AX58400 GPIO / AIO User Guide

Revision 2.20
December 20th, 2024

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

Revision History

Revision	Date	Description
1.00	2021/12/08	Initial release
1.10	2023/10/17	1. Added Evaluation Board License Agreement.
2.00	2024/04/10	1. Moved the content of Chapters 4 and 6 to the "**AX58x00 TwinCAT UserGuide**" document. 2. Moved the content of Chapter 9 to the "**AX58400 Porting SSC Stack UserGuide**" document. 3. Added a new chapter that describes how to set the external clock. 4. Added the CM4 firmware upgrade in Chapter 5. 5. Added explanations for CM7 and CM4 console in Section 5-4.
2.10	2024/11/28	1. Added STM32CubeMX required in Section 2. 2. Added AX58400 pinout patch description in Section 4-1.
2.20	2024/12/20	1. Added Reference Design building flow in Chapter 5. 2. Adjust Chapter 4 & 6 description. 3. Added disabling "**Debug Description**" in Section 6-2. 4. Added Chapter 14 for Firmware Architecture description.

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

CONTENT

1. Introduction 7

2. Requirements 8

3. Board Description 10

3-1 AX58400-TSB 10
3-2 AX58400-EXB-ADIO 11

4. Environment Setup 12

4-1 Hardware Connectivity 12
4-2 AX58400 Pinout Patch for STM32CubeMX 13

5. Building Reference Design Begin with STM32CubeMX 14

5-1 Create New Project in STM32CubeMX 14
5-2 Modify Peripheral Settings in STM32CubeMX 15
5-3 Generate Source Code in STM32CubeMX 21
5-4 Source Code Adjustment 23
5-5 Generate EtherCAT Slave Stack Code 29
5-6 IDE Setting Adjustment 30

6. Build Up and Download Firmware 35

6-1 Build up Firmware Binary 35
6-2 Download Bootloader Firmware 36
6-3 Download Application Firmware 37

7. Basic Operation of TwinCAT 39

8. PLC Application 40

9. Flash Memory Allocation and FoE Upgrading 41

9-1 FoE Firmware Upgrade 42

10. ADC / DAC Description 43

11. Switch ID Description 44

12. Shortest EtherCAT Cycle Evaluation 53

13. Using the High Speed Clock (HSE) 55

13-1 Modify RCC Setting 55

14. Firmware Architecture 56

14-1 Dual Core Shared Modules 56
14-2 CM7 Core Modules 58
14-3 CM4 Core Modules 58

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

List of Figure

Figure 2-1 8
Figure 2-2 8
Figure 2-3 8
Figure 2-4 9
Figure 2-5 9
Figure 2-6 9
Figure 2-7 9
Figure 3-1 10
Figure 3-2 11
Figure 4-1 12
Figure 4-2 13
Figure 5-1 14
Figure 5-2 14
Figure 5-3 14
Figure 5-4 15
Figure 5-5 15
Figure 5-6 16
Figure 5-7 16
Figure 5-8 16
Figure 5-9 17
Figure 5-10 17
Figure 5-11 17
Figure 5-12 18
Figure 5-13 18
Figure 5-14 19
Figure 5-15 19
Figure 5-16 20
Figure 5-17 21
Figure 5-18 21
Figure 5-19 22
Figure 5-20 23
Figure 5-21 23
Figure 5-22 24
Figure 5-23 24
Figure 5-24 25
Figure 5-25 25
Figure 5-26 26
Figure 5-27 26
Figure 5-28 27
Figure 5-29 27
Figure 5-30 28
Figure 5-31 28
Figure 5-32 30
Figure 5-33 30
Figure 5-34 30
Figure 5-35 31
Figure 5-36 31
Figure 5-37 32
Figure 5-38 32
Figure 5-39 33
Figure 5-40 33
Figure 5-41 33
Figure 5-42 34
Figure 5-43 34
Figure 6-1 35
Figure 6-2 35

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

Figure 6-3 36
Figure 6-4 37
Figure 6-4 37
Figure 6-5 37
Figure 6-6 38
Figure 9-1 41
Figure 9-2 42
Figure 9-3 42
Figure 9-4 42
Figure 10-1 43
Figure 11-1 44
Figure 11-2 44
Figure 11-3 45
Figure 11-4 45
Figure 11-5 46
Figure 11-6 46
Figure 11-7 47
Figure 11-8 47
Figure 11-9 48
Figure 11-10 48
Figure 11-11 49
Figure 11-12 50
Figure 11-13 50
Figure 11-14 51
Figure 11-15 52
Figure 12-1 53
Figure 12-2 53
Figure 12-3 54
Figure 13-1 55
Figure 13-2 55
Figure 14-1 56

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

List of Table

Table 4-1 12
Table 9-1 41
Table 14-1 57
Table 14-2 57

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

1. Introduction

This document introduces how to setup entire environment for GPIO / AIO application on AX58400 evaluation platform (TSB + EXB-ADIO boards). The demo application of EtherCAT master side is developed using PLC (Programmable Logic Controller) language running on BECKHOFF TwinCAT engineering (XAE). The IDE relies on Microsoft Visual Studio 2010 / 2013. We can execute the application through the IDE or pure BECKHOFF runtime package (XAR). The firmware part of EtherCAT slave side is developed using C language through ARM KEIL MDK (Microcontroller Development Kit).

The firmware mainly handles the tasks below:
- BECKHOFF SSC (Slave Stack Code)
- Hardware access of SSC through SPI interface at MCU side
- Simple object form for CIA-401 digital/analog IO
- Peripherals (GPIO / ADC / DAC) accessed by EtherCAT PD (Process Data)

The PLC at EtherCAT master side mainly handles the tasks below:
- PD mapping / handling by PLC for GPIO / AIO
- A demo GUI

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

2. Requirements

Before starting to setup the environment, the user should prepare some software and hardware. Of course, they are all running based on general desktop PC with windows 10 / 11 operation systems. The software and hardware requirements are listed as following:

[Software at EtherCAT master side]

- Windows 10 / 11 operation system
- Microsoft Visual Studio 2010 / 2013
- Beckhoff TwinCAT XAE v3.1.4022.28 or later
  You can download the BECKHOFF TwinCAT here

Figure 2-1

[Software at EtherCAT slave side]

- STM32CubeMX version 6.12.1 (for development only)
  You can download the STM32CubeMX here.

Figure 2-2

- In this reference design, two types of Integration Development Environment (IDE) could be supported, you can choose one for the development.

  - ARM KEIL MDK micro-Vision 5 or later (for development only)
    Notes: If you need to download firmware or enter dual-cores debug mode on KEIL MDK IDE, please ensure the version is v5.25 or later.

Figure 2-3

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

- STM32 CubeIDE version 1.15.0 or later (for development only)
  You can download the STM32CubeIDE here.

Figure 2-4

- STM32CubeProgrammer
  You can download the STM32CubeProgrammer here.

Figure 2-5

- SSC Tool V5.13 (for development only)
  You can download the SSC Tool from EtherCAT Technology Group (ETG) here

Figure 2-6

[Hardware]

- AX58400 Test Board (AX58400-TSB) x 1
  For more details, please refer to the link below and the following chapter.
  ASIX official link: https://www.asix.com.tw/

- STLINK-V3MINI programmer x 1
  You can download the driver here

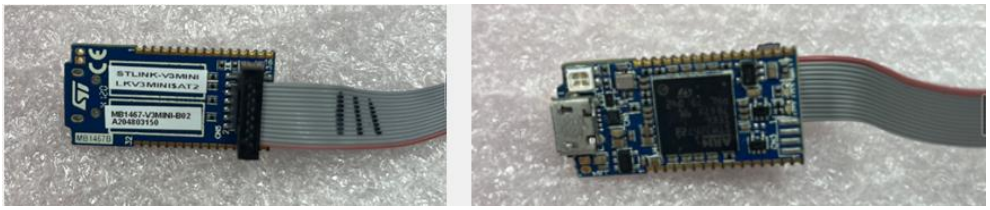

Figure 2-7

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

3. Board Description

3-1 AX58400-TSB

AX58400-TSB is an evaluation board for AX58400 IC and has the components list below:

- AX58400 – An ASIX IC that integrated an ESC (EtherCAT Slave Controller) with a dual-core ARM cortex-M7/M4
- ESC Port0 – Port0 of the ESC
- ESC Port1 – Port1 of the ESC
- USB2.0 OTG – USB2.0 OTG port of AX58400
- Power – Power input port of AX58400-TSB
- Reset – Reset button of AX58400
- Boot Mode Enable – Force AX58400 entering Boot Mode
- 3 Port Mode – Enable ESC port 2, default is Disable
- User Button – Button can be used for user
- ICE – ICE connector for STLink-V3Mini

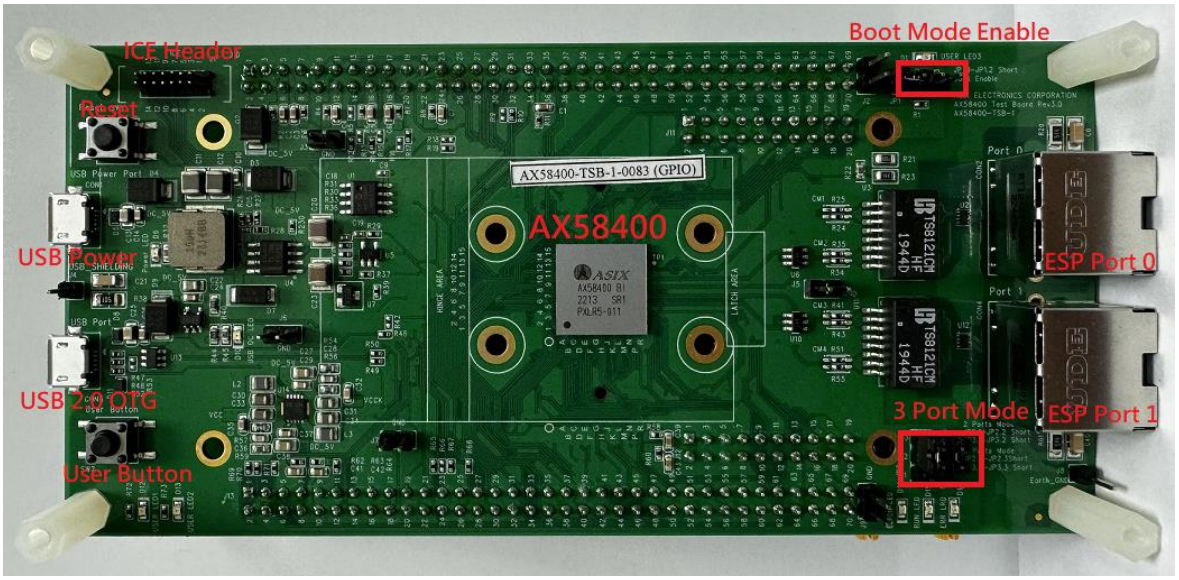

Figure 3-1

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

3-2 AX58400-EXB-ADIO

AX58400-EXB-ADIO is an expansion board for AX58400 GPIO / AIO application and has the components list below:

- LEDs for Digital Output – Digital output (8-bit) to the LEDs
- DIP Switch for Digital Input – Dip switch to the digital input (8-bit)
- Analog Input Pin – You can measure the input voltage of ADC by this PIN
- Analog Output Pin – You can measure the output voltage of DAC by this PIN
- Variable Resistor – You can adjust input voltage of ADC through this component
- ESC Port2 – Port2 of the ESC
- Rotary Dip Switch – A rotary dip switch that can assign a 16-bit digits
- RS-232 Port – The I/O of this RS232 port comes from UART of AX58400-TSB

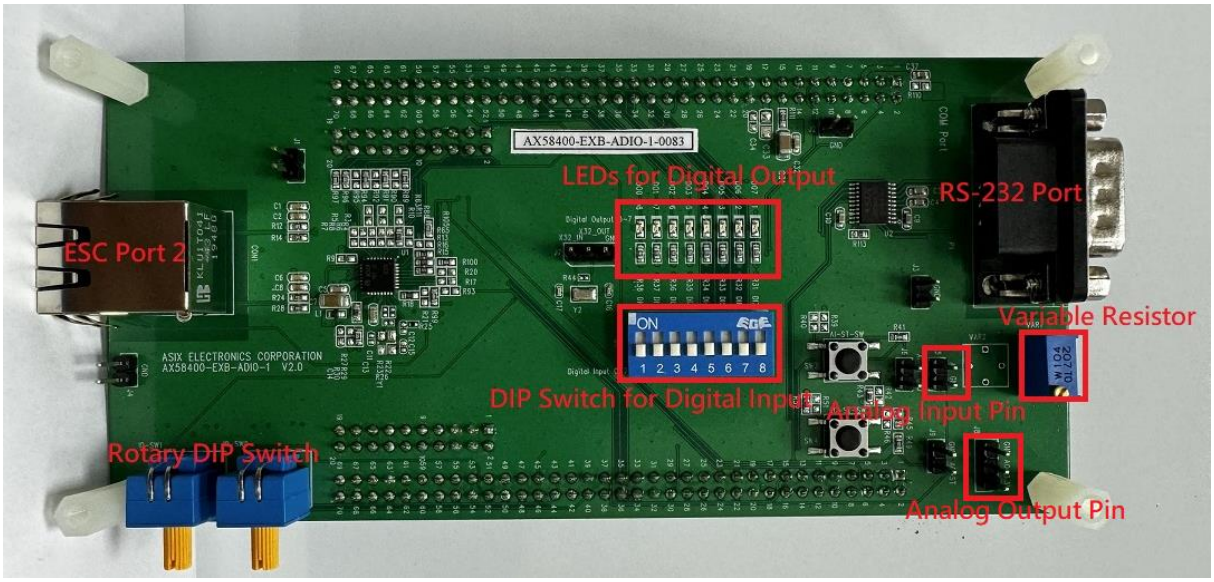

Figure 3-2

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

4. Environment Setup

4-1 Hardware Connectivity

Please based on below table to connect all necessary equipment.

AX58400 GpioAio Reference Design Connectivity					
Connectivity Item	PC/IPC	STLINK-V3MINI	AX58400-TSB-1	AX58400-EXB-ADIO	Remark
EtherCAT network	Ethernet port		CON2 (port0)		
			CON4 (port1)		
Extended Connector			J10	J10	
			J13	J11	
Power USB	USB Port 1		CON1		
ICE USB (for development stage only)	USB Port 2	CN1			
		CN5	J1		Please note the red wire of STLink-V3Mini cable MUST be aligned to J1.1 pin on AX58400-TSB board.

Table 4-1

If your environment is set up ready, it should look like this.

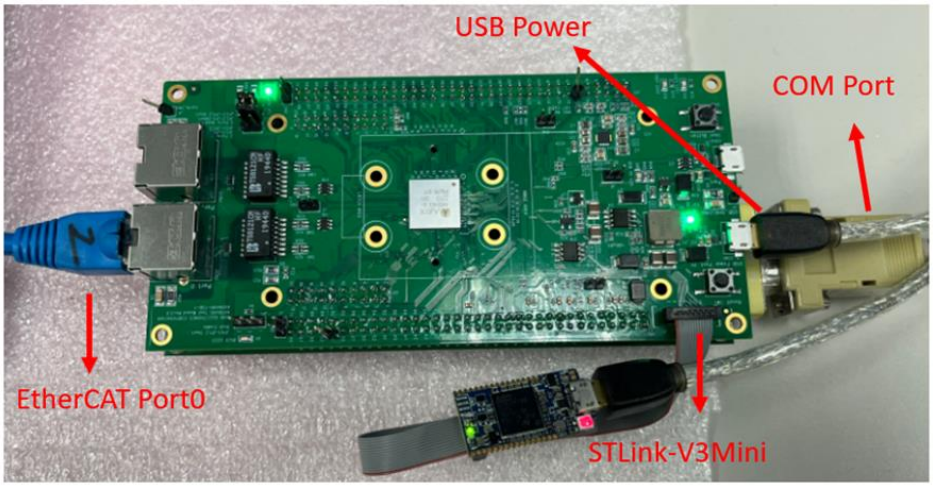

Figure 4-1

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

4-2 AX58400 Pinout Patch for STM32CubeMX

Assume the STM32CubeMX has been installed successfully, and if the developer would like to get a more suitable pinout view of AX58400 on STM32CubeMX, please follow the steps below to achieve this.

Step 1: Open the below files with text editor and modify the "**STMMX_ROOT**" as your **STM32CubeMX** installation path.
AX58400_Pinout_Patch_vXXX\AX58400_Pinout_Patch.cmd
AX58400_Pinout_Patch_vXXX\AX58400_Pinout_UnPatch.cmd

```
rem - STM32CubeMX installation root path configuration
set "STMMX_ROOT=%programfiles%\STMicroelectronics\STM32Cube\STM32CubeMX\"
set "PATCH_SRC_PATH=%~dp0STMMX\"
```

Step 2: Performs the file as below with the administrator permission to apply the patch.
AX58400_Pinout_Patch_vXXX\AX58400_Pinout_Patch.cmd

Step 3: Open the STM32CubeMX and check result. If the patch is successful, you will see the pinout view is as below figure. Depending on the STM32CubeMX version, the MCU name differs:
- Before STM32CubeMX v6.4.0: MCU name is "**STM32_AX58400BIx**".
- STM32CubeMX v6.4.0 and later: MCU name is "**STM32H755ZITx**".

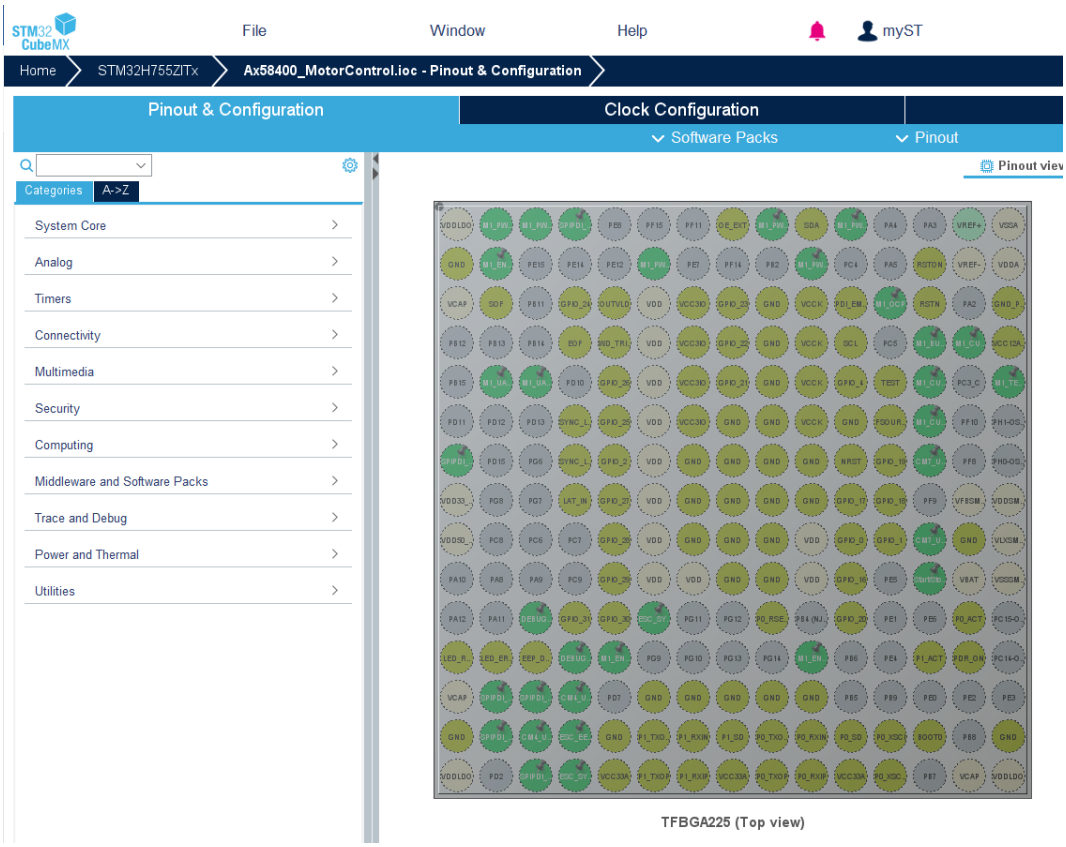

Figure 4-2

Also, if you want go back to original state, please performing the un-patch file below.
AX58400_Pinout_Patch_vXXX\AX58400_Pinout_UnPatch.cmd

Notice: This step requires the additional installation of the "**AX58400 Pinout Patch**" package. Please contact ASIX's Technical Support (support@asix.com.tw) for further support.

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

5. Building Reference Design Begin with STM32CubeMX

This chapter is specifically for designers who need to generate the source code package on their own. Generally, designers can refer to Sections 5-5 to generate the SSC source code, and then Section 6 to build the firmware code directly.

Here will introduce the procedure to build up the complete source code package, please follow the steps below to do this.
Notice: If you just need to generate the SSC code only, please refer to section 5-5 directly.

5-1 Create New Project in STM32CubeMX

Step 1: Execute the STM32CubeMX, then click "**ACCESS TO MCU SELECTOR**" on main page.

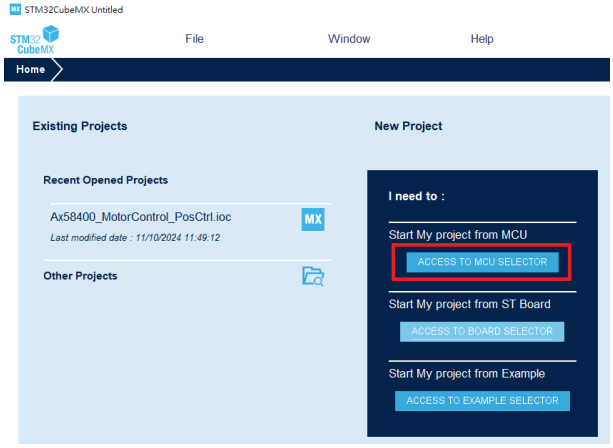

Figure 5-1

Step 2: Searching part number "**STM32H755ZIT6**", then select it on "**MCUs/MPU List**" and press "**Start Project**" to create project.

