آموزش نرم افزار ISPSoft

برای کار با PLCهای شرکت Delta

محمد اذانگو

ISPSoft نرم افزار جدید دلتا می‌باشد. قابلیت استفاده از پنج زبان مختلف برنامه نویسی و پشتیبانی از PLC های خانواده‌های AH500 و DVP، همچنین درایوهای شرکت دلتا به همراه مجموعه کاملی از کتابخانه‌های نرم‌افزاری از جمله ویژگی‌های اصلی این نرم افزار به شمار می‌آید. محیط نرم افزاری بهینه، راحت و ساختاریافته در این نرم‌افزار کاربر را قادر می‌سازد تا بتواند با PLC های دلتا پیچیده‌ترین سیستم‌ها را به شکلی ساده کنترل نماید.

کامیاب مرام

آموزش نرم افزار ISPSoft

نگارش: محمد اذانگو

بهار ۱۳۹۴


بسم الله الرحمن الرحيم

پیشگفتار

نرم افزار ISPSoft برای برنامه نویسی به صورت ساختاریافته جهت پیاده‌سازی در PLCهای کمپانی دلتا عرضه شده است. این نرم افزار دارای ویژگی‌های منحصر به فردی از جمله پشتیبانی از ۵ زبان مختلف برنامه نویسی، شبیه ساز پیشرفته PLCهای سری AH500 و امکان تنظیم و طراحی معماری شبکه‌های صنعتی می‌باشد. در این کتاب مطابق با راهنمای ارائه شده توسط کمپانی دلتا به معرفی این نرم افزار و ویژگی‌های آن خواهیم پرداخت[1]. آموزش کار با PLCهای جدید و پیشرفته سریAH500 کمپانی دلتا و برنامه نویسی به زبان‌های(Ladder Diagrams (LD، (Instruction Lists (IL، Structured Texts (ST)، Sequential Function Chart (SFC) و Function Block Diagram (FBD) از دیگر مباحث این کتاب خواهد بود. زبان برنامه نویسی ST زبانی جدید در اتوماسیون صنعتی و بسیار شبیه به برنامه نویسی زبان C است که برای مهندسان برق، کامپیوتر و مکانیک بسیار جذاب می‌باشد. شبیه‌سازهای در نظر گرفته شده در این نرم افزار به همراه بخش NWCONFIG برای انجام تنظیمات مربوط به پیاده‌سازی شبکه، این نرم افزار را به نرم افزاری کامل تبدیل کرده است، این بخش‌ها به صورت مجزا در این کتاب مورد بررسی قرار خواهند گرفت. گنجاندن مثال‌های صنعتی و نمونه‌های قابل استفاده در محیط‌های آکادمیک از جمله دیگر مزیت‌های این کتاب به شمار می‌آید.

اما مطالعه این کتاب برای چه افرادی مناسب است، با توجه به گروه هدف مشتریان محصولات کمپانی دلتا در ایران که اکثرا تکنسین‌های فنی هستند و البته نیاز این افراد به مرور تمامی جزئیات، این کتاب سعی بر شرح تمامی نکات مورد نیاز در حین کار با ISPSoft را داشته است و اکثر مطالب آن از دستورالعمل نرم افزار به زبان انگلیسی استخراج شده است. این موضوع ممکن است گاهی برای کاربران آکادمیک و اتوماسیون‌کاران حرفه‌ای خسته کننده باشد، با این حال با اضافه شدن نمونه مسائل کاربردی و همچنین مسائل آکادمیک، سعی در جبران آن شده است. ذکر این نکته نیز ضروری است که برای استفاده بهینه از این کتاب، داشتن دانش پایه (اطلاعات اولیه) در زمینه‌های مهندسی برق، برنامه نویسی، اتوماسیون صنعتی و مدارهای منطقی و همچنین آشنایی اولیه با مفاهیم PLC ضروری است. در واقع

[1] عمده مطالب این کتاب و ساختار کلی آن برگرفته از ترجمه مستقیم دستورالعمل انگلیسی نرم افزار ISPSoft است که با توجه به نیاز مخاطب و اهداف نویسنده مطالبی جهت افزایش کارایی به آن افزوده شده است.

کتاب پیش‌رو مرجع مناسبی برای فراگیری PLC نمی‌باشد، بلکه با فرض آشنایی اولیه خواننده با مفاهیم اصلی و مقدماتی PLC[1]، مقدمات آشنایی وی را برای کار با محصولات کمپانی دلتا فراهم می‌کند.

سپاس – اذانگو،بهار۱۳۹٤

[1] با این حال سعی شده در جای جای کتاب، جهت بهره‌وری بیشتر خواننده نکات پایه و اساسی مربوط به PLCها مرور شود.

[2] لطفا موضوع ایمیل خود را ISPSoft بگذارید.

با سپاس از جناب محمد اذانگو در تهیه مجموعه ارزشمند آموزش نرم افزار ISPSoft

از خوانندگان محترم تقاضا می شود سوالات خود را در مورد هر قسمت از متن این مجموعه و یا منوال انگلیسی آن با کامیاب مرام در میان گذارند .

ارتباط با کامیاب مرام : Tech@deltakaran.com

حسین دهقان

کامیاب مرام

فهرست مطالب

عنوان صفحه

فهرست جدول‌ها ..... ی

فهرست شکل‌ها ..... م

فهرست علایم و نشانه‌ها ..... مم

فصل ۱- مقدمه ۴۲

۱-۱- پیشگفتار ۴۲

1-2- معرفی نرم افزار ISPSoft ۴۲

۱-۳- مروری بر مباحث مطرح شده در کتاب ۴۳

۱-۴- راهنمای نمادها ۴۴

فصل ۲- مقدمه کار با نرم افزار ISPSoft ۴۵

۲-۱- مقدمه ۴۵

۲-۱-۱- مروری بر ویژگی های شاخص نرم افزار ۴۵

۲-۱-۲- سیستم مورد نیاز ۴۷

2-2- نصب نرم افزار ISPSoft ۴۸

۲-۳- راه اندازی ISPSoft و مروری بر فضای کلی آن ۵۱

۲-۳-۱- نوارهای ابزار در محیط نرم افزار ISPSoft ۵۷

2-4- چارچوب کار با پروژه ها در ISPSoft ۶۰

۲-۴-۱- نحوه ایجاد پروژه منفرد جدید ۶۲

۲-۴-۲- نحوه ایجاد پروژه گروهی جدید ۶۳

۲-۴-۳- مدیریت پروژه ها ۶۷

۲-۵- تنظیمات مقدماتی در ISPSoft ۶۹

2-5-1- سربرگ Project Setting در قسمت General ۷۰

2-5-2- سربرگ Workspace در قسمت General ۷۱

2-5-3- سربرگ Output Window در قسمت General ۷۲

2-5-4- سربرگ Symbol Table در قسمت General ۷۳

2-5-5- سربرگ [FBD][LADDER] در قسمت Editor ۷۴
2-5-6- سربرگ [SFC] در قسمت Editor ۷۵
2-5-7- سربرگ [ST][IL] در قسمت Editor ۷۶
فصل ۳- پیکربندی سیستم ۷۷
۳-۱- مقدمه ۷۷
3-2- معرفی COMMGR ۷۷
۳-۲-۱- نصب نرم افزار COMMGR ۷۸
۳-۲-۲- تنظیم ارتباط ۸۱
۳-۲-۳- ساخت درایور ۸۳
۳-۲-۴- تنظیم ارتباط بین نرم افزار ISPSoft و COMMGR ۸۵
۳-۲-۵- ارتباط با PLC از طریق پورت های ارتباطی ۸۶
۳-۲-۶- تست عملی اتصال ۸۸
3-3- پیکربندی PLC در ISPSoft ۸۹
3-3-1- پیکربندی مدل های DVP ۹۰
۳-۳-۲- پیکربندی مدل های AH-500 ۹۵
3-3-3- تنظیم پارامترهای پردازنده AH500 ۱۰۲
۳-۳-۴- ذخیره، بارگزاری و استخراج تنظیمات پردازنده ۱۱۱
3-3-5- مدیریت پارامترها و عیب یابی از طریق شبکه در AH500 ۱۱۲
۳-۳-۶- اسکن ورودی و خروجی ها ۱۱۳
۳-۳-۷- عیب یابی آنلاین ۱۱۴
3-3-8- تنظیم ساعت Real-Time ۱۱۸
۳-۳-۹- تنظیم حافظه ی PLC ۱۱۹
فصل ۴- راه اندازی اولیه ۱۲۱
۴-۱- شرح مثال (خط تولید) ۱۲۱
۴-۲- الگوریتم برنامه ۱۲۲
4-3- مراحل ساخت پروژه در ISPSoft ۱۲۳
۴-۳-۱- ساخت پروژه ۱۲۴
۴-۳-۲- برنامه نویسی ۱۲۷
۴-۳-۳- بررسی و کامپایل برنامه نوشته شده ۱۳۹
4-3-4- برقراری ارتباط با PLC و بارگزاری نرم افزار ۱۴۱
۴-۳-۵- بررسی برقراری ارتباط ۱۴۴

فصل ۵- واحد سازماندهی برنامه ها ۱۴۷
۵-۱- معماری واحد سازماندهی برنامه ۱۴۷
۵-۱-۱- POU ها در ISPSoft ۱۴۸
5-1-2- عملکردهای POU در ISPSoft ۱۵۰
۵-۱-۳- فعال کردن POUها ۱۵۶
5-1-4- پاک کردن و کپی POU ۱۵۷
۵-۱-۵- رمز عبور ۱۵۹
۵-۲- مدیریت عملکردها ۱۶۰
۵-۲-۱- تنظیمات عملکردها و شرط رخ دادن وقفه ها ۱۶۲
۵-۲-۲- تخصیص عملکرد به POUها ۱۶۶
۵-۳- مثال های کاربردی POU ۱۶۸
۵-۳-۱- مثال خط تولید تزریق چسب ۱۶۸
5-3-2- مثال وقفه در PLCهای سری DVP ۱۷۳
5-3-3- مثال وقفه در PLCهای سری AH500 ..... ۱۸۱
فصل ۶- سیمبول ها ۱۹۲
۶-۱- داده ها و سیمبول ها ۱۹۲
۶-۱-۱- کلاسهای سیمبول ۱۹۳
۶-۱-۲- انواع داده ۱۹۵
۶-۱-۳- تخصیص حافظه به سیمبول و مقداردهی اولیه ۱۹۷
۶-۱-۴- آدرس دهی غیر مستقیم ۲۰۰
۶-۱-۵- سیمبول بیت های حافظه های داده ۲۰۱
6-2- مدیریت سیمبول ها در ISPSoft ۲۰۳
۶-۲-۱- جدول سیمبول ها ۲۰۳
۶-۲-۲- اضافه کردن سیمبول ۲۰۴
6-2-3- استفاده از سیمبول داده های آرایه و STRING ۲۰۹
۶-۲-۴- ویرایش سیمبول ها ۲۱۲
۶-۲-۵- دانلود مقادیر اولیه سیمبول ها ۲۱۴
۶-۲-۶- ذخیره و بازخوانی جداول سیمبول ۲۱۴
۶-۲-۷- مرتب کردن سیمبول ها ۲۱۷
۶-۲-۸- فیلتر کردن سیمبول ها ۲۱۷
6-2-9- تعیین محدوده حافظه های سیمبول برای سری DVP ۲۱۹

۶-۳- مثال ۲۲۰
۶-۳-۱- جدول سیمبول ها ۲۲۰
۶-۳-۲- تکمیل برنامه ۲۲۲
فصل ۷- بلوک ها ۲۲۶
۷-۱-۱- مقدمه ای بر توابع بلوکی ۲۲۶
۷-۱-۲- ویژگی های توابع بلوکی ۲۲۷
7-2- ساختار توابع بلوکی در ISPSoft ۲۲۸
7-2-1- پایه En در بلوک ۲۲۹
۷-۲-۲- سیمبول ها در بلوک ۲۳۰
۷-۲-۳- ورودی و خروجیهای بلوک ۲۳۱
۷-۲-۴- سیمبولهای با فرمت داده اشاره گر ۲۳۵
۷-۲-۵- مرجع و نسخه تابع بلوکی ۲۴۰
۷-۲-۶- فراخوانی توابع بلوکی ۲۴۵
۷-۲-۷- اختصاص بلوک های حافظه به توابع بلوکی ۲۴۸
۷-۳- به کارگیری توابع بلوکی ۲۵۳
۷-۳-۱- ویژگی های توابع بلوکی ۲۵۳
7-3-2- دستورات پالسی در توابع بلوکی PLC های AH500 ..... ۲۵۵
۷-۳-۳- مانیتورینگ آنلاین برنامه های نوشته شده در تابع بلوکی ۲۵۶
۷-۳-۴- تغییر برنامه تابع بلوکی ۲۵۷
۷-۴- مثال ۲۵۸
۷-۴-۱- ساخت برنامه ۲۵۹
فصل ۸- زبان برنامه نویسی Ladder ۲۷۵
۸-۱-۱- محیط برنامه نویسی ۲۷۵
8-1-2- Network در Ladder ۲۷۸
۸-۱-۳- انتخاب اجزای برنامه ۲۸۰
8-2- Ladder در نرم افزار ISPSoft ۲۸۰
۸-۲-۱- کانتکت ها و کویل ها ۲۸۰
۸-۲-۲- به کار بردن حافظه ها، سیمبول ها و مقادیر ثابت ۲۸۵
۸-۲-۳- نوشتن دستورالعمل ۲۸۶
۸-۲-۴- بلوک ها ۲۸۸
۸-۲-۵- بلوک های مقایسه ای ۲۸۹

۸-۲-۶- بلوک های منطقی ۲۹۰
8-2-7- ساخت چند خروجی در Network ۲۹۳
۸-۲-۸- اضافه کردن برچسب ۲۹۴
۸-۲-۹- ویرایش توضیحات ۲۹۵
۸-۲-۱۰- حالت نمایش سیمبول / آدرس ۲۹۸
۸-۲-۱۱- نشانه گزاری ۲۹۹
8-2-12- فعال و غیرفعال کردن Network ۳۰۰
فصل ۹- زبان برنامه نویسی Structured Text ۳۰۲
۹-۱-۱- ساختار Structured Text ۳۰۲
9-1-2- Statement ۳۰۳
9-1-3- Expression ۳۰۴
۹-۱-۴- عملوند و عملگرها ۳۰۶
۹-۱-۵- کلمات کلیدی و توضیحات ۳۰۸
۹-۱-۶- استفاده از سیمبول های با فرمت آرایه ۳۰۹
9-1-7- نکات مهم پیرامون ST ۳۰۹
۹-۲- ساختار Statement ۳۱۲
۹-۲-۱- تخصیص دهی ۳۱۲
۹-۲-۲- دستور شرطی IF ۳۱۵
۹-۲-۳- دستور شرطی CASE ۳۱۸
۹-۲-۴- دستور حلقه REPEAT ۳۲۱
۹-۲-۵- دستور حلقه WHILE ۳۲۳
۹-۲-۶- دستور حلقه FOR ۳۲۵
۹-۲-۷- استفاده از دستورالعمل ها ۳۲۸
۹-۲-۸- استفاده از Function Block ها ۳۲۹
۹-۲-۹- دستور تاخیر ۳۳۲
۹-۲-۱۰- دستور RETURN ۳۳۲
۹-۲-۱۱- دستور EXIT ۳۳۳
9-3- ساخت برنامه به زبان ST ۳۳۵
۹-۳-۱- استفاده از بلوک ها و دستورالعمل ها ۳۳۸
۹-۳-۲- نشان گزاری ۳۳۹
9-4- مثال برنامه نویسی ST ۳۴۰

۹-۴-۱- شرح برنامه ۳۴۰
۹-۴-۲- تنظیمات سخت افزاری ۳۴۱
۹-۴-۳- عملکرد برنامه ۳۴۲
۹-۴-۴- ساخت برنامه ۳۴۲
فصل ۱۰- زبان های برنامه نویسیFBD ، IL و SFC ۳۴۷
10-1- زبان برنامه نویسی FBD ۳۴۷
۱۰-۱-۱- محیط برنامه نویسی ۳۴۸
10-1-2- Network در FBD ۳۵۰
۱۰-۱-۳- انتخاب اجزای برنامه ۳۵۰
۱۰-۱-۴- تابع بلوکی ۳۵۲
10-1-5- بلوک های منطقی زبان FBD ۳۵۵
10-2- زبان برنامه نویسی IL ۳۶۴
۱۰-۲-۱- فراخوانی توابع ۳۶۷
۱۰-۲-۲- ساخت پروژه ۳۶۸
10-2-3- نکات مهم در برنامه نویسی IL ۳۷۰
۱۰-۳- زبان برنامه نویسی SFC ۳۷۲
10-3-1- مرور مفاهیم زبان برنامه نویسیSFC در ISPSoft ۳۷۵
۱۰-۳-۲- ساخت SFC در ISPSoft ۳۹۴
فصل ۱۱- شبیه ساز PLC ، تِست و عیب یابی ۴۱۰
۱۱-۱- مانیتورینگ آنلاین ۴۱۰
۱۱-۱-۱- تغییر وضعیت ورودی ها ۴۱۶
۱۱-۱-۲- مانیتورینگ آنلاین برنامه ۴۱۸
۱۱-۱-۳- جدول مانیتورینگ ۴۲۳
۱۱-۱-۴- ویرایش آنلاین برنامه ۴۲۸
11-2- شبیه سازهای DVP ۴۲۹
۱۱-۳- حالت عیب یابی برای PLCهای سری DVP ۴۳۱
۱۱-۳-۱- اضافه و حذف نقطه انفصال ۴۳۳
۱۱-۳-۲- اجرای برنامه در حالت عیب یابی ۴۳۳
11-4- شبیه ساز خانواده AH500 ..... ۴۳۵
11-4-1- پنجره شبیه ساز AH500 ..... ۴۳۷
۱۱-۵- عیب یابی در AH500 ..... ۴۴۳

۱۱-۵-۱- نقطه انفصال ۴۴۴
۱۱-۵-۲- اجرای پیوسته ۴۴۵
۱۱-۵-۳- اجرای مرحله به مرحله ۴۴۶
11-6- گزارش عملکرد PLC (فقط برای مدل های AH500) ۴۴۷
۱۱-۶-۱- سربرگ Error Log ۴۴۸
11-6-2- سربرگ Program Change Log ۴۴۹
11-6-3- سربرگ Status Change Log ۴۴۹
فصل ۱۲- ابزارها و امکانات جانبی ISPSoft ۴۵۱
۱۲-۱- ابزارهای اصلاح و توابع کمکی در ISPSoft ۴۵۱
۱۲-۱-۱- تغییر مدل PLC ۴۵۱
۱۲-۱-۲- بارگزاری و استخراج پروژه ۴۵۲
12-1-3- Find/Replace/Goto در Ladder و FBD ۴۵۴
12-1-4- Find/Replace در Instruction List و Structured Text ۴۵۷
12-1-5- Find/Replace در SFC برای STEP و Transition ۴۵۹
۱۲-۱-۶- جستجو در میان سیمبول ها ۴۶۱
۱۲-۱-۷- Find/Replace حافظه ها و سیمبول ها ۴۶۲
12-1-8- چاپ کردن پروژه ۴۶۴
۱۲-۲- مدیریت حافظه ها ۴۶۷
12-2-1- لیست توضیحات حافظه های برای PLCهای سری DVP ۴۶۷
12-2-2- گزارش حافظه های استفاده شده در PLCهای مدل DVP ۴۷۲
۱۲-۲-۳- مدیریت حافظه ها در PLCهای سری AH500 ۴۷۵
12-2-4- اصلاح مقادیر حافظه های T/C/D در PLCهای سری DVP ۴۷۹
12-2-5- اصلاح مقادیر حافظه های M/S در PLCهای سری DVP ۴۸۶
12-2-6- اصلاح مقادیر حافظه های فایل در PLCهای سری DVP ۴۹۱
۱۲-۳- رمزگذاری و محافظت از داده ها ۴۹۸
۱۲-۳-۱- شناسه برنامه و PLC ۵۰۰
12-3-2- رمز عبور پروژه و PLC ۵۰۳
۱۲-۳-۳- رمز عبور POU ۵۰۸
۱۲-۳-۴- رمز عبور Subroutin ۵۱۰
۱۲-۳-۵- روش های دیگر محافظت و رمز گزاری ۵۱۲
۱۲-۴- پشتیبان گیری از داده ها ۵۱۴

۱۲-۴-۱- استفاده از کارت حافظه پشتیبان ۵۱۵
۱۲-۴-۲- پشتیبان گیری دائم ۵۱۶
12-4-3- CARD Utility (برای PLCهای سری AH500) ۵۱۶
۱۲-۵- ارتباط USB ۵۲۹
۱۲-۵-۱- نصب درایور USB ۵۲۹
۱۲-۵-۲- ساخت درایور USB در COMMGR ۵۳۴
12-5-3- تنظیمات USB برای PLCهای مدل DVP-SX2 ۵۳۶
12-6- نکات مهم در ارتباط با PLCهای سری AH500 ۵۳۷
۱۲-۶-۱- آدرس حافظه ها و پورت ها در PLCهای AH500 ۵۳۷
12-6-2- کار با بیت ها در پورت و حافظه های نوع X/Y/D/L ۵۳۹
۱۲-۶-۳- به روز رسانی مستقیم ورودی/خروجی ۵۴۱
۱۲-۶-۴- محدوده حافظه ها در PLCهای AH500 ۵۴۲
۱۲-۶-۵- محدوده حافظه ها در PLCهای DVP ۵۴۶
فصل ۱۳- تنظیمات شبکه و تبادل داده ۵۴۸
13-1- تنظیمات شبکه با استفاده از NWCONFIG ۵۴۸
۱۳-۱-۱- اطلاعات اولیه ۵۵۰
۱۳-۱-۲- تنظیمات ارتباطی در NWCONFIG ۵۵۳
۱۳-۱-۳- مراحل پیاده سازی شبکه ۵۵۷
۱۳-۲- ساخت معماری شبکه و دیگر نکات مربوط به NWCONFIG ۵۶۳
۱۳-۳- دیگر امکانات شبکه دلتا ۵۶۸
ضمیمه أ - مروری بر خانواده های DVP و AH500 ۵۷۱
ضمیمه ب - نمونه مسائل ۵۹۰
ضمیمه ج - مروری بر دستورالعمل های متداول ۵۹۱
ضمیمه د -مروری بر حافظه های خاص PLCهای دلتا ۶۰۵
ضمیمه ه - پیکربندی ماژول DVPEN01 ۶۴۶
ساخت درایور در COMMGR برای شبکه Ethernet ۶۵۸
تنظیم ارتباط بین نرم افزار ISPSoft و COMMGR ۶۵۹
فهرست مراجع ۶۸۶

واژه نامه فارسی به انگلیسی ۶۸۷
واژه نامه انگلیسی به فارسی ۶۸۹

فهرست جدول‌ها

عنوان صفحه

جدول ۱-۱: نمادهای استفاده شده در کتاب ۴۴
جدول ۲-۱: سیستم مورد نیاز برای کار با ISPSoft ۴۷
جدول ۲-۲: فرمت فایل هایی که ISPSoft پشتیبانی میکند ۶۸
جدول ۳-۱: مشخصات پیشفرض ارتباط سریال در خانواده AH500 ..... ۸۸
جدول ۳-۲: مشخصات پیشفرض ارتباط از طریق Ethernet در خانواده AH500 ..... ۸۸
جدول ۴-۱: جدول اختصاص ورودی و خروجی –مثال خط تولید ۱۲۲
جدول ۴-۲: آیکونهای نوار ابزار برنامه نویسی Ladder ۱۲۹
جدول ۴-۳: روش های مانیتورینگ ۱۴۴
جدول ۵-۱: آیکون های POU در بخش مدیریت پروژه ۱۴۹
جدول ۵-۲: عملکردهای تخصیص داده شده به POU ۱۵۲
جدول ۵-۳: مثال۱- پروژه ای با چهار POU ۱۵۴
جدول ۵-۴: تعریف POU برای مثال ۲ ..... ۱۷۳
جدول ۵-۵: مثال۳ – POU های پروژه ۱۸۲
جدول ۶-۱: انواع داده در ISPSoft ۱۹۵
جدول ۶-۲: قواعد تخصیص سیمبول در PLC های سری های گوناگون ۱۹۷
جدول ۶-۳: حافظه های قابل تخصیص به انواع مختلف داده ۱۹۸
جدول ۶-۴: حافظه های اختصاص داده شده ۲۱۰
جدول ۶-۵: نحوه تخصیص مقادیر اولیه به درایه آرایه ها ۲۱۱
جدول ۶-۶: مثال: اختصاص سیمبول ۲۲۰
جدول ۷-۱: انواع کلاس سیمبول ها در توابع بلوکی ۲۳۰
جدول ۷-۲: انواع داده در توابع بلوکی ۲۳۰
جدول ۷-۳: اختصاص حافظه در توابع بلوکی ۲۳۱
جدول ۷-۴: نحوه تطابق داده های حافظه ورودی/خروجی و پایه متناظر آن ۲۳۳
جدول ۷-۵ محدودیت انواع مختلف اشاره گرها ۲۳۶
جدول ۷-۶: مثالی از تعریف نسخه های توابع بلوکی در جدول سیمبول های محلی و سراسری ۲۴۶
جدول ۷-۷: بررسی سیمبولهای محلی ذخیره شده درتابع بلوکی (مثال) ۲۵۱

جدول ۷-۸: ویژگی های توابع بلوکی ۲۵۳
جدول ۷-۹: مرور POUهای برنامه ساخته شده در فصل قبل ۲۵۸
جدول ۸-۱: آیکون های نوار ابزار برنامه نویسی Ladder ۲۷۷
جدول ۸-۲: کانتکت و کویل های نرم افزار ISPSoft ۲۸۰
جدول ۸-۳: فرمت مقادیر ثابت در ISPSoft ۲۸۵
جدول ۸-۴: بلوک های منطقی ۲۹۱
جدول ۹-۱: آیکون های نوار ابزار برنامه نویسی Ladder ۳۰۷
جدول ۹-۲: فرمت اعداد در ISPSoft ۳۱۰
جدول ۹-۳: دستورالعمل هایی که ST پشتیبانی نمی کند ۳۱۱
جدول ۹-۴: مثال هایی از انواع تخصیص دهی ناصحیح ۳۱۴
جدول ۹-۵: آیکون های نوار ابزار برنامه نویسی ST ۳۳۶
جدول ۹-۶: وضعیت های سیستم اصلی و عملکرد سیستم روشنایی ۳۴۰
جدول ۹-۷: تخصیص ورودی و خروجی های سیستم ۳۴۱
جدول ۱۰-۱: آیکون های نوار ابزار برنامه نویسی FBD ۳۴۹
جدول ۱۰-۲: جدول درستی گیت AND ۳۵۵
جدول ۱۰-۳: جدول درستی گیت OR ۳۵۶
جدول ۱۰-۴: جدول درستی گیت NOT ۳۵۸
جدول ۱۰-۵: آیکونهای نوار ابزار برنامه نویسی IL ۳۶۹
جدول ۱۰-۶: فرمت اعداد ثابت در زبان برنامه نویسی IL در ISPSoft ۳۷۰
جدول ۱۰-۷: آیکون های نوار ابزار برنامه نویسی SFC ۳۹۵
جدول ۱۱-۱: روش های مانیتورینگ ۴۱۱
جدول ۱۱-۲: گزینه های تغییر وضعیت کانتکت ها در حالت آنلاین ۴۱۹
جدول ۱۱-۳: ستون های جدول مانیتورینگ ۴۲۷
جدول ۱۱-۴: محدودیت های زبان های مختلف برنامه نویسی برای ویرایش برنامه ۴۲۸
جدول ۱۱-۵: انواع ورودی آنالوگ قابل اعمال در شبیه سازAH500 ..... ۴۴۰
جدول ۱۲-۱: گزینه های پنجره بارگزاری و استخراج ۴۵۳
جدول ۱۲-۲: گزینه های صفحه جستجو و جایگزینی در زبان برنامه نویسی Ladder و FBD ۴۵۶
جدول ۱۲-۳: گزینه های صفحه جستجو و جایگزینی در زبان برنامه نویسی IL و ST ۴۵۸
جدول ۱۲-۴: گزینه های صفحه تایید عملکرد Replace در زبان برنامه نویسی IL و ST ۴۵۸
جدول ۱۲-۵: گزینه های صفحه جستجو و جایگزینی در زبان برنامه نویسیSFC ۴۶۱

جدول ۱۲-۶: گزینه های پنجره جستجوی سیمبول ها ۴۶۲
جدول ۱۲-۷: محتوای پرینت ۴۶۶
جدول ۱۲-۸: شرح روشهای متعدد رمزگذاری در ISPSoft ۴۹۹
جدول ۱۲-۹: رمز عبور در زمان پشتیبان گیری ۵۲۳
جدول ۱۲-۱۰: رمز عبور حین بازیابی ۵۲۸
جدول ۱۲-۱۱: محل ذخیره درایورها ۵۳۰
جدول ۱۲-۱۲: انواع حافظه و پورت در PLCهای AH500 ..... ۵۳۷
جدول ۱۲-۱۳: توابع مجاز به روز رسانی مستقیم در پورتهای ورودی و خروجی ۵۴۱
جدول ۱۲-۱۴: محدوده حافظه های AHCPU500-EN/AHCPU500-RS2 ..... ۵۴۲
جدول ۱۲-۱۵: محدوده حافظه های AHCPU510-EN/AHCPU510-RS2 ..... ۵۴۳
جدول ۱۲-۱۶: محدوده حافظه های AHCPU520-EN/AHCPU520-RS2 ..... ۵۴۴
جدول ۱۲-۱۷: محدوده حافظه های AHCPU520-EN/AHCPU520-RS2 ..... ۵۴۴
جدول ۱۲-۱۷: جدول سراسری مقایسه حافظه های PLCهای سری AH500 ..... ۵۴۵
جدول ۱۲-۱۷: جدول سراسری مقایسه حافظه های PLCهای سری DVP ۵۴۶

فهرست شکل‌ها

عنوان صفحه

شکل ۲-۱ اجرای فایل Setup نرم افزار ISPSoft ۴۸
شکل ۲-۲ صفحه ابتدایی Setup ۴۸
شکل ۲-۳ تعیین نام کاربر و سازمان در Setup ۴۹
شکل ۲-۴ تنظیم آدرس نصب در Setup ۵۰
شکل ۲-۵ بررسی نهایی اطلاعات در Setup ۵۰
شکل ۲-۶ تکمیل پروسه نصب در Setup ۵۰
شکل ۲-۷ اتمام موفقیت آمیز نصب نرم افزار ISPSoft ۵۱
شکل ۲-۸ آیکون نرم افزار ISPSoft در Desktop ۵۱
شکل ۲-۹ اجرای نرم افزار ISPSoft و نمای کلی آن ۵۲
شکل ۲-۱۰ کلیک بر روی آیکون New برای ایجاد پروژه جدید ۵۲
شکل ۲-۱۱ مشخصات مورد نیاز برای ایجاد پروژه جدید ۵۳
شکل ۲-۱۲ بخش پیام و مدیریت پروژه در ISPSoft ۵۴
شکل ۲-۱۳ فعال کردن نمایش بخش پیام و مدیریت پروژه در ISPSoft ۵۵
شکل ۲-۱۴ نوشتن یک برنامه جدید در ISPSoft ۵۵
شکل ۲-۱۵ تنظیم مشخصات برنامه جدید در ISPSoft ۵۶
شکل ۲-۱۶ محیط برنامه نویسی در ISPSoft ۵۶
شکل ۲-۱۷ محیط ISPSoft پس از ایجاد پروژه و اولین برنامه در آن ۵۶
شکل ۲-۱۸ محیط ISPSoft و نوارهای ابزار آن ۵۷
شکل ۲-۱۹ نوار وضعیت ۵۸
شکل ۲-۲۰ نوار منو ۵۸
شکل ۲-۲۱ فعالسازی نوارهای ابزار ۵۹
شکل ۲-۲۲ نوار فایل ۵۹
شکل ۲-۲۳ نوار ویرایش ۵۹
شکل ۲-۲۴ نوار PLC ۵۹
شکل ۲-۲۵ نوار عیب یابی ۶۰
شکل ۲-۲۶ نوار برنامه نویسی ۶۰

شکل ۲-۲۷ پروژه منفرد با پسوند isp.* ۶۱
شکل ۲-۲۸ پروژه گروهی با پسوند pri.* ۶۱
شکل ۲-۲۹ ایجاد و تنظیمات پروژه منفرد جدید ۶۲
شکل ۲-۳۰ پروژه منفرد در بخش مدیریت پروژه ۶۲
شکل ۲-۳۱ ایجاد پروژه گروهی جدید ۶۳
شکل ۲-۳۲ تنظیمات پروژه گروهی جدید ۶۳
شکل ۲-۳۳ بخش مدیریت پروژه، اولین پروژه در یک پروژه گروهی ۶۴
شکل ۲-۳۴ اضافه کردن پروژه به پروژه گروهی- روش اول ۶۴
شکل ۲-۳۵ اضافه کردن پروژه به پروژه گروهی- روش دوم ۶۵
شکل ۲-۳۶ نحوه تنظیمات پروژه جدید در پروژه گروهی ۶۵
شکل ۲-۳۷ فعال کردن پروژه در پروژه ای گروهی ۶۶
شکل ۲-۳۸ اضافه کردن پروژه به پروژه ای گروهی ۶۶
شکل ۲-۳۹ حذف پروژه در پروژه ای گروهی ۶۷
شکل ۲-۴۰ ذخیره پروژه ۶۸
شکل ۲-۴۱ صفحه تنظیمات ISPSoft ۶۹
شکل ۲-۴۲ صفحه تنظیمات Project Setting-ISPSoft ۷۰
شکل ۲-۴۳ صفحه تنظیمات Workspace-ISPSoft ۷۱
شکل ۲-۴۴ صفحه تنظیمات Output Window-ISPSoft ۷۲
شکل ۲-۴۵ صفحه تنظیمات Symbol Table-ISPSoft ۷۳
شکل ۲-۴۶ صفحه تنظیمات [LADDER] [FBD]-ISPSoft ۷۴
شکل ۲-۴۷ صفحه تنظیمات [SFC]-ISPSoft ۷۵
شکل ۲-۴۸ صفحه تنظیمات [IL] [ST]-ISPSoft ۷۶
شکل ۳-۱ بلوک دیاگرام نحوه ارتباط سخت افزار و نرم افزار به وسیله COMMGR ۷۷
شکل ۳-۲ پنجره COMMGR و درایورهای تعریف شده در آن ۷۸
شکل ۳-۳ اجرای فایل Setup برنامه COMMGR ۷۹
شکل ۳-۴ صفحهی ابتدایی Setup برنامه COMMGR ۷۹
شکل ۳-۵ تنظیم نام کاربر و سازمان در Setup برنامه COMMGR ۸۰
شکل ۳-۶ بررسی نهایی اطلاعات در Setup برنامه COMMGR ۸۰
شکل ۳-۷ پایان مراحل نصب برنامه COMMGR ۸۱
شکل ۳-۸ ارتباط ISPSoft با PLC با کمک برنامه COMMGR ۸۱

شکل ۳-۹ ارتباط ISPSoft با PLC با کمک برنامه COMMGR ۸۲
شکل ۳-۱۰ پنجره برنامه COMMGR به همراه درایورهای آن ۸۲
شکل ۳-۱۱ ایجاد خطا در دسترسی به پورت USB در برنامه COMMGR ۸۳
شکل ۳-۱۲ ایجاد پروژه جدید در برنامه COMMGR ۸۴
شکل ۳-۱۳ انواع پروتکل ارتباطی در COMMGR ۸۵
شکل ۳-۱۴ تنظیم ارتباط ISPSoft با PLC با کمک برنامه COMMGR ۸۶
شکل ۳-۱۵ اطلاعات درایور COMMGR در نوار وضعیت ISPSoft ۸۶
شکل ۳-۱۶ سیم بندی ارتباط سریال کامپیوتر با PLC ۸۷
شکل ۳-۱۷ ارتباط مستقیم و یا با استفاده از Hub برای ارتباط از طریق Ethernet ۸۷
شکل ۳-۱۸ برقراری ارتباط بین ISPSoft با PLC ۸۹
شکل ۳-۱۹ مدیریت پروژه خانواده DVP در نرم افزار ISPSoft ۹۰
شکل ۳-۲۰ Retentive Range برای خانواده DVP ۹۱
شکل ۳-۲۱ تعیین حافظه نگهدار در Retentive Range ۹۱
شکل ۳-۲۲ بارگزاری تنظیمات Retentive Range در PLC ۹۲
شکل ۳-۲۳ بارگزاری تنظیمات Retentive Range در PLC بعد از ذخیره سازی ۹۲
شکل ۳-۲۴ انتخاب Device Resource Allocation ۹۳
شکل ۳-۲۵ تنظیمات Device Resource Allocation ۹۴
شکل ۳-۲۶ نمایش اطلاعات PLC با کلیک بر روی Connected Information – قسمت اول ۹۴
شکل ۳-۲۷ نمایش اطلاعات PLC با کلیک بر روی Connected Information – قسمت دوم ۹۵
شکل ۳-۲۸ صفحه HWCONFIG برای تنظیمات شخت افزاری خانواده ۹۶
شکل ۳-۲۹ کتابخانه ماژول های همراه AH500 ..... ۹۷
شکل ۳-۳۰ Rackها و Slotها ۹۷
شکل ۳-۳۱ اضافه کردن ماژول جدید از Product List ۹۸
شکل ۳-۳۲ اضافه کردن ماژول جدید در System Configuration area و یا Information List–قسمت اول ۹۹
شکل ۳-۳۳ اضافه کردن ماژول جدید در System Configuration area و یا Information List–قسمت دوم ۹۹
شکل ۳-۳۴ آدرس ورودی و خروجی ها در پیکربندی ۱۰۰
شکل ۳-۳۵ نحوه تغییر آدرس ورودی و خروجی ها در پیکربندی ۱۰۱
شکل ۳-۳۶ اضافه کردن Rack جدید ۱۰۲

شکل ۳-۳۷ محل تنظیم پارامترهای پردازنده ۱۰۲
شکل ۳-۳۸ پنجره تنظیمات CPU ۱۰۳
شکل ۳-۳۹ تنظیمات CPU – زیربرگ NAME ۱۰۴
شکل ۳-۴۰ اختصاص نام انحصاری بر روی سخت افزار های مشابه ۱۰۴
شکل ۳-۴۱ تنظیمات CPU – زیربرگ System ۱۰۵
شکل ۳-۴۲ Clear Non-latched Devices ۱۰۵
شکل ۳-۴۳ Y State ۱۰۵
شکل ۳-۴۴ Reset & Clear Button ۱۰۶
شکل ۳-۴۵ Watchdog Timeout ۱۰۶
شکل ۳-۴۶ Enable Constant Scan ۱۰۶
شکل ۳-۴۷ Interval Interrupt Time ۱۰۷
شکل ۳-۴۸ Enable Remote Run ۱۰۷
شکل ۳-۴۹ Constant Communication Response ۱۰۷
شکل ۳-۵۰ Error Log Location ۱۰۸
شکل ۳-۵۱ CPU Operation at Program Error and Bus Fault ۱۰۸
شکل ۳-۵۲ محدوده حافظه های نگهدارنده ۱۰۹
شکل ۳-۵۳ تنظیمات پورت سریال در CPU ۱۰۹
شکل ۳-۵۴ تنظیمات پایه Ethernet در CPU ۱۱۰
شکل ۳-۵۵ رخ دادن خطا در هنگام بارگزاری پروژه ۱۱۱
شکل ۳-۵۶ لیست سخت افزارهای موجود در پروژه ۱۱۲
شکل ۳-۵۷ اطلاعات Rackها در پروژه ۱۱۳
شکل ۳-۵۸ ارتباط ISPSoft با PLC با کمک برنامه COMMGR ۱۱۴
شکل ۳-۵۹ Scan سخت افزار ۱۱۴
شکل ۳-۶۰ بررسی مطابقت سخت افزاری در ارتباط آنلاین ۱۱۵
شکل ۳-۶۱ وضعیت آنلاین تنظیمات سخت افزاری سیستم ۱۱۵
شکل ۳-۶۲ اطلاعات آنلاین ماژول ۱۱۶
شکل ۳-۶۳ مشکلات فعلی و قبلی ماژولها ۱۱۶
شکل ۳-۶۴ تغییر وضعیت اجزای سیستم ۱۱۷
شکل ۳-۶۵ Force کردن مقدار ورودی و خروجی ها ۱۱۷
شکل ۳-۶۶ مشاهده وضعیت پارامترهای سیستم ۱۱۸

شکل ۳-۶۷ تنظیم زمان PLC ۱۱۹
شکل ۳-۶۸ استفاده از ساعت کامپیوتر و یا زمان دلخواه برای PLC ۱۱۹
شکل ۳-۶۹ پاک کردن حافظه سیستم – قسمت اول ۱۲۰
شکل ۳-۷۰ پاک کردن حافظه سیستم – قسمت دوم ۱۲۰
شکل ۳-۷۱ ریست کردن CPU ۱۲۰
شکل ۴-۱ نمای مثال خط تولید ۱۲۱
شکل ۴-۲ مراحل انجام پروژه در ISPSoft ۱۲۳
شکل ۴-۳ مراحل انجام پروژه نمونه خط تولید ۱۲۴
شکل ۴-۴ ساخت پروژه جدید – مثال خط تولید ۱۲۴
شکل ۴-۵ صفحه تنظیمات سخت افزاری – مثال خط تولید ۱۲۵
شکل ۴-۶ اضافه شدن ماژول ورودی/خروجی – مثال خط تولید ۱۲۶
شکل ۴-۷ تغییر آدرس ورودی/خروجی – مثال خط تولید ۱۲۶
شکل ۴-۸تنظیم پارامترهای CPU – مثال خط تولید ۱۲۷
شکل ۴-۹ ایجاد برنامه جدید در پروژه – مثال خط تولید ۱۲۸
شکل ۴-۱۰ انتخاب زبان برنامه نویسی Ladder – مثال خط تولید ۱۲۸
شکل ۴-۱۱ محیط برنامه نویسی – مثال خط تولید ۱۲۹
شکل ۴-۱۲ نوار ابزار برنامه نویسی Ladder ۱۲۹
شکل ۴-۱۳ اضافه کردن کانتکت ۱۳۱
شکل ۴-۱۴ انتخاب نوع کانتکت ۱۳۱
شکل ۴-۱۵ اضافه کردن کویل ۱۳۲
شکل ۴-۱۶ انتخاب نوع کویل ۱۳۲
شکل ۴-۱۷ آدرس دهی کانتکت و کویل ۱۳۳
شکل ۴-۱۸ اضافه شدن Network جدید ۱۳۴
شکل ۴-۱۹ برنامه مثال خط تولید – مرحله ۱ ..... ۱۳۴
شکل ۴-۲۰ برنامه مثال خط تولید – مرحله ۲ ..... ۱۳۵
شکل ۴-۲۱ اضافه کردن توضیحات المان ها ۱۳۵
شکل ۴-۲۲ اضافه کردن توضیحات Network ۱۳۵
شکل ۴-۲۳ برنامه مثال خط تولید – مرحله ۳ ..... ۱۳۶
شکل ۴-۲۴ :کتابخانه APIs ۱۳۷
شکل ۴-۲۵ اضافه کردن بلوک از کتابخانه به برنامه ۱۳۷

شکل ۴-۲۶ اضافه کردن بلوک – روش دوم ۱۳۸
شکل ۴-۲۷ برنامه مثال خط تولید – مرحله ۴ ..... ۱۳۸
شکل ۴-۲۸ برنامه مثال خط تولید – مرحله ۵ ..... ۱۳۸
شکل ۴-۲۹ برنامه نهایی مثال خط تولید ۱۳۹
شکل ۴-۳۰ بررسی برنامه نوشته شده ۱۴۰
شکل ۴-۳۱ کامپایل کردن ۱۴۰
شکل ۴-۳۲ فعال کردن درایور در COMMGR- مثال خط تولید ۱۴۱
شکل ۴-۳۳ برقراری ارتباط نرم افزار با درایور - مثال خط تولید ۱۴۱
شکل ۴-۳۴ بررسی اتصال نرم افزار به PLC - مثال خط تولید ۱۴۲
شکل ۴-۳۵ اجرای HWCONFIG- مثال خط تولید ۱۴۲
شکل ۴-۳۶ تنظیمات سخت افزاری - مثال خط تولید ۱۴۳
شکل ۴-۳۷ بارگزاری تنظیمات سخت افزاری - مثال خط تولید ۱۴۳
شکل ۴-۳۸ دانلود برنامه نوشته شده - مثال خط تولید ۱۴۴
شکل ۴-۳۹ تغییر وضعیت سیستم به حالت آنلاین ۱۴۵
شکل ۴-۴۰ نوار وضعیت در حالت آنلاین ۱۴۵
شکل ۵-۱ نحوه عملکرد برنامه ها در ISPSoft ۱۴۷
شکل ۵-۲ لیست POU در بخش مدیریت پروژه ۱۴۹
شکل ۵-۳ بخش های مختلف نام POU ۱۴۹
شکل ۵-۴ POU هایی که به آنها عملکرد (Tsak) های متفاوتی تخصیص داده شده ۱۵۲
شکل ۵-۵ عملکردهای تخصیص داده شده به برنامه ها ۱۵۳
شکل ۵-۶ استفاده از بلوک E1 برای فعال کردن وقفه ۱۵۴
شکل ۵-۷ مثال۱- برنامه های چهار POU ۱۵۵
شکل ۵-۸ مثال۱- برنامه معادل چهار POU ۱۵۶
شکل ۵-۹ در نظر گرفتن مراحل تست برای ارزیابی سیستم ۱۵۶
شکل ۵-۱۰ غیرفعال کردن POU مرتبط با تست پس از پایان ارزیابی سیستم ۱۵۶
شکل ۵-۱۱ POU غیرفعال ۱۵۷
شکل ۵-۱۲ حذف POU ۱۵۸
شکل ۵-۱۳ هشدار در مورد اینکه در صورت حذف شدن POU دیگر قابل بازگردانی نیست ۱۵۸
شکل ۵-۱۴ کپی کردن POU ۱۵۸
شکل ۵-۱۵ الحاق POU کپی شده ۱۵۹

شکل ۵-۱۶ نحوه فعال کردن رمز عبور برای POU –قسمت اول ۱۵۹
شکل ۵-۱۷ نحوه فعال کردن رمز عبور برای POU –قسمت دوم ۱۶۰
شکل ۵-۱۸ وارد کردن رمز عبور برای POU ۱۶۰
شکل ۵-۱۹ مدیریت عملکرد POU ۱۶۱
شکل ۵-۲۰ پنجره مدیریت عملکرد POU ۱۶۱
شکل ۵-۲۱ انتخاب نوع عملکرد و شرح آن ۱۶۲
شکل ۵-۲۲ لیست وقفه های DVP-SV ۱۶۲
شکل ۵-۲۳ عملکرد وقفه External Interruption (X0) در DVP-SV ۱۶۳
شکل ۵-۲۴ عملکرد دوره ای صفر در AHCPU503-EN ۱۶۳
شکل ۵-۲۵ عملکرد وقفه I/O Interruption (2) برایAHCPU530-EN ۱۶۴
شکل ۵-۲۶ باز کردن پنجره HWCONFIG برای تنظیم I/O Intrruption (2) ۱۶۴
شکل ۵-۲۷ انتخاب AH02HC-5A برای تنظیم I/O Intrruption (2) ۱۶۴
شکل ۵-۲۸ تنظیم CH1 Number of Interrupt Setting for Comparison ۱۶۵
شکل ۵-۲۹ تنظیم CH1 Comparison Values Setting ۱۶۵
شکل ۵-۳۰ بارگزاری تنظیمات سخت افزاری عملکرد وقفه ۱۶۵
شکل ۵-۳۱ POU تخصیص داده شده و تخصیص داده نشده به هر عملکرد ۱۶۶
شکل ۵-۳۲ اختصاص عملکرد به POU ۱۶۶
شکل ۵-۳۳ حذف عملکرد POU ۱۶۷
شکل ۵-۳۴ افزایش اولویت اجرای POU ۱۶۷
شکل ۵-۳۵ کاهش اولویت اجرای POU ۱۶۸
شکل ۵-۳۶ مثال۱- بخش های مختلف برنامه ۱۶۹
شکل ۵-۳۷ مثال۱- ساخت سه POU معادل ۱۷۰
شکل ۵-۳۸ مثال۱- برنامه RUN_STOP ۱۷۱
شکل ۵-۳۹ مثال۱- برنامه GLUING ۱۷۱
شکل ۵-۴۰ مثال۱- برنامه COUNTING ۱۷۱
شکل ۵-۴۱ مثال۱- ترتیب اجرای POU ۱۷۲
شکل ۵-۴۲ مثال۱- تغییر ترتیب اجرای POU ۱۷۳
شکل ۵-۴۳ مثال۱- ترتیب نهایی اجرای POU ۱۷۳
شکل ۵-۴۴ شرح مثال ۲ ..... ۱۷۳
شکل ۵-۴۵ مثال۲ – تعریف پروژه بر روی DVP-SV ۱۷۴

شکل ۵-۴۶ مثال۲ – ساخت چهار POU خواسته شده ۱۷۵
شکل ۵-۴۷ مثال۲ – برنامه Run_Stop ۱۷۵
شکل ۵-۴۸ مثال۲ – برنامه LED ۱۷۶
شکل ۵-۴۹ مثال۲ – برنامهShift ۱۷۶
شکل ۵-۵۰ مثال۲ – برنامه Reset ۱۷۶
شکل ۵-۵۱ مثال۲ – مشاهده POU اختصاص داده شده به عملکرد دوره ای صفر ۱۷۷
شکل ۵-۵۲ مثال۲ – حذف Reset از عملکرد دوره ای صفر ۱۷۸
شکل ۵-۵۳ مثال۲ – تنظیم ترتیب اجرای POU در عملکرد دوره ای صفر ۱۷۸
شکل ۵-۵۴ مثال۲ – اختصاص عملکرد External Interruption (X2) به Reset- قسمت ۱ ..... ۱۷۸
شکل ۵-۵۵ اختصاص عملکرد External Interruption (X2) به Reset- قسمت ۲ ..... ۱۷۹
شکل ۵-۵۶ مثال۲ – وضعیت نهایی اختصاص عملکرد به POU ۱۷۹
شکل ۵-۵۷ مثال۲ – استخراج برنامه در WPLSoft ۱۸۰
شکل ۵-۵۸ مثال۲ – حذف عملکرد POU های Run_Stop و غیر فعال کردن عملکرد Shift ۱۸۰
شکل ۵-۵۹ مثال۲ – نتیجه حذف عملکرد POU های Run_Stop و غیر فعال کردن عملکرد Shift در WPLSoft ۱۸۰
شکل ۵-۶۰ شرح مثال ۳ ..... ۱۸۱
شکل ۵-۶۱ شرح نمودار زمانی مثال ۳ ..... ۱۸۱
شکل ۵-۶۲ مثال۳ – ساخت پروژه با AHCPU350-EN ۱۸۳
شکل ۵-۶۳ مثال۳ – تنظیمات سخت افزاری ۱۸۳
شکل ۵-۶۴ مثال۳ – ساخت پنج POU با عملکرد پیشفرض Cyclic (0) ۱۸۴
شکل ۵-۶۵ مثال۳ – برنامه Initialize ۱۸۴
شکل ۵-۶۶ مثال۳ – برنامه INT-Timer ۱۸۵
شکل ۵-۶۷ مثال۳ – برنامه Time_CHK ۱۸۵
شکل ۵-۶۸ مثال۳ – برنامه Signal ۱۸۵
شکل ۵-۶۹ مثال۳ –Control ۱۸۶
شکل ۵-۷۰ مثال۳ – تنظیمات مربوط به تخصیص عملکرد POU ۱۸۶
شکل ۵-۷۱ مثال۳ – حذف Initialize و INT_Timer از عملکرد دوره ای صفر ۱۸۷
شکل ۵-۷۲ مثال۳ – تنظیم ترتیب اجرای POU در عملکرد دوره ای صفر ۱۸۷
شکل ۵-۷۳ مثال۳ – انتخاب عملکرد دوره ای یک ۱۸۷
شکل ۵-۷۴ مثال۳ – اختصاص عملکرد دوره ای یک به Initialize ۱۸۸

شکل ۵-۷۵ مثال۳ – انتخاب عملکرد وقفه Timed Intrupption 0 ..... ۱۸۸
شکل ۵-۷۶ مثال۳ - اختصاص عملکرد Timed Intrupption 0 به INT_Timer ۱۸۹
شکل ۵-۷۷ مثال۳ –وضعیت نهایی اختصاص عملکرد به POU ۱۸۹
شکل ۵-۷۸ مثال۳- تنظیمات سخت افزاری برای وقفه زمانی ۱۸۹
شکل ۵-۷۹ مثال۳- تنظیمات سخت افزاری CPU ۱۹۰
شکل ۵-۸۰ مثال۳- تنظیم مدت زمان وقفه زمانی ۱۹۱
شکل ۵-۸۱ مثال۳- بارگزاری تنظیمات سخت افزاری وقفه زمانی در PLC ۱۹۱
شکل ۶-۱ استفاده از سیمبول ها در برنامه ۱۹۳
شکل ۶-۲ پردازش اطلاعات FB و ارتباط آن با انواع ورودی، خروجی و ورودی/خروجی ۱۹۵
شکل ۶-۳ استفاده از آدرس دهی غیر مستقیم سیمبول ۲۰۰
شکل ۶-۴ عملکرد سیستم حین استفاده از آدرس دهی غیرمستقیم ۲۰۰
شکل ۶-۵ آدرس دهی غیر مستقیم آرایه ها ۲۰۱
شکل ۶-۶ نحوه دسترسی به بیت های حافظه داده ۲۰۲
شکل ۶-۷ پنجره جدول سیمبول سراسری ۲۰۳
شکل ۶-۸ جدول سیمبول های محلی ۲۰۴
شکل ۶-۹ اضافه کردن سیمبول جدید در جدول سیمبول ها ۲۰۵
شکل ۶-۱۰ تنظیم سیمبول جدید در جدول سیمبول ها ۲۰۵
شکل ۶-۱۱ تنظیم کلاس سیمبول جدید در جدول سیمبول ها ۲۰۵
شکل ۶-۱۲ فرمت داده سیمبول جدید در جدول سیمبول ها ۲۰۵
شکل ۶-۱۳ فرمت داده Function Block برای سیمبول جدید در جدول سیمبول ها ۲۰۶
شکل ۶-۱۴ فرمت داده STRING برای سیمبول جدید در جدول سیمبول ها ۲۰۶
شکل ۶-۱۵ فرمت داده ARRAY برای سیمبول جدید در جدول سیمبول ها ۲۰۷
شکل ۶-۱۶ تنظیم آدرس سیمبول جدید در جدول سیمبول ها ۲۰۷
شکل ۶-۱۷ مقدار دهی اولیه سیمبول جدید در جدول سیمبول ها ۲۰۸
شکل ۶-۱۸ توضیحات برای سیمبول جدید در جدول سیمبول ها ۲۰۸
شکل ۶-۱۹ استفاده از سیمبول آرایه ۲۰۹
شکل ۶-۲۰ اختصاص درایه سیمبول های آرایه ای ۲۱۰
شکل ۶-۲۱ مقدار دهی اولیه آرایه ها ۲۱۱
شکل ۶-۲۲ سیمبول STRING ۲۱۲
شکل ۶-۲۳ ویرایش سیمبول ها ۲۱۳

شکل ۶-۲۴ پاک کردن آدرس سیمبول ها ۲۱۳
شکل ۶-۲۵ دانلود مقادیر اولیه سیمبول ها ۲۱۴
شکل ۶-۲۶ ذخیره سازی جدول سیمبول ها ۲۱۵
شکل ۶-۲۷ ویرایش جدول سیمبول ها در اِکسل ۲۱۶
شکل ۶-۲۸ بازخوانی جدول سیمبول ها- قسمت ۱ ..... ۲۱۶
شکل ۶-۲۹ بازخوانی جدول سیمبول ها- قسمت ۲ ..... ۲۱۶
شکل ۶-۳۰ مرتب کردن سیمبول ها و تغییر ترتیب پایه بلوک ها ۲۱۷
شکل ۶-۳۱ فیلتر کردن سیمبول ها - قسمت ۱ ..... ۲۱۸
شکل ۶-۳۲ فیلتر کردن سیمبول ها – قسمت ۲ ..... ۲۱۸
شکل ۶-۳۳ سربرگ Normal در کنار سربرگ های فیلتر شده ۲۱۹
شکل ۶-۳۴ انتخاب Device Resource Allocation ۲۱۹
شکل ۶-۳۵ تنظیم Device Resource Allocation ۲۲۰
شکل ۶-۳۶ مثال: تکمیل جدول سیمبول های سراسری ۲۲۲
شکل ۶-۳۷ مثال: جایگزین شدن سیمبول ها به جای نام کانتکت ها ۲۲۳
شکل ۶-۳۸ تغییر وضعیت حالت نمایش سیمبول و آدرس دهی مستقیم ۲۲۳
شکل ۶-۳۹ مثال: اختصاص سیمبول به پارامترهای سیستم ۲۲۴
شکل ۶-۴۰ مثال: برنامه RUN_STOP پس از اضافه شدن سیمبول ها ۲۲۴
شکل ۶-۴۱ مثال: برنامه GLUING پس از اضافه شدن سیمبول ها ۲۲۵
شکل ۶-۴۲ مثال: برنامه COUNTING پس از اضافه شدن سیمبول ها ۲۲۵
شکل ۶-۴۳ نوار وضعیت در حالت آنلاین ۲۲۵
شکل ۷-۱ نمونه ای از توابع بلوکی ۲۲۶
شکل ۷-۲ مثالی از فراخوانی توابع بلوکی ۲۲۷
شکل ۷-۳ مثالی از پنجره یک تابع بلوکی ۲۲۹
شکل ۷-۴ پایه En و Eno در توابع بلوکی ۲۲۹
شکل ۷-۵ ورودی بلوک و نحوه عملکرد رجیسترهای ورودی ۲۳۲
شکل ۷-۶ خروجی بلوک و نحوه عملکرد رجیسترهای خروجی بلوک ۲۳۲
شکل ۷-۷ نحوه عملکرد ورودی/ خروجیهای بلوک ۲۳۳
شکل ۷-۸ بررسی تطابق حافظه ورودی/خروجی و پایه ورودی/خروجی متناظر در تابع بلوکی ۲۳۳
شکل ۷-۹ سیمبول هایی با فرمت داده اشارهگر ۲۳۶
شکل ۷-۱۰ مثالی جهت استفاده از اشاره گر در توابع بلوکی ۲۳۹

شکل ۷-۱۱ مثالی استفاده از اشاره گرهای زمان سنج و شمارنده در توابع بلوکی ۲۴۰
شکل ۷-۱۲ مثالی جهت آشنایی با مرجع و نسخه توابع بلوکی- قسمت اول ۲۴۱
شکل ۷-۱۳ مثالی جهت آشنایی با مرجع و نسخه توابع بلوکی- قسمت دوم ۲۴۲
شکل ۷-۱۴ مثالی جهت آشنایی با مرجع و نسخه توابع بلوکی- قسمت سوم ۲۴۲
شکل ۷-۱۵ مرجع توابع بلوکی ۲۴۳
شکل ۷-۱۶ نسخه تابع بلوکی- قسمت اول ۲۴۳
شکل ۷-۱۷ نسخه تابع بلوکی- قسمت دوم ۲۴۴
شکل ۷-۱۸ لیست نسخههای مرجع تابع بلوکی در زیر نام آن ۲۴۴
شکل ۷-۱۹ فراخوانی های متوالی توابع بلوکی ۲۴۵
شکل ۷-۲۰ توابع بلوکی به صورت سلسله مراتبی امکان فراخوانی خود را در سطوح پایین تر ندارند ۲۴۵
شکل ۷-۲۱ مثالی از تعریف نسخه های توابع بلوکی در جدول سیمبول های محلی و سراسری ۲۴۶
شکل ۷-۲۲ استفاده از نسخه مشترک در دو تابع بلوکی ۲۴۷
شکل ۷-۲۳ نسخهها در بخش مدیریت برنامه، نام توابع بلوکی با اولویت بالاتر خود را دارند ۲۴۷
شکل ۷-۲۴ اختصاص بلوک های حافظه به توابع بلوکی ۲۴۹
شکل ۷-۲۵ مثالی جهت بررسی روابط بین توابع بلوکی ۲۵۰
شکل ۷-۲۶ تخصیص حافظه به توابع بلوکی ۲۵۱
شکل ۷-۲۷ دستورات پالسی PED و NED ۲۵۵
شکل ۷-۲۸ بلوک های دستورهای پالسی ۲۵۵
شکل ۷-۲۹ دستورات پالسی FB_PN و FB_NP ۲۵۶
شکل ۷-۳۰ مانیتورینگ توابع بلوکی – قسمت اول ۲۵۶
شکل ۷-۳۱ مانیتورینگ توابع بلوکی – قسمت دوم ۲۵۷
شکل ۷-۳۲ در مانیتورینگ مرجع توابع بلوکی مقداری نمایش داده نخواهد شد ۲۵۷
شکل ۷-۳۳ ویرایش آنلاین تابع بلوکی ۲۵۸
شکل ۷-۳۴ امکان ویرایش آنلاین نسخه تابع ورودی وجود ندارد ۲۵۸
شکل ۷-۳۵ بعد از ویرایش تابع بلوکی باید آن را در برنامه حذف و دوباره اضافه کنیم ۲۵۸
شکل ۷-۳۶ مثال – ساخت تابع بلوکی جدید EQUIP_TRIG ۲۶۰
شکل ۷-۳۷ مثال – تابع بلوکی EQUIP_TRIG ۲۶۰
شکل ۷-۳۸ مثال- ساخت تابع بلوکی جدید PARTS_CNT ۲۶۱
شکل ۷-۳۹ مثال- تابع بلوکی PARTS_CNT ۲۶۱
شکل ۷-۴۰ مثال – دوبار فراخوانی تابع بلوکی EQUIP_TRIG ۲۶۲

شکل ۷-۴۱ مثال- فراخوانی تابع بلوکی تابع بلوکی EQUIP_TRIG از تعریف آن ۲۶۲
شکل ۷-۴۲ مثال- تنظیم ورودی و خروجی و دیتا بلاک تابع بلوکی EQUIP_TRIG ۲۶۳
شکل ۷-۴۳ مثال- تنظیم هر دو تابع بلوکی تابع بلوکی EQUIP_TRIG ۲۶۳
شکل ۷-۴۴ مثال – فراخوانی تابع بلوکی PARTS_CNT در COUNTING ۲۶۴
شکل ۷-۴۵ مثال- تنظیم پارامترهای بلوک PARTS_CNT ۲۶۴
شکل ۷-۴۶ مثال- تکمیل برنامه نوشته شده – قسمت اول ۲۶۵
شکل ۷-۴۷ مثال- تکمیل برنامه نوشته شده – قسمت دوم ۲۶۷
شکل ۷-۴۸ مثال- تکمیل برنامه نوشته شده – قسمت سوم ۲۶۷
شکل ۷-۴۹ مثال- بخش مدیریت برنامه پس از کامپایل برنامه ۲۶۸
شکل ۷-۵۰ مثال- بارگزاری برنامه و مقادیر اولیه ۲۶۹
شکل ۷-۵۱ مثال- مانیتورینگ آنلاین برنامه – قسمت اول ۲۶۹
شکل ۷-۵۲ مثال- مانیتورینگ آنلاین برنامه – قسمت دوم ۲۷۰
شکل ۷-۵۳ مثال- مانیتورینگ آنلاین برنامه – قسمت سوم ۲۷۱
شکل ۷-۵۴ مثال- مانیتورینگ آنلاین برنامه – قسمت چهارم ۲۷۲
شکل ۷-۵۵ مثال- مانیتورینگ آنلاین برنامه – قسمت پنجم ۲۷۳
شکل ۷-۵۶ مثال- مانیتورینگ آنلاین برنامه – قسمت ششم ۲۷۳
شکل ۷-۵۷ مثال- مانیتورینگ آنلاین برنامه – قسمت هفتم ۲۷۴
شکل ۷-۵۸ مثال- مانیتورینگ آنلاین برنامه – قسمت هشتم ۲۷۴
شکل ۸-۱ اجزای اصلی زبان برنامه نویسی Ladder در ISPSoft ۲۷۵
شکل ۸-۲ ایجاد برنامه جدید در پروژه ۲۷۶
شکل ۸-۳ انتخاب زبان برنامه نویسی Ladder ۲۷۶
شکل ۸-۴ محیط برنامه نویسی (ویرایش برنامه) ۲۷۷
شکل ۸-۵ نوار ابزار برنامه نویسی Ladder ۲۷۷
شکل ۸-۶ وضعیت Network ها ۲۷۹
شکل ۸-۷ اضافه کردن Network زیر Network انتخاب شده ۲۷۹
شکل ۸-۸ اضافه کردن Network بالای Network انتخاب شده ۲۷۹
شکل ۸-۹ انتخاب اجزای صفحه ویرایش برنامه Ladder ۲۸۰
شکل ۸-۱۰ نوار ابزار Ladder- اضافه کردن کانتکت ۲۸۲
شکل ۸-۱۱ اضافه کردن کانتکت به صورت سری و یا موازی ۲۸۲
شکل ۸-۱۲ اضافه کردن کانتکت در Network جدید ۲۸۳

شکل ۸-۱۳ اضافه شدن کانتکت در موقعیت های مختلف در Network ۲۸۳
شکل ۸-۱۴ انتخاب نوع کانتکت ۲۸۳
شکل ۸-۱۵ نوار ابزار Ladder- اضافه کردن کویل خروجی ۲۸۳
شکل ۸-۱۶ اضافه کردن کویل خروجی به Network ۲۸۴
شکل ۸-۱۷ اضافه کردن کویل به Network جدید ۲۸۴
شکل ۸-۱۸ انتخاب نوع کویل خروجی ۲۸۵
شکل ۸-۱۹ اختصاص حافظه، سیمبول، پورت و یا عدد ثابت به المان های سیستم ۲۸۵
شکل ۸-۲۰ انتخاب خط برای نوشتن دستورالعمل ۲۸۷
شکل ۸-۲۱ نوشتن دستورالعمل ۲۸۷
شکل ۸-۲۲ دستورالعمل کانتکت باز ۲۸۷
شکل ۸-۲۳ دستورالعمل کانتکت بسته ۲۸۷
شکل ۸-۲۴ دستورالعمل کویل خروجی ۲۸۷
شکل ۸-۲۵ بلوکهای API و FB در زبان Ladder ۲۸۸
شکل ۸-۲۶ استفاده نادرست از بلوک و اتصال مستقیم خط ورودی به پایه En ۲۸۸
شکل ۸-۲۷ اضافه کردن بلوک به برنامه ۲۸۹
شکل ۸-۲۸ بلوک مقایسه ای ۲۹۰
شکل ۸-۲۹ اتصال بلوکهای مقایسه ای ۲۹۰
شکل ۸-۳۰ عملیات بیتی لبه گیری ۲۹۱
شکل ۸-۳۱ انتخاب نوع عملیات بیتی در نوار ابزار ۲۹۲
شکل ۸-۳۲ اضافه کردن عملیات بیتی به Network – قسمت اول ۲۹۲
شکل ۸-۳۳ اضافه کردن عملیات بیتی به Network – قسمت دوم ۲۹۳
شکل ۸-۳۴ ایجاد دو کویل خروجی موازی در یک Network ۲۹۳
شکل ۸-۳۵ امکان برنامه نویسی منحصر به فرد هرکدام از کویل های خروجی موازی ۲۹۴
شکل ۸-۳۶ اضافه کردن برچسب ۲۹۴
شکل ۸-۳۷ پرش به برچسب با بلوک CJ ۲۹۴
شکل ۸-۳۸ ایجاد برچسب- قسمت اول ۲۹۴
شکل ۸-۳۹ ایجاد برچسب- قسمت دوم ۲۹۵
شکل ۸-۴۰ نمایش/مخفی کردن توضیحات ۲۹۵
شکل ۸-۴۱ توضیح نویسی برای Network ۲۹۶
شکل ۸-۴۲ نمایش/ مخفی شدن توضیحات المان ها ۲۹۶

شکل ۸-۴۳ توضیح نویسی برای المان ها ۲۹۷
شکل ۸-۴۴ نمایش توضیحات المان ها با بردن نشانگر موس بر روی آن ها ۲۹۷
شکل ۸-۴۵ نمایش سیمبول ها در برنامه ۲۹۸
شکل ۸-۴۶ نمایش حافظه و یا پورت ها در برنامه ۲۹۹
شکل ۸-۴۷ اضافه کردن نشان ۲۹۹
شکل ۸-۴۸ جابجایی بین نشان ها ۳۰۰
شکل ۸-۴۹ انتخاب Network ۳۰۰
شکل ۸-۵۰ فعال / غیر فعال کردن Network ۳۰۰
شکل ۹-۱ برنامهای به زبان ST و برنامه Ladder معادل آن ۳۰۲
شکل ۹-۲ ساختار برنامه ST ۳۰۳
شکل ۹-۳ نمونه ای از یک Statement که یک Statement در خود دارد ۳۰۳
شکل ۹-۴ یک عبارت نادرست در ST ۳۰۴
شکل ۹-۵ مثالی از Statement ها در زبان برنامه نویسی ST ۳۰۴
شکل ۹-۶ مثال بخش Expression ۳۰۶
شکل ۹-۷ عملوندها و عملگرها در Expression ۳۰۶
شکل ۹-۸ اضافه کردن توضیحات ۳۰۹
شکل ۹-۹ استفاده از آرایه در ST ۳۰۹
شکل ۹-۱۰ لزوم یکسان بودن عملوندها در یک Expression ۳۱۱
شکل ۹-۱۱ تخصیص دهی- مثال ۱ ..... ۳۱۳
شکل ۹-۱۲ تخصیص دهی – مثال ۲ ..... ۳۱۳
شکل ۹-۱۳ تخصیص دهی – مثال ۳ ..... ۳۱۳
شکل ۹-۱۴ تخصیص دهی – مثال ۴ ..... ۳۱۳
شکل ۹-۱۵ تخصیص دهی – مثال ۵ ..... ۳۱۴
شکل ۹-۱۶ فرمت دستور شرطی IF ۳۱۶
شکل ۹-۱۷ دستور شرطی IF- مثال ۱ ..... ۳۱۷
شکل ۹-۱۸ دستور شرطی IF- مثال ۲ ..... ۳۱۷
شکل ۹-۱۹ دستور شرطی IF- مثال ۲ ..... ۳۱۸
شکل ۹-۲۰ فرمت دستور شرطی CASE ۳۱۸
شکل ۹-۲۱ استفاده از دستور شرطی CASE- مثال ۱ ..... ۳۲۱
شکل ۹-۲۲ استفاده از دستور شرطی CASE- مثال ۲ ..... ۳۲۱

شکل ۹-۲۳ فرمت دستور حلقه REPEAT ۳۲۱
شکل ۹-۲۴ حلقه REPEAT –مثال ۱ ..... ۳۲۳
شکل ۹-۲۵ حلقه REPEAT –مثال ۲ ..... ۳۲۳
شکل ۹-۲۶ فرمت دستور حلقه WHILE ۳۲۳
شکل ۹-۲۷ حلقه WHILE –مثال ۱ ..... ۳۲۵
شکل ۹-۲۸ حلقه WHILE –مثال ۲ ..... ۳۲۵
شکل ۹-۲۹ فرمت دستور حلقه FOR ۳۲۶
شکل ۹-۳۰ حلقه For - مثال ۳۲۷
شکل ۹-۳۱ فرمت استفاده از دستورالعمل ها ۳۲۸
شکل ۹-۳۲ استفاده از Enter در زمان فراخوانی دستورالعمل ۳۲۸
شکل ۹-۳۳ استفاده از دستورالعمل ها –مثال ۱ ..... ۳۲۹
شکل ۹-۳۴ استفاده از دستورالعمل ها –مثال ۲ ..... ۳۲۹
شکل ۹-۳۵ فرمت استفاده از FB ۳۲۹
شکل ۹-۳۶ ترتیب اعمال پایه ها در ST مطابق ترتیب پایه های بلوک است ۳۳۰
شکل ۹-۳۷ استفاده از FB – مثال ۱ ..... ۳۳۱
شکل ۹-۳۸ استفاده از FB – مثال ۲ ..... ۳۳۱
شکل ۹-۳۹ استفاده از FB – مثال ۳ ..... ۳۳۱
شکل ۹-۴۰ فرمت دستور تاخیر ۳۳۲
شکل ۹-۴۱ دستور تاخیر- مثال ۳۳۲
شکل ۹-۴۲ فرمت دستور RETURN ۳۳۲
شکل ۹-۴۳ دستور RETURN - مثال ۳۳۳
شکل ۹-۴۴ فرمت دستور EXIT ۳۳۳
شکل ۹-۴۵ دستور EXIT- مثال ۱ ..... ۳۳۴
شکل ۹-۴۶ دستور EXIT- مثال ۲ ..... ۳۳۵
شکل ۹-۴۷ ساخت برنامه به زبان ST- قسمت ۱ ..... ۳۳۵
شکل ۹-۴۸ ساخت برنامه به زبان ST- قسمت ۲، محیط ویرایش برنامه ۳۳۵
شکل ۹-۴۹ نوار ابزار برنامه نویسی به زبان ST ۳۳۶
شکل ۹-۵۰ برنامه نویسی به زبان ST ۳۳۶
شکل ۹-۵۱ استفاده از مثال های آماده دستورات نوار ابزار برنامه نویسی ۳۳۷
شکل ۹-۵۲ انتخاب بخشی از برنامه ۳۳۷

شکل ۹-۵۳ تعریف سیمبول حین برنامه نویسی ۳۳۸
شکل ۹-۵۴ تایپ فرمت بلوک و یا دستورالعمل ۳۳۸
شکل ۹-۵۵ استفاده از FB و یا APIهای های لیست شده در بخش مدیریت پروژه ۳۳۸
شکل ۹-۵۶ مشخص کردن عملوند مرتبط هر پایه بلوک ۳۳۹
شکل ۹-۵۷ نشان گزاری در ST ۳۳۹
شکل ۹-۵۸ جابجایی بین نشان ها ۳۴۰
شکل ۹-۵۹ سیستم روشنایی متصل به PLC ۳۴۰
شکل ۹-۶۰ ساخت پروژه - مثال ۳۴۲
شکل ۹-۶۱ تنظیمات سخت افزاری - مثال ۳۴۳
شکل ۹-۶۲ ساخت POU برنامه و بلوکی - مثال ۳۴۴
شکل ۹-۶۳ وارد کردن سیمبول های محلی - مثال ۳۴۵
شکل ۹-۶۴ برنامه نوشته شده برای بلوک - مثال ۳۴۶
شکل ۹-۶۵ برنامه دوره ای اصلی - مثال ۳۴۶
شکل ۱۰-۱ نمایی از محیط برنامه نویسی به زبان FBD ۳۴۷
شکل ۱۰-۲ انتخاب زبان برنامه نویسی FBD ۳۴۸
شکل ۱۰-۳ محیط برنامه نویسی (ویرایش برنامه) با FBD ۳۴۹
شکل ۱۰-۴ نوار ابزار برنامه نویسی FBD ۳۴۹
شکل ۱۰-۵ انتخاب اجزای صفحه ویرایش برنامه FBD ۳۵۱
شکل ۱۰-۶ انتخاب همزمان چند Network در FBD ۳۵۲
شکل ۱۰-۷ برنامه تابع بلوکی به زبان FBD ۳۵۲
شکل ۱۰-۸ برنامه تابع بلوکی معادل به زبان Ladder ۳۵۳
شکل ۱۰-۹ انتقال مستقیم ورودی به خروجی در زبان FBD ۳۵۳
شکل ۱۰-۱۰ انتقال بیتی در دو زبان FBD و Ladder ۳۵۳
شکل ۱۰-۱۱ انتقال داده در دو زبان FBD و Ladder ۳۵۴
شکل ۱۰-۱۲ موقعیت های مجاز برای انتخاب گره خروجی در FBD ۳۵۴
شکل ۱۰-۱۳ گره خروجی در FBD ۳۵۴
شکل ۱۰-۱۴ اختصاص گره خروجی به ورودی گیت های AND و OR در FBD ۳۵۴
شکل ۱۰-۱۵ اضافه کردن گره خروجی به سمت راست گره خروجی دیگر درFBD ۳۵۵
شکل ۱۰-۱۶ گیت OR (راست) و گیت AND (چپ) ۳۵۶
شکل ۱۰-۱۷ معادل گیت OR در زبان Ladder ۳۵۶

شکل ۱۰-۱۸ معادل کانتکت AND در زبان Ladder ۳۵۷
شکل ۱۰-۱۹ معادل Ladder برنامه ای به زبان FBD ۳۵۷
شکل ۱۰-۲۰ اضافه کردن پایه ورودی گیت ۳۵۷
شکل ۱۰-۲۱ کم کردن پایه ورودی گیت ۳۵۸
شکل ۱۰-۲۲ معادل Ladder عملکرد گیت NOT ۳۵۸
شکل ۱۰-۲۳ معادل Ladder برنامه ای به زبان FBD- نمونه دوم ۳۵۹
شکل ۱۰-۲۴ اجرای یکی از توابع بلوکی Black و یا White ۳۵۹
شکل ۱۰-۲۵ از گیت NOT برای داده های غیر بیتی نمیتوان استفاده کرد. ۳۵۹
شکل ۱۰-۲۶ اضافه و حذف کردن گیت NOT. ۳۶۰
شکل ۱۰-۲۷ تشخیص دهنده لبه پایین رونده F (راست) و تشخیص دهنده لبه بالارونده P (چپ). ۳۶۰
شکل ۱۰-۲۸ تشخیص دهنده لبه بالا رونده در زبان FBD و معادل Ladder آن ۳۶۰
شکل ۱۰-۲۹ نمونه برنامه به زبان FBD که در آن از تشخیص دهنده لبه استفاده شده و معادل Ladder آن. ۳۶۰
شکل ۱۰-۳۰ تشخیص دهنده های لبه تنها برای داده های نوع بیتی قابل استفاده هستند. ۳۶۱
شکل ۱۰-۳۱ اضافه کردن تشخیص دهندههای لبه به برنامه ۳۶۱
شکل ۱۰-۳۲ ریست کننده خروجی بیتی (راست)، ست کننده خروجی بیتی (چپ). ۳۶۱
شکل ۱۰-۳۳ معادل Ladder ریست کننده خروجی بیتی ۳۶۲
شکل ۱۰-۳۴ معادل Ladder برنامه ای که در آن از ست و ریست کننده خروجی بیتی استفاده شده است. ۳۶۲
شکل ۱۰-۳۵ از ست و ریست کننده خروجی بیتی نمی توان برای داده های غیربیتی استفاده کرد. ... ۳۶۲
شکل ۱۰-۳۶ اضافه و حذف ست و ریست کننده خروجی های بیتی ۳۶۳
شکل ۱۰-۳۷ استفاده از توابع بلوکی در دو زبان برنامه نویسی Ladder و FBD به یک شکل است. ۳۶۴
شکل ۱۰-۳۸ پنج خط برنامه به زبان IL و معادل Ladder آن. ۳۶۵
شکل ۱۰-۳۹ عملوندها و عملگرها در زبان برنامه نویسی IL ۳۶۶
شکل ۱۰-۴۰ نمونه های مجاز کامنت گذاری در IL ۳۶۶
شکل ۱۰-۴۱ راهنمای عملگرها ۳۶۷
شکل ۱۰-۴۲ استفاده از F1 برای باز شدن راهنمای عملگرها ۳۶۷
شکل ۱۰-۴۳ فراخوانی توابع در IL ۳۶۷
شکل ۱۰-۴۴ معادل Ladder برنامه IL در زمان فراخوانی تابع بلوکی ۳۶۸
شکل ۱۰-۴۵ به کارگیری معادل پایه En با توجه به نتیجه عملیات بیتی قبل فراخوانی تابع بلوکی. ... ۳۶۸

شکل ۱۰-۴۶ ایجاد پروژه جدید به زبان IL ۳۶۸
شکل ۱۰-۴۷ محیط برنامه نویسی به زبان IL ۳۶۹
شکل ۱۰-۴۸ فاصله و یا سطر جدید که فرمت دستورات را به هم ریزد در IL خطا ایجاد می کند. ۳۷۰
شکل ۱۰-۴۹ عملیات بیتی قبل فراخوانی شرط اجرای آن تابع بلوکی را تعیین می کند ۳۷۰
شکل ۱۰-۵۰ برنامه ای به زبان SFC و فلوچارت متناظر آن ۳۷۳
شکل ۱۰-۵۱ نمونه ای از مراحل اجرای یک برنامه به زبان SFC ۳۷۴
شکل ۱۰-۵۲ بخش های مختلف یک برنامه SFC ۳۷۵
شکل ۱۰-۵۳ عملکردها در بخش مدیریت پروژه ۳۷۶
شکل ۱۰-۵۴ تنظیم عملکردها برای پله Process در برنامه نمونه ۳۷۷
شکل ۱۰-۵۵ عملکرد Action0 برای پله Step_0 ..... ۳۷۷
شکل ۱۰-۵۶ فعال بودن Action0 در نتیجه فعال بودن STEP_0 ..... ۳۷۸
شکل ۱۰-۵۷ غیرفعال شدن Action0 پس از اسکن نهایی ۳۷۸
شکل ۱۰-۵۸ برنامه های نوشته شده گذارها در بخش مدیریت پروژه ۳۷۹
شکل ۱۰-۵۹ برنامه گذار به زبان Ladder ۳۸۰
شکل ۱۰-۶۰ برنامه گذار به زبان FBD ۳۸۰
شکل ۱۰-۶۱ برنامه گذار به زبان ST ۳۸۰
شکل ۱۰-۶۲ برنامه گذار به زبان IL ۳۸۰
شکل ۱۰-۶۳ انشعاب اجرای همزمان ۳۸۱
شکل ۱۰-۶۴ انشعاب انتخاب ترتیبی ۳۸۲
شکل ۱۰-۶۵ همگرایی همزمان ۳۸۳
شکل ۱۰-۶۶ همگرایی انتخاب ترتیبی ۳۸۴
شکل ۱۰-۶۷ برنامه نمونه دارای پرش درون برنامه ای ۳۸۵
شکل ۱۰-۶۸ اجرای متوالی دو برنامه SFC با استفاده از پرش ۳۸۶
شکل ۱۰-۶۹ طراحی پرش بین برنامه ای برای جابجایی بین دو برنامه SFC ۳۸۷
شکل ۱۰-۷۰ پنجره تخصیص بررسی کننده به عملکردها ۳۸۸
شکل ۱۰-۷۱ اختصاص عملکرد نوع Delay ۳۸۹
شکل ۱۰-۷۲ مقایسه دو عملکرد SD و DS ۳۸۹
شکل ۱۰-۷۳ مقایسه دو عملکرد L و SL ۳۹۰
شکل ۱۰-۷۴ در صورت ست شدن عملکرد، تنها با ریست کردن می توان آن را متوقف کرد. ۳۹۲
شکل ۱۰-۷۵ پله اولیه در SFC ۳۹۳

شکل ۱۰-۷۶ پیاده سازی مکانیزم پله اولیه در خانواده DVP ۳۹۴
شکل ۱۰-۷۷ ساخت برنامه به زبان SFC ۳۹۴
شکل ۱۰-۷۸ محیط برنامه نویسی به زبان SFC ۳۹۵
شکل ۱۰-۷۹ نوار ابزار برنامه نویسی به زبان SFC ۳۹۵
شکل ۱۰-۸۰ امکان ویرایش برنامه SFC در محیط برنامه ۳۹۷
شکل ۱۰-۸۱ ساخت عملکرد ۳۹۸
شکل ۱۰-۸۲ ساخت گذار ۳۹۸
شکل ۱۰-۸۳ محیط ویرایش برنامه عملکرد و گذار ۳۹۹
شکل ۱۰-۸۴ انتخاب محل اضافه شدن پله جدید ۳۹۹
شکل ۱۰-۸۵ اضافه کردن پله به SFC ۴۰۰
شکل ۱۰-۸۶ انتخاب محل اضافه شدن گذار موازی ۴۰۱
شکل ۱۰-۸۷ اضافه کردن گذار موازی ۴۰۱
شکل ۱۰-۸۸ انتخاب محل اضافه شدن پله موازی ۴۰۲
شکل ۱۰-۸۹ اضافه کردن پله موازی ۴۰۲
شکل ۱۰-۹۰ انتخاب محل اضافه شدن انشعاب همزمان و همگرایی انتخاب ترتیبی ۴۰۳
شکل ۱۰-۹۱ ساخت انشعاب همزمان و همگرایی انتخاب ترتیبی ۴۰۴
شکل ۱۰-۹۲ انتخاب محل اضافه شدن انشعاب انتخاب ترتیبی و همگرایی همزمان ۴۰۵
شکل ۱۰-۹۳ ساخت انشعاب انتخاب ترتیبی و همگرایی همزمان ۴۰۶
شکل ۱۰-۹۴ انتخاب محل اضافه شدن نقطه پرش ۴۰۶
شکل ۱۰-۹۵ایجاد نقطه پرش در برنامه ۴۰۷
شکل ۱۰-۹۶ اختصاص سیمبول به پله ها و گذارها ۴۰۷
شکل ۱۰-۹۷ تعیین پله اولیه ۴۰۸
شکل ۱۰-۹۸ اختصاص پله برای تخصیص عملکرد ۴۰۸
شکل ۱۰-۹۹ انتخاب عملکرد ۴۰۹
شکل ۱۰-۱۰۰ تنظیم بررسی کننده برای عملکرد ۴۰۹
شکل ۱۰-۱۰۱ نمایش عملکردهای اختصاص داده شده به برنامه ۴۰۹
شکل ۱۱-۱ برقراری ارتباط بین ISPSoft با PLC ۴۱۰
شکل ۱۱-۲ شروع به کار و توقف PLC با استفاده از ISPSoft ۴۱۱
شکل ۱۱-۳ تغییر وضعیت سیستم به حالت آنلاین ۴۱۲
شکل ۱۱-۴ فعال کردن مانیتورینگ پورت ها و حافظه و سپس مانیتورینگ برنامه ۴۱۲

شکل ۱۱-۵ خارج شدن از حالت آنلاین ۴۱۲
شکل ۱۱-۶ نوار وضعیت در حالت ارتباط آنلاین ۴۱۳
شکل ۱۱-۷ نوار وضعیت پس از رخ دادن خطا در PLC در حین ارتباط آنلاین ۴۱۳
شکل ۱۱-۸ مانیتورینگ آنلاین برنامه به زبان Ladder ۴۱۴
شکل ۱۱-۹ مانیتورینگ آنلاین برنامه به زبان Function block diagram ۴۱۴
شکل ۱۱-۱۰ مانیتورینگ آنلاین برنامه به زبان Instruction List ۴۱۴
شکل ۱۱-۱۱ مانیتورینگ آنلاین برنامه به زبان Structured Text ۴۱۴
شکل ۱۱-۱۲ مانیتورینگ آنلاین برنامه به زبان Sequential Function Chart ۴۱۵
شکل ۱۱-۱۳ نمایش داده ها در وضعیت Automatic به تنظیمات سیمبول ها بستگی دارد ۴۱۵
شکل ۱۱-۱۴ تنظیم نحوه نمایش اعداد ممیز شناور (اعشاری) در مانیتورینگ آنلاین ۴۱۶
شکل ۱۱-۱۵ پیغام خطا در صورتی که PLC از تغییر آنلاین کانتکت پشتیبانی نکند ۴۱۷
شکل ۱۱-۱۶ تغییر آنلاین ورودی ها ۴۱۷
شکل ۱۱-۱۷ لزوم غیر فعال کردن MPU قبل از خروج از حالت آنلاین ۴۱۸
شکل ۱۱-۱۸ تغییر وضعیت کانتکت ها در Ladder و FBD ۴۱۸
شکل ۱۱-۱۹ تغییر وضعیت کانتکت ها در Instruction List و Structured text ۴۱۹
شکل ۱۱-۲۰ تغییر وضعیت کانتکت ها در SFC ۴۱۹
شکل ۱۱-۲۱ تغییر مقدار حافظه، ورودی و یا غیره با استفاده از گزینه Change Present Value ۴۲۰
شکل ۱۱-۲۲ خروجی در حالت اجبار ۴۲۱
شکل ۱۱-۲۳ حالت اجبار ON ۴۲۱
شکل ۱۱-۲۴ حالت اجبار OFF ۴۲۲
شکل ۱۱-۲۵ حذف حالت اجبار ۴۲۲
شکل ۱۱-۲۶ حذف تمامی حالت های اجبار ۴۲۲
شکل ۱۱-۲۷ لیست حالت اجبار ۴۲۲
شکل ۱۱-۲۸ لیست حالت اجبار- حالت اجبار ON ۴۲۳
شکل ۱۱-۲۹ لیست حالت اجبار- حالت اجبار OFF ۴۲۳
شکل ۱۱-۳۰ لیست حالت اجبار- حذف حالت اجبار ۴۲۳
شکل ۱۱-۳۱ لیست حالت اجبار- حذف تمامی حالت های اجبار ۴۲۳
شکل ۱۱-۳۲ ایجاد جدول مانیتورینگ –قسمت اول ۴۲۴
شکل ۱۱-۳۳ ایجاد جدول مانیتورینگ –قسمت دوم ۴۲۴
شکل ۱۱-۳۴ باز کردن پنجره جدول مانیتورینگ ۴۲۴

شکل ۱۱-۳۵ اضافه کردن حافظه و پورت در جدول مانیتورینگ ۴۲۵
شکل ۱۱-۳۶ اضافه کردن سیمبول به جدول مانیتورینگ –قسمت اول ۴۲۵
شکل ۱۱-۳۷ اضافه کردن سیمبول به جدول مانیتورینگ –قسمت دوم ۴۲۶
شکل ۱۱-۳۸ پاک کردن المان ها در جدول سیمبول ها ۴۲۶
شکل ۱۱-۳۹ انتخاب ستون ها در جدول مانیتورینگ ۴۲۷
شکل ۱۱-۴۰ مانیتورینگ و تغییر وضعیت آنلاین در جدول مانیتورینگ ۴۲۸
شکل ۱۱-۴۱ حالت در حال ویرایش در نوار وضعیت ۴۲۸
شکل ۱۱-۴۲ ساخت درایور نوع DVP Simulator برای شبیه ساز ۴۳۰
شکل ۱۱-۴۳ فعال کردن درایور نوع DVP Simulator ۴۳۰
شکل ۱۱-۴۴ تنظیم درایور ISPSoft به صورت DVP Simulator ۴۳۱
شکل ۱۱-۴۵ اتصال به درایور نوع DVP Simulator در نوار وضعیت ۴۳۱
شکل ۱۱-۴۶ بلوک دیاگرام ارتباط ISPSoft با شبیه ساز DVP از طریق COMMGR ۴۳۱
شکل ۱۱-۴۷ فعال کردن حالت عیب یابی ۴۳۲
شکل ۱۱-۴۸ Instruction List برای حالت عیب یابی ۴۳۲
شکل ۱۱-۴۹ ایجاد نقطه انفصال در حالت عیب یابی ۴۳۳
شکل ۱۱-۵۰ حذف نقطه انفصال در حالت عیب یابی ۴۳۳
شکل ۱۱-۵۱ اجرا از نقطه انفصال تا نقطه انفصال بعدی در حالت عیب یابی ۴۳۴
شکل ۱۱-۵۲ اجرای تک خطی در حالت عیب یابی ۴۳۴
شکل ۱۱-۵۳ اجرای برنامه به دفعات مشخص در حالت عیب یابی ۴۳۵
شکل ۱۱-۵۴ توقف اجرای برنامه در حالت عیب یابی ۴۳۵
شکل ۱۱-۵۵ ساخت درایور جدید ۴۳۵
شکل ۱۱-۵۶ تنظیم درایور AH Simulator ۴۳۶
شکل ۱۱-۵۷ فعال شدن پنجره شبیه ساز AH500 پس از فعال شدن درایور آن در COMMGR ۴۳۷
شکل ۱۱-۵۸ برقراری ارتباط شبیه ساز AH500 با ISPSoft ۴۳۷
شکل ۱۱-۵۹ پنجره شبیه ساز AH500 ..... ۴۳۷
شکل ۱۱-۶۰ وضعیت مشابه فعال بودن درایور و شبیه ساز ۴۳۸
شکل ۱۱-۶۱ صفحه ورودی/خروجی شبیه ساز ۴۳۸
شکل ۱۱-۶۲ تنظیمات سخت افزاری برای پیاده سازی در شبیه ساز AH500 ..... ۴۳۹
شکل ۱۱-۶۳ شبیه ساز آماده شده مشابه تنظیمات سخت افزاری ۴۳۹
شکل ۱۱-۶۴ امکان تغییر ورودی های شبیه ساز و وابسته بودن خروجی های آن به برنامه ۴۳۹

شکل ۱۱-۶۵ ورودی و خروجی های آنالوگ شبیه ساز AH500 ..... ۴۴۰
شکل ۱۱-۶۶ انتخاب نوع سیگنال ورودی آنالوگ در شبیه سازAH500 ..... ۴۴۰
شکل ۱۱-۶۷ اعمال مقدار تعیین شده در شبیه سازAH500 ..... ۴۴۱
شکل ۱۱-۶۸ انتخاب ورودی و خروجی برای نمایش در شبیه ساز AH500 ..... ۴۴۱
شکل ۱۱-۶۹ نمایش تمامی ماژول های ورودی و خروجی ۴۴۲
شکل ۱۱-۷۰ تهیه نسخه پشتیبان از شبیه ساز AH500 ..... ۴۴۲
شکل ۱۱-۷۱ فراخوانی نسخه پشتیبان در شبیه ساز ۴۴۳
شکل ۱۱-۷۲ فعال کردن حالت عیب یابی در AH500 ..... ۴۴۳
شکل ۱۱-۷۳ حالت عیب یابی در نوار وضعیت ۴۴۳
شکل ۱۱-۷۴ ایجاد نقطه انفصال در حالت عیب یابی AH500 ..... ۴۴۴
شکل ۱۱-۷۵ حذف تمامی نقاط انفصال در حالت عیب یابی AH500 ..... ۴۴۴
شکل ۱۱-۷۶ اجرای برنامه از نقطه انفصال به نقطه انفصال بعدی ۴۴۵
شکل ۱۱-۷۷ اجرای برنامه تا نقطه انفصال بعدی و یا اجرای پیوسته ۴۴۶
شکل ۱۱-۷۸ اجرای مرحله به مرحله در حالت عیب یابی AH500 ..... ۴۴۷
شکل ۱۱-۷۹ گزارش عملکرد PLC ۴۴۸
شکل ۱۱-۸۰ گزارش عملکرد PLC- سربرگ Error Log ۴۴۸
شکل ۱۱-۸۱ گزارش عملکرد PLC- سربرگProgram Change Log ۴۴۹
شکل ۱۱-۸۲ گزارش عملکرد PLC- سربرگ Status Change Log ۴۵۰
شکل ۱۲-۱ تغییر نوع PLC ۴۵۱
شکل ۱۲-۲ بارگزاری بخش های مختلف پروژه ۴۵۲
شکل ۱۲-۳ استخراج بخش های مختلف پروژه ۴۵۳
شکل ۱۲-۴ تعیین نوع توضیحات برای بارگزاری ۴۵۴
شکل ۱۲-۵ پنجره Find/Replace/Goto در زبان برنامه نویسی Ladder و FBD ۴۵۵
شکل ۱۲-۶ جستجو و جایگزینی در زبان برنامه نویسی Ladder و FBD ۴۵۵
شکل ۱۲-۷ استفاده از Goto در زبان برنامه نویسی Ladder و FBD ۴۵۶
شکل ۱۲-۸ پنجره Find در زبان برنامه نویسی IL و ST ۴۵۷
شکل ۱۲-۹ پنجره Replace در زبان برنامه نویسی IL و ST ۴۵۷
شکل ۱۲-۱۰ صفحه تایید عملکرد Replace در زبان برنامه نویسی IL و ST ۴۵۸
شکل ۱۲-۱۱ پنجره Find/Replace در زبان برنامه نویسی SFC ۴۶۰
شکل ۱۲-۱۲ جستجو و جایگزینی در زبان برنامه نویسی Ladder و FBD ۴۶۰

شکل ۱۲-۱۳ پنجره جستجوی سیمبول ها ۴۶۲
شکل ۱۲-۱۴ جستجو و جایگزینی حافظه ها و سیمبول ها –قسمت ۱ ..... ۴۶۳
شکل ۱۲-۱۵ جستجو و جایگزینی حافظه ها و سیمبول ها –قسمت ۲ ..... ۴۶۳
شکل ۱۲-۱۶ تنظیمات پرینتر ۴۶۴
شکل ۱۲-۱۷ پرینت پروژه ۴۶۵
شکل ۱۲-۱۸ نمونه ای از پرینت پروژه ۴۶۵
شکل ۱۲-۱۹ پرینت صفحه جاری ۴۶۶
شکل ۱۲-۲۰ پیش نمایش قبل از پرینت ۴۶۷
شکل ۱۲-۲۱ ذخیره سازی و استفاده مجدد از کامنت حافظه ها در PLCهای DVP ۴۶۸
شکل ۱۲-۲۲ ویرایش کامنت حافظه ها در PLCهای DVP ۴۶۸
شکل ۱۲-۲۳ صفحه ویرایش توضیحات حافظه ها ۴۶۹
شکل ۱۲-۲۴ مشاهده حافظه های دارای توضیحات ۴۶۹
شکل ۱۲-۲۵ گزینه های موجود در جدول توضیحات حافظه ها ۴۷۰
شکل ۱۲-۲۶ جستجو (پرش) به حافظه مورد نظر در جدول توضیحات حافظه ها ۴۷۰
شکل ۱۲-۲۷ کپی توضیحات به همراه نام حافظه ها ۴۷۱
شکل ۱۲-۲۸ انتقال توضیحات حافظه ها به سیمبولهای آن ها ۴۷۲
شکل ۱۲-۲۹ گزارش حافظه های استفاده شده در برنامه ۴۷۲
شکل ۱۲-۳۰ گزارش حافظه های استفاده شده در برنامه- شرح بخش چپ ۴۷۳
شکل ۱۲-۳۱ گزارش حافظه های استفاده شده در برنامه- شرح بخش راست ۴۷۳
شکل ۱۲-۳۲ رجوع به برنامه ای که حافظه در آن به کار رفته است از گزارش حافظه های استفاده شده ۴۷۴
شکل ۱۲-۳۳ پرش به حافظه مورد نظر در گزارش حافظه های استفاده شده در برنامه ۴۷۴
شکل ۱۲-۳۴ پنجره مدیریت حافظه ها در PLCهای سری AH500 ..... ۴۷۵
شکل ۱۲-۳۵ پنجره مدیریت حافظه ها در PLCهای سری AH500 سربرگ Vie/Edit Device Comment ۴۷۶
شکل ۱۲-۳۶ تغییر توضیحات حافظه در پنجره مدیریت حافظه ها در PLCهای سری AH500 ..... ۴۷۶
شکل ۱۲-۳۷ انتخاب تمامی خانه های پنجره مدیریت حافظه ها در PLCهای سری AH500 ..... ۴۷۷
شکل ۱۲-۳۸ شکل ۱۲-۳۹ حذف خانه های پنجره مدیریت حافظه ها در PLCهای سری AH500 ..... ۴۷۷
شکل ۱۲-۴۰ تنظیم خانه های پنجره مدیریت حافظه ها در PLCهای سری AH500 ..... ۴۷۷
شکل ۱۲-۴۱ ذخیره و فراخوانی خانه های پنجره مدیریت حافظه ها در PLCهای سری AH500 ..... ۴۷۸

شکل ۱۲-۴۲ پنجره مدیریت حافظه ها در PLCهای سری AH500 سربرگ Used Device ۴۷۸
شکل ۱۲-۴۳ انتخاب Edit Register Memory (T,C,D) ۴۷۹
شکل ۱۲-۴۴ پنجره اصلاح مقادیر حافظه های T/C/D در PLCهای سری DVP ۴۷۹
شکل ۱۲-۴۵ وارد کردن مقادیر حافظه های T/C/D در PLCهای سری DVP ۴۸۰
شکل ۱۲-۴۶ نوع نمایش اعداد در جدول مقادیر حافظه های T/C/D ۴۸۰
شکل ۱۲-۴۷ نمایش اعداد در جدول مقادیر حافظه های T/C/D با طول های مختلف ۴۸۰
شکل ۱۲-۴۸ پاک کردن داده های جدول مقادیر حافظه های T/C/D ۴۸۱
شکل ۱۲-۴۹ پاک کردن داده های مشخصی از جدول مقادیر حافظه های T/C/D ۴۸۱
شکل ۱۲-۵۰ تبادل داده های بین PLC و جدول مقادیر حافظه های T/C/D ۴۸۲
شکل ۱۲-۵۱ ذخیره و بازیابی مقادیر حافظه ها در صفحه ی Edit Register Memory ۴۸۳
شکل ۱۲-۵۲ ذخیره کردن جدول Edit Register Memory به صورت فایل CSV ۴۸۳
شکل ۱۲-۵۳ اصلاح جدول Edit Register Memory به صورت فایل CSV ۴۸۴
شکل ۱۲-۵۴ استفاده از جدول اصلاح شده Edit Register Memory به صورت فایل CSV ۴۸۴
شکل ۱۲-۵۵ اضافه کردن توضیحات به جدول Edit Register Memory ۴۸۵
شکل ۱۲-۵۶ پرینت جدول Edit Register Memory ۴۸۶
شکل ۱۲-۵۷ انتخاب Edit Bit Memory (M,S) ۴۸۷
شکل ۱۲-۵۸ پنجره اصلاح مقادیر حافظه های M/S در PLCهای سری ۴۸۷
شکل ۱۲-۵۹ وارد کردن مقادیر حافظه های M/S در PLCهای سری DVP ۴۸۸
شکل ۱۲-۶۰ اصلاح مقادیر جدول حافظه های M/S در PLCهای سری DVP ۴۸۸
شکل ۱۲-۶۱ تبادل داده های بین PLC و جدول مقادیر حافظه های M/S ۴۸۹
شکل ۱۲-۶۲ ذخیره کردن جدول Edit Bit Memory به صورت فایل CSV ۴۸۹
شکل ۱۲-۶۳ استفاده از جدول اصلاح شده Edit Bit Memory به صورت فایل CSV ۴۸۹
شکل ۱۲-۶۴ ذخیره و بازیابی مقادیر حافظه ها در صفحه ی Edit Bit Memory ۴۹۰
شکل ۱۲-۶۵ پرینت جدول Edit Bit Memory ۴۹۱
شکل ۱۲-۶۶ انتخاب Edit File Register Memory ۴۹۱
شکل ۱۲-۶۷ پنجره اصلاح مقادیر حافظه های فایل در PLCهای سری DVP ۴۹۲
شکل ۱۲-۶۸ وارد کردن مقادیر حافظه های فایل در PLCهای سری DVP ۴۹۲
شکل ۱۲-۶۹ نوع نمایش اعداد در جدول مقادیر حافظه های فایل ۴۹۳
شکل ۱۲-۷۰ پاک کردن داده های جدول مقادیر حافظه های فایل ۴۹۳
شکل ۱۲-۷۱ پاک کردن داده های مشخصی از جدول مقادیر حافظه های فایل ۴۹۴

شکل ۱۲-۷۲ تبادل داده های بین PLC و جدول مقادیر حافظه های فایل ۴۹۴
شکل ۱۲-۷۳ ذخیره و بازیابی مقادیر حافظه ها در صفحه ی Edit File Register Memory..... ۴۹۵
شکل ۱۲-۷۴ ذخیره کردن جدول Edit File Register Memory به صورت فایل CSV..... ۴۹۶
شکل ۱۲-۷۵ استفاده از جدول اصلاح شده Edit File Register Memory به صورت فایل CSV ۴۹۶
شکل ۱۲-۷۶ اضافه کردن توضیحات به جدول Edit File Register Memory ۴۹۷
شکل ۱۲-۷۷ پرینت جدول Edit File Register Memory ۴۹۸
شکل ۱۲-۷۸ روشهای متعدد رمزگذاری در ISPSoft..... ۴۹۹
شکل ۱۲-۷۹ صفحه تنظیم شناسه برنامه..... ۵۰۱
شکل ۱۲-۸۰ ایجاد و حذف شناسه برنامه ۵۰۱
شکل ۱۲-۸۱ صفحه تنظیم شناسه PLC ۵۰۲
شکل ۱۲-۸۲ ایجاد و حذف شناسه PLC ۵۰۳
شکل ۱۲-۸۳ تنظیم رمز عبور پروژه ۵۰۴
شکل ۱۲-۸۴ ایجاد رمز عبور پروژه ۵۰۴
شکل ۱۲-۸۵ تنظیمات مربوط به ایجاد رمز عبور پروژه ۵۰۵
شکل ۱۲-۸۶ حذف رمز عبور پروژه ۵۰۵
شکل ۱۲-۸۷ تنظیم رمز عبور PLC ۵۰۶
شکل ۱۲-۸۸ ایجاد رمز PLC ۵۰۷
شکل ۱۲-۸۹ تبدیل رمز عبور پروژه به رمز عبور PLC ۵۰۷
شکل ۱۲-۹۰ تبدیل رمز عبور PLC به رمز عبور پروژه ۵۰۸
شکل ۱۲-۹۱ تنظیم رمز عبور POU ۵۰۹
شکل ۱۲-۹۲ ایجاد و حذف رمز عبور POU ۵۰۹
شکل ۱۲-۹۳ امکان دسترسی به برنامه ها پس از استخراج از PLC ۵۱۰
شکل ۱۲-۹۴ رمز عبور Subroutine ۵۱۱
شکل ۱۲-۹۵ تنظیم رمز عبور Subroutine ۵۱۱
شکل ۱۲-۹۶ ایجاد و حذف رمز عبور Subroutine ۵۱۲
شکل ۱۲-۹۷ غیرفعال کردن استخراج برنامه ۵۱۳
شکل ۱۲-۹۸ تعیین محدوده فقط خواندنی حافظه ها در PLCهای سری DVP ۵۱۳
شکل ۱۲-۹۹ پشتیبان گیری از داده های PLCهای DVP..... ۵۱۴
شکل ۱۲-۱۰۰ محل نمایش اتصال درست کارت حافظه PLCهای سری DVP..... ۵۱۵
شکل ۱۲-۱۰۱ نحوه تبادل داده بین PLCهای سری DVP و کارت حافظه ۵۱۵

شکل ۱۲-۱۰۲ تنظیم کارتهای حافظه دائم در PLCهای DVP ۵۱۶
شکل ۱۲-۱۰۳ پشتیبان گیری و بازیابی اطلاعات در AH500 ..... ۵۱۸
شکل ۱۲-۱۰۴ تنظیمات CARD Utility ۵۱۸
شکل ۱۲-۱۰۵ تنظیمات CARD Utility – پشتیبان گیری – قسمت ۱ ..... ۵۱۹
شکل ۱۲-۱۰۶ تنظیمات CARD Utility – پشتیبان گیری – قسمت ۲ ..... ۵۲۰
شکل ۱۲-۱۰۷ تنظیمات CARD Utility – پشتیبان گیری – قسمت ۳ ..... ۵۲۱
شکل ۱۲-۱۰۸ تنظیمات CARD Utility – پشتیبان گیری – قسمت ۴ ..... ۵۲۱
شکل ۱۲-۱۰۹ تنظیمات CARD Utility – پشتیبان گیری – قسمت ۵ ..... ۵۲۲
شکل ۱۲-۱۱۰ تنظیمات CARD Utility –هشدار در مورد وجود فایل پشتیبان قدیمی ۵۲۳
شکل ۱۲-۱۱۱ تنظیمات CARD Utility – پشتیبان گیری – قسمت ۶ ..... ۵۲۴
شکل ۱۲-۱۱۲ تنظیمات CARD Utility – بازیابی پشتیبان– قسمت ۱ ..... ۵۲۵
شکل ۱۲-۱۱۳ تنظیمات CARD Utility – بازیابی پشتیبان– قسمت ۲ ..... ۵۲۵
شکل ۱۲-۱۱۴ تنظیمات CARD Utility – بازیابی پشتیبان– قسمت ۳ ..... ۵۲۶
شکل ۱۲-۱۱۵ تنظیمات CARD Utility – بازیابی پشتیبان– قسمت ۴ ..... ۵۲۷
شکل ۱۲-۱۱۶ تنظیمات CARD Utility – بازیابی پشتیبان– قسمت ۵ ..... ۵۲۷
شکل ۱۲-۱۱۷ تنظیمات CARD Utility – بازیابی پشتیبان– قسمت ۶ ..... ۵۲۹
شکل ۱۲-۱۱۸ نصب درایور USB – قسمت ۱ ..... ۵۳۰
شکل ۱۲-۱۱۹ نصب درایور USB – قسمت ۲ ..... ۵۳۰
شکل ۱۲-۱۲۰ نصب درایور USB – قسمت ۳ ..... ۵۳۱
شکل ۱۲-۱۲۱ نصب درایور USB – قسمت ۴ ..... ۵۳۲
شکل ۱۲-۱۲۲ نصب درایور USB – قسمت ۵ ..... ۵۳۳
شکل ۱۲-۱۲۳ بررسی نصب درایور در Device Manager ۵۳۳
شکل ۱۲-۱۲۴ اجرای COMMGR ۵۳۴
شکل ۱۲-۱۲۵ ساخت درایور USB – قسمت ۱ ..... ۵۳۵
شکل ۱۲-۱۲۶ ساخت درایور USB – قسمت ۲ ..... ۵۳۵
شکل ۱۲-۱۲۷ فعال کردن ارتباط درایور USB ۵۳۶
شکل ۱۲-۱۲۸ تنظیمات درایور USB برای PLCهای مدل DVP-SX2 ..... ۵۳۷
شکل ۱۲-۱۲۹ PLC Instructions and Special Register Reference در بخش Help نرم افزار ISPSoft ۵۳۹
شکل ۱۲-۱۳۰ ارتباط اجزای بیتی و تشکیل Word پورتهای ورودی ۵۴۰

شکل ۱۲-۱۳۱ کار با بیت ها در پورت و حافظه ها – مثال ۱ ..... ۵۴۰
شکل ۱۲-۱۳۲ کار با بیت ها در پورت و حافظه ها – مثال ۲ ..... ۵۴۰
شکل ۱۲-۱۳۳ به روز رسانی مستقیم پورت های ورودی و خروجی – مثال ۱ ..... ۵۴۱
شکل ۱۲-۱۳۴ به روز رسانی مستقیم پورت های ورودی و خروجی – مثال ۲ ..... ۵۴۲
شکل ۱۲-۱۳۵ به روز رسانی نادرست مستقیم پورت های ورودی و خروجی – مثال ۳ ..... ۵۴۲
شکل ۱۲-۱۳۰ محیط کار NWCONFIG ۵۴۹
شکل ۱۲-۱۳۰ پارامترهای اجزای شبکه ۵۵۰
شکل ۱۲-۱۳۰ ساختار ارتباط بین اجزا در Ether Link و PLC Link ۵۵۳
شکل ۱۲-۱۳۰ ارتباط مستقیم با یک گره مشخص ۵۵۴
شکل ۱۲-۱۳۰ ارتباط با گرههای یک شبکه ۵۵۴
شکل ۱۲-۱۳۰ تخصیص IPهای منحصر به فرد در شبکه Ethernet ۵۵۵
شکل ۱۲-۱۳۰ ارتباط از طریق مسیریابی ۵۵۵
شکل ۱۲-۱۳۰ انتخاب درایور برای برقراری ارتباط ۵۵۶
شکل ۱۲-۱۳۰ انتخاب ارتباط مسیر یابی و تعیین PLC دارای ارتباط مستقیم با نرم افزار ۵۵۷
شکل ۱۲-۱۳۰ ساخت پروژه گروهی ۵۵۸
شکل ۱۲-۱۳۰ تنظیم پارامترها شبکه در HWCONFIG ۵۵۸
شکل ۱۲-۱۳۰ تنظیم پارامترهای شبکه CPU در HWCONFIG ۵۵۹
شکل ۱۲-۱۳۰ تنظیم پارامترهای شبکه در NWCONFIG ۵۶۰
شکل ۱۲-۱۳۰ جدول تبادل داده در PLC Link ۵۶۱
شکل ۱۲-۱۳۰ جدول تبادل داده در Ether Link ۵۶۲
شکل ۱۲-۱۳۰ صفحهی بارگزاری داده ها ۵۶۳
شکل ۱۲-۱۳۰ اضافه کردن گرهها در NWCONFIG ۵۶۳
شکل ۱۲-۱۳۰ ایجاد شبکه و اتصال گره ها به شبکه ها در NWCONFIG ۵۶۴
شکل ۱۲-۱۳۰ تنظیمات گره ها در NWCONFIG ۵۶۴
شکل ۱۲-۱۳۰ تنظیمات شبکه ها در NWCONFIG ۵۶۵
شکل ۱۲-۱۳۰ ایجاد PLC Link در NWCONFIG ۵۶۵
شکل ۱۲-۱۳۰ تنظیمات جدول تبادل داده PLC Link در NWCONFIG ۵۶۶
شکل ۱۲-۱۳۰ مانیتورینگ آنلاین PLC Link در در NWCONFIG ۵۶۶
شکل ۱۲-۱۳۰ ایجاد و تنظیمات Ether Link در NWCONFIG ۵۶۷
شکل ۱۲-۱۳۰ مانیتورینگ آنلاین Ether Link و دادهها در NWCONFIG ۵۶۷

شکل ۱۲-۱۳۰ نمایی از نرم افزار Smart VIEwer کمپانی دلتا ۵۶۸

فهرست علایم و نشانه‌ها

عنوان	علامت اختصاری
نرم افزار برنامه نویسی ISPSoft برای PLC های کمپانی دلتا	ISPSoft
نرم افزار برنامه نویسی WPLSoft برای PLC های کمپانی دلتا	WPLSoft
نرم افزار مدیریت ارتباطات بین سخت افزارها، کامپیوتر و نرم‌افزارها	COMMGR
بخش تنظیمات سخت افزاری ISPSoft	NWCONFIG
بخش تنظیمات شبکه ISPSoft	HWCONFIG
توابع بلوکی	FB (Function Blocks)
شبکه محلی	LAN
نرم افزار متلب	MATLAB
Program Organization Unit	POU
Simple Mail Transfer Protocol	SMTP
Program organization unit	POU
Programmable Logic Controllers	PLC
Sequential Function Chart Language	SFC
Instruction List Language	IL
Function Block Diagram Language	FBD
Structured Text Language	ST
Ladder Diagram Language	LD
Industrial Ethernet	IE

فصل ۱– مقدمه

۱–۱– پیشگفتار

دقت، سرعت، بهینگی و هزاران مزیت دیگر همگی ویژگی‌هایی هستند که اتوماسیون صنعتی برای ما به ارمغان آورده است. مدت زمان زیادی از دوره‌ای که ربات‌ها و فرآیندهای اتوماتیک جایگزین انسان‌ها در کارخانه‌ها و مراکز صنعتی شده‌اند نمی‌گذرد و هم اکنون به موهبت این موضوع، توان فکر آدمی به جای توان جسمی آن در تولید مورد استفاده قرار گرفته است. پردازنده‌های قابل برنامه ریزی اما در قلب این جهش بزرگ تاریخی قرار دارند که امکان پیاده‌سازی مکانیزم‌های اتوماتیک بر قالب تجهیزات مختلف صنعتی را به صورت هرچه ساده‌تری فراهم کرده‌اند. در این میان PLCها یکی از مهمترین دستگاه‌های قابل برنامه ریزی هستند که مطابق با شرایط محیط‌های صنعتی طراحی می‌شوند و در تمام صنایع بزرگ و کوچک مورد استفاده قرار می‌گیرند.

برای اینکه بتوان در زمینه اتوماسیون صنعتی فعالیت کرد، داشتن دانش کار با فرآیندهای صنعتی و PLCها و نحوه برنامه‌ریزی آن‌ها ضروری است. دانش کار با زبان‌های برنامه نویسی و ویژگی‌های نرم افزار و سخت افزار PLC، توانایی تحلیل سیستماتیک فرآیندها صنعتی با استفاده از تئوری کنترل و آشنایی با مفاهیم پایه شبکه و نحوه تبادل داده‌ها از جمله کلیدی‌ترین مباحثی است که در این رابطه مورد نیاز می‌باشد. در این میان این کتاب می‌تواند راهنمای مناسبی در راستای آشنایی مقدماتی با اتوماسیون صنعتی و PLCها به شمار آید. در این کتاب به معرفی نرم افزار ISPSoft و زبان‌های برنامه نویسی قابل استفاده در آن خواهیم پرداخت. معرفی PLCهای کمپانی دلتا[۱] که از ISPSoft می‌توان برای برنامه‌ریزی آن‌ها استفاده کرد به صورت مفصل در کتاب صورت پذیرفته است. به جرات می‌توان گفت PLCهای دلتا به علت قیمت پایین خود و خدمات پس از فروش گسترده در سطح ایران یکی از پر استفاده ترین PLCهای سطح متوسط برای پروژه‌های عمومی صنعتی به حساب می‌آید.

۱–۲– معرفی نرم افزار ISPSoft

[۱] Delta Electronics: http://www.deltaww.com/

ISPSoft نرم افزار جدید کمپانی دلتا می‌باشد. پشتیبانی از پنج زبان مختلف برنامه نویسی و همچنین اضافه شدن دستورالعمل‌های کاربردی، جدید و متنوع از جمله قابلیت‌های ISPSoft به شمار می‌رود. پشتیبانی از PLC-های خانواده‌های AH500 و DVP به همراه مجموعه کاملی از کتابخانه‌های نرم‌افزاری از جمله ویژگی‌های اصلی این نرم افزار به شمار می‌آید. محیط نرم افزاری بهینه، قابل فهم و ساختاریافته کاربر را قادر می‌سازد تا بتواند با PLC های کمپانی دلتا پیچیده‌ترین سیستم‌های کنترلی را همانند سیستم‌های ساده برنامه نویسی نماید.

۱-۳- مروری بر مباحث مطرح شده در کتاب

در این کتاب سعی شده است مباحث گوناگون مرتبط با نرم افزار ISPSoft مطرح شوند. در فصل اول مرور فضای کلی کتاب و معرفی کمپانی دلتا صورت پذیرفته و سپس در فصل دوم مباحث مقدماتی در مورد ویژگی‌های نرم افزار مورد بررسی قرار خواهد گرفت. در فصل سوم علاوه بر شرح ساختار سخت افزاری PLCهای کمپانی دلتا، نحوه پیکربندی سخت افزاری آن‌ها مرور خواهد شد. در فصل چهارم، برای آنکه کاربران بتوانند همزمان با مطالعه کتاب، دید مناسب‌تری در کار با نرم افزار ISPSoft داشته باشند، بدون توجه به جزئیات اقدام به طرح و ساخت پروژه‌ای عملی با استفاده از نرم افزار ISPSoft شده است، پروژه طرح شده در این فصل به مرور در فصل‌های بعدی کامل‌تر شده است.

در فصل پنجم به مرور طراحی POUها در برنامه پرداخته شده است، POUها واحدهایی است که می-توانیم در آن‌ها برنامه اصلی PLC را به علاوه برنامه‌های وقفه طرح ریزی کنیم. در فصل شش به مرور سیمبول‌ها و در فصل هفت به مرور قواعد ایجاد توابع بلوکی پرداخته خواهد شد.

درفصل هشت و نه دو زبان برنامه نویسی Ladder و ST مرور خواهند شد، زبان برنامه نویسی Ladder پر استفاده ترین زبان برنامه نویسی برای PLC و زبان ST زبانی سطح بالا است که برای افرادی که در کُدنویسی مسلط هستند مناسب است. در فصل ده نیز به صورت خلاصه سه زبان برنامه نویسی IL و FBD و SCF مرور خواهند شد.

فصل یازدهم یکی از مهمترین بخش‌های کتاب می‌باشد که در آن به معرفی شبیه ساز و امکانات تست و عیب‌یابی و مانیتورینگ نرم افزار ISPSoft پرداخته خواهد شد، این مباحث از بخش‌های پرکاربرد در انجام پروژه‌های عملی است. فصل دوازدهم نیز فصل کاملی شامل تمام جزئیات باقیمانده در مورد نرم افزار است که در آن امکانات مختلف نرم افزار و ویژگی‌های PLCهای مختلف مرور خواهد شد. فصل سیزدهم نیز به مرور اجمالی بخش NWCONFIG برای پیکربندی شبکه و تنظیمات مربوط به تبادل داده

خواهد پرداخت. مباحث مربوط به توابع آماده نرم‌افزار، رجیسترهای کاربردی و مثال‌های نمونه برای پیاده سازی در PLC نیز در پیوست آمده است.

۱-۴- راهنمای نمادها

در این قسمت با نمادهایی که در این کتاب از آن‌ها برای سادگی هرچه بیشتر اجرای دستورالعمل‌ها استفاده شده است، آشنا خواهید شد.

جدول ۱-۱: نمادهای استفاده شده در کتاب

نماد	شرح
	کلیک چپ موس[۱]
	کلیک راست موس
	دوبار کلیک چپ[۲]
	فشردن و نگه داشتن کلیک چپ حین جابجا کردن آن[۳]
	تایپ با صفحه کلید
	مشخص کننده ترتیب اجرا، برای مثال در حالت  و  ابتدا باید کلیک چپ انجام و سپس تایپ صورت پذیرد.
	شماره‌های استفاده شده در تصاویر برای مجزا کردن بخش‌های مختلف

[۱] Left Click

[۲] Double Click

[۳] Drag and Drop

فصل ۲– مقدمه کار با نرم افزار ISPSoft

۲–۱– مقدمه

نرم افزار ISPSoft برای برنامه نویسی ساختار یافته PLCهای کمپانی دلتا عرضه شده است و دارای قابلیت‌های متعددی از جمله پشتیبانی بیش از ۵ زبان مختلف برنامه نویسی، شبیه ساز و امکان تنظیمات سخت افزاری و شبکه‌های صنعتی است. در این بخش به مرور کلیات پیرامون این نرم افزار خواهیم پرداخت.

۲–۱–۱– مروری بر ویژگی های شاخص نرم افزار

- منطبق با استاندارد بین‌المللی IEC 61131-3[۱]
- پشتیبانی از پنج زبان مختلف برنامه نویسی (در هر پروژه می‌توان از زبان‌های زیر به صورت ترکیبی در نوشتن برنامه ها استفاده کرد)
  - Ladder Diagrams (LD)
  - Sequential Function Charts (SFC)
  - Function Block Diagram (FBD)
  - Instruction Lists (IL)
  - Structured Texts (ST)
- عملکرد پیشرفته توابع Find و Replace برای استفاده در صفحه جاری یا کل پروژه
- قابلیت تنظیم محیط نرم‌افزار مطابق میل کاربر
- مدیریت پروژه ها در قالب ساختار سلسله مراتبی درختی[۲]
- قابلیت استفاده از زیربرنامه‌های مختلف در مجموعه‌ای از پروژه‌ها

---

[۱] استاندارد IEC 61131 استانداردی برای کنترل کننده های قابل برنامه ریزی است که قسمت سوم آن شامل استانداردهای مربوط به زبان های برنامه نویسی است.

[۲] Hierarchical Tree Structure ، ساختاری که پروژه اصلی مانند تنه درخت شامل زیربخش و زیر پروژه‌هایی مانند شاخه‌هاست. همچنین هر پروژه و یا شاخه ممکن است خود شامل زیر بخش‌ها و یا شاخه‌های کوچک‌تر باشد و بدین ترتیب ساختار سلسله مراتبی به وجود می‌آید.

- اضافه شدن توابع کاربردی متنوع مانند: اضافه کردن توضیحات[۱]، نشانه گزاری[۲]، فعال و غیرفعال سازی شبکه، مدیریت تجهیزات و نمادها[۳]، شبیه سازی و غیره.
- متنوع شدن روش های مانیتورینگ[٤] و ارتباط با PLC مانند: مانیتورینگ آنلاین برنامه‌ها، اصلاح آنلاین برنامه‌ها، مانیتورینگ تجهیزات، عیب یابی برنامه‌ها، فعال سازی و یا تنظیم PLC.
- امکان استخراج برنامه‌ها و همچنین به کارگیری مجدد برنامه‌های استخراج شده در پروژه‌های دیگر
- پشتیبانی از پروژه‌های ایجاد شده توسط نرم افزارWPLSoft[۵] و امکان تبدیل آن‌ها به پروژه-های تحت نرم افزار ISPSoft[۶]
- به کار گیری مکانیزم‌های متعدد رمزگزاری و محافظت از داده‌ها
- پشتیبانی از COMMGR[۷] برای ارتباط با سخت افزار
- پشتیبانی از PLC-های جدید سری AH500
- امکان به کار گیری سه حالت پیکربندی[۸]:
  - HWCONFIG: برای تنظیم سخت افزار و پارامترهای سیستم
  - NWCONFIG: برای تنظیم شبکه و مدیریت تبادل داده‌ها
  - CARD Utility: کارت حافظه برای تهیه نسخه پشتیبان و بازیابی آن
- محیط ساختار یافته برای برنامه نویسی با امکان گسترش سیستماتیک پروژه‌ها

---

[۱] Comment

[۲] Bookmark

[۳] Symbols

[۴] Monitoring

[۵] نسخه قدیمی نرم افزار ISPSoft برای کار با PLC-های DVP شرکت دلتا.

[۶] فرمت نام پروژه‌های تحت نرم افزار WPLSoft به صورت (dvp.*) و فرمت نام پروژه‌های تحت نرم افزار ISPSoft به صورت (isp.*) می-باشند.

[۷] نرم افزار جدید شرکت دلتا برای مدیریت ارتباطات بین سخت افزارها، کامپیوتر و نرم‌افزارها

[۸] Configuration

- افزایش کیفیت بصری محیط نرم افزار جهت انتقال حس مناسب‌تر و دقیق‌تر به کاربر

۲-۱-۲- سیستم مورد نیاز

برای اینکه بتوان از ISPSoft استفاده و از طریق آن با PLC ارتباط برقرار کرد، نیاز به کامپیوتر و تجهیزات با حداقل مشخصات زیر می‌باشد.

جدول ۲-۱: سیستم مورد نیاز برای کار با ISPSoft

تجهیزات	سیستم مورد نیاز
سیستم عامل	ویندوز ۷/Vista/XP/ME/NT/۲۰۰۰
پردازنده (CPU)	Pentium 1.5G و یا بالاتر
حافظه	۵۱۲ مگابایت یا بالاتر
دیسک سخت[۱]	حداقل ظرفیت ۵۰۰ مگابایت
نمایشگر	حداقل رزولوشن ۶۰۰*۸۰۰ (تنظیمات پیشنهادی، استفاده از رزولوشن 96dpi@ 768*1024 می‌باشد)
پورت RS-232	توجه شود که برای ارتباط کامپیوتر، PLC مورد نظر نیز باید از پورت مشابه پشتیبانی کند.
پورت USB	
پورت Ethernet	
نرم افزار ارتباطی	COMMGR، این نرم افزار برای مدیریت ارتباط نرم افزار ISPSoft با PLC باید بر روی کامپیوتر نصب شود[۲].
محصولاتی که با ISPSoft پشتیبانی می‌شوند.	PLCهای خانواده‌های AH500 و DVP درایوهای خانواده VFD-C2000 / VFD-C200 /VFD-CP2000 و VFD-E

[۱] Hard Disk Drive

[۲] در ادامه در مورد COMMGR بحث خواهد شد.

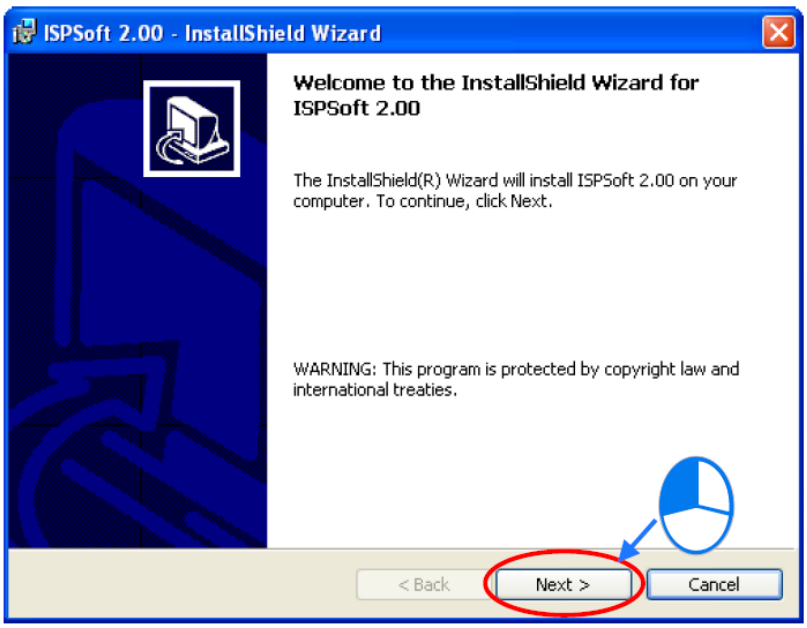
۲-۲- نصب نرم افزار ISPSoft

نصب ISPSoft روند بسیار ساده‌ای دارد. برای اینکار ابتدا نسخه‌های قدیمی ISPSoft بر روی کامپیوتر را پاک کرده و سپس اقدام به نصب نسخه جدید نمایید. می‌توانید آخرین نسخه از نرم افزار ISPSoft را به صورت رایگان از پایگاه http://www.delta.com.tw/ch/index.asp دریافت نمایید.

۱- برای نصب ISPSoft فایل نصب آن را (با پسوند *.exe) اجرا نمایید.

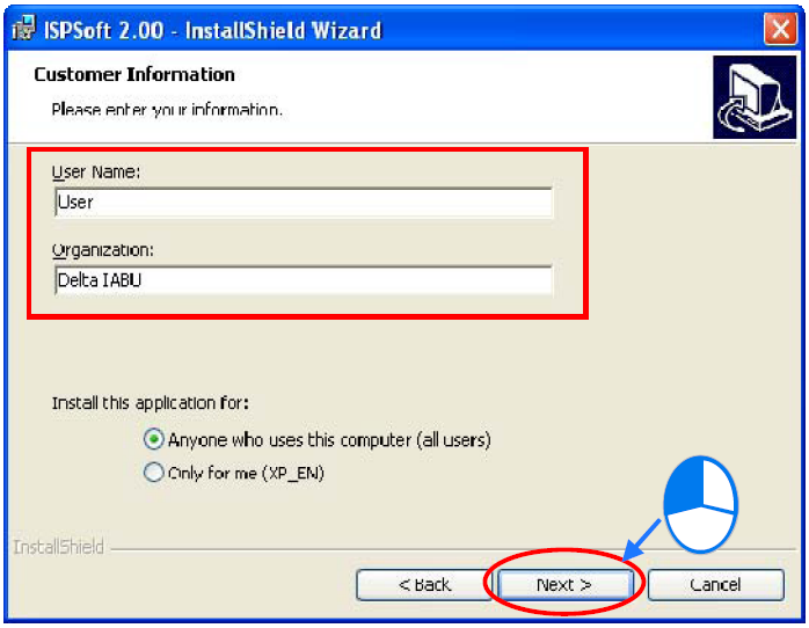
شکل ۲-۱ اجرای فایل Setup نرم افزار ISPSoft

۲- پس از ظاهر شدن صفحه ISPSoft (X.XX) – InstallShield Wizard برای ادامه کار روی گزینه Next کلیک نمایید.



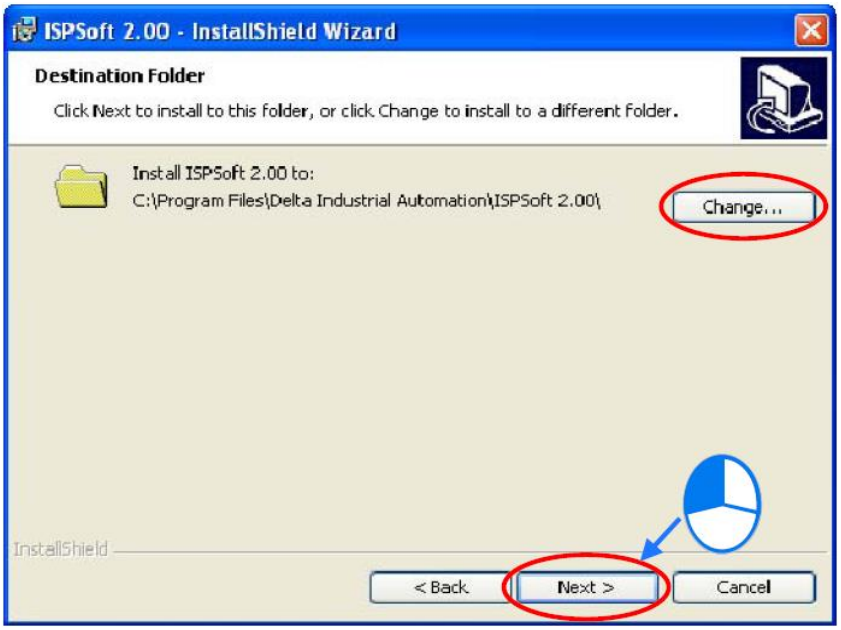
شکل ۲-۲ صفحه ابتدایی Setup

۳- پس از وارد کردن نام خود و سازمان مورد نظرتان در User Name و Organization می‌توانید در قسمت "Install this application for" اگر می‌خواهید دسترسی برای تمام کاربران کامپیوتر آزاد باشد از گزینه "Anyone who uses this computer" و در غیر این صورت از "Only for me" استفاده نمایید.



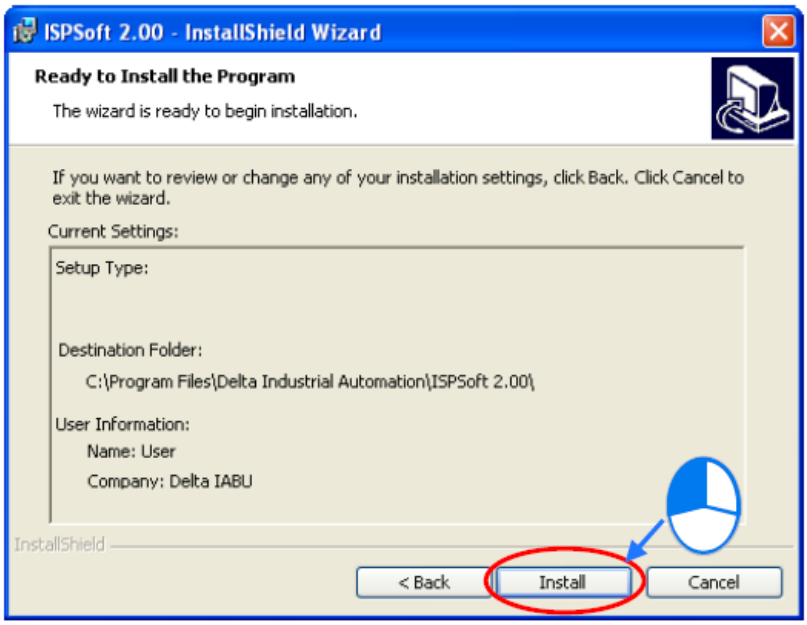
شکل ۲-۳ تعیین نام کاربر و سازمان در Setup

۴- در این مرحله می‌توانید آدرس محل نصب نرم افزار را با کلیک بر روی گزینه Change تعیین نمایید. به صورت پیش فرض، نرم افزار در پوشه‌ای تحت عنوان Delta Industrial Automation در Program Files نصب خواهد شد.



شکل ۲-۴ تنظیم آدرس نصب در Setup

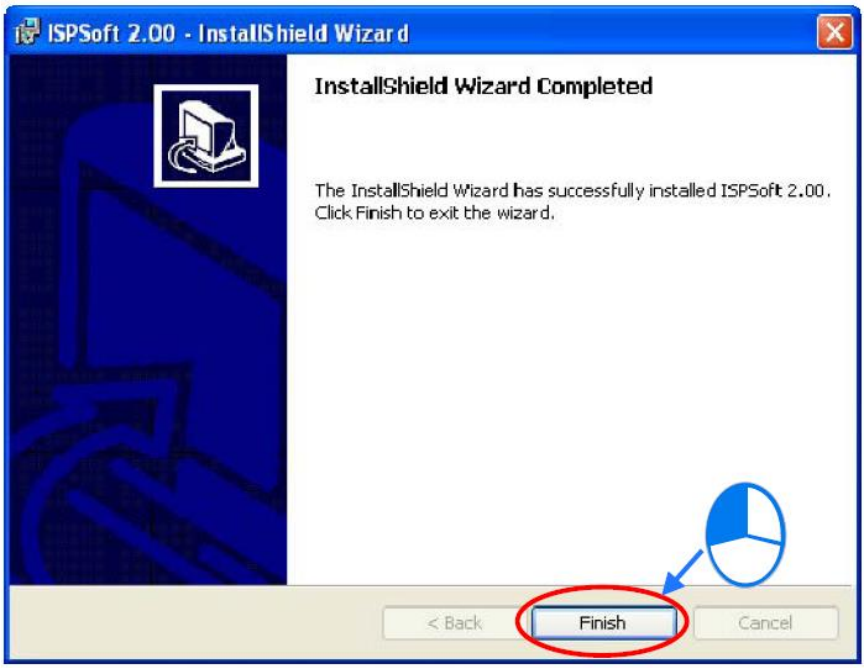
۵- پس از مرور اطلاعات وارد شده با کلیک بر روی گزینه Install برنامه نصب می‌شود.



شکل ۲-۵ بررسی نهایی اطلاعات در Setup

۶- پس از کلیک بر روی Finish میان‌بری از برنامه در دسکتاپ و منوی شروع ویندوز ایجاد خواهد شد.

شکل ۲-۶ تکمیل پروسه نصب در Setup



شکل ۲-۷ اتمام موفقیت آمیز نصب نرم افزار ISPSoft

۲–۳– راه اندازی ISPSoft و مروری بر فضای کلی آن

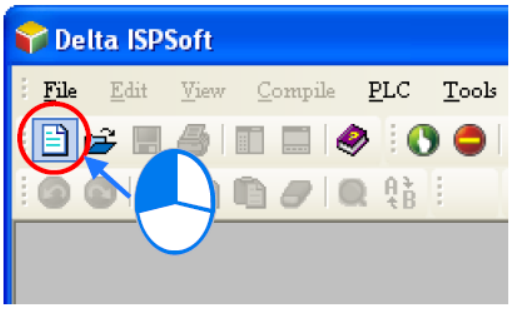
برای اجرای نرم افزار ISPSoft لازم است بر روی آیکون آن دوبار کلیک نماییم.

شکل ۲-۸ آیکون نرم افزار ISPSoft در Desktop

در این مرحله و پس از اجرای برنامه، می‌توانید پنجره نرم افزار ISPSoft را مشاهده نمایید.

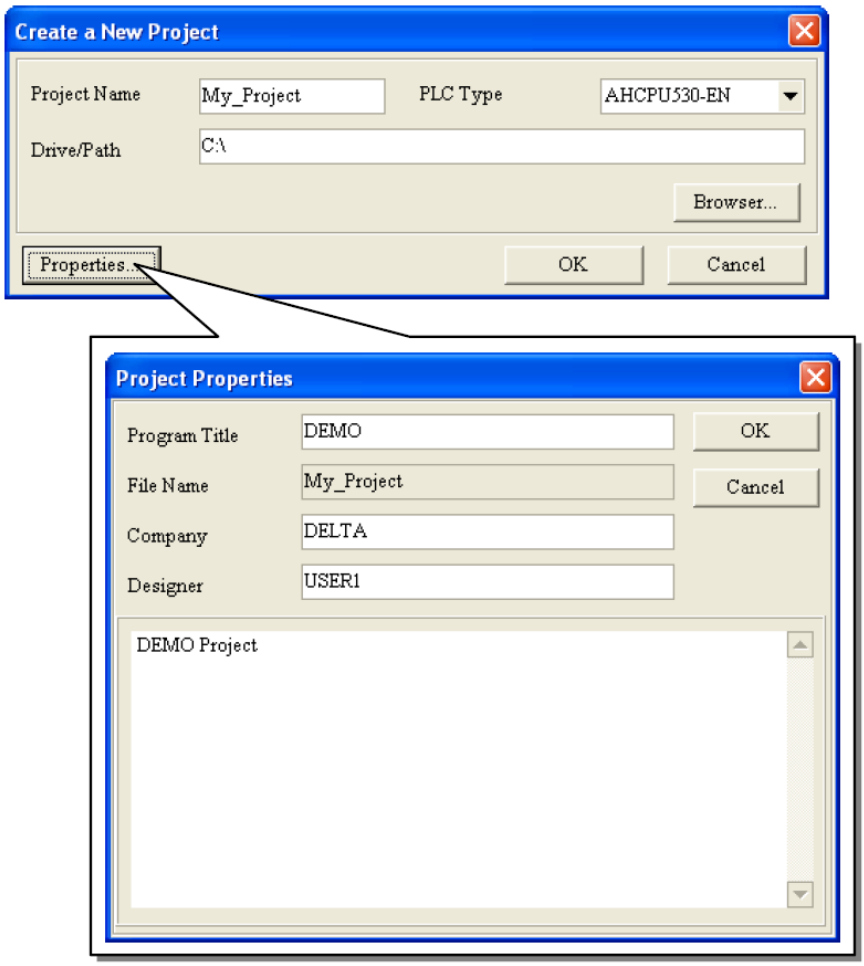
شکل ۲-۹ اجرای نرم افزار ISPSoft و نمای کلی آن

در این مرحله برای ایجاد یک پروژه جدید می‌توانید بر روی گزینه  کلیک کنید.



شکل ۲-۱۰ کلیک بر روی آیکون New برای ایجاد پروژه جدید

در صفحه باز شده "Creat a New Project" می‌توان نام پروژه جدید را در قسمت Project Name و نوع PLC را در قسمت PLC Type و مسیر ذخیره پروژه را در Drive/Path مشخص کرد. همچنین می‌توان با کلیک بر روی دکمه Properties توضیحاتی در رابطه با پروژه اضافه کرد (مانند نام برنامه نویس، شرکت و توضیحات تکمیلی). پس از کلیک بر روی OK پروژه‌ای با مشخصات تعیین شده ایجاد می‌شود.

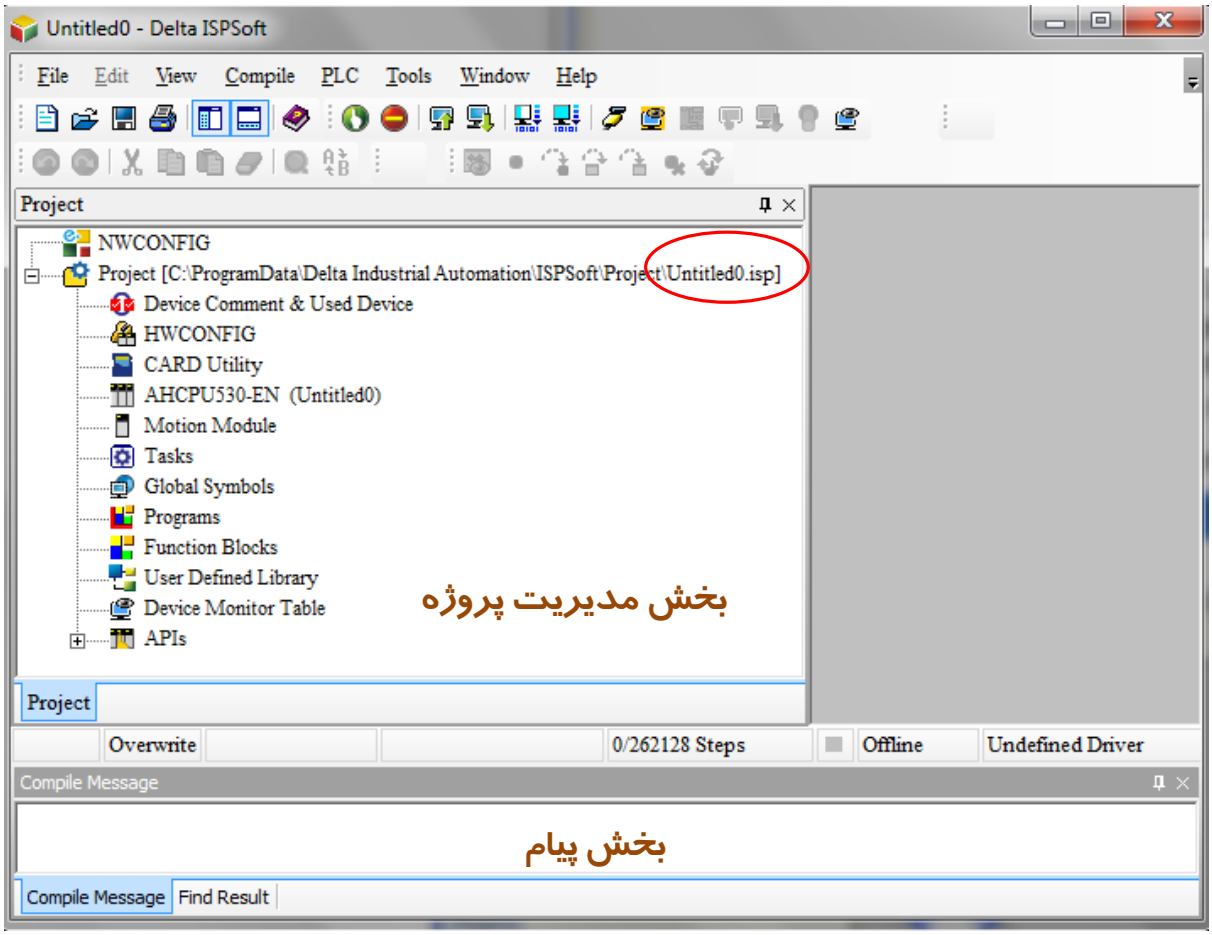


شکل ۲-۱۱ مشخصات مورد نیاز برای ایجاد پروژه جدید

پس از ایجاد پروژه، بخش مدیریت پروژه[۱] مطابق شکل زیر در سمت چپ نرم افزار ایجاد می‌شود، این بخش به صورت درختی بخش‌های مختلف پروژه را لیست می‌کند. همانطور که در شکل مشخص است، پسوند فایل پروژه ایجاد شده به صورت (*.isp) است. در ادامه گزینه‌های لیست شده در این بخش مرور خواهد شد، همچنین بخش پیام[۲] در قسمت زیرین نرم افزار ISPSoft (قسمتی که در شکل زیر به صورت Compile Message مشخص شده) جهت نمایش پیام‌های نرم افزار مورد استفاده قرار می‌گیرد.

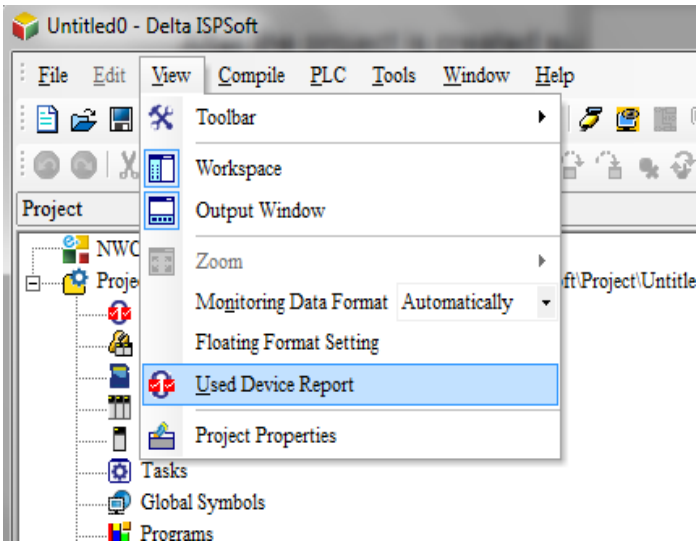
[۱] Project management area

[۲] Message display area



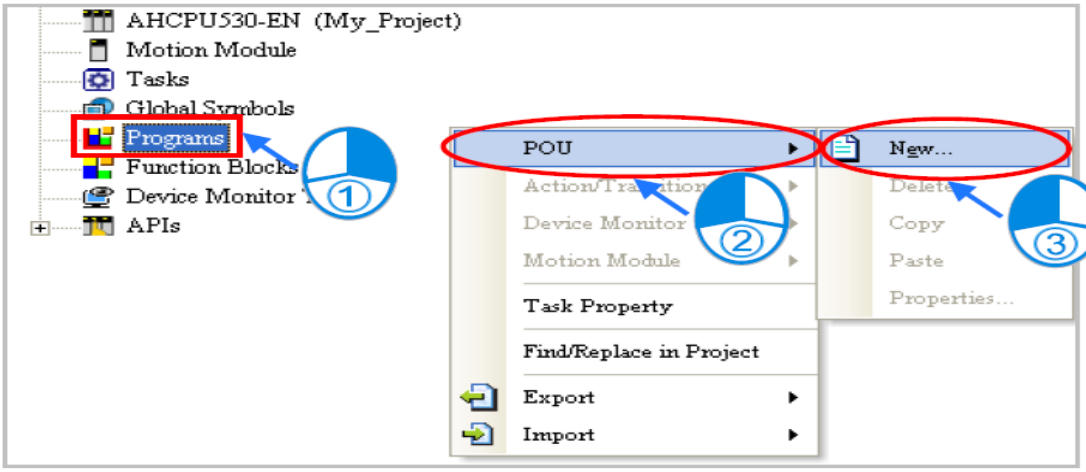
شکل ۲-۱۲ بخش پیام و مدیریت پروژه در ISPSoft

به این نکته توجه شود که اگر پس از ایجاد و یا اجرای پروژه، بخش‌های مدیریت پروژه و پیام نمایش داده نشدند، می‌توان از طریق سربرگ View و (به ترتیب) فعال کردن گزینه‌های  Workspace و  Output Window به آن بخش‌ها دسترسی پیدا کرد.



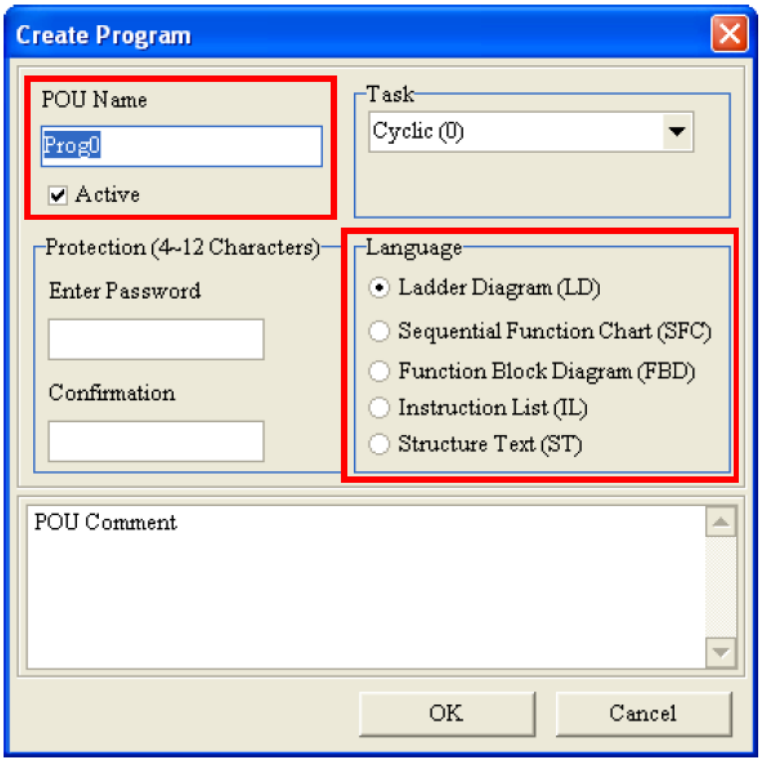
شکل ۲-۱۳ فعال کردن نمایش بخش پیام و مدیریت پروژه در ISPSoft

برای نوشتن برنامه باید بر روی Program در بخش مدیریت پروژه کلیک راست کرده و سپس POU و بعد از آن New را انتخاب نماید.



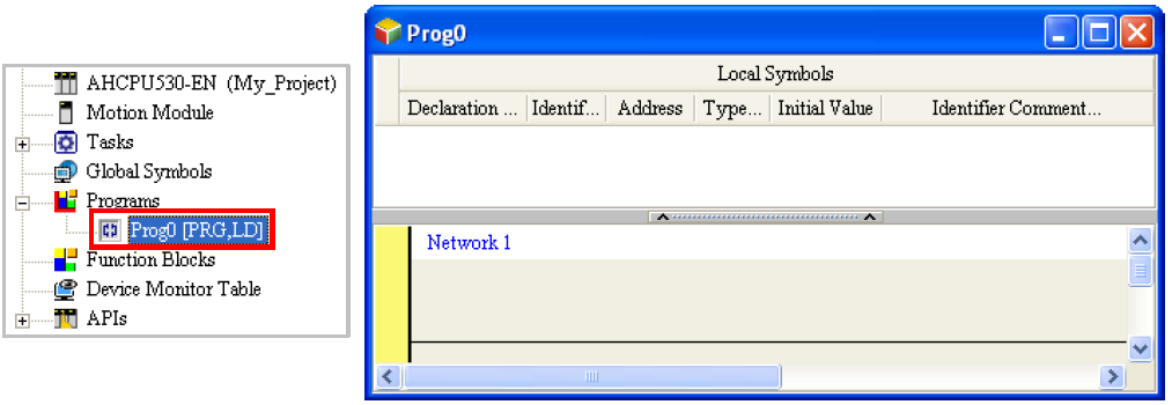
شکل ۲-۱۴ نوشتن یک برنامه جدید در ISPSoft

سپس در پنجره "Create Program" کاربر می‌تواند نام برنامه خود را در قسمت POU Name، نوع برنامه (برنامه دوره‌ای، وقفه خارجی، وقفه زمانی و ...) را در قسمت Task، نوع زبان را در قسمت Language و در صورت نیاز به حفاظت از برنامه کلمه رمز ۴ الی ۱۲ حرفی را در قسمت Protection وارد نماید.



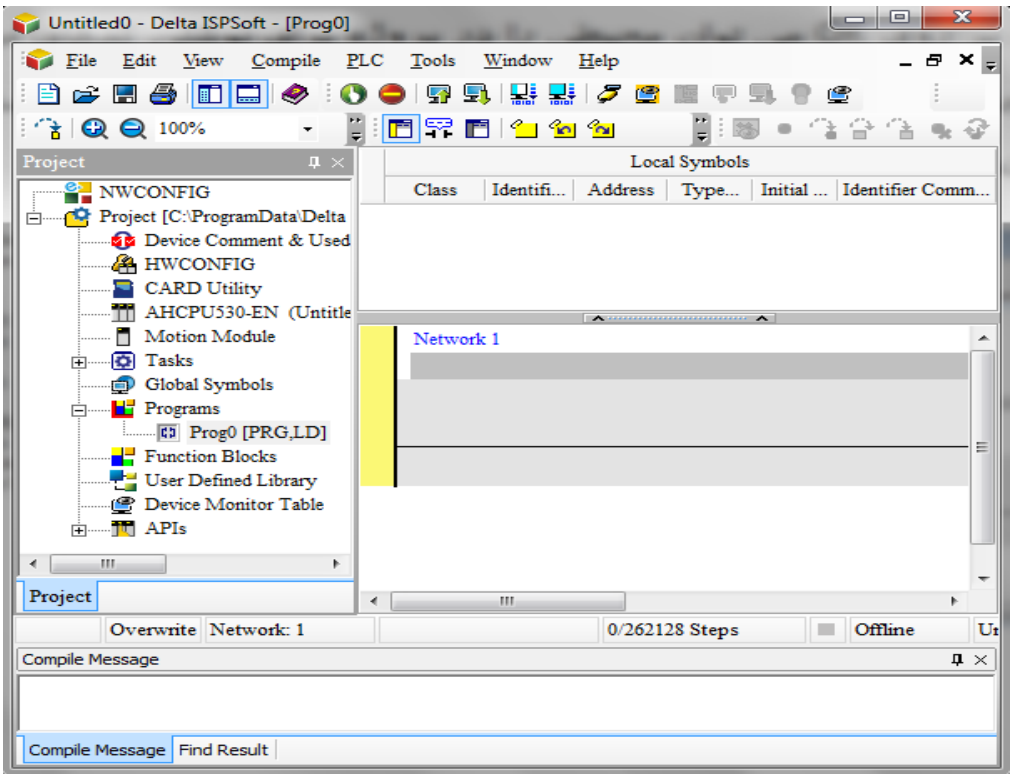
شکل ۲-۱۵ تنظیم مشخصات برنامه جدید در ISPSoft

پس از کلیک بر روی OK محیطی در پروژه برای نوشتن کدهای مورد نیاز PLC ایجاد می‌شود. آیکون برنامه جدید در زیرشاخه Program در بخش مدیریت پروژه ایجاد می‌شود، همچنین در کنار آن محیطی برای نوشتن برنامه ایجاد می‌شود.



شکل ۲-۱۶ محیط برنامه نویسی در ISPSoft

تا این مرحله پنجره زیر را در ISPSoft داریم:

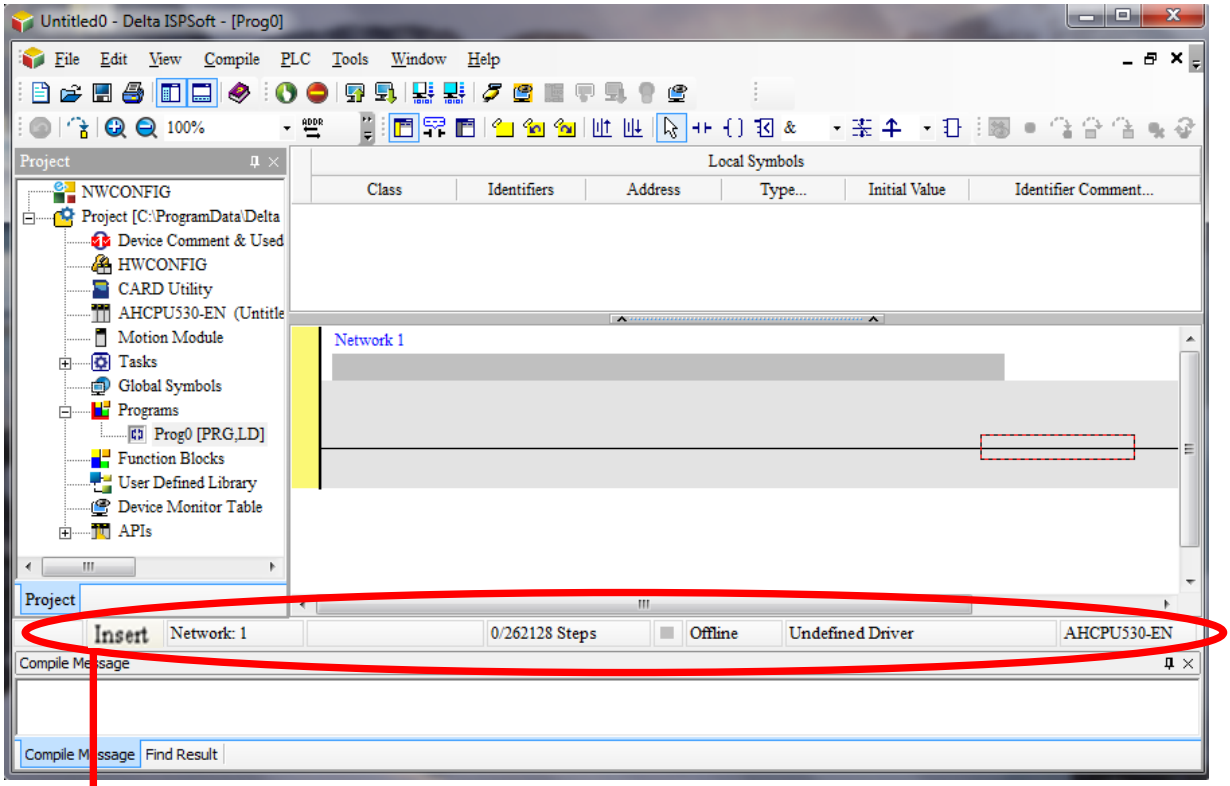


شکل ۲-۱۷ محیط ISPSoft پس از ایجاد پروژه و اولین برنامه در آن

توجه: می‌توانیم با استفاده از منوی Help به اطلاعات مورد نیاز در مورد نرم افزار و برنامه نویسی و PLC ها دست پیدا کنیم. همچنین در بخش‌های مختلف می‌توان با کلیک بر روی آیتم‌های مورد سئوال و فشردن کلید F1 به راهنمای آن بخش دسترسی پیدا کرد.

۲-۳-۱- نوارهای ابزار در محیط نرم افزار ISPSoft

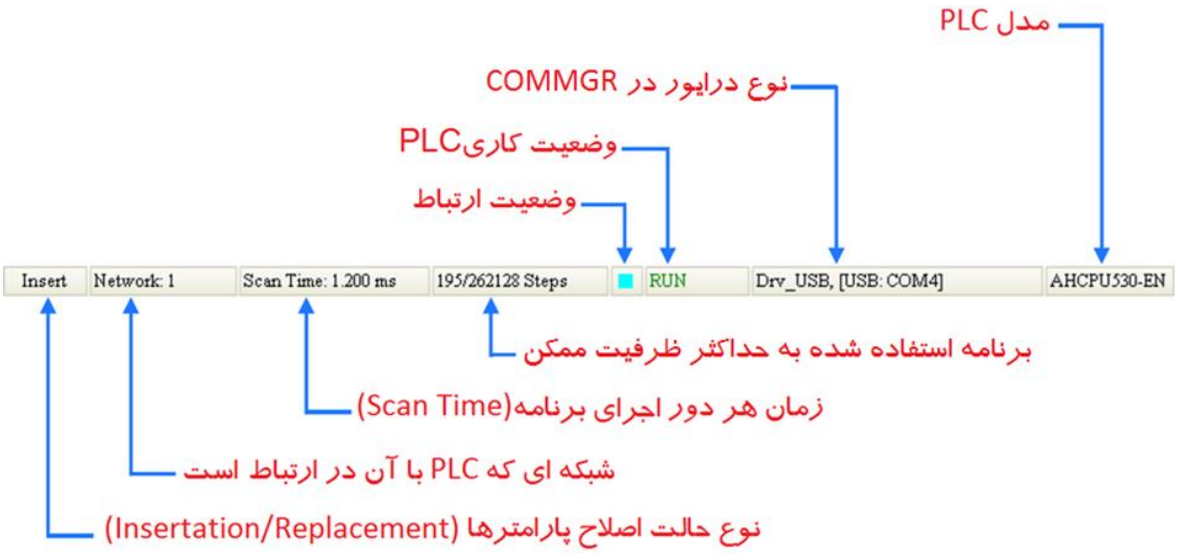
در محیط نرم‌افزار ISPSoft، نوارهای ابزار گوناگونی وجود دارد که در ادامه آن‌ها را مرور خواهیم کرد.



شکل ۲-۱۸ محیط ISPSoft و نوارهای ابزار آن

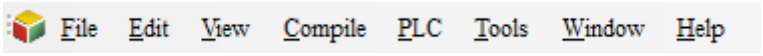
۲-۳-۱-۱- نوار وضعیت

پارامترهای نوار وضعیت در شکل زیر شرح داده شده اند.



شکل ۲-۱۹ نوار وضعیت

۲-۳-۱-۲- نوار منو

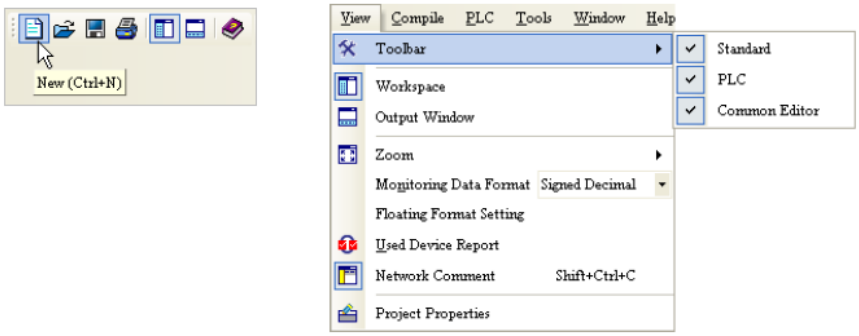


شکل ۲-۲۰ نوار منو

- File: شامل دستورهای مرتبط با ذخیره کردن و بازیابی و امثالهم
- Edit: شامل عملکردهای اصلاح و ویرایش
- View: شامل عملکردهای مرتبط با نمایش اطلاعات پروژه و تنظیم محیط نمایش
- Compile: شامل بررسی ترکیب صحیح کدها و کامپایل برنامه به دستورات اجرایی
- PLC: تنظیمات و کنترل PLC از طریق نرم افزار
- Tools: شامل عملکردهای متنوع جهت تنظیمات شبکه، سخت افزار، برنامه و نرم افزار
- Windows: شامل دستوراتی جهت نحوه نمایش فضای کار و پنجره‌های نرم افزار
- Help: راهنمایی پیرامون نرم افزار و مفاهیم سخت افزاری مرتبط

۲-۳-۱-۳- نوارهای ابزار

نوارهای ابزار شامل پنج نوار متنوع است که به کاربر اجازه می‌دهد تا دسترسی مناسبی به امکانات نرم افزار داشته باشد. کاربر می‌تواند نوارهای ابزار مورد نیاز خود را از طریق سربرگ View در قسمت Toolbar فعال و یا غیرفعال کند. همچنین برای اینکه بتوان از عملکرد آیکونی در هر قسمت از نوارهای ابزار و کلید میانبر[1] آن مطلع شد می‌توان نشانگر را برای مدتی بر روی آیکون مورد نظر نگه داشت تا اطلاعات مورد نظر آن همانند شکل زیر نمایش داده شود.



شکل ۲-۲۱ فعال‌سازی نوارهای ابزار

- نوار فایل: شامل عملکردهایی جهت پردازش پروژه مانند ایجاد و یا ذخیره پروژه

شکل ۲-۲۲ نوار فایل

- نوار ویرایش عملکردهایی جهت ویرایش پروژه مانند کپی ویا تغییر سایز نمایش

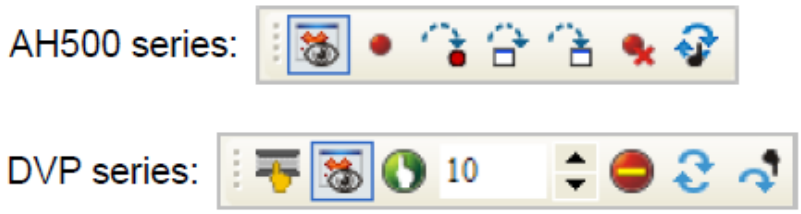
شکل ۲-۲۳ نوار ویرایش

- نوار PLC: شامل عملکردهایی جهت کار و فعال سازی PLC

شکل ۲-۲۴ نوار PLC

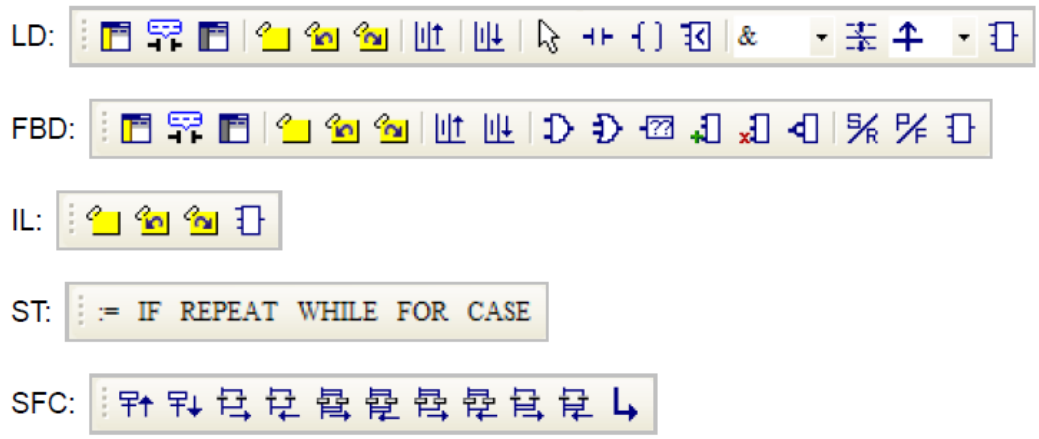
[1] کلید میانبر یا Shortcut، به ما کمک می‌کند که بتوانیم عملکرد خاصی را فقط با استفاده از صفحه کلید فعال کنیم. به عنوان مثال برای ایجاد پروژه جدید می توان از کلیدهای میان بر آن یعنی Ctrl و N استفاده کنیم به این صورت که این دو کلید را به صورت همزمان می‌فشاریم.

- نوار عیب یابی: شامل امکاناتی جهت عیب یابی پروژه (آیکون‌های این نوار با توجه به نوع مدل PLC انتخابی ممکن است متفاوت باشد)



شکل ۲-۲۵ نوار عیب یابی

- نوار برنامه نویسی: شامل توابع پایه و امکاناتی جهت ویرایش برنامه (آیکون های این نوار با توجه به نوع زبان برنامه نویسی انتخاب شده متفاوت است)



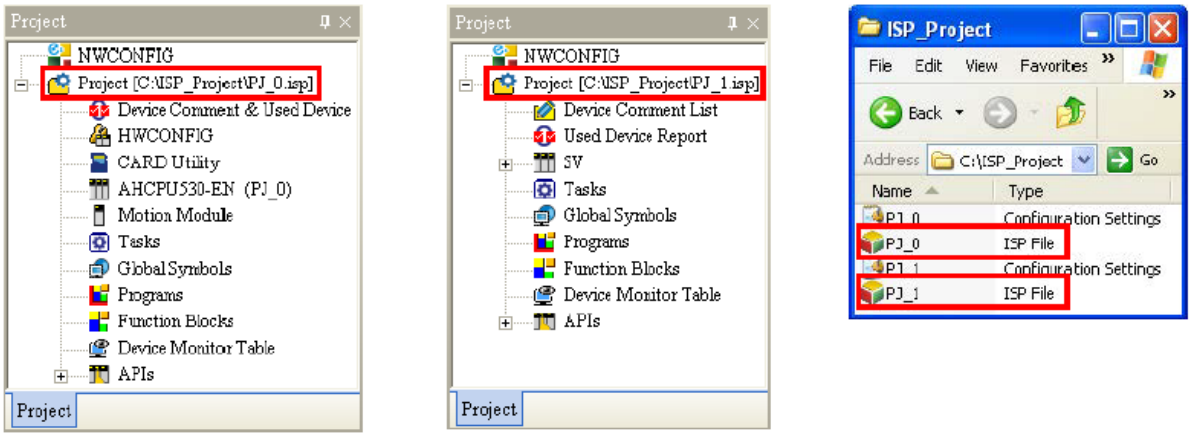
شکل ۲-۲۶ نوار برنامه نویسی

۲-۴- چارچوب کار با پروژه ها در ISPSoft

در ISPSoft می توان به دو صورت منفرد[۱] و یا گروهی[۲] پروژه‌ها را تعریف کرد. در حالت منفرد برنامه برای یک PLC نوشته می‌شود و فایل پروژه با پسوند (*.isp) ذخیره خواهد شد.

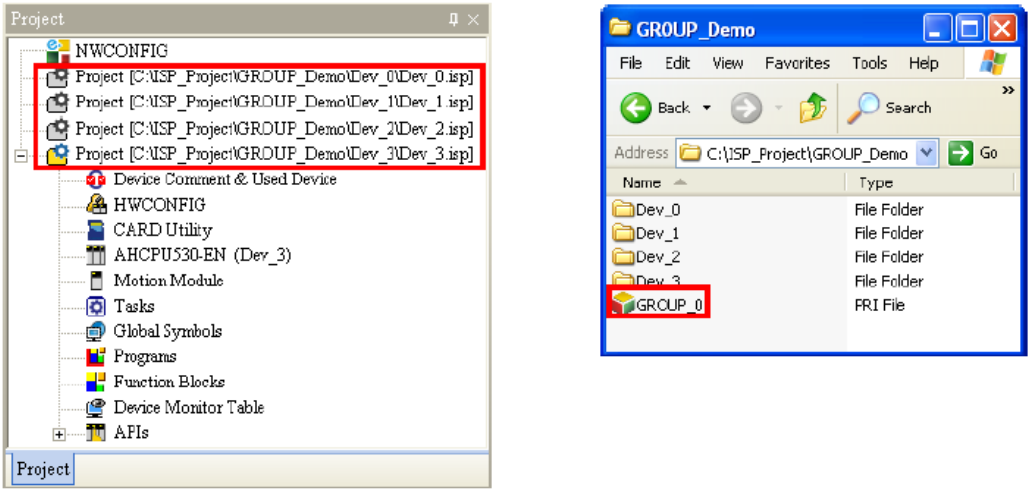
[۱] پروژه‌ای تنها برای یک CPU

[۲] پروژه ای برای چند CPU



شکل ۲-۲۷ پروژه منفرد با پسوند isp.*

همچنین گاهی لازم است پروژه ای توسط چندین PLC یا تجهیزات دیگر به صورت مشترک کنترل شود، در این حالت تجهیزات را باید صورت گروهی برنامه ریزی کرد، بنابراین باید پروژها را به صورت یکجا تحت عنوانی واحد در ISPSoft تعریف نماییم. بدین منظور لازم است تنظیمات شبکه و تجهیزات متصل به آن از طریق NWCONFIG صورت پذیرد[۱]. تعداد پروژه‌هایی که در حالت گروهی می‌توان ایجاد نمود متناظر با تجهیزات متصل به شبکه است و محدودیت خاصی از طرف نرم افزار برای آن وجود نخواهد داشت. فایل ایجاد شده در حالت گروهی پسوند (pri.*) خواهدداشت و پسوند هر پروژه در گروه همانند گذشته (isp.*) خواهد بود. در واقع هرکدام از زیر پروژه‌های پروژه گروهی نماینده یک ایستگاه مستقل دارای کنترل کننده است.

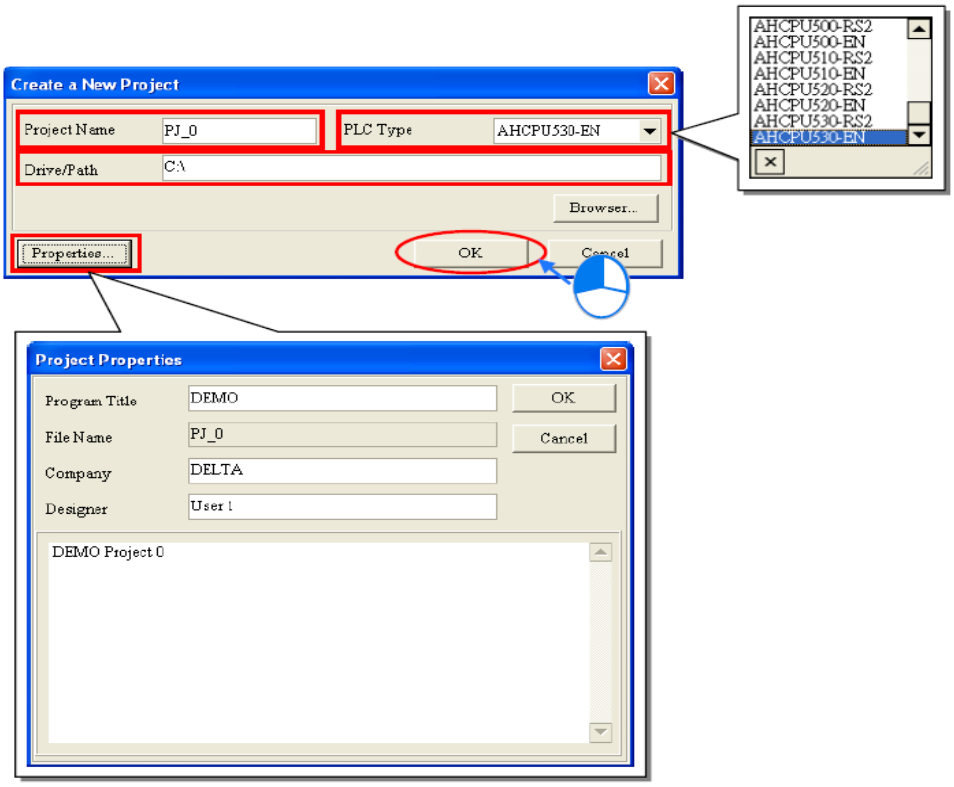


شکل ۲-۲۸ پروژه گروهی با پسوند pri.*

[۱] تنظیمات شبکه در NWCONFIG بسیار راحت است و در ادامه آن را مرور خواهیم کرد.

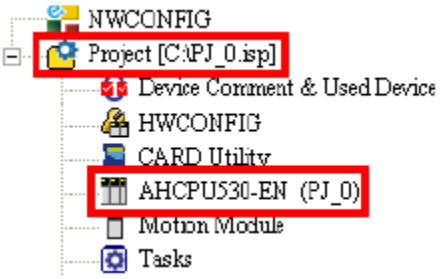
۲-۴-۱- نحوه ایجاد پروژه منفرد جدید

برای این کار از منوی فایل گزینه New را انتخاب نموده و یا اینکه بر روی گزینه  کلیک نمایید. در صفحه باز شده می‌توانید نام پروژه، مسیر ذخیره سازی، نوع PLC و نکات مربوط به پروژه را قید نمایید. در انتها با کلیک بر روی دکمه OK پروژه جدید ایجاد خواهد شد.



شکل ۲-۲۹ ایجاد و تنظیمات پروژه منفرد جدید

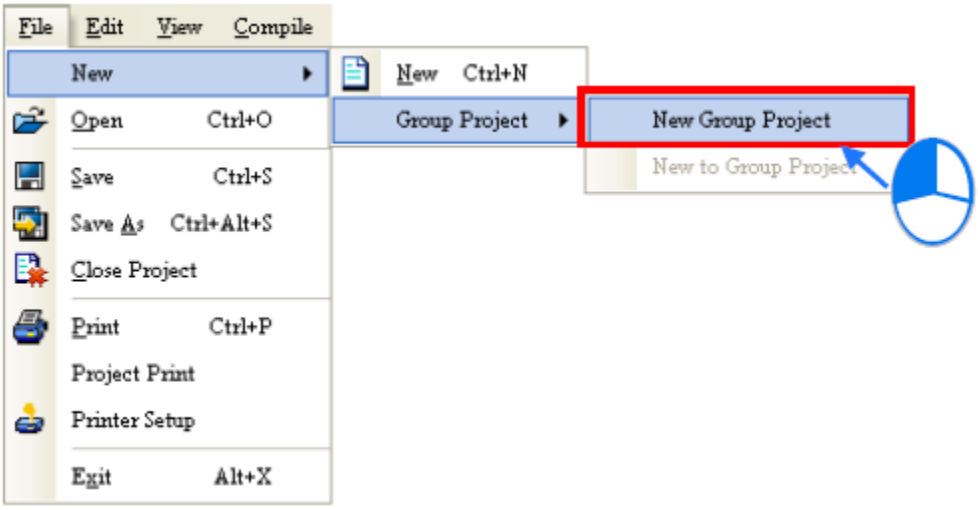
در ادامه نام پروژه و مسیر ذخیره سازی آن به همراه مدل PLC انتخاب شده در قسمت مدیریت پروژه نمایش داده خواهد شد.



شکل ۲-۳۰ پروژه منفرد در بخش مدیریت پروژه

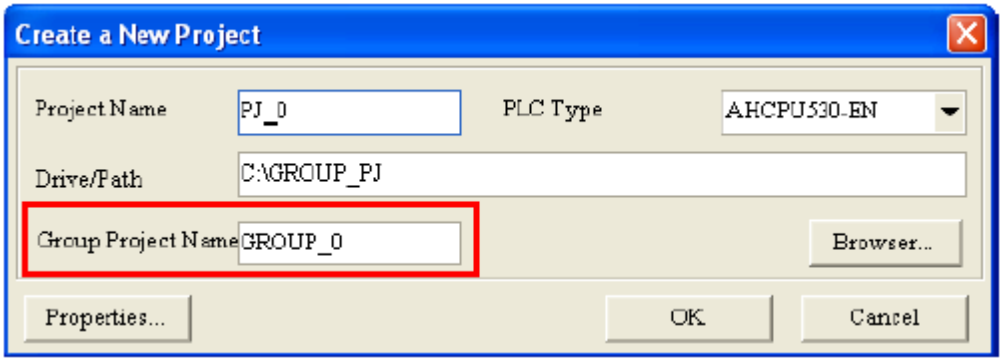
۲-۴-۲- نحوه ایجاد پروژه گروهی جدید

از منوی فایل، Group Project و سپس New Group Project را انتخاب نمایید.



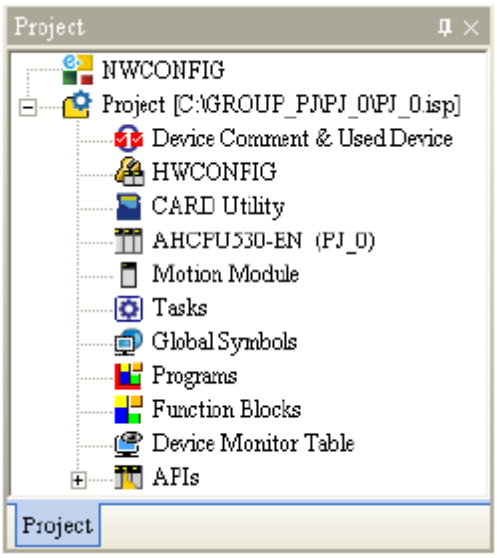
شکل ۲-۳۱ ایجاد پروژه گروهی جدید

در این مرحله باید فقط یکی از پروژه‌های منفرد درون پروژه گروهی را تعریف کرد. این قسمت مشابه ساخت پروژه منفرد است با این تفاوت که در تعریف پروژه مورد نظر باید نام پروژه گروهی مورد نظر را نیز در قسمت Group Project Name تایپ نماید.



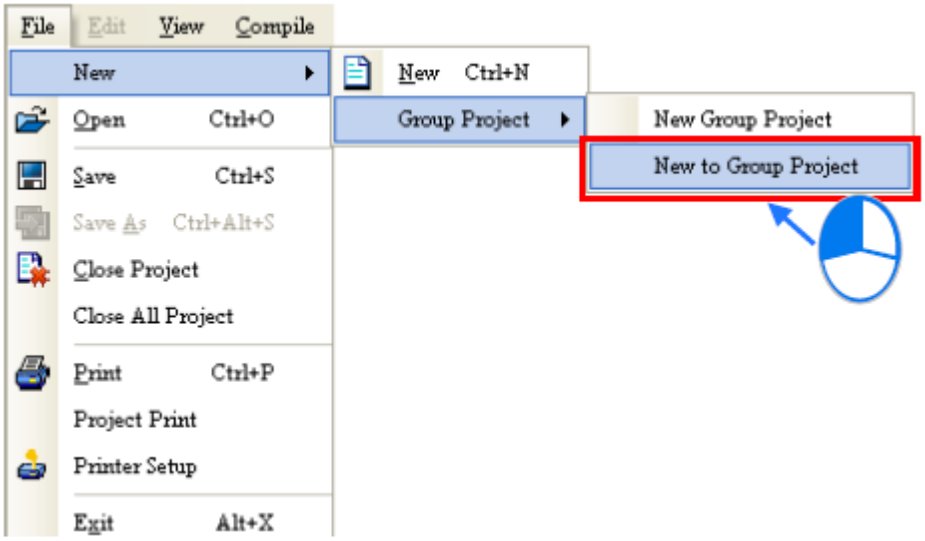
شکل ۲-۳۲ تنظیمات پروژه گروهی جدید

در این مرحله تنها یک پروژه در بخش "مدیریت پروژه" ایجاد می‌شود که دقیقا مشابه یک پروژه منفرد است با این تفاوت که تک پروژه "درون پروژه گروهی" را همانند پروژه منفرد نمی‌توان "Save As" کرد و در مسیری دیگر ذخیره کرد و باید از دستور Export برای استخراج برنامه‌های موجود در پروژه استفاده کرد.

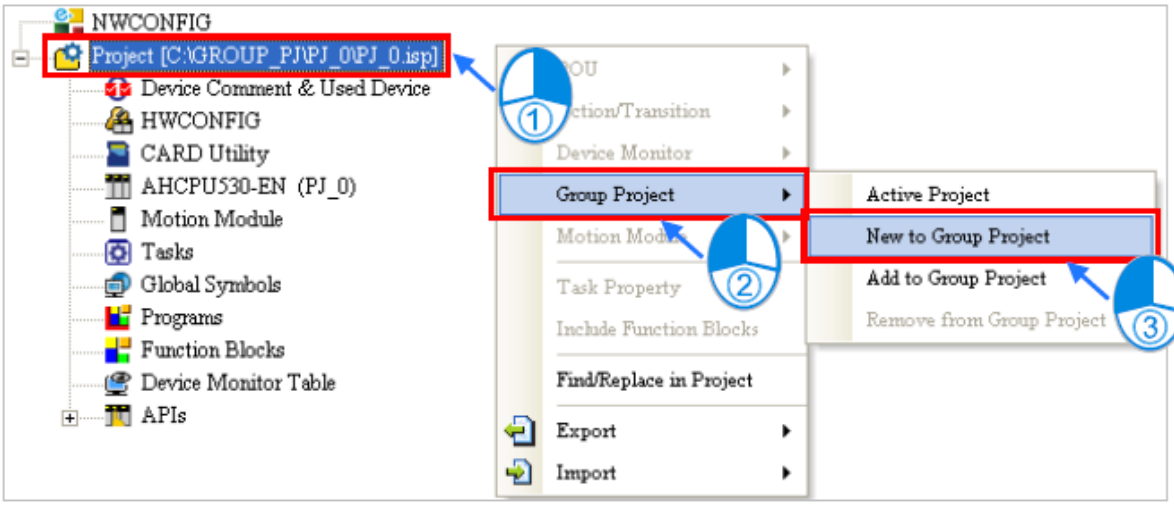


شکل ۲-۳۳ بخش مدیریت پروژه، اولین پروژه در یک پروژه گروهی

حال برای اینکه بتوانیم پروژه ای به پروژه گروهی خود اضافه کنیم، می‌توانیم در منوی فایل، در قسمت NEW، بر روی Group Project و سپس "New **to** Group Project" کلیک نماییم، همچنین می‌توانید این کار را با کلیک راست بر روی پروژه لیست شده در بخش مدیریت پروژه دنبال کنید، در این حالت می‌بایست Group Project وسپس New to Group Project را انتخاب نمایید.

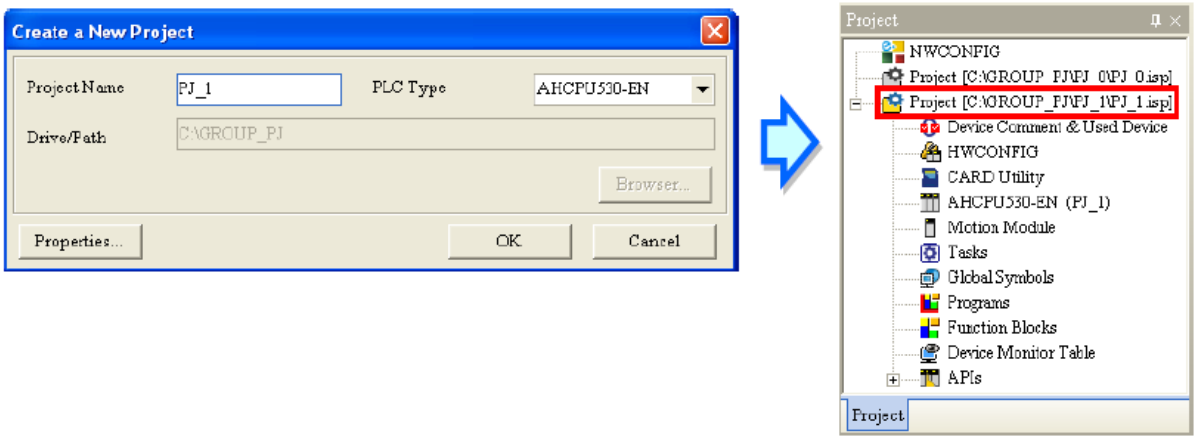


شکل ۲-۳۴ اضافه کردن پروژه به پروژه گروهی- روش اول



شکل ۲-۳۵ اضافه کردن پروژه به پروژه گروهی- روش دوم

با توجه به این موضوع که در آن واحد تنها می‌توان به ویرایش یکی از پروژه‌های درون گروه اقدام کرد (پروژه ای که فعال شده است) و اینکه در زمان ایجاد پروژه‌ای جدید، آن پروژه به عنوان پروژه فعال مشخص می‌شود و پروژه فعال قبلی بسته می‌شود نیاز است تا قبل از ایجاد پروژه جدید، پروژه قبلی خود را ذخیره نمایید[1]. در زمان ایجاد پروژه جدید به علت مشخص بودن مکان و نام پروژه گروهی تنها نیاز است نام پروژه جدید و نوع تجهیزات آن را مشخص نماییم.

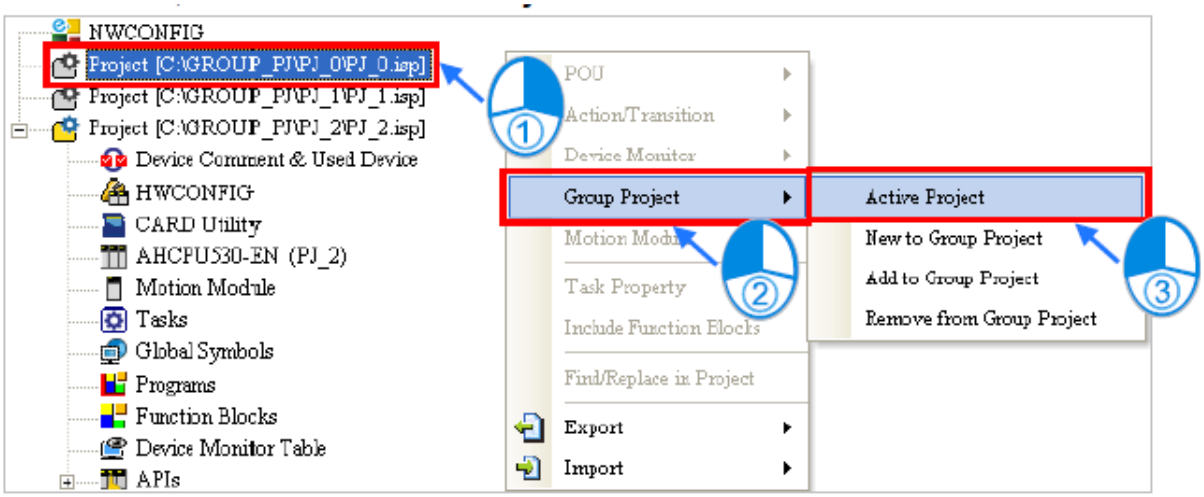


شکل ۲-۳۶ نحوه تنظیمات پروژه جدید در پروژه گروهی

همانطور که در عکس فوق مشخص است پروژه جدید که پروژه فعال گروه است به رنگ زرد و مابقی پروژه‌های غیر فعال به رنگ خاکستری درآمده‌اند. در این مرحله اگر بخواهیم پروژه دیگری را فعال کنیم

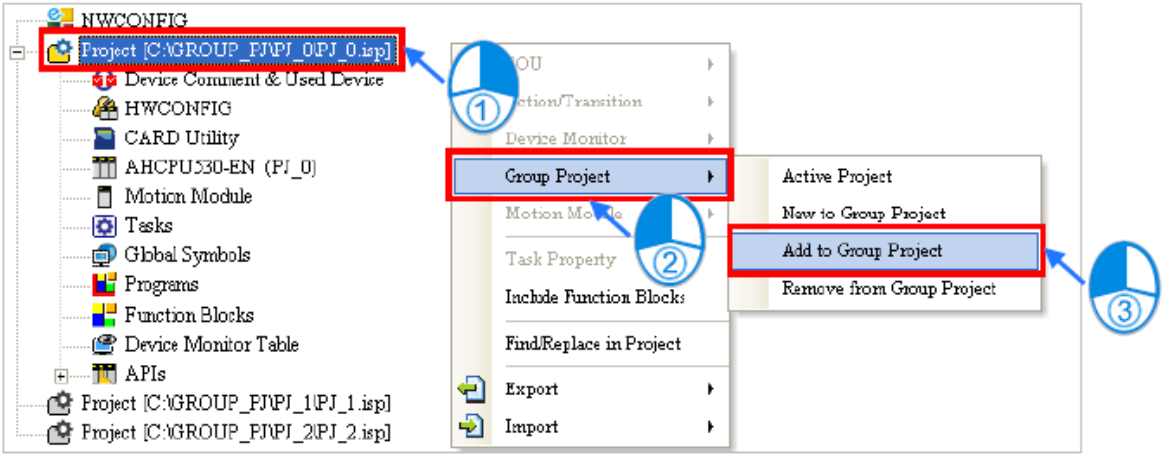
[1] اگرچه نرم افزار قبل از بسته شدن پروژه‌ها در مورد آن‌ها از کاربر سئوال می‌کند.

می‌توانیم بر روی آن دوبار کلیک کرده و یا اینکه پس از کلیک راست بر روی آن، Group Project و سپس Active Project را انتخاب نماییم.



شکل ۲-۳۷ فعال کردن پروژه در پروژه ای گروهی

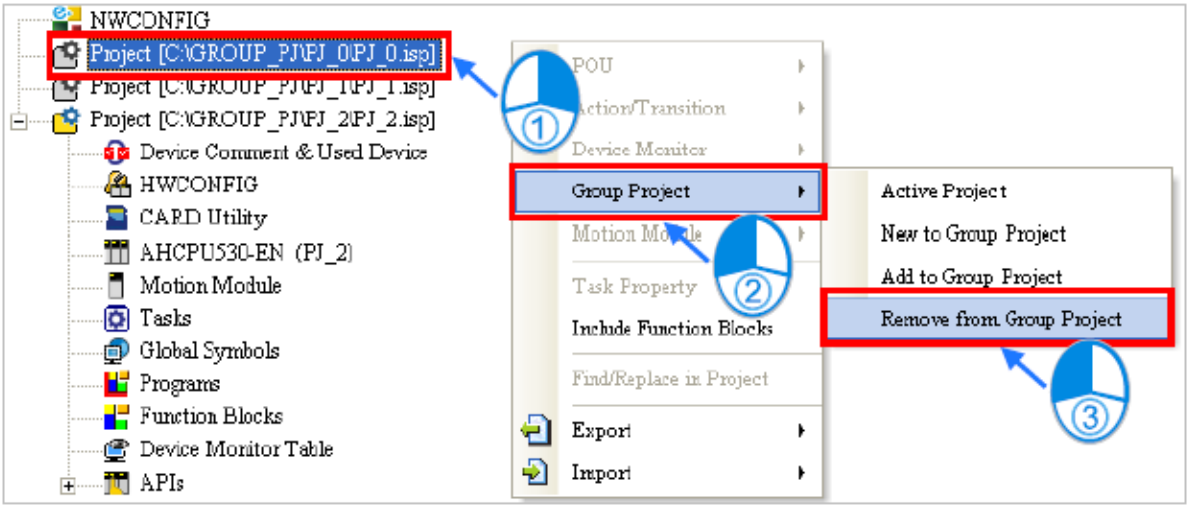
اگر بخواهیم پروژه انجام شده ای (با پسوند isp.* چه به عنوان پروژه منفرد و چه به عنوان بخشی از یک پروژه گروهی) را به پروژه گروهی خود اضافه کنیم می‌توانیم بر روی پروژه فعال کلیک راست کرده و در قسمت Group Project گزینه **Add** to Group Project را انتخاب نماییم. در این مرحله می‌توانیم فایل پروژه مورد نظر خود را انتخاب کرده و آن را به مکان پروژه گروهی فعلی **کپی** کرد. (بدون نگرانی در مورد آسیب دیدن فایل مرجع)



شکل ۲-۳۸ اضافه کردن پروژه به پروژه ای گروهی

اگر بخواهیم پروژه‌ای تکی را در پروژه گروهی حذف کنیم، می‌توانیم بر روی آن کلیک راست کرده و در قسمت Group Project گزینه Remove From Group Project را انتخاب نماییم. البته در حالتی که فقط یک پروژه در گروه باقی مانده باشد، حذف آن امکان پذیر نمی‌باشد. توجه کنید در حالتی که پروژه-

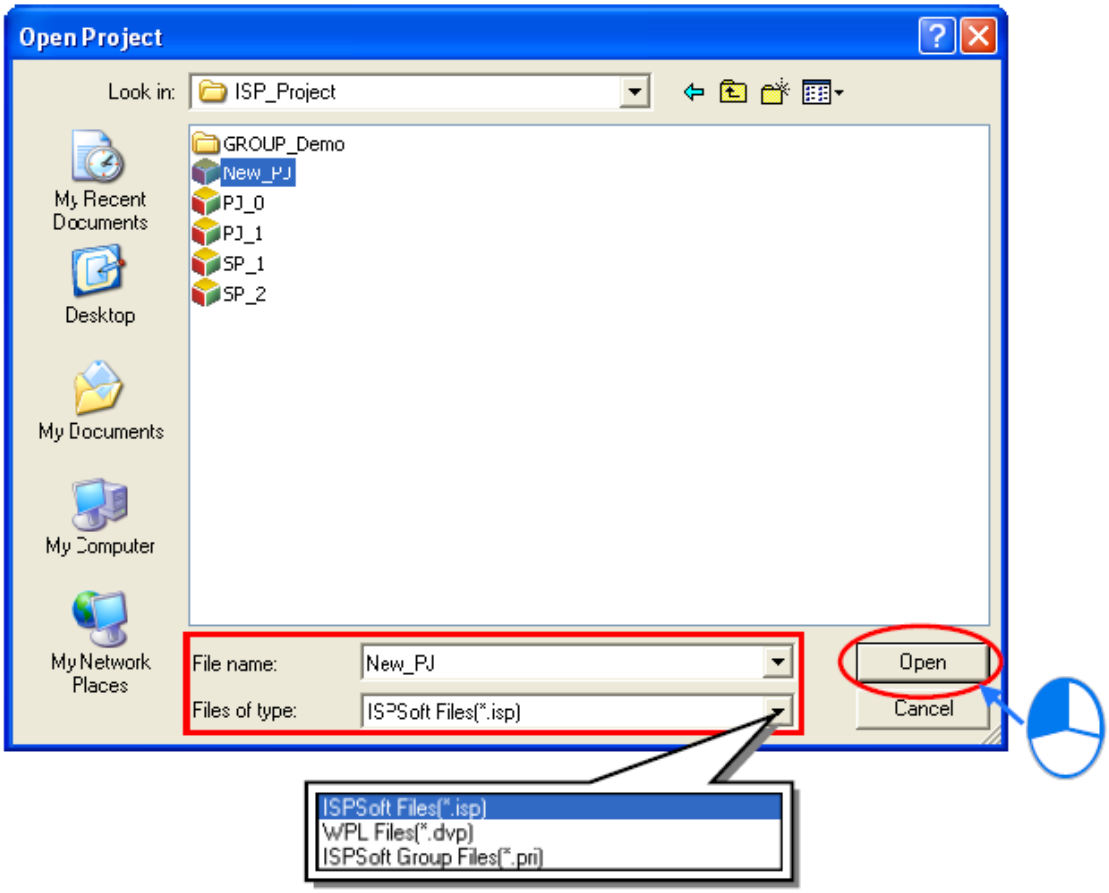
ای را حذف می‌کنیم، صرفا نام آن در برنامه حذف می‌شود و فایل‌های آن در فولدر گروه همچنان باقی خواهند ماند و برای پروژه‌های آتی قابل استفاده خواهند بود.



شکل ۲-۳۹ حذف پروژه در پروژه ای گروهی

۲-۴-۳- مدیریت پروژه ها

در هر دو نوع پروژه منفرد و گروهی برای ذخیره پروژه مورد نظر می‌توان از منوی فایل، گزینه Save را انتخاب یا بر روی در نوار ابزار کلیک کرد. کاربر برای باز کردن پروژه هایی که از قبل ذخیره شده است نیز می‌تواند از منوی فایل، گزینه Open را انتخاب نماید و یا بر روی کلیک نماید.



شکل ۲-۴۰ ذخیره پروژه

همانطور که مشخص است، نرم افزار ISPSoft سه فرمت متفاوت پروژه را پشتیبانی می‌کند.

جدول ۲-۲: فرمت فایل هایی که ISPSoft پشتیبانی می‌کند

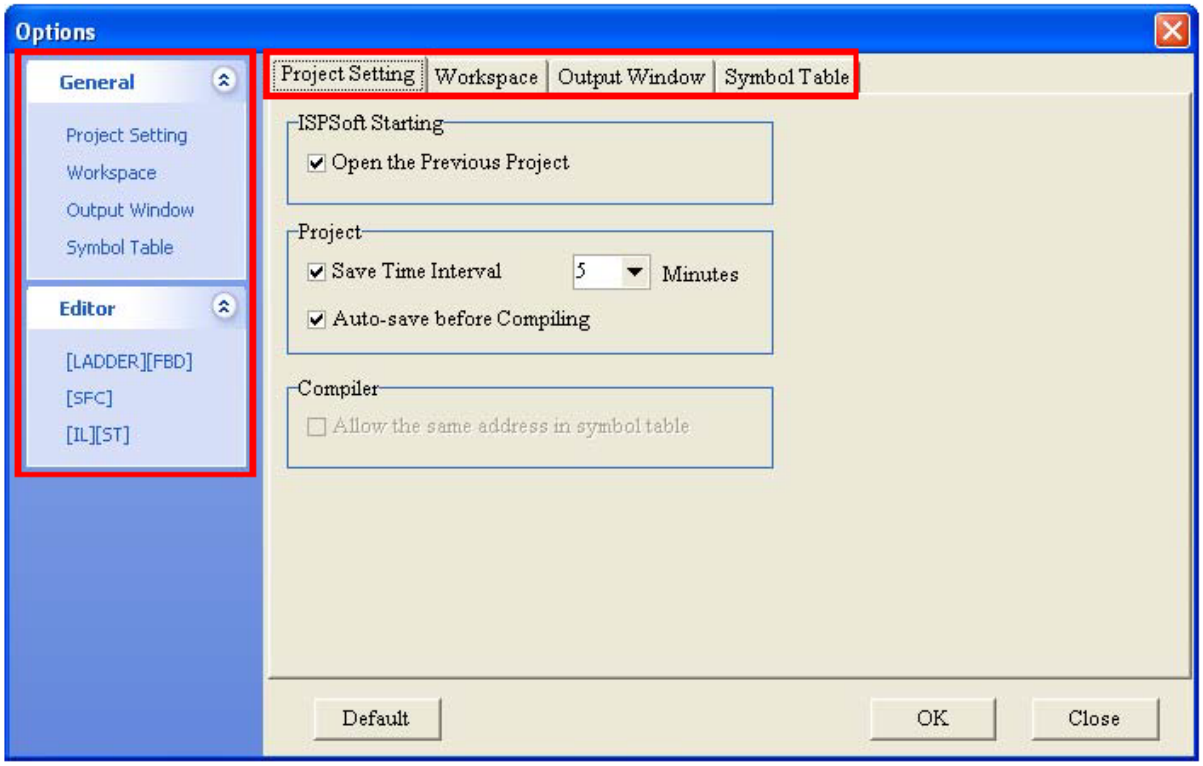
نوع فایل	توضیحات
فایل‌های نرم افزار ISPSoft با پسوند (*.isp)	پسوند فایل‌های ایجاد شده برای پروژه‌های منفرد. همچنین فایل‌های قدیمی با پسوند (*.dvp) هنگامی که بر روی ISPSoft اجرا می‌شوند، پسوند فایل خروجی آن‌ها به (*.isp) تغییر می کند.
فایل‌های نرم افزار WPLSoft با پسوند (*.dvp)	زمانی که اینگونه فایل‌های با پسوند قدیمی (*.dvp) بر روی ISPSoft اجرا می‌شود، پسوند فایل خروجی به

	(*.isp) تغییر می کند.
فایل‌های گروهی نرم افزار ISPSoft با پسوند (*.pri)	پسوند فایل پروژه‌های گروهی ایجاد شده به وسیله ISPSoft

برای بستن پروژه جاری می‌توان از منوی فایل گزینه Close را انتخاب کرد. (برای بستن پروژه ها در پروژه گروهی باید گزینه Close All Project را انتخاب نماییم)

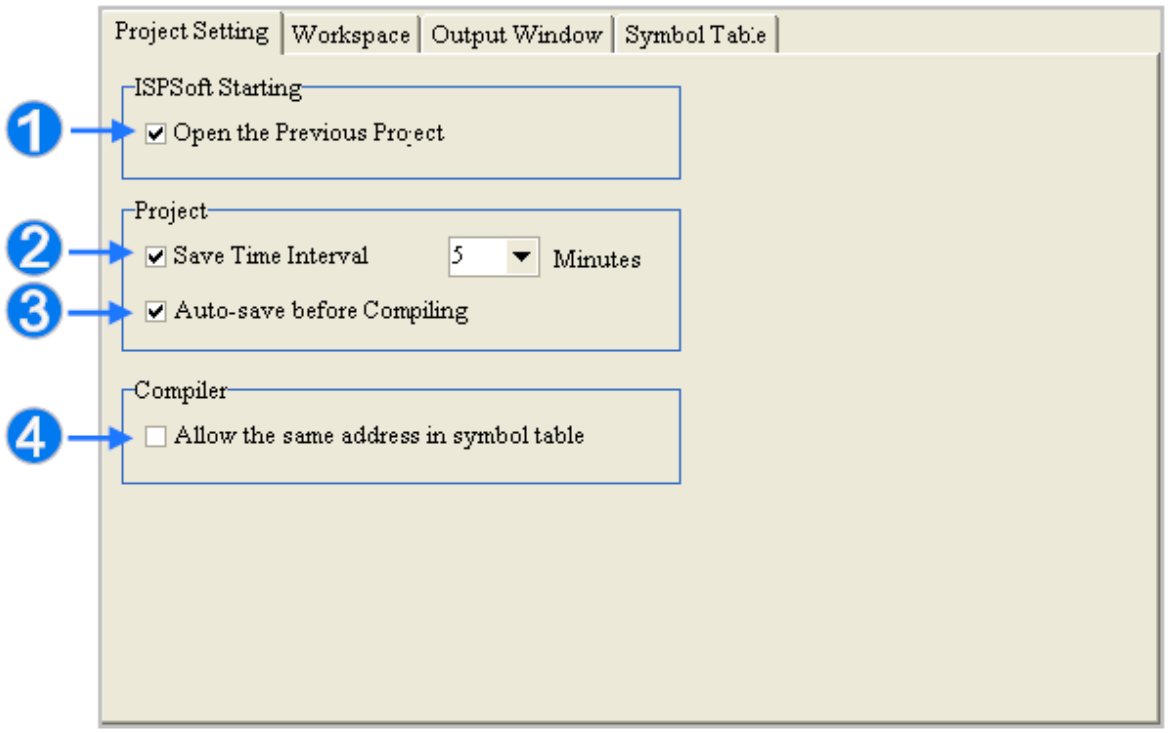
۲-۵- تنظیمات مقدماتی در ISPSoft

پس از کلیک بر روی گزینه Options در سربرگ Tools صفحه‌ای شامل دو بخش General و Editor ظاهر می‌شود که هر کدام دارای قسمت‌های متعددی تحت عنوان سربرگ‌ها مختلف می‌باشند. در این دو بخش و سربرگ‌های مختلف آن‌ها می‌توان تنظیمات مورد نیاز کاربر را در نرم‌افزار ISPSoft پیاده نمود.



شکل ۲-۴۱ صفحه تنظیمات ISPSoft

۲–۵–۱– سربرگ Project Setting در قسمت General



شکل ۲-۴۲ صفحه تنظیمات ISPSoft–Project Setting

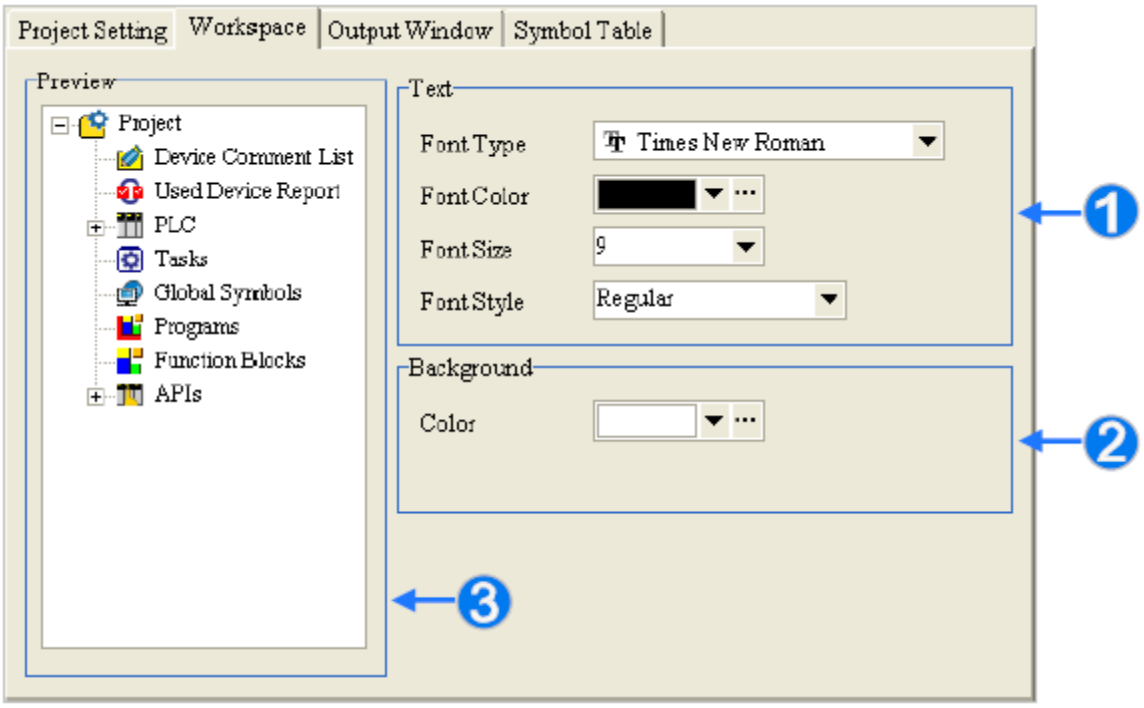
1 اگر این گزینه فعال باشد، پس از بسته شدن ISPSoft پروژه فعال در آن پس از شروع مجدد اجرا خواهد شد. (به عبارتی نرم افزار با آخرین پروژه بالا می آید)

2 کاربران در این قسمت می‌توانند تنظیم کنند که برنامه هر چند دقیقه یکبار از پروژه نسخه پشتیبان تهیه نماید تا در صورت بروز اتفاقی غیر منتظره بتوانند به نسخه پشتیبان پروژه دسترسی داشته باشند.

3 در صورت فعال شدن این گزینه، در هنگام کامپایل شدن، برنامه پروژه را به صورت اتوماتیک ذخیره می‌نماید.

4 در صورتی که این گزینه فعال باشد، می‌توان از چند سیمبول برای یک آدرس حافظه و یا پورت استفاده کرد.

۲–۵–۲– سربرگ Workspace در قسمت General



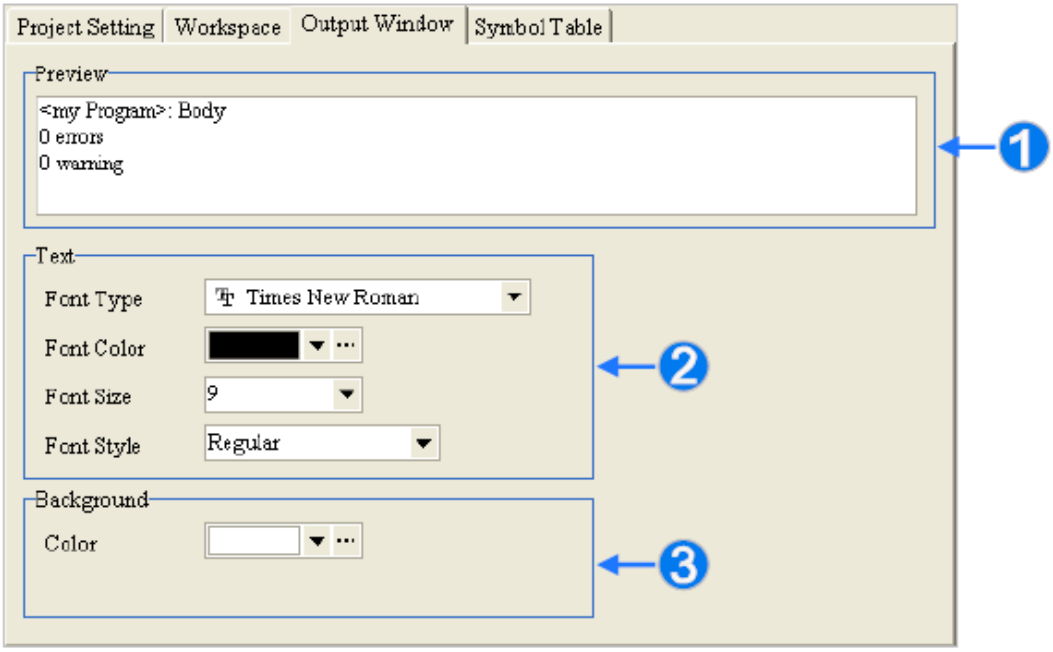
شکل ۲-۴۳ صفحه تنظیمات Workspace–ISPSoft

1 جهت تعیین ویژگی‌های فونت بخش مدیریت پروژه

2 جهت تعیین رنگ پس زمینه در بخش مدیریت پروژه

3 پیش نمایش بخش مدیریت پروژه

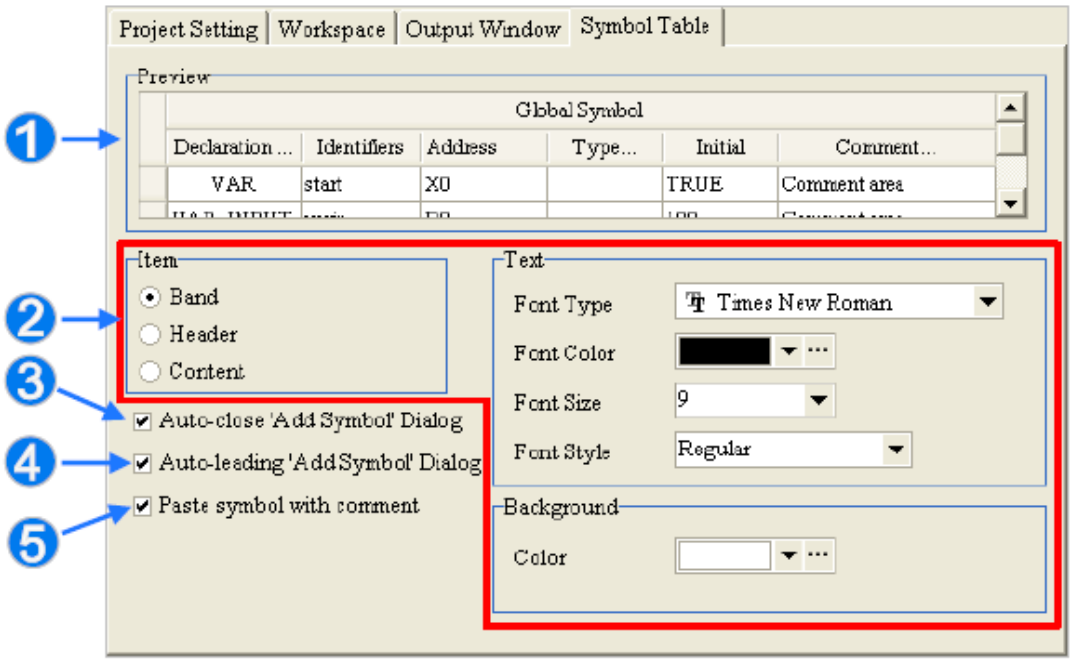
۲–۵–۳– سربرگ Output Window در قسمت General



شکل ۲-۴۴ صفحه تنظیمات Output Window–ISPSoft

1. پیش نمایش بخش پیام
2. جهت تعیین ویژگی‌های فونت بخش پیام
3. جهت تعیین رنگ پس زمینه بخش پیام

۲–۵–۴– سربرگ Symbol Table در قسمت General



شکل ۲-۴۵ صفحه تنظیمات Symbol Table–ISPSoft

1 پیش نمایش بخش سیمبول‌ها (نمادها)[1]

2 کاربر می‌تواند فونت‌های بخش های تیتر، سرستون‌ها و خانه های جدول را به ترتیب با انتخاب Band، Header و یا Content در قسمت Item تعیین نماید. برای اینکار می‌بایست ابتدا Item مورد نظر را انتخاب و سپس ویژگی های آن را در بخش‌های Text و Background تعیین کرد.

3 در صورت فعال بودن این گزینه، پنجره Add Symbol به صورت اتوماتیک پس از وارد شدن سیمبول بسته خواهد شد.

4 در صورت فعال بودن این گزینه، پنجره Add Symbol به صورت خودکار پس از استفاده کاربر از نمادی تعریف نشده و فشردن کلید Enter باز خواهد شد.

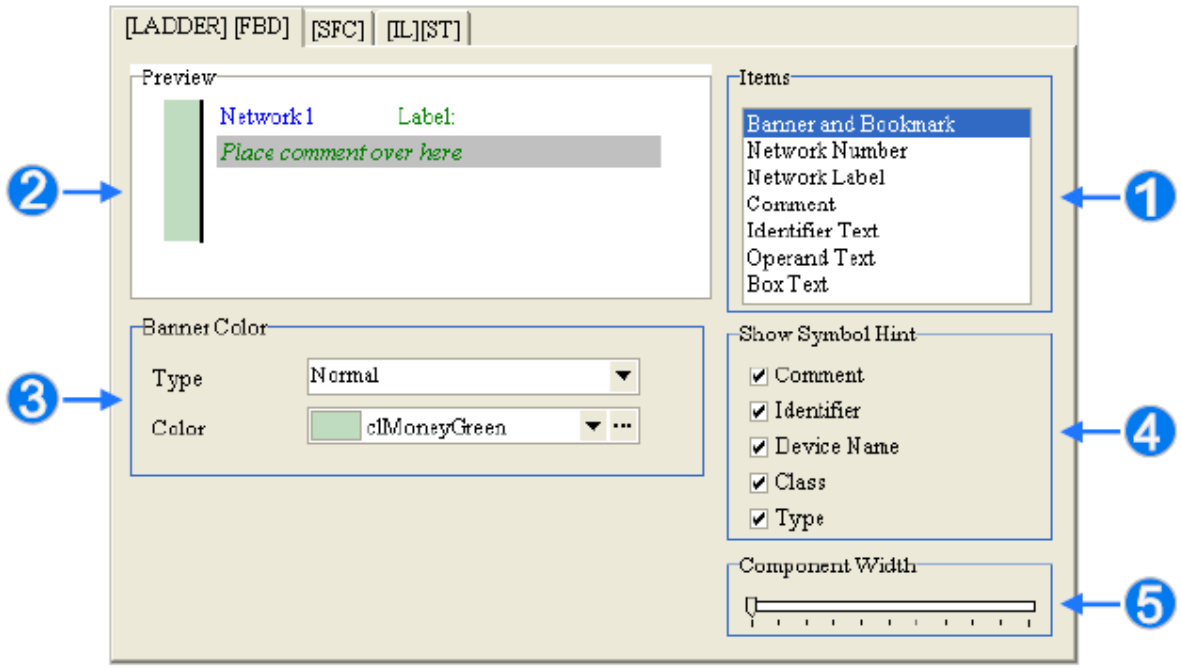
5 در صورتی که این گزینه فعال باشد، سیمبول کپی یا کات شده در صورت الحاق[2] همراه با توضیحات خواهد بود.

---

[1] با این بخش (Symbols) در بخش‌های آینده آشنا خواهید شد.

[2] Past

۲–۵–۵– سربرگ [LADDER] [FBD] در قسمت Editor



شکل ۲-۴۶ صفحه تنظیمات ISPSoft–[LADDER] [FBD]

1 انتخاب آیتم مورد نظر

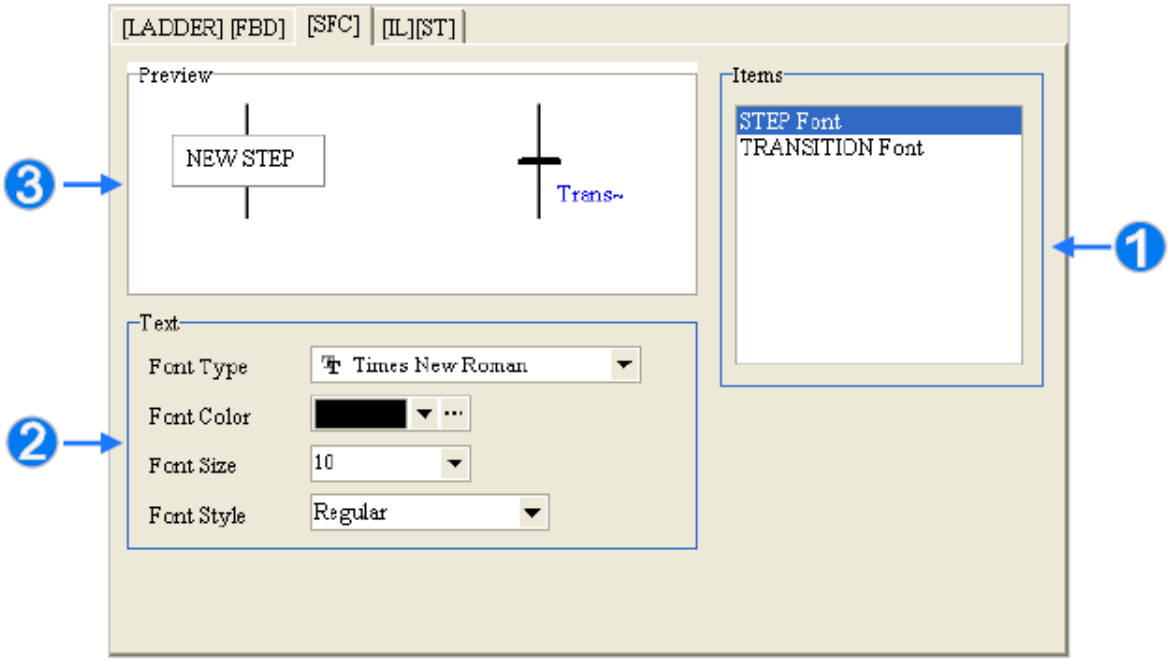
2 پیش نمایش تغییرات در نظر گرفته شده در صفحه

3 تعیین ویژگی های ظاهری و رنگ فونت

4 مواردی که در این قسمت مشخص می‌شوند پس از آنکه کاربر مدتی نشانگر موس را بر روی آیتم مشخص شده نگه دارد، ظاهر می‌شوند.

5 تعیین پهنای بلوک ها

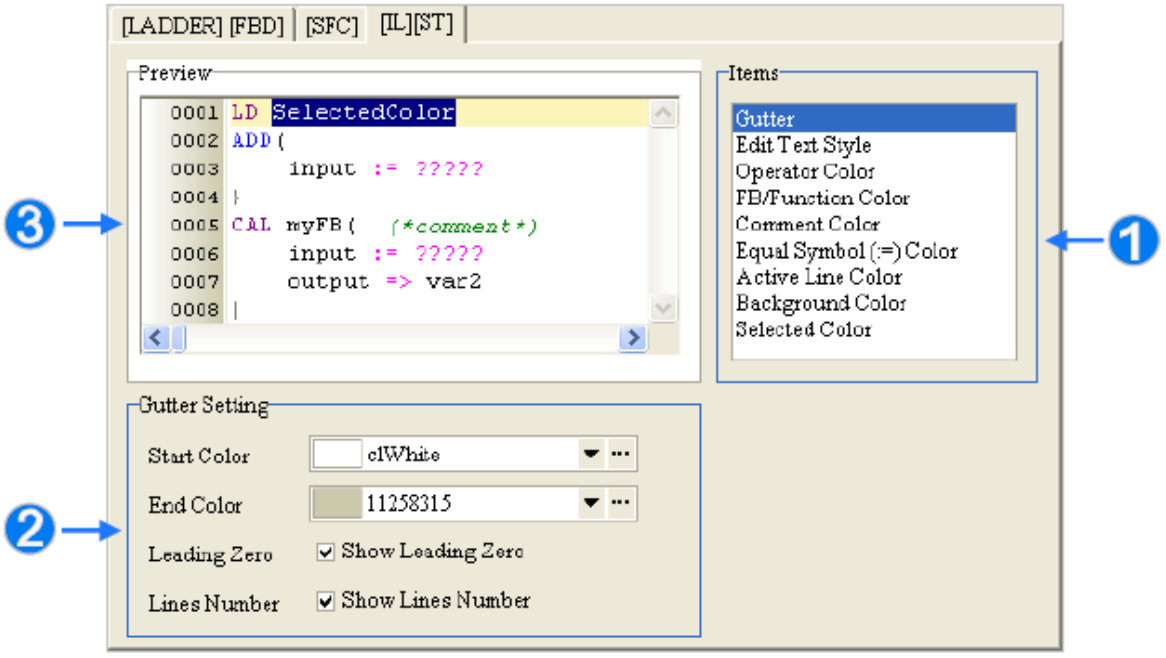
۲–۵–۶– سربرگ [SFC] در قسمت Editor



شکل ۲-۴۷ صفحه تنظیمات ISPSoft–[SFC]

1. انتخاب آیتم مورد نظر برای تغییر فرمت نوشتار
2. تعیین ویژگی های ظاهری و رنگ فونت
3. پیش نمایش تغییرات در نظر گرفته شده در صفحه

۲–۵–۷– سربرگ [ST] [IL] در قسمت Editor



شکل ۲-۴۸ صفحه تنظیمات ISPSoft–[IL] [ST]

1. انتخاب آیتم مورد نظر
2. تعیین ویژگی های بصری صفحه ویرایش پروژه
3. پیش نمایش تغییرات در نظر گرفته شده در صفحه

در صورتی که دکمه Default انتخاب شود، کلیه گزینه‌ها به حالت پیش فرض برخواهد گشت. تنظیمات مورد نظر پس از فشردن دکمه OK به وسیله کاربر ذخیره و اجرا خواهد شد.

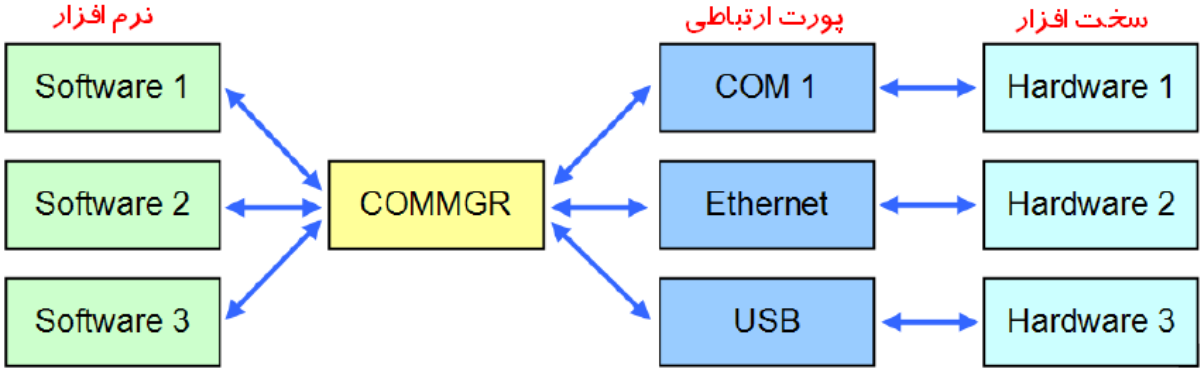
فصل ۳– پیکربندی سیستم

۳–۱– مقدمه

در ISPSoft به راحتی می‌توان تنظیمات مربوط به سخت افزار را برای پیاده‌سازی برنامه نوشته شده انجام داد. قابلیت تنظیم ارتباط نرم افزار و سخت افزار، پیکربندی ساختار سخت افزار و تنظیمات مربوط به شبکه صنعتی[۱] از جمله مزیت‌های نرم افزار ISPSoft می‌باشد.

۳–۲– معرفی COMMGR

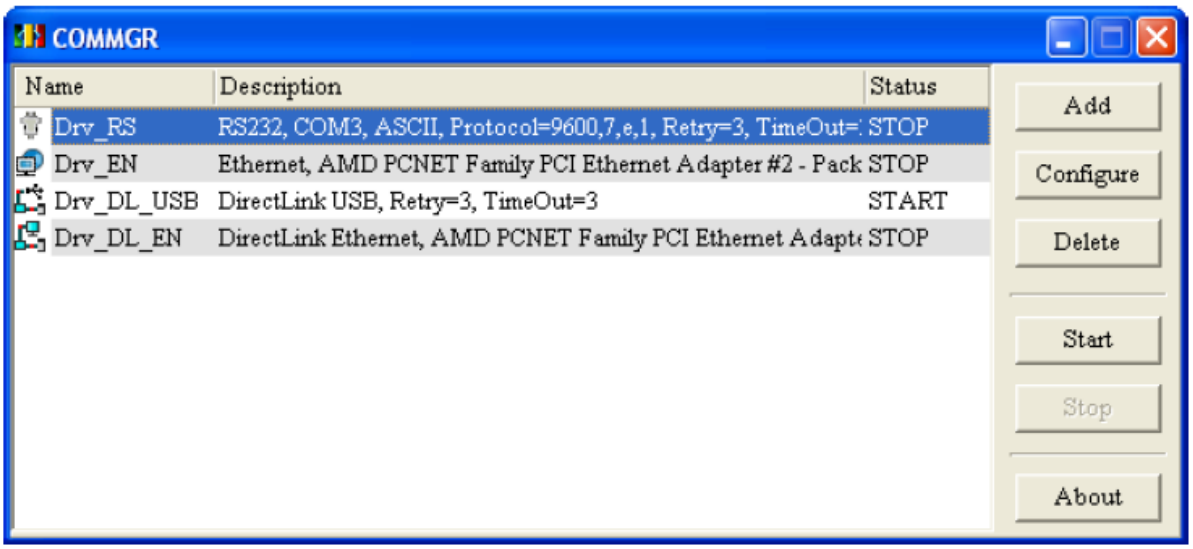
COMMGR ابزار جدید مدیریت شبکه ارتباطی است که برای ارتباط سخت افزار و نرم افزارهای ساخت کمپانی دلتا به کار می‌رود. ارتباط بین این اجزا با استفاده از COMMGR قابل کنترل و راحت خواهد بود. کاربران می‌توانند به سادگی پارامترهای ارتباطی را در این نرم افزار تنظیم نمایند و ارتباط سخت افزارها و نرم افزارهای کمپانی دلتا را به صورت دلخواه در هر لحظه برقرار نمایند.



شکل ۳-۱ بلوک دیاگرام نحوه ارتباط سخت افزار و نرم افزار به وسیله COMMGR

صفحه‌ی COMMGR و لیست ارتباط‌های تحت مدیریت آن در شکل زیر نمایش داده شده‌اند. نام درایورها (که توسط کاربر تنظیم می شود) در قسمت Name، پارامترهای ارتباطی در قسمت Description و وضعیت ارتباط در قسمت Status مشخص شده‌اند.

[۱] تنظیمات مربوط به شبکه صنعتی در فصل ۱۳- مطرح خواهد شد.

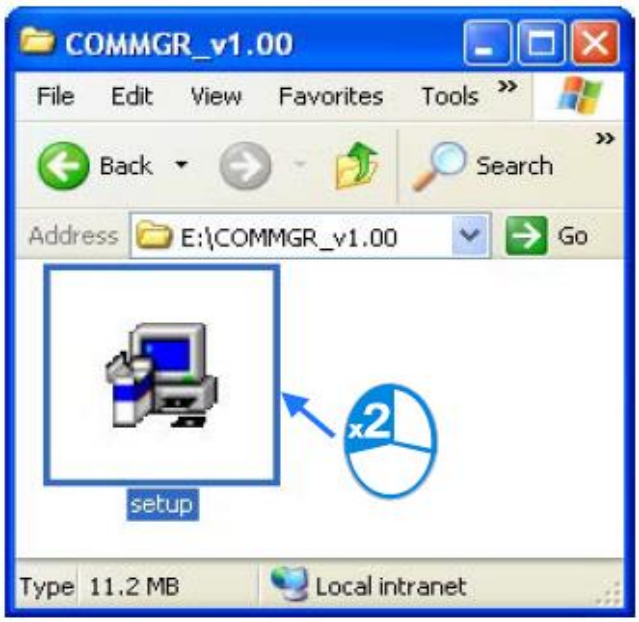


شکل ۳-۲ پنجره COMMGR و درایورهای تعریف شده در آن

۳-۲-۱- نصب نرم افزار COMMGR

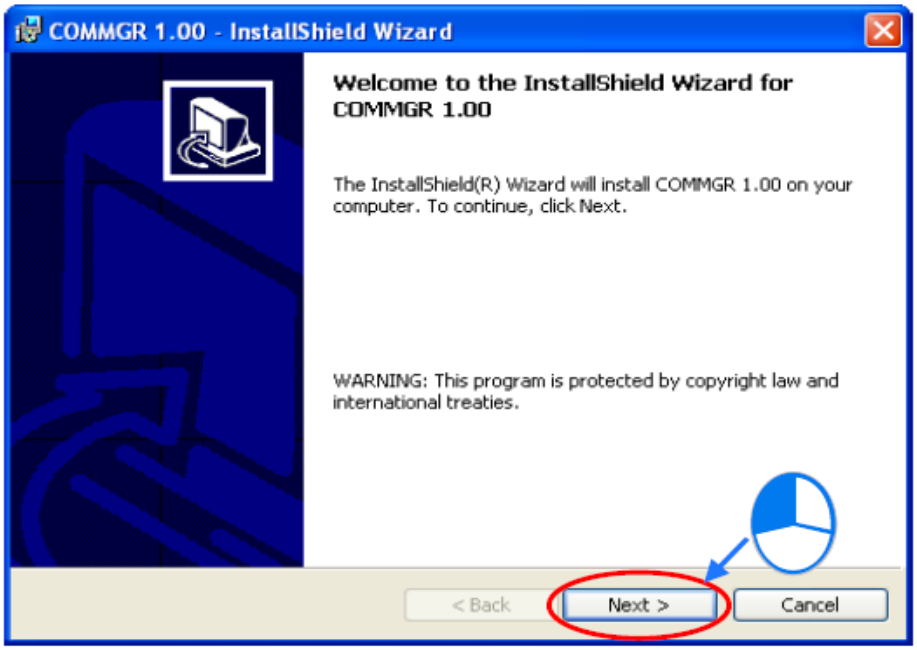
نصب نرم افزار COMMGR روند ساده‌ای دارد. در صورتی که نسخه‌های قدیمی COMMGR بر روی کامپیوتر نصب شده باشد، باید ابتدا آن را حذف و سپس اقدام به نصب نسخه جدید نمود. همچنین شما می‌توانید در هر زمان آخرین نسخه از نرم افزار COMMGR را به صورت رایگان از پایگاه اینترنتی کمپانی دلتا به آدرس http://www.delta.com.tw/ch/index.asp دریافت نمایید. برای هماهنگی بیشتر بهتر است این نرم افزار را پس از نرم افزار ISPSoft نصب نمایید.

۱- برای نصب ISPSoft فایل نصب آن را (با پسوند *.exe) اجرا نمایید.



