



AX58100 Motor Control User Guide

Revision 1.00
January 20th, 2025

Revision History

Revision	Date	Description
1.00	2025/01/20	Initial release

CONTENT

1. Introduction	7
2. Requirements	10
3. Board Description	15
3-1 X-NUCLEO-IHM08M1	15
4. Environment Setup.....	17
4-1 Hardware Connectivity	17
5. Building the Reference Design Begin with MotorControl Workbench	18
5-1 Create New Project and Generate Source Code in MC Workbench	18
5-2 Modify Peripheral Settings in STM32CubeMX	24
5-3 Generate Source Code in STM32CubeMX	29
5-4 Source Code Adjustment	30
5-5 Generate EtherCAT Slave Stack Code	33
5-6 IDE Setting Adjustment	34
6. Build Up and Download Firmware	39
6-1 Build up Firmware Binary	39
6-2 Download Bootloader Firmware	40
6-3 Download Application Firmware	40
7. Basic Operation of TwinCAT	42
8. PLC Application.....	42
9. Flash Memory Allocation and FoE Upgrading	43
9-1 FoE Firmware Upgrade	44
10. Object Dictionary	45
10-1 Objects Description	45
11. Performance Evaluation	48
11-1 Minimum DC Cycle Time.....	48
12. Tuning the PID Loop Gains with TwinCAT	50
12-1 Add New Objects into Dictionary	50
12-2 Link New Objects to Data Structure	50
12-3 Add Handling Functions According to New Object Indexes	51
12-4 Create Scope Project in TwinCAT.....	52
12-5 Position Loop Gain Value Manual Tuning in TwinCAT	55
12-6 Speed Loop Gain Value Manual Tuning in TwinCAT	56

List of Figure

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

Figure 5-40	36
Figure 5-41	37
Figure 5-42	37
Figure 5-43	37
Figure 5-44	38
Figure 6-1	39
Figure 6-2	39
Figure 6-3	40
Figure 6-4	40
Figure 6-5	40
Figure 6-6	41
Figure 9-1	43
Figure 9-2	43
Figure 9-3	44
Figure 9-4	44
Figure 9-5	44
Figure 11-1	48
Figure 11-2	48
Figure 11-3	49
Figure 12-1	50
Figure 12-2	50
Figure 12-3	51
Figure 12-4	51
Figure 12-5	51
Figure 12-6	52
Figure 12-7	52
Figure 12-8	53
Figure 12-9	53
Figure 12-10	53
Figure 12-11	54
Figure 12-12	54
Figure 12-13	55
Figure 12-14	55
Figure 12-15	56

List of Table

Table 2-1.....	13
Table 2-2.....	13
Table 2-3.....	13
Table 2-4.....	14
Table 3-1.....	16
Table 4-1.....	17
Table 10-1.....	45
Table 10-2.....	46
Table 10-3.....	46
Table 10-4.....	47

1. Introduction

EtherCAT (Ethernet for Control Automation Technology) is an Ethernet-based fieldbus system; it is suitable for both hard and soft real-time computing requirements in automation applications.

The two-axes position control system is developed to illustrate the EtherCAT performance and target on motion control application, it integrates slave side FOC (Field-Oriented Control) control algorithm, CANopen over EtherCAT protocol and master side PLC (Programmable Logic Controller) program. This document will introduce how to build-up entire hardware, software environment and how to manipulate the PLC-based dashboard for demonstration. As illustrated in below figure, the EtherCAT master side is a PC/IPC and consist of real-time capable NIC, TwinCAT master and PLC demo program.

- **Real-Time Capable NIC and Driver**

This is mandatory equipment in order to guarantee short data update time (also called cycle time) and precise synchronization (jitter < 1us).

- **TwinCAT Master**

This is run on a real-time sub-system which created by TwinCAT developed environment on Windows OS, it is in charge of the EtherCAT network topology build-up, configuration and cyclic process data transfer.

- **PLC Demo Program**

This is a pre-developed procedure based on TwinCAT engineering (XAE) environment, then run on TwinCAT Master, it provided a visible dashboard for user manipulation.

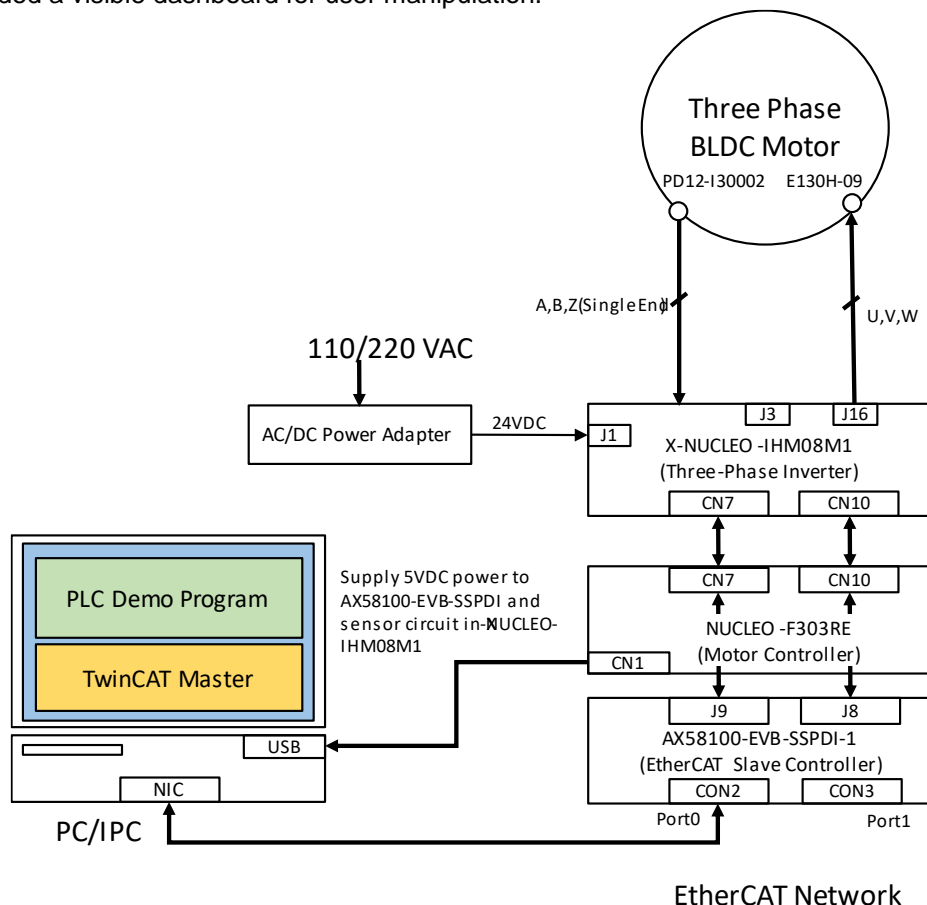


Figure 1-1

EtherCAT slave side is consist of two identical motion axes, each axis contains EtherCAT Slave Controller, Motion Controller, Three-Phase Inverter and Three-Phase BLDC Motor.

- **EtherCAT Slave Controller**

It is responsible of EtherCAT communication, such as EtherCAT datagram encapsulation/de-capsulation, EtherCAT commands performing and distributed clock mechanism build-up. The AX58100 is an EtherCAT slave controller ASIC and used to handling these matters as mention above.

- **Motor Controller**

It is a 32-bits embedded system and in charge of handling to EtherCAT state machine, CiA402 device profile protocol and FOC motion control loops, here we use STM32F303RE to do this.

- **Three-Phase Inverter**

It is a half-bridge power inverter and used to driving three-phase brushless DC motor, here we use ST X-NUCLEO-IHM08M1 expansion board to do this.

- **Three-Phase BLDC**

Here we use MTM's BL60M24D8E00430080 motor.

The furthermore software/hardware structure diagram for EtherCAT Master/Slave side has depicted as below.

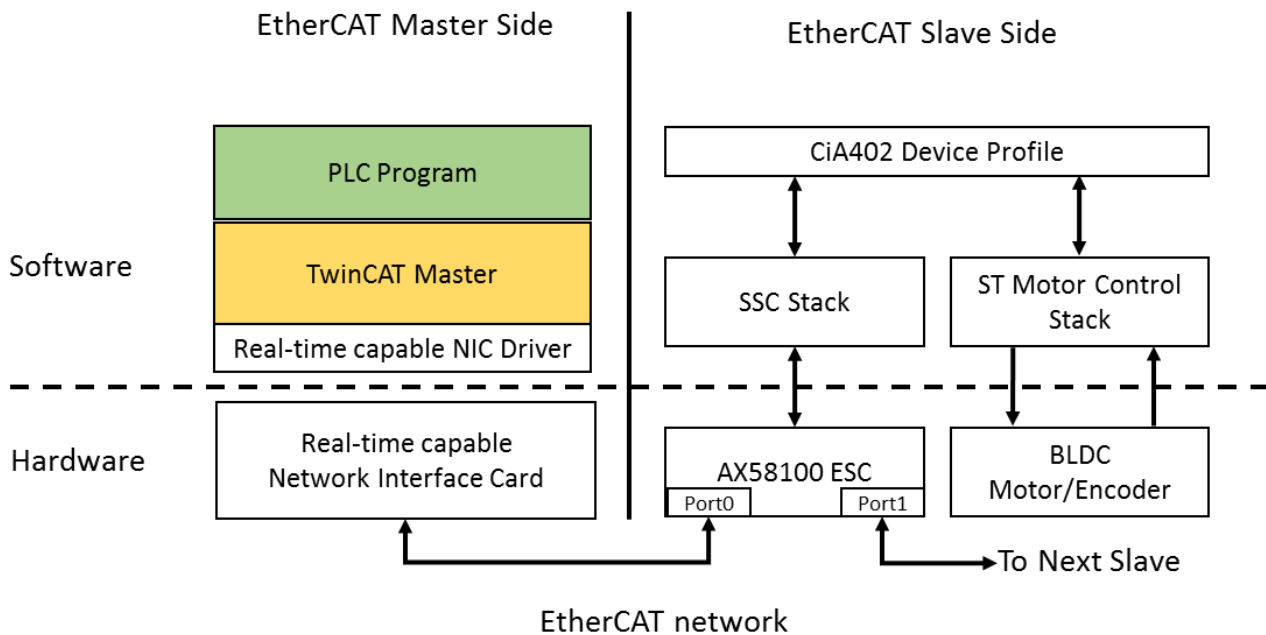


Figure 1-2

- **SSC Stack**

This stack is used to handle ESM (EtherCAT State Machine) and frame communication, which is created by SSC tool and free download for ETG members only.

● **ST Motor Control Stack**

This software stack is provided in STM32 motor control SDK, the position and FOC control loops are illustrated as below.

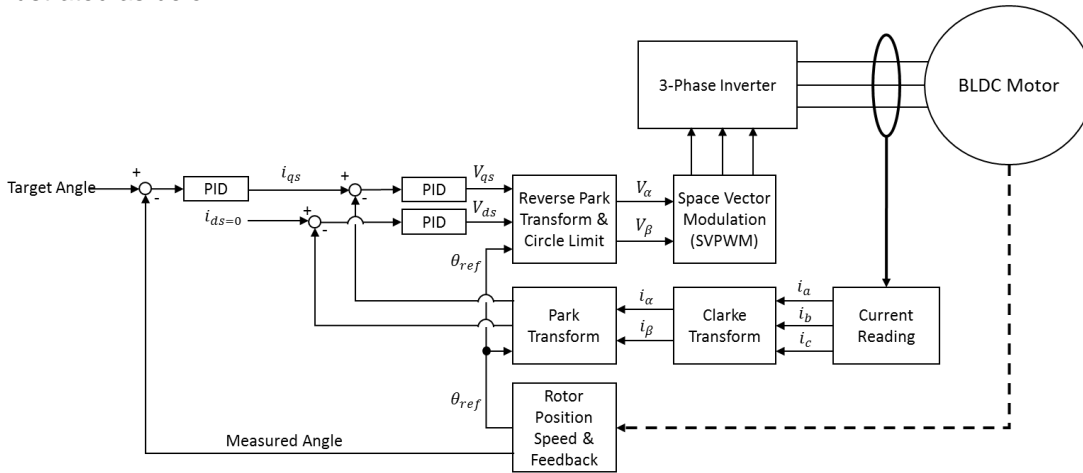


Figure 1-3

2. Requirements

Before starting to setup the environment, we should prepare some software and hardware. Of course, they are all running based on general desktop PC with windows 10 / 11 operation systems. We list software / hardware requirements 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]

- ST MotorControl Workbench6 version 6.2.1 (for development only)
You can download the MotorControl Workbench [here.](#)



Figure 2-2

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



Figure 2-3

- 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)



Figure 2-4

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



Figure 2-5

- STM32CubeProgrammer
You can download the STM32CubeProgrammer [here](#).



Figure 2-6

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



Figure 2-7

[Hardware]

- ST NUCLEO-F303RE Control Board x 1 PCS

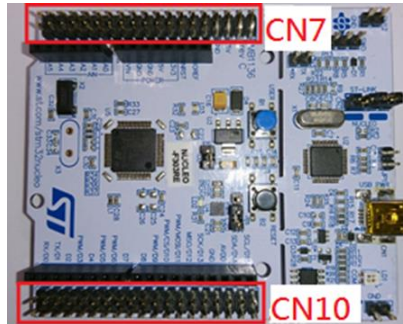


Figure 2-8

- AX58100-EVB-SSPDI-1 V1.0 EVB x 1 PCS
For more details, please refer to the link below and the following chapter.
ASIX official link: <https://www.asix.com.tw/>

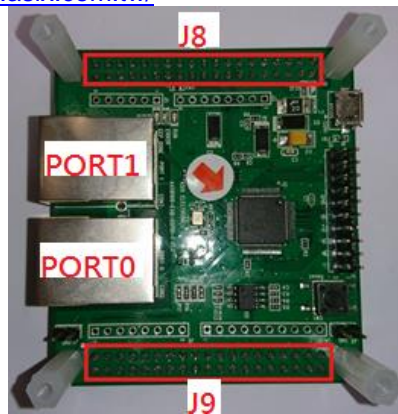


Figure 2-9

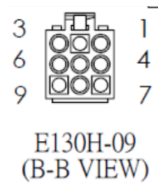
- X-NUCLEO-IHM08M1 three-phase brushless DC motor driver expansion board x 1
For more details, please refer to the link below and the following chapter.
STMicroelectronics official link: <https://www.st.com/en/ecosystems/x-nucleo-ihm08m1.html>
- MOTION TECH BL60M24D8E00430080 Motor x 1
For more details, please refer to the link below.
MTM official link: <http://www.motiontech.com.tw/>



Figure 2-10

BL60M24D8E00430080 Performance Spec.	
Rated Voltage	24VDC
Direction of Rotation	C.W.
Rated Power	80W
Rated Torque	2.6 Kg-cm
Rated Input Current	4.0±5%A
Rated Speed	3000±5%RPM
L-L Resistance	486±5mΩ
Hi-Pot	1000VAC/1mA/1sec.
Insulation Test	500VDC/50MΩ min.

Table 2-1



E130H-09 Connector Wire Definition		
PIN	Definition	Wire color
1	U	Blue
2	V	Purple
3	W	Gray
4	Hall Vcc	Yellow
5	Hall Gnd	Green
6	Null	X
7	S1	Brown
8	S2	Red
9	S3	Orange

Table 2-2



PD12-I30002 Connector Wire Definition		
PIN	Definition	Wire color
1	Null	X
2	Null	X
3	Z+	Yellow
4	B+	Red
5	A+	Black
6	GND	White
7	Null	X
8	Null	X
9	Z-	Brown
10	B-	Orange
11	A-	Green
12	+5V	Blue

Table 2-3

Motor		Sensors	
Magnetic structure		Surface Mounted PMSM	
Electrical parameters			
Pole Pairs	4		
Max. Application Speed	3000	rpm	
Nominal Current	3.50	Apk	
Nominal DC Voltage	23.9	V	
Rs	0.20	Ohm	
LS	0.114	mH	
B-Emf constant	3.9	Vrms/krpm	
Inertia	6.103	$\mu\text{N}^*\text{m}^*\text{s}^2$	
Friction	6.325	$\mu\text{N}^*\text{m}^*\text{s}$	

Table 2-4

- 24VDC/8A Power Supply x 1



Figure 2-11

3. Board Description

3-1 X-NUCLEO-IHM08M1

The X-NUCLEO-IHM08M1 is a three-phase brushless DC motor driver expansion board.

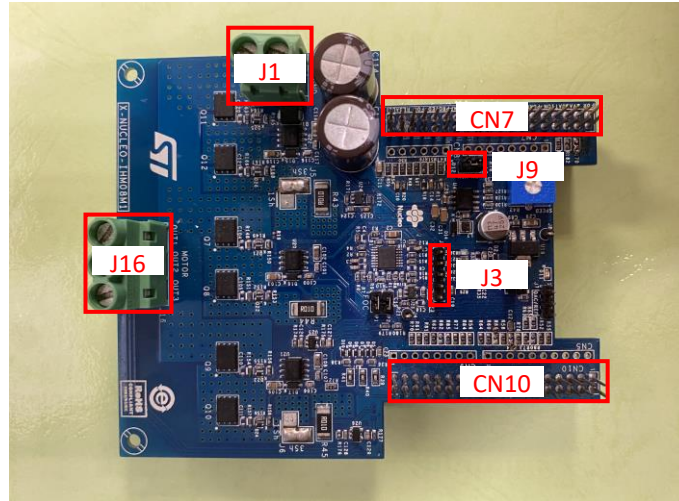


Figure 3-1

- J16: U/V/W three-phase output connector.
- J1: 24V DC link power source connector.
- CN7/CN10: Standard connectors, mounting with J10/J13 on AX58100-EVB-SSPDI board.
- J9: External power selector.
- J3: Incremental encoder interface.

Attention! Before starting to build-up the system, please make sure the X-NUCLEO-IHM08M1 board is ready by check these jumps and components status as below table.