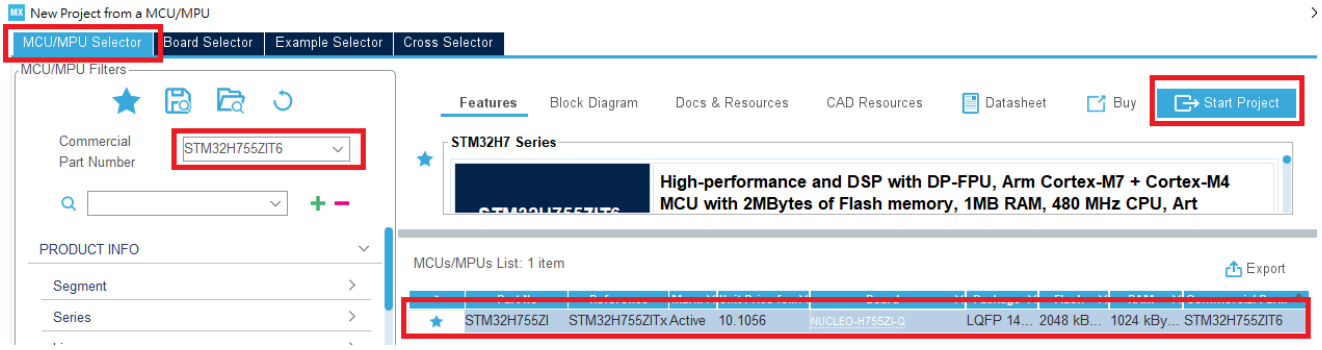

Figure 5-2

Step 3: Press "**Yes**" to enable MPU function.

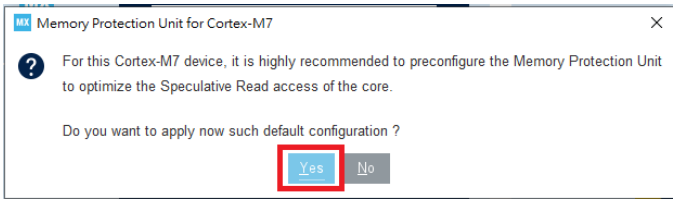

Figure 5-3

Notice: In order to keep the corrected reference path in BSP, please save the project folder in the path below.
BSP_ROOT\SampleCode

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

5-2 Modify Peripheral Settings in STM32CubeMX

Step 1: Select "**Pinout & Configuration > System Core > CORTEX_M7**" and modify the settings as highlighted in red.

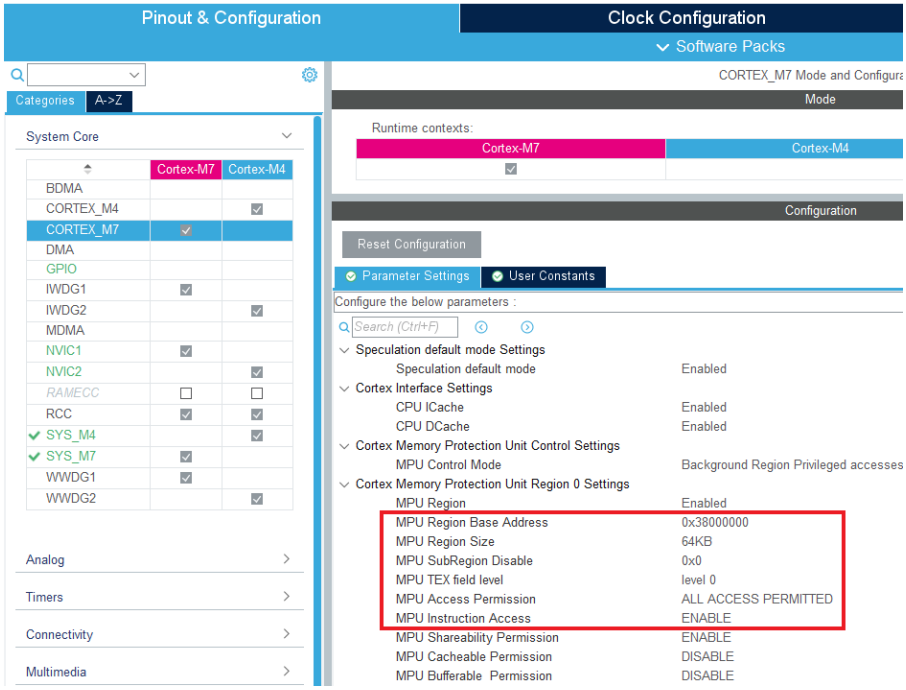

Figure 5-4

Step 2: Add pin assignment as the below in "**Pinout & Configuration > System Core > GPIO > GPIO**".

GPIO | ADC | DAC | SPI | UART | NVIC

Search Signals

Search (Ctrl+F)

Pin Name	Signal on Pin	Pin Context Assignement	GPIO output level	GPIO mode	GPIO Pull-up/Pull-down	Maximum output speed	Fast Mode	User Label
PA0	n/a	ARM Cortex-M7	n/a	Input mode	No pull-up and no pull-down	n/a	n/a	DI0
PB0	n/a	ARM Cortex-M7	n/a	Input mode	No pull-up and no pull-down	n/a	n/a	DI5
PB1	n/a	ARM Cortex-M7	n/a	Input mode	No pull-up and no pull-down	n/a	n/a	SW3
PB7	n/a	ARM Cortex-M7	n/a	Input mode	No pull-up and no pull-down	n/a	n/a	DI4
PB8	n/a	ARM Cortex-M7	n/a	Input mode	No pull-up and no pull-down	n/a	n/a	SW2
PB9	n/a	ARM Cortex-M7	n/a	Input mode	No pull-up and no pull-down	n/a	n/a	SW1
PC0	n/a	ARM Cortex-M7	Low	Output Push Pull	Pull-up	Medium	n/a	DO4
PC2_C	n/a	ARM Cortex-M7	Low	Output Push Pull	Pull-up	Medium	n/a	DO6
PC3_C	n/a	ARM Cortex-M7	Low	Output Push Pull	Pull-up	Medium	n/a	DO7
PC6	n/a	ARM Cortex-M7	n/a	Input mode	No pull-up and no pull-down	n/a	n/a	SW0
PD2	n/a	ARM Cortex-M7	n/a	Input mode	No pull-up and no pull-down	n/a	n/a	DI1
PD3	n/a	ARM Cortex-M4	n/a	Input mode	Pull-down	n/a	n/a	ESC_EEPDONE
PD4	n/a	ARM Cortex-M4	n/a	Input mode	Pull-up	n/a	n/a	SPIPDI_INT
PD5	n/a	ARM Cortex-M4	n/a	Input mode	Pull-up	n/a	n/a	ESC_SYNC0
PD6	n/a	ARM Cortex-M4	n/a	Input mode	Pull-up	n/a	n/a	ESC_SYNC1
PD7	n/a	ARM Cortex-M7	n/a	Input mode	No pull-up and no pull-down	n/a	n/a	DI6
PD8	n/a	ARM Cortex-M7	n/a	Input mode	No pull-up and no pull-down	n/a	n/a	DI2
PD11	n/a	ARM Cortex-M7	n/a	External Interrupt Mode with Rising edge trigger detection	Pull-down	n/a	n/a	ADC TRIG
PD14	n/a	ARM Cortex-M4	High	Output Push Pull	No pull-up and no pull-down	High	n/a	SPIPDI_FUN_CS
PE2	n/a	ARM Cortex-M7	n/a	Input mode	No pull-up and no pull-down	n/a	n/a	SW7
PE3	n/a	ARM Cortex-M7	Low	Output Push Pull	Pull-up	Medium	n/a	DO0
PE4	n/a	ARM Cortex-M7	Low	Output Push Pull	Pull-up	Medium	n/a	DO1
PE5	n/a	ARM Cortex-M7	Low	Output Push Pull	Pull-up	Medium	n/a	DO2
PE6	n/a	ARM Cortex-M7	Low	Output Push Pull	Pull-up	Medium	n/a	DO3
PE7	n/a	ARM Cortex-M7	n/a	Input mode	No pull-up and no pull-down	n/a	n/a	SW5
PE8	n/a	ARM Cortex-M7	n/a	Input mode	No pull-up and no pull-down	n/a	n/a	SW4
PE9	n/a	ARM Cortex-M7	n/a	Input mode	No pull-up and no pull-down	n/a	n/a	SW6
PE10	n/a	ARM Cortex-M4	High	Output Push Pull	Pull-up	High	n/a	SPIPDI_ESC_CS
PF6	n/a	ARM Cortex-M7	Low	Output Push Pull	Pull-up	Medium	n/a	DO5
PF7	n/a	ARM Cortex-M7	n/a	Input mode	No pull-up and no pull-down	n/a	n/a	DI3
PF8	n/a	ARM Cortex-M7	n/a	Input mode	No pull-up and no pull-down	n/a	n/a	DI7

Figure 5-5

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

Step 3: Add pin assignment for ADC, DAC, SPI3, UART4 and UART5 as below.

GPIO | Single Mapped Signals

Search Signals

Search (Ctrl+F) | Show only M

Pin Name	Signal on Pin	Pin Context Assignement	GPIO output level	GPIO mode	GPIO Pull-up/Pull-down	Maximum output speed	Fast Mode	User Label
PA3	ADC1_INP15	Free	n/a	Analog mode	No pull-up and no pull-down	n/a	n/a	
PA4	DAC1_OUT1	Free	n/a	Analog mode	No pull-up and no pull-down	n/a	n/a	
PB12	UART5_RX	Free	n/a	Alternate Function Push Pull	No pull-up and no pull-down	Medium	n/a	
PB13	UART5_TX	Free	n/a	Alternate Function Push Pull	No pull-up and no pull-down	Medium	n/a	
PC10	SPI3_SCK	n/a	n/a	Alternate Function Push Pull	Pull-up	High	n/a	SPIPDI_CLK
PC11	SPI3_MISO	n/a	n/a	Alternate Function Push Pull	No pull-up and no pull-down	High	n/a	SPIPDI_MISO
PC12	SPI3_MOSI	n/a	n/a	Alternate Function Push Pull	Pull-up	High	n/a	SPIPDI_MOSI
PD0	UART4_RX	Free	n/a	Alternate Function Push Pull	No pull-up and no pull-down	Medium	n/a	
PD1	UART4_TX	Free	n/a	Alternate Function Push Pull	No pull-up and no pull-down	Medium	n/a	
PF9	DAC1_EXTI9	n/a	n/a	External Interrupt Mode with ...	Pull-down	n/a	n/a	DAC_TRIG

Figure 5-6

Step 4: Enable ADC1 as the settings highlighted in red.

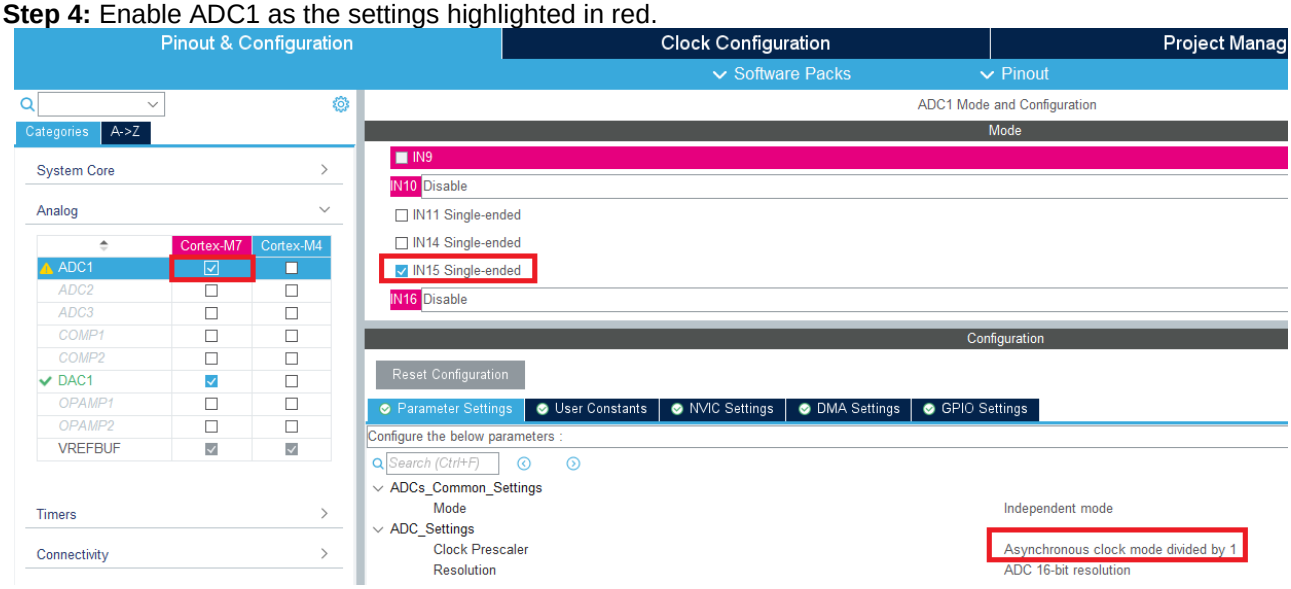

Figure 5-7

Step 5: Enable DAC with the setting highlighted in red.

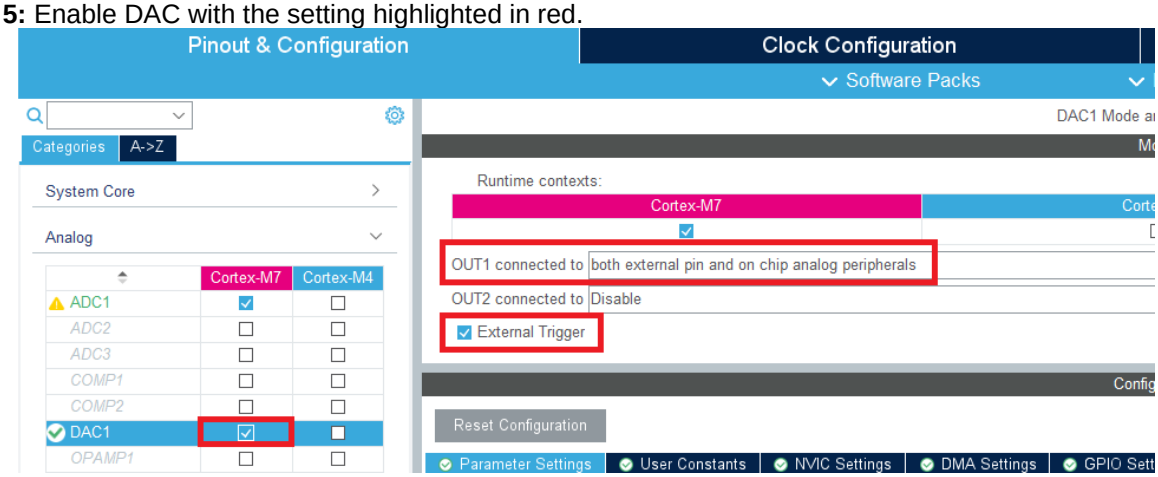

Figure 5-8

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

Step 6: Activated TIM6 & TIM7 as the setting below.

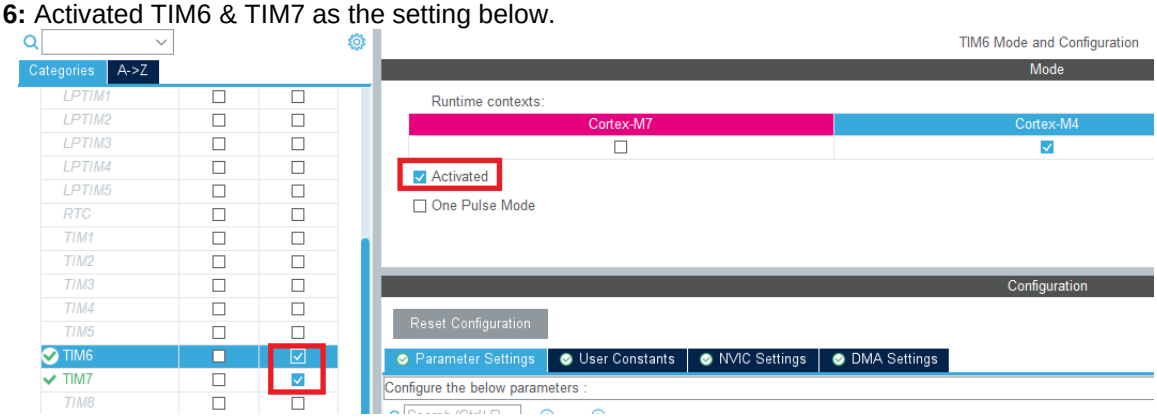

Figure 5-9

Step 7: Enable SPI3 with the setting below.

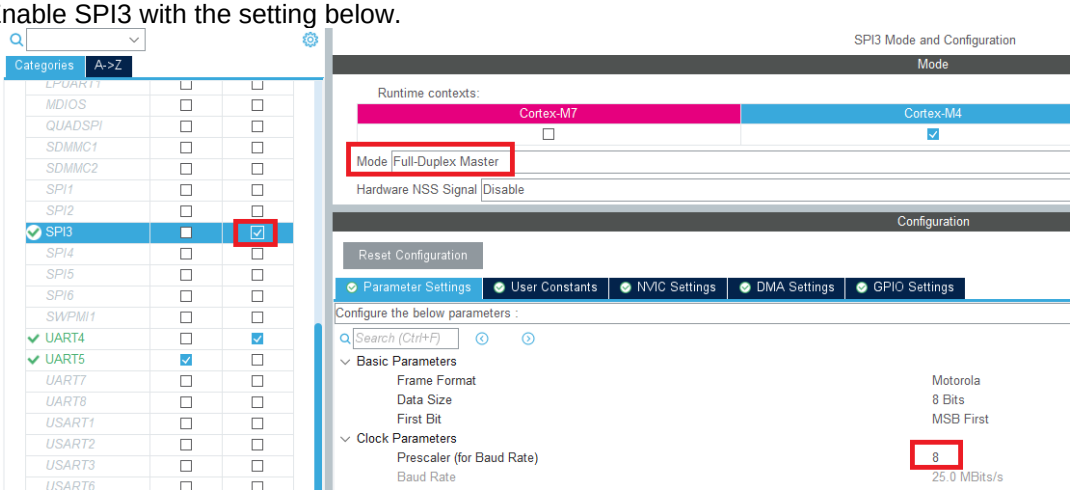

Figure 5-10

Step 8: Enable UART4 as the setting below.

Figure 5-11

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

Step 9: Enable UART5 as the setting below.

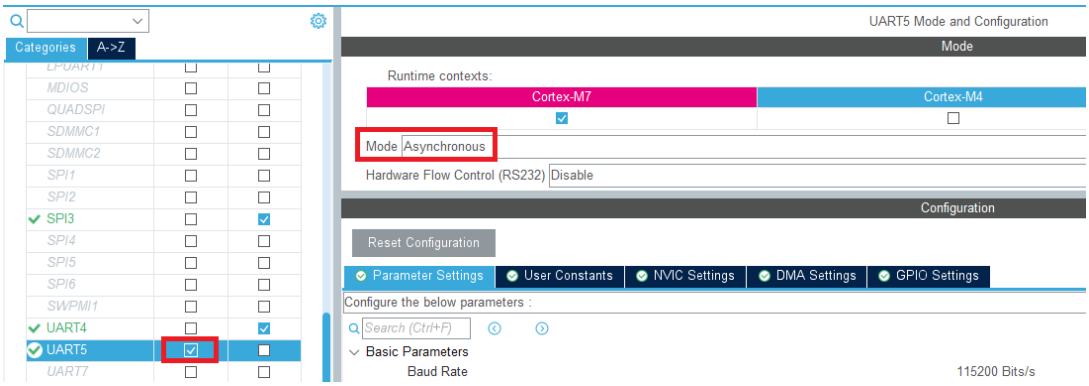

Figure 5-12

Step 10: Modify the NVIC1 setting as below.

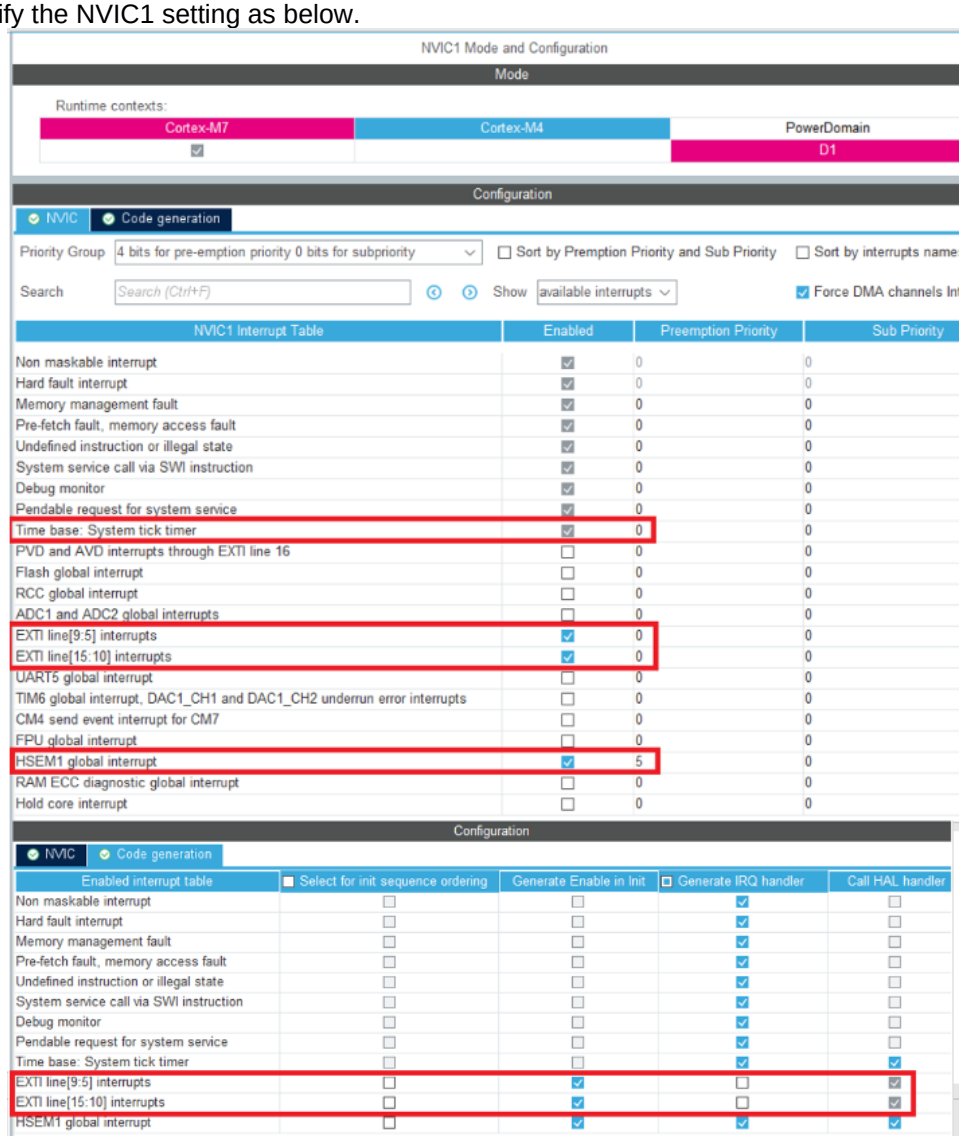

Figure 5-13

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

Step 11: Modify the NVIC2 setting as below.