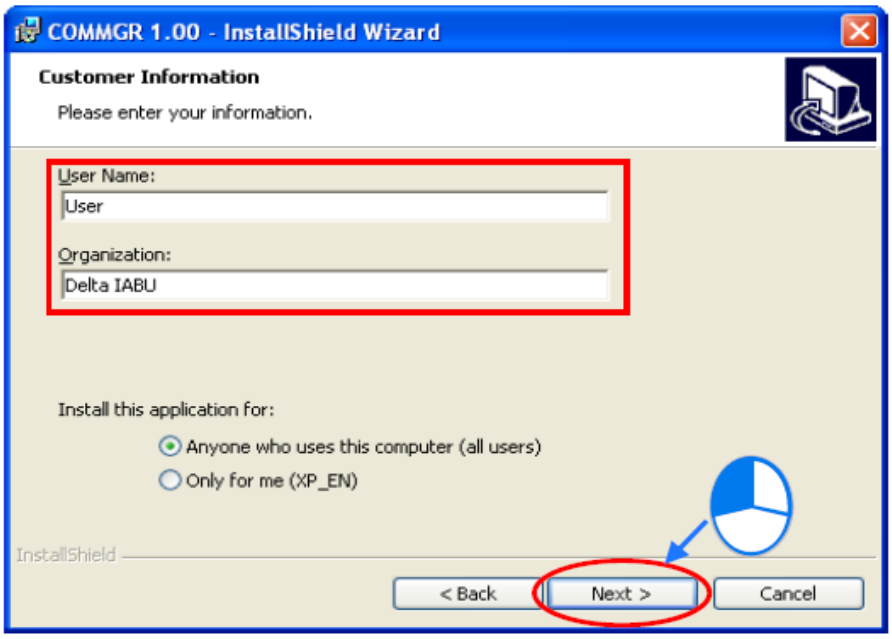
شکل ۳-۳ اجرای فایل Setup برنامه COMMGR

۲- پس از ظاهر شدن صفحه COMMGR (X.xx) – InstallShield Wizard برای ادامه کار باید بر روی گزینه Next کلیک نمایید



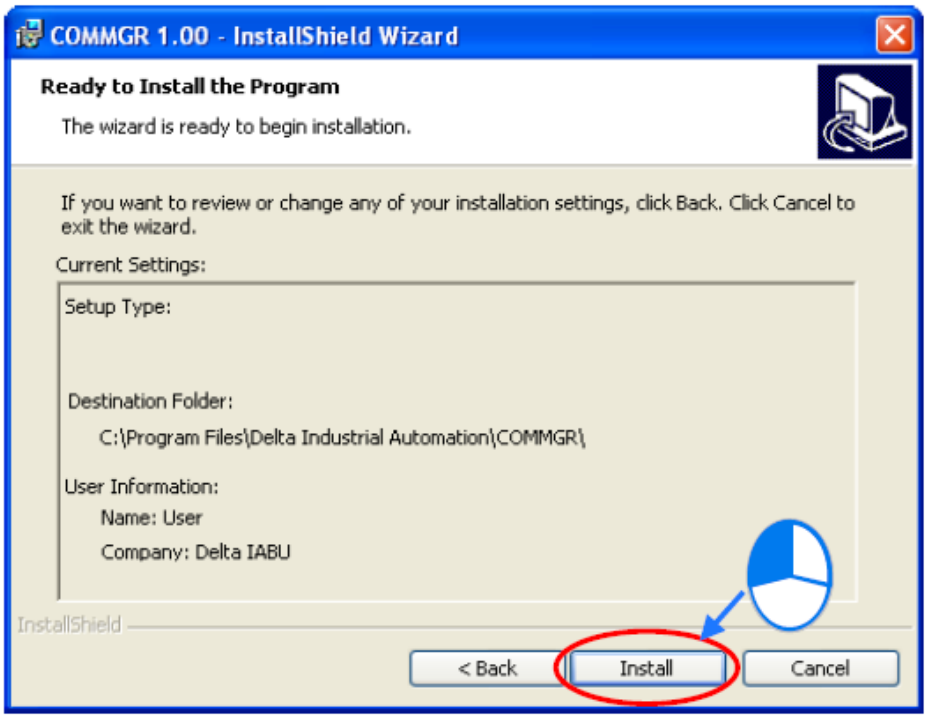
شکل ۳-۴ صفحه‌ی ابتدایی Setup برنامه COMMGR

۳- در صفحه‌ی پیش‌رو پس از وارد کردن نام خود و سازمان مورد نظرتان به ترتیب در دو قسمت User Name و Organization در صورتی که می‌خواهید دسترسی برای تمام کاربران کامپیوتر آزاد باشد در قسمت "Install this application for" می‌توانید گزینه "Anyone who uses this computer" و در غیر این صورت "Only for me" استفاده نمایید.



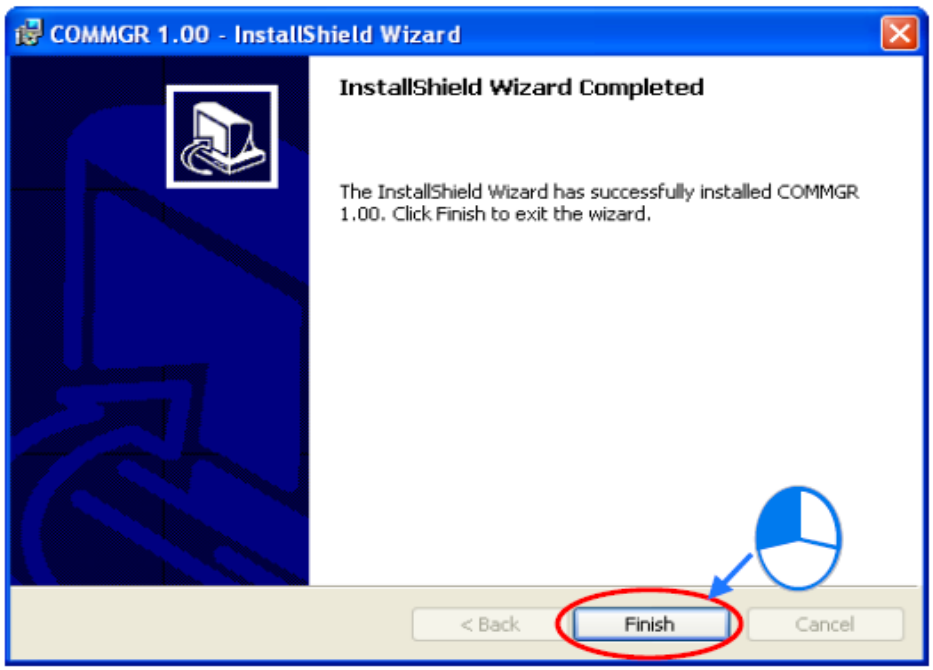
شکل ۳-۵ تنظیم نام کاربر و سازمان در Setup برنامه COMMGR

۴- پس از مرور اطلاعات، می‌توان با کلیک بر روی گزینه Install اقدام به نصب برنامه کرد.



شکل ۳-۶ بررسی نهایی اطلاعات در Setup برنامه COMMGR

۵- پس از طی شدن مراحل نصب نرم افزار، پیغام پایان موفقیت آمیز مراحل نصب ظاهر خواهد شد که برای پایان پروسه نصب می‌توانیم بر روی گزینه Finish کلیک نماییم. در این مرحله میان‌بری از برنامه در دسکتاپ و منوی شروع ویندوز ایجاد خواهد شد.



شکل ۳-۷ پایان مراحل نصب برنامه COMMGR

۳-۲-۲- تنظیم ارتباط

ارتباط بین ISPSoft و PLC-های کمپانی دلتا به صورت بلوک دیاگرام زیر از طریق نرم افزار COMMGR برقرار می‌شود[۱].

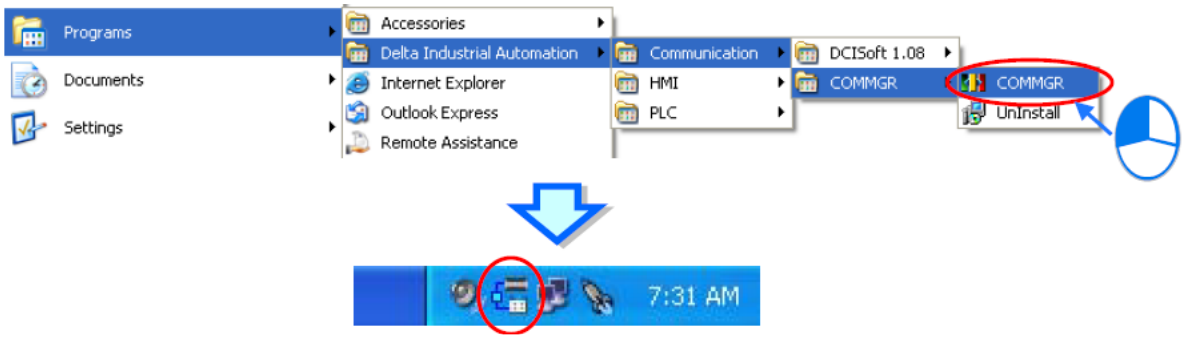
شکل ۳-۸ ارتباط ISPSoft با PLC با کمک برنامه COMMGR

پس از نصب نرم افزار COMMGR آیکون آن () در نوار Taskbar ویندوز ظاهر می‌شود: 

. همچنین پس از هربار شروع مجدد ویندوز، این نرم افزار در ابتدای بالا آمدن ویندوز خود به خود فعال می‌شود. در صورتی هم که به هر دلیلی این نرم افزار غیرفعال باشد و

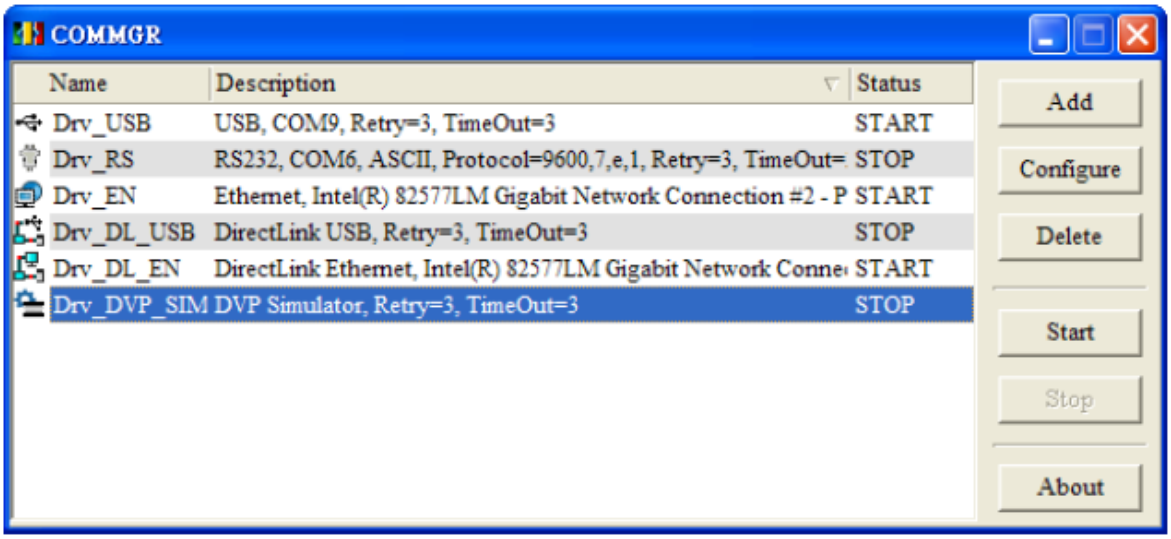
[۱] توجه شود که COMMGR تنها برای نرم افزار ISPSoft نسخه ۲ به بالا کاربرد دارد.

بخواهیم آن را فعال کنیم، در منوی Start، در قسمت نرم افزارها[1] طبق مسیر زیر می‌توانیم COMMGR را فعال کنیم. پس از اجرای COMMGR شما می‌توانید با دوبار کلیک کردن بر روی آیکون آن پنجره COMMGR را فعال نمایید.



شکل ۳-۹ ارتباط ISPSoft با PLC با کمک برنامه COMMGR

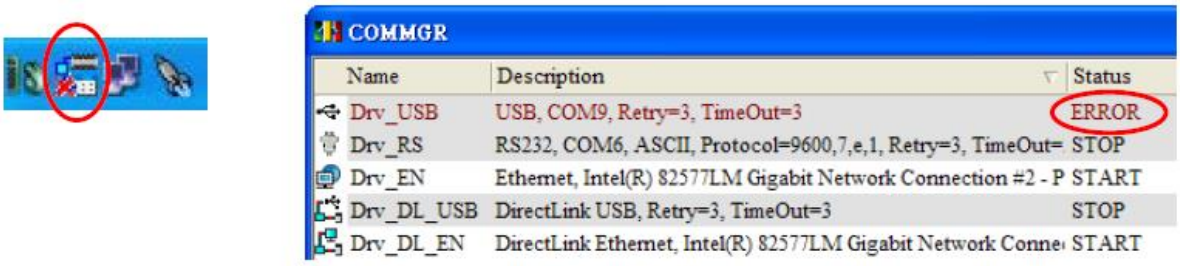
پنجره COMMGR مطابق شکل زیر باز خواهد شد که با گزینه‌های سمت راست می‌توان درایورهای آن را مدیریت کرد. درایورهای لیست شده در COMMGR ارتباط نرم افزار و پورت های ارتباطی کامپیوتر را برقرار می‌کند. (درایور در واقع یک سری دستورالعمل است که کامپیوتر از انها پیروی می کند تا اطلاعات را برای انتقال به دستگاه جانبی خاص یا بازیابی از ان دوباره قالب بندی کند و بدین وسیله امکان ارتباط نرم افزار و سخت افزار را برقرار می‌کند)



شکل ۳-۱۰ پنجره برنامه COMMGR به همراه درایورهای آن

[1] Programs

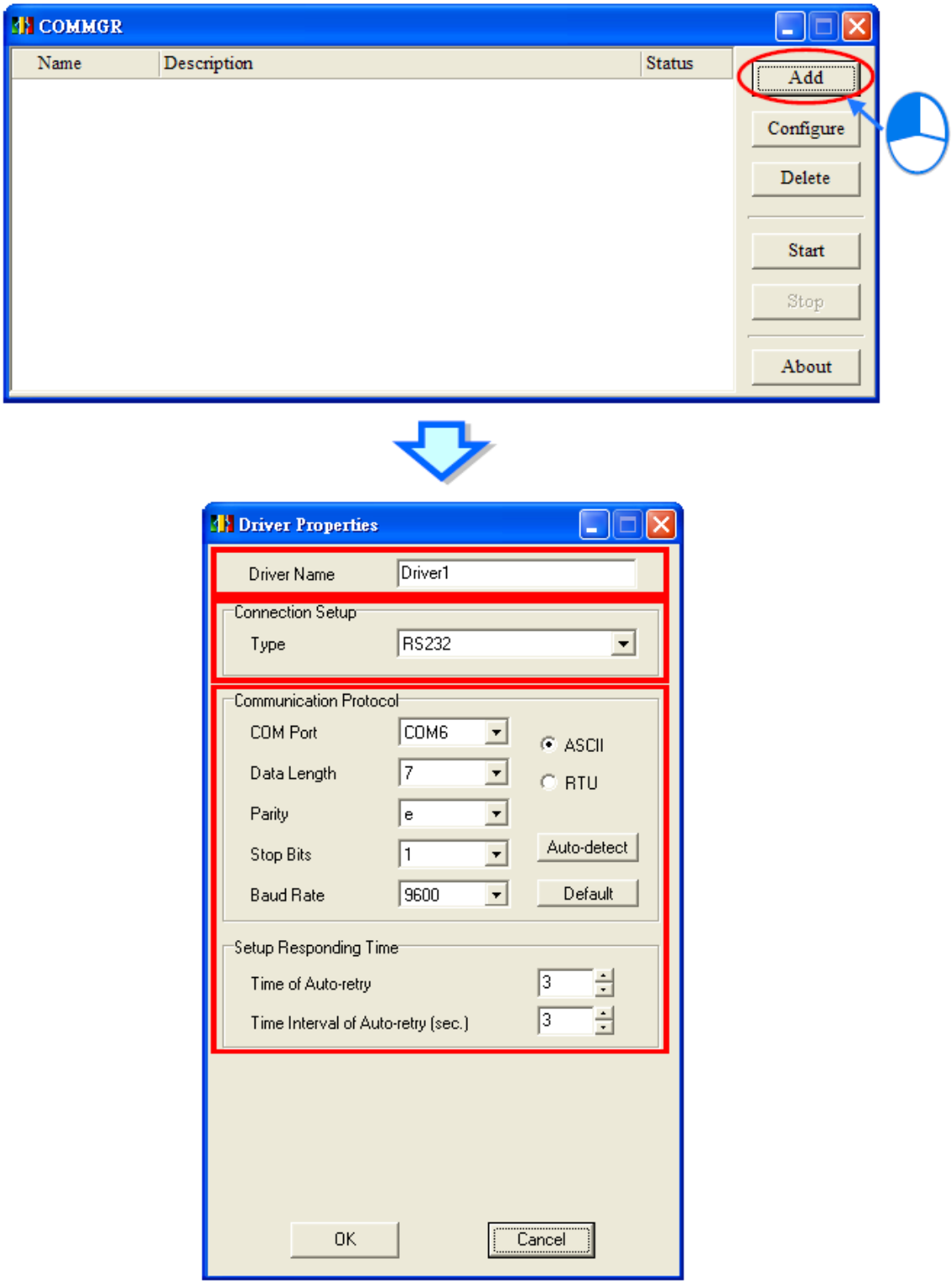
در صورتی که ارتباط نرم افزار با پورت‌های مشخص شده به وسیله COMMGR برقرار باشد، در ستون Status وضعیت ارتباط به صورت START مشخص خواهد شد، با این حال اگر COMMGR نتواند به هر دلیلی با پورت مورد نظر ارتباط برقرار کند (این حالت ممکن است به علت اشغال بودن پورت در اثر استفاده نرم افزاری دیگر باشد)، درایور متوقف خواهد شد و وضعیت ERROR به نمایش خواهد آمد.



شکل ۳-۱۱ ایجاد خطا در دسترسی به پورت USB در برنامه COMMGR

۳-۲-۳- ساخت درایور

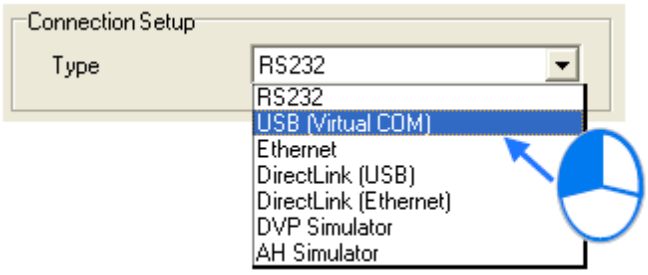
برای اینکار می بایست ابتدا بر روی Add در COMMGR کلیک کنیم تا صفحه Driver Properties باز شود.



شکل ۳-۱۲ ایجاد پروژه جدید در برنامه COMMGR

در این قسمت می‌توانیم اسم درایور را به صورت دلخواه در قسمت Driver Name تعیین نماییم. توجه شود که از این نام در آینده برای ارتباط نرم افزارهای کمپانی دلتا با پورت مورد نظر باید استفاده شود به همین سبب پیشنهاد می‌شود مکانیزم مشخصی برای انتخاب نام درایور شامل در نظر گرفتن نام پورت و

ویژگی‌های آن لحاظ تا از سردرگمی جلوگیری شود. نوع پورت (پروتکل) ارتباطی مورد نظر خود را نیز می‌توانیم در قسمت Connection Setup تنظیم نماییم. نوع ارتباط مورد نظر ما می تواند ارتباط سریال RS232، USB، پورت Ethernet، USB و Ethernet متصل به HMI و یا ارتباط با شبیه سازها باشد.

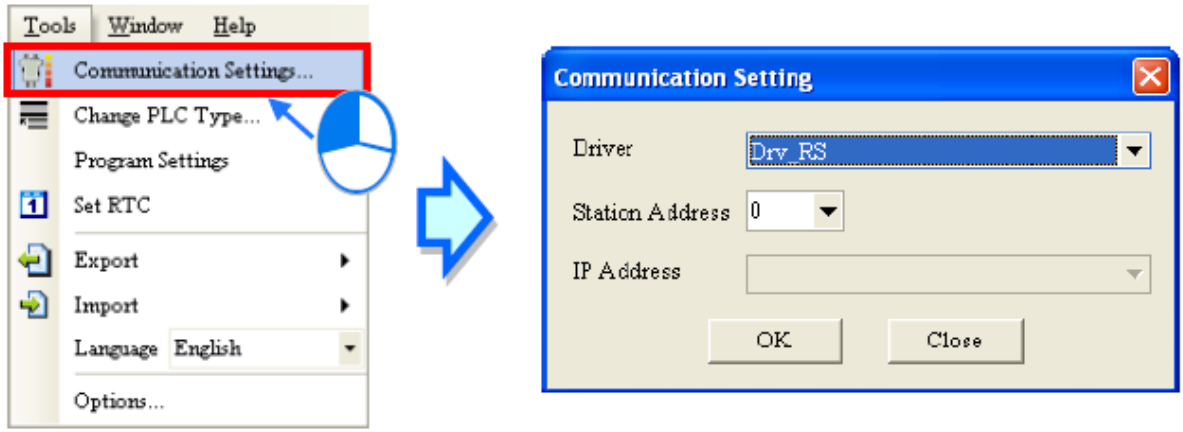


شکل ۳-۱۳ انواع پروتکل ارتباطی در COMMGR

ستون های بعدی COMMGR تنظیمات پروتکل ارتباطی می‌باشد که باید آن‌ها را مطابق با سخت افزار و یا شبکه ارتباطی تنظیم نمایید. به عنوان مثال در ارتباط سریال باید نرخ ارسال داده و یا نوع Parity و طول داده ها را مشخص نمایید. در قسمت Setup Responding Time نیز کاربر می‌تواند حداکثر تعداد تلاش برای ارتباط و حداکثر مدت زمان انتظار برای اینکار را به ترتیب در دو قسمت Time of Auto-retry و Time Interval of Auto-retry مشخص نماید. پس از تایید و ساخت درایور می‌توان از طریق گزینه‌های سمت راست COMMGR آن ها را فعال و یا غیرفعال کرد و یا اینکه توسط گزینه Configure آن‌ها را دوباره تنظیم کرد.

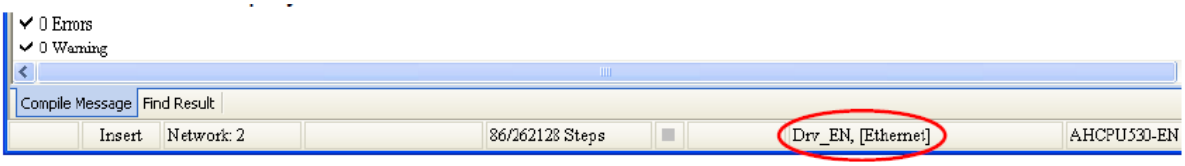
۳-۲-۴- تنظیم ارتباط بین نرم افزار ISPSoft و COMMGR

پس از تنظیم داریورها در COMMGR کاربر می‌تواند در ISPSoft نیز درایور را در هر پروژه برای ارتباط با PLC تعیین نماید. برای اینکار اگر پروژه گروهی است ابتدا باید پروژه داخلی مورد نظر خود را فعال نمایید. سپس در سربرگ Tools گزینه Communication Settings را انتخاب نمایید.



شکل ۳-۱۴ تنظیم ارتباط ISPSoft با PLC با کمک برنامه COMMGR

در پنجره جدید در قسمت Driver، باید نوع درایور را مشخص کرد، همچنین Station Address متناظر با PLC که با PC در ارتباط است باید تعیین شود. اگر کاربر Station Address را نمی داند می-تواند به جای آن صفر را انتخاب نماید (زمانی که قرار است در مراحل بعد چند PLC به صورت همزمان از طریق یک پورت با PC ارتباط برقرار کنند، لازم است در این مرحله به هر PLC یک Station Address منحصر به فرد اختصاص داده شود). اگر نوع ارتباط درایور به صورت Ethernet باشد، آنگاه کاربر باید IP ثبت شده توسط COMMGR[1] را انتخاب نماید. پس از اتمام تنظیمات، اطلاعات در مورد درایور متصل شده در نوار وضعیت نمایش داده می شود.



شکل ۳-۱۵ اطلاعات درایور COMMGR در نوار وضعیت ISPSoft

۳-۲-۵- ارتباط با PLC از طریق پورت های ارتباطی

ارتباط PLC با پورت های ارتباطی به روش‌های گوناگونی ممکن است. در ادامه بعضی نکات در رابطه با ارتباط PLCها از طریق اتصال پورت‌های ارتباطی را مرور خواهیم کرد. ذکر این نکته ضروری است که مراجعه به راهنمای PLCها برای آگاهی از اطلاعات بیشتر پیرامون نحوه سیم‌بندی پورت‌های ارتباطی و جلوگیری از صدمات به سخت افزار ضروری است.

توجه به این نکته بسیار ضروری است که پورت و پروتکل ارتباطی تعیین شده برای درایور با پورت و پروتکل تعبیه شده بر روی PLC باید یکسان باشد.

- PLCهای خانواده DVP (اتصال از طریق RS232)
- PLCهای خانواده DVP-SX2 (اتصال از طریق USB)

  از آنجایی که DVP-SX2 پروتکل USBرا به RS232 تبدیل می‌کند. نوع ارتباط مورد نظر برای درایور باید RS232 باشد.
- PLCهای خانواده AH500 (اتصال از طریق USB)

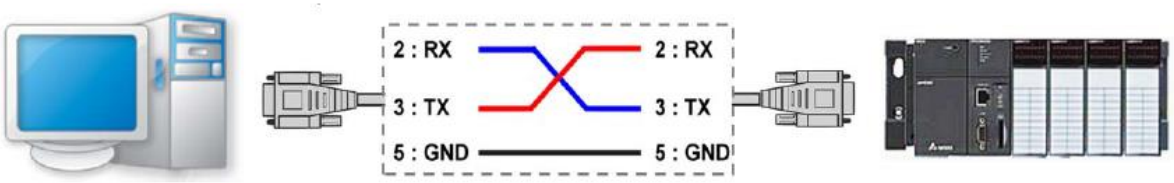
[1] IP همه ماژول‌هایی که می‌خواهیم با آن ها ارتباط برقرار کنیم باید ابتدا در COMMGR ثبت شود. البته لازم نیست این کار را به صورت دستی انجام دهیم و خود COMMGR می‌تواند به صورت اتوماتیک شبکه را برای پیدا کردن دستگاه‌ها جستجو کند.

در این حالت نوع ارتباط برای درایور باید USB باشد (Virtual COM)

در AH500 اتصال از طریق USB مناسب‌ترین و سریعترین راه ممکن برای اتصال PC و PLC می‌باشد.

- PLCهای خانواده AH500 (اتصال از طریق RS232)

در این حالت درایور باید به صورت RS232 باشد و اتصال سیم های RXو TX در کامپیوتر و PLC به صورت زیر خواهد بود (چون AH500 از RS485 و RS422 نیز پشتیبانی می‌کند، باید دقت کرد که برای ارتباط سریال با کامپیوتر صرفا از RS232 استفاده شود)

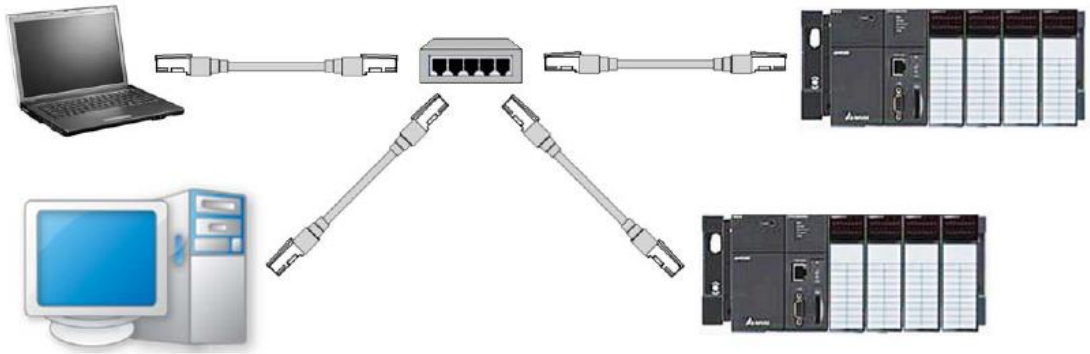


شکل ۳-۱۶ سیم بندی ارتباط سریال کامپیوتر با PLC

- PLCهای خانواده AH500 (اتصال از طریق Ethernet)

مدل‌های AHCPU5xx-EN مجهز به پورت Ethernet هستند و می‌توان PLC را با PC از طریقhub یا به صورت مستقیم به وسیله پورت شبکه به یکدیگر متصل کرد.

و یا استفاده از Hub



شکل ۳-۱۷ ارتباط مستقیم و یا با استفاده از Hub برای ارتباط از طریق Ethernet

مقادیر پیشفرض پارامترها برای ارتباط سریال و Ethernet در خانواده AH500 به صورت زیر است.

جدول ۳-۱: مشخصات پیشفرض ارتباط سریال در خانواده AH500

Serial Port	COM1	COM2
Connection Type	RS232	RS232
Transmition Mode	ASCII	ASCII
Communication Speed (Baud Rate)	9600	9600
Data Length	7	7
Parity Bit	Even	Even
Stop Bit	1	1
Station Address	1	3

جدول ۳-۲: مشخصات پیشفرض ارتباط از طریق Ethernet در خانواده AH500

Ethernet	مقدار پارامترهای
IP Address Assignment	Static
IP Address	192.168.1.1
Subnet mask	255.255.255.0
Gateway Address	192.168.1.1

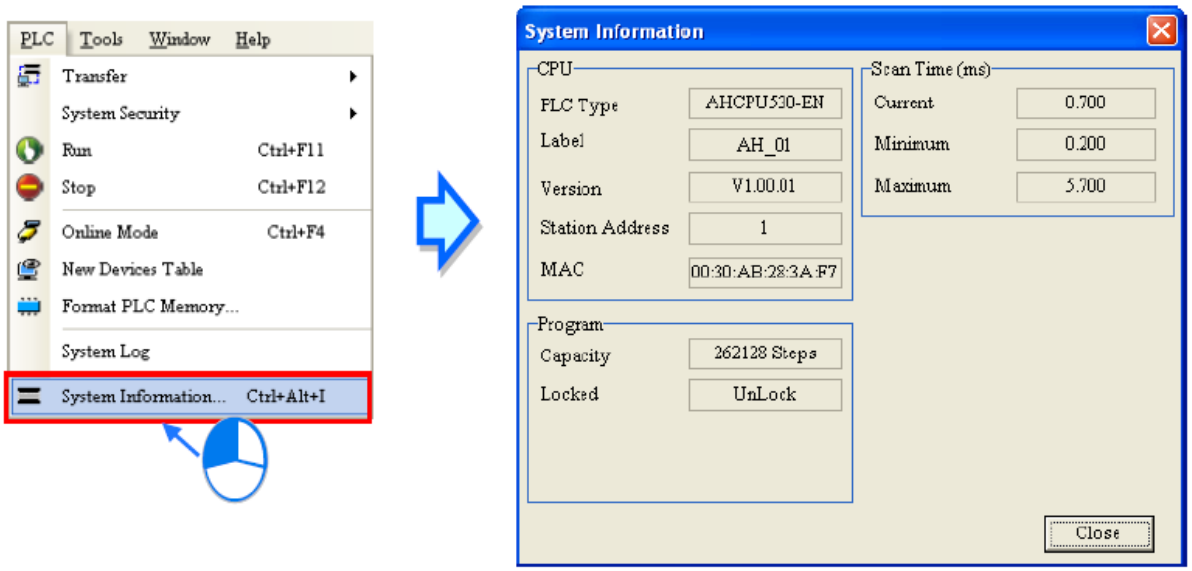
۳-۲-۶- تست عملی اتصال

پس از طی شدن مراحل فوق کاربران برای اطمینان از اتصال کامپیوتر به PLC می‌توانند از تستی ساده استفاده کنند. در ابتدا موارد زیر را بررسی کنید:

- نوع پورت PLC با نوع درایور تنظیم شده یکسان باشد.

- وضعیت درایور در حالت Start باشد.
- وضعیت کانال ارتباطی شامل کارت شبکه کامپیوتر، Hub و پورت سریال عادی باشد.
- درایور، آدرس Station و IP در Communication Setting درست تنظیم شده باشد.
- PLC به صورت صحیحی به طریق کابل ارتباطی متصل باشد، تغذیه PLC متصل و وضعیت آن عادی باشد.

حال می توان در سربرگ PLC گزینه System Information را انتخاب کرد، اگر ارتباط PLC با کامپیوتر به صورت نرمال برقرار شود ، آنگاه صفحه System Information ظاهر شده و اطلاعات رسیده از PLC نمایش داده می‌شود. برای اینکار



شکل ۳-۱۸ برقراری ارتباط بین ISPSoft با PLC

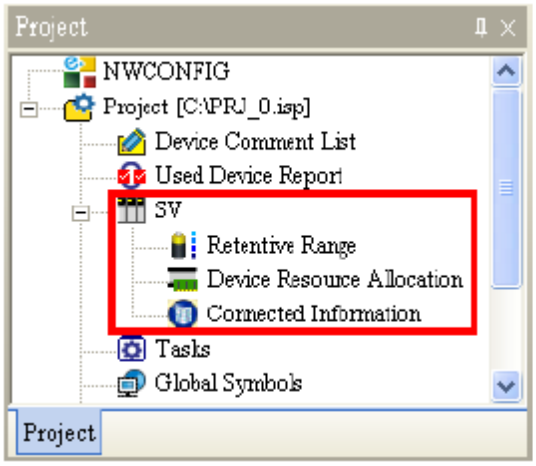
۳-۳- پیکربندی PLC در ISPSoft

در این بخش کاربران با نحوه پیکربندی سخت افزار PLC در نرم افزار ISPSoft آشنا می‌شوند. بخش مدیریت پروژه مخصوصا برای تنظیمات سخت افزاری برای دو خانواده‌ی DVP و AH500 کاملا متفاوت است، این فضا برای DVP تا حدودی مشابه نرم افزار WPLSoft کمپانی دلتا و برای AH500 مشابه STEP7

شرکت زیمنس[1] می‌باشد. گزینه‌ها و امکاناتی که برای مدیریت هرکدام از این دو خانواده PLC وجود دارد متفاوت است. به همین سبب در دو بخش مجزا به معرفی هر کدام از آن‌ها خواهیم پرداخت.

۳-۳-۱- پیکربندی مدل های DVP

زیر شاخه‌های SV در بخش مدیریت پروژه (که بعضی از آن‌ها فقط در حالت اتصال PLC فعال می‌شوند) برای مدیریت سخت افزار PLC به کار می‌روند. عملکرد این بخش‌ها در ادامه شرح داده می‌شوند.



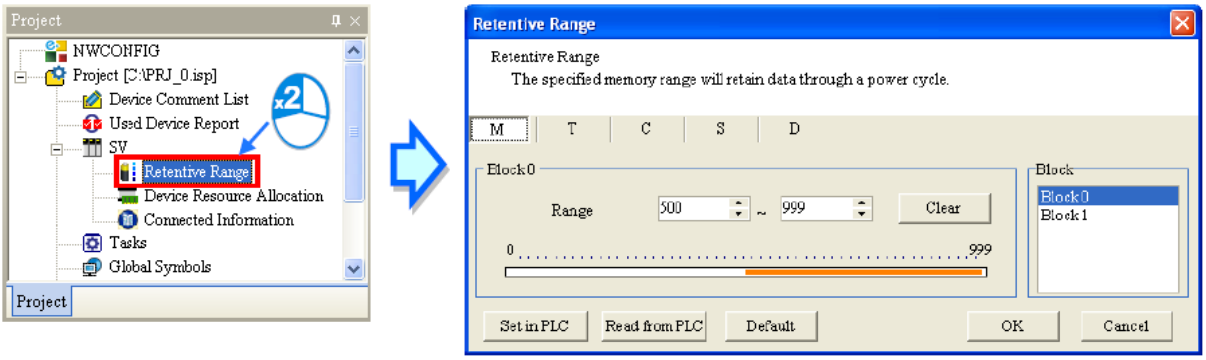
شکل ۳-۱۹ مدیریت پروژه خانواده DVP در نرم افزار ISPSoft

- Retentive Range: تنظیم بازه حافظه‌های نگهدارنده PLC
- Device Resource Allocation: تنظیم رنج حافظه‌هایی که سیستم خود به خود به Symbolها اختصاص می‌دهد.
- Connected Information: مشاهده تنظیمات فعلی و پارامترهای PLC

۳-۳-۱-۱- Retentive Range

پس از دوبار کلیک بر روی Retentive Range، پنجره آن باز می‌شود.

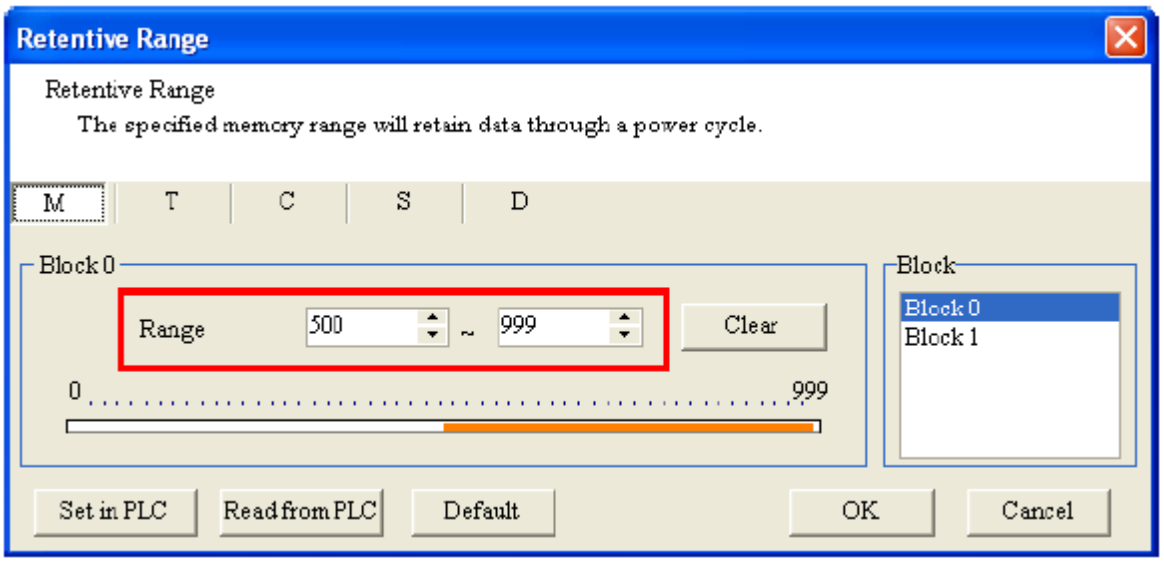
[1] Siemens



شکل ۳-۲۰ Retentive Range برای خانواده DVP

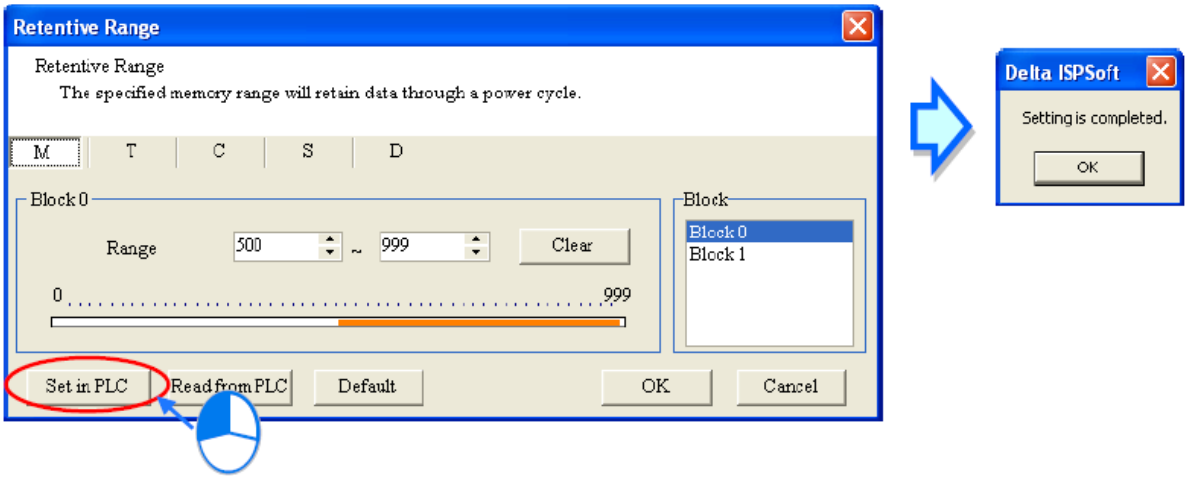
در این پنجره می‌توان محدوده حافظه‌های نگهدارنده داده را تعیین کرد چرا که در PLC ممکن است نوع حافظه،مانا نباشد (مقدار ذخیره شده در آن باقی نماند) و بدین منظور باید حافظه‌هایی که نیاز است مقدار آن حفظ شوند در این بخش مشخص می‌شوند . در سربرگ‌های M، T، C، S و D به ترتیب می‌توان حافظه‌های نگهدارنده رجیستر بیتی، زمان سنج، شمارنده، پله و داده را تعیین کرد، در هر سربرگ نیز می‌توان بلوک مورد نظر را انتخاب کرد.

در نمودار میله‌ای (شاخه‌ای افقی) مشخص شده در این قسمت می‌توانیم محدوده حافظه نگهدارنده‌ای که می‌توانیم تعیین کنیم را مشخص نماییم. بخش رنگی حافظه نگدارنده تعیین شده را مشخص می‌کند. محدوده حافظه نگهدارنده را می‌توانیم در قسمت Range مشخص نماییم (این مقادیر باید در محدوده مشخص شده در نمودار میله‌ای باشد). با کلیک بر روی Clear می‌توان تعیین کرد که هیچ کدام از این حافظه‌ها نگهدارنده نمی باشند (در این حالت مقادیر در قسمت Range به صورت ۱- تا ۱- مشخص خواهند شد)



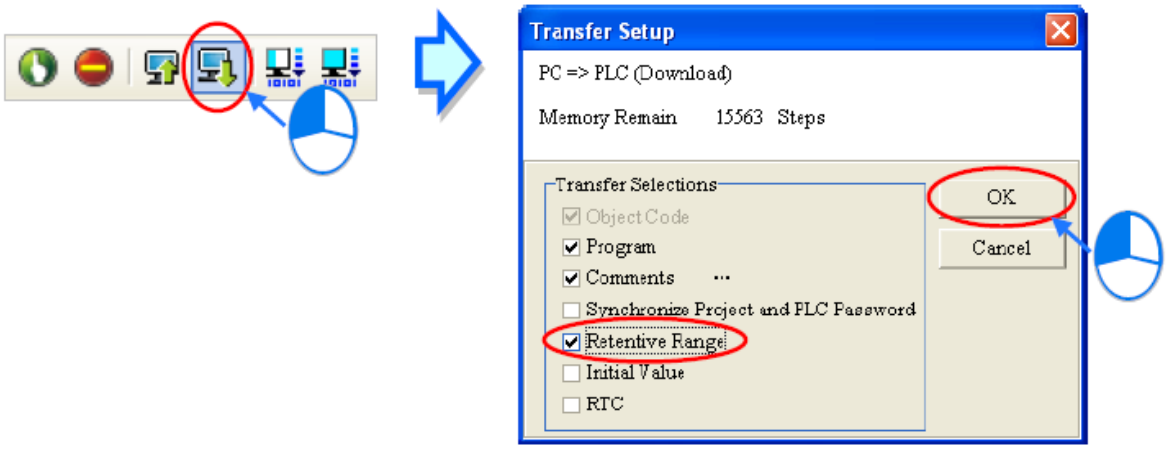
شکل ۳-۲۱ تعیین حافظه نگهدار در Retentive Range

همچنین با کلیک بر روی Default مقادیر حافظه نگهدارنده به حالت پیش فرض بر می گردد و با کلیک بر روی Read from PLC اگر نرم افزار به PLC متصل باشد، این مقادیر مطابق وضعیت فعلی PLC در خواهند آمد. پس از تکمیل تنظیمات در این بخش می‌توان با کلیک بر روی Set in PLC (در صورتی که ISPSoft به PLC متصل باشد) این تنظیمات را در PLC بارگزاری کرد. در این حالت مقادیر پنجره فوق بسته می‌شوند.



شکل ۳-۲۲ بارگزاری تنظیمات Retentive Range در PLC

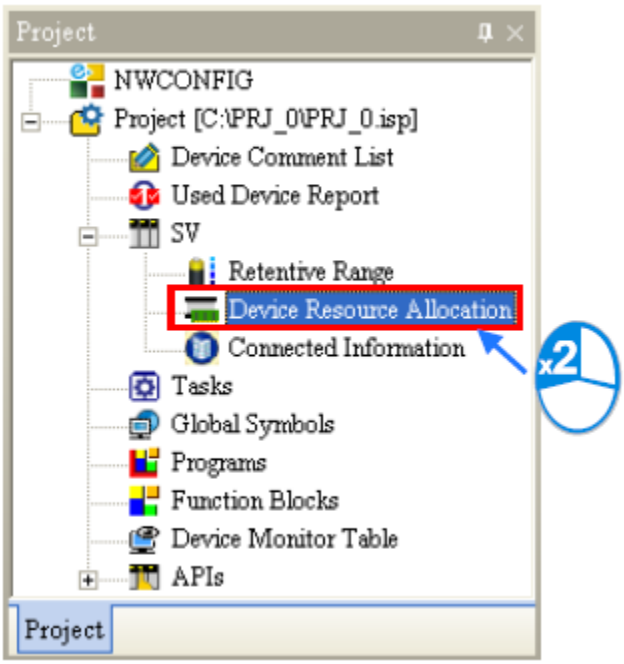
اگر نیازی به بارگزاری مقادیر بر روی PLC ندارید و تنها می‌خواهید آن را ذخیره نمایید نیز کافی است بر روی OK کلیک نمایید. سپس در زمان مورد نیاز برای بارگزاری مقادیر حافظه های نگهدارنده، می‌توان از آیکون  استفاده کرد، در پنجره باز شده با انتخاب Retentive Range مقادیر حافظه نگهدارنده مشخص شده در PLC بارگزاری خواهد شد.



شکل ۳-۲۳ بارگزاری تنظیمات Retentive Range در PLC بعد از ذخیره سازی

۳-۳-۱-۲- Device Resource Allocation

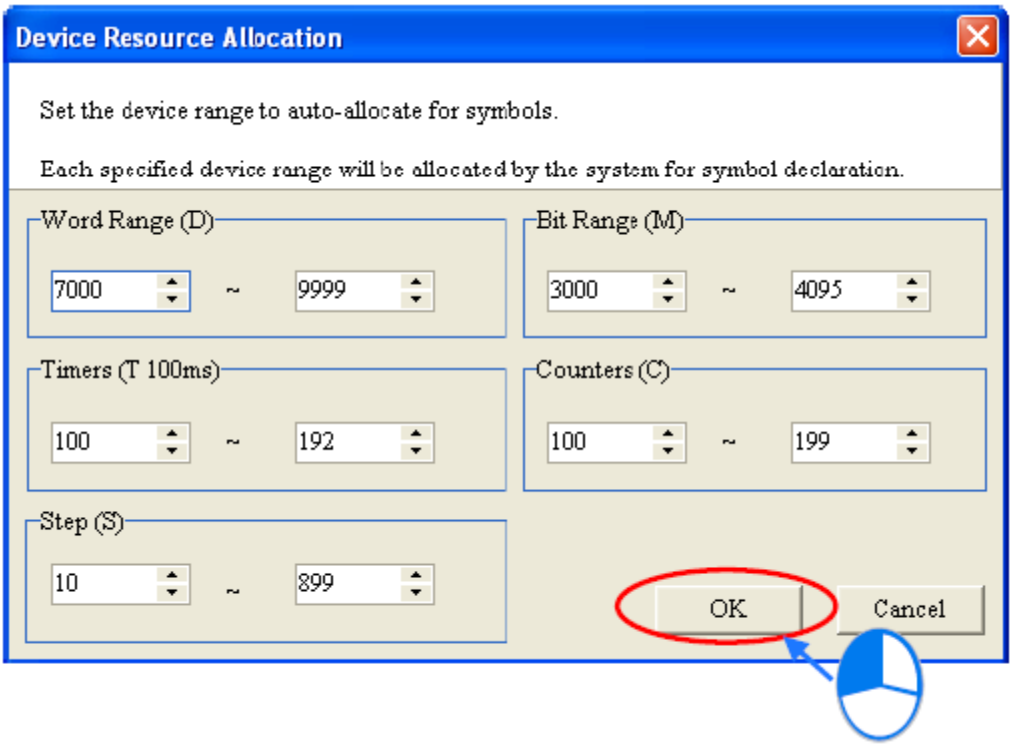
کاربران می‌توانند محدوده حافظه‌هایی که در سری DVP به صورت اتوماتیک به سیمبول‌ها[۱] اختصاص داده می‌شود را تعیین کنند. برنامه پس از کامپایل به صورت اتوماتیک به سیمبول‌هایی که به آن‌ها آدرس اختصاص داده نشده از محدوده تعیین شده حافظه اختصاص می‌دهد. اگر تعداد حافظه‌های مورد نیاز سیمبول‌ها از محدوده تعیین شده بیشتر باشد، در حین کامپایل با پیغام حافظه ناکافی مواجه خواهیم شد. در بخش مدیریت پروژه، زیر شاخه‌های بخش PLC را باز کرده و بر روی Device Resource Allocation دوبار کلیک کنید.



شکل ۳-۲۴ انتخاب Device Resource Allocation

کاربر می‌تواند محدوده حافظه‌های گوناگون را در پنجره ظاهر شده تعیین نماید. توجه شود هر نوع PLC سری DVP محدودیت‌های خاص خودش را دارد، مثلا در DVP-SV رجیسترهای داده از D2000 شروع می‌شوند، حافظه‌های بیتی از M2000 شروع می‌شوند، شمارنده‌ها ۱۶ بیتی هستند و واحد پایه زمان سنج‌ها ۱۰۰ میلی ثانیه است. در صورتی که کاربر محدوده‌ها را درست تنظیم نکرده باشد، سیستم خود آن را تصحیح خواهد کرد. پس کلیک بر روی OK تنظیمات ذخیره خواهد شد.

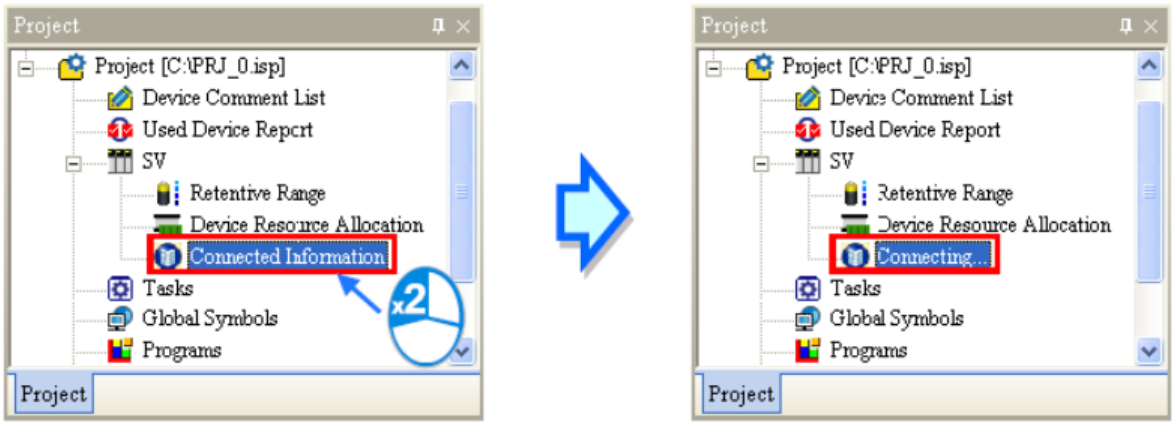
[۱] Symbols



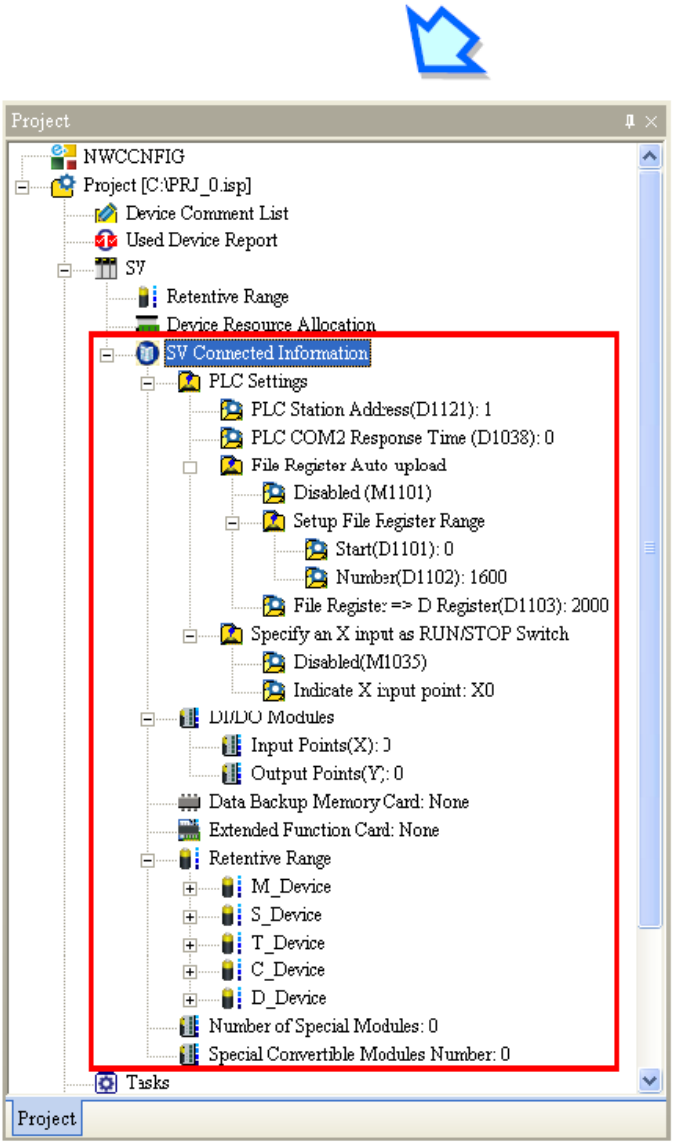
شکل ۳-۲۵ تنظیمات Device Resource Allocation

۳-۳-۱-۳- Connected Information

در صورتی که نرم افزار به PLC متصل باشد با دوبار کلیک کردن بر روی این گزینه می‌توان پارامترهای PLC را مشاهده کرد (بدیهی است که در حالت ارتباط مستقیم امکان تغییر پارامترها وجود ندارد)



شکل ۳-۲۶ نمایش اطلاعات PLC با کلیک بر روی Connected Information – قسمت اول



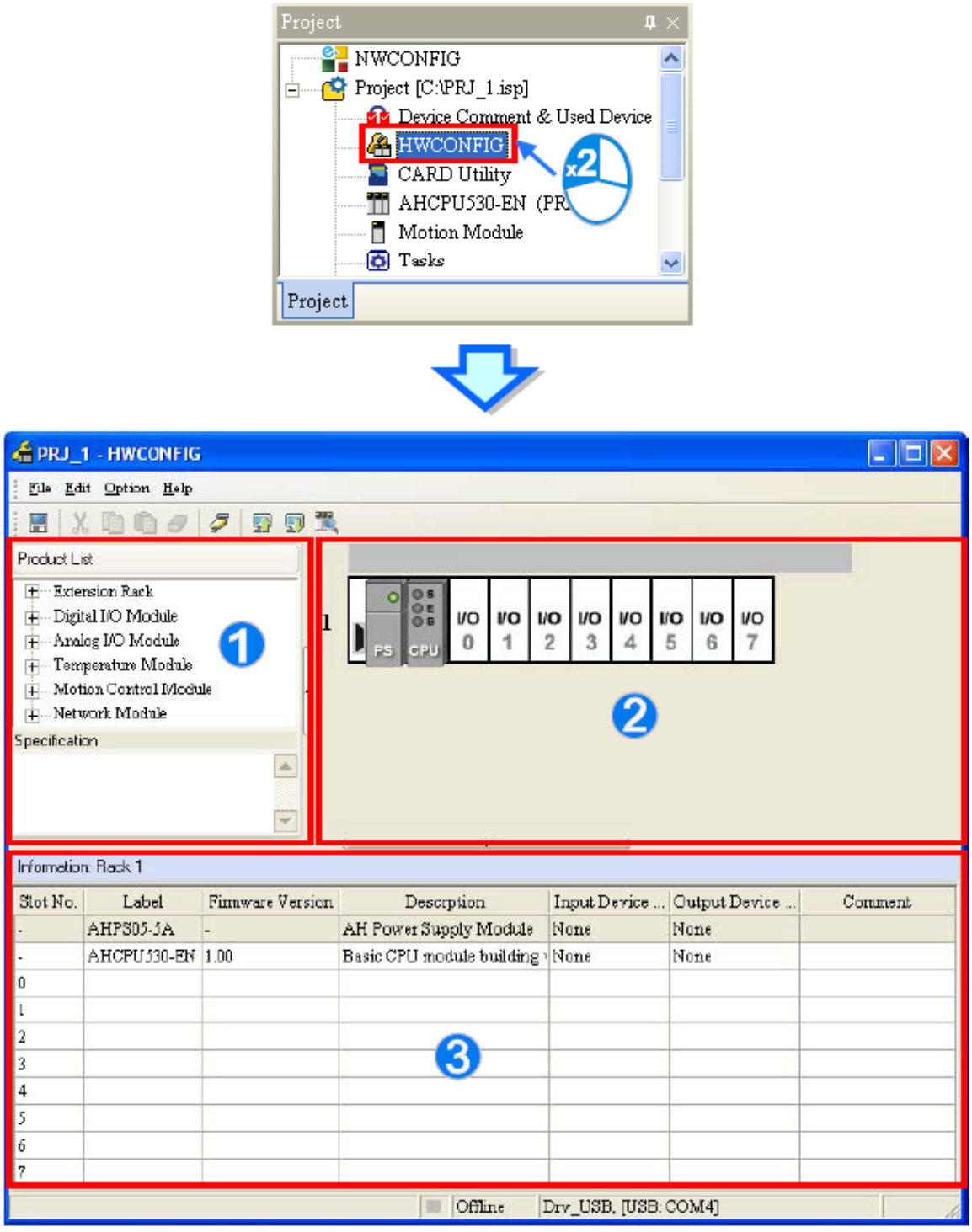
شکل ۳-۲۷ نمایش اطلاعات PLC با کلیک بر روی Connected Information – قسمت دوم

۳-۳-۲- پیکربندی مدل های AH-500

گزینه HWCONFIG در بخش مدیریت پروژه‌ها برای تنظیمات سخت افزاری مدل‌های AH-500 به کار می‌روند. نحوه عملکرد این بخش بسیار شبیه Hardware Configuration نرم افزار STEP7 شرکت زیمنس می‌باشد. در این بخش می‌توان تنظیمات مربوط به Rackها، تنظیم پارامترهای CPU و ماژول‌ها، بارگزاری و استخراج پارامترها، تشخیص آنلاین تنظیمات سیستم و عیب یابی سیستم را انجام داد. توجه نمایید که تمامی تغییرات نرم افزاری و سخت افزاری در نرم افزار ISPSoft باید در PLC بارگزاری شوند، پس در این بخش توجه نمایید که پس از تغییرات سخت افزاری، آن را در سیستم دانلود نمایید.

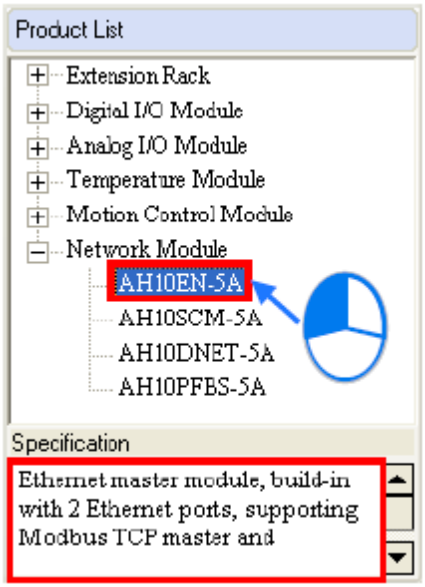
۳-۳-۲-۱- معرفی محیط HWCONFIG

پس از دوبار کلیک کردن بر روی HWCONFIG در بخش مدیریت پروژه، صفحه ای مطابق شکل زیر باز خواهد شد.



شکل ۳-۲۸ صفحه HWCONFIG برای تنظیمات شخت افزاری خانواده

1 Product List: کتابخانه ای است از سخت افزارهایی که می‌توانیم به همراه PLC خود از آن‌ها استفاده کنیم. در این بخش توضیحات مربوط به هر ماژول در قسمت Specification لیست می‌شود.

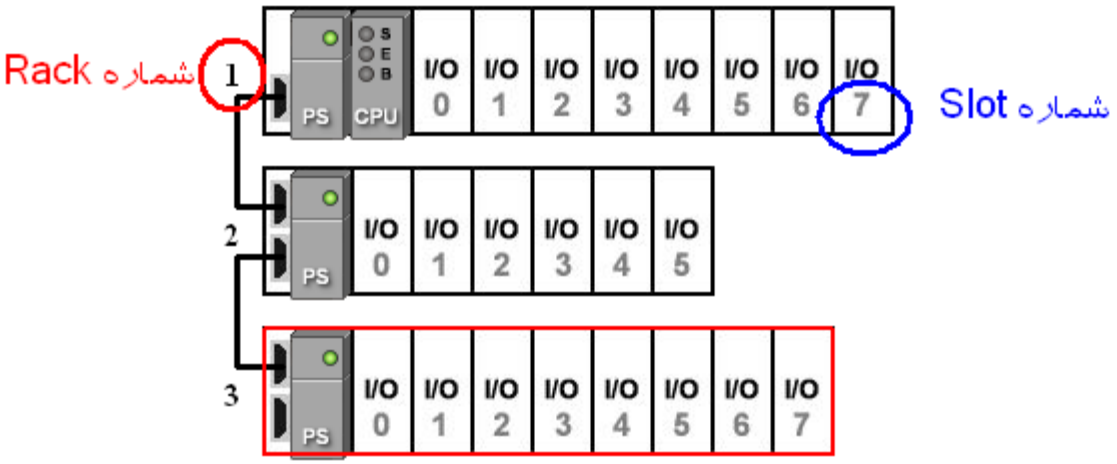


شکل ۳-۲۹ کتابخانه ماژول های همراه AH500

2 System Configuration area: در این قسمت به صورت مصور سیستم پیکربندی شده را می‌توان مشاهده و در صورت نیاز پارامترهای آن را تغییر دهیم.

3 Information List: در این بخش اطلاعاتی پیرامون سیستم تنظیم شده لیست شده است.

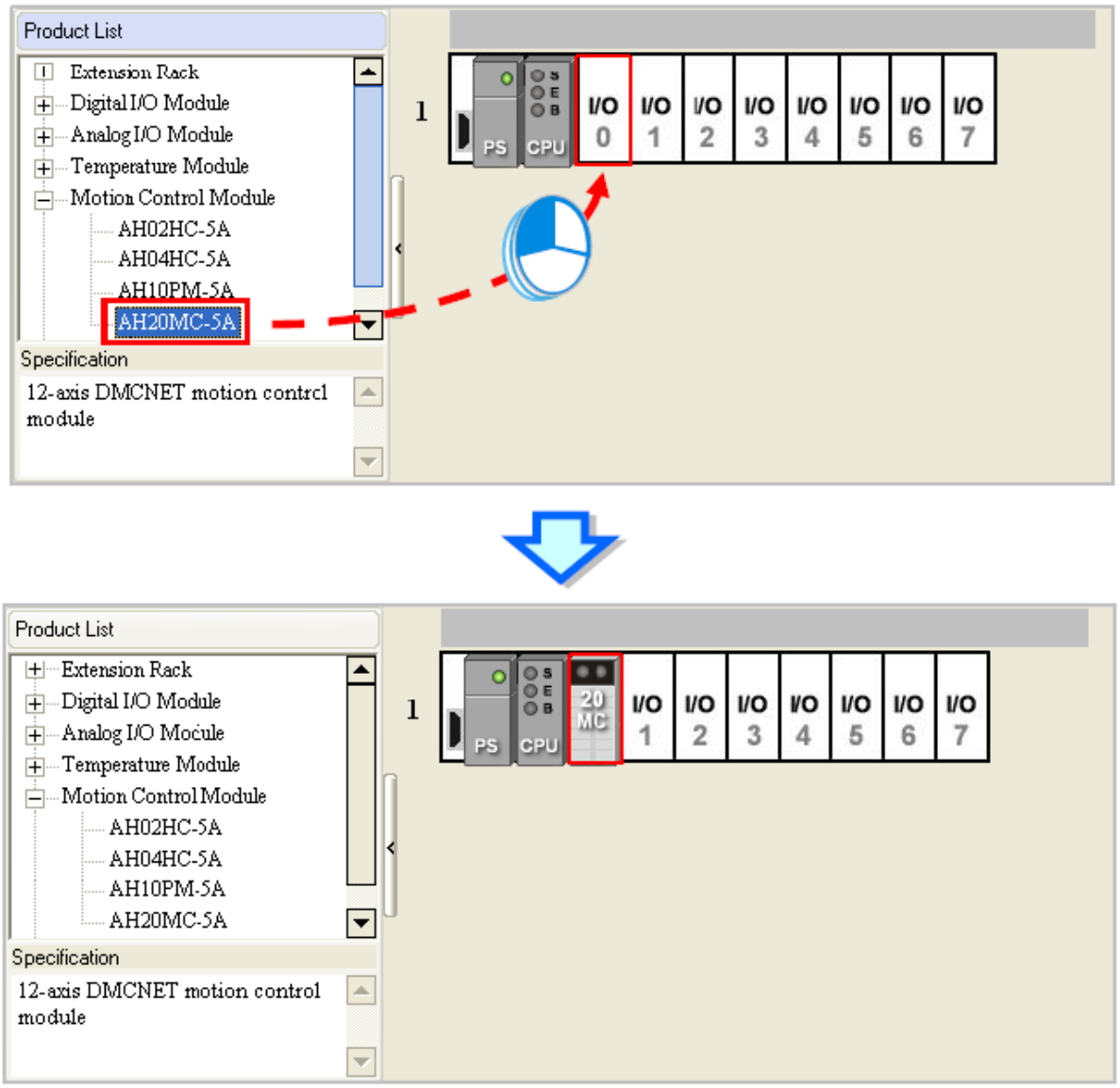
در قسمت مصور می‌توان Rackها و Slot های سیستم را مشاهده نمود. در هر لحظه فقط اطلاعات مربوط به Rack انتخاب شده در قسمت Information List نمایش داده می‌شود.



شکل ۳-۳۰ Rackها و Slotها

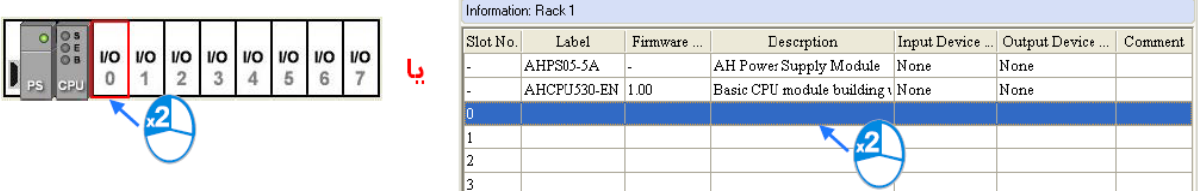
- **اضافه کردن ماژول:**

۱- با کشیدن ماژول مورد نظر از قسمت Product List به Slot خالی

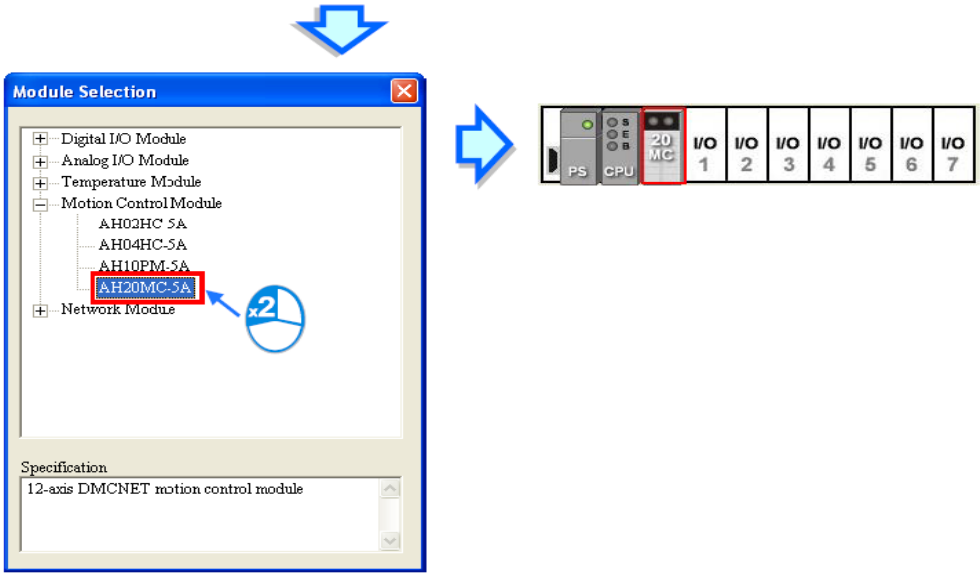


شکل ۳-۳۱ اضافه کردن ماژول جدید از Product List

۲- با دوبار کلیک بر روی Slot خالی در قسمت System Configuration area و یا Information List و انتخاب ماژول مورد نظر از میان ماژول‌های پیشنهادی



شکل ۳-۳۲ اضافه کردن ماژول جدید در System Configuration area و یا Information List–قسمت اول

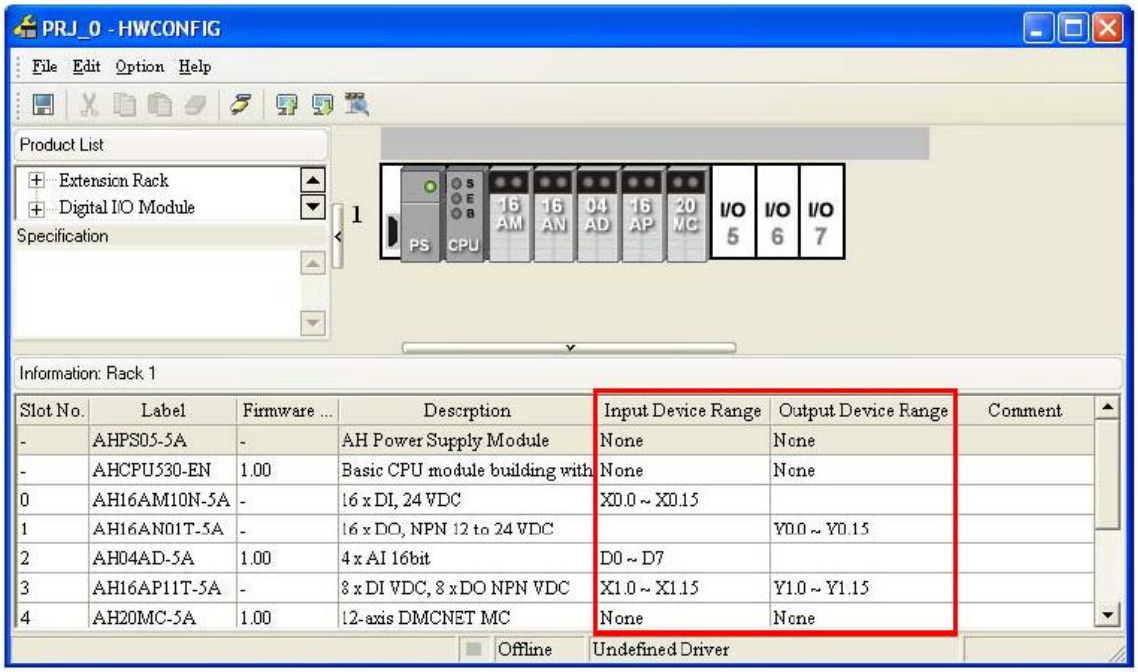


شکل ۳-۳۳ اضافه کردن ماژول جدید در System Configuration area و یا Information List–قسمت دوم

همچنین برای حذف ماژولی خاص می‌توانیم آن را انتخاب کرده و سپس کلیدDelete را در صفحه کلید بفشاریم.

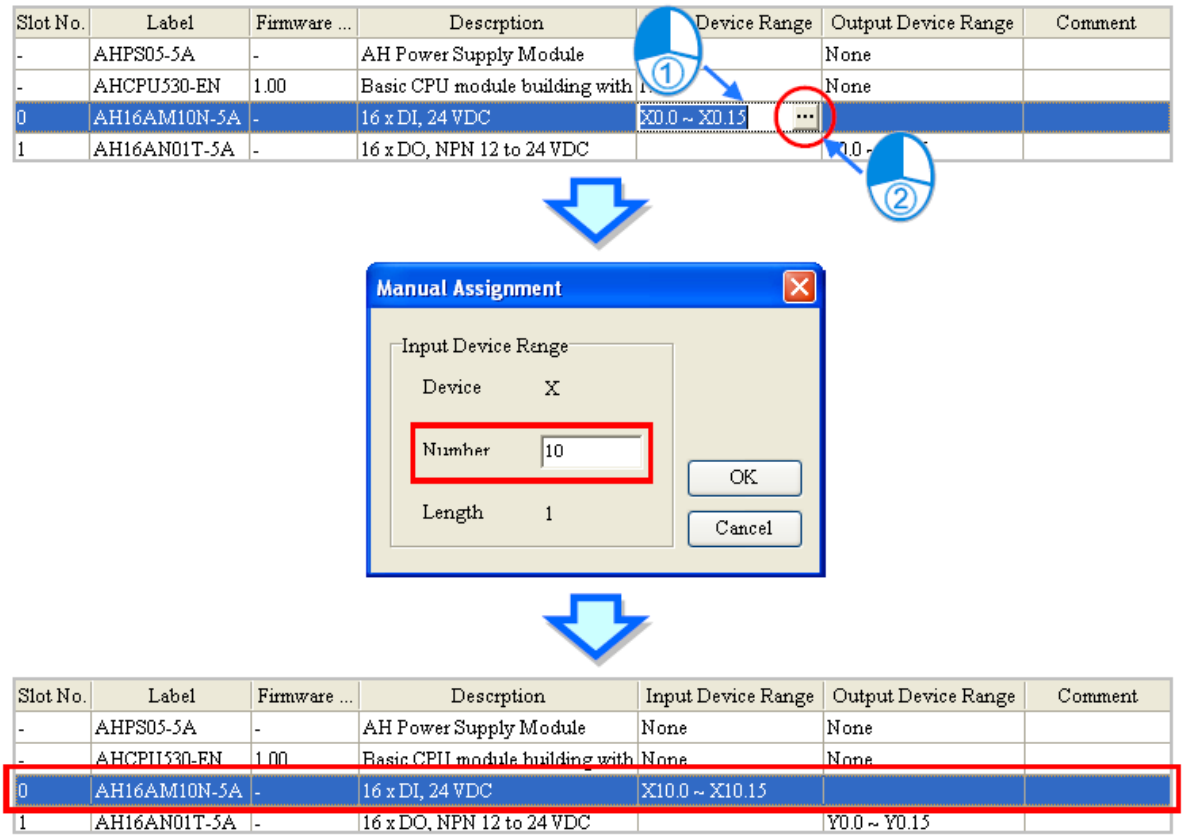
- **تخصیص آدرس به ماژول‌های اضافه شده:**

پس از تعیین ماژول‌های ورودی و خروجی نرم افزار به هرکدام از آن‌ها آدرسی اختصاص می‌دهد که می‌توانیم از آن‌ها در برنامه نویسی استفاده کنیم. این آدرس‌ها در دو ستون Input Device Range و Output Device Range مشخص شده‌اند.



شکل ۳-۳۴ آدرس ورودی و خروجی ها در پیکربندی

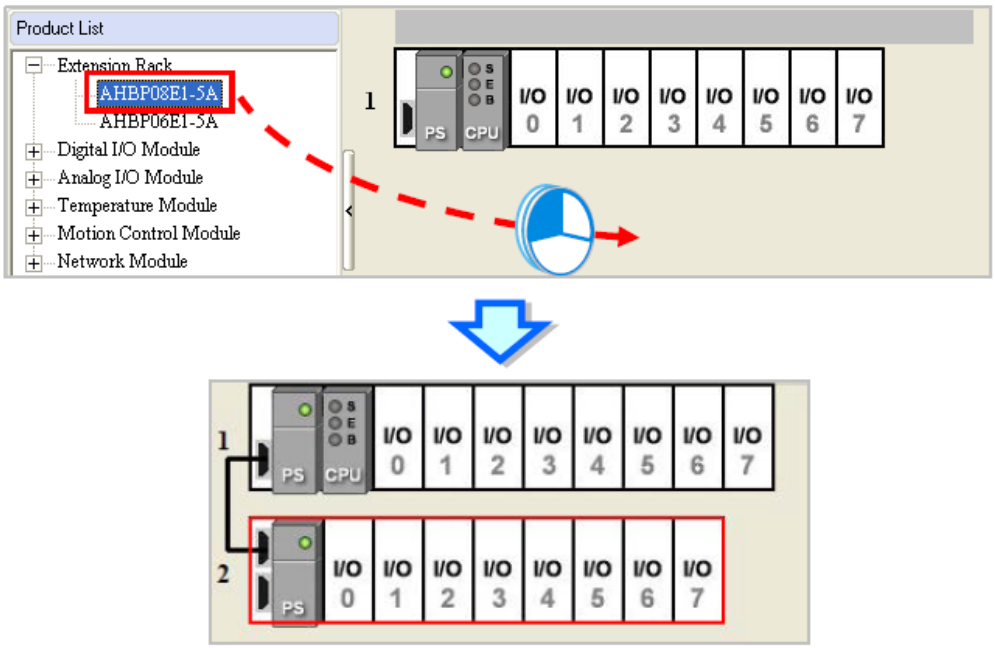
به عنوان مثالی در شکل زیر، در Slot شماره صفر، ماژول AH16AM10N-5A قرار گرفته دارد که ورودی ۱۶ تایی دیجیتال می‌باشد. سیستم آدرس های X0.0 الی X0.15 را به این ورودی اختصاص داده است. البته کاربر می‌تواند با کلیک بر روی سلول‌های آدرس، آن‌ها را به صورت دستی و به صورت دلخواه تغییر دهد. در این حالت اگر به اشتباه از آدرس رزرو شده‌ای استفاده کنیم، سیستم ضمن جلوگیری از اختصاص آدرس به کاربر پیرامون این موضوع هشدار می‌دهد.



شکل ۳-۳۵ نحوه تغییر آدرس ورودی و خروجی ها در پیکربندی

- **اضافه کردن Rackجدید**

در قسمت Extension Rack یکی از دو نوع Rack شش و یا هشت Slot را انتخاب کرده و به فضای خالی System Configuration area منتقل می‌کنیم.

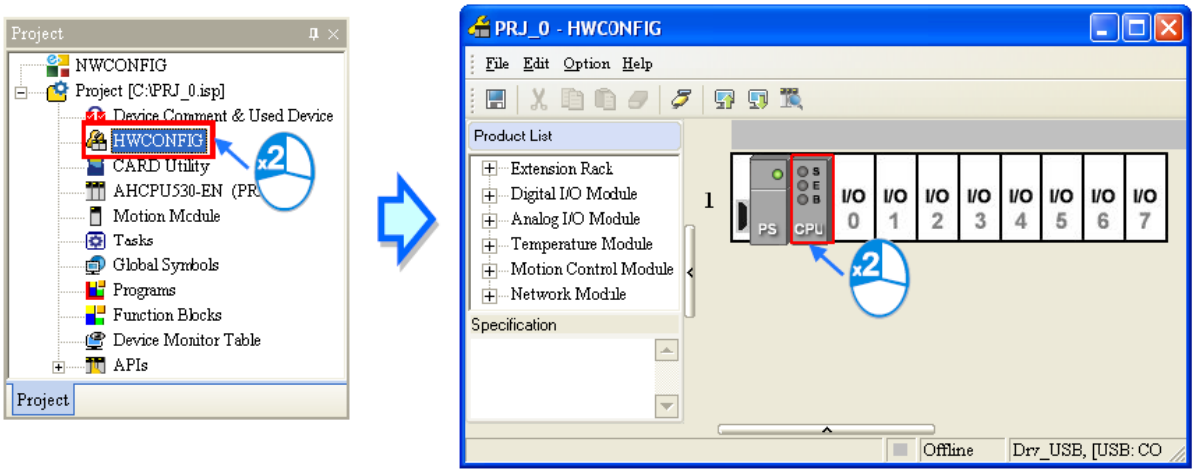


شکل ۳-۳۶ اضافه کردن Rack جدید

۳-۳-۳- تنظیم پارامترهای پردازنده AH500

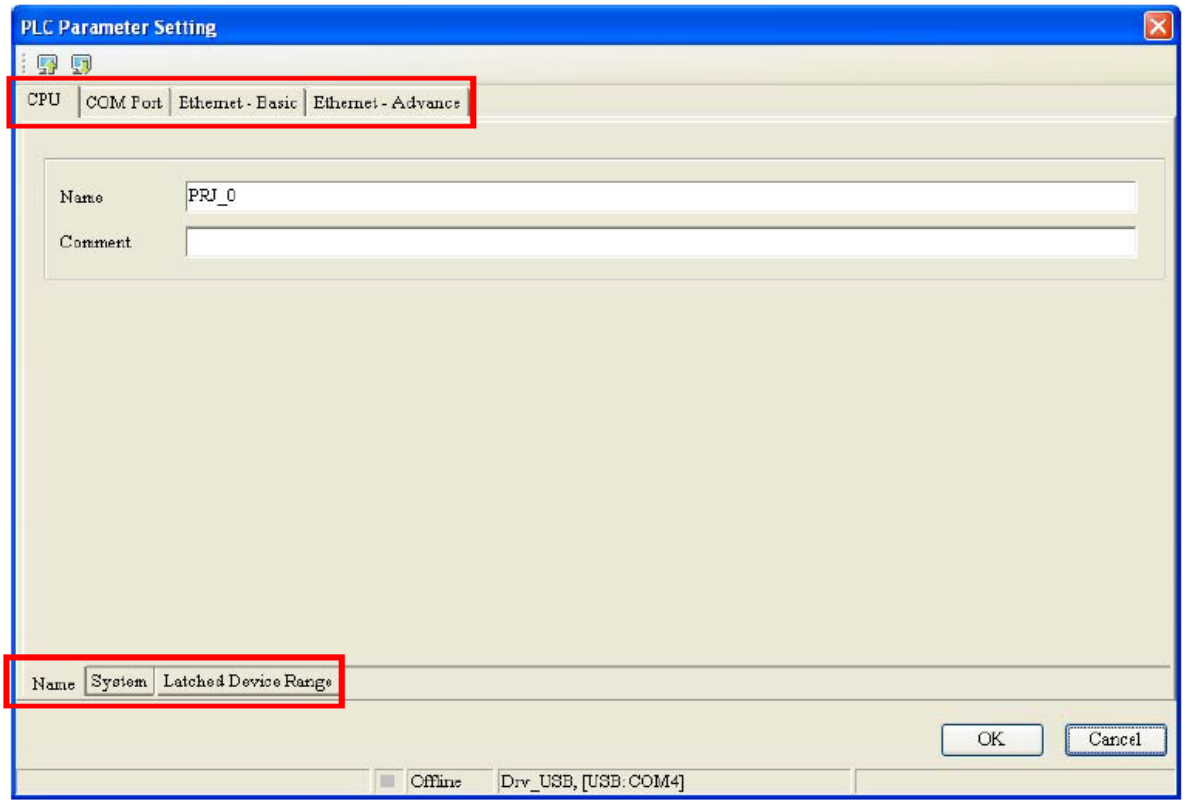
برای اینکار در HWCONFIG بر روی Slot دوم که همان CPU می‌باشد دوبار کلیک نمایید تا صفحه تنظیمات CPU باز شود. در این پنجره با عنوان "PLC Parameter Setting" می‌توانید تنظیمات مربوط به بخش‌های مختلف PLC و CPU را انجام دهید.

تذکر مهم: قبل از تنظیم پارامترهای CPU جهت جلوگیری از آسیب های احتمالی، حتما توضیحات مربوط به آن را در Operation Manual مطالعه فرمایید.



شکل ۳-۳۷ محل تنظیم پارامترهای پردازنده

در صفحه تنظیمات CPU تعدادی سربرگ وجود دارد که هرکدام از آن‌ها شامل چند زیربرگ هستند.



شکل ۳-۳۸ پنجره تنظیمات CPU

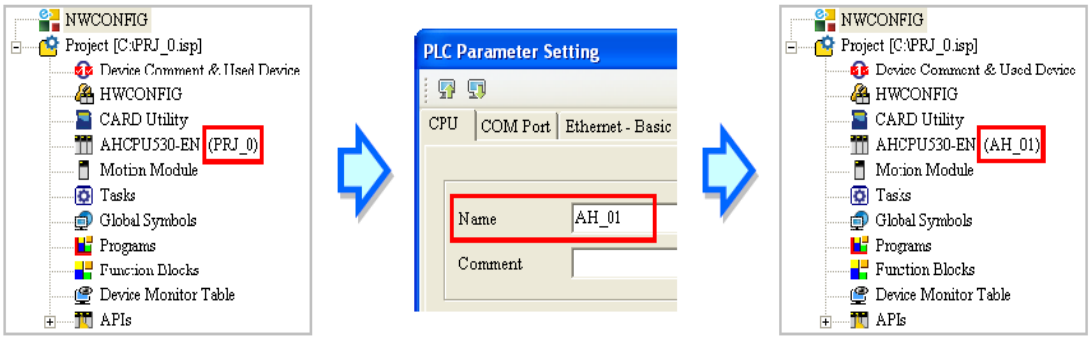
۳-۳-۳-۱- تنظیمات سربرگ CPU

- **زیربرگ Name**

در این قسمت می‌توان نام پردازنده و توضیحاتی پیرامون آن را وارد کرد. به صورت پیش فرض این نام مشابه نام پروژه انتخاب شده است.

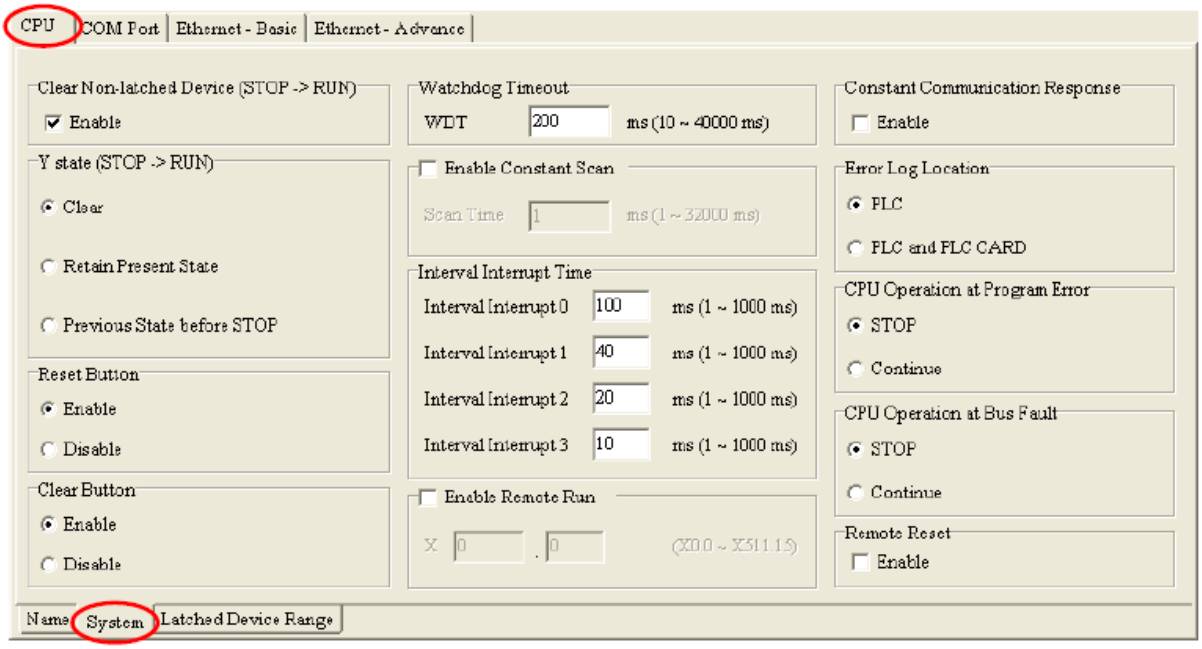
شکل ۳-۳۹ تنظیمات CPU – زیربرگ NAME

زمانی که دستگاه‌های با مدل مشابه در سیستم ما وجود دارند، می‌توانیم با اختصاص نام منحصر به فرد به آن‌ها از بروز اشتباه جلوگیری کنیم.



شکل ۳-۴۰ اختصاص نام انحصاری بر روی سخت افزار های مشابه

- **زیربرگ System**

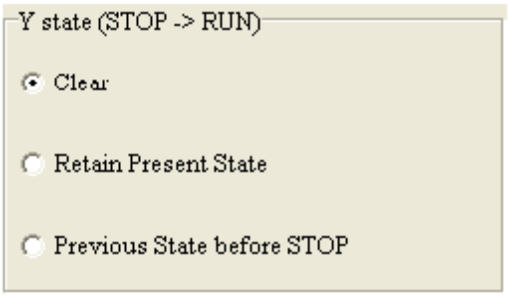


شکل ۳-۴۱ تنظیمات CPU – زیربرگ System

- **Clear Non-latched Device (STOP -> RUN)**: اگر فعال باشد، در زمان شروع به کار پردازنده حافظه‌های غیر نگهدار را پاک می‌کند.

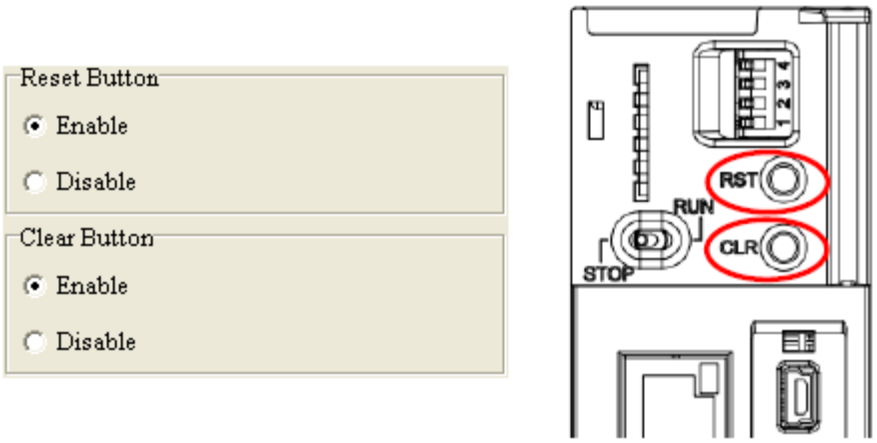
شکل ۳-۴۲ Clear Non-latched Devices

- **Y state (STOP -> RUN)**: در این بخش می‌توانیم مشخص کنیم که وضعیت حافظه‌های Y در زمان شروع به کار پردازنده به چه صورت باشد: پاک شوند، مقدار خود را حفظ کنند و یا اینکه مقدار قبل از Stop خود را بگیرند.



شکل ۳-۴۳ Y State

- Reset Button & Clear Button: مشخص می‌کند که کلیدهای RST و CLR تعبیه شده بر روی ماژول CPU فعال باشند.

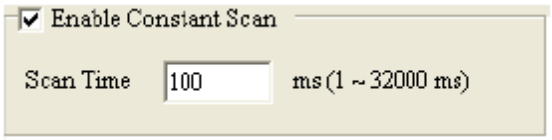


شکل ۳-۴۴ Reset & Clear Button

- Watchdog Timeout: این زمان سنج برای اطمینان از اجرای برنامه در زمان مقرر به کار می‌رود. اگر CPU نتواند اجرای یک دور برنامه را (زمان SCAN[1]) در مدت مشخص شده توسط Watchdog Timer به پایان رساند آنگاه این زمان سنج پردازنده را به حالت خطا می‌برد.

شکل ۳-۴۵ Watchdog Timeout

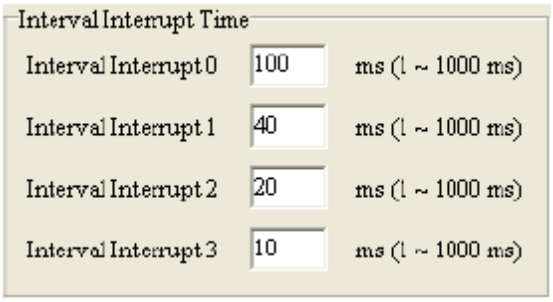
- Enable Constant Scan: با فعال شدن این گزینه اگر زمان SCAN کمتر از زمان تعیین شده در این قسمت باشد، آنگاه CPU تا زمان تعیین شده صبر کرده و سپس سیکل جدید را اجرا می‌کند. در غیر این صورت اگر زمان SCAN بیشتر از زمان تعیین شده باشد، این گزینه تاثیری در اجرا نخواهد داشت.



شکل ۳-۴۶ Enable Constant Scan

[1] زمان SCAN زمانی است که PLC یکبار تمامی دستورات خود را اجرا می‌کند.

- Interval Interrupt Time: چهار وقفه زمانی برای اجرای برنامه در دوره های مشخص زمانی

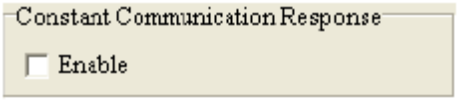


شکل ۳-۴۷ Interval Interrupt Time

- Enable Remote Run: اگر این حالت فعال باشد، می‌توان با استفاده از پورت ورودی مشخصی وضعیت پردازنده را کنترل کرد. به عنوان مثال می‌توان گفت که اگر X0.0 برابر صفر بود پردازنده در حالت STOP و اگر یک بود در حالت RUN باشد.

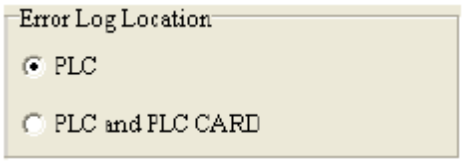
شکل ۳-۴۸ Enable Remote Run

- Constant Communication Response: زمانی که این گزینه فعال نباشد، پیام دریافتی از طریق کابل ارتباطی، در انتهای زمان SCAN مورد بررسی قرار می‌گیرد. در غیر این صورت و با فعال بودن این گزینه، پیام دریافتی با ایجاد وقفه در برنامه مورد بررسی قرار می‌گیرد.



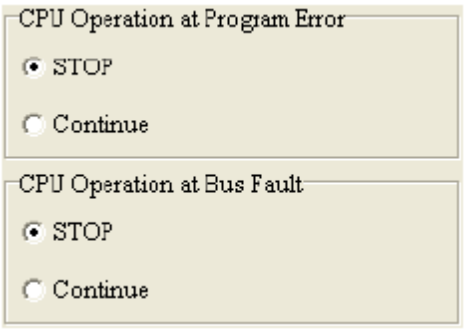
شکل ۳-۴۹ Constant Communication Response

- Error Log Location: برای ذخیره شرح خطاهای رخ داده می‌توانیم از حافظه CPU به همراه کارت حافظه استفاده کنیم. در CPUهای سریAH500 حداکثر بیست شرح خطا قابل ذخیره سازی است و بعد از آن خطاهای جدید جایگزین خطاهای قدیمی تر می‌شوند، در صورت نیاز به فضای ذخیره‌سازی بیشتر لازم است از کارت حافظه نیز کمک گرفته شود.



شکل ۳-۵۰ Error Log Location

- CPU Operation at Program Error & CPU Operation at Bus Fault: می-توان مشخص کرد که پردازنده پس از آنکه در برنامه و یا شبکه خطایی رخ داد، به کار خود ادامه دهد (حالت Countinue) و یا اینکه متوقف شود (حالت Stop)



شکل ۳-۵۱ CPU Operation at Program Error and Bus Fault

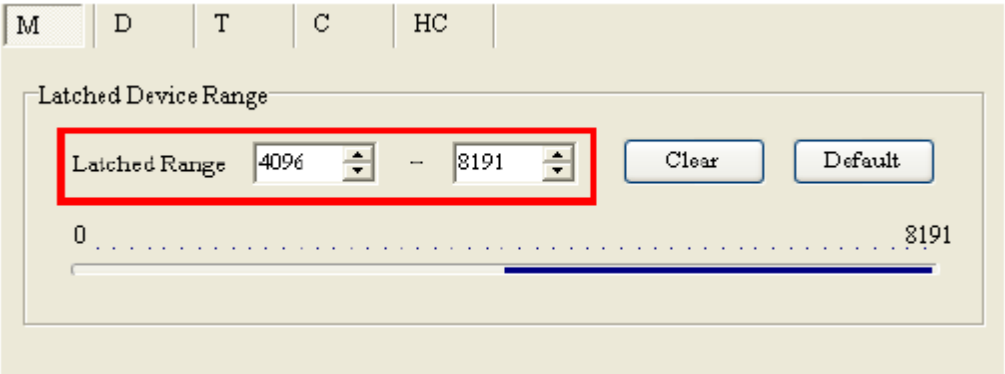
- Remote Reset: با فعال کردن این گزینه می‌توانیم PLC را به صورت مستقیم از طریق کامپیوتر ریست کنیم. برای این کار می‌توان در سربرگ PLC در قسمت Format PLC Setting گزینه Reset PLC Memory(Factory Setting) را انتخاب نمایید.

- **زیربرگ Latched Device Name**

در این پنجره می‌توان محدوده حافظه‌های نگهدارنده داده را تعیین کرد چرا که در CPU ممکن است حافظه‌ها ترتیبی نباشند (مقدار ذخیره شده را نگه ندارند) و بدین منظور باید حافظه‌هایی که نیاز است نگه داشته شوند در این بخش تعیین شوند. در سربرگ‌های M، T، C، HC به ترتیب می‌توان حافظه‌های نگهدارنده رجیستری، زمان سنج، شمارنده و شمارنده‌های پر سرعت را تعیین کرد، در هر سربرگ نیز می‌توان بلوک مورد نظر را انتخاب کرد.

در نمودار میله‌ای مشخص شده در این قسمت می‌توانیم محدوده حافظه را مشخص نماییم. بخش رنگی حافظه نگدارنده تعیین شده را مشخص می‌کند. محدوده حافظه نگهدارنده را می‌توان در قسمت Range مشخص کرد (این مقادیر باید در محدوده مشخص شده در نمودار میله‌ای باشد). با کلیک بر

روی Clear می‌توان تعیین کرد که هیچ کدام از این حافظه‌ها نگهدارنده نمی باشند (در این حالت مقادیر در قسمت Range به صورت ۱- تا ۱- مشخص خواهد شد)

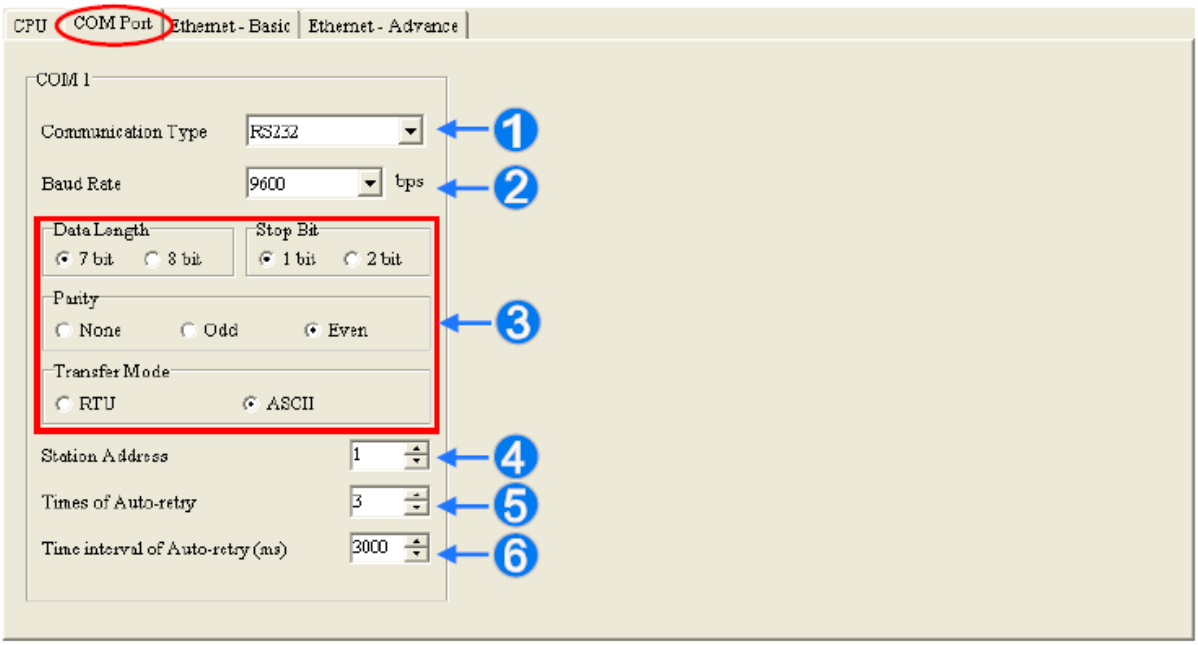


شکل ۳-۵۲ محدوده حافظه های نگهدارنده

با کلیک بر روی Default مقادیر حافظه نگهدارنده به حالت پیش فرض بر می گردد و با کلیک بر روی Read from PLC اگر نرم افزار به PLC متصل باشد، این مقادیر مطابق وضعیت فعلی PLC در خواهند آمد.

۳-۳-۳-۲- سربرگ COM Port

در این بخش می‌توانیم پارامترهای پورت ارتباطی سریال را تنظیم کنیم. اگر CPU دارای دو نوع متفاوت شبکه ارتباطی باشد (مانند خانواده های سری AHCPU5xx-RS2) آنگاه در این قسمت هرکدام از پورت ها دارای قسمت تنظیم مستقل خواهد بود.



شکل ۳-۵۳ تنظیمات پورت سریال در CPU

1 انتخاب نوع پروتکل ارتباطی که می‌تواند RS232 یا RS485 یا RS422 باشد

2 نرخ ارسال و دریافت اطلاعات که باید در فرستنده و گیرنده یکسان تنظیم شود

3 تنظیم پارامترهای پروتکل

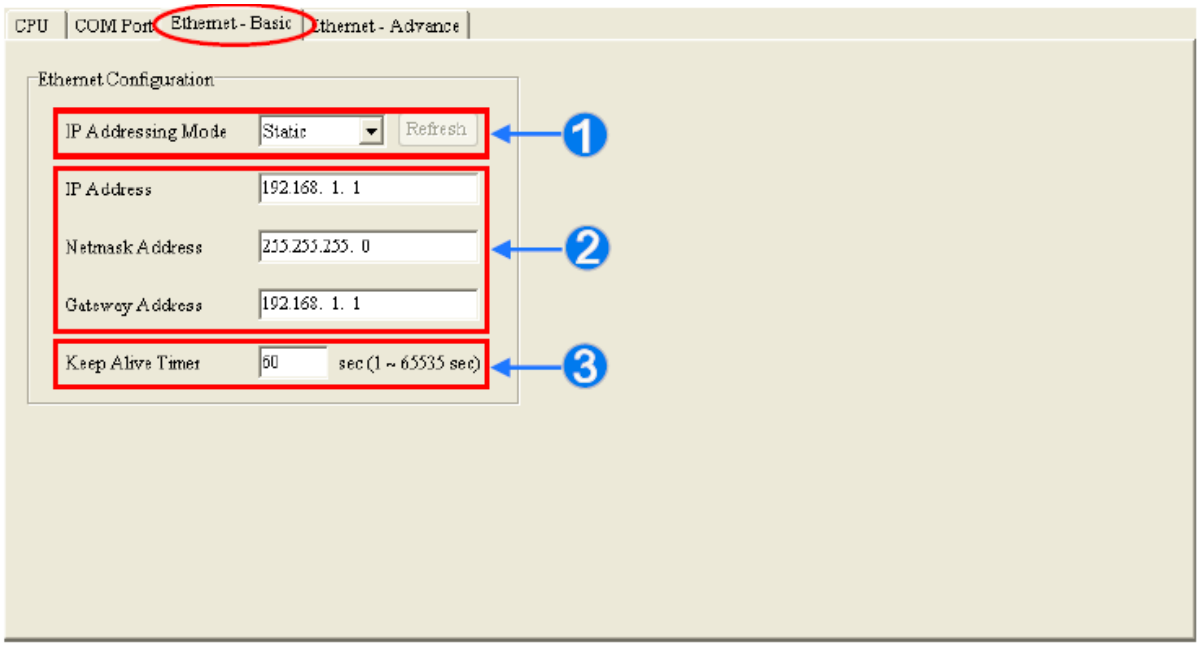
4 تنظیم Station Address، آدرسی برای تجهیزات متصل به شبکه که با استفاده از آن می‌توان داده را به دستگاه مشخصی ارسال کرد. این مقدار می‌تواند بین صفر تا ۲۴۷ باشد. آدرس صفر در شبکه به معنای آن است که داده برای تمامی Slave های درون شبکه است و همه آن‌ها باید آن را دریافت نمایند.

5 تعداد دفعات سعی مجدد CPU برای ارسال پیام

6 حداکثر زمانی که CPU می‌تواند صرف ارسال یک پیام کند

۳-۳-۳-۳- سربرگ Ethernet-Basic

در صورتی که CPU مجهز به پورت Ethernet باشد، می‌توان تنظیمات مربوط به آن را در این قسمت انجام داد.



شکل ۳-۵۴ تنظیمات پایه Ethernet در CPU

1 انتخاب نوعIP به صورت Static و یا Dynamic

2 در حالتی که IP را به صورت Static تنظیم می‌کنیم، کاربر می‌بایست آدرس IP و Subnet Mask و Geteway Address را در این بخش مشخص کند

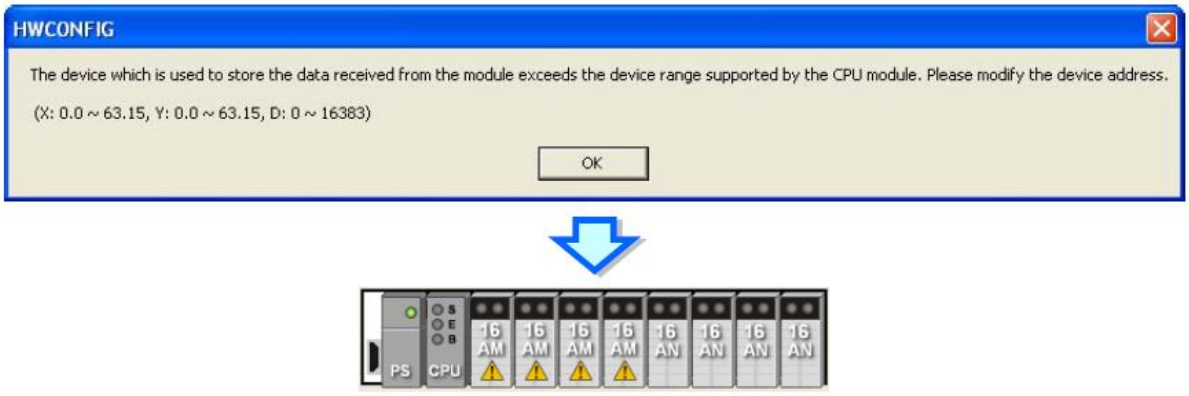
3 اگر به اندازه زمان مشخص شده در این بخش انتقال داده‌ای صورت نگیرد، ارتباط CPU با شبکه قطع خواهد شد.

۳-۳-۴-۳- سربرگ Ethernet-Advanced

در این قسمت کاربر می‌تواند عملکردهای مختلفی را برای CPU در ارتباط با شبکه Ethernet تعریف کند همانند: لحاظ کردن فیلتر برای ارتباط با IP-ها ، تنظیم زمان PLC، ارسال ایمیل، تنظیمات نحوه و زمان و علت ارسال ایمیل، مشاهده وضعیت سیستم بر روی وب و غیره.

۳-۳-۴- ذخیره، بارگزاری و استخراج تنظیمات پردازنده

پس از اتمام تنظیمات پارامترهایPLC در صفحه HWCONFIG، با کلیک بر روی می‌توان تغییرات داده شده را دخیره کرد. با کلیک بر روی گزینه می‌توانیم تنظیمات سخت افزاری را بر روی PLC بارگزاری[۱] نماییم. همچنین برای استخراج[۲] وضعیت فعلی CPU می‌توانیم از استفاده نماییم. در حالتی که خطایی در تنظیمات سخت افزاری رخ دهد (مثلا پردازشگر محدوده خواسته شده در ورودی/خروجی ها را پوشش ندهد) نمی‌توان آن را ذخیره و دانلود کرد، در این حالت ابتدا باید مشکلات پیش آمده را برطرف نمود.



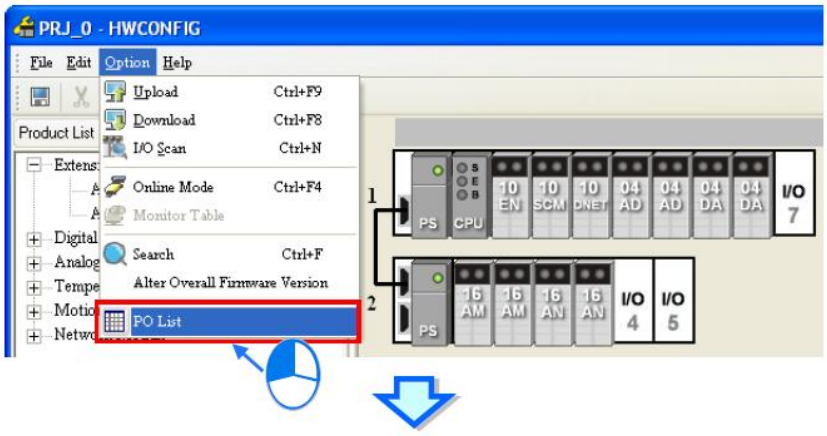
شکل ۳-۵۵ رخ دادن خطا در هنگام بارگزاری پروژه

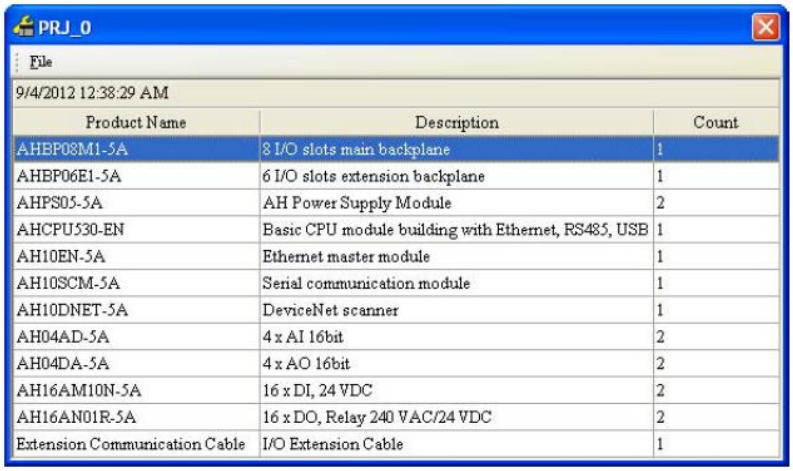
[۱] Download

[۲] Upload

۳-۳-۵- مدیریت پارامترها و عیب یابی از طریق شبکه در AH500

کاربر می‌تواند از طریق منویOption و گزینه PO List به لیست سخت‌افزارهای موجود در پروژه دستیابی داشته باشد.




PRJ_0

File

9/4/2012 12:38:29 AM

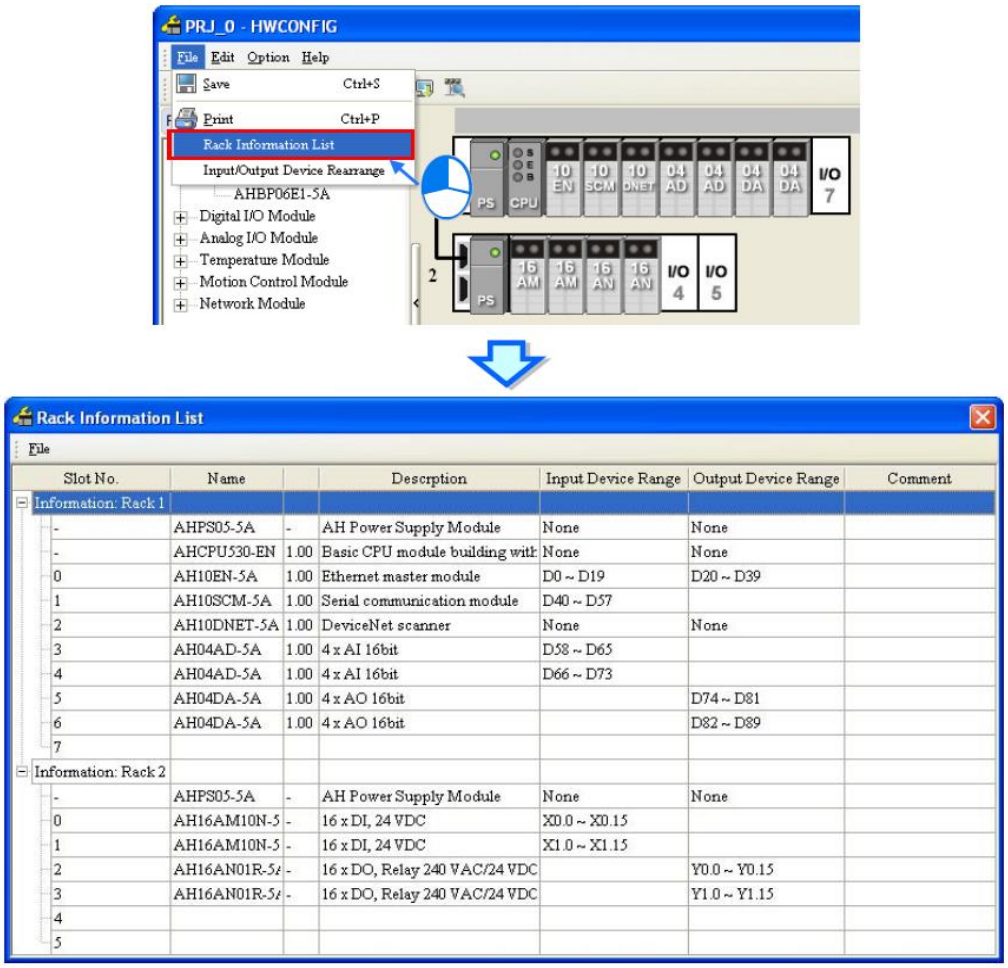
Product Name	Description	Count
AHBP08M1-5A	8 I/O slots main backplane	1
AHBP06E1-5A	6 I/O slots extension backplane	1
AHPS05-5A	AH Power Supply Module	2
AHCPU530-EN	Basic CPU module building with Ethernet, RS485, USB	1
AH10EN-5A	Ethernet master module	1
AH10SCM-5A	Serial communication module	1
AH10DNET-5A	DeviceNet scanner	1
AH04AD-5A	4 x AI 16bit	2
AH04DA-5A	4 x AO 16bit	2
AH16AM10N-5A	16 x DI, 24 VDC	2
AH16AN01R-5A	16 x DO, Relay 240 VAC/24 VDC	2
Extension Communication Cable	I/O Extension Cable	1

شکل ۳-۵۶ لیست سخت افزارهای موجود در پروژه

اطلاعات مورد نظر از طریق منوی فایل و گزینه Export تحت فرمت CSV قابل استخراج می‌باشند. گفتنی است که فرمت استاندارد CSV توسط مایکروسافت اکسل[۱] قابل ویرایش است.

به همین ترتیب در صفحه HWCONFIG از طریق منوی فایل و گزینه Rack Information List می‌توان به اطلاعات Rack ها به صورت لیست شده دسترسی پیدا کرد.

[۱] MicroSoft Excel

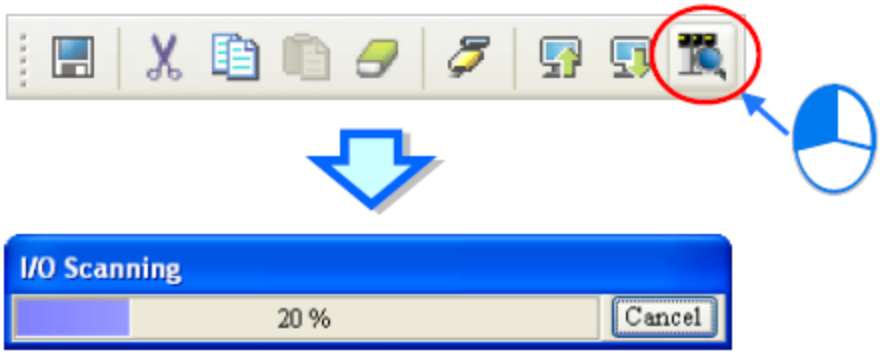


Slot No.	Name		Description	Input Device Range	Output Device Range	Comment
Information: Rack 1						
-	AHPS05-5A	-	AH Power Supply Module	None	None	
-	AHCPU530-EN	1.00	Basic CPU module building with	None	None	
0	AH10EN-5A	1.00	Ethernet master module	D0 ~ D19	D20 ~ D39	
1	AH10SCM-5A	1.00	Serial communication module	D40 ~ D57		
2	AH10DNET-5A	1.00	DeviceNet scanner	None	None	
3	AH04AD-5A	1.00	4 x AI 16bit	D58 ~ D65		
4	AH04AD-5A	1.00	4 x AI 16bit	D66 ~ D73		
5	AH04DA-5A	1.00	4 x AO 16bit		D74 ~ D81	
6	AH04DA-5A	1.00	4 x AO 16bit		D82 ~ D89	
7						
Information: Rack 2						
-	AHPS05-5A	-	AH Power Supply Module	None	None	
0	AH16AM10N-5	-	16 x DI, 24 VDC	X0.0 ~ X0.15		
1	AH16AM10N-5	-	16 x DI, 24 VDC	X1.0 ~ X1.15		
2	AH16AN01R-5A	-	16 x DO, Relay 240 VAC/24 VDC		Y0.0 ~ Y0.15	
3	AH16AN01R-5A	-	16 x DO, Relay 240 VAC/24 VDC		Y1.0 ~ Y1.15	
4						
5						

شکل ۳-۵۷ اطلاعات Rackها در پروژه

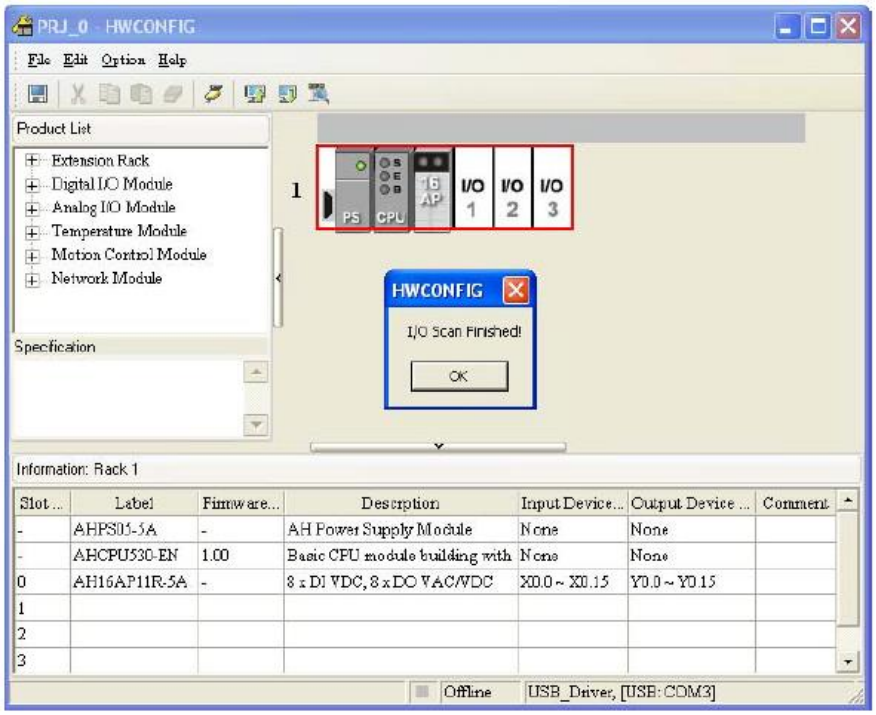
۳-۳-۶- اسکن ورودی و خروجی ها

زمانی که ارتباط با PLC از طریق شبکه برقرار است، می‌توان با کلیک بر روی  در پنجره HWCONFIG وضعیت سخت‌افزارهای مرتبط با PLC را اسکن و آن را جایگزین تنظیمات فعلی نمود، توجه شود که در این حالت وضعیت پارامترهای PLC در CPU تغییر نمی‌کند.



شکل ۳-۵۸ ارتباط ISPSoft با PLC با کمک برنامه COMMGR

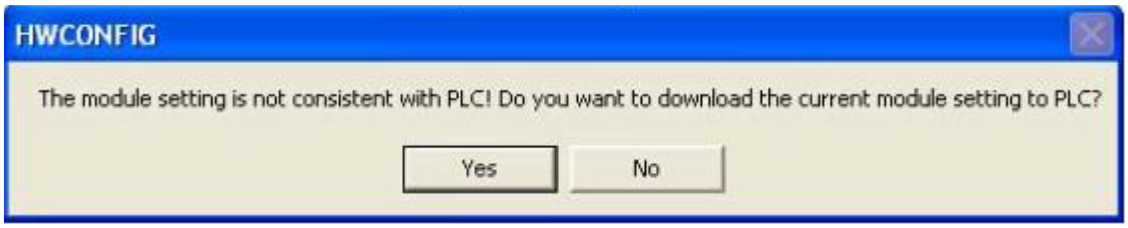
ممکن است این پرسش به وجود آید که تفاوت اسکن و استخراج (Upload) چیست؟ در حالی که Upload تنظیمات CPU را استخراج می‌کند ولی اسکن وضعیت فعلی تجهیزات مرتب با CPU را بررسی می‌کند. برای درک بهتر این موضوع مثالی می‌زنیم، ممکن است CPU به همراه تعدادی ماژول تنظیم شده باشد ولی پس از مدتی آن ماژول‌های به علت خرابی جدا شده باشند، ولی CPU همچنان تنظیماتش شامل آن عناصر است و وضعیت فعلی را نشان نمی‌دهد این در حالی است اسکن وضعیت فعلی و نه تنظیمات CPU را ملاک قرار می‌دهد.



شکل ۳-۵۹ Scan سخت افزار

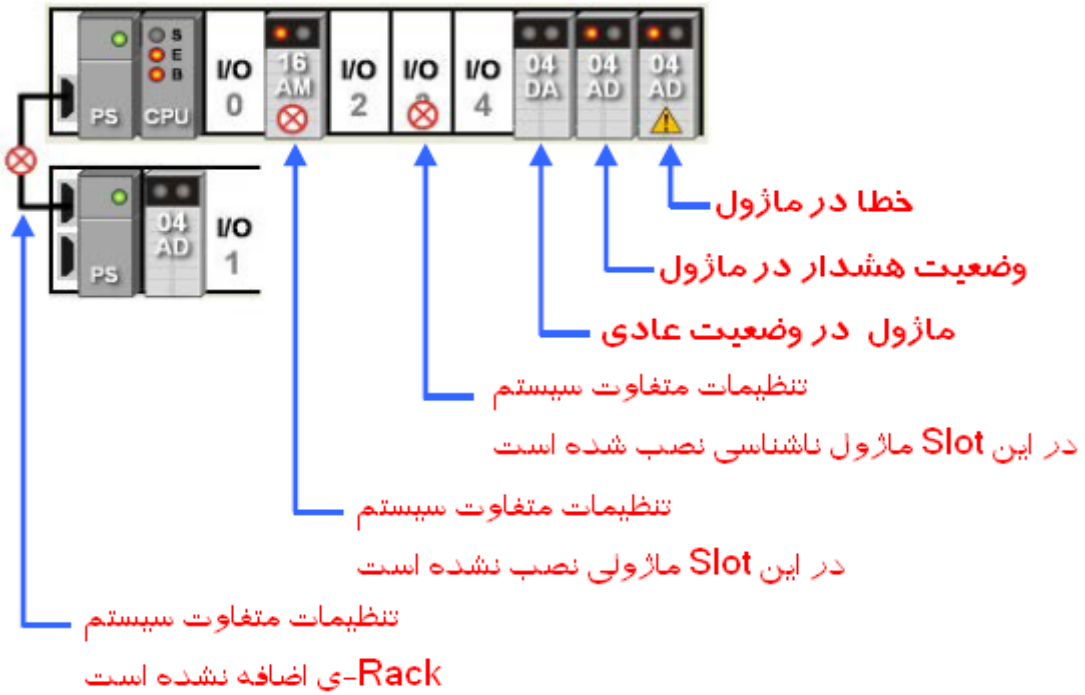
۳-۳-۷- عیب یابی آنلاین

استفاده از نرم افزار ISPSoft برای تنظیمات سخت افزاری نه تنها به صورت آفلاین بلکه به صورت آنلاین نیز امکان پذیر است. با توجه به آنکه HWCONFIG تنظیمات شبکه را برای ارتباط ISPSoft با PLC تنظیم می‌کند، لازم است قبل از هرکاری از ارتباط مناسب ISPSoft با CPU مطمئن شویم. سپس برای برقراری ارتباط آنلاین کافی است بر روی آیکون کلیک کرد (با کلیک دوباره بر روی آن از حالت آنلاین خارج خواهیم شد). در این مرحله نرم افزار بررسی می‌کند که آیا تنظیمات سخت افزاری CPU با تنظیمات مشخص شده در نرم افزار یکی است، در غیر این صورت از کاربر می‌پرسد که آیا لازم است تنظیمات جدید را بر روی CPU دانلود کند؟



شکل ۳-۶۰ بررسی مطابقت سخت افزاری در ارتباط آنلاین

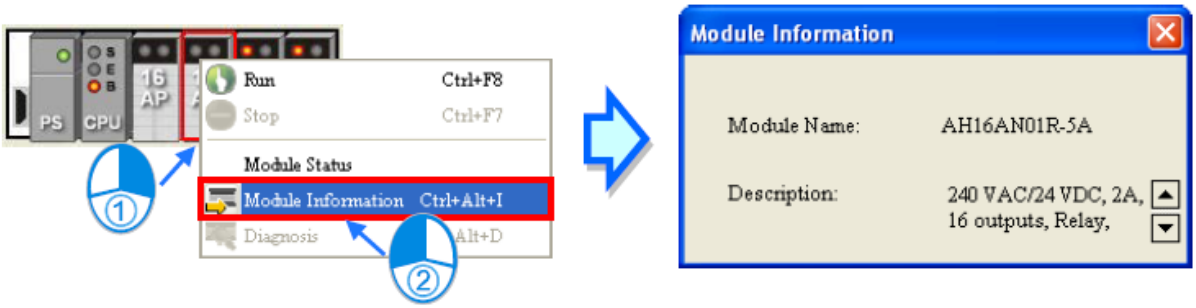
پس از ورود به حالت آنلاین، وضعیت نمایش ماژول در محیط System Configuration ممکن است بسته به وضعیت سیستم واقعی متفاوت باشد. وضعیت LED های نشان داده شده روی CPU شامل LED های RUN، ERROR و BUS FAULT نیز همانند CPU واقعی خواهد بود.



شکل ۳-۶۱ وضعیت آنلاین تنظیمات سخت افزاری سیستم

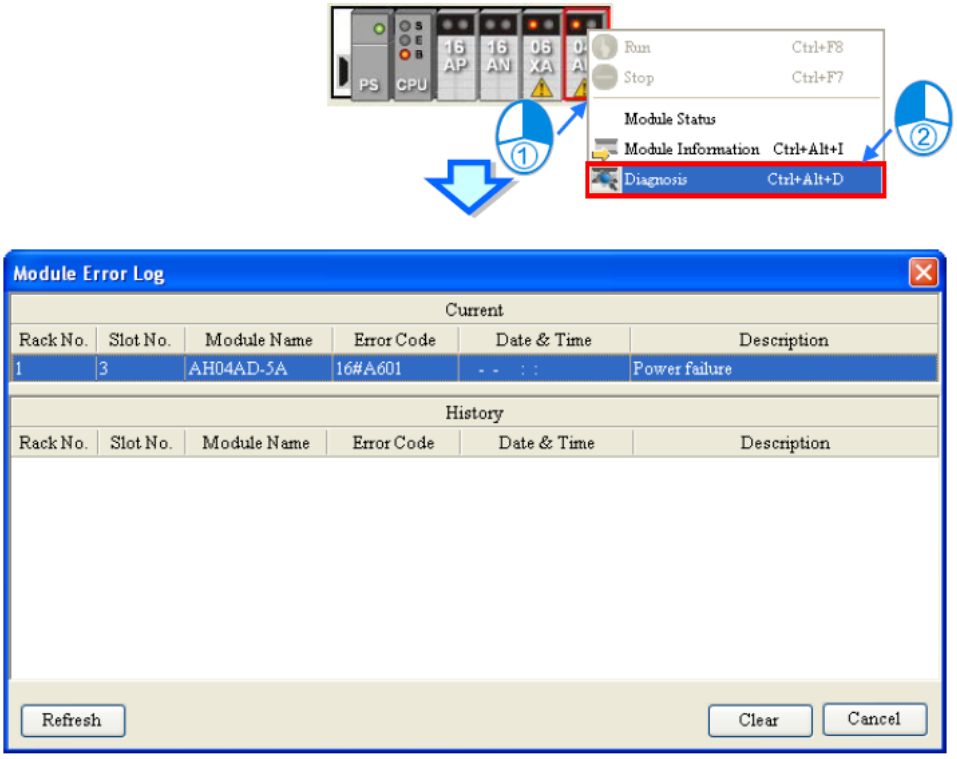
۳-۳-۷-۱- اطلاعات ماژول و عیب یابی

برای آنکه بتوان در حالت آنلاین در قسمت تنظیمات سخت افزاری، اطلاعات مربوط به ماژول را از طریق ارتباط مستقیم با آن استخراج کرد می‌توان بر روی ماژول مورد نظر کلیک راست کرد و سپس Module Information را انتخاب کرد. در این حالت مطابق شکل زیر اطلاعات مربوط به ماژول برای ما به نمایش در خواهد آمد.



شکل ۳-۶۲ اطلاعات آنلاین ماژول

همچنین می‌توان با کلیک راست بر روی هر ماژول دلخواهی در حالت آنلاین و کلیک بر روی Diagnosis پنجره Module Error Log را مشاهده کرد. این پنجره شامل مشکلات فعلی و قبلی ماژول مورد نظر است (مشکلاتی که از قبل ذخیره شده اند). پس از وقوع خطا کاربر باید حتما مشکل را حل نماید چرا که ماژول همراه با خطا عمل نمی‌کند.

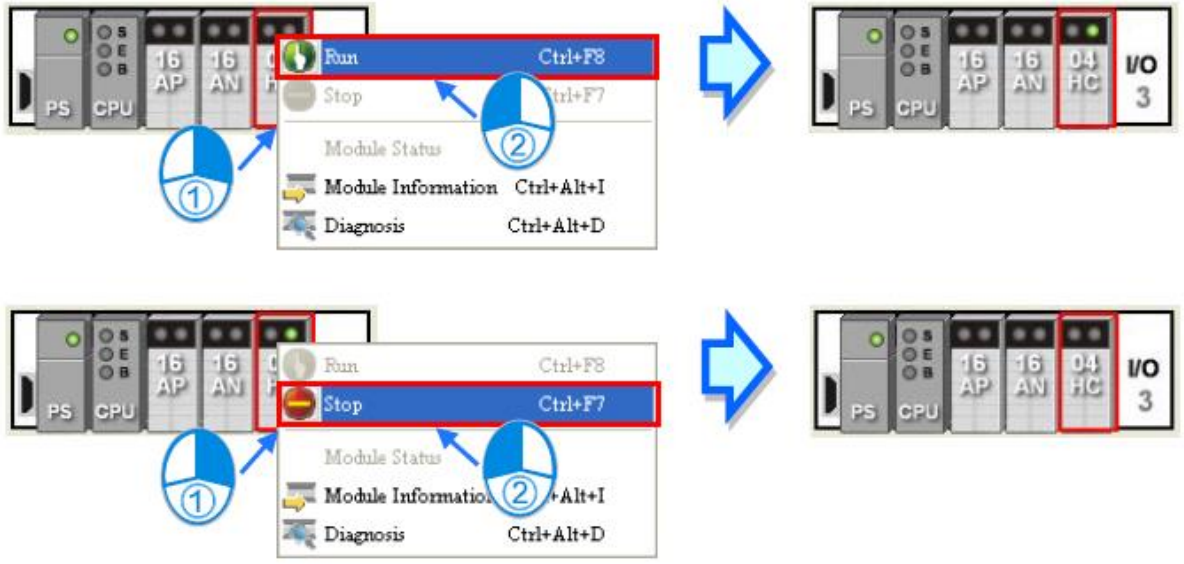


شکل ۳-۶۳ مشکلات فعلی و قبلی ماژول‌ها

۳-۳-۷-۲- تغییر وضعیت اجزای سیستم در حالت آنلاین

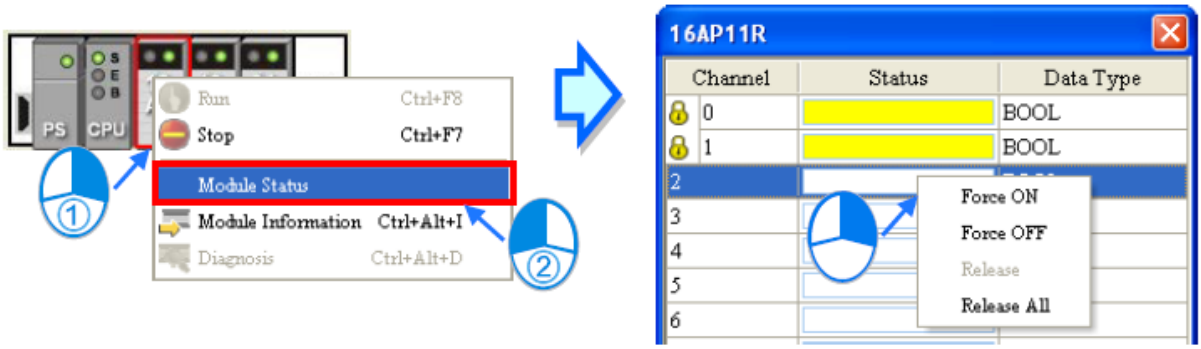
توجه: در این حالت نیاز است کاربر از بی مخاطره بودن عملیات تغییر وضعیت ماژول‌ها در حالت آنلاین مطمئن شود و سپس اقدام به تغییر وضعیت آن نماید.

برای فعال کردن ماژولی در سیستم، می‌توان بر روی آن کلیک راست و سپس RUN را انتخاب کرد، در این حالت LED سبز روی ماژول به معنای روشن بودن آن فعال می‌شود. همچنین برای غیرفعال کردن ماژول می‌توان بر روی آن کلیک چپ کرده و سپس Stop را انتخاب کرد. این در حالی است که اگر ماژول انتخابی ما CPU باشد، حین فعال و غیرفعال کردن آن، تمام ماژول‌های همراه نیز همزمان فعال و غیرفعال خواهند شد.



شکل ۳-۶۴ تغییر وضعیت اجزای سیستم

گاهی اوقات لازم است به منظور خاصی (مانند فراهم کردن ایمنی لازم برای تعمیر دستگاه‌ها) ورودی و یا خروجی مشخصی را در وضعیت ثابتی نگه داریم (به عبارتی Force کنیم). برای اینکار لازم است بر روی آن کلیک راست کرده و سپس Module Status را انتخاب نماید.

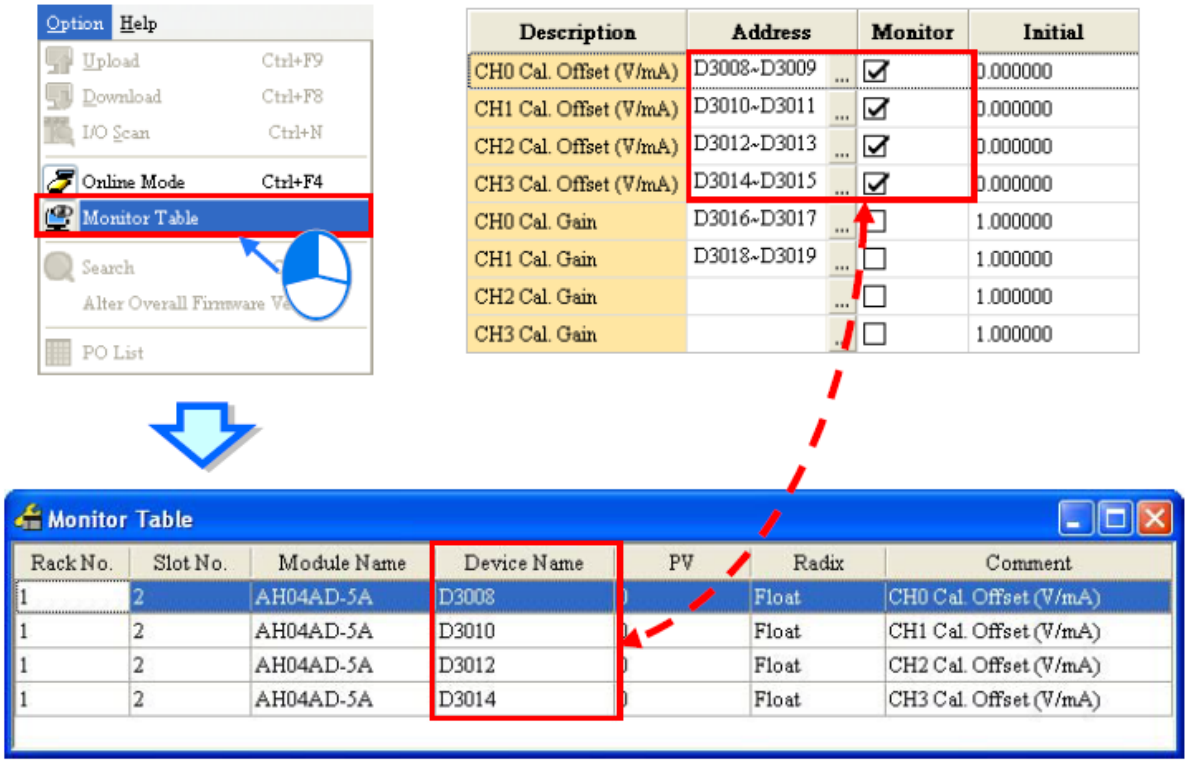


شکل ۳-۶۵ Force کردن مقدار ورودی و خروجی ها

در صفحه مورد نظر کاربر می تواند وضعیت ماژول مورد نظر را مشاهده کند. به عنوان مثال در ورودی یا خروجی‌های دیجیتال کاربر می‌تواند بر روی ماژول مورد نظر کلیک راست کرده و سپس و بر روی

Force ON و یا Force OFF کلیک کرده تا ورودی یا خروجی مورد نظر بالاجبار در وضعیت دلخواه ما ثابت شود. برای اینکه تاثیر موارد فوق را ببینیم لازم است CPU و ماژول‌های ورودی یا خروجی را روشن کنیم. برای خروج از حالت اجبار نیز می‌توانیم بر روی Release کلیک نماییم.

برای مشاهده وضعیت ورودی یا خروجی ها و رجیسترها در حالت آنلاین، کاربر می‌تواند از منوی Option گزینه Monitor Table را انتخاب نماید. پس از انتخاب قسمت های مورد نظر برای مانیتور پارامترها، حافظه‌های متناظر با آن در پنجره Monitor Table لیست خواهد شد.

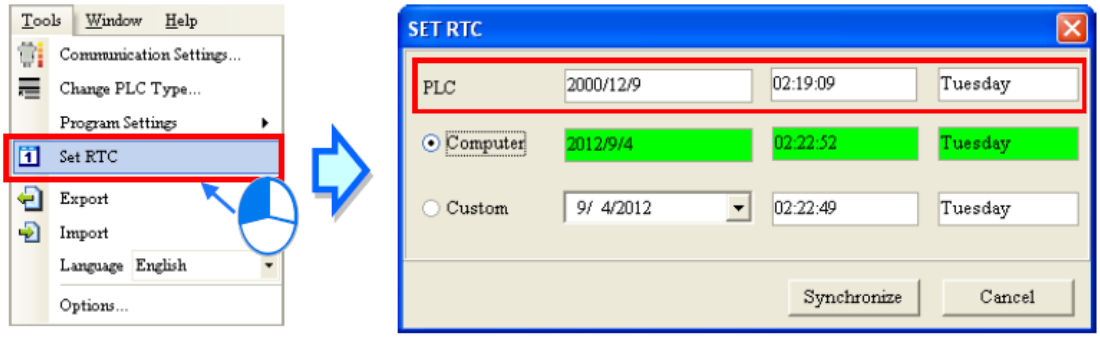


شکل ۳-۶۶ مشاهده وضعیت پارامترهای سیستم

کاربر می‌تواند فرمت نمایش داده در حافظه را با انتخاب سلول متناظر آن در ستون Radix تعیین کند.

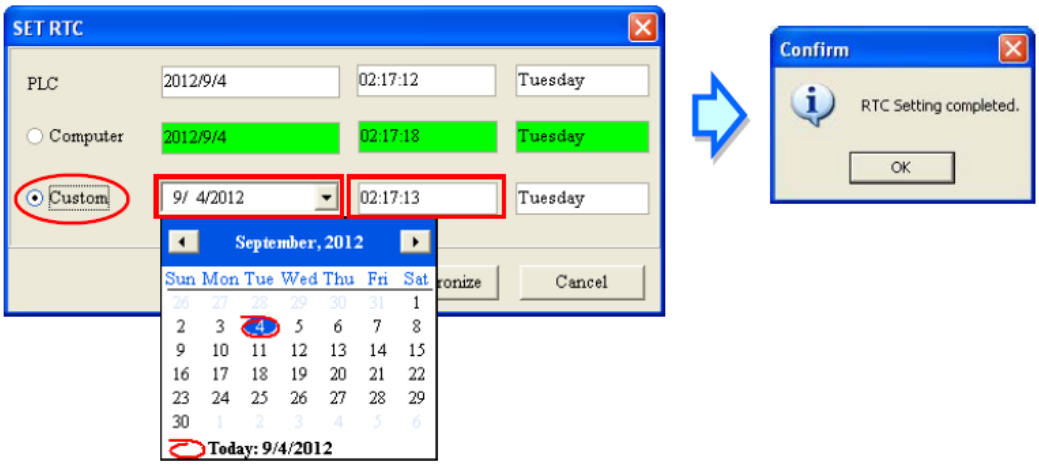
۳-۳-۸- تنظیم ساعت Real-Time

ابتدا در سربرگ Tools در صفحه اصلی ISPSoft بر روی Set RTC کلیک کنید. در صفحه ظاهر شده زمان روبروی عبارت PLC، معادل زمانی است که از حافظه ساعت PLC هنگام باز شدن پنجره در کامپیوتر خوانده شده است.



شکل ۳-۶۷ تنظیم زمان PLC

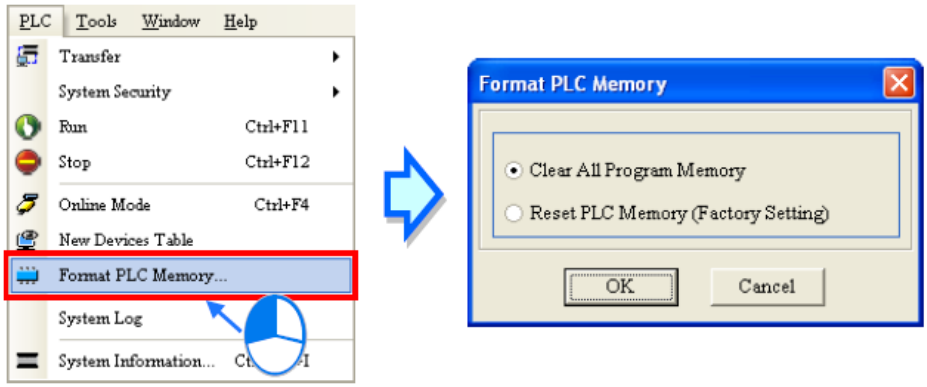
کاربر در این صفحه از طریق انتخاب زمان کامپیوتر با انتخاب Computer یا زمان دلخواه خود با انتخاب Custom می‌تواند زمان PLC را تنظیم نماید. برای اینکار باید بر روی Synchronize کلیک شود.



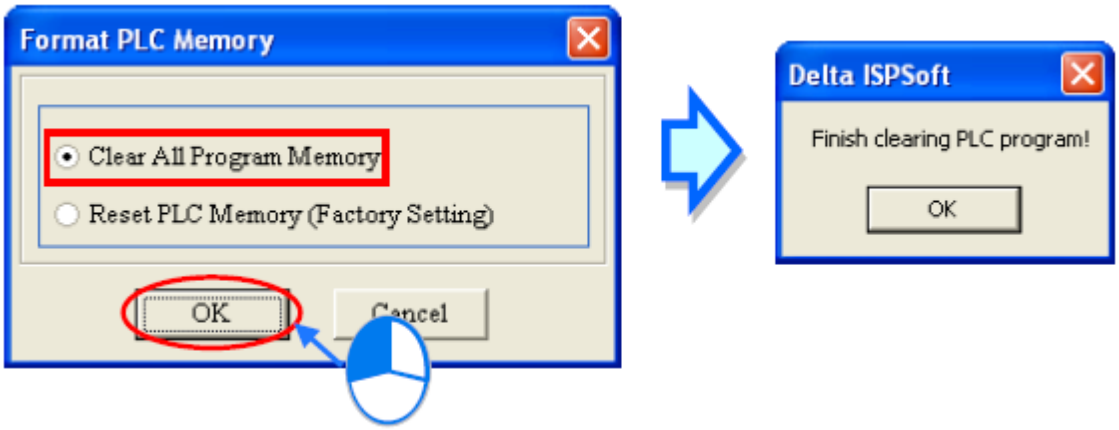
شکل ۳-۶۸ استفاده از ساعت کامپیوتر و یا زمان دلخواه برای PLC

۳-۳-۹- تنظیم حافظه ی PLC

برای پاک کردن حافظه‌ها، یا بازیابی تنظیمات اولیه آن لازم است برنامه به PLC متصل باشد. برای پاک کردن حافظه PLC می‌توان از منوی PLC بر روی Format PLC Memory انتخاب شود. برای اینکار در پنجره ظاهر شده Clear All Program Memory انتخاب شود.

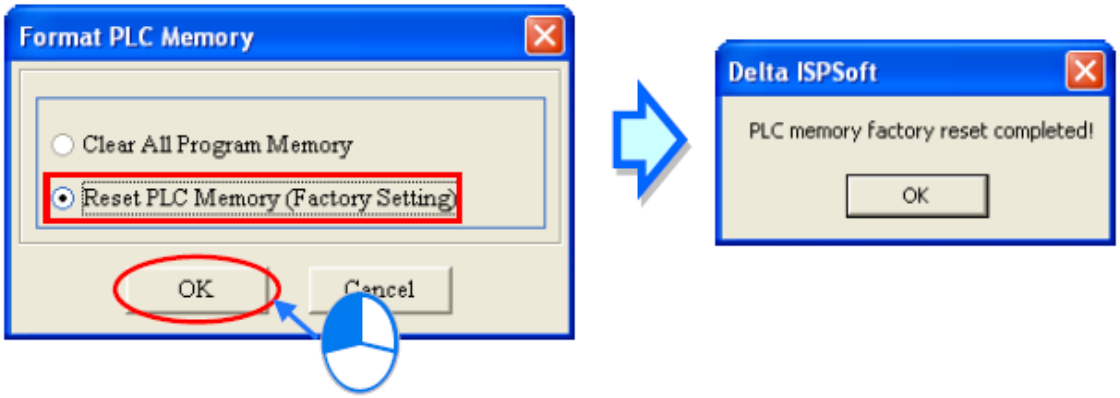


شکل ۳-۶۹ پاک کردن حافظه سیستم – قسمت اول



شکل ۳-۷۰ پاک کردن حافظه سیستم – قسمت دوم

برای بازیابی تنظیمات اولیه و یا همان ریست کردن برنامه نیز می‌توان از گزینه دوم یعنی Reset PLC Memory (Factory Setting) استفاده کرد. توجه شود که در سری AH500 تنها زمانی می‌توانیم از این گزینه استفاده نماییم که گزینه Enable Remote Run در پارامترهای CPU فعال شده باشد.



شکل ۳-۷۱ ریست کردن CPU

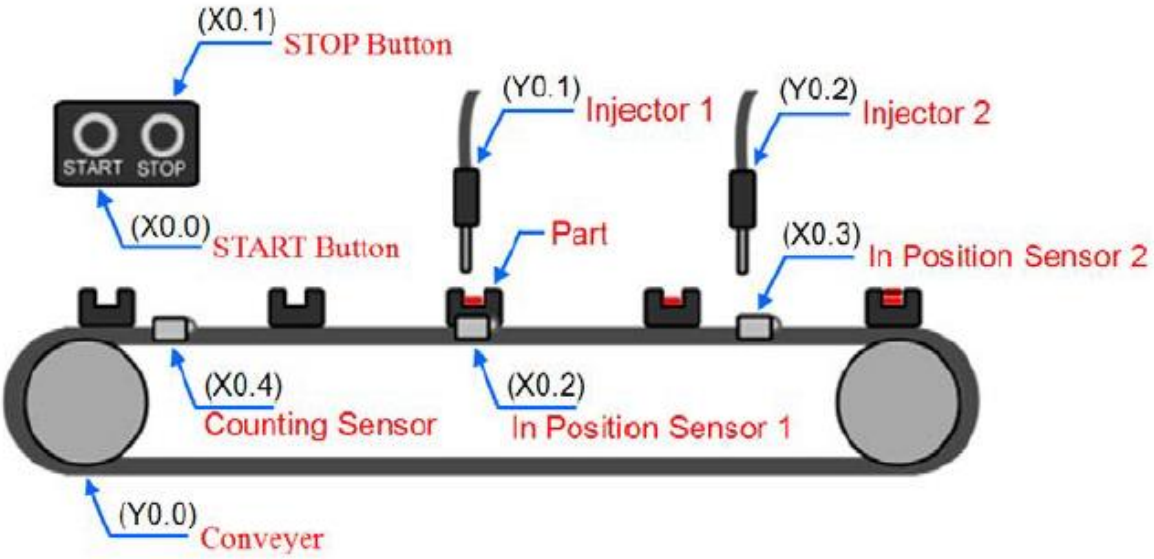
فصل ۴– راه اندازی اولیه

در این فصل به صورت مختصر نحوه شروع کار با ISPSoft را مرور خواهیم کرد. در این بین از زبان برنامه نویسی ladder برای نوشتن برنامه‌ای کوتاه استفاده می‌کنیم.

۴–۱– شرح مثال (خط تولید)

در خط تولیدی مطابق شکل زیر در دو مرحله بر روی قطعات کار تزریق صورت می‌گیرد. تسمه نقاله در این خط تولید قطعات را از چپ به راست منتقل می‌کند. اگر سنسورهای موقعیت (در اینجا در واقع سوئچ‌های دیجیتال نوری جهت تشخیص وجود و یا عدم وجود قطعه کار در موقعیت مورد نظر) قطعه‌ای را حس کنند، دستگاه تزریق باید از طریق سیگنال ارسالی PLC فعال شده و عملیات تزریق را انجام دهد، در این میان نیازی نیست مدت زمانی که تزریق طول می‌کشد در PLC لحاظ شود (این فاکتور به صورت دستی توسط اپراتور تعیین می‌شود.)

همچنین سنسوری در سمت چپ تسمه نقاله تعبیه شده است که هرگاه قطعه‌ای از روبروی آن رد شود، مقدار آن یک واحد افزایش پیدا می‌کند. زمانی که مقدار شمارش شده توسط سنسور به ۱۰۰ برسد، سیگنال "تکمیل" فعال می‌شود. با اینکه در این مثال کاربردی برای این سیگنال در نظر نگرفته‌ایم ولی می‌توان در آینده از آن برای عملکردهای متعددی استفاده کرد.



شکل ۴-۱ نمای مثال خط تولید

می‌خواهیم با استفاده از مدل AHCPU530-EN که از خانواده AH500 است خط تولید مورد نظر را کنترل نماییم. در این میان از ماژول ورودی و خروجی دیجیتال AH16AP11R-5A نیز استفاده خواهیم کرد.

شرح موارد مشخص شده در خط تولید بالا به همراه جدول اختصاص ورودی و خروجی به صورت زیر است (سعی کنید در پروژه‌های خود جدول نام گذاری ورودی و خروجی را برای بررسی راحت تر سیستم تهیه کنید):

جدول ۴-۱: جدول نام گذاری ورودی و خروجی –مثال خط تولید

نام	شرح	سیگنال مرتبط در PLC	نوع سیگنال مرتبط در PLC
START button	کلید شروع	X0.0	ورودی دیجیتال
STOP button	کلید توقف	X0.1	ورودی دیجیتال
In Position Sensor1	سنسور موقعیت ۱	X0.2	ورودی دیجیتال
In Position Sensor2	سنسور موقعیت ۲	X0.3	ورودی دیجیتال
Counting Sensor	سنسور شمارنده	X0.4	ورودی دیجیتال
Conveyer	تسمه نقاله	Y0.0	خروجی دیجیتال
Injector 1	تزریق کننده ۱	Y0.1	خروجی دیجیتال (سیگنال تحریک برای تزریق کننده ۱)
Injector 2	تزریق کننده ۲	Y0.2	خروجی دیجیتال (سیگنال تحریک برای تزریق کننده ۲)

۴–۲– الگوریتم برنامه

۱- زمانی که کلید START (ورودی X0.0) فشرده می‌شود، وضعیت پرچم[۱] "وضعیت" سیستم به حالت "روشن" یا "ON" در می‌آید و تسمه نقاله (خروجی Y0.0) شروع به کار می‌کند. زمانی که کلید STOP (ورودی X0.1) فشرده می‌شود، وضعیت پرچم وضعیت سیستم به حالت

[۱] Flag

"خاموش" و یا "OFF" در می‌آید. همچنین هرگاه پرچم "خطا" روشن می‌شود، تسمه نقاله متوقف می‌شود.

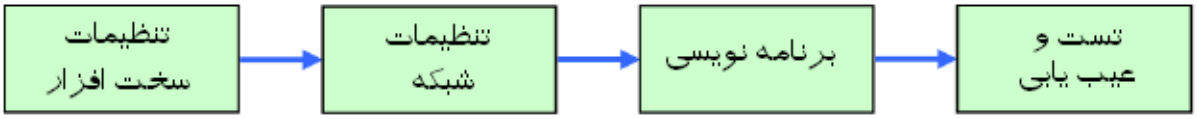
۲- زمانی که سنسور موقعیت۱ (ورودی X0.2) فعال می‌شود، سیگنال تحریک تزریق کننده۱ از طریق خروجی Y0.1 فعال می‌شود. و پس از آن زمانی که سنسور موقعیت۱ به حالت غیرفعال تغییر وضعیت دهد، سیگنال تحریک تزریق کننده۱ نیز غیرفعال می‌شود.

۳- زمانی که سنسور موقعیت۲ (ورودی X0.3) فعال می‌شود، سیگنال تحریک تزریق کننده۲ از طریق خروجی Y0.2 فعال می‌شود. و پس از آن زمانی که سنسور موقعیت۲ به حالت غیرفعال تغییر وضعیت دهد، سیگنال تحریک تزریق کننده۲ نیز غیرفعال می‌شود.

۴- زمانی که ورودی سنسور شمارنده (ورودی X0.4) از حالت خاموش به روشن تغییر وضعیت می‌دهد، مقدار شمارش شده یک عدد افزایش یافته و زمانی که این مقدار مساوی و یا بزرگتر از ۱۰۰ می‌شود، پرچم "تکمیل" به حالت روشن در می‌آید.

۴-۳- مراحل ساخت پروژه در ISPSoft

مراحل ساخت پروژه در ISPSoft مطابق با نمودار زیر است. این مراحل باید بر اساس سیستم واقعی و البته تمایلات شخصی کاربر تنظیم شود.



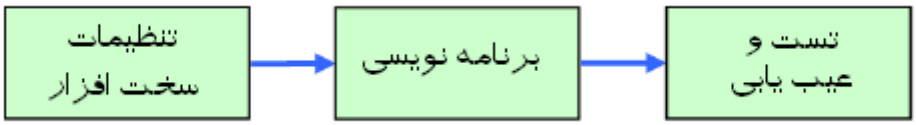
شکل ۴-۲ مراحل انجام پروژه در ISPSoft

- تنظیمات سخت افزاری: کاربران می‌توانند تنظیمات سخت افزاری مانند تعیین حافظه نگه‌دار، اختصاص نام به درگاه‌های ورودی / خروجی و پیکربندی[۱] سخت افزار در نرم افزار را انجام دهند.
- تنظیمات شبکه: (در صورتی که اجزای سیستم از طریق شبکه با یکدیگر در ارتباط باشند) با استفاده از NWCONFIG به راحتی می‌توان تنظیمات مربوط به شبکه را برای تبادل داده انجام داد.
- برنامه نویسی: پس از نوشته شدن برنامه در PLC می‌توان آن را کامپایل و در PLC بارگزاری کرد.

[۱] Configuration

- تست و عیب یابی: با استفاده از امکانات تعبیه شده در ISPSoft کاربر می‌تواند به صورت آنلاین به تست و عیب یابی برنامه پیاده شده در PLC بپردازد.

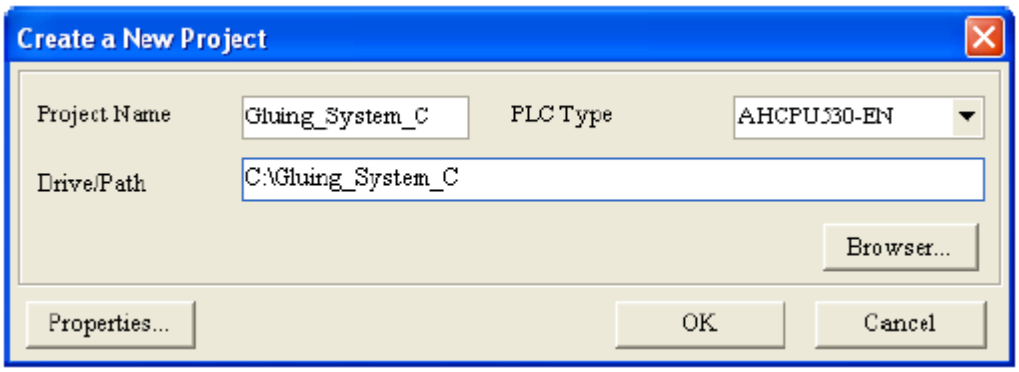
همانطور که مشخص است پروژه نمونه ای که در این فصل مورد بررسی قرار خواهد گرفت، نیازی به شبکه ندارد و در نتیجه پروسه آن به صورت زیر در خواهد آمد:



شکل ۴-۳ مراحل انجام پروژه نمونه خط تولید

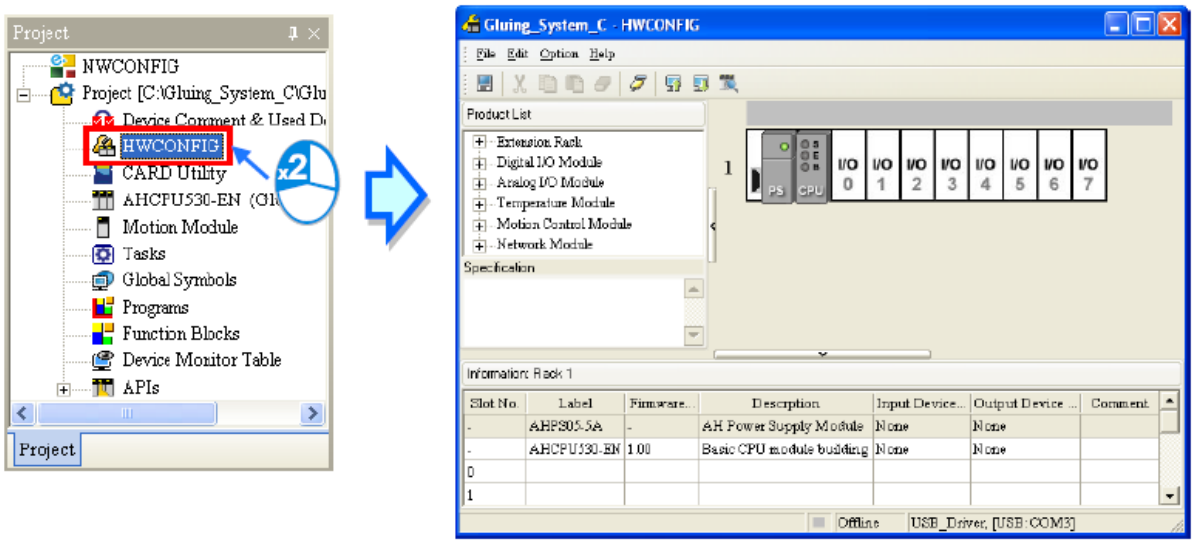
۴-۳-۱- ساخت پروژه

برای ساخت پروژه جدید در ISPSoft بر روی آیکون کلیک می‌کنیم. در پنجره Create a New Project نام پروژه و آدرس ذخیره سازی آن را به همراه مدل PLC مورد نظرمان (در این پروژه AHCPU530-EN) را مطابق شکل زیر تعیین می‌کنیم.



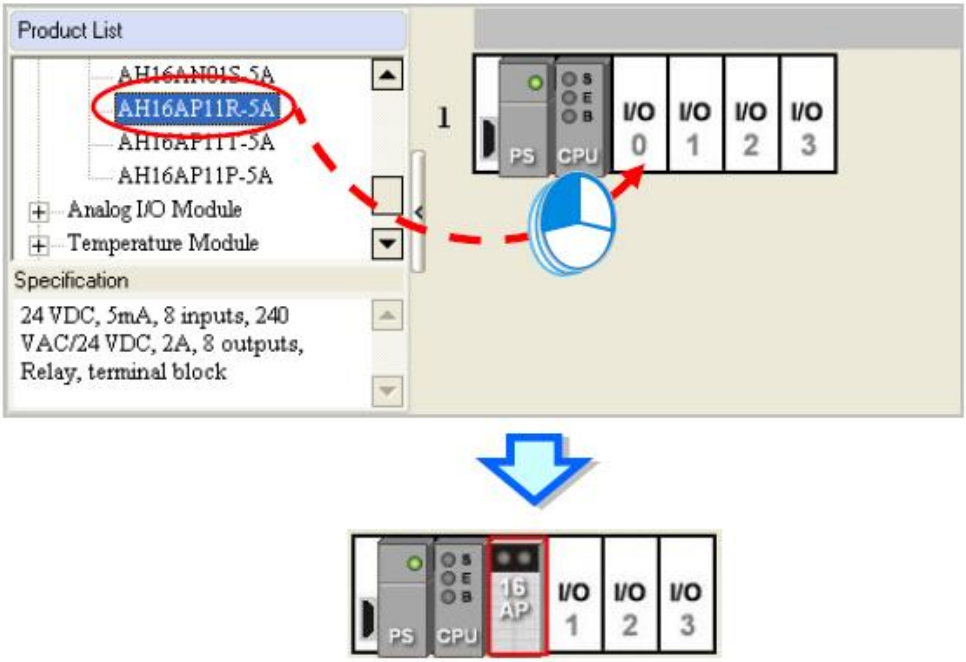
شکل ۴-۴ ساخت پروژه جدید – مثال خط تولید

پس از ایجاد پروژه با کلیک بر روی HWCONFIG در بخش مدیریت پروژه، صفحه تنظیمات سخت افزاری را باز می‌کنیم.



شکل ۴-۵ صفحه تنظیمات سخت افزاری – مثال خط تولید

در این قسمت باید از ورودی/خروجی دیجیتال مدل AH16AP11R-5A استفاده کنیم. آن را از قسمت Product List انتخاب کرده و در Slot خالی شماره صفر قرار می‌دهیم. پس از اضافه شدن این ماژول، اطلاعات مربوط به آن در جدول پایین صفحه اضافه می‌شود.

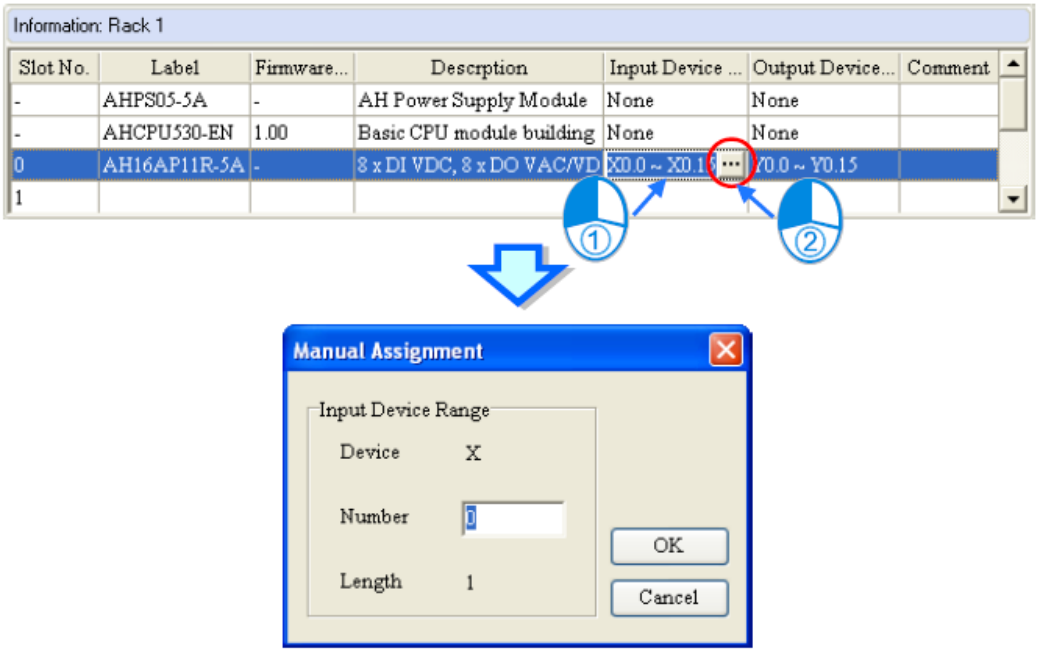


Information: Rack 1

Slot No.	Label	Firmware...	Description	Input Device ...	Output Device...	Comment
-	AHPS05-5A	-	AH Power Supply Module	None	None	
-	AHCPU530-EN	1.00	Basic CPU module building	None	None	
0	AH16AP11R-5A	-	8 x DI VDC, 8 x DO VAC/VD	X0.0 ~ X0.15	Y0.0 ~ Y0.15	
1						

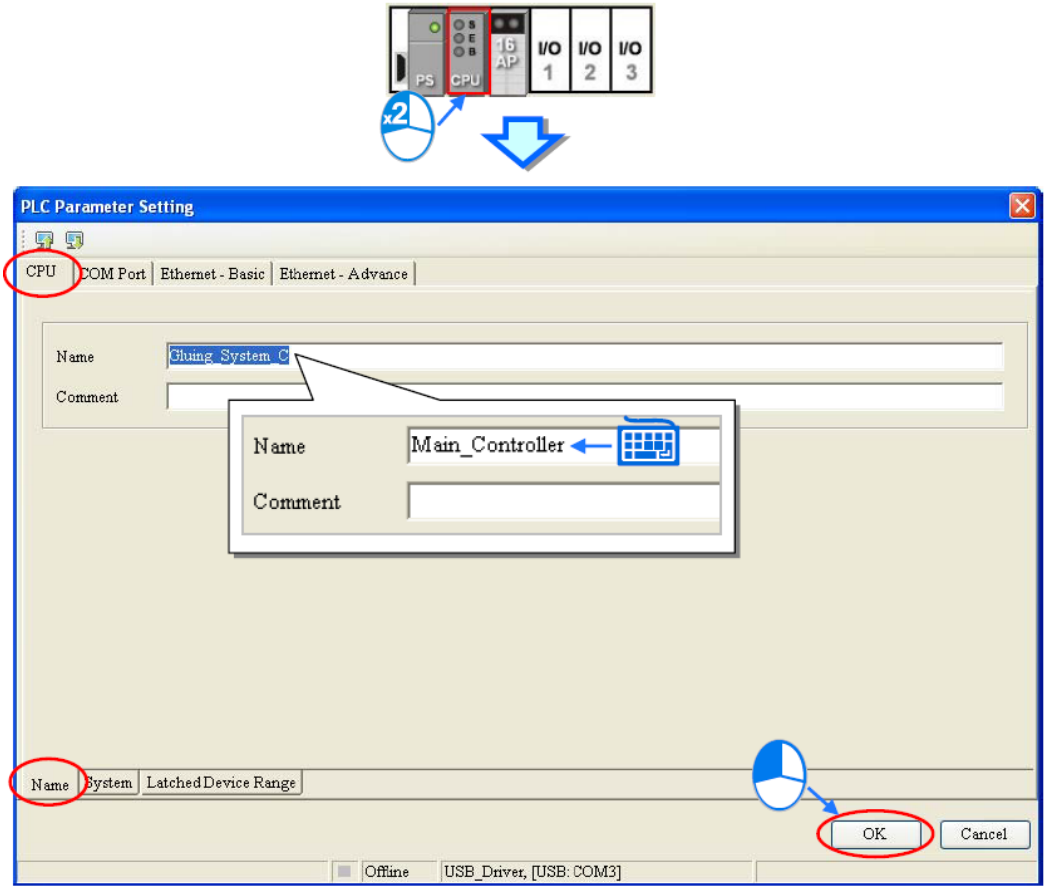
شکل ۴-۶ اضافه شدن ماژول ورودی/خروجی – مثال خط تولید

در این جدول همانطور که ملاحظه می‌شود، آدرس ورودی و خروجی به صورت خودکار تخصیص داده شده است. در صورتی که نیازی به تغییر آدرس‌های اختصاص داده شده باشد، می‌توانیم به صورت دستی با کلیک بر خانه‌های ستون‌های Input/Output Device Range نسبت به تغییر آن‌ها اقدام کنیم.



شکل ۴-۷ تغییر آدرس ورودی/خروجی – مثال خط تولید

برای تنظیم پارامترهای CPU می‌توانیم بر روی ماژول آن دوبار کلیک کرده و در صفحه PLC Parameter Setting به تنظیمات پارامترهای آن در سربرگ‌ها و زیربرگ‌ها مختلف بپردازیم.



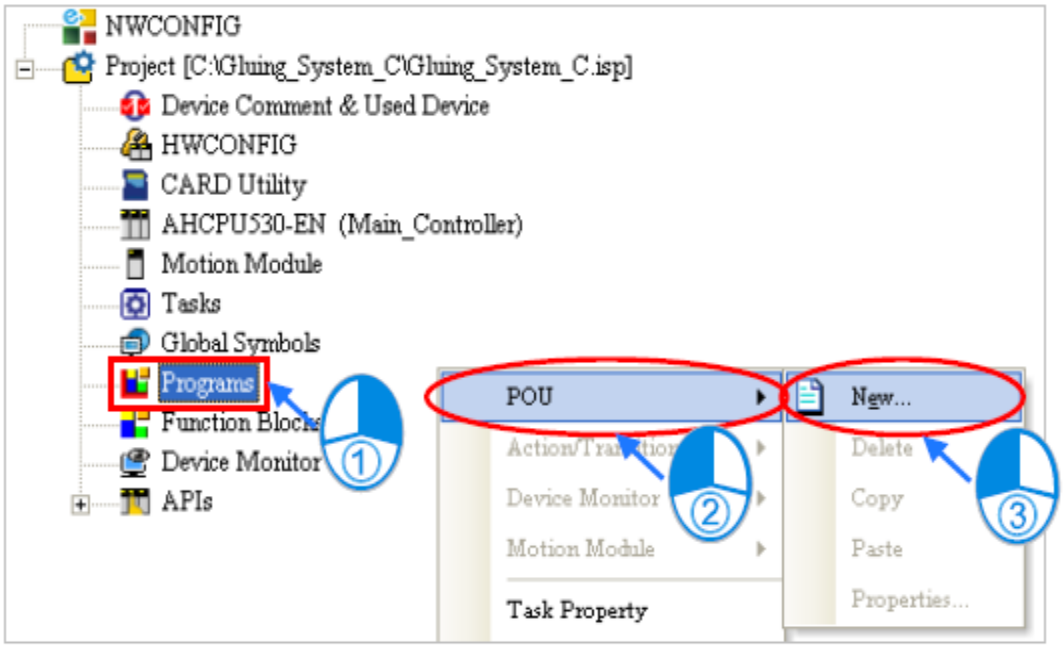
شکل ۴-۸تنظیم پارامترهای CPU – مثال خط تولید

در نهایت با کلیک بر روی گزینه اقدام به ذخیره سازی تغییرات سخت افزار می‌کنیم.

۴-۳-۲- برنامه نویسی

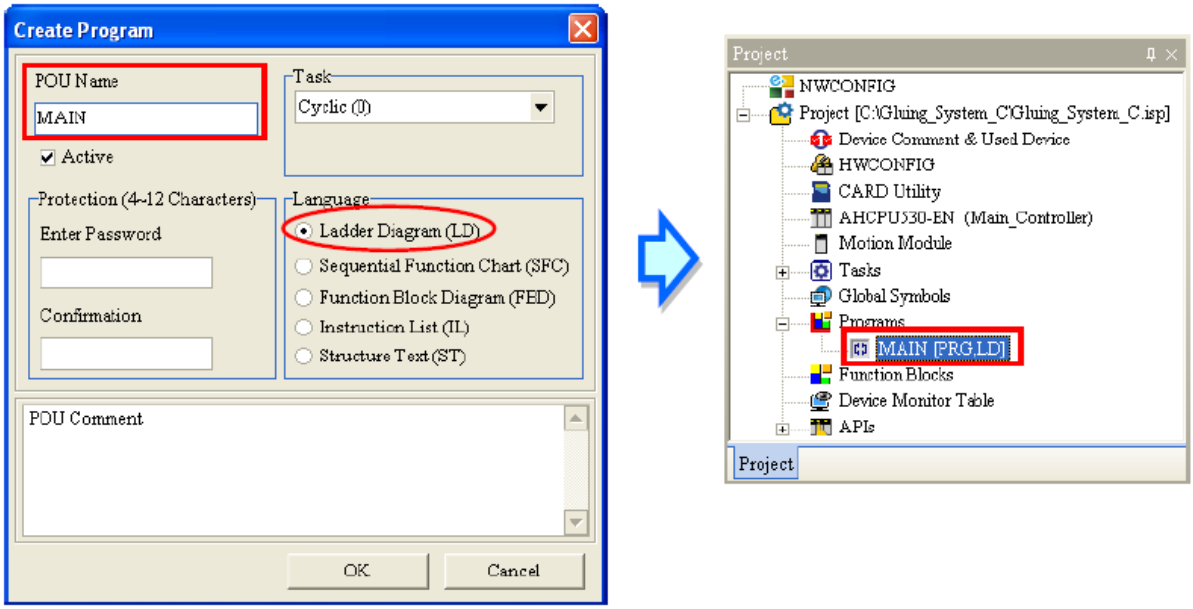
در این مرحله می‌خواهیم با استفاده از زبان برنامه نویسی ladder اقدام به نوشتن برنامه مورد نیاز خط تولید کنیم و سپس برنامه نوشته شده را کامپایل نماییم. در صورتی که آشنایی با زبان برنامه نویسی ladder ندارید می‌توانید ابتدا فصل ۸- را مطالعه نمایید، کار با این زبان برنامه نویسی بسیار راحت است و انتظار می‌رود (و البته نیاز است) کاربران در زمان بسیار کوتاهی بتوانند با آن آشنا شوند.

برای شروع برنامه نویسی نیاز به اضافه کردن محیط برنامه نویسی جدید است که برای اینکار در بخش مدیریت پروژه بر روی Program کلیک راست کرده و POU و سپس New را انتخاب می‌کنیم.



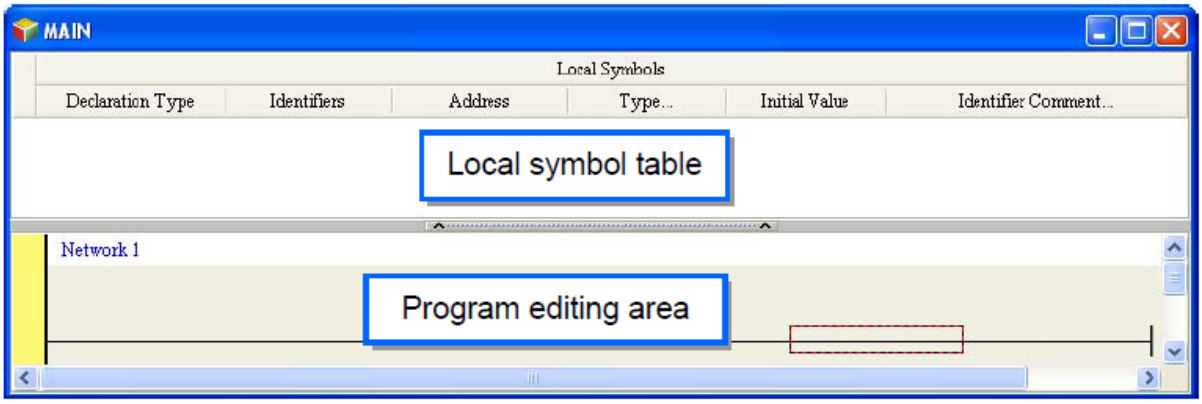
شکل ۴-۹ ایجاد برنامه جدید در پروژه – مثال خط تولید

در پنجره ظاهر شده پس از تعیین نام پروژه زبان برنامه نویسی را Ladder Diagram(LD) انتخاب می‌کنیم.



شکل ۴-۱۰ انتخاب زبان برنامه نویسی Ladder – مثال خط تولید

پس از تایید POU[1] مورد نظر ما برای نوشتن برنامه به عنوان زیرشاخه‌ای از Program در بخش مدیریت پروژه اضافه می‌شود. در این مرحله صفحه‌ای برای ویرایش برنامه باز خواهد شد.



شکل ۴-۱۱ محیط برنامه نویسی – مثال خط تولید

در این مرحله نوار ابزار مرتبط با زبان برنامه نویسی Ladder نیز ظاهر شده و می‌توانیم از آن در برنامه نویسی استفاده کنیم. آیکون های این نوار ابزار را در ادامه مرور خواهیم کرد.

شکل ۴-۱۲ نوار ابزار برنامه نویسی Ladder

جدول ۴-۲: آیکون‌های نوار ابزار برنامه نویسی Ladder

نماد	کلید میانبر	شرح
	Shift+Ctrl+C	نمایش/عدم نمایش توضیحات در networks[2]
		نمایش/ عدم نمایش توضیحات اجزای برنامه ladder
	Shift+Ctrl+A	فعال/غیر فعال سازی network انتخاب شده
	Shift+Ctrl+B	نشانه گذاری و یا پاک کردن نشانه‌ها در network-ها

[1] Program Organization Unit

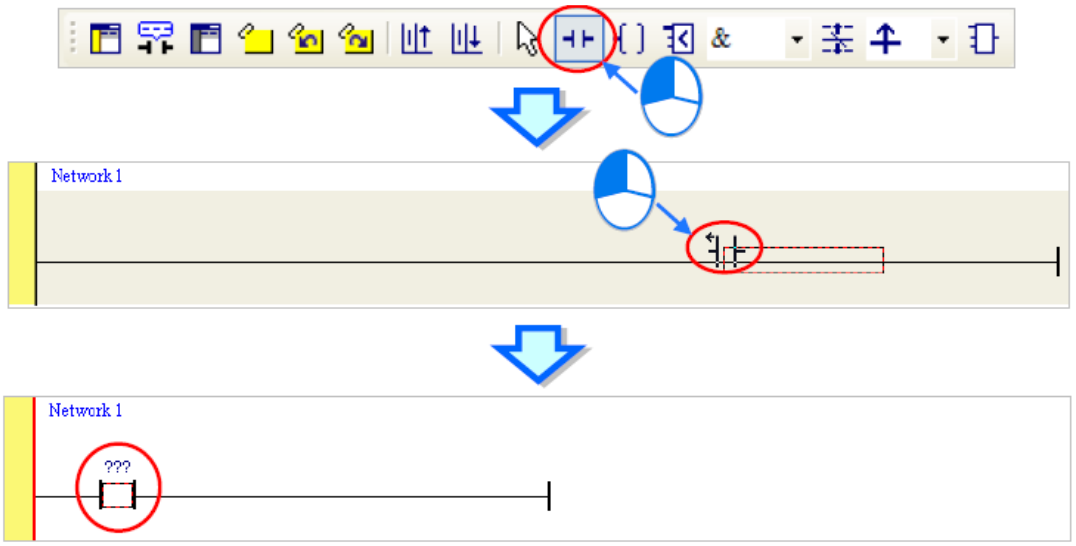
[2] منظور از شبکه در اینجا شبکه‌های زبان برنامه نویسی ladder است که برای جدا کردن بخش‌های مختلف برنامه مورد استفاده قرار می‌گیرد و نباید با شبکه‌های ارتباطی اشتباه شود.

	Shift+Ctrl+P	رفتن به نشان بعدی
	Shift+Ctrl+N	رفتن به نشان قبلی
	Ctrl+l	اضافه کردن network بالای network انتخاب شده
	Shift+Ctrl+l	اضافه کردن network پایین network انتخاب شده
	ESC	نمایشگر موس برای انتخاب اجزای برنامه
		اضافه کردن کانتکت[1]
		اضافه کردن کویل[2]
		اضافه کردن کانتکت مقایسه‌گر
		انتخاب نوع عملیات مقایسه‌ای
		اضافه کردن عملیات منطقی
		انتخاب نوع عملیات منطقی
	Shift+Ctrl+U	اضافه کردن بلوک (نوع آن را پس از کلیک کردن بر روی این آیکون باید تعیین کنید)

در ادامه به صورت مرحله به مرحله مروری خواهیم داشت بر چگونگی شروع کار با این المان‌های پایه در برنامه نویسی. ابتدا بر روی  کلیک کرده و سپس نمایشگر موس را به منطقه قرمز رنگ در network1 برده، در این مرحله مشاهده می‌کنید که شکل کانتکت در آن ناحیه ظاهر شده است. با کلیک در ناحیه قرمز مشاهده می‌کنید که کانتکت به Network1 اضافه شده است.

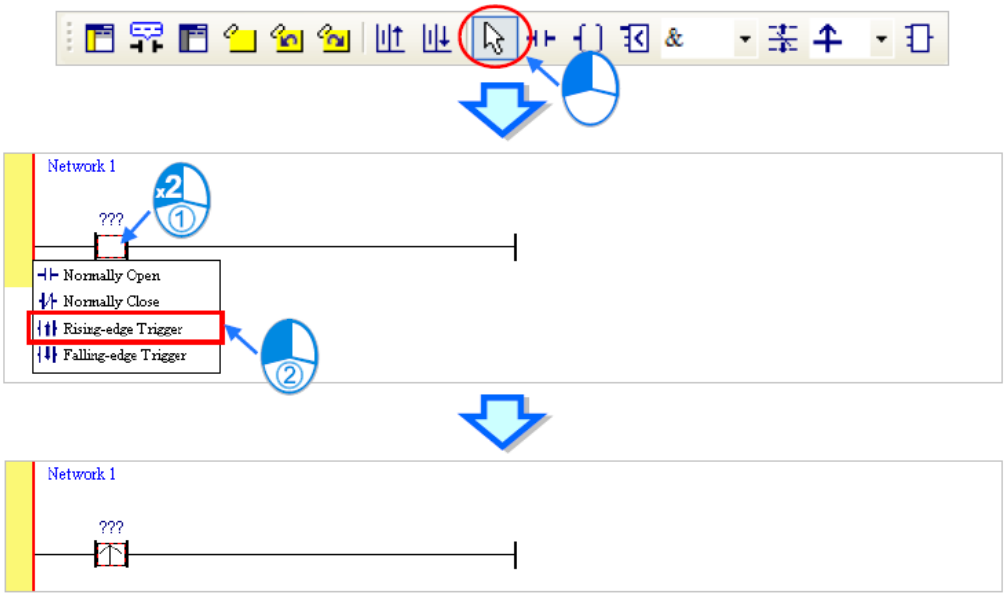
[1] Contact

[2] Coil



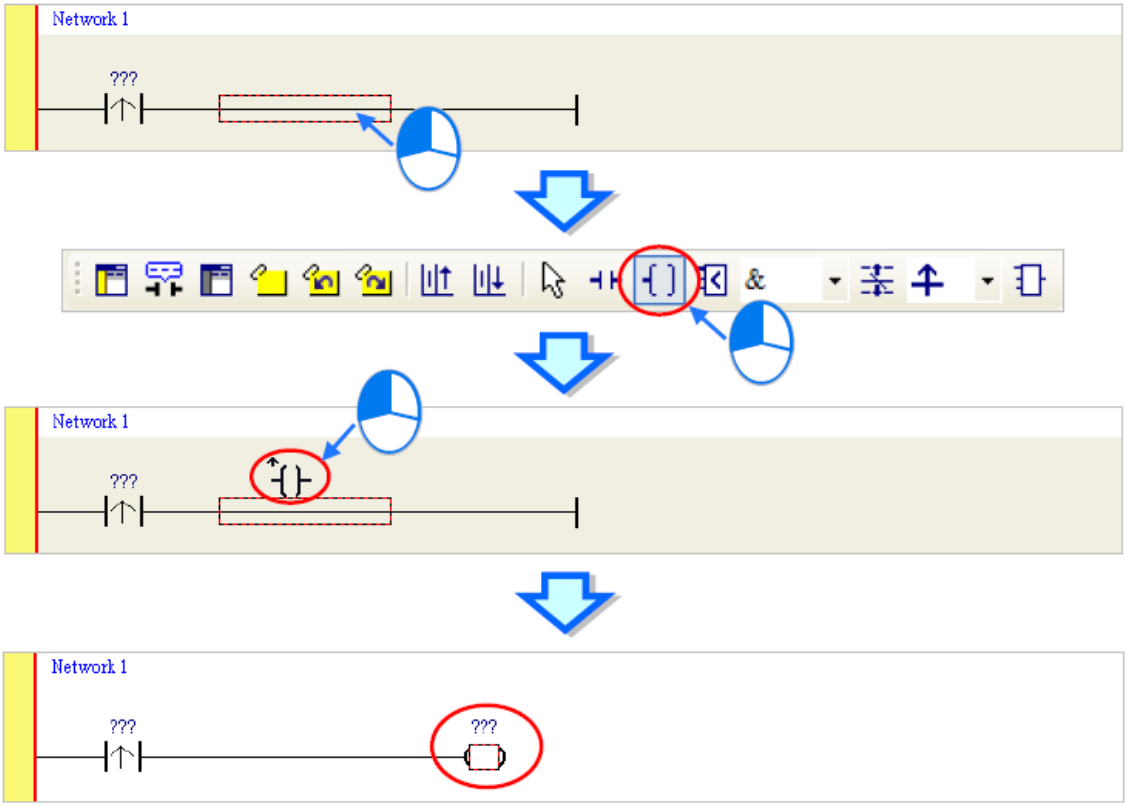
شکل ۴-۱۳ اضافه کردن کانتکت

سپس با انتخاب و دوبار کلیک کردن بر روی کانتکت اضافه شده، لیستی باز خواهد شد که در آن می‌توانید نوع کانتکت خود را انتخاب کنید. در این مثال کانتکت تشخیص دهنده لبه بالارونده را انتخاب می‌کنیم.



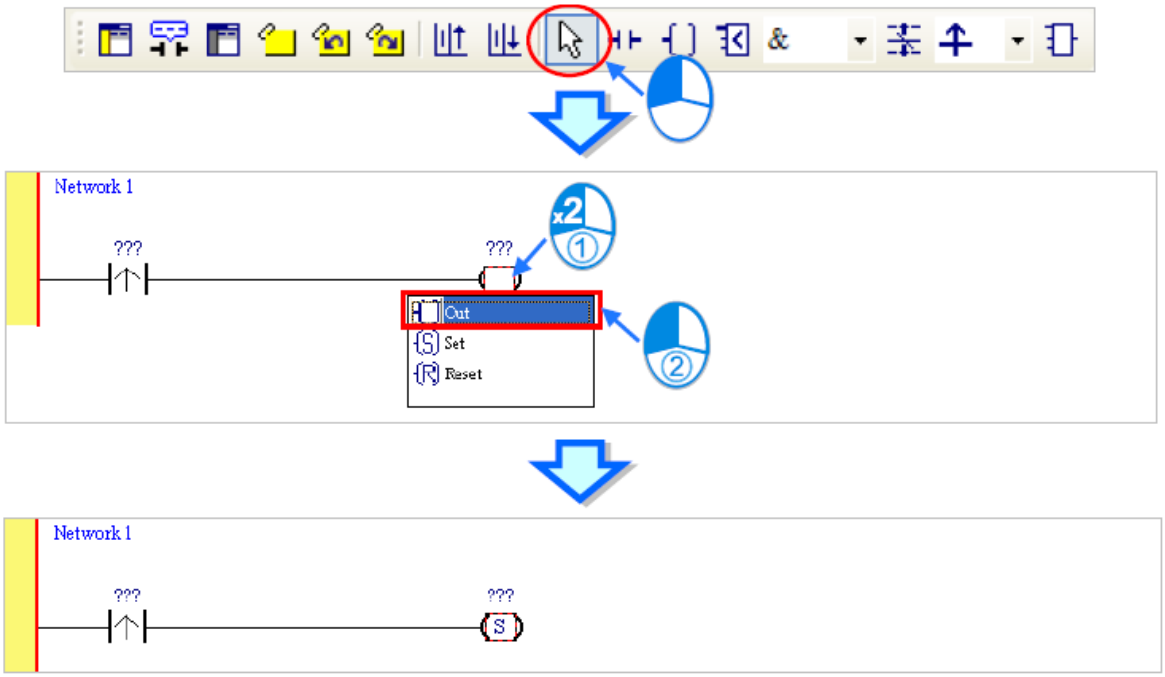
شکل ۴-۱۴ انتخاب نوع کانتکت

پس از کلیک بر روی سمت راست کانتکت در network1، کادر قرمز دوباره ظاهر می‌شود. در ادامه بر روی کلیک کرده و آن را در کادر قرمز قرار می‌دهیم.



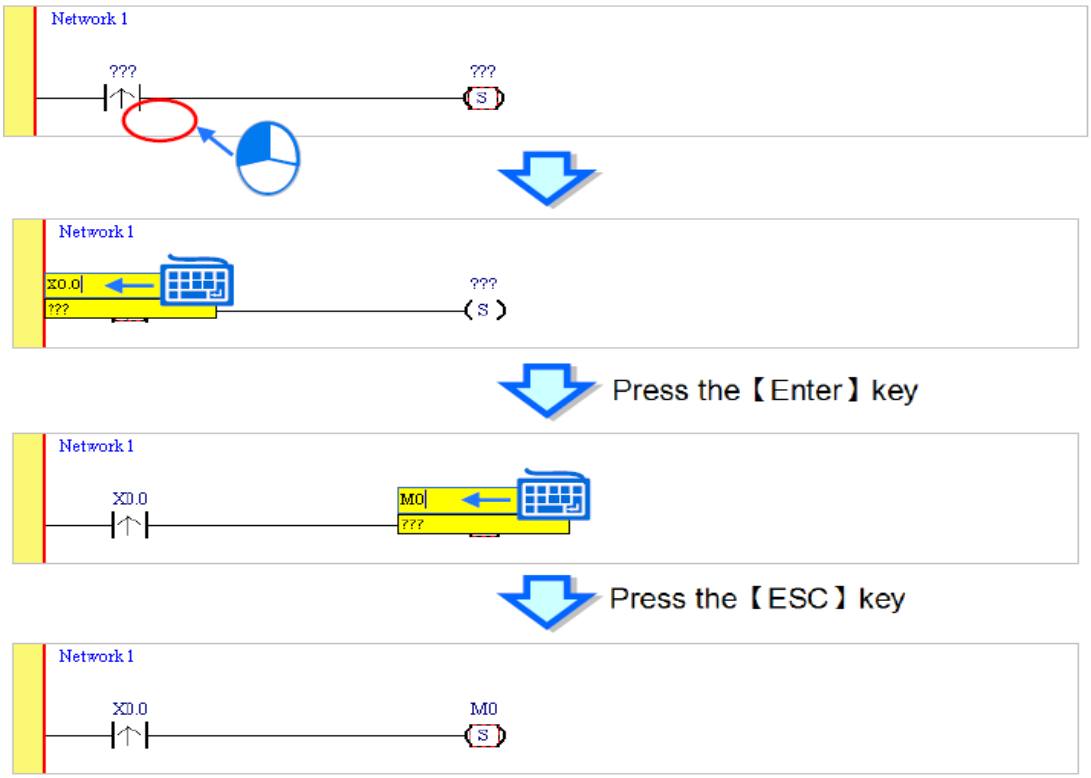
شکل ۴-۱۵ اضافه کردن کویل

همانند قبل نوع کویل خروجی را انتخاب می‌کنیم. آن را به صورت Set برای حافظه‌ای در نظر می-گیریم.



شکل ۴-۱۶ انتخاب نوع کویل

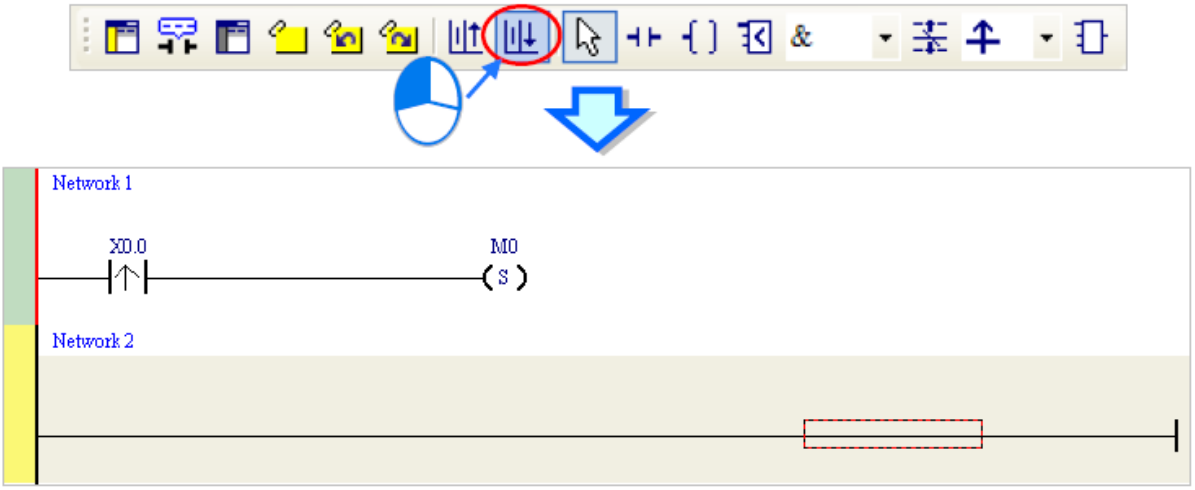
با کلیک بر روی "???" ظاهر شده بر روی کانتکت و کویل خروجی می‌توانیم آدرس حافظه یا پورت مرتبط با آن را مشخص کنیم. در این مثال این مقدار کانتکت را متناظر با ورودی X0.0 و مقدار کویل را متناظر با حافظه M0[1] قرار می‌دهیم قرار می‌دهیم.



شکل ۴-۱۷ آدرس دهی کانتکت و کویل

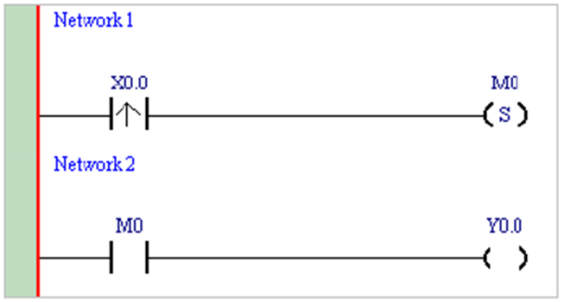
در این مرحله نیاز است برای ادامه برنامه نویسی network جدیدی در زیر network فعلی اضافه شود. برای اینکار بر روی کلیک می‌کنیم.

[1] در مورد آدرس دهی حافظه در آینده بیشتر صحبت خواهیم کرد.



شکل ۴-۱۸ اضافه شدن Network جدید

در خط network جدید نیز یک کانتکت باز و یک کویل خروجی قرار می‌دهیم، کانتکت ورودی در این قسمت توسط حافظه M0 کنترل و وضعیت کویل خروجی Y0.0 را کنترل می‌کند.

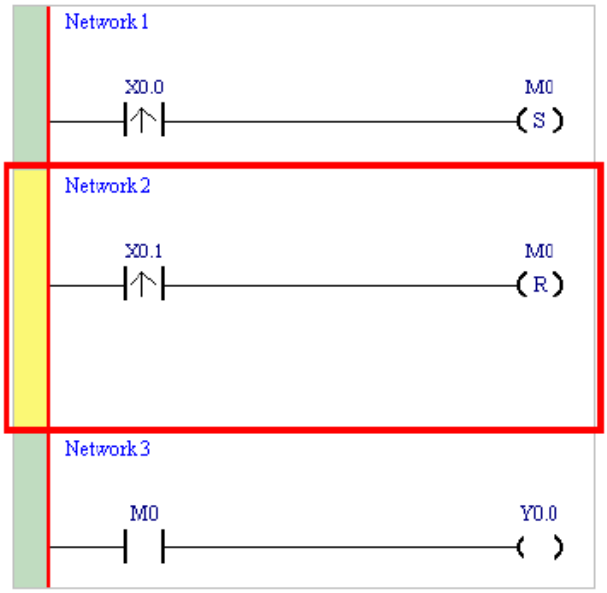


شکل ۴-۱۹ برنامه مثال خط تولید – مرحله ۱

بررسی برنامه نوشته شده تا این مرحله: در این حالت به ازای فشرده شدن کلید START لبه بالا رونده آن در ورودی X0.0 توسط کنتاکت network1 تشخیص داده شده و کویل Set و در نتیجه آن حافظه M0 را یک می‌کند. حافظه M0 در واقع نقش پرچم وضعیت سیستم را دارد که پس فشرده شدن کلید START به وضعیت فعال در می‌آید (و دیگر وضعیت آن به وضعیت کلید START وابسته نخواهد بود و فقط لحظه اول فشرده شدن این کلید، این حالت را فعال خواهد کرد). سپس وضعیت پرچم در خط network دوم موجب فعال شدن کویل خروجی Y0.0 می‌شود که این خروجی فعال بودن یا نبودن تسمه نقاله را مشخص می‌کند. یعنی اگر پرچم وضعیت فعال باشد تسمه نقاله فعال و در غیر این صورت غیرفعال است.

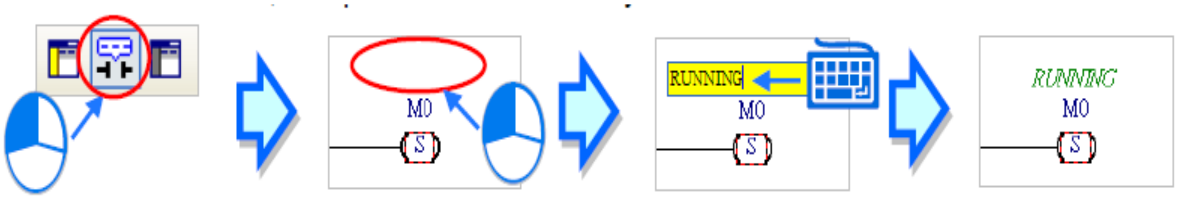
در ادامه باید با لحاظ کردن اثر کلید STOP شرایطی را فراهم کنیم که در صورت فشرده شدن کلید STOP بتوان پرچم وضعیت سیستم را به حالت غیر فعال تغییر دهیم. برای این کار خط network

جدیدی را بدین مضمون ایجاد می‌کنیم که در آن با تشخیص لبه ورودی STOP در X0.1، M0 را ریست کند.

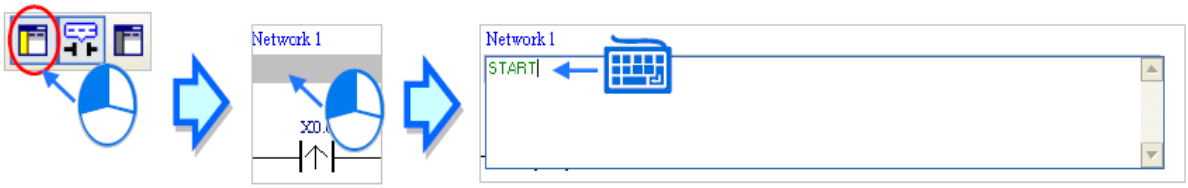


شکل ۴-۲۰ برنامه مثال خط تولید – مرحله ۲

نکته‌ای که برای حرفه‌ای شدن در برنامه نویسی همیشه باید لحاظ کنید استفاده به جا از توضیحات است، برای اضافه کردن توضیحات بر روی المان‌ها می‌توانید از و برای اضافه کردن توضیحات در خطوط network-ها می‌توانید از استفاده کنید.



شکل ۴-۲۱ اضافه کردن توضیحات المان ها

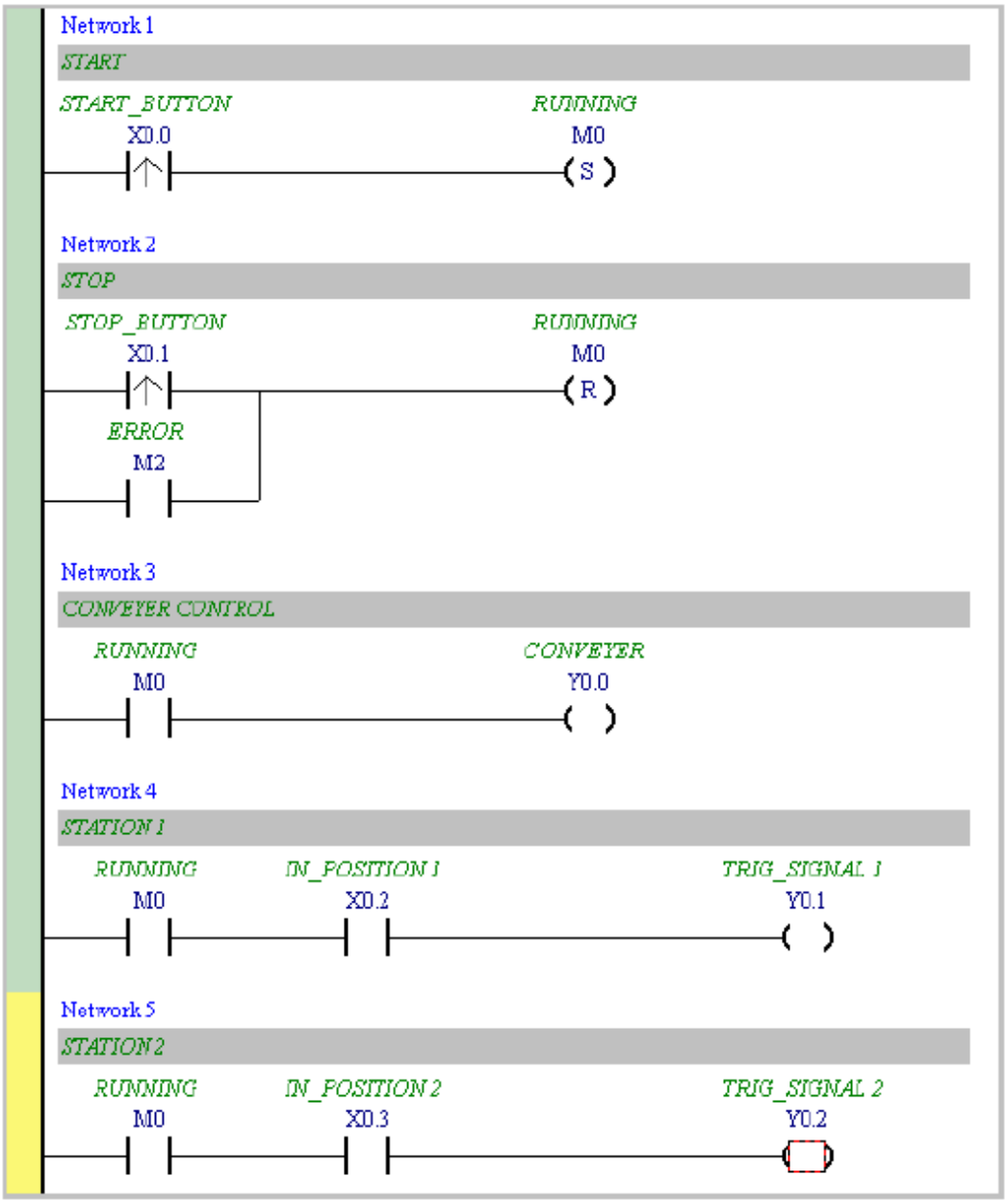


شکل ۴-۲۲ اضافه کردن توضیحات Network

برنامه خود را به صورت زیر کامل می‌کنیم. با توجه به توضیحات اضافه شده در برنامه شرح و البته درک آن بسیار راحت تر شده است. البته ذکر این نکته ضروری است که اضافه کردن توضیحات در

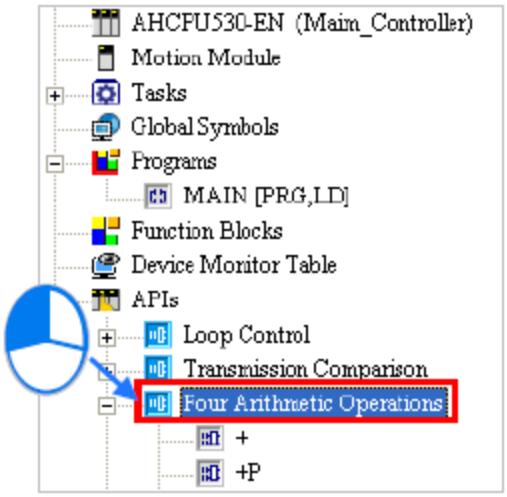
برنامه‌های پیچیده بسیار راه‌گشا است و این موضوع نیز بستگی به توانایی برنامه نویس در توضیح نویسی موثر دارد. تسلط در توضیح نویسی محقق نخواهد شد مگر با افزایش تجربه وتلاش برای بهینه نویسی.

همانطور که در برنامه زیر مشخص است، در خط network2 ایجاد خطا موجب توقف کار سیستم خواهد شد و در خطوط network4&5 نیز در صورتی که سنسور موقعیت، وجود قطعه را تشخیص دهد در حالی که وضعیت سیستم فعال باشد، تزریق کننده مربوط به آن نیز فعال می‌شود.

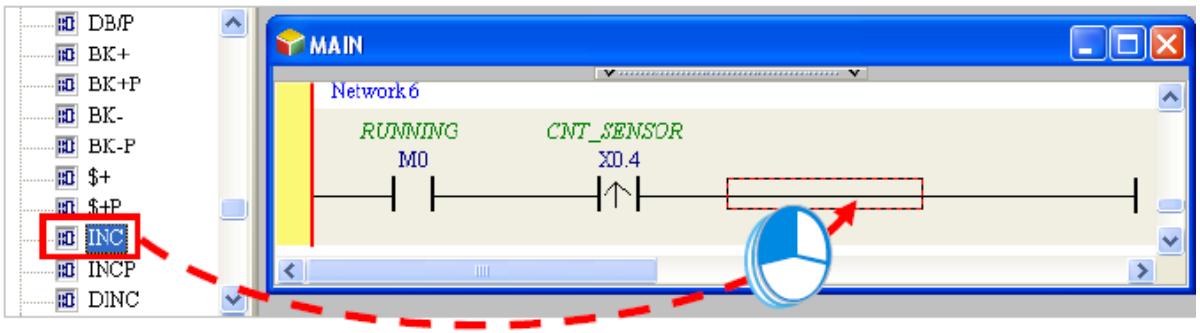


شکل ۴-۲۳ برنامه مثال خط تولید – مرحله ۳

در ادامه می‌خواهیم از جمع کننده‌ای استفاده کنیم که بتواند تعداد قطعات عبوری روبروی سنسور شمارنده متناظر با ورودی X0.4 را بشمارد. برای این کار نیاز به بلوک جمع کننده‌ای مانند INC داریم که می‌توانیم آن را از کتابخانه APIs واقع در قسمت مدیریت پروژه استخراج کنیم. سپس با کشیدن بلوک مورد نظرمان آن را به پروژه اضافه می‌کنیم.

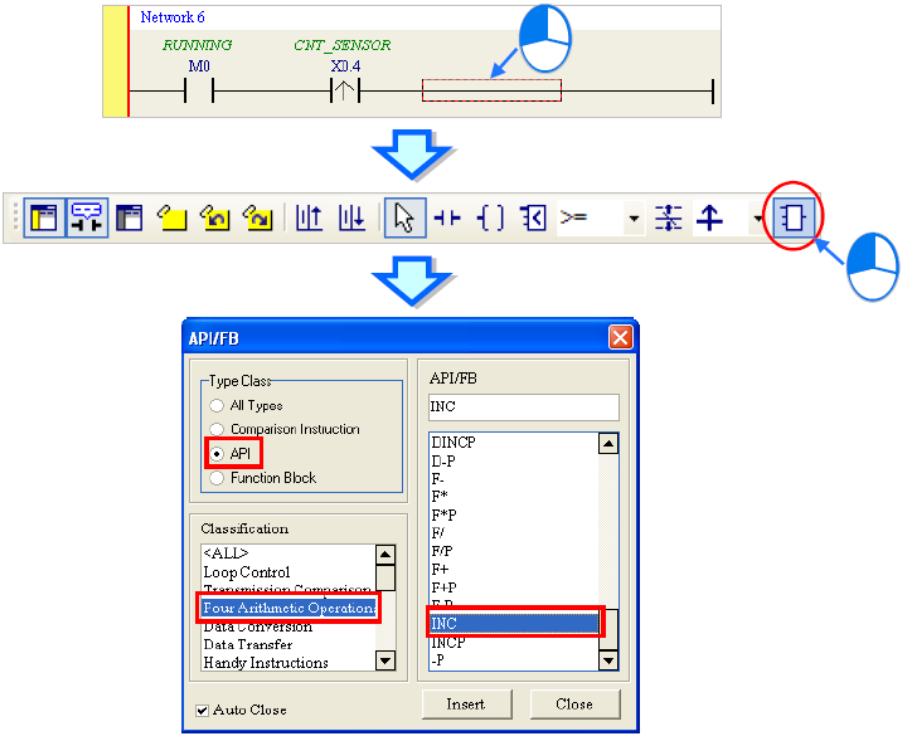


شکل ۴-۲۴ :کتابخانه APIs



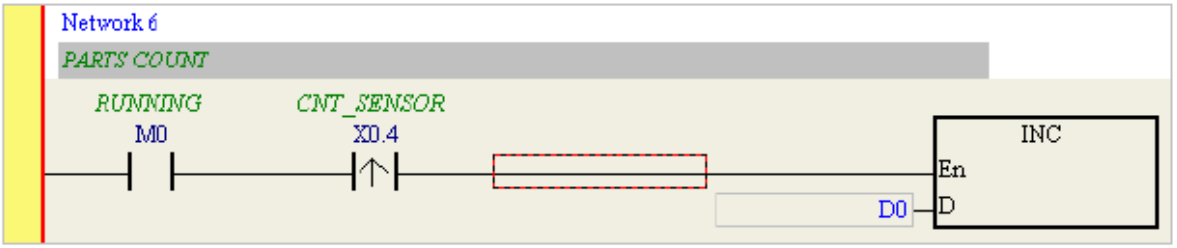
شکل ۴-۲۵ اضافه کردن بلوک از کتابخانه به برنامه

البته این کار را می‌توانیم با استفاده از آیکون نیز انجام دهیم:



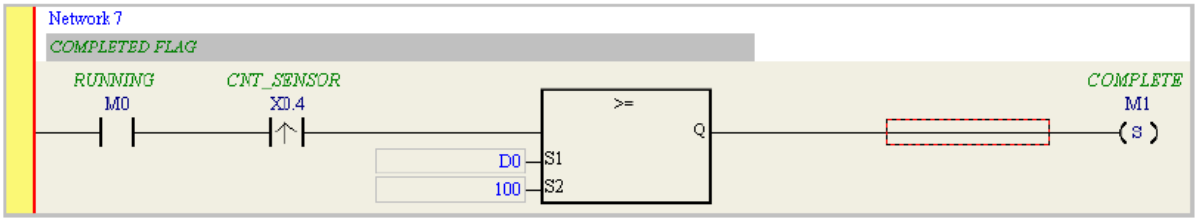
شکل ۴-۲۶ اضافه کردن بلوک – روش دوم

سپس مقدار شمارنده را در حافظه D0 ذخیره می‌کنیم.



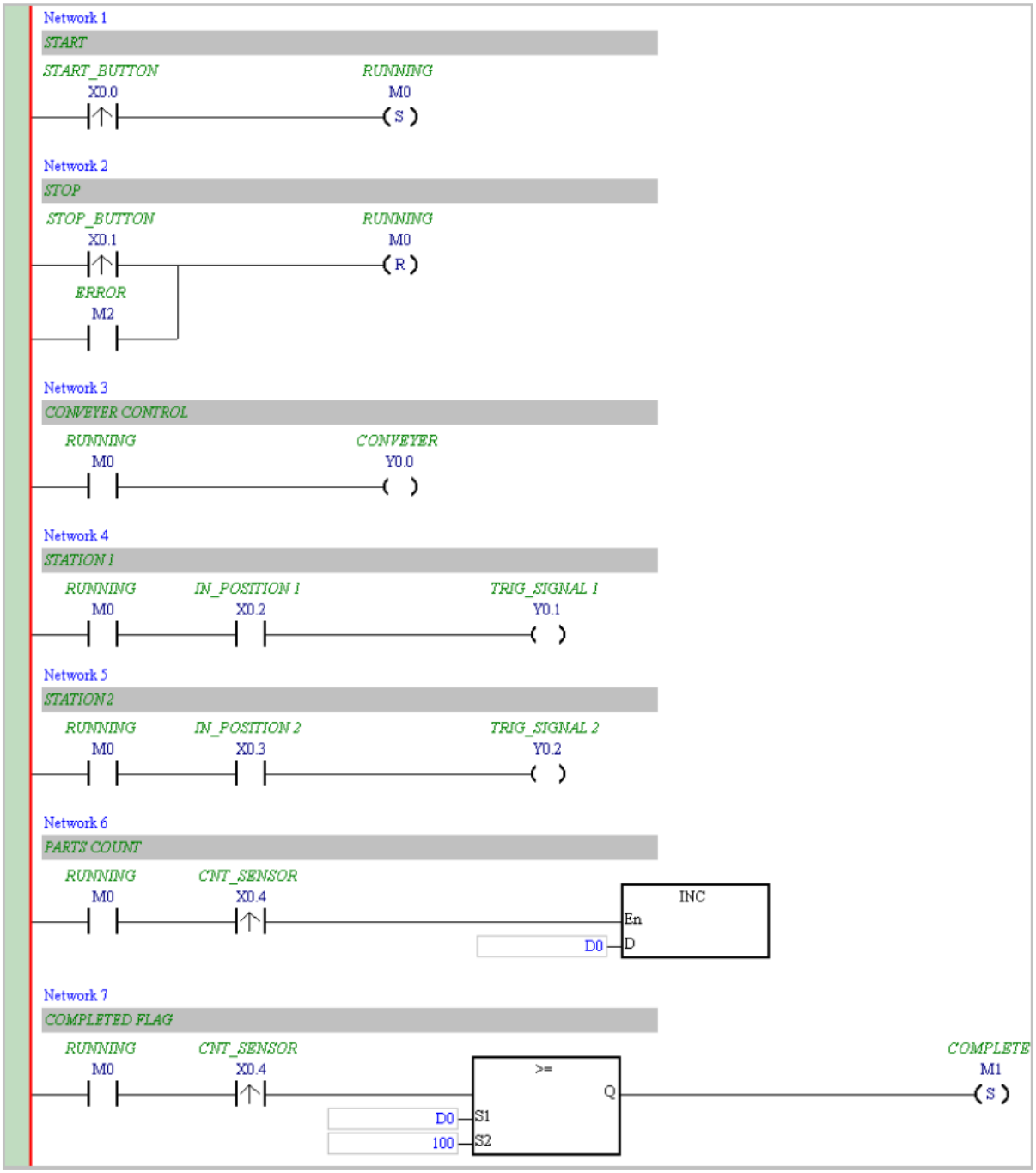
شکل ۴-۲۷ برنامه مثال خط تولید – مرحله ۴

حال با استفاده از APIs یا آیکون در حالی که به صورت بزرگتر مساوی تنظیم شده است، می‌توانیم مقایسه کنیم که آیا تعداد شمارش شده به ۱۰۰ رسیده است؟ اگر این اتفاق بیافتد پرچم "تکمیل" که همان حافظه M1 است فعال می‌شود.



شکل ۴-۲۸ برنامه مثال خط تولید – مرحله ۵

در نهایت برنامه نوشته شده به صورت زیر در آمده است.

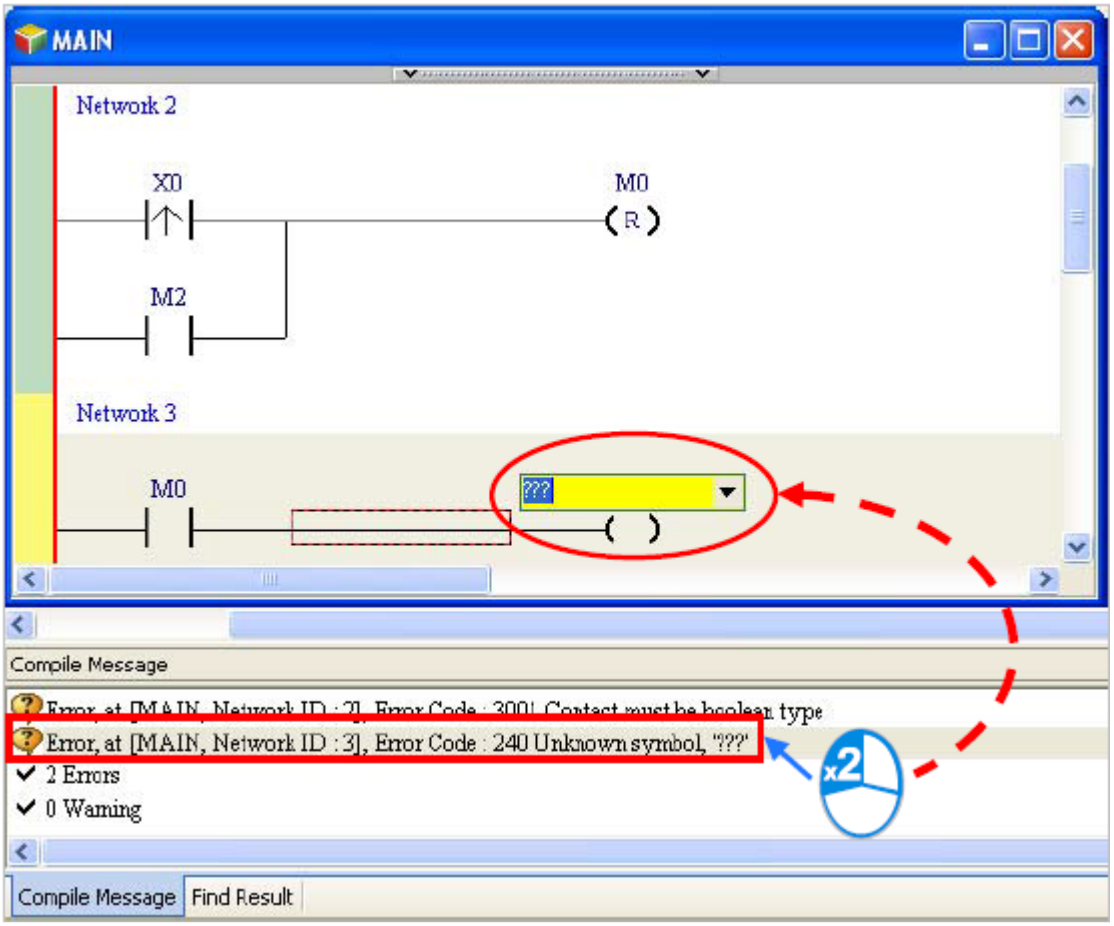


شکل ۴-۲۹ برنامه نهایی مثال خط تولید

۴-۳-۳- بررسی و کامپایل برنامه نوشته شده

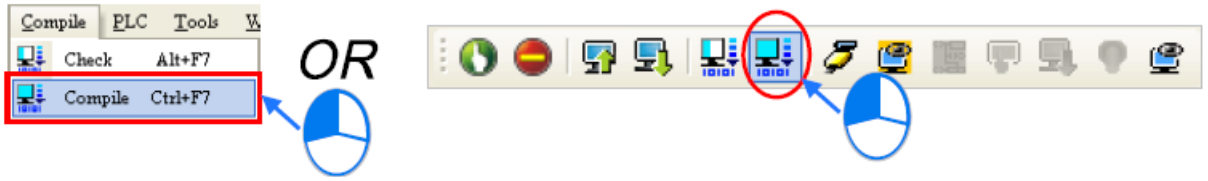
پس از نوشته شدن برنامه، می‌توان درستی آن را بررسی و سپس کامپایل کرد. برای بررسی برنامه می‌توان از منوی Compile گزینه Check را انتخاب و یا بر روی آیکون کلیک کرد. سپس در صورتی

که خطایی حین بررسی محتوای برنامه وجود داشته باشد، این خطا در بخش پیام لیست می‌شود و می-توان با دوبار کلیک بر روی آن موقعیت دقیق خطا را پیدا کرد.



شکل ۴-۳۰ بررسی برنامه نوشته شده

برای کامپایل کردن[1] پروژه نیز می‌توانیم از منوی Compile گزینه Compile را انتخاب و یا از  استفاده کنیم.

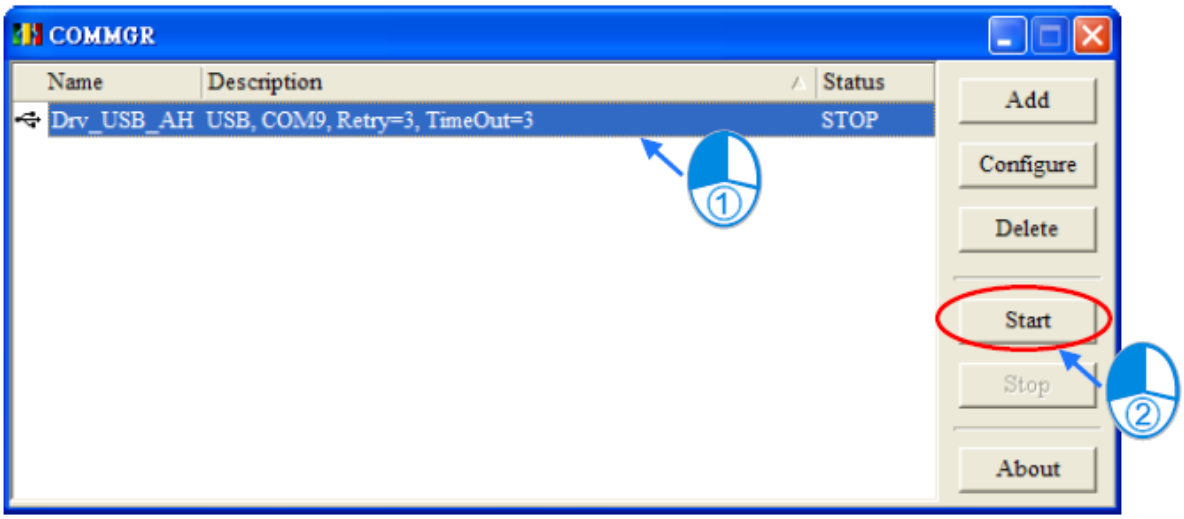


شکل ۴-۳۱ کامپایل کردن

[1] کامپایل کردن یعنی کد برنامه به کد زبان ماشین تبدیل شود، برای آنکه بتوان از آن برای برنامه ریزی و ارتباط با سخت افزار (ماشین) استفاده کرد.

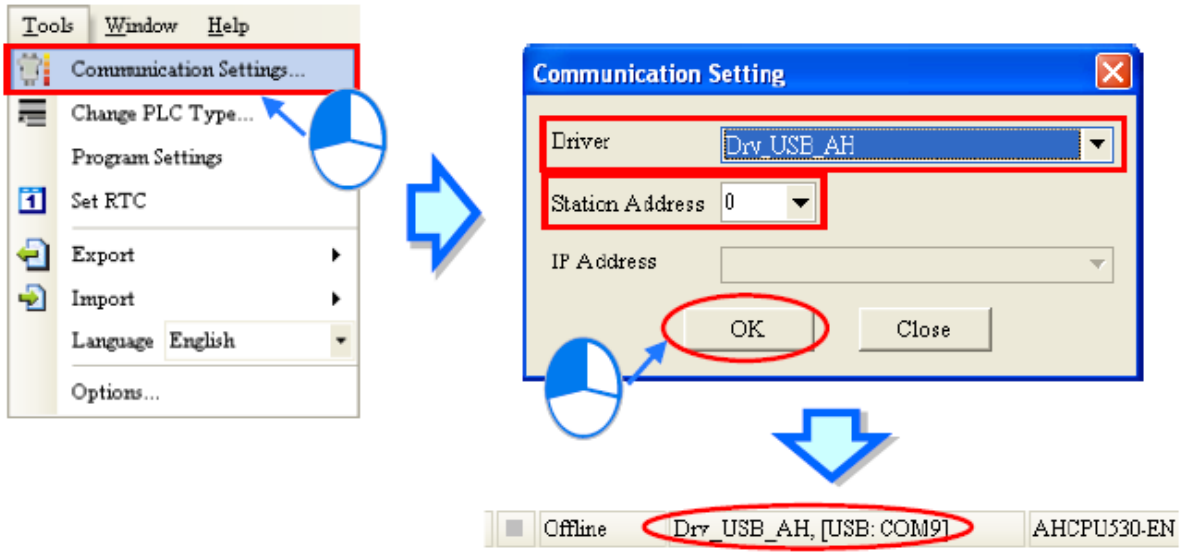
۴-۳-۴- برقراری ارتباط با PLC و بارگزاری نرم افزار

برای اینکار لازم است درایور ساخته شده در COMMGR را فعال کنیم.



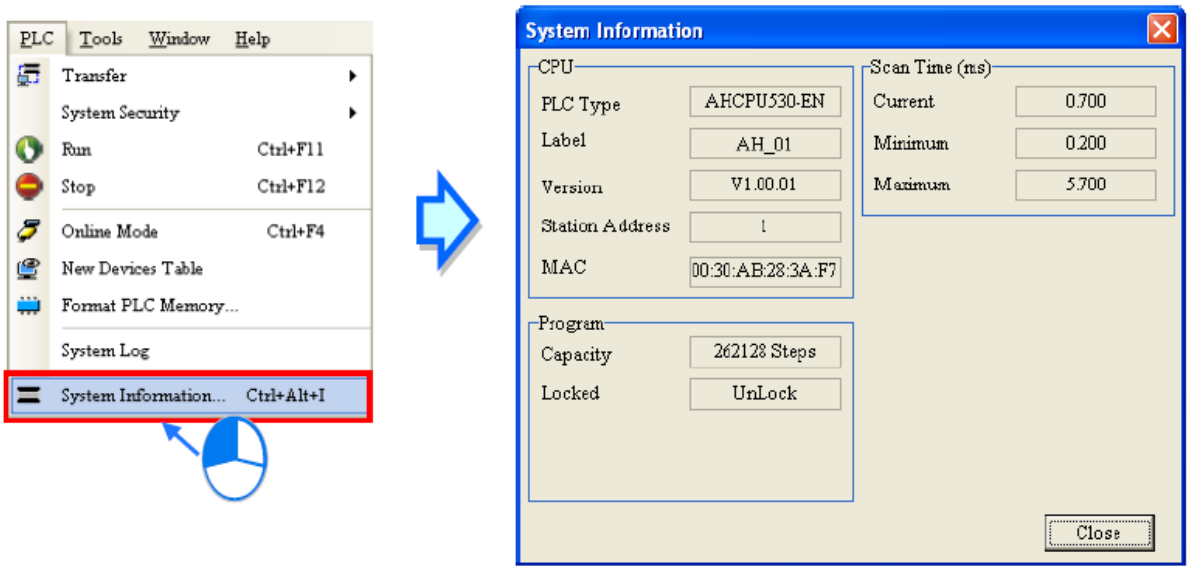
شکل ۴-۳۲ فعال کردن درایور در COMMGR- مثال خط تولید

سپس با کلیک بر روی Communication Settings در منوی Tools درایور مورد نظر را به نرم افزار ISPSoft متصل می‌کنیم. پس از انجام این مرحله می‌توانیم نمایش اتصال درایور مورد نظر در نوار وضعیت را مشاهده کنیم.



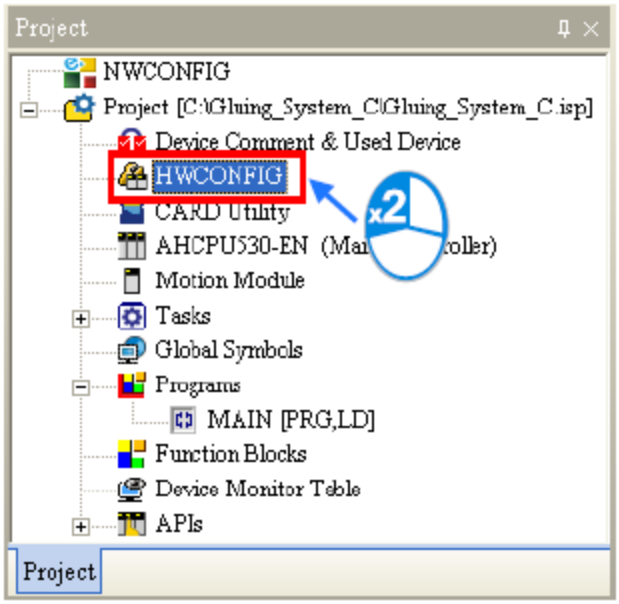
شکل ۴-۳۳ برقراری ارتباط نرم افزار با درایور - مثال خط تولید

برای بررسی وضعیت اتصال کامپیوتر با PLC می‌توانیم از منوی PLC بر روی System Information کلیک کنیم. در صورتی که اتصال با PLC برقرار باشد، صفحه‌ای مشابه صفحه زیر را می‌توان مشاهده کرد.

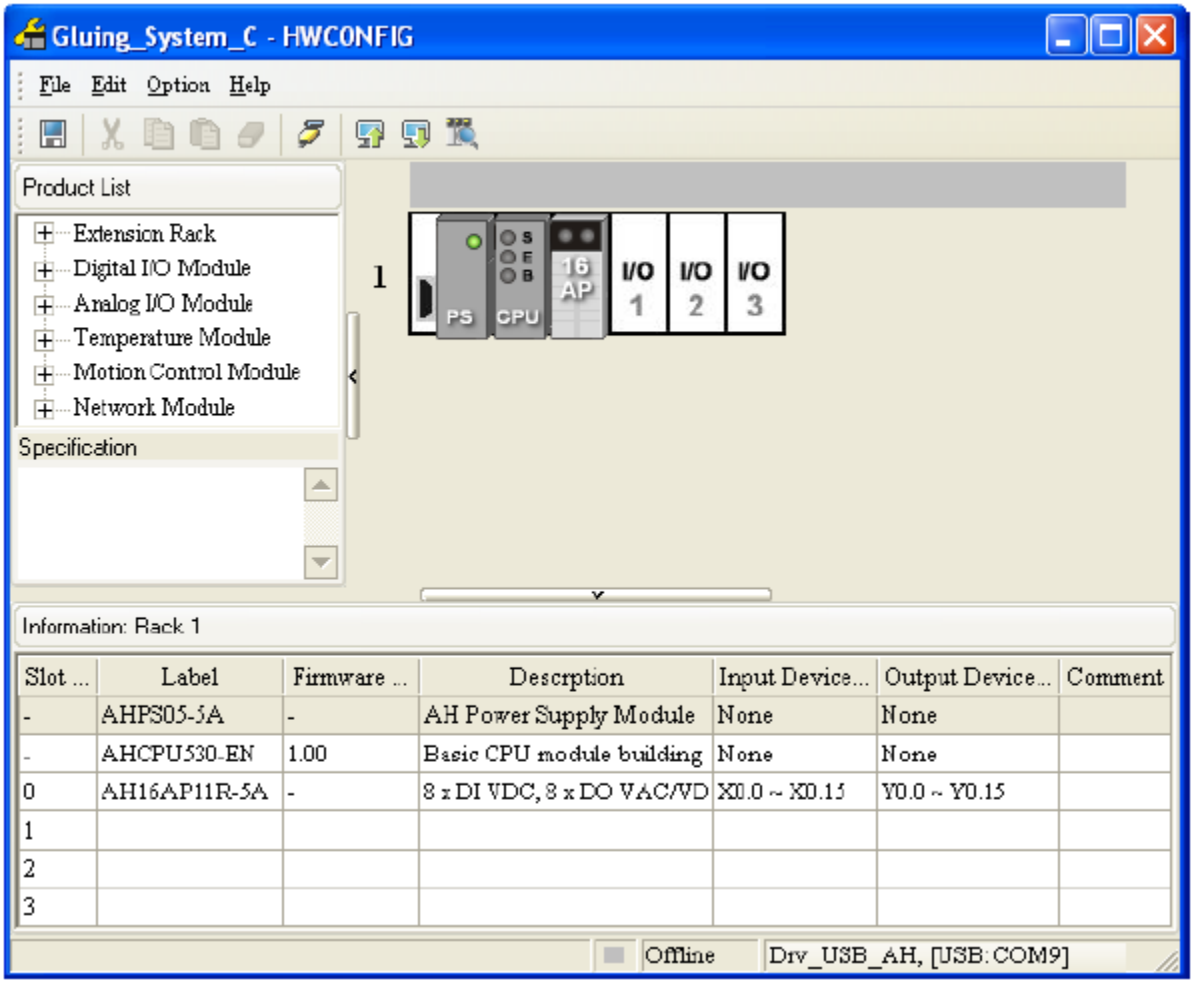


شکل ۴-۳۴ بررسی اتصال نرم افزار به PLC – مثال خط تولید

پس از اینکه از اتصال نرم افزار با PLC مطمئن شدیم، می‌توانیم تنظیمات سخت افزاری و برنامه نوشته شده را بر روی آن بارگزاری نماییم. برای بارگزاری تنظیمات سخت افزاری می‌توانیم بر روی HWCONFIG دوبار کلیک کرده تا صفحه تنظیمات سخت افزاری باز شود.

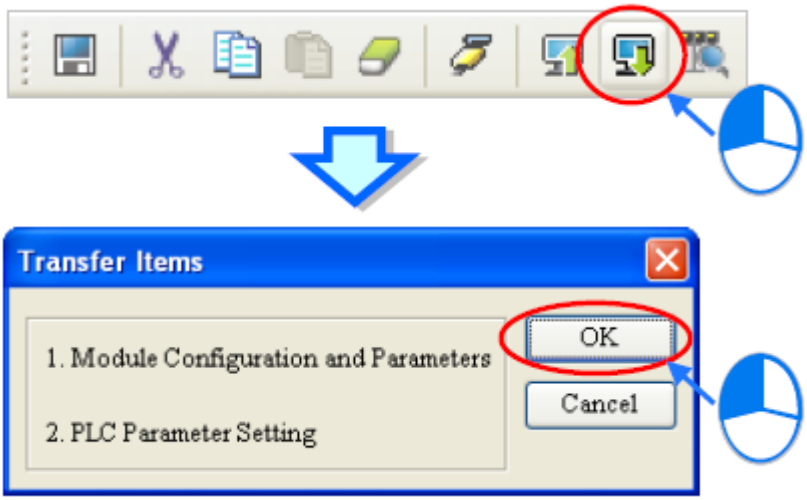


شکل ۴-۳۵ اجرای HWCONFIG– مثال خط تولید



شکل ۴-۳۶ تنظیمات سخت افزاری – مثال خط تولید

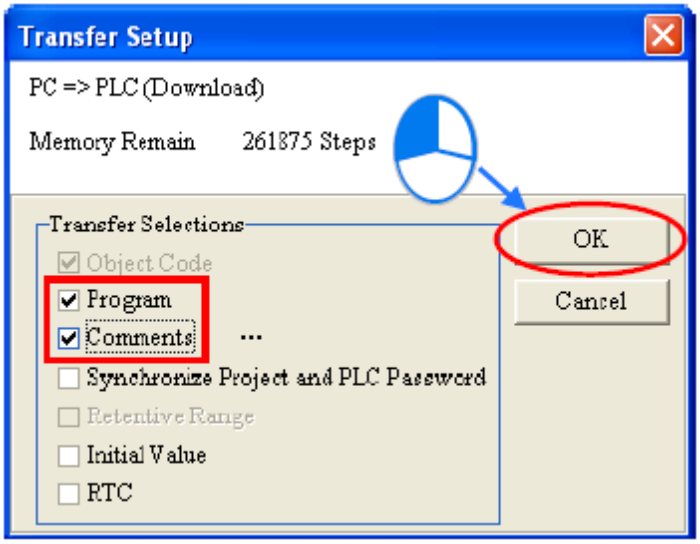
در این مرحله با کلیک بر روی  می‌توان تنظیمات سخت افزاری را در PLC بارگزاری کرد.



