# ECM-XFU-SK USER GUIDE

版本:V.1.1.2 日期:2023.11



目錄
----

第1章 硬體腳位及使用說明	2
1.1 ECM-XFU-SK V1.3 腳位說明	3
1.1.1 P8/P9 腳位說明	3
1.1.2 J1/J2腳位說明	7
1.1.3 J3腳位說明	7
1.1.4 J7腳位說明(ADC/DAC)	8
1.1.5 J8腳位說明(GPIO)	8
1.1.6 J9腳位說明(ENC)	8
1.1.7 J29 <b>腳位</b> 說明 (Reset)	9
1.1.8 J33腳位說明(SPI)	9
1.2 ECM-XFU-SK 燈號顯示	10
1.3 ECM-XFU-SK V1.3 尺寸圖	10
第 2 章 MCU透過SPI介面範例	12
2.1 STM32範例列表	12
2.2 STM32範例環境建立導引(NUCLEO-F401RE)	12
2.3 實際接線範例	27
2.4 Nuvoton範例列表	29
2.5 Nuvoton新唐範例環境建立導引	30
2.6 Nuvoton修改範例程式為實際應用導引	32
2.7 常見錯誤說明	36
2.8 402自動切換與手動切換方法	36
第 3 章 Beaglebone透過SPI介面範例說明	38
3.1 設置Beaglebone系統	38
3.2 測試範例	39
第 4 章 Windows Visual Stdio設置導引	42
4.1 Visual Stdio C++	42
第 5 章 ECM-XFU ITE整合測試環境說明	45
5.1 ITE簡介	45
5.2 一般操作說明	45
5.3 SDO操作說明	50
5.4 透過Load ENI修改RxPDO及TxPDO配置說明	52
5.5 透過SDO修改RxPDO及TxPDO配置說明	54
5.6 更新韌體 (Update firmware)	55
5.7 SII editor	56
5.8 XFU週邊IO (Peripheral)	57
5.9 評估版 與 授權版	57
5.10 代碼產生工具(Code Generator)	57



# 第1章 硬體腳位及使用說明

### **Digital Input and Output**

- 16 non-isolated input/output channels
- Input Voltage: 0V / 3.3V
- Output Voltage: 0V / 3.3V

### Serial Peripheral Interface(SPI)

- One sets of SPI controllers
- Voltage level: 0V / 3.3V

### EtherCAT

• Data transfer medium: Ethernet cable (CAT5e), shield type: S/STP or S/UTP

- Ethernet interface: 1x RJ-45
- Data transfer rate: 100Mbps
- Protocol: EtherCAT

### **USB** Interface

- USB 2.0 with on-chip transceiver
- Implements USB HID class

### Environment

• Operating temperature: 0°C to 65°C



# 1.1 ECM-XFU-SK V1.3 腳位說明



圖1-1

1.1.1 P8/P9 腳位說明

ECM-XFU-SK與Rasperberry Pi對接, 需依照下圖將P8 / P9 上件, 即可 直接與Rasperberry PI的J8對接, PIN1對接PIN1







ECM-XFU-SK預設R130~R141均有上件, Rasperberry PI透過P8的相關 腳位與ECM-XFU進行SPI的傳輸。

**ECM-XFU-SK**與**Beaglebone**對接, 須先把靠近P8的電阻R131、R130、 R141、R132、R137、R138、R139、R134、R133、R136、R135、R140電阻 移除後(紅色方框), 避免影響BeagleBone本身的pin腳



再將P8/P9上件後,將 ECM-XFU-SK PIN1對接Beaglebone PIN1即可, P8/P9與Beaglebone的P8/P9對接即可。





注意P8/P9排針上件方向(上下)在對接Rasperberry Pi與對接BeagleBone 時不同, 恰相反

ECM-XFU-SK V1.3 P9腳位定義如下表, - 代表空腳:

P9				
PIN45	-	PIN46	-	
PIN43	-	PIN44	-	
PIN41	-	PIN42	-	
PIN39	-	PIN40	-	
PIN37	-	PIN38	-	
PIN35	-	PIN36	-	
PIN33	-	PIN34	-	
PIN31	XF_SPI_CLK	PIN32	-	
PIN29	XF_SPI_MOSI	PIN30	XF_SPI_MISO	
PIN27	-	PIN28	XF_SPI_SS	
PIN25	-	PIN26	-	
PIN23	-	PIN24	-	
PIN21	-	PIN22	-	
PIN19	-	PIN20	-	



PIN17	AM335X_I2C0_SCL PIN18		AM335X_I2C0_SDA
PIN15	-	PIN16	-
PIN13	-	PIN14	-
PIN11	-	PIN12	-
PIN9	-	PIN10	-
PIN7	SYS_5V	PIN8	SYS_5V
PIN5	-	PIN6	-
PIN3	VDD_3V3B	PIN4	VDD_3V3B
PIN1	GND	PIN2	GND

ECM-XFU-SK V1.3的P8腳位定義如下表, 可直接與Rasperberry pi對接, - 代表空腳:

P8(V1.3)				
PIN45	-	PIN46	-	
PIN43	-	PIN44	-	
PIN41	-	PIN42	-	
PIN39	-	PIN40	-	
PIN37	-	PIN38	-	
PIN35	-	PIN36	-	
PIN33	-	PIN34	-	
PIN31	-	PIN32	-	
PIN29	-	PIN30	-	
PIN27	-	PIN28	-	
PIN25	GND	PIN26	-	
PIN23	XF_SPI_CLK	PIN24	XF_SPI_SS	
PIN21	XF_SPI_MISO	PIN22	-	
PIN19	XF_SPI_MOSI	PIN20	GND	



PIN17	-	PIN18	-
PIN15	-	PIN16	-
PIN13	-	PIN14	GND
PIN11	-	PIN12	-
PIN9	GND	PIN10	-
PIN7	-	PIN8	-
PIN5	-	PIN6	GND
PIN3	-	PIN4	(USB_VBUS)
PIN1	VCC33	PIN2	(USB_VBUS)

注意:若要與**Beaglebone**對接, 須先把靠近P8的電阻R131、R130、R141、R132、R137、R138、R139、R134、R133、R136、R135、R140電阻移除後, 方能對接。

1.1.2 J1/J2腳位說明

\*若要由ECM-XFU-SK的USB供電給其它外接板 3.3V, J1的PIN1與PIN2請用跳線帽對 接短路。

\*透過P8/P9對接外接板Beaglebone時,若要由外接板Beaglebone的SYS\_5V供電給 ECM-XFU-SK, **J2**的PIN1與PIN2請用跳線帽對接短路。

J1	PIN2	VDD_3V3B	PIN1	VCC33
J2	PIN2	SYS_5V	PIN1	USB_VBUS

1.1.3 J3腳位說明 (韌體更新)

一般使用模式: J3 PIN2空接或使用跳線帽連接J3 PIN1與J3 PIN2 韌體更新模式:使用跳線帽連接J3 PIN2與J3 PIN3 \*僅於初始上電(或**Reset**)時依據**J3**的狀態進入指定的模式



PIN1	VCC33(Pull-up resistor 4.7K in series)
PIN2	CONFIG/LED
PIN3	GND(Pull-down resistor 4.7K in series)

# 1.1.4 J7腳位說明(ADC/DAC)

J7		
PIN1	5V	
PIN2	ADC_IN	
PIN3	DAC_OUT	
PIN4	DAC_ST	
PIN5	GND	

1.1.5 J8腳位說明(GPIO)

ECM-XFU-SK V1.3 J8				
PIN1	VCC33	PIN2	GND	
PIN3	GPIO14	PIN4	GPIO15	
PIN5	GPIO12	PIN6	GPIO13	
PIN7	GPIO10	GPIO10 PIN8		
PIN9	GPIO8	PIN10	GPIO9	
PIN11	GPIO6	PIN12	GPIO7	
PIN13	GPIO4	PIN14	GPIO5	
PIN15	GPIO2	PIN16	GPIO3	
PIN17	GPIO0	PIN18	GPIO1	



## 1.1.6 J9腳位說明(ENC)

J9		
PIN1	5V	
PIN2	ENCI	
PIN3	ENCA	
PIN4	ENCB	
PIN5	GND	

1.1.7 J29腳位說明 (Reset)

### J29可外接Button, 當Button按下, 即RESETn與GND導通, 系統會RESET

J29	PIN2	GND	PIN1	RESETn
-----	------	-----	------	--------

## 1.1.8 J33 *腳*位說明(SPI)

#### 若要使用ST板(ex:NUCLEO-F401RE)可直接從PIN1開始接 D8,D9,D10,D11,D12,D13,GND,AVDD

J33		
PIN1	INT1	
PIN2	INT0	
PIN3	XF_SPI_SS	
PIN4	XF_SPI_MOSI	
PIN5	XF_SPI_MISO	
PIN6	XF_SPI_CLK	
PIN7	GND	
PIN8	VCC33	



# 1.2 ECM-XFU-SK 燈號顯示

LD1	綠燈	亮	亮尚未有命令 或 CRC錯誤		CRC正確
LD2	紅燈	亮	電源正常	滅	無電源
網口	橘燈	恆亮	網路連線 速度100MHz	滅	無連線
網口	綠燈	亮	連線但無傳輸	閃爍	資料傳輸

# 1.3 ECM-XFU-SK V1.3 尺寸圖



圖1-3



# 第2章 MCU透過SPI介面控制ECM-XF範例

### 2.1 範例架構與主要變數介紹

ECM資料夾存放與ECM-XF相關的程式,其他資料夾則存放與MCU硬體相關的程式。ECM資料夾下ecmxfudrv為應用層API轉換為ECM命令,資料夾includes為相關API的標頭檔(.h File), 資料夾platform為與硬體相關的API,如SPI傳輸、延遲Delay等,example資料夾則為各種不同 應用的主程式檔。

以下介紹主要檔案內容

- ECM\example\ex\_home\main.c 驅動器從站回零點(Homing)範例
- ECM\example\ex\_csp\main.c 驅動器從站週期位置命令範例
- ECM\example\ex\_pp\main.c 驅動器從站目標位置令範例
- ECM\example\ex\_pt\main.c 驅動器從站目標扭力令範例
- ECM\example\ex\_pv\main.c 驅動器從站目標速度令範例
- ECM\example\ex\_rta\main.c Real Time Application, 設定條件比對特定TxPDO內容值
   ,若滿足條件則更改特定RxPDO內容值範例
- ECM\example\utility\utility.c 各範例共用之工具程式
- ECM\example\ex\_XXXX\PdoDefine.h 定義EtherCAT Slave週期性資料傳輸內容( RxPDO、TxPDO)

以下介紹進入OP模式前的設定流程

- 1. ECM\_InitLibrary: 設定SPI Data Size與CRC Type, 回傳IC內版本號
  - => 必要命令, 若未通過(返回值為0), 代表硬體線路未接妥
- 2. ECM\_EcatInit: 初始化從站, 設定週期時間與分散時鐘

=> 必要命令, 必須連結從站且網路孔燈亮起後才能執行, 注意, 設定值必須是從站可以支援的設定值

3. ECM\_EcatSlvCntGet: 取得連線從站數量

=> 非必要命令, 若返回的從站數量與實際不同, 請檢查各站網路接線, 或待全部的網口 Link燈亮起, 再進行ECM\_EcatInit.

4. ConfigDrive: 配置從站的RxPDO及TxPDO (定義於Utility)

=> 視需求及從站特性而定。並非所有從站均能由主站配置RxPDO與TxPDO, 請先參閱從 站說明。

5. ECM\_SetTxFIFOCnt / ECM\_SetRxFIFOCnt: 設定RxPDOFIFO/TxPDOFIFO數量

=> 非必要命令, RxPDOFIFO/TxPDOFIFO數量預設均為64筆, 可透過此命令變更數量 6. ECM\_CheckMEMSpace: 檢查設定值記憶體空間是否足夠 (定義於Utility)

=> 非必要但建議命令, 檢查1) SPI Data Size是否則足夠放下所有RxPDO / TxPDO. 2) 記 憶體空間是否滿足TxFIFO及RxFIFO的空間需求

7. ECM\_InitFIFO: 初始化FIFO



=> 必要命令, 初始化FIFO前請確認各設定值沒有超過記憶體容量

8. ECM EnableFIFO: 開啟或關閉FIFO

=> 非必要命令, 初始化FIFO後預設開啟FIFO功能, 當進入OP時就會開始透過FIFO收發資料, 可透過此命令關閉FIFO, 待進入有需求時再開啟

9.ECM\_FifoRxPdoSizeGet/ECM\_FifoTxPdoSizeGet: 取得RxPdoSize / TxPdoSize

=> 非必要但建議命令, 取得所有從站的RxPDO / TxPDO大小, 可比對是否與程式中傳送/ 接收的資料結構大小相符

10. SetDriverCmdOpMode: 設定驅動器的操作模式 (定義於Utility)

=> 驅動器專用命令, 非必要命令但建議, 設定驅動器的操作模式(0x6060), 驅動器所支援 的操作模式請參考驅動器手冊說明

11. SetStateAndCheck: 切換etherCAT模式(Init / perOP / safeOP / OP) (定義於Utility) => 切換etherCAT模式有特定的順序, 請參閱etherCAT技術手冊

12. ECM\_CheckDCStable: 檢查EtherCAT的分散時鐘(DC)是否已經穩定

=> 非必要命令, 當etherCAT模式由PerOP切換至SafeOP時, 若有啟用分散時鐘(DC)機制, 則分散時間會開始校正同步時間, 透過此命令可確認分散時鐘校正是否已經完成

以下介紹進入OP模式後的主要命令

- 1. AlignmentAndWaitServoOn: 將目標位置指定為目前位置, 並激磁 (定義於Utility) => 非必要命令, 當操作模式為CSP時, 需先將目標位置指定為目前位置, 再激磁
- 2. ECM\_EcatPdoFifoDataExchangeAdv: 透過FIFO進行PDO的交換
   => 必要命令,進入OP後,透過此命令交換FIFO中的PDO資料



# 2.2 STM32範例環境建立導引(NUCLEO-F401RE)

- 1. 開啟STM32Cube IDE.
- 2. 在左側視窗點擊滑鼠右鍵並選擇"STM32 Project"

