

BACnetDemo

CS Lab GmbH | Römerstraße 15 | 47809 Krefeld | Germany | © 2016

Generated by Doxygen 1.8.11

Contents

- 1 BACnet API demo application** **1**

- 2 Data Structure Index** **3**
 - 2.1 Data Structures 3

- 3 File Index** **5**
 - 3.1 File List 5

- 4 Data Structure Documentation** **7**
 - 4.1 user_analog_input_t Struct Reference 7
 - 4.1.1 Detailed Description 7
 - 4.1.2 Field Documentation 7
 - 4.1.2.1 adc_sum 7
 - 4.1.2.2 adc_sum_count 7
 - 4.1.2.3 adc_value 7
 - 4.1.2.4 bacnet_value 8
 - 4.1.2.5 resolution 8
 - 4.2 user_binary_input_t Struct Reference 8
 - 4.2.1 Detailed Description 8
 - 4.2.2 Field Documentation 8
 - 4.2.2.1 button 8
 - 4.2.2.2 state 8
 - 4.3 user_button_t Struct Reference 8
 - 4.3.1 Detailed Description 9
 - 4.3.2 Field Documentation 9

4.3.2.1	level	9
4.3.2.2	pin	9
4.4	user_record_t Struct Reference	9
4.4.1	Detailed Description	9
4.4.2	Field Documentation	9
4.4.2.1	data	9
4.5	user_trendlog_t Struct Reference	10
4.5.1	Detailed Description	10
4.5.2	Field Documentation	10
4.5.2.1	count	10
4.5.2.2	pos	10
4.5.2.3	records	10
5	File Documentation	11
5.1	src/demo/adc.c File Reference	11
5.1.1	Macro Definition Documentation	11
5.1.1.1	ADC_REGISTER_COUNT	11
5.1.2	Function Documentation	12
5.1.2.1	AdcCompleteCallback(adc_callback_args_t *p_args)	12
5.1.2.2	AdcGetValue(uint8_t index, uint16_t *p_value)	12
5.1.2.3	AdcInit(void)	12
5.1.3	Variable Documentation	12
5.1.3.1	g_adc	12
5.1.3.2	g_adc_data	12
5.1.3.3	g_adc_register	13
5.1.3.4	g_adc_started	13
5.2	src/demo/adc.h File Reference	13
5.2.1	Function Documentation	13
5.2.1.1	AdcCompleteCallback(adc_callback_args_t *p_args)	13
5.2.1.2	AdcGetValue(uint8_t index, uint16_t *p_value)	13
5.2.1.3	AdcInit(void)	14

5.3	src/demo/bacnetdemo.c File Reference	14
5.3.1	Macro Definition Documentation	15
5.3.1.1	BACNET_DEMO_WITH_FOREIGN_DEVICE	15
5.3.2	Function Documentation	15
5.3.2.1	bacnet_init(BACNET_TIME_STAMP *p_start_time)	15
5.3.2.2	bacnet_reset(void)	16
5.3.2.3	bacnet_storage_delete(void)	16
5.3.2.4	bacnet_storage_read(void)	16
5.3.2.5	bacnet_storage_write(void)	16
5.3.2.6	bacnet_trend(BACNET_READ_RANGE_INFO *p_service_info, BAC_UINT *p_↔ _entries, BAC_BOOLEAN *p_first, BAC_BOOLEAN *p_last, BAC_BOOLEAN ↔ N *p_more)	16
5.3.2.7	bacnet_write(BACNET_INST_NUMBER device_id, BACNET_OBJECT_TYP↔ E obj_type, BACNET_INST_NUMBER inst_number, BACNET_PROPERTY_ID property_id, BACNET_ARRAY_INDEX index, const BACNET_PROPERTY_C↔ ONTENTS *p_property_contents, BACNET_BOOLEAN *p_property_persistent)	17
5.3.2.8	BacnetDemoMain(void)	17
5.3.3	Variable Documentation	18
5.3.3.1	g_bacnet_common_cfg	18
5.3.3.2	g_bacnet_common_ctrl	18
5.3.3.3	g_bacnet_common_instance	18
5.3.3.4	g_bacnet_data_link_cfg	18
5.3.3.5	g_bacnet_data_link_ctrl	19
5.3.3.6	g_bacnet_data_link_ip_ext_cfg	19
5.3.3.7	g_bacnet_instance	19
5.3.3.8	g_log_record	19
5.4	src/demo/bacnetdemo.h File Reference	19
5.4.1	Enumeration Type Documentation	20
5.4.1.1	demo_obj_ids_t	20
5.4.2	Function Documentation	20
5.4.2.1	BacnetDemoMain(void)	20
5.5	src/demo/definitions.h File Reference	21
5.5.1	Macro Definition Documentation	22

5.5.1.1	DEV_DATALINK_COUNT	22
5.5.1.2	DEV_DATALINK_ETH_IP1	22
5.5.1.3	DEV_DATALINK_ETH_IP2	22
5.5.1.4	DEV_DATALINK_ETH_IP3	22
5.5.1.5	DEV_DATALINK_ETH_IP4	22
5.5.1.6	DEV_DATALINK_ETH_NET_ID	22
5.5.1.7	DEV_DATALINK_ETH_PORT	22
5.5.1.8	DEV_DATALINK_ETH_PORT_ID	22
5.5.1.9	DEV_DATALINK_FOREIGN_INTERVAL	22
5.5.1.10	DEV_DATALINK_FOREIGN_IP1	22
5.5.1.11	DEV_DATALINK_FOREIGN_IP2	23
5.5.1.12	DEV_DATALINK_FOREIGN_IP3	23
5.5.1.13	DEV_DATALINK_FOREIGN_IP4	23
5.5.1.14	DEV_DATALINK_FOREIGN_PORT	23
5.5.1.15	DEV_DATALINK_MSTP_ADDRESS	23
5.5.1.16	DEV_DATALINK_MSTP_NET	23
5.5.1.17	DEV_DATALINK_MSTP_PORT	23
5.5.1.18	DEVICE_DDC_PASSWORD_INIT_STRING	23
5.5.1.19	DEVICE_REINIT_PASSWORD_INIT_STRING	23
5.5.2	Function Documentation	24
5.5.2.1	CFG_READONLY_STRING(g_server_device_dcc_password, DEVICE_DDC_PASSWORD_INIT_STRING)	24
5.5.2.2	CFG_READONLY_STRING(g_server_device_reinit_password, DEVICE_REINIT_PASSWORD_INIT_STRING)	24
5.5.2.3	CFG_SERVER_INIT(g_server, g_server_device, DEV_DATALINK_COUNT, g_device_data_links[0], 1)	24
5.5.2.4	CFG_WRITABLE_ADDRESS(g_server_device_address)	24
5.5.2.5	CFG_WRITABLE_APDU_PROPERTIES(g_server_device_apdu_properties, OBJ_DEV_MAX_APDU_SIZE, TARGET_SUPPORTED_SEGMENTATION_TYPE, TARGET_MAX_SUPPORTED_NPDU_SEGMENTS, 1, 2000, 3000, 5, 60, 5)	24
5.5.3	Variable Documentation	24
5.5.3.1	g_device_data_links	24

5.5.3.2	g_obj_array	24
5.5.3.3	g_server_device	25
5.5.3.4	g_server_device_dcc_value	25
5.6	src/demo/obj_ai.c File Reference	25
5.6.1	Function Documentation	26
5.6.1.1	ObjAllInit(void)	26
5.6.1.2	ObjAIPresentValueUpdate(BACNET_INST_NUMBER inst_number, BACNET_REAL value)	26
5.6.1.3	ObjAIReset(void)	26
5.6.2	Variable Documentation	26
5.6.2.1	g_obj_ai1_description	26
5.6.2.2	g_obj_ai1_evt_msg_txt	26
5.6.2.3	g_obj_ai1_out_of_service	27
5.6.2.4	g_obj_ai1_present_value	27
5.6.2.5	g_obj_ai1_state_flags	27
5.6.2.6	g_obj_ai2_description	27
5.6.2.7	g_obj_ai2_evt_msg_txt	27
5.6.2.8	g_obj_ai2_out_of_service	27
5.6.2.9	g_obj_ai2_present_value	27
5.6.2.10	g_obj_ai2_state_flags	27
5.6.2.11	g_obj_ai3_description	27
5.6.2.12	g_obj_ai3_evt_msg_txt	27
5.6.2.13	g_obj_ai3_out_of_service	27
5.6.2.14	g_obj_ai3_present_value	27
5.6.2.15	g_obj_ai3_state_flags	27
5.6.2.16	g_obj_ai4_description	27
5.6.2.17	g_obj_ai4_evt_msg_txt	27
5.6.2.18	g_obj_ai4_out_of_service	27
5.6.2.19	g_obj_ai4_present_value	27
5.6.2.20	g_obj_ai4_state_flags	27
5.6.2.21	g_obj_dev_id	27

5.7	src/demo/obj_ai.h File Reference	27
5.7.1	Macro Definition Documentation	28
5.7.1.1	OBJ_AI1_COV_INCREMENT_INITIAL	28
5.7.1.2	OBJ_AI1_DESCRIPTION_INIT_STRING	29
5.7.1.3	OBJ_AI1_LIMIT_DEADBAND_INITIAL	29
5.7.1.4	OBJ_AI1_LIMIT_HIGH_INITIAL	29
5.7.1.5	OBJ_AI1_LIMIT_LOW_INITIAL	29
5.7.1.6	OBJ_AI1_NAME_INIT_STRING	29
5.7.1.7	OBJ_AI1_PRESENT_VALUE_INITIAL	29
5.7.1.8	OBJ_AI1_UNIT_INITIAL	29
5.7.1.9	OBJ_AI2_COV_INCREMENT_INITIAL	29
5.7.1.10	OBJ_AI2_DESCRIPTION_INIT_STRING	29
5.7.1.11	OBJ_AI2_LIMIT_DEADBAND_INITIAL	29
5.7.1.12	OBJ_AI2_LIMIT_HIGH_INITIAL	30
5.7.1.13	OBJ_AI2_LIMIT_LOW_INITIAL	30
5.7.1.14	OBJ_AI2_NAME_INIT_STRING	30
5.7.1.15	OBJ_AI2_PRESENT_VALUE_INITIAL	30
5.7.1.16	OBJ_AI2_UNIT_INITIAL	30
5.7.1.17	OBJ_AI3_COV_INCREMENT_INITIAL	30
5.7.1.18	OBJ_AI3_DESCRIPTION_INIT_STRING	30
5.7.1.19	OBJ_AI3_LIMIT_DEADBAND_INITIAL	30
5.7.1.20	OBJ_AI3_LIMIT_HIGH_INITIAL	30
5.7.1.21	OBJ_AI3_LIMIT_LOW_INITIAL	30
5.7.1.22	OBJ_AI3_NAME_INIT_STRING	31
5.7.1.23	OBJ_AI3_PRESENT_VALUE_INITIAL	31
5.7.1.24	OBJ_AI3_UNIT_INITIAL	31
5.7.1.25	OBJ_AI4_COV_INCREMENT_INITIAL	31
5.7.1.26	OBJ_AI4_DESCRIPTION_INIT_STRING	31
5.7.1.27	OBJ_AI4_LIMIT_DEADBAND_INITIAL	31
5.7.1.28	OBJ_AI4_LIMIT_HIGH_INITIAL	31

5.7.1.29	OBJ_AI4_LIMIT_LOW_INITIAL	31
5.7.1.30	OBJ_AI4_NAME_INIT_STRING	31
5.7.1.31	OBJ_AI4_PRESENT_VALUE_INITIAL	31
5.7.1.32	OBJ_AI4_UNIT_INITIAL	32
5.7.2	Function Documentation	32
5.7.2.1	ObjAllInit(void)	32
5.7.2.2	ObjAIPresentValueUpdate(BACNET_INST_NUMBER inst_number, BACNET_REAL value)	32
5.7.2.3	ObjAIReset(void)	32
5.8	src/demo/obj_ao.c File Reference	32
5.8.1	Function Documentation	33
5.8.1.1	ObjAOInit(void)	33
5.8.1.2	ObjAOReset(void)	33
5.8.1.3	ObjAOWriteHandling(BACNET_INST_NUMBER inst_number, BACNET_PROPERTY_ID property_id, const BACNET_PROPERTY_CONTENTS *property_contents)	33
5.8.2	Variable Documentation	33
5.8.2.1	g_obj_ao1_description	34
5.8.2.2	g_obj_ao1_evt_msg_txt	34
5.8.2.3	g_obj_ao1_out_of_service	34
5.8.2.4	g_obj_ao1_present_value	34
5.8.2.5	g_obj_ao1_state_flags	34
5.9	src/demo/obj_ao.h File Reference	34
5.9.1	Macro Definition Documentation	34
5.9.1.1	OBJ_AO1_DESCRIPTION_INIT_STRING	34
5.9.1.2	OBJ_AO1_NAME_INIT_STRING	34
5.9.2	Function Documentation	35
5.9.2.1	ObjAOInit(void)	35
5.9.2.2	ObjAOReset(void)	35
5.9.2.3	ObjAOWriteHandling(BACNET_INST_NUMBER inst_number, BACNET_PROPERTY_ID property_id, const BACNET_PROPERTY_CONTENTS *property_contents)	35
5.10	src/demo/obj_bi.c File Reference	35

5.10.1	Function Documentation	36
5.10.1.1	ObjBIInit(void)	36
5.10.1.2	ObjBIPresentValueUpdate(BACNET_INST_NUMBER inst_number, BAC_BOOLEAN value)	36
5.10.1.3	ObjBIReset(void)	36
5.10.2	Variable Documentation	37
5.10.2.1	g_obj_bi1_description	37
5.10.2.2	g_obj_bi1_evt_msg_txt	37
5.10.2.3	g_obj_bi1_out_of_service	37
5.10.2.4	g_obj_bi1_present_value	37
5.10.2.5	g_obj_bi1_state_flags	37
5.10.2.6	g_obj_bi2_description	37
5.10.2.7	g_obj_bi2_evt_msg_txt	37
5.10.2.8	g_obj_bi2_out_of_service	37
5.10.2.9	g_obj_bi2_present_value	37
5.10.2.10	g_obj_bi2_state_flags	37
5.10.2.11	g_obj_dev_id	37
5.11	src/demo/obj_bi.h File Reference	37
5.11.1	Macro Definition Documentation	38
5.11.1.1	OBJ_BI1_ACTIVE_INIT_STRING	38
5.11.1.2	OBJ_BI1_DESCRIPTION_INIT_STRING	38
5.11.1.3	OBJ_BI1_INACTIVE_INIT_STRING	38
5.11.1.4	OBJ_BI1_NAME_INIT_STRING	38
5.11.1.5	OBJ_BI2_ACTIVE_INIT_STRING	38
5.11.1.6	OBJ_BI2_DESCRIPTION_INIT_STRING	38
5.11.1.7	OBJ_BI2_INACTIVE_INIT_STRING	38
5.11.1.8	OBJ_BI2_NAME_INIT_STRING	38
5.11.2	Function Documentation	38
5.11.2.1	ObjBIInit(void)	38
5.11.2.2	ObjBIPresentValueUpdate(BACNET_INST_NUMBER inst_number, BAC_BOOLEAN value)	38

5.11.2.3	ObjBIReset(void)	39
5.12	src/demo/obj_bo.c File Reference	39
5.12.1	Function Documentation	39
5.12.1.1	ObjBOInit(void)	39
5.12.1.2	ObjBOReset(void)	40
5.12.1.3	ObjBOWriteHandling(BACNET_INST_NUMBER inst_number, BACNET_P← ROPERTY_ID property_id, const BACNET_PROPERTY_CONTENTS *p_← property_contents)	40
5.12.2	Variable Documentation	40
5.12.2.1	g_obj_bo1_description	40
5.12.2.2	g_obj_bo1_out_of_service	40
5.12.2.3	g_obj_bo1_present_value	40
5.12.2.4	g_obj_bo1_state_flags	40
5.12.2.5	g_obj_bo2_description	40
5.12.2.6	g_obj_bo2_out_of_service	40
5.12.2.7	g_obj_bo2_present_value	40
5.12.2.8	g_obj_bo2_state_flags	40
5.13	src/demo/obj_bo.h File Reference	40
5.13.1	Macro Definition Documentation	41
5.13.1.1	OBJ_BO1_ACTIVE_INIT_STRING	41
5.13.1.2	OBJ_BO1_DESCRIPTION_INIT_STRING	41
5.13.1.3	OBJ_BO1_INACTIVE_INIT_STRING	41
5.13.1.4	OBJ_BO1_NAME_INIT_STRING	41
5.13.1.5	OBJ_BO2_ACTIVE_INIT_STRING	41
5.13.1.6	OBJ_BO2_DESCRIPTION_INIT_STRING	42
5.13.1.7	OBJ_BO2_INACTIVE_INIT_STRING	42
5.13.1.8	OBJ_BO2_NAME_INIT_STRING	42
5.13.2	Function Documentation	42
5.13.2.1	ObjBOInit(void)	42
5.13.2.2	ObjBOReset(void)	42

5.13.2.3	ObjBOWriteHandling(BACNET_INST_NUMBER inst_number, BACNET_PROPERTY_ID property_id, const BACNET_PROPERTY_CONTENTS *property_contents)	42
5.14	src/demo/obj_ca.c File Reference	43
5.14.1	Function Documentation	43
5.14.1.1	ObjCAInit(void)	43
5.14.2	Variable Documentation	43
5.14.2.1	g_obj_ca1_description	43
5.15	src/demo/obj_ca.h File Reference	43
5.15.1	Macro Definition Documentation	44
5.15.1.1	OBJ_CA1_DESCRIPTION_INIT_STRING	44
5.15.1.2	OBJ_CA1_NAME_INIT_STRING	44
5.15.2	Function Documentation	44
5.15.2.1	ObjCAInit(void)	44
5.16	src/demo/obj_dev.c File Reference	44
5.16.1	Function Documentation	45
5.16.1.1	ObjDevInit(BACNET_TIME_STAMP *p_start_time)	45
5.16.2	Variable Documentation	45
5.16.2.1	g_obj_dev_description	45
5.16.2.2	g_obj_dev_id	45
5.16.2.3	g_obj_dev_location	45
5.16.2.4	g_obj_dev_name	45
5.16.2.5	g_obj_dev_time_of_restart	45
5.17	src/demo/obj_dev.h File Reference	45
5.17.1	Macro Definition Documentation	46
5.17.1.1	OBJ_DEV_DESCRIPTION_INIT_STRING	46
5.17.1.2	OBJ_DEV_FIRMWARE_REV_STRING	46
5.17.1.3	OBJ_DEV_INST_NUMBER	46
5.17.1.4	OBJ_DEV_LOCATION_INIT_STRING	46
5.17.1.5	OBJ_DEV_MAX_APDU_SIZE	46
5.17.1.6	OBJ_DEV_MODEL_NAME_STRING	46

5.17.1.7	OBJ_DEV_NAME_INIT_STRING	46
5.17.1.8	OBJ_DEV_SOFTWARE_VER_STRING	46
5.17.1.9	OBJ_DEV_VENDOR_ID_NUMBER	46
5.17.1.10	OBJ_DEV_VENDOR_NAME_STRING	46
5.17.2	Function Documentation	47
5.17.2.1	ObjDevInit(BACNET_TIME_STAMP *p_start_time)	47
5.18	src/demo/obj_nc.c File Reference	47
5.18.1	Function Documentation	47
5.18.1.1	ObjNCInit(void)	47
5.18.2	Variable Documentation	47
5.18.2.1	g_obj_nc1_description	47
5.19	src/demo/obj_nc.h File Reference	47
5.19.1	Macro Definition Documentation	48
5.19.1.1	OBJ_NC1_DESCRIPTION_INIT_STRING	48
5.19.1.2	OBJ_NC1_NAME_INIT_STRING	48
5.19.2	Function Documentation	48
5.19.2.1	ObjNCInit(void)	48
5.20	src/demo/obj_sc.c File Reference	48
5.20.1	Function Documentation	49
5.20.1.1	ObjSCInit(void)	49
5.20.1.2	ObjSCReset(void)	49
5.20.2	Variable Documentation	49
5.20.2.1	g_obj_sc1_description	49
5.20.2.2	g_obj_sc1_out_of_service	49
5.20.2.3	g_obj_sc1_state_flags	49
5.21	src/demo/obj_sc.h File Reference	49
5.21.1	Macro Definition Documentation	50
5.21.1.1	OBJ_SC1_DAILY_SCHEDULE_SIZE	50
5.21.1.2	OBJ_SC1_DESCRIPTION_INIT_STRING	50
5.21.1.3	OBJ_SC1_NAME_INIT_STRING	50

5.21.2	Function Documentation	50
5.21.2.1	ObjSCInit(void)	50
5.21.2.2	ObjSCReset(void)	50
5.22	src/demo/obj_tr.c File Reference	50
5.22.1	Function Documentation	51
5.22.1.1	ObjTRInit(void)	51
5.22.1.2	ObjTRWriteHandling(BACNET_INST_NUMBER inst_number, BACNET_P↔ ROPERTY_ID property_id, const BACNET_PROPERTY_CONTENTS *p↔ property_contents)	51
5.22.2	Variable Documentation	51
5.22.2.1	g_obj_tr1_description	51
5.22.2.2	g_obj_tr1_evt_msg_txt	51
5.23	src/demo/obj_tr.h File Reference	51
5.23.1	Macro Definition Documentation	52
5.23.1.1	OBJ_TR1_DESCRIPTION_INIT_STRING	52
5.23.1.2	OBJ_TR1_LOG_BUFFER_SIZE	52
5.23.1.3	OBJ_TR1_NAME_INIT_STRING	52
5.23.2	Function Documentation	52
5.23.2.1	ObjTRInit(void)	52
5.23.2.2	ObjTRWriteHandling(BACNET_INST_NUMBER inst_number, BACNET_P↔ ROPERTY_ID property_id, const BACNET_PROPERTY_CONTENTS *p↔ property_contents)	52
5.24	src/demo/user.c File Reference	53
5.24.1	Macro Definition Documentation	56
5.24.1.1	FLASH_DEVICE_FLASH_SIZE	56
5.24.1.2	FLASH_DEVICE_PAGE_SIZE	56
5.24.1.3	FLASH_DEVICE_START_ADDRESS	56
5.24.1.4	FLASH_MEM_MAGIC_SIZE	56
5.24.1.5	USER_ANALOG_AVERAGE_COUNT	56
5.24.1.6	USER_BINARY_VALUE_IS_BUTTON	56
5.24.1.7	USER_MAX_ANALOG	56
5.24.1.8	USER_MAX_BINARY	57