شکل ۴-۳۷ بارگزاری تنظیمات سخت افزاری – مثال خط تولید

پس از دانلود تنظیمات سخت افزاری، LED خطای BUS بر روی CPU خاموش خواهد شد. در صورتی که این LED خاموش نشود، نشانگر آن است که خطایی در تنظیمات سخت افزاری رخ داده است.

در این مرحله برای بارگزاری برنامه نوشته شده در صفحه ISPSoft برروی  کلیک کنیم. در صفحه ظاهر شده گزینه Program و Comments را انتخاب و سپس برای دانلود بر روی OK کلیک می‌کنیم.



شکل ۴-۳۸ دانلود برنامه نوشته شده – مثال خط تولید

۴-۳-۵- بررسی برقراری ارتباط

برای مانیتور کردن اجرای برنامه بارگزاری شده در PLC در نرم افزار ISPSoft دو راه وجود دارد. اولین راه، حالت مانیتورینگ پورت‌ها و حافظه‌هاست و دیگری مانیتورینگ برنامه است.

جدول ۴-۳: روش های مانیتورینگ

حالت مانیتورینگ	شرح
 مانیتورینگ پورت-ها و حافظه‌ها	با استفاده از جدول مانیتورینگ می‌توان وضعیت حافظه‌ها و پورت‌ها را در PLC مانیتور کرد. در این حالت ISPSoft فقط نیاز به بروزرسانی وضعیت این حافظه‌ها و پورت‌ها دارد. در این روش لازم نیست برنامه نوشته شده در نرم افزار و PLC به یک شکل باشند.
 مانیتورینگ برنامه	در صفحه ویرایش برنامه، با استفاده از این حالت می‌توان وضعیت کاری برنامه نوشته شده در تمامی خطوط Network ها را در لحظه

	مشاهده کرد. در این حالت به علت آنکه برنامه نوشته شده در ISPSoft ملاک بررسی است، برنامه نرم افزار و PLC باید به یک صورت باشند.

پس از کلیک بر روی آیکون  و تغییر وضعیت سیستم به حالت آنلاین، سیستم دو حالت مانیتورینگ برنامه و حافظه/ پورت را فعال می‌کند.

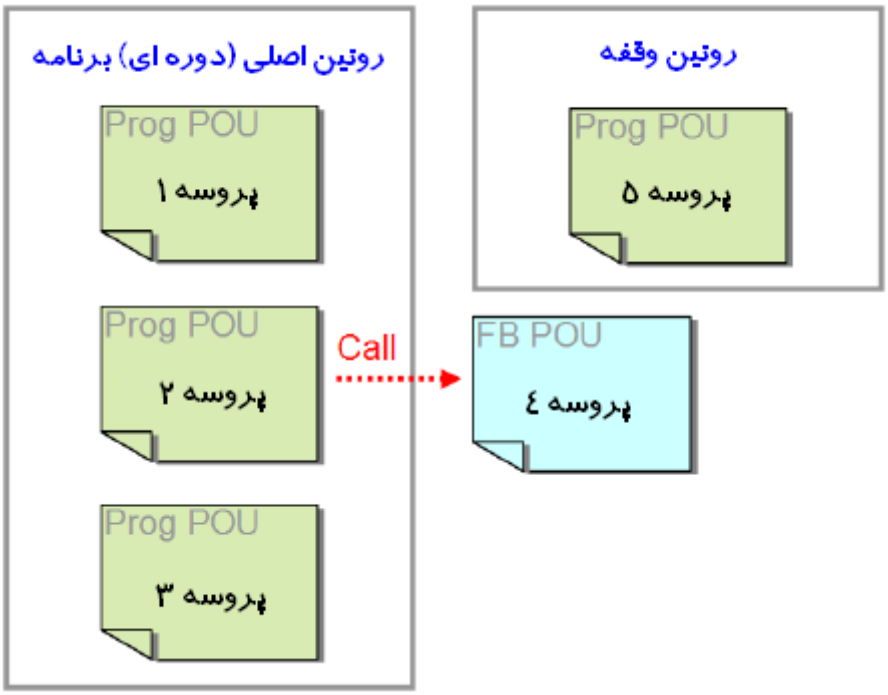
شکل ۴-۳۹ تغییر وضعیت سیستم به حالت آنلاین

پس از ورود به حالت آنلاین، کاربر می‌تواند زمان SCAN فعلی، وضعیت ارتباطی، و وضعیت PLC را در نوار وضعیت مشاهده کند.

شکل ۴-۴۰ نوار وضعیت در حالت آنلاین

فصل ۵– واحد سازماندهی برنامه ها

واحدهای سازماندهی برنامه[۱] یا POUها بخشی کلیدی برای سازماندهی برنامه‌های نوشته شده به صورتی جدید و ساختاریافته است. در این واحدها می‌توانیم بخش‌های مختلف برنامه را به صورت مجزا نوشته و گسترش دهیم. ویژگی مثبت این معماری را می‌توان امکان دسته بندی کردن عملکردهای گوناگون در زیربرنامه‌های مختلف دانست، به صورتی که کاربر می‌تواند در هر زیربرنامه تمرکز خود را تنها بر روی موضوعی خاص معطوف کند.



شکل ۵-۱ نحوه عملکرد برنامه ها در ISPSoft

۵–۱– معماری واحد سازماندهی برنامه

POU یکی از واحدهای اساسی برنامه نویسی PLC است. در POU می‌توان برنامه را به چند بخش مجزا تقسیم کرد. برخلاف نسخه قبلی (برنامه WPLSoft) که یک محیط یکپارچه برای برنامه نویسی داشت، در ISPSoft می‌توان واحدهای مختلفی را در برنامه بر اساس عملکرد یا ویژگی‌های آن در نظر گرفت. این کار در برنامه‌های حجیم امکان گسترش، مدیریت و رفع مشکلات را فراهم می‌کند. همچنین از ان‌جایی که

[۱] Program organization units (POU)

POU به صورت چند بخش مجزا از هم طراحی شده، امکان گسترش هر بخش آن توسط طراح‌های مختلف به صورت مجزا نیز وجود خواهد داشت.

در ISPSoft دو نوع POU وجود دارد، ۱- برنامه‌ها[۱] یا PROGs و ۲- Function blocks یا FBs.

- Program (PROG) یا برنامه

POU برنامه می‌تواند به عنوان روتین اصلی برنامه و یا برنامه وقفه مورد استفاده قرار گیرد. همچنین در آن می‌توان Function blocks (FB) را فراخوانی (اجرا) کرد.

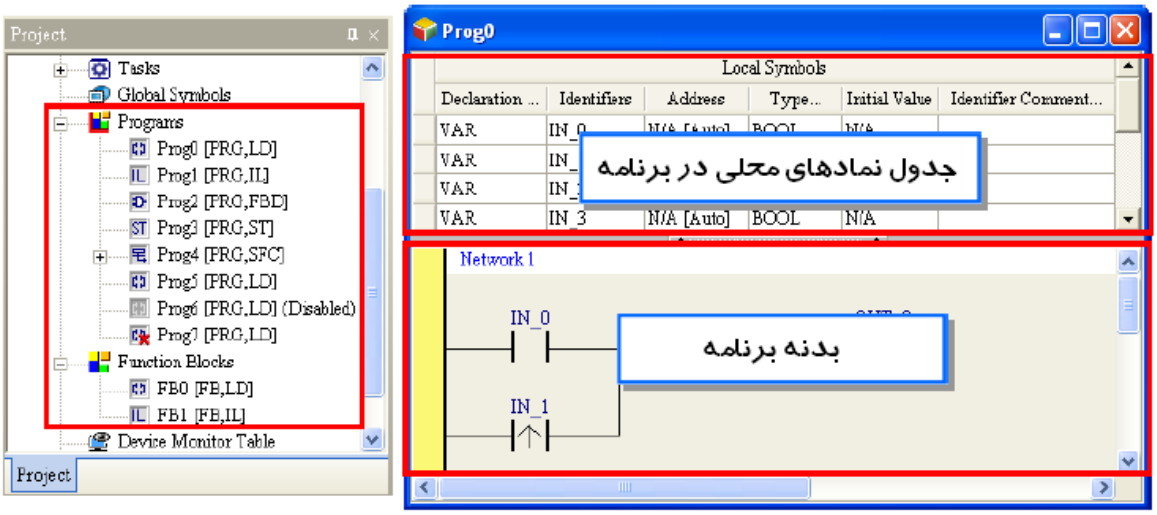
- Function Block (FB)

در FB می‌توان برنامه‌ای نوشت که در هنگام فراخوانی بر اساس ورودی‌ها و حافظه‌ها مقدار خروجی را به ما دهد. با توجه به آنکه مقدار خروجی در FB علاوه بر ورودی می‌تواند به حافظه نیز وابسته باشد، می‌توان انتظار داشت که هرگاه ورودی‌های یکسان به آن اعمال کنیم الزاما نباید خروجی‌ها مشابه باشند. در FB می‌توانیم FB دیگری را فراخوانی (اجرا) کنیم.

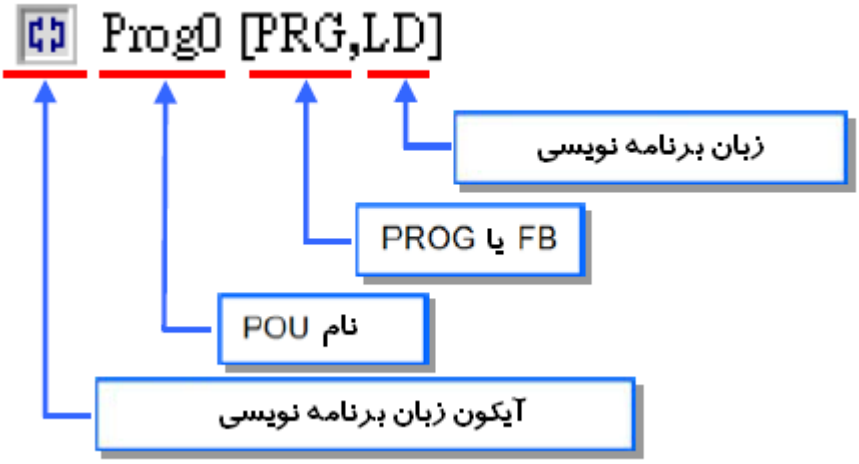
۵-۱-۱- POU ها در ISPSoft

تمام POUها نوشته شده توسط کاربران در بخش مدیریت پروژه لیست می‌شوند. در این بخش POU-های برنامه و FB در دو قسمت جداگانه لیست شده است، همچنین زبان برنامه‌نویسی و اطلاعات هرکدام از POU در ادامه نام آن‌ها آمده است.

[۱] Programs



شکل ۵-۲ لیست POU در بخش مدیریت پروژه



شکل ۵-۳ بخش های مختلف نام POU

جدول ۵-۱: آیکون های POU در بخش مدیریت پروژه

آیکون	شرح
	زبان برنامه نویسی Ladder
	زبان برنامه نویسی Instruction Set
	زبان برنامه نویسی Function block diagram
	زبان برنامه نویسی Structured text
	زبان برنامه نویسی Sequential function chart

	اگر آیکونی POU خاکستری باشد، به معنای غیرفعال بودن آن است. POU زمانی غیرفعال است که برنامه کامپایل شده باشد ولی هنوز اجرا نشده باشد.
	ضربدر قرمز نشانگر آن است که POU در هیچ قسمتی مورد استفاده قرار نگرفته است.

پس از آنکه بر روی آیکون POU در قسمت مدیریت پروژه دوبار کلیک می‌کنیم، صفحه ویرایش POU ظاهر می‌شود. این صفحه شامل دو بخش جدول نمادها (سیمبول‌های) محلی[۱] در بالا و بدنه برنامه برای نوشتن برنامه، در قسمت پایین است. بدنه برنامه با توجه به زبان انتخاب شده برای برنامه نویسی ممکن است ظاهری متفاوت داشته باشد.

۵-۱-۲- عملکردهای POU در ISPSoft

در ISPSoft هر POU دارای وظیفه و در نتیجه آن عملکردی[۲] خاص است، دقیقا مشابه بازی فوتبال که در آن هر بازیکن دارای پُست، وظیفه و عملکردی خاص است (مدافع، هافبک، دروازه‌بان، مهاجم و ...). در ISPSoft هربرنامه وظیفه خاص و به طبع آن اختیارات خاص دارد (مانند دروازه‌بان که بازیکنی است که می‌تواند توپ را با دست بگیرد). هر POU نیاز به اختصاص نوع عملکرد دارد و با توجه به نوع عملکرد آن نحوه‌ی اجرای[۳] آن (وظیفه آن) مشخص می‌شود. اگر به POU عملکردی اختصاص داده نشود، آنگاه POU فقط به عنوان کُدی در پروژه ذخیره شده و در هنگام کامپایل به کُدهای اجرایی[۴] تبدیل نخواهد شد. در ISPSoft سه نوع عملکرد برای POU در نظر گرفته شده است: عملکردهای دوره‌ای، وقفه زمانی و وقفه شرطی[۵].

- عملکرد دوره‌ای

---

[۱] Table of local symbols

[۲] Task

[۳] Execut

[۴] کدهای اجرایی قابلیت اسکن شدن خط به خط را دارند

[۵] Conditional interrupt

POU که به صورت عملکرد دوره ای تنظیم شده باشد، به صورت پی در پی اسکن و اجرا می‌شود. در مدل‌های DVP تنها یک POU به صورت عملکرد دوره‌ای می‌توان تعریف کرد. در حالی که در مدل‌های AH500 می‌توان ۳۲ عملکرد دوره‌ای را در هر پروژه برای CPUهای آن تعریف کرد. عملکردهای دوره‌ای AH500 که با شماره صفر تا ۳۱ مشخص می‌شوند هرچه شماره کوچکتری داشته باشند زودتر اجرا می‌شوند، همچنین می‌توان اجرا و یا عدم اجرای هر کدام از این عملکردهای دوره‌ای را با استفاده از بلوک‌های TKON و TKOFF کنترل کرد.

- عملکرد وقفه زمانی

با توجه به زمان تعیین شده وقفه[1] زمانی اتفاق و POU مرتبط با آن اجرا می‌شود. تعداد وقفه‌های زمانی به نوع PLC وابسته است و به هر وقفه نیز می‌توان تنها یک POU اختصاص داد.

- عملکرد وقفه شرطی

وقفه‌های شرطی گوناگونی مانند وقفه خارجی، وقفه ورودی/خروجی، وقفه شمارنده و ... وجود دارد که با رخداد هرکدام POU متناظر با آن‌ها نیز اجرا خواهد شد. PLC های مختلف انواع متفاوتی از وقفه‌های شرطی را برای ما فراهم می‌کنند.

POUهایی که به آن‌ها عملکردهای خاصی اختصاص داده شده است در قسمت Task در بخش مدیریت پروژه لیست می‌شوند. ترتیب لیست شدن فایل‌ها در قسمت Task مناسب با ترتیب اجرا شدن آن‌ها است. برای مثال در پروژه زیر ۹ عدد POU وجود دارد که عملکردهای آن به صورت زیر است.

---

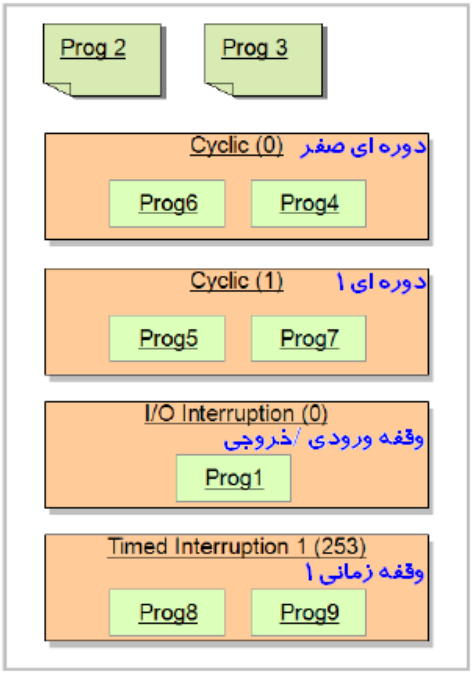
۱ وقفه یا Interrupt تحریکی برای پردازنده (در اینجا CPU) است که به رسیدگی سریع پردازنده نیاز دارد. هنگامی که یک وقفه رخ می‌دهد، پردازنده عملیات جاری خود را متوقف می‌کند تا به درخواست وقفه رسیدگی کند.

شکل ۵-۴ POU هایی که به آن‌ها عملکرد (Tsak) های متفاوتی تخصیص داده شده

جدول ۵-۲: عملکردهای تخصیص داده شده به POU

نام عملکرد	**POU**	**نحوه اجرا**
تخصیص داده نشده	Prog2 و Prog3	با توجه به آنکه Prog2 و Prog3 به عملکردی تخصیص داده نشده‌اند، اجرا نخواهند شد.

Cyclic (0)	Prog6 و Prog4	با توجه به آنکه در عملکردهای دوره‌ای شماره‌های کوچکتر زودتر اجرا می‌شوند، ترتیب اجرا POUها در این بخش به صورت زیر می‌باشد. (در اینجا چون 0<1 است به ترتیب POU های Cyclic(0) و سپس Cyclic(1) اجرا خواهند شد. ترتیب لیست شدن POUهای هر Cyclic نیز به ترتیب لیست شدن آن ها در نمودار درختی خواهد بود) Prog6 --> Prog4 --> Prog5 --> Prog7
Cyclic (1)	Prog5 و Prog7	
وقفه ورودی/خروجی	Prog1	زمانی که شرط وقفه اتفاق بیافتد، Prog1 یکبار اجرا می‌شود.
وقفه زمانی ۱	Prog8 و Prog9	وقفه زمانی۱ در دوره های زمانی تنظیم شده اتفاق و آنگاه POUها به ترتیب زیر یکبار اتفاق می‌افتند. Prog8 -> Prog9



شکل ۵-۵ عملکردهای تخصیص داده شده به برنامه ها

توجه: اگر کاربر بخواهد عملکرد وقفه را فعال کند، باید بلوک EI را در برنامه دوره‌ای خود قرار دهد.

شکل ۵-۶ استفاده از بلوک E1 برای فعال کردن وقفه

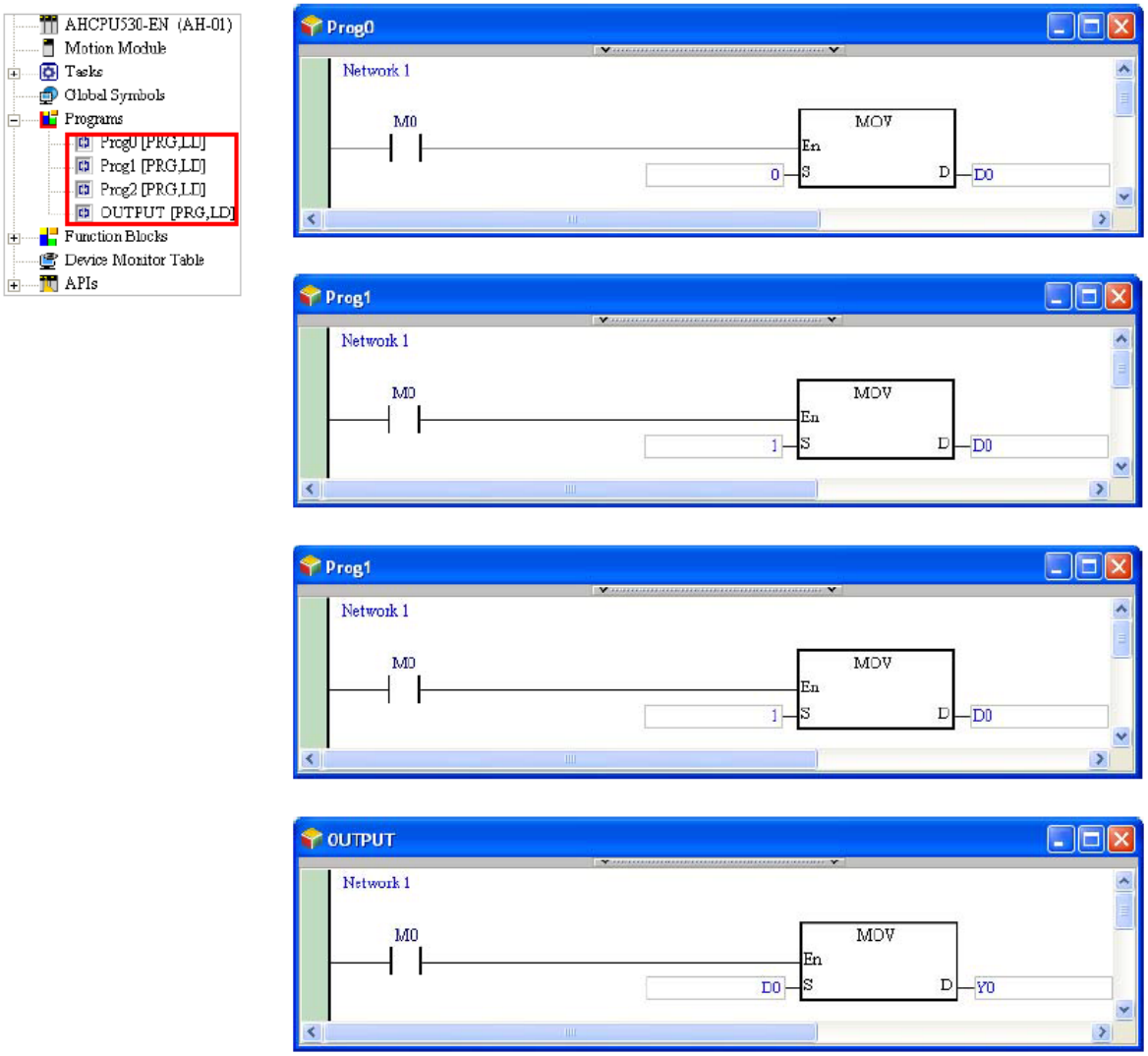
۵-۱-۲-۱- اختصاص یک عملکرد به چند POU

در حالتی که بیشتر از یک POU به عملکردی خاص مانند Cyclic (0) اختصاص داده شود، ترتیب اجرای آن‌ها همانند ترتیب لیست شدن آن‌ها در قسمت Task در بخش مدیریت پروژه خواهد بود. برای روشن شدن موضوع به مثال زیر توجه کنید. در پروژه‌ای چهار POU وجود دارد که شرح آن‌ها در جدول زیر آمده است.

جدول ۵-۳: مثال۱- پروژه ای با چهار POU

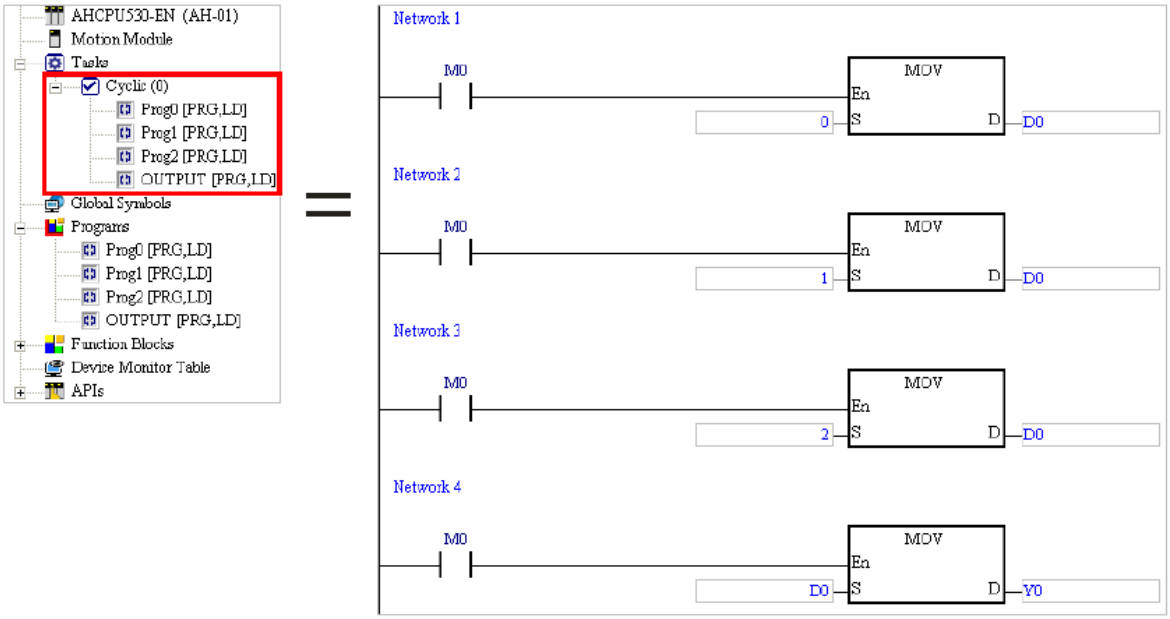
نام POU	شرح
Prog0	زمانی که M0 یک است، D0 برابر صفر است.
Prog1	زمانی که M0 یک است، D0 برابر یک است.
Prog2	زمانی که M0 یک است، D0 برابر دو است.
OUTPUT	زمانی که M0 یک است، مقدار Y0 برابر مقدار D0 شود.

برنامه POUهای ذکر شده به صورت زیر است.



شکل ۵-۷ مثال۱- برنامه های چهار POU

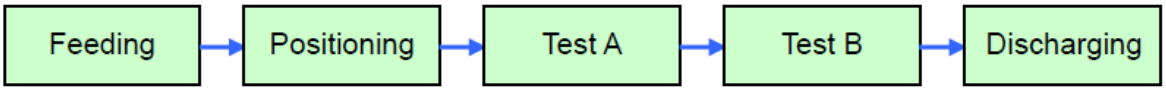
با توجه به آنکه هر چهار POU در Cyclic task (0) قرار دارند، ترتیب اجرای آن‌ها به ترتیب قرار گرفتن نام آن‌ها در قسمت Task می‌باشد. بدین ترتب ابتدا Prog0 و سپس Prog1، Prog2 و OUTPUT اجرا خواهند شد و در نتیجه مقدار خروجی Y0 در پایان زمان Scan برابر مقدار D0 در Prog2 خواهد بود، یعنی D0 و در نتیجه آن Y0 برابر ۲ خواهند شد. برنامه معادل چهار POU در زیر آمده است.



شکل ۵-۸ مثال۱- برنامه معادل چهار POU

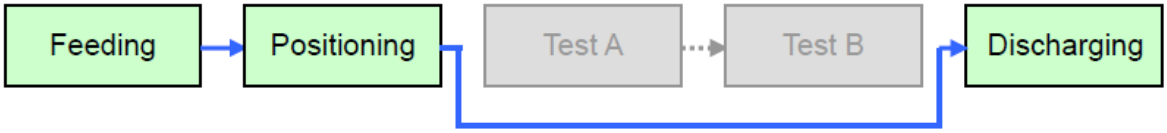
۵-۱-۳- فعال کردن POUها

در ISPSoft امکان غیر فعال کردن POUها به صورت موقت وجود دارد. اگر POU غیر فعال شود، کامپایل نشده و اجرا نخواهد شد. این ویژگی در هنگام تست برنامه و همچنین عیب‌یابی بسیار مفید است. به عنوان مثال می‌توان برای حل مشکلی در سیستم، برنامه تستی نوشت تا با استفاده از آن بتوان عملکرد سیستم را ارزیابی کرد.در نهایت پس از حل شدن مشکل، می‌توان قسمت‌های تست را غیرفعال کرد. مطابق با همین وضعیت در بلوک دیاگرام زیر برای بررسی عملکرد اجزای مختلف سیستم و تکمیل آن‌ها دو مرحله تست در نظر گرفته شده است.



شکل ۵-۹ در نظر گرفتن مراحل تست برای ارزیابی سیستم

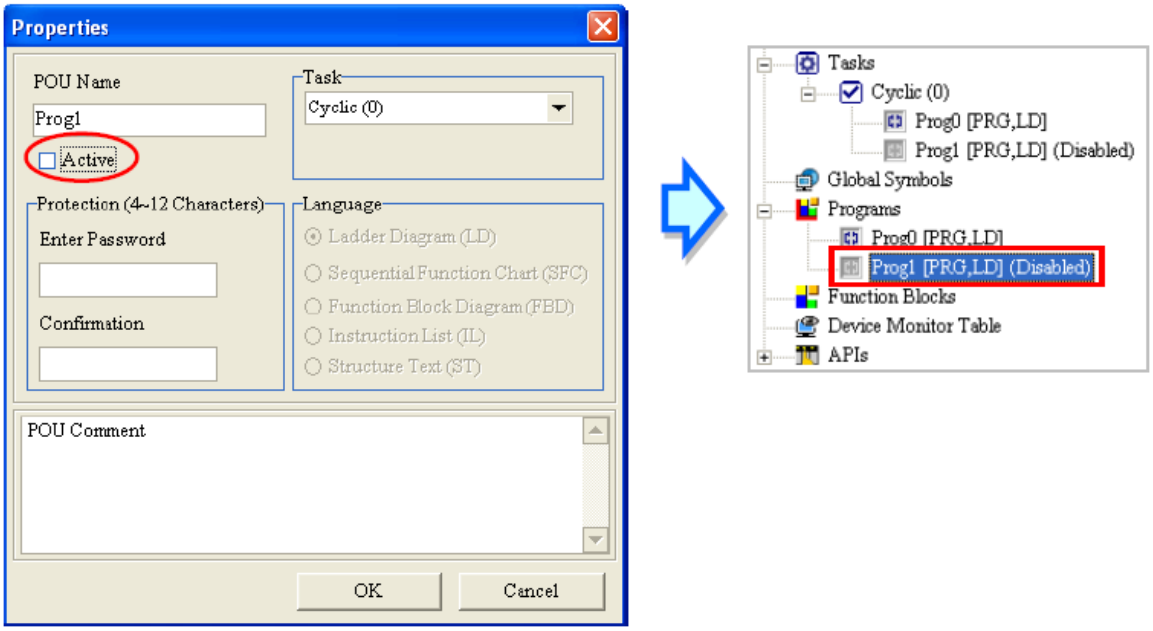
که بعد از تکمیل پروژه و حل مشکلاتِ سیستم، این مراحل تست حذف خواهند شد.



شکل ۵-۱۰ غیرفعال کردن POU مرتبط با تست پس از پایان ارزیابی سیستم

(البته می‌توان به این وضعیت اینگونه نگاه کرد که چون در مراحل راه‌اندازی سیستم اصلی، تجهیزات دو بخش ارزیابی تکمیل نشدند، می‌توانیم از اجرای این دو بخش صرف نظر کنیم و لازم است برنامه‌های مرتبط با این دو بخش غیرفعال شوند)

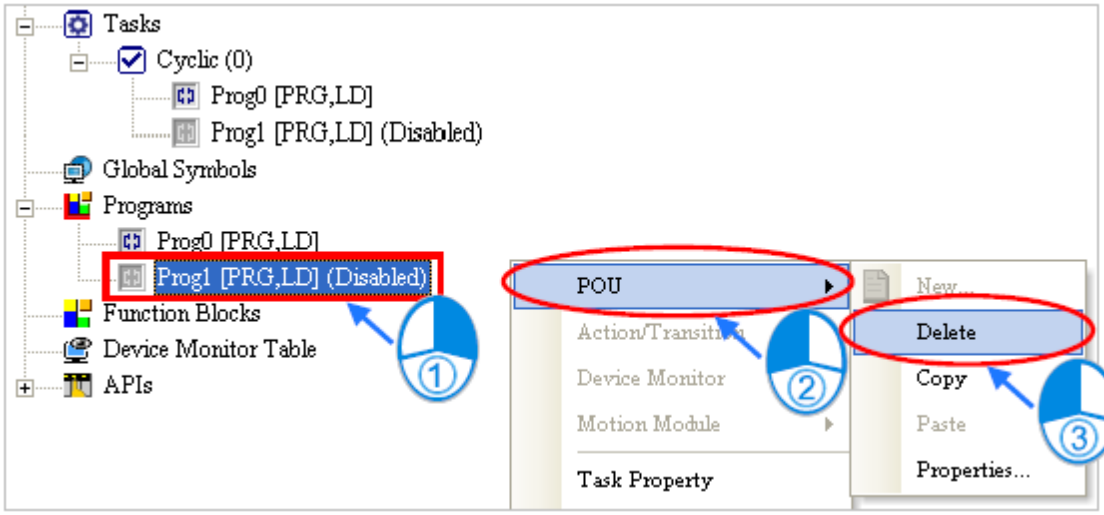
اگر کاربر بخواهد POU را فعال کند باید Active را در صفحه ساخت پروژه انتخاب کند. در غیر اینصورت اگر گزینه Active فعال نباشد، POU غیرفعال و آیکون آن خاکستری می‌شود.



شکل ۵-۱۱ POU غیرفعال

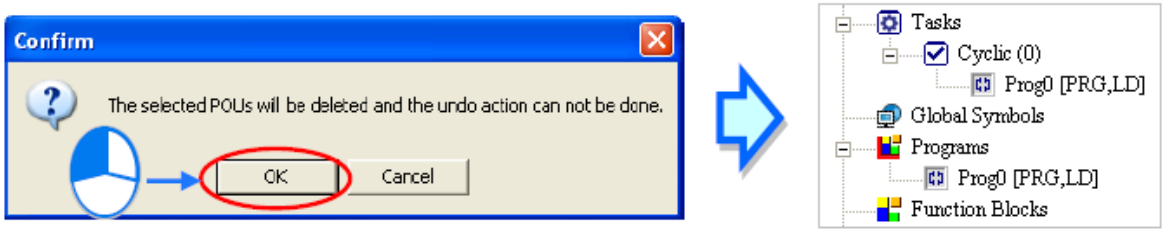
۵-۱-۴- پاک کردن و کپی POU

برای پاک کردن POU می‌توان بر روی آن در قسمت Program کلیک راست کرده و در منوی باز شونده POU بر روی Delete کلیک کرد.



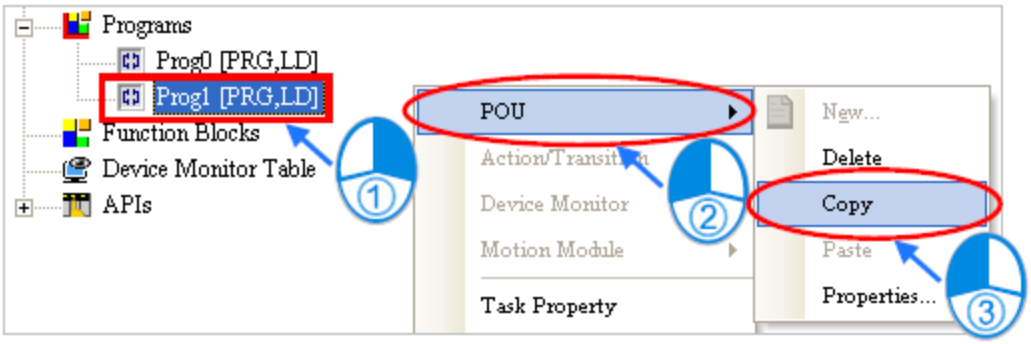
شکل ۵-۱۲ حذف POU

توجه شود که در صورت پاک شدن POU، نمی‌توان به هیچ وجه آن را بازیابی کرد و آیکون آن از قسمت Task نیز حذف خواهد شد.



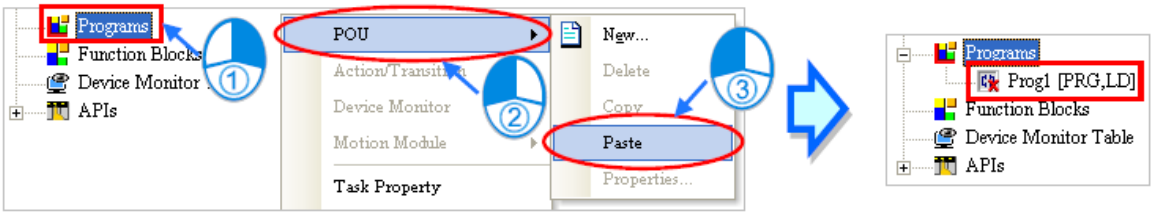
شکل ۵-۱۳ هشدار در مورد اینکه در صورت حذف شدن POU دیگر قابل بازگردانی نیست

برای کپی کردن POU می‌توان بر روی آن کلیک راست کرده و در منوی باز شونده POU بر روی Copy کلیک کرد.



شکل ۵-۱۴ کپی کردن POU

POU کپی شده را می‌توان در همان پروژه و یا پروژه‌های دیگر استفاده کرد. اگر POU کپی شده از نوع برنامه باشد، آن را فقط در قسمت برنامه می‌توان کپی کرد و اگر از نوع FB باشد، آن را فقط در قسمت Function Block می‌توان کپی کرد. برای الحاق POU کپی شده می‌توان بر روی Program یا Function Blocks در بخش مدیریت پروژه کلیک راست کرده و در منوی باز شونده POU گزینه Past را انتخاب می‌کنیم. در این حالت POU الحاق شده در قسمت برنامه به هیچ عملکردی اختصاص داده نشده و باید عملکرد آن را تعیین کرد.



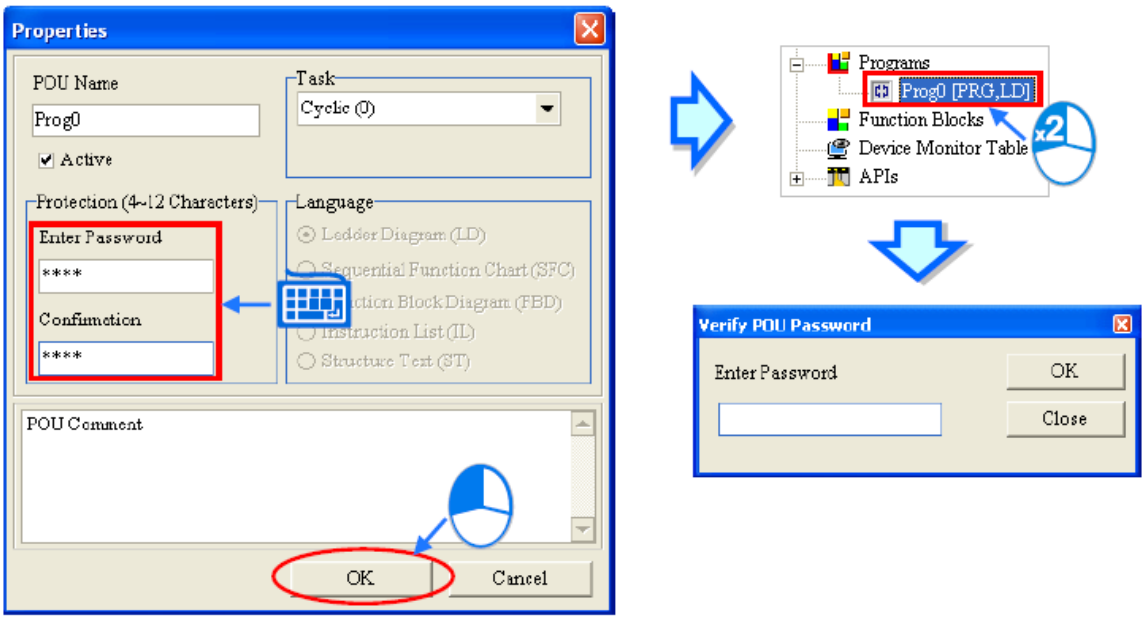
شکل ۵-۱۵ الحاق POU کپی شده

۵-۱-۵- رمز عبور

کاربران می‌توانند برای امنیت بیشتر بر روی POU خود رمز عبور قرار دهند. برای فعال کردن رمز عبور POU می‌توان بر روی آن کلیک راست کرده و در منوی باز شونده POU گزینه Properties را انتخاب کرد.

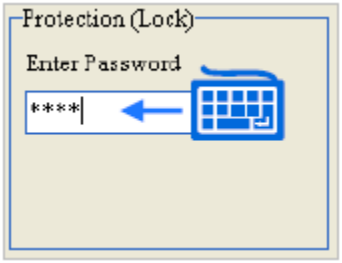
شکل ۵-۱۶ نحوه فعال کردن رمز عبور برای POU –قسمت اول

در پنجره باز شده باید رمزعبور را دوبار در قسمت Enter Password و Confirmation وارد کرد و در نهایت بر روی OK کلیک کرد. رمز عبور انتخابی باید بین۴ الی ۱۲ کاراکتر و شامل حروف، ارقام و یا بعضی نمادهای مجاز مانند "_" باشد. در صورتی که به POU رمز عبور اختصاص داده شود برای دسترسی به آن باید ابتدا رمزعبور وارد شود.



شکل ۵-۱۷ نحوه فعال کردن رمز عبور برای POU –قسمت دوم

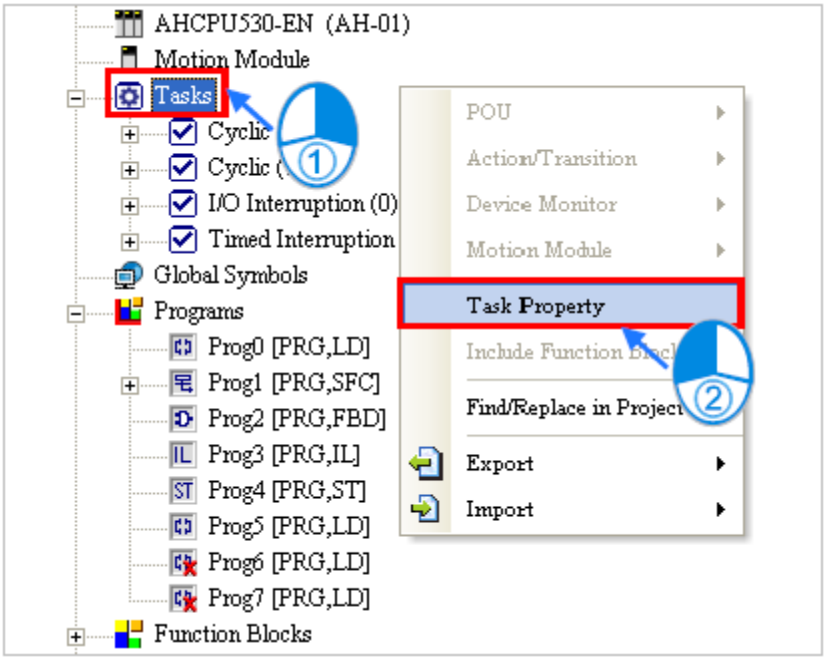
در صورتی که نیاز به حذف شدن رمز عبور باشد می‌توان دوباره به پنجره Properties مراجعه و در قسمت Enter Password رمز عبور را وارد و تایید کرد.



شکل ۵-۱۸ وارد کردن رمز عبور برای POU

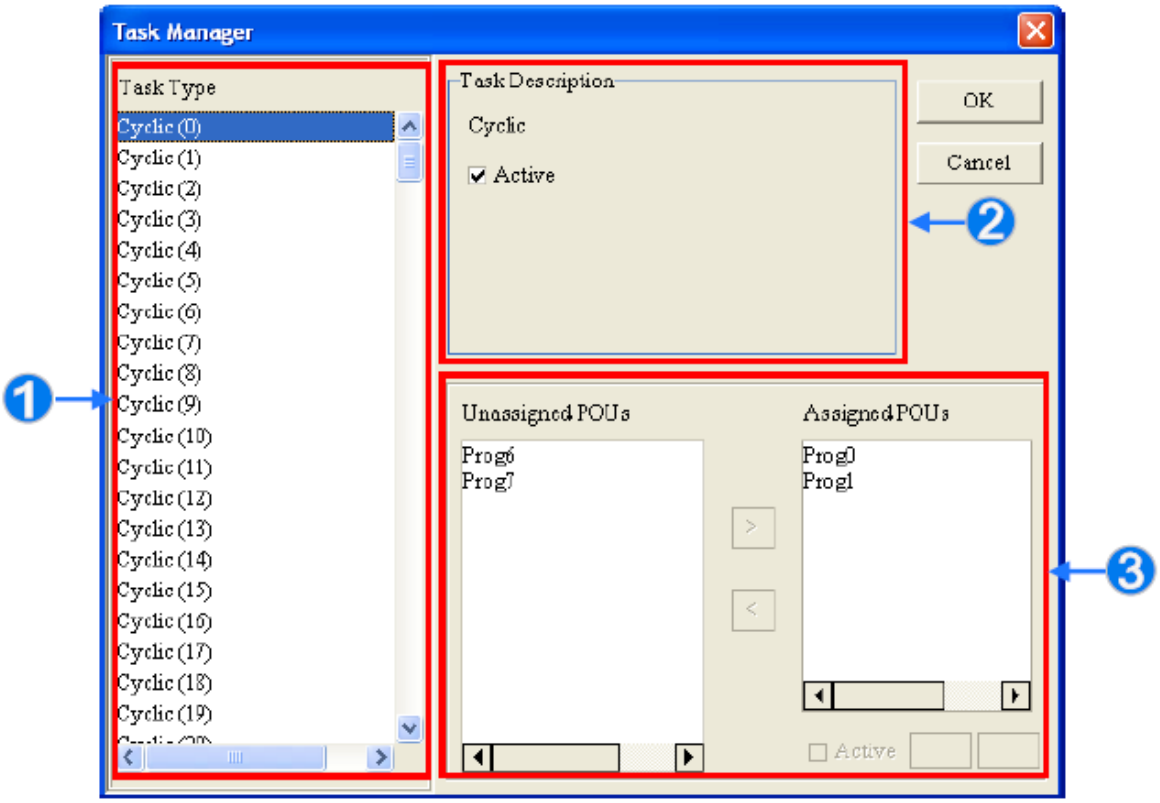
۵-۲- مدیریت عملکردها

برای مدیریت عملکرد POUها می‌توان بر روی Tasks در قسمت مدیریت پروژه کلیک راست کرده و Task Property را انتخاب کرد.



شکل ۵-۱۹ مدیریت عملکرد POU

پنجره باز شده به صورت زیر خواهد بود.



شکل ۵-۲۰ پنجره مدیریت عملکرد POU

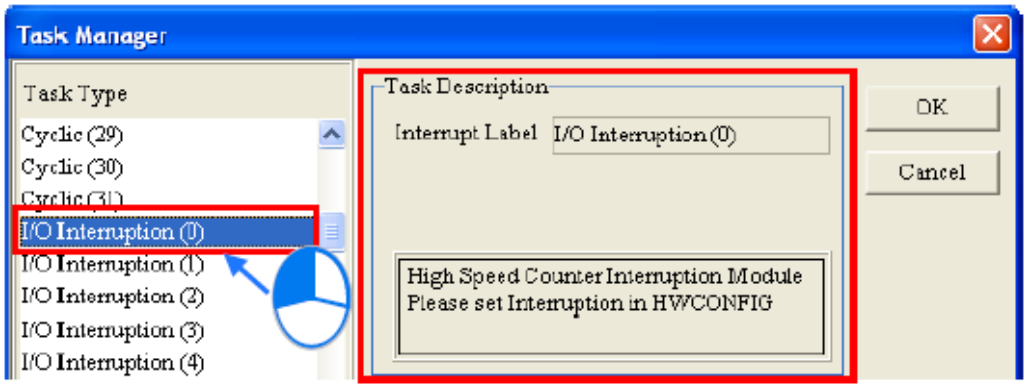
1 Task type: عملکردها قابل استفاده در این قسمت لیست شده‌اند.

2 Task description: توصیف عملکرد انتخاب شده در این قسمت آمده است.

3 Task management area: در این بخش می‌توان POU ها را سازماندهی و به آن‌ها عملکرد اختصاص داد.

۵-۲-۱- تنظیمات عملکردها و شرط رخ دادن وقفه ها

پس از آنکه عملکرد در قسمت Task Type انتخاب شد، شرح توضیحات مربوط به آن در قسمت Task Description داده خواهد شد. در این بخش کاربر می‌تواند نحوه اجرای عملکردها را نیز تعیین نماید. توجه کنید که Taskهای لیست شده در Task manager با توجه به نوع PLC انتخاب شده ممکن است متفاوت باشند، همچنین تنظیمات این عملکردها نیز وابسته به نوع PLC متفاوت است.



شکل ۵-۲۱ انتخاب نوع عملکرد و شرح آن

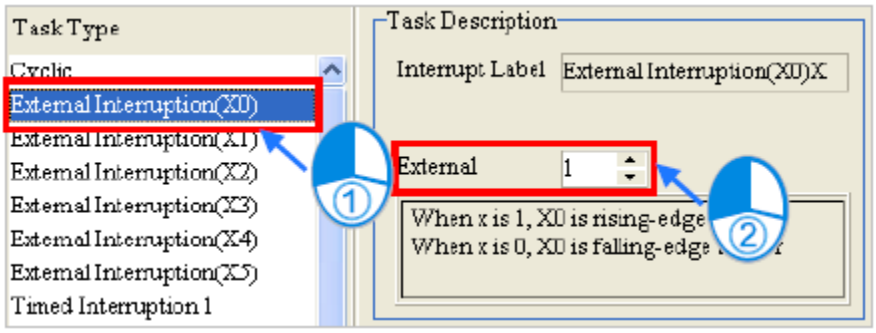
تنظیمات عملکردها در این بخش را با چند مثال شرح خواهیم داد.

مثال ۱: برای تنظیم عملکرد وقفه در DVP-SV ابتدا لیست وقفه‌های آن را مرور خواهیم کرد. (جدول زیر را در DVP-PLC Application Manual-Programming می‌توانید مشاهده کنید)

External interrupt	I00□(X0), I10□(X1), I20□(X2), I30□(X3), I40□(X4), and I50□(X5) (□=1: Rising edge-triggered ; □=0: Falling edge-triggered)
Timed interrupt	I6□□ and I7□□ (□□=1~99; Time unit=1 ms) I8□□ (□□=1~99; Time unit=0.1 ms)
High-speed counter interrupt	I010, I020, I030, I040, I050, and I060
Pulse interrupt	I110, I120, I130, and I140
Communication interrupt	I150, I160, and I170
Frequency measurement card interrupt	I180

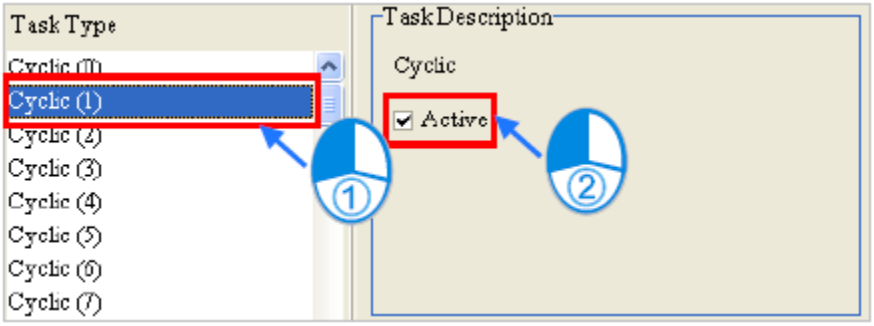
شکل ۵-۲۲ لیست وقفه های DVP-SV

انواع عملکرد برای DVP-SV در زیر آمده است. اگر کاربر بخواهد وقفه I001 را تنظیم کند، باید External Interruption (X0) را انتخاب کند (I001 با لبه بالارونده ورودی X0 تحریک می‌شود). شرح عملکرد External Interruption (X0) در قسمت Task Description آمده است، طبق توضیحات در صورتی که مقدار External یک تعیین شده باشد، به ازای لبه بالارونده I001 تحریک می‌شود.



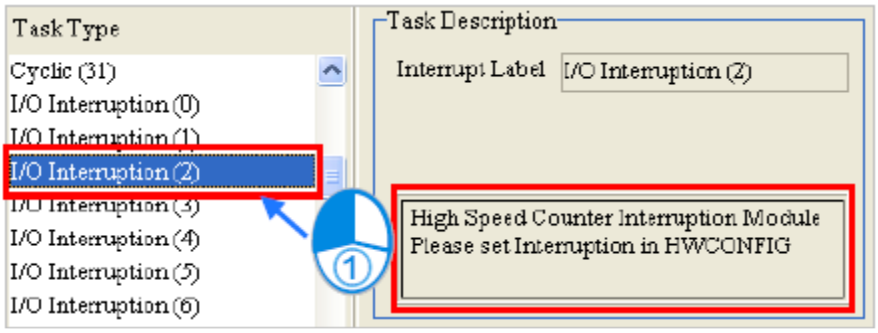
شکل ۵-۲۳ عملکرد وقفه External Interruption (X0) در DVP-SV

مثال ۲: در زیر، پنجره Task Manage را برای PLC مدل AHCPU503-EN مشاهده می‌کنید. به ازای انتخاب عملکرد Cyclic (1) مشاهده می‌شود که کاربر امکان فعال و غیرفعال کردن آن را دارد (در واقع می‌توان تمام عملکردهای دوره‌ای را در AH500 فعال و غیرفعال کرد.



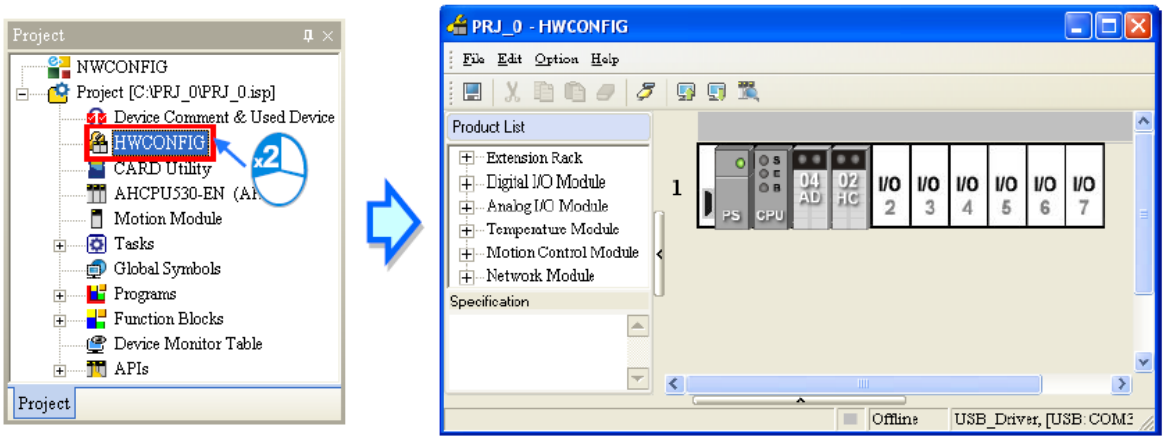
شکل ۵-۲۴ عملکرد دوره ای صفر در AHCPU503-EN

مثال ۳: در زیر Task Manager را برای AHCPU530-EN مشاهده می‌کنید که در آن عملکرد I/O Intrruption (2) انتخاب شده است. طبق توضیحات مشخص است که باید تنظیمات این وقفه را در HWCONFIG انجام داد.



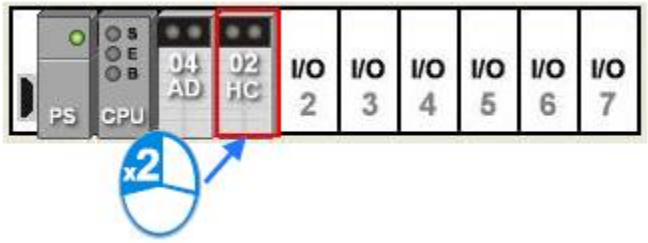
شکل ۵-۲۵ عملکرد وقفه I/O Interruption (2) برای AHCPU530-EN

I/O Intrruption (2) پس از آنکه ماژول AH02HC-5A تعداد مشخصی پالس دریافت کرد، تحریک خواهد شد. برای تنظیم ویژگی‌های آن پنجره HWCONFIG را در بخش مدیریت پروژه باز می‌کنیم.



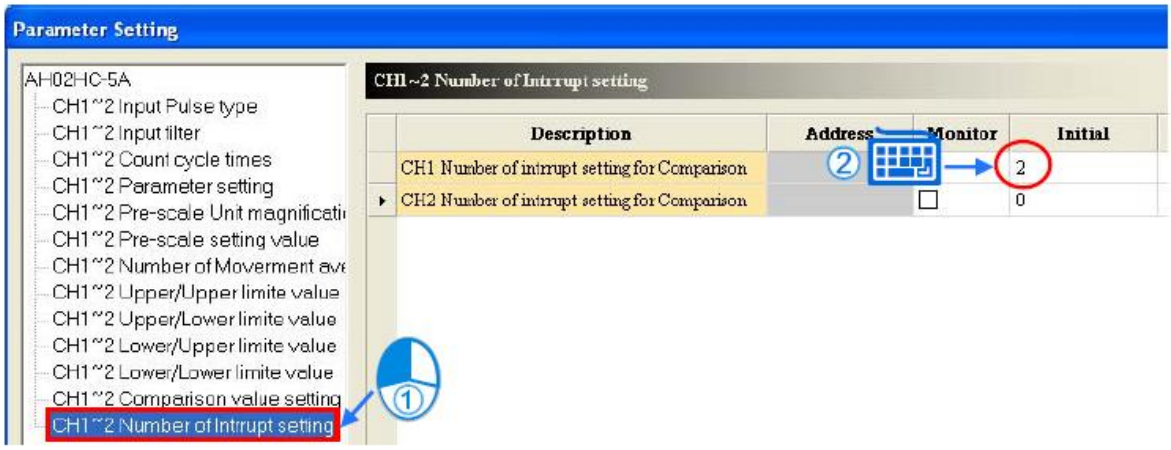
شکل ۵-۲۶ باز کردن پنجره HWCONFIG برای تنظیم I/O Intrruption (2)

در این صفحه دوبار بر روی AH02HC-5A کلیک می کنیم تا صفحه تنظیم پارامترهای آن باز شود.



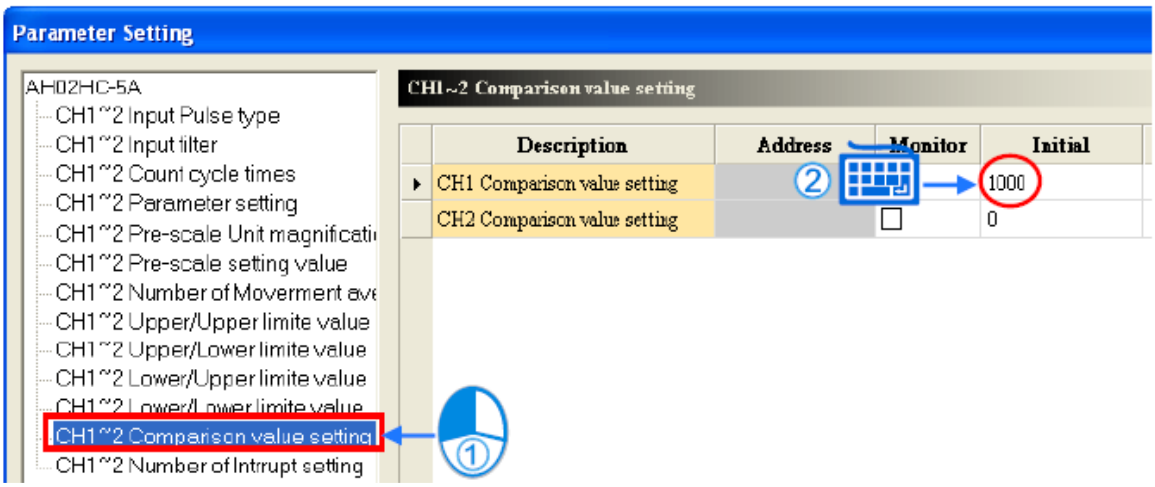
شکل ۵-۲۷ انتخاب AH02HC-5A برای تنظیم I/O Intrruption (2)

در این قسمت CH1~2 Number of Intrrupt Setting را انتخاب و در در ستون Initial مقدار 2 را برای CH1 Number of Interrupt Setting for Comparison تعیین کنید.



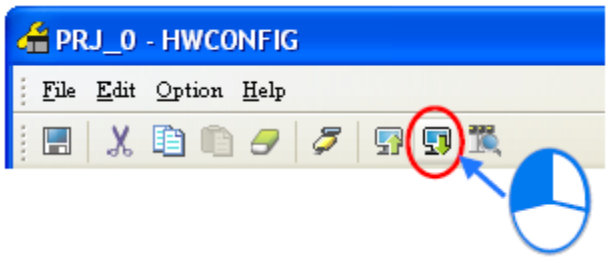
شکل ۵-۲۸ تنظیم CH1 Number of Interrupt Setting for Comparison

حال در CH1~2 Comparison Value Setting را انتخاب و در در ستون Initial مقدار 1000 را برای CH1 Comparison Values Setting تعیین کنید.



شکل ۵-۲۹ تنظیم CH1 Comparison Values Setting

در انتها نیز برای اعمال تنظیمات HWCONFIG باید آن را بر روی PLC بارگزاری کرد.

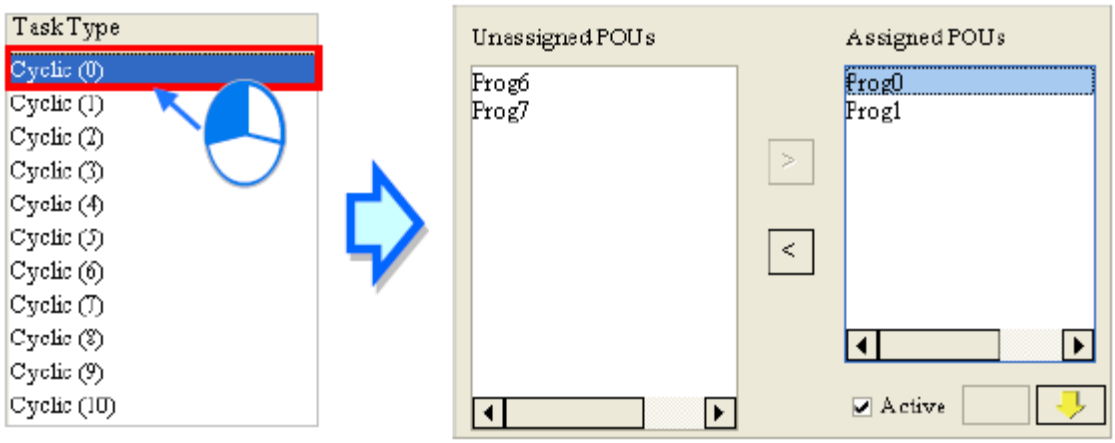


شکل ۵-۳۰ بارگزاری تنظیمات سخت افزاری عملکرد وقفه

(برای اطلاعات بیشتر در مورد تنظیم پارامترهای وقفه PLCهای AH500 به دستورالعمل‌های آن مراجعه نمایید)

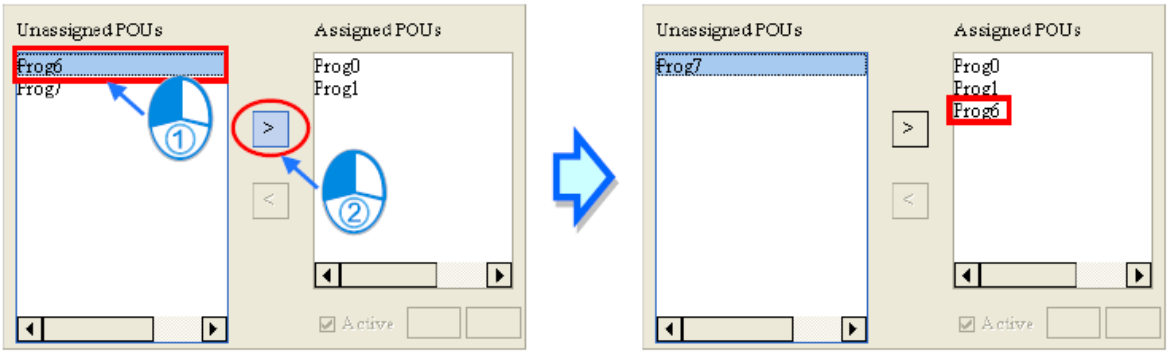
۵-۲-۲- تخصیص عملکرد به POUها

پس از تخصیص عملکردها می‌توان POUها را برای چگونگی اعمال عملکرد مدیریت کرد. پس از انتخاب عملکرد در Task Type، امکان مدیریت POUهای مرتبط در قسمت Assignment of POU ممکن خواهد بود. در قسمت Unassigned POU همه‌ی POUهایی که به عملکرد مورد نظر تخصیص داده نشده‌اند لیست شده‌است. در قسمت Assigned POU نیز POUهایی که به عملکرد مورد نظر تخصیص داده شده‌اند لیست شده‌اند.



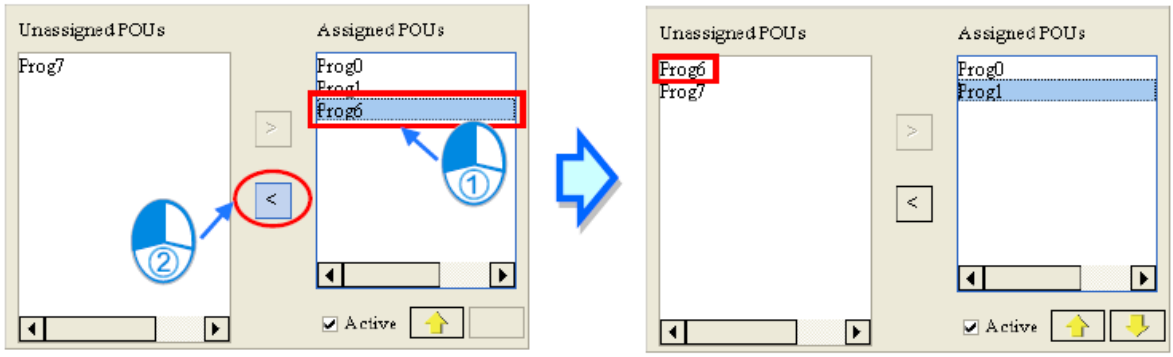
شکل ۵-۳۱ POU تخصیص داده شده و تخصیص داده نشده به هر عملکرد

برای اینکه عملکردی را به POU خاصی اختصاص دهیم، می‌توانیم آن POU را در قسمت Unassigned POU انتخاب و پس از کلیک بر روی > آن را به قسمت Assigned POU منتقل کنیم.



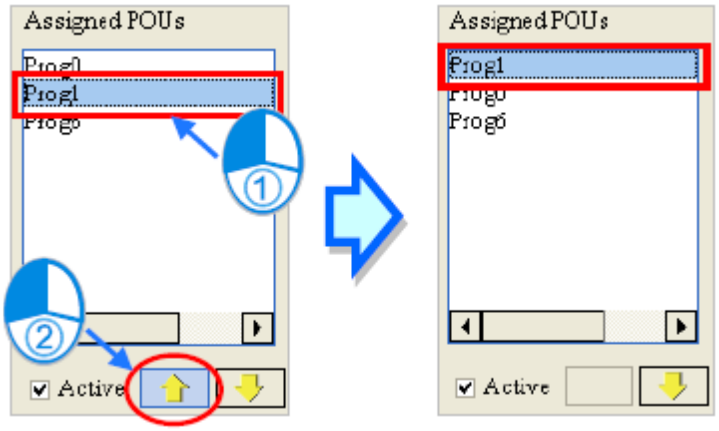
شکل ۵-۳۲ اختصاص عملکرد به POU

برای حذف عملکرد POU نیز می‌توان آن را در قسمت Assigned POU انتخاب و سپس با کلیک بر روی < به قسمت Unassigned POU منتقل کرد.

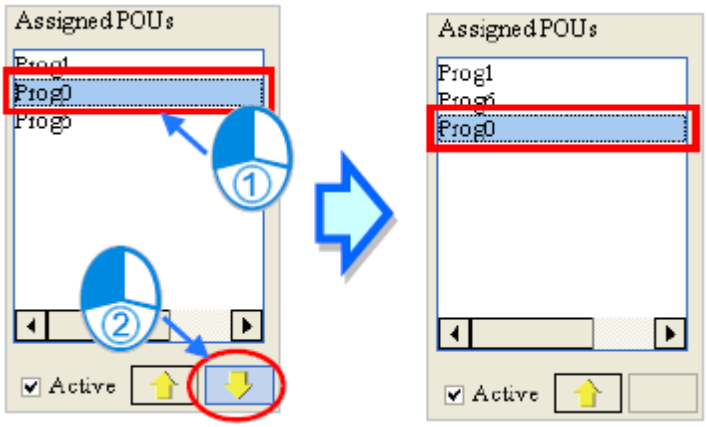


شکل ۵-۳۳ حذف عملکرد POU

در صورتی که بخواهیم ترتیب اجرای POUها را در هر عملکرد تغییر دهیم، می‌توانیم در بخش Assigned POU با استفاده از نوبت آن‌ها را ارتقا و با استفاده از اولویت اجرای آن‌ها را کاهش دهیم.



شکل ۵-۳۴ افزایش اولویت اجرای POU



شکل ۵-۳۵ کاهش اولویت اجرای POU

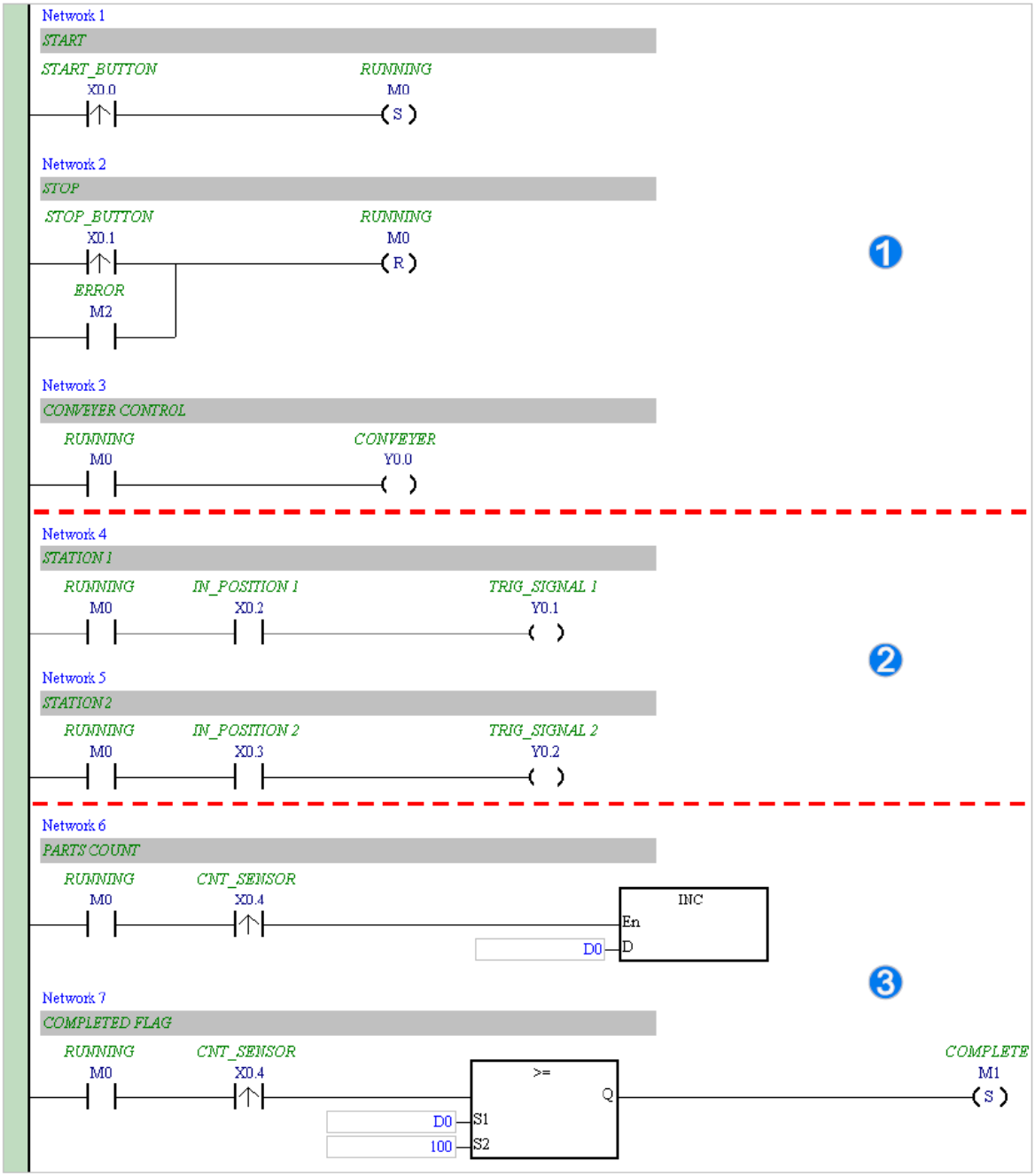
در Assigned POU بعد از انتخاب هر POU می‌توانیم با غیرفعال کردن گزینه Active، POU مورد نظر را نیز غیرفعال کنیم. در این حالت آیکون POU مورد نظر در قسمت Task به رنگ خاکستری در خواهد آمد.

۵–۳– مثال های کاربردی POU

در ادامه برای مرور مفاهیم مطرح شده پیرامون POU در این فصل سه مثال را مطرح خواهیم کرد.

۵–۳–۱– مثال خط تولید تزریق چسب

برنامه نوشته شده در فصل قبل را در نظر بگیرید. می‌خواهیم آن را با استفاده از قابلیت‌های POU به صورتی ساختاریافته تر بنویسیم. این برنامه مطابق شکل زیر شامل سه بخش است.



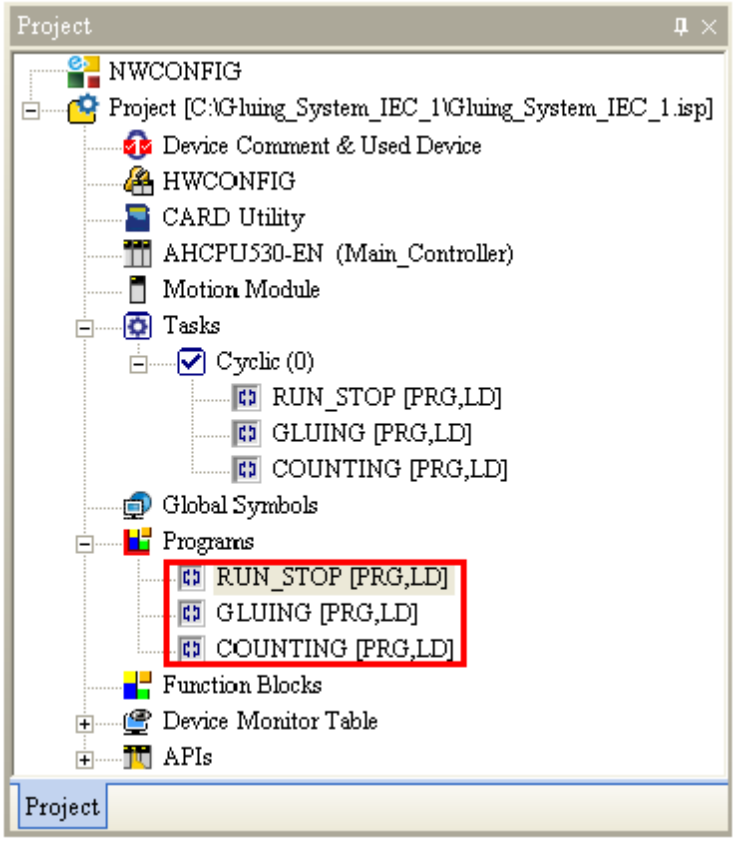
شکل ۵-۳۶ مثال۱- بخش های مختلف برنامه

1 بخش شروع به کار و توقف سیستم

2 بخش تزریق چسب

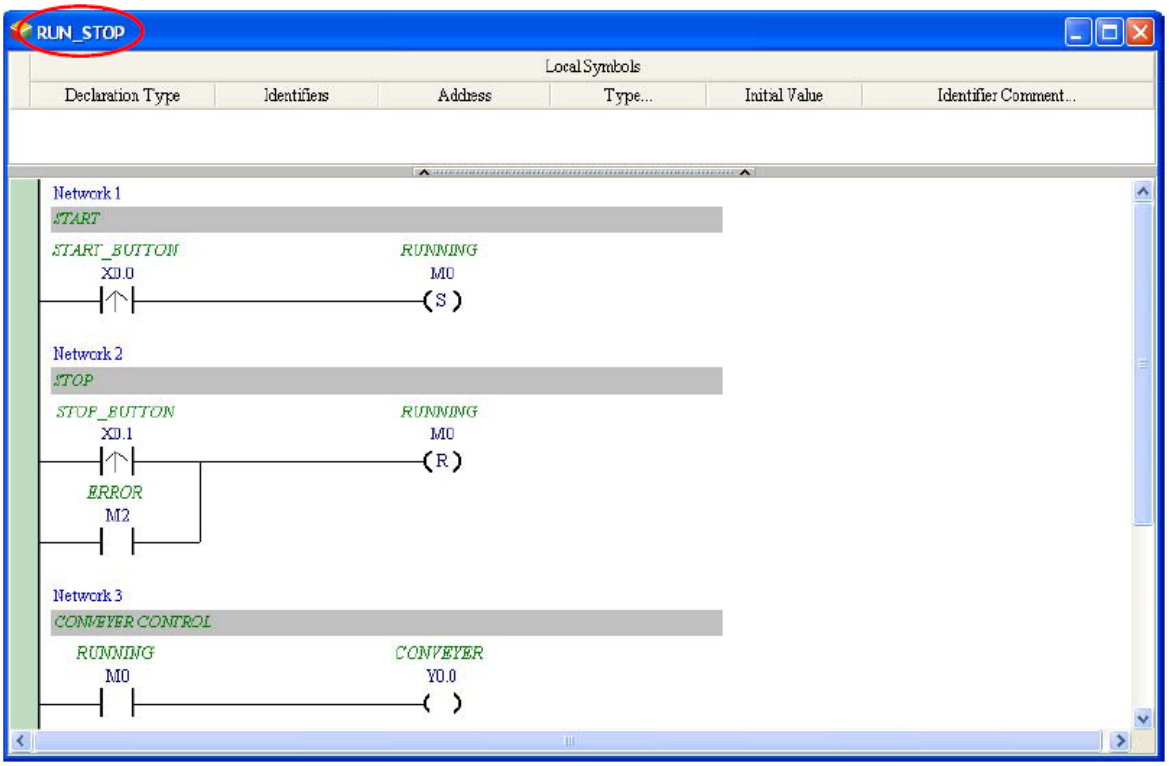
3 بخش شمارش قطعات

در ادامه می‌خواهیم این سه بخش را در پروژه‌ای مشابه در سه POU مجزای اختصاص داده شده به Cyclic (0) تعریف کنیم.



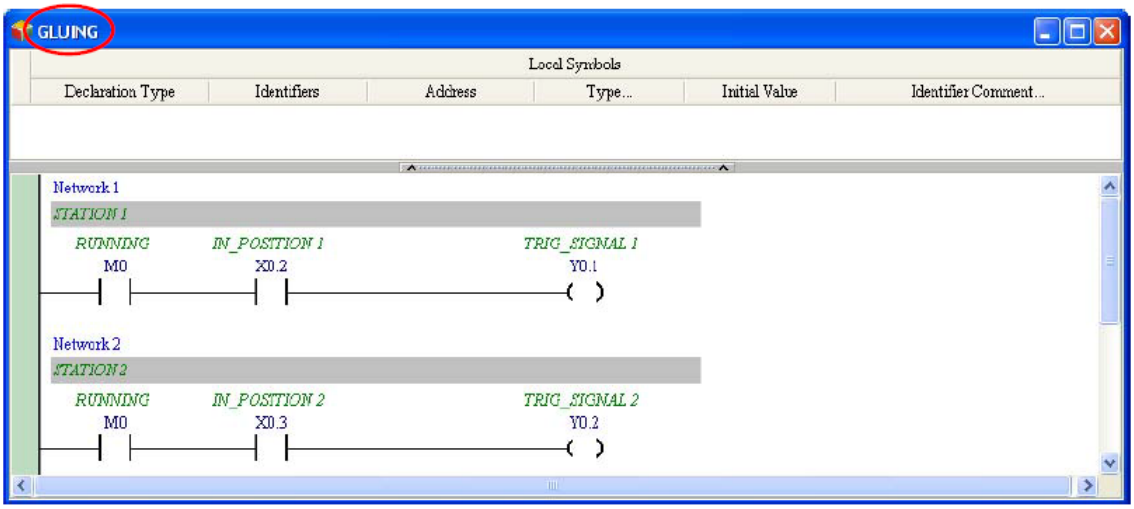
شکل ۵-۳۷ مثال۱- ساخت سه POU معادل

برنامه RUN_STOP ما برای شروع به کار و توقف سیستم به صورت زیر در خواهد آمد.



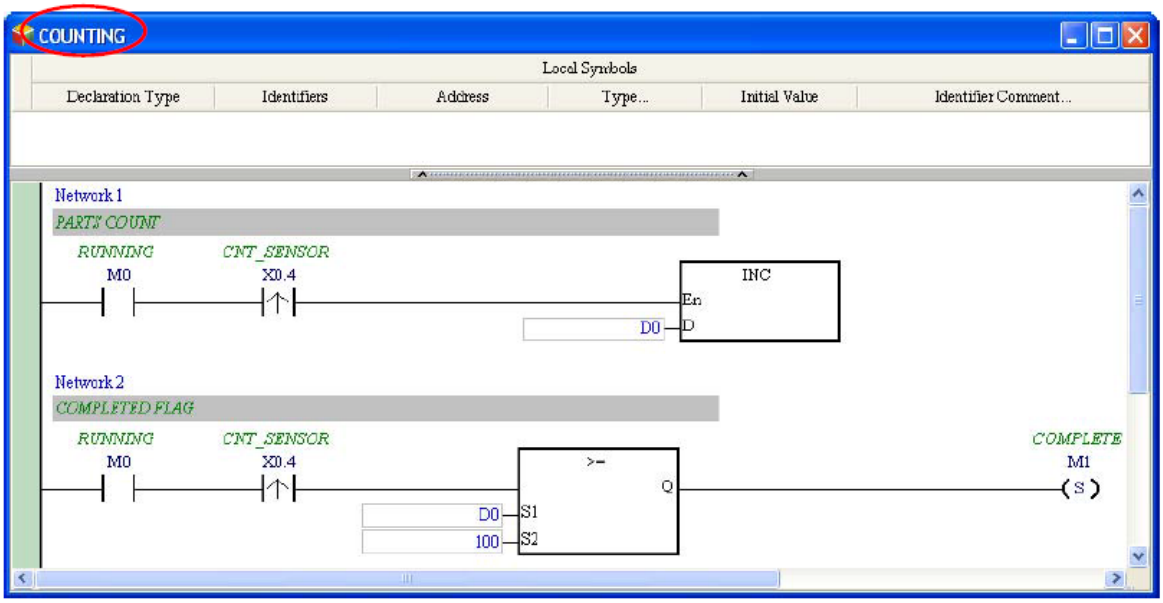
شکل ۵-۳۸ مثال۱- برنامه RUN_STOP

برنامه GLUING برای تزریق چسب نیز به صورت زیر درخواهد آمد:



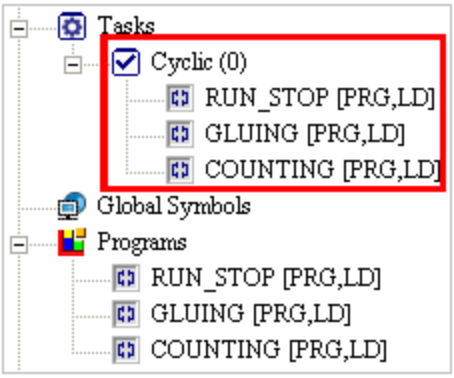
شکل ۵-۳۹ مثال۱- برنامه GLUING

برنامه COUNTING نیز برای شمارش قطعات به صورت زیر در خواهد آمد:



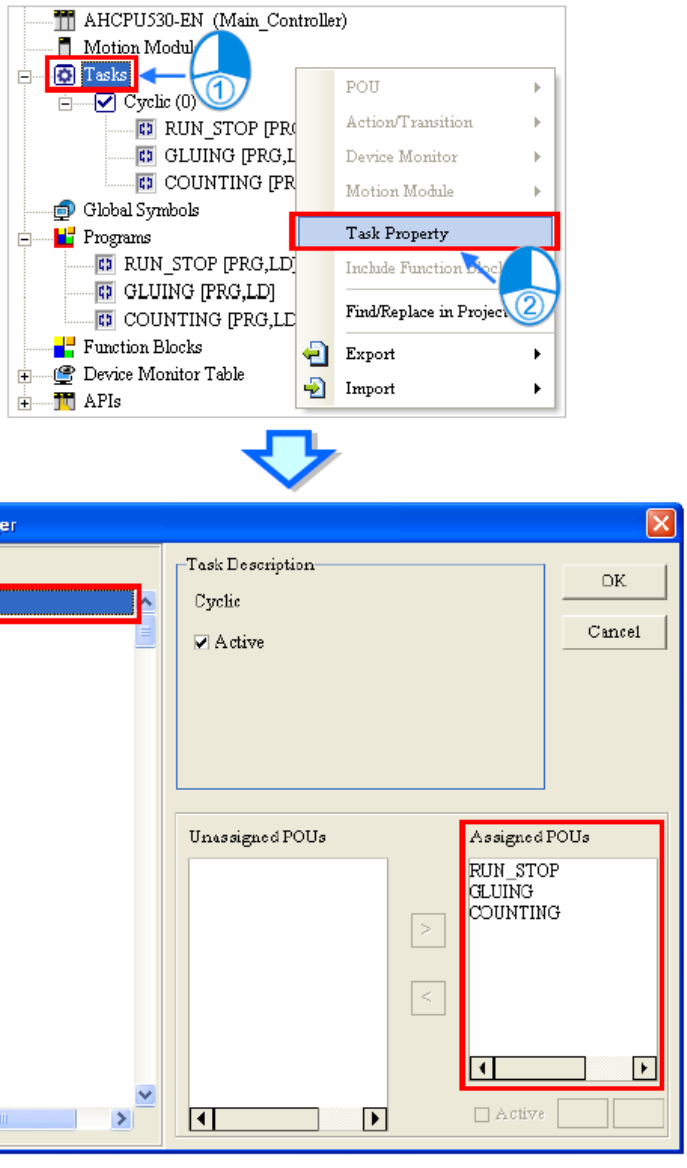
شکل ۵-۴۰ مثال۱- برنامه COUNTING

پس از اتمام ساخت POUها، کاربردر قسمت Task باید ترتیب اجرای آن‌ها را مشخص نمایند.



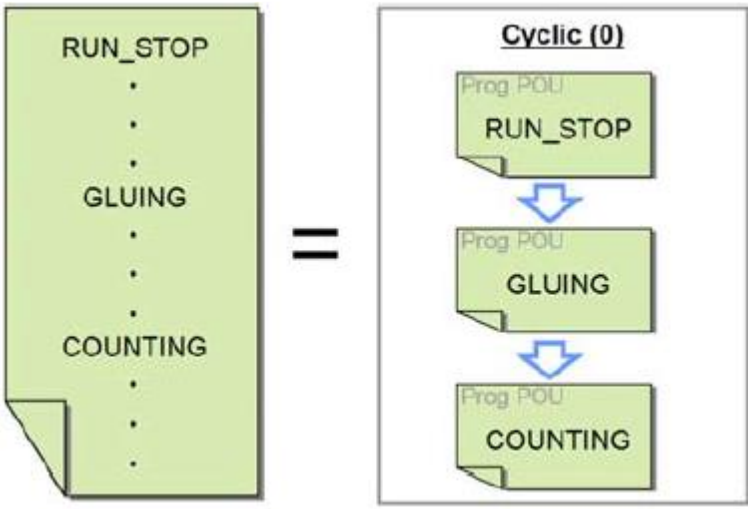
شکل ۵-۴۱ مثال۱- ترتیب اجرای POU

برای تغییر چینش POUها در Task می‌توان با کلیک راست بر روی Task از طریق Task Property اقدام کرد.



شکل ۵-۴۲ مثال۱- تغییر ترتیب اجرای POU

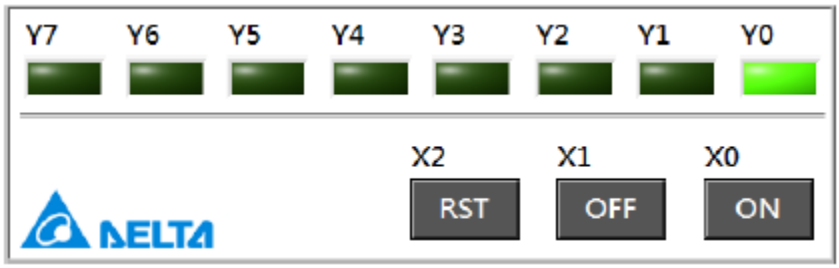
در نهایت بلوک دیاگرام روتین دوره‌ای برنامه به صورت زیر در خواهد آمد.



شکل ۵-۴۳ مثال۱- ترتیب نهایی اجرای POU

۵-۳-۲- مثال وقفه در PLCهای سری DVP

شرح: اگر کلید ON فشرده شد (X0 یک شد) آنگاه Y0 الی Y7 به ترتیب به روز شوند (به ترتیب چراغ روشن جابجا شود). اگر کلید OFF فشرده شد، (X1 یک شود) آنگاه خروجی متوقف و حالت خروجی ذخیره شود. اگر کلید RST فشرده شود (X2 یک شود) آنگاه خروجی ریست شده و Y0 الی Y7 دوباره به روز می‌شوند.



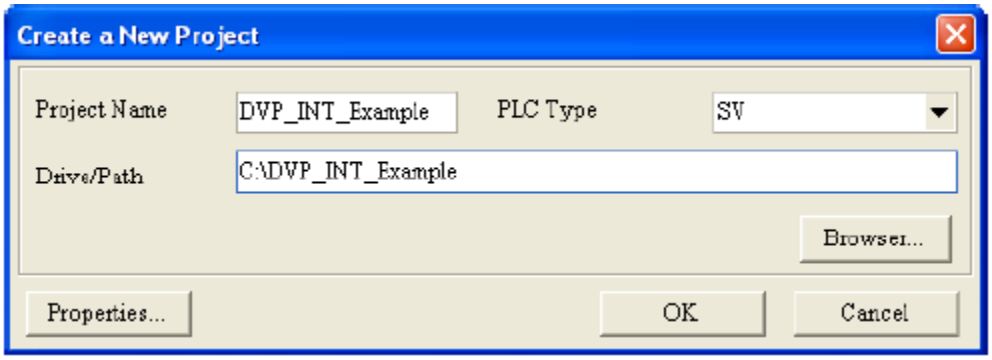
شکل ۵-۴۴ شرح مثال ۲

می‌خواهیم از PLCهای مدل DVP-SV استفاده کرده و مطابق جدول زیر چهار POU برای آن تعریف کنیم.

جدول ۵-۴: تعریف POU برای مثال ۲

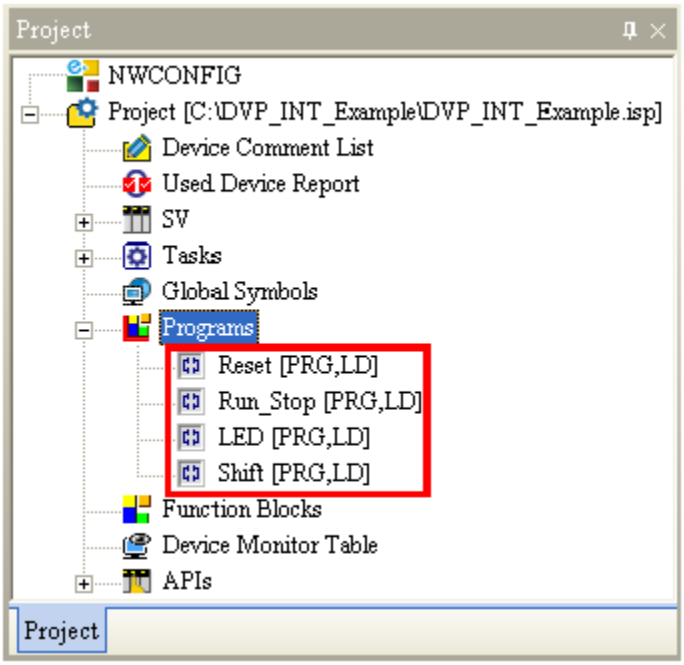
نام POU	نحوه کارکرد برنامه
Run_Stop	۱- صفر کردن پرچم عملکرد M0، انتقال مقدار ۱ به رجیستر خروجی D0 (یعنی بیت ابتدایی آن یک شود) و فعال کردن وقفه ۲- زمانی که X0 از حالت OFF به حالت ON تغییر وضعیت می‌دهد، M0 یک می‌شود. ۳- زمانی که X1 از حالت OFF به حالت ON تغییر وضعیت می‌دهد، M0 صفر خواهد شد.
LED	به روز کردن Y0 الی Y7 بر اساس مقدار رجیستر خروجی D0
Shift (Rotate)	D0 داده‌هایش هر نیم ثانیه یک بیت به سمت چپ انتقال (شیفت) پیدا می‌کند. (بیت آخر جایگزین بیت اول می‌شود)
Reset	انتقال مقدار ۱ به D0 که در نتیجه آن بیت صفر در D0 فعال می‌شود. POU بعد از وقفه خارجی I201 اجرا می‌شود. I201 زمانی تحریک می‌شود که لبه بالارونده X2 اتفاق بیافتد.

ابتدا پروژه‌ای بر روی مدل‌های DVP-SV تعریف می کنیم.



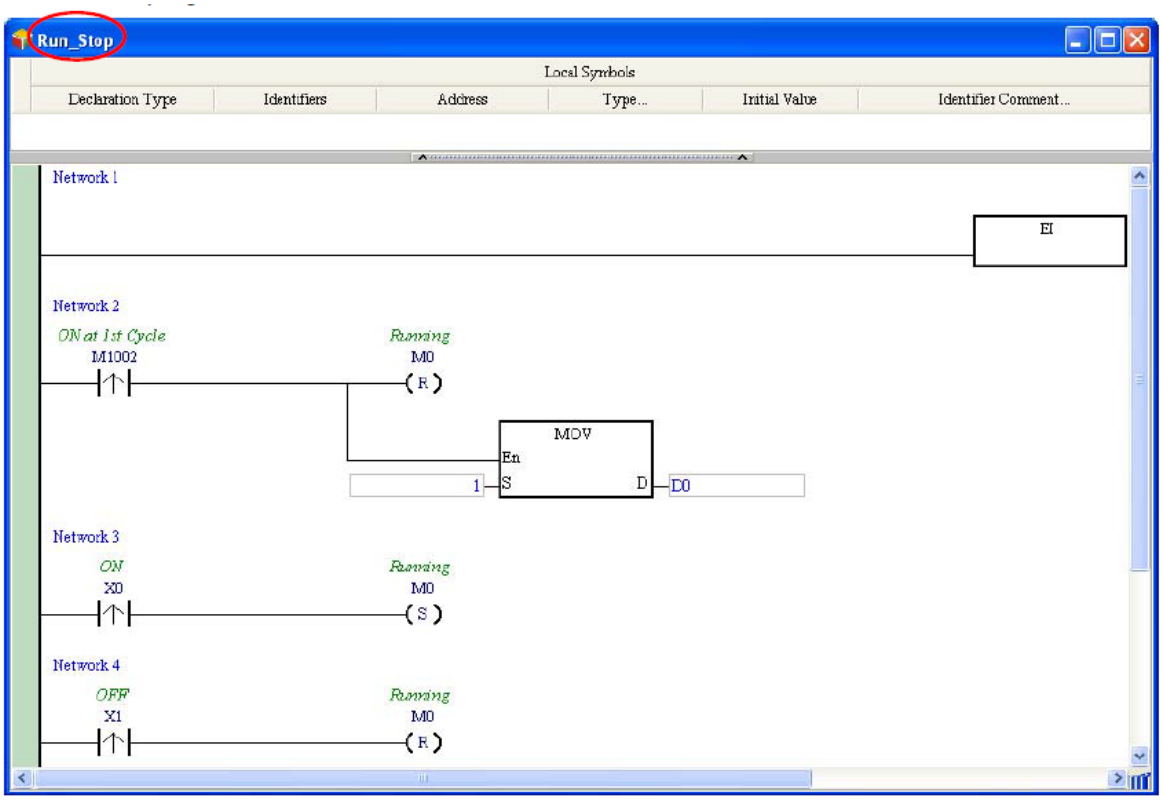
شکل ۵-۴۵ مثال۲ – تعریف پروژه بر روی DVP-SV

پس از انجام تنظیمات اولیه و ارتباطی سیستم، در قسمت Programs در بخش مدیریت پروژه، چهار POU خواسته شده را می‌سازیم و عملکرد دوره‌ای را به آن‌ها اختصاص می‌دهیم.

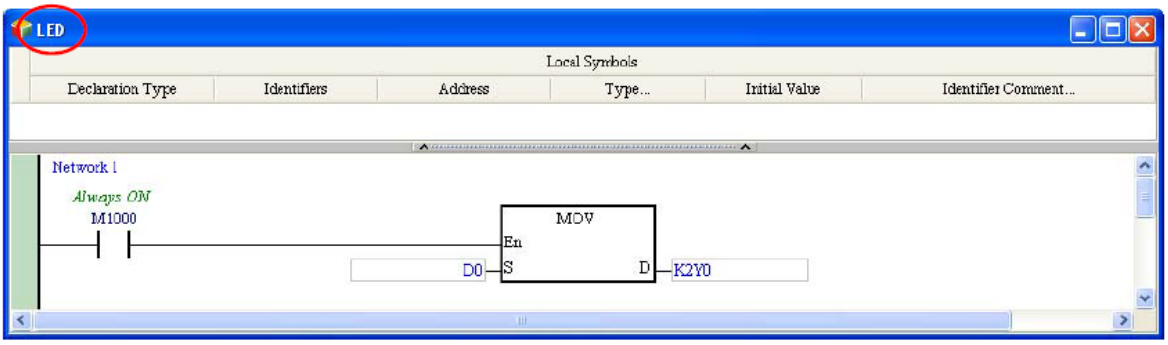


شکل ۵-۴۶ مثال۲ – ساخت چهار POU خواسته شده

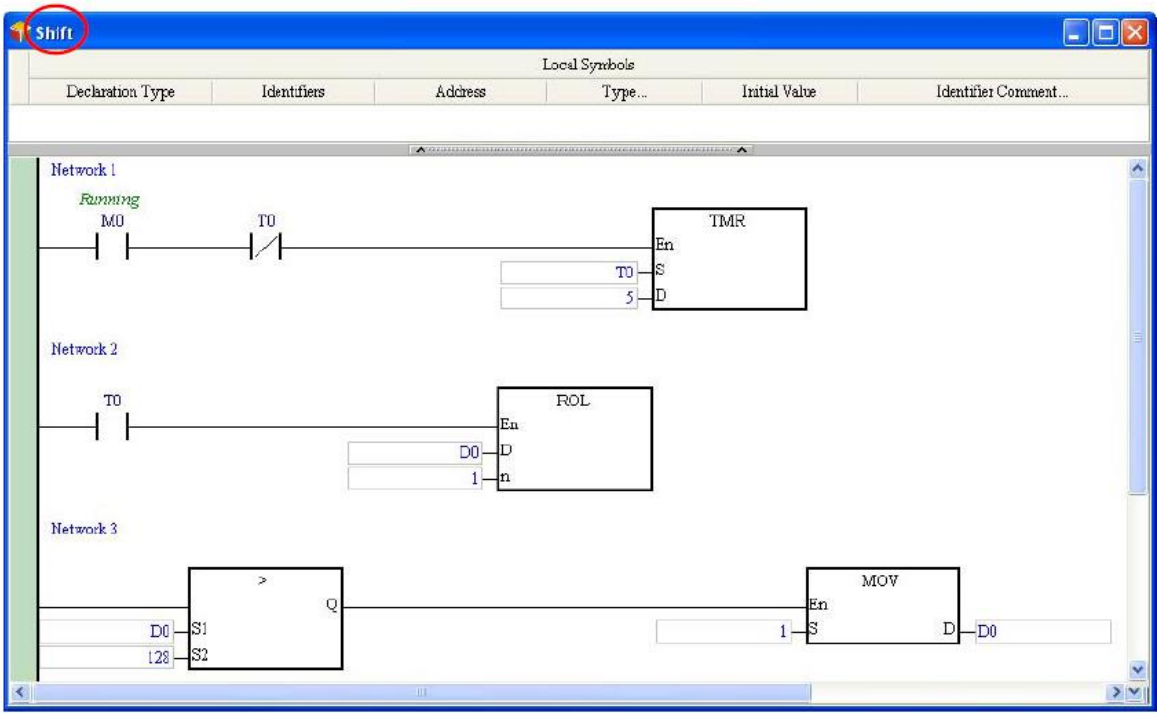
برنامه‌های نوشته شده به صورت زیر است:



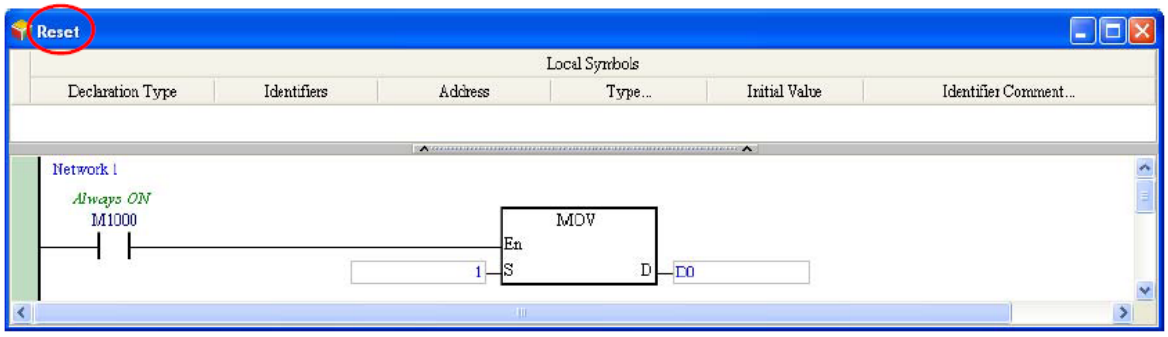
شکل ۵-۴۷ مثال۲ – برنامه Run_Stop



شکل ۵-۴۸ مثال۲ – برنامه LED

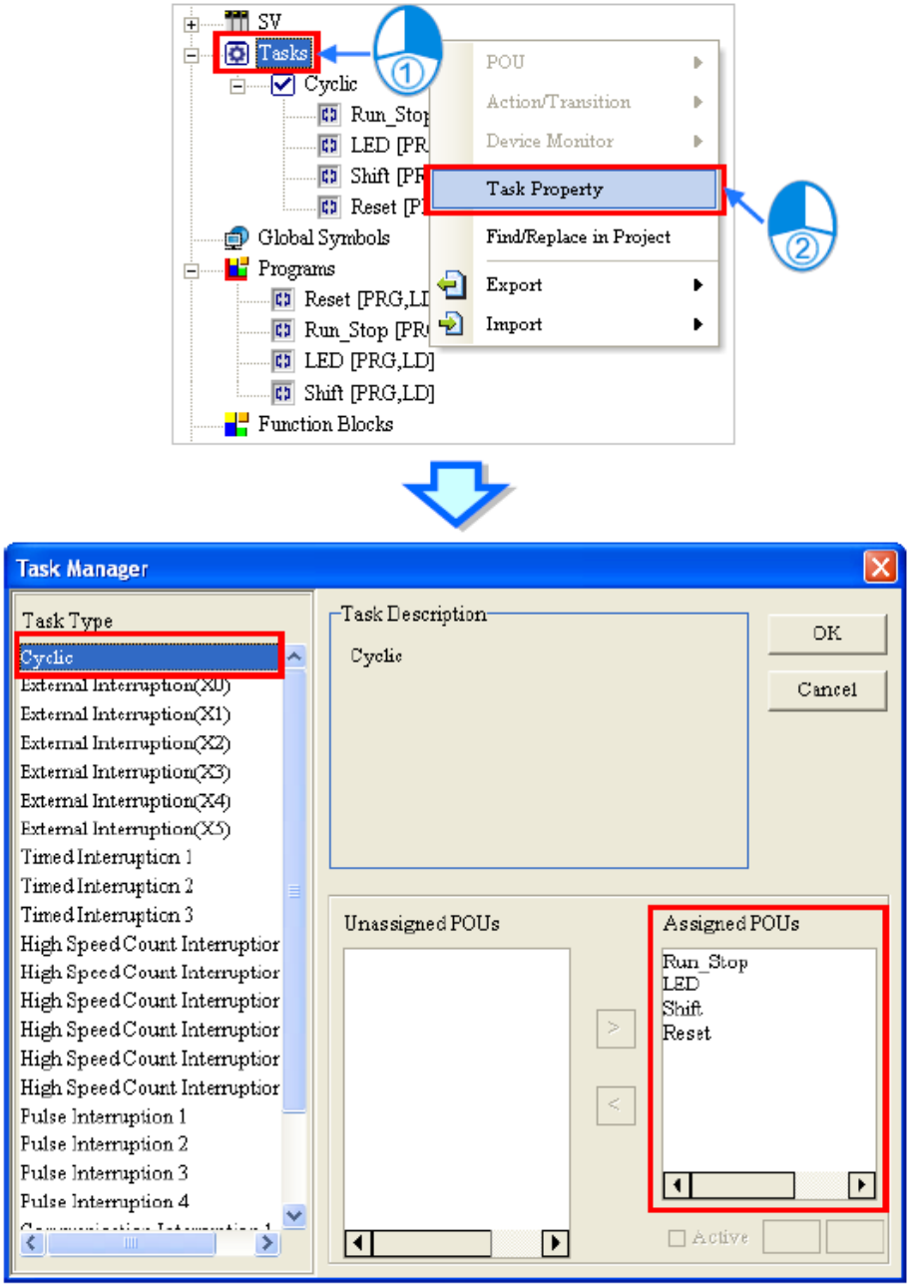


شکل ۵-۴۹ مثال۲ – برنامه Shift



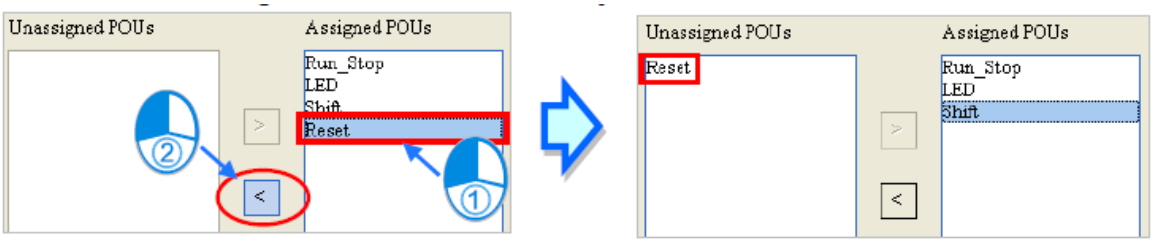
شکل ۵-۵۰ مثال۲ – برنامه Reset

پس از نوشته شدن برنامه‌ها به Task Manager رفته و عملکرد Cyclic را انتخاب می‌کنیم.



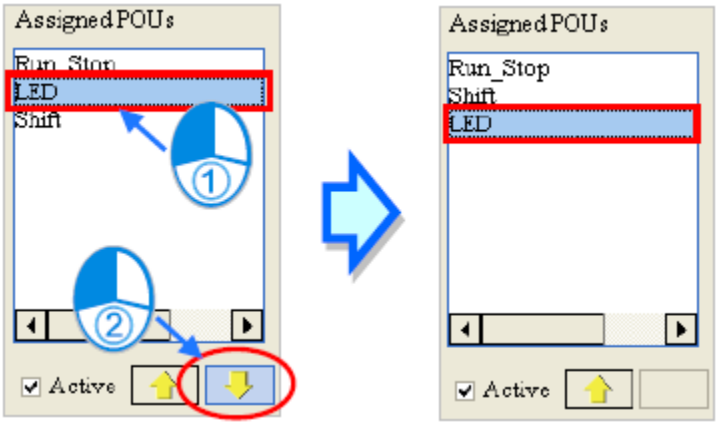
شکل ۵-۵۱ مثال ۲ – مشاهده POU اختصاص داده شده به عملکرد دوره ای صفر

با توجه به آنکه Reset بعد از وقفه خارجی اتفاق می‌افتد، آن را از لیست Assigned POU حذف می-کنیم.



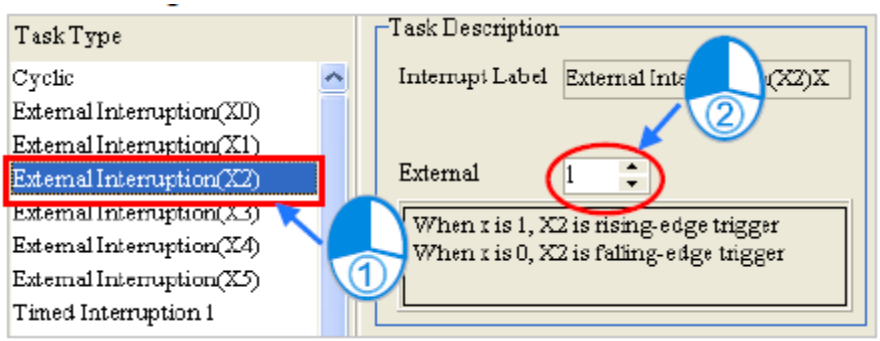
شکل ۵-۵۲ مثال۲ – حذف Reset از عملکرد دوره ای صفر

ترتیب اجرای برنامه‌های دوره‌ای را نیز به صورت Run_Stop، Shift و سپس LED تنظیم می‌کنیم.



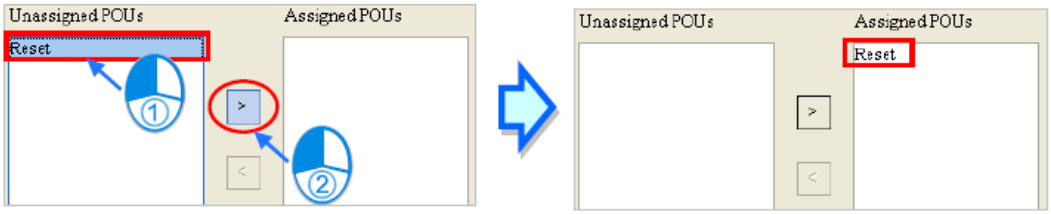
شکل ۵-۵۳ مثال۲ – تنظیم ترتیب اجرای POU در عملکرد دوره ای صفر

حال باید عملکرد وقفه را تعیین کنیم. برای اینکار External Interruption (X2) را در Task Type انتخاب می‌کنیم و در قسمت External آن مقدار ۱ را قرار می‌دهیم. External Interruption (X2) زمانی تحریک می‌شود که لبه بالا رونده سیگنال X2 اتفاق بیافتد.



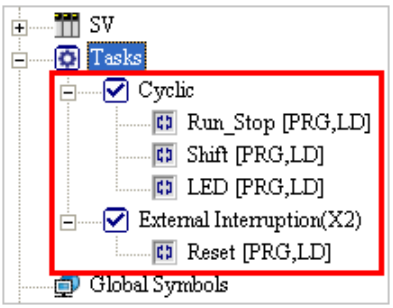
شکل ۵-۵۴ مثال۲ – اختصاص عملکرد External Interruption (X2) به Reset– قسمت ۱

برای اختصاص External Interruption (X2) به Reset آن را در قسمت Assigned POU قرار می-دهیم.



شکل ۵-۵۵ اختصاص عملکرد External Interruption (X2) به Reset– قسمت ۲

پس از کلیک بر روی OK تنظیمات مورد نظر اعمال خواهد شد. در این مرحله تنظیمات برنامه به صورت زیر در خواهد آمد و کاربر می‌تواند آن را کامپایل و بر روی PLC بارگزاری نماید.

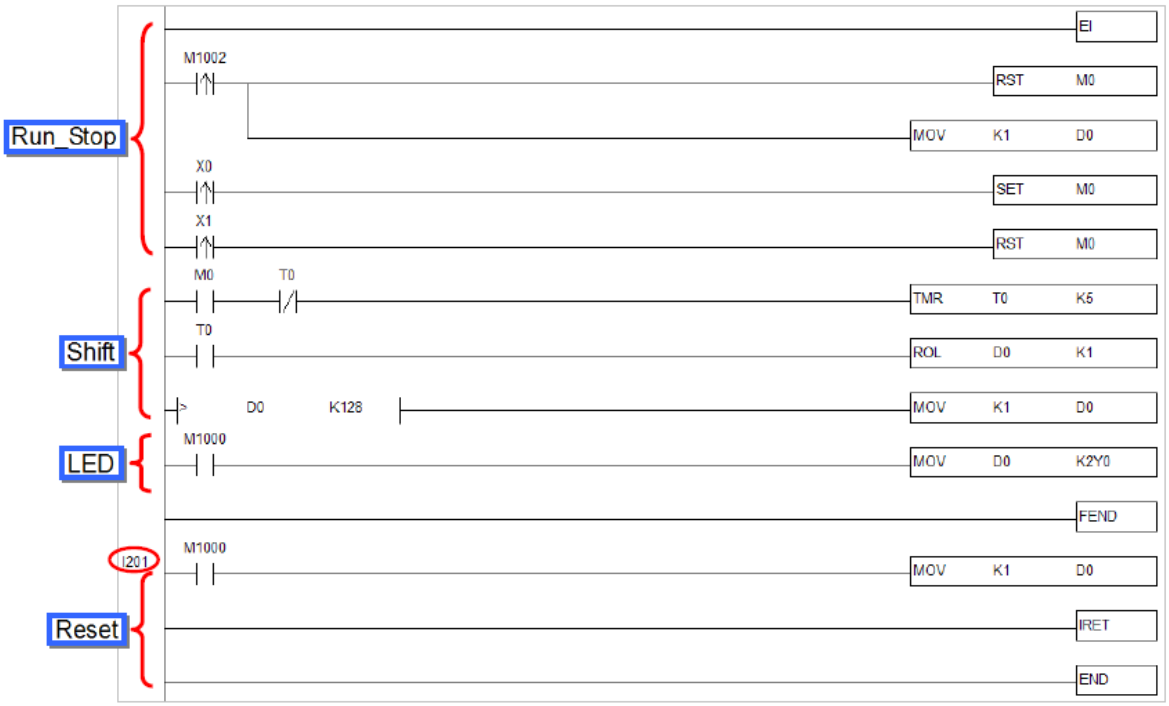


شکل ۵-۵۶ مثال۲ – وضعیت نهایی اختصاص عملکرد به POU

نکته کاربردی:

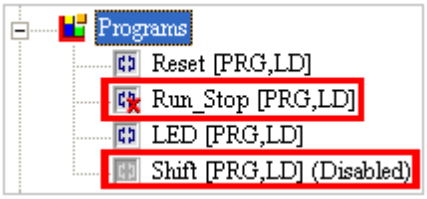
با آنکه پروژه‌های ISPSoft را به صورت مستقیم نمی‌توان در WPLSoft استفاده کرد. ولی می‌توان آن-ها را در PLC بارگزاری و سپس دوباره به وسیله WPLSoft استخراج کرد. در مثال فوق در صورتی که برنامه نوشته شده را در WPLSoft استخراج کنیم، برنامه‌ای مانند برنامه زیر را مشاهده می‌کنیم که در بر گیرنده سه عملکرد دوره‌ای ما به ترتیب Run_Stop، Shift و سپس LED است. وقفه I201 نیز همانند برنامه وقفه Reset عمل می‌کند[۱].

[۱] WPLSoft برنامه ای برای کار با PLCهای کمپانی دلتا است که قبل از ISPSoft مورد استفاده قرار می‌گرفته و دارای قالبی سنتی است.

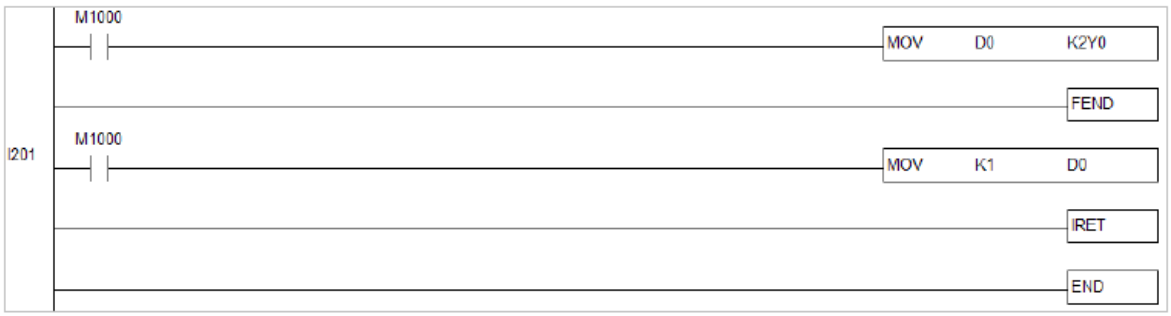


شکل ۵-۵۷ مثال۲ – استخراج برنامه در WPLSoft

برای اینکه روند مورد نظر را تست کنیم می‌توانیم در برنامه اصلی POU مرتبط با Run_Stop را در قسمت Task پاک کرده و Shift را غیرفعال کنیم. در این حالت پس از کامپایل و بارگزاری آن در PLC و استخراج آن به وسیله WPLSoft شاهد برنامه زیر خواهیم بود که دیگر شامل دو قسمت RUN_STOP و Shift نخواهد بود.



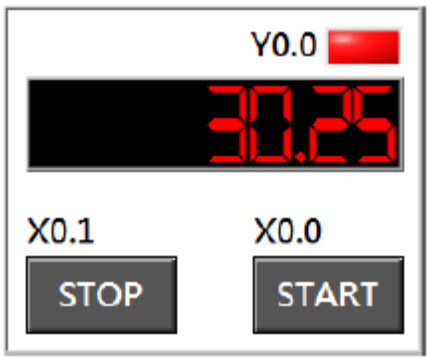
شکل ۵-۵۸ مثال۲ – حذف عملکرد POU های Run_Stop و غیر فعال کردن عملکرد Shift



شکل ۵-۵۹ مثال۲ – نتیجه حذف عملکرد POU های Run_Stop و غیر فعال کردن عملکرد Shift در WPLSoft

۵–۳–۳– مثال وقفه در PLCهای سری AH500

شرح: در صورتی که کلید START فشرده شود (X0.0 یک شود) سیستم شروع به کار خواهد کرد. سپس در صورتی که کلید STOP قبل ۳۰ ثانیه فعال نشود (X0.1 یک نشود)، آنگاه آلارم فعال خواهد شد (Y0.0 یک می‌شود).



شکل ۵-۶۰ شرح مثال ۳

در این مثال از CPU مدل AHCPU530-EN و ورودی/خروجی دیجیتال AH16AP11R-5A استفاده می‌کنیم. می‌خواهیم برنامه ای نوشته که در آن زمان سپری شده بعد START را با تفریق زمان فشرده شدن START از زمان فعلی پیدا کنیم.

شکل ۵-۶۱ شرح نمودار زمانی مثال ۳

توجه شود که برنامه نوشته شده در این مثال منحصر به فرد نبوده (طبیعتا طراح آن دلایل خاصی برای این نوع برنامه نویسی دارد، مثلا نیاز داشتن به مجموع زمان‌های سپری شده پس از شروع به کار PLC که می‌تواند دلیل خوبی برای ذخیره آن و ریست نکردن هرباره آن باشد) و ممکن است بتوان آن را به صورت های متفاوت طراحی کرد.

وقفه زمانی هر ۲۵ میلی ثانیه فعال می‌شود و بازه زمانی سپری شده توسط برنامه شمرده می‌شود. وقفه Timed Intrrupption (0) هر ۲۵ میلی ثانیه تحریک می‌شود و مقدار رجیستر ۳۲ بیتی D0 و D1 را

در مدت زمان فعال بودن PLC یکی افزایش می‌دهد (پس هر واحد در رجیسترها معادل ۲۵ میلی ثانیه است). کاربر می‌تواند به مدت زمان اجرای (فعال بودن) PLC از طریق خواندن ۳۲ بیت رجیستر DO و D1 دستیابی داشته باشد. برای مثال در صورتی که مقدار خوانده شده در این رجیستر برابر با ۲۰۰۰ باشد به معنی گذشتن ۵۰ ثانیه از زمان شروع به کار PLC می‌باشد (0.025 Sec*2000=50 Sec).

پس از آنکه کلید START فشرده شد، مقدار فعلی D0 و D1 در رجیستر D2 و D3 کپی می‌شود. سپس مقدار D0 و D1 همچنان با وقفه افزایش می‌یابد و مقدار D2 و D3 بدون تغییر باقی می‌ماند. با تفریق این دو مقدار (که آن را در D4 و D5 ذخیره می‌کنیم) می‌توان به مدت زمان سپری شده پس از فشرده شدن START برسیم.

سپس هرگاه زمان سپری شده پس از فشرده شدن START از سی ثانیه گذشت و کلید Stop فشرده نشد آلارم را فعال می‌کنیم.

در این پروژه پنج POU در نظر گرفته شده است که به صورت زیر می‌باشند.

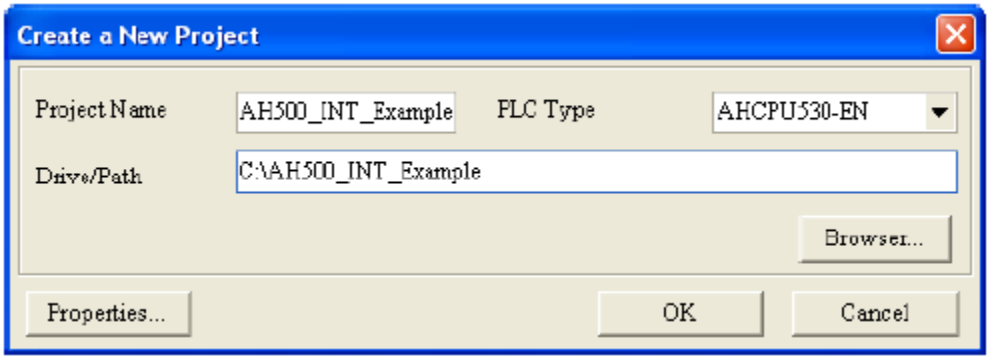
جدول ۵-۵: مثال ۳ – POU های پروژه

نام POU	نحوه کارکرد برنامه
INT_Timer	مقدار رجیستر ۳۲ بیتی D0 و D1 هر ۲۵ میلی ثانیه یک واحد اضافه می‌شود. (POU به (0) Timed Intrruption تخصیص داده شده است)
Initialize	پرچم های M0 و M1 ریست می‌شوند، رجیسترهای D0 تا D7 پاک شده و وقفه فعال می‌شود. در طی سیکل اول اجرای برنامه‌های PLC، POUهای تخصیص داده شده به Cyclic (0) فعال و برنامه‌های تخصیص داده شده به (1) Cyclic که Initialaizeهم جزو آن است غیر فعال می‌شوند. در نتیجه Initialize تنها در سیکل ابتدایی اجرا خواهد شد.
Control	زمانی که X0.0 فعال می‌شود، پرچم M0 فعال و مقدار D0 و D1 به D2 و D3 کپی می‌شوند. زمانی که X0.1 فعال می‌شود یا سرریز[1] رخ می‌دهد، M0 ریست می‌شود.

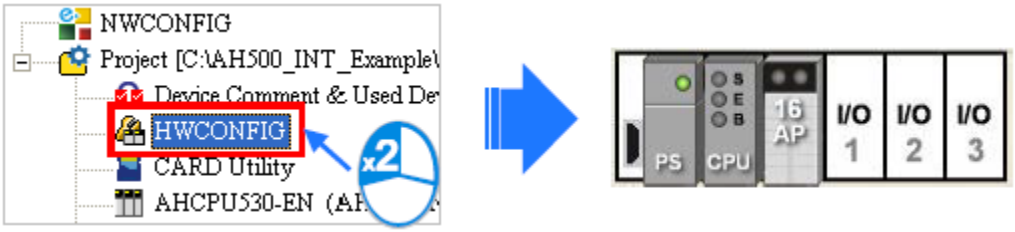
[1] Overflow

Time_CHK	زمان سپری شده پس از فشرده شدن کلید START با تفریق D2 و D3 از D0 و D1 محاسبه و در D4 و D5 ذخیره می‌شود. اگر این مقدار کمتر از صفر بود، پرچم سرریز یعنی M0 فعال می‌شود.
Signal	اگر زمان سپری شده بیش از ۳۰ ثانیه باشد (یعنی D4 و D5 بیشتر از ۱۲۰۰ باشند، چرا که 0.025 Sec*1200=30 Sec) و یا سرریز رخ داده باشد، آلارم Y0.0 فعال می‌شود.

پروژه‌ای با استفاده از CPU مدل AHCPU350-EN می‌سازیم و تنظیمات سخت افزاری آن را انجام می‌دهیم.

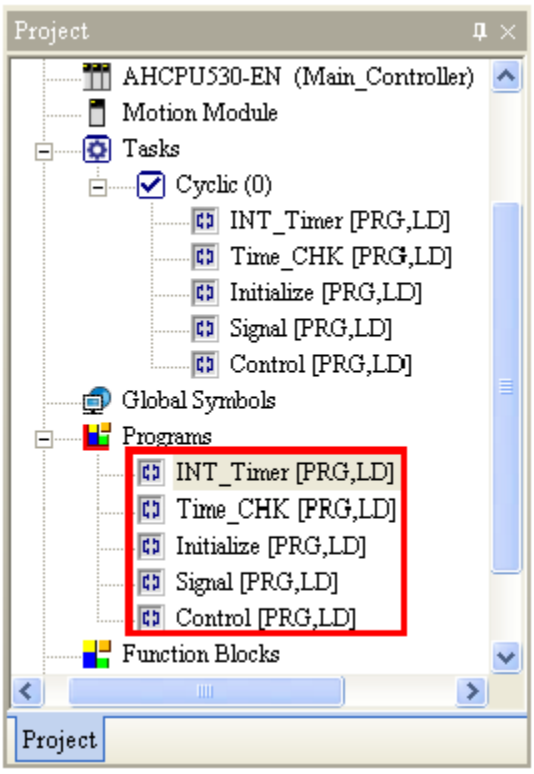


شکل ۵-۶۲ مثال ۳ – ساخت پروژه با AHCPU350-EN



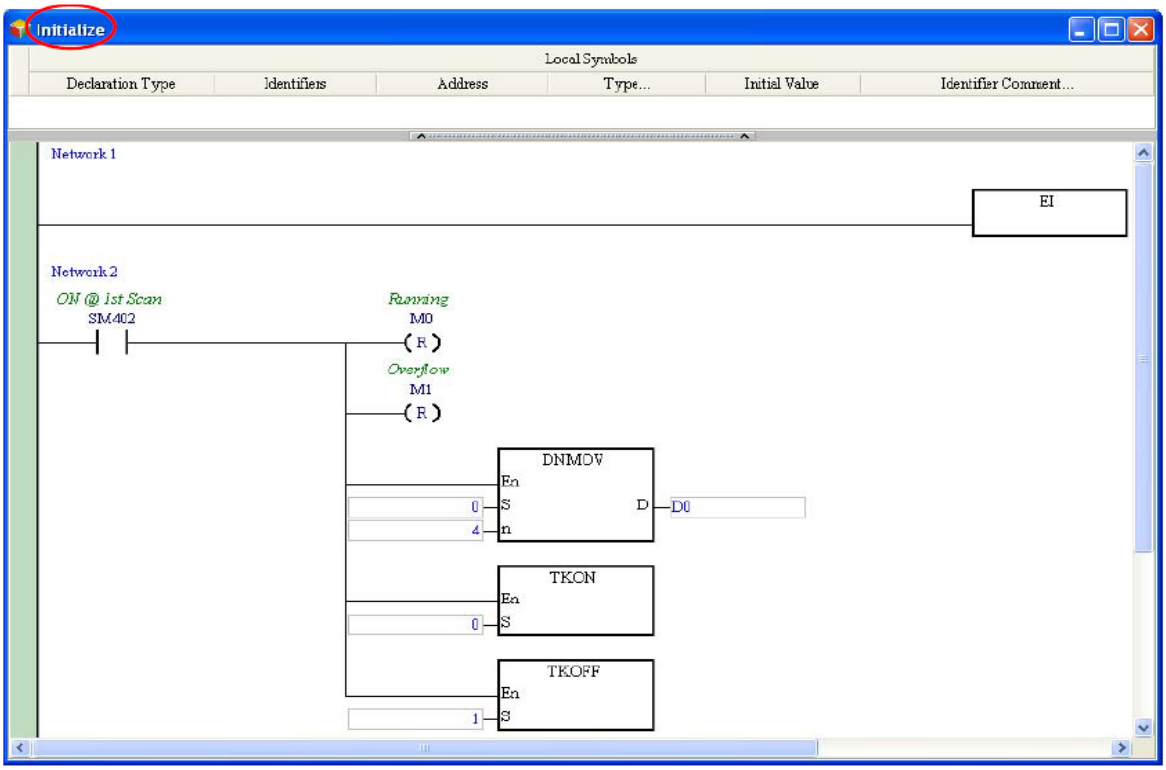
شکل ۵-۶۳ مثال ۳ – تنظیمات سخت افزاری

در ادامه پنج POU شرح شده را ساخته و در حالت پیش فرض آن‌ها را به Cyclic (0) تخیص می‌دهیم.

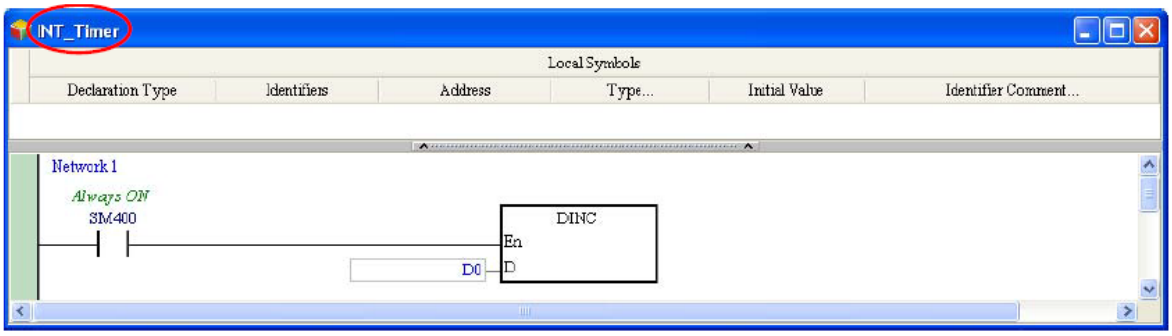


شکل ۵-۶۴ مثال ۳ – ساخت پنج POU با عملکرد پیشفرض Cyclic (0)

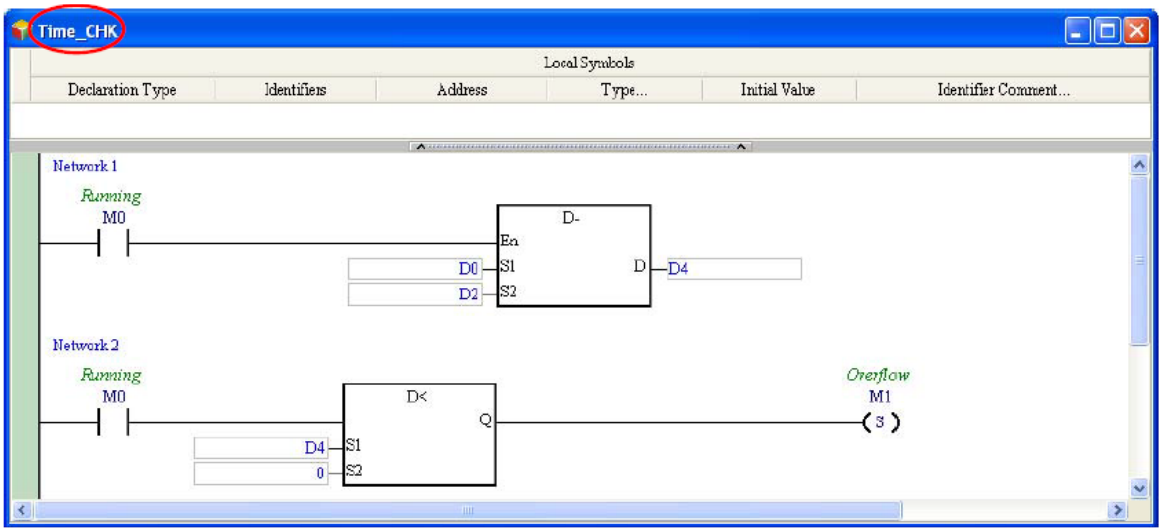
برنامه‌های نوشته شده در هر POU در زیر آمده است.



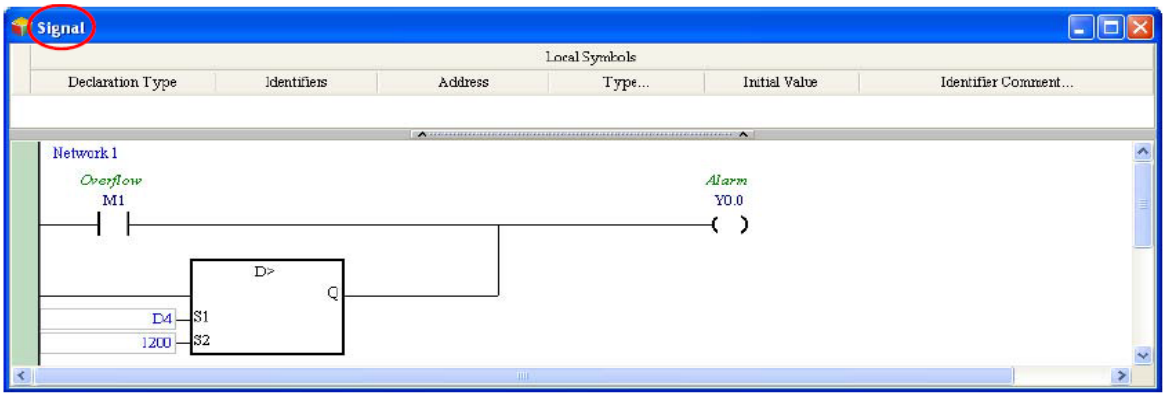
شکل ۵-۶۵ مثال ۳ – برنامه Initialize



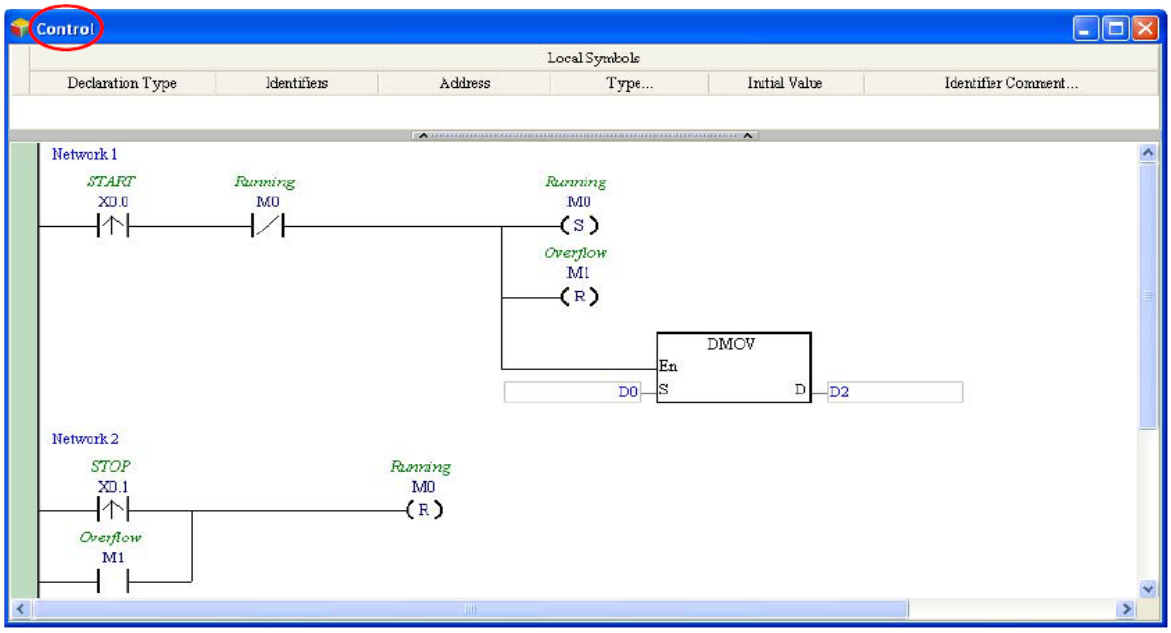
شکل ۵-۶۶ مثال ۳ – برنامه INT-Timer



شکل ۵-۶۷ مثال ۳ – برنامه Time_CHK

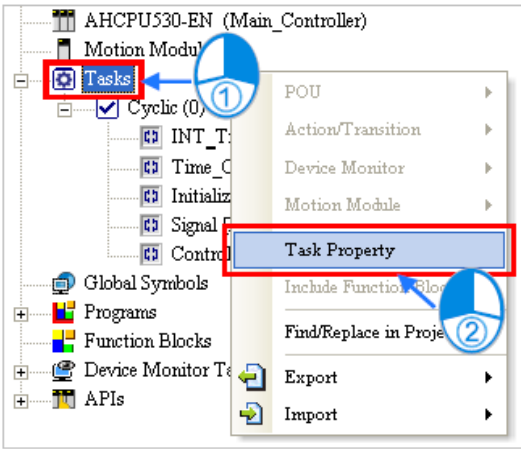


شکل ۵-۶۸ مثال ۳ – برنامه Signal



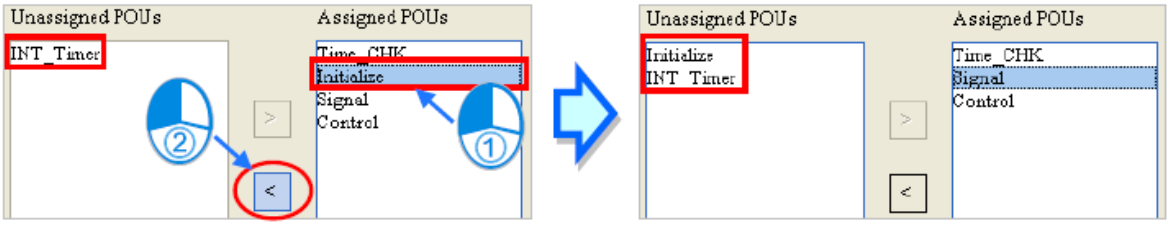
شکل ۵-۶۹ مثال ۳ –Control

در این قسمت برای تخصیص مناسب عملکردها با کلیک راست بر روی Task گزینه Task Property را انتخاب کرده تا پنجره Task Manager باز شود.



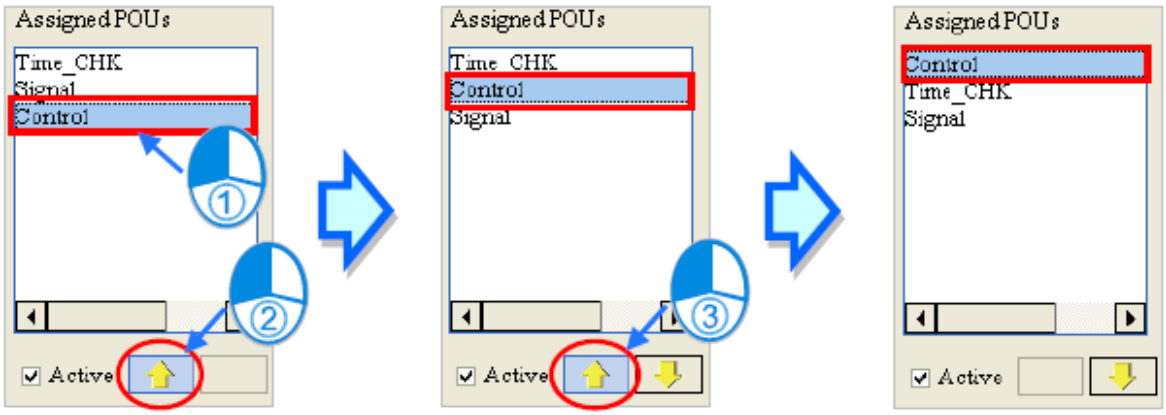
شکل ۵-۷۰ مثال ۳ – تنظیمات مربوط به تخصیص عملکرد POU

برای آنکه بتوانیم Cyclic (0) را فعال کنیم، آن را انتخاب و گزینه Active را فعال می‌کنیم. با توجه به آنکه باید به INT_Timer وقفه زمانی Timed Intrruption (0) تخصیص داده شود و به Initialize نیز باید Cyclic (1) تخصیص داده شود، این دو POU را از قسمت Assigned POU مربوط به Cyclic (0) حذف می‌کنیم.



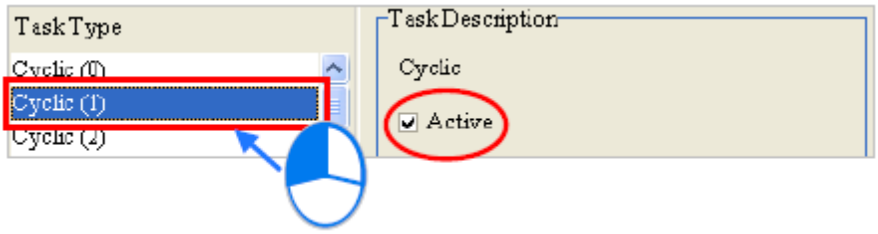
شکل ۵-۷۱ مثال ۳ – حذف Initialize و INT_Timer از عملکرد دوره ای صفر

در ادامه ترتیب اجرای POUهای Cyclic (0) را به ترتیب Control، Time_CHK و Signal قرار می-دهیم.



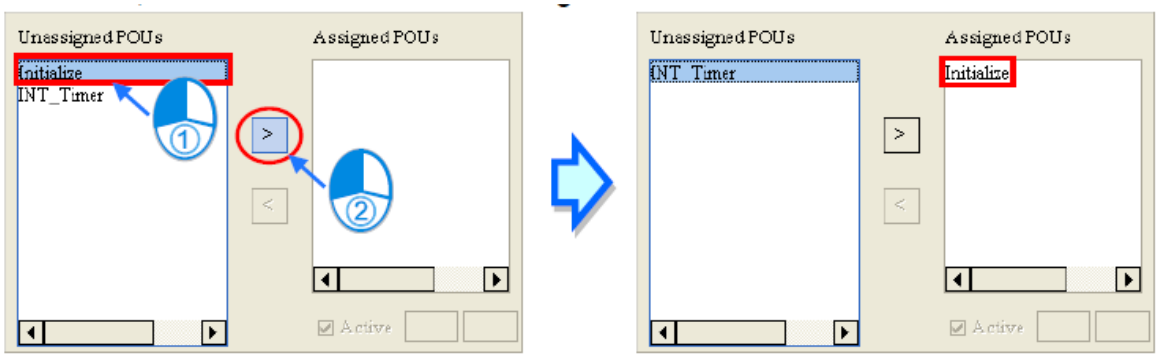
شکل ۵-۷۲ مثال ۳ – تنظیم ترتیب اجرای POU در عملکرد دوره ای صفر

در ادامه Cyclic (1) را در قسمت Task Type انتخاب و آن را نیز Active می‌کنیم.



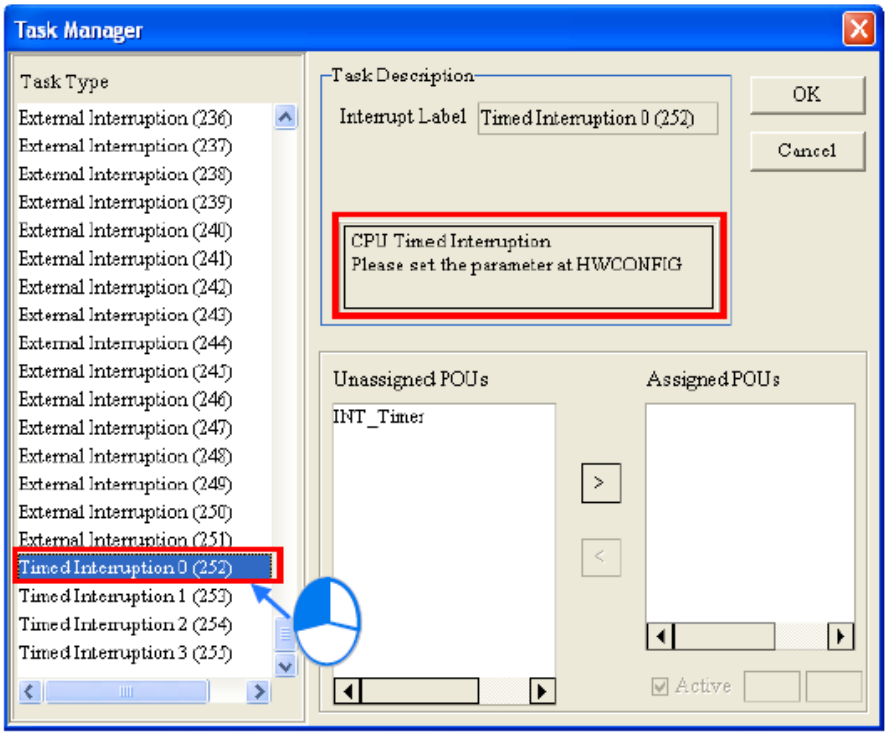
شکل ۵-۷۳ مثال ۳ – انتخاب عملکرد دوره ای یک

سپس Initialize را از بخش Unassigned POU به Assigned POU منتقل می‌کنیم.



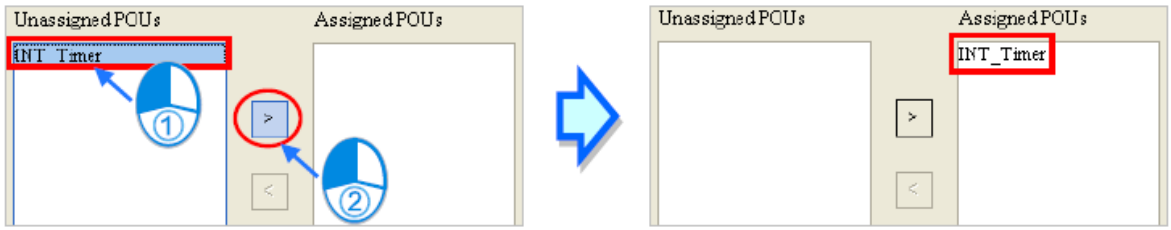
شکل ۵-۷۴ مثال ۳ – اختصاص عملکرد دوره ای یک به Initialize

حال Timed Intrupption 0 را در Task Type انتخاب می کنیم. توضیحات این عملکرد نشان می‌دهد که کاربر برای فعال کردن و تنظیم این وقفه زمانی باید از طریق HWCONFIG اقدام کند.



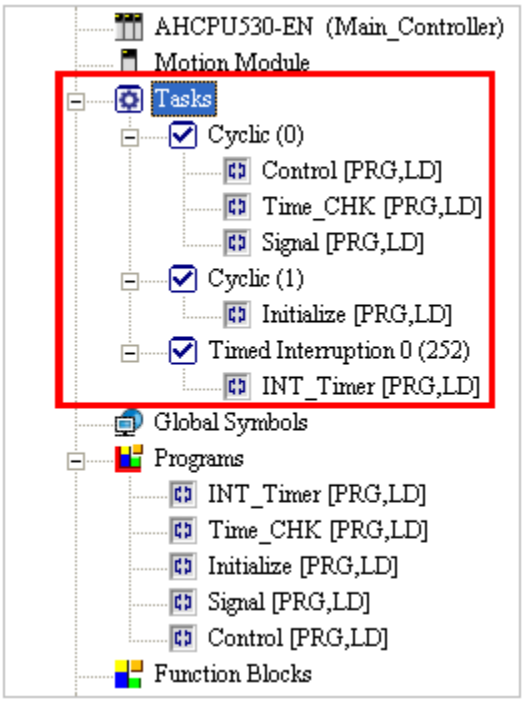
شکل ۵-۷۵ مثال ۳ – انتخاب عملکرد وقفه Timed Intrupption 0

در این مرحله INT_Timer را به بخش Assigned POU در Timed Intrruption 0 منتقل می‌کنیم.



شکل ۵-۷۶ مثال ۳ – اختصاص عملکرد Timed Intrupption 0 به INT_Timer

پس از تایید، تنظیمات عملکردها به صورت زیر خواهد بود.

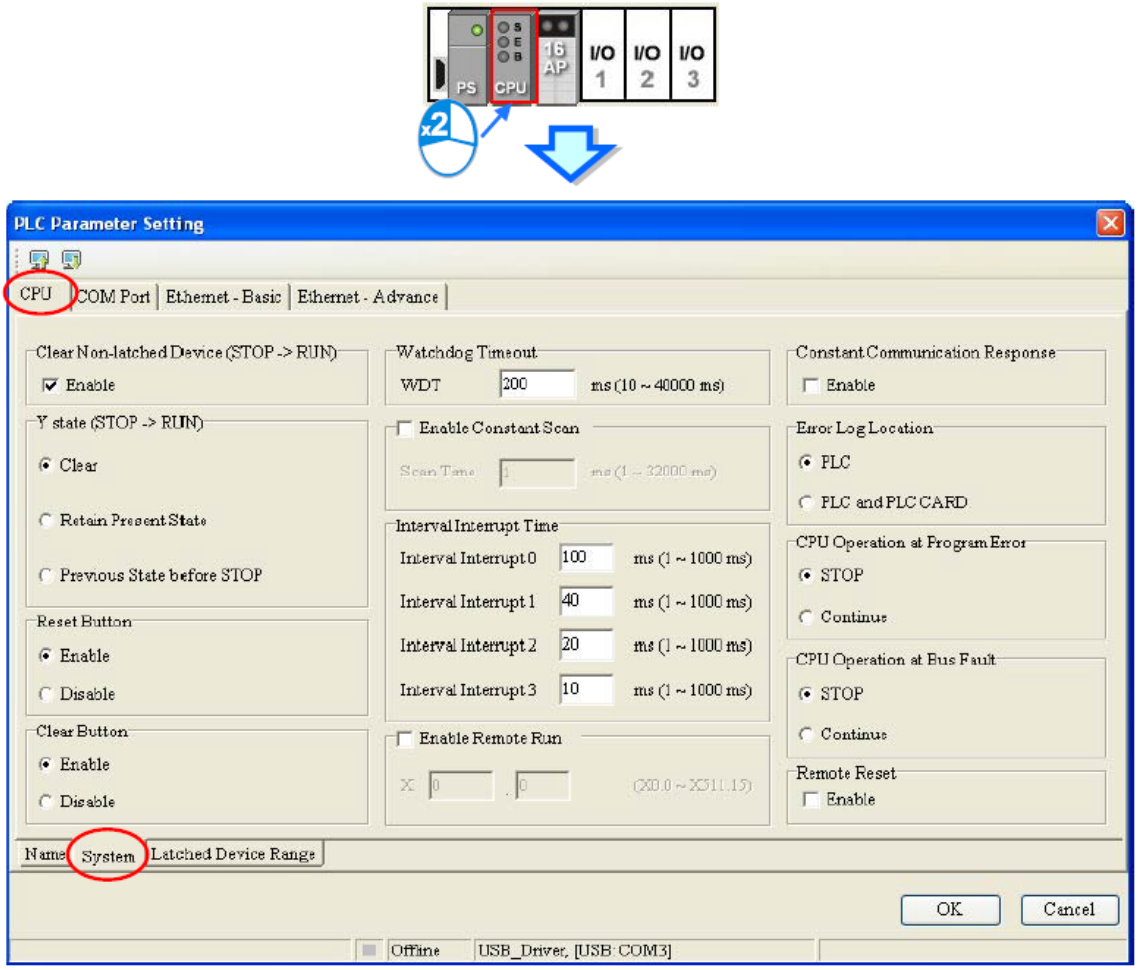


شکل ۵-۷۷ مثال ۳ –وضعیت نهایی اختصاص عملکرد به POU

حال برای تنظیمات مربوط به وقفه زمانی به HWCONFIG می‌رویم.

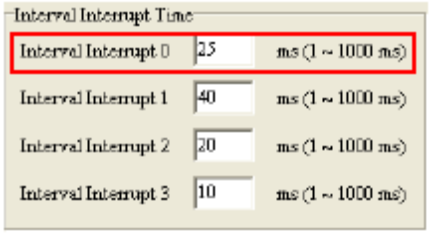
شکل ۵-۷۸ مثال ۳– تنظیمات سخت افزاری برای وقفه زمانی

پس از باز شدن صفحه تنظیمات سخت افزاری، دوبار بر روی ماژول CPU کلیک کرده تا صفحه PLC Parameters Setting باز شود. سربرگ CPU و سپس زیربرگ System را انتخاب می‌کنیم.



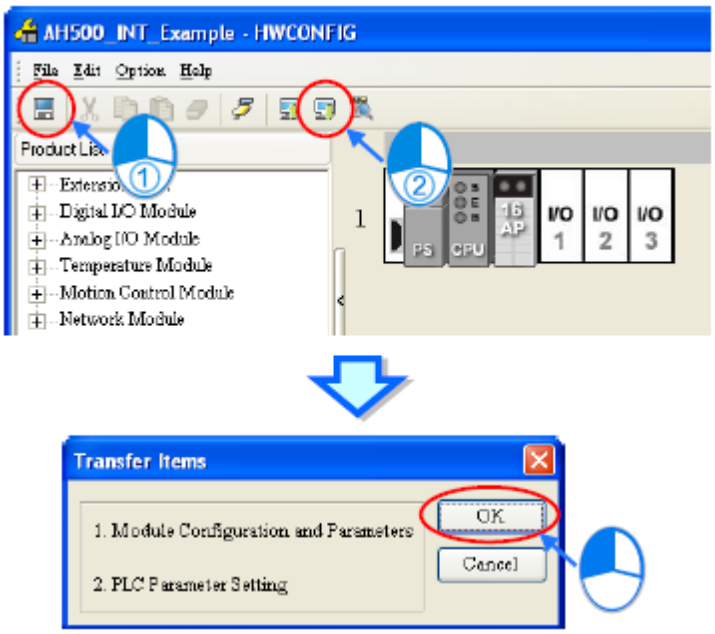
شکل ۵-۷۹ مثال ۳- تنظیمات سخت افزاری CPU

در قسمت Interval Intrrupt Time در مقابل Interval Intrrupt 0 عدد ۲۵ را تایپ می‌کنیم (توجه کنید که مقدار صفر به معنای غیر فعال بودن وقفه و دیگر اعداد مدت زمان اعمال وقفه بر حسب میلی ثانیه را مشخص می‌کند)



شکل ۵-۸۰ مثال ۳- تنظیم مدت زمان وقفه زمانی

در این مرحله تنظیمات را ذخیره و در PLC دانلود می‌کنیم.



شکل ۵-۸۱ مثال ۳- بارگزاری تنظیمات سخت افزاری وقفه زمانی در PLC

در نهایت نیز پس از کامپایل برنامه‌ها را در PLC دانلود می کنیم.

فصل ۶– سیمبول ها

یکی از سختی‌های همیشگی برنامه نویسی PLC مخصوصا در پروژه‌های بزرگ مدیریت آدرس‌دهی حافظه و ورودی/خروجی‌ها است. برای حل این مشکل سیمبول[۱] (نماد) ها مطرح شدند که با استفاده از آن‌ها و تخصیص آدرس به سیمبول‌ها می توان مشکلات مرتبط با آدرس دهی را حل کرد. سیمبول‌ها که جایگزین آدرس دهی مستقیم در برنامه نویسی PLCها شده‌اند، موجب خوانا تر شدن برنامه ها شده و امکان توسعه برنامه‌ها را به راحتی فراهم می‌آورند.

برای اینکه کاربرد سیمبول ها را بهتر به ذهن بسپارید، به این مثال توجه کنید: ممکن است لازم باشد وضعیت موتوری در حافظه M0 ذخیره شود و برنامه نیز آنقدر گسترده باشد که امکان فراموشی این موضوع وجود داشته باشد، در این حالت با اختصاص سیمبول "Motor Status" به حافظه M0، هرجای برنامه که از M0 استفاده شده باشد عبارت Motor Status نمایش داده خواهد شد و اینگونه امکان تحلیل برنامه به راحتی امکان پذیر خواهد بود.

۶–۱– داده ها و سیمبول ها

برای آنکه بتوان از سیمبول‌ها استفاده کرد ابتدا باید آن ها را تعریف کرد. دو نوع سیمبول وجود دارد، سیمبول‌های سراسری[۲] و محلی[۳]. سیمبول‌های سراسری در برنامه تمامی POUهای پروژه قابل استفاده هستند ولی سیمبول‌های محلی تنها در برنامه POUهایی که در آن‌ها تعریف شده‌اند قابل استفاده هستند. بنابراین شناسه یا نماد سیمبول‌های محلی در POUهای مختلف می‌تواند یکسان باشد و عملکرد آن‌ها خللی در کار یکدیگر ایجاد نمی‌کند. البته در صورتی که شناسه تعریف شده برای سیمبولِ متغیری محلی مشابه سیمبول متغیری سراسری تعریف شود، POU اولویت را برای سیمبول محلی در نظر می-گیرد.

در تعریف شناسه سیمبول‌ها باید به موارد زیر توجه کرد.

- شناسه سیمبول‌ها می‌توانند حداکثر از ۳۰ کاراکتر تشکیل شوند.

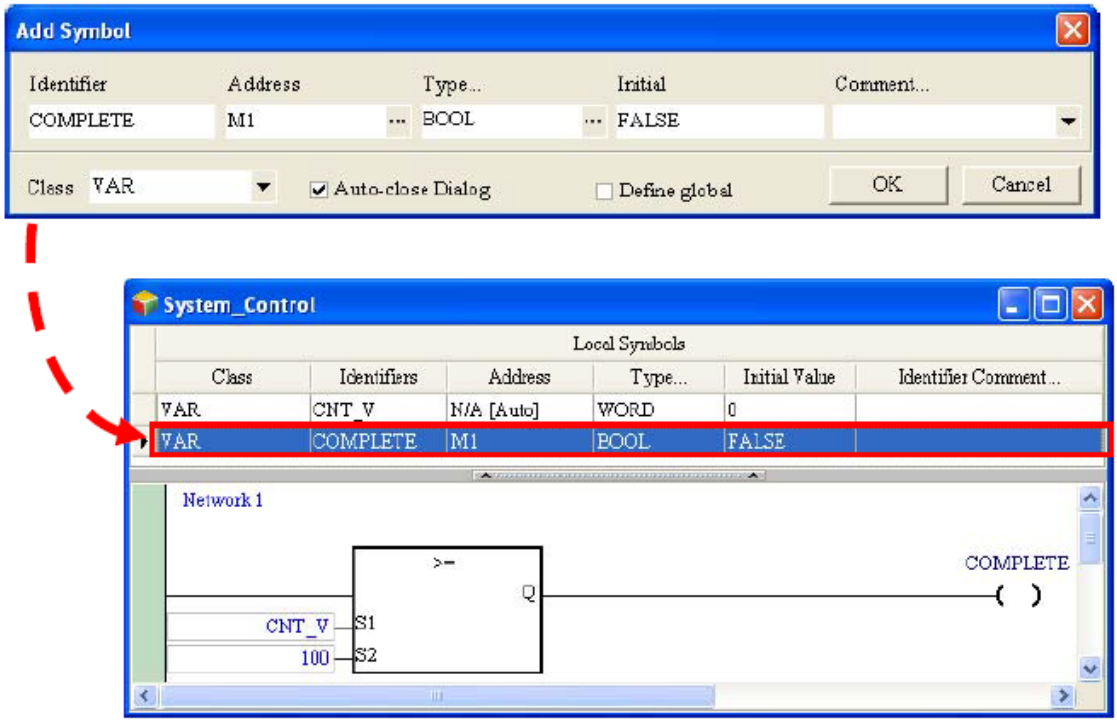
---

[۱] Symbol

[۲] Global symbols

[۳] local symbols

- شناسه ها نباید مشابه نام‌های سیستمی مانند نام حافظه‌ها و یا پورت‌ها باشند، ولی می‌توانند آن‌ها را در ترکیب با دیگر کاراکترها به کار برد. مثلا "M0" که نمایانگر حافظه بیتی صفر است را نمی‌توان به عنوان شناسه سیمبول به کار برد ولی "_M0" ویا "Copy_from_M0" را می‌توان به عنوان شناسه سیمبول استفاده کرد.
- استفاده از فاصله در شناسه سیمبول غیر مجاز است.
- استفاده از UnderLine مجاز است ولی نباید آن‌را به صورت متوالی و یا در انتهای شناسه سیمبول به کار برد.
- نمادهای دیگر مانند *، &، ? و ... را نباید در شناسه سیمبول به کار برد.



شکل ۶-۱ استفاده از سیمبول ها در برنامه

۶-۱-۱- کلاس‌های سیمبول

از لحاظ عملکرد، سیمبول‌ها را می‌توانیم در پنج کلاس طبقه بندی کنیم. در ادامه به معرفی این پنج کلاس می‌پردازیم.

- سیمبول‌های عمومیِ VAR

اینگونه سیمبول‌ها برای عملیات‌های عمومی در برنامه نویسی مورد استفاده قرار می‌گیرند. ویژگی‌های آن نیز به نوع داده، حافظه و یا پورت ورودی/خروجی تعریف شده برای آن بستگی دارد.

- سیمبول‌های نگهدارِ VAR_RETAIN

این کلاس تنها توسط PLCهای سری AH500 پشتیبانی می‌شوند. کاربران از این کلاس در FB نمی‌توانند استفاده کنند. عملکرد این سیمبول‌ها مشابه سیمبول‌های عمومی VAR است با این تفاوت که حافظه‌هایی که به VAR_RETAIN اختصاص داده می‌شوند (به صورت اتوماتیک) نگهدار هستند و داده‌ها را در خود حفظ می‌کنند، در نتیجه داده‌های اینگونه سیمبول‌ها پس از قطع برق PLC نیز نگه داشته خواهند شد. توجه شود از آنجایی که کاربر نمی‌تواند نوع حافظه را در این حالت تعیین کند، نوع داده نمی‌تواند زمان سنج، شمارنده و یا پله[1] باشد.

- سیمبول VAR_INPUT

این سیمبول برای ورودی FB به صورت محلی مورد استفاده قرار می‌گیرد.

- سیمبول VAR_OUTPUT

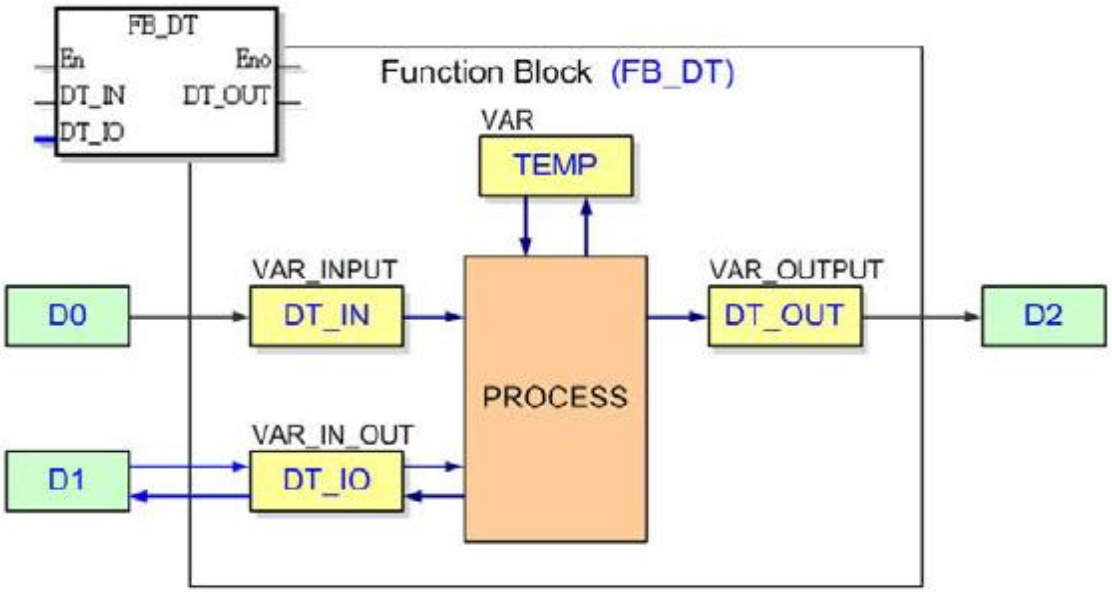
این سیمبول برای خروجی FB به صورت محلی مورد استفاده قرار می‌گیرد.

- سیمبول VAR_IN_ OUT

این سیمبول برای ورودی/خروجی FB به صورت محلی مورد استفاده قرار می‌گیرد. در واقع همانند مثال زیر کاربرد این کلاس از سیمبول برای ایجاد بازخورد[2] است. زمانی که FB فراخوانی می‌شود، مقدار D1 در DT_IO خوانده (سیمبول کلاس نوع VAR_IN_ OUT) و پس از انجام عملیات بلوک، مقدار DT_IO به D1 فرستاده می‌شود.

[1] در مورد این نوع داده در فصل مربوط به زبان برنامه نویسی SFC بیشتر توضیح داده خواهد شد. , STEP

[2] Feedback



شکل ۶-۲ پردازش اطلاعات FB و ارتباط آن با انواع ورودی، خروجی و ورودی/خروجی

۶-۱-۲- انواع داده

سیمبول‌ها می‌توانند به داده‌ها با فرمت‌های گوناگون تخصیص داده شوند، مثلا یک کانتکت نیاز به سیمبولی برای داده‌ای بیتی دارد و مقدار خوانده شده از ورودی آنالوگ نیاز به سیمبولی برای داده‌ای به فرمت Word (۱۶ بیتی) دارد.

انواع مختلف داده که ISPSoft از آن‌ها پشتیبانی می‌کند در جدول زیر آمده است. توجه شود که مدل‌های مختلف PLC و همچنین برنامه اصلی و FBها هرکدام فرمت‌های مشخصی از داده را پشتیبانی می‌کنند.

جدول ۶-۱: انواع داده در ISPSoft

نوع داده	شرح	DVP		AH500	
		FB	Program	FB	Program
BOOL	داده بیتی	√	√	√	√
WORD	داده ۱۶ بیتی	√	√	√	√
DWORD	داده ۳۲ بیتی	√	√	√	√
LWORD	داده ٦٤ بیتی	√	√	√	√

INT	عدد ۱۶ بیتی علامت دار (بیت انتهایی علامت عدد را مشخص می‌کند)			√	√
DINT	عدد ۳۲ بیتی علامت دار (بیت انتهایی علامت عدد را مشخص می‌کند)			√	√
LINT	عدد ٦٤ بیتی علامت دار (بیت انتهایی علامت عدد را مشخص می‌کند)			√	√
REAL	عدد دهدهی ممیز شناور ۳۲ بیتی با این فرمت می‌توان اعداد را به صورت اعشاری نیز نشان داد.	√	√	√	√
LREAL	عدد دهدهی ممیز شناور ٦٤ بیتی با این فرمت می‌توان اعداد را به صورت اعشاری نیز نشان داد.			√	√
ARRAY	آرایه (ماتریس ستونی) زمانی که سیمبول تعیین می‌شود تعداد و نوع داده‌های آرایه مشخص خواهند شد (حداکثر تعداد المان‌های آرایه می تواند ۲۰٤۸ باشد، به این المان‌ها اصطلاحا "درایه" گفته می‌شود.)	√	√	√	√
STRING	رشته (رشته ای از کاراکترهای اسکی[1]) کدهای اسکی با مقادیر ۸ بیتی تعریف می-			√	√

[1] ASCII

	شوند. طول آرایه توسط سیمبول تعیین و حداکثر می‌تواند شامل ۱۲۸ کاراکتر باشد.				
STEP	برای تشخیص پرچم پله (در فصل ۱۰- در مورد آن بیشتر گفته است)	√	√		√
Function Block	برای نام گزاری FB		√	√	√
COUNTER	داده های شمارنده	√	√		√
TIMER	داده های زمان سنج	√	√		√
POINTER	اشاره‌گر به فرمت WORD (در فصل ۷- در مورد آن بیشتر گفته است)			√	
T_ POINTER	اشاره‌گر داده‌های به فرمت زمان سنج			√	
C_ POINTER	اشاره‌گر داده‌های به فرمت شمارنده			√	
HC_ POINTER	اشاره‌گر داده‌های به فرمت شمارنده های پر سرعت[۱]			√	

۶-۱-۳- تخصیص حافظه به سیمبول و مقداردهی اولیه

نوع حافظه‌ای که سیمبول به آن تخصیص داده شده به فرمت داده مورد نظر بستگی دارد. کاربران می‌توانند مقادیر اولیه سیمبول را تعیین و در PLC بارگزاری کنند. مفاهیم مرتبط با تخصیص سیمبول در زیر آمده است. این مفاهیم به مدل PLC مورد استفاده وابسته است.

جدول ۶-۲: قواعد تخصیص سیمبول در PLC های سری های گوناگون

[۱] High-speed counter

مدل	نحوه اختصاص
AH500	• کاربر (و یا سیستم به صورت اتوماتیک) می‌تواند حافظه‌ها را به سیمبول های محلی و سراسری POU برنامه تخصیص دهد[1]. • سیستم به صورت اتوماتیک حافظه‌ها را به سیمبول‌های محلی تعریف شده در FB اختصاص می‌دهد (کاربر نمی‌تواند این کار را انجام دهد). در واقع در اینجا نیاز به یک تعداد حافظه موقت و یا Temp داریم که سیستم تنظیمات مربوط به آن را در پس زمینه انجام می‌دهد. • حافظه‌های اختصاص داده شده توسط سیستم حافظه‌های رزرو شده CPU هستند و از حافظه‌های داده استفاده نمی‌شود. (البته در مورد STEP و شمارنده و زمان سنج اینطور نیست و سیستم به طور اتوماتیک رله و حافظه‌های مرتبط به آن‌ها (که استفاده نشده‌اند) را اختصاص می‌دهد).
DVP	• کاربر (و یا سیستم) می‌توانند حافظه و پورت ها را به هر کدام از سیمبول های محلی و یا سراسری اختصاص دهند. • حافظه‌های تخصیص داده شده توسط سیستم همان حافظه‌هایی قابل استفاده کاربر است (کاربر می‌تواند محدوده حافظه‌های قابل استفاده سیستم را در Device Resource Allocation تعیین کند)

ارتباط بین انواع داده‌ها و حافظه‌های قابل تخصیص به آن‌ها در جدول زیر آمده است.

جدول ۶-۳: حافظه های قابل تخصیص به انواع مختلف داده

نوع داده	DVP		AH500	
	FB	Program	FB	Program

[1] سیمبول‌های محلی را تنها در برنامه‌ای که در آن تعریف شده‌اند می‌توانند استفاده کرد ولی از سیمبول‌های سراسری در کل پروژه می‌توان استفاده کرد.

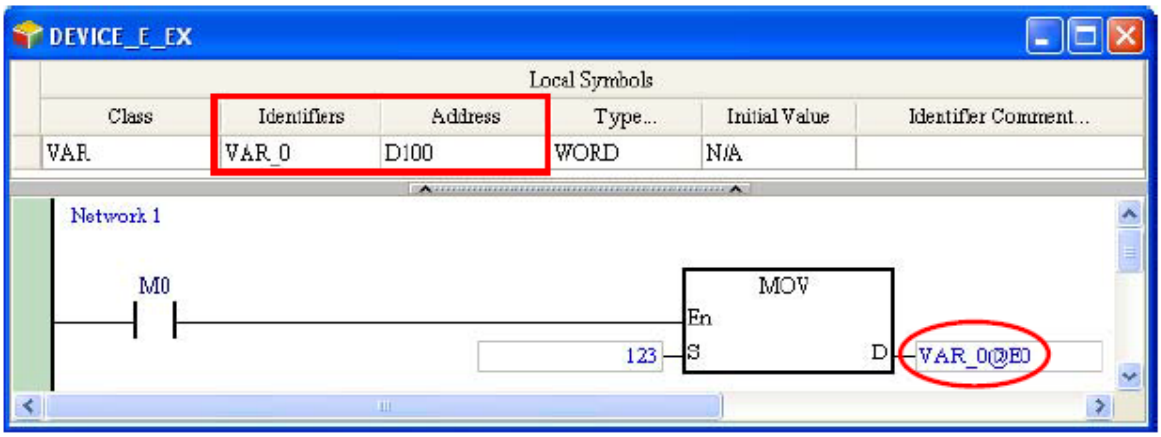
BOOL	کانتکت M	کانتکت M/X/Y	حافظه داخلی[1]	کانتکت M/SM بیت حافظه D/L/X/Y
WORD	D	D/E/F	حافظه داخلی	D/L/X/Y/E/SR
DWORD	D	D/E/F	حافظه داخلی	D/L/X/Y/E/SR
LWORD	D	D/E/F	حافظه داخلی	D/L/X/Y/E/SR
INT			حافظه داخلی	D/L/X/Y
DINT			حافظه داخلی	D/L/X/Y
LINT			حافظه داخلی	D/L/X/Y
REAL	D	D	حافظه داخلی	D/L/X/Y
LREAL			حافظه داخلی	D/L/X/Y
STRING			حافظه داخلی	D/L/X/Y
STEP	S	S	S	S
COUNTER	C	C	C	C/HC
TIMER	T	T	T	T
ARRAY	• حافظه تخصیص داده شده به سیمبول به نوع داده های آرایه وابسته است. داده‌های آرایه به ترتیب از حافظه با آدرس نقطه شروع (که توسط کاربر و یا سیستم تعیین می‌شود) نوشته می‌شوند و طول فضایی که اشغال می‌کنند برابر با طول آرایه است. • نوع داده آرایه نمی‌تواند SR/SM/E/F باشد.			

[1] کاربر به این حافظه ها دسترسی ندارد

۶-۱-۴- آدرس دهی غیر مستقیم

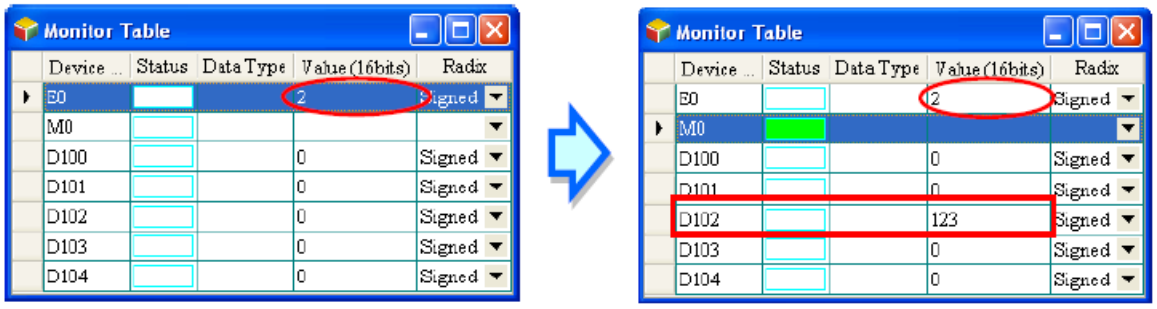
در ISPSoft می‌توان سیمبول را با استفاده از رجیستر میانی[1] (مشابه اشاره گر عمل میکند و با استفاده از آن می‌توان به صورت غیر مستقیم آدرس دهی کرد) تخصیص داد. فرمت مورد استفاده در این وضعیت به صورت Identifier@Index register است. در واقع مقدار ذخیره شده در رجیستر Index Register نشانگر تعداد انتقال از آدرس نقطه شروع (حافظه‌ای که توسط سیمبول Identifier مشخص می‌شود) است. نوع Index Rgister می‌تواند حافظه E، F و یا سیمبولی از حافظه‌های E و F باشد. این نوع آدرس دهی در پیاده سازی الگوریتم‌های مبتی بر مراکز داده و یا Look Up Table ها بسیار کارگشا است.

برای روشن شدن موضوع به مثال زیر توجه کنید. در این مثال VAR_0 سیمبول حافظه D100 است. مقدار E0 هم برابر ۲ است، پس D(100+2) آدرس سیمبول VAR_0@E0 را نشان می‌دهد.



شکل ۶-۳ استفاده از آدرس دهی غیر مستقیم سیمبول

در واقع در این مثال، زمانی که M0 فعال است مقدار ۱۲۳ به رجیستر D102 منتقل می‌شود.

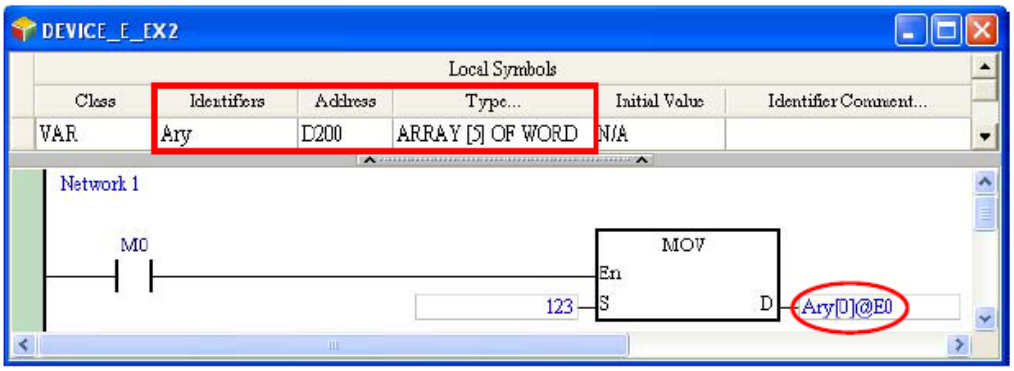


شکل ۶-۴ عملکرد سیستم حین استفاده از آدرس دهی غیرمستقیم

[1] Modifying a Symbol with an Index Register

به همین صورت می‌توان از سیمبول آرایه برای فراخوانی استفاده کرد، در این حالت فرمت آن به صورت Identifier[Index]@Index register خواهد بود. در این حالت Index برای Identifier باید مقداری ثابت باشد (اگر سیمبول است نمی تواند فرمت آرایه داشته باشد).

برای مثال در پروژه زیر Ary آرایه‌ای با ۵ درایه است و آدرس خانه شروع آن D200 است. در نتیجه زمانی که سیستم کامپایل شود آدرس D200 الی D204 به آن اختصاص داده می‌شود. اگر مقدار ذخیره شده در رجیستر Index یعنی E0 برابر ۲ باشد، Ary[0]@E0 بیانگر آن است که آدرس Ary[0] یعنی D200 با دو جمع می‌شود یعنی D(200+2)، پس Ary[0]@E0 اشاره به آدرس خانه D202 دارد (همان Ary[0+2]). توجه شود که اگر مقدار E0 برابر ۶ باشد، با آنکه آدرس از طول درایه بیشتر می شود (چرا که Ary[6] نداریم) ولی این نوع آدرس دهی مجاز به حساب می آید و Ary[0]@E0 به خانه D(200+6) اشاره می‌کند. بنابراین با توجه به آنکه نرم افزار محدوده‌های مجاز نشانگرها را مدیریت نمی‌کنند، لازم است کاربر در استفاده از آن‌ها دقت لازم را به عمل آورد. مثلا همیشه لازم است توجه شود که محدوده آدرس دهی شده در محدوده مجاز CPU باشد.



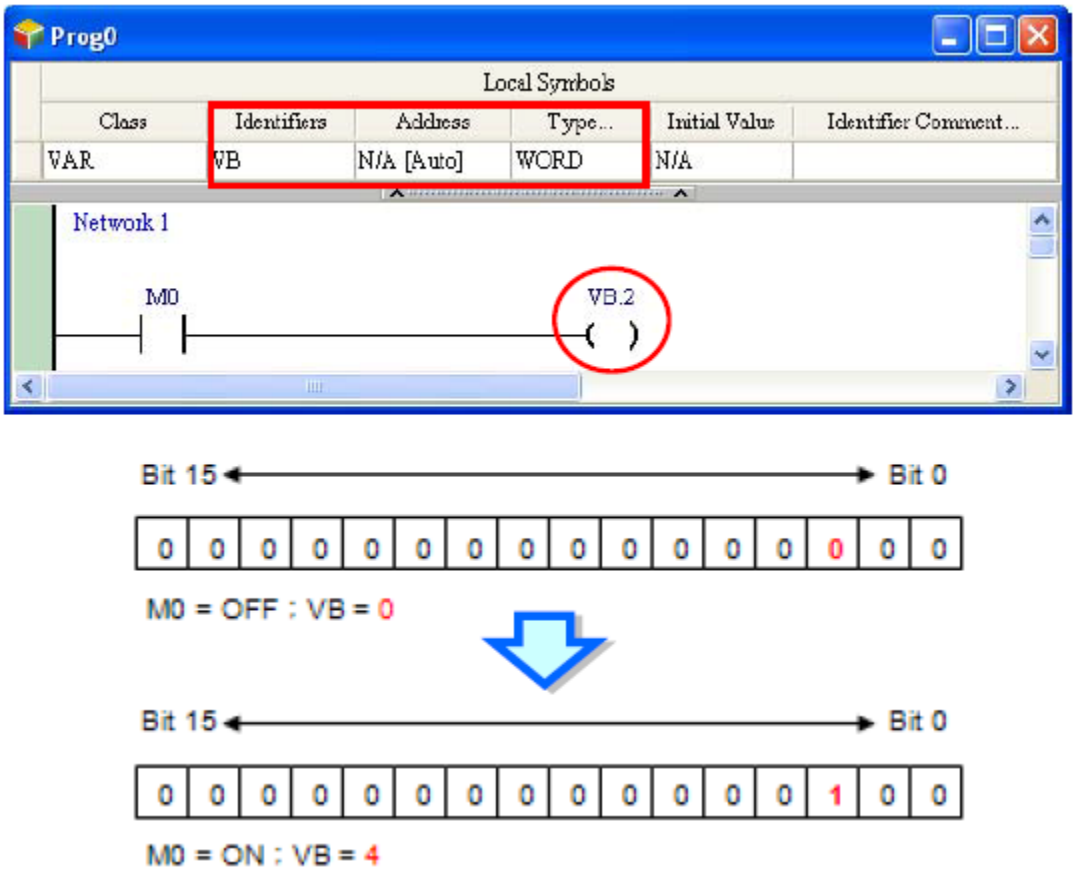
شکل ۶-۵ آدرس دهی غیر مستقیم آرایه ها

۶-۱-۵- سیمبول بیت های حافظه های داده

در پردازنده‌های سری AH500 می‌توان به "بیت‌ها" را در حافظه‌های Word (داده ۱۶ بیتی) دسترسی داشت و آن‌ها را تغییر داد. برای اینکار می‌توانیم از فرمت Word device.Bit number استفاده کنیم، برای نمونه D0.2 نمایانگر بیت دوم در D0 است. دقیقا فرایند مشابهی برای دسترسی به بیت‌ها در سیمبول حافظه‌های Word وجود دارد و می‌توان آن‌ها را به صورت Identifier.Bit number نشانه‌گزاری کرد مثلا VB.2 نشانگر بیت دوم سیمبول VB است. در صورتی که با سیمبول یک آرایه سروکار داشته باشیم، فرمت دسترسی بیتی به آن به صورت

Identifier[Index].Bit number است برای نمونه Ary[0].1 که نمایانگر بیت اول درایه صفرم آرایه است. شماره بیت باید یک عدد دهدهی مثبت بین صفر تا ۱۵ باشد.

در مثال زیر نوع داده VB به صورت Word است، فرض کنید مقدار فعلی آن صفر است. زمانی که M0 فعال می‌شود بیت دوم VB یک شده و مقدار حافظه VB برابر ۴ می‌شود.



شکل ۶-۶ نحوه دسترسی به بیت های حافظه داده

توجه شود که در AH500 می‌توان از رجیستر میانی (Index) نیز برای دسترسی به بیت‌ها استفاده کرد. در این حالت اولویت عملگر "@" از اولویت عملگر "." بیشتر است. برای درک بیشتر این موضوع به مثال‌های زیر توجه کنید.

مثال ۱: عبارت VB.1@E0 را در نظر بگیرید (VB معرف D100 و مقدار E0 برابر ۳ است). در این حالت چون اولویت با عملگر "@" است، می‌توانیم به عبارت فوق به صورت VB.(1@E0) نگاه کنیم. عبارت 1@E0 به معنی انتقال مقدار ۱ به تعداد ذخیره شده در E0 است، پس ۱+۳=۴ مقدار این عبارت است: 1@E0=4. پس عبارت به صورت VB.4 در می‌آید که بیانگر D100.4 یعنی بیت چهارم D100 است.

مثال ۲: عبارت VB@E0.1 را در نظر بگیرید (VB معرف D100 و مقدار E0 برابر ۳ است). در این حالت چون اولویت با عملگر "@" است، می‌توانیم به عبارت فوق به صورت (VB@E0).1 نگاه کنیم. عبارت VB@E0 به معنی D(100+3)=D103 است و در نتیجه کل عبارت VB@E0.1 به صورت D103.1 در می‌آید که به معنی بیت اول D103 می‌باشد.

مثال ۳: عبارت VB@E0.1@E1 را در نظر بگیرید (VB معرف D100 و مقدار E0 برابر ۳ و مقدار E1 برابر ۲ است). در این حالت چون اولویت با عملگر "@" است، می‌توانیم به عبارت فوق به صورت (VB@E0).(1@E1) نگاه کنیم. عبارت VB@E0 به معنی D(100+3)=D103 است و 1@E2 به معنی ۱+۲=۳ است. پس کل عبارت به معنی D103.3 است که بیانگر بیت سوم D103 می‌باشد.

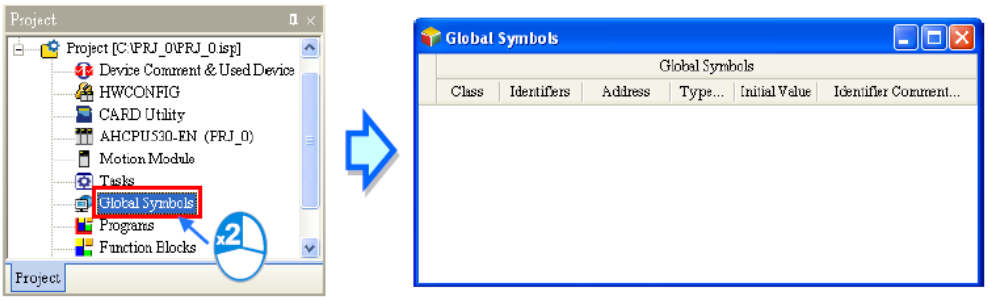
توجه شود که در نسخه فعلی (نسخه دوم) نرم افزار ISPSoft بازه بیتی بین صفر تا ۱۵ است، در حالی که در نسخه‌های آتی این نرم افزار محدوده بیتی قابل دسترس ممکن است از صفر تا ۳۱ و یا ۶۳ افزایش پیدا کند.

۶-۲- مدیریت سیمبول ها در ISPSoft

۶-۲-۱- جدول سیمبول ها

- جدول سیمبول‌های سراسری

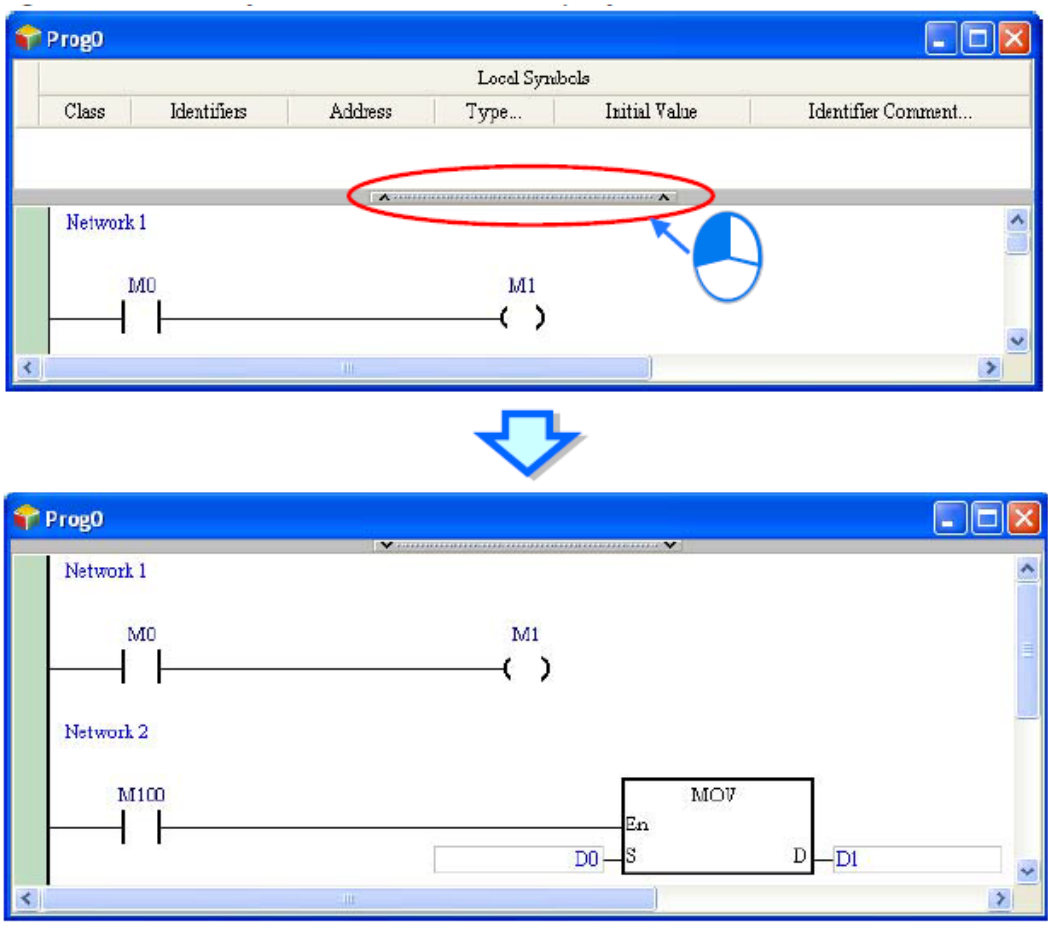
در بخش مدیریت پروژه بر روی Global Symbols دوبار کلیک کرده تا پنجره مرتبط با آن باز شود.



شکل ۶-۷ پنجره جدول سیمبول سراسری

- جدول سیمبول‌های محلی

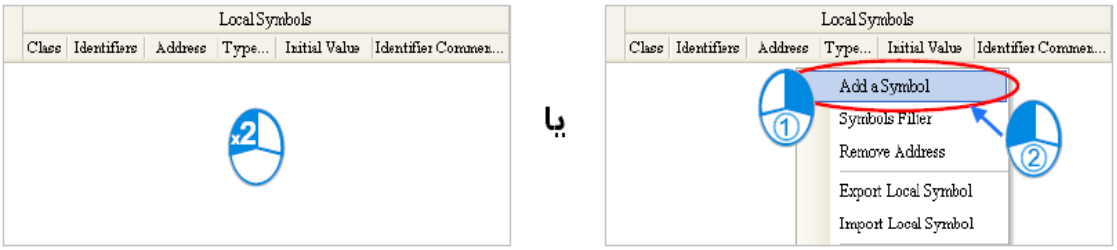
جدول سیمبول‌های محلی در بالای صفحه هر POU قرار دارد. اگر کاربر بر روی کلید نواری زیر بخش سیمبول‌های محلی در پنجره ویرایش برنامه POU کلیک کند، قسمت سیمبول پنهان خواهد شد، پس از دوباره کلیک کردن بر روی این کلید نواری، بخش جدول سیمبول‌های محلی دوباره پدیدار خواهد شد.



شکل ۶-۸ جدول سیمبول های محلی

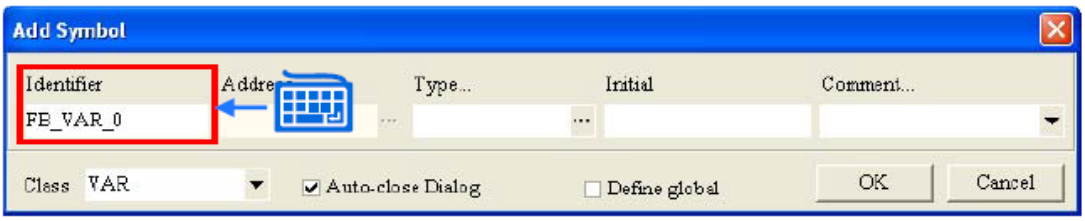
۶-۲-۲- اضافه کردن سیمبول

برای اضافه کردن سیمبول جدید به جدول سیمبول‌ها می‌توان بر روی صفحه دوبار کلیک کرد و یا پس از کلیک راست گزینه Add Symbol را انتخاب کرد.



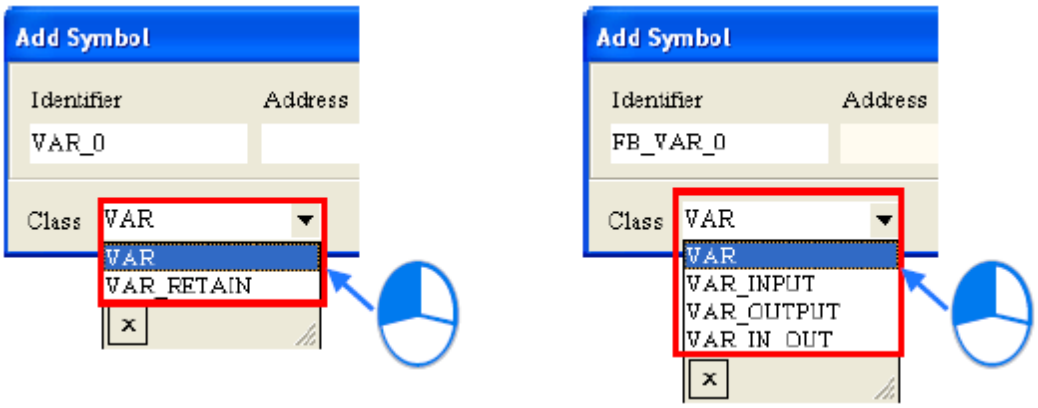
شکل ۶-۹ اضافه کردن سیمبول جدید در جدول سیمبول ها

در پنجره Add Symbol، شناسه سیمبول را در قسمت Identifier وارد کنید.



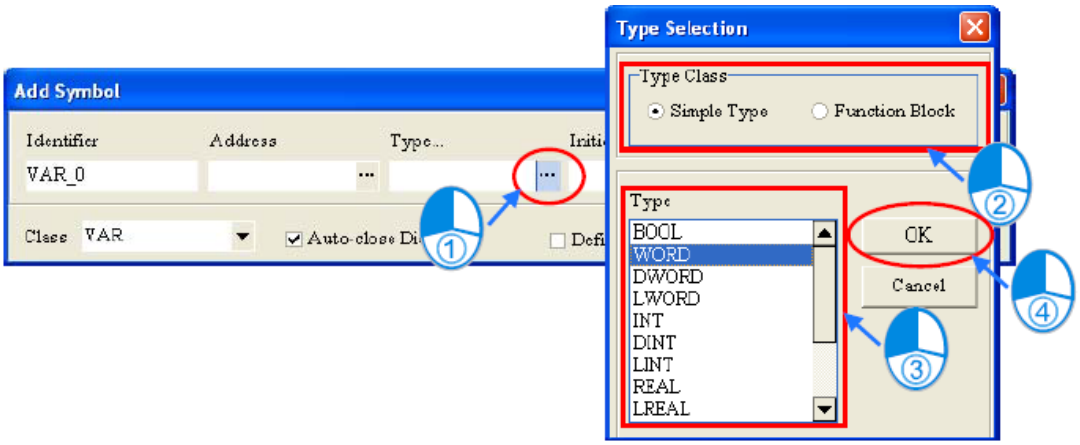
شکل ۶-۱۰ تنظیم سیمبول جدید در جدول سیمبول ها

نوع کلاس را نیز در قسمت Class تعیین کنید. این بخش برای سیمبول‌های سراسری و محلی گزینه‌های متفاوتی دارد.



شکل ۶-۱۱ تنظیم کلاس سیمبول جدید در جدول سیمبول ها

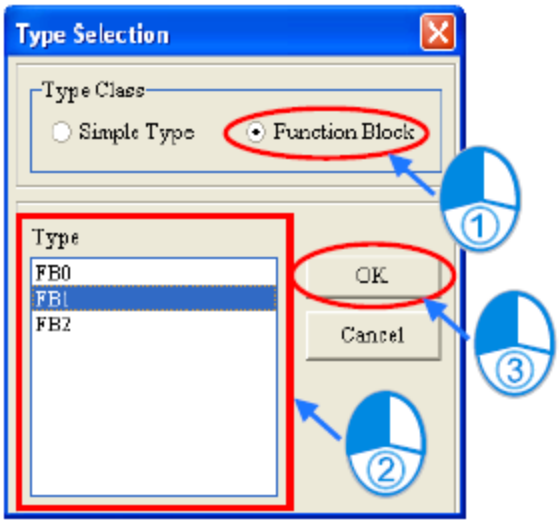
در بخش Type نیز می‌توانیم فرمت داده خود را تعیین کنیم.



شکل ۶-۱۲ فرمت داده سیمبول جدید در جدول سیمبول ها

- Function Block

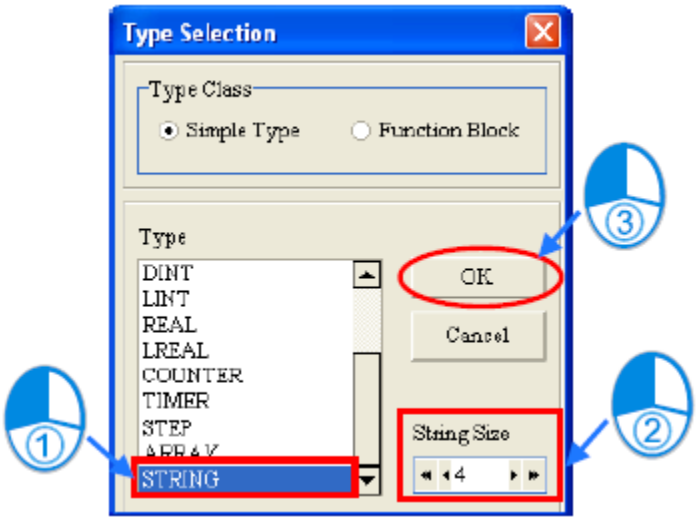
در قسمت Type Class گزینه Function Block را انتخاب کرده و سپس FB مورد نظر را در بخش Type انتخاب می‌کنیم.



شکل ۶-۱۳ فرمت داده Function Block برای سیمبول جدید در جدول سیمبول ها

- STRING

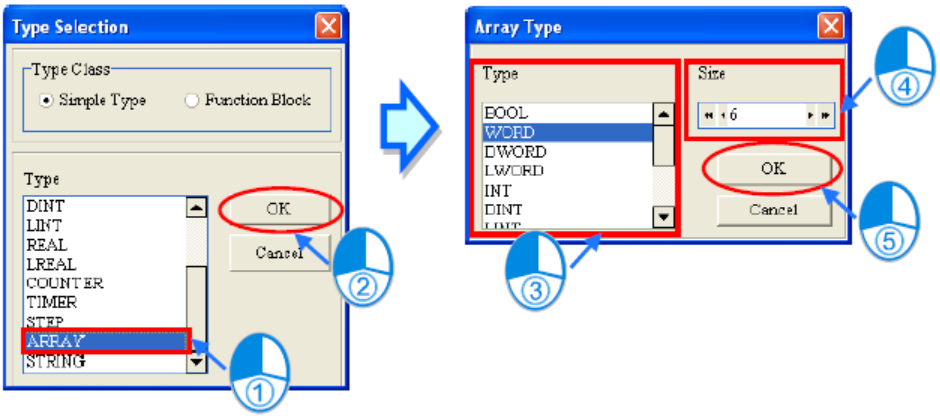
اگر نوع داده سیمبول STRING باشد، کاربر باید اندازه آن را در بخش String Size تعیین کند، این مقدار می‌تواند بین ۱ الی ۱۲۸ کاراکتر باشد.



شکل ۶-۱۴ فرمت داده STRING برای سیمبول جدید در جدول سیمبول ها

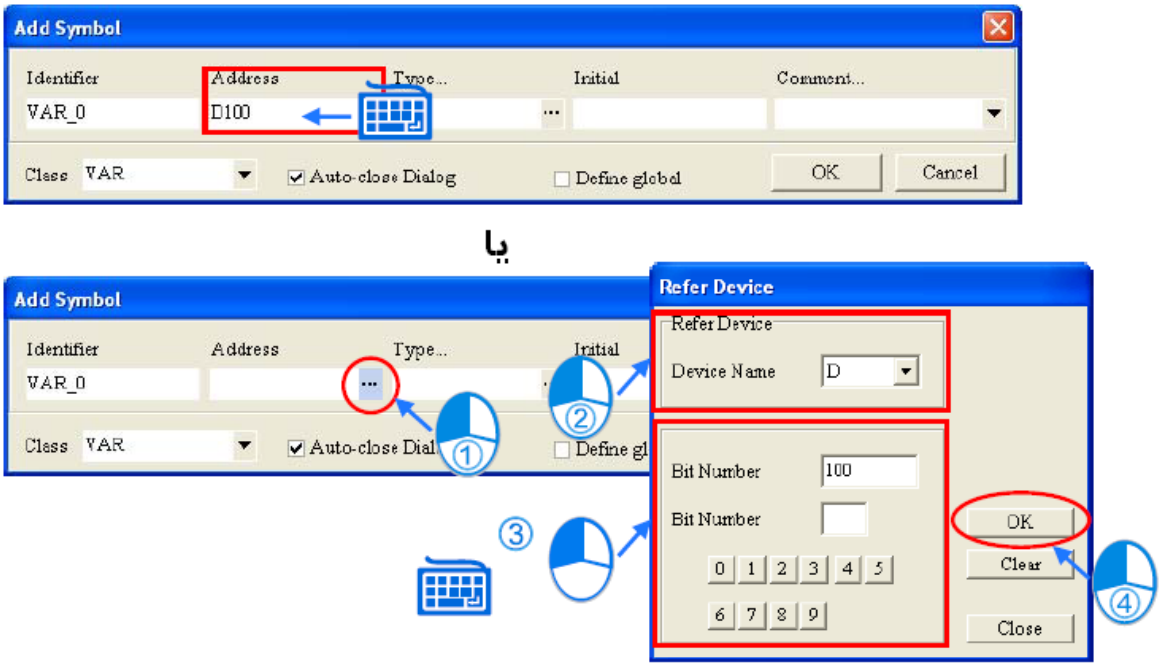
- ARRAY

اگر نوع داده سیمبول به صورت آرایه باشد، پس از کلیک بر روی OK پنجره Array Type باز خواهد شد. در این پنجره باید نوع داده‌های Array را در بخش Type و تعداد آرایه‌های آن را در بخش Size تعیین کنیم (این مقدار می‌تواند بین ۱ تا ۲۰۴۸ باشد).



شکل ۶-۱۵ فرمت داده ARRAY برای سیمبول جدید در جدول سیمبول ها

در صورتی که بخش Address خالی بماند، سیستم به صورت اتوماتیک به سیمبول حافظه اختصاص می‌دهد و در غیر این صورت کاربر خود می‌تواند آدرس حافظه و یا پورت مورد نظر خود را در آن وارد کند. کاربر می‌تواند تعیین حافظه را با کلیک بر روی بخش چپ منوی Address انجام دهد.

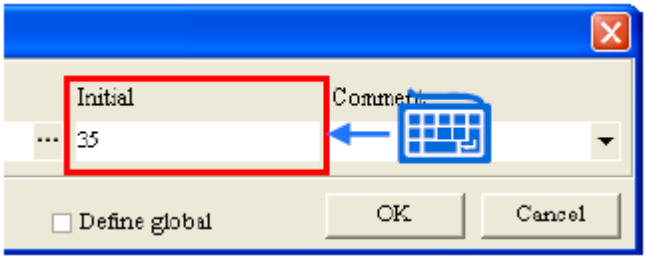


شکل ۶-۱۶ تنظیم آدرس سیمبول جدید در جدول سیمبول ها

توجه شود که برای سیمبول Function Block نمی‌توان آدرس تعیین کرد و در AH500 به داده‌های محلی نمی‌توان آدرس اختصاص داد.

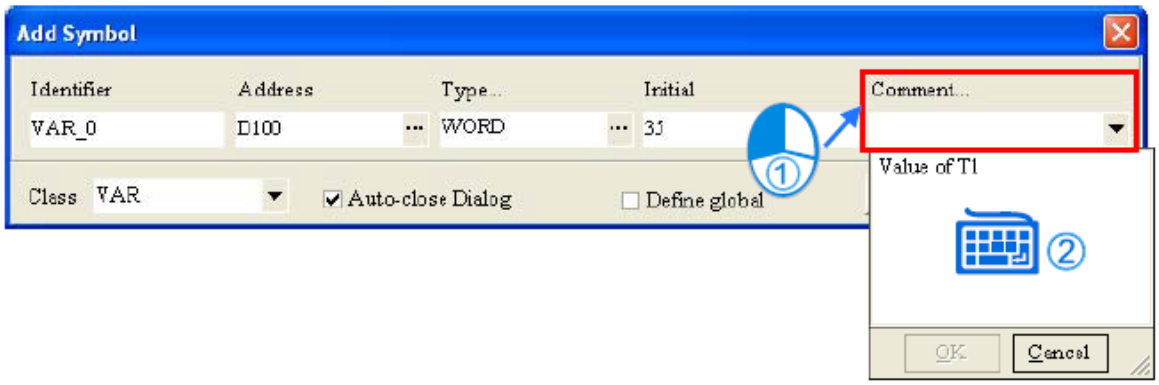
در صورتی که نیاز به تعیین مقدار اولیه برای سیمبول‌ها باشد، می‌توان از بخش Initial استفاده کرد. در صورتی که نوع داده آرایه باشد، پس از کلیک بر روی Initial صفحه‌ای برای مقدار دهی درایه‌های[1] آرایه باز خواهد شد.

در صورتی که نوع داده Bool باشد در آن T به معنای True یا فعال و F به معنای False و یا غیرفعال است. برای داده های با فرمت STRING می‌بایست مقدار اولیه به تعداد اندازه آن، کاراکتر بدون علامت در درون علامت نقل قول" " باشد. در صورتی که نوع داده Function Block باشد قسمت مقدار دهی اولیه غیرفعال خواهد شد.



شکل ۶-۱۷ مقدار دهی اولیه سیمبول جدید در جدول سیمبول ها

برای اضافه کردن توضیحات در مورد سیمبول می‌توان از بخش Comment استفاده کرد.



شکل ۶-۱۸ توضیحات برای سیمبول جدید در جدول سیمبول ها

در صورتی که گزینه Auto-close Dialoge فعال باشد صفحه ورود سیمبول جدید پس کلیک بر روی OK بسته خواهد شد، در غیر این صورت این صفحه برای ورود سیمبول جدید باقی خواهد ماند. در

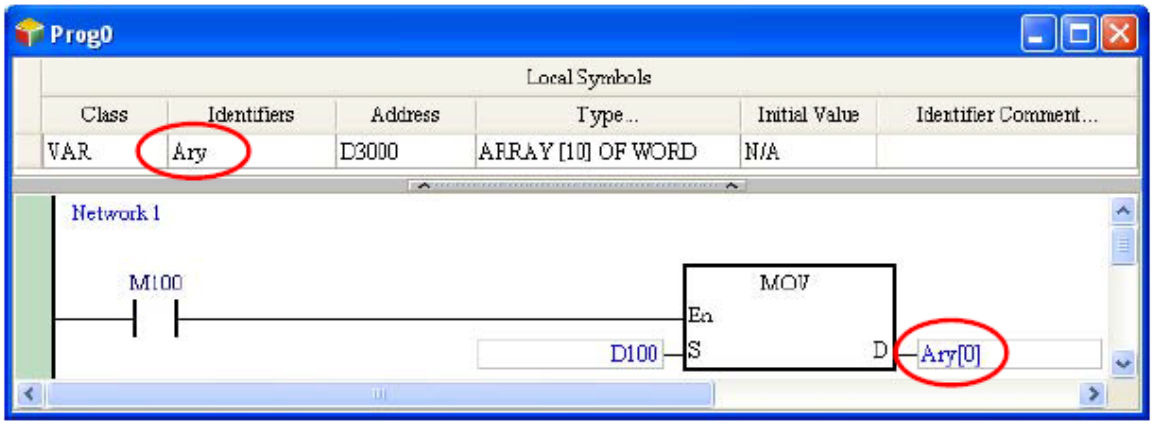
[1] اجزای آرایه

صورتی که گزینه Define Global انتخاب شود (این گزینه تنها در صفحه ورود سیمبول‌های محلی وجود دارد) سیمبول تعریف شده به عنوان سیمبول سراسری تعریف خواهد شد.

۶-۲-۳- استفاده از سیمبول داده های آرایه و STRING

۶-۲-۳-۱- سیمبول آرایه

برای اینکه کاربر بتواند از سیمبول آرایه‌ای در برنامه خود استفاده کند، فرمت آن باید به صورت "شناسه[]" و یا به عبارتی "Identifier[Index]" باشد. در این فراخوانی index باید مقدار ده دهی مثبت و بین صفر تا یکی کمتر از حداکثر طول آرایه باشد. برای مثال اگر آرایه‌ای دارای ۱۰ درایه باشد، مقدار index می‌تواند بین صفر تا 9باشد در غیر این صورت سیمبول پذیرفته نخواهد شد (با پیغام خطا مواجه خواهیم شد).

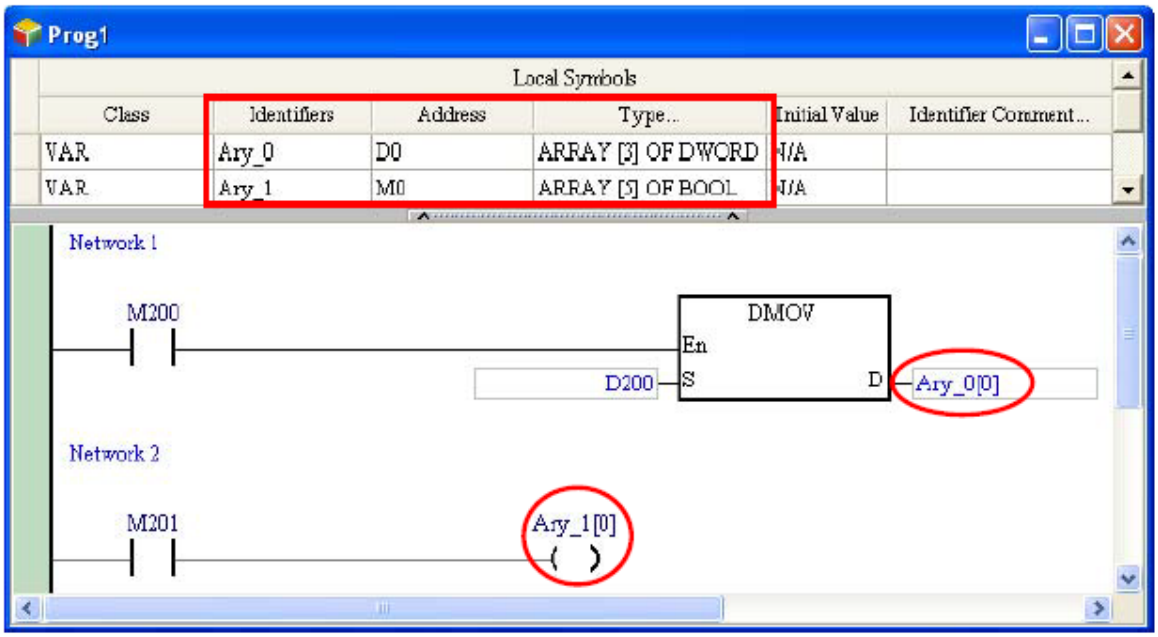


شکل ۶-۱۹ استفاده از سیمبول آرایه

در صورتی که از سیمبول به صورت آرایه استفاده می‌کنیم، باید فرمت آن، آدرس شروع و اندازه آن را تعیین کنیم. اندازه آرایه می‌تواند بین یک الی ۲۰۴۸ باشد، درایه‌های آرایه به ترتیب از آدرس شروع آن (تا طول در نظر گرفته شده برای آرایه) تخصیص داده می‌شوند. به همین جهت باید توجه شود که آرایه از حد مجاز حافظه پردازنده عبور نکند. توجه شود که آرایه نمی‌تواند از نوع SR، SM ، E و یا F باشد.

در شکل زیر عبارت ARRAY[3] OF DWORD در بخش Type برای سیمبول Ary_0 نشانگر آرایه‌ای با سه درایه با فرمت DWORD است. با توجه به آدرس شروع آرایه در بخش Address که D0 تعیین شده است، مشخص می‌شود آرایه فضای D0 الی D5 را اشغال کرده است (چرا که هر حافظه نوع D شانزده بیت دارد، از آنجایی که فرمت DWORD سی و دو بیت برای ذخیره سازی لازم دارد، پس لازم است به هر آرایه با فرمت DWORD دو حافظه نوع D اختصاص داده شود و چون در اینجا سه آرایه داریم پس ۳*۲=۶ حافظه D از D0 تا D5 باید به این سیمبول اختصاص داده شود- مطابق جدول زیر).

همچنین عبارت ARRAY[5] OF BOOL در بخش Type برای سیمبول Ary_1 نشانگر آرایه‌ای با ۵ درایه بیتی است. با توجه به آدرس شروع آرایه که M0 است مشخص است که این سیمبول به حافظه‌های M0 تا M5 تخصیص داده می‌شود.



شکل ۶-۲۰ اختصاص درایه سیمبول های آرایه ای

جدول ۶-۴: حافظه های اختصاص داده شده

آرایه	حافظه های اختصاص داده شده
Ary_0[0]	D0 و D1
Ary_0[1]	D2 و D3
Ary_0[2]	D4 و D5
Ary_1[0]	M0
Ary_1[1]	M1
Ary_1[2]	M2
Ary_1[3]	M3
Ary_1[4]	M4

همچنین کاربران می‌توانند به آرایه مقدار اولیه دهند. برای این کار به مثال زیر توجه کنید. مقدار [1,2,3,4,5] تایپ شده در قسمت Initial Value برای سیمبول A_Ary نشان می‌دهد مقدار اولیه تعریف شده برای A_Ary[0] برابر ۱، برای A_Ary[1] برابر۲، A_Ary[2] برابر ۳، برای A_Ary[3] برابر ۴ و برای A_Ary[4] برابر ۵ است (این نحوه مقدار دهی تا حدودی مشابه مقدار دهی ماتریسی در نرم افزار متلب[1] و سیمولینک[2] است). همچنین مقدار [1,3(0),5] تایپ شده در قسمت Initial Value برای سیمبول B_Ary نشان می‌دهد مقدار اولیه تعریف شده برای B_Ary[0] برابر ۱، برای B_Ary[1] برابر صفر، B_Ary[2] برابر صفر ، برای B_Ary[3]برابر صفر و برای B_Ary[4] برابر ۵ است (در واقع 3(0) نمایانگر سه درایه صفر است).

Local Symbols

Class	Identifiers	Address	Type...	Initial Value	Identifier Comment...
VAR	A_Ary	D7000	ARRAY [5] OF WORD	[1,2,3,4,5]	
VAR	B_Ary	D7005	ARRAY [5] OF WORD	[1,3(0),5]	

شکل ۶-۲۱ مقدار دهی اولیه آرایه ها

جدول ۶-۵: نحوه تخصیص مقادیر اولیه به درایه آرایه ها

B_Ary		A_Ary	
آرایه	مقدار اولیه	آرایه	مقدار اولیه
B_Ary[0]	۱	A_Ary[0]	۱
B_Ary[1]	صفر	A_Ary[1]	۲
B_Ary[2]	صفر	A_Ary[2]	۳
B_Ary[3]	صفر	A_Ary[3]	۴
B_Ary[4]	۵	A_Ary[4]	۵

[1] MATLAB

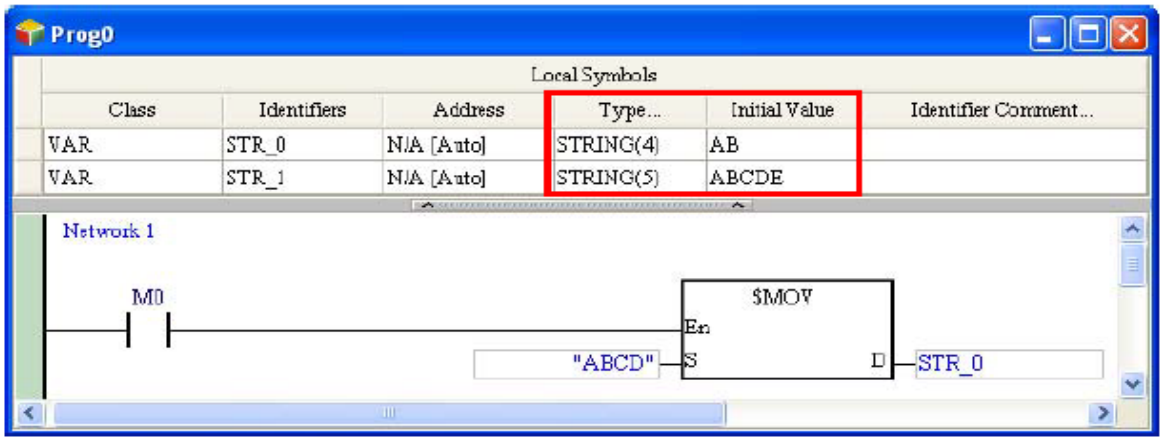
[2] Simulink

۶-۲-۳-۲- سیمبول STRING

در ISPSoft درایه‌های (کاراکترهای) STRING کدهای هشت بیتی اسکی[1] هستند (هر کد اسکی نماینده یک حرف، عدد و یا نماد است). داده‌های STRING را برای نمایش بین علامت نقل قول " " قرار می‌دهند مانند "ABCD123A". توجه شود که STRING برای کلاس‌های VAR_INPUT، VAR_OUTPUT و VAR_IN_OUT تعریف نمی‌شود.

هر STRING شامل یک الی ۱۲۸ کاراکتر می‌تواند باشد (هر کاراکتر هم یک بایت را اشغال می‌کند). STRING نیاز به یک بایت پایانی برای مشخص کردن انتهای خود دارد، برای مثال با دو WORD می توان سه کاراکتر را ذخیره کرد (۳ کاراکتر و یک بایت پایانی) و با سه WORD می‌توان چهار و یا ۵ کاراکتر را ذخیره کرد.

در مثال زیر اعداد داخل پرانتز در بخش Type تعداد کاراکتر هر STRING را مشخص می‌کند. همچنین کاراکترهای بخش Initial Value برای تعیین مقادیر اولیه، نیاز به علامت نقل قول " " ندارند. تعداد کاراکترهای مقدار اولیه باید از تعداد کاراکترهای STRING کمتر باشند.

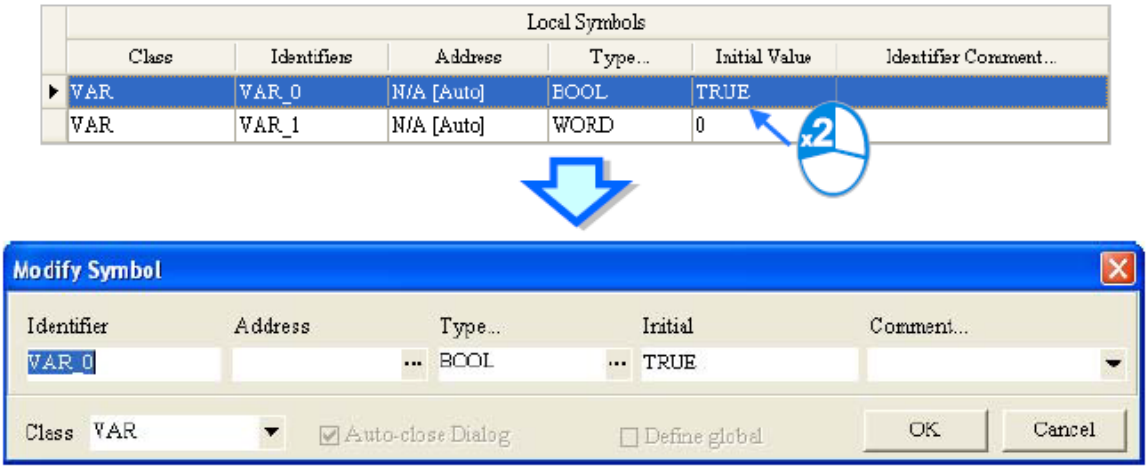


شکل ۶-۲۲ سیمبول STRING

۶-۲-۴- ویرایش سیمبول ها

برای تغییر ویژگی سیمبول‌ها، می‌توان بر روی آن‌ها دو بار کلیک و در صفحه ظاهر شده ویژگی‌های مورد نظر آن‌ها را تغییر داد.

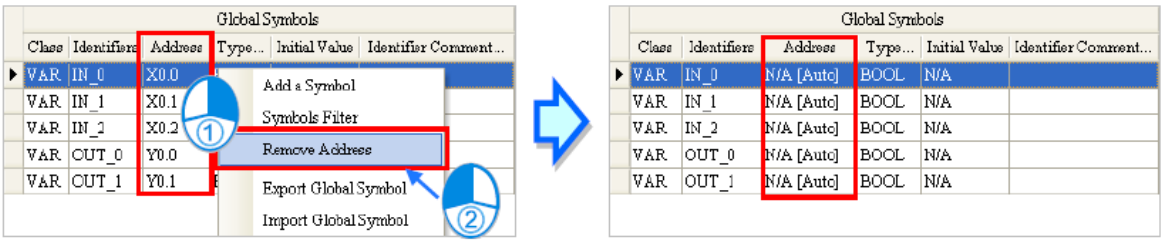
[1] ASCII



شکل ۶-۲۳ ویرایش سیمبول ها

کاربر با کلیک راست بر روی سیمبول‌ها در جدول سیمبول‌ها می‌تواند عملیات‌هایی همچون کپی برداری، الحاق، Undo و انتخاب را برای نسخه برداری و انتقال سیمبول‌ها بین پروژه‌ها انجام دهد. بین مدل‌های دو خانواده AH500 و DVP نمی‌توان سیمبول‌ها را منتقل کرد. همچنین حداکثر تعداد Undo ممکن ۲۰ بار است و عملیات‌های پیشتر از ۲۰ بار قابل بازیابی نیستند. در صورتی که در زمان الحاق سیمبول در جدول سیمبول‌ها نام مشابه آن وجود داشته باشد، سیستم در انتهای نام سیمبولِ الحاقی عبارت "CopyOf_n" را قرار می‌دهد، که n یک عدد غیرتکراری است.

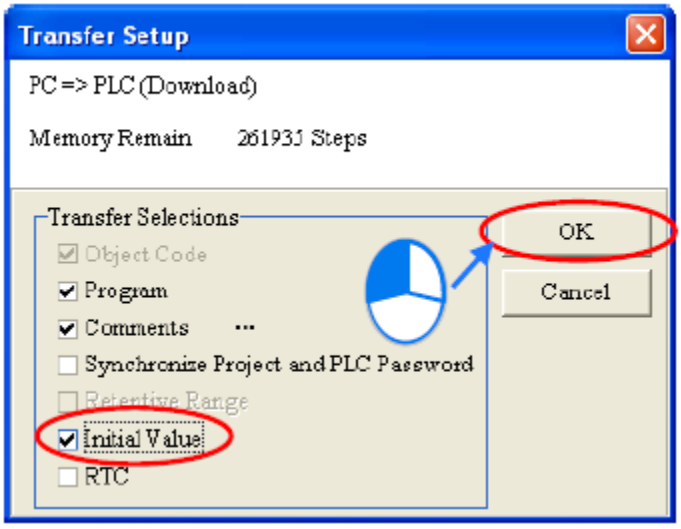
برای پاک کردن آدرس سیمبول‌ها (چه سیستم آن‌ها را اختصاص داده باشد و یا کاربر) می‌توان آن سیمبول‌ها را در جدول انتخاب و سپس پس از کلیک راست، Remove Address را انتخاب کرد. حتی اگر آدرس حافظه سیمبول‌ها پاک شود، سیستم بعد از کامپایل، دوباره به آن‌ها آدرس اختصاص خواهد داد.



شکل ۶-۲۴ پاک کردن آدرس سیمبول ها

۶-۲-۵- دانلود مقادیر اولیه سیمبول ها

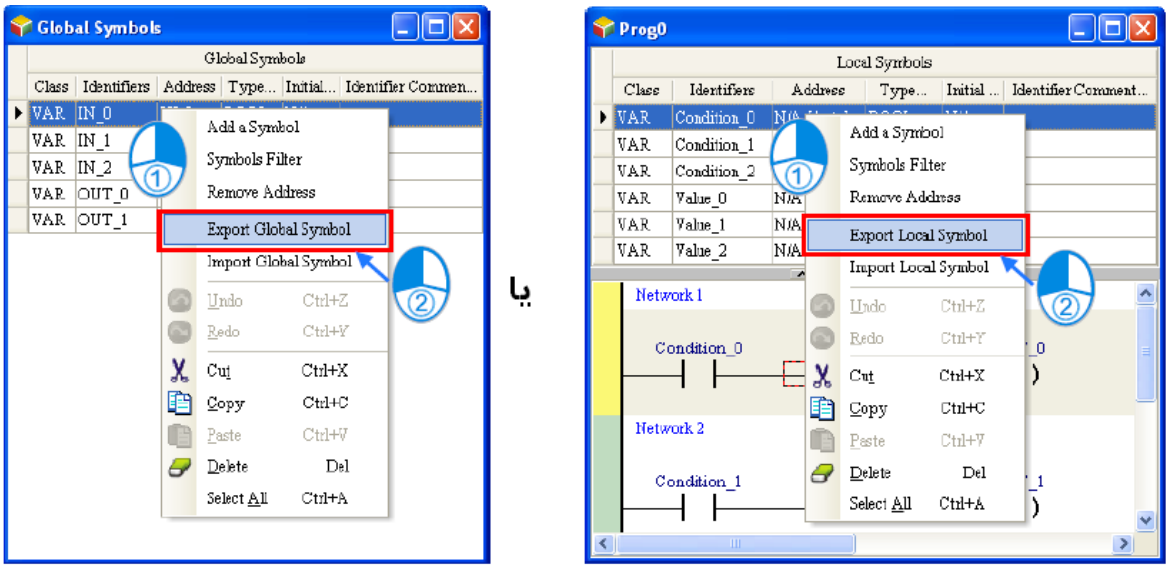
کاربران می‌توانند به سیمبول‌ها مقادیر اولیه اختصاص دهند. این مقادیر را می‌توان همراه با برنامه دانلود کرد تا به حافظه‌های آن سیمبول‌ها در PLC تخصیص داده شوند. البته این مقادیر اولیه تنها در لحظه دانلود برنامه اعمال خواهند شد و پس از قطع ارتباط، توقف اجرا، شروع مجدد و ... تاثیری در اجرای PLC نخواهند داشت. برای اینکه بتوان مقادیر اولیه را بر روی PLC بارگزاری کرد، می‌توان بر روی گزینه  کلیک کرده و در پنجره باز شده گزینه Initial Value را انتخاب و بر روی OK کلیک نمایید.



شکل ۶-۲۵ دانلود مقادیر اولیه سیمبول ها

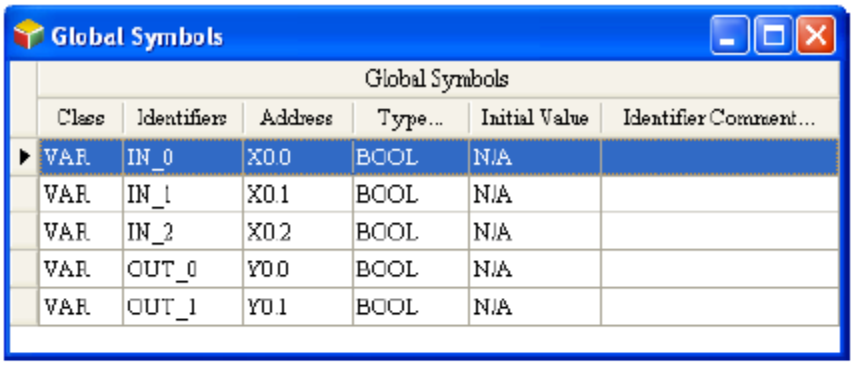
۶-۲-۶- ذخیره و بازخوانی جداول سیمبول

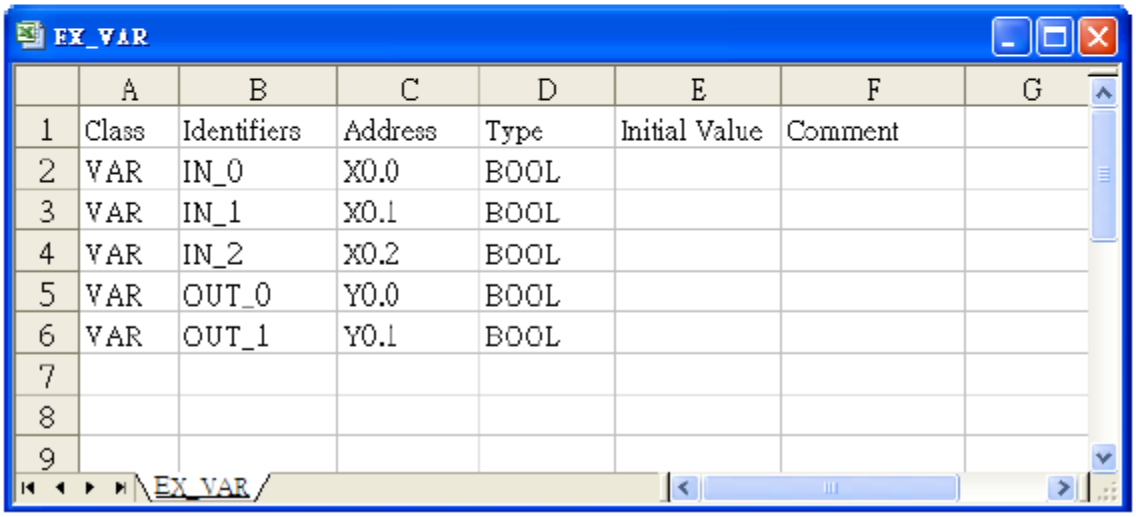
سیمبول‌های ساخته شده در هر پروژه را می‌توان در قالب فایل CSV (که می‌توان آن را در Excel باز و ویرایش کرد) ذخیره کرد. برای ذخیره کردن جدول سیمبول‌ها کاربر می‌تواند بر روی آن کلیک راست کرده و گزینه Export Global/Local Symbol را انتخاب کند.



شکل ۶-۲۶ ذخیره سازی جدول سیمبول ها

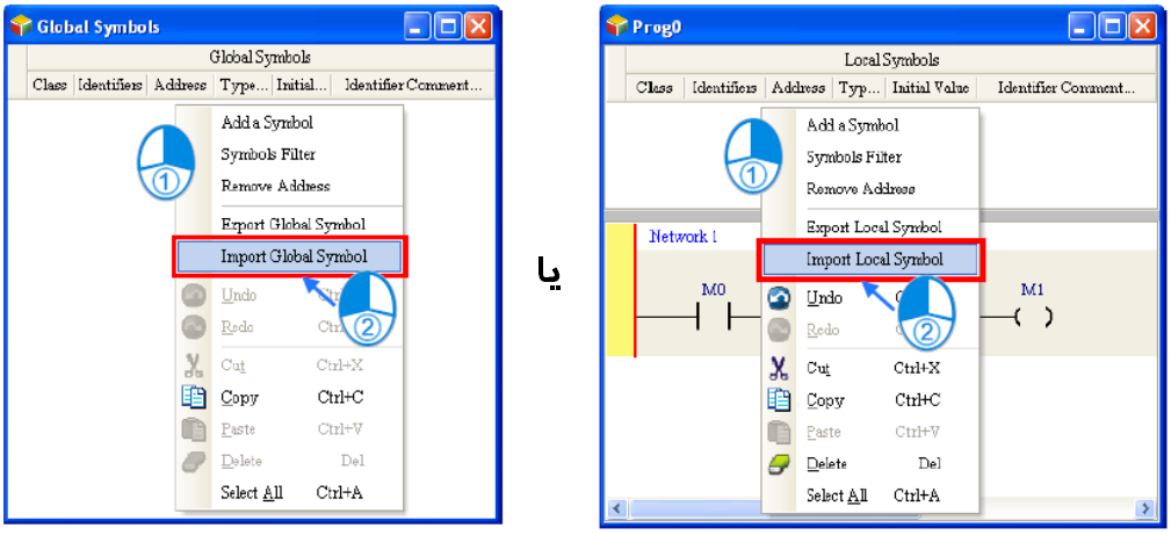
در بسیاری از پروژه‌های بزرگ بسیار راحت است تا جدول سیمبول‌ها را در محیطی مانند نرم افزار اِکسل ویرایش کرده (بسازیم) و سپس در برنامه مورد استفاده قرار دهیم.





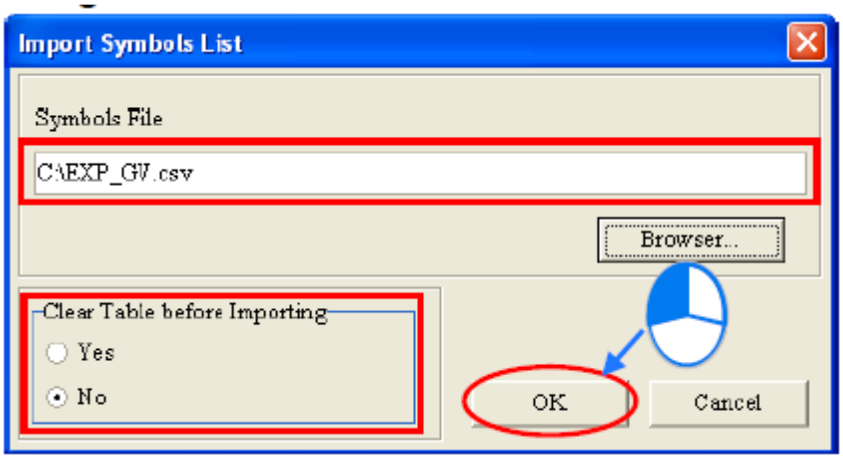
شکل ۶-۲۷ ویرایش جدول سیمبول ها در اِکسل

همچنین برای بازخوانی و استفاده از سیمبول‌های ذخیره شده در فایل می‌توان در جدول سیمبول کلیک راست کرده و گزینه Import Global/Local Symbol را انتخاب کرد.



