

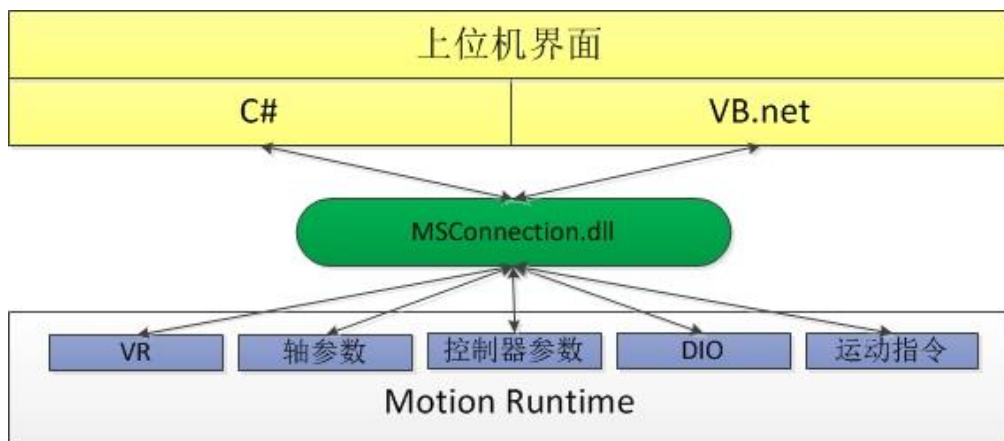
研華**MAS**控制器

MSConnection 函式程式庫使用說明

第 1 章 研華 MSConnection 介紹

1.1 MSConnection 概述

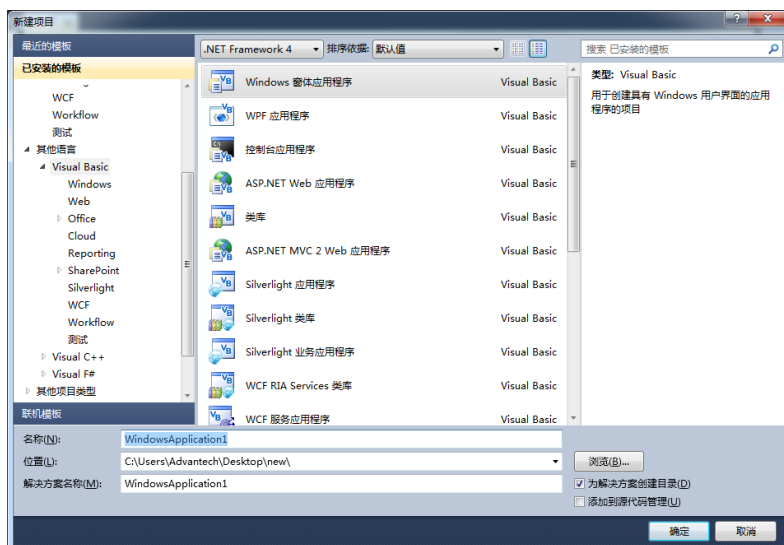
MSConnection.dll 是一個動態連結程式庫，主要是為 C# 或者 VB.net 程式與研華 Motion Runtime 之間的資料交互提供介面。通過提供相應的 API，上位機程式可以直接對 Motion Runtime 中的 VR、軸參數、控制器參數和 DIO 進行讀/寫操作，如下圖所示。



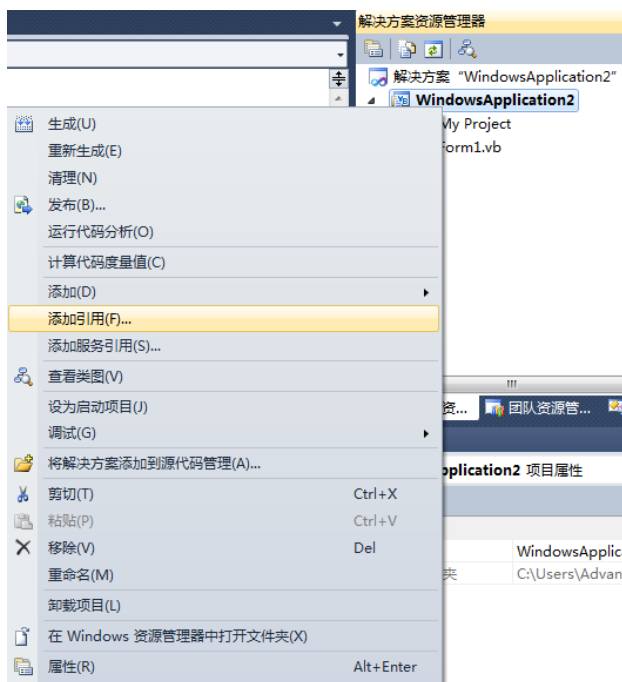
1.2 MSConnection 函式程式庫的使用

1.2.1 Visual Basic 2010 中的使用

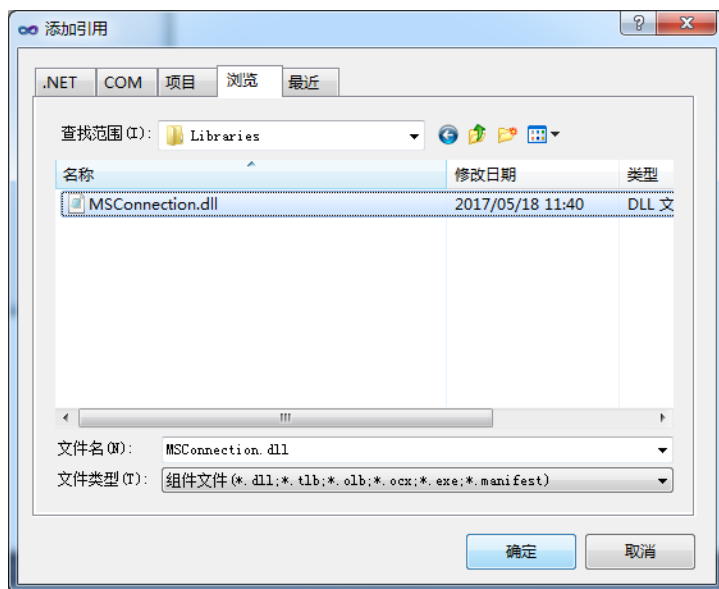
(1) 啟動 Visual Studio 2010，按照“檔”->“新建”，選擇建立 VB 工程；



(2) 按一下開發環境右上角處的工程名，右鍵選擇[添加引用]，如下圖所示；



(3) 按一下[添加引用] 對話方塊的[流覽]，從搜索路徑選擇 “C:\Advantech\Libraries” 文件夾中的 “MSConnection.dll” ，然後按一下[OK]，如下圖所示：



(4) 在編輯介面上右擊，選擇[查看代碼]進入程式原始程式碼編輯介面，在原始參考命名空間下添加 “Imports MSConnection” ，如下圖所示：

```
Imports MSConnection

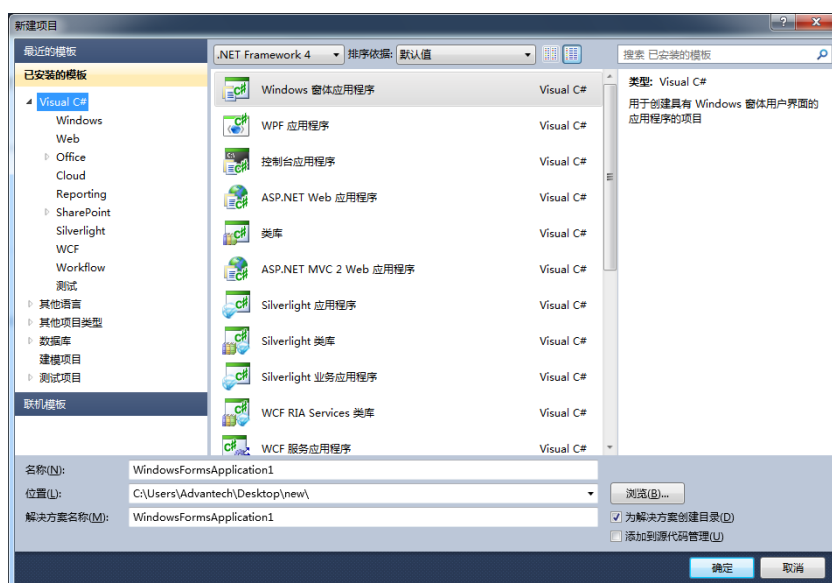
Public Class Form1

End Class
```

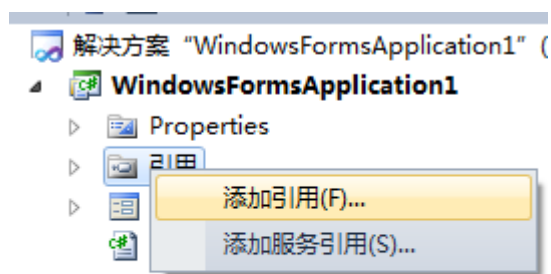
至此，用戶就可以在 Visual Studio 2010 中使用 VB2010 模組調用函式程式庫中的任何函數，開始編寫應用程式。

1.2.2 Visual C# 2010 中的使用

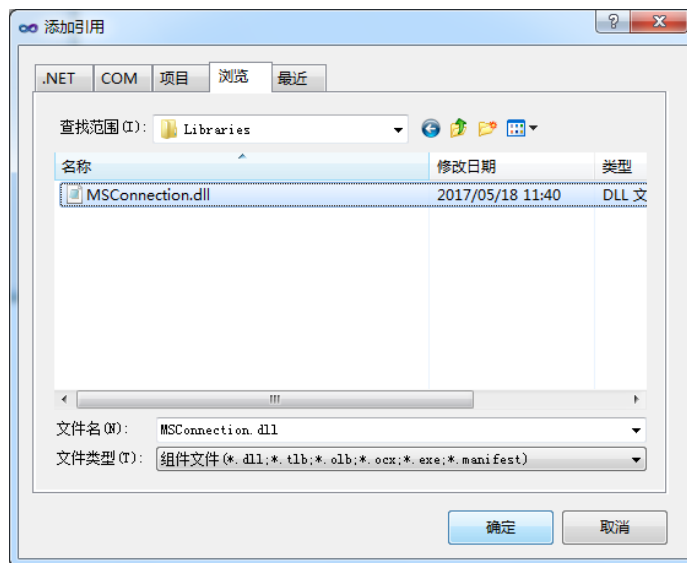
(1) 啟動 Visual Studio 2010，按照“檔”->“新建”，選擇建立 C# 工程；



(2) 按一下開發環境右上角處的[引用]，右鍵選擇[添加引用]，如下圖所示；



(3) 按一下[添加引用] 對話方塊的[瀏覽]，從搜索路徑選擇“C:\Advantech\Libraries” 文件夾中的“MSConnection.dll”，然後按一下[OK]，如下圖所示；



(4) 在編輯介面上右擊，選擇[查看代碼]進入程式原始程式碼編輯介面，在原始參考命名空間下添加“using MSConnection”，如下圖所示：

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using MSConnection;

namespace WindowsFormsApplication1
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            ..
        }
    }
}

```

至此，用戶就可以在 Visual Studio 2010 中使用 C# 模組調用函式程式庫中的任何函數，開始編寫應用程式。

第 2 章 MSConnection 指令詳解

MSConnection 相關注意事項

在引入 MSConnection.dll 檔後，為使用相關 API，需要首先在新建的表單程式內創建 MSClient 用戶端物件，其構造方法為：

```
public MSClient (string IP){...}，
```

參數(IP)為 Motion Runtime 所在電腦的 IP 地址，如果是本機，地址則為“127.0.0.1”，如下所示：

```
public MSClient Wrapper = new MSClient ("127.0.0.1");
```

如果 Motion Runtime 和表單程式不在同一台電腦上，須確保兩邊的 IP 位址在同一網段。比如，Motion Runtime 端的 IP 地址為“192.168.0.1”，則表單程式端的電腦 IP 地址應在“192.168.0.x”(x 為正整數)。創建 MSClient 物件如下所示：

```
public MSClient Wrapper = new MSClient ("192.168.0.1");
```

如果上位機程式有多個表單程式需要和 Motion Runtime 進行資料交互，建議在主表單程式中創建一個靜態的用戶端物件，然後其他副程式來調用這個靜態物件，如下所示：

