AGHAF

4.3.5

Generated by Doxygen 1.8.17

Fri Dec 23 2022 15:39:36

Chapter 1

Main Page

AGHAF is a library for accessing Annecy Electronique's MUX devices.

This documentation is about the C API for accessing AGHAF.

The main functionnalities provided are:

- Enumerate and identify connected devices
- Configure devices for network communication (CAN, Ethernet, LIN, etc.)
- Receive data from configured networks, e.g. for analysis purposes.
- Send and receive data on configured networks, e.g. to diagnose or simulate ECU.

1.1 Environment

1.1.1 Compatibility

The C API and the AGHAF library should be compatible with any C or C++ compiler.

Wrappers for other programming languages like C#, VB.NET and PowerShell are available.

1.1.2 Supported platforms

Supported platforms are:

- Microsoft Windows 32 and 64 bits (Vista and above)

The following platforms will be supported in the near future:

- Linux 32 and 64 bits

1.2 Introduction to the API

The C API has been designed in object oriented mode. To achieve this, most of the APIs use pointers.

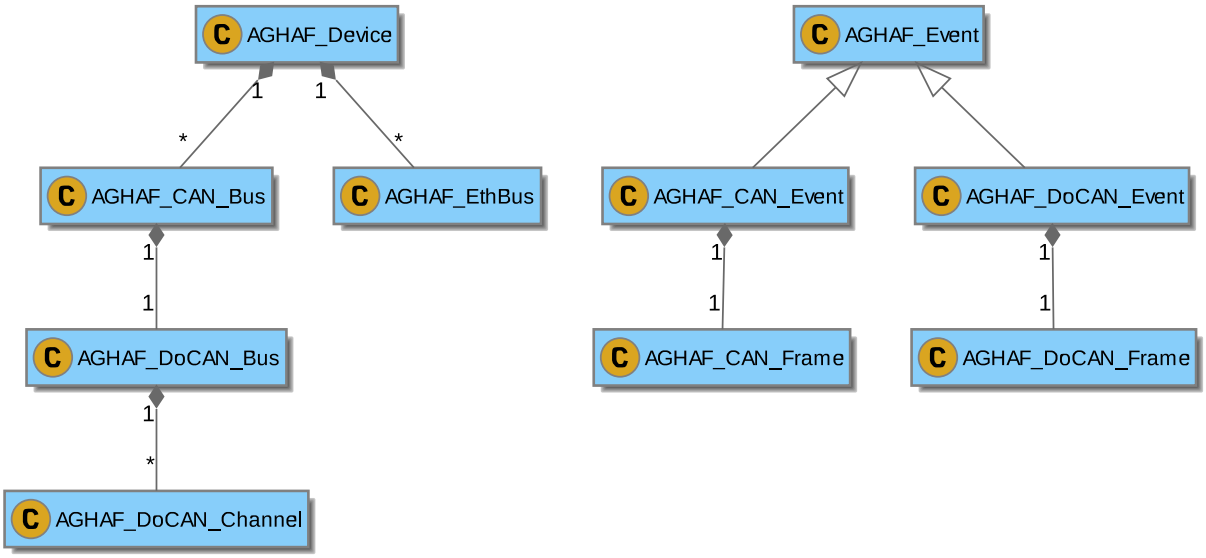
Thus the API will contain a very limited number of structures and most of the objects are accessible through handles (`void *`). Main drawback is that users do not have direct access objects internals. However, this provides many advantages:

- Users do not have to deal with object initialization.
- Permitted actions are clearly defined by the set of functions related to the object.

Below, the description of VCI Muxdiag 3 pinout :

Signal	Pin num	comment
Ethernet Tx(+)	1	
BroadReach N	2	
BroadReach P	3	
Chassis Power Ground	4	
Signal Communication Groud	5	
CAN LS 1 L	6	
CAN LS 1 H	7	
CAN HS 1 L	8	
CAN HS 1 H	9	
Ethernet Rx(+)	10	
Ethernet Tx(-)	11	
ISO K line 1	12	
VBAT	16	
CAN HS 2 H	17	
CAN HS 2 L	18	
Ethernet Rx(-)	19	
ISO L Line 1	20	
internal use	21	do not connect
internal use	22	do not connect
DoIP activation signal	23	
APC	24	
LIN 2	25	
LIN 1	26	

1.3 diagram



1.4 changelog

Version

1.0.1.x to 1.2.x

- First api version

1.3.0 and 1.4.0 – 2019-06-20

- Fixes api for c compliant.
- Fixes AGHAF_Device_getFriendlyNameEthCard.
- AGHAF_Device_delete is deprecated. Device will be deleted internal after the device_leave event.
- Add AGHAF_getDeviceByHardID for internal use.
- Add DOCAN functionalities.

1.5.0 – 2019-07-15

- Fixes DOCAN channel internal management.
- Fixes DOCAN channel setting.
- Change call order for DOCAN.

1.6.0 – 2019-09-11

- Loses binary compatibility because of canBusCount docanBusCount and expert speed configuration.
- Rename function: AGHAF_CAN_getEventBusInfo instead of AGHAF_CAN_getEventBusLoad.
- Rename type: AGHAF_CanEvent_busInfo instead of AGHAF_CanEvent_busLoad.
- Add AGHAF_DOCAN_getChannelCount and AGHAF_DOCAN_getUnusedChannelCount.

1.6.1 – 2019-09-13

- Fix load of AGHAF_DOCAN_getChannelCount and AGHAF_DOCAN_getUnusedChannelCount.
- Fix source and destination inverted.

1.6.2 – 2019-10-07

- Beta with CAN error management.

1.6.3 – 2019-11-04

- Add periodic frame.
- Add data in event tx.
- Loses binary compatibility because of changing timestamp from 32 bits to 64 bits.

1.6.4 – 2019-11-22

- Beta with full DOCAN implementation.

1.6.5 – 2019-12-20

- Start support of last error string: for device and general module.
- Fix some isue with docan.
- Improve max data in docan frame : 64 k.
- Add error event in DoCan.
- Fix event message order. Event messages are no longer mixed.

1.6.6 – 2020-01-28

- remove debug trace.
- only one application can use a device at same time.
- fix error on disconnected device.

1.6.7 – 2020-02-28

- add forgot function AGHAF_DOCAN_getEventChannel.
- add doc for AGHAF_DOCAN_setSource and AGHAF_DOCAN_setDestination

1.6.8 – 2020-03-09

- fix AGHAF_DOCAN_getEventChannel : function return nullptr.

1.6.9 – 2020-03-12

- change version for correlation with development kit.

1.6.10 – 2020-03-16

- fix AGHAF_DOCAN_getEventChannel : function fail if single channel used.

1.6.11 – 2020-03-18

- change version for correlation with development kit.

1.7.0 – 2020-06-12

- change the DoCAN API

1.7.1 – 2020-07-22

- change the CAN API

1.5 compatibility

Below, the compatibility of VCI Muxdiag 3 firmware and service.

Firmwar Version	SDK 3.0.0	SDK 3.0.1	SDK 3.1.0	SDK 3.1.1	SDK 3.1.2	SDK 3.1.3	SDK 3.1.4	SDK 3.1.5	SDK 3.1.6	SDK 4.0.6
1.4 and before	not compatible	not compatible	not compatible	not compatible	not compatible	not compatible	not compatible	not compatible	not compatible	not compatible
1.5	not compatible	not compatible	not compatible	not compatible	not compatible	not compatible	not compatible	not compatible	not compatible	not compatible
1.9	not compatible	not compatible	not compatible	not compatible	not compatible	not compatible	not compatible	not compatible	not compatible	not compatible
1.10	not compatible	not compatible	not compatible	not compatible	not compatible	not compatible	not compatible	not compatible	not compatible	not compatible
1.11	compatible	compatible	compatible	compatible	compatible	compatible	compatible	compatible	compatible	not compatible
1.12	compatible	compatible	compatible	compatible	compatible	compatible	compatible	compatible	compatible	not compatible
1.13	not compatible	not compatible	not compatible	not compatible	not compatible	not compatible	not compatible	not compatible	not compatible	compatible

Chapter 2

Module Index

2.1 Modules

Here is a list of all modules:

Chapter 3

Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 4

File Index

4.1 File List

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

Chapter 5

Module Documentation

5.1 CAN

API to use CAN on exxotest devices.

- enum AGHAF_CAN_Error { AGHAF_CAN_ERROR_NONE = 0, AGHAF_CAN_ERROR_STUFF, AGHAF_CAN_ERROR_FORM, AGHAF_CAN_ERROR_ACK, AGHAF_CAN_ERROR_BIT1, AGHAF_CAN_ERROR_BIT0, AGHAF_CAN_ERROR_CRC }
- enum AGHAF_CAN_Mode { AGHAF_CAN_MODE_FD = 0x00, AGHAF_CAN_MODE_HS = 0x01, AGHAF_CAN_MODE_LS = 0x02 }

  Possible modes for CAN.
- enum AGHAF_CAN_Param { CAN_PARAM_MODE, CAN_PARAM_BAUDRATE, CAN_PARAM_BAUDRATE_FD, CAN_PARAM_SAMPLEPOINT, CAN_PARAM_SAMPLEPOINT_FD, CAN_PARAM_SYNCJUMP_WIDTH, CAN_PARAM_SYNCJUMP_WIDTH_FD, CAN_PARAM_LISTEN_ONLY, CAN_PARAM_TERMINATION, CAN_PARAM_TERMINATION_LS, CAN_PARAM_NO_EVENT }

  List of parameters used to configure a CAN bus.
- #define AGHAF_CAN_PERIODIC_NO_RESTART (1<<0)

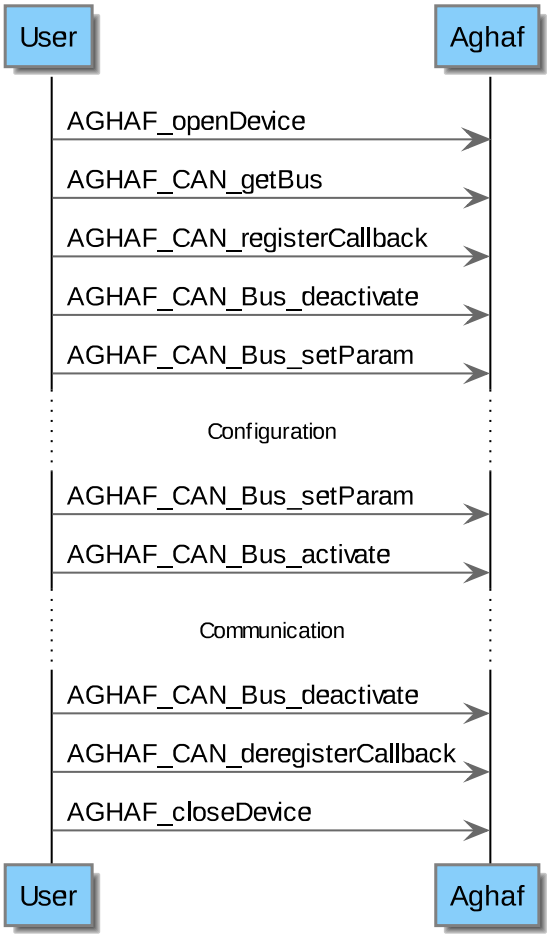
  Don't restart period.

- enum AGHAF_CAN_IdentifierType { AGHAF_CAN_IdStd, AGHAF_CAN_IdXtd }
- typedef void * AGHAF_CAN_Bus

5.1.1 Detailed Description

API to use CAN on exxotest devices.

5.1.2 Usage Example



5.1.3 Call order and priority

Device is in "init" state at startup.

The following section describes state management:

- When calling AGHAF_CAN_Bus_activate, the associated CAN module goes in "active" state.
- When calling AGHAF_CAN_Bus_deactivate, the associated CAN module goes in "config" state.

Symbol	Description
	Forbidden
	Authorized
	Followed status

Function	Config	Active
AGHAF_CAN_getBus		
AGHAF_CAN_getBusCount		
AGHAF_CAN_registerCallback		
AGHAF_CAN_deregisterCallback		
AGHAF_CAN_Bus_getParam		
AGHAF_CAN_Bus_setParam		
AGHAF_CAN_Bus_supportedModes		
AGHAF_CAN_Bus_addFilter		
AGHAF_CAN_Bus_clearFilters		
AGHAF_CAN_Bus_filtersCount		
AGHAF_CAN_Bus_reject		
AGHAF_CAN_Bus_supportedFilters		

Function	Config	Active
AGHAF_CAN_Bus_activate		
AGHAF_CAN_Bus_deactivate		
AGHAF_CAN_Bus_isActivated		
AGHAF_CAN_Bus_supportedTerminations		
AGHAF_CAN_Bus_supportedTerminationsLs		
AGHAF_CAN_Bus_state		
AGHAF_CAN_Bus_BusOn		
AGHAF_CAN_sendFrame		
AGHAF_CAN_Bus_sendPeriodic		
AGHAF_CAN_Bus_getPeriodicCount		

5.1.4 Typedef Documentation

5.1.4.1 AGHAF_CAN_Bus

`AGHAF_CAN_Bus`

Handle to a CAN bus.

5.1.5 Enumeration Type Documentation

5.1.5.1 AGHAF_CAN_Error

enum AGHAF_CAN_Error

Enumerator

AGHAF_CAN_ERROR_NONE	No error.
AGHAF_CAN_ERROR_STUFF	More than 5 equal bits in a sequence have occurred in a part of a received message where this is not allowed.
AGHAF_CAN_ERROR_FORM	A fixed format part of a received frame has the wrong format.
AGHAF_CAN_ERROR_ACK	The message transmitted by the device was not acknowledged by another node.
AGHAF_CAN_ERROR_BIT1	Device tried to send a recessive level but the monitored bus value was dominant.
AGHAF_CAN_ERROR_BIT0	Device tried to send a dominant level but the monitored bus value was recessive.
AGHAF_CAN_ERROR_CRC	The CRC check sum of a received message was incorrect.

5.1.5.2 AGHAF_CAN_IdentifierType

enum AGHAF_CAN_IdentifierType

Enumerator

AGHAF_CAN_IdStd	Standard 11 bit identifier.
AGHAF_CAN_IdXtd	Extended 29 bit identifier.

5.1.5.3 AGHAF_CAN_Mode

enum AGHAF_CAN_Mode

Possible modes for CAN.

Enumerator

AGHAF_CAN_MODE_FD	CAN FD Enable CAN High Speed and FD transmition and reception. Use CAN High Speed and FD termination settings (CAN_PARAM_TERMINATION).

Enumerator

AGHAF_CAN_MODE_HS	CAN High Speed Enable CAN High Speed transmition and reception. Use CAN High Speed and FD termination settings (CAN_PARAM_TERMINATION).
AGHAF_CAN_MODE_LS	CAN Low Speed Enable CAN Low Speed transmition and reception. Use CAN Low Speed termination settings (CAN_PARAM_TERMINATION_LS).

5.1.5.4 AGHAF_CAN_Param

`enum AGHAF_CAN_Param`

List of parameters used to configure a CAN bus.

Enumerator

CAN_PARAM_MODE	CAN mode settings. Use: AGHAF_CAN_Mode Default: AGHAF_CAN_MODE_FD
CAN_PARAM_BAUDRATE	Nominal CAN baudrate in bps Max: 1000000 bps (1 Mbps) Default: 500000 bps (500 kbps)
CAN_PARAM_BAUDRATE_FD	Baudrate to be used during the high bitrate phase of CAN FD frame Max: 5000000 bps (5 Mbps) Default: 1000000 bps (1 Mbps)
CAN_PARAM_SAMPLEPOINT	Desired bit sample point as a percentage of the bit time Range: [0, 100] % Default: 80 %
CAN_PARAM_SAMPLEPOINT_FD	Desired bit sample point as a percentage of the bit time to be used during the high bitrate phase of CAN FD frame Range: [0, 100] % Default: 80 %
CAN_PARAM_SYNCJUMP_WIDTH	Desired synchronization jump bit has a percentage of bit time Range: [0, 100] % Default: 15 %
CAN_PARAM_SYNCJUMP_WIDTH_FD	Desired synchronization jump bit has a percentage of bit time to be used during the high bitrate phase of CAN FD frame Range: [0, 100] % Default: 15 %

Enumerator

CAN_PARAM_LISTEN_ONLY	Enable listen only mode. CAN controler will no longer acknowledge received frames. Range: [0, 1] Default: 0 (disabled)
CAN_PARAM_TERMINATION	CAN High Speed and FD termination settings. Use one of the value returned by supported CAN termination function. 0 no termination Default: 0 (no termination)
CAN_PARAM_TERMINATION_LS	CAN Low Speed termination settings. Use one of the value returned by supported CAN Low Speed termination function. 0 no termination Default: 0 (no termination)
CAN_PARAM_NO_EVENT	Disable CAN events reporting. This will significantly reduce the load on the USB bus if you don't need these events. Protocols requiring CAN will continue to function normally (e.g. DoCAN). Range: [0, 1] Default: 0 (report CAN events)

5.2 DoCAN

API to use DoCAN on exxotest devices.

- enum AGHAF_DoCan_Error { AGHAF_DOCAN_ERROR_TO_A = 0, AGHAF_DOCAN_ERROR_TO_BS = 1, AGHAF_DOCAN_ERROR_TO_CR = 2, AGHAF_DOCAN_ERROR_SN = 3, AGHAF_DOCAN_ERROR_FS = 4, AGHAF_DOCAN_ERROR_PDU = 5, AGHAF_DOCAN_ERROR_WAIT = 6, AGHAF_DOCAN_ERROR_OVFLW = 7, AGHAF_DOCAN_ERROR_OTHER = 8 }
- enum AGHAF_DoCAN_AddrMode { AGHAF_DoCAN_ADDR_MODE_PHYS = 0x00, AGHAF_DoCAN_ADDR_MODE_FUNC = 0x01 }

  Possible addressing modes for DoCAN.
- enum AGHAF_DoCAN_ChannelDirection { AGHAF_DoCAN_CHANNEL_DIR_RX = (1U<<0), AGHAF_DoCAN_CHANNEL_D = (1U<<1), AGHAF_DoCAN_CHANNEL_DIR_2WAY = (AGHAF_DoCAN_CHANNEL_DIR_RX | AGHAF↩ _DoCAN_CHANNEL_DIR_TX) }

  Possible directions for DoCAN channels.
- enum AGHAF_DoCAN_PadHandling { AGHAF_DoCAN_PAD_DISABLE = 0x00, AGHAF_DoCAN_PAD_ENABLE = 0x01 }

  Enable or disable padding handling.
- enum AGHAF_DoCAN_TxDL { AGHAF_DoCAN_TX_DL_CLASSIC_CAN = 0, AGHAF_DoCAN_TX_DL_8 = 8, AGHAF_DoCAN_TX_DL_12 = 9, AGHAF_DoCAN_TX_DL_16 = 10, AGHAF_DoCAN_TX_DL_20 = 11, AGHAF_DoCAN_TX_DL_24 = 12, AGHAF_DoCAN_TX_DL_32 = 13, AGHAF_DoCAN_TX_DL_48 = 14, AGHAF_DoCAN_TX_DL_64 = 15 }

  Possible TX_DL for DoCAN transmissions.
- enum AGHAF_DoCAN_IdFormat { AGHAF_DoCAN_ID_FMT_XTD = (1U<<0), AGHAF_DoCAN_ID_FMT_AE = (1U<<1) }

  Flags to configure DoCAN identifier format.
- enum AGHAF_DoCAN_ChannelParam { AGHAF_DoCAN_CHANNEL_PARAM_DIRECTION = 0, AGHAF_DoCAN_CHANNEL_PARAM_TXDL, AGHAF_DoCAN_CHANNEL_PARAM_PAD_HANDLING, AGHAF_DoCAN_CHANNEL_PARAM_PAD_VALUE, AGHAF_DoCAN_CHANNEL_PARAM_ADDRMODE, AGHAF_DoCAN_CHANNEL_PARAM_TIMING_AS, AGHAF_DoCAN_CHANNEL_PARAM_TIMING_AR, AGHAF_DoCAN_CHANNEL_PARAM_TIMING_BS, AGHAF_DoCAN_CHANNEL_PARAM_TIMING_BR, AGHAF_DoCAN_CHANNEL_PARAM_TIMING_CS, AGHAF_DoCAN_CHANNEL_PARAM_TIMING_CR, AGHAF_DoCAN_CHANNEL_PARAM_TIMING_STMIN, AGHAF_DoCAN_CHANNEL_PARAM_REQ_ID_FMT, AGHAF_DoCAN_CHANNEL_PARAM_REQ_ID_VALUE, AGHAF_DoCAN_CHANNEL_PARAM_REQ_ID_AE, AGHAF_DoCAN_CHANNEL_PARAM_RESP_ID_FMT, AGHAF_DoCAN_CHANNEL_PARAM_RESP_ID_VALUE, AGHAF_DoCAN_CHANNEL_PARAM_RESP_ID_AE, AGHAF_DoCAN_CHANNEL_PARAM_WAITFRAME, AGHAF_DoCAN_CHANNEL_PARAM_BLOCKSIZE, AGHAF_DoCAN_CHANNEL_PARAM_MAXDATALEN, AGHAF_DoCAN_CHANNEL_PARAM_PAD_NOCHECK, AGHAF_DoCAN_CHANNEL_PARAM_CANFD_BRS, AGHAF_DoCAN_CHANNEL_PARAM_TIMING_STMIN_OVRD, AGHAF_DoCAN_CHANNEL_PARAM_BS_OVRD }

  List of parameters used to configure a DoCAN channel.

