目錄 SCARA Cpp API 使用手冊

1. SCARA Cpp 設置說明

1.1. 打開 DROE_API_CPP 資料夾

打開範例資料夾會見到一個範例程式資料夾與 API 專用 Project 資料夾。

(1) DROE_API_Project
- 執行 API project 所需的程式

(2) DROE_CPP_API_Sample
- 範例程式

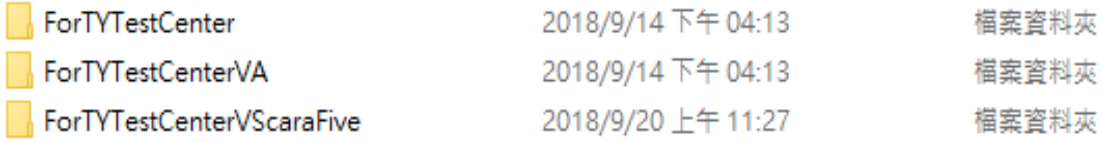
1.2. RABD Library

(1) 安裝 RABD Library
- 選擇與電腦系統類型相符的 COM 組件

(2) 在 Visual Studio 建立純 C++ 之專案
- 匯入 RABD.Lib 之 tlb 檔(#import <RABD.Lib.tlb> raw_interfaces_only)

(3) 初始化 COM 介面
- CoInitialize(NULL)
- IRobotPtr pRobot(__uuidof(Robot))　　\\pRobot is an indicator of a robot object

1.3. DROE_API_Project 資料夾



資料夾內有三個 Project，分別給四軸、五軸及六軸使用的 API 專案。

(1) ForTYTestCenter

- 四軸 SCARA 使用的 API 專案

(2) ForTYTestCenterVA

- 六軸 VA 使用的 API 專案

(3) ForTYTestCenterVScaraFive

- 五軸 SCARA 使用的 API 專案

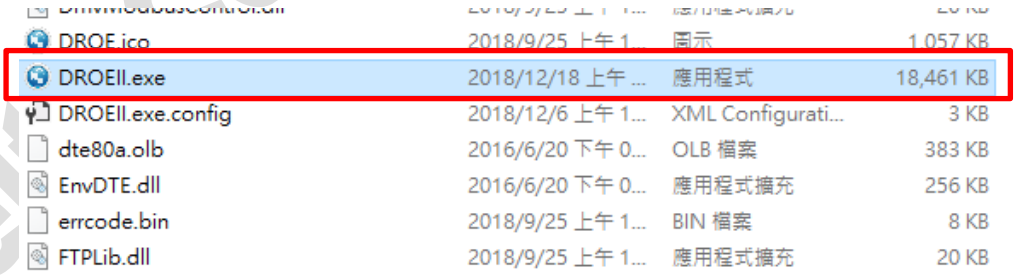
1.4. 上傳特定 API 專案至控制器

將相對應機型的 API 專案放入安裝的 DRASTUDIO 中的 DROEsolution 資料夾內。

(1) 確認 Drastudio 可以讀取此專案

(2) DROEsolution 資料夾之路徑: \\Delta Industrial Automation\DELTA_IA-Robot_DRAStudio-V1-00-07-xx_SW_TSE_20xxxxxx\DROEsolution

