ECM Starter Kit

ECM-SK

快速入门

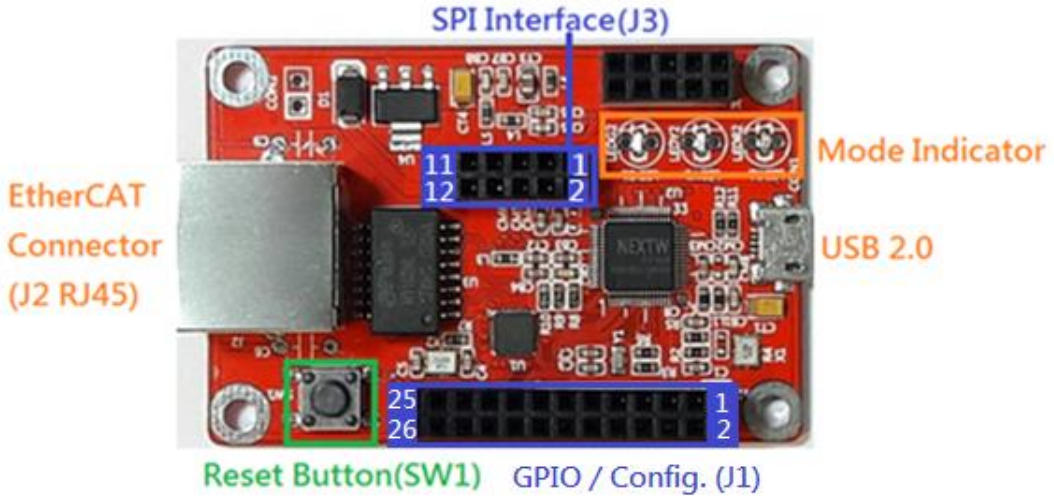
版本 ：V.1.7.4

日期 ：2022.05

目录

第 1 章 硬件脚位及使用说明

1.1 ECM-SK V1.0 脚位说明



1.1.1 GPIO / CONFIG (J1) 脚位说明

Pin 11	**Pin 9**	**Pin 7**	**Pin 5**	**Pin 3**	**Pin 1**
Input 5	Output 0	Input 3	Reserved	Reserved	3.3V Out
Pin 12	**Pin 10**	**Pin 8**	**Pin 6**	**Pin 4**	**Pin 2**
CONFIG0	Output 1	Input 1	Input 2	Reserved	Reserved

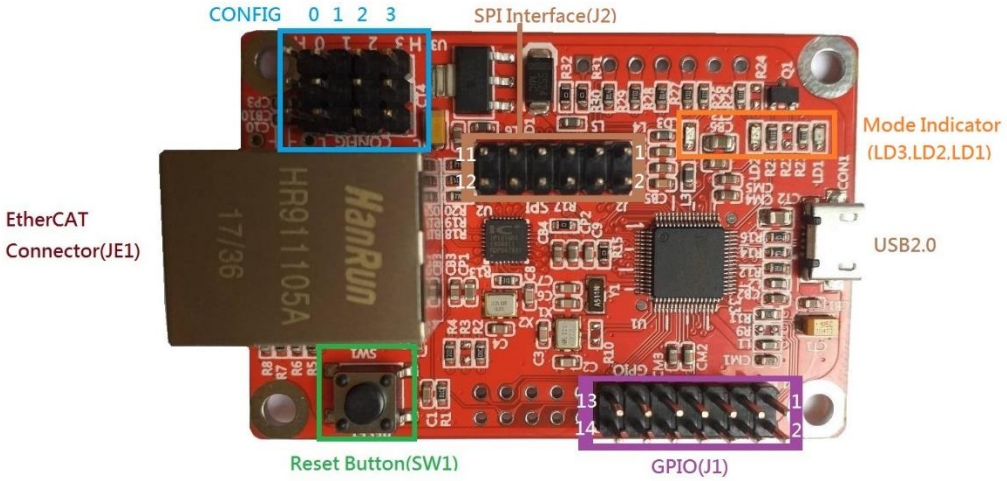
Pin 25	**Pin 23**	**Pin 21**	**Pin 19**	**Pin 17**	**Pin 15**	**Pin 13**
Reserved	Output 5	Output 4	CONFIG3	CONFIG2	Output 2	CONFIG1
Pin 26	**Pin 24**	**Pin 22**	**Pin 20**	**Pin 18**	**Pin 16**	**Pin 14**
GND	Reserved	Input 4	Input 0	Reserved	Busy (Output)	Output 3

* 注意：J1 Input / Output 脚位直接由 ECM IC 提供，请设计适当之隔离电路，以免 IC 损毁，3.3V 为 High，0V 为 Low。请参考 EC01M data sheet。

1.1.2 SPI(J3) 脚位说明

Pin 11	Pin 9	Pin 7	Pin 5	Pin 3	Pin 1
/RESET	SPI_MISO	SPI_/SS	Busy (Output)	Power In (3.3V)	Reserved
Pin 12	**Pin 10**	**Pin 8**	**Pin 6**	**Pin 4**	**Pin 2**
Reserved	SPI_MOSI	SPI_CLK	Reserved	Reserved	GND

1.2 ECM-SK V1.1 脚位说明



1.2.1 GPIO (J1) 脚位说明

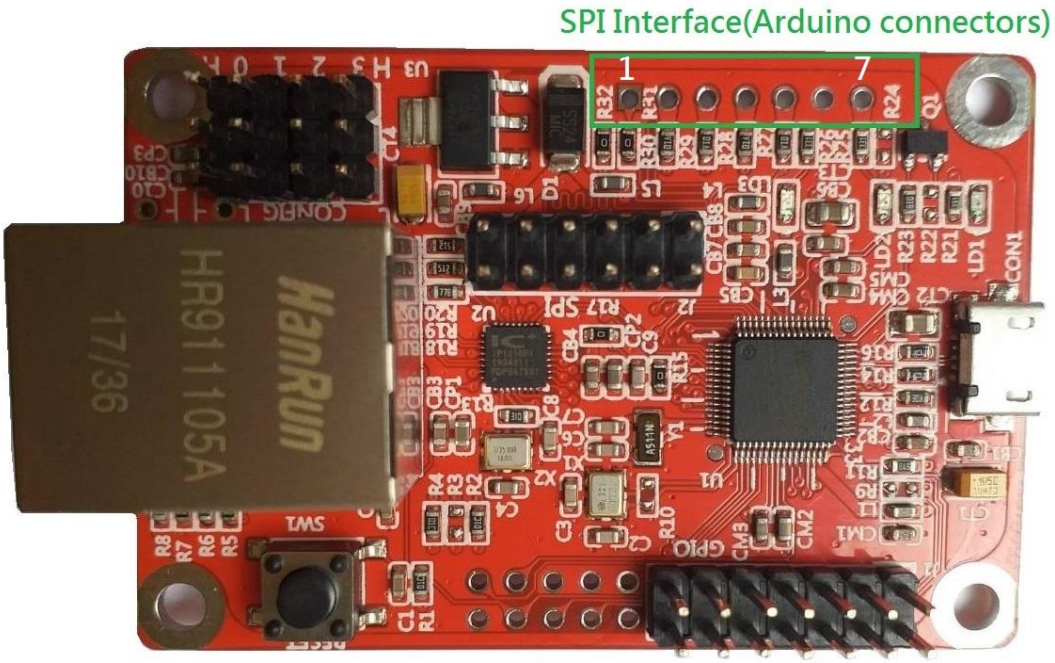
Pin 13	Pin 11	Pin 9	Pin 7	Pin 5	Pin 3	Pin 1
Input 5	Input 4	Input 3	Input 2	Input 1	Input 0	3.3V Out
Pin 14	**Pin 12**	**Pin 10**	**Pin 8**	**Pin 6**	**Pin 4**	**Pin 2**
Output 5	Output 4	Output 3	Output 2	Output 1	Output 0	GND

* 注意：J1 Input / Output 脚位直接由 ECM IC 提供，请设计适当之隔离电路，以免 IC 损毁，3.3V 为 High，0V 为 Low。请参考 EC01M data sheet。

1.2.2 SPI(J2) 脚位说明

Pin 11	Pin 9	Pin 7	Pin 5	Pin 3	Pin 1
/RESET	SPI_MISO	SPI_/SS	Busy (Output)	Power In (3.3V)	Reserved
Pin 12	**Pin 10**	**Pin 8**	**Pin 6**	**Pin 4**	**Pin 2**
Reserved	SPI_MOSI	SPI_CLK	Reserved	Reserved	GND

1.2.3 SPI(Arduino Connectors) 脚位说明



Pin 1	Pin 2	Pin 3	Pin 4	Pin 5	Pin 6	Pin 7
Busy (Output)	SPI_/SS	SPI_MOSI	SPI_MISO	SPI_CLK	GND	Power In (3.3V)

* 注意：SPI(Arduino Connectors)与 SPI(J2)作用相同，仅能择一操作

* Pin7 Power In: 建议提供 3.3V 200mA 以上的电流

1.3 LED 状态显示及 Config.设定说明

1.3.1 LED 指示灯号说明

V1.0 组件编号	V1.1 组件编号	颜色	意义
LEDR1	LD1	红色	电源指示灯
LEDY1	LD2	黄色	状态指示灯 Y
LEDG1	LD3	绿色	状态指示灯 G

1.3.2 LED 状态说明

状态指示灯 Y（黄色）	状态指示灯 G（绿色）	状态
亮 ON	亮 ON	未侦测到 EtherCAT 子站 或 子站未达预期阶段
暗 OFF	暗 OFF	Init State 或 Pre-Operational State
亮 ON	暗 OFF	Safe-Operational State
暗 OFF	亮 ON	Operational State

1.3.3 网络口灯号说明

网络口灯号 Y（黄色）：恒亮代表网络速度 100M Hz，网络口正常工作。
网络口灯号 G（绿色）：闪烁代表数据传输。

1.3.4 CONFIG Pin 说明

CONFIG	状态	说明
CONFIG0	L	Host interface is set as USB
	X	Host interface is set as SPI (Default)
	H	
CONFIG1	X	Normal mode (Default)
	L	
	H	Test mode
CONFIG2	X	Data size of each slave is 12 Bytes (Default)
	L	
	H	Data size of each slave is 16 Bytes
CONFIG3	L	FIFO abandon disable
	X	FIFO abandon enable (Default)
	H	

X: Floating, L: Low (0V), H: High (3.3V)
CONFIG0 与上位控制器链接的接口选择
CONFIG1 模式选择 Normal Mode – 依据用户的命令进行控制
Test Mode – 上电后自动依默认值(Slave Type: IO, Cycle Time: 1ms)进入 OP，并周期性传送不同的 output 命令给所有从站

CONFIG2 单一站的数据数选择

CONFIG3 单笔命令最长传输时间选择

L: disable 单笔命令传输时间无限制

H: enable 单笔命令传输时间最大为 100 个周期时间，超过则放弃整笔命令

1.4 系统连接图

最短通讯周期 250us

最短通讯周期 1000us

* 由 J1 Pin12 CONFIG0 决定 SPI 模式 或 USB 模式

* USB 通讯速度取决于 USB 2.0 Host 效能

1.5 ECM-SK SPI 与控制器间的接线

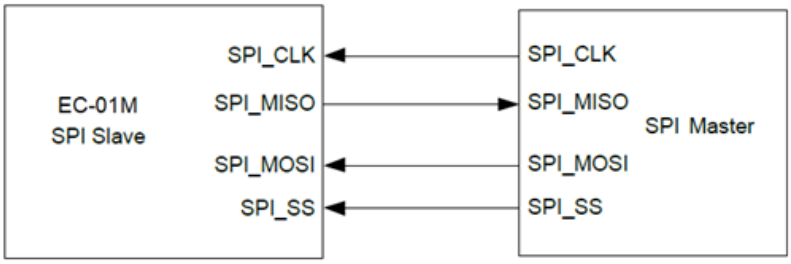


表 1.1 SPI 接脚名称及意义

名称	SPI 脚位	意义	说明
Busy	Pin5	ECM 忙碌	ECM 正处于忙碌状态，不接受新的 SPI 传输

SCLK	Pin8	频率讯号	由 SPI Master 产生并控制，依控制 EtherCAT 通讯周期有最低频率要求
MOSI	Pin10	主出从入	SPI Master 数据输出，SPI Slave 数据输入
MISO	Pin9	主入从出	SPI Master 数据输入，SPI Slave 数据输出
/SS	Pin7	芯片致能	选择信号，由 Master 控制，Slave 只有在 /SS 信号为低电位时，才会对 Master 的操作指令有反应

1.6 SPI 规格与传输模式

ECM 与上位控制器间采用 SPI 通讯格式，ECM 与上位控制器间的接线在低频率时(10MHz 以下)可用杜邦线直接连接，在高频率时建议直接以银线焊接方式连接，以免噪声干扰通讯质量，SCLK 由上位控制器提供，依不同通讯周期有最低频率要求，SCLK 最高支持至 24MHz，支持 40 子站(每个子站含 12-Byte 数据)，或支持 30 个子站(每个子站含 16-Byte 数据)，通讯周期 SCLK 需求，如表 1.2 所示。

