FARO-MD700 / GADN-MD7L0

M.2 to Dual CAN FD / & GNSS Module

User Manual

Rev 1.0

Revision History

Revision	Date	Description
1.0	2023/12/05	Initial Release

Table of Contents

1. Introduction

We provide basic CAN 2.0B and CAN FD API for application programming in Windows and Linux.

The following table shows the corresponding model to these API which can be used.

Part Number	**Basic CAN 2.0B**	**CAN FD**
FARO-MD700	Yes	Yes
GADN-MD7L0	Yes	Yes

Factory Default Setting

Nominal Baud Rate	1M Bit/s
Data Baud Rate	2M Bit/s
Frame Format	Classical CAN CAN FD No Bit Rate **CAN FD Bit Rate (Default)**
Auto Retransmission	ON
Transmit Pause	ON
Error Update	OFF
ISO Mode	ON
Filter Type	**Range(Default)**/Dual/Mask
Filter ID	0
Filter ID Type	**Standard(Default)**/Extend

Supported Operation System

Windows	10(32/64bit)
Linux (cdc-acm driver)	Kernel 4.4 and above, 32/64bit
Linux (SocketCAN driver)	Kernel 4.4 and above, 32/64bit

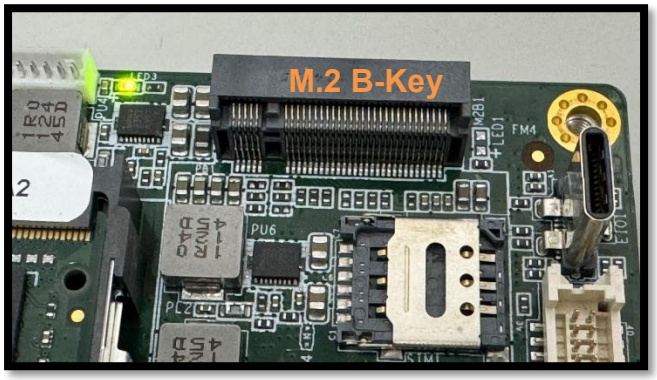
2. Hardware Installation

2.1. FARO-MD700/GADN-MD7L0

FARO-MD700/GADN-MD7L0 CANbus module uses USB 2.0 input interface, there are dual options to install the module.

2.1.1. M.2 B-Key Slot

Install the module to M.2 B-Key (or B-M Key) slot which has USB 2.0 interface.



2.1.2. Check the Switch for Terminal Resistor Function

Before we start, check the CANbus system or connected cable between CAN1/2 whether has 120 ohm terminal resistor or not. To turn on the pin 1 or 4 or even 1 and 4 pin for the terminal resistor function

You can find this setting as below table.

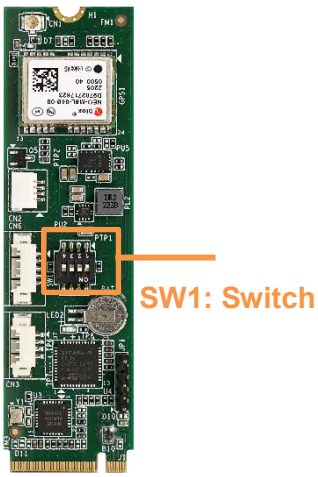


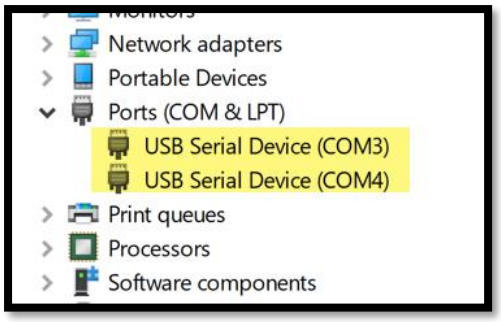
Table 15: Switch SMD 4Poles Switch Setting (SW1)

Pin #	Signal Name
1	CAN Port 1 Terminal Resistor (Default OFF)
2	NC (Default OFF)
3	CAN Port 0 TX On/Off (Default ON)
4	CAN Port 0 Terminal Resistor (Default OFF)

3. Windows OS

3.1. FARO-MD700/GADN-MD7L0 Driver Installation

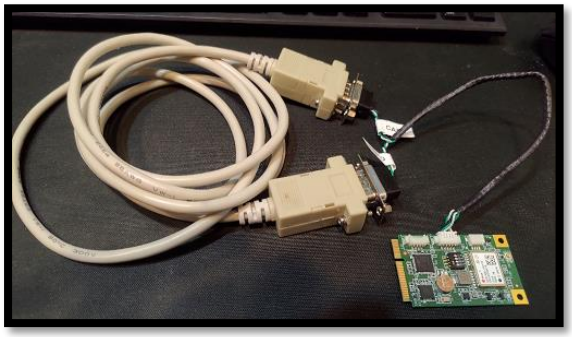
Install FARO-MD700/GADN-MD7L0 into mPCIe slot. Windows 10 inbox driver support module directly, no additional driver needed. USB Serial Device (COM3) is for CAN bus, and USB Serial Device (COM4) is for GNSS.



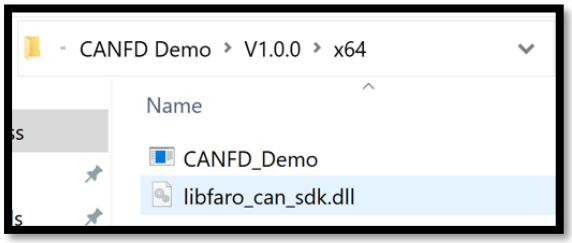
3.2. CANFD DEMO Tool

You can use this GUI utility to test FARO-MD700/GADN-MD7L0 for sending/receiving basic CAN frames. Here show you the loopback test that CAN port 0 and CAN Port 1 send data to each other.

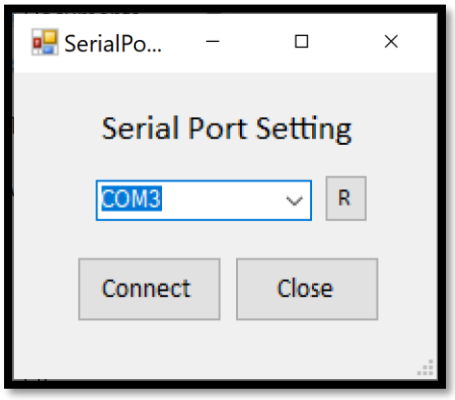
- First connect CAN port 0 and CAN Port 1.



