_STB_DP_LNB_POWER **STB_DPGetLNBPower** (U8BIT path)
- void **STB_DPSetLNBTtype** (U8BIT path, E_STB_DP_LNB_TYPE type)
- E_STB_DP_LNB_TYPE **STB_DPGetLNBTtype** (U8BIT path)
- void **STB_DPSetLNBLoFrequency** (U8BIT path, U16BIT freq)
- U16BIT **STB_DPGetLNBLoFrequency** (U8BIT path)
- void **STB_DPSetLNBHiFrequency** (U8BIT path, U16BIT freq)
- U16BIT **STB_DPGetLNBHiFrequency** (U8BIT path)
- void **STB_DPSetLNB22k** (U8BIT path, BOOLEAN state)
- BOOLEAN **STB_DPGetLNB22k** (U8BIT path)
- void **STB_DPSetLNB12v** (U8BIT path, BOOLEAN state)
- BOOLEAN **STB_DPGetLNB12v** (U8BIT path)
- void **STB_DPSetPulsePosition** (U8BIT path, BOOLEAN state)
- BOOLEAN **STB_DPGetPulsePosition** (U8BIT path)
- void **STB_DPSetDISEQCPosition** (U8BIT path, BOOLEAN state)
- BOOLEAN **STB_DPGetDISEQCPosition** (U8BIT path)
- void **STB_DPSetDISEQCCSwitch** (U8BIT path, E_STB_DP_DISEQC_CSWITCH state)
- E_STB_DP_DISEQC_CSWITCH **STB_DPGetDISEQCCSwitch** (U8BIT path)
- void **STB_DPSetDISEQCUSwitch** (U8BIT path, U8BIT state)
- U8BIT **STB_DPGetDISEQCUSwitch** (U8BIT path)
- void **STB_DPSetDISEQCTone** (U8BIT path, E_STB_DP_DISEQC_TONE state)
- E_STB_DP_DISEQC_TONE **STB_DPGetDISEQCTone** (U8BIT path)
- void **STB_DPSetDISEQCSMATV** (U8BIT path, BOOLEAN state)
- BOOLEAN **STB_DPGetDISEQCSMATV** (U8BIT path)
- void **STB_DPSetDISEQCRepeats** (U8BIT path, U8BIT count)
- U8BIT **STB_DPGetDISEQCRepeats** (U8BIT path)
- U8BIT **STB_DPGetUnicableParams** (U8BIT path, U32BIT unicable_if[MAX_UNICABLE_BANKS], U32BIT *lnb_lo_freq, U32BIT *lnb_hi_freq)

- void **STB_DPSetUncablePositionB** (U8BIT path, BOOLEAN position_b)
- BOOLEAN **STB_DPGetUncablePositionB** (U8BIT path)
- void **STB_DPSetUncableChannel** (U8BIT path, U8BIT chan)
- U8BIT **STB_DPGetUncableChannel** (U8BIT path)
- void **STB_DPSetUncableFrequency** (U8BIT path, U32BIT freq)
- U32BIT **STB_DPGetUncableFrequency** (U8BIT path)
- void **STB_DPSetDishLimitE** (U8BIT path)
- void **STB_DPSetDishLimitW** (U8BIT path)
- void **STB_DPEnableDishLimits** (U8BIT path, U16BIT ecount, U16BIT wcount)
- void **STB_DPDisableDishLimits** (U8BIT path)
- void **STB_DPStartDishMoveE** (U8BIT path, U16BIT count)
- void **STB_DPStartDishMoveW** (U8BIT path, U16BIT count)
- void **STB_DPStopDishMove** (U8BIT path)
- void **STB_DPCentreDishMove** (U8BIT path)
- void **STB_DPStoreDishPosition** (U8BIT path, U16BIT count)
- void **STB_DPSetDishPosition** (U8BIT path, U16BIT count)
- U16BIT **STB_DPGetDishPosition** (U8BIT path)
- U16BIT **STB_DPGetDishRequest** (U8BIT path)
- void **STB_DPSetSkewPosition** (U8BIT path, U16BIT count)
- U16BIT **STB_DPGetSkewPosition** (U8BIT path)
- void **STB_DPSetFrequency** (U8BIT path, U32BIT freq)
- U32BIT **STB_DPGetFrequency** (U8BIT path)
- void **STB_DPSetPolarity** (U8BIT path, E_STB_DP_POLARITY pol)
- E_STB_DP_POLARITY **STB_DPGetPolarity** (U8BIT path)
- void **STB_DPSetSymbolRate** (U8BIT path, U16BIT sym)
- U16BIT **STB_DPGetSymbolRate** (U8BIT path)
- void **STB_DPSetFEC** (U8BIT path, E_STB_DP_FEC fec)
- E_STB_DP_FEC **STB_DPGetFEC** (U8BIT path)
- void **STB_DPSetDVBS2** (U8BIT path, BOOLEAN dvb_s2)
- BOOLEAN **STB_DPGetDVBS2** (U8BIT path)
- void **STB_DPSetModulation** (U8BIT path, E_STB_DP_MODULATION modulation)
- E_STB_DP_MODULATION **STB_DPGetModulation** (U8BIT path)
- void **STB_DPSetTerrMode** (U8BIT path, E_STB_DP_TMODE mode)
- E_STB_DP_TMODE **STB_DPGetTerrMode** (U8BIT path)
- void **STB_DPSetTerrBandwidth** (U8BIT path, E_STB_DP_TBWIDTH bwidth)
- E_STB_DP_TBWIDTH **STB_DPGetTerrBandwidth** (U8BIT path)
- void **STB_DPSetTerrFreqOff** (U8BIT path, S8BIT offset)
- S8BIT **STB_DPGetTerrFreqOff** (U8BIT path)
- void **STB_DPSetTerrType** (U8BIT path, E_STB_DP_TTYPE type)
- E_STB_DP_TTYPE **STB_DPGetTerrType** (U8BIT path)
- void **STB_DPSetTerrPLP** (U8BIT path, U8BIT plp_id)
- U8BIT **STB_DPGetTerrPLP** (U8BIT path)
- void **STB_DPSetCableMode** (U8BIT path, E_STB_DP_CMODE mode)
- E_STB_DP_CMODE **STB_DPGetCableMode** (U8BIT path)
- void **STB_DPSetAnalogVideoType** (U8BIT path, E_STB_DP_ANALOG_VIDEO_TYPE vtype)

- E_STB_DP_ANALOG_VIDEO_TYPE **STB_DPGetAnalogVideoType** (U8BIT path)
- void **STB_DPSetAnalogFreqOff** (U8BIT path, S8BIT offset)
- S8BIT **STB_DPGetAnalogFreqOff** (U8BIT path)
- void **STB_DPSetPCRPID** (U8BIT path, U16BIT pid)
- U16BIT **STB_DPGetPCRPID** (U8BIT path)
- void **STB_DPSetVideoPID** (U8BIT path, U16BIT pid)
- U16BIT **STB_DPGetVideoPID** (U8BIT path)
- void **STB_DPSetAudioPID** (U8BIT path, U16BIT pid)
- U16BIT **STB_DPGetAudioPID** (U8BIT path)
- void **STB_DPSetADPID** (U8BIT path, U16BIT pid)
- U16BIT **STB_DPGetADPID** (U8BIT path)
- void **STB_DPSetTextPID** (U8BIT path, U16BIT pid)
- U16BIT **STB_DPGetTextPID** (U8BIT path)
- void **STB_DPSetDataPID** (U8BIT path, U16BIT pid)
- U16BIT **STB_DPGetDataPID** (U8BIT path)
- void **STB_DPSetDecodePIDs** (U8BIT path, U16BIT pcr_pid, U16BIT video_pid, U16BIT audio_pid, U16BIT text_pid, U16BIT data_pid, U16BIT AD_pid)
- void **STB_DPSetADMode** (U8BIT path, E_STB_DP_AUDIO_MODE mode)
- E_STB_DP_AUDIO_MODE **STB_DPGetADMode** (U8BIT path)
- void **STB_DPSetAudioMode** (U8BIT path, E_STB_DP_AUDIO_MODE mode)
- E_STB_DP_AUDIO_MODE **STB_DPGetAudioMode** (U8BIT path)
- void **STB_DPSetLockMode** (U8BIT path, BOOLEAN mode)
- BOOLEAN **STB_DPGetLockMode** (U8BIT path)
- void **STB_DPSetOTASearchMode** (U8BIT path, E_STB_OTA_SW_UPGRADE_SEARCH_MODE mode)
- E_STB_OTA_SW_UPGRADE_SEARCH_MODE **STB_DPGetOTASearchMode** (U8BIT path)
- BOOLEAN **STB_DPOTASearchEnabled** (U8BIT path)
- BOOLEAN **STB_DPSetVideoCodec** (U8BIT path, E_STB_DP_VIDEO_CODEC codec)
- E_STB_DP_VIDEO_CODEC **STB_DPGetVideoCodec** (U8BIT path)
- BOOLEAN **STB_DPSetAudioCodec** (U8BIT path, E_STB_DP_AUDIO_CODEC codec)
- E_STB_DP_AUDIO_CODEC **STB_DPGetAudioCodec** (U8BIT path)
- BOOLEAN **STB_DPSetADCodec** (U8BIT path, E_STB_DP_AUDIO_CODEC codec)
- E_STB_DP_AUDIO_CODEC **STB_DPGetADCodec** (U8BIT path)
- U8BIT **STB_DPGetPathSecondaryVideoDecoder** (U8BIT path)

4.35.1 Detailed Description

Header file - macros and function prototypes for public use.

Date

26/09/2000

4.35.2 Function Documentation

4.35.2.1 **BOOLEAN STB_DPCanTuneTo** (*E_STB_DP_SIGNAL_TYPE tuner_type*, void * *service*, void * *transport*)

Checks whether there's a tuner available to tune to the given service or transport.

Parameters

<i>service</i>	service to be tuned to, can be NULL
<i>transport</i>	transport to be tuned to

Returns

TRUE if there's a tuner available

4.35.2.2 **U8BIT STB_DPGetNumEnabledTuners** (void)

Returns the number of enabled tuners (i.e. that haven't been disabled)

Returns

number of enabled tuners

4.35.2.3 **void STB_DPSetTunerDisabled** (*U8BIT path*, *BOOLEAN disabled*)

Disables/enables the tuner associated with this decode path.

Parameters

<i>path</i>	decode path
<i>disabled</i>	TRUE if the tuner is to be disabled, FALSE otherwise

4.36 middleware/stb/inc/stbdsapi.h File Reference

Header file - Function prototypes for DVB subtitles api.

Functions

- void **STB_SUBEnable** (U8BIT path, BOOLEAN enable)
- void **STB_SUBStart** (U8BIT path, U16BIT pid, U16BIT composition_id, U16BIT ancillary_id)
- void **STB_SUBStop** (U8BIT path)
- void **STB_SUBSetStream** (U8BIT path, U16BIT pid, U16BIT composition_id, - U16BIT ancillary_id)
- void **STB_SUBStatus** (BOOLEAN *started, BOOLEAN *shown)
- void **STB_SUBReadSettings** (BOOLEAN *running, U16BIT *pid, U16BIT *comp_id, U16BIT *anc_id, BOOLEAN *enabled)

4.36.1 Detailed Description

Header file - Function prototypes for DVB subtitles api.

Date

27/02/2004

Author

Ocean Blue

4.37 middleware/stb/inc/stbebutt.h File Reference

Header file - EBU Teletext driver.

Data Structures

- struct [s_ebutt_font](#)

Typedefs

- typedef enum e_ebutt_event **E_EBUTT_EVENT**
- typedef enum e_ebutt_caching_method **E_EBUTT_CACHING_METHOD**
- typedef enum e_ebutt_character_set_designation **E_EBUTT_CHARACTER_SET_DESIGNATION**
- typedef struct [s_ebutt_font](#) **S_EBUTT_FONT**

Enumerations

- enum **e_ebutt_event** { **EBUTT_EVENT_QUICK_NAVIGATE_1**, **EBUTT_EVENT_QUICK_NAVIGATE_2**, **EBUTT_EVENT_QUICK_NAVIGATE_3**, **EBUTT_EVENT_QUICK_NAVIGATE_4**, **EBUTT_EVENT_0**, **EBUTT_EVENT_1**, **EBUTT_EVENT_2**, **EBUTT_EVENT_3**, **EBUTT_EVENT_4**, **EBUTT_EVENT_5**, **EBUTT_EVENT_6**, **EBUTT_EVENT_7**, **EBUTT_EVENT_8**, **EBUTT_EVENT_9**, **EBUTT_EVENT_INDEXPAGE**, **EBUTT_EVENT_NEXTPAGE**, **EBUTT_EVENT_PREVIOUSPAGE**, **EBUTT_EVENT_NEXTSUBPAGE**, **EBUTT_EVENT_PREVIOUSSUBPAGE**, **EBUTT_EVENT_BACKPAGE**, **EBUTT_EVENT_FORWARDPAGE**, **EBUTT_EVENT_HOLD**, **EBUTT_EVENT_REVEAL**, **EBUTT_EVENT_CLEAR**, **EBUTT_EVENT_MIX_VIDEO**, **EBUTT_EVENT_DOUBLE_HEIGHT**, **EBUTT_EVENT_DOUBLE_SCROLL_UP**, **EBUTT_EVENT_DOUBLE_SCROLL_DOWN**, **EBUTT_EVENT_TIMER** }
- enum **e_ebutt_caching_method** { **EBUTT_CACHING_METHOD_NONE** = 0, **EBUTT_CACHING_METHOD_PREVIOUS_NEXT**, **EBUTT_CACHING_METHOD_HISTORY**, **EBUTT_CACHING_METHOD_NAVIGATION**, **EBUTT_CACHING_METHOD_NAVIGATION_TREE**, **EBUTT_CACHING_METHOD_ALL** }

- enum **e_ebutt_character_set_designation** { EBUTT_CHARACTER_SET_DESIGNATION_LATIN_DEFAULT = 0, EBUTT_CHARACTER_SET_DESIGNATION_LATIN_POLISH = 1, EBUTT_CHARACTER_SET_DESIGNATION_LATIN_TURKISH = 2, EBUTT_CHARACTER_SET_DESIGNATION_LATIN_SERBIAN_RUMANIAN = 3, EBUTT_CHARACTER_SET_DESIGNATION_CYRILLIC = 4, EBUTT_CHARACTER_SET_DESIGNATION_GREEK_TURKISH = 6, EBUTT_CHARACTER_SET_DESIGNATION_LATIN_ARABIC = 8, EBUTT_CHARACTER_SET_DESIGNATION_HEBREW_ARABIC = 10 }

Functions

- **BOOLEAN STB_EBUTT_Initialise** (void)
This must be called to invoke the EBU TeleText driver module before all other functionality can be accessed. This initiates the relevant resources used by the driver module. As a consequence of calling this function, no TeleText processing will occur until other functions are called. It is envisaged that this function may be called within the general initialisation of the STB layer.
- **void STB_EBUTT_Kill** (U8BIT path)
This is the accompanying function to [STB_EBUTT_Initialise\(\)](#), and is called to shut-down the EBU TeleText driver module and release the relevant resources used by the driver module. This may not be used in some implementations.
- **void STB_EBUTT_Start** (U8BIT path, U16BIT text_pid, U8BIT magazine, U8BIT page)
These functions are called to control whether received TeleText data is processed. - Using this functionality ensures that the process of collating page information can be halted when processing resources are a consideration. It is envisaged that some implementations may wish only to collate TeleText content only when the display is being shown, whilst other designs may wish to cache significant page content as a background process. Abstracting this functionality from the initialisation routines gives the option to support both solution types.
- **void STB_EBUTT_Stop** (U8BIT path, BOOLEAN reset_cache)
- **BOOLEAN STB_EBUTT_InjectData** (U8BIT *data_ptr, U32BIT data_length)
Allows teletext PES data packets to be injected by an external module, which will be decoded and displayed.
- **BOOLEAN STB_EBUTT_Show** (E_EBUTT_CHARACTER_SET_DESIGNATION character_set_designation, BOOLEAN navigate_to_index_page, BOOLEAN show_header)
Called to initiate continuous display of TeleText page content. This will result in either the index page defined in the service data stream being selected, or the previously displayed page (prior to a call to [STB_EBUTT_Hide\(\)](#) being made) from being reinstated. This function can be called irrespective of the use of the [STB_EBUTT_Start\(\)](#) / [STB_EBUTT_Stop\(\)](#) functions, but of course may result in no TeleText data being displayed if none already exists within the cache. The return value indicates whether initialisation of the display mechanism has been successful or not.
- **void STB_EBUTT_Hide** (void)
Called to halt the continuous display of TeleText page content, and to clear the relevant area of the application display.
- **void STB_EBUTT_SetCacheMethod** (E_EBUTT_CACHING_METHOD ebutt_caching_method)

Called to define the strategy used to determine what page content is to be displayed from the TeleText carousel content.

- void [STB_EBUTT_NotifyEvent](#) (E_EBUTT_EVENT event_type)

Called whenever a TeleText-specific event being invoked. This is used to pass relevant user input to the driver, and the enumerate type supplied is defined independently of any IR handset / keypad.

- void [STB_EBUTT_NotifyServiceChange](#) (void)

Called whenever the application permits a TV/Radio service change during a TeleText display session. This is provided because an update of the display to new page content is required.

- void [STB_EBUTT_SetDisplayBrightness](#) (U8BIT gun_intensity)

Called to adjust the display brightness (colour intensity) of the Teletext pages.

- void [STB_EBUTT_IncreaseDisplayBrightness](#) (void)

Called to adjust the display brightness (colour intensity) of the Teletext pages.

- void **STB_EBUTT_DecreaseDisplayBrightness** (void)

- void [STB_EBUTT_SetDisplayAntiAliasing](#) (U8BIT antialias_level)

Called to adjust the display anti-aliasing level of the Teletext pages. This is used to control 'flicker' on TV displays.

- void [STB_EBUTT_IncreaseDisplayAntiAliasing](#) (void)

Called to adjust the display anti-aliasing level of the Teletext pages.

- void **STB_EBUTT_DecreaseDisplayAntiAliasing** (void)

- void [STB_EBUTT_SetDisplayMixTransparency](#) (U8BIT transparency_level)

Called to adjust the display video-mix transparency level of the Teletext pages.

- void [STB_EBUTT_IncreaseDisplayMixTransparency](#) (void)

Called to adjust the display video-mix transparency level of the Teletext pages.

- void **STB_EBUTT_DecreaseDisplayMixTransparency** (void)

- BOOLEAN [STB_EBUTT_IsDisplayHeld](#) (void)

Called to ascertain whether the present page is in 'hold' mode or not. Thus function is useful when allocating handset keys to events on the basis of the current functional state of the Teletext driver.

- BOOLEAN [STB_EBUTT_IsDisplayDoubleHeight](#) (void)

Called to ascertain whether the present page is in 'double height' mode or not. Thus function is useful when allocating handset keys to events on the basis of the current functional state of the Teletext driver.

Variables

- const [S_EBUTT_FONT](#) *const **ebutt_font_ptr**

4.37.1 Detailed Description

Header file - EBU Teletext driver.

Date

04/02/2004

Author

Ocean Blue

4.37.2 Function Documentation**4.37.2.1 BOOLEAN STB_EBUTT_Initialise (void)**

This must be called to invoke the EBU TeleText driver module before all other functionality can be accessed. This initiates the relevant resources used by the driver module. As a consequence of calling this function, no TeleText processing will occur until other functions are called. It is envisaged that this function may be called within the general initialisation of the STB layer.

Returns

TRUE if initialisation was successful, FALSE otherwise.

4.37.2.2 BOOLEAN STB_EBUTT_InjectData (U8BIT * data_ptr, U32BIT data_length)

Allows teletext PES data packets to be injected by an external module, which will be decoded and displayed.

Parameters

<i>data_ptr</i>	- pointer to first whole PES packet data
<i>data_length</i>	- number of bytes of data provided

Returns

TRUE if the data is valid EBU teletext PES data, FALSE otherwise

4.37.2.3 BOOLEAN STB_EBUTT_IsDisplayDoubleHeight (void)

Called to ascertain whether the present page is in 'double height' mode or not. Thus function is useful when allocating handset keys to events on the basis of the current functional state of the Teletext driver.

Returns

TRUE if the displayed page is double height, FALSE otherwise.

4.37.2.4 BOOLEAN STB_EBUTT_IsDisplayHeld (void)

Called to ascertain whether the present page is in 'hold' mode or not. Thus function is useful when allocating handset keys to events on the basis of the current functional state of the Teletext driver.

Returns

TRUE if the displayed page is held, FALSE otherwise.

4.37.2.5 void STB_EBUTT_NotifyEvent (E_EBUTT_EVENT *event_type*)

Called whenever a TeleText-specific event being invoked. This is used to pass relevant user input to the driver, and the enumerate type supplied is defined independently of any IR handset / keypad.

Parameters

<i>E_EBUTT_EVENT</i>	event_type -
----------------------	--------------

4.37.2.6 void STB_EBUTT_SetCacheMethod (E_EBUTT_CACHING_METHOD *ebutt_caching_method*)

Called to define the strategy used to determine what page content is to be displayed from the TeleText carousel content.

Parameters

<i>E_EBUTT_CACHING_METHOD</i>	ebutt_caching_method -
-------------------------------	------------------------

4.37.2.7 void STB_EBUTT_SetDisplayAntiAliasing (U8BIT *antialias_level*)

Called to adjust the display anti-aliasing level of the Teletext pages. This is used to control 'flicker' on TV displays.

Parameters

<i>U8BIT</i>	antialias_level - a factor between zero (no anti-aliasing) to 8 (full anti-aliasing)
--------------	--

4.37.2.8 void STB_EBUTT_SetDisplayBrightness (U8BIT *gun_intensity*)

Called to adjust the display brightness (colour intensity) of the Teletext pages.

Parameters

<i>U8BIT</i>	gun_intensity - a RGB gun intensity, ranging from 255 (brightest) to 127 (darkest)
--------------	--

4.37.2.9 void STB_EBUTT_SetDisplayMixTransparency (U8BIT *transparency_level*)

Called to adjust the display video-mix transparency level of the Teletext pages.

Parameters

<i>U8BIT</i>	transparency_level - a factor between zero (no transparency) to 255 (full transparency)
--------------	---

4.37.2.10 **BOOLEAN STB_EBUTT_Show (E_EBUTT_CHARACTER_SET_DESIGNATION character_set_designation, BOOLEAN navigate_to_index_page, BOOLEAN show_header)**

Called to initiate continuous display of TeleText page content. This will result in either the index page defined in the service data stream being selected, or the previously displayed page (prior to a call to [STB_EBUTT_Hide\(\)](#) being made) from being reinstated. This function can be called irrespective of the use of the [STB_EBUTT_Start\(\)](#) / [STB_EBUTT_Stop\(\)](#) functions, but of course may result in no TeleText data being displayed if none already exists within the cache. The return value indicates whether initialisation of the display mechanism has been successful or not.

Parameters

<i>E_STB_EBUTT_CHARACTER_SET_DESIGNATION</i>	character_set_designation - used to specify the default character set designation for a level 1.0 Teletext broadcast. This is useful to configure the implementation to work correctly when used with regionalised broadcasts.
<i>BOOLEAN</i>	navigate_to_index_page - set it TRUE if the index page defined within the Teletext service is used, or the last visited page is re-displayed.

Returns

TRUE if initialisation of display mechanism was successful, FALSE otherwise.

4.37.2.11 **void STB_EBUTT_Start (U8BIT path, U16BIT text_pid, U8BIT magazine, U8BIT page)**

These functions are called to control whether received TeleText data is processed. - Using this functionality ensures that the process of collating page information can be halted when processing resources are a consideration. It is envisaged that some implementations may wish only to collate TeleText content only when the display is being shown, whilst other designs may wish to cache significant page content as a background process. Abstracting this functionality from the initialisation routines gives the option to support both solution types.

Parameters

<i>BOOLEAN</i>	reset_cache - clears all pages from the cache if set to TRUE, saving memory resources.
----------------	--

4.38 middleware/stb/inc/stberc.h File Reference

Header file - Function prototypes for Event Reporting.

Defines

- #define **EV_CLASS_MASK** 0x007f
- #define **EV_CLASS_CAN_REPEAT_FLAG** 0x0080
- #define **HARDWARE_EVENT**(id) (id & EV_CLASS_MASK)
- #define **USER_EVENT**(id) ((id & EV_CLASS_MASK) | EV_CLASS_CAN_REPEAT_FLAG)
- #define **EVENT_CODE**(class, type) (((class) << 16) | (type))
- #define **EVENT_CLASS**(code) ((code >> 16) & 0xff)
- #define **EVENT_TYPE**(code) (code & 0xffff)
- #define **EV_CLASS_HSET** USER_EVENT(0x01)
- #define **EV_CLASS_KEYP** USER_EVENT(0x02)
- #define **EV_CLASS_TUNE** HARDWARE_EVENT(0x03)
- #define **EV_CLASS_DECODE** HARDWARE_EVENT(0x04)
- #define **EV_CLASS_SEARCH** HARDWARE_EVENT(0x05)
- #define **EV_CLASS_LNB** HARDWARE_EVENT(0x06)
- #define **EV_CLASS_DISH** HARDWARE_EVENT(0x07)
- #define **EV_CLASS_SKEW** HARDWARE_EVENT(0x08)
- #define **EV_CLASS_SCART** HARDWARE_EVENT(0x09)
- #define **EV_CLASS_UI** HARDWARE_EVENT(0x0a)
- #define **EV_CLASS_OTA_SW_UPGRADE** HARDWARE_EVENT(0x0b)
- #define **EV_CLASS_MHEG** HARDWARE_EVENT(0xc)
- #define **EV_CLASS_MHEG_TUNE_REQUEST** HARDWARE_EVENT(0xd)
- #define **EV_CLASS_PVR** HARDWARE_EVENT(0xe)
- #define **EV_CLASS_TIMER** HARDWARE_EVENT(0xf)
- #define **EV_CLASS_APPLICATION** HARDWARE_EVENT(0x10)
- #define **EV_CLASS_DVD** HARDWARE_EVENT(0x12)
- #define **EV_CLASS_MHEG_DISPLAY_INFO** HARDWARE_EVENT(0x13)
- #define **EV_CLASS_CI** HARDWARE_EVENT(0x14)
- #define **EV_CLASS_DISK** HARDWARE_EVENT(0x15)
- #define **EV_CLASS_HDMI** HARDWARE_EVENT(0x16)
- #define **EV_CLASS_MHEG_STREAMING_REQUEST** HARDWARE_EVENT(0x17)
- #define **EV_CLASS_MHEG_ACTION_REQUEST** HARDWARE_EVENT(0x18)
- #define **EV_CLASS_CA** HARDWARE_EVENT(0x19)
- #define **EV_CLASS_PRIVATE** HARDWARE_EVENT(0xFF)
- #define **EV_TYPE_FAIL** 0x0000
- #define **EV_TYPE_SUCCESS** 0x0001
- #define **EV_TYPE_FORCE** 0x0002
- #define **EV_TYPE_LOCKED** 0x0003
- #define **EV_TYPE_NOTLOCKED** 0x0004
- #define **EV_TYPE_STARTED** 0x0005

- `#define EV_TYPE_STOPPED 0x0006`
- `#define EV_TYPE_AUDIO_STARTED 0x0007`
- `#define EV_TYPE_AUDIO_STOPPED 0x0008`
- `#define EV_TYPE_VIDEO_STARTED 0x0009`
- `#define EV_TYPE_VIDEO_STOPPED 0x000a`
- `#define EV_TYPE_PAUSED 0x000b`
- `#define EV_TYPE_16_9 0x000c`
- `#define EV_TYPE_4_3 0x000d`
- `#define EV_TYPE_SIGNAL_DATA_BAD 0x000e`
- `#define EV_TYPE_SIGNAL_DATA_OK 0x000f`
- `#define EV_TYPE_DISCONNECTED 0x0010`
- `#define EV_TYPE_SAMPLE_STARTED 0x0011`
- `#define EV_TYPE_SAMPLE_STOPPED 0x0012`
- `#define EV_TYPE_AD_STARTED 0x0013`
- `#define EV_TYPE_AD_STOPPED 0x0014`
- `#define EV_TYPE_CONNECTED 0x0015`
- `#define EV_TYPE_AUDIO_UNDERFLOW 0x0016`
- `#define EV_TYPE_VIDEO_UNDERFLOW 0x0017`
- `#define EV_TYPE_UPDATE 0x0018`
- `#define EV_TYPE_FORCED_SERVICE_CHANGE 0x0019`
- `#define EV_TYPE_OTA_SW_UPGRADE_FOUND 0x3000`
- `#define EV_TYPE_OTA_SW_UPGRADE_NOTFOUND 0x3001`
- `#define EV_TYPE_OTA_SW_UPGRADE_DOWNLOADING 0x3002`
- `#define EV_TYPE_OTA_SW_UPGRADE_ERROR 0x3003`
- `#define EV_TYPE_OTA_SW_UPGRADE_BURNING 0x3004`
- `#define EV_TYPE_OTA_SW_UPGRADE_COMPLETED 0x3005`
- `#define EV_TYPE_START_SUBTITLES 0x4001`
- `#define EV_TYPE_STOP_SUBTITLES 0x4002`
- `#define EV_TYPE_MHEG_TUNE_INDEX 0x4003`
- `#define EV_TYPE_MHEG_TUNE_DSD 0x4004`
- `#define EV_TYPE_MHEG_TUNE_TRIPLET 0x4005`
- `#define EV_TYPE_TERR_NETWORK_CHANGED 0x5000`
- `#define EV_TYPE_PVR_REC_START 0x6000`
- `#define EV_TYPE_PVR_REC_STOP 0x6001`
- `#define EV_TYPE_PVR_PLAY_START 0x6002`
- `#define EV_TYPE_PVR_PLAY_STOP 0x6003`
- `#define EV_TYPE_PVR_PLAY_BOF 0x6004`
- `#define EV_TYPE_PVR_PLAY_EOF 0x6005`
- `#define EV_TYPE_DVD_DISK_INSERTED 0x7000`
- `#define EV_TYPE_DVD_DISK_REMOVED 0x7001`
- `#define EV_TYPE_MHEG_TUNE_NORMALLY 0x8000`
- `#define EV_TYPE_MHEG_TUNE_QUIETLY 0x8001`
- `#define EV_TYPE_MHEG_PROMO_LINK_CHANGE 0x8002`
- `#define EV_TYPE_CI_INSERT 0x9001`
- `#define EV_TYPE_CI_REMOVE 0x9002`
- `#define EV_TYPE_CI_CAM_UPGRADE_PROGRESS 0x9003`

- `#define EV_TYPE_CI_CAM_UPGRADE_COMPLETE 0x9004`
- `#define EV_TYPE_CI_CAM_UPGRADE_FAILED 0x9005`
- `#define EV_TYPE_CI_SLOT_STATUS_UPDATED 0x9006`
- `#define EV_TYPE_CI_URI_UPDATED 0x9007`
- `#define EV_TYPE_CI_KEYS_UPDATED 0x9008`
- `#define EV_TYPE_CI_OPEN 0x9009`
- `#define EV_TYPE_CI_CLOSE 0x900A`
- `#define EV_TYPE_CI_TUNE 0x900B`
- `#define EV_TYPE_CI_RELEASE_REPLY 0x900C`
- `#define EV_TYPE_CI_REQUEST_OPERATOR_SEARCH 0x900D`
- `#define EV_TYPE_CI_OPERATOR_SEARCH_FINISHED 0x900E`
- `#define EV_TYPE_CI_RECORD_START 0x900F`
- `#define EV_TYPE_CI_RECORD_START_FAILED 0x9010`
- `#define EV_TYPE_CI_RECORD_LICENCE_UPDATED 0x9011`
- `#define EV_TYPE_CI_PLAYBACK_LICENCE_UPDATED 0x9012`
- `#define EV_TYPE_CI_PLAYBACK_LICENCE_STATUS 0x9013`
- `#define EV_TYPE_CI_PIN_STATUS 0x9014`
- `#define EV_TYPE_CI_RECORD_PIN 0x9015`
- `#define EV_TYPE_CI_PLAYBACK_BLANK_VIDEO 0x9016`
- `#define EV_TYPE_CI_PLAYBACK_PIN_STATUS 0x9017`
- `#define EV_TYPE_DISK_CONNECTED 0xA000`
- `#define EV_TYPE_DISK_REMOVED 0xA001`
- `#define EV_TYPE_DISK_FULL 0xA002`
- `#define EV_TYPE_MHEG_STREAMING_SETUP 0xB001`
- `#define EV_TYPE_MHEG_STREAMING_STOP 0xB002`
- `#define EV_TYPE_MHEG_STREAMING_PAUSE 0xB003`
- `#define EV_TYPE_MHEG_PIN_REQUEST 0xB004`
- `#define EV_TYPE_MHEG_REQUEST_VIDEO 0xC001`
- `#define EV_TYPE_MHEG_REQUEST_AUDIO 0xC002`
- `#define EV_TYPE_MHEG_REQUEST_RESTORE_VIDEO 0xC003`
- `#define EV_TYPE_MHEG_REQUEST_RESTORE_AUDIO 0xC004`
- `#define STB_EVENT_TUNE_LOCKED EVENT_CODE(EV_CLASS_TUNE, EV_TYPE_LOCKED)`
- `#define STB_EVENT_TUNE_NOTLOCKED EVENT_CODE(EV_CLASS_TUNE, EV_TYPE_NOTLOCKED)`
- `#define STB_EVENT_TUNE_STOPPED EVENT_CODE(EV_CLASS_TUNE, EV_TYPE_STOPPED)`
- `#define STB_EVENT_TUNE_SIGNAL_DATA_BAD EVENT_CODE(EV_CLASS_TUNE, EV_TYPE_SIGNAL_DATA_BAD)`
- `#define STB_EVENT_TUNE_SIGNAL_DATA_OK EVENT_CODE(EV_CLASS_TUNE, EV_TYPE_SIGNAL_DATA_OK)`
- `#define STB_EVENT_SEARCH_FAIL EVENT_CODE(EV_CLASS_SEARCH, EV_TYPE_FAIL)`
- `#define STB_EVENT_SEARCH_SUCCESS EVENT_CODE(EV_CLASS_SEARCH, EV_TYPE_SUCCESS)`
- `#define STB_EVENT_AD_DECODE_STARTED EVENT_CODE(EV_CLASS_DECODE, EV_TYPE_AD_STARTED)`

- **#define STB_EVENT_AD_DECODE_STOPPED** EVENT_CODE(EV_CLASS_DECODE, EV_TYPE_AD_STOPPED)
- **#define STB_EVENT_AUDIO_DECODE_STARTED** EVENT_CODE(EV_CLASS_DECODE, EV_TYPE_AUDIO_STARTED)
- **#define STB_EVENT_AUDIO_DECODE_STOPPED** EVENT_CODE(EV_CLASS_DECODE, EV_TYPE_AUDIO_STOPPED)
- **#define STB_EVENT_AUDIO_DECODE_UNDERFLOW** EVENT_CODE(EV_CLASS_DECODE, EV_TYPE_AUDIO_UNDERFLOW)
- **#define STB_EVENT_VIDEO_DECODE_STARTED** EVENT_CODE(EV_CLASS_DECODE, EV_TYPE_VIDEO_STARTED)
- **#define STB_EVENT_VIDEO_DECODE_STOPPED** EVENT_CODE(EV_CLASS_DECODE, EV_TYPE_VIDEO_STOPPED)
- **#define STB_EVENT_VIDEO_DECODE_UNDERFLOW** EVENT_CODE(EV_CLASS_DECODE, EV_TYPE_VIDEO_UNDERFLOW)
- **#define STB_EVENT_SAMPLE_DECODE_STARTED** EVENT_CODE(EV_CLASS_DECODE, EV_TYPE_SAMPLE_STARTED)
- **#define STB_EVENT_SAMPLE_DECODE_STOPPED** EVENT_CODE(EV_CLASS_DECODE, EV_TYPE_SAMPLE_STOPPED)
- **#define STB_EVENT_DECODE_LOCKED** EVENT_CODE(EV_CLASS_DECODE, EV_TYPE_LOCKED)
- **#define STB_EVENT_FORCED_SERVICE_CHANGE** EVENT_CODE(EV_CLASS_APPLICATION, EV_TYPE_FORCED_SERVICE_CHANGE)
- **#define STB_EVENT_TIMER_NOTIFY** EVENT_CODE(EV_CLASS_TIMER, EV_TYPE_FAIL)
- **#define STB_EVENT_TIMER_EXPIRE** EVENT_CODE(EV_CLASS_TIMER, EV_TYPE_SUCCESS)
- **#define STB_EVENT_HDMI_CONNECTED** EVENT_CODE(EV_CLASS_HDMI, EV_TYPE_CONNECTED)
- **#define STB_EVENT_HDMI_DISCONNECTED** EVENT_CODE(EV_CLASS_HDMI, EV_TYPE_DISCONNECTED)
- **#define STB_EVENT_OTA_SW_UPGRADE_FOUND** EVENT_CODE(EV_CLASS_OTA_SW_UPGRADE, EV_TYPE_OTA_SW_UPGRADE_FOUND)
- **#define STB_EVENT_OTA_SW_UPGRADE_NOTFOUND** EVENT_CODE(EV_CLASS_OTA_SW_UPGRADE, EV_TYPE_OTA_SW_UPGRADE_NOTFOUND)
- **#define STB_EVENT_OTA_SW_UPGRADE_DOWNLOADING** EVENT_CODE(EV_CLASS_OTA_SW_UPGRADE, EV_TYPE_OTA_SW_UPGRADE_DOWNLOADING)
- **#define STB_EVENT_OTA_SW_UPGRADE_BURNING** EVENT_CODE(EV_CLASS_OTA_SW_UPGRADE, EV_TYPE_OTA_SW_UPGRADE_BURNING)
- **#define STB_EVENT_OTA_SW_UPGRADE_RESET_REQD** EVENT_CODE(EV_CLASS_OTA_SW_UPGRADE, EV_TYPE_OTA_SW_UPGRADE_RESET_REQD)
- **#define STB_EVENT_OTA_SW_UPGRADE_ERROR** EVENT_CODE(EV_CLASS_OTA_SW_UPGRADE, EV_TYPE_OTA_SW_UPGRADE_ERROR)
- **#define STB_EVENT_OTA_SW_UPGRADE_COMPLETED** EVENT_CODE(EV_CLASS_OTA_SW_UPGRADE, EV_TYPE_OTA_SW_UPGRADE_COMPLETED)

- `#define STB_EVENT_CI_OPEN_MODULE EVENT_CODE(EV_CLASS_CI, EV_TYPE_CI_OPEN)`
- `#define STB_EVENT_CI_CLOSE_MODULE EVENT_CODE(EV_CLASS_CI, EV_TYPE_CI_CLOSE)`
- `#define STB_EVENT_CI_REMOVE EVENT_CODE(EV_CLASS_CI, EV_TYPE_CI_REMOVE)`
- `#define STB_EVENT_CI_INSERT EVENT_CODE(EV_CLASS_CI, EV_TYPE_CI_INSERT)`
- `#define STB_EVENT_CI_CAM_UPGRADE_PROGRESS EVENT_CODE(EV_CLASS_CI, EV_TYPE_CI_CAM_UPGRADE_PROGRESS)`
- `#define STB_EVENT_CI_CAM_UPGRADE_FAILED EVENT_CODE(EV_CLASS_CI, EV_TYPE_CI_CAM_UPGRADE_FAILED)`
- `#define STB_EVENT_CI_CAM_UPGRADE_COMPLETE EVENT_CODE(EV_CLASS_CI, EV_TYPE_CI_CAM_UPGRADE_COMPLETE)`
- `#define STB_EVENT_CI_SLOT_STATUS_CHANGED EVENT_CODE(EV_CLASS_CI, EV_TYPE_CI_SLOT_STATUS_UPDATED)`
- `#define STB_EVENT_CI_URI_CHANGED EVENT_CODE(EV_CLASS_CI, EV_TYPE_CI_URI_UPDATED)`
- `#define STB_EVENT_CI_KEYS_CHANGED EVENT_CODE(EV_CLASS_CI, EV_TYPE_CI_KEYS_UPDATED)`
- `#define STB_EVENT_CI_TUNE_REQUEST EVENT_CODE(EV_CLASS_CI, EV_TYPE_CI_TUNE)`
- `#define STB_EVENT_CI_RELEASE_REPLY EVENT_CODE(EV_CLASS_CI, EV_TYPE_CI_RELEASE_REPLY)`
- `#define STB_EVENT_CI_REQUEST_OPERATOR_SEARCH EVENT_CODE(EV_CLASS_CI, EV_TYPE_CI_REQUEST_OPERATOR_SEARCH)`
- `#define STB_EVENT_CI_OPERATOR_SEARCH_FINISHED EVENT_CODE(EV_CLASS_CI, EV_TYPE_CI_OPERATOR_SEARCH_FINISHED)`
- `#define STB_EVENT_CI_RECORD_START EVENT_CODE(EV_CLASS_CI, EV_TYPE_CI_RECORD_START)`
- `#define STB_EVENT_CI_RECORD_START_FAILED EVENT_CODE(EV_CLASS_CI, EV_TYPE_CI_RECORD_START_FAILED)`
- `#define STB_EVENT_CI_RECORD_LICENCE_CHANGED EVENT_CODE(EV_CLASS_CI, EV_TYPE_CI_RECORD_LICENCE_UPDATED)`
- `#define STB_EVENT_CI_PLAYBACK_LICENCE_CHANGED EVENT_CODE(EV_CLASS_CI, EV_TYPE_CI_PLAYBACK_LICENCE_UPDATED)`
- `#define STB_EVENT_CI_PLAYBACK_LICENCE_STATUS EVENT_CODE(EV_CLASS_CI, EV_TYPE_CI_PLAYBACK_LICENCE_STATUS)`
- `#define STB_EVENT_CI_PIN_STATUS EVENT_CODE(EV_CLASS_CI, EV_TYPE_CI_PIN_STATUS)`
- `#define STB_EVENT_CI_RECORD_PIN EVENT_CODE(EV_CLASS_CI, EV_TYPE_CI_RECORD_PIN)`
- `#define STB_EVENT_CI_PLAYBACK_BLANK_VIDEO EVENT_CODE(EV_CLASS_CI, EV_TYPE_CI_PLAYBACK_BLANK_VIDEO)`
- `#define STB_EVENT_CI_PLAYBACK_PIN_STATUS EVENT_CODE(EV_CLASS_CI, EV_TYPE_CI_PLAYBACK_PIN_STATUS)`
- `#define STB_EVENT_PVR_REC_START EVENT_CODE(EV_CLASS_PVR, EV_TYPE_PVR_REC_START)`

- **#define STB_EVENT_PVR_REC_STOP** EVENT_CODE(EV_CLASS_PVR, EV_TYPE_PVR_REC_STOP)
- **#define STB_EVENT_PVR_PLAY_START** EVENT_CODE(EV_CLASS_PVR, EV_TYPE_PVR_PLAY_START)
- **#define STB_EVENT_PVR_PLAY_STOP** EVENT_CODE(EV_CLASS_PVR, EV_TYPE_PVR_PLAY_STOP)
- **#define STB_EVENT_PVR_PLAY_BOF** EVENT_CODE(EV_CLASS_PVR, EV_TYPE_PVR_PLAY_BOF)
- **#define STB_EVENT_PVR_PLAY_EOF** EVENT_CODE(EV_CLASS_PVR, EV_TYPE_PVR_PLAY_EOF)
- **#define STB_EVENT_DISK_CONNECTED** EVENT_CODE(EV_CLASS_DISK, EV_TYPE_DISK_CONNECTED)
- **#define STB_EVENT_DISK_REMOVED** EVENT_CODE(EV_CLASS_DISK, EV_TYPE_DISK_REMOVED)
- **#define STB_EVENT_DISK_FULL** EVENT_CODE(EV_CLASS_DISK, EV_TYPE_DISK_FULL)
- **#define UI_EVENT_UPDATE** EVENT_CODE(EV_CLASS_UI, EV_TYPE_UPDATE)
- **#define STB_EVENT_MHEG_PIN_REQUEST** EVENT_CODE(EV_CLASS_MHEG, EV_TYPE_MHEG_PIN_REQUEST)
- **#define STB_EVENT_MHEG_START_SUBTITLES** EVENT_CODE(EV_CLASS_MHEG, EV_TYPE_START_SUBTITLES)
- **#define STB_EVENT_MHEG_STOP_SUBTITLES** EVENT_CODE(EV_CLASS_MHEG, EV_TYPE_STOP_SUBTITLES)
- **#define STB_EVENT_MHEG_LIFE_CYCLE** EVENT_CODE(EV_CLASS_MHEG, EV_TYPE_MHEG_LIFE_CYCLE)
- **#define STB_EVENT_MHEG_TUNE_DSD** EVENT_CODE(EV_CLASS_MHEG, EV_TYPE_MHEG_TUNE_DSD)
- **#define STB_EVENT_MHEG_TUNE_TRIPLET** EVENT_CODE(EV_CLASS_MHEG, EV_TYPE_MHEG_TUNE_TRIPLET)

Functions

- void **STB_ERInitialise** (void)
- void **STB_ERRegisterHandler** (BOOLEAN(*func)(BOOLEAN latched, BOOLEAN repeat, U16BIT event_class, U16BIT event_type, void *data, U32BIT data_size))
- void **STB_ERNotifyEvent** (U8BIT event_class, U16BIT event_type)
- void **STB_ERSendEvent** (BOOLEAN latched, BOOLEAN repeat, U16BIT path_class, U16BIT type, void *data, U32BIT data_size)

4.38.1 Detailed Description

Header file - Function prototypes for Event Reporting.

Date

15/02/2001

4.39 middleware/stb/inc/stbgc.h File Reference

Header file - macros and function prototypes for public use.

This graph shows which files directly or indirectly include this file:

Defines

- `#define D_DSHFT 17`
- `#define D_HSHFT 12`
- `#define D_MSHFT 6`
- `#define DHMS_CREATE(dy, hr, mn, sc) ((U32DHMS)dy << D_DSHFT | (U32DHMS)hr << D_HSHFT | (U32DHMS)mn << D_MSHFT | (sc))`
- `#define DHMS_DAYS(dhms) (dhms >> D_DSHFT)`
- `#define DHMS_DATE32(dhms) ((dhms >> D_DSHFT) | 0x8000)`
- `#define DHMS_HOUR32(dhms) ((dhms >> D_HSHFT) & 0x1f)`
- `#define DHMS_MINS32(dhms) ((dhms >> D_MSHFT) & 0x3f)`
- `#define DHMS_SECS32(dhms) (dhms & 0x3f)`
- `#define DHMS_DATE(dhms) (U16BIT)DHMS_DATE32(dhms)`
- `#define DHMS_HOUR(dhms) (U8BIT) DHMS_HOUR32(dhms)`
- `#define DHMS_MINS(dhms) (U8BIT) DHMS_MINS32(dhms)`
- `#define DHMS_SECS(dhms) (U8BIT) DHMS_SECS32(dhms)`

Typedefs

- `typedef U32BIT U32DHMS`

Enumerations

- `enum E_STB_GC_WEEKDAY { WEEKDAY_MONDAY = 1, WEEKDAY_TUESDAY = 2, WEEKDAY_WEDNESDAY = 3, WEEKDAY_THURSDAY = 4, WEEKDAY_FRIDAY = 5, WEEKDAY_SATURDAY = 6, WEEKDAY_SUNDAY = 7 }`
- `enum E_STB_GC_CALCTYPE { CALC_ADD = 0, CALC_SUB = 1 }`
- `enum E_STB_GC_CONVTYPE { CONV_LOCAL = 0, CONV_GMT = 1 }`
- `enum E_STB_GC_COMPTYPE { COMP_MATCH = 0, COMP_1GT2 = 1, COMP_1GE2 = 2, COMP_1LT2 = 3, COMP_1LE2 = 4, COMP_2GT1 = 5, COMP_2GE1 = 6, COMP_2LT1 = 7, COMP_2LE1 = 8 }`
- `enum E_STB_GC_TIMETYPE { TIME_12H = 0, TIME_24H = 1 }`
- `enum E_STB_GC_DATETYPE { DATE_DMY = 0, DATE_YMD = 1 }`
- `enum E_STB_GC_CLOCKTYPE { CLOCK_HMS = 0, CLOCK_SMH = 1, CLOCK_HM = 2, CLOCK_MH = 3 }`

Functions

- void **STB_GCInitialise** (void)
- void **STB_GCSetSearchLangCode** (U32BIT lang)
- U32BIT **STB_GCGetSearchLangCode** (void)
- U8BIT * **STB_GCGetLangCodeString** (U32BIT lang)
- void **STB_GCSetCiStandby** (BOOLEAN standby)
- U8BIT * **STB_GCGetFullSerialString** (void)
- U8BIT * **STB_GCGetVersionNumberString** (void)
- void **STB_GCSetAudioSignal** (BOOLEAN state)
- BOOLEAN **STB_GCGetAudioSignal** (void)
- void **STB_GCSetLocalTimeOffset** (U8BIT ohour, U8BIT omin, BOOLEAN neg)
- void **STB_GCGetLocalTimeOffset** (U8BIT *ohour, U8BIT *omin, BOOLEAN *neg)
- void **STB_GCSetLocalTimeChange** (U16BIT code, U8BIT hour, U8BIT min, U8BIT secs, U8BIT ohour1, U8BIT omin1, U8BIT ohour2, U8BIT omin2, BOOLEAN neg)
- void **STB_GCGetLocalTimeChange** (U16BIT code, U8BIT hour, U8BIT min, - U8BIT secs, U8BIT *ohour, U8BIT *omin, BOOLEAN *neg)
- void **STB_GCSetGMTTime** (U8BIT hour, U8BIT min, U8BIT secs)
- U8BIT **STB_GCGetGMTHour** (void)
- U8BIT **STB_GCGetGMTMin** (void)
- U8BIT **STB_GCGetGMTSecs** (void)
- void **STB_GCSetGMTDate** (U16BIT code)
- U16BIT **STB_GCGetGMTDate** (void)
- E_STB_GC_WEEKDAY **STB_GCGetGMTWeekDay** (void)
- U8BIT **STB_GCGetGMTDay** (void)
- U8BIT **STB_GCGetGMTMonth** (void)
- U16BIT **STB_GCGetGMTYear** (void)
- void **STB_GCGetGMTDateTime** (U16BIT *code, U8BIT *hour, U8BIT *min, U8BIT *secs)
- BOOLEAN **STB_GCIsFutureDateTime** (U16BIT code, U8BIT hour, U8BIT min, U8BIT secs)
- E_STB_GC_WEEKDAY **STB_GCGetDateWeekDay** (U16BIT code)
- BOOLEAN **STB_GCIsDateDayWeek** (U16BIT code)
- BOOLEAN **STB_GCIsDateDayWeekend** (U16BIT code)
- void **STB_GCCalculateDateTime** (U16BIT code, U8BIT hour, U8BIT min, U8BIT secs, U8BIT ohour, U8BIT omin, U8BIT osecs, U16BIT *rcode, U8BIT *rhour, U8BIT *rmin, U8BIT *rsecs, E_STB_GC_CALCTYPE calc)
- void **STB_GCConvertDateTime** (U16BIT code, U8BIT hour, U8BIT min, U8BIT secs, U16BIT *rcode, U8BIT *rhour, U8BIT *rmin, U8BIT *rsecs, E_STB_GC_CONVTYPE conv)
- BOOLEAN **STB_GCCompareDateTime** (U16BIT code1, U8BIT hour1, U8BIT min1, U8BIT secs1, U16BIT code2, U8BIT hour2, U8BIT min2, U8BIT secs2, E_STB_GC_COMPTYPE comp)
- S32BIT **STB_GCDateTimeDiff** (U16BIT code1, U8BIT hour1, U8BIT min1, U8BIT secs1, U16BIT code2, U8BIT hour2, U8BIT min2, U8BIT secs2)

- U8BIT * **STB_GCGetTimeString** (U16BIT code, U8BIT hour, U8BIT min, E_STB_GC_TIMETYPE format)
- U8BIT * **STB_GCGetDateString** (U16BIT code, U8BIT hour, U8BIT min, E_STB_GC_DATETYPE format)
- U8BIT * **STB_GCGetClockString** (U8BIT hour, U8BIT min, U8BIT secs, E_STB_GC_CLOCKTYPE format)
- void **STB_GCGetDateInfo** (U16BIT code, U8BIT hour, U8BIT min, U8BIT *day, U8BIT *wday, U8BIT *month, U16BIT *year)
- void **STB_GCGetMJDDateInfo** (U16BIT code, U8BIT *day, U8BIT *wday, U8BIT *month, U16BIT *year)
- void **STB_GCGetLocalDateTime** (U16BIT *year, U8BIT *month, U8BIT *day, U8BIT *hour, U8BIT *mins)
- void **STB_GCUseBroadcastTime** (BOOLEAN state)
Sets whether the date/time are taken from the broadcast or the system. The default is to use the broadcast for date/time.
- U32DHMS **STB_GCCreateDebugDHMS** (U32BIT date, U32BIT hour, U32BIT mins, U32BIT secs)
- U32DHMS **STB_GCCreateDHMS** (U16BIT date, U8BIT hour, U8BIT mins, U8BIT secs)
- U32DHMS **STB_GCCalculateDHMS** (U32DHMS dhms, U32DHMS period, E_STB_GC_CALCTYPE calc)
Calculates the date/time when the period is added/subtracted to/from dhms.
- U32DHMS **STB_GCConvertDHMS** (U32DHMS dhms, E_STB_GC_CONVTYPE conv)
Converts the given date/time to local or GMT.
- U32DHMS **STB_GCNowDHMSGmt** (void)
- U32DHMS **STB_GCNowDHMSLocal** (void)
- U8BIT * **STB_GCGetDateStringDHMS** (U32DHMS dhms, E_STB_GC_DATETYPE format)
- U8BIT * **STB_GCGetTimeStringDHMS** (U32DHMS dhms, E_STB_GC_TIMETYPE format)
- U32DHMS **STB_GCConvertTimestamp** (U32BIT timestamp)
Converts a timestamp expressed in number of seconds since midnight (UTC) 1 - January 1970.
- U32BIT **STB_GCConvertToTimestamp** (U32DHMS time)
Returns the number of seconds from midnight (UTC) 1 January 1970 to the specified U32DHMS time.

4.39.1 Detailed Description

Header file - macros and function prototypes for public use.

Date

13/10/2000

4.39.2 Function Documentation

4.39.2.1 U32DHMS STB_GCCalculateDHMS (U32DHMS *dhms*, U32DHMS *period*, E_STB_GC_CALCTYPE *calc*)

Calculates the date/time when the period is added/subtracted to/from dhms.

Parameters

<i>dhms</i>	base date/time
<i>period</i>	days/hours/mins/secs to be added or subtracted
<i>calc</i>	calculation to be performed CALC_ADD adds period to dhms CALC_-SUB subtracts period from dhms

Returns

resulting date/time following the calculation

4.39.2.2 U32DHMS STB_GCConvertDHMS (U32DHMS *dhms*, E_STB_GC_CONVTYPE *conv*)

Converts the given date/time to local or GMT.

Parameters

<i>dhms</i>	date/time to be converted
<i>conv</i>	conversion to be performed CONV_LOCAL = convert to local CONV_-GMT = convert to GMT

Returns

converted date/time

4.39.2.3 U32DHMS STB_GCConvertTimestamp (U32BIT *timestamp*)

Converts a timestamp expressed in number of seconds since midnight (UTC) 1 January 1970.