表 1.2 SPI SCLK 最低要求

通讯周期(Cycle Time)	SCLK 最低要求	接线方式
250us	24 MHz	银线焊接或 PCB（不可用杜邦线）
1ms	6 MHz	可用杜邦线

SPI 模式传输说明

ECM 的 SPI 为 Slave 模式，空闲时为低电位，在下降缘发送，并于上升缘接收，高位数据先传（MSB），请参考下图说明。

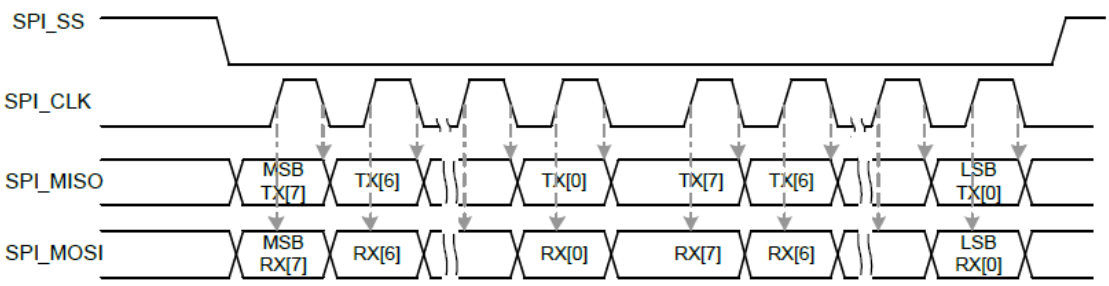


图 1.1 SPI 时序图

上位控制器端的 SPI 为 Master 模式，需产生 CLK 并提供给 SPI Slave，并在上升缘发送，并于下降缘接收。 SPI 传输以 Byte 为单位，每次将由低地址开始传，依序传至最高地址，换言之 SPI 传输将从 Byte0 开始，再依序传 Byte1、Byte2…直至最后一个 Byte 为止。而 SPI 传输单一 Byte 时，采取 MSB 模式，亦即高位先传输。

1.7 SPI 传输时序说明

上位控制器可透过 SPI 与 ECM 沟通，上位控制器侦测到 SPI BUSY(J1 Pin16 或 J3 Pin5)为低电位时，可开始进行 SPI 通讯，当进入 SPI 通讯后，SPI BUSY 讯号立刻升为高电位，直至通讯结束后且 ECM 处理完命令封包后 SPI Busy 讯号才会恢复为低电位，之后控制器即可透过 SPI 取得数据或传送命令。详细时序图如下所示：

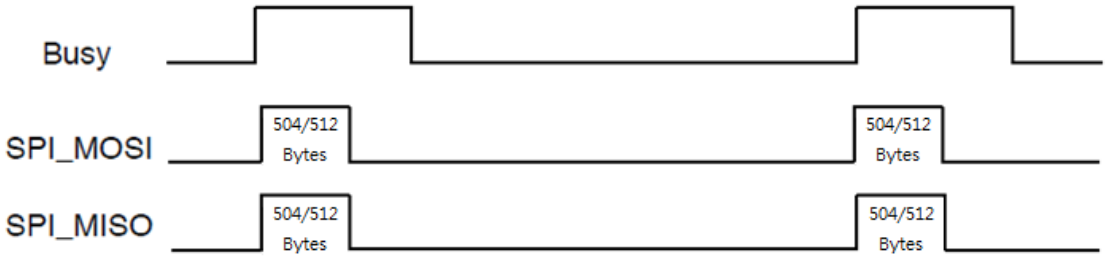


图 1.2 信号时序示意图

若欲实现最短周期 250us 的通讯要求，SPI 传输频率必须为 24MHz，且上位控制端侦测到 Busy Pin 为低电位时，必须立即进行下一笔数据的交换。若通讯周期较慢的应用（如慢于 1ms），则上位控制端可于要进行数据传输时，再判断 BUSY 的讯号，若 Busy Pin 为低电位时即可进行 SPI 数据交换。

1.8 EtherCAT 状态流程

EtherCAT 可分为 4 个执行状态，Init、PreOP、SafeOP、OP，各状态能执行的指令不同，在 PreOP 状态可以进行子站类型及驱动器类型的设置；SafeOP 状态可进行非周期性数据交换，如读写缓存器数据(SDO)；OP 状态则可进行周期性数据交换，即在固定周期内交换固定数据，OP 状态亦可进行非周期性数据交换。使用者只需在 PreOP 状态及 OP 状态即可完成所有指令操作。

EtherCAT 状态间的切换时间会因 EtherCAT 子站而有不同（与子站厂牌、性能、数量有关），用户可从 GET_STATUS 命令的回传值中 Current State 确认目前状态，详细流程与可执行的指令如下图所示：

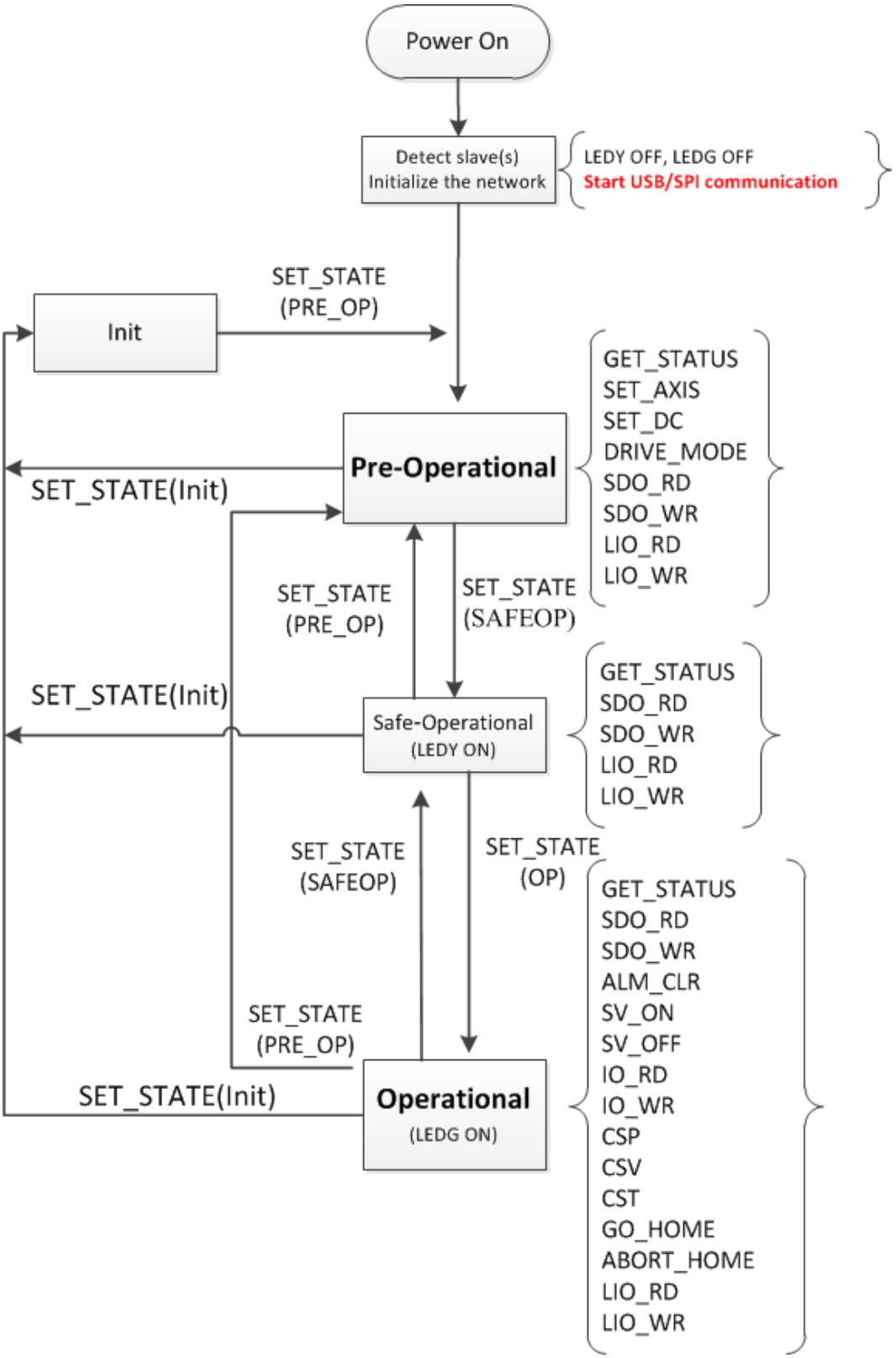


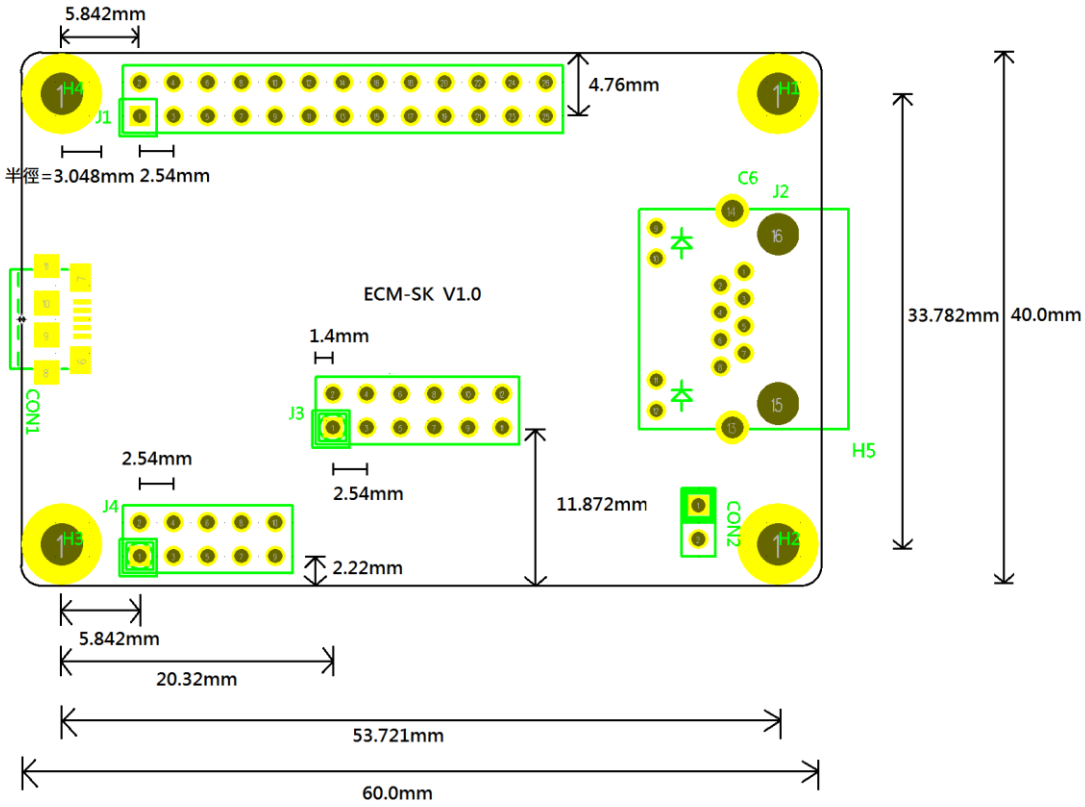
图 1.3 EtherCAT 状态流程

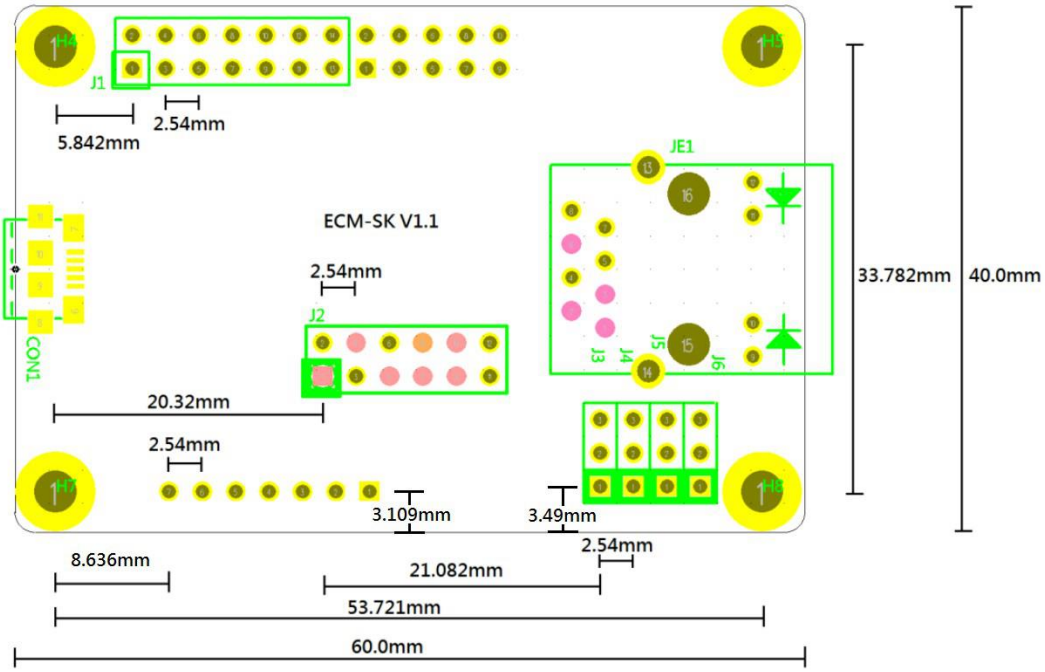
表 1.3 指令列表

Cmd. ID	Command	Operation	ECM	Drive	IO	Init	PreOP	SafeOP	OP	Ref.
0x00	GET_STATUS	Get Status	V	V	V	V	V	V	V	2.1.1
0x01	SET_STATE	Set EtherCAT state	V			V	V	V	V	2.1.2
0x02	SET_AXIS	Set type of slave	V			V	V			2.1.3
0x03	SET_DC	Set DC Mode	V			V	V			2.1.4
0x04	SET_EX	Set Extensions (CRC)	V			V	V	V		2.1.5
0x05	SET_FIFO	Set FIFO	V						V	2.1.6
0x06	DRIVE_MODE	Set mode of servo drive		V		V	V			2.1.7
0x07	SDO_RD	Service data object read		V	V		V	V	V	2.1.8
0x08	SDO _WR	Service data object write		V	V		V	V	V	2.1.9
0x10	ALM_CLR	Alarm clear		V					V	2.1.10
0x11	SV_ON	Servo ON		V					V	2.1.11
0x12	SV_OFF	Servo OFF		V					V	2.1.12
0x13	IO_RD	Digital Input			V				V	2.1.13
0x14	IO_WR	Digital Output			V				V	2.1.14
0x15	CSP	Position control		V					V	2.1.15
0x16	CSV	Velocity control		V					V	2.1.16
0x17	CST	Torque control		V					V	2.1.17
0x18	GO_HOME	Start Homing Procedure		V					V	2.1.18
0x19	ABORT_HOME	Abort Homing Procedure		V					V	2.1.19
0x21	LIO_RD	Read ECM IC input	V			V	V	V	V	2.1.20