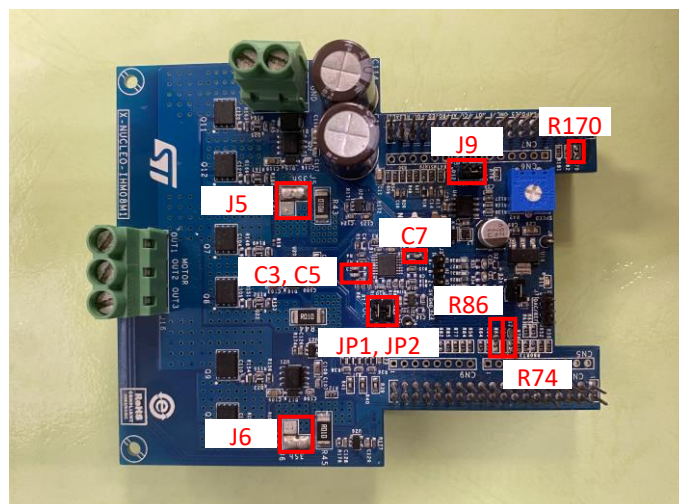


Figure 3-2

X-NUCLEO-IHM08M1 Components	Status
J9	Opened
JP1, JP2,	Closed
J5, J6	Short 3sh side
R74, R86, R170, C3, C5, C7	Unmounted

Table 3-1

4. Environment Setup

4-1 Hardware Connectivity

Please based on below table to connect all necessary equipment.

AX58100 Motor Control Connectivity Table						
Connectivity Item	24VDC	PC/IPC	Axis			
			X-NUCLEO-IHM08M1	NUCLEO-F303RE	AX58100-EVB-SSPDI	MTM Motor
EtherCAT network		Ethernet port			CON2(port0) CON3(port1)	
ESC control bus				CN7 ↔ J9 CN10 ↔ J8		
Three phase inverter control bus			CN7 ↔ CN7 CN10 ↔ CN10			
Three phase inverter power	+24V/GND		J1			
Motor U phase			J16.2 (OUT2)			E130H-09.1 (Blue)
Motor V phase			J16.3 (OUT1)			E130H-09.2 (Purple)
Motor W phase			J16.1 (OUT3)			E130H-09.3 (Gray)
Encoder A				CN7.17		SingleEnd A
Encoder B				CN10.31		SingleEnd B
Encoder Z				CN10.25		SingleEnd Z
Encoder +5V				CN7.18		PD12-I30002.12
Encoder GND				CN7.20		PD12-I30002.6
Power USB		USB port		CN1		

Table 4-1

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

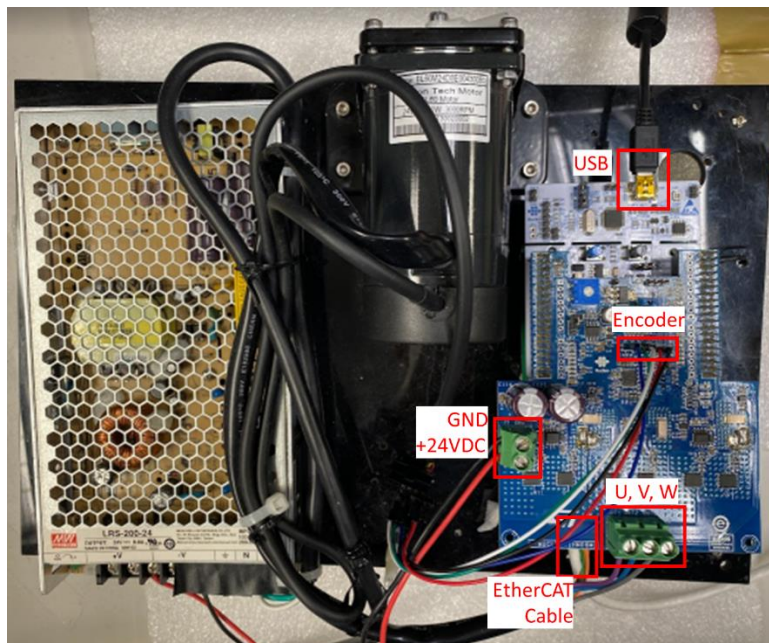


Figure 4-1

5. Building the Reference Design Begin with MotorControl Workbench

Here will introduce the procedure about build up the reference design source package. Please follow the steps below to do this.

5-1 Create New Project and Generate Source Code in MC Workbench

Step 1: Open the MC Workbench and create a new project.

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

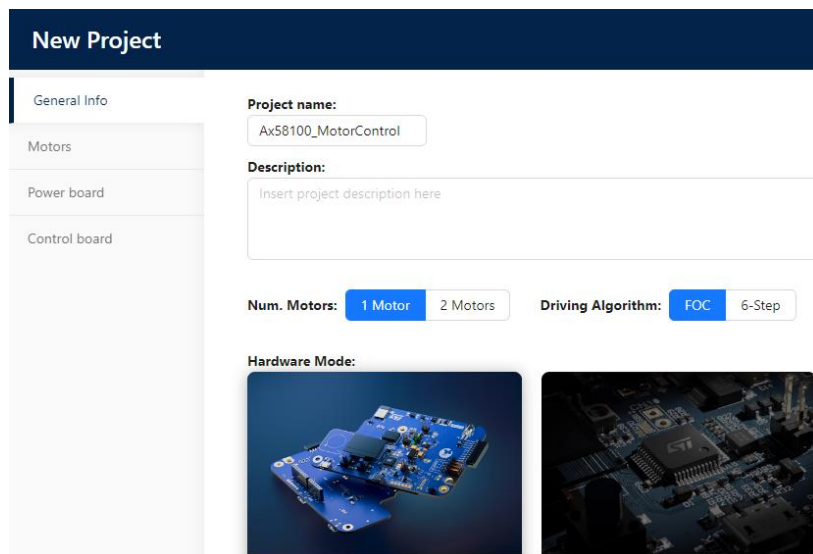


Figure 5-1

Step 2: Select SM-PMSM 24V motor, X-NUCLEO-IHM08M1 power board and NUCLEO-F303RE control board.

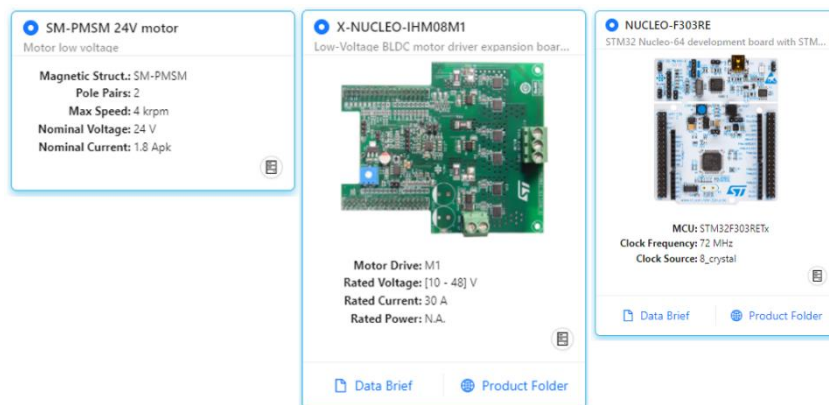
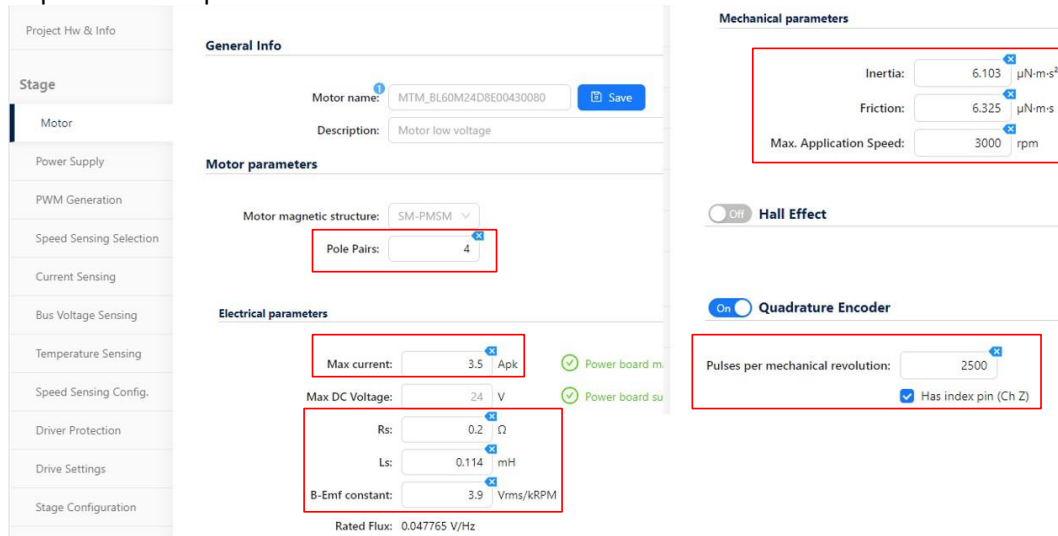


Figure 5-2

Step 3: Setup "Motor" Properties as below



General Info

Motor name: MTM_BL60M24D8E00430080 Save

Description: Motor low voltage

Motor parameters

Motor magnetic structure: SM-PMSM

Pole Pairs: 4

Electrical parameters

Max current: 3.5 Apk Power board m

Max DC Voltage: 24 V Power board su

Rs: 0.2 Ω

Ls: 0.114 mH

B-Emf constant: 3.9 Vrms/kRPM

Rated Flux: 0.047765 V/Hz

Mechanical parameters

Inertia: 6.103 $\mu\text{N}\cdot\text{m}\cdot\text{s}^2$

Friction: 6.325 $\mu\text{N}\cdot\text{m}\cdot\text{s}$

Max. Application Speed: 3000 rpm

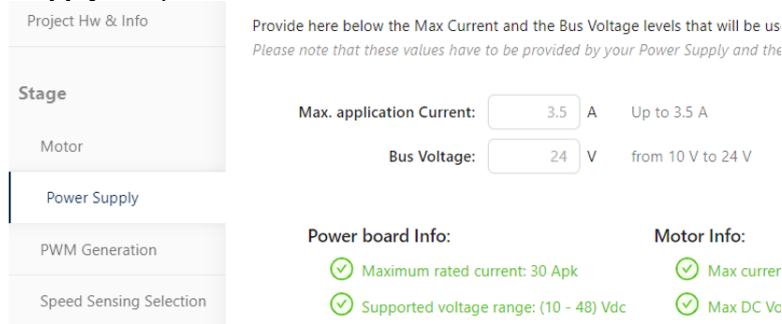
Hall Effect

Quadrature Encoder

Pulses per mechanical revolution: 2500 Has index pin (Ch Z)

Figure 5-3

Step 4: Setup "Power Supply" Properties as below



Provide here below the Max Current and the Bus Voltage levels that will be us
Please note that these values have to be provided by your Power Supply and the

Max. application Current: 3.5 A Up to 3.5 A

Bus Voltage: 24 V from 10 V to 24 V

Power board Info:

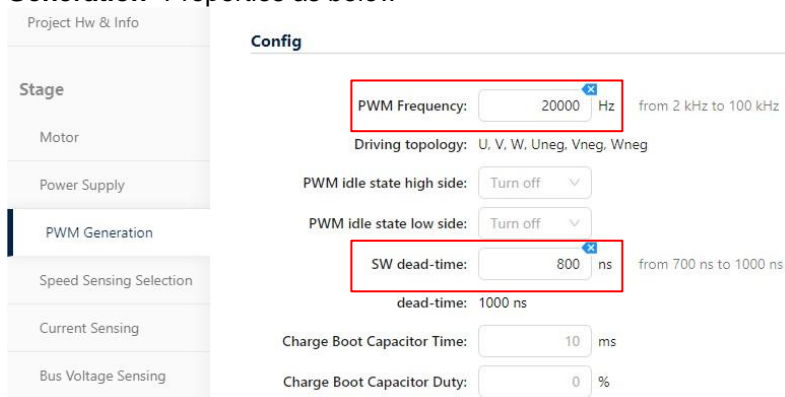
- Maximum rated current: 30 Apk
- Supported voltage range: (10 - 48) Vdc

Motor Info:

- Max curren
- Max DC Vc

Figure 5-4

Step 5: Setup "PWM Generation" Properties as below



Config

PWM Frequency: 20000 Hz from 2 kHz to 100 kHz

Driving topology: U, V, W, Uneg, Vneg, Wneg

PWM idle state high side: Turn off

PWM idle state low side: Turn off

SW dead-time: 800 ns from 700 ns to 1000 ns

dead-time: 1000 ns

Charge Boot Capacitor Time: 10 ms

Charge Boot Capacitor Duty: 0 %

Figure 5-5

Step 6: Setup "Speed Sensing Selection" Properties as below

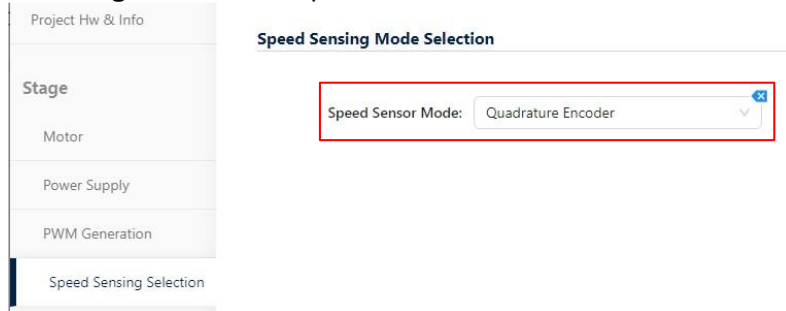


Figure 5-6

Step 7: Setup "Current Sensing" Properties as below

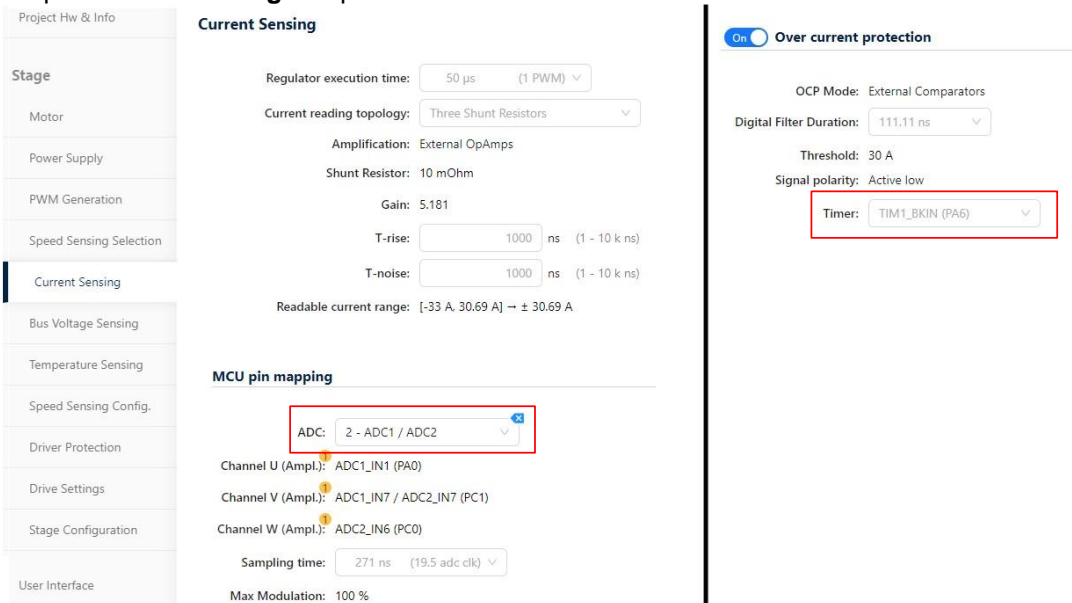


Figure 5-7

Step 8: Setup "Bus Voltage Sensing" Properties as below

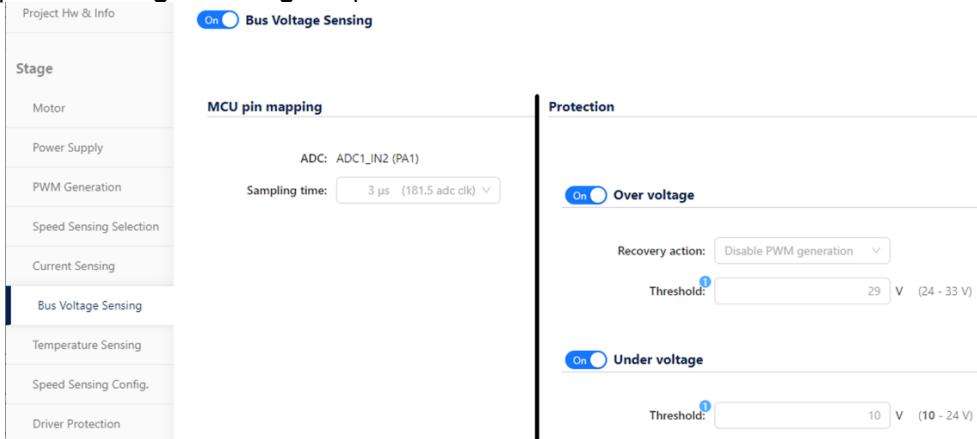
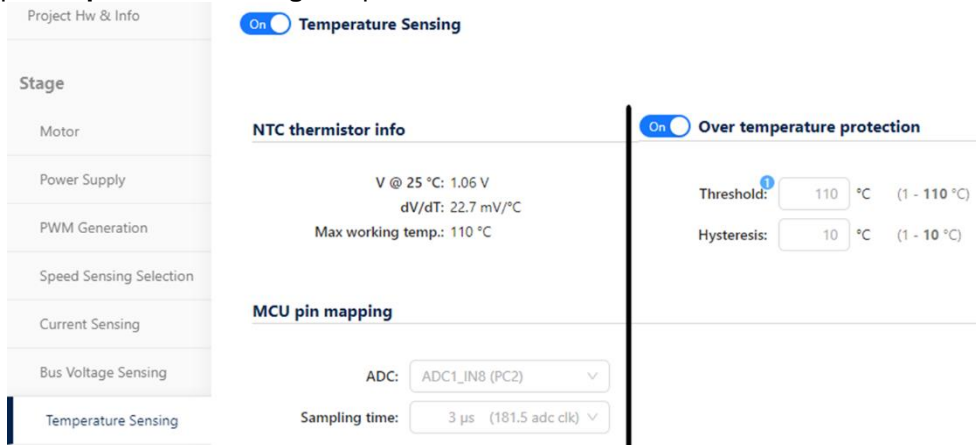