Interpretation of the second secon	Project Run Win	dow	Help	
<b>□</b> •	* 2. ©.e  ₹	R		• 💁 • 🎒 🛷 • [
🎦 Project Explorer 👷 📄 🛱 💙 🗖 🗖				
> 00 117 > 01 118 > 01 118 > 01 118 > 01 120 > 01 120 > 01 122 > 01 122 ■ STM32SPLECMXF_GPIO				
New 🚵 Import	>	<b>1</b>	Project C/C++ Project	
🗾 Export		IDE	STM32 Project	
Refresh	F5	MX	STM32 Project From STM32CubeMX .ioc File	•
🕼 Generate Code			Other	Ctrl+N
	Problems &	) Tas	ks	)
< >				

3. 選擇所使用的板子。這邊例子以STM32 NUCLEO-F401RE為例





4. 選擇完成後點選下一步



5. 輸入專案名稱與專案位置

💴 STM32 Proje	ct			×
🐴 Project locati	on already exists		I	DE
Project				
Project Name:	123			
Use default	location			
Location:	E:\FireFox Download\123		Brows	e
● C ○ C- Targeted B ● Executal Targeted P ● STM32C	inary Type ble O Static Library roject Type ube O Empty			
?	< Back Next > Finis	h	Cance	1



#### 6. 選擇韌體並點選完成。如果沒有目前所選擇的版本,系統會自動下載所選擇的版本

Firmware Library Package Setup         Setup STM32 target's firmware         Image: Target and Firmware Package         Target Reference:       NUCLEO-F401RE         Firmware Package Name and Version:       STM32Cube FW_F4         V1.25.1          Firmware package Repository         Location:         C:\Users\dttb1\STM32Cube\Repository         See 'Firmware Updater' for settings related to firmware package installation         Code Generator Options         Add necessary library files as reference in the toolchain project configuration file         Copy all used libraries into the project folder         © Copy only the necessary library files	DE STM32 Project -	$\times$
Setup STM32 target's firmware         Target and Firmware Package         Target Reference:       NUCLEO-F401RE         Firmware Package Name and Version:       STM32Cube FW_F4         V1.25.1 ~         Firmware package Repository         Location:         C:\Users\dttb1\STM32Cube\Repository         See 'Firmware Updater' for settings related to firmware package installation         Code Generator Options         Add necessary library files as reference in the toolchain project configuration file         Copy all used libraries into the project folder            © Copy only the necessary library files	Firmware Library Package Setup	IDE
Target and Firmware Package         Target Reference:       NUCLEO-F401RE         Firmware Package Name and Version:       STM32Cube FW_F4         V1.25.1          Firmware package Repository       Location:         C:\Users\dttb1\STM32Cube\Repository         See 'Firmware Updater' for settings related to firmware package installation         Code Generator Options         O Add necessary library files as reference in the toolchain project configuration file         Copy all used libraries into the project folder            © Copy only the necessary library files	Setup STM32 target's firmware	IUL
Target Reference:       NUCLEO-F401RE         Firmware Package Name and Version:       STM32Cube FW_F4         V1.25.1          Firmware package Repository       Location:         C:\Users\dttb1\STM32Cube\Repository       See 'Firmware Updater' for settings related to firmware package installation         Code Generator Options       Add necessary library files as reference in the toolchain project configuration file         Copy all used libraries into the project folder       © Copy only the necessary library files	Target and Firmware Package	
Firmware Package Name and Version: STM32Cube FW_F4       V1.25.1         Firmware package Repository       Location:         C:\Users\dttb1\STM32Cube\Repository       See 'Firmware Updater' for settings related to firmware package installation         Code Generator Options       O Add necessary library files as reference in the toolchain project configuration file         Copy all used libraries into the project folder       Ibraries into the project folder	Target Reference: NUCLEO-F401RE	
Firmware package Repository Location: C:\Users\dttb1\STM32Cube\Repository See <u>'Firmware Updater'</u> for settings related to firmware package installation Code Generator Options O Add necessary library files as reference in the toolchain project configuration file Copy all used libraries into the project folder © Copy only the necessary library files	Firmware Package Name and Version: STM32Cube FW_F4 V1.25.1 V	
Location: C:\Users\dttb1\STM32Cube\Repository See <u>'Firmware Updater'</u> for settings related to firmware package installation Code Generator Options O Add necessary library files as reference in the toolchain project configuration file Copy all used libraries into the project folder @ Copy only the necessary library files	Firmware package Repository	
C:\Users\dttb1\STM32Cube\Repository See <u>'Firmware Updater'</u> for settings related to firmware package installation Code Generator Options O Add necessary library files as reference in the toolchain project configuration file Copy all used libraries into the project folder © Copy only the necessary library files	Location:	
See <u>'Firmware Updater'</u> for settings related to firmware package installation Code Generator Options O Add necessary library files as reference in the toolchain project configuration file Copy all used libraries into the project folder © Copy only the necessary library files	C:\Users\dttb1\STM32Cube\Repository	
Code Generator Options O Add necessary library files as reference in the toolchain project configuration file O Copy all used libraries into the project folder © Copy only the necessary library files	See <u>'Firmware Updater'</u> for settings related to firmware package installation	1
<ul> <li>Add necessary library files as reference in the toolchain project configuration file</li> <li>Copy all used libraries into the project folder</li> <li>Copy only the necessary library files</li> </ul>	Code Converter Ontions	
<ul> <li>Copy all used libraries into the project folder</li> <li>Copy only the necessary library files</li> </ul>	• Add necessary library files as reference in the toolchain project configurat	ion file
<ul> <li>Copy only the necessary library files</li> </ul>	Copy all used libraries into the project folder	
	Copy only the necessary library files	
? < Back Next > Finish Cancel	Carter	ncel

#### 7. 視窗詢問初始化所有外部裝置至預設模式時選擇"是"

DE STM32 Project						$\times$
Firmware Library	Package Setup					DE
Setup STM32 targ	get's firmware					DE
- Target and Firm	ware Package					
Target Reference	в;	NUCLEO-F4	401RE			
Firmware Packag	je Name and Versio	on: STM32Cub	e FW_F4 V1.25	.1 ~		
📧 Board Project Op	otions:					×
<b>(</b> )	peripricities martin					
			Yes		No	I
Code Generator	Options		Yes		No	
Code Generator Add necessar	<b>Options</b> y library files as refe	rence in the too	Yes	configu	No uration	file
Code Generator	<b>Options</b> y library files as refe libraries into the pr	rence in the too oject folder	Yes	configu	No uration	file
Code Generator Add necessar Copy all used Copy only the	<b>Options</b> y library files as refe libraries into the pr e necessary library fi	rence in the too oject folder les	Yes	configu	No	file
Code Generator Add necessar Copy all used Copy only the Perform Project Cre	Options y library files as refe libraries into the pr necessary library fi eation. Please Wait	rence in the too oject folder les For Completior	Yes	configu	No	file



#### 8. 詢問開啟STM32CubeMx時選擇"是"並開始定義腳位

DE STM32 Project					×
Firmware Library	Package Setup				DE
Setup STM32 targ	et's firmware				DE
Torget and Eirmu	uara Daskaga				
Target and Firms	иаге Раскаде		401 PE		
rarget Kererence	5. 	NUCLEO-P	4016		_
Eirmware Packad Cpen Associated	e Name and Vers Perspective?	ion: STM32Cub	be FW_F4_1V1.25.1	$\sim$	×
This kind of want to ope	project is associa en this perspectiv	ated with the STI e now?	M32CubeMx pers	pective. De	o you
🗌 Remember my d	lecision				
			Yes	N	D
- Code Generator	Ontions				
Add necessary	librany filas as ra	ference in the to	olchain project co	oficiaria	file
Convallused	librarias into the	araiect folder	forenann project ce	Ingulation	THIC .
Copy an used	indraries into the	Project loider			
Copy only the	necessary library	Tiles			
Perform Project Cre	ation Please Wa	it For Completio	'n		
		it for completie			
?	< Back	Next >	Finish	Canc	el



9. 在腳位定義圖中, 選擇PA5為SPI1\_SCK、PA6為SPI1\_MISO、PA7為SPI1\_MOSI與 PB6為GPIO\_Output





## 10. 點選左側"System Core"對GPIO進行進階設定, 將"GPIO output level"調整至"High"

Q ~	0	GPIO Mode and Configuration						
Categories A->Z	_			Config	guration			
System Core 🗸 🗸	Group	By Periphe	rals					~
DMA	○	⊘ SYS GPIO		⊘ U ⊘ Single N	SART Napped Sig	nals	⊘ NVIC	C RCC
IWDG NVIC VRCC	Search Search	Signals				□ Sho	ow only Mod	ified Pins
WWDG	Pin PB6 PC13-A.	Signal o n/a n/a	GPIO o High n/a	. GPIO m. Output Externa	. GPIO P No pull No pull	Maximu Low n/a	B1 [Blu	Modified ✓ ✓
Analog >								
Timers >								
Connectivity >	PB6 Co	onfiguration	:					
Multimedia >	GPIO o	output level		F	ligh			~
Computing >	_ GPIO r	node		C	Output Pusł	n Pull		~
Middleware >	GPIO F	oull-up/Pull-	-down	1	lo pull-up a	nd no pull-	down	~
	Maxim	um output s	speed	L	.0W			~
	User La	abel						



11. 點選"Timers"中的TIM2並將"Clock Source"改為"Internal Clock"後, 將下方"Counter Period"數值改為"0xFFFFFFF"

Q	~ 🔕	TIM2 Mode and Configuration			
Categories A->2	Z	Mode			
System Core	>	Slave Mode Disable $\checkmark$			
		Trigger Source Disable 🗸 🗸			
Analog		Clock Source Internal Clock ~			
Timers	~	Channel1 Disable ~			
\$		Channel2 Disable ~			
RTC TIM1		Channel3 Disable ~			
A TIM2		Channel4 Disable ~			
TIM3		Combined Channels Disable ~			
▲ TIM5		Use ETR as Clearing Source			
▲ TIM9		XOR activation			
TIM10		Configuration			
		Reset Configuration			
Connectivity	>	🥺 User Constants 🛛 📀 NVIC Settings 📀 DMA Settings			
		Parameter Settings			
Multimedia	>	Configure the below parameters :			
Computing	>	Q Search (CrtI+F) ③ ③			
		✓ Counter Settings			
Middleware	>	Prescaler (PSC - 16 bits val 0			
		Counter Period (AutoReload 0xfffffff			
		Internal Clock Division (CKD) No Division			
		auto-reload preload Disable			



12. 接下來跳至"Connectivity"設定"SPI1", 模式設定"Full-Duplex Master"並將鮑率設置在 2~8之間

V	Ø	SPI1 Mode and Configuration		
ategories A->Z		Mode		
System Core	>	Mode Full-Duplex Master 🗸 🗸		
Analog	>	Hardware NSS Signal Disable ~		
Timers	>			
Connectivity	~			
\$				
I2C1				
Ø 12C2				
I2C3				
SDIO	_			
V SPI1				
SPIZ		Configuration		
		Reset Configuration		
USART2				
USART6		OMA Settings OPIO Settings		
USB OTG FS		Parameter Settings Vic Settings		
000_000_00		Frame Format Motorola		
		Data Size 8 Bits		
Multimedia	>	First Bit MSB First		
mannoula		<ul> <li>Clock Parameters</li> </ul>		
Computing	>	Prescaler (for Baud Rate) 8		
		Clock Polarity (CPOL) Low		
Middleware	>	Clock Phase (CPHA) 1 Edge		
		✓ Advanced Parameters		
		CRC Calculation Disabled		

13. 完成上面步驟後, 點擊儲存標示儲存並自動產生程式



DE Question	×	
Do you want generate Code?		
Remember my decision		
	Yes No	



14. 如果在生成的main.c "Private variables"中看到剛設定的SPI、TIM2表示有之前的設定 有完成

```
/* Private variables ------*/
   SPI HandleTypeDef hspi1;
   TIM HandleTypeDef htim2;
   UART HandleTypeDef huart2;
15. 點開"main.h"並依下圖加入下方程式碼
   /* USER CODE BEGIN Includes */
   #include "platform.h"
#include "EcmUsrDriver.h"
   /* USER CODE END Includes */
   /* Exported macro -----*/
   /* USER CODE BEGIN EM */
   #ifndef PRINTF
   #define PRINTF( str, ...) \
       do{ \
          int n; \
          n = sprintf( printbuf, (str), ## VA ARGS ); \
          HAL UART Transmit( &huart2, (uint8 t *)printbuf, n, 0xffffffff); \
       }while(0)
   #endif
   #ifndef GETCHAR
   #define GETCHAR userGetchar
   #endif
   /* USER CODE END EM */
   /* USER CODE BEGIN EFP */
   extern UART HandleTypeDef huart2;
   extern char printbuf[];
   /* USER CODE END EFP *
16. 開啟"main.c"並加入下方程式碼
    /* USER CODE BEGIN PFP */
   char printbuf[128];
   int main ini(void);
   /* USER CODE END PFP */
   /* USER CODE BEGIN 2 */
   main ini();
   /* USER CODE END 2 */
   /* USER CODE BEGIN TIM2 Init 2 */
   HAL TIM Base Start(&htim2);
   /* USER CODE END TIM2 Init 2 */
```

17. 複製EcmDriver.h、EcmUsrDriver.h、PdoDefine.h、platform.h與Utility.h(部分範例)貼入 "Inc"資料夾



- 18. 複製crc32.c、EcmUsrDriver.c、main\_ini.c(主程式:部分範例為不同名)與platform.c、 Utility.c(部分範例)貼入Src資料夾
- 19. 已準備好執行範例程式

#### 20. 點擊槌子圖示, 並選取欲執行的程式來進行Build, 或是點擊Project底下的Build all來 建置程式

workspace\_1.9.0 - STM/Core/ECM/example/ex\_csp/PdoDefine.h - STM32CubeIDE <u>File Edit Source Refactor Navigate Search Project Run Window H</u>elp | ☆ + 図 | ◎ | ◎ + 冬 + 副 | ▲ | ■ Ⅲ ■ № 2. ۞ ...( 弓 寒 | ◎ | ☆ + 않 + ピ + ♂ + ☆ + ○ + 冬 + | ⊿ ◎ 回 π | ⊴ + බ + ♡ -♡ -> + | ♂ | ◎ | □ 
 Project Explorer ×
 1 Debug