0x22	LIO_WR	Write ECM IC output	V			V	V	V	V	2.1.21
0xBB	SW_RESET	Software Reset	V			V	V	V	V	2.1.22

1.9 尺寸及固定孔位置

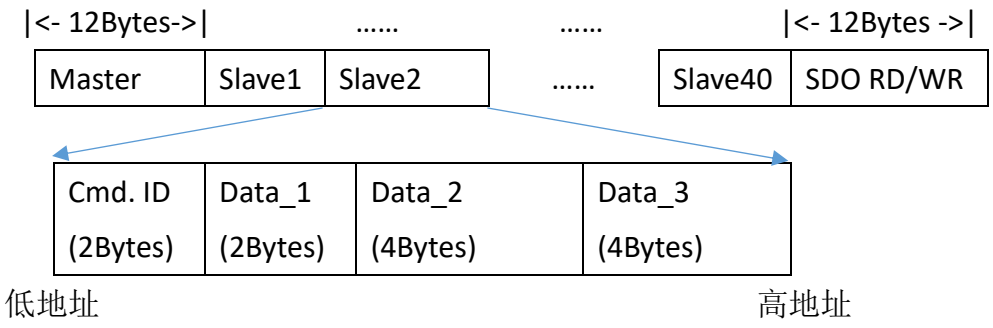

5.842mm
4.76mm
J1
半徑=3.048mm 2.54mm
C6
J2
ECM-SK V1.0
1.4mm
J3
CON1
2.54mm
2.54mm
J4
11.872mm
CON2
2.22mm
H5
33.782mm
40.0mm
5.842mm
20.32mm
53.721mm
60.0mm


J1
2.54mm
5.842mm
JE1
ECM-SK V1.1
2.54mm
J2
CON1
20.32mm
2.54mm
3.109mm
3.49mm
2.54mm
33.782mm
40.0mm
8.636mm
21.082mm
53.721mm
60.0mm

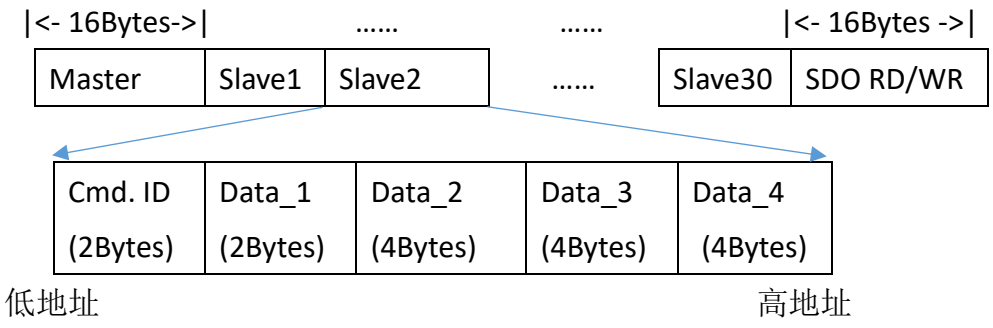
第 2 章 命令与回应

2.1 命令与响应数据结构

上位控制器可在不同的通信阶段对各子站下达命令来控制或取得各子站的状态，子站收到并处理后会响应，每一个命令依 CONFIG 2(J1 Pin17)之设定，为 12Bytes 或 16Bytes 的形式，第一笔命令对应 ECM，第二笔命令对应第一个子站，第三笔命令对应第二个子站，以此类推。数据长度若为 12Bytes 可控制 40 个子站，若为 16Bytes 则可控制 30 个子站，最后一笔命令则是专门对应单一子站进行非周期性的参数读写(SDO Read/Write)。在 CONFIG 2 为 Low 时主站最多可连接 40 个子站，因此每次下达命令均会传递(40+2) * 12 =504 Bytes 的数据，亦会得到(40+2) * 12 =504 Bytes 回应。



若 CONFIG 2 为 High 时主站最多可连接 30 个子站，因此每次下达命令均会传递(30+2) * 16 =512 Bytes 的数据，亦会得到(30+2) * 16 =512 Bytes 回应。



若数据存放方式为 Little Endian，低字节放在低地址，例如 Cmd. ID 为 2Byte 数据，假设值为 0x0001 时，低字节 0x01 会放在 Byte0，高字节 0x00 会放在 Byte1，又

如 Data_2 为 4Byte 数据，假设值为 0x87 65 43 21，最低字节值 0x21 会放在最低地址 Byte0，之后依序为 0x43、0x65，最高字节 0x87 则放在最高地址 Byte3。

SPI 传输以 Byte 为单位，每次将由低地址开始传，依序传至最高地址，换言之 SPI 传输将从 Byte0 开始，再依序传 Byte1、Byte2...直至最后一个 Byte 为止。而 SPI 传输单一 Byte 时，采取 MSB 模式，亦即高位先传输。

此次命令的结果，将于下次命令时传回，但部分指令执行可能超过一个指令周期(如 SET_STATE、SDO_RD、SDO_WR、GO_HOME 等)，完成后的结果可在数个周期后利用 GET_STATUS 命令取回。

2.1.1 GET_STATUS Command

- 指令说明：Get Status，不做任何命令，仅取回状态数据
- Command ID : 0x0000
- 适用阶段：所有阶段
- 适用类别：所有类别

表 2.1 GET_STATUS Command

BYTE	字段定义	Command	Response
0	Command ID	0x0000	0x00 或 0xBF
1			
2	Reserved	--	Pram
3	CRC8 Value**	CRC8 Value	
4	Reserved	--	Data1
5			
6			
7			
8	Reserved	--	Data2
9			
10			
11			
12	Reserved*	--	Data3
13			
14			
15			

* Byte12-15 仅在 CONFIG 2 为 High 时才会生效。

** CRC8 Value 算法请参考 2.2 响应数据 关于 CRC 的部分，是否开启 CRC 检查可透过 2.1.5 SET_EX 来设定。

提示： Response 的 Byte 0 于 EtherCAT OP 状态下回应为 0xBF，其余状态回应为 0x00，Byte1~Byte11 各字段将保持上次命令结果(State 非为 OP)或更新数据(State 为 OP)，响应数据可参考 2.2 响应数据。

2.1.2 SET_STATE Command

- 指令说明：设定 EtherCAT 状态
- Command ID : 0x0001
- 适用阶段：所有阶段
- 适用类别：ECM(第一笔命令)

表 2.2 SET_STATE Command

BYTE	字段定义	Command	Response
0	Command ID	0x0001	0x01
1			Error Code
2	Reserved	--	Current State
3	CRC8 Value**	CRC8 Value	CRC8 Value
4	EtherCAT State -INIT: 0x01 -PRE_OP: 0x02 -SAFE_OP: 0x04 -OP: 0x08	Requested STATE	--
5			
6			
7			
8	Reserved	--	--
9			
10			
11			
12	Reserved*	--	--
13			
14			
15			

* Byte12-15 仅在 CONFIG 2 为 High 时才会生效。

** CRC8 Value 算法请参考 2.2 响应数据 关于 CRC 的部分，是否开启 CRC 检查可透过 2.1.5 SET_EX 来设定。

Error Code CRC 错误次数，因 CRC 错误而被拒绝的命令数

Current State

-INIT: 0x01

-PRE_OP: 0x02

-SAFE_OP: 0x04

-OP: 0x08

提示：变更 EtherCAT State 需耗时数个运算周期，且与总子站数量有关，可对 ECM 下达 GET_STATUS 指令，并取得 Current State 信息。

2.1.3 SET_AXIS Command

- 指令说明：Set type of slave，设定 ECM 连接之子站型态(IO、Drive、HSP 或 STEP)，最多 40 轴
- Command ID : 0x0002
- 适用阶段：Pre-Operational State
- 适用类别：ECM(第一笔命令)

表 2.3 SET_AXIS Command

BYTE	字段定义	Command	Response
0	Command ID	0x0002	0x02
1			Error Code
2	Group(0~4)	Group (Default: 0)	Current State
3	CRC8 Value**	CRC8 Value	CRC8 Value
4	Topology -Drive : 0x0 -IO : 0x1 -HSP : 0x2 -STEP : 0x3	Topology (Default:0x1)	SlaveCount -侦测到的子站数
5			
6			
7			
8	Reserved	--	--
9			
10			
11			
12	Reserved*	--	--
13			
14			

15			

* Byte12-15 仅在 CONFIG 2 为 High 时才会生效。

** CRC8 Value 算法请参考 2.2 响应数据 关于 CRC 的部分，是否开启 CRC 检查可透过 2.1.5 SET_EX 来设定。

Group

一颗 ECM 最多可控制 40 个子站(Drive、IO、NEXTW HSP 或 STEP，不支持其他子站种类)，一次可设定 8 子站型态，Group 0 代表设定子站 1~8，Group 1 代表设定子站 9~16，依此类推，Group 的值可为 0~4。

SlaveCount

侦测到的子站数，若于 Init State 一律回传 0，于 Pre-Operational State 会回传实际侦测到的子站数量。

Topology

Bit28	Bit24	Bit20	Bit16	Bit12	Bit8	Bit4	Bit0
Slave n+8	Slave n+7	Slave n+6	Slave n+5	Slave n+4	Slave n+3	Slave n+2	Slave n+1

n = Group * 8

0: Drive

1: IO

2: NEXTW HSP (High Speed Pulse)

3: Step

Error Code CRC 错误次数，因 CRC 错误而被拒绝的命令数

Current State

-INIT: 0x01

-PRE_OP: 0x02

-SAFE_OP: 0x04

-OP: 0x08

2.1.4 SET_DC Command

- 指令说明：设定 ECM 周期时间(Cycle Time)
- Command ID : 0x0003
- 适用阶段：Pre-Operational State
- 适用类别：ECM(第一笔命令)

表 2.4 SET_DC Command

BYTE	字段定义	Command	Response
0	Command ID	0x0003	0x03
1			Error Code
2	Reserved	--	Current State
3	CRC8 Value**	CRC8 Value	CRC8 Value
4	Cycle Time (us) 须为 250 x 2^n (Minimum : 250)	Cycle Time (Default:1000)	--
5			
6			
7			
8	**FIX Value**	**0xFFFF** (Default:0xFFFF)	--
9			
10			
11			
12	Reserved*	--	--
13			
14			
15			

* Byte12-15 仅在 CONFIG 2 为 High 时才会生效。

** CRC8 Value 算法请参考 2.2 响应数据 关于 CRC 的部分，是否开启 CRC 检查可透过 2.1.5 SET_EX 来设定。

FIX Value

固定值，一律为 0x0000FFFF

Error Code CRC 错误次数，因 CRC 错误而被拒绝的命令数

Current State

-INIT: 0x01

-PRE_OP: 0x02

-SAFE_OP: 0x04

-OP: 0x08

2.1.5 SET_EX Command

- 指令说明：设定 CRC 功能 (设定完成后于下次开机后生效)
- Command ID : 0x0004
- 适用阶段：Init State、Pre-Operational State、Safe-Operational State
- 适用类别：ECM(第一笔命令)

表 2.5 SET_EX Command

BYTE	字段定义	Command	Response
0	Command ID	0x0004	0x04
1			Error Code
2	Reserved	--	Current State
3			CRC8 Value
4	Extension Setting 1~2	Extension Setting Value	--
5			
6			
7			
8	Reserved	--	--
9			
10			
11			
12	Reserved*	--	--
13			
14			
15			

* Byte12-15 仅在 CONFIG 2 为 High 时才会生效。

* **此指令将于下次开机后生效**

Extension Setting

Bit28	Bit24	Bit20	Bit16	Bit12	Bit8	Bit4	Bit0
Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Setting2	Setting1

Setting 1: Command CRC Verification

0: Disable Command CRC Verification

1: Enable Command CRC Verification

Setting 2: Response CRC Verification

0: Disable Response CRC Verification

1: Enable Response CRC Verification

2.1.6 SET_FIFO Command

- 指令说明：设定 FIFO
- Command ID : 0x0005
- 适用阶段：Operational State
- 适用类别：ECM(第一笔命令)

表 2.6 SET_FIFO Command

BYTE	字段定义	Command	Response
0	Command ID	0x0005	0x05
1			Error Code
2	Action	Action(1~2)	Current State
3	CRC8 Value	CRC8 Value	CRC8 Value
4	Value	Value (>=0)	--
5			
6			
7			
8	Reserved	--	--
9			
10			
11			
12	Reserved*	--	--
13			
14			
15			

* Byte12-15 仅在 CONFIG 2 为 High 时才会生效。

Action

0x0001 : Clear FIFO content and FIFO minimum setting.