شکل ۶-۲۸ بازخوانی جدول سیمبول ها- قسمت ۱

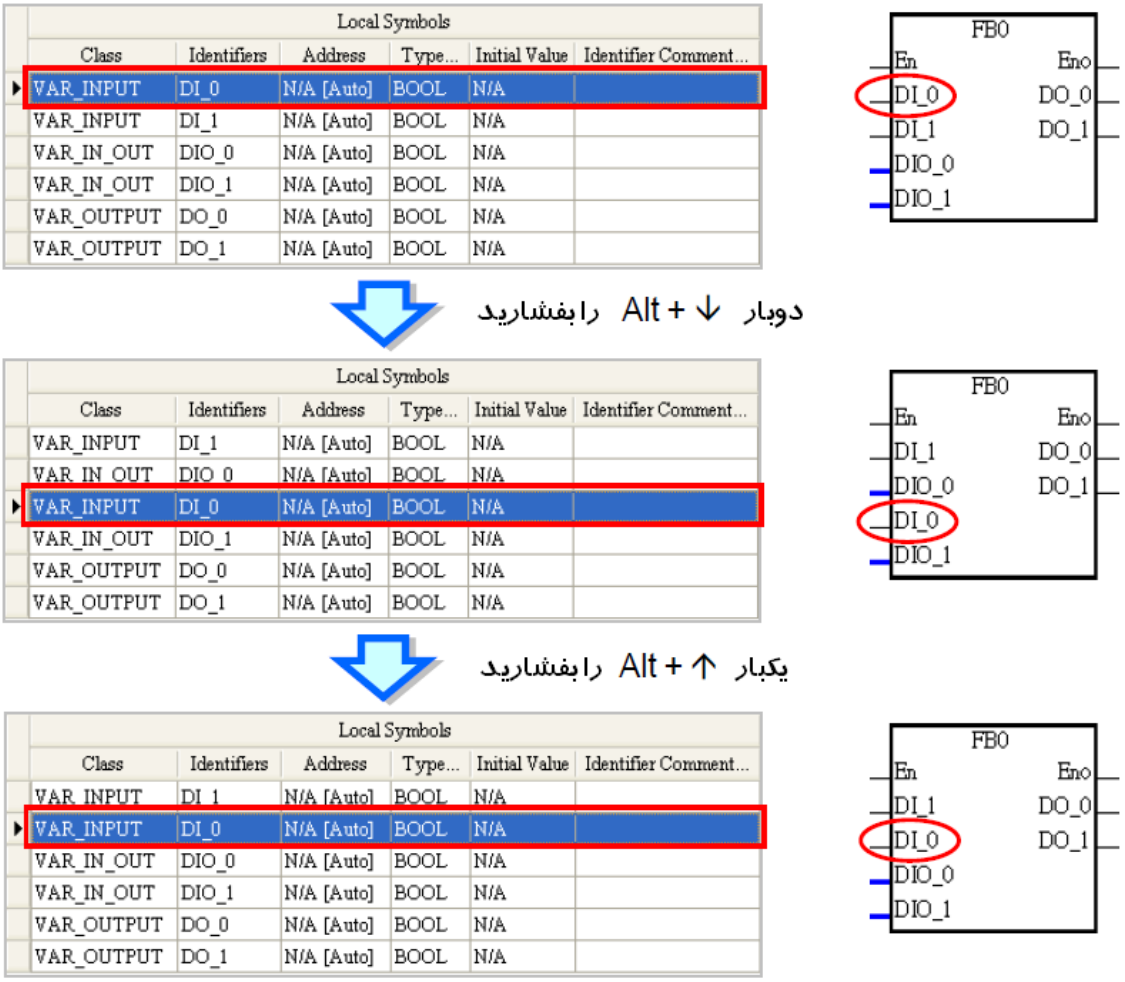
در پنجره باز شده پس از مشخص کردن آدرس فایلِ جدول سیمبول، اگر لازم است سیمبول‌های فعلی جدول سیمبول، قبل از بازخوانی پاک شود در قسمت Clear Table before Importing گزینه Yes و در غیر اینصورت No را انتخاب می‌کنیم.



شکل ۶-۲۹ بازخوانی جدول سیمبول ها- قسمت ۲

۶-۲-۷- مرتب کردن سیمبول ها

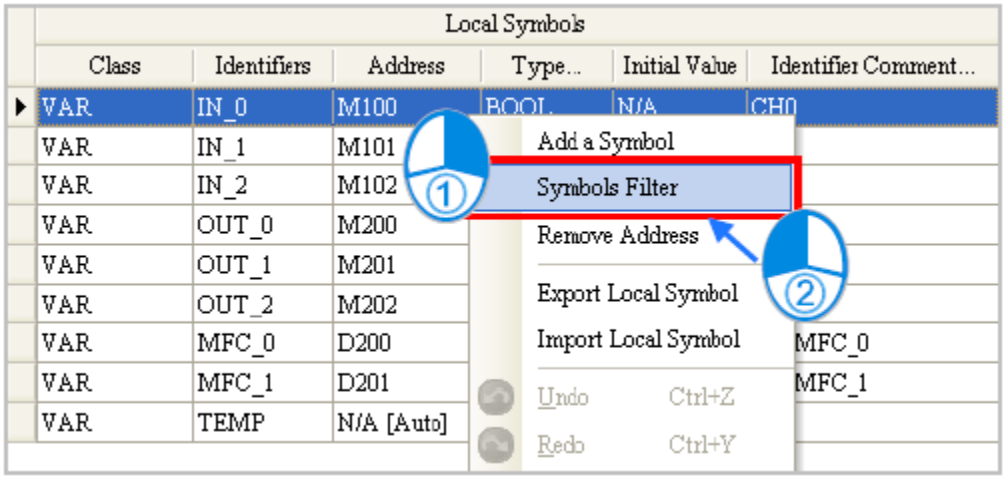
ترتیب سیمبول‌های محلی در جدول سیمبول‌های Function Block ترتیب پایه‌های بلوک را مشخص می‌کند (پایه‌های ورودی و ورودی/خروجی در سمت چپ و پایه‌های خروجی در سمت راست بلوک ایجاد خواهند شد). برای تغییر ترتیب سیمبول‌ها می‌توان بر روی آن‌ها کلیک کرده و کلید ALT را همزمان با جهت بالا/ پایین فشرد، به مثال زیر توجه کنید.



شکل ۶-۳۰ مرتب کردن سیمبول ها و تغییر ترتیب پایه بلوک ها

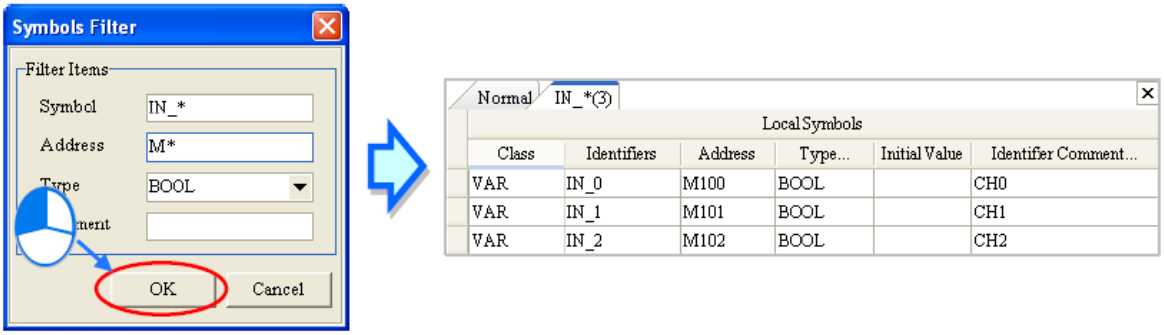
۶-۲-۸- فیلتر کردن سیمبول ها

کاربران می‌توانند سیمبول‌های مشخصی با ویژگی‌های خاص را به صورت یکجا ببینند. برای اینکار بر روی جدول کلیک راست کرده و Symbols Filter را انتخاب کنید.



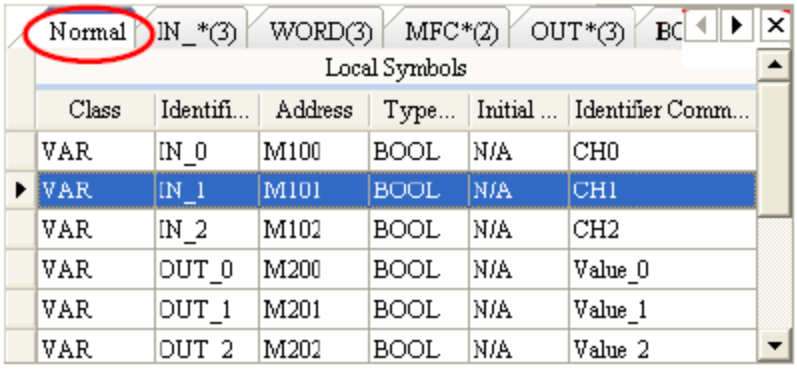
شکل ۶-۳۱ فیلتر کردن سیمبول ها – قسمت ۱

در پنجره باز شده Symbols Filter شرایط فیلتر را تعیین می‌کنیم. می‌توان از کاراکتر " * " به صورت جایگزین به جای هر حرف (یا حروفی) استفاده کرد. اگر قسمتی از این صفحه خالی بماند، در فیلتر سیمبول‌ها نقش نخواهد داشت. در نهایت نیز با کلیک بر روی OK سیمبول‌های فیلتر شده با مشخصات تعیین شده در سربرگی جدید لیست می‌شوند.



شکل ۶-۳۲ فیلتر کردن سیمبول ها – قسمت ۲

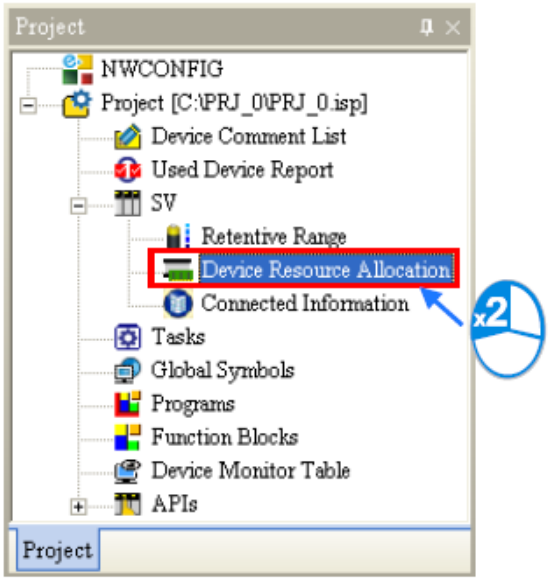
کاربر نمی‌تواند سیمبول‌ها را در صفحات فیلتر شده تغییر دهد، وی می‌بایست ابتدا به سربرگ Normal رفته و در آنجا اقدام به ویرایش سیمبول‌ها کند.



شکل ۶-۳۳ سربرگ Normal در کنار سربرگ های فیلتر شده

۶-۲-۹- تعیین محدوده حافظه های سیمبول برای سری DVP

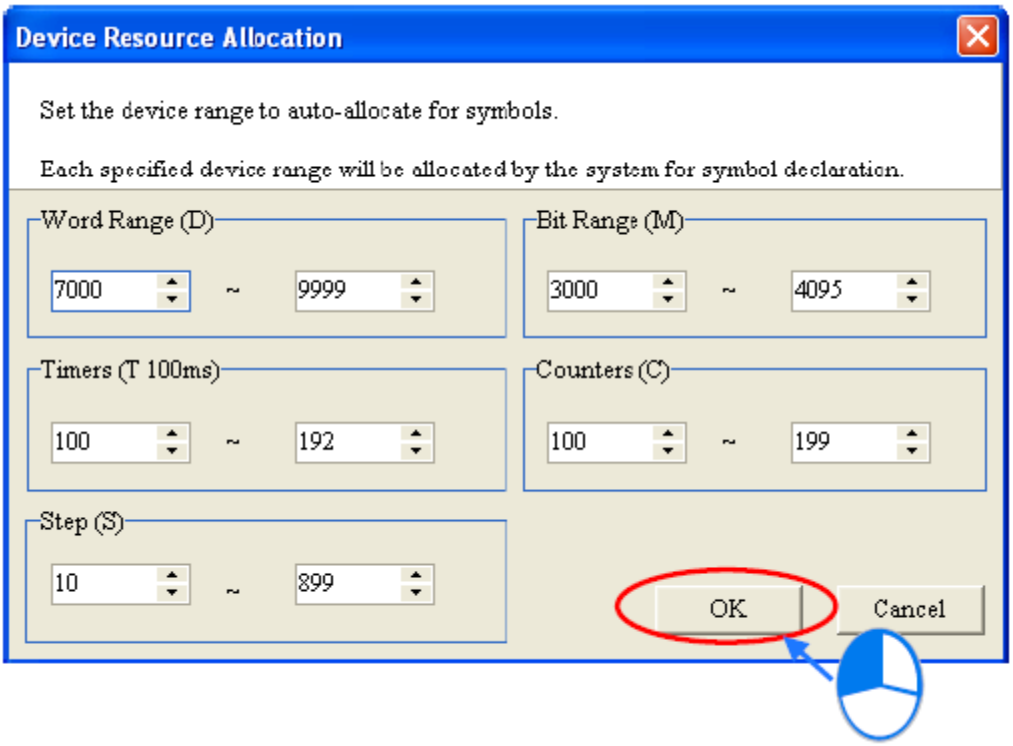
کاربران می‌توانند محدوده حافظه‌هایی که در سری DVP به صورت اتوماتیک به سیمبول‌ها اختصاص داده می‌شود را تعیین کنند. برنامه پس از کامپایل به صورت اتوماتیک به سیمبول‌هایی که به آن‌ها آدرس اختصاص داده نشده از محدوده تعیین شده حافظه اختصاص می‌دهد. اگر تعداد حافظه‌های مورد نیاز سیمبول‌ها از محدوده تعیین شده بیشتر باشد، در حین کامپایل با پیغام حافظه ناکافی مواجه خواهیم شد. در بخش مدیریت پروژه، زیر شاخه‌های بخش PLC را باز کرده و بر روی Device Resource Allocation دوبار کلیک کنید.



شکل ۶-۳۴ انتخاب Device Resource Allocation

کاربر می‌تواند محدوده حافظه‌های گوناگون را در پنجره ظاهر شده تعیین نماید. توجه شود هر نوع PLC سری DVP محدودیت‌های خاص خودش را دارد، مثلا در DVP-SV رجیسترهای داده از D2000

شروع می‌شوند، حافظه‌های بیتی از M2000 شروع می‌شوند، شمارنده‌ها ۱۶ بیتی هستند و واحدهای زمان سنج ۱۰۰ میلی ثانیه‌ای است. در صورتی که کاربر محدوده‌ها را درست به کار نبرده باشد، سیستم خود آن را تصحیح خواهد کرد. پس از کلیک بر روی OK تنظیمات ذخیره خواهد شد.



شکل ۶-۳۵ تنظیم Device Resource Allocation

۶-۳- مثال

در این بخش به تکمیل پروژه کامل شده در دو فصل گذشته با استفاده از سیمبول‌ها خواهیم پرداخت.

۶-۳-۱- جدول سیمبول ها

برای برنامه مورد نظر سیمبول‌هایی مطابق جدول زیر در نظر می‌گیریم. توجه کنید که در پروژه‌های بزرگ مهارت اختصاص سیمبول بسیار مهم می‌باشد و کمک می‌کند تا کاربر بتواند به صورت راحت و بهینه با پروژه ارتباط برقرار کند. این فرآیند میسر نمی‌شود مگر با افزایش تجربه و ایده گرفتن از الگوهایی از پیش تعیین شده.

جدول ۶-۶: مثال: اختصاص سیمبول

نام پارامتر	سیمبول	حافظه/پورت اختصاص داده	نوع	عملکرد

			سیمبول	شده	
X0.0	START_BT	X0.0	سراسری	کلید شروع	
X0.1	STOP_BT	X0.1	سراسری	کلید توقف	
X0.2	InP_SNR_1	X0.2	سراسری	سنسور موقعیت ۱	
X0.3	InP_SNR_2	X0.3	سراسری	سنسور موقعیت ۲	
X0.4	CNT_SNR	X0.4	سراسری	سنسور شمارنده	
Y0.0	CONVEYER	Y0.0	سراسری	تسمه نقاله	
Y0.1	TRIG_1	Y0.1	سراسری	تزریق کننده ۱	
Y0.2	TRIG_2	Y0.2	سراسری	تزریق کننده ۲	
M0	RUNNING	نرم افزار اتوماتیک اختصاص داد	سراسری	پرچم وضعیت	
M1	COMPLETE	نرم افزار اتوماتیک اختصاص داد	سراسری	پرچم تکمیل	
M2	ERROR	نرم افزار اتوماتیک اختصاص داد	سراسری	پرچم خطا	
D0	CNT_DT	نرم افزار اتوماتیک اختصاص داد	محلی	تعداد قطعات منتقل شده	

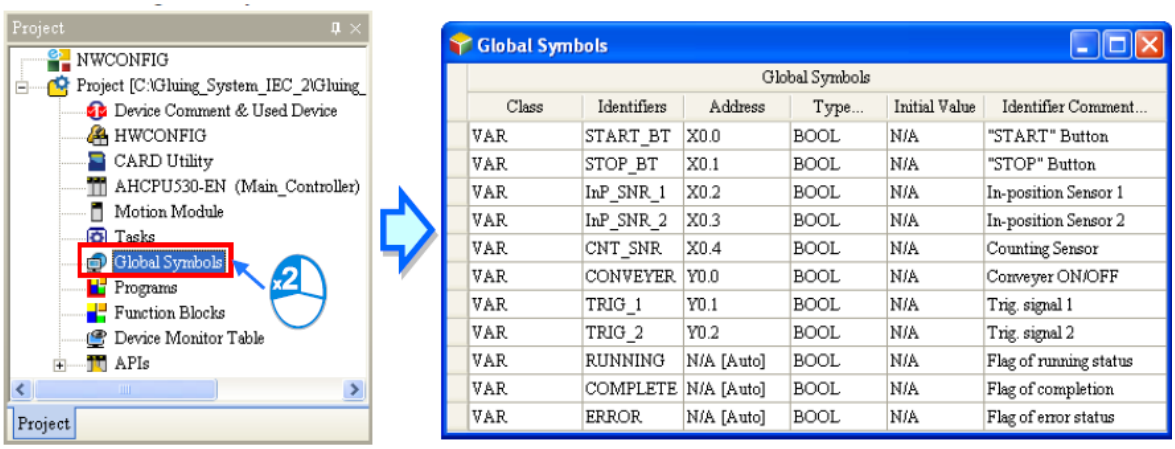
حافظه (کانتکت‌های) X و Y به صورت فیزیکی به ترتیب به پورت‌های ورودی و خروجی PLC متصل هستند. آدرس این کانتکت‌ها به نحوه سیم بندی و اتصال سیستم به PLC مرتبط است و بنابراین کاربران باید آدرس سیمبول‌های این نوع کانتکت‌ها را خودشان تعیین کنند. برای اینکه بتوان راحت‌تر سیم‌بندی و ساختار فیزیکی سیستم را تغییر داد، بهتر است سیمبول کانتکت‌های ورودی و خروجی را به صورت سراسری تعریف کرد تا بتوان به صورت متمرکز آن‌ها را ویرایش کرد.

با توجه به آنکه پرچم‌ها در برنامه فوق در POUهای مختلف مورد استفاده قرار می‌گیرند باید آن‌ها را به صورت سراسری تعریف کرد. با این حال لزومی ندارد که کاربر بخواهد آن‌ها را آدرس‌دهی کند و می‌تواند این کار را به سیستم بسپارد.

با توجه به آنکه تعداد اجزای منتقل شده در D0 ذخیره می‌شود و D0 تنها در برنامه COUNTING مورد استفاده قرار می‌گیرد، سیمبول آن را به صورت محلی تعریف می‌کنیم و سیستم خود، آدرس حافظه متناظر با آن را برابر D0 قرار می‌دهد.

۶-۳-۲- تکمیل برنامه

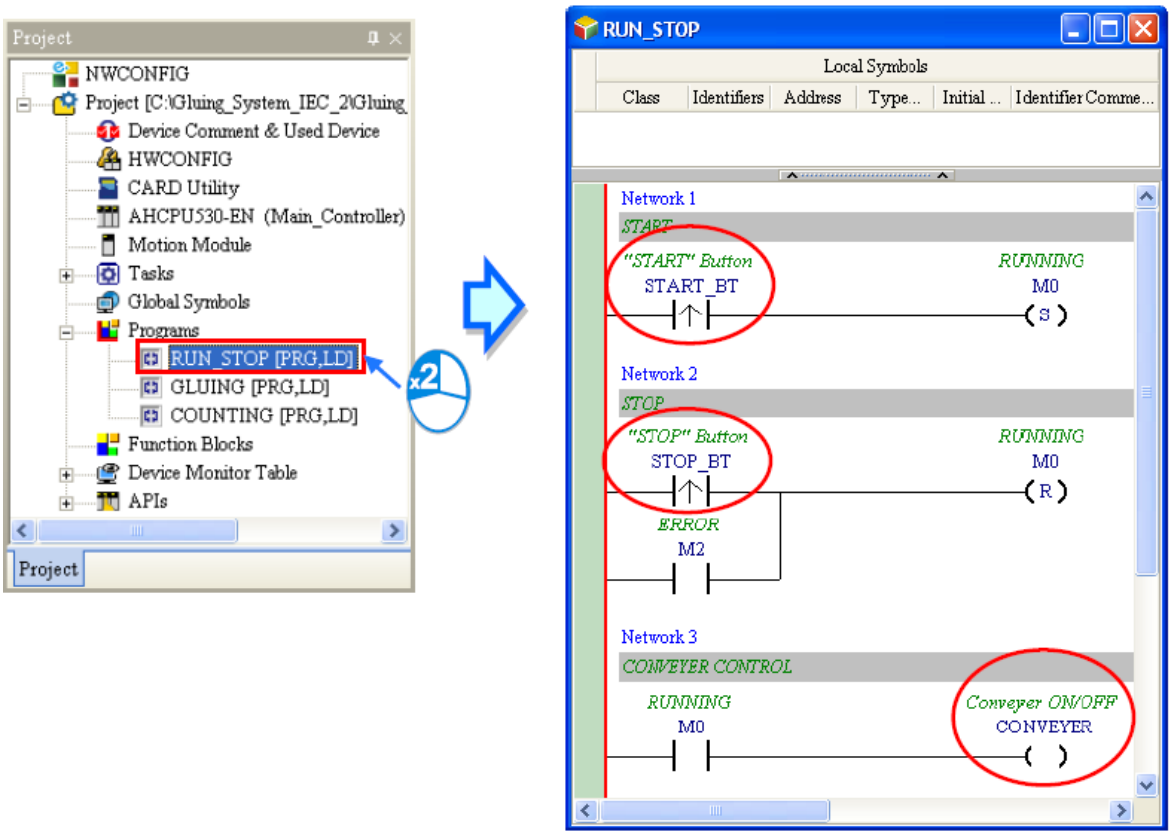
برای تخصیص سیمبول‌های سراسری در بخش مدیریت پروژه دوبار بر روی Global Symbols کلیک می‌کنیم و جدول سیمبول‌های سراسری را کامل می‌کنیم.



Class	Identifiers	Address	Type...	Initial Value	Identifier Comment...
VAR	START_BT	X0.0	BOOL	N/A	"START" Button
VAR	STOP_BT	X0.1	BOOL	N/A	"STOP" Button
VAR	InP_SNR_1	X0.2	BOOL	N/A	In-position Sensor 1
VAR	InP_SNR_2	X0.3	BOOL	N/A	In-position Sensor 2
VAR	CNT_SNR	X0.4	BOOL	N/A	Counting Sensor
VAR	CONVEYER	Y0.0	BOOL	N/A	Conveyer ON/OFF
VAR	TRIG_1	Y0.1	BOOL	N/A	Trig. signal 1
VAR	TRIG_2	Y0.2	BOOL	N/A	Trig. signal 2
VAR	RUNNING	N/A [Auto]	BOOL	N/A	Flag of running status
VAR	COMPLETE	N/A [Auto]	BOOL	N/A	Flag of completion
VAR	ERROR	N/A [Auto]	BOOL	N/A	Flag of error status

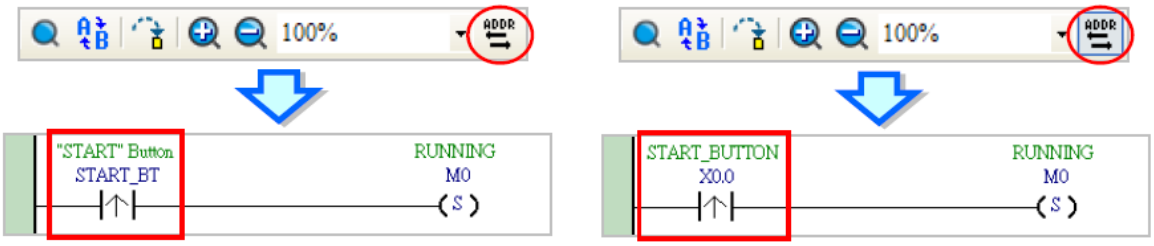
شکل ۶-۳۶ مثال: تکمیل جدول سیمبول های سراسری

در این مرحله با کلیک بر روی یکی از POUها مانند RUN_STOP می‌بینیم که سیمبول‌ها جایگزین نام کانتکت‌ها و کامنت (توضیحات) سیمبول‌ها جایگزین کامنت کانتکت‌ها شده‌اند.



شکل ۶-۳۷ مثال: جایگزین شدن سیمبول ها به جای نام کانتکت ها

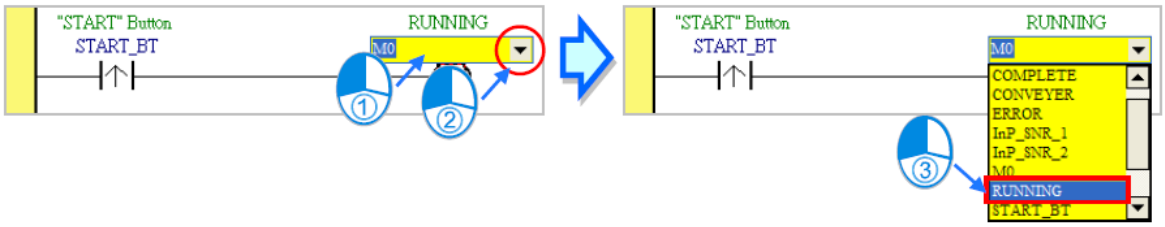
در صورتی که سیمبول‌ها نمایش داده نشده‌اند لازم است کاربر تنظیمات سیستم را تغییر دهد. در زبان Ladder کاربر می‌تواند انتخاب کند که آدرس‌ها و یا سیمبول‌ها را در برنامه مشاهده کند. برای اینکه کاربر آدرس و کامنت حافظه‌ها را ببیند می‌تواند ADDR را بر روی نوار ابزار بفشارد و در صورتی که بخواهد سیمبول‌ها و کامنت سیمبول‌ها را ببیند باید گزینه ADDR را از حالت انتخاب در بیاورد.



شکل ۶-۳۸ تغییر وضعیت حالت نمایش سیمبول و آدرس دهی مستقیم

توجه کنید که چون حافظه M0 به سیمبول مشخصی تخصیص داده نشده، وضعیت آن پس از فشرده شدن ADDR تغییری نمی‌کند. برای همین می‌توان آن را با سیمبول RUNNING نیز جایگزین کرد (که

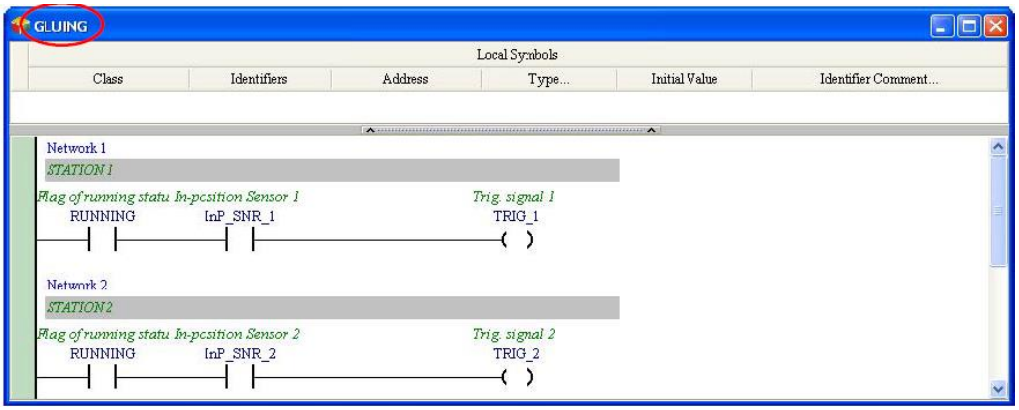
البته آدرس آن توسط سیستم اختصاص داده می‌شود). برای اینکار می‌توانیم بر روی نام M0 کلیک و از لیست سیمبول‌ها RUNNING را انتخاب کنیم.



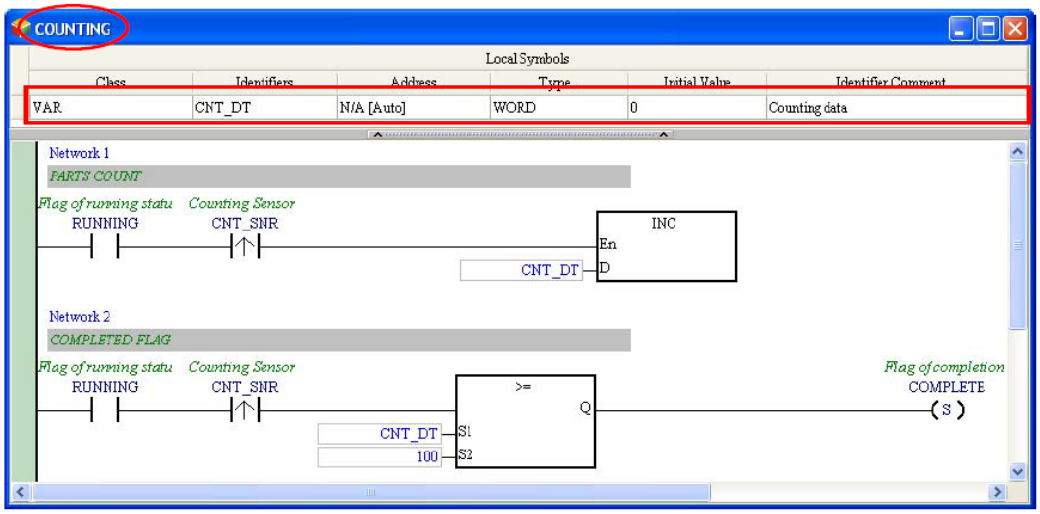
شکل ۶-۳۹ مثال: اختصاص سیمبول به پارامترهای سیستم

در نتیجه نمایش برنامه‌ها به صورت زیر تغییر خواهند کرد.

شکل ۶-۴۰ مثال: برنامه RUN_STOP پس از اضافه شدن سیمبول ها

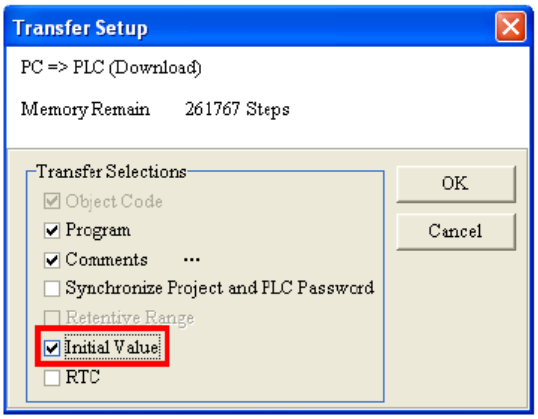


شکل ۶-۴۱ مثال: برنامه GLUING پس از اضافه شدن سیمبول ها



شکل ۶-۴۲ مثال: برنامه COUNTING پس از اضافه شدن سیمبول ها

در این مرحله برای آنکه مقدار اولیه تخصیص داده شده به CNT_DT در برنامه COUNTING در CPU اعمال شود، برنامه را در حالی که گزینه Initial Value را در صفحه Transfer Setup فعال کرده‌ایم در PLC بارگزاری می‌کنیم.



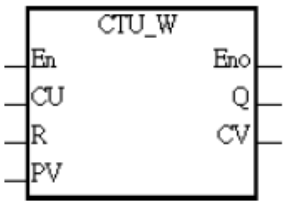
شکل ۶-۴۳ نوار وضعیت در حالت آنلاین

فصل ۷– بلوک ها

در این فصل در مورد توابع بلوکی[۱] و یا به طور مختصر بلوک‌ها (FB) صحبت خواهیم کرد. ISPSoft برخلاف نرم افزار WPLSoft، امکان استفاده از توابع بلوکی است را برای کاربران به صورت بهینه و ساختاریافته فراهم کرده است.

۷–۱–۱– مقدمه ای بر توابع بلوکی

تابع بلوکی جزئی از برنامه است که عملیات مشخصی را انجام می‌دهد. با آنکه بلوک‌ها خود به تنهایی قادر به اجرا نیستند، زمانی که در برنامه اصلی فراخوانی می‌شوند و پارامترهای ورودی به آن‌ها داده می-شود، اجرا شده و نتایج آن به برنامه اصلی ارسال می‌شود. همچنین هر بلوکی خود می‌تواند بلوک دیگری را فراخوانی کند.

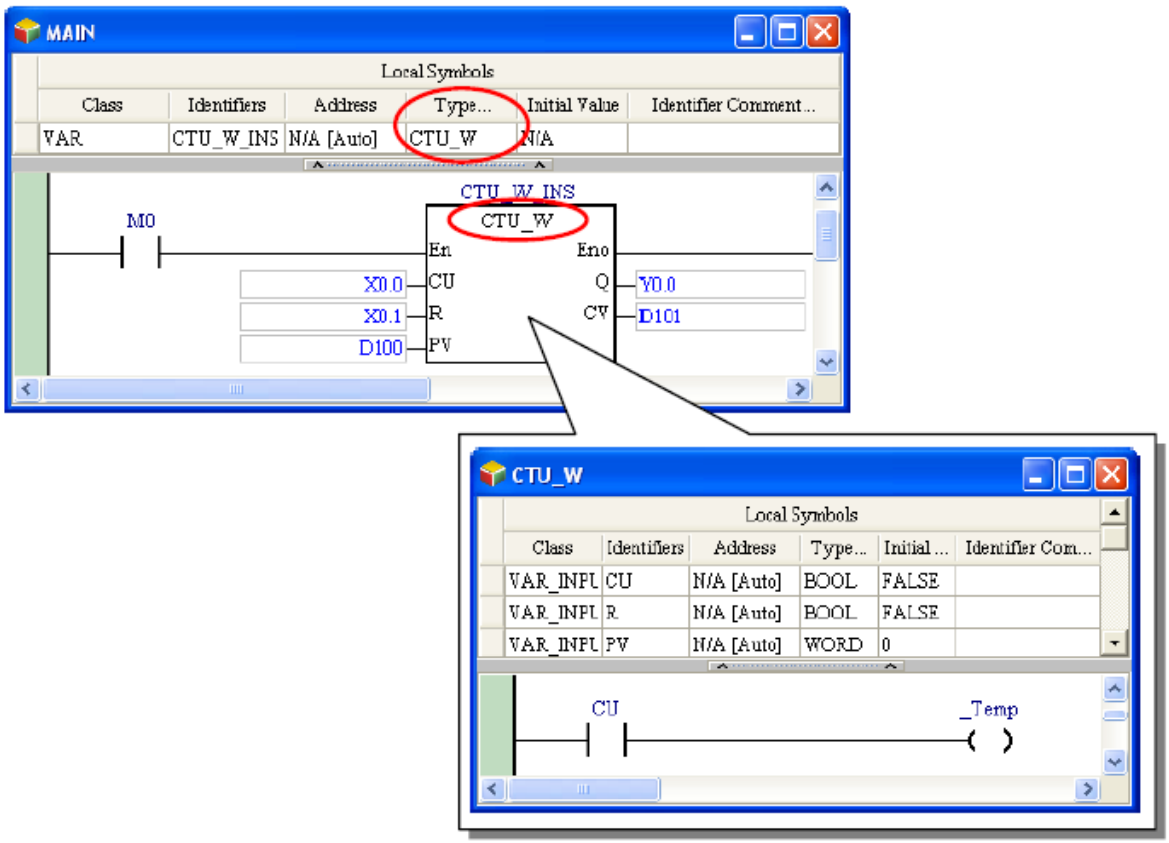


شکل ۷-۱ نمونه ای از توابع بلوکی

هربلوک شامل تعدادی حافظه است که سیمبول‌های محلی در آن ذخیره می‌شوند. با توجه به آنکه مقادیر سیمبول‌ها پس از هر بار اجرا در حافظه باقی می‌ماند، می‌توانند در اجرای بعدی تاثیرگذار باشند. به عبارت دیگر ممکن است به ازای ورودی‌های یکسان بلوک، خروجی‌های مختلفی داشته باشیم چرا که سیمبول‌های ذخیره شده از اجرای‌های قبل نیز در برنامه تاثیرگذار هستند.

مثال زیر شامل فراخوانی یک بلوک در برنامه اصلی است. در برنامه اصلی نام بلوک به صورت سیمبول تعریف شده و نوع سیمبول نیز همانند نام بلوک است.

[۱] Function blocks



شکل ۷-۲ مثالی از فراخوانی توابع بلوکی

۷-۱-۲- ویژگی های توابع بلوکی

توابع بلوکی از استاندارد IEC 61131-3 تبعیت می‌کنند و کمک می‌کنند تا بتوان برنامه‌های PLC را به صورت ساختاریافته گسترش داد. در ادامه تعدادی از ویژگی‌های توابع بلوکی را مرور خواهیم کرد.

- طراحی ماژولار

برنامه‌های بزرگ را می‌توان به چند بخش دارای زیربرنامه و کوچکتر تقسیم کرد و تابع هر زیربرنامه را در یک بلوک نوشت و اینگونه برنامه نویسی مرتب تری داشت.

- وابسته نبودن

کُدهای بلوک‌های مختلف را می‌توان به زبان‌های مختلف و به صورت مستقل از هم نوشت. در هر برنامه‌ای نیز می‌توان بلوک‌های نوشته شده به زبان‌های مختلف را فراخوانی کرد.

- قابلیت چندبار استفاده

هر بلوک را می‌توان به صورت مکرر در یک و یا چند پروژه استفاده کرد.

- قابل کپی و انتقال[1]

هر بلوک به جز پروژه‌ای که در آن مورد استفاده قرار می‌گیرد می‌تواند استخراج و در دیگر پروژه‌ها و دیگر کامپیوترها مورد استفاده قرار گیرد.

- پشتیبانی آسان

در صورتی که خطایی که در بلوک رخ دهد، به طور مستقل از برنامه اصلی می‌توان آن را تست و عیب یابی کرد.

- افزایش خوانایی برنامه

کاربران می‌توانند قسمت‌های پیچیده برنامه را که دائما تکرار می‌شوند را در بلوک نوشته و اینگونه برنامه قابل فهم‌تر و کار با آن راحت‌تر می‌شود.

- امنیت بالا

می‌توان با اختصاص کلمه عبور به بلوک‌ها از دستیابی دیگران به جزئیات کُدهای نوشته شده جلوگیری کرد. این مورد برای افرادی که نمی‌خواهند مشتریان و یا دیگر کاربران از برنامه آن‌ها سر در بیاورند، همچنین در مواردی که ایجاد تغییر بدون در نظر گرفتن مسائل ایمنی خطرناک می‌باشد، مناسب است.

- بهینگی

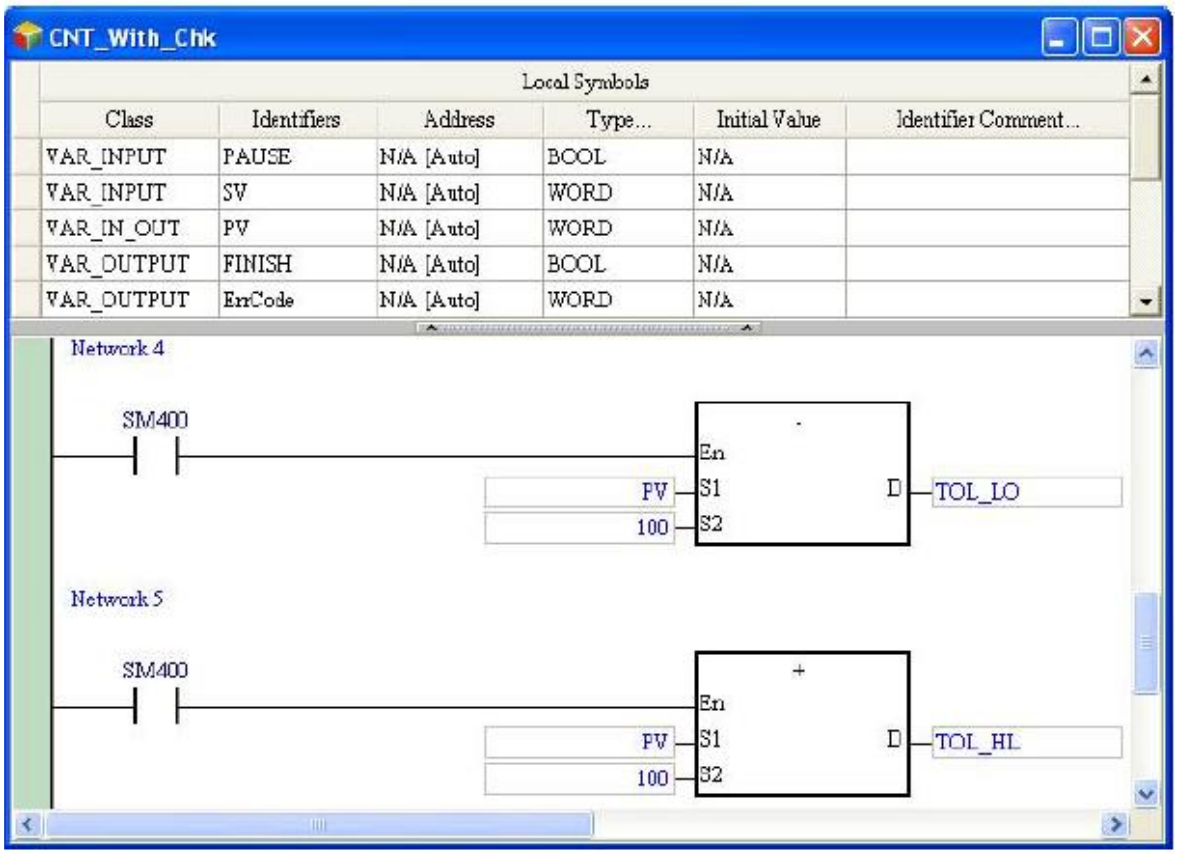
عملکرد بلوک‌ها قابلیت گسترش برنامه را بهینه کرده و امکان مشارکت افراد، سازمان‌ها و گروه‌های مختلف را در توسعه یک برنامه فراهم می‌آورد.

۷–۲– ساختار توابع بلوکی در ISPSoft

پنجره برنامه بلوک همانند پنجره ویرایش برنامه اصلی است. در این پنجره بخش بالایی مربوط به اطلاعات سیمبول‌های محلی و بخش پایین مربوط به محیط ویرایش برنامه است. نحوه کار و ویرایش برنامه در بلوک همانند محیط برنامه اصلی است و به همه زبان‌ها به جز SFC می‌توان در آن برنامه نوشت.

[1] Portable

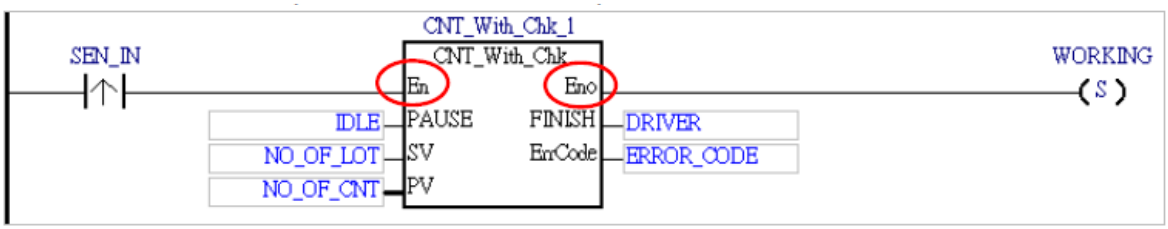
در بلوک‌ها می‌توان دیگر بلوک‌های را فراخوانی کرد و حین فراخوانی زبانی که با آن برنامه بلوک نوشته شده است مهم نخواهد بود.



شکل ۷-۳ مثالی از پنجره یک تابع بلوکی

۷–۲–۱– پایه En در بلوک

اجرای بلوک در ISPSoft به ورودی پایه En وابسته است، اگر این پایه یک باشد بلوک اجرا خواهد شد و در غیر این صورت بلوک اجرا نخواهد شد. پایه خروجی Eno نیز همان منطق ورودی En را به خروجی منتقل می‌کند (برای استفاده بلوک‌های دیگر در شبکه). در زبان برنامه نویسی Structured Texts نیازی به فعال کردن En برای اجرای بلوک نیست.



شکل ۷-۴ پایه En و Eno در توابع بلوکی

۷-۲-۲- سیمبول ها در بلوک

بلوک‌ها همانند برنامه اصلی قابلیت استفاده از سیمبول‌های محلی را دارند. در بلوک‌ها می‌توان از سیمبول‌های سراسری نیز استفاده کرد ولی باید توجه داشت که در این صورت استفاده از بلوک مورد نظر در پروژه‌های دیگر منوط به تعریف همان سیمبول های سراسری خواهد بود.

انواع کلاس سیمبول که می‌توان در بلوک‌ها تعریف کرد به صورت زیر است.

جدول ۷-۱: انواع کلاس سیمبول ها در توابع بلوکی

کلاس	شرح
VAR	سیمبول‌های این کلاس برای عملیات‌های درون بلوک می‌تواند مورد استفاده قرار گیرد و مقدار سیمبول بعد از اجرا حفظ می‌شود.
VAR_INPUT	سیمبول‌های این کلاس برای تعریف ورودی‌های بلوک مورد استفاده قرار می‌گیرند. این سیمبول‌ها برای اجرای بلوک، ورودی‌ها را از برنامه گرفته و در حافظه‌های ورودی بلوک کپی می‌کنند.
VAR_OUTPUT	سیمبول‌های این کلاس برای تعریف خروجی‌های بلوک مورد استفاده قرار می‌گیرند. این سیمبول‌ها برای ارسال خروجی‌های بلوک به برنامه‌ای که بلوک را فراخوانی کرده مورد استفاده قرار می‌گیرد.
VAR_IN_OUT	این سیمبول برای ورودی/خروجی FB به صورت محلی مورد استفاده قرار می‌گیرد. این کلاس از سیمبول برای ایجاد بازخورد[۱] است. زمانی که FB فراخوانی می‌شود، مقدار ورودی از برنامه خوانده (سیمبول کلاس نوع VAR_IN_ OUT) و پس از انجام عملیات بلوک، مقدار خروجی به برنامه فرستاده می‌شود.

جدول ۷-۲: انواع داده در توابع بلوکی

مدل	شرح
AH500	BOOL, WORD, DWORD, LWORD, INT, DINT, LINT, REAL, LREAL, ARRAY,

[۱] Feedback

	STRING, POINTER, T_POINTER, C_POINTER, HC_POINTER, function block
DVP	BOOL, WORD, DWORD, LWORD, REAL, ARRAY, STEP, TIMER, COUNTER
نوع داده کلاس‌های VAR_INPUT و VAR_OUTPUT و VAR_IN_OUT نمی‌تواند STRING، STEP، COUNTER و یا TIMER باشد.	

جدول ۷-۳: اختصاص حافظه در توابع بلوکی

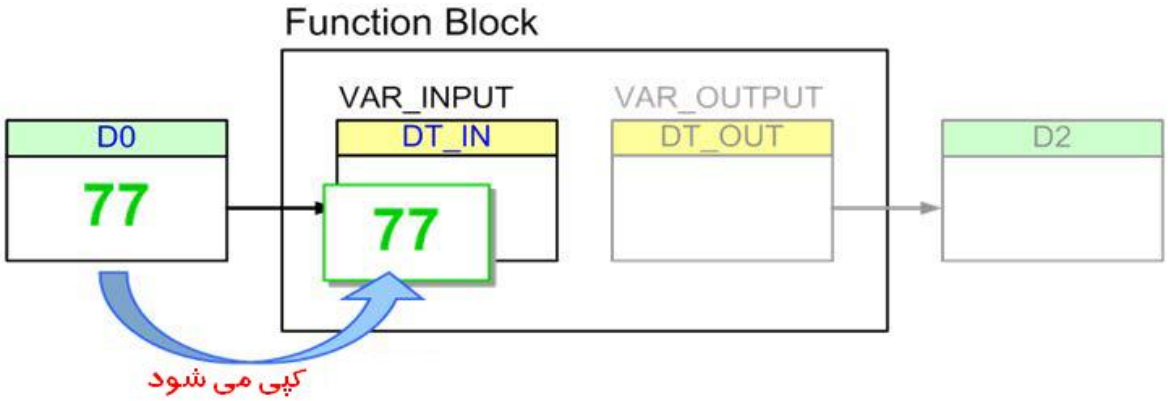
مدل	شرح
AH500	در این مدل کاربران نیازی به اختصاص حافظه به سیمبول‌های محلی درون بلوک و سیمبول نام بلوک ندارند و سیستم به صورت اتوماتیک حافظه‌های رزرو شده خود را به سیمبول‌ها اختصاص می‌دهد.
DVP	به سیمبول نام بلوک باید حافظه نوع P اختصاص داده شود. همچنین سیستم به صورت اتوماتیک می‌تواند به سیمبول‌های محلی بر اساس نوع آن‌ها حافظه اختصاص دهد.

۷-۲-۳- ورودی و خروجی‌های بلوک

زمانی که POU بلوکی را فراخوانی می‌کند، مقادیر حافظه‌های متصل به پایه‌های ورودی بلوک را به بلوک ارسال می‌کند. پس از اجرای بلوک، مقادیر محاسبه شده در بلوک به حافظه‌ی متصل به پایه‌های خروجی منتقل می‌شود. به نمونه زیر توجه کنید:

- قبل از اجرای بلوک

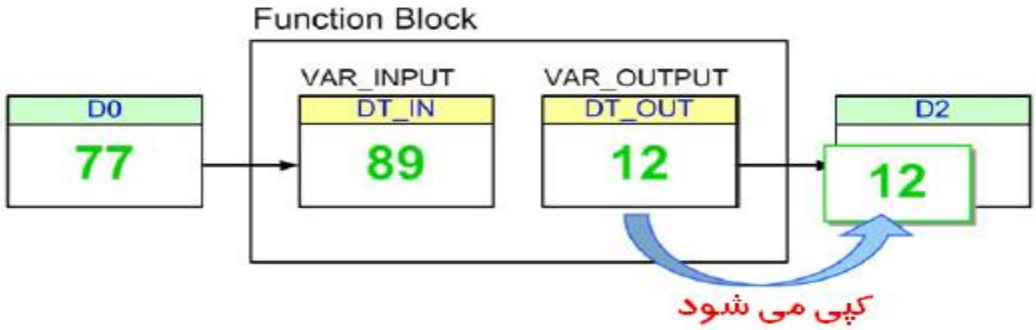
D0 به پایه ورودی DT_IN که پایه ورودی بلوک است تخصیص داده شده است. زمانی که بلوک فراخوانی می‌شود، سیستم مقدار فعلی D0 را به DT_IN منتقل می‌کند. (DT_IN سیمبول نوع VAR_INPUT است)



شکل ۷-۵ ورودی بلوک و نحوه عملکرد رجیسترهای ورودی

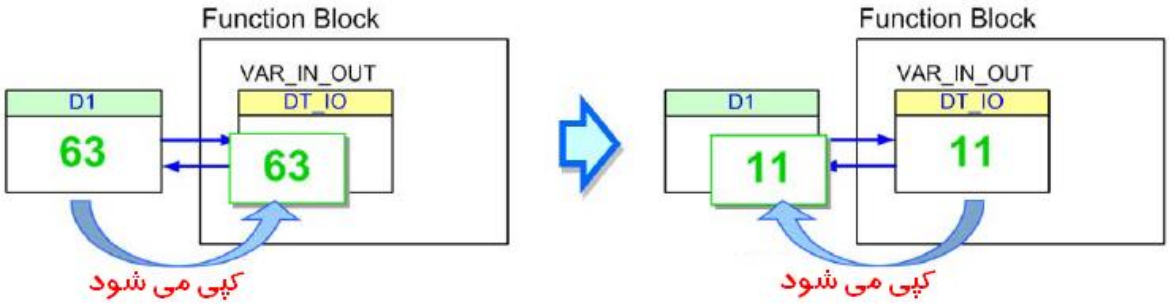
- پس از اجرای بلوک

پس از اجرای بلوک مقدار DT_IN برابر ۸۹ می‌شود، مقدار D0 ثابت می‌ماند. مقدار D0 ثابت می‌ماند و DT_OUT مقدارش پس از اجرای بلوک به D2 منتقل می‌شود. حتی اگر مقدار D2 حین اجرای برنامه تغییر کند، مقدار DT_OUT همچنان ثابت می‌ماند. (DT_OUT سیمبول کلاس VAR_OUTPUT است.)



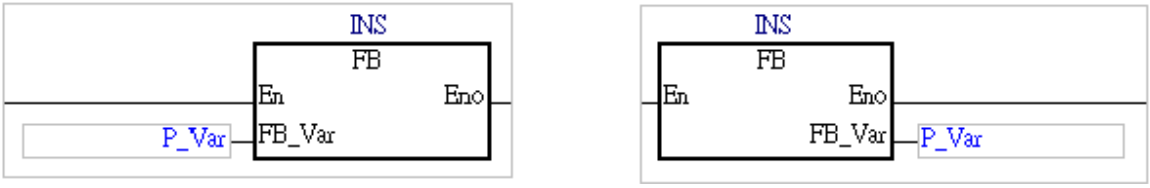
شکل ۷-۶ خروجی بلوک و نحوه عملکرد رجیسترهای خروجی بلوک

- مقادیر سیمبول‌های کلاس VAR_IN_OUT نیز قبل از اجرای بلوک، مقدار را از حافظه گرفته، و در نهایت نتایج محاسبات را در همان حافظه می‌ریزد.



شکل ۷-۷ نحوه عملکرد ورودی/ خروجی‌های بلوک

به بلوک‌های زیر توجه کنید، FB_VAR پایه بلوک و P_VAR حافظه تخصیص داده شده به آن است. نوع داده FB_VAR و P_VAR باید مطابق با قواعدی باشد که در ادامه شرح داده می‌شوند. حتی اگر نوع داده‌ها دقیقا یکسان هم نباشد، در صورتی که از قواعد خاص شرح داده شده در ادامه این بخش پیروی کنند، نرم افزار خطایی برای آن در نظر نخواهد گرفت.



شکل ۷-۸ بررسی تطابق حافظه ورودی/خروجی و پایه ورودی/خروجی متناظر در تابع بلوکی

- قواعد عمومی تطابق داده‌ها

نوع داده P_VAR و FB_VAR باید یکسان باشند.

- قواعد خاص تطابق داده‌ها

اگر داده P_VAR و یا FB_VAR از نوع WORD ویا DWORD و یا LWORD باشند، داده سیمبول دیگر می‌تواندWORD، DWORD، LWORD، INT، DINT، LINT، REAL و یا LREAL باشد. در این صورت داده نوع P_VAR می‌بایست طولش بزرگتر و یا مساوی FB_VAR باشد.

اگر داده P_VAR از نوع STEP باشد، داده FB_VAR می‌تواند از نوع BOOL باشد. وضعیت Step در فراخوان آن، در بلوک مورد استفاده قرار می‌گیرد.

جدول ۷-۴: نحوه تطابق داده های حافظه ورودی/خروجی و پایه متناظر آن

نوع داده		FB_VAR(بلوک)								
		WORD	DWORD	LWORD	INT	DINT	LINT	REAL	LREAL	BOOL
P_VAR (فراخوان)	WORD	√			√					
	DWORD	√	√		√	√		√		

	√	√	√	√	√	√	√	√	LWORD
					√			√	INT
				√			√	√	DINT
			√			√	√	√	LINT
		√					√	√	REAL
	√					√	√	√	LREAL
√									BOOL
√									STEP

قواعد مرتبط با سیمبول‌های نوع آرایه نیز قواعد خاص خود را دارند.

- قواعد عمومی تطابق داده‌ها در آرایه‌ها

۱- داده آرایه P_Var باید مشابه داده آرایه FB_Var باشد.

۲- در صورتی که نوع آرایه P_Var مشابه داده آرایه FB_Var باشد، اندازه آرایه P_VAR می‌تواند بزرگتر از اندازه آرایه FB_VAR باشد.

- قواعد خاص تطابق داده‌ها در آرایه‌ها

۱- فرمت داده آرایه‌ها باید مطابق با "قواعد خاص تطابق داده‌ها" باشد.

۲- در صورتی که نوع داده P_Var و یا FB_Var آرایه باشد، سیستم نوع داده‌ی بقیه سیمبول‌ها را نیز آرایه‌ای با طول یک در نظر می‌گیرد.

۳- اگر داده P_VAR و یا FB_VAR از نوع WORD ویا DWORD و یا LWORD باشند، داده سیمبول دیگر می‌تواندWORD، DWORD، LWORD، INT، DINT، LINT، REAL و یا LREAL باشد. داده نوع P_VAR می‌تواند بزرگتر و یا مساوی FB_VAR باشد.

*** کاربران طول داده‌ها را می‌توانند با استفاده از مفهوم مجموع طول داده‌ها مورد ارزیابی قرار دهند (در صورتی که تطابق نوع داده‌ها برقرار باشد). برای مثال می‌توان

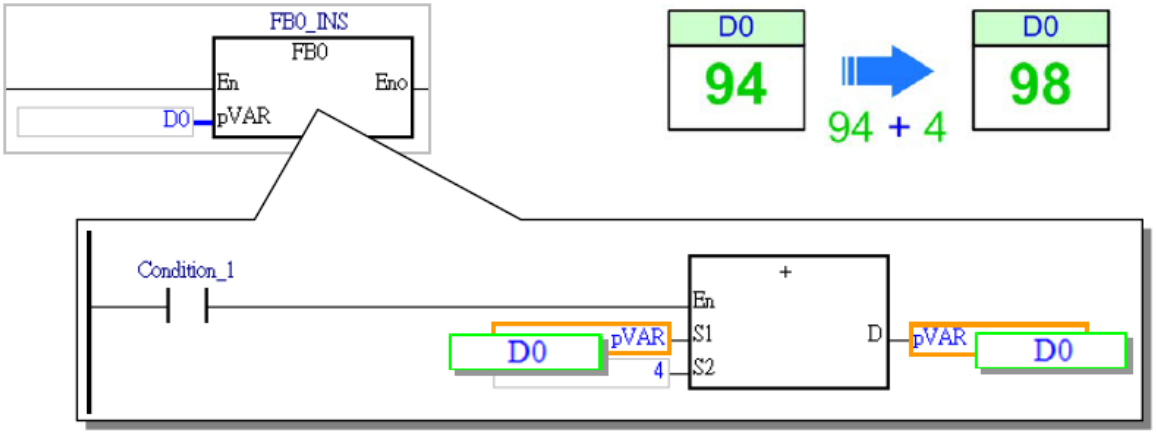
یک سیمبول با فرمت داده DWORD را منطبق بر سیمبولی که به صورت آرایه INT تعریف شده در نظر گرفت با این شرط که طول آرایه دو باشد (تا طول داده‌ها یکسان شود). همچنین داده آرایه WORD را می‌توان منطبق با داده LINT در نظر گرفت در صورتی که طول آرایه WORD برابر چهار باشد. اما داده INT و داده DINT را هیچگاه نمی‌توان با هم منطبق کرد چون نوع داده‌های آن‌ها منطبق نیست (چون حداقل یکی از داده‌ها باید از نوع WORD باشند).

*** در صورتی که قواعد فوق برقرار باشد و طول داده P_Var بزرگتر از FB_Var باشد، داده‌های با بیت کم ارزش‌تر بر هم منطبق می‌شوند. در صورتی که مقدار P_Var به FB_Var ارسال شود، بیت‌های کم ارزش آن (که در FB_Var جا می‌شود) به FB_Var منتقل می‌شود. در صورتی که داده از FB_Var به P_Var منتقل شود، بر روی قسمت کم ارزش آن نشسته و بخش پر ارزش P_Var را تغییر نمی‌دهد، به همین علت باید توجه شود که مقادیر باقی مانده خطایی در محاسبات ایجاد نکند.

۷-۲-۴- سیمبول‌های با فرمت داده اشاره گر

اشاره‌گرها به جای داده در برگیرنده آدرس محلی که داده‌ها در آن ذخیره شده‌اند می‌باشد. سیمبولی که داده آن از نوع اشاره‌گر (POINTER یا T_POINTER یا C_POINTER یا HC_POINTER) باشد، عملکردش ارسال آدرس حافظه و یا سیمبول به تابع بلوکی است. استفاده از سیمبول‌هایی که نوع آن‌ها POINTER یا T_POINTER یا C_POINTER یا HC_POINTER است کاملا نسبت به سیمبول‌هایی که نوع داده‌های آن "داده" است متفاوت است. سیمبول‌های در برگیرنده داده‌های عمومی در توابع بلوکی، ورودی‌ها را از پایه‌های ورودی می‌گیرند و یا داده‌های خود را به پایه‌های خروجی بلوک ارسال می‌کنند. اما اگر داده‌ی پایه‌ی تابع بلوکی از نوع اشاره‌گر (نوع POINTER یا T_POINTER یا C_POINTER یا HC_POINTER) باشد، داده ارسال شده به پایه آن در واقع سیمبول یا حافظه‌ای شامل آدرس حافظه‌ای دیگر خواهد بود.

برای نمونه در مثال زیر، برنامه اصلی D0 را به پایه pVAR در تابع بلوکی اختصاص می‌دهد، نوع داده pVAR در تابع بلوکی نوع POINTER یا T_POINTER یا C_POINTER یا HC_POINTER است و زمانی که تابع بلوکی فراخوانی و اجرا شود، pVAR در تابع بلوکی، به عنوان تابع نماینده D0 عمل می‌کند (در واقع آدرس D0 را ذخیره و سپس هرجا که استفاده شود از D0 در عملیات استفاده می‌کند.



شکل ۷-۹ سیمبول هایی با فرمت داده اشاره‌گر

مواردی که در استفاده از اشاره‌گرها باید به آن‌ها توجه کرد به صورت زیر است:

۱- سیمبول‌های اشاره‌گر نوع نوع POINTER یا T_POINTER یا C_POINTER یا HC_POINTER را تنها PLCهای سری AH500 پشتیبانی می‌کند.

۲- سیمبول‌های با فرمت POINTER تنها در توابع بلوکی و به عنوان سیمبول کلاس VAR_IN_OUT قابل استفاده هستند.

۳- محدودیت انواع مختلف اشاره‌گرها در جدول زیر شرح داده شده است.

جدول ۷-۵ محدودیت انواع مختلف اشاره گرها

اشاره‌گر	محدودیت ها	
اشاره‌گر عمومی POINTER	بیشترین تعداد	در هر تابع بلوکی ۱۶ سیمبول با فرمت POINTER قابل استفاده است.
	مرجع	سیمبول با فرمت WORD /DWORD /LWORD /INT /DINT /LINT ، رجیستر داده و Link و یا پورت‌های ورودی/خروجی
اشاره‌گر زمان	بیشترین تعداد	در هر تابع بلوکی ۸ سیمبول با فرمت T_POINTER قابل استفاده است.

سنج **T_POINTER**	مرجع	زمان سنج و یا سیمبول با داده نوع زمان‌سنج
اشاره‌گر شمارنده **C_POINTER**	بیشترین تعداد	در هر تابع بلوکی ۸ سیمبول با فرمت C_POINTER قابل استفاده است.
	مرجع	شمارنده و یا سیمبول با داده نوع شمارنده
اشاره‌گر شمارنده سرعت بالا **HC_POINTER**	بیشترین تعداد	در هر تابع بلوکی ۸ سیمبول با فرمت HC_POINTER قابل استفاده است.
	مرجع	شمارنده ۳۲ بیتی و یا سیمبول با داده نوع شمارنده که به یک شمارنده ۳۲ بیتی تخصیص داده شده است.

- مثال برای استفاده از سیمبول با فرمت POINTER

تابع بلوکی FB0 دارای سه پایه ورودی است که در برنامه اصلی D0 و D1 و D2 به آن‌ها اختصاص داده شده است (مقدار اولیه این سه حافظه برابر صفر است)، همچنین نوع داده‌ها و کلاس آن‌ها در کادر قرمز رنگ شکل زیر مشخص شده‌اند. اجرای این برنامه به صورت زیر خواهد بود. (SM400 یک رله در CPUهای سری AH500 می‌باشد که همزمان با وصل شدن تغذیه CPU یک می‌شود، همچنین SM402 تنها در سیکل اول کار PLC یک است).

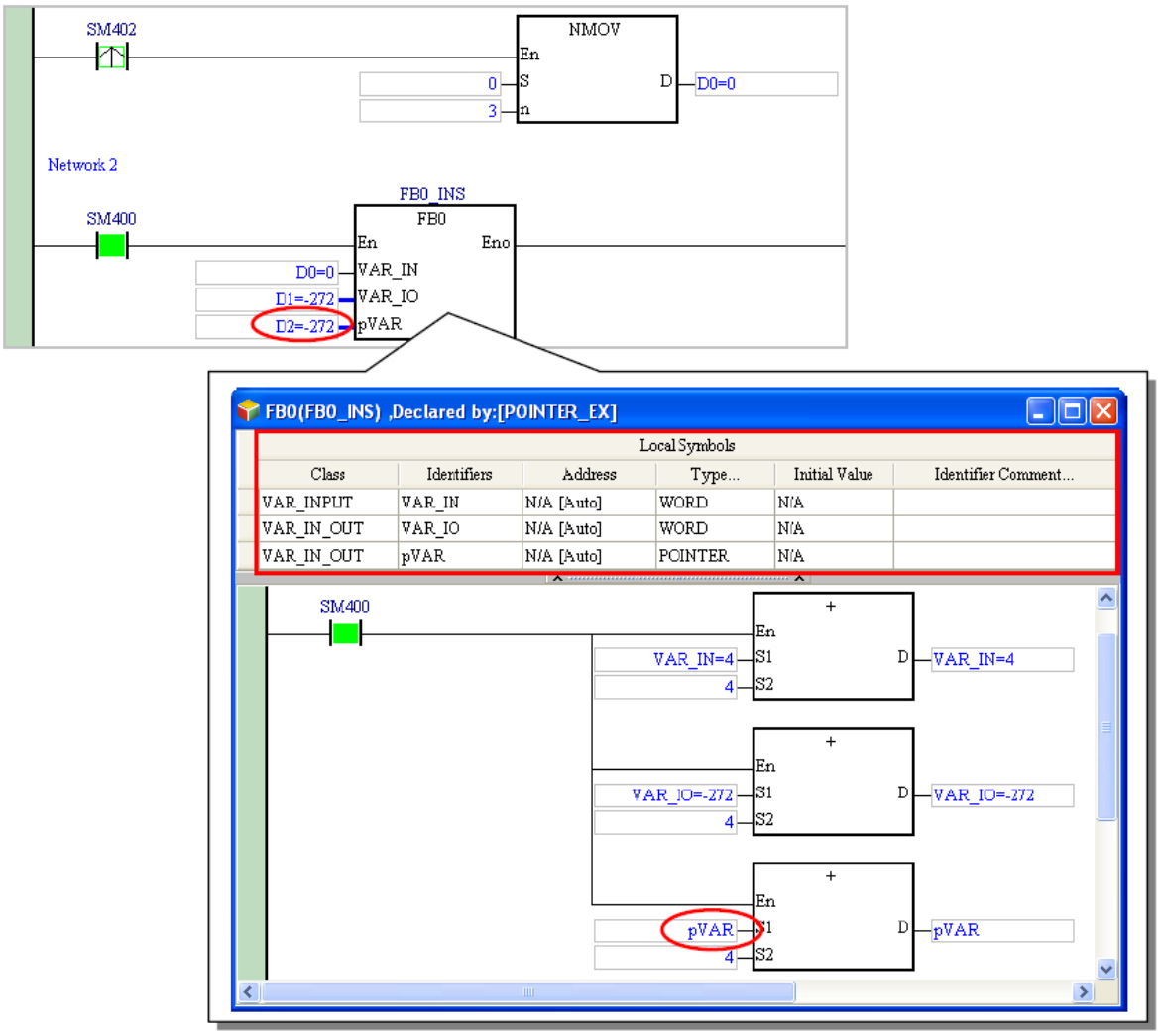
- D0 به پایه VAR_IN اختصاص داده شده است، پایه ای که فرمت آن از نوع کلاس VAR_INPUT و نوع آن WORD است. زمانی که تابع بلوکی فراخوانی می‌شود، مقدار D0 در VAR_IN کپی می‌شود. پس از اجرای برنامه تابع بلوکی، عدد ۴ با مقدار VAR_IN جمع می‌شود. نتیجه جمع تاثیری بر روی D0 ندارد و در نتیجه مقدار آن صفر می‌ماند.

- D1 به پایه VAR_IO اختصاص داده شده است، پایه ای که فرمت آن از نوع کلاس VAR_IN_OUT و نوع آن WORD است. زمانی که تابع بلوکی فراخوانی می‌شود، مقدار D1 در VAR_IO کپی می‌شود. پس از اجرای

برنامه تابع بلوکی، عدد ۴ با مقدار VAR_IO جمع می‌شود. با توجه به آنکه کلاس سیمبول VAR_IO به صورت VAR_IN_OUT است، در نهایت مقدار VAR_IO در D1 کپی می‌شود.

- D2 به پایه pVAR اختصاص داده شده است، پایه ای که فرمت آن از نوع کلاس VAR_IN_OUT و نوع آن POINTER است. بنابراین ۴ بعد از جمع با D2 بلافاصله در حین اجرای تابع بلوکی در D2 کپی می‌شود.

با آنکه نتیجه اجرای D1 و D2 یکسان خواهد بود، اجرای و مقداردهی این دو از دو مسیر متفاوت صورت می‌گیرد. در واقع D1 با واسطه و D2 را از طریق آدرس مستقیم آن مورد استفاده قرار گرفته‌اند. همچنین با توجه به آنکه pVAR اشاره‌گری برای D2 است، مقدار آن در مانیتورینگ آنلاین همانند VAR_IO نمایش داده نمی‌شود.



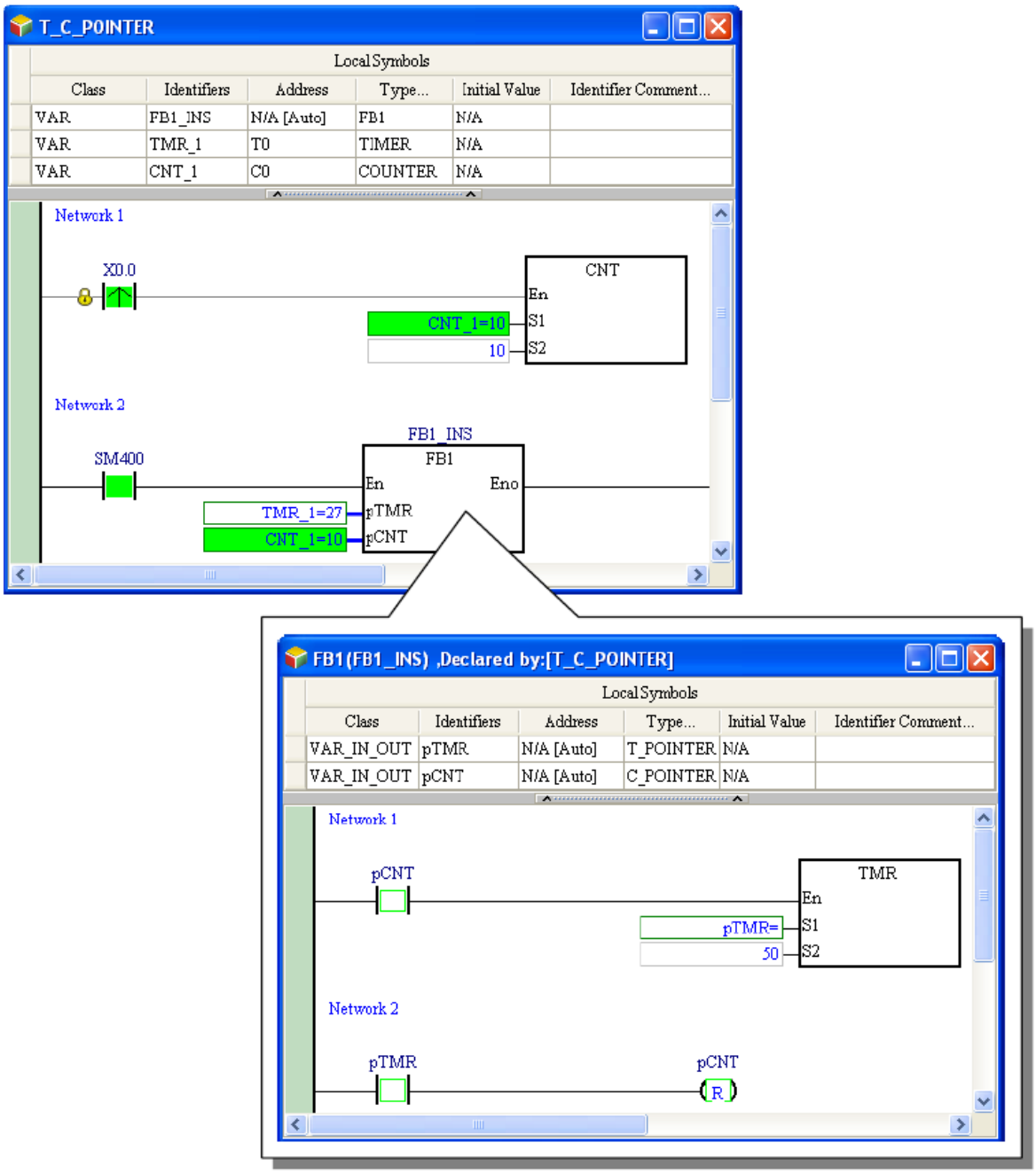
شکل ۷-۱۰ مثالی جهت استفاده از اشاره گر در توابع بلوکی

- مثال برای استفاده از سیمبول با فرمت T_POINTER و C_POINTER

کاربران در بلوک‌ها امکان تعریف سیمبول شمارنده و زمان‌سنج در توابع بلوکی برای PLCهای سری AH500 را ندارند، با این حال با تعریف سیمبول‌هایی با فرمت C_POINTER و T_POINTER و ارسال داده‌های زمان‌سنج و شمارنده از برنامه اصلی به توابع بلوکی می‌توان مقادیر شمارنده و زمان‌سنج را در این توابع نیز مورد استفاده و تغییر قرار داد.

در مثال زیر تابع بلوکی FB1 دارای دو پایه ورودی است. pTMR سیمبولی با نوع T_POINTER و pCNT سیمبولی با نوع C_POINTER است. برنامه اصلی دو ورودی TMR_1 و CNT_1 را به این دو ورودی اختصاص می‌دهد (داده سیمبول TMR_1 زمان-سنج و داده CNT_1 شمارنده است).

در POU به ازای لبه‌های بالارونده X0.0 مقدار CNT_1 افزایش پیدا می‌کند. بیت شمارنده CNT_1 زمانی یک می‌شود که مقدار شمارنده CNT_1 برابر ۱۰ شود. با توجه به آنکه TMR_1 و CNT_1 به پایه‌های ورودی تابع بلوکی وصل هستند، حالت CNT_1 (pMTR) بعد از اجرای تابع بلوکی مشخص خواهد شد. اگر CNT_1 (pCNT) یک شود، TMR_1 (pTMR) شروع به زمان‌سنجی می‌کند. زمانی که مقدار pTMR برابر ۵۰ است، CNT_1 (pCNT) ریست می‌شود. اشاره‌گرهای زمان‌سنج و شمارنده نیز در مانیتورینگ آنلاین مقادیرشان نمایش داده نمی‌شود.



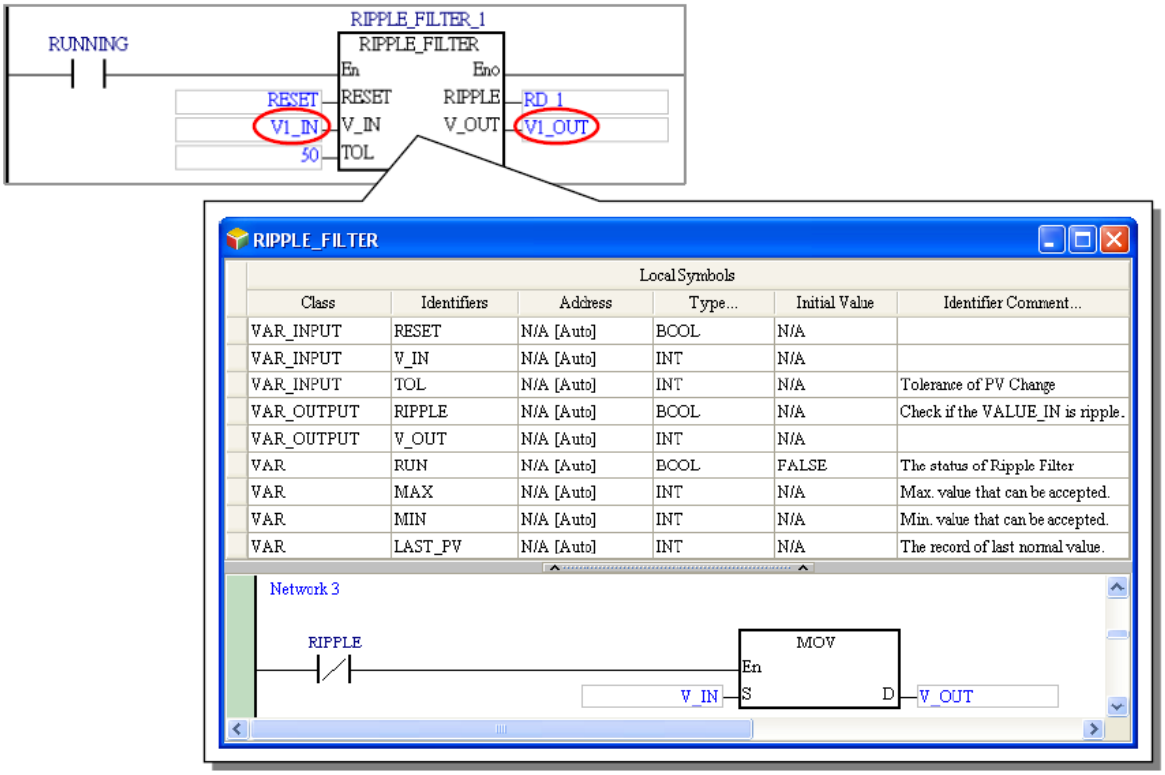
شکل ۷-۱۱ مثالی استفاده از اشاره گرهای زمان سنج و شمارنده در توابع بلوکی

۷-۲-۵- مرجع و نسخه تابع بلوکی

مرجع[۱] تابع بلوکی و نسخه آن مفاهیم مهمی هستند که در ادامه آن‌ها را با ذکر یک مثال مرور خواهیم کرد. در شکل زیر زمانی که برنامه اجرا می‌شود، مقدار V1_IN به تابع بلوکی RIPPLE_FILTER

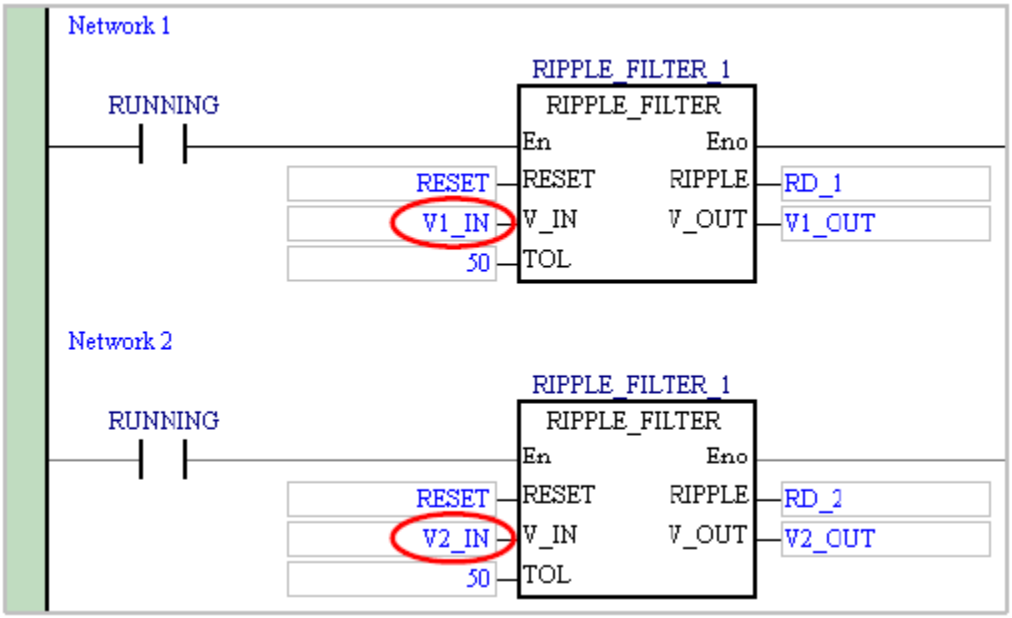
[۱] مرجع و نسخه تابع بلوکی در اینجا به ترتیب ترجمه Function Block Definition و Function Block Instance هستند.

منتقل می‌شود. پس از آنکه این مقدار به تابع بلوکی ارسال می‌شود، با مقادیر آمده به تابع بلوکی در دفعات پیشین مقایسه می‌شود و در نهایت نتیجه به V1_OUT منتقل می‌شود. برای اینکه بتوان نتیجه عملیات و یا حالت‌هایی از اجراهای قبلی را ذخیره کرد (در اینجا سیمبول RIPPLE این کار را می‌کند)، زمانی که برنامه کامپایل می‌شود سیستم به سیمبول‌هایی که نیاز به حافظه دارند حافظه اختصاص می‌دهد.



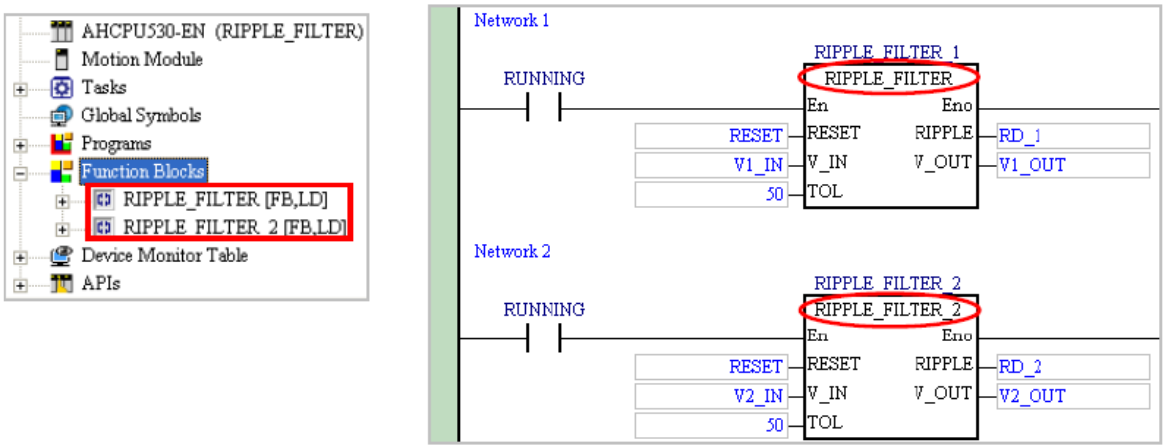
شکل ۷-۱۲ مثالی جهت آشنایی با مرجع و نسخه توابع بلوکی- قسمت اول

زمانی که داده‌های متعددی که مستقل از هم هستند را همزمان بررسی می‌کنیم، ممکن است نتایج اجرای آن‌ها بر روی یکدیگر تاثیر بگذارد و نتایج را نامعتبر کند. به عنوان نمونه در مثال زیر تابع بلوکی RIPPLE_FILTER یکبار با ورودی V1_IN و سپس با ورودی V2_IN اجرا می‌شود، که مرتبه دوم اجرا، تحت تاثیر نتایج اجرای اول قرار می‌گیرد و سیمبول‌های حافظه دار آن تغییر می‌کند.



شکل ۷-۱۳ مثالی جهت آشنایی با مرجع و نسخه توابع بلوکی- قسمت دوم

در صورتی که POU مورد نظر در مثال فوق را دوباره بسازیم، به علت آنکه به آن‌ها هنگام کامپایل حافظه‌های متفاوتی اختصاص داده می‌شود، عملکرد آن‌ها تداخل نخواهد داشت و صحیح خواهد بود.



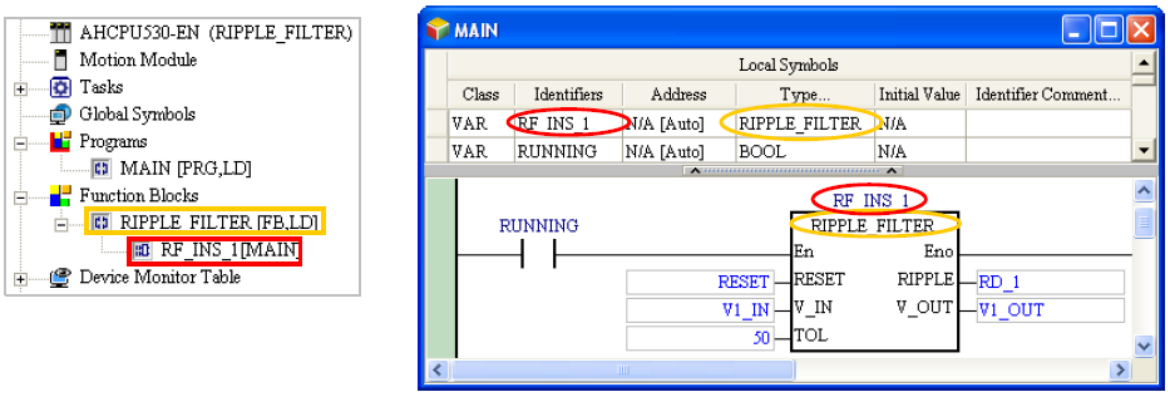
شکل ۷-۱۴ مثالی جهت آشنایی با مرجع و نسخه توابع بلوکی- قسمت سوم

زمانی که کاربر POU تابع بلوکی را اضافه و سیمبول‌های آن را تعیین و برنامه آن را می‌نویسد، در واقع مرجع تابع بلوکی را ساخته است. مرجع تابع بلوکی در واقع همانند فایلی است که عملکردی در هیچ فعالیتی ندارد و مناسب هیچ فعالیتی در PLC نیست و تنها الگوریتم عملکردی تابع بلوکی در آن مشخص شده است.

شکل ۷-۱۵ مرجع توابع بلوکی

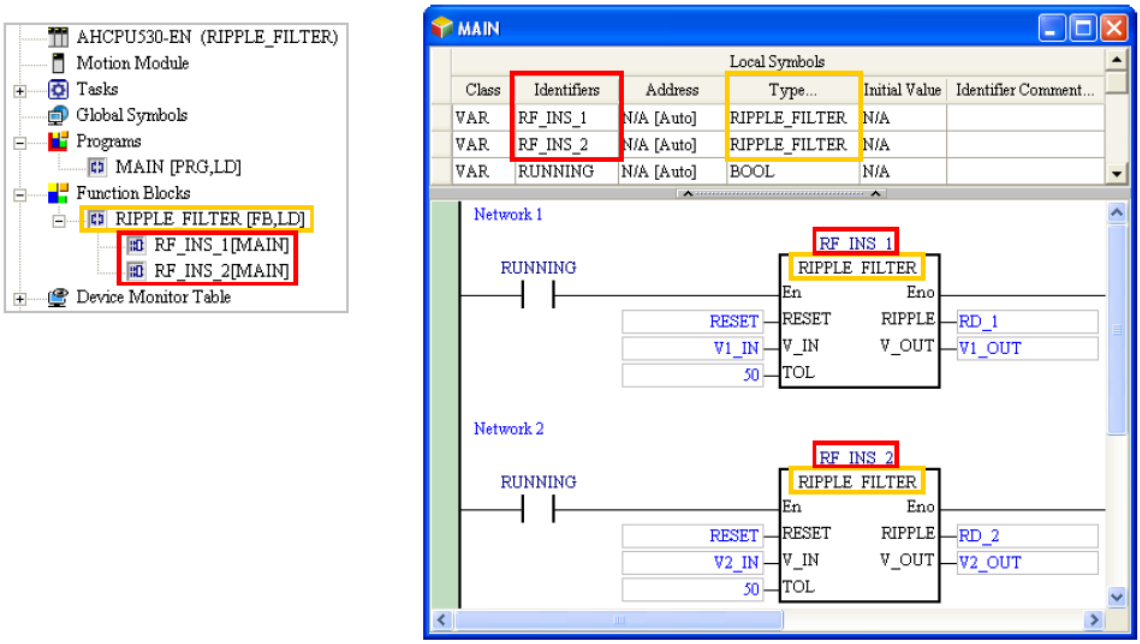
در صورتی که برنامه‌ای این تابع بلوکی را فراخوانی کند، ابتدا کاربر باید سیمبولی را که نوع آن Function Block است را مشخص کند. در واقع کاربر باید شی‌ای را که می‌خواهد طبق مرجع تابع بلوکی با آن کار کند را با این کار مشخص کند، که به این شی، نسخه تابع بلوکی گفته می‌شود.

زمانی که برنامه کامپایل شد ، سیستم به هر "نسخه تابع بلوکی و سیمبول‌های آن، حافظه منحصر به فردی (با توجه به مرجع آن) اختصاص می‌دهد. (در واقع این مفهوم مطابق مفهوم FB در برنامه نویسی محصولات زیمنس است با این تفاوت که دیتا بلاک مرتبط با آن به صورت اتوماتیک و توسط نرم افزار و بدون امکان دسترسی مستقیم اختصاص داده می‌شود.)



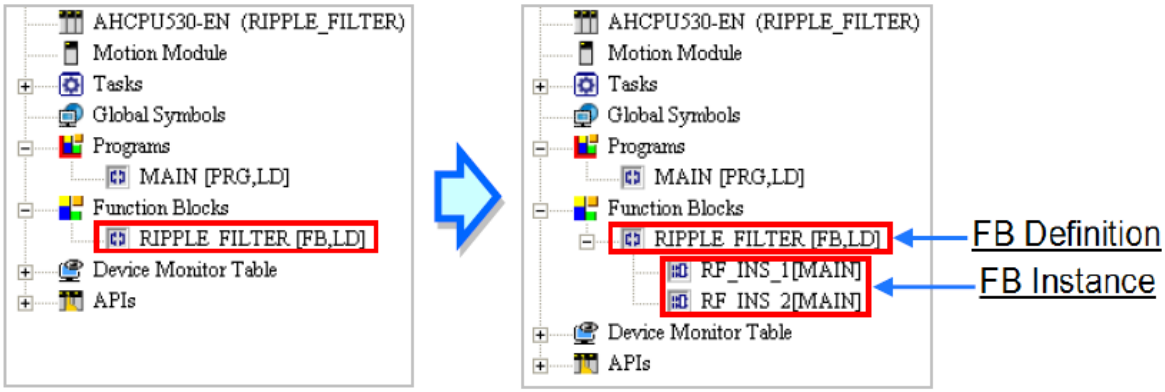
شکل ۷-۱۶ نسخه تابع بلوکی- قسمت اول

در صورتی که نیاز باشد در عملیات مستقل دیگری از تابع بلوکی استفاده شود، با اختصاص سیمبول دیگری به آن، سیستم به صورت اتوماتیک نسخه دیگری برای تابع بلوکی در نظر می‌گیرد، به همین منوال برنامه به آن نسخه تابع بلوکی، حافظه مستقل برای عملیات مستقل اختصاص می‌دهد.



شکل ۷-۱۷ نسخه تابع بلوکی- قسمت دوم

همانطور که در شکل زیر مشخص است، قبل از کامپایل برنامه تنها مرجع تابع بلوکی در بخش مدیریت پروژه در زیر قسمت Function Block لیست شده است. **پس از کامپایل پروژه**، نسخه های هر تابع بلوکی در زیر مرجع آن لیست خواهد شد. فرمت نام این نسخه‌ها تشکیل شده از نام سیمبول و POU-ای که در آن حضور دارد به صورت Instance name[POU name] است.



شکل ۷-۱۸ لیست نسخههای مرجع تابع بلوکی در زیر نام آن

پس از کامپایل برنامه، حافظه‌ها به سیمبول‌های با داده‌های عمومی تخصیص داده می‌شود. همچنین نسخه متناظر با سیمبول اختصاص داده شده به تابع بلوکی (که فرمت آن سیمبول، Function Block است) نیز پس از کامپایل برنامه ایجاد خواهد شد. تعداد مرجع های تابع بلوکی شامل نسخه در CPUهای سری AH500 نمی‌تواند بیشتر از ۱۰۲۴ باشد (توجه شود که مرجع‌های بدون نسخه به حساب نمی‌آیند).

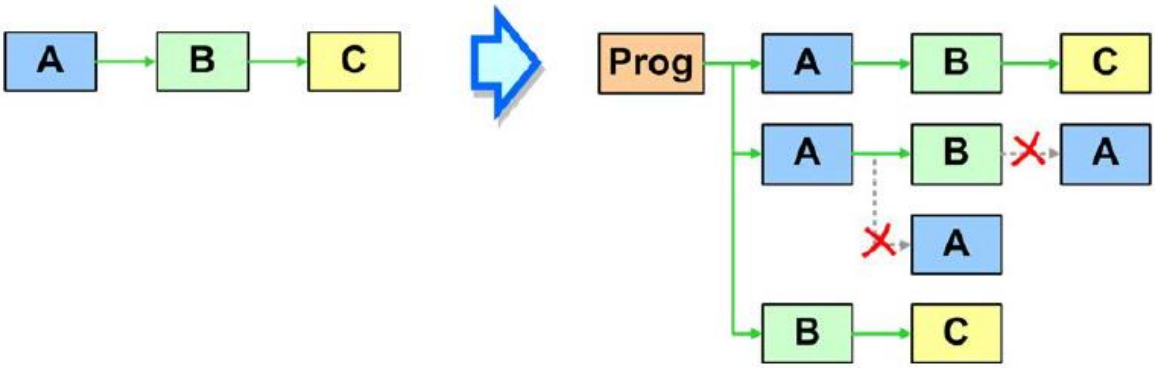
به طور خلاصه می‌توان گفت توابع بلوکی شامل مرجع و نسخه هستند، مرجع در بر گیرنده ساختار و "نسخه تابع بلوکی مرتبط با نحوه اجرای آن و در برگیرنده داده‌های ذخیره شده آن است. از هر مرجع می‌توان در نسخه‌ها مستقلی جهت کاربردهای متفاوتی استفاده کرد.

۷-۲-۶- فراخوانی توابع بلوکی

در ISPSoft، هر تابع بلوکی می‌تواند تابع بلوکی دیگری را فراخوانی کند و به صورت پی در پی می‌توان تا ۳۲ لایه فراخوانی متوالی داشت، در این فراخوانی‌های متوالی، همیشه اولین لایه فراخوانیِ نسخه تابع بلوکی به وسیله برنامه است.

شکل ۷-۱۹ فراخوانی های متوالی توابع بلوکی

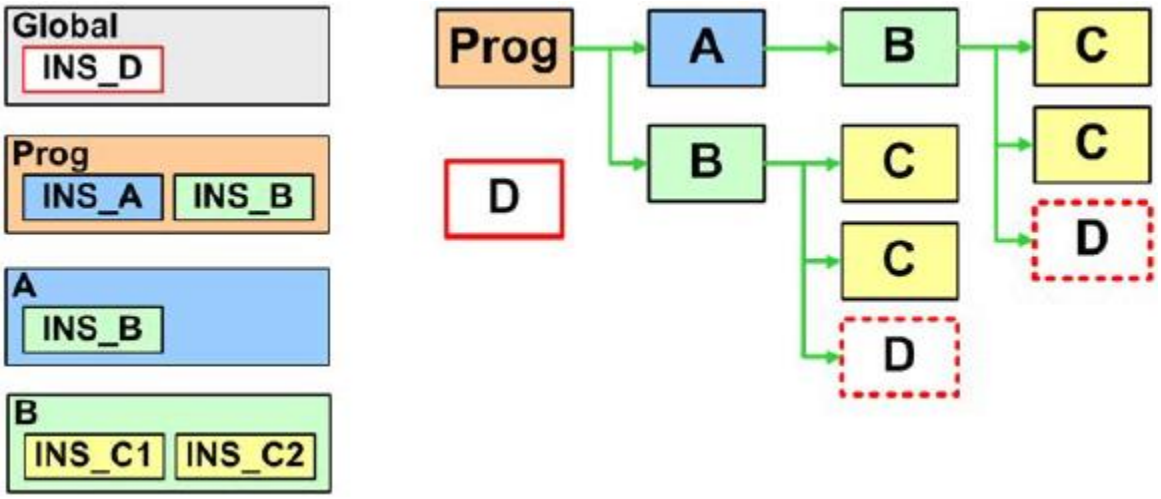
فرض کنیم تابع بلوکی A، تابع بلوکی B را فراخوانی کند، و در تابع بلوکی B نیز تابع بلوکی C فراخوانی شود. در نتیجه تابع بلوکی A در فراخوانی نسبت به B و به تبع آن C مقدم است و اولویت بالاتری دارد، همچنین تابع بلوکی B که چون تابع بلوکی C در آن فراخوانی شده است نسبت به آن اولویت دارد. در ISPSoft، توابع بلوکی اجازه فراخوانی **خود** و **توابع بلوکی با اولویت بالاتر از خود** را ندارند (چرا که در این صورت یک حلقه بی نهایت از فراخوانی‌ها اتفاق خواهد افتاد). POU برنامه امکان فراخوانی تمامی توابع بلوکی را دارد.



شکل ۷-۲۰ توابع بلوکی به صورت سلسله مراتبی امکان فراخوانی خود را در سطوح پایین تر ندارند

در صورتی که توابع بلوکی A و B و C به وسیله هیچ POU برنامه‌ای فراخوانی نشوند، نسخه‌های آن‌ها در زمان کامپایل ساخته نمی‌شود و در اجرای نقشی نخواهند داشت. پس از آنکه تابع بلوکی A در POU برنامه‌ای یا در جدول سیمبول‌های سراسری مورد استفاده قرار گرفت، نسخه‌های متناظر با آن (برای هر

سه تابع بلوکی)، حین کامپایل برنامه ساخته خواهد شد. در مثال زیر تابع بلوکی D در جدول سیمبول-های سراسری تعریف شده ولی بقیه نسخههای تابعهای بلوکی به صورت محلی در جدول سیمبول برنامه با اولویت بالاتر (فراخواننده) آمدهاند.



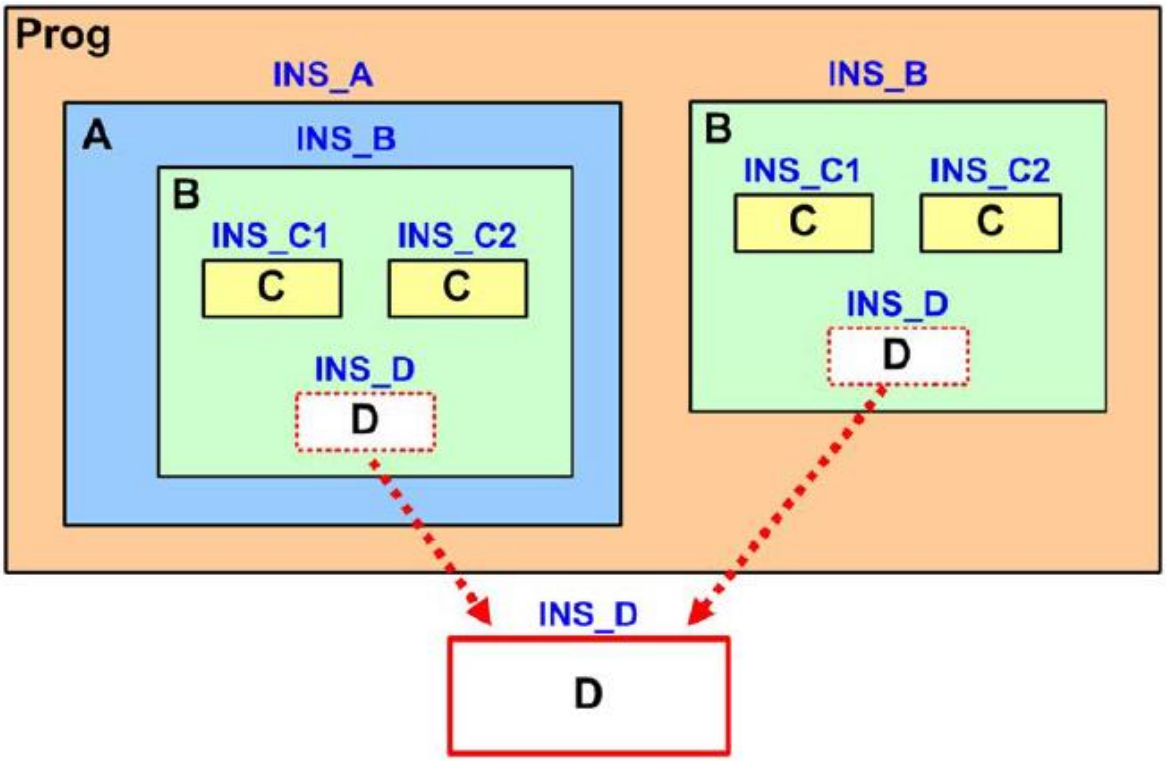
شکل ۷-۲۱ مثالی از تعریف نسخه های توابع بلوکی در جدول سیمبول های محلی و سراسری

جدول ۷-۶: مثالی از تعریف نسخه های توابع بلوکی در جدول سیمبول های محلی و سراسری

محل معرفی		تابع بلوکی معرفی شده
جدول سیمبول سراسری		یک نسخه از تابع بلوکی INS_D : D
جدول سیمبول محلی	POU برنامه	یک نسخه از تابع بلوکی INS_A : A یک نسخه از تابع بلوکی INS_B : B
	تابع بلوکی A	یک نسخه از تابع بلوکی INS_B : B
	تابع بلوکی B	دو نسخه از تابع بلوکی INS_C1 : C و INS_C2

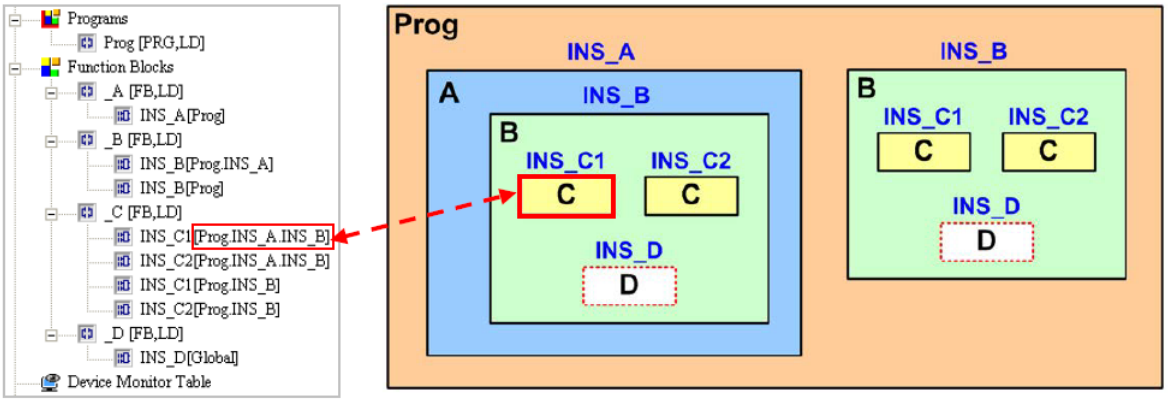
نسخههای تابع بلوکی آمده در جدول فوق پس از کامپایل ساخته خواهند شد. با توجه به آنکه دو نسخه از تابع بلوکی C در جدول سیمبولهای محلی تابع بلوکی B تعریف و در نتیجه فراخوانی شده است، زمانی که نسخه جدیدی از تابع بلوکی B ایجاد میشود، دو نسخه جدید از تابع بلوکی C نیز ایجاد خواهد شد. همچنین چون سیمبول تابع بلوکی D در جدول سیمبول سراسری آمده است، تابع بلوکی آن و دادههای متناظر با آن در تمامی POUها قابل استفاده است، و این یعنی سیستم تنها یک نسخه از تابع

بلوکی D می‌سازد و یک حافظه داده برای سیمبول‌های آن در نظر می‌گیرد. در مثال فوق نیز تابع بلوکی D دوبار به وسیله دو POU متفاوت فراخوانی شده و نسخه مشترکی از آن در هر فراخوانی اجرا شده است.



شکل ۷-۲۲ استفاده از نسخه مشترک در دو تابع بلوکی

آنچه پس از کامپایل برنامه در بخش مدیریت پروژه آمده است را می‌توان در شکل زیر دید. در شکل زیر مشخص است که نسخهها با اولویت پایینتر در ادامه نام خود، در قسمت درون براکت، نام توابع بلوکی با اولویت بالاتر خود را هم دارند.



شکل ۷-۲۳ نسخه‌ها در بخش مدیریت برنامه، نام توابع بلوکی با اولویت بالاتر خود را دارند

در PLCهای سری DVP کاربران نمی‌توانند سیمبول توابع بلوکی را در جداول سیمبول محلی تعریف کنند. در صورتی که بخواهیم از تابع بلوکی در تابع بلوکی دیگری استفاده کنیم، باید سیمبول و در نتیجه نسخه آن را در جدول سیمبول سراسری تعریف کنیم. به همین علت حین برنامه نویسی برای PLCهای سری DVP باید بیش از پیش دقت کرد تا حافظه مشترک نسخه در زمانی که از تابع بلوکی استفاده می-کنیم، مشکل ساز نشود. در مواقعی که نیاز به عملکرد مستقل در تابع بلوکی داریم، می بایست، سیمبول و در نتیجه نسخه‌ای مستقل در جدول سیمبول‌های سراسری برای آن در نظر بگیریم.

محدودیت فوق در مورد سری AH500 وجود ندارد و کاربران می‌توانند به صورت محلی نیز برای توابع بلوکی سیمبول و در نتیجه نسخه تعریف کنند. در سری AH500 بلوک‌های حافظه‌ای که به نسخه توابع بلوکی اختصاص داده می‌شود، بستگی به موقعیت آن توابع دارد که در فصل بعدی آن را مرور خواهیم کرد.

۷-۲-۷- اختصاص بلوک های حافظه به توابع بلوکی

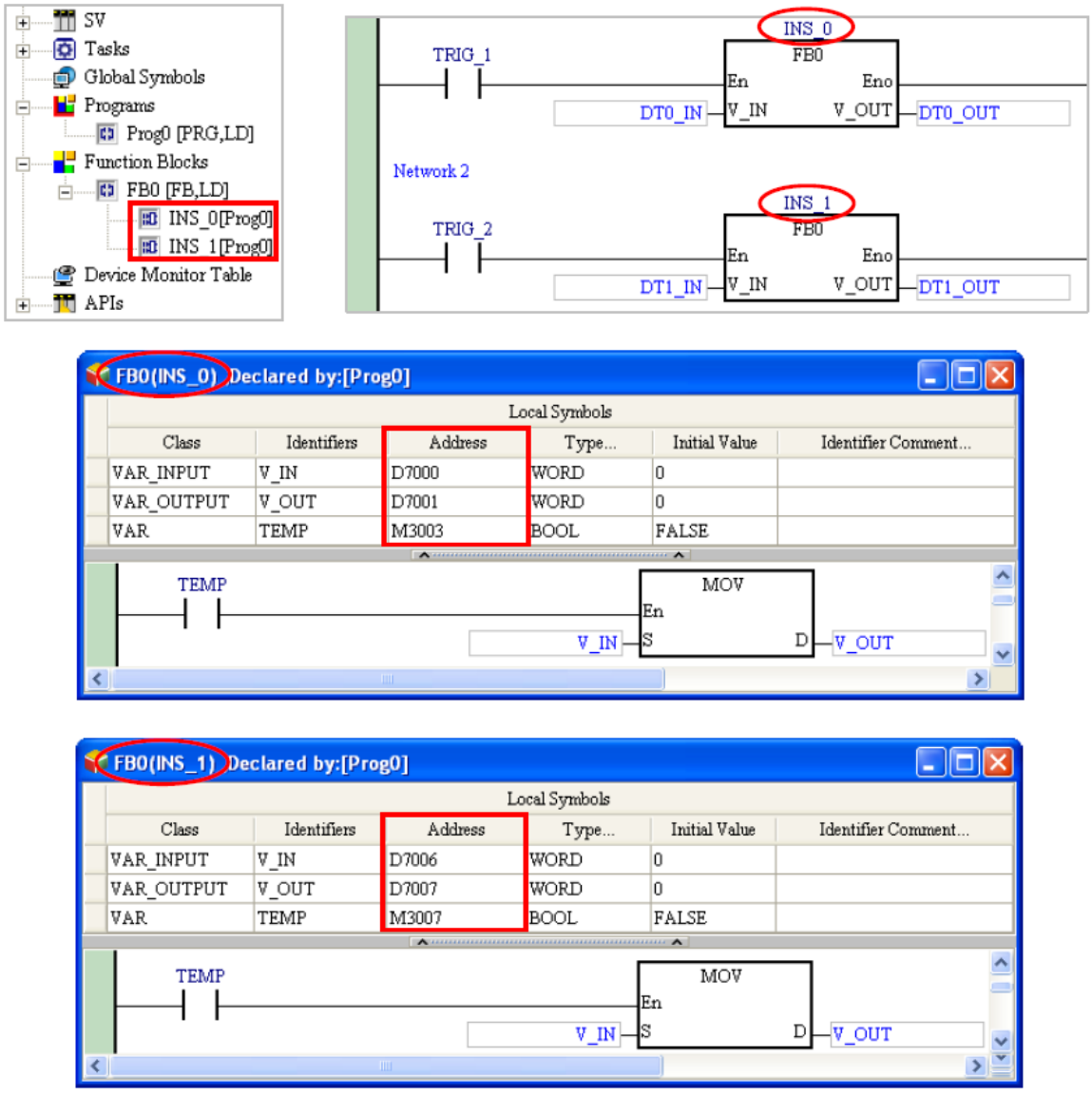
در صورتی که کاربران آدرس دهی سیمبول‌ها را در تابع بلوکی انجام ندهند، سیستم به صورت اتوماتیک (بلوکی از) حافظه‌های استفاده نشده در برنامه را حین کامپایل برنامه و ساخت نسخه برای توابع بلوکی به جدول سیمبول‌های محلی اختصاص می‌دهد. اما زمانی که کاربر حافظه مشخصی را به سیمبول‌ها تخصیص می‌دهد، دیگر هنگام کامپایل، سیستم به تابع بلوکی، "حافظه بلوکی" اختصاص نمی‌دهد، در این گونه توابع بلوکی، حتی در نسخه های دیگر آن نیز باز همان حافظه‌ها در عملیات درگیر خواهند بود.

PLC های سری DVP و AH500 دارای مکانیزم متفاوتی برای تخصیص حافظه هستند که در ادامه مرور خواهیم کرد.

- PLCهای سری DVP

زمانی که در PLCهای سری DVP بلوک‌های حافظه تخصیص داده می‌شود، تمامی نسخه‌های تابع بلوکی به حافظه‌های نوع P تخصیص داده می‌شوند. در صورتی که نام POU تابع بلوکی با "_P0" و یا "_P1" شروع شده باشد، P0 و P1 به نسخههای تابع بلوکی بعد از کامپایل برنامه اختصاص داده می‌شود. در صورتی که بیشتر از یک نسخه تابع بلوکی در نظر گرفته شود، به علت استفاده مجدد از حافظه P مشابه هنگام کامپایل خطا گرفته می‌شود.

سیستم با توجه به نوع داده سیمبول‌های محلی در تابع بلوکی، حافظه مورد نظر را به آن اختصاص می‌دهد. به عنوان مثال سیستم به سیمبول محلی WORD رجیستر داده و به سیمبول محلی Bool رله‌ی بیتی اختصاص می‌دهد. بنابراین اندازه نسخه تابع بلوکی در سری‌های DVP به سیمبول‌های محلی آن وابسته است و هر نسخه به حافظه‌های مختلفی تخصیص داده می‌شود. در شکل زیر دو نسخه از تابع بلوکی را می‌بینید، بعد از آنکه پنجره هرکدام از نسخهها را باز می‌کنیم، آدرس‌های متفاوت تخصیص داده شده به سیمبول‌ها را در دو نسخه می‌بینیم.

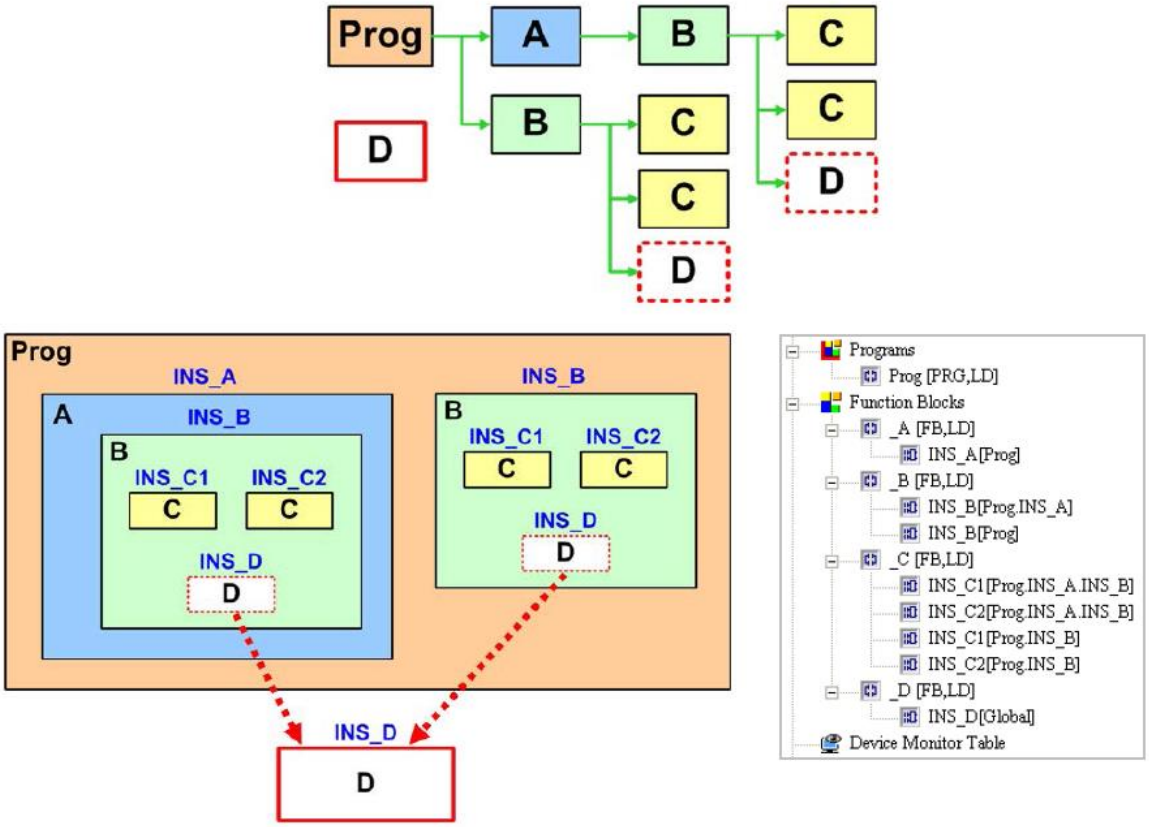


شکل ۷-۲۴ اختصاص بلوک های حافظه به توابع بلوکی

- PLCهای سری AH500

در PLCهای سری AH500 پردازنده تعداد ثابتی از بلوک‌های حافظه را برای نسخه‌های تابع بلوکی به صورت رزرو در نظر گرفته است. این تعداد ثابت از بلوک‌های حافظه با توجه به نوع PLC متفاوت است. اندازه هر بلوک حافظه برابر 4096 حافظهWORD است. بلوک‌های حافظه رزرو توسط پردازنده PLC امکان دسترسی مستقیم به وسیله کاربر را ندارند و به همین علت کاربر نمی‌تواند در پنجره تابع بلوکی نتایج اعمال شده بر روی آن‌ها را ببیند.

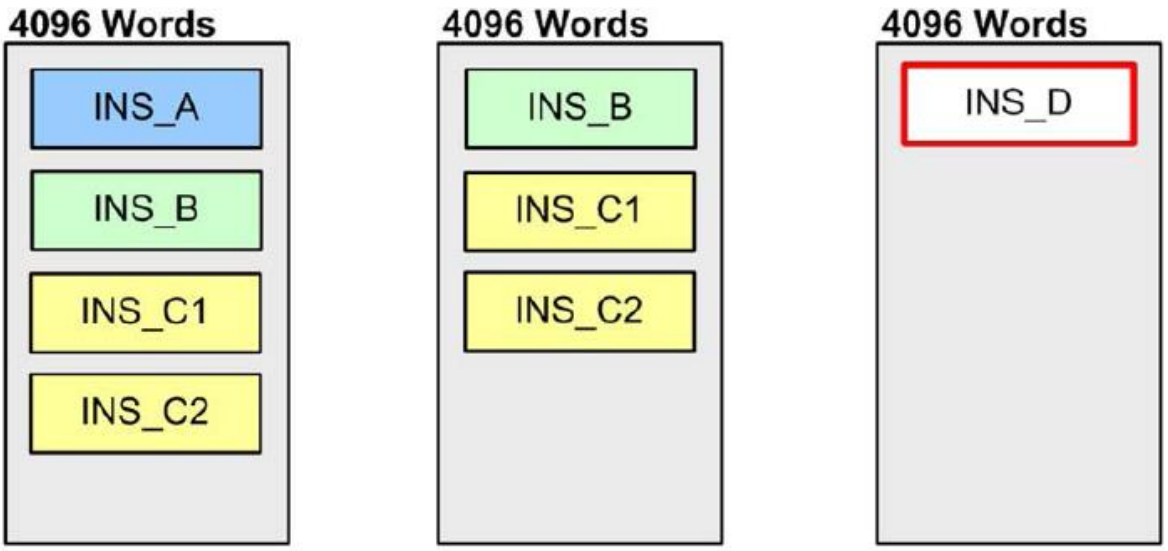
مثال بخش ۷-۲-۶- را در نظر بگیرید، می‌خواهیم ارتباط بین توابع بلوکی را در آن بررسی کنیم.



شکل ۷-۲۵ مثالی جهت بررسی روابط بین توابع بلوکی

سیستم بلوک های حافظه 4096 خانه ای (با خانه‌هایی به فرمت WORD) را به نسخه-های توابع بلوکی فراخوانی شده در POU برنامه پروژه PLCهای AH500 اختصاص می-دهد. به تابع بلوکی فراخوانی شده توسط POU برنامه و توابع بلوکی درون آن همگی

یک بلوک حافظه تخصیص داده می‌شود.همچنین به هر تابع بلوکی در جدول سیمبول-های سراسری نیز یک بلوک حافظه اختصاص داده می‌شود.



شکل ۷-۲۶ تخصیص حافظه به توابع بلوکی

بلوک‌های حافظه تخصیص داده شده به توابع بلوکی در PLCهای سری AH500 در دسترس کاربران نیستند. همچنین اندازه آن‌ها بدون توجه به برنامه تابع بلوکی ثابت و برابر 4096 حافظه WORD است، **به همین علت کاربر باید نسبت به حافظه مورد نیاز تابع بلوکی مورد استفاده دقت لازم را به خرج دهد تا این محدودیت رد نشود**. محاسبه اندازه حافظه نسخه تابع بلوکی به صورت زیر است:

۱- اندازه حافظه نسخه تابع بلوکی برابر با ۲ حافظه WORD است.

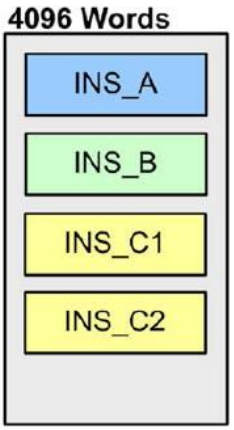
۲- سیمبول‌های مورد استفاده مطابق با فرمت خود حافظه بر حسب WORD نیاز دارند، حافظه‌های کوچکتر از WORD نیز برابر یک WORD محاسبه می‌شوند.

۳- سیمبول نوع تابع بلوکی که در تابع بلوکی (شماره ۱) فراخوانی می‌شود نیز برابر با ۲ حافظه WORD است.

بدین ترتیب برای مثال گذشته داریم:

جدول ۷-۷: بررسی سیمبول‌های محلی ذخیره شده درتابع بلوکی (مثال)

مرجع تابع بلوکی	تعداد سیمبول‌های محلی
تابع بلوکی A	دو سیمبول با فرمت DWORDT، یک سیمبول با فرمت LREAL، یک نسخه تابع بلوکی.
تابع بلوکی B	یک سیمبول با فرمت WORD، دو سیمبول با فرمت REAL، دو نسخه تابع بلوکی.
تابع بلوکی C	دو سیمبول با فرمت BOOL، یکی سیمبول آرایه بولی با اندازه ۲۲



محاسبات تعداد حافظه مورد استفاده در بلوک حافظه مورد نظر به صورت زیر است:

۱- اندازه حافظه تابع بلوکی برابر ۲ حافظه WORD است. در اینجا ۴ نسخه تابع بلوکی به کار رفته است که در نتیجه ۸ حافظه WORD را اشغال می کنند.

۲- اندازه حافظه‌های سیمبول‌های محلی (که نوع آن تابع بلوکی نیست) در زیر محاسبه شده است:

- نسخه تابع بلوکی INS_A: دو حافظه DWORD (برابر ۲*۲=۴ حافظه WORD) و یک حافظه LREAL (برابر ۴ حافظه WORD)، در مجموع ۴+۴=۸ حافظه WORD
- نسخه تابع بلوکی INS_B: یک حافظه WORD و دو حافظه REAL (برابر ۲*۲=۴ حافظه WORD)، در مجموع ۱+۴=۵ حافظه WORD
- نسخه توابع بلوکی INS_C1 و INS_C2: دو حافظه BOOL (برابر ۱*۲=۲ حافظه WORD چرا که هر کدام از حافظه‌های کوچکتر از WORD یک WORD محاسبه می‌شوند) و یک حافظه آرایه ۲۲ بیتی (برابر ۲ حافظه WORD چرا که ۲۲ بیت

کمتر از دو و بیشتر از یک WORD هست، دو WORD به حساب می‌آید)، در مجموع ۲+۲=۴ حافظه WORD بر هرکدام از توابع بلوکی INS_C1 و INS_C2 و در مجموع ۴+۴=۸ حافظه WORD برای هر دو بلوک.

در نهایت اندازه حافظه اشغال شده توسط این تابع بلوکی در حافظه بلوکی برابر ۸+۸+۵+۸=۲۹ حافظه WORD خواهد بود.

۷-۳- به کارگیری توابع بلوکی

۷-۳-۱- ویژگی های توابع بلوکی

ویژگی‌های توابع بلوکی در نرم افزارISPSoft در جدول زیر آمده‌اند. برای جلوگیری از خطا حین کامپایل نیاز است کاربر مطابق با این جدول از توابع بلوکی استفاده کند.

جدول ۷-۸: ویژگی های توابع بلوکی

تعداد مرجع تابع بلوکی	AH500: تعداد مرجع (دارای نسخه) حداکثر ۱۰۲۴ است. DVP: تعداد مرجع تابع بلوکی به تعداد حافظه نوع P وابسته است. (هر نسخه تابع بلوکی یک حافظه P را اشغال می‌کند.)
تعداد نسخه تابع بلوکی	AH500: به مدل PLC وابسته است. DVP: تعداد نسخه تابع بلوکی به تعداد حافظه نوع P وابسته است. (هر نسخه تابع بلوکی یک حافظه P را اشغال می‌کند.)
اندازه تابع بلوکی	AH500: اندازه حافظه بلوکی برابر ۴۰۹۴ واحد WORD است. DVP: اندازه تابع بلوکی به حافظه‌های تخصیص داده شده به آن بستگی دارد.
تعداد فراخوانی های متوالی توابع بلوکی	حداکثر ۳۲ لایه تابع بلوکی به صورت متوالی قابل فراخوانی است.
• پایه‌های ورودی توابع بلوکی باید به حافظه‌ها و یا سیمبول‌های آن‌ها متصل شود، همچنین به پایه‌های ورودی تابع بلوکی، می‌توان مقدار ثابت نیز اختصاص داد.	

- توابع بلوکی امکان فراخوانی خود و یا توابع بلوکی قبل از خود (که توسط آن فراخوانی شده‌اند) را ندارند.
- در صورت بازیابی مجدد (Export) تابع بلوکی فقط مرجع آن بازیابی شده و سیمبول-های سراسری آن و مرجع توابع بلوکی فراخوانی شده در آن، بازیابی نخواهند شد.
- پایه En تابع بلوکی در زبان برنامه نویسی Ladder، SFC، FBD و IS باید به کانتکتی متصل باشد اما در زبان ST نیازی به این کار نیست.
- با استفاده از زبان‌های برنامه نویسی Ladder، ST، FBD و IS (به جز SFC) می‌توان کُد توابع بلوکی را نوشت.
- امکان تغییر مرجع تابع بلوکی وجود دارد ولی تغییر نسخه آن ممکن نیست.
- در صورت استفاده از JUMP در تابع بلوکی، مقصد آن باید آن نیز در همان تابع بلوکی باشد.
- در زمان اصلاح آنلاین برنامه در صورتی که ساختار تابع بلوکی در POUهای قبلی آن عوض شود (مثلا تعداد پایه‌های آن عوض شود) و موجب مشکل در نمایش آن‌ها شود، ابتدا باید از حالت اصلاح آنلاین برنامه خارج شده و سپس بلوک قدیمی را حذف و بلوک جدید را جایگزین آن کرد، برنامه تغییر یافته را کامپایل و سپس دوباره دانلود کرد.
- در PLCهای سری AH500 نمی‌توان از سیمبول‌های با فرمت زمان سنج و شمارنده در توابع بلوکی استفاده کرد، ولی به جای آن‌ها می‌توان از سیمبول‌هایی با فرمت POINTER، T_POINTER، C_POINTER و HC_POINTER استفاده کرد.
- دستورات پالسی همچون MOVP و CML نمی‌توانند در تابع بلوکی برای PLCهای AH500 به کار روند. همچنین دستورات LDP/ANDP/ORP، LDF/ANDF/ORF، MCR، PLS, PLF, NP, PN, MC و GOEND نیز در توابع بلوکی این سری از PLCها قابل استفاده نمی‌باشند.

۷-۳-۲- دستورات پالسی در توابع بلوکی PLC های AH500

دستورات پالسی همچون LDP، LDF، NP و PN نمی‌توانند در تابع بلوکی برای PLCهای AH500 به کار روند. جایگزین این دستورات در توابع بلوکی دستورات زیر هستند.

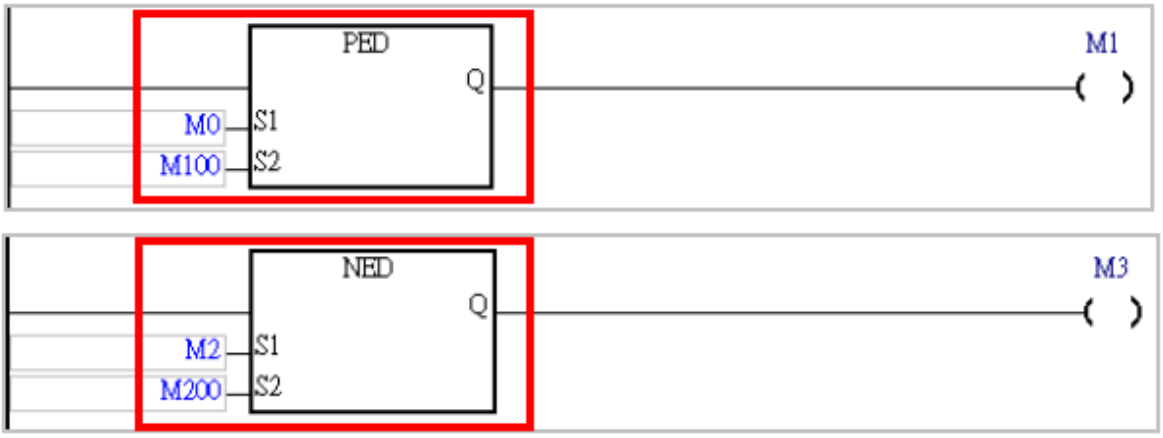
- PED و NED

PED و NED مشابه دستورات LDP و LDF هستند با این تفاوت که امکان استفاده در توابع بلوکی را نیز دارند. همچنین باید به این دو دستور حافظه‌ای بیتی نیز اختصاص داد. حافظه‌های اختصاص داده شده به PED و NED را در هیچ کجای دیگر پروژه نمی-توان استفاده کرد در غیر این صورت با خطا مواجه می‌شویم.

LDP (تشخیص دهنده لبه بالا رونده) و LDF (تشخیص دهنده لبه پایین رونده) در POU برنامه هستند.

شکل ۷-۲۷ دستورات پالسی PED و NED

به جای این دو می‌توان از PED (به جای LDP) و NED (به جای LDF) در توابع بلوکی استفاده کرد. در مثال‌های زیر به ترتیب به این دو حافظه M100 و M200 تخصیص داده شده است.

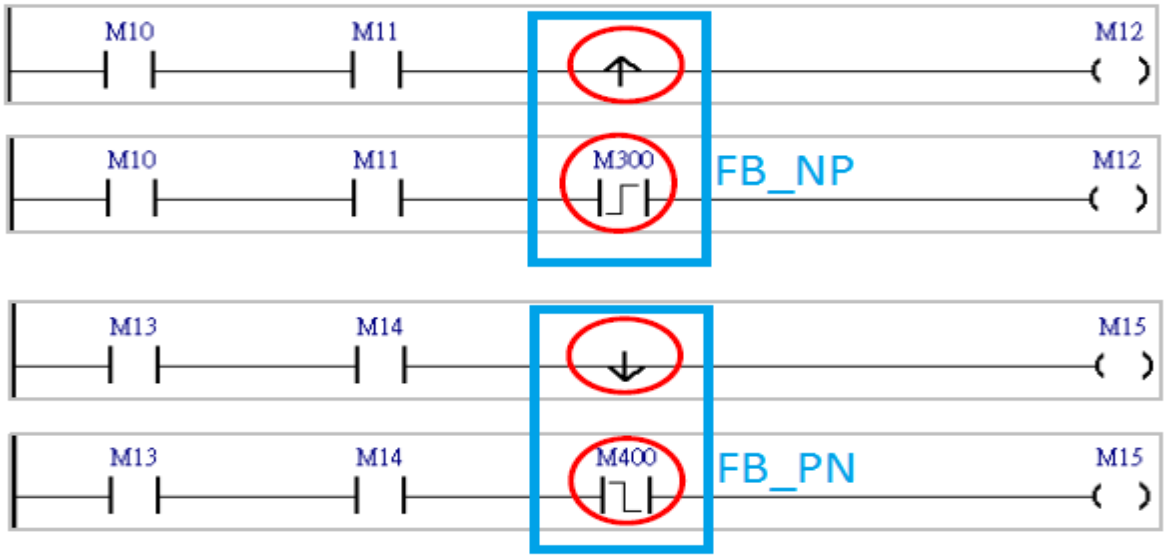


شکل ۷-۲۸ بلوک های دستورهای پالسی

- FB_NP و FB_PN

FB_NP و FB_PN مشابه دستورات NP و PN هستند با این تفاوت که امکان استفاده در توابع بلوکی را نیز دارند. همچنین باید به این دو دستور حافظه‌ای بیتی نیز اختصاص داد. حافظه‌های اختصاص داده شده به FB_NP و FB_PN را در هیچ کجای دیگر پروژه نمی‌توان استفاده کرد در غیر این صورت با خطا مواجه می‌شویم.

NP (تشخیص دهنده لبه بالا رونده) و PN (تشخیص دهنده لبه پایین رونده) در POU برنامه هستند. به جای این دو می‌توان از FB_NP (به جای NP) و FB_PN (به جای PN) در توابع بلوکی استفاده کرد.

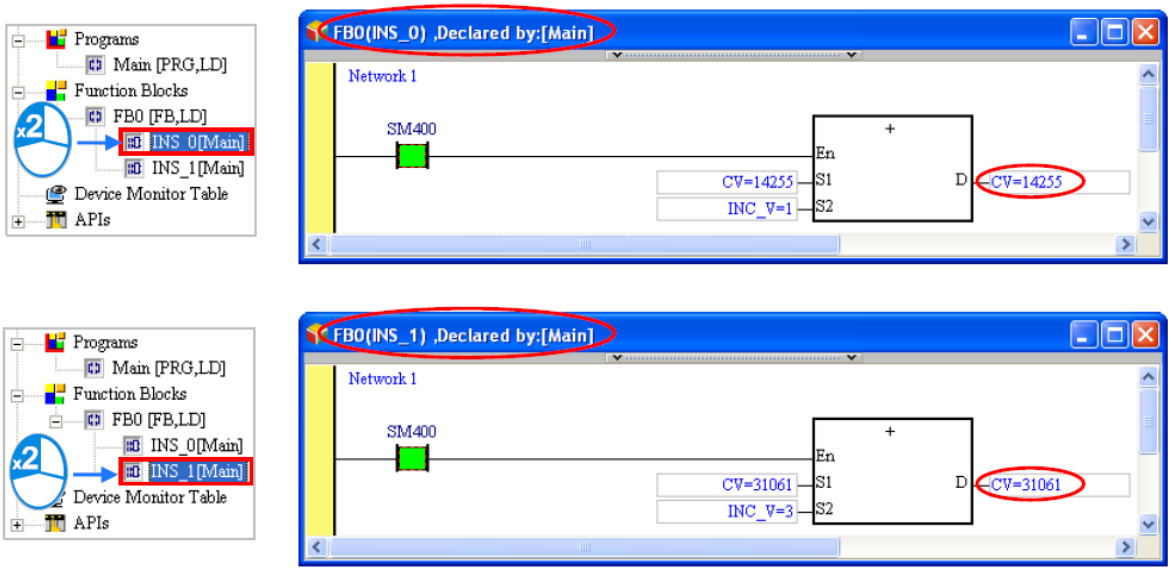


شکل ۷-۲۹ دستورات پالسی FB_PN و FB_NP

۷-۳-۳- مانیتورینگ آنلاین برنامه های نوشته شده در تابع بلوکی

در صورتی که نسخه تابع بلوکی در اجرای برنامه به کار گرفته شود، در حالت آنلاین لازم است برای مانیتور کردن آن، پنجره نسخه مورد نظر را باز کرد.

شکل ۷-۳۰ مانیتورینگ توابع بلوکی – قسمت اول



شکل ۷-۳۱ مانیتورینگ توابع بلوکی – قسمت دوم

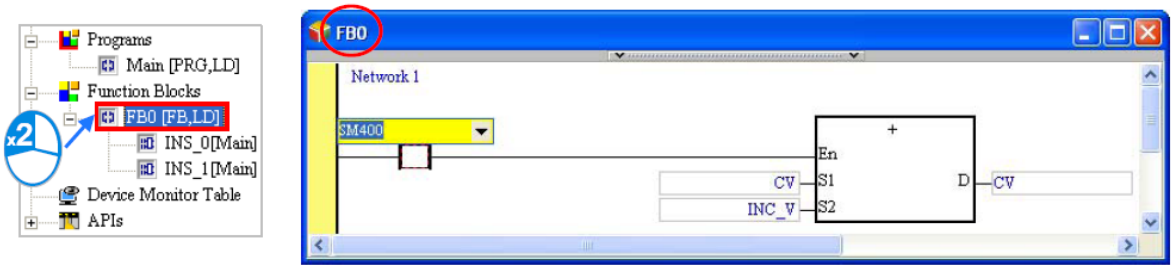
در صورتی که پنجره مرجع تابع بلوکی در حالت مانیتورینگ آنلاین باز شود، مقداری نمایش داده نخواهد شد.

شکل ۷-۳۲ در مانیتورینگ مرجع توابع بلوکی مقداری نمایش داده نخواهد شد

۷-۳-۴- تغییر برنامه تابع بلوکی

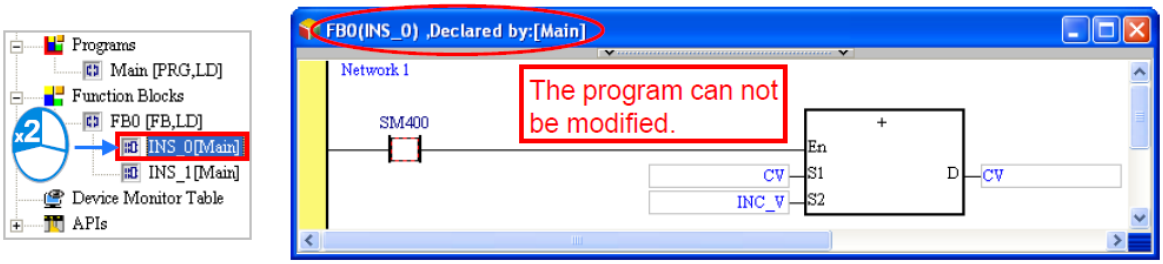
از طریق مرجع توابع بلوکی می‌توان نسخه آن‌ها را ساخت، نسخه توابع بلوکی را نمی-توان ویرایش کرد اما برای مرجع آن‌ها امکان ویرایش وجود دارد. در صورتی که مرجع تابع بلوکی ویرایش یابد، تغییرات بر روی نسخه‌های جدیدی که از مرجع تابع بلوکی ساخته می‌شود اعمال می‌شود. نسخههای جدید بعد از کامپایل دوباره برنامه ساخته خواهند شد.

در صورتی که نیاز به ویرایش آنلاین تابع بلوکی باشد، باید در وضعیت آنلاین مرجع تابع بلوکی را باز و ویرایش کرد.



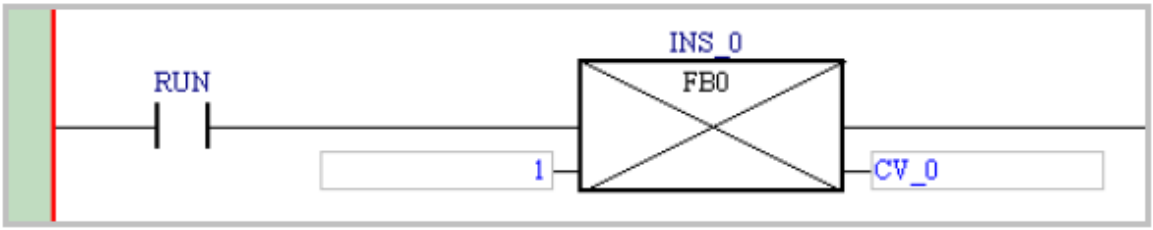
شکل ۷-۳۳ ویرایش آنلاین تابع بلوکی

در صورتی که نسخه تابع بلوکی باز شود، امکان ویرایش آن وجود نخواهد داشت.



شکل ۷-۳۴ امکان ویرایش آنلاین نسخه تابع ورودی وجود ندارد

در صورت که تابع بلوکی در POU دیگری فراخوانی شده باشد، در صورت ویرایش نمایش آن به صورت غیرفعال در آمده و نیاز است تا کاربر با حذف آن، دوباره نسبت به فراخوانی آن در POU مورد نظر اقدام کند.



شکل ۷-۳۵ بعد از ویرایش تابع بلوکی باید آن را در برنامه حذف و دوباره اضافه کنیم

۷-۴- مثال

در این قسمت تکمیل مثالی را که در سه فصل گذشته آورده شده بود را با استفاده از مفهوم توابع بلوکی پی خواهیم گرفت. POUهای ساخته شده در بخش قبل در جدول زیر شرح داده شده‌اند.

جدول ۷-۹: مرور POUهای برنامه ساخته شده در فصل قبل

نام POU	عملکرد

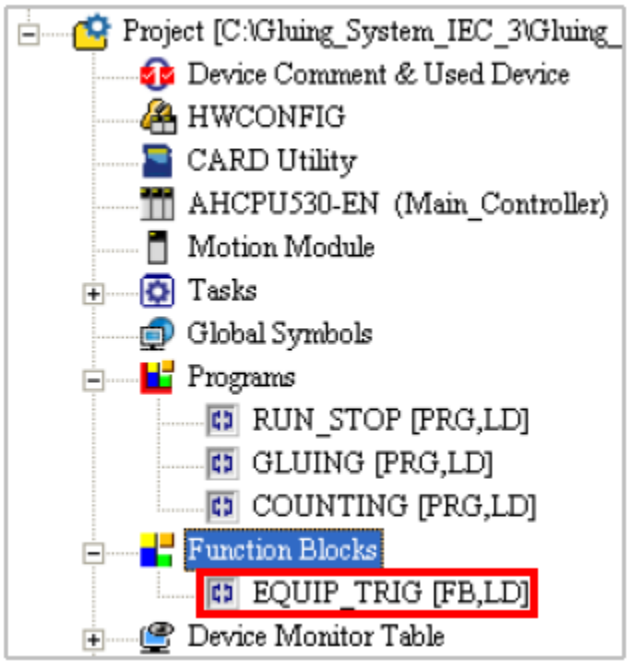
RUN_STOP	در صورتی که کلید START فشرده شود، تجهیزات شروع به کار می‌کنند. در صورتی که STOP فشرده شود و یا خطایی تشخیص داده شود، تجهیزات متوقف می‌شوند.
GLUING	دو دستگاه تزریق در مسیر تسمه نقاله وجود دارد که چسب را به یک شکل تزریق می‌کنند. زمانی که سنسور موقعیت شماره ۱ فعال می‌شود، سیگنال تحریک تزریق کننده شماره ۱ فعال می‌شود. همزمان با غیرفعال شدن (صفر شدن) سنسور موقعیت ۱، سیگنال تحریک تزریق کننده ۱ نیز غیرفعال می‌شود. همین اتفاق‌ها برای سنسور و تزریق کننده شماره دو رخ می‌دهد، زمانی که سنسور موقعیت شماره ۲ فعال می‌شود، سیگنال تحریک تزریق کننده شماره ۲ فعال می‌شود. همزمان با غیرفعال شدن (صفر شدن) سنسور موقعیت ۲، سیگنال تحریک تزریق کننده ۲ نیز غیرفعال می‌شود.
COUNTING	اجزایی که بر روی تسمه نقاله جابجا می‌شوند، شمرده می‌شوند. در صورتی که تعداد آن‌ها از ۱۰۰ بیشتر شود، پرچم "وضعیت تکمیل" فعال می‌شود.

دو تابع بلوکی مطابق با ویژگی POUها ساخته خواهد شد.

- دو بخش تزریق کننده چسب یکسان هستند. پس می‌توان تابع بلوکی برای این فرایند ساخت و از آن دوبار در برنامه استفاده کرد. در نتیجه می‌توان گفت تابع بلوکی مورد نظر دوبار توسط GLUING فراخوانی می‌شود، همچنین چون دو تزریق کننده مستقل عمل می‌کنند، به آن‌ها دو نسخه تابع بلوکی مستقل می‌توان اختصاص داد.
- برنامه COUNTING کاملا منحصر به فرد، غیرتکرار شونده و مستقل از دیگر بخش‌ها است ولی برای خواناتر شدن، آن را به عنوان تابعی بلوکی در نظر می‌گیریم.

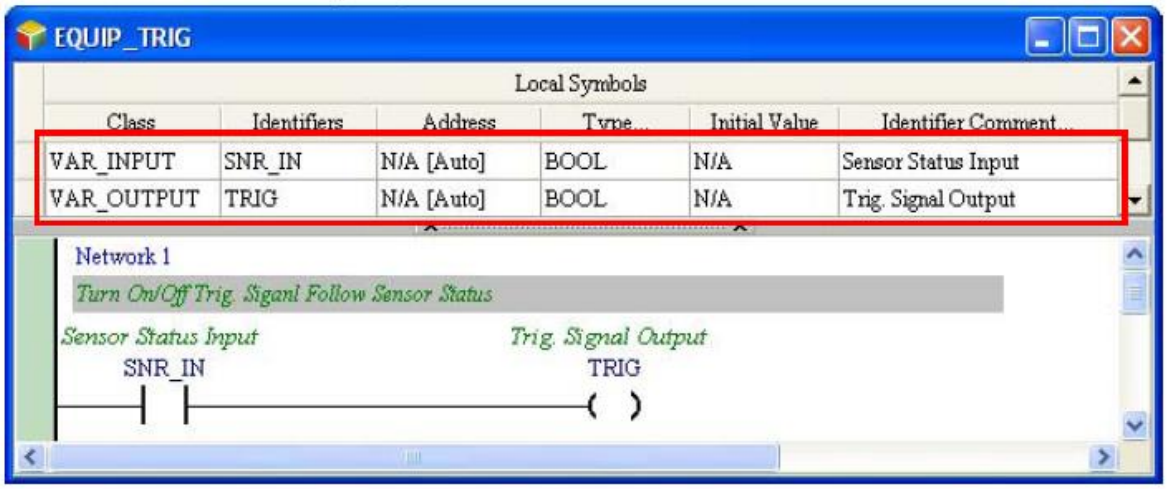
۷-۴-۱- ساخت برنامه

علاوه بر POUهای ساخته شده، تابع بلوکی با نام EQUIP_TRIG را در بخش مدیریت پروژه می‌سازیم.



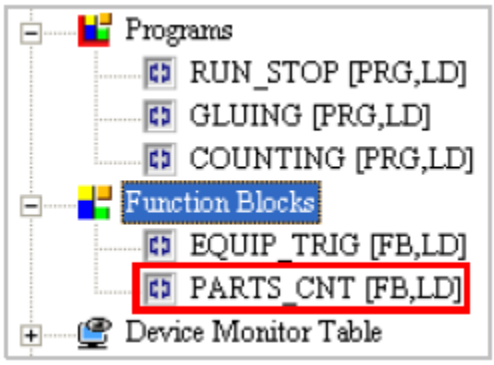
شکل ۷-۳۶ مثال – ساخت تابع بلوکی جدید EQUIP_TRIG

در EQUIP_TRIG برنامه‌ای برای تزریق چسب خواهیم نوشت. سیمبول‌ها و برنامه‌ی آن را مطابق شکل زیر تعیین می‌نویسیم.



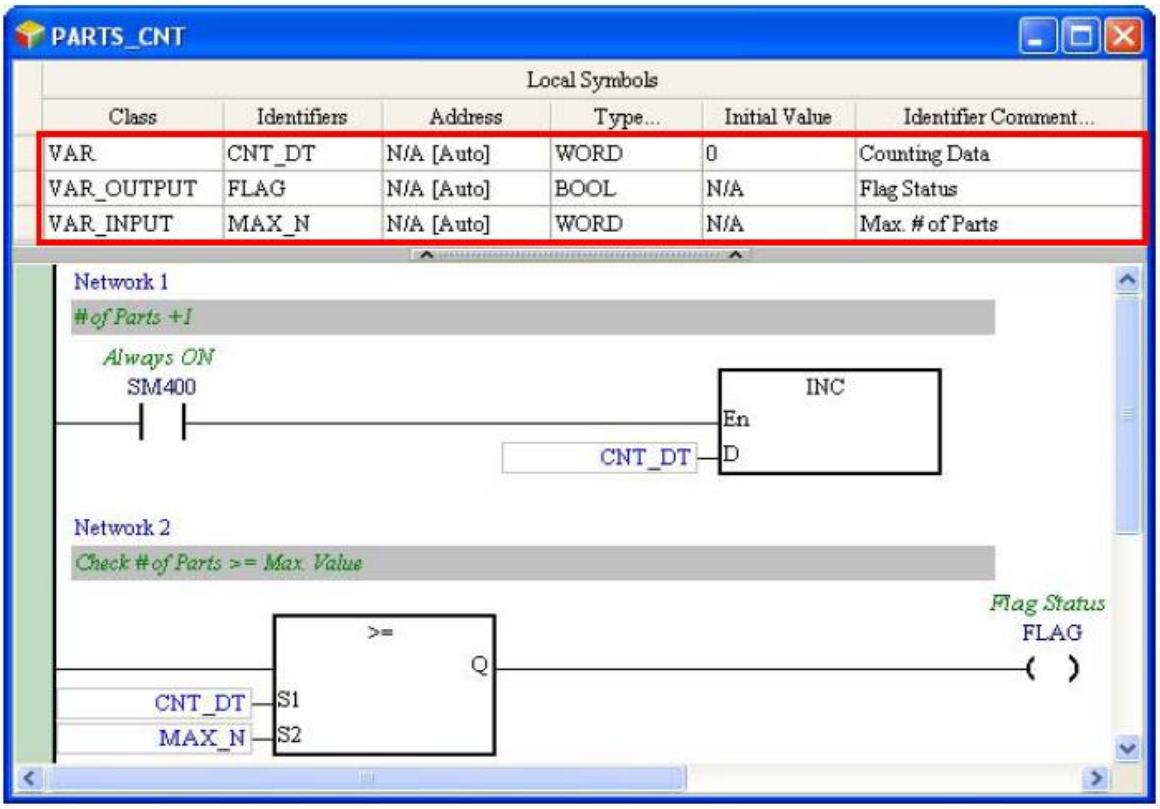
شکل ۷-۳۷ مثال – تابع بلوکی EQUIP_TRIG

همچنین برای شمارش اجزای روی تسمه تابع بلوکی PARTS_CNT را ایجاد می‌کنیم.



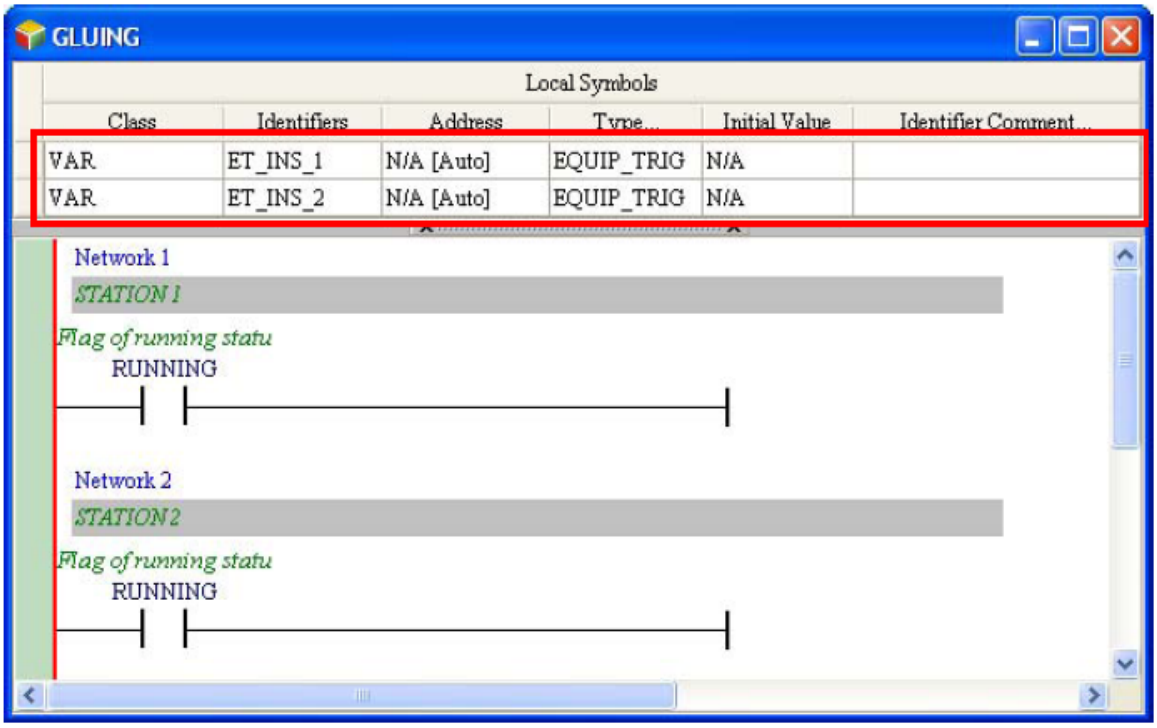
شکل ۷-۳۸ مثال- ساخت تابع بلوکی جدید PARTS_CNT

برنامه شمارنده اجزای روی تسمه نقاله را با مقدار اولیه صفر و جدول سیمبول محلی مطابق شکل زیر می‌نویسیم.



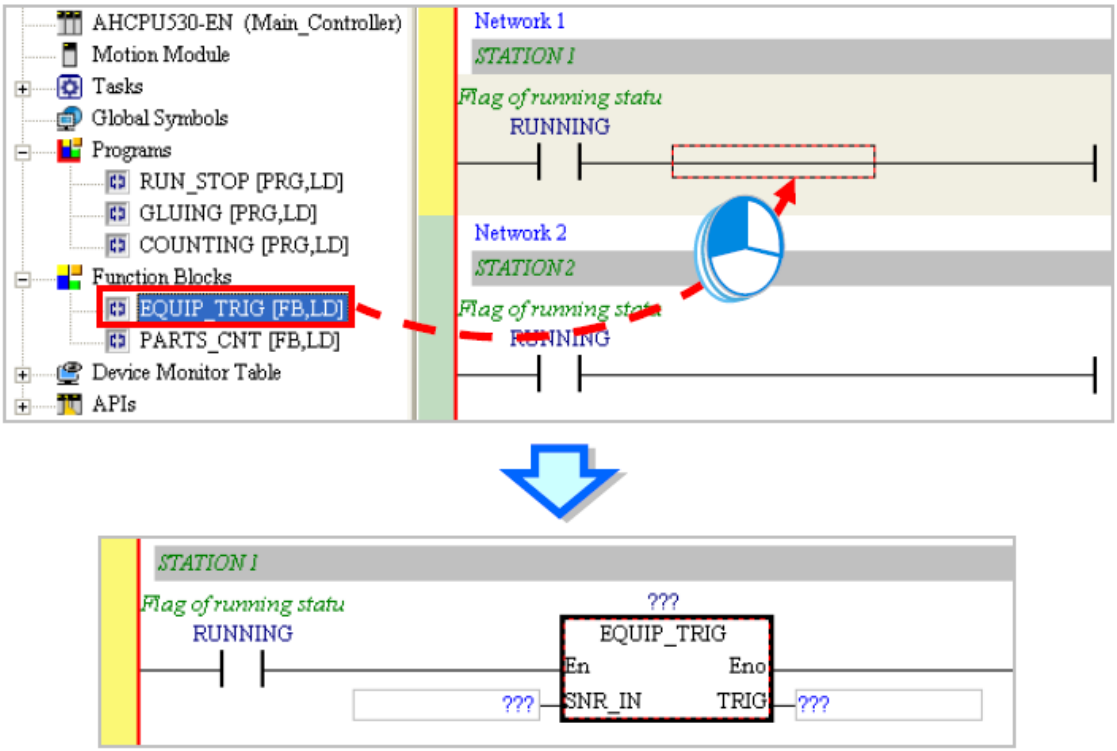
شکل ۷-۳۹ مثال- تابع بلوکی PARTS_CNT

POU ی GLUING تابع بلوکی EQUIP_TRIG را دو بار به صورت مستقل در دو نسخه فراخوانی می‌کند. برای اینکار نیاز به تعریف دو سیمبول محلی با فرمت تابع بلوکی داریم.



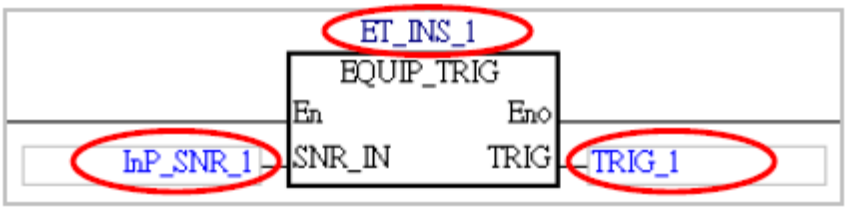
شکل ۷-۴۰ مثال – دوبار فراخوانی تابع بلوکی EQUIP_TRIG

برای فراخوانی در برنامه کافی است تابع بلوکی را از محیط مدیریت پروژه به محیط ویرایش برنامه بکشیم.



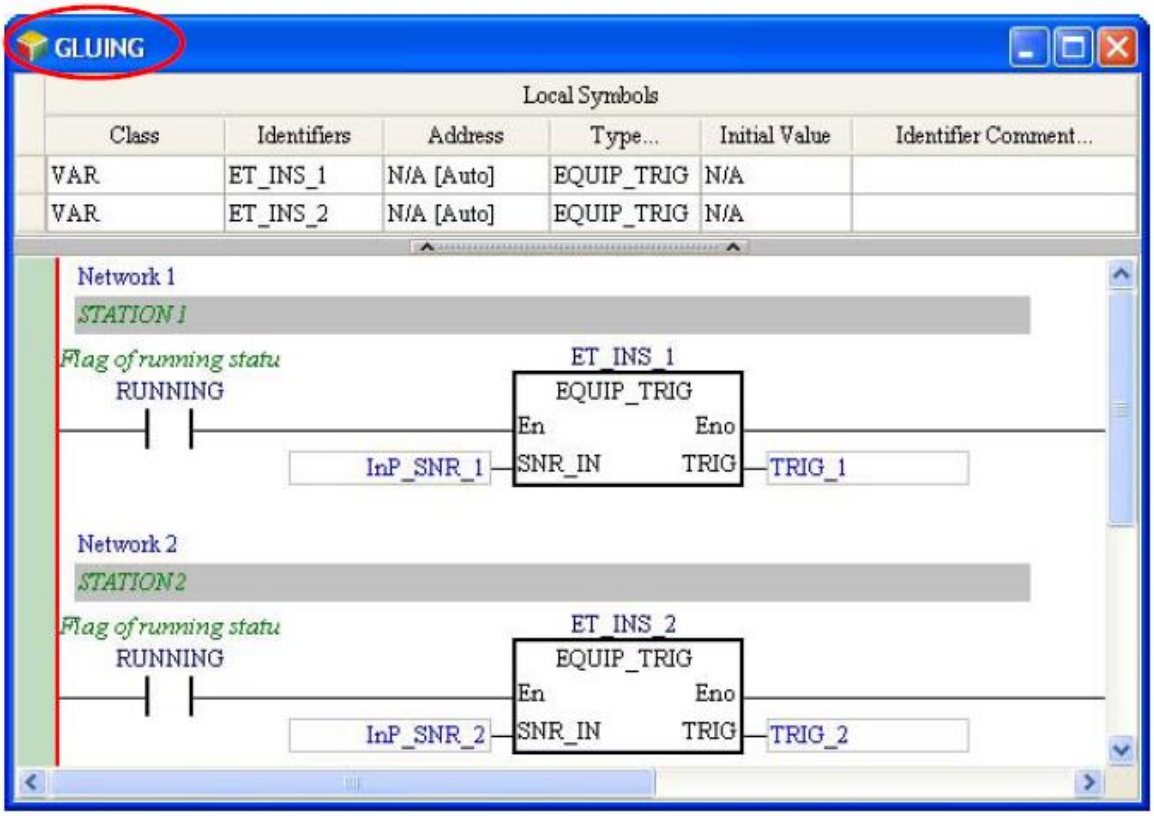
شکل ۷-۴۱ مثال- فراخوانی تابع بلوکی تابع بلوکی EQUIP_TRIG از تعریف آن

در این مرحله نیاز است ورودی و خروجی و دیتا بلاک هر تابع بلوکی را مشخص کنیم.



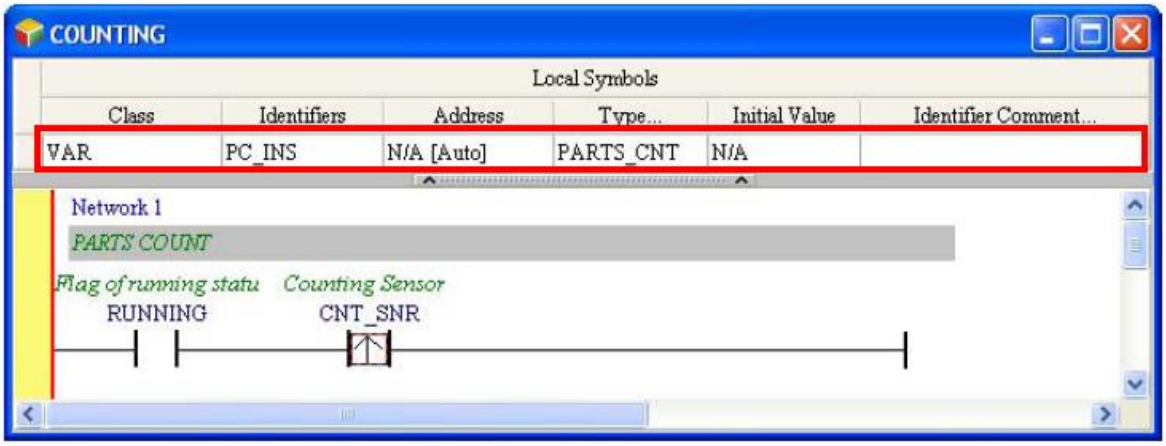
شکل ۷-۴۲ مثال- تنظیم ورودی و خروجی و دیتا بلاک تابع بلوکی EQUIP_TRIG

برنامه نوشته شده به صورت زیر در خواهد آمد.



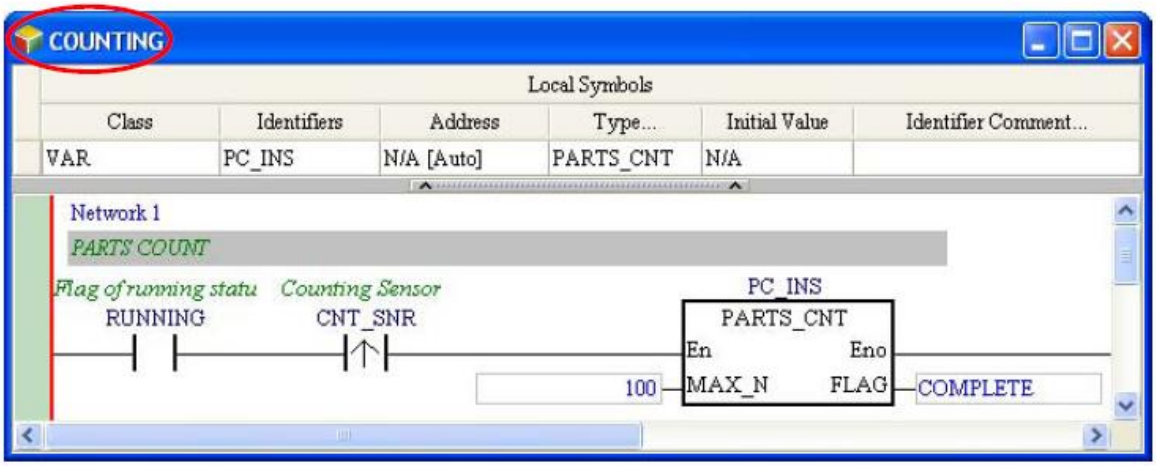
شکل ۷-۴۳ مثال- تنظیم هر دو تابع بلوکی تابع بلوکی EQUIP_TRIG

POU ی COUNTING تابع بلوکی PARTS_CNT را فراخوانی می‌کند. برای اینکار نیاز به تعریف سیمبول محلی با فرمت تابع بلوکی داریم تا از طریق آن مثال و دیتا بلاک تابع بلوکی ایجاد شود.



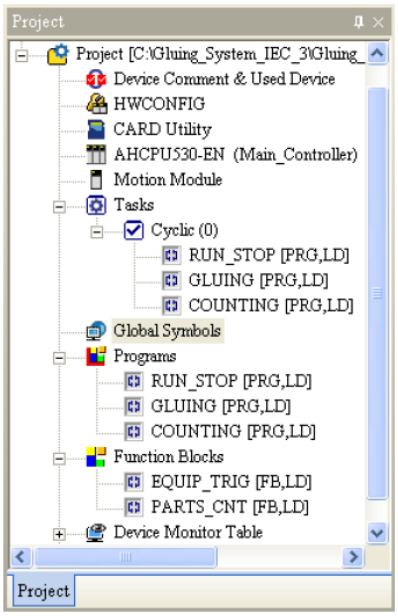
شکل ۷-۴۴ مثال – فراخوانی تابع بلوکی PARTS_CNT در COUNTING

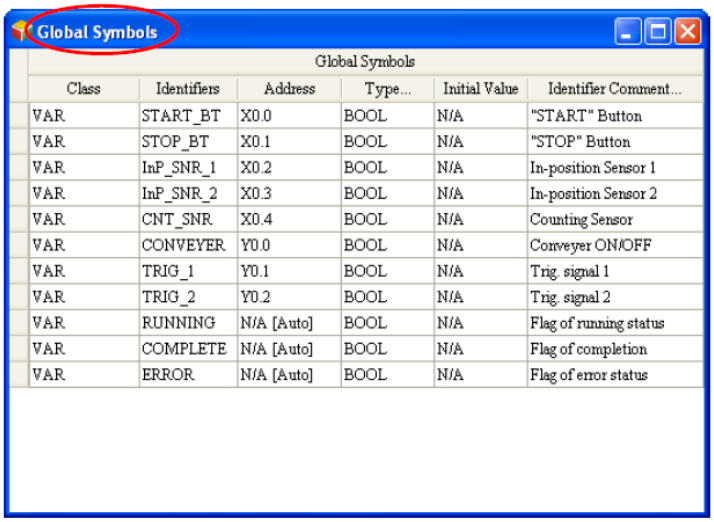
در ادامه این برنامه به صورت زیر در خواهد آمد.



شکل ۷-۴۵ مثال- تنظیم پارامترهای بلوک PARTS_CNT

در نهایت برنامه نوشته شده باید مطابق پروژه زیر شده باشد.

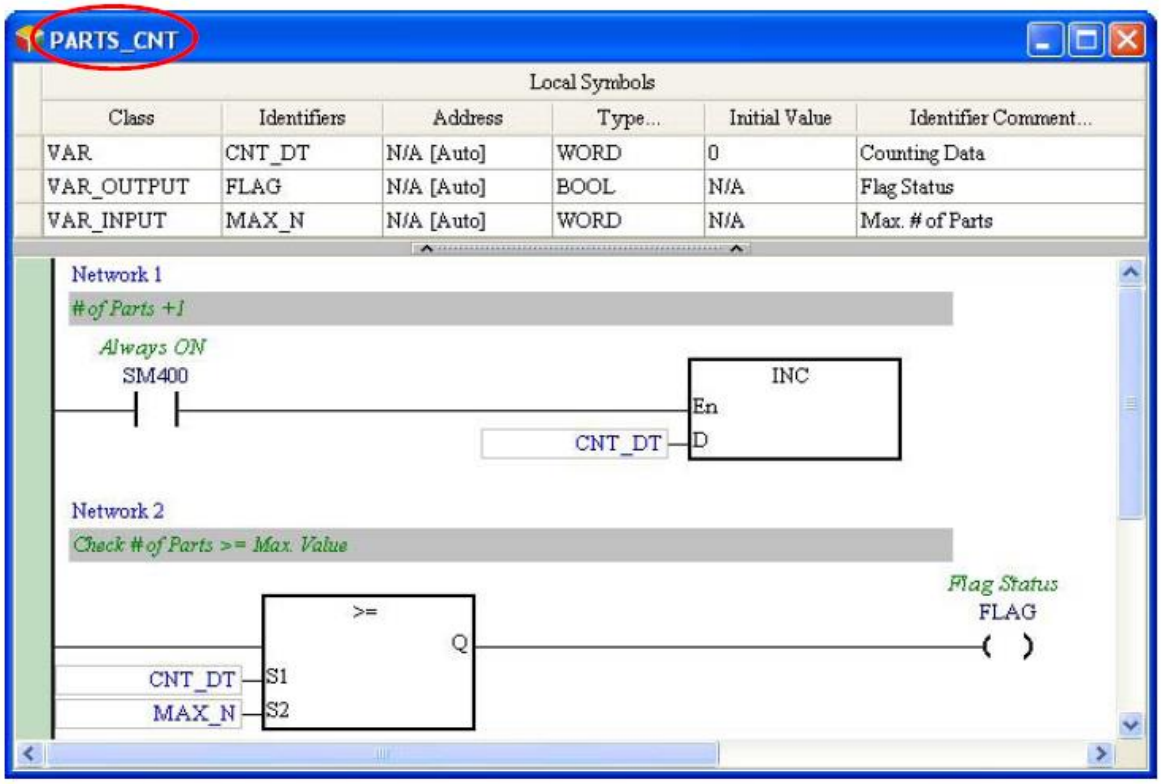


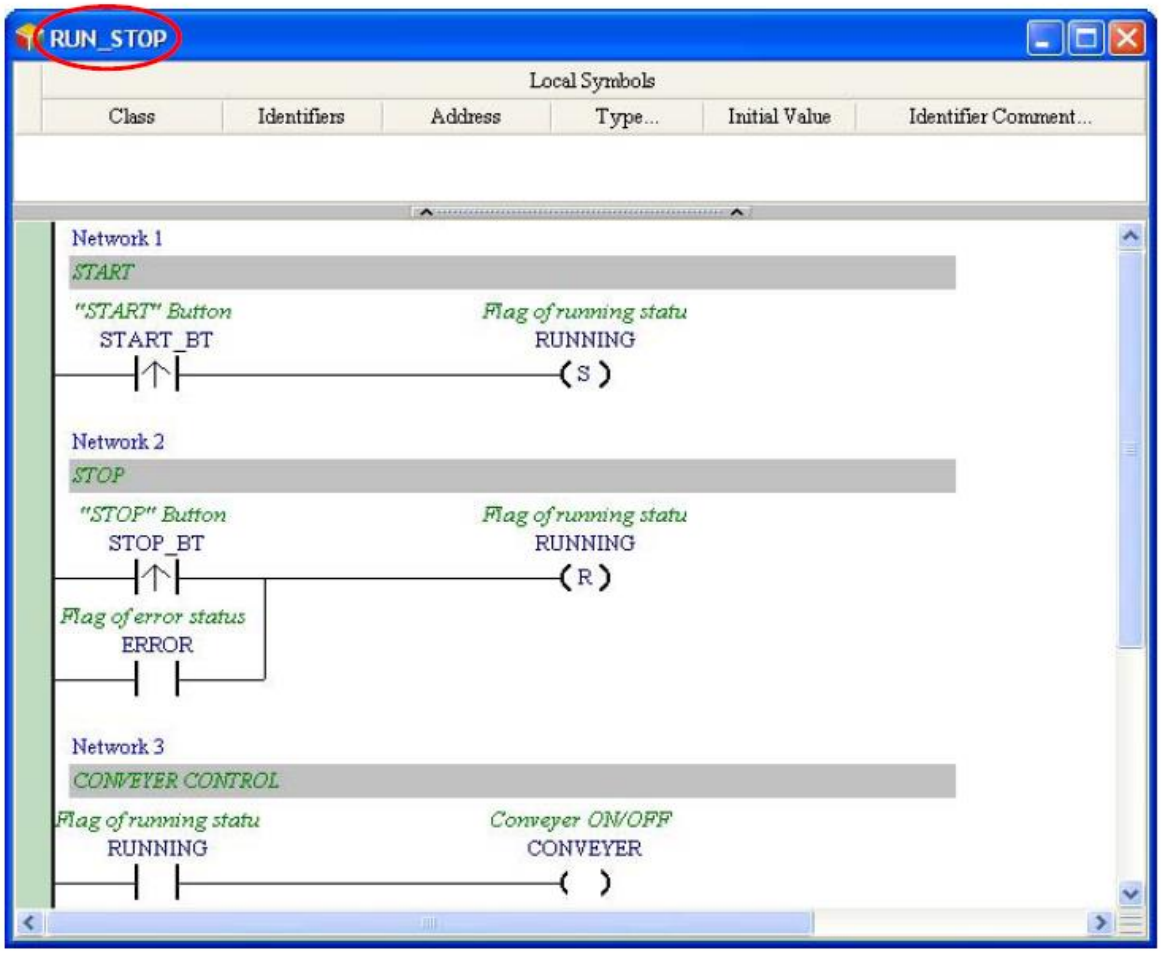


Class	Identifiers	Address	Type...	Initial Value	Identifier Comment...
VAR	START_BT	X0.0	BOOL	N/A	"START" Button
VAR	STOP_BT	X0.1	BOOL	N/A	"STOP" Button
VAR	InP_SNR_1	X0.2	BOOL	N/A	In-position Sensor 1
VAR	InP_SNR_2	X0.3	BOOL	N/A	In-position Sensor 2
VAR	CNT_SNR	X0.4	BOOL	N/A	Counting Sensor
VAR	CONVEYER	Y0.0	BOOL	N/A	Conveyer ON/OFF
VAR	TRIG_1	Y0.1	BOOL	N/A	Trig. signal 1
VAR	TRIG_2	Y0.2	BOOL	N/A	Trig. signal 2
VAR	RUNNING	N/A [Auto]	BOOL	N/A	Flag of running status
VAR	COMPLETE	N/A [Auto]	BOOL	N/A	Flag of completion
VAR	ERROR	N/A [Auto]	BOOL	N/A	Flag of error status

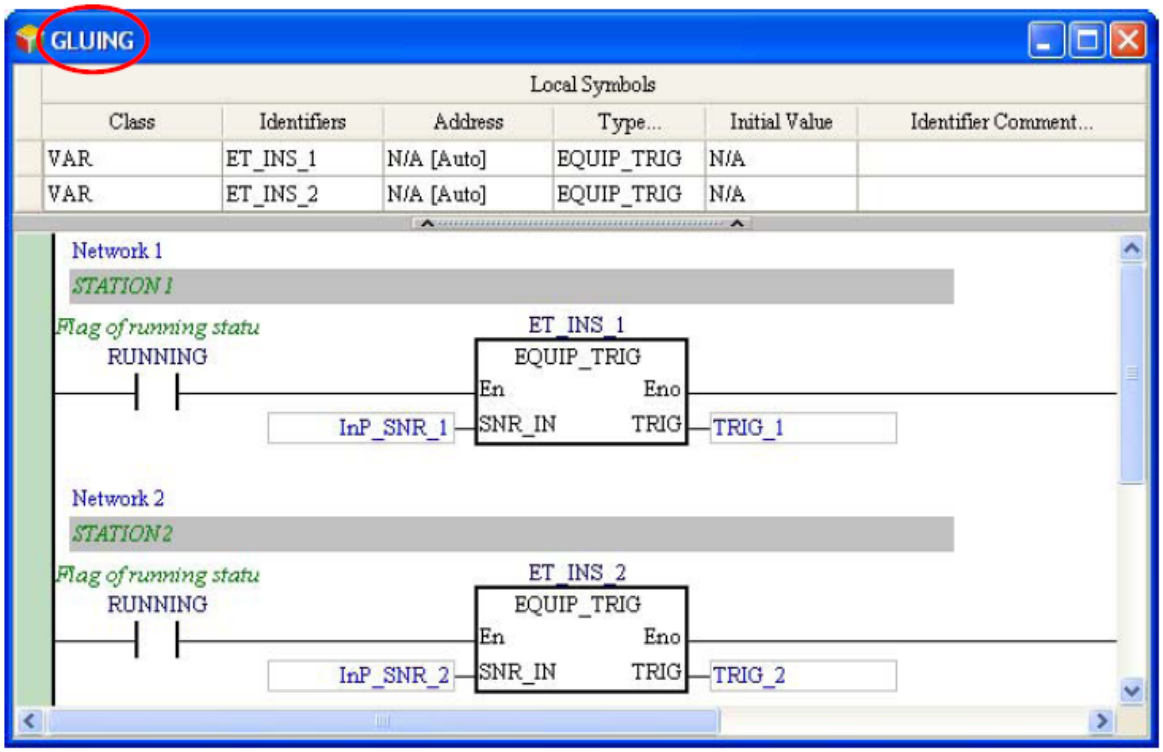
Class	Identifiers	Address	Type...	Initial Value	Identifier Comment...
VAR_INPUT	SNR_IN	N/A [Auto]	BOOL	N/A	Sensor Status Input
VAR_OUTPUT	TRIG	N/A [Auto]	BOOL	N/A	Trig. Signal Output

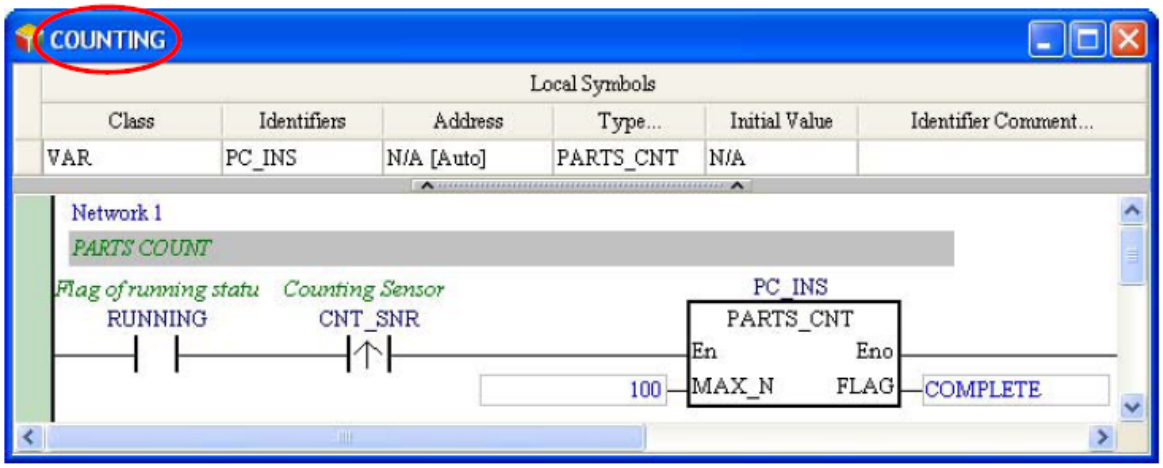
شکل ۷-۴۶ مثال- تکمیل برنامه نوشته شده – قسمت اول


PARTS_CNT
Local Symbols
Class
Identifiers
Address
Type...
Initial Value
Identifier Comment...
VAR
CNT_DT
N/A [Auto]
WORD
0
Counting Data
VAR_OUTPUT
FLAG
N/A [Auto]
BOOL
N/A
Flag Status
VAR_INPUT
MAX_N
N/A [Auto]
WORD
N/A
Max. # of Parts
Network 1
#of Parts +1
Always ON
SM400
INC
En
D
CNT_DT
Network 2
Check #of Parts >= Max. Value
Flag Status
FLAG
>=
Q
CNT_DT
S1
MAX_N
S2


RUN_STOP
Local Symbols
Class
Identifiers
Address
Type...
Initial Value
Identifier Comment...
Network 1
START
"START" Button
START_BT
Flag of running statu
RUNNING
S
Network 2
STOP
"STOP" Button
STOP_BT
Flag of running statu
RUNNING
R
Flag of error status
ERROR
Network 3
CONVEYER CONTROL
Flag of running statu
RUNNING
Conveyer ON/OFF
CONVEYER

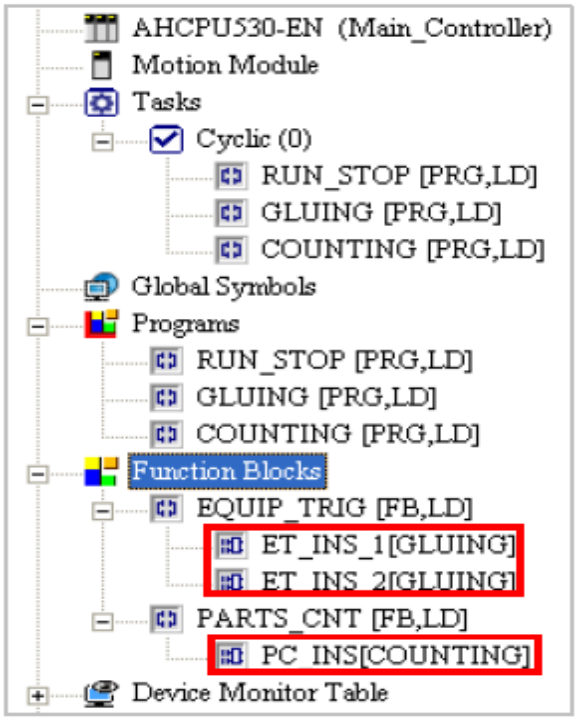
شکل ۷-۴۷ مثال- تکمیل برنامه نوشته شده – قسمت دوم





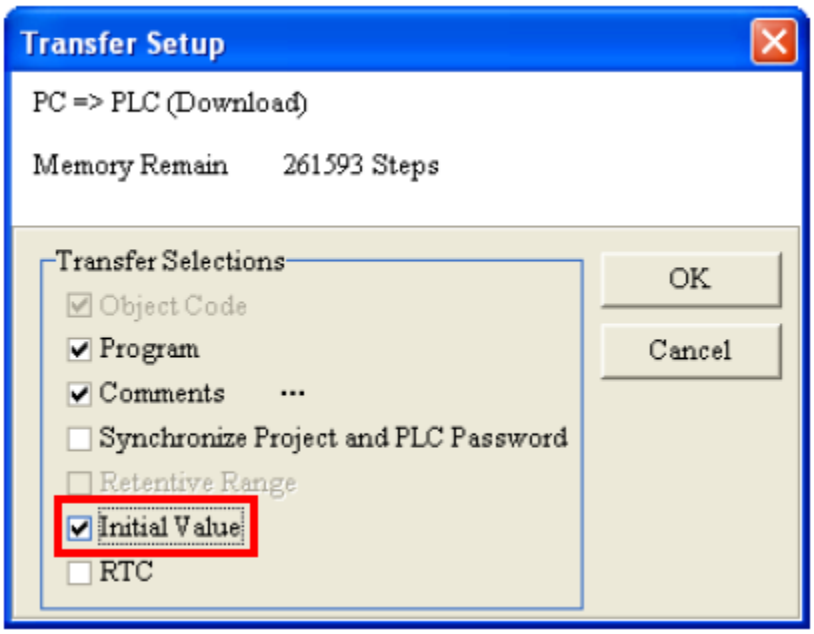
شکل ۷-۴۸ مثال- تکمیل برنامه نوشته شده – قسمت سوم

پس از کامپایل برنامه، نسخههای توابع بلوکی در بخش مدیریت پروژه مانند زیر ایجاد خواهند شد.



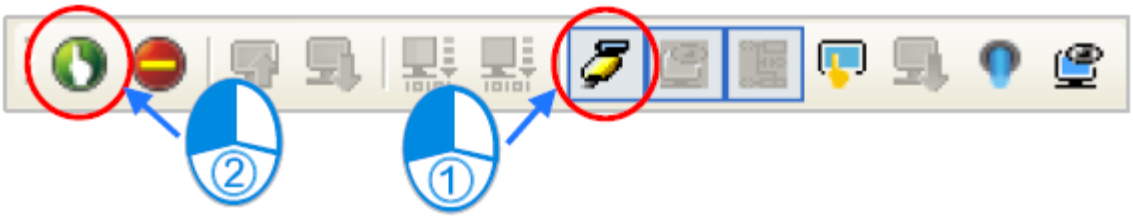
شکل ۷-۴۹ مثال- بخش مدیریت برنامه پس از کامپایل برنامه

پس از آنکه تنظیمات سخت افزاری و پارامترها و همچنین مدیریت Taskها کامل شد، کاربر می‌تواند این پروژه را کامپایل کرده و آن را بر روی PLC بارگزاری نماید. در بارگزاری کاربر باید گزینه Initial Value را در صفحه Transfer Setup فعال کند تا مقدار اولیه CNT_DT در PARTS_CNT نیز در CPU نوشته شود.



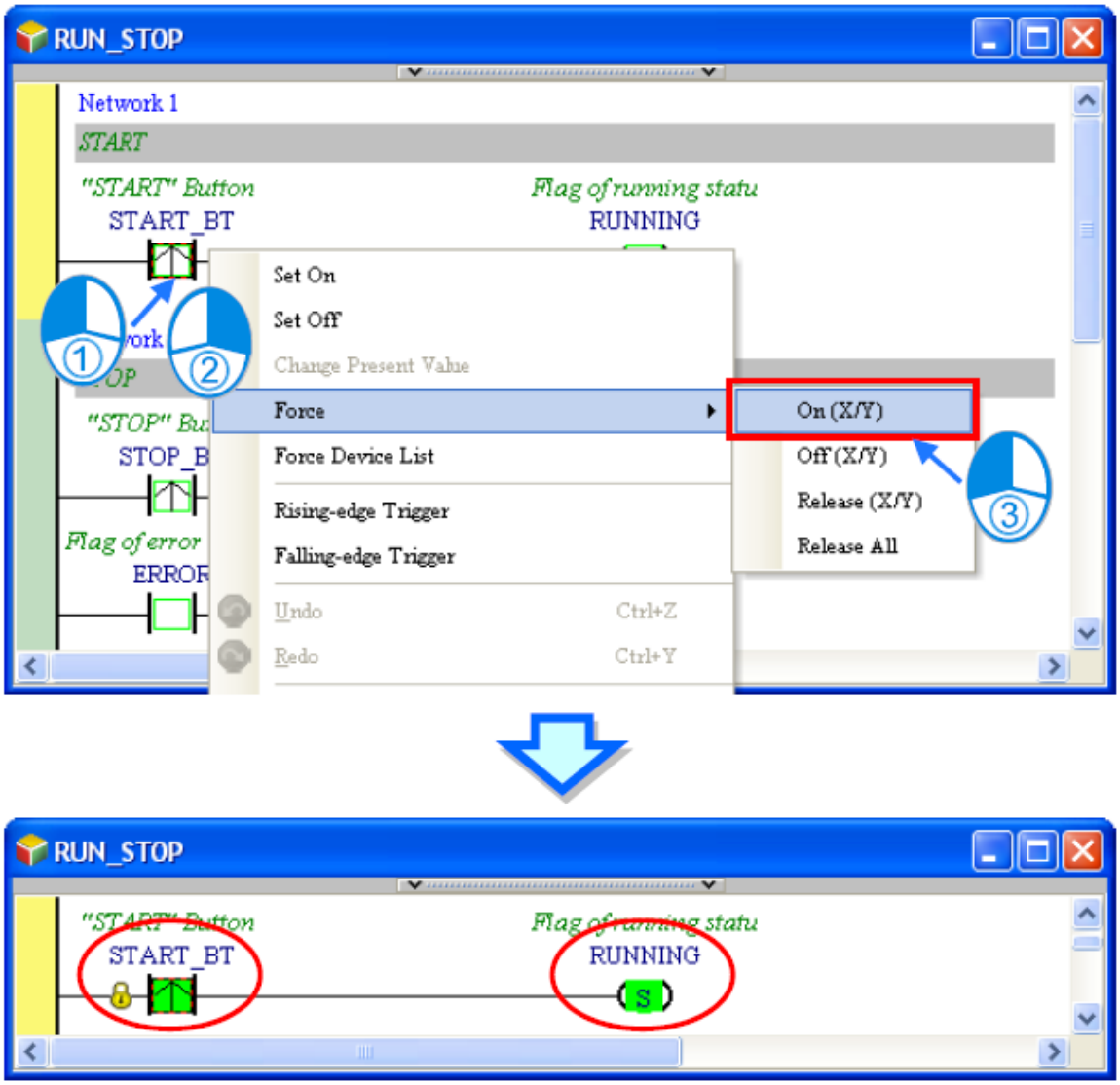
شکل ۷-۵۰ مثال- بارگزاری برنامه و مقادیر اولیه

پس از تکمیل دانلود برنامه با استفاده از ابزارهای و می‌توان در حالت آنلاین عملکرد تابع بلوکی را تست کرد.



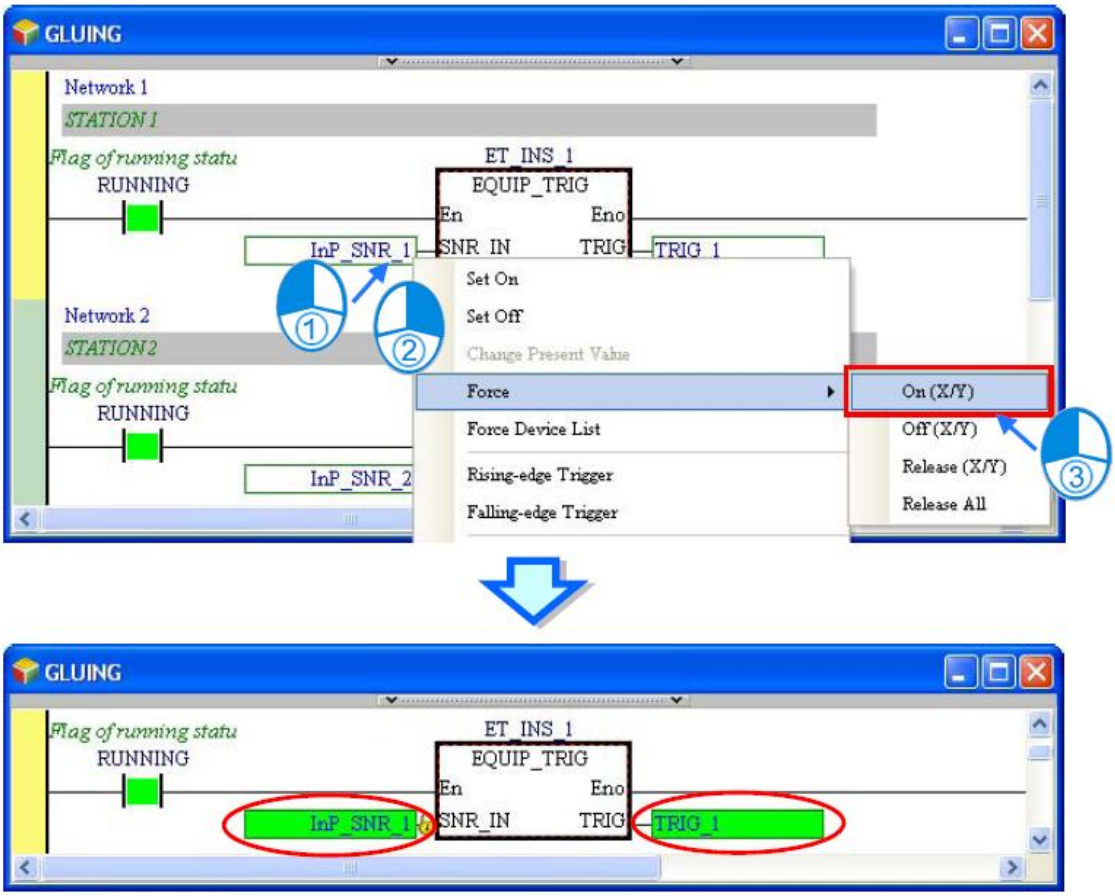
شکل ۷-۵۱ مثال- مانیتورینگ آنلاین برنامه – قسمت اول

پنجره RUN_STOP را باز کرده و START_BT را در وضعیت اجبار فعال کنید. RUNNING فعال شده و تجهیزات شروع به کار می‌کنند.



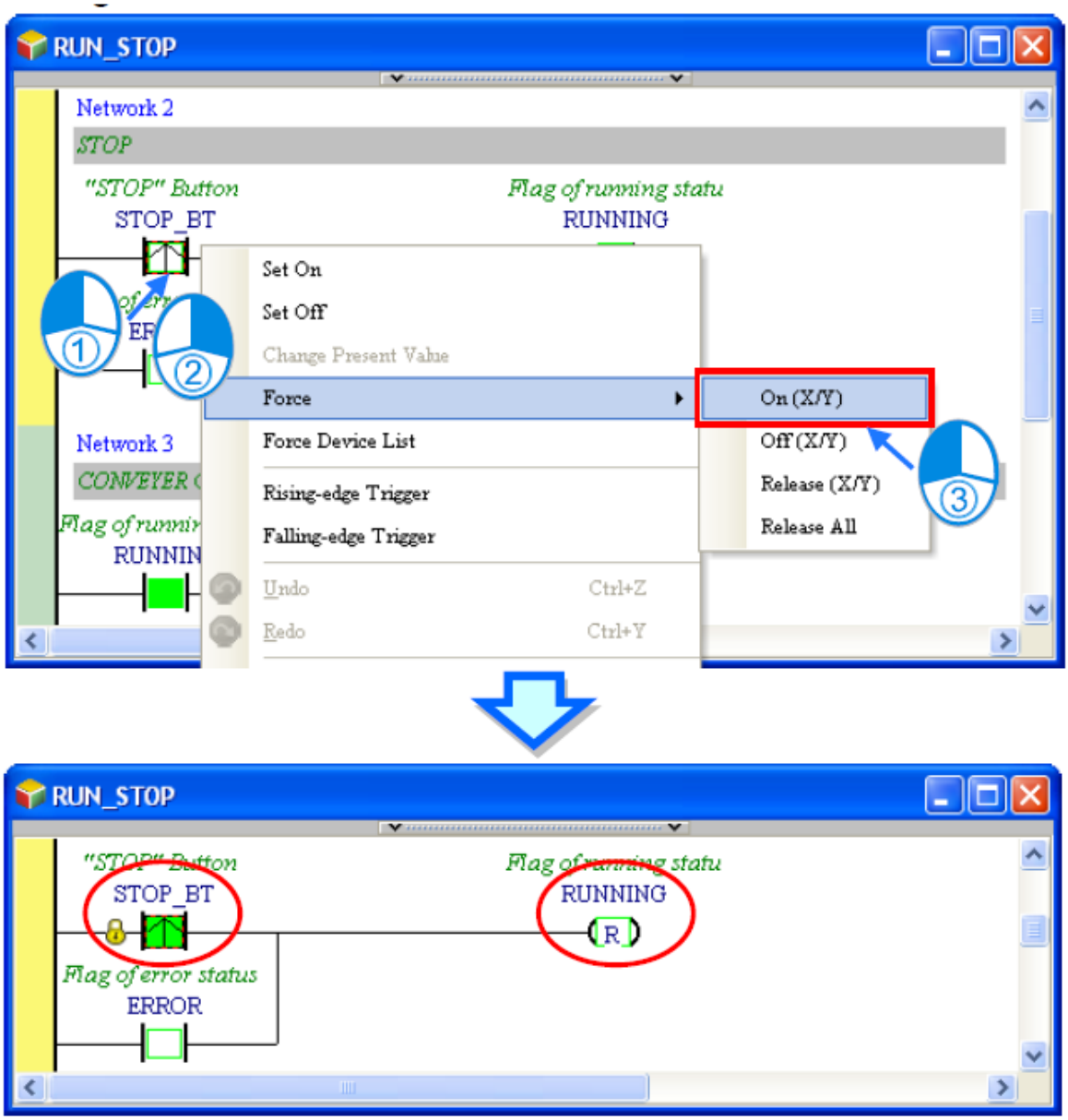
شکل ۷-۵۲ مثال- مانیتورینگ آنلاین برنامه – قسمت دوم

پنجره GLUING را باز کرد و InP_SNR_1 را در حالت اجبار فعال کنید، در این حالت سنسور قطعه کار را تشخیص می‌دهد و سیگنال تحریک را ایجاد می‌کند.



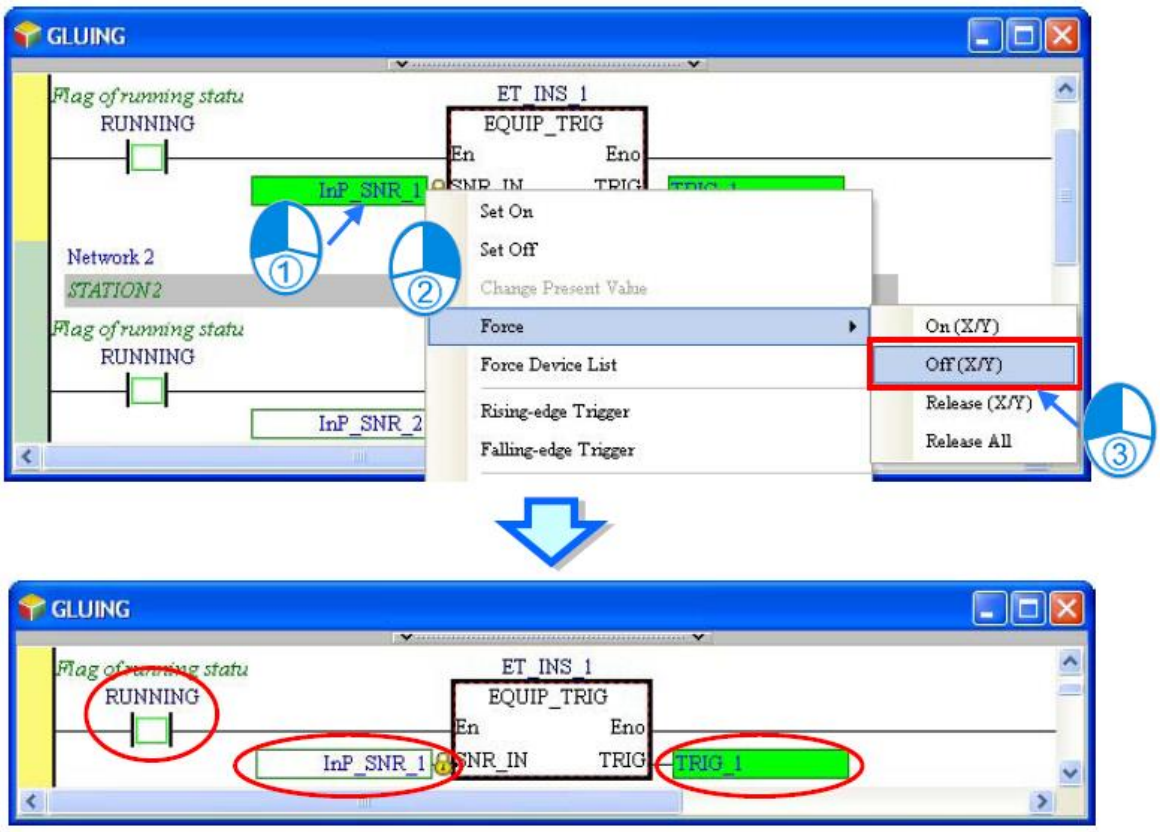
شکل ۷-۵۳ مثال- مانیتورینگ آنلاین برنامه – قسمت سوم

در ادامه با باز کردن پنجره RUN_STOP و فعال کردن STOP_BT در حالت اجبار می‌بینیم که RUNNING غیر فعال شده و تجهیزات از حرکت متوقف می‌شوند.



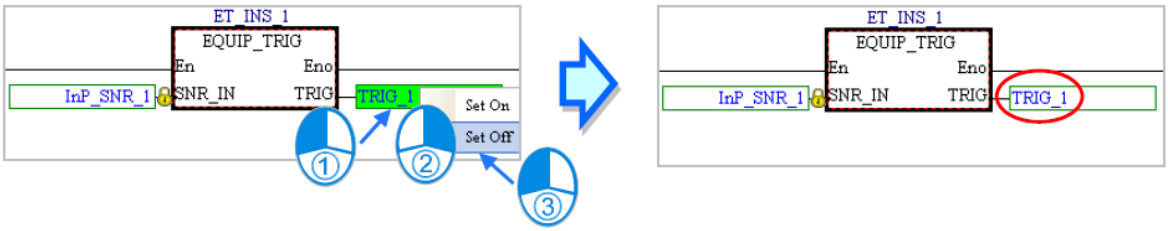
شکل ۷-۵۴ مثال- مانیتورینگ آنلاین برنامه – قسمت چهارم

پنجره GLUING را باز کرده و در حالت اجبار InP_1 را غیرفعال می‌کنیم. در این حالت TRIG_1 همچنان فعال است و وضعیت عادی برقرار خواهد بود. از آن‌جایی که RUNNING پرچم متصل به ورودی En تابع بلوکی است با غیرفعال شدن آن تابع بلوکی نیز اجرا نخواهد شد، در این حالت مقدار TRIG از پایه تابع بلوکی به سیمبول TRIG_1 منتقل نمی‌شود و مقدار TRIG_1 بدون تغییر خواهد ماند.



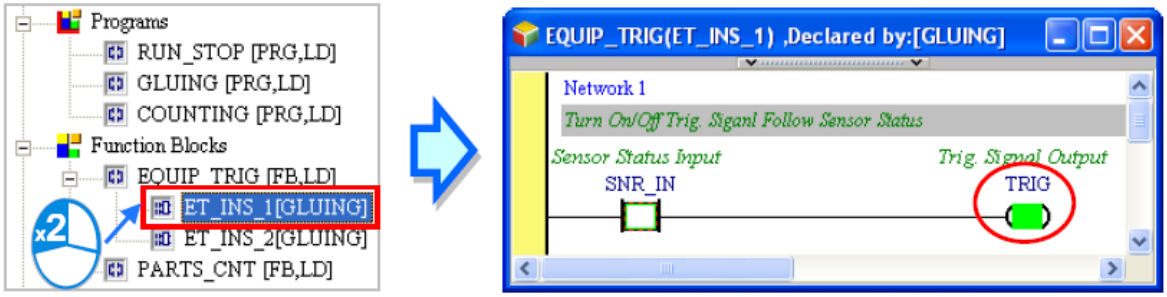
شکل ۷-۵۵ مثال- مانیتورینگ آنلاین برنامه – قسمت پنجم

زمانی که TRIG_1 خاموش شود:



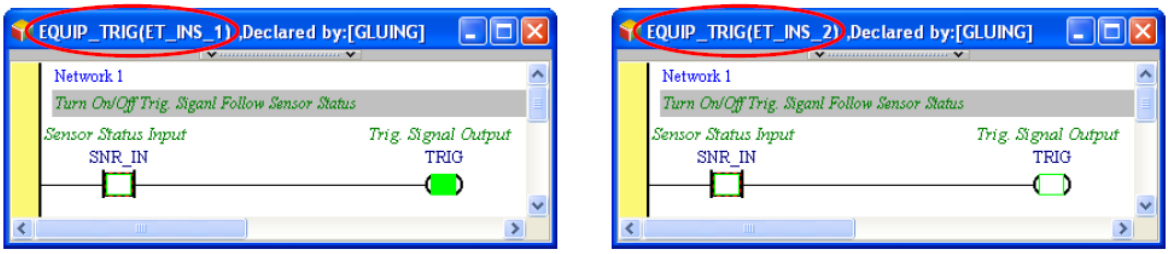
شکل ۷-۵۶ مثال- مانیتورینگ آنلاین برنامه – قسمت ششم

زمانی که پنجره ET_INS_1 باز می‌شود، مقدار TRIG در آن روشن باقی می‌ماند، این موضوع نشان می‌دهد وضعیت سیمبول‌های تابع بلوکی بعد از اجرای آن ثابت مانده است.



شکل ۷-۵۷ مثال- مانیتورینگ آنلاین برنامه – قسمت هفتم

پنجره ET_INS_2 را باز کرده و TRIG را در آن خاموش می‌کنیم. وضعیت TRIG در پنجره ET_INS_2 با وضعیت TRIG در ET_INS_1 متفاوت است و این نشان می‌دهد مثال‌های تابع بلوکی به صورت مستقل از هم اجرا می‌شوند.



شکل ۷-۵۸ مثال- مانیتورینگ آنلاین برنامه – قسمت هشتم

پس از انجام مراحل فوق می‌توان در مورد عملکرد توابع بلوکی به صورت زیر نتیجه گیری کرد:

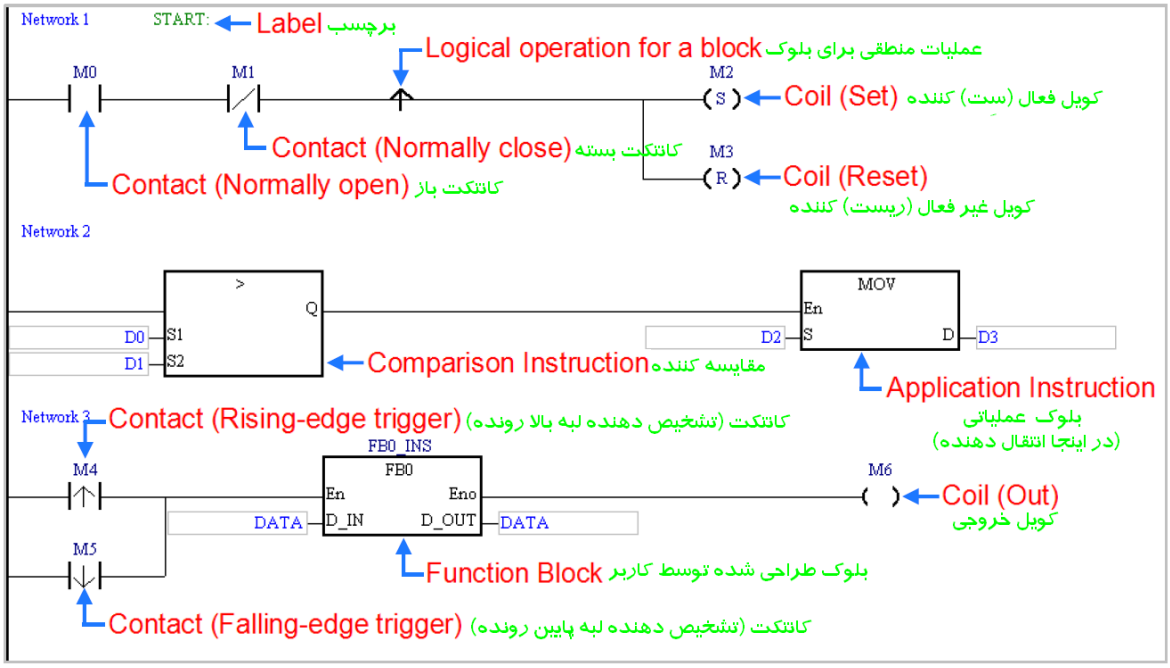
۱- در صورتی که ورودی پایه En تابع بلوکی صفر منطقی باشد، برنامه آن تابع بلوکی اجرا نخواهد شد.

۲- وضعیت سیمبول‌ها در توابع بلوکی پس از اجرا حفظ خواهد شد.

۳- نسخههای هر تابع بلوکی به صورت مستقل از هم اجرا می‌شوند.

فصل ۸– زبان برنامه نویسی Ladder

به جرات می‌توان گفت زبان برنامه نویسی Ladder پر استفاده ترین زبان برنامه نویسی برای PLCها می باشد. زبان Ladder زبانی گرافیکی منطبق بر منطق عملکرد رله‌ها وکانتکت‌ها[1] است. المان‌های اصلی این زبان برنامه نویسی در محیط ISPSoft در شکل زیر نشان داده شده است.



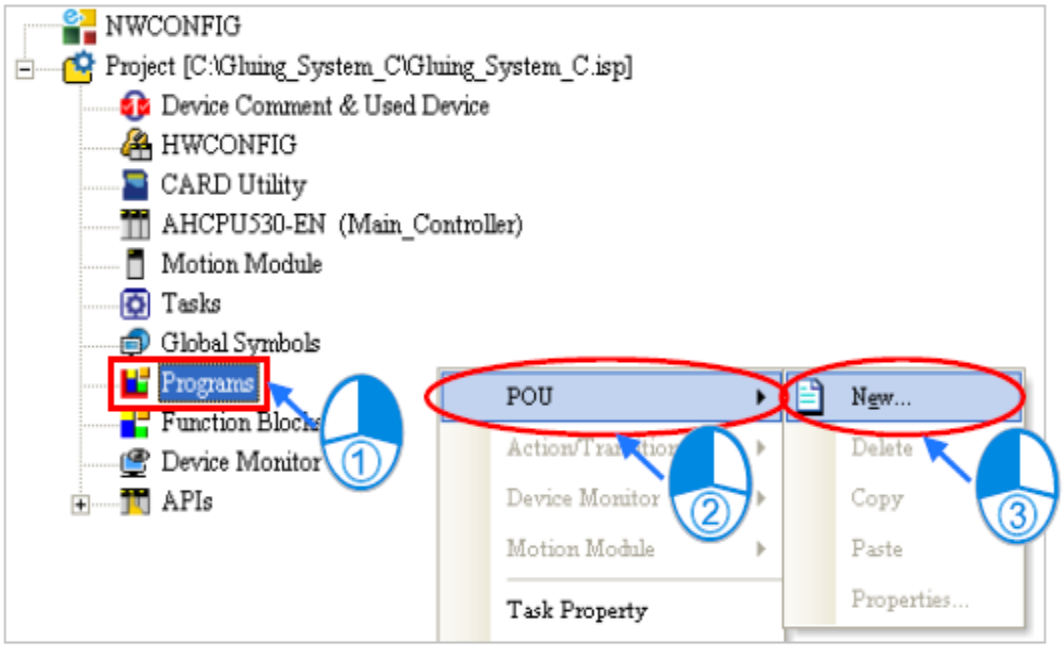
شکل ۸-۱ اجزای اصلی زبان برنامه نویسی Ladder در ISPSoft

۸–۱–۱– محیط برنامه نویسی

(توجه: ابتدای این بخش در فصل ۴–فصل ۴–راه اندازی اولیه" برای آشنایی سریع با اصول برنامه نویسی PLC نیز آمده است)

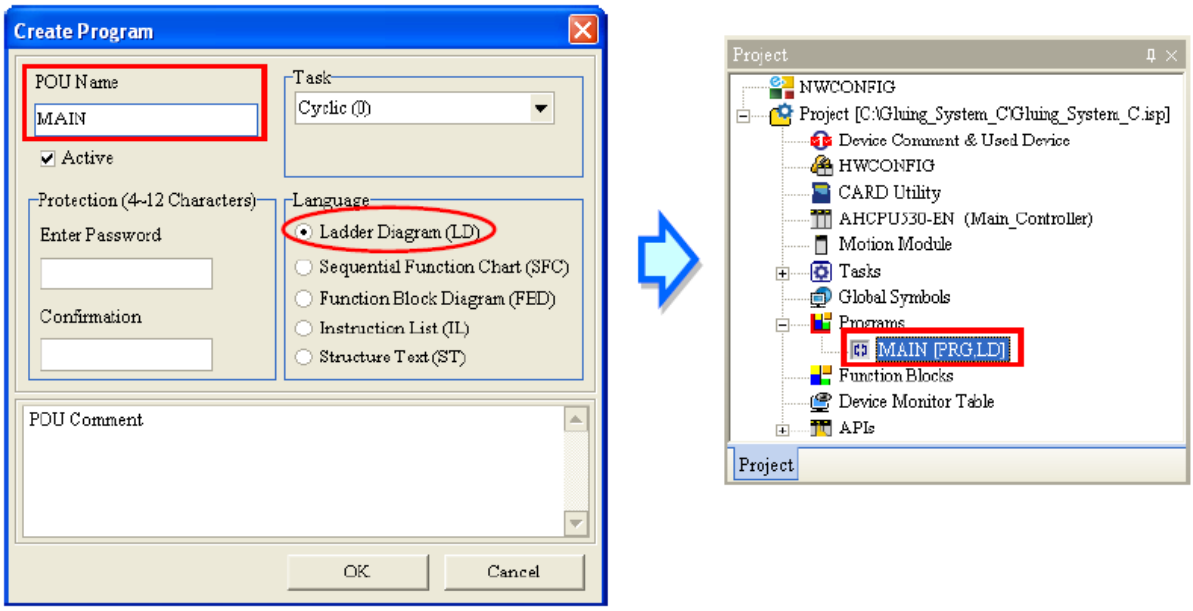
برای شروع برنامه نویسی نیاز به اضافه کردن محیط برنامه نویسی جدید است که برای اینکار در بخش مدیریت پروژه بر روی Program کلیک راست کرده و POU و سپس New را انتخاب می‌کنیم.

[1] Relay Logic روشی برای کنترل مدارات الکترونیک صنعتی با استفاده از رله و کانتکت ها است.



شکل ۸-۲ ایجاد برنامه جدید در پروژه

در پنجره ظاهر شده پس از تعیین نام پروژه زبان برنامه نویسی را Ladder Diagram(LD) انتخاب می‌کنیم.



شکل ۸-۳ انتخاب زبان برنامه نویسی Ladder

پس از تایید، POU[1] مورد نظر ما برای نوشتن برنامه به عنوان زیرشاخه‌ای از Program در بخش مدیریت پروژه اضافه می‌شود. در این مرحله صفحه‌ای برای ویرایش برنامه باز خواهد شد.

شکل ۸-۴ محیط برنامه نویسی (ویرایش برنامه)

در این مرحله نوار ابزار مرتبط با زبان برنامه نویسی Ladder نیز ظاهر شده و می‌توانیم از آن در برنامه نویسی استفاده کنیم. آیکون های این نوار ابزار را در ادامه مرور خواهیم کرد.

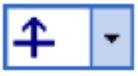
شکل ۸-۵ نوار ابزار برنامه نویسی Ladder

جدول ۸-۱: آیکون های نوار ابزار برنامه نویسی Ladder

نماد	کلید میانبر	شرح
	Shift+Ctrl+C	نمایش/عدم نمایش توضیحات در networks[2]
		نمایش/ عدم نمایش توضیحات اجزای برنامه ladder
	Shift+Ctrl+A	فعال/غیر فعال سازی network انتخاب شده
	Shift+Ctrl+B	نشانه گذاری و یا پاک کردن نشانه‌ها در network-ها
	Shift+Ctrl+P	رفتن به نشان بعدی

[1] Program Organization Unit

[2] منظور از Network در اینجا زنجیره خطوط زبان برنامه نویسی ladder است که برای جدا کردن بخش‌های مختلف برنامه مورد استفاده قرار می‌گیرد و نباید با شبکه‌های ارتباطی اشتباه شود.

	Shift+Ctrl+N	رفتن به نشان قبلی
	Ctrl+l	اضافه کردن network بالای network انتخاب شده
	Shift+Ctrl+l	اضافه کردن network پایین network انتخاب شده
	ESC	نمایشگر موس برای انتخاب اجزای برنامه
		اضافه کردن کانتکت[1]
		اضافه کردن کویل[2]
		اضافه کردن کانتکت مقایسه‌گر
		انتخاب نوع عملیات مقایسه‌ای
		اضافه کردن عملیات منطقی
		انتخاب نوع عملیات منطقی
	Shift+Ctrl+U	اضافه کردن بلوک (نوع آن را پس از کلیک کردن بر روی این آیکون باید تعیین کنید)

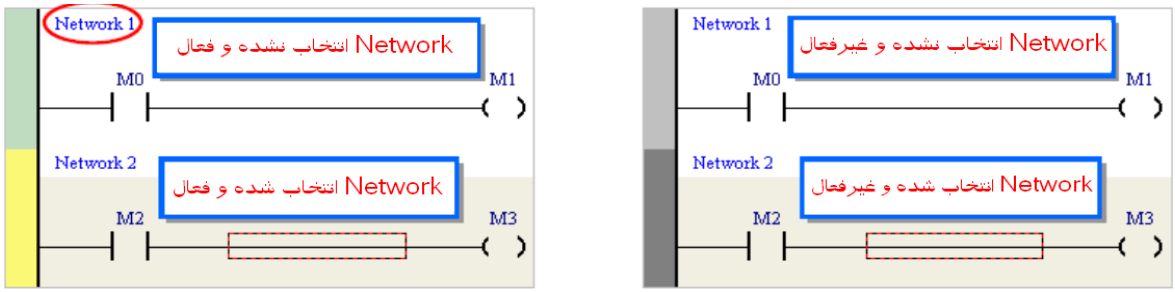
۸-۱-۲- Network در Ladder

برنامه‌ها در زبان برنامه نویسی Ladder از بخش‌هایی به نام Network به صورت پی در پی و زنجیره-وار تشکیل شده‌اند، این بخش‌ها شامل قسمت‌های مستقلی از برنامه هست. در ISPSoft محدودیتی برای تعداد اجزای هر Network وجود ندارد و به همین دلیل می‌توان حتی تمامی برنامه را در یک Network نوشت با این حال برای اینکه برنامه خواناتر و منظم‌تر باشد، استفاده از Networkها ضروری است.

شماره Network در قسمت بالای محیط برنامه نویسی آن مشخص شده است. رنگ سمت چپ Network نمایش دهنده وضعیت آن است: فعال/ غیرفعال، انتخاب شده/انتخاب نشده. این رنگ‌ها در پنجره Options قابل تغییر است.

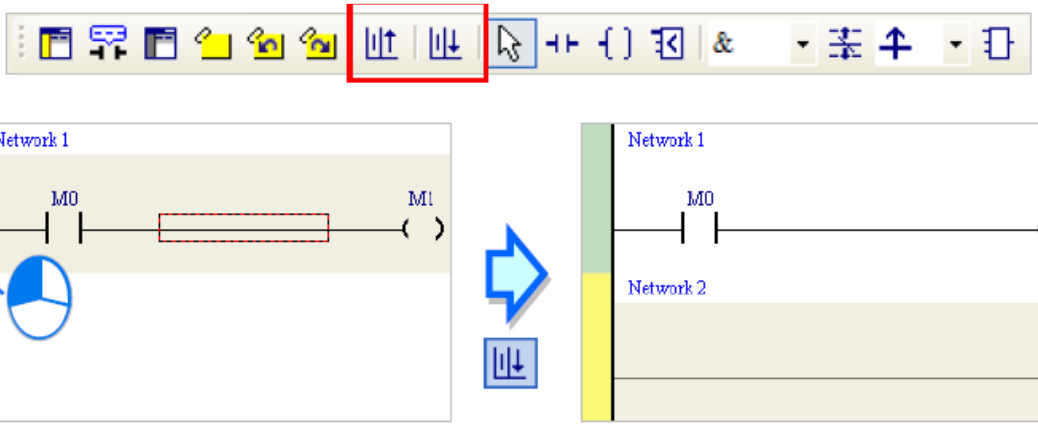
[1] Contact

[2] Coil



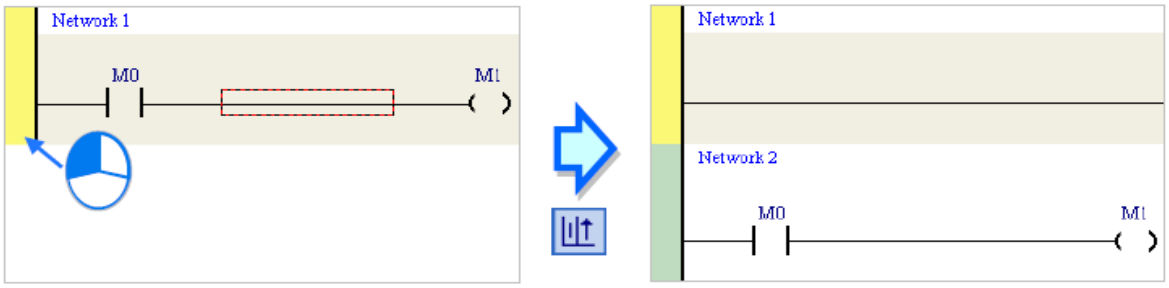
شکل ۸-۶ وضعیت Network ها

پس از آنکه پنجره برنامه POU جدید باز شد، یک Network خالی در آن قرار دارد. اگر کاربر بخواهد Network-ای اضافه کند، می‌تواند ابتدا یکی از Networkهای برنامه را انتخاب کند و پس از کلیک بر روی Network-ای در زیر Network قبلی اضافه خواهد شد.



شکل ۸-۷ اضافه کردن Network زیر Network انتخاب شده

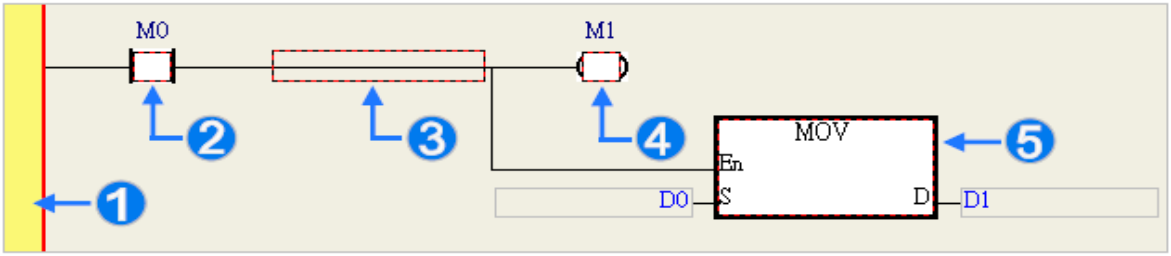
برای اضافه شدن Network در بالای Network انتخاب شده نیز می‌توان بر روی کلیک کرد.



شکل ۸-۸ اضافه کردن Network بالای Network انتخاب شده

۸–۱–۳– انتخاب اجزای برنامه

برای انتخاب اجزای محیط برنامه نویسی Ladder می‌توان از استفاده کرد. همچنین برای خارج شدن از حالت انتخاب نیز می‌توان از کلید Esc بر روی صفحه کلید استفاده کرد. اجزایی که در صفحه برنامه Ladder امکان انتخاب آن‌ها وجود دارد، در شکل زیر آمده‌اند.



شکل ۸-۹ انتخاب اجزای صفحه ویرایش برنامه Ladder

1. انتخاب Network
2. انتخاب کانتکت ورودی
3. انتخاب شبکه برای اضافه کردن المان
4. انتخاب کویل خروجی
5. انتخاب بلوک

۸–۲– Ladder در نرم افزار ISPSoft

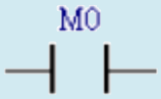
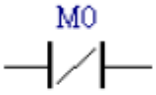
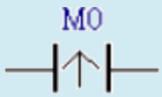
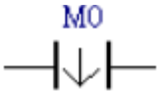
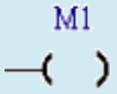
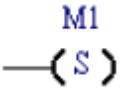
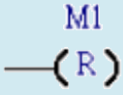
۸–۱–۲– کانتکت ها و کویل ها

با استفاده از کانتکت‌ها ورودی در برنامه می‌توان رجیسترها را خواند و با استفاده از کویل‌ها می‌توان آن‌ها را ویرایش کرد[۱]. توضیحات مربوط به کانتکت‌ها و کویل‌های نرم افزار ISPSoft در جدول زیر آمده است:

جدول ۸-۲: کانتکت و کویل های نرم افزار ISPSoft

نام عملکرد	نماد	شرح

[۱] حتی پورت‌های ورودی و خروجی نیز به حافظه‌های X و Y متصل هستند و PLC از طریق این رجیسترها با پورت‌ها در ارتباط است.

کانتکت (ورودی برنامه[1])		کانتکت در حالت عادی باز[2] (به طور مختصر کانتکت باز): این کانتکت مقدار دقیق ورودی خود را (در اینجا M0) را منتقل می کند.
		کانتکت در حالت عادی بسته[3] (به طور مختصر کانتکت بسته): این کانتکت مقدار معکوس منطقی ورودی خود را (در اینجا M0) را منتقل می کند. مثلا اگر M0 یک باشد، این کانتکتNOT آن یعنی صفر را منتقل و اگر M0 برابر صفر منطقی باشد، این کانتکت یک را منتقل می‌کند.
		کانتکت تشخیص دهنده لبه بالا رونده[4]: این کانتکت در صورتی که ورودی آن (یعنی M0) مقدارش از صفر به یک تغییر کند، به مدت یک سیکل مقدارش یک خواهد شد.
		کانتکت تشخیص دهنده لبه پایین رونده[5]: این کانتکت در صورتی که ورودی آن (یعنی M0) مقدارش از یک به صفر تغییر کند، به مدت یک سیکل مقدارش یک خواهد شد.
کویل (خروجی برنامه)		کویل خروجی: مقدار آن به خروجی (در اینجا M1) "منتقل" می‌شود.
		کویل فعال کننده خروجی: در صورت فعال شدن آن (یک شدن)، خروجی (در اینجا M1) یک می‌شود.
		کویل ریست (غیرفعال) کننده خروجی: در صورت فعال شدن آن (یک شدن)، خروجی (در اینجا M1) صفر می‌شود.

[1] ورودی برنامه در اینجا منظور همان ورودی PLC نیست، ممکن است حتی از رجیستر Y که رجیستر خروجی سیستم است به عنوان ورودی برنامه استفاده کنیم.

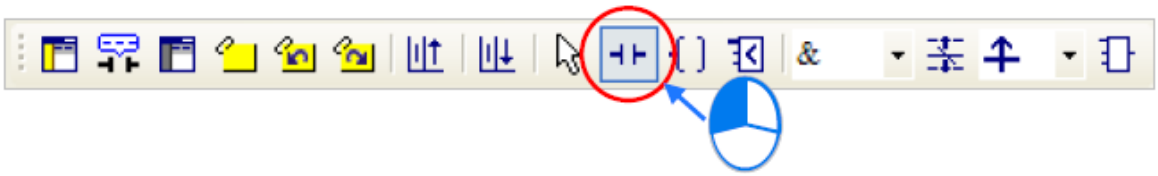
[2] Normally-open contact

[3] Normally-close contact

[4] Rising edge-triggered contact

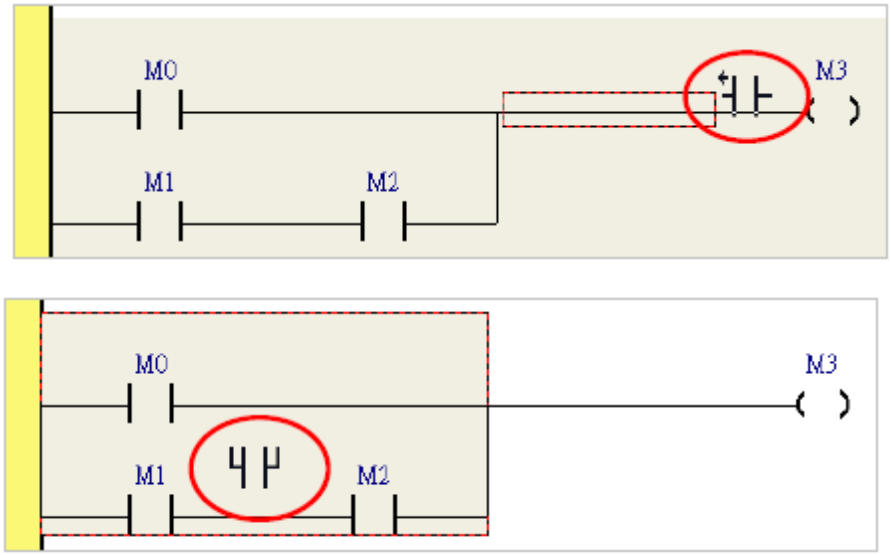
[5] Falling edge-triggered contact

برای اینکه بتوانیم از کانتکت‌ها در برنامه استفاده کنیم، خط مورد نظر در Network را انتخاب و سپس با کلیک بر روی کانکت را به خط انتخاب شده اضافه می‌کنیم.



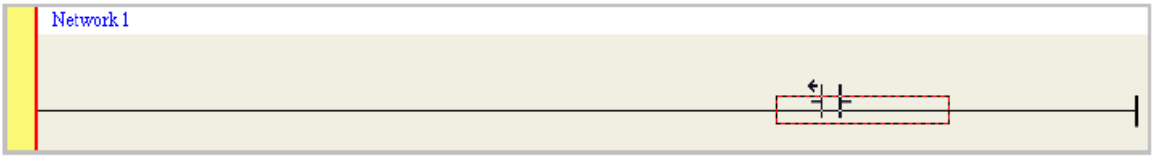
شکل ۸-۱۰ نوار ابزار Ladder- اضافه کردن کانتکت

زمانی که انتخاب شده است با بردن نشانگر موس بر روی Network می‌توان به صورت سری و یا موازی آن را به کانتکت‌های دیگر متصل کرد. در صورتی که نشانگر موس را به خط نزدیک کنیم، ظاهر آن به حالت اتصال سری تغییر می‌کند و اگر نشانگر را به کانتکت‌ها نزدیک کنیم، ظاهر آن به حالت اتصال موازی تغییر می‌کند.



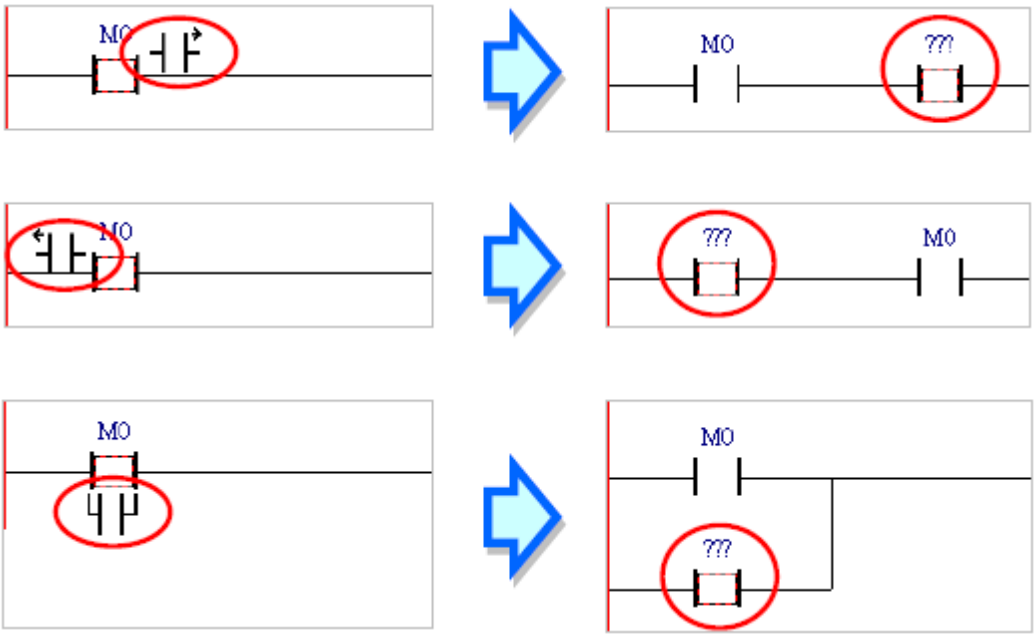
شکل ۸-۱۱ اضافه کردن کانتکت به صورت سری و یا موازی

برای اتصال کانتکت در Network جدید نیز پس از انتخاب Network و فعال کردن باید نشانگر موس را به مستطیل قرمز برد و در آنجا کلیک کرد.



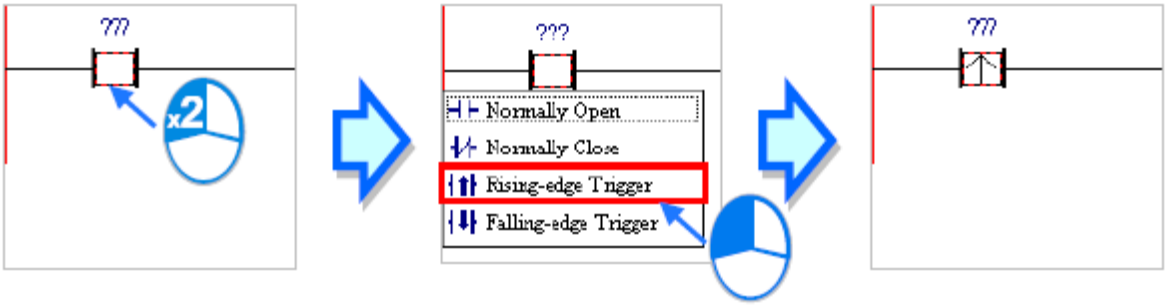
شکل ۸-۱۲ اضافه کردن کانتکت در Network جدید

در شکل زیر نمونه ای از اضافه شدن کانتکت در موقعیت‌های گوناگون در برنامه را مشاهده می‌کنید.



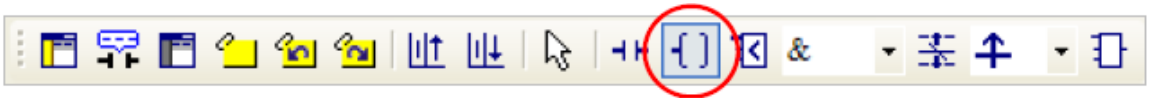
شکل ۸-۱۳ اضافه شدن کانتکت در موقعیت های مختلف در Network

پس از اتصال کانتکت و زدن Esc می‌توان با انتخاب  بر روی هر کانتکت دلخواهی دوبار کلیک و از لیست باز شده نوع آن را انتخاب کرد.



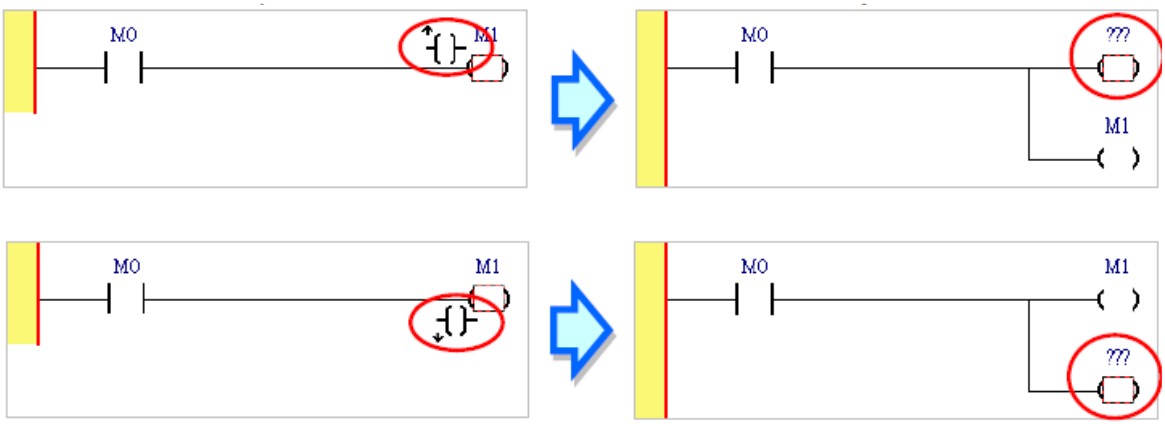
شکل ۸-۱۴ انتخاب نوع کانتکت

برای استفاده از کویل در برنامه پس از انتخاب محل نصب آن در برنامه می‌توان از  استفاده کرد.



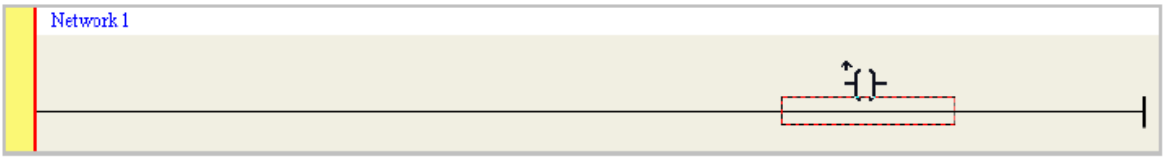
شکل ۸-۱۵ نوار ابزار Ladder- اضافه کردن کویل خروجی

زمانی که [ ] انتخاب شده است با بردن نشانگر موس بر روی Network می‌توان به صورت سری و یا موازی آن را به المان‌های دیگر متصل کرد. در صورتی که نشانگر موس را به خط نزدیک کنیم، حالت آن به حالت اتصال سری تغییر می‌کند (با دیگر کانتکت‌ها سری خواهد شد، باید توجه کرد که کویل‌ها امکان سری شدن با یکدیگر را ندارند) و اگر نشانگر را به المان‌های دیگر نزدیک کنیم، حالت آن به حالت اتصال موازی تغییر می‌کند. در حالت موازی می‌توان کویل را بالا و یا پایین خط موجود فعلی قرار داد.



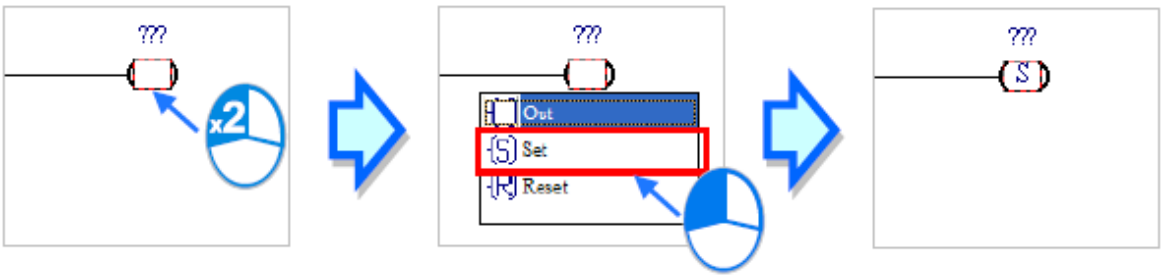
شکل ۸-۱۶ اضافه کردن کویل خروجی به Network

برای اتصال کویل در Network جدید، پس از انتخاب Network و فعال کردن [ ] باید نشانگر موس را به مستطیل قرمز برد و در آنجا کلیک کرد.



شکل ۸-۱۷ اضافه کردن کویل به Network جدید

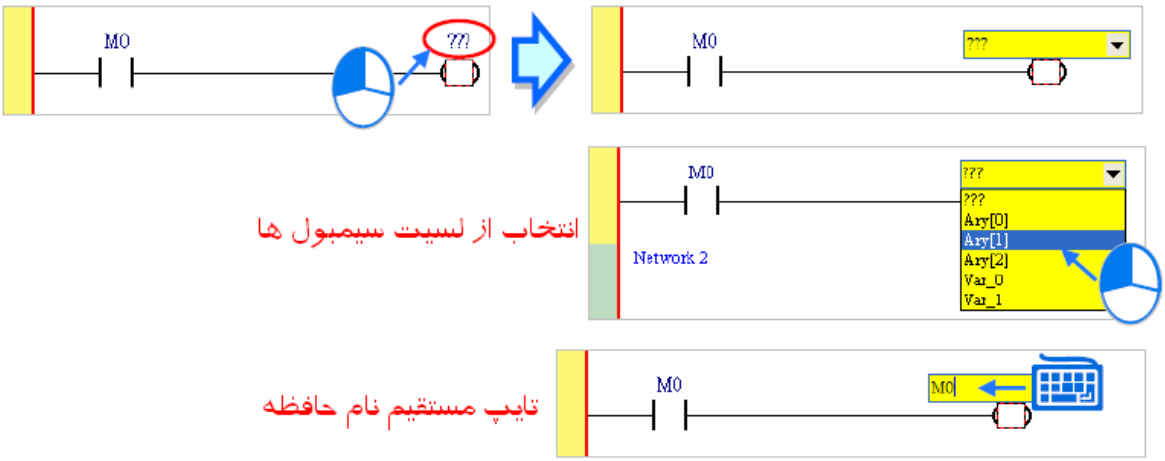
پس از اتصال کویل و زدن Esc می‌توان با انتخاب [ ] بر روی هر کویل دلخواهی دوبار کلیک و از لیست باز شده نوع آن را انتخاب کرد.