- enum AGHAF_DoCAN_EventType { AGHAF_DoCAN_EVENT_MSGTX = 0, AGHAF_DoCAN_EVENT_MSGTXERR = 1, AGHAF_DoCAN_EVENT_MSGRX = 2, AGHAF_DoCAN_EVENT_MSGRXFF = 3, AGHAF_DoCAN_EVENT_MSGRXERR = 4 }
- enum AGHAF_DoCAN_ERROR { **AGHAF_DoCAN_ERROR_UNKNOWN** = 0, AGHAF_DoCAN_ERROR_OK = 1, AGHAF_DoCAN_ERROR_SEQUENCE = 2, AGHAF_DoCAN_ERROR_PARAM = 3, AGHAF_DoCAN_ERROR_UNIMPLEMENTED = 4, AGHAF_DoCAN_ERROR_FIFOFULL = 5, AGHAF_DoCAN_ERROR_BUS = 6 }

- typedef void * AGHAF_DoCAN_Bus
- typedef void * AGHAF_DoCAN_Frame
- typedef void * AGHAF_DoCAN_Event
- typedef void * AGHAF_DoCAN_Channel
- typedef void(* AGHAF_DoCAN_Callback) (AGHAF_DoCAN_Event, void *)
- AGHAF_DoCAN_Bus AGHAF_API AGHAF_DoCAN_getBus (AGHAF_Device device, uint8_t index)

  return the DoCAN bus at index in parameter
- uint32_t AGHAF_API AGHAF_DoCAN_getBusCount (AGHAF_Device device)

  return the number of DoCAN bus from a device
- void AGHAF_API AGHAF_DoCAN_registerCallback (AGHAF_DoCAN_Bus bus, AGHAF_DoCAN_Callback callback, void *userContext)

  Register a callback for the DoCAN events from a bus.
- void AGHAF_API AGHAF_DoCAN_deregisterCallback (AGHAF_DoCAN_Bus bus, AGHAF_DoCAN_Callback callback)

  deregister a callback
- AGHAF_Status AGHAF_API AGHAF_DoCAN_getChannelCount (AGHAF_DoCAN_Bus bus, uint8_t *count)

  Provide the number of channels available on the bus.
- AGHAF_Status AGHAF_API **AGHAF_DoCAN_getFreeChannelCount** (AGHAF_DoCAN_Bus bus, uint8_t *count)
- AGHAF_DoCAN_Channel AGHAF_API AGHAF_DoCAN_createChannel (AGHAF_DoCAN_Bus bus)

  Create a channel.
- AGHAF_Status AGHAF_API AGHAF_DoCAN_releaseChannel (AGHAF_DoCAN_Channel channel)

  Release the channel.
- AGHAF_Status AGHAF_API AGHAF_DoCAN_Channel_setParam (AGHAF_DoCAN_Channel channel, AGHAF_DoCAN_ChannelParam param, uint32_t value)

  Set the parameter from a channel.
- AGHAF_Status AGHAF_API AGHAF_DoCAN_Channel_getParam (AGHAF_DoCAN_Channel channel, AGHAF_DoCAN_ChannelParam param, uint32_t *value)

  Provide the value from a channel parameter.
- AGHAF_Status AGHAF_API AGHAF_DoCAN_Channel_start (AGHAF_DoCAN_Channel channel)

  Start a channel.
- AGHAF_Status AGHAF_API AGHAF_DoCAN_Channel_stop (AGHAF_DoCAN_Channel channel)

  Stop a channel.
- AGHAF_Status AGHAF_API AGHAF_DoCAN_Channel_send (AGHAF_DoCAN_Channel channel, uint32_t dataLen, uint8_t const *data)

  Send on the DoCAN channel in parameter.
- void AGHAF_API AGHAF_DoCAN_Channel_activateTrace (AGHAF_DoCAN_Channel channel, AGHAF_BOOL value)

  Activate the CAN trace.
- AGHAF_DoCAN_EventType AGHAF_API AGHAF_DoCAN_Event_type (AGHAF_EventInfo event)

  return the type from the event
- uint8_t AGHAF_API AGHAF_DoCAN_Event_getBusIndex (AGHAF_DoCAN_Event event)

  return the index from the bus on which occured the event
- uint32_t AGHAF_API AGHAF_DoCAN_Event_getDataSize (AGHAF_DoCAN_Event event)

  return the size from the datas in the event
- AGHAF_Status AGHAF_API AGHAF_DoCAN_Event_getData (AGHAF_DoCAN_Event event, uint8_t *data, uint32_t *size)

  Provide the datas received in the event.
- AGHAF_CAN_Identifier AGHAF_API AGHAF_DoCAN_Event_getId (AGHAF_DoCAN_Event event)

  Return the CAN identifier from the event.
- uint8_t AGHAF_API AGHAF_DoCAN_Event_getError (AGHAF_DoCAN_Event event)

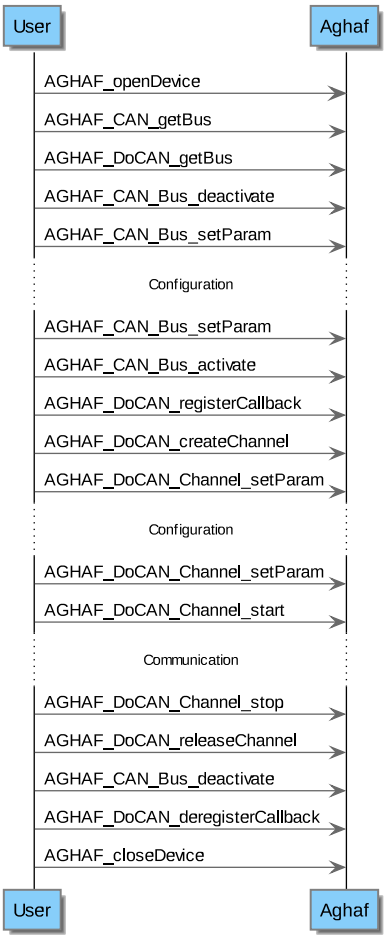
  Return the error value from the event.
- AGHAF_DoCAN_Channel AGHAF_API AGHAF_DoCAN_Event_getChannel (AGHAF_DoCAN_Event event)

  Return the channel on which occured the event.

5.2.1 Detailed Description

API to use DoCAN on exxotest devices.

5.2.2 Usage Example



5.2.3 Call order and priority

below, the description of channel state management:

- When you call AGHAF_DoCAN_Channel_start, the channel goes in "Start" state.
- When you call AGHAF_DoCAN_Channel_stop, the channel goes in "Stop" state.

Symbol	Description
	Forbidden
	Authorized

Symbol	Description
	Next state

Function	Stop	Start
AGHAF_DoCAN_Channel_setParam		
AGHAF_DoCAN_Channel_getParam		
AGHAF_DoCAN_Channel_start		
AGHAF_DoCAN_Channel_stop		
AGHAF_DoCAN_Channel_send		

5.2.4 Typedef Documentation

5.2.4.1 AGHAF_DoCAN_Bus

`AGHAF_DoCAN_Bus`

Handle to a DoCAN bus.

5.2.4.2 AGHAF_DoCAN_Callback

`AGHAF_DoCAN_Callback`

Callback function type to recieve DoCAN events.

Parameters

event	Handle to the event.
userContext	Same parameter as the one passed to AGHAF_DoCAN_registerCallback().

Returns

void

Note

Do not delete *event* within the body of a callback function.

5.2.4.3 AGHAF_DoCAN_Channel

`AGHAF_DoCAN_Channel`

Handle to a DoCAN channel.

5.2.4.4 AGHAF_DoCAN_Event

`AGHAF_DoCAN_Event`

Handle to a DoCAN event.

5.2.4.5 AGHAF_DoCAN_Frame

`AGHAF_DoCAN_Frame`

Handle to a DoCAN frame.

5.2.5 Enumeration Type Documentation

5.2.5.1 AGHAF_DoCAN_AddrMode

`enum AGHAF_DoCAN_AddrMode`

Possible addressing modes for DoCAN.

Enumerator

AGHAF_DoCAN_ADDR_MODE_PHYS	Physical.
AGHAF_DoCAN_ADDR_MODE_FUNC	Functionnal.

5.2.5.2 AGHAF_DoCAN_ChannelDirection

`enum AGHAF_DoCAN_ChannelDirection`

Possible directions for DoCAN channels.

Enumerator

AGHAF_DoCAN_CHANNEL_DIR_RX	Rx only channel.
AGHAF_DoCAN_CHANNEL_DIR_TX	Tx only channel.
AGHAF_DoCAN_CHANNEL_DIR_2WAY	2-way channel

5.2.5.3 AGHAF_DoCAN_ChannelParam

`enum AGHAF_DoCAN_ChannelParam`

List of parameters used to configure a DoCAN channel.

Warning

The set of AGHAF_DoCAN_CHANNEL_PARAM_RESP_ID_xxx must be unique for all channels.
With the exception of functional addressing Tx channel, that don't need a response ID.

Enumerator

AGHAF_DoCAN_CHANNEL_PARAM_DIRECTION	Set channel direction. When functional addressing mode is used, you can't use 2-way channel. Use AGHAF_DoCAN_ChannelDirection Default: AGHAF_DoCAN_CHANNEL_DIR_2WAY
AGHAF_DoCAN_CHANNEL_PARAM_TXDL	Define TX_DL and identitify if CAN FD or Classic CAN is to be used. For Rx only channels, use any value different of AGHAF_DoCAN_TX_DL_CLASSIC_CAN to enable CAN FD. Use AGHAF_DoCAN_TxDL Default: AGHAF_DoCAN_TX_DL_CLASSIC_CAN
AGHAF_DoCAN_CHANNEL_PARAM_PAD_HANDLING	Enable padding handling. Use AGHAF_DoCAN_PadHandling Default: AGHAF_DoCAN_PAD_ENABLE
AGHAF_DoCAN_CHANNEL_PARAM_PAD_VALUE	Padding value to use when padding handling is enabled. This value is also used for mandatory padding of CAN FD frames (TX_DL $>$ 8). Range: [0x00, 0xFF] Default: 0xCC

Enumerator

AGHAF_DoCAN_CHANNEL_PARAM_ADDRMODE	Addressing mode used by the channel When functional adressing mode is used, you can't use 2-way channel. Use AGHAF_DoCAN_AddrMode Default: AGHAF_DoCAN_ADDR_MODE_PHYS
AGHAF_DoCAN_CHANNEL_PARAM_TIMING_AS	As - Time for transmission of the CAN frame (any N_PDU) on the sender side. Max: 20000 ms Default: 1000 ms
AGHAF_DoCAN_CHANNEL_PARAM_TIMING_AR	Ar - Time for transmission of the CAN frame (any N_PDU) on the receiver side. Max: 20000 ms Default: 1000 ms
AGHAF_DoCAN_CHANNEL_PARAM_TIMING_BS	Bs - Timeout until reception of the next FlowControl N_PDU. Max: 20000 ms Default: 1000 ms
AGHAF_DoCAN_CHANNEL_PARAM_TIMING_BR	Br - Performance requirement (not used). Default: NA
AGHAF_DoCAN_CHANNEL_PARAM_TIMING_CS	Cs - Performance requirement (not used). Default: NA
AGHAF_DoCAN_CHANNEL_PARAM_TIMING_CR	Cr - Timeout until reception of the next ConsecutiveFrame. Max: 20000 ms Default: 1000 ms
AGHAF_DoCAN_CHANNEL_PARAM_TIMING_STMIN	STmin - Separation time (Receiver only). The minimum time the sender shall wait between the transmissions of two ConsecutiveFrame N_PDUs The receiver transmit this value to the sender in FlowControl CTS STmin parameter. Range: [0x00, 0xFF] Default: 0x00
AGHAF_DoCAN_CHANNEL_PARAM_REQ_ID_FMT	CAN ID format used to transmit data. Use AGHAF_DoCAN_IdFormat Default: 0x00
AGHAF_DoCAN_CHANNEL_PARAM_REQ_ID_VALUE	CAN ID used to transmit data. Range: [0x00000000, 0x1FFFFFFF] Default: 0x7E0
AGHAF_DoCAN_CHANNEL_PARAM_REQ_ID_AE	Address extension used to transmit data. Range: [0x00,0xFF] Default: 0x00
AGHAF_DoCAN_CHANNEL_PARAM_RESP_ID_FMT	CAN ID format used to receive data. Use AGHAF_DoCAN_IdFormat Default: 0x00

Enumerator

AGHAF_DoCAN_CHANNEL_PARAM_RESP_ID_↩ VALUE	CAN ID used to receive data. Range: [0x00000000, 0x1FFFFFFF] Default: 0x7E8
AGHAF_DoCAN_CHANNEL_PARAM_RESP_ID_AE	Address extension used to receive data. Range: [0x00,0xFF] Default: 0x00
AGHAF_DoCAN_CHANNEL_PARAM_WAITFRAME	Maximum number of WAIT flow control frames allowed during a multi-segment transfer. Max: 1027 Default: 0
AGHAF_DoCAN_CHANNEL_PARAM_BLOCKSIZE	Block Size (Receiver only). Number of CF N_PDUs until transmission of the next FlowControl N_PDU. The receiver transmit this value to the sender in FlowControl CTS BlockSize parameter. Range: [0x00, 0xFF] Default: 0
AGHAF_DoCAN_CHANNEL_PARAM_MAXDATA↩ LEN	Maximum byte length of received segmented message (Receiver only). Max: 131072 Default: 4095
AGHAF_DoCAN_CHANNEL_PARAM_PAD_NOC↩ HECK	If enable, doesn't check padding of response against the AGHAF_DoCAN_CHANNEL_PARAM_PAD_H↩ ANDLING setting. Range: [0, 1] Default: 0 (disabled)
AGHAF_DoCAN_CHANNEL_PARAM_CANFD_BRS	Manage Bit Rate Switch when sending frame. Only evaluated in CAN FD mode (AGHAF_DoCAN_CHANNEL_PARAM_TXDL not equal to AGHAF_DoCAN_TX_DL_CLASSIC_CAN). Range: [0, 1] Default: 1 (enable BRS)
AGHAF_DoCAN_CHANNEL_PARAM_TIMING_S↩ TMIN_OVRD	STmin Override - Separation time (Sender only). The minimum time the sender is to wait between transmission of two CF N_PDUs. The FlowControl CTS STmin value reported by the vehicle should be ignored. Range: [0x00, 0xFF] and 0xFFFFFFFF Default: 0xFFFFFFFF Use the value reported by the vehicle

Enumerator

AGHAF_DoCAN_CHANNEL_PARAM_BS_OVRD	Block Size Override (Sender only). Number of CF N_PDUs until transmission of the next FlowControl N_PDU. The FlowControl CTS BlockSize value reported by the vehicle should be ignored. Range: [0x00, 0xFF] and 0xFFFFFFFF Default: 0xFFFFFFFF Use the value reported by the vehicle

5.2.5.4 AGHAF_DoCan_Error

`enum AGHAF_DoCan_Error`

Enumerator

AGHAF_DOCAN_ERROR_TO_A	As/Ar timeout (N_TIMEOUT_A) (Sender/Receiver)
AGHAF_DOCAN_ERROR_TO_BS	Bs timeout (N_TIMEOUT_Bs) (Sender only)
AGHAF_DOCAN_ERROR_TO_CR	Cr timeout (N_TIMEOUT_Cr) (Receiver only)
AGHAF_DOCAN_ERROR_SN	Unexpected SequenceNumber (N_WRONG_SN) (Receiver only)
AGHAF_DOCAN_ERROR_FS	Invalid FlowStatus in FC (N_INVALID_FS) (Sender only)
AGHAF_DOCAN_ERROR_PDU	Unexpected PDU (N_UNEXP_PDU) (Receiver only)
AGHAF_DOCAN_ERROR_WAIT	Wait overrun (N_WFT_OVRN) (Receiver only)
AGHAF_DOCAN_ERROR_OVFLW	Receiver can't store message (N_BUFFER_OVFLW) (Sender only)
AGHAF_DOCAN_ERROR_OTHER	Other error (N_ERROR) (Sender/Receiver)

5.2.5.5 AGHAF_DoCAN_ERROR

`enum AGHAF_DoCAN_ERROR`

Enumerator

AGHAF_DoCAN_ERROR_OK	Unknown type of error.
AGHAF_DoCAN_ERROR_SEQUENCE	No error.
AGHAF_DoCAN_ERROR_PARAM	Sequence error.
AGHAF_DoCAN_ERROR_UNIMPLEMENTED	Wrong parameters.
AGHAF_DoCAN_ERROR_FIFOFULL	Not implemented.
AGHAF_DoCAN_ERROR_BUSY	Fifo full.

5.2.5.6 AGHAF_DoCAN_EventType

enum AGHAF_DoCAN_EventType

Enumerator

AGHAF_DoCAN_EVENT_MSGTX	End of transmission.
AGHAF_DoCAN_EVENT_MSGTXERR	Transmission Error.
AGHAF_DoCAN_EVENT_MSGRX	Message has been received.
AGHAF_DoCAN_EVENT_MSGRXFF	First Frame has been received.
AGHAF_DoCAN_EVENT_MSGRXERR	Reception error.

5.2.5.7 AGHAF_DoCAN_IdFormat

enum AGHAF_DoCAN_IdFormat

Flags to configure DoCAN identifier format.

- Enable extended CAN identifer
- Enable address extension

Enumerator

AGHAF_DoCAN_ID_FMT_XTD	Extended identifier (29-bit CAN ID)
AGHAF_DoCAN_ID_FMT_AE	Use optional address extension It's the N_TA byte of Extended addressing format It's the N_AE byte of Mixed addressing format

5.2.5.8 AGHAF_DoCAN_PadHandling

enum AGHAF_DoCAN_PadHandling

Enable or disable padding handling.

Note

In CAN FD, even if padding is disabled, transmitted frames are still padded up to the nearest valid length.

Warning

When enabled, if response frames do not use padding they will be rejected.