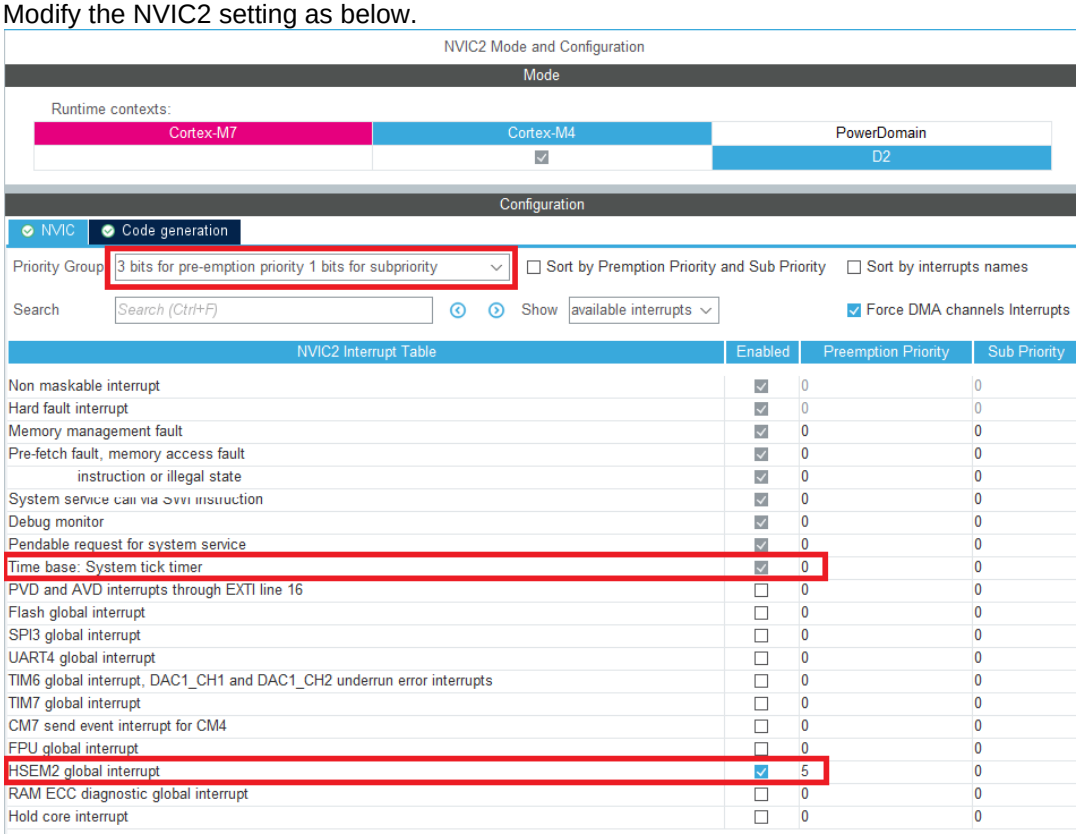

Figure 5-14

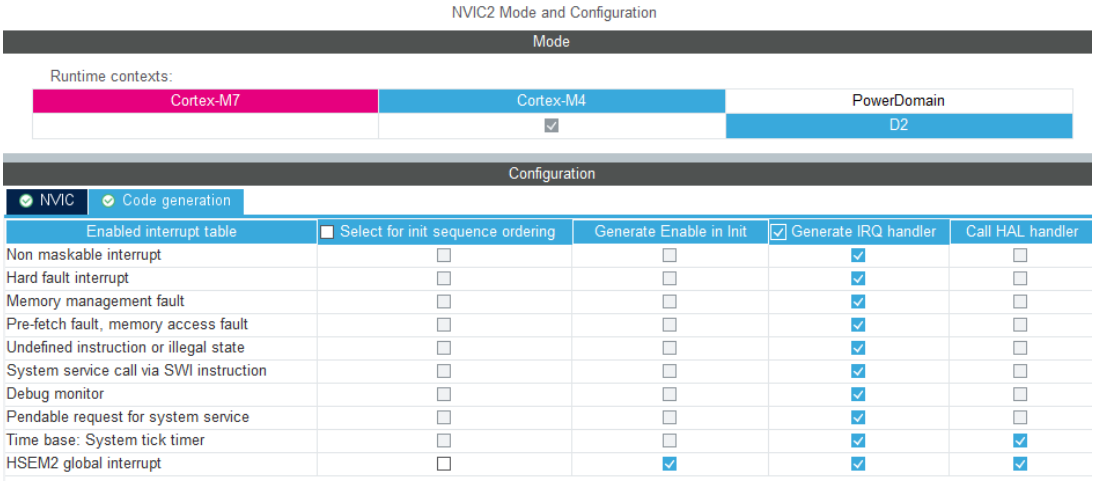

Figure 5-15

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

Step 12: Adjusting the clock tree in "**Clock Configuration**" page.
Press "**NO**" to ignore auto solver process, then modify the settings as the red highlighted below.

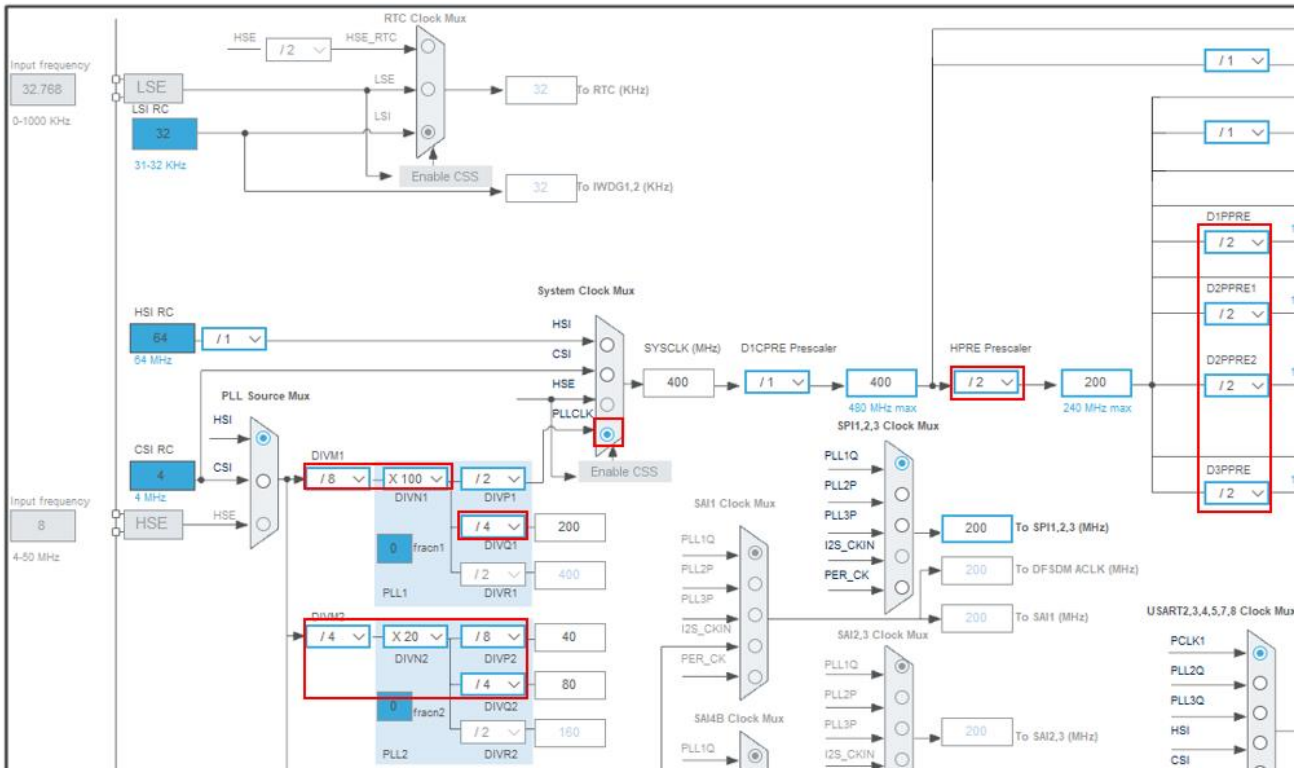

Figure 5-16

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

5-3 Generate Source Code in STM32CubeMX

Step 1: In "**Project Manager > Project**" page, extended the minimum heap size to **0x1000** on CM4, and select Firmware Package as "**STM32Cube FW_H7 V1.11.2**".

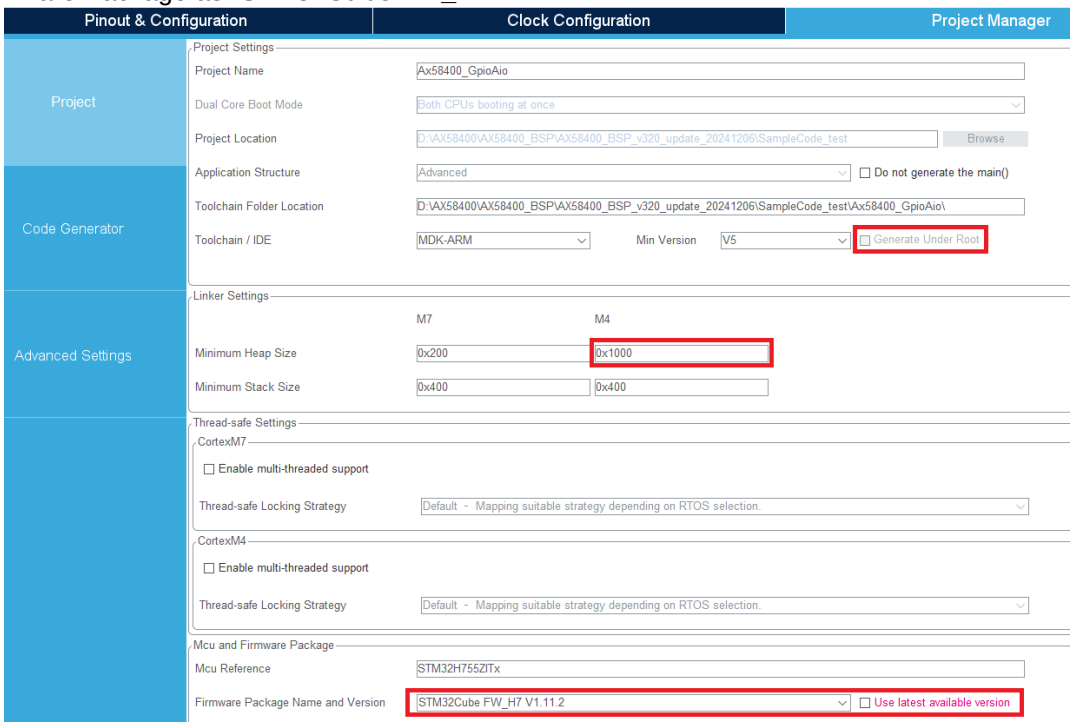

Figure 5-17

Step 2: In "**Project Manager > Code Generator**" page, select "**Copy all used libraries into the project folder**".

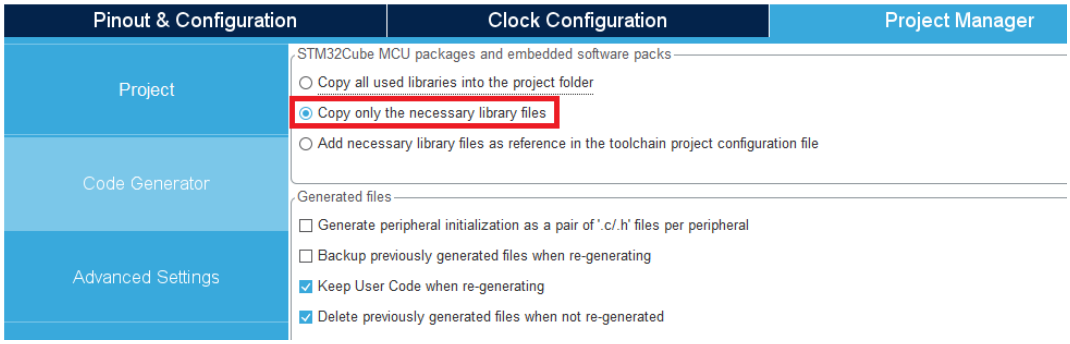

Figure 5-18

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

Step 3: In "**Project Manager > Advanced Settings**" page, disable initial code generation as below.

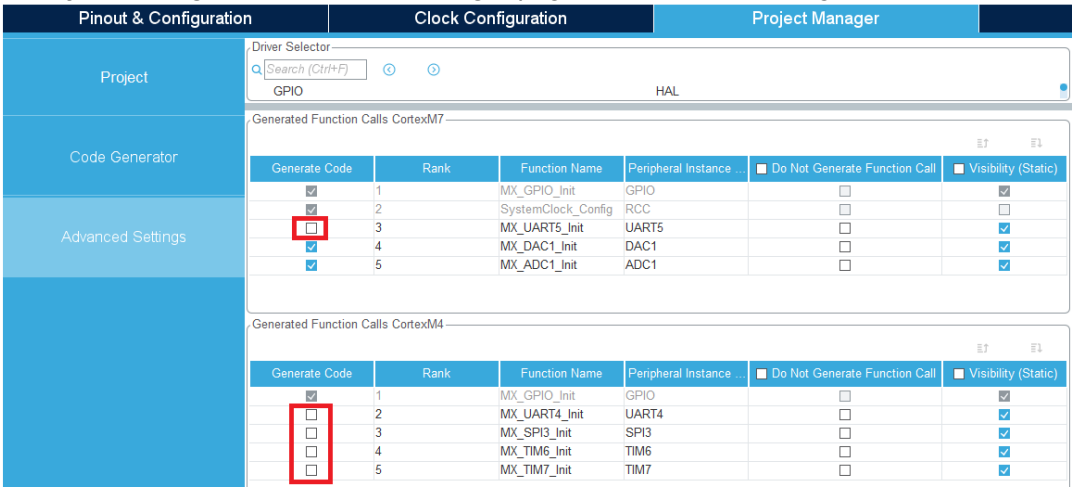

Figure 5-19

Step 4: Generates the source code.

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

5-4 Source Code Adjustment

Step 1: Add the AX58400 source package.
Copy the folders as below to your STM32CubeMX project folder.
BSP_ROOT\SampleCode\Ax58400_GpioAio\AX58400
BSP_ROOT\SampleCode\Ax58400_GpioAio\Binary
BSP_ROOT\SampleCode\Ax58400_GpioAio\For_SSC_Tool
BSP_ROOT\SampleCode\Ax58400_GpioAio\For_TwinCAT

Step 2: Adjust the Vector Table Address.
As highlighted in the figure below, modify file content in the path:
BSP_ROOT\SampleCode\Ax58400_GpioAio\Common\Src\system_stm32h7xx_dualcore_boot_cm4_cm7.c

```
/*!< Uncomment the following line if you need to relocate the vector table
     anywhere in FLASH BANK1 or AXI SRAM, else the vector table is kept at the automatic
     remap of boot address selected */
#define USER_VECT_TAB_ADDRESS

#if defined(USER_VECT_TAB_ADDRESS)
    • • •
#elif defined(CORE_CM7)
/*!< Uncomment the following line if you need to relocate your vector Table
     in D1 AXI SRAM else user remap will be done in FLASH BANK1. */
/* #define VECT_TAB_SRAM */
#if defined(VECT_TAB_SRAM)
#define VECT_TAB_BASE_ADDRESS   D1_AXISRAM_BASE   /*!< Vector Table base address field.
                                                       This value must be a multiple of 0x400. */
#define VECT_TAB_OFFSET         0x00020000U       /*!< Vector Table base offset field.
                                                       This value must be a multiple of 0x400. */
#else
#define VECT_TAB_BASE_ADDRESS   FLASH_BANK1_BASE  /*!< Vector Table base address field.
                                                       This value must be a multiple of 0x400. */
#define VECT_TAB_OFFSET         0x00020000U       /*!< Vector Table base offset field.
                                                       This value must be a multiple of 0x400. */
```

Figure 5-20

Step 3: Add the Share Memory declaration for CM4.

- **For Keil MDK IDE**

As highlighted in the figure below, modify file content in the path:
BSP_ROOT\SampleCode\Ax58400_GpioAio\MDK-ARM\stm32h755xx_flash_CM4.sct

```
; *************************************************************
; *** Scatter-Loading Description File generated by uVision ***
; *************************************************************

LR_IROM1 0x08100000 0x00100000  {    ; load region size_region
  ER_IROM1 0x08100000 0x00100000  {  ; load address = execution address
   *.o (RESET, +First)
   *(InRoot$$Sections)
   .ANY (+RO)
  }
  RW_IRAM1 0x10000000 0x00048000{  ; RW data
   .ANY (+RW +ZI)
  }
  RW_IRAM3 0x38000000 0x00010000{
   *(.RAM_D3)
  }
}
```

Figure 5-21

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

- **For STM32CubeIDE**

As highlighted in the figure below, modify file content in the path:
BSP_ROOT\SampleCode\Ax58400_GpioAio\STM32CubeIDE\CM4\STM32H755ZITX_FLASH.ld

```
43 /* Specify the memory areas */
44 MEMORY
45 {
46 FLASH (rx)      : ORIGIN = 0x08100000, LENGTH = 1024K
47 RAM (xrw)       : ORIGIN = 0x10000000, LENGTH = 288K
48 RAM_D3 (xrw)    : ORIGIN = 0x38000000, LENGTH = 64K
49 }
     ● ● ●
88   .ARM.extab : /* The READONLY keyword is only supported in GCC
89   {
     ● ● ●
91   } >FLASH
92   .ARM : /* The READONLY keyword is only supported in GCC11 and
93   {
     ● ● ●
97   } >FLASH
98
99   .preinit_array : /* The READONLY keyword is only supported in
100  {
     ● ● ●
104  } >FLASH
105  .init_array : /* The READONLY keyword is only supported in GC
106  {
     ● ● ●
111  } >FLASH
112  .fini_array : /* The READONLY keyword is only supported in GC
113  {
     ● ● ●
163  } >RAM
164
165  .RAM_D3 :
166  {
167  } >RAM_D3 AT>FLASH
168
169  /* Remove information from the standard libraries */
```

Figure 5-22

Step 4: Add the Share Memory declaration for CM7.

- **For Keil MDK IDE**

As highlighted in the figure below, modify file content in the path:
BSP_ROOT\SampleCode\Ax58400_GpioAio\MDK-ARM\stm32h755xx_flash_CM7.sct

```
1 ; *************************************************************
2 ; *** Scatter-Loading Description File generated by uVision ***
3 ; *************************************************************
4
5 LR_IROM1 0x08020000 0x000E0000  {    ; load region size_region
6   ER_IROM1 0x08020000 0x000E0000  {  ; load address = execution address
7    *.o (RESET, +First)
8    *(InRoot$$Sections)
9    .ANY (+RO)
10  }
11  RW_IRAM1 0x20000000 0x00020000{  ; RW data
12   .ANY (+RW +ZI)
13  }
14  RW_IRAM3 0x38000000 0x00010000{
15   *(.RAM_D3)
16  }
17 }
```

Figure 5-23

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

- **For STM32CubeIDE**

As highlighted in the figure below, modify file content in the path:
BSP_ROOT\SampleCode\Ax58400_GpioAio\STM32CubeIDE\CM7\STM32H755ZITX_FLASH.ld

```
/* Memories definition */
MEMORY
{
  RAM_D1 (xrw)    : ORIGIN = 0x20000000, LENGTH = 128K
  FLASH   (rx)    : ORIGIN = 0x08020000, LENGTH = 896K     /* Memo

  RAM_D2 (xrw)    : ORIGIN = 0x30000000, LENGTH = 288K
  RAM_D3 (xrw)    : ORIGIN = 0x38000000, LENGTH = 64K
  ITCMRAM (xrw)   : ORIGIN = 0x00000000, LENGTH = 64K
}
    ● ● ●
  .ARM.extab : /* The READONLY keyword is only supported in GCC1
  {
    ● ● ●
  } >FLASH
  .ARM : /* The READONLY keyword is only supported in GCC11 and
  {
    ● ● ●
  } >FLASH

  .preinit_array : /* The READONLY keyword is only supported in
  {
    ● ● ●
  } >FLASH

  .init_array : /* The READONLY keyword is only supported in GCC
  {
    ● ● ●
  } >FLASH

  .fini_array : /* The READONLY keyword is only supported in GCC
  {
    ● ● ●
  } >RAM_D1

  .RAM_D3 :
  {
  } >RAM_D3 AT>FLASH

  /* Remove information from the compiler libraries */
```

Figure 5-24

Step 5: Modify source code on CM4 core.
As highlighted in the figure below, modify file content in the path:
BSP_ROOT\SampleCode\Ax58400_GpioAio\CM4\Src\main.c

```
/* Private includes ----------------------------------------------------------
/* USER CODE BEGIN Includes */
#include "AX58400_Hw.h"
#include "applInterface.h"
#include "ax58400.h"
#include "AX58400_GPIO_8Bit_AIO_16Bit.h"
#include "bootmode.h"
#include "ax58400_FoeAppl.h"
/* USER CODE END Includes */
    ● ● ●
  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  /* USER CODE BEGIN 2 */
#if (AX58400_DEBUG_ENABLE)
  AX_UART_Init();
  CONSOLE_Init();
  DBG_PRINT("%s M4 Firmware v%s\r\n", DEVICE_NAME, DEVICE_SW_VERSION);
#endif //#if (AX58400_DEBUG_ENABLE)
  CC_Init();
#if (SSC_STACK_ENABLE)
  if (HW_Init())
  {
    HW_Release();
  }
  else
  {
    MainInit();
    GpioAio_Init();
    AX58400_FoeInit();
  }
#endif //#if (SSC_STACK_ENABLE)
  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
#if (AX58400_DEBUG_ENABLE)
        CONSOLE_Task();
#endif //#if (AX58400_DEBUG_ENABLE)

#if (SSC_STACK_ENABLE)
        MainLoop();
        BL_Reboot();

#if (ESC_PHY_ENHANCEMENT)
        HW_PhyEnhancement();
#endif //#if (ESC_PHY_ENHANCEMENT)

#endif //#if (SSC_STACK_ENABLE)
```

