**Modbus通訊說明**

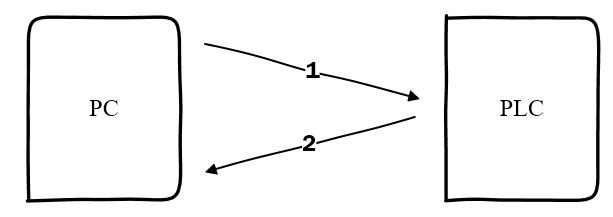
Modbus是工業領域通訊協定的[業界標準](https://zh.wikipedia.org/wiki/De_facto" \o "De facto)，目前分為3種格式，依照各家廠商格式會有些許不同，本次以台達DVP 12SE(網路型PCL)及DVP 32ES(PLC)做介紹

* Modbus Ascii
* Modbus RTU
* Modbus TCP



**通訊概念**

1. **Modbus通訊的流程**



1. 不管是要對PLC做讀取狀態或是控制，都是要PC先發出命令訊號
2. 當PLC收到訊號會回傳值給PC，藉由回傳值可得知PLC狀態或是命令是否有成功。
3. **PLC內部的進制格式與命令關係**

PLC內部有X、Y、M、T、C、D，每一個的進制都有些許不同，在下方表格有做些微的整理。

|  |  |  |  |
| --- | --- | --- | --- |
| 類別 | 類別編號的進制 | 類別儲存範圍 | 類別大小 |
| X | 8 | 不知道大小及範圍  只可以使用FF00、0000 | |
| Y |
| M | 10 |
| T | 0~32767 | 65535  超過32768則無作用，無溢位 |
| C |
| D | -32768 ~ 32767 | 65535  超過65535則無作用，無溢位 |

**在輸入命令的時候需要注意進制之間的關係**

如:在Modbus Ascii來說，需要將命令轉換成Ascii char

Modbus RTU則是轉換成Ascii Hex。

進制之間的轉換和Ascii碼網路都有，請上網爬文~~

**Modbus Ascii**

**通訊:** 9600(串列傳輸速率)、7 (資料位元)、 Even (奇偶校驗位元) 、1 (起始位元)、 1 (停止位元)

**Modbus的組成格式**

隨著功能碼不同，所填入的資料形式也會有些微不同，但格式一定依照下表

| 欄位名 | 組成 | 解釋說明 |
| --- | --- | --- |
| 起始字元 | STX | 起始字元為〝：〞 |
| 通訊站號 | ADR 1 | 通訊站號由兩個ASCII碼組成 |
| ADR 0 |
| 功能碼 | CMD 1 | 功能碼由兩個ASCII碼組成，需查表應用 |
| CMD 0 |
| 資料 | DATA （0） | 資料內容由2n個ASCII碼組成，n≤205，  需查表應用 |
| DATA （1） |
| ………. |
| DATA （n-1） |
| LRC 偵錯碼 | LRC CHK 1 | LRC偵錯碼由2個ASCII碼組成，下方的例題有計算說明。 |
| LRC CHK 0 |

舉例來說(後面會有更詳細的說明)

* SET M0的Modbus Ascii 就是 :01050800FF00F3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **:** (起始字元) | **01**(通訊站號) | **05**  (寫入裝置位元暫存器) | **0800 FF00**  (M0 SET) | **F3**( LRC 偵錯碼) |

* D0=100的Modbus Ascii 就是 :01061000006485

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **:** (起始字元) | **01**(通訊站號) | **06**  (寫入裝置暫存器) | **0100 0064**  (D0 K=100) | **85**( LRC 偵錯碼) |

1. **起始字元 :**

ASCII的起始字元為〝　:　〞(冒號)。

1. **通訊站號 :**

通訊站號為兩個數字組成，意義是如果使用多台PLC，站號則是每台PLC的編號，通常PLC原廠設定站號為01。

1. **功能碼 :**

ASCII的功能碼為兩個數字組成，需查表應用，根據需要的功能和須出做的元件對表應用。

|  |  |  |
| --- | --- | --- |
| 功能碼 | 說明 | 可操作裝置 |
| 0x01 | 讀位元裝置暫存器的值（不可讀輸入節點狀態） | S, Y, M, T, C |
| 0x02 | 讀位元裝置暫存器的值（可讀輸入節點狀態） | S, X, Y, M,T, C |
| 0x03 | 讀單個或多個字符裝置暫存器的值 | T, C, D |
| 0x05 | 寫單個位元裝置位元暫存器的值 | S, Y, M, T, C |
| 0x06 | 寫單個字符裝置暫存器的值 | T, C, D |
| 0x0F | 寫多個位裝置暫存器的值 | S, Y, M, T, C |
| 0x10 | 寫多個字符裝置暫存器的值 | T, C, D |