Enumerator

AGHAF_DoCAN_PAD_DISABLE	Disable padding handling.
AGHAF_DoCAN_PAD_ENABLE	Enable padding handling.

5.2.5.9 AGHAF_DoCAN_TxDL

```
enum AGHAF_DoCAN_TxDL
```

Possible TX_DL for DoCAN transmissions.

Note

This parameter is also used to enable or disable the use of CAN FD.

Enumerator

AGHAF_DoCAN_TX_DL_CLASSIC_CAN	Classic CAN, TX_DL = 8.
AGHAF_DoCAN_TX_DL_8	CAN FD, TX_DL = 8.
AGHAF_DoCAN_TX_DL_12	CAN FD, TX_DL = 12.
AGHAF_DoCAN_TX_DL_16	CAN FD, TX_DL = 16.
AGHAF_DoCAN_TX_DL_20	CAN FD, TX_DL = 20.
AGHAF_DoCAN_TX_DL_24	CAN FD, TX_DL = 24.
AGHAF_DoCAN_TX_DL_32	CAN FD, TX_DL = 32.
AGHAF_DoCAN_TX_DL_48	CAN FD, TX_DL = 48.
AGHAF_DoCAN_TX_DL_64	CAN FD, TX_DL = 64.

5.2.6 Function Documentation

5.2.6.1 AGHAF_DoCAN_Channel_activateTrace()

```
void AGHAF_API AGHAF_DoCAN_Channel_activateTrace (
            AGHAF_DoCAN_Channel channel,
            AGHAF_BOOL value )
```

Activate the CAN trace.

Parameters

channel	handle on the channel
value	AGHAF_TRUE ? activate : deactivate

5.2.6.2 AGHAF_DoCAN_Channel_getParam()

```
AGHAF_Status AGHAF_API AGHAF_DoCAN_Channel_getParam (
            AGHAF_DoCAN_Channel channel,
            AGHAF_DoCAN_ChannelParam id,
            uint32_t * value )
```

Provide the value from a channel parameter.

Parameters

channel	handle on the channel
id	parameter to get
value	value to get

Returns

status request information

5.2.6.3 AGHAF_DoCAN_Channel_send()

```
AGHAF_Status AGHAF_API AGHAF_DoCAN_Channel_send (
            AGHAF_DoCAN_Channel channel,
            uint32_t dataLen,
            uint8_t const * data )
```

Send on the DoCAN channel in parameter.

Parameters

channel	handle on the channel
dataLen	length from the datas
data	pointer on data

Returns

status request information

Examples

DoCan_FD/main.cpp.

5.2.6.4 AGHAF_DoCAN_Channel_setParam()

```
AGHAF_Status AGHAF_API AGHAF_DoCAN_Channel_setParam (
            AGHAF_DoCAN_Channel channel,
            AGHAF_DoCAN_ChannelParam id,
            uint32_t value )
```

Set the parameter from a channel.

Parameters

channel	handle on the channel
id	parameter to set
value	value to set

Returns

status request information

Examples

DoCan_FD/main.cpp.

5.2.6.5 AGHAF_DoCAN_Channel_start()

```
AGHAF_Status AGHAF_API AGHAF_DoCAN_Channel_start (
            AGHAF_DoCAN_Channel channel )
```

Start a channel.

Parameters

channel	handle on the channel

Returns

status request information

Examples

DoCan_FD/main.cpp.

5.2.6.6 AGHAF_DoCAN_Channel_stop()

```
AGHAF_Status AGHAF_API AGHAF_DoCAN_Channel_stop (
            AGHAF_DoCAN_Channel channel )
```

Stop a channel.

Parameters

channel	handle on the channel

Returns

status request information

Examples

DoCan_FD/main.cpp.

5.2.6.7 AGHAF_DoCAN_createChannel()

```
AGHAF_DoCAN_Channel AGHAF_API AGHAF_DoCAN_createChannel (
            AGHAF_DoCAN_Bus bus )
```

Create a channel.

Parameters

bus	handle on the bus

Returns

handle on the channel created

Examples

DoCan_FD/main.cpp.

5.2.6.8 AGHAF_DoCAN_deregisterCallback()

```
void AGHAF_API AGHAF_DoCAN_deregisterCallback (
            AGHAF_DoCAN_Bus bus,
            AGHAF_DoCAN_Callback callback )
```

deregister a callback

Parameters

bus	handle on the bus
callback	function pointer to remove

Examples

DoCan_FD/main.cpp.

5.2.6.9 AGHAF_DoCAN_Event_getBusIndex()

```
uint8_t AGHAF_API AGHAF_DoCAN_Event_getBusIndex (
            AGHAF_DoCAN_Event event )
```

return the index from the bus on which occured the event

Parameters

event	handle on an event

Returns

index from bus

5.2.6.10 AGHAF_DoCAN_Event_getChannel()

```
AGHAF_DoCAN_Channel AGHAF_API AGHAF_DoCAN_Event_getChannel (
            AGHAF_DoCAN_Event event )
```

Return the channel on which occured the event.

Parameters

event	handle on the event

Returns

handle on the channel

Examples

DoCan_FD/main.cpp.

5.2.6.11 AGHAF_DoCAN_Event_getData()

```
AGHAF_Status AGHAF_API AGHAF_DoCAN_Event_getData (
            AGHAF_DoCAN_Event event,
            uint8_t * data,
            uint32_t * size )
```

Provide the datas received in the event.

Parameters

event	handle on the event
data	array to fill with the data
size	[in] data array size [out] size of copied data

Returns

status request information

Examples

DoCan_FD/main.cpp.

5.2.6.12 AGHAF_DoCAN_Event_getDataSize()

```
uint32_t AGHAF_API AGHAF_DoCAN_Event_getDataSize (
            AGHAF_DoCAN_Event event )
```

return the size from the datas in the event

Parameters

event	

Returns

size from the datas

Examples

DoCan_FD/main.cpp.

5.2.6.13 AGHAF_DoCAN_Event_getError()

```
uint8_t AGHAF_API AGHAF_DoCAN_Event_getError (
            AGHAF_DoCAN_Event event )
```

Return the error value from the event.

Parameters

event	handle on the event

Returns

corresponds to AGHAF_DoCan_Error enumerator

Examples

DoCan_FD/main.cpp.

5.2.6.14 AGHAF_DoCAN_Event_getId()

```
AGHAF_CAN_Identifier AGHAF_API AGHAF_DoCAN_Event_getId (
            AGHAF_DoCAN_Event event )
```

Return the CAN identifier from the event.

Parameters

event	handle on the event

Returns

CAN identifier from the event

5.2.6.15 AGHAF_DoCAN_Event_type()

```
AGHAF_DoCAN_EventType AGHAF_API AGHAF_DoCAN_Event_type (
            AGHAF_EventInfo event )
```

return the type from the event

Parameters

event	

Returns

type from the event

Examples

DoCan_FD/main.cpp.

5.2.6.16 AGHAF_DoCAN_getBus()

```
AGHAF_DoCAN_Bus AGHAF_API AGHAF_DoCAN_getBus (
            AGHAF_Device device,
            uint8_t index )
```

return the DoCAN bus at index in parameter

Parameters

device	device handle from the desired DoCAN bus
index	index from the bus

Returns

Handle on a DoCAN bus

Examples

DoCan_FD/main.cpp.

5.2.6.17 AGHAF_DoCAN_getBusCount()

```
uint32_t AGHAF_API AGHAF_DoCAN_getBusCount (
            AGHAF_Device device )
```

return the number of DoCAN bus from a device

Parameters

device	handle from the device

Returns

Number of DoCAN bus on the device

5.2.6.18 AGHAF_DoCAN_getChannelCount()

```
AGHAF_Status AGHAF_API AGHAF_DoCAN_getChannelCount (
            AGHAF_DoCAN_Bus bus,
            uint8_t * count )
```

Provide the number of channels available on the bus.

Parameters

bus	handle on the bus
count	pointer to fill with the data

Returns

status request information

5.2.6.19 AGHAF_DoCAN_registerCallback()

```
void AGHAF_API AGHAF_DoCAN_registerCallback (
            AGHAF_DoCAN_Bus bus,
            AGHAF_DoCAN_Callback callback,
            void * userContext )
```

Register a callback for the DoCAN events from a bus.

Parameters

bus	handle on the bus
callback	function pointer to call on an event
userContext	

Examples

DoCan_FD/main.cpp.

5.2.6.20 AGHAF_DoCAN_releaseChannel()

```
AGHAF_Status AGHAF_API AGHAF_DoCAN_releaseChannel (
            AGHAF_DoCAN_Channel channel )
```

Release the channel.

Parameters

channel	handle on the channel to release

Returns

status request information

Examples

DoCan_FD/main.cpp.

5.3 Ethernet

API to use ethernet on exxotest devices.

- AGHAF_EthBus AGHAF_ETH_getBus (AGHAF_Device device, uint8_t index)

  return the Ethernet bus at index in parameter
- uint8_t AGHAF_ETH_getBusIndex (AGHAF_EthBus bus)

  return the index from a Ethernet
- uint8_t AGHAF_ETH_getBusCount (AGHAF_Device device)

  return the number of CAN bus from a device
- void AGHAF_ETH_getFriendlyNameEthCard (AGHAF_EthBus ethernet, char ∗∗name)

  AGHAF_ETH_getFriendlyNameEthCard.
- void AGHAF_ETH_freeFriendlyName (char ∗friendlyName)

  Free the memory allocated with AGHAF_Device_getFriendlyNameEthCard.
- AGHAF_Status AGHAF_ETH_getMacAddress (AGHAF_EthBus bus, AGHAF_ETH_MacAddress ∗mac↩ Address)

  return the Ethernet mac address on device "device", and channel "index"
- AGHAF_Status **AGHAF_ETH_setPhy** (AGHAF_EthBus bus, AGHAF_ETH_Phy phy)
- AGHAF_Status **AGHAF_ETH_getPhy** (AGHAF_EthBus bus, AGHAF_ETH_Phy ∗phy)
- AGHAF_BOOL **AGHAF_ETH_getEthernetDiagLineState** (AGHAF_EthBus bus)
- AGHAF_Status AGHAF_ETH_setEthernetDiagLineState (AGHAF_EthBus bus, AGHAF_BOOL active)

  AGHAF_ETH_setEthernetDiagLineState.

- #define **AGHAF_ETH_ALEN** 6
- enum **AGHAF_ETH_Phy** { **AGHAF_ETH_InvalidPhy** = -1, **AGHAF_ETH_BroadRReach**, **AGHAF_ETH↩ _IEEE** }
- typedef void ∗ **AGHAF_EthBus**
- typedef struct AGHAF_ETH_MacAddress **AGHAF_ETH_MacAddress**

5.3.1 Detailed Description

API to use ethernet on exxotest devices.

5.3.2 Function Documentation

5.3.2.1 AGHAF_ETH_freeFriendlyName()

```
void AGHAF_ETH_freeFriendlyName (
            char * friendlyName )
```

Free the memory allocated with AGHAF_Device_getFriendlyNameEthCard.

Parameters

friendlyName	name to free

5.3.2.2 AGHAF_ETH_getBus()

```
AGHAF_EthBus AGHAF_ETH_getBus (
            AGHAF_Device device,
            uint8_t index )
```

return the Ethernet bus at index in parameter

Parameters

device	device handle from the desired Ethernet bus
index	index from the bus

Returns

Handle on a Ethernet bus

5.3.2.3 AGHAF_ETH_getBusCount()

```
uint8_t AGHAF_ETH_getBusCount (
            AGHAF_Device device )
```

return the number of CAN bus from a device

Parameters

device	handle from the device

Returns

Number of CAN bus on the device

5.3.2.4 AGHAF_ETH_getBusIndex()

```
uint8_t AGHAF_ETH_getBusIndex (
            AGHAF_EthBus bus )
```

return the index from a Ethernet

Parameters

bus	handle from the Ethernet

Returns

index from the bus

5.3.2.5 AGHAF_ETH_getFriendlyNameEthCard()

```
void AGHAF_ETH_getFriendlyNameEthCard (
            AGHAF_EthBus ethernet,
            char ** name )
```

AGHAF_ETH_getFriendlyNameEthCard.

Parameters

device	
name	

Returns

5.3.2.6 AGHAF_ETH_getMacAddress()

```
AGHAF_Status AGHAF_ETH_getMacAddress (
            AGHAF_EthBus bus,
            AGHAF_ETH_MacAddress * macAddress )
```

return the Ethernet mac address on device "device", and channel "index"

Parameters

bus	handle from the desired Ethernet bus
macAddress	string of size 13 of uint8_t

Returns

status request information

5.3.2.7 AGHAF_ETH_setEthernetDiagLineState()

```
AGHAF_Status AGHAF_ETH_setEthernetDiagLineState (
            AGHAF_EthBus bus,
            AGHAF_BOOL active )
```

AGHAF_ETH_setEthernetDiagLineState.

Parameters

bus	handle on the Ethernet bus
active	AGHAF_TRUE ? activate : deactivate

Returns

status request information

5.4 Global

API to interrogate AEDS about connected devices.

- enum AGHAF_BOOL { AGHAF_FALSE = 0, AGHAF_TRUE = 1 }
- enum AGHAF_Status {
  AGHAF_STATUS_OK = 0, AGHAF_ERROR_PARAM, AGHAF_ERROR_NOT_ENOUGH_MEMORY,
  AGHAF_ERROR_OUT_OF_MEMORY,
  AGHAF_ERROR_NOT_IMPLEMENTED, AGHAF_ERROR_END_REACHED, AGHAF_INTERNAL_ERROR,
  AGHAF_ERROR_CONNECTION,
  AGHAF_ERROR_BUS_NOT_FOUND, AGHAF_ERROR_PENDING, AGHAF_ERROR_WRONG_STATE,
  AGHAF_ERROR_NOT_ENOUGH_BUS,
  AGHAF_WRONG_SERVICE_VERSION, AGHAF_TIMEOUT_COM, AGHAF_DEPRECATED, AGHAF_ERROR_SEQUENCE,
  AGHAF_ERROR_FIFOFULL, AGHAF_ERROR_ALREADY_OPEN, AGHAF_ERROR_BUSY, AGHAF_ERROR_NOMORECH
  AGHAF_ERROR_UNKNOWN, AGHAF_ERROR_WRONG_SESSION_TYPE, AGHAF_ERROR_NO_DATA
  = 0xff }
- enum AGHAF_ConnectionMode { AGHAF_OFFLINE, AGHAF_USB, AGHAF_ETH }
- enum AGHAF_DeviceEvent { AGHAF_Device_noEvent, AGHAF_Device_Arrival, AGHAF_Device_Leave,
  AGHAF_Device_Present }
- enum AGHAF_TypeEvent {
  AGHAF_TypeNoEvent, AGHAF_TypeDeviceEvent, AGHAF_TypeEthernetEvent, AGHAF_TypeCanEvent,
  AGHAF_TypeDoCanEvent }
- enum AGHAF_DeviceState { AGHAF_DeviceState_error = 0, AGHAF_DeviceState_boot, AGHAF_DeviceState_ready
  }
- typedef int AGHAF_EventHandle
- typedef void * **AGHAF_Device**
- typedef void * **AGHAF_Event**
- typedef void(* **AGHAF_Callback**) (AGHAF_Event event, void *)
- typedef enum AGHAF_DeviceState **AGHAF_DeviceState**
- typedef struct AGHAF_DeviceInfo **AGHAF_DeviceInfo**
- typedef struct AGHAF_IpAddress AGHAF_IpAddress

  represent an IP address
- typedef struct AGHAF_EventInfo **AGHAF_EventInfo**
- uint32_t AGHAF_API AGHAF_getVersion (void)

  Return the version from Aghaf.
- const char *AGHAF_API AGHAF_getVersionString (void)

  Return the version from Aghaf.
- uint32_t AGHAF_API AGHAF_getServiceVersion (void)

  Return the version from aeds.
- AGHAF_BOOL AGHAF_API AGHAF_isServiceRunning (void)

  Inform if ghaston is running or not.
- char const *AGHAF_API AGHAF_DeviceEvent_getProductName (AGHAF_Event event)

  return the product name from the device which emitted the device event
- char const *AGHAF_API AGHAF_DeviceEvent_getProductNumber (AGHAF_Event event)

  return the product number from the device which emitted the device event
- char const *AGHAF_API AGHAF_DeviceEvent_getSerialNumber (AGHAF_Event event)

  return the serial number from the device which emitted the device event
- uint8_t const *AGHAF_API AGHAF_DeviceEvent_getHardwareUniqueId (AGHAF_Event event)

  return the hardware unique id from the device which emitted the device event
- uint32_t AGHAF_API AGHAF_getDeviceCount (void)

  Return the number of devices onnected.
- AGHAF_Status AGHAF_API AGHAF_getDeviceList (AGHAF_DeviceInfo **devices, uint32_t *size)

Provide the list of devices connected.

- void AGHAF_API AGHAF_freeDeviceList (AGHAF_DeviceInfo *devices)

  Free the memory allocated with AGHAF_getDeviceList.
- AGHAF_Status AGHAF_API AGHAF_refreshDeviceList (void)

  Call the device event callbacks with the devices connected.
- AGHAF_Device AGHAF_API AGHAF_getDeviceBySN (const char *productNumber, const char *serialNumber)

  Return the handle on the specified device.
- AGHAF_Device AGHAF_API AGHAF_getDeviceByHardID (uint8_t const *hardwareID)

  AGHAF_getDeviceByHardID.
- AGHAF_Status AGHAF_API AGHAF_getDeviceInfo (AGHAF_Device device, AGHAF_DeviceInfo *deviceInfo)

  Return the informations from a device.
- AGHAF_ConnectionMode AGHAF_API AGHAF_Device_getConnectionMode (AGHAF_Device device)

  AGHAF_Device_getConnectionMode.
- void AGHAF_API AGHAF_Device_getFriendlyNameEthCard (AGHAF_Device device, char **name)

  AGHAF_Device_getFriendlyNameEthCard.
- void AGHAF_API AGHAF_Device_freeFriendlyName (char *friendlyName)

  Free the memory allocated with AGHAF_Device_getFriendlyNameEthCard.
- AGHAF_Status AGHAF_API AGHAF_Device_getUsbInfo (AGHAF_Device device, uint16_t *vid, uint16_t *pid, uint16_t *rev)

  AGHAF_Device_getUsbInfo.
- AGHAF_Status AGHAF_API AGHAF_openDevice (AGHAF_Device device)

  Open a device.
- AGHAF_Status AGHAF_API AGHAF_closeDevice (AGHAF_Device device)

  close the device
- AGHAF_Status AGHAF_API AGHAF_Event_getInfo (AGHAF_Event event, AGHAF_EventInfo *info)

  AGHAF_Event_getInfo.
- void AGHAF_API AGHAF_registerCallback (AGHAF_Callback callback, void *userData)

  Register a callback for the device event.
- void AGHAF_API AGHAF_deregisterCallback (AGHAF_Callback callback)

  Deregister a callback.

5.4.1 Detailed Description

API to interrogate AEDS about connected devices.

Warning

A device whose Leave event happened must not be used anymore. In deed, the object is destroyed in the library and is no more usable.

5.4.2 Typedef Documentation

5.4.2.1 AGHAF_EventHandle

`AGHAF_EventHandle`

Typedef of a native event handle that can be used to wait on events.

5.4.3 Enumeration Type Documentation

5.4.3.1 AGHAF_BOOL

enum AGHAF_BOOL

Enumerator

AGHAF_FALSE	false
AGHAF_TRUE	true

5.4.3.2 AGHAF_ConnectionMode

enum AGHAF_ConnectionMode

Enumerator

AGHAF_OFFLINE	The device is disconnected.
AGHAF_USB	The device is connected using USB.
AGHAF_ETH	The device is connected over a network (Ethernet of WiFi)

5.4.3.3 AGHAF_DeviceEvent

enum AGHAF_DeviceEvent

Enumerator

AGHAF_Device_noEvent	No event. Event is invalid.
AGHAF_Device_Arrival	A device was connected to this PC.
AGHAF_Device_Leave	A device was disconnected from this PC.
AGHAF_Device_Present	A device was present.