Figure 5-25

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

As highlighted in the figure below, modify file content in the path:
BSP_ROOT\SampleCode\Ax58400_GpioAio\CM4\Src\stm32h7xx_it.c

```
/* Includes ------------------------------------------------------------------*/
#include "main.h"
#include "stm32h7xx_it.h"
/* Private includes ----------------------------------------------------------*/
/* USER CODE BEGIN Includes */
#include "ax58400.h"
/* USER CODE END Includes */
    ● ● ●
void SysTick_Handler(void)
{
  /* USER CODE BEGIN SysTick_IRQn 0 */

  /* USER CODE END SysTick_IRQn 0 */
  HAL_IncTick();
  /* USER CODE BEGIN SysTick_IRQn 1 */
#if (AX58400_DEBUG_ENABLE)
  CONSOLE_TimeTick();
#endif //#if (AX58400_DEBUG_ENABLE)
  /* USER CODE END SysTick_IRQn 1 */
}
```

Figure 5-26

Step 6: Modify source code on CM7 core.
As highlighted in the figure below, modify file content in the path:
BSP_ROOT\SampleCode\Ax58400_GpioAio\CM7\Src\main.c

```
/* Private includes ----------------------------------------------------------*/
/* USER CODE BEGIN Includes */
#include "ecat_def.h"
#include "ax58400.h"
#include "AX58400_Hw.h"
/* USER CODE END Includes */
    ● ● ●
/* Private macro -------------------------------------------------------------*/
/* USER CODE BEGIN PM */
#define DO_LEVEL_REVERSE            0

#define ADC_STATE_INIT              0
#define ADC_STATE_WAIT_CONVERSION   1
#define ADC_STATE_CONVERSION_FINISH 2
#define ADC_STATE_WAIT_NEXT         3

#define ADC_NEXT_TIMEOUT            1//ms
/* USER CODE END PM */
    ● ● ●
/* USER CODE BEGIN PV */
u8 mainIgnoreEsc2McuInt=1;
/* USER CODE END PV */
    ● ● ●
/* USER CODE BEGIN PFP */
u8 GetDigitalInput(void);
void SetDigitalOutput(u8 val);
u16 GetAnalogInput(void);
void SetAnalogOutput(u16 val);
u8 GetSwitchID(void);

/* USER CODE END PFP */
    ● ● ●
  /* USER CODE BEGIN 2 */
#if (AX58400_DEBUG_ENABLE)
  AX_UART_Init();
  CONSOLE_Init();
  DBG_PRINT("%s M7 Firmware v%s\r\n", DEVICE_NAME, DEVICE_SW_VERSION);
#endif //#if (AX58400_DEBUG_ENABLE)
  /* Configure PC2 and PC2_C : Analog switch close (pads connected)     *
   *           PC3 and PC3_3 : Analog switch close (pads connected)     */
  HAL_SYSCFG_AnalogSwitchConfig(SYSCFG_SWITCH_PC2, SYSCFG_SWITCH_PC2_CLOSE);
  HAL_SYSCFG_AnalogSwitchConfig(SYSCFG_SWITCH_PC3, SYSCFG_SWITCH_PC3_CLOSE);

  CC_Init();
  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
#if (AX58400_DEBUG_ENABLE)
    CONSOLE_Task();
#endif //#if (AX58400_DEBUG_ENABLE)

    CC_AdioStatusChangeReq();
    /* USER CODE END WHILE */
```

Figure 5-27

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

```
508 u8 GetDigitalInput(void)
509 {
510     u8  val = 0;
511 
512     /* Handle input */
513     val |= HAL_GPIO_ReadPin(GPIOA, GPIO_PIN_0) ? 0x01 : 0x00;
514     val |= HAL_GPIO_ReadPin(GPIOD, GPIO_PIN_2) ? 0x02 : 0x00;
515     val |= HAL_GPIO_ReadPin(GPIOD, GPIO_PIN_8) ? 0x04 : 0x00;
516     val |= HAL_GPIO_ReadPin(GPIOF, GPIO_PIN_7) ? 0x08 : 0x00;
517     val |= HAL_GPIO_ReadPin(GPIOB, GPIO_PIN_7) ? 0x10 : 0x00;
518     val |= HAL_GPIO_ReadPin(GPIOB, GPIO_PIN_0) ? 0x20 : 0x00;
519     val |= HAL_GPIO_ReadPin(GPIOD, GPIO_PIN_7) ? 0x40 : 0x00;
520     val |= HAL_GPIO_ReadPin(GPIOF, GPIO_PIN_8) ? 0x80 : 0x00;
521     return val;
522 
523 } /* End of GetDigitalInput() */
```

```
534 void SetDigitalOutput(u8 val)
535 {
536 
537 #if (DO_LEVEL_REVERSE)
538     /* Reverse the output level */
539     val = ~val;
540 #endif //#if (DO_LEVEL_REVERSE)
541 
542     /* Handle output */
543     HAL_GPIO_WritePin(GPIOE, GPIO_PIN_3, ((val & BIT_0) ? GPIO_PIN_SET : GPIO_PIN_RESET));
544     HAL_GPIO_WritePin(GPIOE, GPIO_PIN_4, ((val & BIT_1) ? GPIO_PIN_SET : GPIO_PIN_RESET));
545     HAL_GPIO_WritePin(GPIOE, GPIO_PIN_5, ((val & BIT_2) ? GPIO_PIN_SET : GPIO_PIN_RESET));
546     HAL_GPIO_WritePin(GPIOE, GPIO_PIN_6, ((val & BIT_3) ? GPIO_PIN_SET : GPIO_PIN_RESET));
547     HAL_GPIO_WritePin(GPIOC, GPIO_PIN_0, ((val & BIT_4) ? GPIO_PIN_SET : GPIO_PIN_RESET));
548     HAL_GPIO_WritePin(GPIOF, GPIO_PIN_6, ((val & BIT_5) ? GPIO_PIN_SET : GPIO_PIN_RESET));
549     HAL_GPIO_WritePin(GPIOC, GPIO_PIN_2, ((val & BIT_6) ? GPIO_PIN_SET : GPIO_PIN_RESET));
550     HAL_GPIO_WritePin(GPIOC, GPIO_PIN_3, ((val & BIT_7) ? GPIO_PIN_SET : GPIO_PIN_RESET));
551 
552 } /* End of SetDigitalOutput() */
```

Figure 5-28

```
563 u16 GetAnalogInput(void)
564 {
565     static u32 adcState = ADC_STATE_INIT, tickstart=0;
566     u32 tmp16;
567     static u16 val=0;
568 
569     if (adcState == ADC_STATE_INIT)
570     {
571         /* Start conversion */
572         if (HAL_ADC_Start(&hadc1) != HAL_OK)
573         {
574             HAL_ADC_Stop(&hadc1);
575             return val;
576         }
577         adcState = ADC_STATE_WAIT_CONVERSION;
578     }
579     else if (adcState == ADC_STATE_WAIT_CONVERSION)
580     {
581         /* Ignore handle, just wait conversion complete */
582         if (hadc1.Instance->ISR & ADC_FLAG_EOC)
583         {
584             adcState = ADC_STATE_CONVERSION_FINISH;
585         }
586     }
587     else if (adcState == ADC_STATE_CONVERSION_FINISH)
588     {
589         /* Get the conversion result of the sample module 0 */
590         val = (u16)HAL_ADC_GetValue(&hadc1);
591         tickstart = HAL_GetTick();
592         adcState = ADC_STATE_WAIT_NEXT;
593     }
594     else if (adcState == ADC_STATE_WAIT_NEXT)
595     {
596         /* Wait a while for next conversion */
597         tmp16 = HAL_GetTick();
598         if (tmp16 >= tickstart)
599         {
600             if ((tmp16 - tickstart) >= ADC_NEXT_TIMEOUT)
601             {
602                 adcState = ADC_STATE_INIT;
603             }
604         }
605         else
606         {
607             tickstart = tmp16;
608         }
609     }
610 
611     return val;
612 } /* End of GetAnalogInput() */
```

Figure 5-29

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

```
622 */
623 void SetAnalogOutput(u16 val)
624 {
625     u32 tmp16, tickstart=0;
626
627     tmp16 = val & 0x0FFF;
628
629     /* Start A/D conversion */
630     HAL_DAC_SetValue(&hdac1, DAC_CHANNEL_1, DAC_ALIGN_12B_R, tmp16);
631     HAL_DAC_Start(&hdac1, DAC_CHANNEL_1);
632
633     /* Wait convertion done */
634     tickstart = HAL_GetTick();
635     while ((HAL_GetTick() - tickstart) < 3)
636     {
637         if (0 == READ_BIT(hdac1.Instance->SWTRIGR, DAC_SWTRIGR_SWTRIG1))
638         {
639             break;
640         }
641     }
642 } /* End of SetAnalogOutput() */
```

```
653 void EXTI15_10_IRQHandler(void)
654 {
655     if (MCU_GPIO_EXTI_GET_IT(GPIO_PIN_11))
656     {
657         DBG_PRINT("EXTI15_10_IRQHandler: ADC trigger\r\n");
658         /* Detected ADC trigger signal */
659         MCU_GPIO_EXTI_CLEAR_FLAG(GPIO_PIN_11);
660     }
661
662     if (MCU_GPIO_EXTI_GET_IT(GPIO_PIN_10))
663     {
664         DBG_PRINT("EXTI15_10_IRQHandler: bit 10\r\n");
665         MCU_GPIO_EXTI_CLEAR_FLAG(GPIO_PIN_10);
666         if (mainIgnoreEsc2McuInt)
667         {
668             DBG_PRINT("EXTI15_10_IRQHandler: Ignore interrupt\r\n");
669             mainIgnoreEsc2McuInt = 0;
670         }
671         /* Detected the reset signal from ESC */
672 //        else
673 //        {
674 //            NVIC_SystemReset();
675 //        }
676     }
677
678     HAL_GPIO_EXTI_IRQHandler(GPIO_PIN_10);
679 } /* End of EXTI15_10_IRQHandler() */
```

```
690 u8 GetSwitchID(void)
691 {
692     u8  currentSwitchID = 0;
693
694     currentSwitchID |= HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_6) ? BIT_0 : 0;
695     currentSwitchID |= HAL_GPIO_ReadPin(GPIOB, GPIO_PIN_9) ? BIT_1 : 0;
696     currentSwitchID |= HAL_GPIO_ReadPin(GPIOB, GPIO_PIN_8) ? BIT_2 : 0;
697     currentSwitchID |= HAL_GPIO_ReadPin(GPIOB, GPIO_PIN_1) ? BIT_3 : 0;
698     currentSwitchID |= HAL_GPIO_ReadPin(GPIOE, GPIO_PIN_8) ? BIT_4 : 0;
699     currentSwitchID |= HAL_GPIO_ReadPin(GPIOE, GPIO_PIN_7) ? BIT_5 : 0;
700     currentSwitchID |= HAL_GPIO_ReadPin(GPIOE, GPIO_PIN_9) ? BIT_6 : 0;
701     currentSwitchID |= HAL_GPIO_ReadPin(GPIOE, GPIO_PIN_2) ? BIT_7 : 0;
702
703     return (~currentSwitchID);
704 } /* End of GetSwitchID() */
```

Figure 5-30

As highlighted in the figure below, modify file content in the path:
BSP_ROOT\SampleCode\Ax58400_GpioAio\CM7\Src\stm32h7xx_it.c

```
 20 /* Includes ------------------------------------------------------------------*/
 21 #include "main.h"
 22 #include "stm32h7xx_it.h"
 23 /* Private includes ----------------------------------------------------------*/
 24 /* USER CODE BEGIN Includes */
 25 #include "ax58400.h"
 26 /* USER CODE END Includes */
```

```
184 void SysTick_Handler(void)
185 {
186   /* USER CODE BEGIN SysTick_IRQn 0 */
187
188   /* USER CODE END SysTick_IRQn 0 */
189   HAL_IncTick();
190   /* USER CODE BEGIN SysTick_IRQn 1 */
191 #if (AX58400_DEBUG_ENABLE)
192   CONSOLE_TimeTick();
193 #endif //#if (AX58400_DEBUG_ENABLE)
194   /* USER CODE END SysTick_IRQn 1 */
195 }
```

Figure 5-31

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

5-5 Generate EtherCAT Slave Stack Code

Due to the ETG Slave Stack Code (SSC) license limitation, the user must re-generate the source code by the SSC tool locally, about how to get the SSC tool and how to generate targeted source code, please refer to the user guide as below.

BSP_ROOT\Document\AX58x00_SSC_Tool_Configuration_Import_UserGuide_vXXX.pdf

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

5-6 IDE Setting Adjustment

- **For Keil MDK IDE:**

Double clicks the file as below to open project.
BSP_ROOT\SampleCode\Ax58400_GpioAio\MDK-ARM\Ax58400_GpioAio.uvproj

Step 1: Add source code both on dual cores.
Clock on "**Project > Manage > Project Items…**" to open "**Manage Project Items**" dialog, then add groups and files as the figure below.

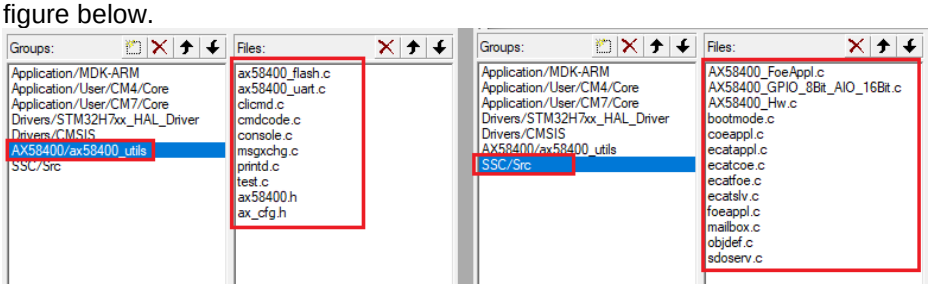

Figure 5-32

Step 2: Add include path both on dual cores.
Click on "**Project > Options for Target 'xxxx'… > C/C++ > Include Path**" to open "**Folder Setup**" dialog, then add paths as the figure below.

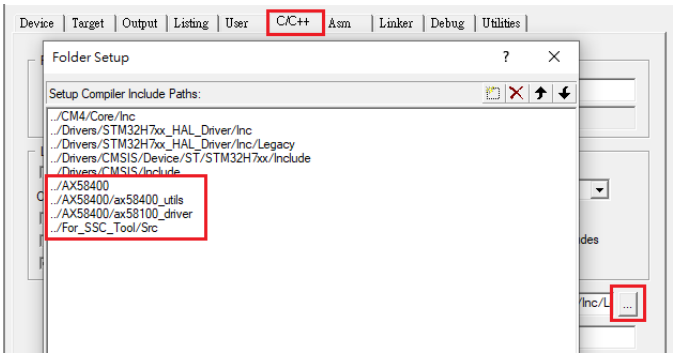

Figure 5-33

Step 3: Setup Utilities.
Click on "**Project > Options for Target 'xxxx'… > Utilities > Settings**", then select "**Flash Download**" page to add "**Programming Algorithm**" as the value below.

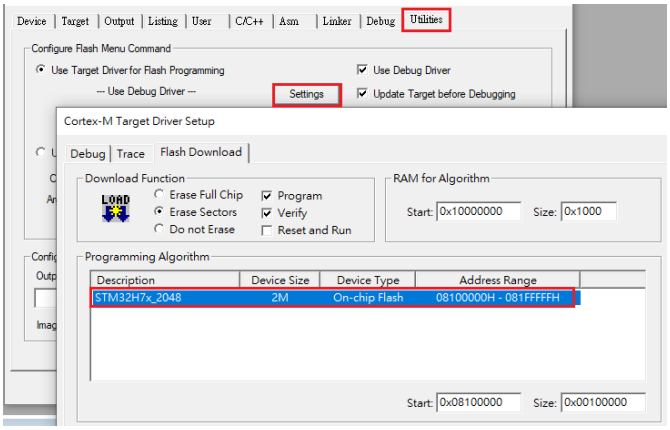

Figure 5-34

For CM4 core: Start = 0x08100000, Size = 0x00100000
For CM7 core: Start = 0x08020000, Size = 0x000E0000

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

Step 4: Add post build actions.
Find the setting of "**Project > Options for Target 'xxxx'… > User > After Build/Rebuild**", then add actions as the value below.

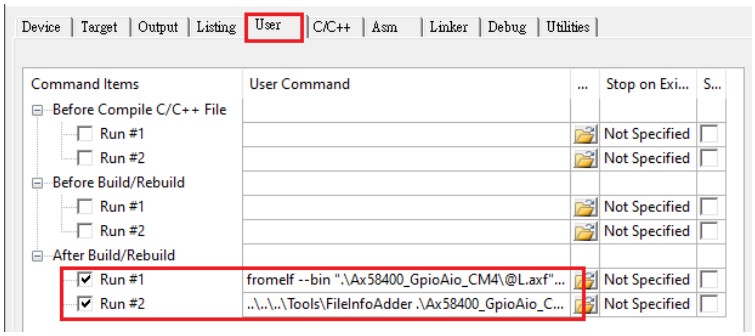

Figure 5-35

For CM4 core:
Run #1: **fromelf --bin ".\Ax58400_GpioAio_CM4\@L.axf" --output ".\Ax58400_GpioAio_CM4\@L.bin"**
Run #2:
..\..\..\Tools\FileInfoAdder .\Ax58400_GpioAio_CM4\Ax58400_GpioAio_CM4.bin ..\For_TwinCAT\FoE\AX58400_GpioAio_CM4.efw -ft 2

For CM7 core:
Run #1: **fromelf --bin ".\Ax58400_GpioAio_CM7\@L.axf" --output ".\Ax58400_GpioAio_CM7\@L.bin"**
Run #2:
..\..\..\Tools\FileInfoAdder .\Ax58400_GpioAio_CM7\Ax58400_GpioAio_CM7.bin ..\For_TwinCAT\FoE\AX58400_GpioAio_CM7.efw -ft 1

Step 5: Modify "**Target Options**".
Find the setting of "**Project > Options for Target 'xxxx'… > Target > Read/Write Memory Areas**", then modify it as the value below.

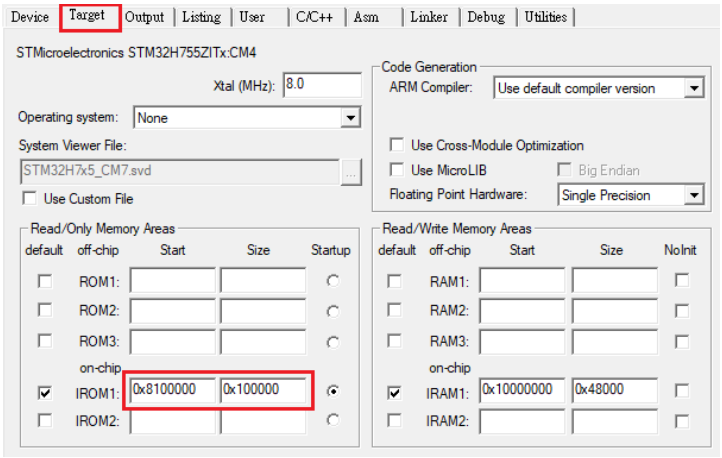

Figure 5-36

For CM4 core:
IROM Start = 0x8100000, Size = 0x100000
IRAM1 Start = 0x10000000, Size = 0x48000

For CM7 core:
IROM Start = 0x8020000, Size = 0xE0000
IRAM1 Start = 0x20000000, Size = 0x20000
IRAM2 Start = 0x0, Size = 0x10000

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

Step 6: Exclude source build on CM7.
Switch to Ax58400_GpioAio_CM7 project, then exclude the modules from target build as the figure below.

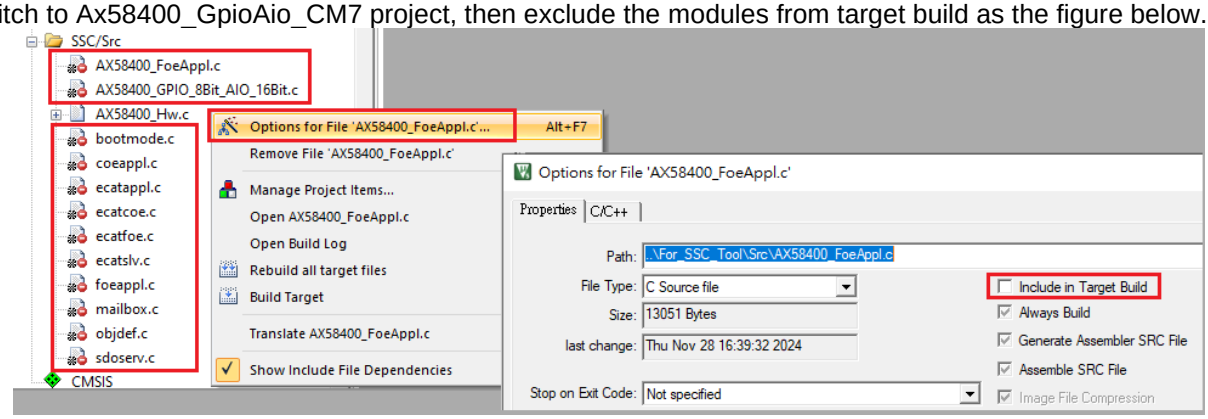

Figure 5-37

Now, entire building procedure has done for Keil MDK IDE.

- **For STM32CubeIDE:**

Double clicks the file as below to open project.
BSP_ROOT\SampleCode\Ax58400_GpioAio\STM32CubeIDE\.project

Step 1: Add source code.
On "**Project Explorer**", add virtual folders and linking the source files for CM4 and CM7 respectively as below.

Figure 5-38

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

- **Add Virtual Folder:** Right click on target project and select "**New > Folder**", then enter new folder name and select "**Folder is not located in the file system (Virtual Folder)**" to create a folder.