```
public static MSClient Wrapper = new MSClient ("127.0.0.1");
```

注意：以下 API 的使用預設程式中已經創建了 MSClient 用戶端對象 Wrapper。

2.1 系統

本節指令概覽

章節	API	說明
2.1.1	CheckStatus	獲取與 Runtime 的連接狀態
2.1.2	Dispose	釋放資源，一般在退出程式時調用

2.1.1 CheckStatus

格 式：public bool CheckStatus()

目 的：獲取與 Runtime 的連接狀態

參 數：無

返回值：true，連接正常；false，連接斷開

2.1.2 Dispose

格 式： public void Dispose()

目 的： 釋放資源，一般在退出程式時調用

參 數： 無

返回值： 無

2.2 VR/參數讀取

本節指令概覽

章節	API	說明
2.2.1	ReadList_I16	獲取 short 類型的 VR 或參數陣列
2.2.2	ReadList_U16	獲取 ushort 類型的 VR 或參數陣列
2.2.3	ReadList_I32	獲取 int 類型的 VR 或參數陣列
2.2.4	ReadList_U32	獲取 uint 類型的 VR 或參數陣列
2.2.5	ReadList_F32	獲取 float 類型的 VR 或參數陣列

2.2.1 ReadList_I16

格式 1： public short[] ReadList_I16(int length, int startAddress)

目 的： 獲取 short 類型的 VR 或參數陣列

參 數：

名稱	類型	說明
length	Int	VR 或參數陣列的個數
startAddress	Int	VR 或參數陣列的 ModBUS 首地址

返回值： 讀取成功，返回 short 類型的陣列，長度等於參數 1；

讀取失敗，返回空陣列

例 程： 請參照 2.2.5 的常式

格式 2： `public bool ReadList_I16(ref short[] arr,int length, int startAddress)`

目的： 獲取 short 類型的 VR 或參數陣列

參數：

名稱	類型	說明
arr	Short[]	輸出參數，short 類型陣列
length	Int	VR 或參數陣列的個數
startAddress	Int	VR 或參數陣列的 ModBUS 首地址

返回值： 讀取成功，返回 true；讀取失敗，返回 false

例程： 請參照 2.2.5 的常式

2.2.2 ReadList_U16

格式 1： `public ushort[] ReadList_U16(int length, int startAddress)`

目的： 獲取 ushort 類型的 VR 或參數陣列

參數：

名稱	類型	說明
length	Int	VR 或參數陣列的個數
startAddress	Int	VR 或參數陣列的 ModBUS 首地址

返回值： 讀取成功，返回 ushort 類型的陣列，長度等於參數 1；

讀取失敗，返回空陣列

例程： 請參照 2.2.5 的常式

格式 2： `public bool ReadList_U16(ref ushort[] arr,int length, int startAddress)`

目的： 獲取 ushort 類型的 VR 或參數陣列

參數：

名稱	類型	說明
arr	Ushort[]	輸出參數，ushort 類型陣列
length	Int	VR 或參數陣列的個數
startAddress	Int	VR 或參數陣列的 ModBUS 首地址

返回值：讀取成功，返回 **true**；讀取失敗，返回 **false**

例 程：請參照 2.2.5 的常式

2.2.3 ReadList_I32

格 式： `public int[] ReadList_I32(int length, int startAddress)`

目 的：獲取 **int** 類型的 VR 或參數陣列

參 數：

名稱	類型	說明
length	Int	VR 或參數陣列的個數
startAddress	Int	VR 或參數陣列的 ModBUS 首地址

返回值：讀取成功，返回 **int** 類型的陣列，長度等於參數 1；
讀取失敗，返回空陣列

例 程：請參照 2.2.5 的常式

格式 2： `public bool ReadList_I32(ref int[] arr,int length, int startAddress)`

目 的：獲取 **int** 類型的 VR 或參數陣列

參 數：

名稱	類型	說明
arr	Int[]	輸出參數， int 類型陣列
length	Int	VR 或參數陣列的個數
startAddress	Int	VR 或參數陣列的 ModBUS 首地址

返回值：讀取成功，返回 **true**；讀取失敗，返回 **false**

例 程：請參照 2.2.5 的常式

2.2.4 ReadList_U32

格式 1： `public uint[] ReadList_U32(int length, int startAddress)`

目 的：獲取 **uint** 類型的 VR 或參數陣列

參 數：

名稱	類型	說明
length	Int	VR 或參數陣列的個數
startAddress	Int	VR 或參數陣列的 ModBUS 首地址

返回值：讀取成功，返回 uint 類型的陣列，長度等於參數 1；

讀取失敗，返回空陣列

例 程：請參照 2.2.5 的常式

格式 2：public bool ReadList_U32(ref uint[] arr,int length, int startAddress)

目 的：獲取 uint 類型的 VR 或參數陣列

參 數：

名稱	類型	說明
arr	UInt[]	輸出參數，uint 類型陣列
length	Int	VR 或參數陣列的個數
startAddress	Int	VR 或參數陣列的 ModBUS 首地址

返回值：讀取成功，返回 true；讀取失敗，返回 false

例 程：請參照 2.2.5 的常式

2.2.5 ReadList_F32

格式 1：public float[] ReadList_F32(int length, int startAddress)

目 的：獲取 float 類型的 VR 或參數陣列

參 數：

名稱	類型	說明
length	Int	VR 或參數陣列的個數
startAddress	Int	VR 或參數陣列的 ModBUS 首地址

返回值：讀取成功，返回 float 類型的陣列，長度等於參數 1；

讀取失敗，返回空陣列

格式 2：public bool ReadList_F32(ref float[] arr,int length, int startAddress)

目 的：獲取 float 類型的 VR 或參數陣列

參 數：

名稱	類型	說明
arr	Float[]	輸出參數，float 類型陣列
length	Int	VR 或參數陣列的個數
startAddress	Int	VR 或參數陣列的 ModBUS 首地址

返回值： 讀取成功，返回 true；讀取失敗，返回 false

例 程：

1. 獲取從 VR(2)到 VR(4)的資料，如下圖所示，VR(2)、VR(3)和 VR(4)的數值分別為-100、1 和 200，VR 陣列長度即為 3，開始位址為 40005：



名称	当前值	描述	初始值	Modbus	数据类型
范围[0-1...					
VR(0)	0		0	40001	BIT_32_F...
VR(1)	0		0	40003	BIT_32_F...
VR(2)	-100		0	40005	BIT_32_F...
VR(3)	1		0	40007	BIT_32_F...
VR(4)	200		0	40009	BIT_32_F...
VR(5)	0		0	40011	BIT_32_F...

float [] a = null; //聲明一個 F32 類型的陣列，並賦值 null

格式 1 的用法：

```
a = Wrapper.ReadList_F32(3, 40005); //將 API 返回結果賦值給 a
```

格式 2 的用法：

```
If ( Wrapper.ReadList_F32(ref a, 3, 40005) ) {  
    //讀取正確後對應的操作...  
}; //將 a 作為輸出參數，返回值為 true 或 false
```

則 a[0]、a[1]和 a[2]的值分別等於-100、1 和 200。

2. 獲取軸 0 的初速度、運行速度、加速度和減速度(參照“3.2.4 軸參數、配置對應的 ModBUS 地址”，這四個參數的 ModBUS 位址是連續的，從 45001 開始，數值為 float 類型)，如下圖所示：

轴参数		
参数	轴(0)	
UNIT_NUM	1	
UNIT_DENO...	1	
Speed		
MAXVEL	5,000,000	
MAXACC	50,000,000	
MAXDEC	50,000,000	
VH	8,000	
VL	2,000	
ACC	10,000	
DEC	10,000	
JK	0	

```
float [] a = null; //聲明一個 F32 類型的陣列，並賦值 null
```

格式 1 的用法：

```
a = Wrapper.ReadList_F32(4, 45001); //將 API 返回結果賦值給 a
```

格式 2 的用法：

```
If ( Wrapper.ReadList_F32(ref a, 4, 45001) ) {  
    //讀取正確後對應的操作...  
}; //將 a 作為輸出參數，返回值為 true 或 false
```

則 a[0]、a[1]、a[2]和 a[3]的值分別等於 2000、8000、10000 和 10000。

2.3 VR/參數設置

本節指令概覽

章節	API	說明
2.3.1	WriteList_I16	設置 short 類型的 VR 或參數陣列
2.3.2	WriteList_U16	設置 ushort 類型的 VR 或參數陣列
2.3.3	WriteList_I32	設置 int 類型的 VR 或參數陣列
2.3.4	WriteList_U32	設置 uint 類型的 VR 或參數陣列
2.3.5	WriteList_F32	設置 float 類型的 VR 或參數陣列

2.3.1 WriteList_I16

格 式： public bool WriteList_I16(short[] List, int index, int startAddress, int length)

目 的： 設置 short 類型的 VR 或參數陣列

參 數：

名稱	類型	說明
List	Short[]	輸入 short 類型陣列
index	Int	陣列的起始索引
startAddress	Int	VR 或參數陣列的 ModBUS 首地址
length	Int	VR 或參數陣列的個數

返回值： 設置成功，返回 true；設置失敗，返回 false

例 程： 請參照 2.3.5 的常式

2.3.2 WriteList_U16

格 式： public bool WriteList_U16(ushort[] List, int index, int startAddress, int length)

目 的： 設置 ushort 類型的 VR 或參數陣列

參 數：

名稱	類型	說明
----	----	----

List	Ushort[]	輸入 ushort 類型陣列
index	Int	陣列的起始索引
startAddress	Int	VR 或參數陣列的 ModBUS 首地址
length	Int	VR 或參數陣列的個數

返回值： 設置成功，返回 true；設置失敗，返回 false

例 程： 請參照 2.3.5 的常式

2.3.3 WriteList_I32

格 式： public bool WriteList_I32(int[] List, int index, int startAddress, int length)

目 的： 設置 int 類型的 VR 或參數陣列

參 數：

名稱	類型	說明
List	Int[]	輸入 int 類型陣列
index	Int	陣列的起始索引
startAddress	Int	VR 或參數陣列的 ModBUS 首地址
length	Int	VR 或參數陣列的個數

返回值： 設置成功，返回 true；設置失敗，返回 false

例 程： 請參照 2.3.5 的常式

2.3.4 WriteList_U32

格 式： public bool WriteList_U32(uint[] List, int index, int startAddress, int length)

目 的： 設置 uint 類型的 VR 或參數陣列

參 數：

名稱	類型	說明
List	UInt[]	輸入 uint 類型陣列
index	Int	陣列的起始索引
startAddress	Int	VR 或參數陣列的 ModBUS 首地址

length Int VR 或參數陣列的個數

返回值： 設置成功，返回 true；設置失敗，返回 false

例 程： 請參照 2.3.5 的常式

2.3.5 WriteList_F32

格 式： public bool WriteList_F32(float[] List, int index, int startAddress, int length)

目 的： 設置 float 類型的 VR 或參數陣列

參 數：

名稱	類型	說明
List	Float[]	輸入 float 類型陣列
index	Int	陣列的起始索引
startAddress	Int	VR 或參數陣列的 ModBUS 首地址
length	Int	VR 或參數陣列的個數

返回值： 設置成功，返回 true；設置失敗，返回 false

例 程：

```
short [] a= new short[5]{1,2,3,4,5}; //聲明一個長度為 5 的 F32 類型陣列
```

1. 將 a 陣列的資料賦值給 VR(1)到 VR(5)：

參數 1 對應 a，參數 2 對應 a 的起始索引 0，參數 3 對應 VR(1)的 ModeBUS 地址 40003，參數 4 對應 VR 的個數 5

```
Wrapper.WriteList_F32( a, 0, 40003, 5);
```

運行結果如下圖所示：

VR表					
名稱	当前值	描述	初始值	Modbus	数据类型
范围[0-1...					
VR(0)	0		0	40001	BIT_32_F...
VR(1)	1		0	40003	BIT_32_F...
VR(2)	2		0	40005	BIT_32_F...
VR(3)	3		0	40007	BIT_32_F...
VR(4)	4		0	40009	BIT_32_F...
VR(5)	5		0	40011	BIT_32_F...
VR(6)	0		0	40013	BIT_32_F...
VR(7)	0		0	40015	BIT_32_F...

2. 將 a[2]的數據賦值給 VR(6)：

參數 1 對應 a,參數 2 對應 a 的起始索引 2,參數 3 對應 VR(6)的 ModBUS 地址 40013, 參數 4 對應 VR 的個數 1

```
Wrapper.WriteList_F32( a, 2, 40013, 1);
```

運行結果如下圖所示：

VR表						
名称	当前值	描述	初始值	Modbu	数据类型	
范围[0-1...						
VR(0)	0		0	40001	BIT_32_F...	▼
VR(1)	1		0	40003	BIT_32_F...	▼
VR(2)	2		0	40005	BIT_32_F...	▼
VR(3)	3		0	40007	BIT_32_F...	▼
VR(4)	4		0	40009	BIT_32_F...	▼
VR(5)	5		0	40011	BIT_32_F...	▼
VR(6)	3		0	40013	BIT_32_F...	▼
VR(7)	0		0	40015	BIT_32_F...	▼

