|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Lớp** | **Tính năng** | | | **Tốt** | **Chưa tốt** | **Mô tả** |
| Ngõ vào | DocNgoVao | | Theo bit |  |  | -Đọc từng bít của ngõ vào  -Method: DocNgoVao(string Chanel) ). Dữ liệu trả về kiểu byte có giá trị 0 hoặc 1  -VD: Byte = MyPLCPi.NgoVao.DocNgoVao("I0.0") |
| Theo byte |  |  | -Đọc 1 lần 8bit của ngõ vào, rồi trả về 1byte của ngõ vào đó.  -Method: DocNgoVao(string Chanel) ). Dữ liệu trả về kiểu byte  -VD: Byte = MyPLCPi.NgoVao.DocNgoVao("I0") |
| Ngõ ra | XuatNgoRa | | Theo bit |  |  | -Xuất 1bit của 1 ngõ ra bất kỳ, bật hoặc tắt  -Method: XuatNgoRa(string Chanel, byte Value)  -VD: myPLCPi.NgoRa.XuatNgoRa("Q0.0", 1) – bật  myPLCPi.NgoRa.XuatNgoRa("Q0.0", 0) – tắt |
| Theo byte |  |  | -Xuất 1 lần 8bit của 1 ngõ ra bất kỳ  -Method: XuatNgoRa(string Chanel, byte Value)  -VD: myPLCPi.NgoRa.XuatNgoRa("Q0", 150); |
| DocNgoRa | | Theo bit |  |  | -Đọc về trạng thái hiện tại của ngõ ra theo từng bit  -Method: DocNgoRa(string Chanel). Dữ liệu trả về kiểu byte có giá trị 0 hoặc 1  -VD: Byte = myPLCPi.NgoRa.DocNgoRa("Q0.0") |
| Theo byte |  |  | -Đọc về trạng thái hiện tại của ngõ ra theo 1 byte  -Method: DocNgoRa(string Chanel). Dữ liệu trả về kiểu byte  -VD: Byte = myPLCPi.NgoRa.DocNgoRa("Q0") |
| **Lớp** | **Tính năng** | | | **Tốt** | **Chưa tốt** | **Mô tả** |
| AI | DocAI | | 0-10VDC |  |  | -có 5 kênh AI 0-10VDC. Từ Chanel 0-4  -Method: DocAI(Int16 Chanel, double X0, double Y0, double X1, double Y1). Dữ liệu trả về kiểu double  -VD: Value = myPLCPi.AI.DocAI(0, 0,0,1024,10)) |
| 4-20mA |  |  | -có 8 kênh AI 4-20mA. Từ Chanel 5-12  -Method: DocAI(Int16 Chanel, double X0, double Y0, double X1, double Y1). Dữ liệu trả về kiểu double  -VD:test với cảm biến PT100  Value = myPLCPi.AI.DocAI(5, 204,0,1024,400)) |
| DS18B20 | DocNhietDo | | |  |  | -Đọc nhiệt độ tuef cảm biến DS18B20. Trả về giá trị nhiệt độ kiểu double.  -Method: DocNhietDo(string Id\_DS18B20). Dữ liệu trả về kiểu double  -VD: nhietdo = myPLCPi.DS18B20.DocNhietDo("28-00042b536dff");  -Nếu giá trị trả về = 1000 là lỗi kết nối với cảm biến |
| DHT21 | DocNhietDo | | |  |  | -Đọc nhiệt độ  -Method: DocNhietDo (). Trả về giá trị nhiệt độ dạng chuỗi. Nếu chuỗi trả về là “1000” thì bị mất kết nối đến DHT21.  -VD: Mang = myPLCPi.DHT21.DocNhietDoDoAm() |
| DocDoAm | | |  |  | -Đọc độ ẩm  -Method: DocDoAm(). Trả về giá trị độ ẩm dạng chuỗi. Nếu chuỗi trả về là “1000” thì bị mất kết nối đến DHT21.  -VD: Mang = myPLCPi.DHT21.DocDoAm() |
| DocNhietDoDoAm | | |  |  | -Đọc nhiệt độ độ ẩm.  -Method: DocNhietDoDoAm(). Trả về mảng string[]. String[0]: nhiệt độ; string[1]: độ ẩm. Nếu chuỗi trả về là “1000” thì bị mất kết nối đến DHT21.  -VD: Mang = myPLCPi.DHT21.DocNhietDoDoAm() |
| **Lớp** | **Tính năng** | | | **Tốt** | **Chưa tốt** | **Mô tả** |