表格 1

1. **資料:**

依照功能不同，填入的資料形式也會有點變動(在後面會用例題說明)

1. **偵錯碼(校驗碼 LRC):**

字串在傳輸時做基本的偵錯。

LRC運算方式為

假設數值為01 03 10 00 00 02

首先先將值相加 01+03+10+00+00+02=16(16進制)

轉成2進制 0001 0110

做反向1110 1001

再加上1變成1110 1010

再轉成16進制 EA

**以下例題(站號皆為01且命令皆由Ascii char所組成)**

**例題一 SET Y11 命令為 :01050509FF00ED 回應值:01050509FF00ED**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **起始字元** | **通訊站號** | **功能碼** | **資料** | | | **LRC偵錯碼** |
| **資料位址** | | **狀態****或數值** |
| **:** | **01** | **05** | **05** | **09** | **FF00** | **ED** |
|  | **第一站** | **寫入位元狀態** | **Y11** | | **SET** |

1. **起始字元 :**

起始字元為〝　**:**　〞(冒號)。

1. **通訊站號 :**

通訊站號為**01**

1. **功能碼 :**

參照[功能碼表格](#功能碼)，本次使用**05**功能碼

1. **資料位址**

目前想得知Y11的位址是多少，首先，[經查表格](#暫存器位置)得知Y0的位址為0500

Y11的位址為Y0之後的第9位(因Y為8進制，但命令為16進制，故需轉換)

若為Y17的話，位址為Y0後的第15位，但命令為16進制，故轉換為0F

|  |  |  |  |
| --- | --- | --- | --- |
|  | 8進制 | 10進制 | 16進制 |
| Y11 | 11 | 9 | 09 |
| Y17 | 17 | 15 | 0F |

經過轉換後為09h，所以Y11的位址就為**0509**

**狀態或數值**

* SET = FF00
* RST = 0000

1. **LRC偵錯碼 (以下是用SET來算，若RST的話是EC)**

* 01+05+05+09+FF=113
* 只取後面2位並轉換為2進制 13h = 0001 0011
* 做反向 1110 1100
* 再加上1變成1110 1101(取2的補數)
* 再轉成16進制ED

**例題二 SET M20 命令為 :01050814FF00DF 回應值:01050814FF00DF**

**RST M20 命令為 :010508140000DE 回應值:010508140000DE**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **起始字元** | **通訊站號** | **功能碼** | **資料** | | | **LRC偵錯碼** |
| **資料位址** | | **狀態或數值** |
| : | 01 | 05 | 08 | 14 | FF00 | DF |
| : | 01 | 05 | 08 | 14 | 0000 | DE |

1. **起始字元 :**

起始字元為”:”(冒號)。

1. **通訊站號 :**

通訊站號為01

1. **功能碼 :**

參照[功能碼表格](#功能碼)，本次使用05功能碼

1. **資料位址**

目前想得知M20的位址是多少，首先，[經查表格](#暫存器位置)得知M0的位址為0800

M20的位址為M0之後的第20位(因M為10進制，但命令為16進制，故需轉換)

|  |  |
| --- | --- |
| 10進制 | 16進制 |
| 20 | 14 |

經過轉換後為14h，所以M20的位址就為0814

**狀態或數值**

* SET = FF00
* RST = 0000

1. **LRC偵錯碼**

* 01+05+08+14+FF=121
* 只取後面2位並轉換為2進制 13h = 0010 0001
* 做反向 1101 1110
* 再加上1變成1101 1111(取2的補數)
* 再轉成16進制DF

**例題三 讀取Y0~Y17狀態 命令為 :010205000010E8回應值:010202078074**

因組成方式與前面相同，故不再介紹相同的地方(只介紹資料的組成)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **起始字元** | **通訊站號** | **功能碼** | **資料** | | | **LRC偵錯碼** |
| **資料位址** | | **要讀取資料數量** |
| **:** | **01** | **02** | **05** | **00** | **0010** | **E8** |
|  | **第一站** | **讀取位元狀態** | **Y0** | | **16個** |

1. **讀取功能的資料**→由 起始位址+要讀取的資料數量
2. **讀取資料數量的算法:**

* 將尾端數字17(轉成8進制)　-　前端數字0(轉換成8進制) + 1
* F-0+1=10

回應值的組成

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **起始字元** | **通訊站號** | **功能碼** | **資料** | | | **LRC偵錯碼** |
| **資料數量(組)** | **狀態** | |
| **:** | **01** | **02** | **02** | **07** | **80** | **74** |