Parameters

<i>timestamp</i>	number of seconds since midnight (UTC) 1 January 1970
------------------	---

Returns

time expressed as U32DHMS

4.39.2.4 U32BIT STB_GCConvertToTimestamp (U32DHMS *time*)

Returns the number of seconds from midnight (UTC) 1 January 1970 to the specified U32DHMS time.

Parameters

<i>time</i>	time in U32DHMS format
-------------	------------------------

4.39.2.5 void STB_GCUseBroadcastTime (BOOLEAN *state*)

Sets whether the date/time are taken from the broadcast or the system. The default is to use the broadcast for date/time.

Parameters

<i>state</i>	- FALSE to use date/time from the system
--------------	--

4.40 middleware/stb/inc/stbheap.h File Reference

Header file - Function prototypes for heap memory.

Functions

- void **STB_HeapInitialise** (void)
- void * **STB_GetMemory** (U32BIT bytes)
- void **STB_FreeMemory** (void *addr)
- void * **STB_ResizeMemory** (void *ptr, U32BIT new_num_bytes)
- void * **STB_AppGetMemory** (U32BIT bytes)
- void **STB_AppFreeMemory** (void *addr)
- void * **STB_AppResizeMemory** (void *ptr, U32BIT new_num_bytes)
- void **STB_AppRegisterCacheFreeFunction** (BOOLEAN(*callback_function)(void))
- void **STB_GetHeapStats** (U32BIT *total_app, U32BIT *max_app, U32BIT *num_app, U32BIT *total_mem, U32BIT *max_mem, U32BIT *num_mem)

4.40.1 Detailed Description

Header file - Function prototypes for heap memory.

Date

06/09/2000

4.41 middleware/stb/inc/stbinit.h File Reference

Header file - macros and function prototypes for public use.

Typedefs

- typedef enum e_stb_ttxt_subt_control **E_STB_TTXT_SUBT_CONTROL**

Enumerations

- enum **e_stb_ttxt_subt_control** { **NO_TELETEXT_OR_SUBTITLES**, **EBU_SUBTITLES**, **DVB_SUBTITLES**, **DVB_SUBTITLES_AND_TELETEXT**, **DECODED_TELETEXT_WITH_VBI**, **DECODED_TELETEXT_WITHOUT_VBI**, **VBI_TELETEXT** }

Functions

- void **STB_Initialise** (E_STB_TTXT_SUBT_CONTROL ttxt_subt_cntrl)

4.41.1 Detailed Description

Header file - macros and function prototypes for public use.

Date

13/10/2000

4.42 middleware/stb/inc/stbllist.h File Reference

Header file - Function prototypes for linked lists.

This graph shows which files directly or indirectly include this file:

Data Structures

- struct [LINK_LIST_PTR_BLK](#)
- struct [LINK_LIST_HEADER](#)

Defines

- #define **CREATE_LINK_LIST_HEADER**(list) static [LINK_LIST_HEADER](#) list = {NULL, &list, &list}

Functions

- void **STB_LLInitialiseHeader** ([LINK_LIST_HEADER](#) *hdr)
- void **STB_LLAddBlockToEnd** ([LINK_LIST_HEADER](#) *hdr, [LINK_LIST_PTR_BLK](#) *new_blk)
- void **STB_LLAddBlockToStart** ([LINK_LIST_HEADER](#) *hdr, [LINK_LIST_PTR_BLK](#) *new_blk)
- void **STB_LLAddBlockBefore** ([LINK_LIST_PTR_BLK](#) *blk, [LINK_LIST_PTR_BLK](#) *new_blk)
- void **STB_LLAddBlockAfter** ([LINK_LIST_PTR_BLK](#) *blk, [LINK_LIST_PTR_BLK](#) *new_blk)

- void **STB_LLRemoveBlock** ([LINK_LIST_PTR_BLK](#) *blk)
- [LINK_LIST_PTR_BLK](#) * **STB_LLGetNextBlock** ([LINK_LIST_PTR_BLK](#) *blk)
- [LINK_LIST_PTR_BLK](#) * **STB_LLGetPrevBlock** ([LINK_LIST_PTR_BLK](#) *blk)
- [LINK_LIST_PTR_BLK](#) * **STB_LLGetFirstBlock** ([LINK_LIST_HEADER](#) *hdr)
- [LINK_LIST_PTR_BLK](#) * **STB_LLGetLastBlock** ([LINK_LIST_HEADER](#) *hdr)
- [LINK_LIST_PTR_BLK](#) * **STB_LLGetBlock** ([LINK_LIST_HEADER](#) *hdr, U16BIT num)
- U16BIT **STB_LLGetNumBlocks** ([LINK_LIST_HEADER](#) *hdr)
- BOOLEAN **STB_LLCheckBlockInList** ([LINK_LIST_HEADER](#) *hdr, [LINK_LIST_PTR_BLK](#) *blk)
- BOOLEAN **STB_LLSort** ([LINK_LIST_HEADER](#) *ll_hdr, S16BIT(*cmp_func)([LINK_LIST_PTR_BLK](#) **, [LINK_LIST_PTR_BLK](#) **))

Sorts the blocks of a link list object in ascending or descending order.

4.42.1 Detailed Description

Header file - Function prototypes for linked lists.

Date

06/09/2000

4.42.2 Function Documentation

4.42.2.1 **BOOLEAN STB_LLSort** ([LINK_LIST_HEADER](#) * ll_hdr,
S16BIT(*)([LINK_LIST_PTR_BLK](#) **, [LINK_LIST_PTR_BLK](#) **) cmp_func)

Sorts the blocks of a link list object in ascending or descending order.

----- NOTE: The order in which the blocks will be sorted is determined solely by the callback compare function NOTE: callback function must Not alter contents of linked list NOTE: STB_LLSort is NOT re-entrant -- ie. It cannot be called recursively -----
--

Parameters

ll_hdr	- pointer to the HEADER of the linked list to be sorted cmp_func - pointer to the compare function ----- -- S16BIT (*cmp_func)(LINK_LIST_PTR_BLK ** blk_1, LINK_LIST_PTR_BLK ** blk_2) ----- The callback function must take two LINK_LIST_PTR_BLK pointer to pointer params and must return an S16BIT value of:- equal to 0 -- if both compare blocks are identical less than 0 -- if block_1 has a lower ordinal position than block_2 greater than 0 -- if block_1 has a higher ordinal position than block_2 ----- NO-TE: It is the order of the ordinal positions that determines whether the linked list is sorted in ascending of descending order ----- -----
------------------------	---

Returns

TRUE if successful, FALSE if failed

4.43 middleware/stb/inc/stbpes.h File Reference

Header file - Function prototypes for PES collection task.

Functions

- void **STB_PesCollectionTaskInitialise** (void)
- void **STB_FlushPesCollectionTask** (void)
- U32BIT **STB_RegisterPesCollectionCallback** (void(*callback_function)(U32BIT, U8BIT, void *, U32BIT), U8BIT lowest_data_identifier, U8BIT highest_data_identifier)
- void **STB_UnregisterPesCollectionCallback** (U32BIT handle)
- void **STB_ChangePesCollectionPID** (U8BIT path, U16BIT text_pid)

4.43.1 Detailed Description

Header file - Function prototypes for PES collection task.

Date

19/02/2004

Author

Ocean Blue

4.44 middleware/stb/inc/stbpvr.h File Reference

Header file - macros and function prototypes for public use.

#include "stbpvrpr.h" #include "stbhwdmx.h" Include dependency graph for stbpvr.h:

Defines

- #define **STB_PVR_NAME_LEN** 255
- #define **STB_PVR_BASENAME_LEN** 9
- #define **STB_PVR_SERVICE_LEN** 32
- #define **STB_PVR_ADDITIONAL_INFO_LEN** 255
- #define **STB_PVR_NORMAL_PLAY_SPEED** 100
- #define **STB_PVR_INVALID_HANDLE** 0
- #define **STB_PVR_MAX_CRID_LEN** 65
- #define **STB_PVR_MAX_ADD_INFO_LEN** 255

Enumerations

- enum **E_STB_PVR_FORMATMODE** { **FORMAT_START** = 0, **FORMAT_PROGRESS** = 1, **FORMAT_END** = 2 }
- enum **E_STB_PVR_REPAIRMODE** { **REPAIR_ENQUIRE** = 0, **REPAIR_START** = 1, **REPAIR_PROGRESS** = 2, **REPAIR_END** = 3 }

Functions

- **BOOLEAN STB_PVRInitialise** (void)
- void **STB_PVRSetStandbyState** (BOOLEAN state)
- **BOOLEAN STB_PVRIsInitialised** (void)
- **U16BIT STB_PVRGetDefaultDisk** (void)
- **BOOLEAN STB_PVRCanDiskBeUsed** (U16BIT disk_id)
- **BOOLEAN STB_PVRFormat** (U16BIT disk_id, E_STB_PVR_FORMATMODE mode, U8BIT *prog)
- **BOOLEAN STB_PVRRepair** (U16BIT disk_id, E_STB_PVR_REPAIRMODE mode, U8BIT *prog)
- void **STB_PVRUpdateRecordings** (BOOLEAN force_load)
- **BOOLEAN STB_PVRIsValidHandle** (U32BIT handle)
- **BOOLEAN STB_PVRCreateRecording** (U16BIT disk_id, U8BIT *name, U32BIT *handle)
- **BOOLEAN STB_PVRDestroyRecording** (U32BIT handle)
- void **STB_PVRSaveRecording** (U32BIT handle)
- **U16BIT STB_PVRGetRecordingHandles** (U32BIT **handle_array)
- void **STB_PVRReleaseRecordingHandles** (U32BIT *handle_array)
- **BOOLEAN STB_PVRFindRecordingFromCrid** (U8BIT *prog_crid, U32BIT *handle)
- **BOOLEAN STB_PVRFindNextSplitRecording** (U32BIT curr_handle, U32BIT *next_handle)
- **U32BIT STB_PVRGetHandleForRecordingIndex** (U8BIT rec_index)
- **U8BIT STB_PVRGetPathForRecordingIndex** (U8BIT rec_index)
- void **STB_PVRRecordingSetName** (U32BIT handle, U8BIT *name)
- **U8BIT * STB_PVRRecordingGetName** (U32BIT handle)
- **BOOLEAN STB_PVRRecordingGetDateTime** (U32BIT handle, U16BIT *date, U8BIT *hours, U8BIT *mins, U8BIT *secs)
- **BOOLEAN STB_PVRRecordingGetLength** (U32BIT handle, U8BIT *length_hours, U8BIT *length_mins, U8BIT *length_secs, U32BIT *rec_size_kb)
- **U16BIT STB_PVRRecordingGetDiskId** (U32BIT handle)
- void **STB_PVRRecordingSetSeries** (U32BIT handle)
- void **STB_PVRRecordingSetRecommendation** (U32BIT handle)
- **BOOLEAN STB_PVRRecordingGetRecommendation** (U32BIT handle)
- **BOOLEAN STB_PVRRecordingGetSeries** (U32BIT handle)
- void **STB_PVRRecordingSetOtherCrid** (U32BIT handle, U8BIT *crid)
- **BOOLEAN STB_PVRRecordingGetOtherCrid** (U32BIT handle, U8BIT *crid, - U16BIT name_len)

- void [STB_PVRRecordingSetAdditionalInfo](#) (U32BIT handle, U8BIT *additional_info)
Sets the additional info string for a recording.
- U8BIT * [STB_PVRRecordingGetAdditionalInfo](#) (U32BIT handle)
Gets the additional info string pointer for a recording.
- void [STB_PVRRecordingSetParentalRatingAge](#) (U32BIT handle, U32BIT parental_rating)
Sets the parental rating age for the specified recording.
- U32BIT [STB_PVRRecordingGetParentalRating](#) (U32BIT handle)
Returns the parental rating age for the specified recording as set by STB_PVRRecordingSetParentalRatingAge.
- void [STB_PVRRecordingSetStartPadding](#) (U32BIT handle, S32BIT start_padding)
Sets the start padding value for the specified recording.
- S32BIT [STB_PVRRecordingGetStartPadding](#) (U32BIT handle)
Gets the start padding value for the specified recording as set by STB_PVRRecordingSetStartPadding.
- void [STB_PVRRecordingSetEndPadding](#) (U32BIT handle, S32BIT end_padding)
Sets the end padding value for the specified recording.
- S32BIT [STB_PVRRecordingGetEndPadding](#) (U32BIT handle)
Gets the end padding value for the specified recording as set by STB_PVRRecordingSetEndPadding.
- void [STB_PVRRecordingSetServiceName](#) (U32BIT handle, U8BIT *service_name)
- BOOLEAN [STB_PVRRecordingGetServiceName](#) (U32BIT handle, U8BIT *service_name, U16BIT name_len)
- void [STB_PVRRecordingSetDescription](#) (U32BIT handle, U8BIT *description)
- BOOLEAN [STB_PVRRecordingGetDescription](#) (U32BIT handle, U8BIT *description, U16BIT desc_len)
- U16BIT [STB_PVRRecordingGetDescriptionLen](#) (U32BIT handle)
- void [STB_PVRRecordingSetExtendedDescription](#) (U32BIT handle, U8BIT *description)
- BOOLEAN [STB_PVRRecordingGetExtendedDescription](#) (U32BIT handle, U8BIT *description, U16BIT desc_len)
- U16BIT [STB_PVRRecordingGetExtendedDescriptionLen](#) (U32BIT handle)
- void [STB_PVRRecordingSetCrid](#) (U32BIT handle, U8BIT *crid)
- BOOLEAN [STB_PVRRecordingGetCrid](#) (U32BIT handle, U8BIT *crid, U16BIT name_len)
- BOOLEAN [STB_PVRRecordingSetLocked](#) (U32BIT handle, BOOLEAN state)
- BOOLEAN [STB_PVRRecordingGetLocked](#) (U32BIT handle)
- BOOLEAN [STB_PVRRecordingSetSelected](#) (U32BIT handle, BOOLEAN state)
- BOOLEAN [STB_PVRRecordingGetSelected](#) (U32BIT handle)
- BOOLEAN [STB_PVRRecordingIsEncrypted](#) (U32BIT handle)
- BOOLEAN [STB_PVRRecordingSetParentalLock](#) (U32BIT handle, BOOLEAN state)

- **BOOLEAN STB_PVRRecordingGetParentalLock** (U32BIT handle)
- **BOOLEAN STB_PVRRecordingSetGuidance** (U32BIT handle, U8BIT *text)
- **BOOLEAN STB_PVRRecordingHasGuidance** (U32BIT handle)
- **BOOLEAN STB_PVRRecordingGetGuidance** (U32BIT handle, U8BIT *text, - U16BIT text_len)
- **U16BIT STB_PVRRecordingGetGuidanceLen** (U32BIT handle)
- **BOOLEAN STB_PVRRecordingSetCicamId** (U32BIT handle, U8BIT *cicam_id, U8BIT id_len)
- **BOOLEAN STB_PVRRecordingGetCicamId** (U32BIT handle, U8BIT *cicam_id, U8BIT id_len)
- **BOOLEAN STB_PVRRecordingAddURI** (U32BIT handle, U8BIT *uri)
- **BOOLEAN STB_PVRRecordingUpdateURI** (U32BIT handle, U8BIT *uri)
- **BOOLEAN STB_PVRRecordingAddLicence** (U32BIT handle, U8BIT *licence, U16BIT licence_len)
- **BOOLEAN STB_PVRRecordingUpdateLicence** (U32BIT handle, U8BIT *licence, U16BIT licence_len)
- **BOOLEAN STB_PVRRecordingAddPin** (U32BIT handle, U8BIT age_rating, - U8BIT *private_data, U16BIT date_code, U8BIT hour, U8BIT min, U8BIT secs)
- **BOOLEAN STB_PVRIsBeingRecorded** (U32BIT handle)
- **U32BIT STB_PVRGetTimeOfAllRecordings** (U16BIT disk_id)
- **U32BIT STB_PVRGetSizeOfAllRecordings** (U16BIT disk_id)
- **BOOLEAN STB_PVRToggleBookmark** (U8BIT path)
- **BOOLEAN STB_PVRGotoNextBookmark** (U8BIT path)
- **U16BIT STB_PVRGetBookmarks** (U8BIT path, void ***bookmarks)
Allocates and returns an array containing the handles of the bookmarks for the recording currently being played.
- **U16BIT STB_PVRGetBookmarksForRecording** (U32BIT handle, void ***bookmarks)
Allocates and returns an array containing the handles of the bookmarks for the recording specified by the given handle.
- **void STB_PVRReleaseBookmarks** (void **bookmarks, U16BIT num)
Frees a previously allocated array of bookmark handles.
- **void STB_PVRPlaybackNotifyTime** (U8BIT path)
- **BOOLEAN STB_PVRStartPlaying** (U8BIT path, U32BIT handle, BOOLEAN resume)
- **BOOLEAN STB_PVRIsPlaying** (U8BIT path, U32BIT *handle)
- **void STB_PVRSavePlayPosition** (U8BIT path)
Saves the bookmark holding the playback position. This bookmark is used to resume a playback when STB_PVRStartPlaying is called with resume = TRUE. This function should be called while decoders are still running, to be sure the position in the playback is accurate.
- **void STB_PVRStopPlaying** (U8BIT path)
- **BOOLEAN STB_PVRRecordingGetTriplet** (U32BIT handle, U16BIT *serv_id, U16BIT *ts_id, U16BIT *orig_net_id)
- **void STB_PVRStartPlayRunning** (U8BIT decoder)
- **void STB_PVRStartPlayPaused** (U8BIT decoder)
- **void STB_PVRStartPlaySync** (U8BIT decoder)

- **BOOLEAN STB_PVRIsPlayAudio** (U8BIT decoder)
- **BOOLEAN STB_PVRIsPlayVideo** (U32BIT handle)
- **BOOLEAN STB_PVRSetRecordingPids** (U8BIT path, U16BIT num_pids, [S_P-VR_PID_INFO](#) *pid_array)
- **BOOLEAN STB_PVRUpdateRecordingPids** (U8BIT path, U16BIT num_pids, [S_PVR_PID_INFO](#) *pid_array)
- **BOOLEAN STB_PVRSetRecoringTriplet** (U8BIT path, U16BIT serv_id, U16BIT ts_id, U16BIT orig_net_id)
- **BOOLEAN STB_PVRStartRecording** (U8BIT path, U32BIT handle)
- **BOOLEAN STB_PVRPauseRecording** (U8BIT path)
- **BOOLEAN STB_PVRResumeRecording** (U8BIT path)
- **void STB_PVRStopRecording** (U8BIT path)
- **BOOLEAN STB_PVRIsRecording** (U8BIT path, U32BIT *handle)
- **void STB_PVRStartRecordRunning** (U8BIT path)
- **void STB_PVRStartRecordPaused** (U8BIT path, U32BIT timeshift_seconds)
- **BOOLEAN STB_PVRIsRecordVideo** (U8BIT path)
- **BOOLEAN STB_PVRIsRecordAudio** (U8BIT path)
- **void STB_PVREncryptRecording** (U8BIT path, **BOOLEAN** state)
- **void STB_PVRPlayNormal** (U8BIT decoder)
- **BOOLEAN STB_PVRIsPlayNormal** (U8BIT decoder)
- **void STB_PVRPlayPause** (U8BIT decoder)
- **BOOLEAN STB_PVRIsPlayPause** (U8BIT decoder)
- **void STB_PVRPlayForward** (U8BIT decoder)
- **BOOLEAN STB_PVRIsPlayForward** (U8BIT decoder)
- **void STB_PVRPlayReverse** (U8BIT decoder)
- **BOOLEAN STB_PVRIsPlayReverse** (U8BIT decoder)
- **void STB_PVRPlayFrameInc** (U8BIT decoder)
- **void STB_PVRPlayFrameDec** (U8BIT decoder)
- **void STB_PVRPlayMedium** (U8BIT decoder)
- **BOOLEAN STB_PVRIsPlayMedium** (U8BIT decoder)
- **void STB_PVRPlaySlower** (U8BIT decoder, **BOOLEAN** include_slow_speeds)
- **BOOLEAN STB_PVRIsPlaySlowest** (U8BIT decoder)
- **void STB_PVRPlayFaster** (U8BIT decoder, **BOOLEAN** include_slow_speeds)
- **BOOLEAN STB_PVRIsPlayFastest** (U8BIT decoder)
- **S16BIT STB_PVRGetMinPlaySpeed** (U8BIT path)
- **S16BIT STB_PVRGetMaxPlaySpeed** (U8BIT path)
- **void STB_PVRDiskUsed** (U16BIT disk_id, U8BIT *hours, U8BIT *mins)
- **void STB_PVRDiskFree** (U16BIT disk_id, U8BIT *hours, U8BIT *mins)
- **U32BIT STB_PVRDiskSize** (U16BIT disk_id)
- **BOOLEAN [STB_PVRCreateBookmark](#)** (U32BIT handle, U32BIT time, U8BIT *name)
Creates a bookmark associated with the a recording.
- **BOOLEAN [STB_PVRDeleteBookmark](#)** (U32BIT handle, U32BIT time, U8BIT *name)
Deletes a bookmark associated with the a recording.
- **U32BIT [STB_PVRGetBookmarkTime](#)** (void *bookmark_handle)
Returns the time associated with a bookmark.
- **U8BIT * [STB_PVRGetBookmarkName](#)** (void *bookmark_handle)
Allocates and returns the name associated with a bookmark.

4.44.1 Detailed Description

Header file - macros and function prototypes for public use.

Date

07/02/2003

4.44.2 Function Documentation

4.44.2.1 **BOOLEAN STB_PVRCreateBookmark (U32BIT *handle*, U32BIT *time*, U8BIT * *name*)**

Creates a bookmark associated with the a recording.

Parameters

<i>handle</i>	Recording handle
<i>time</i>	Time in seconds since the beginning of the recording
<i>name</i>	NULL terminated string representing the bookmark name, maximum length STB_PVR_NAME_LEN. If name is NULL, a string is formed to represent the bookmark time in the format hh:mm:ss

Returns

TRUE if the bookmark was successfully created, FALSE otherwise.

4.44.2.2 **BOOLEAN STB_PVRDeleteBookmark (U32BIT *handle*, U32BIT *time*, U8BIT * *name*)**

Deletes a bookmark associated with the a recording.

Parameters

<i>handle</i>	Recording handle
<i>time</i>	Time in seconds since the beginning of the recording
<i>name</i>	NULL terminated string representing the bookmark name, maximum length STB_PVR_NAME_LEN. If name is NULL, a string is formed to represent the bookmark time in the format hh:mm:ss

Returns

TRUE if the bookmark was successfully deleted, FALSE otherwise.

4.44.2.3 **U8BIT* STB_PVRGetBookmarkName (void * *bookmark.handle*)**

Allocates and returns the name associated with a bookmark.

Parameters

<i>bookmark_ - handle</i>	Bookmark handle
-------------------------------	-----------------

Returns

Pointer to the name string. This pointer must freed with STB_FreeMemory

4.44.2.4 U16BIT STB_PVRGetBookmarks (U8BIT *path*, void * *bookmarks*)**

Allocates and returns an array containing the handles of the bookmarks for the recording currently being played.

Parameters

<i>path</i>	Playback decode path
<i>bookmarks</i>	Pointer to array of returned bookmark handles

Returns

The number of handles in the allocated array. If this is 0 then array will be NULL.

4.44.2.5 U16BIT STB_PVRGetBookmarksForRecording (U32BIT *handle*, void * *bookmarks*)**

Allocates and returns an array containing the handles of the bookmarks for the recording specified by the given handle.

Parameters

<i>handle</i>	Recording handle
<i>bookmarks</i>	Pointer to array of returned bookmark handles

Returns

The number of handles in the allocated array. If this is 0 then array will be NULL.

4.44.2.6 U32BIT STB_PVRGetBookmarkTime (void * *bookmark_handle*)

Returns the time associated with a bookmark.

Parameters

<i>bookmark_ - handle</i>	Bookmark handle
-------------------------------	-----------------

Returns

Bookmark time

4.44.2.7 U8BIT* STB_PVRRecordingGetAdditionalInfo (U32BIT *handle*)

Gets the additional info string pointer for a recording.

Parameters

<i>handle</i>	- recording handle
---------------	--------------------

Returns

Pointer to the additional info

4.44.2.8 S32BIT STB_PVRRecordingGetEndPadding (U32BIT *handle*)

Gets the end padding value for the specified recording as set by STB_PVRRecording-SetEndPadding.

Parameters

<i>handle</i>	recording handle
---------------	------------------

Returns

End padding value

4.44.2.9 U32BIT STB_PVRRecordingGetParentalRating (U32BIT *handle*)

Returns the parental rating age for the specified recording as set by STB_PVR-RecordingSetParentalRatingAge.

Parameters

<i>handle</i>	recording handle
---------------	------------------

Returns

Parental rating age

4.44.2.10 S32BIT STB_PVRRecordingGetStartPadding (U32BIT *handle*)

Gets the start padding value for the specified recording as set by STB_PVRRecording-SetStartPadding.

Parameters

<i>handle</i>	recording handle
---------------	------------------

Returns

Start padding value

4.44.2.11 void **STB_PVRRecordingSetAdditionalInfo** (U32BIT *handle*, U8BIT *
additional_info)

Sets the additional info string for a recording.

Parameters

<i>handle</i>	recording handle
<i>additional_</i> <i>info</i>	additional info to be set

4.44.2.12 void **STB_PVRRecordingSetEndPadding** (U32BIT *handle*, S32BIT
end_padding)

Sets the end padding value for the specified recording.

Parameters

<i>handle</i>	recording handle
<i>start_</i> <i>padding</i>	end padding value

4.44.2.13 void **STB_PVRRecordingSetParentalRatingAge** (U32BIT *handle*, U32BIT
parental_rating)

Sets the parental rating age for the specified recording.

Parameters

<i>handle</i>	recording handle
<i>parental_</i> <i>rating</i>	parental rating age

4.44.2.14 void **STB_PVRRecordingSetStartPadding** (U32BIT *handle*, S32BIT
start_padding)

Sets the start padding value for the specified recording.

Parameters

<i>handle</i>	recording handle
<i>start_ - padding</i>	start padding value

4.44.2.15 void STB_PVRReleaseBookmarks (void ** *bookmarks*, U16BIT *num*)

Frees a previously allocated array of bookmark handles.

Parameters

<i>bookmarks</i>	Array of bookmark handles to be freed
<i>num</i>	Number of handles in the array.

4.44.2.16 void STB_PVRSavePlayPosition (U8BIT *path*)

Saves the bookmark holding the playback position. This bookmark is used to resume a playback when STB_PVRStartPlaying is called with resume = TRUE. This function should be called while decoders are still running, to be sure the position in the playback is accurate.

Parameters

<i>U8BIT</i>	path Playback path
--------------	--------------------

4.45 middleware/stb/inc/stbpvrmsg.h File Reference

PVR messages database access functions header file.

Functions

- BOOLEAN **STB_PVRInitialiseMessages** (void)
- BOOLEAN **STB_PVRAddMessage** (U8BIT *message)
- BOOLEAN **STB_PVRDeleteMessage** (U16BIT handle)
- BOOLEAN **STB_PVRCheckMessages** (void)
- U16BIT **STB_PVRGetNumMessages** (void)
- U16BIT **STB_PVRGetMessages** (U16BIT **handle_array)
- BOOLEAN **STB_PVRGetMessageInfo** (U16BIT handle, U16BIT *date, U8BIT *hour, U8BIT *min)
- U8BIT * **STB_PVRGetMessageText** (U16BIT handle)

4.45.1 Detailed Description

PVR messages database access functions header file.

Date

Nov 2007

Author

Chris Aldworth

4.46 middleware/stb/inc/stbsift.h File Reference

Header file - macros and function prototypes for public use.

This graph shows which files directly or indirectly include this file:

Data Structures

- struct [si_section_record](#)
- struct [si_table_record](#)

Defines

- #define **SI_BUFFER_1K** 1024
- #define **SI_BUFFER_2K** 2048
- #define **SI_BUFFER_4K** 4096
- #define **SI_BUFFER_8K** 8192
- #define **SI_BUFFER_16K** 16384
- #define **SI_BUFFER_32K** 32768
- #define **SI_BUFFER_64K** 65535
- #define **SI_XTID_MATCH_DONT_CARE** 0x0000
- #define **SI_XTID_MASK_DONT_CARE** 0x0000

Typedefs

- typedef struct [si_section_record](#) **SI_SECTION_RECORD**
- typedef struct [si_table_record](#) **SI_TABLE_RECORD**
- typedef void(* **F_AppSiEventHandler**)(U8BIT, E_APP_SI_EVENT_TYPE)

Enumerations

- enum **E_APP_SI_EVENT_TYPE** { **STOP_SI**, **START_SI_SEARCHING**, **START_SI_UPDATING_NEW_TRANSPORT**, **START_SI_UPDATING_SAME_TRANSPORT**, **SI_TIMEOUT** }
- enum **E_SI_REQUEST_TYPE** { **ONE_SHOT_REQUEST**, **CONTINUOUS_REQUEST** }
- enum **E_SI_TABLE_FORMAT_TYPE** { **SINGLE_SECT**, **MULTI_SECT** }

Functions

- void **STB_SIReportCurrentPmt** (U16BIT service_id, [SI_TABLE_RECORD](#) *table_rec, BOOLEAN new_serv, BOOLEAN new_pmt_version)
- F_AppSiEventHandler [STB_SIRegisterAppSiEventHandler](#) (F_AppSiEventHandler func_ptr)
Registers a function to be called to handle SI events. The currently registered event handler is returned, which allows a module to override the existing function and reset it.
- void * **STB_SIRequestTable** (U8BIT path, U16BIT pid, U8BIT tid_match, U8BIT tid_mask, E_SI_TABLE_FORMAT_TYPE format, U16BIT expected_tables, U16BIT xtid_match, U16BIT xtid_mask, E_SI_REQUEST_TYPE req_type, U16BIT buff_size, BOOLEAN crc, BOOLEAN low_priority, void(*callback)(void *, U32BIT, [SI_TABLE_RECORD](#) *), U32BIT ret_param)
- void **STB_SIModifyTableRequest** (void *filter_handle, U8BIT tid_match, U8BIT tid_mask, U16BIT xtid_match, U16BIT xtid_mask, U16BIT expected_tables)
- void **STB_SIRestartTableRequest** (void *filter_handle)
- void **STB_SICancelTableRequest** (void *filter_ptr)
- void **STB_SIReleaseTableRecord** ([SI_TABLE_RECORD](#) *table_rec)
- void **STB_SISearchComplete** (U8BIT path, BOOLEAN success, void *event_data, U32BIT data_size)
- BOOLEAN [STB_SIReportCat](#) ([SI_TABLE_RECORD](#) *table_rec)
Reports the CAT has been received so it can be passed on to other modules.
- BOOLEAN [STB_SIReportBat](#) ([SI_TABLE_RECORD](#) *table_rec)
Reports the BAT has been received so it can be passed on to other modules.
- BOOLEAN [STB_SIReportNit](#) ([SI_TABLE_RECORD](#) *table_rec)
Reports the NIT has been received so it can be passed on to other modules.

4.46.1 Detailed Description

Header file - macros and function prototypes for public use.

Date

06/03/2003

4.46.2 Function Documentation

4.46.2.1 F_AppSiEventHandler [STB_SIRegisterAppSiEventHandler](#) (F_AppSiEventHandler func_ptr)

Registers a function to be called to handle SI events. The currently registered event handler is returned, which allows a module to override the existing function and reset it.

Parameters

<i>func_ptr</i>	function being registered to handle SI events
-----------------	---

Returns

previously registered function

4.46.2.2 **BOOLEAN STB_SIReportBat (SI_TABLE_RECORD * *table_rec*)**

Reports the BAT has been received so it can be passed on to other modules.

Parameters

<i>table_rec</i>	- table record containing the BAT section data
------------------	--

Returns

TRUE if the BAT is passed to the CA module, FALSE otherwise

4.46.2.3 **BOOLEAN STB_SIReportCat (SI_TABLE_RECORD * *table_rec*)**

Reports the CAT has been received so it can be passed on to other modules.

Parameters

<i>table_rec</i>	- table record containing the CAT section data
------------------	--

Returns

TRUE if the CAT is passed to the CA module, FALSE otherwise

4.46.2.4 **BOOLEAN STB_SIReportNit (SI_TABLE_RECORD * *table_rec*)**

Reports the NIT has been received so it can be passed on to other modules.

Parameters

<i>table_rec</i>	- table record containing the NIT section data
------------------	--

Returns

TRUE if the NIT is passed to the CA module, FALSE otherwise

4.47 middleware/stb/inc/stbsipmt.h File Reference

Header file - Function prototypes for operating system.

Typedefs

- typedef void(* **F_PmtObserver**)(U16BIT serviceId, U8BIT *pmtData)

Functions

- void **STB_SIRegisterPmtObserver** (F_PmtObserver func_ptr)
- void **STB_SIUregisterPmtObserver** (F_PmtObserver func_ptr)

4.47.1 Detailed Description

Header file - Function prototypes for operating system.

Date

04/02/2014

Author

Ocean Blue

4.48 middleware/stb/inc/stbsitab.h File Reference

Header file - macros and function prototypes for public use.

#include "stbsiflt.h" Include dependency graph for stbsitab.h: This graph shows which files directly or indirectly include this file:

Data Structures

- struct [si_string](#)
- struct [si_linkage_desc](#)
- struct [si_guidance_desc](#)
- struct [si_fta_content_desc](#)
- struct [si_target_region](#)
- struct [si_target_region_desc](#)
- struct [si_serv_avail_desc](#)
- struct [si_image_icon_desc](#)
- struct [si_pat_service_entry](#)
- struct [si_pat_table](#)
- struct [si_dvb_subt_desc](#)
- struct [si_teletext_desc](#)
- struct [si_ca_desc](#)
- struct [si_iso_lang_desc](#)
- struct [si_ac3_desc](#)
- struct [si_ad_desc](#)
- struct [si_aac_desc](#)
- struct [si_service_move_desc](#)
- struct [si_app_sig_desc](#)
- struct [si_freesat_tunnelled_data_desc](#)
- struct [si_pmt_stream_entry](#)

- struct [si_pmt_table](#)
- struct [si_multiling_net_name_desc](#)
- struct [si_serv_list_desc](#)
- struct [uk_dtt_lcn_desc](#)
- struct [SI_NORDIG_SERV_LCN](#)
- struct [SI_NORDIG_LCN_DESC](#)
- struct [si_terr_del_sys_desc](#)
- struct [si_t2_del_sys_cell](#)
- struct [si_t2_del_sys_desc](#)
- struct [SI_TERR_DEL_SYS](#)
- struct [si_cable_del_sys_desc](#)
- struct [si_sat_del_sys_desc](#)
- union [si_delivery_sys_desc](#)
- struct [si_serv_attribute_desc](#)
- struct [si_nit_region_name](#)
- struct [si_nit_target_region_name_desc](#)
- struct [si_ciplus_service](#)
- struct [si_nit_transport_entry](#)
- struct [si_nit_change_entry](#)
- struct [si_nit_change_notify_desc](#)
- struct [si_nit_message_entry](#)
- struct [si_freesat_linkage_desc](#)
- struct [si_freesat_prefix_desc](#)
- struct [si_nit_table](#)
- struct [si_multiling_serv_name_desc](#)
- struct [SI_MULTILING_SHORT_NAME_DESC](#)
- struct [si_preferred_name_desc](#)
- struct [si_sdt_service_entry](#)
- struct [si_sdt_table](#)
- struct [si_freesat_lcn](#)
- struct [si_bat_freesat_region_lcn_entry](#)
- struct [si_bat_transport_entry](#)
- struct [si_multilang_group_name_desc](#)
- struct [si_bat_freesat_group_name_entry](#)
- struct [si_bat_freesat_serv_group_entry](#)
- struct [si_bat_freesat_iactive_storage_desc](#)
- struct [si_bat_freesat_region](#)
- struct [si_freesat_info_location](#)
- struct [si_bat_table](#)
- struct [si_component_desc](#)
- struct [si_multiling_component_desc](#)
- struct [si_content_desc](#)
- struct [si_parental_rating_desc](#)
- struct [si_short_event_desc](#)
- struct [si_extended_event_desc](#)
- struct [si_crid_desc](#)

- struct [si_eit_event_entry](#)
- struct [si_eit_table](#)
- struct [si_lto_desc](#)
- struct [si_time_table](#)
- struct [si_rct_promo_text](#)
- struct [si_rct_link_info](#)
- struct [si_rct_subtable_data](#)
- struct [si_rct_subtable](#)
- struct [si_rct_table](#)

Defines

- #define **CRID_LOCATION_MASK** 0x03
- #define **CRID_TYPE_MASK** 0xfc
- #define **CRID_TYPE_SHIFT** 2
- #define **CRID_LOCATION_0** 0x00
- #define **CRID_LOCATION_1** 0x01
- #define **CRID_TYPE_PROGRAMME** 0x31
- #define **CRID_TYPE_SERIES** 0x32
- #define **CRID_TYPE_RECOMMENDATION** 0x33
- #define **DVB_INVALID_CAROUSEL_ID** 0xffffffff

Typedefs

- typedef struct [si_string](#) **SI_STRING_DESC**
- typedef struct [si_linkage_desc](#) **SI_LINKAGE_DESC_ENTRY**
- typedef struct [si_guidance_desc](#) **SI_GUIDANCE_DESC**
- typedef struct [si_fta_content_desc](#) **SI_FTA_CONTENT_DESC**
- typedef struct [si_target_region](#) **SI_TARGET_REGION**
- typedef struct [si_target_region_desc](#) **SI_TARGET_REGION_DESC**
- typedef struct [si_serv_avail_desc](#) **SI_SERV_AVAIL_DESC**
- typedef struct [si_image_icon_desc](#) **SI_IMAGE_ICON_DESC**
- typedef struct [si_pat_service_entry](#) **SI_PAT_SERVICE_ENTRY**
- typedef struct [si_pat_table](#) **SI_PAT_TABLE**
- typedef struct [si_dvb_subt_desc](#) **SI_SUBTITLE_DESC**
- typedef struct [si_teletext_desc](#) **SI_TELETEXT_DESC**
- typedef struct [si_ca_desc](#) **SI_CA_DESC**
- typedef struct [si_iso_lang_desc](#) **SI_ISO_LANG_DESC**
- typedef struct [si_ac3_desc](#) **SI_AC3_DESC**
- typedef struct [si_ad_desc](#) **SI_AD_DESC**
- typedef struct [si_aac_desc](#) **SI_AAC_DESC**
- typedef struct [si_service_move_desc](#) **SI_SERVICE_MOVE_DESC**
- typedef struct [si_app_sig_desc](#) **SI_APP_SIG_DESC**
- typedef struct [si_freesat_tunelled_data_desc](#) **SI_FREESAT_TUNNELLED_DATA_DESC**
- typedef struct [si_pmt_stream_entry](#) **SI_PMT_STREAM_ENTRY**

- typedef struct [si_pmt_table](#) **SI_PMT_TABLE**
- typedef struct [si_multiling_net_name_desc](#) **SI_MULTILING_NET_NAME_DESC**
- typedef struct [si_serv_list_desc](#) **SI_SERV_LIST_DESC**
- typedef struct [uk_dtt_lcn_desc](#) **SI_LCN_DESC**
- typedef struct [si_terr_del_sys_desc](#) **SI_TERR_DEL_SYS_DESC**
- typedef struct [si_t2_del_sys_cell](#) **SI_T2_DEL_SYS_CELL**
- typedef struct [si_t2_del_sys_desc](#) **SI_T2_DEL_SYS_DESC**
- typedef struct [si_cable_del_sys_desc](#) **SI_CABLE_DEL_SYS_DESC**
- typedef struct [si_sat_del_sys_desc](#) **SI_SAT_DEL_SYS_DESC**
- typedef enum [si_delivery_sys_desc_type](#) **SI_DELIVERY_SYS_DESC_TYPE**
- typedef union [si_delivery_sys_desc](#) **SI_DELIVERY_SYS_DESC**
- typedef struct [si_serv_attribute_desc](#) **SI_SERV_ATTRIBUTE_DESC**
- typedef struct [si_nit_region_name](#) **SI_NIT_REGION_NAME**
- typedef struct [si_nit_target_region_name_desc](#) **SI_NIT_TARGET_REGION_NAME_DESC**
- typedef struct [si_ciplus_service](#) **SI_CIPLUS_SERVICE**
- typedef struct [si_nit_transport_entry](#) **SI_NIT_TRANSPORT_ENTRY**
- typedef struct [si_nit_change_entry](#) **SI_NIT_CHANGE_ENTRY**
- typedef struct [si_nit_change_notify_desc](#) **SI_NIT_CHANGE_NOTIFY_DESC**
- typedef struct [si_nit_message_entry](#) **SI_NIT_MESSAGE_ENTRY**
- typedef struct [si_freesat_linkage_desc](#) **SI_FREESAT_LINKAGE_DESC**
- typedef struct [si_freesat_prefix_desc](#) **SI_FREESAT_PREFIX_DESC**
- typedef struct [si_nit_table](#) **SI_NIT_TABLE**
- typedef struct [si_multiling_serv_name_desc](#) **SI_MULTILING_SERV_NAME_DESC**
- typedef struct [si_preferred_name_desc](#) **SI_PREFERRED_NAME_DESC**
- typedef struct [si_sdt_service_entry](#) **SI_SDT_SERVICE_ENTRY**
- typedef struct [si_sdt_table](#) **SI_SDT_TABLE**
- typedef struct [si_freesat_lcn](#) **SI_FREESAT_LCN**
- typedef struct [si_bat_freesat_region_lcn_entry](#) **SI_BAT_FREESAT_REGION_LCN_ENTRY**
- typedef struct [si_bat_transport_entry](#) **SI_BAT_TRANSPORT_ENTRY**
- typedef struct [si_multilang_group_name_desc](#) **SI_MULTILANG_GROUP_NAME_DESC**
- typedef struct [si_bat_freesat_group_name_entry](#) **SI_BAT_FREESAT_GROUP_NAME_ENTRY**
- typedef struct [si_bat_freesat_serv_group_entry](#) **SI_BAT_FREESAT_SERV_GROUP_ENTRY**
- typedef struct [si_bat_freesat_iactive_storage_desc](#) **SI_BAT_FREESAT_IACTIVE_STORAGE_DESC**
- typedef struct [si_bat_freesat_region](#) **SI_BAT_FREESAT_REGION**
- typedef struct [si_freesat_info_location](#) **SI_BAT_FREESAT_INFO_LOCATION**
- typedef struct [si_bat_table](#) **SI_BAT_TABLE**
- typedef struct [si_component_desc](#) **SI_COMPONENT_DESC**
- typedef struct [si_multiling_component_desc](#) **SI_MULTILING_COMPONENT_DESC**
- typedef struct [si_content_desc](#) **SI_CONTENT_DESC**

- typedef struct [si_parental_rating_desc](#) SI_PARENTAL_RATING_DESC
- typedef struct [si_short_event_desc](#) SI_SHORT_EVENT_DESC
- typedef struct [si_extended_event_desc](#) SI_EXTENDED_EVENT_DESC
- typedef struct [si_crid_desc](#) SI_CRID_DESC
- typedef struct [si_eit_event_entry](#) SI_EIT_EVENT_ENTRY
- typedef struct [si_eit_table](#) SI_EIT_TABLE
- typedef struct [si_lto_desc](#) SI_LTO_DESC
- typedef struct [si_time_table](#) SI_TIME_TABLE
- typedef struct [si_rct_promo_text](#) SI_RCT_PROMO_TEXT
- typedef struct [si_rct_link_info](#) SI_RCT_LINK_INFO
- typedef struct [si_rct_subtable_data](#) SI_RCT_SUBTABLE_DATA
- typedef struct [si_rct_subtable](#) SI_RCT_SUBTABLE
- typedef struct [si_rct_table](#) SI_RCT_TABLE

Enumerations

- enum **STB_SI_USER_DEF_DESCRIP_FUNCTION** { USER_DEF_DESCRIP_NOT_USED = 0, USER_DEF_DESCRIP_LOGICAL_CHAN_NUM = 1, USER_DEF_DESCRIP_PREF_NAME_LIST = 2, USER_DEF_DESCRIP_PREF_NAME_ID = 3 }
- enum **E_SI_EIT_TABLE_REQ** { EIT_NOW_NEXT_ACT = 0, EIT_NOW_NEXT_OTHER = 1, EIT_NOW_NEXT_ALL = 2, EIT_PF_PLUS = 3 }
- enum **E_SI_SCHED_TABLE_REQ** { EIT_SCHED_ACT = 0, EIT_SCHED_OTHER = 1, EIT_SCHED_ACT_4DAY = 2, EIT_SCHED_OTHER_4DAY = 3, EIT_SCHED_ALL_4DAY = 4, EIT_SCHED_ALL_8DAY = 5, EIT_SCHED_ALL = 6 }
- enum **E_SI_COMPONENT_TYPE_SUBTITLE** { CTYPE_DVB_SUBTITLE = 0x310, CTYPE_DVB_SUBTITLE_4_3 = 0x311, CTYPE_DVB_SUBTITLE_16_9 = 0x312, CTYPE_DVB_SUBTITLE_221_1 = 0x313, CTYPE_DVB_SUBTITLE_HD = 0x314, CTYPE_DVB_SUBTITLE_HH = 0x320, CTYPE_DVB_SUBTITLE_HH_4_3 = 0x321, CTYPE_DVB_SUBTITLE_HH_16_9 = 0x322, CTYPE_DVB_SUBTITLE_HH_221_1 = 0x323, CTYPE_DVB_SUBTITLE_HH_HD = 0x324 }
- enum **E_SI_COMPONENT_TYPE_AUDIO_DESC** { CTYPE_MPEG1_AD_VISUAL = 0x240, CTYPE_MPEG1_AD_HEARING = 0x241, CTYPE_MPEG1_AD_RCVR_MIX = 0x247, CTYPE_MPEG1_AD_BCAST_MIX = 0x248, CTYPE_AC3_FS_VISUAL = 0x450, CTYPE_AC3_FS_HEARING = 0x458, CTYPE_AC3_FS_DIALOGUE = 0x420, CTYPE_AC3_FS_COMMENTARY = 0x468, CTYPE_EAC3_FS_EMERGENCY = 0x470, CTYPE_EAC3_FS_VISUAL = 0x4d0, CTYPE_EAC3_FS_HEARING = 0x4d8, CTYPE_EAC3_FS_DIALOGUE = 0x4a0, CTYPE_EAC3_FS_COMMENTARY = 0x4e8, CTYPE_EAC3_FS_EMERGENCY = 0x4f0, CTYPE_HEAAC_AD_VISUAL = 0x640, CTYPE_HEAAC_AD_HEARING = 0x641, CTYPE_HEAAC_AD_RCVR_MIX = 0x647, CTYPE_HEAAC_AD_BCAST_MIX = 0x648, CTYPE_HEAACV2_AD_VISUAL = 0x644, CTYPE_HEAACV2_AD_HEARING = 0x645, CTYPE_HEAACV2_AD_RCVR_MIX = 0x649, CTYPE_HEAACV2_AD_BCAST_MIX = 0x64a, CTYPE_MPEG1_SA_RCVR_MIX = 0x242, CTYPE_HEAAC_SA_RCVR_MIX = 0x642, CTYPE_HEAACV2_SA_RCVR_MIX = 0x646 }

- enum **E_ICON_TRANSPORT_MODE** { **ICON_TRANS_LOCAL** = 0x0, **ICON_TRANS_URL** = 0x1, **ICON_TRANS_RES1** = 0x2, **ICON_TRANS_RES2** = 0x3 }
- enum **E_ICON_COORD_SYSTEM** { **ICON_COORDS_576** = 0x0, **ICON_COORDS_720** = 0x1, **ICON_COORDS_1080** = 0x2 }
- enum **SI_STREAM_TYPE** { **SI_STREAM_TYPE_VIDEO1** = 0x01, **SI_STREAM_TYPE_VIDEO2** = 0x02, **SI_STREAM_TYPE_AUDIO1** = 0x03, **SI_STREAM_TYPE_AUDIO2** = 0x04, **SI_STREAM_TYPE_PRIVATE** = 0x05, **SI_STREAM_TYPE_PES_PKT** = 0x06, **SI_STREAM_TYPE_MHEG** = 0x07, **SI_STREAM_TYPE_DATA_A** = 0x0a, **SI_STREAM_TYPE_DATA_B** = 0x0b, **SI_STREAM_TYPE_DATA_C** = 0x0c, **SI_STREAM_TYPE_DATA_D** = 0x0d, **SI_STREAM_TYPE_AUX** = 0x0e, **SI_STREAM_TYPE_AAC** = 0x0f, **SI_STREAM_TYPE_HEAAC** = 0x11, **SI_STREAM_TYPE_H264** = 0x1b, **SI_STREAM_TYPE_H265** = 0x24 }
- enum **si_delivery_sys_desc_type** { **SI_DEL_SYS_DESC_TYPE_TERR**, **SI_DEL_SYS_DESC_TYPE_SAT**, **SI_DEL_SYS_DESC_TYPE_CABLE** }
- enum **SI_NIT_RECEIVER_CATEGORY** { **CATEGORY_ALL_RECEIVERS** = 0, **CATEGORY_T2_RECEIVERS** = 1 }
- enum **SI_NIT_NETWORK_CHANGE_TYPE** { **NET_CHANGE_MESSAGE_ONLY** = 0x0, **NET_CHANGE_MINOR_DEFAULT** = 0x1, **NET_CHANGE_MULTIPLEX_REMOVE** = 0x2, **NET_CHANGE_SERVICE_CHANGE** = 0x3, **NET_CHANGE_MINOR_RESERVED_1** = 0x4, **NET_CHANGE_MINOR_RESERVED_2** = 0x5, **NET_CHANGE_MINOR_RESERVED_3** = 0x6, **NET_CHANGE_MINOR_RESERVED_4** = 0x7, **NET_CHANGE_MAJOR_DEFAULT** = 0x8, **NET_CHANGE_MULTIPLEX_FREQ_CHANGE** = 0x9, **NET_CHANGE_MULTIPLEX_COVERAGE_CHANGE** = 0xA, **NET_CHANGE_MULTIPLEX_ADDED** = 0xB, **NET_CHANGE_MAJOR_RESERVED_1** = 0xC, **NET_CHANGE_MAJOR_RESERVED_2** = 0xD, **NET_CHANGE_MAJOR_RESERVED_3** = 0xE, **NET_CHANGE_MAJOR_RESERVED_4** = 0xF }
- enum **E_RUNNING_STATE** { **RUN_STATE_UNDEFINED** = 0, **RUN_STATE_NOT_RUNNING**, **RUN_STATE_STARTS_SOON**, **RUN_STATE_PAUSING**, **RUN_STATE_RUNNING**, **RUN_STATE_OFF_AIR** }
- enum **E_RCT_LINK_TYPE** { **RCT_LINK_TYPE_URI** = 0x0, **RCT_LINK_TYPE_BINARY** = 0x1, **RCT_LINK_TYPE_URI_BINARY** = 0x2, **RCT_LINK_TYPE_DESCRIPTOR** = 0x3 }
- enum **E_RCT_HOW_RELATED** { **RCT_HOW_RELATED_TVA_2004** = 0x00, **RCT_HOW_RELATED_TVA_2005** = 0x01, **RCT_HOW_RELATED_TVA_2007** = 0x02 }
- enum **E_RCT_TERM_ID** { **RCT_TERMID_IS_TRAILER_OF** = 0x0002, **RCT_TERMID_IS_GROUP_TRAILER_OF** = 0x0005 }

Functions

- void **STB_SISetCountryPrivateDataSpecifier** (U32BIT code)
- void **STB_SISetFreesatPrivateDataSpecifierMode** (BOOLEAN mode)
- void **STB_SISetCiplusPrivateDataSpecifierMode** (BOOLEAN mode)
- void **STB_SISetEacemPrivateDataSpecifierMode** (BOOLEAN mode)
- void **STB_SISetNZSatPrivateDataSpecifierMode** (BOOLEAN mode)

- void **STB_SISetUserDefinedDescriptorFunction** (U8BIT dtag, STB_SI_USER_DEF_DESCRIP_FUNCTION func)
- void **STB_SIClearUserDefinedDescriptorFunctions** (void)
- void * **STB_SIRequestPat** (U8BIT path, E_SI_REQUEST_TYPE req_type, void(*callback)(void *, U32BIT, [SI_TABLE_RECORD](#) *), U32BIT ret_param)
- void * **STB_SIRequestPmt** (U8BIT path, E_SI_REQUEST_TYPE req_type, - U16BIT pmt_pid, U16BIT sid_match, U16BIT sid_mask, U16BIT table_count, void(*callback)(void *, U32BIT, [SI_TABLE_RECORD](#) *), U32BIT ret_param)
- void **STB_SIModifyPmtRequest** (void *fhandle, U16BIT sid_match, U16BIT sid_mask, U16BIT table_count)
- void * **STB_SIRequestNit** (U8BIT path, E_SI_REQUEST_TYPE req_type, void(*callback)(void *, U32BIT, [SI_TABLE_RECORD](#) *), U32BIT ret_param)
- void * **STB_SIRequestNitFromPid** (U8BIT path, U16BIT pid, BOOLEAN actual, E_SI_REQUEST_TYPE req_type, void(*callback)(void *, U32BIT, [SI_TABLE_RECORD](#) *), U32BIT ret_param)

Create an SI filter for an NIT table.