 ✓ INI STM
 2 Release

 > ∰ Binaries
 3 ex\_csp.debug

 > ∭ Includes
 4 exect
 E 😨 🎖 🖁 🗖 🗈 🖻 main.c 🕼 EcmUsrDriver.h 🕼 PdoDefine.h 🕅 PdoDefine.h 🗙 1⊖/ 2 \* File : PdoDefine.h 3 \* Version : 1.52 4 \* Date : 2021/11/23 5 \* Author : XFORCE > 🔊 Includes 4 ex\_csp\_release Core
 ✓ 5 ex\_home\_debug ✓ <sup>™</sup> Core > > cmxfuc 6 ex\_home\_release > > cmxfuc 7 ex\_pp\_debug \* Define slave device PDO data structure > 🗁 ex\_hc > 🗁 utility You can configure the PDO structure you wanted,
 or you can just use the default PDO structure.
 No matter what, the example program need to know. 8 ex\_pp\_release 10 9 ex\_pt\_debug ✓ Ø ex\_cs Ø me Ø Me
Me 10 ex\_pt\_release 13 14 11 ex\_pv\_debug \* @copyright (C) 2020 NEXTW TECHNOLOGY CO., LTD.. All rights reserved. 15 16 12 ex\_pv\_release > 💋 ex\_pr > 💋 ex\_pt > 💋 ex\_pv \*\*\*\*\* 13 ex\_rta\_debug #ifndef \_ECM\_USR\_PD0\_DEFINE\_H\_
#define \_ECM\_USR\_PD0\_DEFINE\_H\_ 18 14 ex\_rta\_release 19 > 🇭 ex rta 20 21 22 23 24 #define TEST\_DRV\_CNT #define TEST\_IO\_CNT #define COUPLER\_CNT #define TEST\_SLV\_CNT > 👝 includes > 👝 platform > 👝 Inc > 👝 Src (TEST\_DRV\_CNT+COUPLER\_CNT+TEST\_IO\_CNT) 24 25 26 27 28 29 #define TEST\_CTL\_WORD\_OFFSET 0 #define TEST\_STA\_WORD\_OFFSET 0 👝 Startup > 😕 Drivers > 👝 ex\_csp\_debug #define TEST\_MODEOP 8//CSP\_Mode > 👝 ex\_csp\_release #if (TEST\_MODEOP == 4)
#define TARGET\_TYPE intl6\_t
#elif (TEST\_MODEOP == 10)
#define TARGET\_TYPE intl6\_t > 🧀 ex\_home\_debug > 🗁 ex\_pp\_debug 31 > 👝 ex\_pt\_debug 34 👝 ex\_pv\_debug 35 36 37 #define TARGET\_TYPE int32\_t #endif ex\_pt\_debug 38 39⊖ typedef struct \_attribute\_((\_packed\_)) \_axis\_rxpdo\_st\_def\_t 40 { 41 uint16\_t ui6CtlWord; // 0x6x40 - Control word 42 TARGET\_TYPE Target; // 0x6x7A - Target position 43 // 0x6xFF - Tro STM32F401RETX\_FLASH.Id STM32F401RETX\_RAM.Id uint16\_t u16Ct1Word; // 0x6x40 - Control word TARGET\_TYPE Target; // 0x6x7A - Target position // 0x6x7A - Target velocity // 0x6x7T - Target velocity STM ex csp debug.launch STM ex\_home\_debug.launch STM ex pt debug.launch STM ex\_pv\_debug.launch STM ex\_pv\_debug.launch STM ex\_rta\_debug.launch TestSTM32 🛃 🗉 🕶 📑 🗖 👔 Problems 🧔 Tasks 📮 Console 🗙 🔲 Properties No consoles to display at this time.



workspace\_1.9.0 - STM/Core/ECM/example/ex\_csp/PdoDefine.h - STM32CubeIDE

File Edit Source Refactor Navigate Search	Project Run Window H	Help
	Open Project	
Project Explorer ×	Close Project	: 🗈 EcmUsrDriver.h 🗈 PdoDefine.h 👍 PdoDefine.h 🗙
Project Explorer × Project Explorer × STM Straines Core	Close Project Build All Build Configurations Build Project Build Working Set Clean Build Automatically Build Targets C/C++ Index Generate Report Generate Code Properties	<pre>Ctrl+B Build All Build All File : PdoDefine.h PdoDefine.h × File : PdoDefine.h Version : 1.52 Date : 2021/11/23 Author : XFORCE Define slave device PDO data structure Vou can configure the PDO structure you wanted, or you can just use the default PDO structure. No matter what, the example program need to know. % copyright (C) 2020 NEXTW TECHNOLOGY CO., LTD. All rights reserved. % file rEST_DRV_CNT 2 #define TEST_OR_USR_PDO_DEFINE_H_ define TEST_DRV_CNT 2 #define TEST_ON_CNT 2 #define TEST_SLV_CNT 0 #define TEST_SLV_CN</pre>
STM ex_pv_debug.launch STM ex_rta_debug.launch		Problems @ Tasks  Console ×  Properties
🔲 TestSTM32		No consoles to displey at this time.

#### 點擊後會開始建置build

#### 在Console欄中顯示完成訊息 (若有錯誤請檢查錯誤原因並排除,注意程式路徑不可包含非英文語系)

🖹 Problems 🧟 Tasks 📮 Console 🗙 🔲 Properties 🛛 💥 🐥 🏠 🚰 🚮 🚝 🚉 🗐 🛃 🖉 🖛 😭 🕶	
CDT Build Console [STM]	
10:23:25 **** Incremental Build of configuration ex_home_debug for project STM ****	$\sim$
make -j8 all	
arm-none-eabi-size STM.elf	
text data bss dec hex filename	
37336 140 6396 43872 ab60 STM.elf	
Finished building: default.size.stdout	
10,00,00 Build Sinished Deserve Deservices (tech 378ms)	
10:25:25 Build Finished. 0 errors, 0 warnings. (took 578ms)	
	$\sim$
<	>



21.下一步要進行Debug,在上方工具列中點擊蟲子的圖示,並點選Debug Configurations.

workspace_1.9.0 - STM/Core/ECM/example/ex_csp/PdoDefine.h - STM32CubeID	Ε	
File Edit Source Refactor Navigate Search Project Run Window Hel	) (삶 * 63 * 67 * 69 * i 56 * 10 * 94 * i 59 - 세 10 등 등 i 성 * 집 * 10 다 다 다 이 * i (* i 1)	
C       C	<pre>@ main: @ Ecnuls: 15TM ex.cp.debug between the second of the second</pre>	
STM32F401ETX_FLASHId     STM32F401ETX_FAMId     STM ex_cps_debug.launch     STM ex_pp_debug.launch     STM ex_pp_debug.launch     STM ex_pp_debug.launch     STM ex_pp_debug.launch     STM ex_ps_debug.launch     STM ex_ps_debug.launch     STM ex_tp_debug.launch     TetSTM32	38       Jask         390       typedef struct _attribute_((_packed_)) _axis_rxpdo_st_def_t         40       (intl6_t       ui6CtlWord; // @xdsx40 - Control word         42       TARGET_TYPE       Target; // @xdsx74 - Target poilion         44       // @xdsx74 - Target poilion       // @xdsx71 - Target torque         44       // @xdsx71 - Target torque       // @xdsx71 - Target torque         45       COTBuild Console (STM)       10:23:25 **** Incremental Build of configuration ex_home_debug for project STM **** marks - j8 all         37336       54:40       59:60       570.elf         rinske - j8 all       571.elf         rinske duilding: default.size.stdout       10:23:25 Build Finished. 0 errors, 0 warnings. (took 378ms)	> ug ×

在Debug Configurations中點選STM32, 並選擇要Debug的程式再按下Debug就會開始偵錯

Create, manage, and run configurations	Debug Configurations					
Image: Style and Style a	Create, manage, and run configurations					The state
	Image: Second	Name: STM ex_home_de Main 参调试器 Project: STM C/C++ Application: ex_home_debug/STM.e Build (if required) befor Build Configuration: S O Enable auto build ④ Use workspace settin	sbug Startup Source Comm If re launching Select Automatically	O Disable auto build Configure Workspace Settings	Search Project	Browse
Kevert     Apply	< > Filter matched 13 of 13 items				Revert	Apply
(?) Debug Close	?				Debug	Close



#### 程式會下載至MCU中並在Console顯示相關訊息

	Problems	🧟 Tasks	📮 Console 🗙	Properties		ж	*	k 🚮	N	P	<b>*</b>	1	<b>-</b>	- 📑	• -	
ST	M ex_home	_debug [S1	M32 Cortex-M	C/C++ Applicatio	n]											
																~
Ve	rifving .															
Do	wnload ve	rified su	ccessfully													
		. 1.1200 . 50														
																4
<															3	>

#### 22. 開啟序列通訊埠程式 (如Putty), 並在Connection type中勾選Serial

Category:       Category:	🕵 PuTTY Configuration	? >	👷 PuTTY Configuration		? ×
Basic options for your PuTTY session       Basic options for your PuTTY session         Specify the destination you want to connect to       Logging         Keyboard       Host Name (or IP address)       Port         Keyboard       22         Window       SSH       Serial       Other:         Translation       Save       Selection         Connection       Default Settings       Load         Proxy       SSH       Default Settings       Load         Proxy       SSH       Default Settings       Load         SSH       Default Settings       Load       Save         SSH       Default Settings       Default Settings       Load         SSH       Default Settings       Default Settings       Default Settings         COM6       Save       SSH       SSH	Category:		Category:		
Serial Telnet Rlogin SUPDUP Close window on exit: O Always O Never O Only on clean exit	Session     Logging     Terminal     Keyboard     Bell     Features     Window     Appearance     Behaviour     Translation     Selection     Colours     Connection     Pota     Proxy     SSH     Serial     Telnet     Rlogin     SUPDUP	Basic options for your PuTTY session         Specify the destination you want to connect to         Host Name (or IP address)       Port         [       22         Connection type:       22         © SSH       Serial       Other:         Load, save or delete a stored session       Saved Sessions         Default Settings       Load         COM6       Save         Delete         Close window on exit: <ul> <li>Always</li> <li>Never</li> <li>Only on clean exit</li> </ul>	Session     Logging     Terminal     Keyboard     Bel     Features     Window     Appearance     Behaviour     Translation     Selection     Colours     Connection     Data     Proxy     SSH     Serial     Teinet     Rogin     SUPDUP	Basic options for your PuTTY sessic         Specify the destination you want to connect the serial line         Serial line         COM11         Connection type:         SSH         Serial         Other:         Telnet         Load, save or delete a stored session         Saved Sessions         Default Settings         COM6         Close window on exit:         Always         Never         Only on clear	on to peed 3600  Load Save Delete n exit

#### 在Serial line中輸入對應到的COM port(可以在裝置管理員中的連接埠確認,本例為COM4)

過 装置管理員
福赛(F) 動作(A) 檢視(V) 說明(H)
← ⇒   □□   □   □   □   □   □   □   □   □
V 🛃 DESKTOP-JEVTG1T
> 📹 IDE ATA/ATAPI 控制器
> 🗛 人性化介面裝置
> 📃 可描式装置
> 🖻 列印佇列
> 🎥 存放控制器
> 🔐 安全性装置
> 🌄 系統裝置
> 🖬 音效、視訊及遊戲控制器
> 🔲 處理器
> 📑 軟體元件
> 2 軟體裝置
> 🏺 通用序列匯流排控制器
> 🎍 通用序列匯流排裝置
✓ 単 連接場 (COM 和 LPT)
STMicroelectronics STLink Virtual COM Port (COM4)
₩ 通訊連接埠 (COM1)
Def area



設定好Serial line後,	speed輸入	115200,	接著就可以點擊Open
------------------	---------	---------	-------------

🔀 PuTTY Configuration	? ×				
Category:					
	Basic options for your PuTTY session				
Logging	Specify the destination you want to connect to				
<ul> <li>Terminal</li> <li>Keyboard</li> <li>Bell</li> <li>Features</li> <li>Window</li> <li>Appearance</li> <li>Behaviour</li> <li>Translation</li> <li>Selection</li> <li>Colours</li> <li>Connection</li> <li>Data</li> <li>Proxy</li> <li>SSH</li> <li>Serial</li> <li>Telnet</li> <li>Rlogin</li> <li>SUPDUP</li> </ul>	Serial line Speed				
	COM4 115200				
	Connection type:				
	◯ SSH				
	Load, save or delete a stored session				
	Saved Sessions				
	Default Settings Load				
	Save				
	Delete				
	Close window on exit:				
	○ Always ○ Never				
About Help	Open Cancel				

#### 23.進入到Putty的畫面





#### 按下STM F401RE版上的reset鍵出現畫面上的字樣(或點選程IDE中執行按鈕)

B COM4 - PuTTY	_	$\times$
Enter any key after ECM link done		

確認ECM-XFU-SK已連結至從站,且網路口燈亮起,按下任意鍵就會開始執行程式



有出現版本號表示成功代表SPI連線傳輸正常

若出現failed訊息可能為接觸不良,調整版子接合處並按下ECM-XF-SK上的reset鍵重新執行



# 2.3 實際接線範例

STM32 F401RE 腳位接線





STM32 H7 腳位接線 STM32所生成之main.c與main.h不與F4版本共用,請使用H7範例為範本修改 請使用H7專用platform.h與platform.c 需要F4版本範例可以直接複製main\_ini.c覆蓋至H7 main\_ini.c





# 2.4 Nuvoton範例列表

以下範例為使用新唐M487與Nueclipse IDE進行編程

- NuDAC\_ADC
   ECM-XF晶片上DAC與ADC功能範例
- NuDrive\_IO
   一個驅動器馬達從站與一個IO從站之範例
- NuDrive 驅動器馬達範例
- NuEEPROM 顯示從站EEPROM範例
- NuGPIO
   ECM-XF上GPIO測試範例
- NuHoming 一個驅動器馬達歸零(Homing)範例
- NuHSP
   訊成HSP上的兩個馬達範例
- NuHSP\_A 訊成HSP上的兩個馬達範例 (自動切換402狀態機Servo on)
- NuHSP\_IO
   訊成HSP上的兩個馬達加上一個IO從站範例
- NulO 一個IO從站範例
- NuPP
   Profile position mode範例
- NuQEI 讀取ECM-XF晶片上編碼器訊號範例