1.5. 打開 DRASTUDIO 執行檔



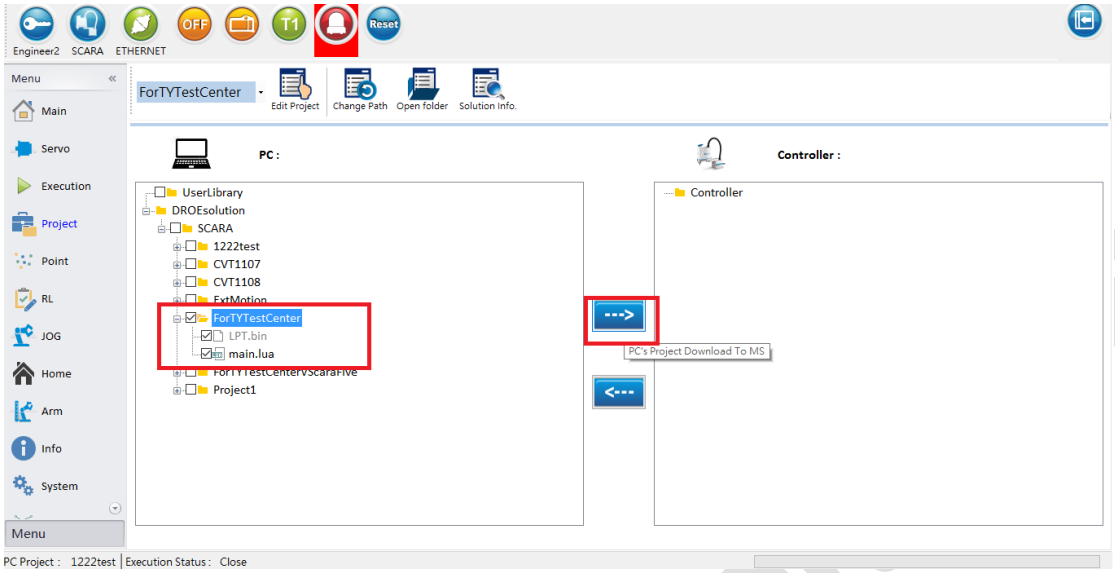
1.6. 登入操作介面

1.7. 連線

點選上方 Ethernet 按鈕來進行連線

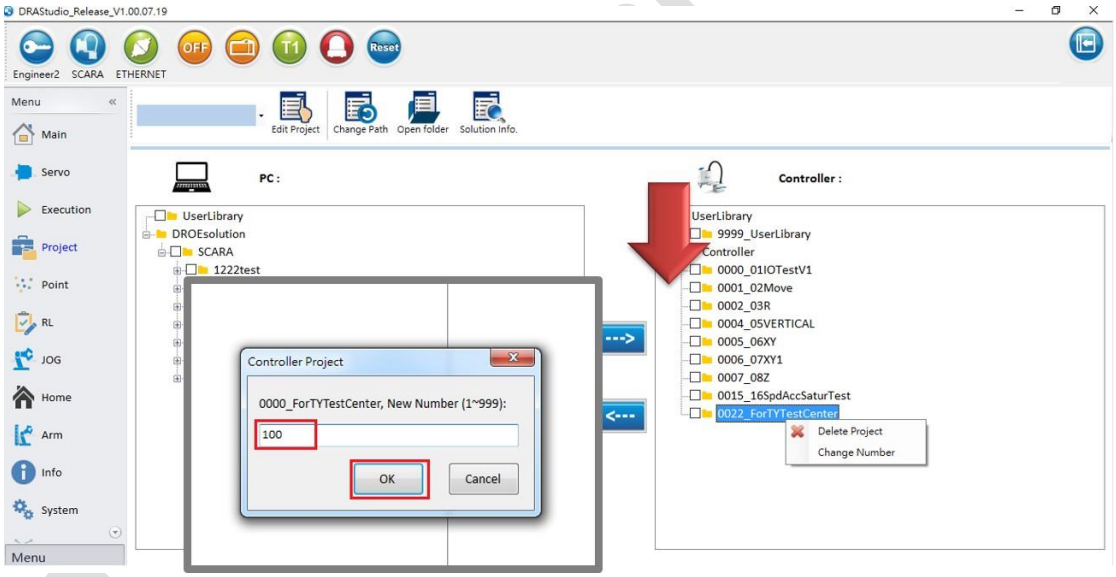
1.8. 將 API 專案放入控制器內

確認 1.4 上傳特定 API 專案至控制器 執行正確，DRATUDIO 可以讀取此專案。

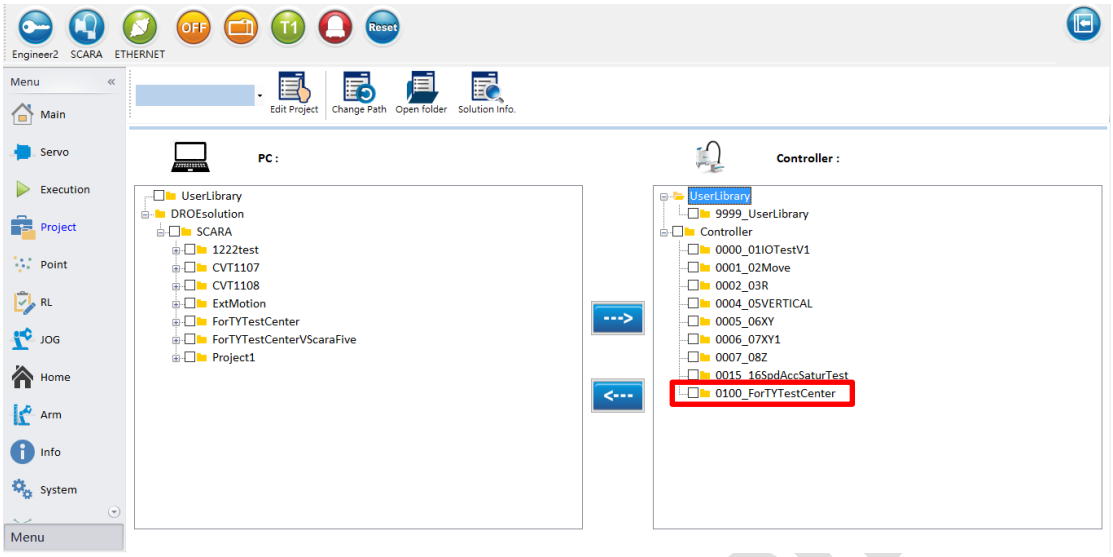


1.9. 專案 ID 修改為 100

(1) 點擊滑鼠右鍵，選擇【Fix Number】，輸入數值 100，點擊【OK】完成修改。



(2) 修改完成後，資料夾名稱將會變成【0100_ForTYTestCenter】。



2. API 使用說明

2.1. 連接機器人(Connect Robot)

功能：與機器人建立連結。

(1) 機器人建立連接的指令函式

pRobot->**ConnectRobot_2**(MSIP, ComputerIP, ComputerPort, &retval)

(2) 輸入參數

以下為 BSTR 型態轉換之過程:

- 參數 MSIP
  - 輸入 IP 地址

    std::wstring ip = L"192.168.1.1";
  - 將 IP 地址轉換為 BSTR 型態

    BSTR MSIP = SysAllocStringLen(ip.data(), ip.size());
- 參數 Computerip
  - 輸入 Computerip

    std::wstring Computerip = L"192.168.1.100";
  - 將 Computerip 轉換為 BSTR 型態

    BSTR ComputerIP = SysAllocStringLen(Computerip.data(), Computerip.size());
- 參數 ComputerPort

  long ComputerPort=10000;

- 回傳值 &retval
  - pRetVal 為返回值，VARIANT_BOOL 類型
  - VARIANT_BOOL 型態為真時，其值為 0xFFFF，為假時則為 0x0000

2.2. 斷開連線(DisConnectRobot)

功能：與手機械手臂斷開連線

(1) 與機器人段開連線之指令函式

pRobot->**DisConnectRobot**(VARIANT_BOOL *retval);