Figure 5-8

Step 9: Setup "Temperature Sensing" Properties as below



Project Hw & Info

Temperature Sensing

Stage

Motor

Power Supply

PWM Generation

Speed Sensing Selection

Current Sensing

Bus Voltage Sensing

Temperature Sensing

NTC thermistor info

V @ 25 °C: 1.06 V
dV/dT: 22.7 mV/°C
Max working temp.: 110 °C

MCU pin mapping

ADC:

Sampling time:

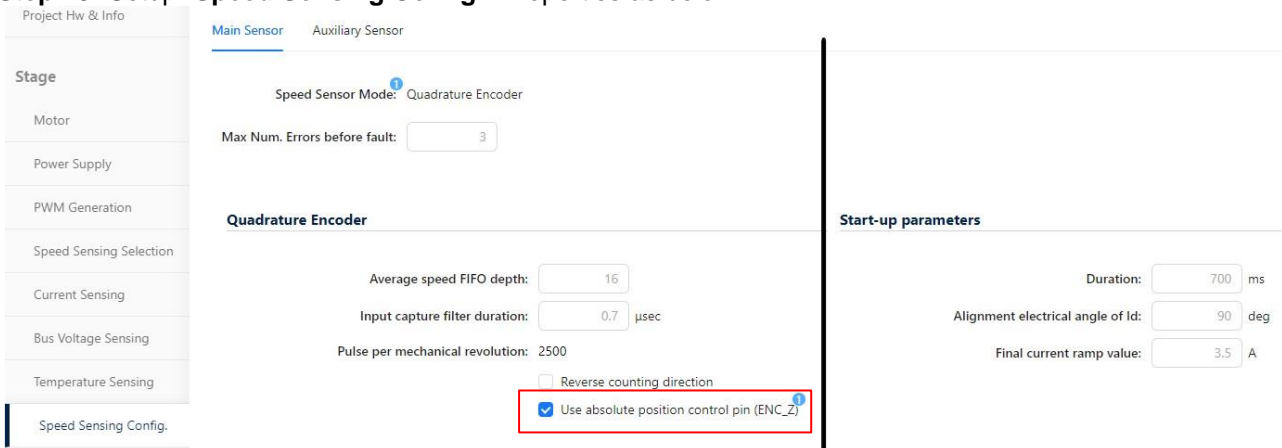
Over temperature protection

Threshold: °C (1 - 110 °C)

Hysteresis: °C (1 - 10 °C)

Figure 5-9

Step 10: Setup "Speed Sensing Config." Properties as below



Project Hw & Info

Main Sensor Auxiliary Sensor

Speed Sensor Mode:

Max Num. Errors before fault:

Quadrature Encoder

Average speed FIFO depth:

Input capture filter duration: μsec

Pulse per mechanical revolution: 2500

Reverse counting direction

Use absolute position control pin (ENC_Z)

Start-up parameters

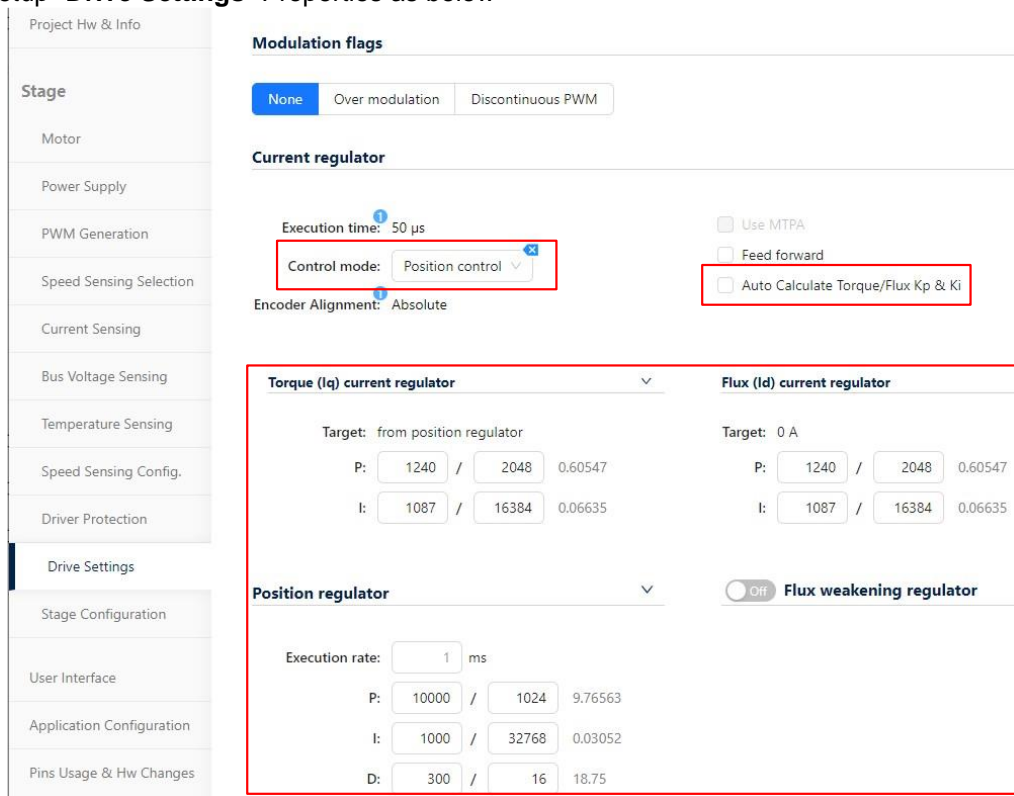
Duration: ms

Alignment electrical angle of Id: deg

Final current ramp value: A

Figure 5-10

Step 11: Setup "Drive Settings" Properties as below



Modulation flags

None Over modulation Discontinuous PWM

Current regulator

Execution time: 50 µs

Control mode: Position control

Encoder Alignment: Absolute

Use MTPA

Feed forward

Auto Calculate Torque/Flux Kp & Ki

Torque (Iq) current regulator

Target: from position regulator

P: 1240 / 2048 0.60547

I: 1087 / 16384 0.06635

Flux (Id) current regulator

Target: 0 A

P: 1240 / 2048 0.60547

I: 1087 / 16384 0.06635

Position regulator

Execution rate: 1 ms

P: 10000 / 1024 9.76563

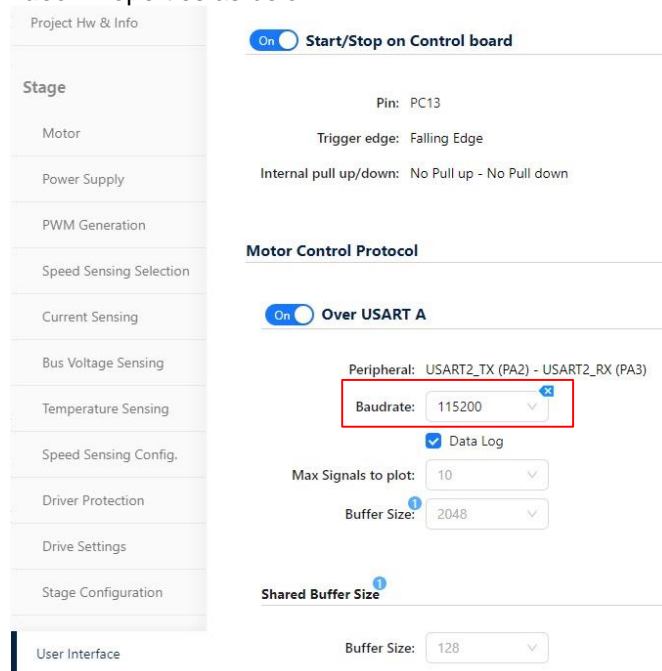
I: 1000 / 32768 0.03052

D: 300 / 16 18.75

Flux weakening regulator Off

Figure 5-11

Step 12: Setup "User Interface" Properties as below



Project Hw & Info

Stage

Motor

Power Supply

PWM Generation

Speed Sensing Selection

Current Sensing

Bus Voltage Sensing

Temperature Sensing

Speed Sensing Config.

Driver Protection

Drive Settings

Stage Configuration

User Interface

Application Configuration

Pins Usage & Hw Changes

Start/Stop on Control board

Pin: PC13

Trigger edge: Falling Edge

Internal pull up/down: No Pull up - No Pull down

Motor Control Protocol

Over USART A

Peripheral: USART2_TX (PA2) - USART2_RX (PA3)

Baudrate: 115200

Data Log

Max Signals to plot: 10

Buffer Size: 2048

Shared Buffer Size

Buffer Size: 128

Figure 5-12

So far, the settings on MC Workbench have done, about the detail please refer to the file as below.
BSP_ROOT\SampleCode\Ax58100_MotorControl\Ax58100_MotorControl.stwb6

Step 13: Generates the source code with the CubeMX v6.12.1, Firmware Package Version v1.11.1.

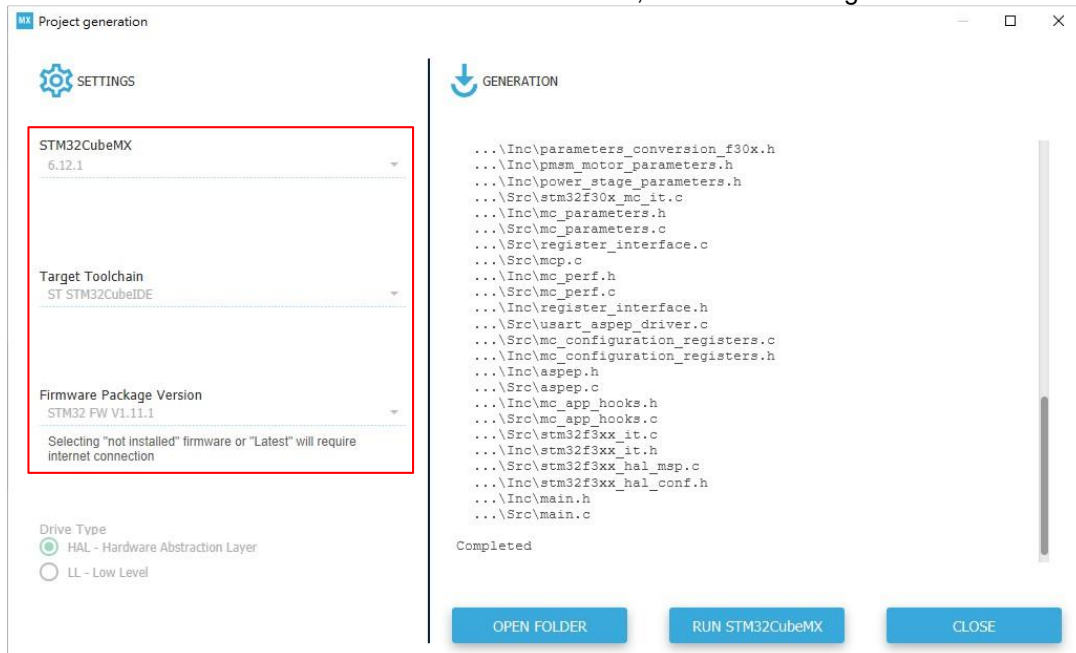


Figure 5-13

5-2 Modify Peripheral Settings in STM32CubeMX

Step 1: Disable the “Pinout & Configuration > System Core > RCC > High Speed Clock (HSE)” setting as the figure below.

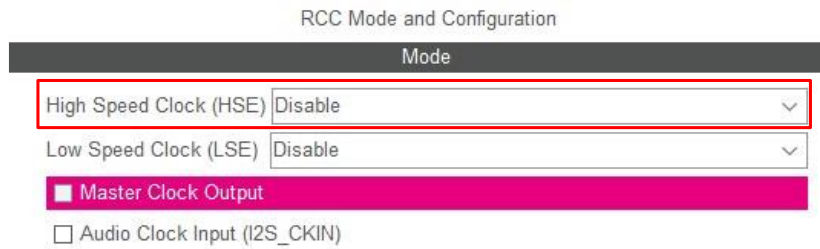
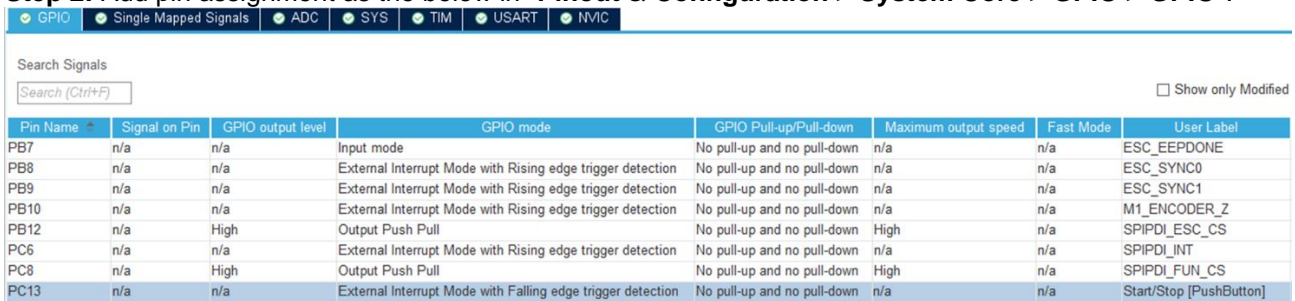


Figure 5-14

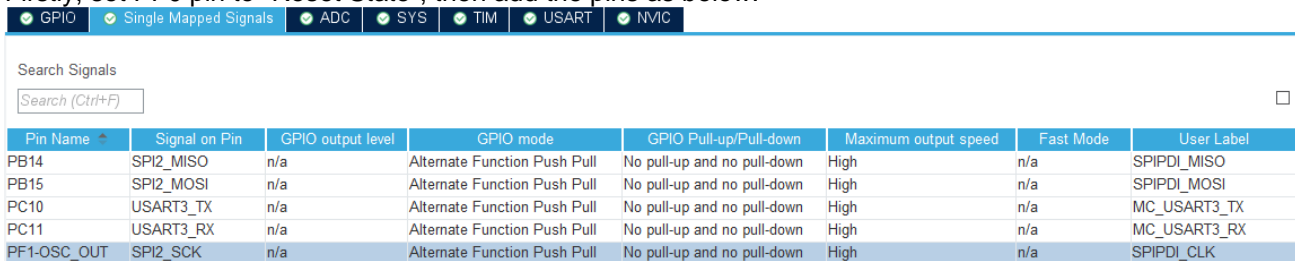
Step 2: Add pin assignment as the below in “Pinout & Configuration > System Core > GPIO > GPIO”.



Pin Name	Signal on Pin	GPIO output level	GPIO mode	GPIO Pull-up/Pull-down	Maximum output speed	Fast Mode	User Label
PB7	n/a	n/a	Input mode	No pull-up and no pull-down	n/a	n/a	ESC_EEPDONE
PB8	n/a	n/a	External Interrupt Mode with Rising edge trigger detection	No pull-up and no pull-down	n/a	n/a	ESC_SYNC0
PB9	n/a	n/a	External Interrupt Mode with Rising edge trigger detection	No pull-up and no pull-down	n/a	n/a	ESC_SYNC1
PB10	n/a	n/a	External Interrupt Mode with Rising edge trigger detection	No pull-up and no pull-down	n/a	n/a	M1_ENCODER_Z
PB12	n/a	High	Output Push Pull	No pull-up and no pull-down	High	n/a	SPIPDI_ESC_CS
PC6	n/a	n/a	External Interrupt Mode with Rising edge trigger detection	No pull-up and no pull-down	n/a	n/a	SPIPDI_INT
PC8	n/a	High	Output Push Pull	No pull-up and no pull-down	High	n/a	SPIPDI_FUN_CS
PC13	n/a	n/a	External Interrupt Mode with Falling edge trigger detection	No pull-up and no pull-down	n/a	n/a	Start/Stop [PushButton]

Figure 5-15

Step 3: Add pin assignment for SPI2 and USART3. Firstly, set PF0 pin to “Reset State”, then add the pins as below.



Pin Name	Signal on Pin	GPIO output level	GPIO mode	GPIO Pull-up/Pull-down	Maximum output speed	Fast Mode	User Label
PB14	SPI2_MISO	n/a	Alternate Function Push Pull	No pull-up and no pull-down	High	n/a	SPIPDI_MISO
PB15	SPI2_MOSI	n/a	Alternate Function Push Pull	No pull-up and no pull-down	High	n/a	SPIPDI_MOSI
PC10	USART3_TX	n/a	Alternate Function Push Pull	No pull-up and no pull-down	High	n/a	MC_USART3_TX
PC11	USART3_RX	n/a	Alternate Function Push Pull	No pull-up and no pull-down	High	n/a	MC_USART3_RX
PF1-OSC_OUT	SPI2_SCK	n/a	Alternate Function Push Pull	No pull-up and no pull-down	High	n/a	SPIPDI_CLK

Figure 5-16

Step 4: Activated TIM6 & TIM7.

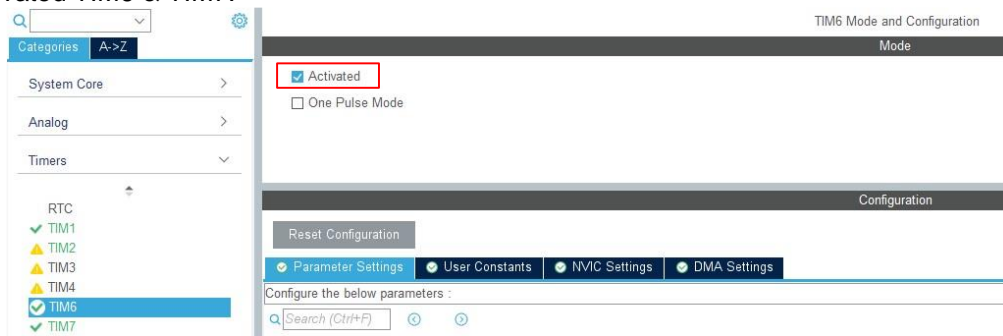


Figure 5-17

Step 5: Enable SPI2 with the setting as highlighted in red.