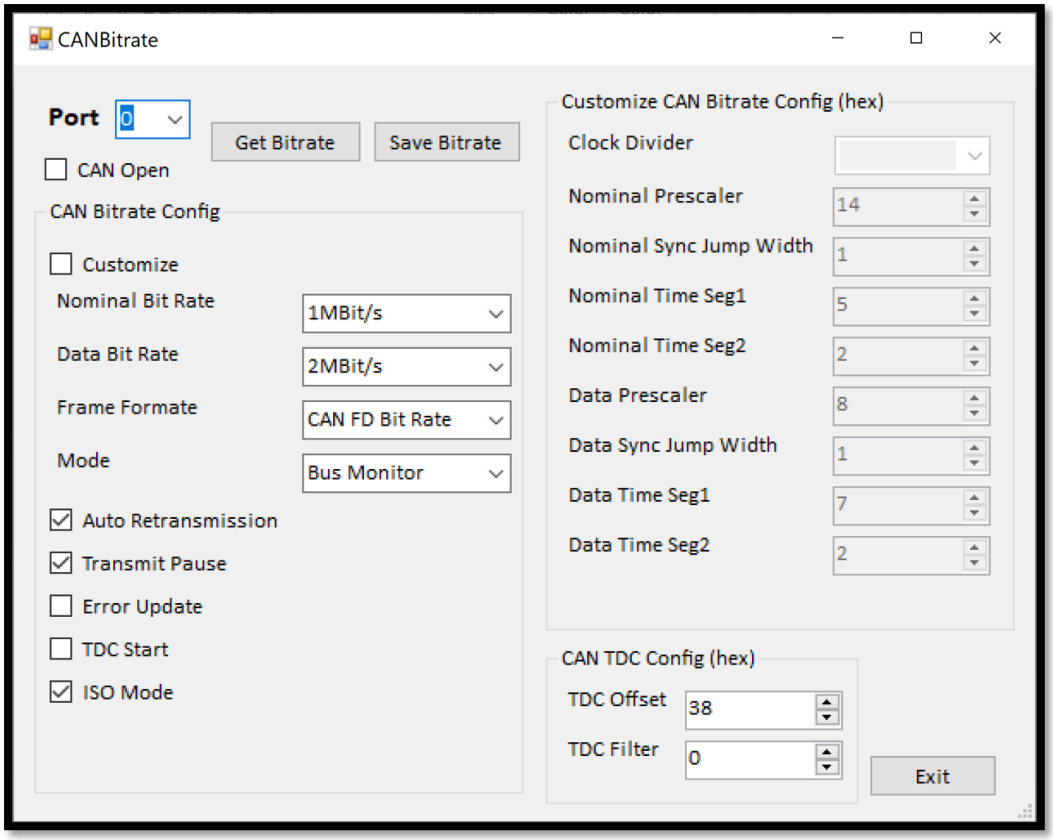
- Execute CANFD_DEMO



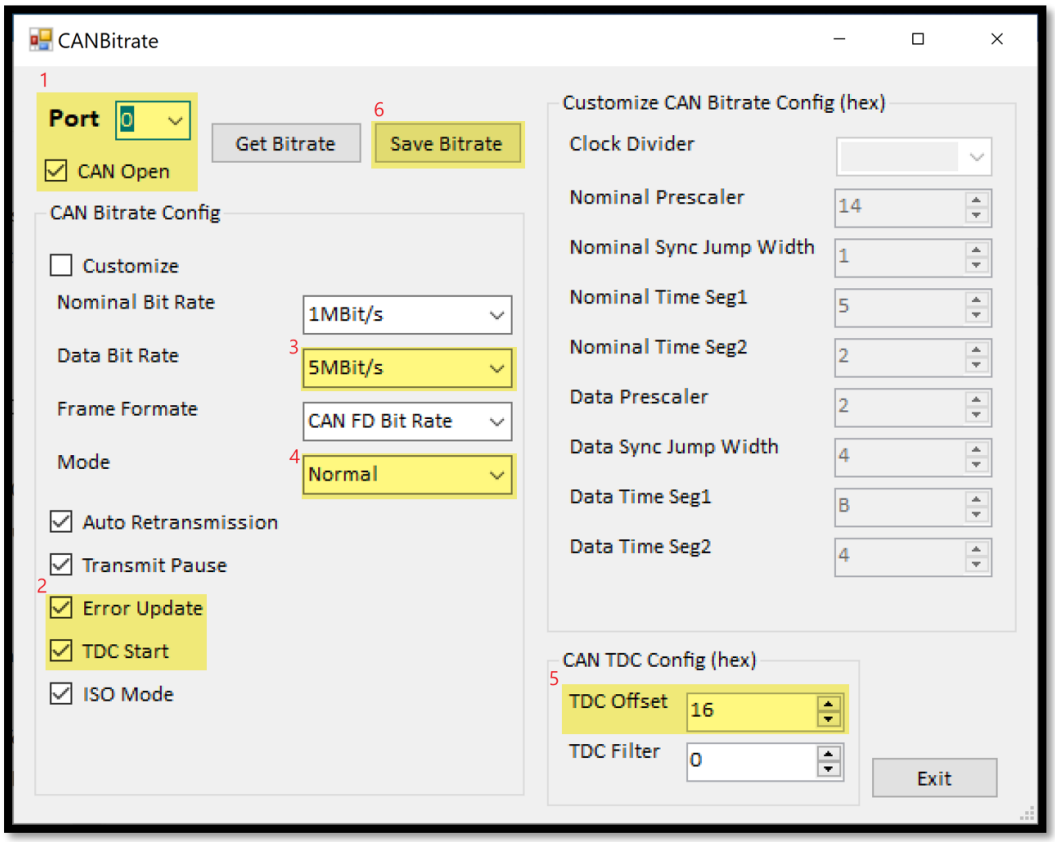
- Select COM3 and Connect



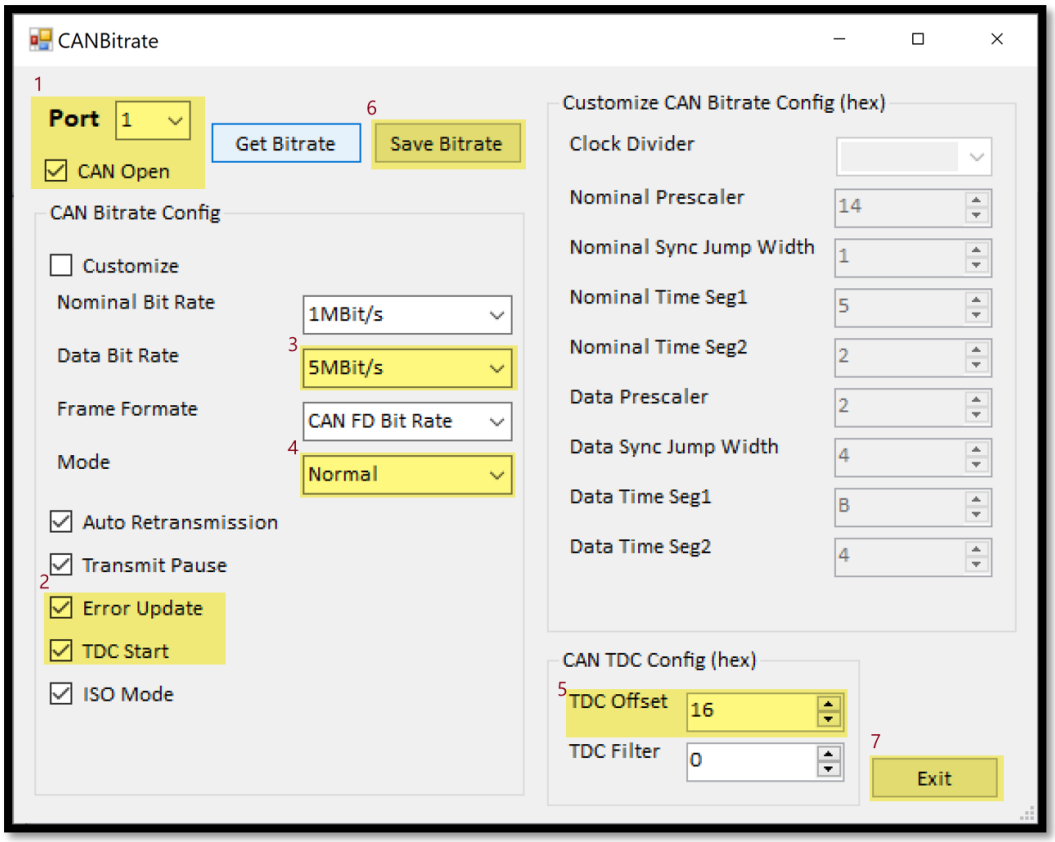
- This is CAN Bitrate setting page.