```
short [] b= new short[4]{1999,2999,29999,29999}; //聲明一個長度為 4 的 F32 類型陣列
```

3. 將 b 陣列賦值給軸 0 的初速度、運行速度、加速度和減速度：

參照 “3.2.4 軸參數、配置對應的 ModBUS 地址”，軸 0 的初速度、運行速度、加速度和減速度的 ModBUS 位址是連續的，從 45001 開始，數值為 float 類型，則參數 1 對應 b,參數 2 對應 b 的起始索引 0,參數 3 對應軸 0 初速度的 ModBUS 地址 45001, 參數 4 對應參數的個數 4

```
Wrapper.WriteList_F32( b, 0, 45001, 4);
```

運行結果如下圖所示：

轴参数		
参数	轴(0)	
UNIT_NUM	1	
UNIT_DENO...	1	
Speed		
MAXVEL	5,000,000	
MAXACC	50,000,000	
MAXDEC	50,000,000	
VH	2,999	
VL	1,999	
ACC	29,999	
DEC	29,999	

2.4 DIO 讀取

本節指令概覽

章節	API	說明
2.4.1	ReadList_DIN	獲取 DIN 數據
2.4.2	ReadList_DOUT	獲取 DOUT 數據

2.4.1 ReadList_DIN

格式 1： public byte[] ReadList_DIN(int length)

目 的： 獲取 DIN 數據

參 數：

名稱	類型	說明
length	Int	DIN 陣列的長度，從索引 0 開始

返回值： 讀取成功，返回 byte 類型的陣列，長度等於參數 1；

讀取失敗，返回空陣列

例 程： 請參照 2.4.2 的常式

格式 2： public bool ReadList_DIN(ref byte[] din,int startAddress,int length)

目 的： 獲取 DIN 數據

參 數：

名稱	類型	說明
din	Byte[]	輸出參數，byte 類型陣列
startAddress	Int	DIN 陣列的 ModBUS 首地址
length	Int	DIN 陣列的個數

返回值： 讀取成功，返回 true；讀取失敗，返回 false

例 程： 請參照 2.4.2 的常式

2.4.2 ReadList_DOUT

格 式： public byte[] ReadList_DOUT(int length)

目 的： 獲取 DOUT 數據

參 數：

名稱	類型	說明
length	Int	DOUT 陣列的長度，從索引 0 開始

返回值： 讀取成功，返回 byte 類型的陣列，長度等於參數 1；

讀取失敗，返回空陣列

格式 2： public bool ReadList_DOUT(ref byte[] dout,int startAddress,int length)

目 的： 獲取 DOUT 數據

參 數：

名稱	類型	說明
dout	Byte[]	輸出參數，byte 類型陣列
startAddress	Int	DOUT 陣列的 ModBUS 首地址
length	Int	DOUT 陣列的個數

返回值： 讀取成功，返回 true；讀取失敗，返回 false

例 程：

獲取 DOUT 陣列從索引 0 到 4 的資料，如下圖所示，DOUT(0)到 DOUT(4)的值分別為 0、1、1、0 和 1，DOUT 陣列的長度即為 5，ModBUS 首地址為 1。

I/O表					
名称	当前值	描述	初始值	Modbus	数据类型
Motion_PCI-1245-MA5					
DOUT(0)	0	Axis-0 OUT...	0	1	BOOL
DOUT(1)	1	Axis-0 OUT...	0	2	BOOL
DOUT(2)	1	Axis-0 OUT...	0	3	BOOL
DOUT(3)	0	Axis-0 OUT...	0	4	BOOL
DOUT(4)	1	Axis-1 OUT...	0	5	BOOL
DOUT(5)	0	Axis-1 OUT...	0	6	BOOL

byte [] a = null; //聲明一個 byte 類型陣列，並賦值 null

格式 1 的用法：

a = Wrapper.ReadList_DOUT(5); //將 API 返回值賦值給 a

格式 2 的用法：

If (Wrapper. ReadList_DOUT (ref a, 1, 5)) {

//讀取正確後對應的操作...

}; //將 a 作為輸出參數，返回值為 true 或 false

則 a[0]到 a[4]的值分別等於 0、1、1、0 和 1。

2.5 D0 設置

本節指令概覽

章節	API	說明
2.5.1	Write_DOUT	設置 DOUT 數據

2.5.1 Write_DOUT

格式 1： public bool Write_DOUT(int address, int value)

目 的： 設置單個 DOUT 數據

參 數：

名稱	類型	說明
address	Int	DOUT 的 ModBUS 地址
value	Int	0 對應關閉 DOUT，1 對應打開 DOUT

返回值： 設置成功，返回 true；設置失敗，返回 false

例 程：

1. 設置 DOUT(5)打開，其 ModBUS 地址為 6，則參數 1 為 6，參數 2 為 1，
Wrapper.Write_DOUT(6, 1);

運行結果如下圖所示：

I/O表					
名称	当前值	描述	初始值	Modbus	数据类型
Motion_PCI-1245-MA					
DOUT(0)	0	Axis-0 OUT...	0	1	BOOL
DOUT(1)	0	Axis-0 OUT...	0	2	BOOL
DOUT(2)	0	Axis-0 OUT...	0	3	BOOL
DOUT(3)	0	Axis-0 OUT...	0	4	BOOL
DOUT(4)	0	Axis-1 OUT...	0	5	BOOL
DOUT(5)	1	Axis-1 OUT...	0	6	BOOL

2. 設置 DOUT(5)關閉，其 ModBUS 地址為 6，則參數 1 為 6，參數 2 為 0，
Wrapper.Write_DOUT(6, 0);

運行結果如下圖所示：

I/O表					
名称	当前值	描述	初始值	Modbus	数据类型
Motion_PCI-1245-MA:					
DOUT(0)	0	Axis-0 OUT...	0	1	BOOL
DOUT(1)	0	Axis-0 OUT...	0	2	BOOL
DOUT(2)	0	Axis-0 OUT...	0	3	BOOL
DOUT(3)	0	Axis-0 OUT...	0	4	BOOL
DOUT(4)	0	Axis-1 OUT...	0	5	BOOL
DOUT(5)	0	Axis-1 OUT...	0	6	BOOL

格式 2： public bool Write_DOUT(bool[] List, int index, int startAddress, int length)

目 的： 設置多個 DOUT 數據

參 數：

名稱	類型	說明
List	Bool[]	輸入 bool 類型陣列
index	Int	陣列的起始索引
startAddress	Int	DOUT 陣列的 ModBUS 首地址
length	Int	DOUT 陣列的個數

返回值： 設置成功，返回 true；設置失敗，返回 false

例 程：

設置 DOUT(1)- DOUT(5)狀態分別為打開，關閉，打開，關閉，打開：

1. bool[] a = new bool[] {true , false, true , false, true }; //聲明並賦值一個 bool 類型陣列
 2. 調用 Write_DOUT 函數，參數 1 為 bool 類型陣列 a，參數 2 為 a 的起始索引 0，參數 3 為 DOUT(1)的 ModBUS 地址 2，參數 4 為 DOUT 個數 5
- ```
If (Wrapper. Write_DOUT (a, 0, 2, 5)) {
 //設置正確後對應的操作...
}; //返回值為 true 或 false
```

運行結果如下圖所示：

| I/O表                |     |               |     |        |      |
|---------------------|-----|---------------|-----|--------|------|
| 名称                  | 当前值 | 描述            | 初始值 | Modbus |      |
| Motion_MAS-328X Sin |     |               |     |        |      |
| DOUT(0)             | 0   | Axis-0 OUT... | 0   | 1      | BOOL |
| DOUT(1)             | 1   | Axis-0 OUT... | 0   | 2      | BOOL |
| DOUT(2)             | 0   | Axis-0 OUT... | 0   | 3      | BOOL |
| DOUT(3)             | 1   | Axis-0 OUT... | 0   | 4      | BOOL |
| DOUT(4)             | 0   | Axis-1 OUT... | 0   | 5      | BOOL |
| DOUT(5)             | 1   | Axis-1 OUT... | 0   | 6      | BOOL |
| DOUT(6)             | 0   | Axis-1 OUT... | 0   | 7      | BOOL |

2.6 軸狀態讀取

本節指令概覽

| 章節    | API         | 說明            |
|-------|-------------|---------------|
| 2.6.1 | GetMIOState | 獲取軸 MIO 狀態    |
| 2.6.2 | GetAxDPOS   | 獲取軸理論位置       |
| 2.6.3 | GetAxMPOS   | 獲取軸回饋位置       |
| 2.6.4 | GetAxState  | 獲取軸當前狀態       |
| 2.6.5 | GetAxCmdVel | 獲取軸理論速度       |
| 2.6.6 | GetAxError  | 獲取軸 LastError |

2.6.1 GetMIOState

格 式： public int GetMIOState (int AxisIndex, int Index)

目 的： 獲取軸 MIO 狀態

參 數：





















| 名稱        | 類型  | 說明                                                         |
|-----------|-----|------------------------------------------------------------|
| AxisIndex | Int | 所選軸號                                                       |
| Index     | Int | MIO 相應 bit 的索引 <div> 0 ：RDY 信號 1 ：ALM 信號 2 ：LMTP 信號 </div> |

- 3 : LMTN 信號
- 4 : ORG 信號
- 5 : DIR 信號
- 6 : EMG 信號
- 7 : PCS 信號
- 8 : ERC 信號
- 9 : EZ 信號
- 10 : CLR 信號
- 11 : LTC 信號
- 12 : SD 信號
- 13 : INP 信號
- 14 : SVON 信號
- 15 : ALRM 信號
- 16 : SLMTP 信號
- 17 : SLMTN 信號
- 18 : CMP 信號
- 19 : CAMDO 信號

返回值： 0 為信號 disable；1 為信號 enable

例 程：

獲取軸 1 的 SVON 信號，如下圖所示，則參數 1 為 1，參數 2 為 14，

| 轴状态  |      |       |                                                                                     |                                                                                     |                                                                                     |                                                                                       |                                                                                       |
|------|------|-------|-------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------|
| 轴    | 描述   | STATE | SVON                                                                                | EL+                                                                                 | EL-                                                                                 | ORG                                                                                   | ALM                                                                                   |
| 轴(0) | 轴(0) | READY |  |  |  |  |  |
| 轴(1) | 轴(1) | READY |  |  |  |  |  |
| 轴(2) | 轴(2) | READY |  |  |  |  |  |
| 轴(3) | 轴(3) | READY |  |  |  |  |  |

int a = 0; //聲明 int 類型的變數

a = Wrapper. GetMIOState (1, 14); //將 API 返回值賦值給 a

則 a 的值等於 1。

## 2.6.2 GetAxDPOS

格 式： public float GetAxDPOS(int AxisIndex)

目 的： 獲取軸理論位置

參 數：

| 名稱        | 類型  | 說明   |
|-----------|-----|------|
| AxisIndex | Int | 所選軸號 |

返回值： 返回 float 類型的資料，值等於對應軸的理論位置

例 程：

獲取軸 1 的理論位置，如下圖所示，則參數為 1，

| 轴状态  |      |       |      |     |     |     |     |      |      |     |     |       |
|------|------|-------|------|-----|-----|-----|-----|------|------|-----|-----|-------|
| 轴    | 描述   | STATE | SVON | EL+ | EL- | ORG | ALM | SEL+ | SEL- | INP | EMG | DPOS  |
| 轴(0) | 轴(0) | READY |      |     |     |     |     |      |      |     |     | 0     |
| 轴(1) | 轴(1) | READY |      |     |     |     |     |      |      |     |     | 1,999 |
| 轴(2) | 轴(2) | READY |      |     |     |     |     |      |      |     |     | 0     |
| 轴(3) | 轴(3) | READY |      |     |     |     |     |      |      |     |     | 0     |

float a = Wrapper.GetAxDPOS(1); //將 API 返回值賦值給 float 類型的變數 a  
則 a 的值等於 1999。

### 2.6.3 GetAxMPOS

格 式： public float GetAxMPOS (int AxisIndex)

目 的： 獲取軸實際位置

參 數：

| 名稱        | 類型  | 說明   |
|-----------|-----|------|
| AxisIndex | Int | 所選軸號 |

返回值： 返回 float 類型的資料，值等於對應軸的實際位置

例 程： 請參照 2.6.2 的常式

### 2.6.4 GetAxState

格 式： public ushort GetAxState(int AxisIndex)

目 的： 獲取軸當前狀態

參 數：

| 名稱        | 類型  | 說明   |
|-----------|-----|------|
| AxisIndex | Int | 所選軸號 |

返回值： 返回 ushort 類型的資料，值等於對應軸的當前狀態，如下

- 0：軸不可用狀態
- 1：Ready 狀態
- 2：軸停止狀態，但未 Ready
- 3：軸處於錯誤狀態，軸被停止運動
- 4：軸正在執行回原點運動中
- 5：軸正在執行單軸點位運動中
- 6：軸正在執行單軸連續運動中
- 7：軸正在參與插補運動中或同步運動中
- 8：軸處於外部 JOG 模式中
- 9：軸處於外部 MPG 模式中

例 程： 請參照 2.6.2 的常式

### 2.6.5 GetAxCmdVel

格 式： public float GetAxCmdVel(int AxisIndex)

目 的： 獲取軸理論速度

參 數：

| 名稱        | 類型  | 說明   |
|-----------|-----|------|
| AxisIndex | Int | 所選軸號 |

返回值： 返回 float 類型的資料，值等於對應軸的理論速度

例 程： 請參照 2.6.2 的常式

### 2.6.6 GetAxError

格 式： public uint GetAxError (int AxisIndex)

目 的： 獲取軸 LastError

參 數：



| 名稱        | 類型  | 說明   |
|-----------|-----|------|
| AxisIndex | Int | 所選軸號 |

返回值： 返回 uint 類型的資料，值等於對應軸的錯誤資訊

例 程： 請參照 2.6.2 的常式

## 2.7 控制器狀態讀取

### 本節指令概覽

| 章節    | API        | 說明              |
|-------|------------|-----------------|
| 2.7.1 | GetCtrlErr | 獲取控制器 LastError |
| 2.7.2 | GetAxCnt   | 獲取控制器軸數量        |
| 2.7.3 | GetDOUTCnt | 獲取控制器 DO 數量     |
| 2.7.4 | GetDINCnt  | 獲取控制器 DI 數量     |

### 2.7.1 GetCtrlErr

格 式： public uint GetCtrlErr()

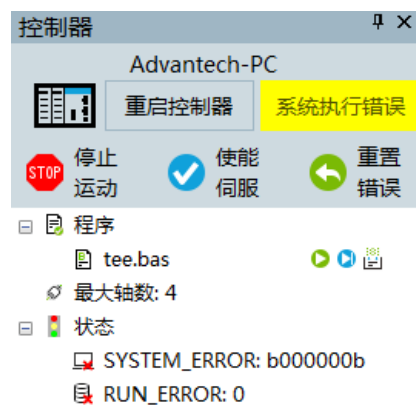
目 的： 獲取控制器 LastError

參 數： 無

返回值： 返回 uint 類型的資料，值等於控制器最近一次錯誤代碼

例 程：

獲取控制器最近一次錯誤代碼，如下圖所示，



uint a = Wrapper.GetCtrlErr(); //將 API 返回值賦值給 unit 類型的 a

則 a 的值等於 2952790027(b000000b 的十進位轉換結果)。

### 2.7.2 GetAxCnt

格 式： public uint GetAxCnt()

目 的： 獲取控制器軸數量

參 數： 無

返回值： 返回 uint 類型的資料，值等於控制器軸的數量

例 程： 請參照 2.7.1 的常式

### 2.7.3 GetDOUTCnt

格 式： public uint GetDOUTCnt()

目 的： 獲取控制器 DOUT 數量

參 數： 無

返回值： 返回 uint 類型的資料，值等於控制器 DOUT 的數量

例 程： 請參照 2.7.1 的常式

### 2.7.4 GetDINCnt

格 式： public uint GetDINCnt()

目 的： 獲取控制器 DIN 數量

參 數： 無

返回值： 返回 uint 類型的資料，值等於控制器 DIN 的數量

例 程： 請參照 2.7.1 的常式

## 2.8 運動指令

### 本節指令概覽

| 章節    | API       | 說明                    |
|-------|-----------|-----------------------|
| 2.8.1 | MOVE 運動指令 | 操作 ModBUS 位址執行單軸相對運動  |
| 2.8.2 | JOG 運動指令  | 操作 ModBUS 位址執行 JOG 運動 |

|       |           |                        |
|-------|-----------|------------------------|
| 2.8.3 | Home 運動指令 | 操作 ModBUS 位址執行 Home 運動 |
| 2.7.4 | 停止運動指令    | 操作 ModBUS 地址停止運動       |

## 2.8.1 MOVE 運動指令

說 明：

當上位機寫 1 到“指令對應的 ModBUS 位址”時，對應的軸會執行單軸相對運動，運動的距離為該軸對應的“運動距離對應的 ModBUS 地址”裡的值；上位機寫 0 到“指令對應的 ModBUS 位址”時，控制器不做任何處理。

| 說明        | 指令對應的 ModBUS 地址<br>(資料類型：Bit) | 運動距離對應的 ModBUS 地址<br>(資料類型：F32) |
|-----------|-------------------------------|---------------------------------|
| 軸 0 相對運動  | 9001                          | 44501                           |
| 軸 1 相對運動  | 9002                          | 44503                           |
| .....     |                               |                                 |
| 軸 15 相對運動 | 9016                          | 44531                           |

例 程：

目的：操作 ModBUS 地址，使軸 1 正向運動 20000 個脈衝當量

步驟：