# 2.5 Nuvoton新唐範例環境建立導引

- 至新唐官網依作業系統下載Nueclipse GCC(開源IDE),連結為官方下載頁面: <u>https://www.nuvoton.com/products/microcontrollers/arm-cortex-m4-mcus/m48</u> <u>7-ethernet-series/?group=Software&tab=2</u>
- 2. 在同一頁面下載Nu-Link Keil driver
- 3. 同一頁面下載M480\_BSP\_CMSIS
- 4. 安裝IDE與Keil driver
- 5. 下載範例程式並放入workspace位置
- 6. 開啟Eclipse並在Project Explorer的位置滑鼠右鍵選擇"Import"
- 7. 在匯入畫面依以下順序操作:General > existing project into workspace



lmport	
Select Create new projects from an archive file or directory.	Ľ
Select an import source:	
<ul> <li>Archive File</li> <li>Existing Projects into Workspace</li> <li>File System</li> <li>Preferences</li> <li>C/C++</li> <li>Git</li> <li>Install</li> <li>Oomph</li> <li>Remote Systems</li> <li>RPM</li> <li>Run/Debug</li> <li>Tasks</li> <li>Team</li> <li>Tracing</li> </ul>	~
? < Back Next > Finish	Cancel

- 8. 選擇預計開啟的根目錄, 範例程式會在中間的視窗出現並點選"Finish"
- 9. 在成功匯入專案後, 右鍵點選或上方工具列點選建置(build)專案。專案可以允許 建置Debug模式與Release模式, 建議使用Debug模式。





10. 當建置完成後, 設定Debug配置與Run配置。點選圖標名稱為"New launch configuration", 接下來應會自動生成剛剛建置完成的專案名稱, 確認名稱正確後 點擊Debug, 系統會自動將程式寫入晶片中

Debug Configurations						×
Create, manage, and run configurations						The
	Name: NuGPIO Debug					~~
New launch configuration           © C/C++ Application           © C/C++ Attach to Application           © C/C++ Postmortem Debugger           © C/C++ Remote Amplication	Main      Moin      Moin      Debugger      Startup      Source      Project:     NuCPIO     Correct      Automation	Common				Browse
C GDB Hardware Debugging C GDB Nuvoton Nu-Link Debugging C ASDF Debug C DRIVE_IO_2 Debug	Debug/NuGPIO.elf Build (if required) before launching			Variables	Search Project	Browse
E Drive Do 2 Debug E Drive Do 2 Debug NuDAC AD Debug E NuDAC AD Debug E NuHY Debug E NuHYP Debug E NuHYP Debug E Space fet Debug E Space fet Debug E TEST_C Debug E TEST_C Debug E TEST_C Debug	Build configuration: Select Automatically O Enable auto build @ Use workspace settings		O Diable auto build Configure Workspace Settings_			~
Filter matched 20 of 31 items					Revert	Apply
0					Debug	Close

## 2.6 Nuvoton修改範例程式為實際應用導引

正確開啟範例程式後可依以下步驟順序進行改寫

使用Drive的應用以STFDrive或NuDrive為範本進行修改,中間會有其他參考別範例中 的做法

PdoDefine.h設定

 點選並進入PdoDefine.h中後,在#define中輸入從站數量與單站軸數。目前馬達 控制器發展已有單控制器可多軸控制,所以可以是一個控制器帶動兩馬達進行 運動,這樣的情況就會是一個從站與兩個軸數。

```
#define TEST_DRV_CNT
#define TEST_IO_CNT
```

 完成設定後拉至下方繼續設定控制器資料結構(PDO structure)。資料結構分為 RxPDO與TxPDO兩種,不同的從站如果資料結夠不相同可以分開設定後最後同 整。

```
typedef struct __attribute__((__packed__)) __axis_rxpdo_st_def_t
{
                            // 0x6x40 - Control word
   uint16_t u16CtlWord;
   int32_t n32Target;
                            // 0x6x7A - Target position
                             // 0x60FF - Target velocity
#ifdef TEST HSP DEV
                          // Øx6xFE
   int32_t
            n320ut;
#endif
}AXIS RXPDO ST DEF T;
typedef struct __attribute__((__packed __)) __axis_txpdo_st_def_t
{
                             // 0x6x41 - Status word
   uint16_t u16StaWord;
   int32_t n32Actual;
                            // 0x6x64 - Actual position
                             // 0x6063 -
#ifdef TEST HSP DEV
                             // Øx6xFD
   int32_t
            n32In;
#endif
}AXIS_TXPDO_ST_DEF_T;
```

1

0



STM32系列中main\_ini.c或Nuvoton系列中main.c設定

 首先要先設定SPI通訊時間, STM32系列中的SPI通訊時間必須另外安裝 STM32CubeMX來設定, Nuvoton系則以main.c中的#define TEST\_SPI\_FREQ 進行設定或是尋找UserSys\_Init()並填入SPI傳輸值。同時輸入欲測試的週期時 間

#define TEST\_SPI\_FREQ
#define DC\_ACTIVE\_CODE
#define BASE\_CYCTIME

24000000 0x300 1000000

UserSys\_Init(TEST\_SPI\_FREQ);

2. 在此範例中有可以進行週期時間測試用的define可以直接倍數測試當前設定的 週期時間,並提供RPM與PPR相關的設定、速度相關設定

```
* TEST CYCTIME DIVID
* 1 : for TEST_CYCTIME_MULTI
 * 2 : 0.5ms
* 4 : 0.25ms
* 8 : 0.125ms
 * TEST CYCTIME MULTI
* 1 : for TEST_CYCTIME_DIVID
* 2 : 2ms
 * 4 : 4ms
* */
#define TEST_CYCTIME_DIVID
                               1
#define TEST_CYCTIME_MULTI
                               1
#define TEST_CYCTIME_NS
                               ((BASE_CYCTIME*TEST_CYCTIME_MULTI)/TEST_CYCTIME_DIVID)
#define ONE_SEC_CYC_CNT
                               (100000000/TEST_CYCTIME_NS)
```

3. 設定完成後下方為FIFO設置, FIFO為一暫存空間並依照順序與週期時間將命令 由主站送至從站,以部分馬達控制為例週期時間為1ms時如果每次只送一次命令 ,很有可能會造成從站控制器以將步驟完成並閒置的狀態,導致馬達控制會有不 連續運作的情況,此時就應加大每週期時間所傳輸的命令數量,這邊就會修改 TEST\_PDO\_TO\_FIFO\_ONCE的數值,2為每周期間所傳輸的命令數量為兩筆。 此步驟需經過測試才可得出最佳數值

 3. 設定完成後跳至主程式int main\_ini()或int main()中開始進行程式的編寫與說明, 一開始會先將記憶體的位置清空以確保接下來運作正常



```
uint64 t u64Data;
int nStartFIFO = 0;
int i = 0, DrvIdx = 0, AxisIdx = 0, YasIdx = 0, nCnt1Sec=0, nRunTimeCnt=0;
int nVel = 0, nSumVel = 0, n32D0=0x5555;
int nret = 0, nServoState = 1;
int nLogStart = 0;
int n32CurPos[TEST AXIS CNT];
uint16_t u16LastSW[TEST_AXIS_CNT], u16StaWord[TEST_DRV_CNT][N_AXIS_IN_ONE_DRV];
uint32 t u32CycTimeCnt = 0, u32RunTimeCnt = 0, u32LogFifoCnt=0;
uint8 t u8LEDAxis = 0;
uint8_t u8LEDBit = 0;
uint8_t u8Version = 0, u8FifoCnt = 0, u8FifoCntMax = TEST_RXFIF0_CNT;
uint8_t u8State = 0, u8WkcErrCnt = 0, u8CrcErrCnt = 0, u8IsSlvAlive = 0;
uint8 t u8LastState = 0, u8LastWkcErrCnt = 0, u8LastCrcErrCnt = 0;
uint16_t u16RxPDOSize = 0, u16TxPDOSize = 0, u16SpiSize = 0;
int nDriveRxPDOSize = 0, nDriveTxPDOSize = 0;
int8 t SlaveCnt = 0;
RXPDO_ST_DEF_T *pAllDevRx;
TXPDO_ST_DEF_T *pAllDevTx;
AXIS_RXPDO_ST_DEF_T *pRxPDOAxis;
AXIS_TXPDO_ST_DEF_T *pTxPDOAxis;
memset(RxPDOData, 0, sizeof(RxPDOData));
memset(TxPDOData, 0, sizeof(TxPDOData));
memset(nPos, 0, sizeof(nPos));
```

- 5. EtherCAT主站的第一個命令為ECM\_InitLibrary(&u16SpiSize)。當中會執行SPI 資料大小的設定與確認,同時會確認主站IC的韌體版本等功能。u16SpiSize為欲 設定新SPI資料大小,如其值為0時則使用預設大小112Bytes。 u8Version = ECM\_InitLibrary(&u16SpiSize);
- 當確認項目皆正常執行後,開始進行EtherCAT初始化 ECM\_EcatInit(DCActCode, CycleTime),並進入Init state ECM\_EcatInit(DCActCode, CycleTime)中DCActCode值0為關閉DC sync功能、 0x300啟動Sync0、0x700同時開啟Sync0與Sync1, CycleTime為週期時間,單位 為ns。

※此函數會針對 "所有" Slave 下達相同的DCActCode, 若對於不同Slave要設定 不同的DCActCode, 則需使用ECM\_CMD\_ECAT\_DCSYNC(命令代碼50) ECM\_EcatInit(DC\_ACTIVE\_CODE, (BASE\_CYCTIME\*TEST\_CYCTIME\_MULTI) / TEST\_CYCTIME\_DIVID);

- 7. 確認主站與從站連接後透過ECM\_StateCheck(Slave, ExpectedState, Timeout) 進入Pro-OP開始設定PDO配置,如果從站已有預設配置則只需要給予相對應的 Map Index即可,如為特殊配置則需告知,SetPdoConfig提供最多3組PDO各8個 Object的配置大小,多數範例配置內容在main\_ini.c或main.c中,目前唯Drive範 例在另外在Utility.c中。如果使用ConfigDrive進行配置,內部已有Reconfig與 ShowPDOConfig的動作,所以無需另外加入Reconfig動作,其他範例則需在完 成Configure PDO後加入Reconfigure的動作與ShowPDOConfig。 ECM\_StateCheck(Slave, ExpectedState, Timeout)中Slave如為0xFF則表示針 對所有從站同時命令,ExpectedState則是欲前往之狀態(Pro-OP、Safe-OP與 OP狀態),Timeout為逾時等待時間,如出現無法狀態切換時可能等待時間過短。 ConfigDrive(1, 0, (TEST\_DRV\_CNT - 1), 1, N\_AXIS\_IN\_ONE\_DRV, 0x1602, 0x1A02);
  - 或



```
RxPD0Config[i].SmaIdx = RxPD0_ASSIGN_IDX;
RxPD0Config[i].PD0Cnt = 1;
RxPD0Config[i].ObjsCnt[0] = 2;
SetPd0ConfIg[i].ObjsCnt[0] = 2;
// the 1st parameter is PD0_CONFIG_HEAD
// the 2nd parameter is 0 due to RxPD0Config.PD0Cnt = 1
// the 3rd parameter is 0 for the first object as RxPD0Config.ObjsCnt[0] = 2
// the 4th parameter is a control word index 0x6040
// the 5th parameter is a control word index 0x6040
// the 6th parameter is a sub-index for 0x6040
// the 6th parameter is a bit size for 4th parameter
SetPd0ConfIp[(&RxPD0Config[i], 0, 1, 0x607A, 0, 32); //target position // 32 bits = 4 bytes for TEST_RXPD0_SIZE
TxPD0Config[i].SmaIdx = TxPD0_ASSIGN_IDX;
TxPD0Config[i].MapIdx[0] = TxPD0_MAP_IDX;
TxPD0Config[i].MapIdx[0] = z;
SetPdoConfIb1(&TxPD0Config[i], 0, 1, 0x6041, 0, 16); //status word // 16 bits = 2 bytes for TEST_TXPD0_SIZE
SetPd0ConfIb1(&TxPD0Config[i], 0, 1, 0x6044, 0, 32); //actual position // 32 bits = 4 bytes for TEST_TXPD0_SIZE
```

ECM\_EcatReconfig();

- 完成配置後進行ECM\_CheckMEMSpace(TEST\_PDO\_FIFO\_ONCE), 此步驟 為確認各部分相關記憶體的大小, 參數需輸入每週期時間塞入多少命令。
   ECM\_CheckMEMSpace(TEST\_PDO\_TO\_FIFO\_ONCE);
- 9. 檢查確認沒有問題後利用ECM\_StateCheck(Slave, ExpectedState, Timeout)進入Safe-OP狀態

ECM\_StateCheck(0xFF, EC\_STATE\_SAFE\_OP, 1000);

- 10.可進入Safe-OP狀態後繼續執行ECM\_StateCheck(Slave, ExpectedState, Timeout)進入OP狀態
- 11. 進入OP狀態後執行ECM\_CheckDCStable()確認DC狀態穩定 ECM\_CheckDCStable();
- 12. 當確認完成後,以ECM\_Drv402SM\_StateSet(Axes, ServoOn/OffState)或 ECM\_Drv402SM\_Enable(Axes, Slaves)啟用402 state machine至servo on階 段,此階段為自動切換402 state machine的方式,如果從站無法使用以上兩種切 換方式時,可以參考STF4HSP或NuHSP中的手動切換模式進行單步切換至 Servo On狀態

```
ECM_Drv402SM_StateSet((DrvIdx*N_AXIS_IN_ONE_DRV) + AxisIdx, SERVO_ON_STATE);
```

PdoExchangeAndGet402State(DrvIdx, AxisIdx, &u8State);

或

ECM\_Drv402SM\_Enable(0, 0);

13. Servo On完成後進行編碼器與控制器的位置對齊AlignmentPosition(), 並使用 ECM\_InitFIFO()初始化FIFO同時使用ECM\_ClearFIFO(Direction)將FIFO內部 清空後以ECM\_EnableFIFO(Enable)來啟用FIFO功能 ECM\_ClearFIFO(0), 0為清除TxPDO與RxPDO ECM EnableFIFO(1), 1為Enable、0為Disable

ECM\_InitFIF0(); ECM\_ClearFIF0(0); // 0 for TX and RX both PRINTF("ClearFIF0\r\n"); ECM\_EnableFIF0(1); // Enable FIF0 PRINTF("EnableFIF0\r\n");