5.4.3.4 AGHAF_DeviceState

enum AGHAF_DeviceState

Enumerator

AGHAF_DeviceState_error	No state. Information is invalid.
AGHAF_DeviceState_boot	Device is in boot state. no firmware detected.
AGHAF_DeviceState_ready	Device is ready to use.

5.4.3.5 AGHAF_Status

```
enum AGHAF_Status
```

Enumerator

AGHAF_STATUS_OK	The operation succeeded.
AGHAF_ERROR_PARAM	One or more parameters are invalid.
AGHAF_ERROR_NOT_ENOUGH_MEMORY	Not enough memory. Generally means that the output buffer of a function is too small to hold all the data.
AGHAF_ERROR_OUT_OF_MEMORY	This is an internal error meaning that the library or its underlying elements have run out of memory.
AGHAF_ERROR_NOT_IMPLEMENTED	The function is not yet implemented. This error probably means that you are using an unstable development version. Please contact software support to get the latest stable version.
AGHAF_ERROR_END_REACHED	There was not enough elements left and the end has been reached. This error happens when trying to access at least 1 element after the end, i.e. reading the last element does not trigger this error.
AGHAF_INTERNAL_ERROR	There was an error in internal object management.
AGHAF_ERROR_CONNECTION	This error happens when communication with service is lost.
AGHAF_ERROR_BUS_NOT_FOUND	This error happens when trying to access a bus or device with a invalid handle.
AGHAF_ERROR_PENDING	Internal use only.
AGHAF_ERROR_WRONG_STATE	This error happens when user call function in wrong state. see can_table docan_table.
AGHAF_ERROR_NOT_ENOUGH_BUS	This error happens when user try to access a wrong AGHAF_DoCAN_Bus.
AGHAF_WRONG_SERVICE_VERSION	This error happens when the service is not compatible with the version of this library.
AGHAF_TIMEOUT_COM	This error happens when service take too long time to response.
AGHAF_DEPRECATED	This error happens when a function is deprecated.
AGHAF_ERROR_SEQUENCE	sequence of function call is wrong
AGHAF_ERROR_FIFOFULL	intern fifo if full
AGHAF_ERROR_ALREADY_OPEN	device already opened
AGHAF_ERROR_BUSY	busy
AGHAF_ERROR_NOMORECHANNEL	no more do can channels are available
AGHAF_ERROR_UNKNOWN	unknown error, contact your provider
AGHAF_ERROR_WRONG_SESSION_TYPE	the type of the session does not match the function called
AGHAF_ERROR_NO_DATA	This error happens when a problem occur between this library and service.

5.4.3.6 AGHAF_TypeEvent

```
enum AGHAF_TypeEvent
```

Enumerator

AGHAF_TypeNoEvent	No event. Event is invalid.
AGHAF_TypeDeviceEvent	Event from Device.
AGHAF_TypeEthernetEvent	Event from Ethernet.
AGHAF_TypeCanEvent	Event from CAN.
AGHAF_TypeDoCanEvent	Event from DOCAN.

5.4.4 Function Documentation

5.4.4.1 AGHAF_closeDevice()

```
AGHAF_Status AGHAF_API AGHAF_closeDevice (
            AGHAF_Device device )
```

close the device

Parameters

device	handle on the device

Returns

status request information

Examples

Can_FD/main.cpp, Can_LS/main.cpp, and DoCan_FD/main.cpp.

5.4.4.2 AGHAF_deregisterCallback()

```
void AGHAF_API AGHAF_deregisterCallback (
            AGHAF_Callback callback )
```

Deregister a callback.

Parameters

callback	

Examples

DeviceManagement/main.cpp.

5.4.4.3 AGHAF_Device_freeFriendlyName()

```
void AGHAF_API AGHAF_Device_freeFriendlyName (
            char * friendlyName )
```

Free the memory allocated with AGHAF_Device_getFriendlyNameEthCard.

Parameters

friendlyName	name to free

5.4.4.4 AGHAF_Device_getConnectionMode()

```
AGHAF_ConnectionMode AGHAF_API AGHAF_Device_getConnectionMode (
            AGHAF_Device device )
```

AGHAF_Device_getConnectionMode.

Parameters

device	

Returns

5.4.4.5 AGHAF_Device_getFriendlyNameEthCard()

```
void AGHAF_API AGHAF_Device_getFriendlyNameEthCard (
            AGHAF_Device device,
            char ** name )
```

AGHAF_Device_getFriendlyNameEthCard.

Parameters

device	
name	

Returns

5.4.4.6 AGHAF_Device_getUsbInfo()

```
AGHAF_Status AGHAF_API AGHAF_Device_getUsbInfo (
            AGHAF_Device device,
            uint16_t * vid,
            uint16_t * pid,
            uint16_t * rev )
```

AGHAF_Device_getUsbInfo.

Parameters

device	
vid	
pid	
rev	

Returns

5.4.4.7 AGHAF_DeviceEvent_getHardwareUniqueId()

```
uint8_t const* AGHAF_API AGHAF_DeviceEvent_getHardwareUniqueId (
            AGHAF_Event event )
```

return the hardware unique id from the device which emitted the device event

Parameters

event	

Returns

hardware unique id

Examples

DeviceManagement/main.cpp.

5.4.4.8 AGHAF_DeviceEvent_getProductName()

```
char const* AGHAF_API AGHAF_DeviceEvent_getProductName (
            AGHAF_Event event )
```

return the product name from the device which emitted the device event

Parameters

event	

Returns

product name

Examples

DeviceManagement/main.cpp.

5.4.4.9 AGHAF_DeviceEvent_getProductNumber()

```
char const* AGHAF_API AGHAF_DeviceEvent_getProductNumber (
            AGHAF_Event event )
```

return the product number from the device which emitted the device event

Parameters

event	

Returns

product number

Examples

DeviceManagement/main.cpp.

5.4.4.10 AGHAF_DeviceEvent_getSerialNumber()

```
char const* AGHAF_API AGHAF_DeviceEvent_getSerialNumber (
            AGHAF_Event event )
```

return the serial number from the device which emitted the device event

Parameters

event	

Returns

serial number

Examples

DeviceManagement/main.cpp.

5.4.4.11 AGHAF_Event_getInfo()

```
AGHAF_Status AGHAF_API AGHAF_Event_getInfo (
            AGHAF_Event event,
            AGHAF_EventInfo * info )
```

AGHAF_Event_getInfo.

Parameters

event	
info	

Returns

Examples

Can_FD/main.cpp, Can_LS/main.cpp, DeviceManagement/main.cpp, and DoCan_FD/main.cpp.

5.4.4.12 AGHAF_freeDeviceList()

```
void AGHAF_API AGHAF_freeDeviceList (
            AGHAF_DeviceInfo * devices )
```

Free the memory allocated with AGHAF_getDeviceList.

Parameters

devices	array to free

Examples

Can_FD/main.cpp, Can_LS/main.cpp, and DoCan_FD/main.cpp.

5.4.4.13 AGHAF_getDeviceByHardID()

```
AGHAF_Device AGHAF_API AGHAF_getDeviceByHardID (
            uint8_t const * hardwareID )
```

AGHAF_getDeviceByHardID.

Parameters

hardwareID	

Returns

5.4.4.14 AGHAF_getDeviceBySN()

```
AGHAF_Device AGHAF_API AGHAF_getDeviceBySN (
            const char * productNumber,
            const char * serialNumber )
```

Return the handle on the specified device.

Parameters

productNumber	product number of the device
serialNumber	serial number of the device

Returns

handle on the device

Examples

Can_FD/main.cpp, Can_LS/main.cpp, DeviceManagement/main.cpp, and DoCan_FD/main.cpp.

5.4.4.15 AGHAF_getDeviceCount()

```
uint32_t AGHAF_API AGHAF_getDeviceCount ( )
```

Return the number of devices onnected.

Returns

number of devices connected

Examples

Can_FD/main.cpp, Can_LS/main.cpp, and DoCan_FD/main.cpp.

5.4.4.16 AGHAF_getDeviceInfo()

```
AGHAF_Status AGHAF_API AGHAF_getDeviceInfo (
            AGHAF_Device device,
            AGHAF_DeviceInfo * deviceInfo )
```

Return the informations from a device.

Parameters

device	handle on the device
deviceInfo	pointer to fill with the data

Returns

status request information

5.4.4.17 AGHAF_getDeviceList()

```
AGHAF_Status AGHAF_API AGHAF_getDeviceList (
            AGHAF_DeviceInfo ** devices,
            uint32_t * size )
```

Provide the list of devices connected.

Parameters

devices	pointer to fill with devices
size	number of devices in devices

Returns

status request information

Warning

the memory allocated in devices must freed with AGHAF_freeDeviceList

Examples

Can_FD/main.cpp, Can_LS/main.cpp, and DoCan_FD/main.cpp.

5.4.4.18 AGHAF_getServiceVersion()

```
uint32_t AGHAF_API AGHAF_getServiceVersion ( )
```

Return the version from aeds.

Returns

AEDS's version

5.4.4.19 AGHAF_getVersion()

```
uint32_t AGHAF_API AGHAF_getVersion ( )
```

Return the version from Aghaf.

Returns

Aghaf version

5.4.4.20 AGHAF_getVersionString()

```
const char* AGHAF_API AGHAF_getVersionString ( )
```

Return the version from Aghaf.

Returns

Aghaf version

5.4.4.21 AGHAF_isServiceRunning()

```
AGHAF_BOOL AGHAF_API AGHAF_isServiceRunning ( )
```

Inform if ghaston is running or not.

Returns

AGHAF_TRUE ? ghaston is running : ghaston is not running

5.4.4.22 AGHAF_openDevice()

```
AGHAF_Status AGHAF_API AGHAF_openDevice (
            AGHAF_Device device )
```

Open a device.

Parameters

device	handle on the device

Returns

status request information

Examples

Can_FD/main.cpp, Can_LS/main.cpp, and DoCan_FD/main.cpp.

5.4.4.23 AGHAF_refreshDeviceList()

```
AGHAF_Status AGHAF_API AGHAF_refreshDeviceList ( )
```

Call the device event callbacks with the devices connected.

Returns

status request information

5.4.4.24 AGHAF_registerCallback()

```
void AGHAF_API AGHAF_registerCallback (
            AGHAF_Callback callback,
            void * userData )
```

Register a callback for the device event.

Parameters

callback	
userData	

Examples

DeviceManagement/main.cpp.

Chapter 6

Class Documentation

6.1 AGHAF_CAN_FilterDesc Struct Reference

Public Attributes

- uint16_t **caps**
- uint16_t **count**

6.2 AGHAF_CAN_Identifier Struct Reference

Public Attributes

- union {
  uint16_t std: 11
  Standard identifier.
  uint32_t xtd: 29
  Extended identifier.
  } id

  Identifier.
- AGHAF_CAN_IdentifierType type

  Identifier type.

6.2.1 Detailed Description

Examples

Can_FD/main.cpp, and Can_LS/main.cpp.

6.3 AGHAF_CAN_Limit Struct Reference

Public Attributes

- uint16_t max

  limit max.
- uint16_t min

  limit min.

6.4 AGHAF_DeviceInfo Struct Reference

Public Attributes

- uint16_t bootVersion

  Boot firmware version. Value was coded in BCD. sample : 0x0113 : V1.13.
- AGHAF_DeviceState deviceState

  State of device.
- uint16_t firmwareVersion

  Firmware version. Value was coded in BCD. sample : 0x0224 : V2.24.
- uint8_t hardwareUniqueID [16]

  unique hardware id of the device.
- char hardwareVersion [64]

  Hardware version. in ASCII format.
- char name [64]

  Device name.
- char productNo [64]

  Product number (PPF).
- char serialNo [64]

  Serial number.

6.4.1 Detailed Description

Examples

Can_FD/main.cpp, Can_LS/main.cpp, and DoCan_FD/main.cpp.

6.5 AGHAF_ETH_MacAddress Struct Reference

Public Attributes

- uint8_t **data** [6]

6.6 AGHAF_EventInfo Struct Reference

Public Attributes

- AGHAF_BOOL absolute

  Define if the timestamp are absolute or relative.
- uint32_t event

  Event information. Provide information about event type (event tx, event rx, device connected...).
- uint64_t timestamp

  Timestamp at which the event happened.
- uint32_t timestampPrecision

  Precision of timestamp.
- AGHAF_TypeEvent type

  Type information. Provide information about origin (device, can, ethernet...).

6.6.1 Detailed Description

Examples

Can_FD/main.cpp, Can_LS/main.cpp, DeviceManagement/main.cpp, and DoCan_FD/main.cpp.

6.7 AGHAF_IpAddress Struct Reference

represent an IP address

`#include <aghaf_global.h>`

Public Attributes

- union {
  uint32_t ipv4Addr
  IP v4 address.
  uint16_t **ipv6Address** [8]
  } **addr**

- uint8_t version

  version

6.7.1 Detailed Description

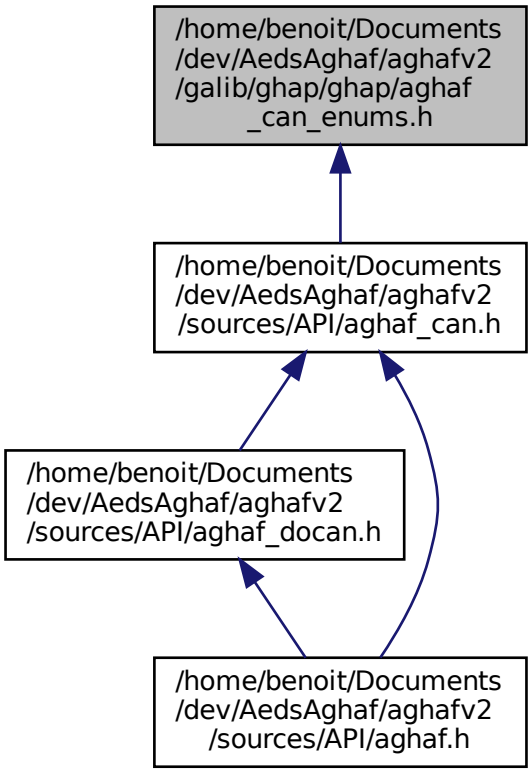
represent an IP address

Chapter 7

File Documentation

7.1 /home/benoit/Documents/dev/Aeds↩ Aghaf/aghafv2/galib/ghap/ghap/aghaf_can_enums.h File Reference

This graph shows which files directly or indirectly include this file:



- enum AGHAF_CAN_Error { AGHAF_CAN_ERROR_NONE = 0, AGHAF_CAN_ERROR_STUFF, AGHAF_CAN_ERROR_FORM, AGHAF_CAN_ERROR_ACK, AGHAF_CAN_ERROR_BIT1, AGHAF_CAN_ERROR_BIT0, AGHAF_CAN_ERROR_CRC }
- enum AGHAF_CAN_Mode { AGHAF_CAN_MODE_FD = 0x00, AGHAF_CAN_MODE_HS = 0x01, AGHAF_CAN_MODE_LS = 0x02 }

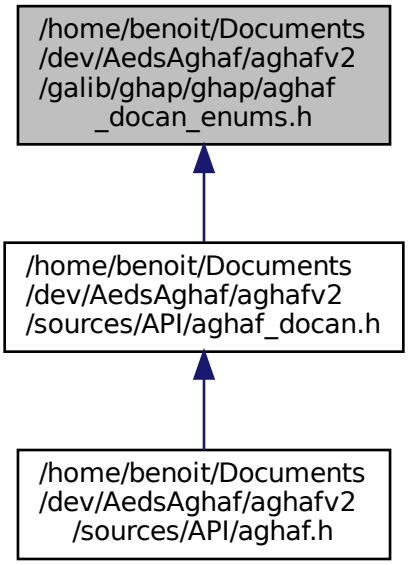
  Possible modes for CAN.
- enum AGHAF_CAN_Param { CAN_PARAM_MODE, CAN_PARAM_BAUDRATE, CAN_PARAM_BAUDRATE_FD, CAN_PARAM_SAMPLEPOINT, CAN_PARAM_SAMPLEPOINT_FD, CAN_PARAM_SYNCJUMP_WIDTH, CAN_PARAM_SYNCJUMP_WIDTH_FD, CAN_PARAM_LISTEN_ONLY, CAN_PARAM_TERMINATION, CAN_PARAM_TERMINATION_LS, CAN_PARAM_NO_EVENT }

  List of parameters used to configure a CAN bus.
- #define AGHAF_CAN_PERIODIC_NO_RESTART (1<<0)

  Don't restart period.

7.2 /home/benoit/Documents/dev/Aeds↩ Aghaf/aghafv2/galib/ghap/ghap/aghaf_docan_enums.h File Reference

This graph shows which files directly or indirectly include this file:



Enumerations

- enum AGHAF_DoCAN_AddrMode { AGHAF_DoCAN_ADDR_MODE_PHYS = 0x00, AGHAF_DoCAN_ADDR_MODE_FUNC = 0x01 }

  Possible addressing modes for DoCAN.

- enum AGHAF_DoCAN_ChannelDirection { AGHAF_DoCAN_CHANNEL_DIR_RX = (1U<<0), AGHAF_DoCAN_CHANNEL_D = (1U<<1), AGHAF_DoCAN_CHANNEL_DIR_2WAY = (AGHAF_DoCAN_CHANNEL_DIR_RX | AGHAF↩ _DoCAN_CHANNEL_DIR_TX) }

  Possible directions for DoCAN channels.

- enum AGHAF_DoCAN_ChannelParam { AGHAF_DoCAN_CHANNEL_PARAM_DIRECTION = 0, AGHAF_DoCAN_CHANNEL_PARAM_TXDL, AGHAF_DoCAN_CHANNEL_PARAM_PAD_HANDLING, AGHAF_DoCAN_CHANNEL_PARAM_PAD_VALUE, AGHAF_DoCAN_CHANNEL_PARAM_ADDRMODE, AGHAF_DoCAN_CHANNEL_PARAM_TIMING_AS, AGHAF_DoCAN_CHANNEL_PARAM_TIMING_AR, AGHAF_DoCAN_CHANNEL_PARAM_TIMING_BS, AGHAF_DoCAN_CHANNEL_PARAM_TIMING_BR, AGHAF_DoCAN_CHANNEL_PARAM_TIMING_CS, AGHAF_DoCAN_CHANNEL_PARAM_TIMING_CR, AGHAF_DoCAN_CHANNEL_PARAM_TIMING_STMIN, AGHAF_DoCAN_CHANNEL_PARAM_REQ_ID_FMT, AGHAF_DoCAN_CHANNEL_PARAM_REQ_ID_VALUE, AGHAF_DoCAN_CHANNEL_PARAM_REQ_ID_AE, AGHAF_DoCAN_CHANNEL_PARAM_RESP_ID_FMT, AGHAF_DoCAN_CHANNEL_PARAM_RESP_ID_VALUE, AGHAF_DoCAN_CHANNEL_PARAM_RESP_ID_AE, AGHAF_DoCAN_CHANNEL_PARAM_WAITFRAME, AGHAF_DoCAN_CHANNEL_PARAM_BLOCKSIZE, AGHAF_DoCAN_CHANNEL_PARAM_MAXDATALEN, AGHAF_DoCAN_CHANNEL_PARAM_PAD_NOCHECK, AGHAF_DoCAN_CHANNEL_PARAM_CANFD_BRS, AGHAF_DoCAN_CHANNEL_PARAM_TIMING_STMIN_OVRD, AGHAF_DoCAN_CHANNEL_PARAM_BS_OVRD }

  List of parameters used to configure a DoCAN channel.