5.24.1.9	USER_MAX_RECORDS	57
5.24.1.10	USER_MIN_VALUE_DIFF	57
5.24.2	Function Documentation	57
5.24.2.1	analog_value_changed(uint8_t index, uint16_t adc_value)	57
5.24.2.2	button_pressed(uint8_t index)	57
5.24.2.3	calc_analog_value(uint8_t index, uint16_t adc_value, float resolution)	58
5.24.2.4	copy_trendlog_data(uint32_t index, uint32_t count, BACNET_LOG_RECORD p_records[], uint32_t *p_count)	58
5.24.2.5	flash_wait(void)	58
5.24.2.6	flash_write(uint32_t flash_address, uint32_t source_address)	58
5.24.2.7	flash_write_page(BAC_BYTE *p_buffer, BAC_WORD buffer_size)	59
5.24.2.8	get_crc_checksum(BAC_BYTE check, BAC_BYTE memory)	59
5.24.2.9	get_switch_state(uint8_t index, bool *p_switch_state)	59
5.24.2.10	get_trendlog_first_last(BACNET_LOG_RECORD p_records[], uint32_t count, bool *p_first, bool *p_last)	60
5.24.2.11	get_trendlog_index_next(uint32_t index)	60
5.24.2.12	get_trendlog_index_prev(uint32_t index)	60
5.24.2.13	get_trendlog_index_start(void)	61
5.24.2.14	init_scheduler(void)	61
5.24.2.15	init_trendlog(void)	61
5.24.2.16	is_flash_magic_activated(void)	61
5.24.2.17	is_property_persistent(const BACNET_SERVER_PROPERTY_INSTANCE *p_prop)	61
5.24.2.18	is_trendlog_index_valid(uint32_t index)	62
5.24.2.19	is_trendlog_time_smaller(BACNET_DATE_TIME record_time, BACNET_DATE_TIME end_time, bool or_equal)	62
5.24.2.20	set_flash_magic(void)	62
5.24.2.21	show_led(uint32_t index, bool on_off)	62
5.24.2.22	user_loop(ULONG entry_input)	63
5.24.2.23	user_reset_values(VOID)	63
5.24.2.24	UserInit(void)	63
5.24.2.25	UserReset(void)	63

5.24.2.26	UserSetLed(uint32_t number, bool on_off)	64
5.24.2.27	UserStorageDelete(const BACNET_SRVR_INIT *p_server)	64
5.24.2.28	UserStorageGetPropertyPersistent(BACNET_INST_NUMBER device_id, BACNET_OBJECT_TYPE obj_type, BACNET_INST_NUMBER inst_number, BACNET_PROPERTY_ID property_id)	64
5.24.2.29	UserStorageRead(const BACNET_SRVR_INIT *p_server)	65
5.24.2.30	UserStorageWrite(const BACNET_SRVR_INIT *p_server)	65
5.24.2.31	UserTrendlogClearBuffer(uint32_t number)	65
5.24.2.32	UserTrendlogGetRecords(uint32_t number, BACNET_LOG_RECORD p_records[], uint32_t *p_count, bool *p_first, bool *p_last, bool *p_more)	66
5.24.2.33	UserTrendlogGetRecordsByPosition(uint32_t number, uint32_t position, int32_t count, BACNET_LOG_RECORD p_records[], uint32_t *p_count, bool *p_first, bool *p_last, bool *p_more)	66
5.24.2.34	UserTrendlogGetRecordsByTime(uint32_t number, BACNET_DATE_TIME start_time, int32_t count, BACNET_LOG_RECORD p_records[], uint32_t *p_count, bool *p_first, bool *p_last, bool *p_more)	66
5.24.2.35	UserTrendlogGetRecordsByTimeRange(uint32_t number, BACNET_DATE_TIME start_time, BACNET_DATE_TIME end_time, BACNET_LOG_RECORD p_records[], uint32_t *p_count, bool *p_first, bool *p_last, bool *p_more)	67
5.24.2.36	UserTrendlogPutRecord(uint32_t number, BACNET_LOG_RECORD *p_record)	67
5.24.2.37	UserTrendlogGetRecordsBySequence(uint32_t number, uint32_t sequence, int32_t count, BACNET_LOG_RECORD p_records[], uint32_t *p_count, bool *p_first, bool *p_last, bool *p_more)	67
5.24.3	Variable Documentation	68
5.24.3.1	g_flash_checksum	68
5.24.3.2	g_flash_magic	68
5.24.3.3	g_flash_mem_page	68
5.24.3.4	g_flash_offset	68
5.24.3.5	g_obj_sc1_daily_schedule	68
5.24.3.6	g_obj_sc1_obj_prop_ref_list	68
5.24.3.7	g_obj_sc1_obj_prop_ref_listCount	68
5.24.3.8	g_page_any_page_changed	68
5.24.3.9	g_page_data_changed	69
5.24.3.10	g_page_is_new	69
5.24.3.11	g_page_offset	69

5.24.3.12	g_sequence	69
5.24.3.13	g_user_analog_values	69
5.24.3.14	g_user_binary_values	69
5.24.3.15	g_user_button	69
5.24.3.16	g_user_led_pins	69
5.24.3.17	g_user_leds	70
5.24.3.18	g_user_mutex	70
5.24.3.19	g_user_thread	70
5.24.3.20	g_user_thread_stack	70
5.24.3.21	g_user_trendlog	70
5.24.3.22	gp_flash_mem	70
5.25	src/demo/user.h File Reference	70
5.25.1	Function Documentation	71
5.25.1.1	UserInit(void)	71
5.25.1.2	UserReset(void)	72
5.25.1.3	UserSetLed(uint32_t number, bool on_off)	72
5.25.1.4	UserStorageDelete(const BACNET_SRVR_INIT *p_server)	72
5.25.1.5	UserStorageIsPropertyPersistent(BACNET_INST_NUMBER device_id, BACNET_OBJECT_TYPE obj_type, BACNET_INST_NUMBER inst_number, BACNET_PROPERTY_ID property_id)	72
5.25.1.6	UserStorageRead(const BACNET_SRVR_INIT *p_server)	73
5.25.1.7	UserStorageWrite(const BACNET_SRVR_INIT *p_server)	73
5.25.1.8	UserTrendlogClearBuffer(uint32_t number)	73
5.25.1.9	UserTrendlogGetRecords(uint32_t number, BACNET_LOG_RECORD p_records[], uint32_t *p_count, bool *p_first, bool *p_last, bool *p_more)	74
5.25.1.10	UserTrendlogGetRecordsByPosition(uint32_t number, uint32_t position, int32_t count, BACNET_LOG_RECORD p_records[], uint32_t *p_count, bool *p_first, bool *p_last, bool *p_more)	74
5.25.1.11	UserTrendlogGetRecordsByTime(uint32_t number, BACNET_DATE_TIME start_time, int32_t count, BACNET_LOG_RECORD p_records[], uint32_t *p_count, bool *p_first, bool *p_last, bool *p_more)	75
5.25.1.12	UserTrendlogGetRecordsByTimeRange(uint32_t number, BACNET_DATE_TIME start_time, BACNET_DATE_TIME end_time, BACNET_LOG_RECORD p_records[], uint32_t *p_count, bool *p_first, bool *p_last, bool *p_more)	75
5.25.1.13	UserTrendlogPutRecord(uint32_t number, BACNET_LOG_RECORD *p_record)	75
5.25.1.14	UserTrendlogGetRecordsBySequence(uint32_t number, uint32_t sequence, int32_t count, BACNET_LOG_RECORD p_records[], uint32_t *p_count, bool *p_first, bool *p_last, bool *p_more)	76

Chapter 1

BACnet API demo application

This demo application creates a BACnet device with following objects: device object binary input objects (2 switches) binary output objects (2 LED's) analog input objects (2 potentiometers and 2 temperature sensors) analog output object (1 object used by the scheduler object) scheduler object (with entries for every full hour from Monday to Sunday) trend log object (connected to potentiometer 1) calendar object (for the scheduler) notification class object (for alarming)

Only the UDP/IP data link is used by this demo application. You can change to MSTP with enabling the macro 'BACNET_DATA_LINK_MSTP'.

The foreign device service is disabled by default for this demo application. You can enable the service with enabling the macro 'BACNET_DEMO_WITH_FOREIGN_DEVICE'.

The entry point for demo application is the function 'BacnetDemoMain'.

Chapter 2

Data Structure Index

2.1 Data Structures

Here are the data structures with brief descriptions:

user_analog_input_t	7
user_binary_input_t	8
user_button_t	8
user_record_t	9
user_trendlog_t	10

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

src/demo/adc.c	11
src/demo/adc.h	13
src/demo/bacnetdemo.c	14
src/demo/bacnetdemo.h	19
src/demo/definitions.h	21
src/demo/obj_ai.c	25
src/demo/obj_ai.h	27
src/demo/obj_ao.c	32
src/demo/obj_ao.h	34
src/demo/obj_bi.c	35
src/demo/obj_bi.h	37
src/demo/obj_bo.c	39
src/demo/obj_bo.h	40
src/demo/obj_ca.c	43
src/demo/obj_ca.h	43
src/demo/obj_dev.c	44
src/demo/obj_dev.h	45
src/demo/obj_nc.c	47
src/demo/obj_nc.h	47
src/demo/obj_sc.c	48
src/demo/obj_sc.h	49
src/demo/obj_tr.c	50
src/demo/obj_tr.h	51
src/demo/user.c	53
src/demo/user.h	70

Chapter 4

Data Structure Documentation

4.1 user_analog_input_t Struct Reference

Data Fields

- uint16_t [adc_sum_count](#)
- uint32_t [adc_sum](#)
- uint16_t [adc_value](#)
- float [bacnet_value](#)
- float [resolution](#)

4.1.1 Detailed Description

Definition at line 83 of file user.c.

4.1.2 Field Documentation

4.1.2.1 uint32_t adc_sum

Definition at line 86 of file user.c.

4.1.2.2 uint16_t adc_sum_count

Definition at line 85 of file user.c.

4.1.2.3 uint16_t adc_value

Definition at line 87 of file user.c.

4.1.2.4 float bacnet_value

Definition at line 88 of file user.c.

4.1.2.5 float resolution

Definition at line 89 of file user.c.

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

- [src/demo/user.c](#)

4.2 user_binary_input_t Struct Reference

Data Fields

- bool [button](#)
- bool [state](#)

4.2.1 Detailed Description

Definition at line 77 of file user.c.

4.2.2 Field Documentation

4.2.2.1 bool button

Definition at line 79 of file user.c.

4.2.2.2 bool state

Definition at line 80 of file user.c.

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

- [src/demo/user.c](#)

4.3 user_button_t Struct Reference

Data Fields

- ioport_port_pin_t [pin](#)
- ioport_level_t [level](#)

4.3.1 Detailed Description

Definition at line 71 of file user.c.

4.3.2 Field Documentation

4.3.2.1 ioport_level_t level

Definition at line 74 of file user.c.

4.3.2.2 ioport_port_pin_t pin

Definition at line 73 of file user.c.

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

- [src/demo/user.c](#)

4.4 user_record_t Struct Reference

```
#include <user.h>
```

Data Fields

- BACNET_LOG_RECORD [data](#)

4.4.1 Detailed Description

Definition at line 28 of file user.h.

4.4.2 Field Documentation

4.4.2.1 BACNET_LOG_RECORD data

Definition at line 30 of file user.h.

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

- [src/demo/user.h](#)

4.5 user_trendlog_t Struct Reference

Data Fields

- [uint8_t pos](#)
- [uint32_t count](#)
- [user_record_t records \[USER_MAX_RECORDS\]](#)

4.5.1 Detailed Description

Definition at line 92 of file user.c.

4.5.2 Field Documentation

4.5.2.1 uint32_t count

Definition at line 95 of file user.c.

4.5.2.2 uint8_t pos

Definition at line 94 of file user.c.

4.5.2.3 user_record_t records[USER_MAX_RECORDS]

Definition at line 96 of file user.c.

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

- [src/demo/user.c](#)

Chapter 5

File Documentation

5.1 src/demo/adc.c File Reference

```
#include "hal_data.h"  
#include "adc.h"  
#include "sf_cslab_bacnet_common_api.h"
```

Macros

- #define [ADC_REGISTER_COUNT](#) (4)

Functions

- bool [AdcInit](#) (void)
Initialize ADC Unit.
- void [AdcCompleteCallback](#) (adc_callback_args_t *p_args)
This Callback is used to store the current values read from used registers.
- bool [AdcGetValue](#) (uint8_t index, uint16_t *p_value)
Returns the current value of a register.

Variables

- const adc_instance_t [g_adc](#)
- const adc_register_t [g_adc_register](#) [[ADC_REGISTER_COUNT](#)]
- static uint16_t [g_adc_data](#) [[ADC_REGISTER_COUNT](#)]
- static bool [g_adc_started](#) = false

5.1.1 Macro Definition Documentation

5.1.1.1 #define [ADC_REGISTER_COUNT](#) (4)

Definition at line 22 of file adc.c.

5.1.2 Function Documentation

5.1.2.1 void AdcCompleteCallback (adc_callback_args_t * p_args)

This Callback is used to store the current values read from used registers.

Definition at line 106 of file adc.c.

5.1.2.2 bool AdcGetValue (uint8_t index, uint16_t * p_value)

Returns the current value of a register.

Parameters

in	<i>index</i>	Index for the register value array.
out	<i>p_value</i>	Pointer for returning the register value.

Return values

<i>true</i>	Returned register value is valid.
<i>false</i>	Demo application is not completely initialized or wrong index for the array.

Definition at line 140 of file adc.c.

5.1.2.3 bool AdcInIt (void)

Initialize ADC Unit.

The Open function applies power to the A/D peripheral, sets the operational mode, trigger sources, interrupt priority, and configurations for the peripheral as a whole. Configure the ADC scan parameters and starts a software scan or enables the hardware trigger for a scan depending on how the triggers were configured in the Open() call.

The global register values were initialized with 0.

Return values

<i>true</i>	Initialization was successful and ADC has started.
<i>false</i>	Invalid p_ctrl or p_cfg pointer.

Definition at line 69 of file adc.c.

5.1.3 Variable Documentation

5.1.3.1 const adc_instance_t g_adc

5.1.3.2 uint16_t g_adc_data[ADC_REGISTER_COUNT] [static]

Definition at line 48 of file adc.c.

5.1.3.3 const adc_register_t g_adc_register[ADC_REGISTER_COUNT]

Initial value:

```
=  
{  
    ADC_SAMPLE_STATE_CHANNEL_0,  
    ADC_SAMPLE_STATE_CHANNEL_1,  
    ADC_SAMPLE_STATE_CHANNEL_3,  
    ADC_SAMPLE_STATE_CHANNEL_4  
}
```

Definition at line 40 of file adc.c.

5.1.3.4 bool g_adc_started = false [static]

Definition at line 50 of file adc.c.

5.2 src/demo/adc.h File Reference

```
#include <stdint.h>  
#include <stdbool.h>
```

Functions

- bool [AdcInit](#) (void)
Initialize ADC Unit.
- void [AdcCompleteCallback](#) (adc_callback_args_t *p_args)
This Callback is used to store the current values read from used registers.
- bool [AdcGetValue](#) (uint8_t index, uint16_t *p_value)
Returns the current value of a register.

5.2.1 Function Documentation

5.2.1.1 void AdcCompleteCallback (adc_callback_args_t * p_args)

This Callback is used to store the current values read from used registers.

Definition at line 106 of file adc.c.

5.2.1.2 bool AdcGetValue (uint8_t index, uint16_t * p_value)

Returns the current value of a register.

Parameters

in	<i>index</i>	Index for the register value array.
out	<i>p_value</i>	Pointer for returning the register value.

Return values

<i>true</i>	Returned register value is valid.
<i>false</i>	Demo application is not completely initialized or wrong index for the array.

Definition at line 140 of file adc.c.

5.2.1.3 bool AdcInit (void)

Initialize ADC Unit.

The Open function applies power to the A/D peripheral, sets the operational mode, trigger sources, interrupt priority, and configurations for the peripheral as a whole. Configure the ADC scan parameters and starts a software scan or enables the hardware trigger for a scan depending on how the triggers were configured in the Open() call.

The global register values were initialized with 0.

Return values

<i>true</i>	Initialization was successful and ADC has started.
<i>false</i>	Invalid p_ctrl or p_cfg pointer.

Definition at line 69 of file adc.c.

5.3 src/demo/bacnetdemo.c File Reference

```
#include <demo/bacnetdemo.h>
#include "nx_api.h"
#include "hal_data.h"
#include "bacnet.h"
#include "definitions.h"
#include "adc.h"
#include "user.h"
#include "sf_cslab_bacnet_common.h"
#include "sf_cslab_bacnet_ip.h"
```

Macros

- #define [BACNET_DEMO_WITH_FOREIGN_DEVICE](#)

Functions

- static bool [bacnet_init](#) (BACNET_TIME_STAMP *p_start_time)
Initialize the BACnet server.
- static void [bacnet_reset](#) (void)
Called in case of a reset request.
- static BACNET_CALLBACK_STATUS [bacnet_write](#) (BACNET_INST_NUMBER device_id, BACNET_OBJECT_TYPE obj_type, BACNET_INST_NUMBER inst_number, BACNET_PROPERTY_ID property_id, BACNET_ARRAY_INDEX index, const BACNET_PROPERTY_CONTENTS *p_property_contents, BACNET_BOOLEAN *p_property_persistent)
This function is called by the BACnet API for every changed property of an object of a device.
- static BACNET_LOG_RECORD * [bacnet_trend](#) (BACNET_READ_RANGE_INFO *p_service_info, BACNET_UINT *p_entries, BAC_BOOLEAN *p_first, BAC_BOOLEAN *p_last, BAC_BOOLEAN *p_more)
Request for trend log data.
- static BAC_BOOLEAN [bacnet_storage_read](#) (void)
Notification that the user should read persistent stored data.
- static BAC_BOOLEAN [bacnet_storage_write](#) (void)
Notification that the user should write persistent stored data.
- static BAC_BOOLEAN [bacnet_storage_delete](#) (void)
Notification that the user should delete persistent stored data.
- void [BacnetDemoMain](#) (void)
This is the main function of the demo application and is called by the thread-x framework.

Variables

- static sf_cslab_bacnet_ctrl_t [g_bacnet_data_link_ctrl](#)
- static const sf_cslab_bacnet_on_sf_cslab_bacnet_ip_cfg [g_bacnet_data_link_ip_ext_cfg](#)
- static const sf_cslab_bacnet_cfg_t [g_bacnet_data_link_cfg](#)
- const sf_cslab_bacnet_instance_t [g_bacnet_instance](#)
- static sf_cslab_bacnet_common_ctrl_t [g_bacnet_common_ctrl](#)
- static sf_cslab_bacnet_common_cfg_t [g_bacnet_common_cfg](#)
- const sf_cslab_bacnet_common_instance_t [g_bacnet_common_instance](#)
- static BACNET_LOG_RECORD [g_log_record](#) [OBJ_TR1_LOG_BUFFER_SIZE]

5.3.1 Macro Definition Documentation

5.3.1.1 #define BACNET_DEMO_WITH_FOREIGN_DEVICE

Enable this macro if the data link is MSTP instead of IP Enable this macro if the demo application use a foreign device

Definition at line 42 of file bacnetdemo.c.

5.3.2 Function Documentation

5.3.2.1 static bool bacnet_init (BACNET_TIME_STAMP * p_start_time) [static]

Initialize the BACnet server.

Returns

Initialization was successful or not.

Definition at line 169 of file bacnetdemo.c.

5.3.2.2 static void bacnet_reset (void) [static]

Called in case of a reset request.

Definition at line 246 of file bacnetdemo.c.

5.3.2.3 static BAC_BOOLEAN bacnet_storage_delete (void) [static]

Notification that the user should delete persistent stored data.

Returns

Deleting was successful or not.

Definition at line 386 of file bacnetdemo.c.

5.3.2.4 static BAC_BOOLEAN bacnet_storage_read (void) [static]

Notification that the user should read persistent stored data.

Returns

Reading was successful or not.

Definition at line 366 of file bacnetdemo.c.

5.3.2.5 static BAC_BOOLEAN bacnet_storage_write (void) [static]

Notification that the user should write persistent stored data.

Returns

Writing was successful or not.

Definition at line 376 of file bacnetdemo.c.

5.3.2.6 static BACNET_LOG_RECORD * bacnet_trend (BACNET_READ_RANGE_INFO * p_service_info, BAC_UINT * p_entries, BAC_BOOLEAN * p_first, BAC_BOOLEAN * p_last, BAC_BOOLEAN * p_more) [static]

Request for trend log data.

Parameters

in	<i>p_service_info</i>	A pointer to a data structure which contains all service parameters this service provides and the application needs to process.
out	<i>p_entries</i>	Number of records in returned trend log array.
out	<i>p_first</i>	First record is in returned trend log array.
out	<i>p_last</i>	Last record is in returned trend log array.
out	<i>p_more</i>	More records available but not in returned trend log array.

Returns

On success a pointer to an array with trend log records will be returned otherwise a NULL pointer.

Definition at line 263 of file bacnetdemo.c.

```
5.3.2.7 static BACNET_CALLBACK_STATUS bacnet_write ( BACNET_INST_NUMBER device_id, BACNET_OBJECT_TYPE
obj_type, BACNET_INST_NUMBER inst_number, BACNET_PROPERTY_ID property_id, BACNET_ARRAY_INDEX
index, const BACNET_PROPERTY_CONTENTS * p_property_contents, BACNET_BOOLEAN * p_property_persistent )
[static]
```

This function is called by the BACnet API for every changed property of an object of a device.

For this demo application only binary output, analog output and trend log objects were processed.

Parameters

in	<i>device_id</i>	The device of the changed property.
in	<i>obj_type</i>	The object type of the changed property.
in	<i>inst_number</i>	The instance number of the changed property.
in	<i>property_id</i>	This property that has been changed.
in	<i>index</i>	Index for array properties.
in	<i>p_property_contents</i>	The new content of the property (property type depending).
out	<i>p_property_persistent</i>	The property should be stored by the user or not.

Return values

<i>status</i>	Informs the BACnet API about success or errors while processing. Is always CALLBACK_STATUS_DEFAULT for this demo application.
---------------	---

Definition at line 407 of file bacnetdemo.c.

```
5.3.2.8 void BacnetDemoMain ( void )
```

This is the main function of the demo application and is called by the thread-x framework.

This function open the defined common bacnet instance and the connected data link. On success the common bacnet instance will be initialized and started. These two calls to the open() below will be called by the ISDE when there are Synergy Configurator XMLs available.

Open BACnet common code.

Open up data link layer. This could be IP or MS/TP depending on the pointers in g_bacnet_data_link_instance.

