

PLC500 NSERIES
Programmable Logic Controllers



راهنمای کاربری
Application Manual

ویرایش اول
آذرماه ۱۳۸۸



فهرست مطالب:

۴	معرفی
۷	آشنایی با استاندارد IEC1131-3
۷	اجزای تشکیل دهنده سامانه در استاندارد IEC1131-3
۷	پیکربندی
۷	منبع پردازش
۷	منبع پردازش یا Resource:
۸	دستور
۸	دستور کار یا Task
۸	چگونگی اجرای Task
۹	تابع یا FUNCTION
۹	بلوک تابع یا FUNCTION_BLOCK
۱۰	برنامه یا PROGRAM
۱۰	نسخه سازی INSTANTIATION
۱۲	مدل اجرای برنامه ها در PLC
۱۳	نوع داده ها (DATA TYPES) در استاندارد IEC1131-3
۱۳	داده های نوع پایه یا Elementary Data Types
۱۴	داده های نوع ژنریک یا Generic data types
۱۴	داده های تعریف شده توسط کاربر یا User-defined data type
۱۵	(VARIABLES) در استاندارد IEC1131-3
۱۵	کلمات کلیدی اعلان متغیر (Variable declaration keywords)
۱۶	متغیرهای عام و متغیرهای محلی Global and Local Variables
۱۷	تعیین محل استقرار متغیرهای عام در حافظه PLC
۱۹	سخت افزارهای PLC500 NSERIES
۱۹	پورتهای یا گذرگاه های ارتباطی CPU
۱۹	کابل های ارتباطی
۱۹	کابل اترنت مستقیم
۱۹	کابل اترنت معکوس
۱۹	کابل سریال RS232
۲۰	راه اندازی سرد و گرم COLD AND WARM RESTARTS
۲۲	قطع و وصل کردن کلید RUN/STOP در هنگام روشن کردن دستگاه
۲۳	گونه های مختلف حافظه TYPES OF MEMORIES
۲۳	RAM باتری دار
۲۳	RAM بدون باتری
۲۴	حافظه Flash
۲۴	محیط برنامه نویسی PLC500 سری N
۲۴	ساختن پروژه CREATING A PROJECT
۲۵	پروژه ساخته شده شامل اجزای زیر خواهد بود
۲۵	یک پروژه ی حداقلی دارای اجزای زیر است
۲۶	درخت پروژه Project tree
۲۷	چند نکته در مورد PLC500 NSERIES

۲۷ مکانیزم عملکرد WATCHDOG در CPU180
۲۸ تصاویر فرآیند در حافظه PLC
۳۰ ساختار POU
۳۱ وظایف یا دستورهای کاری PLC
۳۱ دستور کارهای های دورانی (TASKS CYCLIC)
۳۱ دستور کارهای های سیستمی (TASKS SYSTEM)
۳۱ دستور کارهای های رخدادی (TASKS EVENT)
۳۱ چگونه TASK به پروژه اضافه می شود؟
۳۲ دستورکار دورانی CYCLIC TASK
۳۳ دستورات کاری دورانی در CPU180
۳۴ دستورکار رخدادی یا وقفه ای
۳۵ دستورکار سیستمی
۳۶ SPG3، SPG2، SPG1، SPG0
۳۷ ایجاد و ویرایش برنامه PLC
۴۰ برنامه و متغیرهای آن را کامل کنیم
۴۴ نسبت دادن PROGRAM به TASK
۴۵ آرایش و ساماندهی سخت افزار PLC
۴۷ ارتباط با PLC
۵۳ برنامه ها و فایل هایی که به PLC ارسال می شوند
۵۳ ارسال برنامه ها و فایل ها به PLC
۵۶ عیب یابی (DEBUGGING) برنامه ها با دیدن STATUS اجرای برنامه ها
۵۸ ایجاد و ویرایش FB توسط کاربر
۶۱ استفاده از بلوک تابع DELAYEDAND
۶۲ تکمیل و ارسال برنامه ها به PLC
۶۴ ذخیره پروژه ها در کامپیوتر برنامه نویسی
۶۵ چه وقت از فرمت ZWT استفاده می شود؟
۶۶ ذخیره سازی برنامه های PLC در حافظه FLASH
۶۹ انبار گزاشات CPU
۷۰ ضمیمه ۱: تنظیم آدرس IP کامپیوتر پروگرامر



PLC500 N SERIES نسل دوم PLC های شرکت کنترونیک از سال ۱۳۸۷ به بازار معرفی شده است. در طراحی این محصول از آخرین دستاوردهای تکنولوژی های جدید در زمینه سخت افزار و نرم افزار استفاده شده است.

بیش از دو دهه از تولید نخستین PLC های کنترونیک گذشته است. قبل از آن PLC را به ایرانیان معرفی کردیم، با اکراه صنعتگران از بکارگیری آن مبارزه نمودیم و در همان دوران دریافتیم که یکی از کلیدهای پیشرفت صنعتی PLC است. با توجه به اینکه کنترونیک در آن دوران دستی در الکترونیک صنعتی داشت و در کار طراحی مدارات الکترونیکی و مایکروپروسسوری که در آن دوران نیز فعالیتی روزآمد بود فعالیت میکرد، ورود به عرصه ساخت و تولید PLC بسیار بدیهی بنظر می رسید.

نسل اول تولیدات کنترونیک بسیار موفق بود. هزاران دستگاه از این محصول ساخته شده و توسط صدها دانش آموخته PLC که در مرکز آموزش این شرکت دوره های آموزشی مربوطه را گذرانده بودند در صنایع مختلف ایران بکار گرفته شدند. بسیاری از پروژه های مهم صنعتی ابتدا توسط PLC های کنترونیک اجرا می شدند. بعدها پس از کسب اعتماد و اطمینان از قابلیت های PLC، این بار این تقاضا بود که از عرضه پیشی می گرفت.

با پیشرفت الکترونیک و علوم کامپیوتر، بخصوص در اواخر دهه ۹۰ قرن بیستم میلادی، چند تحول بزرگ در زمینه سیستم های کنترل الکترونیکی ایجاد شد. یکی از این تحولات مربوط به عرضه سیستم عامل Windows توسط شرکت مایکروسافت بود. سیستم عامل ویندوز بنا بر ماهیت خود شکل ارتباط کاربر با کامپیوتر را متحول میکرد. محیط کار و برنامه نویسی در کامپیوتر به زبانهای مختلف بسیار مشابهت پیدا کرده و ظاهر نرم افزارها یکسان می شدند. در سیستم عامل ویندوز امکان اجرای چند برنامه بطور همزمان فراهم گردید. از طرف دیگر استانداردهای مبادله داده ها بین برنامه های مختلف در یک کامپیوتر تدوین گردید. مبادله اطلاعات بین کامپیوترها نیز که از سالها پیش کمابیش انجام می شد شکل یکنواخت تری به خود گرفته و با ترویج اینترنت و مشاهده ی تواناییهای آن دیگر قابلیت های گسترده ارتباطی به یک خواسته اولیه در آمده بود.

از طرف دیگر طراحی نرم افزارها تخصصی تر شده و کاربرد گرافیک و help آنلاین در آنها بسیار رایج شد. این پدیده موجب ایجاد فرهنگی جامع و واژه گانی شد که در زمینه علوم مختلف فراگیر گردید. حجم مطالب آموختنی نیز بتدریج افزایش می یافت. برای کم کردن از این حجم لازم بود که برخی از واژگان پذیرفته شده در سطح کاربران بصورت استاندارد مورد پذیرش همگان قرار گرفته و در زمینه های مختلف بکار روند.

بخش صنعت نیز از این تحولات بی نصیب نماند. استانداردهایی برای برنامه نویسی سیستم های کنترلی تدوین شد. این استانداردها با استفاده از تجربیاتی که در سایر بخش های علوم کامپیوتری و زبان های برنامه نویسی بدست آمده بودند تدوین گردیدند.

برای پیوستن به مسیری که پیش بینی می شد به راه درستی می رود، تصمیم گرفتیم که از همکاریهای بین المللی استفاده کنیم. در سال ۲۰۰۰ میلادی با شرکت آلمانی KW-Software برای خرید لیسانس جامع نرم افزار برنامه نویسی plc بنام MULTIPROG-wt یا به اختصار MWT به توافق رسیدیم.

در بخش سخت افزار نیز پیش بینی می کردیم که علاوه بر بکارگیری پردازنده های مدرن و سریع، یکی از محورهای اصلی توسعه در بخش شبکه های ارتباطی باشد. بر خلاف بسیاری از سازندگان که لایه فیزیکی ارتباطی را همچنان در روشهای سنتی (مثل ارتباطات سریال RS232 و RS485) می جستند، تشخیص ما این بود

که لایه فیزیکی اترنت آینده روشن تری داشته باشد. به موازات آن پشتیبانی از استانداردهای ارتباطی بر مبنای IP انتخاب نهایی بنظر می رسید.

محصول جدید کنترونیک، PLC500 NSERIES، با استفاده از تجربیات ارزشمند کسب شده از حضور طولانی در بازار PLC و با نگاه به آینده ساخته شده و در طراحی آن استفاده از استانداردهای زبانهای برنامه نویسی بنام IEC61131-3 کاملاً رعایت شده است.

در PLC500 NSERIES امکانات گسترده ارتباطی نیز در نظر گرفته شده است. برخلاف سایر PLC ها که برای پیاده سازی ارتباطات متنوع از کارتهای متنوع اضافی در راک PLC استفاده می کنند، کارت CPU در این PLC با دارا بودن پورتهای سریال RS232، RS485 و ETHERNET مستقیماً پروتکل های ارتباطی متنوعی را پشتیبانی کرده و نیازی به کارتهای اضافی نمی باشد. ارتباط اترنت با سرعت ۱۰۰ مگابیت در ثانیه سریعترین ارتباطات موجود در صنعت را برای کاربر فراهم می نماید.

در سمت کامپیوتر نیز نیازی به کارت یا مدول ارتباطی خاصی نداریم. امروزه تمام کامپیوترها به پورتهای ETHERNET یا SERIAL مجهزند. کافی است توسط یک کابل مناسب PLC را به کامپیوتر متصل نمایید.

کار با PLC500 NSERIES به آشنایی اولیه کار با MWT همراه با رابطهای خاصی دارد که کنترونیک به آن اضافه کرده است. این رابطها معمولاً بشکل پنجره های گرافیکی هستند که ارتباط بین MWT و PLC500 NSERIES را برقرار مینمایند.

هدف اصلی این متن، آموزش نحوه کار در محیط MWT برای برنامه نویسی PLC500 NSERIES و نشان دادن شکل پیاده سازی استاندارد IEC1131-3 بر روی این محصول است. همراه با این گفتار مواردی پیش می آید که شرح مختصری درباره MWT و استاندارد IEC1131-2 گریز ناپذیر میشود. در این موارد توضیح مختصری داده میشود لیکن برای توضیح کامل MWT لطفاً به کتاب MWTMAN21-001.PDF مراجعه فرمائید.

از آنجایی که PLC500 NSERIES کنترلی مدرن با قدرت و سرعت پردازش بالاست و توان کنترل فرآیندهای پیچیده و بزرگ را دارد، کار با آن مستلزم آموزش های عمیقی است که استفاده از امکانات آن را میسر سازد. خوشبختانه طراحی سخت افزار و نرم افزار آن به گونه ایست که فراگیری آموزشهای آن در مسیری هموار، مستقیم و روان (Stream lined) انجام می شود.

هر چند که کاربر در ابتدا باید زمان بیشتری را صرف آموزش های اولیه کند، لیکن پس از کسب مهارتهای اولیه پروژه های بعدی را بسیار سریع انجام می دهد. کاربر بزودی درمی یابد که در تمام مراحل کار با فلسفه کاری مشخصی روبروست که بدون استثنا و در همه جا دنبال می شود. این دستاورد بخاطر پیروی کامل از استاندارد جامع IEC1131-3 و استفاده از مفاهیم جا افتاده در علوم کامپیوتر و انفورماتیک بدست آمده است. مفاهیمی که اگر کاربر یکبار با آنها آشنا شود، همیشه با اوست و در کار با تمام سیستمها به او کمک می کند.

کاربر همچنین در می یابد که ساختار نرم افزار این PLC شباهت بسیاری با سایر نرم افزارهای برنامه نویسی های سطح بالا دارد. این خاصیت به او قدرت پیش بینی میدهد که برای برخورد با هر موضوع جدیدی چگونه برخورد کرده و چه مسیری را دنبال کند. از جنبه سخت افزاری نیز کاربر با کارتهای و مدولهای plug & play سروکار دارد که شیوه نصب آنها با استفاده از درایورهای استاندارد بسیار سراسر است.

با برخورداری از پردازندهای قوی PLC500 NSERIES تمام دستورالعملهای استاندارد را پشتیبانی می کند. علاوه بر انجام عملیات منطقی که نیازمندی اولیه سیستمهای کنترلی است، دستورالعملهای محاسباتی چهار عمل اصلی با اعداد صحیح و اعشاری و همچنین دستورالعملهای محاسبات توابع مثلثاتی را انجام می دهد.

تلاش بسیاری به عمل آمده است که خانواده PLC500 NSERIES سیستمی باز (Open System) باشد. پردازنده های مختلف این محصول پروتکل رایج Modbus-TCP را مستقیماً و بدون واسطه پشتیبانی می کنند. این بدین معنی است که کاربر می تواند تمام پانلهای اپراتوری استاندارد موجود در بازار جهانی را که دارای چنین

پروتکلی باشد بعنوان رابط بهره برداری (HMI) مورد استفاده قرار دهند. برای انجام امور مونتورینگ توسط کامپیوترها نیز بسته نرم افزاری OPC-Server آن در اختیار کاربران قرار می گیرد. کاربرانی که آشنایی با هر یک از بسته های نرم افزاری مونتورینگ داشته باشند می توانند بدون واسطه و وابستگی از دانش قبلی خود استفاده نمایند.

برای پیاده سازی برخی از مفاهیم فراتر از استاندارد چون کنترل PID، برقراری شبکه DCS، کار با کارت کنترل مکان و ... توابع استاندارد در CPU این سیستمها تعبیه شده است. شکل استفاده از این توابع در انطباق کامل با شیوه استاندارد بوده و به هیچ آموزش خاصی فراتر از آشنایی با پارامترهای ورودی و خروجی های آنها نیست.

از ویژگی های مهم دیگر CPU های خانواده PLC500 NSERIES می توان به امکانات گسترده ارتباطی آن اشاره کرد. CPU های مختلف این خانواده دارای گذرگاه های ارتباطی سریال و Ethernet است. بنابراین برای بکارگیری این PLC ها در شبکه های مدرن اترنتی به هیچ مدول اضافی دیگری نیازی نیست. برای راه اندازی آن حتی نیازی به دانستن جزئیات کاری و دانش خاص شبکه های اترنتی نیز وجود ندارد. کمترین حسن و نه کم اهمیت ترین خاصیت این گذرگاه در اینست که در سمت کامپیوتر پروگرامر دیگر نیازی به اداپتور مخصوص برای ارتباط با PLC وجود ندارد با بررسی های انجام شده، کاراترین سیستم و استاندارد ارتباطی که چند سالی است رایج شده و تا ده ها سال آینده نیز به عنوان انتخاب برتر خواهد بود همین سیستم ارتباطات اترنتی است.



آشنایی با استاندارد IEC1131-3 و نسخه جدیدتر آن IEC61131-3

استاندارد IEC1131-3 برای استاندارد کردن زبان های برنامه نویسی PLC تدوین شده است. مجموعه دستورالعمل ها و مفاهیم مختلفی که در زمینه سیستمهای اتوماسیون رایج شده موجب تنوع زیاد مفاهیم و مطالب آموختنی در PLC ها و سازندگان گوناگون گردیده است. این تنوع و ناسازگاری کاربران را وادار به سرمایه گذاریهای زیاد در بخش آموزش سخت افزار و نرم افزاری نمایند.

IEC1131 زبانهای برنامه نویسی، مجموعه دستورالعملها و ساختار پروژه ها را بصورت استاندارد تعریف می کند. مزایای استفاده از PLC های منطبق بر استاندارد IEC1131، همسان سازی آموزش و ایجاد زمینه ای مشترک برای انتقال برنامه ها از یک سیستم به سیستمی دیگر است.

استاندارد IEC1131 دارای چندین بخش و شامل گزارشات فنی مختلفی است. بخش سوم استاندارد در مورد زبانهای برنامه نویسی است.

بدیهی است که این استاندارد تاثیر بسیاری بر مفاهیم، ساختارها، ویژگیها و قابلیتهای سیستمهای برنامه نویسی PLC دارد.

بیشترین تاثیراتی که IEC1131-3 با خود آورده در باره چنین مواردی است:

- تعریف متغیرها (Declaration of variables) شبیه زبانهای برنامه نویسی سطح بالا شده است
- تعریف نوع داده ها (Declaration of data types) ممکن گشته است
- متغیرهای محلی (Local Variables) و متغیرهای عام (Global Variables) قابل تعریف و تشخیص شده اند
- مفهوم برنامه نویسی به برنامه نویسی سمبولیک تغییر جهت داده است

اجزای تشکیل دهنده سامانه در استاندارد IEC1131-3

در یک PLC منطبق با استاندارد، اجزای ساختاری (Configuration Elements)، ساختار سخت افزار PLC را منعکس می نماید. اجزای ساختاری عبارتند از:

- پیکربندی یا Configuration
- منابع پردازش یا Resources
- دستورات کاری یا Tasks

پیکربندی یا Configuration

پیکربندی را می توان ساختار عمومی PLC یا همان راک (Rack) در نظر گرفت. در هر پیکربندی می توان یک تا چند منبع پردازشی (Resource) قرار داد.

منبع پردازش یا Resource

Resource را میتوان پردازنده اصلی یا CPU تلقی کرد که می توان آن را در راک PLC نصب کرد. در یک Resource میتوان متغیرهای عام (Global variables) را تعریف کرد که تنها برای همان Resource اعتبار داشته باشند. هر Resource یک یا چند دستور کار (Task) را می تواند اجرا کند.

دستور کار یا Task

Task زمان بندی اجرای برنامه های نسبت داده شده به خود را به عهده دارد. پس باید برنامه ها (Programs) را به Task ها نسبت داد. تنظیمات Task ، زمان اجرای آن را مشخص می کند.

چگونگی اجرای Task

IEC1131-3 سه نوع زمان بندی برای اجرای Task ها را شرح داده است که عبارتند از:

- دستور کارهای دورانی (Cyclic Tasks) که در مقاطع زمانی خاصی پی در پی فعال شده و اجرا می شوند.
- دستور کارهای سیستمی (System Tasks) در شرایط خاصی مثل تغییر وضعیت کاری CPU و یا بروز خطا (مثل استارت های سرد و گرم و یا خرابی کارتهای I/O) بطور اتوماتیک فعال میشوند. Task های سیستمی را بنام برنامه های سیستمی (SPG) که مخفف عبارت System Program است نیز می شناسند.
- دستور کارهای رخدادی (Event Tasks) در صورت وقوع رخدادهای خاص (Interrupt Events) فعال می شوند.

هر Task ای دارای درجه ی اولویت اجرای مشخصی است. اولویت اجرا بدین معنی است که در صورتی که به طور همزمان نوبت اجرای چند Task فرا رسد Task ای که درجه اولویت بالاتری داشته باشد زودتر اجرا می شود.

POU - واحد ساماندهی برنامه

POU مخفف عبارت Program Organization Unit است. POU ها واحدهای کوچک و مستقل زبان برنامه نویسی PLC هستند. نام هر POU باید در کل پروژه منحصر به فرد باشد.

در IEC61131-3 سه نوع POU بر اساس تفاوت در کاربرد آنها قابل تفکیک هستند

- **PROGRAM** یا برنامه که به اختصار **PROG** نیز نامیده میشود
- **FUNCTION** یا تابع که به اختصار **FU** نیز نامیده میشود
- **FUNCTION-BLOCK** یا بلوک تابع که به اختصار **FB** نیز نامیده میشود

هر POU شامل دو بخش متفاوت است. بخش معرفی متغیرها (Variable Declaration) و بخش بدنه کدهای برنامه (Code Body)

در بخش معرفی متغیرها (Variable Declaration) تمام متغیرهای POU معرفی می شوند.

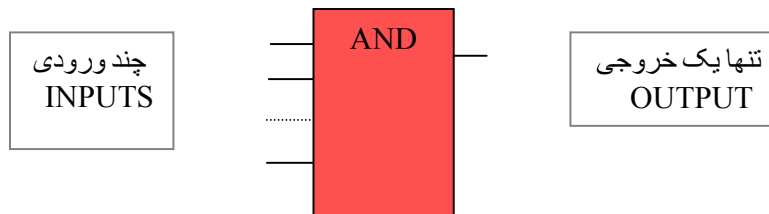
در بخش Code Body دستورات و توابع کنترلی به زبان مورد نظر نوشته می شوند.

تابع یا FUNCTION

FUNCTION یا FU نوعی POU با چند ورودی و تنها یک خروجی است. استفاده از یک FU با مقادیر ورودی یکسان همواره خروجی یکسانی تولید می کند. در داخل یک FU میتوان از FU های دیگری نیز استفاده کرد ولی نمی توان از FB های دیگر بکار برد.

توابعی چون AND, XOR, OR و ... از نوع FU هستند.

استاندارد IEC1131-3 توابع بسیاری را تعریف کرده که هنگام ویرایش برنامه بکار می روند. PLC500 NSERIES تمام توابع موجود در استاندارد را پشتیبانی می کنند. به این توابع Firmware Functions یا به اختصار FW-FU گفته می شود. دلیل این نام گذاری این است که این توابع در حافظه دائمی CPU قرار داشته و غیر قابل تغییرند.



بلوک تابع یا FUNCTION_BLOCK

FUNCTION-BLOCK یا FB نوعی POU با چند ورودی و چند خروجی است که حافظه داخلی نیز دارد. خروجی های FB علاوه بر تاثیر پذیری از وضعیت ورودی ها، بستگی به مقادیر حافظه داخلی هم دارند. به عبارت دیگر تنها ورودی های FB نیستند که وضعیت خروجی ها را تعیین می کنند بلکه تاریخچه فعالیت های قبلی آن نیز در تعیین مقادیر خروجی ها موثرند. در داخل یک FB میتوان از FB ها و FU های دیگر نیز استفاده کرد.

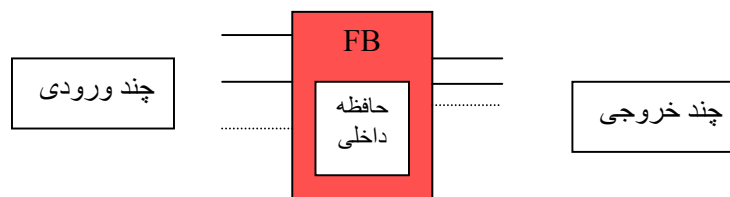
تایمرها نوعی FB هستند. خروجی تایمرها نه تنها بستگی به وضعیت ورودی های آن دارد، بلکه به زمان سپری شده (تاریخچه آن) نیز وابسته است. در این FB یکی از کاربردهای حافظه داخلی، نگهداری زمان سپری شده است.

در استاندارد IEC1131-3 چند FB تعریف شده که می توانند هنگام ویرایش برنامه بکار روند. PLC500 NSERIES تمام توابع بلوکی استاندارد را پشتیبانی می کند. علاوه بر آنها تعدادی FB دیگر نیز در حافظه دائمی CPU اضافه شده اند که کاربردهای ویژه دارند. به تمام FB هایی که در حافظه دائمی CPU نگهداری می شوند Firmware Function Blocks یا به اختصار FW-FB گفته می شود. دلیل این نام گذاری این است که این توابع در حافظه دائمی CPU قرار داشته و غیر قابل تغییرند.

آن دسته از FB هایی که در استاندارد IEC1131-3 تعریف شده اند را **Standard FB** مینامیم. از این نوع FB ها می توان در هر POU ای بکار برد.

FB های مازاد بر استاندارد ی که در هر خانواده ی PLC وجود دارند را **PLC-Specific FB** می نامند.
FB های مازاد بر استاندارد ی که در هر مدل CPU از خانواده ی PLC وجود دارند را **CPU-Specific FB** می نامند.

در MWT هنگام ساختن POU در پنجره مخصوصی که مشخصات آن را وارد می کنید، نوع PLC و نوع CPU را نیز می توانید انتخاب کنید. این انتخاب، دسترسی به FB های افزون بر استاندارد را میسر می سازد.



برنامه یا PROGRAM

برنامه یا Program نوعی POU است که در آن ترکیب های منطقی ای از توابع (FU) و بلوک توابع (FB) بر اساس نیاز های کنترلی فرآیند قرار داده می شوند. PROG ها می توانند پارامترهای ورودی و خروجی نیز داشته باشند (هر چند بندرت از آنها استفاده می شود) ولی معمولاً دارای حافظه ی داخلی (محلی) هستند.

عملکرد و کاربری PROG شبیه FB است با تفاوت های زیر

- PROG ها را برای اجرا باید به Task ها نسبت داد
- FB ها را برای اجرا باید در PROG صدا زد

در داخل PROG می توان از FU ها و FB ها استفاده کرد ولی نمی توان سایر PROG ها را فراخوانی کرد.

نسخه سازی Instantiation

برای استفاده ی مکرر از FB ها، IEC1131-3 امکان نسخه سازی را فراهم کرده است. این موضوع بدین مفهوم است که هر FB یکبار ساخته میشود ولی چندین بار مورد استفاده قرار می گیرد. استفاده مکرر از یک FB چطور امکان دارد در حالیکه FB دارای حافظه ی داخلی است؟ در این صورت حافظه ی داخلی تاریخچه ی عملکرد کدامیک از موارد کاربردی را در خود نگه میدارد؟ این پرسش مهم را با تکنیکی بنام نسخه سازی (Instantiation) می توان پاسخ داد.

زمانی که یک FB ساخته می شود، در حقیقت الگوی عملکرد آن ساخته می شود. هنگامی که از آن FB در برنامه خود استفاده می کنید، در حقیقت نسخه ای مطابق با آن الگو را به برنامه ی خود اضافه می کنید. الگوی همه مشخصات FB از نظر تعداد و نوع I/O ها و میزان حافظه ی مورد نیاز را مشخص می کند. اختصاص این حافظه از محل حافظه ی داخلی POU ایست که FB را در آن قرار داده ایم. پس چنانچه هر نسخه ای از FB در برنامه اضافه شود، به اندازه ی مورد نیاز برای آن حافظه مجزا اختصاص می یابد

برای تفکیک نسخه های مختلف یک FB، از نام نسخه (Instance Name) استفاده می شود. اسامی نسخه ها نباید مثل هم باشند. در غیر این صورت فضای حافظه های آنها مشترک شده و در کار FB اختلال ایجاد خواهد

شد. در صورتیکه در یک POU از یک FB استفاده می کنید، نام نسخه ی FB را باید در لیست متغیر های محلی POU قرار داد. البته این کار در MWT بشکل خودکار انجام میشود. کافی است که یک FB مثل TON را در یک PROG قرار داده و سپس لیست متغیر های محلی آن را مشاهده کنید. مشاهده می کنید که یک نسخه از TON به نام مثلا TON_1 به لیست اضافه شده است. توجه داشته باشید که در MWT با پاک کردن FB از برنامه، نسخه ی FB از لیست پاک نمیشود و شما باید دستی آن را حذف کنید. در مورد متغیر های محلی در ادامه گفتگو خواهیم کرد. نسخه سازی منحصر به FB ها نیست و می توان از PROG ها نیز نسخه سازی کرد. هر چند معمولاً تنها یک نسخه از هر PROG در یک پروژه بکار میرود، ولی به هر حال امکان استفاده از نسخه های متفاوت یک PROG در هر Resource وجود دارد.