- void * **STB_SIRequestSdt** (U8BIT path, E_SI_REQUEST_TYPE req_type, BOOLEAN inc_sdt_actual, BOOLEAN inc_sdt_other, U16BIT tran_id_match, U16BIT tran_id_mask, U16BIT table_count, void(*callback)(void *, U32BIT, [SI_TABLE_RECORD](#) *), U32BIT ret_param)
- void * **STB_SIRequestSdtFromPid** (U8BIT path, U16BIT pid, E_SI_REQUEST_TYPE req_type, BOOLEAN inc_sdt_actual, BOOLEAN inc_sdt_other, U16BIT tran_id_match, U16BIT tran_id_mask, U16BIT table_count, void(*callback)(void *, U32BIT, [SI_TABLE_RECORD](#) *), U32BIT ret_param)
- void **STB_SIModifySdtRequest** (void *fhandle, BOOLEAN inc_sdt_actual, BOOLEAN inc_sdt_other, U16BIT tran_id_match, U16BIT tran_id_mask, U16BIT table_count)
- void * **STB_SIRequestBat** (U8BIT path, E_SI_REQUEST_TYPE req_type, - U16BIT bouquet_id_match, U16BIT bouquet_id_mask, U16BIT table_count, void(*callback)(void *, U32BIT, [SI_TABLE_RECORD](#) *), U32BIT ret_param)
- void * **STB_SIRequestBatFromPid** (U8BIT path, U16BIT pid, E_SI_REQUEST_TYPE req_type, U16BIT tran_id_match, U16BIT tran_id_mask, U16BIT table_count, void(*callback)(void *, U32BIT, [SI_TABLE_RECORD](#) *), U32BIT ret_param)
- void * **STB_SIRequestEit** (U8BIT path, E_SI_REQUEST_TYPE req_type, E_SI_EIT_TABLE_REQ reqd_eit_tables, U16BIT sid_match, U16BIT sid_mask, U16BIT table_count, void(*callback)(void *, U32BIT, [SI_TABLE_RECORD](#) *), U32BIT ret_param)
- void * **STB_SIRequestEitFromPid** (U8BIT path, U16BIT pid, E_SI_REQUEST_TYPE req_type, E_SI_EIT_TABLE_REQ reqd_eit_tables, U16BIT sid_match, U16BIT sid_mask, U16BIT table_count, void(*callback)(void *, U32BIT, [SI_TABLE_RECORD](#) *), U32BIT ret_param)
- void **STB_SIModifyEitRequest** (void *fhandle, E_SI_EIT_TABLE_REQ reqd_eit_tables, U16BIT sid_match, U16BIT sid_mask, U16BIT table_count)
- void * **STB_SIRequestSched** (U8BIT path, E_SI_REQUEST_TYPE req_type, E_SI_SCHED_TABLE_REQ reqd_eit_tables, U16BIT sid_match, U16BIT sid_mask, U16BIT table_count, void(*callback)(void *, U32BIT, [SI_TABLE_RECORD](#) *), U32BIT ret_param)

- void * **STB_SIRequestSchedFromPid** (U8BIT path, U16BIT pid, E_SI_REQUEST_TYPE req_type, E_SI_SCHEDULE_TABLE_REQ reqd_eit_tables, U16BIT sid_match, U16BIT sid_mask, U16BIT table_count, void(*callback)(void *, U32BIT, SI_TABLE_RECORD *), U32BIT ret_param)
- void * **STB_SIRequestTdt** (U8BIT path, E_SI_REQUEST_TYPE req_type, void(*callback)(void *, U32BIT, SI_TABLE_RECORD *), U32BIT ret_param)
- void * **STB_SIRequestTdtFromPid** (U8BIT path, U16BIT pid, E_SI_REQUEST_TYPE req_type, void(*callback)(void *, U32BIT, SI_TABLE_RECORD *), U32BIT ret_param)
- void * **STB_SIRequestTot** (U8BIT path, E_SI_REQUEST_TYPE req_type, void(*callback)(void *, U32BIT, SI_TABLE_RECORD *), U32BIT ret_param)
- void * **STB_SIRequestTotFromPid** (U8BIT path, U16BIT pid, E_SI_REQUEST_TYPE req_type, void(*callback)(void *, U32BIT, SI_TABLE_RECORD *), U32BIT ret_param)
- void * **STB_SIRequestCat** (U8BIT path, E_SI_REQUEST_TYPE req_type, void(*callback)(void *, U32BIT, SI_TABLE_RECORD *), U32BIT ret_param)
- void * **STB_SIRequestRct** (U8BIT path, E_SI_REQUEST_TYPE req_type, U16BIT rct_pid, void(*callback)(void *, U32BIT, SI_TABLE_RECORD *), U32BIT ret_param)
- SI_PAT_TABLE * **STB_SIParsePatTable** (SI_TABLE_RECORD *table_rec)
- SI_PMT_TABLE * **STB_SIParsePmtTable** (SI_TABLE_RECORD *table_rec)
- SI_NIT_TABLE * **STB_SIParseNitTable** (SI_TABLE_RECORD *table_rec)
- SI_SDT_TABLE * **STB_SIParseSdtTable** (SI_TABLE_RECORD *table_rec)
- SI_EIT_TABLE * **STB_SIParseEitTable** (SI_TABLE_RECORD *table_rec)
- SI_TIME_TABLE * **STB_SIParseTimeTable** (SI_TABLE_RECORD *table_rec)
- SI_RCT_TABLE * **STB_SIParseRctTable** (SI_TABLE_RECORD *table_rec)
- SI_BAT_TABLE * **STB_SIParseBatTable** (SI_TABLE_RECORD *table_rec)
- U8BIT * **STB_SIReadString** (U8BIT nbytes, U8BIT *dptr, SI_STRING_DESC **str_ptr)
- BOOLEAN **STB_SIParseDelSysDesc** (U8BIT *data, SI_DELIVERY_SYS_DESC_TYPE *type, SI_DELIVERY_SYS_DESC **desc)
- BOOLEAN **STB_SIParseServiceDescriptor** (U8BIT *data, U8BIT *type, SI_STRING_DESC **provider, SI_STRING_DESC **name)
- BOOLEAN **STB_SIParseShortEventDescriptor** (U8BIT *data, SI_SHORT_EVENT_DESC **event_desc)
- U16BIT **STB_SIGetPmtCaldDescArray** (U8BIT *pmt_data, U16BIT **pmt_cald_ids)

Parses the given PMT to produce an array of the CA system IDs required by the service or streams on the service. The array of IDs will be allocated by this function and should be freed using STB_SIReleaseCaldDescArray.
- void * **STB_SIRequestAit** (U8BIT path, E_SI_REQUEST_TYPE req_type, U16BIT ait_pid, void(*callback)(void *, U32BIT, SI_TABLE_RECORD *), U32BIT ret_param)

Generates request for AIT on given PID.
- void **STB_SIReleasePatTable** (SI_PAT_TABLE *pat_table)
- void **STB_SIReleasePmtTable** (SI_PMT_TABLE *pmt_table)
- void **STB_SIReleaseNitTable** (SI_NIT_TABLE *nit_table)
- void **STB_SIReleaseSdtTable** (SI_SDT_TABLE *sdt_table)

- void **STB_SIReleaseEitTable** ([SI_EIT_TABLE](#) *eit_table)
- void **STB_SIReleaseTimeTable** ([SI_TIME_TABLE](#) *time_table)
- void **STB_SIReleaseRctTable** ([SI_RCT_TABLE](#) *rct_table)
- void **STB_SIReleaseBatTable** ([SI_BAT_TABLE](#) *bat_table)
- void **STB_SIReleasePatStreamEntry** ([SI_PAT_SERVICE_ENTRY](#) *entry_ptr)
- void **STB_SIReleasePmtStreamEntry** ([SI_PMT_STREAM_ENTRY](#) *entry_ptr)
- void **STB_SIReleaseNitTransportEntry** ([SI_NIT_TRANSPORT_ENTRY](#) *entry_ptr)
- void **STB_SIReleaseSdtServiceEntry** ([SI_SDT_SERVICE_ENTRY](#) *entry_ptr)
- void **STB_SIReleaseEitEventEntry** ([SI_EIT_EVENT_ENTRY](#) *entry_ptr)
- void **STB_SIReleaseRctSubtable** ([SI_RCT_SUBTABLE](#) *sub_ptr)
- void **STB_SIReleaseRctSubtableData** ([SI_RCT_SUBTABLE_DATA](#) *data_ptr)
- void **STB_SIReleaseBatTransportEntry** ([SI_BAT_TRANSPORT_ENTRY](#) *entry_ptr)
- void **STB_SIReleaseBatLcnEntry** ([SI_BAT_FREESAT_REGION_LCN_ENTRY](#) *entry_ptr)
- void **STB_SIReleaseDelSysDesc** ([SI_DELIVERY_SYS_DESC](#) *desc, SI_DELIVERY_SYS_DESC_TYPE type)
- void **STB_SIReleaseStringDesc** ([SI_STRING_DESC](#) *desc)
- void **STB_SIReleaseCaDescArray** ([SI_CA_DESC](#) *desc_array, U16BIT num_entries)
- void **STB_SIReleaseCaldDescArray** (U16BIT *desc_array, U8BIT num_entries)
- void **STB_SIReleaseComponentDescArray** ([SI_COMPONENT_DESC](#) *desc_array, U8BIT num_entries)
- void **STB_SIReleaseContentDescArray** ([SI_CONTENT_DESC](#) *desc_array, U8BIT num_entries)
- void **STB_SIReleaseFreqListDescArray** (U32BIT *desc_array, U16BIT num_entries)
- void **STB_SIReleaseGuidanceDesc** ([SI_GUIDANCE_DESC](#) *desc_ptr)
- void **STB_SIReleaseImagelconDescArray** ([SI_IMAGE_ICON_DESC](#) *icon_array, U8BIT num_icons)
- void **STB_SIReleaseIsoLangDescArray** ([SI_ISO_LANG_DESC](#) *desc_array, U16BIT num_entries)
- void **STB_SIReleaseLinkageDescList** ([SI_LINKAGE_DESC_ENTRY](#) *desc_list, U16BIT num_entries)
- void **STB_SIReleaseLtoDescArray** ([SI_LTO_DESC](#) *desc_array, U16BIT num_entries)
- void **STB_SIReleaseMultilingComponentDescArray** ([SI_MULTILING_COMPONENT_DESC](#) *desc_array, U8BIT num_entries)
- void **STB_SIReleaseMultilingNetNameDescArray** ([SI_MULTILING_NET_NAME_DESC](#) *desc_array, U16BIT num_entries)
- void **STB_SIReleaseMultilingServNameDescArray** ([SI_MULTILING_SERVICE_NAME_DESC](#) *desc_array, U16BIT num_entries)
- void **STB_SIReleaseMultilingShortNameArray** ([SI_MULTILING_SHORT_NAME_DESC](#) *desc_array, U16BIT num_entries)
- void **STB_SIReleaseParentalRatingDescArray** ([SI_PARENTAL_RATING_DESC](#) *desc_array, U8BIT num_entries)

- void **STB_SIReleaseServListDescArray** ([SI_SERV_LIST_DESC](#) *desc_array, - U16BIT num_entries)
- void **STB_SIReleaseShortEventDescArray** ([SI_SHORT_EVENT_DESC](#) *desc_array, U8BIT num_entries)
- void **STB_SIReleaseExtendedEventDescArray** ([SI_EXTENDED_EVENT_DESC](#) *desc_array, U8BIT num_entries)
- void **STB_SIReleaseSubtitleDescArray** ([SI_SUBTITLE_DESC](#) *desc_array, - U16BIT num_entries)
- void **STB_SIReleaseTeletextDescArray** ([SI_TELETEXT_DESC](#) *desc_array, - U16BIT num_entries)
- void **STB_SIReleaseLcnDescArray** ([SI_LCN_DESC](#) *desc_array, U16BIT num_entries)
- void **STB_SIReleaseNordigLcn2DescArray** ([SI_NORDIG_LCN_DESC](#) *desc_array, U16BIT num_entries)
- void **STB_SIReleasePrefNameDescArray** ([SI_PREFERRED_NAME_DESC](#) *desc_array, U16BIT num_entries)
- void **STB_SIReleaseCRIDList** ([SI_CRID_DESC](#) *crid_list)
- void **STB_SIReleaseTargetRegionNameList** ([SI_NIT_TARGET_REGION_NAME_DESC](#) *desc_list)
- void **STB_SIReleaseTargetRegionList** ([SI_TARGET_REGION_DESC](#) *desc_list)
- void **STB_SIReleaseRctLinkInfo** ([SI_RCT_LINK_INFO](#) *link_info)
- void **STB_SIReleaseFreesatLinkageDesc** ([SI_FREESAT_LINKAGE_DESC](#) *desc)
- void **STB_SIReleaseFreesatPrefixList** ([SI_FREESAT_PREFIX_DESC](#) *list)
- void **STB_SIReleaseAvailabilityDescriptorList** ([SI_SERV_AVAIL_DESC](#) *desc_list)
- void **STB_SIReleaseCIPlusServiceList** ([SI_CIPLUS_SERVICE](#) *service_list)

4.48.1 Detailed Description

Header file - macros and function prototypes for public use.

Date

06/03/2003

4.48.2 Function Documentation

4.48.2.1 [U16BIT STB_SIGetPmtCaldDescArray \(U8BIT * pmt_data, U16BIT ** pmt_ca_ids \)](#)

Parses the given PMT to produce an array of the CA system IDs required by the service or streams on the service. The array of IDs will be allocated by this function and should be freed using [STB_SIReleaseCaldDescArray](#).

Parameters

<i>pmt_data</i>	- raw PMT section data
<i>pmt_ca_ids</i>	- pointer to an array that will be allocated containing the CA IDs found in the PMT

Returns

Number of CA IDs found in the PMT and returned in the array, 0 if none are found.

4.48.2.2 void* **STB_SIRequestAit** (U8BIT *path*, E_SI_REQUEST_TYPE *req_type*, U16BIT *ait_pid*, void(*) (void *, U32BIT, SI_TABLE_RECORD *) *callback*, U32BIT *ret_param*)

Generates request for AIT on given PID.

Parameters

<i>path</i>	The demux path to use
<i>req_type</i>	ONE_SHOT or CONTINUOUS request
<i>ait_pid</i>	Pid for the AIT
<i>callback</i>	Callback function to receive the parsed pmt table
<i>ret_param</i>	Parameter to be returned to callback function

Returns

Filter handle, NULL if filter not successfully setup

4.48.2.3 void* **STB_SIRequestNitFromPid** (U8BIT *path*, U16BIT *pid*, BOOLEAN *actual*, E_SI_REQUEST_TYPE *req_type*, void(*) (void *, U32BIT, SI_TABLE_RECORD *) *callback*, U32BIT *ret_param*)

Create an SI filter for an NIT table.

Parameters

<i>path</i>	demux path to be used
<i>pid</i>	PID the table will be available on - this allows the standard PID to be overridden
<i>actual</i>	TRUE if NITactual is to be retrieved, FALSE for NITother
<i>req_type</i>	ONE_SHOT_REQUEST or CONTINUOUS_REQUEST
<i>callback</i>	function to be called when table is received
<i>ret_param</i>	value that will be passed to the callback function

Returns

filter handle, NULL if filter isn't created

4.49 middleware/stb/inc/stbuni.h File Reference

Header for STB unicode string handling routines.

Defines

- #define **STB_UTF16_LEN_TO_BYTES_IN_STRING**(x) (((x) *2) + 3)
- #define **INVALID_UNICODE_CHAR** 0xFFFF

Functions

- U8BIT * **STB_SetUnicodeStringChar** (U8BIT *string_ptr, U16BIT char_id, U16-BIT code)
- U8BIT * **STB_DeleteUnicodeStringChar** (U8BIT *string_ptr, U16BIT char_id)
- U32BIT **STB_GetUnicodeStringChar** (U8BIT *string_ptr, U16BIT char_id)
- U8BIT * **STB_ConcatUnicodeStrings** (U8BIT *string1_ptr, U8BIT *string2_ptr)
- U8BIT * **STB_UnicodeStringTokenise** (U8BIT *str, U8BIT **save_ptr)
- U8BIT * **STB_UnicodeStrStr** (U8BIT *str1, U8BIT *str2, BOOLEAN ignore_case)
- S8BIT **STB_CompareUnicodeStrings** (U8BIT *string1_ptr, U8BIT *string2_ptr, BOOLEAN exact_match, BOOLEAN ignore_case)
- U8BIT * **STB_ConvertStringToUnicode** (U8BIT *string, BOOLEAN *reverse_dir, U16BIT *nchar, BOOLEAN strip_DVB_cntrl_char, U32BIT lang_code)
- U8BIT * **STB_ConvertStringToUTF8** (U8BIT *string, U16BIT *nchar, BOOLEAN strip_DVB_cntrl_char, U32BIT lang_code)
- void **STB_ReleaseUnicodeString** (U8BIT *string)
- U8BIT * **STB_ConvertUTF16toUTF8** (U8BIT *src, U32BIT *outlen)
- BOOLEAN **STB_IsUnicodeString** (U8BIT *string_ptr)
- BOOLEAN **STB_IsNormalString** (U8BIT *string_ptr)
- U32BIT **STB_GetNumBytesInString** (U8BIT *string_ptr)
- BOOLEAN **STB_IsUnicodeStringReversed** (U8BIT *string_ptr)
- BOOLEAN **STB_IsStringEmpty** (U8BIT *string_ptr)
- void **STB_SetDefaultAsciiTable** (U8BIT table)
- S8BIT **STB_CompareStringsIgnoreCase** (U8BIT *string1_ptr, U8BIT *string2_ptr)
- U8BIT * **STB_FormatUnicodeString** (BOOLEAN strip_DVB_cntrl_char, BOOLEAN *reverse_dir, const U8BIT *const format_ptr,...)
- U32BIT **STB_UnicodeStringLength** (U8BIT *string_ptr)
- U8BIT * **STB_UnicodeInsertString** (U8BIT *src_str, U16BIT insert_pos, U8BIT *insert_str, BOOLEAN replace_char)

4.49.1 Detailed Description

Header for STB unicode string handling routines.

Date

31/05/2001

4.50 middleware/stb/inc/stbvtc.h File Reference

Header file - macros and function prototypes for public use.

```
#include "osdtype.h" #include "vtctype.h" Include dependency
graph for stbvtc.h:
```

Typedefs

- typedef void(* **F_NOTIFY_VIDEO_AR**)(E_ASPECT_RATIO ar)

Functions

- void **STB_VTInitialise** (void)
Initialise the VTC module.
- void **STB_VTSetVideoOutput** (**S_RECTANGLE** *output)
DVBCore application output window for entire video.
- E_ASPECT_RATIO **STB_VTGetDisplayAspectRatio** (void)
Get display aspect ratio.
- void **STB_VTSetDisplayAspectRatio** (E_ASPECT_RATIO aspect_ratio)
Set display aspect ratio.
- void **STB_VTSetMhegEnable** (BOOLEAN enable)
Turn on or off MHEG5 scaling calculation.
- void **STB_VTSetMhegScalingResolution** (U16BIT width, U16BIT height)
Set MHEG5 scaling resolution for video.
- void **STB_VTSetMhegVideoScaling** (**S_RECTANGLE** *scaling)
Set video scaling by MHEG5.
- void **STB_VTSetMhegAspectRatio** (E_ASPECT_RATIO aspect_ratio)
Set scene aspect ratio (MHEG-5 specific)
- void **STB_VTSetMhegVideoAlignment** (E_VIDEO_ASPECT_MODE mode)
Set MHEG5 widescreen alignment.
- void **STB_VTSetHbbtvEnable** (BOOLEAN enable)
Turn on or off HBBTV scaling calculation.
- void **STB_VTSetHbbtvVideoWindow** (**S_RECTANGLE** *rect)
Set video scaling by HBBTV.
- void **STB_VTSetVideoAlignmentPref** (E_VIDEO_ASPECT_MODE pref)
Set video alignment preference.

- void [STB_VTSetVideoPrefChangedCallback](#) (void(*callback)(void *), void *user_data)
Set video preferences change callback.
- void [STB_VTSetVideoRatioCallback](#) (F_NOTIFY_VIDEO_AR ar)
Set video preferences change callback.
- E_FORMAT_CONVERSION [STB_VTGetDecoderFormatConversion](#) (void)
Return the current decoder format conversion.
- void [STB_VTGetVideoResolution](#) (U16BIT *width, U16BIT *height)
Return the current video resolution.
- E_ASPECT_RATIO [STB_VTGetVideoAspectRatio](#) (void)
Return the current video aspect ratio.

4.50.1 Detailed Description

Header file - macros and function prototypes for public use.

Date

27/02/2012

Author

Ocean Blue

4.50.2 Function Documentation

4.50.2.1 E_FORMAT_CONVERSION STB_VTGetDecoderFormatConversion (void)

Return the current decoder format conversion.

Returns

The current format conversion

4.50.2.2 E_ASPECT_RATIO STB_VTGetDisplayAspectRatio (void)

Get display aspect ratio.

Parameters

<i>aspect_ratio</i>	- display aspect ratio
---------------------	------------------------

Returns

None

4.50.2.3 E_ASPECT_RATIO STB_VTGetVideoAspectRatio (void)

Return the current video aspect ratio.

Returns

E_ASPECT_RATIO aspect ratio

4.50.2.4 void STB_VTGetVideoResolution (U16BIT * width, U16BIT * height)

Return the current video resolution.

Parameters

<i>width</i>	- video width
<i>height</i>	- video height

Returns

None

4.50.2.5 void STB_VTInitialise (void)

Initialise the VTC module.

Parameters

<i>None</i>	
-------------	--

Returns

None

4.50.2.6 void STB_VTSetDisplayAspectRatio (E_ASPECT_RATIO aspect_ratio)

Set display aspect ratio.

Parameters

<i>aspect_ratio</i>	- display aspect ratio
---------------------	------------------------

Returns

None

4.50.2.7 void STB_VTSetHbbtvEnable (BOOLEAN *enable*)

Turn on or off HBBTV scaling calculation.

Parameters

<i>enable</i>	TRUE - turn on HbbTV calculations
---------------	-----------------------------------

Returns

None

4.50.2.8 void STB_VTSetHbbtvVideoWindow (S_RECTANGLE * *rect*)

Set video scaling by HBBTV.

Parameters

<i>rect</i>	output window rectangle
-------------	-------------------------

Returns

None

4.50.2.9 void STB_VTSetMhegAspectRatio (E_ASPECT_RATIO *aspect_ratio*)

Set scene aspect ratio (MHEG-5 specific)

Parameters

<i>aspect_ratio</i>	- scene aspect ratio
---------------------	----------------------

Returns

None

4.50.2.10 void STB_VTSetMhegEnable (BOOLEAN *enable*)

Turn on or off MHEG5 scaling calculation.

Parameters

<i>enable</i>	TRUE - turn on MHEG5 calculations
---------------	-----------------------------------

Returns

None

4.50.2.11 void STB_VTSetMhegScalingResolution (U16BIT *width*, U16BIT *height*)

Set MHEG5 scaling resolution for video.

Parameters

<i>width</i>	- width resolution
<i>height</i>	- height resolution

Returns

None

4.50.2.12 void STB_VTSetMhegVideoAlignment (E_VIDEO_ASPECT_MODE *mode*)

Set MHEG5 widescreen alignment.

Parameters

<i>mode</i>	- widescreen alignment mode
-------------	-----------------------------

Returns

None

4.50.2.13 void STB_VTSetMhegVideoScaling (S_RECTANGLE * *scaling*)

Set video scaling by MHEG5.

Parameters

<i>scaling</i>	- scaling transformation (offset, size)
----------------	---

Returns

None

4.50.2.14 void STB_VTSetVideoAlignmentPref (E_VIDEO_ASPECT_MODE *pref*)

Set video alignment preference.

Parameters

<i>pref</i>	- video alignment
-------------	-------------------

Returns

None

4.50.2.15 void STB_VTSetVideoOutput (S_RECTANGLE * *output*)

DVBCore application output window for entire video.

Parameters

<i>output</i>	- output video rectangle
---------------	--------------------------

Returns

None

4.50.2.16 void STB_VTSetVideoPrefChangedCallback (void(*) (void *) *callback*, void * *user_data*)

Set video preferences change callback.

The callback function is called when video transformation is changed as a result of a user preference change (only).

Parameters

<i>callback</i>	- callback for notification
<i>user_data</i>	- user data for the callback

Returns

None

4.50.2.17 void STB_VTSetVideoRatioCallback (F_NOTIFY_VIDEO_AR *ar*)

Set video preferences change callback.

The callback function is called when video transformation is changed as a result of a user preference change (only).

Parameters

<i>callback</i>	- callback for notification
<i>user_data</i>	- user data for the callback

Returns

None

4.51 middleware/stb/inc/vtctype.h File Reference

Header file - Function prototypes for A/V control.

This graph shows which files directly or indirectly include this file:

Typedefs

- typedef enum E_VIDEO_ASPECT_MODE **E_STB_AV_ASPECT_MODE**

Enumerations

- enum **E_VIDEO_ASPECT_MODE** { **ASPECT_MODE_AUTO**, **ASPECT_MODE_4_3**, **ASPECT_MODE_16_9**, **ASPECT_MODE_14_9**, **ASPECT_MODE_ZOOM** }
- enum **E_FORMAT_CONVERSION** { **FORMAT_CONVERSION_UNKNOWN**, **FORMAT_CONVERSION_IGNORE**, **FORMAT_CONVERSION_PANSCAN**, **FORMAT_CONVERSION_LETTERBOX**, **FORMAT_CONVERSION_LETTERBOX_14_9**, **FORMAT_CONVERSION_PILLAR_BOX**, **FORMAT_CONVERSION_ZOOM_4_3**, **FORMAT_CONVERSION_ZOOM_14_9**, **FORMAT_CONVERSION_PANSCAN_14_9**, **FORMAT_CONVERSION_FULL_4_3**, **FORMAT_CONVERSION_CENTRE_14_9**, **FORMAT_CONVERSION_CENTRE_4_3** }

4.51.1 Detailed Description

Header file - Function prototypes for A/V control.

Date

06/02/2014

Author

Ocean Blue

4.52 middleware/stb/src/asciimap.h File Reference

Contains character map tables for converting single byte ascii codes above 0xa0 to unicode codes.

Defines

- #define **CHAR_TABLE_START_ASCII_CODE** 0xa0
- #define **MAX_CHAR_MAP_TABLES** 16
- #define **NULL_CHAR** 0x0000
- #define **UC_NO_BREAK_SPACE** 0x00a0
- #define **UC_INVERTED_EXCLAMATION_MARK** 0x00a1
- #define **UC_CENT_SIGN** 0x00a2
- #define **UC_POUND_SIGN** 0x00a3
- #define **UC_CURRENCY_SIGN** 0x00a4
- #define **UC_YEN_SIGN** 0x00a5
- #define **UC_BROKEN_BAR** 0x00a6
- #define **UC_SECTION_SIGN** 0x00a7
- #define **UC_DIAERESIS** 0x00a8
- #define **UC_COPYRIGHT_SIGN** 0x00a9
- #define **UC_FEMININE_ORDINAL_INDICATOR** 0x00aa
- #define **UC_LEFT_POINTING_DOUBLE_ANGLE_QUOTATION_MARK** 0x00ab
- #define **UC_NOT_SIGN** 0x00ac
- #define **UC_SOFT_HYPHEN** 0x00ad
- #define **UC_REGISTERED_SIGN** 0x00ae
- #define **UC_MACRON** 0x00af
- #define **UC_DEGREE_SIGN** 0x00b0
- #define **UC_PLUS_MINUS_SIGN** 0x00b1
- #define **UC_SUPERSCRIPT_TWO** 0x00b2
- #define **UC_SUPERSCRIPT_THREE** 0x00b3
- #define **UC_ACUTE_ACCENT** 0x00b4
- #define **UC_MICRO_SIGN** 0x00b5
- #define **UC_PILCROW_SIGN** 0x00b6
- #define **UC_MIDDLE_DOT** 0x00b7
- #define **UC_CEDILLA** 0x00b8
- #define **UC_SUPERSCRIPT_ONE** 0x00b9
- #define **UC_MASCULINE_ORDINAL_INDICATOR** 0x00ba
- #define **UC_RIGHT_POINTING_DOUBLE_ANGLE_QUOTATION_MARK** 0x00bb
- #define **UC_VULGAR_FRACTION_ONE_QUARTER** 0x00bc
- #define **UC_VULGAR_FRACTION_ONE_HALF** 0x00bd
- #define **UC_VULGAR_FRACTION_THREE_QUARTERS** 0x00be
- #define **UC_INVERTED_QUESTION_MARK** 0x00bf
- #define **UC_LATIN_CAPITAL_LETTER_A_WITH_GRAVE** 0x00c0
- #define **UC_LATIN_CAPITAL_LETTER_A_WITH_ACUTE** 0x00c1

- #define UC_LATIN_CAPITAL_LETTER_A_WITH_CIRCUMFLEX 0x00c2
- #define UC_LATIN_CAPITAL_LETTER_A_WITH_TILDE 0x00c3
- #define UC_LATIN_CAPITAL_LETTER_A_WITH_DIAERESIS 0x00c4
- #define UC_LATIN_CAPITAL_LETTER_A_WITH_RING_ABOVE 0x00c5
- #define UC_LATIN_CAPITAL_LIGATURE_AE 0x00c6
- #define UC_LATIN_CAPITAL_LETTER_C_WITH_CEDILLA 0x00c7
- #define UC_LATIN_CAPITAL_LETTER_E_WITH_GRAVE 0x00c8
- #define UC_LATIN_CAPITAL_LETTER_E_WITH_ACUTE 0x00c9
- #define UC_LATIN_CAPITAL_LETTER_E_WITH_CIRCUMFLEX 0x00ca
- #define UC_LATIN_CAPITAL_LETTER_E_WITH_DIAERESIS 0x00cb
- #define UC_LATIN_CAPITAL_LETTER_I_WITH_GRAVE 0x00cc
- #define UC_LATIN_CAPITAL_LETTER_I_WITH_ACUTE 0x00cd
- #define UC_LATIN_CAPITAL_LETTER_I_WITH_CIRCUMFLEX 0x00ce
- #define UC_LATIN_CAPITAL_LETTER_I_WITH_DIAERESIS 0x00cf
- #define UC_LATIN_CAPITAL_LETTER_ETH 0x00d0
- #define UC_LATIN_CAPITAL_LETTER_N_WITH_TILDE 0x00d1
- #define UC_LATIN_CAPITAL_LETTER_O_WITH_GRAVE 0x00d2
- #define UC_LATIN_CAPITAL_LETTER_O_WITH_ACUTE 0x00d3
- #define UC_LATIN_CAPITAL_LETTER_O_WITH_CIRCUMFLEX 0x00d4
- #define UC_LATIN_CAPITAL_LETTER_O_WITH_TILDE 0x00d5
- #define UC_LATIN_CAPITAL_LETTER_O_WITH_DIAERESIS 0x00d6
- #define UC_MULTIPLICATION_SIGN 0x00d7
- #define UC_LATIN_CAPITAL_LETTER_O_WITH_STROKE 0x00d8
- #define UC_LATIN_CAPITAL_LETTER_U_WITH_GRAVE 0x00d9
- #define UC_LATIN_CAPITAL_LETTER_U_WITH_ACUTE 0x00da
- #define UC_LATIN_CAPITAL_LETTER_U_WITH_CIRCUMFLEX 0x00db
- #define UC_LATIN_CAPITAL_LETTER_U_WITH_DIAERESIS 0x00dc
- #define UC_LATIN_CAPITAL_LETTER_Y_WITH_ACUTE 0x00dd
- #define UC_LATIN_CAPITAL_LETTER_THORN 0x00de
- #define UC_LATIN_SMALL_LETTER_SHARP_S 0x00df
- #define UC_LATIN_SMALL_LETTER_A_WITH_GRAVE 0x00e0
- #define UC_LATIN_SMALL_LETTER_A_WITH_ACUTE 0x00e1
- #define UC_LATIN_SMALL_LETTER_A_WITH_CIRCUMFLEX 0x00e2
- #define UC_LATIN_SMALL_LETTER_A_WITH_TILDE 0x00e3
- #define UC_LATIN_SMALL_LETTER_A_WITH_DIAERESIS 0x00e4
- #define UC_LATIN_SMALL_LETTER_A_WITH_RING_ABOVE 0x00e5
- #define UC_LATIN_SMALL_LIGATURE_AE 0x00e6
- #define UC_LATIN_SMALL_LETTER_C_WITH_CEDILLA 0x00e7
- #define UC_LATIN_SMALL_LETTER_E_WITH_GRAVE 0x00e8
- #define UC_LATIN_SMALL_LETTER_E_WITH_ACUTE 0x00e9
- #define UC_LATIN_SMALL_LETTER_E_WITH_CIRCUMFLEX 0x00ea
- #define UC_LATIN_SMALL_LETTER_E_WITH_DIAERESIS 0x00eb
- #define UC_LATIN_SMALL_LETTER_I_WITH_GRAVE 0x00ec
- #define UC_LATIN_SMALL_LETTER_I_WITH_ACUTE 0x00ed
- #define UC_LATIN_SMALL_LETTER_I_WITH_CIRCUMFLEX 0x00ee
- #define UC_LATIN_SMALL_LETTER_I_WITH_DIAERESIS 0x00ef

- `#define UC_LATIN_SMALL_LETTER_ETH 0x00f0`
- `#define UC_LATIN_SMALL_LETTER_N_WITH_TILDE 0x00f1`
- `#define UC_LATIN_SMALL_LETTER_O_WITH_GRAVE 0x00f2`
- `#define UC_LATIN_SMALL_LETTER_O_WITH_ACUTE 0x00f3`
- `#define UC_LATIN_SMALL_LETTER_O_WITH_CIRCUMFLEX 0x00f4`
- `#define UC_LATIN_SMALL_LETTER_O_WITH_TILDE 0x00f5`
- `#define UC_LATIN_SMALL_LETTER_O_WITH_DIAERESIS 0x00f6`
- `#define UC_DIVISION_SIGN 0x00f7`
- `#define UC_LATIN_SMALL_LETTER_O_WITH_STROKE 0x00f8`
- `#define UC_LATIN_SMALL_LETTER_U_WITH_GRAVE 0x00f9`
- `#define UC_LATIN_SMALL_LETTER_U_WITH_ACUTE 0x00fa`
- `#define UC_LATIN_SMALL_LETTER_U_WITH_CIRCUMFLEX 0x00fb`
- `#define UC_LATIN_SMALL_LETTER_U_WITH_DIAERESIS 0x00fc`
- `#define UC_LATIN_SMALL_LETTER_Y_WITH_ACUTE 0x00fd`
- `#define UC_LATIN_SMALL_LETTER_THORN 0x00fe`
- `#define UC_LATIN_SMALL_LETTER_Y_WITH_DIAERESIS 0x00ff`
- `#define UC_LATIN_CAPITAL_LETTER_A_WITH_MACRON 0x0100`
- `#define UC_LATIN_SMALL_LETTER_A_WITH_MACRON 0x0101`
- `#define UC_LATIN_CAPITAL_LETTER_A_WITH_BRIEVE 0x0102`
- `#define UC_LATIN_SMALL_LETTER_A_WITH_BRIEVE 0x0103`
- `#define UC_LATIN_CAPITAL_LETTER_A_WITH_OGONEK 0x0104`
- `#define UC_LATIN_SMALL_LETTER_A_WITH_OGONEK 0x0105`
- `#define UC_LATIN_CAPITAL_LETTER_C_WITH_ACUTE 0x0106`
- `#define UC_LATIN_SMALL_LETTER_C_WITH_ACUTE 0x0107`
- `#define UC_LATIN_CAPITAL_LETTER_C_WITH_CIRCUMFLEX 0x0108`
- `#define UC_LATIN_SMALL_LETTER_C_WITH_CIRCUMFLEX 0x0109`
- `#define UC_LATIN_CAPITAL_LETTER_C_WITH_DOT_ABOVE 0x010a`
- `#define UC_LATIN_SMALL_LETTER_C_WITH_DOT_ABOVE 0x010b`
- `#define UC_LATIN_CAPITAL_LETTER_C_WITH_CARON 0x010c`
- `#define UC_LATIN_SMALL_LETTER_C_WITH_CARON 0x010d`
- `#define UC_LATIN_CAPITAL_LETTER_D_WITH_CARON 0x010e`
- `#define UC_LATIN_SMALL_LETTER_D_WITH_CARON 0x010f`
- `#define UC_LATIN_CAPITAL_LETTER_D_WITH_STROKE 0x0110`
- `#define UC_LATIN_SMALL_LETTER_D_WITH_STROKE 0x0111`
- `#define UC_LATIN_CAPITAL_LETTER_E_WITH_MACRON 0x0112`
- `#define UC_LATIN_SMALL_LETTER_E_WITH_MACRON 0x0113`
- `#define UC_LATIN_CAPITAL_LETTER_E_WITH_DOT_ABOVE 0x0116`
- `#define UC_LATIN_SMALL_LETTER_E_WITH_DOT_ABOVE 0x0117`
- `#define UC_LATIN_CAPITAL_LETTER_E_WITH_OGONEK 0x0118`
- `#define UC_LATIN_SMALL_LETTER_E_WITH_OGONEK 0x0119`
- `#define UC_LATIN_CAPITAL_LETTER_E_WITH_CARON 0x011a`
- `#define UC_LATIN_SMALL_LETTER_E_WITH_CARON 0x011b`
- `#define UC_LATIN_CAPITAL_LETTER_G_WITH_CIRCUMFLEX 0x011c`
- `#define UC_LATIN_SMALL_LETTER_G_WITH_CIRCUMFLEX 0x011d`
- `#define UC_LATIN_CAPITAL_LETTER_G_WITH_BREVE 0x011e`
- `#define UC_LATIN_SMALL_LETTER_G_WITH_BREVE 0x011f`

- #define UC_LATIN_CAPITAL_LETTER_G_WITH_DOT_ABOVE 0x0120
- #define UC_LATIN_SMALL_LETTER_G_WITH_DOT_ABOVE 0x0121
- #define UC_LATIN_CAPITAL_LETTER_G_WITH_CEDILLA 0x0122
- #define UC_LATIN_SMALL_LETTER_G_WITH_CEDILLA 0x0123
- #define UC_LATIN_CAPITAL_LETTER_H_WITH_CIRCUMFLEX 0x0124
- #define UC_LATIN_SMALL_LETTER_H_WITH_CIRCUMFLEX 0x0125
- #define UC_LATIN_CAPITAL_LETTER_H_WITH_STROKE 0x0126
- #define UC_LATIN_SMALL_LETTER_H_WITH_STROKE 0x0127
- #define UC_LATIN_CAPITAL_LETTER_I_WITH_TILDE 0x0128
- #define UC_LATIN_SMALL_LETTER_I_WITH_TILDE 0x0129
- #define UC_LATIN_CAPITAL_LETTER_I_WITH_MACRON 0x012a
- #define UC_LATIN_SMALL_LETTER_I_WITH_MACRON 0x012b
- #define UC_LATIN_CAPITAL_LETTER_I_WITH_OGONEK 0x012e
- #define UC_LATIN_SMALL_LETTER_I_WITH_OGONEK 0x012f
- #define UC_LATIN_CAPITAL_LETTER_I_WITH_DOT_ABOVE 0x0130
- #define UC_LATIN_SMALL_LETTER_DOTLESS_I 0x0131
- #define UC_LATIN_CAPITAL_LIGATURE_IJ 0x0132
- #define UC_LATIN_SMALL_LIGATURE_IJ 0x0133
- #define UC_LATIN_CAPITAL_LETTER_J_WITH_CIRCUMFLEX 0x0134
- #define UC_LATIN_SMALL_LETTER_J_WITH_CIRCUMFLEX 0x0135
- #define UC_LATIN_CAPITAL_LETTER_K_WITH_CEDILLA 0x0136
- #define UC_LATIN_SMALL_LETTER_K_WITH_CEDILLA 0x0137
- #define UC_LATIN_SMALL_LETTER_KRA 0x0138
- #define UC_LATIN_CAPITAL_LETTER_L_WITH_ACUTE 0x0139
- #define UC_LATIN_SMALL_LETTER_L_WITH_ACUTE 0x013a
- #define UC_LATIN_CAPITAL_LETTER_L_WITH_CEDILLA 0x013b
- #define UC_LATIN_SMALL_LETTER_L_WITH_CEDILLA 0x013c
- #define UC_LATIN_CAPITAL_LETTER_L_WITH_CARON 0x013d
- #define UC_LATIN_SMALL_LETTER_L_WITH_CARON 0x013e
- #define UC_LATIN_CAPITAL_LETTER_L_WITH_MIDDLE_DOT 0x013f
- #define UC_LATIN_SMALL_LETTER_L_WITH_MIDDLE_DOT 0x0140
- #define UC_LATIN_CAPITAL_LETTER_L_WITH_STROKE 0x0141
- #define UC_LATIN_SMALL_LETTER_L_WITH_STROKE 0x0142
- #define UC_LATIN_CAPITAL_LETTER_N_WITH_ACUTE 0x0143
- #define UC_LATIN_SMALL_LETTER_N_WITH_ACUTE 0x0144
- #define UC_LATIN_CAPITAL_LETTER_N_WITH_CEDILLA 0x0145
- #define UC_LATIN_SMALL_LETTER_N_WITH_CEDILLA 0x0146
- #define UC_LATIN_CAPITAL_LETTER_N_WITH_CARON 0x0147
- #define UC_LATIN_SMALL_LETTER_N_WITH_CARON 0x0148
- #define UC_LATIN_SMALL_LETTER_N_PRECEDED_BY_APOSTROPH-
E 0x0149
- #define UC_LATIN_CAPITAL_ENG 0x014a
- #define UC_LATIN_CAPITAL_LETTER_ENG 0x014a
- #define UC_LATIN_SMALL_LETTER_ENG 0x014b
- #define UC_LATIN_CAPITAL_LETTER_O_WITH_MACRON 0x014c
- #define UC_LATIN_SMALL_LETTER_O_WITH_MACRON 0x014d

- #define UC_LATIN_CAPITAL_LETTER_O_WITH_DOUBLE_ACUTE 0x0150
- #define UC_LATIN_SMALL_LETTER_O_WITH_DOUBLE_ACUTE 0x0151
- #define UC_LATIN_CAPITAL_LIGATURE_OE 0x0152
- #define UC_LATIN_SMALL_LIGATURE_OE 0x0153
- #define UC_LATIN_CAPITAL_LETTER_R_WITH_ACUTE 0x0154
- #define UC_LATIN_SMALL_LETTER_R_WITH_ACUTE 0x0155
- #define UC_LATIN_CAPITAL_LETTER_R_WITH_CEDILLA 0x0156
- #define UC_LATIN_SMALL_LETTER_R_WITH_CEDILLA 0x0157
- #define UC_LATIN_CAPITAL_LETTER_R_WITH_CARON 0x0158
- #define UC_LATIN_SMALL_LETTER_R_WITH_CARON 0x0159
- #define UC_LATIN_CAPITAL_LETTER_S_WITH_ACUTE 0x015a
- #define UC_LATIN_SMALL_LETTER_S_WITH_ACUTE 0x015b
- #define UC_LATIN_CAPITAL_LETTER_S_WITH_CIRCUMFLEX 0x015c
- #define UC_LATIN_SMALL_LETTER_S_WITH_CIRCUMFLEX 0x015d
- #define UC_LATIN_CAPITAL_LETTER_S_WITH_CEDILLA 0x015e
- #define UC_LATIN_SMALL_LETTER_S_WITH_CEDILLA 0x015f
- #define UC_LATIN_CAPITAL_LETTER_S_WITH_CARON 0x0160
- #define UC_LATIN_SMALL_LETTER_S_WITH_CARON 0x0161
- #define UC_LATIN_CAPITAL_LETTER_T_WITH_CEDILLA 0x0162
- #define UC_LATIN_SMALL_LETTER_T_WITH_CEDILLA 0x0163
- #define UC_LATIN_CAPITAL_LETTER_T_WITH_CARON 0x0164
- #define UC_LATIN_SMALL_LETTER_T_WITH_CARON 0x0165
- #define UC_LATIN_CAPITAL_LETTER_T_WITH_STROKE 0x0166
- #define UC_LATIN_SMALL_LETTER_T_WITH_STROKE 0x0167
- #define UC_LATIN_CAPITAL_LETTER_U_WITH_TILDE 0x0168
- #define UC_LATIN_SMALL_LETTER_U_WITH_TILDE 0x0169
- #define UC_LATIN_CAPITAL_LETTER_U_WITH_MACRON 0x016a
- #define UC_LATIN_SMALL_LETTER_U_WITH_MACRON 0x016b
- #define UC_LATIN_CAPITAL_LETTER_U_WITH_BREVE 0x016c
- #define UC_LATIN_SMALL_LETTER_U_WITH_BREVE 0x016d
- #define UC_LATIN_CAPITAL_LETTER_U_WITH_RING_ABOVE 0x016e
- #define UC_LATIN_SMALL_LETTER_U_WITH_RING_ABOVE 0x016f
- #define UC_LATIN_CAPITAL_LETTER_U_WITH_DOUBLE_ACUTE 0x0170
- #define UC_LATIN_SMALL_LETTER_U_WITH_DOUBLE_ACUTE 0x0171
- #define UC_LATIN_CAPITAL_LETTER_U_WITH_OGONEK 0x0172
- #define UC_LATIN_SMALL_LETTER_U_WITH_OGONEK 0x0173
- #define UC_LATIN_CAPITAL_LETTER_W_WITH_CIRCUMFLEX 0x0174
- #define UC_LATIN_SMALL_LETTER_W_WITH_CIRCUMFLEX 0x0175
- #define UC_LATIN_CAPITAL_LETTER_Y_WITH_CIRCUMFLEX 0x0176
- #define UC_LATIN_SMALL_LETTER_Y_WITH_CIRCUMFLEX 0x0177
- #define UC_LATIN_CAPITAL_LETTER_Y_WITH_DIAERESIS 0x0178
- #define UC_LATIN_CAPITAL_LETTER_Z_WITH_ACUTE 0x0179
- #define UC_LATIN_SMALL_LETTER_Z_WITH_ACUTE 0x017a
- #define UC_LATIN_CAPITAL_LETTER_Z_WITH_DOT_ABOVE 0x017b
- #define UC_LATIN_SMALL_LETTER_Z_WITH_DOT_ABOVE 0x017c
- #define UC_LATIN_CAPITAL_LETTER_Z_WITH_CARON 0x017d

- `#define UC_LATIN_SMALL_LETTER_Z_WITH_CARON 0x017e`
- `#define UC_CARON 0x02c7`
- `#define UC_BREVE 0x02d8`
- `#define UC_DOT_ABOVE 0x02d9`
- `#define UC_OGONEK 0x02db`
- `#define UC_DOUBLE_ACUTE_ACCENT 0x02dd`
- `#define UC_COMBINING_GRAVE_ACCENT 0x0300`
- `#define UC_COMBINING_ACUTE_ACCENT 0x0301`
- `#define UC_COMBINING_CIRCUMFLEX_ACCENT 0x0302`
- `#define UC_COMBINING_TILDE 0x0303`
- `#define UC_COMBINING_MACRON 0x0304`
- `#define UC_COMBINING_BREVE 0x0306`
- `#define UC_COMBINING_DOT_ABOVE 0x0307`
- `#define UC_COMBINING_DIAERESIS 0x0308`
- `#define UC_COMBINING_RING_ABOVE 0x030a`
- `#define UC_COMBINING_DOUBLE_ACUTE_ACCENT 0x030b`
- `#define UC_COMBINING_CARON 0x030c`
- `#define UC_COMBINING_CEDILLA 0x0327`
- `#define UC_COMBINING_OGONEK 0x0328`
- `#define UC_GREEK_TONOS 0x0384`
- `#define UC_GREEK_DIALYTIKA_TONOS 0x0385`
- `#define UC_GREEK_CAPITAL_LETTER_ALPHA_WITH_TONOS 0x0386`
- `#define UC_GREEK_ANO_TELEIA 0x0387`
- `#define UC_GREEK_CAPITAL_LETTER_EPSILON_WITH_TONOS 0x0388`
- `#define UC_GREEK_CAPITAL_LETTER_ETA_WITH_TONOS 0x0389`
- `#define UC_GREEK_CAPITAL_LETTER_IOTA_WITH_TONOS 0x038a`
- `#define UC_GREEK_CAPITAL_LETTER_OMICRON_WITH_TONOS 0x038c`
- `#define UC_GREEK_CAPITAL_LETTER_UPSILON_WITH_TONOS 0x038e`
- `#define UC_GREEK_CAPITAL_LETTER_OMEGA_WITH_TONOS 0x038f`
- `#define UC_GREEK_SMALL_LETTER_IOTA_WITH_DIALYTIKA_AND_TONOS 0x0390`
- `#define UC_GREEK_CAPITAL_LETTER_ALPHA 0x0391`
- `#define UC_GREEK_CAPITAL_LETTER_BETA 0x0392`
- `#define UC_GREEK_CAPITAL_LETTER_GAMMA 0x0393`
- `#define UC_GREEK_CAPITAL_LETTER_DELTA 0x0394`
- `#define UC_GREEK_CAPITAL_LETTER_EPSILON 0x0395`
- `#define UC_GREEK_CAPITAL_LETTER_ZETA 0x0396`
- `#define UC_GREEK_CAPITAL_LETTER_ETA 0x0397`
- `#define UC_GREEK_CAPITAL_LETTER_THETA 0x0398`
- `#define UC_GREEK_CAPITAL_LETTER_IOTA 0x0399`
- `#define UC_GREEK_CAPITAL_LETTER_KAPPA 0x039a`
- `#define UC_GREEK_CAPITAL_LETTER_LAMDA 0x039b`
- `#define UC_GREEK_CAPITAL_LETTER_MU 0x039c`
- `#define UC_GREEK_CAPITAL_LETTER_NU 0x039d`
- `#define UC_GREEK_CAPITAL_LETTER_XI 0x039e`
- `#define UC_GREEK_CAPITAL_LETTER_OMICRON 0x039f`

- `#define UC_GREEK_CAPITAL_LETTER_PI 0x03a0`
- `#define UC_GREEK_CAPITAL_LETTER_RHO 0x03a1`
- `#define UC_GREEK_CAPITAL_LETTER_SIGMA 0x03a3`
- `#define UC_GREEK_CAPITAL_LETTER_TAU 0x03a4`
- `#define UC_GREEK_CAPITAL_LETTER_UPSILON 0x03a5`
- `#define UC_GREEK_CAPITAL_LETTER_PHI 0x03a6`
- `#define UC_GREEK_CAPITAL_LETTER_CHI 0x03a7`
- `#define UC_GREEK_CAPITAL_LETTER_PSI 0x03a8`
- `#define UC_GREEK_CAPITAL_LETTER_OMEGA 0x03a9`
- `#define UC_GREEK_CAPITAL_LETTER_IOTA_WITH_DIALYTIKA 0x03aa`
- `#define UC_GREEK_CAPITAL_LETTER_UPSILON_WITH_DIALYTIKA 0x03ab`
- `#define UC_GREEK_SMALL_LETTER_ALPHA_WITH_TONOS 0x03ac`
- `#define UC_GREEK_SMALL_LETTER_EPSILON_WITH_TONOS 0x03ad`
- `#define UC_GREEK_SMALL_LETTER_ETA_WITH_TONOS 0x03ae`
- `#define UC_GREEK_SMALL_LETTER_IOTA_WITH_TONOS 0x03af`
- `#define UC_GREEK_SMALL_LETTER_UPSILON_WITH_DIALYTIKA_AND_TONOS 0x03b0`
- `#define UC_GREEK_SMALL_LETTER_ALPHA 0x03b1`
- `#define UC_GREEK_SMALL_LETTER_BETA 0x03b2`
- `#define UC_GREEK_SMALL_LETTER_GAMMA 0x03b3`
- `#define UC_GREEK_SMALL_LETTER_DELTA 0x03b4`
- `#define UC_GREEK_SMALL_LETTER_EPSILON 0x03b5`
- `#define UC_GREEK_SMALL_LETTER_ZETA 0x03b6`
- `#define UC_GREEK_SMALL_LETTER_ETA 0x03b7`
- `#define UC_GREEK_SMALL_LETTER_THETA 0x03b8`
- `#define UC_GREEK_SMALL_LETTER_IOTA 0x03b9`
- `#define UC_GREEK_SMALL_LETTER_KAPPA 0x03ba`
- `#define UC_GREEK_SMALL_LETTER_LAMDA 0x03bb`
- `#define UC_GREEK_SMALL_LETTER_MU 0x03bc`
- `#define UC_GREEK_SMALL_LETTER_NU 0x03bd`
- `#define UC_GREEK_SMALL_LETTER_XI 0x03be`
- `#define UC_GREEK_SMALL_LETTER_OMICRON 0x03bf`
- `#define UC_GREEK_SMALL_LETTER_PI 0x03c0`
- `#define UC_GREEK_SMALL_LETTER_RHO 0x03c1`
- `#define UC_GREEK_SMALL_LETTER_FINAL_SIGMA 0x03c2`
- `#define UC_GREEK_SMALL_LETTER_SIGMA 0x03c3`
- `#define UC_GREEK_SMALL_LETTER_TAU 0x03c4`
- `#define UC_GREEK_SMALL_LETTER_UPSILON 0x03c5`
- `#define UC_GREEK_SMALL_LETTER_PHI 0x03c6`
- `#define UC_GREEK_SMALL_LETTER_CHI 0x03c7`
- `#define UC_GREEK_SMALL_LETTER_PSI 0x03c8`
- `#define UC_GREEK_SMALL_LETTER_OMEGA 0x03c9`
- `#define UC_GREEK_SMALL_LETTER_IOTA_WITH_DIALYTIKA 0x03ca`
- `#define UC_GREEK_SMALL_LETTER_UPSILON_WITH_DIALYTIKA 0x03cb`
- `#define UC_GREEK_SMALL_LETTER_OMICRON_WITH_TONOS 0x03cc`

- #define UC_GREEK_SMALL_LETTER_UPSILON_WITH_TONOS 0x03cd
- #define UC_GREEK_SMALL_LETTER_OMEGA_WITH_TONOS 0x03ce
- #define UC_CYRILLIC_CAPITAL_LETTER_IO 0x0401
- #define UC_CYRILLIC_CAPITAL_LETTER_DJE 0x0402
- #define UC_CYRILLIC_CAPITAL_LETTER_GJE 0x0403
- #define UC_CYRILLIC_CAPITAL_LETTER_UKRAINIAN_IE 0x0404
- #define UC_CYRILLIC_CAPITAL_LETTER_DZE 0x0405
- #define UC_CYRILLIC_CAPITAL_LETTER_BYELORUSSIAN_UKRAINIAN_I 0x0406
- #define UC_CYRILLIC_CAPITAL_LETTER_YI 0x0407
- #define UC_CYRILLIC_CAPITAL_LETTER_JE 0x0408
- #define UC_CYRILLIC_CAPITAL_LETTER_LJE 0x0409
- #define UC_CYRILLIC_CAPITAL_LETTER_NJE 0x040a
- #define UC_CYRILLIC_CAPITAL_LETTER_TSHE 0x040b
- #define UC_CYRILLIC_CAPITAL_LETTER_KJE 0x040c
- #define UC_CYRILLIC_CAPITAL_LETTER_SHORT_U 0x040e
- #define UC_CYRILLIC_CAPITAL_LETTER_DZHE 0x040f
- #define UC_CYRILLIC_CAPITAL_LETTER_A 0x0410
- #define UC_CYRILLIC_CAPITAL_LETTER_BE 0x0411
- #define UC_CYRILLIC_CAPITAL_LETTER_VE 0x0412
- #define UC_CYRILLIC_CAPITAL_LETTER_GHE 0x0413
- #define UC_CYRILLIC_CAPITAL_LETTER_DE 0x0414
- #define UC_CYRILLIC_CAPITAL_LETTER_IE 0x0415
- #define UC_CYRILLIC_CAPITAL_LETTER_ZHE 0x0416
- #define UC_CYRILLIC_CAPITAL_LETTER_ZE 0x0417
- #define UC_CYRILLIC_CAPITAL_LETTER_I 0x0418
- #define UC_CYRILLIC_CAPITAL_LETTER_SHORT_I 0x0419
- #define UC_CYRILLIC_CAPITAL_LETTER_KA 0x041a
- #define UC_CYRILLIC_CAPITAL_LETTER_EL 0x041b
- #define UC_CYRILLIC_CAPITAL_LETTER_EM 0x041c
- #define UC_CYRILLIC_CAPITAL_LETTER_EN 0x041d
- #define UC_CYRILLIC_CAPITAL_LETTER_O 0x041e
- #define UC_CYRILLIC_CAPITAL_LETTER_PE 0x041f
- #define UC_CYRILLIC_CAPITAL_LETTER_ER 0x0420
- #define UC_CYRILLIC_CAPITAL_LETTER_ES 0x0421
- #define UC_CYRILLIC_CAPITAL_LETTER_TE 0x0422
- #define UC_CYRILLIC_CAPITAL_LETTER_U 0x0423
- #define UC_CYRILLIC_CAPITAL_LETTER_EF 0x0424
- #define UC_CYRILLIC_CAPITAL_LETTER_HA 0x0425
- #define UC_CYRILLIC_CAPITAL_LETTER_TSE 0x0426
- #define UC_CYRILLIC_CAPITAL_LETTER_CHE 0x0427
- #define UC_CYRILLIC_CAPITAL_LETTER_SHA 0x0428
- #define UC_CYRILLIC_CAPITAL_LETTER_SHCHA 0x0429
- #define UC_CYRILLIC_CAPITAL_LETTER_HARD_SIGN 0x042a
- #define UC_CYRILLIC_CAPITAL_LETTER_YERU 0x042b
- #define UC_CYRILLIC_CAPITAL_LETTER_SOFT_SIGN 0x042c