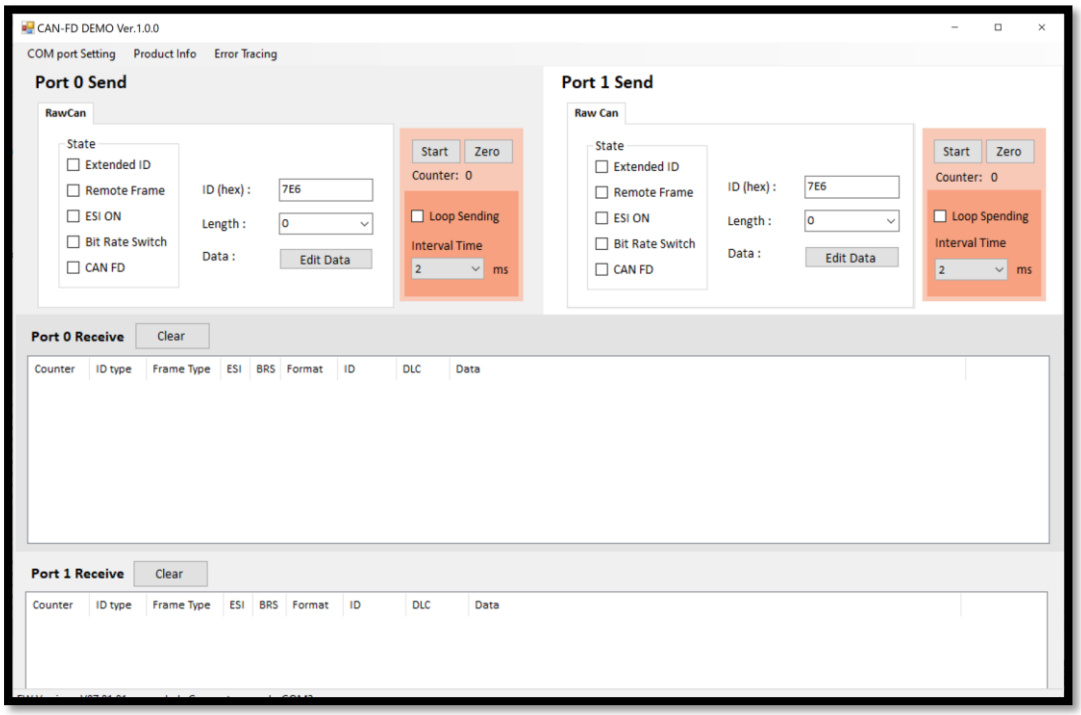
- Follow below settings step by step for CAN Port 0



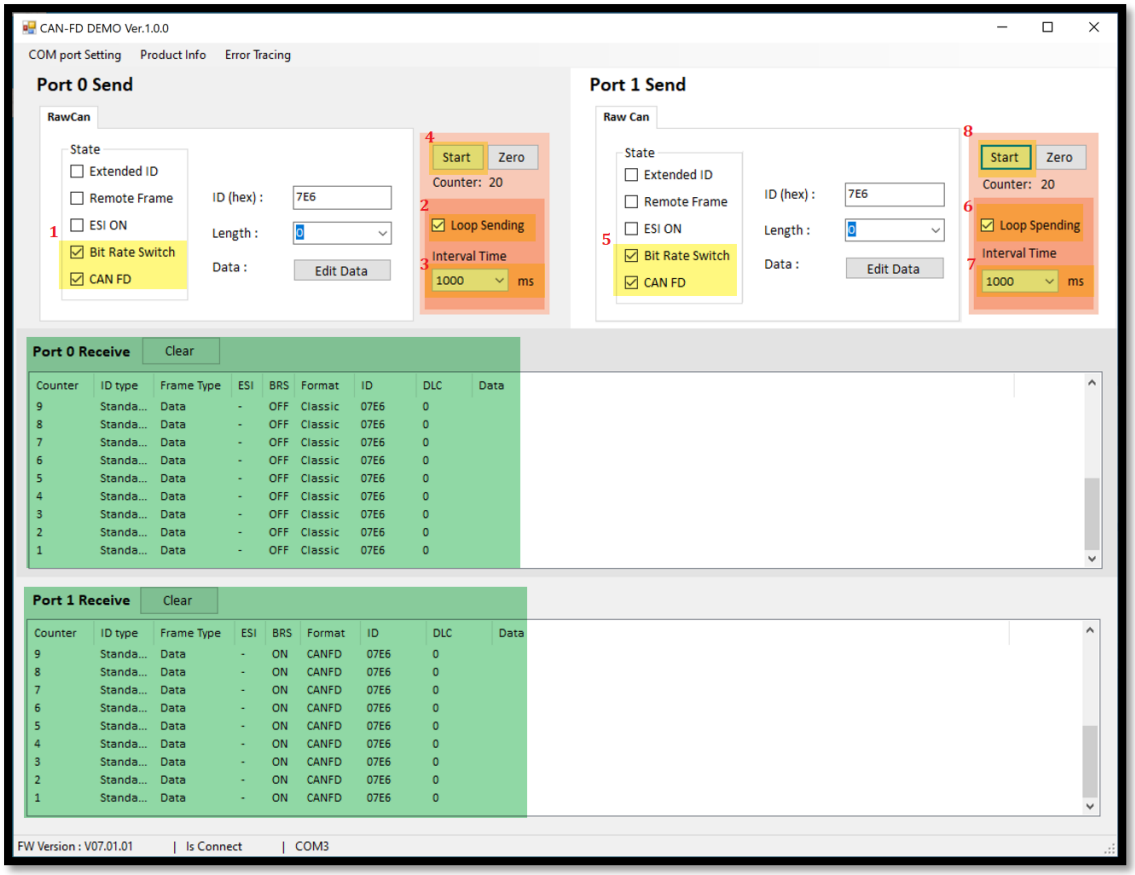
- Follow below settings step by step for CAN Port 1