- enum AGHAF_DoCan_Error { AGHAF_DOCAN_ERROR_TO_A = 0, AGHAF_DOCAN_ERROR_TO_BS = 1, AGHAF_DOCAN_ERROR_TO_CR = 2, AGHAF_DOCAN_ERROR_SN = 3, AGHAF_DOCAN_ERROR_FS = 4, AGHAF_DOCAN_ERROR_PDU = 5, AGHAF_DOCAN_ERROR_WAIT = 6, AGHAF_DOCAN_ERROR_OVFLW = 7, AGHAF_DOCAN_ERROR_OTHER = 8 }
- enum AGHAF_DoCAN_IdFormat { AGHAF_DoCAN_ID_FMT_XTD = (1U<<0), AGHAF_DoCAN_ID_FMT_AE = (1U<<1) }

  Flags to configure DoCAN identifier format.

- enum AGHAF_DoCAN_PadHandling { AGHAF_DoCAN_PAD_DISABLE = 0x00, AGHAF_DoCAN_PAD_ENABLE = 0x01 }

  Enable or disable padding handling.

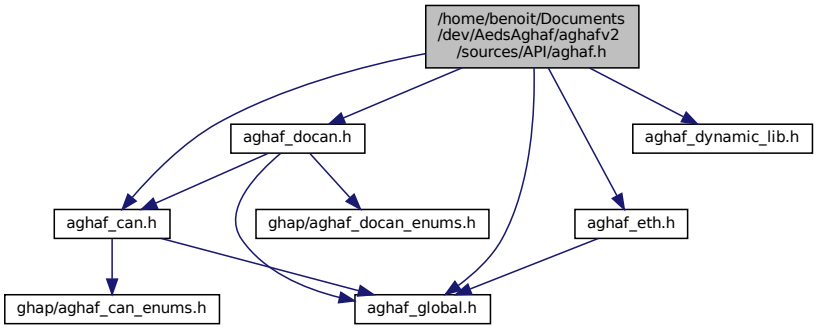
- enum AGHAF_DoCAN_TxDL { AGHAF_DoCAN_TX_DL_CLASSIC_CAN = 0, AGHAF_DoCAN_TX_DL_8 = 8, AGHAF_DoCAN_TX_DL_12 = 9, AGHAF_DoCAN_TX_DL_16 = 10, AGHAF_DoCAN_TX_DL_20 = 11, AGHAF_DoCAN_TX_DL_24 = 12, AGHAF_DoCAN_TX_DL_32 = 13, AGHAF_DoCAN_TX_DL_48 = 14, AGHAF_DoCAN_TX_DL_64 = 15 }

  Possible TX_DL for DoCAN transmissions.

7.3 /home/benoit/Documents/dev/AedsAghaf/aghafv2/sources/API/aghaf.h File Reference

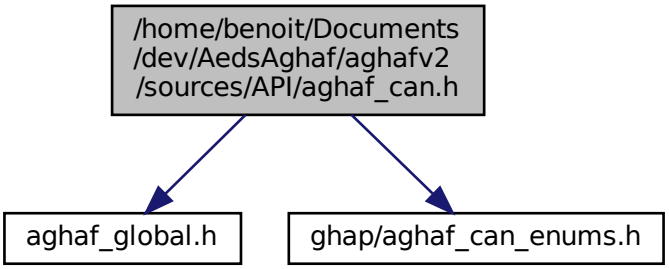
```
#include "aghaf_global.h"
#include "aghaf_can.h"
#include "aghaf_docan.h"
#include "aghaf_eth.h"
```

```
#include "aghaf_dynamic_lib.h"
```
Include dependency graph for aghaf.h:

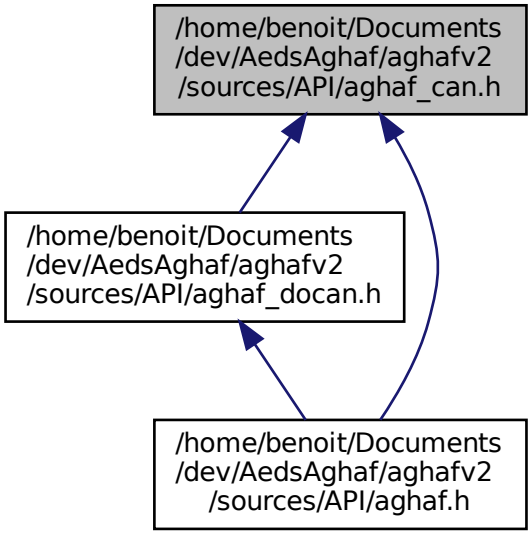


7.4 /home/benoit/Documents/dev/AedsAghaf/aghafv2/sources/API/aghaf_can.h File Reference

```
#include "aghaf_global.h"
#include <ghap/aghaf_can_enums.h>
```
Include dependency graph for aghaf_can.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct AGHAF_CAN_FilterDesc
- struct AGHAF_CAN_Identifier
- struct AGHAF_CAN_Limit

- void AGHAF_API AGHAF_CAN_asyncSendFrame (AGHAF_CAN_Bus bus, AGHAF_CAN_Frame frame, uint32_t *id)

  AGHAF_CAN_asyncSendFrame.
- typedef void(* **AGHAF_CAN_asyncSendFrame_Callback**) (AGHAF_Status, uint32_t, void *)
- void AGHAF_API AGHAF_CAN_asyncSendFrame_deregisterCallback (AGHAF_CAN_Bus bus, AGHAF_↩ CAN_asyncSendFrame_Callback callback)

  AGHAF_CAN_asyncSendFrame_deregisterCallback.
- void AGHAF_API AGHAF_CAN_asyncSendFrame_registerCallback (AGHAF_CAN_Bus bus, AGHAF_CA↩ N_asyncSendFrame_Callback callback, void *userContext)

  AGHAF_CAN_asyncSendFrame_registerCallback.
- typedef void * AGHAF_CAN_Bus
- AGHAF_Status AGHAF_API AGHAF_CAN_Bus_activate (AGHAF_CAN_Bus bus)

  activate a CAN bus
- AGHAF_Status AGHAF_API AGHAF_CAN_Bus_addFilter (AGHAF_CAN_Bus bus, AGHAF_CAN_FilterId idType, AGHAF_CAN_FILTER_Type type, AGHAF_CAN_FilterMode mode, uint32_t id1, uint32_t id2)

  add a filter to the CAN bus
- AGHAF_Status AGHAF_API AGHAF_CAN_Bus_BusOn (AGHAF_CAN_Bus bus)

  reactivate the bus after an error
- AGHAF_Status AGHAF_API AGHAF_CAN_Bus_clearFilters (AGHAF_CAN_Bus bus)

  clear the filters from a bus
- AGHAF_Status AGHAF_API AGHAF_CAN_Bus_deactivate (AGHAF_CAN_Bus bus)

deactivate a CAN bus

- AGHAF_Status AGHAF_API AGHAF_CAN_Bus_filtersCount (AGHAF_CAN_Bus bus, AGHAF_CAN_FilterId idType, uint16_t *count)

  Provide the current number of filters.
- void AGHAF_API AGHAF_CAN_Bus_freeSupportedFilters (AGHAF_CAN_FilterDesc *values)

  Free the memory allocated with AGHAF_CAN_Bus_supportedFilters.
- void AGHAF_API AGHAF_CAN_Bus_freeSupportedModes (AGHAF_CAN_Mode *modes)

  Free the memory allocated with AGHAF_CAN_Bus_supportedModes.
- void AGHAF_API AGHAF_CAN_Bus_freeSupportedTerminations (uint32_t *terminations)

  Free the memory allocated with AGHAF_CAN_Bus_supportedTerminations or AGHAF_CAN_Bus_supported↩ TerminationsLs.
- AGHAF_Status AGHAF_API AGHAF_CAN_Bus_getParam (AGHAF_CAN_Bus bus, AGHAF_CAN_Param param, uint32_t *value)

  Provide the value from a CAN parameter.
- uint8_t AGHAF_API AGHAF_CAN_Bus_getPeriodicCount (AGHAF_CAN_Bus bus)

  Return the number of periodic message available on a CAN bus.
- AGHAF_Status AGHAF_API AGHAF_CAN_Bus_isActivated (AGHAF_CAN_Bus bus, AGHAF_BOOL *is↩ Activated)

  Request if a CAN bus is yet activated or not.
- AGHAF_Status AGHAF_API AGHAF_CAN_Bus_reject (AGHAF_CAN_Bus bus, AGHAF_CAN_RejectType rejectType, AGHAF_BOOL rtr)

  Set a global filter.
- AGHAF_Status AGHAF_API AGHAF_CAN_Bus_sendPeriodic (AGHAF_CAN_Bus bus, AGHAF_CAN_Frame frame, uint8_t index, uint8_t flags, uint32_t period)

  Add a periodic message to send.
- AGHAF_Status AGHAF_API AGHAF_CAN_Bus_setParam (AGHAF_CAN_Bus bus, AGHAF_CAN_Param param, uint32_t value)

  Set the value from a CAN parameter.
- AGHAF_Status AGHAF_API AGHAF_CAN_Bus_state (AGHAF_CAN_Bus bus, AGHAF_CAN_BusState *canBusState)

  AGHAF_CAN_Bus_state.
- AGHAF_Status AGHAF_API AGHAF_CAN_Bus_supportedFilters (AGHAF_CAN_Bus bus, uint16_↩ t *globalFilterCaps, AGHAF_CAN_FilterDesc **values, uint16_t *size)

  Provide the supported filters from a CAN bus.
- AGHAF_Status AGHAF_API AGHAF_CAN_Bus_supportedModes (AGHAF_CAN_Bus bus, AGHAF_CAN_Mode **modes, uint8_t *nbModes)

  Request the supported modes from the CAN bus in parameter.
- AGHAF_Status AGHAF_API AGHAF_CAN_Bus_supportedTerminations (AGHAF_CAN_Bus bus, uint32_t **terminations, uint32_t *nbTerminations)

  Request the supported terminations from a CAN bus.
- AGHAF_Status AGHAF_API AGHAF_CAN_Bus_supportedTerminationsLs (AGHAF_CAN_Bus bus, uint32_t **terminations, uint32_t *nbTerminations)

  Request the supported low speed terminations from a CAN bus.
- enum AGHAF_CAN_BusState { AGHAF_CAN_BUS_UNKNOW = 0, AGHAF_CAN_BUS_ACTIVE, AGHAF_CAN_BUS_PASSIVE, AGHAF_CAN_BUS_BUSOFF }
- typedef void(* AGHAF_CAN_Callback) (AGHAF_CAN_Event, void *)
- void AGHAF_API AGHAF_CAN_deregisterCallback (AGHAF_CAN_Bus bus, AGHAF_CAN_Callback callback)

  deregister a callback
- typedef void * AGHAF_CAN_Event
- uint32_t AGHAF_API AGHAF_CAN_Event_busError (AGHAF_CAN_Event_busInfo busState)

  return the value from the error
- typedef void const * AGHAF_CAN_Event_busInfo

- uint32_t AGHAF_API AGHAF_CAN_Event_busLoad (AGHAF_CAN_Event_busInfo busState)

  return the bus load
- uint32_t AGHAF_API AGHAF_CAN_Event_chipState (AGHAF_CAN_Event_busInfo busState)

  return the chipstate
- AGHAF_CAN_Frame AGHAF_API AGHAF_CAN_Event_frame (AGHAF_CAN_Event event)

  return the frame contained in an event
- enum AGHAF_CAN_EventType { AGHAF_CAN_EVENT_UNKNOW = 0, AGHAF_CAN_EVENT_MSGTX = 1, AGHAF_CAN_EVENT_MSGRX = 2, AGHAF_CAN_EVENT_ERROR = 3, AGHAF_CAN_EVENT_BUSCHANGE = 4, AGHAF_CAN_EVENT_BUSLOAD = 5 }
- enum AGHAF_CAN_FILTER_Type { AGHAF_CAN_FILTER_TYPE_CLASSIC = 0, AGHAF_CAN_FILTER_TYPE_SINGLE = 1, AGHAF_CAN_FILTER_TYPE_DUAL = 2, AGHAF_CAN_FILTER_TYPE_RANGE = 3 }
- typedef struct AGHAF_CAN_FilterDesc **AGHAF_CAN_FilterDesc**
- enum AGHAF_CAN_FilterId { AGHAF_CAN_FILTER_ID_STD = 0, AGHAF_CAN_FILTER_ID_XTD = 1 }
- enum AGHAF_CAN_FilterMode { AGHAF_CAN_FILTER_MODE_ACCEPT = 0, AGHAF_CAN_FILTER_MODE_REJECT = 1 }
- typedef void * AGHAF_CAN_Frame
- AGHAF_BOOL AGHAF_API AGHAF_CAN_Frame_brs (AGHAF_CAN_Frame frame)

  AGHAF_CAN_Frame_brs.
- AGHAF_Status AGHAF_API AGHAF_CAN_Frame_data (AGHAF_CAN_Frame frame, void *data, int64_t size)

  Provide the data contained in a frame.
- int64_t AGHAF_API AGHAF_CAN_Frame_dataSize (AGHAF_CAN_Frame frame)

  return the size from the data
- void AGHAF_API AGHAF_CAN_Frame_delete (AGHAF_CAN_Frame frame)

  Free the memory from a CAN frame.
- AGHAF_BOOL AGHAF_API AGHAF_CAN_Frame_esi (AGHAF_CAN_Frame frame)

  AGHAF_CAN_Frame_esi.
- AGHAF_BOOL AGHAF_API AGHAF_CAN_Frame_fdf (AGHAF_CAN_Frame frame)

  AGHAF_CAN_Frame_fdf.
- AGHAF_CAN_FrameType AGHAF_API AGHAF_CAN_Frame_frameType (AGHAF_CAN_Frame frame)

  Get the type from the frame.
- AGHAF_Status AGHAF_API AGHAF_CAN_Frame_id (AGHAF_CAN_Frame frame, AGHAF_CAN_Identifier *identifier)

  Get the identifier from a CAN frame.
- AGHAF_CAN_Frame AGHAF_API AGHAF_CAN_Frame_new (void)

  Return a handle on a CAN frame.
- AGHAF_Status AGHAF_API AGHAF_CAN_Frame_setData (AGHAF_CAN_Frame frame, const void *data, int64_t size)

  Set the data from a frame.
- AGHAF_Status AGHAF_API AGHAF_CAN_Frame_setFDF (AGHAF_CAN_Frame frame, AGHAF_BOOL fdf, AGHAF_BOOL esi, AGHAF_BOOL brs)

  AGHAF_CAN_Frame_setFDF.
- AGHAF_Status AGHAF_API AGHAF_CAN_Frame_setFrameType (AGHAF_CAN_Frame frame, AGHAF_CAN_FrameType type)

  Set the type from the frame.
- AGHAF_Status AGHAF_API AGHAF_CAN_Frame_setId (AGHAF_CAN_Frame frame, AGHAF_CAN_Identifier *identifier)

  Set the identifier from a CAN frame.
- enum AGHAF_CAN_FrameType { AGHAF_CAN_DataFrame = 0, AGHAF_CAN_RemoteFrame = 1 }
- AGHAF_CAN_Bus AGHAF_API AGHAF_CAN_getBus (AGHAF_Device device, uint8_t index)

  return the CAN bus at index in parameter
- uint8_t AGHAF_API AGHAF_CAN_getBusCount (AGHAF_Device device)

return the number of CAN bus from a device

- uint8_t AGHAF_API AGHAF_CAN_getBusIndex (AGHAF_CAN_Bus bus)

  return the index from a CAN bus
- uint8_t AGHAF_API AGHAF_CAN_getEventBusIndex (AGHAF_CAN_Event event)

  return the index from the bus on which occured the event
- AGHAF_CAN_Event_busInfo AGHAF_API AGHAF_CAN_getEventBusInfo (AGHAF_CAN_Event event)

  return the bus informations from an event
- typedef struct AGHAF_CAN_Identifier **AGHAF_CAN_Identifier**
- enum AGHAF_CAN_IdentifierType { AGHAF_CAN_IdStd, AGHAF_CAN_IdXtd }
- typedef struct AGHAF_CAN_Limit **AGHAF_CAN_Limit**
- #define AGHAF_CAN_PERIODIC_NoRestart (1<<0)

  Don't restart period.
- void AGHAF_API AGHAF_CAN_registerCallback (AGHAF_CAN_Bus bus, AGHAF_CAN_Callback callback, void *userContext)

  Register a callback for the CAN events from a bus.
- enum AGHAF_CAN_RejectType { AGHAF_CAN_REJECT_ALL = 0, AGHAF_CAN_REJECT_STD = 1, AGHAF_CAN_REJECT_EXT = 2, AGHAF_CAN_REJECT_NONE = 3 }
- AGHAF_Status AGHAF_API AGHAF_CAN_sendFrame (AGHAF_CAN_Bus bus, AGHAF_CAN_Frame frame)

  send a frame on a CAN bus
- AGHAF_CAN_EventType AGHAF_API AGHAF_CAN_typeEvent (AGHAF_EventInfo event)

  return the type from the event

7.4.1 Typedef Documentation

7.4.1.1 AGHAF_CAN_Callback

`AGHAF_CAN_Callback`

Callback function type to recieve CAN events.

Parameters

event	Handle to the event.
userContext	Same parameter as the one passed to AGHAF_CAN_registerCallback().

Returns

void

Note

Do not delete *event* within the body of a callback function.

7.4.1.2 AGHAF_CAN_Event

`AGHAF_CAN_Event`

Handle to a CAN event.

7.4.1.3 AGHAF_CAN_Event_busInfo

`AGHAF_CAN_Event_busInfo`

Handle to a CAN event bus info

7.4.1.4 AGHAF_CAN_Frame

`AGHAF_CAN_Frame`

Handle to a CAN frame.

7.4.2 Enumeration Type Documentation

7.4.2.1 AGHAF_CAN_BusState

`enum AGHAF_CAN_BusState`

Enumerator

AGHAF_CAN_BUS_UNKNOW	Invalide mode.
AGHAF_CAN_BUS_ACTIVE	active error state
AGHAF_CAN_BUS_PASSIVE	Passive error state.
AGHAF_CAN_BUS_BUSOFF	Bus off state.

7.4.2.2 AGHAF_CAN_EventType

`enum AGHAF_CAN_EventType`

Enumerator

AGHAF_CAN_EVENT_UNKNOW	Unknown type of event.
AGHAF_CAN_EVENT_MSGTX	End of transmission.
AGHAF_CAN_EVENT_MSGRX	Message has been received.
AGHAF_CAN_EVENT_ERROR	Bus error.
AGHAF_CAN_EVENT_BUSCHANGE	Chip state has chaged active or passive error ou bus off.
AGHAF_CAN_EVENT_BUSLOAD	Bus load information.

7.4.2.3 AGHAF_CAN_FILTER_Type

enum AGHAF_CAN_FILTER_Type

Enumerator

AGHAF_CAN_FILTER_TYPE_CLASSIC	id1 is the identifier and id2 is the mask
AGHAF_CAN_FILTER_TYPE_SINGLE	id1 is the identifier you want to filter. id2 is not used.
AGHAF_CAN_FILTER_TYPE_DUAL	id1 is an identifier and id2 is an identifier too
AGHAF_CAN_FILTER_TYPE_RANGE	allow to filter on a range (id1 is the beginning and id2 is the end)

7.4.2.4 AGHAF_CAN_FilterId

enum AGHAF_CAN_FilterId

Enumerator

AGHAF_CAN_FILTER_ID_STD	Filter for frames with standard can Id.
AGHAF_CAN_FILTER_ID_XTD	filter for frames with an extended can Id

7.4.2.5 AGHAF_CAN_FilterMode

enum AGHAF_CAN_FilterMode

Enumerator

AGHAF_CAN_FILTER_MODE_ACCEPT	Accept mode for a can filter.
AGHAF_CAN_FILTER_MODE_REJECT	Reject mode for a can filter.