- #define UC_CYRILLIC_CAPITAL_LETTER_E 0x042d
- #define UC_CYRILLIC_CAPITAL_LETTER_YU 0x042e
- #define UC_CYRILLIC_CAPITAL_LETTER_YA 0x042f
- #define UC_CYRILLIC_SMALL_LETTER_A 0x0430
- #define UC_CYRILLIC_SMALL_LETTER_BE 0x0431
- #define UC_CYRILLIC_SMALL_LETTER_VE 0x0432
- #define UC_CYRILLIC_SMALL_LETTER_GHE 0x0433
- #define UC_CYRILLIC_SMALL_LETTER_DE 0x0434
- #define UC_CYRILLIC_SMALL_LETTER_IE 0x0435
- #define UC_CYRILLIC_SMALL_LETTER_ZHE 0x0436
- #define UC_CYRILLIC_SMALL_LETTER_ZE 0x0437
- #define UC_CYRILLIC_SMALL_LETTER_I 0x0438
- #define UC_CYRILLIC_SMALL_LETTER_SHORT_I 0x0439
- #define UC_CYRILLIC_SMALL_LETTER_KA 0x043a
- #define UC_CYRILLIC_SMALL_LETTER_EL 0x043b
- #define UC_CYRILLIC_SMALL_LETTER_EM 0x043c
- #define UC_CYRILLIC_SMALL_LETTER_EN 0x043d
- #define UC_CYRILLIC_SMALL_LETTER_O 0x043e
- #define UC_CYRILLIC_SMALL_LETTER_PE 0x043f
- #define UC_CYRILLIC_SMALL_LETTER_ER 0x0440
- #define UC_CYRILLIC_SMALL_LETTER_ES 0x0441
- #define UC_CYRILLIC_SMALL_LETTER_TE 0x0442
- #define UC_CYRILLIC_SMALL_LETTER_U 0x0443
- #define UC_CYRILLIC_SMALL_LETTER_EF 0x0444
- #define UC_CYRILLIC_SMALL_LETTER_HA 0x0445
- #define UC_CYRILLIC_SMALL_LETTER_TSE 0x0446
- #define UC_CYRILLIC_SMALL_LETTER_CHE 0x0447
- #define UC_CYRILLIC_SMALL_LETTER_SHA 0x0448
- #define UC_CYRILLIC_SMALL_LETTER_SHCHA 0x0449
- #define UC_CYRILLIC_SMALL_LETTER_HARD_SIGN 0x044a
- #define UC_CYRILLIC_SMALL_LETTER_YERU 0x044b
- #define UC_CYRILLIC_SMALL_LETTER_SOFT_SIGN 0x044c
- #define UC_CYRILLIC_SMALL_LETTER_E 0x044d
- #define UC_CYRILLIC_SMALL_LETTER_YU 0x044e
- #define UC_CYRILLIC_SMALL_LETTER_YA 0x044f
- #define UC_CYRILLIC_SMALL_LETTER_IO 0x0451
- #define UC_CYRILLIC_SMALL_LETTER_DJE 0x0452
- #define UC_CYRILLIC_SMALL_LETTER_GJE 0x0453
- #define UC_CYRILLIC_SMALL_LETTER_UKRAINIAN_IE 0x0454
- #define UC_CYRILLIC_SMALL_LETTER_DZE 0x0455
- #define UC_CYRILLIC_SMALL_LETTER_BYELORUSSIAN_UKRAINIAN_-
I 0x0456
- #define UC_CYRILLIC_SMALL_LETTER_YI 0x0457
- #define UC_CYRILLIC_SMALL_LETTER_JE 0x0458
- #define UC_CYRILLIC_SMALL_LETTER_LJE 0x0459
- #define UC_CYRILLIC_SMALL_LETTER_NJE 0x045a

- `#define UC_CYRILLIC_SMALL_LETTER_TSHE 0x045b`
- `#define UC_CYRILLIC_SMALL_LETTER_KJE 0x045c`
- `#define UC_CYRILLIC_SMALL_LETTER_SHORT_U 0x045e`
- `#define UC_CYRILLIC_SMALL_LETTER_DZHE 0x045f`
- `#define UC_HEBREW_LETTER_ALEF 0x05d0`
- `#define UC_HEBREW_LETTER_BET 0x05d1`
- `#define UC_HEBREW_LETTER_GIMEL 0x05d2`
- `#define UC_HEBREW_LETTER_DALET 0x05d3`
- `#define UC_HEBREW_LETTER_HE 0x05d4`
- `#define UC_HEBREW_LETTER_VAV 0x05d5`
- `#define UC_HEBREW_LETTER_ZAYIN 0x05d6`
- `#define UC_HEBREW_LETTER_HET 0x05d7`
- `#define UC_HEBREW_LETTER_TET 0x05d8`
- `#define UC_HEBREW_LETTER_YOD 0x05d9`
- `#define UC_HEBREW_LETTER_FINAL_KAF 0x05da`
- `#define UC_HEBREW_LETTER_KAF 0x05db`
- `#define UC_HEBREW_LETTER_LAMED 0x05dc`
- `#define UC_HEBREW_LETTER_FINAL_MEM 0x05dd`
- `#define UC_HEBREW_LETTER_MEM 0x05de`
- `#define UC_HEBREW_LETTER_FINAL_NUN 0x05df`
- `#define UC_HEBREW_LETTER_NUN 0x05e0`
- `#define UC_HEBREW_LETTER_SAMEKH 0x05e1`
- `#define UC_HEBREW_LETTER_AYIN 0x05e2`
- `#define UC_HEBREW_LETTER_FINAL_PE 0x05e3`
- `#define UC_HEBREW_LETTER_PE 0x05e4`
- `#define UC_HEBREW_LETTER_FINAL_TSADI 0x05e5`
- `#define UC_HEBREW_LETTER_TSADI 0x05e6`
- `#define UC_HEBREW_LETTER_QOF 0x05e7`
- `#define UC_HEBREW_LETTER_RESH 0x05e8`
- `#define UC_HEBREW_LETTER_SHIN 0x05e9`
- `#define UC_HEBREW_LETTER_TAV 0x05ea`
- `#define UC_ARABIC_COMMA 0x060c`
- `#define UC_ARABIC_SEMICOLON 0x061b`
- `#define UC_ARABIC_QUESTION_MARK 0x061f`
- `#define UC_ARABIC_LETTER_HAMZA 0x0621`
- `#define UC_ARABIC_LETTER_ALEF_WITH_MADDA_ABOVE 0x0622`
- `#define UC_ARABIC_LETTER_ALEF_WITH_HAMZA_ABOVE 0x0623`
- `#define UC_ARABIC_LETTER_WAW_WITH_HAMZA_ABOVE 0x0624`
- `#define UC_ARABIC_LETTER_ALEF_WITH_HAMZA_BELOW 0x0625`
- `#define UC_ARABIC_LETTER_YEH_WITH_HAMZA_ABOVE 0x0626`
- `#define UC_ARABIC_LETTER_ALEF 0x0627`
- `#define UC_ARABIC_LETTER_BEH 0x0628`
- `#define UC_ARABIC_LETTER_TEH_MARBUTA 0x0629`
- `#define UC_ARABIC_LETTER_TEH 0x062a`
- `#define UC_ARABIC_LETTER_THEH 0x062b`
- `#define UC_ARABIC_LETTER_JEEM 0x062c`

- `#define UC_ARABIC_LETTER_HAH 0x062d`
- `#define UC_ARABIC_LETTER_KHAH 0x062e`
- `#define UC_ARABIC_LETTER_DAL 0x062f`
- `#define UC_ARABIC_LETTER_THAL 0x0630`
- `#define UC_ARABIC_LETTER_REH 0x0631`
- `#define UC_ARABIC_LETTER_ZAIN 0x0632`
- `#define UC_ARABIC_LETTER_SEEN 0x0633`
- `#define UC_ARABIC_LETTER_SHEEN 0x0634`
- `#define UC_ARABIC_LETTER_SAD 0x0635`
- `#define UC_ARABIC_LETTER_DAD 0x0636`
- `#define UC_ARABIC_LETTER_TAH 0x0637`
- `#define UC_ARABIC_LETTER_ZAH 0x0638`
- `#define UC_ARABIC_LETTER_AIN 0x0639`
- `#define UC_ARABIC_LETTER_GHAIN 0x063a`
- `#define UC_ARABIC_TATWEEL 0x0640`
- `#define UC_ARABIC_LETTER_FEH 0x0641`
- `#define UC_ARABIC_LETTER_QAF 0x0642`
- `#define UC_ARABIC_LETTER_KAF 0x0643`
- `#define UC_ARABIC_LETTER_LAM 0x0644`
- `#define UC_ARABIC_LETTER_MEEM 0x0645`
- `#define UC_ARABIC_LETTER_NOON 0x0646`
- `#define UC_ARABIC_LETTER_HEH 0x0647`
- `#define UC_ARABIC_LETTER_WAW 0x0648`
- `#define UC_ARABIC_LETTER_ALEF_MAKSURA 0x0649`
- `#define UC_ARABIC_LETTER_YEH 0x064a`
- `#define UC_ARABIC_FATHATAN 0x064b`
- `#define UC_ARABIC_DAMMATAN 0x064c`
- `#define UC_ARABIC_KASRATAN 0x064d`
- `#define UC_ARABIC_FATHA 0x064e`
- `#define UC_ARABIC_DAMMA 0x064f`
- `#define UC_ARABIC_KASRA 0x0650`
- `#define UC_ARABIC_SHADDA 0x0651`
- `#define UC_ARABIC_SUKUN 0x0652`
- `#define UC_ARABIC_DECIMAL_SEPARATOR 0x066B`
- `#define UC_LATIN_CAPITAL_LETTER_A_WITH_RING_BELOW 0x1E00`
- `#define UC_LATIN_SMALL_LETTER_A_WITH_RING_BELOW 0x1E01`
- `#define UC_LATIN_CAPITAL_LETTER_B_WITH_DOT_ABOVE 0x1E02`
- `#define UC_LATIN_SMALL_LETTER_B_WITH_DOT_ABOVE 0x1E03`
- `#define UC_LATIN_CAPITAL_LETTER_B_WITH_DOT_BELOW 0x1E04`
- `#define UC_LATIN_SMALL_LETTER_B_WITH_DOT_BELOW 0x1E05`
- `#define UC_LATIN_CAPITAL_LETTER_B_WITH_LINE_BELOW 0x1E06`
- `#define UC_LATIN_SMALL_LETTER_B_WITH_LINE_BELOW 0x1E07`
- `#define UC_LATIN_CAPITAL_LETTER_C_WITH_CEDILLA_AND_ACUTE 0x1E08`
- `#define UC_LATIN_SMALL_LETTER_C_WITH_CEDILLA_AND_ACUTE 0x1E09`

- #define UC_LATIN_CAPITAL_LETTER_D_WITH_DOT_ABOVE 0x1E0A
- #define UC_LATIN_SMALL_LETTER_D_WITH_DOT_ABOVE 0x1E0B
- #define UC_LATIN_CAPITAL_LETTER_D_WITH_DOT_BELOW 0x1E0C
- #define UC_LATIN_SMALL_LETTER_D_WITH_DOT_BELOW 0x1E0D
- #define UC_LATIN_CAPITAL_LETTER_D_WITH_LINE_BELOW 0x1E0E
- #define UC_LATIN_SMALL_LETTER_D_WITH_LINE_BELOW 0x1E0F
- #define UC_LATIN_CAPITAL_LETTER_D_WITH_CEDILLA 0x1E10
- #define UC_LATIN_SMALL_LETTER_D_WITH_CEDILLA 0x1E11
- #define UC_LATIN_CAPITAL_LETTER_D_WITH_CIRCUMFLEX_BELOW 0x1E12
- #define UC_LATIN_SMALL_LETTER_D_WITH_CIRCUMFLEX_BELOW 0x1E13
- #define UC_LATIN_CAPITAL_LETTER_E_WITH_MACRON_AND_GRAVE 0x1E14
- #define UC_LATIN_SMALL_LETTER_E_WITH_MACRON_AND_GRAVE 0x1E15
- #define UC_LATIN_CAPITAL_LETTER_E_WITH_MACRON_AND_ACUTE 0x1E16
- #define UC_LATIN_SMALL_LETTER_E_WITH_MACRON_AND_ACUTE 0x1E17
- #define UC_LATIN_CAPITAL_LETTER_E_WITH_CIRCUMFLEX_BELOW 0x1E18
- #define UC_LATIN_SMALL_LETTER_E_WITH_CIRCUMFLEX_BELOW 0x1E19
- #define UC_LATIN_CAPITAL_LETTER_E_WITH_TILDE_BELOW 0x1E1A
- #define UC_LATIN_SMALL_LETTER_E_WITH_TILDE_BELOW 0x1E1B
- #define UC_LATIN_CAPITAL_LETTER_E_WITH_CEDILLA_AND_BREVE 0x1E1C
- #define UC_LATIN_SMALL_LETTER_E_WITH_CEDILLA_AND_BREVE 0x1E1D
- #define UC_LATIN_CAPITAL_LETTER_F_WITH_DOT_ABOVE 0x1E1E
- #define UC_LATIN_SMALL_LETTER_F_WITH_DOT_ABOVE 0x1E1F
- #define UC_LATIN_CAPITAL_LETTER_G_WITH_MACRON 0x1E20
- #define UC_LATIN_SMALL_LETTER_G_WITH_MACRON 0x1E21
- #define UC_LATIN_CAPITAL_LETTER_H_WITH_DOT_ABOVE 0x1E22
- #define UC_LATIN_SMALL_LETTER_H_WITH_DOT_ABOVE 0x1E23
- #define UC_LATIN_CAPITAL_LETTER_H_WITH_DOT_BELOW 0x1E24
- #define UC_LATIN_SMALL_LETTER_H_WITH_DOT_BELOW 0x1E25
- #define UC_LATIN_CAPITAL_LETTER_H_WITH_DIAERESIS 0x1E26
- #define UC_LATIN_SMALL_LETTER_H_WITH_DIAERESIS 0x1E27
- #define UC_LATIN_CAPITAL_LETTER_H_WITH_CEDILLA 0x1E28
- #define UC_LATIN_SMALL_LETTER_H_WITH_CEDILLA 0x1E29
- #define UC_LATIN_CAPITAL_LETTER_H_WITH_BREVE_BELOW 0x1E2A
- #define UC_LATIN_SMALL_LETTER_H_WITH_BREVE_BELOW 0x1E2B
- #define UC_LATIN_CAPITAL_LETTER_I_WITH_TILDE_BELOW 0x1E2C
- #define UC_LATIN_SMALL_LETTER_I_WITH_TILDE_BELOW 0x1E2D

- `#define UC_LATIN_CAPITAL_LETTER_I_WITH_DIAERESIS_AND_ACUTE 0x1E2E`
- `#define UC_LATIN_SMALL_LETTER_I_WITH_DIAERESIS_AND_ACUTE 0x1E2F`
- `#define UC_LATIN_CAPITAL_LETTER_K_WITH_ACUTE 0x1E30`
- `#define UC_LATIN_SMALL_LETTER_K_WITH_ACUTE 0x1E31`
- `#define UC_LATIN_CAPITAL_LETTER_K_WITH_DOT_BELOW 0x1E32`
- `#define UC_LATIN_SMALL_LETTER_K_WITH_DOT_BELOW 0x1E33`
- `#define UC_LATIN_CAPITAL_LETTER_K_WITH_LINE_BELOW 0x1E34`
- `#define UC_LATIN_SMALL_LETTER_K_WITH_LINE_BELOW 0x1E35`
- `#define UC_LATIN_CAPITAL_LETTER_L_WITH_DOT_BELOW 0x1E36`
- `#define UC_LATIN_SMALL_LETTER_L_WITH_DOT_BELOW 0x1E37`
- `#define UC_LATIN_CAPITAL_LETTER_L_WITH_DOT_BELOW_AND_MACRON 0x1E38`
- `#define UC_LATIN_SMALL_LETTER_L_WITH_DOT_BELOW_AND_MACRON 0x1E39`
- `#define UC_LATIN_CAPITAL_LETTER_L_WITH_LINE_BELOW 0x1E3A`
- `#define UC_LATIN_SMALL_LETTER_L_WITH_LINE_BELOW 0x1E3B`
- `#define UC_LATIN_CAPITAL_LETTER_L_WITH_CIRCUMFLEX_BELOW 0x1E3C`
- `#define UC_LATIN_SMALL_LETTER_L_WITH_CIRCUMFLEX_BELOW 0x1E3D`
- `#define UC_LATIN_CAPITAL_LETTER_M_WITH_ACUTE 0x1E3E`
- `#define UC_LATIN_SMALL_LETTER_M_WITH_ACUTE 0x1E3F`
- `#define UC_LATIN_CAPITAL_LETTER_M_WITH_DOT_ABOVE 0x1E40`
- `#define UC_LATIN_SMALL_LETTER_M_WITH_DOT_ABOVE 0x1E41`
- `#define UC_LATIN_CAPITAL_LETTER_M_WITH_DOT_BELOW 0x1E42`
- `#define UC_LATIN_SMALL_LETTER_M_WITH_DOT_BELOW 0x1E43`
- `#define UC_LATIN_CAPITAL_LETTER_N_WITH_DOT_ABOVE 0x1E44`
- `#define UC_LATIN_SMALL_LETTER_N_WITH_DOT_ABOVE 0x1E45`
- `#define UC_LATIN_CAPITAL_LETTER_N_WITH_DOT_BELOW 0x1E46`
- `#define UC_LATIN_SMALL_LETTER_N_WITH_DOT_BELOW 0x1E47`
- `#define UC_LATIN_CAPITAL_LETTER_N_WITH_LINE_BELOW 0x1E48`
- `#define UC_LATIN_SMALL_LETTER_N_WITH_LINE_BELOW 0x1E49`
- `#define UC_LATIN_CAPITAL_LETTER_N_WITH_CIRCUMFLEX_BELOW 0x1E4A`
- `#define UC_LATIN_SMALL_LETTER_N_WITH_CIRCUMFLEX_BELOW 0x1E4B`
- `#define UC_LATIN_CAPITAL_LETTER_O_WITH_TILDE_AND_ACUTE 0x1E4C`
- `#define UC_LATIN_SMALL_LETTER_O_WITH_TILDE_AND_ACUTE 0x1E4D`
- `#define UC_LATIN_CAPITAL_LETTER_O_WITH_TILDE_AND_DIAERESIS 0x1E4E`
- `#define UC_LATIN_SMALL_LETTER_O_WITH_TILDE_AND_DIAERESIS 0x1E4F`
- `#define UC_LATIN_CAPITAL_LETTER_O_WITH_MACRON_AND_GRAVE 0x1E50`

- `#define UC_LATIN_SMALL_LETTER_O_WITH_MACRON_AND_GRAVE 0x1E51`
- `#define UC_LATIN_CAPITAL_LETTER_O_WITH_MACRON_AND_ACUTE 0x1E52`
- `#define UC_LATIN_SMALL_LETTER_O_WITH_MACRON_AND_ACUTE 0x1E53`
- `#define UC_LATIN_CAPITAL_LETTER_P_WITH_ACUTE 0x1E54`
- `#define UC_LATIN_SMALL_LETTER_P_WITH_ACUTE 0x1E55`
- `#define UC_LATIN_CAPITAL_LETTER_P_WITH_DOT_ABOVE 0x1E56`
- `#define UC_LATIN_SMALL_LETTER_P_WITH_DOT_ABOVE 0x1E57`
- `#define UC_LATIN_CAPITAL_LETTER_R_WITH_DOT_ABOVE 0x1E58`
- `#define UC_LATIN_SMALL_LETTER_R_WITH_DOT_ABOVE 0x1E59`
- `#define UC_LATIN_CAPITAL_LETTER_R_WITH_DOT_BELOW 0x1E5A`
- `#define UC_LATIN_SMALL_LETTER_R_WITH_DOT_BELOW 0x1E5B`
- `#define UC_LATIN_CAPITAL_LETTER_R_WITH_DOT_BELOW_AND_MACRON 0x1E5C`
- `#define UC_LATIN_SMALL_LETTER_R_WITH_DOT_BELOW_AND_MACRON 0x1E5D`
- `#define UC_LATIN_CAPITAL_LETTER_R_WITH_LINE_BELOW 0x1E5E`
- `#define UC_LATIN_SMALL_LETTER_R_WITH_LINE_BELOW 0x1E5F`
- `#define UC_LATIN_CAPITAL_LETTER_S_WITH_DOT_ABOVE 0x1E60`
- `#define UC_LATIN_SMALL_LETTER_S_WITH_DOT_ABOVE 0x1E61`
- `#define UC_LATIN_CAPITAL_LETTER_S_WITH_DOT_BELOW 0x1E62`
- `#define UC_LATIN_SMALL_LETTER_S_WITH_DOT_BELOW 0x1E63`
- `#define UC_LATIN_CAPITAL_LETTER_S_WITH_ACUTE_AND_DOT_ABOVE 0x1E64`
- `#define UC_LATIN_SMALL_LETTER_S_WITH_ACUTE_AND_DOT_ABOVE 0x1E65`
- `#define UC_LATIN_CAPITAL_LETTER_S_WITH_CARON_AND_DOT_ABOVE 0x1E66`
- `#define UC_LATIN_SMALL_LETTER_S_WITH_CARON_AND_DOT_ABOVE 0x1E67`
- `#define UC_LATIN_CAPITAL_LETTER_S_WITH_DOT_BELOW_AND_DOT_ABOVE 0x1E68`
- `#define UC_LATIN_SMALL_LETTER_S_WITH_DOT_BELOW_AND_DOT_ABOVE 0x1E69`
- `#define UC_LATIN_CAPITAL_LETTER_T_WITH_DOT_ABOVE 0x1E6A`
- `#define UC_LATIN_SMALL_LETTER_T_WITH_DOT_ABOVE 0x1E6B`
- `#define UC_LATIN_CAPITAL_LETTER_T_WITH_DOT_BELOW 0x1E6C`
- `#define UC_LATIN_SMALL_LETTER_T_WITH_DOT_BELOW 0x1E6D`
- `#define UC_LATIN_CAPITAL_LETTER_T_WITH_LINE_BELOW 0x1E6E`
- `#define UC_LATIN_SMALL_LETTER_T_WITH_LINE_BELOW 0x1E6F`
- `#define UC_LATIN_CAPITAL_LETTER_T_WITH_CIRCUMFLEX_BELOW 0x1E70`
- `#define UC_LATIN_SMALL_LETTER_T_WITH_CIRCUMFLEX_BELOW 0x1E71`
- `#define UC_LATIN_CAPITAL_LETTER_U_WITH_DIAERESIS_BELOW 0x1E72`

- #define UC_LATIN_SMALL_LETTER_U_WITH_DIAERESIS_BELOW 0x1E73
- #define UC_LATIN_CAPITAL_LETTER_U_WITH_TILDE_BELOW 0x1E74
- #define UC_LATIN_SMALL_LETTER_U_WITH_TILDE_BELOW 0x1E75
- #define UC_LATIN_CAPITAL_LETTER_U_WITH_CIRCUMFLEX_BELOW 0x1E76
- #define UC_LATIN_SMALL_LETTER_U_WITH_CIRCUMFLEX_BELOW 0x1E77
- #define UC_LATIN_CAPITAL_LETTER_U_WITH_TILDE_AND_ACUTE 0x1E78
- #define UC_LATIN_SMALL_LETTER_U_WITH_TILDE_AND_ACUTE 0x1E79
- #define UC_LATIN_CAPITAL_LETTER_U_WITH_MACRON_AND_DIAERESIS 0x1E7A
- #define UC_LATIN_SMALL_LETTER_U_WITH_MACRON_AND_DIAERESIS 0x1E7B
- #define UC_LATIN_CAPITAL_LETTER_V_WITH_TILDE 0x1E7C
- #define UC_LATIN_SMALL_LETTER_V_WITH_TILDE 0x1E7D
- #define UC_LATIN_CAPITAL_LETTER_V_WITH_DOT_BELOW 0x1E7E
- #define UC_LATIN_SMALL_LETTER_V_WITH_DOT_BELOW 0x1E7F
- #define UC_LATIN_CAPITAL_LETTER_W_WITH_GRAVE 0x1E80
- #define UC_LATIN_SMALL_LETTER_W_WITH_GRAVE 0x1E81
- #define UC_LATIN_CAPITAL_LETTER_W_WITH_ACUTE 0x1E82
- #define UC_LATIN_SMALL_LETTER_W_WITH_ACUTE 0x1E83
- #define UC_LATIN_CAPITAL_LETTER_W_WITH_DIAERESIS 0x1E84
- #define UC_LATIN_SMALL_LETTER_W_WITH_DIAERESIS 0x1E85
- #define UC_LATIN_CAPITAL_LETTER_W_WITH_DOT_ABOVE 0x1E86
- #define UC_LATIN_SMALL_LETTER_W_WITH_DOT_ABOVE 0x1E87
- #define UC_LATIN_CAPITAL_LETTER_W_WITH_DOT_BELOW 0x1E88
- #define UC_LATIN_SMALL_LETTER_W_WITH_DOT_BELOW 0x1E89
- #define UC_LATIN_CAPITAL_LETTER_X_WITH_DOT_ABOVE 0x1E8A
- #define UC_LATIN_SMALL_LETTER_X_WITH_DOT_ABOVE 0x1E8B
- #define UC_LATIN_CAPITAL_LETTER_X_WITH_DIAERESIS 0x1E8C
- #define UC_LATIN_SMALL_LETTER_X_WITH_DIAERESIS 0x1E8D
- #define UC_LATIN_CAPITAL_LETTER_Y_WITH_DOT_ABOVE 0x1E8E
- #define UC_LATIN_SMALL_LETTER_Y_WITH_DOT_ABOVE 0x1E8F
- #define UC_LATIN_CAPITAL_LETTER_Z_WITH_CIRCUMFLEX 0x1E90
- #define UC_LATIN_SMALL_LETTER_Z_WITH_CIRCUMFLEX 0x1E91
- #define UC_LATIN_CAPITAL_LETTER_Z_WITH_DOT_BELOW 0x1E92
- #define UC_LATIN_SMALL_LETTER_Z_WITH_DOT_BELOW 0x1E93
- #define UC_LATIN_CAPITAL_LETTER_Z_WITH_LINE_BELOW 0x1E94
- #define UC_LATIN_SMALL_LETTER_Z_WITH_LINE_BELOW 0x1E95
- #define UC_LATIN_SMALL_LETTER_H_WITH_LINE_BELOW 0x1E96
- #define UC_LATIN_SMALL_LETTER_T_WITH_DIAERESIS 0x1E97
- #define UC_LATIN_SMALL_LETTER_W_WITH_RING_ABOVE 0x1E98
- #define UC_LATIN_SMALL_LETTER_Y_WITH_RING_ABOVE 0x1E99
- #define UC_LATIN_SMALL_LETTER_A_WITH_RIGHT_HALF_RING 0x1E9A
- #define UC_LATIN_SMALL_LETTER_LONG_S_WITH_DOT_ABOVE 0x1E9B

- **#define UC_LATIN_SMALL_LETTER_LONG_S_WITH_DIAGONAL_STROKE** 0x1E9C
- **#define UC_LATIN_SMALL_LETTER_LONG_S_WITH_HIGH_STROKE** 0x1E9D
- **#define UC_LATIN_CAPITAL_LETTER_SHARP_S** 0x1E9E
- **#define UC_LATIN_SMALL_LETTER_DELTA** 0x1E9F
- **#define UC_LATIN_CAPITAL_LETTER_A_WITH_DOT_BELOW** 0x1EA0
- **#define UC_LATIN_SMALL_LETTER_A_WITH_DOT_BELOW** 0x1EA1
- **#define UC_LATIN_CAPITAL_LETTER_A_WITH_HOOK_ABOVE** 0x1EA2
- **#define UC_LATIN_SMALL_LETTER_A_WITH_HOOK_ABOVE** 0x1EA3
- **#define UC_LATIN_CAPITAL_LETTER_A_WITH_CIRCUMFLEX_AND_ACUTE** 0x1EA4
- **#define UC_LATIN_SMALL_LETTER_A_WITH_CIRCUMFLEX_AND_ACUTE** 0x1EA5
- **#define UC_LATIN_CAPITAL_LETTER_A_WITH_CIRCUMFLEX_AND_GRAVE** 0x1EA6
- **#define UC_LATIN_SMALL_LETTER_A_WITH_CIRCUMFLEX_AND_GRAVE** 0x1EA7
- **#define UC_LATIN_CAPITAL_LETTER_A_WITH_CIRCUMFLEX_AND_HOOK_ABOVE** 0x1EA8
- **#define UC_LATIN_SMALL_LETTER_A_WITH_CIRCUMFLEX_AND_HOOK_ABOVE** 0x1EA9
- **#define UC_LATIN_CAPITAL_LETTER_A_WITH_CIRCUMFLEX_AND_TILDE** 0x1EAA
- **#define UC_LATIN_SMALL_LETTER_A_WITH_CIRCUMFLEX_AND_TILDE** 0x1EAB
- **#define UC_LATIN_CAPITAL_LETTER_A_WITH_CIRCUMFLEX_AND_DOT_BELOW** 0x1EAC
- **#define UC_LATIN_SMALL_LETTER_A_WITH_CIRCUMFLEX_AND_DOT_BELOW** 0x1EAD
- **#define UC_LATIN_CAPITAL_LETTER_A_WITH_BREVE_AND_ACUTE** 0x1EAE
- **#define UC_LATIN_SMALL_LETTER_A_WITH_BREVE_AND_ACUTE** 0x1EAF
- **#define UC_LATIN_CAPITAL_LETTER_A_WITH_BREVE_AND_GRAVE** 0x1EB0
- **#define UC_LATIN_SMALL_LETTER_A_WITH_BREVE_AND_GRAVE** 0x1EB1
- **#define UC_LATIN_CAPITAL_LETTER_A_WITH_BREVE_AND_HOOK_ABOVE** 0x1EB2
- **#define UC_LATIN_SMALL_LETTER_A_WITH_BREVE_AND_HOOK_ABOVE** 0x1EB3
- **#define UC_LATIN_CAPITAL_LETTER_A_WITH_BREVE_AND_TILDE** 0x1EB4
- **#define UC_LATIN_SMALL_LETTER_A_WITH_BREVE_AND_TILDE** 0x1EB5
- **#define UC_LATIN_CAPITAL_LETTER_A_WITH_BREVE_AND_DOT_BELOW** 0x1EB6

- `#define UC_LATIN_SMALL_LETTER_A_WITH_BREVE_AND_DOT_BELOW 0x1EB7`
- `#define UC_LATIN_CAPITAL_LETTER_E_WITH_DOT_BELOW 0x1EB8`
- `#define UC_LATIN_SMALL_LETTER_E_WITH_DOT_BELOW 0x1EB9`
- `#define UC_LATIN_CAPITAL_LETTER_E_WITH_HOOK_ABOVE 0x1EBA`
- `#define UC_LATIN_SMALL_LETTER_E_WITH_HOOK_ABOVE 0x1EBB`
- `#define UC_LATIN_CAPITAL_LETTER_E_WITH_TILDE 0x1EBC`
- `#define UC_LATIN_SMALL_LETTER_E_WITH_TILDE 0x1EBD`
- `#define UC_LATIN_CAPITAL_LETTER_E_WITH_CIRCUMFLEX_AND_ACUTE 0x1EBE`
- `#define UC_LATIN_SMALL_LETTER_E_WITH_CIRCUMFLEX_AND_ACUTE 0x1EBF`
- `#define UC_LATIN_CAPITAL_LETTER_E_WITH_CIRCUMFLEX_AND_GRAVE 0x1EC0`
- `#define UC_LATIN_SMALL_LETTER_E_WITH_CIRCUMFLEX_AND_GRAVE 0x1EC1`
- `#define UC_LATIN_CAPITAL_LETTER_E_WITH_CIRCUMFLEX_AND_HOOK_ABOVE 0x1EC2`
- `#define UC_LATIN_SMALL_LETTER_E_WITH_CIRCUMFLEX_AND_HOOK_ABOVE 0x1EC3`
- `#define UC_LATIN_CAPITAL_LETTER_E_WITH_CIRCUMFLEX_AND_TILDE 0x1EC4`
- `#define UC_LATIN_SMALL_LETTER_E_WITH_CIRCUMFLEX_AND_TILDE 0x1EC5`
- `#define UC_LATIN_CAPITAL_LETTER_E_WITH_CIRCUMFLEX_AND_DOT_BELOW 0x1EC6`
- `#define UC_LATIN_SMALL_LETTER_E_WITH_CIRCUMFLEX_AND_DOT_BELOW 0x1EC7`
- `#define UC_LATIN_CAPITAL_LETTER_I_WITH_HOOK_ABOVE 0x1EC8`
- `#define UC_LATIN_SMALL_LETTER_I_WITH_HOOK_ABOVE 0x1EC9`
- `#define UC_LATIN_CAPITAL_LETTER_I_WITH_DOT_BELOW 0x1ECA`
- `#define UC_LATIN_SMALL_LETTER_I_WITH_DOT_BELOW 0x1ECB`
- `#define UC_LATIN_CAPITAL_LETTER_O_WITH_DOT_BELOW 0x1ECC`
- `#define UC_LATIN_SMALL_LETTER_O_WITH_DOT_BELOW 0x1ECD`
- `#define UC_LATIN_CAPITAL_LETTER_O_WITH_HOOK_ABOVE 0x1ECE`
- `#define UC_LATIN_SMALL_LETTER_O_WITH_HOOK_ABOVE 0x1ECF`
- `#define UC_LATIN_CAPITAL_LETTER_O_WITH_CIRCUMFLEX_AND_ACUTE 0x1ED0`
- `#define UC_LATIN_SMALL_LETTER_O_WITH_CIRCUMFLEX_AND_ACUTE 0x1ED1`
- `#define UC_LATIN_CAPITAL_LETTER_O_WITH_CIRCUMFLEX_AND_GRAVE 0x1ED2`
- `#define UC_LATIN_SMALL_LETTER_O_WITH_CIRCUMFLEX_AND_GRAVE 0x1ED3`
- `#define UC_LATIN_CAPITAL_LETTER_O_WITH_CIRCUMFLEX_AND_HOOK_ABOVE 0x1ED4`
- `#define UC_LATIN_SMALL_LETTER_O_WITH_CIRCUMFLEX_AND_HOOK_ABOVE 0x1ED5`

- `#define UC_LATIN_CAPITAL_LETTER_O_WITH_CIRCUMFLEX_AND_TILDE 0x1ED6`
- `#define UC_LATIN_SMALL_LETTER_O_WITH_CIRCUMFLEX_AND_TILDE 0x1ED7`
- `#define UC_LATIN_CAPITAL_LETTER_O_WITH_CIRCUMFLEX_AND_DOT_BELOW 0x1ED8`
- `#define UC_LATIN_SMALL_LETTER_O_WITH_CIRCUMFLEX_AND_DOT_BELOW 0x1ED9`
- `#define UC_LATIN_CAPITAL_LETTER_O_WITH_HORN_AND_ACUTE 0x1EDA`
- `#define UC_LATIN_SMALL_LETTER_O_WITH_HORN_AND_ACUTE 0x1EDB`
- `#define UC_LATIN_CAPITAL_LETTER_O_WITH_HORN_AND_GRAVE 0x1EDC`
- `#define UC_LATIN_SMALL_LETTER_O_WITH_HORN_AND_GRAVE 0x1EDD`
- `#define UC_LATIN_CAPITAL_LETTER_O_WITH_HORN_AND_HOOK_ABOVE 0x1EDE`
- `#define UC_LATIN_SMALL_LETTER_O_WITH_HORN_AND_HOOK_ABOVE 0x1EDF`
- `#define UC_LATIN_CAPITAL_LETTER_O_WITH_HORN_AND_TILDE 0x1EE0`
- `#define UC_LATIN_SMALL_LETTER_O_WITH_HORN_AND_TILDE 0x1EE1`
- `#define UC_LATIN_CAPITAL_LETTER_O_WITH_HORN_AND_DOT_BELOW 0x1EE2`
- `#define UC_LATIN_SMALL_LETTER_O_WITH_HORN_AND_DOT_BELOW 0x1EE3`
- `#define UC_LATIN_CAPITAL_LETTER_U_WITH_DOT_BELOW 0x1EE4`
- `#define UC_LATIN_SMALL_LETTER_U_WITH_DOT_BELOW 0x1EE5`
- `#define UC_LATIN_CAPITAL_LETTER_U_WITH_HOOK_ABOVE 0x1EE6`
- `#define UC_LATIN_SMALL_LETTER_U_WITH_HOOK_ABOVE 0x1EE7`
- `#define UC_LATIN_CAPITAL_LETTER_U_WITH_HORN_AND_ACUTE 0x1EE8`
- `#define UC_LATIN_SMALL_LETTER_U_WITH_HORN_AND_ACUTE 0x1EE9`
- `#define UC_LATIN_CAPITAL_LETTER_U_WITH_HORN_AND_GRAVE 0x1EEA`
- `#define UC_LATIN_SMALL_LETTER_U_WITH_HORN_AND_GRAVE 0x1EEB`
- `#define UC_LATIN_CAPITAL_LETTER_U_WITH_HORN_AND_HOOK_ABOVE 0x1EEC`
- `#define UC_LATIN_SMALL_LETTER_U_WITH_HORN_AND_HOOK_ABOVE 0x1EED`
- `#define UC_LATIN_CAPITAL_LETTER_U_WITH_HORN_AND_TILDE 0x1EEE`
- `#define UC_LATIN_SMALL_LETTER_U_WITH_HORN_AND_TILDE 0x1EEF`
- `#define UC_LATIN_CAPITAL_LETTER_U_WITH_HORN_AND_DOT_BELOW 0x1EF0`
- `#define UC_LATIN_SMALL_LETTER_U_WITH_HORN_AND_DOT_BELOW 0x1EF1`
- `#define UC_LATIN_CAPITAL_LETTER_Y_WITH_GRAVE 0x1EF2`

- `#define UC_LATIN_SMALL_LETTER_Y_WITH_GRAVE 0x1EF3`
- `#define UC_LATIN_CAPITAL_LETTER_Y_WITH_DOT_BELOW 0x1EF4`
- `#define UC_LATIN_SMALL_LETTER_Y_WITH_DOT_BELOW 0x1EF5`
- `#define UC_LATIN_CAPITAL_LETTER_Y_WITH_HOOK_ABOVE 0x1EF6`
- `#define UC_LATIN_SMALL_LETTER_Y_WITH_HOOK_ABOVE 0x1EF7`
- `#define UC_LATIN_CAPITAL_LETTER_Y_WITH_TILDE 0x1EF8`
- `#define UC_LATIN_SMALL_LETTER_Y_WITH_TILDE 0x1EF9`
- `#define UC_LATIN_CAPITAL_LETTER_MIDDLE_WELSH_LL 0x1EFA`
- `#define UC_LATIN_SMALL_LETTER_MIDDLE_WELSH_LL 0x1EFB`
- `#define UC_LATIN_CAPITAL_LETTER_MIDDLE_WELSH_V 0x1EFC`
- `#define UC_LATIN_SMALL_LETTER_MIDDLE_WELSH_V 0x1EFD`
- `#define UC_LATIN_CAPITAL_LETTER_Y_WITH_LOOP 0x1EFE`
- `#define UC_LATIN_SMALL_LETTER_Y_WITH_LOOP 0x1EFF`
- `#define UC_EM_DASH 0x2014`
- `#define UC_HORIZONTAL_BAR 0x2015`
- `#define UC_DOUBLE_LOW_LINE 0x2017`
- `#define UC_LEFT_SINGLE_QUOTATION_MARK 0x2018`
- `#define UC_RIGHT_SINGLE_QUOTATION_MARK 0x2019`
- `#define UC_LEFT_DOUBLE_QUOTATION_MARK 0x201c`
- `#define UC_RIGHT_DOUBLE_QUOTATION_MARK 0x201d`
- `#define UC_DOUBLE_LOW_9_QUOTATION_MARK 0x201e`
- `#define UC_EURO_SIGN 0x20ac`
- `#define UC_NUMERO_SIGN 0x2116`
- `#define UC_TRADEMARK_SIGN 0x2122`
- `#define UC_OHM_SIGN 0x2126`
- `#define UC_VULGAR_FRACTION_ONE_EIGHTH 0x215b`
- `#define UC_VULGAR_FRACTION_THREE_EIGHTHS 0x215c`
- `#define UC_VULGAR_FRACTION_FIVE_EIGHTHS 0x215d`
- `#define UC_VULGAR_FRACTION_SEVEN_EIGHTHS 0x215e`
- `#define UC_LEFTWARDS_ARROW 0x2190`
- `#define UC_UPWARDS_ARROW 0x2191`
- `#define UC_RIGHTWARDS_ARROW 0x2192`
- `#define UC_DOWNWARDS_ARROW 0x2193`
- `#define UC_EIGHTH_NOTE 0x266a`

4.52.1 Detailed Description

Contains character map tables for converting single byte ascii codes above 0xa0 to unicode codes.

Date

23/05/2001

4.53 `middleware/stb/src/huffman_table1.h` File Reference

The data for this file is subject to the signing of a license agreement that can be obtained by contacting the DTG (www.dtg.org.uk) who will then provide the tables in electronic form.

4.53.1 Detailed Description

The data for this file is subject to the signing of a license agreement that can be obtained by contacting the DTG (www.dtg.org.uk) who will then provide the tables in electronic form. The tables are of the form: 2,0 2,48 2,136 which need to be inserted into this file as a sequence of bytes from table1, as: 2, 0, 2, 48, 2, 136

Date

01/12/2004

Author

Ocean Blue

4.54 `middleware/stb/src/huffman_table2.h` File Reference

The data for this file is subject to the signing of a license agreement that can be obtained by contacting the DTG (www.dtg.org.uk) who will then provide the tables in electronic form.

4.54.1 Detailed Description

The data for this file is subject to the signing of a license agreement that can be obtained by contacting the DTG (www.dtg.org.uk) who will then provide the tables in electronic form. The tables are of the form: 2,0 2,48 2,136 which need to be inserted into this file as a sequence of bytes from table2, as: 2, 0, 2, 48, 2, 136

Date

01/12/2004

Author

Ocean Blue

4.55 `middleware/stb/src/stbds.h` File Reference

Header file - Function prototypes for DVB subtitles.

Data Structures

- struct [object](#)
- struct [clut](#)
- struct [region_object](#)
- struct [epoch_region](#)
- struct [region](#)
- struct [page_composition](#)
- struct [display_set](#)

Defines

- #define **NORMAL_CASE** 0x00
- #define **ACQUISITION_POINT** 0x01
- #define **MODE_CHANGE** 0x02
- #define **RES_PAGE_STATE** 0x03

Typedefs

- typedef struct [object](#) **S_OBJECT**
- typedef struct [clut](#) **S_CLUT**
- typedef struct [region_object](#) **S_REGION_OBJECT**
- typedef struct [epoch_region](#) **S_EPOCH_REGION**
- typedef struct [region](#) **S_REGION**
- typedef struct [page_composition](#) **S_PAGE_COMPOSITION**
- typedef struct [display_set](#) **S_DISPLAY_SET**

Functions

- **BOOLEAN STB_DSInitialiseDVBSubtitlesProcessing** (void)
- **S_CLUT * STB_DSGetClut** (**S_CLUT** *clut_list, U16BIT clut_id)
- **BOOLEAN STB_DSSegmentDDS** (U8BIT *data, U16BIT pes_len, U16BIT processed_bytes)
- **BOOLEAN STB_DSSegmentPCS** (U8BIT *data, U16BIT pes_len, U16BIT processed_bytes, **BOOLEAN** force_acquisition)
- **BOOLEAN STB_DSSegmentRCS** (U8BIT *data, U16BIT pes_len, U16BIT processed_bytes)
- **BOOLEAN STB_DSSegmentCDS** (U8BIT *data, U16BIT pes_len, U16BIT processed_bytes)
- **BOOLEAN STB_DSSegmentODS** (U8BIT *data, U16BIT pes_len, U16BIT processed_bytes)
- **BOOLEAN STB_DSSegmentEDS** (U8BIT *data, U16BIT pes_len, U16BIT processed_bytes)
- **BOOLEAN STB_DSGetPesPts** (U8BIT *data, U8BIT *pts)
- **BOOLEAN STB_DSSetDisplaySetPts** (U8BIT path, U8BIT *pts)
- **S_DISPLAY_SET * STB_DSGetDetails** (void)

- void **STB_DS**ClearDisplaySetStruct (void)
- void **STB_DS**ClearCompositionPageDetails (void)
- void **STB_DS**CheckDisplaySetTimeout ([S_DISPLAY_SET](#) *subtitle_display_set, BOOLEAN timeout_override)
- void **STB_DS**Display (U8BIT path, [S_DISPLAY_SET](#) *subtitle_display_set)
- void **STB_DS**RegisterCharRenderFunction (void(*DSCreateBitmap)(U8BIT *bitmap, U16BIT width, U16BIT height, U8BIT *char_array, U8BIT *tycrb_palette, U8BIT fgnd_col, U8BIT bkgnd_col))
- void **STB_DS**Show (void)
- void **STB_DS**Hide (void)
- void **STB_DS**ResetPhysicalDisplayRegions (void)
- void **STB_DS**ResetPhysicalCompositionRegions (void)
- void **STB_DS**TerminateDisplayCycle (void)
- void **STB_DS**CreateCompositionRegion ([S_EPOCH_REGION](#) *region, BOOLEAN page_reset)
- void **STB_DS**RenderBitmapToRegion ([S_EPOCH_REGION](#) *region_list, [S_OBJECT](#) *object, U8BIT *scan_line, U16BIT y, U16BIT w, U16BIT h)
- void **STB_DS**FillRegion (U16BIT region_id, U8BIT fillcode)
- U32BIT **STB_DS**NumPixelOperations ([S_DISPLAY_SET](#) *subtitle_display_set)
- void **STB_DS**FillEmptyRegions ([S_EPOCH_REGION](#) *epoch_region_list, [S_REGION](#) *region_list)
- void **STB_SUB**Initialise (void)
- void * **STB_DS**GetQueue (void)
- BOOLEAN **STB_DS**GetNextPesPts (U8BIT *next_pts)

4.55.1 Detailed Description

Header file - Function prototypes for DVB subtitles.

Date

25/09/2003

4.56 middleware/stb/src/stbhuffman.h File Reference

STB middleware Huffman decompression routines defined by the BBC.

Functions

- U16BIT **STB_HuffmanDecompress** (U8BIT encoding_type, U8BIT *input, U8BIT *output, U16BIT output_size)

4.56.1 Detailed Description

STB middleware Huffman decompression routines defined by the BBC.

Date

17 February 2010

Author

Steve Ford

4.57 middleware/stb/src/stbresmgr.h File Reference

STB middleware resource management module header file.

Defines

- `#define INVALID_RES_ID ((U8BIT)0xFF)` /* ID used to represent an invalid resource */

Functions

- `BOOLEAN STB_RESInitialise` (void)
- `U8BIT STB_RESNumTuners` (void)
- `U8BIT STB_RESAcquireTuner` (E_STB_DP_SIGNAL_TYPE tuner_type, void *transport, E_STB_DP_RES_OWNER owner, BOOLEAN high_priority, BOOLEAN *tuner_taken)
- `void STB_RESReleaseTuner` (U8BIT tuner_id, BOOLEAN high_priority)
- `void STB_RESSetTunerDisabled` (U8BIT tuner_id, BOOLEAN disable)

Set the disable state for a tuner. When disabled, a tuner will be ignored when acquiring a new tuner resource.
- `BOOLEAN STB_RESIsTunerDisabled` (U8BIT tuner_id)

Returns whether a tuner has been disabled or not.
- `U8BIT STB_RESNumEnabledTuners` (void)

Returns the number of tuners that aren't disabled.
- `void STB_RESSetTunerOwner` (U8BIT tuner_id, E_STB_DP_RES_OWNER owner)
- `E_STB_DP_SIGNAL_TYPE STB_RESGetTunerType` (U8BIT tuner_id)
- `void STB_RESSetTunedTransport` (U8BIT tuner_id, void *t_ptr)
- `void * STB_RESGetTunedTransport` (U8BIT tuner_id)
- `BOOLEAN STB_RESCanTuneToTransport` (E_STB_DP_SIGNAL_TYPE tuner_type, void *transport)
- `U8BIT STB_RESTunerUsageCount` (U8BIT tuner_id)
- `U8BIT STB_RESNumDemuxes` (void)
- `U8BIT STB_RESAcquireDemux` (U8BIT demux_id, U16BIT caps)

- void **STB_RESReleaseDemux** (U8BIT demux_id)
- U16BIT **STB_RESGetDemuxCaps** (U8BIT demux_id)
- U8BIT **STB_RESNumAudioDecoders** (void)
- U8BIT **STB_RESAcquireAudioDecoder** (void)
- void **STB_RESReleaseAudioDecoder** (U8BIT decoder_id)
- U8BIT **STB_RESAcquireADDecoder** (void)
- void **STB_RESReleaseADDecoder** (U8BIT decoder_id)
- U8BIT **STB_RESNumVideoDecoders** (void)
- U8BIT **STB_RESAcquireVideoDecoder** (void)
- void **STB_RESReleaseVideoDecoder** (U8BIT decoder_id)
- U8BIT **STB_RESNumCISlots** (void)
- U8BIT **STB_RESAcquireCISlot** (void *service, U8BIT *pmt_data, U8BIT *ci_protection_desc)
- BOOLEAN **STB_RESUseCISlot** (U8BIT slot_id)
- U8BIT **STB_RESGetCISlotUsageCount** (U8BIT slot_id)
- void **STB_RESReleaseCISlot** (U8BIT slot_id)

4.57.1 Detailed Description

STB middleware resource management module header file.

Date

June 2007

Author

Steve Ford

4.57.2 Function Documentation

4.57.2.1 BOOLEAN STB_RESIsTunerDisabled (U8BIT *tuner_id*)

Returns whether a tuner has been disabled or not.

Parameters

<i>tuner_id</i>	ID of tuner
-----------------	-------------

Returns

TRUE if the tuner is disabled, FALSE otherwise

4.57.2.2 U8BIT STB_RESNumEnabledTuners (void)

Returns the number of tuners that aren't disabled.

Returns

number of enabled tuners

4.57.2.3 void STB_RESSetTunerDisabled (U8BIT *tuner_id*, BOOLEAN *disable*)

Set the disable state for a tuner. When disabled, a tuner will be ignored when acquiring a new tuner resource.

Parameters

<i>tuner_id</i>	ID of tuner
<i>disable</i>	TRUE if the tuner is to be disabled, FALSE otherwise

4.58 middleware/stb/src/stbsic.h File Reference

Header file - macros and function prototypes for public use.

Defines

- #define **SI_EVENT_SEARCH** 0x00000001
- #define **SI_EVENT_UPDATE** 0x00000002
- #define **SI_EVENT_SERVICE_CHANGE** 0x00000004
- #define **SI_EVENT_EXTENDED_EVENT** 0x00000008
- #define **SI_EVENT_STOP** 0x00000010
- #define **SI_EVENT_ACTION_FILTER** 0x00000020

Enumerations

- enum **E_STB_SI_STATUS** { **SI_SEARCHING**, **SI_UPDATING**, **SI_STOPPED** }

Functions

- void **STB_SITerrInitialise** (void)
- void **STB_SITerrSendEvent** (U8BIT path, U32BIT events)
- E_STB_SI_STATUS **STB_SITerrGetStatus** (U8BIT path)

4.58.1 Detailed Description

Header file - macros and function prototypes for public use.

Date

01/03/2001

4.59 middleware/stb/src/stbvbi.h File Reference

Header file for the function prototypes for registering callback function to process - Teletext into the VBI.

Functions

- void **STB_VBIInitialise** (void)
- void **STB_VBIRemoveCallbackFunction** (void)

4.59.1 Detailed Description

Header file for the function prototypes for registering callback function to process - Teletext into the VBI.

Date

22/04/2004

Author

Ocean Blue

4.60 middleware/stb/src/version.h File Reference

Header file - library version number.

4.60.1 Detailed Description

Header file - library version number.

Date

26/03/2004

Author

Ocean Blue

4.61 middleware/stb/src/vtc.h File Reference

Blank description.

```
#include "tectype.h" #include "stbhwav.h" #include "vtctype.h" Include dependency graph for vtc.h:
```

Data Structures

- struct [s_vt_options](#)

Typedefs

- typedef struct [s_vt_options](#) **S_VT_OPTIONS**
- typedef void(* **F_VT_NOTIFICATION_CALLBACK**)(void *userdata)

Functions

- void * [VT_Open](#) ([S_VT_OPTIONS](#) *options)
Open video transformation manager.
- void [VT_Close](#) (void *context)
Close video transformation manager.
- void [VT_Enable](#) (void *context, BOOLEAN enable)
Enable or disable transformation calculations.
- void [VT_SetAfd](#) (void *context, U8BIT afd_value)
Set current AFD (active format descriptor value)
- void [VT_SetVideoAspectRatio](#) (void *context, E_ASPECT_RATIO aspect_ratio)
Set video aspect ratio.
- E_ASPECT_RATIO [VT_GetVideoAspectRatio](#) (void *context)
Get video aspect ratio.
- void [VT_SetMhegAspectRatio](#) (void *context, E_ASPECT_RATIO aspect_ratio)
Set MHEG5 scene aspect ratio.
- E_ASPECT_RATIO [VT_GetDisplayAspectRatio](#) (void *context)
Get display aspect ratio.
- void [VT_SetDisplayAspectRatio](#) (void *context, E_ASPECT_RATIO aspect_ratio)
Set display aspect ratio.
- void [VT_SetMhegScalingResolution](#) (void *context, U16BIT width, U16BIT height)
Set Voyager scaling resolution.
- void [VT_SetMhegScaling](#) (void *context, [S_RECTANGLE](#) *scaling)
Set MHEG-5 scaling information.
- void [VT_SetAppScaling](#) (void *context, [S_RECTANGLE](#) *window)
Set application scaling information.
- void [VT_SetVideoResolution](#) (void *context, U16BIT width, U16BIT height)
Set video resolution.
- void [VT_SetScreenResolution](#) (void *context, U16BIT width, U16BIT height)
Set screen resolution.
- void [VT_SetProfileMheg5](#) (void *context, BOOLEAN enable)

Set profile to apply MHEG5 option.

- void [VT_SetProfileHbbtv](#) (void *context, BOOLEAN enable)

Set profile to apply HBBTV option.

- void [VT_SetHbbtvWindow](#) (void *context, [S_RECTANGLE](#) *output)

Set HBBTV output window.

- void [VT_SetMhegVideoAlignment](#) (void *context, E_VIDEO_ASPECT_MODE wam)

Set widescreen alignment mode for MHEG-5.