```
float[] a = new float[1]{20000}; //聲明 float 類型陣列，存放運動距離
```

調用 WriteList\_F32，設置運動距離，如下所示：

```
Wrapper. WriteList_F32(a,0,44503,1);
```

調用 Write\_DOUT，指令對應的 ModBUS 地址寫 1，如下所示：

```
Wrapper. Write_DOUT(9002,1);
```

結果：

軸 1 執行單軸相對運動一次，運動距離為 20000 個脈衝當量。

## 2.8.2 JOG 運動指令

說 明：

當上位機寫 1 到“指令對應的 ModBUS 位址”時，對應的軸會執行 JOG 運動。當上位機寫 0 到“指令對應的 ModBUS 位址”時，對應的軸會停止當前的 JOG 運動。

| 說明             | 指令對應的 ModBUS 地址(資料類型：Bit) |
|----------------|---------------------------|
| 軸 0 正向 JOG 運動  | 9033                      |
| 軸 1 正向 JOG 運動  | 9034                      |
| .....          |                           |
| 軸 15 正向 JOG 運動 | 9048                      |
|                |                           |
| 軸 0 負向 JOG 運動  | 9065                      |
| 軸 1 負向 JOG 運動  | 9066                      |
| .....          |                           |
| 軸 15 負向 JOG 運動 | 9080                      |

#### 例 程：

目的：操作 ModBUS 地址，上位機按下 JOG 按鈕使軸 1 負向 JOG 運動，鬆開按鈕停止軸 1 的 JOG 運動

步驟：

按下按鈕觸發 mousedown 事件，事件中調用 Write\_DOUT，指令對應的 ModBUS 地址寫 1，如下所示：

```
Wrapper. Write_DOUT(9066,1); //寫 1 使軸 1 開始 JOG 負向運動
```

鬆開按鈕觸發 mouseup 事件，事件中調用 Write\_DOUT，指令對應的 ModBUS 地址寫 0，如下所示：

```
Wrapper. Write_DOUT(9066,0); //寫 0 使軸 1 停止 JOG 運動
```

結果：按下 JOG 按鈕，軸 1 開始負向 JOG 運動，鬆開按鈕則運動停止。

### 2.8.3 Home 運動指令

說 明：

當上位機寫 1 到“指令對應的 ModBUS 位址”時，對應的軸會執行回原點運動。當上位機寫 0 到“指令對應的 ModBUS 位址”時，控制器不做任何處理。

| 說明 | 指令對應的 ModBUS 地址(資料類型：Bit) |
|----|---------------------------|
|----|---------------------------|

|              |      |
|--------------|------|
| 軸 0 正向回原點運動  | 9097 |
| 軸 1 正向回原點運動  | 9098 |
| .....        |      |
| 軸 15 正向回原點運動 | 9112 |
|              |      |
| 軸 0 負向回原點運動  | 9129 |
| 軸 1 負向回原點運動  | 9130 |
| .....        |      |
| 軸 15 負向回原點運動 | 9144 |

#### 例 程：

目的：操作 ModBUS 地址，上位機按下 Home 按鈕使軸 1 負向 Home 運動

步驟：

按下 Home 按鈕觸發 click 事件，事件中調用 Write\_DOUT，指令對應的 ModBUS 地址寫 1，如下所示：

```
Wrapper. Write_DOUT(9130,1); //寫 1 使軸 1 開始 Home 負向運動
```

結果：按下 Home 按鈕，軸 1 開始負向 Home 運動。

### 2.8.4 停止運動指令

#### 說 明：

當上位機寫 1 到“指令對應的 ModBUS 位址”時，控制器會控制對應的軸停止運動。當上位機寫 0 到“指令對應的 ModBUS 位址”時，控制器不做任何處理。

| 說明        | 指令對應的 ModBUS 地址(資料類型：Bit) |
|-----------|---------------------------|
| 停止軸 0 運動  | 9161                      |
| 停止軸 1 運動  | 9162                      |
| .....     |                           |
| 停止軸 15 運動 | 9176                      |

#### 例 程：

目的：操作 ModBUS 地址，上位機按下 Stop 按鈕使軸 1 停止運動

步驟：

按下 **Stop** 按鈕觸發 **click** 事件，事件中調用 **Write\_DOUT**，指令對應的 ModBUS 地址寫 **1**，如下所示：