14.完成上方步驟後已完成相關相關設定,接下來的資料交換皆會在while迴圈中不 斷重複運行,以判斷FIFO的數量為基準,如果FIFO已經快要塞滿時,則選擇暫停 塞入命令,此外持續將命令塞入FIFO,速度與位置相關設定在Utility.c中可依需 求進行編寫。

### 2.7 常見錯誤說明

在執行相關設定時如果跳出Error相關問題,請依以下進行排除

階段	顯示	解決方法
ECM_Init_Library	wait ASYNC done timeout	請確認主站與從站連接是 否正常, 主站與從站網口 燈號是否亮起
ECM_Init_Library	u8ErrorStatus 0x40	初次設定會跳出的警告, Novoton系列按Enter繼續
通期	wait ASYNC done timeout	等待時間不足或從站錯誤 (上次的非即時命令尚未完 成)
通期	CRC Error	SPI傳輸訊號錯誤, 請確認 SPI主站與EtherCAT主站 IC連線

# 2.8 402自動切換與手動切換方法

- 在STF4Drive範例中用ECM\_Drv402SM\_StateSet()與 PdoExchangeAndGet402State()來進行Servo ON
- 其他範例中以ECM\_Drv402SM\_Enable()的方式進行Servo On ECM\_Drv402SM\_Enable中的參數分別為軸與從站 所以當為一對一控制器時兩站Enable就會變成 ECM\_Drv402SM\_Enable(0, 0); ECM\_Drv402SM\_Enable(1, 1); 如果為一對二控制器且有兩站時命令就會變成 ECM\_Drv402SM\_Enable(0, 0); //第0站第0個軸



ECM Drv402SM Enable(1,0); //第0站第1個軸

ECM Drv402SM Enable(2, 1); //第1站第2個軸

ECM Drv402SM Enable(3, 1); //第1站第3個軸

#### 手動切換的方式請參考STF4HSP或NuHSP中的切換邏輯進行切換402 state • machine

for(DrvIdx=0;DrvIdx<TEST\_SLAVE\_CNT;DrvIdx++){</pre> for(AxisIdx=0;AxisIdx<N\_DRV\_IN\_ONE\_SLV;AxisIdx++){

j = 0; while(1){ nret = ECM\_EcatPdoFifoDataExchange(PD0\_FIF0\_DEFAULT\_CNT, RxData, TxData, u16PD0Size, &u8FifoCnt, &u8WkcErrCnt, &u8CrcErrCnt); if(\_\_\_t)]

u16LogStatus[(DrvIdx\*N\_DRV\_IN\_ONE\_SLV)+AxisIdx] = pTxPDOData->DRIVE\_GROUP\_0[DrvIdx].HSP[AxisIdx].u16StaWord;

PDOstate[(DrvIdx\*N\_DRV\_IN\_ONE\_SLV)+AxisIdx] = pTxPDOData->DRIVE\_GROUP\_0[DrvIdx].HSP[AxisIdx].u16StaWord & CIA402\_SW\_STATE\_MASK; PRINTF("DrvIdx = %d, AxisIdx = %d, state = 0x%x\r\n", DrvIdx, AxisIdx, PDOstate[(DrvIdx\*N\_DRV\_IN\_ONE\_SLV)+AxisIdx]);

if(PDOstate[(DrvIdx\*N\_DRV\_IN\_ONE\_SLV)+AxisIdx]==CIA402\_SW\_NOTREADYTOSWITCHON){ UserDelay(1000);

if(PDOstate[(DrvIdx\*N\_DRV\_IN\_ONE\_SLV)+AxisIdx]==CIA402\_SW\_SWITCHEDONDISABLED){
 pRxPDOData->DRIVE\_GROUP\_0[DrvIdx].HSP[AxisIdx].u16CtrlWord = 0x6; //Control word: Shutdown = 0x6

}
if(PDOstate[(DrvIdx\*N\_DRV\_IN\_ONE\_SLV)+AxisIdx]==CIA402\_SW\_READYTOSWITCHON){
 pRxPDOData->DRIVE\_GROUP\_0[DrvIdx].HSP[AxisIdx].u16CtrINord = 0x7; //Control word: Switch on = 0x7

/ if(PDOstate[(DrvIdx\*N\_DRV\_IN\_ONE\_SLV)+AxisIdx]==CIA402\_SW\_SWITCHEDON){ pRxPDOData->DRIVE\_GROUP\_0[DrvIdx].HSP[AxisIdx].u16CtrlWord = 0xF; //Control word: Enable operation = 0xF

if(PDOstate[(DrvIdx\*N\_DRV\_IN\_ONE\_SLV)+AxisIdx]==CIA402\_SW\_OPERATIONENABLED){

j++; if(j==3){ break; }

if(PDOstate[(DrvIdx\*N\_DRV\_IN\_ONE\_SLV)+AxisIdx]==CIA402\_SW\_QUICKSTOPACTIVE){
 pRxPD0Data->DRIVE\_GROUP\_0[DrvIdx].HSP[AxisIdx].u16CtrlWord = 0x0; //Control word: Disable voltage = 0x0

; if(PDOstate[(DrvIdx\*N\_DRV\_IN\_ONE\_SLV)+AxisIdx]==CIA402\_SW\_FAULTREACTIONACTIVE){ UserDelay(1000);

}
if(PDOstate[(DrvIdx\*N\_DRV\_IN\_ONE\_SLV)+AxisIdx]==CIA402\_SW\_FAULT){
 pRxPDOData->DRIVE\_GROUP\_0[DrvIdx].HSP[AxisIdx].u16CtrlWord = 0x0; //Control word: Fault reset = 0x0->0x80
 nret = ECM\_EcatPdoFifoDataExchange(PD0\_FIF0\_DEFAULT\_CNT, RxData, TxData, u16PD0Size, &u8FifoCnt, &u8WkcErrCnt, &u8CrcErrCnt);
 iv = Div/Constr UserDelay(1000);

pRxPDDData->DRIVE\_GROUP\_0[DrvIdx].HSP[AxisIdx].u16CtrlWord = 0x80; nret = ECM\_EcatPdoFifoDataExchange(PD0\_FIF0\_DEFAULT\_CNT, RxData, TxData, u16PD0Size, &u8FifoCnt, &u8WkcErrCnt, &u8CrcErrCnt); UserDelay(1000);



# 第3章 Beaglebone透過SPI介面範例說明

### 3.1 設置Beaglebone系統

STEP1.製作Beaglebone的系統SD卡

製作Beaglebone的系統碟,可至Beaglebone的官方頁面下載後安裝,我們建議使用 Debian 9或之後的Debian穩定發行版。若用戶需要用自行編譯的linux內核,我們建議使 用4.4以後的LTS版本,開啟MCSPI和SPIDEV功能,且支援U-boot overlay。

STEP2. (開機步驟可參考2.2)開機後連接Beaglebone連接網際網路, 安裝 device-tree-compiler套件, 使用Debian系統的可用以下指令:

\$ sudo apt-get install device-tree-compiler

STEP3. 將系統韌體的dtbo反編譯成dts, 以下為指令:

\$ dtc -I dtb -O dts /lib/firmware/BB-SPIDEV1-00A0.dtbo > BB-SPIDEV1-00A0.dts

編輯dts,修改以下兩處:

一, 搜尋 pinctrl-single, pins, 將

pinctrl-single,pins = < 0x190 0x33 0x194 0x33 0x198 0x13 0x19c 0x13 >;

改成

pinctrl-single,pins = < 0x190 0x33 0x194 0x13 0x198 0x33 0x19c 0x13 >;

二, 搜尋 ti,pio-mode, 在下一行插入以下文字:

ti,pindir-d0-out-d1-in;

將修改後的dts檔編譯為dtbo, 編譯完成後將dtbo檔放在/lib/firmware目錄下。以下為編譯的指令:

\$ dtc -I dts -O dtb BB-SPIDEV1-00A0.dts > BB-SPIDEV1-00A0.dtbo



STEP4. 編輯/boot/uEnv.txt, 加上以下兩行:

enable\_uboot\_cape\_universal=1

uboot\_overlay\_addr4=/lib/firmware/BB-SPIDEV1-00A0.dtbo

STEP5. 重啟Beaglebone, 上述步驟完成後可以看到/dev/spidev2.0、/dev/spidev2.1裝置。若SPIDEV裝置沒有出現, 請特別留意P9.28、P9.29、P9.30、P9.31是否有和其他的 overlay有衝突。

### 3.2 測試範例

以Beaglebone+UART為測試範例,安裝USB to TTL驅動程式。

STEP1. TTL UART端接Beaglebone的J1, 打開COM PORT程式, 下圖3-1是以PuTTY 為範例。

Čelovov.	
Category: Session Logging Terminal Keyboard Bull Features Window Appearance Behaviour Translation Selection Colours Connection Data Proxy Telnet Riogin SstH Serial	Basic options for your PuTTY session Specify the destination you want to connect to Serial line Speed COMS COM編號請見 PC 115200 Connection type: 約裝置管理員 ○ Raw ○ Islnet ○ Riogin ○ SSH ⑧ Segial Load, save or delete a stored session Savgd Sessions
	Load Saye Delete
About	Close window on egit. ○ Always ○ Never ⑧ Only on clean exit 上面填完後點Open ②pen ③ancel

圖3-1



STEP2. 使用跳線帽將ECM-XFU-SK的J2短路(下圖3-2紅框處), J2短路後代表 ECM-XFU-SK的電路由外板(即Beaglebone)供電, 插上Beaglebone, EtherCAT網路線 接ECM-XF-SK的網口。



圖3-2

STEP3. 電源接Beaglebone P4(Micro USB), ECM-XFU-SK由Beaglebone 供電。下圖 3-3為接線方式, 下圖為ECM-XFU-SK V1.3接法, 圖下板是Beaglebone, 上板是 ECM-XFU-SK。



圖3-3



上電後靜待數秒進入登入畫面, 輸入帳密登入, 下圖2-4為範例登入畫面



圖2-4

STEP4.下載ECM-XFU-SK驅動和Linux範例,安裝GCC後可以編譯範例程式



# 第4章 Windows Visual Stdio設置導引

#### 4.1 Visual Stdio C++

建立一個新的專案,並於專案中[加入] [現有項目],加入所有.h標頭檔





### 於專案 [屬性] [VC++目錄]中加入相關函式庫名稱及位置

檔案	(F)	編輯(E)	檢視(V)	專案(P)	建置(B)
↑	重定	方案目標			
***	加入	.類別(F)			
œ.‡•	類別	精靈(Z)		Ctrl+S	hift+X
*ł\$	加入	資源(R)			
*ם	加入	.新項目(W)	)	Ctrl+S	hift+A
to.	加入	現有項目(	G)	Shift+	Alt+A
	Арр	lication In	sights(A)		۲
*-	新的	師選條件()	F)		
ß	顯示	所有檔案((	0)		
	重新	掃描解決力	5案(S)		
	顯示	瀏覽資料層	<b>目錯誤</b>		
	清除	瀏覽資料層	齳誤		
t∌	新増	連線服務((	C)		
₽	設定	為啟始專業	롩(A)		
	匯出	範本(E)			
Ě	管理	NuGet 套	件(N)		
ىر	屬性	:(P)			

XFU\_config\_demo 屬性頁



? ×



# 於[連結器] 加入相關的dll檔

XFU_config_demo 屬性頁			? ×
組態(C): 作用中 (Debug)	✓ 平台(P): 作用中 (Win3)	2)	✓ 組態管理員(O)
<ul> <li>▲ 組態屬性         <ul> <li>一般                  <ul></ul></li></ul></li></ul>	其他相依性 忽略所有預設程式庫 忽略特定的預設程式庫 模組定義榴 將模組加入至組件 內嵌受控資源榴 強制符號參考 延遲載入 DLL 組件連結資源 其他相依性 指定可加入連結命令列的其他項目 [f	EcmXfUsbDrv.lib;kernel	32.lib;user32.lib;gdi32.lib;winspool.lib
		H	<b>定 取</b> 満 套用(A)



# 第5章 ECM-XFU ITE整合測試環境說明

# 5.1 ITE簡介

整合測試環境Integrated Test Environment, 簡稱ITE, 是可以透過USB直接讀取/寫入 ECM-XFU-SK, 進而透過EtherCAT通訊協定控制從站

### 5.2 一般操作說明

ITE 支援 使用USB 或 PCIe 連接至電腦的ECM-XFU-SK/ECM-XF-PCIe裝置。 ECM-XFU-SK 透過USB連結到電腦後, 裝置管理員顯示「USB輸入裝置」及「符合HID標 準的廠商定義裝置」(HID-compliant device)。PCIe卡裝到電腦後, 裝置管理員則顯示「 ECM-PCIe EtherCAT Master Controller」

	書 裝置管理員
檔案(F) 動作(A) 檢視(V) 說明(H)	檔案(F) 動作(A) 檢視(V) 說明(H)
🗢 🔿   📰   🗾 🎫   晃	
<ul> <li>✓          ■ DESKTOP-3T0F5CS     </li> <li>&gt; ■ IDE ATA/ATAPI 控制器     <li>✓          ● IDE ATA/ATAPI 控制器     </li> <li>✓          ● Logitech USB Input Device     </li> <li>✓          ● USB 輸入裝置     </li> <li>✓          ● USB 輸入裝置     </li> <li>✓          ● USB 輸入裝置     </li> <li>✓          ○ USB 輸入裝置     </li> <li>✓          ○ OSB 輸入装置     </li> <li>✓          ○ OSB 執入装置     </li> <li>✓</li></li></ul>	<ul> <li>✓ ● 条統裝置</li> <li>▲ ACPI 固定功能按鈕</li> <li>■ ACPI 風扇</li> </ul>
符合 HID 標準的廠商定義裝置 研 符合 HID 標準的廠商定義裝置 行合 HID 標準的廠商定義裝置 ECM-XFU-SK模組	► High Definition Audio 控制器 ECM-XF-PCIe装置



打開ECM-XF ITE軟體, 於左上角Device type選擇連線方式 (USB或PCIe), 確認選擇 正確後, 按下"Open"按鈕。

NEXTW ECM-XF ITE (Evaluation) ECM-XF Tools		– 🗆 X
Device type <pre></pre>		
OPCIe Open		
Run Load ENI		
SDO ReScan	RxPDO	TxPDO
Cycle TIme: 2 ~ ms		
☑DC Active		
EtherCAT State		
Slave		
INIT		
PRE-OP		
SAFE-OP		
OP		