شکل ۸-۱۸ انتخاب نوع کویل خروجی

۸-۲-۲- به کار بردن حافظه ها، سیمبول ها و مقادیر ثابت

پس از کلیک بر روی "???" بالای المان‌های روی Network، کاربر می‌تواند با تایپ کردن نام حافظه، سیمبول، پورت و یا عدد ثابت آن را به المان مورد نظر اختصاص دهد و یا اینکه با کلیک بر روی ▼ از لیست ارائه شده سیمبول خود را انتخاب کند.



شکل ۸-۱۹ اختصاص حافظه، سیمبول، پورت و یا عدد ثابت به المان های سیستم

برای ویرایش حافظه و یا پورت های تخصیص داده شده می‌توان دوباره نام آن المان را انتخاب و سپس ویرایش کرد. حین ویرایش نام (منظور حافظه و یا پورت و یا سیمبول اختصاصی به آن است) استفاده از صفحه کلید می‌تواند به سرعت کار کاربر بیافزاید، پس از انتخاب المان در صورتی که بخواهیم سیمبولی برای آن برگزینیم با فشردن Page Down لیست سیمبول‌ها باز می‌شود که با استفاده از جهت بالا و پایین می‌توانیم آن‌ها را انتخاب کنیم. پس از انتخاب سیمبول با فشردن Enter تغییرات المان ثبت و نام المان بعدی برای ویرایش فعال می‌شود. در انتها نیز با فشردن Esc می‌توان از حالت ویرایش خارج شد.

در حالتی که کاربر بخواهد از مقدار ثابت در Ladder استفاده کند، باید این مقدار مطابق با قواعد جدول زیر آماده شود.

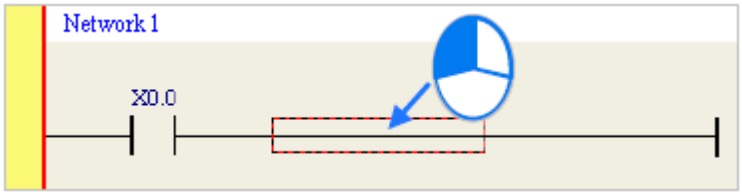
جدول ۸-۳: فرمت مقادیر ثابت در ISPSoft

نوع	شرح

دسیمال[1] (دهدهی= مبنای ده)	مانند 23456 – هر عدد بدون نشانه دسیمال در نظر گرفته می‌شود.
اُکتال[2] (مبنای ۸)	مانند 8#5564 – مقادیر اُکتال با علامت #8 در ابتدایشان مشخص می-شوند.
هگزادسیمال[3] (مبنای ۱۶)	مانند 16#5BAD – مقادیر هگزادسیمال (یا به صورت مختصر هگز) با علامت #16 در ابتدایشان مشخص می‌شوند.
باینری (عدد مبنای دو)	مانند 2#11101010011 – مقادیر باینری با علامت #2 در ابتدایشان مشخص می‌شوند.
String	"XU016S" – کاراکترها در میان علامت نقل قول قرار می‌گیرند.
مقدار بولی (بیتی)	AH500: SM400 (کانتکت باز) یا SM401 (کانتکت بسته) مورد استفاده قرار می‌گیرند. DVP: M1000 (کانتکت باز) یا M1001 (کانتکت بسته) مورد استفاده قرار می‌گیرند.
آرایه	در این حالت حتما باید از سیمبول استفاده کنیم.

۸-۲-۳- نوشتن دستورالعمل

کاربران برای اضافه کردن المان‌ها در ISPSoft می‌توانند به صورت مستقیم دستورالعمل[4] بنویسند. برای اینکار ابتدا مکانی را که می‌خواهید در آن المانی را اضافه کنید را در برنامه انتخاب کنید.



[1] Decimal Value

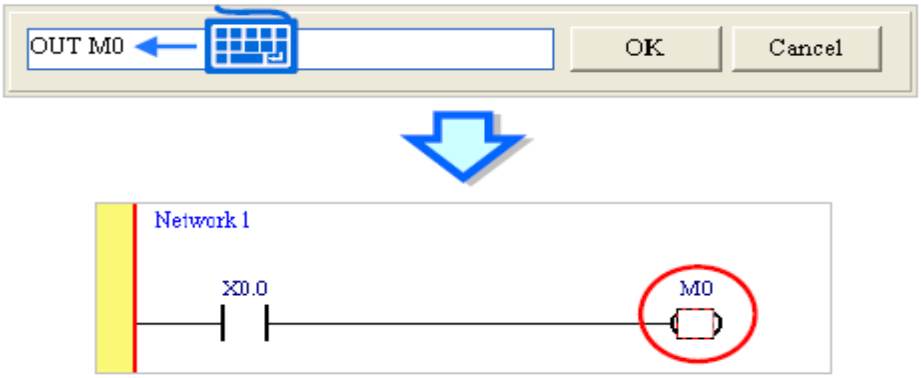
[2] Octal Value

[3] Hexadecimal

[4] Instructions

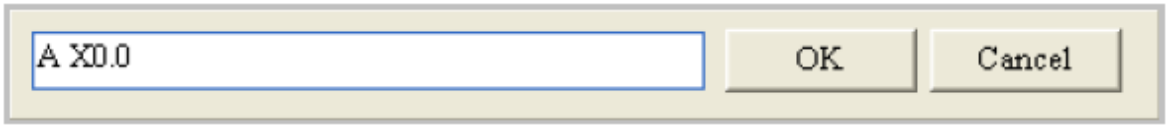
شکل ۸-۲۰ انتخاب خط برای نوشتن دستورالعمل

با صفحه کلید دستورالعمل مورد نظرتان را تایپ کنید (به محض شروع تایپ صفحه مرتبط برای نوشتن دستورالعمل ظاهر می‌شود) و در نهایت آن را تایید کنید. توجه به این نکته ضروری است که دستورالعمل‌ها به بزرگ و کوچک بودن حرف حساس هستند.



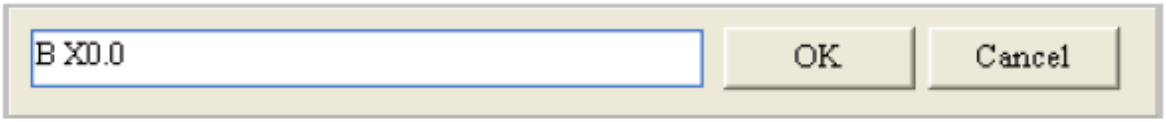
شکل ۸-۲۱ نوشتن دستورالعمل

- دستورالعمل کانتکت باز: A فاصله و سپس آدرس کانتکت



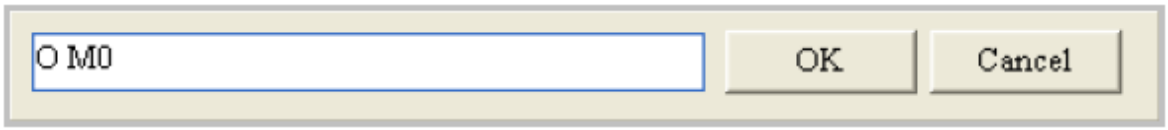
شکل ۸-۲۲ دستورالعمل کانتکت باز

- دستورالعمل کانتکت بسته: B فاصله و سپس آدرس کانتکت



شکل ۸-۲۳ دستورالعمل کانتکت بسته

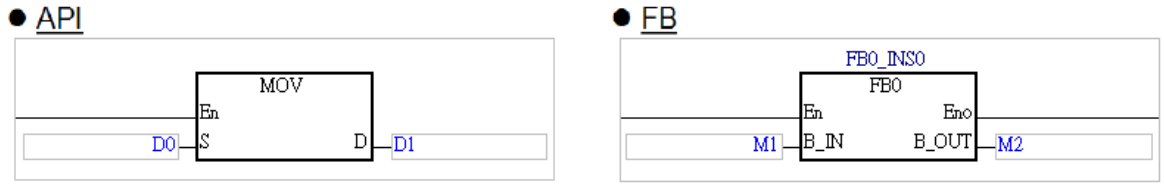
- دستورالعمل کویل خروجی: O فاصله و سپس آدرس کویل



شکل ۸-۲۴ دستورالعمل کویل خروجی

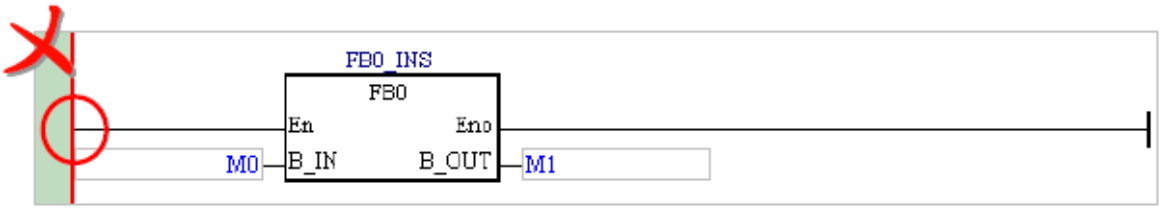
۸-۲-۴- بلوک ها

در ISPSoft دو نوع از توابع از قبل آماده FB و API وجود دارد (FBها توسط کاربر تدارک دیده شده و APIها توسط نرم افزار ISPSoft) که کاربر می‌تواند از آن‌ها در برنامه خود استفاده کند. این توابع در زبان Ladder به صورت بلوک‌هایی هستند که دارای پایه‌های ورودی و خروجی هستند. تمامی این بلوک‌ها شامل پایه ورودی En هستند که در صورتی که فعال نباشد، بلوک اجرا نمی‌شود. در صورتی که بلوک مورد نظر شامل خروجی EnO باشد در صورتی که En فعال باشد EnO هم فعال و در غیر این صورت غیرفعال خواهد بود. بلوک‌هایی که دارای EnO نیستند همانند کویل خروجی باید کویل انتهایی Network باشند و پس از آن‌ها نمی‌تواند بلوک دیگری قرار گیرد.



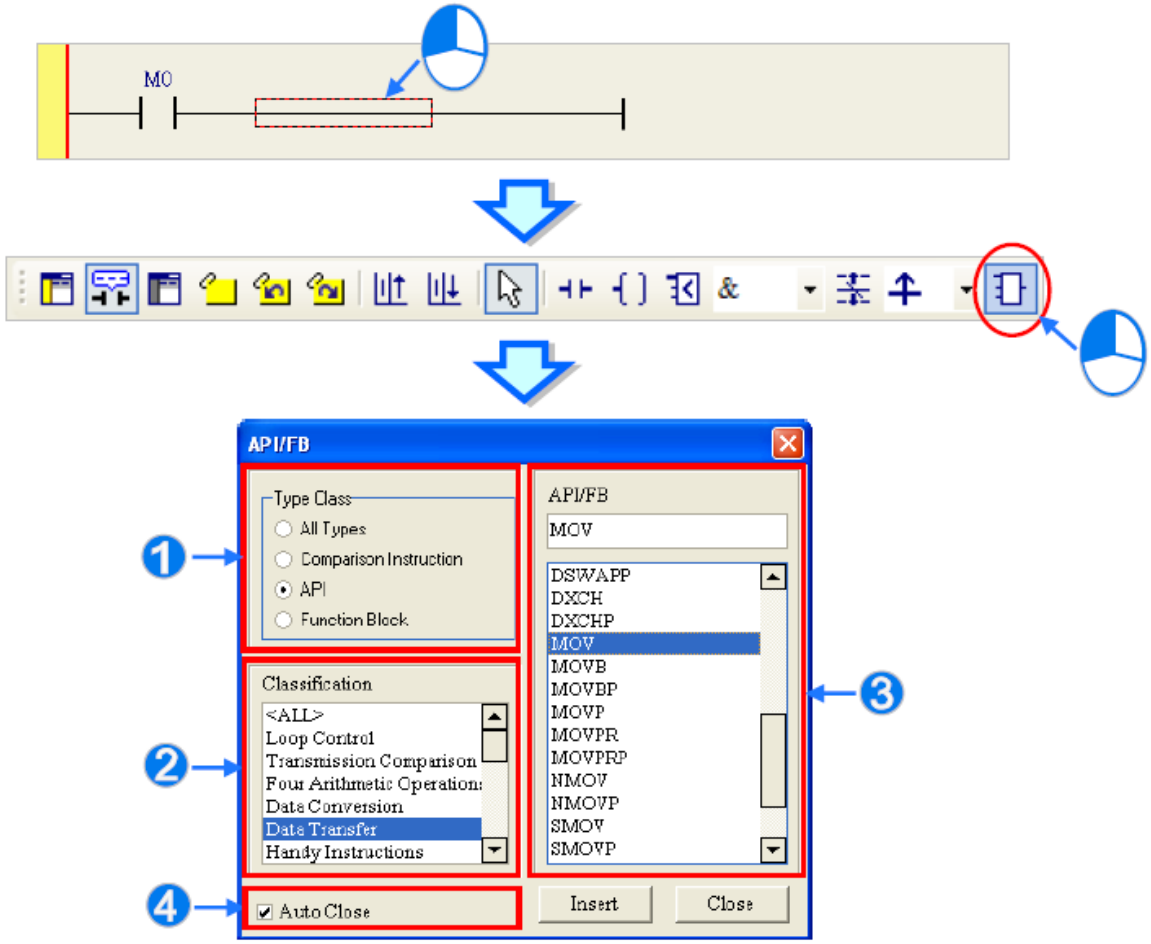
شکل ۸-۲۵ بلوک‌های API و FB در زبان Ladder

توجه شود که حتما قبل از پایه En المان یا بلوک دیگری قرار گیرد، کاربرد شکل زیر نادرست است.



شکل ۸-۲۶ استفاده نادرست از بلوک و اتصال مستقیم خط ورودی به پایه En

برای استفاده از بلوک‌ها در برنامه، پس از انتخاب مکان پیاده سازی آن در برنامه بر روی کلیک کرده و در پنجره ظاهر شده انتخاب کنید نوع بلوک مورد نظرتان API است یا FB.



شکل ۸-۲۷ اضافه کردن بلوک به برنامه

1 انتخاب نوع بلوک: All Type به معنی همه نوع بلوک، Comparison Instruction به معنی بلوک‌های مقایسه‌ای، API بلوک‌های آماده ISPSoft و Function Block بلوک‌های آماده شده توسط کاربر

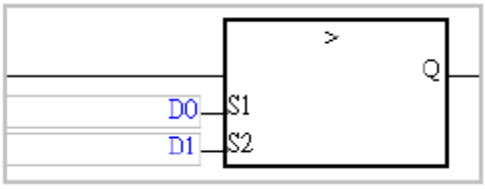
2 با انتخاب API در این قسمت لیست خانواده بلوک‌های مختلف ظاهر می‌شود.

3 انتخاب نوع بلوک

4 با غیرفعال کردن گزینه Auto Close می‌توان به صورت متوالی بلوک به برنامه اضافه کرد.

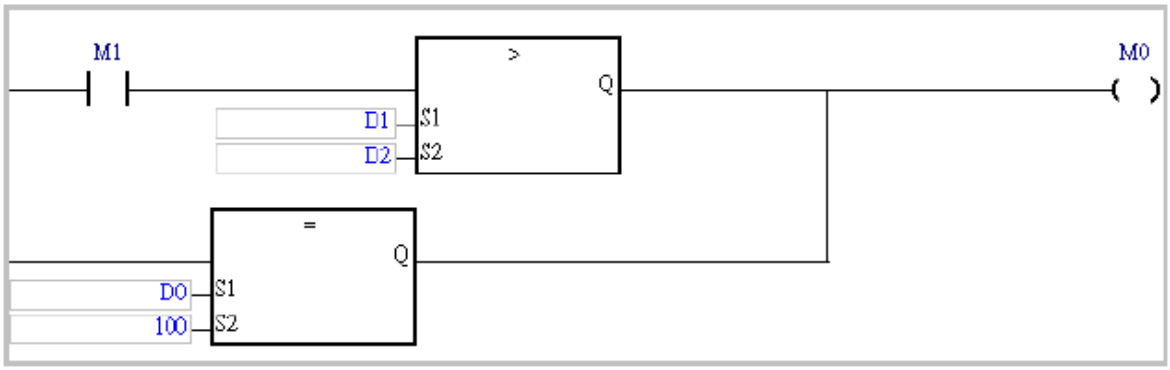
۸-۲-۵- بلوک های مقایسه ای

بلوک‌های مقایسه کننده به بررسی نسبت دو عدد پرداخته و نتیجه را به صورتی بیتی به خروجی می-دهند. مثلا بلوک زیر مقدار دو ورودی D0 و D1 را مقایسه و اگر D0>D1 بود یعنی S1 بزرگتر از S2 بود، خروجی بلوک یعنی Q یک می‌شود (توجه شود برای داده‌ها با فرمت‌های گوناگون بلوک‌های متفاوتی وجود دارد).



شکل ۸-۲۸ بلوک مقایسه ای

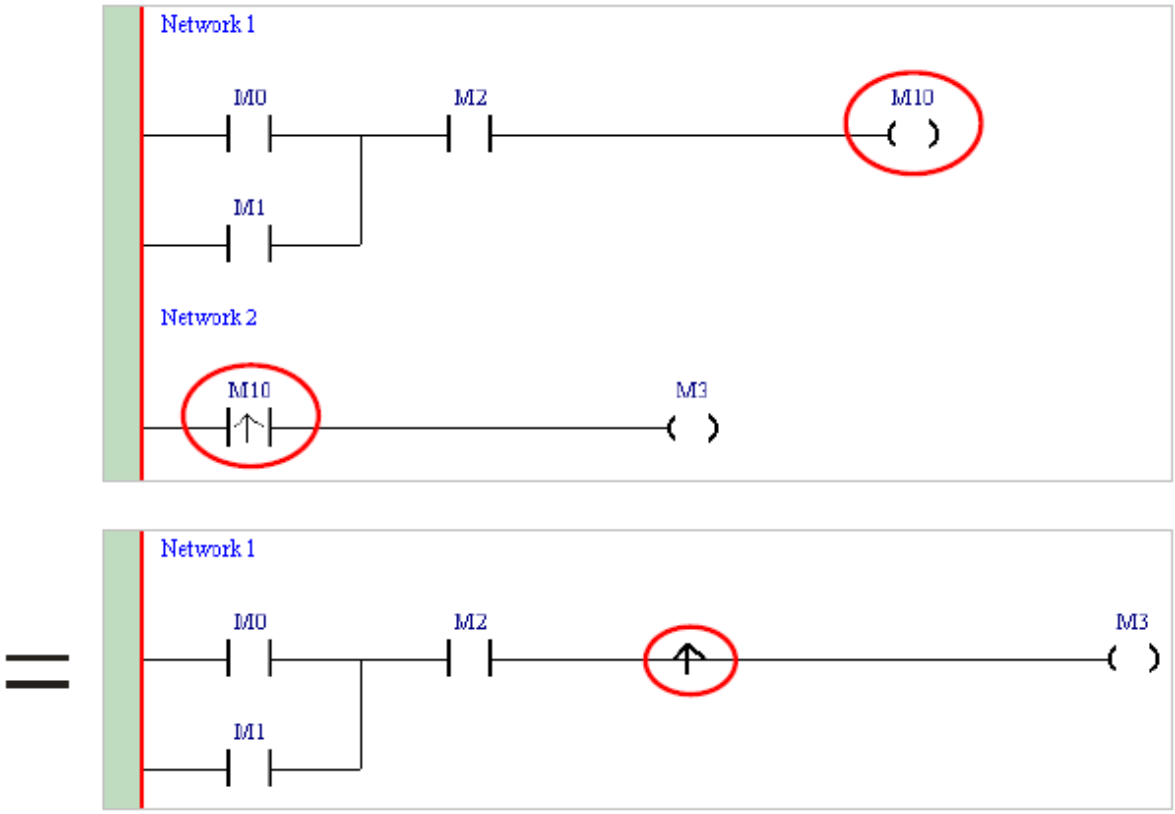
بلوک‌های مقایسه‌ای همانند کانتکت‌ها قابلیت اتصال موازی و سری را دارند.



شکل ۸-۲۹ اتصال بلوک‌های مقایسه ای

۸-۲-۶- بلوک های منطقی

عملیات‌های منطقی یکی از کاربردی‌ترین ابزارهای کار هر برنامه‌نویس می‌باشد. بسیاری از عملکردهای مورد انتظار کاربران مخصوصا دستورات بیتی با استفاده عملیات‌های منطقی به صورت بهینه-تری قابل پیاده سازی است. به عنوان مثال در شکل زیر عملیات لبه گیری در دو Network انجام می-شود که می‌توان با استفاده از عملیات بیتی ↑ آن را در یک Network انجام داد.



شکل ۸-۳۰ عملیات بیتی لبه گیری

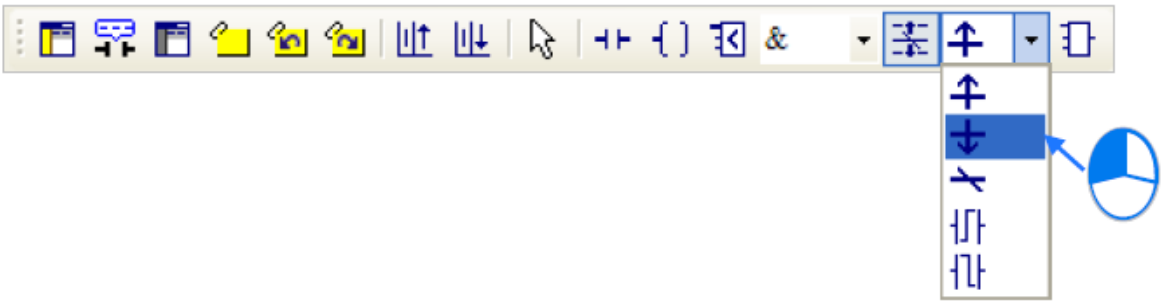
عملیات‌های بیتی در ISPSoft مطابق جدول زیر هستند.

جدول ۸-۴: بلوک های منطقی

نوع	دستورالعمل	شرح عملکرد
	NP	هرگاه خط متصل به آن از صفر به یک تغییر کند (لبه بالا رونده رخ دهد) ، خروجی آن برای مدت یک سیکل زمانی یک می‌شود.
	PN	هرگاه خط متصل به آن از یک به صفر تغییر کند (لبه پایین رونده رخ دهد) ، خروجی آن برای مدت یک سیکل زمانی یک می‌شود.
	INV	هرچه در خط ورودی آن باشد را معکوس می‌کند.
	FB_NP	مشابه NP با این تفاوت که کاربر باید حافظه‌ای را به آن اختصاص دهد. (فقط در AH500)

???	FB_PN	مشابه PN با این تفاوت که کاربر باید حافظه‌ای را به آن اختصاص دهد. (فقط در AH500)

برای اینکه بتوانیم از عملیات بیتی در برنامه خود استفاده کنیم ابتدا باید در نوار ابزار نوع عملیات را در  مشخص کنیم.

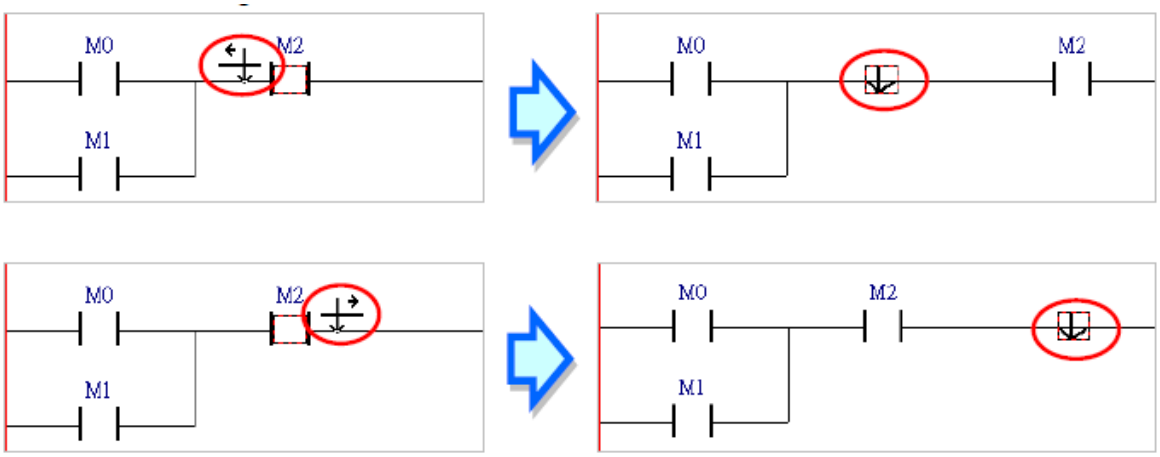


شکل ۸-۳۱ انتخاب نوع عملیات بیتی در نوار ابزار

سپس با کلیک بر روی  می‌توان عملیات بیتی را برای استفاده در برنامه انتخاب کرد.

شکل ۸-۳۲ اضافه کردن عملیات بیتی به Network – قسمت اول

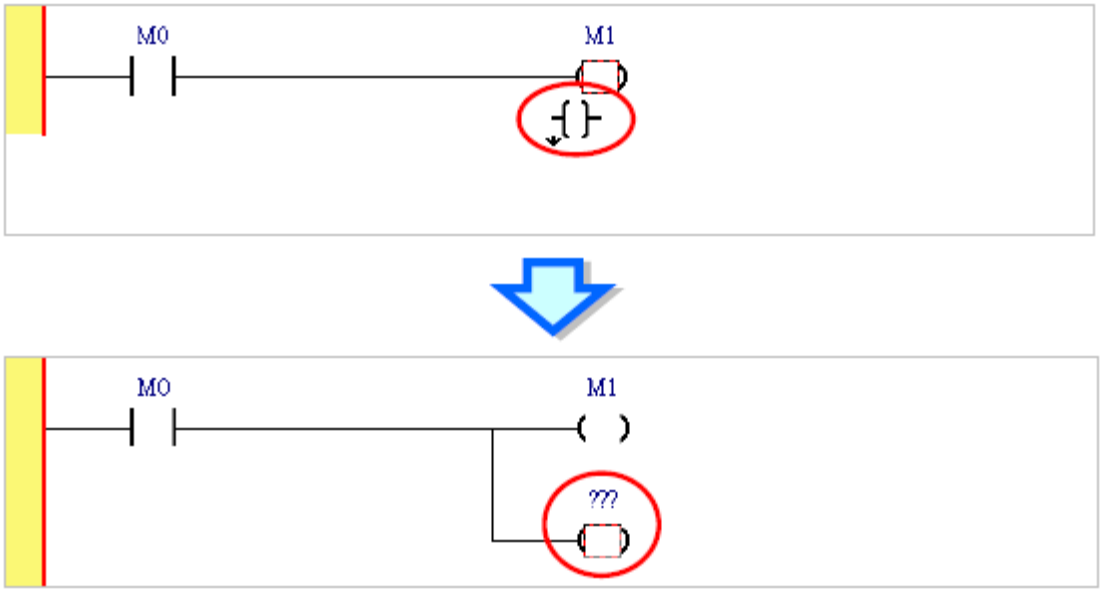
سپس نشانگر موس را به خطی که عملیات بیتی را می‌خواهیم درآنجا قرار دهیم برده و زمانی که نشانگر موس به شکل عملیات بیتی در آمد کلیک می‌کنیم تا به خط اضافه شود.



شکل ۸-۳۳ اضافه کردن عملیات بیتی به Network – قسمت دوم

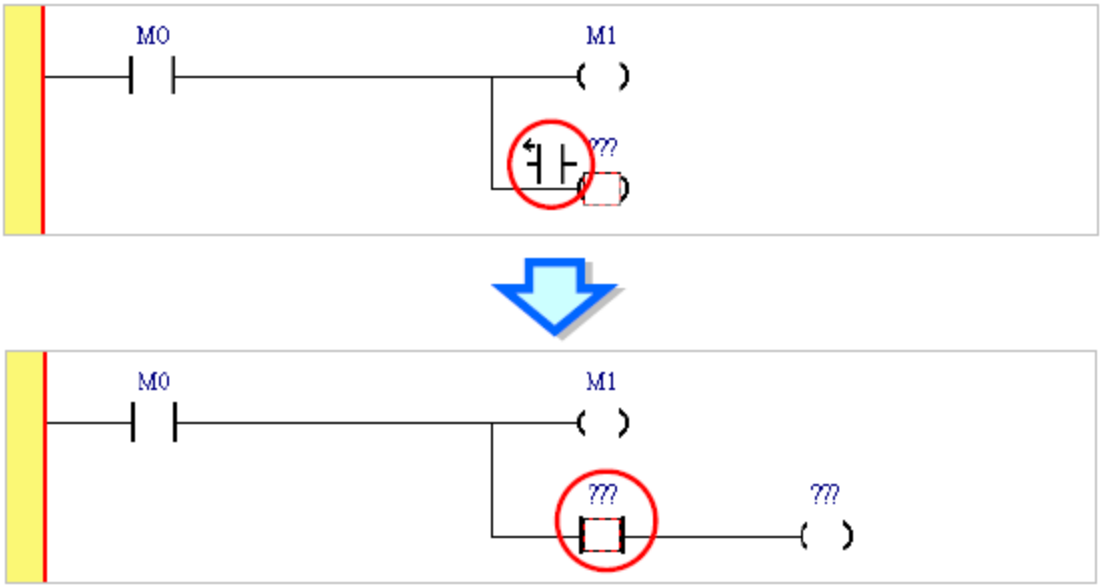
۸-۲-۷- ساخت چند خروجی در Network

برای اینکه بتوان چند خروجی را به صورت همزمان در یک Network داشته باشیم، ابتدا دو کویل خروجی را به صورت موازی با هم در مدار قرار می‌دهیم.



شکل ۸-۳۴ ایجاد دو کویل خروجی موازی در یک Network

سپس در خط هر کدام می‌توانیم المان‌های خاصی را به صورت منحصر به فرد قرار دهیم.



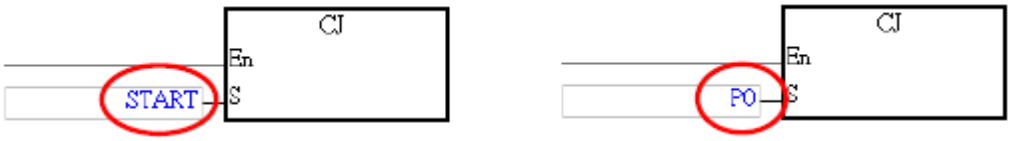
شکل ۸-۳۵ امکان برنامه نویسی منحصر به فرد هرکدام از کویل های خروجی موازی

۸-۲-۸- اضافه کردن برچسب

گاهی اوقات نیاز است در زمان اجرای برنامه به قسمت خاصی از برنامه برای اجرا برویم و یا به عبارتی پرش[۱] داشته باشیم. برای اینکار لازم است برچسبی به قسمت‌هایی از برنامه که می‌خواهیم به آن‌ها پرش داشته باشیم، اختصاص دهیم. این برچسب‌ها باید همراه با اختصاص حافظه نوع P در پردازنده‌های DVP باشند (اختصاص حافظه نوع P به برچسب‌ها در AH500 به صورت اتوماتیک انجام می‌پذیرد).

شکل ۸-۳۶ اضافه کردن برچسب

برای پرش به برچسب مورد نظر نیز می‌توانیم به بلوک CJ آدرس و یا نام برچسب را (بدون علامت دو نقطه":") دهیم.



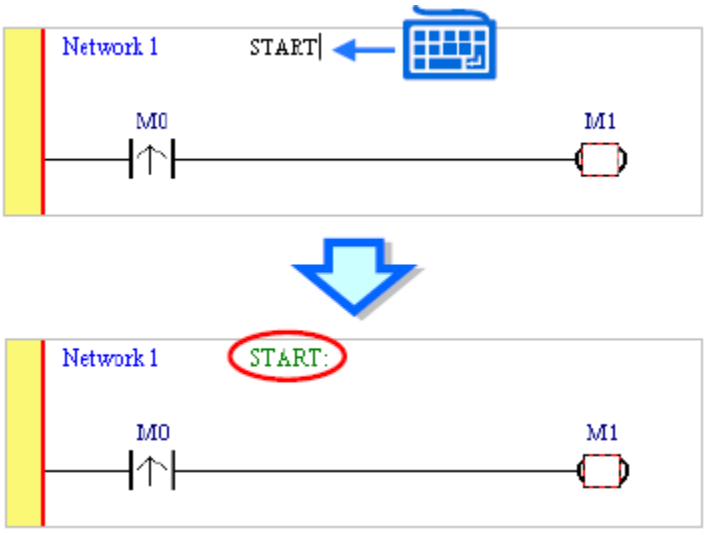
شکل ۸-۳۷ پرش به برچسب با بلوک CJ

برای ایجاد برچسب ابتدا باید سمت راست شماره Network کلیک کرد.

شکل ۸-۳۸ ایجاد برچسب- قسمت اول

[۱] Jump

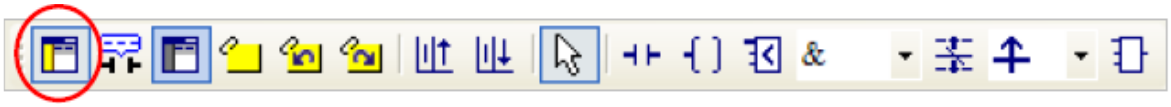
سپس در صورتی که پردازنده نوع DVP باشد کاربر باید حافظه نوع P مورد نظر خود را در این محل تایپ و اگر پردازنده نوع AH500 باشد، کاربر باید نام برچسب را تایپ کند و Enter را بفشارد که در نهایت در جلوی Network آن برچسب به همراه علامت دو نقطه ظاهر شود.



شکل ۸-۳۹ ایجاد برچسب- قسمت دوم

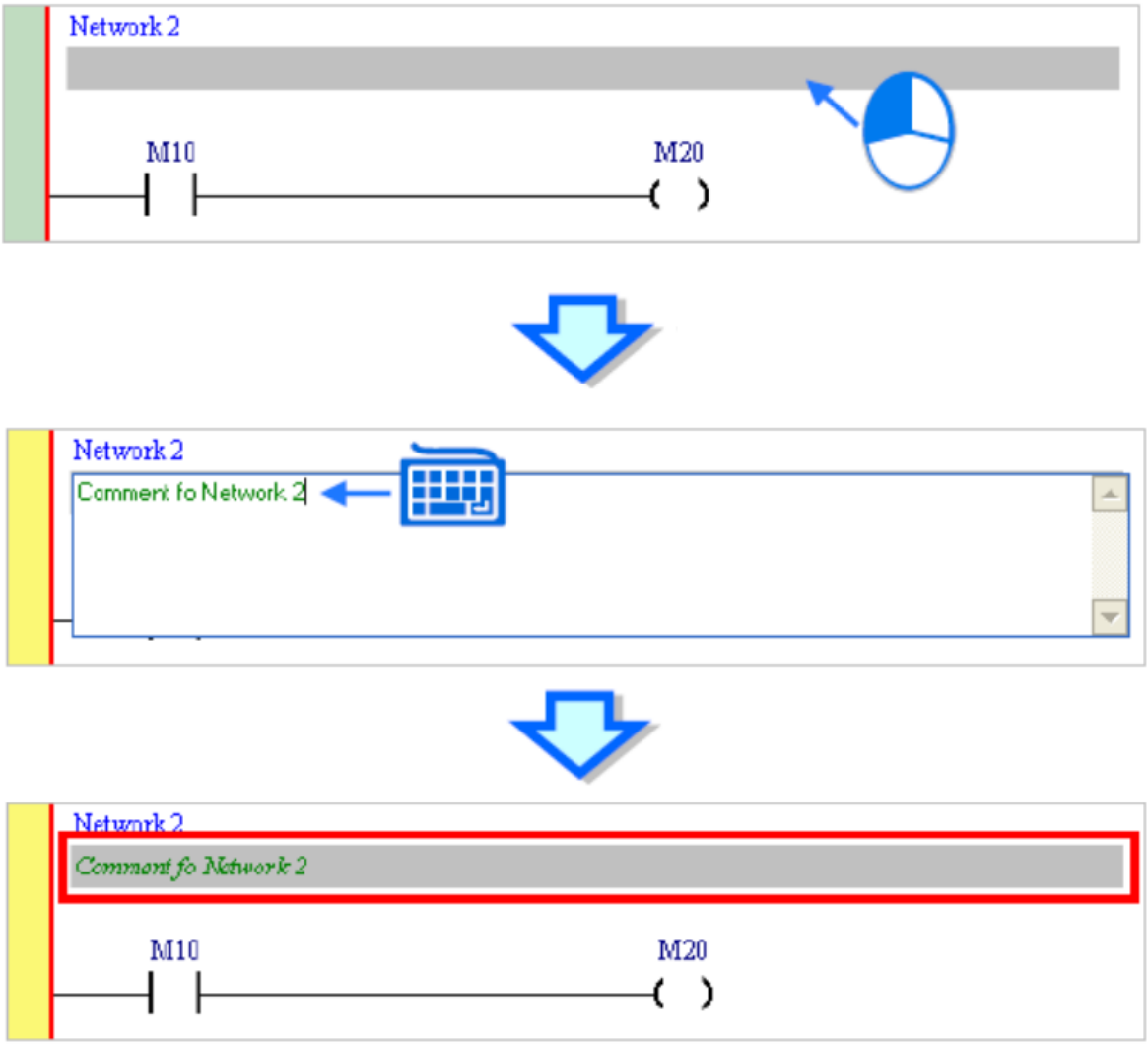
۸-۲-۹- ویرایش توضیحات

برای اینکه بتوانیم **توضیحات برنامه (در Networkها)** را دید می‌توان بر روی  کلیک کرد. اگر دیدن توضیحات محیط برنامه را شلوغ می‌کند و یا اینکه ضروری نیست می‌توان با فشردن دوباره  آن‌ها را مخفی کرد.



شکل ۸-۴۰ نمایش/مخفی کردن توضیحات

در حالت نمایش توضیحات با کلیک بر روی بخش خاکستری زیر نام Network می‌توان توضیحات مورد نیاز آن Network را تایپ کرده و با زدن Enter آن را ثبت کرد (در صورتی که بخواهیم در توضیح نویسی به خط جدید برویم می‌توانیم از Shift+Enter استفاده کنیم).

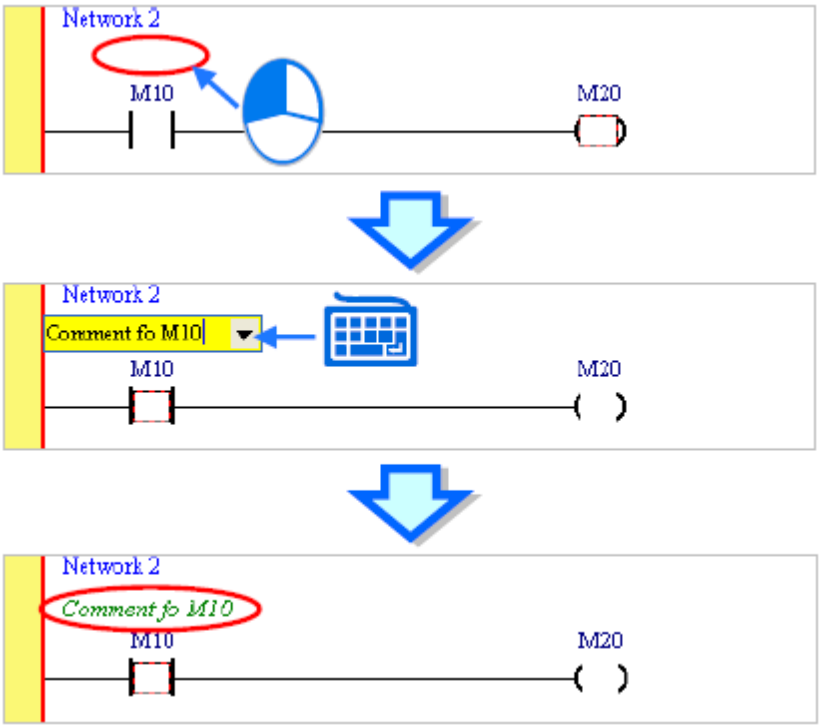


شکل ۸-۴۱ توضیح نویسی برای Network

برای اینکه بتوان **توضیحات المان‌های** برنامه را دید، می‌توان بر روی  کلیک کرد. در صورتی که نیاز به مخفی شدن این توضیحات باشد نیاز است دوباره بر روی  کلیک کرد.

شکل ۸-۴۲ نمایش/ مخفی شدن توضیحات المان ها

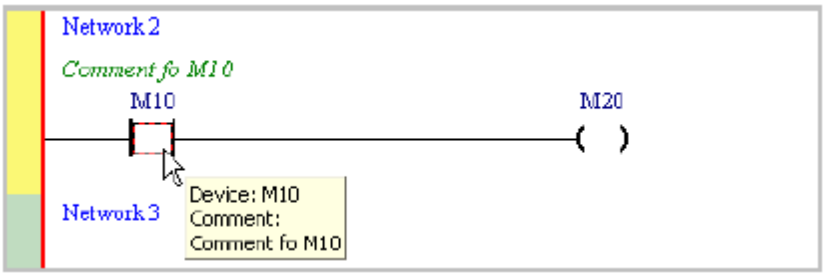
پس از فعال شدن حالت نمایش توضیحات المان‌ها، کاربر می‌تواند با کلیک بر روی بالای المان مورد نظر توضیحات را اضافه و با زدن Enter آن‌ها را ثبت کند.



شکل ۸-۴۳ توضیح نویسی برای المان ها

در صورتی که به المانی سیمبول اختصاص داده شده باشد، با کلیک بر روی  توضیحات مربوط به سیمبول نمایش داده می‌شود. توضیحات مربوط به سیمبول را نمی‌توان به صورت مستقیم تغییر داد و باید آن را از طریق صفحه ویرایش سیمبول‌ها اصلاح کرد.

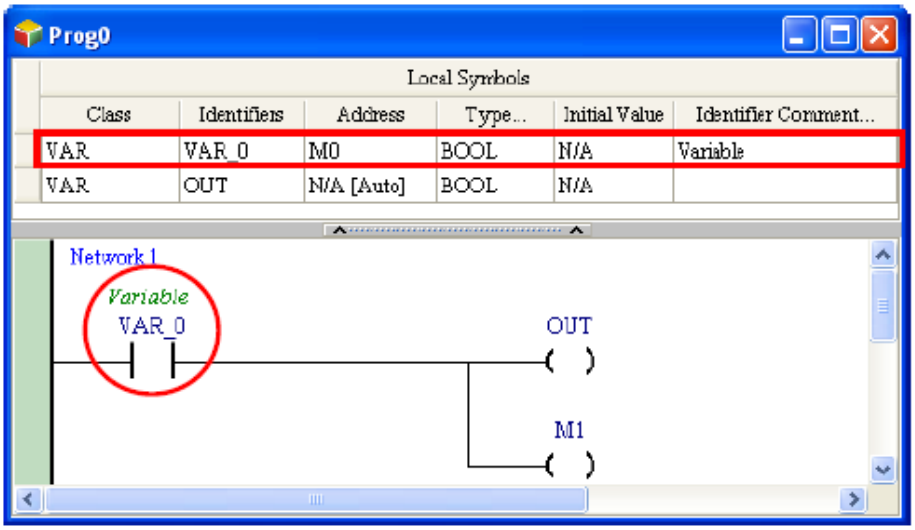
در صورتی که حالت نمایش توضیحات المان‌ها فعال باشد، کاربر می‌تواند با بردن نشانگر موس بر روی المان مورد نظر توضیحات مربوط به آن را ببیند (اینکه چه نوع توضیحاتی نمایش داده شود را می‌توان در بخش Option تنظیم کرد).



شکل ۸-۴۴ نمایش توضیحات المان ها با بردن نشانگر موس بر روی آن ها

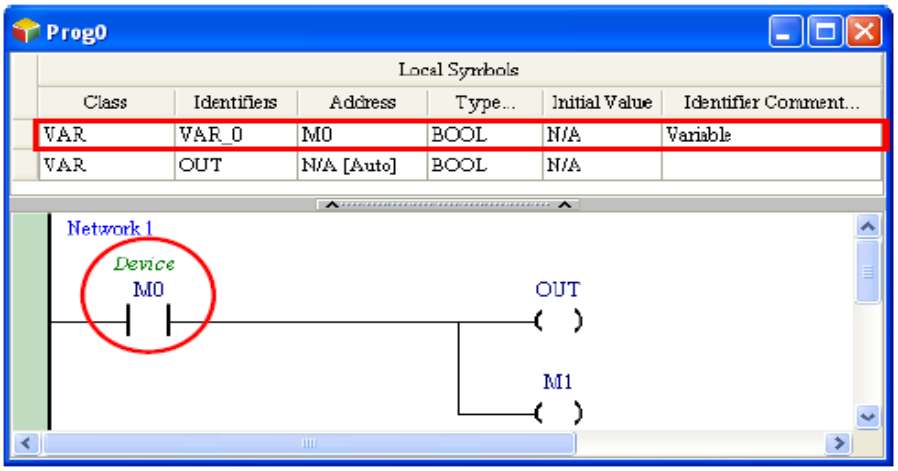
۸-۲-۱۰- حالت نمایش سیمبول / آدرس

در ISPSoft می‌توان تنظیمات صفحه نمایش برنامه را به صورتی تغییر داد که بتوان سیمبول‌ها را به جای آدرس المان‌ها نمایش داد و یا بالعکس. برای اینکه سیمبول‌ها به همراه توضیحات مربوط به هر سیمبول نمایش داده شود باید ADDR انتخاب نشده باشد.



شکل ۸-۴۵ نمایش سیمبول ها در برنامه

در صورتی که گزینه ADDR انتخاب شده باشد آدرس حافظه‌ها و یا پورت‌ها در صفحه برنامه نمایش داده می‌شوند.

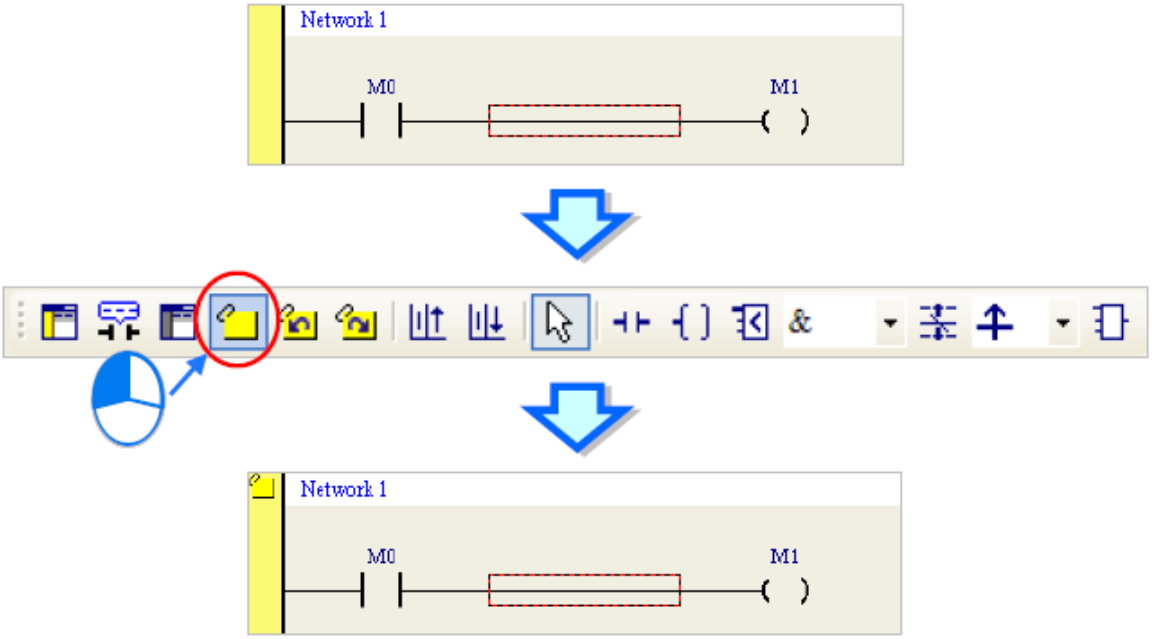


شکل ۸-۴۶ نمایش حافظه و یا پورت ها در برنامه

۸-۲-۱۱- نشانه گزاری

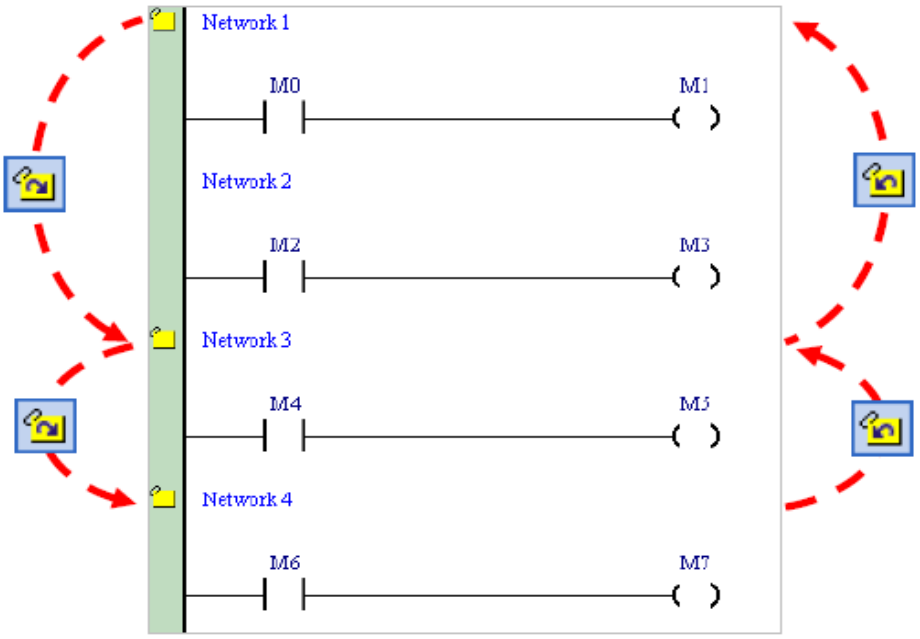
با استفاده از نشانه گزاری برنامه، می‌توان برنامه نویسی را به همراه برجسته کردن توضیحات و بخش-های مهم برنامه انجام داد. نشانه گزاری همچنین باعث ایجاد نظم و جستجوی راحت تر در بین Networkها می‌شود.

برای اضافه کردن نشان به Network، پس از انتخاب کردن آن باید بر روی  کلیک کرد. برای حذف نشان بر روی Network می‌توان یکبار دیگر بر روی  کلیک کرد. (همچنین حذف و اضافه کردن نشان را می‌توان با استفاده از کلیدهای میانبر Shift+Ctrl+B انجام داد).



شکل ۸-۴۷ اضافه کردن نشان

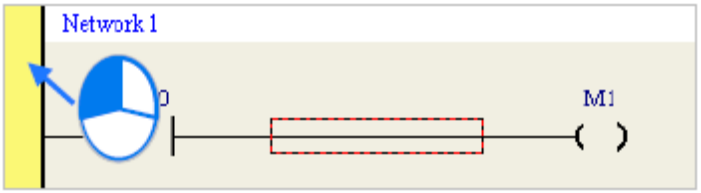
پس از نشان گزاری با استفاده از  و  می‌توان بین نشان‌ها جابجا شد.



شکل ۸-۴۸ جابجایی بین نشان ها

۸-۲-۱۲- فعال و غیرفعال کردن Network

گاهی اوقات لازم است بعضی از Networkهای برنامه در مرحله تست کامپایل نشوند و برای اینکار لازم است تا آن‌ها را غیرفعال کرد. برای اینکار ابتدا Network مورد نظر را انتخاب می کنیم.



شکل ۸-۴۹ انتخاب Network

سپس با کلیک بر روی  می‌توانیم آن را فعال و یا غیرفعال کنیم.

شکل ۸-۵۰ فعال / غیر فعال کردن Network

پس از طی مراحل تست برای اینکه کاربر بتواند تمامی Networkهای غیرفعال را فعال کند، می‌تواند از سربرگ Edit گزینه Activate All Networks را فعال کند.

فصل ۹– زبان برنامه نویسی Structured Text

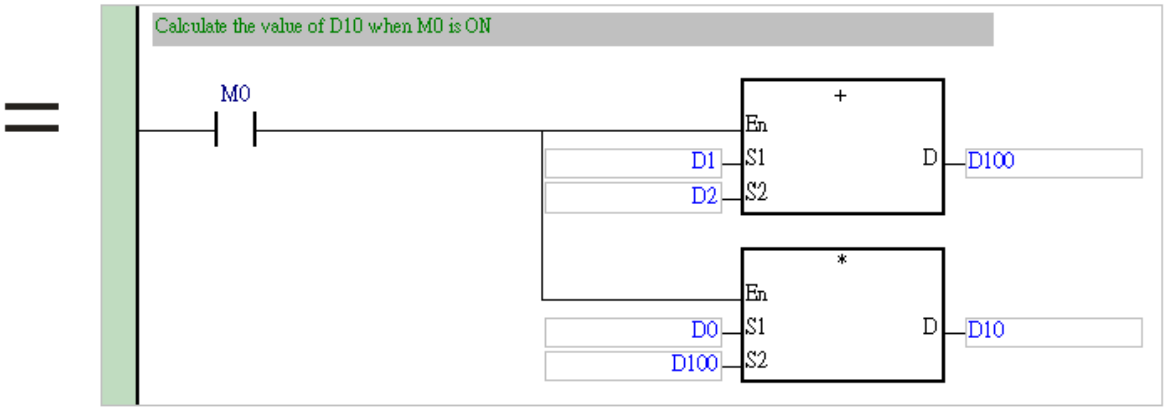
زبان سطح بالای Structured Text یا به صورت مختصر ST دارای ساختاری مشابه زبان برنامه نویسی C و Pascal است. دستورات نوشته شده در ST به نسبت زبان Instruction List قابل فهم‌تر و گسترش آن راحت‌تر است. این زبان بسیار منعطف و بهترین گزینه برای افرادی است که با برنامه نویسی آشنایی دارند. این زبان محیطی مشابه زبان برنامه نویسی C برای کاربران فراهم می‌آورد و این موضوع برای افرادی که با این زبان آشنایی دارند بسیار جذاب است (به خصوص مهندسین برق و کامپیوتر و مکانیک).

*** توجه: با آنکه در مستندات رسمی کمپانی دلتا تا زمان انتشار این کتاب بحثی پیرامون سازگاری زبان‌های برنامه نویسی با PLCهای مختلف مطرح نشده است ولی به نظر می‌رسد زبان برنامه نویسی ST و FBD تنها در PLCهای سری AH500 قابل استفاده هستند. این موضوع در فروم‌های اینترنتی کاربران دلتا نیز مطرح شده است.

۹–۱–۱– ساختار Structured Text

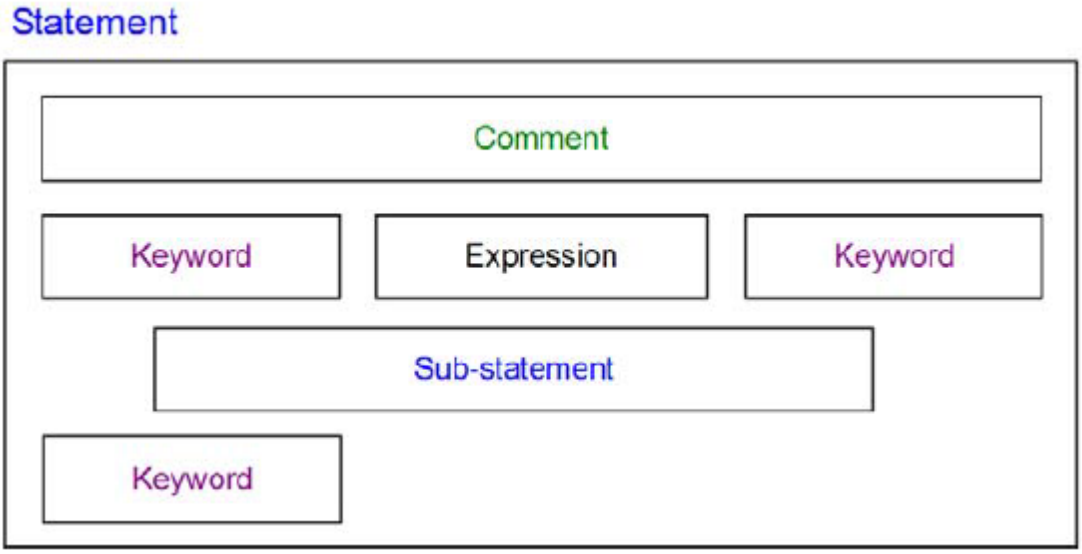
برنامه نوشته شده به زبان ST در مثال زیر معادل برنامه Ladder زیر آن است. در هر دو برنامه در صورتی که بیت M0 یک باشد، مقدار D10 برابر ضرب D0 در جمع D1وD2 خواهد شد.

```
0001 (*Calculate the value of D10 when M0 is ON*)
0002 IF M0 THEN
0003    D10 := D0*(D1+D2) ;
0004 END_IF ;
0005
```



شکل ۹-۱ برنامه‌ای به زبان ST و برنامه Ladder معادل آن

ساختار برنامه ST فوق در زیر آمده است:

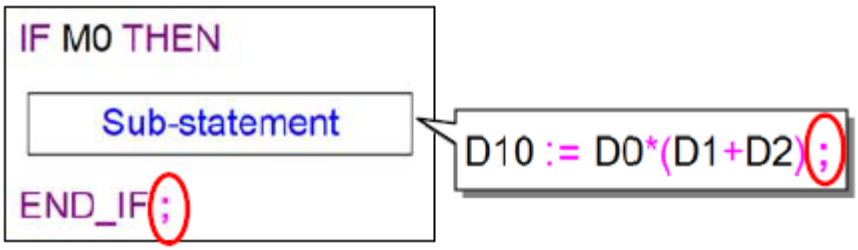


شکل ۹-۲ ساختار برنامه ST

در این بخش بر خلاف رویکرد کلی کتاب مبنی بر ترجمه دقیق محتوا و عبارات غیرفارسی، به علت رایج بودن استفاده از کلمات انگلیسی در میان برنامه نویسان سطح بالا و دوری از ایجاد ابهام از ترجمه اصطلاحات عمومی پرهیز می‌کنیم.

۹-۱-۲- Statement

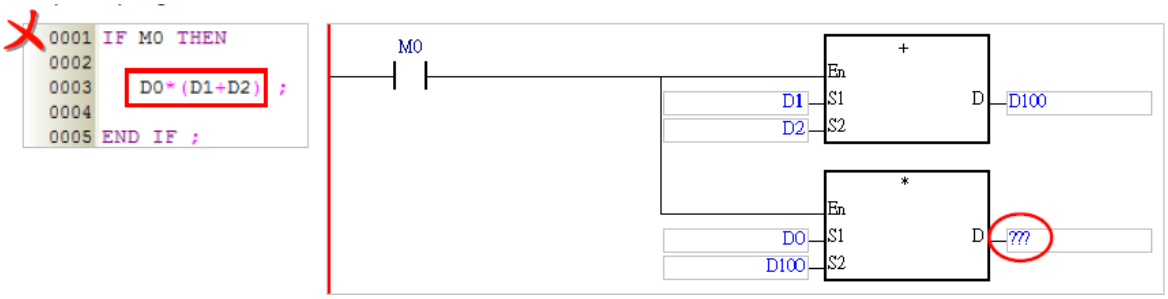
Statement واحدی اولیه برای برنامه نویسی است که عملکرد کاملی را برای اجرا مشخص می‌کند. این واحد کامل ممکن است بیش از یک خط داشته باشد ولی همیشه با نقطه ویرگول ";" به پایان می-رسد.



شکل ۹-۳ نمونه ای از یک Statement که یک Statement در خود دارد

یک Statement صحیح معادل یک Network در برنامه Ladder است. برای مثال Sub-Statement در مثال فوق مقدار D0*(D1+D2) را محاسبه و مقدار آن را در D10 ذخیره می‌کند. اما در Statement زیر که با ضربدر مشخص شده است به علت آنکه محل ذخیره سازی عبارت D0*(D1+D2) مشخص

نشده، دارای معادل Ladder نمی‌باشد. در نتیجه این عبارت بدون المان خروجی صحیح و قابل اجرا نمی-باشد.



شکل ۹-۴ یک عبارت نادرست در ST

ST از تعدادی Statement تشکیل شده است و هر Statement نیز خود شامل چند المان است. مثالی از ST در زیر آمده است، هر Statement کامل در کادر قرمز مشخص شده است.

```
0001
0002 IF M0 THEN
0003     M2 := TRUE ;
0004 ELSIF M1 THEN
0005     M2 := FALSE ;
0006 END_IF ;
0007
0008 M20 := M10 ;
0009
0010 M21 := M11 ;
0011
0012 IF M2 THEN
0013     ROR(D0,2);
0014 END_IF ;
0015
```

شکل ۹-۵ مثالی از Statement ها در زبان برنامه نویسی ST

۹-۱-۳- Expression

Expression از المان‌های مهم Statement به حساب می‌آید و معرف مقداری مشخص است. Expression می‌تواند عبارتی جبری، مقداری ثابت، سیمبول و یا حافظه باشد. چند مثال از Experession را در زیر می‌بینید.

- M0 & M1 (مقدار بولی)

عملگر منطقی & مقدار دو بیت M0 و M1 را گرفته و با هم AND منطقی می‌کند. نتیجه این عملیات مقداری بولی است.

- M0=FALSE (مقدار بولی)

این یک عبارت شرطی است که اگر مقدار M0 برابر ON باشد چون عبارت ON=FALSE عبارتی نادرست است پس مقدار آن صفر خواهد شد. در صورتی که M0 دارای وضعیت OFF باشد، چون OFF=FALSE صحیح است پس این عبارت دارای مقدار یک می‌باشد.

- M0 (مقدار بولی)

مقدار بولی مطابق با مقدار ذخیره شده در M0 است.

- D1+D2

محاسبه مقدار عبارت D1+D2

- D0

مقداری برابر D0 (در صورتی که در آن مقداری ذخیره شده باشد)

- D2=D0+D1 (مقدار بیتی)

در صورتی که مقدار عبارت D0+D1 برابر D2 باشد این عبارت شرطی صحیح خواهد بود و مقدار بولی یک است. در صورتی که D0+D1 برابر D2 نباشد این عبارت نادرست و برابر صفر خواهد بود.

- D2 :=D0+D1

این عبارت شامل دو Expression است، D2 و همچنین D0+D1. مقدار D0+D1 محاسبه و سپس در D2 ذخیره می‌شود (پس همینطور که تا به اینجای کار متوجه شده اید، علامت مساوی "="، معادل دستور شرطی بررسی تساوی طرفین آن است در صورتی که علامت دونقطه مساوی ":=" بیانگر تخصیص Experession سمت راست آن به Experession سمت چپ آن است.

مثال: برنامه زیر از دو IF تشکیل شده است. زمانی که اولین IF اجرا می‌شود، مقدار M0 ارزیابی می-شود، اگر M0 یک بود، مقدار عبار D21+D22 محاسبه می‌شود و در D20 ذخیره می‌شود. سپس در زمان IF دوم در صورتی که عبارت D0*(D1+D2) برابر D20 بود مقدار بولی این عبارت شرطی یک شده و محتوای IF اجرا می‌شود یعنی M1 برابر یک می‌شود، در غیر این صورت M1 برابر صفر می‌شود.

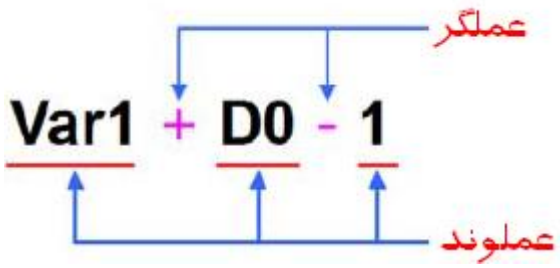
```
0001 IF M0 THEN
0002     D20 := D21 + D22 ;
0003 END_IF;
0004
0005 IF D20 = D0*(D1+D2) THEN
0006     M1 := TRUE ;
0007 ELSE
0008     M1 := FALSE ;
0009 END_IF;
```

شکل ۹-۶ مثال بخش Expression

۹-۱-۴- عملوند و عملگرها

عملوند[۱] و عملگرها[۲] از المان‌های پایه‌ای expressionها به حساب می‌آیند. هر عملیاتی دارای تعدادی مفعول یا عملگر است که عملیات بر روی آن‌ها انجام می‌شود. همچنین نوع عملیات توسط عملگرها مشخص می‌شود. برای مثال در عبارت D0+D1، D0 و D1 عملوند هستند که در این عملیات عملگر جمع یا "+" بر روی آن‌ها اعمال می‌شود.

هر expression می‌تواند شامل چند عملوند و عملگر باشد، عملوندها نیز می‌توانند به صورت حافظه، سیمبول و یا مقدار ثابت باشند.



شکل ۹-۷ عملوندها و عملگرها در Expression

اینکه چگونه عملگرها بر عملوندها اعمال می‌شوند تابع مفهوم اولویت است که طبق آن اول عملگرهای با اولویت بالا اجرا و سپس عملگرهای با اولویت پایین اجرا می‌شوند و زمانی که عملگرهای یکسانی داریم اولویت از چپ به راست است. در جدول زیر عملگرها، نحوه کار و اولویت آن‌ها را می‌بینید. توجه کنید که

---

[۱] Operand

[۲] Operator

در محاسبات جدول، مقدار پیش فرض M0 برابر صفر و مقدار پیش فرض M1 برابر یک و مقدار D0 برابر 5 است.

جدول ۹-۱: آیکون های نوار ابزار برنامه نویسی Ladder

سیمبول	عملکرد	فرمت داده		مثال		اولویت
		عملگر	نتیجه عملیات (مقدار (expression	Expression	مقدار	
()	قسمت‌های داخل پرانتز ابتدا محاسبه می‌شوند.	عمومی	عمومی	(D0+6)*3	۳۳	بالاترین
**	توان[1]	ممیز شناور	ممیز شناور	2.0**5.0	3.2E+1	
-	علامت منفی	اعداد علامت‌دار	اعداد علامت-دار	-(D0+3)	-8	
NOT	NOT منطقی	مقدار بیتی	مقدار بیتی	NOT M0	TRUE	
*	ضرب	عمومی	عمومی	D0*3	15	
/	تقسیم	عمومی	عمومی	15/D0	3	
MOD	باقیمانده	اعداد صحیح	اعداد صحیح	D0 MOD 3	2	

[1] عملوندهای توان حتما باید مقدارثابت و یا سیمبول با فرمت ممیزشناور باشند و نمی‌توان از حافظه‌ها برای آن استفاده کرد.

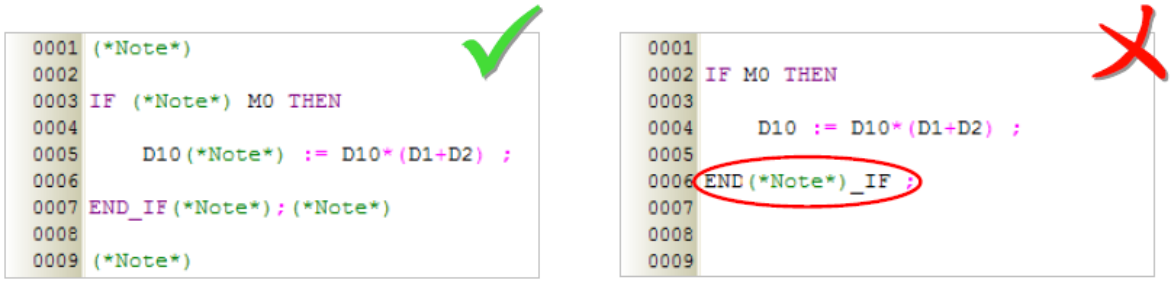
+,-	جمع، تفریق	هر عددی	هر عددی	D0+3	8	
<,>,<=,>=	عملیات مقایسه‌ای	هر عددی	بیتی	D0>2	TRUE	
=,<>	مساوی، ناتساوی	عمومی	بیتی	D0<>2	TRUE	
		بیتی		M0=TRUE	FALSE	
AND,&	AND منطقی	بیتی	بیتی	M0 & M1	FALSE	
XOR	XOR منطقی	بیتی	بیتی	M0 XOR M1	TRUE	
OR	OR منطقی	بیتی	بیتی	M0 OR M1	TRUE	پایین‌ترین

۹-۱-۵- کلمات کلیدی و توضیحات

در زبان‌های برنامه نویسی کلمات کلیدی[۱] دارای معانی مشخصی می‌باشند. برای مثال TRUE و FALSE در ST معرف مقادیر بولی هستند و IF معرف عملکردی اجرایی است. برای آنکه برنامه حین کامپایل شدن دچار اشتباه نشود، نمی‌توان سیمبول‌هایی مشابه کلمات کلیدی ساخت. با این حال استفاده ترکیبی از کلمات کلیدی مانند استفاده IF به صورت "IF_" و یا "FIFO" امکان پذیر است.

برای اضافه کردن توضیحات در برنامه می‌توان آن را بین علامت (* *) قرار داد. زمانی که برنامه اجرا می‌شود، سیستم به صورت اتوماتیک عبارات بین (* *) را نادیده می‌گیرد. همچنین تا زمانی که ساختار منطقی قسمت‌های مختلف برنامه به هم نخورد، کاربر می‌تواند توضیحات را در هر جای صفحه قرار دهد. مثلا در مثال زیر نمونه سمت چپ صحیح و نمونه سمت راست به علت آنکه توضیحات کلمه کلیدی END_IF را به دو قسمت تقسیم کرده است ناصحیح است.

[۱] Keywords



شکل ۹-۸ اضافه کردن توضیحات

۹-۱-۶- استفاده از سیمبول های با فرمت آرایه

در صورتی که کاربر بخواهد از سیمبول‌های با فرمت آرایه استفاده کند، نحوه بیان آن به صورت Identifier[Index] است (همانطور که بارها مطرح شد، آرایه‌ها در ISPSoft یک بعدی هستند). Index می‌تواند مقداری ثابت و یا سیمبول باشد (برای AH500 فقط می‌توان از سیمبول استفاده کرد). مقدار Index از صفر شروع می‌شود تا یکی کمتر از طول آرایه، برای مثال برای یک آرایه ده تایی Index می-تواند بین صفر تا ۹ باشد. در صورتی که مقدار Index از طول آرایه بیشتر باشد، خطایی رخ نمی‌دهد و PLC متوقف نمی‌شود ولی PLC با خواندن مقادیر اشتباه ممکن است وظیفه خود را به خوبی انجام ندهد.

```
0001 IF M0 THEN
0002
0003 Ary_A[0] := Ary_B[IDX] ;
0004
0005 END_IF ;
```

شکل ۹-۹ استفاده از آرایه در ST

۹-۱-۷- نکات مهم پیرامون ST

نکاتی که باید در برنامه نویسی به آن‌ها توجه شود:

- در زبان ST از ":=" برای اختصاص دهی و از "=" به معنای برابری در دستورات شرطی استفاده می‌شود.
- در صورتی که ساختار کلمه کلیدی و یا Statementها تغییر کند، امکان ایجاد خط جدید و ترک فضای خالی وجود خواهد داشت.
- زبان برنامه نویسی ST برخلاف زبان برنامه نویسی C به کوچک و بزرگ بودن حروف حساس نیست و مثلا IF و if در آن تفاوتی ندارند.

- در صورتی که کاربر بخواهد مقدار ثابتی را در ISPSoft با استفاده از زبان ST به کار ببرد، باید آن رابه صورت زیر معرفی کند.

جدول ۹-۲: فرمت اعداد در ISPSoft

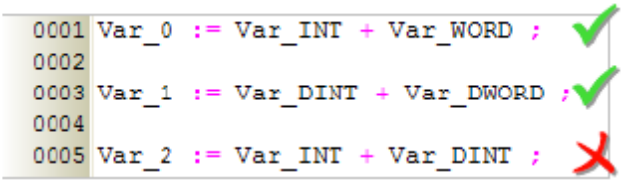
نوع	شرح
دسیمال[۱] (دهدهی= مبنای ده)	مانند 23456 – هر عدد بدون نشانه دسیمال در نظر گرفته می‌شود.
اُکتال[۲] (مبنای ۸)	مانند 8#5564 – مقادیر اُکتال با علامت #8 در ابتدایشان مشخص می-شوند.
هگزادسیمال[۳] (مبنای ۱۶)	مانند 16#5BAD – مقادیر هگزادسیمال (یا به صورت مختصر هگز) با علامت #16 در ابتدایشان مشخص می‌شوند.
باینری (عدد مبنای دو)	مانند 2#11101010011 – مقادیر باینری با علامت #2 در ابتدایشان مشخص می‌شوند.
String	"XU016S" – کاراکترهای در میان علامت نقل قول قرار می‌گیرند.
مقدار بولی (بیتی)	AH500: SM400 (کانتکت باز) یا SM401 (کانتکت بسته) مورد استفاده قرار می‌گیرند. DVP: M1000 (کانتکت باز) یا M1001 (کانتکت بسته) مورد استفاده قرار می‌گیرند.

[۱] Decimal Value

[۲] Octal Value

[۳] Hexadecimal

- در هر Expression عملوندها باید دارای فرمت واحد باشند. با این حال اگر طول عملوندها یکی باشد، امکان انجام عملیات ممکن است، به عنوان نمونه می‌توان از اعداد INT به همراه فرمت WORD و یا اعداد DINT به همراه فرمت DWORD در یک عملیات استفاده کرد.



شکل ۹-۱۰ لزوم یکسان بودن عملوندها در یک Expression

- در صورتی که نوع سیمبول WORD، DWORD، LWORD، INT، DINT و یا LINT باشد، سیستم با آن‌ها همانند اعداد صحیح علامت دار برخود می‌کند.
- اگر چه محدودیتی برای کُدنویسی وجود ندارد ولی کاربر باید به محدودیت حافظه پردازشگر توجه کند.
- بخش‌های مختلف برنامه ST در ISPSoft قابلیت کپی، برش و الحاق دارند.
- دستورالعمل‌هایی که ST پشتیبانی می‌کند شامل موارد زیر نیست:

جدول ۹-۳: دستورالعمل هایی که ST پشتیبانی نمی کند

سری DVP	**سری AH500**
دستورات پایه مانند LD، LDI، OUT، SET، RST، LD& و ... پشتیبانی نمی‌شوند.	
دستورات پالسی مانند ROLP، RORP، BCDP، TRDP و ... پشتیبانی نمی‌شوند.	
PLS-PLF-FOR-NEXT-CJ-MC-MCR-STL-RET و ... پشتیبانی نمی‌شوند.	دستورالعمل چهارعمل حسابی و دستورات منطقی و نوع کانتکت‌ها و دستورات Structure Creation پشتیبانی نمی‌شوند. PLS-PLF-MOV-DMOV-DFMOV-$MOV-MC-MCR-CJ-JMP-GOEND و ... پشتیبانی نمی‌شوند.

۹–۲– ساختار Statement

۹–۲–۱– تخصیص دهی

- فرمت

حافظه و یا سیمبول :=Expression;

(مثال: D10:=100*(D1+D3))

- شرح

مقدار Expression در سمت راست علامت =: به سیمبول و یا حافظه سمت چپ منتقل می شود.

- قواعد

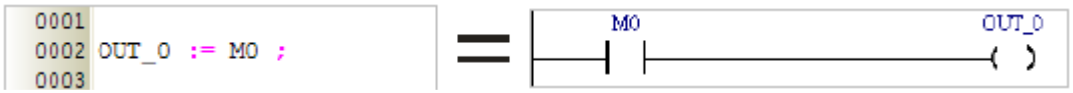
  - Expression سمت راست علامت =: می‌تواند مقدار ثابت، حافظه، سیمبول و یا عملیات حسابی باشد. ولی سمت چپ علامت =: تنها حافظه و یا سیمبول می‌تواند قرار گیرد.
  - عملوندهای دو سوی علامت =: باید از قواعد زیر پیروی کنند.

    - اگر یکی از عملوندها از نوع حافظه M/S/T/C/HC باشد، عملگر دیگر می-تواند سیمبولی با فرمت داده از نوع بولی، شمارنده، زمان‌سنج و یا STEP باشد (سیمبول‌های نوع بیتی).
    - اگر یکی از عملوندها از نوع D و یاL باشند، عملگر دیگر نمی‌تواند سیمبولی با فرمت داده از نوع بولی، شمارنده، زمان‌سنج و یا STEP باشد.
    - اگر یکی از عملوندها از نوع حافظه T و یا C باشد، عملگر دیگر می‌تواند داده با فرمت D ویا L و یا همچنین سیمبولی با فرمت داده از نوع WORD، INT، زمان سنج و یا شمارنده باشد.
    - اگر یکی از عملوندها از نوع حافظه D/L/HC باشد، و یا سیمبولی از نوع شمارنده که به HC اختصاص داده می‌شود باشد. عملوند دیگر سیمبولی با فرمت داده DWORD و یا DINT خواهد بود.

- اگر یکی از عملوندها سیمبولی با فرمت داده نوع WORD، DWORD و یا LWORD باشد، عملوند دیگر باید به صورت INT، DINT و یا LINT باشد. طول داده سمت چپ علامت =: همیشه باید بزرگتر و یا مساوی داده سمت راست آن باشد.
- هر دو عملوند باید به صورت REAL/LREAL/STRING باشند. طول داده در این حالت باید در دو طرف برابر باشد.

- مثال

مثال ۱: مقدار M0 به سیمبول OUT_0 تخصیص داده شود.



شکل ۹-۱۱ تخصیص دهی- مثال ۱

مثال ۲: مقدار یک (فعال بودن=روشن بودن بودن) به M0 تخصیص داده شود.

شکل ۹-۱۲ تخصیص دهی – مثال ۲

مثال ۳: مقدار D2 با D1 جمع شود و نتیجه در D0 ذخیره شود.

شکل ۹-۱۳ تخصیص دهی – مثال ۳

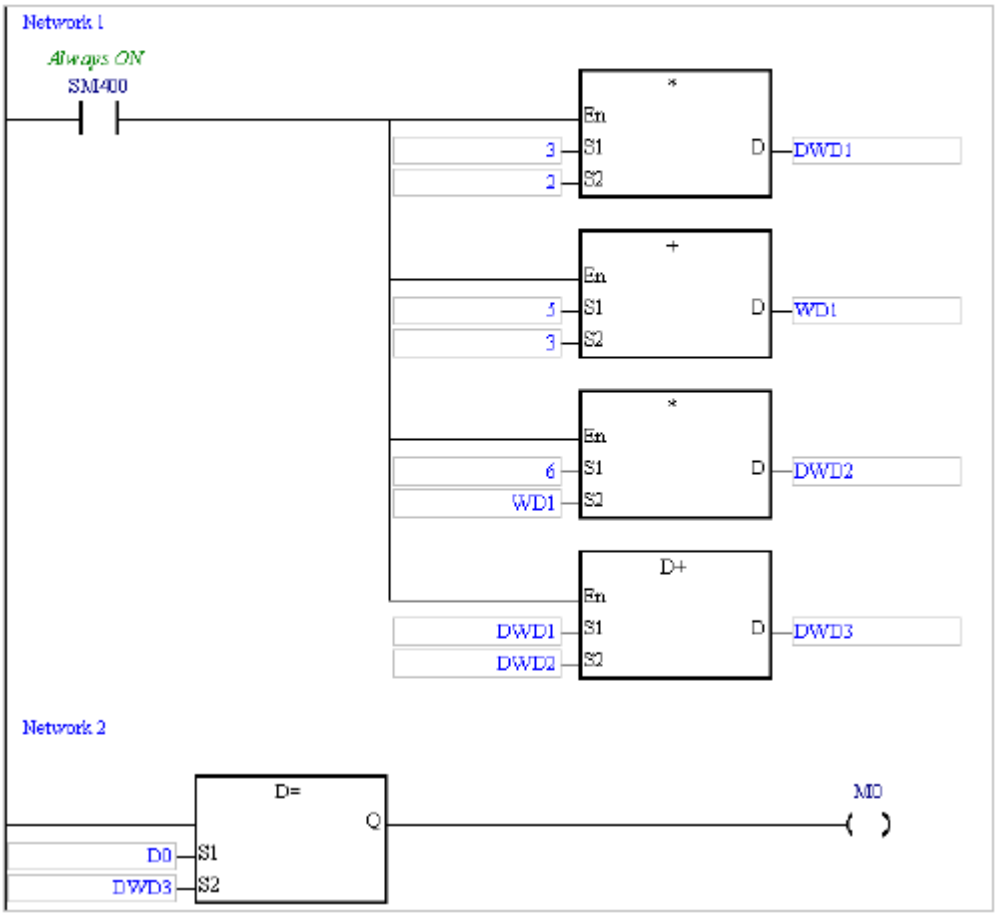
مثال ۴: مقدار ۳ به سیمبول DATA تخصیص داده شود.

شکل ۹-۱۴ تخصیص دهی – مثال ۴

مثال ۵: اگر مقدار عبارت (3+5)*6+2*3 برابر با مقدار ذخیره شده در D0 بود نتیجه برابری (درست و یا غلط بودن آن= نتیجه بیتی) به M0 تخصیص داده شود.

```
0001
0002 M0 := (D0=3*2+6*(5+3)) ;
0003
```

=



شکل ۹-۱۵ تخصیص دهی – مثال ۵

چند مثال از تخصیص ناصحیح را در جدول زیر مشاهده می‌کنید (در نام سیمبول های این بخش فرمت داده آن‌ها نیز گنجانده شده است).

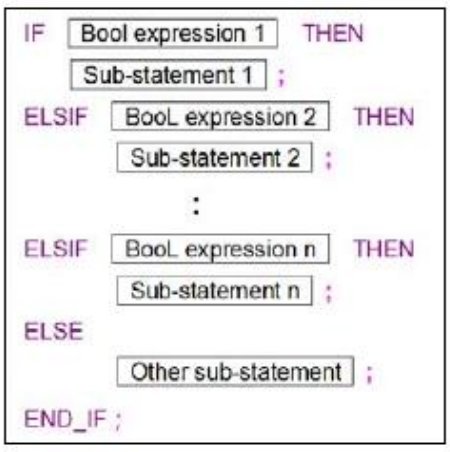
جدول ۹-۴: مثال هایی از انواع تخصیص دهی ناصحیح

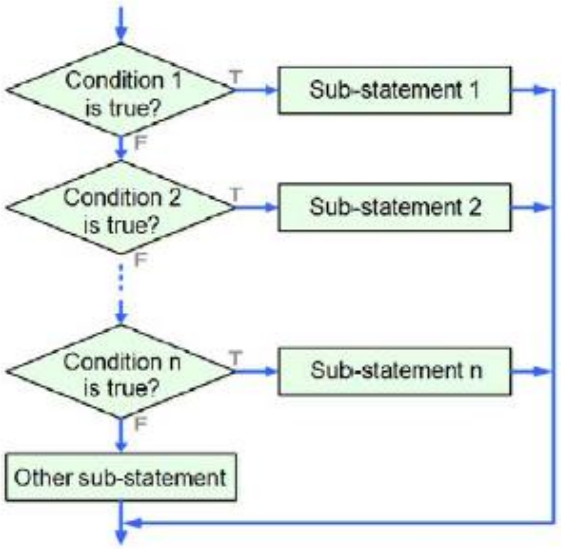
عبارت	شرح
D0 := M0 ;	نوع داده عملوندها نمی‌تواند متفاوت باشد. در اینجا داده سمت راست حافظه بیتی و داده سمت چپ حافظه WORD است.

V_WORD := V_DWORD ;	طول داده سمت چپ علامت =: نمی‌تواند کوچکتر از داده سمت راست علامت =: باشد.
D0 := V_REAL ;	داده های REAL نمی‌توانند به رجیسترهای D تخصیص داده شوند.
V_LREAL := V_REAL ;	عملوندهای REAL فقط به عملوندهای REAL تخصیص داده می‌شوند و عملوندهای LREAL فقط به عملوندهای LREAL تخصیص داده می‌شوند.
V_DWORD := V_TIMER ;	مقدار سیمبول داده زمان‌سنج نمی‌تواند به داده با فرمت DWORD تخصیص داده شود، ولی می‌تواند به داده با فرمت WORD تخصیص داده شود.
V_WORD := HC0 ;	مقدار حافظه‌ی HC امکان تخصیص به سیمبول‌های با فرمت WORD را ندارد ولی می‌تواند به داده با فرمت DWORD تخصیص داده شود.

۹-۲-۲- دستور شرطی IF

- فرمت





شکل ۹-۱۶ فرمت دستور شرطی IF

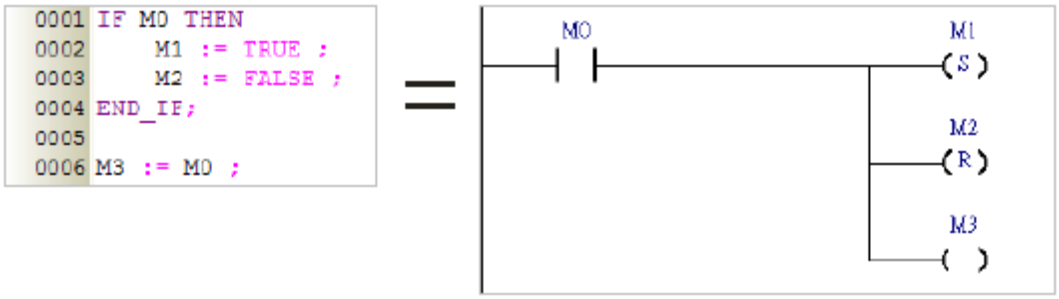
- شرح

مقدار بولی Bool expression 1 اگر یک (صحیح) باشد، Sub-statement 1 اجرا خواهد شد. در صورتی که مقدار بولی Bool expression 1 صفر (نادرست) باشد، Bool expression 2 بررسی می‌شود و در صورت یک بودن آن Sub-statement 2 اجرا می-شود. این روند ادامه دارد و اگر تمامی Bool expression ها صفر بودند آنگاه Other sub-statement اجرا خواهد شد.

- قواعد
  - مقدار Expression بولی می‌تواند حافظه، سیمبول و یا عبارت حسابی باشد ولی نمی‌تواند مقدار ثابت باشد.
  - در Sub-Statement ها نیز می‌توانیم از دستور IF بار دیگر استفاده کنیم.
  - محدودیتی در دستورات مورد استفاده بعد از THEN و یا ELSE وجود ندارد.
  - می‌توان از بخش‌های مربوط به ELSEIF و ELSE صرف نظر کرد. در صورتی که از ELSE صرف نظر کنیم و مقادیر بولی Statementهای IF و ELSEIF نادرست باشند، هیچ statement-ای اجرا نخواهد شد.
  - کاربر به هر تعداد می‌تواند از ELSEIF پس از IF استفاده کند ولی تنها می‌تواند از یک ELSE آن هم پس از تمام ELSEIFها استفاده کند. در انتهای حلقه شرطی نیز حتما باید ;END_IF را قرار داد.

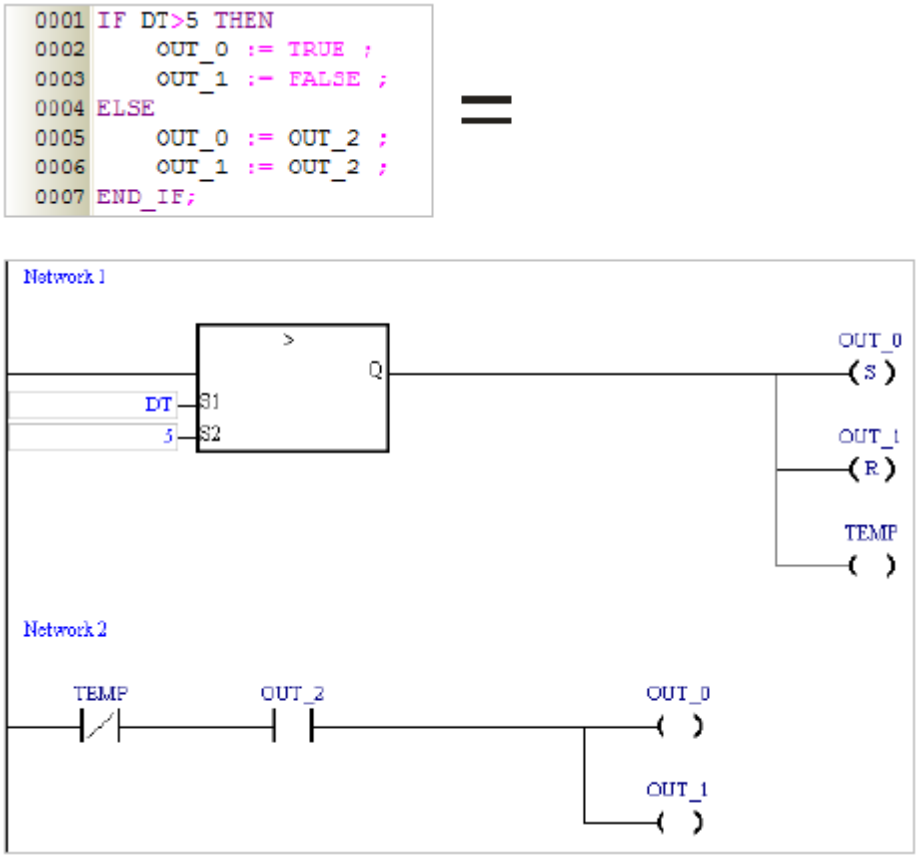
- مثال

مثال۱: اگر M0 یک بود، M1 یک شود و M2 صفر شود، در غیر این صورت اگر M0 صفر بود هیچ اتفاقی نیافتد. همچنین مستقل از صفر یا یک بودن M0 مقدار آن به M3 تخصیص داده شود.



شکل ۹-۱۷ دستور شرطی IF- مثال ۱

مثال ۲: در صورتی که مقدار DT بزرگتر از ۵ بود، OUT_0 فعال و OUT_1 غیرفعال شود. در غیر این صورت مقدار OUT_2 به OUT_1 و OUT_0 تخصیص داده شود.

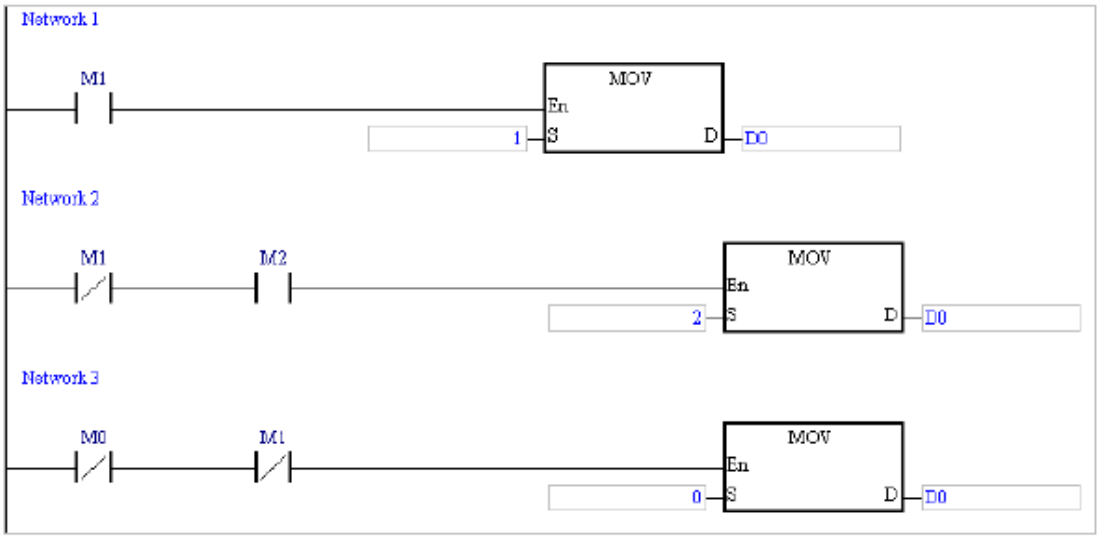


شکل ۹-۱۸ دستور شرطی IF- مثال ۲

مثال ۳: اگر M1 یک بود D0 برابر یک شود و اگر M1 صفر بود M2 ارزیابی شود. در صورتی که M2 یک بود مقدار D0 برابر ۲ شود و در غیر این صورت اگر M2 صفر بود مقدار D0 صفر شود.

```
0001 IF M1 THEN
0002     DO := 1 ;
0003 ELSIF M2 THEN
0004     DO := 2 ;
0005 ELSE
0006     DO := 0 ;
0007 END_IF;
```

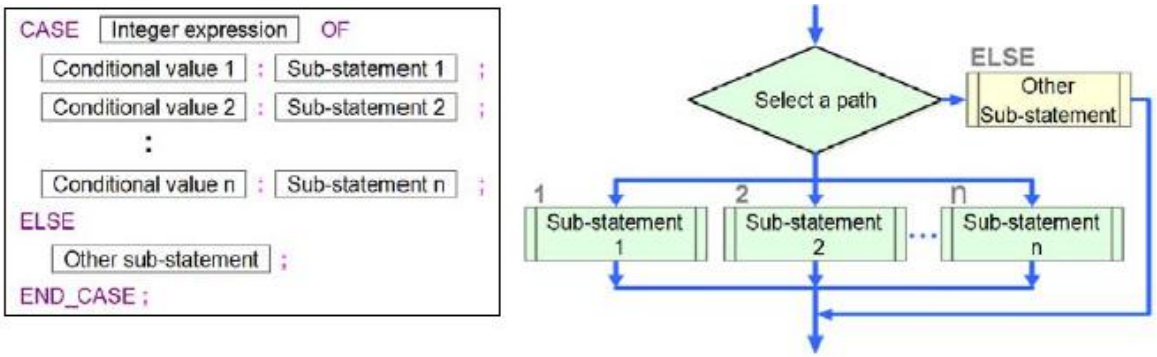
=



شکل ۹-۱۹ دستور شرطی IF- مثال ۲

۹-۲-۳- دستور شرطی CASE

- فرمت



شکل ۹-۲۰ فرمت دستور شرطی CASE

- شرح

PLC بررسی می‌کند که آیا مقدار Integer expression با هیچ کدام از مقادیر Conditional Value شماره یک تا n برابر است، در این صورت، Sub-statement متناظر با آن را اجرا خواهد کرد در غیر این صورت زمانی که هیچ مقدار برابری در Conditional Valueها وجود نداشت، Other sub-statement اجرا خواهد شد.

- قواعد
  - Integer expression می‌تواند حافظه، سیمبول و یا عبارت حسابی باشد ولی نباید مقدار ثابت باشد. مقدار Integer expression باید از نوع صحیح (INT بین منفی ۳۲۷۶۸ الی ۳۲۷۶۷ و یا DINT بین منفی ۲۱۴۷۴۸۳۶۴۸ الی ۲۱۴۷۴۸۳۶۴۷) باشد. در صورتی که از حافظه برای Integer expression استفاده شود، سیستم نوع داده آن را صحیح در نظر می‌گیرد و اگر عبارت حسابی برای Integer expression مورد استفاده قرار گیرد، نتیجه به سمت نزدیکترین عدد صحیح کوچکتر و یا مساوی گِرد خواهد شد.
  - هرکدام از Condition Value ها باید منحصر به فرد باشند. محدوده Condition Value به نوع داده Integer expression وابسته است، در صورتی که Integer expression به فرمت INT باشد مقدار Condition Value باید بین منفی ۳۲۷۶۸ الی ۳۲۷۶۷ و اگر DINT باشد، مقدار Condition Value باید منفی ۲۱۴۷۴۸۳۶۴۸ الی ۲۱۴۷۴۸۳۶۴۷ باشد.
  - اگر sub-statement برای چند Condition Value یکسان باشند، کاربر می-تواند آن‌ها را با هم ترکیب کند.
    - در صورتی که مقادیر Condition Valueهایِ دارایِ sub-statement یکسان اعداد صحیح متوالی نباشند، می‌توان آن‌ها را در یک Condition Value با ویرگول در کنار هم قرار داد به عنوان مثال "1, 3, 5: Sub-statement" که یعنی Sub-statement زمانی که مقدار Integer expression برابر ۱ و یا ۳ و یا ۵ است اجرا شود.
    - در صورتی که مقادیر Condition Valueهایِ دارایِ sub-statement یکسان، متوالی باشند می‌توان محدوده آن‌ها را با دو نقطه ".." بین عدد آغاز و پایان مشخص کرد. به عنوان مثال "3..6: Sub-

statement" که یعنی Sub-statement زمانی که مقدار Integer expression برابر ۳ الی ۶ است (شامل خود ۳ و ۶ یعنی ۳ یا ۴ یا ۵ یا ۶) اجرا شود. توجه شود که مقدار سمت چپ باید کمتر از مقدار سمت راست باشد و گرنه عبارتی همچون "6..3" ناصحیح است.

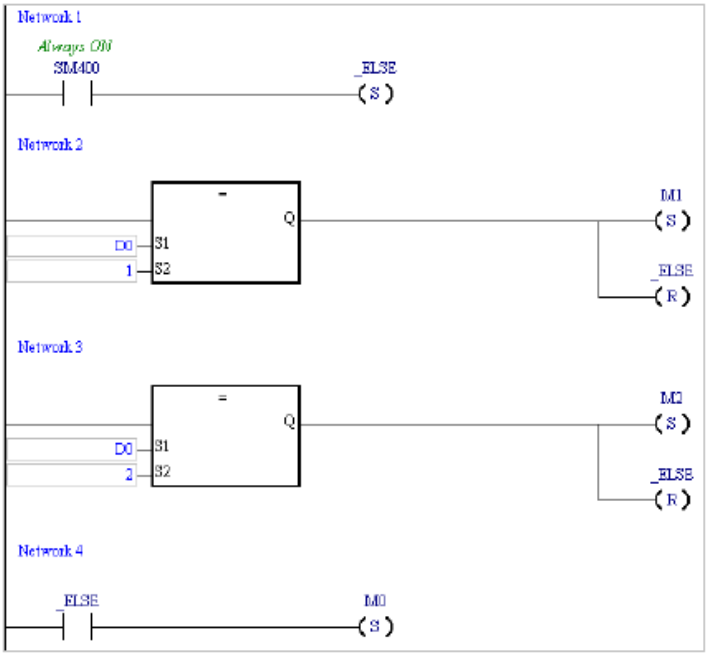
- Sub-Statement می‌تواند هر عبارت مُجازی شامل IF و یا CASE باشد.
- محدودیتی در تعداد Condition Value و میزان عبارت های پس از آن در Sub-Statement ها وجود ندارد.
- می‌توان از بخش ELSE صرف نظر کرد، در این صورت اگر Integer expression با هیچکدام از Condition Valueها برابر نبود، هیچ عبارتی اجرا نخواهد شد.
- پس از اجرای Sub-Statement برنامه خود به خود به خارج از حلقه CASE می‌پَرد و نیاز به تابع پَرش نیست.
- ELSE باید همیشه پس از تمامی Condition Valueها بیاید و در انتها END_CASE پایان حلقه را مشخص می‌کند.

- مثال

مثال ۱: اگر مقدار D0 برابر ۱ بود، M1 روشن شود و اگر برابر ۲ بود M2 روشن شود، در غیر این صورت M0 روشن شود.

```
0001 CASE D0 OF
0002   1: M1 := TRUE ;
0003   2: M2 := TRUE ;
0004 ELSE
0005   M0 :=TRUE ;
0006 END_CASE;
```

=

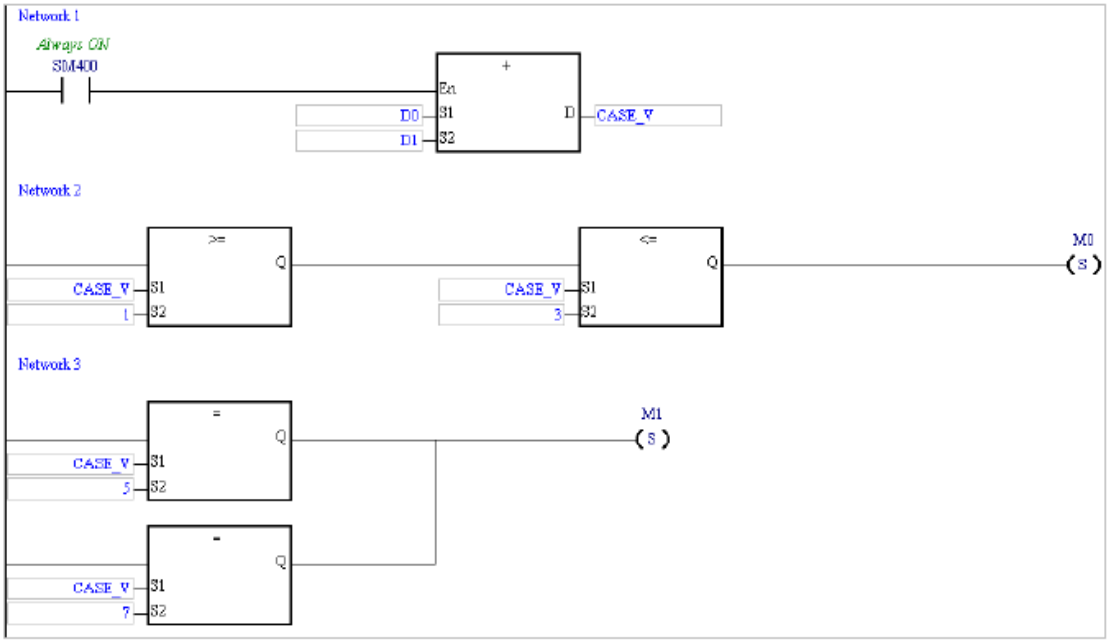


شکل ۹-۲۱ استفاده از دستور شرطی CASE- مثال ۱

مثال ۲: اگر مقدار عبارت حسابی D0+D1 بین ۱ الی ۳ بود، M0 روشن شود و اگر ۵ و یا ۷ بود، M1 روشن شود. در غیر این صورت هیچ اتفاقی نیافتد.

```
0001 CASE D0+D1 OF
0002    1..3: M0 := TRUE ;
0003    5,7 : M1 := TRUE ;
0004 END_CASE;
```

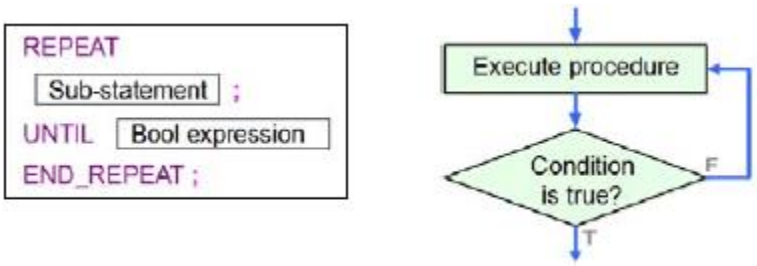
=



شکل ۹-۲۲ استفاده از دستور شرطی CASE- مثال ۲

۹-۲-۴- دستور حلقه REPEAT

- فرمت



شکل ۹-۲۳ فرمت دستور حلقه REPEAT

- شرح

Sub-statement اجرا می‌شود و سپس مقدار Bool expression بررسی می‌شود که اگر یک بود ، حلقه به پایان می‌رسد، در غیر این صورت اگر مقدار Bool expression صفر بود برنامه به صورت متوالی Sub-statement را اجرا می‌کند تا مقدار Bool Expression یک شود و از حلقه خارج شود.

- قواعد
  - Bool expression می‌تواند حافظه، سیمبول و یا عملیات حسابی باشد ولی نباید مقدار ثابت باشد.
  - قبل از ارزیابی Bool expression همیشه ابتدا یکبار Sub-statement اجرا می‌شود.
  - Sub-statement هر عبارت مُجازی شامل IF و یا CASE و ... می‌تواند باشد و محدودیتی برای کُد مربوط به آن وجود ندارد. می‌توان از حلقه‌های تو در تو در برنامه استفاده کرد ولی این تعداد حلقه‌های تو در تو نباید بیشتر از ۶۴ عدد باشد.
  - در صورتی که مقدار Bool expression مقابل UNTIL یک شود، حلقه متوقف می‌شود. برای جلوگیری از ایجاد حلقه بی نهایت (حلقه‌ای که دائما اجرا می-شود) نباید حافظه و یا سیمبولی که ثابت است را قرار داد (باید بتوان با به وجود آوردن تغییر در آن‌ها شرط خروج را ایجاد کرد).
  - حافظه‌های پورت‌های ورودی X پس از اسکن برنامه به روز می‌شوند و اگر از آن‌ها به عنوان Bool expression استفاده شود، به روز نخواهند شد (چون زمان اسکن به پایان نمی‌رسد![1] و در حلقه گرفتار شده‌است). به همین خاطر زمانی که از X در Bool expression استفاده می‌شود یا باید از دستورالعمل‌های به روز رسانی ورودی خروجی استفاده شود و یا اینکه X را از طریق وقفه به روز کرد.

[1] علت این موضوع آن است که PLC پس از اجرای کل برنامه (یک اسکن تایم) مقدار ورودی و خروجی‌ها را به روز می‌کند در حالی که در اینجا به علت گیر کردن در حلقه هیچگاه قادر به اتمام اسکن تایم نخواهد شد.

- مثال

مثال ۱: مقدار اولیه DT صفر است. در ساختار حلقه به صورت متوالی آن را دو واحد دو واحد زیاد کنید تا مقدار آن از ۱۰۰ بیشتر شود، سپس می‌توانید از حلقه خارج شوید.

```
0001 DT := 0 ;
0002 REPEAT
0003   DT := DT + 2 ;
0004 UNTIL DT > 100
0005 END_REPEAT;
```

شکل ۹-۲۴ حلقه REPEAT –مثال ۱

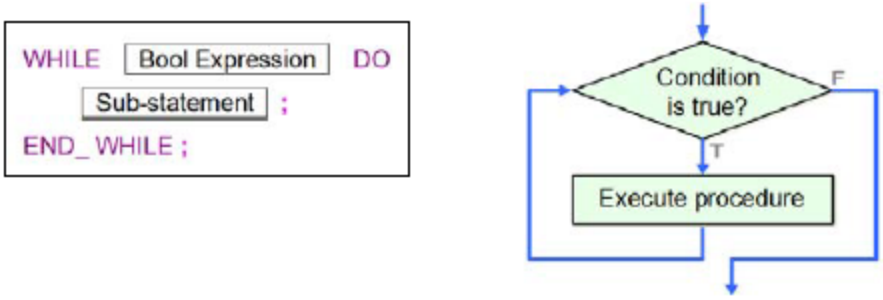
مثال ۲: مقدار اولیه D0 یک است و مقدار اولیه D10 برابر ۵ است. در انتهای حلقه باید D0 برابر 1*2*3*4*5 شود.

```
0001 D10 := 5 ;
0002
0003 D0 := 1 ;
0004 REPEAT
0005   D0 := D0 * D10 ;
0006   D10 := D10 - 1 ;
0007 UNTIL  D10 = 0
0008 END_REPEAT ;
```

شکل ۹-۲۵ حلقه REPEAT –مثال ۲

۹–۲–۵– دستور حلقه WHILE

- فرمت



شکل ۹-۲۶ فرمت دستور حلقه WHILE

- شرح

مقدار Bool expression بررسی می‌شود و اگر یک بود Sub-statement اجرا می‌شود و این فرایند ادامه پیدا می‌کند تا Bool expression صفر شود و از حلقه خارج شویم.

- قواعد
  - Bool expression می‌تواند حافظه، سیمبول و یا عملیات حسابی باشد ولی نباید مقدار ثابت باشد.
  - برخلاف REPEAT (که حتما Sub-statement در آن حداقل یکبار اجرا می‌شود) چون در اینجا ابتدا Bool expression بررسی می‌شود ممکن است در صورت صفر بودن آن Sub-statement هیچگاه اجرا نشود.
  - Sub-statement هر عبارت مُجازی شامل IF و یا CASE و ... می‌تواند باشد و محدودیتی برای کُد مربوط به آن وجود ندارد. می‌توان از حلقه‌های تو در تو در برنامه استفاده کرد ولی این تعداد حلقه‌های تو در تو نباید بیشتر از ۶۴ عدد باشد.
  - در صورتی که مقدار Bool expression مقابل WHILE صفر شود، حلقه متوقف می‌شود. برای جلوگیری از ایجاد حلقه بی نهایت (حلقه‌ای که دائما اجرا می‌شود) نباید حافظه و یا سیمبولی ثابت قرار داد (باید بتوان با به وجود آوردن تغییر در آن‌ها شرط خروج را ایجاد کرد).
  - حافظه‌های پورت‌های ورودی X پس از اسکن برنامه به روز می‌شوند و اگر از آن‌ها به عنوان Bool expression استفاده شود، به روز نخواهند شد (چون زمان اسکن به پایان نمی‌رسد! و در حلقه گرفتار شده‌ایم). به همین خاطر زمانی که از X در Bool expression استفاده می‌شود یا باید از دستورالعمل‌های به روز رسانی ورودی خروجی استفاده شود و یا اینکه X را از طریق وقفه به روز کرد.
- مثال

مثال ۱: مقدار اولیه DT صفر است. در ساختار حلقه به صورت متوالی آن را دو واحد دو واحد زیاد کنید تا مقدار آن از ۱۰۰ بیشتر شود، سپس می‌توانید از حلقه خارج شوید.

```
0001 DT := 0 ;
0002
0003 WHILE DT <= 100 DO
0004   DT := DT + 2 ;
0005 END_WHILE;
```

شکل ۹-۲۷ حلقه WHILE –مثال ۱

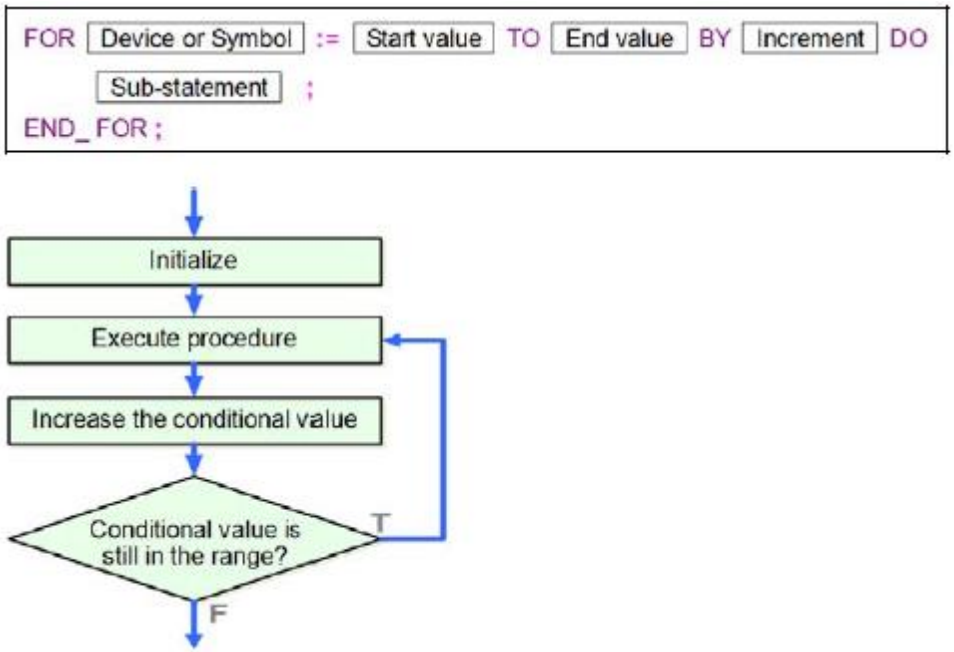
مثال ۲: در مثال زیر حلقه REAPEAT پس از اجرای یکباره دستوراتش در sub-statement (D0=0+1=1) مقدار Bool Expression را چک می‌کند و چون مقدار آن (M0) برابر یک است از حلقه REPEAT خارج می‌شود. حال نوبت به اجرای حلقه WHILE می‌رسد که ابتدا باید مقدار Bool Expression در آن بررسی شود. مقدار Bool Expression در آن برابر صفر است (معکوس M0) پس حلقه WHILE اجرا نمی‌شود و همچنان مقدار D1 برابر یک می‌ماند.

```
0001 M0 := TRUE ;
0002 D0 := 0 ;
0003 D1 := 0 ;
0004
0005 REPEAT
0006  D0 := D0 + 1 ;
0007 UNTIL  M0
0008 END_REPEAT ;
0009
0010 WHILE (NOT M0) DO
0011    D1 := D1 + 1 ;
0012 END_WHILE ;
```

شکل ۹-۲۸ حلقه WHILE –مثال ۲

۹–۲–۶– دستور حلقه FOR

- فرمت



شکل ۹-۲۹ فرمت دستور حلقه FOR

- شرح

مقدار اولیه سیمبول و یا حافظه با start value مشخص می‌شود. هر بار که حلقه اجرا می‌شود مقدار سیمبول و یا حافظه به میزان increment واحد افزوده می‌شود. اگر مقدار سیمبول و یا حافظه در محدوده Start value تا end value باشد، حلقه اجرا می‌شود و اگر مقدار سیمبول و یا حافظه از محدوده Start value تا end value خارج شود، حلقه به پایان می‌رسد.

- قواعد

  - مقدار start value، end value و increment می‌تواند حافظه، سیمبول، مقدار ثابت و یا عبارت حسابی باشد. نوع داده‌های start value، end value و increment نیز باید INT و یا DINT باشند. در صورتی که مقدار start value، end value ویا increment عملیات حسابی باشند، نتیجه عملیات آن‌ها به پایین (به نزدیکترین عدد صحیح کوچکتر مساوی) گِرد می‌شود.
  - اگر end value از start value بیشتر باشد، مقدار increment باید منفی باشد، در غیر اینصورت حلقه بی‌نهایت خواهیم داشت.

- نوع داده سیمبول باید INT و یا DINT باشد، در صورتی که سیمبول و یا حافظه در عملیات درون حلقه دستکاری می‌شود، باید دقت کرد که حلقه بی نهایت ایجاد نشود.

- Sub-statement هر عبارت مجازی شامل IF و یا CASE و ... می‌تواند باشد و محدودیتی برای کُد مربوط به آن وجود ندارد. می‌توان از حلقه‌های تو در تو در برنامه استفاده کرد ولی این تعداد حلقه‌های تو در تو نباید بیشتر از ۶۴ عدد باشد.

- پس از آنکه مقدار سیمبول و یا حافظه به اندازه increment افزایش پیدا کرد، سیستم بررسی می‌کند که آیا مقدار مورد نظر بین محدوده start value و end value هست. در صورتی که حین افزایش به میزان increment، سرریز[۱] رخ دهد، شاهد حلقه بی نهایت خواهیم بود.

- مثال

مقدار اولیه سیمبول INDEX برابر یک است، تا زمانی که حلقه اجرا می‌شود مقدار INDEX دو واحد اضافه می‌شود. پس از پنج بار اجرای حلقه مقدار INDEX به ۱۱ می‌رسد و چون دیگر INDEX بین محدوده ۱ تا ۹ نیست حلقه متوقف می‌شود.مقدار اولیه سیمبول SU صفر است، پس از پایان کار حلقه این مقدار برابر $1^2+3^2+5^2+7^2+9^2$ خواهد شد.

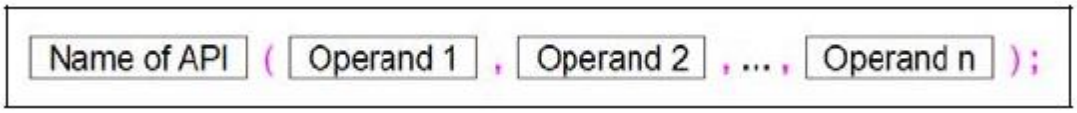
```
0001 SU := 0 ;
0002
0003 FOR  INDEX := 1  TO  9  BY  2  DO
0004      SU :=  INDEX * INDEX + SU ;
0005 END_FOR ;
0006
```

شکل ۹-۳۰ حلقه For – مثال

[۱] Overflow

۹-۲-۷- استفاده از دستورالعمل ها

- فرمت



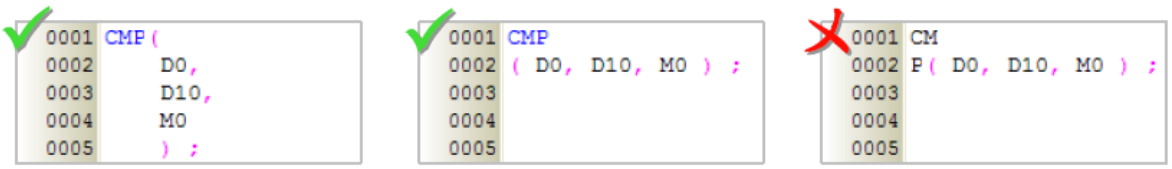
شکل ۹-۳۱ فرمت استفاده از دستورالعمل ها

- شرح

دستورالعمل (یا بلوک آماده نرم افزار) مورد نظر تنها با تعیین المان‌های ورودی و خروجی آن اجرا خواهد شد.

- قواعد

  - نیازی به اعمال و در نظر گرفتن پایه En وجود ندارد. زمان اجرای Statement شامل دستورالعمل، آن دستورالعمل نیز اجرا خواهد شد. اگر نیاز به در نظر گرفتن پایه En وجود دارد می‌توان از دستور شرطی IF استفاده کرد.
  - تا زمانی که ساختار نام عملوندها و کلمات کلیدی به هم نریزد می‌توان از Enter برای رفتن به خط جدید استفاده کرد.



شکل ۹-۳۲ استفاده از Enter در زمان فراخوانی دستورالعمل

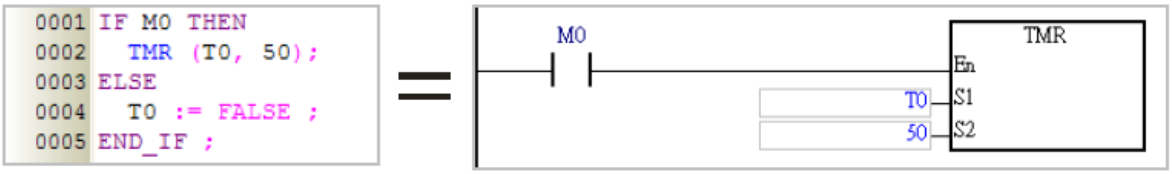
  - چون دستورالعمل‌ها Statement به حساب می‌آیند نیاز است در انتهای آن‌ها نقطه ویرگول ";" قرار دهیم.
  - عملوندهای دستورالعمل باید بین پرانتز قرار گیرند.
  - ترتیب عملوندها مهم و مطابق ترتیب پایه‌های (ورودی و ورودی/خروجی) و سپس خروجی است.
  - دستورالعمل‌ها در ISPSoft را نمی‌توان تخصیص داد و برای نمونه عبارت M0:=CMP(D0,D10,M0); نادرست است.

- مثال

مثال ۱: دستورالعمل CMP با استفاده از دستور شرطی IF.

شکل ۹-۳۳ استفاده از دستورالعمل ها –مثال ۱

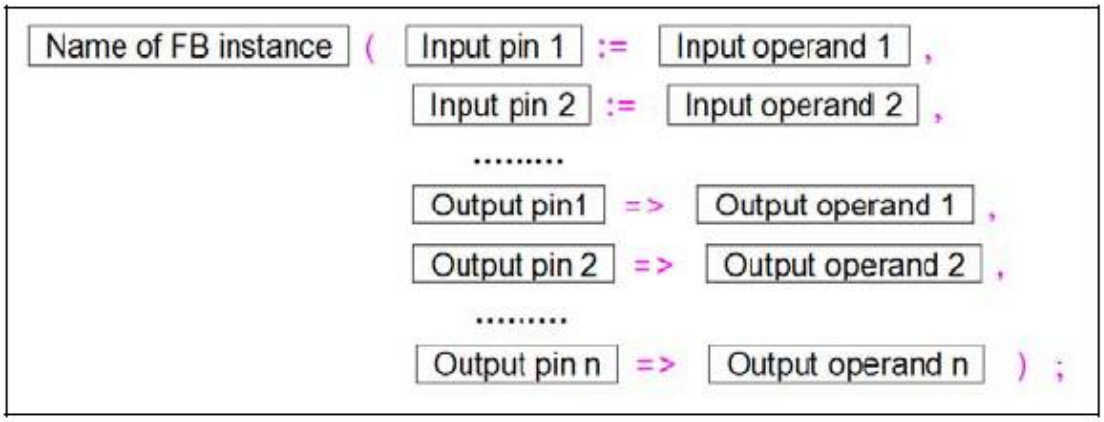
مثال ۲: با آنکه پایه En نیازی نیست در دستورالعمل‌های به زبان ST لحاظ شود، در مواقعی مانند زمانی که از TMR استفاده می‌کنیم، برای ریست کردن نیاز به دسترسی به En داریم تا بتوانیم مقدار آن را صفر کنیم برای این موضوع می‌توان عملکرد En را با IF پیاده کرد. در این مثال در خط 0004 عمل ریست کردن زمان سنج صورت می‌پذیرد.



شکل ۹-۳۴ استفاده از دستورالعمل ها –مثال ۲

۹–۲–۸– استفاده از Function Block ها

- فرمت



شکل ۹-۳۵ فرمت استفاده از FB

- شرح

تابع بلوکی مورد نظر تنها با تعیین المان‌های ورودی و خروجی آن اجرا خواهد شد.

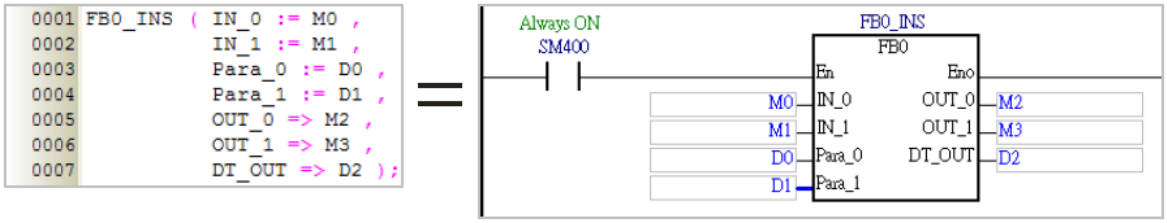
- قواعد

  - نیازی به اعمال و در نظر گرفتن پایه En وجود ندارد. زمان اجرای Statement شامل بلوک، آن بلوک نیز اجرا خواهد شد. اگر نیاز به در نظر گرفتن پایه En وجود دارد می‌توان از دستور شرطی IF استفاده کرد.
  - تا زمانی که ساختار نام عملوندها و کلمات کلیدی به هم نریزد می‌توان از Enter برای رفتن به خط جدید استفاده کرد.
  - چون بلوک‌ها Statement به حساب می‌آیند نیاز است در انتهای آن‌ها نقطه ویرگول ";" قرار دهیم.
  - عملوندهای و نوع داده ورودی و یا خروجی باید بین پرانتز قرار گیرند، هر دسته از ورودی و یا خروجی‌ها با ویرگول جدا می‌شوند و داده‌های اختصاص داده شده به آن‌ها باید مطابق با نوع داده داده‌های تعریف شده در بلوک باشد.
  - نام بلوک قبل از پرانتز به نوعی نسخه است و نه مرجع بلوک. در واقع این نام سیمبولی است برای داده به فرمت آن بلوک.
  - اختصاص پایه ورودی با علامت := و اختصاص پایه خروجی با علامت => صورت می‌پذیرد.
  - ترتیب عملوندها مهم و مطابق ترتیب پایه‌های (ورودی و ورودی/خروجی) و سپس خروجی است. پایه‌های ورودی/خروجی مطابق پایه‌های ورودی تخصیص داده می‌شوند. نیازی به تخصیص پایه‌های En و Eno نیست.

شکل ۹-۳۶ ترتیب اعمال پایه ها در ST مطابق ترتیب پایه های بلوک است

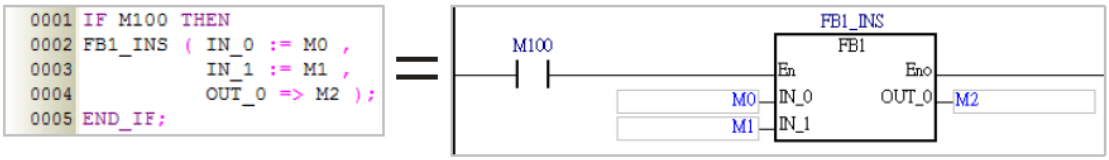
- مثال

مثال ۱: زمانی که بلوکی در ST فراخوانی می‌شود، اجرا نیز می‌شود.



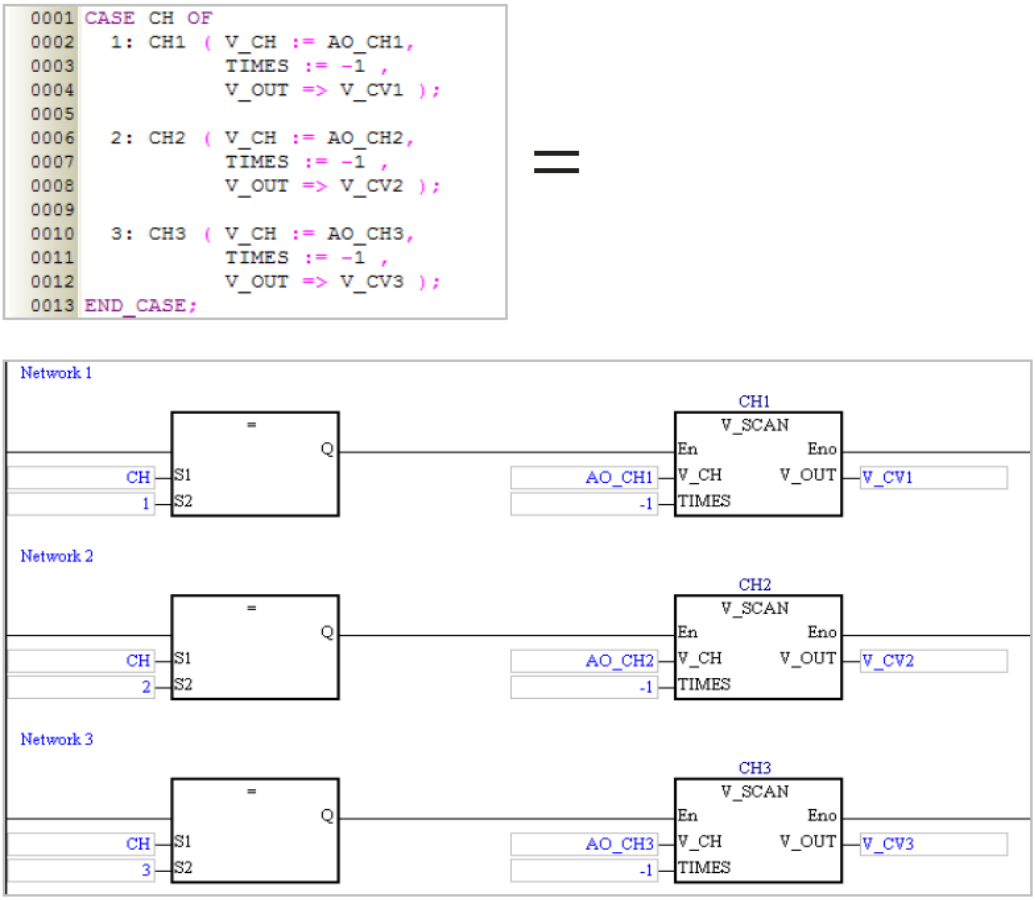
شکل ۹-۳۷ استفاده از FB – مثال ۱

مثال ۲: استفاده از بلوک با استفاده از IF برای اعمال عملکرد پایه En



شکل ۹-۳۸ استفاده از FB – مثال ۲

مثال ۳: مقدار سیمبول CH تصمیم می‌گیرد کدام بلوک اجرا شود.



شکل ۹-۳۹ استفاده از FB – مثال ۳

۹-۲-۹- دستور تاخیر

- فرمت

شکل ۹-۴۰ فرمت دستور تاخیر

- شرح

با استفاده از نقطه ویرگول، عملکردی در PLC اجرا نمی‌شود و تنها در اجرای دستورات بعدی تاخیر اتفاق می‌افتد.

- مثال

حلقه FOR زیر عملکردی را اجرا نمی‌کند و تنها در اجرای دستورات بعدی تاخیر ایجاد می‌کند. کاربر می‌تواند با تنظیم دفعات تکرار حلقه میزان تاخیر را مشخص کند.

```
0001 FOR D100:=1 TO 3000 BY 1 DO
0002     ;
0003 END_FOR;
```

شکل ۹-۴۱ دستور تاخیر- مثال

۹-۲-۱۰- دستور RETURN

- فرمت

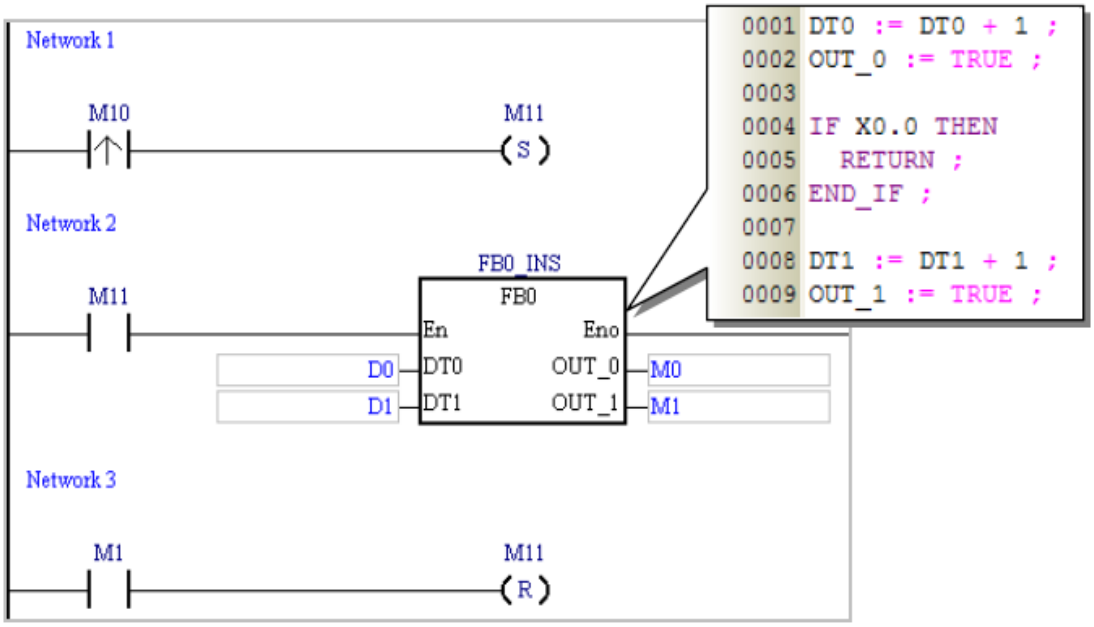
شکل ۹-۴۲ فرمت دستور RETURN

- شرح

RETURN در POU بلوک‌ها به زبان ST مورد استفاده قرار می‌گیرد و به اجرای بلوک خاتمه می‌دهد.

- مثال

از زبان ST برای برنامه نویسی بلوک FB0 استفاده می‌کنیم. پس از اتمام بخش اول برنامه X0.0 را مورد بررسی قرار می‌دهیم، اگر یک بود اجرای ادامه برنامه را متوقف می‌کنیم و اگر صفر بود تمام برنامه بلوک را اجرا می‌کنیم.



شکل ۹-۴۳ دستور RETURN – مثال

۹–۲–۱۱– دستور EXIT

- فرمت

شکل ۹-۴۴ فرمت دستور EXIT

- شرح

EXIT در حلقه‌های REPEAT، WHILE و FOR مورد استفاده قرار می‌گیرد و در صورت اجرای آن از حلقه خارج می‌شویم.

- مثال

مثال ۱: حلقه REPEAT تا زمانی که DT برابر صفر نشود ادامه خواهد داشت. مقدار اولیه D0 برابر صفر است که هر بار اجرای حلقه یک واحد به آن اضافه می‌کند. همچنین دستور شرطی IF زمانی که D0 برابر ۱۰۰ است EXIT را اجرا می‌کند و از حلقه خارج

خواهیم شد. پس اگر حلقه ۱۰۰ بار اجرا شود، D0 به ۱۰۰ رسیده و به وسیله EXIT از حلقه خارج می‌شویم. همچنین مقدار اولیه DT برابر ۱۱۰ است و با هر بار اجرای حلقه یک واحد از آن کم می‌شود تا بعد از ۱۱۰ بار اجرا مقدار آن صفر شده و بتواند با شرط روبروی UNTIL اجرای حلقه را متوقف کند. با این حال پس از ۱۰۰ بار اجرای حلقه، زمانی که DT برابر ۱۰ است به وسیله EXIT از حلقه خارج خواهیم شد.

```
0001 DT := 110 ;
0002 D0 := 0 ;
0003
0004 REPEAT
0005
0006    DT  :=  DT - 1 ;
0007    D0  :=  D0 + 1 ;
0008
0009    IF  D0=100  THEN
0010      EXIT ;
0011    END_IF ;
0012
0013 UNTIL  DT = 0
0014 END_REPEAT ;
```

شکل ۹-۴۵ دستور EXIT- مثال ۱

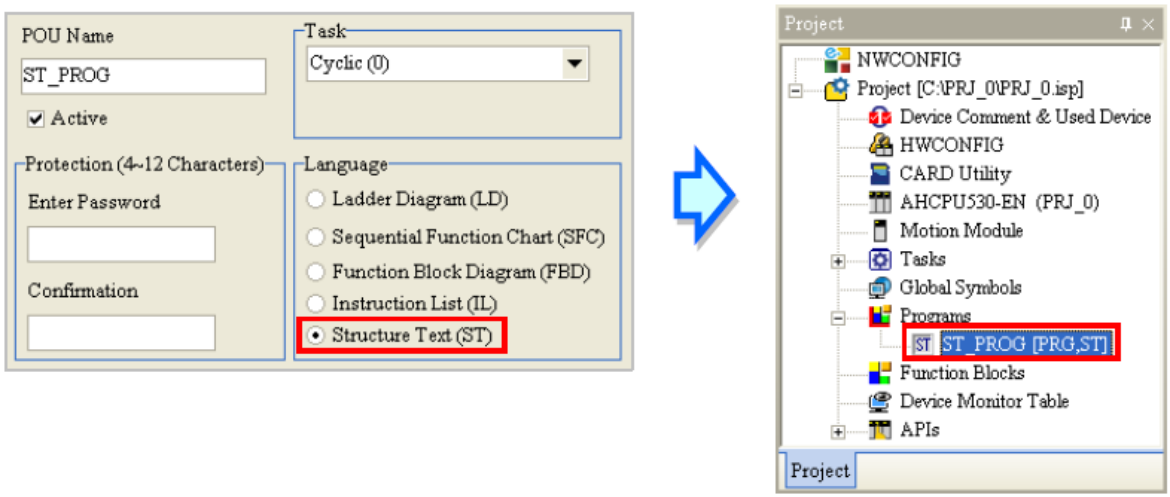
مثال ۲: در حلقه FOR مقدار D0*D10 محاسبه و نتیجه در D0 ذخیره می‌شود. دستور شرطی IF در صورتی که D0 بزرگتر از ۱۰۰ باشد، EXIT را اجرا می‌کند تا از حلقه خارج شویم. این اتفاق زمانی که D10 به ۵ می رسد رخ می‌دهد، در این هنگام D0=1*2*3*4*5 می‌شود.

```
0001 DT := 10 ;
0002 D0 := 1 ;
0003
0004 FOR D10 := 1 TO DT BY 1 DO
0005
0006     D0 := D0 * D10 ;
0007
0008     IF  D0>100  THEN
0009       EXIT ;
0010     END_IF ;
0011
0012 END_FOR ;
```

شکل ۹-۴۶ دستور EXIT- مثال ۲

۹-۳- ساخت برنامه به زبان ST

برای ساخت برنامه‌ای به زبان ST در پروژه، حین ایجاد POU جدید، زبان برنامه نویسی را در قسمت Language برابر Structured Text (ST) قرار می‌دهیم.



شکل ۹-۴۷ ساخت برنامه به زبان ST- قسمت ۱

محیط ویرایش برنامه به زبان ST در شکل زیر آمده است. در بخش بالایی این پنجره می‌توانیم سیمبول‌ها را تعریف و در بخش پایین صفحه می‌توانیم به ویرایش برنامه خود بپردازیم.

شکل ۹-۴۸ ساخت برنامه به زبان ST- قسمت ۲، محیط ویرایش برنامه

با باز شدن محیط ویرایش به زبان برنامه نوار ابزار مربوط به آن نیز باز خواهد شد (یک نکته در مورد زبان‌های نوشتاری برنامه نویسی آن است که استفاده از ماوس سرعت برنامه نویس را کاهش می‌دهد، به

همین علت به شما پیشنهاد می‌شود تا جای ممکن سرعت تایپ خود را بهبود ببخشید و از ماوس استفاده نکنید)

شکل ۹-۴۹ نوار ابزار برنامه نویسی به زبان ST

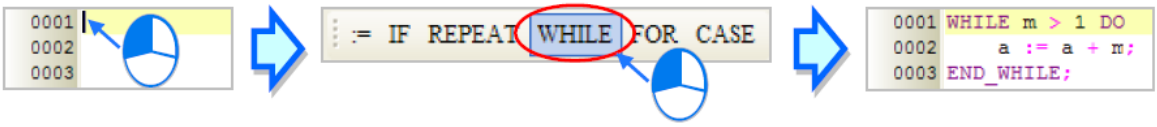
جدول ۹-۵: آیکون های نوار ابزار برنامه نویسی ST

نماد	عملکرد
:=	اضافه کردن نمونه‌ای از ساختار تخصیص دهی
IF	اضافه کردن نمونه‌ای از ساختار دستور شرطی IF
REPEAT	اضافه کردن نمونه‌ای از ساختار حلقه شرطی REPEAT
WHILE	اضافه کردن نمونه‌ای از ساختار حلقه شرطی WHILE
FOR	اضافه کردن نمونه‌ای از ساختار حلقه FOR
CASE	اضافه کردن نمونه‌ای از ساختار دستور شرطی CASE

نوشتن برنامه ST همانند تایپ کردن در فایل Text می‌ماند، کاربر کُدها را تایپ می‌کند و هرکجا که لازم باشد، با فشردن Enter به خط جدید می‌رود.

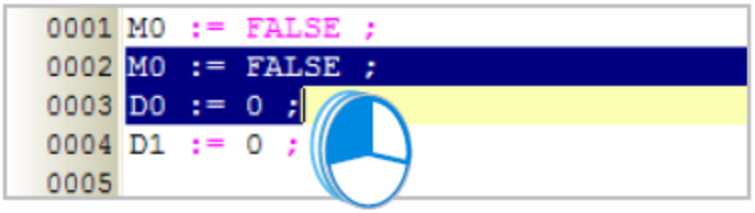
شکل ۹-۵۰ برنامه نویسی به زبان ST

همچنین با کلیک بر روی نوار ابزار زبان برنامه نویسی ST، می‌توان نمونه‌ای از ساختار تابع مورد نظر را در برنامه اضافه کرد.



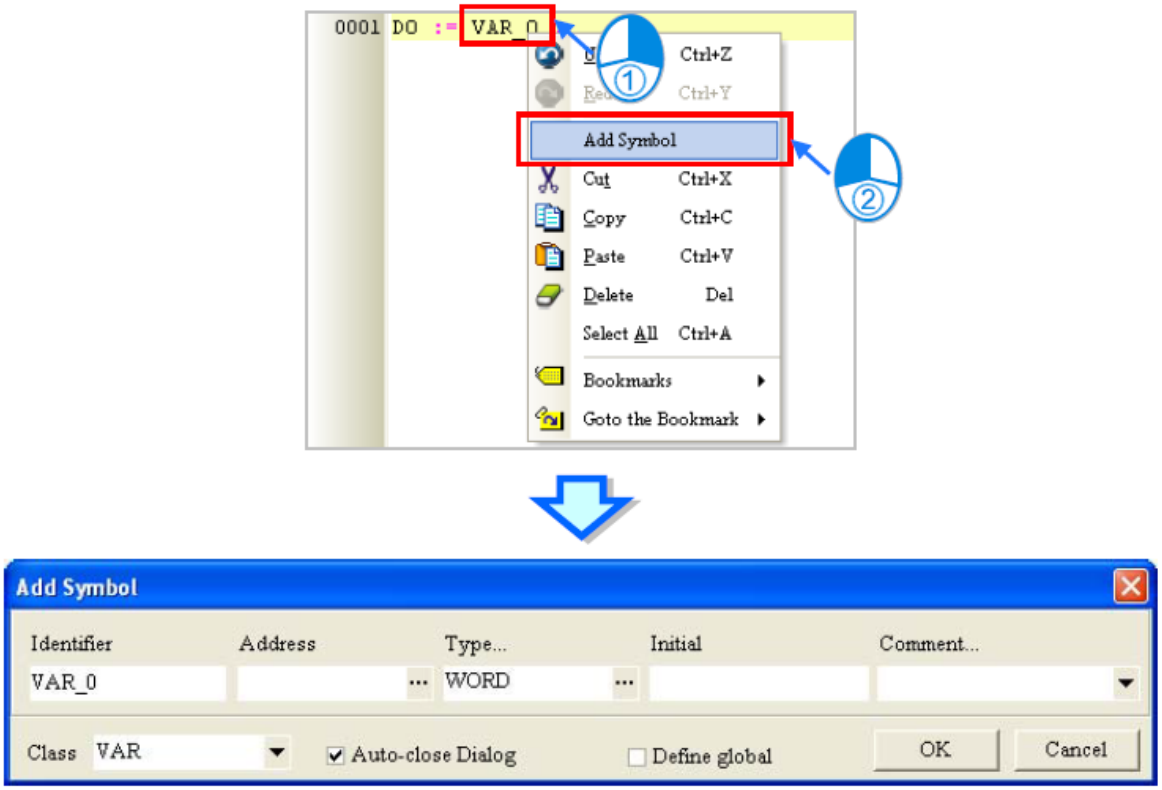
شکل ۹-۵۱ استفاده از مثال های آماده دستورات نوار ابزار برنامه نویسی

برای انتخاب قسمتی از برنامه می‌توان بر روی نقطه ابتدایی آن کلیک کرد و با کشیدن نشانگر موس تا نقطه انتهایی (در این حین می‌توان از اسکرول موس نیز استفاده کرد)، قسمت مورد نظر در برنامه را انتخاب کرد و بر روی آن عملیات‌هایی چون کپی و الحاق را انجام داد.



شکل ۹-۵۲ انتخاب بخشی از برنامه

اگر کاربر از سیمبولی که هنوز در جدول سیمبول‌ها وارد نشده در برنامه استفاده کند، می‌تواند بر روی سیمبول تایپ شده کلیک و سپس کلیک راست کرده و Add Symbol را انتخاب کند و در پنجره باز شده سیمبول را تعریف کند.



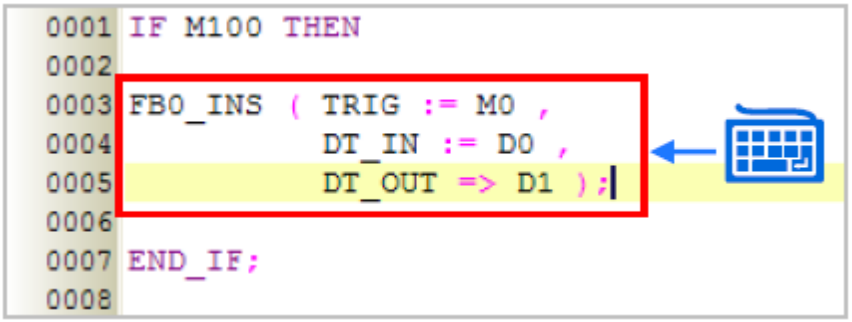
شکل ۹-۵۳ تعریف سیمبول حین برنامه نویسی

۹-۳-۱- استفاده از بلوک ها و دستورالعمل ها

کاربر به یکی از دو روش زیر می‌تواند بلوک و یا دستورالعمل‌ها را در برنامه به کار بگیرد.

- روش ۱

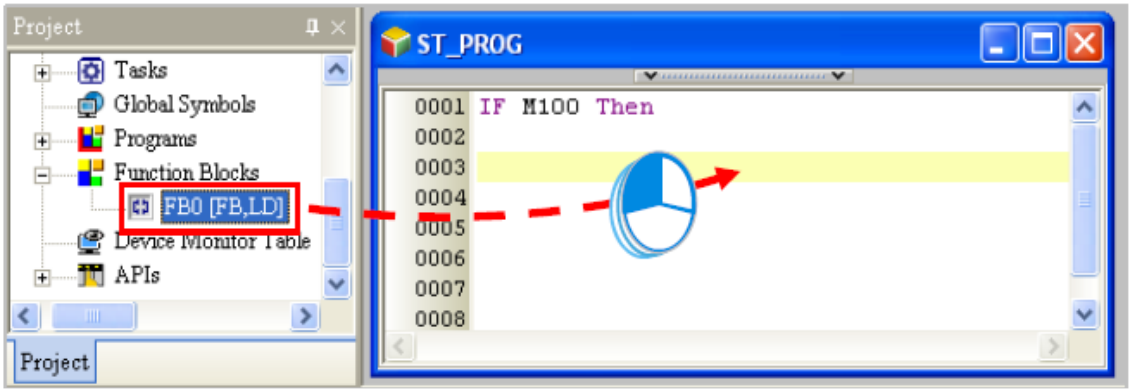
آن را مطابق فرمت‌های معرفی شده با در نظر گرفتن ترتیب پایه‌ها تایپ کرد.



شکل ۹-۵۴ تایپ فرمت بلوک و یا دستورالعمل

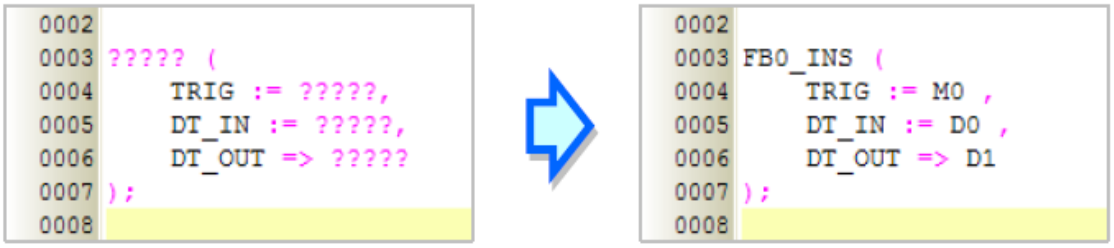
- روش ۲

از قسمت APIs و یا Functio Blocks در بخش مدیریت پروژه آن‌ها را انتخاب و به بخش مورد نظر در برنامه می‌کشیم.



شکل ۹-۵۵ استفاده از FB و یا APIهای های لیست شده در بخش مدیریت پروژه

پس از اضافه کردن بلوک و یا دستورالعمل‌ها، باید عملوندهای مرتبط با هر پایه را مشخص کنیم.

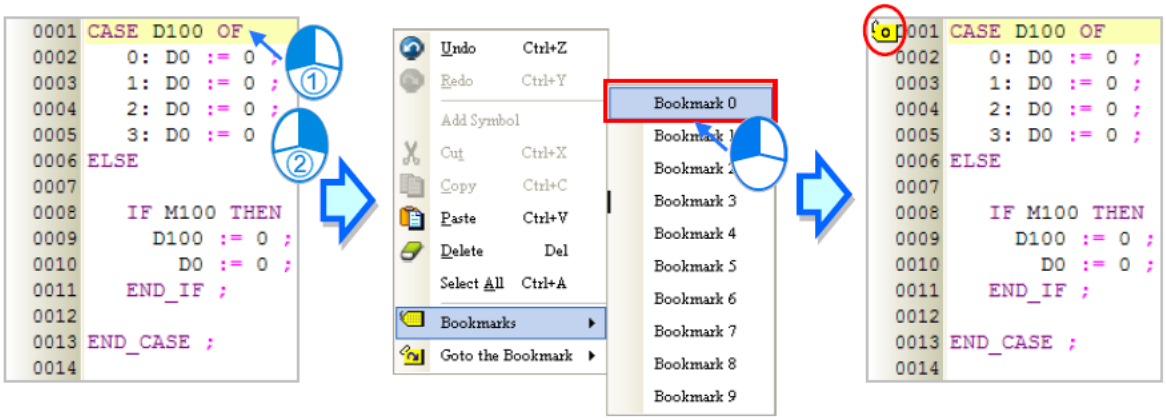


شکل ۹-۵۶ مشخص کردن عملوند مرتبط هر پایه بلوک

۹-۳-۲- نشان گزاری

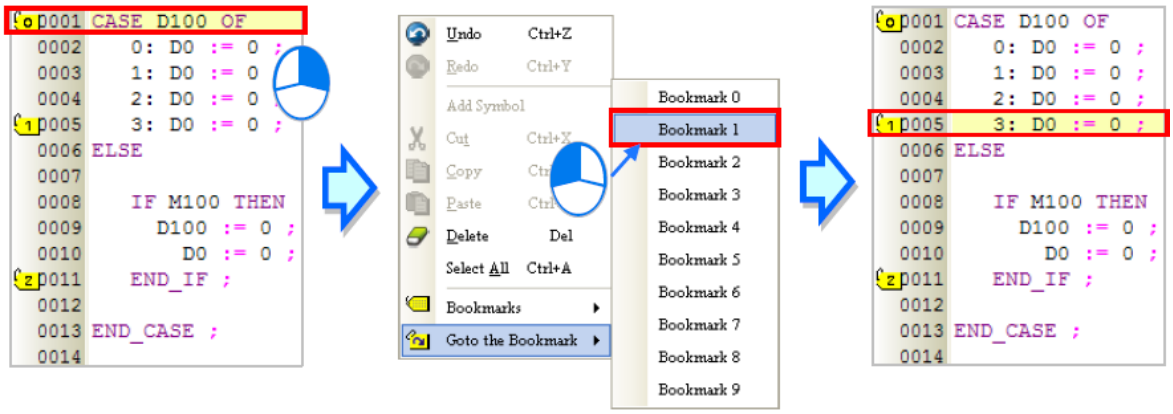
با استفاده از نشانه گزاری برنامه می‌توان برنامه نویسی را به همراه برجسته کردن توضیحات و بخش-های مهم برنامه انجام داد. نشانه گزاری همچنین باعث ایجاد نظم و جستجوی راحت تر در بین خطوط برنامه می‌شود. با استفاده از نشان‌ها به راحتی می‌توان بین خطوط نشان‌دار جابجا شد و به آن‌ها دسترسی داشت.

برای اضافه کردن نشان به یک خط ابتدا باید آن را انتخاب و سپس کلیک راست کرده و در بخش Bookmark شماره نشان را انتخاب کرد. در صورتی که شماره نشان انتخاب شده از قبل وجود داشته باشد، نشان مورد نظر حذف خواهد شد و نشان جدید جایگزین می‌شود. کاربر می‌تواند تمامی نشان‌ها را از منوی Edit با انتخاب Remove All Bookmarks از بخش Bookmarks پاک کند.



شکل ۹-۵۷ نشان گزاری در ST

پس از نشان گزاری با کلیک راست در محیط ویرایش برنامه و انتخاب Goto the Bookmark و انتخاب شماره مورد نظر می‌توان به محل خط نشان‌گزاری شده مورد نظر رفت.



شکل ۹-۵۸ جابجایی بین نشان ها

۹-۴- مثال برنامه نویسی ST

۹-۴-۱- شرح برنامه

می‌خواهیم برنامه‌ای برای کنترل سیستم روشنایی با سه رنگ (شامل سه لامپ با رنگ‌های متفاوت) و یک زنگ موجود در آن را طراحی کنیم. مکانیزم کنترل روشنایی با توجه به وضعیت عملکردی سیستمی که سیستم روشنایی بر روی آن نصب است در جدول زیر لیست شده است. در صورتی چند وضعیت در سیستم به طور همزمان رخ دهند، تقدم سیستم با وضعیت اولویت بالاتر است.

شکل ۹-۵۹ سیستم روشنایی متصل به PLC

جدول ۹-۶: وضعیت های سیستم اصلی و عملکرد سیستم روشنایی

شماره وضعیت	وضعیت سیستم	سیستم روشنایی				اولویت
		قرمز	زرد	سبز	زنگ	
۴	بد عمل کردن	روشن	خاموش	خاموش	روشن و خاموش	بالاتر

	سیستم				(معادل چشمک زن)	
۳	بد عمل کردن سیستم (زنگ خاموش شده)	چشمک زن	خاموش	خاموش	خاموش	
۲	سیستم در حال کار	خاموش	خاموش	روشن	خاموش	
۱	کار تمام شده	خاموش	چشمک زن	خاموش	خاموش	
۰	دستگاه بیکار	خاموش	روشن	خاموش	خاموش	پایین‌تر

۹-۴-۲- تنظیمات سخت افزاری

در این مثال از پردازنده سری AH500 مدل AHCPU530-EN، ماژول ورودی/خروجی دیجیتال AH16AP11R-5A استفاده می‌کنیم. تخصیص ورودی و خروجی‌های سیستم مطابق جدول زیر است.

جدول ۹-۷: تخصیص ورودی و خروجی های سیستم

نوع	آدرس	شرح
خروجی دیجیتال	Y0.0	نور قرمز
خروجی دیجیتال	Y0.1	نور زرد
خروجی دیجیتال	Y0.2	نور سبز
خروجی دیجیتال	Y0.3	زنگ

۹-۴-۳- عملکرد برنامه

۱- اگر پرچم ERROR فعال بود و پرچم BUZZER_OFF غیرفعال بود، کد وضعیت برابر ۴ خواهد بود.

۲- اگر پرچم ERROR فعال بود و پرچم BUZZER_OFF فعال بود، کد وضعیت برابر ۳ خواهد بود.

۳- اگر پرچم ERROR غیرفعال بود و پرچم RUNNING فعال بود، کد وضعیت برابر ۲ خواهد بود.

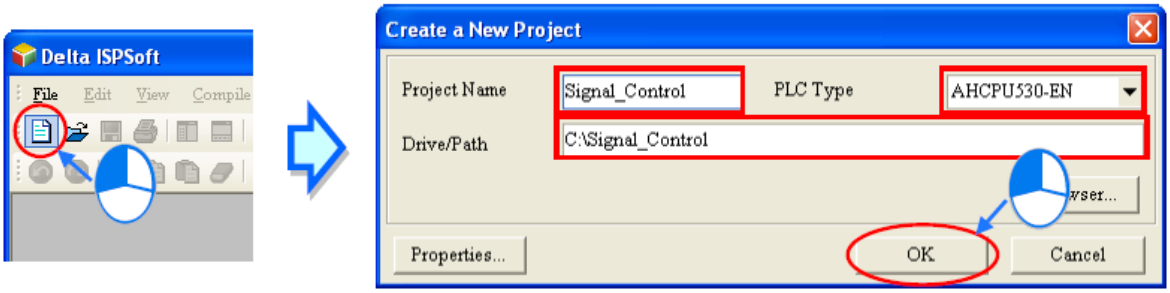
۴- اگر پرچم ERROR و RUNNING غیرفعال بودند و پرچم COMPLETE فعال بود، کد وضعیت برابر ۱ خواهد بود.

۵- اگر پرچم ERROR و RUNNING و COMPLETE غیرفعال بودند کد وضعیت برابر ۰ خواهد بود.

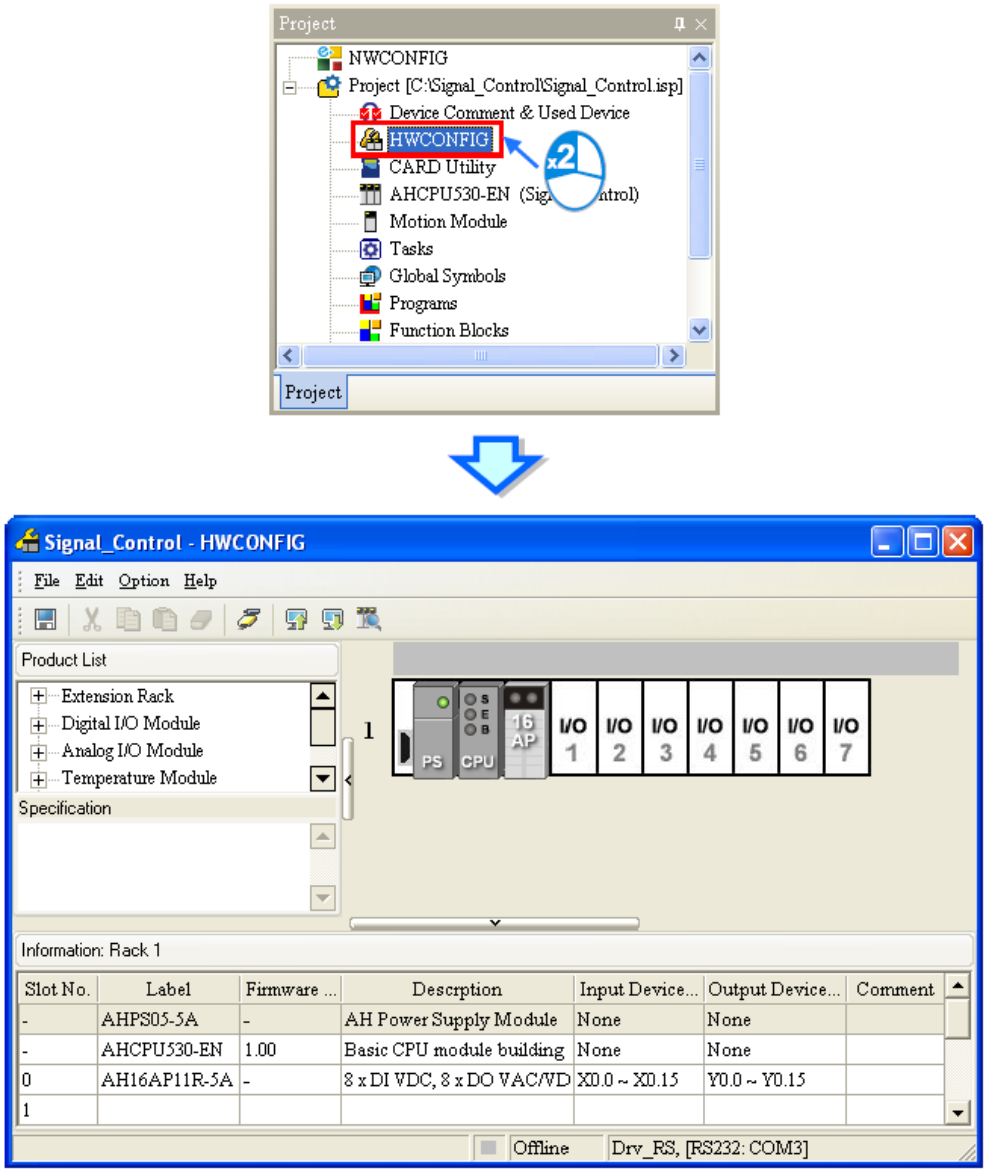
۶- هر کد وضعیت، حالتی از روشنایی را ایجاد می‌کند و سه رنگ آن می‌تواند جهت کاربردهای عملی متفاوتی مورد استفاده قرار گیرد. باید در بلوکی که طراحی می‌شود، امکان گسترش عملکرد سیستم روشنایی لحاظ شود (با کد نویسی مناسب و قابل گسترش).

۹-۴-۴- ساخت برنامه

در ISPSoft پروژه جدیدی را می‌سازیم. و تنظیمات سخت افزاری آن را همانند پروژه‌های نمونه آمده در فصل ۴- انجام می‌دهیم.

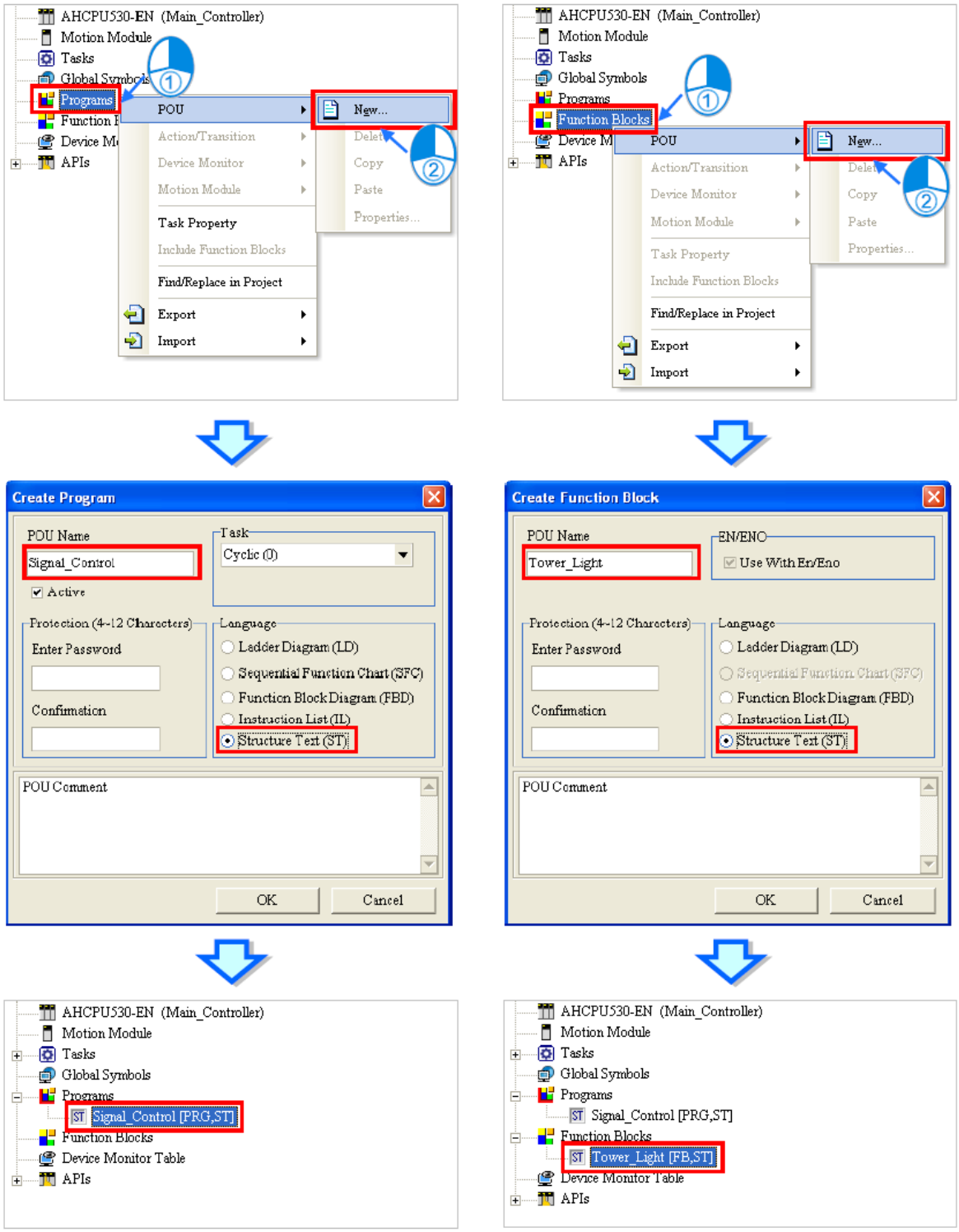


شکل ۹-۶۰ ساخت پروژه – مثال



شکل ۹-۶۱ تنظیمات سخت افزاری – مثال

یک POU برنامه و یک POU بلوکی با زبان برنامه نویسی ST می‌سازیم.



شکل ۹-۶۲ ساخت POU برنامه و بلوکی - مثال

پرچم‌ها و خروجی‌های دیجیتال را در جدول سیمبول سراسری وارد می‌کنیم.

Global Symbols

Global Symbols

Class	Identifiers	Address	Type...	Initial Value	Identifier Comment...
VAR	ERROR	N/A [Auto]	BOOL	FALSE	
VAR	RUNNING	N/A [Auto]	BOOL	FALSE	
VAR	COMPLETE	N/A [Auto]	BOOL	FALSE	
VAR	BUZZER_OFF	N/A [Auto]	BOOL	FALSE	
VAR	RED	Y0.0	BOOL	FALSE	
VAR	YELLOW	Y0.1	BOOL	FALSE	
VAR	GREEN	Y0.2	BOOL	FALSE	
VAR	BUZZER	Y0.3	BOOL	FALSE	

شکل ۹-۶۳ وارد کردن سیمبول های محلی – مثال

POU بلوکی باید کد وضعیت را از برنامه اصلی به عنوان ورودی دریافت کرده و وضعیت سه لامپ و زنگ را به عنوان خروجی بلوک به برنامه اصلی بفرستد. با توجه به آنکه به ازای هر کد وضعیت، یک حالت برای سیستم روشنایی در نظر گرفته می‌شود، استفاده از دستور شرطی CASE منطقی به نظر می‌رسد.

Tower_Light

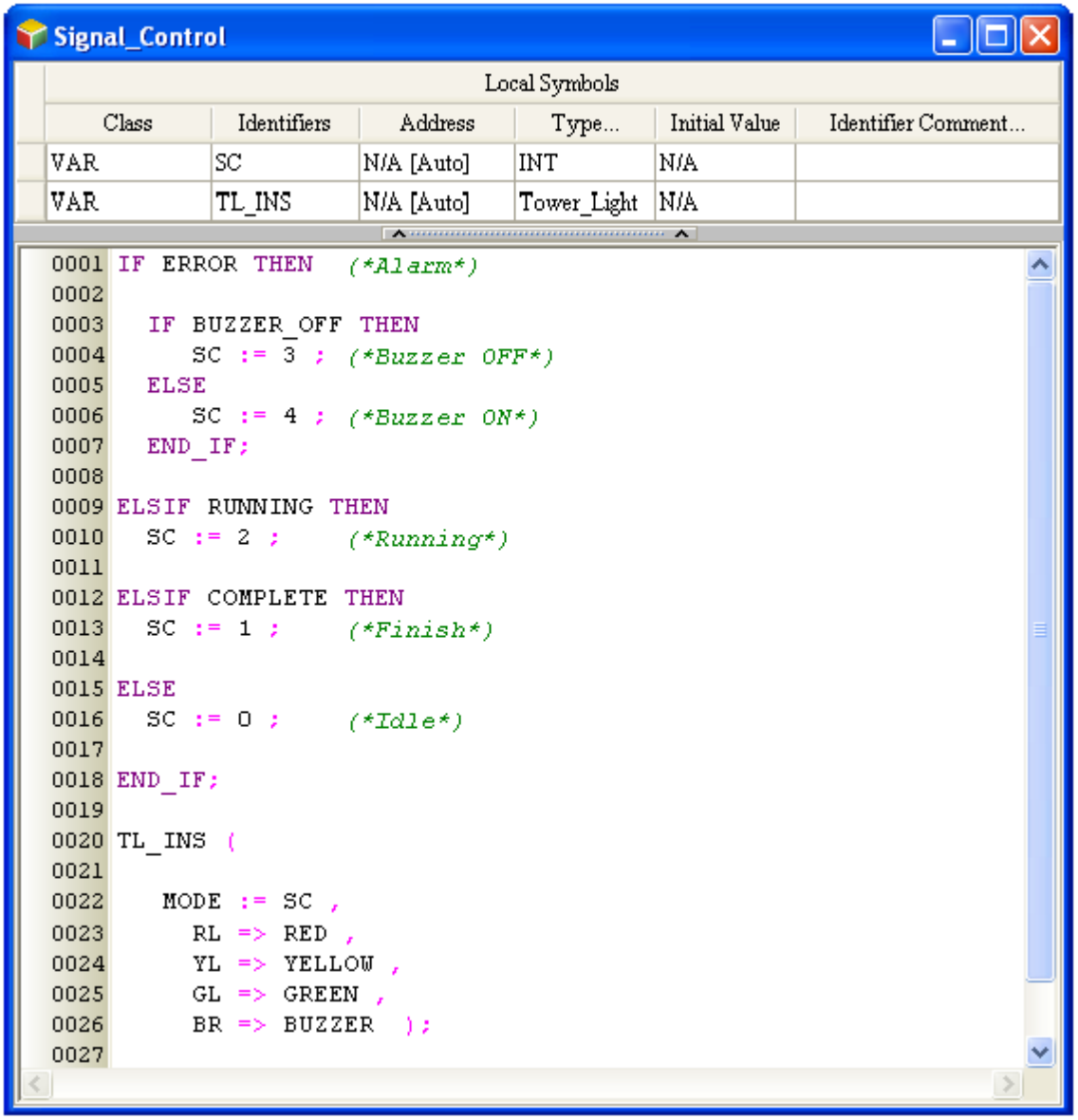
Local Symbols

Class	Identifiers	Address	Type...	Initial Value	Identifier Comment...
VAR_INPUT	MODE	N/A [Auto]	INT	N/A	
VAR_OUTPUT	RL	N/A [Auto]	BOOL	FALSE	
VAR_OUTPUT	YL	N/A [Auto]	BOOL	FALSE	
VAR_OUTPUT	GL	N/A [Auto]	BOOL	FALSE	
VAR_OUTPUT	BR	N/A [Auto]	BOOL	FALSE	

```
0001 CASE MODE OF
0002
0003 (* SM407:0.5S on/ 0.5S off *)
0004
0005    0:  RL := FALSE ;
0006        YL := TRUE  ;
0007        GL := FALSE ;
0008        BR := FALSE ;
0009
0010    1:  RL := FALSE ;
0011        YL := SM407 ;
0012        GL := FALSE ;
0013        BR := FALSE ;
0014
0015    2:  RL := FALSE ;
0016        YL := FALSE ;
0017        GL := TRUE ;
0018        BR := FALSE ;
0019
0020    3:  RL := SM407 ;
0021        YL := FALSE ;
0022        GL := FALSE ;
0023        BR := FALSE ;
0024
0025    4:  RL := TRUE ;
0026        YL := FALSE ;
0027        GL := FALSE ;
0028        BR := SM407 ;
0029
0030 END_CASE;
```

شکل ۹-۶۴ برنامه نوشته شده برای بلوک – مثال

در POU برنامه دو سیمبول SC را برای معرفی کُد وضعیت و سیمبول بلوک را تعریف کنیم. با توجه به آنکه کُد وضعیت با توجه به اتفاق افتادن و یا نیافتادن بعضی حالت‌ها و در نتیجه مقدار بیتی پرچم‌ها تعیین می‌شود، از دستور شرطی IF برای تعیین مقدار کد وضعیت استفاده می‌کنیم. سپس کد وضعیت را به عنوان ورودی به بلوک داده و از خروجی آن وضعیت سه لامپ را می‌خوانیم.



شکل ۹-۶۵ برنامه دوره ای اصلی – مثال

در نهایت نیز می‌توانیم برنامه را کامپایل و بر روی PLC دانلود و تست کنیم.

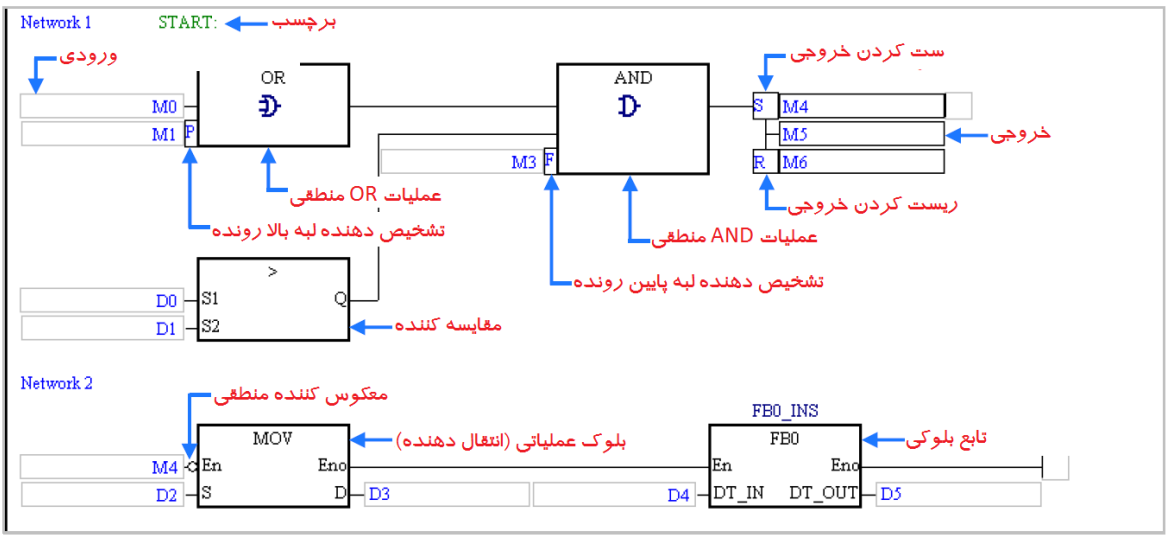
فصل ۱۰– زبان های برنامه نویسیFBD ، IL و SFC

در این فصل به مرور سه زبان برنامه نویسی FBD، IL و SFC خواهیم پرداخت. مرور این سه زبان برنامه نویسی با توجه به مخاطبان این کتاب و همچنین تشابه توضیحات این زبان‌های برنامه نویسی ، با جزئیات کمتری نسبت به دو فصل قبل انجام خواهد پذیرفت. برای اطلاعات بیشتر در مورد آن‌ها می‌توانید به فایل‌های راهنمای محصولات کمپانی دلتا مراجعه کنید.

۱۰–۱– زبان برنامه نویسی FBD

*** توجه: با آنکه در مستندات رسمی کمپانی دلتا تا زمان انتشار این کتاب بحثی پیرامون سازگاری زبان‌های برنامه نویسی با PLCهای مختلف مطرح نشده است ولی به نظر می‌رسد زبان برنامه نویسی ST و FBD تنها در PLCهای سری AH500 قابل استفاده هستند. این موضوع در فروم‌های اینترنتی کاربران دلتا نیز مطرح شده است.

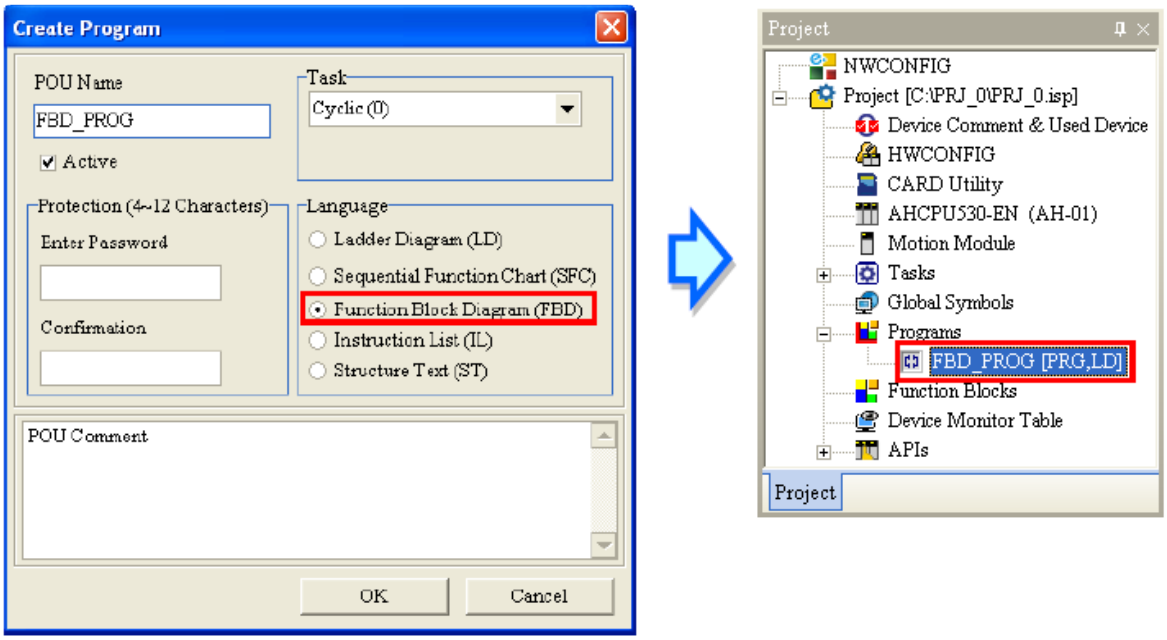
زبان برنامه نویسی FBD همانند زبان برنامه نویس Ladder زبانی گرافیکی است که اساس آن مطابق مدارات منطقی (ساخته شده با AND، OR و NOT) است. از این رو کار با آن برای مهندسین کامپیوتر و برقی که با این المان‌های منطقی آشنایی دارند به راحتی ممکن است. المان‌های اصلی این زبان برنامه نویسی در محیط ISPSoft در شکل زیر نشان داده شده است.



شکل ۱۰-۱ نمایی از محیط برنامه نویسی به زبان FBD

۱۰-۱-۱- محیط برنامه نویسی

برای شروع برنامه نویسی نیاز به اضافه کردن محیط برنامه نویسی جدید است که برای اینکار در بخش مدیریت پروژه بر روی Program کلیک راست کرده و POU و سپس New را انتخاب می‌کنیم. در پنجره باز شده پس از تعیین نام پروژه زبان برنامه نویسی را Function Block Diagram (FBD) انتخاب می-کنیم.



شکل ۱۰-۲ انتخاب زبان برنامه نویسی FBD

پس از تایید POU[۱] مورد نظر ما برای نوشتن برنامه به عنوان زیرشاخه‌ای از Program در بخش مدیریت پروژه اضافه می‌شود. در این مرحله صفحه‌ای برای ویرایش برنامه باز خواهد شد.

[۱] Program Organization Unit

شکل ۱۰-۳ محیط برنامه نویسی (ویرایش برنامه) با FBD

در این مرحله نوار ابزار مرتبط با زبان برنامه نویسی FBD نیز ظاهر شده و می‌توانیم از آن در برنامه نویسی استفاده کنیم. آیکون های این نوار ابزار را در ادامه مرور خواهیم کرد.

شکل ۱۰-۴ نوار ابزار برنامه نویسی FBD

جدول ۱۰-۱: آیکون های نوار ابزار برنامه نویسی FBD

نماد	کلید میانبر	شرح
	Shift+Ctrl+C	نمایش/عدم نمایش توضیحات در networks
		نمایش/ عدم نمایش توضیحات حافظه ها
	Shift+Ctrl+A	فعال/غیر فعال سازی network انتخاب شده
	Shift+Ctrl+B	نشانه گذاری و یا پاک کردن نشانه‌ها در network-ها
	Shift+Ctrl+P	رفتن به نشان بعدی
	Shift+Ctrl+N	رفتن به نشان قبلی
	Ctrl+l	اضافه کردن network بالای network انتخاب شده
	Shift+Ctrl+l	اضافه کردن network پایین network انتخاب شده

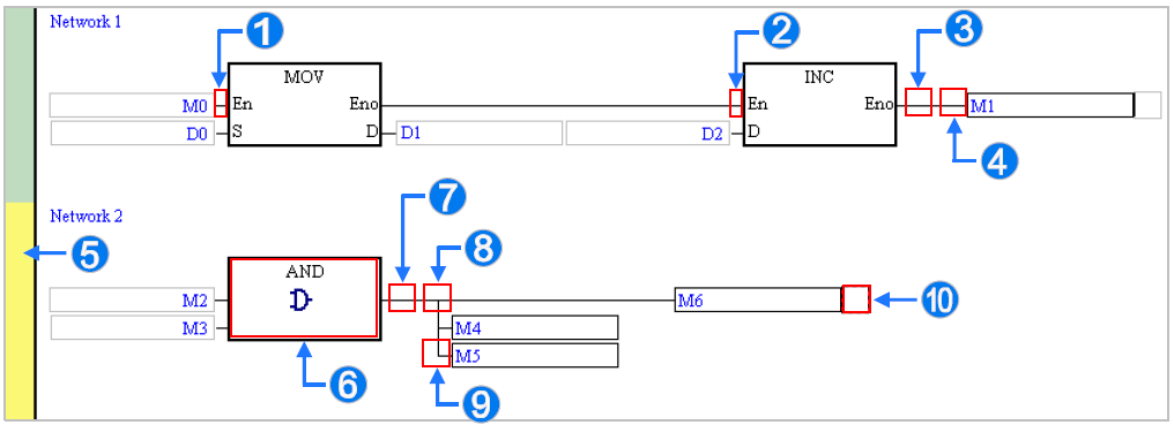
	Shift+F1	اضافه کردن AND
	Shift+F2	اضافه کردن OR
	Shift+F3	اضافه کردن خروجی
	Shift+F4	اضافه کردن پایه ورودی به بلوکی AND/OR
	Shift+F5	حذف کردن پایه ورودی از بلوکی AND/OR
	Shift+F6	معکوس کردن منطق (= اضافه کردن NOT)
	Shift+F7	ست و یا ریست کردن ورودی
	Shift+F8	تشخیص دهنده لبه
	Shift+Ctrl+U	اضافه کردن بلوک (نوع آن را پس از کلیک کردن بر روی این آیکون باید تعیین کنید)

۱۰–۱–۲– Network در FBD

برنامه‌ها در زبان برنامه نویسی FBD از بخش‌هایی به نام Network تشکیل شده‌اند، این بخش‌ها شامل قسمت‌های مستقلی از برنامه هست. در ISPSoft محدودیتی برای تعداد اجزای هر Network وجود ندارد و به همین دلیل می‌توان حتی تمامی برنامه را در یک Network نوشت با این حال برای اینکه برنامه خواناتر و منظم‌تر باشد، استفاده از Networkها ضروری است. با توجه به تشابه توضیحات Network در دو زبان Ladder و FBD از آوردن توضیحات بیشتر در مورد آن خودداری می‌کنیم.

۱۰–۱–۳– انتخاب اجزای برنامه

با انتخاب اجزای محیط برنامه نویسی FBD می‌توان اقدام به ویرایش و اصلاح آن‌ها کرد. اجزای مجازی که در صفحه برنامه FBD امکان انتخاب آن‌ها وجود دارد، در شکل زیر آمده‌اند.



شکل ۱۰-۵ انتخاب اجزای صفحه ویرایش برنامه FBD

1 گره ورودی قبل این موقعیت(در اینجا M0) انتخاب شده است.

2 بلوک قبل این موقعیت انتخاب شده است.

3 تمامی بلوک‌های قبل این موقعیت انتخاب شده‌اند.

4 گره خروجی پس از این موقعیت انتخاب شده است.

5 انتخاب تمام Network

6 انتخاب بلوک

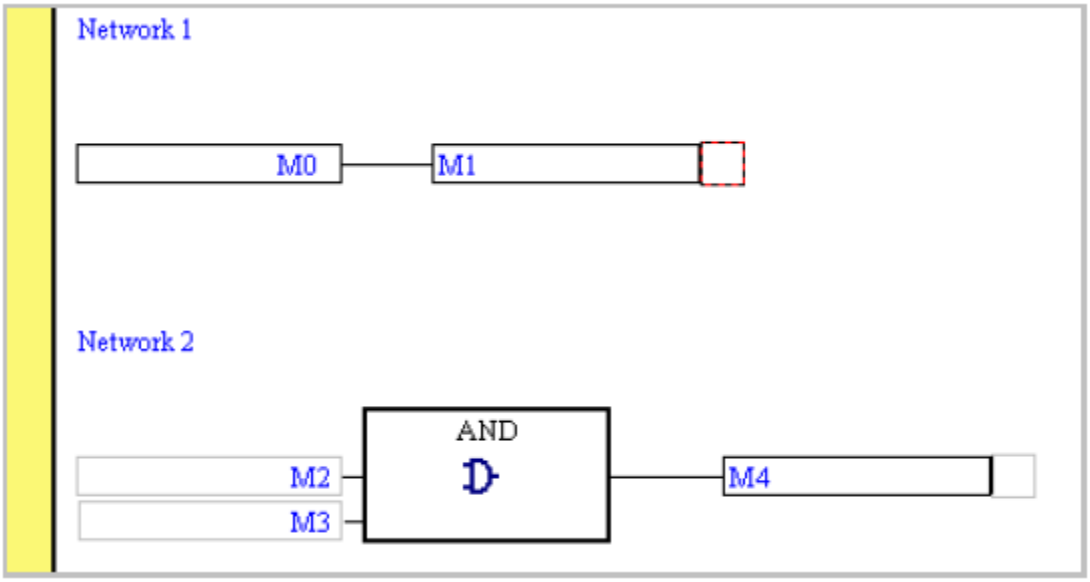
7 بلوک قبل این موقعیت انتخاب شده است.

8 گره‌های خروجی زیر این موقعیت انتخاب شده‌است.

9 گره‌های خروجی سمت راست این موقعیت انتخاب شده است.

10 بلوک‌های قبل این موقعیت انتخاب شده‌اند.

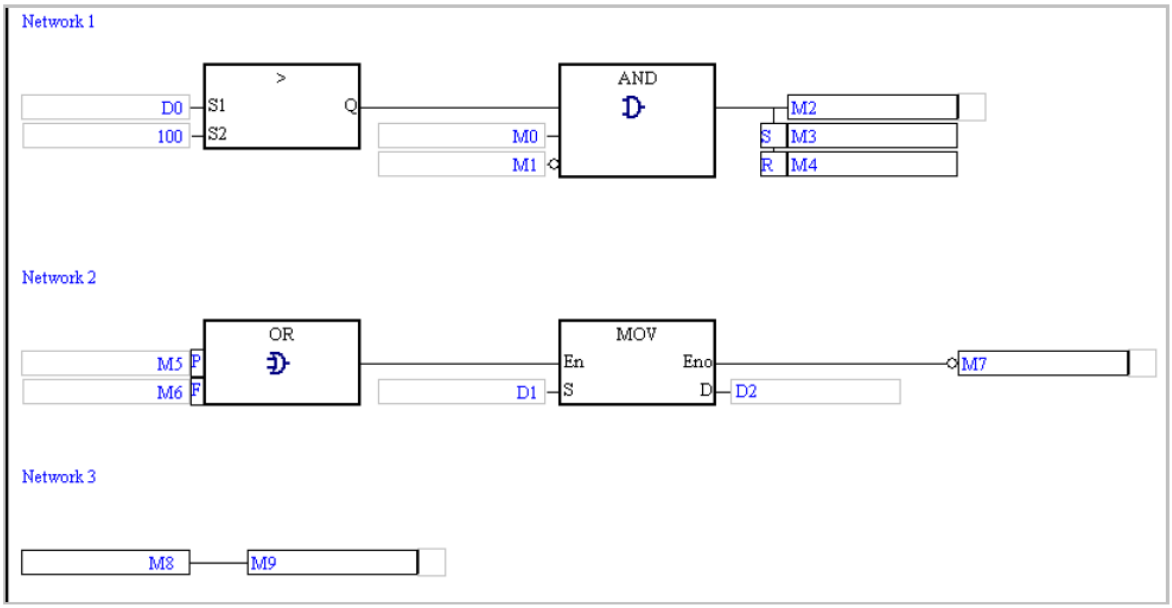
همچنین با نگه داشتن کلید Ctrl در صفحه کلید می‌توان نسبت به انتخاب چند network همزمان اقدام کرد. با استفاده از کلید Shift نیز می‌توان محدوده‌ای از Network ها را انتخاب کرد، برای اینکار ابتدا Shift را نگه داشته و سپس Network ابتدایی و بعد انتهایی را انتخاب نمایید.



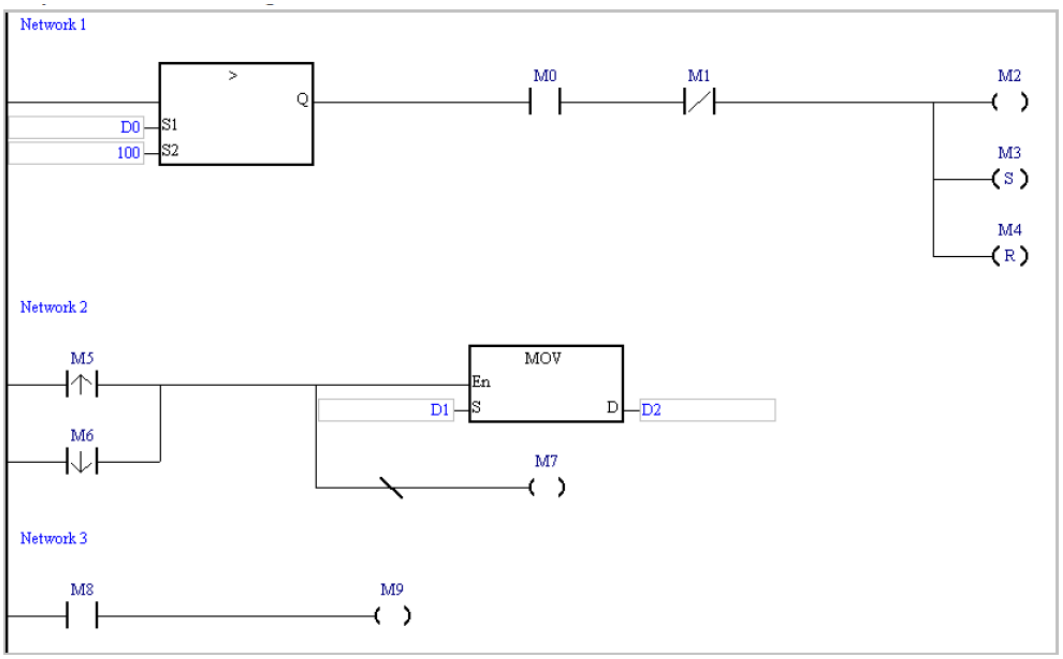
شکل ۱۰-۶ انتخاب همزمان چند Network در FBD

۱۰-۱-۴- تابع بلوکی

ساخت تابع بلوکی با استفاده از FBD نیز دقیقا مشابه با زبان Ladder است، تنها می‌بایست از بلوک-های مربوط به این زبان برنامه نویسی استفاده شود. یک نمونه از برنامه تابع بلوکی به زبان FBD و معادل برنامه Ladder آن در زیر آمده است.



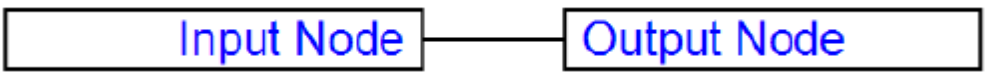
شکل ۱۰-۷ برنامه تابع بلوکی به زبان FBD



شکل ۱۰-۸ برنامه تابع بلوکی معادل به زبان Ladder

۱۰-۱-۴-۱- گره‌های ورودی و خروجی

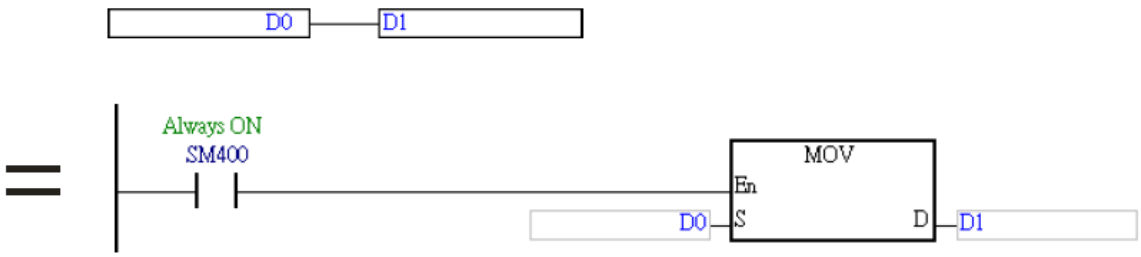
برخلاف زبان Ladder که برای معرفی وضعیت ورودی و خروجی و حافظه‌ها نیاز به کانتکت و یا کویل داشتیم، در زبان FBD صرفا از نام ورودی، خروجی و حافظه‌ها برای خواندن و نوشتن استفاده می‌کنیم. این کار برنامه نویسی را کوتاه‌تر می‌کند.



شکل ۱۰-۹ انتقال مستقیم ورودی به خروجی در زبان FBD

برای مثال دو نمونه انتقال وضعیت بیت و داده و معادل Ladder آن را می‌بینیم.

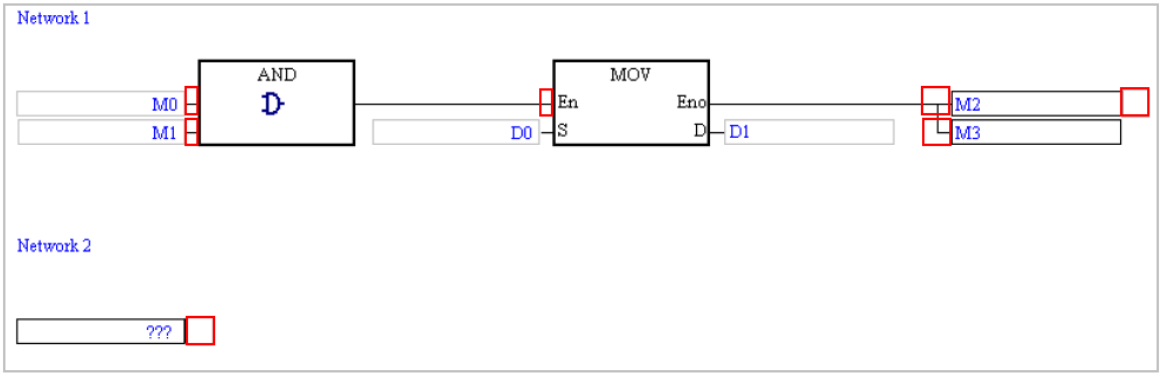
شکل ۱۰-۱۰ انتقال بیتی در دو زبان FBD و Ladder



شکل ۱۰-۱۱ انتقال داده در دو زبان FBD و Ladder

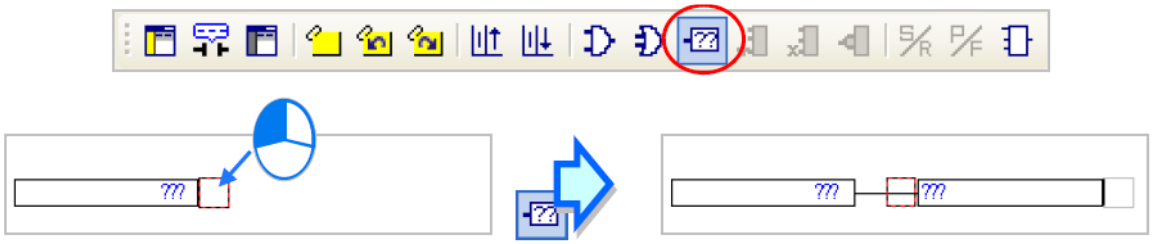
در این انتقال داده‌ها باید توجه کرد که نوع داده ورودی با خروجی متفاوت نباشد، طول خروجی از ورودی کمتر نباشد و اعداد حقیقی نیز تنها می‌توانند به حافظه‌های با فرمت مشابه خود منتقل شوند.

برای اضافه کردن گره خروجی، موقعیتی را که می‌خواهید گره مورد نظر را در آن قرار دهید را انتخاب کنید، موقعیت‌های مشخص شده در شکل زیر موقعیت‌های مجاز به حساب می‌آیند.



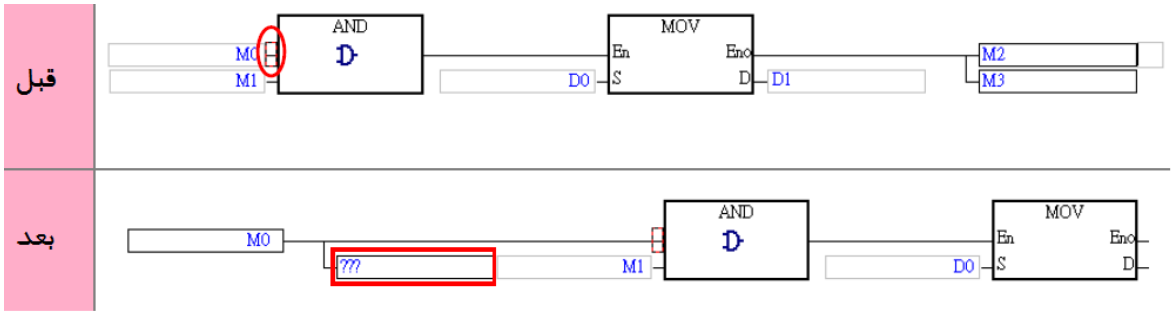
شکل ۱۰-۱۲ موقعیت های مجاز برای انتخاب گره خروجی در FBD

سپس با کلیک بر روی  ، گره خروجی در مکان مورد نظر ظاهر می‌شود.

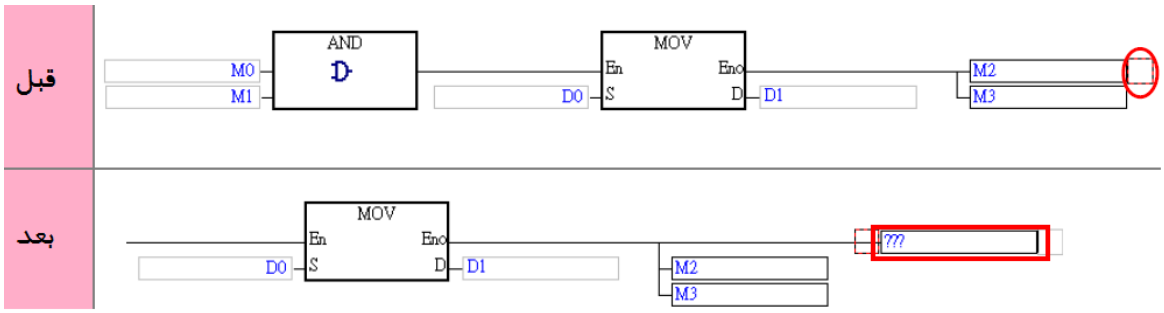


شکل ۱۰-۱۳ گره خروجی در FBD

در شکل‌های زیر می‌توانید چند نمونه تخصیص گره خروجی را ببینید.



شکل ۱۰-۱۴ اختصاص گره خروجی به ورودی گیت های AND و OR در FBD



شکل ۱۰-۱۵ اضافه کردن گره خروجی به سمت راست گره خروجی دیگر در FBD

توجه شود که گره های خروجی را در ادامه پایه‌های خروجی توابع بلوکی نمی‌توان اضافه کرد.

۱۰-۱-۵- بلوک های منطقی زبان FBD

بسیاری از مباحث زبان FBD مانند تخصیص سیمبول و تخصیص ورودی و خروجی، اضافه کردن توضیحات و برچسب و فعال و غیرفعال سازی Networkها همانند زبان Ladder است به همین جهت با صرف نظر کردن از مباحث مشترک به تشریح بلوک‌های خاص زبان FBD در این بخش خواهیم پرداخت. دانش آموختگان علوم کامپیوتر و برق به علت آشنایی با مدارات منطقی به راحتی می‌توانند با منطق زبان برنامه نویسی FBD آشنا شده و آن را به کار ببرند.

توابع منطقی AND و OR

ورودی این دو تابع (و یا گیت[۱]) منطقی باید از نوع بیتی (BOOL) و یا سیمبولی به فرمت آن باشد. این بلوک‌ها حداکثر تا ۳۲ ورودی را پشتیبانی می‌کنند.

تابع منطقی AND هرگاه تمامی ورودی‌های آن یک باشد، خروجی آن برابر یک منطقی می‌شود. این عملکرد به ازای هر تعداد پایه ورودی صحیح است.جدول درستی این تابع به صورت زیر است.

جدول ۱۰-۲: جدول درستی گیت AND

پایه اول	پایه دوم	خروجی AND
0	0	0
1	0	0
0	1	0

[۱] Gate

1	1	1

تابع منطقی OR هرگاه تنها یکی از ورودی‌های آن یک باشد، خروجی آن برابر یک منطقی می‌شود. این عملکرد به ازای هر تعداد پایه ورودی صحیح است. جدول درستی این تابع به صورت زیر است.

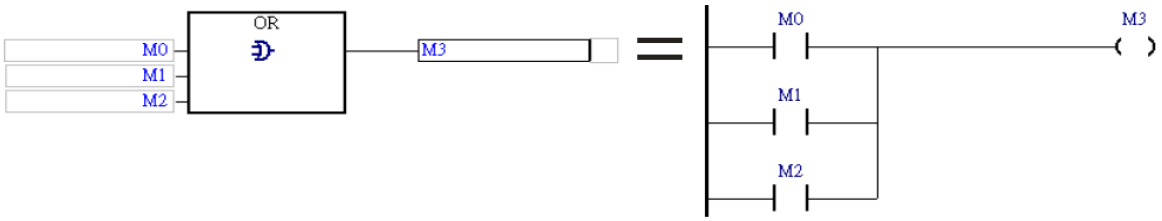
جدول ۱۰-۳: جدول درستی گیت OR

پایه اول	پایه دوم	خروجی OR
0	0	0
1	0	1
0	1	1
1	1	1

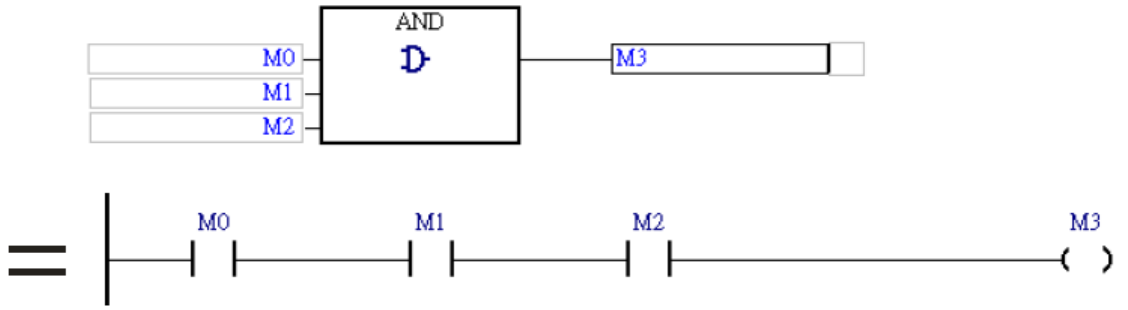
نمای گرافیکی این دو گیت در زیر آمده است.

شکل ۱۰-۱۶ گیت OR (راست) و گیت AND (چپ)

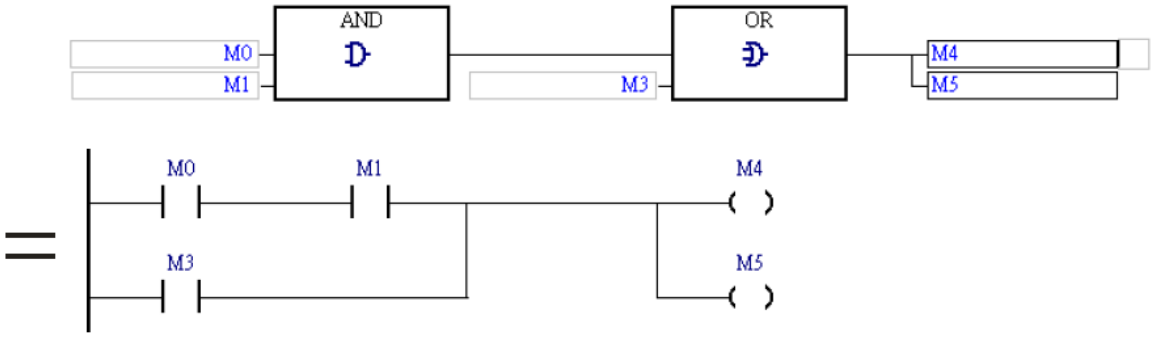
بنابر منطق شرح داده شده برای این دو بلوک مشخص است که بلوک AND همانند کانتکت‌های سری و بلوک OR همانند کانتکت‌های موازی عمل می‌کند. در زیر دو مثال از این دو بلوک و معادل Ladder آن آورده شده است.



شکل ۱۰-۱۷ معادل گیت OR در زبان Ladder

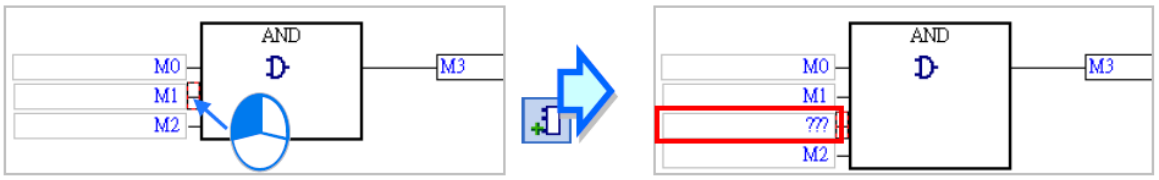


شکل ۱۰-۱۸ معادل کانتکت AND در زبان Ladder

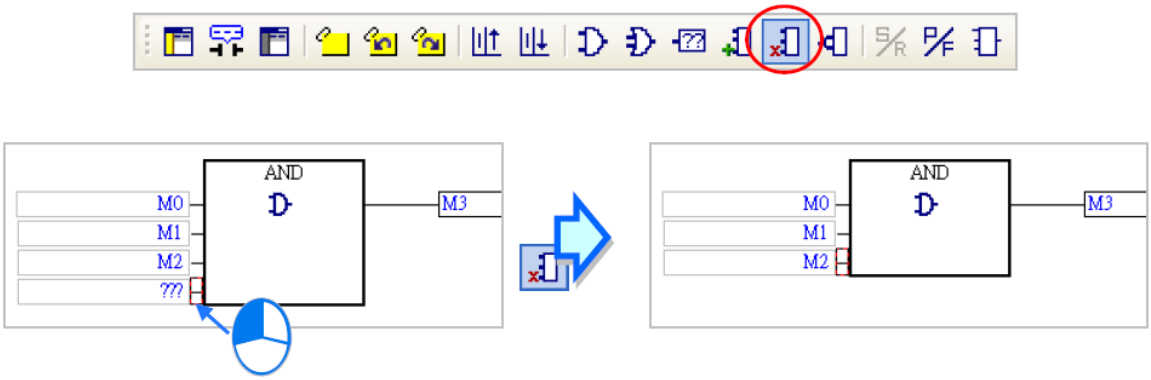


شکل ۱۰-۱۹ معادل Ladder برنامه ای به زبان FBD

برای به کار بردن این دو بلوک کافی است موقعیت مورد نظر در برنامه را انتخاب و بر روی علامت AND یعنی  و یا علامت OR یعنی  کلیک نمایید. برای اضافه کردن ورودی به این گیت‌ها نیز می‌توان با انتخاب یکی از پایه‌های ورودی آن و کلیک بر روی  پایه‌ای به زیر پایه انتخاب شده اضافه کرد. برای حذف پایه‌های ورودی نیز می‌توان از  استفاده کرد.



شکل ۱۰-۲۰ اضافه کردن پایه ورودی گیت



شکل ۱۰-۲۱ کم کردن پایه ورودی گیت

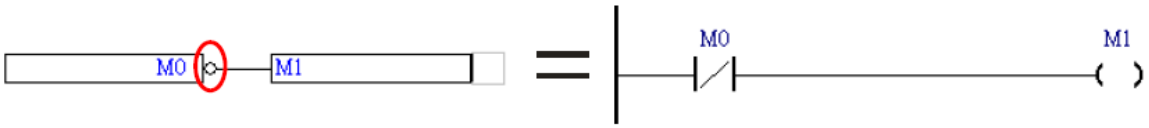
منطق معکوس

با استفاده از منطق معکوس (گیت NOT) می‌توان مقدار بولی را معکوس کرد. این عملیات برای پورت‌های داده امکان پذیر نیست.

جدول ۱۰-۴: جدول درستی گیت NOT

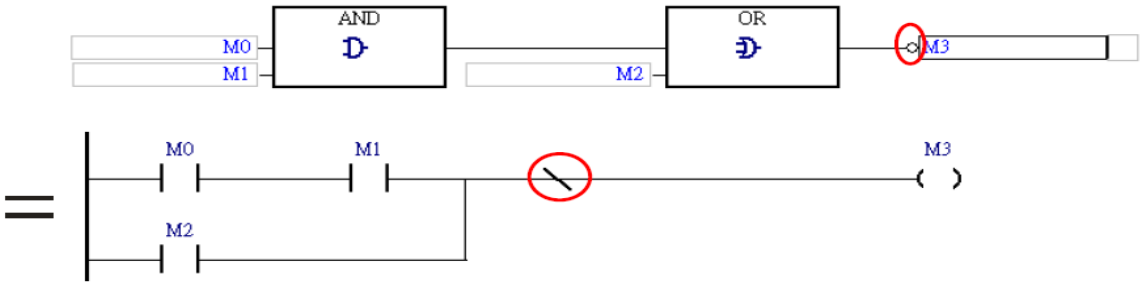
ورودی	خروجی NOT
1	0
0	1

مثال های از عملکرد این گیت و معادل Ladder آن‌ها را می‌بینیم.



شکل ۱۰-۲۲ معادل Ladder عملکرد گیت NOT

در مثال زیر نمونه دیگری از برنامه شامل سه گیت AND و OR و NOT و معادل Ladder آن آمده است.

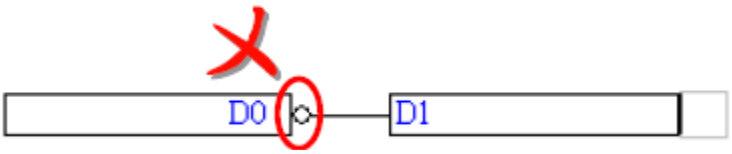


شکل ۱۰-۲۳ معادل Ladder برنامه ای به زبان FBD- نمونه دوم

در مثال زیر با استفاده از گیت NOT بین پایه EnO و En دو بلوک می‌خواهیم حداکثر یکی از آن‌ها اجرا شوند.

شکل ۱۰-۲۴ اجرای یکی از توابع بلوکی Black و یا White

توجه شود که همانطور که گفته شده، گیت های منطقی مخصوص عملیات‌های بیتی می‌باشند و برای داده‌ها به صورت مستقیم قابل استفاده نمی‌باشند.



شکل ۱۰-۲۵ از گیت NOT برای داده های غیر بیتی نمی‌توان استفاده کرد.

برای استفاده از گیت NOT در برنامه نویسی، بر روی مکان مورد نظرمان کلیک کرده و از استفاده می‌کنیم. با کلیک دوباره بر روی همین گزینه گیت NOT حذف می‌شود.

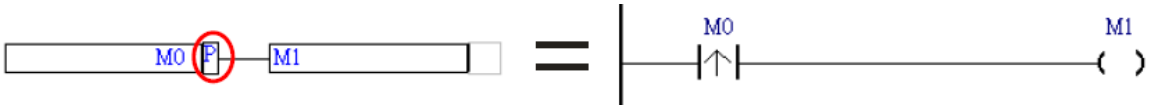
شکل ۱۰-۲۶ اضافه و حذف کردن گیت NOT.

تشخیص دهنده‌های لبه

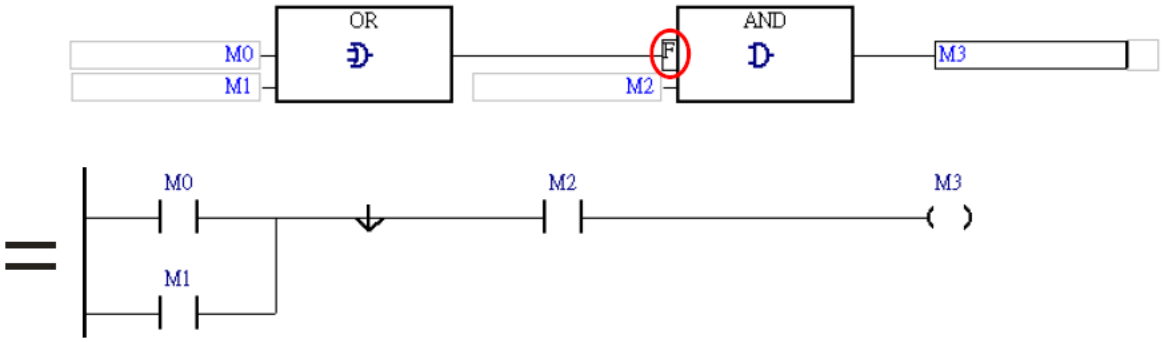
با استفاده از تشخیص دهنده لبه بالارونده P (تشخیص دهنده لبه پایین رونده F) می‌توان لبه بالارونده سیگنال بیتی را تشخیص داد و خروجی این گیت برای یک زمان اسکن یک خواهد شد. تشخیص دهنده های فوق تنها می‌توانند در سر راه سیگنال‌های بیتی قرار گیرند.

شکل ۱۰-۲۷ تشخیص دهنده لبه پایین رونده F (راست) و تشخیص دهنده لبه بالارونده P (چپ).

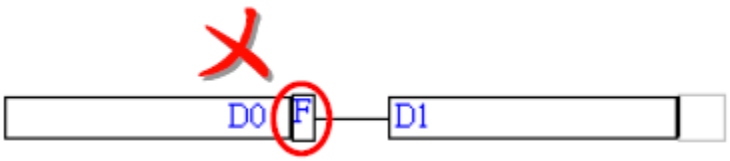
در ادامه مثال‌های از عملکرد این دو تشخیص دهنده لبه و معادل Ladder آن‌ها را مشاهده می‌کنیم.



شکل ۱۰-۲۸ تشخیص دهنده لبه بالا رونده در زبان FBD و معادل Ladder آن

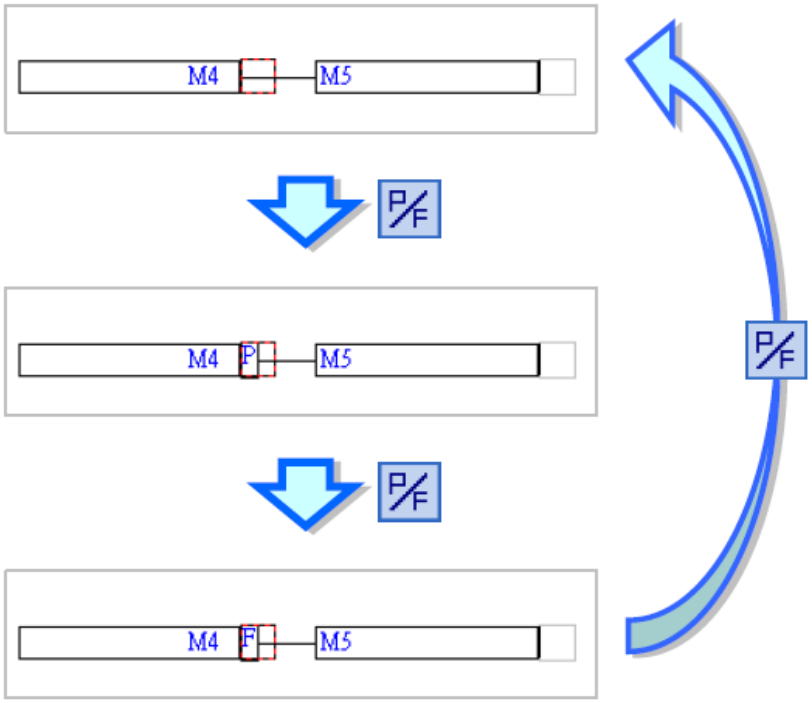


شکل ۱۰-۲۹ نمونه برنامه به زبان FBD که در آن از تشخیص دهنده لبه استفاده شده و معادل Ladder آن.



شکل ۱۰-۳۰ تشخیص دهنده های لبه تنها برای داده های نوع بیتی قابل استفاده هستند.

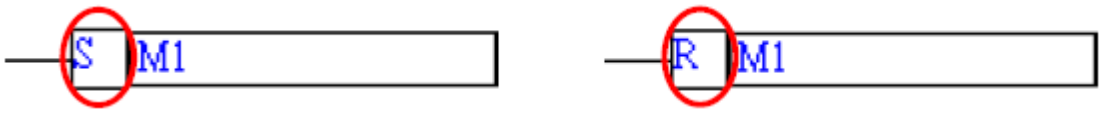
برای استفاده از تشخیص دهنده‌های لبه ابتدا بر روی مکان مورد نظر در برنامه کلیک کرده و سپس با استفاده از  تشخیص دهنده P در مکان مورد نظر ظاهر می‌شود، در صورتی که بخواهیم از تشخیص دهنده F استفاده کنیم، باید یکبار دیگر بر روی  کلیک کنیم، و در صورتی که بخواهیم تشخیص دهنده لبه را حذف کنیم باید برای بار سوم بر روی  کلیک نماییم.



شکل ۱۰-۳۱ اضافه کردن تشخیص دهنده‌های لبه به برنامه

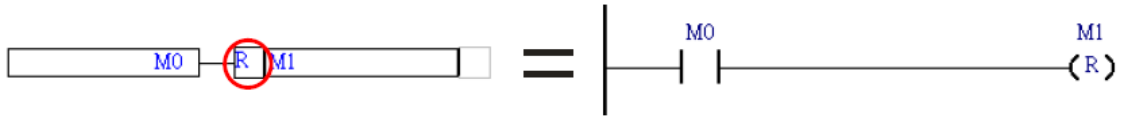
ست و ریست کردن خروجی‌ها

برای ست و ریست کردن خروجی‌های بیتی می‌توانیم از S و R استفاده کنیم.

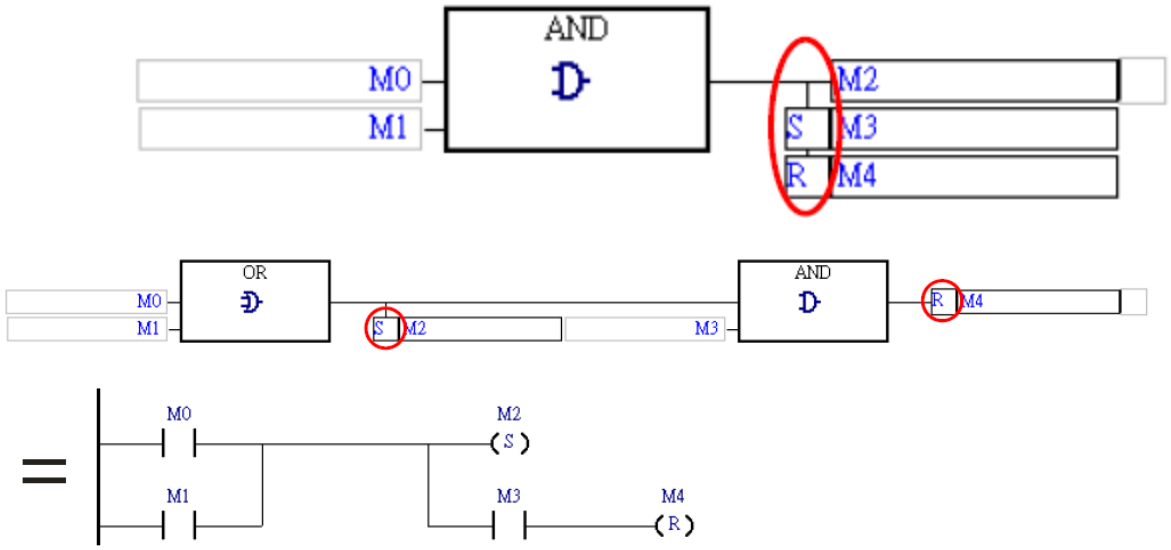


شکل ۱۰-۳۲ ریست کننده خروجی بیتی (راست)، ست کننده خروجی بیتی (چپ).

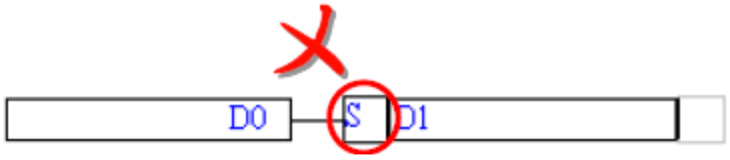
در ادامه مثال‌هایی از عملکرد S و R و معادل Ladder آن‌ها را مشاهده می‌کنیم.



شکل ۱۰-۳۳ معادل Ladder ریست کننده خروجی بیتی

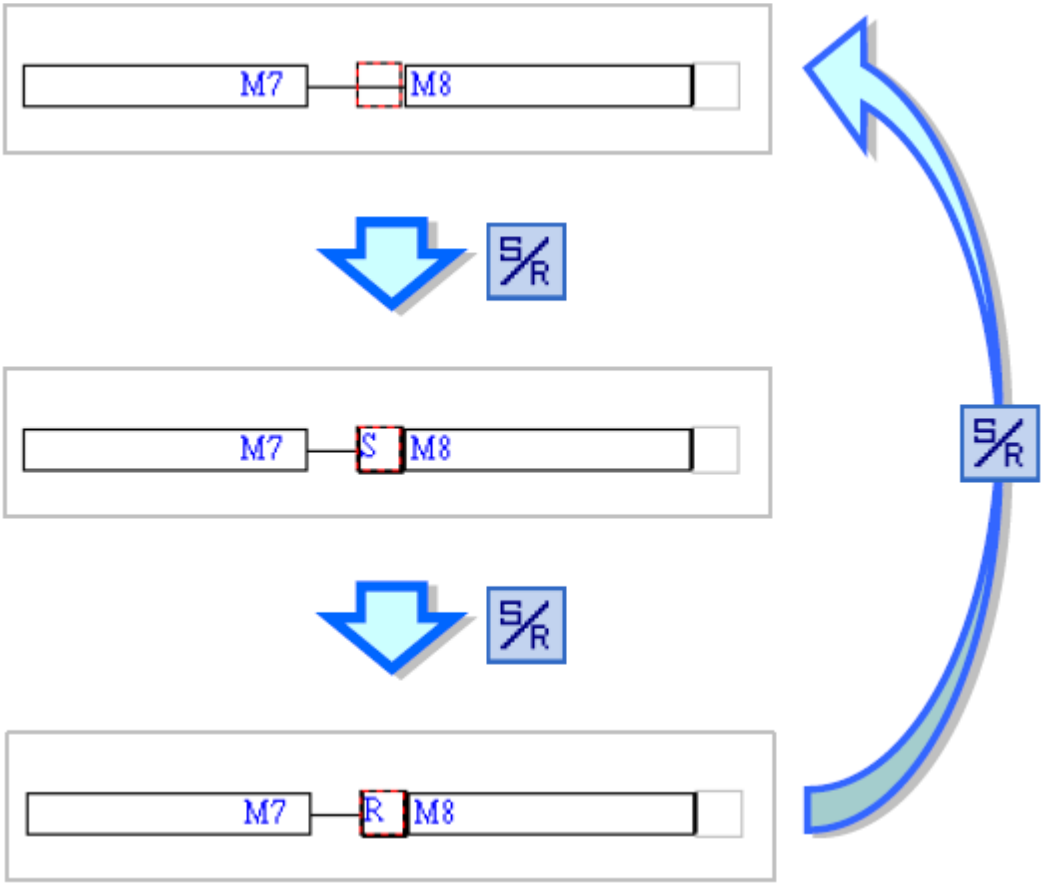


شکل ۱۰-۳۴ معادل Ladder برنامه ای که در آن از ست و ریست کننده خروجی بیتی استفاده شده است.



شکل ۱۰-۳۵ از ست و ریست کننده خروجی بیتی نمی توان برای داده های غیربیتی استفاده کرد.

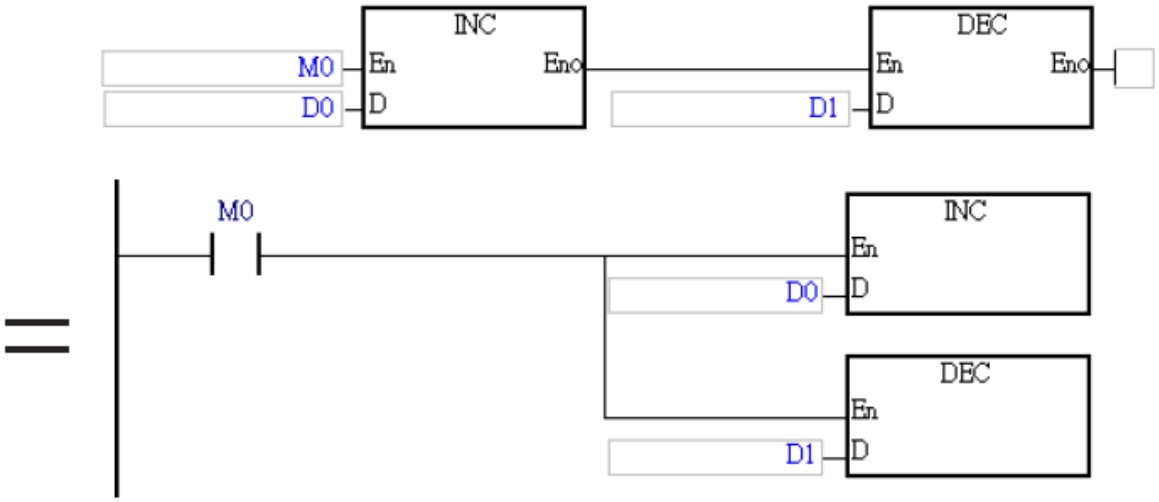
برای استفاده از S و R ابتدا بر روی مکان مورد نظر کلیک کرده و سپس با استفاده از  ست کننده در مکان مورد نظر ظاهر می‌شود، در صورتی که بخواهیم از ریست کننده استفاده کنیم، باید یکبار دیگر بر روی  کلیک کنیم، و در صورتی که بخواهیم آن‌ها را حذف کنیم باید بار دیگر بر روی  کلیک نماییم.



شکل ۱۰-۳۶ اضافه و حذف ست و ریست کننده خروجی های بیتی

توابع بلوکی و دیگر بلوک ها

استفاده (فراخوانی و به کارگیری) از توابع بلوکی و دیگر انواع بلوک‌ها در زبان FBD مشابه زبان Ladder است. همچنین اینکه بلوک‌ها به چه زبانی نوشته شده باشند مهم نیست و از آن‌ها می‌توان در تمامی زبان‌های برنامه نویسی استفاده کرد.



شکل ۱۰-۳۷ استفاده از توابع بلوکی در دو زبان برنامه نویسی Ladder و FBD به یک شکل است.

۱۰–۲– زبان برنامه نویسی IL

زبان برنامه نویسی Instruction List (IL) یک زبان سطح پایین (نزدیک به زبان ماشین) مشابه زبان اسمبلی[۱] است که آن را در این بخش به صورت مختصر مرور خواهیم کرد. به علت تشابه بخش‌های مختلف برنامه نویسی به این روش با روش ST از تشریح جزئیات این روش برنامه نویسی خودداری می-کنیم.

زبان IL تشکیل شده از مجموعه عباراتی است که هر کدام معرف عملکردی هستند. عملکردهایی که به وسیله زبان Ladder می‌توان نوشت را هرکدام به وسیله زبان IL نیز می‌توان پیاده کرد. در مثال زیر پنج خط IL و معادل Ladder آن آورده شده است.

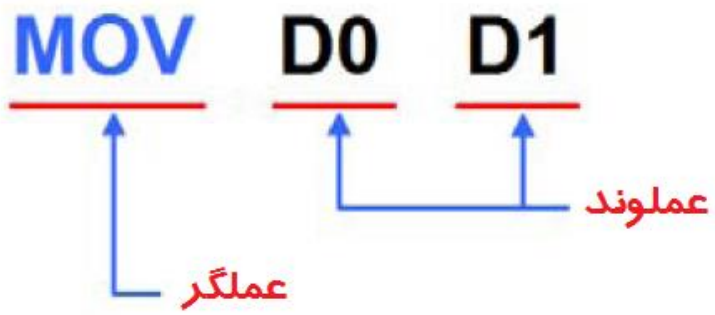
[۱] Assembly

شکل ۱۰-۳۸ پنج خط برنامه به زبان IL و معادل Ladder آن.

عملوند[۱] و عملگرها[۲] از المان‌های پایه‌ای دستورات IL به حساب می‌آیند. هر عملیاتی دارای مفعول یا عملگری است که عملیات بر روی آن‌ها انجام می‌شود، عملگر می‌تواند حافظه، سیمبول و یا مقداری ثابت باشد. همچنین نوع عملیات توسط عملگرها مشخص می‌شود. در نوشتار عملوندها و عملگرها با فاصله از یکدیگر جدا می‌شوند. برای مثال در شکل زیر عملگر MOV اشاره به فرایند انتقال داده دارد و در آن داده از عملوند D0 به D1 منتقل می‌شود.

[۱] Operand

[۲] Operator



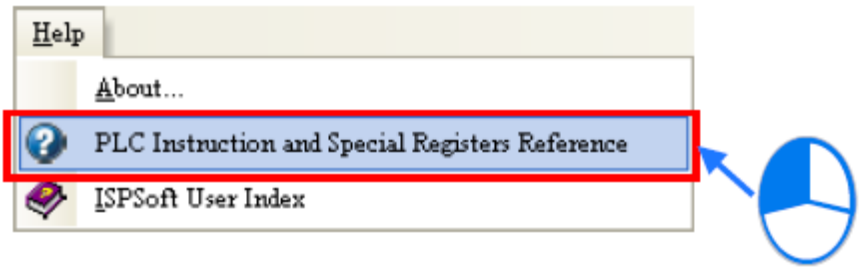
شکل ۱۰-۳۹ عملوندها و عملگرها در زبان برنامه نویسی IL

برای کامنت گزاری در IL می‌توان از فرمت (*کامنت*) استفاده کرد. کامنت‌ها را باید طوری در برنامه نوشت که آن را خوانا کرده و ساختار شرح داده شده در بالا را به هم نریزد. کامنت‌ها حین کامپایل برنامه به صورت اتوماتیک توسط سیستم لحاظ نخواهند شد. مثال‌های زیر نمونه‌های مجاز کامنت‌گذاری در IL هست.

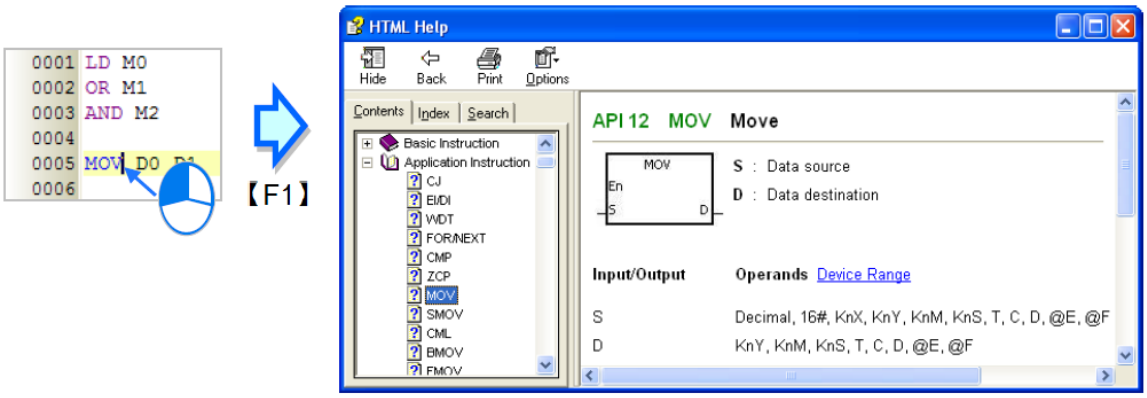
```
0001 LD M0  (*Note*)
0002 (*Note*)   OR M1
0003 AND    (*Note*)   M2
0004 OUT M3
0005 (*Note 1
0006   NOte 2
0007   Note 3*)
```

شکل ۱۰-۴۰ نمونه های مجاز کامنت گذاری در IL

برای دیدن توضیحات بیشتر در مورد عملگرها می‌توانید از سربرگ Help گزینه PLC Instruction and Special Registers Reference را انتخاب و یا پس از تایپ عملگر F1 را بفشارید.



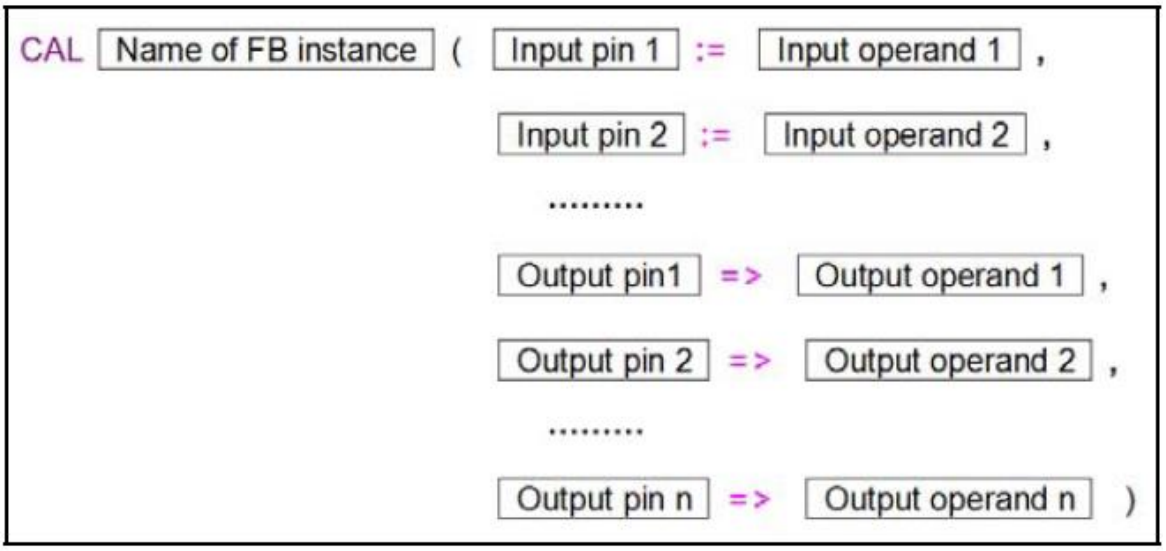
شکل ۱۰-۴۱ راهنمای عملگرها



شکل ۱۰-۴۲ استفاده از F1 برای باز شدن راهنمای عملگرها

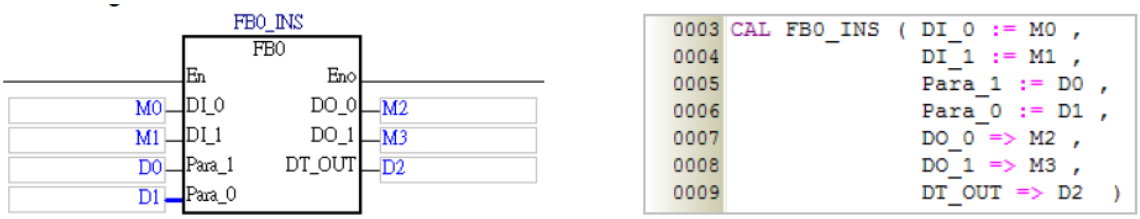
۱۰-۲-۱- فراخوانی توابع

برای فراخوانی توابع می‌توان از الگوی زیر استفاده کرد.



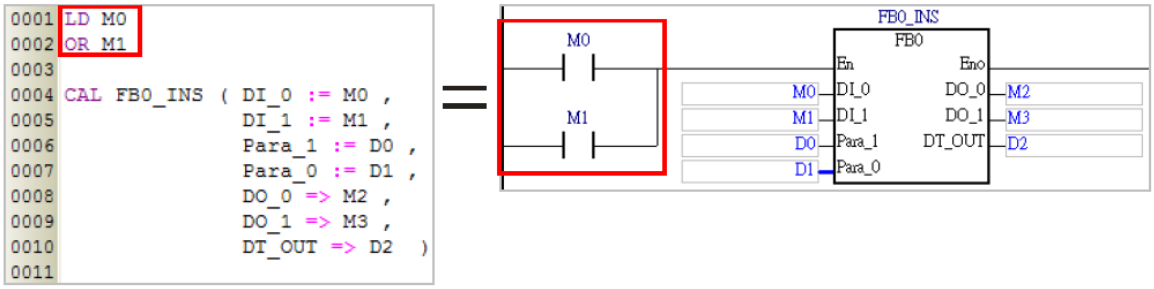
شکل ۱۰-۴۳ فراخوانی توابع در IL

در الگوی فوق CAL، نام نسخه تابع بلوکی و پرانتز سمت چپ باید در خط اول بیایند ولی اجزای دیگر می‌توانند در خطوط بعدی لیست شوند. ترتیب ورودی و خروجی‌ها باید مطابق مرجع تابع بلوکی رعایت شود (ورودی/خروجی‌ها ورودی در نظر گرفته می‌شوند) البته نیازی به در نظر گرفتن En و Eno نیست.