```
Wrapper. Write_DOUT(9162,1); //寫 1 使軸 1 停止運動
```

結果：按下 **Stop** 按鈕，軸 **1** 停止運動。

## 第 3 章 附錄

### 3.1 錯誤代碼說明

#### 3.1.1 RUN\_ERROR 錯誤代碼表

| 錯誤代碼       | 說明                   |
|------------|----------------------|
| 0x00000000 | SUCCESS              |
| 0x80000000 | InvalidDevNumber     |
| 0x80000001 | DevRegDataLost       |
| 0x80000002 | LoadDllFailed        |
| 0x80000003 | GetProcAddressFailed |
| 0x80000004 | MemAllocateFailed    |
| 0x80000005 | InvalidHandle        |
| 0x80000006 | CreateFileFailed     |
| 0x80000007 | OpenEventFailed      |
| 0x80000008 | EventTimeOut         |
| 0x80000009 | InvalidInputParam    |
| 0x8000000a | PropertyIDNotSupport |
| 0x8000000b | PropertyIDReadOnly   |
| 0x8000000c | ConnectWinIrqFailed  |
| 0x8000000d | InvalidAxCfgVel      |
| 0x8000000e | InvalidAxCfgAcc      |
| 0x8000000f | InvalidAxCfgDec      |
| 0x80000010 | InvalidAxCfgJerk     |
| 0x80000011 | InvalidAxParVelLow   |
| 0x80000012 | InvalidAxParVelHigh  |
| 0x80000013 | InvalidAxParAcc      |
| 0x80000014 | InvalidAxParDec      |
| 0x80000015 | InvalidAxParJerk     |
| 0x80000016 | InvalidAxPulseInMode |

|            |                       |
|------------|-----------------------|
| 0x80000017 | InvalidAxPulseOutMode |
| 0x80000018 | InvalidAxAlarmEn      |
| 0x80000019 | InvalidAxAlarmLogic   |
| 0x8000001a | InvalidAxInPEn        |
| 0x8000001b | InvalidAxInPLogic     |
| 0x8000001c | InvalidAxHLmtEn       |
| 0x8000001d | InvalidAxHLmtLogic    |
| 0x8000001e | InvalidAxHLmtReact    |
| 0x8000001f | InvalidAxSLmtPEn      |
| 0x80000020 | InvalidAxSLmtPReact   |
| 0x80000021 | InvalidAxSLmtPValue   |
| 0x80000022 | InvalidAxSLmtMEn      |
| 0x80000023 | InvalidAxSLmtMReact   |
| 0x80000024 | InvalidAxSLmtMValue   |
| 0x80000025 | InvalidAxOrgLogic     |
| 0x80000026 | InvalidAxOrgEnable    |
| 0x80000027 | InvalidAxEzLogic      |
| 0x80000028 | InvalidAxEzEnable     |
| 0x80000029 | InvalidAxEzCount      |
| 0x8000002a | InvalidAxState        |
| 0x8000002b | InvalidAxInEnable     |
| 0x8000002c | InvalidAxSvOnOff      |
| 0x8000002d | InvalidAxDistance     |
| 0x8000002e | InvalidAxPosition     |
| 0x8000002f | InvalidAxHomeModeKw   |
| 0x80000030 | InvalidAxCntInGp      |
| 0x80000031 | AxInGpNotFound        |
| 0x80000032 | AxIsInOtherGp         |
| 0x80000033 | AxCannotIntoGp        |



|            |                         |
|------------|-------------------------|
| 0x80000034 | GpInDevNotFound         |
| 0x80000035 | InvalidGpCfgVel         |
| 0x80000036 | InvalidGpCfgAcc         |
| 0x80000037 | InvalidGpCfgDec         |
| 0x80000038 | InvalidGpCfgJerk        |
| 0x80000039 | InvalidGpParVelLow      |
| 0x8000003a | InvalidGpParVelHigh     |
| 0x8000003b | InvalidGpParAcc         |
| 0x8000003c | InvalidGpParDec         |
| 0x8000003d | InvalidGpParJerk        |
| 0x8000003e | JerkNotSupport          |
| 0x8000003f | ThreeAxNotSupport       |
| 0x80000040 | DevIpoNotFinished       |
| 0x80000041 | InvalidGpState          |
| 0x80000042 | OpenFileFailed          |
| 0x80000043 | InvalidPathCnt          |
| 0x80000044 | InvalidPathHandle       |
| 0x80000045 | InvalidPath             |
| 0x80000046 | IoctlError              |
| 0x80000047 | AmnetRingUsed           |
| 0x80000048 | DeviceNotOpened         |
| 0x80000049 | InvalidRing             |
| 0x8000004a | InvalidSlaveIP          |
| 0x8000004b | InvalidParameter        |
| 0x8000004c | InvalidGpCenterPosition |
| 0x8000004d | InvalidGpEndPosition    |
| 0x8000004e | InvalidAddress          |
| 0x8000004f | DeviceDisconnect        |
| 0x80000050 | DataOutBufExceeded      |

|            |                           |
|------------|---------------------------|
| 0x80000051 | SlaveDeviceNotMatch       |
| 0x80000052 | SlaveDeviceError          |
| 0x80000053 | SlaveDeviceUnknow         |
| 0x80000054 | FunctionNotSupport        |
| 0x80000055 | InvalidPhysicalAxis       |
| 0x80000056 | InvalidVelocity           |
| 0x80000057 | InvalidAxPulseInLogic     |
| 0x80000058 | InvalidAxPulseInSource    |
| 0x80000059 | InvalidAxErcLogic         |
| 0x8000005a | InvalidAxErcOnTime        |
| 0x8000005b | InvalidAxErcOffTime       |
| 0x8000005c | InvalidAxErcEnableMode    |
| 0x8000005d | InvalidAxSdEnable         |
| 0x8000005e | InvalidAxSdLogic          |
| 0x8000005f | InvalidAxSdReact          |
| 0x80000060 | InvalidAxSdLatch          |
| 0x80000061 | InvalidAxHomeResetEnable  |
| 0x80000062 | InvalidAxBacklashEnable   |
| 0x80000063 | InvalidAxBacklashPulses   |
| 0x80000064 | InvalidAxVibrationEnable  |
| 0x80000065 | InvalidAxVibrationRevTime |
| 0x80000066 | InvalidAxVibrationFwdTime |
| 0x80000067 | InvalidAxAlarmReact       |
| 0x80000068 | InvalidAxLatchLogic       |
| 0x80000069 | InvalidFwMemoryMode       |
| 0x8000006a | InvalidConfigFile         |
| 0x8000006b | InvalidAxEnEvtArraySize   |
| 0x8000006c | InvalidAxEnEvtArray       |
| 0x8000006d | InvalidGpEnEvtArraySize   |

|            |                          |
|------------|--------------------------|
| 0x8000006e | InvalidGpEnEvtArray      |
| 0x8000006f | InvalidIntervalData      |
| 0x80000070 | InvalidEndPosition       |
| 0x80000071 | InvalidAxisSelect        |
| 0x80000072 | InvalidTableSize         |
| 0x80000073 | InvalidGpHandle          |
| 0x80000074 | InvalidCmpSource         |
| 0x80000075 | InvalidCmpMethod         |
| 0x80000076 | InvalidCmpPulseMode      |
| 0x80000077 | InvalidCmpPulseLogic     |
| 0x80000078 | InvalidCmpPulseWidth     |
| 0x80000079 | InvalidPathFunctionID    |
| 0x8000007a | SysBufAllocateFailed     |
| 0x8000007b | SpeedFordFunNotSpported  |
| 0x8000007c | InvalidNormVector        |
| 0x8000007d | InvalidCmpTimeTableCount |
| 0x8000007e | InvalidCmpTime           |
| 0x8000007f | FWDownLoading            |
| 0x80000080 | FWVersionNotMatch        |
| 0x80000081 | InvalidAxParHomeVelLow   |
| 0x80000082 | InvalidAxParHomeVelHigh  |
| 0x80000083 | InvalidAxParHomeAcc      |
| 0x80000084 | InvalidAxParHomeDec      |
| 0x80000085 | InvalidAxParHomeJerk     |
| 0x80000086 | InvalidAxCfgJogVelLow    |
| 0x80000087 | InvalidAxCfgJogVelHigh   |
| 0x80000088 | InvalidAxCfgJogAcc       |
| 0x80000089 | InvalidAxCfgJogDec       |
| 0x8000008a | InvalidAxCfgJogJerk      |

|            |                       |
|------------|-----------------------|
| 0x8000008b | InvalidAxCfgKillDec   |
| 0x8000008c | NotOpenAllAxes        |
| 0x8000008d | NotSetServoComPort    |
| 0x8000008e | OpenComPortFailed     |
| 0x8000008f | ReadComPortTimeOut    |
| 0x80000090 | SetComPortStateFailed |
| 0x80000091 | SevroTypeNotSupport   |
| 0x80000092 | ReadComBufFailed      |
| 0x80000096 | SlaveIOUpdateError    |
| 0x80000097 | NoSlaveDevFound       |
| 0x80000098 | MasterDevNotOpen      |
| 0x80000099 | MasterRingNotOpen     |
| 0x800000c8 | InvalidDIPort         |
| 0x800000c9 | InvalidDOPort         |
| 0x800000ca | InvalidDOValue        |
| 0x800000cb | CreateEventFailed     |
| 0x800000cc | CreateThreadFailed    |
| 0x800000cd | InvalidHomeModeEx     |
| 0x800000ce | InvalidDirMode        |
| 0x800000cf | AxHomeMotionFailed    |
| 0x800000d0 | ReadFileFailed        |
| 0x800000d1 | PathBufIsFull         |
| 0x800000d2 | PathBufIsEmpty        |
| 0x800000d3 | GetAuthorityFailed    |
| 0x800000d4 | GpIDAllocatedFailed   |
| 0x800000d5 | FirmWareDown          |
| 0x800000d6 | InvalidGpRadius       |
| 0x800000d7 | InvalidAxCmd          |
| 0x800000d8 | InvalidaxExtDrv       |

|            |                                        |
|------------|----------------------------------------|
| 0x800000d9 | InvalidGpMovCmd                        |
| 0x800000da | SpeedCurveNotSupported                 |
| 0x800000db | InvalidCounterNo                       |
| 0x800000dc | InvalidPathMoveMode                    |
| 0x800000dd | PathSelStartCantRunInSpeedForewareMode |
| 0x800000de | InvalidCamTableID                      |
| 0x800000df | InvalidCamPointRange                   |
| 0x800000e0 | CamTableIsEmpty                        |
| 0x800000e1 | InvalidPlaneVector                     |
| 0x800000e2 | MasAxIDSameSlvAxID                     |
| 0x800000e3 | InvalidGpRefPlane                      |
| 0x800000e4 | InvalidAxModuleRange                   |
| 0x800000e5 | DownloadFileFailed                     |
| 0x800000e6 | InvalidFileLength                      |
| 0x800000e7 | InvalidCmpCnt                          |
| 0x800000e8 | JerkExceededMaxValue                   |
| 0x800000e9 | AbsMotionNotSupport                    |
| 0x800000ea | invalidAiRange                         |
| 0x800000eb | AIScaleFailed                          |
| 0x800000ec | AxInRobot                              |
| 0x800000ed | Invalid3DarcFlat                       |
| 0x800000ee | InvalidIpoMap                          |
| 0x800000ef | DataSizeNotCorrect                     |
| 0x800000f0 | AxisNotFound                           |
| 0x800000f1 | InvalidPathVelHigh                     |
|            |                                        |
| 0x80002000 | HlmtPExceeded                          |
| 0x80002001 | HlmtNExceeded                          |
| 0x80002002 | SlmtPExceeded                          |

|            |                                   |
|------------|-----------------------------------|
| 0x80002003 | SlmtNExceeded                     |
| 0x80002004 | AlarmHappened                     |
| 0x80002005 | EmgHappened                       |
| 0x80002006 | TimeLmtExceeded                   |
| 0x80002007 | DistLmtExceeded                   |
| 0x80002008 | InvalidPositionOverride           |
| 0x80002009 | OperationErrorHappened            |
| 0x8000200a | SimultaneousStopHappened          |
| 0x8000200b | OverflowInPAPB                    |
| 0x8000200c | OverflowInIPO                     |
| 0x8000200d | STPHappened                       |
| 0x8000200e | SDHappened                        |
| 0x8000200f | AxisNoCmpDataLeft                 |
|            |                                   |
| 0x10000001 | Warning_AxWasInGp                 |
| 0x10000002 | Warning_GpInconsistRate           |