2.3. 保持連線狀態

功能：由於 MS 連線後四秒沒有進行讀取或寫入會將連線中斷，建立一個 Thread 定時取值來保持連線狀態。

程式碼如下：

```
HANDLE h = CreateThread( NULL, 0, ThreadFunction, 0, 0L, NULL );

bool threadRun = true;

DWORD WINAPI ThreadFunction(void* pContext )  {
while(threadRun){
                    eExecutorState state;
                    pRobot->ExecutorState(&state);
                     Sleep(500);
                }
                     return 0;
}
```

2.4. 獲取連線狀態(IsConnected)

功能：獲取連線狀態 VARIANT_BOOL 類型的 pRetVal 接受返回值。

- Get Connect Status

  VARIANT_BOOL Connect_Status;

  pRobot->**IsConnected**(&Connect_Status);

2.5. 伺服開關以及回 Home(ServoOff / ServoOn、GoHome)

■ 伺服開關及回 Home 的函式:

1. pRobot->**ServoOff**(); //關閉手臂所有伺服使能
2. pRobot->**ServoOn**(); //打開手臂所有伺服使能
3. pRobot->**GoHome**() //手臂全軸回伺服原點
4. pRobot->DynamicbrakeOff(false); //開啟動態煞車
5. pRobot->DynamicbrakeOff(true);
   //關閉動態煞車。(Servo on 後會將動態煞車重新設置)

2.6. 伺服狀態(ServoState)

功能：獲取伺服使能狀態 VARIANT_BOOL 類型的 pRetVal 接受返回值

■ 讀取伺服的函式:

VARIANT_BOOL **ServoState**;
pRobot->ServoState(**&ServoState)**

2.7. 單軸回 Home(GoHomeSingle)

功能：手臂單軸回伺服原點，輸入枚舉型 eRobotAxisName 類型的軸名稱 dir

■ The function to GoHomeSingle is:

Robot->**GoHomeSingle(eRobotAxisName_J1**);

The list of axis selections is as follows:

1. eRobotAxisName_J1
2. eRobotAxisName_J2
3. eRobotAxisName_J3
4. eRobotAxisName_J4
5. eRobotAxisName_X
6. eRobotAxisName_Y
7. eRobotAxisName_Z
8. eRobotAxisName_RZ

2.8. 程序(SelectProgram，RunProgram ， PauseProgram，StopProgram)

■ 運行、暫停、停止及選擇程序的函式:

1. short ProjectID = 5;
   pRobot-> SelectProgram(ProjectID);

   //從控制器中選擇要執行程序的編號 輸入 long 類型的編號 index

2. pRobot->**RunProgram**(); //運行專案

3.pRobot->**PauseProgram**();//暫停專案

4. pRobot->**StopProgram**(); //停止專案

2.9. 執行選擇程序(RunSelectProgram)

■ 執行選擇程序的函式:

```
short ProjectID = 5;
pRobot->RunSelectProgram(ProjectID);

//運行選擇的程序 輸入 short 類型的 index 程序編號
```

2.10. 獲取輸入/輸出狀態(GetInputState , GetOutputState)

功能：取得 User DI 以及 User DO 狀態。

■ Read User DI/DO status

VARIANT_BOOL IOStatus;

pRobot->**GetInputState(long** index, &IOStatus);

//获取某个输入点的状态 输入 long 类型输入点编号，VARIANT_BOOL 类

型的 pRetVal 接受返回值 。

pRobot->**GetOutputState(long** index, &IOStatus);

2.11. 獲取所有輸入狀態(GetAllInputState)

功能：獲取所有輸入點的狀態 long 類型的 pRetVal 接受返回值，

Ex: DI 1 On(1), DI3 On(4), return 1+4 = 5 二進制。

- Read all User DI status
  long UserInput;
  pRobot->**GetAllInputState**(&UserInput);

2.12. 獲取所有輸出狀態(GetAllOutputState)

功能：獲取所有輸出點的狀態 long 類型的 pRetVal 接受返回值。

- Read all User DO status
  long UserOutput;
  pRobot->**GetAllOutputState**(&UserOutput);

2.13. 獲取外部輸入狀態(GetExtInputState)

功能：獲取擴展輸入點的狀態 輸入 long 類型擴展站號 id，輸入 long 類型

輸入點編號 index，VARIANT_BOOL 類型的 pRetVal 接受返回值

- Read External DI status

```
VARIANT_BOOL External_Input_Status;
long ExtermalID = 1;
long External_intput = 1;
pRobot->GetExtInputState(ExtermalID,
External_intput,&External_Input_Status);
```

2.14. 獲取外部輸出狀態(GetExtOutputState)

功能：獲取擴展輸出點的狀態 輸入 long 類型擴展站號 id，輸入 long 類型

輸出點編號，indexVARIANT_BOOL 類型的 pRetVal 接受返回值。

- Read External DO status

```
VARIANT_BOOL External_Output_Status;
long ExtermalID = 1;
long External_intput = 1;
pRobot->GetExtOutputState
(ExtermalID,External_intput,&External_Output_Status);
```

2.15. 設置外部輸出狀態(SetExtOutputState)

功能：設置擴展外部輸出狀態 輸入 long 類型的站號 id，輸入 long 類型的

輸出點編號 index，輸入 VARIANT_BOOL 類型的 TOnFOff 狀態。

- Set External DO status

```
long exteral_id = 1;
long exteral_output_nuber = 1;
```

pRobot->**SetExtOutputState**(exteral_id, exteral_output_nuber,true);

2.16. 設置輸出狀態(SetOutputState)