- CAN-FD DEMO Tool Popup



- Follow below settings step by step, CAN port 0 and CAN Port 1 send data to each other, and you can find the test result in green area.



4. Linux OS

The following sections use Ubuntu 22.04.

4.1. Driver Installation

The device will be recognized as ttyACM% (%=0, 1…) by using CDC-ACM kernel driver.

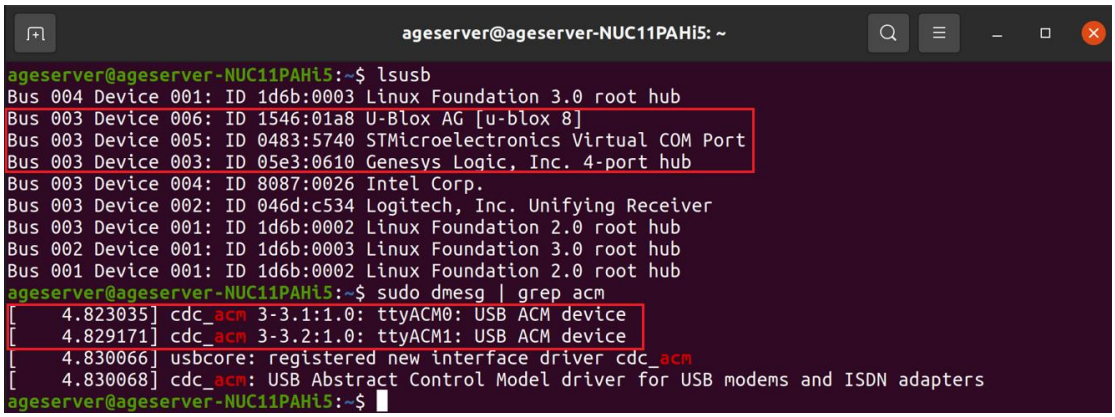
Note: *Linux kernel 2.6 and above have native CDC-ACM kernel driver. Some Linux OS may need to add CDC-ACM configuration manually in building process. In different Linux OS may have different tty name.*

Type command “lsusb” to see messages below.

There are U-blox AG, STMicroelectronics VCOM, and Genesys Logic 4-port hub.

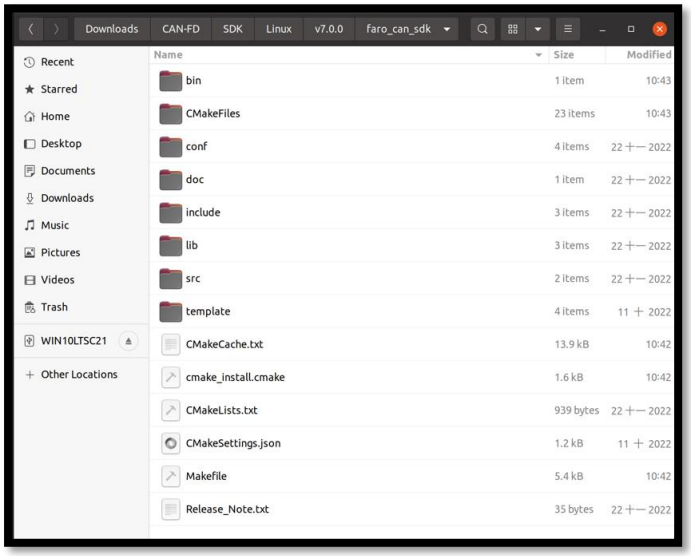
Type command “sudo dmesg | grep acm” to see messages below.

Generally the name would be ttyACM0 or ttyACM1 in Linux.