| 0x10000003 | Warning_GpInconsistPPU            |
| 0x10000004 | Warning_GpMoveDistanceCanntBeZero |
|            |                                   |
| 0x80004001 | DevEvtTimeOut                     |
| 0x80004002 | DevNoEvt                          |
|            |                                   |
| 0x80005001 | ERR_SYS_TIME_OUT                  |
| 0x80005002 | Dsp_PropertyIDNotSupport          |
| 0x80005003 | Dsp_PropertyIDReadOnly            |
| 0x80005004 | Dsp_InvalidParameter              |
| 0x80005005 | Dsp_DataOutBufExceeded            |
| 0x80005006 | Dsp_FunctionNotSupport            |
| 0x80005007 | Dsp_InvalidConfigFile             |

|            |                             |
|------------|-----------------------------|
| 0x80005008 | Dsp_InvalidIntervalData     |
| 0x80005009 | Dsp_InvalidTableSize        |
| 0x8000500a | Dsp_InvalidTableID          |
| 0x8000500b | Dsp_DataIndexExceedBufSize  |
| 0x8000500c | Dsp_InvalidCompareInterval  |
| 0x8000500d | Dsp_InvalidCompareRange     |
| 0x8000500e | Dsp_PropertyIDWriteOnly     |
| 0x8000500f | Dsp_NcError                 |
| 0x80005010 | Dsp_CamTableIsInUse         |
| 0x80005011 | Dsp_EraseBlockFailed        |
| 0x80005012 | Dsp_ProgramFlashFailed      |
| 0x80005013 | Dsp_WatchdogError           |
| 0x80005014 | Dsp_ReadPrivateOverMaxTimes |
| 0x80005015 | Dsp_InvalidPrivateID        |
| 0x80005016 | Dsp_DataNotReady            |
| 0x80005017 | Dsp_LastOperationNotOver    |
| 0x80005018 | Dsp_WritePrivateTimeout     |
| 0x80005019 | Dsp_FwIsDownloading         |
| 0x80005020 | Dsp_FwDownloadStepError     |
|            |                             |
| 0x80005101 | Dsp_InvalidAxCfgVel         |
| 0x80005102 | Dsp_InvalidAxCfgAcc         |
| 0x80005103 | Dsp_InvalidAxCfgDec         |
| 0x80005104 | Dsp_InvalidAxCfgJerk        |
| 0x80005105 | Dsp_InvalidAxParVelLow      |
| 0x80005106 | Dsp_InvalidAxParVelHigh     |
| 0x80005107 | Dsp_InvalidAxParAcc         |
| 0x80005108 | Dsp_InvalidAxParDec         |
| 0x80005109 | Dsp_InvalidAxParJerk        |

|            |                                 |
|------------|---------------------------------|
| 0x8000510a | Dsp_InvalidAxPptValue           |
| 0x8000510b | Dsp_InvalidAxState              |
| 0x8000510c | Dsp_InvalidAxSvOnOff            |
| 0x8000510d | Dsp_InvalidAxDistance           |
| 0x8000510e | Dsp_InvalidAxPosition           |
| 0x8000510f | Dsp_InvalidAxHomeMode           |
| 0x80005110 | Dsp_InvalidPhysicalAxis         |
| 0x80005111 | Dsp_HlmtPExceeded               |
| 0x80005112 | Dsp_HlmtNExceeded               |
| 0x80005113 | Dsp_SlmtPExceeded               |
| 0x80005114 | Dsp_SlmtNExceeded               |
| 0x80005115 | Dsp_AlarmHappened               |
| 0x80005116 | Dsp_EmgHappened                 |
| 0x80005117 | Dsp_CmdValidOnlyInConstSec      |
| 0x80005118 | Dsp_InvalidAxCmd                |
| 0x80005119 | Dsp_InvalidAxHomeDirMode        |
| 0x8000511a | Dsp_AxisMustBeModuloAxis        |
| 0x8000511b | Dsp_AxIdCantSameAsMasId         |
| 0x8000511c | Dsp_CantResetPosiOfMasAxis      |
| 0x8000511d | Dsp_InvalidAxExtDrvOperation    |
| 0x8000511e | Dsp_AxAccExceededMaxAcc         |
| 0x8000511f | Dsp_AxVelExceededMaxVel         |
| 0x80005120 | Dsp_NotEnoughPulseForChgV       |
| 0x80005121 | Dsp_NewVelMustGreaterThanVelLow |
| 0x80005122 | Dsp_InvalidAxGearMode           |
| 0x80005123 | Dsp_InvalidGearRatio            |
| 0x80005124 | Dsp_InvalidPWMDDataCount        |
| 0x80005125 | Dsp_InvalidAxPWMFreq            |
| 0x80005126 | Dsp_InvalidAxPWMDuty            |



|            |                                |
|------------|--------------------------------|
| 0x80005127 | Dsp_AxGantryExceedMaxDiffValue |
| 0x80005128 | Dsp_ChanelIsDisable            |
| 0x80005129 | Dsp_ChanelBufferIsFull         |
| 0x80005130 | Dsp_ChanelBufferIsEmpty        |
| 0x80005131 | Dsp_InvalidDoChanelID          |
| 0x80005132 | Dsp_LatchHappened              |
|            |                                |
| 0x80005201 | Dsp_InvalidAxCntInGp           |
| 0x80005202 | Dsp_AxInGpNotFound             |
| 0x80005203 | Dsp_AxIsInOtherGp              |
| 0x80005204 | Dsp_AxCannotIntoGp             |
| 0x80005205 | Dsp_GpInDevNotFound            |
| 0x80005206 | Dsp_InvalidGpCfgVel            |
| 0x80005207 | Dsp_InvalidGpCfgAcc            |
| 0x80005208 | Dsp_InvalidGpCfgDec            |
| 0x80005209 | Dsp_InvalidGpCfgJerk           |
| 0x8000520a | Dsp_InvalidGpParVelLow         |
| 0x8000520b | Dsp_InvalidGpParVelHigh        |
| 0x8000520c | Dsp_InvalidGpParAcc            |
| 0x8000520d | Dsp_InvalidGpParDec            |
| 0x8000520e | Dsp_InvalidGpParJerk           |
| 0x8000520f | Dsp_JerkNotSupport             |
| 0x80005210 | Dsp_ThreeAxNotSupport          |
| 0x80005211 | Dsp_DevIpoNotFinished          |
| 0x80005212 | Dsp_InvalidGpState             |
| 0x80005213 | Dsp_OpenFileFailed             |
| 0x80005214 | Dsp_InvalidPathCnt             |
| 0x80005215 | Dsp_InvalidPathHandle          |
| 0x80005216 | Dsp_InvalidPath                |

|            |                               |
|------------|-------------------------------|
| 0x80005217 | Dsp_GpSlavePositionOverMaster |
| 0x80005218 | Dsp_GpPathBufferOverflow      |
| 0x80005219 | Dsp_InvalidPathFunctionID     |
| 0x8000521a | Dsp_SysBufAllocateFailed      |
| 0x8000521b | Dsp_InvalidGpCenterPosition   |
| 0x8000521c | Dsp_InvalidGpEndPosition      |
| 0x8000521d | Dsp_InvalidGpCmd              |
| 0x8000521e | Dsp_AxHasBeenInInGp           |
| 0x8000521f | Dsp_ThreeAxNotSupport         |
| 0x80005220 | Dsp_InvalidPathRange          |
| 0x80005221 | Dsp_InvalidNormVector         |

### 3.1.2 SYSTEM\_ERROR 錯誤代碼表

| 錯誤代碼       | 說明                        |
|------------|---------------------------|
| 0          | SUCCESS                   |
| 0x90000000 | AMI_NULL_PROJECT_EXIST    |
| 0x90000001 | AMI_INVALID_INPUT_PARAMS  |
| 0x90000002 | AMI_INVALID_RETURN        |
| 0x90000003 | AMI_INVALID_CTRL_MODE     |
| 0x90000004 | AMI_CONTROLLER_LOCKED     |
| 0x90000005 | AMI_GET_MAC_FAILED        |
| 0x90000006 | AMI_INVALID_COMMAND       |
| 0x90000007 | AMI_SET_MEM_FAILED        |
| 0x90000008 | AMI_GET_VERSION_FAILED    |
| 0x9000000a | AMI_CTRL_ENCODED_ALREADY  |
| 0x9000000b | AMI_CTRL_INVALID_PASSWORD |
| 0x9000000c | AMI_GET_VARIABLE_FAILED   |
| 0x9000000d | AMI_NUM_CONVERT_FAILED    |
| 0x90000032 | AMI_ACTION_NOT_ALLOWED    |

| 0x90000064 | AMI SOCK_TIME_OUT           |
|------------|-----------------------------|
| 0x90000065 | AMI_LOAD_FILE_FAILED        |
| 0x90000066 | AMI_DOWN_FILE_FAILED        |
| 0x90000067 | AMI_LOAD_PROJECT_FAILED     |
| 0x90000068 | AMI_DOWN_PROJECT_FAILED     |
| 0x90000069 | AMI SOCK_ALREADY_CONNECTED  |
| 0x9000006A | AMI SOCK_COMMU_FAILED       |
| 0x90000096 | AMI_CONNECTION_FAILED       |
| 0x90000097 | AMI_DISCONNECTION_FAILED    |
| 0x90000098 | AMI_SEND_COMMAND_TIMEOUT    |
|            |                             |
| 0x900000C8 | AMI_OPEN_FILE_FAILED        |
| 0x900000C9 | AMI_CREATE_FILE_FAILED      |
| 0x900000CA | AMI_REMOVE_FILE_FAILED      |
| 0x900000CB | AMI_PATH_NOT_EXIST          |
| 0x900000CC | AMI_SET_NON_BLOCK_FAILED    |
| 0x900000CD | AMI_SET_BLOCK_FAILED        |
| 0x900000CE | AMI_CFG_FILE_NOT_EXISTS     |
| 0x900000CF | AMI_REF_FILE_NOT_EXISTS     |
| 0x900000D0 | AMI_HEAD_FILE_NOT_EXISTS    |
| 0x900000D1 | AMI_FILE_NOT_EXISTS         |
| 0x900000D2 | AMI_FILE_INVALID_FORMAT     |
| 0x900000DC | AMI_PRJ_FILE_LOAD_FAILED    |
| 0x900000DD | AMI_SOURCE_FILE_NOT_EXISTS  |
| 0x900000DE | AMI_DST_FILE_EXISTS_ALREADY |
| 0x900000FA | AMI_XML_LOAD_FAILED         |
| 0x900000FB | AMI_XML_CHECK_FAILED        |
| 0x900000FC | AMI_XML_SAVE_FAILED         |

|            |                            |
|------------|----------------------------|
| 0x900000FD | AMI_XML_ADD_FAILED         |
| 0x900000FE | AMI_XML_DELETE_FAILED      |
| 0x900000FF | AMI_XML_CREATE_FAILED      |
| 0x90000100 | AMI_XML_INVALID_ELEMENT    |
|            |                            |
| 0x9000012C | AMI_TASK_NOT_EXIST         |
| 0x9000012D | AMI_FORK_PROCESS_FAILED    |
| 0x9000012E | AMI_POPEN_FILE_FAILED      |
| 0x9000012F | AMI_STILL_RUNNING          |
| 0x90000130 | AMI_NOT_IN_IDLE            |
| 0x90000131 | AMI_GET_NO_ERROR           |
| 0x90000132 | AMI_GET_NO_INFO            |
| 0x90000133 | AMI_GET_MSG_NOTFINISHED    |
| 0x90000134 | AMI_CREAT_PIPE_FAILED      |
| 0x90000136 | AMI_RUN_FAILED             |
| 0x90000137 | AMI_STOP_FAILED            |
| 0x90000138 | AMI_NOT_RUNNING            |
| 0x90000140 | AMI_DB_INIT_FAILED         |
| 0x90000141 | AMI_DB_COMPILE_FAILED      |
| 0x90000142 | AMI_DB_BREAKPOINT_FAILED   |
| 0x90000143 | AMI_DB_CLEARPOINT_FAILED   |
| 0x90000144 | AMI_DB_DELETEPOINTS_FAILED |
| 0x90000145 | AMI_DB_RUN_FAILED          |
| 0x90000146 | AMI_DB_CONTINUE_FAILED     |
| 0x90000147 | AMI_DB_NEXT_FAILED         |
| 0x90000148 | AMI_DB_PROGRAM_NOT_RUN     |
| 0x90000149 | AMI_DB_STOP_FAILED         |
| 0x9000014A | AMI_DB_OUT_OF_RANGE        |
| 0x9000014B | AMI_DB_EXIT_NOMARLLY       |

|            |                                    |
|------------|------------------------------------|
| 0x9000014C | AMI_DB_GET_LOCAL_VAR_FAILED        |
| 0x9000014D | AMI_DB_NOT_READY                   |
| 0x9000014E | AMI_DB_ALREADY_PAUSED              |
| 0x9000014F | AMI_GET_RUNNING_TASKLIST_FAILED    |
|            |                                    |
| 0x90000190 | AMI_MB_ILLEGAL_FUNCTION            |
| 0x90000191 | AMI_MB_CRC_FAILED                  |
| 0x90000192 | AMI_MB_ILLEGAL_LENGTH              |
|            |                                    |
| 0x900001F4 | AMI_MEM_UPDATE_VR_MBADDR_ERROR     |
| 0x900001F5 | AMI_MEM_UPDATE_TABLE_MBADDR_ERROR  |
| 0x900001F6 | AMI_MEM_UPDATE_DO_INIT_VALUE_ERROR |
|            |                                    |
| 0x90000258 | AMI_BASIC_RESET_ERROR              |
| 0x90000259 | AMI_BASIC_INITIAL_ERROR            |
| 0x9000025A | AMI_BASIC_REFRESH_ERROR            |
| 0x9000025B | AMI_BASIC_GET_OFFSET_VALUE_FAILED  |