功能：設置某個輸出點的狀態 輸入 long 類型輸出點編號，輸入

VARIANT_BOOL 類型的 TOnFOff 狀態

- Set User DO status

VARIANT_BOOL out = true;

pRobot->**SetOutputState(long** index, out);

2.17. 設置所有輸出狀態(SetAllOutputState)

功能:設置所有輸出狀態 輸入 short 類型 Doinfo 。 DOOutput = 7 (DO0 (on 1)+DO1(on 2)+DO2(on 4))

- Set all user DO status

```
short DOOutput = 7;
pRobot->SetAllOutputState(DOOutput);
```

2.18. 獲取機器人輸入/輸出狀態(VA)(GetRobotInputState、 GetRobotOutputState)

功能：獲取某個輸入點的狀態（用於六軸） 輸入 long 類型輸入點編號，

VARIANT_BOOL 類型的 pRetVal 接受返回值 。

- Read User DI/DO status

```
VARIANT_BOOL IOStatus;
pRobot->GetRobotInputState(long index, &IOStatus);
pRobot->GetRobotOutputState(long index, &IOStatus);
```

2.19. 設置機器人輸出狀態(VA)(SetRobotOutputState)

功能：設置某個輸出點的狀態（用於六軸） 輸入 long 類型輸出點編號，輸入 VARIANT_BOOL 類型的 TOnFOff 狀態 。

- Set User DO status
  VARIANT_BOOL out = true/false;

  pRobot->**SetRobotOutputState(long** index, out);

2.20. Modbus 單筆資料寫入(WriteSingleModbus)

功能：寫入單筆資料到 Modubs 地址(address 限制 0x1000~0x1FFF)。

- Write Modbus funciton
  std::wstring value = L"32767";
  BSTR modbus_value = SysAllocStringLen(value.data(), value.size());

  pRobot>**WriteSingleModbus**(short address, eRobotModbusWordTyp
  e Type, modbus_value) ;
  enum eRobotModbusWordType {
  eRobotModbusWordType_Word,
  eRobotModbusWordType_DoubleWord
  };

2.21. Modbus 多筆資料寫入(WriteMulitModbus)

功能：寫入多筆資料到 Modubs 地址(address 限制 0x1000~0x1FFF)。

- Modbus Multiple Write
  SAFEARRAY *data;
  SAFEARRAYBOUND rgsabound[1];

```
rgsabound[0].cElements = 3;//word 數量
rgsabound[0].lLbound = 0;
data = SafeArrayCreate(VT_I2,1,rgsabound);
short bVal[3] = {32767 ,32767 ,32767 };//要寫入的資料
for(long index = 0; index <(long)rgsabound[0].cElements; index++)
 SafeArrayPutElement(data,&index,&bVal[index]);
 pRobot->WriteMulitModbus((short) 0x1000, data, &retval);
 SafeArrayDestroy(data);   data = NULL;
```

2.22. Modbus 單筆資料讀取(ReadSingleModbus)

功能：讀取單筆 Modbus 地址 輸入 short 類型 address，輸入枚舉 eRobotModbusWordType 類型的 WordType，BSTR 類型的 pRetVal 接受返回值。

- Write Modbus funciton

```
BSTR modbus_value;
pRobot->ReadSingleModbus((short)(0x1000 + i),
eRobotModbusWordType_Word, &modbus_value);
enum eRobotModbusWordType   {
  eRobotModbusWordType_Word,
  eRobotModbusWordType_DoubleWord
};
```

2.23. Modbus 多筆資料讀取(ReadMulitModbus)

功能：讀取多筆 Modbus 地址 輸入 short 類型 address，輸入 short 類型的 count，數組 SAFEARRAY 類型的 pRetVal 接受返回值。

- Modbus Multiple Read

```
short bData[3];
SAFEARRAY *modbus_data;

 pRobot->ReadMulitModbus((short) 0x1100, 3, &modbus_data);

for(long index = 0;index < 3; index++)
 {
 SafeArrayGetElement(modbus_data, &index, &bData[index]);
  std::cout << bData[index] << std::endl;
  }
  SafeArrayDestroy(modbus_data);
   modbus_data = NULL;
```

2.24. 警報狀態(HasWarning、HasAlarm)

功能 : 獲取報警狀態 VARIANT_BOOL 類型的 pRetVal 接受返回值， 獲取警告狀態 VARIANT_BOOL 類型的 pRetVal 接受返回值。

- Get Alarm , Warning Status.

```
VARIANT_BOOL WarningStatus , AlarnStatus;
pRobot->HasWarning(&WarningStatus);
pRobot->HasAlarm(&AlarnStatus);
```

2.25. 警報復位(ResetAlarm)

功能：復位系統警報。

- Reset Alarm status

```
pRobot->ResetAlarm();
```

2.26. 警報代碼(MonitorAlarmCode)

功能：獲取報警代碼 long 類型的 pRetVal 接受返回值報警代碼。

- Reset Alarm status

```
long alarmcode;
pRobot->MonitorAlarmCode(&alarmcode);
```

2.27. 機器人類型(RobotType)

功能：獲取手臂類型 枚舉 RobotTypes 類型的 pRetVal 接受返回值。

- Get Robot Type function

```
RobotTypes RobotType;
pRobot->RobotType(&RobotType);
```

2.28. 選擇座標系(FrameSelect)

功能：選擇工具坐標系 ID 和使用者坐標系 ID 輸入 long 類型工具坐標系編號 toolFrameID，輸入 long 類型使用者坐標系編號 userFrameID

- Change UserFrame and ToolFrame

```
pRobot->FrameSelect(int toolFrameID, int userFrameID);
```