7.4.2.6 AGHAF_CAN_FrameType

enum AGHAF_CAN_FrameType

Enumerator

AGHAF_CAN_DataFrame	standard frame of data
AGHAF_CAN_RemoteFrame	a remote frame means we send only the id and the data field is completed by another device on the bus

7.4.2.7 AGHAF_CAN_RejectType

```
enum AGHAF_CAN_RejectType
```

Enumerator

AGHAF_CAN_REJECT_ALL	Reject all non matching frames.
AGHAF_CAN_REJECT_STD	Reject all non matching frames with a standard can Id.
AGHAF_CAN_REJECT_EXT	Reject all non matching frames with an extended can Id.
AGHAF_CAN_REJECT_NONE	Accept all the frames.

7.4.3 Function Documentation

7.4.3.1 AGHAF_CAN_asyncSendFrame()

```
void AGHAF_API AGHAF_CAN_asyncSendFrame (
            AGHAF_CAN_Bus bus,
            AGHAF_CAN_Frame frame,
            uint32_t * id )
```

AGHAF_CAN_asyncSendFrame.

Parameters

bus	handle on the CAN bus
frame	handle on the frame

Returns

7.4.3.2 AGHAF_CAN_asyncSendFrame_deregisterCallback()

```
void AGHAF_API AGHAF_CAN_asyncSendFrame_deregisterCallback (
            AGHAF_CAN_Bus bus,
            AGHAF_CAN_asyncSendFrame_Callback callback )
```

AGHAF_CAN_asyncSendFrame_deregisterCallback.

Parameters

bus	handle on the bus
callback	function pointer to remove

7.4.3.3 AGHAF_CAN_asyncSendFrame_registerCallback()

```
void AGHAF_API AGHAF_CAN_asyncSendFrame_registerCallback (
            AGHAF_CAN_Bus bus,
            AGHAF_CAN_asyncSendFrame_Callback callback,
            void * userContext )
```

AGHAF_CAN_asyncSendFrame_registerCallback.

Parameters

bus	handle on the bus
callback	function pointer to call when a frame has been sent
userContext	

7.4.3.4 AGHAF_CAN_Bus_activate()

```
AGHAF_Status AGHAF_API AGHAF_CAN_Bus_activate (
            AGHAF_CAN_Bus bus )
```

activate a CAN bus

Parameters

bus	handle on the bus

Returns

status request information

Examples

Can_FD/main.cpp, Can_LS/main.cpp, and DoCan_FD/main.cpp.

7.4.3.5 AGHAF_CAN_Bus_addFilter()

```
AGHAF_Status AGHAF_API AGHAF_CAN_Bus_addFilter (
            AGHAF_CAN_Bus bus,
```

```
            AGHAF_CAN_FilterId idType,
            AGHAF_CAN_FILTER_Type type,
            AGHAF_CAN_FilterMode mode,
            uint32_t id1,
            uint32_t id2 )
```

add a filter to the CAN bus

Parameters

bus	handle on the bus
idType	standard or extended CAN
type	type from the filter
mode	mode from the filter
id1	
id2	

Returns

status request information

7.4.3.6 AGHAF_CAN_Bus_BusOn()

```
AGHAF_Status AGHAF_API AGHAF_CAN_Bus_BusOn (
            AGHAF_CAN_Bus bus )
```

reactivate the bus after an error

Parameters

bus	handle on the bus

Returns

status request information

7.4.3.7 AGHAF_CAN_Bus_clearFilters()

```
AGHAF_Status AGHAF_API AGHAF_CAN_Bus_clearFilters (
            AGHAF_CAN_Bus bus )
```

clear the filters from a bus

Parameters

bus	handle on the bus

Returns

status request information

7.4.3.8 AGHAF_CAN_Bus_deactivate()

```
AGHAF_Status AGHAF_API AGHAF_CAN_Bus_deactivate (
            AGHAF_CAN_Bus bus )
```

deactivate a CAN bus

Parameters

bus	handle on the bus

Returns

status request information

Examples

Can_FD/main.cpp, Can_LS/main.cpp, and DoCan_FD/main.cpp.

7.4.3.9 AGHAF_CAN_Bus_filtersCount()

```
AGHAF_Status AGHAF_API AGHAF_CAN_Bus_filtersCount (
            AGHAF_CAN_Bus bus,
            AGHAF_CAN_FilterId idType,
            uint16_t * count )
```

Provide the current number of filters.

Parameters

bus	handle on the bus
idType	standard or extended CAN
count	pointer to fill with the data

Returns

status request information

7.4.3.10 AGHAF_CAN_Bus_freeSupportedFilters()

```
void AGHAF_API AGHAF_CAN_Bus_freeSupportedFilters (
            AGHAF_CAN_FilterDesc * values )
```

Free the memory allocated with AGHAF_CAN_Bus_supportedFilters.

Parameters

values	array to free

7.4.3.11 AGHAF_CAN_Bus_freeSupportedModes()

```
void AGHAF_API AGHAF_CAN_Bus_freeSupportedModes (
            AGHAF_CAN_Mode * modes )
```

Free the memory allocated with AGHAF_CAN_Bus_supportedModes.

Parameters

modes	array to free

7.4.3.12 AGHAF_CAN_Bus_freeSupportedTerminations()

```
void AGHAF_API AGHAF_CAN_Bus_freeSupportedTerminations (
            uint32_t * terminations )
```

Free the memory allocated with AGHAF_CAN_Bus_supportedTerminations or AGHAF_CAN_Bus_supported↩ TerminationsLs.

Parameters

terminations	array to free

7.4.3.13 AGHAF_CAN_Bus_getParam()

```
AGHAF_Status AGHAF_API AGHAF_CAN_Bus_getParam (
            AGHAF_CAN_Bus bus,
            AGHAF_CAN_Param param,
            uint32_t * value )
```

Provide the value from a CAN parameter.

Parameters

bus	handle on the CAN bus
param	parameter to request
value	pointer to fill with the content

Returns

status request information

7.4.3.14 AGHAF_CAN_Bus_getPeriodicCount()

```
uint8_t AGHAF_API AGHAF_CAN_Bus_getPeriodicCount (
            AGHAF_CAN_Bus bus )
```

Return the number of periodic message available on a CAN bus.

Parameters

bus	handle on the bus

Returns

the number of periodic message available

7.4.3.15 AGHAF_CAN_Bus_isActivated()

```
AGHAF_Status AGHAF_API AGHAF_CAN_Bus_isActivated (
            AGHAF_CAN_Bus bus,
            AGHAF_BOOL * isActivated )
```

Request if a CAN bus is yet activated or not.

Parameters

bus	handle on the bus
isActivated	pointer to fill with the information

Returns

status request information

7.4.3.16 AGHAF_CAN_Bus_reject()

```
AGHAF_Status AGHAF_API AGHAF_CAN_Bus_reject (
            AGHAF_CAN_Bus bus,
            AGHAF_CAN_RejectType rejectType,
            AGHAF_BOOL rtr )
```

Set a global filter.

Parameters

bus	handle on the bus
rejectType	type of reject desired
rtr	

Returns

status request information

7.4.3.17 AGHAF_CAN_Bus_sendPeriodic()

```
AGHAF_Status AGHAF_API AGHAF_CAN_Bus_sendPeriodic (
            AGHAF_CAN_Bus bus,
            AGHAF_CAN_Frame frame,
            uint8_t index,
            uint8_t flags,
            uint32_t period )
```

Add a periodic message to send.

Parameters

bus	handle on the bus
frame	Frame to send periodically
index	index in which to save the frame
flags	corresponds to flags starting with AGHAF_CAN_PERIODIC
period	period at which sending the frame

Returns

status request information

7.4.3.18 AGHAF_CAN_Bus_setParam()

```
AGHAF_Status AGHAF_API AGHAF_CAN_Bus_setParam (
            AGHAF_CAN_Bus bus,
```

```
            AGHAF_CAN_Param param,
            uint32_t value )
```

Set the value from a CAN parameter.

Parameters

bus	handle on the CAN bus
param	parameter to set
value	value to set

Returns

status request information

Examples

Can_FD/main.cpp, Can_LS/main.cpp, and DoCan_FD/main.cpp.

7.4.3.19 AGHAF_CAN_Bus_state()

```
AGHAF_Status AGHAF_API AGHAF_CAN_Bus_state (
            AGHAF_CAN_Bus bus,
            AGHAF_CAN_BusState * canBusState )
```

AGHAF_CAN_Bus_state.

Parameters

bus	
canBusState	UNKNOWN if can chip is not started, can chip state otherwise

Returns

status request information

7.4.3.20 AGHAF_CAN_Bus_supportedFilters()

```
AGHAF_Status AGHAF_API AGHAF_CAN_Bus_supportedFilters (
            AGHAF_CAN_Bus bus,
            uint16_t * globalFilterCaps,
            AGHAF_CAN_FilterDesc ** values,
            uint16_t * size )
```

Provide the supported filters from a CAN bus.

Parameters

bus	handle on the bus
globalFilterCaps	pointer to fill with global cappabilites informations
values	pointer to fill with the datas
size	size from values arrays

Returns

status request information

Warning

the memory allocated in values must freed with AGHAF_CAN_Bus_freeSupportedFilters

7.4.3.21 AGHAF_CAN_Bus_supportedModes()

```
AGHAF_Status AGHAF_API AGHAF_CAN_Bus_supportedModes (
            AGHAF_CAN_Bus bus,
            AGHAF_CAN_Mode ** modes,
            uint8_t * nbModes )
```

Request the supported modes from the CAN bus in parameter.

Parameters

bus	handle on the bus
modes	pointer on the datas
nbModes	size from the modes array

Returns

status request information

Warning

the memory allocated in modes must freed with AGHAF_CAN_Bus_freeSupportedModes

7.4.3.22 AGHAF_CAN_Bus_supportedTerminations()

```
AGHAF_Status AGHAF_API AGHAF_CAN_Bus_supportedTerminations (
            AGHAF_CAN_Bus bus,
            uint32_t ** terminations,
            uint32_t * nbTerminations )
```

Request the supported terminations from a CAN bus.

Parameters

bus	handle on the bus
terminations	pointer on the datas
nbTerminations	size from the modes array

Returns

status request information

Warning

the memory allocated in terminations must freed with AGHAF_CAN_Bus_freeSupportedTerminations

7.4.3.23 AGHAF_CAN_Bus_supportedTerminationsLs()

```
AGHAF_Status AGHAF_API AGHAF_CAN_Bus_supportedTerminationsLs (
            AGHAF_CAN_Bus bus,
            uint32_t ** terminations,
            uint32_t * nbTerminations )
```

Request the supported low speed terminations from a CAN bus.

Parameters

bus	handle on the bus
terminations	pointer on the datas
nbTerminations	size from the modes array

Returns

status request information

Warning

the memory allocated in terminations must freed with AGHAF_CAN_Bus_freeSupportedTerminations

7.4.3.24 AGHAF_CAN_deregisterCallback()

```
void AGHAF_API AGHAF_CAN_deregisterCallback (
            AGHAF_CAN_Bus bus,
            AGHAF_CAN_Callback callback )
```

deregister a callback

Parameters

bus	handle on the bus
callback	function pointer to remove

Examples

Can_FD/main.cpp, and Can_LS/main.cpp.

7.4.3.25 AGHAF_CAN_Event_busError()

```
uint32_t AGHAF_API AGHAF_CAN_Event_busError (
            AGHAF_CAN_Event_busInfo busState )
```

return the value from the error

Parameters

busState	

Returns

corresponds to AGHAF_CAN_Error enumerator

7.4.3.26 AGHAF_CAN_Event_busLoad()

```
uint32_t AGHAF_API AGHAF_CAN_Event_busLoad (
            AGHAF_CAN_Event_busInfo busState )
```

return the bus load

Parameters

busState	

Returns

7.4.3.27 AGHAF_CAN_Event_chipState()

```
uint32_t AGHAF_API AGHAF_CAN_Event_chipState (
            AGHAF_CAN_Event_busInfo busState )
```

return the chipstate

Parameters

busState	

Returns

7.4.3.28 AGHAF_CAN_Event_frame()

```
AGHAF_CAN_Frame AGHAF_API AGHAF_CAN_Event_frame (
            AGHAF_CAN_Event event )
```

return the frame contained in an event

Parameters

event	handle on an event

Returns

the frame

Examples

Can_FD/main.cpp, and Can_LS/main.cpp.

7.4.3.29 AGHAF_CAN_Frame_brs()

```
AGHAF_BOOL AGHAF_API AGHAF_CAN_Frame_brs (
            AGHAF_CAN_Frame frame )
```

AGHAF_CAN_Frame_brs.

Parameters

frame	handle on the frame

Returns

brs value which indicates if data field is sent to the FD baudrate or not

7.4.3.30 AGHAF_CAN_Frame_data()

```
AGHAF_Status AGHAF_API AGHAF_CAN_Frame_data (
            AGHAF_CAN_Frame frame,
            void * data,
            int64_t size )
```

Provide the data contained in a frame.

Parameters

frame	handle on the frame
data	array to fill
size	size that has been filled

Returns

status request information

Examples

Can_FD/main.cpp, and Can_LS/main.cpp.

7.4.3.31 AGHAF_CAN_Frame_dataSize()

```
int64_t AGHAF_API AGHAF_CAN_Frame_dataSize (
            AGHAF_CAN_Frame frame )
```

return the size from the data

Parameters

frame	handle on the frame

Returns

size from the datas in the frame

Examples

Can_FD/main.cpp, and Can_LS/main.cpp.

7.4.3.32 AGHAF_CAN_Frame_delete()

```
void AGHAF_API AGHAF_CAN_Frame_delete (
            AGHAF_CAN_Frame frame )
```

Free the memory from a CAN frame.

Parameters

frame	handle to free

Examples

Can_FD/main.cpp, and Can_LS/main.cpp.

7.4.3.33 AGHAF_CAN_Frame_esi()

```
AGHAF_BOOL AGHAF_API AGHAF_CAN_Frame_esi (
            AGHAF_CAN_Frame frame )
```

AGHAF_CAN_Frame_esi.

Parameters

frame	handle on the frame

Returns

error state indicator value

7.4.3.34 AGHAF_CAN_Frame_fdf()

```
AGHAF_BOOL AGHAF_API AGHAF_CAN_Frame_fdf (
            AGHAF_CAN_Frame frame )
```

AGHAF_CAN_Frame_fdf.

Parameters

frame	handle on the frame

Returns

fdf value which indicates if FD is activated or not

7.4.3.35 AGHAF_CAN_Frame_frameType()

```
AGHAF_CAN_FrameType AGHAF_API AGHAF_CAN_Frame_frameType (
            AGHAF_CAN_Frame frame )
```

Get the type from the frame.

Parameters

frame	handle on the frame

Returns

type from the frame

7.4.3.36 AGHAF_CAN_Frame_id()

```
AGHAF_Status AGHAF_API AGHAF_CAN_Frame_id (
            AGHAF_CAN_Frame frame,
            AGHAF_CAN_Identifier * identifier )
```

Get the identifier from a CAN frame.

Parameters

frame	handle on the frame
identifier	

Returns

status request information

Examples

Can_FD/main.cpp, and Can_LS/main.cpp.

7.4.3.37 AGHAF_CAN_Frame_new()

```
AGHAF_CAN_Frame AGHAF_API AGHAF_CAN_Frame_new ( )
```

Return a handle on a CAN frame.

Returns

handle on a CAN frame

Warning

the handle must be freed with AGHAF_CAN_Frame_delete

Examples

Can_FD/main.cpp, and Can_LS/main.cpp.

7.4.3.38 AGHAF_CAN_Frame_setData()

```
AGHAF_Status AGHAF_API AGHAF_CAN_Frame_setData (
            AGHAF_CAN_Frame frame,
            const void * data,
            int64_t size )
```

Set the data from a frame.

Parameters

frame	handle on the frame
data	data to set
size	size from data array

Returns

status request information

Examples

Can_FD/main.cpp, and Can_LS/main.cpp.

7.4.3.39 AGHAF_CAN_Frame_setFDF()

```
AGHAF_Status AGHAF_API AGHAF_CAN_Frame_setFDF (
            AGHAF_CAN_Frame frame,
            AGHAF_BOOL fdf,
            AGHAF_BOOL esi,
            AGHAF_BOOL brs )
```

AGHAF_CAN_Frame_setFDF.

Parameters

frame	handle on the frame
fdf	activate the FD for the frame (allow to send more than 8 bytes)
esi	error state indicator
brs	if true then the field data is sent to the FD baudrate

Returns

Examples

Can_FD/main.cpp.

7.4.3.40 AGHAF_CAN_Frame_setFrameType()

```
AGHAF_Status AGHAF_API AGHAF_CAN_Frame_setFrameType (
            AGHAF_CAN_Frame frame,
            AGHAF_CAN_FrameType type )
```

Set the type from the frame.

Parameters

frame	handle on the frame
type	data or remote (a remote frame is completed by another device on the CAN bus)

Returns

status request information

7.4.3.41 AGHAF_CAN_Frame_setId()

```
AGHAF_Status AGHAF_API AGHAF_CAN_Frame_setId (
            AGHAF_CAN_Frame frame,
            AGHAF_CAN_Identifier * identifier )
```

Set the identifier from a CAN frame.

Parameters

frame	handle on the frame
identifier	

Returns

status request information

Examples

Can_FD/main.cpp, and Can_LS/main.cpp.

7.4.3.42 AGHAF_CAN_getBus()

```
AGHAF_CAN_Bus AGHAF_API AGHAF_CAN_getBus (
            AGHAF_Device device,
            uint8_t index )
```

return the CAN bus at index in parameter

Parameters

device	device handle from the desired CAN bus
index	index from the bus

Returns

Handle on a CAN bus

Examples

Can_FD/main.cpp, Can_LS/main.cpp, and DoCan_FD/main.cpp.

7.4.3.43 AGHAF_CAN_getBusCount()

```
uint8_t AGHAF_API AGHAF_CAN_getBusCount (
            AGHAF_Device device )
```

return the number of CAN bus from a device

Parameters

device	handle from the device

Returns

Number of CAN bus on the device

7.4.3.44 AGHAF_CAN_getBusIndex()

```
uint8_t AGHAF_API AGHAF_CAN_getBusIndex (
            AGHAF_CAN_Bus bus )
```

return the index from a CAN bus

Parameters

bus	handle from the CAN bus

Returns

index from the bus

7.4.3.45 AGHAF_CAN_getEventBusIndex()

```
uint8_t AGHAF_API AGHAF_CAN_getEventBusIndex (
            AGHAF_CAN_Event event )
```

return the index from the bus on which occured the event

Parameters

event	handle on an event

Returns

index from bus

7.4.3.46 AGHAF_CAN_getEventBusInfo()

```
AGHAF_CAN_Event_busInfo AGHAF_API AGHAF_CAN_getEventBusInfo (
            AGHAF_CAN_Event event )
```

return the bus informations from an event

Parameters

event	handle on an event

Returns

7.4.3.47 AGHAF_CAN_registerCallback()

```
void AGHAF_API AGHAF_CAN_registerCallback (
            AGHAF_CAN_Bus bus,
            AGHAF_CAN_Callback callback,
            void * userContext )
```

Register a callback for the CAN events from a bus.

Parameters

bus	handle on the bus
callback	function pointer to call on an event
userContext	

Examples

Can_FD/main.cpp, and Can_LS/main.cpp.