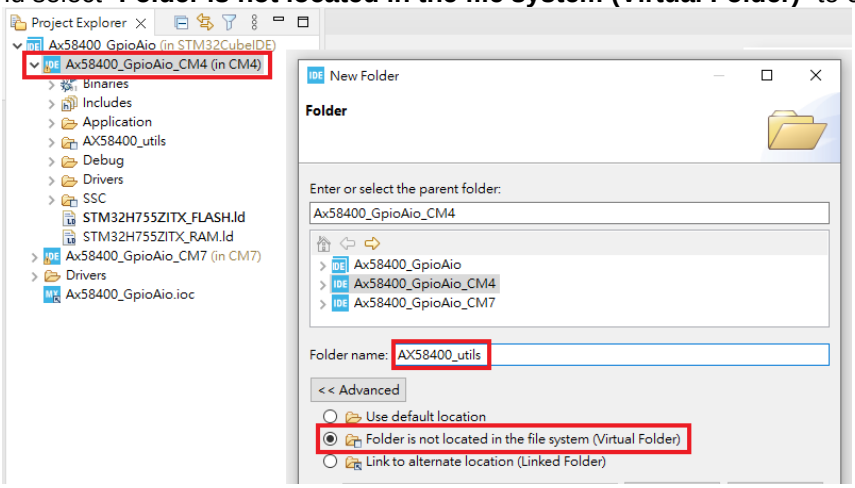

Figure 5-39

- **Add File Link:** Select and drag the files onto the created virtual folder.

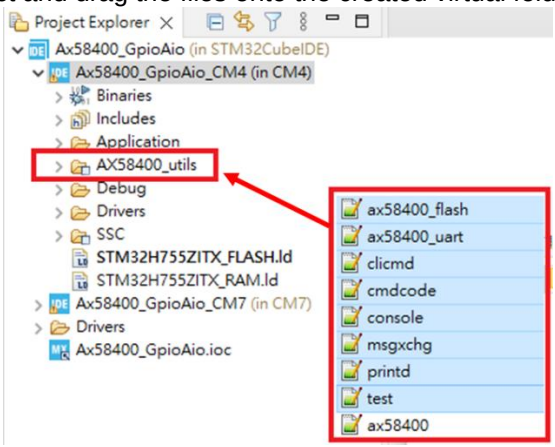

Figure 5-40

Step 2: Enable binary & Hex output files conversion on dual cores.
Select target project, then find setting on "**Project > Properties > C/C++ Build > Settings > Tool Settings > MCU Post build outputs**", then give the setting as highlighted as below.

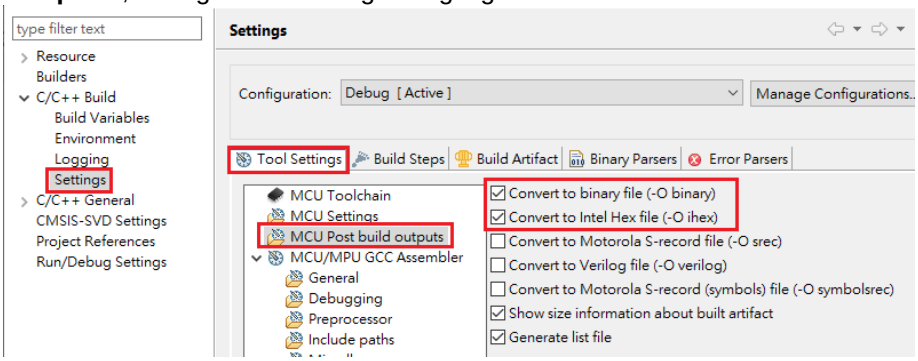

Figure 5-41

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

Step 3: Add include path both on dual cores.
Select target project, then find setting on "**Project > Properties > C/C++ Build > Settings > Tool Settings > MCU/MCP GCC Compiler > Include paths**", then add paths as the figure below.

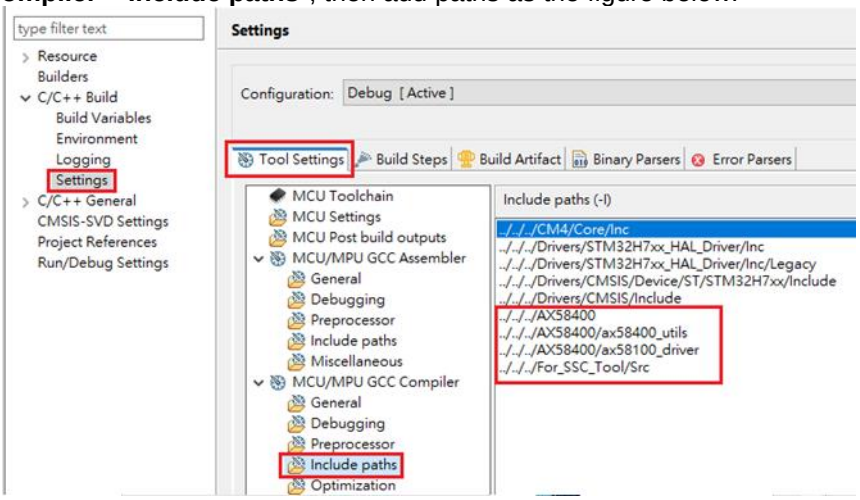

Figure 5-42

Step 4: Add post build actions.
Find the setting of "**Project > Properties > C/C++ Build > Settings > Build Steps > Post-build steps**", then add actions as the value below.

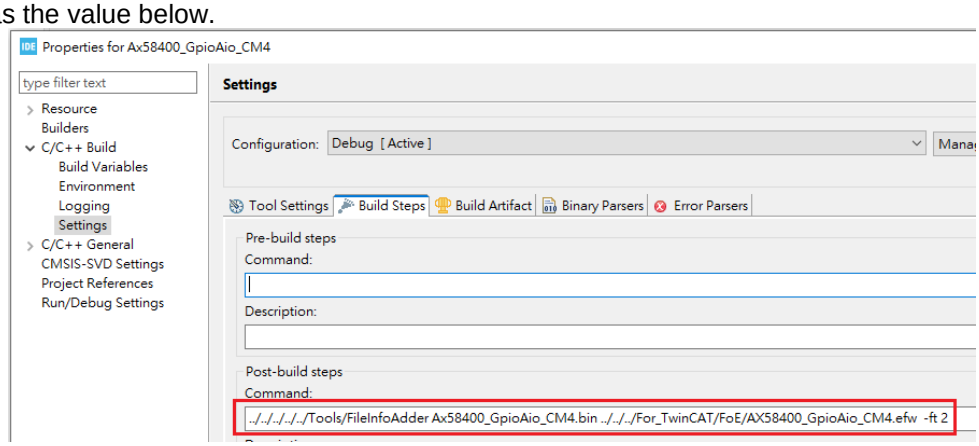

Figure 5-43

For CM4 core:
../../../../../Tools/FileInfoAdder Ax58400_GpioAio_CM4.bin ../../../For_TwinCAT/FoE/AX58400_GpioAio_CM4.efw -ft 2

For CM7 core:
../../../../../Tools/FileInfoAdder Ax58400_GpioAio_CM7.bin ../../../For_TwinCAT/FoE/AX58400_GpioAio_CM7.efw -ft 1

Now, entire building procedure has done for STM32CubeIDE.

So far, the entire project source has been ready for building firmware binary.

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

6. Build Up and Download Firmware

Here will introduce how to build up firmware binary in your IDE, and how to download firmware using IDE or other utilities

6-1 Build up Firmware Binary

- **For Keil MDK IDE:**

Step 1: Double clicks on project file in the folder below to open project.
BSP_ROOT\SampleCode\Ax58400_GpioAio\MDK-ARM\Ax58400_GpioAio.uvprojx.

Step 2: Execute "**Rebuild**" under "**Menu Bar > Project > Batch Build…**" to build CM4 & CM7 firmware binary files.

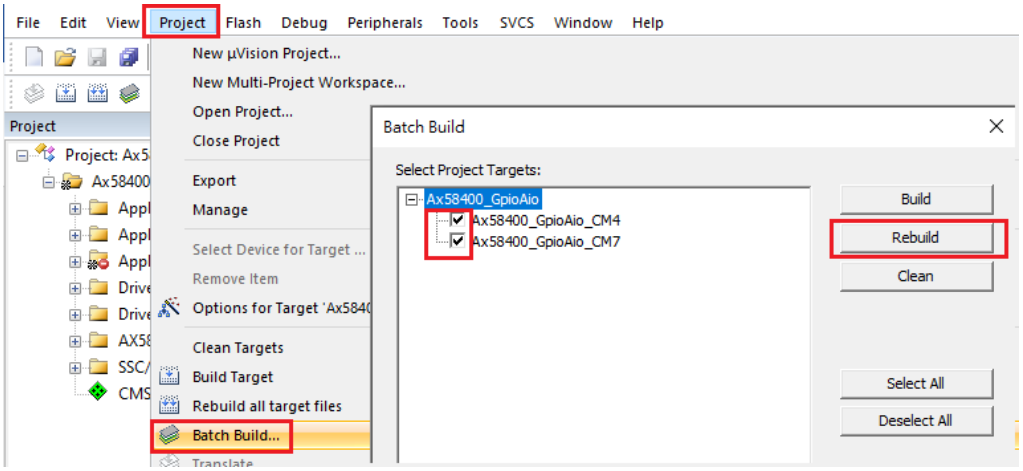

Figure 6-1

- **For STM32CubeIDE:**

Step 1: Double clicks the project file as below path to open project.
BSP_ROOT\SampleCode\Ax58400_GpioAio\STM32CubeIDE\.project

Step 2: Execute "**Build All**" under "**Menu Bar > Project**" to build CM4 & CM7 firmware binary files.

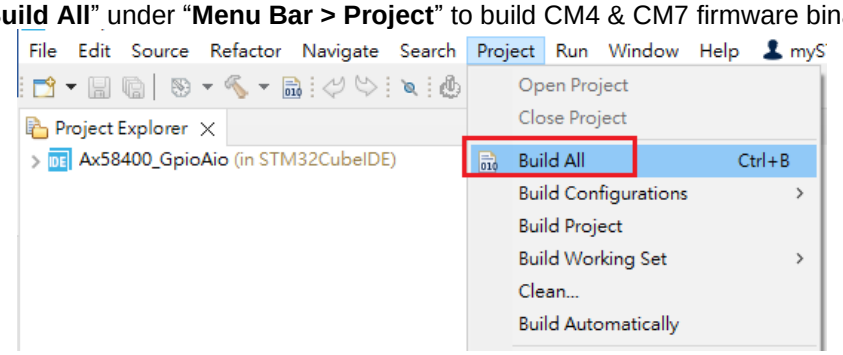

Figure 6-2

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

6-2 Download Bootloader Firmware

Step 1: Open STM32CubeProgrammer utility and connect to the AX58400-TSB board via STLINK-V3MINI programmer.

Step 2: Select the "**Erasing & Programming**" page, then click "**Browse**" button to select bootloader firmware binary file in the path below.
BSP_ROOT\SampleCode\Ax58400_Bldr\Binary\Ax58400_Bldr_CM7_vX.X.X

Please ensure the "**Start address**" is 0x08000000.

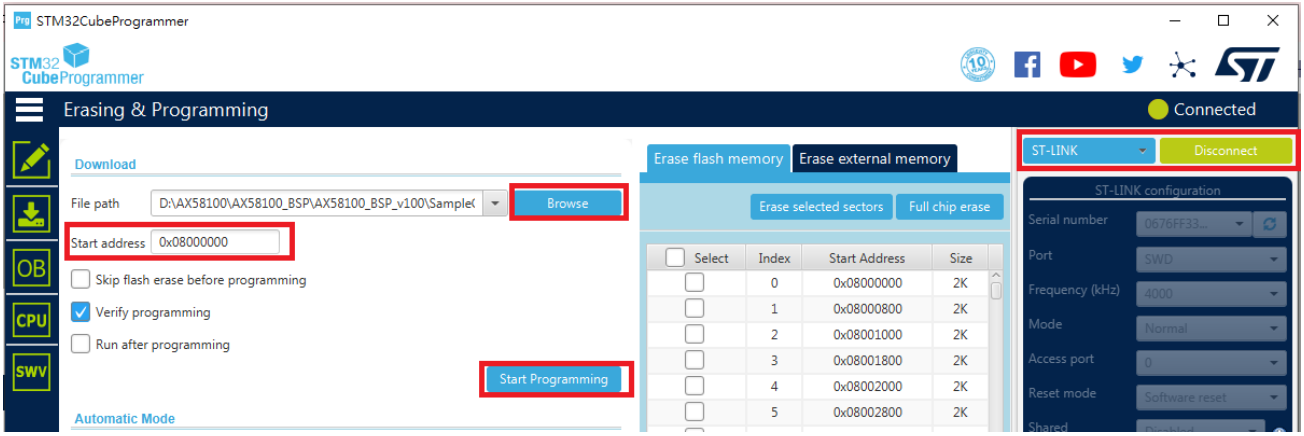

Figure 6-3

Step 3: Press "**Start Programming**" button to download bootloader firmware.

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

6-3 Download Application Firmware

- **For Keil MDK IDE:**

Step 1: Select CM4 & CM7 projects respectively, then find the "**Menu bar > Project > Options for Target 'xxxx' > Debug**" tab, select "**ST-Link Debugger**" and click "**Settings**" button, select "**Pack**" tab and uncheck "**Enable**" item.

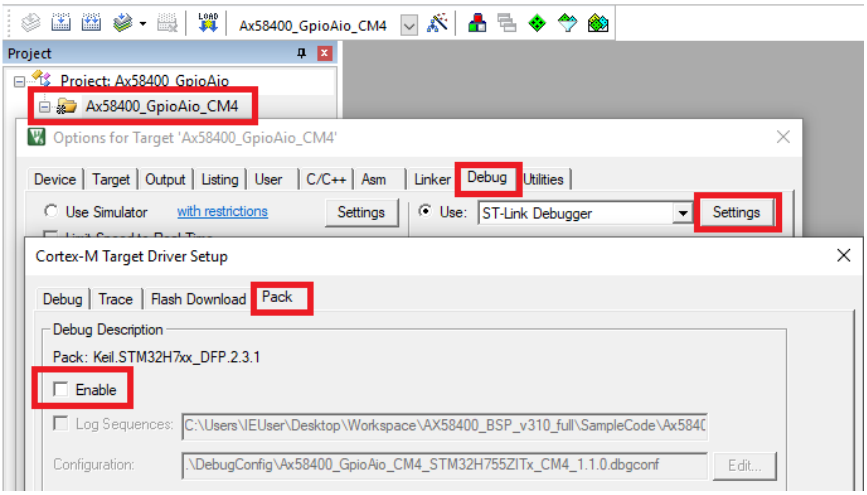

Figure 6-4

Step 1: Select target project then click "**Menu bar > Flash > Download**" to download CM4 & CM7 application firmware respectively.

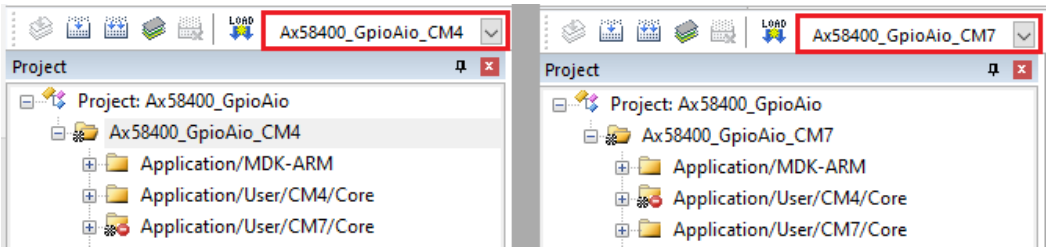

Figure 6-5

- **For STM32CubeIDE:**

Step 1: Select target project and click "**Menu bar > Run > Run As > STM32 C/C++ Application**" to download CM4 & CM7 application firmware respectively.

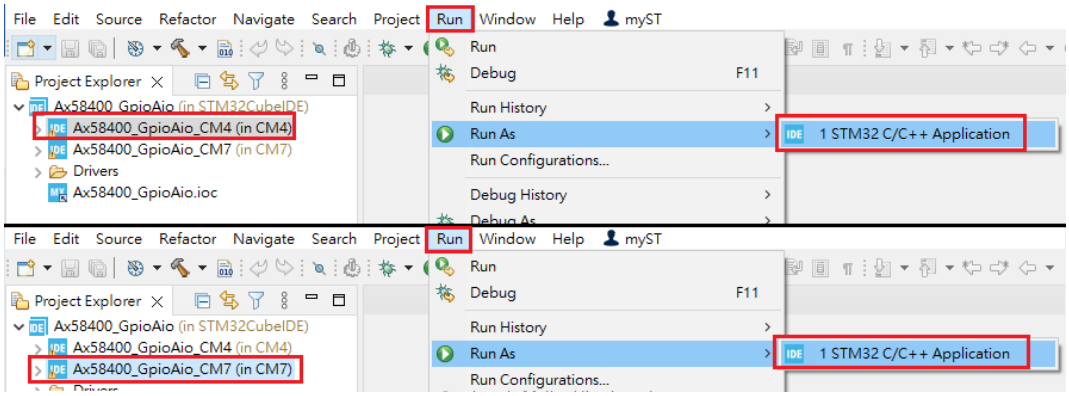

Figure 6-6

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

- **Using STM32CubeProgrammer Utility:**

Step 1: Similarly, please ensure the utility has connected to target AX58400-TSB board.

Step 2: Select the "**Erasing & Programming**" page, then click "**Browse**" button to select CM7 application firmware file in the path below.

- For Keil MDK IDE:
  BSP_ROOT\SampleCode\Ax58400_GpioAio\MDK-ARM\Ax58400_GpioAio_CM7\Ax58400_GpioAio_CM7.bin
- For STM32CubeIDE:
  BSP_ROOT\SampleCode\Ax58400_GpioAio\STM32CubeIDE\Debug\Ax58400_GpioAio_CM7.bin

Please ensure the "**Start address**" is 0x08020000.

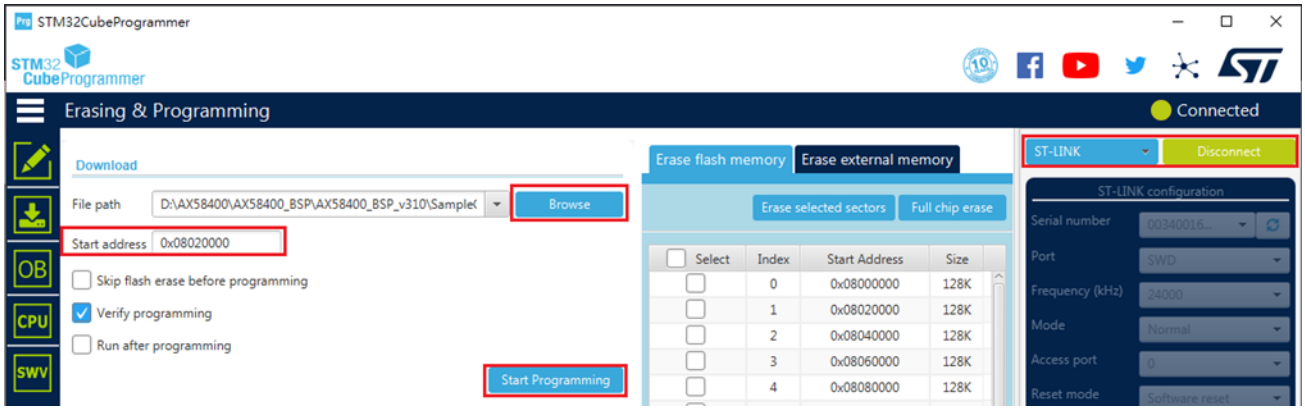

Figure 6-7

Step 3: Press "**Start Programming**" button to download CM7 application firmware.

Step 4: Similarly, for CM4 application firmware download, the firmware path and flash address are as below.

- For Keil MDK IDE:
  BSP_ROOT\SampleCode\Ax58400_GpioAio\MDK-ARM\Ax58400_GpioAio_CM4\Ax58400_GpioAio_CM4.bin
- For STM32CubeIDE:
  BSP_ROOT\SampleCode\Ax58400_GpioAio\STM32CubeIDE\Debug\Ax58400_GpioAio_CM4.bin

Please ensure the "**Start address**" is 0x08100000.

Step 5: Press "**Start Programming**" button to download CM4 application firmware.

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

7. Basic Operation of TwinCAT

Please refer to Section 3 of “**AX58x00_TwinCAT_UserGuide**” for more details.

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

8. PLC Application

Please refer to Section 5 of “**AX58x00_TwinCAT_UserGuide**” for more details.

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

9. Flash Memory Allocation and FoE Upgrading

In order to support “File Access over EtherCAT” (FoE) feature, this reference design has the flash memory allocation on the AX58400 as below.

Flash Area	Flash Memory Addresses	Size (bytes)
Bootloader	0x08000000 ~ 0x0801FFFF	128K
CM7 Application	0x08020000 ~ 0x0809FFFF	512K
Reserved	0x080A0000 ~ 0x080FFFFF	384K
CM4 Application	0x08100000 ~ 0x0817FFFF	512K
Flash Buffer	0x08180000 ~ 0x081FFFFF	512K

Table 9-1

- **Bootloader Area** (Address=0x08000000, Size=128Kbytes):
  The Bootloader firmware must be pre-burn in this area, it’ll responsible for coping the received new firmware from “Flash buffer area” into “Runtime image area”, then executes the new firmware.

- **CM7 Application Area** (Address=0x08020000, Size=512Kbytes):
  Used to run ADC, DAC and Digital Input/Output operation firmware.

- **CM4 Application Area** (Address=0x08100000, Size=512Kbytes):
  Used to run EtherCAT Slave Code (SSC) firmware, it is also responsible for store the received .efw file into “Flash buffer area” via FoE data transfer.

- **Flash Buffer Area** (Address=0x08180000, Size=512Kbytes):
  Used to store entire .efw file.

The .efw file is dedicate use for FoE transfer, it’s a firmware binary file with 32-bytes header pre-fix.
Every time the project build done, the .efw file will auto be generated in the path below by “**FileInfoAdder.exe**” utility.
BSP_ROOT\SampleCode\Ax58400_GpioAio\For_TwinCAT\FoE\AX58400_GpioAio_CM4.efw
BSP_ROOT\SampleCode\Ax58400_GpioAio\For_TwinCAT\FoE\AX58400_GpioAio_CM7.efw

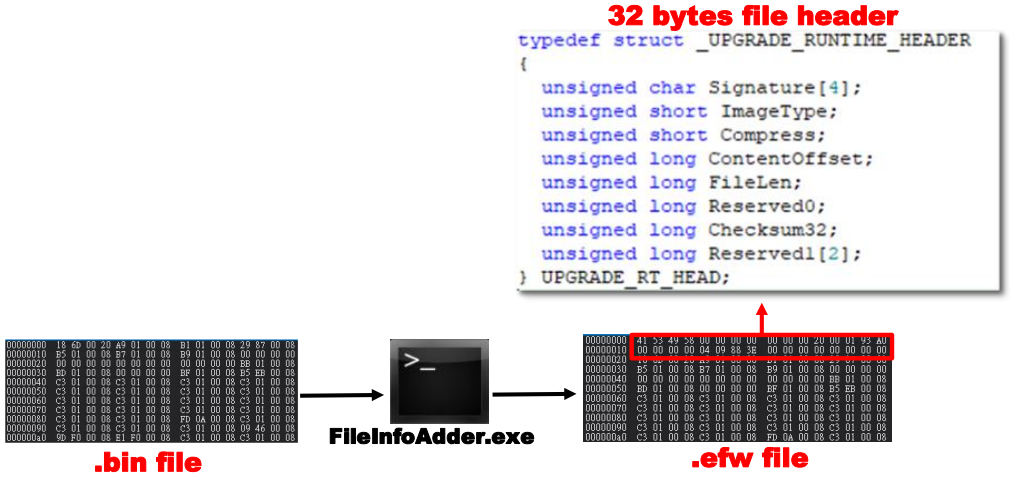

Figure 9-1

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

9-1 FoE Firmware Upgrade

This section shows the steps of firmware update with EtherCAT FoE.