شکل ۱۰-۴۴ معادل Ladder برنامه IL در زمان فراخوانی تابع بلوکی

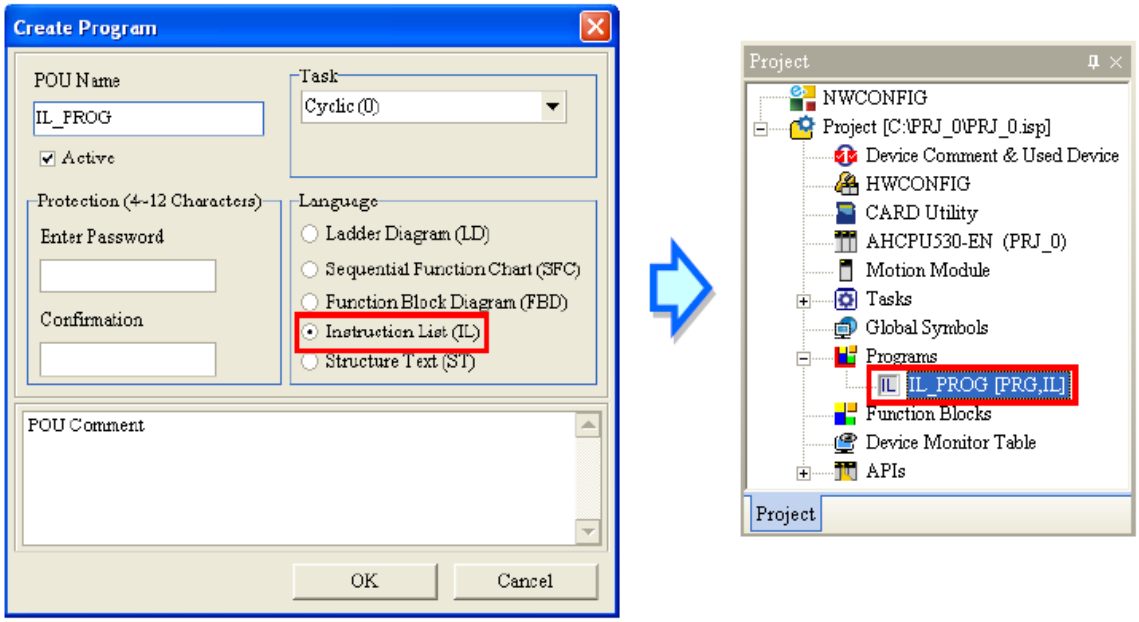
با آنکه En در دستورات فوق در نظر نگرفته شده ولی با توجه به آنکه اجرای CAL به نتیجه عملیات منطقی قبل خود وابسته است، می‌توان مانند مثال زیر معادلی برای En ایجاد کرد.



شکل ۱۰-۴۵ به کارگیری معادل پایه En با توجه به نتیجه عملیات بیتی قبل فراخوانی تابع بلوکی.

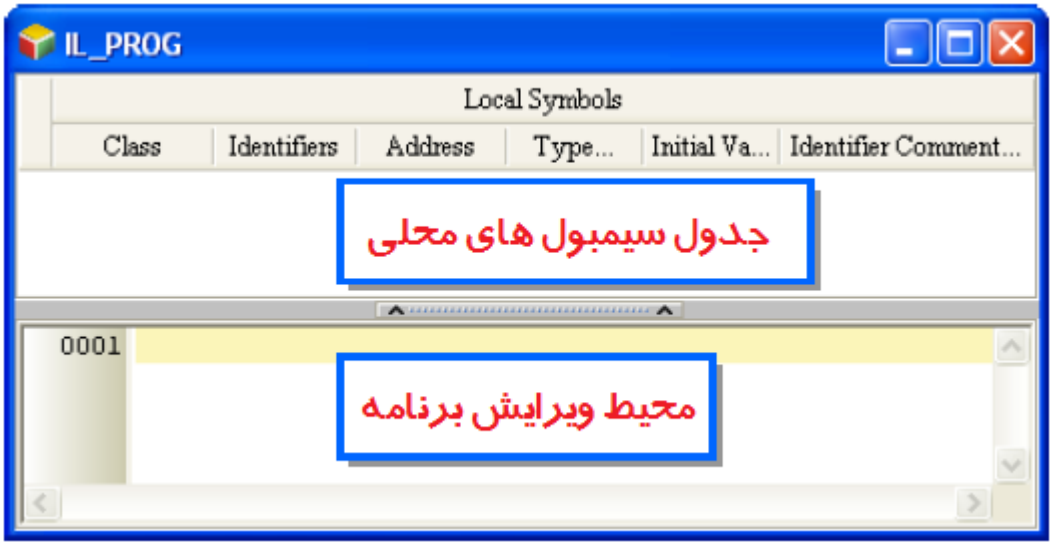
۱۰-۲-۲- ساخت پروژه

برای نوشتن برنامه‌ای جدید، باید در صفحه ایجاد برنامه زبان برنامه نویسی آن را IL انتخاب کرد.



شکل ۱۰-۴۶ ایجاد پروژه جدید به زبان IL

سپس محیط مشابه شکل زیر ایجاد خواهد شد که قسمت بالای آن برای وارد کردن سیمبول‌های محلی و قسمت زیرین آن برای کد نویسی برنامه (ویرایش برنامه) می باشد.



شکل ۱۰-۴۷ محیط برنامه نویسی به زبان IL

در این مرحله نوار ابزار مرتبط با زبان برنامه نویسی IL نیز ظاهر خواهد شد و می‌توانیم از آن در برنامه نویسی استفاده کنیم. آیکون های این نوار ابزار را در ادامه مرور خواهیم کرد.

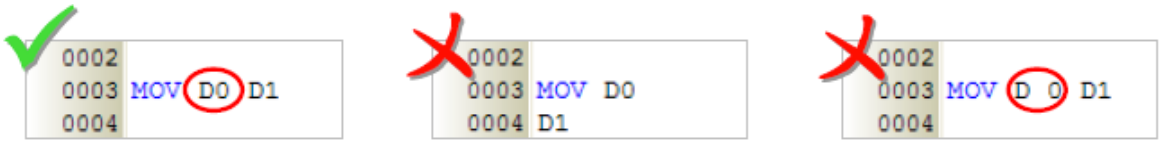
جدول ۱۰-۵: آیکون‌های نوار ابزار برنامه نویسی IL

نماد	کلید میانبر	شرح
	Shift+Ctrl+B	نشانه گذاری و یا پاک کردن نشانه‌ها در network-ها
	Shift+Ctrl+P	رفتن به نشان بعدی
	Shift+Ctrl+N	رفتن به نشان قبلی
	Shift+Ctrl+U	اضافه کردن بلوک (نوع آن را پس از کلیک کردن بر روی این آیکون باید تعیین کنید)

۱۰–۲–۳– نکات مهم در برنامه نویسی IL

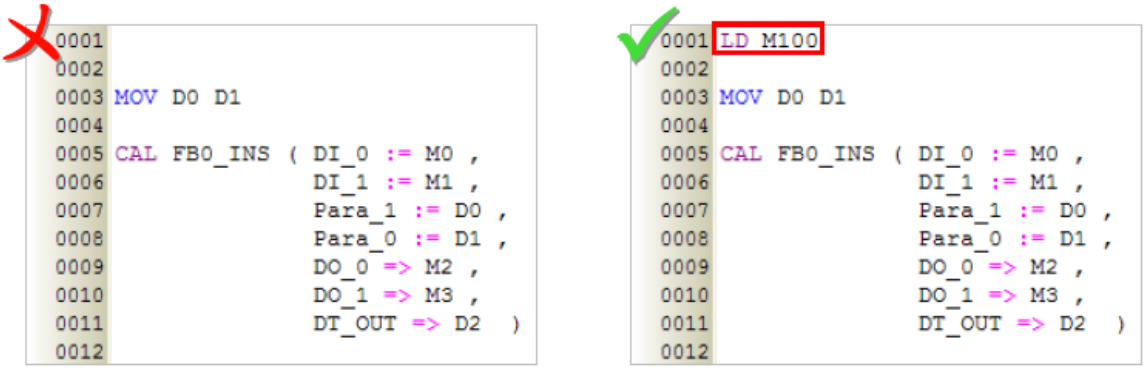
برای برنامه نویسی به زبان IL نیاز است موارد زیر در نظر گرفته شود.

- دستورات در زبان برنامه نویسی (شامل عملگر و عملوندها) باید تماما در یک سطر قرار گیرند. همچنین باید از اضافه کردن فاصله و یا سطر جدید که فرمت دستورات را به هم ریزد اجتناب کرد.



شکل ۱۰-۴۸ فاصله و یا سطر جدید که فرمت دستورات را به هم ریزد در IL خطا ایجاد می کند.

- IL نسبت به بزرگ و کوچک بودن حروف حساس نیست.
- حتما باید قبل از فراخوانی توابع بلوکی، عملیاتی بیتی وجود داشته باشد که نتیجه آن شرط اجرای آن فراخوانی شود.



شکل ۱۰-۴۹ عملیات بیتی قبل فراخوانی شرط اجرای آن تابع بلوکی را تعیین می کند

- با آنکه تعداد سطرهای کد نویسی شده تابع بلوکی و برنامه مهم نیست اما باید به میزان حافظه مورد استفاده PLC توجه کرد.
- در صورتی که کاربر بخواهد مقدار ثابتی را در ISPSoft با استفاده از زبان IL به کار ببرد، باید آن رابه صورت‌های استاندارد زیر معرفی کند.

جدول ۱۰-۶: فرمت اعداد ثابت در زبان برنامه نویسی IL در ISPSoft

نوع	شرح
دسیمال[1] (دهدهی= مبنای ده)	مانند 23456 – هر عدد بدون نشانه دسیمال در نظر گرفته می‌شود.
اُکتال[2] (مبنای ۸)	مانند 8#5564 – مقادیر اُکتال با علامت #8 در ابتدایشان مشخص می-شوند.
هگزادسیمال[3] (مبنای ۱۶)	مانند 16#5BAD – مقادیر هگزادسیمال (یا به صورت مختصر هگز) با علامت #16 در ابتدایشان مشخص می‌شوند.
باینری (عدد مبنای دو)	مانند 2#11101010011 – مقادیر باینری با علامت #2 در ابتدایشان مشخص می‌شوند.
String	"XU016S" – کاراکترهای در میان علامت نقل قول قرار می‌گیرند.
مقدار بولی (بیتی)	AH500: SM400 (کانتکت باز) یا SM401 (کانتکت بسته) مورد استفاده قرار می‌گیرند. DVP: M1000 (کانتکت باز) یا M1001 (کانتکت بسته) مورد استفاده قرار می‌گیرند.

- برای استفاده از سیمبول با نوع آرایه نیز می‌توان از فرمت Identifier[Index] استفاده کرد. در این بین Index نمی‌تواند سیمبول باشد و باید عددی صحیح بین صفر و کمتر از طول آرایه باشد.

بسیاری دیگر از مباحث مربوط به این زبان برنامه نویسی مانند تعریف سیمبول و استفاده از توابع بلوکی آماده و نشان گزاری مانند دیگر زبان‌های برنامه نویسی است که مورد بررسی قرار گرفته‌اند و از تکرار آن‌ها صرف نظر می‌کنیم. برای مرور تکنیک‌ها و مفاهیم پایه برنامه نویسی با استفاده از این زبان می‌توانید به مستندات کمپانی دلتا و مثال‌های مطرح شده در بخش‌های دیگر کتاب مراجعه نمایید.

[1] Decimal Value

[2] Octal Value

[3] Hexadecimal

۱۰–۳–زبان برنامه نویسی SFC

زبان برنامه نویسی Sequential Function Chart یک روش گرافیکی مناسب برای توصیف پروسه سلسه مراتبی برنامه کنترلی است. نمونه ای از برنامه به این زبان برنامه نویسی به همراه فلوچارت متناظر آن در ادامه آمده است. اجزای اصلی این زبان برنامه نویسی "پله[۱]ها" (به همراه "عملکردهایی[۲]" است که باید در ارتباط با آن انجام شوند)، "گذار[۳]ها" (به همراه شرایط منطقی رخداد آن) و ارتباط بین این دو می‌باشد.

Step یا پله معرف مرحله ای از برنامه کنترلی است که در آن اتفاقاتی، براساس تعاریف نویسنده برنامه، به وقوع خواهد پیوست، هر step بایک مستطیل نشان داده می شود وشماره آن معرف مرحله ای خاص از برنامه است که داخل آن نوشته می شود. عملکرد متناظر با هر پله نیز در درون مستطیل روبروی آن نوشته می شود. در هر سیکل برنامه، پله مربوط به آن فعال خواهد شد و دستورات مربوط به پله فعال در همان لحظه اجرا و پله‌های غیر فعال کاری انجام نمی دهند. برای نشان دادن وضعیت ابتدایی و در شروع برنامه SFC می بایست از یک Initial step استفاده کنیم که نشان دهنده مرحله آغاز برنامه است ، نماد آن نیز یک مستطیل دو خطی است. هر برنامه SFC فقط دارای یک Initial step می‌باشد.

Transition یا گذار بصورت یک خط افقی مسیر ارتباطی بین دو step را قطع می کند و نام آن در سمت راست آن آمده است. گذارها در هر مرحله از برنامه شروط موجود در قسمتهای قبلی خود را می-بینند و بر آورده شدن و عدم برآورده شدن آن شروط بررسی را می کنند و اگر شروط هر مرحله برآورده شده باشد، گذارها مربوطه اجازه عبور از آن مرحله را صادر می کند و بالعکس.

دو قانون پایه ای در استفاده از زبان SFC به صورت زیر است:

۱.هرگز و در هیچ قسمتی از برنامه دو step بدون وجود Transition، پشت سر هم قرار نمی گیرند.

۲. هرگز و در هیچ قسمت از برنامه دو Transition بدون وجود step پشت سر هم قرار نمی گیرند[۴].

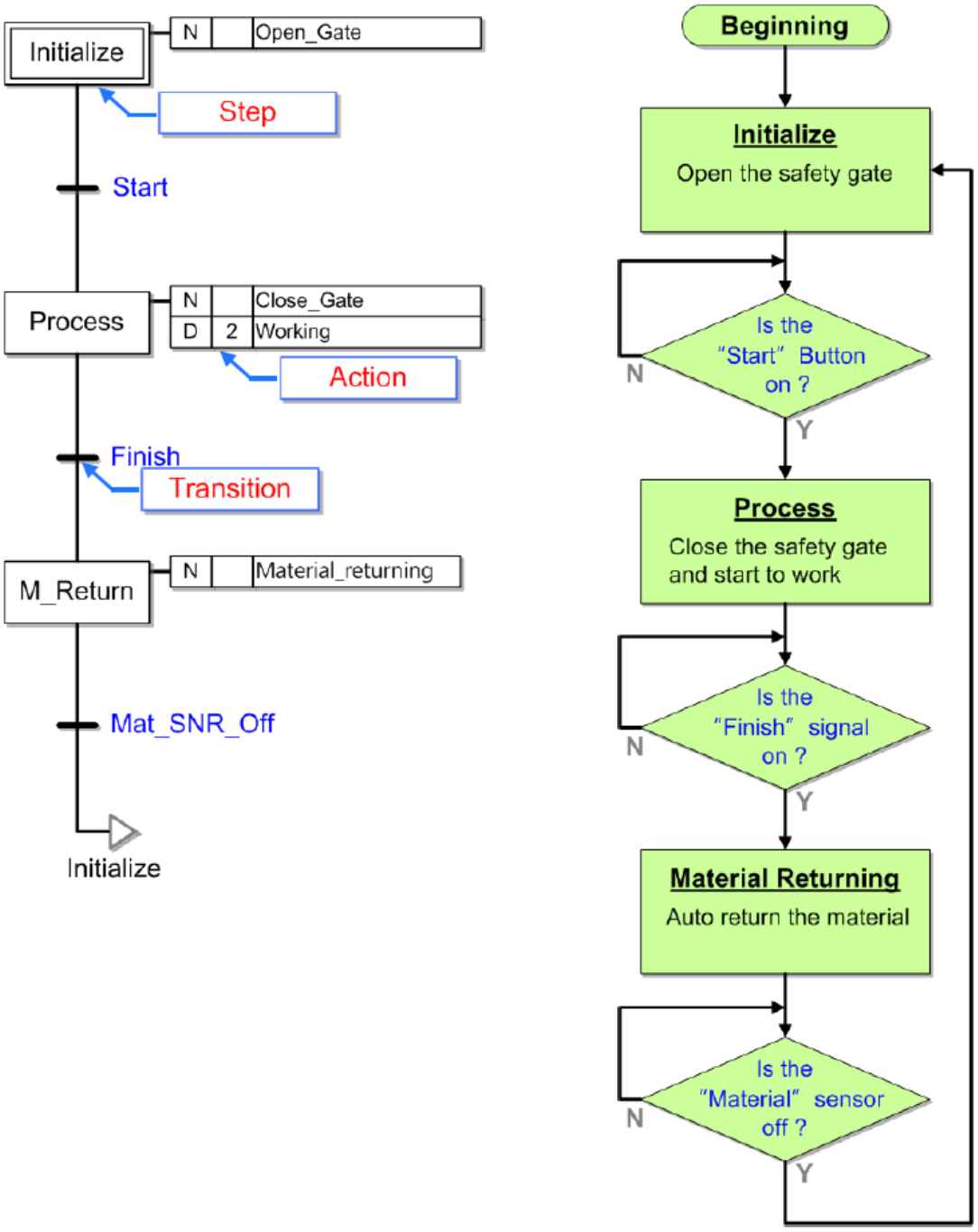
---

[۱] Step

[۲] Action

[۳] Transition

[۴] منبع بخشی از متن: "بررسی سیستم های تله متری کنترل و اتوماسیون صنعتی"



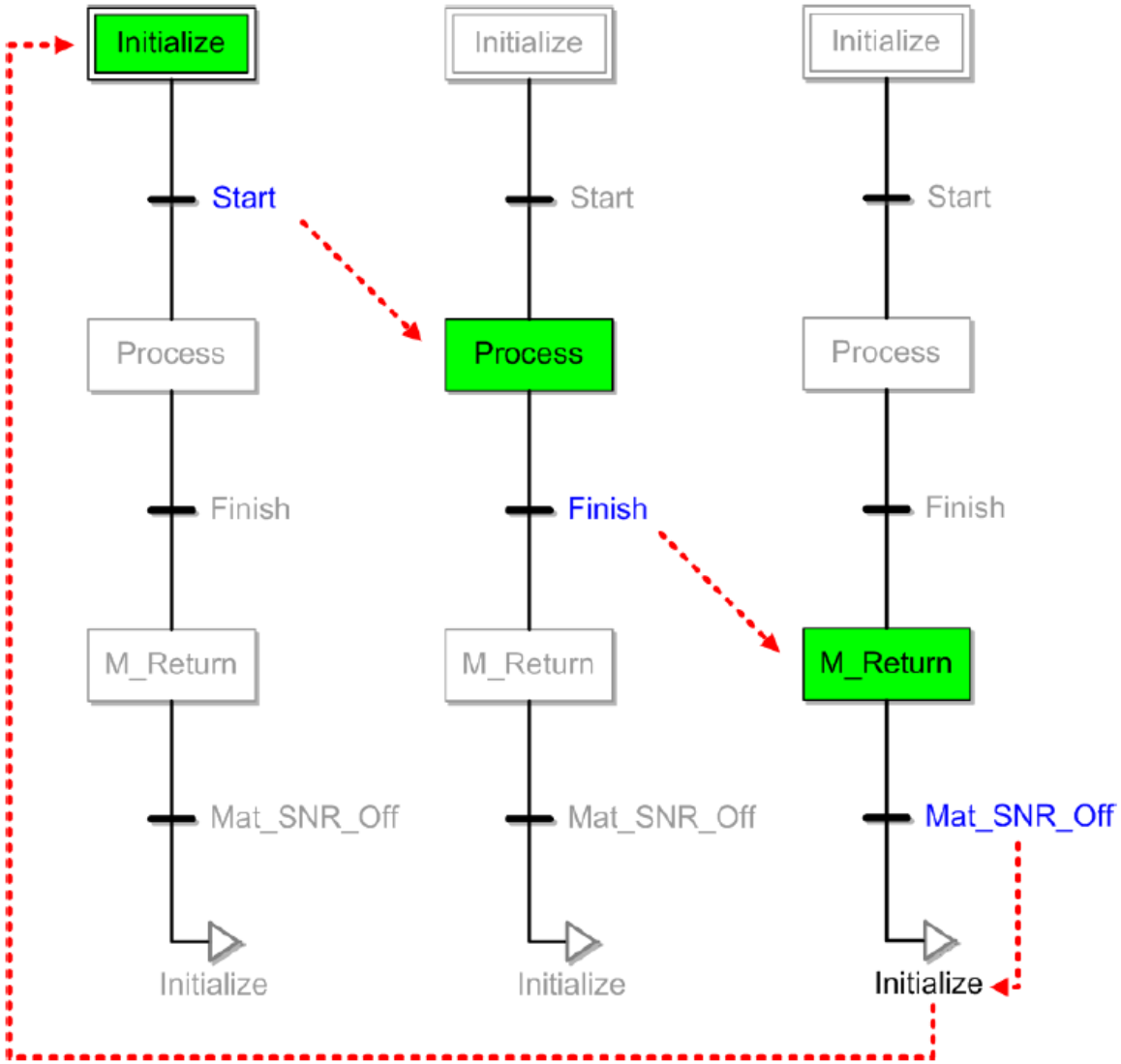
شکل ۱۰-۵۰ برنامه ای به زبان SFC و فلوچارت متناظر آن

سلسله مراتب یا به عبارتی ترتیب فراخوانی پله‌ها در SFC به صورت مستطیل‌های متوالی در برنامه مشخص شده است. هر پله حالت[۱] تحت کنترلی از سیستم را معرفی می‌کند. گذار در SFC دارای شرایطی است که اگر صحیح باشد، پله قبل گذار غیرفعال و پله بعد آن فعال می‌شود.

[۱] State

پله‌ها در SFC می‌توانند فعال و یا غیرفعال باشند ولی تنها عملکرد پله های فعال اجرا خواهد شد. در صورتی که پله‌ای فعال نباشد، گذار بعد آن در اجرای برنامه نقشی نخواهد داشت و عملکردهای آن پله نیز اجرا نخواهند شد.

در زبان برنامه نویسی SFC در ISPSoft حتما باید پله‌ها و گذارها یکی در میان بیایند وگرنه برنامه قابل کامپایل نخواهد بود.

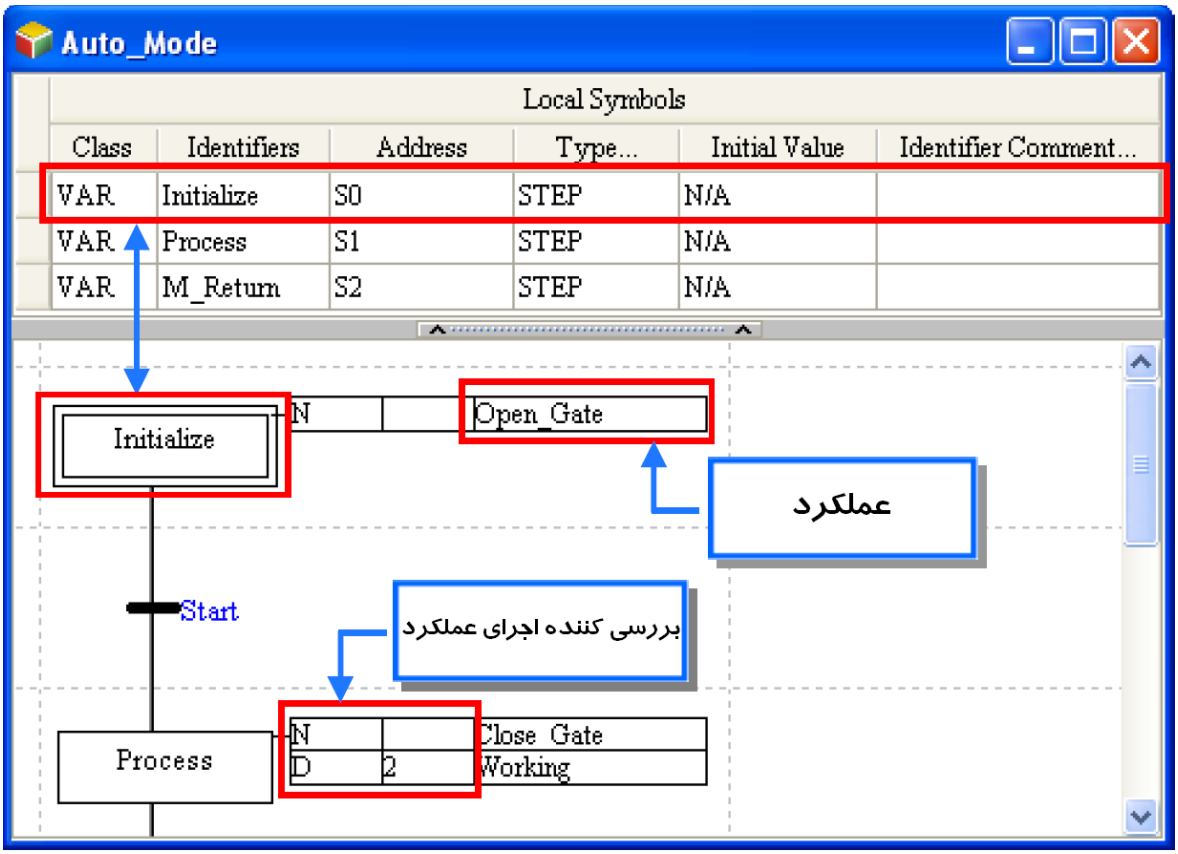


شکل ۱۰-۵۱ نمونه ای از مراحل اجرای یک برنامه به زبان SFC

۱۰-۳-۱- مرور مفاهیم زبان برنامه نویسیSFC در ISPSoft

۱۰-۱-۳-۱- پله ها و عملکردها

هر پله در ISPSoft باید به متغییری از نوع STEP تخصیص داده شود و به صورت یک پرچم وضعیت[۱] عمل کند. در صورتی که این پرچم وضعیت فعال باشد پله نیز فعال خواهد بود و عملکرد مرتبط با آن نیز بررسی خواهد شد. در ISPSoft عملکردها می‌توانند چندبخشی باشند و همچنین با چند پله در ارتباط باشند. برعکس این موضوع نیز برقرار است و می‌توان به هر پله چند عملکرد اختصاص داد.

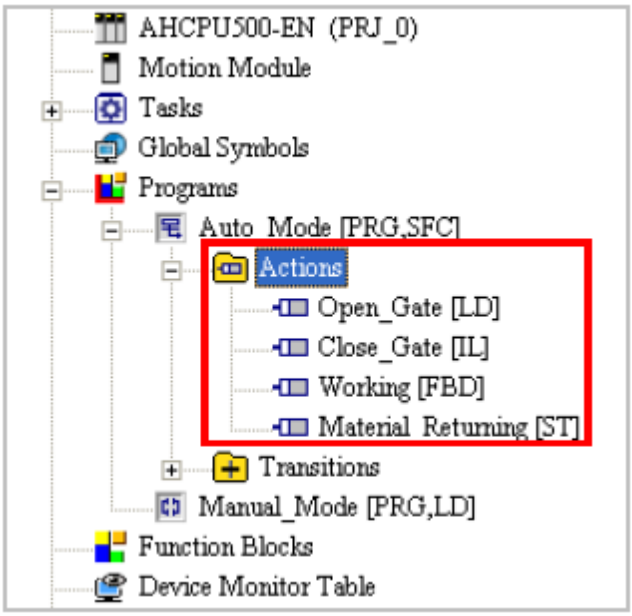


شکل ۱۰-۵۲ بخش های مختلف یک برنامه SFC

در ISPSoft عملکرد کدی از برنامه است و می‌تواند به وسیله چهار زبان برنامه نویسی Ladder، FBD، IL و ST نوشته شود. عملکردهای ساخته شده نیز در بخش مدیریت پروژه لیست می‌شوند. در محیط ویرایش برنامه‌ی عملکردها، بر خلاف محیط ویرایش برنامه در دیگر زبان‌های برنامه نویسی دارای قسمت

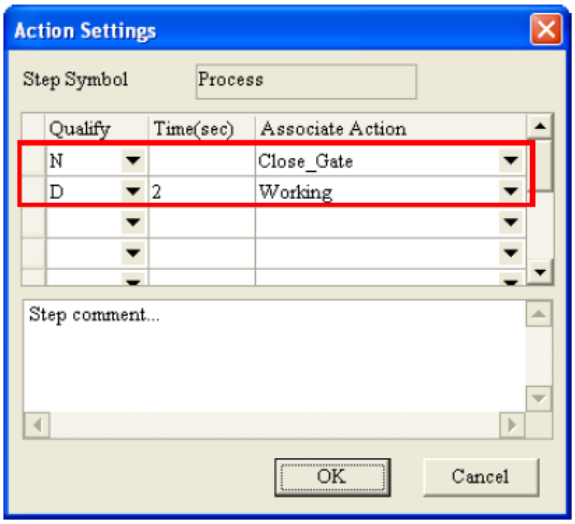
[۱] پرچم وضعیت، بیانگر بیتی است که دو حالت فعال و یا غیرفعال بودن را (درست و یا نادرست بودن را) تعیین می‌کند.

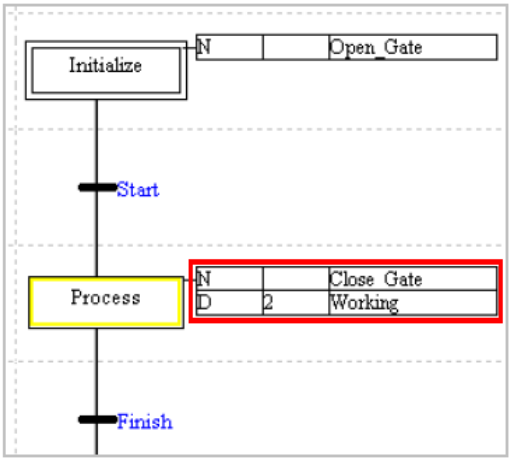
تعریف سیمبول نیست. در ISPSoft عملکردها و گذارها در هر برنامه SFC دارای یک جدول سیمبول محلی مشترک هستند.



شکل ۱۰-۵۳ عملکردها در بخش مدیریت پروژه

برای ساخت SFC باید عملکردها را ساخت و به پله‌ها اختصاص داد، همچنین باید بررسی‌کننده[۱] هایی شرطی را برای اجرای هر عملکرد در پله‌ها در نظر گرفت. پنجره Action Setting و لیست عملکردهای لیست شده در آن برای پله‌ی Process در شکل زیر آمده است.

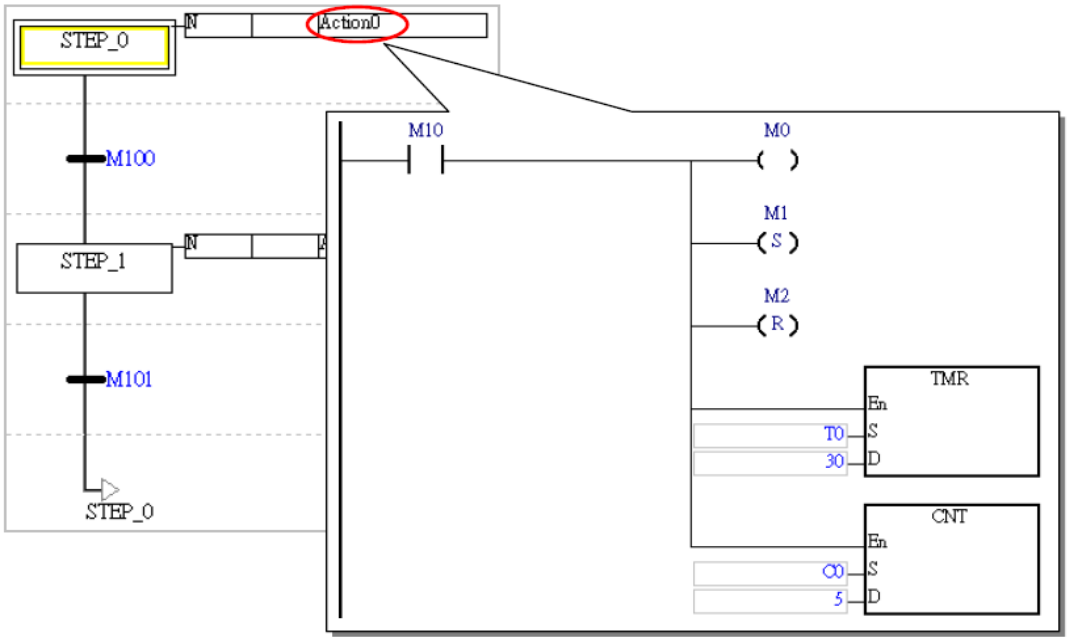




[۱] Qualifier

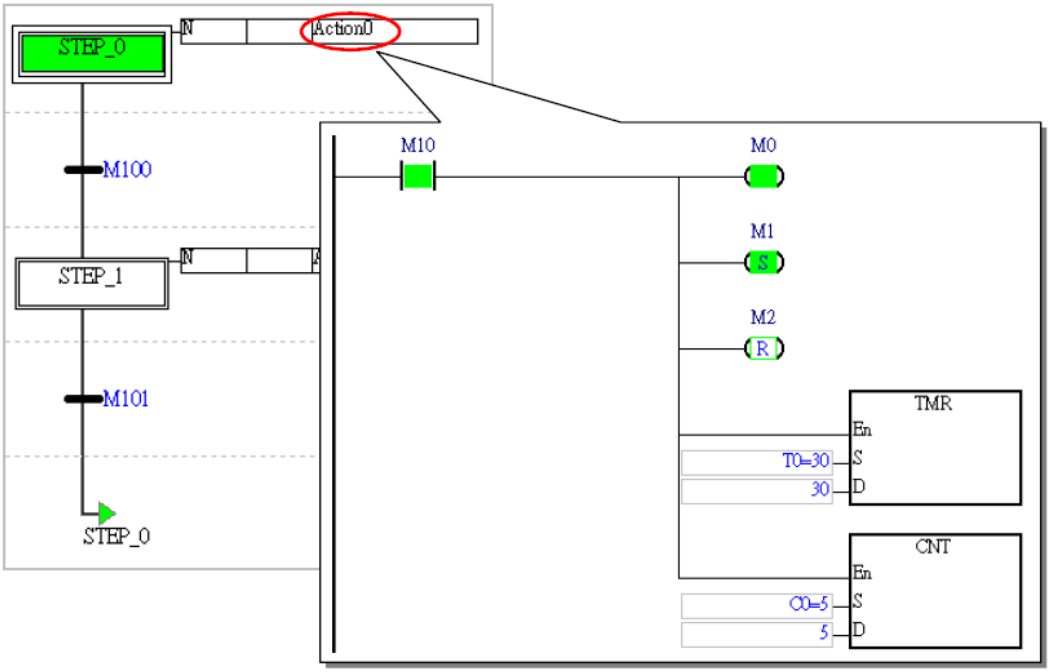
شکل ۱۰-۵۴ تنظیم عملکردها برای پله Process در برنامه نمونه

اسکن‌های نهایی (مرحله بعد از اجرای عملکردها) مرحله مهمی در اجرای برنامه‌های SFC هستند. پس از آنکه عملکردها اجرا شدند، سیستم به صورت اتوماتیک خروجی‌های عملکرد را غیرفعال می‌کند. در اسکن نهایی پس از اجرای عملکرد، کویل های خروجی به حالت OFF در می‌آیند، بلوک‌ها غیرفعال می‌شوند، تایمرها ریست می‌شوند، کویل‌های ست و ریست عمل نمی‌کنند، شمارنده‌ها متوقف می‌شوند و حافظه‌های شمارنده‌ها ثابت می‌ماند. در شکل زیر، عملکرد Action0 نوع N است و به پله STEP_0 اختصاص داده شده است. زمانی که پله STEP_0 غیرفعال و STEP_1 فعال شود، سیستم اسکن نهایی را برای عملکرد Action0 اجرا می‌کند.



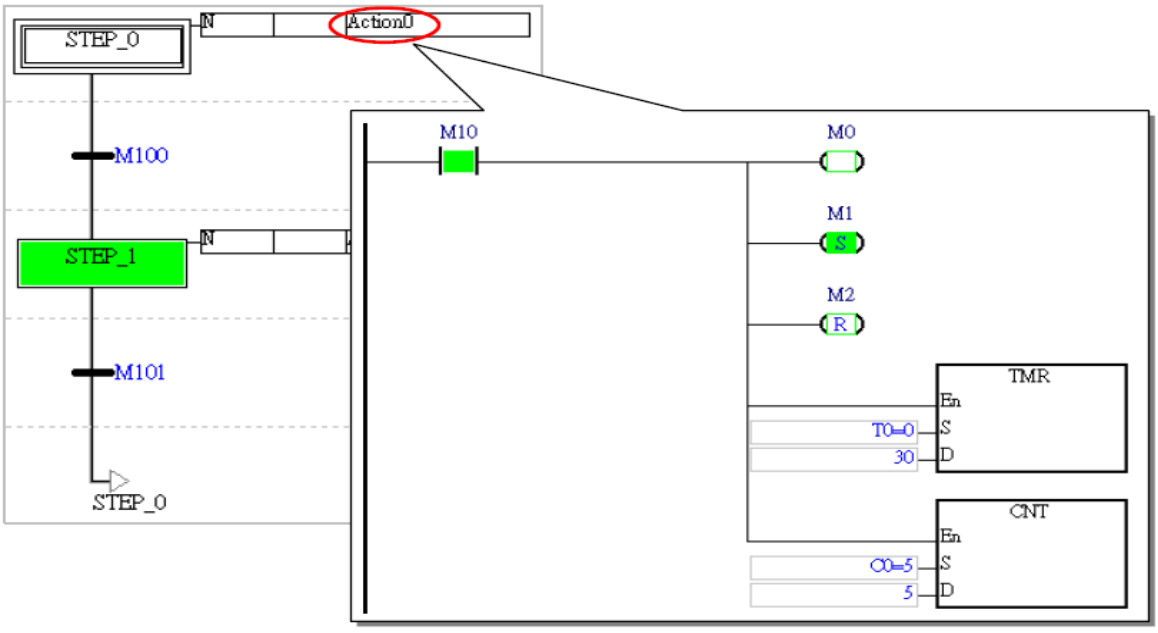
شکل ۱۰-۵۵ عملکرد Action0 برای پله Step_0

در شکل زیر STEP_0 فعال است. M10 در عملکرد Action0 فعال و در نتیجه M0 نیز فعال است، M1 ست و M2 ریست شده است، مقدار T0 برابر ۳۰ و مقدار C0 برابر ۵ است.



شکل ۱۰-۵۶ فعال بودن Action0 در نتیجه فعال بودن STEP_0

در شکل زیر STEP_1 فعال می‌شود و در نتیجه آن طی اسکن نهایی Action0 حافظه M10 روشن، M0 خاموش و T0 ریست شده است. مقدار M1، M2 و C0 بدون تغییر مانده‌اند.

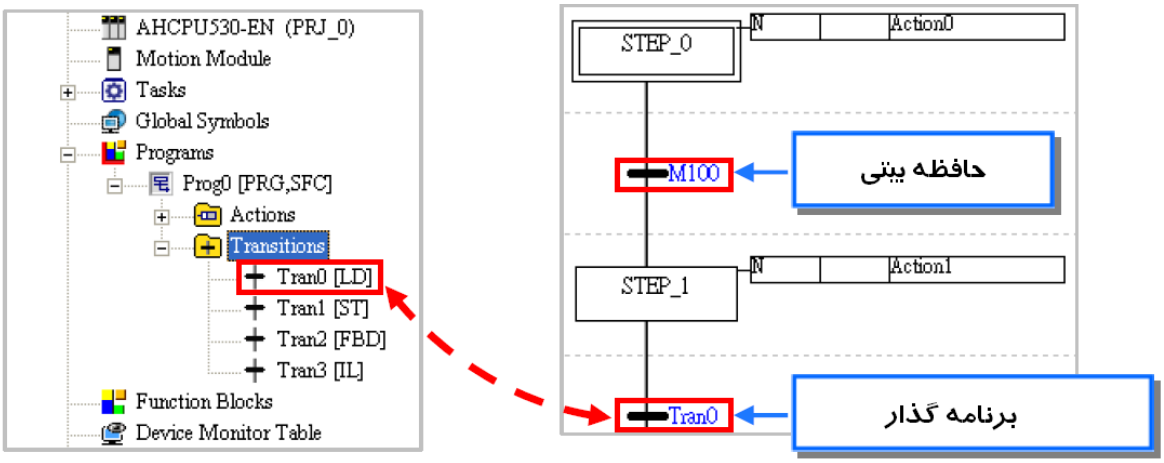


شکل ۱۰-۵۷ غیرفعال شدن Action0 پس از اسکن نهایی

۱۰-۳-۱-۲- گذارها

زمانی که گذاری فعال باشد، پله قبل از آن غیرفعال و پله بعد از آن فعال می‌شود. گذار می‌تواند هر زمانی فعال شود، با این حال تنها زمانی می‌تواند پله بعد خود را فعال کند که پله قبل آن فعال باشد.

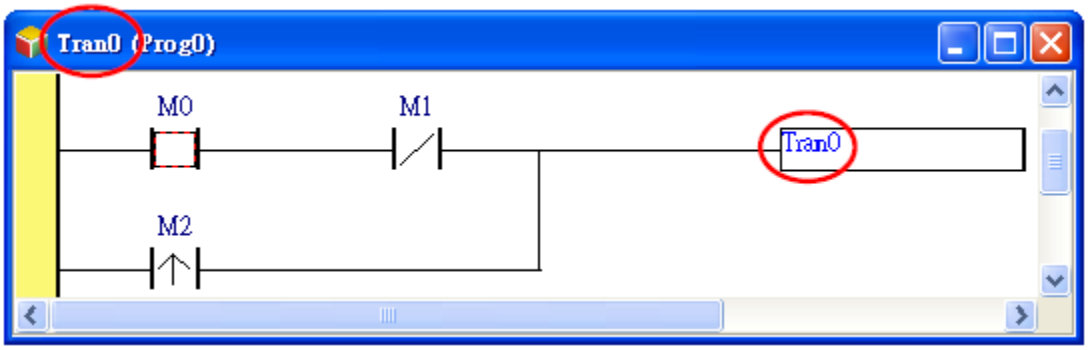
در ISPSoft گذار می‌تواند حافظه و یا سیمبولی بیتی و یا حتی کُد برنامه باشد. در صورتی که گذار عملیاتی منطقی باشد و نیاز به نوشتن کُد داشته باشد کاربر می‌تواند برای آن برنامه بنویسد و آن را به گذار مورد نظر تخصیص دهد. برنامه‌های نوشته شده برای گذارها در بخش مدیریت پروژه لیست خواهند شد.



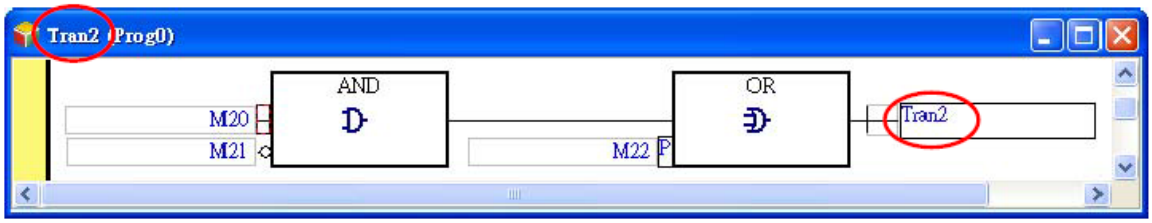
شکل ۱۰-۵۸ برنامه های نوشته شده گذارها در بخش مدیریت پروژه

از هرچهار زبان برنامه نویسیLadder، FBD، IL و ST می‌توان برای ساخت برنامه گذار انتخاب کرد. برنامه گذار باید در نهایت نتیجه بیتی خود را به سیمبولی هم نام برنامه گذار ارسال کند، این سیمبول نیاز به تعریف شدن ندارد (از قبل تعریف شده است) و در صورت تعریف سیمبولی با نام برنامه، کامپایل آن دچار خطا می‌شود. توابع بلوکی و توابع سیستمی را نمی‌توان در برنامه گذار نمی‌توان استفاده کرد اما کانتکت-های مقایسه‌ای بلوک‌های منطقی مانند PN وNP وINV را می‌توان در آن‌ها مورد استفاده قرار داد.

در صورتی که زبان برنامه نویسی مورد استفاده برای ساخت برنامه گذار Ladder و یا FBD باشد، تنها باید در یک Network نوشته شود و کویل خروجی آن باید تنها یکی و همنام نام برنامه گذار باشد.

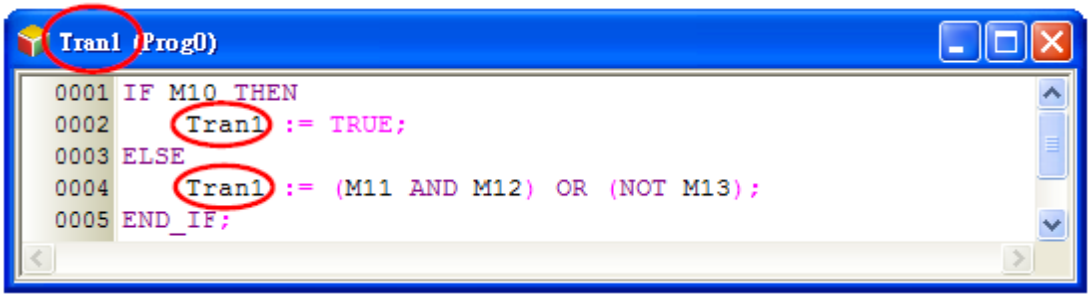


شکل ۱۰-۵۹ برنامه گذار به زبان Ladder



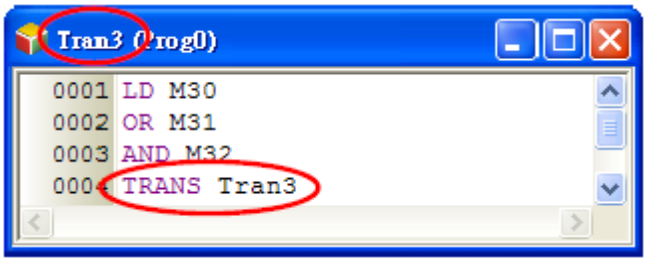
شکل ۱۰-۶۰ برنامه گذار به زبان FBD

در صورتی که زبان برنامه نویسی مورد استفاده برای برنامه گذار ST باشد محدودیتی در میزان کدنویسی وجود ندارد و تنها باید سیمبولی بیتی همنام با برنامه گذار برای اختصاص نتیجه برنامه در نظر گرفته شود.



شکل ۱۰-۶۱ برنامه گذار به زبان ST

در صورتی که زبان برنامه نویسی مورد استفاده برای برنامه گذار IL باشد محدودیتی در میزان کدنویسی وجود ندارد و تنها باید از TRANS برای انتقال نتیجه از طریق سیمبولی بیتی همنام با برنامه گذار استفاده کرد. در این برنامه نمی‌توان از OUT و SET و RST استفاده کرد.



شکل ۱۰-۶۲ برنامه گذار به زبان IL

زمانی که کاربر برنامه گذار را می‌سازد، سیمبولی با نام برنامه گذار به آن اضافه می‌شود و کاربر تنها لازم است برنامه عملکردی را طرح بریزد. در واقع از آنجایی که عملکردها و گذارها در SFC دارای یک

جدول سیمبول‌های محلی مشترک در SFC هستند به همین علت در محیط ویرایش برنامه محلی برای وارد کردن سیمبول‌های محلی برای آن‌ها در نظر گرفته نشده است.

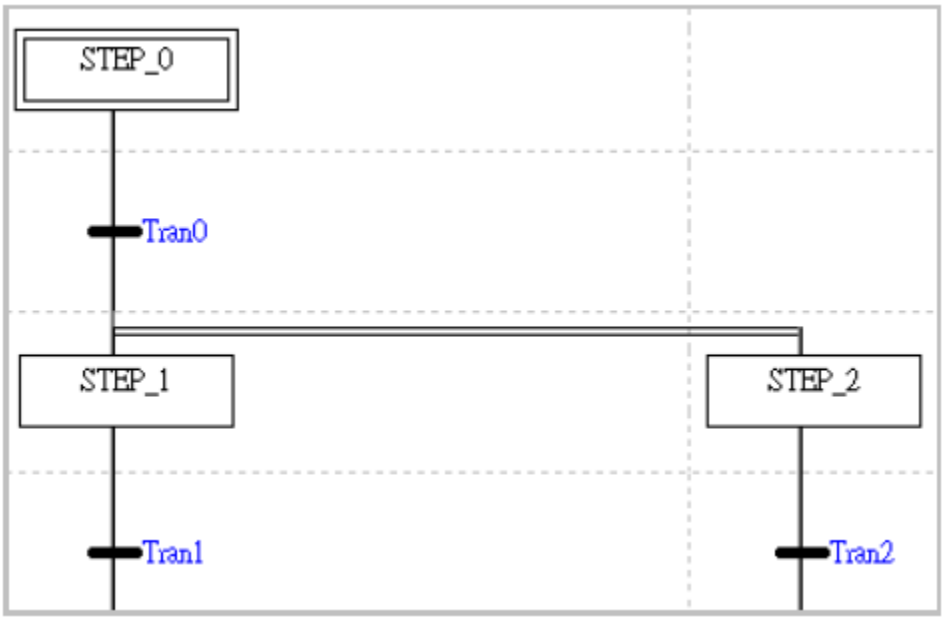
کاربر نباید برنامه‌ای پیچیده را برای گذار بنویسد. در صورتی که برای تعیین شرایط گذار نیاز به بررسی‌های پیچیده‌ای باشد، کاربر می‌تواند آن را در پله قبل گذار بنویسد و نتیجه آن را در پرچم وضعیتی ذخیره و در گذار مورد استفاده قرار دهد.

۱۰-۳-۱-۳- انشعاب در برنامه SFC

در مسیر برنامه SFC می‌توان به دو روش مختلف انشعاب ساخت: انشعاب اجرای همزمان و یا انشعاب انتخاب ترتیبی. در ادامه این دو روش انشعاب گیری را مرور خواهیم کرد.

- انشعاب اجرای همزمان

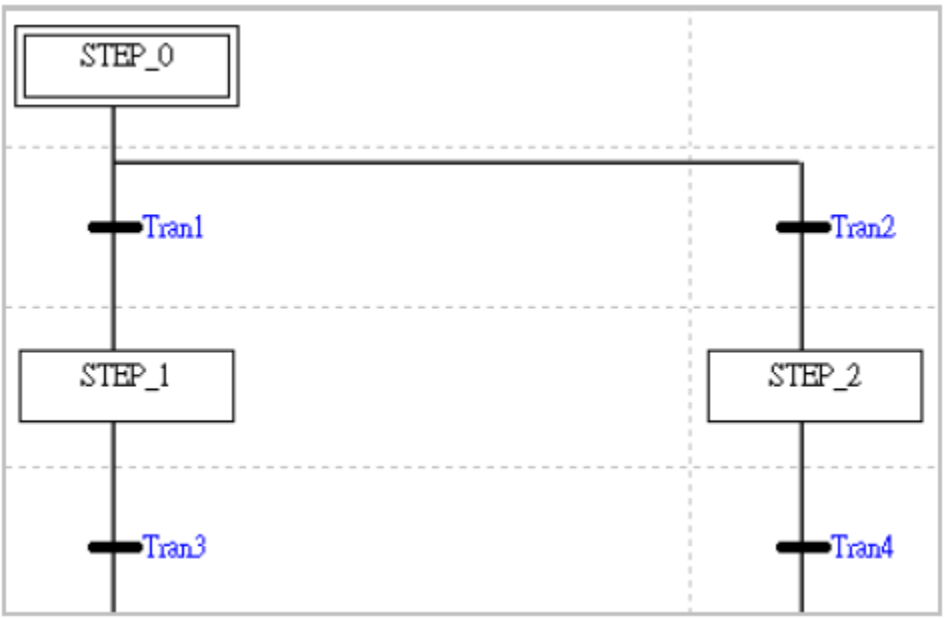
انشعاب اجرای همزمان در شکل زیر نمایش داده شده است. این انشعاب با دو خط مجزا شده. از ویژگی‌های این نوع انشعاب آن است که از یک گذار مشترک استفاده می‌کنند و زمانی که گذار فعال باشد پله‌های انشعاب همزمان فعال می‌شوند. مثلا در زیر زمانی که Tran0 فعال باشد، STEP_0 غیرفعال و STEP_1 و STEP_2 فعال می‌شوند.



شکل ۱۰-۶۳ انشعاب اجرای همزمان

- انشعاب انتخاب ترتیبی

انشعاب انتخاب ترتیبی در زیر نمایش داده شده است. ویژگی این نوع انشعاب آن است که پله‌های هر انشعاب هرکدام گذار منحصر به فرد خود را دارند. زمانی که یک گذار فعال است، پله‌های در امتداد آن فعال می‌شوند و گذارهای مسیر دیگر بررسی نمی‌شوند. در نتیجه اگر مسیرهای انشعاب متعددی وجود داشته باشد، فقط یک مسیر در هر لحظه انتخاب و اجرا می‌شود. این نکته نیز مهم است که گذارها به ترتیب از مسیر چپ به راست مورد بررسی قرار می‌گیرند پس به عنوان مثال در صورتی که چند گذار متوالی فعال باشند ، تنها پله‌های مسیر چپ اجرا خواهند شد. در مثال زیر Tran2 فعال و به طبع آن STEP_0 غیرفعال خواهد شد. Tran1 بررسی نخواهد شد و در نتیجه آن STEP_1 غیرفعال خواهد ماند.



شکل ۱۰-۶۴ انشعاب انتخاب ترتیبی

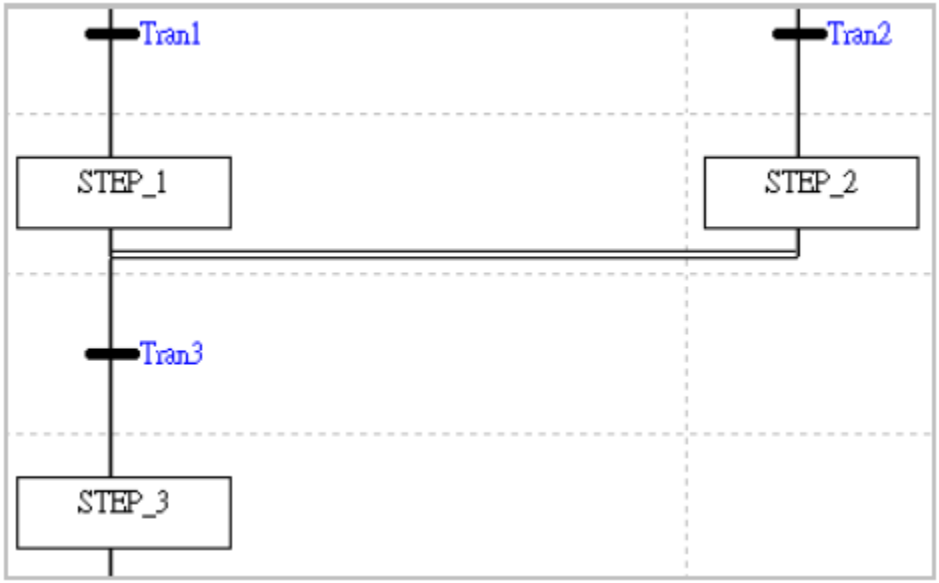
۱۰-۳-۱-۴- همگرایی در برنامه SFC

مسیرهای واگرا یا انشعاب گرفته شده در SFC می‌توانند در نهایت همگرا شوند. دو روش همگرایی وجود دارد، همگرایی همزمان و همگرایی انتخاب ترتیبی که در ادامه شرح داده خواهند شد.

- همگرایی همزمان

نمایشی از همگرایی همزمان در زیر آمده است. همگرایی همزمان به وسیله مسیر دو خطه مشخص شده است و در آن دو مسیر مجزای انشعاب گرفته شده به هم متصل شده‌اند. ویژگی این نوع همگرایی آن است که پله‌های دو مسیر به یک گذار متصل شده‌اند. این گذار نیز تا زمانی که پله‌های قبل آن فعال

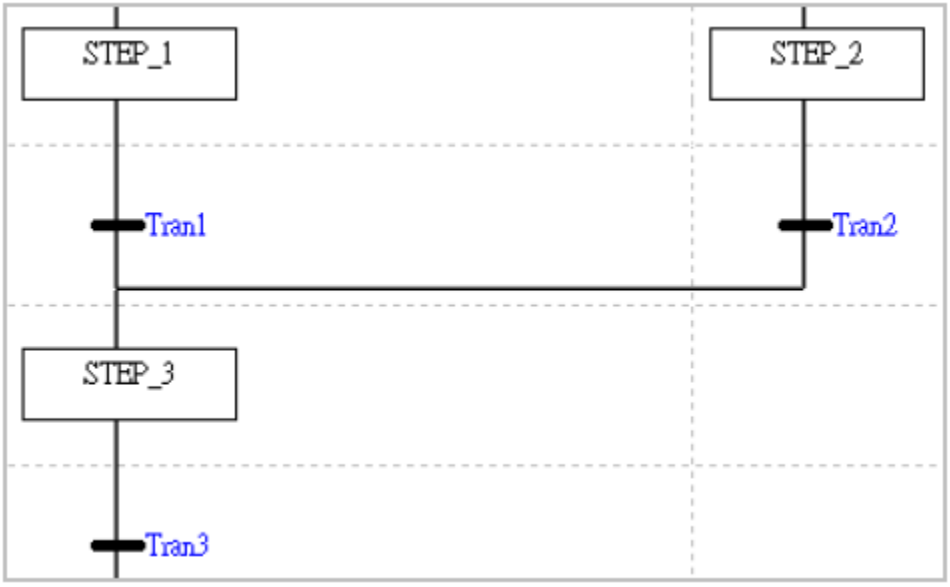
شوند مورد ارزیابی قرار نمی‌گیرد. در مثال زیر پس از آنکه STEP_1 و STEP_2 فعال شوند، گذار Tran3 مورد ارزیابی قرار می‌گیرد، در صورتی که Tran3 فعال باشد،STEP_1 و STEP_2 غیرفعال خواهند شد و STEP_3 فعال می‌شود.



شکل ۱۰-۶۵ همگرایی همزمان

- همگرایی انتخاب ترتیبی

نمونه‌ای از همگرایی انتخاب ترتیبی در شکل زیر نمایش داده شده است. ویژگی این نوع همگرایی آن است که پله‌های همگرا هرکدام گذار مربوط به خود را دارند و زمانی که یکی از این گذارها یک شود، پله بعدی فعال می‌شود. در صورتی که چند مسیر موازی به این صورت همگرا می‌شوند ممکن است هر مسیر به صورت مجزا یکبار دیگر پله بعد همگرایی را فعال کند. به همین خاطر باید در برنامه نویسی دقت لازم صورت پذیرد. مثلا در شکل زیر زمانی که Tran1 فعال شود، STEP_1 غیرفعال و STEP_3 فعال می‌شود. همچنین اگر Tran2 فعال شود، ابتدا STEP_2 غیرفعال و سپس دوباره STEP_3 فعال خواهد شد.

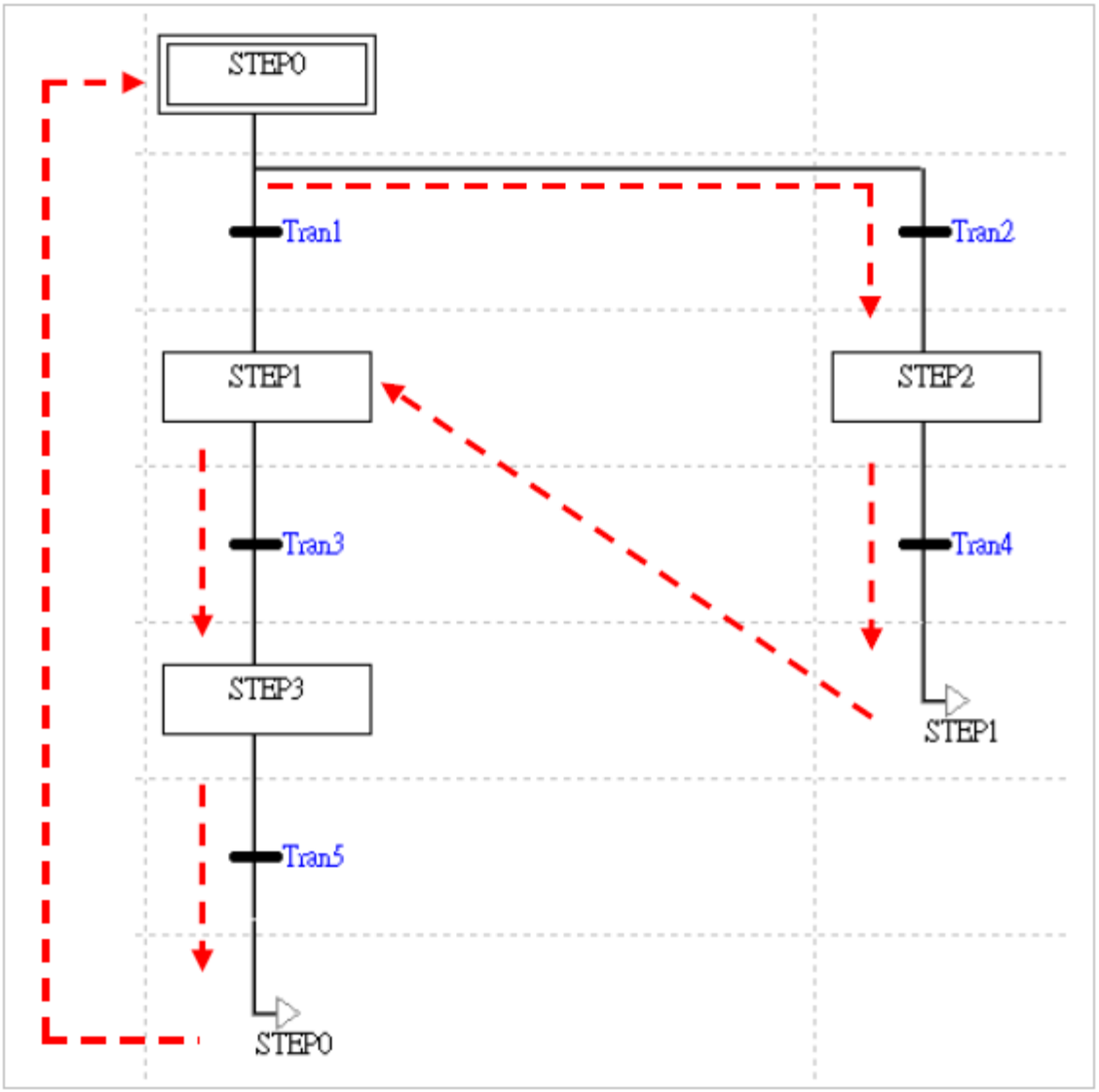


شکل ۱۰-۶۶ همگرایی انتخاب ترتیبی

۱۰-۳-۱-۵- پرش

برای کنترل هرچه بیشتر مراحل در نظرگرفته شده در SFC می‌توان از پرش[۱] استفاده کرد. در مثال زیر در شاخه سمت چپ انشعاب انتخاب ترتیبی در صورتی برنامه به Tran4 برسد و آن فعال باشد، برنامه از طریق گذر از پرش به STEP1 می‌رسد. زمانی نیز که Tran5 فعال شود، برنامه از طریق پرش بعد آن به STEP0 می‌رسد.

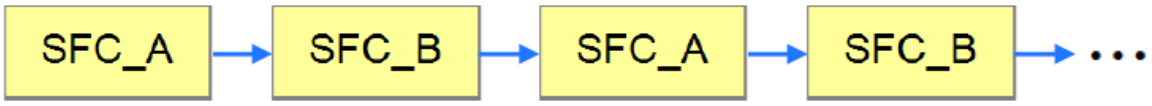
[۱] Jump



شکل ۱۰-۶۷ برنامه نمونه دارای پرش درون برنامه ای

پرش در SFC برای رفتن به یک SFC دیگر قابل استفاده نیست و در صورتی که مقصدی بیرون از SFC مورد نظر برای پرش در نظر گرفته شود، کامپایل برنامه با مشکل مواجه خواهد شد. برای اینکه بتوان بین دو SFC جابجا شد به جای پرش مستقیم می‌توان از ویژگی‌های پله‌ها و گذارها استفاده کرد. برای اینکار می‌توان پله و یا گذاری را در جدول سیمبول سراسری تعریف کرد و آن را در SFC مقصد قرار داد. سپس عملکردی را به صورتی در SFC مبدا به کار برد که در صورت برآورده شدن شرط پرش، پله و یا گذار تعریف شده در مقصد را فعال کرده و پرش رخ دهد.

برای درک بهتر این موضوع به مثال زیر توجه کنید. شکل زیر نشان می‌دهد که ابتدا SFC_A اجرا شده و بعد از آن SFC_B اجرا خواهد شد. و به همین ترتیب این دو SFC پس از یکدیگر اجرا خواهند شد. برای پیاده سازی این برنامه با استفاده از پرش از ویژگی‌های گذار استفاده می‌کنیم.

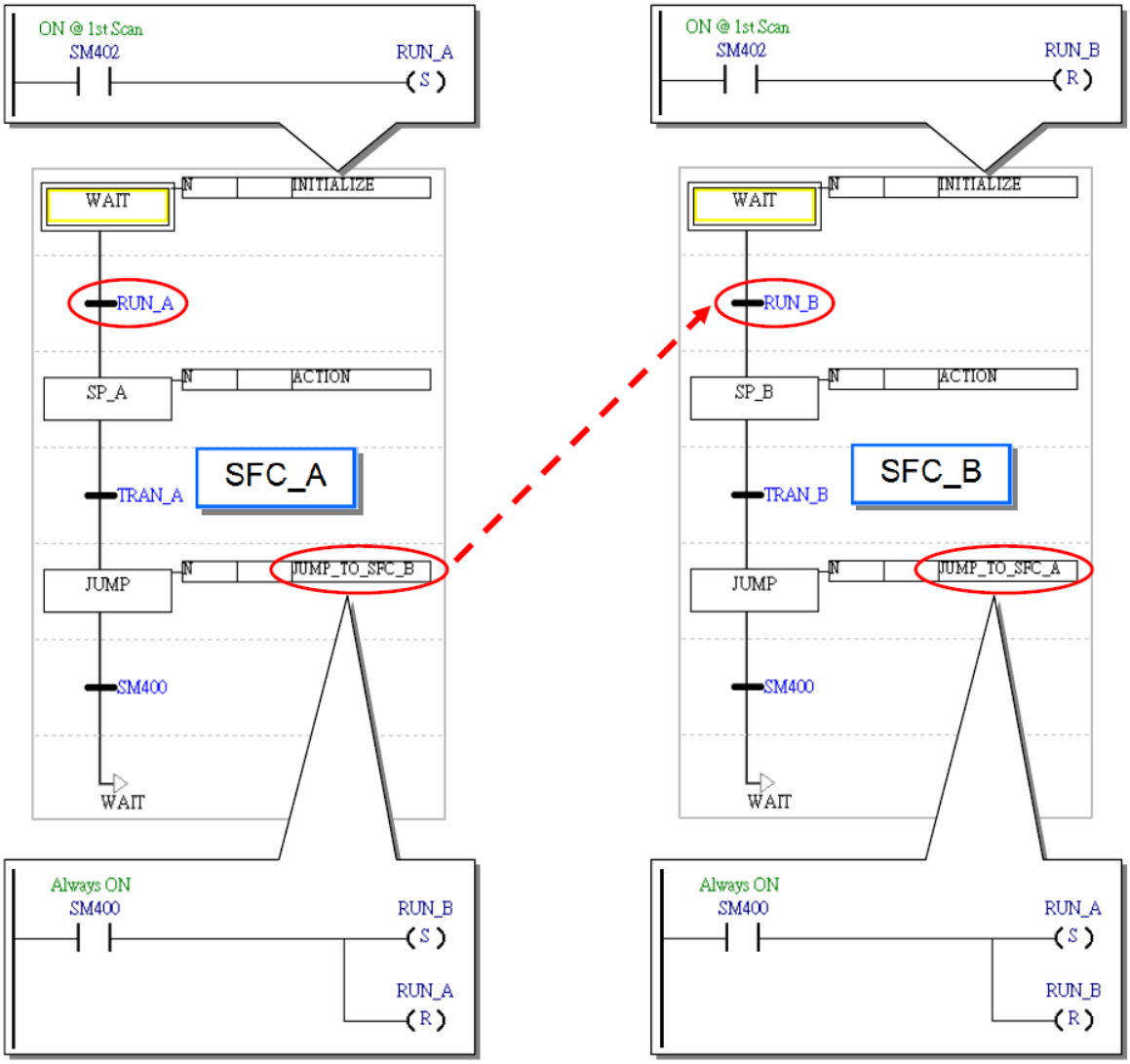


شکل ۱۰-۶۸ اجرای متوالی دو برنامه SFC با استفاده از پرش

(با مطالعه ادامه این بخش درک فرایند پیش رو راحت تر خواهد بود، به همین علت پیشنهاد می‌شود این قسمت را پس از مطالعه فصل دوباره مرور نمایید) دو پله SP_A و SP_B در مثال زیر برای مشخص شدن SFC در حال اجرا به کار می‌روند و همچنین دو گذار RUN_A و RUN_B در جدول سیمبول‌های سراسری تعریف شده‌اند. بعد از اجرای برنامه، پله WAIT در SFC_A فعال خواهد شد و طی آن RUN_A فعال خواهد شد و همچنین پله WAIT در SFC_B فعال و RUN_B را غیر فعال خواهد کرد، پس کنترل[1] در SFC_A به حرکت خود در برنامه ادامه می‌دهد ولی در SFC_B قبل از گذار RUN_B متوقف می‌شود. پس برنامه مسیر SFC_A را ادامه می‌دهد تا کنترل پس از عبور از SP_A به JUMP برسد و آن را فعال کند. عملکرد JUMP گذار RUN_B را فعال و RUN_A را غیرفعال می‌کند. حال اجرا در SFC_B با فعال شدن RUN_B از سرگرفته می‌شود و کنترل از SP_B می‌گذرد و به WAIT می‌رسد. به صورت همزمان چون گذار بعد JUMP در SFC_A همیشه فعال است، برنامه در آن به WAIT می‌رسد ولی چون RUN_A غیر فعال شده، کنترل نمی‌تواند از SP_A عبور کند و متوقف می‌شود. در واقع RUN_A تا زمانی که JUMP در SFC_B اجرا نشود فعال نخواهد شد.

با استفاده از WAIT و JUMP در SFC_A توانستیم RUN_B را فعال و با استفاده از WAIT و JUMP در SFC_B توانستیم RUN_A را فعال کنیم و این روند را به صورت زنجیره‌ای ادامه دار طراحی کنیم.

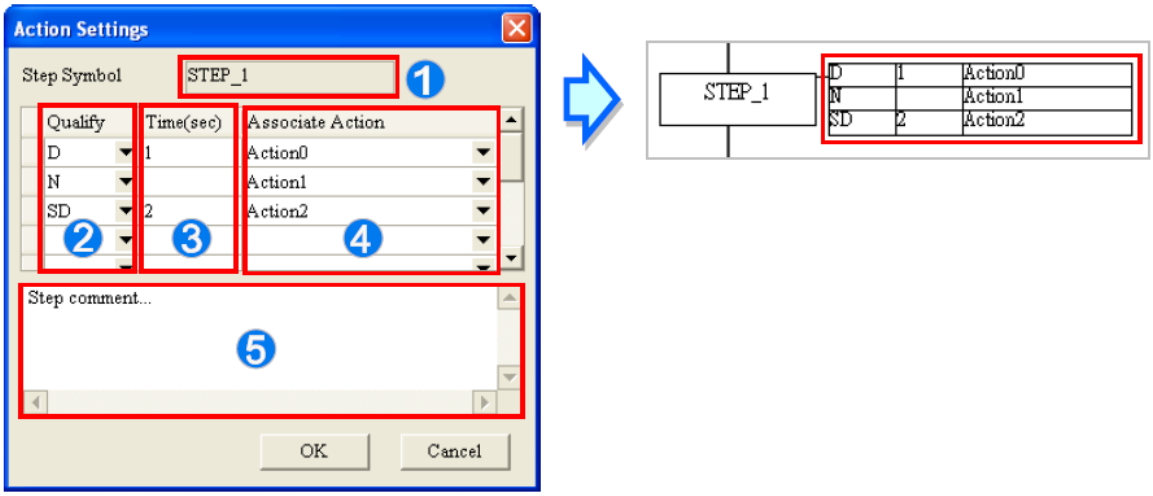
[1] اصطلاحا کنترل در مسیر اجرای برنامه عبور کرده و هر لحظه بر روی قسمت در حال اجرا قرار می‌گیرد.



شکل ۱۰-۶۹ طراحی پرش بین برنامه ای برای جابجایی بین دو برنامه SFC

۱۰-۳-۱-۶- بررسی کننده عملگرها

در ISPSoft با استفاده از بررسی‌کننده‌های عملکردها می‌توان چگونگی اجرای آن‌ها را تعیین کرد. با این حال، عملکردهای PLCهای سری DVP از نوع N (انواع بررسی کننده‌ها را در ادامه این قسمت مرور خواهیم کرد) هستند و امکان بررسی و تعیین چگونگی اجرای آن‌ها وجود ندارد.



شکل ۱۰-۷۰ پنجره تخصیص بررسی کننده به عملکردها

1 نام پله

2 انتخاب نوع بررسی کننده

3 تعیین زمان برای بررسی کننده

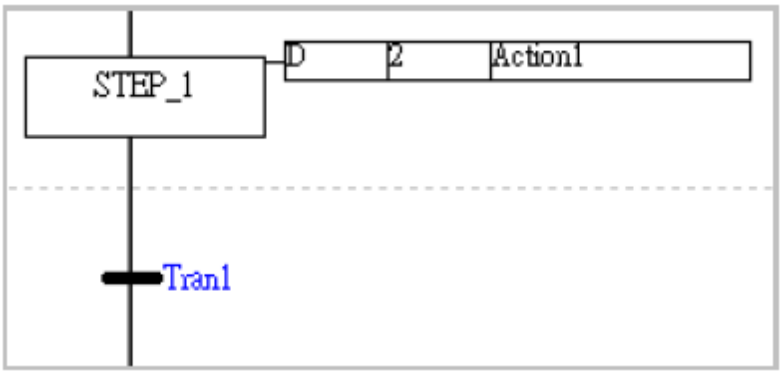
4 عملکرد مورد بررسی

5 کامنت گزاری بر روی پله

در اینجا به مرور انواع بررسی کننده‌ها می‌پردازیم.

- N (Normal): زمانی که پله فعال می‌شود، عملکرد آن نیز اجرا می‌شود. زمانی که پله فعال نیست، سیستم اسکن نهایی را برای غیرفعال کردن خروجی‌های عملکرد مورد نظر اجرا می‌کند (البته همانطور که قبلا گفته شد در اسکن نهایی، خروجی دستوراتی مانند SET بدون تغییر می‌مانند). تمامی عملکردهای PLCهای سری DVP از نوع N هستند (یعنی بررسی کننده‌های نوع N دارند).
- S (SET): زمانی که پله فعال است، عملکرد S فعال می‌شود و پس از آن حتی اگر پله متوقف شود، عملکرد با استفاده از اسکن نهایی متوقف نمی‌شود و فعال می‌ماند.
- D (Delay): اگر پله فعال شود، بعد از مدت مشخص شده برای تاخیر، عملکرد اجرا می‌شود. پس از غیرفعال شدن پله نیز عملکرد غیرفعال می‌شود. در صورتی که قبل از

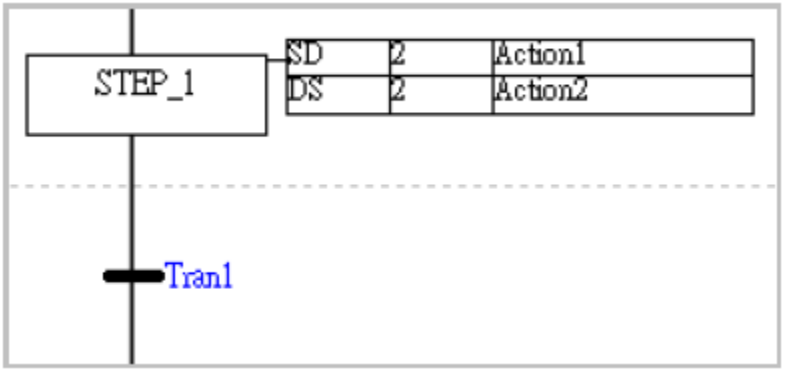
گذشت زمان تاخیر، پله غیرفعال شود، عملکرد اصلا اجرا نخواهد شد و این موضوع در اجراهای آتی نیز تاثیری نخواهد داشت. در مثال زیر، عملکرد Action به محض فعال شدن STEP_1 اجرا نمی‌شود بلکه دو ثانیه صبر می‌کند و سپس اجرا می‌شود. اگر STEP_1 در این دو ثانیه غیرفعال شود، عملکرد Action نیز اصلا اجرا نخواهد شد.



شکل ۱۰-۷۱ اختصاص عملکرد نوع Delay

- SD (Set Delay): در صورتی که پله آن فعال شود، به میزان تاخیر صبر کرده و بعد اجرا شده و ست می‌شود (یعنی پس از غیرفعال شدن پله نیز اجرای آن متوقف نمی‌شود). در صورتی که قبل از گذشتن زمان تاخیر، پله غیر فعال شود، آنگاه تاخیر همچنان محاسبه و پس از طی شدن مدت زمان مورد نظر، عملکرد اجرا می‌شود.
- DS (Delay Set): همانند SD با این تفاوت که در صورت غیرفعال شدن پله قبل از گذشتن زمان تاخیر، عملکرد دیگر اجرا نمی‌شود.

برای درک تفاوت دو عملکرد SD و DS به مثال زیر توجه کنید:



شکل ۱۰-۷۲ مقایسه دو عملکرد SD و DS

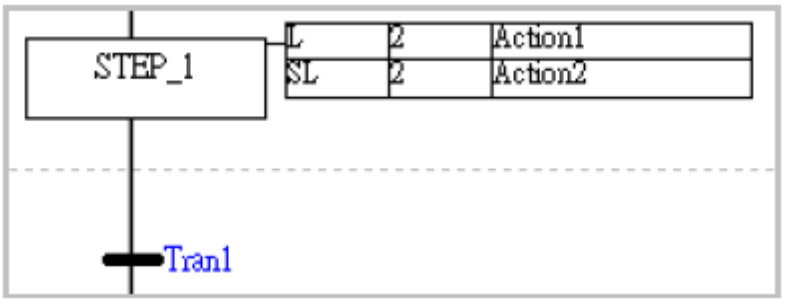
در مثال فوق در صورت فعال شدن STEP_1 دو حالت ممکن است رخ دهد.

الف) در صورتی که STEP_1 برای بیش از دو ثانیه فعال بماند، Action1 و Action2 پس از دو ثانیه فعال شده و مستقل از فعال بودن یا نبودن STEP_1، فعال خواهند ماند.

ب) در صورتی که STEP_1 برای کمتر از دو ثانیه فعال بماند، پس از عبور کنترل به پله بعد، تاخیر Action1 همچنان شمرده می‌شود و پس از دو ثانیه این عملکرد اجرا می‌شود، در حالی که Action2 اجرا نخواهد شد.

- L (Limit): همانند نوع N با این تفاوت که اگر عملکرد آن برای بیش از مدت مشخصی طول بکشد، اجرای آن متوقف و اسکن نهایی برای آن اجرا خواهد شد.
- SL (Set Limit): نوع L-ای که در صورت اجرای عملکرد، تنها اتمام مدت زمان محدود شده برای آن، عملکرد را متوقف می‌کند و غیرفعال شدن پله نیز نمی‌تواند آن را متوقف کند.

برای درک تفاوت دو عملکرد L و SL به مثال زیر توجه کنید:



شکل ۱۰-۷۳ مقایسه دو عملکرد L و SL

در مثال فوق در صورت فعال شدن STEP_1 ، هر دو عملکرد Action1 و Action2 فعال می‌شوند. در اینجا دو حالت ممکن است رخ دهد.

الف) در صورتی که STEP_1 برای بیش از دو ثانیه فعال بماند، Action1 و Action2 برای دوثانیه اجرا و بعد متوقف می‌شوند.

ب) در صورتی که STEP_1 برای کمتر از دو ثانیه فعال بماند، پس از عبور کنترل به پله بعد، اجرای Action1 متوقف می‌شود ولی اجرای Action2 برای دو ثانیه ادامه خواهد داشت و سپس متوقف خواهد شد.

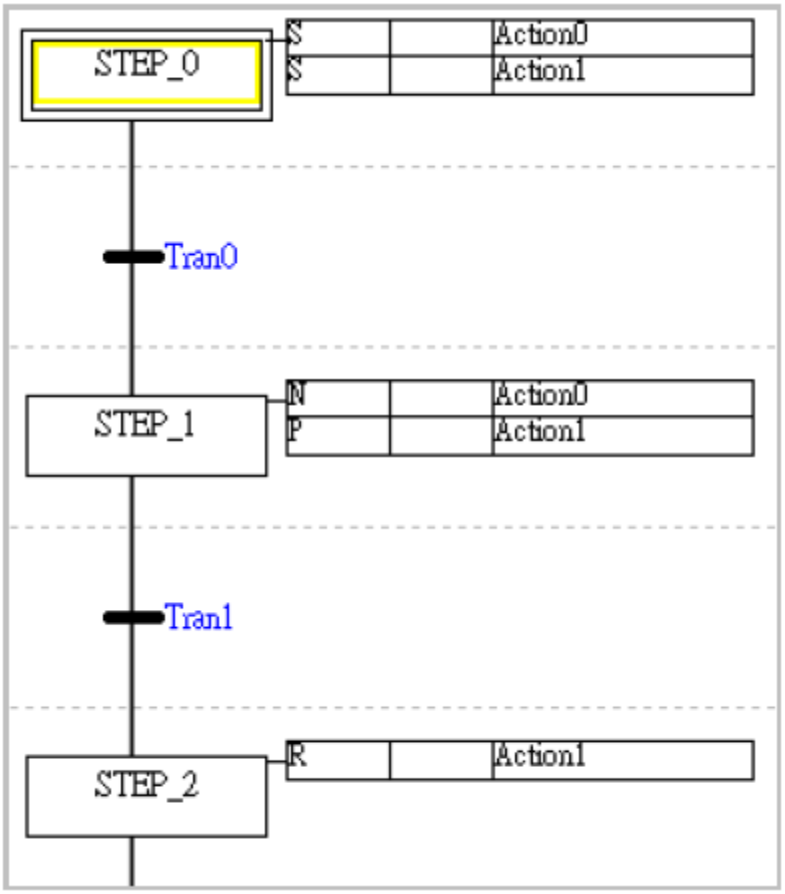
- R (Reset): این نوع عملکرد در صورت فعال شدن پله آن، عملکرد مورد نظر خود را غیر فعال می‌کند (حتی اگر در پله های قبل توسط S و SD و DS و SL فعال شده باشد) و سپس اسکن نهایی برای آن اجرا خواهد شد.

- P (Pulse): تنها در اسکن اول در صورتی که پله آن فعال باشد، عملکرد اجرا خواهد شد. و پس از غیر فعال شدن پله آن، اسکن نهایی اجرا می‌شود.
- P1 (Raising Pulse): تنها در اسکن اول در صورتی که پله آن فعال باشد، عملکرد اجرا خواهد شد. ولی پس از غیر فعال شدن عملکرد، اسکن نهایی اجرا نمی‌شود.
- P0 (Falling Pulse): تنها در اسکن اول در صورتی که پله آن غیرفعال باشد، عملکرد اجرا خواهد شد. ولی پس از غیر فعال شدن عملکرد، اسکن نهایی اجرا نمی‌شود.

۱۰-۳-۱-۷- نکاتی مهم پیرامون بررسی عملکردها

- پس از بررسی شدن عملکردی به وسیله بررسی کننده‌های نوع S، SD، DS و یا SL در صورت ست شدن آن‌ها، اجرای آن‌ها حتی با بررسی مجدد آن‌ها به وسیله بررسی کننده‌های نوع N یا P متوقف نمی‌شود. تنها بررسی کننده R است که می‌تواند جلوی اجرای آن‌ها را بگیرد. بنابر این لازم است هرجا از بررسی کننده‌های نوع S، SD، DS و یا SL استفاده می‌شود، به ضرورت توقف عملکردهای مورد نظر با استفاده از بررسی کننده R توجه شود.

در مثال زیر با فعال شدن STEP_0 عملکردهای Action1 و Action2 فعال می‌شوند. پس از عبور کنترل به STEP_1 این دو عملکرد همچنان اجرا خواهند شد (حتی اگر با N و P بررسی شوند، اجرای آن‌ها متوقف نمی‌شود). پس از رسیدن کنترل به STEP_2، Action1 متوقف می‌شود ولی چون Action0 هیچگاه با بررسی کننده R مواجه نمی‌شود متوقف نیز نمی‌شود.

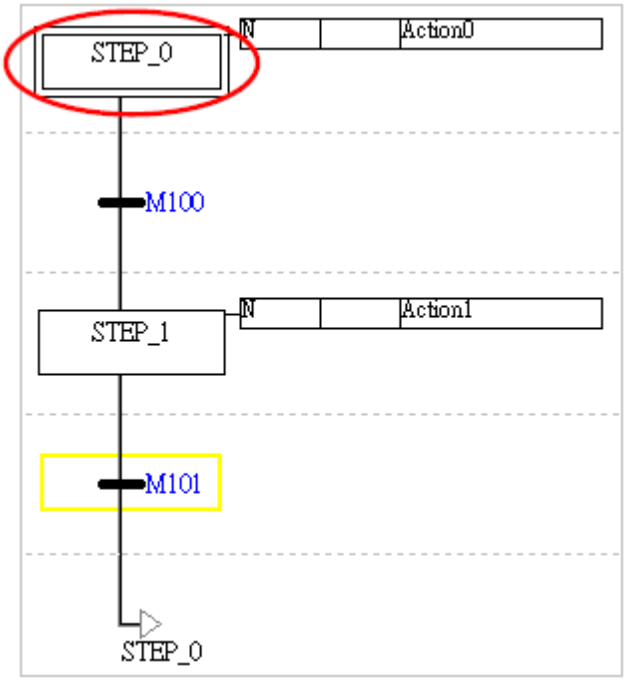


شکل ۱۰-۷۴ در صورت ست شدن عملکرد، تنها با ریست کردن می توان آن را متوقف کرد.

- هر عملکرد تنها یکبار می‌تواند به بررسی کننده زمانی اختصاص داده شود. یعنی در صورتی که عملکردی با یکی از بررسی کننده‌های D، SD، DS، L و یا SL مورد ارزیابی قرار گیرد، دیگر نمی‌تواند بار دیگر با هیچکدام از آن‌ها (بررسی کننده‌های زمانی) مورد بررسی قرار گیرد، در غیر این صورت در حین کامپایل خطا رخ خواهد داد.

۱۰-۳-۱-۸- پله اولیه

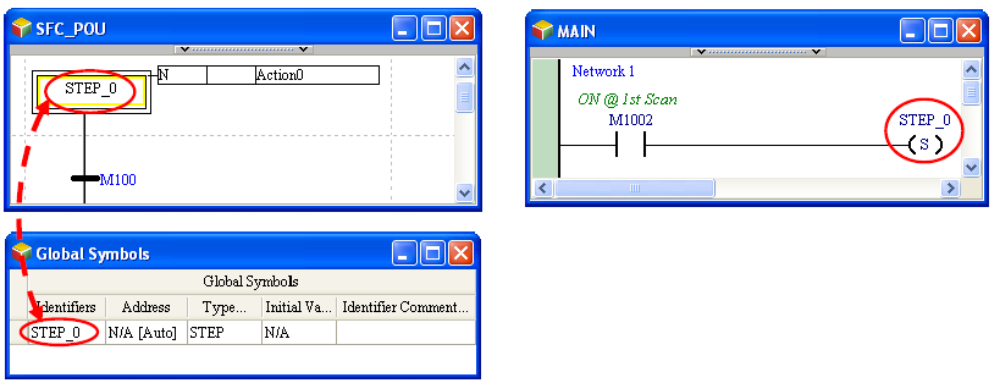
در زمان اجرای SFC، هرپله به نوبت و به صورت دوره‌ای فعال می‌شود. در این حال تعریف پله‌ی اولیه-ای که در لحظه‌ی شروع به کار برنامه، برای اولین بار فعال باشد، ضروری است (چون قبل آن گذاری برای فعال کردن آن پله وجود ندارد). در ISPSoft پله اولیه با مستطیلی دو خطی مشخص می‌شود و در هر SFC تنها می‌توان یک پله اولیه می‌توان تعریف کرد.



شکل ۱۰-۷۵ پله اولیه در SFC

هر پله در خانواده AH500 را می‌توان به عنوان پله اولیه تعریف کرد. در این خانواده پله اولیه، اولین پله فعال در زمان فراخوانی SFC می‌باشد. اما در خانواده DVP امکان تعریف پله اولیه و جود ندارد و لازم است هرجا نیاز به فراخوانی SFC وجود دارد، از دستور SET برای فعال کردن پله اولیه در SFC مورد نظر استفاده شود. توجه شود که همانطور که قبلا گفته شد چون در اینجا نیاز به پرش بین دو POU وجود دارد، باید پله مقصد (پله اولیه‌ی SFC) در جدول سیمبول سراسری تعریف شود. مثالی از این موضوع در ادامه آمده است.

در مثال زیر بعد از آنکه SET بر روی پله STEP_0 اعمال شد (تنها در سیکل اول نیز اجرا می‌شود؛ در واقع باید کنترل شود که در سیکل‌های دیگر دوباره فعال نشود تا در میانه اجرای SFC، خللی ایجاد نشود) برنامه SFC، پله STEP_0 را به عنوان پله اولیه اجرا می‌کند.

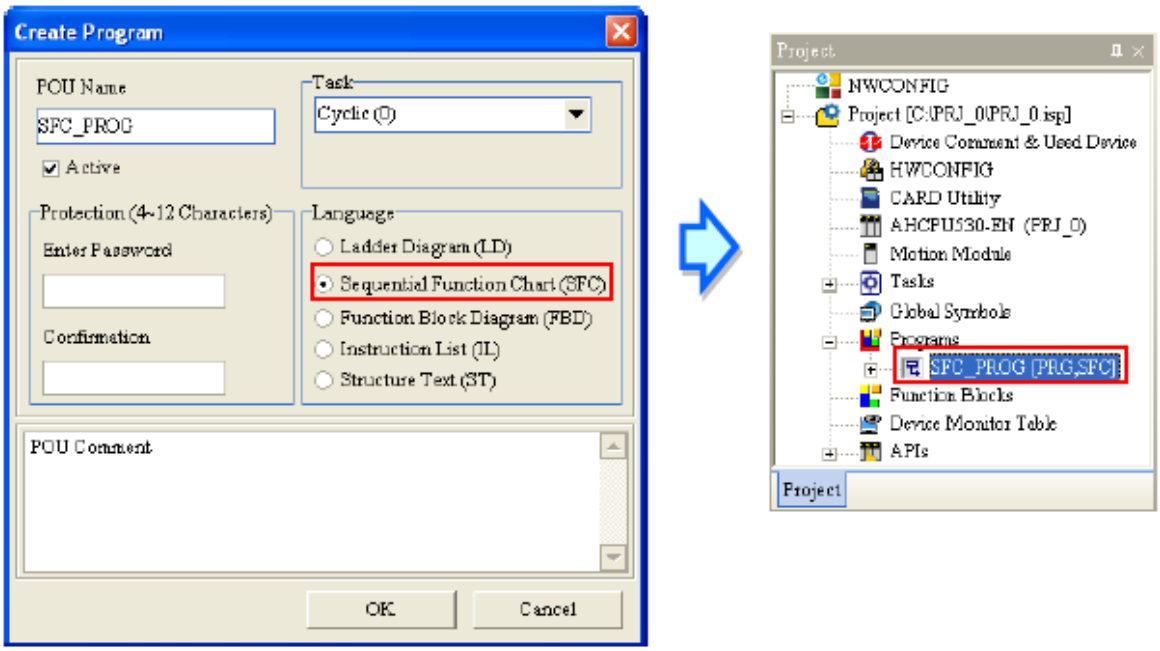


شکل ۱۰-۷۶ پیاده سازی مکانیزم پله اولیه در خانواده DVP

۱۰-۳-۲- ساخت SFC در ISPSoft

۱۰-۳-۲-۱- محیط ویرایش برنامه

گزینه Sequential Function Chart (SFC) را در قسمت زبان برنامه نویسی در پنجره Creat Program انتخاب کنید. به این نکته بسیار مهم توجه کنید که از SFC برای ساخت توابع بلوکی نمی‌شود استفاده کرد. همچنین تنها برنامه‌های دوره‌ای را می‌توان با این زبان برنامه نویسی کرد.



شکل ۱۰-۷۷ ساخت برنامه به زبان SFC

محیط ویرایش برنامه به زبان SFC در شکل زیر نمایش داده شده است، قسمت بالای این برنامه محل تخصیص سیمبول‌های محلی و قسمت پایین آن مربوط به محیط ویرایش برنامه است. محیط ویرایش برنامه شطرنجی و گذارها و پله‌ها درون خانه‌های مربعی آن قرار دارند. در صورتی که بر روی مربعی دارای عضوی کلیک شود، آن عضو انتخاب و رنگ آن زرد می‌شود.

شکل ۱۰-۷۸ محیط برنامه نویسی به زبان SFC

پس از آنکه محیط ویرایش برنامه به زبان SFC فعال شد، نوار ابزار مربوط به آن نیز در بالای نرم افزار ظاهر خواهد شد. این نوار ابزار و امکانات آن را در ادامه مرور خواهیم کرد.

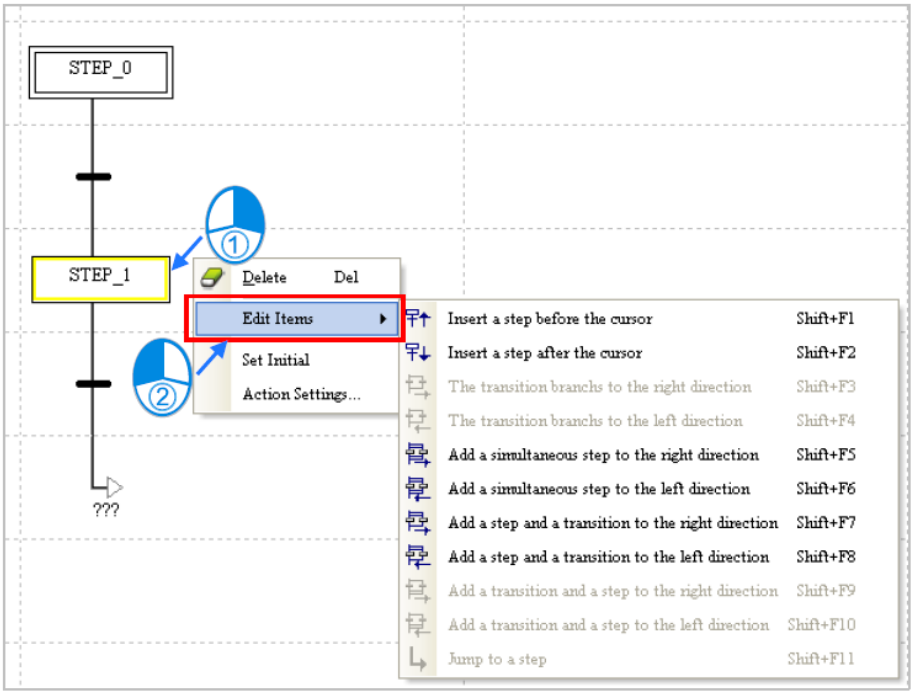
شکل ۱۰-۷۹ نوار ابزار برنامه نویسی به زبان SFC

جدول ۱۰-۷: آیکون های نوار ابزار برنامه نویسی SFC

نماد	کلید میانبر	شرح
	Shift+F1	اضافه شدن پله به بالای قسمت انتخاب شده.
	Shift+F2	اضافه شدن پله به پایین قسمت انتخاب شده.
	Shift+F3	اتصال گذار موازی در سمت راست گذار انتخاب شده

	Shift+F4	اتصال گذار موازی در سمت چپ گذار انتخاب شده
	Shift+F5	اتصال پله در سمت راست پله انتخاب شده
	Shift+F6	اتصال پله در سمت چپ پله انتخاب شده
	Shift+F7	ساخت انشعاب همزمان و همگرایی انتخاب ترتیبی در سمت راست پله انتخاب شده
	Shift+F8	ساخت انشعاب همزمان و همگرایی انتخاب ترتیبی در سمت چپ پله انتخاب شده
	Shift+F9	ساخت انشعاب انتخاب ترتیبی و همگرایی همزمان در سمت راست پله انتخاب شده
	Shift+F10	ساخت انشعاب انتخاب ترتیبی و همگرایی همزمان در سمت چپ پله انتخاب شده
	Shift+F11	تخصیص پرش

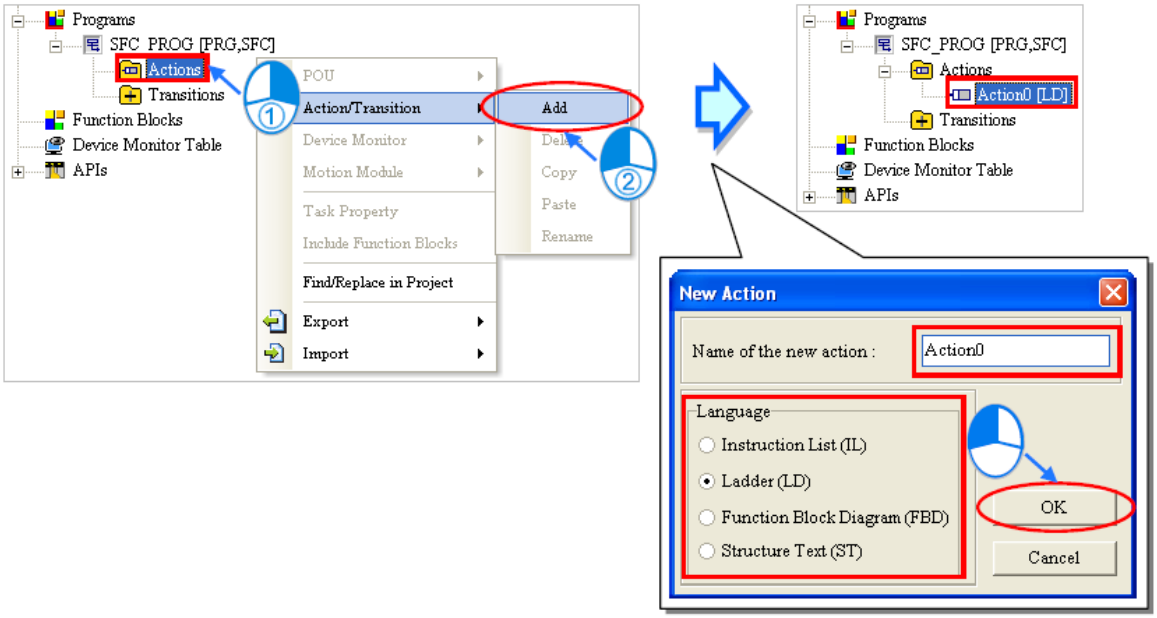
بسیاری از این امکانات با کلیک راست بر روی اشیای برنامه و در قسمت Edit Items نیز قابل دسترس هستند.



شکل ۱۰-۸۰ امکان ویرایش برنامه SFC در محیط برنامه

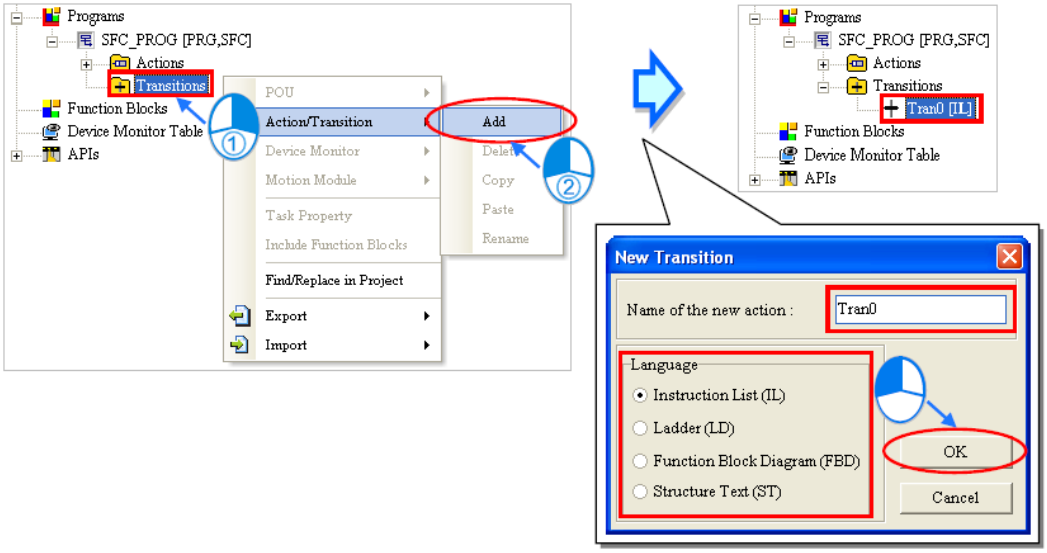
۱۰-۳-۲-۲- ساخت عملکرد ها و گذارها

برای ساخت عملکرد، لازم است در بخش مدیریت پروژه در بخش Programs، در SFC مورد نظر Actions را انتخاب و پس از کلیک راست بر روی آن، در Action/Transition گزینه Add را انتخاب کرد. در پنجره باز شده، ضمن انتخاب نام عملکرد باید زبان مورد نظر برای برنامه نویسی آن را نیز انتخاب و سپس تایید کرد.



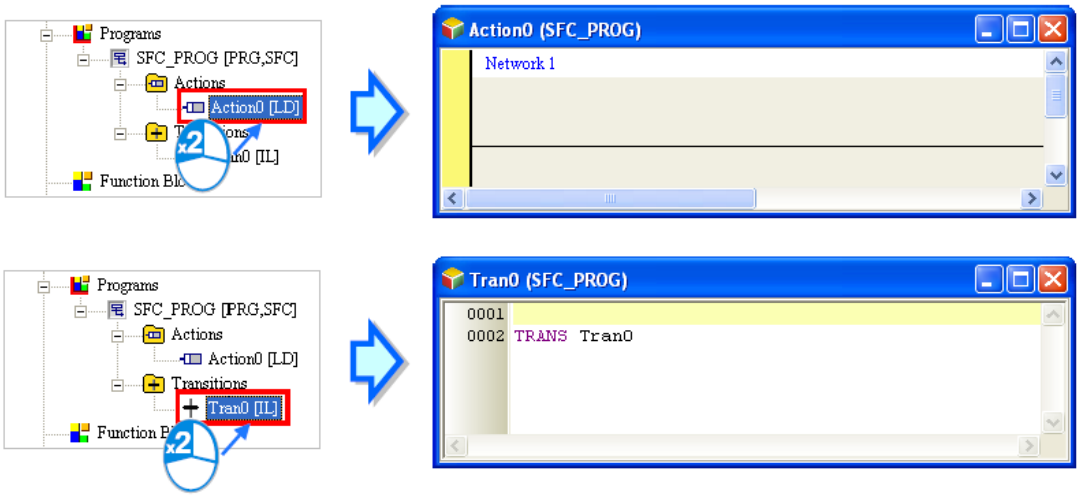
شکل ۱۰-۸۱ ساخت عملکرد

برای ساخت گذار نیز لازم است در بخش مدیریت پروژه در بخش Programs، در SFC مورد نظر Transition، را انتخاب و پس از کلیک راست بر روی آن، در Action/Transition گزینه Add را انتخاب کرد. در پنجره باز شده، ضمن انتخاب نام گذار باید زبان مورد نظر برای برنامه نویسی آن را نیز انتخاب و سپس تایید کرد.



شکل ۱۰-۸۲ ساخت گذار

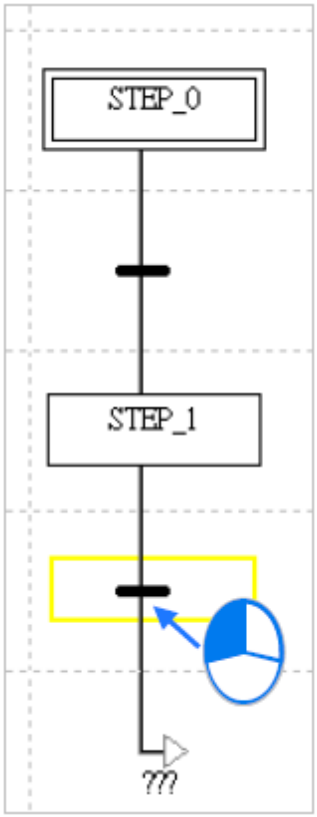
پس از ساخته شدن برنامه گذار و یا عملکرد با دوبار کلیک کردن بر روی آن‌ها در بخش مدیریت پروژه، پنجره ویرایش برنامه آن‌ها باز خواهد شد. همانطور که قبلا گفته شد، در برنامه گذار سیمبولی از پیش تعریف شده با نام همان گذار وجود دارد که باید نتیجه برنامه را به آن اختصاص داد. برنامه‌های عملکردها و گذارها را همانند POUهای معمولی می‌توان ویرایش کرد. با این حال جدول سیمبول محلی برای آن‌ها وجود نخواهد داشت و کاربر باید سیمبول‌های مورد نظر خود را در جدول سیمبول‌های محلی SFC وارد کند.



شکل ۱۰-۸۳ محیط ویرایش برنامه عملکرد و گذار

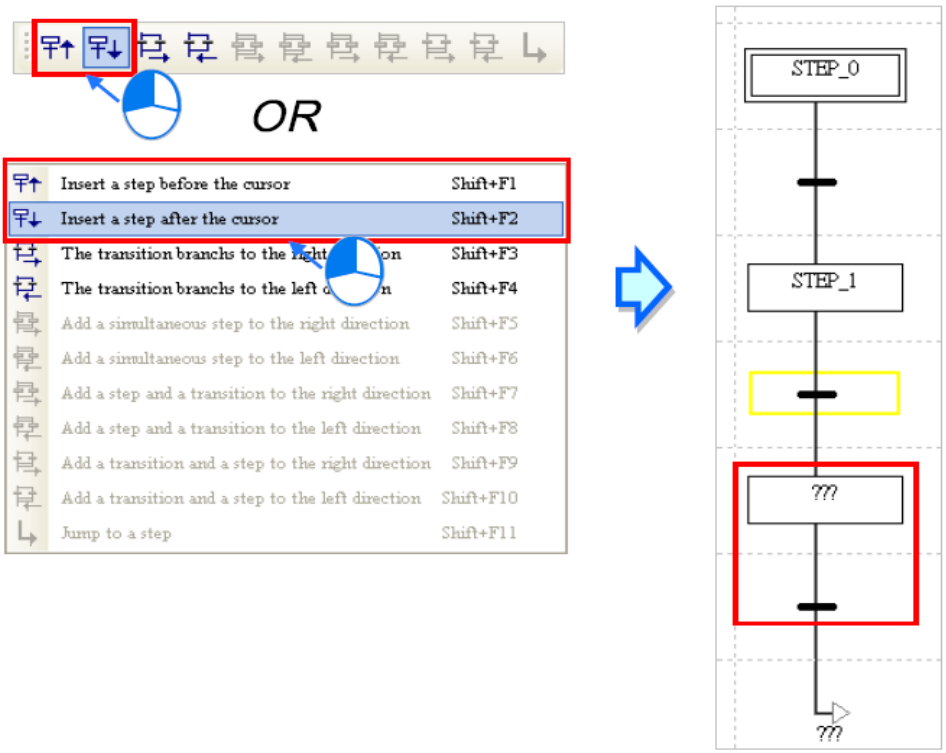
۱۰-۳-۲-۳- اضافه کردن پله

پله (مجموعه پله و گذار بعد آن) که می‌خواهید پله جدید را در اطراف آن اضافه کنید، انتخاب کنید.



شکل ۱۰-۸۴ انتخاب محل اضافه شدن پله جدید

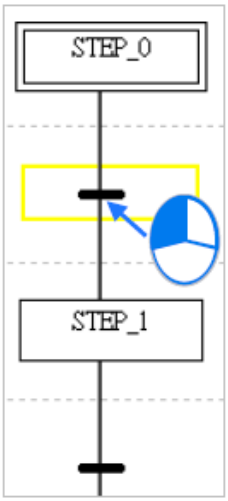
در صورتی که لازم است پله‌ای در بالای محل انتخابی اضافه شود بر روی  و در صورتی که بخواهید پله‌ای در پایین محل انتخابی اضافه شود بر روی  کلیک کنید. (این فرایندها را با کلیک راست و انتخاب گزینه‌های منوی Edit Items نیز می‌توان انجام داد).



شکل ۱۰-۸۵ اضافه کردن پله به SFC

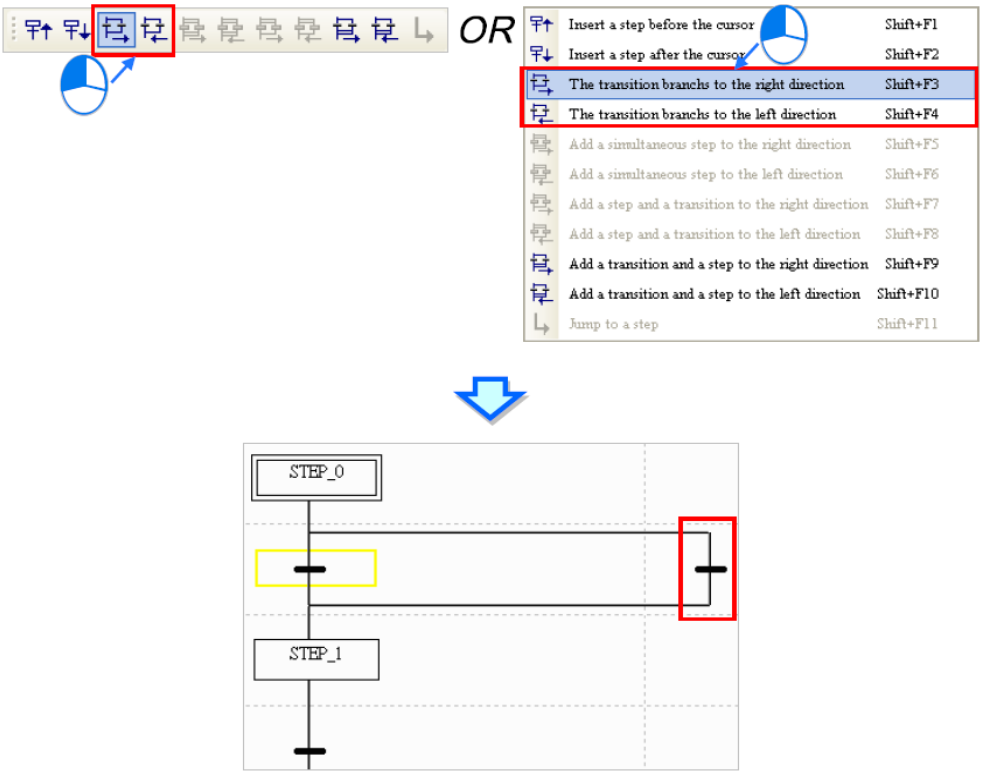
۱۰-۳-۲-۴- اتصال گذارها به صورت موازی

گذار مورد نظر را در محیط ویرایش برنامه انتخاب نمایید.



شکل ۱۰-۸۶ انتخاب محل اضافه شدن گذار موازی

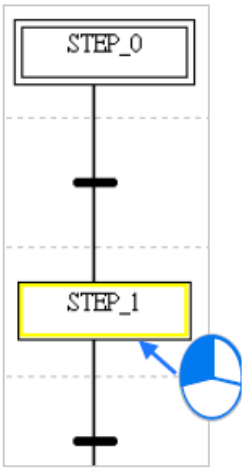
اگر لازم به اضافه کردن به گذاری در سمت راست گذار انتخابی باشد بر روی  کلیک و در صورتی که لازم به اضافه کردن گذاری در سمت چپ گذار انتخابی باشد بر روی  کلیک نمایید. (این فرایندها را با کلیک راست و انتخاب گزینه‌های منوی Edit Items نیز می‌توان انجام داد).



شکل ۱۰-۸۷ اضافه کردن گذار موازی

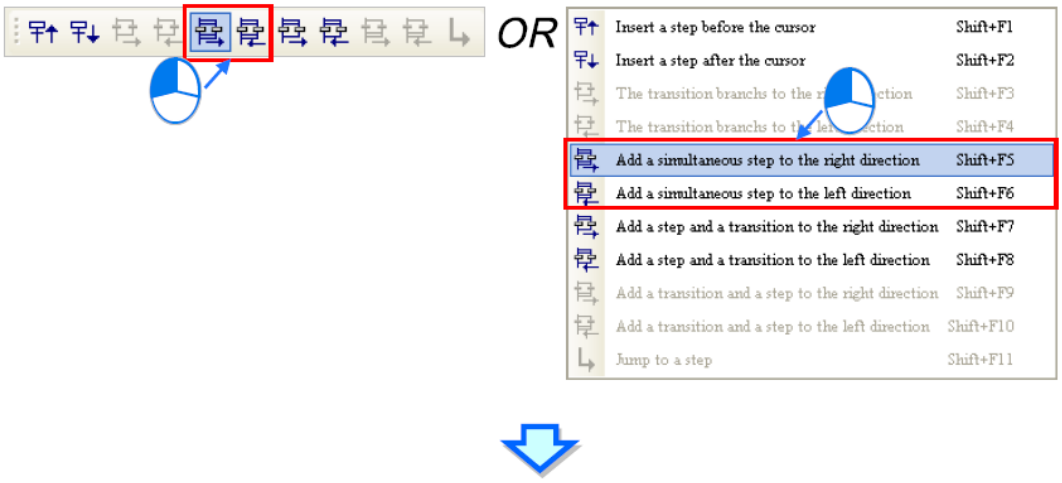
۱۰-۳-۲-۵- اتصال پله‌ها به صورت موازی

پله مورد نظر را انتخاب نمایید (توجه کنید که بالاترین پله یعنی پله اولیه را نمی‌توان موازی کرد)



شکل ۱۰-۸۸ انتخاب محل اضافه شدن پله موازی

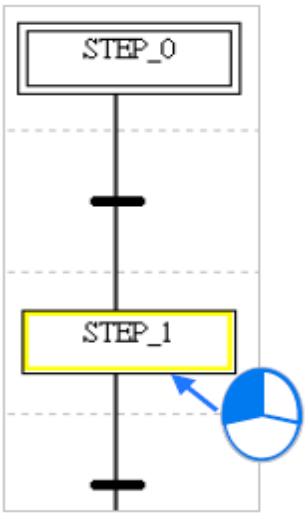
در صورتی که لازم است پله‌ای در سمت راست پله انتخابی اضافه شود بر روی  و در صورتی که بخواهید پله‌ای در سمت چپ پله انتخابی اضافه شود بر روی  کلیک کنید. (این فرایندها را با کلیک راست و انتخاب گزینه‌های منوی Edit Items نیز می‌توان انجام داد).



شکل ۱۰-۸۹ اضافه کردن پله موازی

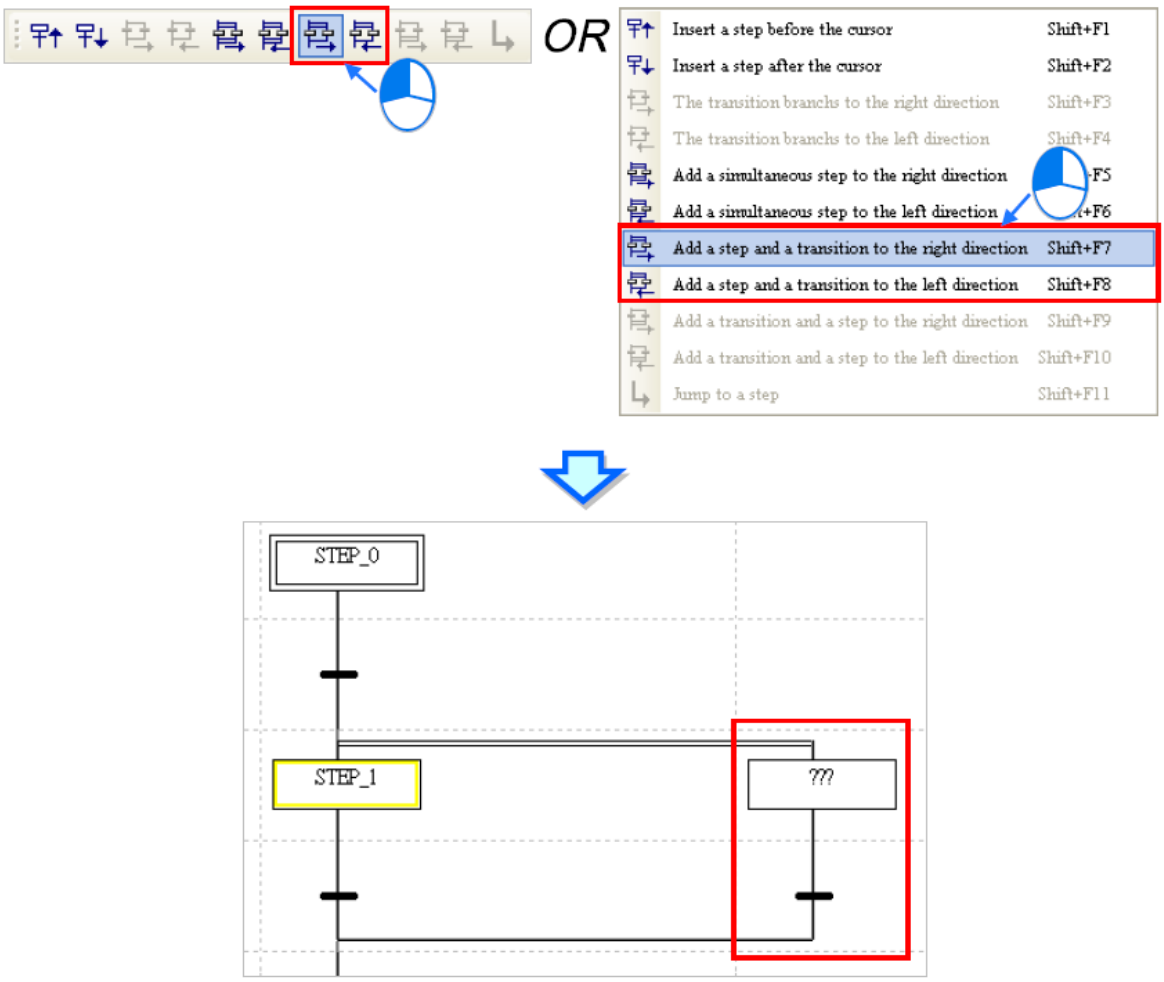
۱۰-۳-۲-۶- ساخت انشعاب همزمان و همگرایی انتخاب ترتیبی

طبق توضیحات داده شده در بخش‌های قبل پیرامون انواع انشعاب و همگرایی در این بخش می-خواهیم نحوه ساخت انشعاب همزمان و همگرایی انتخاب ترتیبی را بررسی نماییم. پله مورد نظر را انتخاب نمایید (توجه کنید که بالاترین پله یعنی پله اولیه را نمی‌توان برای اینکار انتخاب کرد)



شکل ۱۰-۹۰ انتخاب محل اضافه شدن انشعاب همزمان و همگرایی انتخاب ترتیبی

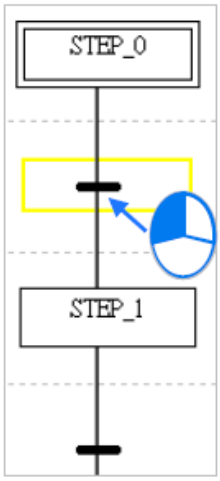
در صورتی که لازم است "انشعاب همزمان و همگرایی انتخاب ترتیبی" در سمت راست پله انتخابی اضافه شود بر روی  و در صورتی که بخواهید "انشعاب همزمان و همگرایی انتخاب ترتیبی" در سمت چپ پله انتخابی اضافه شود بر روی  کلیک کنید. (این فرایندها را با کلیک راست و انتخاب گزینه‌های منوی Edit Items نیز می‌توان انجام داد).



شکل ۱۰-۹۱ ساخت انشعاب همزمان و همگرایی انتخاب ترتیبی

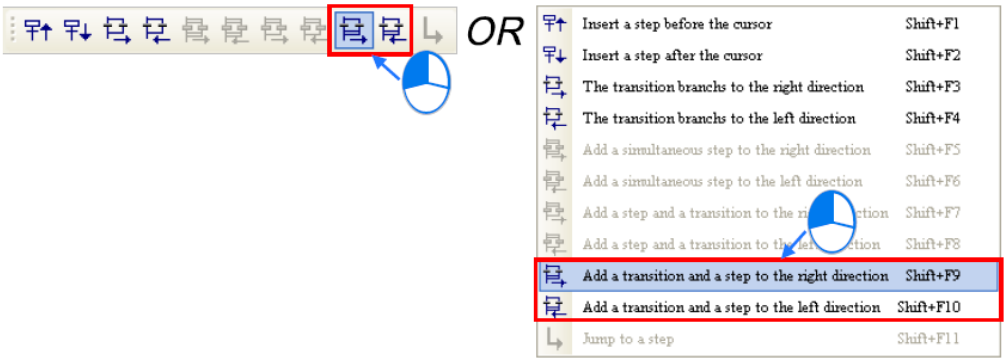
۱۰-۳-۲-۷- ساخت انشعاب انتخاب ترتیبی و همگرایی همزمان

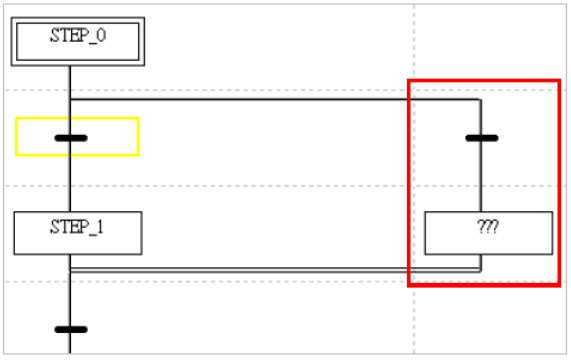
طبق توضیحات داده شده در بخش‌های قبل پیرامون انواع انشعاب و همگرایی در این بخش می-خواهیم نحوه ساخت انشعاب انتخاب ترتیبی و همگرایی همزمان را بررسی نماییم. گذار مورد نظر را انتخاب نمایید.



شکل ۱۰-۹۲ انتخاب محل اضافه شدن انشعاب انتخاب ترتیبی و همگرایی همزمان

در صورتی که لازم است "انشعاب انتخاب ترتیبی و همگرایی همزمان" در سمت راست پله انتخابی اضافه شود بر روی  و در صورتی که بخواهید "انشعاب انتخاب ترتیبی و همگرایی همزمان" در سمت چپ پله انتخابی اضافه شود بر روی  کلیک کنید. (این فرایندها را با کلیک راست و انتخاب گزینه‌های منوی Edit Items نیز می‌توان انجام داد).

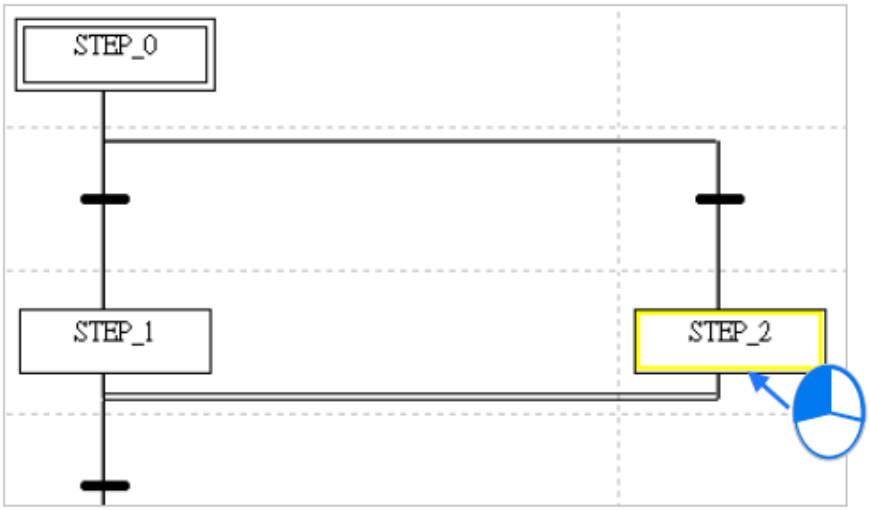




شکل ۱۰-۹۳ ساخت انشعاب انتخاب ترتیبی و همگرایی همزمان

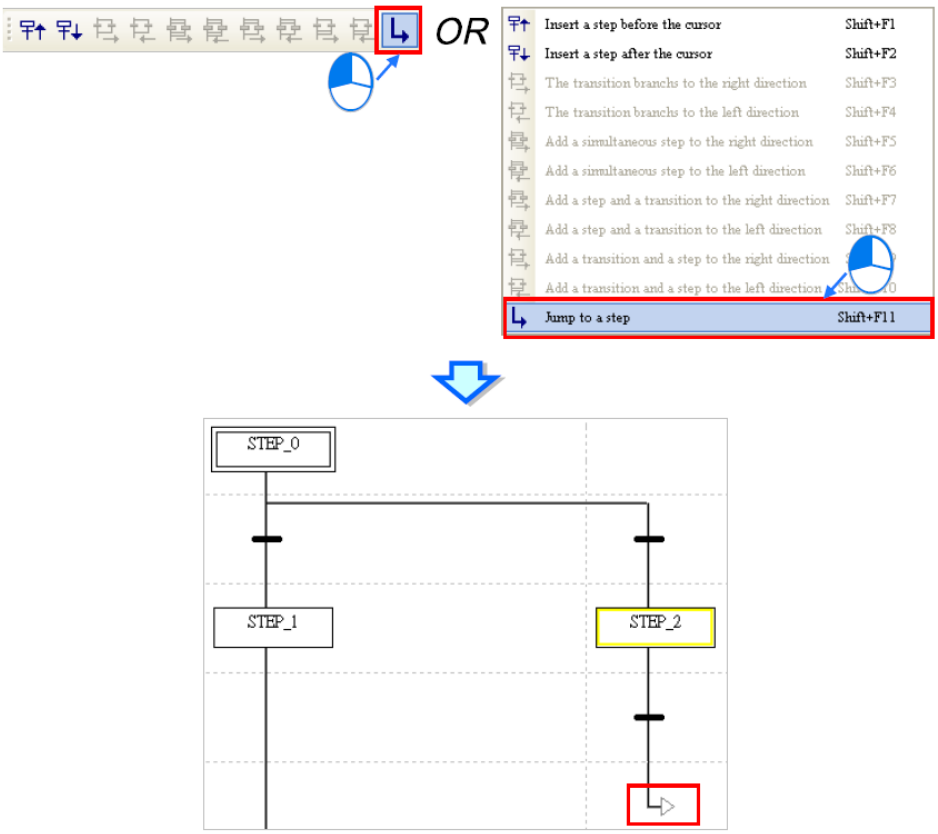
۱۰-۳-۲-۸- اختصاص نقطه پرش

انتهایی ترین پله یا گذار در مسیری موازی را انتخاب کنید.



شکل ۱۰-۹۴ انتخاب محل اضافه شدن نقطه پرش

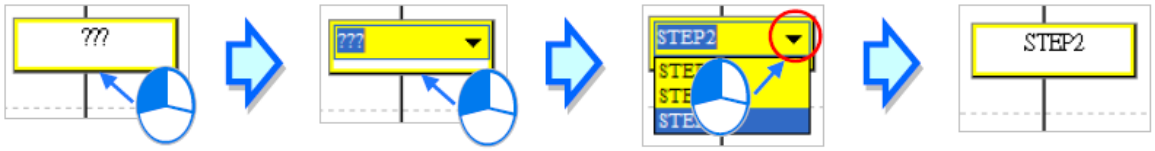
پس از کلیک بر روی  نقطه پرش ایجاد خواهد شد. این فرایندها را با کلیک راست در قسمت مورد نظر در برنامه و انتخاب گزینه Jump to a Step در منوی Edit Items نیز می‌توان انجام داد.



شکل ۱۰-۹۵ایجاد نقطه پرش در برنامه

۱۰-۳-۲-۹- اختصاص سیمبول به پله ها و گذارها

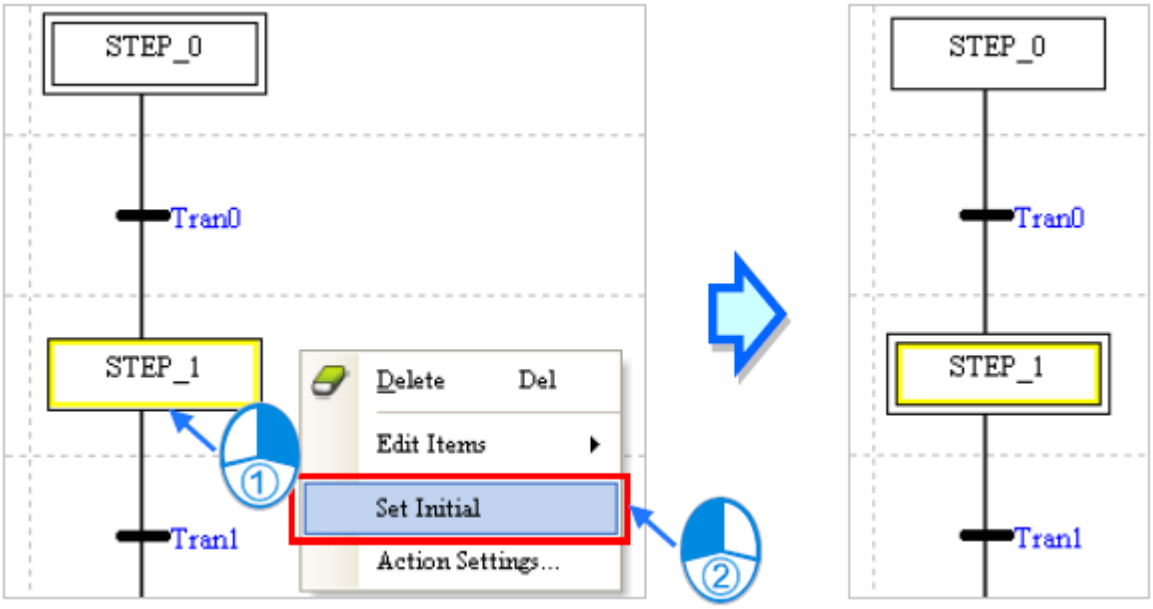
پس از پایان ویرایش ساختار برنامه SFC، باید به پله‌ها و گذارهای درون برنامه سیمبول اختصاص داد. برای اینکار ابتدا بر روی پله یا گذار مورد نظر یکبار کلیک نمایید تا رنگ آن زرد شود سپس دوباره بر روی آن کلیک نمایید. منوی باز شونده را انتخاب و سیمبول مورد نظر را انتخاب نمایید.



شکل ۱۰-۹۶ اختصاص سیمبول به پله ها و گذارها

۱۰-۳-۲-۱۰- تعیین پله اولیه

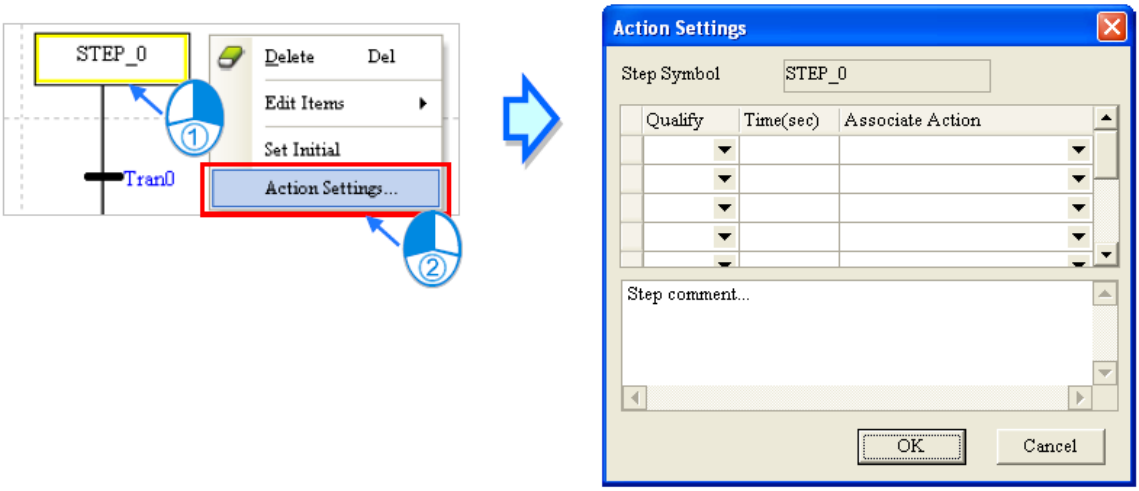
برای فعال سازی پله اولیه در PLCهای خانواده AH500 لازم است بر روی پله مورد نظر کلیک راست کرده و Set Initial را انتخاب کرد.



شکل ۱۰-۹۷ تعیین پله اولیه

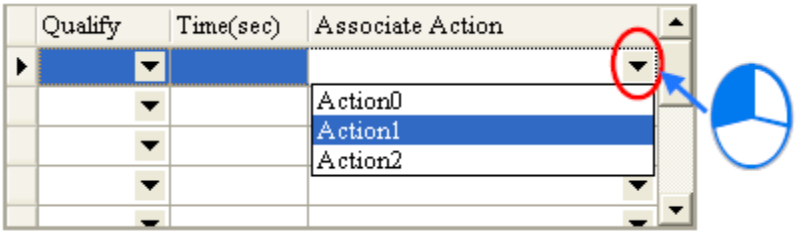
۱۰-۳-۲-۱۱- تخصیص عملکرد

بر روی پله‌ای که می‌خواهید به آن عملکرد اختصاص دهید کلیک راست کنید و گزینه Action Setting را انتخاب کنید. در صورتی که هنوز پله معرفی نشده باشد در این مرحله سیستم اجازه تخصیص عملکرد را نخواهد داد.



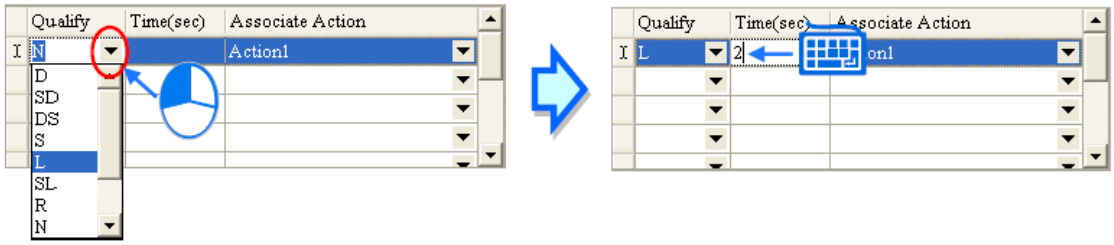
شکل ۱۰-۹۸ اختصاص پله برای تخصیص عملکرد

در قسمت Associate Action عملکرد مورد نظر را انتخاب نمایید.



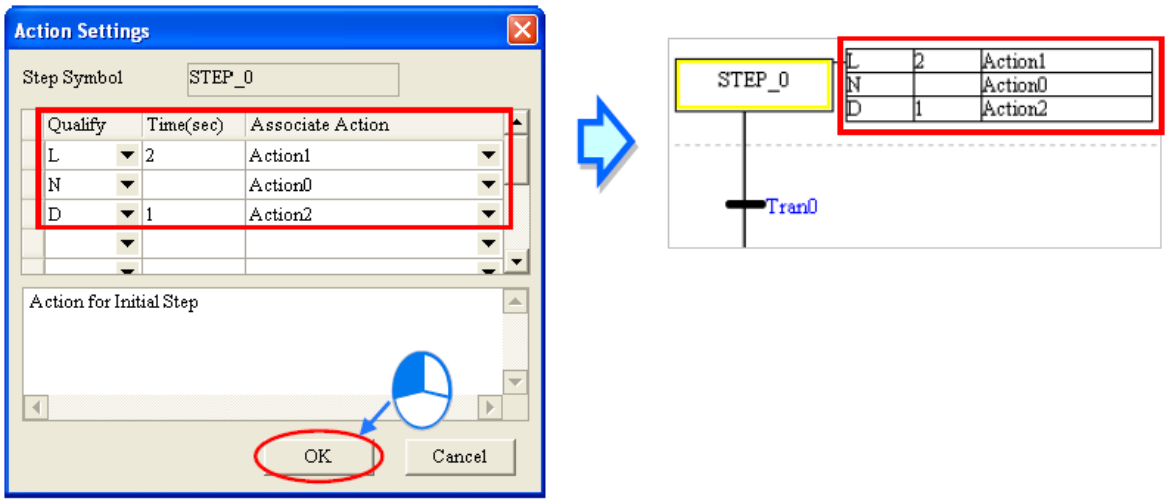
شکل ۱۰-۹۹ انتخاب عملکرد

سپس بررسی کننده مورد نظر برای عملکرد را در ستون Qualify انتخاب نمایید. در صورتی که بررسی کننده مورد نظر "زمانی" باشد باید مدت زمان مرتبط با آن در ستون Time بر حسب ثانیه وارد شود. (توجه شود که بررسی کننده در سری DVP قابل انتخاب نخواهد بود و نوع آن به صورت پیش فرض N است).



شکل ۱۰-۱۰۰ تنظیم بررسی کننده برای عملکرد

در نهایت نیز با کلیک بر روی OK عملکردهای مورد نظر اختصاص داده خواهند شد و در سمت راست پله نمایش داده می‌شوند.

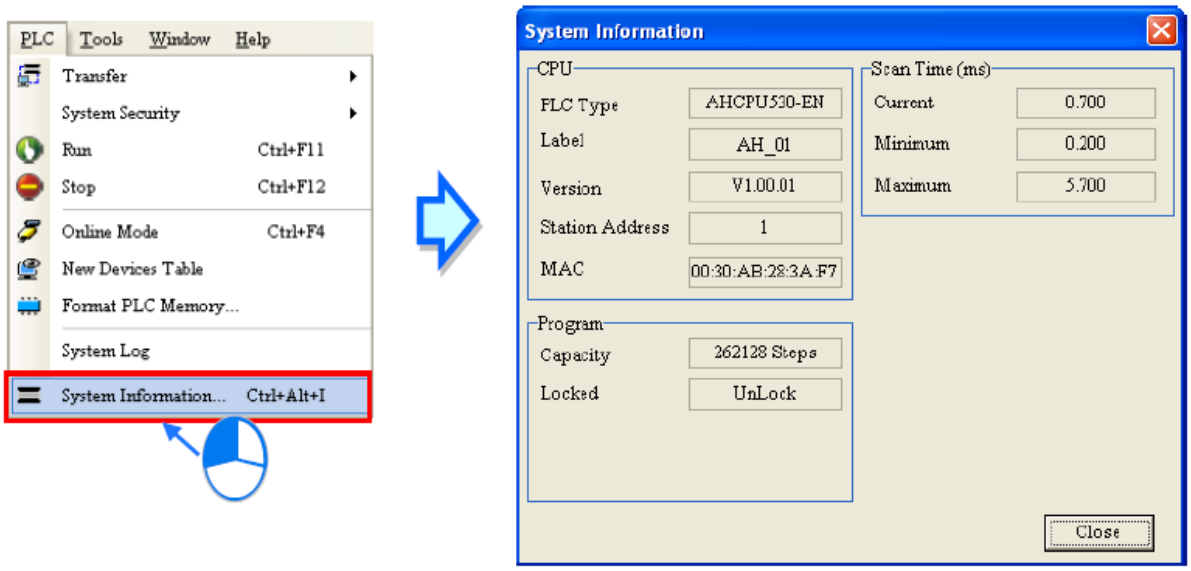


شکل ۱۰-۱۰۱ نمایش عملکردهای اختصاص داده شده به برنامه

فصل ۱۱– شبیه ساز PLC ، تِست و عیب یابی

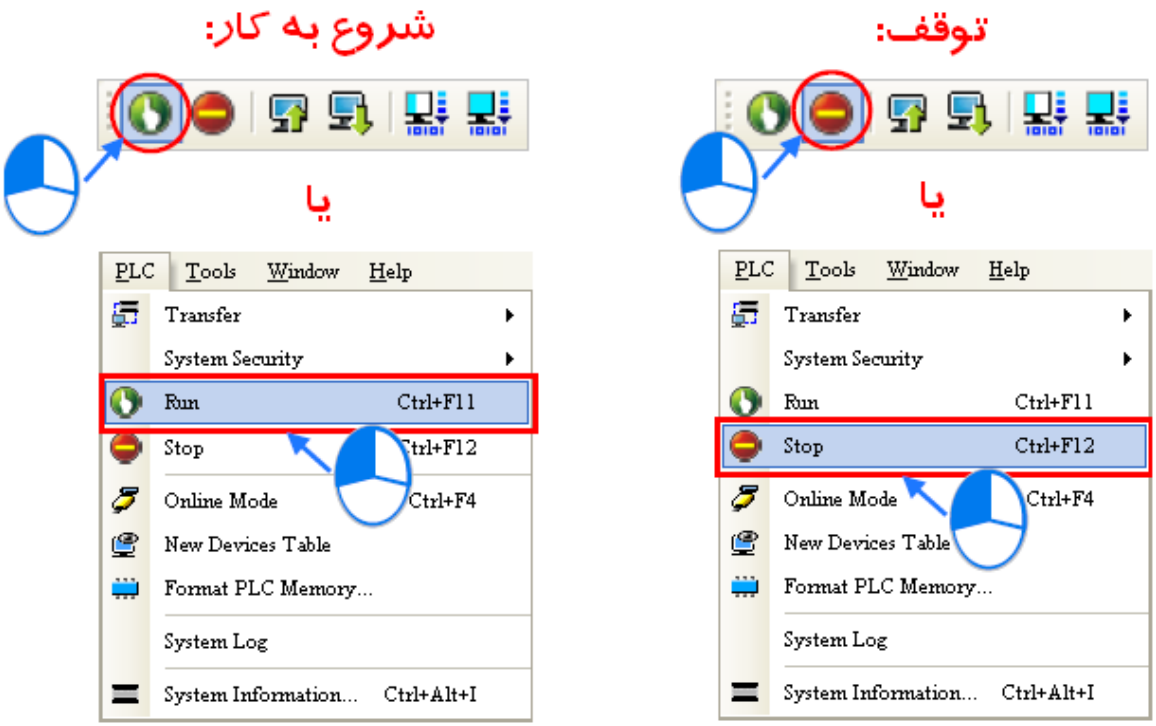
۱۱–۱– مانیتورینگ آنلاین

کاربران برای بررسی عملکرد PLC و تکمیل برنامه خود نیاز دارند نحوه اجرای برنامه نوشته شده خود بر روی PLC را به صورت آنلاین بررسی کنند. برای اینکار ابتدا نیاز است تا در مورد ارتباط PLC با کامپیوتر خود مطمئن شویم، برای این کار در سربرگ PLC گزینه System Information را انتخاب کنید، اگر ارتباط PLC با کامپیوتر به صورت نرمال برقرار شود ، آنگاه صفحه System Information ظاهر شده و اطلاعات رسیده از PLC نمایش داده می‌شود.



شکل ۱۱-۱ برقراری ارتباط بین ISPSoft با PLC

برای اینکه PLC شروع به کار کند می‌توان Run را در سربرگ PLC انتخاب و یا بر روی کلیک کرد. در صورتی که نیاز به توقف PLC باشد، می‌توان از سربرگ PLC گزینه Stop را انتخاب و یا بر روی کلیک کرد.



شکل ۱۱-۲ شروع به کار و توقف PLC با استفاده از ISPSoft

در صورتی که وضعیت اجرا/توقف PLC به وسیله ISPSoft تعیین شود، دیگر نباید وضعیت سوئیچ‌های روی PLC را ملاک قرار داد. در این حالت در صورتی که وضعیت سوئیچ‌های روی PLC تغییر کند، وضعیت اجرا در PLC نیز مطابق با آن تغییر خواهد کرد (در واقع چه در نرم افزار چه در سخت افزار، همیشه آخرین تغییرات اعمال خواهد شد).

برای مانیتور کردن آنلاین اجرای برنامه بارگزاری شده در PLC در نرم افزار ISPSoft دو روش وجود دارد. اولین روش حالت مانیتورینگ پورت‌ها و حافظه‌هاست و دیگری مانیتورینگ برنامه است.

جدول ۱۱-۱: روش های مانیتورینگ

حالت مانیتورینگ	شرح
مانیتورینگ پورت‌ها و حافظه‌ها	با استفاده از جدول مانیتورینگ می‌توان وضعیت حافظه‌ها و پورت‌ها را در PLC مانیتور کرد. در این حالت ISPSoft فقط نیاز به بروزرسانی وضعیت این حافظه‌ها و پورت‌ها دارد. در این روش لازم نیست برنامه نوشته شده در نرم افزار و PLC به یک شکل باشند.
مانیتورینگ برنامه	در صفحه ویرایش برنامه، با استفاده از این حالت می‌توان وضعیت کاری برنامه نوشته شده در تمامی Network ها را در لحظه مشاهده کرد. در این حالت به

علت آنکه برنامه نوشته شده در ISPSoft ملاک بررسی است، برنامه نرم افزار و PLC باید به یک صورت باشند.	

پس از کلیک بر روی آیکون و تغییر وضعیت سیستم به حالت آنلاین، سیستم دو حالت مانیتورینگ برنامه و حافظه/ پورت را فعال می‌کند.

شکل ۱۱-۳ تغییر وضعیت سیستم به حالت آنلاین

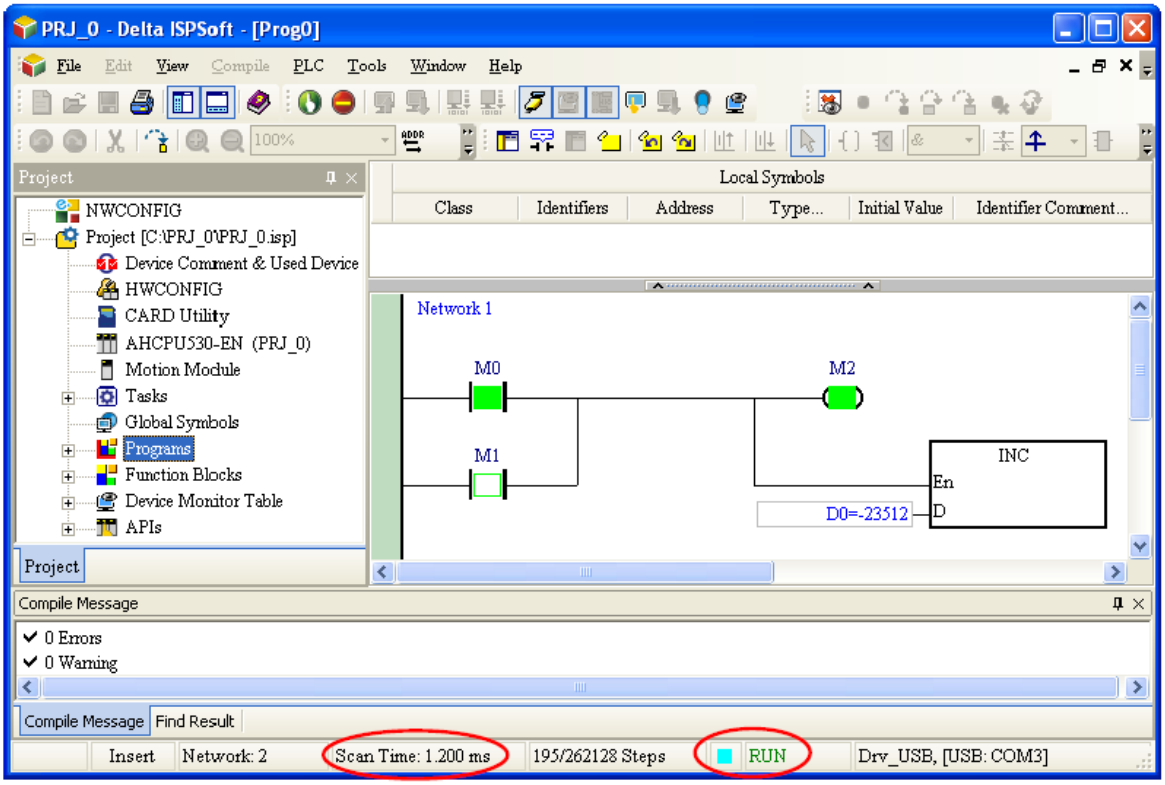
اگر کاربر فقط بخواهد وضعیت حافظه‌ها و پورت‌ها را بداند می‌تواند بر روی کلیک کند. در صورتی که کاربر بخواهد برنامه را نیز مانیتور کند باید بر روی هم کلیک کند.

شکل ۱۱-۴ فعال کردن مانیتورینگ پورت ها و حافظه و سپس مانیتورینگ برنامه

برای غیرفعال کردن حالت آنلاین، یکبار دیگر باید بر روی کلیک کرد.

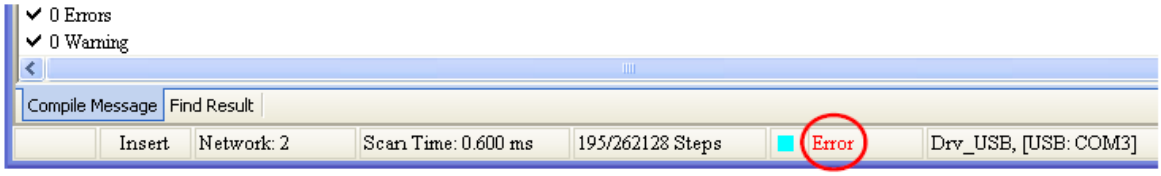
شکل ۱۱-۵ خارج شدن از حالت آنلاین

در حالت آنلاین، کاربر می‌تواند زمان SCAN فعلی، وضعیت ارتباطی، و وضعیت PLC را در نوار وضعیت مشاهده کند. در زمان اتصال ISPSoft با PLC نماد رنگی روی نوارِ وضعیت، چشمک زن خواهد شد.



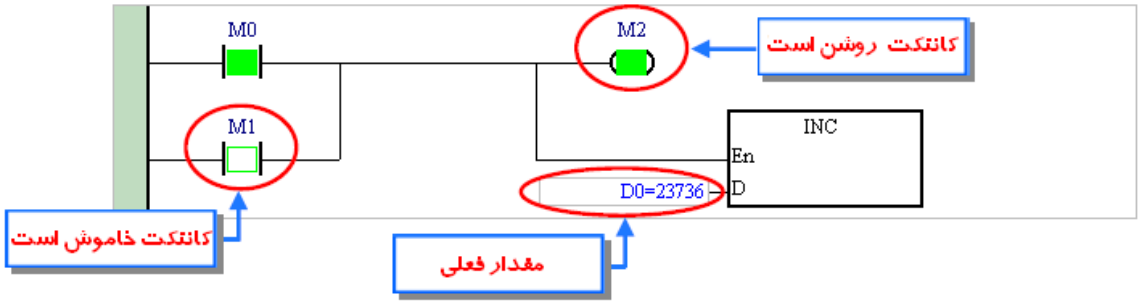
شکل ۱۱-۶ نوار وضعیت در حالت ارتباط آنلاین

در حالتی که خطایی در PLC رخ می‌دهد، کاربر نمی‌تواند با PLC کار کند، همچنین ممکن است PLC متوقف شود. در این حالت می‌توان نشان خطا را در نوار وضعیت دید.



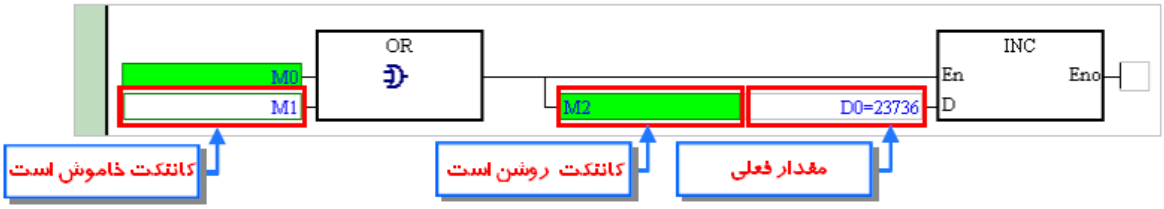
شکل ۱۱-۷ نوار وضعیت پس از رخ دادن خطا در PLC در حین ارتباط آنلاین

اگر بخواهیم وضعیت برنامه‌ای را به صورت آنلاین مانیتور کنیم، پس از فعال کردن مانیتورینگ برنامه باید POU مرتبط با آن را باز کنیم. در این حالت کاربر قادر به مشاهده وضعیت کارکردی PLC خواهد بود و از آن برای ارزیابی و ویرایش برنامه خود می‌تواند استفاده کند. نوع نمایش مانیتورینگ آنلاین برنامه PLC به زبان برنامه نویسی مورد استفاده بستگی دارد. در زیر نمایی از مانیتورینگ آنلاین برنامه به زبان Ladder را می‌بینید.



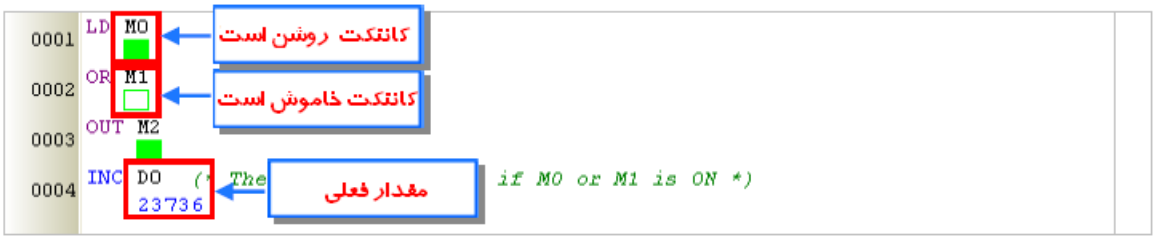
شکل ۱۱-۸ مانیتورینگ آنلاین برنامه به زبان Ladder

در زیر نمایی از مانیتورینگ آنلاین برنامه به زبان Function block diagram را می‌بینید.



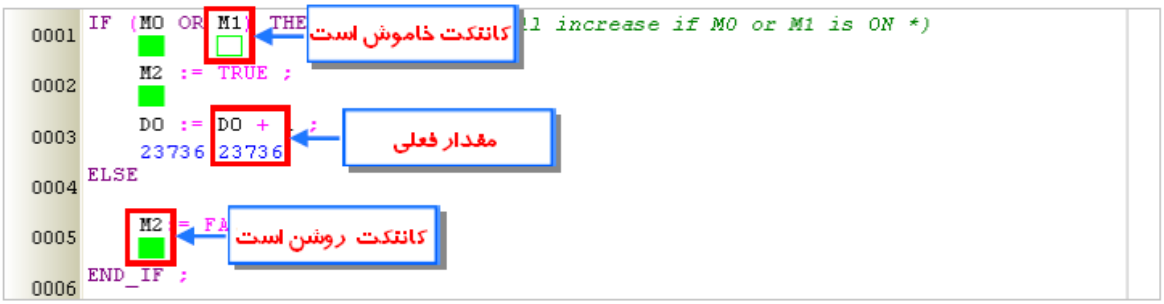
شکل ۱۱-۹ مانیتورینگ آنلاین برنامه به زبان Function block diagram

در زیر نمایی از مانیتورینگ آنلاین برنامه به زبان Instruction List را می‌بینید.



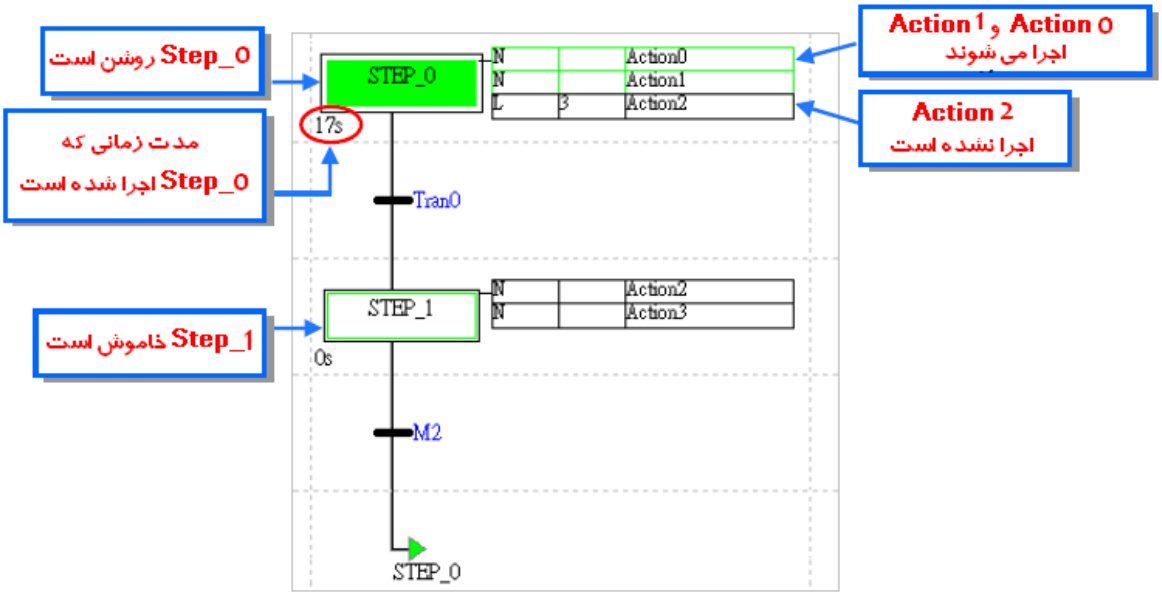
شکل ۱۱-۱۰ مانیتورینگ آنلاین برنامه به زبان Instruction List

در زیر نمایی از مانیتورینگ آنلاین برنامه به زبان Structured Text را می‌بینید.



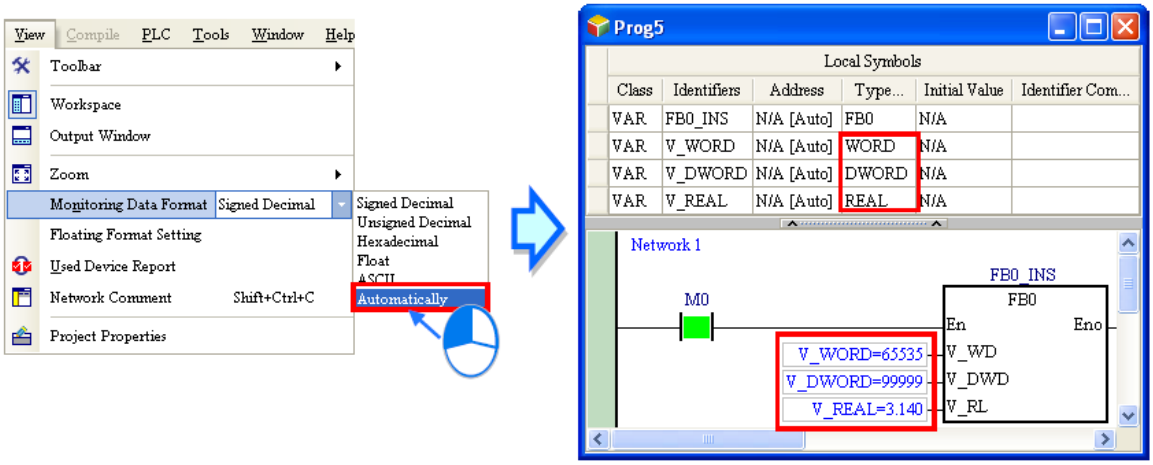
شکل ۱۱-۱۱ مانیتورینگ آنلاین برنامه به زبان Structured Text

در زیر نمایی از مانیتورینگ آنلاین برنامه به زبان Sequential Function Chart را می‌بینید. نه تنها خود آن، بلکه نحوه عملکرد پله‌ها و گذارها نیز مانیتور می‌شوند.



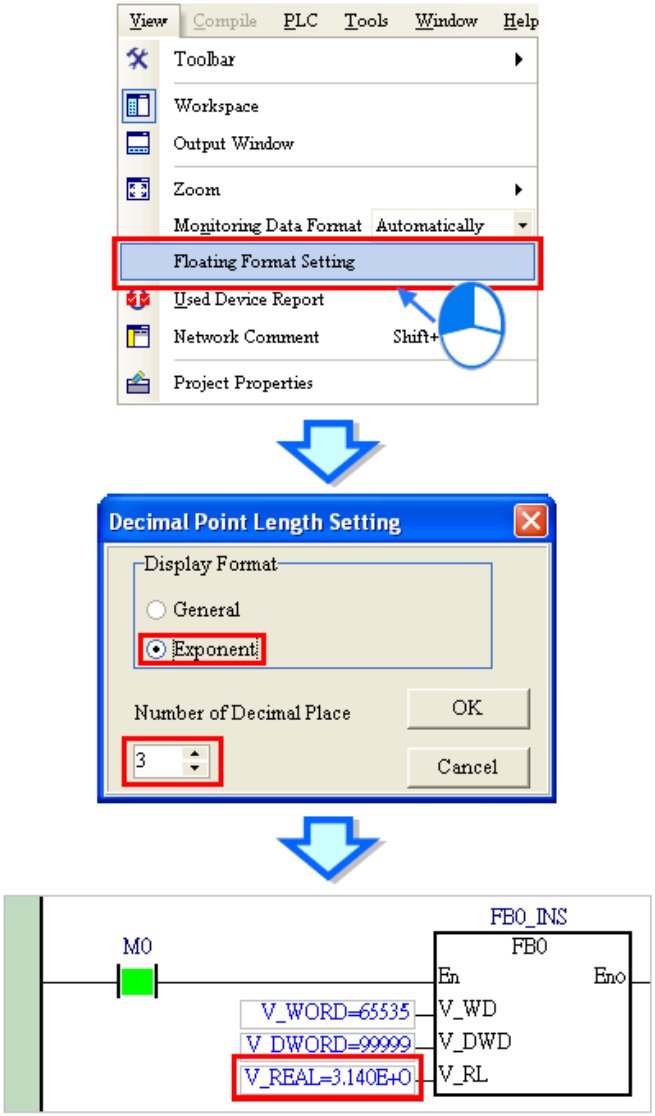
شکل ۱۱-۱۲ مانیتورینگ آنلاین برنامه به زبان Sequential Function Chart

در حالت مانیتور برنامه، کاربر می‌تواند وضعیت حافظه‌ها، مقدار رجیسترها و پورت ها را ببیند. همچنین کاربر نحوه نمایش داده ها را نیز می‌تواند تعیین کند. برای اینکار بر روی Monitoring Data Format در سربرگ View کلیک کنید. در صورتی که در پنجره باز شده، گزینه Automatically را انتخاب کنید (فرمت نمایش به نوع داده‌ای که در قسمت سیمبول مشخص شده است بستگی خواهد داشت).



شکل ۱۱-۱۳ نمایش داده ها در وضعیت Automatic به تنظیمات سیمبول ها بستگی دارد

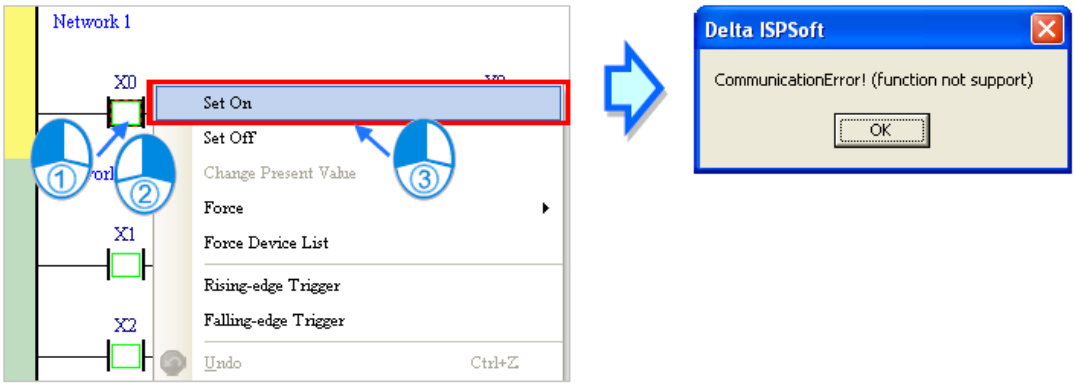
در صورتی که کاربر در سربرگ View گزینه Floating Format Setting را انتخاب کند، می‌تواند در پنجره باز شده فرمت نمایش اعداد ممیز شناور (اعشاری) را تعیین کند.



شکل ۱۱-۱۴ تنظیم نحوه نمایش اعداد ممیز شناور (اعشاری) در مانیتورینگ آنلاین

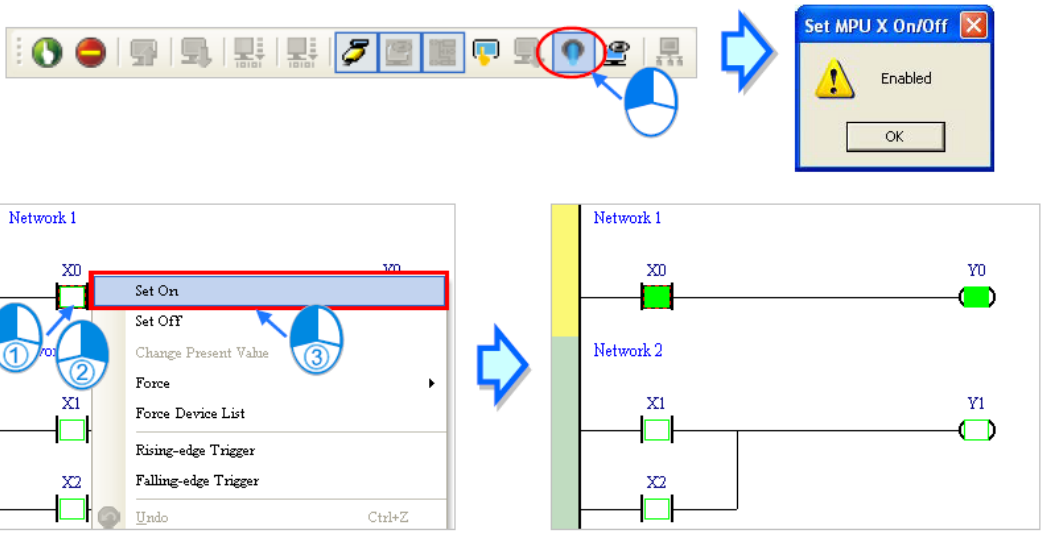
۱۱-۱-۱- تغییر وضعیت ورودی ها

در حالت آنلاین، کاربران می‌توانند وضعیت کانتکت ها را در صفحه ویرایش برنامه تغییر دهند. البته در بعضی مدل‌ها ممکن است به خاطر نوع معماری داخلی PLCها کاربران امکان تغییر وضعیت کانتکت ها را نداشته باشند، در این وضعیت کاربر حین تغییر وضعیت ورودی ها با پیغام خطا مواجه می‌شود.



شکل ۱۱-۱۵ پیغام خطا در صورتی که PLC از تغییر آنلاین کانتکت پشتیبانی نکند

درصورتی که بخواهیم کانتکت ها (رجیسترهای X) را حین مانیتورینگ آنلاین تغییر دهیم، بر روی [1] کلیک کرده تا امکان تغییر آن‌ها برای ما فراهم آید. همچنین مقدار ورودی‌های X پس از فعال شدن این گزینه دیگر اسکن نشده و به روز نمی‌شود.



شکل ۱۱-۱۶ تغییر آنلاین ورودی ها

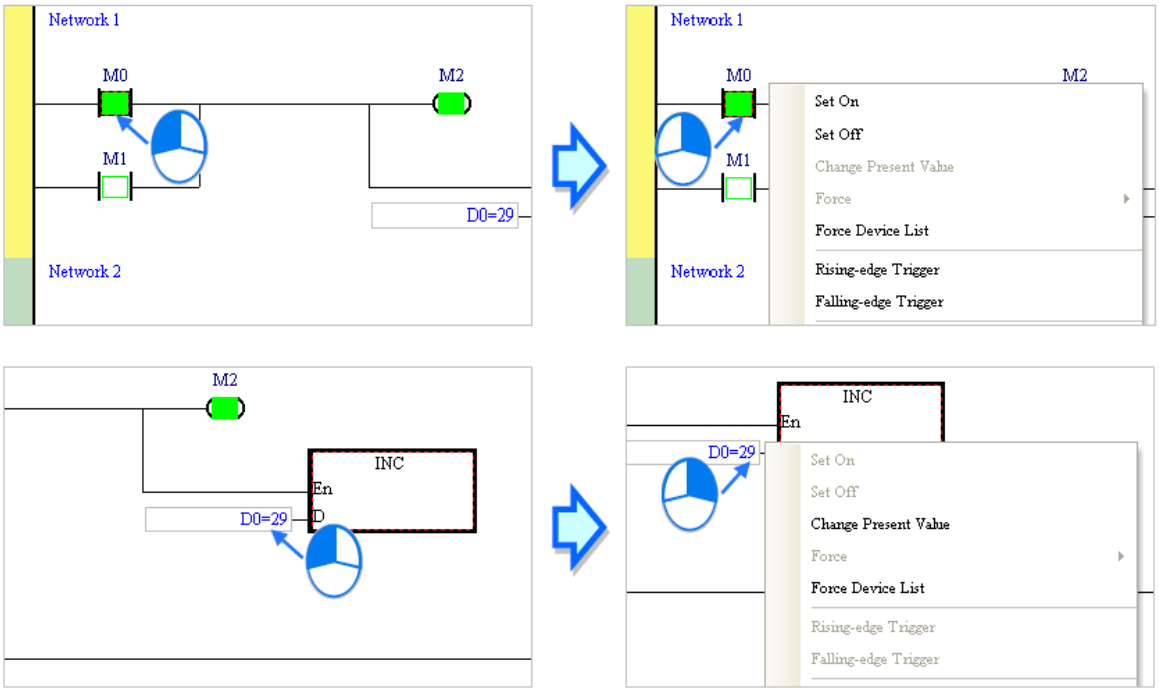
پس از غیرفعال شدن مانیتورینگ آنلاین، Set MPU X On/Off غیرفعال نمی‌شود، پس لازم است حتما قبل از خارج شدن از حالت مانیتورینگ آنلاین این تابع را غیرفعال کنیم. در غیر اینصورت ورودی PLC خوانده نمی‌شود و به روز نخواهد شد.

[1] Set MPU X On/Off

شکل ۱۱-۱۷ لزوم غیر فعال کردن MPU قبل از خروج از حالت آنلاین

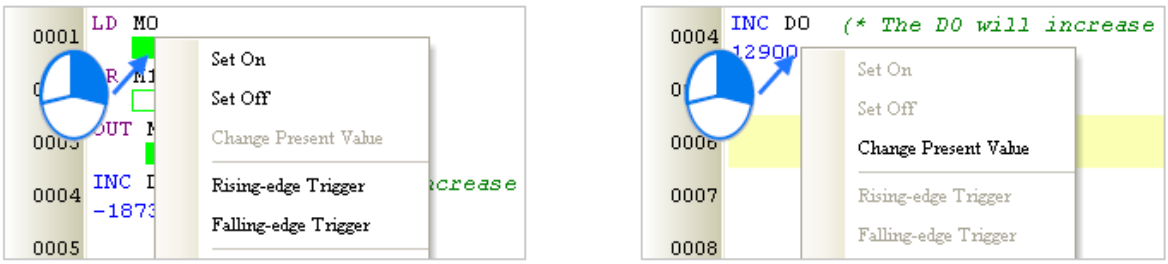
۱۱-۱-۲- مانیتورینگ آنلاین برنامه

در حالت مانیتورینگ برنامه، کاربر می‌تواند وضعیت کانتکت‌های ورودی و حافظه‌ها را تغییر دهد. با استفاده از این قابلیت کاربر قادر به عیب یابی و تست برنامه خود خواهد بود. در زبان برنامه نویسی Ladder و FBD کاربر می‌تواند ابتدا با کلیک، بلوک مورد نظر خود را انتخاب و سپس با کلیک راست بر روی آن مقدار وضعیت دلخواه خود را تعیین کند.



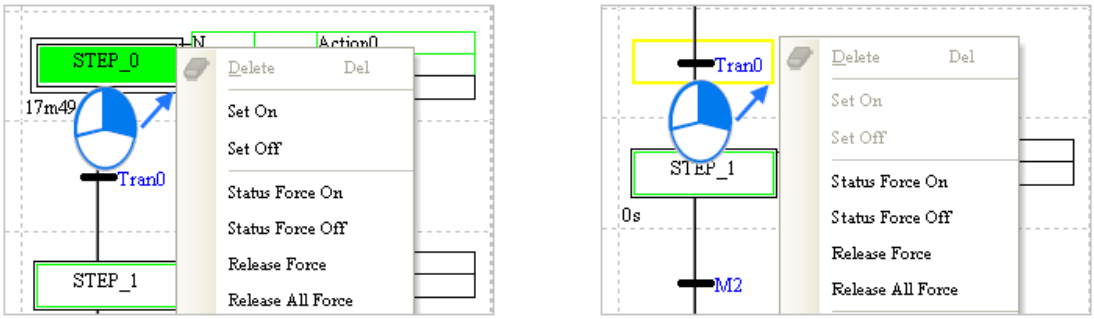
شکل ۱۱-۱۸ تغییر وضعیت کانتکت ها در Ladder و FBD

در صورتی که کاربر بخواهد تغییری در Instruction List و یا Structured text ایجاد کند، می‌تواند قسمت مورد نظر خود را با کلیک انتخاب و سپس با کلیک راست بر روی آن تغییر مورد نظر خود را انتخاب کند.



شکل ۱۱-۱۹ تغییر وضعیت کانتکت ها در Instruction List و Structured text

اگر کاربر بخواهد تغییری در Step یا Transition زبان برنامه نویسی Sequential function chart ایجاد کند، می‌تواند بر روی Step و یا Transition مورد نظر کلیک راست کرده و تغییر مورد نظر خود را انتخاب کند.



شکل ۱۱-۲۰ تغییر وضعیت کانتکت ها در SFC

گزینه‌هایی که برای تغییر در زبان‌های برنامه نویسی مختلف می‌توانیم از آن‌ها استفاده کنیم به صورت زیر است.

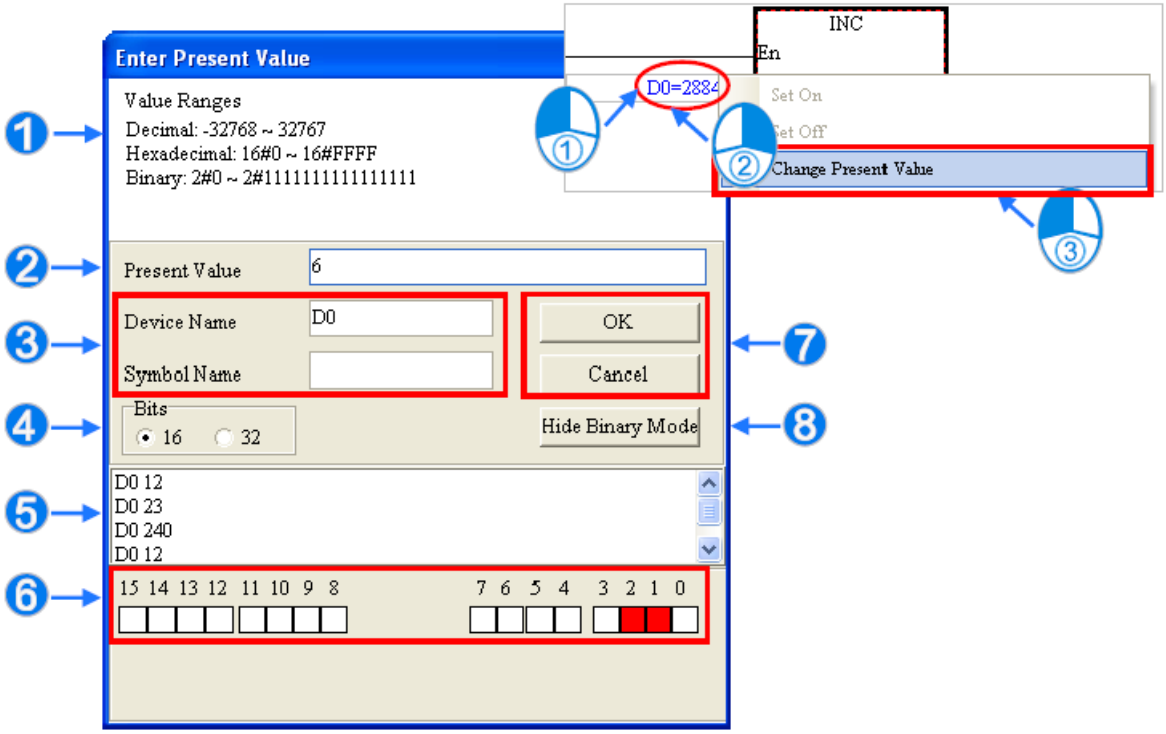
جدول ۱۱-۲: گزینه های تغییر وضعیت کانتکت ها در حالت آنلاین

گزینه	÷شرح
Set On	روشن کردن کانتکت
Set Off	خاموش کردن کانتکت
Rising-Edge Trigger	بدون توجه به وضعیت فعلی، سیستم کانتکت را خاموش و سپس روشن می‌کند (ایجاد لبه بالا رونده).

Falling-edge Trigger	بدون توجه به وضعیت فعلی، سیستم کانتکت را روشن و سپس خاموش می‌کند (ایجاد لبه پایین رونده).
Change Present Value	می‌توان مقدار تعیین کرد (در حافظه‌ها یا ...)
Force	حالت اجبار برای کانتکت ورودی و یا خروجی
Force Device List	حالت اجبار چند کانتکت ورودی و خروجی در جدول ورودی، خروجی ها

۱۱-۱-۲-۱- **Change Present Value**

برای اینکه مقداری را با گزینه Change Present Value تغییر دهیم، در پنجره باز شده با عنوان Enter Present Value مقدار مورد نظر را وارد کنید.



شکل ۱۱-۲۱ تغییر مقدار حافظه، ورودی و یا غیره با استفاده از گزینه Change Present Value

1. توضیحات
2. مقدار مورد نظر را می‌توان در این قسمت وارد کرد
3. نام حافظه یا سیمبول

4 نوع حافظه

5 تاریخچه تغییرات (ابتدا نام حافظه و سپس مقدار آن آمده است)

6 در فرمت باینری کاربران می‌توانند مقدار مطلوب خود را با کلیک موس انتخاب کنند.

7 برای ذخیره تغییرات می‌توان بر روی OK کلیک کرد.

8 نمایش و یا عدم نمایش حالت باینری

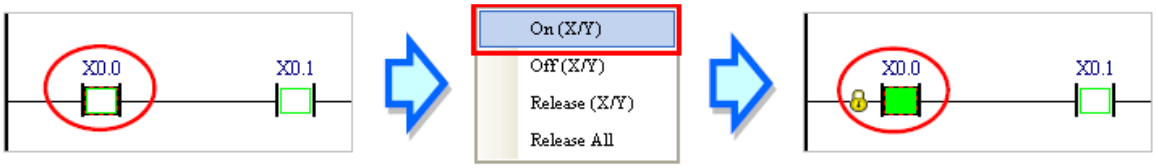
۱۱-۱-۲-۲- حالت اجبار[1]

در این حالت می‌توان ورودی و خروجی ها را بر روی مقدار مشخصی ثابت نگه داشت. در صورتی که ورودی و یا خروجی در حالت اجبار قرار گیرند، مقدار آن تحت تاثیر ورودی و یا برنامه تغییر نخواهد کرد. این ویژگی در مواقعی که ایمنی مطرح است بسیار پر کاربرد است. مثلا در مثال زیر حتی اگر حافظه مقدار صفر هم داشته باشد، خروجی Y0.0 در وضعیت اجبار روشن و ثابت خواهد ماند. در این حالت علامت قفل زرد رنگ در سمت چپِ خروجی برای مشخص کردن حالت اجبار نمایش داده خواهد شد.

شکل ۱۱-۲۲ خروجی در حالت اجبار

با انتخاب کانتکت X و یا Y با استفاده از کلیک چپ و سپس کلیک راست بر روی آن، در قسمت Force با گزینه های زیر مواجه خواهیم شد.

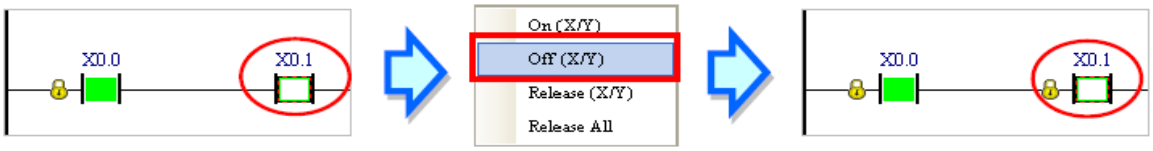
- ON (X/Y): کانتکت X و یا Y به حالتِ اجبارِ روشن در آید.



شکل ۱۱-۲۳ حالت اجبار ON

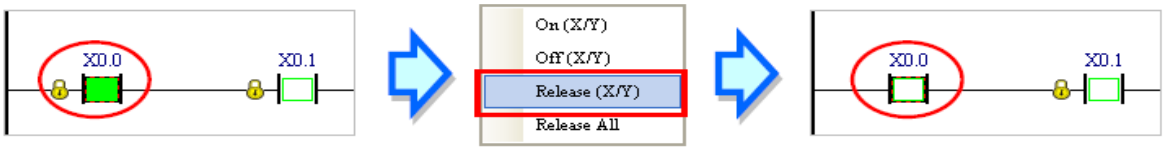
- Off (X/Y): کانتکت X و یا Y به حالتِ اجبارِ خاموش در آید.

[1] Force



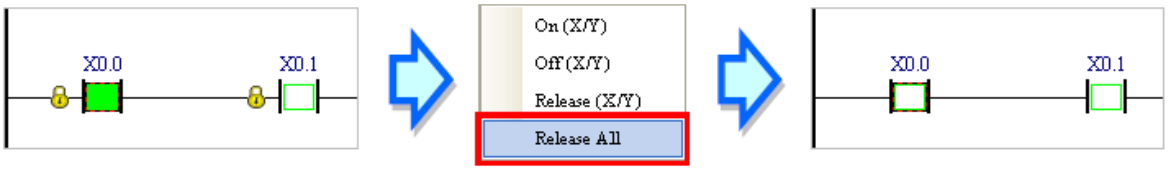
شکل ۱۱-۲۴ حالت اجبار OFF

- Release (X/Y): آزاد شدن کانتکت X و یا Y از حالتِ اجبار.



شکل ۱۱-۲۵ حذف حالت اجبار

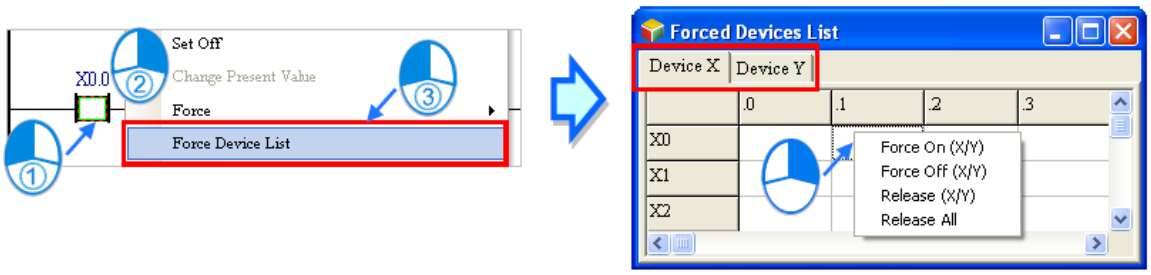
- Realease All: باز شدن تمامی کانتکت ها از حالتِ اجبار.



شکل ۱۱-۲۶ حذف تمامی حالت های اجبار

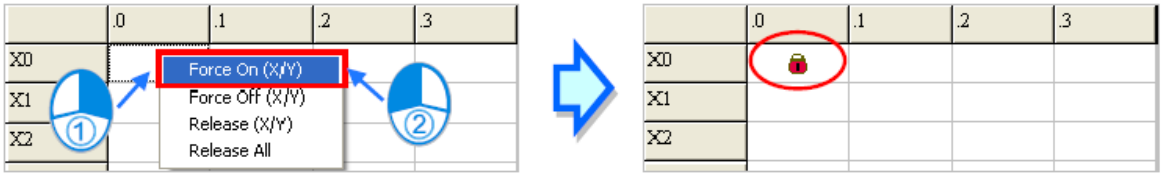
۱۱-۱-۲-۳- Force Device List

با کلیک راست بر روی صفحه ویرایش برنامه و یا یکی از کانتکت ها و انتخاب Force Device List می‌توان به لیست حالت های اجبار کانتکت های ورودی و خروجی دسترسی داشت. در این بخش با انتخاب سربرگ ورودی و یا خروجی و همچنین رفتن به سطر مورد نظر می‌توان وضعیت حالت اجبار را در کانتکت مورد نظر تعین کرد.



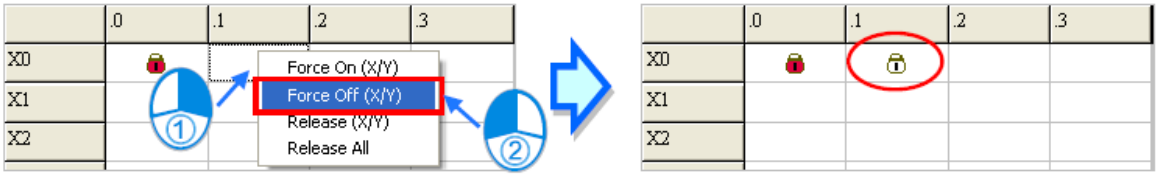
شکل ۱۱-۲۷ لیست حالت اجبار

- ON (X/Y): کانتکت X و یا Y به حالت اجبار روشن در آید.



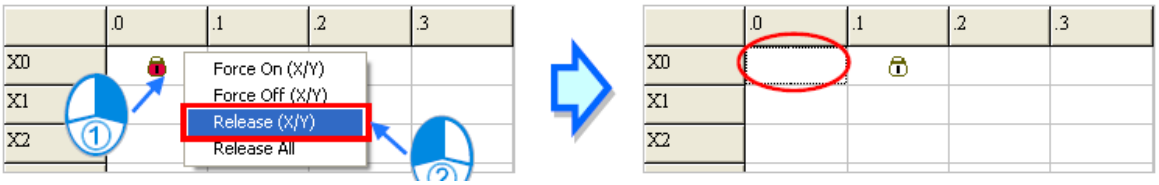
شکل ۱۱-۲۸ لیست حالت اجبار- حالت اجبار ON

- Off (X/Y): کانتکت X و یا Y به حالت اجبار خاموش در آید.



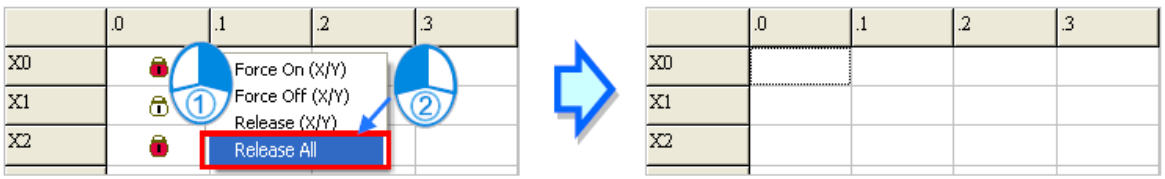
شکل ۱۱-۲۹ لیست حالت اجبار- حالت اجبار OFF

- Release (X/Y): آزاد شدن کانتکت X و یا Y از حالت اجبار.



شکل ۱۱-۳۰ لیست حالت اجبار- حذف حالت اجبار

- Realease All: خارج شدن تمامی کانتکت ها از حالت اجبار.



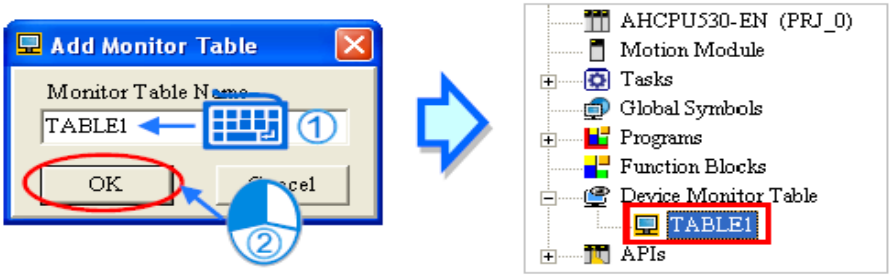
شکل ۱۱-۳۱ لیست حالت اجبار- حذف تمامی حالت های اجبار

۱۱-۱-۳- جدول مانیتورینگ

پس از آنکه کاربر جدول مانیتورینگ را تشکیل داد می‌تواند سیمبول‌ها، حافظه‌ها و ورودی و خروجی-ها را در جدول مانیتورینگ مشاهده کند. همانطور که گفته شد، کاربران می‌توانند بیش از یک جدول برای مانیتورینگ در ISPSoft ایجاد کنند. برای باز کردن جدول مانیتورینگ می‌توان بر روی آیکون کلیک کرد.

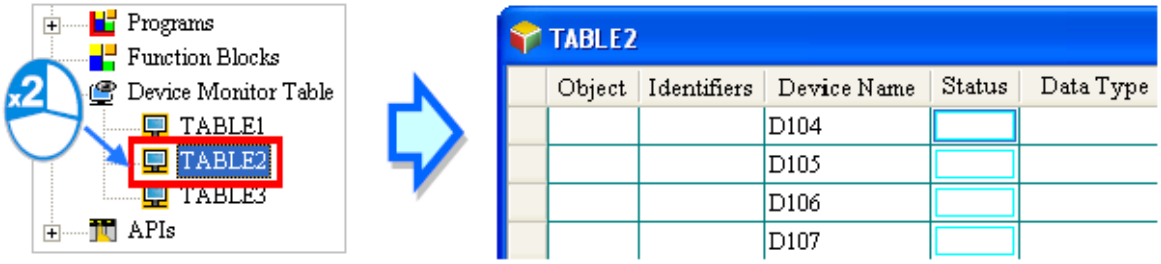
شکل ۱۱-۳۲ ایجاد جدول مانیتورینگ –قسمت اول

در صفحه باز شده، نام جدول را تایپ و بر روی OK کلیک کنید. سپس جدول مورد نظر در بخش مدیریت پروژه در زیر شاخه Device Monitor Table ایجاد خواهد شد. در این حالت پنجره جدول مانیتورینگ نیز باز خواهد شد.



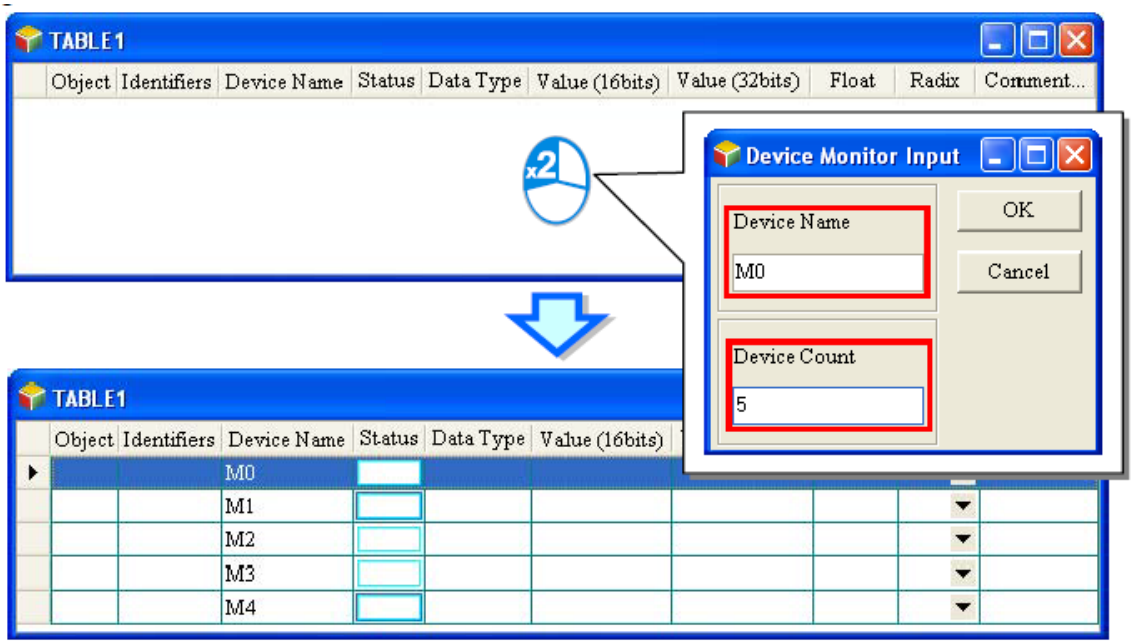
شکل ۱۱-۳۳ ایجاد جدول مانیتورینگ –قسمت دوم

برای باز کردن پنجره جدول مانیتورینگ، می‌توان بر روی نام آن در صفحه مدیریت پروژه دوبار کلیک کرد.



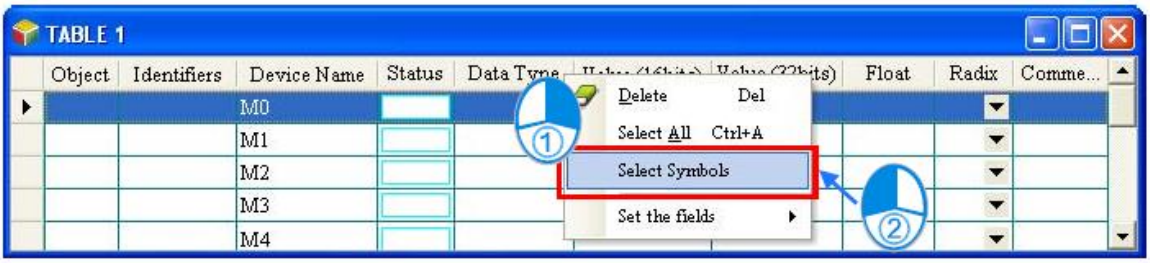
شکل ۱۱-۳۴ باز کردن پنجره جدول مانیتورینگ

اگر کاربر می‌خواهد پورت و یا حافظه و یا سیمبولی را به جدول اضافه کند می‌تواند در فضای خالی جدول دوبار کلیک و درپنجره باز شده در قسمت Device Name نوع داده برای مانیتورینگ و در قسمت Device Count تعداد داده مورد نظر خود را وارد کند (مثلا با انتخاب M5 در Device Name و وارد کردن 3 در Device Count مانیتورینگ M5 و M6 و M7 یعنی سه حافظه با شروع از M5 لحاظ خواهد شد). توجه به این نکته ضروری است که حداکثر ۱۰۰ آیتم را می‌توان در جدول مانیتور کرد.



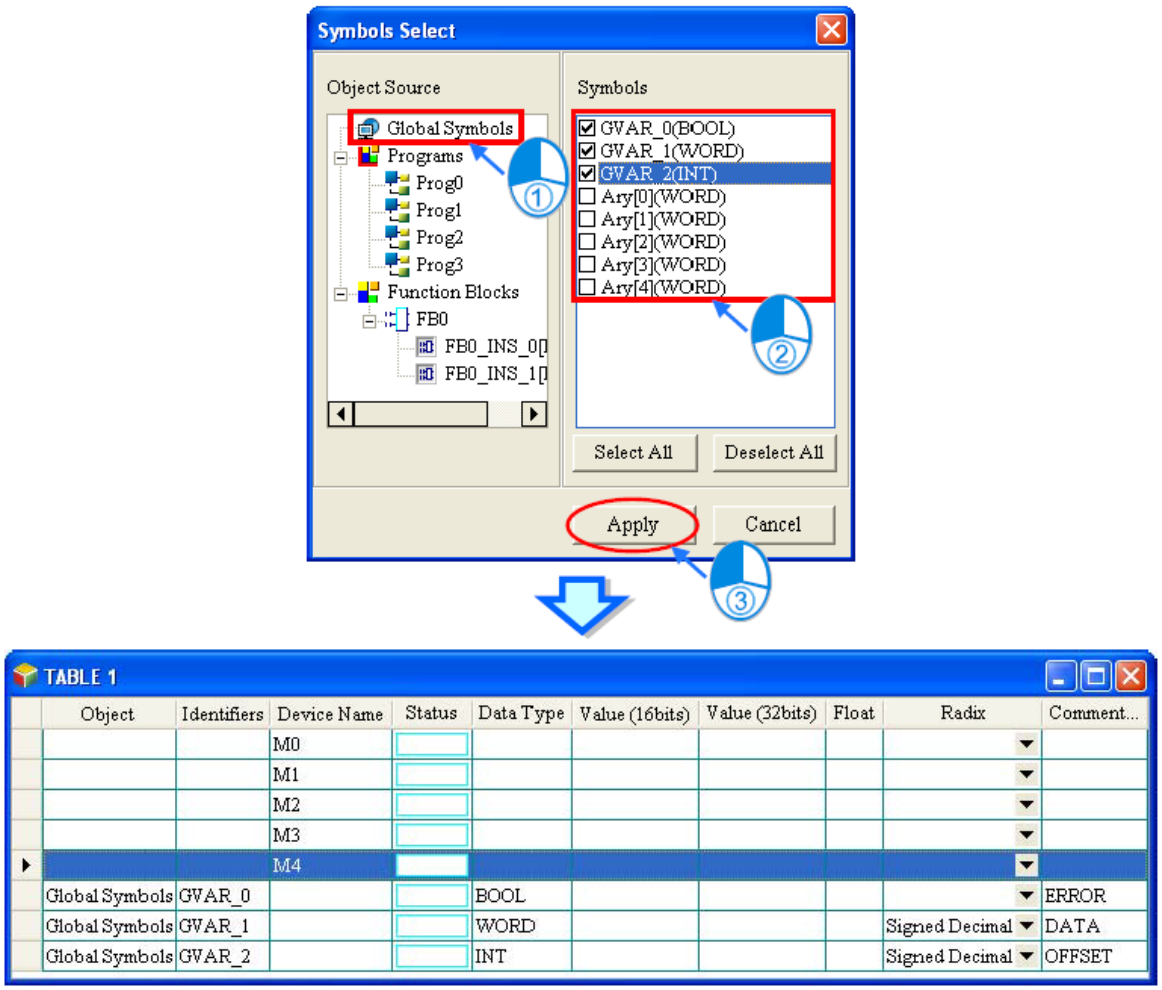
شکل ۱۱-۳۵ اضافه کردن حافظه و پورت در جدول مانیتورینگ

کاربران همچنین می‌توانند به جدول فوق، سیمبول نیز اضافه کنند، برای اینکار در پنجره جدول کلیک راست کرده و گزینه Select Symbol را انتخاب نمایید.



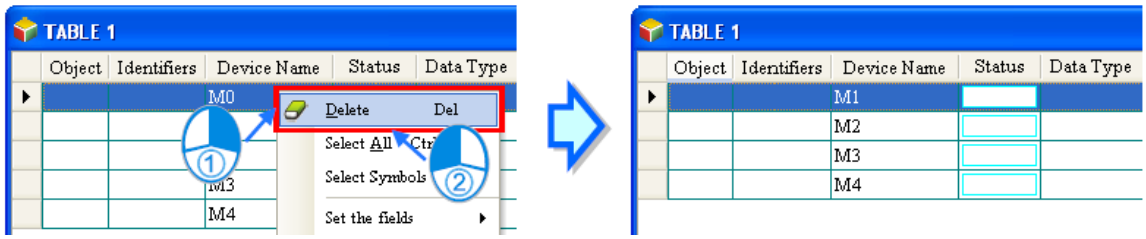
شکل ۱۱-۳۶ اضافه کردن سیمبول به جدول مانیتورینگ –قسمت اول

در پنجره Symbols Select می‌توان نوع سیمبول سراسری و یا محلی (در Function Blocks) را انتخاب کرد. قبل از انتخاب سیمبول های محلی، برنامه باید کامپایل شده باشد. پس از انتخاب POU در قسمت Object Source می‌توان سیمبول‌های متناظر با آن را در قسمت Symbols انتخاب کرد. پس از کلیک بر روی Apply سیمبول‌های تعیین شده در جدول مانیتورینگ اضافه خواهد شد.



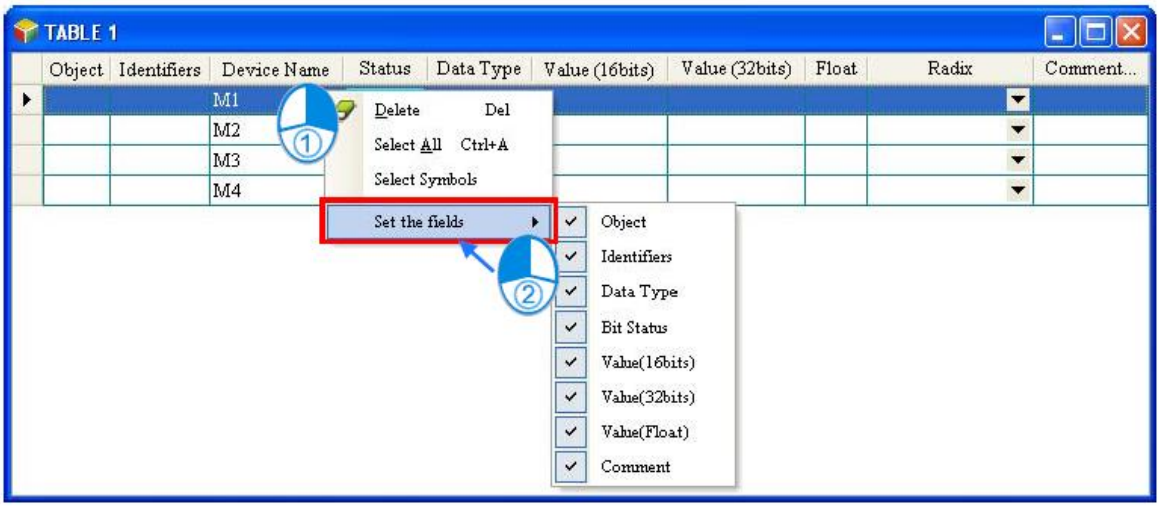
شکل ۱۱-۳۷ اضافه کردن سیمبول به جدول مانیتورینگ –قسمت دوم

برای پاک کردن پورت، حافظه و یا سیمبول‌ها در جدول می‌توان آن را انتخاب و سپس کلیک راست و Delete کرد.



شکل ۱۱-۳۸ پاک کردن المان ها در جدول سیمبول ها

برای مدیریت ستون‌ها برای مانیتورینگ بهینه می‌توان در جدول کلیک راست کرده و در قسمت Set the Fields ستون‌های دلخواه را انتخاب کرد.



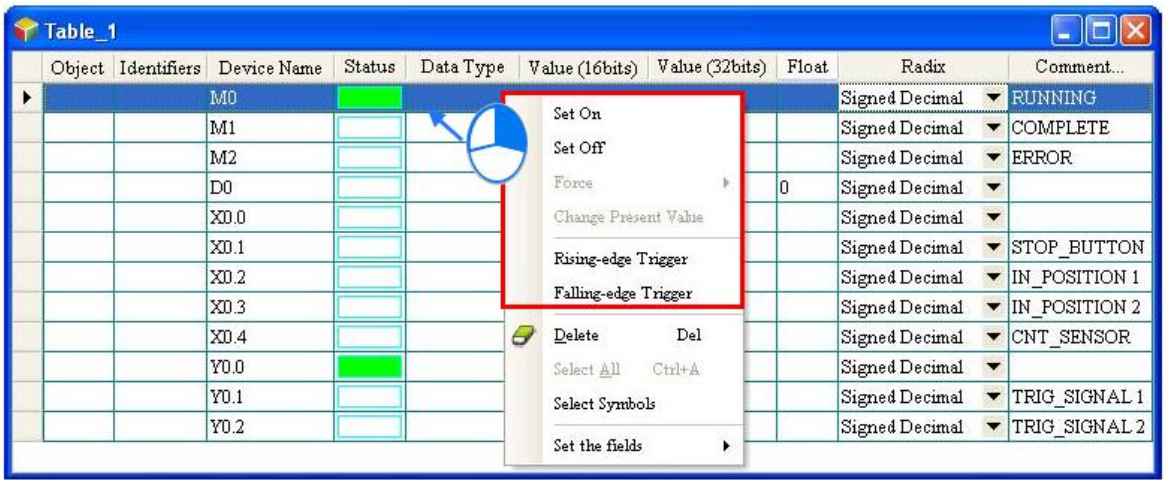
شکل ۱۱-۳۹ انتخاب ستون ها در جدول مانیتورینگ

شرح ستون‌های صفحه مانیتورینگ به صورت زیر است:

جدول ۱۱-۳: ستون های جدول مانیتورینگ

ستون	شرح
Object	منبع سیمبول: مثلا اینکه سیمبول سراسری و یا متعلق به FB مشخصی است
Identifier	نماد سیمبول
Device Name	نام حافظه و یا ورودی و یا خروجی
Status	وضعیت پورت و یا حافظه و یا سیمبول بیتی نمایش داده می‌شود
Data Type	نوع داده سیمبول
Value (16 bits)	وضعیت پورت و یا حافظه و یا سیمبول ۱۶ بیتی نمایش داده می‌شود.
Value (32 bits)	وضعیت پورت و یا حافظه و یا سیمبول ۳۲ بیتی نمایش داده می‌شود.
Float	وضعیت پورت و یا حافظه و یا سیمبول ۳۲ بیتی ممیز شناور نمایش داده می‌شود.
Radix	می‌توان در این ستون نوع فرمت نمایش داده را تعیین کرد.
Comment	توضیحات

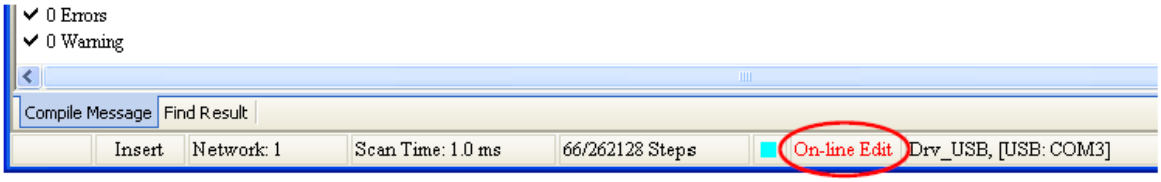
پس از ساخته شدن جدول مانیتورینگ، کاربر می‌تواند با کلیک بر روی  اجزای آن را به صورت آنلاین مانیتور کند. همچنین پس از کلیک راست بر روی هر کدام از سطرها می‌توان وضعیت فعلی آن را به صورت دلخواه تغییر داد.



شکل ۱۱-۴۰ مانیتورینگ و تغییر وضعیت آنلاین در جدول مانیتورینگ

۱۱-۱-۴- ویرایش آنلاین برنامه

زمانی که سیستم در حالت آنلاین است و PLC در حالت اجرا است، کاربر می‌تواند برنامه‌ای که مانیتور می‌کند را به صورت آنلاین ویرایش نماید. برای اینکار پس از اطمینان از در حال اجرا بودن PLC بر روی  کلیک کرده و پس از پایان ویرایش نیز برنامه را کامپایل و در PLC دانلود کند. در حالت ویرایش عبارت On-Line Edit در نوار وضعیت ظاهر خواهد شد.



شکل ۱۱-۴۱ حالت در حال ویرایش در نوار وضعیت

برای ویرایش آنلاین در زبان‌های مختلف برنامه نویسی محدودیت هایی وجود دارد که در جدول زیر لیست شده است.

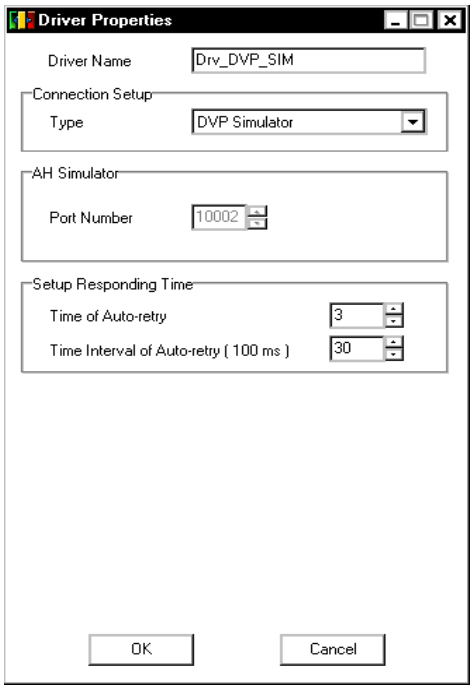
جدول ۱۱-۴: محدودیت های زبان های مختلف برنامه نویسی برای ویرایش برنامه

زبان برنامه نویسی	شرح محدودیت

Ladder Diagram	در هر زمان یک خط network را در آن واحد می‌توان ویرایش کرد. پس از اضافه شدن network جدید هم تنها آن را می‌توان ویرایش کرد.
Function Block Diagram	در هر زمان یک خط network را در آن واحد می‌توان ویرایش کرد. پس از اضافه شدن network جدید هم تنها آن را می‌توان ویرایش کرد.
Instruction List	محدودیتی در تعداد خطوط برای اصلاح نیست.
Structured Text	محدودیتی در تعداد خطوط برای اصلاح نیست.
Sequential function Chart	نمی‌توان ساختار چارت SFC را به صورت آنلاین ویرایش کرد، اما Transitions و Actions آن را می‌توان اصلاح کرد. محدودیت موجود برای ویرایش Transitions و Actions نیز به زبان برنامه نویسی آن وابسته است.
محدودیت های مشترک	تنها می‌توان یک POU را ویرایش کرد. امکان اضافه و یا تغییر سیمبول در حالت آنلاین وجود ندارد.

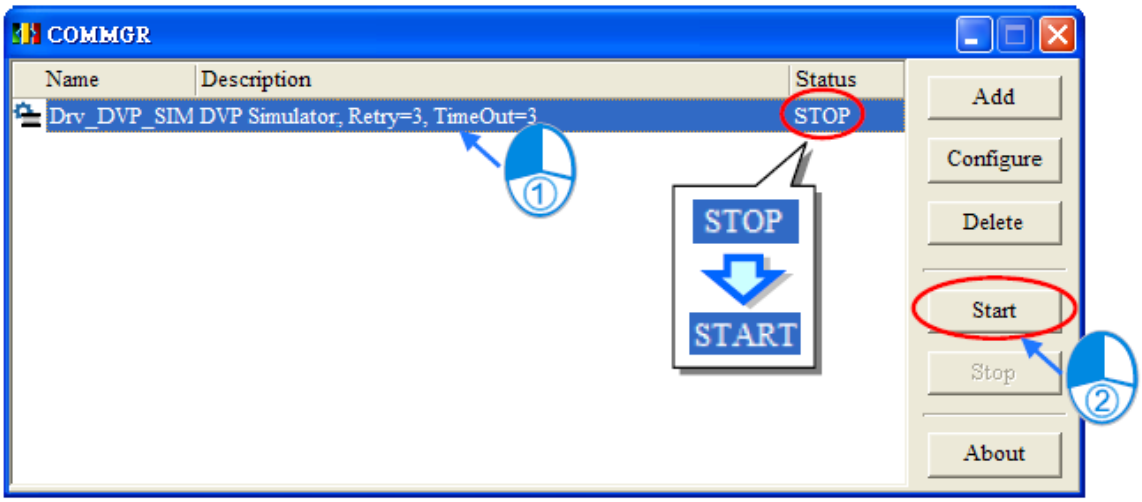
۱۱-۲- شبیه سازهای DVP

برای اینکه بتوانیم از شبیه ساز PLC استفاده کنیم، باید ارتباط مجازی ISPSoft با آن را از طریق درایور مجازی که COMMGR می سازد برقرار کنیم. برای اینکار کافی است در COMMGR نوع ارتباط را DVP Simulator تعیین کنیم.



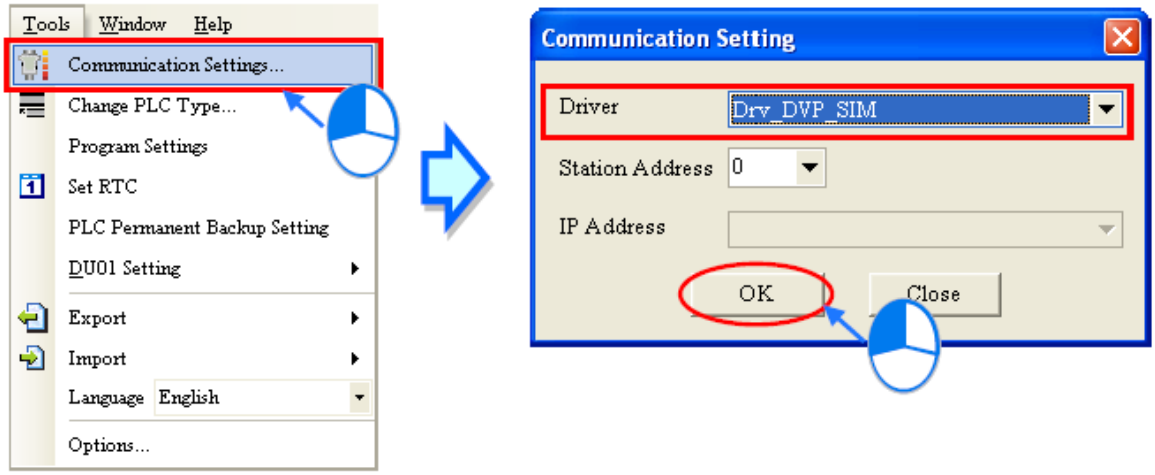
شکل ۱۱-۴۲ ساخت درایور نوع DVP Simulator برای شبیه ساز

سپس در صفحه COMMGR آن را Start کنیم.



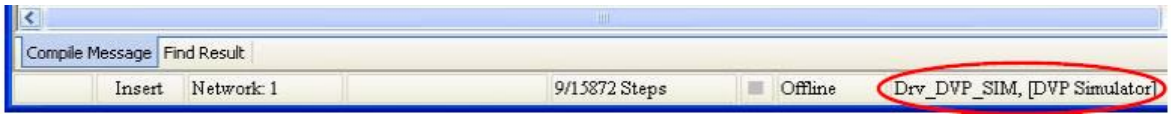
شکل ۱۱-۴۳ فعال کردن درایور نوع DVP Simulator

در ادامه در ISPSoft نیز در سربرگ Tools و قسمت Communication Setting نوع درایور را DVP Simulator انتخاب کرده و بر روی OK کلیک می کنیم.



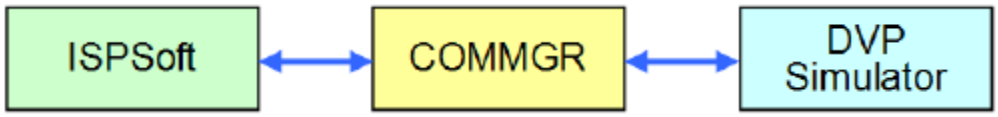
شکل ۱۱-۴۴ تنظیم درایور ISPSoft به صورت DVP Simulator

پس از تنظیمات ارتباطی نوع ارتباط در نوار وضعیت به صورت DVP Simulator مشخص خواهد شد.



شکل ۱۱-۴۵ اتصال به درایور نوع DVP Simulator در نوار وضعیت

COMMGR انتقال داده بین ISPSoft و شبیه ساز DVP را بر عهده خواهد داشت. در حین برقراری این ارتباط برنامه‌های کاربر بر روی شبیه ساز DVP بارگزاری و از روی آن استخراج خواهد شد، کاربر می‌تواند مانیتورینگ آنلاین انجام دهد، ورودی و خروجی ها را در حالت اجبار قرار دهد و پورت ها و حافظه ها را در شبیه ساز ویرایش کند.

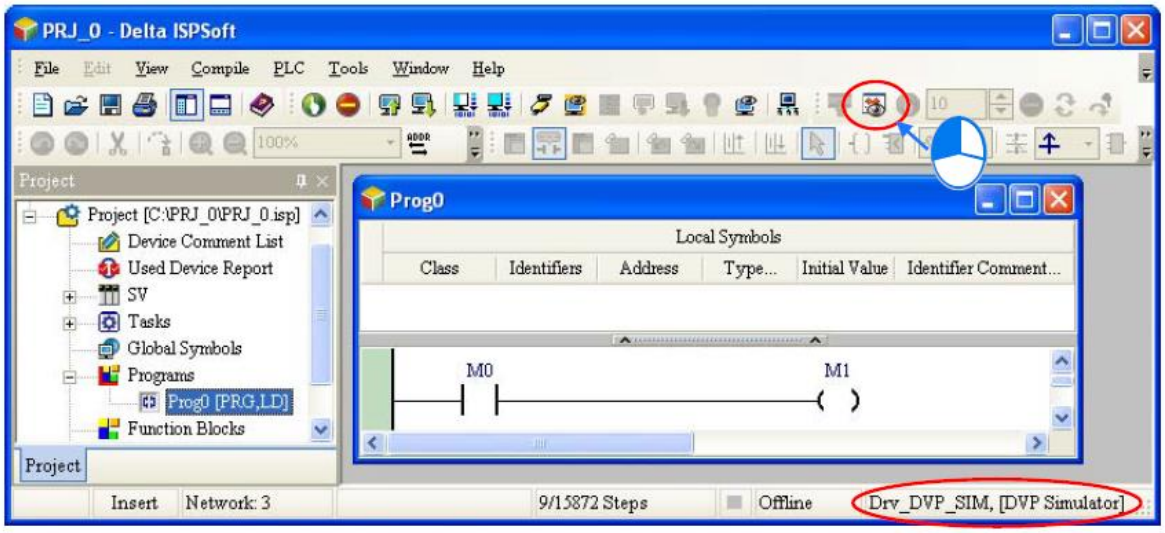


شکل ۱۱-۴۶ بلوک دیاگرام ارتباط ISPSoft با شبیه ساز DVP از طریق COMMGR

۱۱-۳- حالت عیب یابی برای PLCهای سری DVP

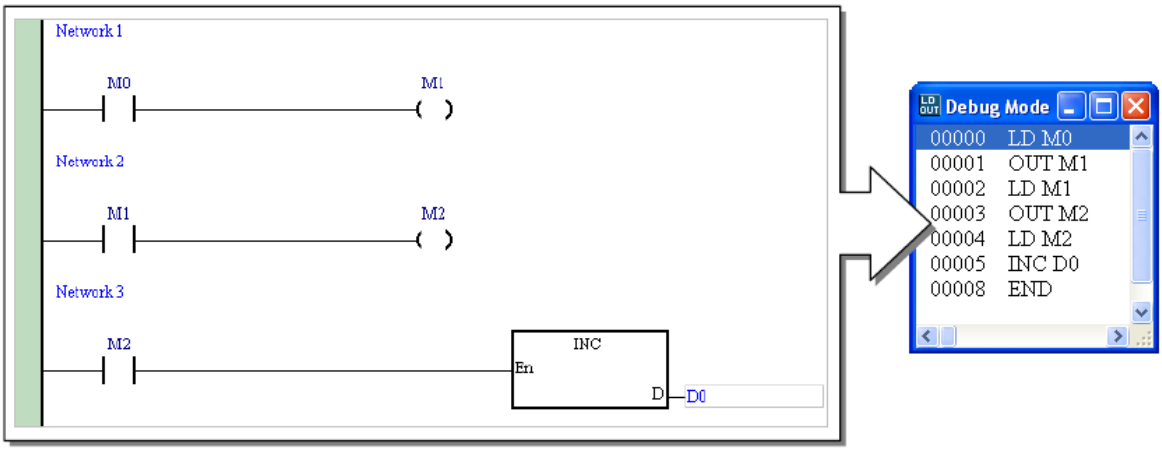
حالت عیب یابی برای PLCهای سری DVP با استفاده از ابزار عیب یابی انجام می‌شود. باید توجه شود که این حالت تنها در زمان ارتباط نرم افزار با شبیه ساز قابل استفاده است (زمانی که حالت آنلاین ارتباط با PLC فعال باشد نمی‌توان حالت عیب یابی را فعال کرد، با این وجود پس از فعال شدن حالت عیب یابی، می‌توان برای مانیتور کردن وضعیت شبیه ساز، می‌توان به حالت آنلاین رفت).

برای اجرای ابزار عیب یابی ابتدا نوع درایور را DVP Simulator انتخاب و سپس بر روی  کلیک کنید. در این حالت پنجره عیب یابی باز خواهد شد (برای غیر فعال شدن این حالت نیز می‌توان دوباره بر روی  کلیک کرد).



شکل ۱۱-۴۷ فعال کردن حالت عیب یابی

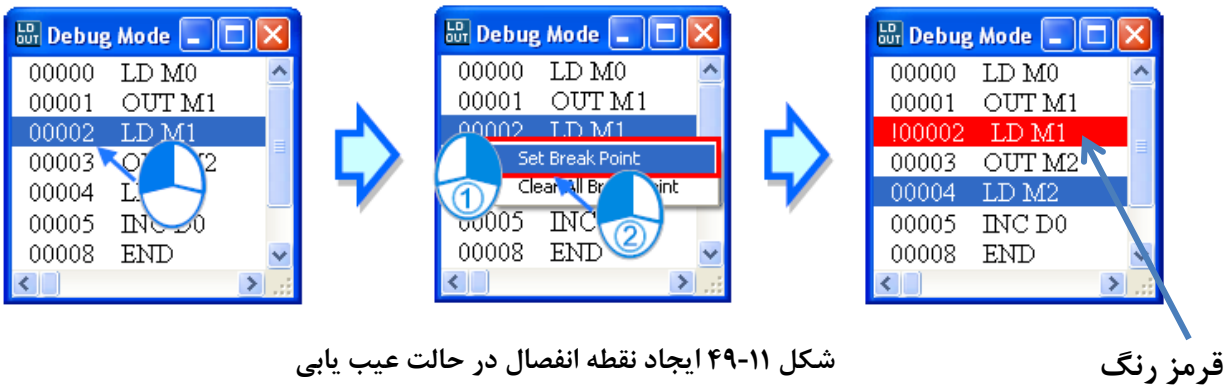
پس از تبدیل برنامه به Instruction List در پنجره حالت عیب یابی، قسمتی از برنامه که دارای پس زمینه (احتمالا آبی) است موقعیت کدی که در آنجا اجرای برنامه متوقف شده است را نشان می‌دهد (یعنی خود آن خط نیز اجرا نشده است).



شکل ۱۱-۴۸ Instruction List برای حالت عیب یابی

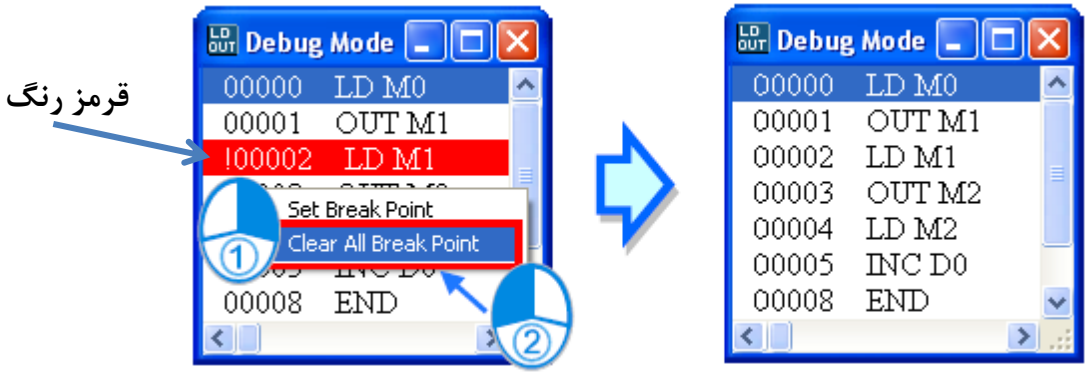
۱۱-۳-۱- اضافه و حذف نقطه انفصال

برای ارزیابی دقیق و عیب یابی برنامه گاهی لازم است کُدهای Instruction List را مرحله به مرحله اعمال و اجرا را حین رسیدن به نقاط خاصی متوقف کنیم. برای اینکار می‌توانیم از Breakpoint یا نقطه انفصال استفاده کنیم. بر روی خط مورد نظر کلیک راست کرده و Set Break Point را انتخاب کنید، در این حالت پس زمینه آن خط به رنگ قرمز در خواهد آمد.



شکل ۱۱-۴۹ ایجاد نقطه انفصال در حالت عیب یابی

برای حذف نقطه انفصال نیز می‌توان بر روی خط قرمز شده کلیک راست کرده و دوباره Set Break Point را انتخاب کنیم. در صورتی که بخواهیم تمام نقاط انفصال پاک شوند می‌توانیم Clear All Break Point را انتخاب کنیم.

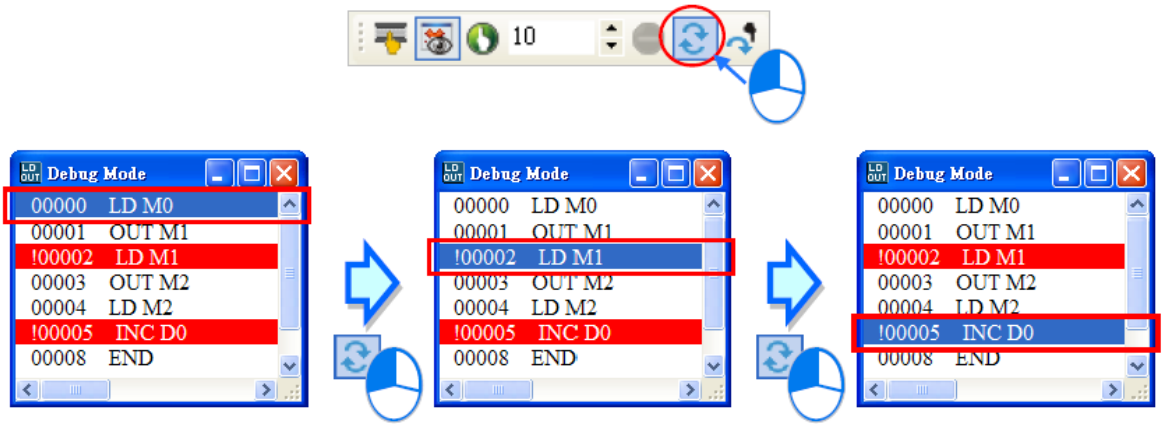


شکل ۱۱-۵۰ حذف نقطه انفصال در حالت عیب یابی

۱۱-۳-۲- اجرای برنامه در حالت عیب یابی

- اجرای پیوسته

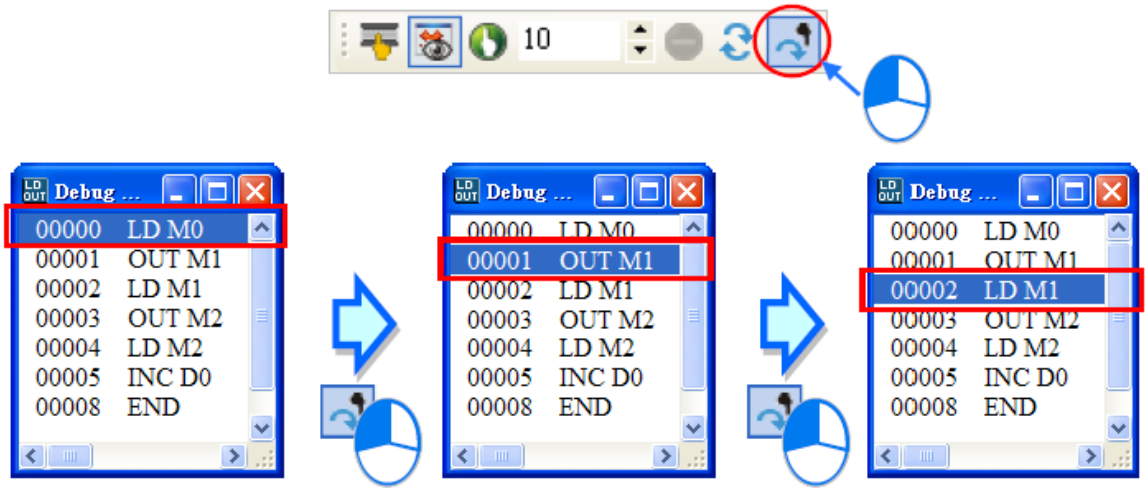
پس از کلیک بر روی اجرای برنامه در نقطه انفصال متوقف خواهد شد. اگر یکبار دیگر کلیک شود برنامه تا نقطه انفصال بعدی اجرا خواهد شد. در صورتی که برنامه نقطه انفصالی نداشته باشد، برنامه به صورت پیوسته اجرا خواهد شد.



شکل ۱۱-۵۱ اجرا از نقطه انفصال تا نقطه انفصال بعدی در حالت عیب یابی

- **اجرای یک خطی**

پس از توقف اجرای برنامه در صورتی که بر روی کلیک شود، یک خط از Instruction Set اجرا خواهد شد.



شکل ۱۱-۵۲ اجرای تک خطی در حالت عیب یابی

- **اجرای برنامه به دفعات مشخص**

برای اینکه برنامه ای به تعداد دفعات مشخصی اجرا شود می‌توانیم از 100 استفاده و تعداد دفعات مطلوب برای اجرا را وارد کرده و بر روی کلیک کنیم. در این حالت نقاط انفصال تاثیری بر اجرای برنامه نخواهد داشت. حداکثر تعداد قابل قبول برای اجرای متوالی برنامه ۳۲۷۶۷ است.

شکل ۱۱-۵۳ اجرای برنامه به دفعات مشخص در حالت عیب یابی

- **توقف اجرای برنامه**

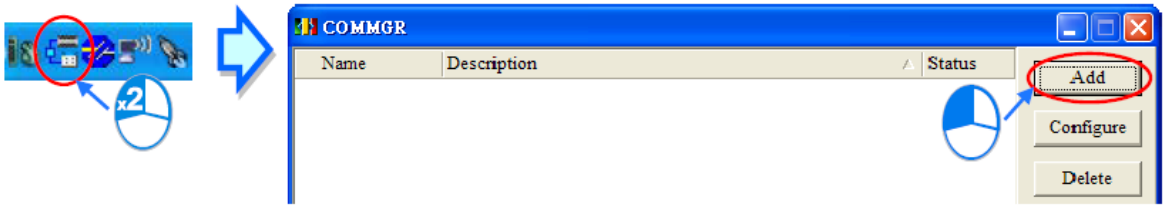
پس از کلیک بر روی اجرای برنامه در حالت عیب یابی متوقف می‌شود.

شکل ۱۱-۵۴ توقف اجرای برنامه در حالت عیب یابی

پس از کلیک بر روی ترتیب اجرای برنامه و مقادیر ریست خواهند شد. زمان بعدی، اجرای برنامه از خط اول شروع به اجرا خواهد کرد.

۱۱-۴- شبیه ساز خانواده AH500

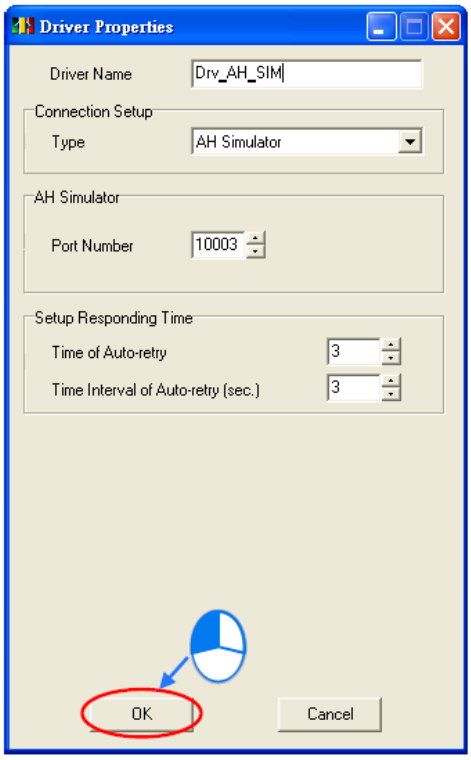
ابتدا باید ارتباط مجازی ISPSoft با یکی از PLCهای خانواده AH500 را از طریق درایور مجازی که COMMGR می سازد برقرار کنیم. برای اینکار لازم است در COMMGR نوع ارتباط (درایور) را AH Simulator تعیین کنیم. برای ساخت درایور جدید برای شبیه ساز AH500 در COMMGR بر روی Add کلیک می‌کنیم.



شکل ۱۱-۵۵ ساخت درایور جدید

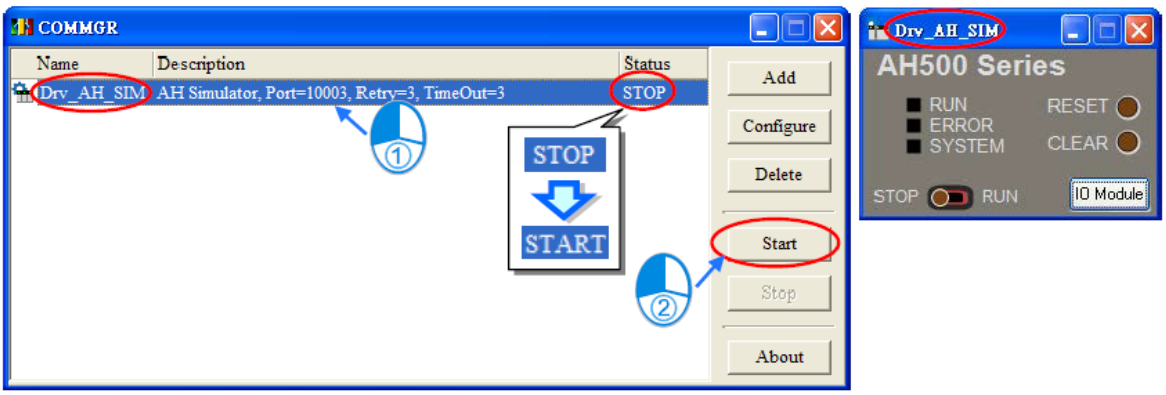
در پنجره باز شده در قسمت Driver Name باید نام درایور نوشته و در Type باید AH Simulator انتخاب شود. در قسمت Port Number نیز باید شماره پورت ارتباطی نوشته شود. Port Number را در

صورتی که برای شما مهم نیست به صورت پیش فرض رها کنید. اگر بیش از یک سیمولاتور برای شبیه سازی نیاز دارید، آن‌ها نمی‌توانند port Number های یکسان داشته باشند. (در شبیه ساز DVP تنها می-توانستیم یک شبیه ساز تعریف کنیم).



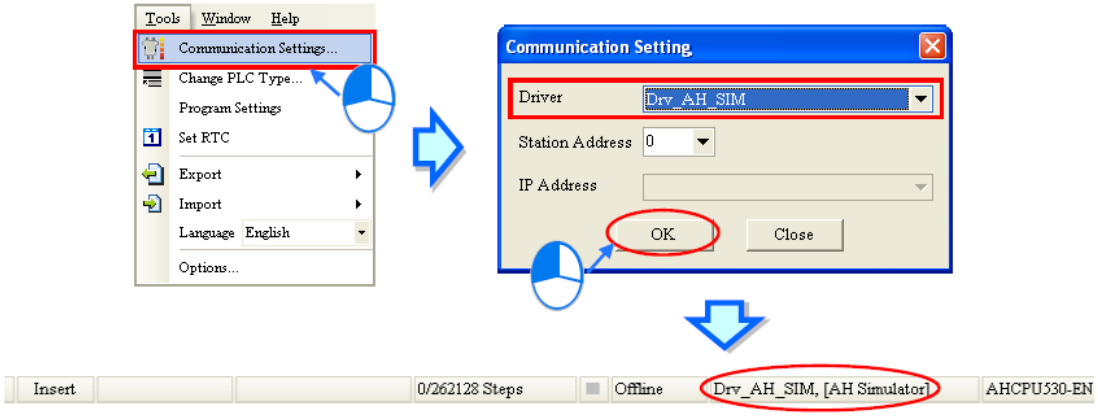
شکل ۱۱-۵۶ تنظیم درایور AH Simulator

پس از ساختن درایور شبیه ساز AH500 در COMMGR می‌توان با کلیک بر روی Start شبیه ساز را فعال کرد تا صفحه شبیه ساز فعال شود. نام این پنجره مشابه نام درایور شبیه ساز AH500 خواهد بود. با آنکه تعداد نامحدودی شبیه ساز AH500 را می‌توان ساخت ولی حداکثر از ۸ شبیه ساز به صورت هم زمان می‌توان استفاده کرد (برای بررسی عملکرد PLCهای موجود در یک شبکه، لازم است شبیه سازهای آن‌ها را به صورت همزمان اجرا کنیم).