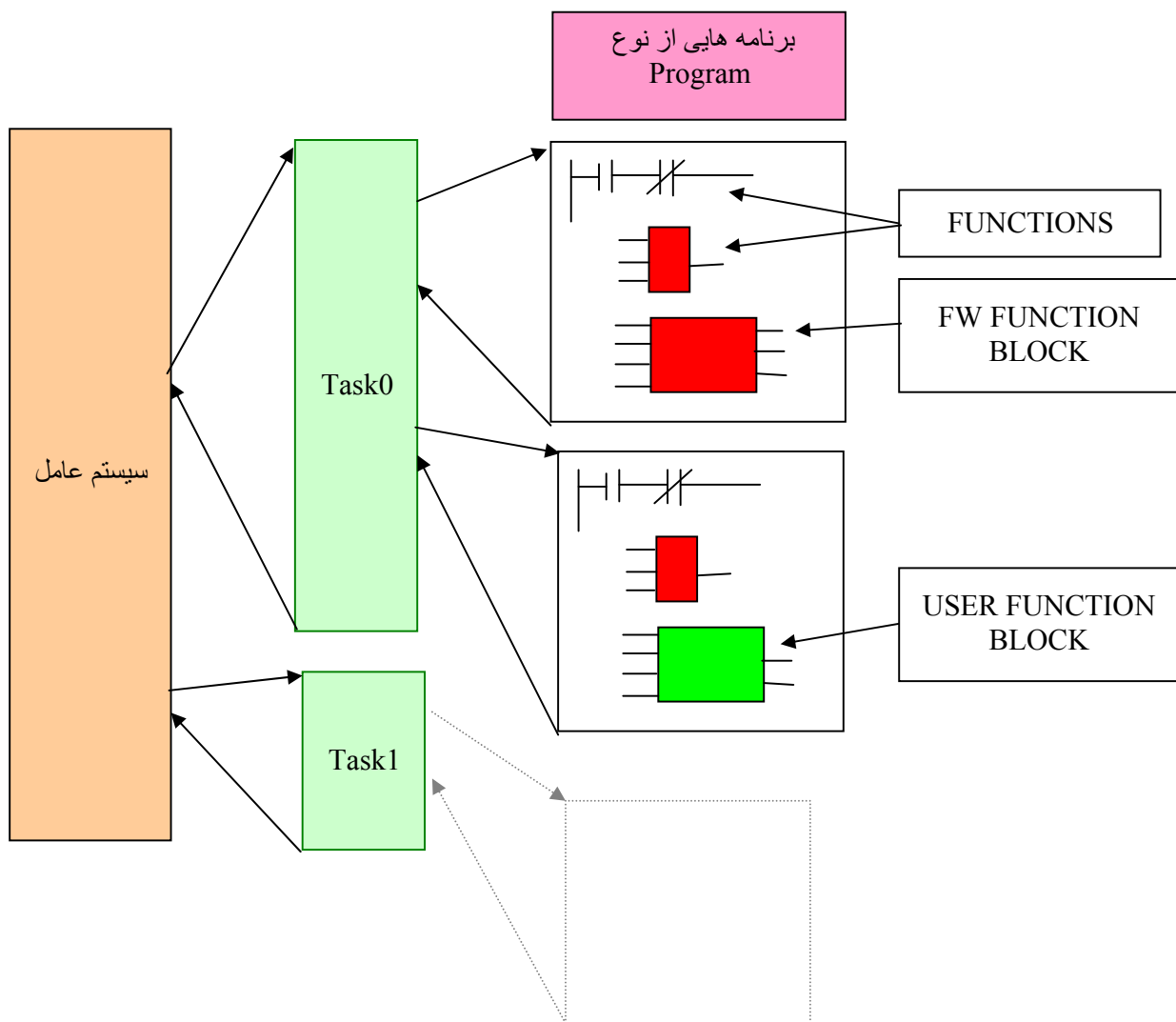


مدل اجرای برنامه ها در PLC

در شکل زیر چگونگی اجرای برنامه در PLC دیده می شود. سیستم عامل دستورکارهایی بنام Task را مطابق با مشخصات Task برای اجرا فرامیخواند. با فرارسیدن نوبت اجرای هر Task تمام POU هایی از جنس Program که به آن Task نسبت داده شده اند نیز اجرا خواهند شد.

توابع (Functions) و بلوکهای تابع (Function blocks) ای که در Program ها قرار دارند نیز به همین صورت اجرا می شوند. برخی از این توابع از نوع داخلی (Firmware) و برخی دیگر توسط کاربر نوشته شده اند (User Function Block).

بنا بر این توجه داشته باشید که تنها POU های از جنس Program را می توان به Task نسبت داد.



نوع داده ها (Data Types) در استاندارد IEC1131-3

نوع داده یا Data type مشخص می کند که هر متغیر (variable) چه مقادیری را می تواند اختیار نماید. نوع داده یا Data type مقدار اولیه، محدوده مقادیر ممکن و تعداد بیتها یا بایتهای مورد نیاز را تعریف می کند.

در استاندارد IEC1131-3 سه خانواده از Data type ها تعریف شده اند:

- داده های نوع پایه یا Elementary data types
- داده های نوع ژنریک یا Generic data types
- داده های نوع تعریف شده توسط کاربر یا User defined data types

داده های نوع پایه یا Elementary Data Types

مقادیر و اندازه داده های elementary data type در استاندارد IEC1131-3 مشخص شده و در جدول زیر مشاهده می شوند.

نوع داد Data type	شرح Description	اندازه Size	محدوده Range	مقدار اولیه Initial Value
BOOL	عدد بول یا عدد منطقی Boolean	1	0...1	0
SINT	عدد صحیح کوتاه Short Integer	8	-128...127	0
INT	عدد صحیح Integer	16	-32768...32767	0
DINT	عدد صحیح بلند Double Integer	32	-2,147,483,648 ... 2,147,483,647	0
USINT	عدد صحیح بی علامت کوتاه Unsigned Short Integer	8	0...255	0
UINT	عدد صحیح بی علامت Unsigned Integer	16	0...65535	0
UDINT	عدد صحیح بلند بی علامت Unsigned Double Integer	32	0...4,294,967,295	0
REAL	عدد حقیقی یا اعشاری Real Numbers	32	$3.4^{-38} \dots 3.4^{38}$	0
TIME	زمان Duration	32		T#0s
BYTE	زنجیره ای از بیتها بطول ۸ Bit string of length 8	8		0
WORD	زنجیره ای از بیتها بطول ۱۶ Bit string of length 16	16		0
DWORD	زنجیره ای از بیتها بطول ۳۲ Bit string of length 32	32		0

داده های نوع ژنریک یا Generic data types

این نوع داده ها در حقیقت شامل گروه های طبقه بندی شده ای از همان داده های Elementary data type هستند. مثلاً ANY_INT به معنی "هر عدد صحیحی" شامل تمام انواع DINT, INT, SINT, UDINT, UINT و USINT میشود. اگر مثلاً نوع ورودی های تابعی از جنس ANY_INT باشد، به معنی آن است که آن تابع هر یک از انواع عدد صحیح بالا را می تواند به عنوان ورودی بپذیرد.

داده های نوع ژنریک در جدول زیر نشان داده شده اند.

ANY

ANY_NUM

ANY_REAL
REAL

ANY_INT
DINT, INT, SINT
UDINT, UINT, USINT

ANY_BIT
DWORD, WORD, BYTE, BOOL

STRING

TIME



مثال: در help تابع AND، نوع ورودی ها بصورت زیر ذکر شده است.

Parameter	Data types	Description
IN1	ANY_BIT	input value
IN2	ANY_BIT	input value
OUT	ANY_BIT	output value

یعنی تابع AND ورودی های خود را از هر یک از انواع BOOL، BYTE، WORD و DWORD پذیرفته و آنها را با هم AND می کند. بدیهی است که البته باید همه آنها یکسان باشند. مثلاً می تواند دو عدد منطقی (Bool) را با هم AND کرده و پاسخ آن را در بصورت Bool در خروجی ظاهر کند. همین تابع میتواند دو بایت را با هم AND کرده و نتیجه حاصل را بصورت بایت در خروجی قرار دهد.

داده های تعریف شده توسط کاربر یا User-defined data types

علاوه بر داده های پایه ای کاربران نیز می توانند انواع دیگری از داده ها را تعریف کنند. داده هایی مانند آرایه ها و ساختارها از این نوع داده ها هستند. در حال حاضر PLC500 NSERIES این نوع داده ها را پشتیبانی نمی کند.

مطالب ارائه شده در مبحثی که بدنبال می آید ممکن است در حلقه اول کمی انتزاعی و غیر قابل لمس باشد. بنا براین توصیه می شود ابتدا مطالب را بصورت کلی مرور کنید. در مرور اولیه ممکن است که همه مطالب را صد در صد درک نکنید. در مباحث بعدی که کاربری آنها را ضمن برنامه نویسی برای PLC مشاهده خواهید کرد، بتدریج با جزییات آنها آشنا خواهید شد. پس از کسب مهارت های اولیه بسیار مفید خواهد بود که مجدداً به مطالعه این بخش بازگردید.

IEC1131-3 (Variables) در استاندارد

در استاندارد IEC1131-3، سه نوع متغیر تشریح شده است

- متغیرهای سمبولی Symbolic variables
- متغیرهایی که مستقیماً ارائه می شوند Directly represented variables
- متغیرهای مستقر (با محل مشخص در حافظه) Located variables

متغیرها را باید در صفحه کار متغیرهای POU با کمک کلمات کلیدی (Keywords) اعلان کرد.

نکته: بدلیل کاربرد اندک، ایجاد ابهام و زمینه دادن به تکرار عادت های قدیمی، PLC500 NSERIES متغیرهای نوع دوم (Directly represented) را پشتیبانی نمی کند و ما نیز این موضوع را شرح نخواهیم داد.

کلمات کلیدی اعلان متغیر (Variable declaration keywords)

هنگام اعلان متغیرها باید از "کلمات کلیدی اعلان متغیر" استفاده کرد. این کلمات کلیدی در جدول زیر تشریح شده اند.

شرح	کلمه کلیدی (Keyword)
<ul style="list-style-type: none"> • برای اعلان متغیرهای محلی یا داخلی که می توانند تنها در داخل POU بکار روند • برای اعلان نسخه ای از یک FB (FB instance) • برای اعلان متغیرهایی که مستقیماً اعلان می شوند • چنانچه همراه با کلمه کلیدی "RETAIN" باشد ویژگی پایداری (Retentive) را هم خواهد داشت 	VAR
<ul style="list-style-type: none"> • برای اعلان متغیرهایی که به عنوان ورودی به FU یا FB ای که در حال تدوین آن هستید بکار می روند • تنها برای اعلان متغیرها بصورت سمبلی بکار می رود 	VAR_INPUT
<ul style="list-style-type: none"> • برای اعلان متغیرهایی که به عنوان خروجی از FU یا FB ای که در حال تدوین آن هستید بکار می رود • مقدار آن در FU یا FB نوشته می شود • ضمناً می توان مقدار آن را خواند • با کمک کلمه کلیدی RETAIN، می توان آن را پایدار نیز نمود 	VAR_OUTPUT
<ul style="list-style-type: none"> • آدرس متغیر به POU ی دیگر پاس داده می شود • متغیر هم خواندنی و هم نوشتنی است • معمولاً برای داده های بی نوع پیچیده مثل آرایه ها و ساختارها بکار می رود 	VAR_IN_OUT

<ul style="list-style-type: none"> • برای اعلان متغیرهای عام مورد استفاده در POU بکار می رود • تعریف اصلی آن قبلا توسط کلمه کلیدی VAR_GLOBAL در لیست متغیرهای عام اعلان شده است • این متغیرها را میتوان در هر POU ای تغییر داد 	VAR_EXTERNAL
<ul style="list-style-type: none"> • برای اعلان متغیرهایی که در سراسر پروژه و در همه POUها اعتبار دارند • با کمک کلمه کلیدی RETAIN، می توان آن را پایدار نیز نمود 	VAR_GLOBAL
<ul style="list-style-type: none"> • برای پایان دادن به بلوک اعلان متغیرها 	END_VAR

علاوه بر کلمات کلیدی تشریح شده، دو کلمه کلیدی دیگر نیز هنگام معرفی و اعلان متغیرها وجود دارد.

- کلمه کلیدی RETAIN برای متغیرهای پایدار که در WARM_RESTART آخرین مقدار خود را حفظ می کنند.
- کلمه کلیدی AT که محل قرارگیری متغیر در حافظه را مشخص می کند.

اعلان متغیرها یا در صفحه کار متغیرهای POU و یا در صفحه کار متغیرهای عام (Global Variables) انجام می شود.

متغیرهای عام و متغیرهای محلی Global and Local Variables

اعتبار هر متغیر یا به یک POU یا به کل پروژه محدود می شود. بنا براین از دیدگاه حوزه اعتبار متغیرها دو نوع متغیر قابل تشخیص است.

- متغیرهای محلی یا Local variables
- متغیرهای عام یا Global variables

اگر متغیری تنها در محدوده یک POU اعتبار داشته باشد به آن متغیر محلی گفته می شود. در چنین مواردی می توان از کلمه های کلیدی VAR، VAR_INPUT و VAR_OUTPUT برای تعریف آن استفاده کرد.

اگر متغیری در تمام پروژه اعتبار داشته باشد به آن متغیر عام گفته می شود. برای تعریف و اعلان متغیرهای عام از کلمه کلیدی VAR_GLOBAL استفاده می شود. در POU هایی که قصد استفاده از آنها را داریم نیز از کلمه کلیدی VAR_EXTERNAL استفاده می کنیم.

نکته ۱: برای تمام متغیرهای عام میتوان مقدار اولیه ای (Initial Value) در نظر گرفت. مقدار اولیه متغیرهای عام بلافاصله پس از ارسال فایل GlobalVar به PLC اعمال می شوند. در سایر موارد به چگونگی راه اندازی سرد و گرم PLC مراجعه بستگی دارد.

نکته ۲: برای تمام متغیرهای محلی میتوان مقدار اولیه ای (Initial Value) در نظر گرفت. مقدار اولیه متغیرهای محلی بلافاصله پس از ارسال POU به PLC اعمال می شوند. در سایر موارد به چگونگی راه اندازی سرد و گرم PLC بستگی دارد.

تعیین محل استقرار متغیرهای عام در حافظه PLC

ورودی ها پس از ورود به PLC در ناحیه ای از حافظه ی PLC بنام ناحیه I قرار می گیرند. خروجی های ساخته شده نیز قبل از اعمال به ترمینالهای خروجی PLC در ناحیه ای از حافظه بنام Q قرار میگیرند. بجز I/O ها نواحی دیگری نیز برای مبادله اطلاعات بین بخشهای مختلف و POU های مختلف پیش بینی شده است که در دسترس تمام POU ها قرار دارند. یکی از این نواحی ناحیه ای بنام M است که حرف اول حافظه یا Memory است.

برای استقرار متغیرهای مستقر (Located variables) در هر یک از این نواحی، باید فضای متناسبی را با توجه به اندازه ی آنها اختصاص داد. در استاندارد IEC1131-3 از بیانیه ها و کلمات کلیدی خاصی به این منظور استفاده می شود.

برای اعلان متغیر از یک نام و یک آدرس منطقی استفاده میشود. بیانیه ی تعیین محل استقرار شامل کلمه کلیدی AT، علامت %، پیشوند محل استقرار، پیشوند سایز و آدرس استقرار می باشد.

جدول زیر پیشوندهای محل و سایز متغیرهای مستقر را نشان می دهد.

پیشوند محل استقرار	شرح
I	Physical input
Q	Physical output
M	Physical address in the PLC memory
پیشوند سایز	شرح
X	Single bit size (BOOL نوع های)
None	Single bit size
B	Byte size (8 bits)
W	Word size (16 bits)
D	Double word size (32 bits)

معمولا بهتر است که تمام I/O ها را از نوع عام و با مشخص کردن محل آن تعریف کنیم.

مثلا اگر در صفحه کار متغیرهای عام (GlobalVars) چند ورودی منطقی و یک متغیر از نوع BYTE از PLC را مطابق عبارات زیر داشته باشیم:

```

VAR_GLOBAL
    Start_Command    AT    %IX0.0    :    BOOL;
    Stop_Command     AT    %IX0.1    :    BOOL;
    MyByte            AT    %MB34     :    BYTE;
END_VAR
    
```

سپس در صفحه کار متغیرهای POU میتوان از آنها به این شکل استفاده کرد:

```

VAR_EXTERNAL
    Start_Command    :    BOOL;
    Stop_Command     :    BOOL;
    MyByte            :    BYTE;
END_VAR
    
```

در PLC500 NSERIES ناحیه دیگری از حافظه بنام S نیز پیش بینی شده است. اگر کاربر متغیری را بصورت عام اعلان کند ولی محل استقرار آن را معلوم نکند، محل استقرار آن از ناحیه S در نظر گرفته می شود. بدین ترتیب برنامه نویس مدیریت استقرار متغیرهای عام مستقر را به عهده خود سیستم واگذار می کند. در حالیکه اگر استقرار آنها را خود به عهده بگیرد باید مراقب هم پوشانی نواحی حافظه باشد.

برای اعلان متغیرهای محلی در POU هایی از جنس PROG همانطور که گفته شد از کلمه کلیدی VAR استفاده می شود. به این متغیرها، متغیرهای سمبولیک هم گفته می شود. برای تعریف آنها تنها کافی است نام و نوع متغیر را مشخص کنید. آدرس و محل قرار گیری چنین متغیرهایی از دید کاربر پنهان خواهد بود.

VAR

```
LubOil_Ready      :      BOOL ;  
Fault              :      BOOL ;  
Motor1_Temperature :      INT  ;
```

....

END_VAR

مادام که در این POU قرار داریم همواره میتوانیم از این متغیرها استفاده کنیم. در خارج از این POU متغیرهای فوق ناشناسند.

مزیت عمده این روش در این است که برای موارد غیر ضروری لازم نیست از فضای حافظه با ارزشی چون ناحیه M یا S استفاده کنیم.

در PLC500 NSERIES متغیرهای محلی هر POU در محلی از همان POU مستقر می شوند. بدین ترتیب با ایجاد هر POU، حافظه اختصاصی برای متغیرهای محلی همان POU ایجاد میشود و با پاک کردن آن POU، حافظه اختصاص یافته نیز آزاد میشود. این کار بصورت خودکار انجام شده و کاربر از خطاهای احتمالی مصون خواهد بود.

سخت افزارهای PLC500 NSERIES

سخت افزار PLC500 NSERIES شامل مدولها یا کارتهای متنوعی است که در راکهای ۱۹ اینچ و ۱۲ اینچ نصب می گردند. علاوه بر کارت مجتمع CPU و منبع تغذیه، در راکهای ۱۹ اینچ ۱۰ شیار و در راکهای ۱۲ اینچ ۵ شیار برای نصب سایر کارتهای I/O پیش بینی شده است.

- کارت CPU و منبع تغذیه در انتهای سمت چپ راک نصب می شود. مدلهای مختلف CPU با توانایی های مختلف وجود دارد
- کارت ورودی دیجیتال ۱۶ یا ۳۲ کاناله
- کارت خروجی دیجیتال ۱۶ یا ۳۲ کاناله
- کارت ورودی آنالوگ ۴ یا ۸ کاناله
- کارت خروجی آنالوگ ۴ یا ۸ کاناله
- کارت ورودی آنالوگ ۴ یا ۸ کاناله برای اندازه گیری دما توسط RTD
- کارت ورودی آنالوگ ۴ یا ۸ کاناله برای اندازه گیری دما توسط ترموکوپل نوع J
- کارت ورودی آنالوگ ۴ یا ۸ کاناله برای اندازه گیری دما توسط ترموکوپل نوع K
- کارت کنترل مکان یا سرعت ۲ محوره
- کارت واکنش سریع با ۱۶ ورودی دیجیتال سریع و ۸ خروجی دیجیتال سریع
- و سایر کارتهای دیگر که در کتابهای مربوطه تشریح شده اند

کلیه کارتها دارای ایزولاسیون نوری بوده و بصورت Plug & Play در راک نصب می شوند. برای نصب آنها در راک نیازی به تنظیمات سخت افزاری مثل تنظیم (Dip switch) و یا رعایت قانون مندی خاص در محل نصب هر کارت نیست. مشخصات کامل سخت افزارها در کتاب مربوطه تشریح شده است.

پورتهای یا گذرگاه های ارتباطی CPU

بر روی مدول CPU دو گذرگاه ارتباطی (Port) استاندارد و یک گذرگاه انتخابی دیگر نصب شده است.

- پورت سریال RS232 که مستقیماً میتواند به کامپیوتر یا سایر وسایل سریال وصل شود. از این پورت می توان برای برنامه نویسی PLC استفاده نمود. در سمت کامپیوتر و در برنامه MULTIPROG باید پورت سریال بعنوان گذرگاه ارتباطی انتخاب شود
- پورت Ethernet که می تواند مستقیم و یا از طریق شبکه LAN، کامپیوتر و PLC را به یکدیگر ارتباط دهد. برای اتصال مستقیم از کابل اترنتی معکوس (Cross) و برای اتصال شبکه ای از کابل اترنتی مستقیم (Straight) استفاده می شود.
- گذرگاه RS485 نیز بصورت Option میتواند سفارش داده شود.

کابلهای ارتباطی

برای ارتباط کامپیوتر و PLC500 سری N، سه نوع کابل وجود دارد. دو تا از آنها برای ارتباط Ethernet و دیگری برای ارتباط سریال RS232.

- **کابل اترنت مستقیم (Staright Cable Ethernet)** هنگامی بکار می روند که کامپیوتر و PLC از طریق شبکه LAN و از راه Hub یا Switch اترنتی به یکدیگر متصل می گردند
- **کابل اترنت معکوس (Cross Cable Ethernet)** هنگامی بکار می روند که کامپیوتر و PLC مستقیماً یکدیگر متصل می گردند
- **کابل سریال RS232** هنگامی بکار می روند که کامپیوتر و PLC مستقیماً از راه پورت سریال به یکدیگر متصل می گردند
-

راه اندازی سرد و گرم Cold and Warm Restarts

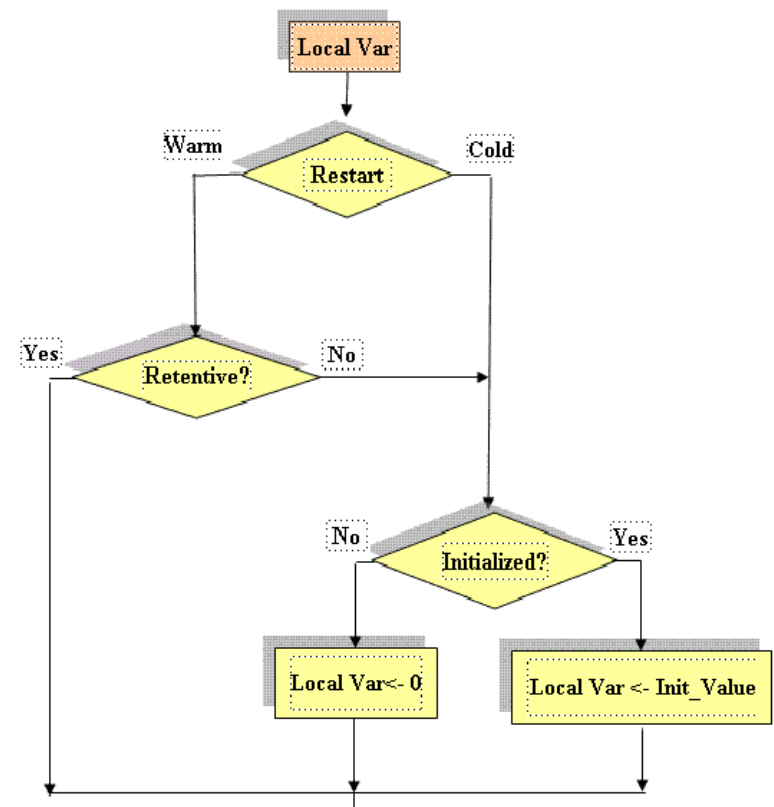
روشن کردن عادی PLC500 NSERIES راه اندازی گرم محسوب می شود. در چنین شرایطی متغیرهای پایدار (RETAIN) با آخرین مقادیر خود که در زمان خاموش کردن دستگاه داشته اند بالا می آیند. ضمناً چنانچه دستور سیستمی SPG1 در PLC وجود داشته باشند، در راه اندازی گرم اجرا می شود.

روشن کردن PLC می تواند به شکل سرد نیز انجام شود. شیوه انجام این کار در میحث بعدی که مانورهای کلید Run/Stop را توضیح میدهد روشن خواهد شد. در راه اندازی سرد، متغیرهایی که دارای مقدار اولیه باشند با مقدار اولیه پر می شوند.

باید خاطر نشان کرد که در راه اندازی سرد و گرم، چنانچه دستورهای سیستمی SPG0 و SPG1 وجود داشته باشند، آنها نیز اجرا میشوند.

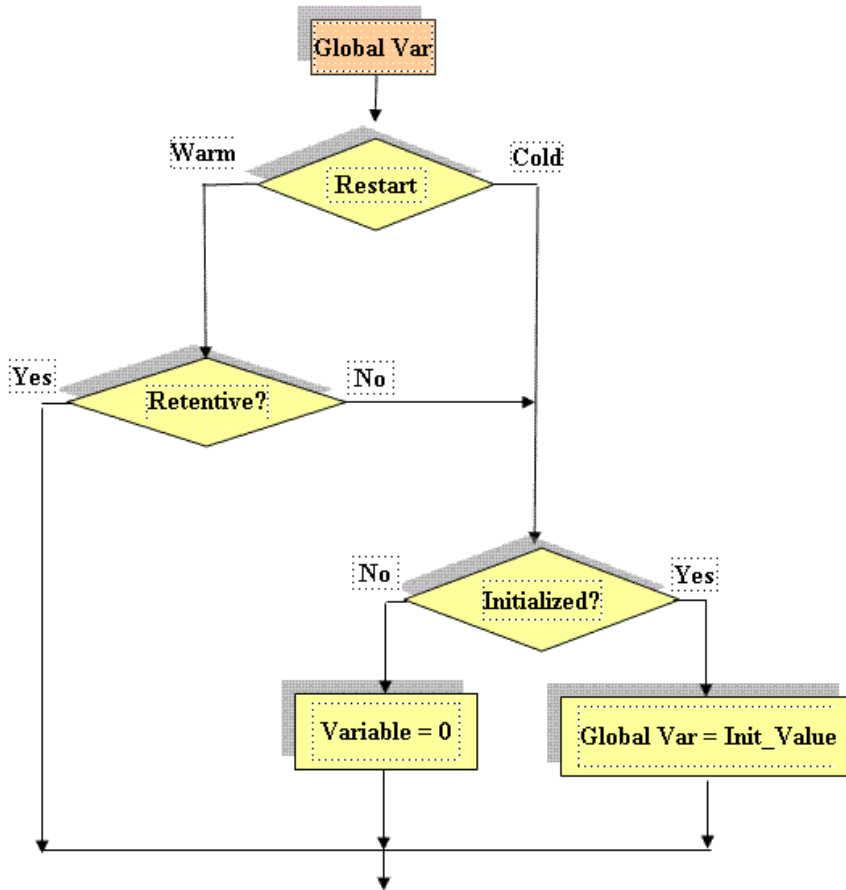
نکته ۱: خواص متغیرهای عام در لحظه روشن شدن دستگاه در صورتی اعمال می شوند که فایل متغیرهای عام (GlobalVariables) به PLC ارسال شده باشد.