At this point this is basically the user's application. This is code they will be responsible for writing. All of the BACnet objects are taken care of in a different file.

Definition at line 454 of file bacnetdemo.c.

5.3.3 Variable Documentation

5.3.3.1 `sf_cslab_bacnet_common_cfg_t g_bacnet_common_cfg` [static]

Initial value:

```
=
{
    .period_ticks = 1
}
```

Definition at line 140 of file bacnetdemo.c.

5.3.3.2 `sf_cslab_bacnet_common_ctrl_t g_bacnet_common_ctrl` [static]

Initial value:

```
=
{
    .p_instance = &g_bacnet_instance
}
```

Definition at line 135 of file bacnetdemo.c.

5.3.3.3 `const sf_cslab_bacnet_common_instance_t g_bacnet_common_instance`

Initial value:

```
=
{
    .p_ctrl = &g_bacnet_common_ctrl,
    .p_cfg = &g_bacnet_common_cfg,
    .p_api = &g_sf_cslab_bacnet_common_on_sf_cslab_bacnet_common,
}
```

Definition at line 145 of file bacnetdemo.c.

5.3.3.4 `const sf_cslab_bacnet_cfg_t g_bacnet_data_link_cfg` [static]

Initial value:

```
=
{
    .p_server = &g_server,
    .priority = 2,

    .p_extend = &g_bacnet_data_link_ip_ext_cfg
}
```

Definition at line 113 of file bacnetdemo.c.

5.3.3.5 sf_cslab_bacnet_ctrl_t g_bacnet_data_link_ctrl [static]

Initial value:

```
=
{
    .p_instance_ctrl = NULL,
    .init            = bacnet_init,
    .reset          = bacnet_reset,
    .write          = bacnet_write,
    .trend         = bacnet_trend,
    .storage_read   = bacnet_storage_read,
    .storage_write  = bacnet_storage_write,
    .storage_delete = bacnet_storage_delete
}
```

Definition at line 88 of file bacnetdemo.c.

5.3.3.6 const sf_cslab_bacnet_on_sf_cslab_bacnet_ip_cfg g_bacnet_data_link_ip_ext_cfg [static]

Initial value:

```
=
{
    .ip    = {DEV_DATALINK_ETH_IP1, DEV_DATALINK_ETH_IP2,
              DEV_DATALINK_ETH_IP3, DEV_DATALINK_ETH_IP4},
    .port  = DEV_DATALINK_ETH_PORT
}
```

Definition at line 106 of file bacnetdemo.c.

5.3.3.7 const sf_cslab_bacnet_instance_t g_bacnet_instance

Initial value:

```
=
{
    .p_ctrl = &g_bacnet_data_link_ctrl,
    .p_cfg  = &g_bacnet_data_link_cfg,

    .p_api = &g_sf_cslab_bacnet_on_sf_cslab_bacnet_ip
}
```

Definition at line 124 of file bacnetdemo.c.

5.3.3.8 BACNET_LOG_RECORD g_log_record[OBJ_TR1_LOG_BUFFER_SIZE] [static]

Definition at line 158 of file bacnetdemo.c.

5.4 src/demo/bacnetdemo.h File Reference

```
#include "bacnet.h"
```

Enumerations

- enum `demo_obj_ids_t` {
 `DEMO_OBJ_ID_DEV` = 0, `DEMO_OBJ_ID_BI1`, `DEMO_OBJ_ID_BI2`, `DEMO_OBJ_ID_BO1`,
 `DEMO_OBJ_ID_BO2`, `DEMO_OBJ_ID_AI1`, `DEMO_OBJ_ID_AI2`, `DEMO_OBJ_ID_AI3`,
 `DEMO_OBJ_ID_AI4`, `DEMO_OBJ_ID_AO1`, `DEMO_OBJ_ID_SC1`, `DEMO_OBJ_ID_CA1`,
 `DEMO_OBJ_ID_TR1`, `DEMO_OBJ_ID_NC1`, `DEMO_OBJ_ID_MAX` }

Functions

- void `BacnetDemoMain` (void)

This is the main function of the demo application and is called by the thread-x framework.

5.4.1 Enumeration Type Documentation

5.4.1.1 enum `demo_obj_ids_t`

Enumerator

`DEMO_OBJ_ID_DEV`
`DEMO_OBJ_ID_BI1`
`DEMO_OBJ_ID_BI2`
`DEMO_OBJ_ID_BO1`
`DEMO_OBJ_ID_BO2`
`DEMO_OBJ_ID_AI1`
`DEMO_OBJ_ID_AI2`
`DEMO_OBJ_ID_AI3`
`DEMO_OBJ_ID_AI4`
`DEMO_OBJ_ID_AO1`
`DEMO_OBJ_ID_SC1`
`DEMO_OBJ_ID_CA1`
`DEMO_OBJ_ID_TR1`
`DEMO_OBJ_ID_NC1`
`DEMO_OBJ_ID_MAX`

Definition at line 25 of file `bacnetdemo.h`.

5.4.2 Function Documentation

5.4.2.1 void `BacnetDemoMain` (void)

This is the main function of the demo application and is called by the thread-x framework.

This function open the defined common bacnet instance and the connected data link. On success the common bacnet instance will be initialized and started. These two calls to the `open()` below will be called by the ISDE when there are Synergy Configurator XMLs available.

Open BACnet common code.

Open up data link layer. This could be IP or MS/TP depending on the pointers in `g_bacnet_data_link_instance`.

At this point this is basically the user's application. This is code they will be responsible for writing. All of the BACnet objects are taken care of in a different file.

Definition at line 454 of file `bacnetdemo.c`.

5.5 src/demo/definitions.h File Reference

```
#include "obj_dev.h"
#include "obj_bi.h"
#include "obj_bo.h"
#include "obj_ai.h"
#include "obj_ao.h"
#include "obj_sc.h"
#include "obj_ca.h"
#include "obj_tr.h"
#include "obj_nc.h"
```

Macros

- `#define DEV_DATALINK_COUNT` (1)
- `#define DEV_DATALINK_ETH_IP1` (192)
- `#define DEV_DATALINK_ETH_IP2` (168)
- `#define DEV_DATALINK_ETH_IP3` (0)
- `#define DEV_DATALINK_ETH_IP4` (111)
- `#define DEV_DATALINK_ETH_PORT` (47808)
- `#define DEV_DATALINK_ETH_PORT_ID` (1)
- `#define DEV_DATALINK_ETH_NET_ID` (1)
- `#define DEV_DATALINK_FOREIGN_IP1` (192)
- `#define DEV_DATALINK_FOREIGN_IP2` (168)
- `#define DEV_DATALINK_FOREIGN_IP3` (0)
- `#define DEV_DATALINK_FOREIGN_IP4` (100)
- `#define DEV_DATALINK_FOREIGN_PORT` (47808)
- `#define DEV_DATALINK_FOREIGN_INTERVAL` (150)
- `#define DEV_DATALINK_MSTP_PORT` (3)
- `#define DEV_DATALINK_MSTP_NET` (33)
- `#define DEV_DATALINK_MSTP_ADDRESS` (12)
- `#define DEVICE_DDC_PASSWORD_INIT_STRING` ("s7g2-DCC")
- `#define DEVICE_REINIT_PASSWORD_INIT_STRING` ("s7g2-Reinit")

Functions

- `CFG_WRITABLE_ADDRESS` (`g_server_device_address`)
- `CFG_WRITABLE_APDU_PROPERTIES` (`g_server_device_apdu_properties`, `OBJ_DEV_MAX_APDU_SIZE`, `TARGET_SUPPORTED_SEGMENTATION_TYPE`, `TARGET_MAX_SUPPORTED_NPDU_SEGMENTATION_SIZE`, 1, 2000, 3000, 5, 60, 5)
- `CFG_READONLY_STRING` (`g_server_device_dcc_password`, `DEVICE_DDC_PASSWORD_INIT_STRING`)
- `CFG_READONLY_STRING` (`g_server_device_reinit_password`, `DEVICE_REINIT_PASSWORD_INIT_STRING`)
- `CFG_SERVER_INIT` (`g_server`, `g_server_device`, `DEV_DATALINK_COUNT`, `g_device_data_links`[0], 1)

Variables

- `const BACNET_SERVER_TEMPLATE_DEVICE` `g_server_device`
- `const BACNET_SERVER_TEMPLATE_OBJECT` `g_obj_array` [`DEMO_OBJ_ID_MAX`]
- `BACNET_DCC_VALUE` `g_server_device_dcc_value` = `DCC_ENABLE`
- `BACNET_SERVER_DATALINK` `g_device_data_links` [`DEV_DATALINK_COUNT`]

5.5.1 Macro Definition Documentation

5.5.1.1 #define DEV_DATALINK_COUNT (1)

Definition at line 30 of file definitions.h.

5.5.1.2 #define DEV_DATALINK_ETH_IP1 (192)

Definition at line 32 of file definitions.h.

5.5.1.3 #define DEV_DATALINK_ETH_IP2 (168)

Definition at line 33 of file definitions.h.

5.5.1.4 #define DEV_DATALINK_ETH_IP3 (0)

Definition at line 34 of file definitions.h.

5.5.1.5 #define DEV_DATALINK_ETH_IP4 (111)

Definition at line 35 of file definitions.h.

5.5.1.6 #define DEV_DATALINK_ETH_NET_ID (1)

Definition at line 38 of file definitions.h.

5.5.1.7 #define DEV_DATALINK_ETH_PORT (47808)

Definition at line 36 of file definitions.h.

5.5.1.8 #define DEV_DATALINK_ETH_PORT_ID (1)

Definition at line 37 of file definitions.h.

5.5.1.9 #define DEV_DATALINK_FOREIGN_INTERVAL (150)

Definition at line 45 of file definitions.h.

5.5.1.10 #define DEV_DATALINK_FOREIGN_IP1 (192)

Definition at line 40 of file definitions.h.

5.5.1.11 #define DEV_DATALINK_FOREIGN_IP2 (168)

Definition at line 41 of file definitions.h.

5.5.1.12 #define DEV_DATALINK_FOREIGN_IP3 (0)

Definition at line 42 of file definitions.h.

5.5.1.13 #define DEV_DATALINK_FOREIGN_IP4 (100)

Definition at line 43 of file definitions.h.

5.5.1.14 #define DEV_DATALINK_FOREIGN_PORT (47808)

Definition at line 44 of file definitions.h.

5.5.1.15 #define DEV_DATALINK_MSTP_ADDRESS (12)

Definition at line 49 of file definitions.h.

5.5.1.16 #define DEV_DATALINK_MSTP_NET (33)

Definition at line 48 of file definitions.h.

5.5.1.17 #define DEV_DATALINK_MSTP_PORT (3)

Definition at line 47 of file definitions.h.

5.5.1.18 #define DEVICE_DDC_PASSWORD_INIT_STRING ("s7g2-DCC")

Definition at line 51 of file definitions.h.

5.5.1.19 #define DEVICE_REINIT_PASSWORD_INIT_STRING ("s7g2-Reinit")

Definition at line 52 of file definitions.h.

5.5.2 Function Documentation

5.5.2.1 `CFG_READONLY_STRING (g_server_device_dcc_password , DEVICE_DDC_PASSWORD_INIT_STRING)`

5.5.2.2 `CFG_READONLY_STRING (g_server_device_reinit_password , DEVICE_REINIT_PASSWORD_INIT_STRING)`

5.5.2.3 `CFG_SERVER_INIT (g_server , g_server_device , DEV_DATALINK_COUNT , g_device_data_links [0], 1)`

5.5.2.4 `CFG_WRITABLE_ADDRESS (g_server_device_address)`

5.5.2.5 `CFG_WRITABLE_APDU_PROPERTIES (g_server_device_apdu_properties , OBJ_DEV_MAX_APDU_SIZE , TARGET_SUPPORTED_SEGMENTATION_TYPE , TARGET_MAX_SUPPORTED_NPDU_SEGMENTS , 1 , 2000 , 3000 , 5 , 60 , 5)`

5.5.3 Variable Documentation

5.5.3.1 `BACNET_SERVER_DATALINK g_device_data_links[DEV_DATALINK_COUNT]`

Definition at line 108 of file definitions.h.

5.5.3.2 `const BACNET_SERVER_TEMPLATE_OBJECT g_obj_array[DEMO_OBJ_ID_MAX]`

Initial value:

```
=
{
    CFG_OBJECT(g_obj_dev, g_server_device, g_obj_dev_cov_buffer, NULL,
               g_obj_dev_function_buffer),
    CFG_OBJECT(g_obj_bi1, g_server_device, g_obj_bi1_cov_buffer, g_obj_bi1_int_buffer,
               g_obj_bi1_function_buffer),
    CFG_OBJECT(g_obj_bi2, g_server_device, g_obj_bi2_cov_buffer, g_obj_bi2_int_buffer,
               g_obj_bi2_function_buffer),
    CFG_OBJECT(g_obj_bo1, g_server_device, NULL, NULL, NULL),
    CFG_OBJECT(g_obj_bo2, g_server_device, NULL, NULL, NULL),
    CFG_OBJECT(g_obj_ai1, g_server_device, g_obj_ai1_cov_buffer, g_obj_ai1_int_buffer, NULL),
    CFG_OBJECT(g_obj_ai2, g_server_device, g_obj_ai2_cov_buffer, g_obj_ai2_int_buffer, NULL),
    CFG_OBJECT(g_obj_ai3, g_server_device, g_obj_ai3_cov_buffer, g_obj_ai3_int_buffer, NULL),
    CFG_OBJECT(g_obj_ai4, g_server_device, g_obj_ai4_cov_buffer, g_obj_ai4_int_buffer, NULL),
    CFG_OBJECT(g_obj_ao1, g_server_device, g_obj_ao1_cov_buffer, g_obj_ao1_int_buffer, NULL),
    CFG_OBJECT(g_obj_sc1, g_server_device, NULL, g_obj_sc1_int_buffer,
               g_obj_sc1_function_buffer),
    CFG_OBJECT(g_obj_ca1, g_server_device, NULL, NULL, NULL),
    CFG_OBJECT(g_obj_tr1, g_server_device, NULL, g_obj_tr1_int_buffer,
               g_obj_tr1_function_buffer),
    CFG_OBJECT(g_obj_nc1, g_server_device, NULL, NULL, NULL),
}
```

Definition at line 59 of file definitions.h.

5.5.3.3 const BACNET_SERVER_TEMPLATE_DEVICE g_server_device

Initial value:

```
=  
{  
    &g_server_device_address,  
    &g_server_device_dcc_value,  
    &g_server_device_dcc_password,  
    &g_server_device_reinit_password,  
    &g_server_device_apdu_properties,  
    &g_obj_array[0],  
}
```

Definition at line 94 of file definitions.h.

5.5.3.4 BACNET_DCC_VALUE g_server_device_dcc_value = DCC_ENABLE

Definition at line 79 of file definitions.h.

5.6 src/demo/obj_ai.c File Reference

```
#include <string.h>  
#include "obj_ai.h"  
#include "obj_dev.h"
```

Functions

- void [ObjAllInit](#) (void)
Initialize the analog input objects for this demo application.
- void [ObjAIReset](#) (void)
Reset the analog input object for this demo application.
- void [ObjAIPresentValueUpdate](#) (BACNET_INST_NUMBER inst_number, BACNET_REAL value)
Notifies the BACnet API that the property Present_Value has changed.

Variables

- BACNET_OBJECT_ID [g_obj_dev_id](#)
- BAC_UINT [g_obj_ai1_description](#) []
- BAC_UINT [g_obj_ai2_description](#) []
- BAC_UINT [g_obj_ai3_description](#) []
- BAC_UINT [g_obj_ai4_description](#) []
- BAC_UINT [g_obj_ai1_evt_msg_txt](#) []
- BAC_UINT [g_obj_ai2_evt_msg_txt](#) []
- BAC_UINT [g_obj_ai3_evt_msg_txt](#) []
- BAC_UINT [g_obj_ai4_evt_msg_txt](#) []
- BACNET_REAL [g_obj_ai1_present_value](#)
- BACNET_REAL [g_obj_ai2_present_value](#)
- BACNET_REAL [g_obj_ai3_present_value](#)

- BACNET_REAL [g_obj_ai4_present_value](#)
- BACNET_BOOLEAN [g_obj_ai1_out_of_service](#)
- BACNET_BOOLEAN [g_obj_ai2_out_of_service](#)
- BACNET_BOOLEAN [g_obj_ai3_out_of_service](#)
- BACNET_BOOLEAN [g_obj_ai4_out_of_service](#)
- BACNET_SHORT_BIT_STRING [g_obj_ai1_state_flags](#)
- BACNET_SHORT_BIT_STRING [g_obj_ai2_state_flags](#)
- BACNET_SHORT_BIT_STRING [g_obj_ai3_state_flags](#)
- BACNET_SHORT_BIT_STRING [g_obj_ai4_state_flags](#)

5.6.1 Function Documentation

5.6.1.1 void ObjAllnit (void)

Initialize the analog input objects for this demo application.

Used here for setting some writable texts.

Definition at line 71 of file obj_ai.c.

5.6.1.2 void ObjAIPresentValueUpdate (BACNET_INST_NUMBER *inst_number*, BACNET_REAL *value*)

Notifies the BACnet API that the property Present_Value has changed.

This demo application provide 4 analog input objects. The values for these objects where updated by 2 potentiometers and 2 temperature sensors.

Parameters

in	<i>inst_number</i>	The instance number for the analog input object.
in	<i>value</i>	The new value for the analog input object.

Definition at line 109 of file obj_ai.c.

5.6.1.3 void ObjAIReset (void)

Reset the analog input object for this demo application.

Definition at line 87 of file obj_ai.c.

5.6.2 Variable Documentation

5.6.2.1 BAC_UINT [g_obj_ai1_description\[\]](#)

5.6.2.2 BAC_UINT [g_obj_ai1_evt_msg_txt\[\]](#)

- 5.6.2.3 BACNET_BOOLEAN g_obj_ai1_out_of_service
- 5.6.2.4 BACNET_REAL g_obj_ai1_present_value
- 5.6.2.5 BACNET_SHORT_BIT_STRING g_obj_ai1_state_flags
- 5.6.2.6 BAC_UINT g_obj_ai2_description[]
- 5.6.2.7 BAC_UINT g_obj_ai2_evt_msg_txt[]
- 5.6.2.8 BACNET_BOOLEAN g_obj_ai2_out_of_service
- 5.6.2.9 BACNET_REAL g_obj_ai2_present_value
- 5.6.2.10 BACNET_SHORT_BIT_STRING g_obj_ai2_state_flags
- 5.6.2.11 BAC_UINT g_obj_ai3_description[]
- 5.6.2.12 BAC_UINT g_obj_ai3_evt_msg_txt[]
- 5.6.2.13 BACNET_BOOLEAN g_obj_ai3_out_of_service
- 5.6.2.14 BACNET_REAL g_obj_ai3_present_value
- 5.6.2.15 BACNET_SHORT_BIT_STRING g_obj_ai3_state_flags
- 5.6.2.16 BAC_UINT g_obj_ai4_description[]
- 5.6.2.17 BAC_UINT g_obj_ai4_evt_msg_txt[]
- 5.6.2.18 BACNET_BOOLEAN g_obj_ai4_out_of_service
- 5.6.2.19 BACNET_REAL g_obj_ai4_present_value
- 5.6.2.20 BACNET_SHORT_BIT_STRING g_obj_ai4_state_flags
- 5.6.2.21 BACNET_OBJECT_ID g_obj_dev_id

5.7 src/demo/obj_ai.h File Reference

```
#include "bacnetdemo.h"
```

Macros

- #define `OBJ_AI1_NAME_INIT_STRING` ("Demo - AI.1")
- #define `OBJ_AI1_DESCRIPTION_INIT_STRING` ("Potentiometer 1")
- #define `OBJ_AI1_PRESENT_VALUE_INITIAL` (50)
- #define `OBJ_AI1_COV_INCREMENT_INITIAL` (1)
- #define `OBJ_AI1_UNIT_INITIAL` (UNIT_PERCENT)
- #define `OBJ_AI1_LIMIT_HIGH_INITIAL` (90)
- #define `OBJ_AI1_LIMIT_LOW_INITIAL` (10)
- #define `OBJ_AI1_LIMIT_DEADBAND_INITIAL` (5)
- #define `OBJ_AI2_NAME_INIT_STRING` ("Demo - AI.2")
- #define `OBJ_AI2_DESCRIPTION_INIT_STRING` ("Potentiometer 2")
- #define `OBJ_AI2_PRESENT_VALUE_INITIAL` (50)
- #define `OBJ_AI2_COV_INCREMENT_INITIAL` (1)
- #define `OBJ_AI2_UNIT_INITIAL` (UNIT_PERCENT)
- #define `OBJ_AI2_LIMIT_HIGH_INITIAL` (90)
- #define `OBJ_AI2_LIMIT_LOW_INITIAL` (10)
- #define `OBJ_AI2_LIMIT_DEADBAND_INITIAL` (5)
- #define `OBJ_AI3_NAME_INIT_STRING` ("Demo - AI.3")
- #define `OBJ_AI3_DESCRIPTION_INIT_STRING` ("Temperature 1")
- #define `OBJ_AI3_PRESENT_VALUE_INITIAL` (20)
- #define `OBJ_AI3_COV_INCREMENT_INITIAL` (0.5)
- #define `OBJ_AI3_UNIT_INITIAL` (UNIT_DEGREES_C)
- #define `OBJ_AI3_LIMIT_HIGH_INITIAL` (50)
- #define `OBJ_AI3_LIMIT_LOW_INITIAL` (0)
- #define `OBJ_AI3_LIMIT_DEADBAND_INITIAL` (2)
- #define `OBJ_AI4_NAME_INIT_STRING` ("Demo - AI.4")
- #define `OBJ_AI4_DESCRIPTION_INIT_STRING` ("Temperature 2")
- #define `OBJ_AI4_PRESENT_VALUE_INITIAL` (20)
- #define `OBJ_AI4_COV_INCREMENT_INITIAL` (0.5)
- #define `OBJ_AI4_UNIT_INITIAL` (UNIT_DEGREES_C)
- #define `OBJ_AI4_LIMIT_HIGH_INITIAL` (50)
- #define `OBJ_AI4_LIMIT_LOW_INITIAL` (0)
- #define `OBJ_AI4_LIMIT_DEADBAND_INITIAL` (2)

Functions

- void `ObjAllInit` (void)
Initialize the analog input objects for this demo application.
- void `ObjAIReset` (void)
Reset the analog input object for this demo application.
- void `ObjAIPresentValueUpdate` (BACNET_INST_NUMBER inst_number, BACNET_REAL value)
Notifies the BACnet API that the property Present_Value has changed.