شکل ۱۱-۵۷ فعال شدن پنجره شبیه ساز AH500 پس از فعال شدن درایور آن در COMMGR

حال برای ارتباط با نرم افزار ISPSoft می‌توان در سربرگ Tools گزینه Communication Setting را انتخاب و درایور ساخته شده برای شبیه ساز AH500 را انتخاب کرده و تایید کرد. در این مرحله ارتباط با شبیه ساز در نوار وضعیت به نمایش در خواهد آمد.



شکل ۱۱-۵۸ برقراری ارتباط شبیه ساز AH500 با ISPSoft

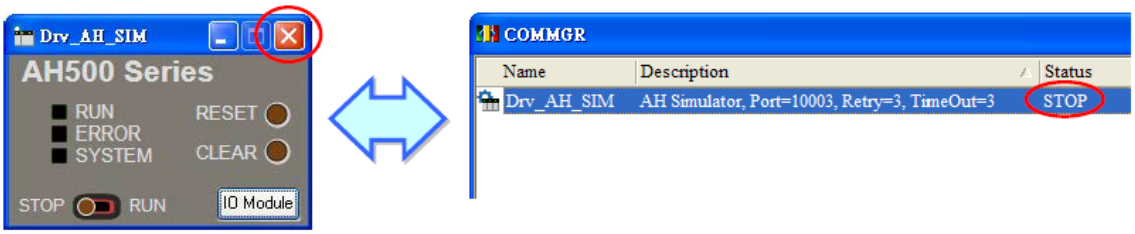
توجه به این نکته ضروری است که در حالت استفاده از شبیه ساز، دسترسی به تمامی امکانات، مشابه حالت اتصال مستقیم نرم افزار به PLC امکان پذیر نیست. به عنوان نمونه ایجاد حالت آنلاین در HWCONFIG و یا استفاده از NWCONFIG در آن امکان پذیر نخواهد بود. برای مشاهده لیست کامل محدودیت ها به راهنمای نرم افزار مراجعه فرمایید.

۱۱-۴-۱- پنجره شبیه ساز AH500

LEDها وکلیدهای روی این پنجره مشابه مدل‌های واقعی CPUهای خانواده سری ۵۰۰ می‌باشد. کاربر می‌تواند همانند یک CPU سری ۵۰۰ واقعی با این پنجره ارتباط برقرار کند و عملکرد آن را مورد ارزیابی قرار دهد.

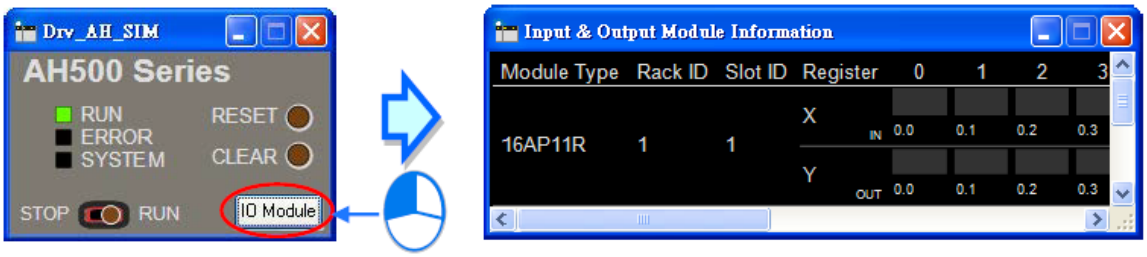
شکل ۱۱-۵۹ پنجره شبیه ساز AH500

پس از بسته شده صفحه شبیه ساز، وضعیت درایور آن در COMMGR به حالت STOP در می‌آید و برعکس. همچنین پس از بسته شدن شبیه ساز برنامه نوشته شده در آن تا اجرای مجدد باقی خواهد ماند.



شکل ۱۱-۶۰ وضعیت مشابه فعال بودن درایور و شبیه ساز

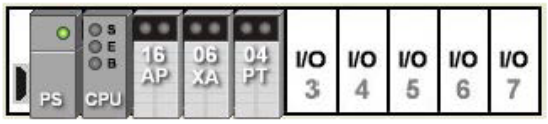
تنظیمات سخت افزاری را می‌توان در شبیه ساز دانلود و نتیجه را مشاهده کرد. اگر ماژول دانلود شده شامل ورودی/خروجی‌های دیجیتال، آنالوگ و یا دما باشد، صفحه مرتبط با آن پس از دانلود سخت افزار با کلیک بر روی IO Module قابل دسترسی است.



شکل ۱۱-۶۱ صفحه ورودی/خروجی شبیه ساز

شبیه ساز قابلیت اعمال سناریو بر روی ورودی، خروجی و یا حافظه‌ها را ندارد ولی با این حال می‌توان با استفاده از آن وضعیت ورودی و خروجی و حافظه ها را تغییر و با استفاده از مانیتورینگ آنلاین عیب یابی و در نتیجه برنامه را ویرایش کرد.

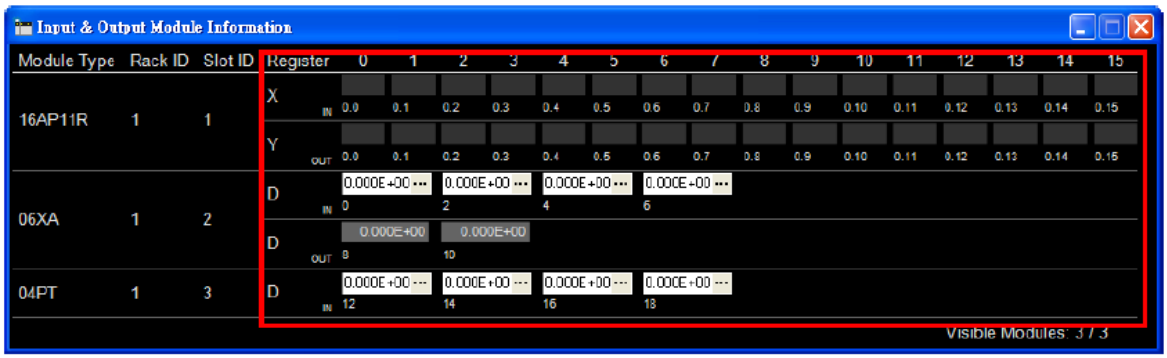
در صفحه‌ی مربوط به شبیه سازی پورت‌های ورودی و خروجی می‌توان ورودی، خروجی و حافظه‌های سیستم مونتاژ شده در تنظیمات سخت افزاری را مشاهده کرد. به عنوان نمونه در زیر شبیه ساز را برای 16AP11R-5A، AH06XA-5A و AH04PT-5A مشاهده می‌کنید. نام ماژول‌های مورد استفاده، Rack مورد نظر، شماره Slot و وضعیت پورت های ورودی و خروجی از جمله مواردی است که در شبیه ساز ورودی/خروجی نمایش داده خواهد شد. مشاهده می‌شود که وضعیت شبیه ساز و نگاشت شماره پورت ها بر روی آن و دیگر تنظیمات سخت افزاری مشابه تنظیمات HWCONFIG است.



Information: Rack 1

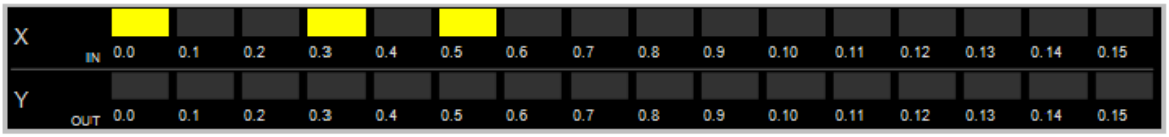
Slot No.	Label	Firmware...	Description	Input Device ...	Output Device ...	Comment
-	AHPS05-5A	-	AH Power Supply Module	None	None	
-	AHCPU530-EN	1.00	Basic CPU module building	None	None	
0	AH16AP11R-5A	-	8 x DI VDC, 8 x DO VAC/VI	X0.0 ~ X0.15	Y0.0 ~ Y0.15	
1	AH06XA-5A	1.00	4 x 16bit AI, 2 x 16bit AO	D0 ~ D7	D8 ~ D11	
2	AH04PT-5A	1.00	4 x 3/4 wires RTD input 0.1	D12 ~ D19		
3						

شکل ۱۱-۶۲ تنظیمات سخت افزاری برای پیاده سازی در شبیه ساز AH500



شکل ۱۱-۶۳ شبیه ساز آماده شده مشابه تنظیمات سخت افزاری

کاربر می‌تواند وضعیت خروجی ها را بدون آنکه بتواند آن‌ها را تغییر دهد، در شبیه ساز مشاهده کند. ولی ورودی را می‌تواند با کلیک بر روی آن‌ها در شبیه ساز تغییر دهد.

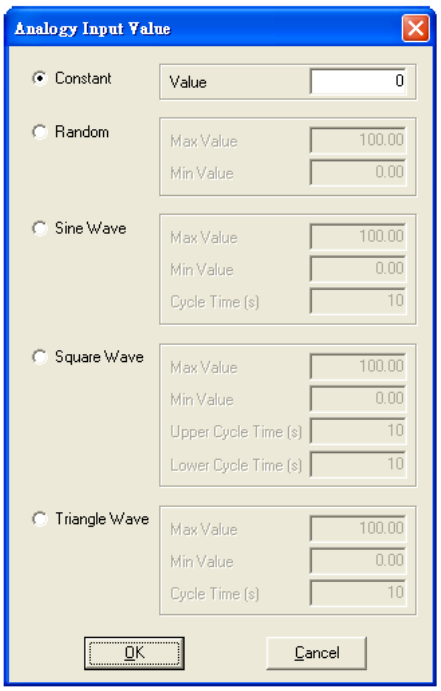


شکل ۱۱-۶۴ امکان تغییر ورودی های شبیه ساز و وابسته بودن خروجی های آن به برنامه

همچنین در AH06XA-5A ورودی و خروجی های تخصیص داده شده مشابه زیر خواهند بود. مقادیر مشخص شده ۳۲ بیت ممیز شناور می‌باشند، کاربر مقادیر خروجی D8 و D10 را نمی‌تواند تغییر دهد ولی ورودی های D0، D2، D4 و D6 را می‌تواند مقدار دهی کند. توجه شود که این مقادیر در حافظه‌های D ثبت شده و مقداری آنالوگ نمی‌باشند (تبدیل به دیجیتال شده اند).

شکل ۱۱-۶۵ ورودی و خروجی های آنالوگ شبیه ساز AH500

برای اینکه بتوان مقدار ورودی آنالوگ را تغییر داد (و یا الگویی به آن اختصاص داد) باید بر روی ... در سمت راست نمایه آن کلیک کرد تا پنجره Analog Input Value باز شود. در این پنجره می‌توان نوع سیگنال ورودی را انتخاب کرد.

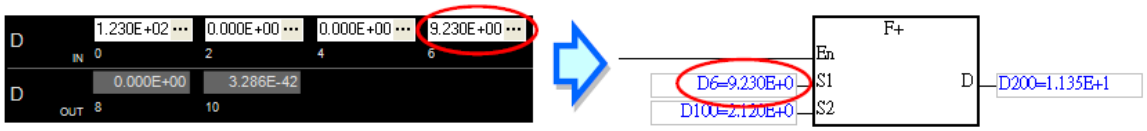


شکل ۱۱-۶۶ انتخاب نوع سیگنال ورودی آنالوگ در شبیه ساز AH500

جدول ۱۱-۵: انواع ورودی آنالوگ قابل اعمال در شبیه سازAH500

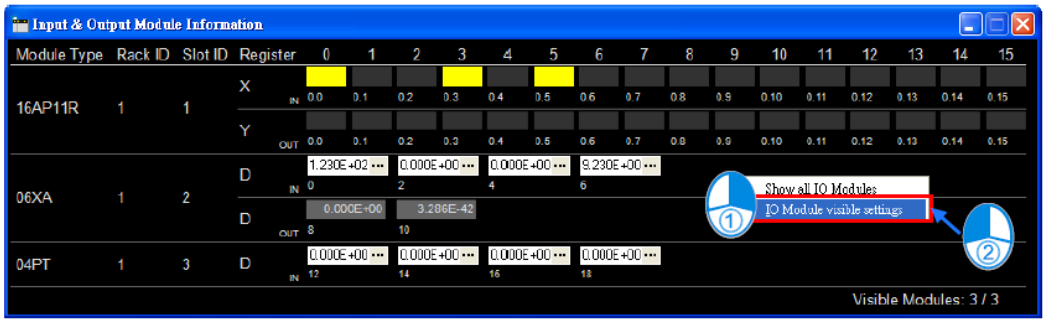
نوع سیگنال	شرح
Constant	مقدار ثابت
Random	مقدار تصادفی با امکان تعیین حداقل و حداکثر
Sine Wave	موج سینوسی با امکان تعریف فرکانس و حداقل و حداکثر دامنه
Square Wave	موج مربعی با امکان تعریف حداقل و حداکثر، زمان یک بودن و زمان صفر بودن
Triangle Wave	موج مثلثی با امکان تعریف فرکانس و حداقل و حداکثر دامنه

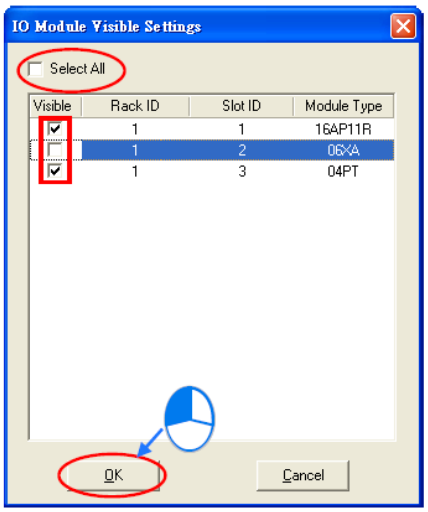
بعد از پایان تنظیمات، سیگنال ارسالی از شبیه ساز به COMMGR همان سیگنال تعیین شده در قسمت قبل خواهد بود. ISPSoft نیز سیگنال ورودی خود را از شبیه ساز به واسطه COMMGR دریافت می‌کند.



شکل ۱۱-۶۷ اعمال مقدار تعیین شده در شبیه ساز AH500

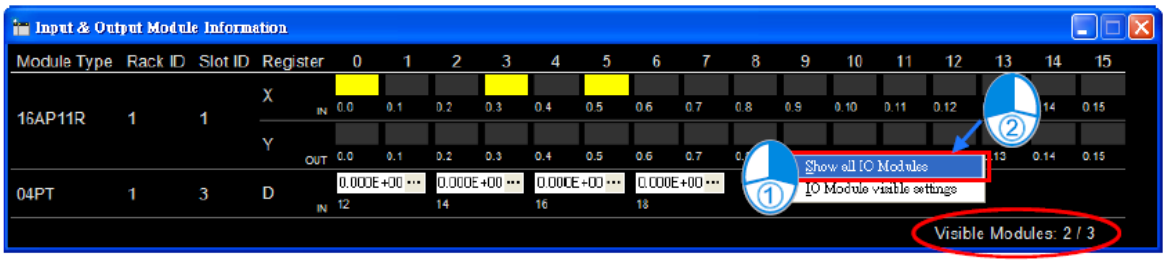
در صورتی که نخواهیم بعضی از ورودی یا خروجی ها را در شبیه ساز مشاهده کنیم، می‌توانیم بر روی صفحه ورودی و خروجی شبیه ساز کلیک راست کرده و IO Module Visible Settings را انتخاب کنیم. در پنجره باز شده می‌توانیم ورودی و خروجی های مورد نیاز خود برای نمایش را انتخاب کرده و بر روی OK کلیک کنیم.





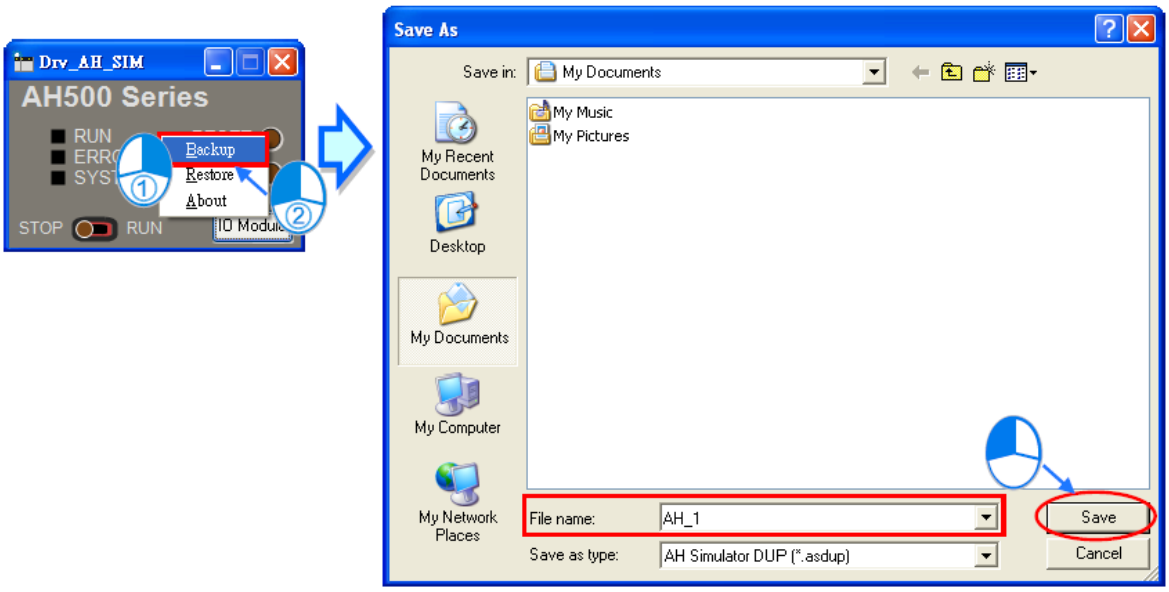
شکل ۱۱-۶۸ انتخاب ورودی و خروجی برای نمایش در شبیه ساز AH500

عبارت گوشه سمت چپ پایینِ پنجره ورودی و خروجی شبیه ساز، تعداد ماژول های ورودی و خروجی قابل مشاهده به نسبت کل ماژول های ورودی و خروجی را نمایش می‌دهد. در صورتی که بخواهیم تمامی ماژول های ورودی و خروجی نمایش داده شوند، می‌توانیم بر روی پنجره کلیک راست کرده و Show All IO Module را انتخاب کنیم.



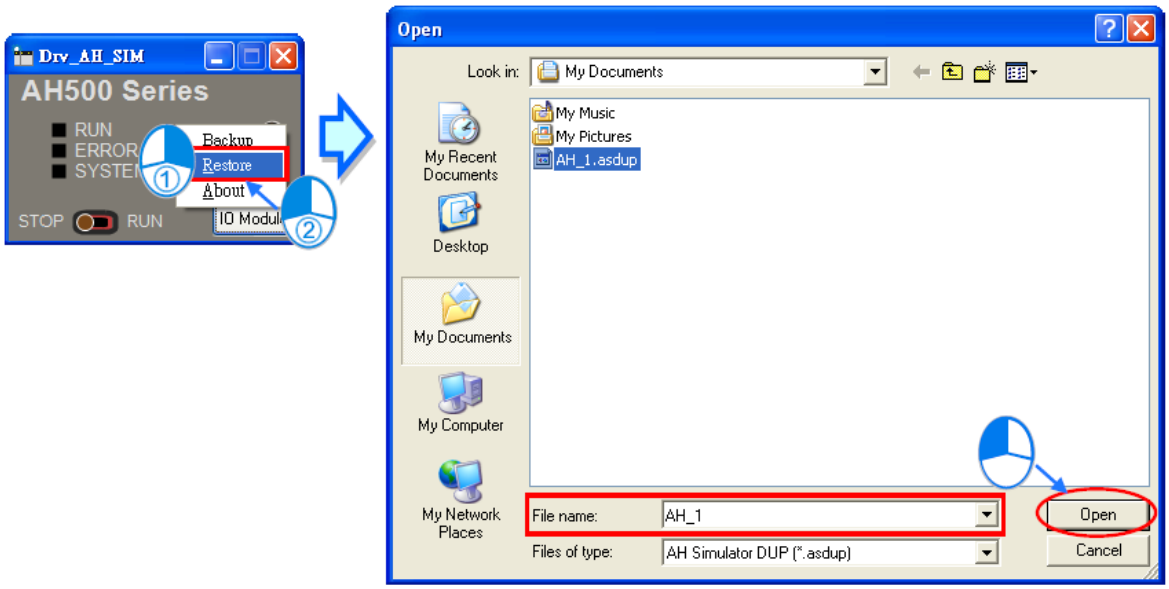
شکل ۱۱-۶۹ نمایش تمامی ماژول های ورودی و خروجی

برای تهیه نسخه پشتیبان از داده های سیمولاتور (پارامترها و برنامه بارگزاری شده در آن) در کامپیوتر، می‌توان در صفحه اصلی شبیه ساز کلیک راست کرده و Backup را انتخاب کرد سپس در Save as آدرس محل ذخیره را تعیین، نام را نیز در File Name مشخص و بر روی Save کلیک کرد.



شکل ۱۱-۷۰ تهیه نسخه پشتیبان از شبیه ساز AH500

اگر بخواهیم نسخه پشتیبان ذخیره شده را در شبیه ساز دیگری مورد استفاده قرار دهیم، می‌توانیم بر روی صفحه اصلی آن کلیک راست کرده و Restore را انتخاب نماییم. سپس با انتخاب فایل پشتیبان ذخیره شده، آن را فراخوانی کنیم.



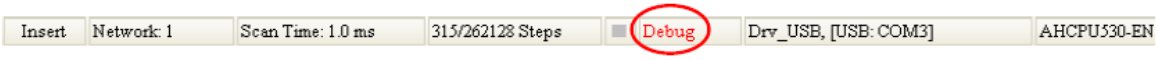
شکل ۱۱-۷۱ فراخوانی نسخه پشتیبان در شبیه ساز

۱۱-۵- عیب یابی در AH500

حالت عیب یابی برای خانواده AH500 در نرم افزار ISPSoft همه زبان‌های برنامه نویسی را پشتیبانی می‌کند. برای رفتن به حالت عیب یابی به ترتیب بر روی  و  و  کلیک می‌کنیم.

شکل ۱۱-۷۲ فعال کردن حالت عیب یابی در AH500

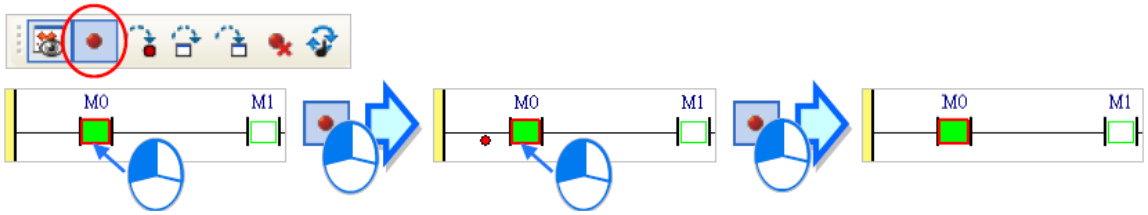
در صورتی که CPU از گذشته در حالت عیب یابی مانده باشد، نمی‌توان حالت عیب یابی را دوباره برای آن در نرم افزار فعال کرد. در این حالت Debug در نوار وضعیت ظاهر خواهد شد. دلیل این موضوع آن است که وضعیت عیب یابی دفعه‌ی گذشته به صورت عادی غیرفعال نشده است، برای اینکه کاربر بتواند حالت عیب یابی را دوباره در ISPSoft فعال کند باید CPU را Stop کرده و دوباره Start کند و سپس حالت عیب یابی را فعال کند.



شکل ۱۱-۷۳ حالت عیب یابی در نوار وضعیت

۱۱-۵-۱- نقطه انفصال

در پروژه‌های AH500 حداکثر ۱۰ نقطه انفصال برای عیب یابی سیستم می‌توان قرار داد. پس از آنکه کاربر نقطه مورد نظر برای ایجاد نقطه انفصال در برنامه را با کلیک انتخاب نماید، می‌تواند در آن محل با کلیک بر روی یک نقطه انفصال ایجاد کند. با کلیک دوباره بر روی موقعیت نقطه انفصال ایجاد شده و کلیک بر روی ، نیز می توان نقطه انفصال ایجاد شده را حذف کرد.

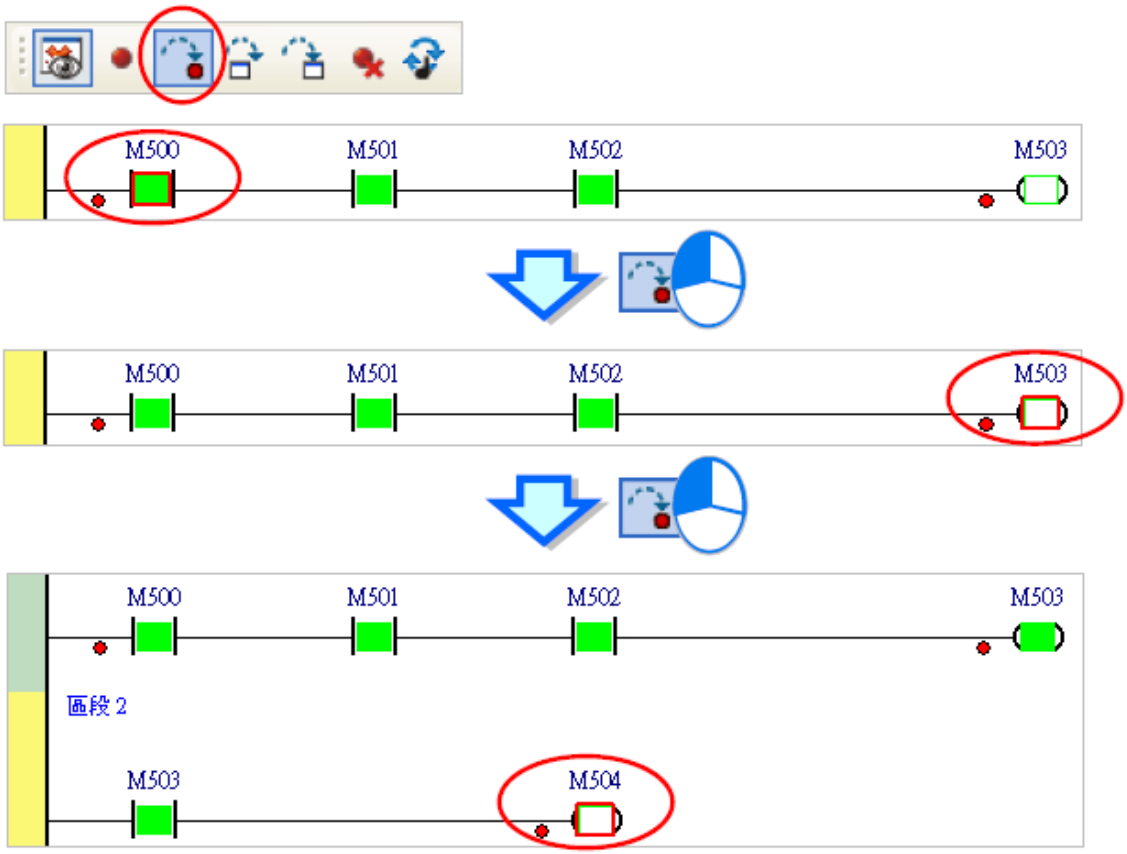


شکل ۱۱-۷۴ ایجاد نقطه انفصال در حالت عیب یابی AH500

برای حذف تمامی نقاط انفصال در برنامه می‌توان از استفاده کرد.

شکل ۱۱-۷۵ حذف تمامی نقاط انفصال در حالت عیب یابی AH500

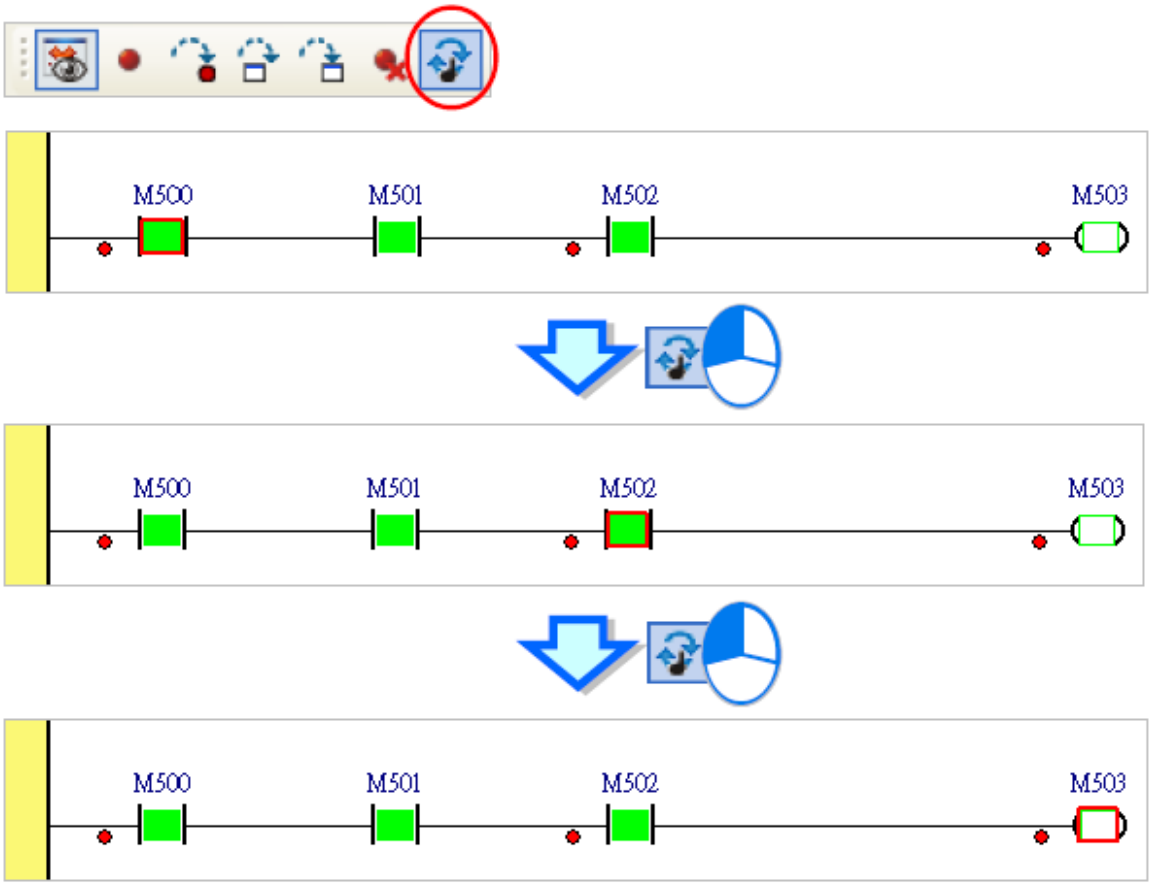
با کلیک بر روی ، اجرای برنامه تا اولین نقطه انفصال ادامه پیدا می‌کند. با کلیک دوباره بر روی ، برنامه تا نقطه انفصال بعدی اجرا خواهد شد. توجه شود که نقطه ای که برنامه در آن متوقف می-شود، جایی است که خود آن نقطه اجرا نشده است.



شکل ۱۱-۷۶ اجرای برنامه از نقطه انفصال به نقطه انفصال بعدی

۱۱-۵-۲- اجرای پیوسته

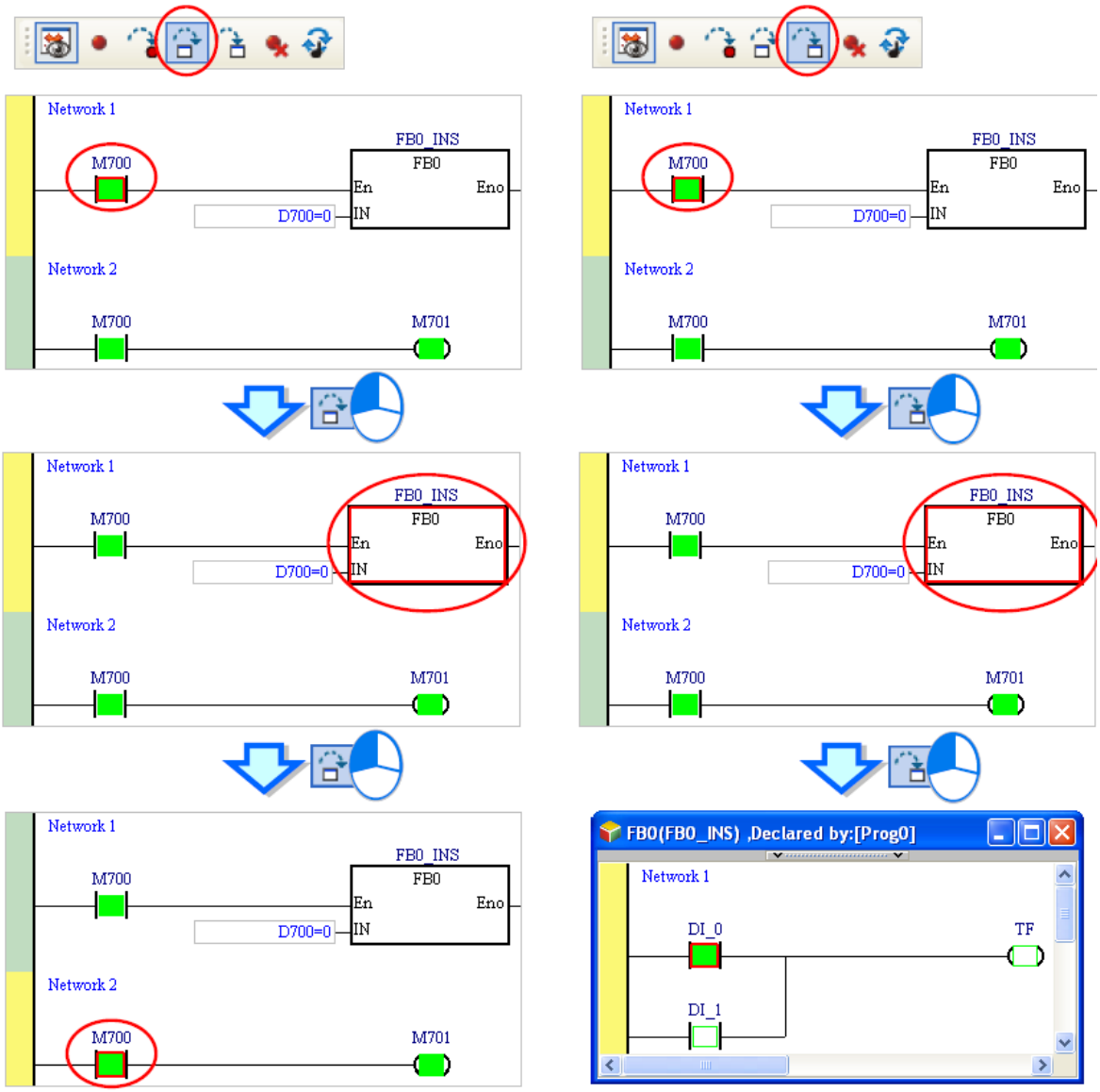
با کلیک بر روی اجرای برنامه تا نقطه انفصال بعدی ادامه خواهد داشت. در صورتی که نقطه انفصالی در برنامه وجود نداشته باشد، با کلیک بر روی اجرای برنامه به صورت پیوسته ادامه خواهد داشت.



شکل ۱۱-۷۷ اجرای برنامه تا نقطه انفصال بعدی و یا اجرای پیوسته

۱۱-۵-۳- اجرای مرحله به مرحله

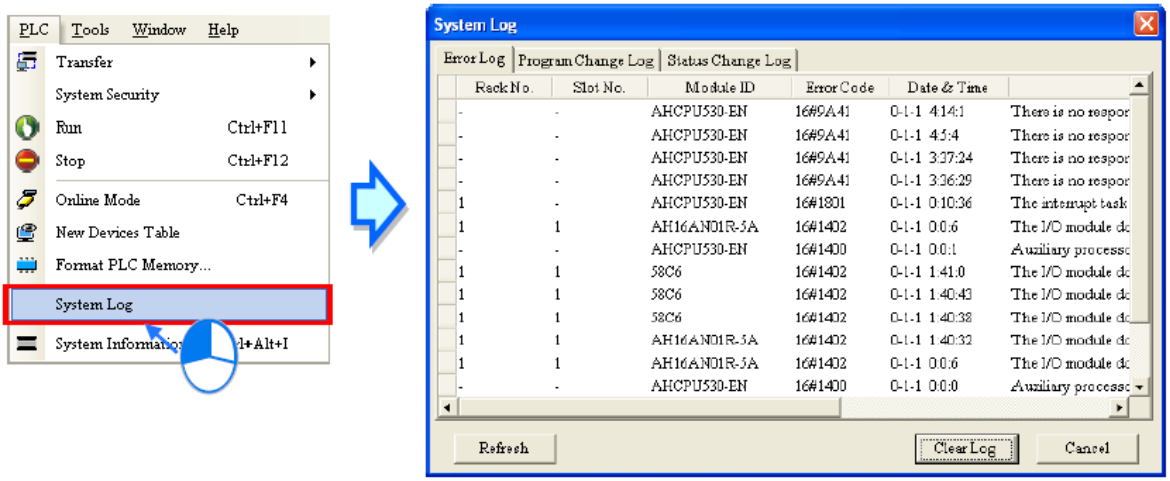
با استفاده از و می‌توان اجرای مرحله به مرحله داشت. این دو عملکرد در هر مرحله تنها یک بلوک در برنامه نویسی اجرا می‌کنند. تنها تفاوت این دو آن است که اولی یعنی FB را یک بلوک در نظر می‌گیرد و در یک مرحله اجرا می‌کند ولی المان‌های درون FB را نیز مرحله به مرحله اجرا می‌کند (در صورتی که FB کلمه عبور داشته باشد و یا اینکه EN آن فعال نباشد، برنامه در آن اجرا نخواهد شد). تفاوت این دو بلوک را در شکل زیر می‌توان دید.



شکل ۱۱-۷۸ اجرای مرحله به مرحله در حالت عیب یابی AH500

۱۱–۶– گزارش عملکرد PLC (فقط برای مدل های AH500)

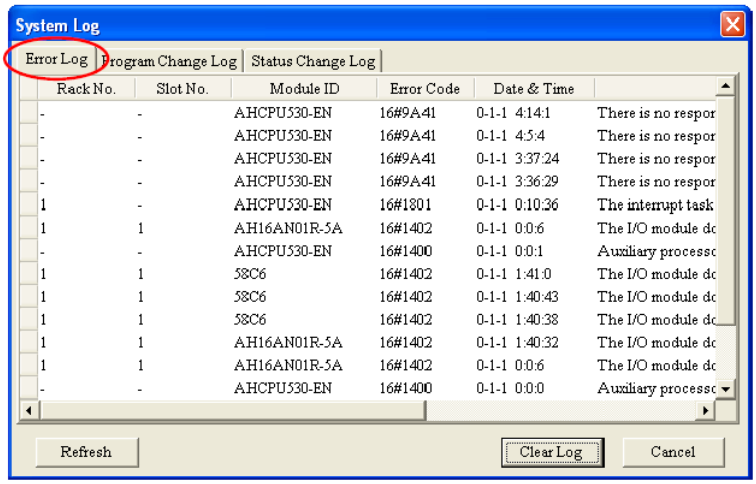
عملیات‌ها و خطاهایی که در CPUهای سری AH500 رخ می‌دهد در حافظه‌ای خاص در PLC ذخیره می‌شود که با استفاده از ISPSoft می‌توان به آن‌ها دسترسی داشت. برای اینکار پس از اتصال نرم افزار به PLC به System Log در سربرگ PLC رفته تا پنجره آن باز شود. در این مرحله داده‌ها از PLC فراخوانی و برای کاربر نمایش داده می‌شود (توجه شود که این پنجره به صورت اتوماتیک به روز نمی شود و برای به روز رسانی آن باید بر روی Refresh کلیک شود).



شکل ۱۱-۷۹ گزارش عملکرد PLC

۱۱-۶-۱- سربرگ Error Log

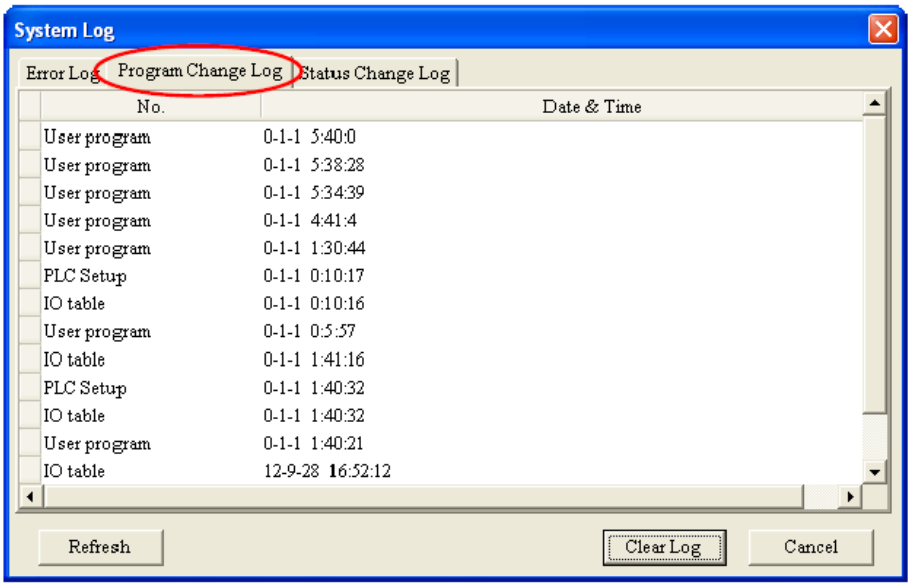
با انتخاب سربرگ Erorr Log می‌توان خطاهای رخ داده در ماژول CPU و ورودی/خروجی را مشاهده کرد. کد خطا و شرح و زمان آن کد به همراه مشخصات ماژولی که در آن خطا رخ داده است را می‌توانیم در این بخش مشاهده کنیم. با کلیک بر روی Refresh اطلاعات این صفحه به روز می‌شود. همچنین با کلیک بر روی Clear Log لیست خطاها در CPU پاک می‌شود. توجه شود که در صورتی که خطا را مرتفع نکنیم و فقط آن را از لیست پاک کنیم، آن خطا دوباره اتفاق افتاده و لیست خواهد شد.



شکل ۱۱-۸۰ گزارش عملکرد PLC- سربرگ Error Log

۱۱-۶-۲- سربرگ Program Change Log

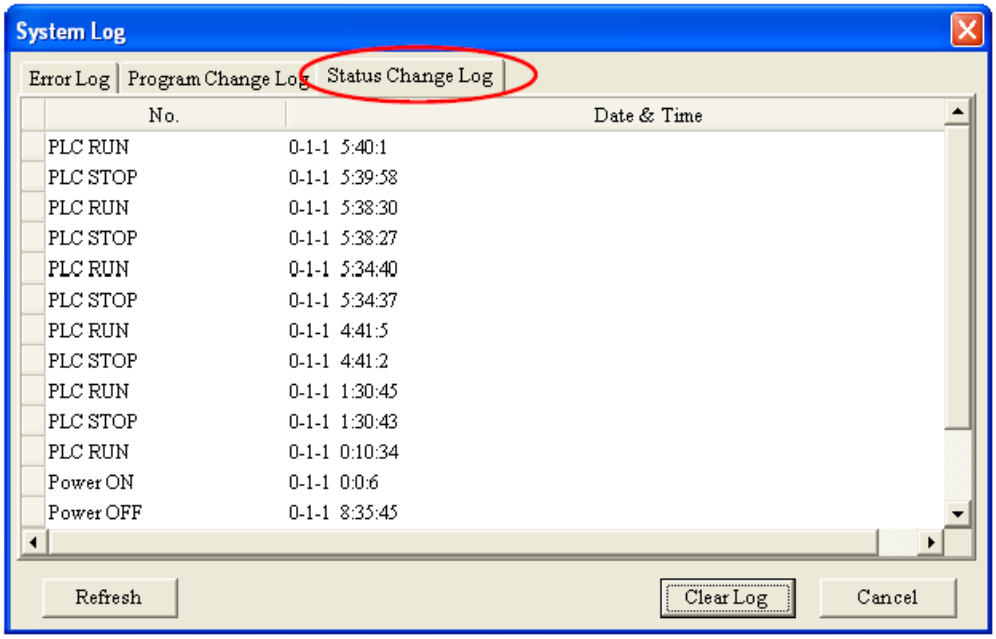
در این صفحه می‌توان اطلاعات مربوط به بارگزاری و استخراج‌های قبلی برنامه و یا پارامترهای PLC را مشاهده کرد. با کلیک بر روی Refresh اطلاعات این صفحه به روز می‌شود. همچنین با کلیک بر روی Clear Log این لیست در CPU پاک می‌شود.



شکل ۱۱-۸۱ گزارش عملکرد PLC- سربرگProgram Change Log

۱۱-۶-۳- سربرگ Status Change Log

در این بخش می‌توان تغییر وضعیت‌ها در اجرای CPU را مشاهده کرد. با کلیک بر روی Refresh اطلاعات این صفحه به روز می‌شود. همچنین با کلیک بر روی Clear Log این لیست در CPU پاک می-شود.



شکل ۱۱-۸۲ گزارش عملکرد PLC- سربرگ Status Change Log

فصل ۱۲– ابزارها و امکانات جانبی ISPSoft

۱۲–۱– ابزارهای اصلاح و توابع کمکی در ISPSoft

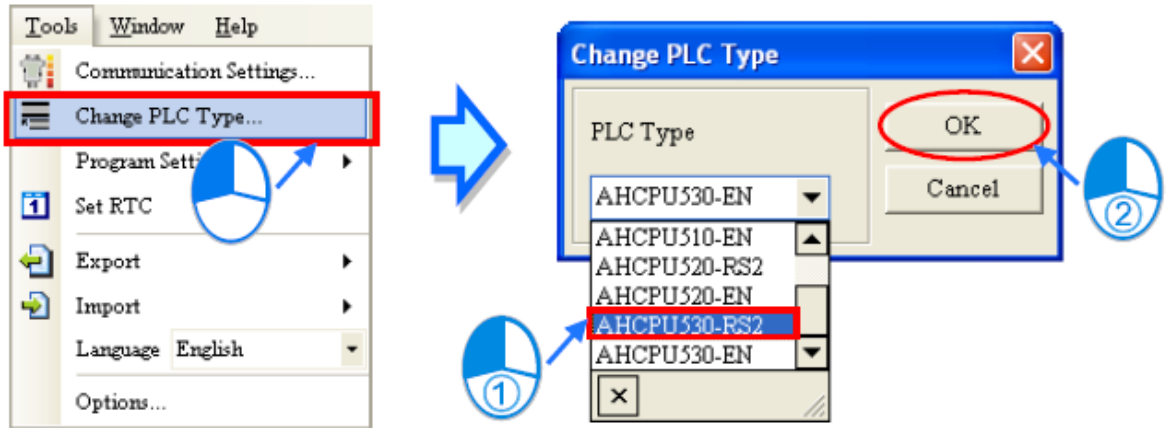
۱۲–۱–۱– تغییر مدل PLC

می‌توان با توجه به دو نکته زیر مدل PLC را در پروژه تغییر داد:

۱- مدل‌های AH500 با مدل‌های DVP سازگار نیستند و کاربر نمی‌تواند پردازنده سری AH500 را به DVP تبدیل کند.

۲- در صورتی که نوع PLC تغییر کند، ممکن است عملکرد، محدوده حافظه‌ها، و دستورالعمل‌هایی که سیستم پشتیبانی می‌کند تغییر کنند. برای همین لازم است کاربر بررسی کند که آیا برنامه، پارامترها، تنظیمات سخت افزاری و تنظیمات شبکه همچنان معتبر هستند و یا اینکه باید تغییر کنند.

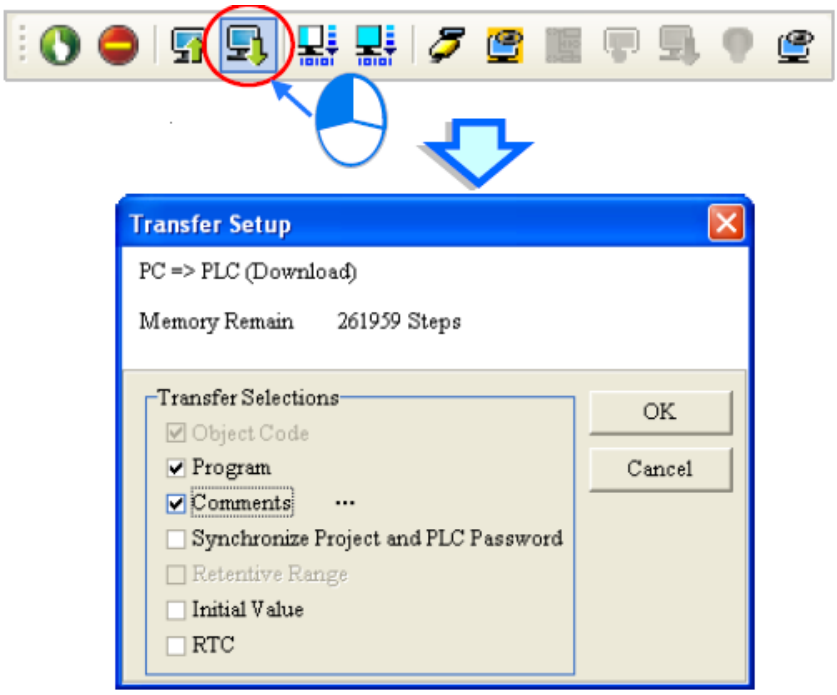
برای تغییر نوع PLC در منوی Tools بر روی Change PLC Type کلیک کرده و در پنجره باز شده نوع PLC جایگزین را وارد می‌کنیم.



شکل ۱۲-۱ تغییر نوع PLC

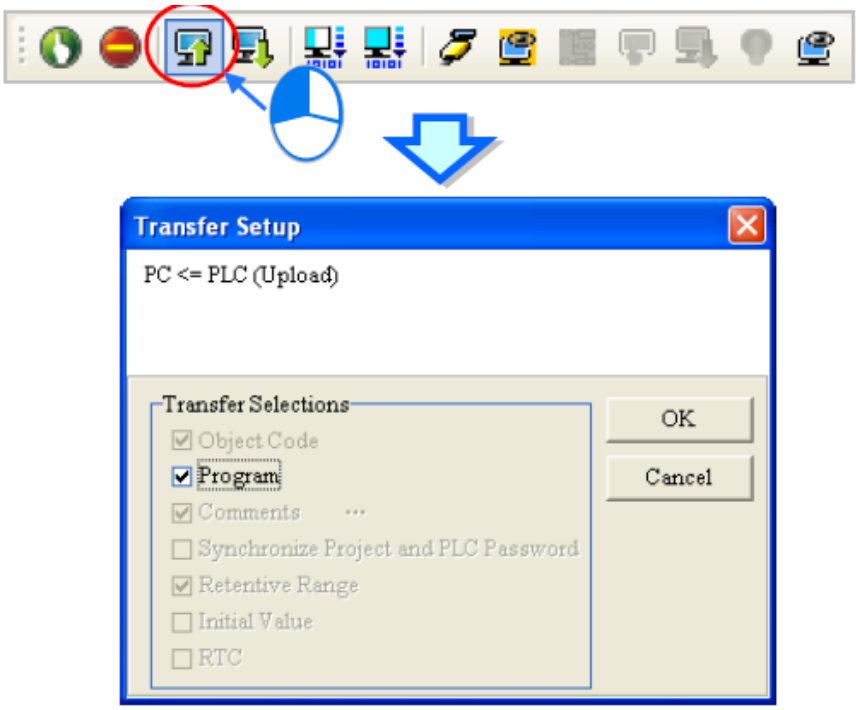
۱۲-۱-۲- بارگزاری و استخراج پروژه

می‌توان علاوه بر برنامه، پارامترهای پروژه را نیز بارگزاری و استخراج کرد. برای اینکه بتوان برنامه‌ای را در PLC بارگزاری کرد می‌توان در نوار ابزار بر روی  کلیک کرد تا پنجره Transfer Setup باز شود، در این پنجره می‌توان موارد مورد نیاز برای بارگزاری را انتخاب و سپس بر روی OK کلیک کرد.



شکل ۱۲-۲ بارگزاری بخش های مختلف پروژه

برای استخراج پروژه‌ای که قبلا در PLC بارگزاری شده نیز می‌توان بر روی  در نوار ابزار کلیک کرده تا پنجره Transfer Setup باز شود، در این پنجره می‌توان موارد مورد نیاز برای استخراج را انتخاب و سپس بر روی OK کلیک کرد.



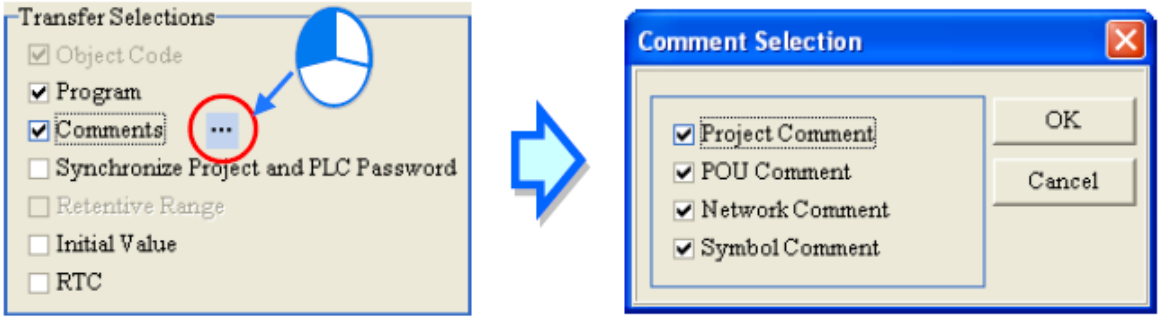
شکل ۱۲-۳ استخراج بخش های مختلف پروژه

با آنکه گزینه‌ها در پنجره Transfer Setup برای بارگزاری و استخراج یکسان است ولی گزینه‌های لازم و گزینه‌های غیرفعال برای هرکدام از این دو فرایند متفاوت است.

جدول ۱۲-۱: گزینه های پنجره بارگزاری و استخراج

گزینه انتقال	شرح
Object Code	Object Code کُدی است که کامپایلر تولید می‌کند (باید انتخاب شود)
Program	Program کُدی است که کاربر تولید می‌کند
Comments	توضیحات اضافه شده به پروژه
Synchronize Project and PLC Password	در حین انتقال پروژه، گذرواژه پروژه با PLC یکی خواهد شد.
Retentive Range	محدوده حافظه‌های نگهدار (فقط برای CPUهای خانواده DVP)
Initial Value	مقادیر اولیه سیمبول‌ها در حافظه مرتبط در PLC کپی می‌شوند.
RTC	زمان و تاریخ کامپیوتر به PLC منتقل می‌شود.

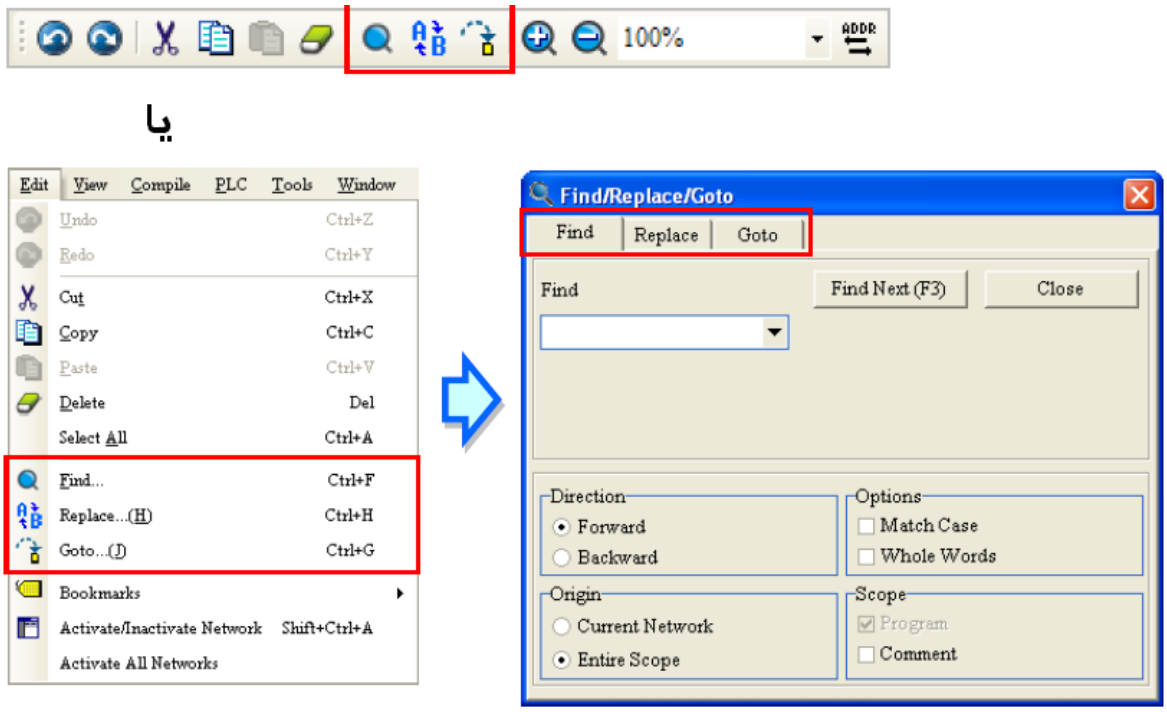
نکته: در صورتی که گزینه Comments را انتخاب کنیم، در کنار آن علامت ... فعال می‌شود که به کاربر اجازه می‌دهد نوع توضیحات را انتخاب کند.



شکل ۱۲-۴ تعیین نوع توضیحات برای بارگزاری

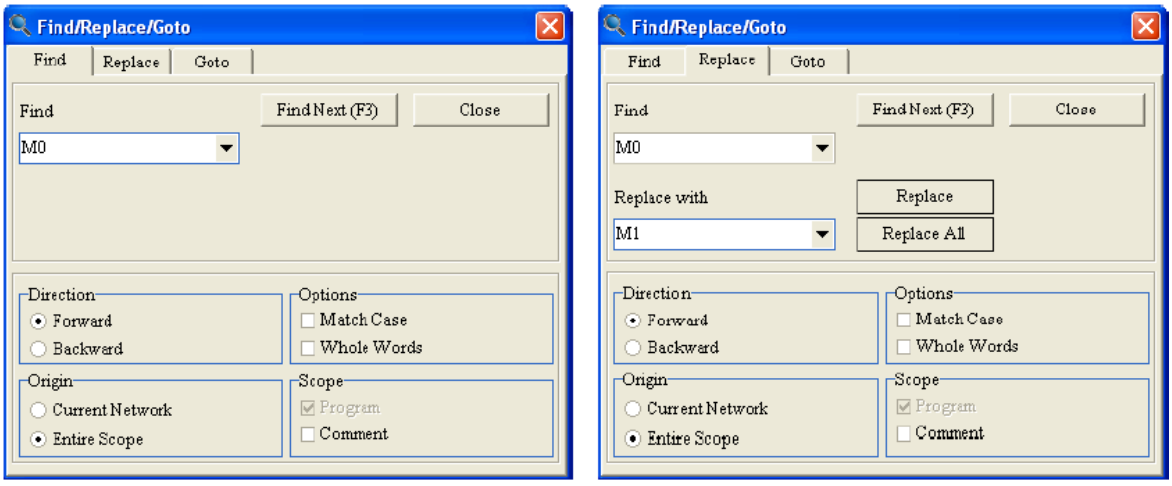
۱۲-۱-۳- Find/Replace/Goto در Ladder و FBD

عملکرد توابع Find/Replace/Goto برای زبان‌های مختلف برنامه نویسی متفاوت است. در این بخش به شرح این توابع در زبان برنامه نویسی Ladder و FBD می‌پردازیم. از این توابع زمانی که در محیط ویرایش برنامه قرار داریم می‌توانیم استفاده کنیم. برای اینکار باید بر روی محیط برنامه نویسی و یا بخش سیمبولی که می‌خواهیم در آن ویرایش را انجام دهیم کلیک کرده و سپس از منوی Edit، گزینه Find، Replace و یا Goto را انتخاب کنیم، پنجره‌ای که در این مرحله باز می‌شود با توجه به آیتم انتخابی ممکن است متفاوت باشد. همچنین می‌توان برای اعمال هرکدام از این عملکردها بر روی آیکون آن‌ها در نوار ابزار () کلیک کرد. در پنجره باز شده سه سربرگ Find، Replace و Goto را می‌توان انتخاب کرد.



شکل ۱۲-۵ پنجره Find/Replace/Goto در زبان برنامه نویسی Ladder و FBD

در سربرگ Find می‌توان عبارتی را که می‌خواهیم آن را جستجو کنیم را وارد کرده و با کلیک بر روی Find Next و یا فشردن F3 آن را بیابیم. حتی اگر پنجره بسته شود با فشردن دوباره F3 می‌توان عملیات جستجو را ادامه داد. همچنین در سربرگ Replace می‌توان با نوشتن کلمه‌ای در بخش Find، آن را یافته و کلمه مقابل بخش Replace را جایگزین آن کرد. در سربرگ Replace با کلیک بر گزینه Replace، آخرین نتیجه جستجو جایگزین می‌شود، و با کلیک بر روی Replace All تمامی کلمات جستجو شده (مطابق با عبارت Find) جایگزین می‌شوند.



شکل ۱۲-۶ جستجو و جایگزینی در زبان برنامه نویسی Ladder و FBD

شرح گزینه‌هایی که در این دو صفحه وجود دارد به صورت زیر است.

جدول ۱۲-۲: گزینه های صفحه جستجو و جایگزینی در زبان برنامه نویسی Ladder و FBD

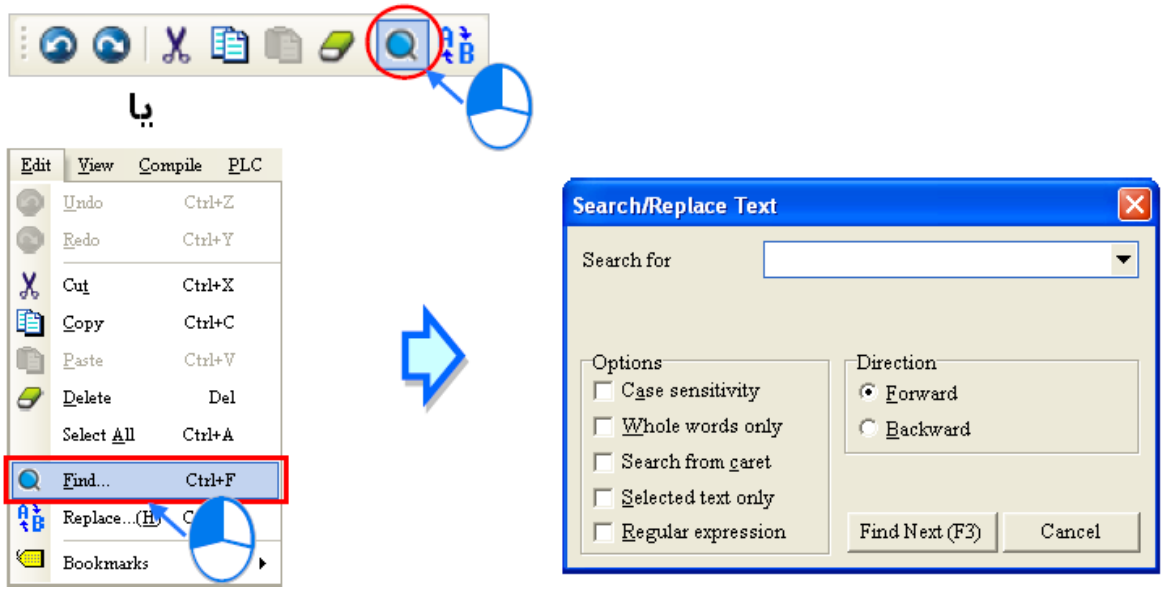
گزینه‌ها	شرح
Find	کاربر می‌تواند پارامتری را که می‌خواهد جستجو کند در این قسمت تایپ و یا انتخاب کند.
Direction	جهت جستجو (به طرف بالا و یا پایین)
Option	• Match Case: جستجو برای کلماتی که شامل کلمه جستجو شده هستند. • Whole Words: جستجو برای کلماتی که دقیقا مطابق کلمه جستجو شده باشند (نه اینکه کلمه جستجو شده فقط قسمتی از آن‌ها باشد).
Origin	• Current Network: جستجو از موقعیت انتخاب شده آغاز می‌شود. • Entire Scope: جستجو از ابتدای برنامه شروع می‌شود.
Scope	انتخاب محدوده جستجو

اگر کاربر بخواهد به بخش مشخصی از برنامه برود می‌تواند در سربرگ Goto، نوع مقصد (سربرگ Network و یا نشان) را در قسمت Location Type وارد کند و مقصد را در بخش Index انتخاب کرده و بر روی Goto کلیک کند.

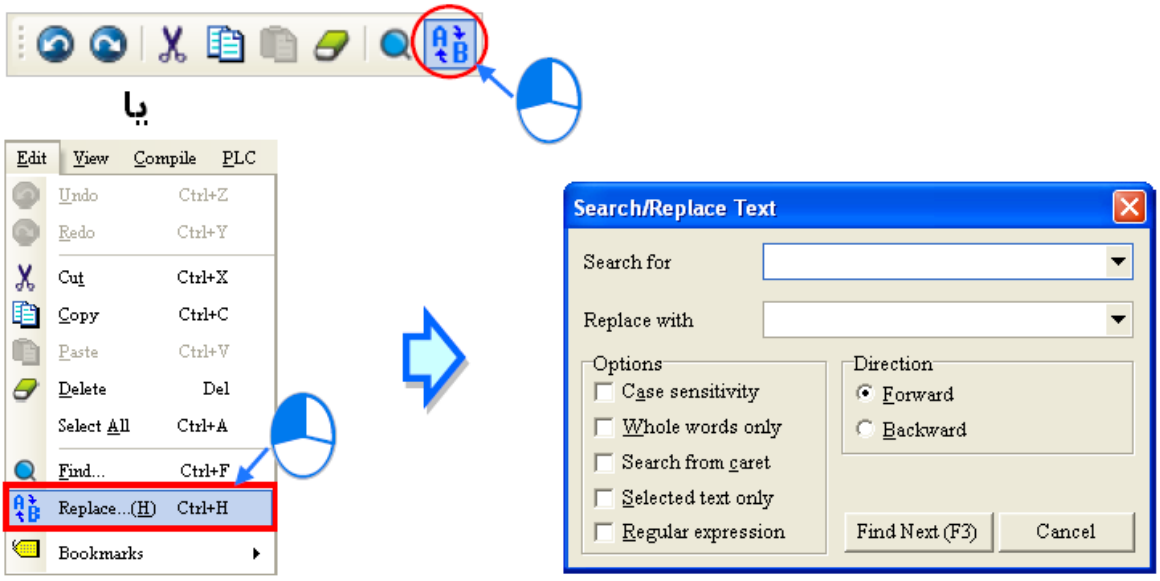
شکل ۱۲-۷ استفاده از Goto در زبان برنامه نویسی Ladder و FBD

۱۲-۱-۴- Find/Replace در Instruction List و Structured Text

در دو زبان IL و ST می‌توان از دو ابزار جستجو و جایگزینی استفاده کرد، از این توابع زمانی که در محیط ویرایش برنامه قرار داریم می‌توانیم استفاده کنیم. برای اینکار باید بر روی محیط ویرایش برنامه و یا بخش سیمبولی که می‌خواهیم در آن ویرایش را انجام دهیم کلیک کرده و سپس از منوی Edit، گزینه Find و یا Replace را انتخاب کنیم. همچنین می‌توان برای اعمال هرکدام از این عملکردها بر روی آیکون آن‌ها در نوار ابزار ( ) کلیک کرد.



شکل ۱۲-۸ پنجره Find در زبان برنامه نویسی IL و ST



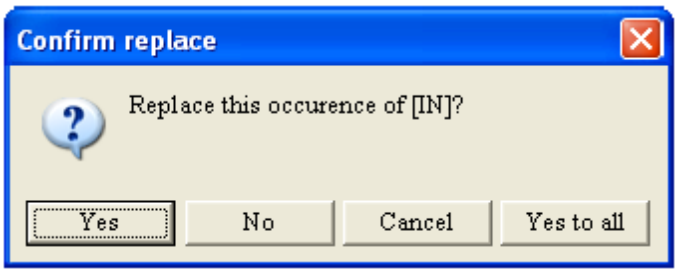
شکل ۱۲-۹ پنجره Replace در زبان برنامه نویسی IL و ST

گزینه‌هایی که در این دو صفحه وجود دارد به صورت زیر است.

جدول ۱۲-۳: گزینه های صفحه جستجو و جایگزینی در زبان برنامه نویسی IL و ST

گزینه‌ها	شرح
Search for	کاربر می‌تواند پارامتری را که می‌خواهد جستجو کند در این قسمت تایپ و یا انتخاب کند.
Case Sensivity	جستجو برای کلماتی که شامل کلمه جستجو شده هستند.
Whole Words only	جستجو برای کلماتی که دقیقا مطابق کلمه جستجو شده باشند (نه اینکه کلمه جستجو شده فقط قسمتی از آن‌ها باشد).
Search from Caret	جستجو از محل کلیک شده شروع شود (در غیر این صورت از ابتدا شروع خواهد شد)
Search text only	جستجو در قسمت انتخاب شده انجام شود (در غیر این صورت محدوده جستجو کل برنامه خواهد بود)
Regular expression	فعال کردن امکان جستجوی Regular expressions برای جستجوی عبارات استاندارد
Direction	جهت جستجو (به طرف بالا و یا پایین)

در صورت استفاده از عملکرد Replace، صفحه تایید آن باز خواهد شد.



شکل ۱۲-۱۰ صفحه تایید عملکرد Replace در زبان برنامه نویسی IL و ST

گزینه‌های آمده در این صفحه به شرح زیر می‌باشند.

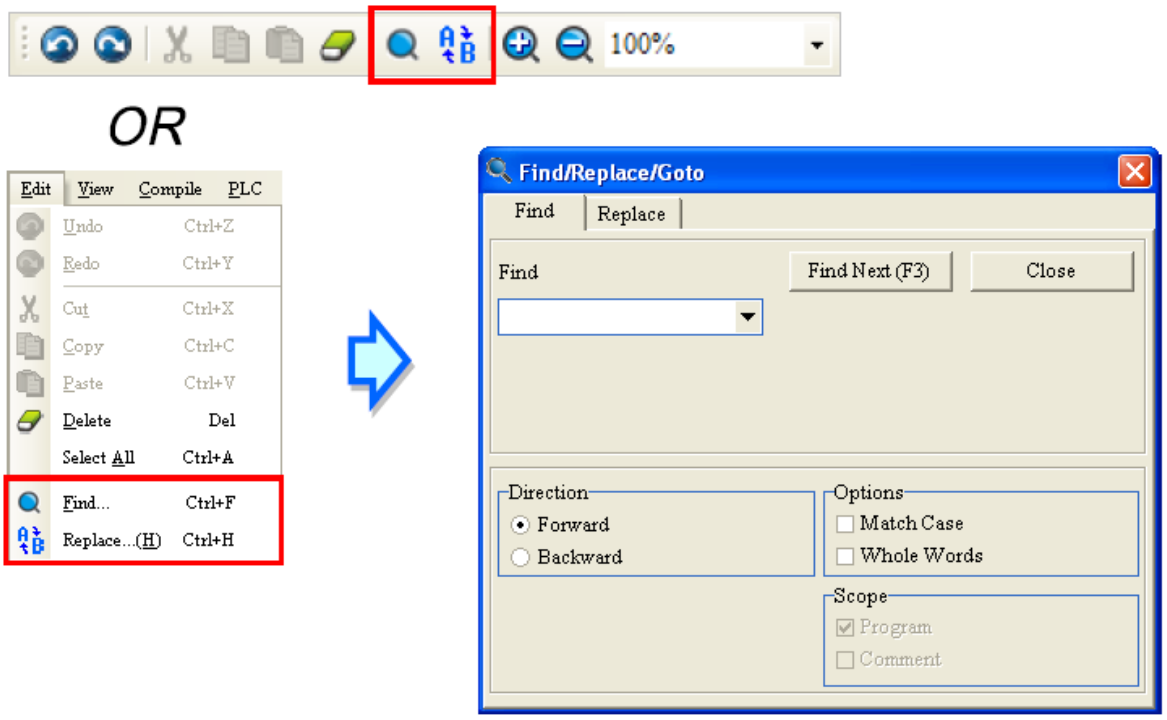
جدول ۱۲-۴: گزینه های صفحه تایید عملکرد Replace در زبان برنامه نویسی IL و ST

گزینه	عملکرد
Yes	جایگزینی صورت می‌پذیرد و جستجو ادامه پیدا می‌کند.
No	جایگزینی صورت نمی‌پذیرد و جستجو ادامه پیدا می‌کند.
Cancel	جستجو لغو و پنجره بسته می‌شود.
Yes to all	تمامی نتایج جستجو جایگزین می‌شوند.

۱۲-۱-۵- Find/Replace در SFC برای STEP و Transition

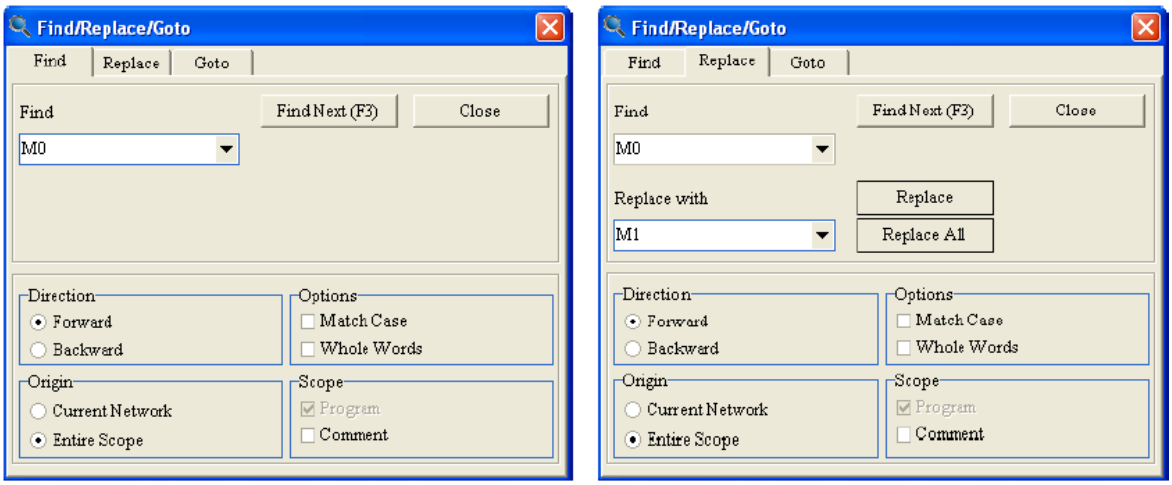
در زبان SFC می‌توان از دو ابزار جستجو و جایگزینی فقط برای Stepها و Transitionها استفاده کرد. در صورتی که کاربر بخواهد عملکرد جستجو و جایگزینی را در action/transition به کار ببرد، باید صفحات آن‌ها را باز کند و با توجه به زبان برنامه نویسی آن‌ها و توضیحات بخش‌های قبل اقدام به این کار کند.

از این توابع زمانی که در محیط ویرایش برنامه قرار داریم می‌توانیم استفاده کنیم. برای اینکار باید بر روی محیط ویرایش برنامه و یا بخش سیمبولی که می‌خواهیم در آن ویرایش را انجام دهیم کلیک کرده و سپس از منوی Edit، گزینه Find و یا Replace را انتخاب کنیم. همچنین می‌توان برای اعمال هرکدام از این عملکردها بر روی آیکون آن‌ها در نوار ابزار () کلیک کرد. توجه کنید که در صورتی که در صفحه برنامه SFC اقدام به جستجو و یا جایگزینی کنیم، تغییرات صرفا در همان صفحه (و نه در دیگر صفحات مرتبط برنامه) اعمال می‌شود.



شکل ۱۲-۱۱ پنجره Find/Replace در زبان برنامه نویسی SFC

در سربرگ Find می‌توان عبارتی را که می‌خواهیم آن را جستجو کنیم را وارد کرده و با کلیک بر روی Find Next و یا فشردن F3 آن را بیابیم. حتی اگر پنجره بسته شود با فشردن دوباره F3 می‌توان عملیات جستجو را ادامه داد. همچنین در سربرگ Replace می‌توان با نوشتن کلمه‌ای در بخش Find، آن را یافته و کلمه مقابل بخش Replace را جایگزین آن کرد. در سربرگ Replace با کلیک بر گزینه Replace، آخرین نتیجه جستجو جایگزین می‌شود، و با کلیک بر روی Replace All تمامی کلمات جستجو شده (مطابق با عبارت Find) جایگزین می‌شوند.



شکل ۱۲-۱۲ جستجو و جایگزینی در زبان برنامه نویسی Ladder و FBD

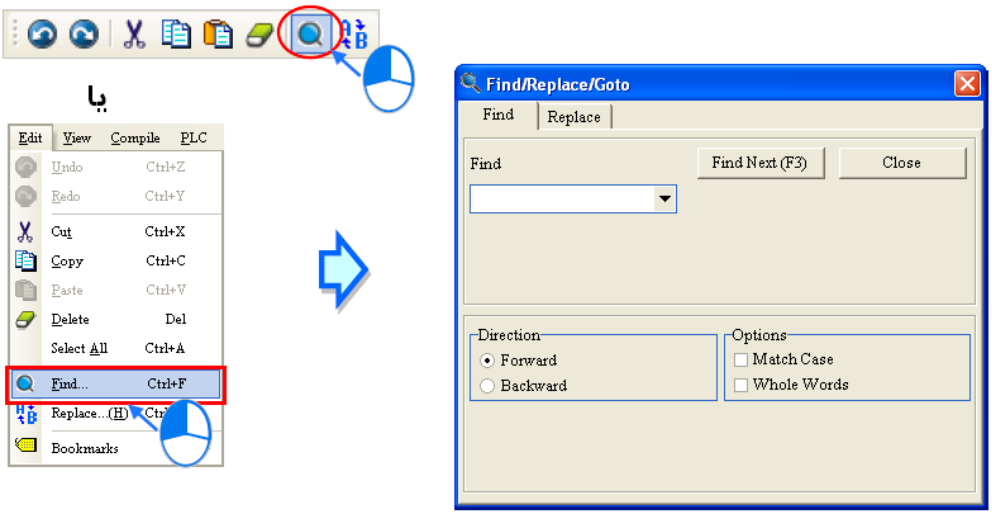
گزینه‌هایی که در این دو صفحه وجود دارد به صورت زیر است.

جدول ۱۲-۵: گزینه های صفحه جستجو و جایگزینی در زبان برنامه نویسیSFC

گزینه‌ها	شرح
Find	کاربر می‌تواند پارامتری را که می‌خواهد جستجو کند در این قسمت تایپ و یا انتخاب کند.
Direction	جهت جستجو (به طرف بالا و یا پایین)
Option	• Match Case: جستجو برای کلماتی که شامل کلمه جستجو شده هستند. • Whole Words: جستجو برای کلماتی که دقیقا مطابق کلمه جستجو شده باشند (نه اینکه کلمه جستجو شده فقط قسمتی از آن‌ها باشد).
Scope	انتخاب محدوده جستجو

۱۲-۱-۶- جستجو در میان سیمبول ها

می‌توان از Find برای جستجو در جداول سیمبول نیز استفاده کرد. جستجو تنها در جدول صفحه ویرایشِ برنامه‌ی انتخاب شده امکان پذیر است و محدوده آن شامل کل پروژه نمی‌شود. برای جستجو کاربر باید در صفحه ویرایش برنامه، در قسمت سیمبول‌ها بر روی سطری که می‌خواهیم جستجو از آن شروع شود کلیک می‌کنیم. سپس با انتخاب  و یا انتخاب Find در سربرگ Edit می‌توان پنجره جستجو را باز کرد.



شکل ۱۲-۱۳ پنجره جستجوی سیمبول ها

در این پنجره می‌توان عبارتی را که می‌خواهیم آن را جستجو کنیم را وارد کرده و با کلیک بر روی Find Next و یا فشردن F3 آن را بیابیم. حتی اگر پنجره بسته شود با فشردن دوباره F3 می‌توان عملیات جستجو را ادامه داد. گزینه‌هایی که در صفحه این پنجره وجود دارد به صورت زیر است.

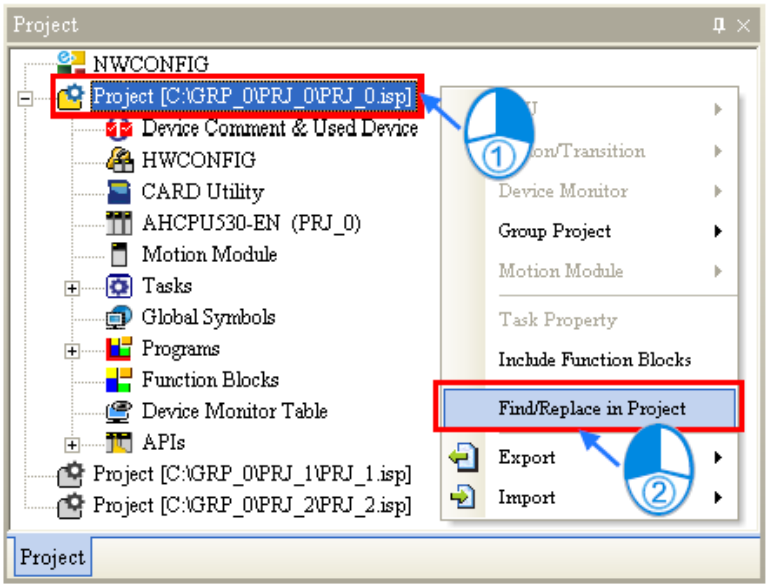
جدول ۱۲-۶: گزینه های **پنجره جستجوی سیمبول ها**

گزینه‌ها	شرح
Find	کاربر می‌تواند پارامتری را که می‌خواهد جستجو کند در این قسمت تایپ و یا انتخاب کند.
Direction	جهت جستجو (به طرف بالا و یا پایین)
Option	• Match Case: جستجو برای کلماتی که شامل کلمه جستجو شده هستند. • Whole Words: جستجو برای کلماتی که دقیقا مطابق کلمه جستجو شده باشند (نه اینکه کلمه جستجو شده فقط قسمتی از آن‌ها باشد).

۱۲–۱–۷– Find/Replace حافظه ها و سیمبول ها

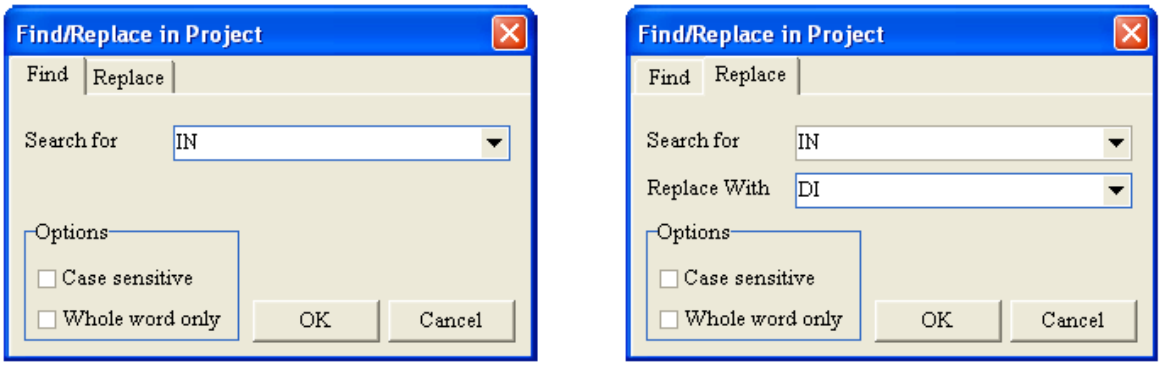
از عملکرد Find و Replace برای جستجو و جایگزینی حافظه‌ها و سیمبول‌ها در پروژه می‌توان استفاده کرد. در این بین جداول سیمبول‌های سراسری و محلی در POUها و محیط‌های ویرایش برنامه

مورد جستجو قرار می‌گیرند. در صورتی که پروژه گروهی باشد، عملکرد Find و Replace تنها برای پروژه فعال مورد استفاده قرار می‌گیرد. بر روی بخش مدیریت پروژه کلیک راست کرده و Find/Replace in Project را انتخاب کنید.



شکل ۱۲-۱۴ جستجو و جایگزینی حافظه ها و سیمبول ها –قسمت ۱

در پنجره باز شده می‌توان گزینه مناسب برای جستجو و یا جایگزینی را در سربرگ‌های مربوطه وارد کرده و با کلیک بر روی OK نتیجه اعمال شده بر روی کل پروژه را مشاهده کرد. توجه شود که در صورت جایگزینی امکان بازگشت تغییرات وجود نخواهد داشت.



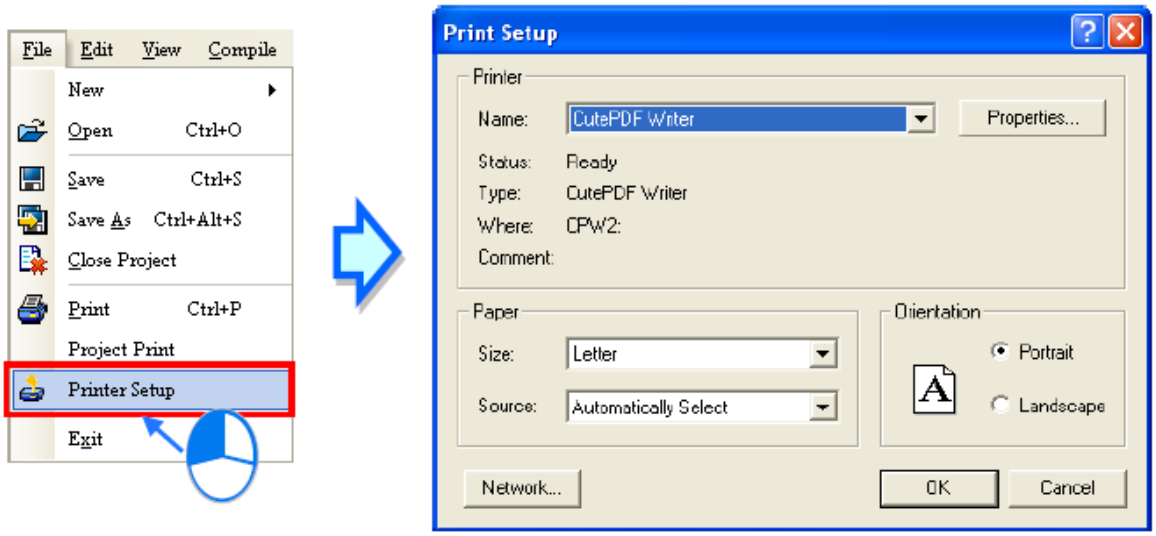
شکل ۱۲-۱۵ جستجو و جایگزینی حافظه ها و سیمبول ها –قسمت ۲

پس از تایید، نتایجِ جستجو در بخش پیام نمایش داده می‌شود. در صورت دوبار کلیک کردن بر روی هر کدام از نتایج، سیستم پنجره مربوط به آن را باز خواهد کرد.

۱۲-۱-۸- چاپ کردن پروژه

۱۲-۱-۸-۱- تنظیمات

در سربرگ File، گزینه Printer Setup را انتخاب کرده تا بتوان تنظیمات مرتبط با آن را انجام داد.

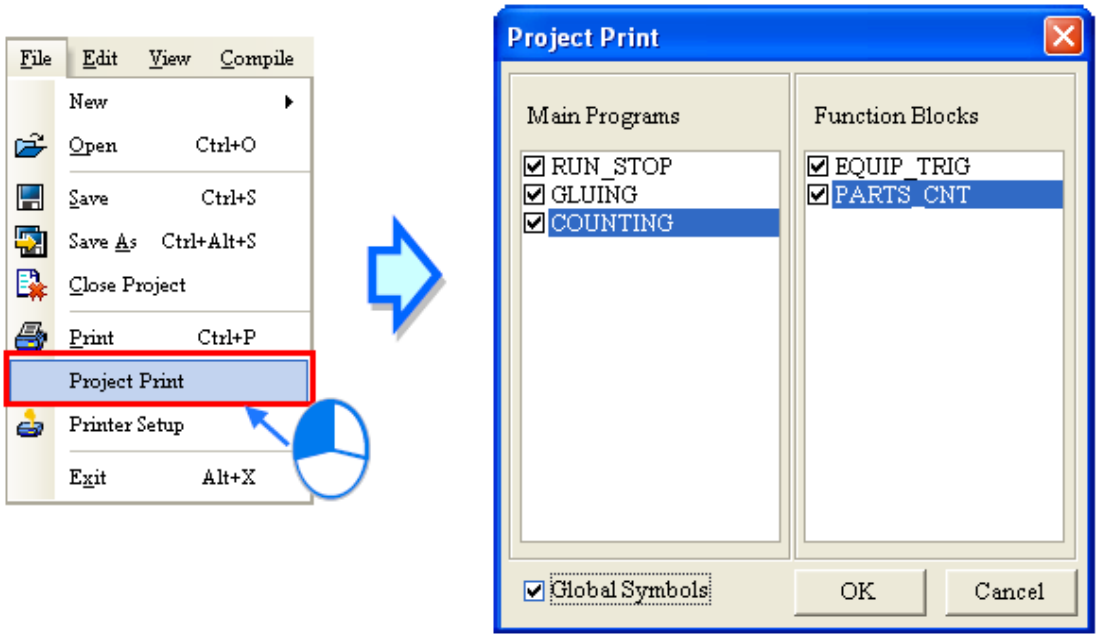


شکل ۱۲-۱۶ تنظیمات پرینتر

در این صفحه می‌توان نوع پرینتر (حتی می‌توانیم به جای پرینتر واقعی از پرینترهای مجازی برای ساخت فایل‌های PDF استفاده کنیم) را انتخاب کرده و تنظیمات مربوط به اندازه و فرمت کاغذ را انجام داد.

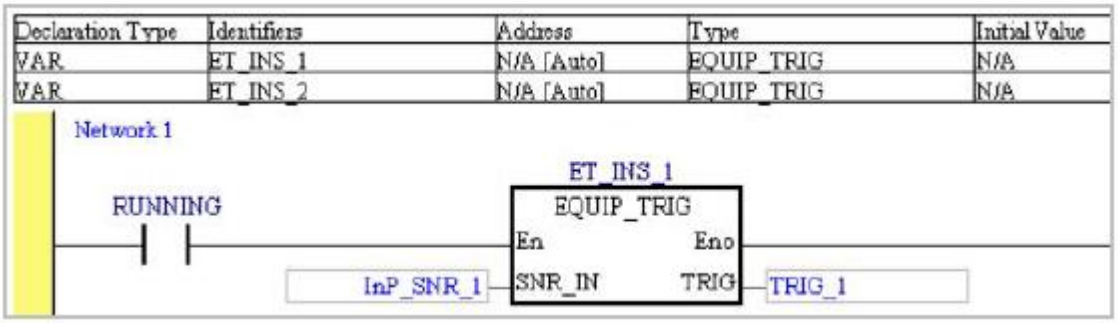
۱۲-۱-۸-۲- پرینت پروژه

پس از کلیک بر روی Project Print در سربرگ File، می‌تواند برنامه‌های اصلی، بلوک‌ها و سیمبول‌های سراسری را برای پرینت انتخاب کند.



شکل ۱۲-۱۷ پرینت پروژه

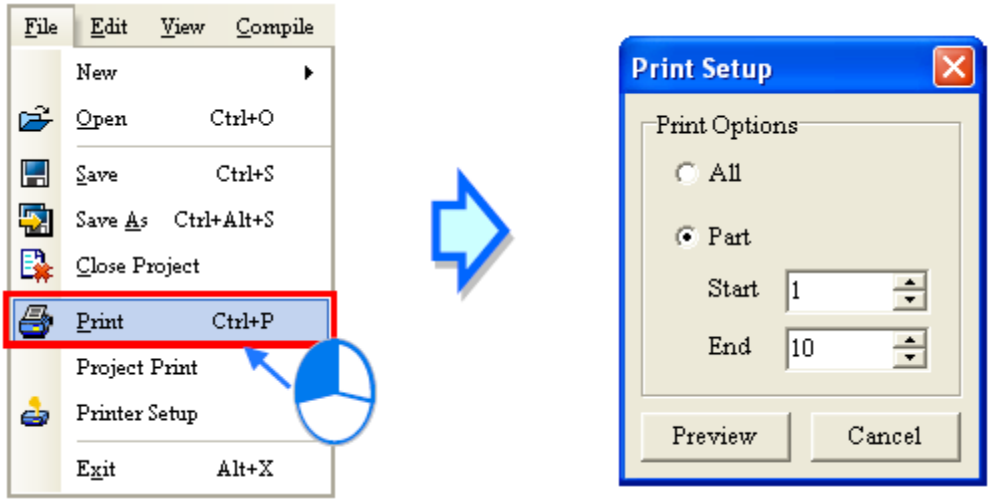
نمونه‌ای از نتیجه پرینت بخش برنامه در زیر آمده است، این برنامه به همراه سیمبول‌های محلی آن در پرینت نشان داده می‌شود. در صورتی که توضیحاتی نیز در برنامه وجود داشته باشد پرینت گرفته می‌شود.



شکل ۱۲-۱۸ نمونه ای از پرینت پروژه

۱۲-۱-۸-۳- پرینت صفحه جاری

در صورتی که بخواهیم صفحه جاری را پرینت بگیریم، می‌توانیم از منوی File گزینه Print را انتخاب و یا بر روی  کلیک کنیم. در پنجره باز شده می‌توان تمام و یا بخشی از پروژه جاری را برای پرینت انتخاب کرد.



شکل ۱۲-۱۹ پرینت صفحه جاری

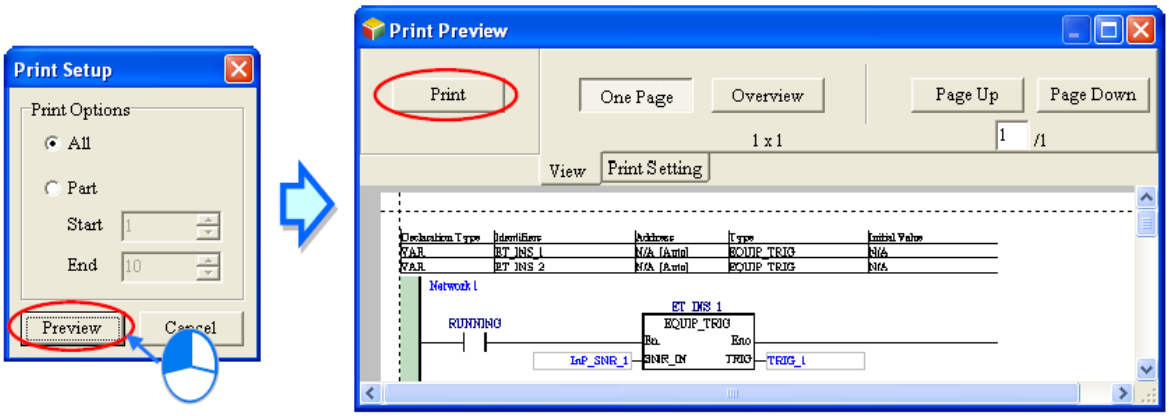
نتایج پرینت با توجه به محتوای جاری متفاوت است، و کاربر باید با توجه به نوع نیاز خود به اطلاعات به پنجره مورد نظر رفته و اطلاعات مورد نیاز را برای پرینت مشخص نماید برای اطلاعات بیشتر به جدول زیر مراجعه کنید.

جدول ۱۲-۷: محتوای پرینت

پنجره	همه (کل صفحه جاری)	بخشی
جدول سیمبول‌های سراسری	تمامی جدول سیمبول‌ها پرینت گرفته می‌شود.	کاربر نمی‌تواند بخشی از جدول سیمبول‌ها را پرینت بگیرد.
LD و FBD	تمام برنامه و جدول سیمبول محلی آن پرینت گرفته می‌شود.	تمام جدول سیمبول‌های محلی و بخشی از برنامه شامل Network با شماره Start تا Network با شماره End پرینت گرفته می‌شود.
IL و ST	تمام برنامه و جدول سیمبول محلی آن پرینت گرفته می‌شود.	تمام جدول سیمبول‌های محلی و بخشی از برنامه شامل سطر با شماره Start تا سطر با شماره End پرینت گرفته می‌شود.
SFC	تمام چارت‌ها و جدول سیمبول محلی آن در برنامه پرینت گرفته	کاربر نمی‌تواند بخشی از SFC را پرینت بگیرد.

	می‌شود (ولی Action و Transitionها پرینت گرفته نمی‌شود با این حال کاربر می‌تواند با باز کردن صفحه مرتبط با آن‌ها برنامه‌های آن‌ها را پرینت بگیرد).	

در نهایت کاربر با کلیک بر روی Preview در پنجره Print Setup می‌تواند نمایشی از آنچه قرار است پرینت گرفته شود را قبل از پرینت ببیند و سپس با کلیک بر روی Print اقدام به پرینت کند.

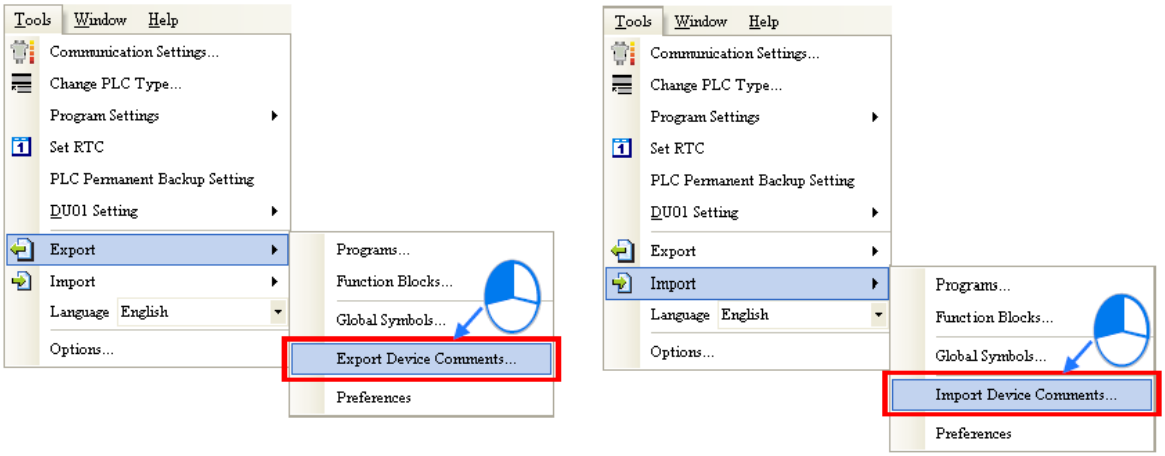


شکل ۱۲-۲۰ پیش نمایش قبل از پرینت

۱۲-۲-مدیریت حافظه ها

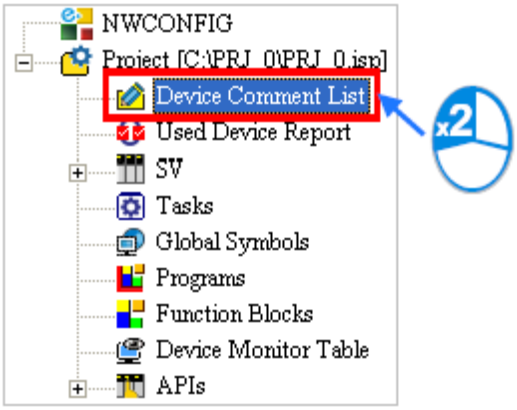
۱۲-۲-۱- لیست توضیحات حافظه ها برای PLCهای سری DVP

برای ذخیره سازی و استفاده مجدد از کامنت حافظه‌ها در پروژه‌های ISPSoft می‌توان از سربرگ File، به ترتیب در قسمت Export، گزینه Export Device Comments و در قسمت Import، گزینه Import Device Comment را انتخاب کرد.



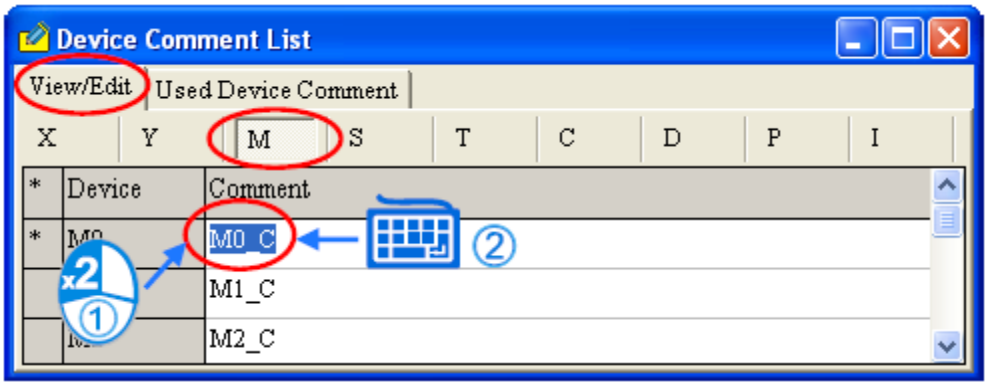
شکل ۱۲-۲۱ ذخیره سازی و استفاده مجدد از کامنت حافظه ها در PLCهای DVP

برای مدیریت توضیحات حافظه‌ها هم می‌توان در بخش مدیریت پروژه دوبار بر روی Device Comment List کلیک کرد تا پنجره مرتبط با آن باز شود.



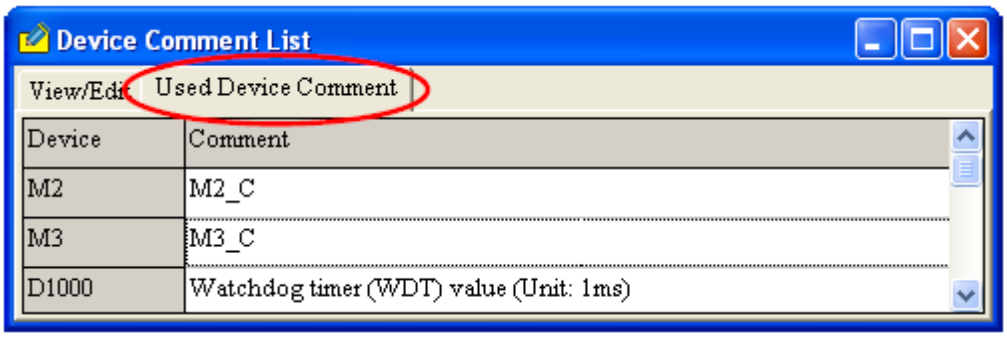
شکل ۱۲-۲۲ ویرایش کامنت حافظه ها در PLCهای DVP

در سربرگ View/Edit می‌توان توضیحات مربوط به حافظه‌ها را دید. در این سربرگ پس از انتخاب نوع حافظه، توضیحات مربوط به آن نمایش داده می‌شود. علامت ستاره "*" در سمت چپ حافظه نشانگر آن است که حافظه در برنامه مورد استفاده قرار گرفته است. برای ایجاد و یا تغییر توضیحات می‌توان در ستون Comment مرتبط با هر حافظه دوبار کلیک کرده و متن مورد نظر را تایپ کنیم.



شکل ۱۲-۲۳ صفحه ویرایش توضیحات حافظه ها

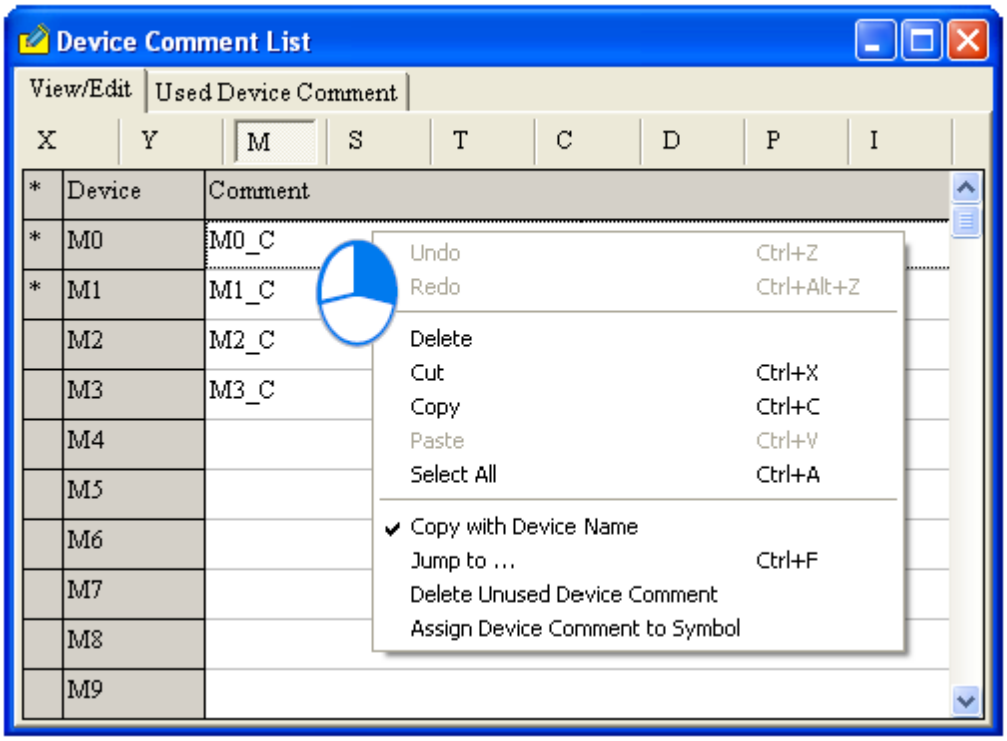
در سربرگ Used Device Comment نیز می‌توان حافظه‌های دارای توضیحات را یکجا مشاهده کرد، در اینجا نیز برای ایجاد و یا تغییر توضیحات می‌توان در ستون Comment مرتبط با هر حافظه دوبار کلیک کرده و متن مورد نظر را تایپ کنیم.



شکل ۱۲-۲۴ مشاهده حافظه های دارای توضیحات

توجه شود که توضیحات حافظه‌ها با توضیحات سیمبول‌ها متفاوت است و توضیحات سیمبول‌ها در لیست توضیحات حافظه‌ها نمی‌آید.

در صورتی که در پنجره Device Comment List کلیک راست کنیم، با گزینه‌های زیر روبرو می‌شویم.



شکل ۱۲-۲۵ گزینه های موجود در جدول توضیحات حافظه ها

- Jump to

پس از کلیک بر روی Jump to، پنجره مرتبط با آن فعال می‌شود. در این پنجره می‌توانیم نام حافظه مورد نظر را تایپ و تایید کرده تا آن حافظه به همراه توضیحاتش برای ما نمایش داده شود. توجه شود که در این بخش، نوع حافظه‌ای که تایپ می‌شود باید مشابه حافظه‌ای باشد که بر روی آن کلیک راست کرده‌ایم در غیر این صورت نتیجه‌ای برای پرش نخواهیم داشت.

در صورتی که Jump to در صفحه مربوط به Used Device Comment باز شده باشد، حافظه تایپ شده در آن نیز باید در این صفحه قرار داشته باشد.

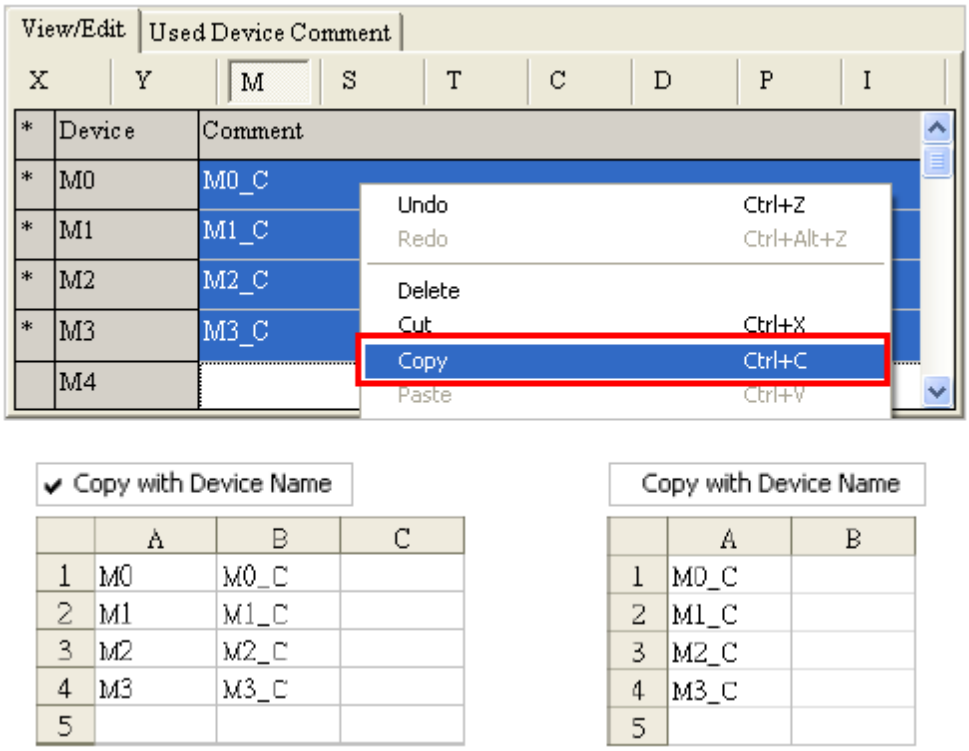
شکل ۱۲-۲۶ جستجو (پرش) به حافظه مورد نظر در جدول توضیحات حافظه ها

- Delete Unused Device Comment

با کلیک بر روی این گزینه، توضیحات حافظه‌هایی که در برنامه استفاده نشده است حذف خواهد شد.

- Copy with Device Name

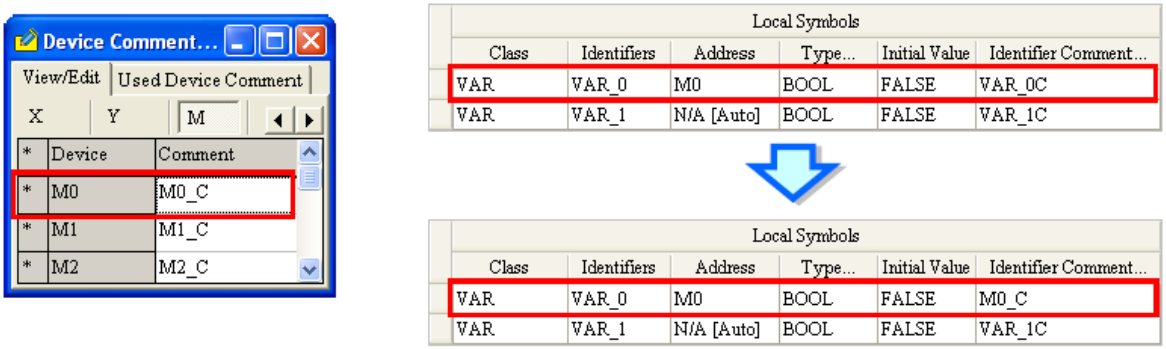
در صورت فعال کردن این گزینه، می‌توان کامنت‌ها را به همراه نام حافظه‌ها کپی کرد (می‌توان آن‌ها را کپی و برای بررسی و مرور در مایکروسافت اکسل جایگذاری کرد).



شکل ۱۲-۲۷ کپی توضیحات به همراه نام حافظه ها

- Assign Device Comment to Symbol

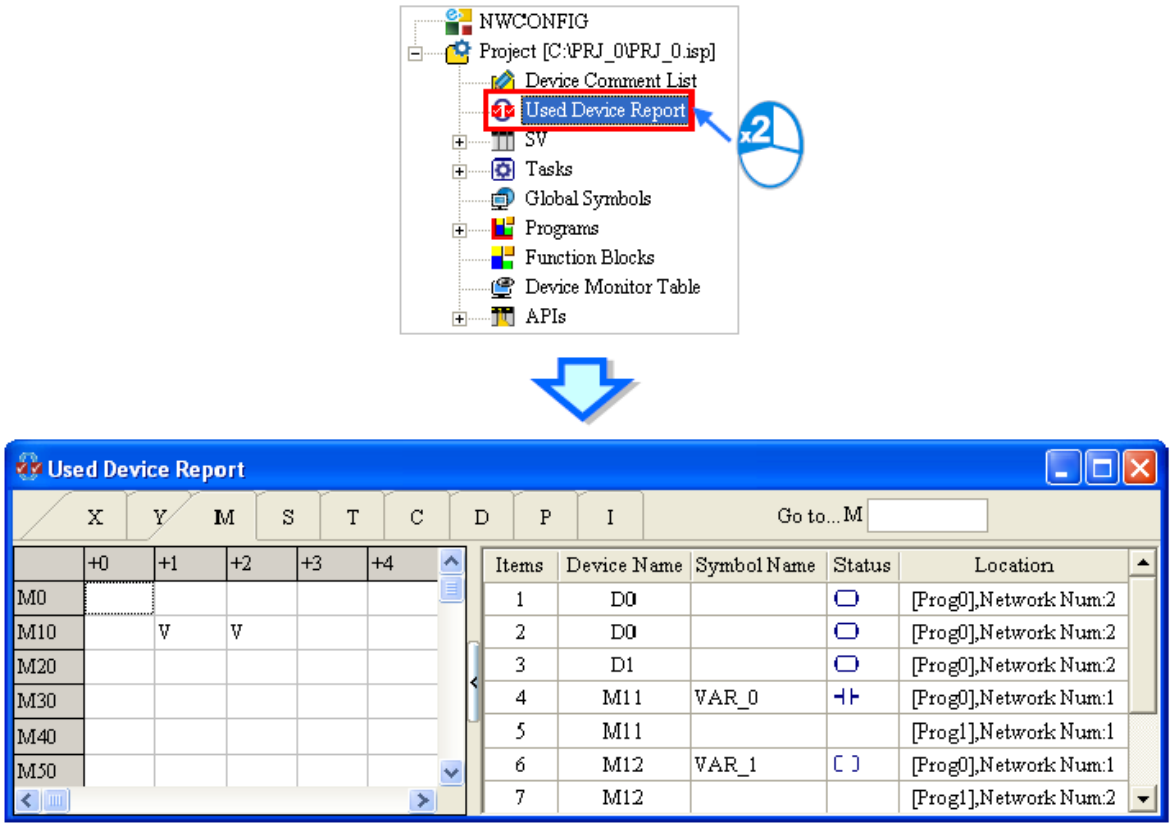
در صورتی که برای بعضی از حافظه‌ها سیمبول تخصیص داده شده باشد، پس از کلیک بر روی این گزینه، توضیحات به سیمبول‌ها نیز منتقل می‌شود.



شکل ۱۲-۲۸ انتقال توضیحات حافظه ها به سیمبول‌های آن ها

۱۲-۲-۲- گزارش حافظه های استفاده شده در PLCهای مدل DVP

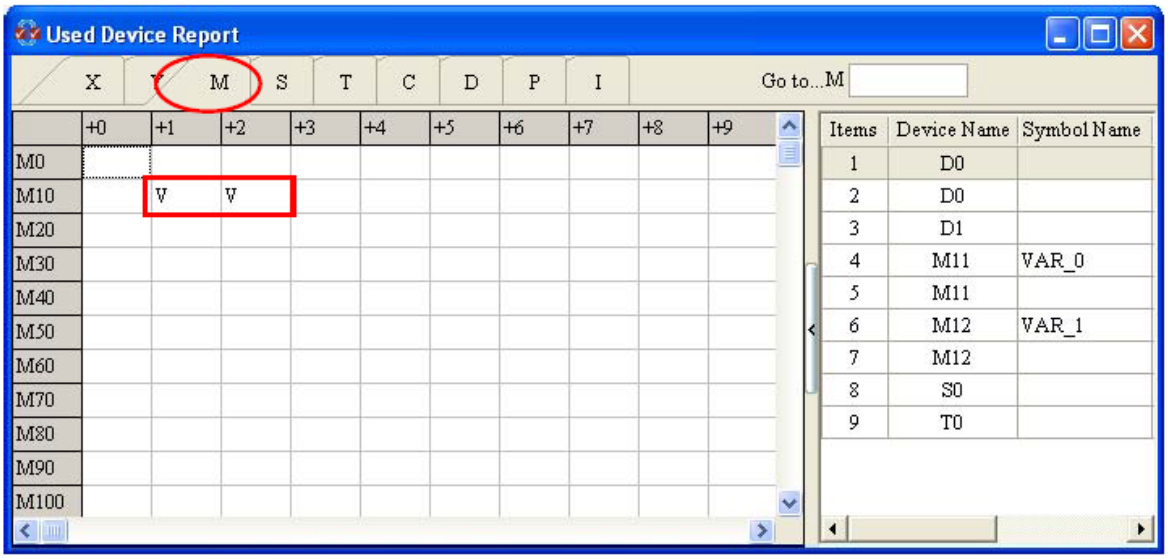
برای اینکه کاربران بتوانند گزارشی از حافظه‌های مورد استفاده در برنامه داشته باشند، نیاز است ابتدا برنامه خود را کامپایل کند. سپس با دوبار کلیک بر روی Used Device Report در قسمت مدیریت پروژه، می‌تواند پنجره حاوی گزارش حافظه‌های استفاده شده را مشاهده کند.



شکل ۱۲-۲۹ گزارش حافظه های استفاده شده در برنامه

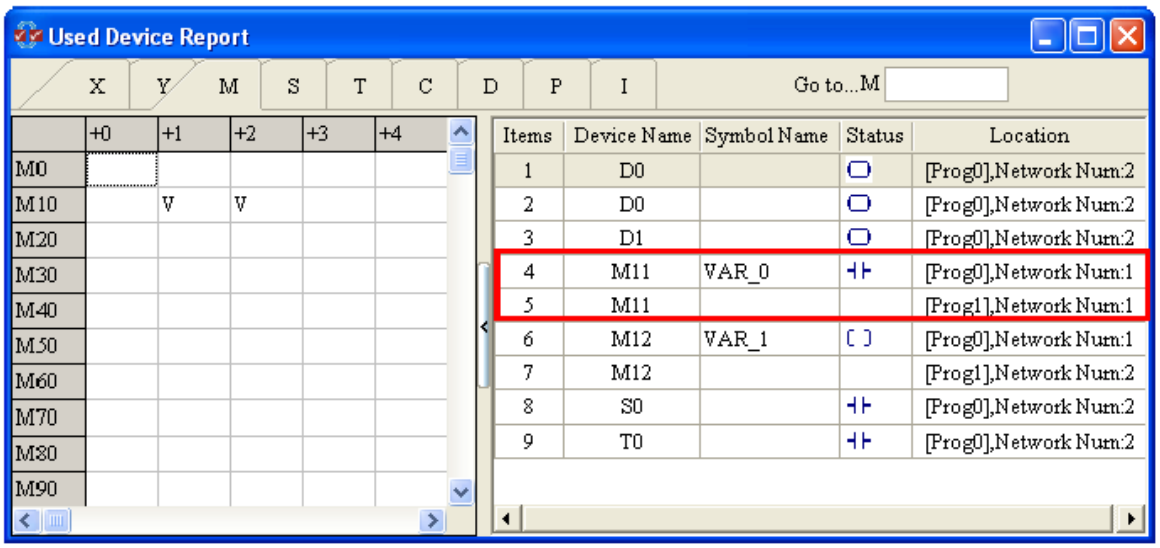
در صورتی که برنامه تغییری کند، برای به روز شدن این گزارش لازم است پنجره آن را بسته، برنامه را دوباره کامپایل کرده و مجدد گزارش را باز کنیم. این پنجره دارای دو بخش است، جدول حافظه‌ها در

سمت چپ که در صورتی که حافظه‌ای در برنامه مورد استفاده قرار گرفته باشد سلول مرتبط با آن تیک می‌خورد، به عنوان نمونه در مثال زیر تیک حافظه‌های M11 و M12 نشان می‌دهد که آن‌ها در برنامه مورد استفاده قرار گرفته‌اند.



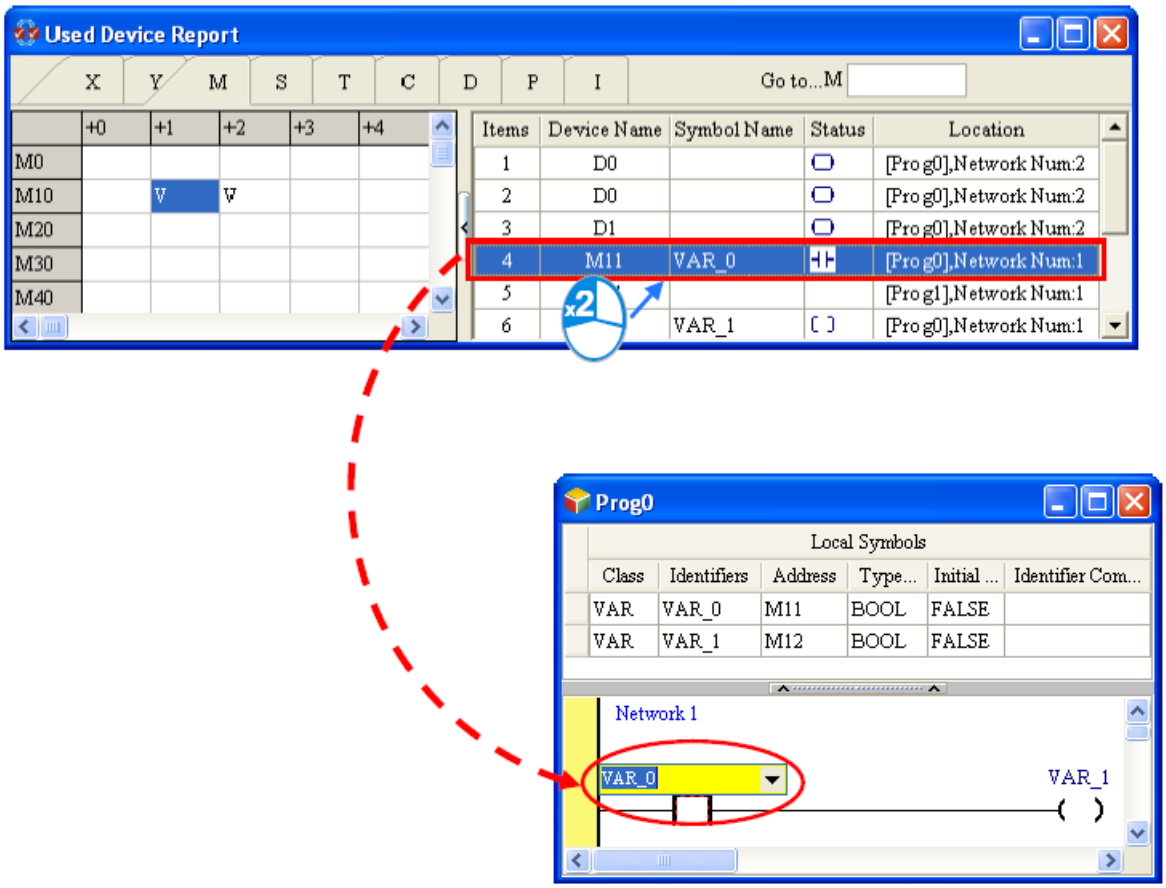
شکل ۱۲-۳۰ گزارش حافظه های استفاده شده در برنامه- شرح بخش چپ

در سمت راست این پنجره هم لیست حافظه‌های استفاده شده در برنامه و آدرس محلی که مورد استفاده قرار گرفته‌اند آمده است، حافظه‌ها در این قسمت طبقه بندی نشده‌اند. اگر حافظه‌ای در چند جای برنامه مورد استفاده قرار گرفته باشد، اطلاعات در مورد این موقعیت‌های متفاوت در سطرهای مختلف لیست می‌شود. برای مثال در شکل زیر، M11 در دوجای برنامه مورد استفاده قرار گرفته است.



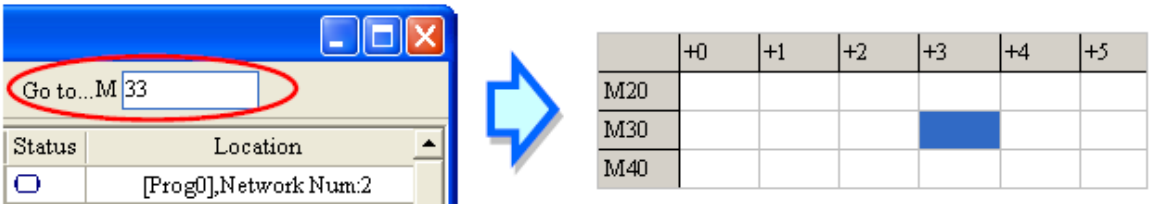
شکل ۱۲-۳۱ گزارش حافظه های استفاده شده در برنامه- شرح بخش راست

در صورتی که کاربر بر روی هرکدام از حافظه‌های استفاده شده در برنامه که در لیست سمت راست آمده‌اند دوبار کلیک کند، بخشی از برنامه که آن حافظه مورد استفاده قرار گرفته است نشان داده خواهد شد.



شکل ۱۲-۳۲ رجوع به برنامه ای که حافظه در آن به کار رفته است از گزارش حافظه های استفاده شده

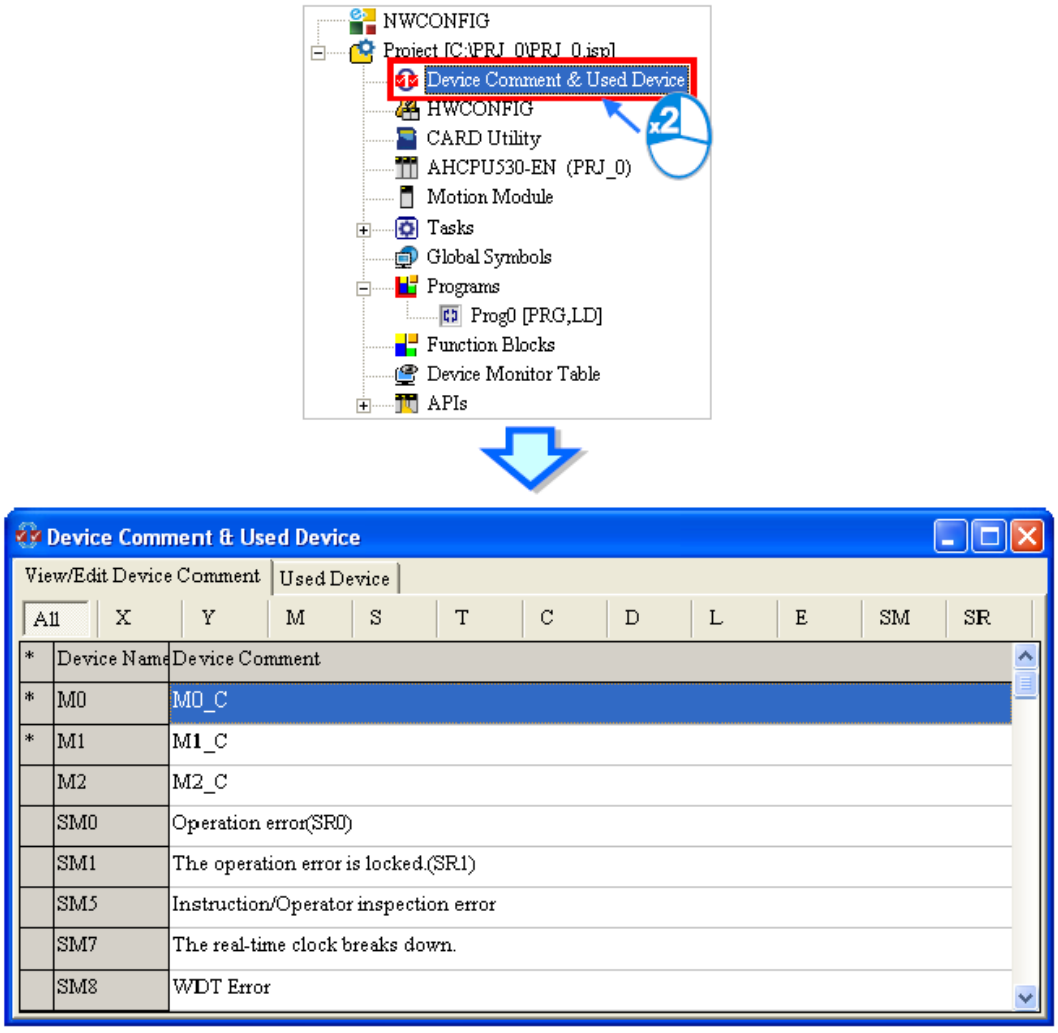
با نوشتن آدرس حافظه در قسمت Go to، وضعیت حافظه مورد نظر نشان داده خواهد شد.



شکل ۱۲-۳۳ پرش به حافظه مورد نظر در گزارش حافظه های استفاده شده در برنامه

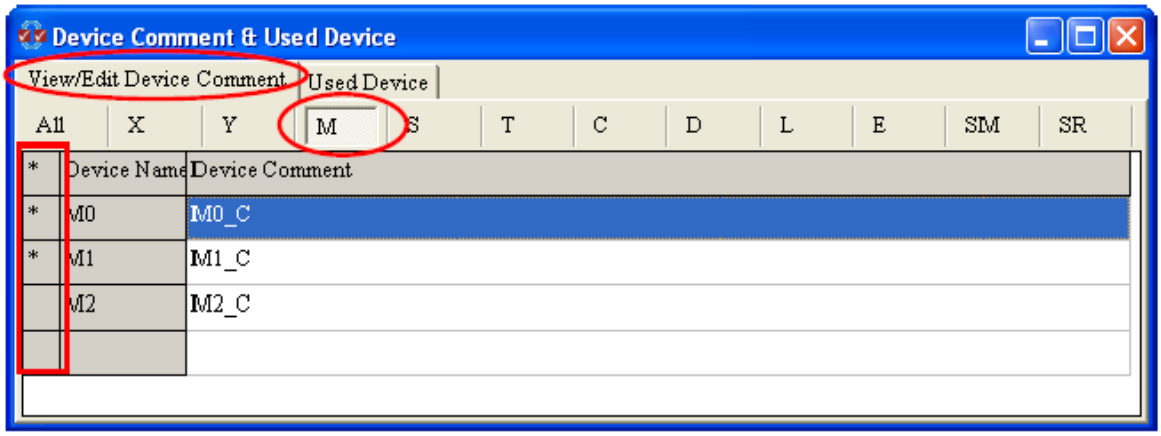
۱۲-۲-۳- مدیریت حافظه ها در PLCهای سری AH500

برای مدیریت حافظه‌ها در PLCهای سری AH500 می‌توان در بخش مدیریت پروژه دوبار بر روی Device Comment & Used Device کلیک کرد تا پنجره مرتبط با آن باز شود. در صورتی که برنامه تغییری کند، برای به روز شدن اطلاعات این پنجره لازم است آن را بسته، برنامه را دوباره کامپایل کرده و دوبار پنجره تنظیمات حافظه را باز کنیم.



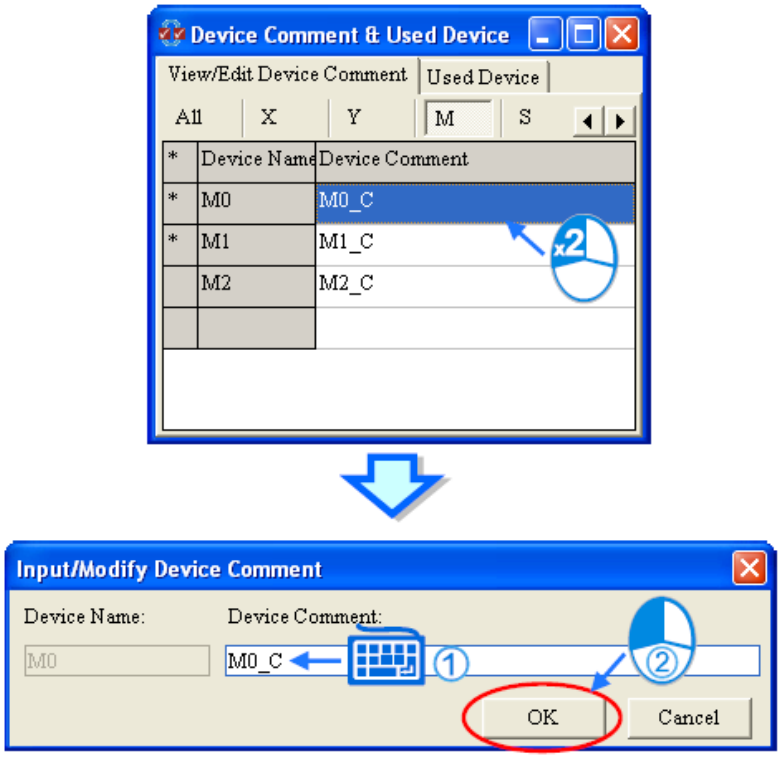
شکل ۱۲-۳۴ پنجره مدیریت حافظه ها در PLCهای سری AH500

در این پنجره دو سربرگ وجود دارد، سربرگ View/Edit Device Comment شامل توضیحات مربوط به حافظه‌ها است. در این سربرگ می‌توان نوع حافظه را تعیین کرد، همچنین علامت ستاره در سمت چپ حافظه نشانگر استفاده از آن حافظه در برنامه است.



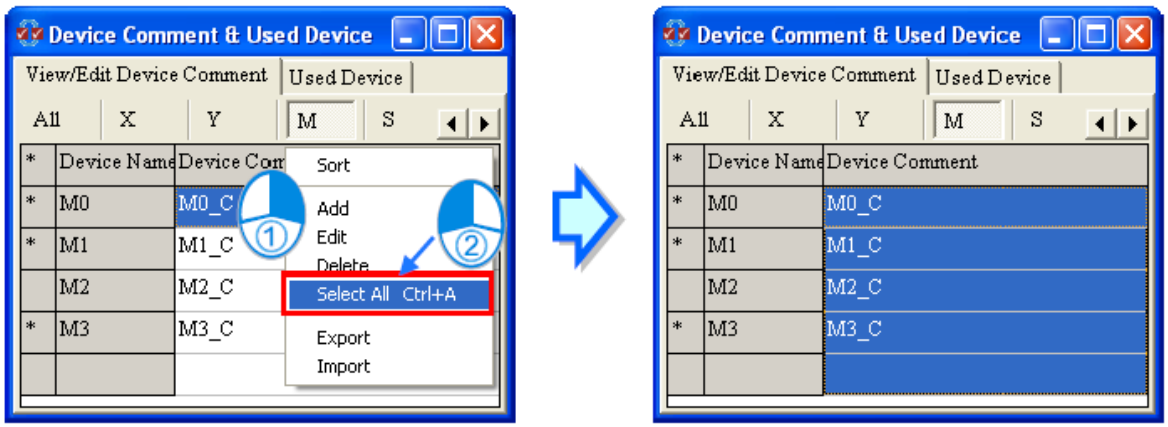
شکل ۱۲-۳۵ پنجره مدیریت حافظه ها در PLCهای سری AH500 سربرگ View/Edit Device Comment

برای اضافه کردن یا تغییر توضیحات حافظه می‌توان بر روی آن دوبار کلیک کرده و در پنجره باز شده توضیحات خود را برای حافظه مورد نظر وارد کنیم.

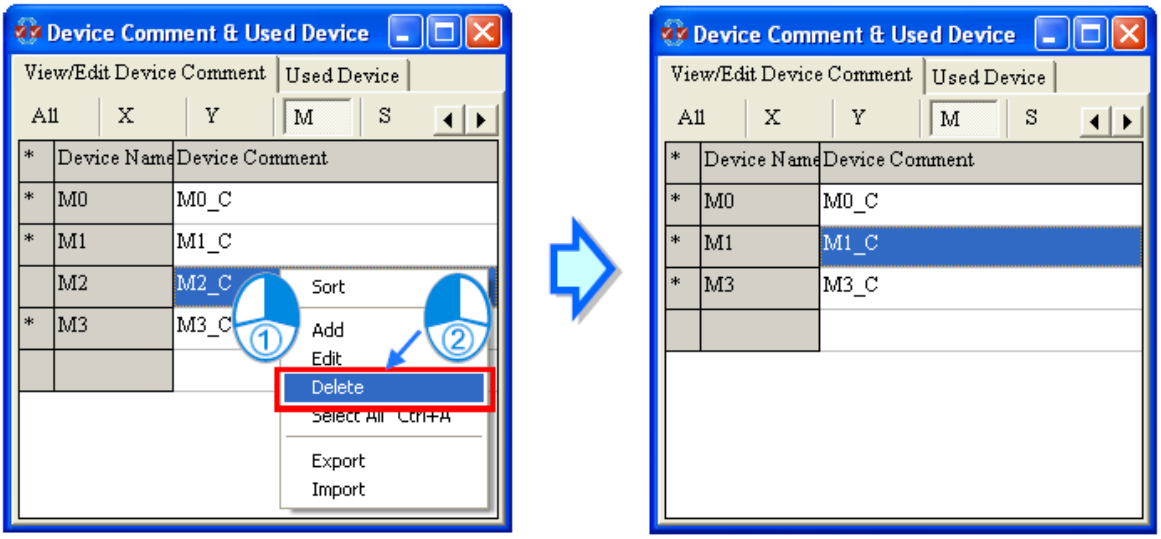


شکل ۱۲-۳۶ تغییر توضیحات حافظه در پنجره مدیریت حافظه ها در PLCهای سری AH500

برای حذف کردن توضیحات نیز می‌توان بر روی سطر حافظه مورد نظر در لیست (یا می‌توان مجموعه‌ای از سطرها را ابتدا انتخاب کرده) کلیک راست کرده و سپس Delete را انتخاب کرد. برای انتخاب تمامی سطرها می‌توان کلیک راست کرده و Select All را برگزید.

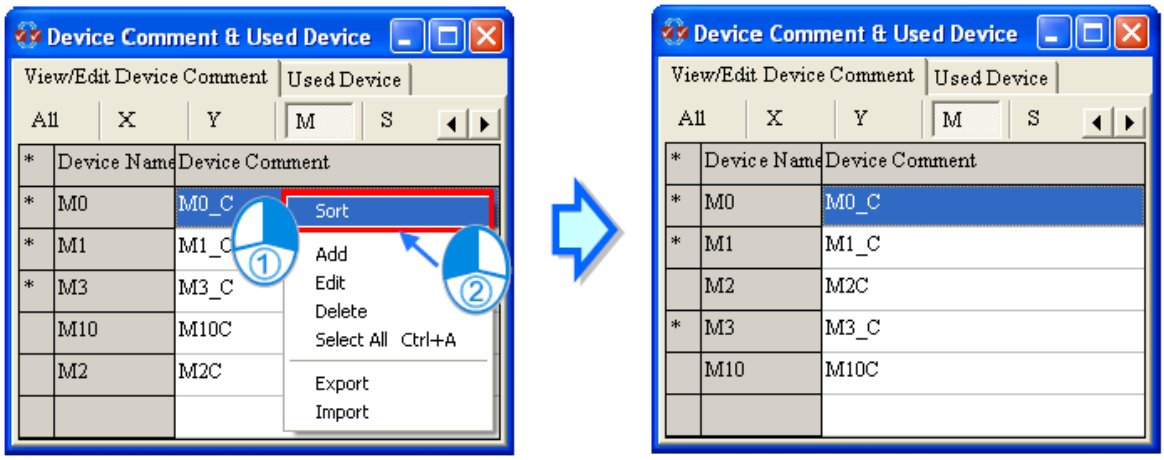


شکل ۱۲-۳۷ انتخاب تمامی خانه های پنجره مدیریت حافظه ها در PLCهای سری AH500



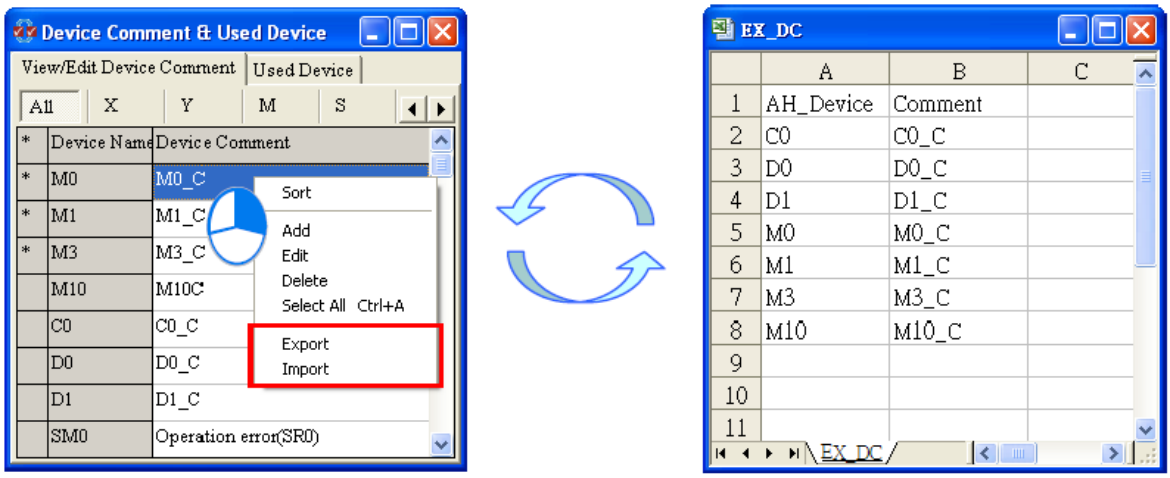
شکل ۱۲-۳۸ شکل ۱۲-۳۹ حذف خانه های پنجره مدیریت حافظه ها در PLCهای سری AH500

برای مرتب کردن لیست بر اساس ترتیب حافظه‌ها می‌توان از گزینه Sort استفاده کرد.



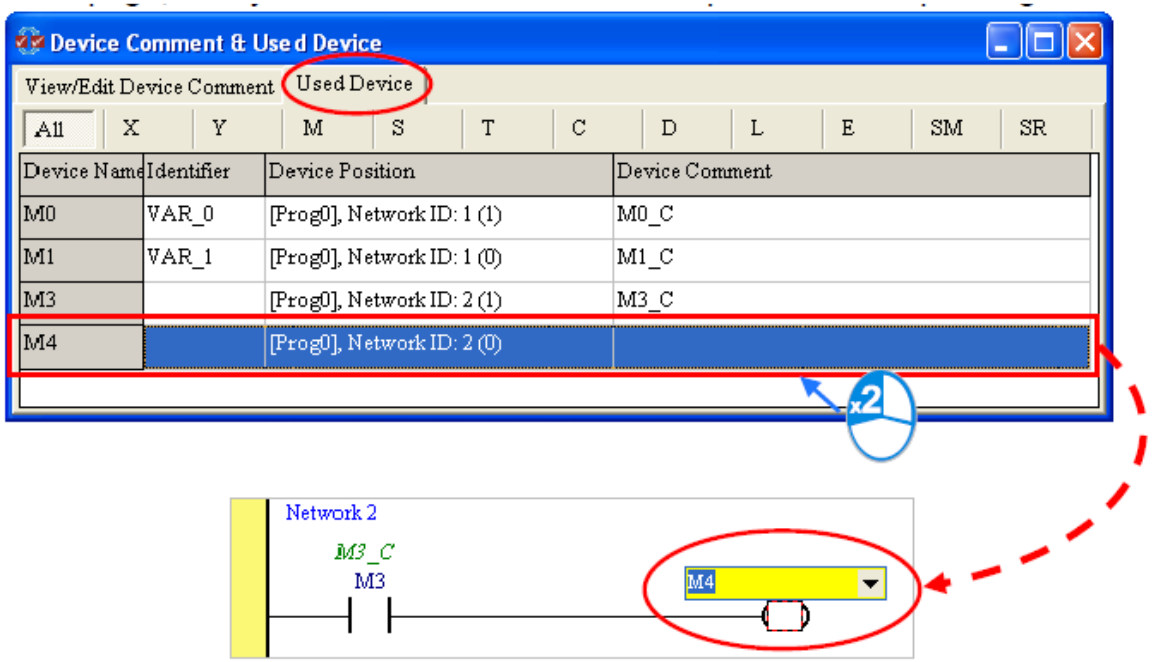
شکل ۱۲-۴۰ تنظیم خانه های پنجره مدیریت حافظه ها در PLCهای سری AH500

می‌توان لیست توضیحات را به صورت فایل CSV که در نرم افزار اِکسل قابل باز کردن است، ذخیره و دوباره فراخوانی کرد. برای این کار می‌توان از Import برای ذخیره لیست و Export برای فراخوانی آن استفاده کرد.



شکل ۱۲-۴۱ ذخیره و فراخوانی خانه های پنجره مدیریت حافظه ها در PLCهای سری AH500

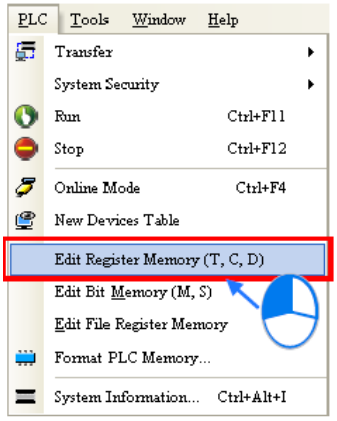
در سربرگ Used Device نیز اطلاعات مربوط به حافظه‌های استفاده شده در برنامه آمده است که با دو بار کلیک بر روی آن‌ها قسمتی از برنامه که حافظه مورد نظر در آن به کار رفته است باز خواهد شد.



شکل ۱۲-۴۲ پنجره مدیریت حافظه ها در PLCهای سری AH500 سربرگ Used Device

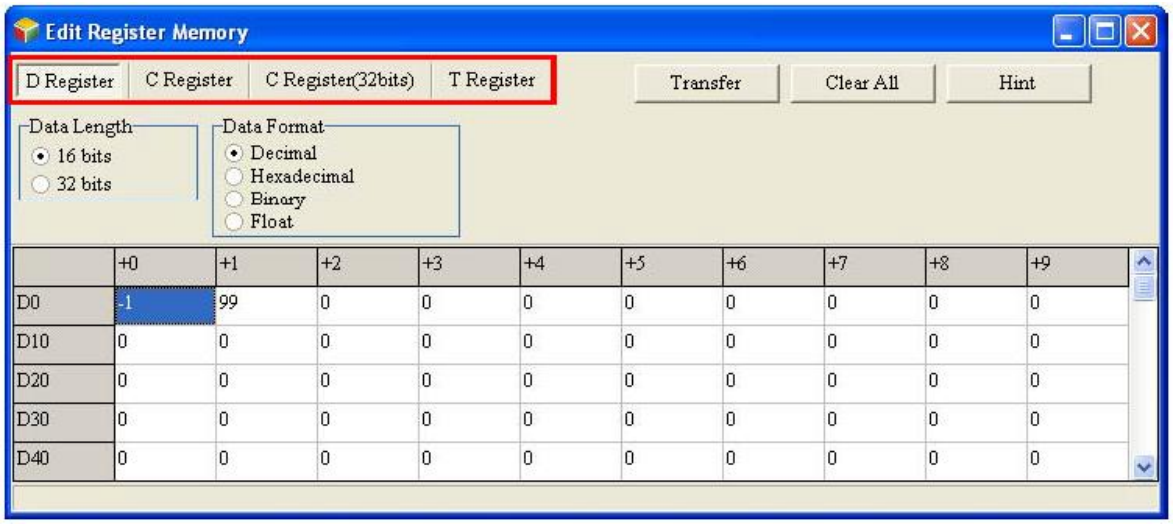
۱۲-۲-۴- اصلاح مقادیر حافظه های T/C/D در PLCهای سری DVP

کاربران می‌توانند مقادیر حافظه‌های T/C/D را در PLCهای سری DVP تغییر داده و مقادیر جدید را ذخیره و بارگزاری کنند. برای اینکار می‌توان از سربرگ PLC گزینه Edit Register Memory (T,C,D) را انتخاب کرد.



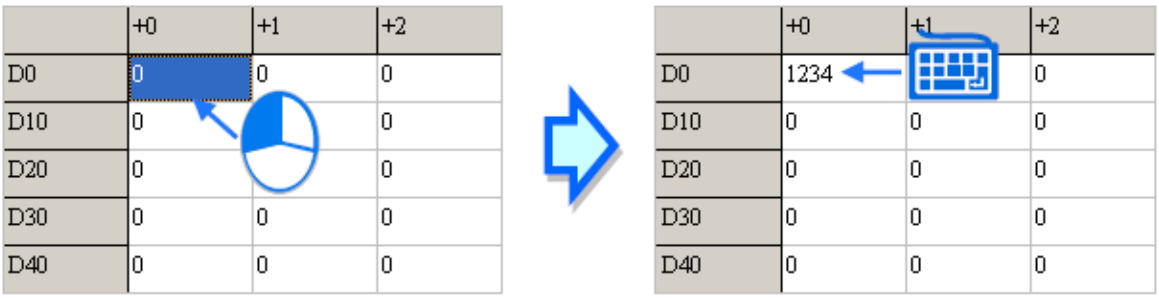
شکل ۱۲-۴۳ انتخاب Edit Register Memory (T,C,D)

مقادیر موجود در جدول باز شده مقادیر استخراج شده از PLC نیستند بلکه مقادیر ذخیره شده از دفعه قبل می‌باشند، در صورتی که این پنجره برای اولین بار باز شده باشد، تمامی مقادیر صفر هستند. با استفاده از سربرگ‌ها می‌توان نوع حافظه را انتخاب کرد.



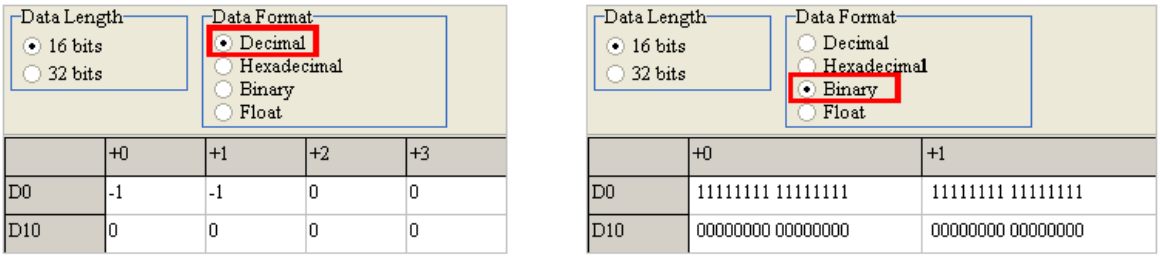
شکل ۱۲-۴۴ پنجره اصلاح مقادیر حافظه های T/C/D در PLCهای سری DVP

برای تغییر مقادیر حافظه‌ها می‌توان بر روی خانه جدول مرتبط با آن‌ها کلیک و مقدار مطلوب را تایپ کرد.



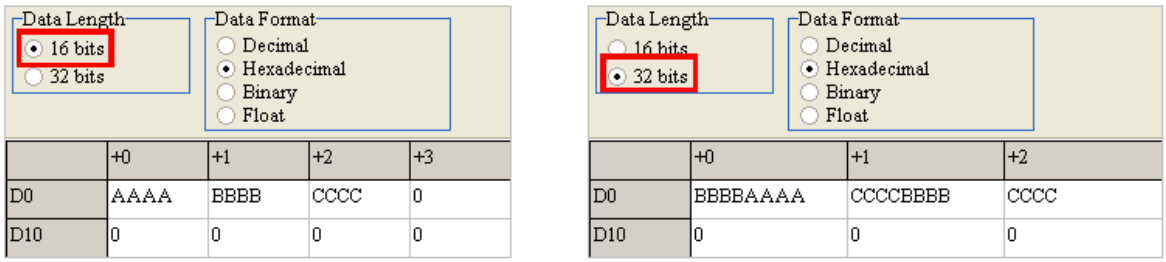
شکل ۱۲-۴۵ وارد کردن مقادیر حافظه های T/C/D در PLCهای سری DVP

می‌توان نوع نمایش اعداد حافظه‌ها را در قسمت Data Format تعیین کرد.



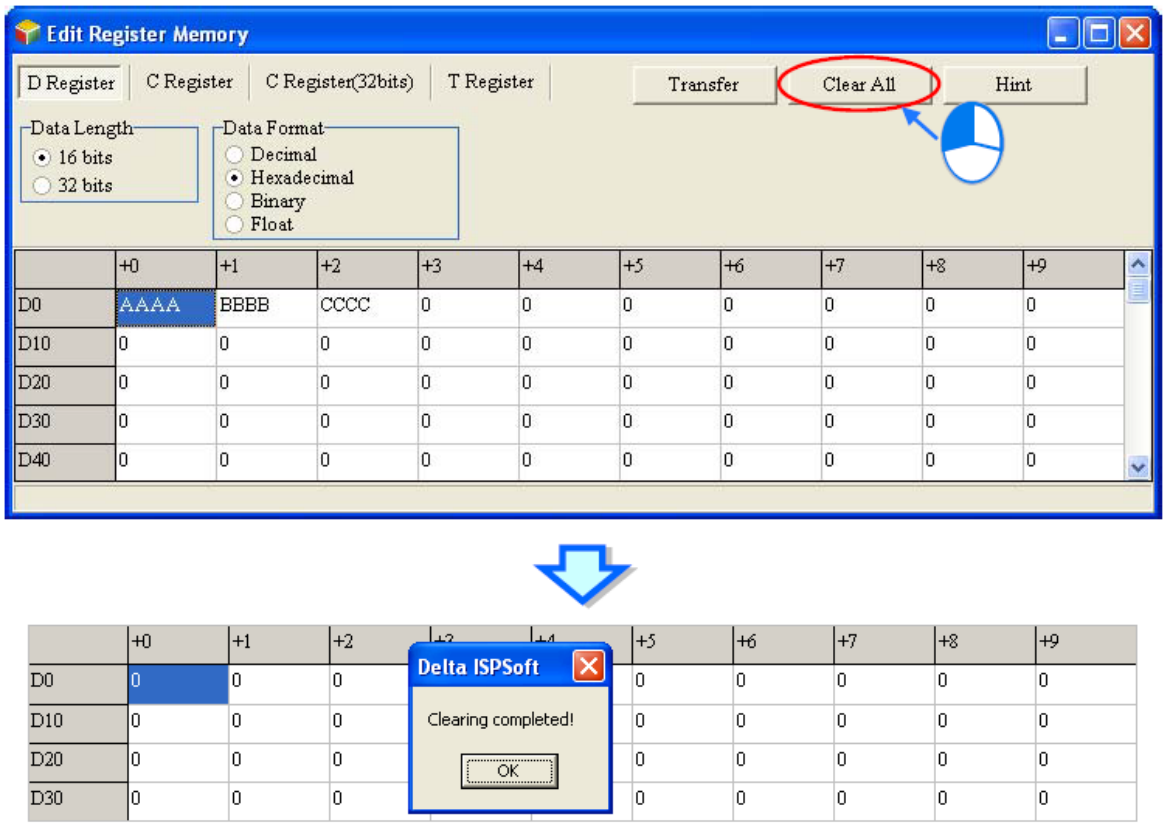
شکل ۱۲-۴۶ نوع نمایش اعداد در جدول مقادیر حافظه های T/C/D

کاربر برای نمایش داده‌ها می‌تواند فرمت ۱۶ و یا ۳۲ بیتی را در قسمت Data Length انتخاب کند، در صورتی که حافظه ۳۲ بیتی انتخاب شود در واقع داده هر خانه جدول نمایانگر مقدار دو حافظه ۱۶ بیتی خواهد بود (هر حافظه در برگیرنده حافظه ۱۶ بیتی بعد از خود نیز هست). برای نمونه در مثال زیر مقدار هگزادسیمال D0 برابر AAAA، مقدار هگزادسیمال D1 برابر BBBB و مقدار هگزادسیمال D2 برابر CCCC است. با تغییر فرمت طول داده‌ها به ۳۲ بیت، مقدار حافظه خانه D0 شامل D1، مقدار حافظه خانه D1 شامل D2 و مقدار حافظه خانه D2 شامل حافظه D3 نیز خواهد بود. به عبارت دیگر مقدار حافظه D0 به صورت BBBBAAAA، مقدار حافظه D1 به صورت CCCCBBBB و مقدار حافظه D2 به صورت CCCC در خواهد آمد. (در ISPSoft حافظه با شماره بزرگتر ارزش عددی بیشتری دارند.)



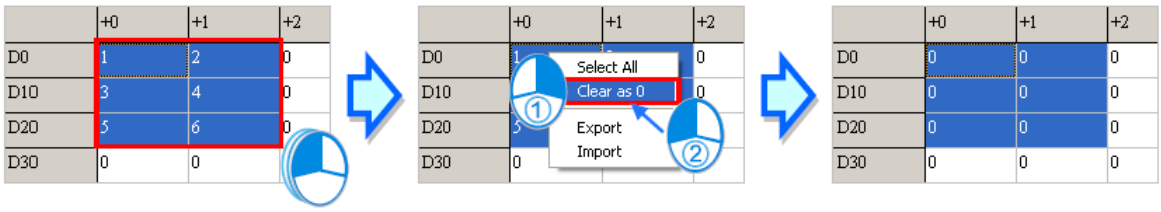
شکل ۱۲-۴۷ نمایش اعداد در جدول مقادیر حافظه های T/C/D با طول های مختلف

در هر سربرگ می‌توان با کلیک بر روی Clear All تمامی حافظه‌های همان سربرگ را صفر کرد.



شکل ۱۲-۴۸ پاک کردن داده های جدول مقادیر حافظه های T/C/D

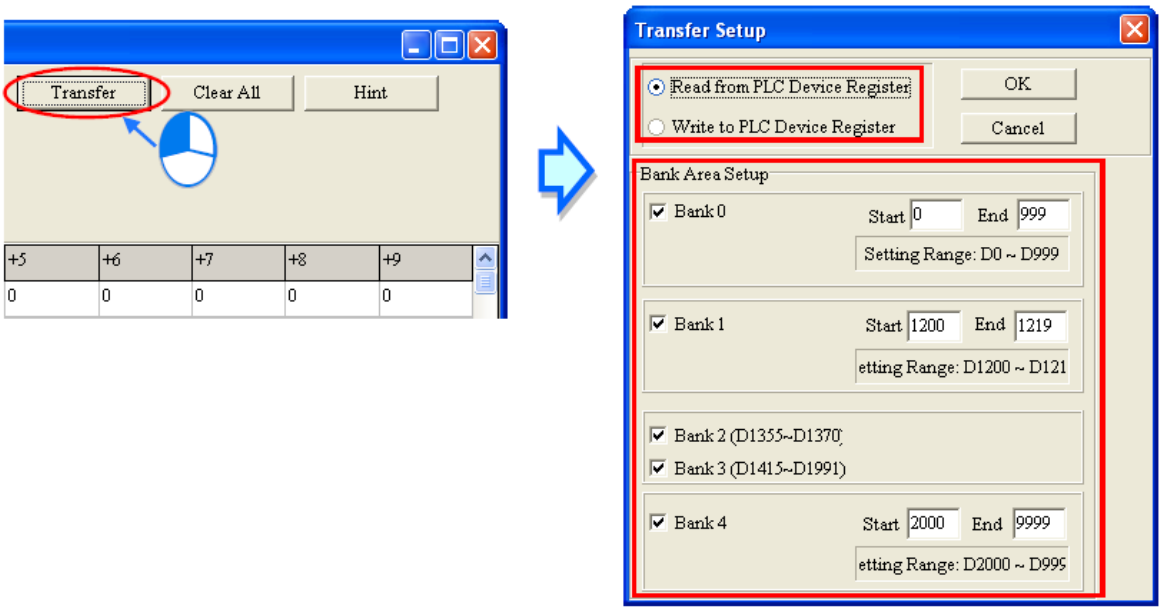
همچنین با کشیدن موس می‌توان مجموعه‌ای از خانه‌ها را انتخاب کرد و سپس با کلیک راست Clear as 0 را برای صفر کردن حافظه‌های انتخاب شده کلیک کرد.



شکل ۱۲-۴۹ پاک کردن داده های مشخصی از جدول مقادیر حافظه های T/C/D

با استفاده از گزینه Transfer کاربر می‌تواند مقادیر جدول را در PLC بارگزاری و یا مقادیر درون PLC (حافظه‌هایT و C و D) را به جدول منتقل کند. برای اینکار ابتدا باید ISPSoft به PLC متصل باشد، در صفحه‌ای که پس از کلیک بر روی Transfer ظاهر می‌شود، با استفاده از گزینه Read from PLC Device Register می‌توان حافظه‌های PLC را استخراج و با استفاده از Write to PLC Device Register می‌توان داده‌های جدول را در PLC بارگزاری کرد. در بخش Bank Area Setup نیز می‌توان محدوده حافظه‌ها را برای انتقال انتخاب کرد، حافظه‌های خارج این محدوده طی فرایند انتقال تغییری نمی کنند. انتقال داده-

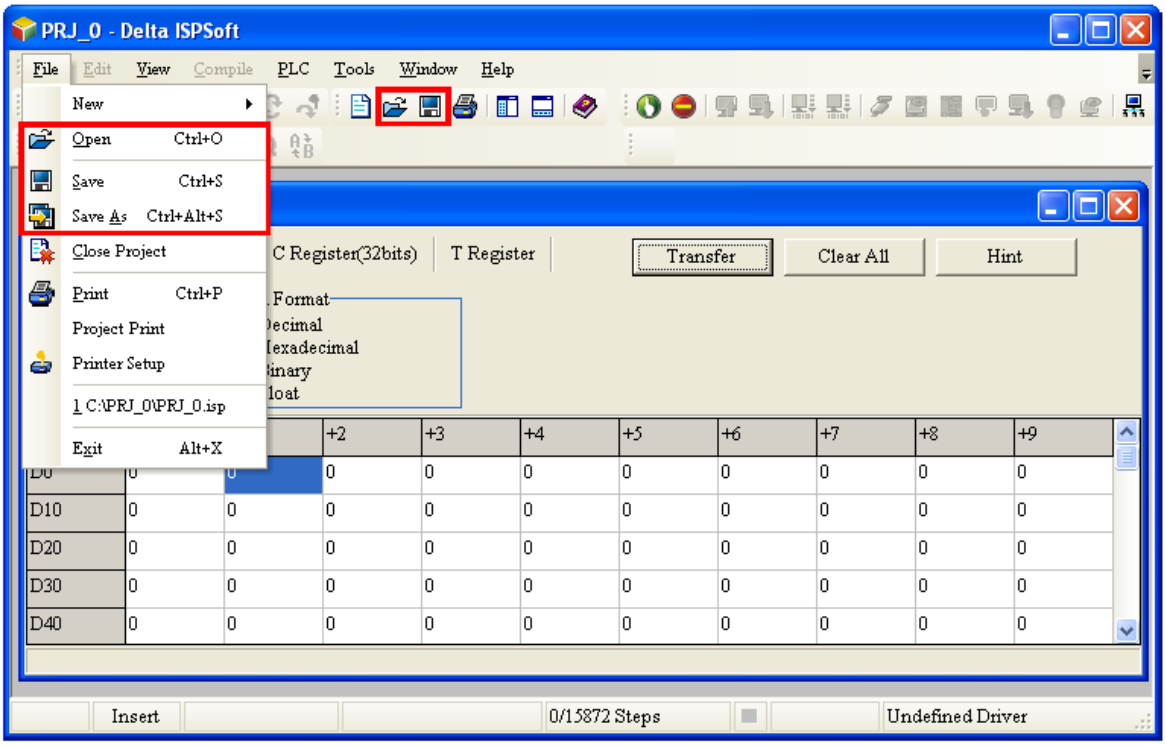
ها پس از کلیک بر روی OK صورت می‌پذیرد (توجه کنید که باید دقت کرد که فرایند انتقال داده از ISPSoft به PLC و به طبع آن تغییر حالت‌های PLC مشکلی برای فرایند سیستم متصل به PLC به وجود نیاورد)



شکل ۱۲-۵۰ تبادل داده های بین PLC و جدول مقادیر حافظه های T/C/D

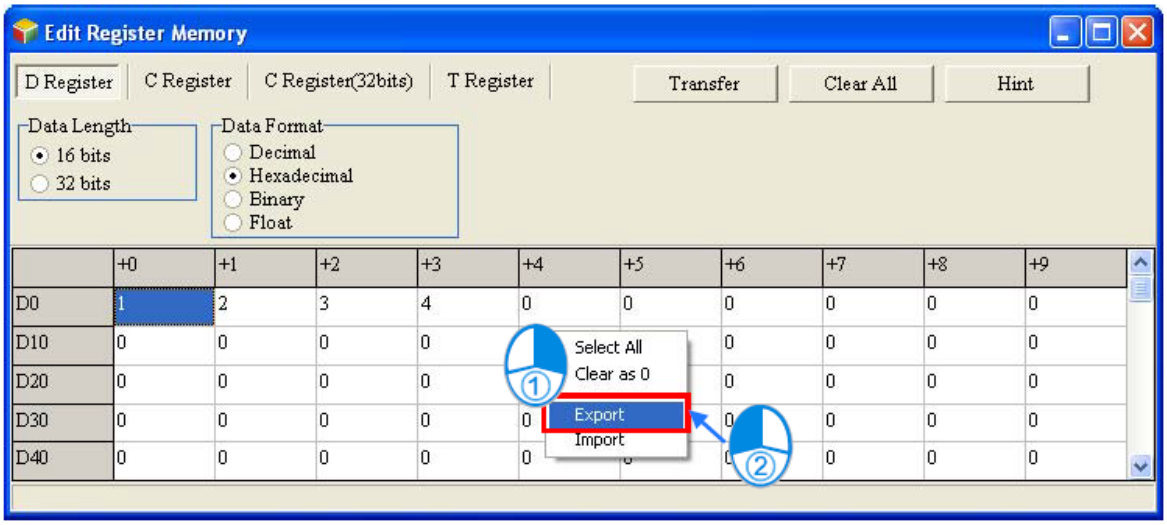
مقادیر حافظه‌ها در صفحه‌ی Edit Register Memory را می‌توان به صورت فایل dvl ذخیره کرد، اگر پنجره فعال، Edit Register Memory باشد، فایل مربوط به آن بعد از کلیک بر روی Save در سربرگ File در پوشه پروژه ذخیره خواهد شد. در صورتی که کاربر بار دیگر صفحه Edit Register Memory را باز کند، نرم افزار فایل مقادیر حافظه‌ها را از فایل dvl داخل پوشه پروژه فراخوانی می‌کند و اگر این فایل وجود نداشته باشد، مقادیر حافظه‌ها را برابر صفر قرار می‌دهد.

برای ذخیره کردن فایل dvl در پوشه‌ای دیگر می‌توان از گزینه Save as در سربرگ File استفاده کرد. اگر بخواهیم فایل dvl در پوشه دیگری را باز کنیم، زمانی که صفحه Edit Register Memory باز و فعال (انتخاب شده توسط موس) است می‌توانیم از منوی File گزینه Open را انتخاب کنیم.

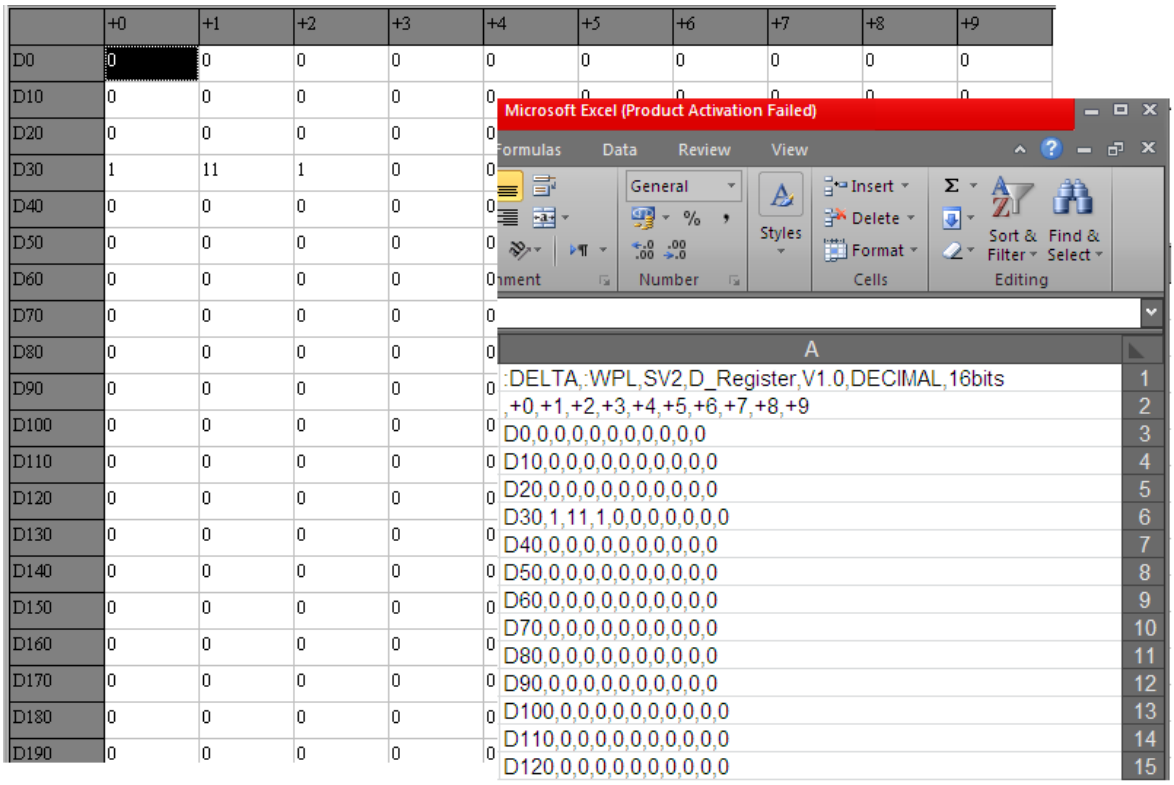


شکل ۱۲-۵۱ ذخیره و بازیابی مقادیر حافظه ها در صفحه ی Edit Register Memory

می‌توان مقادیر صفحه Edit Register Memory را به صورت فایل CSV ذخیره و آن را در نرم افزار مایکروسافت اکسل ذخیره و دوباره مورد استفاده قرار داد. برای ذخیره کردن جدول به صورت فایل CSV باید بر روی صفحه کلیک راست کرده و Export را انتخاب کرد.

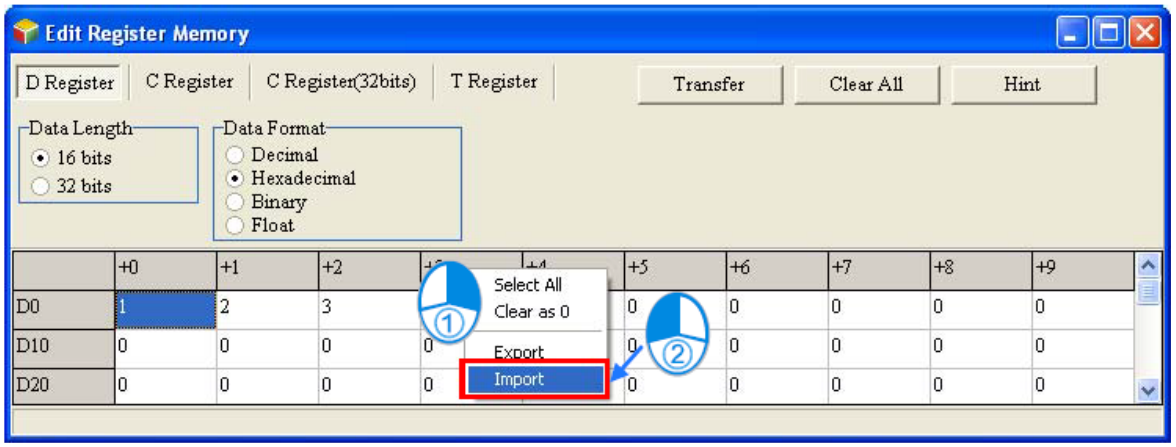


شکل ۱۲-۵۲ ذخیره کردن جدول Edit Register Memory به صورت فایل CSV



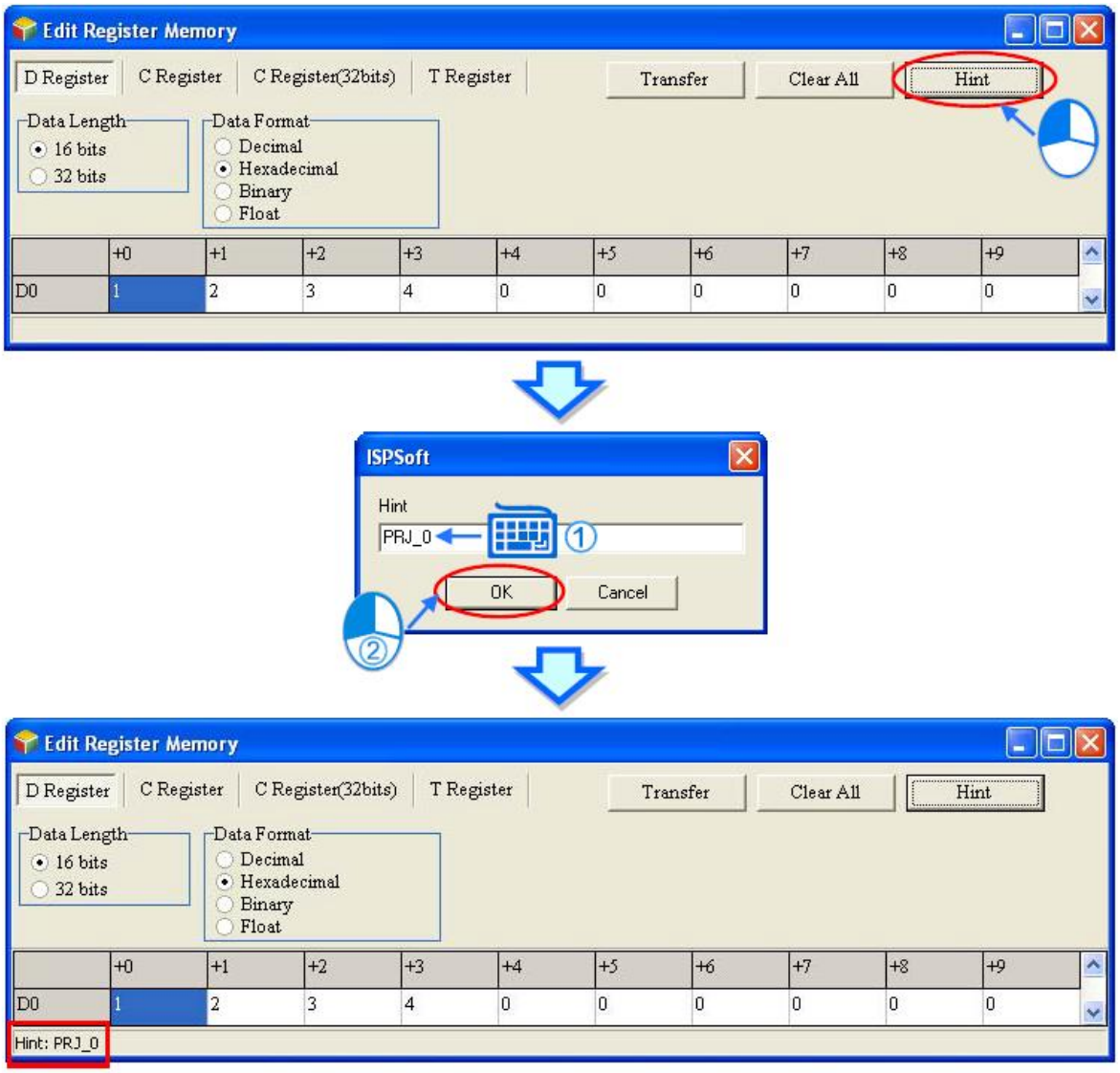
شکل ۱۲-۵۳ اصلاح جدول Edit Register Memory به صورت فایل CSV

همچنین برای استفاده مجدد از فایل اصلاح شده باید بر روی صفحه کلیک راست کرده و Import را انتخاب و سپس آدرس فایل مورد نظر را تعیین کرد.



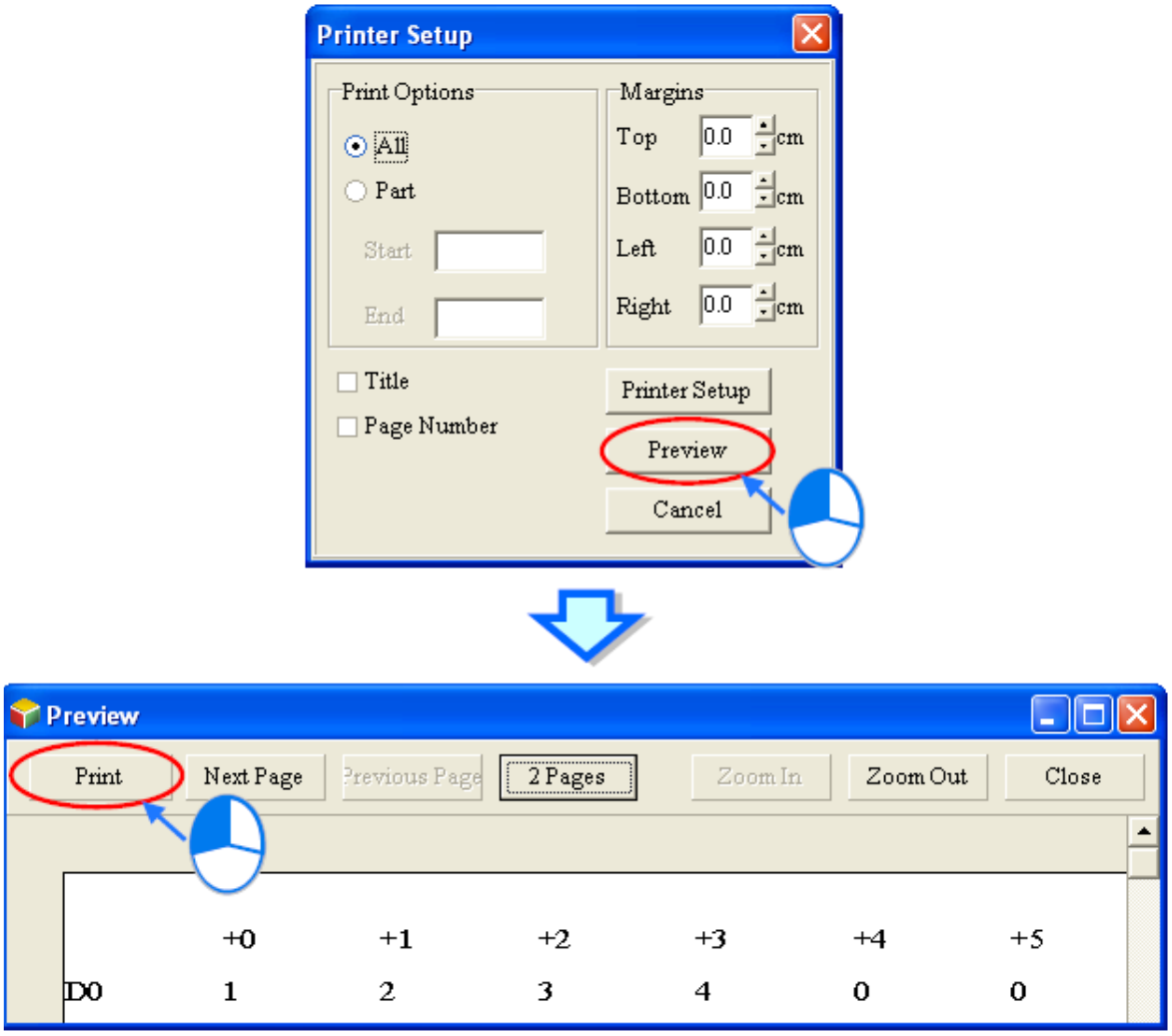
شکل ۱۲-۵۴ استفاده از جدول اصلاح شده Edit Register Memory به صورت فایل CSV

با کلیک بر روی Hint می‌توان توضیحاتی را نیز به همراه وضعیت حافظه‌ها ذخیره کرد، این توضیحات در فایل dvl ذخیره می‌شود.



شکل ۱۲-۵۵ اضافه کردن توضیحات به جدول Edit Register Memory

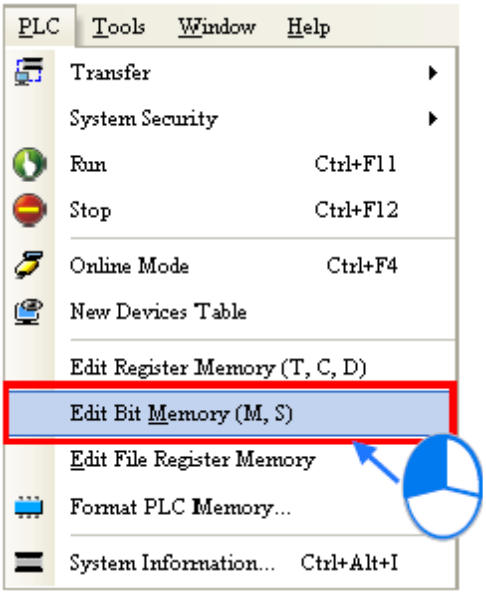
زمانی که پنجره Edit Register Memory انتخاب شده است، با کلیک بر روی  می‌توان جدول مقادیر حافظه‌ها را پرینت گرفت. در پنجره ظاهر شده پس از مشخص کردن محدوده دانلود در قسمت Print Option بر روی Preview کلیک کرده و پس از مشاهده پیش نمایش اقدام به پرینت با فشردن گزینه Print می کنیم.



شکل ۱۲-۵۶ پرینت جدول Edit Register Memory

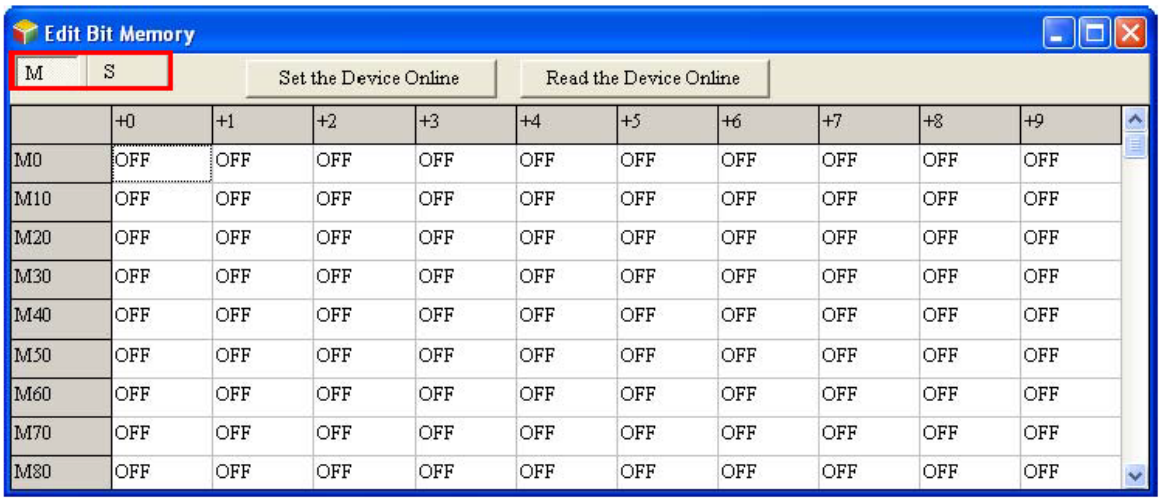
۱۲-۲-۵- اصلاح مقادیر حافظه های M/S در PLCهای سری DVP

کاربران می‌توانند مقادیر حافظه‌های M/S را در PLCهای سری DVP تغییر داده و مقادیر جدید را ذخیره و بارگزاری کنند. برای اینکار می‌توان از سربرگ PLC گزینه Edit Bit Memory (M,S) را انتخاب کرد.



شکل ۱۲-۵۷ انتخاب Edit Bit Memory (M,S)

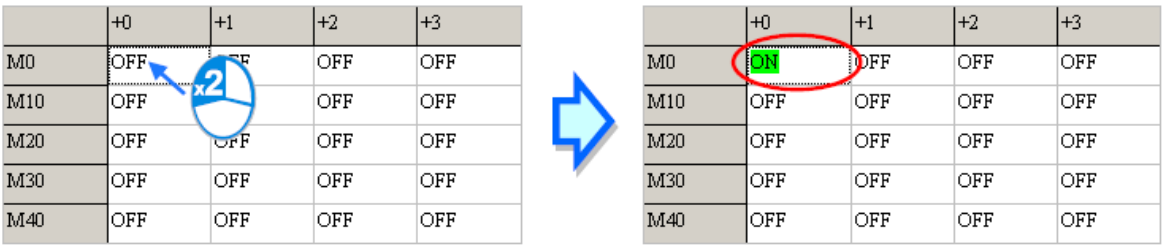
مقادیر موجود در جدول باز شده مقادیر استخراج شده از PLC نیستند بلکه مقادیر ذخیره شده از دفعه قبل می‌باشند، در صورتی که این پنجره برای اولین بار باز شده باشد، تمامی مقادیر صفر (غیرفعال) هستند. با استفاده از سربرگ‌ها می‌توان نوع حافظه را انتخاب کرد.



	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9
M0	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
M10	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
M20	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
M30	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
M40	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
M50	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
M60	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
M70	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
M80	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF

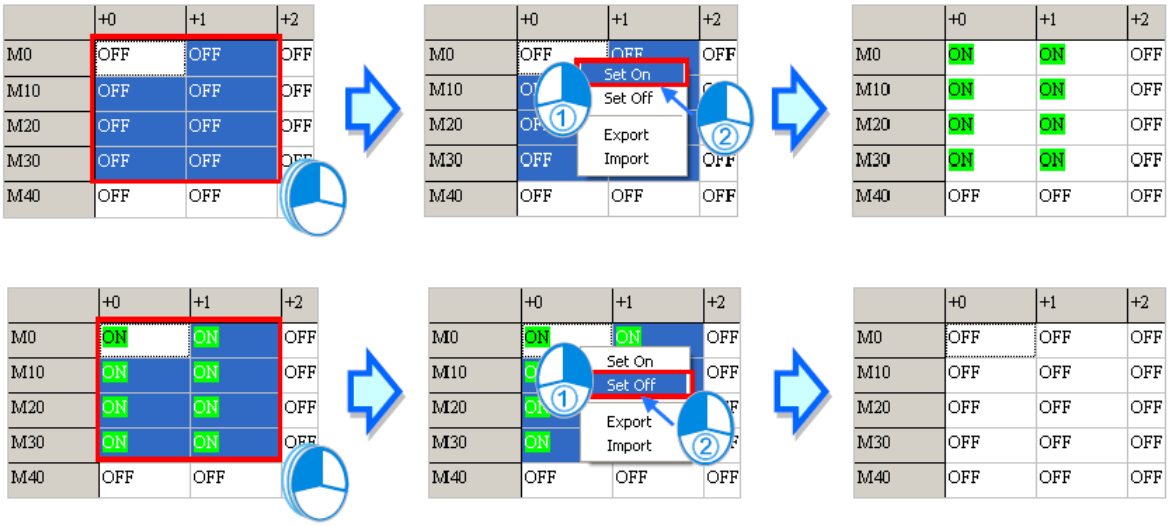
شکل ۱۲-۵۸ پنجره اصلاح مقادیر حافظه های M/S در PLCهای سری

برای تغییر مقادیر حافظه‌های بیتی می‌توان بر روی خانه جدول مرتبط با آن‌ها دوبار کلیک و مقدار مطلوب را تایپ کرد.



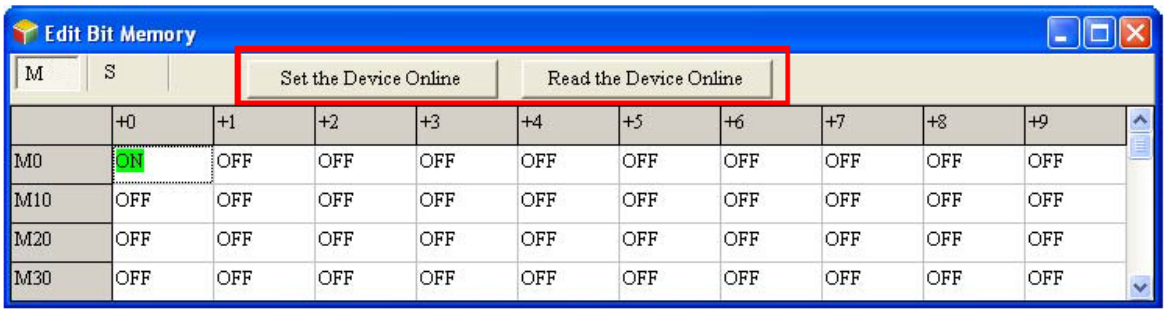
شکل ۱۲-۵۹ وارد کردن مقادیر حافظه های M/S در PLCهای سری DVP

با کشیدن موس می‌توان مجموعه‌ای از خانه‌ها را انتخاب کرد و سپس با کلیک راست با انتخاب Set On حافظه‌های بیتی انتخاب شده را فعال و یا با Set Off آن‌ها را غیرفعال کرد.



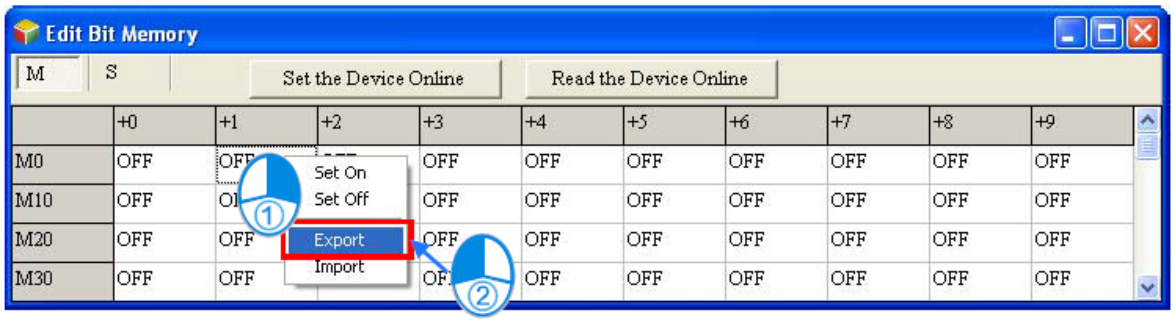
شکل ۱۲-۶۰ اصلاح مقادیر جدول حافظه های M/S در PLCهای سری DVP

با استفاده از گزینه Read the Device Online می‌توان حافظه‌های بیتی PLC را استخراج و با استفاده از Set the Device Online می‌توان داده‌های جدول را در PLC بارگزاری کرد. انتقال داده‌ها پس از کلیک بر روی OK صورت می‌پذیرد و باید طی آن PLC با ISPSoft در ارتباط باشد. (توجه کنید که باید دقت کرد که فرایند انتقال داده از ISPSoft به PLC و به طبع آن تغییر حالت‌های PLC مشکلی برای فرایند سیستم به وجود نیاورد).



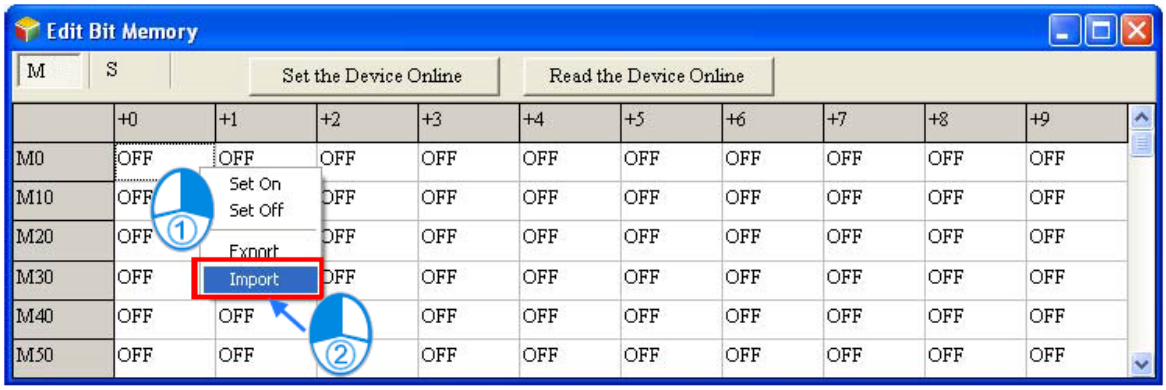
شکل ۱۲-۶۱ تبادل داده های بین PLC و جدول مقادیر حافظه های M/S

می‌توان مقادیر صفحه Edit Bit Memory را به صورت فایل CSV ذخیره و آن را در نرم افزار مایکروسافت اکسل ذخیره و دوباره مورد استفاده قرار داد. برای ذخیره کردن جدول به صورت فایل CSV باید بر روی صفحه کلیک راست کرده و Export را انتخاب کرد.



شکل ۱۲-۶۲ ذخیره کردن جدول Edit Bit Memory به صورت فایل CSV

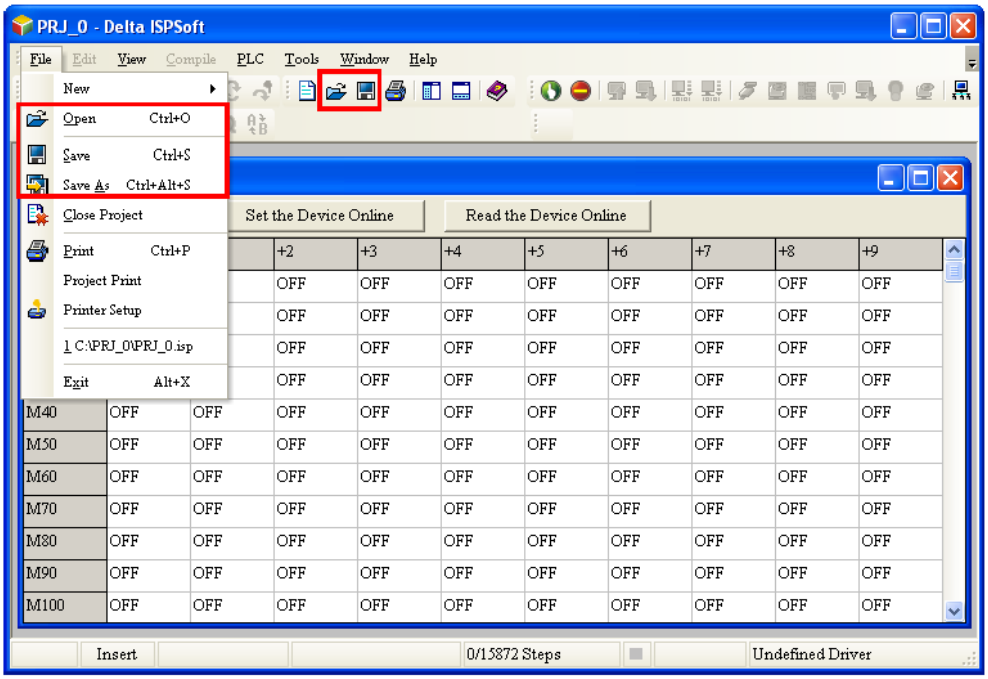
همچنین برای استفاده مجدد از فایل اصلاح شده باید بر روی صفحه کلیک راست کرده و Import را انتخاب و آدرس فایل مورد نظر را نیز تعیین کرد.



شکل ۱۲-۶۳ استفاده از جدول اصلاح شده Edit Bit Memory به صورت فایل CSV

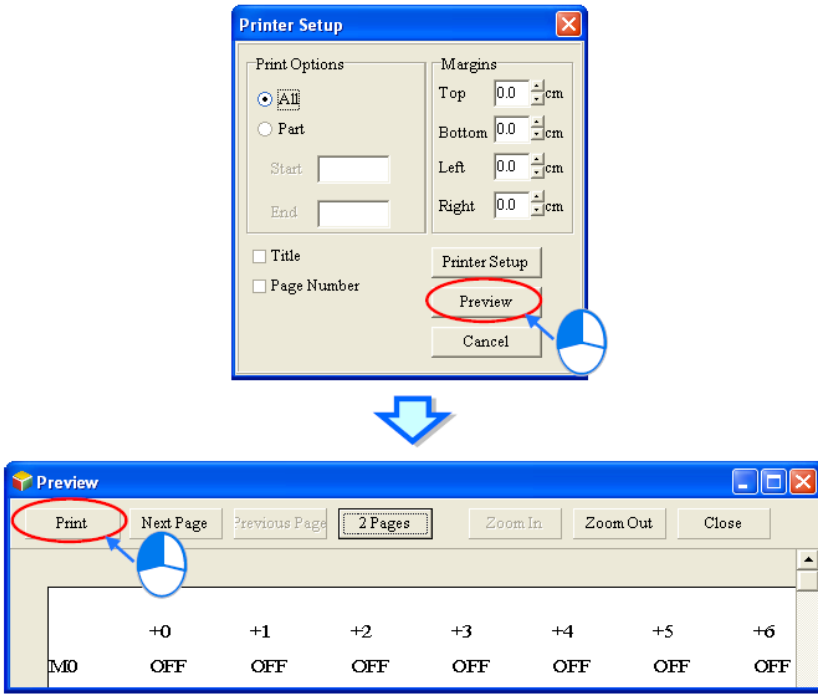
مقادیر حافظه‌ها در صفحه‌ی Edit Bit Memory را می‌توان به صورت فایل با پسوند dvb ذخیره کرد، اگر پنجره فعال Edit Bit Memory باشد، فایل مربوط به آن بعد از کلیک بر روی Save در سربرگ File در پوشه پروژه ذخیره خواهد شد. در صورتی که کاربر بار دیگر صفحه Edit Bit Memory را باز کند، نرم افزار فایل مقادیر حافظه‌ها را از فایل dvb داخل پوشه پروژه فراخوانی می‌کند و اگر این فایل وجود نداشته باشد، مقادیر حافظه‌ها را برابر صفر (غیرفعال) قرار می‌دهد.

برای ذخیره کردن فایل dvb در پوشه‌ای دیگر می‌توان از گزینه Save as در سربرگ File استفاده کرد. اگر بخواهیم فایل dvb را در پوشه دیگری را باز کنیم، زمانی که صفحه Edit Bit Memory باز و فعال است می‌توانیم از منوی File گزینه Open را انتخاب کنیم.



شکل ۱۲-۶۴ ذخیره و بازیابی مقادیر حافظه ها در صفحه ی Edit Bit Memory

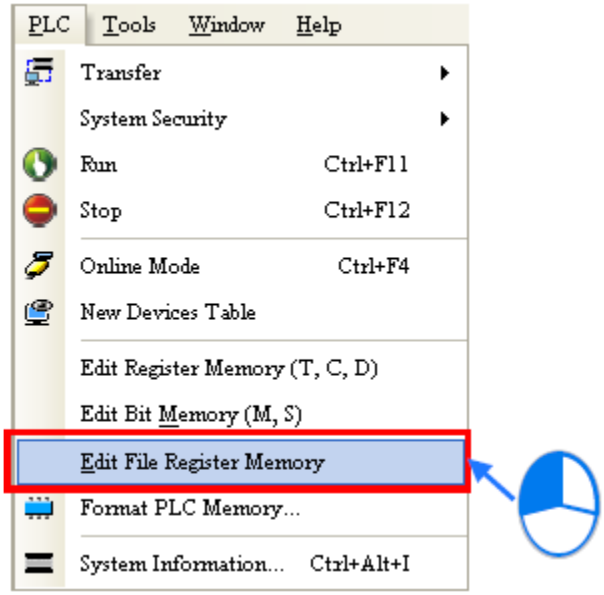
در زمانی که پنجره Edit Bit Memory انتخاب شده است، با کلیک بر روی  می‌توان جدول مقادیر حافظه‌ها را پرینت گرفت. در پنجره ظاهر شده با مشخص کردن محدوده دانلود در قسمت Print Option بر روی Preview کلیک کرده و پس از مشاهده پیش نمایش اقدام به پرینت گرفتن با فشردن Print می کنیم.



شکل ۱۲-۶۵ پرینت جدول Edit Bit Memory

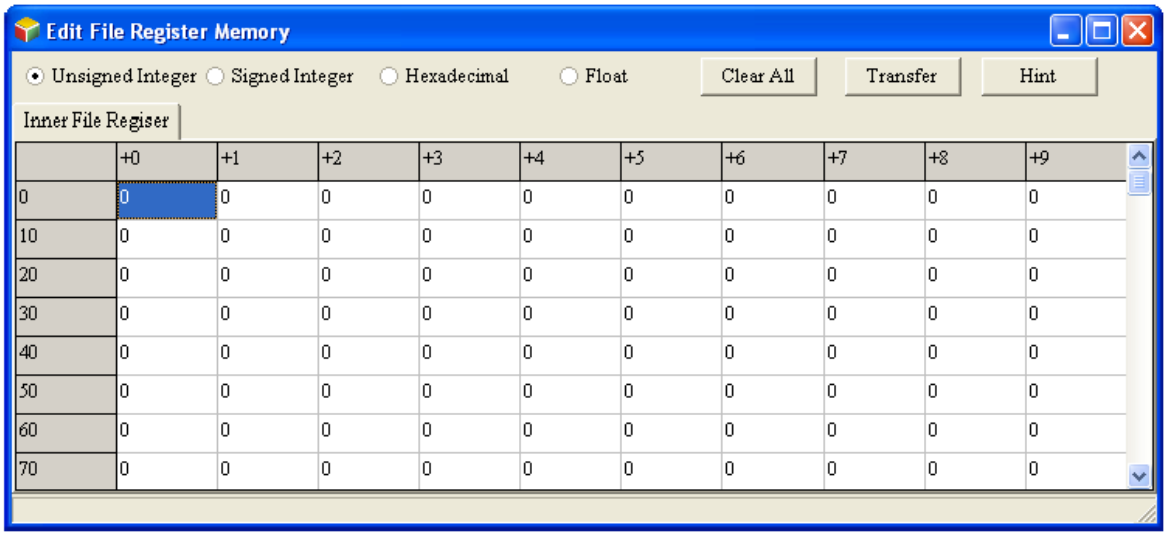
۱۲-۲-۶- اصلاح مقادیر حافظه های فایل در PLCهای سری DVP

کاربران می‌توانند مقادیر حافظه‌های فایل را در PLCهای سری DVP تغییر داده و مقادیر جدید را ذخیره و بارگزاری کنند. برای اینکار می‌توان از سربرگ PLC گزینه Edit File Register Memory را انتخاب کرد.



شکل ۱۲-۶۶ انتخاب Edit File Register Memory

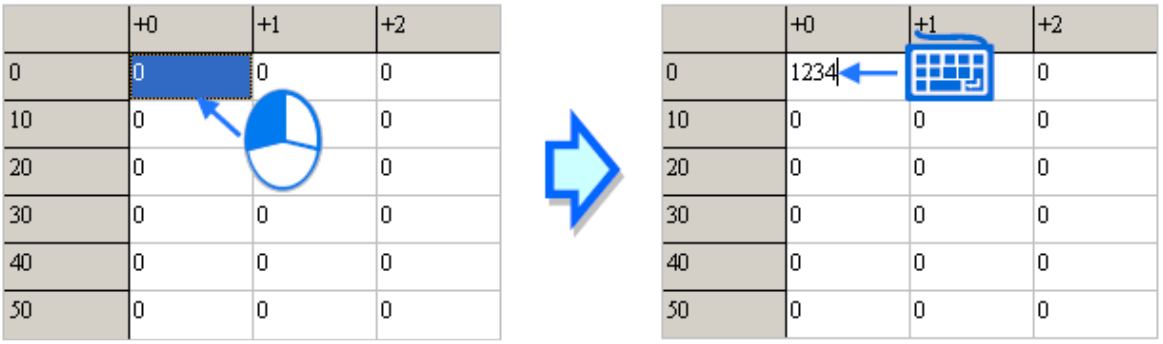
مقادیر موجود در جدول باز شده مقادیر استخراج شده از PLC نیستند بلکه مقادیر ذخیره شده از دفعه قبل می‌باشند، در صورتی که این پنجره برای اولین بار باز شده باشد، تمامی مقادیر صفر هستند.



	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9
0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0
30	0	0	0	0	0	0	0	0	0	0
40	0	0	0	0	0	0	0	0	0	0
50	0	0	0	0	0	0	0	0	0	0
60	0	0	0	0	0	0	0	0	0	0
70	0	0	0	0	0	0	0	0	0	0

شکل ۱۲-۶۷ پنجره اصلاح مقادیر حافظه های فایل در PLCهای سری DVP

برای تغییر مقادیر حافظه‌ها می‌توان بر روی خانه جدول مرتبط با آن‌ها کلیک و مقدار مطلوب را تایپ کرد.

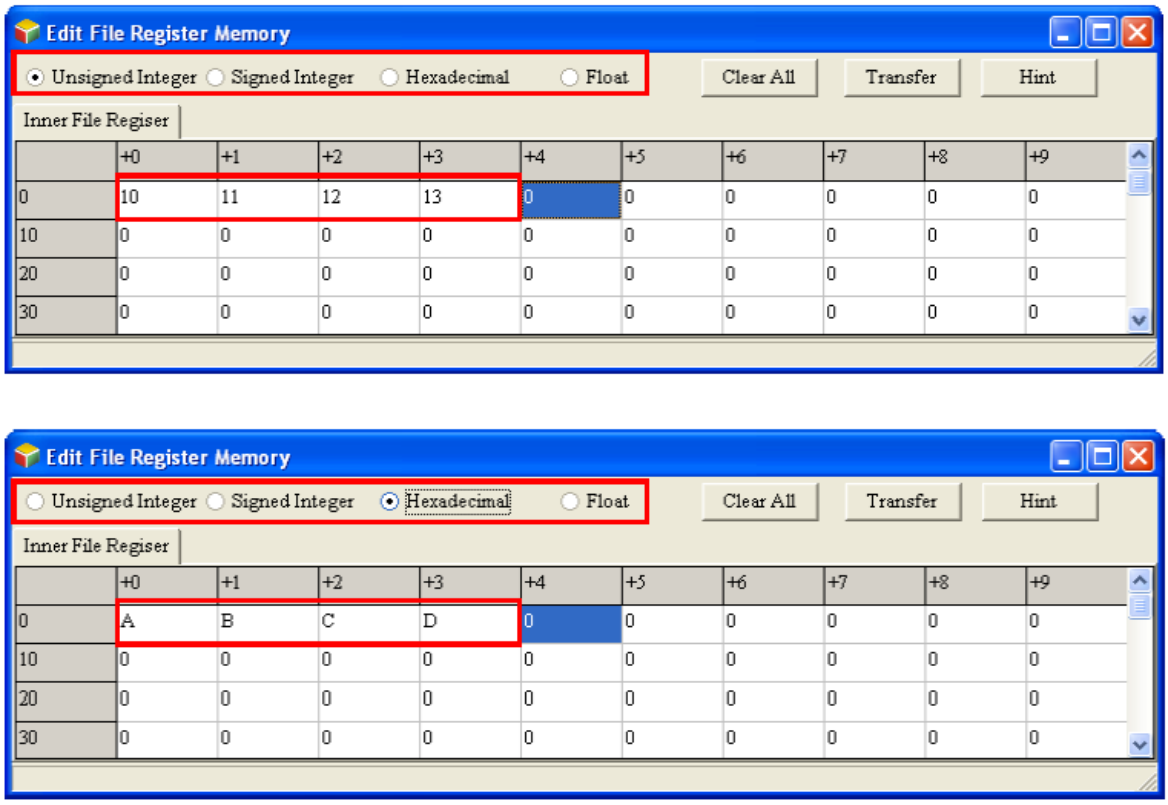


	+0	+1	+2
0	0	0	0
10	0		0
20	0		0
30	0	0	0
40	0	0	0
50	0	0	0

	+0	+1	+2
0	1234		0
10	0	0	0
20	0	0	0
30	0	0	0
40	0	0	0
50	0	0	0

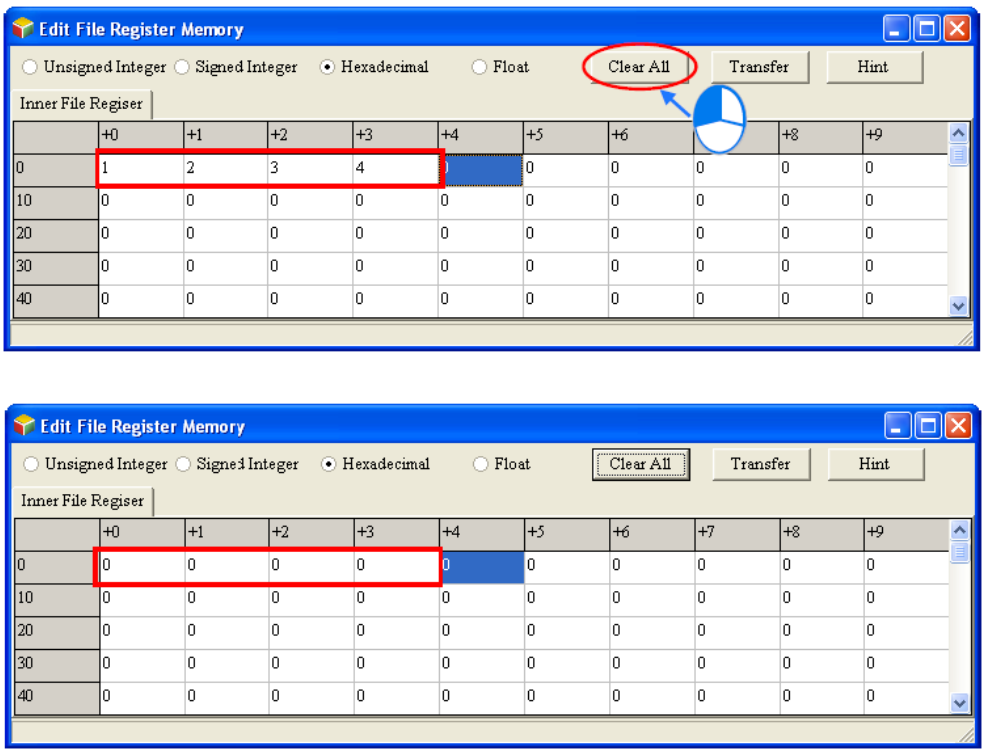
شکل ۱۲-۶۸ وارد کردن مقادیر حافظه های فایل در PLCهای سری DVP

می‌توان نوع نمایش اعداد حافظه‌ها را در بالای پنجره Edit File Register Memory تعیین کرد.



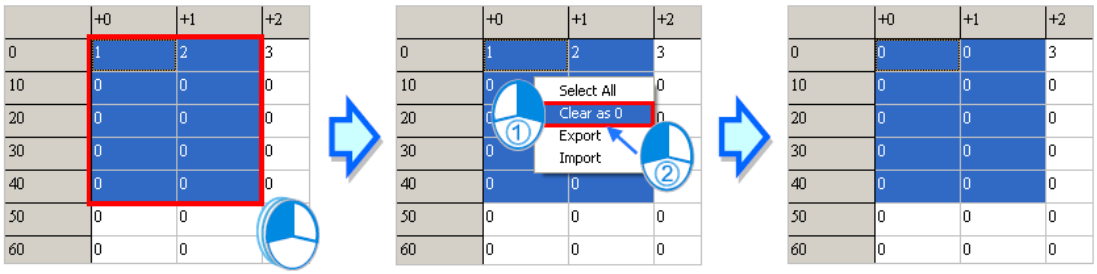
شکل ۱۲-۶۹ نوع نمایش اعداد در جدول مقادیر حافظه های فایل

می‌توان با کلیک بر روی Clear All تمامی حافظه‌ها را صفر کرد.



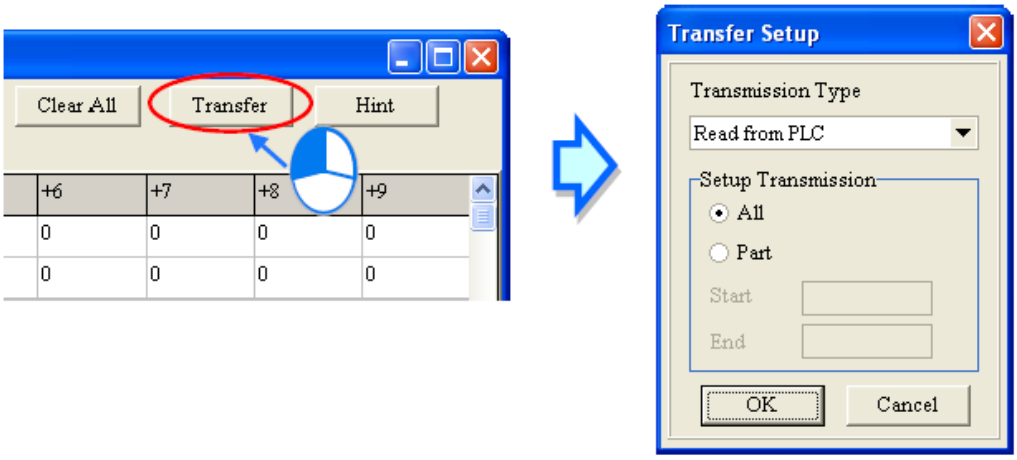
شکل ۱۲-۷۰ پاک کردن داده های جدول مقادیر حافظه های فایل

همچنین با کشیدن موس می‌توان مجموعه‌ای از خانه‌ها را انتخاب کرد و سپس با کلیک راست Clear as 0 را برای صفر کردن حافظه‌های انتخاب شده کلیک کرد.



شکل ۱۲-۷۱ پاک کردن داده های مشخصی از جدول مقادیر حافظه های فایل

با استفاده از گزینه Transfer کاربر می‌تواند مقادیر جدول را در PLC بارگزاری و یا مقادیر درون PLC را به جدول استخراج کند. برای اینکار ابتدا باید ISPSoft به PLC متصل باشد. در صفحه‌ای که پس از کلیک بر روی Transfer ظاهر می‌شود، در قسمت Transmission Type می‌توان با استفاده از گزینه Read from PLC حافظه‌های PLC را استخراج و با استفاده از Write to PLC می‌توان داده‌های جدول را در PLC بارگزاری کرد. در بخش Setup Transmission نیز می‌توان محدوده حافظه‌ها را برای انتقال انتخاب کرد، حافظه‌های خارج این محدوده طی فرایند انتقال تغییری نمی کنند. انتقال داده‌ها پس از کلیک بر روی OK صورت می‌پذیرد (توجه کنید که باید دقت کرد که فرایند انتقال داده از ISPSoft به PLC و به طبع آن تغییر حالت‌های PLC مشکلی برای فرایند سیستم به وجود نیاورد)

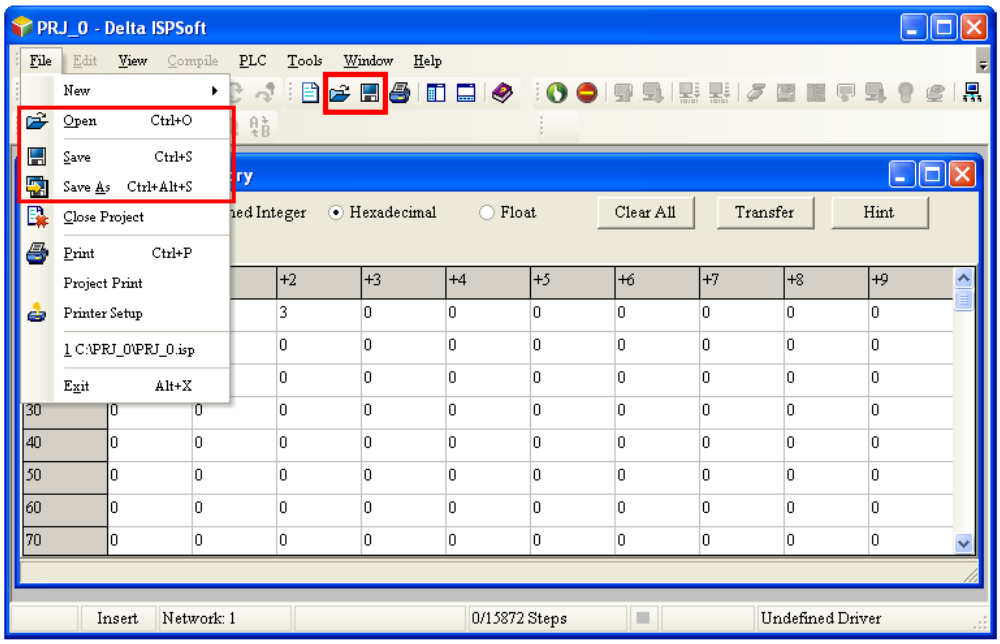


شکل ۱۲-۷۲ تبادل داده های بین PLC و جدول مقادیر حافظه های فایل

مقادیر حافظه‌ها در صفحه‌ی Edit File Register Memory را می‌توان به صورت فایل wft ذخیره کرد، اگر پنجره فعال، Edit File Register Memory باشد، فایل مربوط به آن بعد از کلیک بر روی Save در سربرگ File در پوشه پروژه ذخیره خواهد شد. در صورتی که کاربر بار دیگر صفحه Edit File

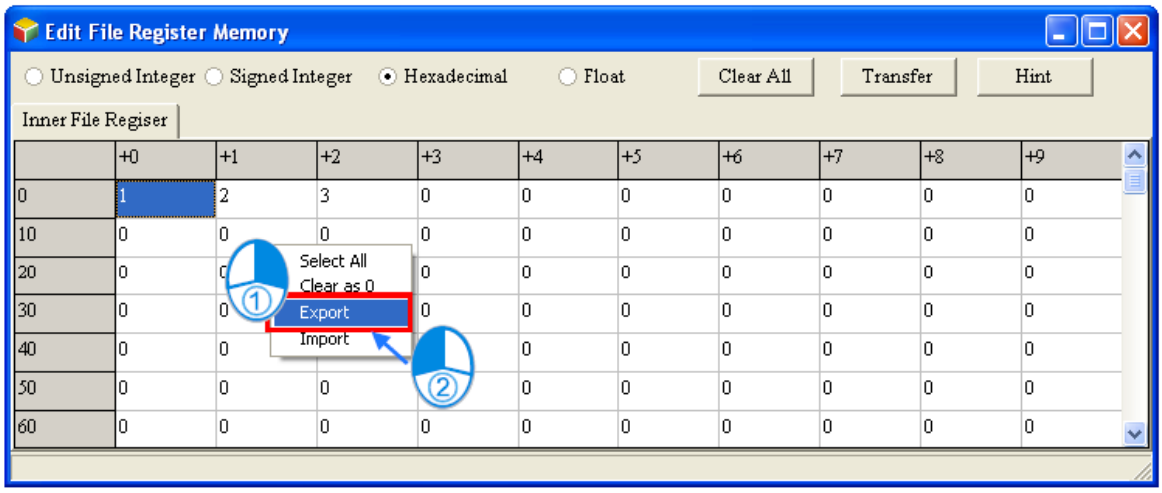
Register Memory را باز کند، نرم افزار فایل مقادیر حافظه‌ها را از فایل wft داخل پوشه پروژه فراخوانی می‌کند و اگر این فایل وجود نداشته باشد، مقادیر حافظه‌ها را برابر صفر قرار می‌دهد.

برای ذخیره کردن فایل wft در پوشه‌ای دیگر می‌توان از گزینه Save as در سربرگ File استفاده کرد. اگر بخواهیم فایل wft در پوشه دیگری را باز کنیم، زمانی که صفحه Edit File Register Memory باز و فعال است می‌توانیم از منوی File گزینه Open را انتخاب کنیم.



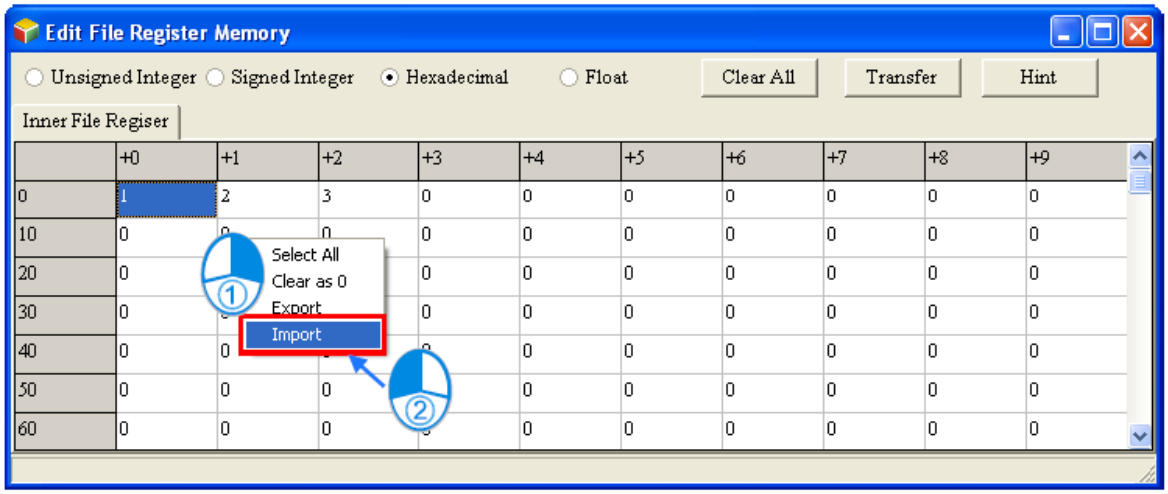
شکل ۱۲-۷۳ ذخیره و بازیابی مقادیر حافظه ها در صفحه ی Edit File Register Memory

می‌توان مقادیر صفحه Edit File Register Memory را به صورت فایل CSV ذخیره و آن را در نرم افزار مایکروسافت اکسل ذخیره و دوباره مورد استفاده قرار داد. برای ذخیره کردن جدول به صورت فایل CSV باید بر روی صفحه کلیک راست کرده و Export را انتخاب کرد.



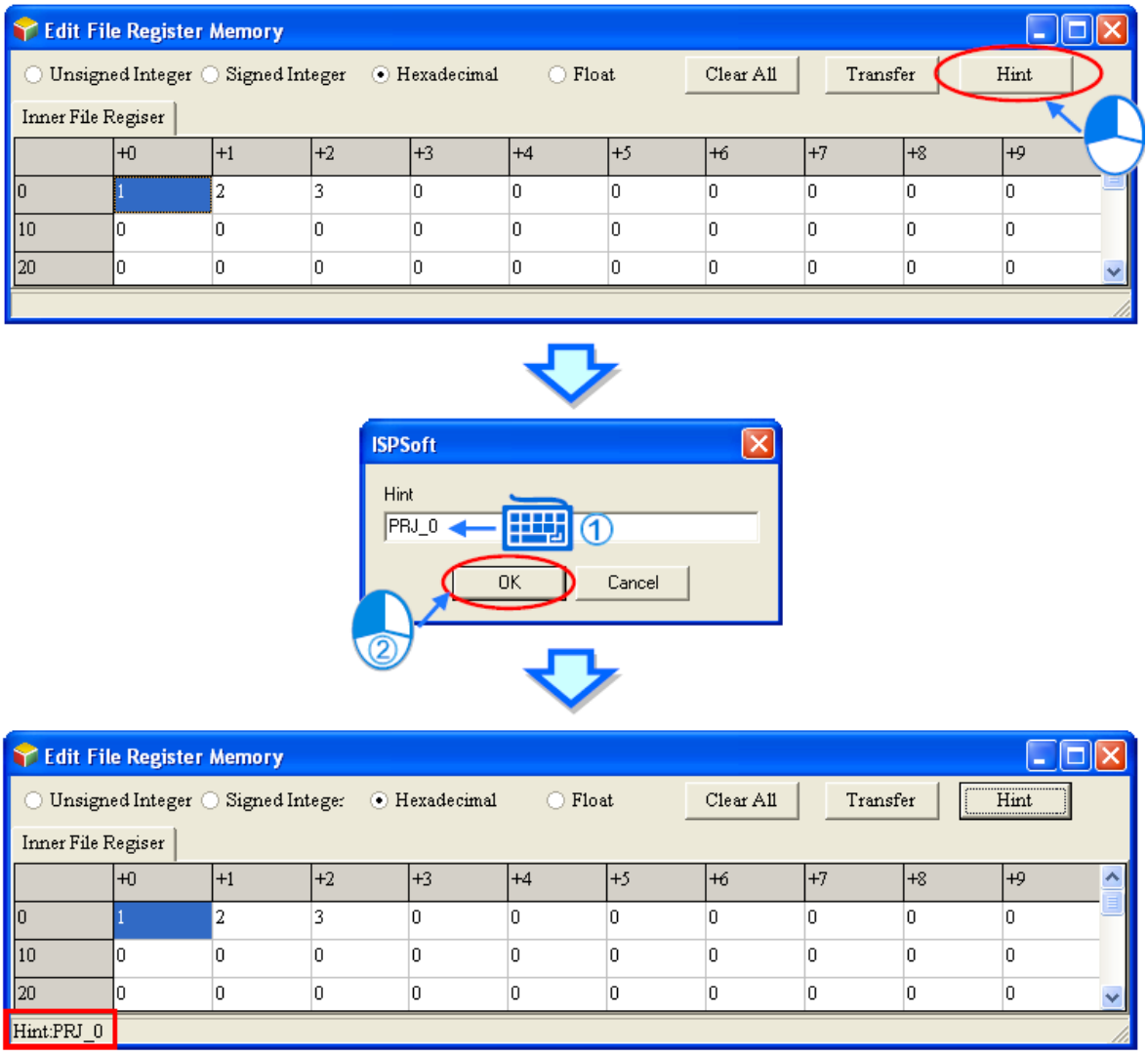
شکل ۱۲-۷۴ ذخیره کردن جدول Edit File Register Memory به صورت فایل CSV

همچنین برای استفاده مجدد از فایل اصلاح شده باید بر روی صفحه کلیک راست کرده و Import را انتخاب و سپس آدرس فایل مورد نظر را تعیین کرد.



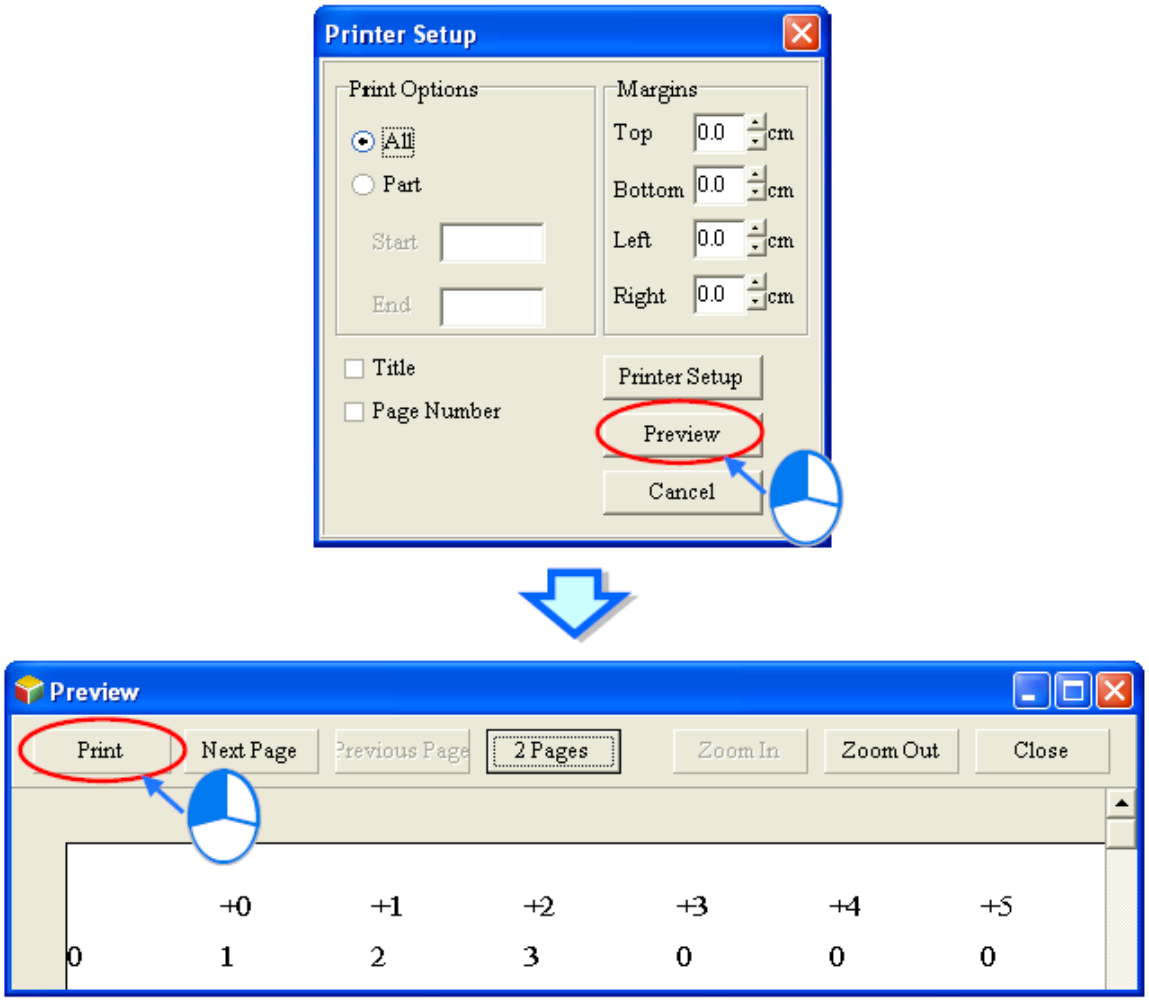
شکل ۱۲-۷۵ استفاده از جدول اصلاح شده Edit File Register Memory به صورت فایل CSV

با کلیک بر روی Hint می‌توان توضیحاتی را نیز به همراه وضعیت حافظه‌ها ذخیره کرد، این توضیحات در فایل wft ذخیره می‌شود.



شکل ۱۲-۷۶ اضافه کردن توضیحات به جدول Edit File Register Memory

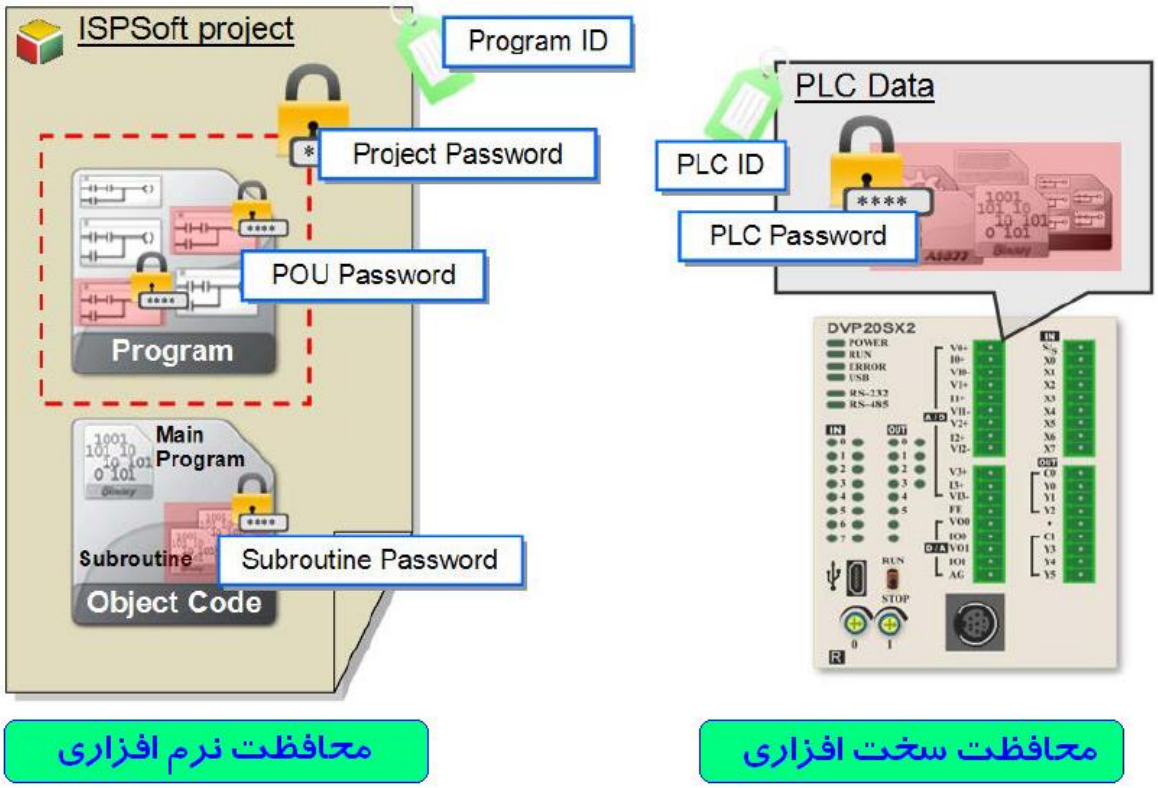
در زمانی که پنجره Edit File Register Memory انتخاب شده است، با کلیک بر روی  می‌توان جدول مقادیر حافظه‌ها را پرینت گرفت. در پنجره ظاهر شده با مشخص کردن محدوده دانلود در قسمت Print Option بر روی Preview کلیک کرده و پس از مشاهده پیش نمایش اقدام به پرینت با فشردن Print می کنیم.



شکل ۱۲-۷۷ پرینت جدول Edit File Register Memory

۱۲-۳- رمزگذاری و محافظت از داده ها

در ISPSoft مکانیزم‌های متعددی برای محافظت از پروژه‌های کاربران در نظر گرفته شده است. روش-های مختلف محافظت و رمزگذاری در ISPSoft را می‌توان در شکل زیر دید.



شکل ۱۲-۷۸ روش‌های متعدد رمزگذاری در ISPSoft

جدول ۱۲-۸: شرح روش‌های متعدد رمزگذاری در ISPSoft

نوع رمز عبور		شرح
در نرم افزار ISPSoft	Program ID (شناسه برنامه)	اولین بررسی امنیتی در PLC است. در صورتی که کاربر بخواهد پروژه‌ای را در PLC بارگزاری کند، Program ID تنظیم شده برای برنامه باید مشابه PLC ID باشد. در نتیجه با استفاده از آن می‌توان پروژه‌های تعیین شده را صرفا بر روی PLCهای مشخصی بارگزاری کرد.
	Project password	برای محافظت از برنامه‌های درون پروژه. در پروژه‌های داری رمز عبور، کاربر برای کار با POU ابتدا باید رمز عبور صحیح را وارد کند.
	POU Password	برای محافظت از POU. در صورتی که کاربر بخواهد از کُد داخل POU و یا تکنیک‌های به کار رفته در آن محافظت کند، می‌تواند بر روی آن رمز عبور تعبیه کند.
	Subroutine password	برای محافظت از زیربرنامه‌ها پس از استخراج. این رمز عبور امکان

		آنالیز بلوک بعد از استخراج از PLC را ناممکن می‌کند. تنها PLCهای سری DVP از این مکانیزم پشتیبانی می‌کنند.
در PLC	PLC ID (شناسه PLC)	اولین بررسی امنیتی در PLC است. در صورتی که کاربر بخواهد پروژه‌ای را در PLC بارگزاری کند، Program ID تنظیم شده برای برنامه باید مشابه PLC ID باشد. در نتیجه با استفاده از آن می‌توان پروژه‌های تعیین شده را صرفا بر روی PLCهای مشخصی بارگزاری کرد.
	PLC Password	برای محافظت از داده‌های PLC. برای بارگزاری و استخراج داده و یا برنامه در PLC باید رمز عبور PLC را وارد کرد.

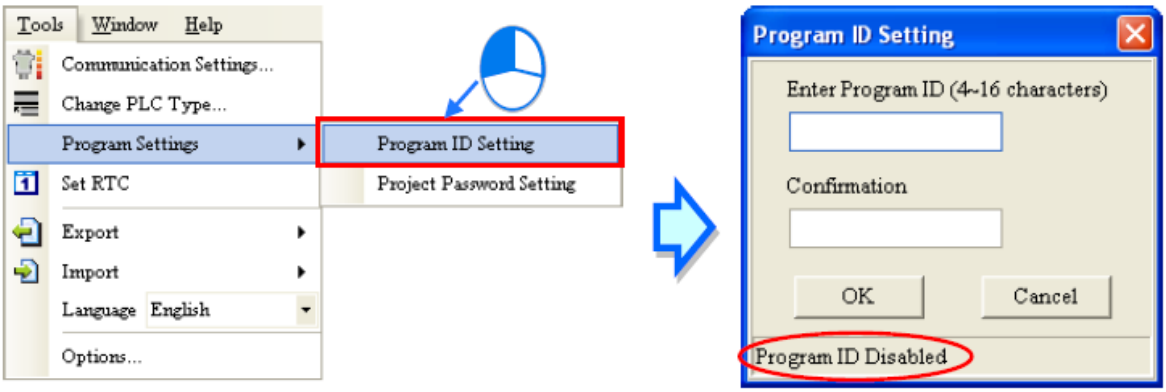
۱۲-۳-۱- شناسه برنامه و PLC

اولین بررسی امنیتی در PLC است. در صورتی که کاربر بخواهد پروژه‌ای را در PLC بارگزاری کند، Program ID تنظیم شده برای برنامه باید مشابه PLC ID باشد. در نتیجه با استفاده از آن می‌توان پروژه-های تعیین شده را صرفا بر روی PLCهای مشخصی بارگزاری کرد.