نکته ۲: با توجه به انعطاف پیش بینی شده در اعلان متغیرها که میتوان برای هر یک از آنها به تنهایی خواصی چون مقدار اولیه (Initial value) و پایداری (RETAIN) تعریف کرد، دیگر نیازی به استفاده از دستورات سیستمی چون SPG0 و SPG1 بدین منظور نیست. این Task ها زمانی مورد نیاز هستند که بخواهید پارامترهای توابع پیچیده ای چون TCP_Init و INIT_POS2_00 را در زمان روشن شدن PLC به توابع و یا کارتهای مربوطه ارسال نمایید.



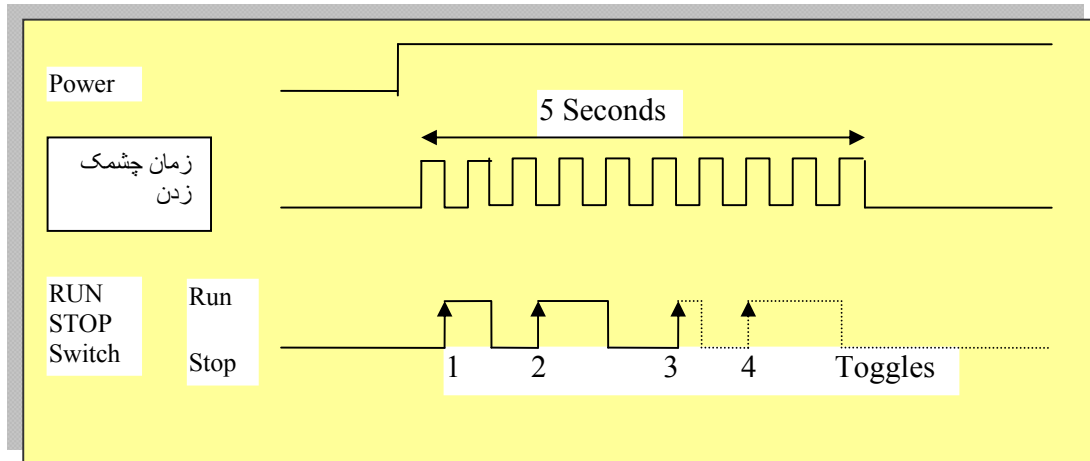
شکل روبرو روش کار با متغیرهای محلی را در راه اندازی های سرد و گرم نشان میدهد.

شکل زیر نیز چگونگی کار با متغیرهای عام در راه اندازی های سرد و گرم را نشان می دهد.



قطع و وصل کردن کلید Run/Stop در هنگام روشن کردن دستگاه Toggling Run/Stop switch in power-up

در لحظه روشن کردن PLC چنانچه کلید Run/Stop در حالت Stop باشد دو LED بالایی (سبز و زرد) شروع به چشمک زدن می کنند. این چشمک زدن به مدت ۵ ثانیه طول می کشد و در این مدت یک شمارنده تعداد دفعاتی که این کلید از حالت Stop به Run تغییر وضعیت میدهد را می شمارد.



بسته به تعداد این شمارش یکی از عملیات زیر انجام می شود.

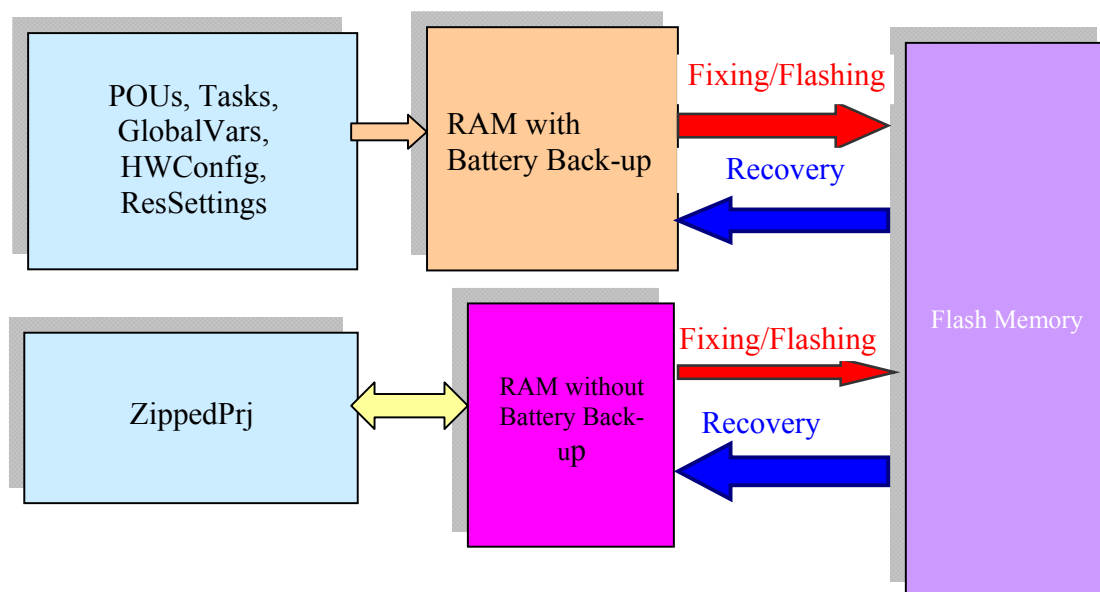
تعداد شمارش شده	عملی که انجام می شود	شرح بیشتر
0	Cold Restart راه اندازی سرد	اگر در این مدت هیچ کاری انجام ندهید و فقط صبر کنید تا چشمک زدن تمام شود راه اندازی سرد انجام می شود. کلید را به حالت Run برده و شاهد اجرای برنامه باشید.
=2*	Recovery محتویات حافظه Flash برای اجرا بر روی RAM کپی میشود	تمام POU ها، Task ها، متغیرها و سایر فایل هایی که قبلا طی عملیات Fix کردن در حافظه Flash نوشته شده اند مجددا از حافظه Flash بر روی RAM کپی شده و آماده اجرا می شوند.
>2	حافظه RAM پاک می شود	تمام POU ها، Task ها، متغیرها و فایل های مختلف از حافظه اجرایی RAM پاک گشته و آدرس اترنت و سریال به مقادیر پیش فرض Node Address = 48 و IP=192.168.0.20 باز می گردند.

احتیاط: قویا تاکید می شود که برای انجام عملیات کپی گرفتن از Flash به RAM (شمارش = ۲) باید مطمئن باشید که Flash حاوی نسخه ای معتبر از برنامه کنترلی و مناسب برای فرایند تحت کنترل است. در غیر اینصورت این کار شبیه ارسال برنامه نامناسب برای کنترل فرایند شماست. اگر از این موضوع اطلاع کافی ندارید ابتدا پروژه Zip شده را Upload کرده و آن را مطالعه نمایید.

گونه های مختلف حافظه Types of Memories

سه نوع حافظه در CPU180 برای برنامه ها و فایل ها وجود دارد. این حافظه ها عبارتند از " حافظه ی RAM با پشتیبانی باتری یا باتری-دار" ، "RAM بدون باتری" و "حافظه Flash". در حالیکه داده های حافظه RAM باتری دار با قطع برق ثابت می مانند، داده های RAM بدون باتری در چنین شرایطی خراب می شوند. حافظه نوع Flash نقش یک دیسک سخت را دارد که داده های آن را تنها با اجرای روش خاصی بنام Flashing یا Fixing می توان تغییر داد. این نوع حافظه برای نگهداری داده ها نیازی به باتری نداشته و با قطع برق داده های درون آن ثابت می مانند.

RAM با باتری می تواند داده هایش را پس از قطع برق تا ماه ها ثابت نگاه دارد. انتقال داده به حافظه ها و بین آنها در شکل های زیر شرح داده شده اند



RAM باتری دار مهم ترین حافظه ی CPU است. برنامه های کنترلی از این بخش از حافظه اجرا می شوند و به همین دلیل است که برنامه هایی که در حافظه نوع Flash قرار دارند برای اجرا باید ابتدا بر روی حافظه ی نوع RAM کپی شوند.

RAM بدون باتری واسطه انتقال فایل ZippedPrj بین PLC و کامپیوتر است. وقتی که فایل Zipped Project را به PLC می فرستید، در حافظه ی RAM بدون باتری قرار می گیرد. بنا براین با خاموش و روشن کردن PLC از بین می رود. معمولاً ارسال این فایل به PLC زمانی سودمند است که پروژه ی شما نهایی شده و قصد دارید آنها را در حافظه ی Flash تثبیت نمایید. در اینصورت همه برنامه ها را به PLC ارسال می کنیم، فایل ZippedPrj را نیز ارسال می نماییم و سپس دستور Fix را صادر می کنیم.

بنابراین می توان گفت که این حافظه برای ذخیره موقت فایل Zipped Project قبل از نوشتن در حافظه ی Flash بکار می رود. در صورتی که چنین فایلی در حافظه ی Flash وجود داشته باشد در فرایند Recovery مجدداً روی RAM بدون باتری کپی می شود. در این زمان می توان فایل زیپ شده را از PLC دریافت کرد. یادآوری می شود که Zipped Project معمولاً فایل بسیار بزرگی است که نگهداری آن روی حافظه باتری دار مقرون به صرفه نیست و به همین دلیل است که از حافظه بدون باتری استفاده شده است.

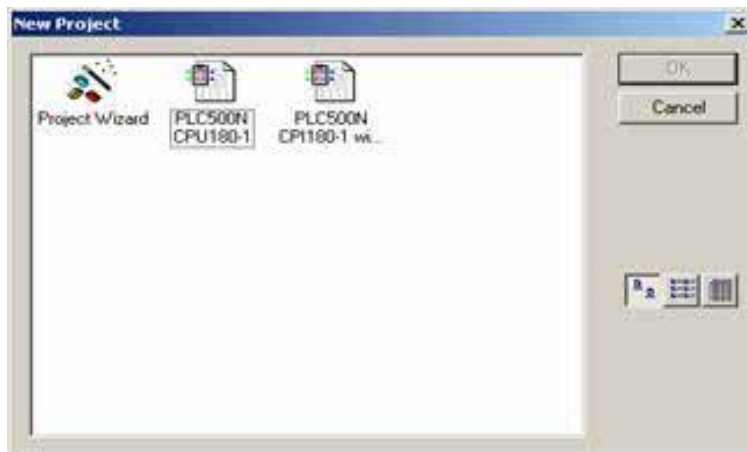
حافظه Flash محیط ذخیره دائمی داده ها و برنامه هاست. محتویات RAM توسط فرآیند Fixing/Flashing بر روی Flash نوشته می شود و در Recovery برعکس از روی Flash به RAM کپی می شود.

محیط برنامه نویسی PLC500 سری N Programming of PLC500 Nseries

برنامه نویسی PLC500 سری N شامل مراحل مختلفی است که در اینجا شرح داده می شوند. در خلال آن هر گاه نیاز به آشنایی بیشتر با ساختار و مشخصات PLC باشد، در آن موارد نیز مطالبی عنوان خواهد شد.

ساختن پروژه Creating a project

اولین گام در برنامه نویسی PLC، تشکیل پروژه است. پروژه در کامپیوتر برنامه نویسی یا کامپیوتر پروگرامر تشکیل میشود. برای تشکیل پروژه، از منوی "File" گزینه "New Project" را انتخاب کنید. پنجره زیر پیدا خواهد شد.



چند انتخاب برای ساختن یک پروژه جدید وجود دارد

- اگر CPU از نوع CPU180-1 است شما میتوانید با انتخاب الگوی "PLC500N CPU180-1" پروژه ای کوچک با مقادیر اولیه و صفحه های کار (Work sheet) پیش فرض تشکیل دهید.
- اگر CPU از نوع CPU180-1 است شما میتوانید با انتخاب الگوی "PLC500N CPU180-1 with Modbus" پروژه ای کوچک با مقادیر اولیه و صفحه های کار (Work sheet) پیش فرض و نیز صفحه ی کاری برای متغیرهای Modbus تشکیل دهید.
- اگر CPU از نوع CPU180-1G است شما میتوانید با انتخاب الگوی "PLC500N CPU180-1G" پروژه ای کوچک با مقادیر اولیه و صفحه های کار (sheet Work) پیش فرض تشکیل دهید.
- اگر CPU از نوع CPU180-1G است شما میتوانید با انتخاب الگوی "PLC500N CPU180-1G with Modbus" پروژه ای کوچک با مقادیر اولیه و صفحه های کار (Work sheet) پیش فرض و نیز صفحه ی کاری برای متغیرهای Modbus تشکیل دهید.

- یا میتوانید با استفاده از جادوگر پروژه "Project Wizard" با پاسخ دادن به سوالاتی که در چند مرحله پرسیده می شود پروژه ای تشکیل دهید.

یادآوری می شود که الگوهای مورد اشاره (Templates) ممکن است همراه با تکمیل نسخه های نرم افزار تغییر کند.

به هر حال با هر یک از روش های بالا پروژه ی کوچکی ساخته می شود که ساختار کلی آن در پنجره ی سمت چپ تصویر که همان "درخت پروژه" است مشاهده می شود.

پروژه ساخته شده شامل اجزای زیر خواهد بود.

- یک POU از نوع PROGRAM بنام Main و یا Untitled
- یک ساختار یا Configuration بر اساس PLC500 Nseries
- یک Resource یا CPU بر اساس CPU180-1 یا CPU180-1G
- یک Cyclic-Task با الویت اجرای صفر (Execution order 0) و بازه زمانی ۱۰ ms
- نسخه ای از PROGRAM به Task اضافه شده تا هر ۱۰ ms یکبار اجرا شود

توجه: راهنمای MULTIPROG بنام "MWTMAN21_001.PDF" در هنگام نصب نرم افزار و در مسیر نصب برنامه قرار خواهد گرفت. در این کتاب شما می توانید به تمام جزییات ایجاد پروژه و مراحل ویرایش POU ها به زبان های مختلف SFC, FBD, LD, IL, ST آشنا شوید.

پروژه در کامپیوتر برنامه نویسی یا کامپیوتر پروگرامر تشکیل میشود. پروژه شامل چند فایل است که باید به PLC ارسال گردند.

- برخی از فایل های پروژه برای اجرا به PLC ارسال می شوند. مثل POU ای از جنس PROGRAM و TASK
- یکی از فایلها ساختار سخت افزار PLC را معرفی می کند (HwConfig).
- یکی از فایلها پارامتر هایی را در سمت PLC تنظیم می کند
- و چند فایل دیگر که کارکردهای ویژه دارند و بتدریج تشریح خواهند شد

یک پروژه ی حداقلی دارای اجزای زیر است.

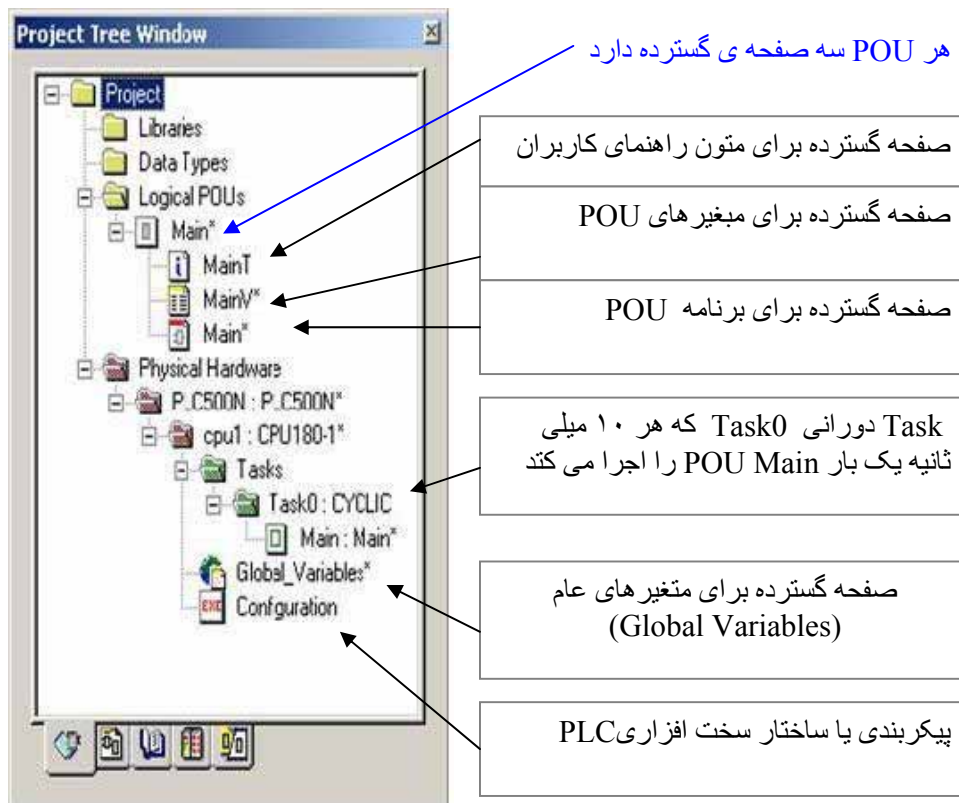
- یک POU از نوع PROGRAM
- یک دستورکار دورانی یا CYCLIC TASK با اولویت اجرای صفر که PROGRAM را اجرا کند
- ساختار سخت افزار PLC یا Hardware Configuration

با ارسال این فایل ها به PLC برنامه ی کنترلی نوشته شده در PROGRAM بصورت سیکلیک یا دورانی و پی در پی اجرا می شود.

به تدریج میتوان این ساختار را گسترش داده و انعطاف بیشتری در برنامه PLC ایجاد کرد.

درخت پروژه Project tree

معمولاً در سمت چپ تصویر MWT پنجره درخت پروژه "Project Tree Windows" دیده میشود. چنانچه این پنجره را مشاهده نمیکنید به منوی View رفته "Windows Project Tree" را انتخاب کنید.



چند نکته در مورد PLC500 N SERIES

با توجه به اینکه شناختن برخی از مشخصات PLC500 سری N در درک بهتر مباحث بعدی کمک می کند، در اینجا به شرح چند نکته از چگونگی کارکرد watchdog و برخی از نواحی حافظه ی PLC می پردازیم.

مکانیزم عملکرد Watchdog در CPU180

سیستم عامل بر اجرای دستورهای کاری (Tasks) نظارت می کند تا مطمئن شود که آنها در لحظه ی مناسب و بطور کامل اجرا می شوند. در سیستم عاملهای Multi-task معمولی اجرای کامل برنامه در بازه ی زمانی خاص (time-slot) چندان مهم نیست زیرا ادامه کار می تواند در نوبت بعدی فعال شدن Task انجام شود.

در سیستمهای کنترلی آنی (Real-time) صنعتی که Taskها بیکدیگر وابسته اند و هر یک بخشی از یک فرایند واحد را پردازش می کند، پیوستگی و سازگاری داده ها (Integrity and Consistency Data) اهمیت خطیری دارد.

از سوی دیگر برنامه هایی که ساختار بدی داشته باشند و دارای حلقه های بزرگ و حتی بی پایان باشند، میتوانند CPU را برای مدتی طولانی و یا برای همیشه وقف اجرای بخش کوچکی از برنامه کرده و حوادث بزرگ و مخربی را باعث شوند. به همین دلیل سیستم عامل PLC باید اطمینان دهد که در بازه ی زمانی اختصاص یافته برای هر Task فعالیتهای تعریف شده برای آن Task بطور کامل اجرا میشود.

PLC500 Nseries برای جلوگیری از وقوع چنین شرایطی از مکانیزم Watchdog یا سگ نگهبان استفاده می کند. اساس کار بدین شکل است که هر Task ای باید بتواند در بازه زمانی اختصاص داده شده به آن تمام PROG های وابسته به خود را از شروع تا پایان بطور کامل اجرا کند. بدین معنی که اگر Task ای در سیکل گذشته ی خود وظایفش را تکمیل نکرده باشد، دیگر مجالی برای تکمیل کار خود در سیکل بعدی نخواهد داشت. در صورت بروز چنین مواردی سیستم عامل، CPU را به STOP برده تا امنیت فرایند را تضمین نماید. همراه با این کار خطای Watchdog را نیز در حافظه CPU ثبت می نماید.

برنامه ی خوب و ساختار یافته برنامه ای است که متغیرهای سریع فرایند (Inputs and Outputs) را در Task های سریع (بازه زمانی کوچک) و بخشهایی از فرایند که نیازی به پردازش سریع ندارند را در Task هایی با بازه زمانی بزرگتر پردازش نمایند. معمولاً ورودی ها و خروجی های دیجیتالی باید در Task های سریع و حلقه های کنترلی آنالوگ که اغلب دارای محاسبات طولانی ریاضی هستند در Task های کندتر قرار گیرند.

این نکته را نیز باید در نظر داشت که گرچه می توان فعالیتها را بین Task هایی با بازه های زمانی مختلف تقسیم کرد، لیکن مواردی پیش می آید که تمام کارها باید در یک بازه زمانی مشخص انجام شوند و راه دیگری وجود ندارد. در چنین مواردی کوچکترین بازه زمانی باید آنقدر بزرگ باشد تا مجال اجرای تمام کارها امکان پذیر باشد در غیر این صورت خطای Watchdog گریز ناپذیر خواهد بود.

تصاویر فرآیند در حافظه PLC500 NSERIES Process Images in PLC memory

ورودیها (Inputs) ، خروجی ها (Outputs) و متغیر های عام (Global Variables) در فضاهای ثابت و اختصاصی از حافظه PLC که به فضای تصاویر فرآیند (Process Image) شهرت دارند نگهداری میشوند. شکل زیر این نواحی را نشان می دهد.

I=
Inputs

بخش I از حافظه، تصویری از ورودی های فیزیکی را در خود نگاه میدارد. PLC ورودی ها را از کارت های ورودی میخواند و مقادیر آنها را در این بخش از حافظه قرار میدهد.

Q=
Outputs

بخش Q از حافظه، تصویری از خروجی های فیزیکی را در خود نگاه میدارد. PLC مقادیر خروجی ها را بر اساس برنامه کاری محاسبه کرده و نتایج آن را در این بخش از حافظه قرار میدهد. سپس در زمان مناسب این مقادیر را به کارت های خروجی ارسال مینماید.

M=
Memory

بخش M از حافظه، متغیرهای عامی از فرآیند را در خود نگاه میدارد که آدرس آنها را برنامه نویس با مسئولیت خود مشخص کرده است. این متغیر ها را Variables Located Global و یا به اختصار Located Var مینامند.

S=
Memory

بخش S از حافظه، متغیرهای عامی از فرآیند را در خود نگاه میدارد که آدرس آنها را برنامه نویس مشخص نکرده و این مسئولیت را به عهده سیستم قرار داده است. این متغیر ها را Unlocated Global Variables و یا به اختصار Unlocated Var مینامند.

M1=
Memory

بخش M1 از حافظه بیشتر برای ارتباطات Modbus ساخته شده است. متغیرهای M1 نیز عام هستند و می توان از آنها مانند سایر متغیرهای عام استفاده کرد. لیکن چنانچه تجهیزاتی که با زبان Modbus با CPU تماس بگیرند برخی از متغیرهای Modbus مثل Coils و Holding registers با این ناحیه از حافظه PLC در ارتباط خواهند بود.

نحوه آدرس دهی نواحی تصاویر فرایند

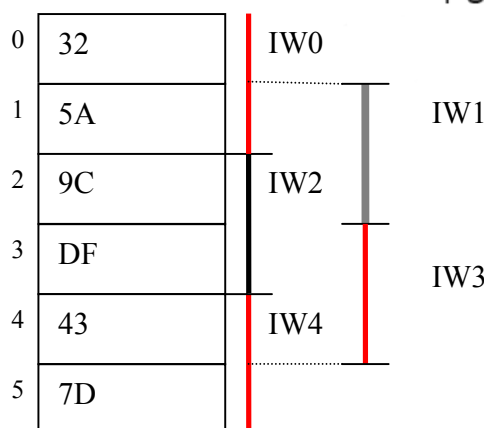
در اغلب موارد برنامه نویسی نباید نگران محل استقرار متغیرها در حافظه باشد زیرا اختصاص حافظه برای متغیرها می تواند کاملا اتوماتیک انجام شود. تنها استثنا در این مورد ورودی ها و خروجی های PLC هستند. ورودی ها و خروجی های PLC (I/O ها) بنا بر ماهیت سخت افزاری خود الزاما در آدرسهای مشخصی مستقر می شوند و لذا مدیریت آدرس دهی آنها را برنامه نویسی باید به عهده بگیرد. در مواردی نیز کاربر بدلایلی مایل است خود بخشی از ساماندهی حافظه را به عهده بگیرد. به همین دلایل به اختصار نحوه آدرس دهی نواحی تصاویر فرایند در حافظه را شرح می دهیم.

همانطور که میدانید در موقع اعلان متغیرهای مستقر (Located Variables) باید اندازه و محل استقرار آنها را مشخص کنیم. اشکال مختلف آدرس دهی بشکل زیر است.

<i>Variable</i>	<i>description</i>
%IX4.5	Input Bit 5 of byte 4
%QX6.2	Output Bit 2 of byte 6
%Q6.2	Output Bit 2 of byte 6
%MB56	Memory Byte 56
%QW32	Output Word 32 (Bytes 33, 32)
%ID73	Input Double 73 (Bytes 76, 75, 74, 73)

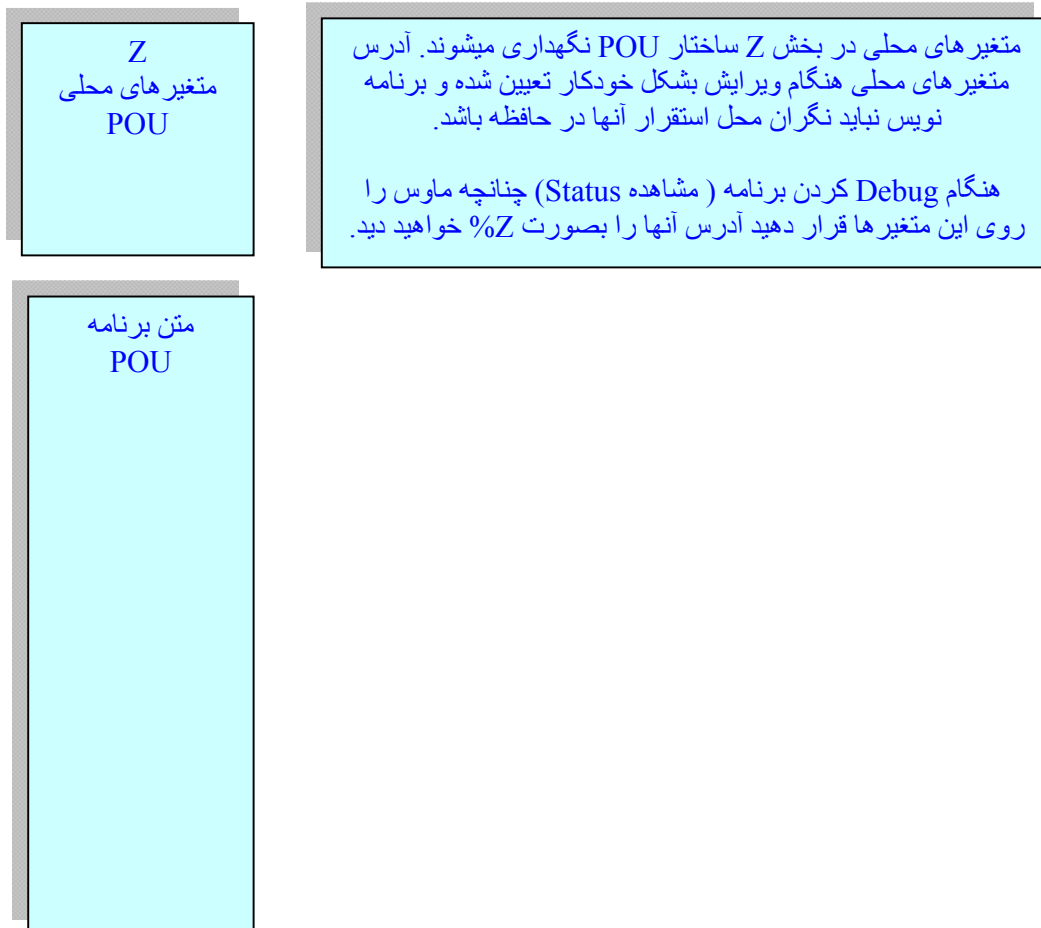
لطفا توجه کنید که آدرس متغیرهای چند بایتی آدرس کمترین بایت آن است (LSB). همچنین توجه داشته باشید که ممکن است هنگام آدرس دهی متغیرهای چند بایتی هم پوشانی در آدرسها ایجاد شود. شکل زیر پدیده ی هم پوشانی را نشان می دهد. مقادیر بعضی از متغیرها عبارت خواهند بود از:

IB0 = 32
IW0 = 5A32
IW1 = 9C5A
ID2 = 7D43DF9C



ساختار POU در PLC500 NSERIES

در CPU180 در هر POU فضایی از حافظه برای متغیرهای محلی (Local Variables) اختصاص یافته است. این بخش از حافظه هنگام ویرایش POU بطور خودکار و همراه با متن برنامه (Source Code) ایجاد میشود. هنگامیکه POU به PLC ارسال میشود متغیرهای محلی با مقادیر اولیه پر میشوند. (Initialization) یادآوری می شود که متغیرهای محلی در محدوده POU مربوطه اعتبار دارند.