5.7.1 Macro Definition Documentation

5.7.1.1 #define `OBJ_AI1_COV_INCREMENT_INITIAL` (1)

Definition at line 42 of file `obj_ai.h`.

5.7.1.2 #define OBJ_AI1_DESCRIPTION_INIT_STRING ("Potentiometer 1")

Definition at line 40 of file obj_ai.h.

5.7.1.3 #define OBJ_AI1_LIMIT_DEADBAND_INITIAL (5)

Definition at line 46 of file obj_ai.h.

5.7.1.4 #define OBJ_AI1_LIMIT_HIGH_INITIAL (90)

Definition at line 44 of file obj_ai.h.

5.7.1.5 #define OBJ_AI1_LIMIT_LOW_INITIAL (10)

Definition at line 45 of file obj_ai.h.

5.7.1.6 #define OBJ_AI1_NAME_INIT_STRING ("Demo - AI.1")

Definition at line 39 of file obj_ai.h.

5.7.1.7 #define OBJ_AI1_PRESENT_VALUE_INITIAL (50)

Definition at line 41 of file obj_ai.h.

5.7.1.8 #define OBJ_AI1_UNIT_INITIAL (UNIT_PERCENT)

Definition at line 43 of file obj_ai.h.

5.7.1.9 #define OBJ_AI2_COV_INCREMENT_INITIAL (1)

Definition at line 51 of file obj_ai.h.

5.7.1.10 #define OBJ_AI2_DESCRIPTION_INIT_STRING ("Potentiometer 2")

Definition at line 49 of file obj_ai.h.

5.7.1.11 #define OBJ_AI2_LIMIT_DEADBAND_INITIAL (5)

Definition at line 55 of file obj_ai.h.

5.7.1.12 `#define OBJ_AI2_LIMIT_HIGH_INITIAL (90)`

Definition at line 53 of file obj_ai.h.

5.7.1.13 `#define OBJ_AI2_LIMIT_LOW_INITIAL (10)`

Definition at line 54 of file obj_ai.h.

5.7.1.14 `#define OBJ_AI2_NAME_INIT_STRING ("Demo - AI.2")`

Definition at line 48 of file obj_ai.h.

5.7.1.15 `#define OBJ_AI2_PRESENT_VALUE_INITIAL (50)`

Definition at line 50 of file obj_ai.h.

5.7.1.16 `#define OBJ_AI2_UNIT_INITIAL (UNIT_PERCENT)`

Definition at line 52 of file obj_ai.h.

5.7.1.17 `#define OBJ_AI3_COV_INCREMENT_INITIAL (0.5)`

Definition at line 60 of file obj_ai.h.

5.7.1.18 `#define OBJ_AI3_DESCRIPTION_INIT_STRING ("Temperature 1")`

Definition at line 58 of file obj_ai.h.

5.7.1.19 `#define OBJ_AI3_LIMIT_DEADBAND_INITIAL (2)`

Definition at line 64 of file obj_ai.h.

5.7.1.20 `#define OBJ_AI3_LIMIT_HIGH_INITIAL (50)`

Definition at line 62 of file obj_ai.h.

5.7.1.21 `#define OBJ_AI3_LIMIT_LOW_INITIAL (0)`

Definition at line 63 of file obj_ai.h.

5.7.1.22 `#define OBJ_AI3_NAME_INIT_STRING ("Demo - AI.3")`

Definition at line 57 of file obj_ai.h.

5.7.1.23 `#define OBJ_AI3_PRESENT_VALUE_INITIAL (20)`

Definition at line 59 of file obj_ai.h.

5.7.1.24 `#define OBJ_AI3_UNIT_INITIAL (UNIT_DEGREES_C)`

Definition at line 61 of file obj_ai.h.

5.7.1.25 `#define OBJ_AI4_COV_INCREMENT_INITIAL (0.5)`

Definition at line 69 of file obj_ai.h.

5.7.1.26 `#define OBJ_AI4_DESCRIPTION_INIT_STRING ("Temperature 2")`

Definition at line 67 of file obj_ai.h.

5.7.1.27 `#define OBJ_AI4_LIMIT_DEADBAND_INITIAL (2)`

Definition at line 73 of file obj_ai.h.

5.7.1.28 `#define OBJ_AI4_LIMIT_HIGH_INITIAL (50)`

Definition at line 71 of file obj_ai.h.

5.7.1.29 `#define OBJ_AI4_LIMIT_LOW_INITIAL (0)`

Definition at line 72 of file obj_ai.h.

5.7.1.30 `#define OBJ_AI4_NAME_INIT_STRING ("Demo - AI.4")`

Definition at line 66 of file obj_ai.h.

5.7.1.31 `#define OBJ_AI4_PRESENT_VALUE_INITIAL (20)`

Definition at line 68 of file obj_ai.h.

5.7.1.32 #define OBJ_AI4_UNIT_INITIAL (UNIT_DEGREES_C)

Definition at line 70 of file obj_ai.h.

5.7.2 Function Documentation

5.7.2.1 void ObjAllnit (void)

Initialize the analog input objects for this demo application.

Used here for setting some writable texts.

Definition at line 71 of file obj_ai.c.

5.7.2.2 void ObjAIPresentValueUpdate (BACNET_INST_NUMBER inst_number, BACNET_REAL value)

Notifies the BACnet API that the property Present_Value has changed.

This demo application provide 4 analog input objects. The values for these objects where updated by 2 potentiometers and 2 temperature sensors.

Parameters

in	<i>inst_number</i>	The instance number for the analog input object.
in	<i>value</i>	The new value for the analog input object.

Definition at line 109 of file obj_ai.c.

5.7.2.3 void ObjAIReset (void)

Reset the analog input object for this demo application.

Definition at line 87 of file obj_ai.c.

5.8 src/demo/obj_ao.c File Reference

```
#include <string.h>
#include "obj_ao.h"
#include "user.h"
```

Functions

- void [ObjAOInit](#) (void)
 - Initialize the analog output object for this demo application.*
- void [ObjAOReset](#) (void)
 - Reset the analog output object for this demo application.*
- BACNET_CALLBACK_STATUS [ObjAOWriteHandling](#) (BACNET_INST_NUMBER inst_number, BACNET←_PROPERTY_ID property_id, const BACNET_PROPERTY_CONTENTS *p_property_contents)
 - Notification that a property has changed for an analog output object.*

Variables

- BAC_UINT [g_obj_ao1_description](#) []
- BAC_UINT [g_obj_ao1_evt_msg_txt](#) []
- BACNET_REAL [g_obj_ao1_present_value](#)
- BACNET_BOOLEAN [g_obj_ao1_out_of_service](#)
- BACNET_SHORT_BIT_STRING [g_obj_ao1_state_flags](#)

5.8.1 Function Documentation

5.8.1.1 void ObjAOInit (void)

Initialize the analog output object for this demo application.

Used here for setting some writable texts.

Definition at line 55 of file obj_ao.c.

5.8.1.2 void ObjAOReset (void)

Reset the analog output object for this demo application.

Definition at line 65 of file obj_ao.c.

5.8.1.3 BACNET_CALLBACK_STATUS ObjAOWriteHandling (BACNET_INST_NUMBER *inst_number*, BACNET_PROPERTY_ID *property_id*, const BACNET_PROPERTY_CONTENTS * *p_property_contents*)

Notification that a property has changed for an analog output object.

For this demo application the analog output object is only created to be used by the scheduler. There is no hardware connected which is controlled by that analog output object.

Parameters

in	<i>inst_number</i>	The instance number of the analog output object.
in	<i>property_id</i>	Id of the changed property.
in	<i>p_property_contents</i>	Buffer with content of the property.

Return values

<i>CALLBACK_STATUS_DEFAULT</i>	Always <i>CALLBACK_STATUS_DEFAULT</i> for this demo application.
--------------------------------	--

Definition at line 84 of file obj_ao.c.

5.8.2 Variable Documentation

5.8.2.1 BAC_UINT g_obj_ao1_description[]

5.8.2.2 BAC_UINT g_obj_ao1_evt_msg_txt[]

5.8.2.3 BACNET_BOOLEAN g_obj_ao1_out_of_service

5.8.2.4 BACNET_REAL g_obj_ao1_present_value

5.8.2.5 BACNET_SHORT_BIT_STRING g_obj_ao1_state_flags

5.9 src/demo/obj_ao.h File Reference

```
#include "bacnetdemo.h"
```

Macros

- #define OBJ_AO1_NAME_INIT_STRING ("Demo - AO.1")
- #define OBJ_AO1_DESCRIPTION_INIT_STRING ("Scheduler modified")

Functions

- void [ObjAOInit](#) (void)
Initialize the analog output object for this demo application.
- void [ObjAOReset](#) (void)
Reset the analog output object for this demo application.
- BACNET_CALLBACK_STATUS [ObjAOWriteHandling](#) (BACNET_INST_NUMBER inst_number, BACNET_PROPERTY_ID property_id, const BACNET_PROPERTY_CONTENTS *p_property_contents)
Notification that a property has changed for an analog output object.

5.9.1 Macro Definition Documentation

5.9.1.1 #define OBJ_AO1_DESCRIPTION_INIT_STRING ("Scheduler modified")

Definition at line 36 of file obj_ao.h.

5.9.1.2 #define OBJ_AO1_NAME_INIT_STRING ("Demo - AO.1")

Definition at line 35 of file obj_ao.h.

5.9.2 Function Documentation

5.9.2.1 void ObjA0Init (void)

Initialize the analog output object for this demo application.

Used here for setting some writable texts.

Definition at line 55 of file obj_ao.c.

5.9.2.2 void ObjAOReset (void)

Reset the analog output object for this demo application.

Definition at line 65 of file obj_ao.c.

5.9.2.3 BACNET_CALLBACK_STATUS ObjAOWriteHandling (BACNET_INST_NUMBER *inst_number*, BACNET_PROPERTY_ID *property_id*, const BACNET_PROPERTY_CONTENTS * *p_property_contents*)

Notification that a property has changed for an analog output object.

For this demo application the analog output object is only created to be used by the scheduler. There is no hardware connected which is controlled by that analog output object.

Parameters

in	<i>inst_number</i>	The instance number of the analog output object.
in	<i>property_id</i>	Id of the changed property.
in	<i>p_property_contents</i>	Buffer with content of the property.

Return values

<i>CALLBACK_STATUS_DEFAULT</i>	Always <i>CALLBACK_STATUS_DEFAULT</i> for this demo application.
--------------------------------	--

Definition at line 84 of file obj_ao.c.

5.10 src/demo/obj_bi.c File Reference

```
#include <string.h>
#include "obj_bi.h"
#include "obj_dev.h"
```

Functions

- void [ObjB1Init](#) (void)

Initialize the binary input object for this demo application. Used here for setting some writable texts.

- void [ObjBIReset](#) (void)

Reset the binary input object for this demo application.

- void [ObjBIPresentValueUpdate](#) (BACNET_INST_NUMBER inst_number, BAC_BOOLEAN value)

Notifies the BACnet API that the property Present_Value has changed.

Variables

- BACNET_OBJECT_ID [g_obj_dev_id](#)
- BAC_UINT [g_obj_bi1_description](#) []
- BAC_UINT [g_obj_bi2_description](#) []
- BAC_UINT [g_obj_bi1_evt_msg_txt](#) []
- BAC_UINT [g_obj_bi2_evt_msg_txt](#) []
- BACNET_ENUM [g_obj_bi1_present_value](#)
- BACNET_ENUM [g_obj_bi2_present_value](#)
- BACNET_BOOLEAN [g_obj_bi1_out_of_service](#)
- BACNET_BOOLEAN [g_obj_bi2_out_of_service](#)
- BACNET_SHORT_BIT_STRING [g_obj_bi1_state_flags](#)
- BACNET_SHORT_BIT_STRING [g_obj_bi2_state_flags](#)

5.10.1 Function Documentation

5.10.1.1 void ObjBIInit (void)

Initialize the binary input object for this demo application. Used here for setting some writable texts.

Definition at line 59 of file obj_bi.c.

5.10.1.2 void ObjBIPresentValueUpdate (BACNET_INST_NUMBER inst_number, BAC_BOOLEAN value)

Notifies the BACnet API that the property Present_Value has changed.

This demo application provide 2 binary input objects. The values for these objects where updated by 2 switches.

Parameters

in	<i>inst_number</i>	The instance number for the binary input object.
in	<i>value</i>	The new value for the binary input object.

Definition at line 89 of file obj_bi.c.

5.10.1.3 void ObjBIReset (void)

Reset the binary input object for this demo application.

Definition at line 71 of file obj_bi.c.

5.10.2 Variable Documentation

5.10.2.1 BAC_UINT g_obj_bi1_description[]

5.10.2.2 BAC_UINT g_obj_bi1_evt_msg_txt[]

5.10.2.3 BACNET_BOOLEAN g_obj_bi1_out_of_service

5.10.2.4 BACNET_ENUM g_obj_bi1_present_value

5.10.2.5 BACNET_SHORT_BIT_STRING g_obj_bi1_state_flags

5.10.2.6 BAC_UINT g_obj_bi2_description[]

5.10.2.7 BAC_UINT g_obj_bi2_evt_msg_txt[]

5.10.2.8 BACNET_BOOLEAN g_obj_bi2_out_of_service

5.10.2.9 BACNET_ENUM g_obj_bi2_present_value

5.10.2.10 BACNET_SHORT_BIT_STRING g_obj_bi2_state_flags

5.10.2.11 BACNET_OBJECT_ID g_obj_dev_id

5.11 src/demo/obj_bi.h File Reference

```
#include "bacnetdemo.h"
```

Macros

- #define OBJ_BI1_NAME_INIT_STRING ("Demo - BI.1")
- #define OBJ_BI1_DESCRIPTION_INIT_STRING ("Switch 1")
- #define OBJ_BI1_INACTIVE_INIT_STRING ("off")
- #define OBJ_BI1_ACTIVE_INIT_STRING ("on")
- #define OBJ_BI2_NAME_INIT_STRING ("Demo - BI.2")
- #define OBJ_BI2_DESCRIPTION_INIT_STRING ("Switch 2")
- #define OBJ_BI2_INACTIVE_INIT_STRING ("off")
- #define OBJ_BI2_ACTIVE_INIT_STRING ("on")

Functions

- void `ObjBIInit` (void)
Initialize the binary input object for this demo application. Used here for setting some writable texts.
- void `ObjBIReset` (void)
Reset the binary input object for this demo application.
- void `ObjBIPresentValueUpdate` (BACNET_INST_NUMBER inst_number, BAC_BOOLEAN value)
Notifies the BACnet API that the property Present_Value has changed.

5.11.1 Macro Definition Documentation

5.11.1.1 `#define OBJ_BI1_ACTIVE_INIT_STRING ("on")`

Definition at line 36 of file `obj_bi.h`.

5.11.1.2 `#define OBJ_BI1_DESCRIPTION_INIT_STRING ("Switch 1")`

Definition at line 34 of file `obj_bi.h`.

5.11.1.3 `#define OBJ_BI1_INACTIVE_INIT_STRING ("off")`

Definition at line 35 of file `obj_bi.h`.

5.11.1.4 `#define OBJ_BI1_NAME_INIT_STRING ("Demo - BI.1")`

Definition at line 33 of file `obj_bi.h`.

5.11.1.5 `#define OBJ_BI2_ACTIVE_INIT_STRING ("on")`

Definition at line 41 of file `obj_bi.h`.

5.11.1.6 `#define OBJ_BI2_DESCRIPTION_INIT_STRING ("Switch 2")`

Definition at line 39 of file `obj_bi.h`.

5.11.1.7 `#define OBJ_BI2_INACTIVE_INIT_STRING ("off")`

Definition at line 40 of file `obj_bi.h`.

5.11.1.8 `#define OBJ_BI2_NAME_INIT_STRING ("Demo - BI.2")`

Definition at line 38 of file `obj_bi.h`.

5.11.2 Function Documentation

5.11.2.1 `void ObjBIInit (void)`

Initialize the binary input object for this demo application. Used here for setting some writable texts.

Definition at line 59 of file `obj_bi.c`.

5.11.2.2 `void ObjBIPresentValueUpdate (BACNET_INST_NUMBER inst_number, BAC_BOOLEAN value)`

Notifies the BACnet API that the property `Present_Value` has changed.

This demo application provide 2 binary input objects. The values for these objects where updated by 2 switches.

Parameters

in	<i>inst_number</i>	The instance number for the binary input object.
in	<i>value</i>	The new value for the binary input object.

Definition at line 89 of file obj_bi.c.

5.11.2.3 void ObjBIReset (void)

Reset the binary input object for this demo application.

Definition at line 71 of file obj_bi.c.

5.12 src/demo/obj_bo.c File Reference

```
#include <string.h>
#include "obj_bo.h"
#include "obj_dev.h"
#include "user.h"
```

Functions

- void [ObjBOInit](#) (void)

Initialize the binary output object for this demo application. Used here for setting some writable texts.
- void [ObjBOReset](#) (void)

Reset the binary output object for this demo application.
- BACNET_CALLBACK_STATUS [ObjBOWriteHandling](#) (BACNET_INST_NUMBER inst_number, BACNET↔
_PROPERTY_ID property_id, const BACNET_PROPERTY_CONTENTS *p_property_contents)

Notification that a property has changed for a binary output object.

Variables

- BAC_UINT [g_obj_bo1_description](#) []
- BAC_UINT [g_obj_bo2_description](#) []
- BACNET_ENUM [g_obj_bo1_present_value](#)
- BACNET_ENUM [g_obj_bo2_present_value](#)
- BACNET_BOOLEAN [g_obj_bo1_out_of_service](#)
- BACNET_BOOLEAN [g_obj_bo2_out_of_service](#)
- BACNET_SHORT_BIT_STRING [g_obj_bo1_state_flags](#)
- BACNET_SHORT_BIT_STRING [g_obj_bo2_state_flags](#)

5.12.1 Function Documentation

5.12.1.1 void ObjBOInit (void)

Initialize the binary output object for this demo application. Used here for setting some writable texts.

Definition at line 57 of file obj_bo.c.

5.12.1.2 void ObjBOReset (void)

Reset the binary output object for this demo application.

Definition at line 66 of file obj_bo.c.

5.12.1.3 BACNET_CALLBACK_STATUS ObjBOWriteHandling (BACNET_INST_NUMBER *inst_number*, BACNET_PROPERTY_ID *property_id*, const BACNET_PROPERTY_CONTENTS * *p_property_contents*)

Notification that a property has changed for a binary output object.

For this demo application only a changed Present_Value for the binary output objects will be processed. This property controls the state of a LED. The LED will glow if the Present_Value of the corresponding binary object is on.

Parameters

in	<i>inst_number</i>	The instance number of the binary output object.
in	<i>property_id</i>	Id of the changed property.
in	<i>p_property_contents</i>	Buffer with content of the property.

Return values

<i>CALLBACK_STATUS_DEFAULT</i>	Always <i>CALLBACK_STATUS_DEFAULT</i> for this demo application.
--------------------------------	--

Definition at line 91 of file obj_bo.c.

5.12.2 Variable Documentation

5.12.2.1 BAC_UINT g_obj_bo1_description[]

5.12.2.2 BACNET_BOOLEAN g_obj_bo1_out_of_service

5.12.2.3 BACNET_ENUM g_obj_bo1_present_value

5.12.2.4 BACNET_SHORT_BIT_STRING g_obj_bo1_state_flags

5.12.2.5 BAC_UINT g_obj_bo2_description[]

5.12.2.6 BACNET_BOOLEAN g_obj_bo2_out_of_service

5.12.2.7 BACNET_ENUM g_obj_bo2_present_value

5.12.2.8 BACNET_SHORT_BIT_STRING g_obj_bo2_state_flags

5.13 src/demo/obj_bo.h File Reference

```
#include "bacnetdemo.h"
```

Macros

- #define `OBJ_BO1_NAME_INIT_STRING` ("Demo - BO.1")
- #define `OBJ_BO1_DESCRIPTION_INIT_STRING` ("LED 1")
- #define `OBJ_BO1_INACTIVE_INIT_STRING` ("off")
- #define `OBJ_BO1_ACTIVE_INIT_STRING` ("on")
- #define `OBJ_BO2_NAME_INIT_STRING` ("Demo - BO.2")
- #define `OBJ_BO2_DESCRIPTION_INIT_STRING` ("LED 2")
- #define `OBJ_BO2_INACTIVE_INIT_STRING` ("off")
- #define `OBJ_BO2_ACTIVE_INIT_STRING` ("on")

Functions

- void `ObjBOInit` (void)
Initialize the binary output object for this demo application. Used here for setting some writable texts.
- void `ObjBOReset` (void)
Reset the binary output object for this demo application.
- BACNET_CALLBACK_STATUS `ObjBOWriteHandling` (BACNET_INST_NUMBER inst_number, BACNET←
_PROPERTY_ID property_id, const BACNET_PROPERTY_CONTENTS *p_property_contents)
Notification that a property has changed for a binary output object.

5.13.1 Macro Definition Documentation

5.13.1.1 #define OBJ_BO1_ACTIVE_INIT_STRING ("on")

Definition at line 38 of file obj_bo.h.

5.13.1.2 #define OBJ_BO1_DESCRIPTION_INIT_STRING ("LED 1")

Definition at line 36 of file obj_bo.h.

5.13.1.3 #define OBJ_BO1_INACTIVE_INIT_STRING ("off")

Definition at line 37 of file obj_bo.h.

5.13.1.4 #define OBJ_BO1_NAME_INIT_STRING ("Demo - BO.1")

Definition at line 35 of file obj_bo.h.

5.13.1.5 #define OBJ_BO2_ACTIVE_INIT_STRING ("on")

Definition at line 43 of file obj_bo.h.

5.13.1.6 `#define OBJ_BO2_DESCRIPTION_INIT_STRING ("LED 2")`

Definition at line 41 of file `obj_bo.h`.

5.13.1.7 `#define OBJ_BO2_INACTIVE_INIT_STRING ("off")`

Definition at line 42 of file `obj_bo.h`.

5.13.1.8 `#define OBJ_BO2_NAME_INIT_STRING ("Demo - BO.2")`

Definition at line 40 of file `obj_bo.h`.

5.13.2 Function Documentation

5.13.2.1 `void ObjBOInit (void)`

Initialize the binary output object for this demo application. Used here for setting some writable texts.

Definition at line 57 of file `obj_bo.c`.

5.13.2.2 `void ObjBOReset (void)`

Reset the binary output object for this demo application.

Definition at line 66 of file `obj_bo.c`.

5.13.2.3 `BACNET_CALLBACK_STATUS ObjBOWriteHandling (BACNET_INST_NUMBER inst_number, BACNET_PROPERTY_ID property_id, const BACNET_PROPERTY_CONTENTS * p_property_contents)`

Notification that a property has changed for a binary output object.