- void [VT_GetVideoTransformation](#) (void *context, [S_RECTANGLE](#) *input_rect, [S_RECTANGLE](#) *output_rect)

Get the current video transformation rectangles.

- void [VT_GetWss](#) (void *context, U8BIT *wss)

Return WSS (wide-screen signalling) value.

- void [VT_SetVideoChangedCallback](#) (void *context, F_VT_NOTIFICATION_CALLBACK callback, void *user_data)

Set video change callback.

- void [VT_SetUserPreferenceChangedCallback](#) (void *context, F_VT_NOTIFICATION_CALLBACK callback, void *user_data)

Set user preference change callback.

- E_FORMAT_CONVERSION [VT_GetDecoderFormatConversion](#) (void *context)

Return the current decoder format conversion.

- void [VT_GetScreenResolution](#) (void *context, U16BIT *width, U16BIT *height)

Return the current screen resolution.

- void [VT_GetVideoResolution](#) (void *context, U16BIT *width, U16BIT *height)

Return the current video resolution.

- BOOLEAN [VT_IsOsdScaled](#) (void *context)

Check if osd must be scaled due to MHEG scene aspect ratio.

- void [VT_SetVideoAlignmentPref](#) (void *context, E_VIDEO_ASPECT_MODE alignment)

Set user preference for video aspect ratio.

- void [VT_SetDecoderStatus](#) (void *context, E_STB_AV_DECODER_STATUS status)

Set the decoder status.

- E_STB_AV_DECODER_STATUS [VT_GetDecoderStatus](#) (void *context)

Get the decoder status.

4.61.1 Detailed Description

Blank description.

Date

01/12/2004

Author

Ocean Blue

4.61.2 Function Documentation**4.61.2.1 void VT_Close (void * *context*)**

Close video transformation manager.

Parameters

<i>context</i>	- transformation calculator context
----------------	-------------------------------------

4.61.2.2 void VT_Enable (void * *context*, BOOLEAN *enable*)

Enable or disable transformation calculations.

Parameters

<i>context</i>	- transformation calculator context
<i>enable</i>	- TRUE if calculations are enabled, FALSE otherwise

4.61.2.3 E_FORMAT_CONVERSION VT_GetDecoderFormatConversion (void * *context*)

Return the current decoder format conversion.

Parameters

<i>context</i>	- transformation calculator context
----------------	-------------------------------------

Returns

Decoder format conversion

4.61.2.4 E_STB_AV_DECODER_STATUS VT_GetDecoderStatus (void * *context*)

Get the decoder status.

Parameters

<i>context</i>	- transformation calculator context
----------------	-------------------------------------

Returns

The decoder status

4.61.2.5 E_ASPECT_RATIO VT_GetDisplayAspectRatio (void * context)

Get display aspect ratio.

Parameters

<i>context</i>	- transformation calculator context
----------------	-------------------------------------

Returns

E_ASPECT_RATIO of display

4.61.2.6 void VT_GetScreenResolution (void * context, U16BIT * width, U16BIT * height)

Return the current screen resolution.

Parameters

<i>context</i>	- transformation calculator context
<i>width</i>	- screen width
<i>height</i>	- screen height

4.61.2.7 E_ASPECT_RATIO VT_GetVideoAspectRatio (void * context)

Get video aspect ratio.

Parameters

<i>context</i>	- transformation calculator context
----------------	-------------------------------------

Returns

aspect_ratio - video aspect ratio

4.61.2.8 void VT_GetVideoResolution (void * context, U16BIT * width, U16BIT * height)

Return the current video resolution.

Parameters

<i>context</i>	- transformation calculator context
<i>width</i>	- video width
<i>height</i>	- video height

4.61.2.9 void VT_GetVideoTransformation (void * *context*, S_RECTANGLE * *input_rect*, S_RECTANGLE * *output_rect*)

Get the current video transformation rectangles.

Parameters

<i>context</i>	- transformation calculator context
<i>input_rect</i>	- input rectangle for transformation
<i>output_rect</i>	- output rectangle for transformation

4.61.2.10 void VT_GetWss (void * *context*, U8BIT * *wss*)

Return WSS (wide-screen signalling) value.

Parameters

<i>context</i>	- transformation calculator context
<i>wss</i>	- WSS value

4.61.2.11 BOOLEAN VT_IsOsdScaled (void * *context*)

Check if osd must be scaled due to MHEG scene aspect ratio.

Parameters

<i>context</i>	- transformation calculator context
----------------	-------------------------------------

Returns

TRUE if osd scaling is required, FALSE otherwise

4.61.2.12 void* VT_Open (S_VT_OPTIONS * *options*)

Open video transformation manager.

Parameters

<i>options</i>	- transformation manager options
----------------	----------------------------------

Returns

Pointer to manager context, NULL if cannot be created

4.61.2.13 void VT_SetAfd (void * *context*, U8BIT *afd_value*)

Set current AFD (active format descriptor value)

Parameters

<i>context</i>	- transformation calculator context
<i>afd_value</i>	- AFD value

4.61.2.14 void VT_SetAppScaling (void * *context*, S_RECTANGLE * *window*)

Set application scaling information.

Parameters

<i>context</i>	- transformation calculator context
<i>window</i>	- output window (screen CS)

Note

When window is NULL, application scaling is turned off

4.61.2.15 void VT_SetDecoderStatus (void * *context*, E_STB_AV_DECODER_STATUS *status*)

Set the decoder status.

Parameters

<i>context</i>	- transformation calculator context
<i>status</i>	- New decoder status

4.61.2.16 void VT_SetDisplayAspectRatio (void * *context*, E_ASPECT_RATIO *aspect_ratio*)

Set display aspect ratio.

Parameters

<i>context</i>	- transformation calculator context
<i>aspect_ratio</i>	- video aspect ratio

4.61.2.17 void VT_SetHbbtvWindow (void * *context*, S_RECTANGLE * *output*)

Set HBBTV output window.

Parameters

<i>context</i>	- transformation calculator context
<i>output</i>	- window

4.61.2.18 void VT_SetMhegAspectRatio (void * *context*, E_ASPECT_RATIO *aspect_ratio*)

Set MHEG5 scene aspect ratio.

Parameters

<i>context</i>	- transformation calculator context
<i>aspect_ratio</i>	- scene aspect ratio

4.61.2.19 void VT_SetMhegScaling (void * *context*, S_RECTANGLE * *scaling*)

Set MHEG-5 scaling information.

Parameters

<i>context</i>	- transformation calculator context
<i>scaling</i>	- scaling and positioning transformation

Note

When scaling is NULL, scaling is ignored and the behaviour will be as if full screen video is mapped to the full screen.

4.61.2.20 void VT_SetMhegScalingResolution (void * *context*, U16BIT *width*, U16BIT *height*)

Set Voyager scaling resolution.

Parameters

<i>context</i>	- transformation calculator context
<i>width</i>	
<i>height</i>	

4.61.2.21 void VT_SetMhegVideoAlignment (void * *context*, E_VIDEO_ASPECT_MODE *wam*)

Set widescreen alignment mode for MHEG-5.

Parameters

<i>context</i>	- transformation calculator context
<i>wam</i>	- widescreen alignment mode

4.61.2.22 void VT_SetProfileHbbtv (void * *context*, BOOLEAN *enable*)

Set profile to apply HBBTV option.

Parameters

<i>context</i>	- transformation calculator context
<i>enable</i>	- TRUE turns MHEG5 option on

4.61.2.23 void VT_SetProfileMheg5 (void * *context*, BOOLEAN *enable*)

Set profile to apply MHEG5 option.

Parameters

<i>context</i>	- transformation calculator context
<i>enable</i>	- TRUE turns MHEG5 option on

4.61.2.24 void VT_SetScreenResolution (void * *context*, U16BIT *width*, U16BIT *height*)

Set screen resolution.

Parameters

<i>context</i>	- transformation calculator context
<i>width</i>	- screen width (in pixels)
<i>height</i>	- screen height (in pixels)

4.61.2.25 void VT_SetUserPreferenceChangedCallback (void * *context*,
F_VT_NOTIFICATION_CALLBACK *callback*, void * *user_data*)

Set user preference change callback.

The callback is called whenever the video transformation changes as a result of a change in user preferences.

Parameters

<i>context</i>	- transformation calculator context
<i>callback</i>	- the callback to call
<i>user_data</i>	- user data to pass to the callback

4.61.2.26 void VT_SetVideoAlignmentPref (void * *context*, E_VIDEO_ASPECT_MODE
alignment)

Set user preference for video aspect ratio.

Parameters

<i>context</i>	- transformation calculator context
<i>alignment</i>	- New video alignment preference:

4.61.2.27 void VT_SetVideoAspectRatio (void * *context*, E_ASPECT_RATIO *aspect_ratio*)

Set video aspect ratio.

Parameters

<i>context</i>	- transformation calculator context
<i>aspect_ratio</i>	- video aspect ratio

4.61.2.28 void VT_SetVideoChangedCallback (void * *context*,
F_VT_NOTIFICATION_CALLBACK *callback*, void * *user_data*)

Set video change callback.

The callback is called whenever the video transformation is changed for any reason.

Parameters

<i>context</i>	- transformation calculator context
<i>callback</i>	- the callback to call
<i>user_data</i>	- user data to pass to the callback

4.61.2.29 void VT_SetVideoResolution (void * *context*, U16BIT *width*, U16BIT *height*)

Set video resolution.

Parameters

<i>context</i>	- transformation calculator context
<i>width</i>	- video width (in pixels)
<i>height</i>	- video height (in pixels)

4.62 platform/inc/osdtype.h File Reference

Header file - Function prototypes for A/V control.

This graph shows which files directly or indirectly include this file:

Data Structures

- struct [S_RECTANGLE](#)

Defines

- #define **SD_HEIGHT** 576
- #define **SD_WIDTH** 720
- #define **HD_HEIGHT** 720
- #define **HD_WIDTH** 1280
- #define **FULL_HD_HEIGHT** 1080
- #define **FULL_HD_WIDTH** 1920

Enumerations

- enum **E_ASPECT_RATIO** { **ASPECT_RATIO_4_3**, **ASPECT_RATIO_16_9**, **ASPECT_UNDEFINED** = 255 }

4.62.1 Detailed Description

Header file - Function prototypes for A/V control.

Date

06/02/2014

4.63 platform/inc/stbhwav.h File Reference

Header file - Function prototypes for A/V control.

#include "stbhwc.h" #include "osdtype.h" Include dependency graph for stbhwav.h: This graph shows which files directly or indirectly include this file:

Data Structures

- struct [S_STB_AV_COPY_PROTECTION](#)
- struct [S_STB_AV_VIDEO_INFO](#)

Typedefs

- typedef enum e_stb_av_audio_mode **E_STB_AV_AUDIO_MODE**
- typedef enum e_stb_av_decode_source **E_STB_AV_DECODE_SOURCE**
- typedef enum e_stb_av_video_codec **E_STB_AV_VIDEO_CODEC**
- typedef enum e_stb_av_audio_codec **E_STB_AV_AUDIO_CODEC**
- typedef E_ASPECT_RATIO **E_STB_AV_ASPECT_RATIO**
- typedef enum e_stb_av_video_format **E_STB_AV_VIDEO_FORMAT**
- typedef enum e_stb_av_outputs **E_STB_AV_OUTPUTS**
- typedef enum e_stb_av_sources **E_STB_AV_SOURCES**
- typedef enum e_stb_av_srm_reply **E_STB_AV_SRM_REPLY**
- typedef enum e_stb_av_audio_route **E_STB_AV_AUDIO_ROUTE**

Enumerations

- enum **e_stb_av_audio_mode** { **AV_AUDIO_STEREO** = 0, **AV_AUDIO_LEFT** = 1, **AV_AUDIO_RIGHT** = 2, **AV_AUDIO_MONO** = 3, **AV_AUDIO_MULTICHANNEL** = 4 }
- enum **e_stb_av_decode_source** { **AV_DEMUX** = 0, **AV_MEMORY** = 1 }
- enum **e_stb_av_video_codec** { **AV_VIDEO_CODEC_AUTO** = 0, **AV_VIDEO_CODEC_MPEG1** = 1, **AV_VIDEO_CODEC_MPEG2** = 2, **AV_VIDEO_CODEC_H264** = 3, **AV_VIDEO_CODEC_H265** = 4 }
- enum **e_stb_av_audio_codec** { **AV_AUDIO_CODEC_AUTO** = 0, **AV_AUDIO_CODEC_MP2** = 1, **AV_AUDIO_CODEC_MP3** = 2, **AV_AUDIO_CODEC_AC3** = 3, **AV_AUDIO_CODEC_EAC3** = 4, **AV_AUDIO_CODEC_AAC** = 5, **AV_AUDIO_CODEC_HEAAC** = 6, **AV_AUDIO_CODEC_AAC_ADTS** = 7 }
- enum **e_stb_av_video_format** { **VIDEO_FORMAT_UNDEFINED** = 255, **VIDEO_FORMAT_AUTO** = 0, **VIDEO_FORMAT_ORIGINAL**, **VIDEO_FORMAT_PALBGH**, **VIDEO_FORMAT_PALDKL**, **VIDEO_FORMAT_PALI**, **VIDEO_FORMAT_PALM**, **VIDEO_FORMAT_PALN**, **VIDEO_FORMAT_NTSC**, **VIDEO_FORMAT_SECAMBGH**, **VIDEO_FORMAT_SECAMDKL**, **VIDEO_FORMAT_576IHD**, **VIDEO_FORMAT_576PHD**, **VIDEO_FORMAT_720HD**, **VIDEO_FORMAT_720P50HD** = **VIDEO_FORMAT_720HD**, **VIDEO_FORMAT_720P60HD**, **VIDEO_FORMAT_1080IHD**, **VIDEO_FORMAT_1080P25HD**, **VIDEO_FORMAT_1080P30HD**, **VIDEO_FORMAT_1080I50HD**, **VIDEO_FORMAT_1080P50HD**, **VIDE-**

- `O_FORMAT_1080P60HD, VIDEO_FORMAT_2160P24UHD, VIDEO_FORMAT_2160P25UHD, VIDEO_FORMAT_2160P30UHD, VIDEO_FORMAT_2160P50UHD, VIDEO_FORMAT_2160P60UHD }`
- enum `e_stb_av_outputs` { `AV_OUTPUT_TV_SCART` = 0, `AV_OUTPUT_VCR_SCART` = 1, `AV_OUTPUT_AUX_SCART` = 2, `AV_OUTPUT_HDMI` = 3 }
- enum `e_stb_av_sources` { `AV_SOURCE_ENCODER_SVIDEO` = 0, `AV_SOURCE_ENCODER_RGB` = 1, `AV_SOURCE_ENCODER_COMPOSITE` = 2, `AV_SOURCE_VCR_SVIDEO` = 3, `AV_SOURCE_VCR_RGB` = 4, `AV_SOURCE_VCR_COMPOSITE` = 5, `AV_SOURCE_AUX_SVIDEO` = 6, `AV_SOURCE_AUX_RGB` = 7, `AV_SOURCE_AUX_COMPOSITE` = 8, `AV_SOURCE_DVD_SVIDEO` = 9, `AV_SOURCE_DVD_RGB` = 10, `AV_SOURCE_DVD_COMPOSITE` = 11, `AV_SOURCE_ANALOG_TUNER` = 12, `AV_SOURCE_ANALOG_TUNER_WITH_OSD` = 13, `AV_SOURCE_HDMI` = 14, `AV_SOURCE_CVBS` = 15, `AV_SOURCE_TUNER` = 16, `AV_SOURCE_NONE` = 255 }
- enum `e_stb_av_srm_reply` { `SRM_OK` = 0, `SRM_BUSY` = 1, `SRM_NOT_REQUIRED` = 2 }
- enum `E_STB_AV_VIDEO_INFO_TYPE` { `VIDEO_INFO_VIDEO_RESOLUTION` = 0x01, `VIDEO_INFO_SCREEN_RESOLUTION` = 0x02, `VIDEO_INFO_VIDEO_ASPECT_RATIO` = 0x04, `VIDEO_INFO_DISPLAY_ASPECT_RATIO` = 0x08, `VIDEO_INFO_ASPECT_MODE` = 0x10, `VIDEO_INFO_AFD` = 0x20, `VIDEO_INFO_DECODER_STATUS` = 0x40 }
- enum `E_STB_AV_DECODER_STATUS` { `DECODER_STATUS_NONE`, `DECODER_STATUS_VIDEO`, `DECODER_STATUS_IFRAME` }
- enum `E_STB_DIGITAL_AUDIO_TYPE` { `DIGITAL_AUDIO_PCM`, `DIGITAL_AUDIO_COMPRESSED` }
- enum `e_stb_av_audio_route` { `SPEAKER_TV_N_AD_HEADPHONE_TV_N_AD`, `SPEAKER_TV_ONLY_HEADPHONE_AD_ONLY`, `SPEAKER_TV_N_AD_HEADPHONE_AD_ONLY`, `SPEAKER_TV_ONLY_HEADPHONE_TV_N_AD`, `AUTO_ROUTE_UNKNOWN` }

Functions

- void [STB_AVInitialise](#) (U8BIT audio_paths, U8BIT video_paths)
Initialises the AV components.
- void [STB_AVBlankVideo](#) (U8BIT path, BOOLEAN blank)
Blanks or unblanks the video display.
- void [STB_AVSetVideoSource](#) (U8BIT path, E_STB_AV_DECODE_SOURCE source, U32BIT param)
Sets the source of the input to the given video decoder path.
- BOOLEAN [STB_AVSetVideoCodec](#) (U8BIT path, E_STB_AV_VIDEO_CODEC codec)
Sets the video codec to be used when decoding video with the given video decoder path.
- E_STB_AV_VIDEO_CODEC [STB_AVGetVideoCodec](#) (U8BIT path)
Returns the video codec previously set for the given video path. This function is currently unused within DVBCore.
- void [STB_AVStartVideoDecoding](#) (U8BIT path)

- Starts video decoding on the given video path.*

 - void [STB_AVPauseVideoDecoding](#) (U8BIT path)
- Pauses video decoding on the given video path. The video should not be blanked.*

 - void [STB_AVResumeVideoDecoding](#) (U8BIT path)
- Resumes video decoding on the given video path that has previously been paused
The video should not be unblanked.*

 - void [STB_AVStopVideoDecoding](#) (U8BIT decoder)
- Stops video decoding on the given video path. The video is not expected to be blanked.*

 - void [STB_AVSetAudioSource](#) (U8BIT path, E_STB_AV_DECODE_SOURCE source, U32BIT param)
- Sets the source of the input for the main audio on the given audio decoder path.*

 - BOOLEAN [STB_AVSetAudioCodec](#) (U8BIT path, E_STB_AV_AUDIO_CODEC codec)
- Sets the audio codec to be used when decoding audio with the given audio decoder path.*

 - E_STB_AV_AUDIO_CODEC [STB_AVGetAudioCodec](#) (U8BIT path)
- Returns the audio codec previously set for the given audio path. This function is currently unused within DVBCore.*

 - void [STB_AVChangeAudioMode](#) (U8BIT path, E_STB_AV_AUDIO_MODE mode)
- Configures the main audio channel mode (stereo/left/right) in the case where dual-mono audio is used, such that only the audio from one channel is heard.*

 - void [STB_AVStartAudioDecoding](#) (U8BIT decoder)
- Starts audio decoding on the given audio path.*

 - void [STB_AVStopAudioDecoding](#) (U8BIT decoder)
- Stops audio decoding on the given audio path.*

 - void [STB_AVSetAudioVolume](#) (U8BIT path, U8BIT vol)
- Sets the volume of the main audio output.*

 - U8BIT [STB_AVGetAudioVolume](#) (U8BIT path)
- Returns the current volume of the main audio output.*

 - void [STB_AVSetAudioMute](#) (U8BIT path, BOOLEAN mute)
- Mute or unmute the audio output on the given audio decoder path.*

 - BOOLEAN [STB_AVGetAudioMute](#) (U8BIT path)
- Returns the current mute setting of the audio output on the given path.*

 - void [STB_AVSetADSource](#) (U8BIT path, E_STB_AV_DECODE_SOURCE source, U32BIT param)
- Sets the source of the input for the audio description audio on the given audio decoder path.*

 - BOOLEAN [STB_AVSetADCodec](#) (U8BIT path, E_STB_AV_AUDIO_CODEC codec)
- Sets the codec to be used for audio description when decoding audio with the given audio decoder path.*

 - E_STB_AV_AUDIO_CODEC [STB_AVGetADCodec](#) (U8BIT path)
- Returns the codec previously set for audio description on the given audio path. This function is currently unused within DVBCore.*

- void [STB_AVChangeADMode](#) (U8BIT path, E_STB_AV_AUDIO_MODE mode)
Configures the audio description channel mode (stereo/left/right) in the case where dual-mono audio is used, such that only the audio from one channel is heard.
- void [STB_AVStartADDecoding](#) (U8BIT decoder)
Starts decoding audio description on the given audio path.
- void [STB_AVStopADDecoding](#) (U8BIT decoder)
Stops decoding audio description on the given audio path.
- void [STB_AVSetADVolume](#) (U8BIT path, U8BIT vol)
Sets the volume of the audio description output.
- U8BIT [STB_AVGetADVolume](#) (U8BIT path)
Returns the current volume of the audio description output.
- void [STB_AVGetSTC](#) (U8BIT path, U8BIT stc[5])
Returns the current 33-bit System Time Clock from the PCR PES. On some systems, this information may need to be obtained from the associated demux, which will be contained in the 'param' value when STB_AVSetVideoSource is called.
- void [STB_AVSetTVType](#) (U8BIT path, E_STB_AV_ASPECT_RATIO ratio, E_STB_AV_VIDEO_FORMAT format)
Sets the aspect ratio and signal format for the connected television.
- void [STB_AVGetScreenSize](#) (U8BIT path, U16BIT *width, U16BIT *height)
Returns the current size of the screen in pixels.
- void [STB_AVSetAVOutputSource](#) (E_STB_AV_OUTPUTS output, E_STB_AV_SOURCES source, U32BIT param)
Routes a specified AV source to a specified AV output.
- void [STB_AVSetAVOutput](#) (BOOLEAN av_on)
Turns on/off all AV outputs (e.g. for standby mode)
- void [STB_AVSetHDMIStandby](#) (BOOLEAN standby)
Sets the standby state of the HDMI output.
- U8BIT [STB_AVGetHDMISupportedModes](#) (E_STB_AV_VIDEO_FORMAT **modes)
Returns the resolutions supported by the HDMI.
- void [STB_AVGetHDMINativeResolution](#) (U16BIT *width, U16BIT *height)
Returns the native resolution, i.e. the resolution that is set with a call to STB_AVSetTVType(path, ratio, VIDEO_FORMAT_AUTO);.
- void [STB_AVENableHMDIDecoding](#) (void)
Enables AV output to HDMI.
- void [STB_AVDisableHMDIDecoding](#) (void)
Disables AV output to HDMI.
- BOOLEAN [STB_AVIsHDCPAuthenticated](#) (void)
Returns whether HDCP has authenticated.
- BOOLEAN [STB_AVSetIFrameCodec](#) (U8BIT path, E_STB_AV_VIDEO_CODEC codec)
Sets the codec to be used when decoding the next i-frame from memory.
- void [STB_AVLoadIFrame](#) (U8BIT path, U8BIT *data, U32BIT size)
Provides the video data for an i-frame for subsequent decode and display.

- void [STB_AVShowIFrame](#) (U8BIT path)
Decodes and displays the previously loaded i-frame data.
- void [STB_AVHideIFrame](#) (U8BIT path)
Hides the i-frame currently being displayed.
- E_HW_STATUS [STB_AVPlayAudioSample](#) (U8BIT path, U32BIT loop_count)
Plays back a previously loaded audio sample.
- E_HW_STATUS [STB_AVLoadAudioSample](#) (U8BIT path, U8BIT *data, U32BIT size)
Loads an audio sample for subsequent playback.
- E_HW_STATUS [STB_AVPauseAudioSample](#) (U8BIT path)
Pauses playback of an audio sample.
- E_HW_STATUS [STB_AVResumeAudioSample](#) (U8BIT path)
Resumes playback of an audio sample.
- void [STB_AVStopAudioSample](#) (U8BIT path)
Stops playback of an audio sample.
- void [STB_AVSetSpdifMode](#) (U8BIT path, E_STB_DIGITAL_AUDIO_TYPE audio_type)
Sets the SPDIF output mode, PCM or compressed audio.
- void [STB_AVSetHDMIAudioMode](#) (U8BIT path, E_STB_DIGITAL_AUDIO_TYPE audio_type)
Sets the HDMI audio output mode, PCM or compressed.
- void [STB_AVSetAudioDelay](#) (U8BIT path, U16BIT millisecond)
Sets the audio delay on the given path.
- void [STB_AVSetVideoCallback](#) (U8BIT path, void(*callback)(S_STB_AV_VIDEO_INFO *, void *), void *user_data)
Register callback for updated video information.
- void [STB_AVApplyVideoTransformation](#) (U8BIT path, S_RECTANGLE *input, S_RECTANGLE *output)
Apply video transformation.
- S16BIT [STB_AVGetMinPlaySpeed](#) (U8BIT video_decoder)
Returns minimum video play speed as a percentage.
- S16BIT [STB_AVGetMaxPlaySpeed](#) (U8BIT video_decoder)
Returns maximum video play speed as a percentage.
- S16BIT [STB_AVGetNextPlaySpeed](#) (U8BIT video_decoder, S16BIT speed, S16BIT inc, BOOLEAN include_slow_speeds)
Returns the next valid speed that is +/- inc above or below the given speed. Slow motion speeds (>-100% and < 100%) can be included.
- void [STB_AVSetCopyProtection](#) (S_STB_AV_COPY_PROTECTION *copy_protection)
Apply the specified copy protection. This function is used for CI+.
- void [STB_AVSetUhfModulatorChannel](#) (U8BIT chan)
Sets the output channel of the RF Modulator.
- U8BIT [STB_AVGetUhfModulatorChannel](#) (void)
Gets the current RF modulator channel.

- U8BIT [STB_AVGetVideoFrameRate](#) (U8BIT path)
Returns the frame rate of the video being decoded.
- E_STB_AV_SRM_REPLY [STB_AVApplySRM](#) (U8BIT path, U8BIT *data, U32BIT len)
Apply System Renewability Message (SRM) to HDCP function.
- void **STB_AVCECOneTouchPlay** (void)
- E_STB_AV_VIDEO_FORMAT **STB_AVGetHDMINativeFormat** (void)
- void **STB_AVPlayMp3** (U8BIT path, U8BIT *buffer, U32BIT buffer_size)
- void **STB_AVStopMp3** (U8BIT path)

4.63.1 Detailed Description

Header file - Function prototypes for A/V control.

Date

06/02/2001

4.63.2 Function Documentation

4.63.2.1 E_STB_AV_SRM_REPLY STB_AVApplySRM (U8BIT path, U8BIT * data, U32BIT len)

Apply System Renewability Message (SRM) to HDCP function.

Parameters

<i>path</i>	output path
<i>data</i>	SRM data
<i>len</i>	length of SRM data in bytes

Returns

SRM_OK, SRM_BUSY or SRM_NOT_REQUIRED

4.63.2.2 void STB_AVApplyVideoTransformation (U8BIT path, S_RECTANGLE * input, S_RECTANGLE * output)

Apply video transformation.

Parameters

<i>path</i>	decoder path
<i>input</i>	input video rectangle
<i>output</i>	output video rectangle

4.63.2.3 void STB_AVBlankVideo (U8BIT *path*, BOOLEAN *blank*)

Blanks or unblanks the video display.

Parameters

<i>path</i>	video path
<i>blank</i>	TRUE to blank, FALSE to unblank

4.63.2.4 void STB_AVChangeADMode (U8BIT *path*, E_STB_AV_AUDIO_MODE *mode*)

Configures the audio description channel mode (stereo/left/right) in the case where dual-mono audio is used, such that only the audio from one channel is heard.

Parameters

<i>path</i>	audio path to be configured
<i>mode</i>	audio mode to use

4.63.2.5 void STB_AVChangeAudioMode (U8BIT *path*, E_STB_AV_AUDIO_MODE *mode*)

Configures the main audio channel mode (stereo/left/right) in the case where dual-mono audio is used, such that only the audio from one channel is heard.

Parameters

<i>path</i>	audio path to be configured
<i>mode</i>	audio mode to use

4.63.2.6 E_STB_AV_AUDIO_CODEC STB_AVGetADCodec (U8BIT *path*)

Returns the codec previously set for audio description on the given audio path. This function is currently unused within DVBCore.

Parameters

<i>path</i>	audio path
-------------	------------

Returns

codec previously set

4.63.2.7 U8BIT STB_AVGetADVolume (U8BIT *path*)

Returns the current volume of the audio description output.

Parameters

<i>path</i>	audio path
-------------	------------

Returns

audio volume (0-100%)

4.63.2.8 E_STB_AV_AUDIO_CODEC STB_AVGetAudioCodec (U8BIT *path*)

Returns the audio codec previously set for the given audio path. This function is currently unused within DVBCore.

Parameters

<i>path</i>	audio path
-------------	------------

Returns

codec previously set

4.63.2.9 BOOLEAN STB_AVGetAudioMute (U8BIT *path*)

Returns the current mute setting of the audio output on the given path.

Parameters

<i>path</i>	audio path
-------------	------------

Returns

TRUE if audio is muted, FALSE otherwise

4.63.2.10 U8BIT STB_AVGetAudioVolume (U8BIT *path*)

Returns the current volume of the main audio output.

Parameters

<i>path</i>	audio path
-------------	------------

Returns

audio volume (0-100%)

4.63.2.11 void STB_AVGetHDMINativeResolution (U16BIT * *width*, U16BIT * *height*)

Returns the native resolution, i.e. the resolution that is set with a call to STB_AVSetTV-Type(*path*, *ratio*, VIDEO_FORMAT_AUTO);.

Parameters

<i>width</i>	pointer to the variable where to store the width
<i>height</i>	pointer to the variable where to store the height

**4.63.2.12 U8BIT STB_AVGetHDMISupportedModes (E_STB_AV_VIDEO_FORMAT **
modes)**

Returns the resolutions supported by the HDMI.

Parameters

<i>modes</i>	array of supported modes
--------------	--------------------------

Returns

number of supported modes

4.63.2.13 S16BIT STB_AVGetMaxPlaySpeed (U8BIT video_decoder)

Returns maximum video play speed as a percentage.

Parameters

<i>video_ - decoder</i>	video decoder path
-----------------------------	--------------------

Returns

Maximum play speed.

4.63.2.14 S16BIT STB_AVGetMinPlaySpeed (U8BIT video_decoder)

Returns minimum video play speed as a percentage.

Parameters

<i>video_ - decoder</i>	video decoder path
-----------------------------	--------------------

Returns

Minimum play speed.

**4.63.2.15 S16BIT STB_AVGetNextPlaySpeed (U8BIT video_decoder, S16BIT speed,
S16BIT inc, BOOLEAN include_slow_speeds)**

Returns the next valid speed that is +/- inc above or below the given speed. Slow motion speeds (>-100% and < 100%) can be included.

Parameters

<i>path</i>	Decode path
<i>speed</i>	Percentage speed above/below which the new speed is calculated

<i>inc</i>	number of speeds above that specified to return
<i>include_-slow_speeds</i>	selects whether speeds >-100% and <100% are included

Returns

Speed as a percentage

4.63.2.16 void STB_AVGetScreenSize (U8BIT *path*, U16BIT * *width*, U16BIT * *height*)

Returns the current size of the screen in pixels.

Parameters

<i>path</i>	0
<i>width</i>	returned width
<i>height</i>	returned height

4.63.2.17 void STB_AVGetSTC (U8BIT *path*, U8BIT *stc*[5])

Returns the current 33-bit System Time Clock from the PCR PES. On some systems, this information may need to be obtained from the associated demux, which will be contained in the 'param' value when STB_AVSetVideoSource is called.

Parameters

<i>path</i>	video path
<i>stc</i>	an array in which the STC will be returned, ordered such that stc[0] contains the MS bit (33) of the STC value and stc[4] contains the LS bits (0-7).

4.63.2.18 U8BIT STB_AVGetUhfModulatorChannel (void)

Gets the current RF modulator channel.

Returns

The RF frequency as a UHF channel number

4.63.2.19 E_STB_AV_VIDEO_CODEC STB_AVGetVideoCodec (U8BIT *path*)

Returns the video codec previously set for the given video path. This function is currently unused within DVBCore.

Parameters

<i>path</i>	video path
-------------	------------

Returns

codec previously set

4.63.2.20 U8BIT STB_AVGetVideoFrameRate (U8BIT *path*)

Returns the frame rate of the video being decoded.

Parameters

<i>path</i>	video path
-------------	------------

Returns

video frame rate in frame per seconds

4.63.2.21 void STB_AVHidelFrame (U8BIT *path*)

Hides the i-frame currently being displayed.

Parameters

<i>path</i>	video path to use
-------------	-------------------

4.63.2.22 void STB_AVInitialise (U8BIT *audio_paths*, U8BIT *video_paths*)

Initialises the AV components.

Parameters

<i>audio_paths</i>	number of audio decoders
<i>video_paths</i>	number of video decoders

4.63.2.23 BOOLEAN STB_AVIshDCPAuthenticated (void)

Returns whether HDCP has authenticated.

Returns

TRUE if authenticated, FALSE otherwise

4.63.2.24 E_HW_STATUS STB_AVLoadAudioSample (U8BIT *path*, U8BIT * *data*, U32BIT *size*)

Loads an audio sample for subsequent playback.

Parameters

<i>path</i>	the decoder path to use for playback
<i>data</i>	the audio sample data to be loaded
<i>size</i>	the size of the audio sample in bytes

Returns

HW_OK on success, error code otherwise

4.63.2.25 void STB_AVLoadIFrame (U8BIT *path*, U8BIT * *data*, U32BIT *size*)

Provides the video data for an i-frame for subsequent decode and display.

Parameters

<i>path</i>	video decode path to be used
<i>data</i>	i-frame data to be saved for decoding
<i>size</i>	size of the data in bytes

4.63.2.26 E_HW_STATUS STB_AVPauseAudioSample (U8BIT *path*)

Pauses playback of an audio sample.

Parameters

<i>path</i>	Audio path on which to pause
-------------	------------------------------

Returns

HW_OK on success, error code otherwise

4.63.2.27 void STB_AVPauseVideoDecoding (U8BIT *path*)

Pauses video decoding on the given video path. The video should not be blanked.

Parameters

<i>path</i>	video decoder path to be paused
-------------	---------------------------------

4.63.2.28 E_HW_STATUS STB_AVPlayAudioSample (U8BIT *path*, U32BIT *loop_count*)

Plays back a previously loaded audio sample.

Parameters

<i>path</i>	the audio path to use for playback
<i>loop_count</i>	the number of times to play the sample, 0=forever

Returns

HW_OK on success, error code otherwise

4.63.2.29 E_HW_STATUS STB_AVResumeAudioSample (U8BIT *path*)

Resumes playback of an audio sample.

Parameters

<i>path</i>	Audio path on which to resume
-------------	-------------------------------

Returns

HW_OK on success, error code otherwise

4.63.2.30 void STB_AVResumeVideoDecoding (U8BIT *path*)

Resumes video decoding on the given video path that has previously been paused The video should not be unblanked.

Parameters

<i>path</i>	video decoder path to be resumed
-------------	----------------------------------

4.63.2.31 BOOLEAN STB_AVSetADCodec (U8BIT *path*, E_STB_AV_AUDIO_CODEC *codec*)

Sets the codec to be used for audio description when decoding audio with the given audio decoder path.

Parameters

<i>path</i>	audio path
<i>codec</i>	codec to be used

Returns

TRUE if the codec is supported and is set correctly, FALSE otherwise

4.63.2.32 void STB_AVSetADSource (U8BIT *path*, E_STB_AV_DECODE_SOURCE *source*, U32BIT *param*)

Sets the source of the input for the audio description audio on the given audio decoder path.

Parameters

<i>path</i>	audio path to configure
<i>source</i>	the source device to use
<i>param</i>	parameter set with the source, typically the tuner or demux number

4.63.2.33 void STB_AVSetADVolume (U8BIT *path*, U8BIT *vol*)

Sets the volume of the audio description output.

Parameters

<i>path</i>	audio path to be configured
<i>vol</i>	audio volume (0-100%)

4.63.2.34 `BOOLEAN STB_AVSetAudioCodec (U8BIT path, E_STB_AV_AUDIO_CODEC codec)`

Sets the audio codec to be used when decoding audio with the given audio decoder path.

Parameters

<i>path</i>	audio path
<i>codec</i>	codec to be used

Returns

TRUE if the codec is supported and is set correctly, FALSE otherwise

4.63.2.35 `void STB_AVSetAudioDelay (U8BIT path, U16BIT millisecond)`

Sets the audio delay on the given path.

Parameters

<i>path</i>	decoder path
<i>millisecond</i>	audio delay to be applied

4.63.2.36 `void STB_AVSetAudioMute (U8BIT path, BOOLEAN mute)`

Mute or unmute the audio output on the given audio decoder path.

Parameters

<i>path</i>	audio path to be configured
<i>mute</i>	TRUE to mute, FALSE to unmute

4.63.2.37 `void STB_AVSetAudioSource (U8BIT path, E_STB_AV_DECODE_SOURCE source, U32BIT param)`

Sets the source of the input for the main audio on the given audio decoder path.

Parameters

<i>path</i>	audio path to configure
<i>source</i>	the source device to use
<i>param</i>	parameter set with the source, typically the tuner or demux number

4.63.2.38 void STB_AVSetAudioVolume (U8BIT *path*, U8BIT *vol*)

Sets the volume of the main audio output.

Parameters

<i>path</i>	audio path to be configured
<i>vol</i>	audio volume (0-100%)

4.63.2.39 void STB_AVSetAVOutput (BOOLEAN *av_on*)

Turns on/off all AV outputs (e.g. for standby mode)

Parameters

<i>av_on</i>	TRUE to turn on, FALSE to turn off
--------------	------------------------------------

4.63.2.40 void STB_AVSetAVOutputSource (E_STB_AV_OUTPUTS *output*, E_STB_AV_SOURCES *source*, U32BIT *param*)

Routes a specified AV source to a specified AV output.

Parameters

<i>output</i>	The output to be connected
<i>source</i>	The source signal to be connected
<i>param</i>	parameter associated with the source

4.63.2.41 void STB_AVSetCopyProtection (S_STB_AV_COPY_PROTECTION * *copy_protection*)

Apply the specified copy protection. This function is used for CI+.

Parameters

<i>copy_protection</i>	
------------------------	--

4.63.2.42 void STB_AVSetHDMIAudioMode (U8BIT *path*, E_STB_DIGITAL_AUDIO_TYPE *audio_type*)

Sets the HDMI audio output mode, PCM or compressed.

Parameters

<i>path</i>	decoder path
<i>audio_type</i>	PCM or compressed

4.63.2.43 void STB_AVSetHDMIShutdown (BOOLEAN *shutdown*)

Sets the shutdown state of the HDMI output.

Parameters

<i>shutdown</i>	TRUE to put the HDMI in shutdown, FALSE to come out of shutdown
-----------------	---

4.63.2.44 BOOLEAN STB_AVSetIframeCodec (U8BIT *path*, E_STB_AV_VIDEO_CODEC *codec*)

Sets the codec to be used when decoding the next i-frame from memory.

Parameters

<i>path</i>	video path
<i>codec</i>	codec to be used

Returns

TRUE if the codec is supported and is set correctly, FALSE otherwise

4.63.2.45 void STB_AVSetSpdifMode (U8BIT *path*, E_STB_DIGITAL_AUDIO_TYPE *audio_type*)

Sets the SPDIF output mode, PCM or compressed audio.

Parameters

<i>path</i>	decoder path
<i>spdif_type</i>	PCM or compressed

4.63.2.46 void STB_AVSetTVType (U8BIT *path*, E_STB_AV_ASPECT_RATIO *ratio*, E_STB_AV_VIDEO_FORMAT *format*)

Sets the aspect ratio and signal format for the connected television.

Parameters

<i>path</i>	The video path to be set
<i>ratio</i>	The aspect ratio of the tv
<i>format</i>	The signal format of the tv

4.63.2.47 void STB_AVSetUhfModulatorChannel (U8BIT *chan*)

Sets the output channel of the RF Modulator.

Parameters

<i>chan</i>	the UHF channel number
-------------	------------------------

4.63.2.48 void **STB_AVSetVideoCallback** (U8BIT *path*, void(*)(**S_STB_AV_VIDEO_INFO** *, void *) *callback*, void * *user_data*)

Register callback for updated video information.

Parameters

<i>path</i>	decoder path
<i>callback</i>	the callback to call when video information is updated
<i>user_data</i>	user data to pass to the callback

4.63.2.49 **BOOLEAN STB_AVSetVideoCodec** (U8BIT *path*, **E_STB_AV_VIDEO_CODEC** *codec*)

Sets the video codec to be used when decoding video with the given video decoder path.

Parameters

<i>path</i>	video path
<i>codec</i>	codec to be used

Returns

TRUE if the codec is supported and is set correctly, FALSE otherwise

4.63.2.50 void **STB_AVSetVideoSource** (U8BIT *path*, **E_STB_AV_DECODE_SOURCE** *source*, U32BIT *param*)

Sets the source of the input to the given video decoder path.

Parameters

<i>path</i>	video path to configure
<i>source</i>	the source device to use
<i>param</i>	parameter set with the source, typically the tuner or demux number

4.63.2.51 void **STB_AVShowIframe** (U8BIT *path*)

Decodes and displays the previously loaded i-frame data.

Parameters

<i>path</i>	video path to use
-------------	-------------------

4.63.2.52 void STB_AVStartADDecoding (U8BIT *decoder*)

Starts decoding audio description on the given audio path.

Parameters

<i>path</i>	audio decoder path to be started
-------------	----------------------------------

4.63.2.53 void STB_AVStartAudioDecoding (U8BIT *decoder*)

Starts audio decoding on the given audio path.

Parameters

<i>path</i>	audio decoder path to be started
-------------	----------------------------------

4.63.2.54 void STB_AVStartVideoDecoding (U8BIT *path*)

Starts video decoding on the given video path.

Parameters

<i>path</i>	video decoder path to be started
-------------	----------------------------------

4.63.2.55 void STB_AVStopADDecoding (U8BIT *decoder*)

Stops decoding audio description on the given audio path.

Parameters

<i>path</i>	audio decoder path to be stopped
-------------	----------------------------------

4.63.2.56 void STB_AVStopAudioDecoding (U8BIT *decoder*)

Stops audio decoding on the given audio path.

Parameters

<i>path</i>	audio decoder path to be stopped
-------------	----------------------------------

4.63.2.57 void STB_AVStopAudioSample (U8BIT *path*)

Stops playback of an audio sample.

Parameters

<i>path</i>	Audio path on which to stop
-------------	-----------------------------

4.63.2.58 void STB_AVStopVideoDecoding (U8BIT *decoder*)

Stops video decoding on the given video path. The video is not expected to be blanked.

Parameters

<i>path</i>	video decoder path to be stopped
-------------	----------------------------------

4.64 platform/inc/stbhwc.h File Reference

Function prototypes for HW control.

This graph shows which files directly or indirectly include this file:

Typedefs

- typedef enum e_hw_status **E_HW_STATUS**

Enumerations

- enum **E_STB_SP_HANDSHAKING** { HANDSHAKING_NONE, HANDSHAKING_XONXOFF, HANDSHAKING_RTSCTS }
- enum **e_hw_status** { HW_OK = 0, HW_GEN_ERROR, HW_BAD_PARAM, HW_NO_MEMORY }

Functions

- void **STB_SPWrite** (U8BIT *data, U32BIT len, U32BIT timeout, U32BIT *written)
- void **STB_SPRead** (U8BIT *data, U32BIT len, U32BIT timeout, U32BIT *read)
- void **STB_SPDebugWrite** (const char *format,...)
Write debug string to serial/debug port. <CR><LF> characters will be automatically added to the end of the string.
- void **STB_SPDebugNoCnWrite** (const char *format,...)
Writes debug string to the serial port without <CR><LF>
- void **STB_SPDebugAssertFail** (const char *file, int line, const char *eval_str)
Report Assertion failure.
- U8BIT **STB_HWGetTunerPaths** (void)
Returns the number of front end (Tuner) paths on the platform.
- U8BIT **STB_HWGetNumRecorders** (void)
Returns the number of recordings that can take place at the same time.
- U8BIT **STB_HWGetAudioDecodePaths** (void)
Returns the number of audio decoding paths on the platform.
- U8BIT **STB_HWGetVideoDecodePaths** (void)
Returns the number of video decoding paths on the platform.
- U8BIT **STB_HWGetDemuxPaths** (void)

Queries the number of demux paths available.

- U8BIT [STB_HWGetNumCISlots](#) (void)

Returns the serial number of the Set Top Box.

- U8BIT [STB_HWGetNumSCSlots](#) (void)

Returns the number of CI slots on the platform.

- U8BIT * [STB_HWGetOUI](#) (void)

Returns the number of smart card slots on the platform.

- U16BIT [STB_HWGetHwId](#) (void)

Returns the platform hardware identifier code.

- U16BIT [STB_HWGetCustomerId](#) (void)

Returns the platform customer identifier code.

- U32BIT [STB_HWGetBoxSerialNumber](#) (void)

Returns the serial number of the Set Top Box.

- BOOLEAN [STB_GetSqlFileName](#) (U8BIT *pathname, U16BIT max_pathname_len)

Gives the name (optionally including path) of the SQL database file.

- void [STB_HWInitialiseVBI](#) (void)

- void [STB_HWVBIInsert](#) (U8BIT *pes_data_field, U32BIT num_bytes)

4.64.1 Detailed Description

Function prototypes for HW control.

Date

26/09/2000

4.64.2 Function Documentation

4.64.2.1 **BOOLEAN STB_GetSqlFileName** (U8BIT * *pathname*, U16BIT *max_pathname_len*)

Gives the name (optionally including path) of the SQL database file.

Parameters

<i>pathname</i>	- array into which the full path will be written
<i>max_pathname_len</i>	- size of the pathname array

Returns

TRUE if the pathname is returned successfully.

4.64.2.2 U8BIT STB_HWGetAudioDecodePaths (void)

Returns the number of audio decoding paths on the platform.

Returns

The number of audio decode paths

4.64.2.3 U32BIT STB_HWGetBoxSerialNumber (void)

Returns the serial number of the Set Top Box.

Returns

The serial number

4.64.2.4 U16BIT STB_HWGetCustomerId (void)

Returns the platform customer identifier code.

Returns

The platform id

4.64.2.5 U8BIT STB_HWGetDemuxPaths (void)

Queries the number of demux paths available.

Returns

The number of demux paths

4.64.2.6 U16BIT STB_HWGetHwId (void)

Returns the platform hardware identifier code.

Returns

The platform id

4.64.2.7 U8BIT STB_HWGetNumCISlots (void)

Returns the serial number of the Set Top Box.

Returns

The serial number

4.64.2.8 U8BIT STB_HWGetNumRecorders (void)

Returns the number of recordings that can take place at the same time.

Returns

The number of recordings

4.64.2.9 U8BIT STB_HWGetNumSCSlots (void)

Returns the number of CI slots on the platform.

Returns

The number of CI slots

4.64.2.10 U8BIT* STB_HWGetOUI (void)

Returns the number of smart card slots on the platform.

Returns

The number of smart card slots

4.64.2.11 U8BIT STB_HWGetTunerPaths (void)

Returns the number of front end (Tuner) paths on the platform.

Returns

The number of tuner paths

4.64.2.12 U8BIT STB_HWGetVideoDecodePaths (void)

Returns the number of video decoding paths on the platform.

Returns

The number of video decode paths

4.64.2.13 void STB_SPDebugAssertFail (const char * *file*, int *line*, const char * *eval_str*)

Report Assertion failure.

Parameters

<i>file</i>	name of source file
<i>line</i>	line number of source file
<i>eval_str</i>	evaluation string that failed

4.64.2.14 void STB_SPDebugNoCnWrite (const char * *format*, ...)

Writes debug string to the serial port without <CR><LF>

Parameters

<i>format</i>	string & format
---------------	-----------------

4.64.2.15 void STB_SPDebugWrite (const char * *format*, ...)

Write debug string to serial/debug port. <CR><LF> characters will be automatically added to the end of the string.

Parameters

<i>format</i>	string & format
---------------	-----------------

4.65 platform/inc/stbhwc.h File Reference

Header File for DVB-CI low-level Physical layer.

Functions

- void [STB_CIStandbyOn](#) (void)
Puts CI control into standby mode (power off)
- void [STB_CIStandbyOff](#) (void)
Brings CI out of standby mode (power on)
- BOOLEAN [STB_CIRouteTS](#) (U8BIT tuner, U8BIT slot_id, BOOLEAN pass_through)
Sets up routing between a tuner and slot.

4.65.1 Detailed Description

Header File for DVB-CI low-level Physical layer.

Date

11/04/2011

4.65.2 Function Documentation

4.65.2.1 BOOLEAN STB_CIRouteTS (U8BIT *tuner*, U8BIT *slot_id*, BOOLEAN *pass_through*)

Sets up routing between a tuner and slot.

Parameters

<i>tuner</i>	tuner to be routed
<i>slot_id</i>	slot to apply routing to
<i>pass_through</i>	TRUE if the TS should be routed through the slot, FALSE otherwise

Returns

TRUE if successful, FALSE otherwise

4.66 platform/inc/stbhwdmx.h File Reference

Header file - Function prototypes for Demux control.

This graph shows which files directly or indirectly include this file:

Defines

- #define **MAX_HW_SECT_FILT_LEN** 8
- #define **STB_DMX_PID_FILTER_INVALID** 0xffff
- #define **STB_DMX_SECT_FILTER_INVALID** 0xffff
- #define **STB_TPID_CBUFF_PRIORITY** 10

Typedefs

- typedef void(* **FILTER_CALLBACK**)(U8BIT path, U16BIT bytes, U16BIT pfilt_id)

Enumerations

- enum **E_STB_DMX_DESC_TRACK** { **DESC_TRACK_AUDIO**, **DESC_TRACK_VIDEO**, **DESC_TRACK_TEXT**, **DESC_NUM_TRACKS** }
- enum **E_STB_DMX_DESC_KEY_PARITY** { **KEY_PARITY_EVEN**, **KEY_PARITY_ODD** }
- enum **E_STB_DMX_DESC_TYPE** { **DESC_TYPE_DVB**, **DESC_TYPE_AES**, **DESC_TYPE_AES_SCTE_52**, **DESC_TYPE_DES**, **DESC_TYPE_TDES** }
- enum **E_STB_DMX_KEY_USAGE** { **KEY_USAGE_PES**, **KEY_USAGE_TRANSPORT**, **KEY_USAGE_ALL** }
- enum **E_STB_DMX_DEMUX_SOURCE** { **DMX_TUNER**, **DMX_1394**, **DMX_MEMORY** }
- enum **E_STB_DMX_CAPS** { **DMX_CAPS_LIVE** = 0x0001, **DMX_CAPS_PIP** = 0x0002, **DMX_CAPS_RECORDING** = 0x0004, **DMX_CAPS_PLAYBACK** = 0x0008, **DMX_CAPS_MONITOR_SI** = 0x0010 }

Functions

- void [STB_DMXInitialise](#) (U8BIT paths, BOOLEAN inc_pes_collection)
Initialises the demux / programmable transport interface.
- U16BIT [STB_DMXGetCapabilities](#) (U8BIT path)
Returns the capability flags of the given demux.
- U8BIT [STB_DMXGetMaxSectionFilters](#) (void)
Returns the maximum number of section filters available on this hw.
- void [STB_DMXSetDemuxSource](#) (U8BIT path, E_STB_DMX_DEMUX_SOURCE source, U8BIT param)
Configures the source of the demux.
- void [STB_DMXGetDemuxSource](#) (U8BIT path, E_STB_DMX_DEMUX_SOURCE *source, U8BIT *param)
Gets the current source of a given demux.
- void [STB_DMXReadTextPES](#) (U8BIT path, U8BIT **buffer, U32BIT *num_bytes)
Reads Teletext PES data from the demux.
- void [STB_DMXChangeDecodePIDs](#) (U8BIT path, U16BIT pcr_pid, U16BIT video_pid, U16BIT audio_pid, U16BIT text_pid, U16BIT data_pid, U16BIT ad_pid)
Changes the packet IDs for the PCR Video, Audio, Text and Data.
- void [STB_DMXChangeTextPID](#) (U8BIT path, U16BIT text_pid)
Changes just the teletext PID.
- U16BIT [STB_DMXGrabPIDFilter](#) (U8BIT path, U16BIT pid, FILTER_CALLBACK func)
Get a New PID Filter & Setup Associated Buffer and Callback Function Address.
- BOOLEAN [STB_DMXCopyPIDFilterSect](#) (U8BIT path, U8BIT *buffer, U16BIT size, U16BIT pfilt_id)
Copies a filtered section to caller's buffer.
- void [STB_DMXSkipPIDFilterSect](#) (U8BIT path, U16BIT pfilt_id)
Skips (discards) a section in the PID filter buffer.
- U16BIT [STB_DMXGrabSectFilter](#) (U8BIT path, U16BIT pfilt_id)
Allocated a new section filter on the specified PID filter.
- void [STB_DMXSetupSectFilter](#) (U8BIT path, U16BIT sfilt_id, U8BIT *match_ptr, U8BIT *mask_ptr, U8BIT not_equal_byte_index, BOOLEAN crc)
Configures a match and mask for a specified section filter.
- void [STB_DMXStartPIDFilter](#) (U8BIT path, U16BIT pfilt_id)
Start Specified PID Filter Collecting Data.
- void [STB_DMXStopPIDFilter](#) (U8BIT path, U16BIT pfilt_id)
Stop Specified PID Filter Collecting Data.
- void [STB_DMXReleasePIDFilter](#) (U8BIT path, U16BIT pfilt_id)
Releases a previously allocated PID filter.
- void [STB_DMXReleaseSectFilter](#) (U8BIT path, U16BIT sfilt_id)
Releases a previously allocated section filter.

- BOOLEAN [STB_DMXGetDescramblerKey](#) (U8BIT path, E_STB_DMX_DESC_TRACK track)
Acquires a descrambler for the specified track on this path.
- BOOLEAN [STB_DMXFreeDescramblerKey](#) (U8BIT path, E_STB_DMX_DESC_TRACK track)
Frees the descrambler for the specified track on this path.
- BOOLEAN [STB_DMXSetDescramblerKeyData](#) (U8BIT path, E_STB_DMX_DESC_TRACK track, E_STB_DMX_DESC_KEY_PARITY parity, U8BIT *data)
Set the descrambler key data for the specified track on this path.
- BOOLEAN [STB_DMXGetKeyUsage](#) (U8BIT path, E_STB_DMX_DESC_TRACK track, E_STB_DMX_KEY_USAGE *key_usage)
Get the descrambler key usage for the specified track on this path as set by STB_DMXSetKeyUsage.
- BOOLEAN [STB_DMXSetKeyUsage](#) (U8BIT path, E_STB_DMX_DESC_TRACK track, E_STB_DMX_KEY_USAGE key_usage)
Set the descrambler key usage for the specified track on this path.
- BOOLEAN [STB_DMXGetDescramblerType](#) (U8BIT path, E_STB_DMX_DESC_TRACK track, E_STB_DMX_DESC_TYPE *type)
Get the descrambler type for the specified track on this path, as set by STB_DMXSetDescramblerType.
- BOOLEAN [STB_DMXSetDescramblerType](#) (U8BIT path, E_STB_DMX_DESC_TRACK track, E_STB_DMX_DESC_TYPE type)
Set the descrambler type for the specified track on this path.
- void [STB_DMXWriteDemux](#) (U8BIT path, U8BIT *data, U32BIT size)
Writes data to the demux from memory.