清除 FIFO 内所有数据，并清除 FIFO threshold 设定

0x0002 : Set FIFO threshold to **Value**.

设定 FIFO 阀值为 Value。**OP 状态时**，当 FIFO 剩余空间小于 Value 后，才开始在每周期取一笔数据执行，满足条件后自动将最小数值重设为 160

2.1.7 DRIVE_MODE Command

- 指令说明：Set mode of servo drive，设定 ECM 连接之伺服驱动器命令模式及同步模式
- Command ID : 0x0006
- 适用阶段：Pre-Operational State
- 适用类别：Drive 子站、NEXTW HSP 子站、STEP 子站

表 2.7 DRIVE_MODE Command

BYTE	字段定义	Command	Response
0	Command ID	0x06	0x06
1		Command Index	Command Index
2	Reserved	--	--
3			
4	OP_Mode -CSP: 0x08 -CSV: 0x09 -CST: 0x0A	OP_Mode (Default:0x08)	--
5			
6			
7			
8	DRIVE_TYPE - FREERUN: 0x00 - DCSYNC: 0x01	DRIVE_TYPE (Default:0x01)	--
9			
10			
11			
12	Reserved*	--	--
13			
14			
15			

* Byte12-15 仅在 CONFIG 2 为 High 时才会生效。

- Command Index：用户自定义之数字，命令与响应的 Command Index 会相同。

- OP_MODE：
  CSP(Cyclic Synchronous Position Mode)：

ECM在CSP模式下定期发送PDO，在传送每一笔PDO时，均会将**目标绝对位置**命令同时传送至子站。

CSV(Cycle Synchronized Velocity Mode)：

ECM在CSV模式下定期发送PDO，在传送每一笔PDO时，会将**目标速度**命令同时传送至子站。

CST (Cyclic Synchronous Torque Mode)

ECM在CSV模式下定期发送PDO，在传送每一笔PDO时，会将目标扭力命令同时传送至子站。

- DRIVE_TYPE:

FREE RUN

各子站间异步，各子站间根据自己的内部时间处理EtherCAT资料，与主站的周期、其他子站的周期及EtherCAT数据到达时间均无关。

DCSYNC

DCSYNC要求主站有很强的实时性能，是高精度的时间同步模式（所有子站与第一个有DC的子站同步），以第一个有DC的子站时间为基准时间，再用此基准时间作为所有子站的参考时间，加上传输延时、抖动等时间误差产生同步信号。

2.1.8 SDO_RD Command

- 指令说明：SDO (Service Data Objects) Read，设定 ECM 对指定的子站下达欲读取的 Object Index 及 Object Sub Index。
- Command ID : 0x0007
- 适用阶段：Pre-Operational State、Safe-Operational State、Operational State
- 适用类别：SDO RD/WR (最后一笔命令)

表 2.8 SDO _RD Command

BYTE	字段定义	Command	Response
0	Command ID	0x0007	0x0007
1			
2	Slave Index	Slave Index(1~40)	Slave Index
3			
4	Object Index	Object Sub Index	Object Sub Index
5			

6		Object Index	Object Index
7			
8	Value	--	Value
9			
10			
11			
12	Reserved*	--	--
13			
14			
15			

* Byte12-15 仅在 CONFIG 2 为 High 时才会生效。

Index

Bit16	Bit0
Object Index	Object Sub Index

提示：SDO_RD 需耗时，可在数个周期后对 SDO 下达 GET_STATUS 指令，取回结果。

2.1.9 SDO_WR Command

- 指令说明：SDO (Service Data Objects) Write，设定 ECM 对指定的子站 Object Index 及 Object Sub Index 写入指定数值。
- Command ID : 0x0008
- 适用阶段：Pre-Operational State、Safe-Operational State、Operational State
- 适用类别：SDO RD/WR (最后一笔命令)

表 2.9 SDO_WR Command

BYTE	字段定义	Command	Response
0	Command ID	0x0008	0x0008
1			
2	Slave Index	Slave Index (1~40)	Slave Index
3	Size	Object Size	
4	Object Index	Object Sub Index	Object Sub Index
5			
6		Object Index	Object Index
7			

8	Value	Object Value	Object Value
9			
10			
11			
12	Reserved*	--	--
13			
14			
15			

* Byte12-15 仅在 CONFIG 2 为 High 时才会生效。

Index

Bit16	Bit0
Object Index	Object Sub Index

提示：SDO_WR 需耗时，可在数个周期后对 SDO 下达 GET_STATUS 指令取回状态，以确认写入。

2.1.10 ALM_CLR Command

- 指令说明：Clear alarm of slave，清除子站上的警告讯息
- Command ID : 0x10
- 适用阶段：Operational State
- 适用类别：Drive 子站、NEXTW HSP 子站、STEP 子站

表 2.10 ALM_CLR Command

BYTE	字段定义	Command	Response
0	Command ID	0x10	0x10
1		Command Index	Command Index
2	Reserved	--	Statusword
3			
4	Reserved	--	--
5			
6			
7			
8	Reserved	--	--
9			

10			
11			
12	Reserved*	--	--
13			
14			
15			

* Byte12-15 仅在 CONFIG 2 为 High 时才会生效。

Command Index 使用者自定义的 Index 值，可藉此值判断该次响应是针对哪个命令

提示：并非所有的 Alarm 均可以被清除，可以清除的警告请参阅子站说明手册。

2.1.11 SV_ON Command

- 指令说明：Set Servo ON，设定驱动器子站 Servo ON
- Command ID : 0x11
- 适用阶段：Operational State
- 适用类别：Drive 子站、NEXTW HSP 子站、STEP 子站

表 2.11 SV_ON Command

BYTE	字段定义	Command	Response
0	Command ID	0x11	0x11
1		Command Index	Command Index
2	Reserved	--	Statusword
3			
4	Reserved	--	--
5			
6			
7			
8	Reserved	--	--
9			
10			
11			
12	Reserved*	--	--
13			
14			
15			

* Byte12-15 仅在 CONFIG 2 为 High 时才会生效。

Command Index 使用者自定义的 Index 值，可藉此值判断该次响应是针对哪个命令

提示：SV_ON 指令会在一个通讯周期送达子站，但每个子站进行 Servo on 的程序及所需时间可能不同，使用者可透 GET_STATUS 指定取得最新的 Statusword，Statusword 可判断子站 Servo on/off 状态。

2.1.12 SV_OFF Command

- 指令说明：Set Servo OFF，设定驱动器子站 Servo OFF
- Command ID : 0x12
- 适用阶段：Operational State
- 适用类别：Drive 子站、NEXTW HSP 子站、STEP 子站

表 2.12 SV_OFF Command

BYTE	字段定义	Command	Response
0	Command ID	0x12	0x12
1		Command Index	Command Index
2	Reserved	--	Statusword
3			
4	Reserved	--	--
5			
6			
7			
8	Reserved	--	--
9			
10			
11			
12	Reserved*	--	--
13			
14			
15			

* Byte12-15 仅在 CONFIG 2 为 High 时才会生效。

Command Index 使用者自定义的 Index 值，可藉此值判断该次响应是针对哪个命令

提示：SV_OFF 指令会在一个通讯周期送达子站，但每个子站进行 Servo off 所需时间可能不同，使用者可透 GET_STATUS 指定取得最新的 Statusword，Statusword 可判断子站 Servo on/off 状态。

2.1.13 IO_RD Command

- 指令说明：Read digital input value，读取子站数字输入状态数值
- Command ID : 0x13
- 适用阶段：Operational State
- 适用类别：IO 子站

表 2.13 IO_RD Command

BYTE	字段定义	Command	Response
0	Command ID	0x0013	0x13
1			--
2	Reserved	--	--
3			
4	Input - 子站上的输入状态值	--	Input
5			
6			
7			
8	Reserved	--	--
9			
10			
11			
12	Reserved*	--	--
13			
14			
15			

* Byte12-15 仅在 CONFIG 2 为 High 时才会生效。

2.1.14 IO_WR Command

- 指令说明：Write digital output value，设定子站输出状态
- Command ID : 0x14
- 适用阶段：Operational State
- 适用类别：IO 子站

表 2.14 IO_WR Command

BYTE	字段定义	Command	Response

0	Command ID	0x0014	0x14
1		Command Index	Command Index
2	Reserved	--	--
3			
4	Output - 子站上的输出状态值	Output	--
5			
6			
7			
8	Reserved	--	--
9			
10			
11			
12	Reserved*	--	--
13			
14			
15			

* Byte12-15 仅在 CONFIG 2 为 High 时才会生效。

Command Index 使用者自定义的 Index 值，可藉此值判断该次响应是针对哪个命令

2.1.15 CSP Command

- 指令说明：Cyclic Synchronous Position Command，设定子站之周期性同步位置命令
- Command ID : 0x15
- 适用阶段：Operational State
- 适用类别：OP_Mode 为 CSP 的 Drive 或 HSP 或 STEP 子站

表 2.15 CSP Command for Drive and 1 channel HSP

BYTE	字段定义	Command	Response	Master Cyclic Response
0	Command ID	0x15	0x15	0xBF
1		Cmd. Index	Cmd. Index	Error Code
2	Statusword – Drive 子站状态	--	Statusword	Current State
3				CRC8 Value
4	Target Position – 目标位置	Target Position (绝对位置)	Current Position	--
5				

6 7	Current Position – 目前位置			
8 9 10 11	ALM Code - Alarm code 警告代码 Current Torque – 目前扭力值	Output Value	Current Torque **	FIFO Remaining FIFO 尚余空间
			ALM Code**	FIFO Full Count FIFO 空间满而被拒绝的命令数
12 13 14 15	Reserved*	--	--	--

* Byte12-15 仅在 CONFIG 2 为 High 时才会生效。

** HSP(High Speed Pulse)子站回传 Input 状态，且无 ALM Code，Drive 子站回传 Current Torque 及 ALM Code。

Command Index 使用者自定义的 Index 值，可藉此值判断该次响应是针对哪个命令

Error Code CRC 错误次数，因 CRC 错误而被拒绝的命令数

Current State 应为 OP: 0x08

Output Value 设定 Drive 或 HSP 上的 Output (部分 Drive 可能不支持)

提示：CSP 指令为设定子站目标绝对位置，建议用户自行规划加减速，并且至少在一个通讯周期内送出一个 CSP 命令。用户可以在一个通讯周期内送出两个或以上的 CSP 命令，尚未送至子站的命令会暂存于 FIFO 中，FIFO 的空间有限，请注意，**若 FIFO 已无空间存放则会直接忽略该命令**，使用者应检查 Master 响应中的 FIFO Remaining，若所剩余空间过小，建议暂停数个通讯周期后，再恢复命令的传送。

表 2.16 CSP Command for 2-channel HSP

BYTE	字段定义	Command	Response	Master Cyclic Response
0	Command ID	0x15	0x15	0xBF
1		Cmd. Index	Cmd. Index	Error Code
2	Statusword – Drive 子站状态	--	Statusword	Current State
3				CRC8 Value
4 5 6	Target Position - Pulse 输出的目标绝对位置	Target Position 1 (绝对位置)	Current Position 1	--

7	Current Position - Encoder 回传的目前位置 Output Value – 输出值 Input Status – 输入状态			
8		Output Value 1	Input Status 1	FIFO Remaining FIFO 尚余空间
9				
10		Target Position 2(绝对位置)	Current Position 2	FIFO Full Count FIFO 空间满而被拒绝的命令数
11				
12				
13				
14		Output Value 2	Input Status 2	
15				

提示：若要输出两组 Pulse，CONFIG 2 必须为 High。

2.1.16 CSV Command

- 指令说明：Cyclic Synchronous Velocity Command，设定子站之周期性同步速度命令
- Command ID : 0x16
- 适用阶段：Operational State
- 适用类别：OP_Mode 为 CSV 的 Drive 或 HSP 或 STEP 子站

表 2.17 CSV Command for Drive and 1 channel HSP

BYTE	字段定义	Command	Response	Master Cyclic Response
0	Command ID	0x16	0x16	0xBF
1		Cmd. Index	Cmd. Index	Error Code
2	Statusword – Drive 子站状态	--	Statusword	Current State
3				CRC8 Value
4	Velocity–目标速度 Current Position – 目前位置	Velocity	Current Position	--
5				
6				
7				
8	ALM Code - Alarm code 警告代码 Current Torque – 目前扭力值	--	Current Torque **	FIFO Remaining FIFO 尚余空间
9				
10			ALM Code**	FIFO Full Count FIFO 空间满而被拒绝的命令数
11				