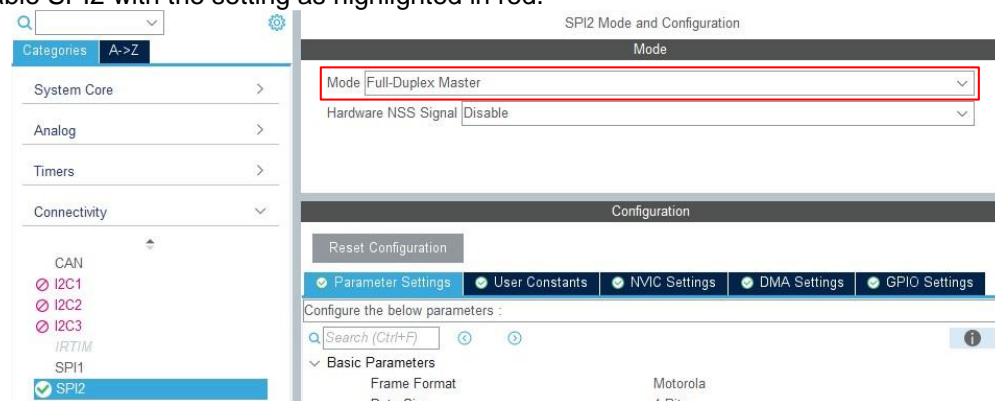


Figure 5-18

Step 6: Disable DMA function on USART2 for debug console.

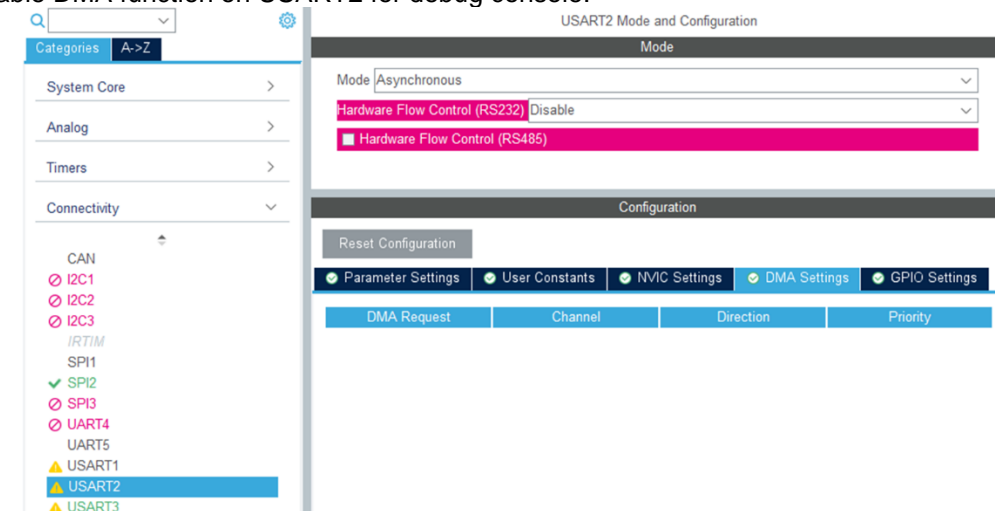
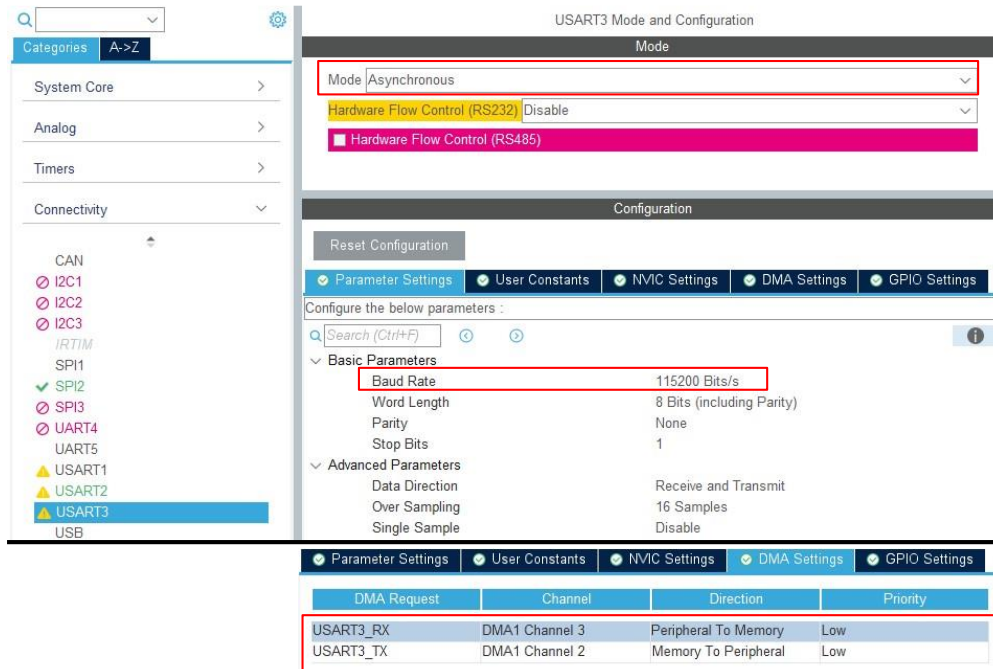


Figure 5-19

Step 7: Enable USART3 with DMA for MCP.



DMA Request	Channel	Direction	Priority
USART3_RX	DMA1 Channel 3	Peripheral To Memory	Low
USART3_TX	DMA1 Channel 2	Memory To Peripheral	Low

Figure 5-20

Step 8: Modify MCP settings in “Pinout & Configuration > Middleware and Software Packs > MotorControl” as below.

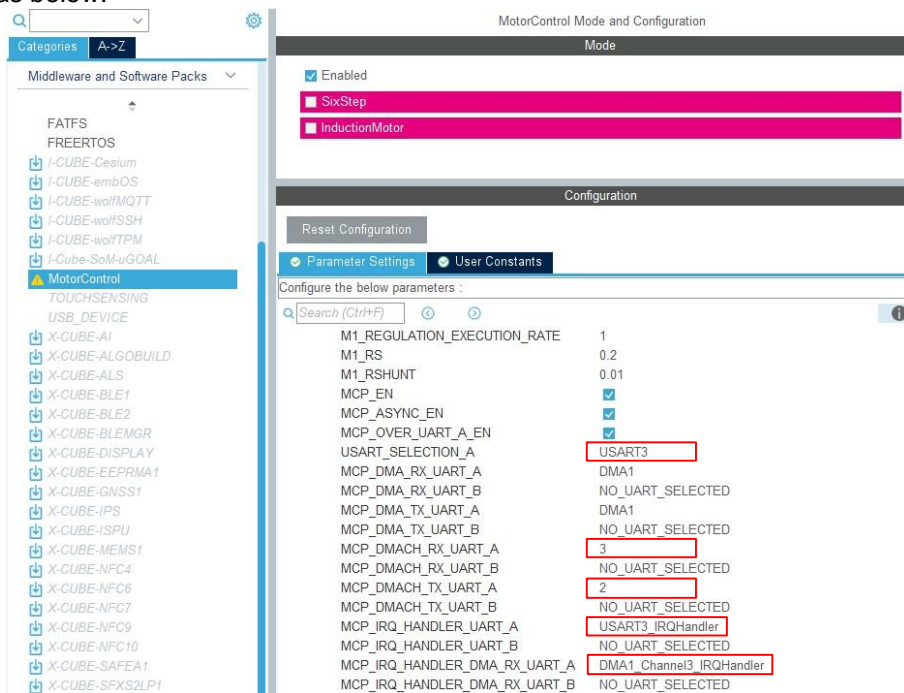


Figure 5-21

Step 9: Modify the NVIC setting as below.

NVIC Mode and Configuration

Configuration			
<input checked="" type="radio"/> NVIC <input checked="" type="radio"/> Code generation			
Priority Group	3 bits for pre-emption priority 1 bi...	<input type="checkbox"/> Sort by Preemption Priority and Sub Priority	<input type="checkbox"/> Sort by interrupts names
Search	Search (Ctrl+F)	Show available interrupts	<input type="checkbox"/> Force DMA channels Interrupts
NVIC Interrupt Table	Enabled	Preemption Priority	Sub Priority
Non maskable interrupt	<input checked="" type="checkbox"/>	0	0
Hard fault interrupt	<input checked="" type="checkbox"/>	0	0
Memory management fault	<input checked="" type="checkbox"/>	0	0
Pre-fetch fault, memory access fault	<input checked="" type="checkbox"/>	0	0
Undefined instruction or illegal state	<input checked="" type="checkbox"/>	0	0
System service call via SWI instruction	<input checked="" type="checkbox"/>	0	0
Debug monitor	<input checked="" type="checkbox"/>	0	0
Pendable request for system service	<input checked="" type="checkbox"/>	0	0
Time base: System tick timer	<input checked="" type="checkbox"/>	5	0
PVD interrupt through EXTI line 16	<input type="checkbox"/>	0	0
Flash global interrupt	<input type="checkbox"/>	0	0
RCC global interrupt	<input type="checkbox"/>	0	0
DMA1 channel2 global interrupt	<input type="checkbox"/>	0	0
DMA1 channel3 global interrupt	<input checked="" type="checkbox"/>	3	0
ADC1 and ADC2 interrupts	<input checked="" type="checkbox"/>	2	0
EXTI line[9:5] interrupts	<input type="checkbox"/>	0	0
TIM1 break and TIM15 interrupts	<input checked="" type="checkbox"/>	4	1
TIM1 update and TIM16 interrupts	<input checked="" type="checkbox"/>	4	0
TIM1 trigger, commutation and TIM17 interrupts	<input type="checkbox"/>	0	0
TIM1 capture compare interrupt	<input type="checkbox"/>	0	0
TIM2 global interrupt	<input checked="" type="checkbox"/>	3	0
SPI2 global interrupt	<input type="checkbox"/>	0	0
USART2 global interrupt / USART2 wake-up interrupt through EXTI line 26	<input type="checkbox"/>	0	0
USART3 global interrupt / USART3 wake-up interrupt through EXTI line 28	<input checked="" type="checkbox"/>	7	0
EXTI line[15:10] interrupts	<input checked="" type="checkbox"/>	3	0
TIM6 global interrupt and DAC1 underrun interrupt	<input type="checkbox"/>	0	0
TIM7 global interrupt	<input type="checkbox"/>	0	0
Floating point unit interrupt	<input type="checkbox"/>	0	0

Figure 5-22

NVIC Mode and Configuration

Configuration				
<input checked="" type="radio"/> NVIC <input checked="" type="radio"/> Code generation				
Enabled interrupt table	Select for init sequence ordering	Generate Enable in Init	Generate IRQ handler	Call HAL handler
Non maskable interrupt	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Hard fault interrupt	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Memory management fault	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Pre-fetch fault, memory access fault	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Undefined instruction or illegal state	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
System service call via SWI instruction	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Debug monitor	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Pendable request for system service	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Time base: System tick timer	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
DMA1 channel3 global interrupt	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
ADC1 and ADC2 interrupts	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
TIM1 break and TIM15 interrupts	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
TIM1 update and TIM16 interrupts	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
TIM2 global interrupt	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
USART3 global interrupt / USART3 wake-up interrupt through EXTI line 28	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
EXTI line[15:10] interrupts	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>

Figure 5-23

Step 10: Adjusting the clock tree in “Clock Configuration” page.
Modify the settings as the red highlighted as below.

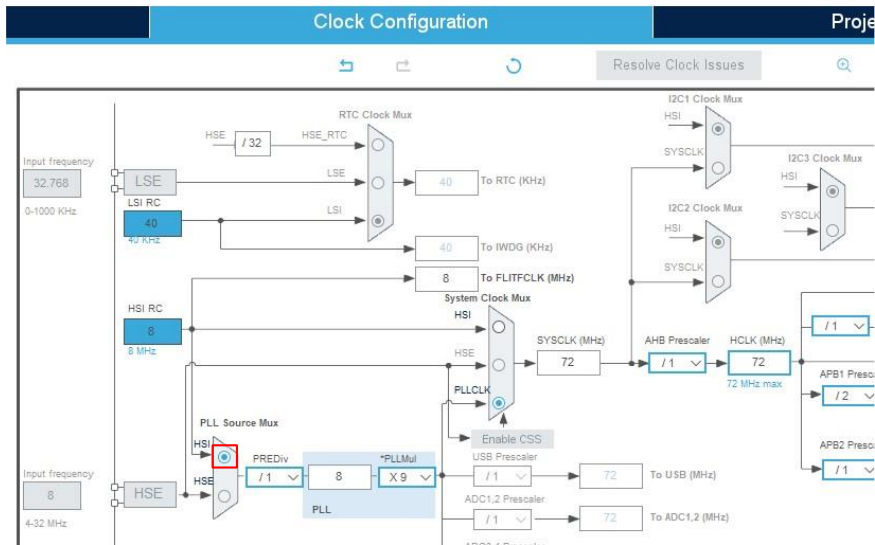


Figure 5-24

5-3 Generate Source Code in STM32CubeMX

Step 1: In “**Project Manager > Project**” page, extended the minimum heap size to **0x1000**, and select Firmware Package as “**STM32Cube FW_F3 V1.11.1**”.

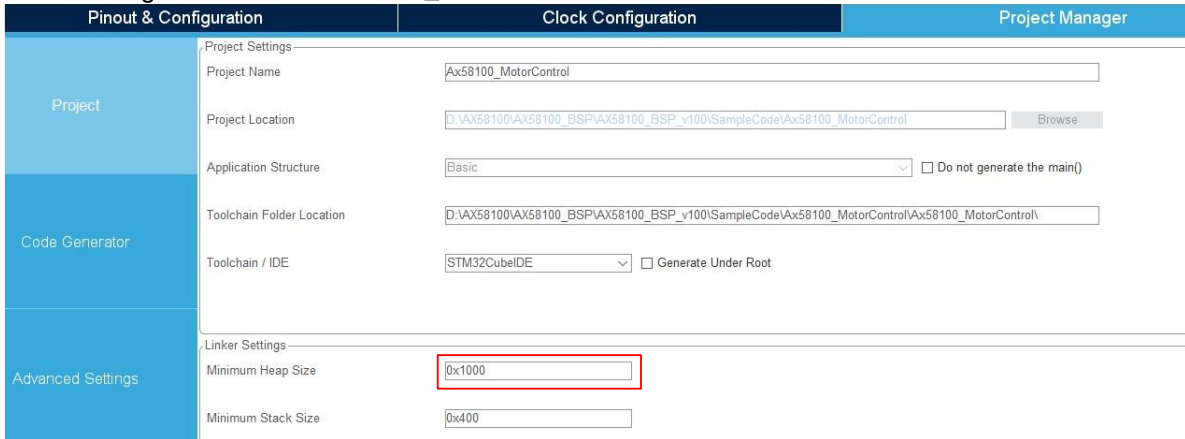


Figure 5-25

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

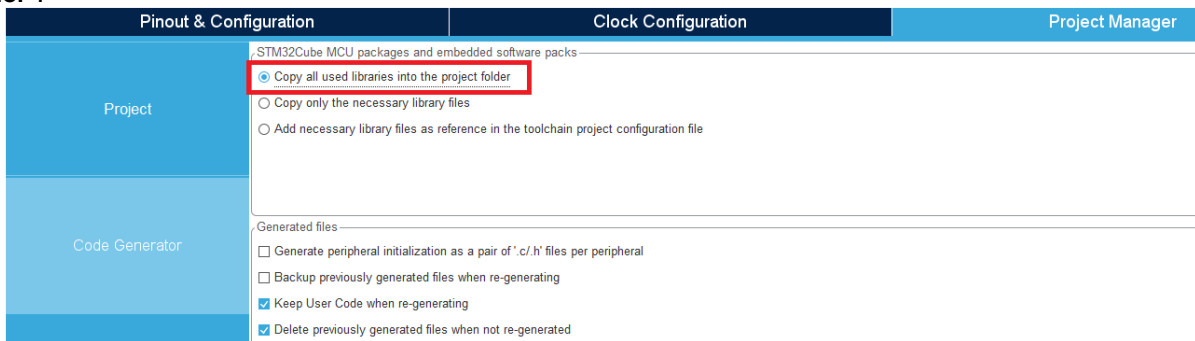
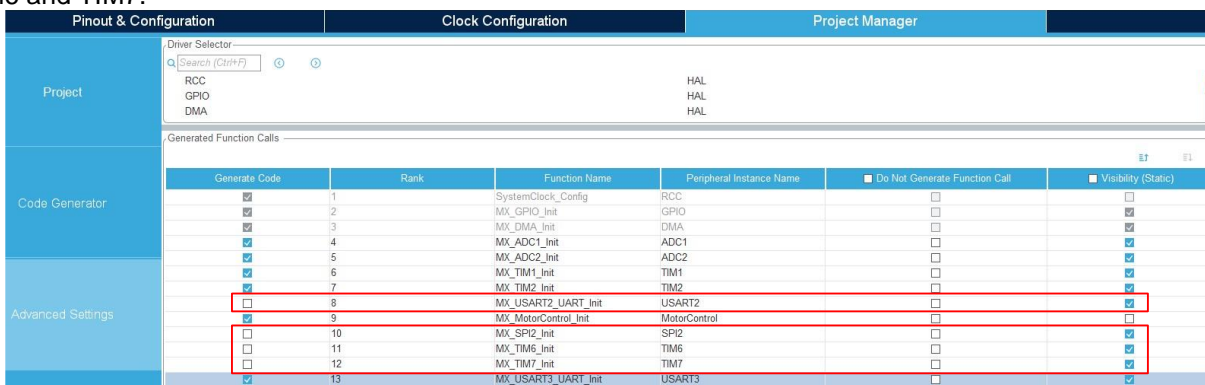


Figure 5-26

Step 3: In “**Project Manager > Advanced Settings**” page, disable initial code generation of USART2, SPI2, TIM6 and TIM7.



Generate Code	Rank	Function Name	Peripheral Instance Name	Do Not Generate Function Call	Visibility (Static)
<input checked="" type="checkbox"/>	1	SystemClock_Config	RCC	<input type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	2	MX_GPIO_Init	GPIO	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/>	3	MX_DMA_Init	DMA	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/>	4	MX_ADC1_Init	ADC1	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/>	5	MX_ADC2_Init	ADC2	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/>	6	MX_TIM1_Init	TIM1	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/>	7	MX_TIM2_Init	TIM2	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<input type="checkbox"/>	8	MX_USART2_UART_Init	USART2	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<input type="checkbox"/>	9	MX_MotorControl_Init	MotorControl	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	10	MX_SPI2_Init	SPI2	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<input type="checkbox"/>	11	MX_TIM6_Init	TIM6	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<input type="checkbox"/>	12	MX_TIM7_Init	TIM7	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/>	13	MX_USART3_UART_Init	USART3	<input type="checkbox"/>	<input checked="" type="checkbox"/>

Figure 5-27

Step 4: Generates the source code.