2.29. 獲取工具座標系信息(GetToolFrame)

功能：獲取給定 ID 下的工具坐標系信息 輸入 long 類型 toolFrameID 的工具坐標系編號，SAFEARRAY 類型的 pRetVal 接受返回工具坐標系信息。

- Get ToolFrame

```
SAFEARRAY *ToolFrame;
long ToolFrame_id= 2;
pRobot->GetToolFrame(ToolFrame_id, &ToolFrame);
```

2.30. 獲取使用者座標系信息(GetUserFrame)

功能：獲取給定 ID 下的使用者坐標系信息 輸入 long 類型 userFrameID 的使用者坐標系編號，SAFEARRAY 類型的 pRetVal 接受返回使用者坐標系信息。

- Get UserFrame

```
SAFEARRAY *UserFrame;
long UserFrame_id= 2;
pRobot->GetUserFrame(UserFrame_id, &UserFrame);
```

2.31. 獲取當前點為(GetPos)

功能：獲取當前點位信息 sPoint 類型的 p 接受返回值點位信息。

- Get Robot Position

```
sPoint p;

pRobot->GetPos(&p);

std::cout << p.X << std::endl
```

2.32. 移動點位(sPoint)

功能：設置移動點位資訊。

- Set Robot position point

```
long JrcValue,J4JRC=0,J6JRC=0,JRCMode=4;
sPoint GoalPoint;
GoalPoint.X = 450000.0;
GoalPoint.Y = 0.0;
GoalPoint.Z = 0.0;
GoalPoint.RX = 0;
GoalPoint.RY = 50;
GoalPoint.RZ = 0.0;
GoalPoint.Hand = 0;
GoalPoint.Shoulder = 0;
GoalPoint.Flip = 0;
GoalPoint.UserFrame = 1;
GoalPoint.ToolFrame = 1;
pRobot->SetJointIndexValue(J4JRC, J6JRC, JRCMode, &JrcValue);
GoalPoint.JointIndex = JrcValue;
```

2.33. 機器人移動狀態(RobotMovingStatus)

功能:獲取手臂的移動狀態 VARIANT_BOOL 類型的pRetVal 接受返回值。

- Get Robot Moving Status

  pRobot->**RobotMovingStatus**(&retval);

2.34. 控制器溫度(TemperatureStatus)

功能：獲取控制器溫度狀態 枚舉類型 eRobotTemperatureStatus 的 pRetVal 接受返回值。

- Get Robot temprature Status

  eRobotTemperatureStatus TemperatureStatus;
  pRobot->**TemperatureStatus**(&TemperatureStatus);
  switch(Temperaturestate){
  case 0:
  std::cout << “Normal " << std::endl;
  break;
  case 1:
  std::cout << “Modify " << std::endl;
  break;
  case 2:
  std::cout << “OverHeat " << std::endl;
  break;
  }

2.35. 功能性暫停狀態(FunctionPauseStatus)

功能：獲取功能性暫停(安全門)的狀態 VARIANT_BOOL 類型的 pRetVal 接受返回值。

- Get Functional Pause Status

  pRobot->**FunctionalPauseStatus**(&retval);

2.36. TP 狀態(TPStatus)

功能：獲取 TP 是否使用狀態 VARIANT_BOOL 類型的 pRetVal 接受返回值。

- Get Teach panel Status

```
VARIANT_BOOL Tpstatus;
pRobot->TPStatus(&Tpstatus);
```

2.37. 獲取操作模式(ExecutorState)

功能：獲取操作模式 枚舉 eRobotOperationModeStatus 類型的 pRetVal 接受返回值操作模式。

- The function of Read OperationMode status is:

```
eRobotOperationModeStatus operation_state;
pRobot->OperatingStatus(&operation_state);
switch(operation_state){
case 0:
      std::cout << “T1" << std::endl;
      break;
case 1:
      std::cout << “T2" << std::endl;
      break;
case 2:
       std::cout << “Auto" << std::endl;
      break;
                        }
```

2.38. 角度轉換 PUU(DegreeToPUU)

功能：將角度轉換成 PUU。

- Convert angle to PUU

```
SAFEARRAY *puu;
long count;
pRobot->DegreeToPUU(&puu,& count);
```

2.39. 工作空間狀態轉換(WorkSpace_Switch)

功能：設置工作空間狀態 輸入 VARIANT_BOOL 的 WS_Switch 開關工作空間。

- The function of swtich Workspace function is:

```
pRobot->WorkSpace_Switch(true);
```

2.40. 獲取圓柱形工作空間訊息(Get_Cylinder_WS)

功能：獲取圓柱形工作空間信息 輸入 long 類型 WS_ID，SAFEARRAY 類型的 pRetVal 接受返回值。

- The function of Get Cylinder Workspace is:

```
SAFEARRAY *gain_data;
long CylinderWorkSpaceId = 2;
pRobot->Get_Cylinder_WS(CylinderWorkSpaceId, &gain_data);
```

2.41. 獲取長方形工作空間訊息(Get_Rectangle_WS)

功能：獲取長方形工作空間信息 輸入 long 類型 WS_ID， SAFEARRAY 類型的 pRetVal 接受返回值。

- The function of Get Cylinder Workspace is:

```
SAFEARRAY *gain_data;
long RetangleWorkSpaceId =1;
pRobot-> Get_Rectangle_WS(RetangleWorkSpaceId, &gain_data);
```

2.42. 刪除控制器專案 (DeleteControllerProjec)

功能：刪除控制器專案 輸入 BSTR 類型的 name，VARIANT_BOOL 類型的 pRetVal 接受返回值。

- Start ID 100 project，Stop ID 100 project

```
BSTR PorjectName;
pRobot->DeleteControllerProjec(PorjectName, &retval);
```