12	Reserved*	--	--	--
13				
14				
15				

* Byte12-15 仅在 CONFIG 2 为 High 时才会生效。

** HSP(High Speed Pulse)子站回传 Input 状态，且无 ALM Code，Drive 子站回传 Current Torque 及 ALM Code。

Command Index 使用者自定义的 Index 值，可藉此值判断该次响应是针对哪个命令

Error Code CRC 错误次数，因 CRC 错误而被拒绝的命令数

Current State 应为 OP: 0x08

提示：CSV 指令为设定子站目标转速，建议用户自行规划加减速度，并且至少在一个通讯周期内送出一个 CSV 命令。用户可以在一个通讯周期内送出两个或以上的 CSV 命令，尚未送至子站的命令会暂存于 FIFO 中，FIFO 的空间有限，请注意，**若 FIFO 已无空间存放则会直接忽略该命令**，使用者应检查 Master 响应中的 FIFO Remaining，若所剩余空间过小，建议暂停数个通讯周期后，再恢复命令的传送。

表 2.18 CSV Command for 2 channel HSP

BYTE	字段定义	Command	Response	Master Cyclic Response
0	Command ID	0x16	0x16	0xBF
1		Cmd. Index	Cmd. Index	Error Code
2	Statusword – Drive 子站状态	--	Statusword	Current State
3				CRC8 Value
4	Target Velocity -Pulse 输出的目标速度	Target Velocity 1	Current Position 1	--
5				
6				
7				
8	Current Position - Encoder 回传的目前位置	Output Value 1	Input Status 1	FIFO Remaining FIFO 尚余空间
9				
10		Target Velocity 2	Current Position 2	FIFO Full Count FIFO 空间满而被拒绝的命令数
11				
12	Output Value – 输出值			
13				
14	Input Status –	Output Value	Input Status 2	

15	输入状态	2		

提示：若要输出两组 Pulse，CONFIG 2 必须为 High。

2.1.17 CST Command

- 指令说明：Cyclic Synchronous Torque Command，设定子站之周期性同步扭力命令
- Command ID : 0x17
- 适用阶段：Operational State
- 适用类别：OP_Mode 为 CST 的 Drive 子站

表 2.19 CST Command

BYTE	字段定义	Command	Response	Master Cyclic Response
0	Command ID	0x17	0x17	0xBF
1		Cmd. Index	Cmd. Index	Error Code
2	Statusword – Drive 子站状态	--	Statusword	Current State
3				CRC8 Value
4	Torque – 目标扭力 Current Position – 目前位置	Torque	Current Position	
5				
6				
7				
8	ALM Code - Alarm code 警告代码 Current Torque – 目前扭力值	--	Current Torque	FIFO Remaining FIFO 尚余空间
9				
10			ALM Code	FIFO Full Count FIFO 空间满而被拒绝的命令数
11				
12	Reserved*	--	--	--
13				
14				
15				

* Byte12-15 仅在 CONFIG 2 为 High 时才会生效。

Command Index 使用者自定义的 Index 值，可藉此值判断该次响应是针对哪个命令
Error Code CRC 错误次数，因 CRC 错误而被拒绝的命令数
Current State 应为 OP: 0x08

提示：CST 指令为设定子站的目标扭力，建议用户自行规划连续的扭力输出，并且至少在一个通讯周期内送出一个 CST 命令。用户可以在一个通讯周期内送出两个或以上的 CST 命令，尚未送至子站的命令会暂存于 FIFO 中，FIFO 的空间有限，请注意，**若 FIFO 已无空间存放则会直接忽略该命令**，使用者应检查 Master 响应中的 FIFO Remaining，若所剩余空间过小，建议暂停数个通讯周期后，再恢复命令的传送。

2.1.18 GO_HOME Command

- 指令说明：设定子站开始回 Home 程序
- Command ID : 0x18
- 适用阶段：Operational State
- 适用类别：Drive 子站、NEXTW HSP 子站、STEP 子站

表 2.20 GO_HOME Command

BYTE	字段定义	Command	Response
0	Command ID	0x18	0x18
1		Command Index	Command Index
2	Statusword – Drive 子站状态	--	Statusword
3			
4	Current Position – 目前位置	--	Current Position
5			
6			
7			
8	ALM Code - Alarm code 警告代码	--	Current Torque
9			
10	Current Torque – 目前扭力值		ALM Code
11			
12	Reserved*	--	--
13			
14			
15			

* Byte12-15 仅在 CONFIG 2 为 High 时才会生效。

Command Index 使用者自定义的 Index 值，可藉此值判断该次响应是针对哪个命令

提示 1：执行 GO_ HOME 前须透过 SDO_WR 来指定操作模式，请参考子站使用说明，并确认 Homing Method 种类

Index	SubIndex	Name	Size	Value	Description
6060h	0	Modes of Operation	1 Byte	6	Homing Mode
6098h	0	Homing Method	1 Byte	0~35	Set Homing Method

另亦可透过 SDO_WR 来设定 Homing 相关参数，包括但不限于：

Index	SubIndex	Name	Size
607Ch	0	Home Offset	4 Bytes
6099h	1	Speed during search for switch	4 Bytes
6099h	2	Speed during search for zero	4 Bytes
609Ah	0	Homing Acceleration	4 Bytes

提示 2：GO_ HOME 完成后会自动将操作模式切回原设定模式(CSP / CSV / CST)

提示 3：HSP 第二轴相关设定位置为第一轴 Index + 0x800(详细内容请参考 NEXTW HSP 使用手册)

提示 4：使用 GO_ HOME 指令后，必须等待所有子站 GO_ HOME 完成（或终止）后，才能进行下一个 GO_ HOME 指令。

2.1.19 ABORT_HOME Command

- 指令说明：强制终止**所有子站** Home 程序
- Command ID : 0x19
- 适用阶段：Operational State
- 适用类别：Drive 子站、NEXTW HSP 子站、STEP 子站

表 2.21 ABORT_HOME Command

BYTE	字段定义	Command	Response
0	Command ID	0x19	0x19
1		Command Index	Command Index
2	Statusword – Drive 子站状态	--	Statusword
3			
4	Current Position – 目前位置	--	Current Position
5			
6			
7			
8	ALM Code - Alarm code 警告代码	--	Current Torque
9			
10	Current Torque – 目前扭力值		ALM Code
11			

12	Reserved*	--	--
13			
14			
15			

* Byte12-15 仅在 CONFIG 2 为 High 时才会生效。

Command Index 使用者自定义的 Index 值，可藉此值判断该次响应是针对哪个命令

提示 1：子站回传的 Statusword bit10 及 bit12 可判断回 Home 的状态，当 bit10 及 bit12 均为 1 时，代表回 Home 程序已经完成。透过 ABORT_HOME 指令可强制所有子站终止 Home 程序。

提示 2：ABORT_HOME 会终止所有子站的 Home 程序，不限于单一子站。

2.1.20 LIO_RD Command

- 指令说明：读取 ECM IC 上的 Input 状态
- Command ID : 0x0021
- 适用阶段：所有阶段
- 适用类别：ECM(第一笔命令)

表 2.22 LIO_RD Command

BYTE	字段定义	Command	Response
0	Command ID	0x0021	0x21
1			Error Code
2	Reserved	--	Current State
3	CRC8 Value**	CRC8 Value	CRC8 Value
4	Input Status – Input 状态	--	Input Status
5			
6			
7			
8	Reserved	--	--
9			
10			
11			
12	Reserved*	--	--
13			
14			
15			

* Byte12-15 仅在 CONFIG 2 为 High 时才会生效。
注意：LIO 脚位直接由 ECM IC 提供，请设计适当之隔离电路，以免 IC 损毁，3.3V 为 High，0V 为 Low。
** CRC8 Value 算法请参考 2.2 响应数据 关于 CRC 的部分，是否开启 CRC 检查可透过 2.1.5 SET_EX 来设定。

2.1.21 LIO_WR Command

- 指令说明：设定 ECM IC 上的 Output 状态
- Command ID : 0x0022
- 适用阶段：所有阶段
- 适用类别：ECM(第一笔命令)

表 2.23 LIO_WR Command

BYTE	字段定义	Command	Response
0	Command ID	0x0022	0x22
1			Error Code
2	Reserved	--	Current State
3	CRC8 Value**	CRC8 Value	CRC8 Value
4	Output Value–Output 值	Output Value	--
5			
6			
7			
8	Reserved	--	--
9			
10			
11			
12	Reserved*	--	--
13			
14			
15			

* Byte12-15 仅在 CONFIG 2 为 High 时才会生效。
注意：LIO 脚位直接由 ECM IC 提供，请设计适当之隔离电路，以免 IC 损毁，3.3V 为 High，0V 为 Low。
** CRC8 Value 算法请参考 2.2 响应数据 关于 CRC 的部分，是否开启 CRC 检查可透过 2.1.5 SET_EX 来设定。

2.1.22 SW_RESET Command

- 指令说明：Reset，回到初始状态
- Command ID : 0x00BB
- 适用阶段：所有阶段
- 适用类别：ECM(第一笔命令)

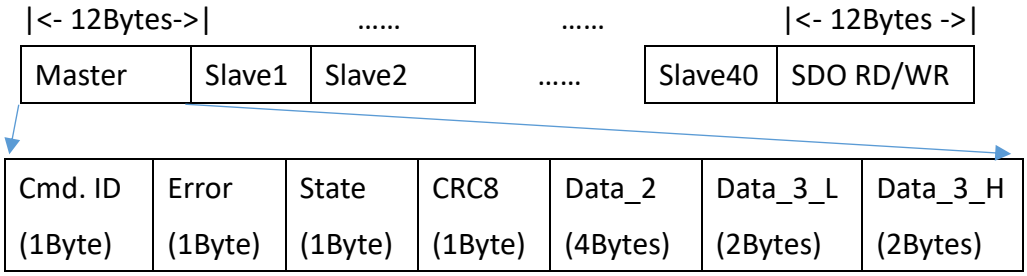
表 2.24 SW_RESET Command

BYTE	字段定义	Command	Response
0	Command ID	0x00BB	--
1			
2	Reserved	--	--
3			
4	Reserved	--	--
5			
6			
7			
8	Reserved	--	--
9			
10			
11			
12	Reserved*	--	--
13			
14			
15			

* Reset 后将重置所有的变量，已经开启的 USB 联机会中断并关闭。要继续传输必须再重新开启 USB 联机。

2.2 响应数据

每次下达命令后均会得到响应数据，由于数据传递周期的关系，当次收到响应数据为上次命令后的执行结果响应，部分命令(如 SET_STATE、SDO_RD、SDO_WR 等)会经过数个指令周期后响应，响应数据的前 12 Bytes 为主站信息，主站信息的前 4 Bytes 固定为 Cmd. ID、Error Code、Current State 及 CRC8 Value，当 CONFIG 2 为 Low 时，每子站响应资料为 12Byte，其代表意义分述如下：



Cmd. ID

若对 Master 下达的指令会回传该指令代码(如 SET_STATE、SET_AXIS、SET_DC 等)，若 State 为 OP 状态则会回传 0xBF

Error CRC 错误次数，因 CRC 错误而被拒绝的命令数

State 目前主站的运行时间

-INIT: 0x01

-PRE_OP: 0x02

-SAFE_OP: 0x04

-OP: 0x08

CRC8 循环冗余校验值(Cyclic Redundancy Check)

该次回传信息的 CRC8 校验值，验算方式如下：

STEP 1：先将 CRC8 值记录起来，并将该字段填 0

STEP 2：将回传值共 504 个 Byte(或 512 个 Byte)依序带入以下 CRC8 多项式

CRC-8： $X^8 + X^2 + X + 1$，初值设为 0x5A

STEP 3：算出 CRC8 值，此值应与 STEP 1 记录的值相符

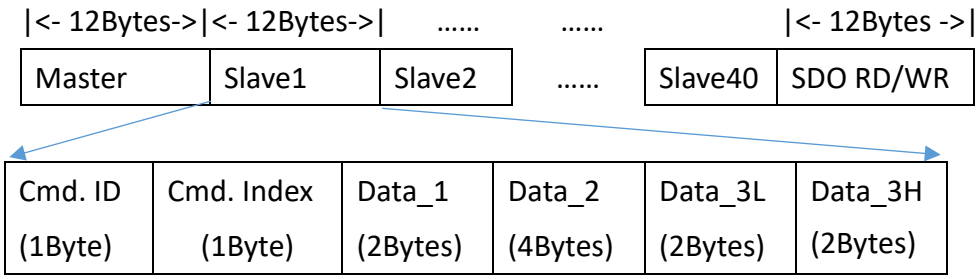
此值是否传送可透过 **2.1.5 SET_EX** 命令来设定

Data_2 显示响应数据(依命令不同代表不同意义)

Data_3_L 在 OP 状态下显示剩余 FIFO 命令缓冲空间

Data_3_H 在 OP 状态下显示因已无命令缓冲空间而被拒绝没有执行的命令数量

响应数据的第 13 Byte 开始至第 492Byte 为 40 个子站信息，每个子站信息为 12 个 Byte，驱动器子站信息的前 2 Bytes 固定为 Cmd. ID 及 Cmd. Index，其代表意义分述如下：