4.2. CAN-FD SDK

- Enter in faro_can_sdk folder

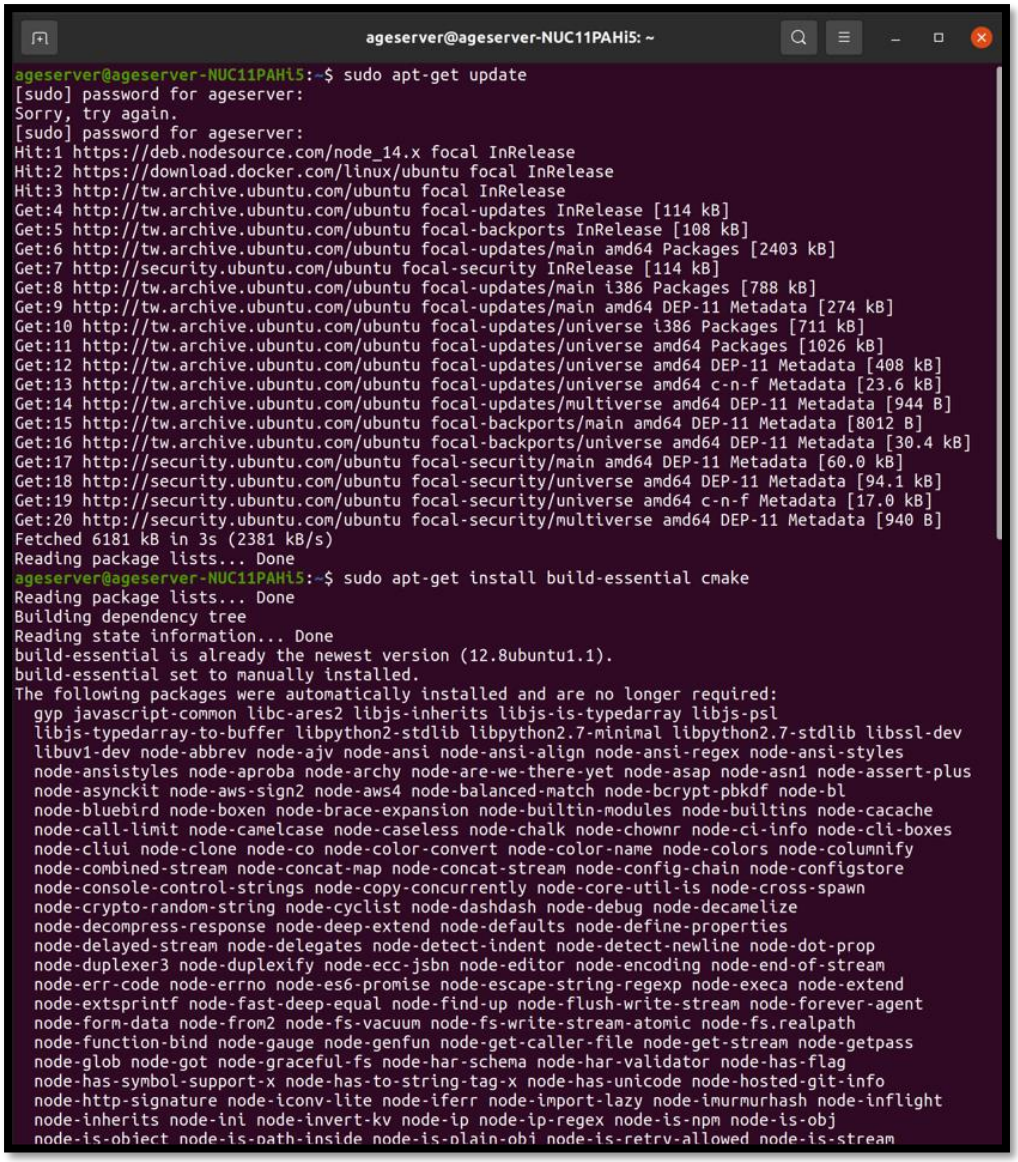


- Open terminal and install build-essential and cmake

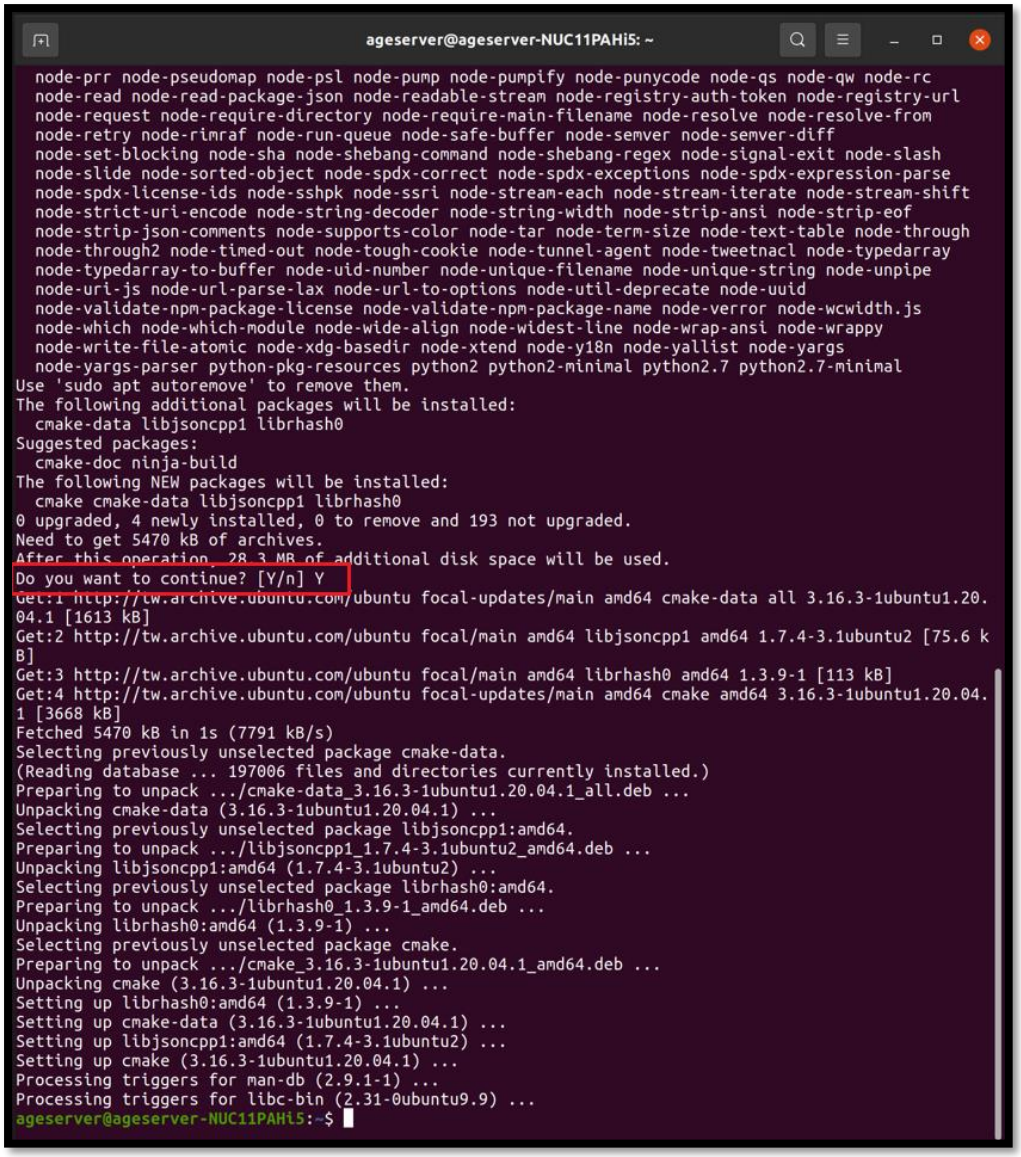
  Command:

  #sudo apt-get update

  #sudo apt-get install build-essential and cmake



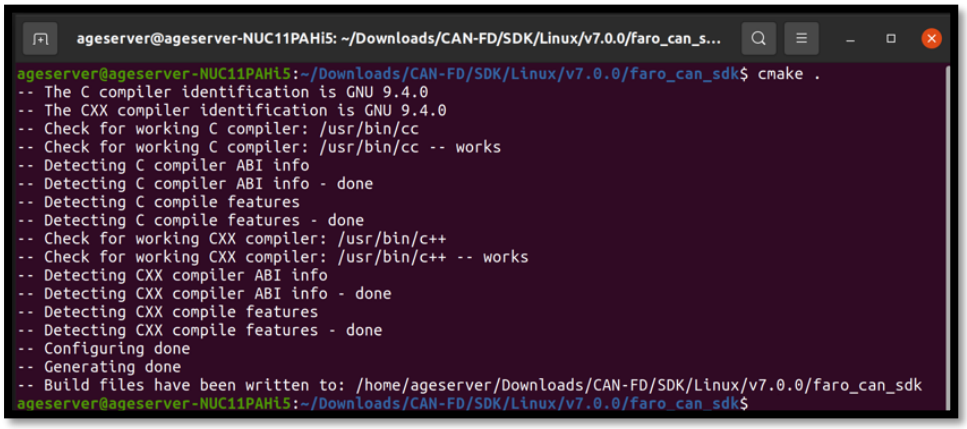
- Press Y and enter



- Use cmake to generate makefile

  Command:

  #cmake .