7.4.3.48 AGHAF_CAN_sendFrame()

```
AGHAF_Status AGHAF_API AGHAF_CAN_sendFrame (
            AGHAF_CAN_Bus bus,
            AGHAF_CAN_Frame frame )
```

send a frame on a CAN bus

Parameters

bus	handle on the CAN bus
frame	handle on the frame

Returns

status request information

Examples

Can_FD/main.cpp, and Can_LS/main.cpp.

7.4.3.49 AGHAF_CAN_typeEvent()

```
AGHAF_CAN_EventType AGHAF_API AGHAF_CAN_typeEvent (
            AGHAF_EventInfo event )
```

return the type from the event

Parameters

event	

Returns

type from the event

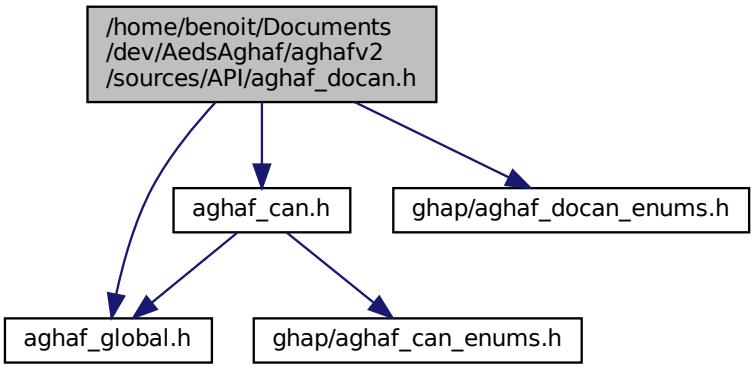
Examples

Can_FD/main.cpp, and Can_LS/main.cpp.

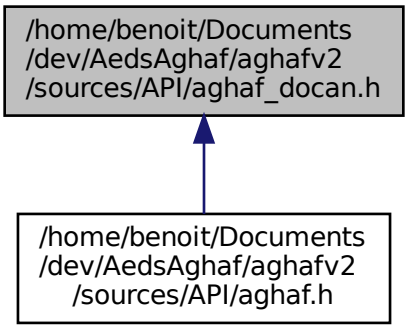
7.5 /home/benoit/Documents/dev/AedsAghaf/aghafv2/sources/API/aghaf_docan.h File Reference

```
#include "aghaf_global.h"
#include "aghaf_can.h"
#include <ghap/aghaf_docan_enums.h>
```

Include dependency graph for aghaf_docan.h:



This graph shows which files directly or indirectly include this file:



- typedef void * AGHAF_DoCAN_Bus
- typedef void(* AGHAF_DoCAN_Callback) (AGHAF_DoCAN_Event, void *)
- typedef void * AGHAF_DoCAN_Channel
- void AGHAF_API AGHAF_DoCAN_Channel_activateTrace (AGHAF_DoCAN_Channel channel, AGHAF_BOOL value)

  Activate the CAN trace.

- AGHAF_Status AGHAF_API AGHAF_DoCAN_Channel_getParam (AGHAF_DoCAN_Channel channel, AGHAF_DoCAN_ChannelParam param, uint32_t *value)

  Provide the value from a channel parameter.
- AGHAF_Status AGHAF_API AGHAF_DoCAN_Channel_send (AGHAF_DoCAN_Channel channel, uint32_t dataLen, uint8_t const *data)

  Send on the DoCAN channel in parameter.
- AGHAF_Status AGHAF_API AGHAF_DoCAN_Channel_setParam (AGHAF_DoCAN_Channel channel, AGHAF_DoCAN_ChannelParam param, uint32_t value)

  Set the parameter from a channel.
- AGHAF_Status AGHAF_API AGHAF_DoCAN_Channel_start (AGHAF_DoCAN_Channel channel)

  Start a channel.
- AGHAF_Status AGHAF_API AGHAF_DoCAN_Channel_stop (AGHAF_DoCAN_Channel channel)

  Stop a channel.
- AGHAF_DoCAN_Channel AGHAF_API AGHAF_DoCAN_createChannel (AGHAF_DoCAN_Bus bus)

  Create a channel.
- void AGHAF_API AGHAF_DoCAN_deregisterCallback (AGHAF_DoCAN_Bus bus, AGHAF_DoCAN_Callback callback)

  deregister a callback
- enum AGHAF_DoCAN_ERROR { **AGHAF_DoCAN_ERROR_UNKNOWN** = 0, AGHAF_DoCAN_ERROR_OK = 1, AGHAF_DoCAN_ERROR_SEQUENCE = 2, AGHAF_DoCAN_ERROR_PARAM = 3, AGHAF_DoCAN_ERROR_UNIMPLEMENTED = 4, AGHAF_DoCAN_ERROR_FIFOFULL = 5, AGHAF_DoCAN_ERROR_BUS = 6 }
- typedef void * AGHAF_DoCAN_Event
- uint8_t AGHAF_API AGHAF_DoCAN_Event_getBusIndex (AGHAF_DoCAN_Event event)

  return the index from the bus on which occured the event
- AGHAF_DoCAN_Channel AGHAF_API AGHAF_DoCAN_Event_getChannel (AGHAF_DoCAN_Event event)

  Return the channel on which occured the event.
- AGHAF_Status AGHAF_API AGHAF_DoCAN_Event_getData (AGHAF_DoCAN_Event event, uint8_t *data, uint32_t *size)

  Provide the datas received in the event.
- uint32_t AGHAF_API AGHAF_DoCAN_Event_getDataSize (AGHAF_DoCAN_Event event)

  return the size from the datas in the event
- uint8_t AGHAF_API AGHAF_DoCAN_Event_getError (AGHAF_DoCAN_Event event)

  Return the error value from the event.
- AGHAF_CAN_Identifier AGHAF_API AGHAF_DoCAN_Event_getId (AGHAF_DoCAN_Event event)

  Return the CAN identifier from the event.
- AGHAF_DoCAN_EventType AGHAF_API AGHAF_DoCAN_Event_type (AGHAF_EventInfo event)

  return the type from the event
- enum AGHAF_DoCAN_EventType { AGHAF_DoCAN_EVENT_MSGTX = 0, AGHAF_DoCAN_EVENT_MSGTXERR = 1, AGHAF_DoCAN_EVENT_MSGRX = 2, AGHAF_DoCAN_EVENT_MSGRXFF = 3, AGHAF_DoCAN_EVENT_MSGRXERR = 4 }
- typedef void * AGHAF_DoCAN_Frame
- AGHAF_DoCAN_Bus AGHAF_API AGHAF_DoCAN_getBus (AGHAF_Device device, uint8_t index)

  return the DoCAN bus at index in parameter
- uint32_t AGHAF_API AGHAF_DoCAN_getBusCount (AGHAF_Device device)

  return the number of DoCAN bus from a device
- AGHAF_Status AGHAF_API AGHAF_DoCAN_getChannelCount (AGHAF_DoCAN_Bus bus, uint8_t *count)

  Provide the number of channels available on the bus.
- AGHAF_Status AGHAF_API **AGHAF_DoCAN_getFreeChannelCount** (AGHAF_DoCAN_Bus bus, uint8_t *count)

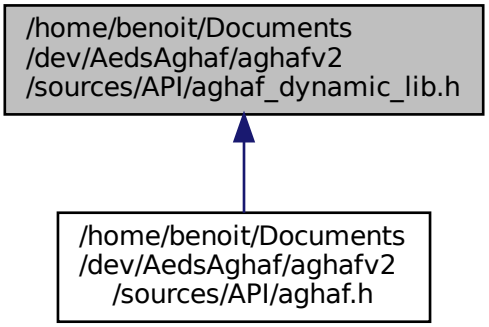
- void AGHAF_API AGHAF_DoCAN_registerCallback (AGHAF_DoCAN_Bus bus, AGHAF_DoCAN_Callback callback, void ∗userContext)

  Register a callback for the DoCAN events from a bus.
- AGHAF_Status AGHAF_API AGHAF_DoCAN_releaseChannel (AGHAF_DoCAN_Channel channel)

  Release the channel.

7.6 /home/benoit/Documents/dev/AedsAghaf/aghafv2/sources/API/aghaf_dynamic_lib.h File Reference

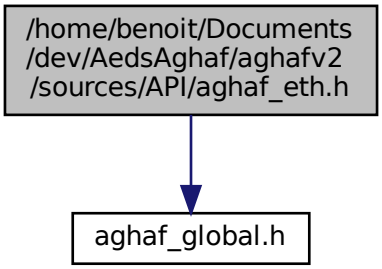
This graph shows which files directly or indirectly include this file:



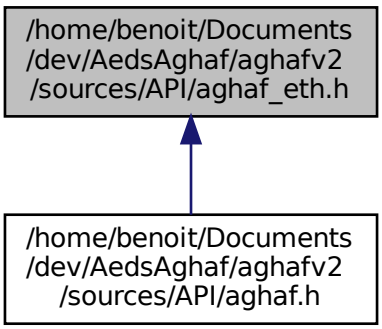
7.7 /home/benoit/Documents/dev/AedsAghaf/aghafv2/sources/API/aghaf_eth.h File Reference

```
#include "aghaf_global.h"
```

Include dependency graph for aghaf_eth.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct AGHAF_ETH_MacAddress

- #define **AGHAF_ETH_ALEN** 6
- void AGHAF_API AGHAF_ETH_freeFriendlyName (char ∗friendlyName)

  Free the memory allocated with AGHAF_Device_getFriendlyNameEthCard.
- AGHAF_EthBus AGHAF_API AGHAF_ETH_getBus (AGHAF_Device device, uint8_t index)

  return the Ethernet bus at index in parameter
- uint8_t AGHAF_API AGHAF_ETH_getBusCount (AGHAF_Device device)

  return the number of CAN bus from a device
- uint8_t AGHAF_API AGHAF_ETH_getBusIndex (AGHAF_EthBus bus)

  return the index from a Ethernet
- void AGHAF_API AGHAF_ETH_getFriendlyNameEthCard (AGHAF_EthBus bus, char ∗∗name)

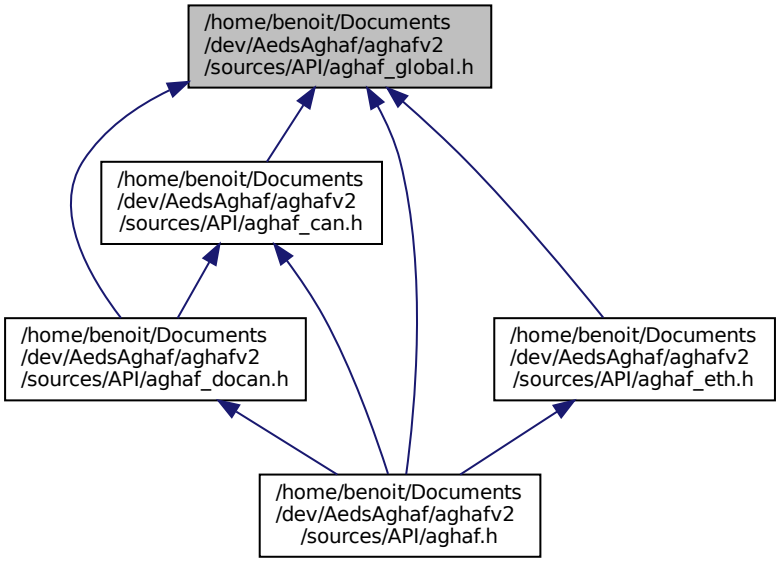
  AGHAF_ETH_getFriendlyNameEthCard.
- AGHAF_Status AGHAF_API AGHAF_ETH_getMacAddress (AGHAF_EthBus bus, AGHAF_ETH_MacAddress ∗macAddress)

  return the Ethernet mac address on device "device", and channel "index"
- typedef struct AGHAF_ETH_MacAddress **AGHAF_ETH_MacAddress**
- enum **AGHAF_ETH_Phy** { **AGHAF_ETH_InvalidPhy** = -1, **AGHAF_ETH_BroadRReach**, **AGHAF_ETH_IEEE** }
- AGHAF_Status AGHAF_API AGHAF_ETH_setEthernetDiagLineState (AGHAF_EthBus bus, AGHAF_BOOL active)

  AGHAF_ETH_setEthernetDiagLineState.
- typedef void ∗ **AGHAF_EthBus**

7.8 /home/benoit/Documents/dev/AedsAghaf/aghafv2/sources/API/aghaf_global.h File Reference

This graph shows which files directly or indirectly include this file:



Classes

- struct AGHAF_DeviceInfo
- struct AGHAF_EventInfo
- struct AGHAF_IpAddress

  represent an IP address

Macros

- #define **AGHAF_API**
- #define **AGHAF_DEPRECATED_FOR**(f)
- #define **AGHAF_IMPORTEXPORT**
- #define **AGHAF_PTR_SIZE** 4
- #define **AGHAF_PUBLIC_FUNCTION**(ret, name, params) AGHAF_IMPORTEXPORT ret AGHAF_API name params

Typedefs

- typedef short **int16_t**
- typedef int **int32_t**
- typedef long long **int64_t**
- typedef signed char **int8_t**
- typedef int32_t **intptr_t**
- typedef unsigned short **uint16_t**
- typedef unsigned int **uint32_t**
- typedef unsigned long long **uint64_t**
- typedef unsigned char **uint8_t**
- typedef uint32_t **uintptr_t**

- enum AGHAF_BOOL { AGHAF_FALSE = 0, AGHAF_TRUE = 1 }
- typedef void(∗ **AGHAF_Callback**) (AGHAF_Event event, void ∗)
- AGHAF_Status AGHAF_API AGHAF_closeDevice (AGHAF_Device device)

  close the device
- enum AGHAF_ConnectionMode { AGHAF_OFFLINE, AGHAF_USB, AGHAF_ETH }
- void AGHAF_API AGHAF_deregisterCallback (AGHAF_Callback callback)

  Deregister a callback.
- typedef void ∗ **AGHAF_Device**
- void AGHAF_API AGHAF_Device_freeFriendlyName (char ∗friendlyName)

  Free the memory allocated with AGHAF_Device_getFriendlyNameEthCard.
- AGHAF_ConnectionMode AGHAF_API AGHAF_Device_getConnectionMode (AGHAF_Device device)

  AGHAF_Device_getConnectionMode.
- void AGHAF_API AGHAF_Device_getFriendlyNameEthCard (AGHAF_Device device, char ∗∗name)

  AGHAF_Device_getFriendlyNameEthCard.
- AGHAF_Status AGHAF_API AGHAF_Device_getUsbInfo (AGHAF_Device device, uint16_t ∗vid, uint16_t ∗pid, uint16_t ∗rev)

  AGHAF_Device_getUsbInfo.
- enum AGHAF_DeviceEvent { AGHAF_Device_noEvent, AGHAF_Device_Arrival, AGHAF_Device_Leave, AGHAF_Device_Present }
- uint8_t const ∗AGHAF_API AGHAF_DeviceEvent_getHardwareUniqueId (AGHAF_Event event)

  return the hardware unique id from the device which emitted the device event
- char const ∗AGHAF_API AGHAF_DeviceEvent_getProductName (AGHAF_Event event)

  return the product name from the device which emitted the device event
- char const ∗AGHAF_API AGHAF_DeviceEvent_getProductNumber (AGHAF_Event event)

  return the product number from the device which emitted the device event
- char const ∗AGHAF_API AGHAF_DeviceEvent_getSerialNumber (AGHAF_Event event)

  return the serial number from the device which emitted the device event
- typedef struct AGHAF_DeviceInfo **AGHAF_DeviceInfo**
- enum AGHAF_DeviceState { AGHAF_DeviceState_error = 0, AGHAF_DeviceState_boot, AGHAF_DeviceState_ready }
- typedef enum AGHAF_DeviceState **AGHAF_DeviceState**
- typedef void ∗ **AGHAF_Event**
- AGHAF_Status AGHAF_API AGHAF_Event_getInfo (AGHAF_Event event, AGHAF_EventInfo ∗info)

  AGHAF_Event_getInfo.
- typedef int AGHAF_EventHandle
- typedef struct AGHAF_EventInfo **AGHAF_EventInfo**
- void AGHAF_API AGHAF_freeDeviceList (AGHAF_DeviceInfo ∗devices)

  Free the memory allocated with AGHAF_getDeviceList.
- AGHAF_Device AGHAF_API AGHAF_getDeviceByHardID (uint8_t const ∗hardwareID)

AGHAF_getDeviceByHardID.

- AGHAF_Device AGHAF_API AGHAF_getDeviceBySN (const char *productNumber, const char *serialNumber)

  Return the handle on the specified device.
- uint32_t AGHAF_API AGHAF_getDeviceCount (void)

  Return the number of devices onnected.
- AGHAF_Status AGHAF_API AGHAF_getDeviceInfo (AGHAF_Device device, AGHAF_DeviceInfo *deviceInfo)

  Return the informations from a device.
- AGHAF_Status AGHAF_API AGHAF_getDeviceList (AGHAF_DeviceInfo **devices, uint32_t *size)

  Provide the list of devices connected.
- uint32_t AGHAF_API AGHAF_getServiceVersion (void)

  Return the version from aeds.
- uint32_t AGHAF_API AGHAF_getVersion (void)

  Return the version from Aghaf.
- const char *AGHAF_API AGHAF_getVersionString (void)

  Return the version from Aghaf.
- typedef struct AGHAF_IpAddress AGHAF_IpAddress

  represent an IP address
- AGHAF_BOOL AGHAF_API AGHAF_isServiceRunning (void)

  Inform if ghaston is running or not.
- AGHAF_Status AGHAF_API AGHAF_openDevice (AGHAF_Device device)

  Open a device.
- AGHAF_Status AGHAF_API AGHAF_refreshDeviceList (void)

  Call the device event callbacks with the devices connected.
- void AGHAF_API AGHAF_registerCallback (AGHAF_Callback callback, void *userData)

  Register a callback for the device event.
- enum AGHAF_Status {
  AGHAF_STATUS_OK = 0, AGHAF_ERROR_PARAM, AGHAF_ERROR_NOT_ENOUGH_MEMORY,
  AGHAF_ERROR_OUT_OF_MEMORY,
  AGHAF_ERROR_NOT_IMPLEMENTED, AGHAF_ERROR_END_REACHED, AGHAF_INTERNAL_ERROR,
  AGHAF_ERROR_CONNECTION,
  AGHAF_ERROR_BUS_NOT_FOUND, AGHAF_ERROR_PENDING, AGHAF_ERROR_WRONG_STATE,
  AGHAF_ERROR_NOT_ENOUGH_BUS,
  AGHAF_WRONG_SERVICE_VERSION, AGHAF_TIMEOUT_COM, AGHAF_DEPRECATED, AGHAF_ERROR_SEQUENCE,
  AGHAF_ERROR_FIFOFULL, AGHAF_ERROR_ALREADY_OPEN, AGHAF_ERROR_BUSY, AGHAF_ERROR_NOMORECH
  AGHAF_ERROR_UNKNOWN, AGHAF_ERROR_WRONG_SESSION_TYPE, AGHAF_ERROR_NO_DATA
  = 0xff }
- enum AGHAF_TypeEvent {
  AGHAF_TypeNoEvent, AGHAF_TypeDeviceEvent, AGHAF_TypeEthernetEvent, AGHAF_TypeCanEvent,
  AGHAF_TypeDoCanEvent }