Cmd. ID 对该子站下达的指令会回传该指令代码
Cmd. Index 指此数据为针对哪个 Command Index 命令的响应

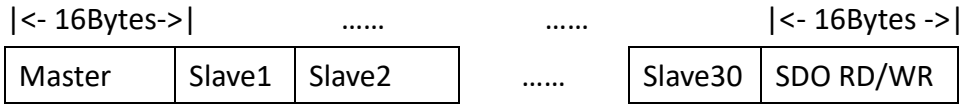
Data_1 在 OP 状态下显示驱动器子站的 Status Word(状态字)，Status Word 为 EtherCAT CoE Object 0x6041 的标准，相关定义请参考子站使用说明

Data_2 在 OP 状态下显示驱动器子站的 Current Position(目前位置)，此为 EtherCAT CoE Object 0x6064 的标准

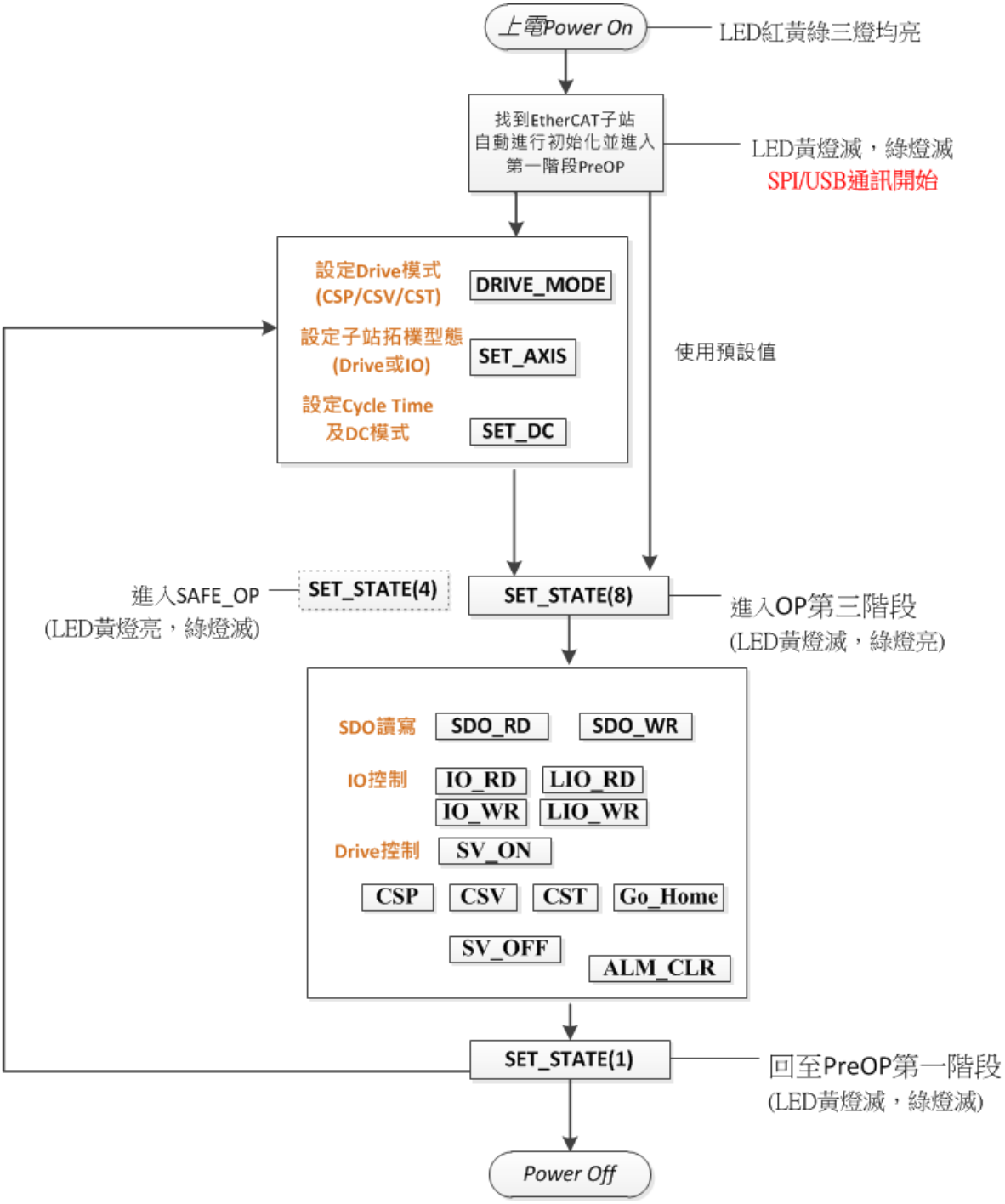
Data_3_L 在 OP 状态下显示驱动器子站的 Current Torque(目前扭力)，此为 EtherCAT CoE Object 0x6077 的标准

Data_3_H 在 OP 状态下显示驱动器子站的 Error Code(错误代码)，此为 EtherCAT CoE Object 0x603F 的标准

当 CONFIG 2 为 High 时，响应信息均为 16Bytes，前 12Bytes 与上述相同，目前仅 NEXTW HSP 子站在 CSP/CSV 命令时，会使用增加的 4Bytes 空间。



第 3 章 流程范例


上電Power On
LED紅黃綠三燈均亮
找到EtherCAT子站
自動進行初始化並進入
第一階段PreOP
LED黃燈滅，綠燈滅
SPI/USB通訊開始
設定Drive模式
(CSP/CSV/CST)
DRIVE_MODE
設定子站拓樸型態
(Drive或IO)
SET_AXIS
設定Cycle Time
及DC模式
SET_DC
使用預設值
進入SAFE_OP
(LED黃燈亮，綠燈滅)
SET_STATE(4)
SET_STATE(8)
進入OP第三階段
(LED黃燈滅，綠燈亮)
SDO讀寫
SDO_RD
SDO_WR
IO控制
IO_RD
LIO_RD
IO_WR
LIO_WR
Drive控制
SV_ON
CSP
CSV
CST
Go_Home
SV_OFF
ALM_CLR
SET_STATE(1)
回至PreOP第一階段
(LED黃燈滅，綠燈滅)
Power Off

第 4 章 动态函式库

4.1 动态函式库简介

「NEXTWUSBLib.dll」为 ECM-SK 使用 USB 做为接口，在 Windows 操作系统中的动态函式库，支持语言包含 C++、Visual Basic .NET、C# .NET 等，NEXTWUSBLib.dll 须与执行档.exe 放置相同目录。「NEXTWUSBLib.lib」为开发 C++程序时的参考文件，「NEXTWUSB_dotNET.dll」为开发微软.NET 程序时的参考函式原型，程序开发时加入项目中的参考。

4.2 NEXTWUSBLib 函式库

NEXTWUSBLib 函式库支援 C++、Visual Basic .NET、C# .NET 等.NET 系列程序语言，因在程序运行时间被参考，故须放置于执行文件相同目录，表 4.1 为「NEXTWUSBLib.dll」动态函式库主要功能。

表 4.1 「NEXTWUSBLib.dll」主要功能列表

函式名称	说明	参考
OpenECMUSB()	开启与 ECM-SK 间的 USB 联机	4.2.1
CloseECMUSB()	关闭与 ECM-SK 间的 USB 联机	4.2.2
ECMUSBWrite()	透过 USB 写入数据至 ECM-SK	4.2.3
ECMUSBRead()	透过 USB 读取数据从 ECM-SK	4.2.4

4.2.1 OpenECMUSB 函式说明

函数声明　bool OpenECMUSB()

传入参数　无

回传参数　回传一个 bool 值，指出开启联机是否成功

说明　使用本函式来开启与 ECM-SK 间的 USB 联机，若回传值为 False，有可能有下列原因：

- ECM-SK 并未连接到本机端的任一 USB 埠
- ECM-SK CONFIG 0 未正确设定

- ECM-SK 找不到任何 EtherCAT Slave
- ECM-SK 已经被其他程序开启
- USB 埠的供电不足，导致 ECM-SK 无法正常运作

4.2.2 CloseECMUSB 函式说明

函数声明	void CloseECMUSB()
传入参数	无
回传参数	无
说明	使用本函式来关闭与 ECM-SK 间的 USB 联机，并释放联机资源

4.2.3 ECMUSBWrite 函式说明

函数声明	ECMUSBWrite(unsigned char * data, unsigned long dwlength)	
传入参数	data	指针形态，指向 unsigned char 数组，该数组中存放欲写入的数据
	dwlength	4 bytes unsigned long，指出欲写入的资料 byte 数，由于每次交换数据为 504 或 512bytes，此值应为 504 或 512
回传参数	回传一个 bool 值，指出写入资料是否成功	
说明	使用本函式来透过 USB 写入数据至 ECM-SK，若回传值为 False，有可能有下列原因：	

- ECM-SK 未开启，请先呼叫 OpenECMUSB 以开启联机
- 已开启的联机失效，可能为 USB 接触不良或已移除

4.2.4 ECMUSBRead 函式说明

函数声明	ECMUSBRead(unsigned char * data, unsigned long dwlength)	
传入参数	data	指针形态，指向 unsigned char 数组，读取的数据会存放于该数组中
	dwlength	4 bytes unsigned long，指出欲读出的资料 byte 数，由于每次交换数据为 504 或 512bytes，此值应为 504 或 512
回传参数	回传一个 bool 值，指出读取资料是否成功	

说明	使用本函式来透过 USB 从 ECM-SK 读取数据，若回传值为 False，有可能有下列原因：

- ECM-SK 未开启，请先呼叫 OpenECMUSB 以开启联机
- 已开启的联机失效，可能为 USB 接触不良或已移除

4.3 NEXTWUSB_dotNET 函式库

NEXTWUSB_dotNET 函式库支援 Visual Basic .NET、C# .NET 等.NET 系列程序语言，提供.NET 环境下与 ECM-SK 相关的常数定义、结构与函式，提供更为直觉的开发流程，此函式库在设计时间需加入项目中的参考，运行时间仍会参考「NEXTWUSBLib.dll」，故「NEXTWUSBLib.dll」*须*放置于执行文件相同目录，而「NEXTWUSB_dotNET_XXB.dll」文件则*不须*放置于执行文件相同目录，表 4.2 为「NEXTWUSB_dotNET_XXB.dll」动态函式库主要功能，表 4.3 为「NEXTWUSB_dotNET_12B.dll」动态函式库命令结构。

表 4.2 「NEXTWUSB_dotNET_XXB.dll」主要常数列表

群组	常数名称	形态	常数值	说明
最大轴数	DEF_MA_MAX	Int32	42	NEXTWUSB_dotNET_12B.dll 最多 1 ECM、40 Slaves 及 1 组 SDO RD/WR
			32	NEXTWUSB_dotNET_16B.dll 最多 1 ECM、30 Slaves 及 1 组 SDO RD/WR
EthetCAT 状态	NIC_INIT	Int32	0	初始化网络阶段
	STATE_INIT	Int32	1	EtherCAT Init 阶段
	STATE_PRE_OP	Int32	2	EtherCAT PreOP 阶段
	STATE_SAFE_OP	Int32	4	EtherCAT SafeOP 阶段
	STATE_OPERATIONAL	Int32	8	EtherCAT OP 阶段
Drive 模式	CSP_MODE	Int32	8	CSP 模式
	CSV_MODE	Int32	9	CSV 模式
	CST_MODE	Int32	10	CST 模式

DC 模式	FREERUN	Int32	0	无 DC 同步
	DCSYNC	Int32	1	DC 同步
子站类别	DRIVE	Int16	0x0	驱动器类型子站
	IO	Int16	0x1	IO 类型子站
	HSP	Int16	0x2	HSP 类型子站
	STEP	Int16	0x3	STEP 类型子站
	None	Int16	0xF	无子站
指令代码	GET_STATUS	Int16	0x00	取回状态
	SET_STATE	Int16	0x01	设定 ECM 状态
	SET_AXIS	Int16	0x02	设定各子站类别
	SET_DC	Int16	0x03	设定周期时间及 DC 模式
	SET_EX	Int16	0x04	设定 CRC 功能
	SET_FIFO	Int16	0x05	设定 FIFO
	DRIVE_MODE	Int16	0x06	设定驱动器模式
	SDO_RD	Int16	0x07	读取参数
	SDO_WR	Int16	0x08	写入参数
指令代码	ALM_CLR	Int16	0x10	清除警告
	SV_ON	Int16	0x11	激磁
	SV_OFF	Int16	0x12	退磁
	IO_RD	Int16	0x13	读取 IO 子站 Input 状态
	IO_WR	Int16	0x14	设定 IO 子站 Output 状态
	CSP	Int16	0x15	CSP 周期指令
	CSV	Int16	0x16	CSV 周期指令
	CST	Int16	0x17	CST 周期指令
	GO_HOME	Int16	0x18	开始回 Home 程序
	ABORT_HOME	Int16	0x19	终止回 Home 程序
指令代码	LIO_RD	Int16	0x21	读取 ECM 上的 Input 状态
	LIO_WR	Int16	0x22	设定 ECM 上的 Output 状态
指令代码	SW_RESET	Int16	0xBB	软件重置

表 4.3 「NEXTWUSB_dotNET_12B.dll」命令结构

结构名称	成员变量 名称	形态
transData	CMD	Int16
	Pram	Int16
	Data1	Int32
	Data2	Int32

「NEXTWUSB_dotNET_12B.dll」含有两个主要的公用变量 cmdData 及 respData，均为 transData 结构数组，长度均为 42，可存放 1 个 ECM 及 40 个子站的数据，再加上一笔 SDO RD/ WR 专用命令位置。表 4.4 为「NEXTWUSB_dotNET_16B.dll」动态函式库命令结构。