- Use make to build antzer_faro_can_fd

  Command:

  #make

```
ageserver@ageserver-NUC11PAHi5:~/Downloads/CAN-FD/SDK/Linux/v7.0.0/faro_can_sdk$ make
Scanning dependencies of target antzer_faro_can_fd
[ 50%] Building C object CMakeFiles/antzer_faro_can_fd.dir/src/antzer_faro_can_fd.c.o
[100%] Linking C executable bin/antzer_faro_can_fd
[100%] Built target antzer_faro_can_fd
ageserver@ageserver-NUC11PAHi5:~/Downloads/CAN-FD/SDK/Linux/v7.0.0/faro_can_sdk$
```

- Execute antzer_faro_can_fd to do loopback test for CAN Port 1 & CAN Port 2

  Command:

  #sudo ./bin/ antzer_faro_can_fd

  Press Ctrl + C to stop the loopback test and show the test result.

```
ageserver@ageserver-NUC11PAHi5:~/Downloads/CAN-FD/SDK/Linux/v7.0.0/faro_can_sdk$ sudo ./bin/antzer_fa
ro_can_fd
[sudo] password for ageserver:
Testing port = /dev/ttyACM0
***** Adopting port = /dev/ttyACM0, baudrate = B115200
Get firmware version : v07.01.01
send CAN Port0 cnt: 1, CAN ID:18FEC100
send CAN Port1 cnt: 1, CAN ID:CF00400
rec CAN port1 Frame Cnt:1, ID:18FEC100, Port:1, FD:8, BRS:10
Data: 00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f
 20 21 22 23 24 25 26 27 28 29 2a 2b 2c 2d 2e 2f 30 31 32 33 34 35 36 37 38 39 3a 3b 3c 3d 3e 3f
rec CAN port0 Frame Cnt:1, ID:CF00400, Port:0, FD:8, BRS:10
Data: 00 03 06 09 0c 0f 12 15 18 1b 1e 21 24 27 2a 2d 30 33 36 39 3c 3f 42 45 48 4b 4e 51 54 57 5a 5d
 60 63 66 69 6c 6f 72 75 78 7b 7e 81 84 87 8a 8d 90 93 96 99 9c 9f a2 a5 a8 ab ae b1 b4 b7 ba bd
send CAN Port0 cnt: 2, CAN ID:18FEC100
send CAN Port1 cnt: 2, CAN ID:CF00400
rec CAN port1 Frame Cnt:2, ID:18FEC100, Port:1, FD:8, BRS:10
Data: 00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f
 20 21 22 23 24 25 26 27 28 29 2a 2b 2c 2d 2e 2f 30 31 32 33 34 35 36 37 38 39 3a 3b 3c 3d 3e 3f
rec CAN port0 Frame Cnt:2, ID:CF00400, Port:0, FD:8, BRS:10
Data: 00 03 06 09 0c 0f 12 15 18 1b 1e 21 24 27 2a 2d 30 33 36 39 3c 3f 42 45 48 4b 4e 51 54 57 5a 5d
 60 63 66 69 6c 6f 72 75 78 7b 7e 81 84 87 8a 8d 90 93 96 99 9c 9f a2 a5 a8 ab ae b1 b4 b7 ba bd
send CAN Port0 cnt: 3, CAN ID:18FEC100
send CAN Port1 cnt: 3, CAN ID:CF00400
rec CAN port1 Frame Cnt:3, ID:18FEC100, Port:1, FD:8, BRS:10
Data: 00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f
 20 21 22 23 24 25 26 27 28 29 2a 2b 2c 2d 2e 2f 30 31 32 33 34 35 36 37 38 39 3a 3b 3c 3d 3e 3f
rec CAN port0 Frame Cnt:3, ID:CF00400, Port:0, FD:8, BRS:10
Data: 00 03 06 09 0c 0f 12 15 18 1b 1e 21 24 27 2a 2d 30 33 36 39 3c 3f 42 45 48 4b 4e 51 54 57 5a 5d
 60 63 66 69 6c 6f 72 75 78 7b 7e 81 84 87 8a 8d 90 93 96 99 9c 9f a2 a5 a8 ab ae b1 b4 b7 ba bd
send CAN Port0 cnt: 4, CAN ID:18FEC100
send CAN Port1 cnt: 4, CAN ID:CF00400
rec CAN port1 Frame Cnt:4, ID:18FEC100, Port:1, FD:8, BRS:10
Data: 00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f
 20 21 22 23 24 25 26 27 28 29 2a 2b 2c 2d 2e 2f 30 31 32 33 34 35 36 37 38 39 3a 3b 3c 3d 3e 3f
rec CAN port0 Frame Cnt:4, ID:CF00400, Port:0, FD:8, BRS:10
Data: 00 03 06 09 0c 0f 12 15 18 1b 1e 21 24 27 2a 2d 30 33 36 39 3c 3f 42 45 48 4b 4e 51 54 57 5a 5d
 60 63 66 69 6c 6f 72 75 78 7b 7e 81 84 87 8a 8d 90 93 96 99 9c 9f a2 a5 a8 ab ae b1 b4 b7 ba bd
send CAN Port0 cnt: 5, CAN ID:18FEC100
send CAN Port1 cnt: 5, CAN ID:CF00400
rec CAN port1 Frame Cnt:5, ID:18FEC100, Port:1, FD:8, BRS:10
Data: 00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f
 20 21 22 23 24 25 26 27 28 29 2a 2b 2c 2d 2e 2f 30 31 32 33 34 35 36 37 38 39 3a 3b 3c 3d 3e 3f
rec CAN port0 Frame Cnt:5, ID:CF00400, Port:0, FD:8, BRS:10
Data: 00 03 06 09 0c 0f 12 15 18 1b 1e 21 24 27 2a 2d 30 33 36 39 3c 3f 42 45 48 4b 4e 51 54 57 5a 5d
 60 63 66 69 6c 6f 72 75 78 7b 7e 81 84 87 8a 8d 90 93 96 99 9c 9f a2 a5 a8 ab ae b1 b4 b7 ba bd
^Ccatch SIGINT
CAN port0 send: 5, rec: 5
CAN port1 send: 5, rec: 5
ageserver@ageserver-NUC11PAHi5:~/Downloads/CAN-FD/SDK/Linux/v7.0.0/faro_can_sdk$
```

5. Software API

Faro-FD SDK is based on a dynamic library (DLL) in Windows and shared library (.so) in Linux to control Faro-FD.