打開連接後, 左下角狀態應該為Pre-OP狀態。右上角的文字框將顯示 訊息(如 IC韌體 版本、執行過程訊息等)。從站的預設的設置將顯示在 RxPDO 和 TxPDO 文字框中。

前面的00、01代表從站順序編號,後面則代表該從站的製造商編號(Vendor ID)及產品 編號(Product ID),透過點擊 RxPDO 和 TxPDO 文字框中內容前面的"+"符號,會顯示 該從站的詳細的PDO內容。

下圖為連結2台的 YASKAWA 驅動器的預設配置。



在 Pre-OP 狀態下, 可以進行讀/寫 SDO、使用 ENI中的配置, 或是使用Tools中的 SII editor 和 code generator。

左下角可改變從站的狀態, 欲切至OP狀態, 必須依照Init->PREOP->SafeOP->OP依序 切換, 於OP狀態可返回之前任一狀態。需要注意狀態變更的順序(從上到下), 否則切換 狀態會失敗。若切換狀態失敗, 可參考訊息區的AL Status Code, 以瞭解失敗原因。常 見由PreOP切換至SafeOP失敗的原因在於(1) Cycle Time設定不支援、(2) DC 設定不 支援、(3) RxPDO或TxPDO設定不支援、(4) Slave EEPROM設定數值有誤、(5) 其他。 常見由SafeOP切換至OP失敗的原因為 DC 尚未穩定或其他因素。

當進入 OP 狀態時,將會彈出視窗顯示是否要進行cyclic data exchange 的詢問視窗, 選擇[確定],即開始與從站進行實際的週期性資料交換,若選擇[取消],使用介面上的數



值並不會與從站進行資料交換。按下左邊的[Run]按鈕則會開始進行週期性資料交換, 再按下[Stop]則會停止將介面上的數值與從站進行週期性資料交換。



RxPDO為主站送至從站的命令, 在OP模式下, 可變更RxPDO內各Object的內容值。 TxPDO為從站送至主站的狀態, 下圖 顯示 2個從站的當前狀態。如果需要, 可以右鍵 單擊Hex旁的方框以切換為十六進制。而在RxPDO雙擊值的部分可以變更數值。





雙擊RxPDO中 0x6040 (Control Word)可以改變運行模式下的 402 狀態機。但不需要 每次都按 [確定]來改變狀態。可以觀察到TxPDO中0x6041 (Status Word)發生變化。

針對伺服馬達若要激磁(Servo on), 可透過改變0x6040的數值, 依序選擇 "Shutdown(0x6)"、"Switch On(0x7)"、"Enable Operation(0xF)", 切至 Enable Operation後馬達即會激磁, 詳細的說明可參考從站關於0x6040的說明。





在操作模式為CSP的情況下(即Object 0x6060:0 設定為8), 驅動器會於每個週期嘗試 轉動馬達至目標位置(0x607A), 而回傳目前的實際位置(0x6064)。激磁(Servo on)後 RxPDO中的0x6064(Position actual value)會與TxPDO中的0x607A(Target position) 值相同。接著可以雙擊 0x607A (Target position)設置下一個週期時間的新目標位置。 亦可使用拉bar來改變數值。

更多的運行模式請參閱從站的使用手冊關於Object 0x6060:0 的說明。



## 5.3 SDO操作說明

SDO是非週期性資料交換, 在PRE-OP、SAFE-OP及OP階段均可進行, 按下[SDO]按鈕 後會出現SDO視窗。例如, 欲讀取運行模式(Modes of Operation Display), 可讀取 Object Index為0x6061, Sub Index為0, Byte length(Size) 為1, 然後按下[Read]即可於 右側Value地方讀到數值。更多關於Object的定義請參閱從站的使用手冊。

NEXTW ECM-XF ITE (Licensed) ECM-XF Tools	- D X
Device type ⊛USB ○PCIe Close	Waiting ECM_EcatStateUpdate ECM_EcatStateUpdate done Waiting ECM_EcatStateGet(255) (255) EcState(0x2)
Run Load ENI	ECM_EcatStateGet(255) done
SDO ReScan Cycle TIme: 2 ✓ r ☑ DC Active EtherCAT State Slave All ✓ INIT PRE-OP SAFE-OP OP	Dv DDO       Y         SDO       Y         000539/0x02200001         1A01]         11:00         000         64:00         Slave index       Y         Download       Upload

若欲透過SDO機制寫入Object, 則除了Index、Sub Index、Byte length均需填寫外, Value亦需填寫。例如欲寫入運行模式(Modes of Operation), 可寫入Object Index為 0x6060, Sub Index為0, Byte length(Size) 為1, Value則為欲寫入的值, 然後按下 [Write]即會寫入。



NEXTW ECM-XF ITE (Licensed) ECM-XF Tools	- 🗆 X
Device type © USB ○ PCIe Close	<pre>Waiting ECM_EcatStateUpdate ECM_EcatStateUpdate done Waiting ECM_EcatStateGet(255) (255) EcState(0x2) ECM_EcatStateGet(255) done</pre>
Run Load ENI	·
SDO ReScan Cycle TIme: 2 ✓ r ☑ DC Active EtherCAT State Slave All ✓ INIT PRE-OP SAFE-OP OP	Dv DDN         Tv DDN           SDO         X           ECM_EcatSdoRead(0, 0x6061, 0x00, 1)         0005339/0x02200001           ECM_EcatSdoWite(0, 0x6060, 0x00, 1, 0x8)         1:00           ECM_EcatSdoWite(0, 0x6060, 0x00, 1, 0x8)         0000           ECM_EcatSdoWite(0, 0x6060, 0x00, 1, 0x8)         0000           Save index         0             Download         Upload

若欲確認是否正確寫入,則可使用SDO Read讀取該數值。

按下[Get OD]按鈕, 等待數秒後, 可自從站讀回所有的Object, [Access]顯示R代表可讀, W代表可寫, 前兩位文字代表於PerOP階段的讀寫狀態, 中間兩文字代表於SafeOP階段的讀寫狀態, 最後兩文字代表於OP階段的讀寫狀態。請注意, 列出之資訊係由從站提供, 主站無法驗證其正確性, 建議仍應參考從站手冊的說明。可選擇欲讀寫的Object, 其 資料會自動帶入至下方欄位。

No.     Index     Sub     Length     Type     Access     Name       1034     0x60FE     2     32     UNSIGNED32     RWRWW     Digital outputs (2) Bit mask       1035     0x60FF     0     32     INTEGER32     RWRWW     Target velocity       1036     0x6403     0     272     VISIBLE_STR(272)     R_R_R_     Motor catalogue number       1037     0x6502     0     32     UNSIGNED32     R_R_R_     Supported drive modes	
1034         0x60FE         2         32         UNSIGNED32         RWRWRW         Digital outputs (2) Bit mask           1035         0x60FF         0         32         INTEGER32         RWRWRW         Target velocity           1036         0x6403         0         272         VISIBLE_STR(272)         R_R_R_         Motor catalogue number           1037         0x6502         0         32         UNSIGNED32         R_R_R_         Supported drive modes	
1035         0x60FF         0         32         INTEGER32         RWRWRW         Target velocity           1036         0x6403         0         272         VISIBLE_STR(272)         R_R_R_         Motor catalogue number           1037         0x6502         0         32         UNSIGNED32         R_R_R_         Supported drive modes	
1036         0x6403         0         272         VISIBLE_STR(272)         R_R_R_         Motor catalogue number           1037         0x6502         0         32         UNSIGNED32         R_R_R_         Supported drive modes	
1037 0x6502 0 32 UNSIGNED32 R_R_R_ Supported drive modes	
	>



### 5.4 透過Load ENI修改RxPDO及TxPDO配置說明

ITE支援讀取 ENI (EtherCAT Network Information)內的 PDO 配置。按下[Open]後再 按下[Load ENI]按鈕, 選擇ENI文件, ITE會根據所選的ENI文件嘗試配置從站PDO結 構。下圖顯示 RxPDO 和 TxPDO 經過ENI檔案中設置更改實體配置。







### 5.5 透過SDO修改RxPDO及TxPDO配置說明

若無ENI檔案亦可在PRE-OP階段自行透過SDO寫入0x1C12、0x1C13的方式來配置 PDO,例如欲設定RxPDO的Sync Manager PDO assign objects (0x1C12)設定其PDO mapping objects 為0x1601,詳細的作法及步驟如下:

1. SDO Write 0x1C12:0 (size 1) Value = 0

2. SDO Read 0x1C12:0 (size 1) 確認為 Value = 0

3. SDO Read 0x1C12:1 (size 2) 確認預設值(例如0x1600)

4. SDO Write 0x1C12:1 (size 2) Value為欲設定的PDO值(例如0x1601)

5. SDO Read 0x1C12:1 (size 2) 確認設定是否正確 (如0x1601)

6. SDO Write 0x1C12:0 (size 1) Value= 1 (PDO mapping objects的數量)

7. SDO Read 0x1C12:0 (size 1) 確認Value為1

8. 按下[ReScan]

若欲進一步改變PDO mapping objects中的內容mapping entry,則可在上述2和3中間 加做以下步驟: (如欲將0x1601設有2個mapping,分別為0x6040:0 size 2 bytes及 0x607A:0 size 4 bytes)

1 SDO Write 0x1601:0 (size 1) Value = 0

2 SDO Read 0x1601:0 (size 1) 確認為 Value = 0

3 SDO Write 0x1601:1 (size 4) Value = 0x60400010, 其中bit0-7為mapped object的 長度(單位bit), bit8-15為mapped object的sub Index, bit16-31為mapped object的 Index。

4 SDO Read 0x1601:1 (size 4) 確認Value = 0x60400010。

5 SDO Write 0x1601:2 (size 4) Value = 0x607A0020, 其中bit0-7為mapped object的 長度(單位bit), bit8-15為mapped object的sub Index, bit16-31為mapped object的 Index

6 SDO Read 0x1601:2 (size 4) 確認Value = 0x607A0020。

7 SDO Write 0x1601:0 (size 1) Value = 2(mapping entry的數量)



8 SDO Read 0x1601:0 (size 1) 確認為 Value = 2

## 5.6 更新韌體 (Update firmware)

ITE 支援了 ECM-XF/ECM-XFU 的韌體更新。請先將硬體模式切換到update mode 並 單擊工具欄"ECM-XF", 然後選擇"FW update"。update mode詳情請參見硬體板的手冊 , 以ECM-XF-SK V1.2為例, 須使用跳線帽將J3中間針腳與L針腳短路即為updata mode。如果您的開發板未處於 update mode, 會出現如下圖所示的視窗。

EcmITE	×
Please clsoe device and set config pin	
確定	

如果您處於 update mode, 則在按下[FW update]後可選擇連結介面( USB 或 PCle), 之後再選擇更新檔案即可。請注意, 請務必確認所選的更新檔案是正確的, 錯誤的更新 檔可能導致無法正常運作, 且無法復原。

Firmware	Update		×
● USB			
		Update	Cancel



# 5.7 SII editor

Tools 中的[SII Editor], 可讀取從站 SII文件和並存成檔案, 或讀取檔案並寫入從站。請 注意必須 Pre-OP 狀態下使用此editor。



SII Editor		×
Number of columns	an <mark>a</mark> n th	
Number of columns	an <mark>a</mark> r th	Log
0000 06 00 01 EE	E8 03 00 00 01 00 00 00 00 00 14 00	Reading - 100%
0010 39 05 00 00	01 00 20 02 04 00 05 00 00 00 00 00 9	
0020 00 00 00 00	00 00 00 00 10 80 00 80 10 80 00	
0030 00 10 80 00	80 10 80 00 0C 00 00 00 00 00 00 00	
0040 00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00	
0050 00 00 00 00	00 00 00 00 00 00 00 00 00 00 00	
0060 00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00	
0070 00 00 00 00	00 00 00 00 00 00 00 00 01 00 01 00	
0080 28 00 02 00	01 02 03 00 1E 00 10 00 00 00 00 00 (	
0090 05 OF 00 00	00 01 00 02 00 00 00 00 00 00 00 00	
00 00 00 00 0A00	00 00 00 00 00 00 00 00 29 00 10 00)	
00B0 00 10 80 00	36 00 01 00 80 10 80 00 32 00 01 0062	
00C0 00 11 06 00	74 00 01 00 00 14 06 00 30 00 01 00t0	
00D0 FF FF FF FF	FF	
OOEO FF FF FF FF	FF	
OOFO FF FF FF FF	FF	
0100 FF FF FF FF	FF	
<	>	
Operation for		State Control
	VID: 0x539_PID: 0x2200001	PREOP
() File	Read Write Clear	Forward Backward



# 5.8 XFU週邊IO (Peripheral)

Peripheral的視窗可控制或顯示 ECM-XFU / ECM-XF-PCIe GPIO/QEI/ADC 的功能。

Peripł	hral																					$\times$
GPIC		00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	
	In	۲	۲	۲	۲	۲	۲	۲	۲	۲	۲	۲	۲	۲	۲	۲	۲	۲	۲	۲	۲	
	Out	Ο	Ο	Ο	Ο	Ο	$\odot$	Ο	Ο	Ο	$\odot$	Ο	Ο	$\odot$	$\odot$	$\odot$	Ο	$\bigcirc$	Ο	Ο	0	
	Data																					
		00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	
	ENC				<b>4</b> X		~	A					IFFER	ENTIA	L	DAC	0		]		Run	

### 5.9 評估版 與 授權版

左上方視窗標題欄中顯示Evaluation為評估版本,評估版本可切換狀態、讀取和寫入從站 PDO、與從站進行SDO操作、韌體升級等功能。

正式版本可操作完整的功能,包含Code Generator等。

正式版本於完成付費後點擊[ECM-XF]選單內[License], 會有詢問是否已取得License視窗, 點 選[取消], 會產生"activation.lis"文件, 將此文件連同ECM-XFU-SK或ECM-XF-PCIe序號寄至 service@nextw.com.tw, 將可取得專屬的授權檔案。

獲得授權後,將授權文件"EcmXF.lis"放入ITE執行檔 的同一文件夾中。然後單擊工具欄上的 [ECM-XF]中[License],於詢問視窗中選擇[確定],選擇授權文件並單擊打開。左上方的視窗將 顯示"Licensed"表示已升級為正式版。