Chapter 8

Example Documentation

8.1 Can_FD/main.cpp

```
#include <vector>
#include <iostream>
#define AGHAF_DYNAMIC_LIBRARY_INIT
#include <aghaf.h>
void canBus_callback(AGHAF_CAN_Event event, void *userContext)
{
    (void) userContext;
    AGHAF_EventInfo info;
    AGHAF_Status status = AGHAF_Event_getInfo(event, &info);
    if (status == AGHAF_STATUS_OK) {
        switch (info.type) {
        case AGHAF_TypeNoEvent:
        case AGHAF_TypeDeviceEvent:
        case AGHAF_TypeEthernetEvent:
        case AGHAF_TypeDoCanEvent:
            break;
        case AGHAF_TypeCanEvent: {
            AGHAF_CAN_EventType eventType = AGHAF_CAN_typeEvent(info);
            switch (eventType) {
            case AGHAF_CAN_EVENT_UNKNOW:
            case AGHAF_CAN_EVENT_MSGTX:
            case AGHAF_CAN_EVENT_ERROR:
            case AGHAF_CAN_EVENT_BUSCHANGE:
            case AGHAF_CAN_EVENT_BUSLOAD:
                break;
            case AGHAF_CAN_EVENT_MSGRX: {
                AGHAF_CAN_Frame frame = AGHAF_CAN_Event_frame(event);
                AGHAF_CAN_Identifier id;
                AGHAF_CAN_Frame_id(frame, &id);
                int64_t size = AGHAF_CAN_Frame_dataSize(frame);
                uint8_t *data = new uint8_t[size];
                AGHAF_CAN_Frame_data(frame, data, size);
                delete [] data;
            }
                break;
            }
        }
            break;
        }
    }
}
int main()
{
#ifdef _WIN32
    AGHAF_loadLibrary(AGHAF_DEFAULT_FILENAME);
#endif
    uint32_t count = AGHAF_getDeviceCount();
    if (count > 0) {
        AGHAF_DeviceInfo *devicesInfo = nullptr;
        uint32_t deviceInfoNumber = 0;
        AGHAF_getDeviceList(&devicesInfo, &deviceInfoNumber);
        AGHAF_Device device = AGHAF_getDeviceBySN(
                    devicesInfo[0].productNo,
                devicesInfo[0].serialNo);
        AGHAF_freeDeviceList(devicesInfo);
        AGHAF_openDevice(device);
        AGHAF_CAN_Bus canBus = AGHAF_CAN_getBus(device, 0);
        AGHAF_CAN_registerCallback(canBus,
```

```
                                &canBus_callback,
                                nullptr);
        // BUS CAN FD ------------------------------
        AGHAF_CAN_Bus_deactivate(canBus);
        AGHAF_CAN_Bus_setParam(canBus,
                               CAN_PARAM_LISTEN_ONLY,
                               AGHAF_FALSE);
        AGHAF_CAN_Bus_setParam(canBus,
                               CAN_PARAM_MODE,
                               AGHAF_CAN_MODE_FD);
        AGHAF_CAN_Bus_setParam(canBus,
                               CAN_PARAM_BAUDRATE,
                               1000000);
        AGHAF_CAN_Bus_setParam(canBus,
                               CAN_PARAM_BAUDRATE_FD,
                               5000000);
        AGHAF_CAN_Bus_setParam(canBus,
                               CAN_PARAM_TERMINATION,
                               120);
        AGHAF_CAN_Bus_activate(canBus);
        // -----------------------------------------
        std::vector<uint8_t> dataToSend;
        dataToSend.push_back(0x01);
        AGHAF_CAN_Identifier identifier;
        identifier.type = AGHAF_CAN_IdStd;
        identifier.id.std = 0x100;
        AGHAF_CAN_Frame frameToSend = AGHAF_CAN_Frame_new();
        AGHAF_CAN_Frame_setFDF(frameToSend, AGHAF_TRUE, AGHAF_FALSE, AGHAF_TRUE);
        AGHAF_CAN_Frame_setId(frameToSend, &identifier);
        AGHAF_CAN_Frame_setData(frameToSend,
                                dataToSend.data(),
                                static_cast<uint32_t>(
                                    dataToSend.size()));
        AGHAF_CAN_sendFrame(canBus, frameToSend);
        AGHAF_CAN_Frame_delete(frameToSend);
        AGHAF_CAN_Bus_deactivate(canBus);
        AGHAF_CAN_deregisterCallback(canBus, &canBus_callback);
        AGHAF_closeDevice(device);
    }
#ifdef _WIN32
    AGHAF_unloadLibrary();
#endif
    return 0;
}
```

8.2 Can_LS/main.cpp

```
#include <vector>
#include <iostream>
#define AGHAF_DYNAMIC_LIBRARY_INIT
#include <aghaf.h>
void canBus_callback(AGHAF_CAN_Event event, void *userContext)
{
    (void) userContext;
    AGHAF_EventInfo info;
    AGHAF_Status status = AGHAF_Event_getInfo(event, &info);
    if (status == AGHAF_STATUS_OK) {
        switch (info.type) {
        case AGHAF_TypeNoEvent:
        case AGHAF_TypeDeviceEvent:
        case AGHAF_TypeEthernetEvent:
        case AGHAF_TypeDoCanEvent:
            break;
        case AGHAF_TypeCanEvent: {
            AGHAF_CAN_EventType eventType = AGHAF_CAN_typeEvent(info);
            switch (eventType) {
            case AGHAF_CAN_EVENT_UNKNOW:
            case AGHAF_CAN_EVENT_MSGTX:
            case AGHAF_CAN_EVENT_ERROR:
            case AGHAF_CAN_EVENT_BUSCHANGE:
            case AGHAF_CAN_EVENT_BUSLOAD:
                break;
            case AGHAF_CAN_EVENT_MSGRX: {
                AGHAF_CAN_Frame frame = AGHAF_CAN_Event_frame(event);
                AGHAF_CAN_Identifier id;
                AGHAF_CAN_Frame_id(frame, &id);
                int64_t size = AGHAF_CAN_Frame_dataSize(frame);
                uint8_t *data = new uint8_t[size];
                AGHAF_CAN_Frame_data(frame, data, size);
                delete [] data;
            }
                break;
            }
```

```
        }
            break;
        }
    }
}
int main()
{
#ifdef _WIN32
    AGHAF_loadLibrary(AGHAF_DEFAULT_FILENAME);
#endif
    uint32_t count = AGHAF_getDeviceCount();
    if (count > 0) {
        AGHAF_DeviceInfo *devicesInfo = nullptr;
        uint32_t deviceInfoNumber = 0;
        AGHAF_getDeviceList(&devicesInfo, &deviceInfoNumber);
        AGHAF_Device device = AGHAF_getDeviceBySN(
                    devicesInfo[0].productNo,
                devicesInfo[0].serialNo);
        AGHAF_freeDeviceList(devicesInfo);
        AGHAF_openDevice(device);
        AGHAF_CAN_Bus canBus = AGHAF_CAN_getBus(device, 0);
        AGHAF_CAN_registerCallback(canBus,
                                   &canBus_callback,
                                   nullptr);
        // BUS CAN LS ------------------------------
        AGHAF_CAN_Bus_deactivate(canBus);
        AGHAF_CAN_Bus_setParam(canBus,
                               CAN_PARAM_LISTEN_ONLY,
                               AGHAF_FALSE);
        AGHAF_CAN_Bus_setParam(canBus,
                               CAN_PARAM_MODE,
                               AGHAF_CAN_MODE_LS);
        AGHAF_CAN_Bus_setParam(canBus,
                               CAN_PARAM_BAUDRATE,
                               125000);
        AGHAF_CAN_Bus_setParam(canBus,
                               CAN_PARAM_TERMINATION,
                               120);
        AGHAF_CAN_Bus_activate(canBus);
        // -----------------------------------------
        std::vector<uint8_t> dataToSend;
        dataToSend.push_back(0x01);
        AGHAF_CAN_Identifier identifier;
        identifier.type = AGHAF_CAN_IdStd;
        identifier.id.std = 0x100;
        AGHAF_CAN_Frame frameToSend = AGHAF_CAN_Frame_new();
        AGHAF_CAN_Frame_setId(frameToSend, &identifier);
        AGHAF_CAN_Frame_setData(frameToSend,
                                dataToSend.data(),
                                static_cast<uint32_t>(
                                    dataToSend.size()));
        AGHAF_CAN_sendFrame(canBus, frameToSend);
        AGHAF_CAN_Frame_delete(frameToSend);
        AGHAF_CAN_Bus_deactivate(canBus);
        AGHAF_CAN_deregisterCallback(canBus, &canBus_callback);
        AGHAF_closeDevice(device);
    }
#ifdef _WIN32
    AGHAF_unloadLibrary();
#endif
    return 0;
}
```

8.3 DeviceManagement/main.cpp

```
#include <vector>
#include <unordered_map>
#include <iostream>
#include <string>
#include <algorithm>
#include <array>
#define AGHAF_DYNAMIC_LIBRARY_INIT
#include <aghaf.h>
struct pair_hash
{
    template <class T1, class T2>
    std::size_t operator() (const std::pair<T1, T2> &pair) const
    {
        return std::hash<T1>()(pair.first) ^ std::hash<T2>()(pair.second);
    }
};
std::unordered_map<std::pair<std::string, std::string>, AGHAF_Device, pair_hash> devices; // key:
       productNumber, serialNumber
```

```
void deviceEvent_callback(void * event, void *userContext)
{
    (void) userContext;
    AGHAF_EventInfo info;
    AGHAF_Event_getInfo(event, &info);
    switch (info.type) {
    case AGHAF_TypeNoEvent:
    case AGHAF_TypeEthernetEvent:
    case AGHAF_TypeCanEvent:
    case AGHAF_TypeDoCanEvent:
        break;
    case AGHAF_TypeDeviceEvent: {
        std::string productName(AGHAF_DeviceEvent_getProductName(event));
        std::string productNumber(AGHAF_DeviceEvent_getProductNumber(event));
        std::array<uint8_t, 16> hardareUniqueId = { 0 };
        std::copy_n(AGHAF_DeviceEvent_getHardwareUniqueId(event), 16, hardareUniqueId.data());
        std::string serialNumber(AGHAF_DeviceEvent_getSerialNumber(event));
        std::pair<std::string, std::string> key(productNumber, serialNumber);
        switch (static_cast<AGHAF_DeviceEvent>(info.event)) {
        case AGHAF_Device_noEvent:
            break;
        case AGHAF_Device_Arrival: {
            AGHAF_Device device = AGHAF_getDeviceBySN(productNumber.c_str(),
                                                      serialNumber.c_str());
            devices.insert(std::pair<std::pair<std::string, std::string>, AGHAF_Device>(key, device));
        }
            break;
        case AGHAF_Device_Leave: {
            auto it = devices.find(key);
            if (it != devices.end()) {
                devices.erase(it);
            }
        }
            break;
        case AGHAF_Device_Present:
            break;
        }
    }
        break;
    }
}
int main()
{
#ifdef _WIN32
    AGHAF_loadLibrary(AGHAF_DEFAULT_FILENAME);
#endif
    AGHAF_registerCallback(&deviceEvent_callback, nullptr);
    std::string input;
    while (input != "exit") {
        std::cin » input;
    }
    AGHAF_deregisterCallback(&deviceEvent_callback);
#ifdef _WIN32
    AGHAF_unloadLibrary();
#endif
    return 0;
}
```

8.4 DoCan_FD/main.cpp

```
#include <vector>
#include <iostream>
#define AGHAF_DYNAMIC_LIBRARY_INIT
#include <aghaf.h>
void doCanBus_1_callback(AGHAF_DoCAN_Event event, void *userContext)
{
    (void) userContext;
    AGHAF_DoCAN_Channel channel = AGHAF_DoCAN_Event_getChannel(event);
    AGHAF_EventInfo info;
    AGHAF_Status status = AGHAF_Event_getInfo(event, &info);
    if (status == AGHAF_STATUS_OK) {
        switch (info.type) {
        case AGHAF_TypeNoEvent:
        case AGHAF_TypeDeviceEvent:
        case AGHAF_TypeEthernetEvent:
        case AGHAF_TypeCanEvent:
            break;
        case AGHAF_TypeDoCanEvent: {
            AGHAF_DoCAN_EventType eventType = AGHAF_DoCAN_Event_type(info);
            switch (eventType) {
            case AGHAF_DoCAN_EVENT_MSGTX:
                std::cout « __FUNCTION__ « "AGHAF_DOCAN_EVENT_MSGTX" «
      reinterpret_cast<std::size_t>(channel) « std::endl;
```

```
            break;
        case AGHAF_DoCAN_EVENT_MSGTXERR: {
            std::cout « __FUNCTION__ « "AGHAF_DoCAN_EVENT_MSGTXERR" «
    reinterpret_cast<std::size_t>(channel) « std::endl;
            uint8_t error = AGHAF_DoCAN_Event_getError(event);
            std::cout « static_cast<int>(error) « std::endl;
        }
            break;
        case AGHAF_DoCAN_EVENT_MSGRX: {
            std::cout « __FUNCTION__ « "AGHAF_DoCAN_EVENT_MSGRX" «
    reinterpret_cast<std::size_t>(channel) « std::endl;
            uint32_t size = AGHAF_DoCAN_Event_getDataSize(event);
            uint8_t *data = new uint8_t[size];
            AGHAF_DoCAN_Event_getData(
                        event,
                        data,
                        &size);
            delete [] data;
        }
            break;
        case AGHAF_DoCAN_EVENT_MSGRXFF:
            std::cout « __FUNCTION__ « "AGHAF_DoCAN_MSGRXFF" « std::endl;
            break;
        case AGHAF_DoCAN_EVENT_MSGRXERR: {
            std::cout « __FUNCTION__ « "AGHAF_DoCAN_EVENT_MSGRXERR" «
    reinterpret_cast<std::size_t>(channel) « std::endl;
            uint8_t error = AGHAF_DoCAN_Event_getError(event);
            std::cout « static_cast<int>(error) « std::endl;
        }
            break;
        }
     }
        break;
     }
    }
}
void doCanBus_2_callback(AGHAF_CAN_Event event, void *userContext)
{
    (void) event;
    (void) userContext;
}
int main()
{
#ifdef _WIN32
    AGHAF_loadLibrary(AGHAF_DEFAULT_FILENAME);
#endif
    uint32_t count = AGHAF_getDeviceCount();
    std::cout « count « std::endl;
    if (count > 0) {
        AGHAF_DeviceInfo *devicesInfo = nullptr;
        uint32_t deviceInfoNumber = 0;
        AGHAF_getDeviceList(&devicesInfo, &deviceInfoNumber);
        AGHAF_Device device = AGHAF_getDeviceBySN(
                    devicesInfo[0].productNo,
                devicesInfo[0].serialNo);
        AGHAF_freeDeviceList(devicesInfo);
        AGHAF_openDevice(device);
        AGHAF_CAN_Bus canBus = AGHAF_CAN_getBus(device, 0);
        AGHAF_DoCAN_Bus doCanbus = AGHAF_DoCAN_getBus(device, 0);
        // BUS CAN FD -------------------------------
        AGHAF_CAN_Bus_deactivate(canBus);
        AGHAF_CAN_Bus_setParam(canBus,
                               CAN_PARAM_LISTEN_ONLY,
                               AGHAF_FALSE);
        AGHAF_CAN_Bus_setParam(canBus,
                               CAN_PARAM_MODE,
                               AGHAF_CAN_MODE_FD);
        AGHAF_CAN_Bus_setParam(canBus,
                               CAN_PARAM_BAUDRATE,
                               1000000);
        AGHAF_CAN_Bus_setParam(canBus,
                               CAN_PARAM_BAUDRATE_FD,
                               5000000);
        AGHAF_CAN_Bus_setParam(canBus,
                               CAN_PARAM_TERMINATION,
                               120);
        AGHAF_CAN_Bus_activate(canBus);
        // ------------------------------------------
        AGHAF_DoCAN_registerCallback(
                    doCanbus,
                    &doCanBus_1_callback,
                    nullptr);
        AGHAF_DoCAN_Channel channel = AGHAF_DoCAN_createChannel(doCanbus);
        AGHAF_DoCAN_Channel_setParam(channel, AGHAF_DoCAN_CHANNEL_PARAM_DIRECTION,
       AGHAF_DoCAN_CHANNEL_DIR_2WAY);
        AGHAF_DoCAN_Channel_setParam(channel, AGHAF_DoCAN_CHANNEL_PARAM_TXDL,
       AGHAF_DoCAN_TX_DL_CLASSIC_CAN);
```

```
        AGHAF_DoCAN_Channel_setParam(channel, AGHAF_DoCAN_CHANNEL_PARAM_PAD_HANDLING,
       AGHAF_DoCAN_PAD_ENABLE);
        AGHAF_DoCAN_Channel_setParam(channel, AGHAF_DoCAN_CHANNEL_PARAM_PAD_VALUE, 0x06);
        AGHAF_DoCAN_Channel_setParam(channel, AGHAF_DoCAN_CHANNEL_PARAM_ADDRMODE,
       AGHAF_DoCAN_ADDR_MODE_PHYS);
        AGHAF_DoCAN_Channel_setParam(channel, AGHAF_DoCAN_CHANNEL_PARAM_TIMING_AS, 1000);
        AGHAF_DoCAN_Channel_setParam(channel, AGHAF_DoCAN_CHANNEL_PARAM_TIMING_BS, 2500);
        AGHAF_DoCAN_Channel_setParam(channel, AGHAF_DoCAN_CHANNEL_PARAM_TIMING_CS, 0);
        AGHAF_DoCAN_Channel_setParam(channel, AGHAF_DoCAN_CHANNEL_PARAM_TIMING_AR, 1000);
        AGHAF_DoCAN_Channel_setParam(channel, AGHAF_DoCAN_CHANNEL_PARAM_TIMING_BR, 0);
        AGHAF_DoCAN_Channel_setParam(channel, AGHAF_DoCAN_CHANNEL_PARAM_TIMING_CR, 1000);
        AGHAF_DoCAN_Channel_setParam(channel, AGHAF_DoCAN_CHANNEL_PARAM_TIMING_STMIN, 4);
        AGHAF_DoCAN_Channel_setParam(channel, AGHAF_DoCAN_CHANNEL_PARAM_REQ_ID_FMT, AGHAF_DoCAN_ID_FMT_XTD);
        AGHAF_DoCAN_Channel_setParam(channel, AGHAF_DoCAN_CHANNEL_PARAM_REQ_ID_VALUE, 0x400);
        AGHAF_DoCAN_Channel_setParam(channel, AGHAF_DoCAN_CHANNEL_PARAM_REQ_ID_AE, 0);
        AGHAF_DoCAN_Channel_setParam(channel, AGHAF_DoCAN_CHANNEL_PARAM_RESP_ID_FMT,
       AGHAF_DoCAN_ID_FMT_XTD);
        AGHAF_DoCAN_Channel_setParam(channel, AGHAF_DoCAN_CHANNEL_PARAM_RESP_ID_VALUE, 0x900);
        AGHAF_DoCAN_Channel_setParam(channel, AGHAF_DoCAN_CHANNEL_PARAM_RESP_ID_AE, 0);
        AGHAF_DoCAN_Channel_setParam(channel, AGHAF_DoCAN_CHANNEL_PARAM_WAITFRAME, 0);
        AGHAF_DoCAN_Channel_setParam(channel, AGHAF_DoCAN_CHANNEL_PARAM_BLOCKSIZE, 0);
        AGHAF_DoCAN_Channel_setParam(channel, AGHAF_DoCAN_CHANNEL_PARAM_MAXDATALEN, 65537);
        AGHAF_DoCAN_Channel_start(channel);
        std::vector<uint8_t> data;
        data.push_back(0x01);
        AGHAF_DoCAN_Channel_send(
                    channel, static_cast<uint32_t>(data.size()), data.data());
        AGHAF_DoCAN_Channel_stop(channel);
        AGHAF_DoCAN_releaseChannel(channel);
        AGHAF_CAN_Bus_deactivate(canBus);
        AGHAF_DoCAN_deregisterCallback(doCanbus, &doCanBus_1_callback);
        AGHAF_closeDevice(device);
    }
#ifdef _WIN32
    AGHAF_unloadLibrary();
#endif
    return 0;
}
```

Index