1. **資料數量**

功能碼後面的兩位數字代表後面會有幾組資料

如上例題，02代表後面會有2組資料

1. **狀態**

先將2組數字轉換成2進制

|  |  |  |
| --- | --- | --- |
|  | 第一組資料 | 第二組資料 |
| 16進制 | 07 | 80 |
| 2進制 | 0000 0111 | 1000 0000 |

二進制的每一個數字分別代表Y0~Y17的每一個位置

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | | | | 7 | | | | 8 | | | | 0 | | | |
| 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Y7 | Y6 | Y5 | Y4 | Y3 | Y2 | Y1 | Y0 | Y17 | Y16 | Y15 | Y14 | Y13 | Y12 | Y11 | Y10 |
|  | | | | | | | |  | | | | | | | |

經由上表可以得知Y0~Y17之中只有Y0、Y1、Y2、Y17是SET的狀態

**例題四 寫入D0=291 命令為 :010610000123C5回應值:010610000123C5**

因組成方式與前面相同，故不再介紹相同的地方(只介紹資料的組成)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **起始字元** | **通訊站號** | **功能碼** | **資料** | | | **LRC偵錯碼** |
| **資料位址** | | **資料內容** |
| **:** | **01** | **06** | **10** | **00** | **0123** | **C5** |
|  | **第一站** | **寫入暫存器值** | **D0** | | **K=291** |

1. **寫入功能**→由 資料位址+資料內容
2. **資料內容的算法:**

* 因命令組成為16進制，但D所存放的數值為10進制，所以需要做轉換
* 首先，將要將291(10進制)轉換成0123(16進制)
* D的值若是負的，需要用另一種算法(例題6會說)

**例題五 讀取D0~D1 :010310000002EA 回應值:01030401230000D4**

因組成方式與前面相同，故不再介紹相同的地方(只介紹資料的組成)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **起始字元** | **通訊站號** | **功能碼** | **資料** | | | **LRC偵錯碼** |
| **資料位址** | | **資料數量** |
| **:** | **01** | **03** | **10** | **00** | **0002** | **EA** |
|  | **第一站** | **讀取位元狀態** | **D0** | | **2個** |

1. **資料位址**

此處填入要查詢的起始資料位址。

1. **資料數量**

填入要讀取幾個資料，如例題所示(要記得將讀取數量(10進制)轉換成命令格式(16進制) )

回應值的組成

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **起始字元** | **通訊站號** | **功能碼** | **資料** | | | **LRC偵錯碼** |
| **資料數量(組)** | **資料內容** | |
| **:** | **01** | **03** | **04** | **0123** | **0000** | **D4** |

1. **資料數量**

**後面會有幾組資料。**

1. **資料內容**

D的大小為2^16=65535 (-32768~32767)，所以資料內容一組為4個16進制組成。

由上表可得知D0=0123、D1=0000，但都是16進制，一般都會再轉成10進制較方便閱讀，轉換後D0=291、D1=0。

**例題六 寫入D0= -11 命令為 :01061000FFF5F5回應值:01061000FFF5F5**

因組成方式與前面相同，故不再介紹相同的地方(只介紹資料的組成)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **起始字元** | **通訊站號** | **功能碼** | **資料** | | | **LRC偵錯碼** |
| **資料位址** | | **資料內容** |
| **:** | **01** | **06** | **10** | **00** | **FFF5** | **F5** |
|  | **第一站** | **寫入暫存器值** | **D0** | | **K=-11** |

1. **寫入功能**→由 資料位址+資料內容
2. **資料內容的算法:**

首先要知道D的大小為2^16(65535)、範圍(-32768~32767)

**資料內容**是由4個16進制組成(65535)，並沒有負號

**那在命令輸入32768、32769(10進制)會如何呢?**

結果為32768 → -32768 、32769 → 32767

所以得到一個算法

|-32768-(要輸入的負值)|+1+32767

|-32768-(-11)|+1+32767 = 65525

再將此值65525(10進制)轉換成FFF5(16進制)

**C、T的大小為2^16(65535)，但範圍(0~32767)，所以命令輸入32768會沒有反應，其他部分就跟D值的組成方式一樣，在此就不再多做說明。**

**Modbus TCP**

通訊:IP(需進入PLC設定。預設:192.168.1.5)、Port(預設502)

基本上Modbus TCP的命令組成方式跟Modbus Ascii差一點

**Modbus TCP比Modbus Ascii**

* **前面多6組(12碼)**
* **不用起始碼**
* **最後不用校驗碼**

**下面是簡單範例，其餘部分請參考上面範例**

Write SET Y0

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Transaction Identifier | Protocol Identifier | Length Field | Unit ID | Function Code | Data | | | |
| 資料位址 | | 高低位元 | |
| 0000 | 0000 | 0006 | 01 | 05 | 05 | 00 | FF | 00 |
| 不知道作用 用NI標準Modbus模組輸出都是也都是0 | | 表示後面6Byte | 站號 | 寫入 | Y0 = 0500  Y1 = 0501 | | SET = FF00  RESET = 0000 | |