## 5.10 代碼產生工具(Code Generator)

代碼產生工具必須是Lincese版本才能使用, 可讀取 ENI文件(EtherCAT Network Information File), 並生成配置 c 文件(PDO Configure code), 此文件可作為撰寫程式時 的配置參考。

代碼產生工具亦可將當前的 PDO 結構定義(PDO Structure define)另存為 .h 文件, 此 PDO結構定義以當下實際的PDO定義為主, 您可先使用[Load ENI] 自動配置ENI內的 定義, 或是使用SDO讀寫0x1C12、0x1C13後再生成 .h 文件之前, 此文件可作為撰寫程 式時的結構定義參考。

ECM-XF	Code Generator	×	ć
	PDO Configure code	PDO Structure define	
ENI:			
Save to			
		Generate Cancel	

配置參考程式(.c 文件)和 PDO 結構定義的參考文件(.h 文件)可以放入您自行撰寫的 應用程式中。

以下為簡單的的導入說明將 c 文件函式 ECM\_ConfigPDO\_CodeGen(void) 複製到 main.c 中, 如下圖所示。



#de	fine <u>ECM_TBL_CNT_CC</u> 4
Ģint	ECM_ConfigPDO_CodeCen(void)
{	<pre>int nDone = 0; int nRet = 0;</pre>
₽;	PDO_CONFIG_HEAD stEcmPdoTbl[ECM_TEL_CNT_CC] = {
	<pre>{     (0x539,0x2200001,0,1,0x1C12,(0x1601,0x0,0x0,0x0),(3,0,0,0),((16,0,0x6040),(32,0,0x607A),(8,0,0x6060)}),     {0x539,0x2200001,0,1,0x1C13,(0x1A01,0x0,0x0,0x0),(3,0,0,0),((16,0,0x6041),(32,0,0x6064),(8,0,0x6061))},     {0x539,0x2200001,1,1,0x1C13,(0x1A01,0x0,0x0,0x0),(3,0,0,0),((16,0,0x6041),(32,0,0x6064),(8,0,0x6061))});     for (int i = 0; i &lt; ECM_TEL_ONT_OC; i++) {         if (stEcmPdoTbl[i].PDOCnt) {             nRet = ECM_EcatSdoSetPdoConfig(&amp;stEcmPdoTbl[i]);             if (nRet == 1) {                 nDone++;             }         }     } } </pre>
	return nDone;
[}	
⊟int  {	main()
	uint16_t u16SpiDataSize = 992; //fixed in USB uint8_t u8Version;
	PD0_CONFIG_HEAD RxPD0Config[2]; PD0_CONFIG_HEAD TxPD0Config[2];
	uinto_t RxData[IESI_SPI_DATA_SIZE] = { 0 }; uinto_t TxData[TEST_SPI_DATA_SIZE] = { 0 };

在配置部分只需使用函數"ECM\_ConfigPDO\_CodeGen()"

```
nret = ECM_StateCheck(0xFF, EC_STATE_PRE_OP, 10000);// Set mode must be at PRE-OP state
if (nret == 0) {
    return -1;
}
std::cout << "All Slaves are in PRE-OP state\n";
std::cout << "Assign and configure PD0\n";
ECM_ConfigPD0_CodeCen();
// Call ECM_EcatReconfig() after setting PD0 configures
nret = ECM_EcatReconfig();
if (nret < 0) {
    std::cout << "ECM_ECAT_CONFIG error : " << nret << "\n";
    return -1;
}
```



EcmPdoDefine.h 是與 SPI 版本中的 PdoDefine.h 相同的功能文件。在main.c中使用 #include <EcmPdoDefine.h> 即可。



# 5.11 FoE (File over EtherCAT)

ITE可以利用FoE技術透過EtherCAT傳送檔案至從站,可透過ITE更新ES-6P6E從站的 韌體,請注意,從站將辨識檔案的正確性,僅正確的檔案才會用於更新從站的韌體。

請先選定從站編號(一次僅能選定一個從站),再按下Update 6P6E FW,選擇欲更新的 檔案(ECATFW\_\_),按下確認後即可。傳輸時間依檔案大小而定,一般需數十秒,請耐 心等候。

NEXTW ECM-XF ITE(Evaluation:23061701)		-  ×
ECM-XF Tools		
D SII Editor Periphral © Cyclic test CodeGenerator Update Pdo size info Update 6P6E FW	<pre>Waiting ECM_EcatStateUpdate ECM_EcatStateUpdate done Waiting ECM_EcatStateGet(0) (0) EcState(0x2) ECM_EcatStateGet(0) done</pre>	··· ^
Run Load ENT		~
	RxPDO	TxPDO
SDO ReScan CycleTime 2 ms DC Active EtherCAT State Slave 0 INIT PRE-OP SAFE-OP	■ 00-0x00101010/0x26483052	■ 00-0x00101010/0x26483052

\* 若針對其他從站進行FoE傳輸, 請注意從站是否支援FoE, 且檔案需小於256KB



# 第6章 ECM-XFU 範例函數說明

## 6.1 ECMXFUDRV簡介

ECMXFUDRV驅動函式庫支援ECM裝置(ECM-XF PCIe / ECM-XFU USB),使用者可在 Windows平台上開發EtherCAT應用,進而開發PC-base控制器。

ECM裝置皆具有ECM-XF晶片(從端), 提供PCIe或USB介面與PC連接(主端), 為主從通訊的架構, ECM-XFU交換資料封包大小為1024位元組(ECM-XF IC預設為144位元組, 資料長度可變更), 由PC傳送給ECM-XF/XFU的資料封包稱為命令封包, 由ECM-XF/XFU回傳給PC的資料 封包稱為回應封包。

函式庫內具有兩個大小為1024位元組的全域變數分別代表命令封包及回應封包,函式皆透過 全域變數pCmd指標存取命令封包,及全域變數pRet存取回應封包。

ECMXFUDRV的原始程式碼可於以下網址查閱與下載:

https://bitbucket.org/x-force/nextwecmxfudrv/src/master/

# 6.2 ECMXFUDRV API介紹

Ecr	EcmUsrDriver.cpp			
int3	int32_t ECM_PrintErrStatus(int32_t nCleanFlag)			
印出	印出錯誤名稱, 並根據參數nCleanFlag決定是否清除錯誤			
返回	-1	有錯誤, 並依據參數nCleanFlag決定是否清除錯誤		
值	1	無錯誤		
參數	int32_t nCleanFlag	0:不清除錯誤 非零:清除所有錯誤		

Ecr	EcmUsrDriver.cpp				
int32_t EcmSpiPackSizeCal()					
透過頭碼尾碼計算SPIDataSize					
返回	0	沒有偵測到頭尾碼,請檢查SPI傳輸			
直值	回 值 >0 SPI Data Size				



int32\_t SpiDataExchange(uint8\_t \*RetIdx, uint8\_t \*RetCmd)

上位機(PC)透過PCIe To SPI或USB介面與ECMXF進行資料交換, 主端將會送出一筆命令 的長度(XFU:1024位元組 / XF 預設144位元組, 資料長度可變更) 資料並取回從端回傳的資 料。

返 回 值	1	正確	
	0	CRC錯誤	
	<0	CRC連續錯誤次數達設定值	
參數	uint8_t *RetIdx	返回資料的索引號	
	uint8_t *RetCmd	返回資料的命令碼	

#### EcmUsrDriver.cpp

int32\_t EcmCmdTransceiver(int32\_t nTryCnt)

發送命令並接收回應,將會送出命令並取回該命令的回應。

呼叫本函式前應先將pCmd的內容填上必要欄位後,本函式會透過SpiDataExchange交換資料,取回來的資料會存放在pRet指標所指的空間。

驅動函式庫內函式大部分都是基於本函式實現, 實現流程如下:

- 1. 透過pCmd將命令封包的命令碼及必要欄位填上
- 2. 呼叫EcmCmdTransceiver給定錯誤重傳次數
- 3. 判斷返回值是否正確取得回應封包
- 4. 透過pRet將回應封包資料取回

返回	1	成功
值	0	超時
參 數	int32_t nTryCnt	錯誤嘗試次數

#### EcmUsrDriver.cpp

int32\_t EcmCmdTransceiver\_WaitAyncDone(int32\_t nTryCnt, uint16\_t \*pu16ReturnDataSize, uint8\_t \*pu8ReturnData)

發送命令並接收回應,將會送出命令並取回該命令的回應後,等待非同步命令完成後才返回。



本函 等行	本函式是由EcmCmdTransceiver為基礎, 當EcmCmdTransceiver返回後, 會輪詢ECM-XF 等待非同步命令完成或超時後返回。				
返回	1	成功			
值	0	超時			
	-1	資料成功取回但是有錯誤發生			
	-2	資料成功取回但是等待非同步命令完成超時			
	-3	參數錯誤			
參	int32_t nTryCnt	錯誤嘗試次數			
数	uint16_t *pu16ReturnDataSize	返回資料大小			
	uint8_t *pu8ReturnData	使用者提供空間及指標傳入,函式將儲存返回資料			

int32\_t EcmUniversalCmd( uint8\_t u8Cmd, uint8\_t u8Param, uint8\_t u8Param0, uint8\_t u8Param1, uint8\_t u8Param2, uint8\_t u8Param3, uint8\_t \*pu8Return, uint8\_t \*pu8Status, uint8\_t \*pu8ErrStatus, uint16\_t \*pu16ReturnDataSize, uint8\_t \*pu8ReturnData)

— 舟	一般命令,請參考XF命令					
返回	1	成功				
值	0	超時				
	-1	資料成功取回但是有錯誤發生				
	-2	資料成功取回但是等待非同步命令完成超時				
	-3	參數錯誤				
參數	uint8_t u8Cmd	命令代碼				
2	uint8_t u8Param	命令參數				
	uint8_t u8Param0	命令資料0				
	uint8_t u8Param1	命令資料1				
	uint8_t u8Param2	命令資料2				



	uint8_t u8Param3	命令資料3		
	uint8_t *pu8Return	回應的返回值		
	uint8_t *pu8Status	回應的ECAT狀態		
	uint8_t *pu8ErrStatus	回應的錯誤狀態		
	uint16_t *pu16ReturnDataSize	回應的Data大小		
	uint8_t *pu8ReturnData	回應的Data		

Ecr	EcmUsrDriver.cpp				
int32_t ECM_CheckDCStable()					
回傳	回傳DC Sync發生時間間隔是否穩定				
返回	1	DC Sync發生時間間隔趨近週期時間			
值	0	DC Sync發生時間間隔差異仍大			

EcmUsrDriver.cpp			
int32_t ECM_IsSlaveAlive(uint8_t *pEcmStatus, uint8_t *pRxPDOFifoCnt, uint8_t *CrcErrCnt, uint8_t *WkcErrCnt)			
確認	確認從站狀態		
返回	1	從站已完成配置且FIFO運作中	
值	0	非上述情況	
	-1	無法取得從站資料	
參動	uint8_t *pEcmStatus	返回從站狀態	
20	uint8_t *pRxPDOFifoCnt	返回RxPDOFIFO中已有的命令數量	
	uint8_t *CrcErrCnt	返回CRC錯誤次數計數值	
	uint8_t *WkcErrCnt	返回Working Counter錯誤計數值	



int32\_t ECM\_InfoUpdate(uint8\_t \*pEcmStatus, uint8\_t \*pRxPDOFifoCnt, uint8\_t \*CrcErrCnt, uint8\_t \*WkcErrCnt)

int32\_t ECM\_InfoUpdateCri(uint8\_t \*pEcmStatus, uint8\_t \*pRxPDOFifoCnt, uint8\_t \*CrcErrCnt, uint8\_t \*WkcErrCnt)

取回從站狀態, ECM\_InfoUpdateCri 不使用Critical Section

返回值	1	CRC正確,成功取回從站資料
	0	CRC錯誤
參數	uint8_t *pEcmStatus	返回從站狀態
	uint8_t *pRxPDOFifoCnt	返回RxPDOFIFO中已有的命令數量
	uint8_t *CrcErrCnt	返回CRC錯誤次數計數值
	uint8_t *WkcErrCnt	返回Working Counter錯誤計數值

Ecr	EcmUsrDriver.cpp		
int32_t ECM_SetCrcType(uint8_t u8Type)			
設置CRC的方式			
返 回 值	1	成功設置CRC Type	
	0	設置失敗	
參 數	uint8_t u8Type	0: 固定值檢查(0x12345678) 3: CRC-32	

void ECM_CloseLibrary()	
關閉並釋放驅動函式庫內記憶體空間	



uint8\_t ECM\_InitLibrary(uint16\_t \*pu16SpiDataSize)

初始化驅動函式庫內所需的變數及記憶體空間

返	0	失敗
值	Others	成功並返回晶片韌體版本號
參數	uint16_t *pu16SpiDataSize	因為PCIe卡是透過PCIe to SPI與晶片連接, SPI封包大 小是可以變更的, 故此參數僅對ECM-XF PCIe裝置有 效 0:不變更SPI封包資料段大小 32~1408:變更SPI封包資料段大小

EcmUsrDriver.cpp		
int3	2_t ECM_EcatInit(uint16_t DC	ActCode, uint32_t CycTime)
初如	台化EtherCAT網路, 取回從站資	訊,清除錯誤並配置從站。
返回	1	成功
值	0	超時
	-3	參數錯誤
參數	uint16_t DCActCode	DC AssignActivate code 例如: 0x0300 for DC Sync0 0x0000 for Free run
	uint32_t CycTime	DC週期時間(單位ns)

EcmUsrDriver.cpp		
int8_t ECM_EcatSlvCntGet()		
取回連線中從站數量		
返回值	-1	失敗
	>0	從站數量
	-3	參數錯誤



int32\_t ECM\_SetTxFIFOCnt(uint8\_t u8TxCnt)

設定ECM-XF內TxPDO的FIFO緩存最大筆數。 ECM-XF內建FIFO緩存空間,可確保PDO週期資料交換不會丟失資料,預設筆數為64筆。

返回值	參考EcmCmdTransceiver返回值	
參 數	uint8_t u8TxCnt	TxPDO的FIFO緩存最大筆數。

EcmUsrDriver.cpp			
int3	int32_t ECM_SetRxFIFOCnt(uint8_t u8TxCnt)		
設定ECM-XF內RxPDO的緩存筆數。 ECM-XF內建FIFO緩存空間, 可確保PDO週期資料交換不會丟失資料, 預設筆數為64筆。			
返回值	参考EcmCmdTransceiver返回值		
參數	uint8_t u8RxCnt	RxPDO的緩存筆數	