2.43. 設置速度、軸運動距離、大地坐標下移動距離(SetSpeed，SetJointDistance，SetCartesianDistance)

功能：設置運動速度 ，設置軸運動的距離 輸入 long 類型的距離 dis ，設置大地坐標系下移動的距離 輸入 short 類型的 dis 距離值。

- Set motion speed，joint distance ,Cartesian distance

```
long spd = 50;   (1~100%)
pRobot->SetSpeed(spd);
long dis = 1000;
pRobot->SetJointDistance(dis);
long dis = 1000;
pRobot->SetCartesianDistance(dis);
```

2.44. 吋動(Jog)

功能：寸動手臂（不開 100 號專案） 輸入枚舉型 eRobotDirection 類型的 dir 寸動方向 , MovStopEx，寸動時停止運動（不會開啟 100 程序）。

- Jog function (Continued) ， Step function (Step)

```
pRobot->Jog(eRobotDirection_XP);
pRobot->MovStop();
pRobot->Step(eRobotDirection_XP);
pRobot->MovStop();
```

2.45. 外部伺服(ExteranlServo)

功能：控制外部軸伺服使能 pRobot->ExternalServoOn(ID,&retval); pRobot->ExternalServoOff(ID, &retval);。

- Servo

```
pRobot->ExternalServoOn(ID,&retval);

pRobot->ExternalServoOff(ID, &retval);
```

2.46. 外部吋動(ExteranlJog)

功能：
pRobot->ExternalJogForward(ID); pRobot->ExternalJogReverse(ID);控制外部軸 吋動功能 pRobot->ExternalPUU(ID,&ExteralPuu);
讀取外部軸 PUU 數值。

- Jog

```
pRobot->ExternalJogForward(ID);
pRobot->MovStop();
pRobot->ExternalJogReverse(ID);
pRobot->MovStop();
long ExteralPuu;
long ID = 1;
pRobot->ExternalPUU(ID,&ExteralPuu);
```

2.47. 前往點位點對點運動(GotoMovP)

功能：點對點的方式運動給定點位信息 輸入 sPoint 類型的 p 點位，若中途要停止請下 MovStop();。

- Start Goto MovP function

```
sPoint GoalPoint;
GoalPoint.X = 450000.0;
GoalPoint.Y = 0.0;
GoalPoint.Z = 0.0;
GoalPoint.RZ = 0.0;
GoalPoint.Hand = 0;
pRobot->GotoMovP(GoalPoint);
pRobot->MovStop(); // If you want to stop motion，you can use this function。
```

2.48. 前往點位直線運動(GotoMovL)

功能：直線插補的方式運動給定點位信息 輸入 sPoint 類型的 p 點位 ，若中途要停止請下 MovStop();。

- Start Goto MovL function

```
sPoint GoalPoint;
GoalPoint.X = 450000.0;
GoalPoint.Y = 0.0;
GoalPoint.Z = 0.0;
GoalPoint.RZ = 0.0;
GoalPoint.Hand = 0;
pRobot->GotoMovL(GoalPoint);
pRobot->MovStop(); // If you want to stop motion，you can use this function。
```

2.49. 示教全域點位(TechGlobalPoint)

功能：示教全局點位信息 輸入 long 類型的全局點位編號 index。

- TeachGlobalPoint

```
long GlobalPointIndex = 5,JrcMode=4;
pRobot->TechGlobalPoint(GlobalPointIndex, JrcMode);
```

2.50. 開啟控制器專案、結束控制器專案(StartCmd,EndCmd)

功能：開啟控制器專案、結束控制器專案。

- Start ID 100 project，Stop ID 100 project

```
pRobot->StartCmd();
pRobot->EndCmd();
```

2.51. 動作確認(Ready signal)

功能：獲得動作是否完成。

- Start ID 100 project，you can get IsReady Signal

```
VARIANT_BOOL retval;
pRobot->IsReady(&retval);
```

2.52. 錯誤碼表(Error Code Table)

功能 : 錯誤碼表如上，在使用 100 號專案相關移動指令時會有錯誤碼回傳，0 代表未連線，1 代表完成 ， 2 代表 未收到資料，3 代表逾時。

- If you use MovP，MovL, MArchP, MArchL,SetSpeeedEx，SetAccelEx, SetDecelEx, SetAccurEx，you can get ErrorCode.

ErrorCode	Message
0	No connect
1	Finish
2	No data
3	TimeOut
Exception	

2.53. 錯誤碼(ErrorCode)

功能：錯誤碼轉換格式。

- Some functions will return Error Code

```
BSTR ErrorCode;
 pRobot->SetSpeedEx(int spd, &ErrorCode);
_bstr_t test = _bstr_t(ErrorCode);
int length = test.length();
char* char_array = new char[length];
Strcpy_s(char_array, length + 1, test);
printf(char_array);
```

2.54. 沿給定方向移動(StepEx)

功能：沿給定方向運動給定距離（會開啟 100 程序） 輸入枚舉 eRobotDirection 的 dir 運動方向輸入 double 類型的 dis 運動距離。

- Jog Function (step)

```
pRobot->StepEx(eRobotDirection_XP, 20, &ErrorCode)
// X postive ,  20 um
```

2.55. 吋動手臂(JogEx)

功能：寸動手臂（會開啟 100 程序） 輸入枚舉型 eRobotDirection 類型的 dir 寸動方向 。MovStopEx，寸動時停止運動（會開啟 100 程序）。

- Jog Function (Continued)

```
pRobot->SetSpeedEx(int speed);
pRobot->JogEx(eRobotDirection dir);
pRobot-> MovStopEx();
```