For this demo application only a changed `Present_Value` for the binary output objects will be processed. This property controls the state of a LED. The LED will glow if the `Present_Value` of the corresponding binary object is on.

Parameters

in	<i>inst_number</i>	The instance number of the binary output object.
in	<i>property_id</i>	Id of the changed property.
in	<i>p_property_contents</i>	Buffer with content of the property.

Return values

<code>CALLBACK_STATUS_DEFAULT</code>	Always <code>CALLBACK_STATUS_DEFAULT</code> for this demo application.
--------------------------------------	--

Definition at line 91 of file obj_bo.c.

5.14 src/demo/obj_ca.c File Reference

```
#include <stdbool.h>
#include <stdint.h>
#include <string.h>
#include "bacnet.h"
#include "obj_ca.h"
#include "obj_dev.h"
```

Functions

- void [ObjCAInit](#) (void)
Initialize the schedule object for this demo application. Used here for setting some writable texts.

Variables

- BAC_UINT [g_obj_ca1_description](#) []

5.14.1 Function Documentation

5.14.1.1 void ObjCAInit (void)

Initialize the schedule object for this demo application. Used here for setting some writable texts.

Definition at line 56 of file obj_ca.c.

5.14.2 Variable Documentation

5.14.2.1 BAC_UINT g_obj_ca1_description[]

5.15 src/demo/obj_ca.h File Reference

```
#include "bacnetdemo.h"
```

Macros

- #define [OBJ_CA1_NAME_INIT_STRING](#) ("Demo - CA.1")
- #define [OBJ_CA1_DESCRIPTION_INIT_STRING](#) ("Calendar for the scheduler")

Functions

- void [ObjCAInit](#) (void)

Initialize the schedule object for this demo application. Used here for setting some writable texts.

5.15.1 Macro Definition Documentation

5.15.1.1 #define OBJ_CA1_DESCRIPTION_INIT_STRING ("Calendar for the scheduler")

Definition at line 33 of file obj_ca.h.

5.15.1.2 #define OBJ_CA1_NAME_INIT_STRING ("Demo - CA.1")

Definition at line 32 of file obj_ca.h.

5.15.2 Function Documentation

5.15.2.1 void ObjCAInit (void)

Initialize the schedule object for this demo application. Used here for setting some writable texts.

Definition at line 56 of file obj_ca.c.

5.16 src/demo/obj_dev.c File Reference

```
#include <string.h>
#include <stdio.h>
#include "obj_dev.h"
```

Functions

- void [ObjDevInit](#) (BACNET_TIME_STAMP *p_start_time)

Initialize the device object for this demo application. Used here for setting the instance number of the device and some writable texts.

Variables

- BACNET_OBJECT_ID [g_obj_dev_id](#)
- BAC_UINT [g_obj_dev_name](#) []
- BAC_UINT [g_obj_dev_description](#) []
- BAC_UINT [g_obj_dev_location](#) []
- BACNET_TIME_STAMP [g_obj_dev_time_of_restart](#) []

5.16.1 Function Documentation

5.16.1.1 void ObjDevInit (BACNET_TIME_STAMP * p_start_time)

Initialize the device object for this demo application. Used here for setting the instance number of the device and some writable texts.

Definition at line 52 of file obj_dev.c.

5.16.2 Variable Documentation

5.16.2.1 BAC_UINT g_obj_dev_description[]

5.16.2.2 BACNET_OBJECT_ID g_obj_dev_id

5.16.2.3 BAC_UINT g_obj_dev_location[]

5.16.2.4 BAC_UINT g_obj_dev_name[]

5.16.2.5 BACNET_TIME_STAMP g_obj_dev_time_of_restart[]

5.17 src/demo/obj_dev.h File Reference

```
#include "bacnetdemo.h"
```

Macros

- #define OBJ_DEV_INST_NUMBER (2000)
- #define OBJ_DEV_NAME_INIT_STRING ("Demo - Device")
- #define OBJ_DEV_DESCRIPTION_INIT_STRING ("BACnet API for Renesas Synergy™")
- #define OBJ_DEV_LOCATION_INIT_STRING ("Demo - Board")
- #define OBJ_DEV_VENDOR_NAME_STRING ("CS Lab GmbH")
- #define OBJ_DEV_VENDOR_ID_NUMBER (794)
- #define OBJ_DEV_MODEL_NAME_STRING ("Demo Renesas Synergy™")
- #define OBJ_DEV_FIRMWARE_REV_STRING ("FW 1.0.0.0")
- #define OBJ_DEV_SOFTWARE_VER_STRING ("SW 1.0.0.0")
- #define OBJ_DEV_MAX_APDU_SIZE (1024)

Functions

- void ObjDevInit (BACNET_TIME_STAMP *p_start_time)

Initialize the device object for this demo application. Used here for setting the instance number of the device and some writable texts.

5.17.1 Macro Definition Documentation

5.17.1.1 #define OBJ_DEV_DESCRIPTION_INIT_STRING ("BACnet API for Renesas Synergy™")

Definition at line 25 of file obj_dev.h.

5.17.1.2 #define OBJ_DEV_FIRMWARE_REV_STRING ("FW 1.0.0.0")

Definition at line 30 of file obj_dev.h.

5.17.1.3 #define OBJ_DEV_INST_NUMBER (2000)

Definition at line 22 of file obj_dev.h.

5.17.1.4 #define OBJ_DEV_LOCATION_INIT_STRING ("Demo - Board")

Definition at line 26 of file obj_dev.h.

5.17.1.5 #define OBJ_DEV_MAX_APDU_SIZE (1024)

Definition at line 33 of file obj_dev.h.

5.17.1.6 #define OBJ_DEV_MODEL_NAME_STRING ("Demo Renesas Synergy™")

Definition at line 29 of file obj_dev.h.

5.17.1.7 #define OBJ_DEV_NAME_INIT_STRING ("Demo - Device")

Definition at line 24 of file obj_dev.h.

5.17.1.8 #define OBJ_DEV_SOFTWARE_VER_STRING ("SW 1.0.0.0")

Definition at line 31 of file obj_dev.h.

5.17.1.9 #define OBJ_DEV_VENDOR_ID_NUMBER (794)

Definition at line 28 of file obj_dev.h.

5.17.1.10 #define OBJ_DEV_VENDOR_NAME_STRING ("CS Lab GmbH")

Definition at line 27 of file obj_dev.h.

5.17.2 Function Documentation

5.17.2.1 void ObjDevInit (BACNET_TIME_STAMP * p_start_time)

Initialize the device object for this demo application. Used here for setting the instance number of the device and some writable texts.

Definition at line 52 of file obj_dev.c.

5.18 src/demo/obj_nc.c File Reference

```
#include <stdbool.h>
#include <stdint.h>
#include <string.h>
#include "bacnet.h"
#include "obj_nc.h"
#include "obj_dev.h"
```

Functions

- void [ObjNCInit](#) (void)
Initialize the schedule object for this demo application. Used here for setting some writable texts.

Variables

- BAC_UINT [g_obj_nc1_description](#) []

5.18.1 Function Documentation

5.18.1.1 void ObjNCInit (void)

Initialize the schedule object for this demo application. Used here for setting some writable texts.

Definition at line 56 of file obj_nc.c.

5.18.2 Variable Documentation

5.18.2.1 BAC_UINT g_obj_nc1_description[]

5.19 src/demo/obj_nc.h File Reference

```
#include "bacnetdemo.h"
```

Macros

- `#define OBJ_NC1_NAME_INIT_STRING ("Demo - NC.1")`
- `#define OBJ_NC1_DESCRIPTION_INIT_STRING ("Notification class")`

Functions

- `void ObjNCInit (void)`
Initialize the schedule object for this demo application. Used here for setting some writable texts.

5.19.1 Macro Definition Documentation

5.19.1.1 `#define OBJ_NC1_DESCRIPTION_INIT_STRING ("Notification class")`

Definition at line 35 of file `obj_nc.h`.

5.19.1.2 `#define OBJ_NC1_NAME_INIT_STRING ("Demo - NC.1")`

Definition at line 34 of file `obj_nc.h`.

5.19.2 Function Documentation

5.19.2.1 `void ObjNCInit (void)`

Initialize the schedule object for this demo application. Used here for setting some writable texts.

Definition at line 56 of file `obj_nc.c`.

5.20 `src/demo/obj_sc.c` File Reference

```
#include <stdbool.h>
#include <stdint.h>
#include <string.h>
#include "bacnet.h"
#include "obj_sc.h"
#include "obj_dev.h"
```

Functions

- `void ObjSCInit (void)`
Initialize the schedule object for this demo application. Used here for setting some writable texts.
- `void ObjSCReset (void)`
Reset the schedule object for this demo application.

Variables

- BAC_UINT [g_obj_sc1_description](#) []
- BACNET_BOOLEAN [g_obj_sc1_out_of_service](#)
- BACNET_SHORT_BIT_STRING [g_obj_sc1_state_flags](#)

5.20.1 Function Documentation

5.20.1.1 void ObjSCInit (void)

Initialize the schedule object for this demo application. Used here for setting some writable texts.

Definition at line 58 of file obj_sc.c.

5.20.1.2 void ObjSCReset (void)

Reset the schedule object for this demo application.

Definition at line 66 of file obj_sc.c.

5.20.2 Variable Documentation

5.20.2.1 BAC_UINT g_obj_sc1_description[]

5.20.2.2 BACNET_BOOLEAN g_obj_sc1_out_of_service

5.20.2.3 BACNET_SHORT_BIT_STRING g_obj_sc1_state_flags

5.21 src/demo/obj_sc.h File Reference

```
#include "bacnetdemo.h"
```

Macros

- #define [OBJ_SC1_DAILY_SCHEDULE_SIZE](#) (24)
- #define [OBJ_SC1_NAME_INIT_STRING](#) ("Demo - SC.1")
- #define [OBJ_SC1_DESCRIPTION_INIT_STRING](#) ("Every full hour")

Functions

- void [ObjSCInit](#) (void)
Initialize the schedule object for this demo application. Used here for setting some writable texts.
- void [ObjSCReset](#) (void)
Reset the schedule object for this demo application.

5.21.1 Macro Definition Documentation

5.21.1.1 #define OBJ_SC1_DAILY_SCHEDULE_SIZE (24)

Definition at line 38 of file obj_sc.h.

5.21.1.2 #define OBJ_SC1_DESCRIPTION_INIT_STRING ("Every full hour")

Definition at line 41 of file obj_sc.h.

5.21.1.3 #define OBJ_SC1_NAME_INIT_STRING ("Demo - SC.1")

Definition at line 40 of file obj_sc.h.

5.21.2 Function Documentation

5.21.2.1 void ObjSCInit (void)

Initialize the schedule object for this demo application. Used here for setting some writable texts.

Definition at line 58 of file obj_sc.c.

5.21.2.2 void ObjSCReset (void)

Reset the schedule object for this demo application.

Definition at line 66 of file obj_sc.c.

5.22 src/demo/obj_tr.c File Reference

```
#include <stdbool.h>
#include <stdint.h>
#include <string.h>
#include "bacnet.h"
#include "obj_tr.h"
#include "obj_dev.h"
#include "user.h"
```

Functions

- void [ObjTRInit](#) (void)
Initialize the trend log object for this demo application. Used here for setting some writable texts.
- BACNET_CALLBACK_STATUS [ObjTRWriteHandling](#) (BACNET_INST_NUMBER inst_number, BACNET↔
_PROPERTY_ID property_id, const BACNET_PROPERTY_CONTENTS *p_property_contents)
Notification that a property has changed for a trend log object.

Variables

- BAC_UINT [g_obj_tr1_description](#) []
- BAC_UINT [g_obj_tr1_evt_msg_txt](#) []

5.22.1 Function Documentation

5.22.1.1 void ObjTRInit (void)

Initialize the trend log object for this demo application. Used here for setting some writable texts.

Definition at line 58 of file obj_tr.c.

5.22.1.2 BACNET_CALLBACK_STATUS ObjTRWriteHandling (BACNET_INST_NUMBER *inst_number*, BACNET_PROPERTY_ID *property_id*, const BACNET_PROPERTY_CONTENTS * *p_property_contents*)

Notification that a property has changed for a trend log object.

The BACnet API is only an interface, the data handling must be done on user side.

Parameters

in	<i>inst_number</i>	The instance number of the trend log object.
in	<i>property_id</i>	Id of the changed property.
in	<i>p_property_contents</i>	Buffer with content of the property.

Return values

<i>CALLBACK_STATUS_DEFAULT</i>	Always <i>CALLBACK_STATUS_DEFAULT</i> for this demo application.
--------------------------------	--

Definition at line 76 of file obj_tr.c.

5.22.2 Variable Documentation

5.22.2.1 BAC_UINT g_obj_tr1_description[]

5.22.2.2 BAC_UINT g_obj_tr1_evt_msg_txt[]

5.23 src/demo/obj_tr.h File Reference

```
#include "bacnetdemo.h"
```

Macros

- #define `OBJ_TR1_NAME_INIT_STRING` ("Demo - TR.1")
- #define `OBJ_TR1_DESCRIPTION_INIT_STRING` ("Trend for AI.1")
- #define `OBJ_TR1_LOG_BUFFER_SIZE` (20)

Functions

- void `ObjTRInit` (void)
 - Initialize the trend log object for this demo application. Used here for setting some writable texts.*
- BACNET_CALLBACK_STATUS `ObjTRWriteHandling` (BACNET_INST_NUMBER *inst_number*, BACNET_PROPERTY_ID *property_id*, const BACNET_PROPERTY_CONTENTS **p_property_contents*)
 - Notification that a property has changed for a trend log object.*

5.23.1 Macro Definition Documentation

5.23.1.1 #define OBJ_TR1_DESCRIPTION_INIT_STRING ("Trend for AI.1")

Definition at line 40 of file `obj_tr.h`.

5.23.1.2 #define OBJ_TR1_LOG_BUFFER_SIZE (20)

Definition at line 41 of file `obj_tr.h`.

5.23.1.3 #define OBJ_TR1_NAME_INIT_STRING ("Demo - TR.1")

Definition at line 39 of file `obj_tr.h`.

5.23.2 Function Documentation

5.23.2.1 void ObjTRInit (void)

Initialize the trend log object for this demo application. Used here for setting some writable texts.

Definition at line 58 of file `obj_tr.c`.

5.23.2.2 BACNET_CALLBACK_STATUS ObjTRWriteHandling (BACNET_INST_NUMBER *inst_number*, BACNET_PROPERTY_ID *property_id*, const BACNET_PROPERTY_CONTENTS * *p_property_contents*)

Notification that a property has changed for a trend log object.

The BACnet API is only an interface, the data handling must be done on user side.

Parameters

in	<i>inst_number</i>	The instance number of the trend log object.
in	<i>property_id</i>	Id of the changed property.
in	<i>p_property_contents</i>	Buffer with content of the property.

Return values

<code>CALLBACK_STATUS_DEFAULT</code>	Always <code>CALLBACK_STATUS_DEFAULT</code> for this demo application.
--------------------------------------	--

Definition at line 76 of file obj_tr.c.

5.24 src/demo/user.c File Reference

```
#include <math.h>
#include "tx_api.h"
#include "hal_data.h"
#include "adc.h"
#include "user.h"
#include "obj_bi.h"
#include "obj_bo.h"
#include "obj_ai.h"
#include "obj_ao.h"
#include "obj_sc.h"
#include "obj_dev.h"
```

Data Structures

- struct [user_button_t](#)
- struct [user_binary_input_t](#)
- struct [user_analog_input_t](#)
- struct [user_trendlog_t](#)

Macros

- `#define` [USER_MAX_BINARY](#) (2)
- `#define` [USER_MAX_ANALOG](#) (4)
- `#define` [USER_MAX_RECORDS](#) (20)
- `#define` [USER_MIN_VALUE_DIFF](#) (0.01)
- `#define` [USER_BINARY_VALUE_IS_BUTTON](#) (false)
- `#define` [USER_ANALOG_AVERAGE_COUNT](#) (50)
- `#define` [FLASH_DEVICE_PAGE_SIZE](#) (64)
- `#define` [FLASH_DEVICE_FLASH_SIZE](#) ([FLASH_DEVICE_PAGE_SIZE](#) * 1024)
- `#define` [FLASH_DEVICE_START_ADDRESS](#) (0x40100000)
- `#define` [FLASH_MEM_MAGIC_SIZE](#) (4)

Functions

- void `user_loop` (ULONG entry_input)
Checking the state of the input devices (Switches, potentiometers and temperature sensors).
- static bool `show_led` (uint32_t index, bool on_off)
Setting the state of a LED.
- static bool `button_pressed` (uint8_t index)
Checking the state of a button.
- static bool `get_switch_state` (uint8_t index, bool *p_switch_state)
Checking the state of a switch.
- static bool `analog_value_changed` (uint8_t index, uint16_t adc_value)
Calculate the average value for an analog input and compare it with the current stored value.
- static float `calc_analog_value` (uint8_t index, uint16_t adc_value, float resolution)
Calculate the float value for an ADC value.
- static void `init_scheduler` (void)
Initialize the scheduler with user defined settings.
- static void `init_trendlog` (void)
Initialize the structure for the trend log data.
- static bool `is_trendlog_index_valid` (uint32_t index)
Check if the entry with the given index is valid or not. True for all entries up to the array limit except the array is still not completed.
- static uint32_t `get_trendlog_index_next` (uint32_t index)
Getting the next index in the trend log array. Looping between 0 and array limit.
- static uint32_t `get_trendlog_index_prev` (uint32_t index)
Getting the previous index in the trend log array. Looping between 0 and array limit.
- static uint32_t `get_trendlog_index_start` (void)
Getting the start index for the trend log array. Is 0 for a not completed array. After completion the start index is running between 0 and array limit.
- static bool `is_trendlog_time_smaller` (BACNET_DATE_TIME record_time, BACNET_DATE_TIME end_time, bool or_equal)
Compare 2 BACNET_DATE_TIME's.
- static bool `copy_trendlog_data` (uint32_t index, uint32_t count, BACNET_LOG_RECORD p_records[], uint32_t *p_count)
Copy data from internal memory to the given array p_records.
- static void `get_trendlog_first_last` (BACNET_LOG_RECORD p_records[], uint32_t count, bool *p_first, bool *p_last)
Check if the trend log array contains the first and last record.
- static void `user_reset_values` (VOID)
Set the user variables to initial state.
- static bool `is_property_persistent` (const BACNET_SERVER_PROPERTY_INSTANCE *p_prop)
User decision if the given property should be stored or not.
- static BAC_BYTE `get_crc_checksum` (BAC_BYTE check, BAC_BYTE memory)
Calculate the crc checksum for the stored data.
- static void `flash_wait` (void)
Wait until flash is ready for next operation.
- static bool `flash_write` (uint32_t flash_address, uint32_t source_address)
Write data to the flash memory.
- static bool `is_flash_magic_activated` (void)
Check if the flash magic token is set or not.
- static void `set_flash_magic` (void)
Set the flash magic token.
- static bool `flash_write_page` (BAC_BYTE *p_buffer, BAC_WORD buffer_size)

- Function which writes a page of changed persistent property data.*

 - bool [UserInit](#) (void)
 - Initialize the user interface.*
 - void [UserReset](#) (void)
 - Reset the user interface.*
 - bool [UserSetLed](#) (uint32_t number, bool on_off)
 - Setting the state of the LED for the binary output objects present value.*
 - bool [UserTrendlogClearBuffer](#) (uint32_t number)
 - Clears the internal memory to store the trend log records.*
 - bool [UserTrendlogPutRecord](#) (uint32_t number, BACNET_LOG_RECORD *p_record)
 - Stores a record in the internal memory for a trend log object.*
 - bool [UserTrendlogGetRecords](#) (uint32_t number, BACNET_LOG_RECORD p_records[], uint32_t *p_count, bool *p_first, bool *p_last, bool *p_more)
 - Getting all stored records for a trend log object.*
 - bool [UserTrendlogGetRecordsByPosition](#) (uint32_t number, uint32_t position, int32_t count, BACNET_LOG_RECORD p_records[], uint32_t *p_count, bool *p_first, bool *p_last, bool *p_more)
 - Getting stored records for a trend log object. Beginning with the given record position and limited by the given count.*
 - bool [UserTrendlogGetRecordsBySequence](#) (uint32_t number, uint32_t sequence, int32_t count, BACNET_LOG_RECORD p_records[], uint32_t *p_count, bool *p_first, bool *p_last, bool *p_more)
 - Getting stored records for a trend log object. Beginning with the record with the given sequence number and limited by the given count.*
 - bool [UserTrendlogGetRecordsByTime](#) (uint32_t number, BACNET_DATE_TIME start_time, int32_t count, BACNET_LOG_RECORD p_records[], uint32_t *p_count, bool *p_first, bool *p_last, bool *p_more)
 - Getting stored records for a trend log object. Beginning with the record with a time stamp greater or equal to the given start_time and limited by the given count.*
 - bool [UserTrendlogGetRecordsByTimeRange](#) (uint32_t number, BACNET_DATE_TIME start_time, BACNET_DATE_TIME end_time, BACNET_LOG_RECORD p_records[], uint32_t *p_count, bool *p_first, bool *p_last, bool *p_more)
 - Getting stored records for a trend log object. Beginning with the record with a time stamp greater or equal to the given start_time and limited by the given end_time.*
 - bool [UserStorageIsPropertyPersistent](#) (BACNET_INST_NUMBER device_id, BACNET_OBJECT_TYPE obj_type, BACNET_INST_NUMBER inst_number, BACNET_PROPERTY_ID property_id)
 - Request if the given property should be stored persistent.*
 - bool [UserStorageRead](#) (const BACNET_SRVR_INIT *p_server)
 - Function which reads out the data stored in the flash pages and copies them into the object property structures.*
 - bool [UserStorageWrite](#) (const BACNET_SRVR_INIT *p_server)
 - Function which writes changed persistent property data.*
 - bool [UserStorageDelete](#) (const BACNET_SRVR_INIT *p_server)
 - Request for deleting the persistent stored data.*

Variables

- BAC_UINT [g_obj_sc1_daily_schedule](#) [CALC_WRITABLE_DAILY_SCHEDULE_ARRAY_SIZE(OBJ_SC1_DAILY_SCHEDULE_SIZE)]
- BACNET_DEV_OBJ_PROP_REFERENCE [g_obj_sc1_obj_prop_ref_list](#) []
- BAC_UINT [g_obj_sc1_obj_prop_ref_listCount](#)
- static [user_button_t](#) [g_user_button](#) [2]
- static const [ioport_port_pin_t](#) [g_user_led_pins](#) []
- static const [bsp_leds_t](#) [g_user_leds](#)
- static [user_binary_input_t](#) [g_user_binary_values](#) [USER_MAX_BINARY]
- static [user_analog_input_t](#) [g_user_analog_values](#) [USER_MAX_ANALOG]
- static [user_trendlog_t](#) [g_user_trendlog](#)

- static uint32_t [g_sequence](#)
- static TX_THREAD [g_user_thread](#)
- static uint8_t [g_user_thread_stack](#) [0x1800]
- static TX_MUTEX [g_user_mutex](#)
- static BAC_BYTE [g_flash_mem_page](#) [FLASH_DEVICE_PAGE_SIZE]
- static BAC_BYTE * [gp_flash_mem](#) = (BAC_BYTE *)FLASH_DEVICE_START_ADDRESS
- static BAC_BYTE [g_flash_magic](#) [FLASH_MEM_MAGIC_SIZE] = {0xEF, 0xBE, 0xC0, 0xBA}
- static BAC_WORD [g_flash_offset](#)
- static BAC_BYTE [g_flash_checksum](#)
- static BAC_WORD [g_page_offset](#)
- static BAC_BOOLEAN [g_page_is_new](#)
- static BAC_BOOLEAN [g_page_data_changed](#)
- static BAC_BOOLEAN [g_page_any_page_changed](#)

5.24.1 Macro Definition Documentation

5.24.1.1 #define FLASH_DEVICE_FLASH_SIZE (FLASH_DEVICE_PAGE_SIZE * 1024)

Definition at line 62 of file user.c.