5-4 Source Code Adjustment

Step 1: Add the AX58100 source package.

Copy the folders as below to your STM32CubeMX project folder.

BSP_ROOT\SampleCode\Ax58100_MotorControl\AX58100

BSP_ROOT\SampleCode\Ax58100_MotorControl\Binary

BSP_ROOT\SampleCode\Ax58100_MotorControl\For_SSC_Tool

BSP_ROOT\SampleCode\Ax58100_MotorControl\For_TwinCAT

Step 2: Adjust the Vector Table Address.

As highlighted in the figure below, modify file content in the path:

BSP_ROOT\SampleCode\Ax58100_MotorControl\Src\system_stm32f3xx.c

```

91
92 #if !defined (HSI_VALUE)
93 #define HSI_VALUE ((uint32_t)8000000) /*!< Default value of the Internal oscillator
94                                     This value can be provided and adjusted in the
95 #endif /* HSI_VALUE */
96
97 /*!< Uncomment the following line if you need to relocate your vector Table in
98 Internal SRAM. */
99 /* #define VECT_TAB_SRAM */
100 #define VECT_TAB_OFFSET 0x4000 /*!< Vector Table base offset field.
101                                This value must be a multiple of 0x200. */
102 /**
103  * @}
104  */
105

```

Figure 5-28

Step 3: Adjust the Linker setting.

As highlighted in the figure below, modify file content in the path:

- **For Keil MDK IDE:**

Please skip this step.

- **For STM32CubeIDE:**

BSP_ROOT\SampleCode\Ax58100_MotorControl\STM32CubeIDE\STM32F303RETX_FLASH.ld

```

45 /* Memories definition */
46 MEMORY
47 {
48   CCMRAM   (xrw)   : ORIGIN = 0x10000000, LENGTH = 16K
49   RAM      (xrw)   : ORIGIN = 0x20000000, LENGTH = 64K
50   FLASH   (rx)    : ORIGIN = 0x8004000,  LENGTH = 246K
51 }
52

```

Figure 5-29

Step 4: Modify source code.

As highlighted in the figure below, modify file content in the path:
BSP_ROOT\SampleCode\Ax58100_MotorControl\Src\main.c

```

22 /* Private includes -----
23 /* USER CODE BEGIN Includes */
24 #include "AX58400_Hw.h"
25 #include "applInterface.h"
26 #include "ax58400.h"
27 #include "AX58400_MotorControl.h"
28 #include "bootmode.h"
29 #include "ax58400_FoeAppl.h"
30 /* USER CODE END Includes */
    ● ● ●
159 /* Initialize all configured peripherals */
160 MX_GPIO_Init();
161 /* USER CODE BEGIN 2 */
162 #if (AX58400_DEBUG_ENABLE)
163 AX_UART_Init();
164 console_Init();
165 DBG_PRINT("%s M4 Firmware v%s\r\n", DEVICE_NAME, DEVICE_SW_VERSION);
166 #endif //if (AX58400_DEBUG_ENABLE)
167 CC_Init();
168 #if (SSC_STACK_ENABLE)
169 if (HW_Init())
170 {
171     HW_Release();
172 }
173 else
174 {
175     MainInit();
176     Cia402_Init();
177     AX58400_FoeInit();
178 }
179 #endif
180 /* USER CODE END 2 */
181
182 /* Infinite loop */
183 /* USER CODE BEGIN WHILE */
184 while (1)
185 {
186 #if (AX58400_DEBUG_ENABLE)
187     console_Task();
188 #endif //if (AX58400_DEBUG_ENABLE)
189
190 #if (SSC_STACK_ENABLE)
191     MainLoop();
192     BL_Reboot();
193 }
194 #if (ESC_PHY_ENHANCEMENT)
195     HW_PhyEnhancement();
196 #endif //if (ESC_PHY_ENHANCEMENT)
197
198 #endif //if (SSC_STACK_ENABLE)
199

```

Figure 5-30

As highlighted in the figure below, modify file content in the path:

BSP_ROOT\SampleCode\Ax58100_MotorControl\Src\stm32f30x_mc_it.c

```

33 #include "mcp_config.h"
34 /* USER CODE BEGIN Includes */
35 #include "ax58100_utils.h"
36 #include "AX58100_Hw.h"
37 /* USER CODE END Includes */
    ● ● ●
263 void SysTick_Handler(void)
264 {
265
266 #ifdef MC_HAL_1S_USED
267 static uint8_t SystickDividerCounter = SYSTICK_DIVIDER;
268 /* USER CODE BEGIN SysTick_IR0n 0 */
269 #if (AX58100_DEBUG_ENABLE)
270 if (SystickDividerCounter == SYSTICK_DIVIDER)
271 {
272     console_TimeTick();
273 }
274 #endif //if (AX58100_DEBUG_ENABLE)
275
276 /* USER CODE END SysTick_IR0n 0 */
277 if (SystickDividerCounter == SYSTICK_DIVIDER)
278 {

```

Figure 5-31

As highlighted in the figure below, modify file content in the path:
BSP_ROOT\SampleCode\Ax58100_MotorControl\Src\mc_tasks.c

```

37
38 /* USER CODE BEGIN Includes */
39 #include "ax58100_utils.h"
40 /* USER CODE END Includes */
    • • •
351 __weak void TSK_MediumFrequencyTaskM1(void)
352 {
353 /* USER CODE BEGIN MediumFrequencyTask M1 0 */
    • • •
501     case RUN:
502     {
503         if (MCL_STOP == Mci[M1].DirectCommand)
504         {
505             TSK_MF_StopProcessing(M1);
506         }
507         else
508         {
509             /* USER CODE BEGIN MediumFrequencyTask M1 2 */
510             // #if (MC_STACK_ENABLE)
511             if (McInfo.Status.LoopCtrlMode == LC_SPD_CTRL)
512             {
513                 MCL_ExecBufferedCommands(&Mci[M1]);
514                 FOC_CalcCurrRef(M1);
515             }
516             else
517             {
518                 TC_PositionRegulation(pPosCtrl[M1]);
519                 MCL_ExecBufferedCommands(&Mci[M1]);
520                 FOC_CalcCurrRef(M1);
521             }
522             break;
523             // #endif (MC_STACK_ENABLE)
524             /* USER CODE END MediumFrequencyTask M1 2 */
525

```

Figure 5-32

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

5-6 IDE Setting Adjustment

- **For Keil MDK IDE:**

Double clicks the file as below to open project.

BSP_ROOT\SampleCode\Ax58100_MotorControl\MDK-ARM\Ax58100_MotorControl.uvproj

Step 1: Add source code both on dual cores.

Click on **"Project > Manage > Project Items..."** to open **"Manage Project Items"** dialog, then add groups and files as the figure below.

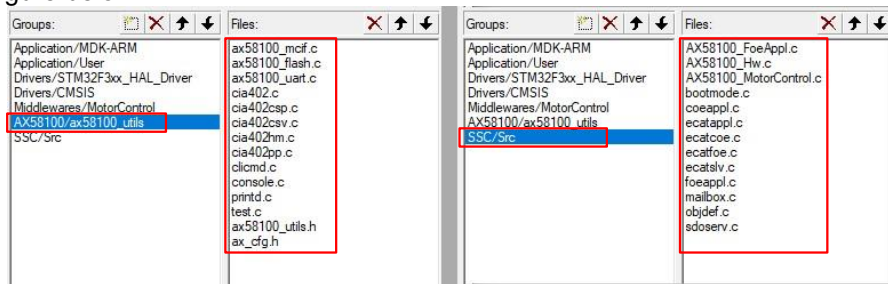


Figure 5-33

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.

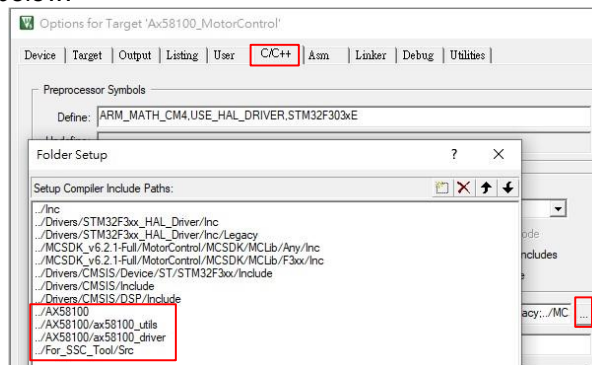


Figure 5-34

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.

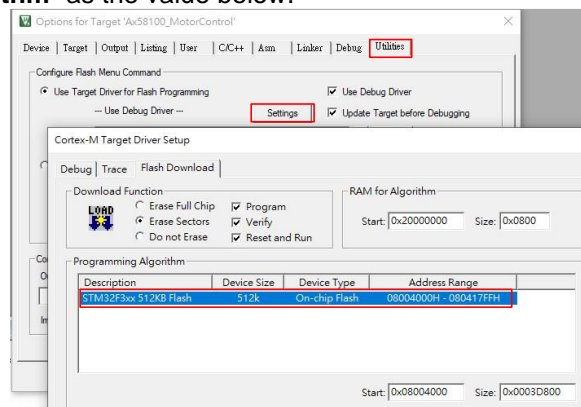


Figure 5-35

Start = 0x08004000, Size = 0x0003D800

Step 4: Change “Linker” settings.

Find the setting of “**Project > Options for Target ‘xxx’... > Linker**”, then checked the “**Use Memory Layout from Target Dialog**”.

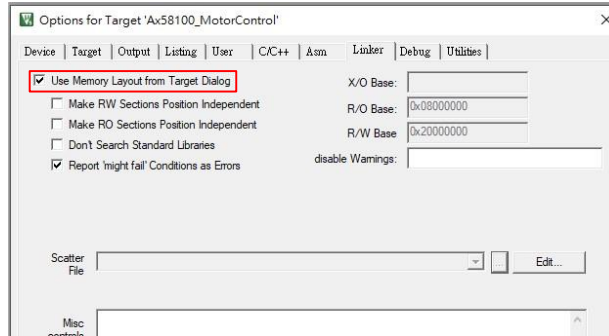


Figure 5-36

Step 5: Add post build actions.

Find the setting of “**Project > Options for Target ‘xxx’... > User > After Build/Rebuild**”, then add actions as the value below.

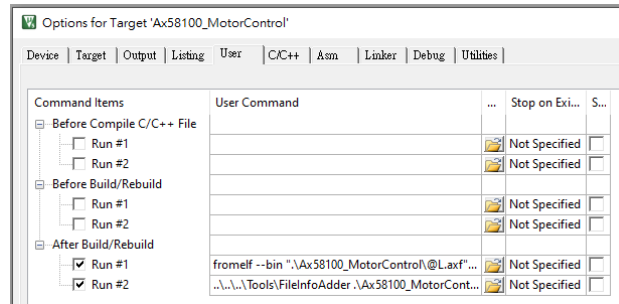


Figure 5-37

Run #1: `fromelf --bin ".\Ax58100_MotorControl\@L.axf" --output ".\Ax58100_MotorControl\@L.bin"`

Run #2:

`..\..\Tools\FileInfoAdder .\Ax58100_MotorControl\Ax58100_MotorControl.bin ..\For_TwinCAT\FoE\AX58100_MotorControl.efw`

Step 6: Modify “Target Options”.

Find the setting of “**Project > Options for Target ‘xxx’... > Target > Read/Write Memory Areas**”, then modify it as the value below.

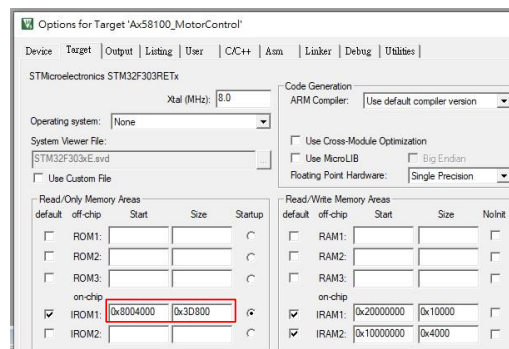


Figure 5-38

IROM1 Start = 0x8004000, Size = 0x3D800

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

- **For STM32CubeIDE:**

Double clicks the file as below to open project.

BSP_ROOT\SampleCode\Ax58100_MotorControl\STM32CubeIDE\project

Step 1: Add source code.

On “**Project Explorer**”, add virtual folders and linking the source files.

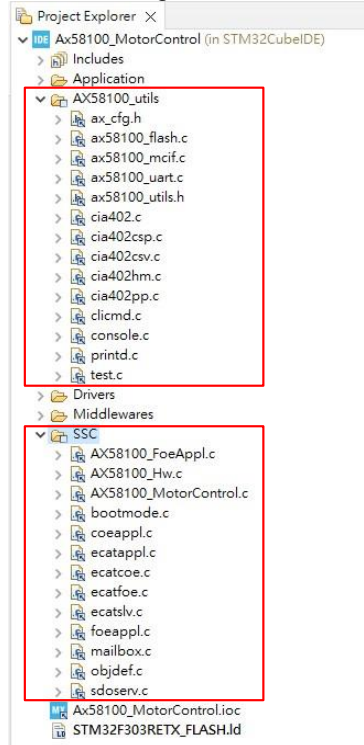


Figure 5-39

- **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.

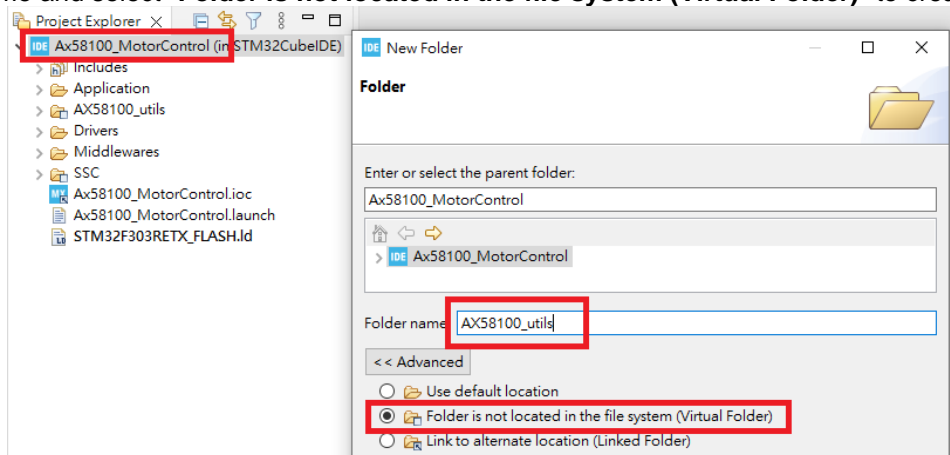


Figure 5-40

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

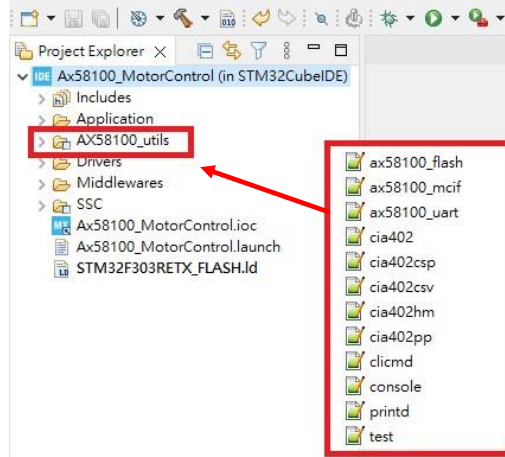


Figure 5-41

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.

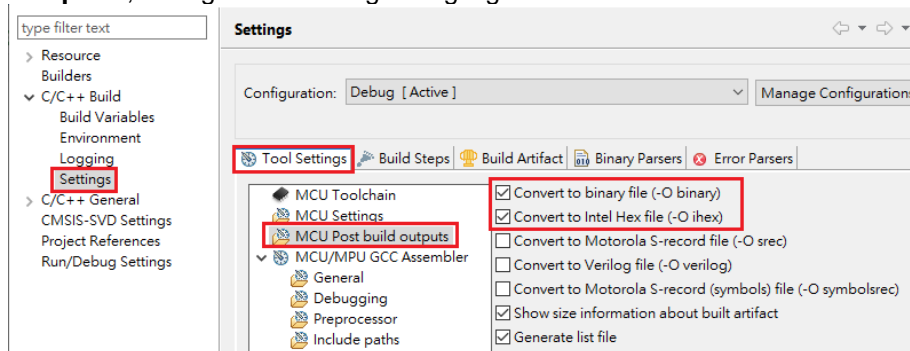


Figure 5-42

Step 3: Add include path both on dual cores.

Find setting on “**Project > Properties > C/C++ Build > Settings > Tool Settings > MCU/MCP GCC Compiler > Include paths**”, then add paths as the figure below.

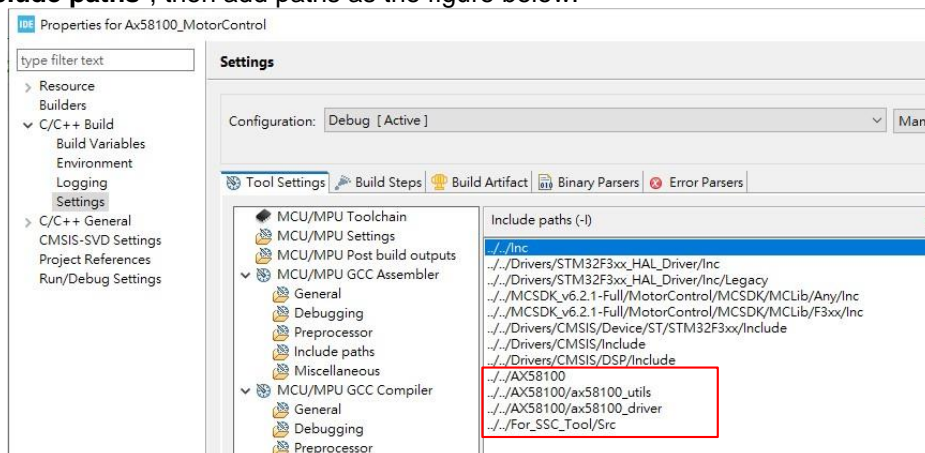


Figure 5-43

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.