表 4.4 「NEXTWUSB_dotNET_16B.dll」命令结构

结构名称	成员变量 名称	形态
transData	CMD	Int16
	Pram	Int16
	Data1	Int32
	Data2	Int32
	Data3	Int32

「NEXTWUSB_dotNET_16B.dll」含有两个主要的公用变量 cmdData 及 respData，均为 transData 结构数组，长度均为 32，可存放 1 个 ECM 及 30 个子站的数据，再加上一笔 SDO RD/ WR 专用命令位置，表 4.5 为「NEXTWUSB_dotNET_XXB.dll」动态函式库主要函式。

表 4.5 「NEXTWUSB_dotNET_XXB.dll」动态函式库主要函式

函式名称	说明	参考
OpenECMUSB()	开启与 ECM-SK 间的 USB 联机	4.3.1
CloseECMUSB()	关闭与 ECM-SK 间的 USB 联机	4.3.2
ECMUSBWrite()	透过 USB 写入数据至 ECM-SK	4.3.3
ECMUSBRead()	透过 USB 读取数据从 ECM-SK	4.3.4
ClearCmdData()	清除指令	4.3.5

4.3.1 OpenECMUSB 函式说明

函数声明	bool OpenECMUSB()
传入参数	无
回传参数	回传一个 bool 值，指出开启联机是否成功
说明	使用本函式来开启与 ECM-SK 间的 USB 联机，若回传值为 False，有可能有下列原因： ● ECM-SK 未连接到本机端的任一 USB 埠 ● ECM-SK CONFIG 0 未正确设定 ● ECM-SK 找不到任何 EtherCAT Slave ● ECM-SK 已经被其他程序开启 ● USB 埠的供电不足，导致 ECM-SK 无法正常运作

4.3.2 CloseECMUSB 函式说明

函数声明	void CloseECMUSB()
传入参数	无
回传参数	无
说明	使用本函式来关闭与 ECM-SK 间的 USB 联机，并释放联机资源

4.3.3 ECMUSBWrite 函式说明

函数声明	bool ECMUSBWrite()
传入参数	无
回传参数	回传一个 bool 值，指出写入资料是否成功
说明	使用本函式透过 USB 将 cmdData 的数据写入至 ECM-SK，若回传值为 False，有可能有下列原因： ● ECM-SK 未开启，请先呼叫 OpenECMUSB 以开启联机 ● 已开启的联机失效，可能为 USB 接触不良或已移除

4.3.4 ECMUSBRead 函式说明

函数声明	bool ECMUSBRead()
传入参数	无
回传参数	回传一个 bool 值，指出读取资料是否成功
说明	使用本函式来透过 USB 从 ECM-SK 读取数据至 respData，若回传

值为 False，有可能有下列原因：

- ECM-SK 未开启，请先呼叫 OpenECMUSB 以开启联机
- 已开启的联机失效，可能为 USB 接触不良或已移除
- 读回数据的 CRC8 值验证错误

4.3.5 ClearCmdData 函式说明

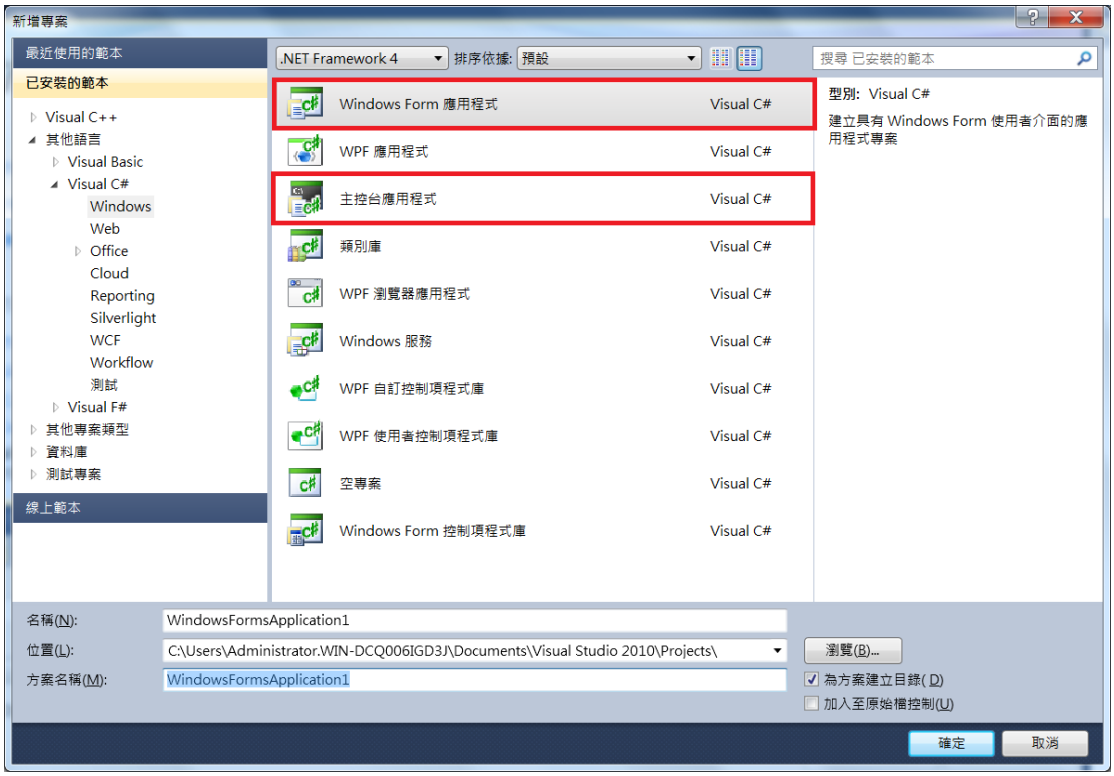
函数声明	void ClearCmdData()
传入参数	无
回传参数	无
说明	使用本函式来清除 cmdData 的内容，使得内容值均为 0

4.4 Visual Studio 项目环境设定

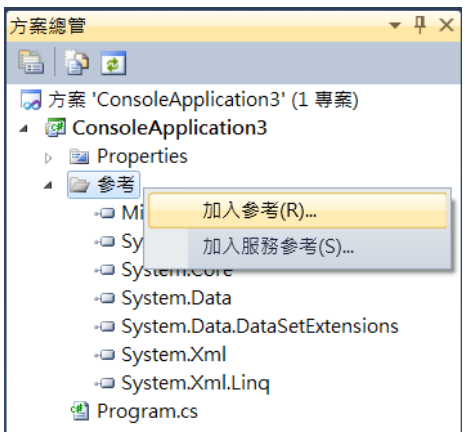
本节以微软公司的 Visual Studio 2010 的环境为例，逐步说明项目环境设定的步骤。

4.4.1 C# .NET 项目环境设定

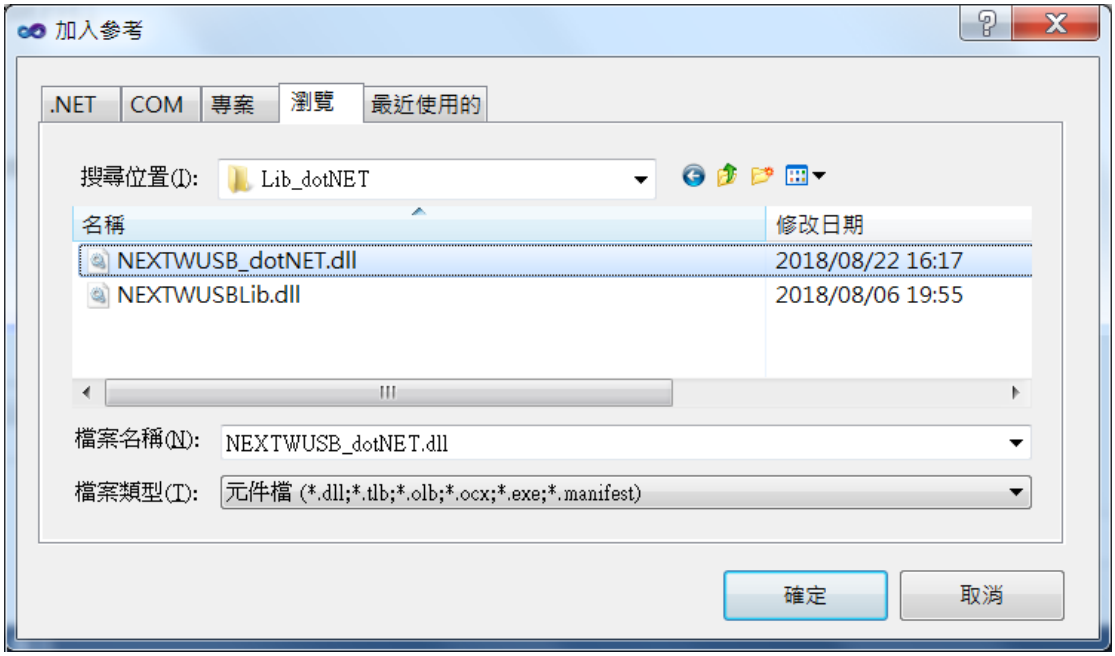
Step 1：新增项目 – 依需求选择「Windows Form 应用程序」或「控制面板应用程序」