5.24.1.2 #define FLASH_DEVICE_PAGE_SIZE (64)

Definition at line 60 of file user.c.

5.24.1.3 #define FLASH_DEVICE_START_ADDRESS (0x40100000)

Definition at line 64 of file user.c.

5.24.1.4 #define FLASH_MEM_MAGIC_SIZE (4)

Definition at line 66 of file user.c.

5.24.1.5 #define USER_ANALOG_AVERAGE_COUNT (50)

Definition at line 57 of file user.c.

5.24.1.6 #define USER_BINARY_VALUE_IS_BUTTON (false)

Definition at line 52 of file user.c.

5.24.1.7 #define USER_MAX_ANALOG (4)

Definition at line 39 of file user.c.

5.24.1.8 #define USER_MAX_BINARY (2)

Definition at line 38 of file user.c.

5.24.1.9 #define USER_MAX_RECORDS (20)

Definition at line 40 of file user.c.

5.24.1.10 #define USER_MIN_VALUE_DIFF (0.01)

Definition at line 41 of file user.c.

5.24.2 Function Documentation

5.24.2.1 static bool analog_value_changed (uint8_t *index*, uint16_t *adc_value*) [static]

Calculate the average value for an analog input and compare it with the current stored value.

Parameters

in	<i>index</i>	Index for the ADC value.
in	<i>adc_value</i>	New measured value for the analog input.

Returns

True, if there is a new calculated value for the analog input.

Definition at line 259 of file user.c.

5.24.2.2 static bool button_pressed (uint8_t *index*) [static]

Checking the state of a button.

Parameters

in	<i>index</i>	Index of the button.
----	--------------	----------------------

Return values

<i>true</i>	Button state changed from unpressed to pressed pressed.
<i>false</i>	Button was released or is still pressed.

Definition at line 189 of file user.c.

5.24.2.3 `static float calc_analog_value (uint8_t index, uint16_t adc_value, float resolution) [static]`

Calculate the float value for an ADC value.

A ADC value is a 12 bit value between 0 and 4095.

Parameters

in	<i>index</i>	Index for the ADC value.
in	<i>adc_value</i>	ADC value from ADC Interface.
in	<i>resolution</i>	Resolution of the ADC value.

Return values

<i>true</i>	Button state changed from released to pressed.
<i>false</i>	Button was released or is still pressed.

Definition at line 301 of file user.c.

5.24.2.4 `static bool copy_trendlog_data (uint32_t index, uint32_t count, BACNET_LOG_RECORD p_records[], uint32_t * p_count) [static]`

Copy data from internal memory to the given array p_records.

Parameters

in	<i>index</i>	Start index in the trend log array.
in	<i>count</i>	Maximum count of entries to copy.
out	<i>p_records</i>	Array which hold the returned records.
out	<i>p_count</i>	Number of records really copied.

Returns

Returns true if the records could be copied from internal memory to the array p_records.

Definition at line 575 of file user.c.

5.24.2.5 `static void flash_wait (void) [static]`

Wait until flash is ready for next operation.

Definition at line 772 of file user.c.

5.24.2.6 `static bool flash_write (uint32_t flash_address, uint32_t source_address) [static]`

Write data to the flash memory.

Parameters

in	<i>flash_address</i>	Address of the flash memory.
in	<i>source_address</i>	Address of the source.

Returns

Returns true if the data could be written to the flash memory.

Definition at line 795 of file user.c.

5.24.2.7 `static bool flash_write_page (BAC_BYTE * p_buffer, BAC_WORD buffer_size) [static]`

Function which writes a page of changed persistent property data.

Parameters

in	<i>p_buffer</i>	Pointer to the data which should be stored.
in	<i>buffer_size</i>	Size of the data.

Returns

Returns true if the data could be written to the flash memory.

Definition at line 874 of file user.c.

5.24.2.8 `static BAC_BYTE get_crc_checksum (BAC_BYTE check, BAC_BYTE memory) [static]`

Calculate the crc checksum for the stored data.

Parameters

in	<i>check</i>	Current checksum.
in	<i>memory</i>	New byte in memory.

Returns

Returns the checksum for the input parameter.

Definition at line 756 of file user.c.

5.24.2.9 `static bool get_switch_state (uint8_t index, bool * p_switch_state) [static]`

Checking the state of a switch.

Parameters

in	<i>index</i>	Index of the switch.
out	<i>p_switch_state</i>	Switch setting.

Returns

Switch setting has changed or not.

Definition at line 230 of file user.c.

5.24.2.10 `static void get_trendlog_first_last (BACNET_LOG_RECORD p_records[], uint32_t count, bool * p_first, bool * p_last) [static]`

Check if the trend log array contains the first and last record.

Parameters

in	<i>p_records</i>	Array with trend log records.
in	<i>count</i>	Maximum count of entries in the trend log array.
out	<i>p_first</i>	Flag if the first record is in the trend log array.
out	<i>p_last</i>	Flag if the last record is in the trend log array.

Definition at line 616 of file user.c.

5.24.2.11 `static uint32_t get_trendlog_index_next (uint32_t index) [static]`

Getting the next index in the trend log array. Looping between 0 and array limit.

Parameters

in	<i>index</i>	Index in the trend log array.
----	--------------	-------------------------------

Returns

Index for the next entry.

Definition at line 438 of file user.c.

5.24.2.12 `static uint32_t get_trendlog_index_prev (uint32_t index) [static]`

Getting the previous index in the trend log array. Looping between 0 and array limit.

Parameters

in	<i>index</i>	Index in the trend log array.
----	--------------	-------------------------------

Returns

Index for the previous entry.

Definition at line 453 of file user.c.

5.24.2.13 static uint32_t get_trendlog_index_start (void) [static]

Getting the start index for the trend log array. Is 0 for a not completed array. After completion the start index is running between 0 and array limit.

Returns

Index for the first entry.

Definition at line 467 of file user.c.

5.24.2.14 static void init_scheduler (void) [static]

Initialize the scheduler with user defined settings.

Definition at line 355 of file user.c.

5.24.2.15 static void init_trendlog (void) [static]

Initialize the structure for the trend log data.

Definition at line 400 of file user.c.

5.24.2.16 static bool is_flash_magic_activated (void) [static]

Check if the flash magic token is set or not.

Returns

Returns true if the flash magic token is set.

Definition at line 836 of file user.c.

5.24.2.17 static bool is_property_persistent (const BACNET_SERVER_PROPERTY_INSTANCE * p_prop) [static]

User decision if the given property should be stored or not.

Parameters

in	<i>p_prop</i>	Pointer to property.
----	---------------	----------------------

Returns

Returns true if the property should be stored.

Definition at line 714 of file user.c.

5.24.2.18 static bool is_trendlog_index_valid (uint32_t *index*) [static]

Check if the entry with the given index is valid or not. True for all entries up to the array limit except the array is still not completed.

Parameters

in	<i>index</i>	Index in the trend log array.
----	--------------	-------------------------------

Returns

The trend log array item can be used or not.

Definition at line 415 of file user.c.

5.24.2.19 static bool is_trendlog_time_smaller (BACNET_DATE_TIME *record_time*, BACNET_DATE_TIME *end_time*, bool *or_equal*) [static]

Compare 2 BACNET_DATE_TIME's.

Parameters

in	<i>record_time</i>	Time of a record.
in	<i>end_time</i>	Time to compare with.
in	<i>or_equal</i>	Equality is allowed or not.

Returns

Returns true for record times which are before the end_time and if the equal flag is set true will be returned if the times are equal.

Definition at line 482 of file user.c.

5.24.2.20 static void set_flash_magic (void) [static]

Set the flash magic token.

Definition at line 855 of file user.c.

5.24.2.21 static bool show_led (uint32_t *index*, bool *on_off*) [static]

Setting the state of a LED.

Parameters

in	<i>index</i>	Index of the LED.
in	<i>on_off</i>	State of the LED (on or off).

Returns

Successfully changed state of the LED.

Definition at line 164 of file user.c.

5.24.2.22 void user_loop (ULONG entry_input)

Checking the state of the input devices (Switches, potentiometers and temperature sensors).

Note

This function must be called by the main loop.

Sleep for 1/100 second

Definition at line 980 of file user.c.

5.24.2.23 static void user_reset_values (VOID) [static]

Set the user variables to initial state.

Definition at line 665 of file user.c.

5.24.2.24 bool UserInit (void)

Initialize the user interface.

Set the initial values for: binary input objects (Switches) analog input objects (Potentiometers and temperature sensors) schedule object (Every full hour from Monday to Sunday) trend log object (Internal array to store the data given and requested by the BACnet API)

Return values

<i>true</i>	Always true for this demo application.
-------------	--

Definition at line 1064 of file user.c.

5.24.2.25 void UserReset (void)

Reset the user interface.

Set the user variables and device state to initial state

Definition at line 1097 of file user.c.

5.24.2.26 `bool UserSetLed (uint32_t number, bool on_off)`

Setting the state of the LED for the binary output objects present value.

Parameters

in	<i>number</i>	Number of the LED (ID of binary output object).
in	<i>on_off</i>	New State of the LED (Present value of binary output object).

Returns

Successfully changed state of the LED.

Definition at line 1112 of file user.c.

5.24.2.27 `bool UserStorageDelete (const BACNET_SRVR_INIT * p_server)`

Request for deleting the persistent stored data.

Parameters

in	<i>p_server</i>	Pointer to the server structure of the demo application.
----	-----------------	--

Returns

Successful deleting of the stored data or not.

Definition at line 1789 of file user.c.

5.24.2.28 `bool UserStorageIsPropertyPersistant (BACNET_INST_NUMBER device_id, BACNET_OBJECT_TYPE obj_type, BACNET_INST_NUMBER inst_number, BACNET_PROPERTY_ID property_id)`

Request if the given property should the stored persistent.

Parameters

in	<i>device_id</i>	Device number of the property.
in	<i>obj_type</i>	Object type of the property.
in	<i>inst_number</i>	Object instance of the property.
in	<i>property_id</i>	Property id of the property.

Returns

User decision if the requested should be stored or not.

Definition at line 1543 of file user.c.

5.24.2.29 bool UserStorageRead (const BACNET_SRVR_INIT * *p_server*)

Function which reads out the data stored in the flash pages and copies them into the object property structures.

Parameters

in	<i>p_server</i>	Pointer to the server structure of the demo application.
----	-----------------	--

Returns

Successful reading of the stored data or not.

Definition at line 1572 of file user.c.

5.24.2.30 bool UserStorageWrite (const BACNET_SRVR_INIT * *p_server*)

Function which writes changed persistent property data.

Parameters

in	<i>p_server</i>	Pointer to the server structure of the demo application.
----	-----------------	--

Returns

Successful writing of the stored data or not.

Definition at line 1702 of file user.c.

5.24.2.31 bool UserTrendlogClearBuffer (uint32_t *number*)

Clears the internal memory to store the trend log records.

Parameters

in	<i>number</i>	Number of the trend log object.
----	---------------	---------------------------------

Returns

For this demo application only true for a trend log object with number 1.

Definition at line 1127 of file user.c.

5.24.2.32 `bool UserTrendlogGetRecords (uint32_t number, BACNET_LOG_RECORD p_records[], uint32_t * p_count, bool * p_first, bool * p_last, bool * p_more)`

Getting all stored records for a trend log object.

Parameters

in	<i>number</i>	Number of the trend log object.
out	<i>p_records</i>	Pointer to an array to store the record data.
out	<i>p_count</i>	Number of returned records.

Returns

For this demo application only true for a trend log object with number 1 and a valid index.

Definition at line 1180 of file user.c.

5.24.2.33 `bool UserTrendlogGetRecordsByPosition (uint32_t number, uint32_t position, int32_t count, BACNET_LOG_RECORD p_records[], uint32_t * p_count, bool * p_first, bool * p_last, bool * p_more)`

Getting stored records for a trend log object. Beginning with the given record position and limited by the given count.

Parameters

in	<i>number</i>	Number of the trend log object.
in	<i>position</i>	Start position in the trend log array.
in	<i>count</i>	Maximum count of records to be returned.
out	<i>p_records</i>	Pointer to an array to store the record data.
out	<i>p_count</i>	Number of returned records.

Returns

For this demo application only true for a trend log object with number 1 and a valid index.

Definition at line 1220 of file user.c.

5.24.2.34 `bool UserTrendlogGetRecordsByTime (uint32_t number, BACNET_DATE_TIME start_time, int32_t count, BACNET_LOG_RECORD p_records[], uint32_t * p_count, bool * p_first, bool * p_last, bool * p_more)`

Getting stored records for a trend log object. Beginning with the record with a time stamp greater or equal to the given start_time and limited by the given count.

Parameters

in	<i>number</i>	Number of the trend log object.
in	<i>start_time</i>	Minimum time stamp for a record time stamp.
in	<i>count</i>	Maximum count of records to be returned.
out	<i>p_records</i>	Pointer to an array to store the record data.
out	<i>p_count</i>	Number of returned records.

Returns

For this demo application only true for a trend log object with number 1 and a valid index.

Definition at line 1390 of file user.c.

5.24.2.35 `bool UserTrendlogGetRecordsByTimeRange (uint32_t number, BACNET_DATE_TIME start_time, BACNET_DATE_TIME end_time, BACNET_LOG_RECORD p_records[], uint32_t * p_count, bool * p_first, bool * p_last, bool * p_more)`

Getting stored records for a trend log object. Beginning with the record with a time stamp greater or equal to the given start_time and limited by the given end_time.

Parameters

in	<i>number</i>	Number of the trend log object.
in	<i>start_time</i>	Minimum time stamp for a record time stamp.
in	<i>end_time</i>	Maximum time stamp for a record time stamp.
out	<i>p_records</i>	Pointer to an array to store the record data.
out	<i>p_count</i>	Number of returned records.

Returns

For this demo application only true for a trend log object with number 1 and a valid index.

Definition at line 1477 of file user.c.

5.24.2.36 `bool UserTrendlogPutRecord (uint32_t number, BACNET_LOG_RECORD * p_record)`

Stores a record in the internal memory for a trend log object.

Parameters

in	<i>number</i>	Number of the trend log object.
in	<i>p_record</i>	Pointer to the record data.

Returns

For this demo application only true for a trend log object with number 1.

Definition at line 1148 of file user.c.

5.24.2.37 `bool UserUserTrendlogGetRecordsBySequence (uint32_t number, uint32_t sequence, int32_t count, BACNET_LOG_RECORD p_records[], uint32_t * p_count, bool * p_first, bool * p_last, bool * p_more)`

Getting stored records for a trend log object. Beginning with the record with the given sequence number and limited by the given count.

Parameters

in	<i>number</i>	Number of the trend log object.
in	<i>sequence</i>	Sequence number to start with.
in	<i>count</i>	Maximum count of records to be returned.
out	<i>p_records</i>	Pointer to an array to store the record data.
out	<i>p_count</i>	Number of returned records.

Returns

For this demo application only true for a trend log object with number 1 and a valid index.

Definition at line 1306 of file user.c.

5.24.3 Variable Documentation

5.24.3.1 `BAC_BYTE g_flash_checksum` [static]

Definition at line 147 of file user.c.

5.24.3.2 `BAC_BYTE g_flash_magic[FLASH_MEM_MAGIC_SIZE] = {0xEF, 0xBE, 0xC0, 0xBA}` [static]

Definition at line 145 of file user.c.

5.24.3.3 `BAC_BYTE g_flash_mem_page[FLASH_DEVICE_PAGE_SIZE]` [static]

Definition at line 143 of file user.c.

5.24.3.4 `BAC_WORD g_flash_offset` [static]

Definition at line 146 of file user.c.

5.24.3.5 `BAC_UINT g_obj_sc1_daily_schedule[$\text{CALC_WRITABLE_DAILY_SCHEDULE_ARRAY_SIZE}(\text{OBJ_SC1_DAILY_SCHEDULE_SIZE})$]`

5.24.3.6 `BACNET_DEV_OBJ_PROP_REFERENCE g_obj_sc1_obj_prop_ref_list[]`

5.24.3.7 `BAC_UINT g_obj_sc1_obj_prop_ref_listCount`

5.24.3.8 `BAC_BOOLEAN g_page_any_page_changed` [static]

Definition at line 151 of file user.c.

5.24.3.9 BAC_BOOLEAN g_page_data_changed [static]

Definition at line 150 of file user.c.

5.24.3.10 BAC_BOOLEAN g_page_is_new [static]

Definition at line 149 of file user.c.

5.24.3.11 BAC_WORD g_page_offset [static]

Definition at line 148 of file user.c.

5.24.3.12 uint32_t g_sequence [static]

Definition at line 137 of file user.c.

5.24.3.13 user_analog_input_t g_user_analog_values[USER_MAX_ANALOG] [static]

Definition at line 134 of file user.c.

5.24.3.14 user_binary_input_t g_user_binary_values[USER_MAX_BINARY] [static]

Definition at line 133 of file user.c.

5.24.3.15 user_button_t g_user_button[2] [static]

Initial value:

```
=  
{  
    {IOPORT_PORT_00_PIN_05, IOPORT_LEVEL_HIGH},  
    {IOPORT_PORT_00_PIN_06, IOPORT_LEVEL_HIGH}  
}
```

Definition at line 115 of file user.c.

5.24.3.16 const ioport_port_pin_t g_user_led_pins[] [static]

Initial value:

```
=  
{  
    IOPORT_PORT_06_PIN_00,  
    IOPORT_PORT_06_PIN_01,  
}
```

Definition at line 121 of file user.c.

5.24.3.17 `const bsp_leds_t g_user_leds` [static]

Initial value:

```
=  
{  
    .led_count = 2,  
    .p_leds    = g_user_led_pins  
}
```

Definition at line 127 of file user.c.

5.24.3.18 `TX_MUTEX g_user_mutex` [static]

Definition at line 141 of file user.c.

5.24.3.19 `TX_THREAD g_user_thread` [static]

Definition at line 139 of file user.c.

5.24.3.20 `uint8_t g_user_thread_stack[0x1800]` [static]

Definition at line 140 of file user.c.

5.24.3.21 `user_trendlog_t g_user_trendlog` [static]

Definition at line 136 of file user.c.

5.24.3.22 `BAC_BYTE* gp_flash_mem = (BAC_BYTE *)FLASH_DEVICE_START_ADDRESS` [static]

Definition at line 144 of file user.c.

5.25 `src/demo/user.h` File Reference

```
#include <stdint.h>  
#include <stdbool.h>  
#include "bacnet.h"
```

Data Structures

- struct [user_record_t](#)

Functions

- bool [UserInit](#) (void)
Initialize the user interface.
- void [UserReset](#) (void)
Reset the user interface.
- bool [UserSetLed](#) (uint32_t number, bool on_off)
Setting the state of the LED for the binary output objects present value.
- bool [UserTrendlogClearBuffer](#) (uint32_t number)
Clears the internal memory to store the trend log records.
- bool [UserTrendlogPutRecord](#) (uint32_t number, BACNET_LOG_RECORD *p_record)
Stores a record in the internal memory for a trend log object.
- bool [UserTrendlogGetRecords](#) (uint32_t number, BACNET_LOG_RECORD p_records[], uint32_t *p_count, bool *p_first, bool *p_last, bool *p_more)
Getting all stored records for a trend log object.
- bool [UserTrendlogGetRecordsByPosition](#) (uint32_t number, uint32_t position, int32_t count, BACNET_LOG_RECORD p_records[], uint32_t *p_count, bool *p_first, bool *p_last, bool *p_more)
Getting stored records for a trend log object. Beginning with the given record position and limited by the given count.
- bool [UserTrendlogGetRecordsBySequence](#) (uint32_t number, uint32_t sequence, int32_t count, BACNET_LOG_RECORD p_records[], uint32_t *p_count, bool *p_first, bool *p_last, bool *p_more)
Getting stored records for a trend log object. Beginning with the record with the given sequence number and limited by the given count.
- bool [UserTrendlogGetRecordsByTime](#) (uint32_t number, BACNET_DATE_TIME start_time, int32_t count, BACNET_LOG_RECORD p_records[], uint32_t *p_count, bool *p_first, bool *p_last, bool *p_more)
Getting stored records for a trend log object. Beginning with the record with a time stamp greater or equal to the given start_time and limited by the given count.
- bool [UserTrendlogGetRecordsByTimeRange](#) (uint32_t number, BACNET_DATE_TIME start_time, BACNET_DATE_TIME end_time, BACNET_LOG_RECORD p_records[], uint32_t *p_count, bool *p_first, bool *p_last, bool *p_more)
Getting stored records for a trend log object. Beginning with the record with a time stamp greater or equal to the given start_time and limited by the given end_time.
- bool [UserStorageIsPropertyPersistent](#) (BACNET_INST_NUMBER device_id, BACNET_OBJECT_TYPE obj_type, BACNET_INST_NUMBER inst_number, BACNET_PROPERTY_ID property_id)
Request if the given property should be stored persistent.
- bool [UserStorageRead](#) (const BACNET_SRVR_INIT *p_server)
Function which reads out the data stored in the flash pages and copies them into the object property structures.
- bool [UserStorageWrite](#) (const BACNET_SRVR_INIT *p_server)
Function which writes changed persistent property data.
- bool [UserStorageDelete](#) (const BACNET_SRVR_INIT *p_server)
Request for deleting the persistent stored data.