متغیرهای محلی POU هم مثل متغیرهای عام می توانند دارای مقدار اولیه باشند و خاصیت پایداری یا RETAIN داشته باشند. خواص پایداری یا مقدار اولیه در مورد این متغیرها در راه اندازی های سرد و گرم درست به همان شکلی عمل می کنند که در مورد متغیرهای عام عمل میشود.

وظایف یا دستورهای کاری PLC Tasks

از میان POU هایی که مینویسید، آنهایی که بصورت Program باشند می توانند برای اجرا به CPU معرفی شوند. این برنامه ها پس از ارسال به CPU بخودی خود اجرا نمی شوند زیرا زمان و نحوه ی اجرای برنامه ها هنوز مشخص نیست. آیا اجرای این POU را بصورت پی در پی می خواهیم یا تنها یکبار در زمان روشن کردن دستگاه و یا با وقوع رخداد خاصی؟ اگر اجرای آن را پی در پی می خواهیم با چه فاصله ی زمانی؟ پاسخ این سوالات را چیزی بنام دستور کار یا Task می دهد.

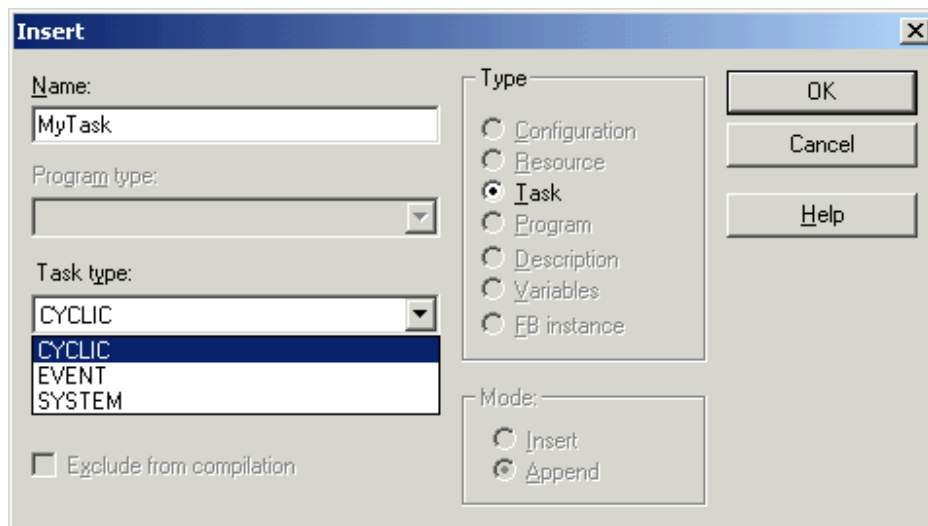
معرفی PROGRAM ها به CPU برای اجرا، از طریق نسبت دادن آنها به دستور کار یا Task انجام می شود. در حقیقت زمان و چگونگی اجرای هر Program بستگی به زمان و چگونگی اجرای Task ای دارد که به آن وابسته است.

در استاندارد IEC61131-3 سه نوع دستور کار یا Task در نظر گرفته است:

- دستور کارهای های دورانی (Tasks Cyclic) در فواصل زمانی معینی بصورت پریودیک فعال می شوند.
- دستور کارهای های سیستمی (Tasks System) در شرایط خاصی بسته به کارکرد CPU فعال میشوند. مثلا در استارت های سرد و گرم و ...
- دستور کارهای های رخدادی (Tasks Event) در صورت وقوع رخدادهای خاص (Interrupt Events) فعال می شوند.

چگونه Task به پروژه اضافه می شود؟

در درخت پروژه بر روی گره ی Tasks کلیک راست کرده و Insert را انتخاب کنید. پنجره ی زیر باز می شود.

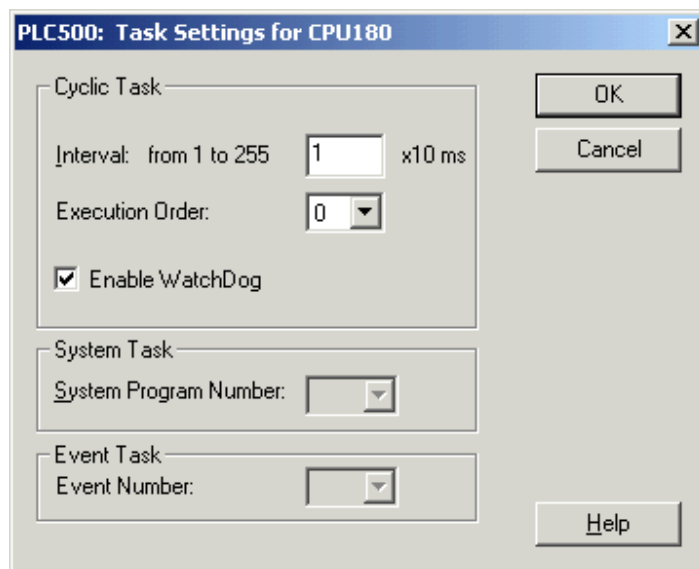


همانطور که دیده می شود، سه نوع Task میتوان انتخاب کرد. این سه نوع عبارتند از Cyclic ، Event و System. در ادامه این سه نوع دستور کار یا Task تشریح می شوند.

دستورکار دورانی Cyclic Task

در پنجره ی وارد کردن Task – شکل صفحه قبل- پس از تعیین نام برای Task نوع دستور کار را CYCLIC انتخاب کنید و کلید OK را بزنید.

با زدن کلید OK پنجره ی جدیدی باز می شود که پارامترهای مربوطه را متناسب با نوع Task وارد نمایید. در پنجره جدید تنها بخشهایی که مناسب برای نوع Task باشد فعال خواهد بود.



در این پنجره پارامترهای Task را وارد کرده و کلید OK را بزنید.

- مهمترین Task ها از نوع Cyclic هستند. برای این Task ها دو پارامتر وجود دارد.
 - **Interval**: این پارامتر فواصل زمانی اجرای Task را تعیین میکند. هر عددی که در این محل قرار داده شود با ضریب ۱۰ فواصل اجرای آن را بر حسب میلی ثانیه محاسبه می کند. مثلا اگر عدد ۳ وارد شود، این Task در فواصل زمانی ۳۰ میلی ثانیه یکبار اجرا می شود.
 - **Execution Order**: این پارامتر اولویت اجرای Task را تعیین می کند. عدد صفر بالاترین اولویت و عدد ۹ کمترین اولویت را داراست.
 - **Enable Watchdog**: با فعال کردن این گزینه، چنانچه به هر دلیلی CPU نتواند برنامه های مربوط به Task را در فواصل زمانی تنظیم شده اجرای کامل نماید، CPU از حالت Run به Stop می رود.

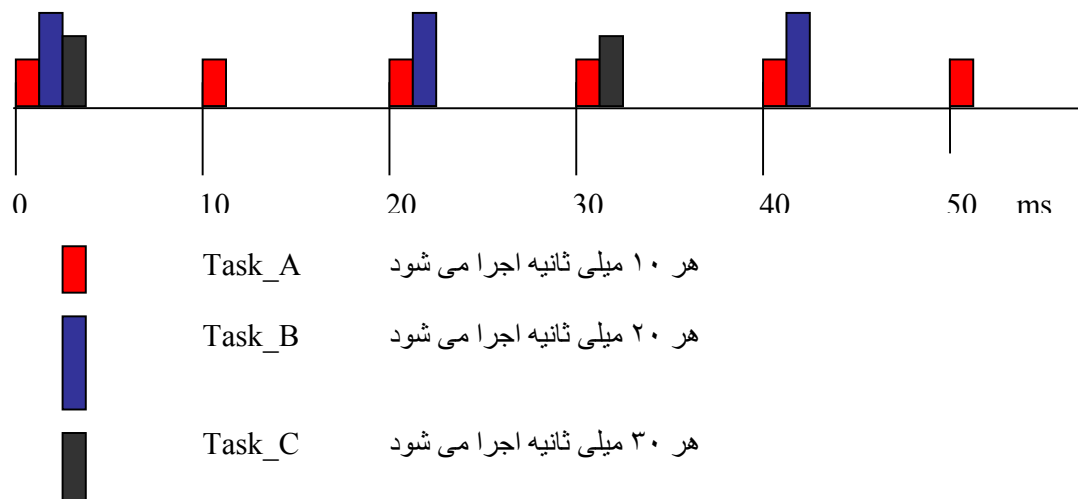
نکته ۱: در مراحل کار PLC، مقاطع زمانی خاصی پیش می آید که نوبت اجرای چند Task با هم فرا می رسد. در چنین حالتی آن Task ای اولویت دارد و زودتر اجرا می شود که پارامتر Execution Order آن کوچکتر است.

نکته ۲: گزینه ی Enable Watchdog را همواره فعال نکه دارید مگر در مواردی که می خواهید موقتا آزمایشی انجام دهید که خطای مربوطه مشکل ساز است. در این گونه موارد بخاطر داشته باشید که پس از آزمایشات خود حتما آن را به حالت فعال بازگردانید.

دستورات کاری دورانی در CPU180 Cyclic Task in CPU180

CPU180 می تواند تا ۱۰ Task دورانی داشته باشد. هر Task دورانی به دو پارامتر نیاز دارد. با این دو پارامتر فاصله زمانی اجرا و اولویت اجرای Task مشخص می شود. فاصله زمانی اجرا توسط پارامتر Interval Time مشخص شده که می توان عددی بین ۱ تا ۲۵۵ را به آن نسبت داد. در نتیجه فاصله زمانی اجرای ۱۰ تا ۲۵۵۰ میلی ثانیه برای آن در نظر گرفته می شود. هنگامی که در یک مقطع زمانی نوبت اجرای چند Task بطور هم زمان فرا برسد، پارامتر اولویت اجرای آنهاست که نوبت را معین میکند. این اولویت توسط پارامتری بنام *execution order* تعیین میشود. اولویت اجرا برای این ده Task دورانی را می توان بین اعداد ۰ تا ۹ انتخاب کرد. Task های دورانی با عدد کوچکتر برای اجرا مقدم ترند.

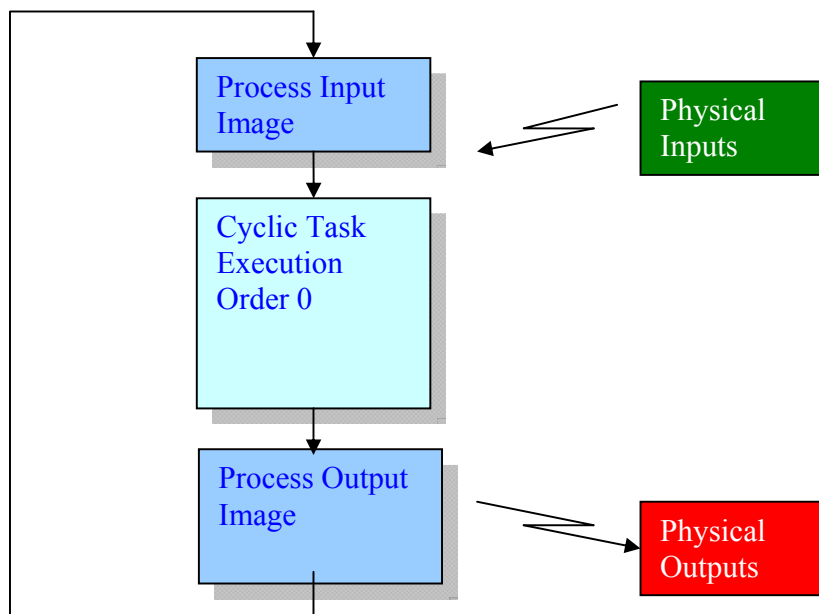
فرض کنید که سه Task دورانی به اسامی Task_A ، Task_C ، Task_B با فواصل زمانی اجرای ۱۰ و ۲۰ و ۳۰ میلی ثانیه با اولویت های اجرای ۰ و ۱ و ۲ داریم. شکل دیاگرام زمانی زیر نحوه اجرای آنها را نشان می دهد.



لطفا توجه کنید که Task دورانی اصلی Task ای است که پارامتر order execution آن ۰ باشد. هر CPU حداقل نیاز به یک Task با اولویت اجرای ۰ دارد زیرا اصلی ترین فعالیتهای CPU مثل خواندن ورودی ها و نوشتن در خروجی های PLC با اجرای این Task همزمان گردیده اند.

Task دورانی با اولویت اجرای صفر (Order 0 Execution) دارای این ویژگی خاص است که خواندن ورودی ها و نوشتن خروجی ها قبل و بعد از اجرای آن صورت میگیرد.

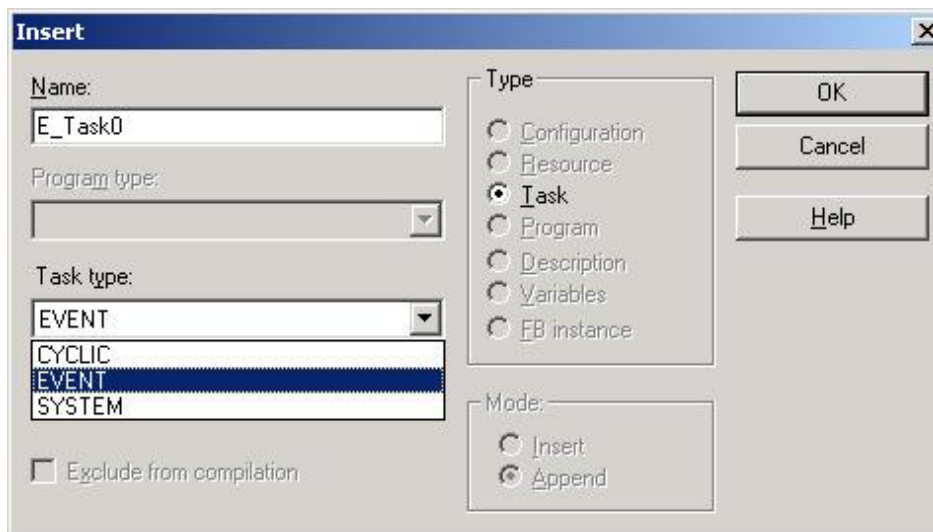
لطفا به شکل زیر توجه کنید.



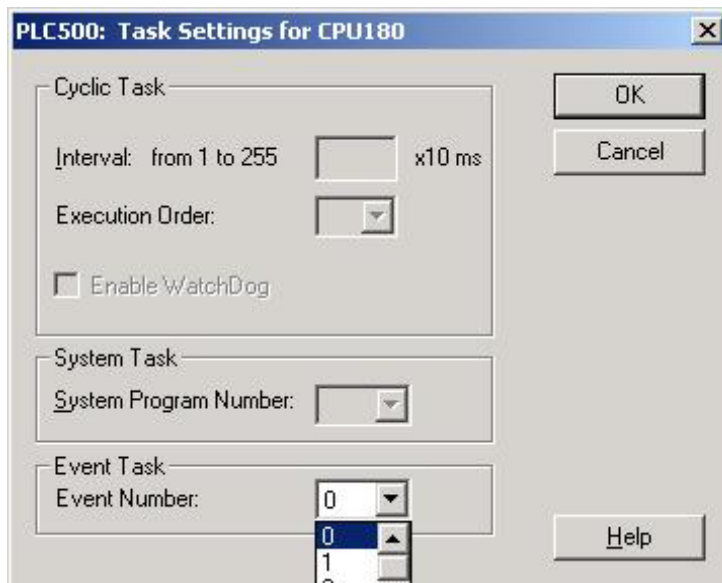
خواندن مقادیر ورودی (I) از ورودی های فیزیکی PLC قبل از اجرای Task با اولویت اجرای صفر انجام میشود.
 بشکل مشابهی نوشتن مقادیر خروجی (Q) بر روی خروجی های فیزیکی PLC پس از اجرای Task با اولویت اجرای صفر انجام میشود.

دستورکار رخدادی یا وقفه ای Event Task

در پنجره ی وارد کردن Task پس از تعیین نام برای Task نوع دستور کار را EVENT انتخاب کنید و کلید OK را بزنید.



با زدن کلید OK پنجره ی تنظیمات Task پدیدار میشود و تنها شماره ی Event را می پرسد.



Event Task ها زمانی اجرا می شوند که اتفاقی افتاده باشد که نیاز به سرویسی سریع و خارج از نوبت داشته باشد. مثلا کارت واکنش سریع FR24-00 دارای ۸ ورودی و ۱۶ خروجی دیجیتالی است. در صورتی که هر

یک از ورودی های از صفر به یک تغییر کند، بلافاصله 0 Event task را فعال می کند. شما می توانید کارهای مناسب برای چنین رخدادهایی را در POU هایی نوشته و آن را به این Task نسبت دهید.

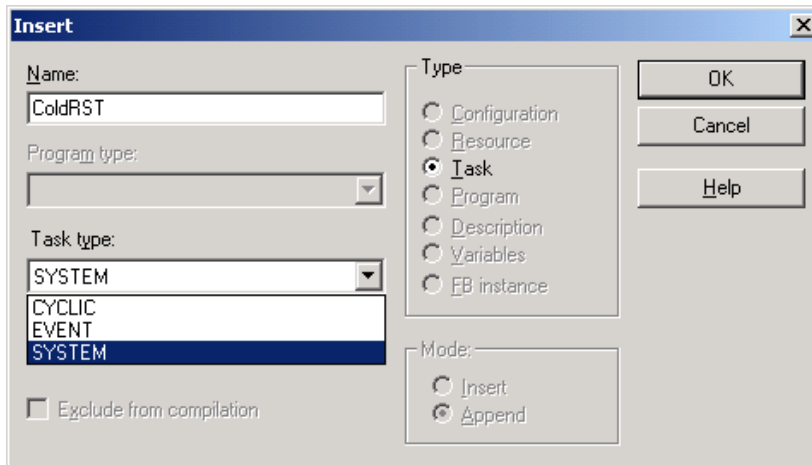
توجه داشته باشید، گرچه برای CPU180 شش Event Task پیش بینی شده است، اما تاکنون تنها یکی از آنها فعال شده است. بقیه آنها برای گسترش های آینده رزرو شده اند. کاربرد سایر Event Task ها در آینده مشخص خواهد شد.

EVENT TASK0	فعال است
EVENT TASK1	رزرو شده است
EVENT TASK2	رزرو شده است
EVENT TASK3	رزرو شده است
EVENT TASK4	رزرو شده است
EVENT TASK5	رزرو شده است

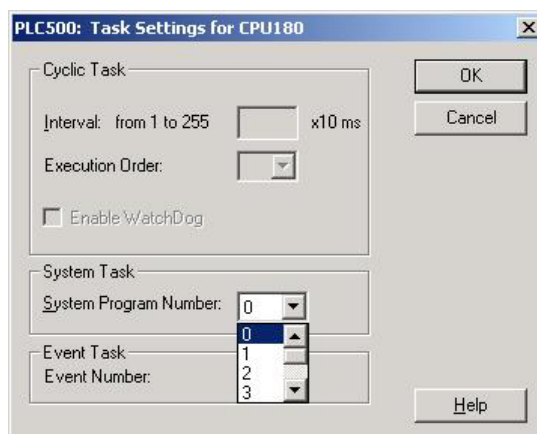
دستورکار سیستمی System Task

دستورات سیستمی توسط سیستم عامل در شرایط خاص و بشکل خودکار برای اجرا فرا خوانده می شوند. این شرایط شامل تغییر وضعیت کاری PLC یا بروز خطا و نظایر آن است. به این Task ها System Program یا SPG نیز گفته می شود.

در پنجره ی وارد کردن Task پس از تعیین نام برای Task نوع دستور کار را SYSTEM انتخاب کنید و کلید OK را بزنید.



بازدن کلید OK پنجره ی تنظیم های Task باز میشود و تنها شماره ی System Program Number را می پرسد.



CPU180-1 می تواند تا شش دستور کار سیستمی (System Task) داشته باشد. از میان این Task ها شماره های ۰، ۱، ۲، و ۳ در حال حاضر برای کارهای خاص اختصاص یافته و بقیه برای گسترش احتمالی رزرو شده اند.

- SYSTEM TASK0 (SPG0):** یک بار در راه اندازی سرد اجرا می شود
 - SYSTEM TASK0 (SPG1):** یک بار در راه اندازی گرم اجرا می شود
 - SYSTEM TASK2 (SPG2):** هنگامی که PLC به Stop رفته باشد، این Task در صورت وجود بجای Cyclic Task0 اجرا می شود.
 - SYSTEM TASK3 (SPG3):** هنگامی که یکی از کارت های PLC از راک خارج شود و یا خراب شود اجرا می شود.
- سایر SYSTEM TASK ها برای گسترش در آینده پیش بینی شده اند.

ایجاد و ویرایش برنامه PLC

در پنجره ی درخت پروژه و دربخش Logical POU's کلیک راست کنید. نماد Insert به معنی گذاردن ظاهر می شود. به منظور پذیرش، بر روی آن کلیک چپ کنید. پنجره ی زیر باز می شود.

در این پنجره چند پارامتر را باید مشخص کنید.

۱. نام POU که می تواند تا ۲۴ حرف شامل حروف و اعداد باشد و با حرف شروع شود.

۲. نوع یا Type بلوک POU را از بین گزینه های Program، Function یا Function Block انتخاب کنید.

۳. زبان برنامه نویسی را از بین پنج زبان مختلف انتخاب کنید.

۴. کلید OK را بزنید.

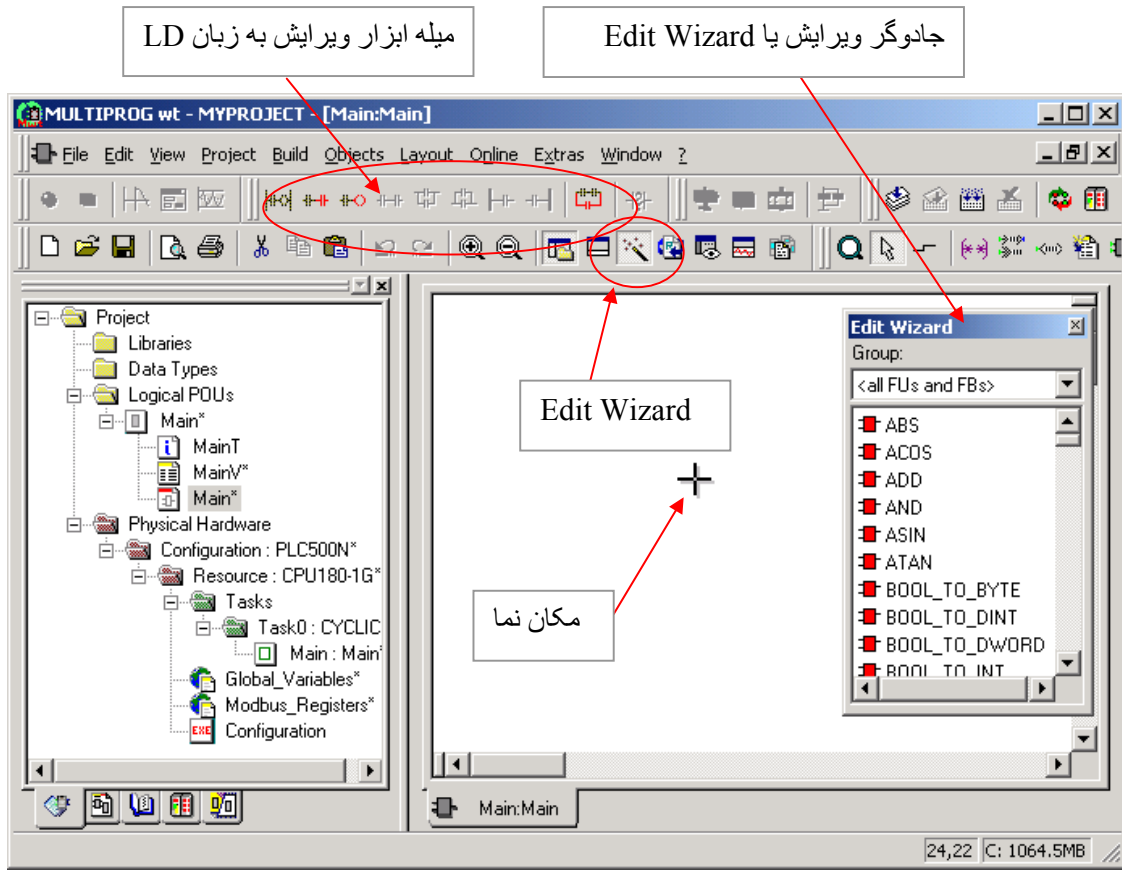
مطالب پیشرفته:

- انتخاب Use Reserve در خانواده PLC500N و در انواع CPU های آن نقشی ندارد.
- در برنامه های ساده گزینه های PLC type و Processor type را معمولاً تغییری نمی دهیم. عبارت غیر وابسته یا <independent> مشخص می سازد که هدف برنامه نویس ایجاد POU ایست که در آن تنها از توابع استاندارد IEC1131-3 استفاده خواهد شد.
- در خانواده ی بزرگ PLC500 سری N، برخی توابع اضافه بر استاندارد نیز وجود دارند که میتوان از آنها استفاده کرد. در صورتیکه چنین هدفی دارید از گزینه PLC type، نوع آن را PLC500N انتخاب کنید.
- هر یک از CPU های خانواده ی PLC500N دارای تواناییهای متفاوتی است. در نتیجه برای هر یک، مجموعه ای از توابع خاص وجود دارد که با انتخاب CPU ی مورد نظر در گزینه ی Processor type به آنها دسترسی خواهید داشت.