4.66.1 Detailed Description

Header file - Function prototypes for Demux control.

Date

06/02/2001

4.66.2 Function Documentation

4.66.2.1 void [STB_DMXChangeDecodePIDs](#) (U8BIT path, U16BIT pcr_pid, U16BIT video_pid, U16BIT audio_pid, U16BIT text_pid, U16BIT data_pid, U16BIT ad_pid)

Changes the packet IDs for the PCR Video, Audio, Text and Data.

Parameters

<i>path</i>	The demux path to be configured
<i>pcr_pid</i>	The PID to use for the Program Clock Reference
<i>video_pid</i>	The PID to use for the Video PES
<i>audio_pid</i>	The PID to use for the Audio PES
<i>text_pid</i>	The PID to use for the Teletext data
<i>data_pid</i>	The PID to use for the data

Generated on Wed Apr 1 2015 11:33:19 for DVBCore by Doxygen

4.66.2.2 void STB_DMXChangeTextPID (U8BIT *path*, U16BIT *text_pid*)

Changes just the teletext PID.

Parameters

<i>path</i>	The demux path to configure
<i>text_pid</i>	The PID to use for the teletext data

4.66.2.3 BOOLEAN STB_DMXCpyPIDFilterSect (U8BIT *path*, U8BIT * *buffer*, U16BIT *size*, U16BIT *pfilt_id*)

Copies a filtered section to caller's buffer.

Parameters

<i>path</i>	the demux path to use
<i>buffer</i>	the caller's buffer
<i>size</i>	the size of the caller's buffer
<i>pfilt_id</i>	the handle of the PID filter to read from

Returns

TRUE copied ok
FALSE no data to copy

4.66.2.4 BOOLEAN STB_DMxFreeDescramblerKey (U8BIT *path*, E_STB_DMX_DESC_TRACK *track*)

Frees the descrambler for the specified track on this path.

Parameters

<i>path</i>	the demux path for which the descrambler is freed
<i>track</i>	enum representing audio, video or subtitles PES

Returns

TRUE on success, FALSE otherwise

4.66.2.5 U16BIT STB_DMXGetCapabilities (U8BIT *path*)

Returns the capability flags of the given demux.

Parameters

<i>path</i>	- demux
-------------	---------

Returns

Capability flags

4.66.2.6 void STB_DMXGetDemuxSource (U8BIT *path*, E_STB_DMX_DEMUX_SOURCE * *source*, U8BIT * *param*)

Gets the current source of a given demux.

Parameters

<i>path</i>	the demux path to query
<i>source</i>	the source of the demux
<i>param</i>	the source specific parameter (e.g. tuner number)

4.66.2.7 BOOLEAN STB_DMXGetDescramblerKey (U8BIT *path*, E_STB_DMX_DESC_TRACK *track*)

Acquires a descrambler for the specified track on this path.

Parameters

<i>path</i>	the demux path for which the descrambler is acquired
<i>track</i>	enum representing audio, video or subtitles PES

Returns

TRUE on success, FALSE otherwise

4.66.2.8 BOOLEAN STB_DMXGetDescramblerType (U8BIT *path*, E_STB_DMX_DESC_TRACK *track*, E_STB_DMX_DESC_TYPE * *type*)

Get the descrambler type for the specified track on this path, as set by STB_DMXSetDescramblerType.

Parameters

<i>path</i>	the demux path that the descrambler type refers to
<i>type</i>	descrambler type (DES, AES, etc...)

Returns

TRUE on success, FALSE otherwise

4.66.2.9 BOOLEAN STB_DMXGetKeyUsage (U8BIT *path*, E_STB_DMX_DESC_TRACK *track*, E_STB_DMX_KEY_USAGE * *key_usage*)

Get the descrambler key usage for the specified track on this path as set by STB_DMXSetKeyUsage.

Parameters

<i>path</i>	the demux path that the descrambler key usage refers to
<i>track</i>	enum representing audio, video or subtitles PES
<i>key_usage</i>	whether the descrambler has been set to operate at PES level, transport level or all.

Returns

TRUE on success, FALSE otherwise

4.66.2.10 U8BIT STB_DMXGetMaxSectionFilters (void)

Returns the maximum number of section filters available on this hw.

Returns

The number of filters

4.66.2.11 U16BIT STB_DMXGrabPIDFilter (U8BIT *path*, U16BIT *pid*, FILTER_CALLBACK *func*)

Get a New PID Filter & Setup Associated Buffer and Callback Function Address.

Parameters

<i>path</i>	Required Decode Path Number.
<i>pid</i>	Required PID to Demux.
<i>func_ptr</i>	User's Interrupt Procedure Function Address.

Returns

New PID filter identifier or invalid id.

4.66.2.12 U16BIT STB_DMXGrabSectFilter (U8BIT *path*, U16BIT *pfilt_id*)

Allocated a new section filter on the specified PID filter.

Parameters

<i>path</i>	the demux path to use
<i>pfilt_id</i>	the PID filter to assign the section filter to

Returns

The section filter handle

4.66.2.13 void STB_DMXInitialise (U8BIT *paths*, BOOLEAN *inc_pes_collection*)

Initialises the demux / programmable transport interface.

Parameters

<i>paths</i>	Number of demux paths to be initialised
<i>inc_pes_ - collection</i>	Not used

4.66.2.14 void **STB_DMReadTextPES** (U8BIT *path*, U8BIT ** *buffer*, U32BIT * *num_bytes*)

Reads Teletext PES data from the demux.

Parameters

<i>path</i>	the demux path to read
<i>buffer</i>	pointer to PES data
<i>num_bytes</i>	the number of bytes of data

4.66.2.15 void **STB_DMReleasePIDFilter** (U8BIT *path*, U16BIT *pfilt_id*)

Releases a previously allocated PID filter.

Parameters

<i>path</i>	the demux path of the filter
<i>pfilt_id</i>	the handle of the filter

4.66.2.16 void **STB_DMReleaseSectFilter** (U8BIT *path*, U16BIT *sfilt_id*)

Releases a previously allocated section filter.

Parameters

<i>path</i>	the demux path of the filter
<i>sfilt_id</i>	the handle of the section filter

4.66.2.17 void **STB_DMSetDemuxSource** (U8BIT *path*, E_STB_DMX_DEMUX_SOURCE *source*, U8BIT *param*)

Configures the source of the demux.

Parameters

<i>path</i>	the demux path to configure
<i>source</i>	the source to use
<i>param</i>	source specific parameters (e.g. tuner number)

4.66.2.18 `BOOLEAN STB_DMXSetDescramblerKeyData (U8BIT path,
E_STB_DMX_DESC_TRACK track, E_STB_DMX_DESC_KEY_PARITY parity, U8BIT *
data)`

Set the descrambler key data for the specified track on this path.

Parameters

<i>path</i>	the demux path for which the descrambler key data is set
<i>track</i>	enum representing audio, video or subtitles PES
<i>parity</i>	even or odd
<i>data</i>	pointer to the key data, its length depends on the descrambler type (see STB_DMXSetDescramblerType)

Returns

TRUE on success, FALSE otherwise

4.66.2.19 `BOOLEAN STB_DMXSetDescramblerType (U8BIT path,
E_STB_DMX_DESC_TRACK track, E_STB_DMX_DESC_TYPE type)`

Set the descrambler type for the specified track on this path.

Parameters

<i>path</i>	the demux path that the descrambler type refers to
<i>type</i>	descrambler type (DES, AES, etc...)

Returns

TRUE on success, FALSE otherwise

4.66.2.20 `BOOLEAN STB_DMXSetKeyUsage (U8BIT path, E_STB_DMX_DESC_TRACK
track, E_STB_DMX_KEY_USAGE key_usage)`

Set the descrambler key usage for the specified track on this path.

Parameters

<i>path</i>	the demux path that the descrambler key usage refers to
<i>track</i>	enum representing audio, video or subtitles PES
<i>key_usage</i>	whether the descrambler operates at PES level, transport level or all.

Returns

TRUE on success, FALSE otherwise

4.66.2.21 void STB_DMXSetupSectFilter (U8BIT *path*, U16BIT *sfilt_id*, U8BIT * *match_ptr*, U8BIT * *mask_ptr*, U8BIT *not_equal_byte_index*, BOOLEAN *crc*)

Configures a match and mask for a specified section filter.

Parameters

<i>path</i>	the demux path of the section filter
<i>sfilt_id</i>	the handle of the section filter
<i>match_ptr</i>	pointer to the match bytes
<i>mask_ptr</i>	pointer to the mask bytes
<i>not_equal_byte_index</i>	the byte position for a not equal compare
<i>crc</i>	TRUE to use CRC checking FALSE to ignore

4.66.2.22 void STB_DMXSkipPIDFilterSect (U8BIT *path*, U16BIT *pfilt_id*)

Skips (discards) a section in the PID filter buffer.

Parameters

<i>path</i>	the demux path of the filter
<i>pfilt_id</i>	the PID filter handle

4.66.2.23 void STB_DMXStartPIDFilter (U8BIT *path*, U16BIT *pfilt_id*)

Start Specified PID Filter Collecting Data.

Parameters

<i>path</i>	Required Decode Path Number.
<i>pfilt_id</i>	Required PID Filter Identifier.

4.66.2.24 void STB_DMXStopPIDFilter (U8BIT *path*, U16BIT *pfilt_id*)

Stop Specified PID Filter Collecting Data.

Parameters

<i>path</i>	Required Decode Path Number.
<i>pfilt_id</i>	Required PID Filter Identifier.

4.66.2.25 void STB_DMXWriteDemux (U8BIT *path*, U8BIT * *data*, U32BIT *size*)

Writes data to the demux from memory.

Parameters

<i>path</i>	the demux path to be written
<i>data</i>	the data to be written
<i>size</i>	the number of bytes to be written

4.67 platform/inc/stbhwdsk.h File Reference

Function prototypes for disk functions.

This graph shows which files directly or indirectly include this file:

Defines

- #define **INVALID_DISK_ID** 0xffff

Enumerations

- enum **E_STB_DSK_FILE_MODE** { **FILE_MODE_READ**, **FILE_MODE_WRITE**, **FILE_MODE_OVERWRITE** }
- enum **E_STB_DSK_FILE_POSITION** { **FILE_POSITION_START**, **FILE_POSITION_END**, **FILE_POSITION_CURRENT** }
- enum **E_STB_DIR_ENTRY_TYPE** { **DIR_ENTRY_FILE**, **DIR_ENTRY_DIRECTORY**, **DIR_ENTRY_OTHER** }

Functions

- void **STB_DSKInitialise** (void)
Initialise the hard disk component.
- U16BIT **STB_DSKGetNumDisks** (void)
Returns the number of disks currently detected.
- U16BIT **STB_DSKGetDiskIdByIndex** (U16BIT index)
Returns the id of the disk at the given index.
- BOOLEAN **STB_DSKIsRemoveable** (U16BIT disk_id)
Checks if the given disk is removeable.
- BOOLEAN **STB_DSKIsMounted** (U16BIT disk_id)
Checks if the given disk is mounted.
- BOOLEAN **STB_DSKMountDisk** (U16BIT disk_id)
Attempts to mount the given disk, if it isn't already mounted.
- BOOLEAN **STB_DSKUnmountDisk** (U16BIT disk_id)
Attempts to unmount the given disk, if it isn't already unmounted.
- BOOLEAN **STB_DSKGetDiskName** (U16BIT disk_id, U8BIT *name, U16BIT name_len)
Gets the name of a disk and copies it into the array provided.
- void **STB_DSKSetStandby** (BOOLEAN state)

Put all disks into or out of standby mode.

- void * [STB_DSKOpenFile](#) (U16BIT disk_id, U8BIT *name, E_STB_DSK_FILE_MODE mode)

Opens an existing file or creates a new one.

- U32BIT [STB_DSKReadFile](#) (void *file, U8BIT *data, U32BIT size)

Reads data from an open file.

- U32BIT [STB_DSKWriteFile](#) (void *file, U8BIT *data, U32BIT size)

Writes data to an open file.

- BOOLEAN [STB_DSKSeekFile](#) (void *file, E_STB_DSK_FILE_POSITION position, S32BIT offset)

Sets the read/write position of an open file.

- BOOLEAN [STB_DSKTellFile](#) (void *file, U32BIT *offset)

Gets the current position in an open file.

- void [STB_DSKCloseFile](#) (void *file)

Flushes and closes an open file.

- BOOLEAN [STB_DSKDeleteFile](#) (U16BIT disk_id, U8BIT *filename)

Deletes the file from the given disk.

- BOOLEAN [STB_DSKFileExists](#) (U16BIT disk_id, U8BIT *filename)

Checks whether a file/directory with the given name exists.

- BOOLEAN [STB_DSKFileSize](#) (U16BIT disk_id, U8BIT *filename, U32BIT *filesize)

Returns the size in KB of the given file.

- void * [STB_DSKOpenDirectory](#) (U16BIT disk_id, U8BIT *dir_name)

Opens a directory in order to read the entries.

- BOOLEAN [STB_DSKReadDirectory](#) (void *dir, U8BIT *filename, U16BIT filename_len, E_STB_DIR_ENTRY_TYPE *entry_type)

Reads the next entry from the directory, returning the name of the entry and the type of the entry.

- void [STB_DSKCloseDirectory](#) (void *dir)

Closes the directory for reading.

- BOOLEAN [STB_DSKCreateDirectory](#) (U16BIT disk_id, U8BIT *dir_path)

Creates a directory with the given name.

- BOOLEAN [STB_DSKDeleteDirectory](#) (U16BIT disk_id, U8BIT *dir_name)

Deletes a directory and all its contents, so use with care!

- BOOLEAN [STB_DSKFullPathname](#) (U16BIT disk_id, U8BIT *filename, U8BIT *pathname, U16BIT max_pathname_len)

Copies the full pathname for the given filename, including the mount directory, to the given string array.

- U32BIT [STB_DSKGetSize](#) (U16BIT disk_id)

Returns the size (capacity) of the disk.

- U32BIT [STB_DSKGetUsed](#) (U16BIT disk_id)

Returns the amount of space used on the disk.

- void [STB_DSKFormat](#) (U16BIT disk_id)

Initiates formatting and partitioning of the hard disk. This will erase all data on the disk!

- U8BIT [STB_DSKGetFormatProgress](#) (U16BIT disk_id)
Gets the progress of the format operation.
- BOOLEAN [STB_DSKIsFormatted](#) (U16BIT disk_id)
Queries whether the disk is formatted.
- void [STB_DSKRepair](#) (U16BIT disk_id)
*Initiates a data repair of the hard disk. This **may** cause the disk to become unreadable.*
- U8BIT [STB_DSKGetRepairProgress](#) (U16BIT disk_id)
Gets the progress of the repair operation.
- BOOLEAN [STB_DSKGetIntegrity](#) (U16BIT disk_id)
Returns a summary of the disk integrity.

4.67.1 Detailed Description

Function prototypes for disk functions.

Date

20/08/2003

4.67.2 Function Documentation

4.67.2.1 void [STB_DSKCloseDirectory](#) (void * *dir*)

Closes the directory for reading.

Parameters

<i>dir</i>	directory handle
------------	------------------

4.67.2.2 void [STB_DSKCloseFile](#) (void * *file*)

Flushes and closes an open file.

Parameters

<i>file</i>	The file handle
-------------	-----------------

4.67.2.3 BOOLEAN [STB_DSKCreateDirectory](#) (U16BIT *disk_id*, U8BIT * *dir_path*)

Creates a directory with the given name.

Parameters

<i>disk_id</i>	disk on which the directory is to be created
<i>dir_name</i>	name of the directory to be created

Returns

TRUE is the directory is successfully created

4.67.2.4 BOOLEAN STB_DSKDeleteDirectory (U16BIT *disk_id*, U8BIT * *dir_name*)

Deletes a directory and all its contents, so use with care!

Parameters

<i>disk_id</i>	disk
<i>dir_name</i>	name of the directory to be deleted

4.67.2.5 BOOLEAN STB_DSKDeleteFile (U16BIT *disk_id*, U8BIT * *filename*)

Deletes the file from the given disk.

Parameters

<i>disk_id</i>	disk ID
<i>filename</i>	pathname on the disk of the file to be deleted

Returns

TRUE if successful, FALSE otherwise

4.67.2.6 BOOLEAN STB_DSKFileExists (U16BIT *disk_id*, U8BIT * *filename*)

Checks whether a file/directory with the given name exists.

Parameters

<i>disk_id</i>	disk ID
<i>filename</i>	pathname on the disk of the file

Returns

TRUE if exists, FALSE otherwise

4.67.2.7 BOOLEAN STB_DSKFileSize (U16BIT *disk_id*, U8BIT * *filename*, U32BIT * *filesize*)

Returns the size in KB of the given file.

Parameters

<i>disk_id</i>	disk on which the file exists
<i>filename</i>	name of the file on disk
<i>filesize</i>	returned value giving the file size in KB

Returns

TRUE if the file exists, FALSE otherwise

4.67.2.8 **BOOLEAN STB_DSKFullPathname** (U16BIT *disk_id*, U8BIT * *filename*, U8BIT * *pathname*, U16BIT *max_pathname_len*)

Copies the full pathname for the given filename, including the mount directory, to the given string array.

Parameters

<i>disk_id</i>	disk
<i>filename</i>	name of the file on the disk
<i>pathname</i>	array into which the full pathname will be copied
<i>max_pathname_len</i>	size of the pathname array

Returns

TRUE if the disk is valid exists and is mounted and the destination string is long enough to take the full pathname

4.67.2.9 **U16BIT STB_DSKGetDiskIdByIndex** (U16BIT *index*)

Returns the id of the disk at the given index.

Parameters

<i>index</i>	zero based index
--------------	------------------

Returns

Disk id, 0 if no disk found

4.67.2.10 **BOOLEAN STB_DSKGetDiskName** (U16BIT *disk_id*, U8BIT * *name*, U16BIT *name_len*)

Gets the name of a disk and copies it into the array provided.

Parameters

<i>disk_id</i>	ID of the disk
<i>name</i>	array of U8BIT into which the name will be copied
<i>name_len</i>	max number of characters in the name

Returns

TRUE if the disk is found, FALSE otherwise

4.67.2.11 U8BIT STB_DSKGetFormatProgress (U16BIT *disk_id*)

Gets the progress of the format operation.

Returns

The progress as percent complete

4.67.2.12 BOOLEAN STB_DSKGetIntegrity (U16BIT *disk_id*)

Returns a summary of the disk integrity.

Returns

TRUE if disk integrity ok

4.67.2.13 U16BIT STB_DSKGetNumDisks (void)

Returns the number of disks currently detected.

Returns

Number of disks

4.67.2.14 U8BIT STB_DSKGetRepairProgress (U16BIT *disk_id*)

Gets the progress of the repair operation.

Returns

The progress as percent complete

4.67.2.15 U32BIT STB_DSKGetSize (U16BIT *disk_id*)

Returns the size (capacity) of the disk.

Returns

The size of the disk in kilobytes

4.67.2.16 U32BIT STB_DSKGetUsed (U16BIT *disk_id*)

Returns the amount of space used on the disk.

Parameters

<i>disk_id</i>	ID of the disk
----------------	----------------

Returns

Space used in kilobytes

4.67.2.17 BOOLEAN STB_DSKIsFormatted (U16BIT *disk_id*)

Queries whether the disk is formatted.

Returns

FALSE unformatted

4.67.2.18 BOOLEAN STB_DSKIsMounted (U16BIT *disk_id*)

Checks if the given disk is mounted.

Parameters

<i>disk_id</i>	ID of the disk to be checked
----------------	------------------------------

Returns

TRUE if the disk is mounted, FALSE otherwise

4.67.2.19 BOOLEAN STB_DSKIsRemoveable (U16BIT *disk_id*)

Checks if the given disk is removeable.

Parameters

<i>disk_id</i>	ID of the disk to be checked
----------------	------------------------------

Returns

TRUE if the disk is removeable, FALSE otherwise

4.67.2.20 BOOLEAN STB_DSKMountDisk (U16BIT *disk_id*)

Attempts to mount the given disk, if it isn't already mounted.

Parameters

<i>disk_id</i>	ID of the disk to be mounted
----------------	------------------------------

Returns

TRUE if the disk is mounted, FALSE otherwise

4.67.2.21 void* STB_DSKOpenDirectory (U16BIT *disk_id*, U8BIT * *dir_name*)

Opens a directory in order to read the entries.

Parameters

<i>disk_id</i>	disk containing to the directory to be read
<i>dir_name</i>	name of the directory to open

Returns

Handle to be used in all other operations, NULL if the open fails

4.67.2.22 void* STB_DSKOpenFile (U16BIT *disk_id*, U8BIT * *name*, E_STB_DSK_FILE_MODE *mode*)

Opens an existing file or creates a new one.

Parameters

<i>name</i>	The filename (including path)
<i>mode</i>	The access mode

Returns

The file handle

4.67.2.23 BOOLEAN STB_DSKReadDirectory (void * *dir*, U8BIT * *filename*, U16BIT *filename_len*, E_STB_DIR_ENTRY_TYPE * *entry_type*)

Reads the next entry from the directory, returning the name of the entry and the type of the entry.

Parameters

<i>dir</i>	handle returned when the directory was opened
<i>filename</i>	array in which the name is returned
<i>filename_len</i>	size of the filename array
<i>entry_type</i>	type of entry

Returns

TRUE if an entry is read, FALSE otherwise which could indicate end of the directory

4.67.2.24 U32BIT STB_DSKReadFile (void * *file*, U8BIT * *data*, U32BIT *size*)

Reads data from an open file.

Parameters

<i>file</i>	The file handle
<i>data</i>	The caller's buffer
<i>size</i>	Number of bytes to be read

Returns

Number of bytes successfully read

4.67.2.25 BOOLEAN STB_DSKSeekFile (void * *file*, E_STB_DSK_FILE_POSITION *position*, S32BIT *offset*)

Sets the read/write position of an open file.

Parameters

<i>file</i>	The file handle
<i>position</i>	Position to move relative to, i.e. start,end,current
<i>offset</i>	Where to move to relative to position

Returns

TRUE success, FALSE otherwise

4.67.2.26 void STB_DSKSetStandby (BOOLEAN *state*)

Put all disks into or out of standby mode.

Parameters

<i>state</i>	standby mode, TRUE=Standby FALSE=On
--------------	-------------------------------------

4.67.2.27 BOOLEAN STB_DSKTellFile (void * *file*, U32BIT * *offset*)

Gets the current position in an open file.

Parameters

<i>file</i>	The file handle
<i>offset</i>	Variable to contain result (byte position in file)

Returns

TRUE on success

4.67.2.28 BOOLEAN STB_DSKUnmountDisk (U16BIT *disk_id*)

Attempts to unmount the given disk, if it isn't already unmounted.

Parameters

<i>disk_id</i>	ID of the disk to be unmounted
----------------	--------------------------------

Returns

TRUE if the disk is unmounted, FALSE otherwise

4.67.2.29 U32BIT STB_DSKWriteFile (void * *file*, U8BIT * *data*, U32BIT *size*)

Writes data to an open file.

Parameters

<i>file</i>	The file handle
<i>data</i>	Pointer to the data to be written
<i>size</i>	Number of bytes to write

Returns

Number of bytes successfully written

4.68 platform/inc/stbhwp.h File Reference

Header file - Function prototypes for front panel control.

Enumerations

- enum **E_FRONT_PANEL_WAKEUP_TYPE** { WAKEUP_UNKNOWN = 0, WAKEUP_TIMER, WAKEUP_USER }
- enum **E_LED_STATE** { LED_OFF = 0, LED_ON, LED_BLINKING }

Functions

- void [STB_FPInitialise](#) (void)
Initialise the Front Panel and Remote Control components.
- void [STB_FPKeyEnable](#) (BOOLEAN enable)
Globally enables the front panel keys.
- void [STB_FPRemoteEnable](#) (BOOLEAN enable)
Globally enables the remote control keys.
- void [STB_FPSetHandsetCode](#) (U16BIT hset_code)
Sets the handset code/id of the remote control.

- void [STB_FPDisplayString](#) (U32BIT str_addr, U8BIT scrollstep, U16BIT scrollrate)
Show a static or scrolling text string on the front panel.
- void [STB_FPSetClock](#) (U16BIT mjd, U16BIT year, U8BIT month, U8BIT day, U8BIT hours, U8BIT minutes, U8BIT seconds)
Set the current date on the front panel.
- void [STB_FPShowClock](#) (BOOLEAN show)
Instructs the front panel to show/hide the time.
- void [STB_FPSetWakeUpTime](#) (U16BIT mjd, U16BIT year, U8BIT month, U8BIT day, U8BIT hours, U8BIT minutes, U32BIT time_in_mins)
Sets the local time when the front panel should come out of standby. The date is passed as mjd and as year, month and day and the number of minutes until the wake up is also passed for an easier implementation. It is implementation specific which parameters to use.
- void [STB_FPSetStandby](#) (BOOLEAN standby)
Controls the box low power standby status.
- E_FRONT_PANEL_WAKEUP_TYPE [STB_FPGetWakeUpType](#) (void)
Returns the wake up type.
- void [STB_FPSetLedState](#) (U8BIT led_id, E_LED_STATE led_state, U16BIT period, U8BIT duty_cycle)
Turns front panel LEDs on or off.
- void [STB_FPWaitForUserInput](#) (void)
- U8BIT [STB_FPGetDisplaySize](#) (void)
- void [STB_FPAnimation](#) (U8BIT anim_type, U8BIT frame_rate)
- void [STB_FPRefreshWatchdog](#) (void)
- void [STB_FPSetWdogPeriod](#) (U16BIT watchdog_period)
- void [STB_FPSetAutoStandby](#) (BOOLEAN auto_standby)
- void [STB_FPSetAutoStandbyTime](#) (U8BIT minute)

4.68.1 Detailed Description

Header file - Function prototypes for front panel control.

Date

06/02/2001

4.68.2 Function Documentation

4.68.2.1 void [STB_FPDisplayString](#) (U32BIT str_addr, U8BIT scrollstep, U16BIT scrollrate)

Show a static or scrolling text string on the front panel.

Parameters

<i>string_addr</i>	address of text string to be displayed
<i>scrollstep</i>	characters to move each step (0=static)
<i>scrollrate</i>	delay between steps (in units of 10mS)

4.68.2.2 E_FRONTANEL_WAKEUP_TYPE STB_FPGetWakeUpType (void)

Returns the wake up type.

Returns

WAKEUP_UNKNOWN when the front panel didn't actively power up the board, i.e. the power has been unplugged and plugged back; WAKEUP_TIMER the front panel is turning the board up after a timer has expired (see STB_FPSetWakeUp-Time) WAKEUP_USER is returned when the front panel is turning the board up because the user pressed the stand-by button.

4.68.2.3 void STB_FPKeyEnable (BOOLEAN enable)

Globally enables the front panel keys.

Parameters

<i>enable</i>	set to TRUE to enable keys, FALSE otherwise
---------------	---

4.68.2.4 void STB_FPRemoteEnable (BOOLEAN enable)

Globally enables the remote control keys.

Parameters

<i>enable</i>	set to TRUE to enable keys, FALSE otherwise
---------------	---

4.68.2.5 void STB_FPSetClock (U16BIT *mjd*, U16BIT *year*, U8BIT *month*, U8BIT *day*, U8BIT *hours*, U8BIT *minutes*, U8BIT *seconds*)

Set the current date on the front panel.

Parameters

<i>mjd</i>	modified Julian day
<i>year</i>	year
<i>month</i>	month
<i>day</i>	day
<i>hours</i>	hours
<i>minutes</i>	minutes
<i>seconds</i>	seconds

4.68.2.6 void STB_FPSetHandsetCode (U16BIT *hset_code*)

Sets the handset code/id of the remote control.

Parameters

<i>hset_code</i>	The 16 bit handset code
------------------	-------------------------

4.68.2.7 void STB_FPSetLedState (U8BIT *led_id*, E_LED_STATE *led_state*, U16BIT *period*, U8BIT *duty_cycle*)

Turns front panel LEDs on or off.

Parameters

<i>led_id</i>	The LED number to control
<i>led_state</i>	TRUE to turn on, FALSE to turn off

4.68.2.8 void STB_FPSetStandby (BOOLEAN *standby*)

Controls the box low power standby status.

Parameters

<i>standby</i>	if TRUE instructs the front panel to put the box in low power standby, if FALSE instructs the front panel to wake the box up from low power stand-by
----------------	--

4.68.2.9 void STB_FPSetWakeUpTime (U16BIT *mjd*, U16BIT *year*, U8BIT *month*, U8BIT *day*, U8BIT *hours*, U8BIT *minutes*, U32BIT *time_in_mins*)

Sets the local time when the front panel should come out of standby. The date is passed as mjd and as year, month and day and the number of minutes until the wake up is also passed for an easier implementation. It is implementation specific which parameters to use.

Parameters

<i>mjd</i>	modified Julian day of the wake up time
<i>year</i>	wake up year
<i>month</i>	wake up month
<i>day</i>	wake up day
<i>hours</i>	wake up hour
<i>minutes</i>	wake up minute
<i>time_in_mins</i>	Number of minutes until wakeup

4.68.2.10 void STB_FPShowClock (BOOLEAN *show*)

Instructs the front panel to show/hide the time.

Parameters

<i>BOOLEAN</i>	show if TRUE, show the time, else hide it.
----------------	--

4.69 platform/inc/stbhwin.h File Reference

Header file - Function prototype for init function.

Typedefs

- typedef enum e_hw_subt_control **E_HW_SUBT_CONTROL_MASK**

Enumerations

- enum **e_hw_subt_control** { **HW_SUBT_NONE** = 0, **HW_SUBT_EBU** = 1, **HW_SUBT_DVB** = 2 }

Functions

- void [STB_HWInitialise](#) (E_HW_SUBT_CONTROL_MASK hw_subt)
Initialise the platform layer by calling the platform layer initialisation functions that are appropriate for this platform.
- void [STB_HWEnterStandby](#) (void)
Perform any operations required before entering standby.
- void [STB_HWLeaveStandby](#) (void)
Perform any operations required when coming out of standby.

4.69.1 Detailed Description

Header file - Function prototype for init function.

Date

20/06/2002

4.69.2 Function Documentation

4.69.2.1 void [STB_HWEnterStandby](#) (void)

Perform any operations required before entering standby.

Parameters

<i>none</i>	
-------------	--

4.69.2.2 void STB_HWInitialise (E_HW_SUBT_CONTROL_MASK *hw_subt*)

Initialise the platform layer by calling the platform layer initialisation functions that are appropriate for this platform.

Parameters

<i>hw_subt</i>	The subtitling mode mask, indicates which types of subtitles are required
----------------	---

4.69.2.3 void STB_HWLeaveStandby (void)

Perform any operations required when coming out of standby.

Parameters

<i>none</i>	
-------------	--

4.70 platform/inc/stbhwp.h File Reference

macros and function prototypes for public use

Functions

- **BOOLEAN STB_IPGetIPAddress** (U8BIT *ip_addr)
Gets the current IPv4 address of the default IP connection.
- **BOOLEAN STB_IPGetSubnetMask** (U8BIT *subnet_mask)
Gets the current IPv4 subnet mask of the default IP connection.
- **BOOLEAN STB_IPGetGatewayIPAddress** (U8BIT *gateway_addr)
Gets the current IPv4 gateway IP address.
- **BOOLEAN STB_IPGetDnsServerIPAddress** (U8BIT *dns_addr)
Gets the current IPv4 DNS server IP address.
- **void STB_IPSetIPAddress** (const U8BIT *ip_addr)
Sets the IPv4 format IP address of default network connection.
- **void STB_IPSetSubnetMask** (const U8BIT *subnet_mask)
Sets the IPv4 format subnet mask of default network connection.
- **void STB_IPSetGatewayIPAddress** (const U8BIT *gateway_addr)
Sets the IPv4 format gateway IP address.
- **void STB_IPSetDnsServerIPAddress** (const U8BIT *dns_addr)
Sets the IPv4 format DNS server IP address.
- **void STB_IPGetIPByDhcp** (BOOLEAN wait_for_completion)
Cause the IP address to be set using DHCP.

4.70.1 Detailed Description

macros and function prototypes for public use

Date

06/08/2009

4.70.2 Function Documentation

4.70.2.1 BOOLEAN STB_IPGetDnsServerIPAddress (U8BIT * *dns_addr*)

Gets the current IPv4 DNS server IP address.

Parameters

<i>dns_addr</i>	4 byte array in which the address will be returned with the most significant byte in <i>dns_addr</i> [0]
-----------------	--

Return values

<i>TRUE</i>	if address is returned, FALSE otherwise
-------------	---

4.70.2.2 BOOLEAN STB_IPGetGatewayIPAddress (U8BIT * *gateway_addr*)

Gets the current IPv4 gateway IP address.

Parameters

<i>gateway_addr</i>	4 byte array in which the address will be returned with the most significant byte in <i>gateway_addr</i> [0]
---------------------	--

Return values

<i>TRUE</i>	if address is returned, FALSE otherwise
-------------	---

4.70.2.3 BOOLEAN STB_IPGetIPAddress (U8BIT * *ip_addr*)

Gets the current IPv4 address of the default IP connection.

Parameters

<i>ip_addr</i>	4 byte array in which the address will be returned with the most significant byte in <i>ip_addr</i> [0]
----------------	---

Return values

<i>TRUE</i>	if address is returned, FALSE otherwise
-------------	---

4.70.2.4 void STB_IPGetIPByDhcp (BOOLEAN *wait_for_completion*)

Cause the IP address to be set using DHCP.

Parameters

<i>wait_for_completion</i>	set as TRUE if the call shouldn't return until an IP address has been assigned or times out
----------------------------	---

4.70.2.5 BOOLEAN STB_IPGetSubnetMask (U8BIT * *subnet_mask*)

Gets the current IPv4 subnet mask of the default IP connection.

Parameters

<i>subnet_mask</i>	4 byte array in which the address will be returned with the most significant byte in subnet_mask[0]
--------------------	---

Return values

<i>TRUE</i>	if address is returned, FALSE otherwise
-------------	---

4.70.2.6 void STB_IPSetDnsServerIPAddress (const U8BIT * *dns_addr*)

Sets the IPv4 format DNS server IP address.

Parameters

<i>dns_addr</i>	4 byte array containing the mask to be set with the most significant byte in dns_addr[0]
-----------------	--

4.70.2.7 void STB_IPSetGatewayIPAddress (const U8BIT * *gateway_addr*)

Sets the IPv4 format gateway IP address.

Parameters

<i>gateway_addr</i>	4 byte array containing the mask to be set with the most significant byte in gateway_addr[0]
---------------------	--

4.70.2.8 void STB_IPSetIPAddress (const U8BIT * *ip_addr*)

Sets the IPv4 format IP address of default network connection.

Parameters

<i>ip_addr</i>	4 byte array containing the IP address to be set with the most significant byte in ip_addr[0]
----------------	---

4.70.2.9 void STB_IPSetSubnetMask (const U8BIT * subnet_mask)

Sets the IPv4 format subnet mask of default network connection.

Parameters

<i>subnet_mask</i>	4 byte array containing the mask to be set with the most significant byte in subnet_mask[0]
--------------------	---

4.71 platform/inc/stbhwmediaplayer.h File Reference

Media player API.

Data Structures

- struct [S_STB_MP_COMPONENT_DETAILS](#)
- struct [S_STB_MP_START_PARAMS](#)

Typedefs

- typedef void(* [STB_MP_CALLBACK](#))(void *handle, E_STB_MP_STATE state)

Enumerations

- enum [E_STB_MP_STATE](#) { [STB_MP_STATE_STOPPED](#) = 0, [STB_MP_STATE_PLAYING](#), [STB_MP_STATE_PAUSED](#), [STB_MP_STATE_CONNECTING](#), [STB_MP_STATE_BUFFERING](#), [STB_MP_STATE_FINISHED](#), [STB_MP_STATE_ERROR](#) }
- enum [E_STB_MP_COMPONENT_TYPE](#) { [STB_MP_COMPONENT_VIDEO](#) = 0, [STB_MP_COMPONENT_AUDIO](#), [STB_MP_COMPONENT_SUBTITLE](#), [STB_MP_COMPONENT_ALL](#) }
- enum [E_STB_MP_ERROR](#) { [STB_MP_NO_ERROR](#) = -1, [STB_MP_FORMAT_NOT_SUPPORTED](#) = 0, [STB_MP_CONNECTION_ERROR](#), [STB_MP_UNDEFINED](#), [STB_MP_NO_RESOURCES](#), [STB_MP_CORRUPT](#), [STB_MP_NOT_AVAILABLE](#), [STB_MP_NOT_AVAILABLE_POSITION](#), [STB_MP_BLOCKED](#) }

Functions

- void * [STB_MPInit](#) (U8BIT *source_url)
Initialises the media player module for the specified source.
- E_HW_STATUS [STB_MPStart](#) (void *handle, [S_STB_MP_START_PARAMS](#) *params)
Starts the presentation of content.
- E_HW_STATUS [STB_MPPause](#) (void *handle)
Pauses the presentation of content.

- E_HW_STATUS [STB_MPResume](#) (void *handle)
Resumes the presentation of content.
- E_HW_STATUS [STB_MPResize](#) (void *handle, [S_RECTANGLE](#) *rect)
Changes position of video on screen.
- E_HW_STATUS [STB_MPStop](#) (void *handle)
Stops the presentation of content.
- void [STB_MPExit](#) (void *handle)
Uninitialises the media player module and invalidate the handle.
- E_HW_STATUS [STB_MPGetTimes](#) (void *handle, U32BIT *begin, U32BIT *current, U32BIT *end)
Returns start, current and end times in milliseconds for the content currently being presented.
- void [STB_MPRegisterCallback](#) (void *handle, STB_MP_CALLBACK callback)
Registers a callback to receive notifications of media player change of state.
- E_HW_STATUS [STB_MPSeek](#) (void *handle, U32BIT position)
Seeks the currently presented content to the specified position.
- E_HW_STATUS [STB_MPObtainComponentList](#) (void *handle, E_STB_MP_COMPONENT_TYPE type, U32BIT *num_ptr, [S_STB_MP_COMPONENT_DETAILS](#) **list_ptr)
Returns a list of components of the specified type available in the currently presented content.
- void [STB_MPReleaseComponentList](#) (void *handle, [S_STB_MP_COMPONENT_DETAILS](#) *list_ptr)
Releases the list of components returned by STB_MPObtainComponentList.
- E_HW_STATUS [STB_MPSelectComponent](#) (void *handle, [S_STB_MP_COMPONENT_DETAILS](#) *component)
Forces the specified component to be presented, if another component of the same type is already presented, it will be removed.
- E_HW_STATUS [STB_MPUnselectComponent](#) (void *handle, [S_STB_MP_COMPONENT_DETAILS](#) *component)
Forces the specified component to be stopped.
- E_STB_MP_ERROR [STB_MPGetError](#) (void *handle)
*Returns the actual value of the error when the media player status is STB_MP_STAT-
E_ERROR.*

4.71.1 Detailed Description

Media player API.

Date

22/03/2013

Author

Sergio Panseri

4.71.2 Function Documentation

4.71.2.1 void **STB_MPExit** (void * *handle*)

Uninitialises the media player module and invalidate the handle.

Parameters

<i>handle</i>	media player handle
---------------	---------------------

4.71.2.2 E_STB_MP_ERROR **STB_MPGetError** (void * *handle*)

Returns the actual value of the error when the media player status is STB_MP_STAT-E_ERROR.

Returns

Error code when the media player is in STB_MP_STATE_ERROR status, STB_MP_NO_ERROR otherwise

4.71.2.3 E_HW_STATUS **STB_MPGetTimes** (void * *handle*, U32BIT * *begin*, U32BIT * *current*, U32BIT * *end*)

Returns start, current and end times in milliseconds for the content currently being presented.

Parameters

<i>handle</i>	media player handle
<i>begin</i>	pointer to the variable where the begin time is stored. The value returned for the begin time depend on the platform implementation and might not be always 0.
<i>current</i>	pointer to the variable where the current time is stored.
<i>end</i>	pointer to the variable where the end time is stored

Returns

HW_OK if successful, error code otherwise

4.71.2.4 void* **STB_MPInit** (U8BIT * *source_url*)

Initialises the media player module for the specified source.

Parameters

<i>source_url</i>	source URL to be presented
-------------------	----------------------------

Returns

media player handle

4.71.2.5 `E_HW_STATUS STB_MPObtainComponentList (void *
handle, E_STB_MP_COMPONENT_TYPE type, U32BIT * num_ptr,
S_STB_MP_COMPONENT_DETAILS ** list_ptr)`

Returns a list of components of the specified type available in the currently presented content.

Parameters

<i>handle</i>	media player handle
<i>type</i>	type of component or STB_MP_COMPONENT_ALL to receive all of them
<i>num_ptr</i>	pointer to the number of components found
<i>list_ptr</i>	pointer to the list, must be freed using STB_MPReleaseComponentList

Returns

HW_OK if successful, error code otherwise

4.71.2.6 `E_HW_STATUS STB_MPPause (void * handle)`

Pauses the presentation of content.

Parameters

<i>handle</i>	media player handle
---------------	---------------------

Returns

HW_OK if successful, error code otherwise

4.71.2.7 `void STB_MPRegisterCallback (void * handle, STB_MP_CALLBACK callback)`

Registers a callback to receive notifications of media player change of state.

Parameters

<i>handle</i>	media player handle
<i>callback</i>	pointer to the callback function

4.71.2.8 `void STB_MPReleaseComponentList (void * handle,
S_STB_MP_COMPONENT_DETAILS * list_ptr)`

Releases the list of components returned by STB_MPObtainComponentList.

Parameters

<i>handle</i>	media player handle
<i>list_ptr</i>	list to be freed

4.71.2.9 E_HW_STATUS STB_MPResize (void * *handle*, S_RECTANGLE * *rect*)

Changes position of video on screen.

Parameters

<i>handle</i>	media player handle
<i>rect</i>	rectangle structure representing the expected position of video

Returns

HW_OK if successful, error code otherwise

4.71.2.10 E_HW_STATUS STB_MPResume (void * *handle*)

Resumes the presentation of content.

Parameters

<i>handle</i>	media player handle
---------------	---------------------

Returns

HW_OK if successful, error code otherwise

4.71.2.11 E_HW_STATUS STB_MPSeek (void * *handle*, U32BIT *position*)

Seeks the currently presented content to the specified position.

Parameters

<i>handle</i>	media player handle
<i>position</i>	position in milliseconds

Returns

HW_OK if successful, error code otherwise

**4.71.2.12 E_HW_STATUS STB_MPSelectComponent (void * *handle*,
S_STB_MP_COMPONENT_DETAILS * *component*)**

Forces the specied component to be presented, if another component of the same type is already presented, it will be removed.

Parameters

<i>handle</i>	media player handle
<i>component</i>	pointer to the component to be presented, this must be one of the elements of the component list returned by STB_MPObtainComponentList.

Returns

HW_OK if successful, error code otherwise

4.71.2.13 E_HW_STATUS STB_MPStart (void * *handle*, S_STB_MP_START_PARAMS * *params*)

Starts the presentation of content.

Parameters

<i>handle</i>	media player handle
<i>params</i>	parameters to control the display

Returns

HW_OK if successful, error code otherwise

4.71.2.14 E_HW_STATUS STB_MPStop (void * *handle*)

Stops the presentation of content.

Parameters

<i>handle</i>	media player handle
---------------	---------------------

Returns

HW_OK if successful, error code otherwise

4.71.2.15 E_HW_STATUS STB_MPUnselectComponent (void * *handle*, S_STB_MP_COMPONENT_DETAILS * *component*)

Forces the specified component to be stopped.

Parameters

<i>handle</i>	media player handle
<i>component</i>	pointer to the component to be presented, this must be one of the elements of the component list returned by STB_MPObtainComponentList.

Returns

HW_OK if successful, error code otherwise

4.72 platform/inc/stbhwmem.h File Reference

Header file - Function prototypes for NVM and Heap.

#include "stbhwdsk.h" Include dependency graph for stbhwmem.h:

Defines

- #define **ELOAD_RESERVED_NVM_OFFSET** 0
- #define **ELOAD_RESERVED_NVM_SIZE_WIDTH** 2
- #define **ELOAD_NVM_DATA_OFFSET** ELOAD_RESERVED_NVM_SIZE_WIDTH

Enumerations

- enum { **SECURE_NVM_DEFAULT_ENCRYPTION_KEY**, **SECURE_NVM_DEFAULT_ENC_INIT_VECTOR**, **SECURE_NVM_CI_PLUS_ROOT_CERT**, **SECURE_NVM_CI_PLUS_BRAND_CERT**, **SECURE_NVM_CI_PLUS_DEVICE_CERT**, **SECURE_NVM_CI_PLUS_DEVICE_KEY**, **SECURE_NVM_CI_PLUS_DH_P**, **SECURE_NVM_CI_PLUS_DH_G**, **SECURE_NVM_CI_PLUS_DH_Q**, **SECURE_NVM_CI_PLUS_PRNG_KEY_K**, **SECURE_NVM_CI_PLUS_PRNG_SEED**, **SECURE_NVM_CI_PLUS_SIV**, **SECURE_NVM_CI_PLUS_SLK**, **SECURE_NVM_CI_PLUS_CLK**, **SECURE_NVM_CI_PLUS_CONTEXTS**, **SECURE_NVM_MHEG5_PRIVATE_KEY**, **SECURE_NVM_MHEG5_PRIVATE_KEY_VERSION**, **SECURE_NVM_SOFTWARE_UPGRADE_PUBLIC_KEY**, **SECURE_NVM_SOFTWARE_UPGRADE_SHARED_KEY** }

Functions

- void **STB_MEMInitialiseRAM** (void)
Initialises the heap.
- void **STB_MEMInitialiseNVM** (void)
Initialises Non-Volatile memory access. For a new NVM device, this function formats it ready for use, otherwise it prepares existing data for access.
- BOOLEAN **STB_MEMReadNVM** (U32BIT addr, U32BIT bytes, U8BIT *dst_addr)
Read data from the NVM.
- BOOLEAN **STB_MEMWriteNVM** (U32BIT addr, U32BIT bytes, U8BIT *src_addr)
Writes data to the NVM.
- void * **STB_MEMReadSecureVariable** (U8BIT key, U32BIT *len)
Read variable defined by the given key from secure NVM. The data must be released using STB_MEMReleaseSecureVariable. This function is used for CI+.

- void [STB_MEMReleaseSecureVariable](#) (void *value)
Releases the data returned by STB_MEMReadSecureVariable. This function is used for CI+.
- BOOLEAN [STB_MEMWriteSecureVariable](#) (U8BIT key, void *value, U32BIT len)
Writes data defined by the given key to secure NVM. This function is used for CI+.
- const void * [STB_MEMReadSecureConstant](#) (U8BIT key, U32BIT *len)
Read constant defined by the given key from secure NVM. The data must not be released (it is managed by the platform layer). This function is used for CI+.
- U32BIT [STB_MEMGetNVMSize](#) (void)
Returns the size (capacity) of the NVM.
- U32BIT [STB_MEMGetNVMOffset](#) (void)
Returns any offset required in NVM to avoid private data.
- U8BIT [STB_MEMGetNVMAalign](#) (void)
Returns the word alignment size of the NVM device.
- void * [STB_MEMGetSysRAM](#) (U32BIT bytes)
Allocates a new block of memory for system use.
- void * [STB_MEMResizeSysRAM](#) (void *ptr, U32BIT new_num_bytes)
Changes the size of the given block of memory ensuring data contained within the original memory block is maintained.
- void [STB_MEMFreeSysRAM](#) (void *block)
Releases a previously allocated block of system memory.
- U8BIT [STB_MEMSysRAMUsed](#) (void)
Returns the amount of available system memory consumed.
- void * [STB_MEMGetAppRAM](#) (U32BIT bytes)
Allocates a new block of memory for application use.
- void * [STB_MEMResizeAppRAM](#) (void *ptr, U32BIT new_num_bytes)
Changes the size of the given block of memory ensuring data contained within the original memory block is maintained.
- void [STB_MEMFreeAppRAM](#) (void *block)
Releases a previously allocated block of system memory.
- U8BIT [STB_MEMAppRAMUsed](#) (void)
Returns the amount of available application memory consumed.
- BOOLEAN [STB_MEMConfigMloaderForUpgrade](#) (void *loader_data, U32BIT data_size)
- BOOLEAN [STB_MEMReadMloaderData](#) (void *buffer, U32BIT size)
- BOOLEAN [STB_MEMWriteMloaderData](#) (void *buffer, U32BIT size)
- BOOLEAN [STB_NVMFileSize](#) (U8BIT *filename, U32BIT *filesize)
- void * [STB_NVMOpenFile](#) (U8BIT *name, E_STB_DSK_FILE_MODE mode)
- void [STB_NVMCloseFile](#) (void *file)
- U32BIT [STB_NVMReadFile](#) (void *file, U8BIT *data, U32BIT size)
- U32BIT [STB_NVMWriteFile](#) (void *file, U8BIT *data, U32BIT size)
- BOOLEAN [STB_NVMDeleteFile](#) (U8BIT *filename)
- void * [STB_NVMOpenDirectory](#) (U8BIT *dir_name)
- BOOLEAN [STB_NVMReadDirectory](#) (void *dir, U8BIT *filename, U16BIT filename_len, E_STB_DIR_ENTRY_TYPE *entry_type)
- void [STB_NVMCloseDirectory](#) (void *dir)

4.72.1 Detailed Description

Header file - Function prototypes for NVM and Heap.

Date

26/04/2002

4.72.2 Function Documentation

4.72.2.1 U8BIT STB_MEMAppRAMUsed (void)

Returns the amount of available application memory consumed.

Returns

Memory used as a percentage of available

4.72.2.2 void STB_MEMFreeAppRAM (void * *block*)

Releases a previously allocated block of system memory.

Parameters

<i>block_ptr</i>	address of block to be released
------------------	---------------------------------

4.72.2.3 void STB_MEMFreeSysRAM (void * *block*)

Releases a previously allocated block of system memory.

Parameters

<i>block_ptr</i>	address of block to be released
------------------	---------------------------------

4.72.2.4 void* STB_MEMGetAppRAM (U32BIT *bytes*)

Allocates a new block of memory for application use.

Parameters

<i>bytes</i>	Size of required block in bytes
--------------	---------------------------------

Returns

A pointer to the allocated block of memory
NULL allocation failed

4.72.2.5 U8BIT STB_MEMGetNVMAAlign (void)

Returns the word alignment size of the NVM device.

Returns

Alignment in bytes

4.72.2.6 U32BIT STB_MEMGetNVMOffset (void)

Returns any offset required in NVM to avoid private data.

Returns

The offset in bytes

4.72.2.7 U32BIT STB_MEMGetNVMSize (void)

Returns the size (capacity) of the NVM.

Returns

Size of the NVM in bytes

4.72.2.8 void* STB_MEMGetSysRAM (U32BIT bytes)

Allocates a new block of memory for system use.

Parameters

<i>bytes</i>	Size of required block in bytes
--------------	---------------------------------

Returns

Pointer to the allocated block of memory or NULL if allocation failed

4.72.2.9 BOOLEAN STB_MEMReadNVM (U32BIT addr, U32BIT bytes, U8BIT * dst_addr)

Read data from the NVM.

Parameters

<i>addr</i>	The NVM start address
<i>bytes</i>	The number of bytes to read
<i>dst_addr</i>	Caller's buffer for the data

Returns

TRUE if successful FALSE otherwise

4.72.2.10 const void* STB_MEMReadSecureConstant (U8BIT key, U32BIT * len)

Read constant defined by the given key from secure NVM. The data must not be released (it is managed by the platform layer). This function is used for CI+.

Parameters

<i>key</i>	data item to be read
<i>len</i>	returned length of data read

Returns

Pointer to data read, or NULL on failure.

4.72.2.11 void* STB_MEMReadSecureVariable (U8BIT *key*, U32BIT * *len*)

Read variable defined by the given key from secure NVM. The data must be released using STB_MEMReleaseSecureVariable. This function is used for CI+.

Parameters

<i>key</i>	data item to be read
<i>len</i>	returned length of data read

Returns

Pointer to data read, or NULL on failure.

4.72.2.12 void STB_MEMReleaseSecureVariable (void * *value*)

Releases the data returned by STB_MEMReadSecureVariable. This function is used for CI+.

Parameters

<i>value</i>	pointer to data to be released
--------------	--------------------------------

4.72.2.13 void* STB_MEMResizeAppRAM (void * *ptr*, U32BIT *new_num_bytes*)

Changes the size of the given block of memory ensuring data contained within the original memory block is maintained.

Parameters

<i>ptr</i>	pointer to memory already allocated from the app heap
<i>new_num_bytes</i>	size of the memory block to be returned

Returns

Address of new block of memory

4.72.2.14 void* STB_MEMResizeSysRAM (void * *ptr*, U32BIT *new_num_bytes*)

Changes the size of the given block of memory ensuring data contained within the original memory block is maintained.

Parameters

<i>ptr</i>	pointer to memory already allocated from the system heap
<i>new_num_bytes</i>	size of the memory block to be returned

Returns

Address of new block of memory

4.72.2.15 U8BIT STB_MEMSysRAMUsed (void)

Returns the amount of available system memory consumed.

Returns

Memory used as a percentage of available

4.72.2.16 BOOLEAN STB_MEMWriteNVM (U32BIT *addr*, U32BIT *bytes*, U8BIT * *src_addr*)

Writes data to the NVM.

Parameters

<i>addr</i>	The NVM start address to write to
<i>bytes</i>	The number of bytes to be written
<i>src_addr</i>	Pointer to the data to be written

Returns

TRUE if successful, FALSE otherwise

4.72.2.17 BOOLEAN STB_MEMWriteSecureVariable (U8BIT *key*, void * *value*, U32BIT *len*)

Writes data defined by the given key to secure NVM. This function is used for CI+.

Parameters

<i>key</i>	data item to be written
<i>value</i>	pointer to data to be written
<i>len</i>	data size

Returns

TRUE if the data is written successfully, FALSE otherwise

4.73 platform/inc/stbhwnet.h File Reference

Socket functions.