5.25.1 Function Documentation

5.25.1.1 bool UserInit (void)

Initialize the user interface.

Set the initial values for: binary input objects (Switches) analog input objects (Potentiometers and temperature sensors) schedule object (Every full hour from Monday to Sunday) trend log object (Internal array to store the data given and requested by the BACnet API)

Return values

<i>true</i>	Always true for this demo application.
-------------	--

Definition at line 1064 of file user.c.

5.25.1.2 void UserReset (void)

Reset the user interface.

Set the user variables and device state to initial state

Definition at line 1097 of file user.c.

5.25.1.3 bool UserSetLed (uint32_t number, bool on_off)

Setting the state of the LED for the binary output objects present value.

Parameters

in	<i>number</i>	Number of the LED (ID of binary output object).
in	<i>on_off</i>	New State of the LED (Present value of binary output object).

Returns

Successfully changed state of the LED.

Definition at line 1112 of file user.c.

5.25.1.4 bool UserStorageDelete (const BACNET_SRVR_INIT * p_server)

Request for deleting the persistent stored data.

Parameters

in	<i>p_server</i>	Pointer to the server structure of the demo application.
----	-----------------	--

Returns

Successful deleting of the stored data or not.

Definition at line 1789 of file user.c.

5.25.1.5 bool UserStorageIsPropertyPersistent (BACNET_INST_NUMBER device_id, BACNET_OBJECT_TYPE obj_type, BACNET_INST_NUMBER inst_number, BACNET_PROPERTY_ID property_id)

Request if the given property should the stored persistent.

Parameters

in	<i>device_id</i>	Device number of the property.
in	<i>obj_type</i>	Object type of the property.
in	<i>inst_number</i>	Object instance of the property.
in	<i>property_id</i>	Property id of the property.

Returns

User decision if the requested should be stored or not.

Definition at line 1543 of file user.c.

5.25.1.6 bool UserStorageRead (const BACNET_SRVR_INIT * *p_server*)

Function which reads out the data stored in the flash pages and copies them into the object property structures.

Parameters

in	<i>p_server</i>	Pointer to the server structure of the demo application.
----	-----------------	--

Returns

Successful reading of the stored data or not.

Definition at line 1572 of file user.c.

5.25.1.7 bool UserStorageWrite (const BACNET_SRVR_INIT * *p_server*)

Function which writes changed persistent property data.

Parameters

in	<i>p_server</i>	Pointer to the server structure of the demo application.
----	-----------------	--

Returns

Successful writing of the stored data or not.

Definition at line 1702 of file user.c.

5.25.1.8 bool UserTrendlogClearBuffer (uint32_t *number*)

Clears the internal memory to store the trend log records.

Parameters

in	<i>number</i>	Number of the trend log object.
----	---------------	---------------------------------

Returns

For this demo application only true for a trend log object with number 1.

Definition at line 1127 of file user.c.

5.25.1.9 `bool UserTrendlogGetRecords (uint32_t number, BACNET_LOG_RECORD p_records[], uint32_t * p_count, bool * p_first, bool * p_last, bool * p_more)`

Getting all stored records for a trend log object.

Parameters

in	<i>number</i>	Number of the trend log object.
out	<i>p_records</i>	Pointer to an array to store the record data.
out	<i>p_count</i>	Number of returned records.

Returns

For this demo application only true for a trend log object with number 1 and a valid index.

Definition at line 1180 of file user.c.

5.25.1.10 `bool UserTrendlogGetRecordsByPosition (uint32_t number, uint32_t position, int32_t count, BACNET_LOG_RECORD p_records[], uint32_t * p_count, bool * p_first, bool * p_last, bool * p_more)`

Getting stored records for a trend log object. Beginning with the given record position and limited by the given count.

Parameters

in	<i>number</i>	Number of the trend log object.
in	<i>position</i>	Start position in the trend log array.
in	<i>count</i>	Maximum count of records to be returned.
out	<i>p_records</i>	Pointer to an array to store the record data.
out	<i>p_count</i>	Number of returned records.

Returns

For this demo application only true for a trend log object with number 1 and a valid index.

Definition at line 1220 of file user.c.

5.25.1.11 `bool UserTrendlogGetRecordsByTime (uint32_t number, BACNET_DATE_TIME start_time, int32_t count, BACNET_LOG_RECORD p_records[], uint32_t * p_count, bool * p_first, bool * p_last, bool * p_more)`

Getting stored records for a trend log object. Beginning with the record with a time stamp greater or equal to the given `start_time` and limited by the given count.

Parameters

in	<i>number</i>	Number of the trend log object.
in	<i>start_time</i>	Minimum time stamp for a record time stamp.
in	<i>count</i>	Maximum count of records to be returned.
out	<i>p_records</i>	Pointer to an array to store the record data.
out	<i>p_count</i>	Number of returned records.

Returns

For this demo application only true for a trend log object with number 1 and a valid index.

Definition at line 1390 of file user.c.

5.25.1.12 `bool UserTrendlogGetRecordsByTimeRange (uint32_t number, BACNET_DATE_TIME start_time, BACNET_DATE_TIME end_time, BACNET_LOG_RECORD p_records[], uint32_t * p_count, bool * p_first, bool * p_last, bool * p_more)`

Getting stored records for a trend log object. Beginning with the record with a time stamp greater or equal to the given `start_time` and limited by the given `end_time`.

Parameters

in	<i>number</i>	Number of the trend log object.
in	<i>start_time</i>	Minimum time stamp for a record time stamp.
in	<i>end_time</i>	Maximum time stamp for a record time stamp.
out	<i>p_records</i>	Pointer to an array to store the record data.
out	<i>p_count</i>	Number of returned records.

Returns

For this demo application only true for a trend log object with number 1 and a valid index.

Definition at line 1477 of file user.c.

5.25.1.13 `bool UserTrendlogPutRecord (uint32_t number, BACNET_LOG_RECORD * p_record)`

Stores a record in the internal memory for a trend log object.

Parameters

in	<i>number</i>	Number of the trend log object.
in	<i>p_record</i>	Pointer to the record data.

Returns

For this demo application only true for a trend log object with number 1.

Definition at line 1148 of file user.c.

5.25.1.14 `bool UserUserTrendlogGetRecordsBySequence (uint32_t number, uint32_t sequence, int32_t count, BACNET_LOG_RECORD p_records[], uint32_t * p_count, bool * p_first, bool * p_last, bool * p_more)`

Getting stored records for a trend log object. Beginning with the record with the given sequence number and limited by the given count.

Parameters

in	<i>number</i>	Number of the trend log object.
in	<i>sequence</i>	Sequence number to start with.
in	<i>count</i>	Maximum count of records to be returned.
out	<i>p_records</i>	Pointer to an array to store the record data.
out	<i>p_count</i>	Number of returned records.

Returns

For this demo application only true for a trend log object with number 1 and a valid index.

Definition at line 1306 of file user.c.

Index

- ADC_REGISTER_COUNT
 - adc.c, 11
- adc.c
 - ADC_REGISTER_COUNT, 11
 - AdcCompleteCallback, 12
 - AdcGetValue, 12
 - AdcInit, 12
 - g_adc, 12
 - g_adc_data, 12
 - g_adc_register, 12
 - g_adc_started, 13
- adc.h
 - AdcCompleteCallback, 13
 - AdcGetValue, 13
 - AdcInit, 14
- adc_sum
 - user_analog_input_t, 7
- adc_sum_count
 - user_analog_input_t, 7
- adc_value
 - user_analog_input_t, 7
- AdcCompleteCallback
 - adc.c, 12
 - adc.h, 13
- AdcGetValue
 - adc.c, 12
 - adc.h, 13
- AdcInit
 - adc.c, 12
 - adc.h, 14
- analog_value_changed
 - user.c, 57
- BACNET_DEMO_WITH_FOREIGN_DEVICE
 - bacnetdemo.c, 15
- bacnet_init
 - bacnetdemo.c, 15
- bacnet_reset
 - bacnetdemo.c, 15
- bacnet_storage_delete
 - bacnetdemo.c, 16
- bacnet_storage_read
 - bacnetdemo.c, 16
- bacnet_storage_write
 - bacnetdemo.c, 16
- bacnet_trend
 - bacnetdemo.c, 16
- bacnet_value
 - user_analog_input_t, 7
- bacnet_write
 - bacnetdemo.c, 17
- BacnetDemoMain
 - bacnetdemo.c, 17
 - bacnetdemo.h, 20
- bacnetdemo.c
 - BACNET_DEMO_WITH_FOREIGN_DEVICE, 15
 - bacnet_init, 15
 - bacnet_reset, 15
 - bacnet_storage_delete, 16
 - bacnet_storage_read, 16
 - bacnet_storage_write, 16
 - bacnet_trend, 16
 - bacnet_write, 17
 - BacnetDemoMain, 17
 - g_bacnet_common_cfg, 18
 - g_bacnet_common_ctrl, 18
 - g_bacnet_common_instance, 18
 - g_bacnet_data_link_cfg, 18
 - g_bacnet_data_link_ctrl, 18
 - g_bacnet_data_link_ip_ext_cfg, 19
 - g_bacnet_instance, 19
 - g_log_record, 19
- bacnetdemo.h
 - BacnetDemoMain, 20
 - DEMO_OBJ_ID_AI1, 20
 - DEMO_OBJ_ID_AI2, 20
 - DEMO_OBJ_ID_AI3, 20
 - DEMO_OBJ_ID_AI4, 20
 - DEMO_OBJ_ID_AO1, 20
 - DEMO_OBJ_ID_BI1, 20
 - DEMO_OBJ_ID_BI2, 20
 - DEMO_OBJ_ID_BO1, 20
 - DEMO_OBJ_ID_BO2, 20
 - DEMO_OBJ_ID_CA1, 20
 - DEMO_OBJ_ID_DEV, 20
 - DEMO_OBJ_ID_MAX, 20
 - DEMO_OBJ_ID_NC1, 20
 - DEMO_OBJ_ID_SC1, 20
 - DEMO_OBJ_ID_TR1, 20
 - demo_obj_ids_t, 20
- button
 - user_binary_input_t, 8
- button_pressed
 - user.c, 57
- CFG_READONLY_STRING
 - definitions.h, 24
- CFG_SERVER_INIT
 - definitions.h, 24
- CFG_WRITABLE_ADDRESS

- definitions.h, 24
- CFG_WRITABLE_APDU_PROPERTIES
 - definitions.h, 24
- calc_analog_value
 - user.c, 57
- copy_trendlog_data
 - user.c, 58
- count
 - user_trendlog_t, 10
- DEMO_OBJ_ID_AI1
 - bacnetdemo.h, 20
- DEMO_OBJ_ID_AI2
 - bacnetdemo.h, 20
- DEMO_OBJ_ID_AI3
 - bacnetdemo.h, 20
- DEMO_OBJ_ID_AI4
 - bacnetdemo.h, 20
- DEMO_OBJ_ID_AO1
 - bacnetdemo.h, 20
- DEMO_OBJ_ID_BI1
 - bacnetdemo.h, 20
- DEMO_OBJ_ID_BI2
 - bacnetdemo.h, 20
- DEMO_OBJ_ID_BO1
 - bacnetdemo.h, 20
- DEMO_OBJ_ID_BO2
 - bacnetdemo.h, 20
- DEMO_OBJ_ID_CA1
 - bacnetdemo.h, 20
- DEMO_OBJ_ID_DEV
 - bacnetdemo.h, 20
- DEMO_OBJ_ID_MAX
 - bacnetdemo.h, 20
- DEMO_OBJ_ID_NC1
 - bacnetdemo.h, 20
- DEMO_OBJ_ID_SC1
 - bacnetdemo.h, 20
- DEMO_OBJ_ID_TR1
 - bacnetdemo.h, 20
- DEV_DATALINK_COUNT
 - definitions.h, 22
- DEV_DATALINK_ETH_IP1
 - definitions.h, 22
- DEV_DATALINK_ETH_IP2
 - definitions.h, 22
- DEV_DATALINK_ETH_IP3
 - definitions.h, 22
- DEV_DATALINK_ETH_IP4
 - definitions.h, 22
- DEV_DATALINK_ETH_NET_ID
 - definitions.h, 22
- DEV_DATALINK_ETH_PORT_ID
 - definitions.h, 22
- DEV_DATALINK_ETH_PORT
 - definitions.h, 22
- DEV_DATALINK_FOREIGN_INTERVAL
 - definitions.h, 22
- DEV_DATALINK_FOREIGN_IP1
 - definitions.h, 22
- DEV_DATALINK_FOREIGN_IP2
 - definitions.h, 22
- DEV_DATALINK_FOREIGN_IP3
 - definitions.h, 23
- DEV_DATALINK_FOREIGN_IP4
 - definitions.h, 23
- DEV_DATALINK_FOREIGN_PORT
 - definitions.h, 23
- DEV_DATALINK_MSTP_ADDRESS
 - definitions.h, 23
- DEV_DATALINK_MSTP_NET
 - definitions.h, 23
- DEV_DATALINK_MSTP_PORT
 - definitions.h, 23
- DEVICE_DDC_PASSWORD_INIT_STRING
 - definitions.h, 23
- DEVICE_REINIT_PASSWORD_INIT_STRING
 - definitions.h, 23
- data
 - user_record_t, 9
- definitions.h
 - CFG_READONLY_STRING, 24
 - CFG_SERVER_INIT, 24
 - CFG_WRITABLE_ADDRESS, 24
 - CFG_WRITABLE_APDU_PROPERTIES, 24
 - DEV_DATALINK_COUNT, 22
 - DEV_DATALINK_ETH_IP1, 22
 - DEV_DATALINK_ETH_IP2, 22
 - DEV_DATALINK_ETH_IP3, 22
 - DEV_DATALINK_ETH_IP4, 22
 - DEV_DATALINK_ETH_NET_ID, 22
 - DEV_DATALINK_ETH_PORT_ID, 22
 - DEV_DATALINK_ETH_PORT, 22
 - DEV_DATALINK_FOREIGN_INTERVAL, 22
 - DEV_DATALINK_FOREIGN_IP1, 22
 - DEV_DATALINK_FOREIGN_IP2, 22
 - DEV_DATALINK_FOREIGN_IP3, 23
 - DEV_DATALINK_FOREIGN_IP4, 23
 - DEV_DATALINK_FOREIGN_PORT, 23
 - DEV_DATALINK_MSTP_ADDRESS, 23
 - DEV_DATALINK_MSTP_NET, 23
 - DEV_DATALINK_MSTP_PORT, 23
 - DEVICE_DDC_PASSWORD_INIT_STRING, 23
 - DEVICE_REINIT_PASSWORD_INIT_STRING, 23
 - g_device_data_links, 24
 - g_obj_array, 24
 - g_server_device, 24
 - g_server_device_dcc_value, 25
- demo_obj_ids_t
 - bacnetdemo.h, 20
- FLASH_DEVICE_FLASH_SIZE
 - user.c, 56
- FLASH_DEVICE_PAGE_SIZE
 - user.c, 56
- FLASH_DEVICE_START_ADDRESS
 - user.c, 56
- FLASH_MEM_MAGIC_SIZE

- user.c, [56](#)
- flash_wait
 - user.c, [58](#)
- flash_write
 - user.c, [58](#)
- flash_write_page
 - user.c, [59](#)
- g_adc
 - adc.c, [12](#)
- g_adc_data
 - adc.c, [12](#)
- g_adc_register
 - adc.c, [12](#)
- g_adc_started
 - adc.c, [13](#)
- g_bacnet_common_cfg
 - bacnetdemo.c, [18](#)
- g_bacnet_common_ctrl
 - bacnetdemo.c, [18](#)
- g_bacnet_common_instance
 - bacnetdemo.c, [18](#)
- g_bacnet_data_link_cfg
 - bacnetdemo.c, [18](#)
- g_bacnet_data_link_ctrl
 - bacnetdemo.c, [18](#)
- g_bacnet_data_link_ip_ext_cfg
 - bacnetdemo.c, [19](#)
- g_bacnet_instance
 - bacnetdemo.c, [19](#)
- g_device_data_links
 - definitions.h, [24](#)
- g_flash_checksum
 - user.c, [68](#)
- g_flash_magic
 - user.c, [68](#)
- g_flash_mem_page
 - user.c, [68](#)
- g_flash_offset
 - user.c, [68](#)
- g_log_record
 - bacnetdemo.c, [19](#)
- g_obj_ai1_description
 - obj_ai.c, [26](#)
- g_obj_ai1_evt_msg_txt
 - obj_ai.c, [26](#)
- g_obj_ai1_out_of_service
 - obj_ai.c, [26](#)
- g_obj_ai1_present_value
 - obj_ai.c, [27](#)
- g_obj_ai1_state_flags
 - obj_ai.c, [27](#)
- g_obj_ai2_description
 - obj_ai.c, [27](#)
- g_obj_ai2_evt_msg_txt
 - obj_ai.c, [27](#)
- g_obj_ai2_out_of_service
 - obj_ai.c, [27](#)
- g_obj_ai2_present_value
 - obj_ai.c, [27](#)
- g_obj_ai3_description
 - obj_ai.c, [27](#)
- g_obj_ai3_evt_msg_txt
 - obj_ai.c, [27](#)
- g_obj_ai3_out_of_service
 - obj_ai.c, [27](#)
- g_obj_ai3_present_value
 - obj_ai.c, [27](#)
- g_obj_ai3_state_flags
 - obj_ai.c, [27](#)
- g_obj_ai4_description
 - obj_ai.c, [27](#)
- g_obj_ai4_evt_msg_txt
 - obj_ai.c, [27](#)
- g_obj_ai4_out_of_service
 - obj_ai.c, [27](#)
- g_obj_ai4_present_value
 - obj_ai.c, [27](#)
- g_obj_ai4_state_flags
 - obj_ai.c, [27](#)
- g_obj_ao1_description
 - obj_ao.c, [33](#)
- g_obj_ao1_evt_msg_txt
 - obj_ao.c, [34](#)
- g_obj_ao1_out_of_service
 - obj_ao.c, [34](#)
- g_obj_ao1_present_value
 - obj_ao.c, [34](#)
- g_obj_ao1_state_flags
 - obj_ao.c, [34](#)
- g_obj_array
 - definitions.h, [24](#)
- g_obj_bi1_description
 - obj_bi.c, [37](#)
- g_obj_bi1_evt_msg_txt
 - obj_bi.c, [37](#)
- g_obj_bi1_out_of_service
 - obj_bi.c, [37](#)
- g_obj_bi1_present_value
 - obj_bi.c, [37](#)
- g_obj_bi1_state_flags
 - obj_bi.c, [37](#)
- g_obj_bi2_description
 - obj_bi.c, [37](#)
- g_obj_bi2_evt_msg_txt
 - obj_bi.c, [37](#)
- g_obj_bi2_out_of_service
 - obj_bi.c, [37](#)
- g_obj_bi2_present_value
 - obj_bi.c, [37](#)
- g_obj_bi2_state_flags
 - obj_bi.c, [37](#)
- g_obj_bo1_description
 - obj_bo.c, [40](#)
- g_obj_bo1_out_of_service

- obj_bo.c, 40
- g_obj_bo1_present_value
 - obj_bo.c, 40
- g_obj_bo1_state_flags
 - obj_bo.c, 40
- g_obj_bo2_description
 - obj_bo.c, 40
- g_obj_bo2_out_of_service
 - obj_bo.c, 40
- g_obj_bo2_present_value
 - obj_bo.c, 40
- g_obj_bo2_state_flags
 - obj_bo.c, 40
- g_obj_ca1_description
 - obj_ca.c, 43
- g_obj_dev_description
 - obj_dev.c, 45
- g_obj_dev_id
 - obj_ai.c, 27
 - obj_bi.c, 37
 - obj_dev.c, 45
- g_obj_dev_location
 - obj_dev.c, 45
- g_obj_dev_name
 - obj_dev.c, 45
- g_obj_dev_time_of_restart
 - obj_dev.c, 45
- g_obj_nc1_description
 - obj_nc.c, 47
- g_obj_sc1_daily_schedule
 - user.c, 68
- g_obj_sc1_description
 - obj_sc.c, 49
- g_obj_sc1_obj_prop_ref_list
 - user.c, 68
- g_obj_sc1_obj_prop_ref_listCount
 - user.c, 68
- g_obj_sc1_out_of_service
 - obj_sc.c, 49
- g_obj_sc1_state_flags
 - obj_sc.c, 49
- g_obj_tr1_description
 - obj_tr.c, 51
- g_obj_tr1_evt_msg_txt
 - obj_tr.c, 51
- g_page_any_page_changed
 - user.c, 68
- g_page_data_changed
 - user.c, 68
- g_page_is_new
 - user.c, 69
- g_page_offset
 - user.c, 69
- g_sequence
 - user.c, 69
- g_server_device
 - definitions.h, 24
- g_server_device_dcc_value
 - definitions.h, 25
- g_user_analog_values
 - user.c, 69
- g_user_binary_values
 - user.c, 69
- g_user_button
 - user.c, 69
- g_user_led_pins
 - user.c, 69
- g_user_leds
 - user.c, 69
- g_user_mutex
 - user.c, 70
- g_user_thread
 - user.c, 70
- g_user_thread_stack
 - user.c, 70
- g_user_trendlog
 - user.c, 70
- get_crc_checksum
 - user.c, 59
- get_switch_state
 - user.c, 59
- get_trendlog_first_last
 - user.c, 60
- get_trendlog_index_next
 - user.c, 60
- get_trendlog_index_prev
 - user.c, 60
- get_trendlog_index_start
 - user.c, 61
- gp_flash_mem
 - user.c, 70
- init_scheduler
 - user.c, 61
- init_trendlog
 - user.c, 61
- is_flash_magic_activated
 - user.c, 61
- is_property_persistent
 - user.c, 61
- is_trendlog_index_valid
 - user.c, 62
- is_trendlog_time_smaller
 - user.c, 62
- level
 - user_button_t, 9
- OBJ_AI1_COV_INCREMENT_INITIAL
 - obj_ai.h, 28
- OBJ_AI1_DESCRIPTION_INIT_STRING
 - obj_ai.h, 28
- OBJ_AI1_LIMIT_DEADBAND_INITIAL
 - obj_ai.h, 29
- OBJ_AI1_LIMIT_HIGH_INITIAL
 - obj_ai.h, 29
- OBJ_AI1_LIMIT_LOW_INITIAL