2.56. 吋動方向使用列舉(enum eRobotDirection)

功能：寸動方向使用之列舉如下。

- eRobotDirection

eRobotDirection_J1P,eRobotDirection_J1N,
eRobotDirection_J2P,eRobotDirection_J2N,
eRobotDirection_J3P,eRobotDirection_J3N,
eRobotDirection_J4P,eRobotDirection_J4N,
eRobotDirection_XP, eRobotDirection_XN,
eRobotDirection_YP, eRobotDirection_YN,
eRobotDirection_ZP, eRobotDirection_ZN,
eRobotDirection_RZP, eRobotDirection_RZN

2.57. 設置運動速度(SetSpeedEx)

功能：設置運動速度（開啟 100 程序）。

- Set the speed function, spd is the speed (1~100%)

```
pRobot->SetSpeedEx(int spd , &ErrorCode);
do
{
pRobot->IsReady(&retval);
} while (retval != -1);
```

2.58. 設置運動減速度(SetDecelEx)

功能：設置運動減速度（開啟 100 程序）。

- Set the decel function, del is the Decel (1~100%)

```
pRobot->SetDecelEx(int del, &ErrorCode);
do
{
pRobot->IsReady(&retval);
} while (retval != -1);
```

2.59. 設置運動加速度(SetAccelEx)

功能：設置運動加速度（開啟 100 程序）。

- Set the accel function, acc is the speed (1~100%)

```
pRobot->SetAccelEx(int acc, &ErrorCode);
do
{
pRobot->IsReady(&retval);
} while (retval != -1);
```

2.60. 讀取運動速度(GetSpeedEx)

功能：讀取運動速度（開啟 100 程序）。

- Get the speed function, spd is the speed (1~100%)

```
long spd;
pRobot->GetSpeedEx(&spd);
```

2.61. 獲取加速度值(GetAccelEx)

功能:獲取加速度值(會開啟 100 程序) long 類型的 pRetVal 接受返回值。

- Get the Accel function, acc is the Accel (1~100%)

```
long acc;
pRobot->GetAccelEx(&acc);
```

2.62. 獲取減速度值(GetDecelEx)

功能:獲取減速度值(會開啟 100 程序) long 類型的 pRetVal 接受返回值。

- Get the Decel function, del is the Decel (1~100%)

```
long del;
pRobot->GetDecelEx(&del);
```

2.63. 點對點運動(MovP)

功能：點到點方式運動到給定信息點位 輸入 sPoint 類型的 p 點位信息。

- Move point to point Function

```
sPoint MovePoint;
MovePoint.X = 300000.0;
MovePoint.Y = 50000.0;
MovePoint.Z = 0.0;
MovePoint.RZ = 0.0;
MovePoint.Hand = 1;
pRobot->MovP(MovePoint, &ErrorCode);
do
{
pRobot->IsReady(&retval);
} while (retval != -1);
```

2.64. 直線差補運動(MovL)

功能：直線插補方式運動到給定信息點位 輸入 sPoint 類型的 p 點位信息。

- Linear interpolation Move Function

```
sPoint MovePoint;
MovePoint.X = 300000.0;
MovePoint.Y = 50000.0;
MovePoint.Z = 0.0;
```

```
MovePoint.RZ = 0.0;
MovePoint.Hand = 1;
pRobot->MovL(MovePoint, &ErrorCode);
do
{
pRobot->IsReady(&retval);
} while (retval != -1);
```

2.65. 直線差補拱門運動(MArchL)

功能：直線插補的拱門方式運動給定點位信息 輸入 sPoint 類型的 p 點位，輸入 long 類型的 h1 的絕對高度，輸入 long 類型的 h2 的最高安全高度，輸入 long 類型的 h3 的最低安全高度。

- Linear interpolation arch move function

```
sPoint MovePoint;
MovePoint.X = 300000.0;
MovePoint.Y = 50000.0;
MovePoint.Z = 0.0;
MovePoint.RZ = 0.0;
MovePoint.Hand = 1;
long h1 = -100, h2 = 0, h3 = 0;
pRobot->MArchL(GoalPoint, h1,h2, h3,&ErrorCode);
do
{
pRobot->IsReady(&retval);
} while (retval != -1);
```

2.66. 點對點拱門運動(MArchP)

功能：點對點的拱門方式運動給定點位信息輸入 sPoint 類型的 p 點位，輸入 long 類型的 h1 的絕對高度，輸入 long 類型的 h2 的最高安全高度，輸入 long 類型的 h3 的最低安全高度。

- Move point to point and arch move function

```
sPoint MovePoint;
MovePoint.X = 300000.0;
MovePoint.Y = 50000.0;
MovePoint.Z = 0.0;
MovePoint.RZ = 0.0;
MovePoint.Hand = 1;
```

```
long h1 = -100, h2 = 0, h3 = 0;
pRobot->MArchP(GoalPoint, h1,h2, h3,&ErrorCode);
do
{
pRobot->IsReady(&retval);
} while (retval != -1);
```

2.67. 圓弧差補運動(MCircle)

功能 : 指定目标点位及经过点位方式进行圆弧插捕的圆形运动，三点成圆，给定点位信息输入 sPoint 类型的 p 点位，Point1 為經過點，Point2 為目標點。

- Specify the target point and the circular motion of the arc insertion through the point mode, three points into a circle

```
sPoint Point1, Point2;
Point1.X = 213242;
Point1.Y = 150000;
Point1.Z = -50000;
Point1.RZ = 90.0;
Point1.Hand = 0;
Point2.X = 258573;
Point2.Y = 150000;
Point2.Z = -50000;
Point2.RZ = 90.0;
Point2.Hand = 0;
pRobot->MCircle(Poitnt1, Point2, &ErrorCode);
do
{
    pRobot->IsReady(&retval);
} while (retval != -1);
```