Data Structures

- struct [S_NW_SOCKET](#)
- struct [S_NW_ADDR_INFO](#)
- struct [S_NW_ACCESS_POINT](#)

Defines

- #define **MAX_ADDR_STRING_LEN** 255
- #define **SOCK_SETSIZE** 64
- #define **SOCK_CLR**(sock, set) [STB_NWSockClear](#)((sock), (set));
- #define **SOCK_SET**(sock, set) [STB_NWSockSet](#)((sock), (set));
- #define **SOCK_ZERO**(set) [STB_NWSockZero](#)((set))
- #define **SOCK_ISSET**(sock, set) [STB_NWSockIsSet](#)((sock), (set))
- #define **INVALID_SOCKET** 0

Typedefs

- typedef void(* **NW_eth_callback**)(void)
- typedef void * **NW_handle**

Enumerations

- enum **E_NW_INTERFACE** { **NW_WIRED**, **NW_WIRELESS** }
- enum **E_NW_AF** { **NW_AF_INET**, **NW_AF_INET6** }
- enum **E_NW_TYPE** { **NW_SOCKET_STREAM**, **NW_SOCKET_DGRAM** }
- enum **E_NW_PROTOCOL** { **NW_PROTOCOL_TCP**, **NW_PROTOCOL_UDP** }
- enum **E_NW_LINK_STATUS** { **NW_LINK_ACTIVE**, **NW_LINK_INACTIVE**, **NW_LINK_DISABLED** }
- enum **E_NW_ERROR** { **NW_OK**, **NW_ERROR**, **NW_BAD_PARAM**, **NW_PENDING** }

Functions

- **BOOLEAN** [STB_NWInitialise](#) (void)
Initialises the socket API, must be called once before using API.
- **BOOLEAN** [STB_NWSelectInterface](#) (E_NW_INTERFACE iface)
Sets the network interface that will be used for all network or IP operations.
- **E_NW_INTERFACE** [STB_NWGetSelectedInterface](#) (void)
Returns the selected interface.
- **E_NW_LINK_STATUS** [STB_NWGetLinkStatus](#) (void)
Get the selected interface link status.
- **NW_handle** [STB_NWStartEthernetMonitor](#) (NW_eth_callback func)
Start monitoring the ethernet status.
- **void** [STB_NWStopEthernetMonitor](#) (NW_handle hdl)
Stop monitoring the ethernet status.
- **BOOLEAN** [STB_NWGetMACAddress](#) (E_NW_INTERFACE iface, U8BIT *mac_addr)
Gets the MAC address of the default ethernet connection.
- **U16BIT** [STB_NWLookupAddress](#) (U8BIT *name, [S_NW_ADDR_INFO](#) **nw_addrs)
Performs a lookup to find the IP address(es) of the given host name.
- **void *** [STB_NWOpenSocket](#) (E_NW_AF af, E_NW_TYPE type, E_NW_PROTOCOL protocol, **BOOLEAN** nonblock)
Opens (creates) a new socket for subsequent use.
- **BOOLEAN** [STB_NWCloseSocket](#) (void *socket)
Closes (destroys) a socket instance.
- **BOOLEAN** [STB_NWBind](#) (void *socket, U8BIT *address, U32BIT port)
Binds (names) a socket in the local address space.
- **BOOLEAN** [STB_NWSetReuseaddr](#) (void *socket, **BOOLEAN** state)
Sets the socket option SO_REUSEADDR.
- **BOOLEAN** [STB_NWGetReuseaddr](#) (void *socket, **BOOLEAN** *state)
Gets the socket option SO_REUSEADDR.
- **BOOLEAN** [STB_NWAddMembership](#) (void *socket, U8BIT *group_address)
Sets the ip option IP_ADD_MEMBERSHIP.
- **BOOLEAN** [STB_NWDropMembership](#) (void *socket, U8BIT *group_address)
Sets the ip option IP_DROP_MEMBERSHIP.
- **BOOLEAN** [STB_NWGetSocketName](#) (void *socket, E_NW_AF *af, U8BIT *address, U32BIT *port)
Gets the socket's "name" (family, address and port)
- **E_NW_ERROR** [STB_NWConnect](#) (void *socket, U8BIT *address, U32BIT port)
Connects the socket to a remote host.
- **BOOLEAN** [STB_NWListen](#) (void *socket, S32BIT backlog)
Put socket into state of waiting for incoming connection.
- **void *** [STB_NWAccept](#) (void *socket, U8BIT *address, U32BIT *port)

- Accepts an incoming connection attempt on a socket.*
- S32BIT [STB_NWSend](#) (void *socket, U8BIT *buf, U32BIT num_bytes)
 - Sends data on a connected socket.*
- S32BIT [STB_NWReceive](#) (void *socket, U8BIT *buf, U32BIT max_bytes)
 - Receives data from a connected socket.*
- S32BIT [STB_NWReceiveFrom](#) (void *socket, U8BIT *buf, U32BIT max_bytes, U8BIT *address, U32BIT *port)
 - Receives a datagram and returns the data and its source address.*
- S32BIT [STB_NWSendTo](#) (void *socket, U8BIT *buf, U32BIT num_bytes, U8BIT *address, U32BIT port)
 - Sends data to a specific destination.*
- BOOLEAN [STB_NWSockIsSet](#) (void *socket, [S_NW_SOCKETSET](#) *socks)
 - Returns whether a specified socket is a member of a specified set.*
- void [STB_NWSockZero](#) ([S_NW_SOCKETSET](#) *socks)
 - Clears the socket set.*
- void [STB_NWSockClear](#) (void *socket, [S_NW_SOCKETSET](#) *socks)
 - Clears the specified socket from the specified set.*
- void [STB_NWSockSet](#) (void *socket, [S_NW_SOCKETSET](#) *socks)
 - Sets a specified socket in a specified set.*
- S32BIT [STB_NWSelect](#) ([S_NW_SOCKETSET](#) *read_sockets, [S_NW_SOCKETSET](#) *write_sockets, [S_NW_SOCKETSET](#) *except_sockets, S32BIT timeout_ms)
 - Determines the status of one or more sockets, blocking if necessary.*
- U16BIT [STB_NWGetWirelessAccessPoints](#) ([S_NW_ACCESS_POINT](#) **access_points)
 - Scans and returns all access points visible on the wireless network.*
- void [STB_NWFreeWirelessAccessPoints](#) ([S_NW_ACCESS_POINT](#) *access_points, U16BIT num_aps)
 - Frees the array of access points returned by STB_NWGetWirelessAccessPoints.*
- BOOLEAN [STB_NWConnectToAccessPoint](#) (U8BIT *ssid, U8BIT *password)
 - Attempts to connect to the wireless network with the given SSID. If the network is open then 'password' can be NULL.*

4.73.1 Detailed Description

Socket functions.

Date

24/06/2009

Author

Sergio Panseri

4.73.2 Function Documentation

4.73.2.1 void* STB_NWAccept (void * *socket*, U8BIT * *address*, U32BIT * *port*)

Accepts an incoming connection attempt on a socket.

Parameters

<i>void*</i>	socket the handle of the socket in the listening state
<i>U8BIT*</i>	address pointer to the returned address (string) of connecting entity
<i>U32BIT*</i>	port pointer to the returned port of connecting entity

Returns

handle of new socket actually connected, NULL if failed

4.73.2.2 BOOLEAN STB_NWAddMembership (void * *socket*, U8BIT * *group_address*)

Sets the ip option IP_ADD_MEMBERSHIP.

Parameters

<i>void*</i>	socket the handle of the socket
<i>U8BIT</i>	*group_address address to add

Returns

TRUE successfully set, FALSE failed to set

4.73.2.3 BOOLEAN STB_NWBind (void * *socket*, U8BIT * *address*, U32BIT *port*)

Binds (names) a socket in the local address space.

Parameters

<i>socket</i>	the handle of the socket to be bound
<i>address</i>	the string address (NULL to let OS choose)
<i>port</i>	the port number to be bound to (zero to let OS choose)

Returns

TRUE successfully bound, FALSE failed to bind

4.73.2.4 BOOLEAN STB_NWCloseSocket (void * *socket*)

Closes (destroys) a socket instance.

Parameters

<i>socket</i>	handle of the socket to be closed
---------------	-----------------------------------

Returns

TRUE success, FALSE error

4.73.2.5 E_NW_ERROR STB_NWConnect (void * *socket*, U8BIT * *address*, U32BIT *port*)

Connects the socket to a remote host.

Parameters

<i>void*</i>	socket the handle of the socket
<i>U8BIT*</i>	address address string
<i>U32BIT</i>	port port

Returns

TRUE successfully connected, FALSE failed to set

4.73.2.6 BOOLEAN STB_NWConnectToAccessPoint (U8BIT * *essid*, U8BIT * *password*)

Attempts to connect to the wireless network with the given SSID. If the network is open then 'password' can be NULL.

Parameters

<i>essid</i>	network to connect to
<i>password</i>	password to be used for an encrypted network, can be NULL for an open network

Returns

TRUE if connected successfully to the network, FALSE otherwise

4.73.2.7 BOOLEAN STB_NWDropMembership (void * *socket*, U8BIT * *group_address*)

Sets the ip option IP_DROP_MEMBERSHIP.

Parameters

<i>void*</i>	socket the handle of the socket
<i>U8BIT</i>	*group_address address to drop

Returns

TRUE successfully set, FALSE failed to set

4.73.2.8 void **STB_NWFreeWirelessAccessPoints** (**S_NW_ACCESS_POINT** *
access_points, **U16BIT** *num_aps*)

Frees the array of access points returned by STB_NWGetWirelessAccessPoints.

Parameters

<i>access_points</i>	array of access points to be freed
<i>num_aps</i>	number of access points in the array

4.73.2.9 **E_NW_LINK_STATUS** **STB_NWGetLinkStatus** (void)

Get the selected interface link status.

Returns

NW_LINK_ACTIVE cable connected, NW_LINK_INACTIVE cable dis-connected -
NW_LINK_DISABLED no ethernet device or other error

4.73.2.10 **BOOLEAN** **STB_NWGetMACAddress** (**E_NW_INTERFACE** *iface*, **U8BIT** *
mac_addr)

Gets the MAC address of the default ethernet connection.

Parameters

<i>interface</i>	NW_WIRED or NW_WIRELESS
<i>mac_address</i>	6 byte array in which the address will be returned with the most significant byte in <i>mac_addr[0]</i>

Return values

<i>TRUE</i>	if address is returned, FALSE otherwise
-------------	---

4.73.2.11 **BOOLEAN** **STB_NWGetReuseaddr** (void * *socket*, **BOOLEAN** * *state*)

Gets the socket option SO_REUSEADDR.

Parameters

<i>void</i>	*socket the handle of the socket
<i>BOOLEAN</i>	*state pointer to the returned state, TRUE or FALSE if the option is active/inactive

Returns

TRUE successfully got, FALSE failed to get

4.73.2.12 **E_NW_INTERFACE STB_NWGetSelectedInterface (void)**

Returns the selected interface.

Returns

Selected interface

4.73.2.13 **BOOLEAN STB_NWGetSocketName (void * socket, E_NW_AF * af, U8BIT * address, U32BIT * port)**

Gets the socket's "name" (family, address and port)

Parameters

<i>void*</i>	socket the handle of the socket
<i>E_NW_AF*</i>	af pointer to the returned family
<i>U8BIT*</i>	address pointer to the returned address string
<i>U32BIT*</i>	port pointer to the returned port

Returns

TRUE successfully set, FALSE failed to set

4.73.2.14 **U16BIT STB_NWGetWirelessAccessPoints (S_NW_ACCESS_POINT ** access_points)**

Scans and returns all access points visible on the wireless network.

Parameters

<i>access_ - points</i>	pointer to an array that will be allocated containing info on each access point found
-------------------------	---

Returns

number of access points found

4.73.2.15 **BOOLEAN STB_NWInitialise (void)**

Initialises the socket API, must be called once before using API.

Returns

TRUE Socket API initialised ok, FALSE error

4.73.2.16 `BOOLEAN STB_NWListen (void * socket, S32BIT backlog)`

Put socket into state of waiting for incoming connection.

Parameters

<i>void*</i>	socket the handle of the socket to begin listening
<i>backlog</i>	the maximum length of the queue of pending connections

Returns

TRUE successfully connected, FALSE failed to set

4.73.2.17 `U16BIT STB_NWLookupAddress (U8BIT * name, S_NW_ADDR_INFO ** nw_addrs)`

Performs a lookup to find the IP address(es) of the given host name.

Parameters

<i>name</i>	host name
<i>nw_addrs</i>	array of structures containing the results of the lookup. This array must be deleted using STB_MEMFreeSysRAM.

Returns

The number of addresses returned in the array

4.73.2.18 `void* STB_NWOpenSocket (E_NW_AF af, E_NW_TYPE type, E_NW_PROTOCOL protocol, BOOLEAN nonblock)`

Opens (creates) a new socket for subsequent use.

Parameters

<i>af</i>	the address family that the socket will be used with
<i>type</i>	the stream type the socket will be used with
<i>protocol</i>	the protocol the socket will be used with

Returns

The socket handle, or NULL if failed

4.73.2.19 `S32BIT STB_NWReceive (void * socket, U8BIT * buf, U32BIT max_bytes)`

Receives data from a connected socket.

Parameters

<i>void*</i>	socket the handle of the socket from which to read
<i>U8BIT*</i>	buf the buffer to hold the read data
<i>U32BIT</i>	max_bytes the maximum bytes the caller can accept in the buffer

Returns

the number of bytes read, -1 if there was an error reading 0 if the connection was closed

4.73.2.20 S32BIT STB_NWReceiveFrom (void * socket, U8BIT * buf, U32BIT max_bytes, U8BIT * address, U32BIT * port)

Receives a datagram and returns the data and its source address.

Parameters

<i>void*</i>	socket the handle of the socket from which to read
<i>U8BIT*</i>	buf the buffer to hold the read data
<i>U32BIT</i>	max_bytes the maximum bytes the caller can accept in the buffer
<i>U8BIT*</i>	address the source address of the data in string form
<i>U32BIT*</i>	port the source port of the data

Returns

the number of bytes read, -1 if there was an error reading 0 if the connection was closed

4.73.2.21 S32BIT STB_NWSelect (S_NW_SOCKET * read_sockets, S_NW_SOCKET * write_sockets, S_NW_SOCKET * except_sockets, S32BIT timeout_ms)

Determines the status of one or more sockets, blocking if necessary.

Parameters

<i>S_NW_SOCKET</i>	*read_sockets set of sockets to be checked for readability
<i>S_NW_SOCKET</i>	*write_sockets set of sockets to be checked for writability
<i>S_NW_SOCKET</i>	*exceptfds set of sockets to be checked for errors
<i>U32BIT</i>	timeout_ms maximum number of milliseconds to wait (-1 to block, 0 to return immediately)

Returns

the total number of sockets that are ready, 0 time out exceeded, -1 an error occurred

4.73.2.22 **BOOLEAN** STB_NWSelectInterface (**E_NW_INTERFACE** *iface*)

Sets the network interface that will be used for all network or IP operations.

Parameters

<i>interface</i>	network interface type to use
------------------	-------------------------------

Return values

<i>TRUE</i>	if interface is available, <i>FALSE</i> otherwise
-------------	---

4.73.2.23 **S32BIT** STB_NWSend (**void *** *socket*, **U8BIT *** *buf*, **U32BIT** *num_bytes*)

Sends data on a connected socket.

Parameters

<i>void*</i>	socket the handle of the (connected) socket on which to send
<i>U8BIT*</i>	buf the buffer holding the data to be sent
<i>U32BIT</i>	num_bytes the number of bytes in buf to be sent

Returns

the number of bytes sent (may be less than num_bytes)' -1 if failed

4.73.2.24 **S32BIT** STB_NWSendTo (**void *** *socket*, **U8BIT *** *buf*, **U32BIT** *num_bytes*, **U8BIT *** *address*, **U32BIT** *port*)

Sends data to a specific destination.

Parameters

<i>void*</i>	socket the handle of the socket on which to send
<i>U8BIT*</i>	buf the buffer holding the data to be sent
<i>U32BIT</i>	num_bytes the number of bytes in buf to be sent
<i>U8BIT*</i>	address the address (in string form) of the target socket
<i>U32BIT</i>	port the port number of the target socket

Returns

the number of bytes sent (may be less than num_bytes), -1 if failed

4.73.2.25 **BOOLEAN** STB_NWSetReuseaddr (**void *** *socket*, **BOOLEAN** *state*)

Sets the socket option SO_REUSEADDR.

Parameters

<i>void</i>	*socket the handle of the socket to be bound
<i>BOOLEAN</i>	state TRUE or FALSE to activate/deactivate the option REUSEADDR

Returns

TRUE successfully set, FALSE failed to set

4.73.2.26 void STB_NWSockClear (void * *socket*, S_NW_SOCKETSET * *socks*)

Clears the specified socket from the specified set.

Parameters

<i>void*</i>	socket The socket to clear
S_NW_SOCKETSET	*socks The set of sockets

4.73.2.27 BOOLEAN STB_NWSockIsSet (void * *socket*, S_NW_SOCKETSET * *socks*)

Returns whether a specified socket is a member of a specified set.

Parameters

<i>void*</i>	socket The socket to check
S_NW_SOCKETSET	*socks The set of sockets to check

Returns

TRUE socket is a member of set, FALSE socket is not a member of set

4.73.2.28 void STB_NWSockSet (void * *socket*, S_NW_SOCKETSET * *socks*)

Sets a specified socket in a specified set.

Parameters

<i>void*</i>	socket The socket to set
S_NW_SOCKETSET	*socks The set of sockets

4.73.2.29 void STB_NWSockZero (S_NW_SOCKETSET * *socks*)

Clears the socket set.

Parameters

S_NW_SOCKETSET	*socks The set of sockets to check
--------------------------------	------------------------------------

4.73.2.30 NW_handle STB_NWStartEthernetMonitor (NW_eth_callback *func*)

Start monitoring the ethernet status.

Parameters

<i>func</i>	callback function to notify status change to ethernet device
-------------	--

Returns

handle

4.74 platform/inc/stbhvos.h File Reference

Header file - Function prototypes for operating system.

Defines

- #define HW_EV_CLASS_HANDSET 0
- #define HW_EV_CLASS_KEYPAD 1
- #define HW_EV_CLASS_LNB 2
- #define HW_EV_CLASS_TUNER 3
- #define HW_EV_CLASS_DECODE 4
- #define HW_EV_CLASS_SCART 5
- #define HW_EV_CLASS_DISK 6
- #define HW_EV_CLASS_DVD 7
- #define HW_EV_CLASS_PVR 8
- #define HW_EV_CLASS_USB 9
- #define HW_EV_CLASS_HDMI 10
- #define HW_EV_CLASS_CEC 11
- #define HW_EV_CLASS_PRIVATE 255 /* Class to allow events to be created that are unique to a platform */
- #define HW_EV_TYPE_FALSE 0
- #define HW_EV_TYPE_TRUE 1
- #define HW_EV_TYPE_LOCKED 2
- #define HW_EV_TYPE_NOTLOCKED 3
- #define HW_EV_TYPE_AUDIO_STARTED 4
- #define HW_EV_TYPE_VIDEO_STARTED 5
- #define HW_EV_TYPE_AUDIO_STOPPED 6
- #define HW_EV_TYPE_VIDEO_STOPPED 7
- #define HW_EV_TYPE_4_3 8
- #define HW_EV_TYPE_16_9 9

- #define **HW_EV_TYPE_SIGNAL_DATA_BAD** 10
- #define **HW_EV_TYPE_SIGNAL_DATA_OK** 11
- #define **HW_EV_TYPE_FORMAT_COMPLETE** 12
- #define **HW_EV_TYPE_FORMAT_FAILED** 13
- #define **HW_EV_TYPE_REPAIR_COMPLETE** 14
- #define **HW_EV_TYPE_REPAIR_FAILED** 15
- #define **HW_EV_TYPE_DVD_DISK_INSERTED** 16
- #define **HW_EV_TYPE_DVD_DISK_REMOVED** 17
- #define **HW_EV_TYPE_PVR_REC_START** 18
- #define **HW_EV_TYPE_PVR_REC_STOP** 19
- #define **HW_EV_TYPE_PVR_PLAY_START** 20
- #define **HW_EV_TYPE_PVR_PLAY_STOP** 21
- #define **HW_EV_TYPE_PVR_PLAY_BOF** 22 /* Playback has reached the beginning of the file */
- #define **HW_EV_TYPE_PVR_PLAY_EOF** 23 /* Playback has reached the end of the file */
- #define **HW_EV_TYPE_PVR_PLAY_NOTIFY_TIME** 24 /* Playback has reached the end of the file */
- #define **HW_EV_TYPE_SAMPLE_STOPPED** 25
- #define **HW_EV_TYPE_DISK_CONNECTED** 26
- #define **HW_EV_TYPE_DISK_REMOVED** 27
- #define **HW_EV_TYPE_DISK_FULL** 28
- #define **HW_EV_TYPE_HDMI_CONNECT** 29
- #define **HW_EV_TYPE_HDMI_DISCONNECT** 30
- #define **HW_EV_TYPE_CEC_PLAY** 31
- #define **HW_EV_TYPE_CEC_STANDBY** 32
- #define **HW_EV_TYPE_AD_STARTED** 34
- #define **HW_EV_TYPE_AD_STOPPED** 35
- #define **HW_EV_TYPE_VIDEO_UNDERFLOW** 36
- #define **HW_EV_TYPE_AUDIO_UNDERFLOW** 37
- #define **TIMEOUT_NOW** 0
- #define **TIMEOUT_NEVER** 0xffff

Functions

- void [STB_OSInitialise](#) (void)
Allows setting of initial boot time.
- void [STB_OSRegisterCallback](#) (void(*func)(BOOLEAN, U16BIT, U16BIT, void *, U32BIT))
Register the function that will be called when STB_OSSendEvent is used.
- void [STB_OSSendEvent](#) (BOOLEAN repeat, U16BIT event_class, U16BIT event_type, void *data, U32BIT data_size)
Send an event by calling the registered callback function.
- void [STB_OSSetClockRTC](#) (U32BIT num_seconds)
Set the local time in seconds since midnight 1-1-1970.
- U32BIT [STB_OSGetClockRTC](#) (void)

Returns the current time in seconds. This is calculated by using the set UTC time and adding the difference between the system boot time when it was set (sync_time) and the system boot time now.

- void [STB_OSClockGMT](#) (U32BIT num_seconds)
Set the time in seconds since midnight 1-1-1970 in GMT.
- U32BIT [STB_OSGetClockGMT](#) (void)
Returns the system time in seconds.
- U32BIT [STB_OSGetClockMilliseconds](#) (void)
Get Current Computer Clock Time.
- U32BIT [STB_OSGetClockDiff](#) (U32BIT timestamp)
Get Difference between Given Time and Current Time.
- void * [STB_OSCreateTask](#) (void(*function)(void *), void *param, U32BIT stack, U8BIT priority, U8BIT *name)
Create a New Task to the calling process. Upon success, the created task runs on its own stack, as part of the calling process context. Task termination must be achieved by calling [STB_OSDestroyTask\(\)](#)
- void [STB_OSTaskSuspend](#) (void)
Suspend the calling task.
- void [STB_OSTaskDelay](#) (U16BIT timeout)
Delay Task for Specified Time Period.
- void [STB_OSTaskSleep](#) (void)
Put Calling Task to Sleep. This is the equivalent of a Task Reschedule which is useful when a task wants to release the CPU on an application preemption point.
- void [STB_OSTaskWakeUp](#) (void)
Wake up a task that was put to sleep when it called [STB_OSTaskSleep\(\)](#).
- void [STB_OSDestroyTask](#) (void *task)
Delete Task must be called upon termination of each task as it frees all OS specific resources allocated for the specific task.
- U8BIT [STB_OSTaskPriority](#) (void *task, U8BIT priority)
Set a New Priority Level for Specified Task.
- void [STB_OSTaskLock](#) (void)
Lock the calling task. Prevents task scheduler from preempting calling task and should always be used as a pair with [STB_OSTaskUnlock\(\)](#) to create a critical region of code execution that cannot be interrupted by another task.
- void [STB_OSTaskUnlock](#) (void)
Unlock the calling task. Allows task scheduler to preempting calling task and should always be used as a pair with [STB_OSTaskLock\(\)](#) to create a critical region of code execution that cannot be interrupted by another task.
- void * [STB_OSGetCurrentTask](#) (void)
Returns the handle of the current task.
- void * [STB_OSCreateQueue](#) (U16BIT msg_size, U16BIT msg_max)
Create Queue of given number of messages and size of message.
- BOOLEAN [STB_OSReadQueue](#) (void *queue, void *msg, U16BIT msg_size, - U16BIT timeout)
Read a message from a queue.

- BOOLEAN [STB_OSWriteQueue](#) (void *queue, void *msg, U16BIT msg_size, - U16BIT timeout)
Write a message to the queue.
- BOOLEAN [STB_OSDestroyQueue](#) (void *queue)
Destroy Queue.
- void * [STB_OSCreateSemaphore](#) (void)
Create a Semaphore.
- void * [STB_OSCreateCountSemaphore](#) (U32BIT value)
Create a counting semaphore.
- void [STB_OSInitCountSemaphore](#) (void *semaphore, U32BIT value)
Initialise a counting semaphore.
- void [STB_OSDeleteSemaphore](#) (void *semaphore)
Delete a Semaphore.
- void [STB_OSSemaphoreWait](#) (void *semaphore)
Wait on Semaphore Indefinitely or Until Released.
- void [STB_OSSemaphoreSignal](#) (void *semaphore)
Signal a Semaphore to Release it by decrementing its counter.
- BOOLEAN [STB_OSSemaphoreWaitTimeout](#) (void *semaphore, U16BIT timeout)
Wait on Semaphore for Set Time Period in an Attempt to Acquire.
- void * [STB_OSCreateMutex](#) (void)
Create a mutex.
- void [STB_OSMutexLock](#) (void *mutex)
Lock a mutex (a.k.a. 'enter', 'wait' or 'get').
- void [STB_OSMutexUnlock](#) (void *mutex)
Unlock a mutex (a.k.a. 'leave', 'signal' or 'release').
- void [STB_OSDeleteMutex](#) (void *mutex)
Delete a mutex.
- void [STB_OSResetCPU](#) (void)
Reset the board.

4.74.1 Detailed Description

Header file - Function prototypes for operating system.

Date

12/02/2002

4.74.2 Function Documentation

4.74.2.1 void* [STB_OSCreateCountSemaphore](#) (U32BIT value)

Create a counting semaphore.

Parameters

<i>value</i>	initial value for semaphore.
--------------	------------------------------

Returns

Semaphore handle upon success, or NULL upon failure.

4.74.2.2 void* STB_OSCreateMutex (void)

Create a mutex.

Returns

Newly created mutex, or NULL.

4.74.2.3 void* STB_OSCreateQueue (U16BIT *msg_size*, U16BIT *msg_max*)

Create Queue of given number of messages and size of message.

Parameters

<i>msg_size</i>	Queue Message Packet Size
<i>num_msgs</i>	Queue Message Depth in Packets

Returns

Queue Handle - Number for success, NULL upon failure.

4.74.2.4 void* STB_OSCreateSemaphore (void)

Create a Semaphore.

Returns

Semaphore Handle Address upon success, or NULL upon failure.

4.74.2.5 void* STB_OSCreateTask (void(*) (void *) *function*, void * *param*, U32BIT *stack*, U8BIT *priority*, U8BIT * *name*)

Create a New Task to the calling process. Upon success, the created task runs on its own stack, as part of the calling process context. Task termination must be achieved by calling [STB_OSDestroyTask\(\)](#)

Parameters

<i>function</i>	task entry point
<i>param</i>	user defined parameter passed when task is started
<i>stack</i>	stack size
<i>priority</i>	task priority, min 0, max 15
<i>name</i>	task name

Returns

handle of task

4.74.2.6 void STB_OSDeleteMutex (void * *mutex*)

Delete a mutex.

Parameters

<i>mutex_var</i>	The mutex to delete.
------------------	----------------------

4.74.2.7 void STB_OSDeleteSemaphore (void * *semaphore*)

Delete a Semaphore.

Parameters

<i>semaphore</i>	Semaphore handle.
------------------	-------------------

Returns

TRUE for success, FALSE upon failure.

4.74.2.8 BOOLEAN STB_OSDestroyQueue (void * *queue*)

Destroy Queue.

Parameters

<i>queue</i>	Unique Queue Handle Identifier Variable Address.
--------------	--

Returns

TRUE for success, FALSE upon failure.

4.74.2.9 U32BIT STB_OSGetClockDiff (U32BIT *timestamp*)

Get Difference between Given Time and Current Time.

Parameters

<i>timestamp</i>	Given Clock Value to Compare Against.
------------------	---------------------------------------

Returns

Time Difference in MilliSeconds.

4.74.2.10 U32BIT STB_OSGetClockGMT (void)

Returns the system time in seconds.

Returns

system time in seconds

4.74.2.11 U32BIT STB_OSGetClockMilliseconds (void)

Get Current Computer Clock Time.

Returns

Time in Milliseconds.

4.74.2.12 U32BIT STB_OSGetClockRTC (void)

Returns the current time in seconds. This is calculated by using the set UTC time and adding the difference between the system boot time when it was set (sync_time) and the system boot time now.

Returns

The current time in seconds since midnight 1-1-1970.

4.74.2.13 void* STB_OSGetCurrentTask (void)

Returns the handle of the current task.

Returns

Handle of current task

4.74.2.14 void STB_OSInitCountSemaphore (void * semaphore, U32BIT value)

Initialise a counting semaphore.

Parameters

<i>semaphore</i>	Semaphore handle.
<i>value</i>	New value for semaphore.

Warning

This is a very dangerous function, and should be used very carefully.

4.74.2.15 void STB_OSMutexLock (void * *mutex*)

Lock a mutex (a.k.a. 'enter', 'wait' or 'get').

Parameters

<i>mutex_var</i>	The mutex to lock.
------------------	--------------------

4.74.2.16 void STB_OSMutexUnlock (void * *mutex*)

Unlock a mutex (a.k.a. 'leave', 'signal' or 'release')

Parameters

<i>mutex_var</i>	The mutex to unlock.
------------------	----------------------

4.74.2.17 BOOLEAN STB_OSReadQueue (void * *queue*, void * *msg*, U16BIT *msg_size*, U16BIT *timeout*)

Read a message from a queue.

Parameters

<i>queue</i>	Queue Handle
<i>data</i>	User's Read Message Buffer Start Address.
<i>msg_size</i>	Message Packet Size in Bytes.
<i>timeout</i>	timeout in milliseconds

Returns

TRUE for success, FALSE upon failure.

4.74.2.18 void STB_OSRegisterCallback (void(*) (BOOLEAN, U16BIT, U16BIT, void *, U32BIT) *func*)

Register the function that will be called when STB_OSSendEvent is used.

Parameters

<i>func</i>	callback function
-------------	-------------------

4.74.2.19 void STB_OSSemaphoreSignal (void * *semaphore*)

Signal a Semaphore to Release it by decrementing its counter.

Parameters

<i>semaphore</i>	Semaphore handle.
------------------	-------------------

4.74.2.20 void STB_OSSemaphoreWait (void * *semaphore*)

Wait on Semaphore Indefinity or Until Released.

Parameters

<i>semaphore</i>	Semaphore handle.
------------------	-------------------

Returns

TRUE for success, FALSE upon failure.

4.74.2.21 BOOLEAN STB_OSSemaphoreWaitTimeout (void * *semaphore*, U16BIT *timeout*)

Wait on Semaphore for Set Time Period in an Attempt to Acquire.

Parameters

<i>semaphore</i>	Semaphore handle.
<i>timeout</i>	Time Period to Wait in milliseconds.

Returns

TRUE for success, FALSE upon failure.

4.74.2.22 void STB_OSSendEvent (BOOLEAN *repeat*, U16BIT *event_class*, U16BIT *event_type*, void * *data*, U32BIT *data_size*)

Send an event by calling the registered callback function.

Parameters

<i>repeat</i>	TRUE if the event is a repeat of the last event
<i>event_class</i>	event class
<i>event_type</i>	event identifier.
<i>data</i>	pointer to the data associated with the event
<i>data_size</i>	size of the data pointed by <i>data_pointer</i>

4.74.2.23 void STB_OSClockGMT (U32BIT *num_seconds*)

Set the time in seconds since midnight 1-1-1970 in GMT.

Parameters

<i>num_ - seconds</i>	time in seconds
-----------------------	-----------------

4.74.2.24 void STB_OSClockRTC (U32BIT *num_seconds*)

Set the local time in seconds since midnight 1-1-1970.

Parameters

<i>num_ - seconds</i>	time in seconds
-----------------------	-----------------

4.74.2.25 void STB_OSTaskDelay (U16BIT *timeout*)

Delay Task for Specified Time Period.

Parameters

<i>timeout</i>	delay in milliSeconds.
----------------	------------------------

4.74.2.26 U8BIT STB_OSTaskPriority (void * *task*, U8BIT *priority*)

Set a New Priority Level for Specified Task.

Parameters

<i>task</i>	task whose priority is to be changed
<i>priority</i>	new priority

Returns

Old task priority

4.74.2.27 BOOLEAN STB_OSWriteQueue (void * *queue*, void * *msg*, U16BIT *msg_size*, U16BIT *timeout*)

Write a message to the queue.

Parameters

<i>queue</i>	Queue Handle
<i>data</i>	message to be queued
<i>msg_size</i>	size of message in bytes
<i>timeout</i>	timeout in milliseconds

Returns

TRUE for success, FALSE upon failure.

4.75 platform/inc/stbhwsd.h File Reference

Header file - Function prototypes for OSD control.

```
#include "osdtype.h" Include dependency graph for stbhwsd.h:
```

Typedefs

- typedef enum e_blit_op **E_BLIT_OP**

Enumerations

- enum **e_blit_op** { **STB_BLIT_COPY**, **STB_BLIT_A_BLEND** }

Functions

- void [STB_OSDInitialise](#) (U8BIT num_max_regions)
Initialised the OSD hardware layer functions.
- BOOLEAN [STB_OSDEnable](#) (BOOLEAN enable)
Enable/Disable the OSD.
- BOOLEAN [STB_OSDDisableUIRegion](#) (void)
Disables (makes invisible) the OSD. This function needs to be implemented on platforms that cannot display the UI at the same time as subtitles or teletext.
- BOOLEAN [STB_OSDEnableUIRegion](#) (void)
Disables (makes invisible) the OSD. This function needs to be implemented on platforms that cannot display the UI at the same time as subtitles or teletext.
- void [STB_OSDSetTransparency](#) (U8BIT trans)
Sets the UI transparency level (0-100%)
- U8BIT [STB_OSDGetTransparency](#) (void)
Returns the current UI transparency level.
- void [STB_OSDSetPalette](#) (U16BIT index, U16BIT num, U32BIT *trgb)
Sets a range of palette entries to Trans/Red/Grn/Blue levels. This function is used for 8 bit colour depth only.
- U32BIT * [STB_OSDGetCurrentPalette](#) (void)
Returns a pointer to the current TRGB palette (clut). This function is used for 8 bit colour depth only.
- void [STB_OSDDrawBitmap](#) (U16BIT x, U16BIT y, U16BIT width, U16BIT height, U8BIT bits, U8BIT *data)
Draw a bitmap into the UI composition (invisible) buffer.
- void [STB_OSDReadBitmap](#) (U16BIT x, U16BIT y, U16BIT width, U16BIT height, U8BIT bits, U8BIT *data)

- Read a bitmap from the UI composition (invisible) buffer.*
- void [STB_OSDDrawPixel](#) (U16BIT x, U16BIT y, U32BIT colour)
- Draw a single pixel in the UI composition buffer.*
- void [STB_OSDReadPixel](#) (U16BIT x, U16BIT y, U32BIT *colour)
- Read a single pixel from the UI composition buffer.*
- void [STB_OSDDrawHLine](#) (U16BIT x, U16BIT y, U16BIT width, U32BIT colour)
- Draw a horizontal line in the UI composition buffer.*
- void [STB_OSDDrawVLine](#) (U16BIT x, U16BIT y, U16BIT height, U32BIT colour)
- Draw a vertical line in the UI composition buffer.*
- void [STB_OSDDrawRectangle](#) (U16BIT x, U16BIT y, U16BIT width, U16BIT height, U32BIT colour, U8BIT thick, BOOLEAN fill)
- Draw a rectangle in the UI composition buffer.*
- void [STB_OSDClear](#) (U32BIT colour)
- Clear the entire UI composition buffer to a single colour.*
- void [STB_OSDFill](#) (U32BIT colour, E_BLIT_OP bflg)
- Clear the user interface layer to the given colour using the given blit op.*
- void [STB_OSDGetSize](#) (U16BIT *width, U16BIT *height)
- Returns the current width and height of the OSD.*
- void [STB_OSDUpdate](#) (void)
- Commit invisible UI buffer to visible surface and copy back.*
- void [STB_OSDRegisterRefreshHandler](#) (void(*func)(void))
- Register app fn that can be used by the platform code OSD module to notify the application when the UI needs to be redrawn.*
- void [STB_OSDRegisterInUseCallback](#) (U8BIT(*func)(void))
- App registered callback to indicate if OSD is in use.*
- void [STB_OSDRefreshDisplayCallback](#) (void)
- Can be called by anyone to force redraw of the OSD by the UI. This function will cause the platform code OSD module to call the registered refresh handler (see STB_OSD-RegisterRefreshHandler).*
- void [STB_OSDResize](#) (BOOLEAN scaling, U16BIT width, U16BIT height, U16BIT x_offset, U16BIT y_offset)
- Reconfigures the OSD for a new screen size.*
- void [STB_OSDSetRegionDisplaySize](#) (U16BIT width, U16BIT height)
- Should be called to set the size of the display so that SD subtitles can be scaled correctly for an HD display, or vice versa.*
- void * [STB_OSDCreateRegion](#) (U16BIT width, U16BIT height, U8BIT depth)
- Creates a new OSD region (for subtitling)*
- void [STB_OSDDestroyRegion](#) (void *handle)
- Destroys (free the resources used by) a region.*
- void [STB_OSDSetYCrCbPalette](#) (void *region_handle, U32BIT *tycrb)
- Sets a regions entire palette to a T,Y,CR,CB clut.*
- void [STB_OSDMoveRegion](#) (void *handle, U16BIT x, U16BIT y)
- Move a region to new coordinates.*
- void [STB_OSDHideRegion](#) (void *handle)

- Makes a region invisible.*

 - void [STB_OSDShowRegion](#) (void *handle)
- Makes a region visible.*

 - void [STB_OSDDrawBitmapInRegion](#) (void *handle, U16BIT x, U16BIT y, U16BIT w, U16BIT h, U8BIT *bitmap, BOOLEAN non_modifying_colour)
- Draw a bitmap in a specified region.*

 - void [STB_OSDRegionToRegionCopy](#) (void *handle_new, void *handle_orig)
- Copy a region to another region, including palette.*

 - void [STB_OSDFillRegion](#) (void *handle, U8BIT colour)
- Fill a region with a colour.*

 - void [STB_OSDRegionFillRect](#) (void *handle, U16BIT left, U16BIT top, U16BIT width, U16BIT height, U8BIT colour)
- Fill the rectangle within the given region with the given colour.*

 - void [STB_OSDUpdateRegions](#) (void)
- Updates the display of all subtitle regions.*

 - void [STB_OSDSetRGBPalette](#) (void *handle, U32BIT *trgb)
- Sets the RGB palette for the given region. This function is used for Teletext.*

 - void [STB_OSDMhegSetPalette](#) (U16BIT index, U16BIT number, const U32BIT *argb)
- Sets Colour Palette array of up to 256 values, for single byte colour depth. This palette being an array of 'U32BIT' (8 Alpha, 8 Red, 8 Green, 8 Blue).*

 - void * [STB_OSDMhegSetResolution](#) (U16BIT width, U16BIT height, U8BIT bits)
- Sets the size of the OSD to be used by MHEG engine. The return may be NULL, or pointer to valid Surface for the entire screen back-buffer. If it is the latter, then the Engine will draw to this surface using STB_OSDMhegBmpToSurf and STB_OSDMhegFillSurface, or by calling STB_OSDMhegLockBuffer to draw directly to buffer.*

 - void * [STB_OSDMhegCreateSurface](#) (U16BIT width, U16BIT height, BOOLEAN init, U32BIT colour)
- Creates a hardware surface on which MHEG5 engine will draw an individual MHEG object. At its basic the function can just allocate the buffer to be returned by [STB_OSDMhegLockBuffer\(\)](#). Its size being: (width * height * bytes_per_pixel) Also, when 'init' is TRUE, function initialises surface buffer to the specified colour. For pixel colour format of less than four bytes, use least significant bits of 'colour'.*

 - void * [STB_OSDMhegLockBuffer](#) (void *surface, U32BIT *pPitch)
- Converts hardware surface handle returned by [STB_OSDMhegCreateSurface\(\)](#) to buffer address that the engine needs in order to draw the MHEG object. This function can inform HW that the engine needs write access to buffer. MHEG5 will use the return address and 'pitch' (or stride) value to locate pixel data. Before calling this function, 'pitch' is initialised to width as given by [STB_OSDMhegCreateSurface\(\)](#), but platform can alter this here.*

 - void [STB_OSDMhegUnlockBuffer](#) (void *surface)
- This function informs HW that MHEG5 is finished writing to the buffer.*

 - void [STB_OSDMhegDestroySurface](#) (void *surface)
- This function destroys surface and all data allocated by [STB_OSDMhegCreateSurface\(\)](#)*

- void [STB_OSDMhegBlitBitmap](#) (void *surface, [S_RECTANGLE](#) *pRect, U32BIT pitch, U16BIT screen_x, U16BIT screen_y, E_BLIT_OP bflg)
Render bitmap on OSD back buffer in the given screen location, with given operation. The bitmap is referenced 'surface' - a handle returned by [STB_OSDMhegCreateSurface\(\)](#) 1. It is a one-to-one mapping between surface pixels and screen pixels, so rect.width and rect.height give size of rectangle on the screen as well. 2. (rect.top + rect.height) is guaranteed to be less than or equal to height given to [STB_OSDMhegCreateSurface\(\)](#)
- void [STB_OSDMhegFillRectangle](#) ([S_RECTANGLE](#) *pRect, U32BIT colour, E_BLIT_OP bflg)
Draw a filled rectangle on OSD back buffer in the location given. Where pixel colour is less than four bytes, use least significant bits in 'colour'. 'rect' can be part of the screen or the entire screen.
- void [STB_OSDMhegBlitStretch](#) ([S_RECTANGLE](#) *pSrcRect, void *src_surf, [S_RECTANGLE](#) *pDstRect, void *dst_surf, U8BIT alpha_blend)
Stretch blit bitmap data from source surface to destination surface using source and destination rectangles.
- void [STB_OSDMhegFillSurface](#) (void *surface, [S_RECTANGLE](#) *pRect, U32BIT colour, E_BLIT_OP bflg)
Draw a filled rectangle on surface in the location given. Where pixel colour is less than four bytes, use least significant bits in 'colour'.
- void [STB_OSDMhegUpdate](#) (void)
Commit OSD changes to the screen - changes given by previous calls to [STB_OSDMhegDrawRectangle\(\)](#) and [STB_OSDMhegDrawBitmap\(\)](#).
- void [STB_OSDMhegClear](#) (void)
Clear MHEG's entire OSD.

4.75.1 Detailed Description

Header file - Function prototypes for OSD control.

Date

06/02/2001

4.75.2 Function Documentation

4.75.2.1 void [STB_OSDClear](#) (U32BIT colour)

Clear the entire UI composition buffer to a single colour.

Parameters

<i>colour</i>	colour to clear to
---------------	--------------------

4.75.2.2 void* [STB_OSDCreateRegion](#) (U16BIT width, U16BIT height, U8BIT depth)

Creates a new OSD region (for subtitling)

Parameters

<i>width</i>	width of new region
<i>height</i>	height of new region
<i>depth</i>	bits per pixel of new region

Returns

handle (pointer to) new region

4.75.2.3 void STB_OSDDestroyRegion (void * *handle*)

Destroys (free the resources used by) a region.

Parameters

<i>handle</i>	handle of (pointer to) the region
---------------	-----------------------------------

4.75.2.4 BOOLEAN STB_OSDDisableUIRegion (void)

Disables (makes invisible) the OSD. This function needs to be implemented on platforms that cannot display the UI at the same time as subtitles or teletext.

Returns

TRUE if succesful, FALSE otherwise

4.75.2.5 void STB_OSDDrawBitmap (U16BIT *x*, U16BIT *y*, U16BIT *width*, U16BIT *height*, U8BIT *bits*, U8BIT * *data*)

Draw a bitmap into the UI composition (invisible) buffer.

Parameters

<i>x</i>	the x coordinate where to draw
<i>y</i>	the x coordinate where to draw
<i>width</i>	width of bitmap in pixels
<i>height</i>	height of bitmap in pixels
<i>bits</i>	bits per pixel of source bitmap
<i>data</i>	the bitmap data

4.75.2.6 void STB_OSDDrawBitmapInRegion (void * *handle*, U16BIT *x*, U16BIT *y*, U16BIT *w*, U16BIT *h*, U8BIT * *bitmap*, BOOLEAN *non_modifying_colour*)

Draw a bitmap in a specified region.

Parameters

<i>handle</i>	handle of (pointer to) the region
<i>x</i>	x coordinate to draw bitmap

<i>y</i>	y coordinate to draw bitmap
<i>w</i>	width of bitmap
<i>h</i>	height of bitmap
<i>bitmap</i>	the bitmap data
<i>non_ - modifying_ - colour</i>	not used

4.75.2.7 void STB_OSDDrawHLine (U16BIT x, U16BIT y, U16BIT width, U32BIT colour)

Draw a horizontal line in the UI composition buffer.

Parameters

<i>x</i>	x coordinate of line
<i>y</i>	y coordinate of line
<i>width</i>	width of line in pixels
<i>colour</i>	colour of line

4.75.2.8 void STB_OSDDrawPixel (U16BIT x, U16BIT y, U32BIT colour)

Draw a single pixel in the UI composition buffer.

Parameters

<i>x</i>	x coordinate of pixel
<i>y</i>	y coordinate of pixel
<i>colour</i>	colour of pixel

4.75.2.9 void STB_OSDDrawRectangle (U16BIT x, U16BIT y, U16BIT width, U16BIT height, U32BIT colour, U8BIT thick, BOOLEAN fill)

Draw a rectangle in the UI composition buffer.

Parameters

<i>x</i>	x coordinate of rectangle
<i>y</i>	y coordinate of rectangle
<i>width</i>	width of rectangle
<i>height</i>	height of rectangle
<i>colour</i>	colour of rectangle
<i>thick</i>	thickness of outline for hollow rectangles
<i>fill</i>	TRUE for solid (filled) rectangle

4.75.2.10 void STB_OSDDrawVLine (U16BIT x, U16BIT y, U16BIT height, U32BIT colour)

Draw a vertical line in the UI composition buffer.

Parameters

<i>x</i>	x coordinate of line
<i>y</i>	y coordinate of line
<i>height</i>	height of line in pixels
<i>colour</i>	colour of line

4.75.2.11 **BOOLEAN STB_OSDEnable (BOOLEAN *enable*)**

Enable/Disable the OSD.

Parameters

<i>enable</i>	TRUE to enable
---------------	----------------

Returns

The new state (i.e. will = param if successful)

4.75.2.12 **BOOLEAN STB_OSDEnableUIRegion (void)**

Disables (makes invisible) the OSD. This function needs to be implemented on platforms that cannot display the UI at the same time as subtitles or teletext.

Returns

TRUE if succesful, FALSE otherwise

4.75.2.13 **void STB_OSDFill (U32BIT *colour*, E_BLIT_OP *bflg*)**

Clear the user interface layer to the given colour using the given blit op.

Parameters

<i>colour</i>	colour to clear to
<i>bflg</i>	blit operation

4.75.2.14 **void STB_OSDFillRegion (void * *handle*, U8BIT *colour*)**

Fill a region with a colour.

Parameters

<i>handle</i>	handle of (pointer to) the region
<i>colour</i>	the colour index to fill with

4.75.2.15 U32BIT* STB_OSDGetCurrentPalette (void)

Returns a pointer to the current TRGB palette (clut). This function is used for 8 bit colour depth only.

Returns

Pointer to an array of 32bit trans,red,green,blue values

4.75.2.16 void STB_OSDGetSize (U16BIT * width, U16BIT * height)

Returns the current width and height of the OSD.

Parameters

<i>width</i>	width of OSD in pixels
<i>height</i>	height of OSD in pixels

4.75.2.17 U8BIT STB_OSDGetTransparency (void)

Returns the current UI transparency level.

Returns

The current transparency in percent

4.75.2.18 void STB_OSDHideRegion (void * handle)

Makes a region invisible.

Parameters

<i>handle</i>	handle of (pointer to) the region
---------------	-----------------------------------

4.75.2.19 void STB_OSDInitialise (U8BIT num_max_regions)

Initialised the OSD hardware layer functions.

Parameters

<i>num_max_regions</i>	number of regions required (not used here)
------------------------	--

4.75.2.20 void STB_OSDMhegBlitBitmap (void * surface, S_RECTANGLE * pRect, U32BIT pitch, U16BIT screen_x, U16BIT screen_y, E_BLIT_OP bflg)

Render bitmap on OSD back buffer in the given screen location, with given operation. The bitmap is referenced 'surface' - a handle returned by [STB_OSDMhegCreateSurface\(\)](#) 1. It is a one-to-one mapping between surface pixels and screen pixels, so

rect.width and rect.height give size of rectangle on the screen as well. 2. (rect.top + rect.height) is guaranteed to be less than or equal to height given to [STB_OSDMhegCreateSurface\(\)](#)

Parameters

<i>surface</i>	Handle of surface returned by STB_OSDMhegCreateSurface
<i>rect</i>	source rectangle within surface - top/left is offset into bitmap referenced by 'surface', width/height gives size.
<i>pitch</i>	Width of line of source bitmap data - as returned by STB_OSDMhegLockBuffer()
<i>screen_x</i>	Left or X position on screen to draw bitmap
<i>screen_y</i>	Top or Y position on screen to draw bitmap
<i>bflg</i>	Operation - COPY or ALPHA BLEND

Returns

void

4.75.2.21 void STB_OSDMhegBlitStretch (S_RECTANGLE * pSrcRect, void * src_surf, S_RECTANGLE * pDstRect, void * dst_surf, U8BIT alpha_blend)

Stretch blit bitmap data from source surface to destination surface using source and destination rectangles.

Parameters

<i>pSrcRect</i>	rectangle for bitmap data
<i>src_surf</i>	handle returned by STB_OSDMhegCreateSurface
<i>pDstRect</i>	rectangle for destination on surface
<i>dst_surf</i>	handle returned by STB_OSDMhegCreateSurface
<i>alpha_blend</i>	Operation - COPY or ALPHA BLEND (zero or 0xff)

Returns

void

4.75.2.22 void* STB_OSDMhegCreateSurface (U16BIT width, U16BIT height, BOOLEAN init, U32BIT colour)

Creates a hardware surface on which MHEG5 engine will draw an individual MHEG object. At its basic the function can just allocate the buffer to be returned by [STB_OSDMhegLockBuffer\(\)](#). It's size being: (width * height * bytes_per_pixel) Also, when 'init' is TRUE, function initialises surface buffer to the specified colour. For pixel colour format of less than four bytes, use least significant bits of 'colour'.

Parameters

<i>width</i>	Width of requested surface in pixels
<i>height</i>	Height of requested surface in pixels
<i>init</i>	If TRUE, initialise buffer with colour.
<i>colour</i>	colour for all pixels in buffer.

Returns

void* Success - Handle to surface. Failure - NULL (or zero)

4.75.2.23 void STB_OSDMhegDestroySurface (void * *surface*)

This function destroys surface and all data allocated by [STB_OSDMhegCreateSurface\(\)](#)

Parameters

<i>surface</i>	Handle of surface returned by STB_OSDMhegCreateSurface
----------------	--

Returns

void

4.75.2.24 void STB_OSDMhegFillRectangle (S_RECTANGLE * *pRect*, U32BIT *colour*, E_BLIT_OP *bflg*)

Draw a filled rectangle on OSD back buffer in the location given. Where pixel colour is less than four bytes, use least significant bits in 'colour'. 'rect' can be part of the screen or the entire screen.

Parameters

<i>rect</i>	rectangle on screen - with top,left starting position
<i>colour</i>	colour for all pixels in rectangle.
<i>bflg</i>	Operation - COPY or ALPHA BLEND

Returns

void

4.75.2.25 void STB_OSDMhegFillSurface (void * *surface*, S_RECTANGLE * *pRect*, U32BIT *colour*, E_BLIT_OP *bflg*)

Draw a filled rectangle on surface in the location given. Where pixel colour is less than four bytes, use least significant bits in 'colour'.

Parameters

<i>surface</i>	handle returned by STB_OSDMhegCreateSurface
<i>rect</i>	rectangle on screen - with top,left starting position

<i>colour</i>	colour for all pixels in rectangle.
<i>bflg</i>	Operation - COPY or ALPHA BLEND

Returns

void

4.75.2.26 void* STB_OSDMhegLockBuffer (void * *surface*, U32BIT * *pPitch*)

Converts hardware surface handle returned by [STB_OSDMhegCreateSurface\(\)](#) to buffer address that the engine needs in order to draw the MHEG object. This function can inform HW that the engine needs write access to buffer. MHEG5 will use the return address and 'pitch' (or stride) value to locate pixel data. Before calling this function, 'pitch' is initialised to width as given by [STB_OSDMhegCreateSurface\(\)](#), but platform can alter this here.

Parameters

<i>surface</i>	Handle of surface returned by STB_OSDMhegCreateSurface
<i>pitch</i>	width in bytes of one line of pixel data in buffer

Returns

void* Address of the buffer

4.75.2.27 void STB_OSDMhegSetPalette (U16BIT *index*, U16BIT *number*, const U32BIT * *argb*)

Sets Colour Palette array of up to 256 values, for single byte colour depth. This palette being an array of 'U32BIT' (8 Alpha, 8 Red, 8 Green, 8 Blue).

Parameters

<i>index</i>	
<i>number</i>	Size of palette array
<i>argb</i>	pointer to palette array

4.75.2.28 void* STB_OSDMhegSetResolution (U16BIT *width*, U16BIT *height*, U8BIT *bits*)

Sets the size of the OSD to be used by MHEG engine. The return may be NULL, or pointer to valid Surface for the entire screen back-buffer. If it is the latter, then the Engine will draw to this surface using STB_OSDMhegBmpToSurf and STB_OSDMhegFillSurface, or by calling STB_OSDMhegLockBuffer to draw directly to buffer.

Parameters

<i>width</i>	Width of MHEG OSD resolution
<i>height</i>	Height of MHEG OSD resolution
<i>bits</i>	Number of bits per pixel

4.75.2.29 void STB_OSDMhegUnlockBuffer (void * *surface*)

This function informs HW that MHEG5 is finished writing to the buffer.

Parameters

<i>surface</i>	Handle of surface returned by STB_OSDMhegCreateSurface
----------------	--

Returns

void

4.75.2.30 void STB_OSDMhegUpdate (void)

Commit OSD changes to the screen - changes given by previous calls to STB_OSDMhegDrawRectangle() and STB_OSDMhegDrawBitmap().

Returns

void

4.75.2.31 void STB_OSDMoveRegion (void * *handle*, U16BIT *x*, U16BIT *y*)

Move a region to new coordinates.

Parameters

<i>handle</i>	handle of (pointer to) the region
<i>x</i>	new x coordinate of region
<i>y</i>	new y coordinate of region

4.75.2.32 void STB_OSReadBitmap (U16BIT *x*, U16BIT *y*, U16BIT *width*, U16BIT *height*, U8BIT *bits*, U8BIT * *data*)

Read a bitmap from the UI composition (invisible) buffer.

Parameters

<i>x</i>	the x coordinate where to read
<i>y</i>	the x coordinate where to read
<i>width</i>	width of bitmap in pixels
<i>height</i>	height of bitmap in pixels
<i>bits</i>	bits per pixel of destination bitmap
<i>data</i>	the resultant bitmap data

4.75.2.33 void STB_OSDReadPixel (U16BIT *x*, U16BIT *y*, U32BIT * *colour*)

Read a single pixel from the UI composition buffer.