با زدن کلید OK در زیر مجموعه Logical POU's، نام برنامه ی وارد شده اضافه می شود. هر POU دارای ۳ صفحه ی کار (Worksheet) خواهد بود. مثلاً اگر نام برنامه را Main نامیده اید، نام سه صفحه ی کار بترتیب عبارت خواهند بود از:

- MainT
 - MainV
 - Main
- این صفحه ی کار برای نوشتن متون راهنمای شماست و نقشی در نحوه ی کار ندارد. این صفحه ی کار برای نگهداری متغیر های برنامه ای است که شما می نویسید. صفحه ی اصلی برنامه است که منطق کاری و برنامه کنترلی را در آن می نویسید

بر روی نماد صفحه ی کار Main کلیک دابل کنید. صفحه ی کار مربوطه باز می شود.

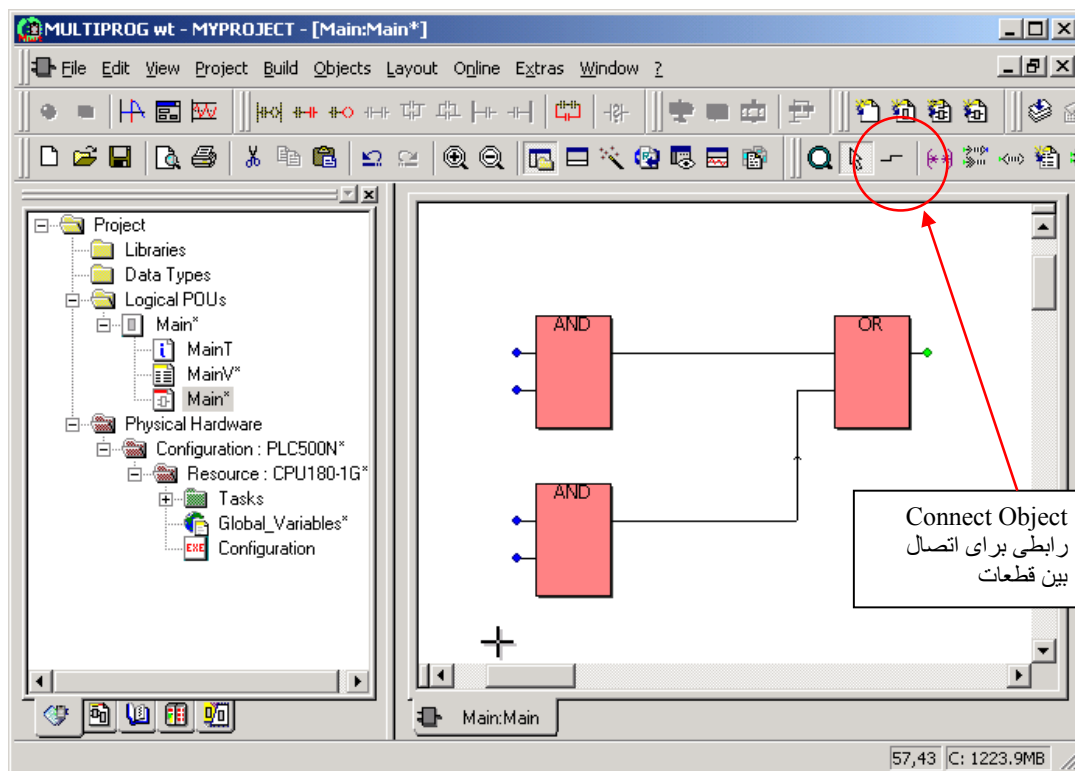


برای ویرایش برنامه، افزون بر در اختیار داشتن میله ی ابزارها در بالای صفحه، جادوگر ویرایش (Edit Wizard) هم در اختیار شماست. چنانچه آن را در صفحه نمی بینید، از منوی View، گزینه ی Edit Wizard را فعال کنید.

در محلی از صفحه ی کار کلیک کنید. علامت بزرگی بشکل + ظاهر شده و پنجره Edit Wizard روشن می شود. هر گاه این پنجره تاریک بود کافیست که در محلی از صفحه ی کار کلیک کنید. برای قرار دادن هر یک از توابع موجود از لیست Edit Wizard، کافی است روی آن کلیک دابل نمایید. تابع مورد اشاره در محل + نصب خواهد شد.

تابعی مثل AND را روی صفحه قرار دهید. در صورت لزوم ماوس را روی آن گذاشته و همراه با فشرده نگهداشتن کلید چپ ماوس آن را در صفحه جابجا کنید. با قراردادن ماوس روی تابع و یک کلیک ساده، رنگ آن خاکستری میشود. در این حالت میگویند که این شیء انتخاب یا Mark شده است. روی هر یک از ورودی ها و خروجی های تابع هم می توان کلیک ساده و در نتیجه آن را مارک کرد.

خروجی تابع را مارک کنید و از Edit Wizard، تابع OR را کلیک دابل کنید. خواهید دید که تابع OR دیگری به آن متصل خواهد شد. به همین ترتیب شکل زیر را کامل کنید.



برای اتصال خروجی هر تابع به ورودی تابع دیگر از رابطی که به همین منظور وجود دارد استفاده کنید. سیم رابط بین دو تابع را مارک کرده و کلید delete را بزنید. برای متصل کردن مجدد این دو تابع می توانید مجدداً از ابزار Connect Objects که در شکل بالا می بینید استفاده کنید. اگر دیگر به این وسیله نیازی ندارید، کلید Esc را بزنید و یا در جایی در صفحه کار کلیک دوبل کنید.

هنگام ویرایش برنامه گاهی لازم است قطعات را بر روی صفحه ی کار جابجا کنید. اگر تنها یک قطعه را جابجا می کنید، تنها آن را مارک کرده و با نگه داشتن کلید چپ ماوس آن را جابجا کنید. اگر گروهی از قطعات و متغیرها را می خواهید جابجا کنید، ابتدا کلید چپ ماوس را فشرده و فشرده نگهدارید. در همین حال ماوس را روی صفحه حرکت دهید تا قطعات موجود در منطقه ای مارک شود. حالا کلید چپ ماوس را روی یکی از اجزای مارک شده قرار دهید و بلوک مارک شده را در صفحه جابجا کنید. قطعه گذاری اتوماتیک روی صفحه ی کار گاهی با پیام خطای برخورد قطعات (Collision) همراه است. در این موارد باید بشکل دستی برای نصب قطعات فضا ایجاد کنید.

با استفاده از میله ابزار LD می توانید به زبان ladder هم برنامه بنویسید. برنامه نویسی بصورت ترکیبی از زبانهای LD و FBD همواره امکانپذیر است.

با توجه به گستردگی و حجم زیاد مطالب، آموزش زبان های برنامه نویسی مورد نظر این نوشتار نیست لیکن در اغلب موارد استفاده از Help نرم افزار مفید است.

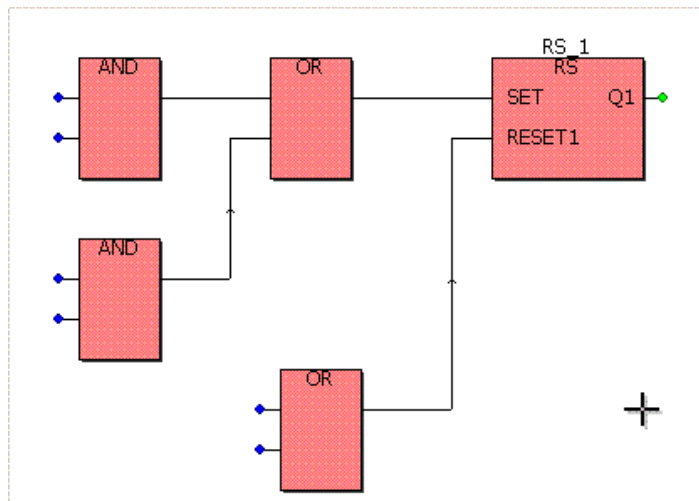
برای دسترسی کامل به مطالب مربوطه و ویرایش کامل برنامه لطفاً به کتاب راهنمای **MULTIPROG** مراجعه کنید.

برنامه و متغیر های آن را کامل کنیم تاکنون چند تابع را بر روی صفحه کار قرار داده ایم. در ادامه باید برنامه را کمی تکمیل تر کرده و ورودی و خروجی های این توابع را مشخص کنیم. در مثال قبل فرض کنید که میخواهیم موتور (Motor_1) را با فرمان Start در دو حالت بهره برداری دستی (HAND) و اتوماتیک (AUTO) روشن کنیم و با ایجاد خطای Fault و یا زدن کلید Stop آن را خاموش کنیم.

ورودی ها و خروجی های PLC در آدرسهای زیر قرار دارند.

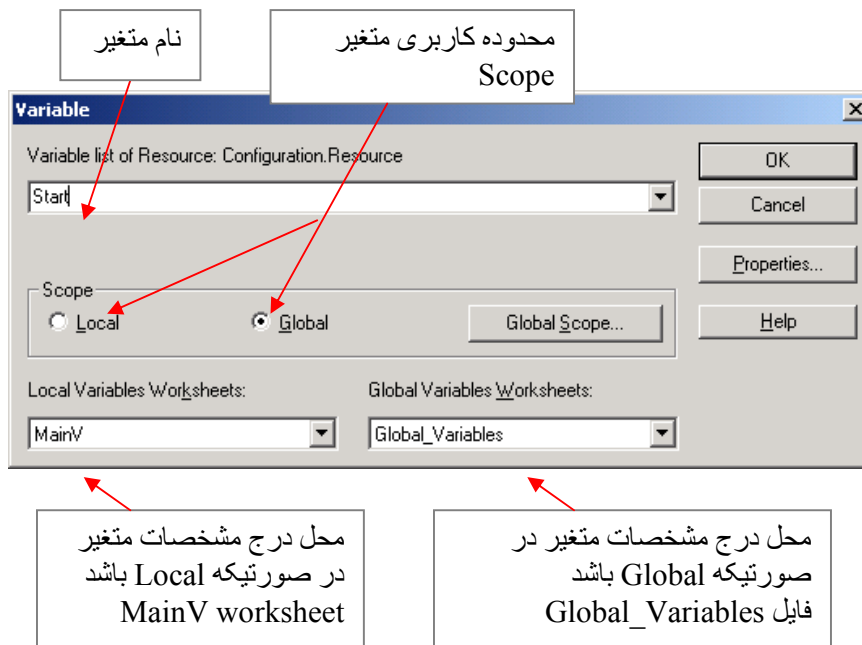
نام متغیر Variable Name	نوع داده Data type	محدوده کاربری Scope	آدرس Address
Start	BOOL	عام یا Global	I0.0
Stop	BOOL	عام یا Global	I0.1
HAND	BOOL	عام یا Global	I0.2
AUTO	BOOL	عام یا Global	I0.3
Fault	BOOL	محلی یا Local	-
Motor 1	BOOL	عام یا Global	Q0.0

همانطور که دیده می شود بجز سیگنال Fault بقیه سیگنالها از طریق I/O های PLC اعمال می شوند. برنامه را مطابق شکل زیر تکمیل نمایید. ملاحظه می کنید که از یک FB نیز در برنامه استفاده کرده ایم. این بلوک تابع RS نام دارد که نوعی Flip-flop است.

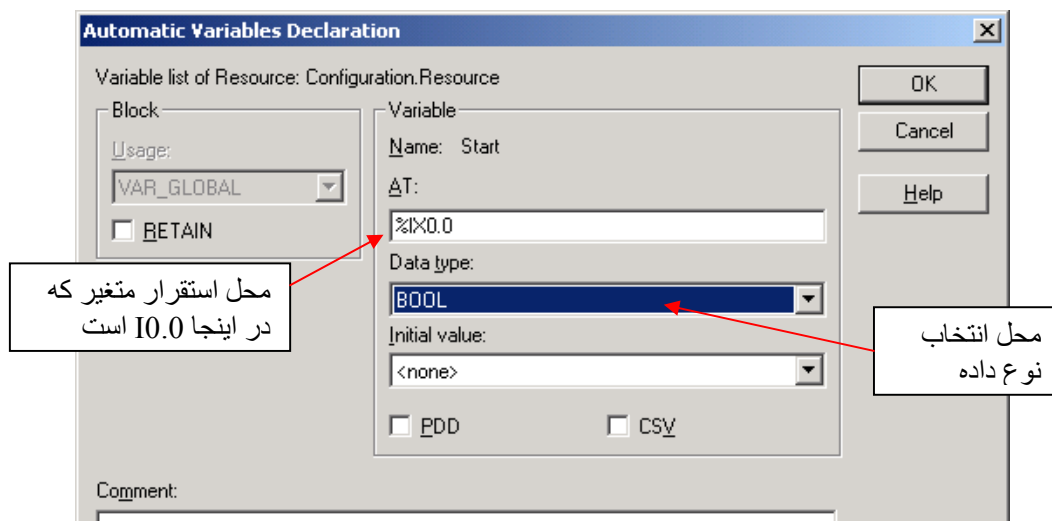


برای توضیح بیشتر در مورد کار بلوک تابع فلیپ فلاپ نوع RS، در پنجره ی Edit Wizard، تابع مورد نظر را مارک کنید و روی ماوس کلیک راست نمایید. در پنجره ی کوچکی که ظاهر می شود گزینه Help on FB/FU را انتخاب کنید.

حالا باید متغیرها را به برنامه نسبت داد. برای این کار روی اولین ورودی تابع AND کلیک دبل کنید. پنجره زیر برای معرفی متغیر ظاهر می شود.

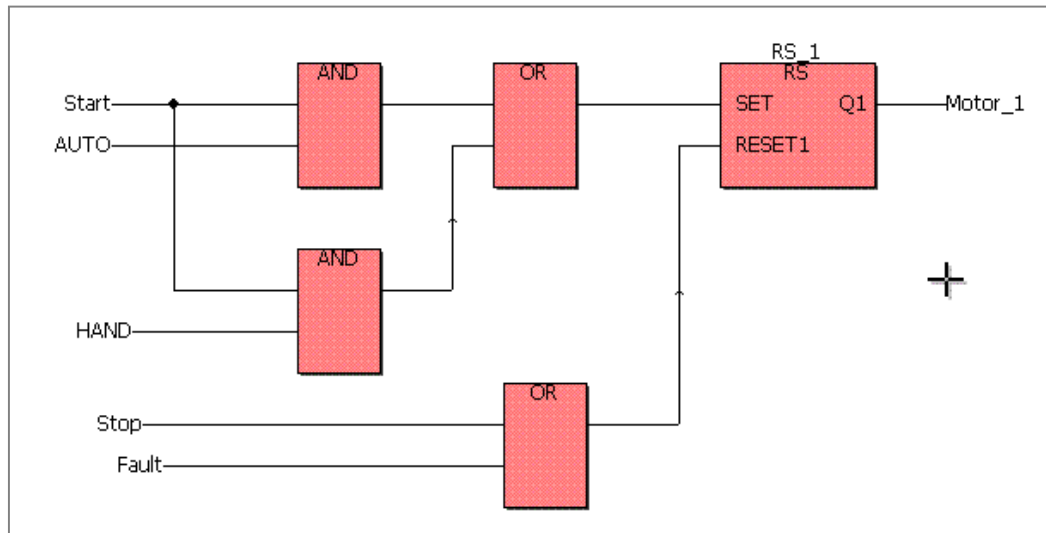


همانطور که دیده می شود در این پنجره نام متغیر و محدوده ی کاربری آن را می توان وارد نمود. متغیر بعنوان ورودی PLC است پس باید محدوده کاربری آن را عام در نظر گرفت. دو پنجره ی پایین تر محل ثبت مشخصات متغیر را یادآوری می کند. هنوز برخی از خواص متغیر Start معلوم نیست. به همین دلیل با زدن کلید OK و یا کلید Properties پنجره ی زیر باز می شود.



با زدن کلید OK مشخصات داده شده ثبت و پنجره بسته می شود.

همین روش را برای سایر متغیرها ادامه دهید و برنامه را مانند شکل زیر کامل کنید.



تنها نکته ی متفاوتی که باید در مورد متغیرها رعایت کنید مربوط به سیگنال Fault است. همانطور که در جدول متغیرها خواسته بودیم، این متغیر را از نوع محلی می خواهیم. پس دقت کنید که در پنجره ی تعریف متغیر مربوطه Scope آن را Local انتخاب کنید و در پنجره ی بعدی تنها نوع داده را وارد نمایید. هرگز از گزینه ی AT استفاده نکنید. آدرس استقرار متغیرهای محلی همراه با خود POU بوده و از دید کاربران مخفی است.

حالا اگر روی صفحه کار Global_Variables از درخت پروژه کلیک دبل کنید فایل باز شده و عبارات زیر را در آن خواهید دید:

```
VAR_GLOBAL (*AUTOINSERT*)
Start      AT %IX0.0 :   BOOL;
Stop       AT %IX0.1 :   BOOL;
HAND       AT %IX0.2 :   BOOL;
AUTO       AT %IX0.3 :   BOOL;
Motor_1    AT %QX0.0 :   BOOL;
END_VAR
```

اینها همان متغیرهایی هستند که شما نوشته اید و بصورت خودکار در این فایل ذخیره شده اند. در ادامه کار میتوانید به همین شیوه ادامه دهید یا می توانید از این پس حداقل سایر I/O را خودتان تایپ کنید. حالا که الگوهای ویرایش را در اختیار دارید احتمالاً تایپ دستی سریعتر است بخصوص با استفاده از دستورات Copy و Paste. از طرف دیگر همواره خودتان می توانید این جدول را منظم تر کنید. مثلاً تمام ورودی ها را به ترتیب شماره آنها مرتب کنید بعد تمام خروجی ها و نهایتاً سایر متغیرهای عام را مرتب کنید.

حالا نگاهی هم به صفحه کار متغیرهای محلی بیندازیم. برای این کار از پنجره ی درخت پروژه، و در بخش Logical POU's صفحه کار متغیرهای POU ی مورد نظر یعنی MainV را انتخاب و بر روی آن کلیک دبل نمایید. صفحه کار مربوطه باز می شود

اطلاعات مندرج در این فایل بشکل زیر خواهد بود.

```
VAR_EXTERNAL (*AUTOINSERT*)
```

```
Start : BOOL;  
Stop : BOOL;  
AUTO : BOOL;  
HAND : BOOL;  
Motor_1 : BOOL;  
END_VAR
```

```
VAR (*AUTOINSERT*)
```

```
Fault : BOOL;  
RS_1 : RS;  
END_VAR
```



ملاحظه می کنید که متغیرهایی که قبلاً بصورت عام تعریف شده بودند به این POU نیز معرفی شده اند. این معرفی با کلمه کلیدی **VAR_EXTERNAL** انجام شده است. توجه داشته باشید که در اینجا دیگر ذکر از محل استقرار متغیر نشده است. از این پس POU این متغیرها را هم شبیه متغیرهای داخلی خود می داند.

متغیر محلی **Fault** توسط کلمه کلیدی دیگری بنام **VAR** به POU معرفی شده است.

نکته مهم دیگری که دیده می شود این است که چون فلیپ فلاپ **RS** از نوع تابع یا **FU** نبوده و از نوع بلوک تابع یا **FB** است، هر نسخه از آن نیاز به حافظه ی داخلی مستقلی دارد. اما این **FB** چند بایت حافظه ی داخلی نیاز دارد؟ این حجم از حافظه را چگونه و توسط چه مکانیزمی دریافت می کند؟

در پاسخ باید گفت: از آنجایی که این قطعه برای PLC شناخته شده است، با اضافه شدن یک نسخه از متغیری بنام **RS_1** (که در حقیقت نام نسخه -instance- ای از بلوک تابع **RS** است)، POU به اندازه ی مناسبی برای آن از حافظه ی محلی خود اختصاص می دهد. بدین ترتیب در هر POU می توان تعداد بیشماری فلیپ فلاپ، تایمر، کانتر و ... بطور کلی از هر نوع **FB** ای قرار داد.

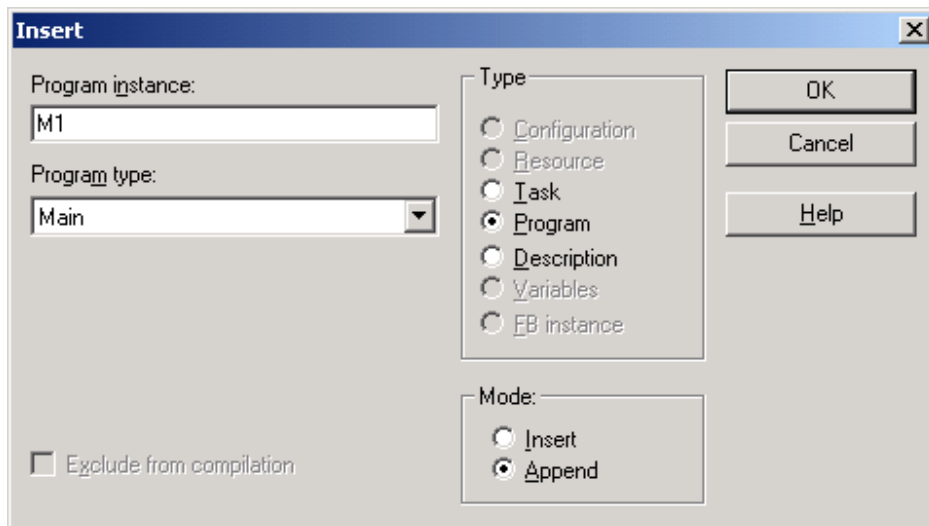
آنچه که بیان گردید تنها برای شرح ماقع در داخل سیستم بوده و کاربر نیازی به دانستن آنها ندارد.

همواره کاربر می توان برنامه کنترلی را به چند بخش منطقی و با معنی تقسیم کرده و هر بخش را در یک **PROG** با نامی مناسب پیاده سازی کند. پس از تهیه برنامه ها، کاربر باید آنها را برای اجرا به **Task** ها نسبت دهد. این موضوع را در بخش "نسبت دادن Program به Task" خواهیم دید.

نسبت دادن Program به Task

اگر POU ای از جنس Program را به PLC ارسال کنید بخودی خود اجرا نمی شود. برای اجرای Program ابتدا باید نسخه ای از آن را به یک Task مربوط کنید. از طرف دیگر ممکن است شرایط مختلفی برای اجرای یک برنامه وجود داشته باشد. مثلا ممکن است بخواهیم برنامه ای را یکبار در لحظه روشن شدن PLC، یکبار در اثر وقوع پدیده ای خاص و ... اجرا کنیم. پس لازم است نسخه های مختلفی از Program را برای هر یک از این موارد اختصاص دهیم.

- برای نسبت دادن Program به Task به شکل زیر عمل کنید.
- اگر در پروژه شما Task ای وجود ندارد، آن را ایجاد کنید
 - بروی Task کلیک راست کنید و از پنجره ی کوچکی که در سمت راست مکان نمای ماوس ظاهر می شود، گزینه Insert بمعنی وارد کردن را انتخاب کنید
 - پنجره ای بشکل زیر باز می شود. این پنجره چند منظوره است. بر اساس انتخاب شما بخشهایی از آن روشن و یا تاریک می شود.



- در بخش Type گزینه Program را انتخاب کنید.
 - الگوی تمام Program های موجود در پروژه در لیست کشویی Program type وجود دارد. آن Program ای که مورد نظر شماست انتخاب نمایید
 - در بخش Program instance نامی برای این نسخه از برنامه ی خود انتخاب کنید.
 - کلید OK را بزنید پنجره بسته شده و برنامه ای بنام M1:Main به Task شما وابسته میشود
- بدین ترتیب هر گاه که نوبت اجرای Task فرا برسد، نسخه ی M1 از الگوی برنامه ی Main را نیز اجرا می کند.

بدیهی است که میتوان Program های بیشتری را نیز به Task نسبت داد. ترتیب اجرای آنها در PLC بترتیب وارد کردن آنها در زیرشاخه Task مربوطه خواهد بود.

آرایش و ساماندهی سخت افزار PLC PLC Hardware Configuration

همواره PLC باید از سخت افزارهای مربوط به خود آگاه و از سلامت آنها مطمئن باشد. هرچند که PLC میتواند با مراجعه به کارتهای مختلف از ماهیت و حضور آنها مطلع شود ولی مجموعه سخت افزارهای آن را باید در فایل ساماندهی کرده و برای آن ارسال نمود. در این صورت با مقایسه سخت افزار موجود و سخت افزار مشخص شده در فایل مزبور، معلوم می شود که آیا سیستم کاملا سالم است یا نه.

برای ایجاد فایل ساماندهی سخت افزار (HwConfig)، در درخت پروژه روی گره Configuration کلیک دابل نمایید. جدول Hardware config باز می شود. این جدول دارای ۱۰ ردیف (معادل ۱۰ شیار موجود در راک بزرگ ۱۹ اینچی) برای وارد کردن ۱۰ نوع کارت سخت افزاری است.

در سمت چپ این جدول نام کارتهای مختلف PLC دیده می شود. اگر درخت سخت افزار بسته است آن را باز کنید. برای قرار دادن کارتهای مختلف در شیارهای مختلف، صرفا ماوس را روی نام کارت برده، کلید چپ آن را فشار داده و آن را فشرده نگه دارید. با همین شرایط، کارت مورد نظر را با حرکت ماوس کشیده، به سمت شیارها ببرید و در شیار مناسب کلید چپ را رها کنید. به مجموعه این حرکات Drag & drop می گویند.

همه کارتهای مورد نظر را به همین صورت در Slot های معادل در راک PLC قرار دهید.

Save کلید

Slot	Modules	Assigned Inputs	Assigned Outputs	Status
	Power Supply			
	CPU			
Slot 0	DI32_00	IB0 .. IB3		Enable
Slot 1	DI32_00	IB4 .. IB7		Enable
Slot 2	DI32_00	IB8 .. IB11		Enable
Slot 3	DO32_00		QB0 .. QB3	Enable
Slot 4	DO32_00		QB4 .. QB7	Enable
Slot 5				
Slot 6				
Slot 7				
Slot 8				
Slot 9				

شیارهای نصب کارت

درخت سخت افزار

CPU IP Address : 192.168.0.0.20

Date Of File : 11/28/2009 1:22:16 AM

S9:
saved correctly .

Build

ملاحظه می کنید که آدرس شروع تمام کارتها بصورت خودکار تعیین می شود. مثلا اگر اولین کارت را از نوع DI32-00 در شیار صفر قرار دهید، آدرس شروع کارت صفر می شود. چون این کارت دارای ۴ بیت ورودی دیجیتالی است، بایتهای ۰ و ۱ و ۲ و ۳ برای ورودیهای IB0, IB1, IB2, IB3 اختصاص می یابد. سایر کارتهای ورودی نیز در آدرسهای بعدی قرار می گیرند. کارتهای خروجی نیز از آدرس صفر شروع می شوند. هر کارت به تعداد مورد نیاز از این آدرسها برای خود رزرو می کند و کارتهای بعدی در آدرسهای آزاد بعدی قرار می گیرند.

با زدن کلید Save که بصورت نماد فلاپی دیسک نمایان است محتویات جدول بصورت فایل HwConfig در کامپیوتر ذخیره می شود. سپس باید این فایل را به PLC ارسال نمود. مادام که CPU این فایل را نداشته باشد و یا بین کارتهای نصب شده در راک و اطلاعات فایل HwConfig مغایرتی وجود داشته باشد، LED زرد رنگ Hardware Error روشن باقی می ماند. پس از ارسال فایل مناسب LED خاموش میشود.


گاهی اعتقاد دارید که فایل HwConfig را بدرستی ساخته و به PLC ارسال کرده اید ولی همچنان LED زرد خطا روشن مانده است. در این صورت به Information Stack مراجعه کنید تا شماره شیاری که کارت با مشکل مواجه شده است را پیدا کنید.

یاد آوری می شود که چنانچه تغذیه 24 Vdc کارت وجود نداشته باشد، CPU کارت را شناسایی نخواهد کرد. بنابراین ابتدا مطمئن شوید که این تغذیه سالم بوده و بدرستی به کارت داده شده است.