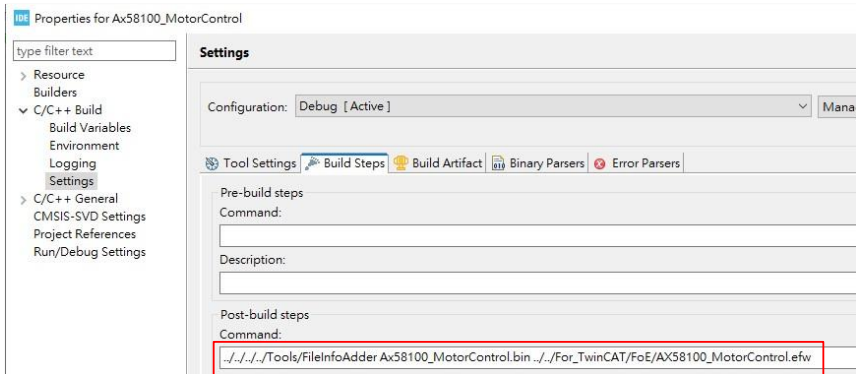


Figure 5-44

..\..\..\Tools/FileInfoAdder Ax58100_MotorControl.bin ..\..\For_TwinCAT/FoE/AX58100_MotorControl.efw

Now, entire building procedure has done for STM32CubeIDE.

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

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\Ax58100_MotorControl\MDK-ARM\Ax58100_MotorControl.uvprojx.

Step 2: Execute “Rebuild all target files” under “Menu Bar > Project”.

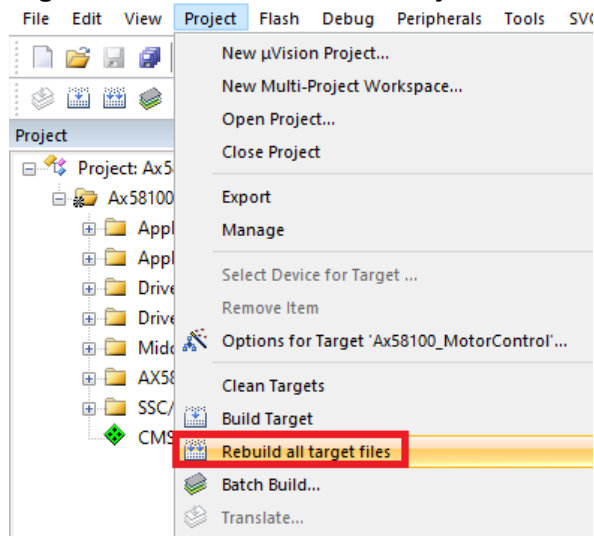


Figure 6-1

- For STM32CubeIDE:

Step 1: Double clicks the project file as below path to open project.

BSP_ROOT\SampleCode\Ax58100_MotorControl\STM32CubeIDE\project

Step 2: Select the project node and execute “Build Project” under “Menu Bar > Project”.

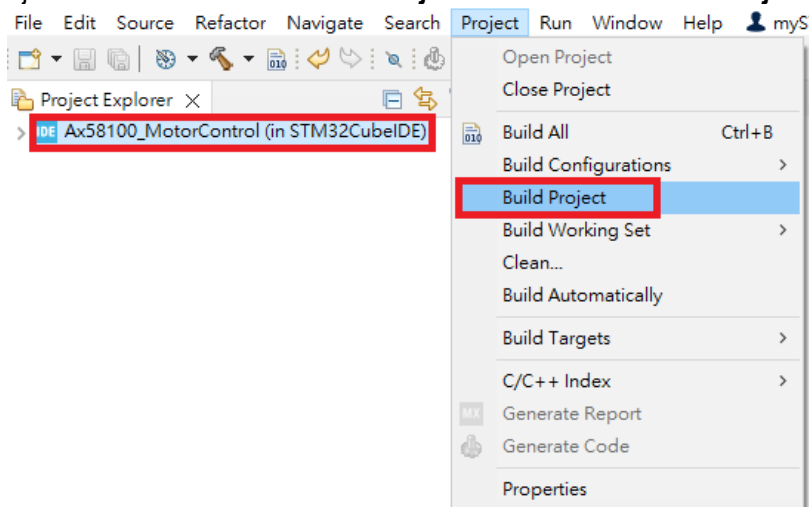


Figure 6-2

6-2 Download Bootloader Firmware

Step 1: Open STM32CubeProgrammer utility and connect to the MCU board.

Step 2: Select the “Erasing & Programming” page, then click “Browse” button to select bootloader firmware binary file in the path below.

BSP_ROOT\SampleCode\Ax58100_Bldr\Binary\Ax58100_Bldr_vX.X.X

Please ensure the “Start address” is 0x08000000.

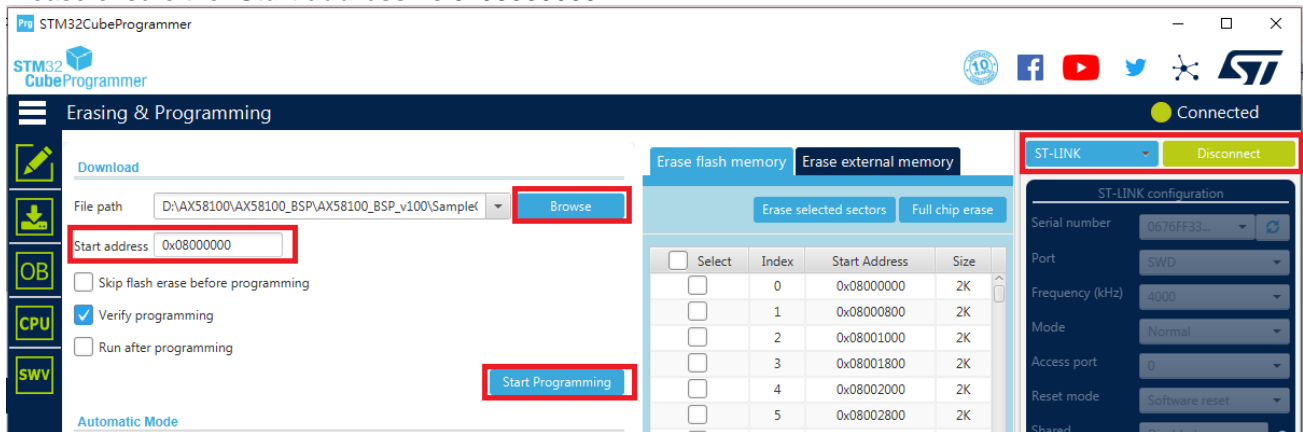


Figure 6-3

Step 3: Press “Start Programming” button to download bootloader firmware.

6-3 Download Application Firmware

- For Keil MDK IDE:

Step 1: Click “Menu bar > Flash > Download” to download firmware into MCU.

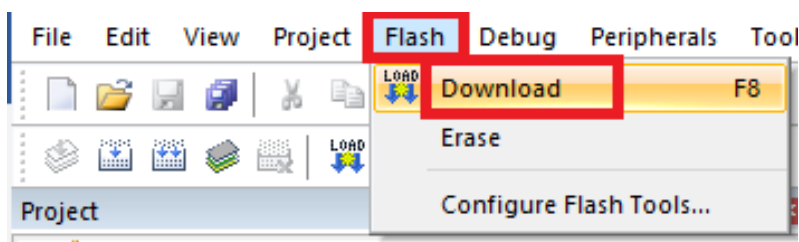


Figure 6-4

- For STM32CubeIDE:

Step 1: Click “Menu bar > Run > Run As > STM32 C/C++ Application” to download firmware into MCU.

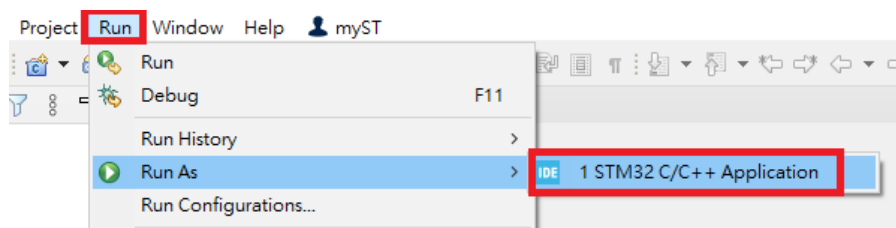


Figure 6-5

- **Using STM32CubeProgrammer Utility:**

Step 1: Similarly, please ensure the utility has connected to target MCU board.

Step 2: Select the “**Erasing & Programming**” page, then click “**Browse**” button to select Motor Control firmware binary file in the path below.

- **For Keil MDK IDE:**
BSP_ROOT\SampleCode\Ax58100_MotorControl\MDK-ARM\Ax58100_MotorControl\Ax58100_MotorControl.bin
- **For STM32CubeIDE:**
BSP_ROOT\SampleCode\Ax58100_MotorControl\STM32CubeIDE\Debug\Ax58100_MotorControl.bin

Please ensure the “**Start address**” is 0x08004000.

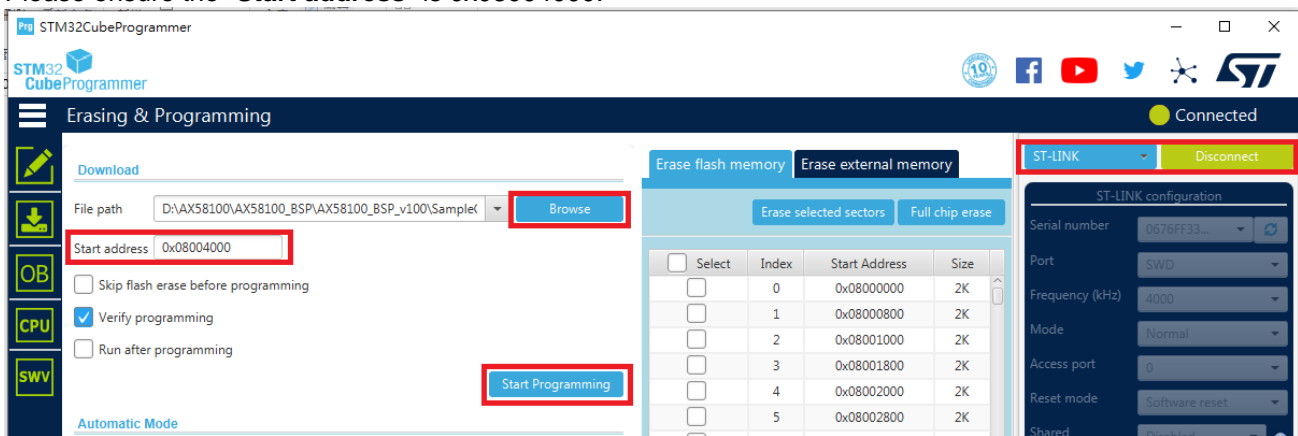


Figure 6-6

Step 3: Press “**Start Programming**” button to download Motor Control firmware.

7. Basic Operation of TwinCAT

Please refer to the file “AX58x00_TwinCAT_UserGuide” Chapter 3.

8. PLC Application

Please refer to the file “AX58x00_TwinCAT_UserGuide” Chapter 5.

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 STM32F303RE as below.

Flash area	Flash memory addresses	Size (bytes)	Name
Bootloader area (Page 0~7)	0x0800 0000 - 0x0800 07FF	2 K	Page 0
	0x0800 0800 - 0x0800 0FFF	2 K	Page 1
	0x0800 1000 - 0x0800 17FF	2 K	Page 2
	0x0800 1800 - 0x0800 1FFF	2 K	Page 3
	0x0800 3800 - 0x0800 3FFF		Page 7
Main memory	0x0800 4000 - 0x0800 47FF	.	Page 8
	0x0804 1000 - 0x0804 17FF	.	Page 130
	0x0804 1800 - 0x0804 1FFF	.	Page 131
	0x0807 F000 - 0x0807 F7FF	.	Page 254
Information block	0x0807 F800 - 0x0807 FFFF	2 K	Page 255
	0x1FFF D800 - 0x1FFF F7FF	8 K	System memory
	0x1FFF F800 - 0x1FFF F80F	16	Option bytes

Figure 9-1

- **Bootloader area (Address=0x08000000, Size=16Kbytes):**
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.
- **Runtime image area (Address=0x08004000, Size=246Kbytes):**
Used to run application firmware (as the Motor Control firmware in here), it is also responsible for store the received .efw file into “Flash buffer area” via FoE data transfer.
- **Flash buffer area (Address=0x08041800, Size=247Kbytes):**
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\Ax58100_MotorControl\For_TwinCAT\FoE\AX58100_MotorControl.efw

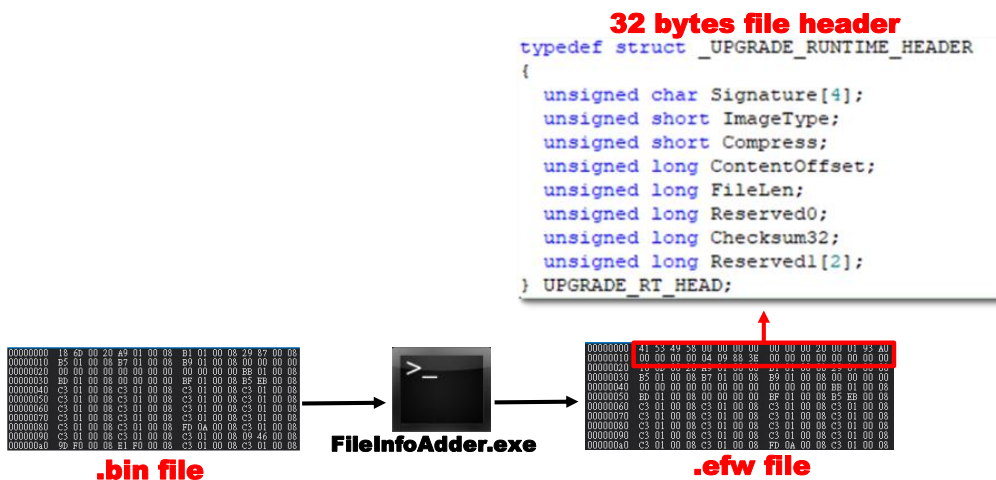


Figure 9-2

9-1 FoE Firmware Upgrade

Here assume the EtherCAT network has established and the device has been under OP mode, then please follow the steps below to do firmware upgrade via FoE.

Step 1: Select “**Solution Explorer > AX58100_MotorControl > I/O > Devices > XXX > Online**”, then right click on target device and select “**Firmware Update**”..

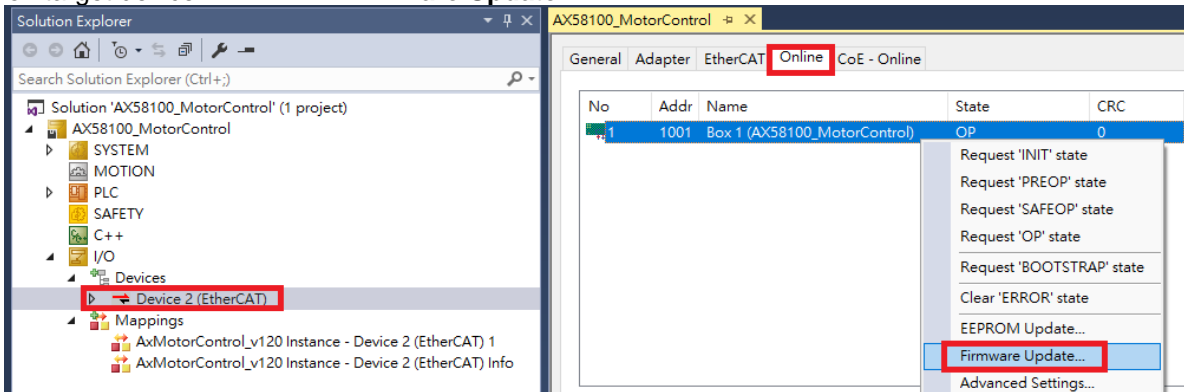


Figure 9-3

Step 2: Select the file as path below, and click “**OK**” to start application firmware upgrading.
“**BSP_ROOT\SampleCode\Ax58100_MotorControl\For_TwinCAT\FoE\AX58100_MotorControl.efw**”

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



Figure 9-4

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

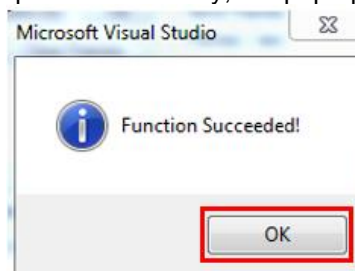


Figure 9-5

10. Object Dictionary

These reference design follows CiA402 device profile and used as an interface to integrate Motor Control Stack and EtherCAT Slave Stack, the CiA402 protocol implement a state machine and object dictionary for external EtherCAT master access, Master side can control motor, switch operation mode and monitor motor status by manipulation of object dictionary.

10-1 Objects Description

This section lists out all supported objects in the reference design, please note that some objects may not implemented actual functions with Motor Control Stack side, but they are accessible in EtherCAT Slave side, so developer can still add application functions on them.

Object type definition is as below

Object Type	Description
VAR	A single value
ARRAY	A multiple data field object where each data field is a simple variable of same basic data type.
RECORD	A multiple data field object where the data fields may be any combination of simple variables.