Step 1: Select "**Solution Explorer > AX58400_GpioAio > I/O > Devices > XXX > Online**", then right click on target device and select "**Firmware Update**".

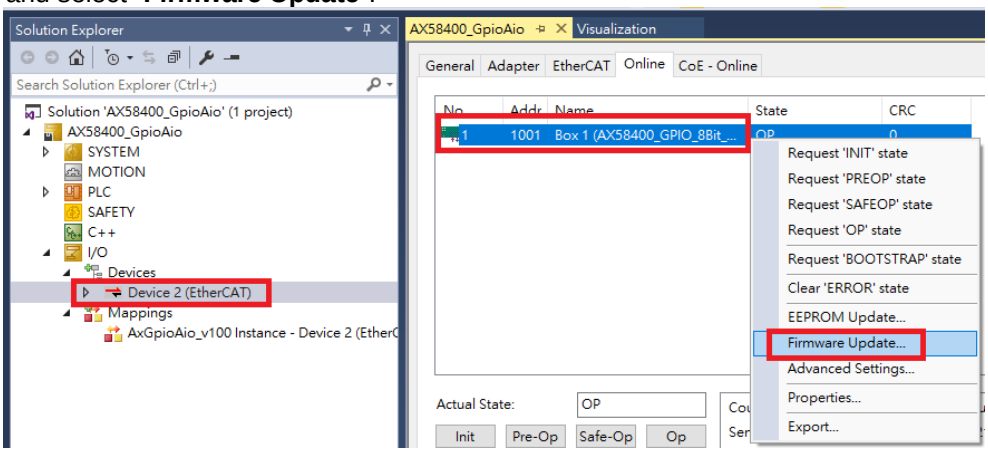

Figure 9-2

Step 2: Select the file as path below, and click "**OK**" to start CM4 application firmware upgrading.
BSP_ROOT\SampleCode\Ax58400_GpioAio\For_TwinCAT\FoE\AX58400_GpioAio_CM4.efw

Step 3: Check the progress bar status and waiting for the processing done.

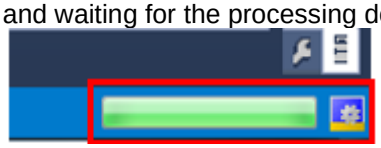

Figure 9-3

Step 4: If the firmware upgrading is complete successfully, the pop up dialog will appear as the figure below.

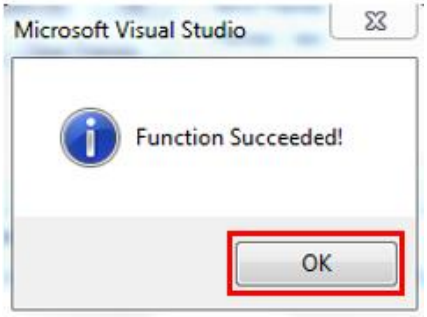

Figure 9-4

Step 5: Similarly, upgrading the CM7 application firmware by select the file as path below.
BSP_ROOT\SampleCode\Ax58400_GpioAio\For_TwinCAT\FoE\AX58400_GpioAio_CM7.efw

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

10. ADC / DAC Description

AX58400 has ADC / DAC in 12-bit resolution. This application demonstrates them through AX58400-EXB-ADIO board. If you want to observe or measure voltage of ADC / DAC, please refer to below figure:

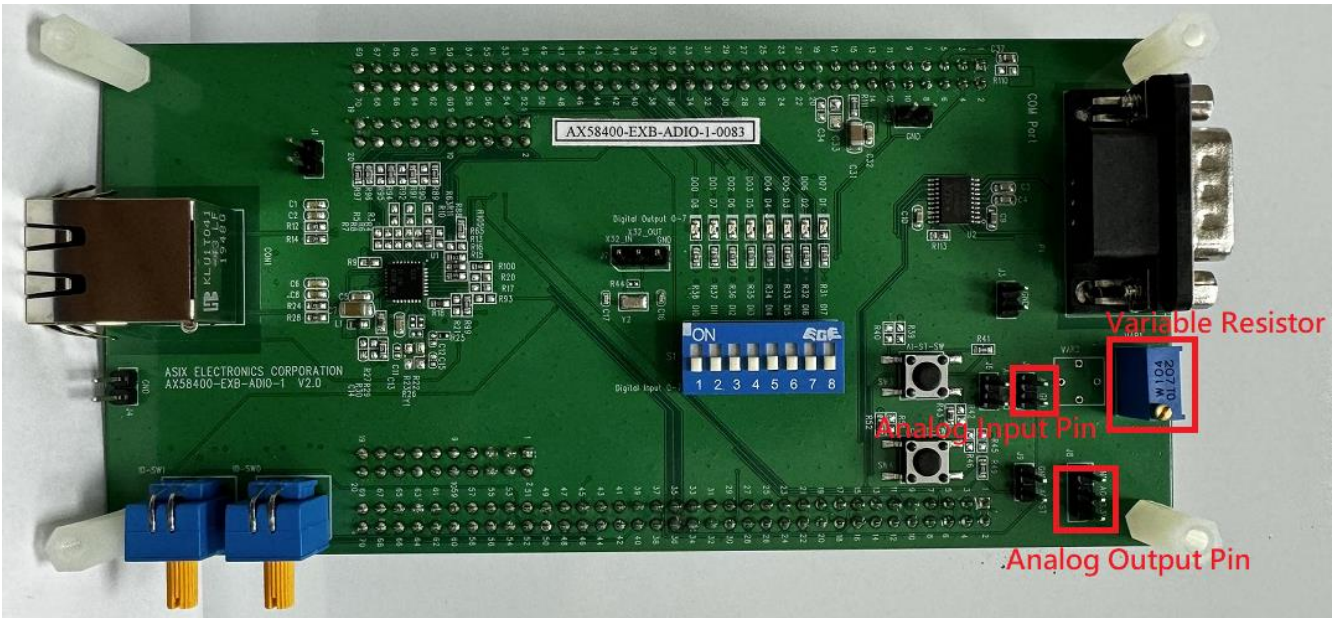

Figure 10-1

You can adjust the "**Variable Resistance**" for changing input voltage of ADC:

- Clockwise rotation – Resistance ↓ and input voltage ↓
- Counterclockwise rotation - Resistance ↑ and input voltage ↑

And change DAC output hex value in PLC GUI for control output voltage of DAC.

- Output hex value ↓ - Output voltage ↓
- Output hex value ↑ - Output voltage ↑

Assume measured VCC = 3.27v, the voltage of ADC / DAC will be:

- Analog Input Voltage = 3.16 x (ADC_HEX_VALUE_IN_PLC_GUI / 0xFFFF)
- Analog Output Voltage = 3.16 x (DAC_HEX_VALUE_IN_PLC_GUI / 0xFFFF)

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

11. Switch ID Description

The Switch ID in this application is used for identification function of TwinCAT master. User can specify an ID at EtherCAT slave side and setup identification at TwinCAT master side. TwinCAT master will check if the ID of the slave matched with the expected identification value during startup of the slave. If you want to know how to setup identification, please follow below example.

You can find out the “IdentificationReg134” setting in ESI file after SSC Tool generated source files and ESI file through import ASIX’s configuration file.

```
<Info>
    <StateMachine>
        <Timeout>
            <PreopTimeout>2000</PreopTimeout>
            <SafeopOpTimeout>9000</SafeopOpTimeout>
            <BackToInitTimeout>5000</BackToInitTimeout>
            <BackToSafeopTimeout>200</BackToSafeopTimeout>
        </Timeout>
    </StateMachine>
    <Mailbox>
        <Timeout>
            <RequestTimeout>100</RequestTimeout>
            <ResponseTimeout>2000</ResponseTimeout>
        </Timeout>
    </Mailbox>
    <IdentificationReg134>true</IdentificationReg134>
</Info>
```

Figure 11-1

This setting will make TwinCAT master enable the identification function for the slave.
Select the slave, right-click and select “**Advanced Settings…**”.

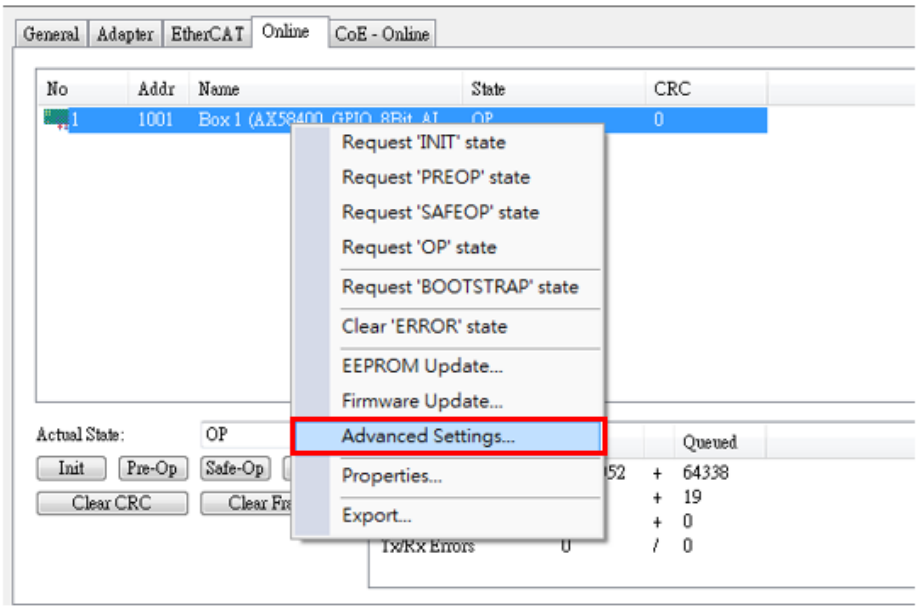

Figure 11-2

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

Go to “**General -> Behavior**” and turn on the “**Check Identification**”.

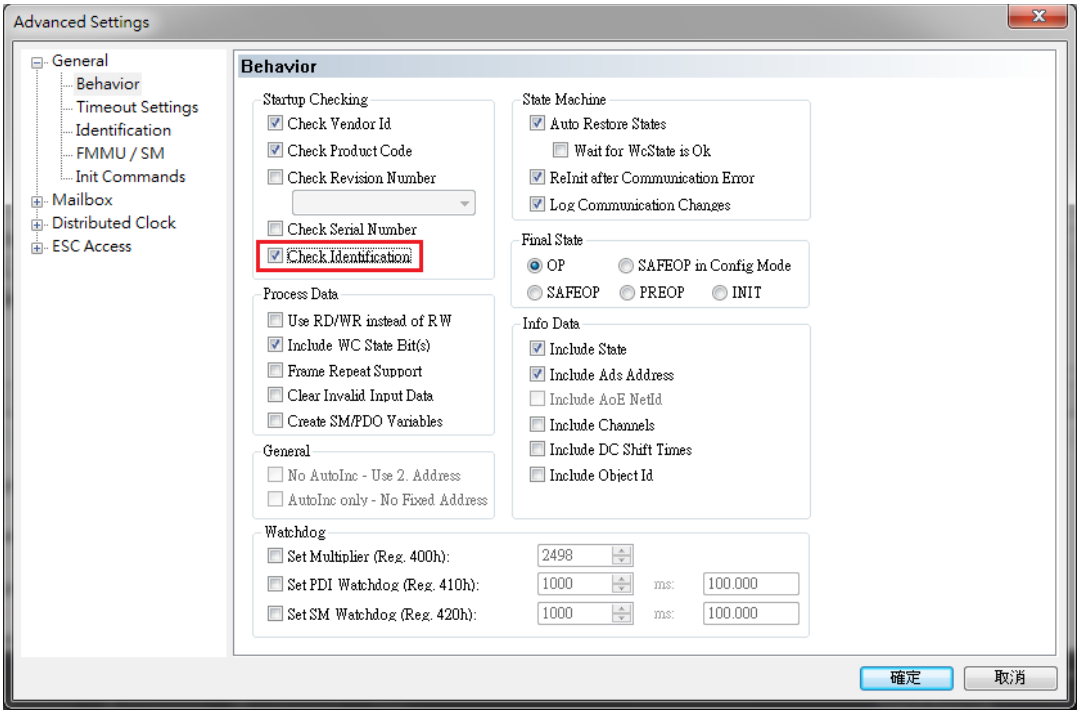

Figure 11-3

Confirm current identification ADO (Address Offset) is “**(ADO 0x0134)**”.
Don’t forget to assign the value in decimal. The value “**55**” is just an example here.

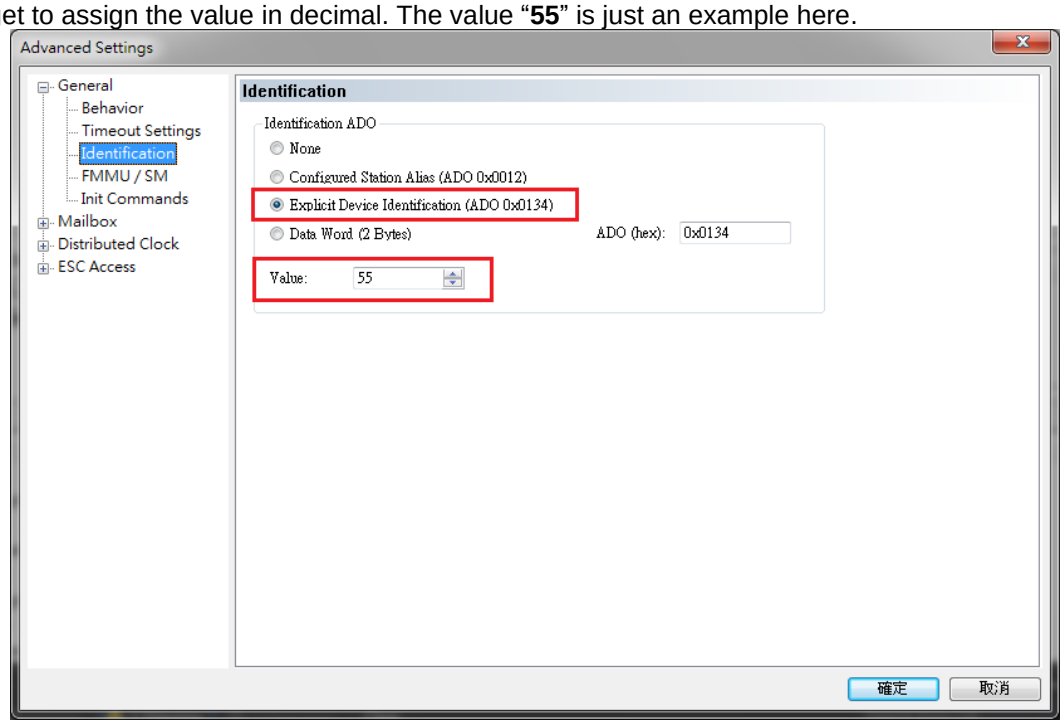

Figure 11-4

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

Of course, you should also setup the same value on AX58400-EXB-ADIO board like following:
The digit format “**Rotary Dip Switch**” is hex so that we should set it to “**0x37**”.

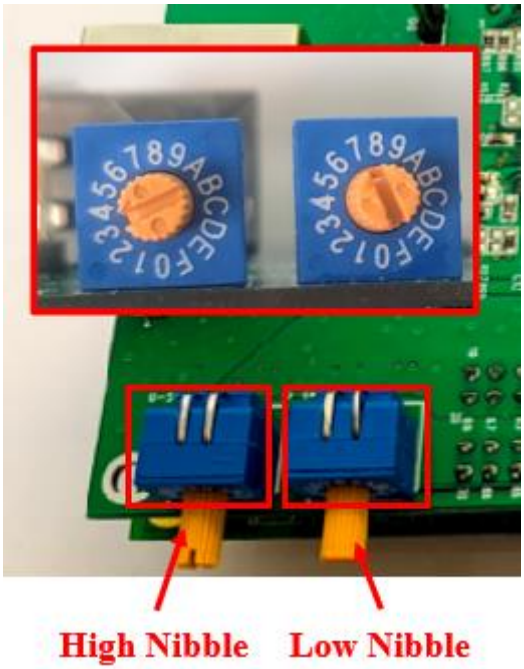

Figure 11-5

Reboot the AX58400-TSB board by click “Reset” button.

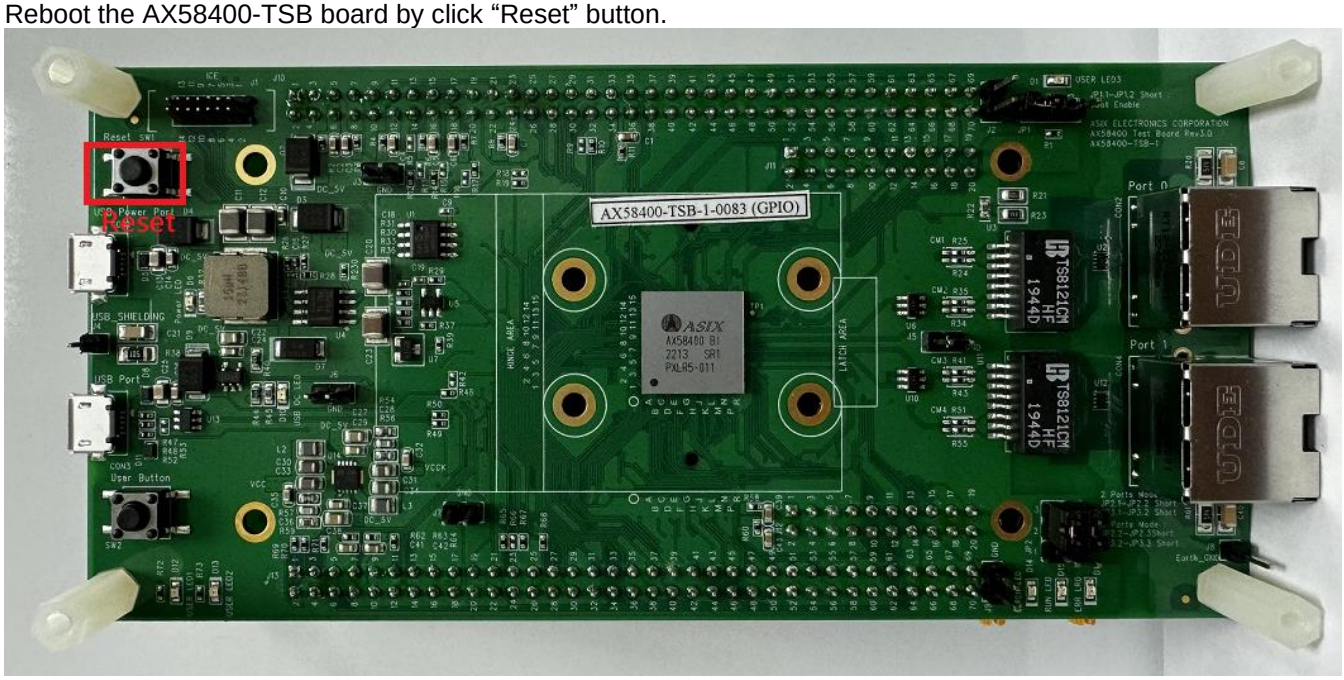

Figure 11-6

Re-activate the TwinCAT master.

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

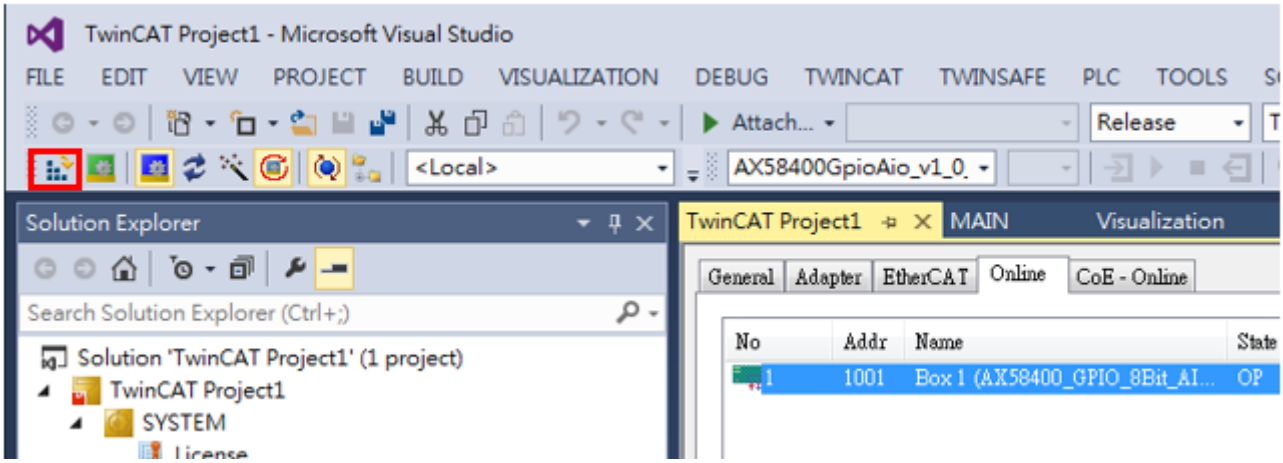

Figure 11-7

If you see the below screen, that means that TwinCAT master already enter running mode successfully and the slave also passed the identification checking so that it can enter OP state.