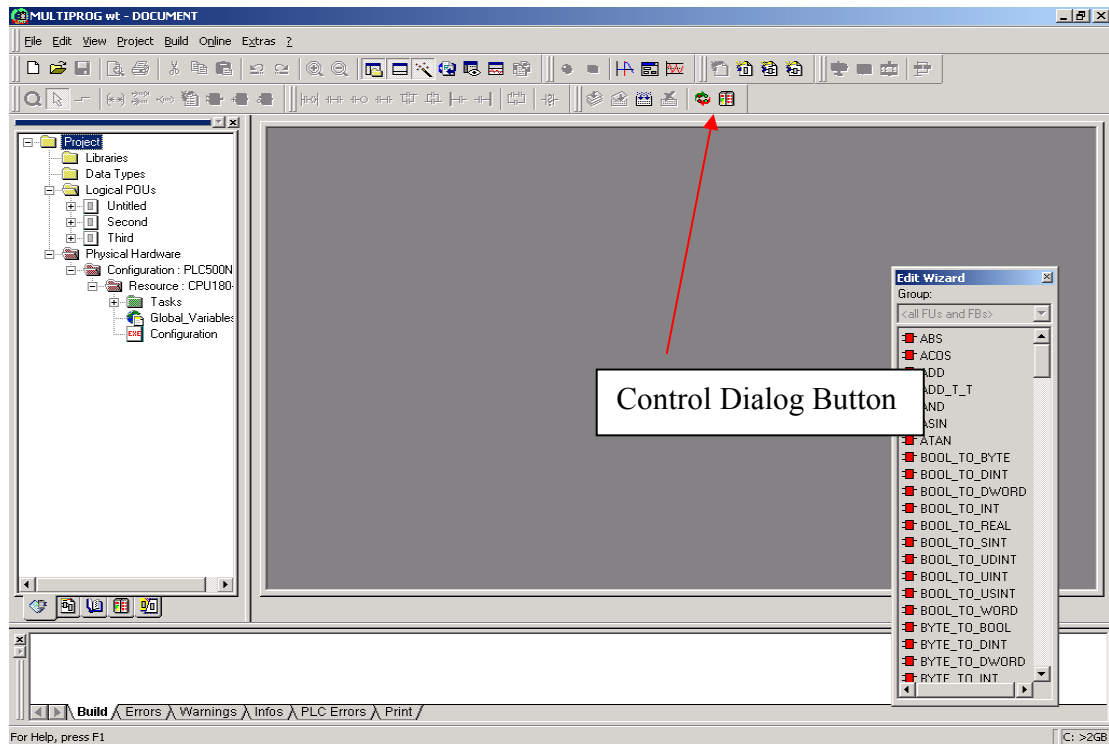
در جدول تنظیم سخت افزارها ستونی بنام Status نیز وجود دارد که می توان کارت را فعال (Enable) یا غیر فعال (Disable) کرد. حتی با نصب سخت افزار غیر فعال نیز آدرس های مناسب برای کارت رزرو می شود. بنابراین میتوان مادامی که هنوز سیم کشی های سخت افزاری را کامل نشده اند، PLC را برنامه ریزی کرده و اجرای برنامه را آزمایش نمود. گفتنی است که با غیر فعال کردن هر کارت، CPU حضور یا عدم حضور کارت را چک نمی کند، اگر کارت از نوع ورودی است ورودی های آن خوانده نمی شود و اگر از نوع خروجی است، خروجی برای آن هم ارسال نمی شود. این خاصیت کمک می کند که تنها با داشتن یک CPU برنامه نویسی و تست برنامه را شروع کنید.

نکته: در مورد کارتهای غیر فعال، هنگام Debug کردن برنامه (دیدن status اجرا در plc) مقدار ورودی ها و خروجی ها را می توان force کرد. گویی که واقعا چنین سیگنالی به PLC ارسال شده است. مقدار force شده در حافظه ی process image قرار می گیرد.

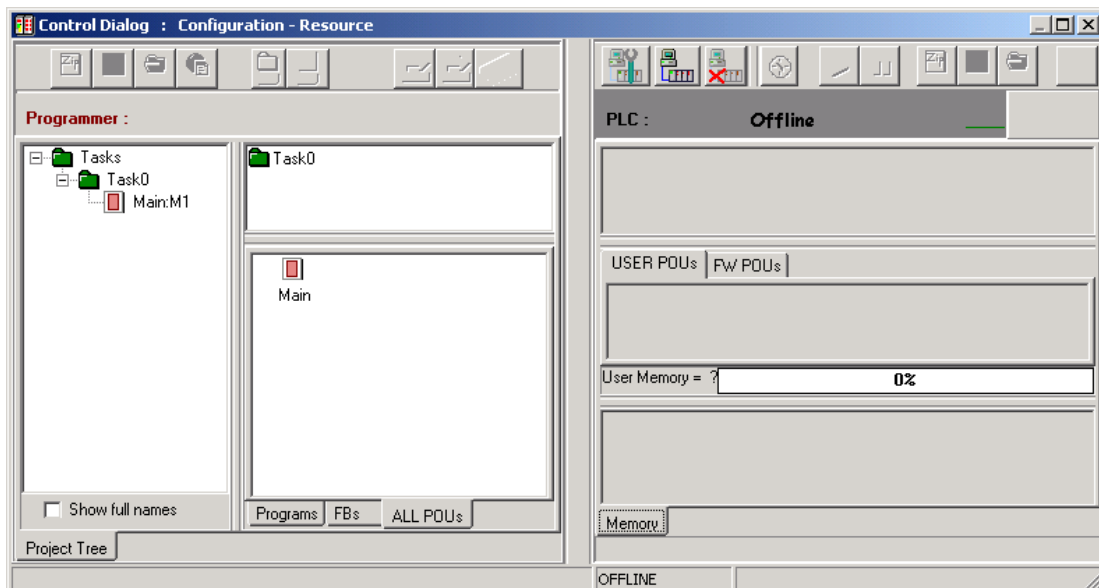
ارتباط با PLC

هرگونه ارسال و دریافت از PLC به PC و بالعکس از مسیر پنجره ی کنترل یا Control Dialog انجام می شود. برای وارد شدن به این پنجره کلیدی با نماد  که در میله ابزار های MWT قرار دارد را کلیک کنید.

بخش مهمی از ارتباطات با CPU از طریق پنجره ی کنترل (Control Dialog Box) انجام میشود. این پنجره هنگامی ظاهر میشود که دکمه ی مربوطه در میله ابزار MWT فشرده شود.

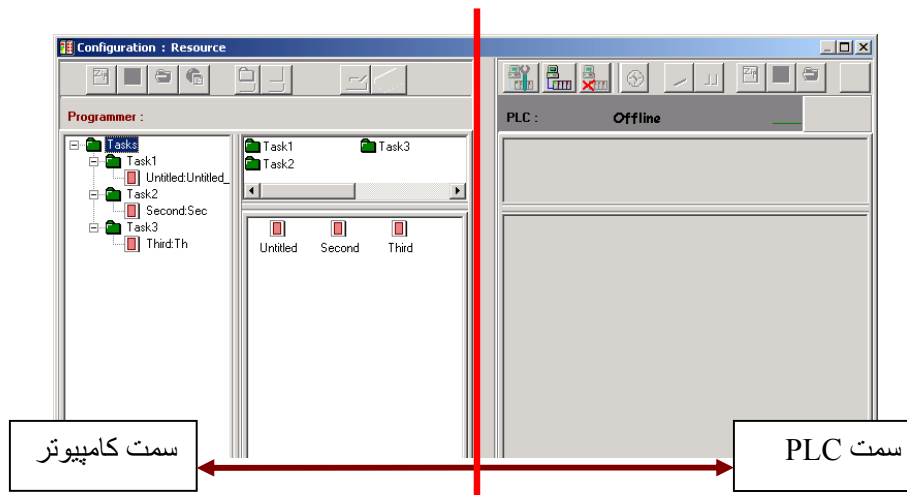


پنجره ی کنترل بشکل زیر است.



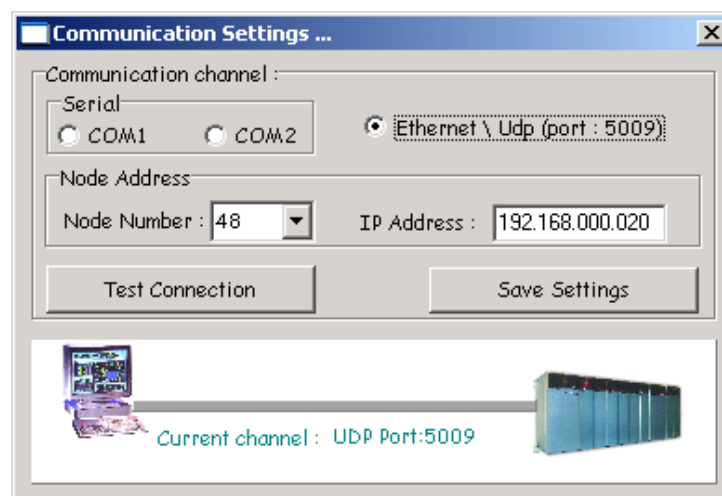
پنجره ی کنترل مهمترین رابطی است که از طریق آن میتوان برنامه ها، دستورات کاری، فایل ساماندهی سخت افزار و ... را به PLC ارسال و یا از آن دریافت نمود.

هنگامی که پنجره ی کنترل برای باز شدن آماده میشود، محتویات پروژه در کامپیوتر و نیز حافظه ی PLC را میخواند. پنجره ی کنترل دو بخش اصلی دارد. در سمت چپ فایلهای پروژه ای که در کامپیوتر قرار دارد و در سمت راست محتوای حافظه ی PLC را نمایش می دهد. اگر ارتباط با PLC برقرار نباشد، سمت راست این پنجره تاریک و بیشتر دکمه های آن بجز سه کلید غیر فعال خواهند بود. به شکل زیر توجه کنید.



سه دکمه ای که هنوز فعالاند در شکل روبرو دیده می شوند.

این سه کلید برای برقراری ارتباط، قطع ارتباط و تغییر در مقادیر پارامترهای ارتباطی بکار میروند. دکمه ی تنظیم پارامترهای ارتباطی (Communication Settings)، پنجره ی مربوطه را باز می کند.



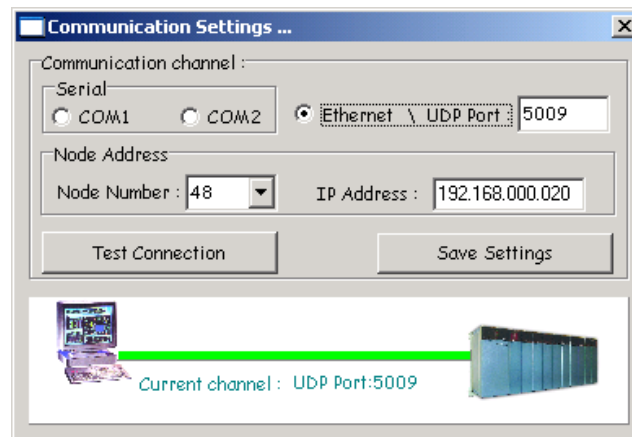
در پنجره ی تنظیم پارامترهای مخابراتی میتوانید آدرس IP و شماره گره ی مناسب برای تماس با PLC را وارد کنید. در ابتدا و بعنوان پیش فرض این مقادیر عبارتند از IP=192.168.000.020 و Node Number = 48.

کاربرد آدرس IP برای ارتباط با کابل اترنتی و شماره گره (Node Number) برای ارتباط با کابل سریالی است. کانال مخابراتی (Communication channel) مورد نظر خود را انتخاب کنید.

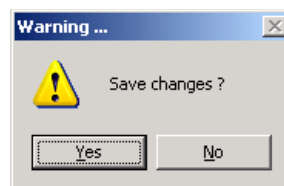
اگر اترنت انتخاب شود هر یک از دو نوع کابل مستقیم (Straight Ethernet cable) و یا معکوس (Cross Ethernet cable) را می توان بکار برد.

اگر مخابرات سریالی انتخاب شود، باید از کابل سریال استفاده نمود.

به هر صورت چنانچه پارامترها و کابل ارتباطی بدرستی انتخاب شوند، با زدن کلید Test ، نوار سبز رنگی در کانال ارتباطی ظاهر می شود و چنانچه ارتباط برقرار نشود این نوار برنگ خاکستری باقی می ماند.

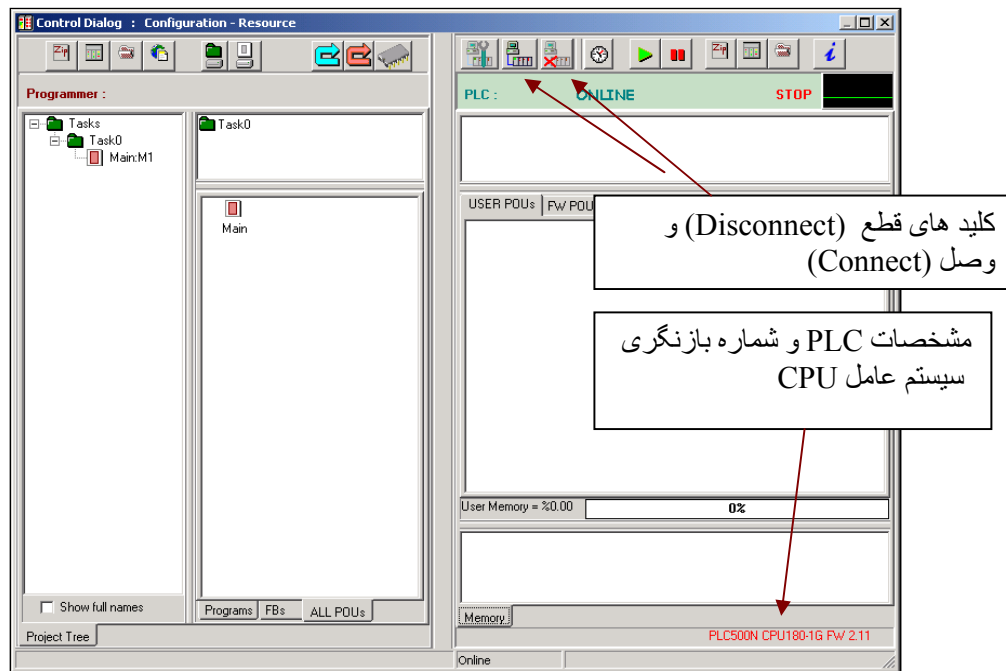


پس از نهایی کردن انتخابها، پنجره را ببندید. بوسیله پیامی به کاربر هشدار داده می شود که آیا مایل به ذخیره ی دائمی تنظیمات است؟



اگر گزینه Yes انتخاب شود، از این پس و در تمام طول پروژه، ارتباط با PLC با پارامترهای جدید صورت میگیرد. به عبارت دیگر، هر پروژه ای که در کامپیوتر قرار دارد دارای پارامترهای مخابراتی مخصوص به خود است به همان شکلی که هر PLC دارای پارامترهای مخصوص به خود می باشد.

اگر بین پارامترهای دو سمت پروژه و PLC هماهنگی بوجود آید، سمت راست پنجره ی کنترل نیز فعال و روشن می شود.



با دکمه های Disconnect و Reconnect ارتباط با PLC قطع و وصل می شود.

نکته: اگر از مخابرات اترنتی استفاده می کنید، کامپیوتر شما باید دارای آدرس IP استاتیکی و مناسبی باشد. منظور از آدرس مناسب، آدرسی است که سه بخش اول آدرس آنها مساوی باشند. مثلا اگر آدرس IP در سمت PLC برابر با 192.168.0.20 است، در سمت کامپیوتر آدرس IP را 192.168.0.11 قرار دهید. این دو آدرس در سه بخش اول آدرس یعنی 192.168.0. مساویند.

آدرس IP کامپیوتر خود را در Control Panel سیستم عامل ویندوز تنظیم نمایید.

با برقراری ارتباط، می توان برنامه ها و سایر فایلها را برای PLC ارسال نمود.

توجه: قبل از ارسال برنامه ها به PLC لطفا ابتدا مطمئن شوید که پروژه ی شما کامپایل شده و هیچ خطایی ندارد.

تنظیمات CPU180

برای CPU180 پارامترهایی در نظر گرفته شده است. این پارامترها عمدتاً آدرس ها و پارامترهای ارتباطی را تغییر می دهند. چند پارامتر فرعی دیگر نیز برای CPU180 تعریف شده که گرچه نقشی در چگونگی پردازش برنامه کنترلی یا در رفتار مخابراتی آن ندارند ولی از دیدگاه حفظ و نگهداری برنامه های PLC در دراز مدت بسیار اهمیت دارند.

در اینجا ضمن معرفی این پارامترها، چگونگی تأثیر آنها در کار CPU مورد بررسی قرار می گیرد. دو پارامتر بنامهای "IP Address" و "Node Number" آدرس CPU در ارتباطات اترنتی و سریالی را مشخص می سازند. Node Number یا شماره گره مخابراتی در شبکه سریال RS485 و آدرس IP در شبکه Ethernet کاربرد و اهمیت دارند.

مقدار پیش فرض این دو پارامتر عبارتند از:

IP Address = 192.168.0.20

Node Number = 48

هر گاه که حافظه ی CPU را کاملاً پاک می کنید، مقادیر پیش فرض بالا در این دو پارامتر قرار می گیرند.

مادامی که کامپیوتر مستقیماً به PLC متصل می شود (با ارتباط سریال یا اترنتی) و یا مادامی که در شبکه مخابراتی وسیله ی دیگری با این آدرسها قرار نداشته باشند، نیازی به تغییر این آدرسها نیست. ولی چنانچه بخواهید بیش از یک دستگاه PLC500 NSERIES در شبکه داشته باشید، بلافاصله شناسایی دستگاه ها در شبکه ناممکن شده و کشمکش بین آنها آغاز می گردد. این مشکل منحصر به PLC ها نبوده و در مورد هر وسیله ی دیگری نیز صادق است از جمله آدرس کامپیوترهای حاضر در شبکه. بنابراین فراهم آوردن امکان تغییر در آدرسهای تجهیزات الزام آور است.

آدرس IP کامپیوترها را می توان از راه Control Panel و آدرس PLC را هم می توان از پنجره ی Resource Settings تغییر داد.

در پنجره درخت پروژه، بر روی Resource کلیک راست کرده و گزینه Settings را انتخاب کنید. پنجره ای بشکل زیر باز می شود. (شکل این پنجره در مدل های مختلف CPU با هم کمی متفاوت است)

در شبکه سریالی Node Number را می توان از ۲ تا ۵۶ انتخاب کرد. آدرس ۱ مخصوص کامپیوتر است که نیازی به تنظیم ندارد. در شبکه ارتباطی سریال تنها یک کامپیوتر می تواند به عنوان Master وجود داشته باشد. بقیه plc ها Slave بوده و صرفاً پاسخگوی درخواستهای Master می باشند. کامپیوتر با ارسال درخواستهای خود به آدرس مورد نظر پاسخ خود را گرفته و بعد به سراغ آدرس دیگر می رود.

در شبکه Ethernet آدرس plc ها توسط IP Address مشخص میشود. آدرس IP تمام کامپیوترها و plc ها باید در یک محدوده مشخص باشند. این محدوده معمولاً با سه رقم اول آدرس تعیین می شود. تمام تجهیزاتی که سه رقم اول آدرس آنها یکسان باشد، روی یک شبکه محلی (LAN) قرار داشته و میتوانند مستقیماً با هم در ارتباط باشند. مثلاً آدرسهای زیر روی یک LAN می باشند.

IP Address = 192.168.0.21

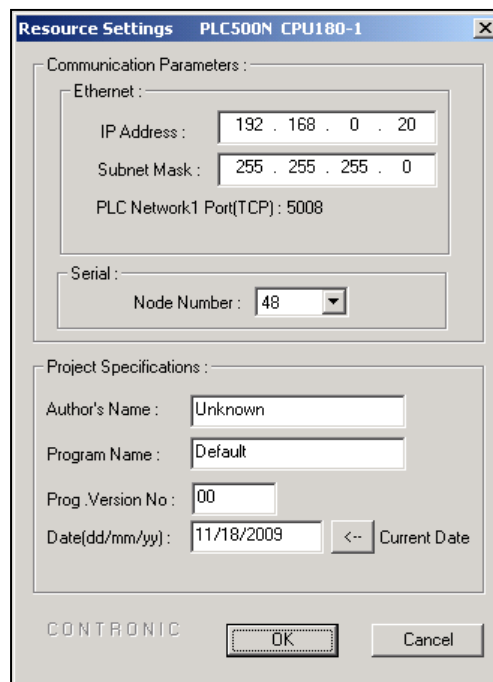
IP Address = 192.168.0.22

IP Address = 192.168.0.203

IP Address = 192.168.0.205

در شبکه های اترنتی تعداد وسایل Master محدود نبوده و هر دستگاه کامپیوتر می تواند با چند دستگاه plc به عنوان Slave ارتباط بگیرد. بنابراین با یک دستگاه کامپیوتری که نرم افزار MULTIPROG را اجرا می کند می توان پروژه ای ساخت که شامل چند دستگاه plc باشد. به همین ترتیب می توان برنامه نویسی plc ها را بصورت تیمی توسط چند دستگاه کامپیوتر انجام داد.

امکان حضور چند Master در یک شبکه این امکان را هم فراهم می آورد که مثلا ضمن اینکه چند دستگاه کامپیوتر مونیتورینگ و پانل اپراتوری در حالت جمع آوری اطلاعات از یک plc هستند، کامپیوتر یا کامپیوترهای دیگری در حال برنامه نویسی و debugging آن باشند.



در پنجره ی Resource Settings علاوه بر پارامتر های مخابراتی به شرحی که بیان شد، چند پارامتر دیگر که شامل مشخصات پروژه ، نام نویسنده و ... است نیز وجود دارد که میتواند برای شناسایی برنامه ای که در حال اجراست اطلاعات مفیدی در اختیار بگذارد.

با زدن کلید OK اطلاعات این پنجره بصورت فایل بنام ResConfig در می آید. بعدا خواهید دید که با ارسال این فایل به CPU ، آدرس ارتباطی آن تغییر خواهد کرد.

برنامه ها و فایل هایی که به PLC ارسال می شوند

فایل های زیر را می توان به PLC ارسال نمود. ارسال برخی از این فایلها انتخابی، بعضی اجباری و یکی از آنها را بهتر است در آخرین مرحله ی طراحی پروژه ارسال کرد.

- | | |
|---------|--|
| اجباری | • فایل ساماندهی سخت افزار - HWConfig |
| اجباری | • Program Organization Units (POUs) |
| اجباری | • دستورات کاری- Tasks : |
| انتخابی | • فایل تنظیم های سی پی یو یا Resource بنام - ResConfig |
| انتخابی | • فایل متغیرهای عام - GlobalVars |
| انتخابی | • فایل فشرده ی پروژه - ZippedPrj |

فایل ResConfig هنگامی مورد نیاز است که PLC در شبکه ی LAN قرار داشته و باید دارای آدرس منحصر به فردی باشد. البته در این فایل اطلاعاتی در مورد برنامه، تاریخ نگارش و ... نیز نوشته می شود.

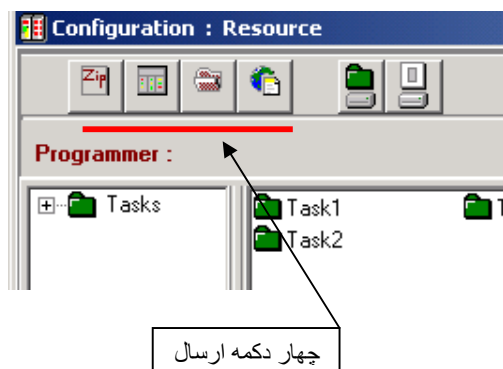
فایل GlobalVar هنگامی در PLC بکار می رود که برخی از متغیرهای عام مقدار اولیه (Initial value) داشته باشند و یا از نوع پایدار (RETAIN) تعریف شده باشند.

فایل فشرده ی ZippedPrj را زمانی به PLC ارسال می کنیم که پروژه کاملاً نهایی شده باشد و آماده ی Fix شدن در حافظه ی Flash باشد. با باز خوانی این فایل از PLC، تمام پروژه قابل بازیابی است.

ارسال سایر فایلها به PLC مطلقاً ضروری بوده و باید به PLC فرستاده شوند.

ارسال برنامه ها و فایل ها به PLC

چهار دکمه برای ارسال فایل های ZippedPrj، HWConfig، ResConfig و GlobalVars به PLC در پنجره کنترل پیش بینی شده است. با کلیک کردن روی هر یک از این چهار کلید که در سمت چپ این پنجره قرار دارند فایل مربوطه ارسال می شود.



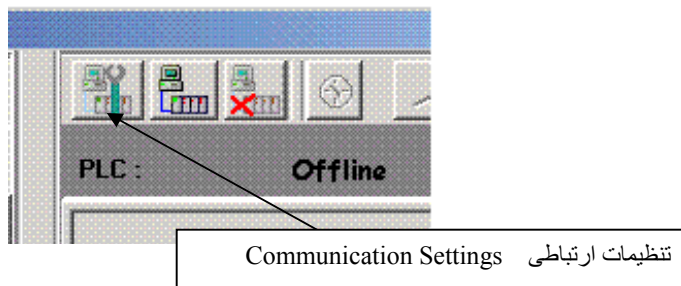
از این چهار دکمه برای ارسال فایل های فشرده پروژه (ZippedPrj)، سامانه سخت افزار (HWConfig)، سامانه پردازشگر (ResConfig) و متغیرهای عام (GlobalVars) استفاده میشود.

برای اطمینان از عملکرد هر یک از این چهار دکمه، ماوس را روی آنها قرار دهید تا متن مربوط به عملکرد هر یک ظاهر شود.

فایل ساماندهی سخت افزار را بدون هیچ محدودیتی می توان برای PLC فرستاد. اگر این فایل با سخت افزار نصب شده مطابقت داشته باشد، LED کوچک زرد رنگ خطای Hardware که روی مدول CPU قرار دارد

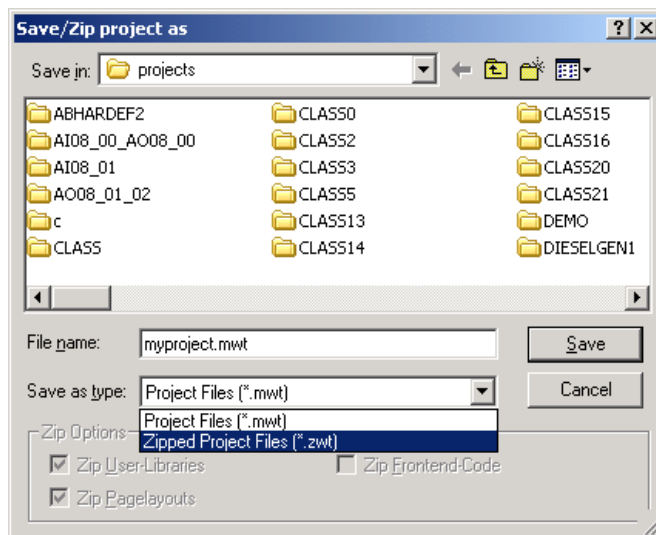
خاموش میشود. خاموش شدن این LED نمایش دهنده ی آن است که ساماندهی سخت افزار بدرستی صورت گرفته است.