Parameters

<i>x</i>	x coordinate of pixel
<i>y</i>	y coordinate of pixel
<i>colour</i>	colour of pixel

4.75.2.34 void STB_OSDRegionFillRect (void * *handle*, U16BIT *left*, U16BIT *top*, U16BIT *width*, U16BIT *height*, U8BIT *colour*)

Fill the rectangle within the given region with the given colour.

Parameters

<i>handle</i>	- region handle
<i>left</i>	- x position of rectangle within the region
<i>top</i>	- y position of rectangle within the region
<i>width</i>	- rectangle width
<i>height</i>	rectangle height
<i>colour</i>	- fill colour

4.75.2.35 void STB_OSDRegionToRegionCopy (void * *handle_new*, void * *handle_orig*)

Copy a region to another region, including palette.

Parameters

<i>handle_new</i>	handle of (pointer to) new (destination) region
<i>handle_old</i>	handle of (pointer to) old (source) region

4.75.2.36 void STB_OSDRegisterInUseCallback (U8BIT(*) (void) *func*)

App registered callback to indicate if OSD is in use.

Parameters

<i>func</i>	the callback function
-------------	-----------------------

4.75.2.37 void STB_OSDRegisterRefreshHandler (void(*) (void) *func*)

Register app fn that can be used by the platform code OSD module to notify the application when the UI needs to be redrawn.

Parameters

<i>func</i>	the callback function
-------------	-----------------------

4.75.2.38 void **STB_OSDResize** (*BOOLEAN scaling*, *U16BIT width*, *U16BIT height*, *U16BIT x_offset*, *U16BIT y_offset*)

Reconfigures the OSD for a new screen size.

Parameters

<i>scaling</i>	TRUE if osd scaling is required due to MHEG scene aspect ratio, FALSE otherwise
<i>width</i>	width of OSD in pixels
<i>height</i>	height of OSD in pixels
<i>x_offset</i>	offset of OSD from left of screen, in pixels
<i>y_offset</i>	offset of OSD from top of screen, in pixels

4.75.2.39 void **STB_OSDSetPalette** (*U16BIT index*, *U16BIT num*, *U32BIT * trgb*)

Sets a range of palette entries to Trans/Red/Grn/Blue levels. This function is used for 8 bit colour depth only.

Parameters

<i>index</i>	starting number of palette entry to be set
<i>num</i>	number of consecutive palette entries to set
<i>trgb</i>	the colour value array

4.75.2.40 void **STB_OSDSetRegionDisplaySize** (*U16BIT width*, *U16BIT height*)

Should be called to set the size of the display so that SD subtitles can be scaled correctly for an HD display, or vice versa.

Parameters

<i>width</i>	- display width defined by the subtitle DDS
<i>height</i>	- display height defined by the subtitle DDS

4.75.2.41 void **STB_OSDSetRGBPalette** (*void * handle*, *U32BIT * trgb*)

Sets the RGB palette for the given region. This function is used for Teletext.

Parameters

<i>handle</i>	region handle as returned by STB_OSDCreateRegion.
<i>trgb</i>	pointer to the palette array

4.75.2.42 void **STB_OSDSetTransparency** (*U8BIT trans*)

Sets the UI transparency level (0-100%)

Parameters

<i>trans</i>	transparency in percent
--------------	-------------------------

4.75.2.43 void STB_OSDSetYCrCbPalette (void * *region_handle*, U32BIT * *tycrCb*)

Sets a regions entire palette to a T,Y,CR,CB clut.

Parameters

<i>handle</i>	handle (pointer to) the region to configure
<i>tycrCb</i>	pointer to the CLUT entries

4.75.2.44 void STB_OSDShowRegion (void * *handle*)

Makes a region visible.

Parameters

<i>handle</i>	handle of (pointer to) the region
---------------	-----------------------------------

4.76 platform/inc/stbhwtun.h File Reference

Header file - Function prototypes for tuner control.

#include "stbhwc.h" Include dependency graph for stbhwtun.h: This graph shows which files directly or indirectly include this file:

Defines

- #define **SYMBOL_RATE_AUTO** 0

Typedefs

- typedef enum e_stb_tune_system_type **E_STB_TUNE_SYSTEM_TYPE**
- typedef enum e_stb_tune_signal_type **E_STB_TUNE_SIGNAL_TYPE**
- typedef enum e_stb_tune_tmode **E_STB_TUNE_TMODE**
- typedef enum e_stb_tune_tbwidht **E_STB_TUNE_TBWIDHT**
- typedef enum e_stb_tune_tconst **E_STB_TUNE_TCONST**
- typedef enum E_STB_TUNE_THIERARCHY **E_STB_TUNE_THIERARCHY**
- typedef enum e_stb_tune_tcoderate **E_STB_TUNE_TCODERATE**
- typedef enum e_stb_tune_tguardint **E_STB_TUNE_TGUARDINT**
- typedef enum e_stb_tune_cmode **E_STB_TUNE_CMODE**
- typedef enum e_stb_tune_lnb_voltage **E_STB_TUNE_LNB_VOLTAGE**
- typedef enum e_stb_tune_fec **E_STB_TUNE_FEC**
- typedef enum e_stb_tune_analog_video_type **E_STB_TUNE_ANALOG_VIDEO_TYPE**

Enumerations

- enum `e_stb_tune_system_type` { `TUNE_SYSTEM_TYPE_UNKNOWN` = 0, `TUNE_SYSTEM_TYPE_DVBT` = 1, `TUNE_SYSTEM_TYPE_DVBT2` = 2, `TUNE_SYSTEM_TYPE_DVBS` = 3, `TUNE_SYSTEM_TYPE_DVBS2` = 4, `TUNE_SYSTEM_TYPE_DVBC` = 5 }
- enum `e_stb_tune_signal_type` { `TUNE_SIGNAL_NONE` = 0, `TUNE_SIGNAL_QPSK` = 1, `TUNE_SIGNAL_COFDM` = 2, `TUNE_SIGNAL_QAM` = 3, `TUNE_SIGNAL_ANALOG` = 4 }
- enum `e_stb_tune_tmode` { `TUNE_MODE_COFDM_1K` = 0, `TUNE_MODE_COFDM_2K` = 1, `TUNE_MODE_COFDM_4K` = 2, `TUNE_MODE_COFDM_8K` = 3, `TUNE_MODE_COFDM_16K` = 4, `TUNE_MODE_COFDM_32K` = 5, `TUNE_MODE_COFDM_UNDEFINED` = 255 }
- enum `e_stb_tune_twidth` { `TUNE_TBWIDTH_8MHZ` = 0, `TUNE_TBWIDTH_7MHZ` = 1, `TUNE_TBWIDTH_6MHZ` = 2, `TUNE_TBWIDTH_5MHZ` = 3 }
- enum `e_stb_tune_tconst` { `TUNE_TCONST_QPSK` = 0, `TUNE_TCONST_QAM16` = 1, `TUNE_TCONST_QAM64` = 2, `TUNE_TCONST_QAM128` = 3, `TUNE_TCONST_QAM256` = 4, `TUNE_TCONST_UNDEFINED` = 255 }
- enum `E_STB_TUNE_THIERARCHY` { `TUNE_THIERARCHY_NONE` = 0, `TUNE_THIERARCHY_1` = 1, `TUNE_THIERARCHY_2` = 2, `TUNE_THIERARCHY_4` = 4, `TUNE_THIERARCHY_8` = 8, `TUNE_THIERARCHY_16` = 16, `TUNE_THIERARCHY_32` = 32, `TUNE_THIERARCHY_64` = 64, `TUNE_THIERARCHY_128` = 128, `TUNE_THIERARCHY_UNDEFINED` = 255 }
- enum `e_stb_tune_tcoderate` { `TUNE_TCODERATE_1_2` = 0, `TUNE_TCODERATE_2_3` = 1, `TUNE_TCODERATE_3_4` = 2, `TUNE_TCODERATE_5_6` = 3, `TUNE_TCODERATE_7_8` = 4, `TUNE_TCODERATE_UNDEFINED` = 255 }
- enum `e_stb_tune_tguardint` { `TUNE_TGUARDINT_1_32` = 0, `TUNE_TGUARDINT_1_16` = 1, `TUNE_TGUARDINT_1_8` = 2, `TUNE_TGUARDINT_1_4` = 3, `TUNE_TGUARDINT_1_128` = 4, `TUNE_TGUARDINT_19_128` = 5, `TUNE_TGUARDINT_19_256` = 6, `TUNE_TGUARDINT_UNDEFINED` = 255 }
- enum `e_stb_tune_cmode` { `TUNE_MODE_QAM_4` = 0, `TUNE_MODE_QAM_8` = 1, `TUNE_MODE_QAM_16` = 2, `TUNE_MODE_QAM_32` = 3, `TUNE_MODE_QAM_64` = 4, `TUNE_MODE_QAM_128` = 5, `TUNE_MODE_QAM_256` = 6, `TUNE_MODE_QAM_UNDEFINED` = 255 }
- enum `E_STB_TUNE_MODULATION` { `TUNE_MOD_AUTO`, `TUNE_MOD_QPSK`, `TUNE_MOD_8PSK`, `TUNE_MOD_16QAM` }
- enum `e_stb_tune_lnb_voltage` { `LNB_VOLTAGE_OFF` = 0, `LNB_VOLTAGE_14V` = 1, `LNB_VOLTAGE_18V` = 2 }
- enum `e_stb_tune_fec` { `TUNE_FEC_AUTOMATIC` = 0, `TUNE_FEC_1_2` = 1, `TUNE_FEC_2_3` = 2, `TUNE_FEC_3_4` = 3, `TUNE_FEC_5_6` = 4, `TUNE_FEC_7_8` = 5, `TUNE_FEC_1_4` = 6, `TUNE_FEC_1_3` = 7, `TUNE_FEC_2_5` = 8, `TUNE_FEC_8_9` = 9, `TUNE_FEC_9_10` = 10 }
- enum `e_stb_tune_analog_video_type` { `TUNE_ANLG_VIDEO_PAL_I` = 0, `TUNE_ANLG_VIDEO_PAL_B` = 1, `TUNE_ANLG_VIDEO_PAL_G` = 2, `TUNE_ANLG_VIDEO_PAL_D` = 3, `TUNE_ANLG_VIDEO_PAL_K` = 4, `TUNE_ANLG_VIDEO_PAL_L` = 5, `TUNE_ANLG_VIDEO_PAL_LDASH` = 6 }

Functions

- void [STB_TuneInitialise](#) (U8BIT paths)
Initialises the tuner component.
- void [STB_TuneSetSystemType](#) (U8BIT path, E_STB_TUNE_SYSTEM_TYPE type)
Set the demodulator's signal type. This function must be called before each call to STB_TuneStartTuner in a dvb-t2 system and never in a dvb-t system.
- E_STB_TUNE_SYSTEM_TYPE [STB_TuneGetSystemType](#) (U8BIT path)
Returns the signal type as set by STB_TuneSetTerrType or as re-written by the driver.
- void [STB_TuneAutoRelock](#) (U8BIT path, BOOLEAN state)
Enables or disabled auto tuner relocking.
- E_STB_TUNE_SIGNAL_TYPE [STB_TuneGetSignalType](#) (U8BIT path)
Gets the signal type of the tuner path.
- U32BIT [STB_TuneGetMinTunerFreqKHz](#) (U8BIT path)
Returns the minimum tuner frequency in KHz.
- U32BIT [STB_TuneGetMaxTunerFreqKHz](#) (U8BIT path)
Returns the maximum tuner frequency in KHz.
- void [STB_TuneStartTuner](#) (U8BIT path, U32BIT freq, U32BIT srate, E_STB_TUNE_FEC fec, S8BIT freq_off, E_STB_TUNE_TMODE tmode, E_STB_TUNE_TBWIDTH tbwidth, E_STB_TUNE_CMODE cmode, E_STB_TUNE_ANALOG_VIDEO_TYPE anlg_vtype)
Starts the tuner, it will then attempt to lock specified signal. Unrequired parameters can be passed as 0 (zero)
- void [STB_TuneRestartTuner](#) (U8BIT path)
Restarts tuner and attempts to lock to signal in StartTuner call.
- void [STB_TuneStopTuner](#) (U8BIT path)
Stops any locking attempt, or unlocks if locked.
- U8BIT [STB_TuneGetSignalStrength](#) (U8BIT path)
Returns the current signal strength.
- U8BIT [STB_TuneGetDataIntegrity](#) (U8BIT path)
Returns the current data integrity.
- U32BIT [STB_TuneGetActualTerrFrequency](#) (U8BIT path)
Returns the actual frequency of the current terrestrial signal.
- S8BIT [STB_TuneGetActualTerrFreqOffset](#) (U8BIT path)
Returns the actual freq offset of the current terrestrial signal.
- E_STB_TUNE_TMODE [STB_TuneGetActualTerrMode](#) (U8BIT path)
Returns the actual mode of the current terrestrial signal.
- E_STB_TUNE_TBWIDTH [STB_TuneGetActualTerrBwidth](#) (U8BIT path)
Returns the actual bandwidth of the current terrestrial signal.
- E_STB_TUNE_TCONST [STB_TuneGetActualTerrConstellation](#) (U8BIT path)
Returns the constellation of the current terrestrial signal.
- E_STB_TUNE_THIERARCHY [STB_TuneGetActualTerrHierarchy](#) (U8BIT path)
Returns the heirarchy of the current terrestrial signal.

- E_STB_TUNE_TCODERATE [STB_TuneGetActualTerrLpCodeRate](#) (U8BIT path)
Returns the LP code rate of the current terrestrial signal.
- E_STB_TUNE_TCODERATE [STB_TuneGetActualTerrHpCodeRate](#) (U8BIT path)
Returns the HP code rate of the current terrestrial signal.
- E_STB_TUNE_TGUARDINT [STB_TuneGetActualTerrGuardInt](#) (U8BIT path)
Returns the guard interval of the current terrestrial signal.
- U16BIT [STB_TuneGetActualTerrCellId](#) (U8BIT path)
Returns the cell id the current terrestrial signal.
- void [STB_TuneSetPLP](#) (U8BIT path, U8BIT plp)
Sets the Physical Layer Pipe to be acquired.
- U8BIT [STB_TuneGetPLP](#) (U8BIT path)
Gets the Physical Layer Pipe to be acquired.
- void [STB_TuneActiveAerialPower](#) (U8BIT path, BOOLEAN enabled)
Enables/disables aerial power for DVB-T.
- void [STB_TuneSetLNBfrequencies](#) (U8BIT path, U16BIT low_freq_mhz, U16BIT high_freq_mhz)
Gets the Physical Layer Pipe to be acquired.
- void [STB_TuneSetModulation](#) (U8BIT path, E_STB_TUNE_MODULATION modulation)
Sets the type of modulation for the specified tuner.
- void [STB_TuneSetLNBVoltage](#) (U8BIT path, E_STB_TUNE_LNB_VOLTAGE voltage)
Sets the LNB voltage for the given tuner.
- void [STB_TuneSet22kState](#) (U8BIT path, BOOLEAN state)
Turns the 22 kHz tone on or off.
- void [STB_TuneSet12VSwitch](#) (U8BIT path, BOOLEAN state)
Sets the 12V switch for the given tuner.
- U8BIT [STB_TuneGetDISEQCReply](#) (U8BIT path, U8BIT *data, U32BIT timeout)
Receives a DisEqc reply.
- void [STB_TuneSendDISEQCMessage](#) (U8BIT path, U8BIT *data, U8BIT size)
Sends the DisEqc message.
- void [STB_TuneSetPulseLimitEast](#) (U8BIT path, U16BIT count)
Sets the pulse limit for the east.
- void [STB_TuneSetPulseLimitWest](#) (U8BIT path, U16BIT count)
Sets the pulse limit for the west.
- void [STB_TuneChangePulsePosition](#) (U8BIT path, U16BIT count)
- U16BIT [STB_TuneGetPulsePosition](#) (U8BIT path)
Returns the current pulse position.
- void [STB_TuneAtPulsePosition](#) (U8BIT path, U16BIT position)
- void [STB_TuneChangeSkewPosition](#) (U8BIT path, U16BIT count)
Changes the value of skew position count.

- U8BIT [STB_TuneSatGetCarrierStrength](#) (U8BIT path, U32BIT freq)
Returns the carrier signal strength as a percentage.
- U32BIT [STB_TuneGetActualSymbolRate](#) (U8BIT path)
Returns the actual symbol rate when a tuner has locked.
- E_STB_TUNE_CMODE [STB_TuneGetActualCableMode](#) (U8BIT path)
Returns the cable mode when the tuner has locked.
- E_STB_TUNE_SYSTEM_TYPE [STB_TuneGetSupportedSystemType](#) (U8BIT path)
Returns the system type supported by the path. This function differs from STB_TuneGetSystemType which only returns T2 or S2 if the tuner is currently performing T2 or S2 operations.

4.76.1 Detailed Description

Header file - Function prototypes for tuner control.

Date

06/02/2001

4.76.2 Function Documentation

4.76.2.1 void STB_TuneActiveAerialPower (U8BIT *path*, BOOLEAN *enabled*)

Enables/disables aerial power for DVB-T.

Parameters

<i>path</i>	tuner path
<i>enabled</i>	TRUE to enable

4.76.2.2 void STB_TuneAutoRelock (U8BIT *path*, BOOLEAN *state*)

Enables or disabled auto tuner relocking.

Parameters

<i>path</i>	the tuner path to configure
<i>state</i>	TRUE enables relocking, FALSE disables it

4.76.2.3 void STB_TuneChangeSkewPosition (U8BIT *path*, U16BIT *count*)

Changes the value of skew position count.

Parameters

<i>path</i>	tuner path
<i>count</i>	skew position count

4.76.2.4 E_STB_TUNE_CMODE STB_TuneGetActualCableMode (U8BIT *path*)

Returns the cable mode when the tuner has locked.

Parameters

<i>path</i>	tuner path
-------------	------------

Returns

QAM mode

4.76.2.5 U32BIT STB_TuneGetActualSymbolRate (U8BIT *path*)

Returns the actual symbol rate when a tuner has locked.

Parameters

<i>path</i>	tuner path
-------------	------------

Returns

Symbol rate in symbols per second

4.76.2.6 E_STB_TUNE_TBWIDTH STB_TuneGetActualTerrBwidth (U8BIT *path*)

Returns the actual bandwidth of the current terrestrial signal.

Parameters

<i>path</i>	the tuner path to query
-------------	-------------------------

Returns

the signal bandwidth

4.76.2.7 U16BIT STB_TuneGetActualTerrCellId (U8BIT *path*)

Returns the cell id the current terrestrial signal.

Parameters

<i>path</i>	the tuner path to query
-------------	-------------------------

Returns

the cell id

4.76.2.8 E_STB_TUNE_TCONST STB_TuneGetActualTerrConstellation (U8BIT *path*)

Returns the constellation of the current terrestrial signal.

Parameters

<i>path</i>	the tuner path to query
-------------	-------------------------

Returns

the constellation

4.76.2.9 S8BIT STB_TuneGetActualTerrFreqOffset (U8BIT *path*)

Returns the actual freq offset of the current terrestrial signal.

Parameters

<i>path</i>	the tuner path to query
-------------	-------------------------

Returns

the frequency offset in Hz

4.76.2.10 U32BIT STB_TuneGetActualTerrFrequency (U8BIT *path*)

Returns the actual frequency of the current terrestrial signal.

Parameters

<i>path</i>	the tuner path to query
-------------	-------------------------

Returns

the frequency in Hz

4.76.2.11 E_STB_TUNE_TGUARDINT STB_TuneGetActualTerrGuardInt (U8BIT *path*)

Returns the guard interval of the current terrestrial signal.

Parameters

<i>path</i>	the tuner path to query
-------------	-------------------------

Returns

the guard interval

4.76.2.12 E_STB_TUNE_THIERARCHY STB_TuneGetActualTerrHierarchy (U8BIT *path*)

Returns the heirarchy of the current terrestrial signal.

Parameters

<i>path</i>	the tuner path to query
-------------	-------------------------

Returns

the heirarchy, i.e. the muximum PLP id possibly present at the current frequency.

4.76.2.13 E_STB_TUNE_TCODERATE STB_TuneGetActualTerrHpCodeRate (U8BIT *path*)

Returns the HP code rate of the current terrestrial signal.

Parameters

<i>path</i>	the tuner path to query
-------------	-------------------------

Returns

The HP code rate

4.76.2.14 E_STB_TUNE_TCODERATE STB_TuneGetActualTerrLpCodeRate (U8BIT *path*)

Returns the LP code rate of the current terrestrial signal.

Parameters

<i>path</i>	the tuner path to query
-------------	-------------------------

Returns

The LP code rate

4.76.2.15 E_STB_TUNE_TMODE STB_TuneGetActualTerrMode (U8BIT *path*)

Returns the actual mode of the current terrestrial signal.

Parameters

<i>path</i>	the tuner path to query
-------------	-------------------------

Returns

the tuning mode

4.76.2.16 U8BIT STB_TuneGetDataIntegrity (U8BIT *path*)

Returns the current data integrity.

Parameters

<i>path</i>	the tuner path to query
-------------	-------------------------

Returns

the data integrity as percentage of maximum possible (0-100)

4.76.2.17 U8BIT STB_TuneGetDISEQCReply (U8BIT *path*, U8BIT * *data*, U32BIT *timeout*)

Receives a DisEqc reply.

Parameters

<i>path</i>	tuner path
<i>data</i>	pointer to the received data
<i>timeout</i>	maximum number of milliseconds to wait for a reply

Returns

The number of bytes received

4.76.2.18 U32BIT STB_TuneGetMaxTunerFreqKHz (U8BIT *path*)

Returns the maximum tuner frequency in KHz.

Parameters

<i>path</i>	the tuner path to query
-------------	-------------------------

Returns

maximum frequency in KHz

4.76.2.19 U32BIT STB_TuneGetMinTunerFreqKHz (U8BIT *path*)

Returns the minimum tuner frequency in KHz.

Parameters

<i>path</i>	the tuner path to query
-------------	-------------------------

Returns

minimum frequency in Khz

4.76.2.20 U8BIT STB_TuneGetPLP (U8BIT *path*)

Gets the Physical Layer Pipe to be acquired.

Parameters

<i>path</i>	the tuner path to query
-------------	-------------------------

Returns

Physical Layer Pipe to be acquired

4.76.2.21 U16BIT STB_TuneGetPulsePosition (U8BIT *path*)

Returns the current pulse position.

Parameters

<i>path</i>	tuner path
-------------	------------

Returns

Current puls position

4.76.2.22 U8BIT STB_TuneGetSignalStrength (U8BIT *path*)

Returns the current signal strength.

Parameters

<i>path</i>	the tuner path to query
-------------	-------------------------

Returns

the signal strength as percentage of maximum (0-100)

4.76.2.23 E_STB_TUNE_SIGNAL_TYPE STB_TuneGetSignalType (U8BIT *path*)

Gets the signal type of the tuner path.

Parameters

<i>path</i>	the tuner path to configure
-------------	-----------------------------

Returns

the tuner signal type

4.76.2.24 E_STB_TUNE_SYSTEM_TYPE STB_TuneGetSupportedSystemType (U8BIT *path*)

Returns the system type supported by the path. This function differs from STB_TuneGetSystemType which only returns T2 or S2 if the tuner is currently performing T2 or S2 operations.

Parameters

<i>path</i>	the tuner path to query
-------------	-------------------------

Returns

(E_STB_TUNE_SYSTEM_TYPE) the system type supported by this path. TUNE_SYSTEM_TYPE_DVBT2 means both DVBT and DVBT2 are supported, TUNE_SYSTEM_TYPE_DVBS2 means both DVBS and DVBS2 are supported

4.76.2.25 E_STB_TUNE_SYSTEM_TYPE STB_TuneGetSystemType (U8BIT *path*)

Returns the signal type as set by STB_TuneSetTerrType or as re-written by the driver.

Parameters

<i>path</i>	the tuner path to query
-------------	-------------------------

Returns

Signal type.

4.76.2.26 void STB_TuneInitialise (U8BIT *paths*)

Initialises the tuner component.

Parameters

<i>paths</i>	number of tuning paths to initialise
--------------	--------------------------------------

4.76.2.27 void STB_TuneRestartTuner (U8BIT *path*)

Restarts tuner and attempts to lock to signal in StartTuner call.

Parameters

<i>path</i>	the tuner path to restart
-------------	---------------------------

4.76.2.28 U8BIT STB_TuneSatGetCarrierStrength (U8BIT *path*, U32BIT *freq*)

Returns the carrier signal strength as a percentage.

Parameters

<i>path</i>	tuner path
<i>freq</i>	carrier frequency

Returns

Strength as a percentage

4.76.2.29 void STB_TuneSendDISEQCMessage (U8BIT *path*, U8BIT * *data*, U8BIT *size*)

Sends the DisEqc message.

Parameters

<i>path</i>	- tuner path
<i>data</i>	- message data
<i>size</i>	- number of bytes in message data

4.76.2.30 void STB_TuneSet12VSwitch (U8BIT *path*, BOOLEAN *state*)

Sets the 12V switch for the given tuner.

Parameters

<i>path</i>	tuner path
<i>state</i>	TRUE for on

4.76.2.31 void STB_TuneSet22kState (U8BIT *path*, BOOLEAN *state*)

Turns the 22 kHz tone on or off.

Parameters

<i>path</i>	tuner path
<i>state</i>	TRUE to turn the tone on, FALSE to turn it off

4.76.2.32 void STB_TuneSetLNBfrequencies (U8BIT *path*, U16BIT *low_freq_mhz*, U16BIT *high_freq_mhz*)

Gets the Physical Layer Pipe to be acquired.

Parameters

<i>path</i>	the tuner path to query
-------------	-------------------------

Returns

Physical Layer Pipe to be acquired

4.76.2.33 void **STB_TuneSetLNBVoltage** (U8BIT *path*, E_STB_TUNE_LNB_VOLTAGE *voltage*)

Sets the LNB voltage for the given tuner.

Parameters

<i>path</i>	tuner path
<i>voltage</i>	voltage setting

4.76.2.34 void **STB_TuneSetModulation** (U8BIT *path*, E_STB_TUNE_MODULATION *modulation*)

Sets the type of modulation for the specified tuner.

Parameters

<i>path</i>	tuner path
<i>modulation</i>	type of modulation

4.76.2.35 void **STB_TuneSetPLP** (U8BIT *path*, U8BIT *plp*)

Sets the Physical Layer Pipe to be acquired.

Parameters

<i>path</i>	the tuner path to set up
<i>plp</i>	Physical Layer Pipe to be acquired

4.76.2.36 void **STB_TuneSetPulseLimitEast** (U8BIT *path*, U16BIT *count*)

Sets the pulse limit for the east.

Parameters

<i>path</i>	tuner path
<i>count</i>	east limit count

4.76.2.37 void **STB_TuneSetPulseLimitWest** (U8BIT *path*, U16BIT *count*)

Sets the pulse limit for the west.

Parameters

<i>path</i>	tuner path
<i>count</i>	west limit count

4.76.2.38 void **STB_TuneSetSystemType** (U8BIT *path*, E_STB_TUNE_SYSTEM_TYPE *type*)

Set the demodulator's signal type. This function must be called before each call to STB_TuneStartTuner in a dvb-t2 system and never in a dvb-t system.

Parameters

<i>U8BIT</i>	path - the tuner path to set up
<i>E_STB_TUNE_TERR_TYPE</i>	type: TUNE_TERR_TYPE_DVBT, TUNE_TERR_TYPE_DVBT2 or TUNE_TERR_TYPE_UNKNOWN. When the signal type has been set to TUNE_TERR_TYPE_UNKNOWN, a call to STB_TuneStartTuner will force the driver to try with DVB-T first, and if no signal is found, with DVB-T2. When a signal has been found, STB_TuneGetTerrType will return the actual signal type.

4.76.2.39 void **STB_TuneStartTuner** (U8BIT *path*, U32BIT *freq*, U32BIT *srate*, E_STB_TUNE_FEC *fec*, S8BIT *freq_off*, E_STB_TUNE_TMODE *tmode*, E_STB_TUNE_TBWIDTH *tbwidth*, E_STB_TUNE_CMODE *cmode*, E_STB_TUNE_ANALOG_VIDEO_TYPE *anlg_vtype*)

Starts the tuner, it will then attempt to lock specified signal. Unrequired parameters can be passed as 0 (zero)

Parameters

<i>path</i>	the tuner path to start
<i>freq</i>	the frequency to tune to
<i>srate</i>	the symbol rate to lock
<i>fec</i>	The forward error correction rate
<i>freq_off</i>	The frequency offset to use
<i>tmode</i>	The COFDM mode
<i>tbwidth</i>	The signal bandwidth
<i>cmode</i>	The QAM mode
<i>anlg_vtype</i>	The type of video for analogue tuner

4.76.2.40 void **STB_TuneStopTuner** (U8BIT *path*)

Stops any locking attempt, or unlocks if locked.

Parameters

<i>path</i>	the tuner path to stop
-------------	------------------------

4.77 platform/inc/stbhwupg.h File Reference

Functions for writing upgrade modules to non volatile memory.

Functions

- void [STB_UPGInitialise](#) (void)
Initialisation of the necessary structures.
- BOOLEAN [STB_UPGStart](#) (U8BIT image_type, U8BIT *filename)
Specifies the file path that STB_UPWrite would write to and STB_UPGRead would read from when performing upgrade. If this function is not called, a default file name will be used.
- BOOLEAN [STB_UPGWrite](#) (U8BIT image_type, U32BIT offset, U32BIT size, U8-BIT *buffer)
Writes size bytes to the upgrade storage area specified by image_type.
- BOOLEAN [STB_UPGRead](#) (U8BIT image_type, U32BIT offset, U32BIT size, U8-BIT *buffer)
Read size bytes from the upgrade storage area specified by image_type.
- BOOLEAN [STB_UPGFinish](#) (U8BIT image_type, BOOLEAN upgrade_successful)
Finalises the upgrade performing all the required actions needed when all the upgrade data have been written or the upgrade process failed.
- U32BIT [STB_UPGGetApplicationSize](#) (void)
Returns the size of the NVM area available for the application and all the upgradable modules. In the last part of this area Intellibyte loader places some version information.
- U32BIT [STB_UPGGetApplicationOffset](#) (void)
Returns the application's offset inside its area.

4.77.1 Detailed Description

Functions for writing upgrade modules to non volatile memory.

Date

28/10/2009

Author

Sergio Panseri

4.77.2 Function Documentation

4.77.2.1 BOOLEAN [STB_UPGFinish](#) (U8BIT image_type, BOOLEAN upgrade_successful)

Finalises the upgrade performing all the required actions needed when all the upgrade data have been written or the upgrade process failed.

Parameters

<i>image_type</i>	type of image to be read, this is meaningful for the platform code only, as each platform may have different upgradable modules (kernel, application, kernel modules...). The middleware passes this parameter as it finds it in the upgrade stream without interpretation.
<i>upgrade_successful</i>	TRUE if the upgrade data has been successfully written, FALSE otherwise

Returns

TRUE if the required actions have been performed successfully, FALSE otherwise

4.77.2.2 U32BIT STB_UPGGetApplicationOffset (void)

Returns the application's offset inside its area.

Returns

Application offset.

4.77.2.3 U32BIT STB_UPGGetApplicationSize (void)

Returns the size of the NVM area available for the application and all the upgradable modules. In the last part of this area Intellibyte loader places some version information.

Returns

Size of the application area.

4.77.2.4 BOOLEAN STB_UPGRead (U8BIT *image_type*, U32BIT *offset*, U32BIT *size*, U8BIT * *buffer*)

Read size bytes from the upgrade storage area specified by *image_type*.

Parameters

<i>image_type</i>	type of image to be read, this is meaningful for the platform code only, as each platform may have different upgradable modules (kernel, application, kernel modules...). The middleware passes this parameter as it finds it in the upgrade stream without interpretation.
<i>offset</i>	offset inside the specified area where to read from
<i>size</i>	number of bytes to read
<i>buffer</i>	buffer where to return the read data.

Returns

TRUE if successful, FALSE otherwise

4.77.2.5 **BOOLEAN STB_UPGStart (U8BIT *image_type*, U8BIT * *filename*)**

Specifies the file path that STB_UPWrite would write to and STB_UPGRead would read from when performing upgrade. If this function is not called, a default file name will be used.

Parameters

<i>image_type</i>	type of image to be written, this is meaningful for the platform code only, as each platform may have different upgradable modules (kernel, application, kernel modules...). The middleware passes this parameter as it finds it in the upgrade stream without interpretation.
<i>filename</i>	

Returns

TRUE if successful, FALSE otherwise

4.77.2.6 **BOOLEAN STB_UPGWrite (U8BIT *image_type*, U32BIT *offset*, U32BIT *size*, U8BIT * *buffer*)**

Writes size bytes to the upgrade storage area specified by image_type.

Parameters

<i>image_type</i>	type of image to be written, this is meaningful for the platform code only, as each platform may have different upgradable modules (kernel, application, kernel modules...). The middleware passes this parameter as it finds it in the upgrade stream without interpretation.
<i>offset</i>	offset inside the specified area where to write to
<i>size</i>	number of bytes to write
<i>buffer</i>	buffer containing the data to be written

Returns

TRUE if successful, FALSE otherwise

4.78 platform/inc/stbpvrpr.h File Reference

Header file - macros and function prototypes for public use.

#include "stbhwav.h" #include "stbhwdmx.h" Include dependency graph for stbpvrpr.h: This graph shows which files directly or indirectly include this file:

Data Structures

- struct [s_pvr_pid_info](#)

Typedefs

- typedef enum e_stb_pvr_start_mode **E_STB_PVR_START_MODE**
- typedef enum e_stb_pvr_play_mode **E_STB_PVR_PLAY_MODE**
- typedef struct s_pvr_pid_info **S_PVR_PID_INFO**

Enumerations

- enum **E_PVR_PID_TYPE** { **PVR_PID_TYPE_VIDEO**, **PVR_PID_TYPE_AUDIO**, **PVR_PID_TYPE_PCR**, **PVR_PID_TYPE_SUBTITLES**, **PVR_PID_TYPE_SECTION**, **PVR_PID_TYPE_TELETEXT** }
- enum **e_stb_pvr_start_mode** { **START_RUNNING** = 0, **START_PAUSED** = 1, **START_AVSYN** = 2, **START_EXPORT_PES** = 3, **START_IMPORT_TS** = 4, **START_IMPORT_PES** = 5 }
- enum **e_stb_pvr_play_mode** { **PLAY_NO_TRICK** = 0, **PLAY_TRICK_FORWARD** = 1, **PLAY_TRICK_REVERSE** = 2, **PLAY_TRICK_FRAME** = 3 }

Functions

- U8BIT **STB_PVRInitPlayback** (U8BIT num_audio_decoders, U8BIT num_video_decoders)
Initialisation for playback.
- U8BIT **STB_PVRInitRecording** (U8BIT num_tuners)
Initialisation for recording.
- void **STB_PVRSetPlayStartMode** (U8BIT audio_decoder, U8BIT video_decoder, **E_STB_PVR_START_MODE** mode)
Set startup mode for playback.
- void **STB_PVRPlayHasVideo** (U8BIT audio_decoder, U8BIT video_decoder, BOOLEAN has_video)
Informs the platform whether there's video in the file to be played. Should be called before playback is started.
- void **STB_PVRSetPlaybackNotifyTime** (U8BIT audio_decoder, U8BIT video_decoder, U32BIT notify_time)
Sets the time the next notification event should be sent during playback. This is required for CI+, but may also be used for other purposes.
- BOOLEAN **STB_PVRPlayStart** (U16BIT disk_id, U8BIT audio_decoder, U8BIT video_decoder, U8BIT demux, U8BIT *basename)
Starts playback.
- BOOLEAN **STB_PVRPlayChangeAudio** (U8BIT audio_decoder, U8BIT video_decoder, U16BIT pid, U8BIT codec)
Changes the main audio PID being decoded during playback. This can be used to switch between main audio and broadcaster mix AD.
- BOOLEAN **STB_PVRIsPlayStarted** (U8BIT audio_decoder, U8BIT video_decoder)
Returns status of playback with the given decoders.

- **BOOLEAN STB_PVRPlaySetPosition** (U8BIT audio_decoder, U8BIT video_decoder, U32BIT position_in_seconds)
Sets the playback position after playback has started (i.e. jump to bookmark)
- **void STB_PVRPlayStop** (U8BIT audio_decoder, U8BIT video_decoder)
Stops playback.
- **void STB_PVRPlayEnabled** (U8BIT audio_decoder, U8BIT video_decoder, **BOOLEAN** *video, **BOOLEAN** *audio)
Returns whether audio and video playback has been started.
- **void STB_PVRPlaySetRetentionLimit** (U8BIT audio_decoder, U8BIT video_decoder, U32BIT retention_limit, U16BIT rec_date, U8BIT rec_hour, U8BIT rec_min)
Set the retention limit for the playback. This function is used for CI+.
- **void STB_PVRSetRecordStartMode** (U8BIT rec_index, **E_STB_PVR_START_MODE** mode, U32BIT param)
Sets the startup mode for a recording. This function should be called before the recording is started and is used to when pausing live TV in which case the additional param defines the length of the pause buffer to be used, in seconds.
- **BOOLEAN STB_PVRRecordStart** (U16BIT disk_id, U8BIT rec_index, U8BIT *basename, U16BIT num_pids, **S_PVR_PID_INFO** *pid_array)
- **BOOLEAN STB_PVRRecordPause** (U8BIT rec_index)
Pauses a recording currently taking place.
- **BOOLEAN STB_PVRRecordResume** (U8BIT rec_index)
Resumes a paused recording.
- **BOOLEAN STB_PVRRecordChangePids** (U8BIT rec_index, U16BIT num_pids, **S_PVR_PID_INFO** *pids_array)
Changes the PIDs while recording.
- **void STB_PVRRecordStop** (U8BIT rec_index)
Stops a recording.
- **BOOLEAN STB_PVRIsRecordStarted** (U8BIT rec_index)
Returns whether recording has been started.
- **void STB_PVRRecordEnabled** (U8BIT rec_index, **BOOLEAN** *video, **BOOLEAN** *audio)
Returns status of audio/video recording.
- **void STB_PVRSetRecordEncryptionKey** (U8BIT rec_index, **BOOLEAN** state, U8BIT *key, U8BIT *iv, U32BIT key_len)
Enables or disables encryption and sets the encryption key to be used.
- **void STB_PVRSetPlaybackDecryptionKey** (U8BIT audio_decoder, U8BIT video_decoder, **BOOLEAN** state, U8BIT *key, U8BIT *iv, U32BIT key_len)
Enables and sets the key that will be used to decrypt an encrypted recording during playback.
- **void STB_PVRPlayTrickMode** (U8BIT audio_decoder, U8BIT video_decoder, **E_STB_PVR_PLAY_MODE** mode, S16BIT speed)
Sets trick mode during playback.
- **void STB_PVRSaveFrame** (U8BIT audio_decoder, U8BIT video_decoder)
Unused function.

- S16BIT [STB_PVRGetPlaySpeed](#) (U8BIT audio_decoder, U8BIT video_decoder)
Returns the current playback speed.
- BOOLEAN [STB_PVRSetPlaySpeed](#) (U8BIT audio_decoder, U8BIT video_decoder, S16BIT speed)
Set the play speed for the specified decoder.
- BOOLEAN [STB_PVRIsValidRecording](#) (U16BIT disk_id, U8BIT *basename)
Checks whether any of the files already exist that would be created by a recording with the given base filename.
- BOOLEAN [STB_PVRCanBeUsedForRecording](#) (U16BIT disk_id, U8BIT *basename)
Checks whether any of the files already exist that would be created by a recording with the given base filename.
- BOOLEAN [STB_PVRDeleteRecording](#) (U16BIT disk_id, U8BIT *basename)
Deletes any files associated with the given base filename that were created as a result of the recording being performed.
- BOOLEAN [STB_PVRGetRecordingSize](#) (U16BIT disk_id, U8BIT *basename, - U32BIT *rec_size_kb)
Returns the size in kilobytes of the recording defined by the given base filename.
- BOOLEAN [STB_PVRGetElapsedTime](#) (U8BIT audio_decoder, U8BIT video_decoder, U8BIT *elapsed_hours, U8BIT *elapsed_mins, U8BIT *elapsed_secs)
Returns the elapsed playback time in hours, mins & secs.
- U8BIT [STB_PVRAcquireRecorderIndex](#) (U8BIT tuner, U8BIT demux)
Acquires an index to be used to reference a recording.
- void [STB_PVRReleaseRecorderIndex](#) (U8BIT rec_index)
Releases a recording index when no longer needed.
- BOOLEAN [STB_PVRApplyDescramblerKey](#) (U8BIT rec_index, E_STB_DMX_D-ESC_TYPE desc_type, E_STB_DMX_DESC_KEY_PARITY parity, U8BIT *key, U8BIT *iv)
Called to apply the given descrambler key to the PID data being recorded. This function may be called before the recording has actually started.

4.78.1 Detailed Description

Header file - macros and function prototypes for public use.

Date

07/02/2003

4.78.2 Function Documentation

4.78.2.1 U8BIT STB_PVRAcquireRecorderIndex (U8BIT tuner, U8BIT demux)

Acquires an index to be used to reference a recording.

Parameters

<i>tuner</i>	tuner to be used for the recording
<i>demux</i>	demux to be used for the recording

Returns

recording index, 255 if none available

4.78.2.2 **BOOLEAN STB_PVRApplyDescramblerKey** (U8BIT *rec_index*,
E_STB_DMX_DESC_TYPE *desc_type*, E_STB_DMX_DESC_KEY_PARITY *parity*, U8BIT *
key, U8BIT * *iv*)

Called to apply the given descrambler key to the PID data being recorded. This function may be called before the recording has actually started.

Parameters

<i>rec_index</i>	recording index
<i>desc_type</i>	descrambler type
<i>parity</i>	key parity
<i>key</i>	key data
<i>iv</i>	provides an initialisation vector data, if required for the descrambler type

Returns

TRUE if successful, FALSE otherwise

4.78.2.3 **BOOLEAN STB_PVRCanBeUsedForRecording** (U16BIT *disk_id*, U8BIT *
basename)

Checks whether any of the files already exist that would be created by a recording with the given base filename.

Parameters

<i>disk_id</i>	disk to be checked
<i>basename</i>	base filename to be used for a recording

Returns

TRUE if none of the files exist, FALSE otherwise

4.78.2.4 **BOOLEAN STB_PVRDeleteRecording** (U16BIT *disk_id*, U8BIT * *basename*)

Deletes any files associated with the given base filename that were created as a result of the recording being performed.

Parameters

<i>disk_id</i>	disk containing the recording to be deleted
<i>basename</i>	base filename used for the recording

Returns

TRUE if successful, FALSE otherwise

4.78.2.5 **BOOLEAN STB_PVRGetElapsedTime (U8BIT *audio_decoder*, U8BIT *video_decoder*, U8BIT * *elapsed_hours*, U8BIT * *elapsed_mins*, U8BIT * *elapsed_secs*)**

Returns the elapsed playback time in hours, mins & secs.

Parameters

<i>audio_decoder</i>	audio decoder being used for playback
<i>video_decoder</i>	video decoder being used for playback
<i>elapsed_hours</i>	current number of hours into the playback
<i>elapsed_mins</i>	current number of minutes into the playback
<i>elapsed_secs</i>	current number of seconds into the playback

Returns

TRUE if the info has been successfully gathered

4.78.2.6 **S16BIT STB_PVRGetPlaySpeed (U8BIT *audio_decoder*, U8BIT *video_decoder*)**

Returns the current playback speed.

Parameters

<i>audio_decoder</i>	audio decoder being used for playback
<i>video_decoder</i>	video decoder being used for playback

Returns

current playback speed as a percentage

4.78.2.7 BOOLEAN STB_PVRGetRecordingSize (U16BIT *disk_id*, U8BIT * *basename*, U32BIT * *rec_size_kb*)

Returns the size in kilobytes of the recording defined by the given base filename.

Parameters

<i>disk_id</i>	disk containing the recording to be queried
<i>basename</i>	base filename of recording to get info about
<i>rec_size_kb</i>	returned size of recording in kilobytes

Returns

TRUE if the information is successfully gathered

4.78.2.8 U8BIT STB_PVRInitPlayback (U8BIT *num_audio_decoders*, U8BIT *num_video_decoders*)

Initialisation for playback.

Parameters

<i>num_audio_decoders</i>	number of audio decoders available
<i>num_video_decoders</i>	number of video decoders available

Returns

Number of players, 0 if unsuccessful or unsupported

4.78.2.9 U8BIT STB_PVRInitRecording (U8BIT *num_tuners*)

Initialisation for recording.

Parameters

<i>num_tuners</i>	number of tuners available for recording
-------------------	--

Returns

Number of recorders, 0 if unsuccessful or unsupported

4.78.2.10 **BOOLEAN STB_PVRIsPlayStarted (U8BIT *audio_decoder*, U8BIT *video_decoder*)**

Returns status of playback with the given decoders.

Parameters

<i>audio_decoder</i>	audio decoder being used for playback
<i>video_decoder</i>	video decoder being used for playback

Returns

TRUE if playback is in progress with the given decoders

4.78.2.11 **BOOLEAN STB_PVRIsRecordStarted (U8BIT *rec_index*)**

Returns whether recording has been started.

Parameters

<i>rec_index</i>	recording index being queried
------------------	-------------------------------

Returns

TRUE if recording has been started

4.78.2.12 **BOOLEAN STB_PVRIsValidRecording (U16BIT *disk_id*, U8BIT * *basename*)**

Checks whether any of the files already exist that would be created by a recording with the given base filename.

Parameters

<i>disk_id</i>	disk to be checked
<i>basename</i>	base filename to be used for a recording

Returns

TRUE if none of the files exist, FALSE otherwise

4.78.2.13 **BOOLEAN STB_PVRPlayChangeAudio** (U8BIT *audio_decoder*, U8BIT *video_decoder*, U16BIT *pid*, U8BIT *codec*)

Changes the main audio PID being decoded during playback. This can be used to switch between main audio and broadcaster mix AD.

Parameters

<i>audio_decoder</i>	- audio decoder for playback
<i>video_decoder</i>	- video decoder for playback
<i>pid</i>	- new audio PID to decode
<i>codec</i>	- new audio codec

Returns

TRUE if the PID is changed successfully, FALSE otherwise

4.78.2.14 **void STB_PVRPlayEnabled** (U8BIT *audio_decoder*, U8BIT *video_decoder*, BOOLEAN * *video*, BOOLEAN * *audio*)

Returns whether audio and video playback has been started.

Parameters

<i>audio_decoder</i>	audio decoder being used for playback
<i>video_decoder</i>	video decoder being used for playback
<i>video</i>	returned as TRUE if video is being decoded
<i>audio</i>	returned as TRUE if audio is being decoded

4.78.2.15 **void STB_PVRPlayHasVideo** (U8BIT *audio_decoder*, U8BIT *video_decoder*, BOOLEAN *has_video*)

Informs the platform whether there's video in the file to be played. Should be called before playback is started.

Parameters

<i>audio_decoder</i>	audio decoder being used for playback
<i>video_decoder</i>	video decoder being used for playback
<i>has_video</i>	TRUE if the recording contains video, FALSE otherwise

4.78.2.16 **BOOLEAN STB_PVRPlaySetPosition** (U8BIT *audio_decoder*, U8BIT *video_decoder*, U32BIT *position_in_seconds*)

Sets the playback position after playback has started (i.e. jump to bookmark)

Parameters

<i>audio_decoder</i>	audio decoder being used for playback
<i>video_decoder</i>	video decoder being used for playback
<i>position_in_seconds</i>	position to jump to in the recording in seconds from the beginning

Returns

TRUE if position is set successfully, FALSE otherwise

4.78.2.17 **void STB_PVRPlaySetRetentionLimit** (U8BIT *audio_decoder*, U8BIT *video_decoder*, U32BIT *retention_limit*, U16BIT *rec_date*, U8BIT *rec_hour*, U8BIT *rec_min*)

Set the retention limit for the playback. This function is used for CI+.

Parameters

<i>audio_decoder</i>	audio decoder being used for playback
<i>video_decoder</i>	video decoder being used for playback
<i>retention_limit</i>	Retention limit in minutes
<i>rec_data</i>	data when the recording was taken
<i>rec_hour</i>	hour when the recording was taken
<i>rec_min</i>	minute when the recording was taken

4.78.2.18 **BOOLEAN STB_PVRPlayStart** (U16BIT *disk_id*, U8BIT *audio_decoder*, U8BIT *video_decoder*, U8BIT *demux*, U8BIT * *basename*)

Starts playback.

Parameters

<i>disk_id</i>	disk containing the recording to be played
<i>audio_decoder</i>	audio decoder to be used for playback
<i>video_decoder</i>	video decoder to be used for playback
<i>demux</i>	demux to be used for playback
<i>basename</i>	basename of the recording to be played

Returns

TRUE if playback is started, FALSE otherwise

4.78.2.19 void STB_PVRPlayStop (U8BIT *audio_decoder*, U8BIT *video_decoder*)

Stops playback.

Parameters

<i>audio_decoder</i>	audio decoder being used for playback
<i>video_decoder</i>	video decoder being used for playback

4.78.2.20 void STB_PVRPlayTrickMode (U8BIT *audio_decoder*, U8BIT *video_decoder*, E_STB_PVR_PLAY_MODE *mode*, S16BIT *speed*)

Sets trick mode during playback.

Parameters

<i>audio_decoder</i>	audio decoder being used for playback
<i>video_decoder</i>	video decoder being used for playback
<i>mode</i>	trick mode to be used
<i>speed</i>	playback speed to be used as a percentage (100% = normal playback)

4.78.2.21 BOOLEAN STB_PVRRecordChangePids (U8BIT *rec_index*, U16BIT *num_pids*, S_PVR_PID_INFO * *pids_array*)

Changes the PIDs while recording.

Parameters

<i>rec_index</i>	current recording index to be updated
<i>num_pids</i>	number of PIDs in PID array
<i>pid_array</i>	new PID list to be recorded

Returns

TRUE if the PIDs have been successfully changed, FALSE otherwise

4.78.2.22 void STB_PVRRecordEnabled (U8BIT *rec_index*, BOOLEAN * *video*, BOOLEAN * *audio*)

Returns status of audio/video recording.

Parameters

<i>rec_index</i>	recording index being used for recording
<i>video</i>	pointer to a boolean value that indicates whether the video data is being recorded
<i>audio</i>	pointer to a boolean value that indicates whether the audio data is being recorded

4.78.2.23 **BOOLEAN STB_PVRRecordPause (U8BIT *rec_index*)**

Pauses a recording currently taking place.

Parameters

<i>rec_index</i>	recording index
------------------	-----------------

Returns

TRUE if the recording is successfully paused, FALSE otherwise

4.78.2.24 **BOOLEAN STB_PVRRecordResume (U8BIT *rec_index*)**

Resumes a paused recording.

Parameters

<i>rec_index</i>	recording index
------------------	-----------------

Returns

TRUE if the recording is successfully resumed, FALSE otherwise

4.78.2.25 **void STB_PVRRecordStop (U8BIT *rec_index*)**

Stops a recording.

Parameters

<i>rec_index</i>	recording index
------------------	-----------------

4.78.2.26 **void STB_PVRReleaseRecorderIndex (U8BIT *rec_index*)**

Releases a recording index when no longer needed.

Parameters

<i>rec_index</i>	recoding index
------------------	----------------

4.78.2.27 void STB_PVRSaveFrame (U8BIT *audio_decoder*, U8BIT *video_decoder*)

Unused function.

Parameters

<i>audio_decoder</i>	audio decoder being used for playback
<i>video_decoder</i>	video decoder being used for playback

4.78.2.28 void STB_PVRSetPlaybackDecryptionKey (U8BIT *audio_decoder*, U8BIT *video_decoder*, BOOLEAN *state*, U8BIT * *key*, U8BIT * *iv*, U32BIT *key_len*)

Enables and sets the key that will be used to decrypt an encrypted recording during playback.

Parameters

<i>audio_decoder</i>	audio decoder used for playback
<i>video_decoder</i>	video decoder used for playback
<i>state</i>	whether decryption is enabled or disabled
<i>key</i>	decryption key, ignored if state is FALSE
<i>iv</i>	initialisation vector, ignored if state is FALSE
<i>key_len</i>	length of decryption key, ignored if state is FALSE

4.78.2.29 void STB_PVRSetPlaybackNotifyTime (U8BIT *audio_decoder*, U8BIT *video_decoder*, U32BIT *notify_time*)

Sets the time the next notification event should be sent during playback. This is required for CI+, but may also be used for other purposes.

Parameters

<i>audio_decoder</i>	audio decoder being used for playback
<i>video_decoder</i>	video decoder being used for playback
<i>notify_time</i>	time in seconds the next notification event is to be sent

4.78.2.30 BOOLEAN STB_PVRSetPlaySpeed (U8BIT *audio_decoder*, U8BIT *video_decoder*, S16BIT *speed*)

Set the play speed for the specified decoder.

Parameters

<i>audio_ - decoder</i>	audio decoder being used for playback
<i>video_ - decoder</i>	video decoder being used for playback
<i>speed</i>	Play speed as a percentage (i.e 100% = normal playback)

Returns

TRUE if successful

4.78.2.31 void **STB_PVRSetPlayStartMode** (U8BIT *audio_decoder*, U8BIT *video_decoder*, E_STB_PVR_START_MODE *mode*)

Set startup mode for playback.

Parameters

<i>audio_ - decoder</i>	audio decoder being used for playback
<i>video_ - decoder</i>	video decoder being used for playback
<i>mode</i>	playback startup mode

4.78.2.32 void **STB_PVRSetRecordEncryptionKey** (U8BIT *rec_index*, BOOLEAN *state*, U8BIT * *key*, U8BIT * *iv*, U32BIT *key_len*)

Enables or disables encryption and sets the encryption key to be used.

Parameters

<i>rec_index</i>	recording index to be set
<i>state</i>	whether encryption is enabled or disabled
<i>key</i>	encryption key, ignored if state is FALSE
<i>iv</i>	initialisation vector, ignored if state is FALSE
<i>key_len</i>	length of encryption key, ignored if state is FALSE

4.78.2.33 void **STB_PVRSetRecordStartMode** (U8BIT *rec_index*, E_STB_PVR_START_MODE *mode*, U32BIT *param*)

Sets the startup mode for a recording. This function should be called before the recording is started and is used to when pausing live TV in which case the additional param defines the length of the pause buffer to be used, in seconds.

Parameters

<i>rec_index</i>	recording index to be used for the recording
<i>mode</i>	startup mode

<i>param</i>	additional parameter linked to the mode. When pausing live TV, this is the length of the pause buffer, in seconds.
--------------	--

4.79 platform/inc/stbver.h File Reference

Header file - Function prototypes for version numbers.

Data Structures

- struct [APP_SW_VER_STRUCT](#)
- struct [T_BOOT_SW_VER_NUMBER](#)

Functions

- void [STB_OSGetAppSwVersionNum](#) ([APP_SW_VER_STRUCT](#) *vptr)
Get Software Version Number for Application.
- void [STB_OSGetBootSwVersionNum](#) ([T_BOOT_SW_VER_NUMBER](#) *vptr)
Get Software Version Number for Boot Loader.

4.79.1 Detailed Description

Header file - Function prototypes for version numbers.

Date

11/12/2008