5.1. Basic Function Description

This chapter describes basic Faro-FD SDK functions and parameters.

Header file (faro_can_sdk.h) includes declaration and data structure requested for programming.

5.1.1. int AZ_VC_Init (void)

Description: Initialize and setup SDK.

Member:

None.

Return Status Code:

Value	Description
0	Success
!= 0	Error

5.1.2. int AZ_VC_Init_SetBuf (int bufsize)

Description: Initialize and setup SDK.

Member:

bufsize: [input] Internal buffer multiply by a multiple of parameter. The parameter must >= 1.

Return Status Code:

Value	Description
0	Success
!= 0	Error

5.1.3. int AZ_VC_DeInit (void)

Description: De-Initialize SDK.

Member:

None.

Return Status Code:

Value	Description
0	Success
!= 0	Error

5.1.4. int AZ_VC_Open_Port (char *lpszDevice)

Description: Serial port open.

Member:

lpszDevice: [input] serial port name.

Example:

Windows – COM1

Linux – /dev/ttyACM0

Return Status Code:

Value	Description
0	Success
!= 0	Error

5.1.5. int AZ_VC_Close_Port (void)

Description: Serial port close.

Member:

None.

Return Status Code:

Value	Description
0	Success
!= 0	Error

5.1.6. int AZ_VC_GetFWVer (struct can_fw_ver_t *canfwver)

Description: Get FaroFD CAN FD module firmware version.

Member:

canfwver: [output] struct can_fw_ver_t.

```
struct can_fw_ver_t {
    uint8_t main;        // major version
```

```
    uint8_t minor;      // minor verion
    uint8_t type;       // build version
    uint8_t year;       // release year
    uint8_t week;       // release week
};
```

Return Status Code:

Value	Description
0	Success
!= 0	Error

5.1.7. int AZ_VC_GetSDKVer(struct can_sdk_ver_t* cansdkver)

Description: Get FaroFD CAN FD module SDK version.

Member:

cansdkver: [output] struct can_sdk_ver_t.

```
struct can_fw_ver_t {
    uint8_t main;       // major version
    uint8_t minor;      // minor version
    uint8_t patch;      // build version
};
```

Return Status Code:

Value	Description
0	Success
!= 0	Error

5.1.8. int AZ_VC_CAN_Baudrate_Set(struct can_config_t canCFG)

Description: Set CAN FD baudrate.

Member:

canCFG: [input] struct can_config_t.

```
struct can_config_t {
    enum can_port_no Port;          // 0 or 1
    enum can_flag_switch CustomizeSetFlag;    // open customize bit rate or not
    enum can_speed NominalBR;   // can_speed from 125kHz to 1MHz
    enum can_speed DataBR;      // can_speed from 2MHz to 5MHz
    enum can_frame_type FrameFormat;
    enum can_mode_type Mode;
```

```
        enum can_flag_switch AutoRetransmission;
        enum can_flag_switch TransmitPause;
        enum can_br_div ClockDivider;
        uint16_t NominalPrescaler;
        uint8_t NominalSyncJumpWidth;
        uint16_t NominalTimeSeg1;
        uint8_t NominalTimeSeg2;
        uint8_t DataPrescaler;
        uint8_t DataSyncJumpWidth;
        uint8_t DataTimeSeg1;
        uint8_t DataTimeSeg2;
        enum can_flag_switch ErrorUpdateEn;
        enum can_flag_switch TdcStart;
        uint16_t TdcOffset;
        uint16_t TdcFilter;
        enum can_flag_switch IsoModeEn;
        enum can_flag_switch CANEn; // have to enable it before transmit
};

enum can_port_no {
        CAN_CAN1 = 0,     // port 0
        CAN_CAN2,  // port 1
        CAN_CAN_MAX
};

enum can_flag_switch {
        CAN_FLAG_DISABLE = 0,
        CAN_FLAG_ENABLE,
        CAN_FLAG_MAX
};

enum can_speed {
        CAN_SPEED_125KHZ = 0,
        CAN_SPEED_250KHZ,
        CAN_SPEED_500KHZ,
        CAN_SPEED_800KHZ,
        CAN_SPEED_1MHZ,
        CAN_SPEED_2MHZ,
```

```
        CAN_SPEED_4MHZ,
        CAN_SPEED_5MHZ,
        CAN_SPEED_BPS_MAX
};

enum can_frame_type {
    CAN_FRAME_CLASSIC = 0, // Classic CAN frame
    CAN_FRAME_FD_NO_BRS, // CAN FD frame, Bit Rate Switch Off
    CAN_FRAME_FD_BRS, // CAN FD frame, Bit Rate Switch On
    CAN_FRAME_MAX
};

enum can_mode_type {
    CAN_MODE_NORMAL = 0,
    CAN_MODE_BUS_MONITOR = 2, // Listening mode
    CAN_MODE_EXTERNAL_LOOPBACK = 4,
    CAN_MODE_MAX
};

enum can_br_div {
    CAN_CLOCK_DIV1 = 1, // 160 / 1 = 160
    CAN_CLOCK_DIV2 = 2, // 160 / 2 = 80
    CAN_CLOCK_DIV4 = 4, // 160 / 4 = 40
    CAN_CLOCK_MAX
};
```

Return Status Code:

Value	Description
0	Success
!= 0	Error

5.1.9. int AZ_VC_CAN_Baudrate_Get(uint8_t port, struct can_config_t* canCFG)

Description: Get CAN FD baudrate.

Member:

port: [input] 0 = CAN1, 1 = CAN2.

canCFG: [output] struct can_config_t. (description see *6.1.8*)

Return Status Code:

Value	Description
0	Success
!= 0	Error

5.1.10. int AZ_VC_CAN_Error_Amount(void)

Description: Check the amount of CAN FD error received.

Member:

None.

Return Status Code:

Value	Description
0	No error occurs.
> 0	The number that error occurs.

5.1.11. int AZ_VC_CAN_Error(struct error_data_t* errordata)

Description: Get CAN FD received error state.