فایل ResConfig را نیز با دکمه مربوطه می توان به PLC ارسال نمود. چون پارامترهای ارتباطی در زمان استارت CPU فعال می شوند، پس از ارسال این فایل، CPU ری ست شده و پس از آن پارامترهای جدید فعال می شوند. توجه داشته باشید که پس از ارسال این فایل ارتباط با PLC ممکن است قطع شود. در این صورت باید کانال ارتباطی در سمت کامپیوتر نیز متناسباً تغییر کند. برای برقراری ارتباط مجدد باید تنظیمات کانال ارتباطی با PLC را با دکمه Communication Settings مجدداً تنظیم کنید.



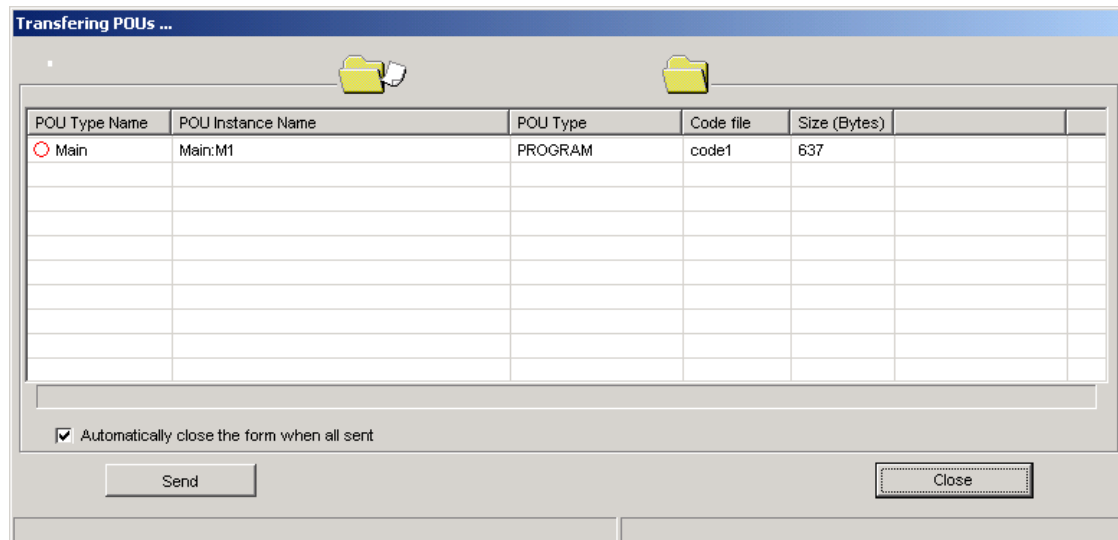
فایل GlobalVar را با کلیک بر روی دکمه مربوطه به PLC ارسال کنید. توجه داشته باشید که پس از ارسال این فایل خواص متغیرهای عام برای PLC قابل شناسایی خواهند بود. اگر برخی از متغیرها مقدار اولیه داشته باشند، یکبار پس از ارسال این فایل، مقادیر اولیه متغیرها تنظیم میشوند و در سایر موارد در هنگام استارت های سرد و گرم CPU.

قبل از ارسال فایل ZippedPrj، لازم است که ابتدا این فایل را تولید کنید. تولید این فایل با save کردن فایل بصورت zipped انجام می شود. از منوی File گزینه ی Save project as/ Zip project as را انتخاب کنید.

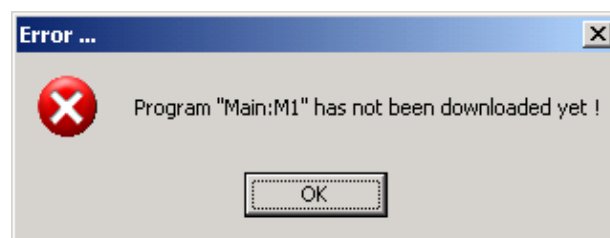


در قسمت Save as type گزینه * .zwt را انتخاب کرده و کلید Save را بزنید. فایل فشرده ی پروژه در کامپیوتر با دنباله zwt ذخیره می شود. پس از ایجاد این فایل می توانید آن را برای PLC ارسال کنید. در این مورد در بخش های بعد بیشتر توضیح داده خواهد شد.

ارسال POU ها و Task ها بسادگی و به روش آشنای drag & drop انجام می شود. ماوس را روی POUی مورد نظر برده کلید سمت چپ را بفشارید. کلید را فشرده نگه داشته آن را به پنجره ی مشابه روبروی آن در سمت PLC کشانده و کلید ماوس را رها کنید. با انجام این کار پنجره ی شکل زیر ظاهر خواهد شد. کلید Send را بزنید تا فایل ارسال شود. پس از ارسال پنجره را ببندید.

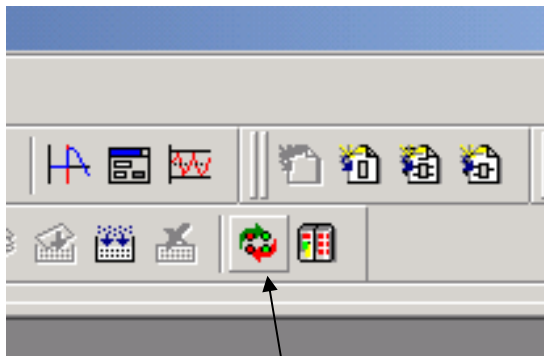


ارسال دستورات کاری (Tasks) نیز به همین ترتیب انجام می شود لیکن قبل از آنکه Task را بتوان به PLC ارسال نمود، POUهای مورد نیاز آن باید به PLC ارسال شده باشند. در غیر اینصورت پنجره ای باز شده و پیامی را مبنی بر همین موضوع را به اطلاع می رساند.



عیب یابی (Debugging) برنامه ها با دیدن Status اجرای برنامه ها

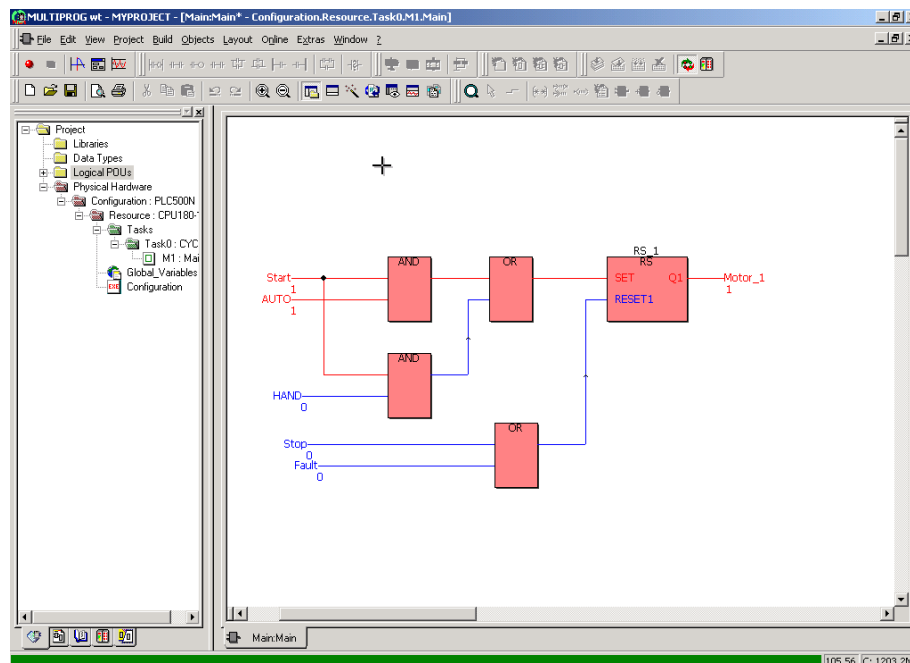
پس از ارسال برنامه ها به CPU و فراخوانی آنها در task ها، می توان در تماس آنلاین با CPU نحوه ی اجرای برنامه ها را مشاهده و عیب یابی کرد.
روی دکمه Debug کلیک کنید. نوار سبز رنگی در پایین صفحه MWT ظاهر میشود تا یادآوری کند که تماس آنلاین با PLC برقرار شده است.



Debug button

کلید Debug برای برقراری ارتباط آنلاین با PLC و مشاهده نحوه اجرای برنامه بکار میرود.

صفحه کار برنامه ای را که مایلید عیب یابی کنید باز کنید. این کار با کلیک دوبل روی صفحه کار POU ی مورد نظر انجام می شود.



چنانچه نسخه های متعددی از POU وجود داشته باشد، MWT برای برقراری ارتباط، نام نسخه ی مورد نظر را سوال می کند.

با باز کردن صفحه ی کار سایر POU ها، نمایش آنلاین آنها را نیز مشاهده خواهید کرد. تعداد پنجره هایی که می توانید بدین شکل باز کرده و اجرای همزمان آنها را مشاهده کنید حد اکثر ۳۲ پنجره است.

برای مشاهده Status همزمان چند POU میتوان از منوی Windows گزینه های Cascade، Tile Horizontally و یا Tile Vertically را انتخاب کرد.

ایجاد و ویرایش FB توسط کاربر

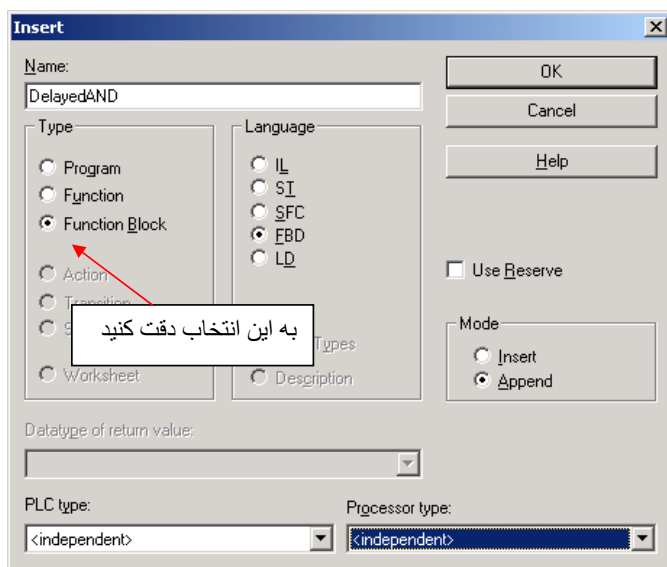
کاربر یا USER می تواند علاوه بر توابع و بلوک های تابع موجود در PLC500 NSERIES، خود نیز بلوک توابع مورد نیازش را تولید کند. نیاز به نگارش این توابع زمانی احساس میشود که کاری تکراری داشته و یا به عملکرد خاصی نیاز داشته باشیم که از عهده توابع موجود ساخته نباشد. در این صورت کاربر می تواند عملکرد مورد نظر خود را در قالب بلوک تابعی FB نوشته و بعد به هر تعداد از آن استفاده کند. گفتنی است که اگرچنین توابعی بسیار پر کاربرد باشد، کاربر می تواند از آنها توابع کتابخانه ای ساخته و در سایر پروژه ها نیز از آنها استفاده بعمل آورد.

برای تشریح مراحل تدوین یک USER FUNCTION_BLOCK، با یک مثال کوچک شروع می کنیم. بلوک تابع ساده ی ما بلوک تابعی بنام DelayedAND است. ورودی ها و خروجی های این بلوک تابع عبارتند از:

- ورودی IN1 از نوع BOOL
- ورودی IN2 از نوع BOOL
- ورودی DELAY از نوع TIME
- خروجی OUT از نوع BOOL
- خروجی ET از نوع TIME

عملکرد این FB بدین شرح است که می خواهیم بر خلاف تابع ساده ی AND که خروجی آن بلافاصله پس از TRUE شدن ورودی هایش TRUE میشود، چنانچه هر دو ورودی آن TRUE شد و بمدت مشخصی TRUE ماند خروجی آن TRUE شود. از طرف دیگر می خواهیم زمان سپری شده را در خروجی تابع داشته باشیم.

احتمالا کاربرد چنین تابعی زیاد است. مثلا در هنگام پر شدن مخزنی که در سطح آن تلاطم زیادی وجود دارد، وقتی اقدام به بستن شیر ورودی می کنیم که مدتی از TRUE شدن سیگنالهای ورودی گذشته و واقعا سطح مخزن به حد نصاب رسیده باشد. این مدت نیز در مخازن مختلف متفاوت است. بنابراین تصمیم گرفته ایم که آن را بشکل FB نوشته تا بتوانیم از آن در چند مورد استفاده کنیم.



یادآوری می شود که FB یکی از انواع POU هاست. بنابراین برای ایجاد آن ابتدا باید به روش گفته شده در بخش "ایجاد و ویرایش برنامه PLC" عمل کنیم با این تفاوت که در پنجره ی Insert، نوع یا Type برنامه را Function Block انتخاب کنیم.

نام مورد نظر را هم تایپ کنید. در این

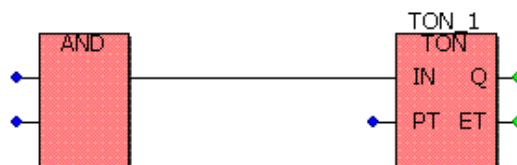
مثال DelayedAND را نوشته ایم. زبان نگارش FB را هم انتخاب کنید.

با زدن کلید OK مطابق معمول صفحات کاری سه گانه برای این FB ایجاد شده و در درخت پروژه و در زیر گروه Logical POU's قرار می گیرند. ای صفحات کاری عبارتند از:

- DelayedANDT
- DelayedANDV
- DelayedAND

- اولین صفحه کار یا worksheet که به انتهایش حرف T اضافه شده برای نوشتن Text و متون راهنمای کاربر است.
- دومین صفحه کار یا worksheet که به انتهایش حرف V هم اضافه شده برای ثبت متغیرها یا Variable های FB است.
- سومین صفحه کار یا worksheet که تنها از نام FB استفاده کرده برای نوشتن کد برنامه است.

صفحه کار DelayedAND را باز کرده و توابعی بشکل زیر را در آن قرار دهید.

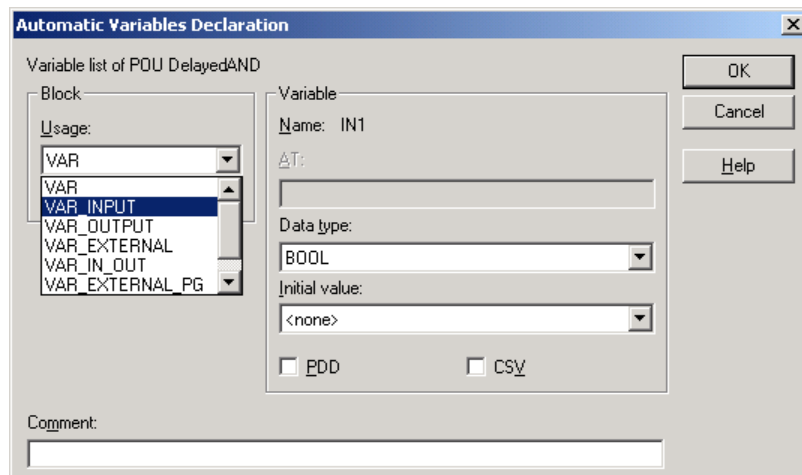


حالا باید متغیرهای آن را تعریف کنید.

۱. روی اولین ورودی گیت AND کلیک دابل کنید. پنجره ای بنام Variable مثل شکل زیر باز می شود. مانند شکل نام متغیر را IN1 و محدوده کاربری متغیر را محلی انتخاب کنید.

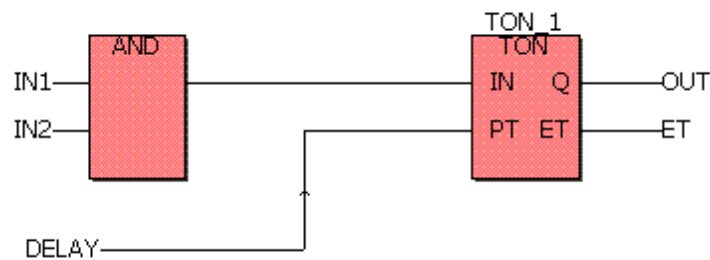


۲. با زدن کلید OK و یا Properties پنجره ی زیر باز می شود. مطابق شکل نوع متغیر را BOOL و کاربرد یا Usage آن را VAR_INPUT انتخاب کنید.



۳. کلید OK را بزنید. اولین متغیر بنام IN1، از نوع BOOL و با کاربری VAR_INPUT ثبت شده است.
۴. همین مراحل را برای ورودی دوم AND با نام IN2 تکرار کنید.
۵. مراحل مشابهی را برای ورودی سوم با نام DELAY و داده ی نوع TIME تکرار کنید.
۶. مراحل مشابهی را برای خروجی اول FB بنام OUT با داده ی نوع BOOL و کاربری VAR_OUTPUT تکرار کنید.
۷. مراحل مشابهی را برای خروجی دوم بنام ET (مخفف عبارت انگلیسی Elapsed Time بمعنی زمان سپری شده) با داده ی نوع TIME و کاربری VAR_OUTPUT تکرار کنید.

شکل درونی تابع را در زیر مشاهده می کنید.



اگر به صفحه کار متغیرهای DelayedAND مراجعه کنید، تعریف متغیرها را به این شکل خواهید دید.

```

VAR      (*AUTOINSERT*)
  TON_1  :    TON;
END_VAR

VAR_INPUT  (*AUTOINSERT*)
  IN1    :    BOOL;
  IN2    :    BOOL;
  DELAY  :    TIME;
END_VAR

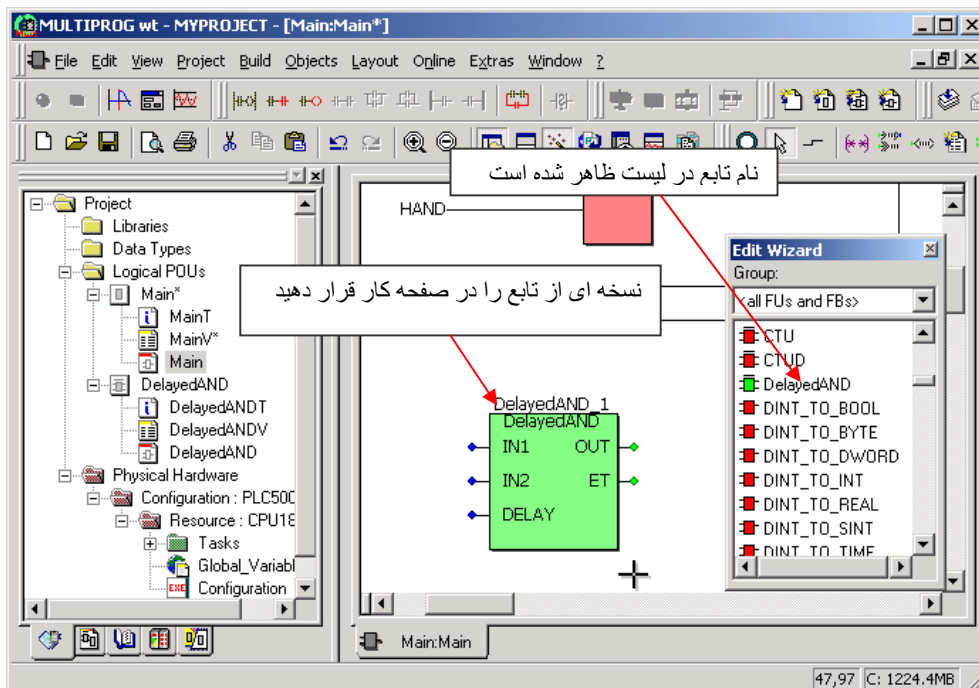
VAR_OUTPUT (*AUTOINSERT*)
  OUT    :    BOOL;
  ET     :    TIME;
END_VAR
    
```

همانگونه که می بینید متغیرهای این POU در سه نوع محلی، ورودی و خروجی طبقه بندی شده اند.

اگر همه مراحل را بدرستی انجام داده باشید با زدن کلید F9 برنامه کامپایل شده و به خطایی برخورد نمی کنید. بلوک تابع شما برای استفاده آماده است.

استفاده از بلوک تابع DelayedAND

از بلوک تابع DelayedAND مثل هر FB دیگری می توان استفاده کرد. صفحه کار Main را باز کنید و در نقطه ای از آن کلیک کنید تا علامت + ظاهر شود. اگر پنجره ی Edit Wizard بسته است آنرا باز کنید. حالا در لیست توابع نام DelayedAND را هم می بینید. بر روی آن کلیک دابل کرده و نسخه ای از آن را مطابق شکل زیر در صفحه کار قرار دهید.



تکمیل و ارسال برنامه ها به PLC

مثل سایر FB ها روی ورودی ها و خروجی های آن کلیک دابل کرده و متغیرهای I/O را معرفی کنید. لیست I/Oهای تابع را بشکل زیر قرار دهید.

نام متغیر Variable Name	نوع داده Data type	محدوده کاربری Scope	آدرس Address
LS100	BOOL	عام یا Global	I0.4
LS101	BOOL	عام یا Global	I0.5
TANK_FULL	BOOL	عام یا Global	Q0.1
ElapsedTime	TIME	محلی یا Local	-

LS100 و LS101 دو Level Switch هستند که به عنوان ورودی به PLC داده شده اند. خروجی TANK_FULL زمانی که مخزن پر شود و زمان مورد نظر هم سپری شده باشد فعال می گردد. پارامتر خروجی ElapsedTime به معنی زمان سپری شده نشان می دهد که از زمان مورد نظر چه مقدار گذشته است.

توجه داشته باشید که در این مثال برای ورودی DELAY متغیر زمان را بصورت صریح یا به عبارت دیگر عدد ثابت وارد کرده ایم. اگر پارامتر زمان در متغیر دیگری قرار دارد نام آن را به تابع معرفی کنید و اگر می خواهید زمان را بشکل صریح و عدد ثابت وارد کنید از یکی از عبارتهای زیر استفاده کنید.

T#5S, t#5S, time#5s, TIME#5S

در مورد مقادیر ثابت در help مربوط به About IEC1131 شرح کاملی وجود دارد. در MWT صفحه help مربوطه را باز کنید و در بخش Index آن عبارت Literals را تایپ کنید. در بخش duration انواع مختلف دیگری از اشکال وارد کردن عدد ثابت را برای زمان مشاهده می کنید. در مورد سایر انواع متغیرها هم بررسی نمایید.

```

VAR_EXTERNAL (*AUTOINSERT*)
  Start : BOOL;
  Stop : BOOL;
  AUTO : BOOL;
  HAND : BOOL;
  Motor_1 : BOOL;
  LS100 : BOOL;
  LS101 : BOOL;
  TANK_FULL : BOOL;
END_VAR

VAR (*AUTOINSERT*)
  Fault : BOOL;
  RS_1 : RS;
  DelayedAND_1 :
  DelayedAND;
  ElapsedTime : TIME;
END_VAR
    
```

لیست کامل متغیرهای Local از صفحه کار MainV را در شکل رویو مشاهده می کنید.

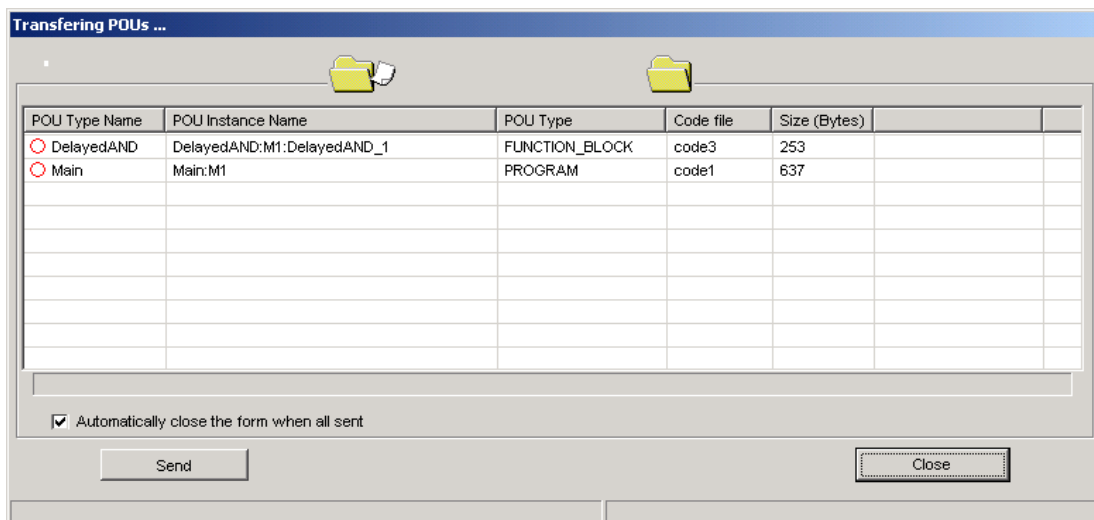


```
VAR_GLOBAL (*AUTOINSERT*)
  Start   AT %IX0.0   :   BOOL;
  Stop    AT %IX0.1   :   BOOL;
  HAND    AT %IX0.2   :   BOOL;
  AUTO    AT %IX0.3   :   BOOL;
  Motor_1 AT %QX0.0   :   BOOL;
  LS100   AT %IX0.4   :   BOOL;
  LS101   AT %IX0.5   :   BOOL;
  TANK_FULL AT %QX0.1 :   BOOL;
END_VAR
```

لیست کامل متغیرهای Global از صفحه کار Global_Variables را در شکل روبرو مشاهده می کنید.

پروژه را کامپایل کنید. وقتی که تمام برنامه بدرستی کامپایل شود و خطایی نداشته باشد می توان آنها را به PLC ارسال نمود.

در برنامه ی Main تغییر ایجاد کرده ایم پس باید آن را یکبار دیگر برای PLC بفرستیم. در موقع ارسال متوجه می شویم که در لیست ارسال نسخه ای از بلوک تابعی بنام DelayedAND هم در لیست ارسال ظاهر می شود. از آنجایی که کامپایلر می داند که در لیست متغیرهای برنامه Main متغیری از نوع DelayedAND دیده می شود و PLC آن را نمی شناسد پس یک نسخه از آن را برای CPU و قبل از ارسال برنامه Main ارسال می کند تا هنگام مراجعه به آن چیزی برای اجرا وجود داشته باشد. (شکل زیر).



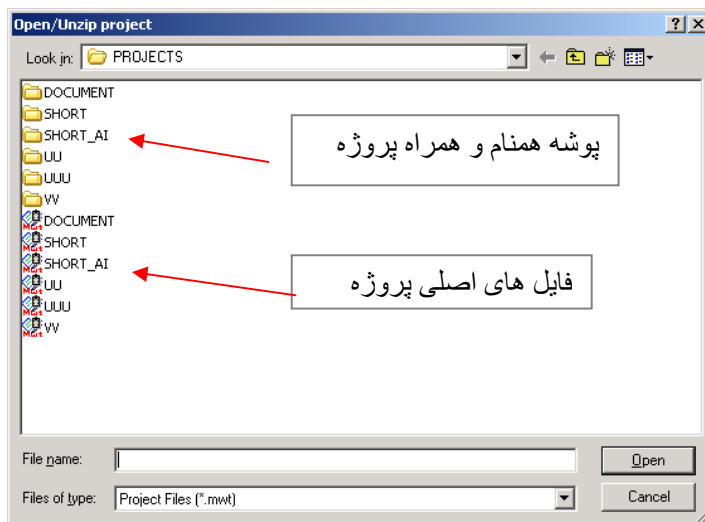
POU Type Name	POU Instance Name	POU Type	Code file	Size (Bytes)
DelayedAND	DelayedAND:M1:DelayedAND_1	FUNCTION_BLOCK	code3	253
Main	Main:M1	PROGRAM	code1	637

توجه داشته باشید در موقع ارسال تنها فایل هایی را ارسال کنید که تغییری در آنها ایجاد کرده اید. پس در این مثال لازم نیست که مجدداً Task را ارسال کنید.