| Hiển thị | HienThi | | Kênh 1 |  |  | - hiển thị giá trị cần hiển thị lên kênh 1 của module hiển thị  -Method: HienThi(string SoCanHienThi, Int16 KenhSo)  -VD: myPLCPi.HienThiLed7.HienThi(Temp,1) |
| Kênh 2 |  |  | - hiển thị giá trị cần hiển thị lên kênh 2 của module hiển thị  -Method: HienThi(string SoCanHienThi, Int16 KenhSo)  -VD: myPLCPi.HienThiLed7.HienThi(Temp,2) |
| Tắt kênh 1 |  |  | -Tắt kênh 1  -Method: HienThi(string SoCanHienThi, Int16 KenhSo)  -VD: myPLCPi.HienThiLed7.HienThi(“Tat”,1) |
| Tắt kênh 2 |  |  | -Tắt kênh 2  -Method: HienThi(string SoCanHienThi, Int16 KenhSo)  -VD: myPLCPi.HienThiLed7.HienThi(“Tat”,2) |
| Thời gian | DocThoiGian | | |  |  | -Lấy ngày giờ từ PLCPi  -Method: DocThoiGian().Trả về mảng 6 phần tử kiểu Int16[]={ngày, tháng, năm, giờ, phút, giây}  -VD: mang = myPLCPi.ThoiGian.DocThoiGian() |
| CaiDat | | |  |  | -Dùng để cài đặt thời gian cho PLCPi. Đối số DateTime có dạng ("dd-MM-yyyy HH:mm:ss")  -Method: CaiDat(string DateTime)  -VD: myPLCPi.ThoiGian.CaiDat("25-05-2015 23:00:00") |
| CaiDatRTC | | |  |  | -Dùng để cài đặt thời gian từ PLCPi xuống module thời gian thực  -Method: CaiDatRTC()  -VD: myPLCPi.ThoiGian.CaiDatRTC() |
| Snap7 | Server | | Kết nối |  |  | -Dùng để tạo PLCPi trở thành 1 Snap7Server, gồm các vùng nhớ: ngõ vào; ngõ ra; DB  -Method: KetNoi(). Khởi chạy Snap7Server. Trả về Chuỗi “GOOD”: thành công; khác “GOOD”: không thành công  -VD: string error = myPLCPi.Snap7.Server.KetNoi(); |
| Cập nhật giá trị của các vùng nhớ |  |  | -thì ta chỉ cần thay đổi các vùng nhớ ngõ vào, ngõ ra , DB |
| Client | Kết nối đến thiết bị Snap7Server | |  |  | -Đầu tiên ta phải chạy Snap7 Client trên PLCPi kết nối đến 1 thiết bị Snap7Server  myPLCPi.Snap7.Client.KetNoi(ip)  ip: là địa chỉ IP của thiết bị Snap7Server |
| DocDB | |  |  | -Đọc vùng nhớ DB  -Method: DocDB(int DB\_Num, int BatDau, int SoByte)  -Trả về mảng kiểu byte có số phần tử bằng SoByte |
| DocNgoVao | |  |  | -Đọc vùng nhớ ngõ vào  -Method: DocNgoVao(int BatDau, int SoByte)  -Trả về mảng kiểu byte có số phần tử bằng SoByte |
| DocNgoRa | |  |  | -Đọc vùng nhớ ngõ ra  -Method: DocNgoRa(int BatDau, int SoByte)  -Trả về mảng kiểu byte có số phần tử bằng SoByte |
| DocMB | |  |  | -Đọc vùng nhớ MB  -Method: DocMB(int BatDau, int SoByte)  -Trả về mảng kiểu byte có số phần tử bằng SoByte |
| GhiDB | |  |  | -Ghi dữ liệu xuống vùng nhớ DB của Snap7Server  -Method: GhiDB(int DB\_Num, int BatDau, int SoByte, byte[] Value) |
| GhiNgoVao | |  |  | -Ghi dữ liệu xuống vùng nhớ ngõ vào của Snap7Server  -Method: GhiNgoVao(int BatDau, int SoByte, byte[] Value) |
| GhiNgoRa | |  |  | -Ghi dữ liệu xuống vùng nhớ ngõ ra của Snap7Server  -Method: GhiNgoRa(int BatDau, int SoByte, byte[] Value) |
| GhiMB | |  |  | -Ghi dữ liệu xuống vùng nhớ MB của Snap7Server  -Method: GhiMB(int BatDau, int SoByte, byte[] Value) |