2.68. 手臂單軸運動(MovJ)

功能：手臂單軸運動 輸入枚舉 eRobotAxisName 的 axis 軸名稱，輸入 double 類型的角度。

- Move single joint to assign degree

```
double degree = 50;
pRobot->MovJ(eRobotAxisName_J1, degree, &ErrorCode);
do
```

```
{
    pRobot->IsReady(&retval);
} while (retval != -1);
```

2.69. 設置手臂到位精度(SetAccurEx)

功能：設置手臂的到位精度（會開啟 100 程序） 輸入枚舉型 eRobotAccur 的 accur 到位精度。

- Set accuracy function

```
pRobot->SetAccurEx(eRobotAccur_ROUGH, &ErrorCode);
do
{
    pRobot->IsReady(&retval);
}
while (retval == -1 ? false : true);
enum eRobotAccur  {
    MAXROUGH,
    ROUGH,
    STANDARD,
    MEDIUM,
    HIGH
                    };
```

2.70. 獲取軸到位精度(GetJointAccurEx)

功能：獲取所有軸的到位精度（會開啟 100 程序） long 類型的 J1 接受返回值 J1 軸到位精度，long 類型的 J2 接受返回值 J2 軸到位精度，long 類型的 J3 接受返回值 J3 軸到位精度，long 類型的 J4 接受返回值 J4 軸到位精度。

- Get Joint accuracy function

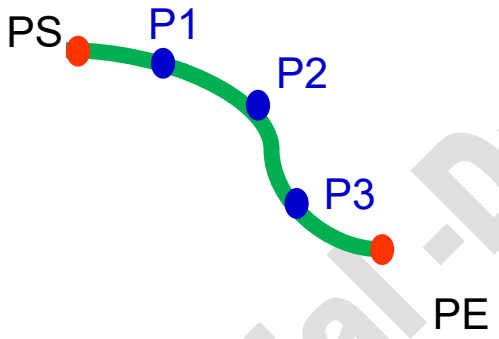
```
long J1_accuarcy, J2_accuarcy, J3_accuarcy, J4_accuarcy;
pRobot->GetJointAccurEx(&J1_accuarcy,
&J2_accuarcy,&J3_accuarcy, &J4_accuarcy);
```

2.71. 連續移動(Continuously MovP (by PASS))

- **StartContinuousMovP**(PS)：起始點(PS)

- 中繼點:：PathP(P1), PathP(P2), PathP(P3)
- **EndContinuousMovP**(PE)：最後一個點，啟動動作
  - do
    {
    pRobot->IsReady(&retval);
    } while (retval == -1 ? false : true);

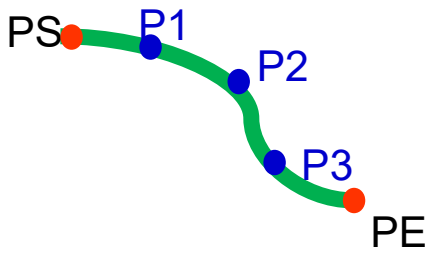
功能：沿著數個點位形成之路徑連續移動，此命令屬於合併命令(combine command)。



2.72. 連續直線移動(Continuously MovL (by PASS))

- **StartContinuousMovL**(PS)：起始點(PS)。
- 中繼點：PathL (P1),PathL (P2),PathL (P3)。
- **EndContinuousMovL** (PE)：最後一個點，啟動動作。
  - do
    {
    pRobot->IsReady(&retval);
    } while (retval == -1 ? false : true);

功能：沿著數個點位形成之路徑連續直線移動，此命令屬於合併命令(combine command)。



2.73. 初始化飛拍功能(InitialOnFly)

- **InitialOnFly**(&ErrorCode)

功能：初始化飛拍功能。

2.74. 設置飛拍功能(SetOnFly)

- **SetOnFly**(FlyIndex,GoalPoint, Radius, Time, DoIndex, Do_time,&ErrorCode)
  - Function 參數
    - FlyIndex = 1, long 類型
    - Radius = 50, double 類型
    - Time = 1, double 類型
    - DoIndex = 1, long 類型
    - Do_time = 1, double 類型

功能：設置飛拍功能，參數輸入 long 類型的飛拍組號 index，輸入 sPoint 類型的拍照點位 p，輸入 double 類型的距離點位半徑 Radius，輸入 double 類型的拍照後記錄點位坐標延時 Time，輸入 long 類型的觸發拍照 DO 編號 DOIndex，輸入 double 類型的觸發 DO 保持時間 DO_Time。

2.75. 獲取飛拍手臂位置(GetOnFlyRobotPos)

- **GetOnFlyRobotPos**(2, &check, &GoalPoint)

功能:獲取飛拍手臂位置 ，輸入 long 類型 index 飛拍組號，VARIANT_BOOL 類型的 check 接受返回值，Point 類型的 p 接受返回值點位信息。

2.76. 開啟 log 文檔(LogOpenFunction)

- **LogOpenFuntion**(true)

功能:log 文檔生產開關，輸入 VARIANT_BOOL 類型的 ON/OFF 開關 log。

- RLInitailDoneFlagAPI

功能：API 專案初始化是否完成。

2.77. 其他涵式(other)

- GetCVEncoderData(ID, &Value)

功能：讀取 MS 接上的編碼器數值(P5-37)。

- APIRLProjectVersion (&VerSion)

功能：讀取目前 API 與 API 專案版本號。