Ecr	EcmUsrDriver.cpp		
int3	int32_t ECM_GetTxFIFOCnt(uint8_t *pu8Cnt)		
取回	取回晶片內TxPDO的FIFO緩存最大筆數設定值。		
返回值	参考EcmCmdTransceiver返回值		
參 uint8_t *pu8Cnt 取回TxPDO的FIFO緩存最大筆數 數			

EcmUsrDriver.cpp

int32\_t ECM\_GetRxFIFOCnt(uint8\_t \*pu8Cnt)

取回晶片內RxPDO的FIFO緩存最大筆數設定值。



返回值	參考EcmCmdTransceiver返回值	
參 數	uint8_t *pu8Cnt	將取回RxPDO的FIFO緩存最大筆數

Utility.c			
int3	int32_t ECM_CheckMEMSpace(int32_t nPdoCntOnce)		
根據TxPDO及RxPDO的大小、設定最大筆數,判斷晶片內FIFO緩存空間是否足夠,也會 SPI/USB的資料封包大小是否足夠。			
返回	1	成功	
值	-1	讀取封包大小錯誤	
	-2	讀取PDO大小錯誤	
	-3	讀取FIFO最大筆數錯誤	
	-4	FIFO緩存空間不足	
	-5	資料封包大小不足	
參 數	int32_t nPdoCntOnce	每次資料交換會包含的PDO筆數	

int ECM\_EcatPdoFifoDataExchangeAdv(uint8\_t u8Op, uint8\_t u8Cnt, uint8\_t \*pRxData, uint8\_t \*pTxData, uint16\_t u16DataSize, uint8\_t \*pu8RxPdoFifoCnt, uint8\_t \*CrcErrCnt, uint8\_t \*WkcErrCnt, uint8\_t \*IsSIvAlive)

發送多筆RxPDO並接收多筆TxPDO		
返	-1	CRC錯誤
值	4	上次命令無寫入RxPDO到FIFO,也無從FIFO讀取到 TxPDO
	5	上次命令 從FIFO讀取到TxPDO成功
	6	上次命令 寫入RxPDO到FIFO成功
	7	上次命令 從FIFO讀取到TxPDO成功, 且寫入RxPDO到 FIFO成功



參 數	uint8_t u8Op	將執行的讀寫操作 bit0 : 寫入RxPDO到RxFIFO bit1 : 從TxFIFO讀取到TxPDO
	uint8_t u8Cnt	每次讀寫的PDO筆數
	uint8_t *pRxData	將此指標指向的RxPDO資料複製並送入FIFO
	uint8_t *pTxData	從FIFO讀取讀取TxPDO資料,並複製到此指標空間
	uint16_t u16DataSize	欲送出的RxPDO資料大小
	uint8_t *pu8RxPdoFifoCnt	透過此指標得知預存的RxPDO筆數
	uint8_t *CrcErrCnt	透過此指標取回SPI或USB資料錯誤次數
	uint8_t *WkcErrCnt	透過此指標取回EtherCAT資料錯誤次數
	uint8_t *IsSIvAlive	透過此指標得知從站是否斷線

int32\_t ECM\_InitFIFO(void)

初始化FIFO空間,使用FIFO前需要先初始化,晶片會根據設置的FIFO筆數及PDO大小配置 適合的空間。

返 參考EcmCmdTransceiver\_WaitAyncDone返回值定義 回 值

#### EcmUsrDriver.cpp

int32\_t ECM\_EnableFIFO(uint8\_t u8Enable)

始能FIFO, 當晶片進入SAFEOP或OP狀態後, 將會週期的從FIFO內取出RxPDO送出, 並將 收到的TxPDO存入。需先初始化**FIFO**空間。

返回值	參考EcmCmdTransceiver_WaitAyncDone返回值定義	
參 數	uint8_t u8Enable	0: DISABLE 1: ENABLE

#### EcmUsrDriver.cpp



int32_t ECM_ClearFIFO(uint8_t u8TxRx)		
清除FIFO內的資料。需先初始化 <i>FIFO</i> 空間。		
返回值	参考EcmCmdTransceiver_WaitAyncDone返回值定義	
參 數	uint8_t u8TxRx	0 : 清除TxPDO FIFO 及 RxPDO FIFO 1 : 僅清除TxPDO FIFO 2 : 僅清除RxPDO FIFO

EcmUsrDriver.cpp		
int32_t ECM_EcatStateSet(uint8_t u8Slave, uint8_t u8State)		
切換從站EtherCAT狀態		
返回值		
參數	uint8_t u8Slave	0~127 : 從站位置 255 : 全部從站
	uint8_t u8State	欲切換至的狀態 0x01: INIT 0x02: PRE_OP 0x03: BOOT 0x04: SAFE_OP 0x08: OPERATIONAL 0x10: Clear ERROR

Ecn	EcmUsrDriver.cpp		
int32_t ECM_EcatStateGet(uint8_t u8Slave, uint8_t *pu8State)			
讀取從站EtherCAT狀態			
返 回 值	參考EcmCmdTransceiver_WaitAyncDone返回值定義		
<del>參</del> 數	uint8_t u8Slave	0~127 : 從站位置 255 : 全部從站	



uint8	t *pu8State
_	

會將狀態值複製至此址標空間

EcmUsrDriver.cpp		
int32_t ECM_EcatSdoWrite(uint8_t Slave, uint16_t Index, uint8_t SubIndex, uint16_t size, int32_t Timeout, uint8_t *Data)		
透過SDO寫入從站物件。		
返回值	参考EcmCmdTransceiver_WaitAyncDone返回值定義	
物	uint8_t Slave	0~127 : 從站位置
뀘	uint16_t Index	索引號
	uint8_t SubIndex	副索引號
	uint16_t size	資料大小, 單位byte
	int32_t Timeout	超時等待時間, 單位ns
	uint8_t *Data	指標指向欲寫入資料的記憶體位置

EcmUsrDriver.cpp		
int32_t ECM_EcatSdoRead(uint8_t Slave, uint16_t Index, uint8_t SubIndex, uint16_t size, int32_t Timeout, uint8_t *Data)		
透過SDO讀取從站物件。		
返回值	參考EcmCmdTransceiver_WaitAyncDone返回值定義	
參	uint8_t Slave	0~127 : 從站位置
敫	uint16_t Index	索引號
	uint8_t SubIndex	副索引號
	uint16_t size	資料大小, 單位byte
	int32_t Timeout	超時等待時間, 單位ns


uint8 t \*Data

讀取的資料會複製至指標指向的記憶體位置

# 附錄 疑難排解

#### \*如何確認主站IC正常工作?

請依下列步驟確認

STEP1: 將ECM-XFU-SK J3 接至L (即XF IC Pin12拉低, Firmware update模式) 再上電, 綠燈滅, USB插上電腦可以找到裝置 => XF正常工作

STEP2: 上電時pin12拉高, 綠燈亮 => 模式選擇正常

STEP3: 開始例程後, 綠燈滅 => XF 收到正確命令(SPI MOSI/CLK/CS正常)

STEP4: 讀到版本號 => SPI讀寫正常

STEP5: 讀數slave數量 => PHY / RJ45 正常

#### \*接USB線後,電腦找不到裝置?

請確認IC型號為ECM-XFU, 若為ECM-XF則僅在Firmware Update模式支援USB通訊。

請確認USB線材,部分線材僅供充電,不具信號傳輸功能,請更換線材進行測試測試。

#### \*執行ITE測試程式後,出現「找不到VCRUNTIME140.dll」信息?

#### 請安裝Visual stdio 2017可轉發套件,參考網站

https://learn.microsoft.com/cpp/windows/latest-supported-vc-redist?view=msvc-170#visual-st udio-2015-2017-2019-and-2022

#### \* 參考電路上SPI接有100R電阻, 是什麼用途?

SPI上的100R電阻是阻抗匹配使用的,一般來說若SPI兩端距離很短且在同一板上,可以使用 0R取代。請注意,過大的電阻會造成訊號延遲,而使得SPI速度受限。



#### \* 為何12MHz無源振盪器不起振?

12MHz振盪器為USB功能使用, ECM-XF在一般模式下, 所接的12MHz無源振盪器不起振為 正常現象。

若IC型號為ECM-XFU或ECM-XF IC在Firmware Update模式時12MHz應起振, 若無請檢查電路或確認IC是否正常工作。

25MHz振盪器供給PHY做為網路參考時鐘使用,在所有情況下上電後應起振。



\*可以更換有源晶振嗎?

25M為PHY使用,可以更換為有源振盪器。

12M不可以, XF內部設定12M的晶振來源是無源振盪器, 當有使用到USB連線功能時會 參考12MHz的振盪器, 如下圖



# Crystal oscillator



#### \* 可以某些從站使用DC Sync, 某些從站使用Free Run嗎?

可以先使用ECM\_EcatInit 來設定大多數的從站DC狀態, 在配置完PDO並呼叫ReConfig後, 再使用ECM\_EcatSetDCSync (參照命令 ECM\_CMD\_ECAT\_DCSYNC)來設定個別從站, 並 指定特定的DC Code.可參考下圖藍框處。



### \* AL Status Code 是什麼意義?

AL Status Code是從站EtherCAT處理的錯誤資訊,通常在轉換狀態失敗後,可以透過 此代碼瞭解失敗的原因。若此代碼為0代表沒有錯誤,其他值的意義請參考Beckhoff網 站



https://infosys.beckhoff.com/english.php?content=../content/1033/ethercatsystem/10 37010571.html&id=

## \* AL Status Code 顯示0x1E是什麼意義?

根據Beckhoff網站所查詢到的AL Status Code 0x1E代表「SM configuration for input process data is invalid」, input process data 代表TxPDO的配置不被從站接受或是SM3(通常 SM3為input的SM, 可透過ESI檔確認)的配置錯誤。

1. 若有透過0x1C13來配置PDO, 請注意配置內容是否正確

2. 若使用預設的配置, 代表主站晶片讀取到錯誤的從站配置, 需手動指定, 指定方式如下:

STEP1:於ECM\_EcatInit之前指定強制指定模式 ECM\_SetEcatInitMode((uint8\_t)'0');

STEP2:於ECM\_EcatInit 之後 及 ECM\_EcatReconfig之後 強制指定SM的大小

ECM\_DirectAssign(uint8\_t u8Slv, uint8\_t u8Mode, uint16\_t u16Obits, uint16\_t u16Ibits)

第一個參數u8Slv為從站的順序站號,從0開始。

第二個參數u8Mode是否開啟強制指定,0代表使用從站預設值,1代表強制指定。

第三個參數u16Obits代表RxPDO的大小,單位是bit。

第四個參數u16lbits代表TxPDO的大小,單位是bit。

STEP3:在指定完全部的從站後(若從站能使用預設值則不須指定)呼叫 ECM\_EcatConfigMap();

若仍無法順利進到OP, 則可將STEP2換成ECM\_EcatConfigSM(uint8\_t slave, uint8\_t nSM, uint16\_t StartAddr, uint16\_t SMlength, uint8\_t ControlReg, uint8\_t Activate)

第一個參數slave為從站的順序站號,從0開始。

第二個參數uSM代表欲指定的SyncManager編號,請參考從站ESI檔案說明,通常output為2, input為3。

第三個參數StartAddr代表起始位置,請參考從站ESI檔案說明。

第四個參數SMlength代表SM的大小,單位是byte,請填入RxPDO(output)或TxPDO(input)的 大小,此值必須為從站ESI檔中該SM的最大值與最小值之間。

第五個參數ControlReg代表SM的控制碼,請參考從站ESI檔案說明。



第六個參數Activate代表SM是否啟用,請參考從站ESI檔案中Enable的值。

以下為ESI檔案中關於SM的說明。

<Sm MinSize="128" MaxSize="128" DefaultSize="128" StartAddress="#x1000" ControlByte="#x26" Enable="1">MBoxOut</Sm>
<Sm MinSize="128" MaxSize="128" DefaultSize="128" StartAddress="#x1080" ControlByte="#x22" Enable="1">MBoxIn</Sm>
<Sm MinSize="1" MaxSize="64" DefaultSize="8" StartAddress="#x1000" ControlByte="#x24" Enable="1">Outputs</Sm>
<Sm MinSize="1" MaxSize="64" DefaultSize="8" StartAddress="#x1000" ControlByte="#x24" Enable="1">Outputs</Sm>

#### 以下為設定範例



# \*執行時出現「無法載入DLL」「找不到指定的模組」「 DIINotFoundException」訊息。

請將「ECMPLATFORM.dll」及「EcmXFUDrv.dll」放入與執行檔相同的資料夾

# \*執行時出現「試圖載入格式錯誤的程式」「 BadImageFormatException」訊息。

請確認dll為32位元版本或是64位元版本, 兩者不相同

#### \* 如何透過SPI傳輸做到高度即時性?

ECM-XF內有TxFIFO用於暫存從站送回主站的資料,亦有RxFIFO用於暫存主站欲送至從站 的命令。在某些應用中,希望命令儘快送至從站,且從站的資料能儘速回到上位機中。此種應 用上位機必須具備高度即時性,能在中斷發生後立即處理SPI資料交換,具體說明如下: 1. 啟用INT1中斷,中斷來源可為BIT31收到封包時 或 BIT25 RxFIFO低極限 或 BIT24 TxFIFO 高極限

2. 若使用RxFIFO低極限或TxFIFO高極限, 需先透過ECM\_CMD\_ECAT\_SET\_FIFO\_TH(命令碼 70) 設定高/低極限

2. 當發生中斷時, 立刻進行SPI資料交換, 此時會收到最新的TxPDO, 並將下一週期的 RxPDO放入RxFIFO中。



3. SPI資料交換後的UserDelay延時可為0 (此定義在EcmUsrDriver.c中的SpiDataExchange 中), 或 TEST\_SPI\_IDLE\_TIME =0.

上述方式必須透過SPI傳輸及INT1的中斷來進行,透過USB傳輸受限於Windows作業系統為 非即時作業系統(non-real-time OS)無法實現高度即時性。