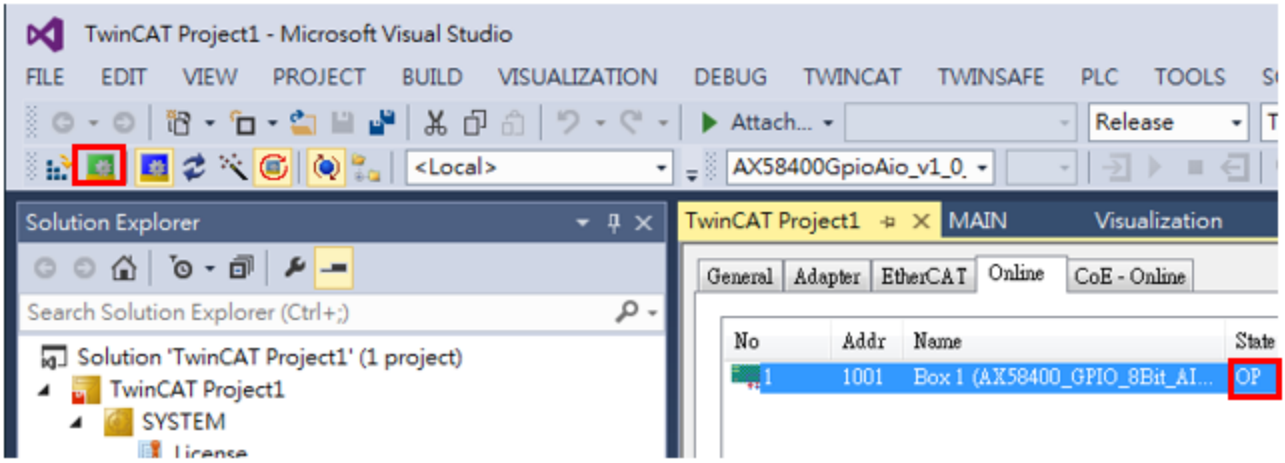

Figure 11-8

The slave will also accept frames from TwinCAT master with 2 kinds of address (1001 or 55) at the same time. But it needs to turn on second address function in ESC register. However, we arrange command for observe this behavior first.

You can append an EtherCAT command to TwinCAT master for confirm the second address.
Click “**Device N (EtherCAT) -> Append EtherCAT Cmd**”.

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

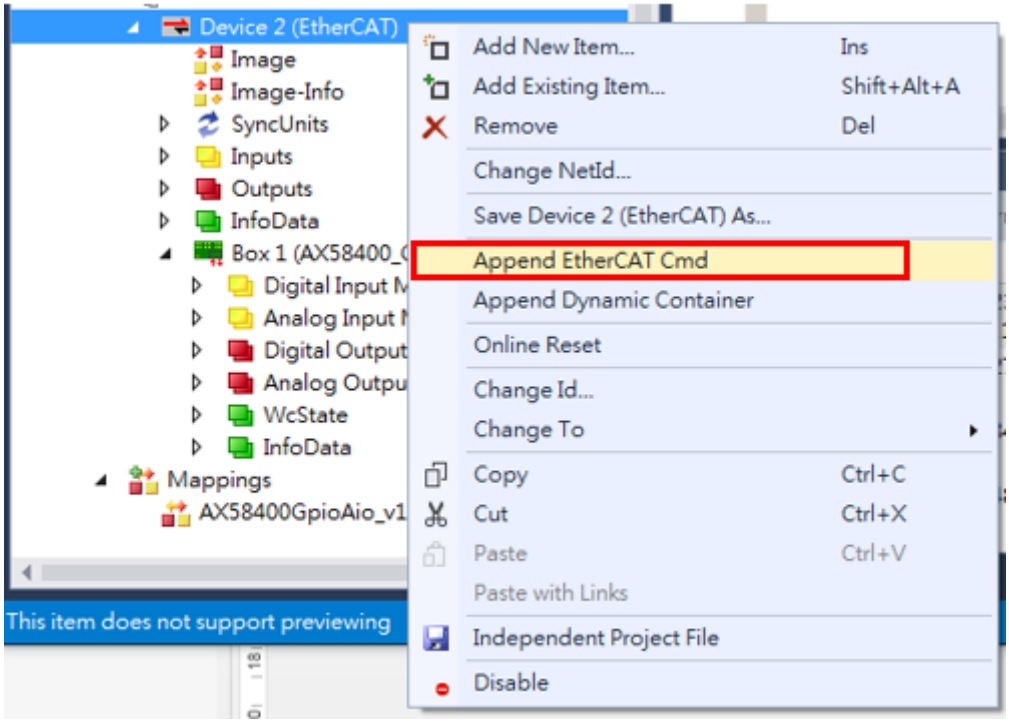

Figure 11-9

Please assign “**Command**” and “**Type**”.
I demonstrate “**FPRD**” and “**UINT**” in this example.

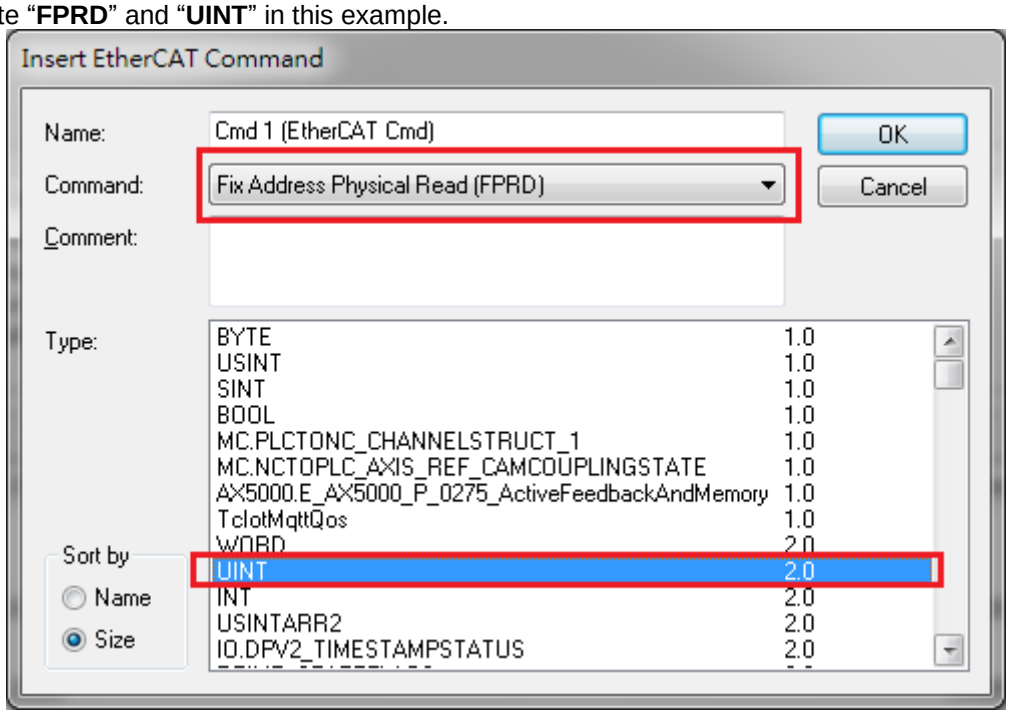

Figure 11-10

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

Double-click “**Cmd 1**” and setup “**Slave Address**” and “**Address Offset**”.
I setup slave address = 0x37 and address offset = 0x12 in this example.

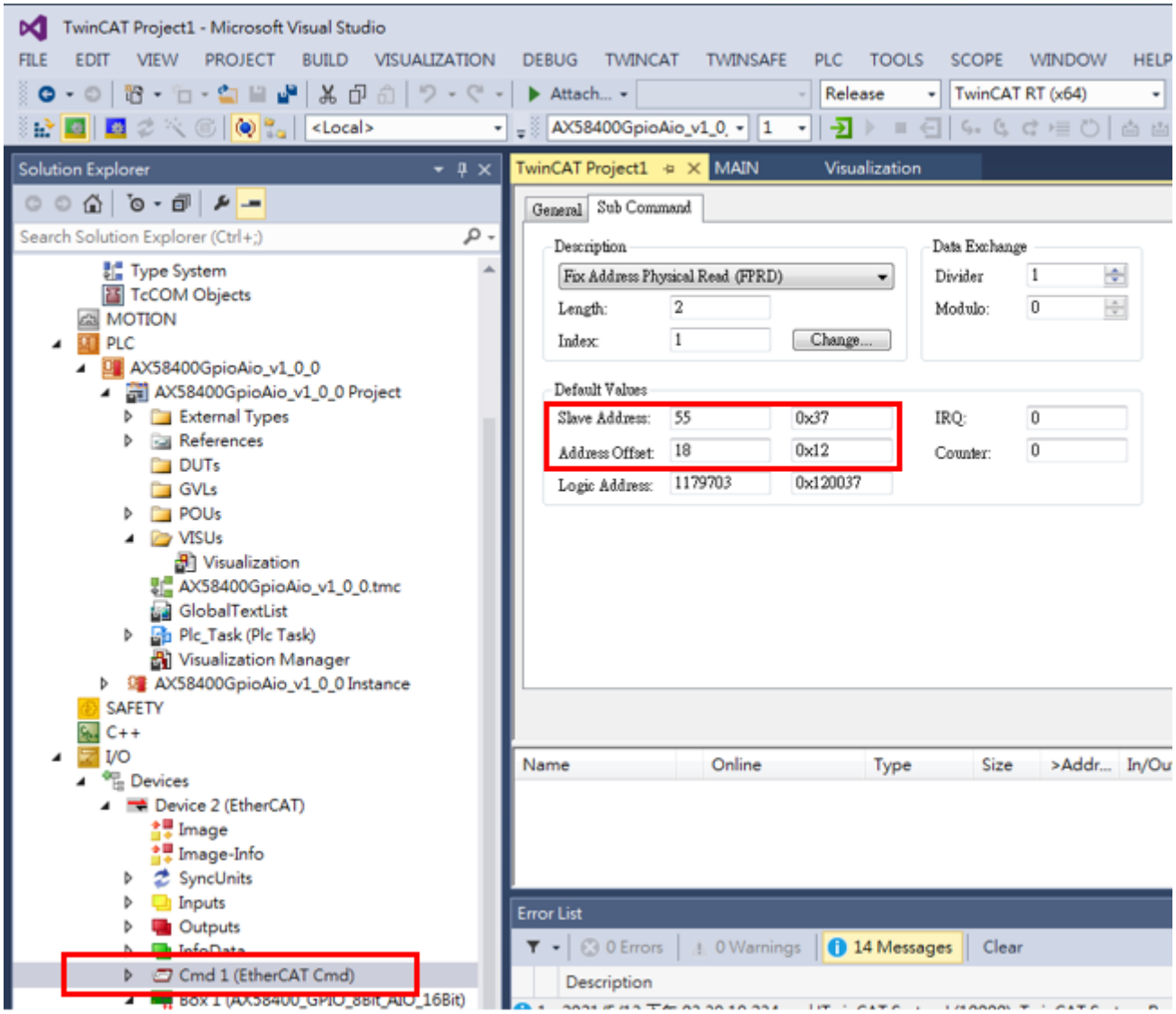

Figure 11-11

Re-activate the TwinCAT master again.

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

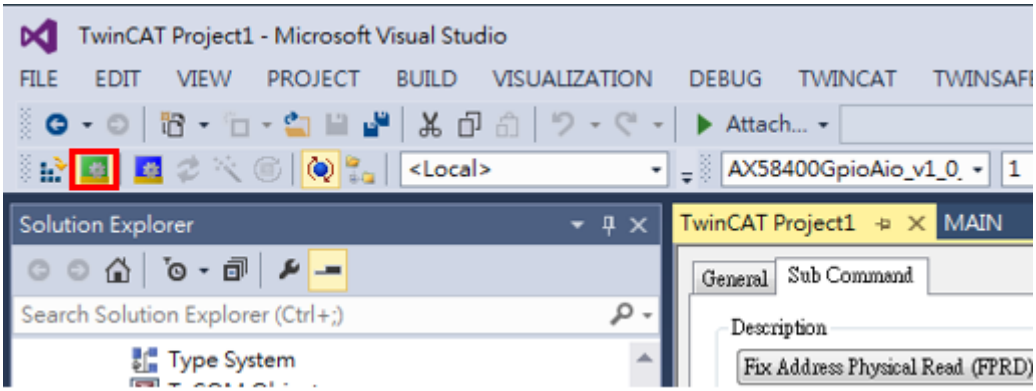

Figure 11-12

Go to "**ESC Access -> Memory**" in "**Advanced Settings**" and check out the address 0x0012.
The firmware will write the user specified Switch ID value into ESC register 0x0012 (the register named Configured Station Alias).

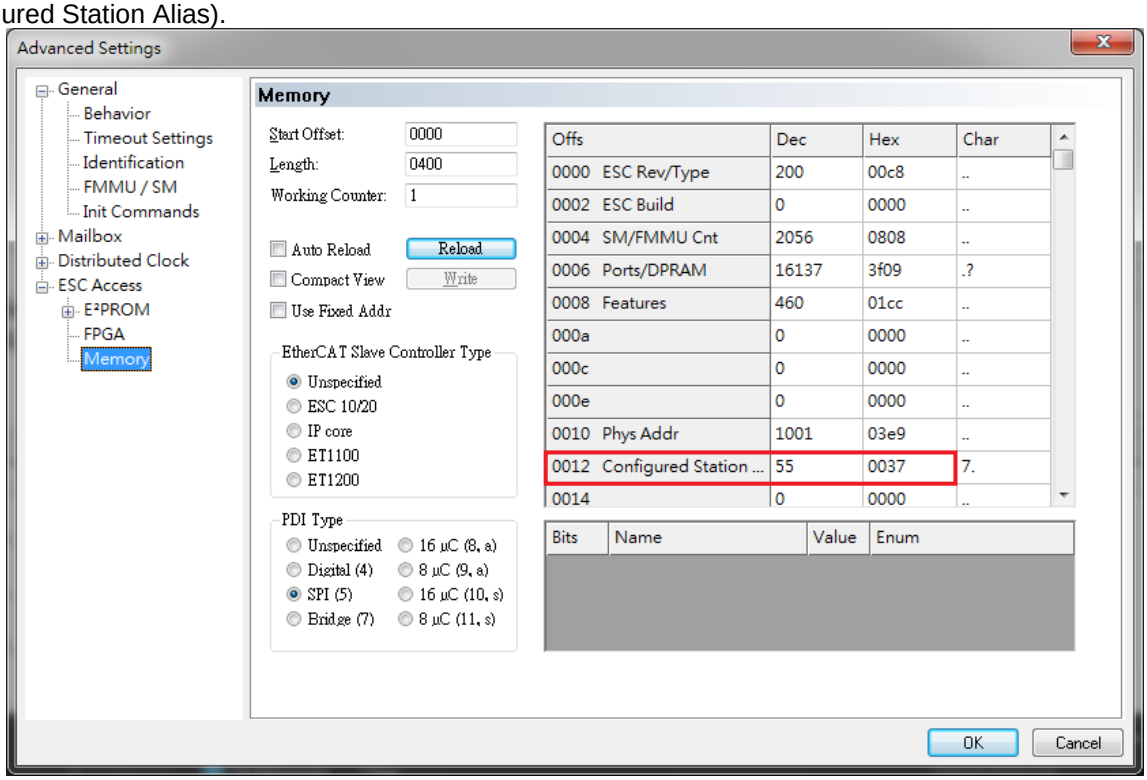

Figure 11-13

If you expect that the slave can also use this Switch ID value as second slave address. You just turn on "**Seconds address**" bit (bit8 = 1) in ESC register 0x0102 (don't forget click "**Write**" button).

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

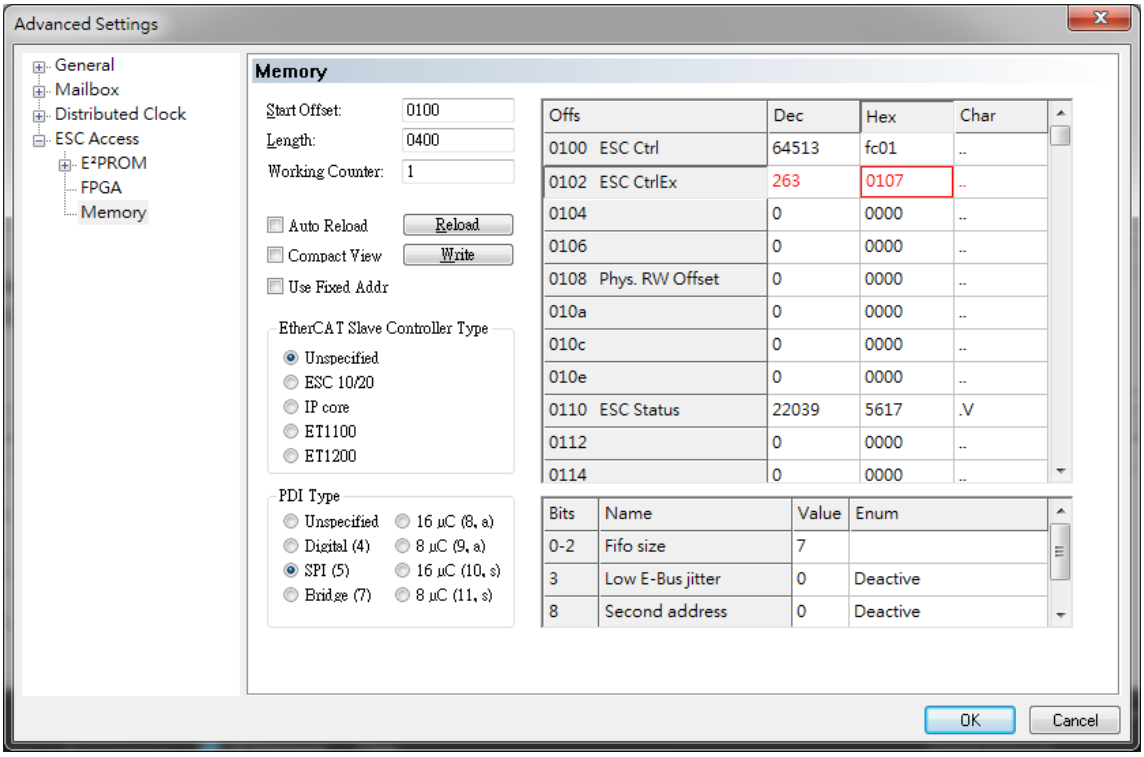

Figure 11-14

Click “**Device N (EtherCAT) -> Cmd 1 -> DATA**”, you will see the command get the value of ESC register 0x0012 back successfully. It is correct, the “**0x0037**” in UINT data type.

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

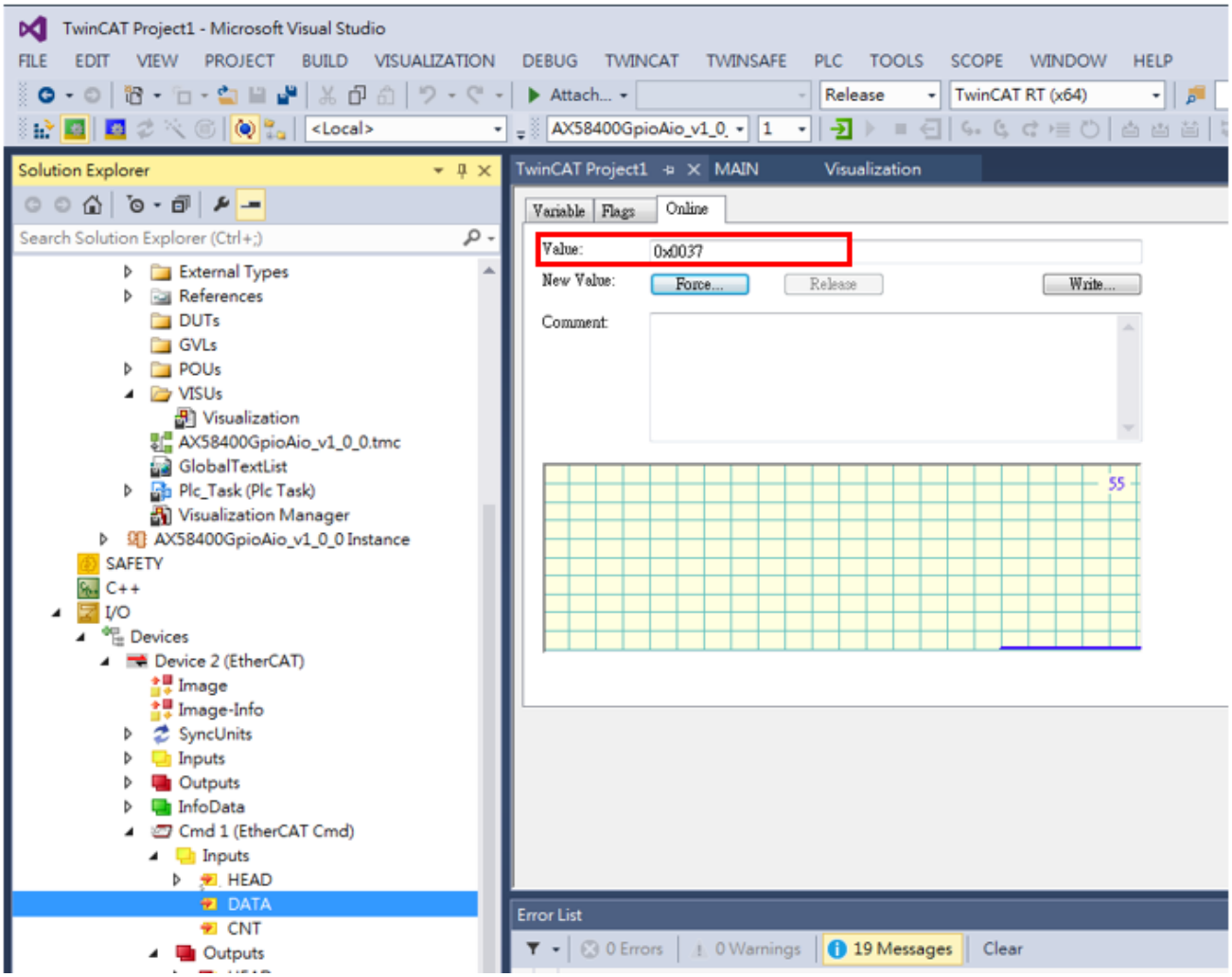

Figure 11-15

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

12. Shortest EtherCAT Cycle Evaluation

We tried to decrease the TwinCAT cycle to evaluate shortest cycle of AX58400 GPIO / AIO.
Shortest workable cycle of AX58400 GPIO / AIO is "**100us**" now.
This reference design allocates TX-PDO (3 bytes) and RX-PDO (3 bytes) totally.
Please refer to below figures.