Member:

errordata: [output] struct error_data_t.

```
struct error_data_t {
    enum can_error_data_type port0_error; // port 0 error state
    enum can_error_data_type port1_error; // port 1 error state
    enum can_error_ITtype port0_ITerror; // port 0 error IT state
    enum can_error_ITtype port1_ITerror; // port 1 error IT state
};

enum can_error_data_type {
    CAN_ERROR_NO_ERROR = 0x00000000,
    CAN_ERROR_TIMEOUT_ERROR = 0x00000001,
    CAN_ERROR_PERIPHERAL_NOT_INITIALIZED = 0x00000002,
    CAN_ERROR_PERIPHERAL_NOT_READY = 0x00000004,
    CAN_ERROR_PERIPHERAL_NOT_STARTED = 0x00000008,
    CAN_ERROR_MODE_NOT_SUPPORTED = 0x00000010,
    CAN_ERROR_PARAMETER_ERROR = 0x00000020,
    CAN_ERROR_PENDING_OPERATION = 0x00000040,
    CAN_ERROR_MESSAGE_RAM_ACCESS_FAILURE = 0x00000080,
    CAN_ERROR_PUT_ELEMENT_IN_FULL_FIFO = 0x00000100,
```

```
        CAN_ERROR_GET_ELEMENT_FROM_EMPTY_FIFO = 0x00000200,
        CAN_ERROR_OVERFLOW_OF_CAN_ERROR_LOGGING_COUNTER = 0x00010000,
        CAN_ERROR_MESSAGE_RAM_WATCHDOG_EVENT_OCCURRED = 0x00100000,
        CAN_ERROR_PROTOCOL_ERROR_IN_ARBITRATION_PHASE = 0x00200000,
        CAN_ERROR_PROTOCOL_ERROR_IN_DATA_PHASE        = 0x00400000,
        CAN_ERROR_ACCESS_TO_RESERVED_ADDRESS = 0x00800000,
};

enum can_error_ITtype {
        CAN_IT_ERROR_NO_ERROR     = 0x00000000,
        CAN_IT_ERROR_PASSIVE      = 0x00020000,
        CAN_IT_ERROR_WARNING      = 0x00040000,
        CAN_IT_ERROR_BUS_OFF      = 0x00080000,
};
```

Return Status Code:

Value	Description
0	Success
!= 0	Error

5.1.12. int AZ_VC_CAN_Read_Amount(void)

Description: Check the amount of CAN FD data received.

Member:

None.

Return Code:

Value	Description
0	No data occur.
> 0	The number that data occurs.

5.1.13. int AZ_VC_CAN_Read(struct can_message_t *candata, uint32_t *dwread)

Description: The SDK support message queue method to read CAN FD data.

Member:

candata: [output] struct can_message_t.

dwread: [output] Actual reading bytes.

```
struct can_message_t {
```

```
    uint8_t TxRxState;
    // bit 7 : IdType – 0 is standard ID, 1 is extended ID
    // bit 6 : FrameType – 0 is data frame, 1 is remote frame
    // bit 5 : ErrorStateIndicator – 0 is active, 1 is passive
    // bit 4 : BitRateSwitch – 0 is OFF, 1 is ON
    // bit 3 : FDFormat – 0 is Classic CAN, 1 is FA CAN
    // bit 2 : reserved 0
    // bit 1 : reserved 0
    // bit 0 : Port – 0 is port 0, 1 is port 1
    uint32_t Id; // 11-bit if IdType is standard, 29-bit if IdType is extended
    uint8_t Length; // 64, 48, 32, 24, 20, 16, 12, 8, 7, 6, 5, 4, 3, 2, 1, 0
    uint8_t data[64];
};
```

Return Code:

Value	Description
0	Success
!= 0	Error

5.1.14. int AZ_VC_CAN_Write(struct can_message_t candata)

Description: Write raw CAN FD data to CAN FD controller.

Member:

candata: [input] struct can_message_t. (description see *6.1.13*)

Return Code:

Value	Description
0	Success
!= 0	Error

5.1.15. int AZ_VC_CAN_Filter_Set(struct can_filter_config_t canFilterCFG)

Description: Set CAN FD controller filters configuration.

Member:

canFIlterCFG: [input] struct can_filter_config_t.

```
struct can_filter_config_t {
    enum can_port_no Port; // (description see 6.1.8)
    uint8_t Index;    //std:0~13, ext:0~7
    enum can_filter_type FilterType;
```

```
    enum can_id_type IdType;
    enum can_filter_cfg FilterConfig;
    uint32_t Id1;
    uint32_t Id2;
};

enum can_filter_type {
    CAN_FILTER_RANGE = 0, //Range filter from FilterID1 to FilterID2
    CAN_FILTER_DUAL = 1, // Dual ID filter for FilterID1 or FilterID2
    CAN_FILTER_MASK = 2, // Classic filter: FilterID1 = filter, FilterID2 = mask  CAN_FILTER_TYPE_MAX
};

enum can_id_type {
   CAN_ID_STD = 0, // standard ID
   CAN_ID_EXT, // extended ID
   CAN_ID_TYPE_MAX
};

enum can_filter_cfg {
   CAN_FILTER_CFG_DISABLE = 0,
   CAN_FILTER_CFG_ENABLE,
   CAN_FILTER_CFG_MAX
} ;
```

Return Status Code:

Value	Description
0	Success
!= 0	Error

5.1.16. int AZ_VC_CAN_Filter_Get(struct can_filter_config_t *canFilterCFG)

Description: Get CAN FD controller filters configuration.

Member:

canFilterCFG: [input / output] struct can_filter_config_t. (description see *6.1.15*)

```
struct can_filter_config_t {
    enum can_port_no Port; // input
    uint8_t Index; // input
    enum can_filter_type FilterType; //output
    enum can_id_type IdType; // input
```

```
enum can_filter_cfg FilterConfig; //output
uint32_t Id1; // output
uint32_t Id2; // output
};
```

Return Status Code:

Value	Description
0	Success
!= 0	Error

5.1.17. int AZ_VC_CAN_Product_Info_Read(struct product_information* canProductInfo)

Description: Get Product Information.

Member:

canProductInfo: [output] struct product_information.

```
struct product_information {
    uint8_t BomName[30];
    uint8_t PnNumber[20];
    uint8_t SnNmuber[17];
};
```

Return Status Code:

Value	Description
0	Success
!= 0	Error

5.1.18. int AZ_VC_Save_All_Config(uint8_t mode)

Description: Save or reset all config (baudrate & filter).

Member:

mode: [input] 1 - Store system configuration, 2 – Factory reset.

Return Status Code:

Value	Description
0	Success
!= 0	Error

5.1.19. int AZ_VC_Reboot_Module(void)

Description: Reboot module.

Member:

None.

Return Status Code:

Value	Description
0	Success
!= 0	Error

Contact us

Headquarters (Taiwan)

5F., No. 237, Sec. 1, Datong Rd., Xizhi Dist., New Taipei City 221, Taiwan

Tel: +886-2-77033000

Email: sales@antzer-tech.com

www.antzer-tech.com

December 5, 2023