نکته: گاهی دیده می شود که کاربران سعی می کنند نسخه ای از FB (مثل DelayedAND) را برای اجرا به Task اضافه کنند. باید توجه داشت که وظیفه ی Task اجرای POU هایی از جنس Program هاست. اجرای POU هایی از جنس FU یا FB به عهده Program هاست.

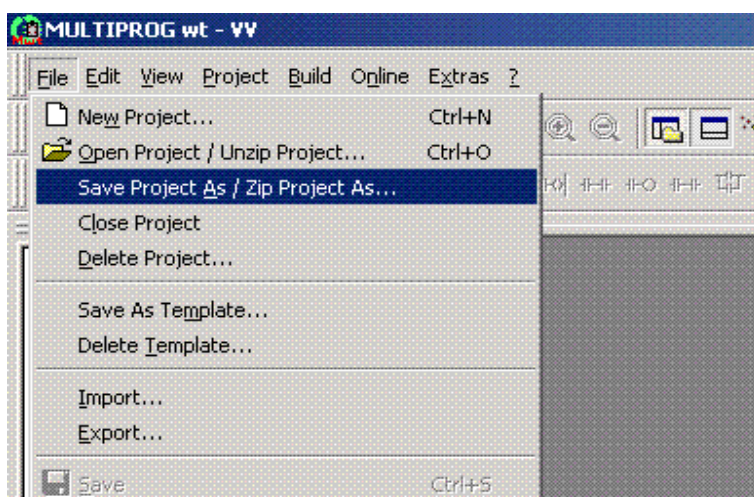
ذخیره پروژه ها در کامپیوتر برنامه نویسی

پروژه ها بصورت پیش فرض در پوشه ای (folder) بنام Projects از مسیر نصب MWT تشکیل می شوند. هر پروژه ای شامل یک فایل و یک پوشه همنام نام پروژه است. مثلا پروژه ای بنام SHORT دارای فایل بنام SHORT.mwt و پوشه ای بنام SHORT است. در شکل زیر چند پروژ دیده می شود.

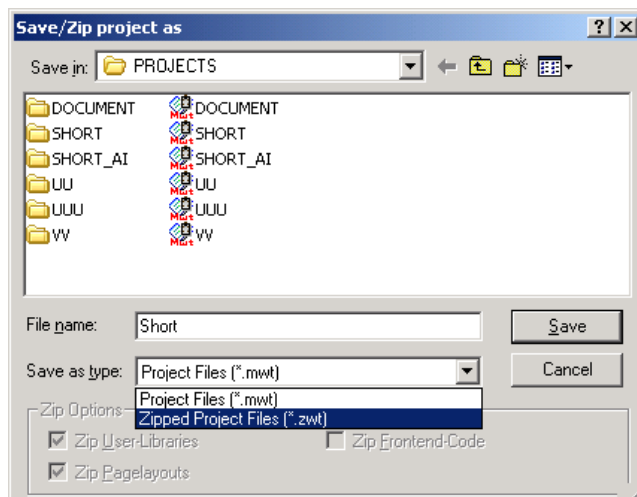


برای باز کردن پروژه در MWT باید فایل "name.mwt" آن را باز کرد. کاربر معمولا با پوشه ی همنام و همراه آن کاری ندارد.

کاربر پروژه ی خود را در پوشه Projects در مسیر نصب MWT به یکی از فرمت های عادی mwt و یا فشرده شده (Zipped) با پسوند zwt ذخیره می کند. برای save کردن به منوی File رفته و گزینه Save Project As / Zip Project As... را انتخاب کنید همان طوری که در شکل زیر دیده میشود.



نامی برای پروژه انتخاب کرده و یکی از فرمت های mwt یا zwt را انتخاب کنید.



در بیشتر موارد در زمان ایجاد پروژه ، شما با فرمت mwt کار می کنید.

چه وقت از فرمت zwt استفاده می شود؟

برنامه های plc در استاندارد IEC1131-3 به ۵ زبان مختلف تولید می شوند. قبل از ارسال برنامه ها به plc نهایتا همه ی آنها به زبانی که بسیار شبیه زبان IL است ترجمه می شوند.

آنالیز برنامه بزبان IL گرچه امکانپذیر ولی بسیار دشوار است. درست بهمین دلیل بازخوانی برنامه از plc به زبان IL کارسودمندی نیست. در plc های مدرن هم که نوشتن برنامه های پیچیده بزبان های مختلف بسیار متداول گشته، هر روز به پیچیدگی های این کار افزوده می شود.

در MWT، کدها و فایل های ایجاد شده برای پروژه گرچه بسیار دقیق و حاوی جزئیات کاملی است، لیکن برای نگهداری در حافظه ی plc بسیار بزرگ است. خوشبختانه MWT می تواند پروژه شما را با فرمت zip نیز ایجاد کند که برای نگهداری در حافظه plc حجم قابل قبولی دارد. پس از بازخوانی این فایل از plc، MWT میتواند مجددا آن را باز کرده و نسخه کاملی از پروژه را باز یابی کند.

در پایانی ترین بخش کار یعنی هنگامی که پروژه ی شما کاملا آماده شده و تصحیحات نهایی آن بعمل آمده باشد، احتمالا می خواهید نسخه ی zip شده پروژه را برای ارسال و نگه داری دائمی در حافظه Flash تهیه کنید.

ارسال و دریافت فایل های zip شده ی پروژه به روشهایی که در بخش های قبل توضیح داده شد انجام می شود.

کاربرد دیگری که برای فایل zip متصورست استفاده از آن برای ارسال اینترنتی پروژه از شهری به شهر دیگر است. برخی از کاربران plc که ابهامات و سوالاتی داشته باشند می توانند فایل zip را ضمیمه email خود کرده و برای کنترونیک ارسال نمایند. کارشناسان کنترونیک با بررسی کامل پروژه می توانند راهنمایی های با ارزشی به کاربران ارائه نمایند.

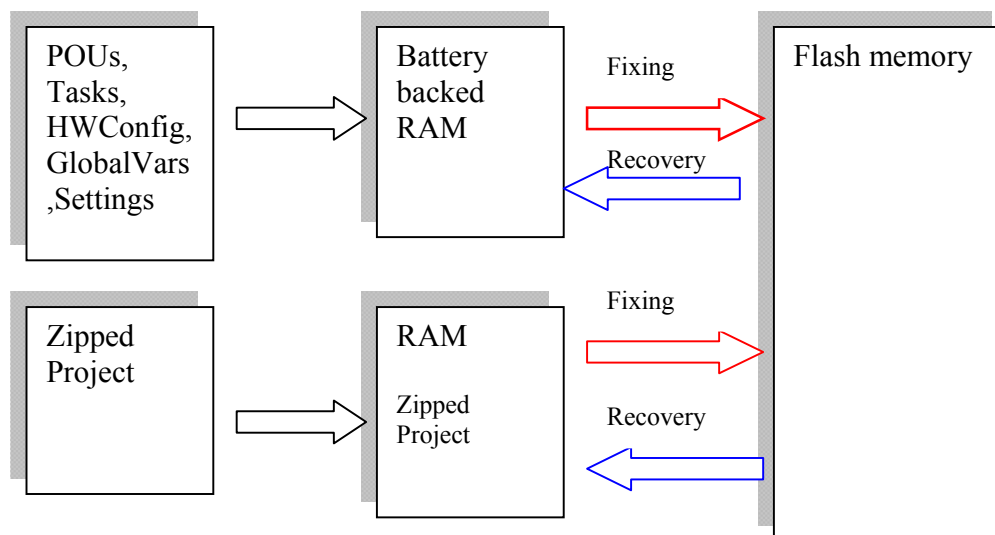
ذخیره سازی برنامه های PLC در حافظه Flash Fixing Programs in Flash Memory

برنامه های کاربر پس از ارسال به plc در حافظه RAM باتری دار قرار می گیرد. این برنامه ها از همان ناحیه RAM نیز اجرا می شوند.

برای نگهداری برنامه های بسیار ارزشمند در دراز مدت، باید آنها را در حافظه Flash نیز کپی کنید. قبل از کپی کردن در flash، باید پروژه zip شده را هم به plc ارسال نمایید.

پروژه zip شده حاوی تمام جزئیات برنامه های شما شامل POUها، Taskها، متغیرهای عام، ساختار سخت افزار، تنظیمات cpu، اسامی سمبولیک متغیرها و حتی متون راهنماهای شماست. با توجه به اینکه تمامی این اطلاعات در فایل zip ذخیره شده اند، این فایل هنوز هم فایل بزرگی است. خوشبختانه این فایل قرار نیست که در cpu پردازش شود لذا لازم نیست در ناحیه RAM باتری دار ذخیره شود. به همین دلیل ابتدا برنامه های اصلی را ارسال می کنیم که همگی در حافظه RAM باتری دار قرار می گیرند. سپس فایل zip شده پروژه را ارسال می کنیم که در حافظه RAM بدون باتری قرار می گیرد. پس از آن با صدور فرمان Fix تمام آنها در حافظه Flash کپی می شوند. همانطور که در شکل زیر ملاحظه می شود. با خاموش و روشن کردن عادی، فایل zip در حافظه باقی نمی ماند ولی بقیه فایلها باقی می مانند. اگر می خواهید فایل zip شده را از plc باز خوانی کنید، باید عملیاتی بنام Recovery را انجام دهید.

در عملیات Recovery داده ها مطابق شکل از روی Flash به روی حافظه های RAM کپی می شوند.



- برای انجام کامل مراحل *Fixing*، مراحل زیر را انجام دهید.
۱. تمام برنامه ها را به plc ارسال کنید.
 ۲. فایل zip شده ی پروژه را نیز ارسال کنید.
 ۳. کلید "Fix Memory" را مطابق شکل زیر کلیک کنید.



Fix Memory Button

کپی کردن پروژه روی Flash فرآیند کندی است و ممکن است تا چند ده ثانیه طول بکشد. طی این عملیات، LED زرد رنگ Memory روی cpu با فرکانسهای مختلفی چشمک میزند. لطفا صبور باشید تا عملیات خاتمه یابد.

طی عملیات Flashing، برنامه های plc بدون اختلال بکار خود ادامه می دهند.

کپی کردن برنامه ها از Flash بر روی RAM

پس از عملیات Fix یا Flash کردن، برنامه های اجرایی و فایل ZippePrj در حافظه ی Flash در دراز مدت باقی می مانند.

برای آزمایش، تمام حافظه cpu را با روشی که میدانید کاملا پاک کنید. (کلید Run/Stop را در حالت Stop قرار دهید، plc را روشن کنید و در مدت ۵ ثانیه اول که led های سبز و قرمز در حال چشمک زدن هستند بیش از دو بار کلید را قطع و وصل کنید)

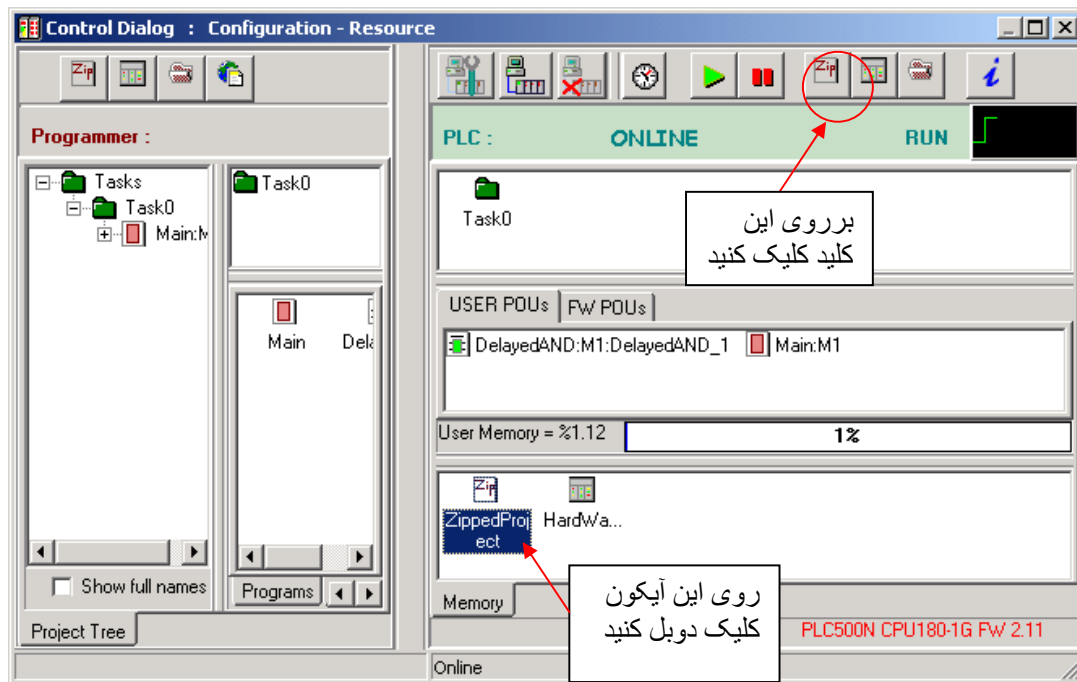
با پاک کردن حافظه، plc به stop رفته و لامپ های قرمز stop و زرد hardware روشن می شوند. این شرایط نشان می دهد که حافظه plc پاک شده است. با کامپیوتر نیز که به plc وصل شوید، خواهید دید که چیزی در حافظه plc دیده نمی شود.

مجددا plc را خاموش کنید کلید را در وضعیت stop قرار داده و مجددا روشن نمایید. در مدت ۵ ثانیه اول که led های سبز و قرمز در حال چشمک زدن هستند دقیقا دو بار کلید را قطع و وصل کنید (عملیات Recovery).

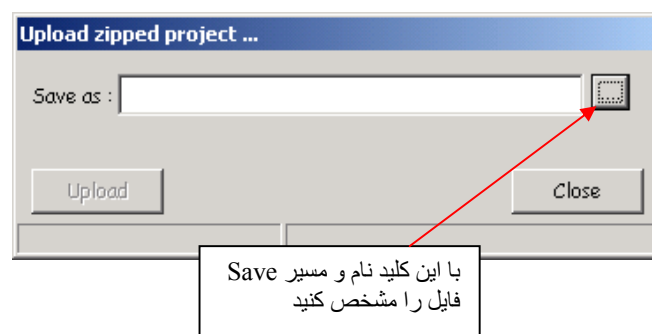
این بار plc به stop نرفته و led های خطا هم روشن نخواهد شد. یک نسخه از برنامه از حافظه flash بر روی ram کپی شده است. از کامپیوتر به plc آنلاین شده و برنامه ها را ببینید.

دریافت فایل ZippedPrj از PLC

برای دریافت فایل ZippedProject از plc، ابتدا صفحه ارتباط با plc را باز کنید.



حالا با بر روی کلیک دریافت فایل ZippedProject در بالای پنجره کلیک کنید و یا بر روی آیکون پایین کلیک دوبل کنید. به هر صورت پنجره ی زیر باز می شود.

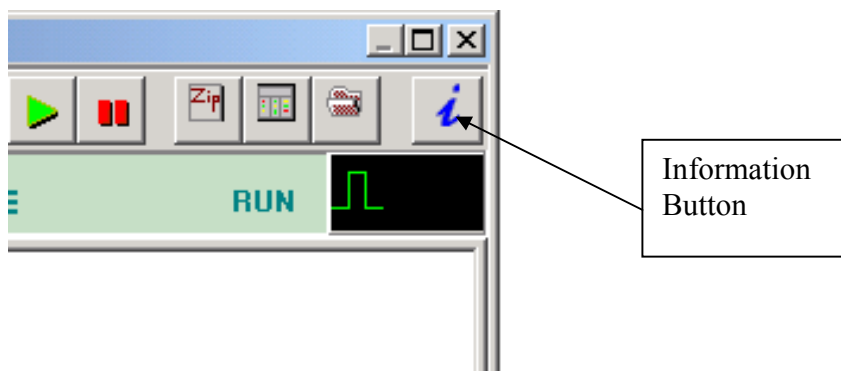


پس از مشخص کردن نام و مسیر ذخیره کردن فایل کلیک Upload را بزنید. فرایند دریافت فایل آغاز شده و به پایان میرسد.

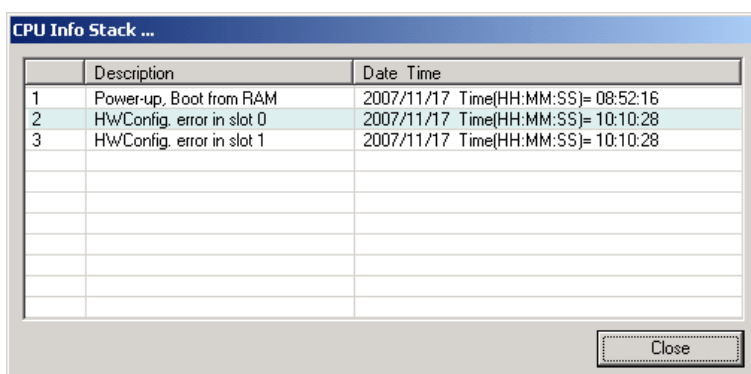
حالا می توانید فایل دریافتی را که با فرمت zwt دریافت شده را بوسیله MWT باز کنید. برای کار با این پروژه باید ابتدا آن را کامپایل کنید.

انباره گزارشات CPU Information Stack of CPU

CPU دارای انباره ای از نوع FIFO برای تهیه گزارشات کاری خود است. FIFO مخفف عبارت انگلیسی "First In First Out" انباره ای برای گزارشات cpu است که متون وقایع مهم را به ترتیبی که رخ می دهند در خود نگه میدارد. این انباره در CPU180 برای ذخیره کردن حداکثر ده متن خطا گنجایش دارد. در FIFO های معمولی چنانچه داده هایی بزرگتر از گنجایش FIFO وارد شود، به انتهای انباره وارد می شود و بقیه داده ها به بالا منتقل می شوند. بدین ترتیب اولین داده (قدیمی ترین داده) از بالای انباره حذف می شود. در انباره گزارشات CPU180 تفاوت کوچکی وجود دارد. با توجه به اینکه اولین گزارش نحوه روشن شدن CPU را منعکس میکند از اهمیت خاصی در بررسی سیستمها برخوردار است. به همین دلیل پس از روشن شدن دستگاه، گزارش اول همواره ثابت بوده و هیچگاه حذف نمی شود ولی بقیه ی گزارشات (در صورت زیاد شدن) مانند FIFO عمل می کنند. طبیعتا این گزارشات گزارش وقایع غیر طبیعی هستند که موجب اقدامات امنیتی توسط CPU می شوند مثل نامناسب بودن فایل HwConfig، خرابی سخت افزار، مشاهده خطای watchdog و غیره. محتویات انباره گزارشات cpu را با کلیک کردن روی دکمه "i" می توان مشاهده کرد. همانطوری که در شکل زیر دیده می شود.



شکل زیر نمونه ای از گزارشات cpu را نشان می دهد.



	Description	Date Time
1	Power-up, Boot from RAM	2007/11/17 Time(HH:MM:SS)= 08:52:16
2	HwConfig, error in slot 0	2007/11/17 Time(HH:MM:SS)= 10:10:28
3	HwConfig, error in slot 1	2007/11/17 Time(HH:MM:SS)= 10:10:28

گزارشات بصورت متون ساده انگلیسی همراه با تاریخ و زمان وقوع در جدول ثبت شده و نیازی به تفسیرهای پیچیده ندارند.

ضمیمه یک:

تنظیم آدرس IP کامپیوتر پروگرامر

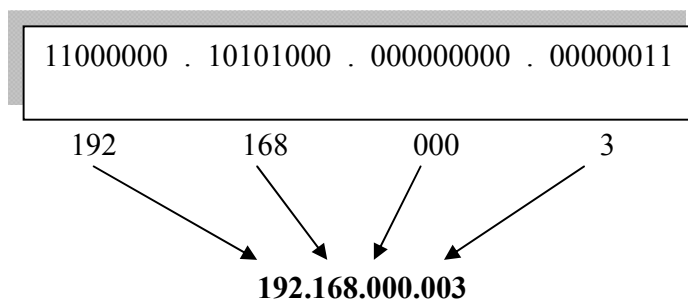
این ضمیمه شکل تنظیم آدرس IP کامپیوتر برنامه ریزی شما را بصورتی ساده نشان میدهد. فرض این است که سیستم عامل کامپیوتر شما Windows XP باشد. سایر سیستمهای عامل نیز کمابیش مشابه یکدیگر بوده و به روش مشابهی عمل می کنند.

برای اینکه کامپیوتر شما در یک شبکه محلی (LAN) حضور داشته باشد این است که کامپیوتر شما دارای یک آدرس IP منحصر به فرد در محدوده ی مشخصی باشد.

هر آدرس IP عددی ۳۲ بیتی باینری است. بنابراین حداکثر 4,294,967,296 آدرس منحصر به فرد را میتوان در یک شبکه تعریف کرد. برای اینکه انسانها آدرس های IP را بهتر بخوانند و بنویسند، آن را بصورت چهار عدد مبنای ده مینویسند که با یک نقطه از هم جدا می شوند.

این شکل نمایش آدرس ها را "dotted-decimal notation" می گویند. در این شکل نمایش، ۳۲ بیت را به چهار بایت ۸ بیتی تقسیم میکنند و هر بخش را بصورت یک عدد مستقل مبنای ۱۰ در نظر گرفته که با علامت نقطه از هم جدا شده اند.

شکل زیر یک آدرس اینترنتی را نشان می دهد که با فرمت "dotted-decimal notation" بیان شده است.



با توجه به اینکه قرار نیست که تمام وسایل مستقیماً به یک شبکه واحد وصل شوند، روش استاندارد برای پیاده سازی زیر شبکه ها یا شبکه های کوچکتر توسط سازمان های بین المللی تدوین شده است. در این استانداردها، گروه ها و کلاسهای تعریف شده اند که شرح آنها خارج از حدود این مقاله است.

آنچه که ما واقعا باید بدانیم این است که در یک شبکه کوچک LAN، تمام تجهیزات حاضر در شبکه باید دارای آدرس منحصر به فردی در شبکه باشند که سه رقم اول آنها با هم مساوی باشند. آخرین رقم را می توان آزادانه به تجهیزات مختلف اختصاص داد. برای مثال این آدرسها را می توان در یک LAN معمولی استفاده کرد.

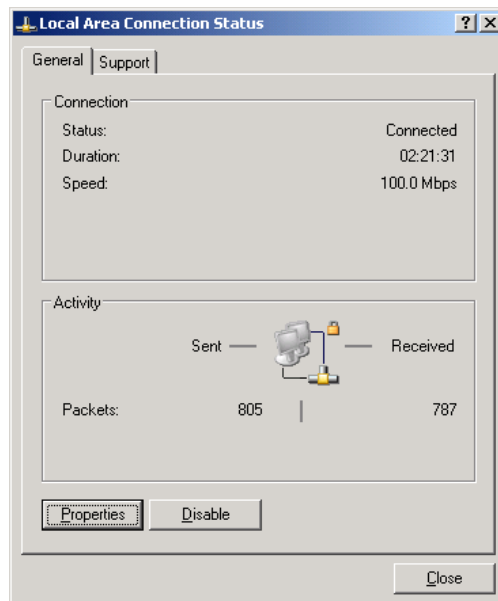
192.168.000.020, 192.168.000.011, 192,168,000,030

با دو روش میتوان آدرس IP یک کامپیوتر را تنظیم کرد. در یکی از روشها آدرس را یکبار بصورت ثابت (Static) اختصاص می دهند در روش دوم اجازه می دهند که این آدرس توسط کامپیوتر دیگری بنام سرور در هنگام روشن شدن کامپیوتر شما اختصاص داده شود. (Dynamic)

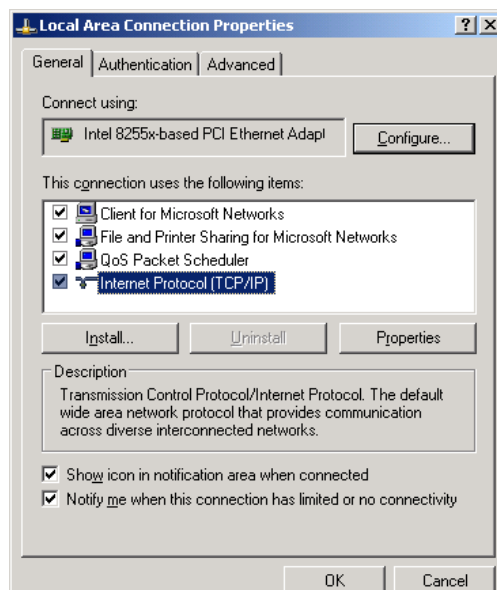
در این بحث فرض می کنیم که شبکه شما کامپیوتر Server نداشته و باید خودمان آدرس را بصورت Static وارد کنیم.

لطفا مراحل زیر را به ترتیب انجام دهید:

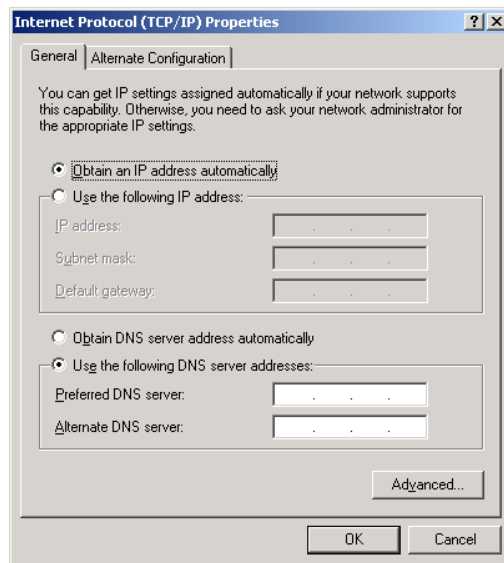
۱. به Control Panel ویندوز رفته و Network Connections را انتخاب کنید.
۲. بر روی "LAN or high speed internet" کلیک دابل کنید.
۳. پنجره شکل زیر پیدا خواهد شد.



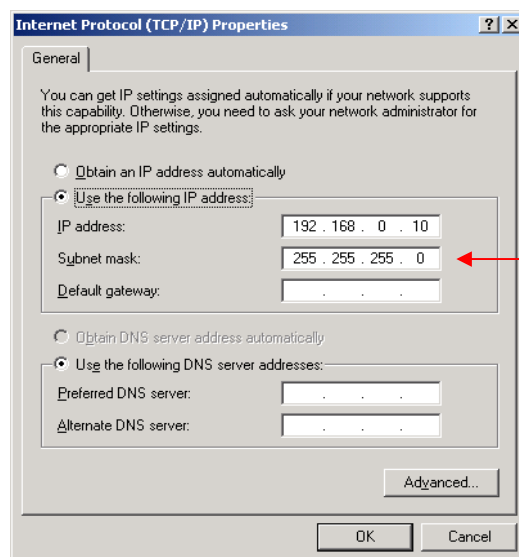
۴. روی دکمه Properties کلیک کنید. پنجره شکل زیر پیدا می شود. گزینه InternetProtocol (TCP/IP) را انتخاب کنید.



۵. دکمه Properties را کلیک کنید. اگر قبلاً آدرس استاتیکی ثبت نشده باشد، پنجره شکل زیر پیدا خواهد شد.



۶. گزینه "Use the following IP address" را فعال کرده و آدرس IP مورد نظر خود را وارد کنید.
در شکل زیر مثالی در این مورد مشاهده می کنید.



همیشه عدد
را Subnet Mask
255,255,255,0
انتخاب کنید.

۷. کلید OK را زده و کار را خاتمه دهید. آدرس IP کامپیوتر شما بصورت استاتیکی تنظیم شده است.