Read X0~X7a

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Transaction Identifier | Protocol Identifier | Length Field | Unit ID | Function Code | Data | | | |
| 資料位址 | | 數量 | |
| 0000 | 0000 | 0006 | 01 | 02 | 04 | 00 | 00 | 08 |
| 不知道作用 用NI標準Modbus模組輸出都是也都是0 | | 表示後面6Byte | 站號 | 寫入 | 初始位置 | | 往後讀取幾個 | |

**Modbus RTU**

**沒用過有空再測**

| 裝置名稱 | 裝置編號 | 裝置類別 | 裝置位址 |
| --- | --- | --- | --- |
| S | 000~255 | bit | 0000~00FF |
| S | 256~511 | bit | 0100~01FF |
| S | 512~767 | bit | 0200~02FF |
| S | 768~1023 | bit | 0300~03FF |
| X | 000~377 （Octal） | bit | 0400~04FF |
| Y | 000~377 （Octal） | bit | 0500~05FF |
| T | 000~255 | bit | 0600~06FF |
| C | 000~199 | bit | 0E00~0EC7 |
| C | 200~255 | bit | 0EC8~0EFF |
| M | 000~255 | bit | 0800~08FF |
| M | 256~511 | bit | 0900~09FF |
| M | 512~767 | bit | 0A00~0AFF |
| M | 768~1023 | bit | 0B00~0BFF |
| M | 1024~1279 | bit | 0C00~0CFF |
| M | 1280~1535 | bit | 0D00~0DFF |
| M | 1536~1791 | bit | B000~B0FF |
| M | 1792~2047 | bit | B100~B1FF |
| M | 2048~2303 | bit | B200~B2FF |
| M | 2304~2559 | bit | B300~B3FF |
| M | 2560~2815 | bit | B400~B4FF |
| M | 2816~3071 | bit | B500~B5FF |
| M | 3072~3327 | bit | B600~B6FF |
| M | 3328~3583 | bit | B700~B7FF |
| M | 3584~3839 | bit | B800~B8FF |
| M | 3840~4095 | bit | B900~B9FF |
| T | 000~255 | Word | 0600~06FF |
| C | 000~199 | Word | 0E00~0EC7 |
| C | 200~255 | double Word | 0700~076F |
| D | 000~255 | Word | 1000~10FF |
| D | 256~511 | Word | 1100~11FF |
| D | 512~767 | Word | 1200~12FF |
| D | 768~1023 | Word | 1300~13FF |
| D | 1024~1279 | Word | 1400~14FF |
| D | 1280~1535 | Word | 1500~15FF |
| D | 1536~1791 | Word | 1600~16FF |
| D | 1792~2047 | Word | 1700~17FF |
| D | 2048~2303 | Word | 1800~18FF |
| D | 2304~2559 | Word | 1900~19FF |
| D | 2560~2815 | Word | 1A00~1AFF |
| D | 2816~3071 | Word | 1B00~1BFF |
| D | 3072~3327 | Word | 1C00~1CFF |
| D | 3328~3583 | Word | 1D00~1DFF |
| D | 3584~3839 | Word | 1E00~1EFF |
| D | 3840~4095 | Word | 1F00~1FFF |
| D | 4096~4351 | Word | 9000~90FF |
| D | 4352~4607 | Word | 9100~91FF |
| D | 4608~4863 | Word | 9200~92FF |
| D | 4864~5119 | Word | 9300~93FF |
| D | 5120~5375 | Word | 9400~94FF |
| D | 5376~5631 | Word | 9500~95FF |
| D | 5632~5887 | Word | 9600~96FF |
| D | 5888~6143 | Word | 9700~97FF |
| D | 6144~6399 | Word | 9800~98FF |
| D | 6400~6655 | Word | 9900~99FF |
| D | 6656~6911 | Word | 9A00~9AFF |
| D | 6912~7167 | Word | 9B00~9BFF |
| D | 7168~7423 | Word | 9C00~9CFF |
| D | 7424~7679 | Word | 9D00~9DFF |
| D | 7680~7935 | Word | 9E00~9EFF |
| D | 7936~8191 | Word | 9F00~9FFF |
| D | 8192~8447 | Word | A000~A0FF |
| D | 8448~8703 | Word | A100~A1FF |
| D | 8704~8959 | Word | A200~A2FF |
| D | 8960~9215 | Word | A300~A3FF |
| D | 9216~9471 | Word | A400~A4FF |
| D | 9472~9727 | Word | A500~A5FF |