obj_ai.h, 29
OBJ_AI1_NAME_INIT_STRING
obj_ai.h, 29
OBJ_AI1_PRESENT_VALUE_INITIAL
obj_ai.h, 29
OBJ_AI1_UNIT_INITIAL
obj_ai.h, 29
OBJ_AI2_COV_INCREMENT_INITIAL
obj_ai.h, 29
OBJ_AI2_DESCRIPTION_INIT_STRING
obj_ai.h, 29
OBJ_AI2_LIMIT_DEADBAND_INITIAL
obj_ai.h, 29
OBJ_AI2_LIMIT_HIGH_INITIAL
obj_ai.h, 29
OBJ_AI2_LIMIT_LOW_INITIAL
obj_ai.h, 30
OBJ_AI2_NAME_INIT_STRING
obj_ai.h, 30
OBJ_AI2_PRESENT_VALUE_INITIAL
obj_ai.h, 30
OBJ_AI2_UNIT_INITIAL
obj_ai.h, 30
OBJ_AI3_COV_INCREMENT_INITIAL
obj_ai.h, 30
OBJ_AI3_DESCRIPTION_INIT_STRING
obj_ai.h, 30
OBJ_AI3_LIMIT_DEADBAND_INITIAL
obj_ai.h, 30
OBJ_AI3_LIMIT_HIGH_INITIAL
obj_ai.h, 30
OBJ_AI3_LIMIT_LOW_INITIAL
obj_ai.h, 30
OBJ_AI3_NAME_INIT_STRING
obj_ai.h, 30
OBJ_AI3_PRESENT_VALUE_INITIAL
obj_ai.h, 31
OBJ_AI3_UNIT_INITIAL
obj_ai.h, 31
OBJ_AI4_COV_INCREMENT_INITIAL
obj_ai.h, 31
OBJ_AI4_DESCRIPTION_INIT_STRING
obj_ai.h, 31
OBJ_AI4_LIMIT_DEADBAND_INITIAL
obj_ai.h, 31
OBJ_AI4_LIMIT_HIGH_INITIAL
obj_ai.h, 31
OBJ_AI4_LIMIT_LOW_INITIAL
obj_ai.h, 31
OBJ_AI4_NAME_INIT_STRING
obj_ai.h, 31
OBJ_AI4_PRESENT_VALUE_INITIAL
obj_ai.h, 31
OBJ_AI4_UNIT_INITIAL
obj_ai.h, 31
OBJ_AO1_DESCRIPTION_INIT_STRING
obj_ao.h, 34
OBJ_AO1_NAME_INIT_STRING
obj_ao.h, 34
OBJ_BI1_ACTIVE_INIT_STRING
obj_bi.h, 38
OBJ_BI1_DESCRIPTION_INIT_STRING
obj_bi.h, 38
OBJ_BI1_INACTIVE_INIT_STRING
obj_bi.h, 38
OBJ_BI1_NAME_INIT_STRING
obj_bi.h, 38
OBJ_BI2_ACTIVE_INIT_STRING
obj_bi.h, 38
OBJ_BI2_DESCRIPTION_INIT_STRING
obj_bi.h, 38
OBJ_BI2_INACTIVE_INIT_STRING
obj_bi.h, 38
OBJ_BI2_NAME_INIT_STRING
obj_bi.h, 38
OBJ_BO1_ACTIVE_INIT_STRING
obj_bo.h, 41
OBJ_BO1_DESCRIPTION_INIT_STRING
obj_bo.h, 41
OBJ_BO1_INACTIVE_INIT_STRING
obj_bo.h, 41
OBJ_BO1_NAME_INIT_STRING
obj_bo.h, 41
OBJ_BO2_ACTIVE_INIT_STRING
obj_bo.h, 41
OBJ_BO2_DESCRIPTION_INIT_STRING
obj_bo.h, 41
OBJ_BO2_INACTIVE_INIT_STRING
obj_bo.h, 42
OBJ_BO2_NAME_INIT_STRING
obj_bo.h, 42
OBJ_CA1_DESCRIPTION_INIT_STRING
obj_ca.h, 44
OBJ_CA1_NAME_INIT_STRING
obj_ca.h, 44
OBJ_DEV_DESCRIPTION_INIT_STRING
obj_dev.h, 46
OBJ_DEV_FIRMWARE_REV_STRING
obj_dev.h, 46
OBJ_DEV_INST_NUMBER
obj_dev.h, 46
OBJ_DEV_LOCATION_INIT_STRING
obj_dev.h, 46
OBJ_DEV_MAX_APDU_SIZE
obj_dev.h, 46
OBJ_DEV_MODEL_NAME_STRING
obj_dev.h, 46
OBJ_DEV_NAME_INIT_STRING
obj_dev.h, 46
OBJ_DEV_SOFTWARE_VER_STRING
obj_dev.h, 46
OBJ_DEV_VENDOR_ID_NUMBER
obj_dev.h, 46
OBJ_DEV_VENDOR_NAME_STRING
obj_dev.h, 46
OBJ_NC1_DESCRIPTION_INIT_STRING

- obj_nc.h, 48
- OBJ_NC1_NAME_INIT_STRING
 - obj_nc.h, 48
- OBJ_SC1_DAILY_SCHEDULE_SIZE
 - obj_sc.h, 50
- OBJ_SC1_DESCRIPTION_INIT_STRING
 - obj_sc.h, 50
- OBJ_SC1_NAME_INIT_STRING
 - obj_sc.h, 50
- OBJ_TR1_DESCRIPTION_INIT_STRING
 - obj_tr.h, 52
- OBJ_TR1_LOG_BUFFER_SIZE
 - obj_tr.h, 52
- OBJ_TR1_NAME_INIT_STRING
 - obj_tr.h, 52
- obj_ai.c
 - g_obj_ai1_description, 26
 - g_obj_ai1_evt_msg_txt, 26
 - g_obj_ai1_out_of_service, 26
 - g_obj_ai1_present_value, 27
 - g_obj_ai1_state_flags, 27
 - g_obj_ai2_description, 27
 - g_obj_ai2_evt_msg_txt, 27
 - g_obj_ai2_out_of_service, 27
 - g_obj_ai2_present_value, 27
 - g_obj_ai2_state_flags, 27
 - g_obj_ai3_description, 27
 - g_obj_ai3_evt_msg_txt, 27
 - g_obj_ai3_out_of_service, 27
 - g_obj_ai3_present_value, 27
 - g_obj_ai3_state_flags, 27
 - g_obj_ai4_description, 27
 - g_obj_ai4_evt_msg_txt, 27
 - g_obj_ai4_out_of_service, 27
 - g_obj_ai4_present_value, 27
 - g_obj_ai4_state_flags, 27
 - g_obj_dev_id, 27
 - ObjAllInit, 26
 - ObjAIPresentValueUpdate, 26
 - ObjAIReset, 26
- obj_ai.h
 - OBJ_AI1_COV_INCREMENT_INITIAL, 28
 - OBJ_AI1_DESCRIPTION_INIT_STRING, 28
 - OBJ_AI1_LIMIT_DEADBAND_INITIAL, 29
 - OBJ_AI1_LIMIT_HIGH_INITIAL, 29
 - OBJ_AI1_LIMIT_LOW_INITIAL, 29
 - OBJ_AI1_NAME_INIT_STRING, 29
 - OBJ_AI1_PRESENT_VALUE_INITIAL, 29
 - OBJ_AI1_UNIT_INITIAL, 29
 - OBJ_AI2_COV_INCREMENT_INITIAL, 29
 - OBJ_AI2_DESCRIPTION_INIT_STRING, 29
 - OBJ_AI2_LIMIT_DEADBAND_INITIAL, 29
 - OBJ_AI2_LIMIT_HIGH_INITIAL, 29
 - OBJ_AI2_LIMIT_LOW_INITIAL, 30
 - OBJ_AI2_NAME_INIT_STRING, 30
 - OBJ_AI2_PRESENT_VALUE_INITIAL, 30
 - OBJ_AI2_UNIT_INITIAL, 30
 - OBJ_AI3_COV_INCREMENT_INITIAL, 30
 - OBJ_AI3_DESCRIPTION_INIT_STRING, 30
 - OBJ_AI3_LIMIT_DEADBAND_INITIAL, 30
 - OBJ_AI3_LIMIT_HIGH_INITIAL, 30
 - OBJ_AI3_LIMIT_LOW_INITIAL, 30
 - OBJ_AI3_NAME_INIT_STRING, 30
 - OBJ_AI3_PRESENT_VALUE_INITIAL, 31
 - OBJ_AI3_UNIT_INITIAL, 31
 - OBJ_AI4_COV_INCREMENT_INITIAL, 31
 - OBJ_AI4_DESCRIPTION_INIT_STRING, 31
 - OBJ_AI4_LIMIT_DEADBAND_INITIAL, 31
 - OBJ_AI4_LIMIT_HIGH_INITIAL, 31
 - OBJ_AI4_LIMIT_LOW_INITIAL, 31
 - OBJ_AI4_NAME_INIT_STRING, 31
 - OBJ_AI4_PRESENT_VALUE_INITIAL, 31
 - OBJ_AI4_UNIT_INITIAL, 31
 - ObjAllInit, 32
 - ObjAIPresentValueUpdate, 32
 - ObjAIReset, 32
- obj_ao.c
 - g_obj_ao1_description, 33
 - g_obj_ao1_evt_msg_txt, 34
 - g_obj_ao1_out_of_service, 34
 - g_obj_ao1_present_value, 34
 - g_obj_ao1_state_flags, 34
 - ObjAONit, 33
 - ObjAOReset, 33
 - ObjAOWriteHandling, 33
- obj_ao.h
 - OBJ_AO1_DESCRIPTION_INIT_STRING, 34
 - OBJ_AO1_NAME_INIT_STRING, 34
 - ObjAONit, 35
 - ObjAOReset, 35
 - ObjAOWriteHandling, 35
- obj_bi.c
 - g_obj_bi1_description, 37
 - g_obj_bi1_evt_msg_txt, 37
 - g_obj_bi1_out_of_service, 37
 - g_obj_bi1_present_value, 37
 - g_obj_bi1_state_flags, 37
 - g_obj_bi2_description, 37
 - g_obj_bi2_evt_msg_txt, 37
 - g_obj_bi2_out_of_service, 37
 - g_obj_bi2_present_value, 37
 - g_obj_bi2_state_flags, 37
 - g_obj_dev_id, 37
 - ObjBIInit, 36
 - ObjBIPresentValueUpdate, 36
 - ObjBIReset, 36
- obj_bi.h
 - OBJ_BI1_ACTIVE_INIT_STRING, 38
 - OBJ_BI1_DESCRIPTION_INIT_STRING, 38
 - OBJ_BI1_INACTIVE_INIT_STRING, 38
 - OBJ_BI1_NAME_INIT_STRING, 38
 - OBJ_BI2_ACTIVE_INIT_STRING, 38
 - OBJ_BI2_DESCRIPTION_INIT_STRING, 38
 - OBJ_BI2_INACTIVE_INIT_STRING, 38
 - OBJ_BI2_NAME_INIT_STRING, 38
 - ObjBIInit, 38

- ObjBIPresentValueUpdate, 38
- ObjBIReset, 39
- obj_bo.c
 - g_obj_bo1_description, 40
 - g_obj_bo1_out_of_service, 40
 - g_obj_bo1_present_value, 40
 - g_obj_bo1_state_flags, 40
 - g_obj_bo2_description, 40
 - g_obj_bo2_out_of_service, 40
 - g_obj_bo2_present_value, 40
 - g_obj_bo2_state_flags, 40
 - ObjBOInit, 39
 - ObjBOReset, 39
 - ObjBOWriteHandling, 40
- obj_bo.h
 - OBJ_BO1_ACTIVE_INIT_STRING, 41
 - OBJ_BO1_DESCRIPTION_INIT_STRING, 41
 - OBJ_BO1_INACTIVE_INIT_STRING, 41
 - OBJ_BO1_NAME_INIT_STRING, 41
 - OBJ_BO2_ACTIVE_INIT_STRING, 41
 - OBJ_BO2_DESCRIPTION_INIT_STRING, 41
 - OBJ_BO2_INACTIVE_INIT_STRING, 42
 - OBJ_BO2_NAME_INIT_STRING, 42
 - ObjBOInit, 42
 - ObjBOReset, 42
 - ObjBOWriteHandling, 42
- obj_ca.c
 - g_obj_ca1_description, 43
 - ObjCAInit, 43
- obj_ca.h
 - OBJ_CA1_DESCRIPTION_INIT_STRING, 44
 - OBJ_CA1_NAME_INIT_STRING, 44
 - ObjCAInit, 44
- obj_dev.c
 - g_obj_dev_description, 45
 - g_obj_dev_id, 45
 - g_obj_dev_location, 45
 - g_obj_dev_name, 45
 - g_obj_dev_time_of_restart, 45
 - ObjDevInit, 45
- obj_dev.h
 - OBJ_DEV_DESCRIPTION_INIT_STRING, 46
 - OBJ_DEV_FIRMWARE_REV_STRING, 46
 - OBJ_DEV_INST_NUMBER, 46
 - OBJ_DEV_LOCATION_INIT_STRING, 46
 - OBJ_DEV_MAX_APDU_SIZE, 46
 - OBJ_DEV_MODEL_NAME_STRING, 46
 - OBJ_DEV_NAME_INIT_STRING, 46
 - OBJ_DEV_SOFTWARE_VER_STRING, 46
 - OBJ_DEV_VENDOR_ID_NUMBER, 46
 - OBJ_DEV_VENDOR_NAME_STRING, 46
 - ObjDevInit, 47
- obj_nc.c
 - g_obj_nc1_description, 47
 - ObjNCInit, 47
- obj_nc.h
 - OBJ_NC1_DESCRIPTION_INIT_STRING, 48
 - OBJ_NC1_NAME_INIT_STRING, 48
- ObjNCInit, 48
- obj_sc.c
 - g_obj_sc1_description, 49
 - g_obj_sc1_out_of_service, 49
 - g_obj_sc1_state_flags, 49
 - ObjSCInit, 49
 - ObjSCReset, 49
- obj_sc.h
 - OBJ_SC1_DAILY_SCHEDULE_SIZE, 50
 - OBJ_SC1_DESCRIPTION_INIT_STRING, 50
 - OBJ_SC1_NAME_INIT_STRING, 50
 - ObjSCInit, 50
 - ObjSCReset, 50
- obj_tr.c
 - g_obj_tr1_description, 51
 - g_obj_tr1_evt_msg_txt, 51
 - ObjTRInit, 51
 - ObjTRWriteHandling, 51
- obj_tr.h
 - OBJ_TR1_DESCRIPTION_INIT_STRING, 52
 - OBJ_TR1_LOG_BUFFER_SIZE, 52
 - OBJ_TR1_NAME_INIT_STRING, 52
 - ObjTRInit, 52
 - ObjTRWriteHandling, 52
- ObjAllInit
 - obj_ai.c, 26
 - obj_ai.h, 32
- ObjAIPresentValueUpdate
 - obj_ai.c, 26
 - obj_ai.h, 32
- ObjAIReset
 - obj_ai.c, 26
 - obj_ai.h, 32
- ObjAOLInit
 - obj_ao.c, 33
 - obj_ao.h, 35
- ObjAOReset
 - obj_ao.c, 33
 - obj_ao.h, 35
- ObjAOWriteHandling
 - obj_ao.c, 33
 - obj_ao.h, 35
- ObjBIInit
 - obj_bi.c, 36
 - obj_bi.h, 38
- ObjBIPresentValueUpdate
 - obj_bi.c, 36
 - obj_bi.h, 38
- ObjBIReset
 - obj_bi.c, 36
 - obj_bi.h, 39
- ObjBOInit
 - obj_bo.c, 39
 - obj_bo.h, 42
- ObjBOReset
 - obj_bo.c, 39
 - obj_bo.h, 42
- ObjBOWriteHandling

- obj_bo.c, 40
- obj_bo.h, 42
- ObjCAInit
 - obj_ca.c, 43
 - obj_ca.h, 44
- ObjDevInit
 - obj_dev.c, 45
 - obj_dev.h, 47
- ObjNCInit
 - obj_nc.c, 47
 - obj_nc.h, 48
- ObjSCInit
 - obj_sc.c, 49
 - obj_sc.h, 50
- ObjSCReset
 - obj_sc.c, 49
 - obj_sc.h, 50
- ObjTRInit
 - obj_tr.c, 51
 - obj_tr.h, 52
- ObjTRWriteHandling
 - obj_tr.c, 51
 - obj_tr.h, 52
- pin
 - user_button_t, 9
- pos
 - user_trendlog_t, 10
- records
 - user_trendlog_t, 10
- resolution
 - user_analog_input_t, 8
- set_flash_magic
 - user.c, 62
- show_led
 - user.c, 62
- src/demo/adc.c, 11
- src/demo/adc.h, 13
- src/demo/bacnetdemo.c, 14
- src/demo/bacnetdemo.h, 19
- src/demo/definitions.h, 21
- src/demo/obj_ai.c, 25
- src/demo/obj_ai.h, 27
- src/demo/obj_ao.c, 32
- src/demo/obj_ao.h, 34
- src/demo/obj_bi.c, 35
- src/demo/obj_bi.h, 37
- src/demo/obj_bo.c, 39
- src/demo/obj_bo.h, 40
- src/demo/obj_ca.c, 43
- src/demo/obj_ca.h, 43
- src/demo/obj_dev.c, 44
- src/demo/obj_dev.h, 45
- src/demo/obj_nc.c, 47
- src/demo/obj_nc.h, 47
- src/demo/obj_sc.c, 48
- src/demo/obj_sc.h, 49
- src/demo/obj_tr.c, 50
- src/demo/obj_tr.h, 51
- src/demo/user.c, 53
- src/demo/user.h, 70
- state
 - user_binary_input_t, 8
- USER_ANALOG_AVERAGE_COUNT
 - user.c, 56
- USER_BINARY_VALUE_IS_BUTTON
 - user.c, 56
- USER_MAX_ANALOG
 - user.c, 56
- USER_MAX_BINARY
 - user.c, 56
- USER_MAX_RECORDS
 - user.c, 57
- USER_MIN_VALUE_DIFF
 - user.c, 57
- user.c
 - analog_value_changed, 57
 - button_pressed, 57
 - calc_analog_value, 57
 - copy_trendlog_data, 58
 - FLASH_DEVICE_FLASH_SIZE, 56
 - FLASH_DEVICE_PAGE_SIZE, 56
 - FLASH_DEVICE_START_ADDRESS, 56
 - FLASH_MEM_MAGIC_SIZE, 56
 - flash_wait, 58
 - flash_write, 58
 - flash_write_page, 59
 - g_flash_checksum, 68
 - g_flash_magic, 68
 - g_flash_mem_page, 68
 - g_flash_offset, 68
 - g_obj_sc1_daily_schedule, 68
 - g_obj_sc1_obj_prop_ref_list, 68
 - g_obj_sc1_obj_prop_ref_listCount, 68
 - g_page_any_page_changed, 68
 - g_page_data_changed, 68
 - g_page_is_new, 69
 - g_page_offset, 69
 - g_sequence, 69
 - g_user_analog_values, 69
 - g_user_binary_values, 69
 - g_user_button, 69
 - g_user_led_pins, 69
 - g_user_leds, 69
 - g_user_mutex, 70
 - g_user_thread, 70
 - g_user_thread_stack, 70
 - g_user_trendlog, 70
 - get_crc_checksum, 59
 - get_switch_state, 59
 - get_trendlog_first_last, 60
 - get_trendlog_index_next, 60
 - get_trendlog_index_prev, 60
 - get_trendlog_index_start, 61
 - gp_flash_mem, 70

- init_scheduler, 61
- init_trendlog, 61
- is_flash_magic_activated, 61
- is_property_persistent, 61
- is_trendlog_index_valid, 62
- is_trendlog_time_smaller, 62
- set_flash_magic, 62
- show_led, 62
- USER_ANALOG_AVERAGE_COUNT, 56
- USER_BINARY_VALUE_IS_BUTTON, 56
- USER_MAX_ANALOG, 56
- USER_MAX_BINARY, 56
- USER_MAX_RECORDS, 57
- USER_MIN_VALUE_DIFF, 57
- user_loop, 63
- user_reset_values, 63
- UserInit, 63
- UserReset, 63
- UserSetLed, 64
- UserStorageDelete, 64
- UserStorageIsPropertyPersistent, 64
- UserStorageRead, 65
- UserStorageWrite, 65
- UserTrendlogClearBuffer, 65
- UserTrendlogGetRecords, 65
- UserTrendlogGetRecordsByPosition, 66
- UserTrendlogGetRecordsByTime, 66
- UserTrendlogGetRecordsByTimeRange, 67
- UserTrendlogPutRecord, 67
- UserUserTrendlogGetRecordsBySequence, 67
- user.h
 - UserInit, 71
 - UserReset, 72
 - UserSetLed, 72
 - UserStorageDelete, 72
 - UserStorageIsPropertyPersistent, 72
 - UserStorageRead, 73
 - UserStorageWrite, 73
 - UserTrendlogClearBuffer, 73
 - UserTrendlogGetRecords, 74
 - UserTrendlogGetRecordsByPosition, 74
 - UserTrendlogGetRecordsByTime, 74
 - UserTrendlogGetRecordsByTimeRange, 75
 - UserTrendlogPutRecord, 75
 - UserUserTrendlogGetRecordsBySequence, 76
- user_analog_input_t, 7
 - adc_sum, 7
 - adc_sum_count, 7
 - adc_value, 7
 - bacnet_value, 7
 - resolution, 8
- user_binary_input_t, 8
 - button, 8
 - state, 8
- user_button_t, 8
 - level, 9
 - pin, 9
- user_loop
 - user.c, 63
- user_record_t, 9
 - data, 9
- user_reset_values
 - user.c, 63
- user_trendlog_t, 10
 - count, 10
 - pos, 10
 - records, 10
- UserInit
 - user.c, 63
 - user.h, 71
- UserReset
 - user.c, 63
 - user.h, 72
- UserSetLed
 - user.c, 64
 - user.h, 72
- UserStorageDelete
 - user.c, 64
 - user.h, 72
- UserStorageIsPropertyPersistent
 - user.c, 64
 - user.h, 72
- UserStorageRead
 - user.c, 65
 - user.h, 73
- UserStorageWrite
 - user.c, 65
 - user.h, 73
- UserTrendlogClearBuffer
 - user.c, 65
 - user.h, 73
- UserTrendlogGetRecords
 - user.c, 65
 - user.h, 74
- UserTrendlogGetRecordsByPosition
 - user.c, 66
 - user.h, 74
- UserTrendlogGetRecordsByTime
 - user.c, 66
 - user.h, 74
- UserTrendlogGetRecordsByTimeRange
 - user.c, 67
 - user.h, 75
- UserTrendlogPutRecord
 - user.c, 67
 - user.h, 75
- UserUserTrendlogGetRecordsBySequence
 - user.c, 67
 - user.h, 76