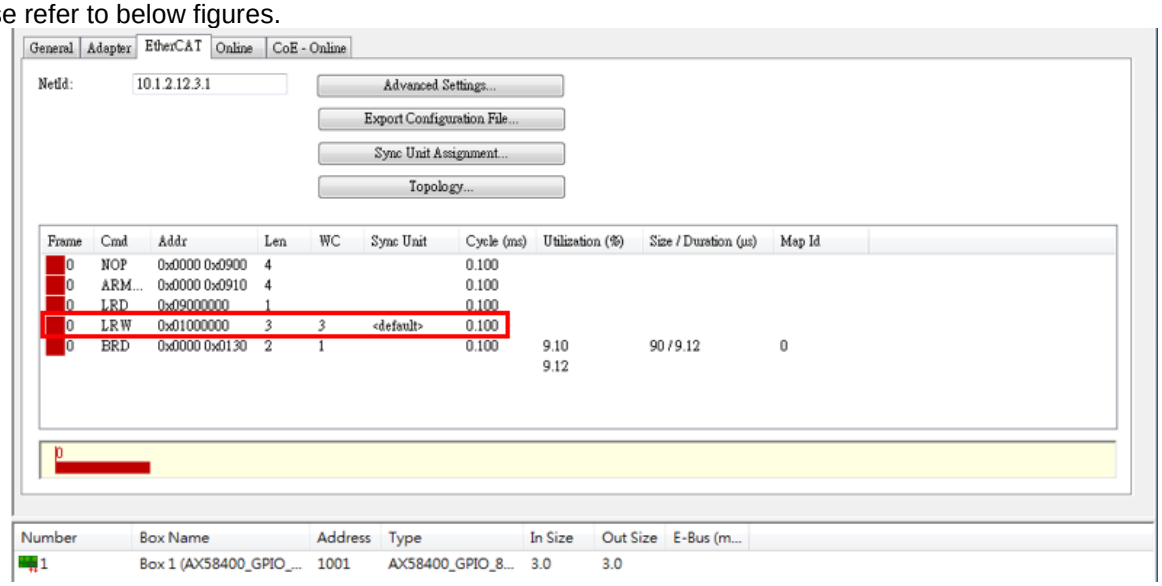

Figure 12-1

Under this condition, the "**Lost Frames**" and "**Tx/Rx Errors**" does not count.

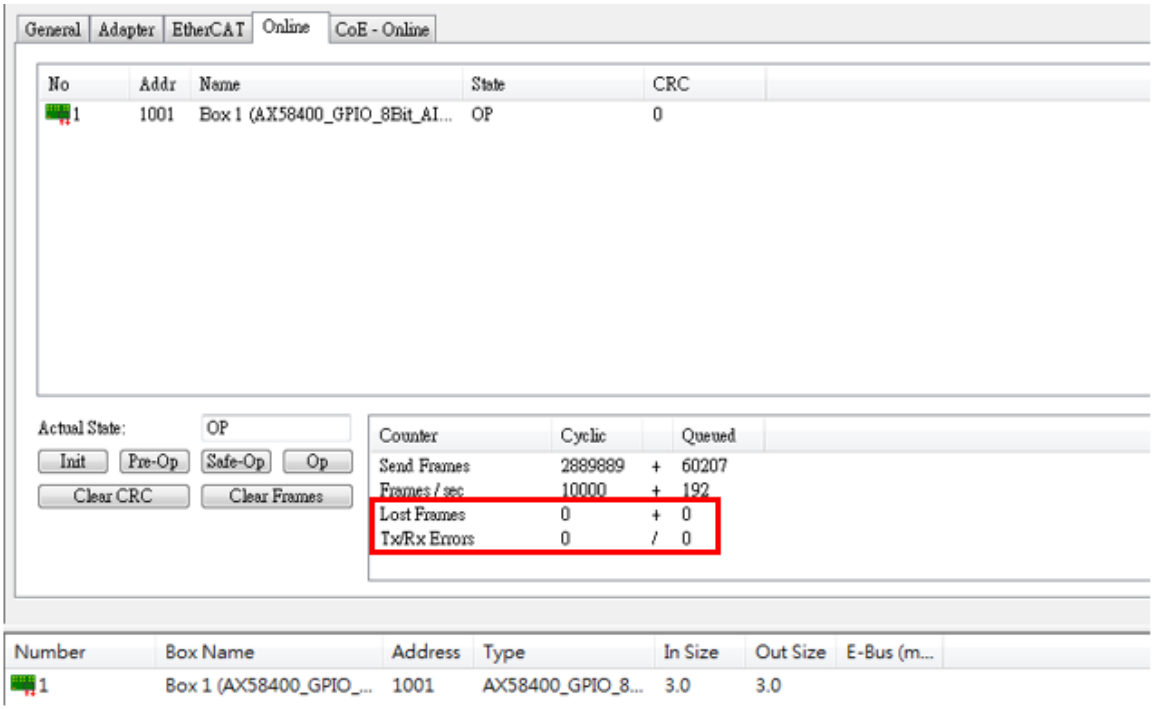

Figure 12-2

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

The slave stack also does not indicate any errors in SM output parameter / SM input parameter objects of CoE-Online table.

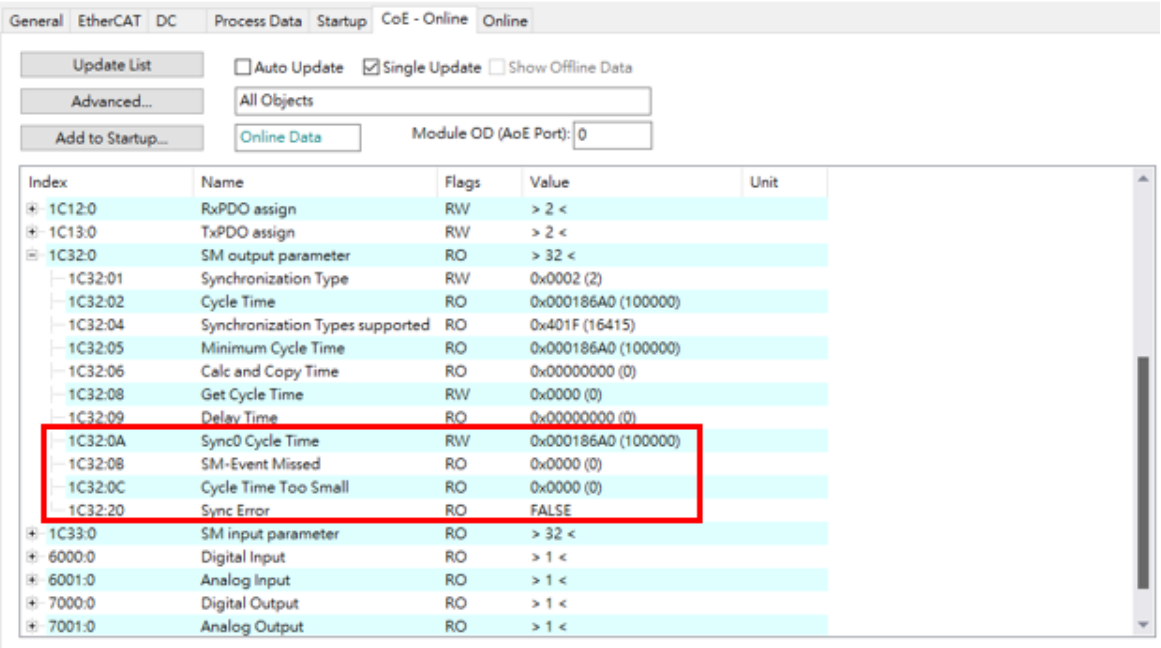

Figure 12-3

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

13. Using the High Speed Clock (HSE)

13-1 Modify RCC Setting

For this reference design, the HSI is used for default clock source, if higher precision clock is needed, please follow the steps to change the clock source to an external crystal (HSE).

Step 1: Double clicks on the file below to open STM32CubeMX.
BSP_ROOT\SampleCode\Ax58400_GpioAio\Ax58400_GpioAio.ioc

Step 2: Select "**Pinout & Configuration > RCC**" and enable "**High Speed Clock (HSE)**" as "**Crystal/Ceramic Resonator**" setting.

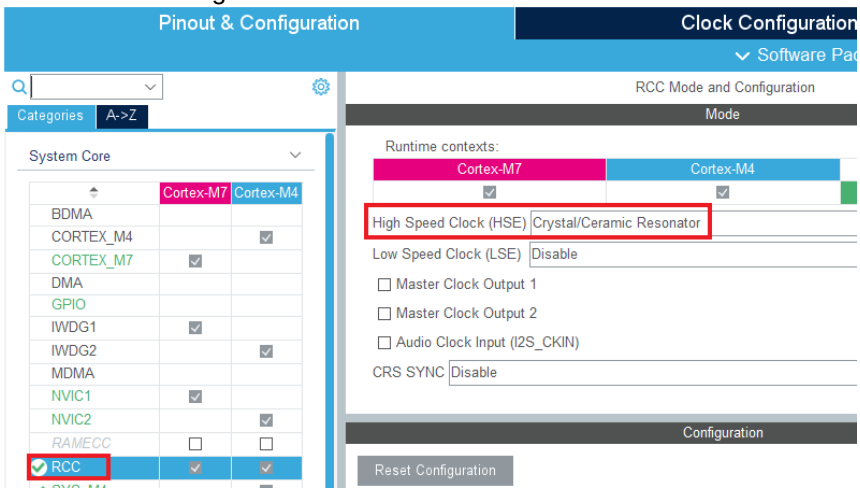

Figure 13-1

Step 3: Select "**Clock Configuration**" page, then modify the setting as highlighted in red as below.

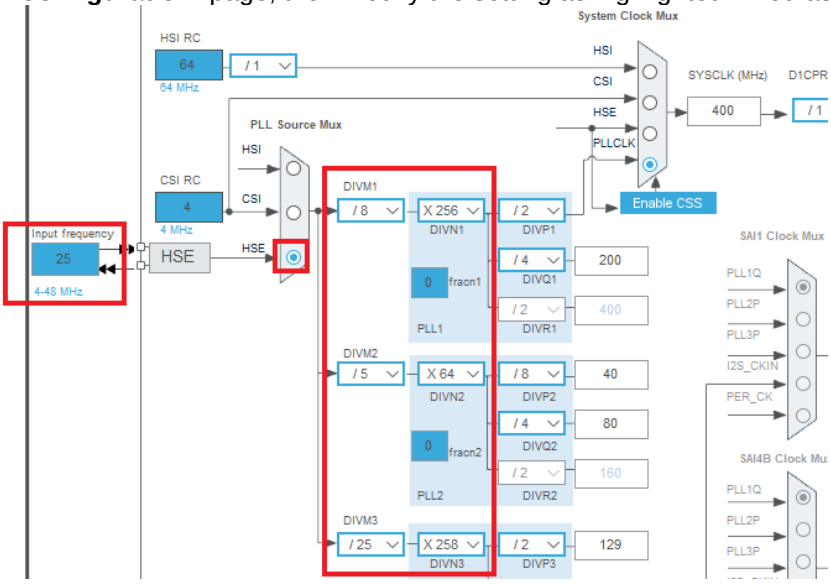

Figure 13-2

Step 4: Re-generate the source code, then re-build the target and upgrade the firmware as mentioned previously.

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

14. Firmware Architecture

Below figure shows the simple software architecture of AX58400 GpioAio Reference Design.
All the firmware modules could be classified three parts as below.

- Dual Core shared modules
- CM7 Core modules
- CM4 Core modules

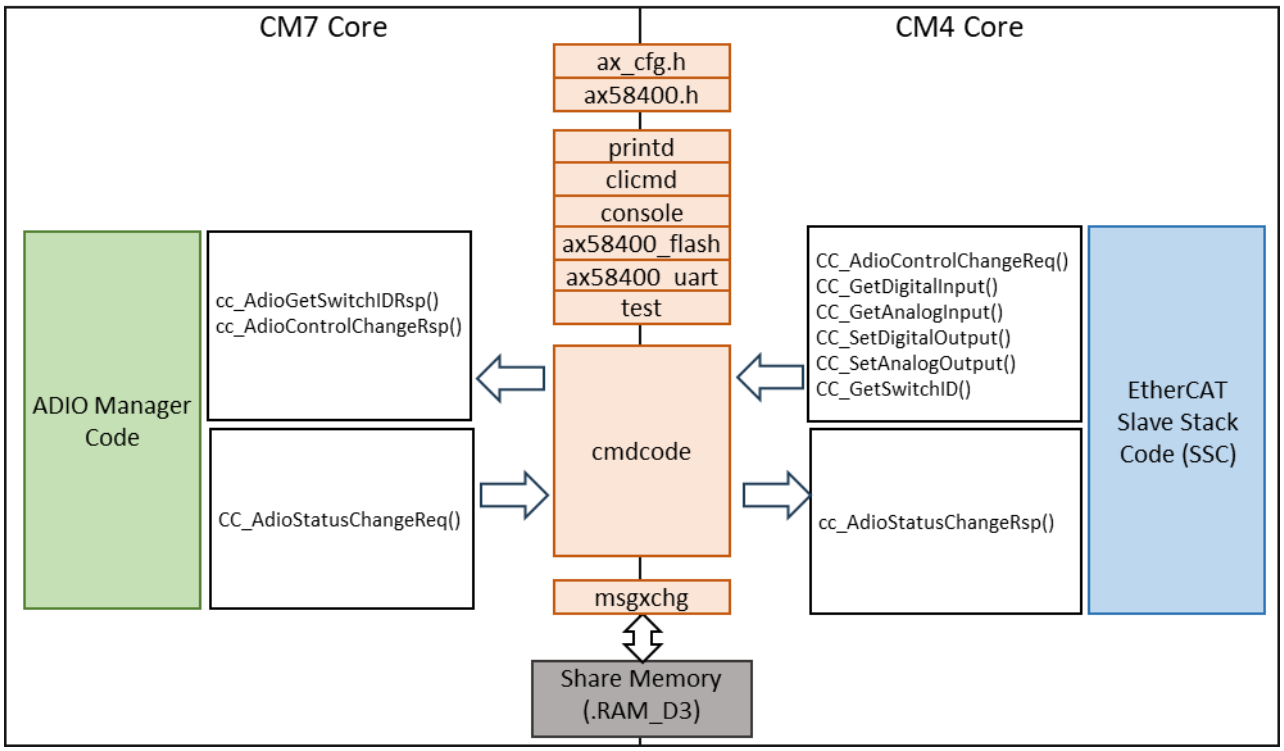

Figure 14-1

14-1 Dual Core Shared Modules

These modules are shared by CM4 and CM7 cores, some of shared modules allow be configured as suitable functions to run on CM4 or CM7 core.

- “ax_cfg.h” and “ax58400.h”
  These modules provide some configurations and allow developer to allocate suitable shared modules or functions for CM4 or CM7 core.

- “printd”, “clicmd”, “console”, “ax58400_uart” and “test”
  These modules provide a serial port (RS232 or USB VCP) for debugging or functional test during the development stage, it’s also useful for troubleshooting & diagnostics.

- “cmdcode” and “msgxchg”
  These modules are important for handling the dual core data exchange, the “msaxchg” module provides a channel configure structure named “MSGXCHG_ChannelConfig_T”, the developer should depend on the application to allocate channel number and TX/RX buffer size,

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

For example, in this reference design, there are three channels will be allocated as below.

- BufOffset field: Indicates the memory offset in Share Memory (.RAM_D3).
- Buf1Size field: Indicates the buffer size in bytes for CM4 to CM7 transmit direction.
- Buf2Size field: Indicates the buffer size in bytes for CM7 to CM4 transmit direction.
- Ctrl field: Specifies that whether or not to enable data integrity check.

Channel Index	Channel Name	BufOffset	Buf1Size	Buf2Size	Ctrl
0	MSGXCHG_ADIO_PD_CH	0x0000	0x0080	0x0080	0
1	MSGXCHG_ADIO_CMD_CH	0x0100	0x0080	0x0080	MSGXCHG_CTL_CHKSUM
2	MSGXCHG_ADIO_CFG_CH	0x0200	0x0080	0x0080	MSGXCHG_CTL_CHKSUM

Table 14-1

- Channel 0: This channel is used to exchange process data between dual cores and without response need from receiver. In this reference design, if there are control data changes in CM4 core, the whole control data will be transmitted to CM7 core, similarly, if there are status data changes in CM7 core, the status data will be transmitted to CM4 core.

- Channel 1: This channel is used to transmit command data from CM4 to CM7 core and with response need from receiver. The CM4 to CM7 buffer address is 0x38000100 with 0x80 byte size and CM7 to CM4 buffer address is 0x38000180, also with 0x80 byte size.

- Channel 2: This channel is used to transmit configure data from CM4 to CM7 core and with response need from receiver. Similarly, the CM4 to Cm7 buffer address is 0x38000200, CM7 to CM4 buffer address is 0x38000280, and two buffers have the same size as 0x80 bytes.

Based on these channels, the “cmdcode” module is responsible for providing upper-layer API functions for application calling. In this reference design, “cmdcode” provided the API functions and dependent channel index are as below.

CM4 to CM7 API Function	CM7 to CM4 API Function	Used Channel Name
CC_AdioControlChangeReq() cc_AdioStatusChangeRsp()	cc_AdioControlChangeRsp() CC_AdioStatusChangeReq()	MSGXCHG_ADIO_PD_CH
CC_GetSwitchID()	cc_AdioGetSwitchIDRsp()	MSGXCHG_ADIO_CMD_CH
None	None	MSGXCHG_ADIO_CFG_CH

Table 14-2

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

By the way, all the allocated channel buffers are mapped to the Share Memory (.RAM_D3), which be declared in the files as below, the developer can adjust the Share Memory size properly according to the application.
For Keil_MDK IDE:
BSP_ROOT\SampleCode\Ax58400_GpioAio\MDK-ARM\stm32h755xx_flash_CM4.sct
BSP_ROOT\SampleCode\Ax58400_GpioAio\MDK-ARM\stm32h755xx_flash_CM7.sct

For STM32CubeMX IDE:
BSP_ROOT\SampleCode\Ax58400_GpioAio\STM32CubeIDE\CM4\ STM32H755ZITX_FLASH.sct
BSP_ROOT\SampleCode\Ax58400_GpioAio\STM32CubeIDE\CM7\ STM32H755ZITX_FLASH.sct

14-2 CM7 Core Modules

The CM7 core is responsible for executing ADIO Manager Code and receiving process/control/command data from CM4 core. The related modules are listed in below.

- ADIO Manager Code
  This code is responsible for accessing the input/output data on analog/digital pins, then exchange the data with CM4 core.

14-3 CM4 Core Modules

The CM4 core is mainly executing SSC stack for handling EtherCAT communication and CiA401 device profiler. It includes the modules as below.

- EtherCAT Slave Stack Code (SSC)
  This stack is generated by ETG's SSC tool.

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

Evaluation Board License Agreement

By using this evaluation board or kit (together with all associated software, hardware and documentation provided by ASIX, collectively, the "Evaluation Board"), you ("You") agree to be bound by the terms and conditions of this Evaluation Board License Agreement ("Agreement "). Do not use the evaluation board until you have read and agreed to this agreement. Your use of the Evaluation Board constitutes your acceptance of this Agreement.

License:
ASIX Electronics (“ASIX”) grants you the right to use the accompanying evaluation board, which provides limited functionality, for the sole purpose of evaluating and testing ASIX products for your evaluation and testing purposes in a research and development environment. Under no circumstances should the evaluation board be assembled, directly or indirectly, as part of any of your products, as it has been developed for evaluation purposes only and has no direct functionality and is not a finished product.

Notice:
(1) The description of hardware circuit, software tools and other related information in this document are only used to illustrate the operation of application examples for ASIX Electronic products. You are solely responsible for the use of such circuits, software and information in the design of your equipment. ASIX Electronics is not responsible for any losses arising out of the use of the circuit, software or information by you or a third party.
(2) ASIX Electronics has taken reasonable care in describing the information in this document, but ASIX Electronics does not warrant that such information is error-free. ASIX Electronics disclaims all liability for any damages you may suffer as a result of errors or omissions in the information contained herein.
(3) ASIX Electronics shall not be liable for infringement of patents, copyrights or other intellectual property rights of third parties due to the use of ASIX Electronics products or the technical information described in this document. No license, express or implied, or otherwise, is granted under any patent, copyright or other intellectual property rights of ASIX Electronics or others.
(4) You may not alter, modify, copy or otherwise misappropriate any ASIX electronic product, whether in whole or in part. ASIX Electronics shall not be liable for any loss incurred by you or a third party as a result of altering, modifying, copying or otherwise misappropriating ASIX Electronics products.
(5) You should use the ASIX Electronic products described in this document within the range specified by ASIX Electronic, especially the maximum rating, operating power supply voltage range, mobile power supply voltage range, heat radiation characteristics, installation and other product characteristics. ASIX Electronics accepts no liability for malfunctions or damages resulting from use of ASIX Electronics products outside of such designations.
(6) When using ASIX electronic products, please comply with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including but not limited to the EU RoHS Directive. ASIX Electronics disclaims all liability for damage or loss resulting from your failure to comply with applicable laws and regulations.
(7) The buyer or distributor of ASIX electronic products is responsible for distributing, disposing of or otherwise placing the product to a third party, and notifying the third party of the contents and conditions specified in this document in advance, ASIX Electronics shall not be liable for any loss suffered as a result of authorized use of ASIX Electronics products.
(8) This document may not be reproduced or reproduced in any form, in whole or in part, without the prior written consent of ASIX Electronics.

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

Copyright © 2021-2024 ASIX Electronics Corporation. All rights reserved.

DISCLAIMER

No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying and recording, for any purpose, without the express written permission of ASIX. ASIX may make changes to the product specifications and descriptions in this document at any time, without notice.

ASIX provides this document “as is” without warranty of any kind, either expressed or implied, including without limitation warranties of merchantability, fitness for a particular purpose, and non-infringement.

Designers must not rely on the absence or characteristics of any features or registers marked “reserved”, “undefined” or “NC”. ASIX reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. Always contact ASIX to get the latest document before starting a design of ASIX products.

TRADEMARKS

ASIX, the ASIX logo are registered trademarks of ASIX Electronics Corporation. All other trademarks are the property of their respective owners.

EtherCAT® is a registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany.

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation

ASIX Electronics Corporation

4F, No.8, Hsin Ann RD., Hsinchu Science Park,
Hsinchu, Taiwan, R.O.C.

TEL: +886-3-5799500
FAX: +886-3-5799558

Email: support@asix.com.tw
Web: https://www.asix.com.tw

Copyright (C) 2021-2024 Reserved by ASIX Electronics Corporation