Table 10-1

0x1000~0x1FFF area object definition

Index	Object Type	Name	Data Type	Access	Remark
0x1000	VAR	Device type	UNSIGNED32	RO	
0x1001	VAR	Error register	UNSIGNED8	RO	
0x1008	VAR	Device name	Visible String	RO	
0x1009	VAR	Hardware version	Visible String	RO	
0x100A	VAR	Software version	Visible String	RO	
0x1018	RECORD	Identity	Identity	RO	
0x10F1	RECORD	Error settings	-	RO	
0x10F8	VAR	Time stamp	UNSIGNED64	RO	
0x1C00	ARRAY	Sync manager type	UNSIGNED8	RO	
0x1C32	RECORD	Sync manager output parameters	SM Parameter	RO	
0x1C33	RECORD	Sync manager input parameters	SM Parameter	RO	
0x1600	RECORD	Rx pdo mapping of csp/csv	PDO Mapping	RO	Mapped objects: Controlword Target position Target velocity Modes of operation
0x1601	RECORD	Rx pdo mapping of csp	PDO Mapping	RO	Mapped objects: Controlword Target position
0x1602	RECORD	Rx pdo mapping of csv	PDO Mapping	RO	Mapped objects: Controlword Target velocity
0x1A00	RECORD	Tx pdo mapping of csp/csv	PDO Mapping	RO	Mapped objects Statusword Position actual value Velocity actual value Modes of operation display
0x1A01	RECORD	Tx pdo mapping of csp	PDO Mapping	RO	Mapped objects: Statusword Position actual value

0x1A02	RECORD	Tx pdo mapping of csv	PDO Mapping	RO	Mapped objects: Statusword Position actual value
0x1C12	VAR	SM2 assignment	UNSIGNED16	RW	Assign by default the csv process data mapping
0x1C13	VAR	SM3 assignment	UNSIGNED16	RW	Assign by default the csv process data mapping

Table 10-2

0x2000–0x3FFF object definition

Index	Object Type	Name	Data Type	Access	Mappable	Remark
0x3000	VAR	PosKpGain	INTEGER16	RW	N	
0x3001	VAR	PosKpDivisor	INTEGER16	RO	N	
0x3002	VAR	PosKiGain	INTEGER16	RW	N	
0x3003	VAR	PosKiDivisor	INTEGER16	RO	N	
0x3004	VAR	PosKdGain	INTEGER16	RW	N	
0x3005	VAR	PosKdDivisor	INTEGER16	RO	N	
0x3010	VAR	SpdKpGain	INTEGER16	RW	N	
0x3011	VAR	SpdKpDivisor	INTEGER16	RO	N	
0x3012	VAR	SpdKiGain	INTEGER16	RW	N	
0x3013	VAR	SpdKiDivisor	INTEGER16	RO	N	
0x3014	VAR	SpdKdGain	INTEGER16	RW	N	
0x3015	VAR	SpdKdDivisor	INTEGER16	RO	N	
0x3020	VAR	TrqKpGain	INTEGER16	RW	N	
0x3021	VAR	TrqKpDivisor	INTEGER16	RO	N	
0x3022	VAR	TrqKiGain	INTEGER16	RW	N	
0x3023	VAR	TrqKiDivisor	INTEGER16	RO	N	
0x3024	VAR	TrqKdGain	INTEGER16	RW	N	
0x3025	VAR	TrqKdDivisor	INTEGER16	RO	N	

Table 10-3

0x6000~0xFFFF object definition

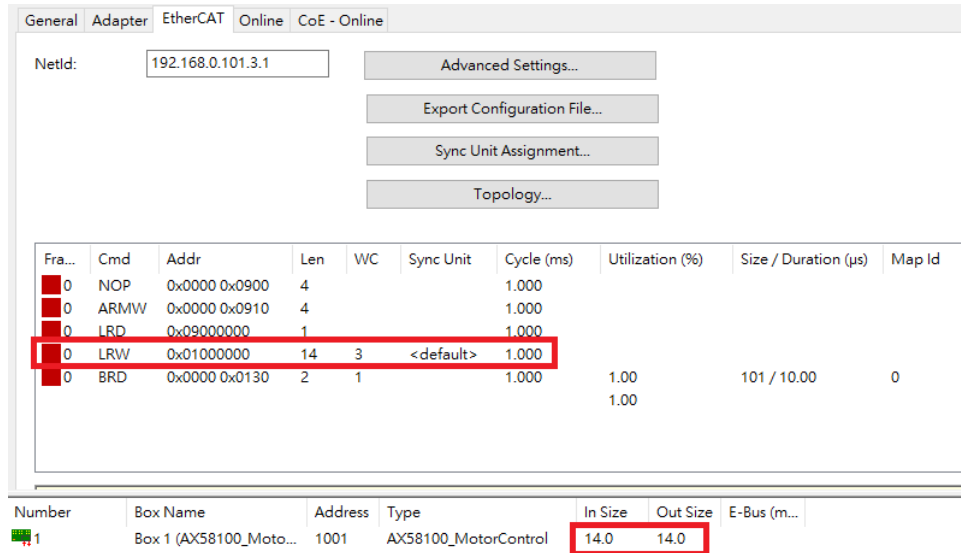
Index	Object Type	Name	Data Type	Access	Mappable	Remark
0x603F	VAR	Error code	UNSIGNED16	RO	N	
0x6040	VAR	Controlword	UNSIGNED16	RW	Y	
0x6041	VAR	Statusword	UNSIGNED16	RO	Y	
0x604F	VAR	VI Ramp Function Time	UNSIGNED 32	RW	N	
0x605A	VAR	Quick stop option code	INTEGER16	RW	N	
0x605B	VAR	Shutdown option code	INTEGER16	RW	N	
0x605C	VAR	Disable operation option code	INTEGER16	RW	N	
0x605E	VAR	Fault reaction option code	INTEGER16	RW	N	
0x6060	VAR	Modes of operation	INTEGER8	RW	Y	
0x6061	VAR	Modes of operation display	INTEGER8	RO	Y	
0x6064	VAR	Position actual value	INTEGER32	RO	Y	
0x606C	VAR	Velocity actual value	INTEGER32	RO	Y	
0x6077	VAR	Torque actual value	INTEGER16	RO	N	
0x607A	VAR	Target position	INTEGER32	RW	Y	
0x607C	VAR	Home offset	INTEGER32	RW	N	
0x607D	ARRAY	Software position limit	-	RW	N	
0x6085	VAR	Quick stop deceleration	UNSIGNED32	RW	N	
0x6098	VAR	Homing method	INTEGER8	RW	N	
0x6099	RECORD	Homing speed	-	RW	N	
0x609A	VAR	Homing acceleration	UNSIGNED32	RW	N	
0x60C2	RECORD	Interpolation time period	-	RW	N	
0x60E3	RECORD	Supported homing methods	-	RO	N	
0x60FD	VAR	Digital Inputs	UNSIGNED32	RO	N	
0x60FF	VAR	Target velocity	INTEGER32	RW	Y	
0x6502	VAR	Supported drive modes	UNSIGNED32	RO	N	
0xF000	RECORD	Modular device profile	-	RO	N	
0xF010	VAR	Module profile list	UNSIGNED32	RO	N	

Table 10-4

11. Performance Evaluation

11-1 Minimum DC Cycle Time

With TwinCAT Master, the DC and PD conditions are shown as below. Master use LRW command to do PD transfer with 14bytes input and 14bytes output, under this setting, the minimum DC cycle time can down to 250us.



The screenshot shows the TwinCAT configuration interface for an EtherCAT adapter. The 'EtherCAT' tab is active, displaying the NetId as 192.168.0.101.3.1. Below the configuration options is a table of data points:

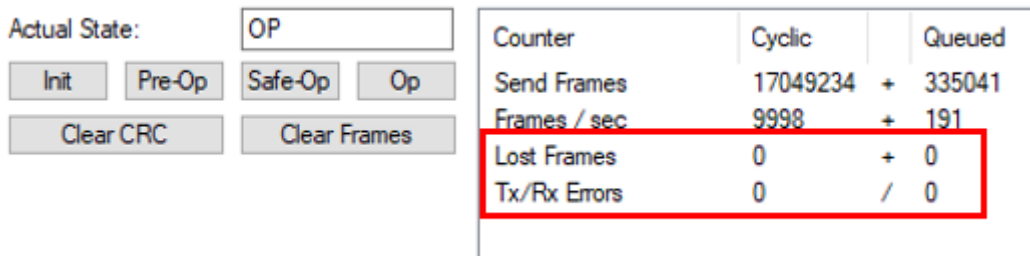
Fra...	Cmd	Addr	Len	WC	Sync Unit	Cycle (ms)	Utilization (%)	Size / Duration (µs)	Map Id
0	NOP	0x0000 0x0900	4			1.000			
0	ARMW	0x0000 0x0910	4			1.000			
0	LRD	0x09000000	1			1.000			
0	LRW	0x01000000	14	3	<default>	1.000			
0	BRD	0x0000 0x0130	2	1		1.000	1.00	101 / 10.00	0

Below the table is a summary table for the motor control box:

Number	Box Name	Address	Type	In Size	Out Size	E-Bus (m...
1	Box 1 (AX58100_Moto...	1001	AX58100_MotorControl	14.0	14.0	

Figure 11-1

Without frame lost and Tx/Rx error at Master side.



The screenshot shows the TwinCAT status interface. The 'Actual State' is 'OP'. Below the state indicators are buttons for 'Init', 'Pre-Op', 'Safe-Op', 'Op', 'Clear CRC', and 'Clear Frames'. To the right is a table of performance counters:

Counter	Cyclic	Queued
Send Frames	17049234	+ 335041
Frames / sec	9998	+ 191
Lost Frames	0	+ 0
Tx/Rx Errors	0	/ 0

Figure 11-2

Also, without detected SM-Event Missed, Cycle Time Too Small and Sync Error at Slave side.

Index	Name	Flags	Value	Unit
1C32:0	SM output parameter	RO	> 32 <	
1C32:01	Synchronization Type	RW	0x0002 (2)	
1C32:02	Cycle Time	RO	0x000F4240 (1000000)	
1C32:04	Synchronization Types supported	RO	0x401F (16415)	
1C32:05	Minimum Cycle Time	RO	0x0003D090 (250000)	
1C32:06	Calc and Copy Time	RO	0x00000000 (0)	
1C32:08	Get Cycle Time	RW	0x0000 (0)	
1C32:09	Delay Time	RO	0x00000000 (0)	
1C32:0A	Sync0 Cycle Time	RW	0x000F4240 (1000000)	
1C32:0B	SM-Event Missed	RO	0x0000 (0)	
1C32:0C	Cycle Time Too Small	RO	0x0000 (0)	
1C32:0D	Shift Time Too Short Counter	RO	0x0000 (0)	
1C32:20	Sync Error	RO	FALSE	
1C33:0	SM input parameter	RO	> 32 <	
1C33:01	Synchronization Type	RW	0x0002 (2)	
1C33:02	Cycle Time	RO	0x000F4240 (1000000)	
1C33:04	Synchronization Types supported	RO	0x401F (16415)	
1C33:05	Minimum Cycle Time	RO	0x0003D090 (250000)	
1C33:06	Calc and Copy Time	RO	0x00000000 (0)	
1C33:08	Get Cycle Time	RW	0x0000 (0)	
1C33:09	Delay Time	RO	0x00000000 (0)	
1C33:0A	Sync0 Cycle Time	RW	0x000F4240 (1000000)	
1C33:0B	SM-Event Missed	RO	0x0000 (0)	
1C33:0C	Cycle Time Too Small	RO	0x0000 (0)	
1C33:0D	Shift Time Too Short Counter	RO	0x0000 (0)	
1C33:20	Sync Error	RO	FALSE	

Figure 11-3

12. Tuning the PID Loop Gains with TwinCAT

Here will show you how to add PID gain objects into object dictionary and tuning PID gain over EtherCAT. For most of applications, you may need to extend your own objects, please follow the procedure below to do this.

12-1 Add New Objects into Dictionary

Step 1: Open the “AX58100_MotorControl.xlsx” file in below path.

BSP_ROOT\SampleCode\Ax58100_MotorControl\For_SSC_Tool\Import\Configuration\files

Step 2: In Vendor specify area (0x2000~0x3FFF), add PID gain objects for position, speed and torque loops as below.

0x2000 - 0x3FFF	Axis objects (0x2000 - 0x3FFF)							
0x3000	INT16	PosKpGain		rw	CoeRead	CoeWrite		Position loop proportional gain (Unit: digit)
0x3001	INT16	PosKpDivisor		ro	CoeRead	CoeWrite		Position loop proportional divisor (Unit: digit)
0x3002	INT16	PosKiGain	●	rw	CoeRead	CoeWrite		Position loop integral gain (Unit: digit)
0x3003	INT16	PosKiDivisor	●	ro	CoeRead	CoeWrite		Position loop integral divisor (Unit: digit)
0x3004	INT16	PosKdGain	●	rw	CoeRead	CoeWrite		Position loop derivative gain (Unit: digit)
0x3005	INT16	PosKdDivisor	●	ro	CoeRead	CoeWrite		Position loop derivative divisor (Unit: digit)
0x3010	INT16	SpdKpGain		rw	CoeRead	CoeWrite		Speed loop proportional gain (Unit: digit)
0x3011	INT16	SpdKpDivisor		ro	CoeRead	CoeWrite		Speed loop proportional divisor (Unit: digit)
0x3012	INT16	SpdKiGain		rw	CoeRead	CoeWrite		Speed loop integral gain (Unit: digit)
0x3013	INT16	SpdKiDivisor		ro	CoeRead	CoeWrite		Speed loop integral divisor (Unit: digit)
0x3014	INT16	SpdKdGain		rw	CoeRead	CoeWrite		Speed loop derivative gain (Unit: digit)
0x3015	INT16	SpdKdDivisor		ro	CoeRead	CoeWrite		Speed loop derivative divisor (Unit: digit)
0x3020	INT16	TrqKpGain		rw	CoeRead	CoeWrite		Torque loop proportional gain (Unit: digit)
0x3021	INT16	TrqKpDivisor		ro	CoeRead	CoeWrite		Torque loop proportional divisor (Unit: digit)
0x3022	INT16	TrqKiGain		rw	CoeRead	CoeWrite		Torque loop integral gain (Unit: digit)
0x3023	INT16	TrqKiDivisor		ro	CoeRead	CoeWrite		Torque loop integral divisor (Unit: digit)
0x3024	INT16	TrqKdGain		rw	CoeRead	CoeWrite		Torque loop derivative gain (Unit: digit)
0x3025	INT16	TrqKdDivisor		ro	CoeRead	CoeWrite		Torque loop derivative divisor (Unit: digit)

Figure 12-1

Step 3: Re-generate the SSC source code, for more detail, please refer to Chapter 5-6.

Step 4: Update the new ESI file into TwinCAT data base.

Normally, the ESI data base path is “C:\TwinCAT\3.1\Config\Io\EtherCAT”.

12-2 Link New Objects to Data Structure

Step 1: Add new members in CiA402_OBJ_T data structure, notes each data type must be same as the new objects on “AX58100_MotorControl.xlsx”, like this.

```

AX58100_MotorControl.h
415 typedef struct STRUCT_PACKED_START
416 {
417     .
418     .
419     .
420     .
421     .
422     .
423     .
424     .
425     .
426     .
427     .
428     .
429     .
430     .
431     .
432     .
433     .
434     .
435     .
436     .
437     .
438     .
439     .
440     .
441     .
442     .
443     .
444     .
445     .
446     .
447     .
448     .
449     .
450     .
451     /* Vendor Specified Objects */
452     INT16 PosKpGain0x3000;
453     INT16 PosKpDivisor0x3001;
454     INT16 PosKiGain0x3002;
455     INT16 PosKiDivisor0x3003;
456     INT16 PosKdGain0x3004;
457     INT16 PosKdDivisor0x3005;
458     INT16 SpdKpGain0x3010;
459     INT16 SpdKpDivisor0x3011;
460     INT16 SpdKiGain0x3012;
461     INT16 SpdKiDivisor0x3013;
462     INT16 SpdKdGain0x3014;
463     INT16 SpdKdDivisor0x3015;
464     INT16 TrqKpGain0x3020;
465     INT16 TrqKpDivisor0x3021;
466     INT16 TrqKiGain0x3022;
467     INT16 TrqKiDivisor0x3023;
468     INT16 TrqKdGain0x3024;
469     INT16 TrqKdDivisor0x3025;
470 } STRUCT_PACKED_END
    
```

Figure 12-2

Step 2: Links new object dictionary (OD) to new members of CiA402_OBJ_T structure in CiA402_Init () function, like this.

```

AX58100_MotorControl.c
321  /* Vendor Specified Objects */
322  case 0x3000:
323      HMEMCPY((UINT8*)&(pCiA402->PosKpGain0x3000), (UINT8*)&(PosKpGain0x3000), sizeof(PosKpGain0x3000));
324      pCiA402->PosKpGain0x3000 = PosKpGain0x3000;
325      pDiCEntry->pVarPtr = &(pCiA402->PosKpGain0x3000);
326      break;
327
328  case 0x3001:
329      HMEMCPY((UINT8*)&(pCiA402->PosKpDivisor0x3001), (UINT8*)&(PosKpDivisor0x3001), sizeof(PosKpDivisor0x3001));
330      pCiA402->PosKpDivisor0x3001 = PosKpDivisor0x3001;
331      pDiCEntry->pVarPtr = &(pCiA402->PosKpDivisor0x3001);
332      break;
333
334  case 0x3002:
335      HMEMCPY((UINT8*)&(pCiA402->PosKiGain0x3002), (UINT8*)&(PosKiGain0x3002), sizeof(PosKiGain0x3002));
336      pCiA402->PosKiGain0x3002 = PosKiGain0x3002;
337      pDiCEntry->pVarPtr = &(pCiA402->PosKiGain0x3002);
338      break;
339
340  case 0x3003:
341      HMEMCPY((UINT8*)&(pCiA402->PosKiDivisor0x3003), (UINT8*)&(PosKiDivisor0x3003), sizeof(PosKiDivisor0x3003));
342      pCiA402->PosKiDivisor0x3003 = PosKiDivisor0x3003;
343      pDiCEntry->pVarPtr = &(pCiA402->PosKiDivisor0x3003);
344      break;

```

Figure 12-3

12-3 Add Handling Functions According to New Object Indexes

Step 1: Add new objects index for write operations as below.

```

ax58100_mcif.c
588  s32 MC_SetParameter(u16 index, u8 subindex, u32 dataSize, u8 *pData)
589  {
590      MCI_Handle_t *pMC = pMCI[M1];
591      PID_Handle_t *pPidPos = (pMC->pPosCtrl->PIDPosRegulator);
592      PID_Handle_t *pPidSpd = pMC->pSTC->PISpeed;
593      PID_Handle_t *pPidIq = pPIDIq[M1];
594
595      switch (index)
596      {
597          /* Set position PID */
598          case 0x3000:
599              memcpy((u8*)&(pPidPos->hKpGain), pData, sizeof(pPidPos->hKpGain));
600              break;
601          case 0x3001:
602              memcpy((u8*)&(pPidPos->hKpDivisor), pData, sizeof(pPidPos->hKpDivisor));
603              break;
604          case 0x3002:
605              memcpy((u8*)&(pPidPos->hKiGain), pData, sizeof(pPidPos->hKiGain));
606              break;

```

Figure 12-4

Step 2: Add new objects index for read operations as below.

```

ax58100_mcif.c
505  s32 MC_GetParameter(u16 index, u8 subindex, u32 dataSize, u8 *pData)
506  {
507      MCI_Handle_t *pMC = pMCI[M1];
508      PID_Handle_t *pPidPos = (pMC->pPosCtrl->PIDPosRegulator);
509      PID_Handle_t *pPidSpd = pMC->pSTC->PISpeed;
510      PID_Handle_t *pPidIq = pPIDIq[M1];
511
512      switch (index)
513      {
514          /* Response position PID */
515          case 0x3000:
516              memcpy(pData, (u8*)&(pPidPos->hKpGain), sizeof(pPidPos->hKpGain));
517              break;
518          case 0x3001:
519              memcpy(pData, (u8*)&(pPidPos->hKpDivisor), sizeof(pPidPos->hKpDivisor));
520              break;
521          case 0x3002:
522              memcpy(pData, (u8*)&(pPidPos->hKiGain), sizeof(pPidPos->hKiGain));
523              break;

```

Figure 12-5

Step 3: Re-build the project and update new firmware into MCU.