|            |                                    |
| 0xb0000000 | AMI_GetSharedMemFailed             |
| 0xb0000001 | AMI_GetTaskNameFailed              |
| 0xb0000002 | AMI_IsNotInitialized               |
| 0xb0000003 | AMI_IsAlreadyInitialized           |
| 0xb0000004 | AMI_LoadXMLFailed                  |
| 0xb0000005 | AMI_ParseXMLFailed                 |
| 0xb0000006 | AMI_CreateDevListFailed            |
| 0xb0000007 | AMI_InitializeDeviceFailed         |
| 0xb0000008 | AMI_InitializeSharedMemFailed      |
| 0xb0000009 | AMI_RefreshSharedMemFailed         |
| 0xb000000a | AMI_SetDeviceCfgFailed             |

|            |                                    |
|------------|------------------------------------|
| 0xb000000b | AMI_IncorrectCommand               |
| 0xb000000c | AMI_DeviceLargerList               |
| 0xb000000d | AMI_SerialPortError                |
| 0xb000000e | AMI_EthernetError                  |
| 0xb000000f | AMI_LogOpenFailed                  |
| 0xb0000010 | AMI_StartTaskFailed                |
| 0xb0000011 | AMI_StopTaskFailed                 |
| 0xb0000012 | AMI_CreatEventFailed               |
| 0xb0000013 | AMI_CreatEThreadsFailed            |
| 0xb0000014 | AMI_AllocatePMotionInfoFiled       |
| 0xb0000015 | AMI_FAILEDTOCHECKEVENT             |
| 0xb0000016 | AMI_FailedCloseCheckingEventThread |
|            |                                    |
| 0xb0001000 | AMI_CardsNotFound                  |
| 0xb0001001 | AMI_MotionBoardIDNotFound          |
| 0xb0001002 | AMI_AxesOrGroupCountNotFound       |
| 0xb0001003 | AMI_AxisIDorPhyIDNotFound          |
| 0xb0001004 | AMI_GroupNotFound                  |
| 0xb0001005 | AMI_AxisInfoError                  |
| 0xb0001006 | AMI_MotionDeviceCountError         |
| 0xb0001007 | AMI_InputBoardIDNotFound           |
| 0xb0001008 | AMI_InputDeviceCountError          |
| 0xb0001009 | AMI_InDAQdeviceCountError          |
| 0xb000100a | AMI_OutputBoardIDNotFound          |
| 0xb000100b | AMI_OutputDeviceCountError         |
| 0xb000100c | AMI_OutDAQdeviceCountError         |
| 0xb000100d | AMI_ActualDeviceCountError         |
|            |                                    |
| 0xb0002000 | AMI_WrongAxisIndex                 |

|            |                              |
|------------|------------------------------|
| 0xb0002001 | AMI_WrongDoIndex             |
| 0xb0002002 | AMI_WrongDiIndex             |
| 0xb0002003 | AMI_NoAxis                   |
| 0xb0002004 | AMI_AxisInDifferentDevice    |
| 0xb0002005 | AMI_WaitModeNotMatch         |
| 0xb0002006 | AMI_MasterAxisIndexError     |
| 0xb0002007 | AMI_GANTRYAxisNotInSameDev   |
| 0xb0002008 | AMI_GEARAxisNotInSameDev     |
| 0xb0002009 | AMI_AddPathAxisCntError      |
| 0xb000200a | AMI_AddPathHELIX3PnotSupport |
|            |                              |
| 0xb0003000 | AMI_EthernetModeError        |
| 0xb0003001 | AMI_EthernetOpened           |
| 0xb0003002 | AMI_EthernetOpenFailed       |
| 0xb0003003 | AMI_EthernetCloseFailed      |
| 0xb0003004 | AMI_EthernetWrongNum         |
| 0xb0003005 | AMI_EthernetNotOpen          |
| 0xb0003006 | AMI_EthernetReadFailed       |
| 0xb0003007 | AMI_EthernetResetFailed      |
| 0xb0003008 | AMI_EthernetWriteFailed      |
| 0xb0003009 | AMI_EthernetReadVRFailed     |
| 0xb000300a | AMI_EthernetWriteVRFailed    |
| 0xb0003100 | AMI_SerialPortWrongID        |
| 0xb0003101 | AMI_SerialPortOpenFailed     |
| 0xb0003102 | AMI_SerialPortCloseFailed    |
| 0xb0003103 | AMI_SerialPortNotOpen        |
| 0xb0003104 | AMI_SerialPortWrongCfg       |
| 0xb0003105 | AMI_SerialPortSetCfgFailed   |
| 0xb0003106 | AMI_SerialPortWriteFailed    |

|            |                             |
|------------|-----------------------------|
| 0xb0003107 | AMI_SerialPortReadFailed    |
| 0xb0003108 | AMI_SerialPortResetFailed   |
| 0xb0003109 | AMI_SerialPortWriteVRFailed |
| 0xb000310a | AMI_SerialPortReadVRFailed  |



## 3.2 ModBUS 地址

Motion Studio 中的變數內建有 ModBUS 位址，用於與外部的通信。變數與 ModBUS 位址的對應關係分五大區塊，如下表：

| 區塊                     | ModBUS 組別        | ModBUS 地址範圍 | 讀/寫   |
|------------------------|------------------|-------------|-------|
| 標準變數—DOUT              | COIL STATUS      | 1~1024      | 可讀/可寫 |
| 標準變數—DI                | INPUT STATUS     | 10001~11024 | 唯讀    |
| 標準變數—狀態<br>(運動控制相關)    | INPUT REGISTER   | 30001~31600 | 唯讀    |
| 標準變數—參數、配置<br>(運動控制相關) | HOLDING REGISTER | 45001~49800 | 可讀/可寫 |
| 內建自訂變數—VR              | HOLDING REGISTER | 40001~44000 | 可讀/可寫 |

注：以下各區塊變數與 ModBUS 對應關係中資料類型說明如下：

- **Bit**：位
- **UINT16**: 16 位元無符號整型
- **INT16**: 16 位整型
- **UINT32**：32 位元無符號整型
- **INT 32**: 32 位整型
- **F32**：32 位浮點型

### 3.2.1 DOUT 對應的 ModBUS 地址

| 變數         | 說明              | ModBUS 地址 | 資料類型 |
|------------|-----------------|-----------|------|
| DOUT(0)    | 索引為 0 的數位量輸出    | 1         | Bit  |
| DOUT(1)    | 索引為 1 的數位量輸出    | 2         | Bit  |
| .....      |                 |           | Bit  |
| DOUT(1023) | 索引為 1023 的數位量輸出 | 1024      | Bit  |

### 3.2.2 DI 對應的 ModBUS 地址

| 變數       | 說明              | ModBUS 地址 | 資料類型 |
|----------|-----------------|-----------|------|
| DI(0)    | 索引為 0 的數位量輸入    | 10001     | Bit  |
| DI(1)    | 索引為 1 的數位量輸入    | 10002     | Bit  |
| .....    |                 |           | Bit  |
| DI(1023) | 索引為 1023 的數位量輸入 | 11024     | Bit  |

### 3.2.3 軸狀態對應的 ModBUS 地址

| 軸狀態   | ModBUS 地址範圍 |
|-------|-------------|
| 軸 0   | 30001~30100 |
| 軸 1   | 30101~30200 |
| ..... |             |
| 軸 15  | 31501~31600 |

#### ● 軸 0 狀態的 ModBUS 地址表

| 變數        | 說明              | ModBUS 地址 | 資料類型   |
|-----------|-----------------|-----------|--------|
| DPOS      | 累計指令位置（理論位置）    | 30001     | F32    |
| MPOS      | 累計回饋位置（實際位置）    | 30003     | F32    |
| STATE     | 當前軸運動狀態         | 30005     | UINT16 |
| M_STATUS  | 暫無作用，預留         | 30006     | UINT32 |
| DSPEED    | 當前軸運動指令速度（理論速度） | 30008     | F32    |
| MIO       | 運動相關 I/O 的狀態    | 30010     | UINT32 |
| RUN_ERROR | 軸錯誤資訊           | 30012     | UINT32 |

#### ● 軸 1 狀態的 ModBUS 地址表

| 變數        | 說明              | ModBUS 地址 | 資料類型   |
|-----------|-----------------|-----------|--------|
| DPOS      | 累計指令位置（理論位置）    | 30101     | F32    |
| MPOS      | 累計回饋位置（實際位置）    | 30103     | F32    |
| STATE     | 當前軸運動狀態         | 30105     | UINT16 |
| M_STATUS  | 暫無作用，預留         | 30106     | UINT32 |
| DSPEED    | 當前軸運動指令速度（理論速度） | 30108     | F32    |
| MIO       | 運動相關 I/O 的狀態    | 30110     | UINT32 |
| RUN_ERROR | 軸錯誤資訊           | 30112     | UINT32 |

#### ● .....

● 軸 15 狀態的 ModBUS 地址表

| 變數        | 說明              | ModBUS 地址 | 資料類型   |
|-----------|-----------------|-----------|--------|
| DPOS      | 累計指令位置（理論位置）    | 31501     | F32    |
| MPOS      | 累計回饋位置（實際位置）    | 31503     | F32    |
| STATE     | 當前軸運動狀態         | 31505     | UINT16 |
| M_STATUS  | 暫無作用，預留         | 31506     | UINT32 |
| DSPEED    | 當前軸運動指令速度（理論速度） | 31508     | F32    |
| MIO       | 運動相關 I/O 的狀態    | 31510     | UINT32 |
| RUN_ERROR | 軸錯誤資訊           | 31512     | UINT32 |

### 3.2.4 軸參數、配置對應的 ModBUS 地址

| 軸參數、配置 | ModBUS 地址範圍 |
|--------|-------------|
| 軸 0    | 45001~45300 |
| 軸 1    | 45301~45600 |
| .....  |             |
| 軸 15   | 49501~49800 |

● 軸 0 參數、配置的 ModBUS 地址表

| 變數              | 說明              | ModBUS 地址 | 資料類型   |
|-----------------|-----------------|-----------|--------|
| VL              | 軸初速度            | 45001     | F32    |
| VH              | 軸運行速度           | 45003     | F32    |
| ACC             | 軸加速度            | 45005     | F32    |
| DEC             | 軸減速度            | 45007     | F32    |
| JK              | 速度曲線類型：S 型或 T 型 | 45009     | F32    |
| MAXVEL          | 軸運行速度限值         | 45011     | F32    |
| MAXACC          | 軸加速度限值          | 45013     | F32    |
| MAXDEC          | 軸減速度限值          | 45015     | F32    |
| MAXJK           | 暫無作用，預留         | 45017     | F32    |
| INSTOP_DEC      | INSTOP 功能中的減速度  | 45019     | F32    |
| UNIT_NUM        | 脈衝當量分子          | 45031     | UINT32 |
| UNIT_DENOM      | 脈衝當量分母          | 45033     | UINT32 |
| HOME_CROSS      | 回原點運行中的跨越距離     | 45051     | F32    |
| HOME_EX         | 暫無作用，預留         | 45053     | UINT16 |
| ORG_LOGIC       | ORG 信號的有效邏輯電平   | 45054     | UINT16 |
| ORG_MODE        | 回原點運動結束時的停止模式   | 45055     | UINT16 |
| EZ_LOGIC        | Z 相輸入信號的有效邏輯電平  | 45056     | UINT16 |
| HOME_RESET      | 回原點後清零位置值功能     | 45057     | UINT16 |
| HOME_OFFSETDIST | 回原點成後的偏移距離      | 45058     | F32    |
| HOME_OFFSETVEL  | 回原點成後偏移距離的移動速   | 45060     | F32    |

|              |                 |       |        |
|--------------|-----------------|-------|--------|
|              | 度               |       |        |
| HOME_MODE    | 回原點模式           | 45062 | UINT16 |
| HOME_VL      | 回原點運動的初速度       | 45063 | F32    |
| HOME_VH      | 回原點運動的運行速度      | 45065 | F32    |
| HOME_ACC     | 回原點運動的加速度       | 45067 | F32    |
| HOME_DEC     | 回原點運動的減速度       | 45069 | F32    |
| HOME_JK      | 回原點運動的速度曲線類型    | 45071 | F32    |
| JOG_VLTIME   | JOG 運動低段速度運行的時間 | 45081 | UINT32 |
| JOG_VL       | JOG 運動低段速度      | 45083 | F32    |
| JOG_VH       | JOG 運動高段速度      | 45085 | F32    |
| JOG_ACC      | JOG 運動的加速度      | 45087 | F32    |
| JOG_DEC      | JOG 運動的減速度      | 45089 | F32    |
| EL_EN        | 使能硬體限位元功能       | 45101 | UINT16 |
| EL_LOGIC     | 硬體限位元信號的有效邏輯電平  | 45102 | UINT16 |
| EL_MODE      | 硬限位元觸發時的停止模式    | 45103 | UINT16 |
| PEL_TOL_EN   | 正方向硬極限容差功能使能    | 45104 | UINT16 |
| PEL_TOL      | 正方向硬極限容差值       | 45105 | UINT32 |
| NEL_TOL_EN   | 負方向硬極限容差功能使能    | 45107 | UINT16 |
| NEL_TOL      | 負方向硬極限容差值       | 45108 | UINT32 |
| PIN_MODE     | 編碼器脈衝輸入類型       | 45121 | UINT16 |
| PIN_LOGIC    | 編碼器脈衝輸入邏輯反相     | 45122 | UINT16 |
| PIN_MAXFREQ  | 編碼器脈衝輸入的最高頻率限值  | 45123 | UINT16 |
| POUT_MODE    | 指令脈衝輸出類型        | 45131 | UINT16 |
| POUT_REVERSE | 指令脈衝輸出邏輯反相      | 45132 | UINT16 |
| ALM_FILTER   | 報警(ALM) 埠的濾波時間  | 45141 | UINT16 |
| NEL_FILTER   | 負方向硬限位埠濾波時間     | 45142 | UINT16 |
| PEL_FILTER   | 正方向硬限位埠濾波時間     | 45143 | UINT16 |

|            |                                |       |        |