Step 2 – 于「方案总管」中的「参考」，右键单击选择「加入参考」



Step 3 – 于「浏览」页签，选择「NEXTWUSB_dotNET_XXB.dll」后按确定



Step 4 – 于程序顶端加入命名空间「NEXTWUSB_dotNET_XXB」

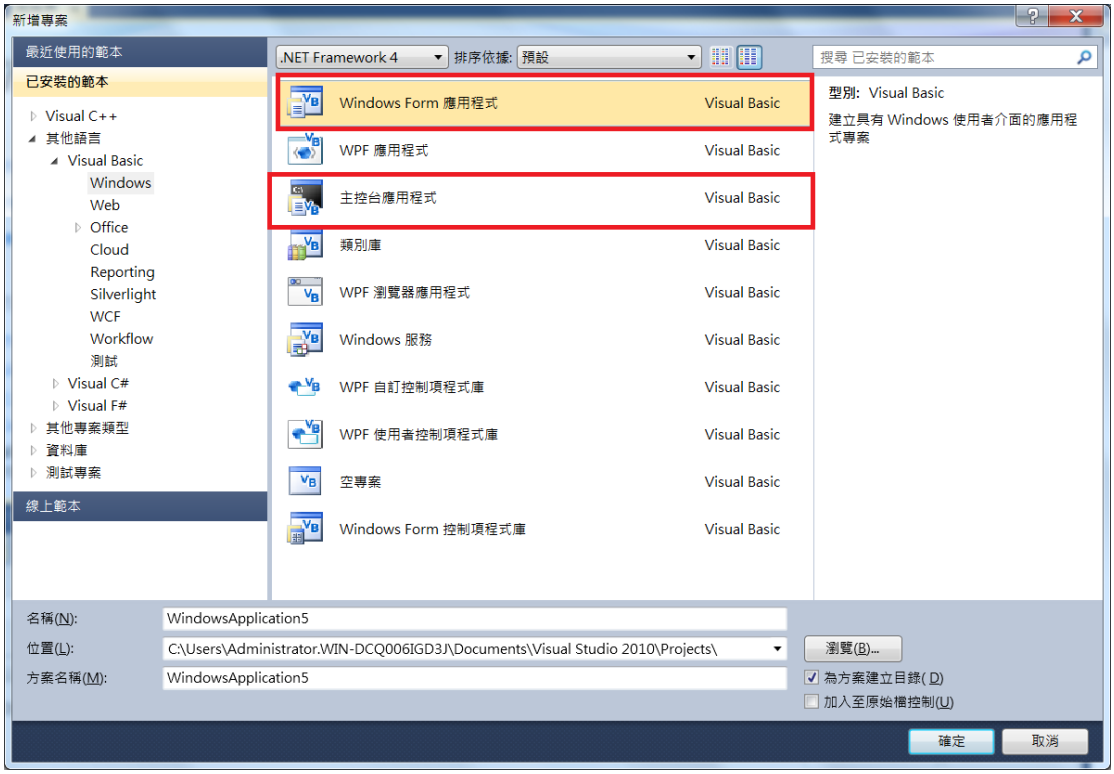
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using NEXTWUSB_dotNET;
```

Step 5 – 即可于主程序使用「NEXTWUSB_dotNET.dll」动态函式库提供功能

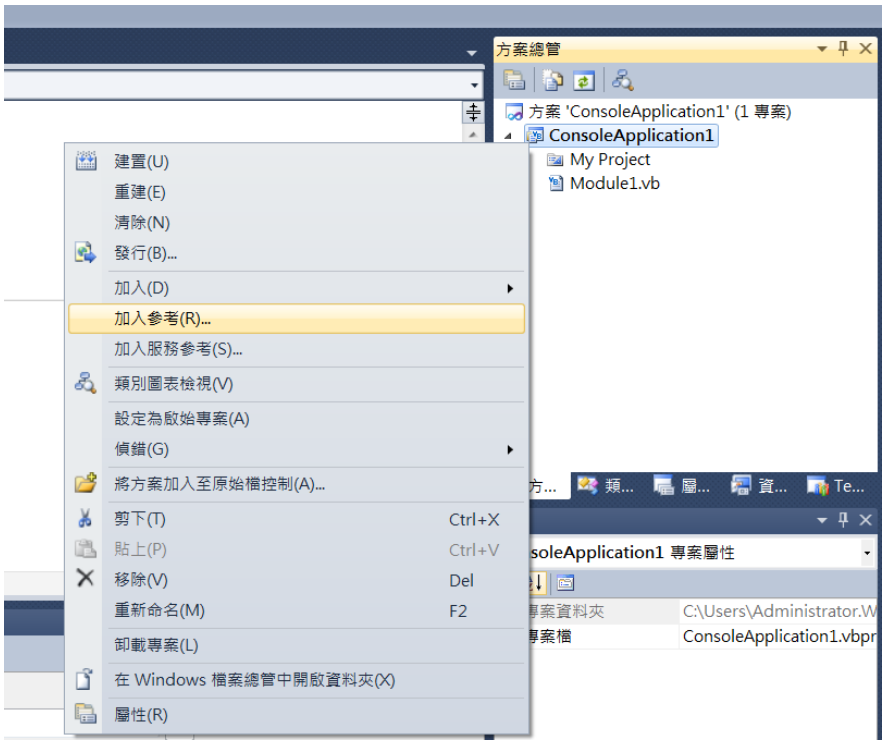
```
class Program
{
    static void Main(string[] args)
    {
        NEXTWUSB.OpenECMUSB();
        NEXTWUSB.ClearCmdData();
        NEXTWUSB.ECMUSBWrite();
        NEXTWUSB.ECMUSBRead();
        NEXTWUSB.CloseECMUSB();
    }
}
```

4.4.2 Visual Basic .NET 项目环境设定

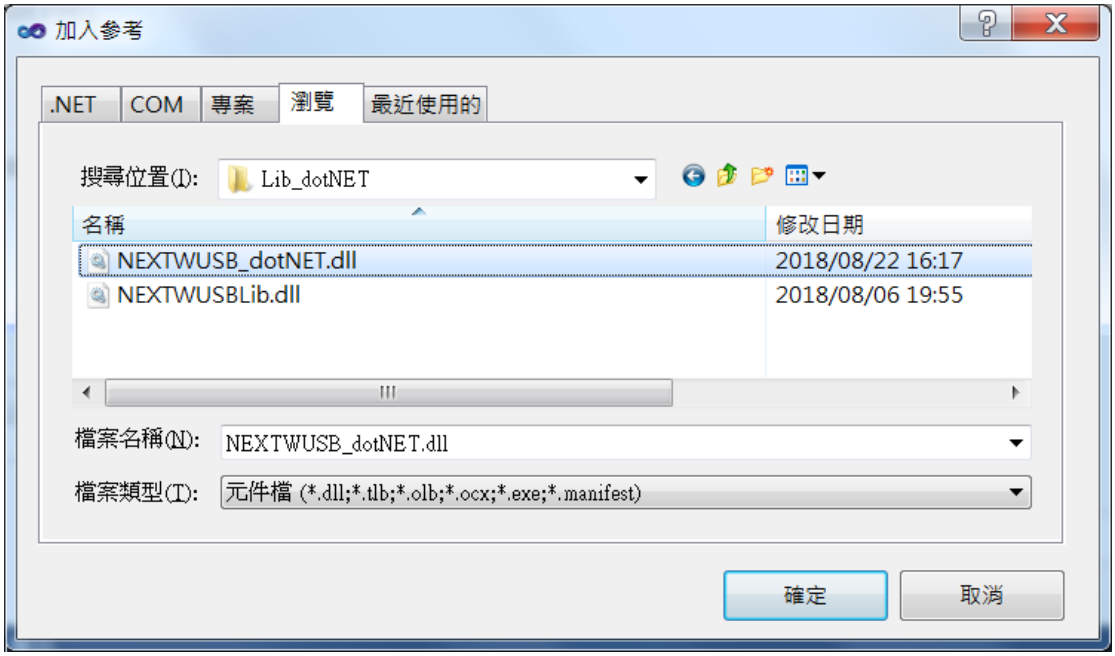
Step 1：新增项目 – 依需求选择「Windows Form 应用程序」或「控制面板应用程序」



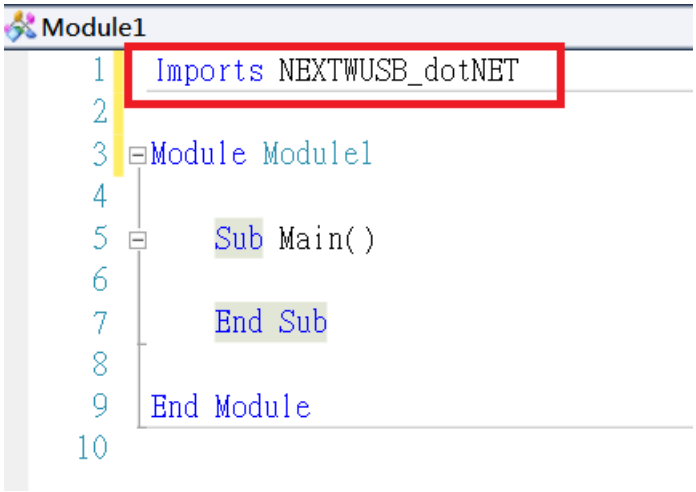
Step 2 – 于「方案总管」中的项目名称，右键单击选择「加入参考」



Step 3 – 于「浏览」页签，选择「NEXTWUSB_dotNET_XXB.dll」后按确定



Step 4 – 于程序顶端加入命名空间「NEXTWUSB_dotNET_XXB」

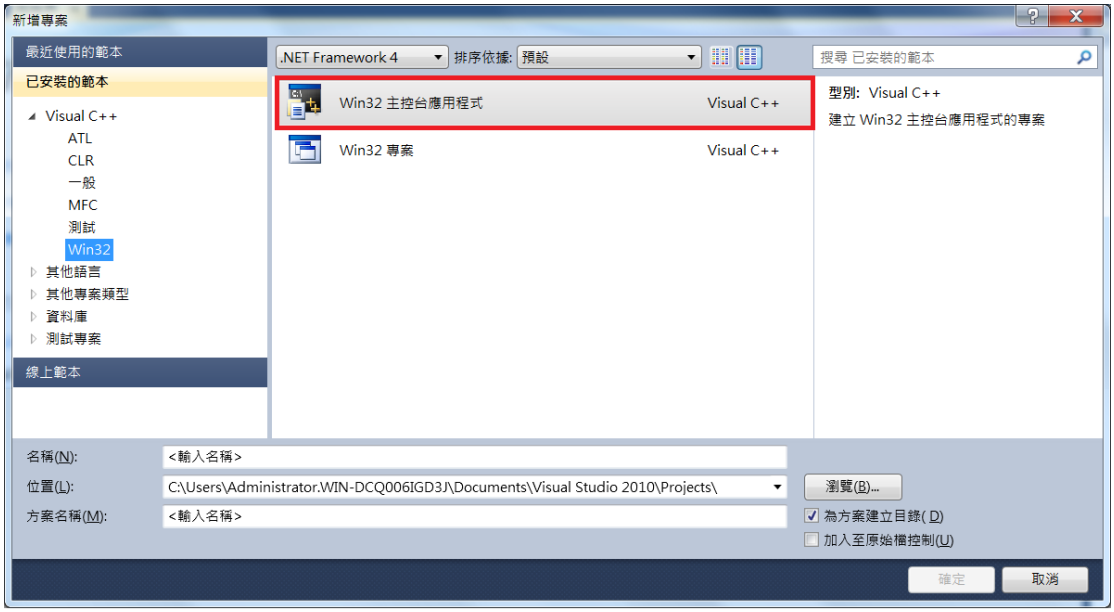


Step 5 – 即可于主程序使用动态函式库提供功能

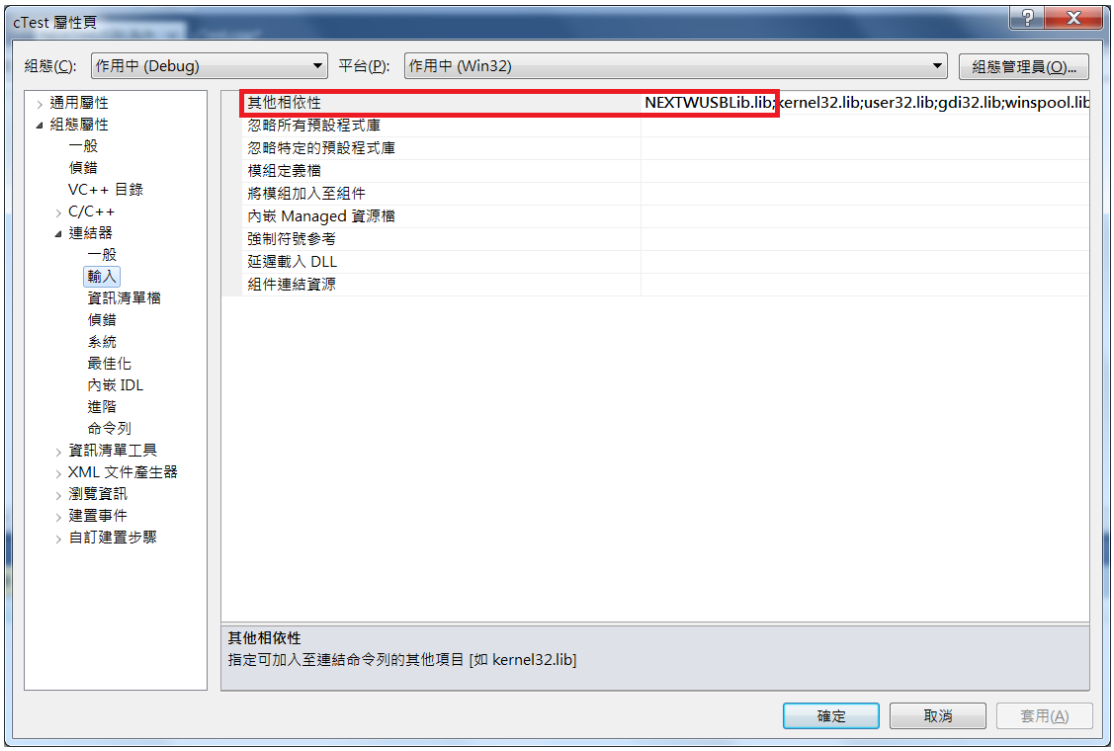
```
Sub Main()
    NEXTWUSB.OpenECMUSB()
    NEXTWUSB.ClearCmdData()
    NEXTWUSB.ECMUSBWrite()
    NEXTWUSB.ECMUSBRead()
    NEXTWUSB.CloseECMUSB()
End Sub
```

4.4.3 C++项目环境设定

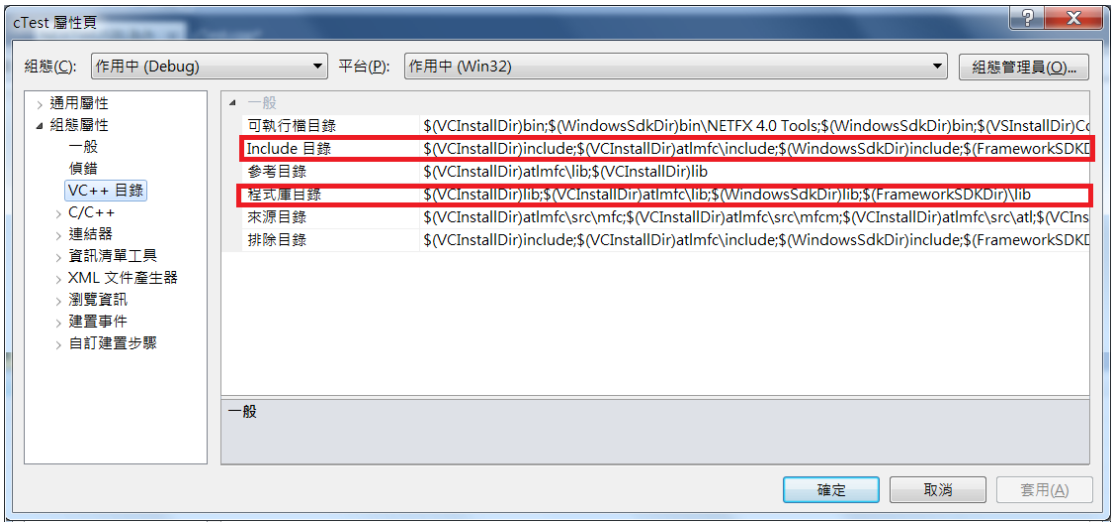
Step 1：新增项目 – 依需求选择项目类型，此处以「Win32 控制面板应用程序」为例



Step 2 – 于项目组态属性页中的「链接器」下的「输入」，右侧「其他相依性」字段，输入「NEXTWUSBLib.lib」



Step 3 – 于项目组态属性页中的「VC++目录」指定「Include 目录」及「链接库目录」，此处目录仅供参考，请依实际状况输入「NEXTWUSBLib.h」及「NEXTWUSBLib.lib」实际目录



Step 4 – 于程序顶端加入 Include 定义档「NEXTWUSBLib.h」

```
#include "stdafx.h"
#include "NEXTWUSBLib.h"
```

Step 5 – 即可于主程序使用「NEXTWUSBLib.dll」动态函式库提供功能

```
int _tmain(int argc, _TCHAR* argv[])
{
    OpenECMUSB();
    printf("Hello ECM USB");
    CloseECMUSB();
    return 0;
}
```