12-4 Create Scope Project in TwinCAT

In order to do manual tuning of the control loop gain, we need to observe the position and speed dynamic response, so here must create a scope project to plot some important curves. Please follows the steps below to do this.

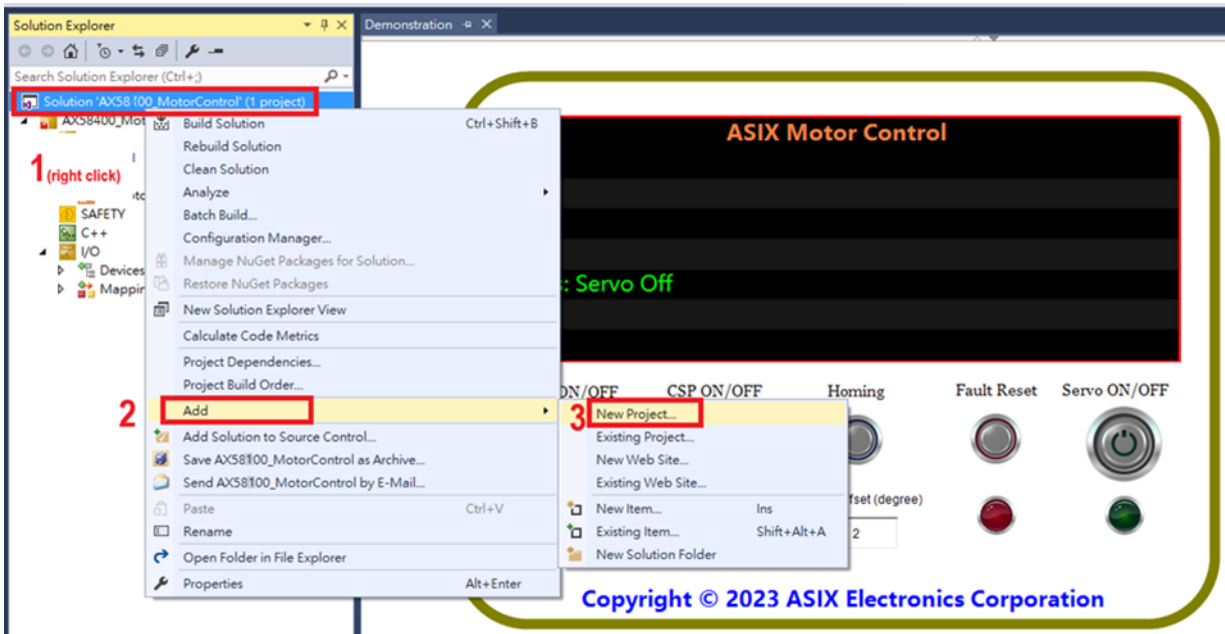


Figure 12-6

Here selects the “**Scope YT Project**” then click “**OK**” button to complete the scope creation.

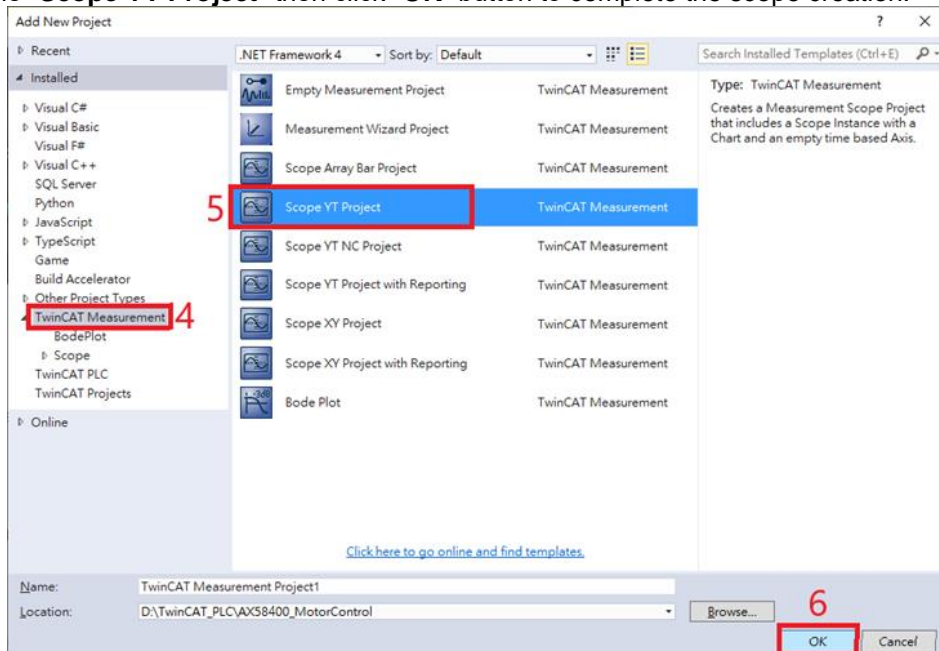


Figure 12-7

Immediately, need to add some process data that we want observing, that includes poTargetPosition, piPositionActualValue, poTargetSpeed and piSpeedActualValue. Please follows the steps below to do this.

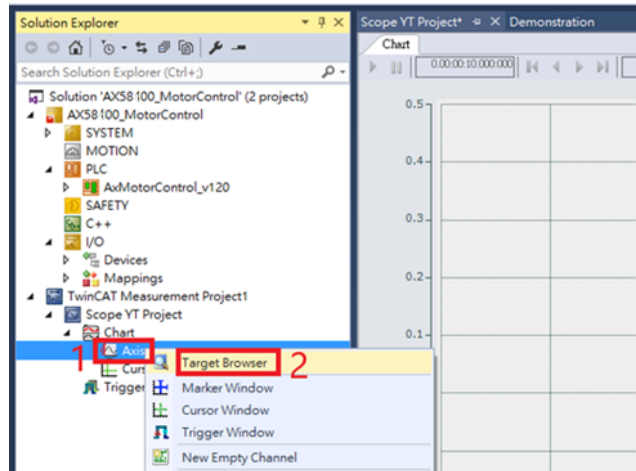


Figure 12-8

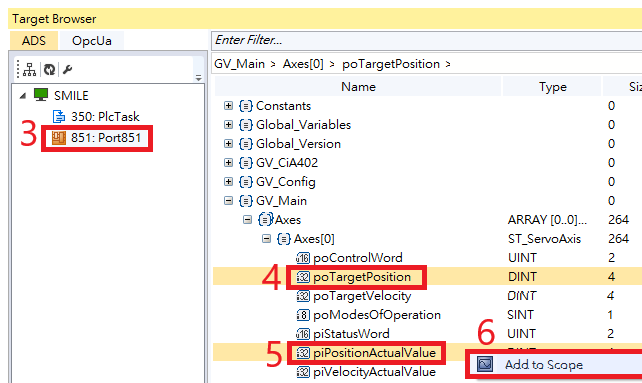


Figure 12-9

Here we create a new plot axis to display speed response curve.

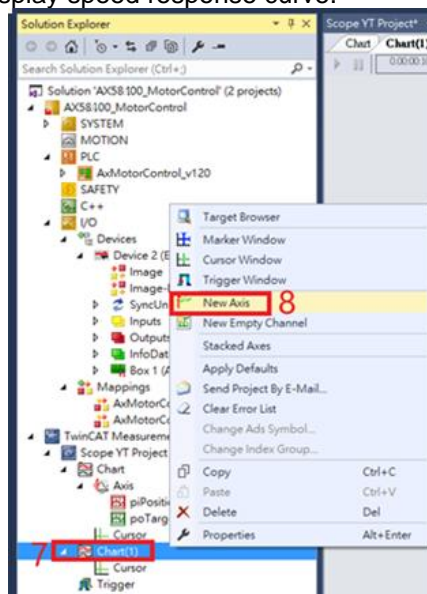


Figure 12-10

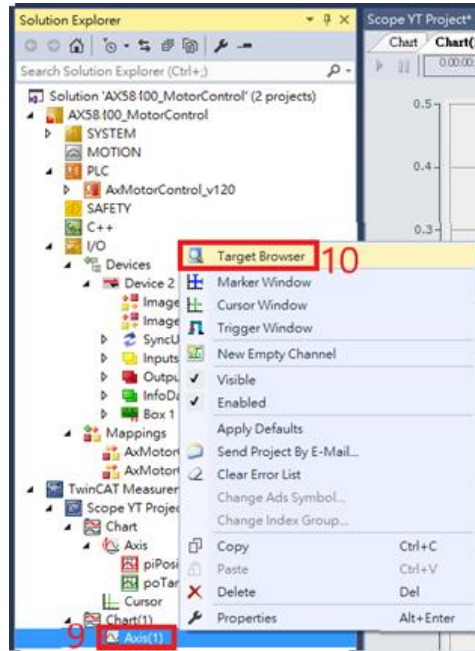


Figure 12-11

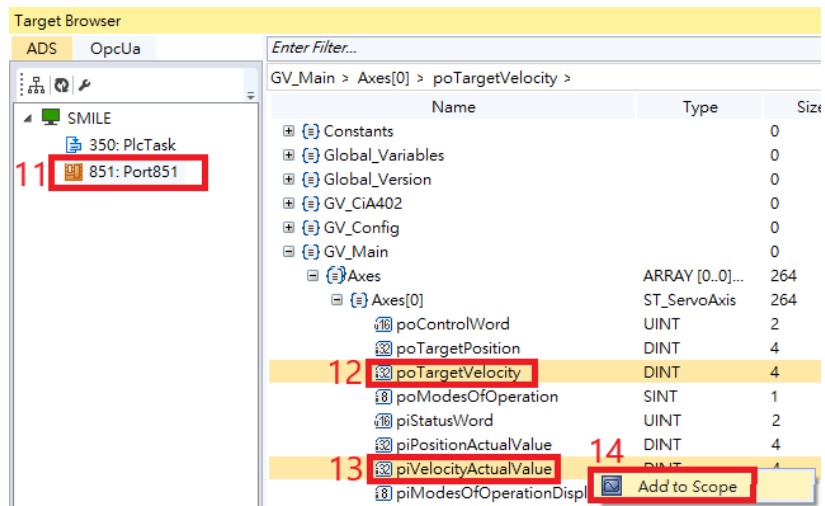


Figure 12-12

So far, the scope is ready to record the process data, we can start to tuning the position and speed loop gains.

12-5 Position Loop Gain Value Manual Tuning in TwinCAT

Let TwinCAT entered running mode, then press “**Servo ON/OFF**” button to enter CiA402 operation state. Turn on “**CSP ON/OFF**” switch to do position loop gain tuning.

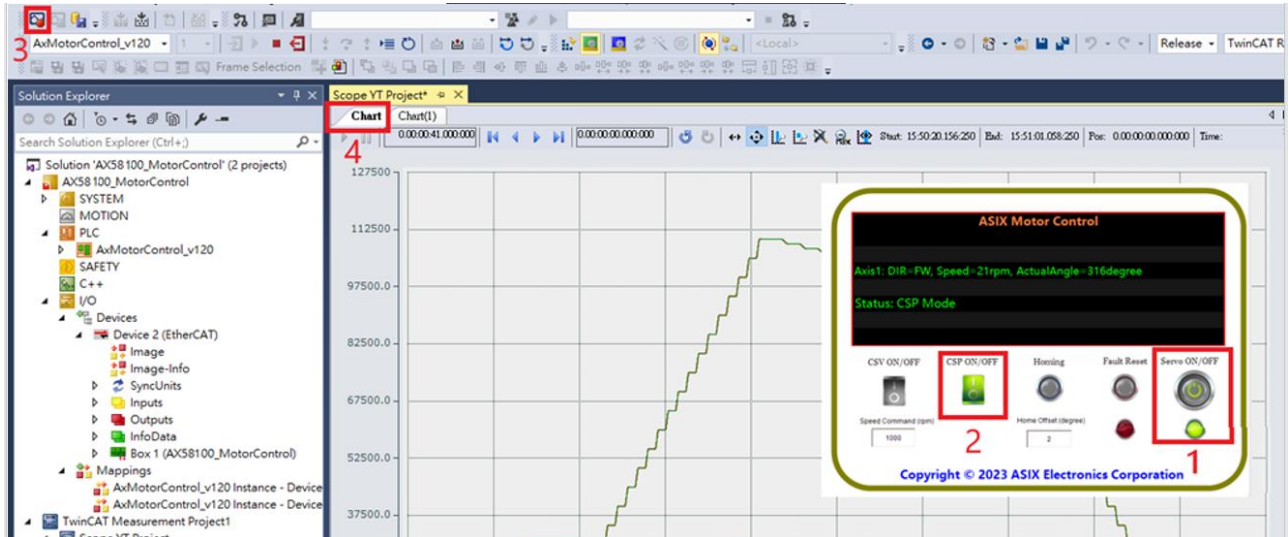


Figure 12-13

If you want to adjust the gain values, please switch back to “**CoE-Online**” function tab as below and write PosKpGain, PosKiGain and PosKdGain via SDO transfer directly.

Please note that, the gain value modification over here is not permanent, if the gain tuning is finalized, please update these gain values in “drive_parameters.h” file, then re-build and upgrade the firmware.

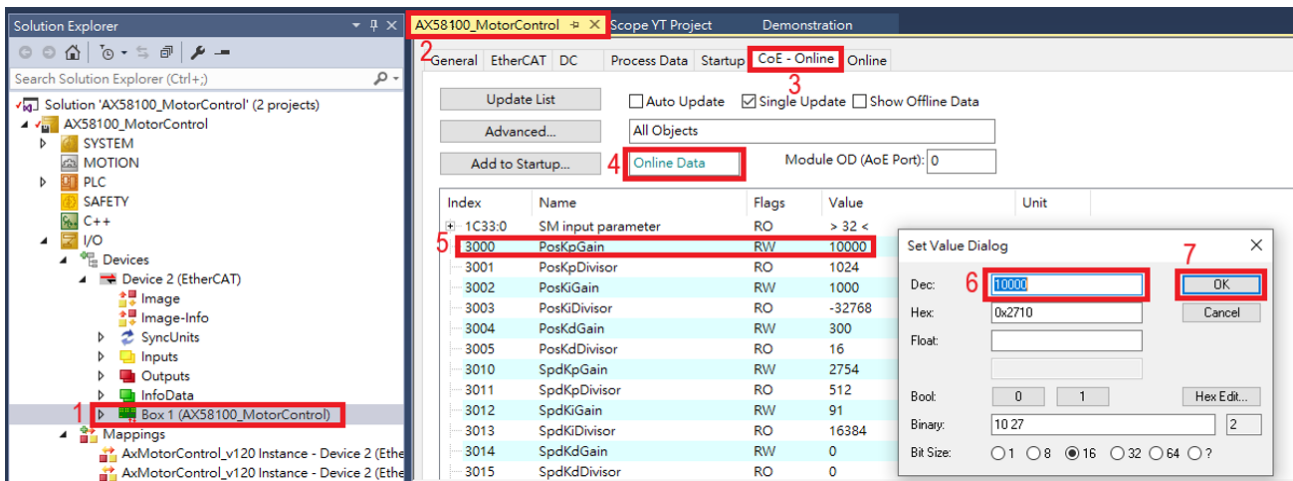


Figure 12-14

If the tuning is complete, please turn off the “**CSP ON/OFF**” switch and wait for the motor stop.

12-6 Speed Loop Gain Value Manual Tuning in TwinCAT

Want to do speed loop gain tuning, please turn on “CSV ON/OFF” switch and change the speed command, then record the curves to observe its overshoot, settling time or following error. Similarly, switch back to “CoE-Online” function tab to adjust SpdKpGain and SpdKiGain until satisfying the application requirement.

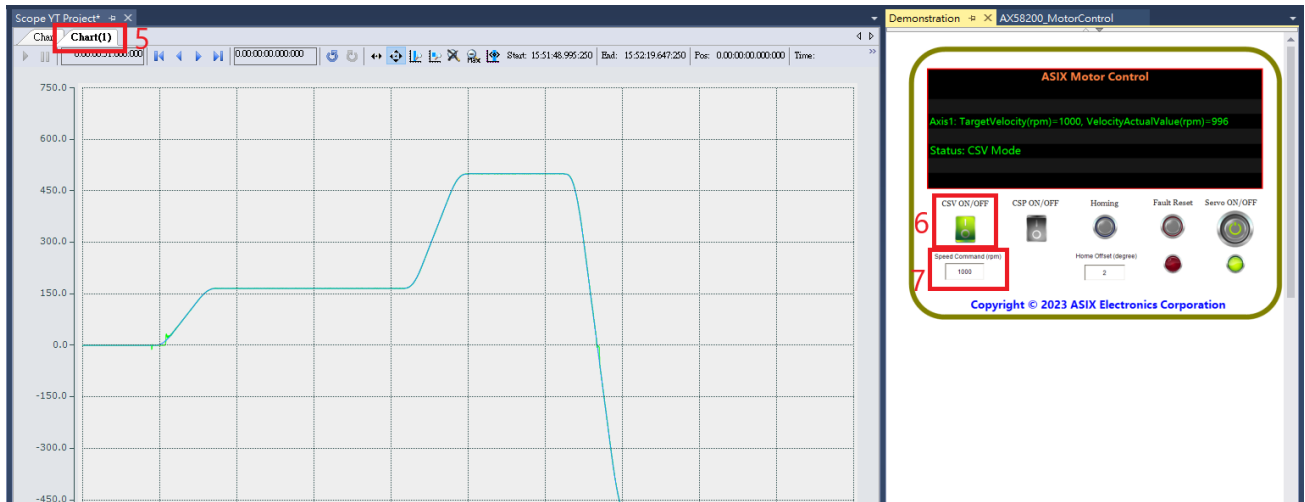


Figure 12-15

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 © 2025 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.



**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>