|------------|--------------------------------|-------|--------|
| ORG_FILTER | 原點信號埠濾波時間                      | 45144 | UINT16 |
| IN1_FILTER | IN1 埠的濾波時間                     | 45145 | UINT16 |
| IN2_FILTER | IN2 埠的濾波時間                     | 45146 | UINT16 |
| IN4_FILTER | IN4 埠的濾波時間                     | 45147 | UINT16 |
| IN5_FILTER | IN5 埠的濾波時間                     | 45148 | UINT16 |
| EXT_SRC    | 哪個軸的外部驅動輸入埠作為外部驅動信號源           | 45161 | UINT16 |
| EXT_EN     | 暫無作用，預留                        | 45162 | UINT16 |
| EXT_MODE   | 手輪模式外部驅動的脈衝輸入模式                | 45163 | UINT16 |
| EXT_PULSE  | 手輪模式外部驅動時，每個手輪脈衝輸入對應多少個指令脈衝輸出值 | 45164 | UINT32 |

● 軸 1 參數、配置的 ModBUS 地址表

| 變數         | 說明              | ModBUS 地址 | 資料類型   |
|------------|-----------------|-----------|--------|
| VL         | 軸初速度            | 45301     | F32    |
| VH         | 軸運行速度           | 45303     | F32    |
| ACC        | 軸加速度            | 45305     | F32    |
| DEC        | 軸減速度            | 45307     | F32    |
| JK         | 速度曲線類型：S 型或 T 型 | 45309     | F32    |
| MAXVEL     | 軸運行速度限值         | 45311     | F32    |
| MAXACC     | 軸加速度限值          | 45313     | F32    |
| MAXDEC     | 軸減速度限值          | 45315     | F32    |
| MAXJK      | 暫無作用，預留         | 45317     | F32    |
| INSTOP_DEC | INSTOP 功能中的減速度  | 45319     | F32    |
| UNIT_NUM   | 脈衝當量分子          | 45331     | UINT32 |
| UNIT_DENOM | 脈衝當量分母          | 45333     | UINT32 |
| HOME_CROSS | 回原點運行中的跨越距離     | 45351     | F32    |
| HOME_EX    | 暫無作用，預留         | 45353     | UINT16 |



|                 |                 |       |        |
|-----------------|-----------------|-------|--------|
| ORG_LOGIC       | ORG 信號的有效邏輯電平   | 45354 | UINT16 |
| ORG_MODE        | 回原點運動結束時的停止模式   | 45355 | UINT16 |
| EZ_LOGIC        | Z 相輸入信號的有效邏輯電平  | 45356 | UINT16 |
| HOME_RESET      | 回原點後清零位置值功能     | 45357 | UINT16 |
| HOME_OFFSETDIST | 回原點成後的偏移距離      | 45358 | F32    |
| HOME_OFFSETVEL  | 回原點成後偏移距離的移動速度  | 45360 | F32    |
| HOME_MODE       | 回原點模式           | 45362 | UINT16 |
| HOME_VL         | 回原點運動的初速度       | 45363 | F32    |
| HOME_VH         | 回原點運動的運行速度      | 45365 | F32    |
| HOME_ACC        | 回原點運動的加速度       | 45367 | F32    |
| HOME_DEC        | 回原點運動的減速度       | 45369 | F32    |
| HOME_JK         | 回原點運動的速度曲線類型    | 45371 | F32    |
| JOG_VLTIME      | JOG 運動低段速度運行的時間 | 45381 | UINT32 |
| JOG_VL          | JOG 運動低段速度      | 45383 | F32    |
| JOG_VH          | JOG 運動高段速度      | 45385 | F32    |
| JOG_ACC         | JOG 運動的加速度      | 45387 | F32    |
| JOG_DEC         | JOG 運動的減速度      | 45389 | F32    |
| EL_EN           | 使能硬體限位元功能       | 45401 | UINT16 |
| EL_LOGIC        | 硬體限位元信號的有效邏輯電平  | 45402 | UINT16 |
| EL_MODE         | 硬限位元觸發時的停止模式    | 45403 | UINT16 |
| PEL_TOL_EN      | 正方向硬極限容差功能使能    | 45404 | UINT16 |
| PEL_TOL         | 正方向硬極限容差值       | 45405 | UINT32 |
| NEL_TOL_EN      | 負方向硬極限容差功能使能    | 45407 | UINT16 |
| NEL_TOL         | 負方向硬極限容差值       | 45408 | UINT32 |
| PIN_MODE        | 編碼器脈衝輸入類型       | 45421 | UINT16 |
| PIN_LOGIC       | 編碼器脈衝輸入邏輯反相     | 45422 | UINT16 |
| PIN_MAXFREQ     | 編碼器脈衝輸入的最高頻率限值  | 45423 | UINT16 |

|              |                                |       |        |
|--------------|--------------------------------|-------|--------|
| POUT_MODE    | 指令脈衝輸出類型                       | 45431 | UINT16 |
| POUT_REVERSE | 指令脈衝輸出邏輯反相                     | 45432 | UINT16 |
| ALM_FILTER   | 報警(ALM) 埠的濾波時間                 | 45441 | UINT16 |
| NEL_FILTER   | 負方向硬限位埠濾波時間                    | 45442 | UINT16 |
| PEL_FILTER   | 正方向硬限位埠濾波時間                    | 45443 | UINT16 |
| ORG_FILTER   | 原點信號埠濾波時間                      | 45444 | UINT16 |
| IN1_FILTER   | IN1 埠的濾波時間                     | 45445 | UINT16 |
| IN2_FILTER   | IN2 埠的濾波時間                     | 45446 | UINT16 |
| IN4_FILTER   | IN4 埠的濾波時間                     | 45447 | UINT16 |
| IN5_FILTER   | IN5 埠的濾波時間                     | 45448 | UINT16 |
| EXT_SRC      | 哪個軸的外部驅動輸入埠作為外部驅動信號源           | 45461 | UINT16 |
| EXT_EN       | 暫無作用，預留                        | 45462 | UINT16 |
| EXT_MODE     | 手輪模式外部驅動的脈衝輸入模式                | 45463 | UINT16 |
| EXT_PULSE    | 手輪模式外部驅動時，每個手輪脈衝輸入對應多少個指令脈衝輸出值 | 45464 | UINT32 |

● .....

● 軸 15 參數、配置的 ModBUS 地址表

| 變數     | 說明              | ModBUS 地址 | 資料類型 |
|--------|-----------------|-----------|------|
| VL     | 軸初速度            | 49501     | F32  |
| VH     | 軸運行速度           | 49503     | F32  |
| ACC    | 軸加速度            | 49505     | F32  |
| DEC    | 軸減速度            | 49507     | F32  |
| JK     | 速度曲線類型：S 型或 T 型 | 49509     | F32  |
| MAXVEL | 軸運行速度限值         | 49511     | F32  |
| MAXACC | 軸加速度限值          | 49513     | F32  |

|                 |                 |       |        |
|-----------------|-----------------|-------|--------|
| MAXDEC          | 軸減速度限值          | 49515 | F32    |
| MAXJK           | 暫無作用，預留         | 49517 | F32    |
| INSTOP_DEC      | INSTOP 功能中的減速度  | 49519 | F32    |
| UNIT_NUM        | 脈衝當量分子          | 49531 | UINT32 |
| UNIT_DENOM      | 脈衝當量分母          | 49533 | UINT32 |
| HOME_CROSS      | 回原點運行中的跨越距離     | 49551 | F32    |
| HOME_EX         | 暫無作用，預留         | 49553 | UINT16 |
| ORG_LOGIC       | ORG 信號的有效邏輯電平   | 49554 | UINT16 |
| ORG_MODE        | 回原點運動結束時的停止模式   | 49555 | UINT16 |
| EZ_LOGIC        | Z 相輸入信號的有效邏輯電平  | 49556 | UINT16 |
| HOME_RESET      | 回原點後清零位置值功能     | 49557 | UINT16 |
| HOME_OFFSETDIST | 回原點成後的偏移距離      | 49558 | F32    |
| HOME_OFFSETVEL  | 回原點成後偏移距離的移動速度  | 49560 | F32    |
| HOME_MODE       | 回原點模式           | 49562 | UINT16 |
| HOME_VL         | 回原點運動的初速度       | 49563 | F32    |
| HOME_VH         | 回原點運動的運行速度      | 49565 | F32    |
| HOME_ACC        | 回原點運動的加速度       | 49567 | F32    |
| HOME_DEC        | 回原點運動的減速度       | 49569 | F32    |
| HOME_JK         | 回原點運動的速度曲線類型    | 49571 | F32    |
| JOG_VLTIME      | JOG 運動低段速度運行的時間 | 49581 | UINT32 |
| JOG_VL          | JOG 運動低段速度      | 49583 | F32    |
| JOG_VH          | JOG 運動高段速度      | 49585 | F32    |
| JOG_ACC         | JOG 運動的加速度      | 49587 | F32    |
| JOG_DEC         | JOG 運動的減速度      | 49589 | F32    |
| EL_EN           | 使能硬體限位元功能       | 49601 | UINT16 |
| EL_LOGIC        | 硬體限位元信號的有效邏輯電平  | 49602 | UINT16 |
| EL_MODE         | 硬限位元觸發時的停止模式    | 49603 | UINT16 |

|              |                                        |       |        |
|--------------|----------------------------------------|-------|--------|
| PEL_TOL_EN   | 正方向硬極限容差功能使能                           | 49604 | UINT16 |
| PEL_TOL      | 正方向硬極限容差值                              | 49605 | UINT32 |
| NEL_TOL_EN   | 負方向硬極限容差功能使能                           | 49607 | UINT16 |
| NEL_TOL      | 負方向硬極限容差值                              | 49608 | UINT32 |
| PIN_MODE     | 編碼器脈衝輸入類型                              | 49621 | UINT16 |
| PIN_LOGIC    | 編碼器脈衝輸入邏輯反相                            | 49622 | UINT16 |
| PIN_MAXFREQ  | 編碼器脈衝輸入的最高頻率限<br>值                     | 49623 | UINT16 |
| POUT_MODE    | 指令脈衝輸出類型                               | 49631 | UINT16 |
| POUT_REVERSE | 指令脈衝輸出邏輯反相                             | 49632 | UINT16 |
| ALM_FILTER   | 報警(ALM) 埠的濾波時間                         | 49641 | UINT16 |
| NEL_FILTER   | 負方向硬限位埠濾波時間                            | 49642 | UINT16 |
| PEL_FILTER   | 正方向硬限位埠濾波時間                            | 49643 | UINT16 |
| ORG_FILTER   | 原點信號埠濾波時間                              | 49644 | UINT16 |
| IN1_FILTER   | IN1 埠的濾波時間                             | 49645 | UINT16 |
| IN2_FILTER   | IN2 埠的濾波時間                             | 49646 | UINT16 |
| IN4_FILTER   | IN4 埠的濾波時間                             | 49647 | UINT16 |
| IN5_FILTER   | IN5 埠的濾波時間                             | 49648 | UINT16 |
| EXT_SRC      | 哪個軸的外部驅動輸入埠作為<br>外部驅動信號源               | 49661 | UINT16 |
| EXT_EN       | 暫無作用，預留                                | 49662 | UINT16 |
| EXT_MODE     | 手輪模式外部驅動的脈衝輸入<br>模式                    | 49663 | UINT16 |
| EXT_PULSE    | 手輪模式外部驅動時，每個手輪<br>脈衝輸入對應多少個指令脈衝<br>輸出值 | 49664 | UINT32 |

### 3.2.5 運動指令對應的 Modbus 地址

#### 3.2.5.1 MOVE 運動指令

當上位機寫 1 到“指令對應的 Modbus 位址”時，對應的軸會執行單軸相對運動，運動的距離為該軸對應的“運動距離對應的 Modbus 地址”裡的值；上位機寫 0 到“指令對應的 Modbus 位址”時，控制器不做任何處理。

| 說明        | 指令對應的 Modbus 地址<br>(資料類型：Bit) | 運動距離對應的 Modbus 地址<br>(資料類型：F32) |
|-----------|-------------------------------|---------------------------------|
| 軸 0 相對運動  | 9001                          | 44501                           |
| 軸 1 相對運動  | 9002                          | 44503                           |
| .....     |                               |                                 |
| 軸 15 相對運動 | 9016                          | 44531                           |

### 3.2.5.2 JOG 運動指令

當上位機寫 1 到“指令對應的 Modbus 位址”時，對應的軸會執行 JOG 運動。當上位機寫 0 到“指令對應的 Modbus 位址”時，對應的軸會停止當前的 JOG 運動。

| 說明             | 指令對應的 Modbus 地址(資料類型：Bit) |
|----------------|---------------------------|
| 軸 0 正向 JOG 運動  | 9033                      |
| 軸 1 正向 JOG 運動  | 9034                      |
| .....          |                           |
| 軸 15 正向 JOG 運動 | 9048                      |
|                |                           |
| 軸 0 負向 JOG 運動  | 9065                      |
| 軸 1 負向 JOG 運動  | 9066                      |
| .....          |                           |
| 軸 15 負向 JOG 運動 | 9080                      |

### 3.2.5.3 回原點運動指令

當上位機寫 1 到“指令對應的 Modbus 位址”時，對應的軸會執行回原點運動。當上位機寫 0 到“指令對應的 Modbus 位址”時，控制器不做任何處理。

| 說明          | 指令對應的 Modbus 地址(資料類型：Bit) |
|-------------|---------------------------|
| 軸 0 正向回原點運動 | 9097                      |
| 軸 1 正向回原點運動 | 9098                      |

|              |      |
|--------------|------|
| .....        |      |
| 軸 15 正向回原點運動 | 9112 |
|              |      |
| 軸 0 負向回原點運動  | 9129 |
| 軸 1 負向回原點運動  | 9130 |
| .....        |      |
| 軸 15 負向回原點運動 | 9144 |

#### 3.2.5.4 停止運動指令

當上位機寫 1 到“指令對應的 Modbus 位址”時，控制器會控制對應的軸停止運動。當上位機寫 0 到“指令對應的 Modbus 位址”時，控制器不做任何處理。

| 說明        | 指令對應的 Modbus 地址(資料類型：Bit) |
|-----------|---------------------------|
| 停止軸 0 運動  | 9161                      |
| 停止軸 1 運動  | 9162                      |
| .....     |                           |
| 停止軸 15 運動 | 9176                      |