در صورتی که کاربر بخواهد اطلاعات درون PLC دارای شناسه‌ای را استخراج کند، ابتدا باید شناسه (ID) آن را وارد کند. همچنین برای اینکه بتوان پروژه‌ای شامل شناسه در ISPSoft را باز کرد ابتدا باید شناسه آن به وسیله کاربر وارد شود.

۱۲-۳-۱-۱- تنظیمات شناسه برنامه

در سربرگ Tools و قسمت Program Setting می‌توان بر روی Program ID Setting کلیک کرد تا صفحه تنظیم شناسه برنامه باز شود. جمله‌ی زیر این صفحه نشان می‌دهد که آیا این پروژه با شناسه برنامه محافظت شده است و یا خیر.

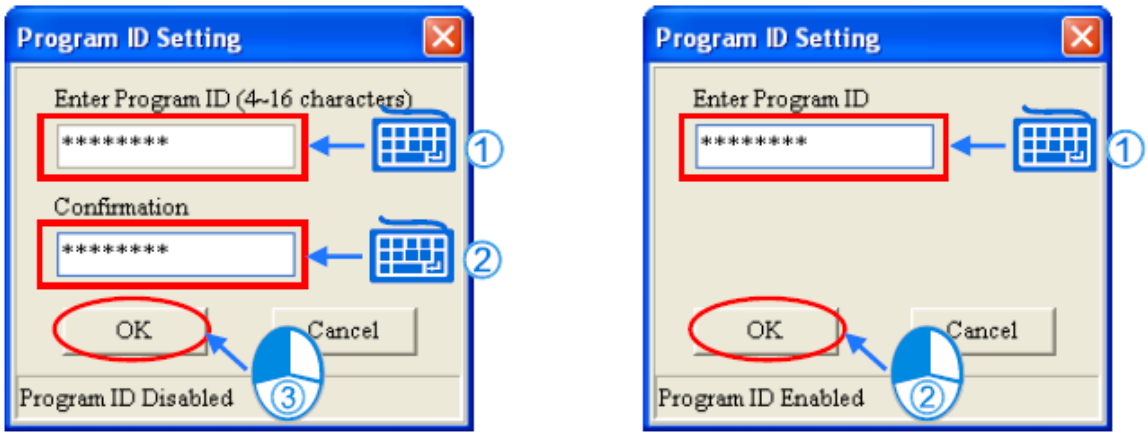


شکل ۱۲-۷۹ صفحه تنظیم شناسه برنامه

در صورتی که برنامه با شناسه محافظت نشده باشد پیام "Program ID Disabled" مواجه می‌شویم و در پنجره باز شده دو محل برای وارد کردن شناسه برنامه و تایید آن وجود دارد که با وارد کردن آن‌ها و کلیک بر روی OK برنامه دارای شناسه برنامه می‌شود.

در صورتی که برنامه با شناسه محافظت شده باشد پیام "Program ID Enabled" مواجه می‌شویم و در پنجره باز شده محلی برای وارد کردن شناسه برنامه وجود دارد که با وارد کردن آن‌ها و کلیک بر روی OK شناسه برنامه حذف می‌شود.

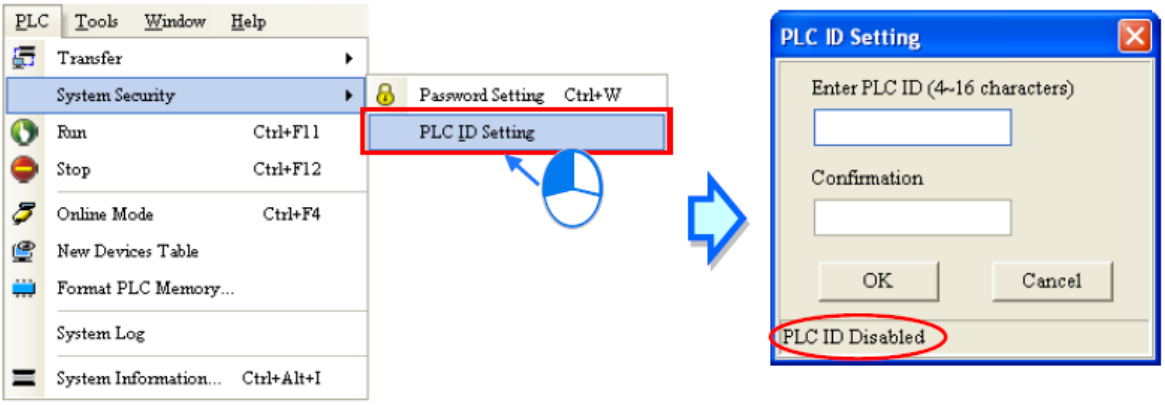
رمز عبور را می‌توانیم بین تعداد محدودی کاراکتر (این تعداد به نوع PLC وابسته است) از حروف انگلیسی، اعداد و بعضی علائم مانند "_" انتخاب کنیم. رمز عبور به بزرگ و کوچک بودن حروف حساس است.



شکل ۱۲-۸۰ ایجاد و حذف شناسه برنامه

۱۲-۳-۱-۲- تنظیمات شناسه PLC

برای تنظیم شناسه PLC ابتدا باید ISPSoft را به PLC متصل کرد. در سربرگ PLC و قسمت System Security می‌توان بر روی PLC ID Setting کلیک کرد تا صفحه تنظیم شناسه PLC باز شود. در این مرحله ISPSoft به PLC متصل می‌شود و داده‌های آن را می‌خواند، در صورت موفقیت آمیز بودن این ارتباط پنجره‌ای باز می‌شود و جمله‌ی زیر آن نشان می‌دهد که آیا PLC با شناسه محافظت شده است و یا خیر.

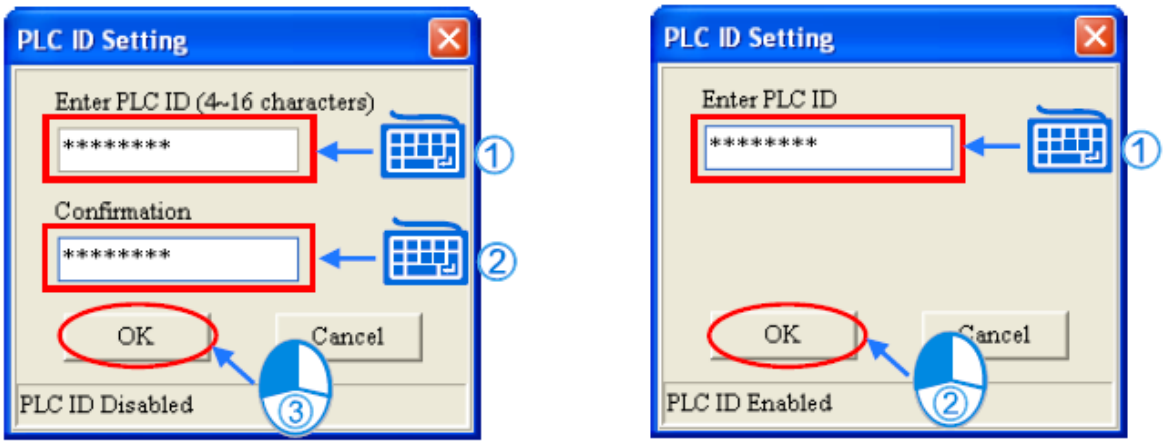


شکل ۱۲-۸۱ صفحه تنظیم شناسه PLC

در صورتی که PLC با شناسه محافظت نشده باشد پیام "PLC ID Disabled" مواجه می‌شویم و در پنجره باز شده دو محل برای وارد کردن شناسه PLC و تایید آن وجود دارد که با وارد کردن آن‌ها و کلیک بر روی OK سخت افزار دارای شناسه PLC می‌شود.

در صورتی که PLC با شناسه محافظت شده باشد پیام "PLC ID Enabled" مواجه می‌شویم و در پنجره باز شده محلی برای وارد کردن شناسه PLC وجود دارد که با وارد کردن آن‌ها و کلیک بر روی OK شناسه PLC حذف می‌شود.

رمز عبور را می‌توانیم بین تعداد محدودی کاراکتر (این تعداد به نوع PLC وابسته است) از حروف انگلیسی، اعداد و بعضی علائم مانند "_" انتخاب کنیم. رمز عبور به بزرگ و کوچک بودن حروف حساس است.



شکل ۱۲-۸۲ ایجاد و حذف شناسه PLC

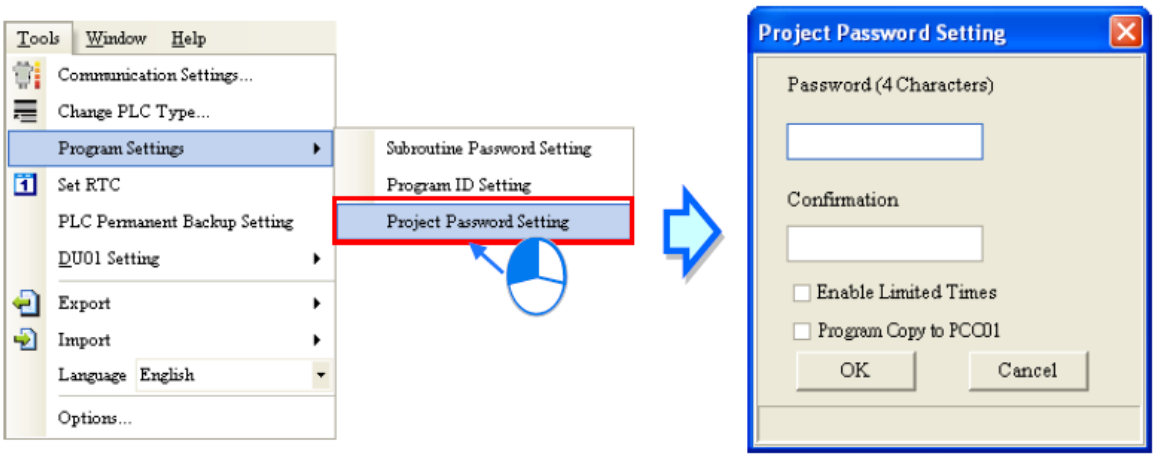
۱۲-۳-۲- رمز عبور پروژه و PLC

رمز عبور پروژه برای محافظت برنامه‌های درون پروژه مورد استفاده قرار می‌گیرد. برای باز کردن POUهای درون پروژه ابتدا باید رمز عبور پروژه را وارد کرد. همچنین کاربر می‌تواند تعداد دفعات وارد کردن رمز عبور ناصحیح را محدود کند بدین صورت که کسی نتواند با سعی و خطا به صورت شانسی رمز عبور را وارد کند، در این حالت اگر از تعداد مجاز وارد کردن رمز عبور رد شود، پروژه بسته می‌شود.

رمز عبور PLC برای محافظت برنامه‌های درون PLC از تغییر و به سرقت رفتن مورد استفاده قرار می‌گیرد. برای بارگزاری پروژه درون PLC و یا استخراج برنامه از آن ابتدا باید رمز عبور PLC را وارد کرد. همچنین کاربر می‌تواند تعداد دفعات وارد کردن رمز عبور ناصحیح را برای PLC نیز محدود کند بدین صورت که کسی نتواند با سعی و خطا به صورت شانسی رمز عبور را وارد کند، در این حالت اگر تعداد مجاز وارد کردن رمز عبور بگذرد PLC قفل می‌شود و امکان بارگزاری و استخراج داده‌ها وجود نخواهد داشت. برای خارج کردن PLC از حالت قفل کاربر باید تنظیمات آن را به حالت پیشفرض ببرد (آن را Reset نماید).

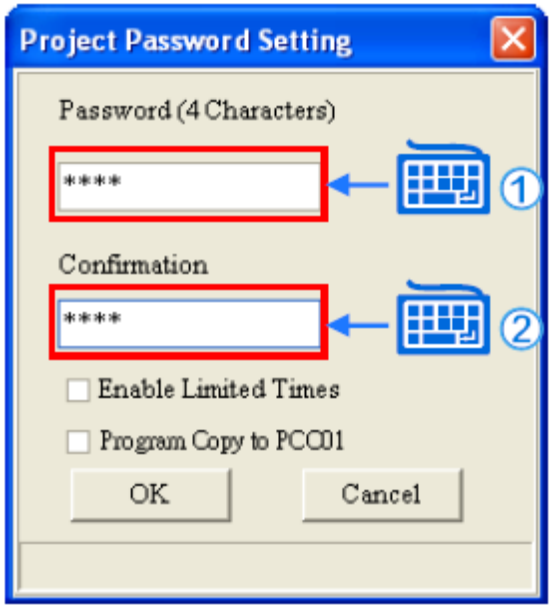
۱۲-۳-۲-۱- تنظیم رمز عبور پروژه

در سربرگ Tools و قسمت Program Setting می‌توان بر روی Project Password Setting کلیک کرد تا صفحه تنظیم رمز عبور پروژه باز شود. جمله‌ی زیر این صفحه نشان می‌دهد که آیا این پروژه با رمز عبور محافظت شده است و یا خیر. در این صفحه در صورتی که مدل PLC از خانواده DVP باشد، گزینه Program Copy to PCC01 نیز در آن ظاهر می‌شود.



شکل ۱۲-۸۳ تنظیم رمز عبور پروژه

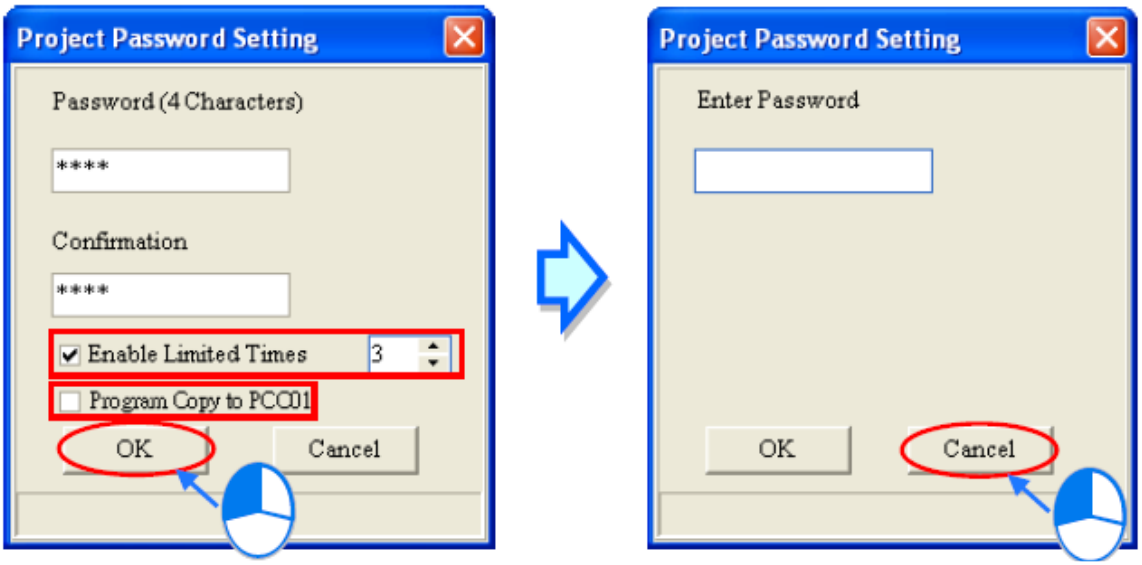
در صورتی که پروژه با رمز عبور محافظت نشده باشد در پنجره باز شده دو محل برای وارد کردن رمز عبور پروژه و تایید آن وجود دارد که با وارد کردن آن‌ها و کلیک بر روی OK پروژه دارای رمز عبور می-شود. رمز عبور را می‌توانیم بین تعداد محدودی کاراکتر (این تعداد به نوع PLC وابسته است) از حروف انگلیسی، اعداد و بعضی علائم مانند "_" انتخاب کنیم. رمز عبور به بزرگ و کوچک بودن حروف حساس است.



شکل ۱۲-۸۴ ایجاد رمز عبور پروژه

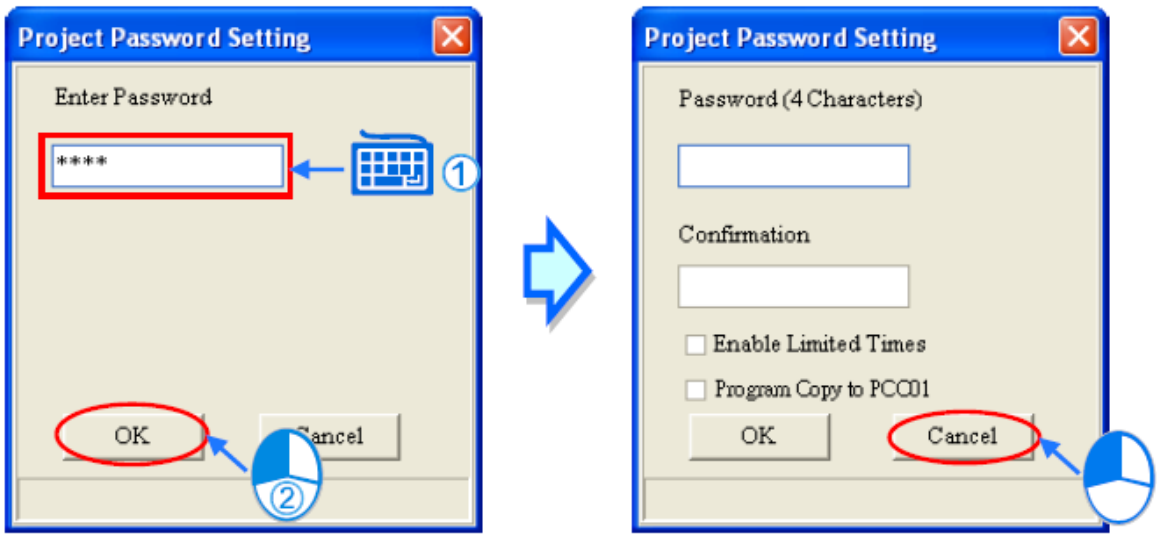
در صورتی که بخواهیم تعداد حدس‌های ممکن برای وارد کردن رمز عبور را محدود کنیم، می‌توانیم گزینه Enable Limited Times را فعال کنیم و در جلوی آن تعداد دفعات محدودیت را وارد کنیم. در

صورتی که کاربر بخواهد برنامه را به DVPPCC01[1] نیز کپی کند می‌تواند گزینه Program Copy to PCC01 را فعال کند. در نهایت نیز با کلیک بر روی OK تغییرات اعمال می‌شود.



شکل ۱۲-۸۵ تنظیمات مربوط به ایجاد رمز عبور پروژه

در صورتی که پروژه با رمز عبور محافظت شده باشد در پنجره باز شده محلی برای وارد کردن رمز عبور پروژه وجود دارد که با وارد کردن آن‌ها و کلیک بر روی OK رمز عبور پروژه حذف می‌شود.

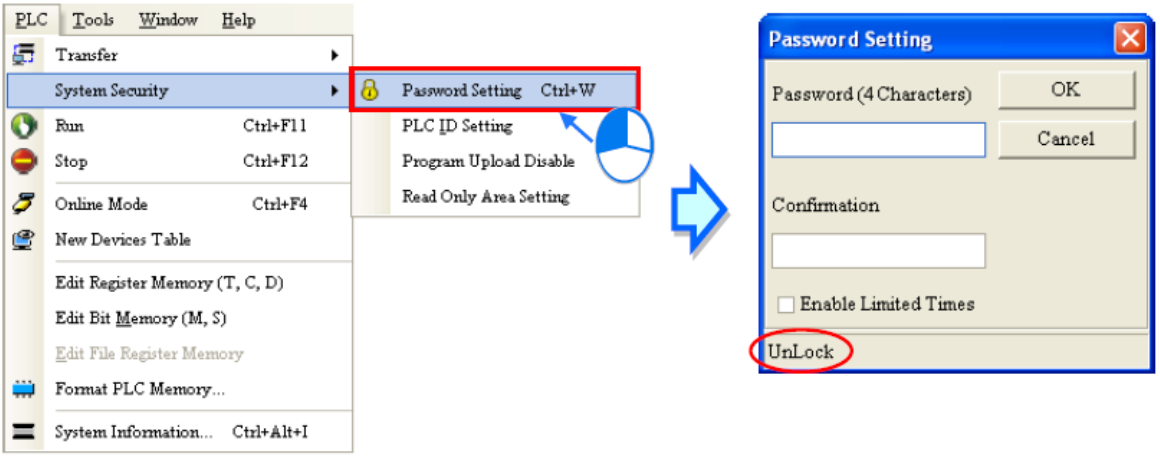


شکل ۱۲-۸۶ حذف رمز عبور پروژه

[1] DVPPCC01 کارت حافظه پشتیبان PLCهای سری DVP است. در صورتی که گزینه Program Copy to PCC01 در پنجره Project Password Setting فعال شود. با استفاده از این کارت می‌توان از برنامه‌هایی که در PLC دانلود می‌شود پشتیبان تهیه کرد.

۱۲-۳-۲-۲- تنظیم رمز عبور PLC

قبل از تنظیم رمز عبور PLC باید از ارتباط آن با ISPSoft مطمئن شد. در سربرگ PLC و قسمت System Security می‌توان بر روی Password Setting کلیک کرد تا صفحه تنظیم رمز عبور PLC باز شود. در صورت ارتباط صحیح بین PLC و ISPSoft، نرم افزار داده‌های PLC را می‌خواند و پنجره Password Setting ظاهر می‌شود. جمله‌ی زیر این صفحه نشان می‌دهد که آیا این PLC با رمز عبور محافظت شده است و یا خیر.



شکل ۱۲-۸۷ تنظیم رمز عبور PLC

در صورتی که PLC با رمز عبور محافظت نشده باشد در پنجره باز شده دو محل برای وارد کردن رمز عبور PLC و تایید آن وجود دارد که با وارد کردن آن‌ها و کلیک بر روی OK سیستم PLC دارای رمز عبور می‌شود. رمز عبور را می‌توانیم بین تعداد محدودی کاراکتر (این تعداد به نوع PLC وابسته است) از حروف انگلیسی، اعداد و بعضی علائم مانند "_" انتخاب کنیم. رمز عبور به بزرگ و کوچک بودن حروف حساس است. همچنین در صورتی که بخواهیم تعداد حدس‌های ممکن برای وارد کردن رمز عبور را محدود کنیم، می‌توانیم گزینه Enable Limited Times را فعال کنیم و در جلوی آن تعداد دفعات محدودیت را وارد کنیم.

در صورتی که PLC با رمز عبور محافظت شده باشد در پنجره باز شده محلی برای وارد کردن رمز عبور PLC وجود دارد که با وارد کردن آن‌ها و کلیک بر روی OK رمز عبور PLC حذف می‌شود.

شکل ۱۲-۸۸ ایجاد رمز PLC

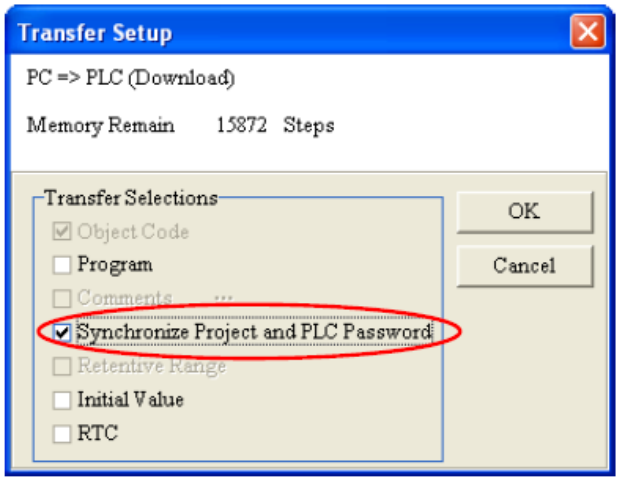
۱۲-۳-۲-۳- یکسان سازی رمز عبور پروژه و PLC

در حین بارگزاری و استخراج برنامه از PLC می‌توان رمز عبور برنامه و PLC را یکسان کرد.

- بارگزاری پروژه

پس از انتخاب آیکون  و باز شدن صفحه Transfer Setup با انتخاب گزینه Synchronize Project and PLC Password رمز عبور PLC پس از بارگزاری همانند رمز عبور پروژه خواهد شد.

در صفحه Transfer Setup با انتخاب گزینه Synchronize Project and PLC Password در صورتی که برای پروژه رمز عبوری تعیین نشده باشد، سیستم از کاربر می‌خواهد که برای پروژه رمز عبوری تعیین کند.

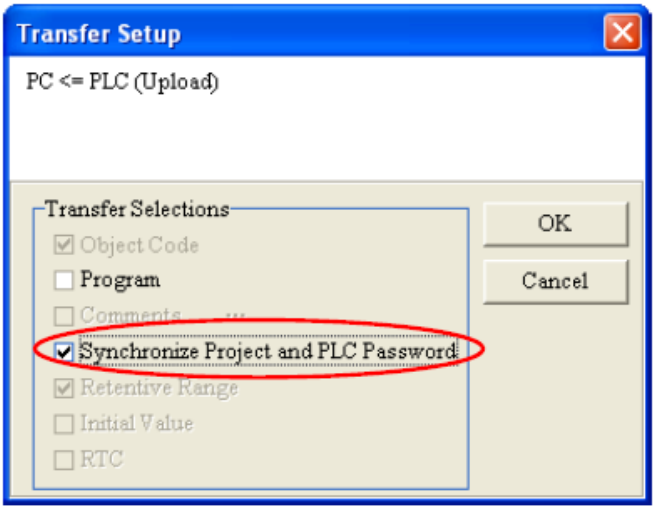


شکل ۱۲-۸۹ تبدیل رمز عبور پروژه به رمز عبور PLC

- استخراج پروژه

پس از انتخاب آیکون و باز شدن صفحه Transfer Setup با انتخاب گزینه Synchronize Project and PLC Password رمز عبور پروژه پس از استخراج همانند رمز عبور PLC خواهد شد. در صورتی که برروی PLC رمز عبوری تعبیه نشده باشد، رمز عبوری در پروژه استخراج نخواهد شد. همچنین چون از PLCهای AH500 نمی‌توان به صورت مستقل برنامه را استخراج کرد، این ویژگی برای آن‌ها بلااستفاده است.

در صورتی که در صفحه Transfer Setup گزینه Program انتخاب شده باشد، گزینه Synchronize Project and PLC Password قابل انتخاب نخواهد بود و رمز عبور پروژه تغییری نخواهد کرد.

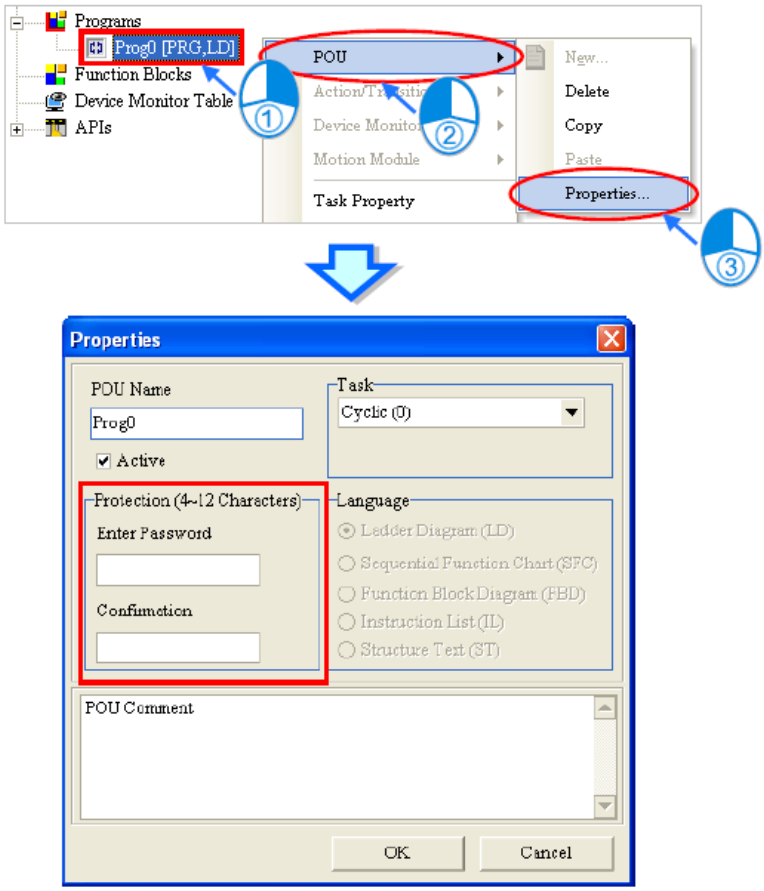


شکل ۱۲-۹۰ تبدیل رمز عبور PLC به رمز عبور پروژه

۱۲-۳-۳- رمز عبور POU

در صورتی که کاربر بخواهد از کُد داخل POU و یا تکنیک‌های به کار رفته در آن محافظت کند، می‌تواند بر روی آن رمز عبور تعبیه کند. در این صورت برای باز کردن POU حتما باید آن رمز عبور به طور صحیح وارد شود.

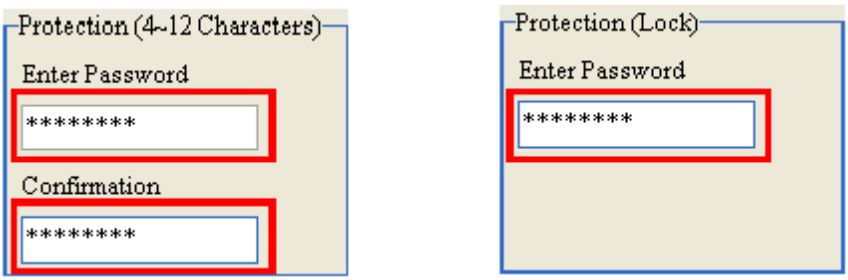
برای ایجاد رمز عبور برای POU، بر روی POU مورد نظر در بخش مدیریت پروژه کلیک راست کرده و در قسمت POU بر روی Properties کلیک می‌کنیم. در پنجره باز شده قسمت Protection که مرتبط با رمز POU است، می‌توانیم متوجه شویم آیا POU رمز عبور دارد و یا خیر.



شکل ۱۲-۹۱ تنظیم رمز عبور POU

در صورتی که POU با رمز عبور محافظت نشده باشد در بخش Protection دو محل برای وارد کردن رمز عبور POU و تایید آن وجود دارد که با وارد کردن آن‌ها و کلیک بر روی OK سیستم POU دارای رمز عبور می‌شود. رمز عبور را می‌توانیم بین تعداد محدودی کاراکتر از حروف انگلیسی، اعداد و بعضی علائم مانند "_" انتخاب کنیم. رمز عبور به بزرگ و کوچک بودن حروف حساس است.

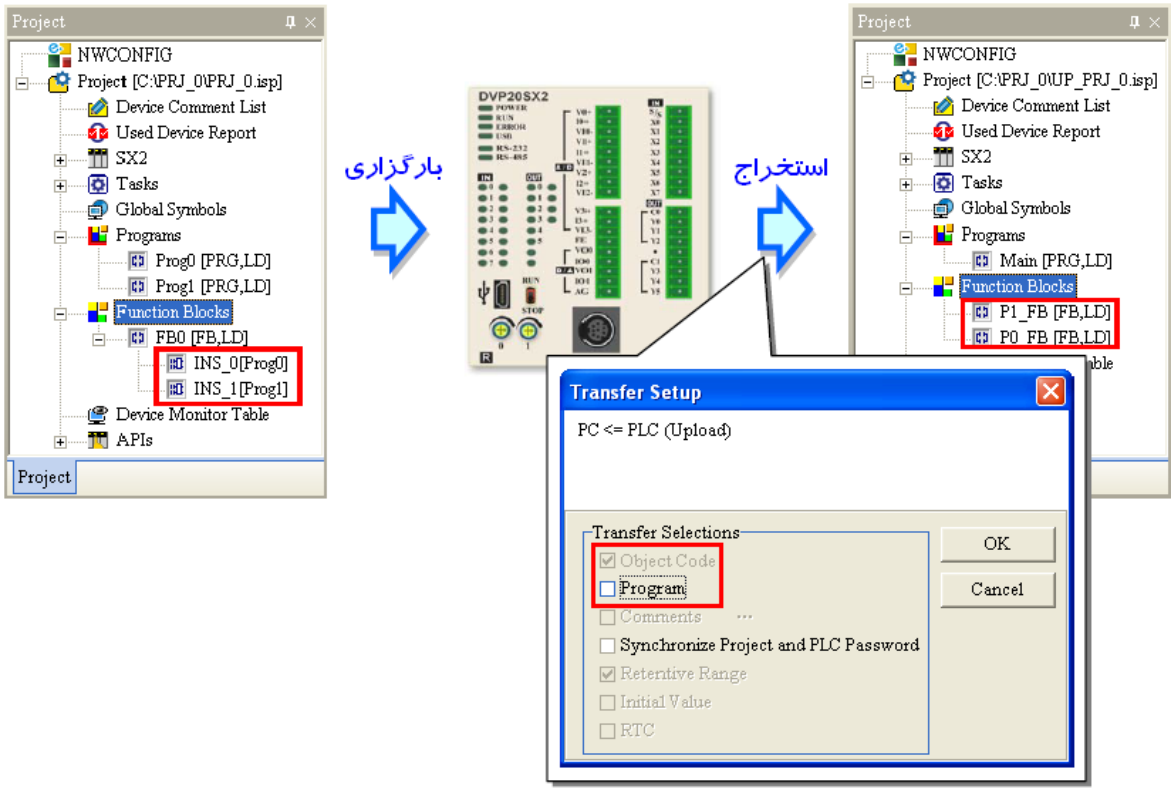
در صورتی که POU با رمز عبور محافظت شده باشد در بخش Protection محلی برای وارد کردن رمز عبور POU وجود دارد که با وارد کردن آن‌ها و کلیک بر روی OK رمز عبور POU حذف می‌شود.



شکل ۱۲-۹۲ ایجاد و حذف رمز عبور POU

۱۲-۳-۴- رمز عبور Subroutin

در PLCهای سری DVP برنامه‌های کامپایل شده ممکن است شامل بخش‌های مختلفی مانند برنامه اصلی و برنامه های جانبی (بلوک‌ها) و وقفه‌ها باشد. در هنگام استخراج برنامه از PLC در صورتی که گزینه Program انتخاب نشده باشد و Object Code انتخاب شده باشد، به عنوان توابع بلوکی استخراج می-شوند. از آنجایی رمز عبور POU فقط برای کُد اصلی است و در برنامه استخراج شده اعمال نمی‌شود، با این ترفند امکان دسترسی به برنامه وجود دارد.

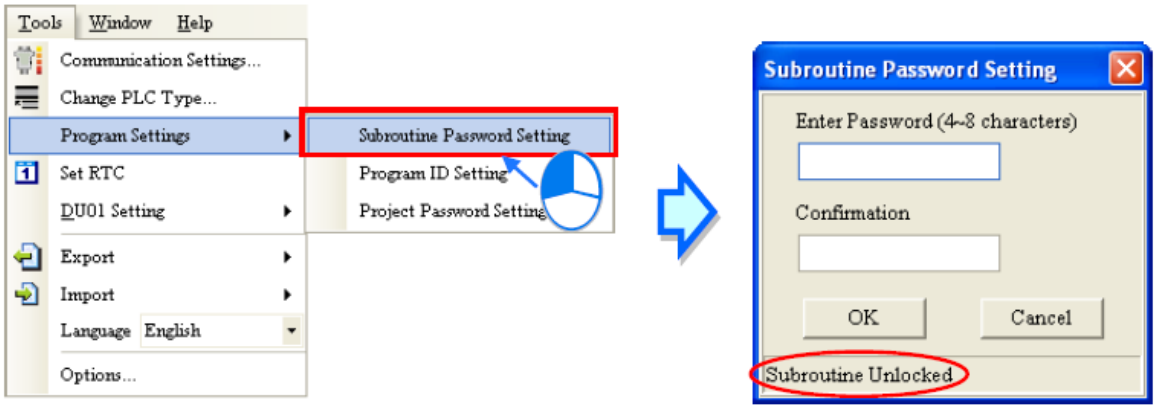


شکل ۱۲-۹۳ امکان دسترسی به برنامه ها پس از استخراج از PLC

همانطور که در بالا گفته شد، امکان دسترسی به کُدهای برنامه به صورت فوق وجود دارد، برای جلوگیری از این موضوع می‌توان از رمز عبور Subroutine قبل از بارگزاری برنامه استفاده کرد، در این صورت برنامه حتی پس از استخراج از PLC نیز محافظت شده خواهد ماند و برای دسترسی به آن رمز عبور لازم خواهد بود.

شکل ۱۲-۹۴ رمز عبور Subroutine

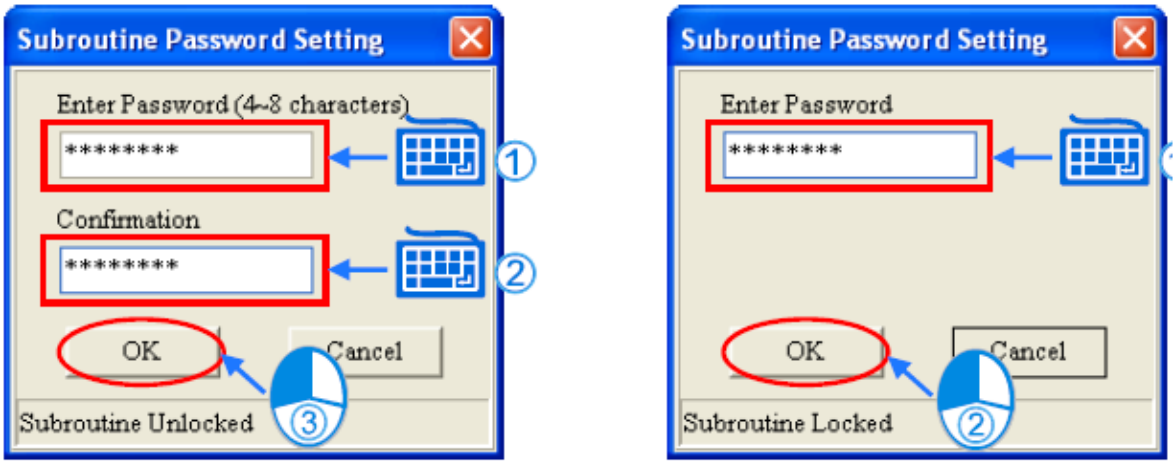
برای تنظیمات رمز عبور Subroutine می‌توان در سربرگ Tools و در قسمت Program Settings بر روی Subroutine Password کلیک کرد تا صفحه تنظیم رمز عبور Subroutine باز شود. جمله‌ی زیر این صفحه نشان می‌دهد که آیا این Subroutine با رمز عبور محافظت شده است و یا خیر.



شکل ۱۲-۹۵ تنظیم رمز عبور Subroutine

در صورتی که Subroutine با رمز عبور محافظت نشده باشد، در زیر صفحه عبارت "Subroutine Unlocked" نمایش داده می‌شود و دو محل برای وارد کردن رمز عبور Subroutine و تایید آن وجود دارد که با وارد کردن آن‌ها و کلیک بر روی OK سیستم Subroutine دارای رمز عبور می‌شود. رمز عبور را می‌توانیم بین تعداد محدودی کاراکتر (این تعداد به نوع PLC وابسته است) از حروف انگلیسی، اعداد و بعضی علائم مانند "_" انتخاب کنیم. رمز عبور به بزرگ و کوچک بودن حروف حساس است.

در صورتی که Subroutine با رمز عبور محافظت شده باشد، در زیر صفحه عبارت "Subroutine Locked" نمایش داده می‌شود و محلی برای وارد کردن رمز عبور Subroutine وجود دارد که با وارد کردن آن‌ها و کلیک بر روی OK رمز عبور Subroutine حذف می‌شود.



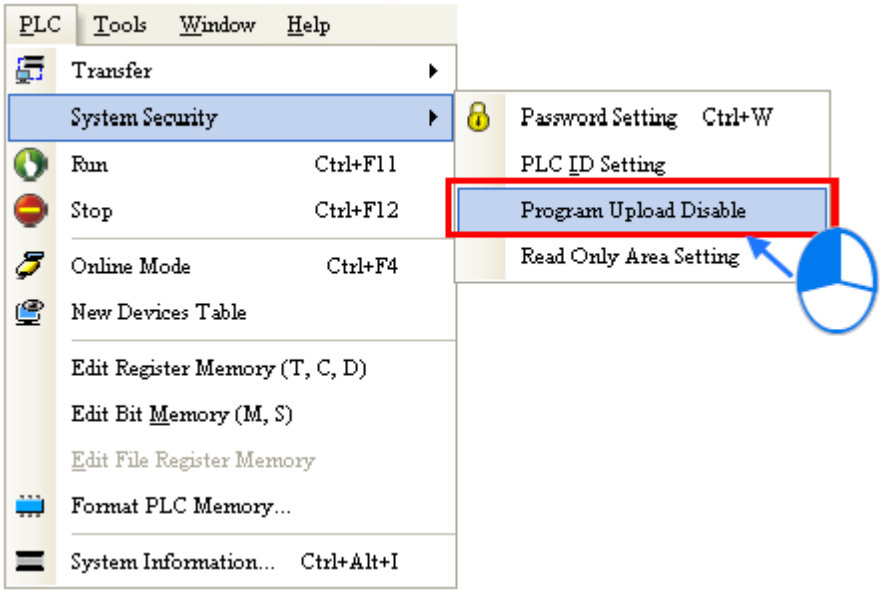
شکل ۱۲-۹۶ ایجاد و حذف رمز عبور Subroutine

۱۲-۳-۵- روش های دیگر محافظت و رمز گزاری

۱۲-۳-۵-۱- غیرفعال کردن استخراج برنامه از PLCهای سری DVP

می‌توان امکان استخراج برنامه از PLCهای سری DVP را حذف کرد و در این صورت امکان استخراج تنها در صورت ریست کردن PLC به حالت پیش فرض امکان پذیر است. برای فعال کردن این گزینه لازم است در صورتی که PLC دارای رمز عبور است، آن را وارد کنیم تا PLC از حالت قفل درآید. بعد از غیرفعال کردن استخراج برنامه از PLCهای سری DVP چون دیگر امکان دسترسی به داده‌های PLC وجود ندارد، پس رمزگزاری برای PLC لزومی نخواهد داشت و دیگر نمی‌توان بر روی آن رمز گذاشت.

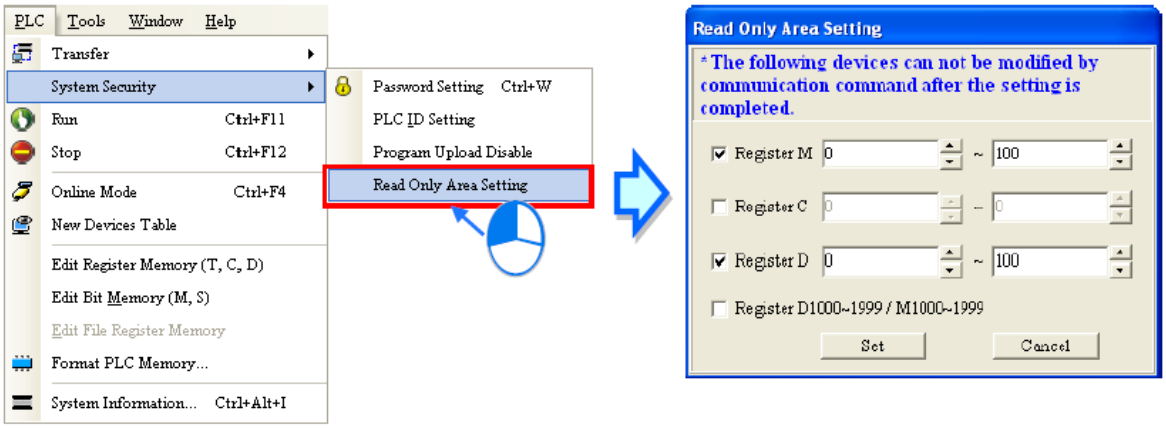
برای غیرفعال کردن استخراج برنامه از PLCهای سری DVP باید از سربرگ PLC، در قسمت System Security گزینه Program Upload Disable را انتخاب کرد. پس از غیرفعال کردن امکان استخراج برنامه از PLC دیگر از PLC مورد نظر نمی‌توان داده استخراج کرد.



شکل ۱۲-۹۷ غیرفعال کردن استخراج برنامه

۱۲-۳-۵-۲- تعیین محدوده فقط خواندنی حافظه ها در PLCهای سری DVP

در PLCهای سری DVP می‌توان محدوده‌ای از حافظه‌ها را تعیین کرد که دیگر آن‌ها را نمی‌توان از طریق دستورات ارتباطی[۱] (مثلا از طریق HMI) تغییر داد. برای تعیین این محدوده باید در سربرگ PLC، در قسمت System Security بر روی Read Only Area Setting کلیک کرد و در پنجره باز شده برای هر نوع حافظه محدوده "فقط خواندنی" را تعیین کرد. در صورتی که نیاز به تغییر حافظه‌های این محدوده وجود دارد، باید PLC را به تنظیمات پیشفرض ببریم.

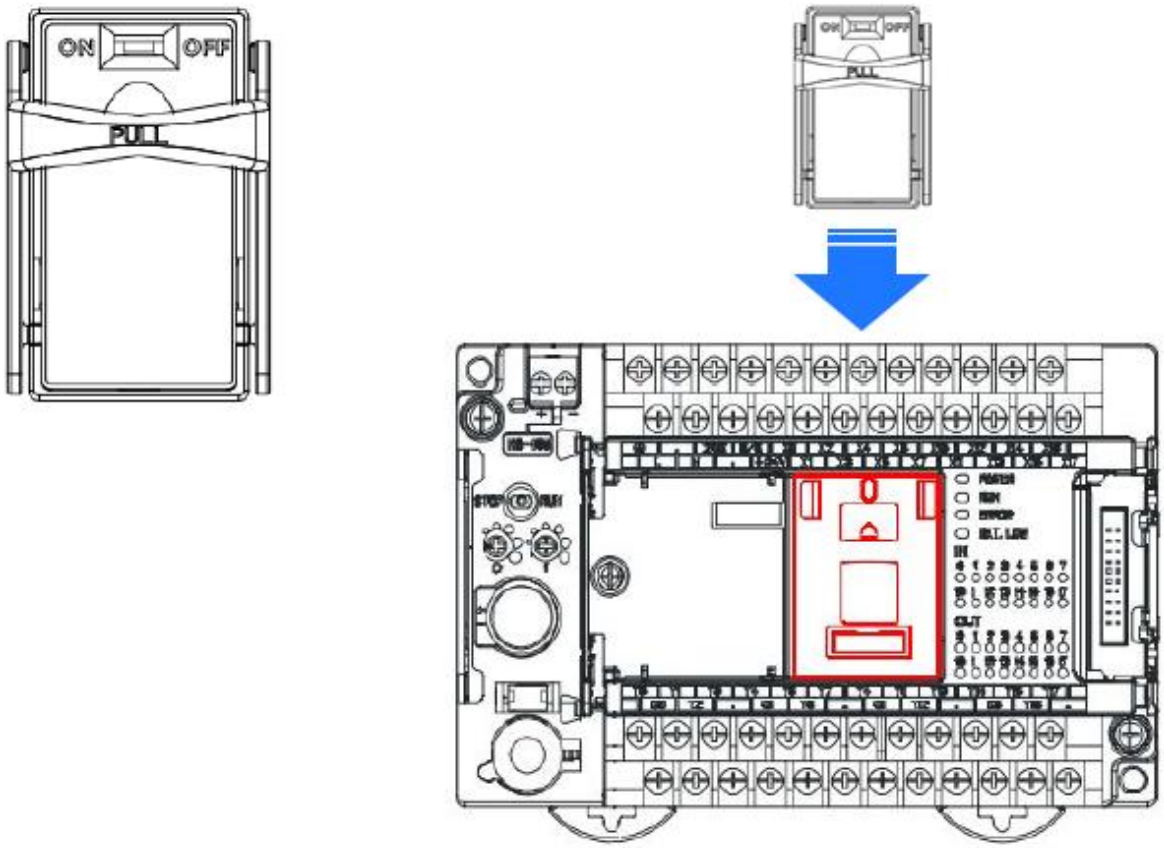


شکل ۱۲-۹۸ تعیین محدوده فقط خواندنی حافظه ها در PLCهای سری DVP

[۱] Communication Commands

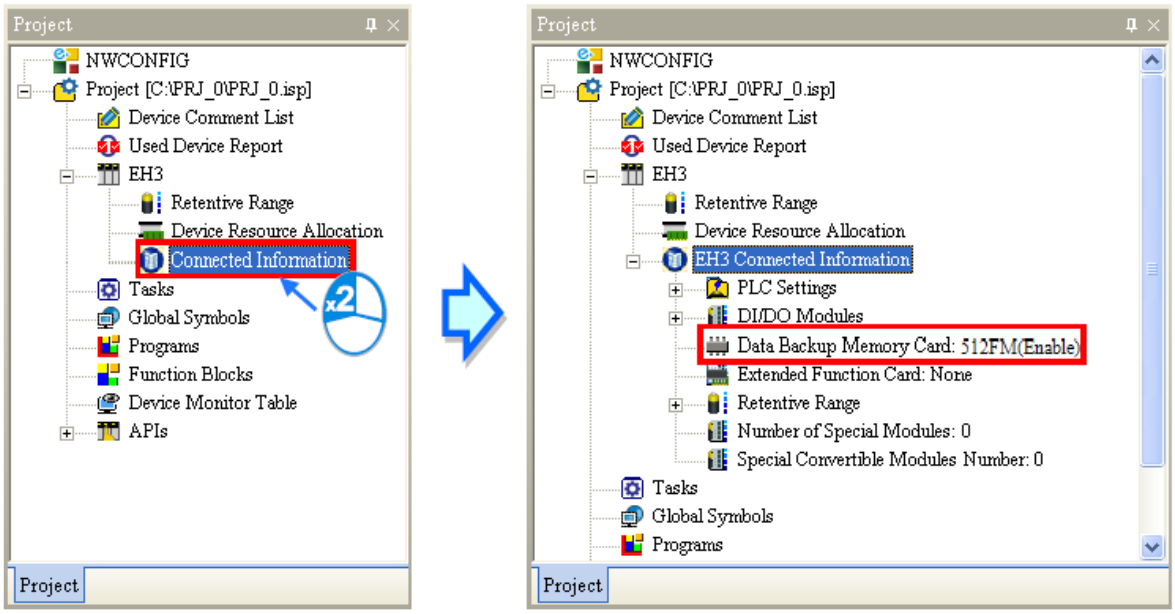
۱۲-۴-پشتیبان گیری از داده ها

PLCهای سری DVP-EH2 می‌توانند همراه با کارت حافظه پشتیبان DVP-256FM به کار روند (این کارت‌ها دارای پوشش شیری رنگ هستند). همچنین PLCهای سری DVP-EH3 می‌توانند با کارت حافظه پشتیبان DVP-512FM به کار روند (دارای پوشش سیاه رنگ).



شکل ۱۲-۹۹ پشتیبان گیری از داده های PLCهای DVP

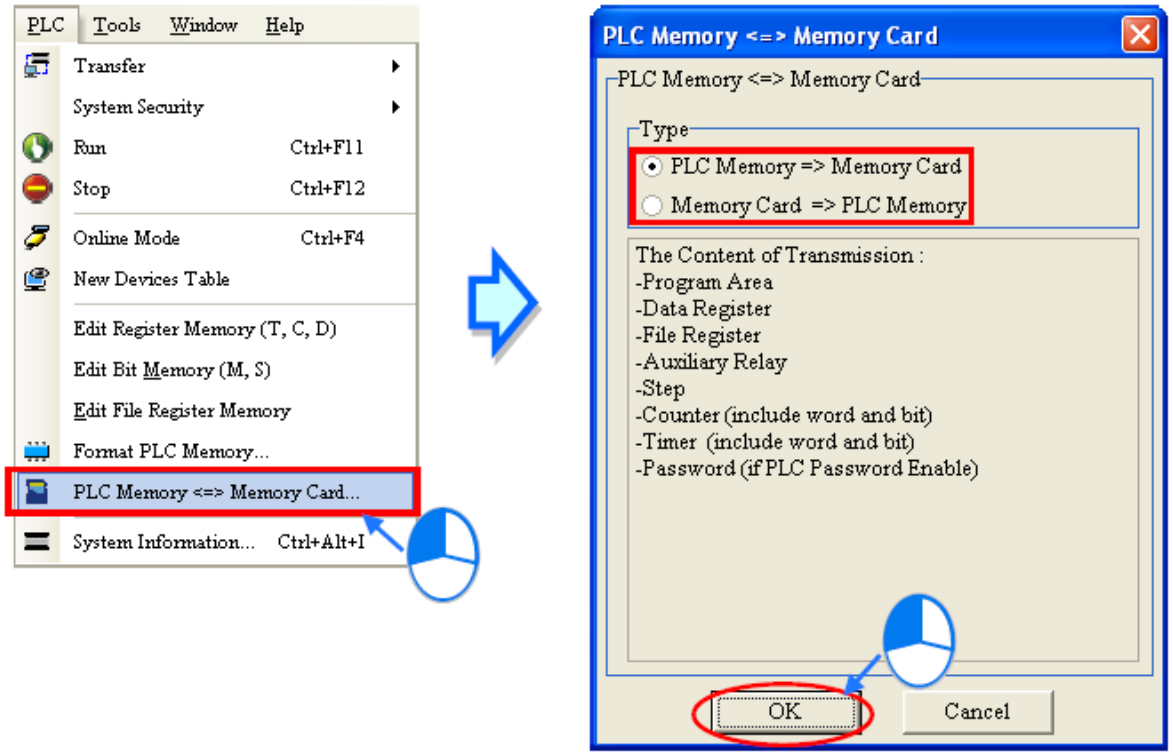
پس از نصب کارت حافظه پشتیبان، متصل کردن PLC به برق و برقراری ارتباط آن با ISPSoft، می-توان در بخش مدیریت پروژه در زیرشاخه مدل PLC دوبار بر روی Connected Information کلیک کرد. در صورتی که ارتباط PLC با ISPSoft به صورت مناسبی برقرار شده باشد گزینه Data Backup Memory Card نشان‌دهنده فعال بودن عملکرد کارت حافظه خواهد بود.



شکل ۱۲-۱۰۰ محل نمایش اتصال درست کارت حافظه PLCهای سری DVP

۱۲-۴-۱- استفاده از کارت حافظه پشتیبان

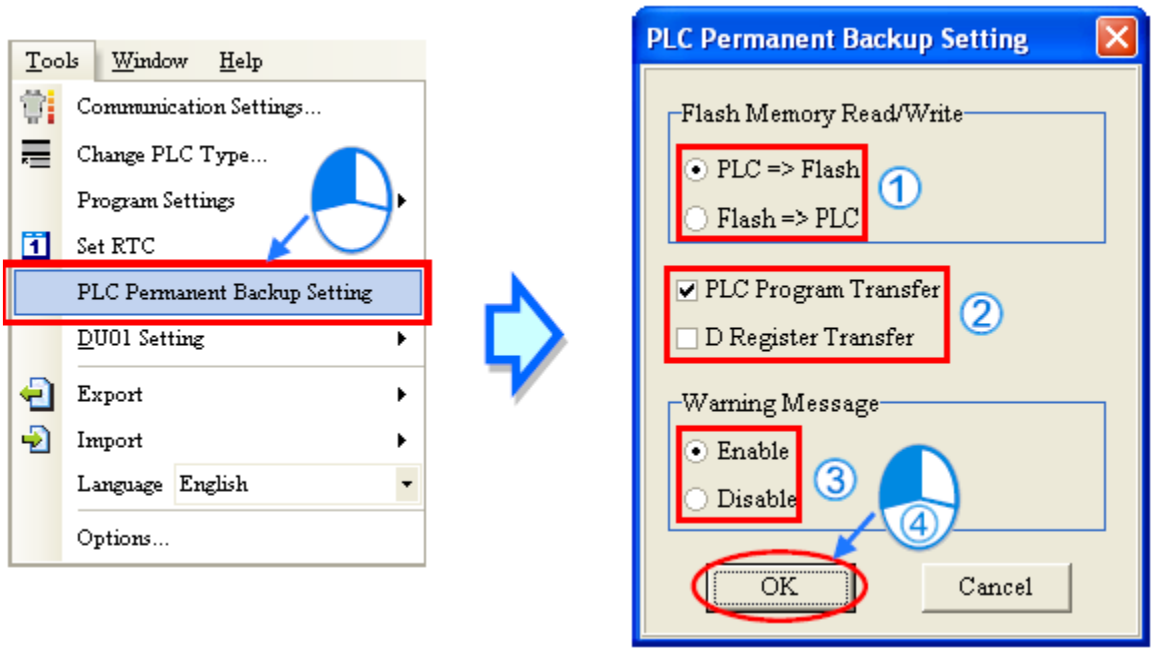
پس از اطمینان از ارتباط PLC با ISPSoft می‌توان در سربرگ PLC با انتخاب PLC Memory <=> Memory Card ، جهت انتقال داده را در پنجره باز شده تعیین و سپس تایید کرد.



شکل ۱۲-۱۰۱ نحوه تبادل داده بین PLCهای سری DVP و کارت حافظه

۱۲-۴-۲- پشتیبان گیری دائم

PLC های سری DVP-EH2/DVP-EH2-L/DVP-EH3/DVP-EH3-L/DVP-SV/DVP-SV2 دارای کارت حافظه هستند که برای پشتیبان گیری از داده‌های PLC مورد استفاده قرار می‌گیرد. برای تنظیم کارت حافظه ابتدا باید از ارتباط PLC با ISPSoft مطمئن شد، سپس در سربرگ Tools بر روی PLC Permanent Backup Setting کلیک می‌کنیم. در پنجره ظاهر شده کاربر باید جهت انتقال داه‌ها را در بخش Flash Memory Read/Write section تعیین کند. همچنین نوع محتوای قابل انتقال (برنامه و یا حافظه‌های D) و در قسمت Warning Message فعال و یا غیرفعال بودن پیام‌های هشدار[1] را مشخص کند. در نهایت با کلیک بر روی OK تنظیمات را اعمال می‌کنیم.



شکل ۱۲-۱۰۲ تنظیم کارت‌های حافظه دائم در PLCهای DVP

۱۲-۴-۳- CARD Utility (برای PLCهای سری AH500)

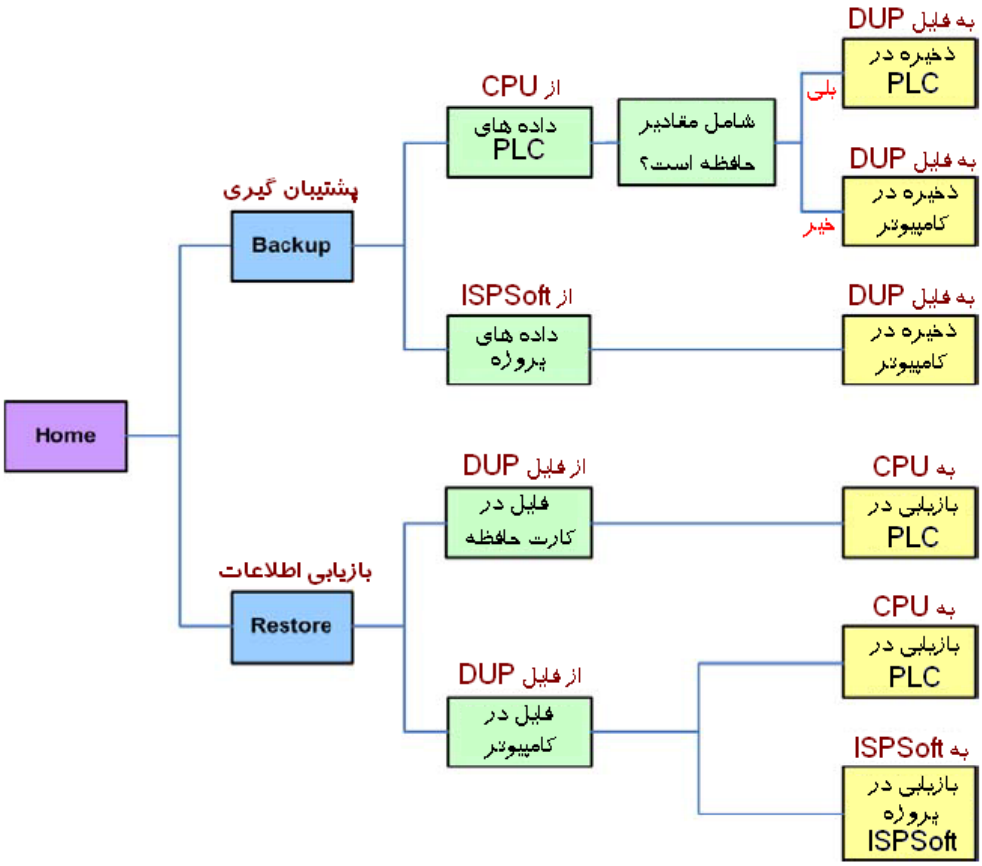
در PLCهای سری AH500 محلی برای کارت‌های حافظه SD وجود دارد که با استفاده از آن (با قرار دادن کارت حافظه در آن) می‌توان از اطلاعات ماژول CPU پشتیبان تهیه کرد. کاربر با استفاده از ابزار Wizardمی‌تواند از ماژول‌های سری AH500 و یا پروژه‌های ISPSoft پشتیبان تهیه کند. می‌توان از کُد

[1] در صورت تمام شدن باتری PLC و از دست رفتن داده‌های PLC بازیابی داده‌های پشتیبان گرفته شده به صورت اتوماتیک صورت می‌گیرد. در صورت فعال بودن پیام هشدار، در صورت بازیابی داده‌ها به صورت اتوماتیک پیام مربوط به آن نیز نمایش داده می‌شود.

برنامه، پارامترها، تنظیمات سخت افزاری، مقادیر حافظه‌ها و تنظیمات شبکه در PLCهای AH500 و یا پروژه‌های ISPSoft پشتیبان گرفت. هنگام پشتیبان گیری از شبکه، صرفا تنظیمات مرتبط با PLCهای AH500 و جدول مسیریابی و Ether Link جمع آوری می‌شوند.

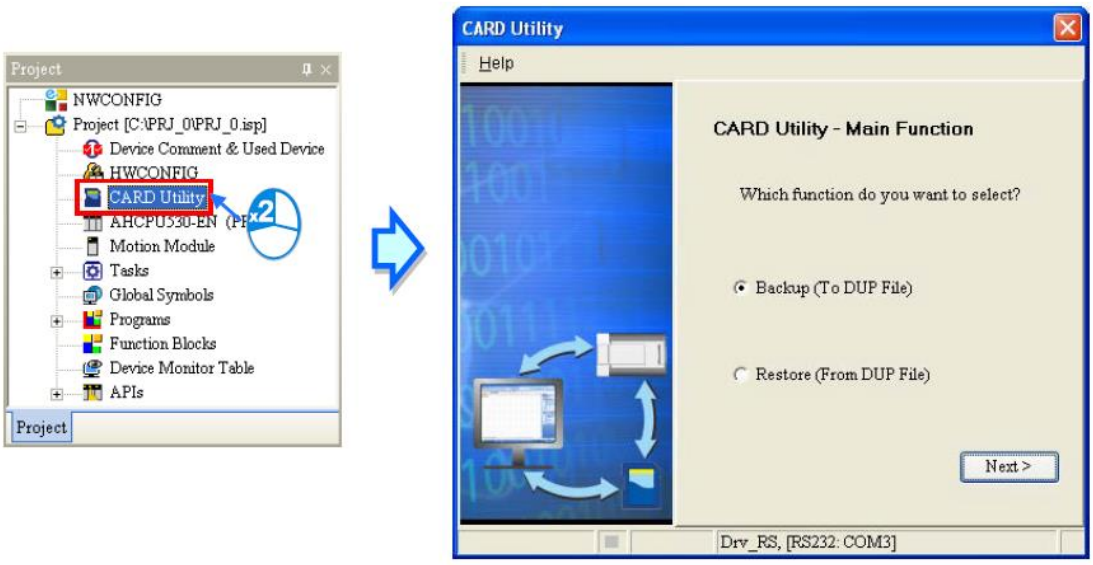
عملکردهایی که به وسیله CARD UTILITY پشتیبانی می‌شوند به صورت زیر است.

- می‌توان داده‌های PLCهای AH500 را به صورت فایل پشتیبان (با پسوند dup) در کامپیوتر و یا کارت حافظه روی PLC ذخیره کرد. تصمیم گیری در مورد محل پشتیبان گیری به وسیله کاربر صورت می‌گیرد.
- در صورتی که کاربر از پروژه ISPSoft پشتیبان تهیه کند (فایل با پسوند dup)، آن را فقط در کامپیوتر می‌تواند ذخیره کند و مقادیر حافظه‌های CPU نیز ذخیره نخواهد شد.
- کاربر می‌تواند، فایل‌های پشتیبان گرفته شده داخل کارت حافظه را به CPU منتقل کند.
- کاربر می‌تواند فایل‌های پشتیبان گرفته شده داخل کامپیوتر را بعد از اتصال به PLC به CPU منتقل کند یا اینکه فایل را در پروژه ISPSoft بازیابی کند. در صورتی که کاربر بازیابی فایل پشتیبان را در پروژه ISPSoft انجام دهد، سیستم به صورت خودکار مقادیر درون حافظه‌ها و تنظیمات سخت افزاری را بازیابی نمی‌کند.



شکل ۱۲-۱۰۳ پشتیبان گیری و بازیابی اطلاعات در AH500

برای انجام عملیات فوق باید در بخش مدیریت پروژه بر روی CARD Utility دوبار کلیک کرد تا صفحه آن ظاهر شود. حال می‌توان عملیات پشتیبان گیری و یا بازیابی داده‌ها را انجام داد.

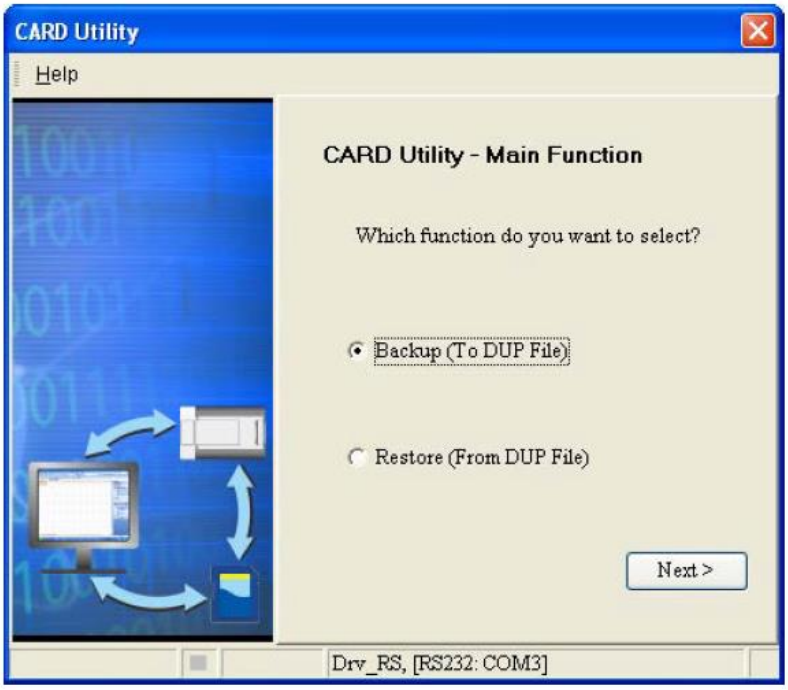


شکل ۱۲-۱۰۴ تنظیمات CARD Utility

۱۲-۴-۳-۱- تهیه پشتیبان

در صورتی که مبدا و یا مقصد پشتیبان گیری ماژول CPU و یا کارت حافظه روی PLCهای سری AH500 باشد، کاربر باید از ارتباط بینISPSoft و PLC مطمئن شود. سپس مراحل زیر را طی کند.

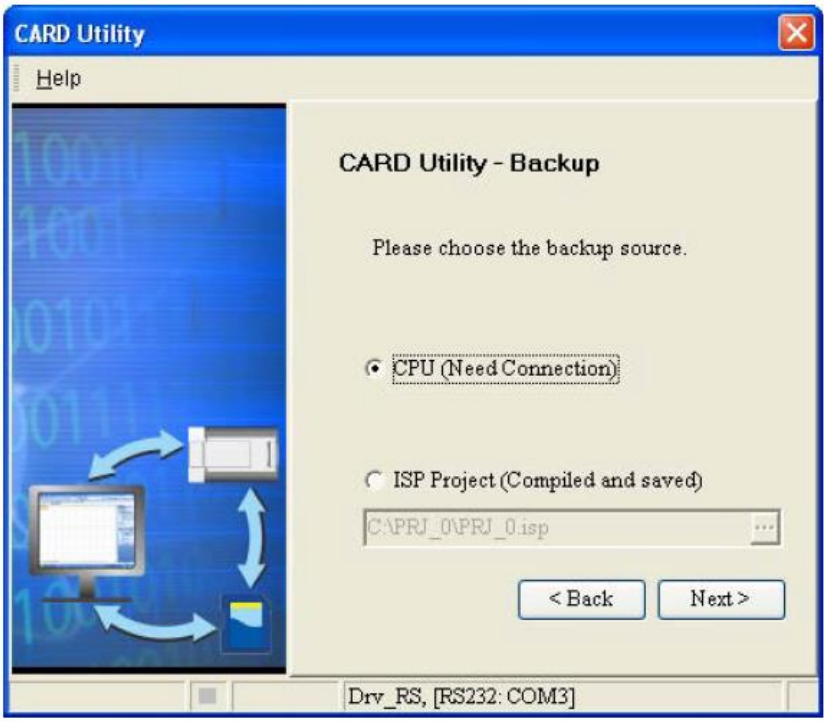
۱- در صفحه CARD Utility گزینه Backup (To DUP File) را انتخاب و بر روی Next کلیک کنید.



شکل ۱۲-۱۰۵ تنظیمات CARD Utility – پشتیبان گیری – قسمت ۱

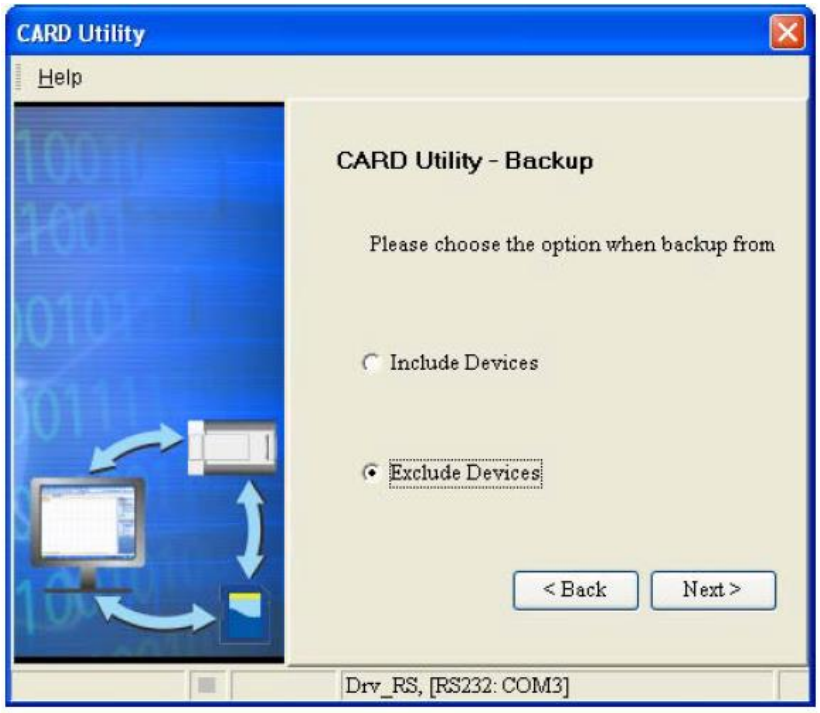
۲- منبع پشتیبان گیری را انتخاب کنید.

در صورتی ISP Project (Compiled and saved) انتخاب شود، نیاز است، تا آدرس پروژه مورد نظرمان برای پشتیبان گیری با پسوند isp را تعیین کنیم. در صورتی که این پروژه در حال ویرایش باشد و کامپایل نشده باشد، پیامی مبتنی بر نیاز به کامپایل آن ظاهر می‌شود که ضرورت کامپایل آن را گوشزد می‌کند. در این حالت ابتدا لازم است پروژه را کامپایل و سپس از آن پشتیبان تهیه کنیم.



شکل ۱۲-۱۰۶ تنظیمات CARD Utility – پشتیبان گیری – قسمت ۲

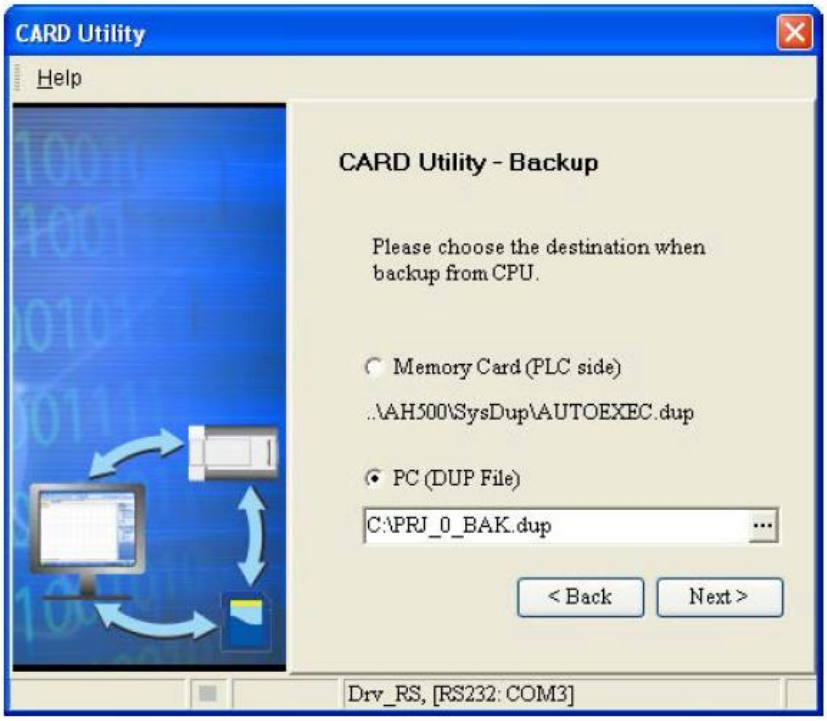
۳- در صورتی که در صفحه فوق گزینه CPU (Need Connection) انتخاب شده باشد، باید در مورد ذخیره شدن و یا نشدن مقادیر حافظه‌های CPU سری AH500 نیز تصمیم گیری کنیم، برای اینکار با انتخاب گزینه Include Devices از حافظه‌ها پشتیبان گرفته می‌شود و با انتخاب گزینه Exclude Devices از آن‌ها پشتیبان تهیه نمی‌شود.



شکل ۱۲-۱۰۷ تنظیمات CARD Utility – پشتیبان گیری – قسمت ۳

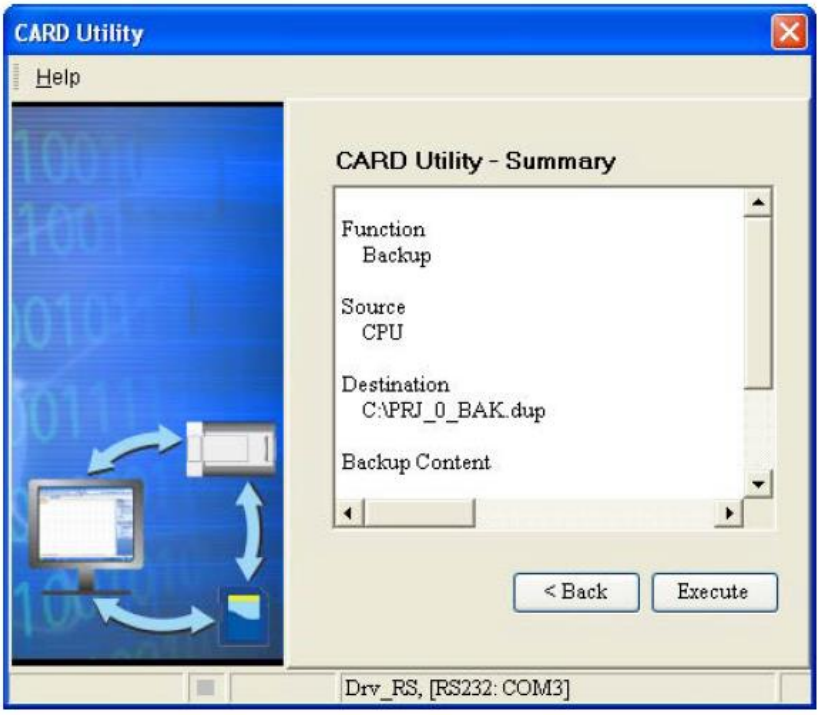
۴- در صفحه بعد باید مکان ذخیره سازی پشتیبان را مشخص کنیم. در صورتی که منبع پشتیبان پروژه ISPSoft باشد، حتما باید ذخیره سازی در کامپیوتر صورت بگیرد.

- اگر گزینه Memory Card (PLC Side) انتخاب شود، فایل پشتیبان با عنوان AUTOEXEC.dup ایجاد و مسیر ذخیره سازی آن به صورت "Root directory of the memory card\AH500\SysDup\AUTOEXEC.dup" خواهد بود.
- اگر گزینه **PC (DUP File)** انتخاب شود، نام و آدرس ذخیره سازی فایل پشتیبان را باید تعیین کرد.



شکل ۱۲-۱۰۸ تنظیمات CARD Utility – پشتیبان گیری – قسمت ۴

۵- کاربر در صفحه ظاهر شده این مرحله خلاصه ای از تنظیمات پشتیبان گیری را می‌بیند و در صورت تایید می‌تواند، بر روی Execute کلیک کند.

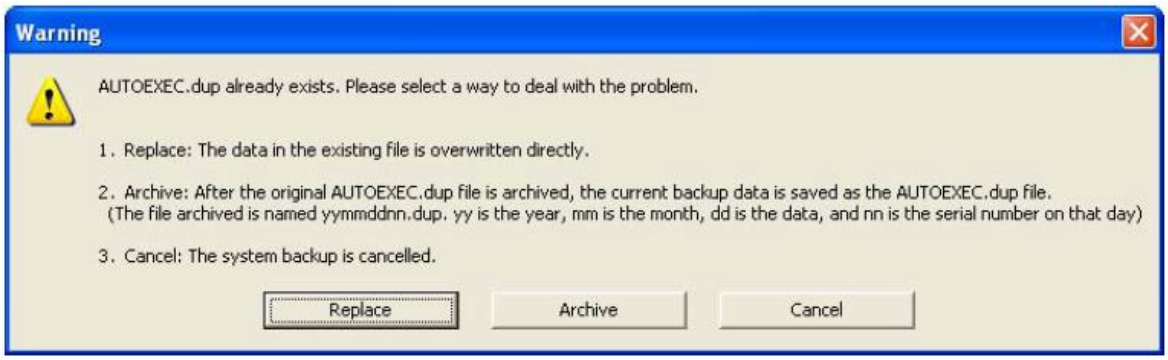


شکل ۱۲-۱۰۹ تنظیمات CARD Utility – پشتیبان گیری – قسمت ۵

در صورتی که بر روی کارت حافظه از قبل پشتیبانی ذخیره شده باشد، در صورت پشتیبان گیری دوباره، برنامه پیام هشداری می‌دهد که ما را از این موضوع مطلع می‌کند.

در صورتی که بخواهیم فایل قبلی حذف و فایل جدید جایگزین شود می‌توانیم گزینه Replace را انتخاب کنیم. در صورتی که بخواهیم فایل قبلی با نام دیگری (yymmddnn.dup که در آن yy سال، mm ماه، dd روز و nn شماره پشتیبان در آن روز را مشخص می‌کند) ذخیره شود و فایل جدید به عنوان پشتیبان اصلی ذخیره شود می‌توان گزینه Archive را انتخاب و در غیر این صورت برای لغو عملیات Cancel را انتخاب می‌کنیم.

حتی در صورت کلیک بر روی Cancel (حین پشتیبان گیری) برای جلوگیری از پشتیبان گیری از داده‌های PLC در کارت پشتیبان آن، پشتیبان گیری صورت می‌گیرد که به علت ناقص بودن فایل پشتیبان، باید آن را حذف کنیم.



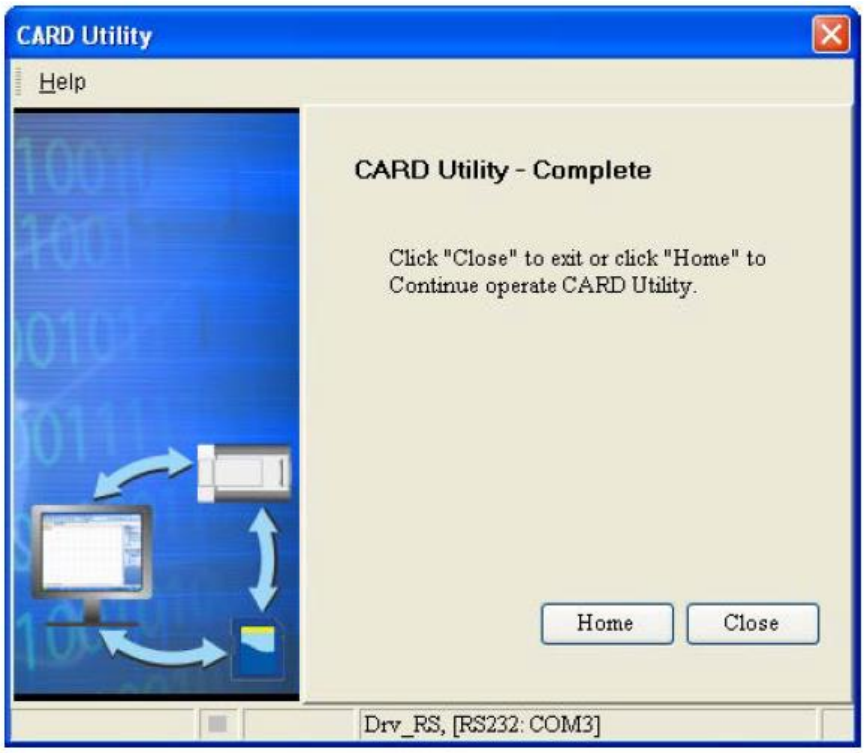
شکل ۱۲-۱۱۰ تنظیمات CARD Utility –هشدار در مورد وجود فایل پشتیبان قدیمی

در صورتی که داده‌های پشتیبان گرفته شده دارای رمز عبور باشند، رمز عبور آن‌ها نیز در فایل پشتیبان ذخیره می‌شود.

جدول ۱۲-۹: رمز عبور در زمان پشتیبان گیری

نوع پشتیبان گیری	شرح
CPU به کارت حافظه	داده‌هایی که پشتیبانی گرفته می‌شوند شامل شناسه PLC و رمز عبور PLC ماژول CPU نیز هستند.
CPU به کامپیوتر	ابتدا سیستم درخواست می‌کند که کاربر شناسه PLC و رمز عبور آن را وارد کند. در صورت صحیح بودن رمز عبور و شناسه PLC از داده‌ها پشتیبان گرفته می‌شود. داده‌هایی که پشتیبانی گرفته می‌شوند شامل شناسه PLC و رمز عبور PLC ماژول CPU نیز هستند.
پروژه ISPSoft به کامپیوتر	داده‌هایی که پشتیبانی گرفته می‌شوند شامل شناسه برنامه و رمز عبور پروژه ISPSoft نیز هستند.

۶- در انتها نیز در صورت موفقیت آمیز بودن صفحه CARD Utility – Complete ظاهر می‌شود.

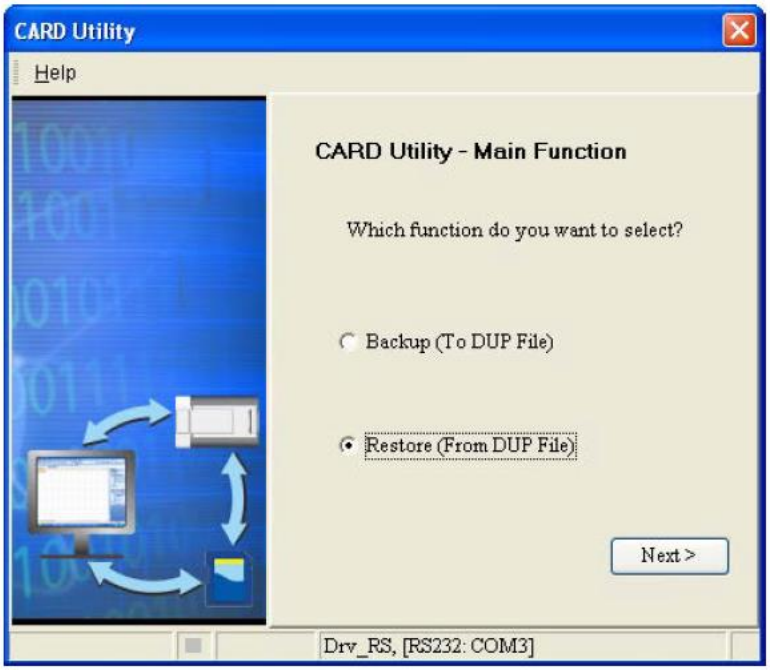


شکل ۱۲-۱۱۱ تنظیمات CARD Utility – پشتیبان گیری – قسمت ۶

۱۲-۴-۳-۲- بازیابی پشتیبان

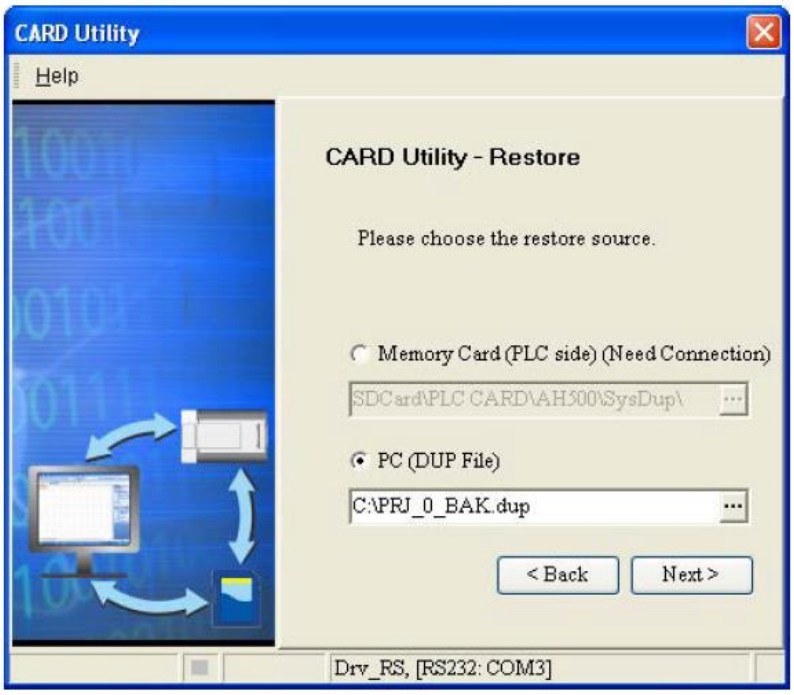
در صورتی که مبدا و یا مقصد بازیابی پشتیبان، ماژول CPU و یا کارت حافظه روی PLCهای سری AH500 باشد، کاربر باید از ارتباط بینISPSoft و PLC مطمئن شود. سپس مراحل زیر را طی کند.

۱- در صفحه CARD Utility گزینه Restore (From DUP File) را انتخاب و بر روی Next کلیک کنید.



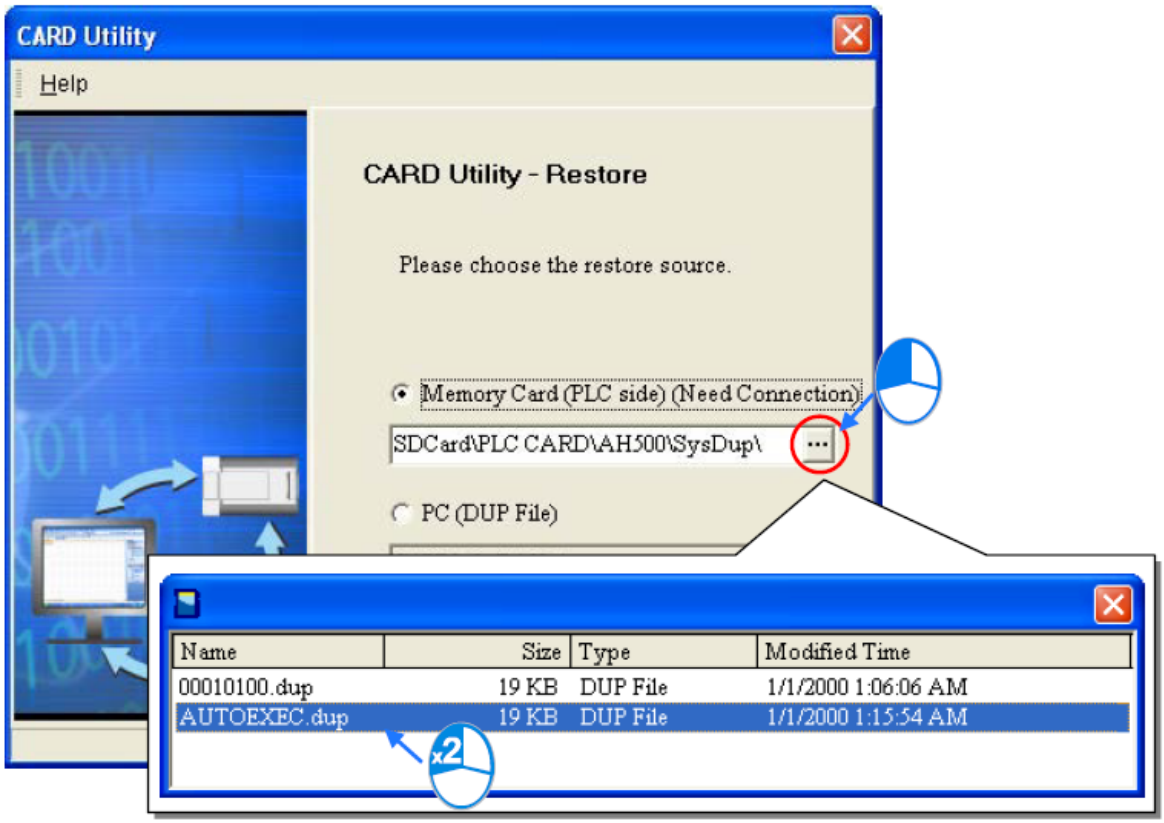
شکل ۱۲-۱۱۲ تنظیمات CARD Utility – بازیابی پشتیبان– قسمت ۱

۲- منبع بازیابی پشتیبان را انتخاب کنید.



شکل ۱۲-۱۱۳ تنظیمات CARD Utility – بازیابی پشتیبان– قسمت ۲

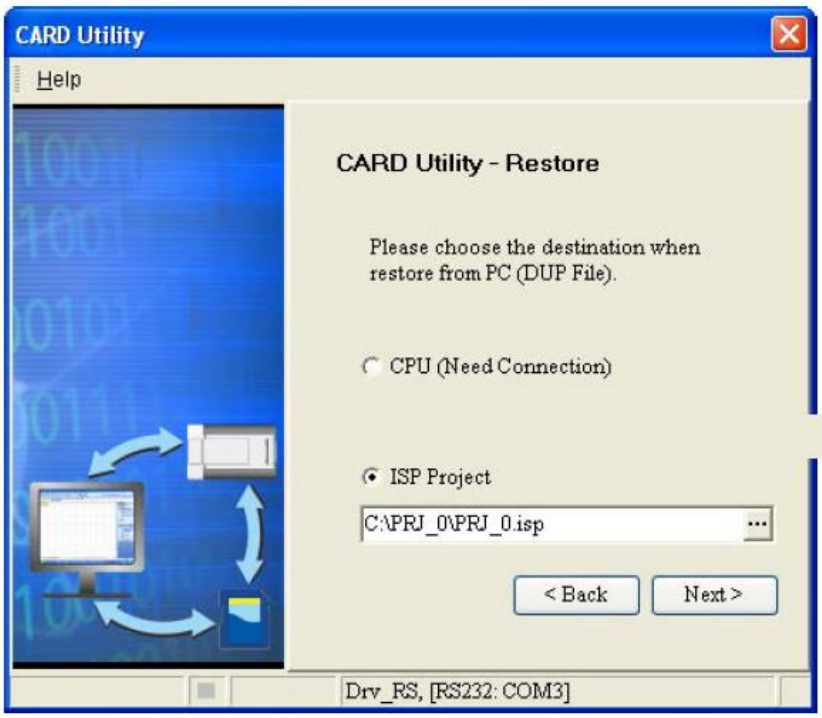
در صورتی Memory Card (PLC side) (Need Connection) انتخاب شود، لیستی از فایل‌های پشتیبان داخل حافظه نمایش داده می‌شود که با دوبار کلیک بر روی منبع مورد نظرمان می‌توانیم آن را انتخاب کنیم.



شکل ۱۲-۱۱۴ تنظیمات CARD Utility – بازیابی پشتیبان– قسمت ۳

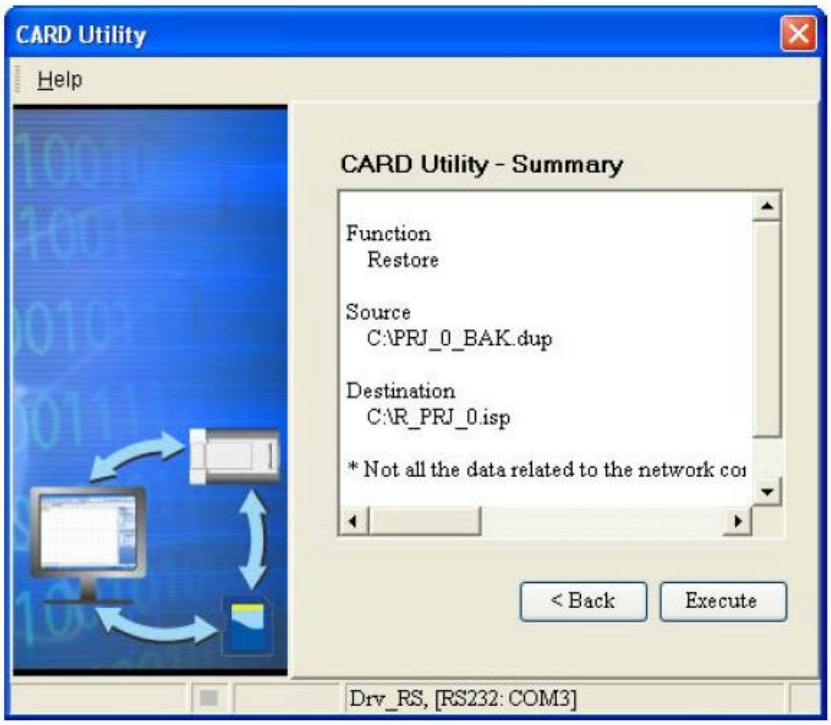
۳- در صفحه بعد باید مکان مقصد بازیابی را مشخص کنیم.

- در صورت انتخاب گزینه CPU (Need Connection) بازیابی بر روی CPU انجام می‌شود. در صورتی که فایل پشتیبان در حافظه ذخیره شده باشد، مقصد بازیابی حتما باید CPU باشد و نمی‌توان آن را در پروژه‌های ISPSoft بازیابی کرد.
- اگر گزینه ISP Project انتخاب شود، پروژه‌ای با نام و آدرسی که کاربر تعیین می‌کند ایجاد می‌شود.



شکل ۱۲-۱۱۵ تنظیمات CARD Utility – بازیابی پشتیبان– قسمت ۴

۴- کاربر در صفحه ظاهر شده در این مرحله خلاصه ای از تنظیمات بازیابی پشتیبان را می‌بیند و در صورت تایید می‌تواند، بر روی Execute کلیک کند.



شکل ۱۲-۱۱۶ تنظیمات CARD Utility – بازیابی پشتیبان– قسمت ۵

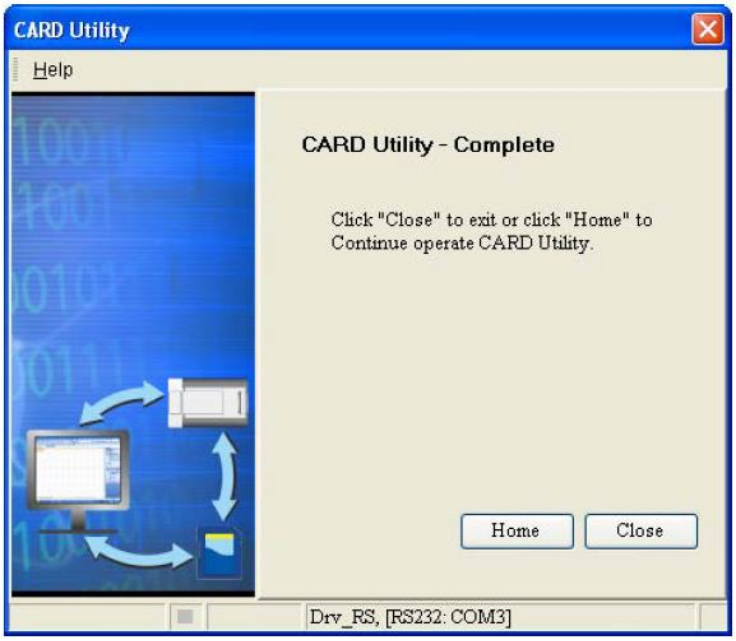
در این مرحله حتی در صورت کلیک بر روی Cancel (حین بازیابی) برای جلوگیری از بازیابی پشتیبان در PLC، بازیابی تا حدودی انجام می‌شود و چون داده‌ها به طور کامل بازیابی نشده‌اند. نیاز است در این حال بازیابی دوباره صورت گیرد و یا PLC به تنظیمات پیشفرض بازگردانده شود. همچنین چون ممکن است حتی با کلیک بر روی Cancel عملیات بازیابی متوقف نشود، کاربر می‌تواند برای اطمینان از قطع شدن عملیات بازیابی، PLC را خاموش کند.

در صورتی که منبع ویا مقصد بازیابی دارای رمز عبور و یا شناسه باشند، عملکرد سیستم به صورت زیر خواهد بود.

جدول ۱۲-۱۰: رمز عبور حین بازیابی

نوع بازیابی	شرح
حافظه به CPU	- شناسه پشتیبان و CPU باید یکی باشند. - رمز عبور PLC و فایل پشتیبان باید یکی باشند. - در صورتی که PLC دارای رمز عبور نباشد، رمز عبور پشتیبان پس از بازیابی، رمز عبور PLC خواهد شد.
کامپیوتر به CPU	- شناسه پشتیبان و CPU باید یکی باشند. - رمز عبور PLC و فایل پشتیبان باید یکی باشند. - در صورتی که PLC دارای رمز عبور نباشد، رمز عبور پشتیبان پس از بازیابی، رمز عبور PLC خواهد شد.
کامپیوتر به ISPSoft	- شناسه و رمز عبور پشتیبان همان شناسه برنامه و رمز عبور پروژه ISPSoft خواهد شد.

۵- در انتها نیز در صورت موفقیت آمیز بودن بازیابی صفحه CARD Utility – Complete ظاهر می‌شود.



شکل ۱۲-۱۱۷ تنظیمات CARD Utility – بازیابی پشتیبان– قسمت ۶

۱۲–۵–ارتباط USB

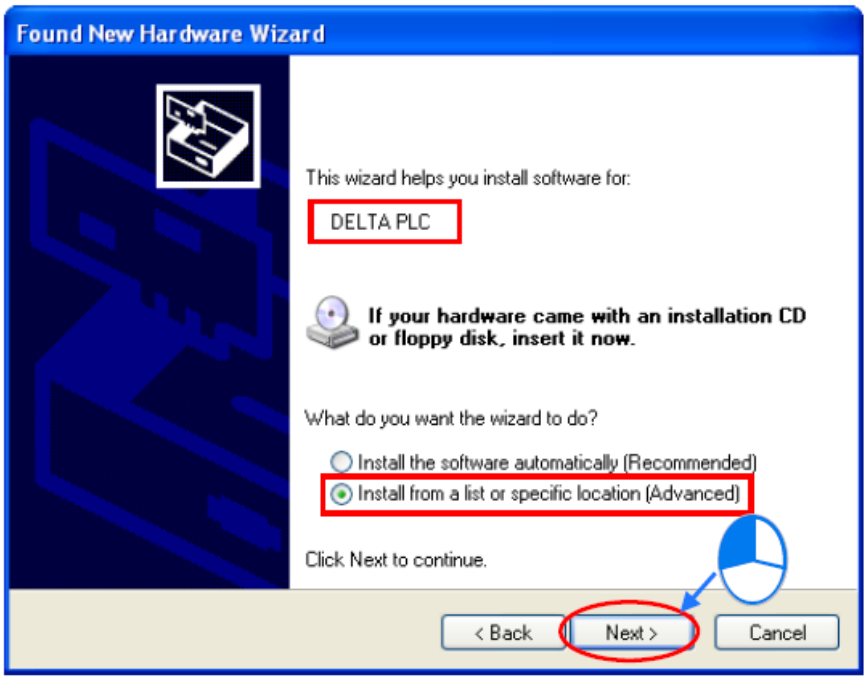
۱۲–۵–۱– نصب درایور USB

نصب درایور USB برای PLC در ویندوز XP در ادامه شرح داده شده است.

۱- مطمئن شوید تغذیه PLC برقرار است و ارتباط PLC با کامپیوتر از طریق پورت USB برقرار است. در این هنگام صفحه Found New Hardware Wizard باز خواهد شد که باید در آن گزینه No, not this time را انتخاب کرده و بر روی Next کلیک کنید.

شکل ۱۲-۱۱۸ نصب درایور USB – قسمت ۱

۲- در این صفحه نام دستگاه متصل به USB نشان داده می‌شود، نام نشان داده شده زیر برای PLCهای سری AH500 است و برای دیگر ماژول‌های USB دار کمپانی دلتا نام‌های متفاوتی ممکن است نشان داده شود. در ادامه Install from a lost or specific location (Advanced) را انتخاب و بر روی Next کلیک کنید.



شکل ۱۲-۱۱۹ نصب درایور USB – قسمت ۲

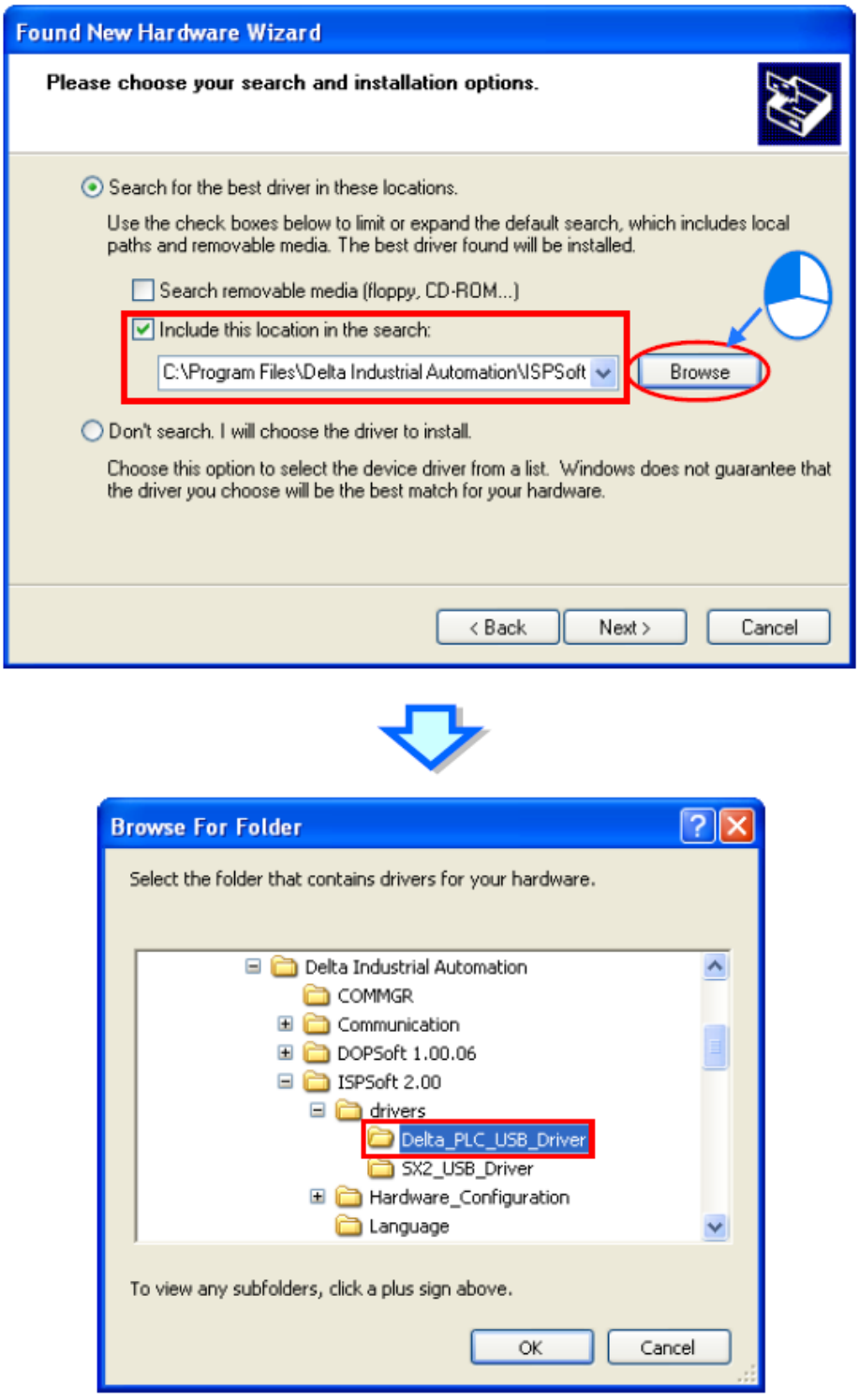
۳- محصولات دلتا که از ارتباط USB پشتیبانی می‌کنند در جدول زیر آمده‌اند، ISPSoft درایور این محصولات را در مسیرهای مشخص شده زیر ذخیره می‌کند[۱] که می‌توان از آن‌ها استفاده کرد.

جدول ۱۲-۱۱: محل ذخیره درایورها

مدل PLC	مسیر ذخیره شدن درایور
DVP-SX2	Installation path of ISPSoft\drivers\SX2_USB_Driver\
DVP-SE	Installation path of ISPSoft\drivers\SX2_USB_Driver\
AH500	Installation path of ISPSoft \drivers\Delta_PLC_USB_Driver\

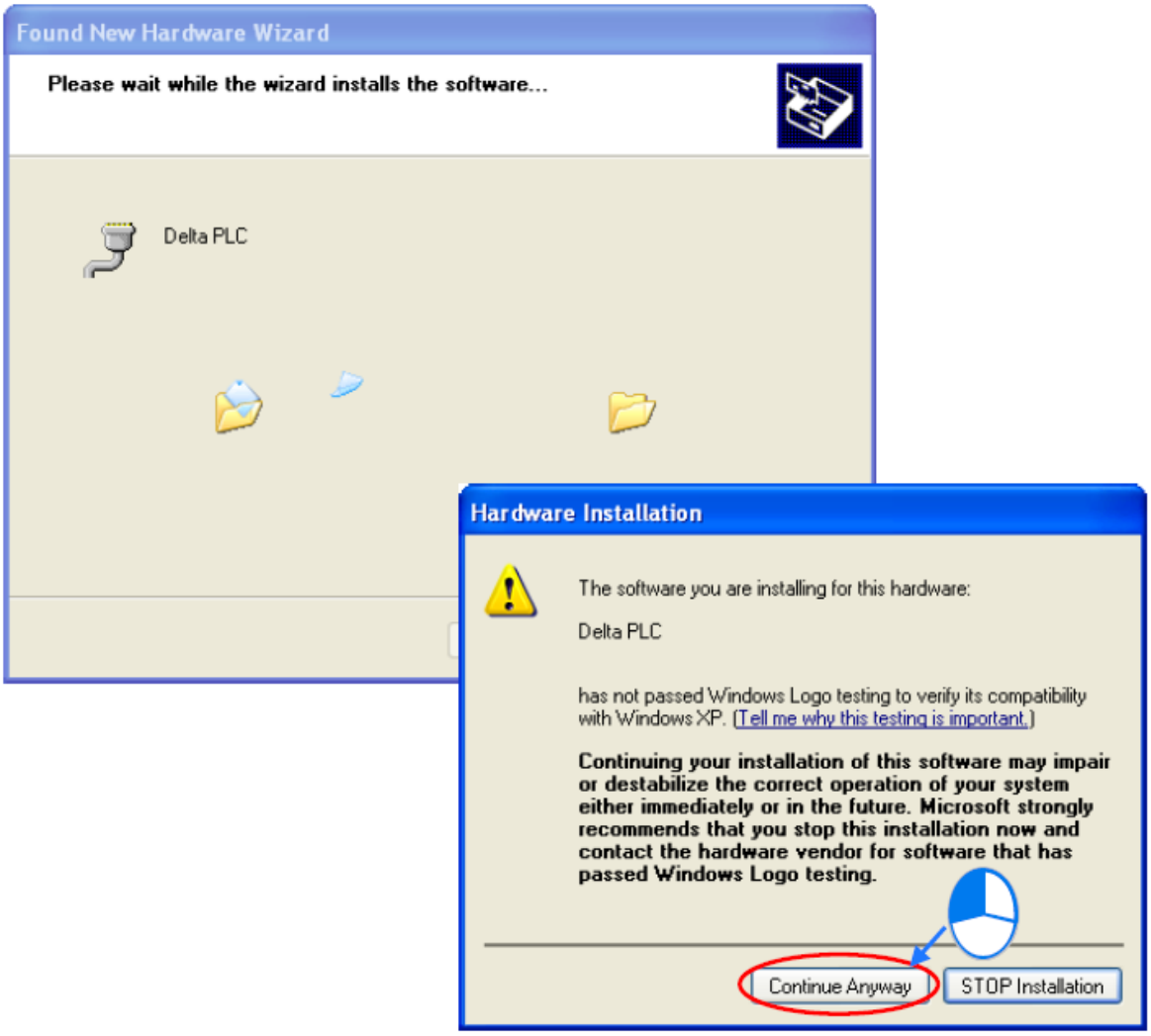
[۱] مسیر پیشفرض نصب ISPSoft به صورت C:\Program Files\Delta Industrial Automation\ISPSoftx.xx است.

در صفحه ظاهر شده آدرس مورد نظر را وارد می‌کنیم و بر روی Next کلیک می‌کنیم.



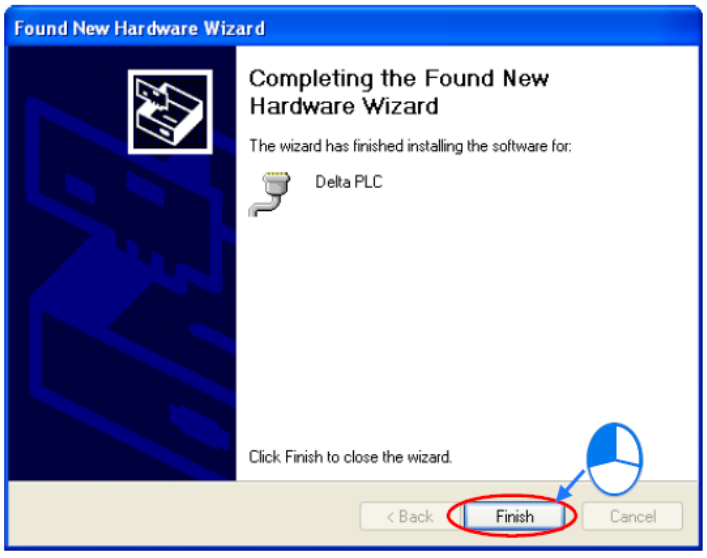
شکل ۱۲-۱۲۰ نصب درایور USB – قسمت ۳

۴- پس از آنکه نرم افزار درایور را پیدا کرد، سیستم درایور را نصب خواهد کرد. در حین نصب اگر پنجره Hardware Installation ظاهر شد می‌توانید گزینه Continue Anyway را انتخاب کنید.



شکل ۱۲-۱۲۱ نصب درایور USB – قسمت ۴

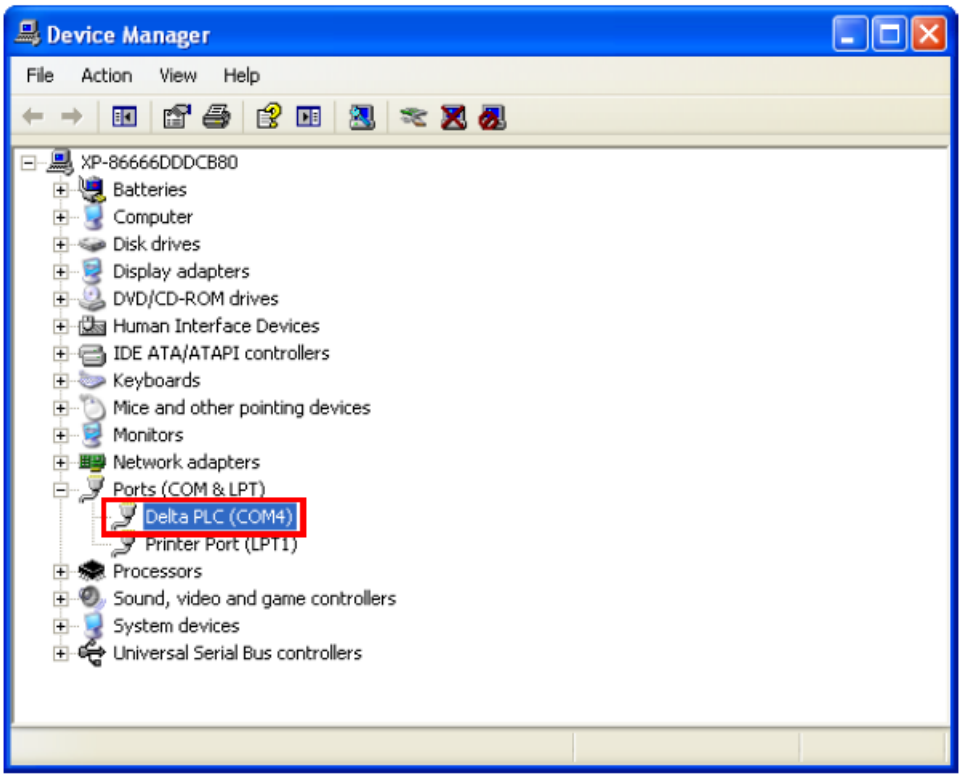
۵- برای پایان نصب می‌توان در صفحه پایانی بر روی Finish کلیک کرد.



شکل ۱۲-۱۲۲ نصب درایور USB – قسمت ۵

۶- برای اطمینان از نصب صفحه Device Manager[1] را باز کنید، در صورتی که نام دستگاه USB دار در قسمت Ports (COM&LPT) آمده بود، مطمئن می‌شویم که نصب درایور به درستی انجام گرفته است. در این حالت سیستم عامل، عددی را به پورت مورد نظر اختصاص (در مثال زیر COM4) خواهد داد که باید آن را در تنظیمات COMMGR مورد استفاده قرار دهیم.

در صورتی که USB به دیگر پورت‌های کامپیوتر وصل شود، ممکن است نیاز باشد تا دوباره درایور را برای پورت جدید نیز نصب کنیم. همچنین با هر بار اتصال مجدد USB به کامپیوتر ممکن است عدد اختصاصی کامپیوتر به COM آن عوض شود.



شکل ۱۲-۱۲۳ بررسی نصب درایور در Device Manager

[1] برای باز کردن Device Manager در My Computer می‌توان کلیک راست کرده و Properties را انتخاب کرد. در پنجره باز شده بر روی سربرگ Hardware کلیک و Device manager را انتخاب می‌کنیم.

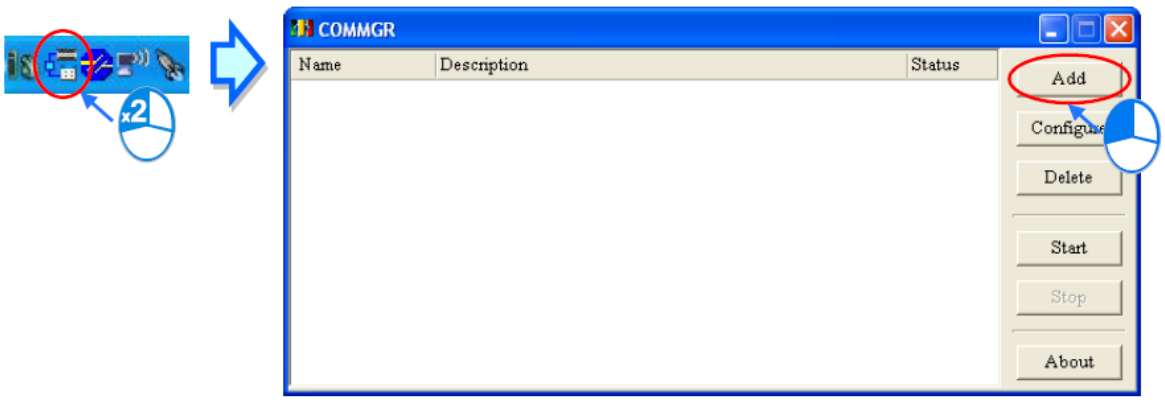
۱۲-۵-۲- ساخت درایور USB در COMMGR

برای ارتباط ISPSoft با PLC از طریق پورت USB نیاز است در COMMGR درایور USB ایجاد کنیم. در اینجا ابتدا نیاز است درایور USB دستگاه مورد نظر نصب شود.

۱- مطمئن شوید تغذیه PLC برقرار و ارتباط کامپیوتر و PLC از طریق پورت USB برقرار است.

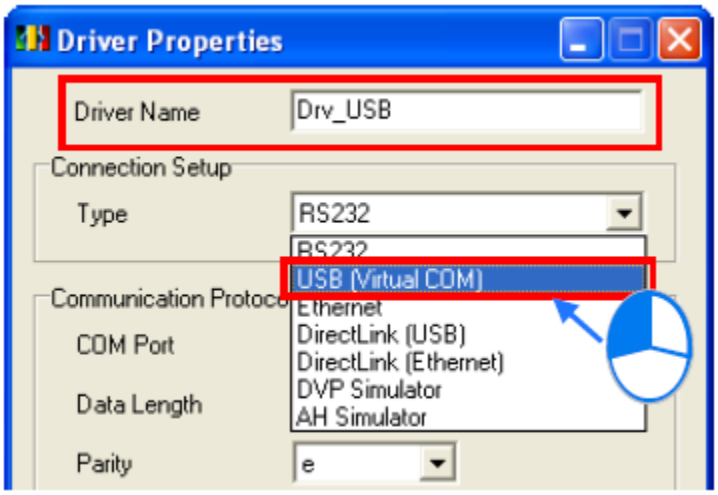
۲- مطمئن شوید COMMGR در حال اجراست، یعنی آیکون آن در Taskbar ویندوز وجود داشته باشد. (در صورتی که فعال نباشد، می‌توانید آن را از آدرس Start>Programs>Delta Industrial Automation>Communication>COMMGR فراخوانی کنید).

۳- با دوبار کلیک بر روی آیکون COMMGR در Taskbar ویندوز پنجره آن را باز کنید. سپس برای ایجاد درایور جدید بر روی Add کلیک کنید.



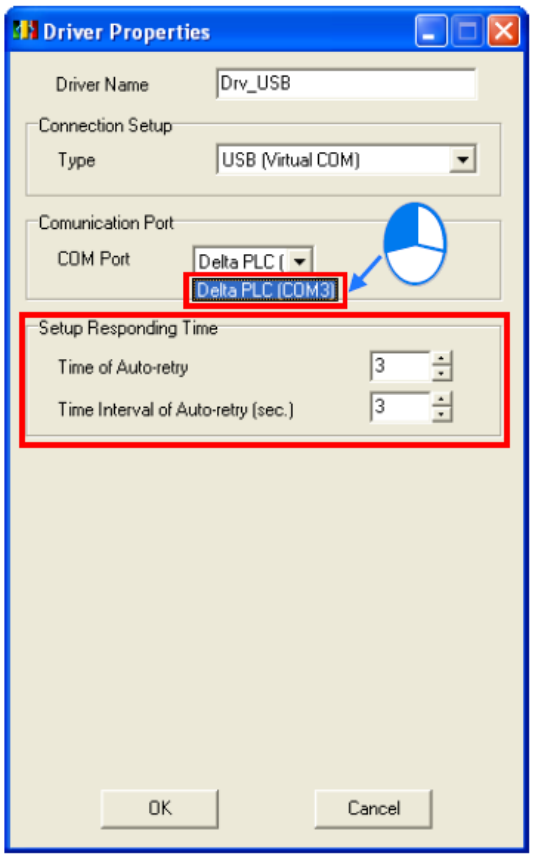
شکل ۱۲-۱۲۴ اجرای COMMGR

۴- در صفحه ظاهر شده، نام درایور را در قسمت Driver Name وارد کنید. نوع درایور باید در قسمت Type به صورت USB (Virtual COM) انتخاب شود. (در صورتی که نوع PLC مدل DVP-SX2 هست باید نوع درایور RS232 انتخاب شود.)



شکل ۱۲-۱۲۵ ساخت درایور USB – قسمت ۱

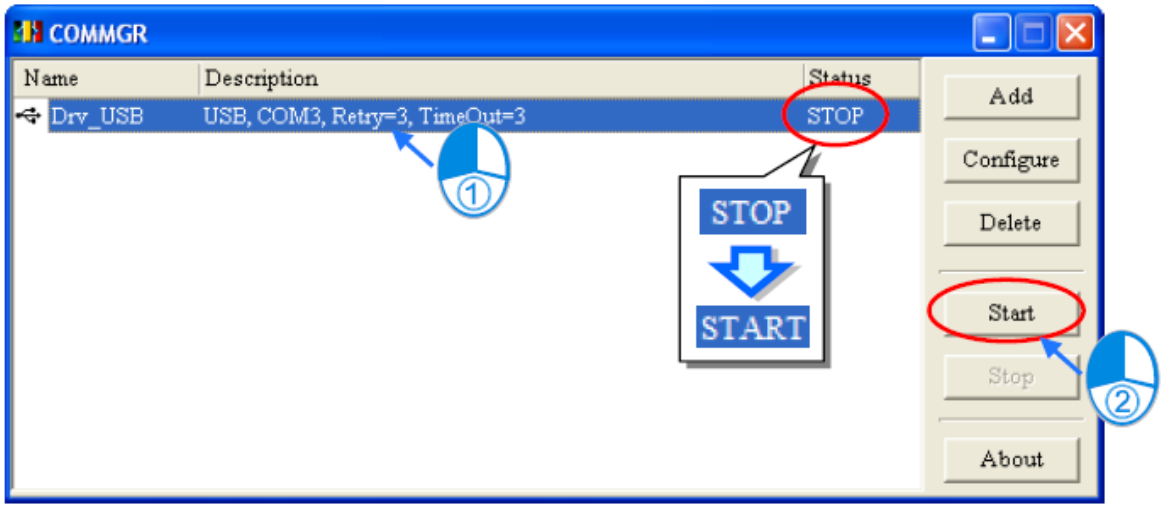
۵- نام دستگاه مورد نظر خود و شماره پورت COM آن را در قسمت Communication Port انتخاب کنید. در قسمت Setup Responding Time نیز کاربر می‌تواند حداکثر تعداد تلاش برای ارتباط و حداکثر مدت زمان انتظار برای اینکار را به ترتیب در دو قسمت Time of Auto-retry و Time Interval of Auto-retry مشخص نماید. در نهایت نیز بر روی OK کلیک کنید.



شکل ۱۲-۱۲۶ ساخت درایور USB – قسمت ۲

در صورتی که نام PLC مورد نظر در بخش COM Port ظاهر نشد، باید کاربر در Device Manager بررسی کند که نام سخت افزار مورد نظر در قسمت Ports (COM&LPT) وجود دارد. در غیر این صورت باید وضعیت اتصال USB و نصب بودن درایور را دوباره بررسی کند.

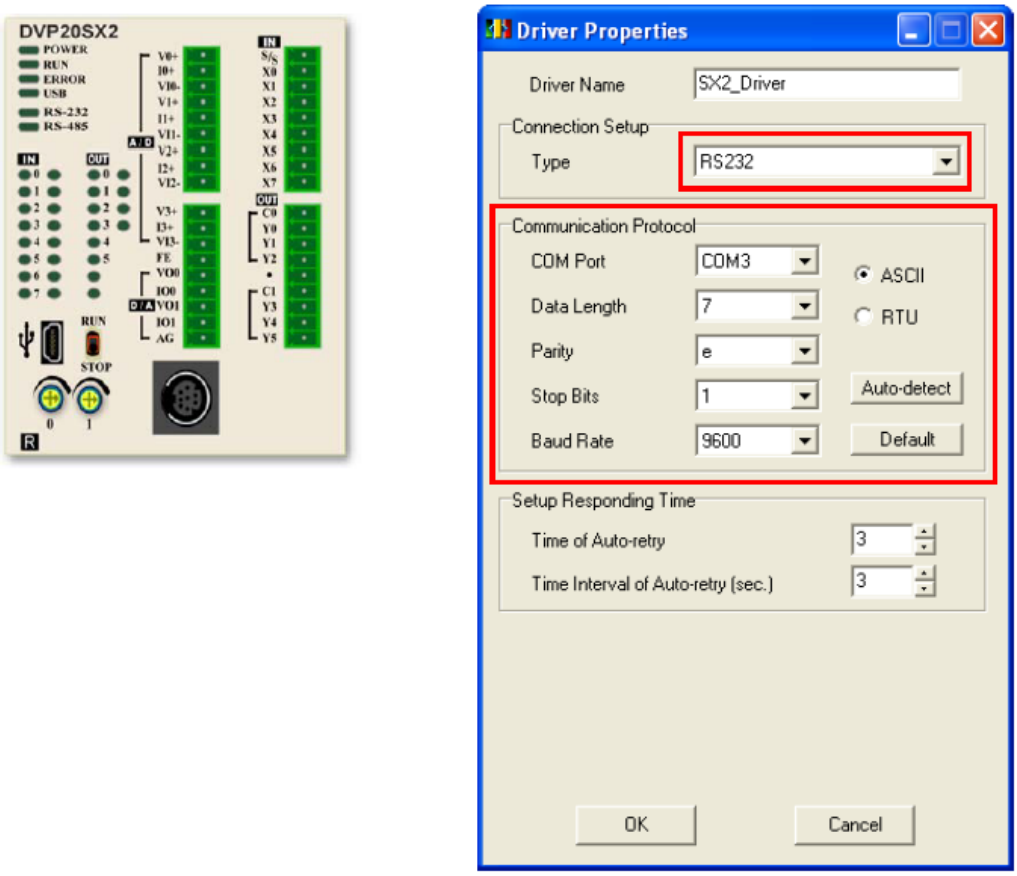
۶- در صفحه COMMGR درایور مورد نظر را انتخاب کرده و برای فعال شدن ارتباط بر روی Start کلیک کنید.



شکل ۱۲-۱۲۷ فعال کردن ارتباط درایور USB

۱۲-۵-۳- تنظیمات USB برای PLCهای مدل DVP-SX2

در PLCهای مدل DVP-SX2 به علت آنکه مبدل USB به RS-232 بر روی آن نصب شده است، با آنکه در ظاهر از کابل و پورت ورودی USB برای ارتباط بین PLC و کامپیوتر استفاده می‌کنیم ولی در واقع پروتکل ارتباطی بین دو دستگاه RS-232 است و باید تنظیمات درایور در COMMGR را برای آن به عنوان ارتباط RS-232 انجام دهیم. در واقع نوع درایور را باید RS-232 انتخاب کنیم. همچنین کاربران می‌توانند از این پورت ارتباطی برای انتقال اطلاعات بین کامپیوتر و PLC از طریق حافظه D1109 اقدام کنند (این گزینه برای کاربردهای آکادمیک به عنوان نمونه ارتباط با نرم افزار متلب بسیار مفید است).



شکل ۱۲-۱۲۸ تنظیمات درایور USB برای PLCهای مدل DVP-SX2

۱۲-۶-نکات مهم در ارتباط با PLCهای سری AH500

PLCهای سایز متوسط AH500 به وسیله شرکت دلتا به بازار عرضه شده و نسبت به PLCهای قبلی این شرکت یعنی خانواده DVP دارای مزیت‌های بسیار بیشتری می‌باشند. به همین علت برای کار با آن‌ها نیاز است راهنمای آن به صورت کامل مرور شود.

۱۲-۶-۱- آدرس حافظه ها و پورت ها در PLCهای AH500

انواع مختلف حافظه و پورت در PLCهای AH500 به صورت زیر است.

جدول ۱۲-۱۲: انواع حافظه و پورت در PLCهای AH500

نوع	مشخصه	شرح
X	Bit/Word	رله ورودی: معرف ورودی دیجیتال(*)

Y	Bit/Word	رله خروجی: معرف خروجی دیجیتال(*)
M	Bit	رله کمکی: معرف حافظه بیتی
S	Bit	رله Step: برای ذخیره حالت پرچم برای Step در SFC
D	Bit/Word	رجیستر داده: حافظه برای ذخیره داده‌ها(*)
L	Bit/Word	رجیستر واسط (Link): حافظه برای ردوبدل کردن اطلاعات(*) (**)
T	Bit	زمان سنج: در صورتی که زمان سنج به عنوان حافظه بیتی استفاده شود، زمانی "یک" می‌شود که مقدار زمان‌سنج به مقدار معین شده رسیده باشد و در غیر این صورت صفر خواهد بود.
	Word	زمان سنج: مقدار فعلی زمان‌سنج که در یک حافظه Word ذخیره می-شود.
C	Bit	شمارنده: در صورتی که شمارنده به عنوان حافظه بیتی استفاده شود، زمانی "یک" می‌شود که تعداد شمارش شده به مقدار معین شده رسیده باشد و در غیر این صورت صفر خواهد بود.
	Word	شمارنده: مقدار فعلی شمارنده که در یک حافظه Word ذخیره می‌شود.
HC	Bit	شمارنده ۳۲ بیتی: در صورتی که شمارنده ۳۲ بیتی به عنوان حافظه بیتی استفاده شود، زمانی "یک" می‌شود که تعداد شمارش شده به مقدار معین شده رسیده باشد و در غیر این صورت صفر خواهد بود.
	Word	شمارنده ۳۲ بیتی: مقدار فعلی شمارنده ۳۲ بیتی که در حافظه Word ذخیره می‌شود.
E	Word	حافظه Index: مقدار آن نشان‌دهنده میزان Offset در حافظه مورد ویرایش است.
SM	Bit	رله کمکی خاص: پرچم برای عملکردهای خاص سیستم. (****)
SR	Word	رله کمکی خاص: رجیستر برای حافظه‌های خاص سیستم. (****)

توضیحات جدول:

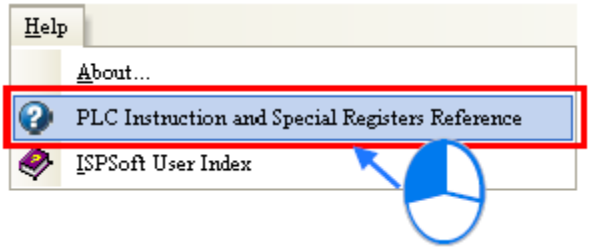
(*) در PLCهای سری AH500، می‌توان بیت‌های رله‌های ورودی، خروجی و رجیستر داده و رجیستر واسط را تغییر داد. برای مثال می‌توان از X0.1 برای دسترسی به بیت اول X0 استفاده کرد، از Y0.1 برای دسترسی به بیت اول Y0 استفاده کرد، از D0.1 برای دسترسی به بیت اول D0 استفاده کرد و از D0.1 برای دسترسی به بیت اول D0 استفاده کرد.

(**) رجیستر واسط برای تبادل داده‌ها مورد استفاده قرار می‌گیرد، با این حال از آن می‌توانیم به عنوان حافظه داده به صورت عمومی استفاده کنیم.

(***) رجیسترهای Index در CPUهای سری AH500 حافظه‌های نوع E هستند و از حافظه‌های نوع F پشتیبانی نمی‌کند.

(****) این رجیسترهای خاص (SM و SR) مخصوص سری AH500 هستند.

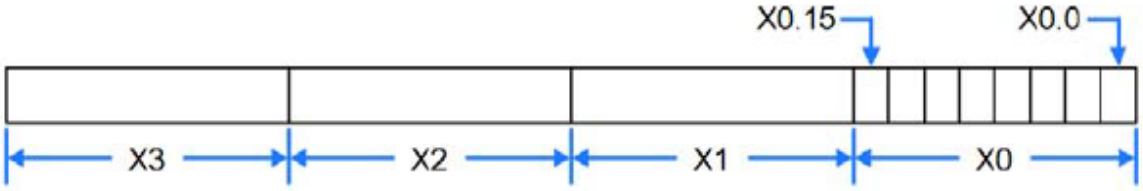
برای اطلاعات بیشتر در مورد رجیسترهای به راهنمای برنامه نویسی AH500 و یا PLC Instructions and Special Register Reference در بخش Help نرم افزار ISPSoft مراجعه کنید.



شکل ۱۲-۱۲۹ PLC Instructions and Special Register Reference در بخش Help نرم افزار ISPSoft

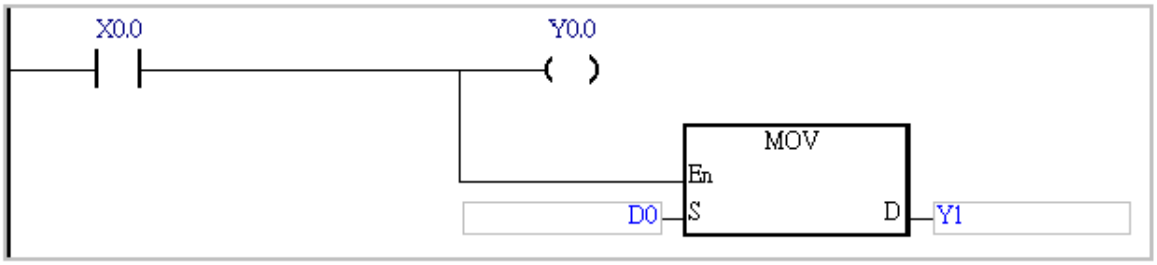
۱۲-۶-۲- کار با بیت ها در پورت و حافظه های نوع X/Y/D/L

پورت‌های X/Y در PLCهای سری DVP با X0، Y0 و ... مشخص می‌شوند. پورت‌های X/Y در PLCهای سری AH500 به صورت بیتی به فرم X0.0، Y0.0 و ... مشخص می‌شوند همچنین پورت‌های X/Y در PLCهای سری AH500 به صورت Word به فرم X0، Y0 و ... مشخص می‌شوند. در نتیجه X0.0 کم ارزش ترین بیت در X0 و X0.15 پر ارزش‌ترین بیت در X0 به حساب می‌آید.



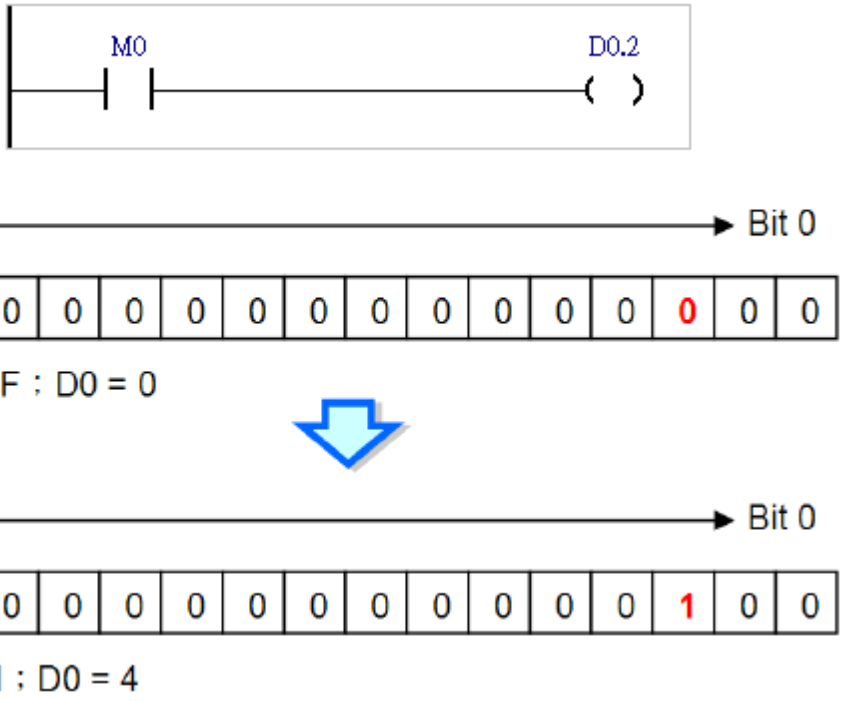
شکل ۱۲-۱۳۰ ارتباط اجزای بیتی و تشکیل Word پورت‌های ورودی

برای مثال در صورتی که D0 را به Y1 منتقل کنیم، حالت Y1.0 الی Y1.15 تغییر خواهند کرد.



شکل ۱۲-۱۳۱ کار با بیت ها در پورت و حافظه ها – مثال ۱

در PLCهای سری AH500، می‌توان بیت‌های رله‌های ورودی، خروجی و رجیستر داده و رجیستر واسط را تغییر داد. در مثال زیر حالت M0 به D0.2 منتقل می‌شود. فرض کنید مقدار اولیه D0 برابر صفر است، زمانی که M0 فعال است، مقدار D0 برابر چهار می‌شود.



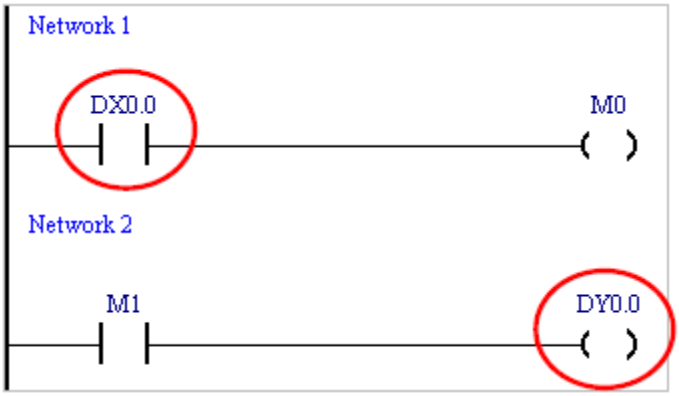
شکل ۱۲-۱۳۲ کار با بیت ها در پورت و حافظه ها – مثال ۲

در صورتی که بیتی در حافظه‌های X/Y/D/L تغییر کند، حافظه Word شامل آن بیت نیز تغییر خواهد کرد. در حالی که حافظه‌های بیتی T/C/HC با حافظه‌های داده آن‌ها متفاوت است و تغییر حافظه‌های بیتی ممکن است تاثیری در حافظه داده آن‌ها نداشته باشد..

۱۲-۶-۳- به روز رسانی مستقیم ورودی/خروجی

روشی که ورودی‌ها و خروجی‌ها در AH500 به روز می‌شوند، مشابه روش به روز شدن ورودی و خروجی‌ها در PLCهای سری DVP است به این صورت که به روز رسانی پس از مرحله Scan کل برنامه صورت می‌گیرد. همچنین در PLCهای سری AH500 روش دیگری برای به روز رسانی با استفاده از کُد "D" وجود دارد. در صورتی که X و یا Y با استفاده از "D" به کار روند، سیستم تغییرات آن‌ها را حین اجرای کُد Network مورد نظر اعمال می‌کندو تا پایان زمان SCAN منتظر نمی‌ماند. تنها پورت‌های ورودی و خروجی X و Y می‌توانند با استفاده از "D" بدین شکل به روز رسانی شوند، حتی از "D" برای سیمبول‌ها نیز نمی‌توانیم استفاده کنیم.

در مثال زیر زمانی که Network 1 اجرا می‌شود، همزمان X0.0 خوانده می‌شود، همچنین سیستم بعد از اجرای کُد Network 2 مقدار M1 را بلافاصله در Y0.0 قرار می‌دهد.



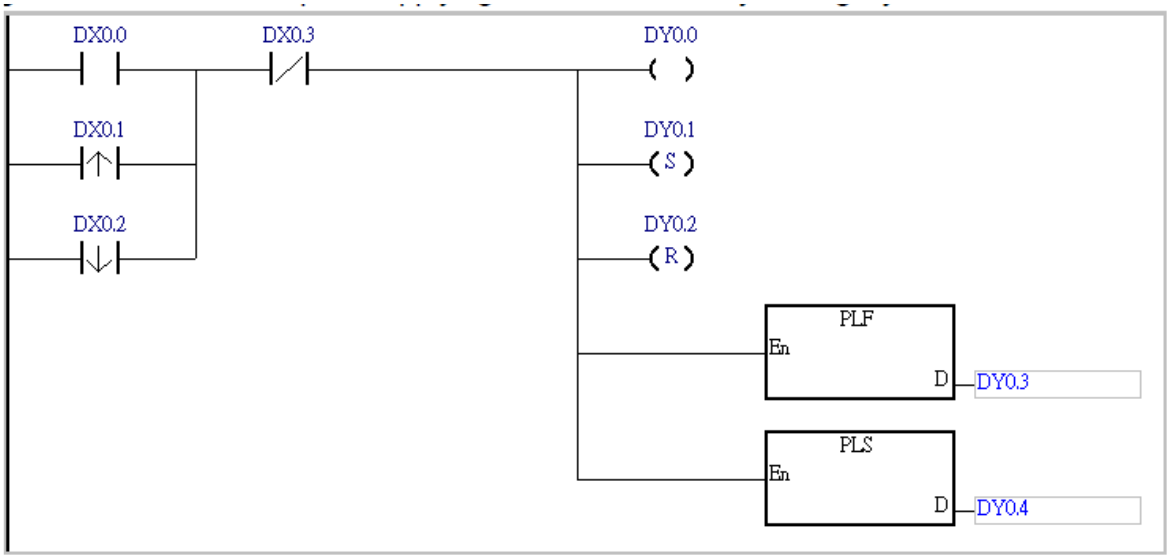
شکل ۱۲-۱۳۳ به روز رسانی مستقیم پورت های ورودی و خروجی – مثال ۱

توابعی که با آن‌ها می‌توانیم از "D" استفاده کنیم در جدول زیر لیست شده است. در صورتی که از"D" در X و یا Y در توابعی که در جدول لیست نشده اند استفاده کنیم، سیستم هنگام کامپایل کردن خطا می‌دهد.

جدول ۱۲-۱۳: توابع مجاز به روز رسانی مستقیم در پورت‌های ورودی و خروجی

نام پورت	توابع مجاز برای استفاده با "D"
X	LD/LDI/LDP/LDF/OR/ORI/ORP/ORF/AND/ANI/ANDP/ANDF
Y	LD/LDI/LDP/LDF/OR/ORI/ORP/ORF/AND/ANI/ANDP/ANDF

مثال زیر چند نمونه استفاده درست از "D" را نمایش می‌دهد.



شکل ۱۲-۱۳۴ به روز رسانی مستقیم پورت های ورودی و خروجی – مثال ۲

همچنین در مثال زیر "D" برای Y به همراه کانتکت ورودی آمده که مجاز نیست.

شکل ۱۲-۱۳۵ به روز رسانی نادرست مستقیم پورت های ورودی و خروجی – مثال ۳

۱۲–۶–۴– محدوده حافظه ها در PLCهای AH500

این محدوده‌ها را در تنظیمات سخت افزاری نیز می‌توان مشاهده کرد.

جدول ۱۲-۱۴: محدوده حافظه های AHCPU500-EN/AHCPU500-RS2

AHCPU500-EN/AHCPU500-RS2		
نوع حافظه و یا پورت	آدرس	
X	Word: X0~X63	Bit: X0.0~X63.15
Y	Word: Y0~Y63	Bit: Y0.0~Y63.15
D	Word: D0~D16383	Bit: D0.0~D16383.15
L	Word: L0~L16383	Bit: L0.0~L16383.15
M	M0~M8191	

S	S0~S2047
C	C0~C2047
HC	HC0~HC63
T	T0~T2047
E	E0~E31
SR	SR0~SR2047
SM	SM0~SM2047
تعداد بلوک‌های حافظه قابل تخصیص به توابع بلوکی instances	64

جدول ۱۲-۱۵: محدوده حافظه های AHCPU510-EN/AHCPU510-RS2

AHCPU510-EN/AHCPU510-RS2		
نوع حافظه و یا پورت	محدوده آدرس	
X	Word: X0~X127	Bit: X0.0~X127.15
Y	Word: Y0~Y127	Bit: Y0.0~Y127.15
D	Word: D0~D32767	Bit: D0.0~D32767.15
L	Word: L0~L32767	Bit: L0.0~L32767.15
M	M0~M8191	
S	S0~S2047	
C	C0~C2047	
HC	HC0~HC63	
T	T0~T2047	
E	E0~E31	
SR	SR0~SR2047	
SM	SM0~SM2047	
تعداد بلوک‌های حافظه	256	

قابل تخصیص به توابع بلوکی instances	

جدول ۱۲-۱۶: محدوده حافظه های AHCPU520-EN/AHCPU520-RS2

AHCPU520-EN/AHCPU520-RS2		
نوع حافظه و یا پورت	محدوده آدرس	
X	Word: X0~X255	Bit: X0.0~X255.15
Y	Word: Y0~Y255	Bit: Y0.0~Y255.15
D	Word: D0~D65535	Bit: D0.0~D65535.15
L	Word: L0~L65535	Bit: L0.0~L65535.15
M	M0~M8191	
S	S0~S2047	
C	C0~C2047	
HC	HC0~HC63	
T	T0~T2047	
E	E0~E31	
SR	SR0~SR2047	
SM	SM0~SM2047	
تعداد بلوک‌های حافظه قابل تخصیص به توابع بلوکی instances	512	

جدول ۱۲-۱۷: محدوده حافظه های AHCPU520-EN/AHCPU520-RS2

AHCPU530-EN/AHCPU530-RS2		
نوع حافظه و یا پورت	محدوده آدرس	
X	Word: X0~X511	Bit: X0.0~X511.15

Y	Word: Y0~Y511	Bit: Y0.0~Y511.15
D	Word: D0~D65535	Bit: D0.0~D65535.15
L	Word: L0~L65535	Bit: L0.0~L65535.15
M	M0~M8191	
S	S0~S2047	
C	C0~C2047	
HC	HC0~HC63	
T	T0~T2047	
E	E0~E31	
SR	SR0~SR2047	
SM	SM0~SM2047	
تعداد بلوک‌های حافظه قابل تخصیص به توابع بلوکی instances	1024	

جدول ۱۲-۱۸: جدول سراسری مقایسه حافظه های PLCهای سری AH500

Type	Device name		Number of devices	Range
Bit device	Input relay	X	1024(AHCPU500)	X0.0~X63.15
			2048(AHCPU510)	X0.0~X127.15
			4096(AHCPU520)	X0.0~X255.15
			8192(AHCPU530)	X0.0~X511.15
	Output relay	Y	1024(AHCPU500)	Y0.0~X63.15
			2048(AHCPU510)	Y0.0~X127.15
			4096(AHCPU520)	Y0.0~X255.15
			8192(AHCPU530)	Y0.0~Y511.15
	Data register	D	16384 (AHCPU500)	D0.0~D16383.15
			32768 (AHCPU510)	D0.0~D32767.15
			65536 (AHCPU520/530)	D0.0~D65535.15
	Link register	L	16384 (AHCPU500)	L0.0~ L16383.15
			32768 (AHCPU510)	L0.0~ L32767.15
			65536 (AHCPU520/530)	L0.0~ L65535.15
	Auxiliary relay	M	8192	M0~M8191
	Special auxiliary relay	SM	2048	SM0~SM2047
	Stepping relay	S	2048	S0~S2047
	Timer	T	2048	T0~T2047
	Counter	C	2048	C0~C2047

Type	Device name		Number of devices	Range
	32-bit counter	HC	64	HC0~HC63
Word device	Input relay	X	512	X0~X511
	Output relay	Y	512	Y0~Y511
	Data register	D	16384 (AHCPU500)	D0~D16383
			32768 (AHCPU510)	D0~D32767
			65536 (AHCPU520/530)	D0~D65535
	Special data register	SR	2048	SR0~SR2047
	Link register	L	16384 (AHCPU500)	L0~L16383
			32768 (AHCPU510)	L0~L32767
			65536 (AHCPU520/530)	L0~L65535
	Timer	T	2048	T0~T2047
	Counter	C	2048	C0~C2047
	32-bit counter	HC	64 (128 words)	HC0~HC63
	Index register	E	32	E0~E31
Constant*	Decimal system	K	16 bits: -32768~32767 32 bits: -2147483648~2147483647	
	Hexadecimal system	16#	16 bits: 16#0~16#FFFF 32 bits: 16#0~16#FFFFFFFF	
	Single-precision floating-point number	F	32 bits: $\pm1.17549435^{-38}$~$\pm3.40282347^{+38}$	
	Double-precision floating-point number	DF	64 bits: $\pm2.2250738585072014^{-308}$~ $\pm1.7976931348623157^{+308}$	
String*	String	"$"	1~31 characters	

۱۲-۶-۵- محدوده حافظه ها در PLCهای DVP

این محدوده‌ها را در تنظیمات سخت افزاری نیز می‌توان مشاهده کرد.

جدول ۱۲-۱۹: جدول سراسری مقایسه حافظه های PLCهای سری DVP

Device	Range	Type	Applicable to			
			ES/EX/SS/EC	ES2/EX2	SA/SX/SC	EH2/SV
P	0~255	-	0~63	0~255	0~255	0~255
N	0~7	-	0~7	0~7	0~7	0~7
I	0~899	-	I001, I101, I201, I301 /I6□□ (10~99) / I150	I000 / I001(X0), I100 / I101(X1), I200 / I201(X2), I300 / I301(X3), I400 / I401(X4), I500 / I501(X5), I600 / I601(X6)/ I700 / I701(X7), I605~I699, I705~I799, I010, I020,	I001, I101, I201, I301, I401, I501 / I6□□,I7□□ (1~99) / I010, I020, I030,	I00□(X0), I10□(X1), I20□(X2), I30□(X3), I40□(X4), I50□(X5) / I6□□, I7□□, I8□□, (1~99) / I010, I020, I030, I040, I050, I060,I110,

				I030, I040, I050, I060, I070, I080, I140, I150, I160	I040, I050, I060, I150	I120, I130, I140, I150, I160, I170, I180
S	0~1023	Bit	0~127	0~1024	0~1024	0~1024
X	0~377 (Octal)	Bit	0~177	0~377	0~177	0~377
Y	0~377 (Octal)	Bit	0~177	0~377	0~177	0~377
T	0~255	Bit/ Word	0~127	0~255	0~255	0~255
M	0~4095	Bit	0~1279	0~4095	0~4095	0~4095
C	0~199 16-bit	Bit/ Word	0~199	0~199	0~199	0~199
	200~255 32-bit	Bit/ Dword	200~255	200~255	200~255	200~255
D	0~9999	Word	0~1311	0~9999	0~4999	0~9999
E	0~7	Word	0	0~7	0~3	0~7
F	0~7	Word	0	0~7	0~3	0~7

فصل ۱۳– تنظیمات شبکه و تبادل داده

در این فصل تنها به صورت عمومی شما را با فضای شبکه‌های صنعتی مورد استفاده در محصولات دلتا به خصوص نرم افزار ISPSoft آشنا می‌کنیم و به مرور مباحث مربوط به شبکه در NWCONFIG می‌پردازیم. همچنین در ادامه به معرفی امکانات کمپانی دلتا جهت بکارگیری شبکه‌های صنعتی خواهیم پرداخت.
مباحث شرح داده شده در این فصل مقدماتی بوده و جهت آشنایی عمومی با مفاهیم مورد نظر آورده شده است. مباحث تکمیلی پیرامون به کارگیری انواع سخت افزارهای مربوط به شبکه‌های صنعتی کمپانی دلتا (IE, Field Bus, Profibus, CANOpen, Industrial Ethernet و ...)، نرم افزار مربوطه، مفاهیم مرتبط با پروتکل‌ها، روش‌های انتقال داده، شبکه‌های بیسیم، استفاده از شبکه اینترنت برای تبادل داده، سیستم-های اسکادا و DCS و دیگر مباحث مرتبط به زودی در کتابی جامع با همکاری شرکت کامیاب مرام، نمایندگی کمپانی دلتا در ایران منتشر خواهد شد.
همچنین جهت آشنایی با راه اندازی یک ماژول شبکه، در ضمیمه ه - به معرفی و راه اندازی ماژول EN01 پرداخته شده است، از این ماژول می‌توان برای اتصال PLCهای سری DVP به شبکه Industrial Ethernet استفاده کرد.

۱۳–۱– تنظیمات شبکه با استفاده از NWCONFIG

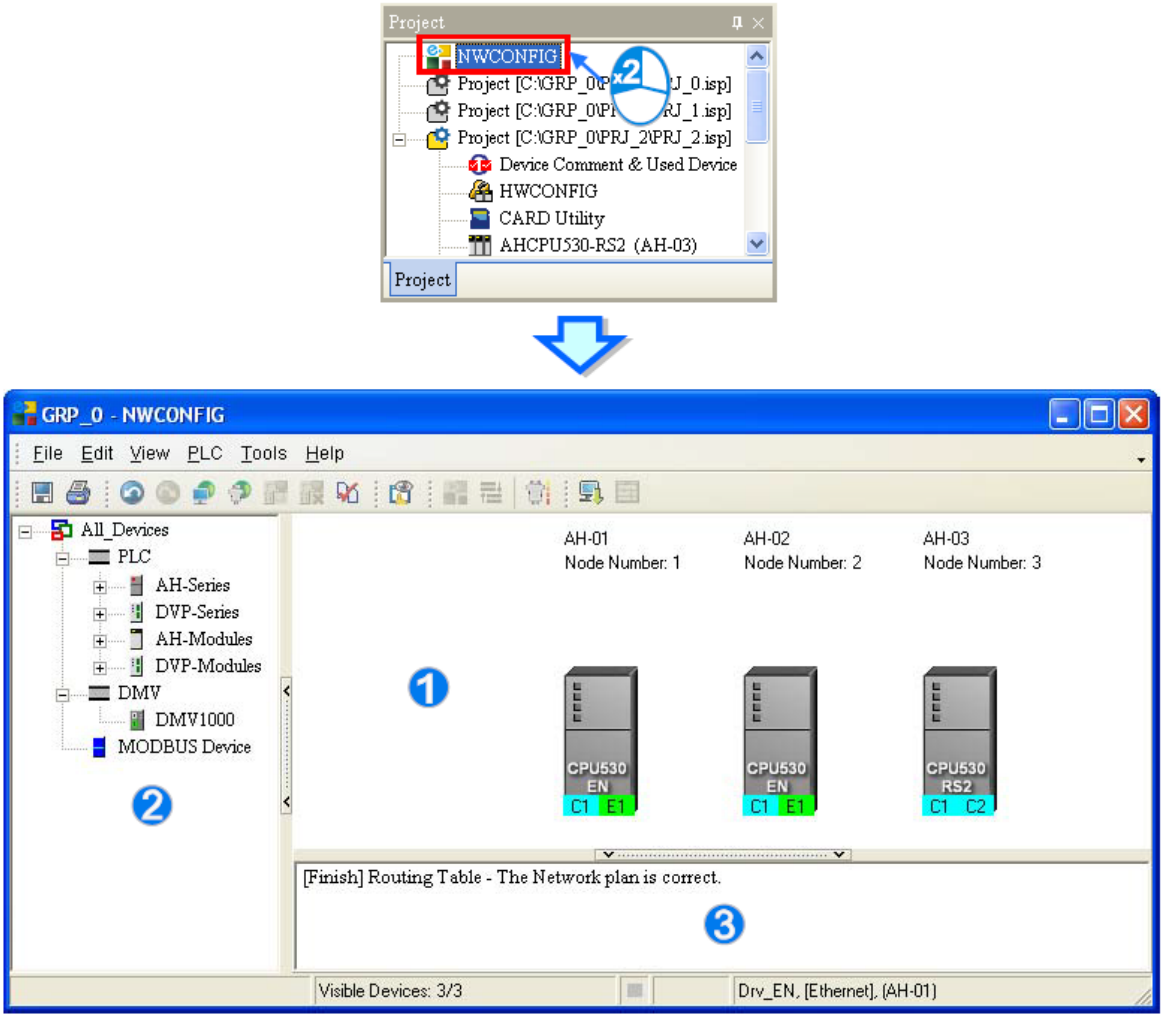
ابزار NWCONFIG جهت تنظیمات شبکه در نرم افزار ISPSoft در نظر گرفته شده است. با استفاده از این ابزار می‌توان تنظیمات شبکه و مکانیزم تبادل داده‌ها را در پروژه‌ها انجام داد. این قسمت تا حدود زیادی مشابه NetPro در نرم افزار Step7 شرکت زیمنس عمل می‌کند. عملکردهای NWCONFIG در زیر لیست و در ادامه این بخش به آن‌ها پرداخته خواهد شد[1].

- ساخت شبکه‌های مورد نیاز در پروژه و انتخاب مسیرهای ارسال داده
- ایجاد قابلیت تبادل داده از طریق لینک RS-485 در PLC

[1] به علت تخصصی بودن واژگان حوزه شبکه های صنعتی و متداول بودن کلمات انگلیسی این بخش در بین کاربران، از ترجمه کلمات تخصصی به دلیل جلوگیری از ایجاد ایهام در متن خودداری می‌شود. همچنین در این فصل مفاهیم شبکه به صورت اولیه مورد بررسی قرار می‌گیرد و مباحث تکمیلی در کتاب شبکه دلتا که به زودی منتشر خواهد شد، مطرح خواهد شد.

- ایجاد قابلیت تبادل داده از طریق لینک اِترنت PLC

NWCONFIG چارچوب ساختاری شبکه‌های درون پروژه را پایه‌ریزی می‌کند و به همین علت در بالاترین قسمت در بخش مدیریت پروژه قرار دارد. برای باز شدن پنجره آن دوبار بر روی آن کلیک کنید.



شکل ۱۳-۱ محیط کار NWCONFIG

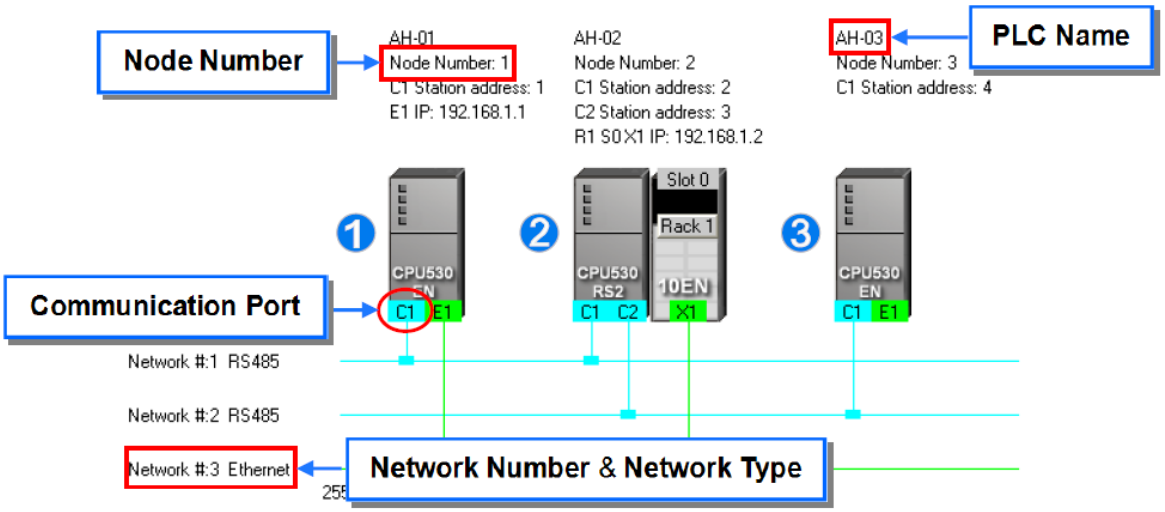
1 محیط کار: محیط اصلی طراحی ساختار سیستم.

2 لیست دستگاه‌ها: تمامی دستگاه‌هایی که امکان استفاده از آن‌ها در برنامه وجود دارد در این قسمت لیست شده است.

3 محیط نمایش پیام: پیام‌های منتشر شده توسط سیستم در این قسمت نمایش داده می‌شوند.

۱۳-۱-۱- اطلاعات اولیه

قبل از طراحی شبکه‌های ارتباطی بین PLCها کاربران می‌بایست اطلاعات اولیه‌ای در مورد مفاهیم پایه شبکه داشته باشند که بعضی از آن‌ها را در ادامه معرفی خواهیم کرد.



شکل ۱۳-۲ پارامترهای اجزای شبکه

- دستگاه‌های پایه درون شبکه

این دستگاه‌ها شامل PLC، ماژول‌ها و تجهیزات تعریف شده توسط کاربر هستند که در ساختار شبکه مورد استفاده قرار گرفته اند. در نهایت نیز هر شبکه شامل تعدادی از این دستگاه‌هاست که با هم دیگر در ارتباط هستند. هر شبکه دارای شماره‌ای هست و نوع آن یا اِترنت و یا RS-485 می‌باشد. در دنیای واقعی (لایه فیزیکی شبکه) دستگاه‌ها برای ارتباط با یکدیگر از طریق پورت‌های خود به هم متصل می‌شوند.

- نام PLC

در شکل بالا سه نام AH-01 و AH-02 و AH-03 نام PLCهای درون شبکه می‌باشند. و می‌بایست برای شناسایی بهتر هرکدام از آن‌ها اسم منحصر به فردی به آن‌ها اختصاص داد. در سری AH500 برای تخصیص نام می‌توان به HWCONFIG مراجعه کرد. در مدل‌های DVP نیز این نام مشابه کامنت روی دستگاه است.

- گره‌ها و شماره آن‌ها

به هر جز مستقل سیستم که به صورت مستقل در شبکه فعالیت می‌کند گره[1] گفته می‌شود. گره‌ها پایه‌ای ترین جز سیستم هستند. شماره‌های 1 الی 3 در شکل فوق گره هستند. شماره ۲ شامل CPU و ماژول شبکه است. از آنجایی که ماژول شبکه به تنهایی قادر به ارتباط و پردازش در سیستم نیست و به یک CPU حتما احتیاج دارد، در نتیجه مجموع این دو یک گره را تشکیل می‌دهند.

همچنین CPUهای سری AH500 می‌توانند مسیریابی[2] انجام داده و داده‌های خود را ارسال نمایند. برای مثال از این طریق در شکل فوق گره ۳ می‌تواند به وسیله گره ۱ مانیتور شود (از طریق گره میانی شماره ۲). قبل از این که این امکان برای مسیریابی فراهم شود، کاربر باید مسیرهای ارتباطی برای ارسال داده‌ها را مشخص کند و به هر گره شماره خاصی اختصاص دهد (در واقع این شماره‌ها آدرس آن گره در شبکه است). تنها به CPUهای سری AH500 می‌توان شماره اختصاص داد و در ضمن به این نکته توجه کرد که نمی‌توان هیچ دو گره‌ای دارای آدرس مشابه باشند. پس از آنکه مسیرهای انتقال داده در نرم افزار ساخته شد و در PLCها دانلود شد، هر PLC می‌تواند جدول مسیرهای خود را به PLCهای دیگر را تشکیل داده و از این طریق داده‌های خود را به آن‌ها منتقل کند.

- آدرس ایستگاه‌ها

در شبکه‌های RS-485 هر پورت ورودی با آدرس ایستگاه[3] منحصر به فردی مشخص می‌شوند. در واقع در اکثر مواقع آدرس ایستگاه معرف همان آدرس گره است. در صورتی که گره‌ای دارای چندین پورت باشد، هر پورت متصل به شبکه باید دارای آدرسی مجزا باشد.

- آدرس IP و حالت DHCP

هرکدام از پورت‌های شبکه اِترنت با آدرس IP منحصر به فردی (عدد انتهایی آن باید بین یک الی ۲۵۴ باشد) مشخص می‌شوند. در اینجا نیز در صورتی که گره‌ای دارای چندین پورت باشد، هر پورت متصل به شبکه باید دارای آدرس IP مجزا باشد.

پروتکل DHCP برای تخصیص دینامیک و البته اتوماتیک آدرس‌های IP گره‌های شبکه اِترنت مورد استفاده قرار می‌گیرد.

---

[1] Node

[2] Routing

[3] Station Address

- Subnet mask

به پورت‌های یک شبکه باید یک Subnet mask یکسان داده شود تا آن‌ها بتوانند در آن دامنه Subnet با هم ارتباط برقرار کنند.

- PLC Link

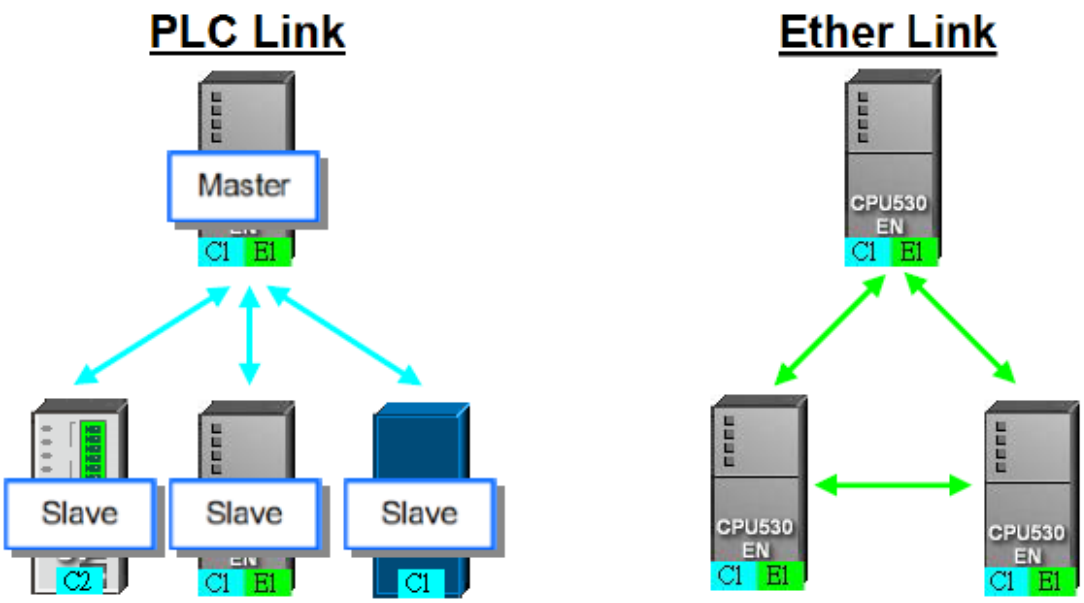
مکانیزم شبکه‌ای برای ارتباط PLC ها از طریق کابل RS-485. در این شبکه یک PLC به عنوان Master وجود خواهد داشت و مابقی PLCها به عنوان Slave عمل خواهند کرد. در این نوع ارتباط PLC-ای که به عنوان Master تعیین شده تبادل اطلاعات را مدیریت می‌کند و دیگر PLCها نمی‌توانند به صورت مستقل به تبادل اطلاعات بپردازند.

- Ether Link

مکانیزم شبکه‌ای برای تبادل داده بین PLCها از طریق بستر اِترنت[1]. در صورتی که PLCها تنظیم شوند، می‌توانند از طریق شبکه اترنت با یکدیگر ارتباط برقرار کنند و به تبادل داده بپردازند. به صورت نرمال (بدون استفاده از تجهیزات جانبی) تنها CPUهای سری AH500 این مکانیزم را پشتیبانی می‌کنند.

این نوع ارتباط به صورت Master/Slave نیست و هر گره در آن می‌تواند با گره‌های دیگر تبادل اطلاعات داشته باشد. تبادل اطلاعات در این مکانیزم شبکه نیز به این صورت است که گره مبدا درخواست اطلاعات خود را به گره مقصد می‌فرستد و گره مقصد پس از دریافت آن، داده‌های مورد نظر را برای گره مبدا ارسال می‌کند. با توجه به اینکه در این ارتباط گره‌ها امکان ارسال درخواست نوشتن داده بر روی گره مقصد را ندارند، این روش به نسبت PLC Link مطمئن‌تر است. در این مکانیزم، ارسال بسته-های داده به صورت خودکار به روش TCP/IP مدیریت می‌شود. به صورت کلی نیز می‌توان گفت استفاده از Ether Link به نسبت PLC Link بهینه‌تر است.

[1] Ethernet



شکل ۱۳-۳ ساختار ارتباط بین اجزا در Ether Link و PLC Link

(لازم است برای اطلاعات بیشتر در مورد پروتکل‌های شبکه و نحوه تبادل داده‌ها در آن‌ها و مقایسه ویژگی‌های کلیدی آن‌ها به کتاب‌های پایه شبکه (یا شبکه‌های صنعتی) مراجعه کنید.)

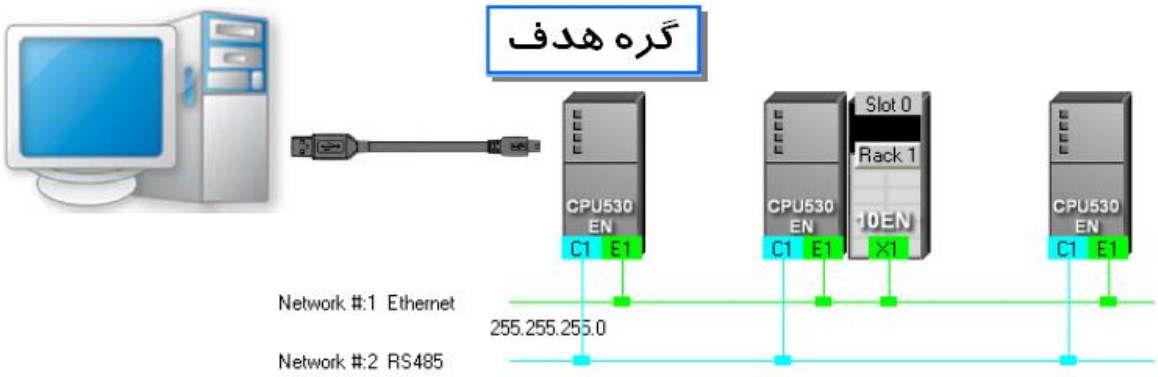
۱۳-۱-۲- تنظیمات ارتباطی در NWCONFIG

نرم‌افزار NWCONFIG برای پیکربندی ساختار و پارامترهای شبکه به کار می‌رود. پس از پیکربندی و بارگزاری آن بر روی گره‌ها کاربر قادر به دانلود بر روی گره‌ها، استخراج پارامترهای گره‌ها و یا مانیتورینگ گره‌های شبکه خواهد بود. همچنین ممکن است شبکه شامل دستگاه‌هایی باشد که در پروژه تعریف نشده باشد، در این حال باید بتوان پارامترهای سیستم را به صورتی تنظیم کرد که اجزای مختلف آن بتوانند به صورت مناسبی در کنار یکدیگر عمل نمایند. برای درک بهتر این موضوع ابتدا به مرور مکانیزم ارتباطی در NWCONFIG می‌پردازیم.

۱۳-۱-۲-۱- مکانیزم ارتباطی در NWCONFIG

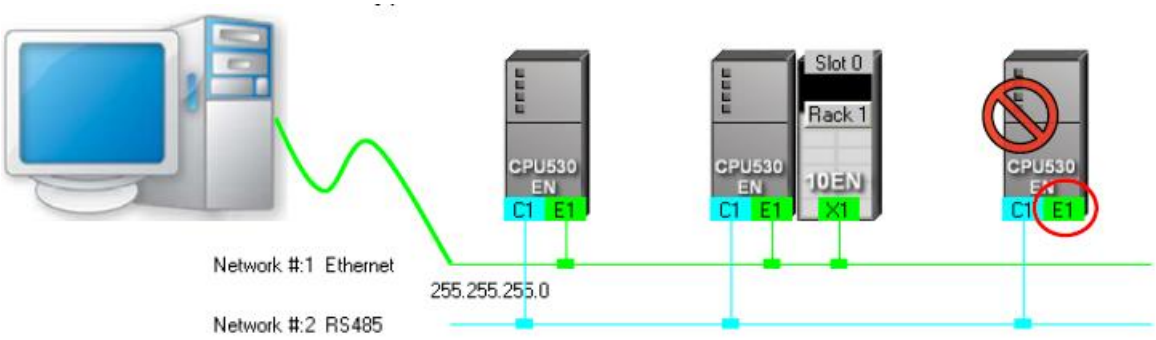
در شبکه ساخته شده در NWCONFIG کاربران می‌توانند پارامترها را به صورت مستقیم در یک یا به صورت گروهی از طریق شبکه در چند گره دانلود و یا داده‌های آن‌ها را استخراج یا وضعیت آن‌ها را مانیتور کنند. در این قسمت به شرح فاکتورهای لازم در تنظیم پارامترهای شبکه می‌پردازیم.

- ارتباط با یک گره مشخص: از طریق ارتباط مستقیم کامپیوتر و نرم افزار ISPSoft می‌تواند با هر کدام از گره‌های شبکه ارتباط برقرار کند، پارامترهای مورد نیاز را بر روی آن‌ها دانلود کند، پارامترهای گره را استخراج کند و داده‌های گره را نیز مانیتور نماید.



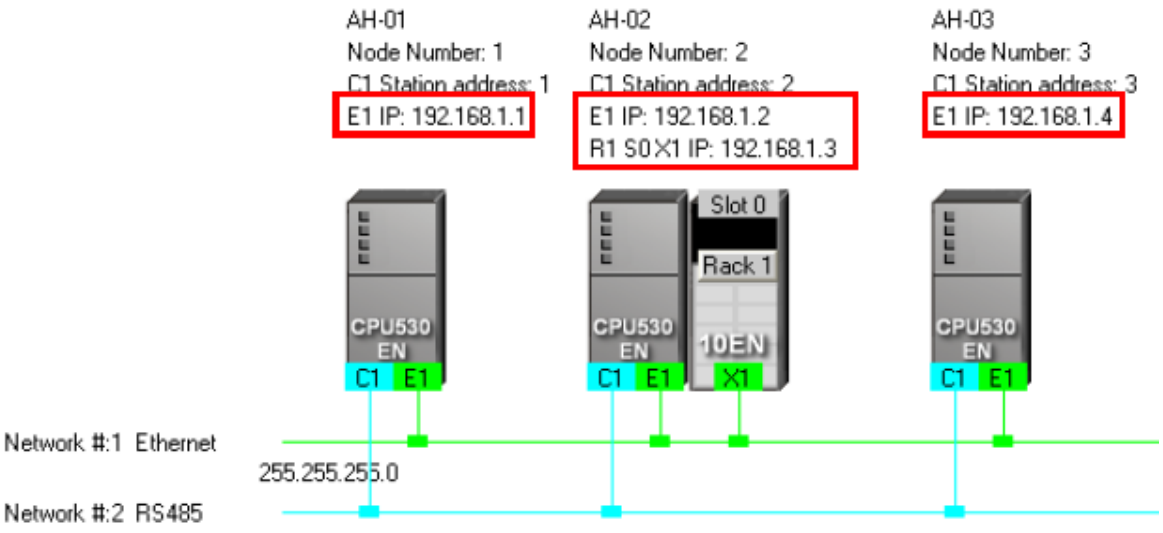
شکل ۱۳-۴ ارتباط مستقیم با یک گره مشخص

- ارتباط با چند گره: از طریق شبکه، کاربر قادر به دانلود پارامترها بر روی گره‌های متصل به شبکه، استخراج پارامترهای گره‌های متصل به شبکه و مانیتورینگ داده‌های شبکه خواهد بود. اما برای اینکه بتوان این شرایط را فراهم کرد لازم است تمامی گره‌ها به شبکه اترنت متصل و به آن‌ها IP مناسب تخصیص داده شده باشد و نوع درایور تنظیم شده برای ارتباط نرم افزار اترنت باشد.



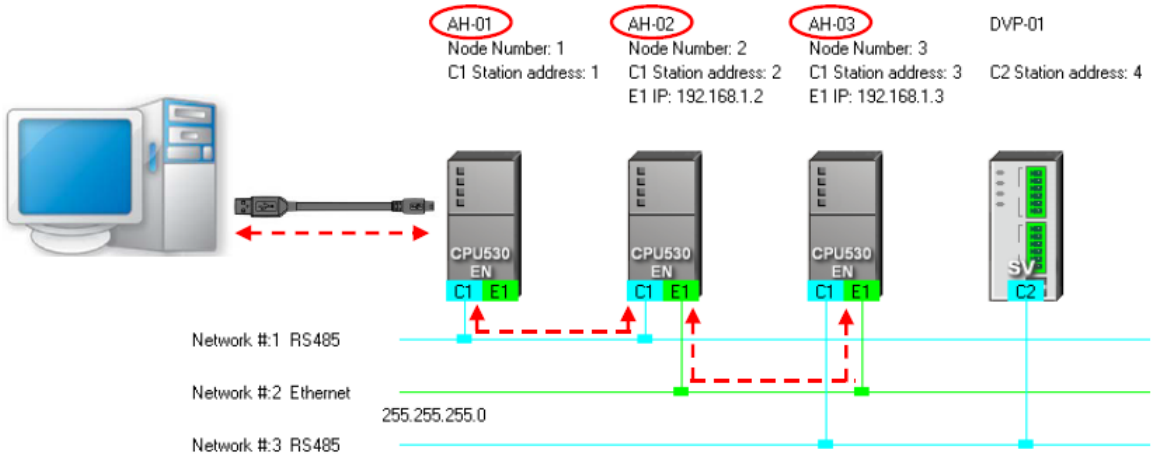
شکل ۱۳-۵ ارتباط با گره‌های یک شبکه

در صورتی که نوع درایور اترنت باشد، سیستم ارتباط را بر اساس IPهای تنظیم شده در NWCONFIG برقرار می‌کند. البته برای اینکه این ارتباط برقرار شود باید تمام گره‌ها IP مطابق با NWCONFIG داشته و ساختار شبکه مشابه طراحی درون NWCONFIG باشد در غیر این صورت ارتباط دچار مشکل خواهد شد.



شکل ۱۳-۶ تخصیص IPهای منحصر به فرد در شبکه Ethernet

NWCONFIG همچنین می‌تواند از طریق مسیریابی[۱] بین شبکه‌های مختلف با گره‌ها ارتباط برقرار کند. مسیریابی عملکردی در PLCهای سری AH500 است که در آن بسته‌های داده از طریق گره‌های میانی در شبکه‌های مختلف جابجا می‌شود تا به گره مقصد برسد. در مثال زیر گره AH-01 به کامپیوتر متصل است. در صورتی که کامپیوتر بخواهد با گره AH-03 ارتباط برقرار کند می‌تواند از طریق مسیریابی ابتدا با AH-01 و سپس از طریق آن با AH-02 و در نهایت با AH-03 متصل شود.



شکل ۱۳-۷ ارتباط از طریق مسیریابی

لازم است به این نکته توجه شود که PLCهای سری DVP از مسیریابی پشتیبانی نمی‌کنند و تنها می‌توان از آن‌ها به عنوان گره‌های انتهایی استفاده کرد.

[۱] Routing

۱۳-۱-۲-۲- تنظیم پارامترهای ارتباطی

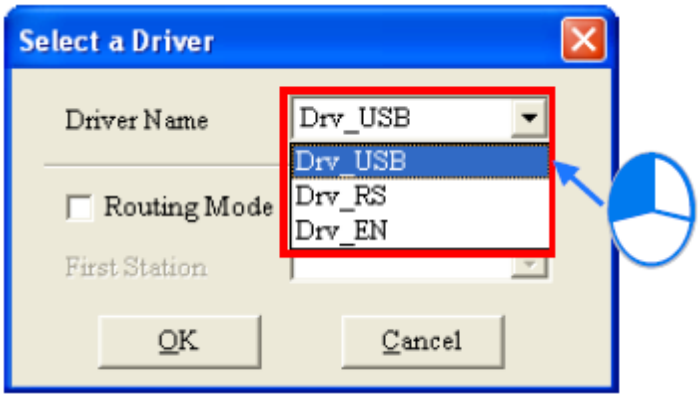
مراحل تنظیم پارامترهای ارتباطی در NWCONFIG مطابق زیر است:

۱- پس از اطمینان از اتصال کامپیوتر به PLC یا شبکه، نرم افزار COMMGR را باز کرده و درایوری در آن بسازید.

۲- در صورتی که می خواهدی با یک دستگاه ارتباط برقرار کنید، مطمئن شوید دستگاه متصل به کامپیوتر همان دستگاه مورد نظر شماست. همچنین در صورتی که می‌خواهید با چند دستگاه در شبکه ارتباط برقرار کنید، باید مطمئن شوید دستگاه‌های مورد نظر به شبکه اترنت متصل باشند، به آن‌ها IP تخصیص داده شده باشد، IP تخصیص داده شده به آن‌ها مشابه IP تنظیم شده در NWCONFIG است و شبکه مورد نظر مشابه شبکه ساخته شده در NWCONFIG باشد.

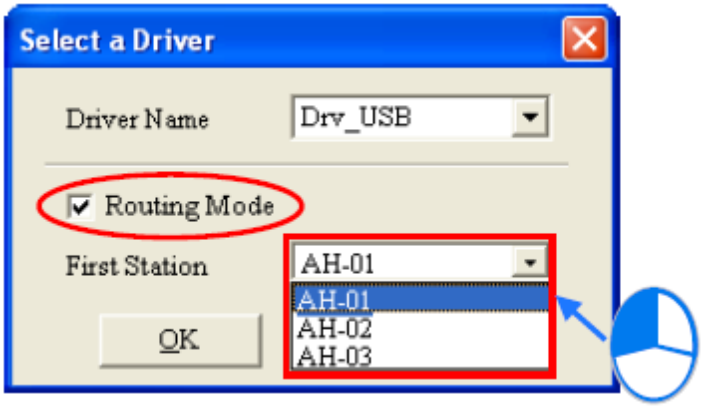
۳- گزینه  را در نوار ابزار پنجره NWCONFIG انتخاب کنید تا پنجرهCommunication Setting باز شود.

۴- در پنجره باز شده باید درایور مورد نظر خود را که از قبل در COMMGR تنظیم و آن را Start کرده‌اید را انتخاب کنید.



شکل ۱۳-۸ انتخاب درایور برای برقراری ارتباط

۵- در صورتی که کاربر می‌خواهد از مسیریابی در ارتباط خود با دیگر PLCها استفاده کند، باید تیک گزینه Routing Mode را فعال کند و دستگاه PLC که به صورت مستقیم به کامپیوتر متصل است را در First Station مشخص نماید. البته کاربر باید قبل از این فرایند حتما جدول‌های مسیریابی را در تمام گره‌های شبکه‌های مختلف دانلود نماید.



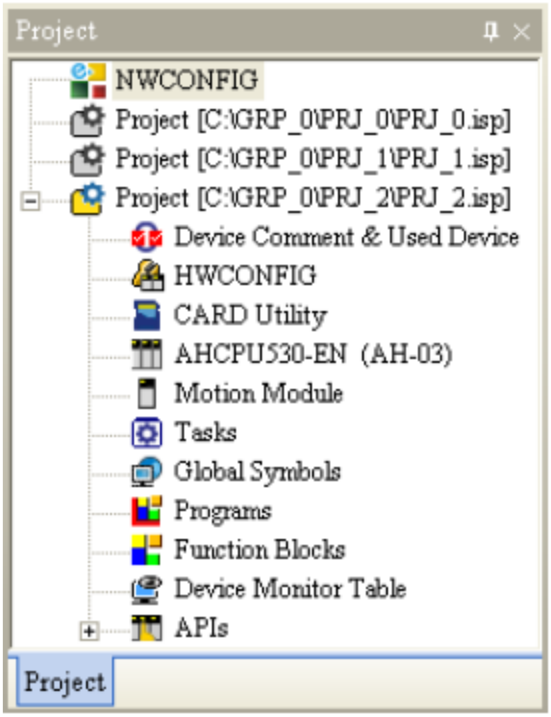
شکل ۱۳-۹ انتخاب ارتباط مسیر یابی و تعیین PLC دارای ارتباط مستقیم با نرم افزار

۱۳-۱-۳- مراحل پیاده سازی شبکه

برای ساخت شبکه‌ای که تمام عملکرد مورد نظر سیستم را پوشش دهد باید مراحلی مطابق زیر در پروژه‌های تحت ISPSoft طی شود. در این قسمت، این مراحل را تنها به صورت مختصر و بدون توجه به جزئیات برای شکل گیری تصویر کلی در ذهن خواننده مرور می کنیم.

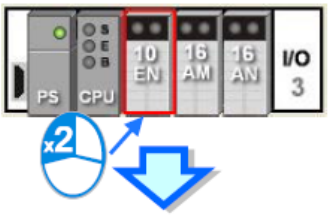
۱- قبل از آنکه سیستم مورد نظر خود را در ISPSoft پیاده کنید، باید شبکه‌های لازم را در NWCONFIG طراحی نماید. برای اینکار، کاربر باید در مورد PLCها و حافظه‌های مورد استفاده در شبکه تصمیم بگیرد، اینکه PLC به چه تجهیزات شبکه‌ای متصل خواهد شد، چگونگی اتصال گره-ها، آدرس‌های IP یا آدرس Station در RS-485 که به پورت‌ها اختصاص داده می‌شود، و همچنین مقدار پارامترهای ارتباط RS-485 نیز از دیگر مواردی هستند که کاربر باید در مورد آن‌ها تصمیم گیری نماید. همچنین کاربر باید در مورد حافظه‌هایی که تبادل داده را فراهم می‌کنند مطابق برنامه PLCها، برای پیاده سازی بخش شبکه تصمیم بگیرد، با انجام این تصمیم گیری‌ها و انجام طراحی پارامترهای شبکه، کاربر قادر به ساخت شبکه‌های مورد نظر خود در ISPSoft خواهد بود.

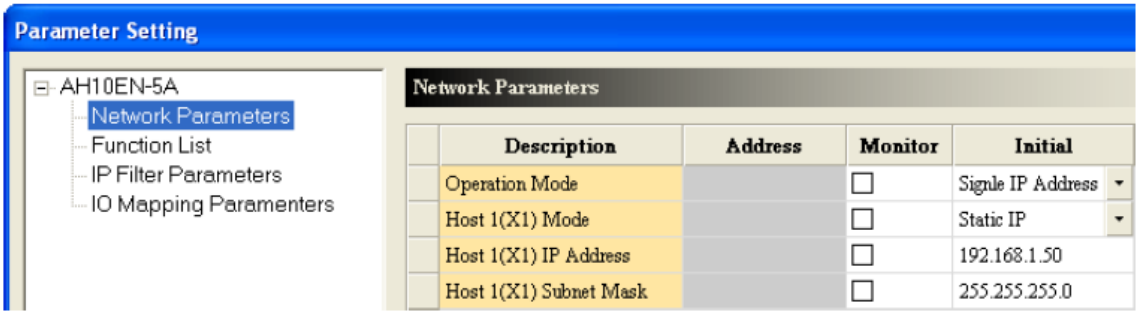
۲- ساخت پروژه در ISPSoft، در این مرحله مشخص است که در صورتی که بیش از دو PLC در سیستم وجود دارد همانند توضیحات فصل دوم، کاربر باید پروژه گروهی بسازد.



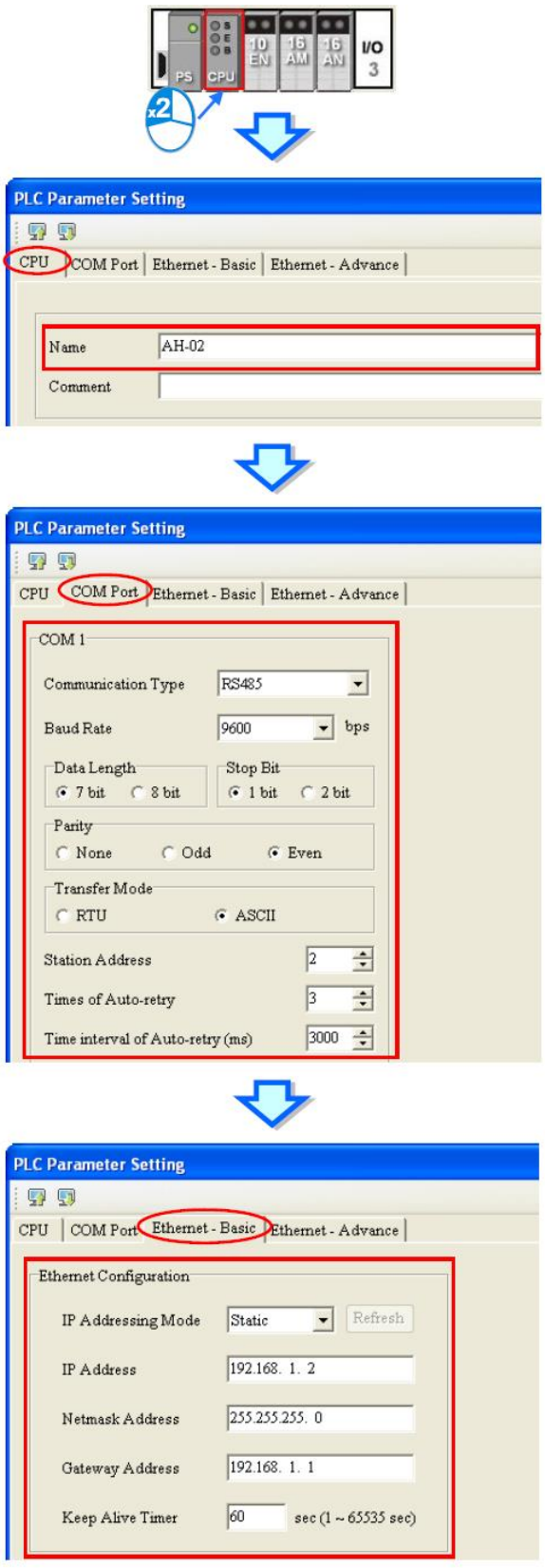
شکل ۱۳-۱۰ ساخت پروژه گروهی

۳- در صورتی که پروژه شامل PLCهای سری AH500 می‌باشد، کاربر باید HWCONFIG را باز کرده و تنظیمات سخت افزاری سیستم را در آنجا انجام دهد. کاربر در این مرحله باید تنظیمات ماژول-ها را انجام داده، پارامترهای ماژول‌های شبکه را تعیین نماید، به CPUها نام اختصاص دهد و پورت‌ها را مطابق توضیحات فصل سوم معین کند.



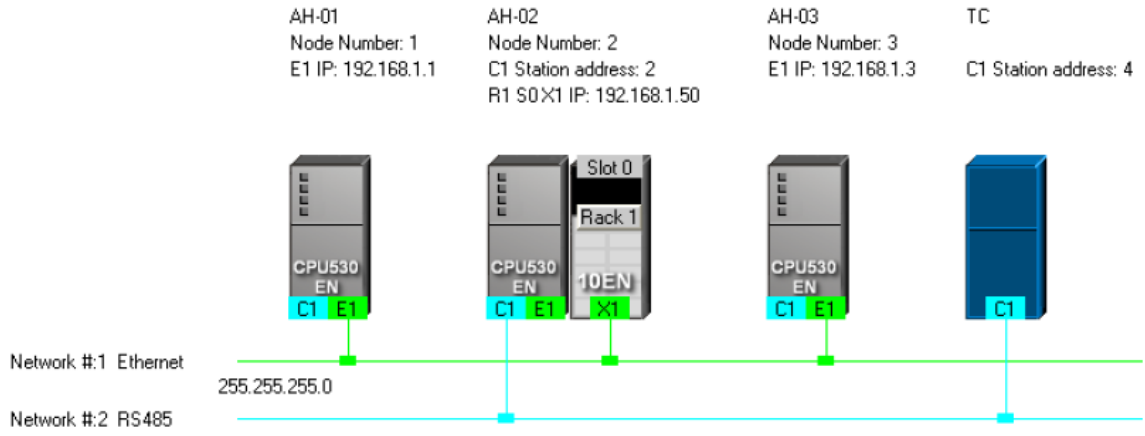


شکل ۱۳-۱۱ تنظیم پارامترها شبکه در HWCONFIG



شکل ۱۳-۱۲ تنظیم پارامترهای شبکه CPU در HWCONFIG

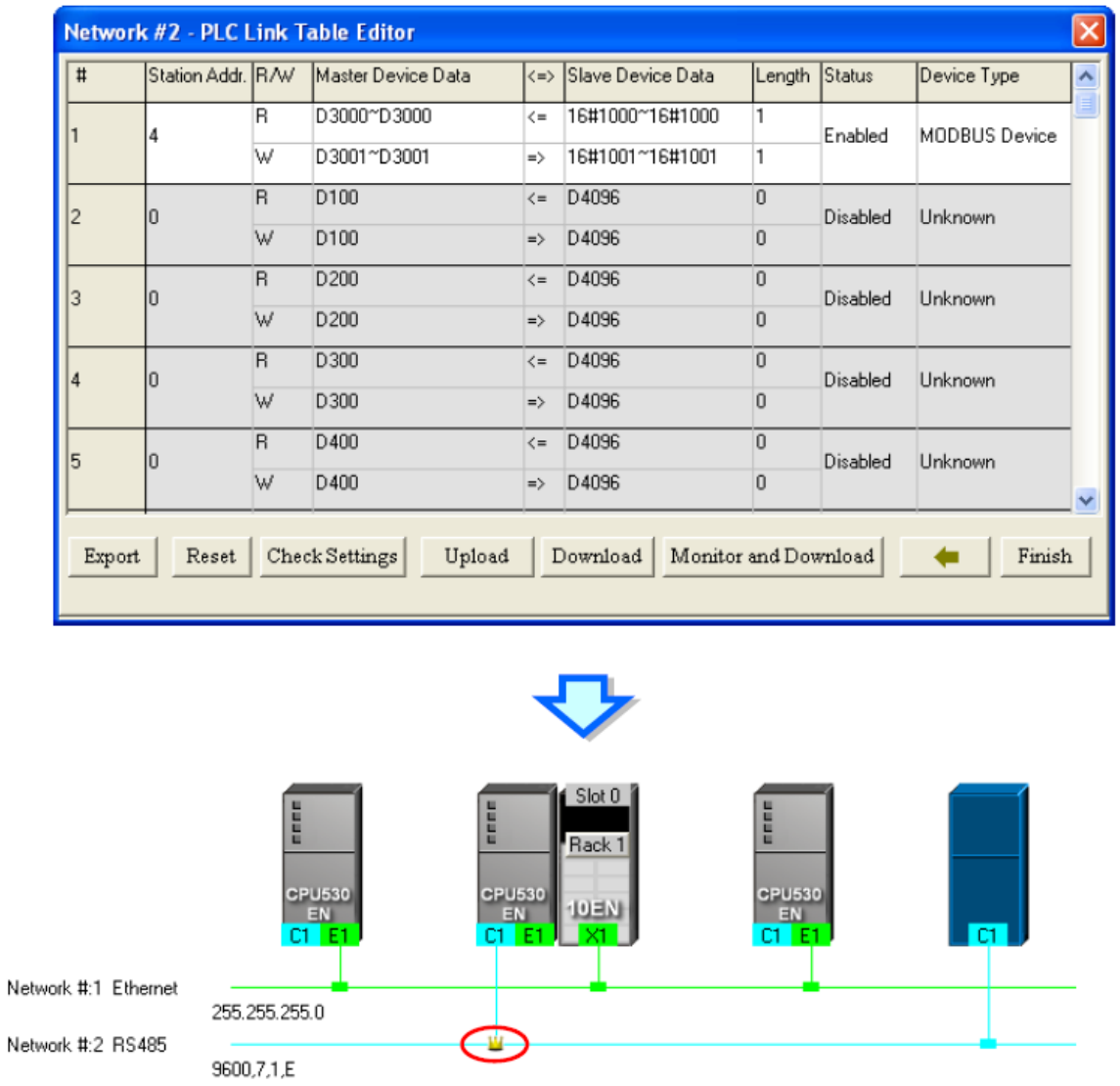
۴- کامل کردن تنظیمات شبکه اینبار در NWCONFIG



شکل ۱۳-۱۳ تنظیم پارامترهای شبکه در NWCONFIG

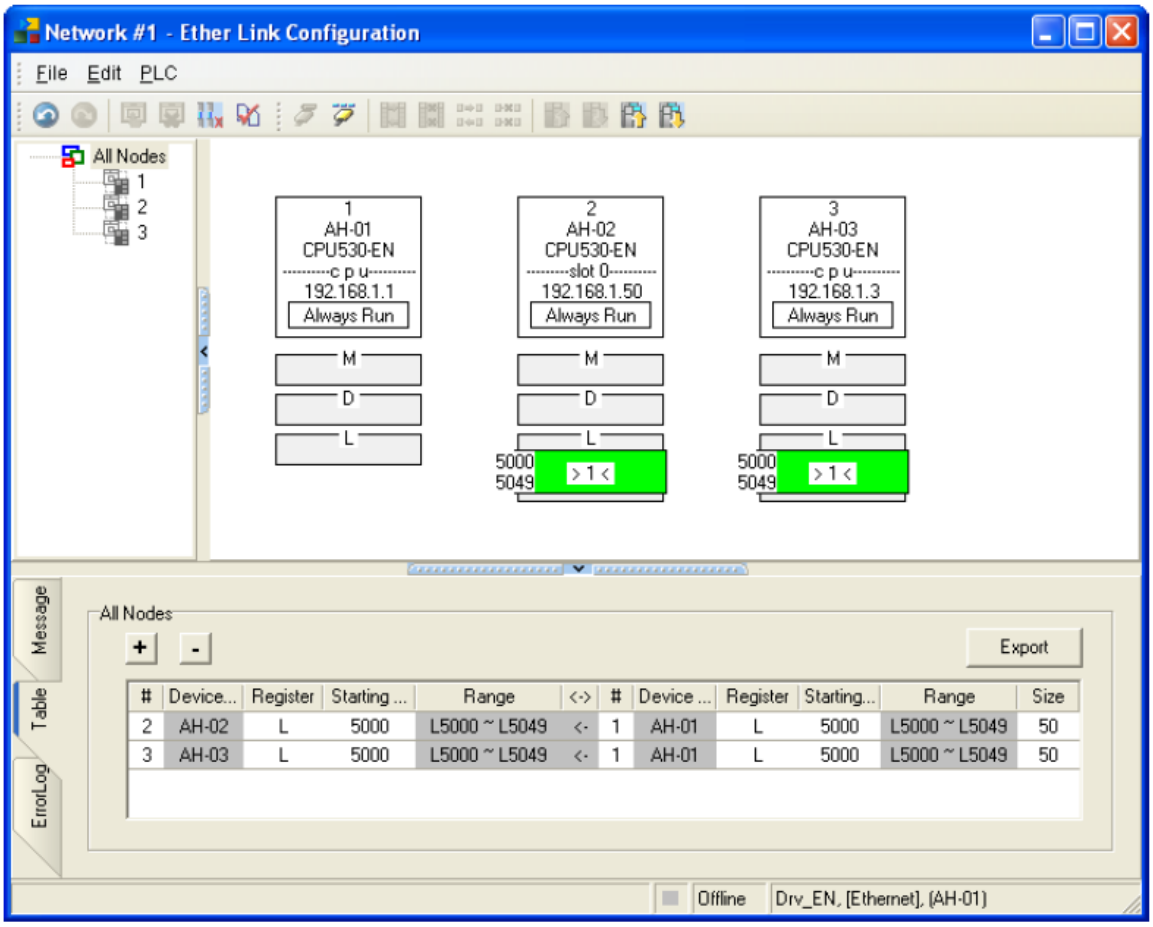
۵- در این مرحله ساخت مکانیزم تبادل داده با استفاده از مفهوم PLC LINK و Ether LINK ممکن خواهد بود. هر کدام از این دو یعنی PLC LINK و Ether LINK به صورت مستقل فعالیت می‌کنند و ترتیب در آن‌ها مهم نخواهد بود. همچنین باید توجه کرد که آدرس‌های مورد استفاده جهت تبادل داده نباید همپوشانی داشته باشند، طبیعی است که این آدرس‌ها باید مطابق با برنامه نوشته شده در هر پروژه در نظر گرفته شوند.

شکل زیر جدول مرتبط با تبادل داده در PLC Link را نمایش می‌دهد. پس از آنکه کاربر تنظیمات جدول را تکمیل کرد، ایستگاهِ Master در NWCONFIG تعیین خواهد شد.



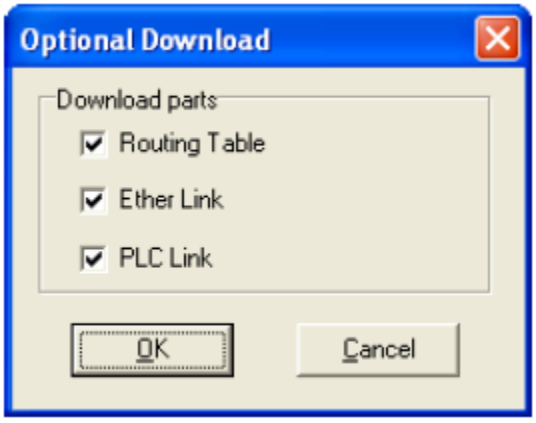
شکل ۱۳-۱۴ جدول تبادل داده در PLC Link

شکل زیر نیز مربوط به جدول تبادل داده در Ether Link می‌باشد.



شکل ۱۳-۱۵ جدول تبادل داده در Ether Link

۶- برنامه‌های پروژه ، پارامترها تنظیم شده HWCONFIG و NWCONFIG را در PLC بارگزاری نمایید. در صورتی که دستگاهی در شبکه‌ی طراحی شده شامل PLCهای سری DVP یا تجهیزاتی به جز PLCهای سری AH500 باشد، کاربر باید پارامترهای ارتباطی را در آن‌ها (گاهی به صورت مجزا و دستی) تنظیم نماید. در نهایت صفحه بارگزاری در NWCONFIG مانند صفحه زیر است که در آن می‌توان محتوای مناسب برای بارگزاری را kdcتعیین کرد.

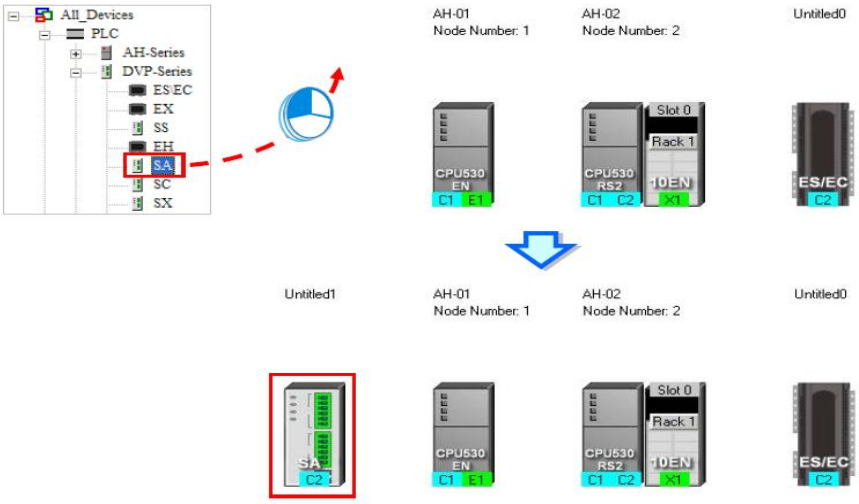


شکل ۱۳-۱۶ صفحه‌ی بارگزاری داده ها

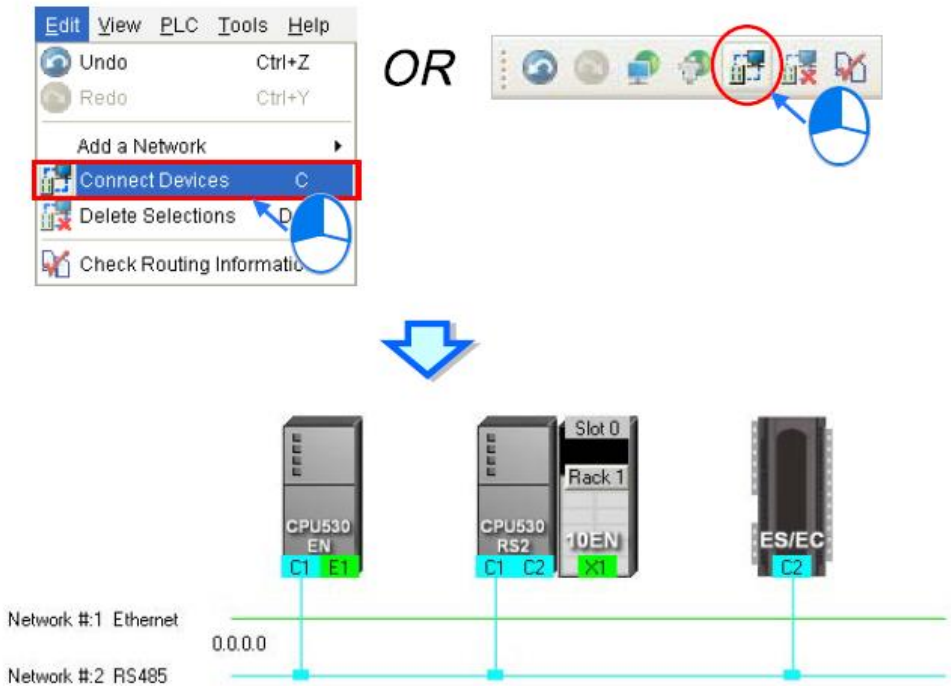
۷- قبل از شروع به کار سیستم، باید برای آن شبکه‌ای مشابه شبکه تنظیم شده فراهم شود.

۱۳-۲- ساخت معماری شبکه و دیگر نکات مربوط به NWCONFIG

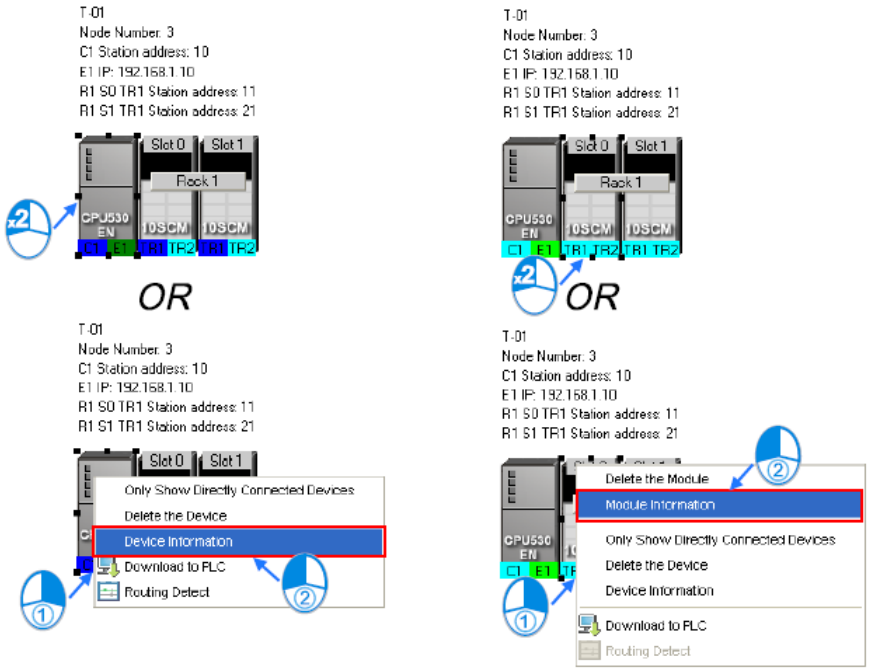
برای ساخت شبکه در NWCONFIG مراحلی چون اضافه کردن گره ها ، اضافه کردن شبکه‌های صنعتی و اتصال گره‌ها به شبکه‌ها، تنظیم پارامترهای ماژول‌ها وشبکه‌های صنعتی، تنظیم جداول انتقال داده و مسیریابی و بارگزاری در سیستم باید در نظر گرفته شود، همچنین امکاناتی جهت بررسی وضعیت شبکه و ویرایش آن در NWCONFIG وجود دارد که شما برای دسترسی به این اطلاعات می‌توانید به راهنمای نرم افزار و یا کتاب شبکه‌های صنعتی دلتا (که به زودی منتشر خواهد شد) مراجعه نمایید. در ادامه جهت مرور این روند تصاویر نمونه‌ای از مراحل ساخت معماری شبکه در NWCONFIG را مشاهده می‌کنید.



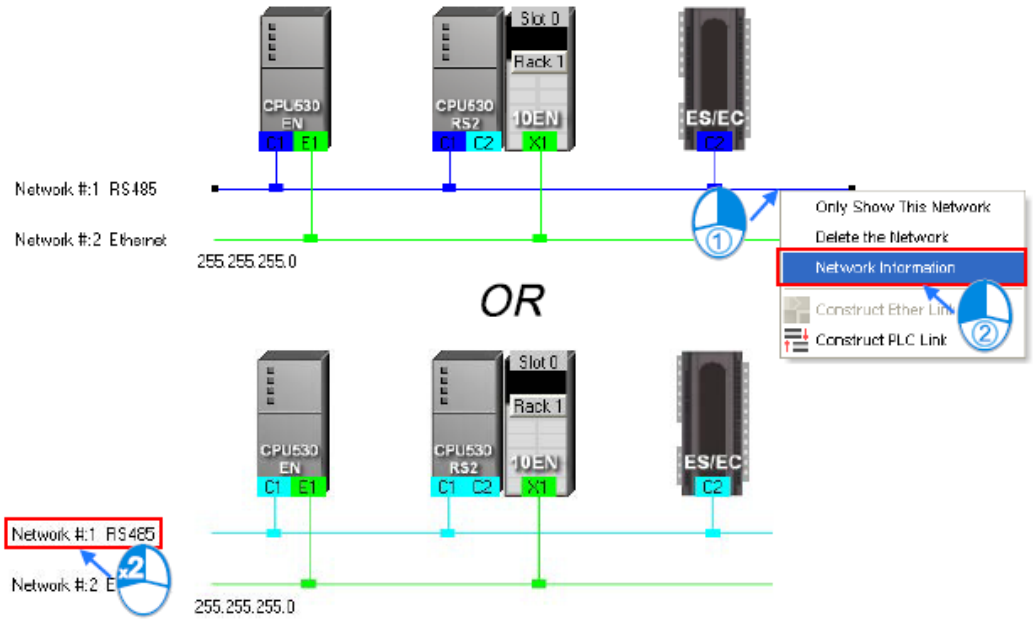
شکل ۱۳-۱۷ اضافه کردن گره‌ها در NWCONFIG



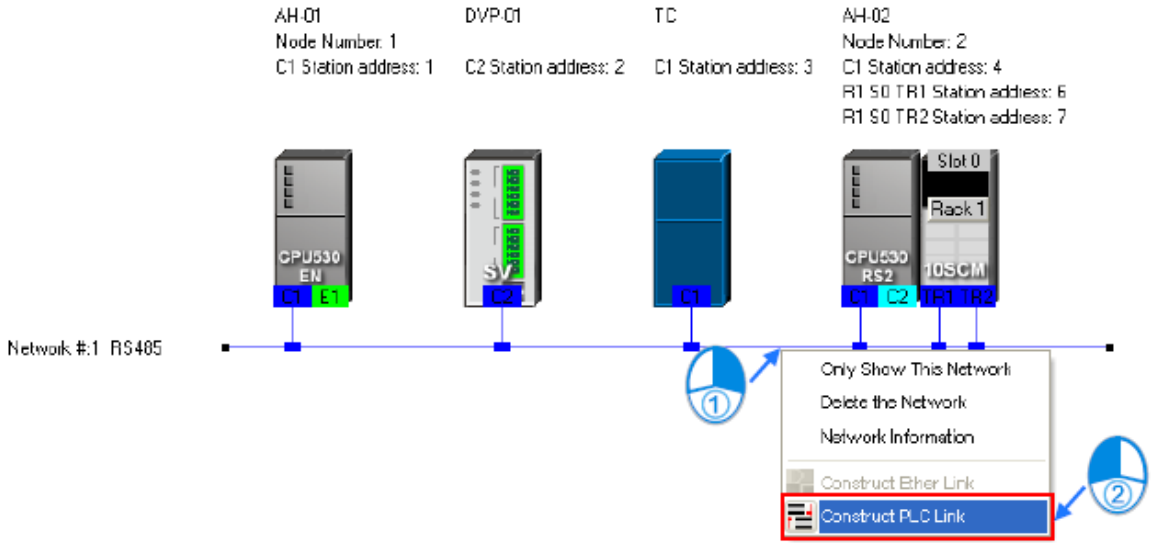
شکل ۱۳-۱۸ ایجاد شبکه و اتصال گره ها به شبکه ها در NWCONFIG



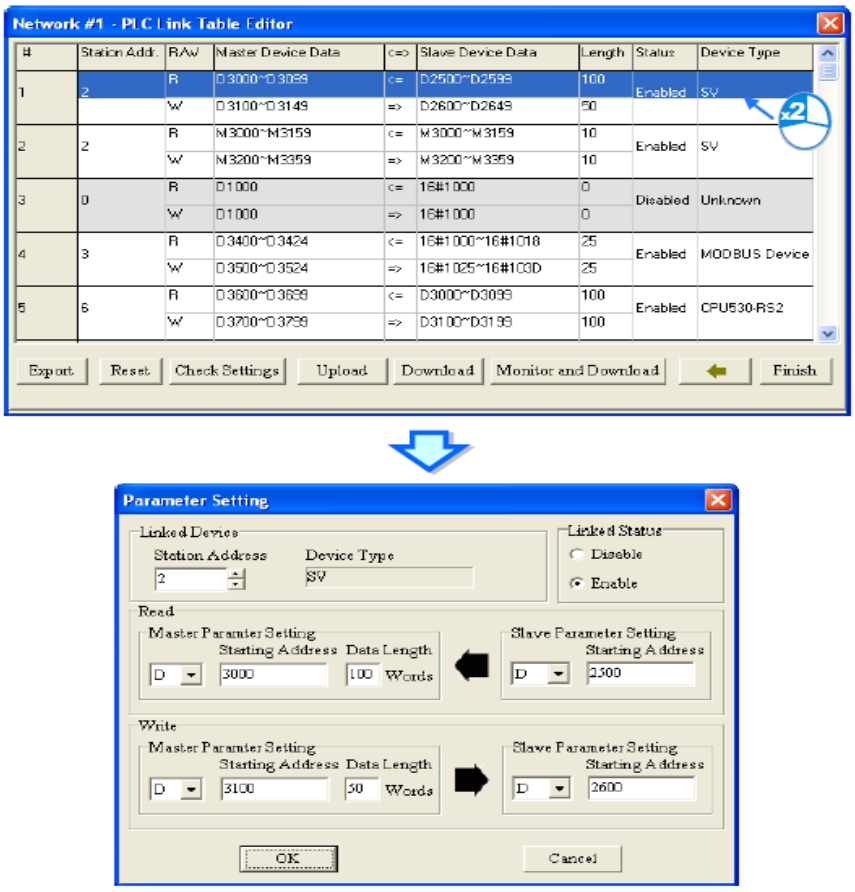
شکل ۱۳-۱۹ تنظیمات گره ها در NWCONFIG



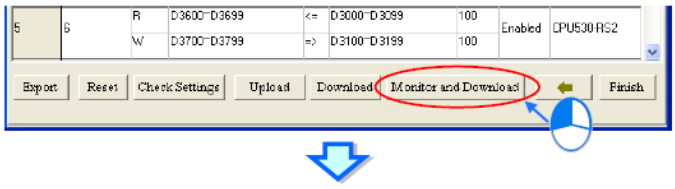
شکل ۱۳-۲۰ تنظیمات شبکه ها در NWCONFIG

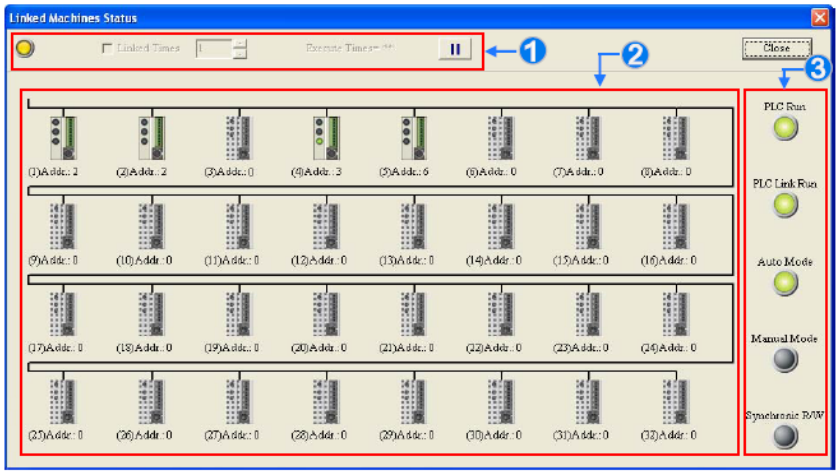


شکل ۱۳-۲۱ ایجاد PLC Link در NWCONFIG

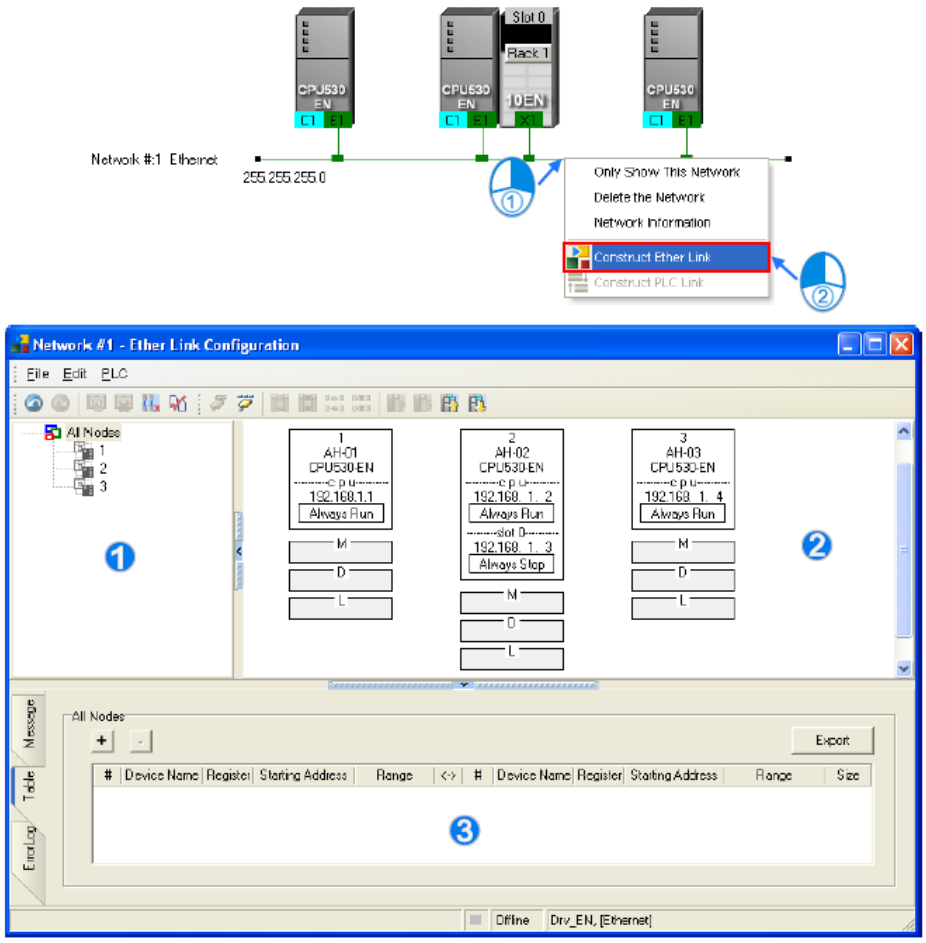


شکل ۱۳-۲۲ تنظیمات جدول تبادل داده PLC Link در NWCONFIG

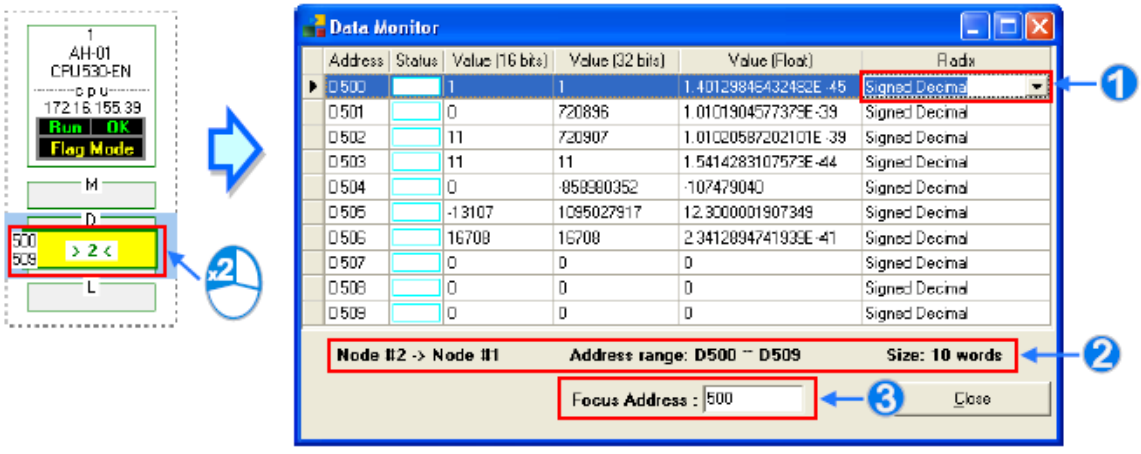




شکل ۱۳-۲۳ مانیتورینگ آنلاین PLC Link در در NWCONFIG



شکل ۱۳-۲۴ ایجاد و تنظیمات Ether Link در NWCONFIG

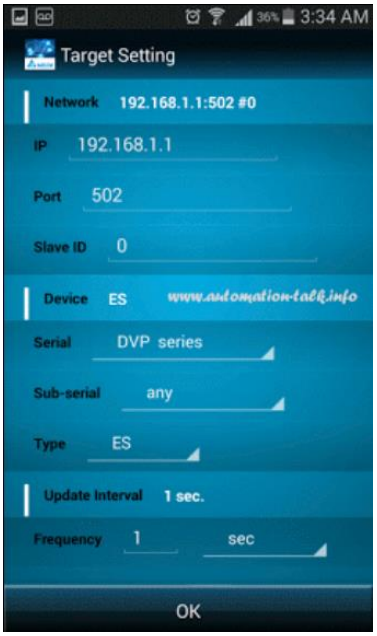


شکل ۱۳-۲۵ مانیتورینگ آنلاین Ether Link و داده‌ها در NWCONFIG

۱۳-۳- دیگر امکانات شبکه دلتا

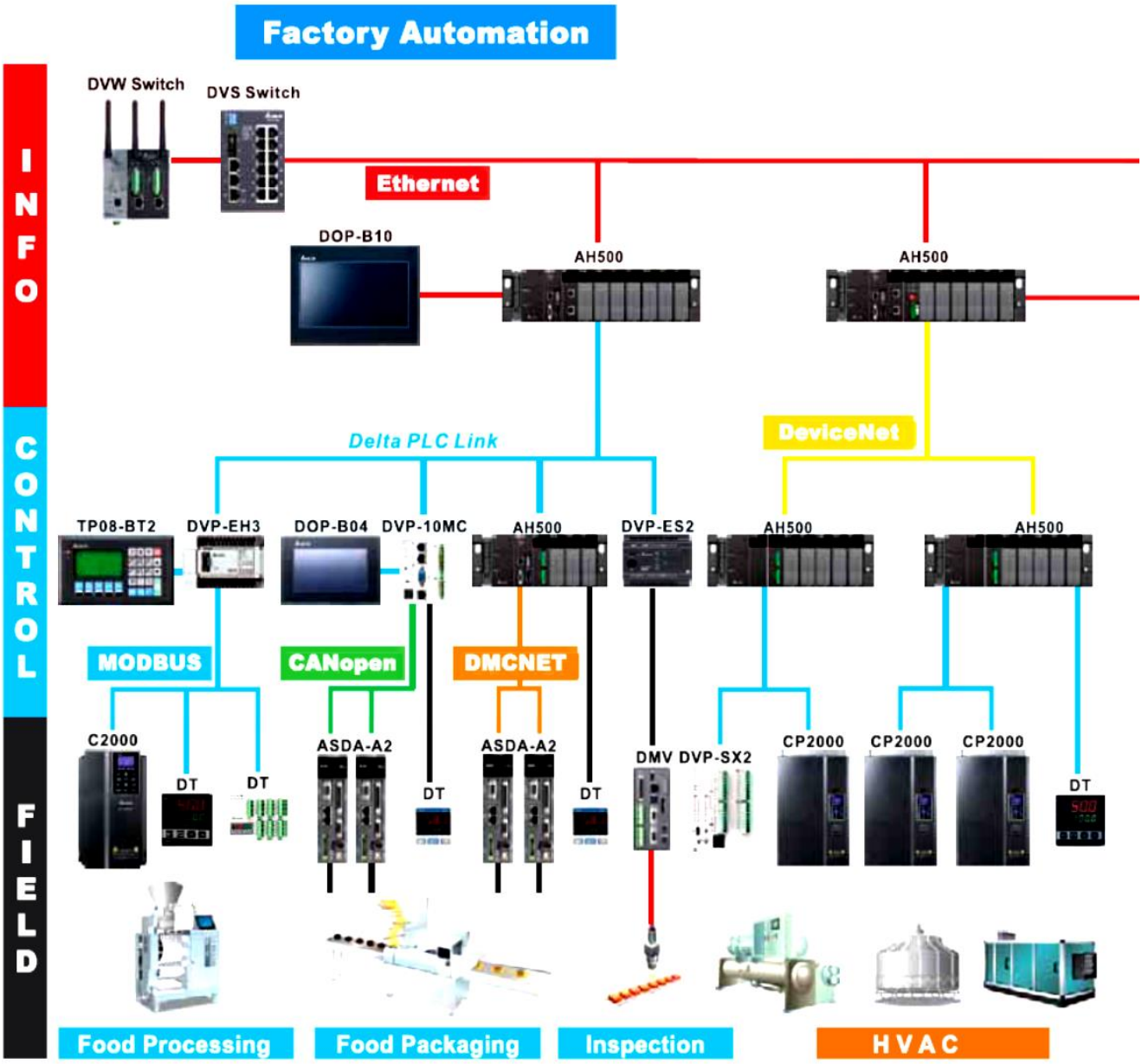
با توجه به آنکه کمپانی دلتا ارائه دهنده General Solution در زمینه اتوماسیون در حوزه‌های مختلف است و محصولات آن در تمام سطوح Field، کنترل و مانیتورینگ و Supervisory گسترده است، می‌توان انتظار داشت که راه حل جامعی برای ارتباط بین اجزای مختلف این سیستم به وسیله شبکه‌های صنعتی در نظر گرفته شده باشد. دو صفحه آتی که از کاتالوگ محصولات شرکت دلتا گرفته شده است، این امر را به خوبی نشان داده و نقش شبکه‌های صنعتی در ایجاد ارتباط بین اجزای اتوماسیون دلتا را به روشنی نمایش می‌دهد.

همچنین دلتا نرم افزارهای رایگانی را نیز جهت کار با شبکه های صنعتی ارائه می‌دهد. به عنوان نمونه DeviceNet Builder برای کار با شبکه DeviceNet و یا CANopen Builder و EDS Builder برای کار با شبکه CAN و یا DCISoft که برای تنظیم پارامترهای ماژول‌های شبکه دلتا می‌تواند مورد استفاده قرار - گیرد. همچنین یکی از محصولات جدید این کمپانی، نرم افزار رایگان اندرویدی Smart VIEwer است که با استفاده از آن از طریق شبکه WIFI می‌توان حافظه‌های PLC را مانیتور کرد و از آن‌ها Trend گرفت، این ویژگی برای بررسی وضعیت سیستم و مانیتور کردن آن بسیار مناسب است و قدرت مانور بیشتری را در اختیار مهندس و یا اپراتور سیستم قرار می‌دهد.



شکل ۱۳-۲۶ نمایی از نرم افزار Smart VIEwer کمپانی دلتا

Industrial Automation Solutions



Ethernet

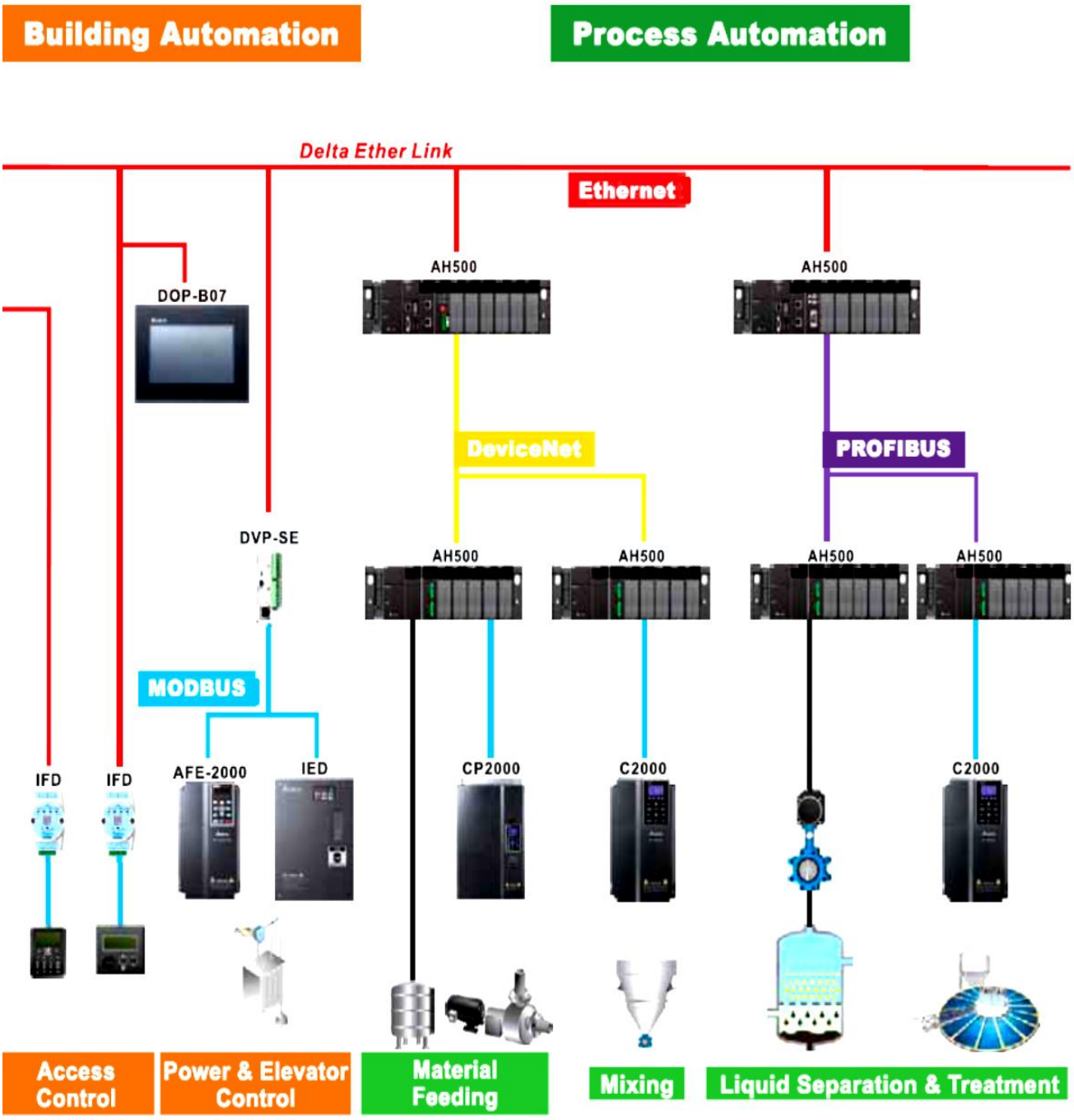
Delta Ethernet products transcend the limits of transmission distance, offering 10/100Mbps high-speed transmission and efficient remote monitoring.

CANopen

Delta CANopen products support CANopen DS301 and DSP402 protocols, and are able to achieve multi-axis, high-speed and complex motion control with max. speed 1Mbps.

DMCNET

Delta DMCNET offers 10Mbps communication speed, constructing a real time control system which supports multi-axis synchronous motion. The system can be connected to servo motors, remote digital or analog I/O modules, step motors, DD motors, linear motors, MPG modules, and more.



MODBUS

Delta MODBUS serial products integrate easily with devices of other brands, and for communication among RS-232, RS-422, RS-485 and custom-defined formats, offering greater flexibility for on-site applications.

DeviceNet

Delta DeviceNet products support interconnections among products of different brands and wire-saving network topology. The 500kbps stable and noise resistant fieldbus data transmission is suitable for harsh industrial sites.

PROFIBUS

Delta PROFIBUS products support 12Mbps communication speed and are suitable for distributed automated industrial control networks.

ضمیمه أ – مروری بر خانواده های DVP و AH500

در این ضمیمه اطلاعات کلی محصولات دلتا برای انتخاب محصول مناسب از کاتالوگ‌های مربوطه آن آورده شده است.

۱– خانواده DVP

Model Name Explanation

• PLC

DVP ○○□□○○□○ (Series ❶ ❷ ❸ ❹❺)

1. Total I/O
2. Model
   ES / ES2 : ES / ES2 series PLC
   EX / EX2 : EX / EX2 series PLC
   SS / SS2 : SS / SS2 series PLC
   SA / SA2 : SA / SA2 series PLC
   SX / SX2 : SX / SX2 series PLC
   SC : SC series PLC
   SV : SV series PLC
   SE : SE series PLC
   PM : PM series PLC
   MC : MC series PLC
   EH : EH series PLC
   EC : EC series PLC
3. Power supply
   00 : AC power input
   11 : DC power input
4. Output type
   R : Relay
   T : Transistor (NPN)
   M : Mixed with differential signal
   S : Transistor (PNP)
   RC : Relay + CANopen
   TC : Transistor + CANopen
5. Version

• PI/PO Module

DVP ○○□□- □○ (Series ❶ ❷ ❸)

1. Total I/O
2. Model
   HC : High-speed counter
   PU : Single-axis positioning module
3. Model
   S : SS / SA / SX / SC / SV / SS2 / SA2 SX2 / SV2 / SE / MC series PLC
   H2 / H3 : EH2 / EH3 / PM series PLC
   SL : left-side extension for SV series PLC

• Accessory: Other

DVP A BT ○○ (Series ❶ ❷ ❸)

1. Accessory
2. Type
   Bt : Battery
3. Type : 01, 02 .....

• DI/DO Module

DVP ○○□□○○□ (Series ❶ ❷❸ ❹ ❺)

1. Total I/O
2. Model
   X : ES / EX / ES2 / EX2 series PLC
   S : SS / SA / SX / SC / SV / SS2 / SA2 / SX2 / SV2 / SE / MC series PLC
   H : EH2 / EH3 / PM series PLC
3. I/O type
   M : Input point
   N : Output point
   P : Input + output
4. Power supply
   00 : AC power input
   11 : DC power input
5. Output type
   R : Relay
   T : Transistor (NPN)
   TS : Transistor (PNP
   N : None output

• Peripherals

DVP □□□ ○○ (Series ❶ ❷)

1. Product name
   DU : Digital display panel
2. Type / function
   01 : Type 01
   02 : Type 02
   03 : Type 03

• Function Card

DVP - F ○○○ (Series ❶ ❷ ❸)

1. Function Card
2. Type
   232 : RS-232 card
   422 : RS-422 card
   485 : RS-485 card
   2AD : 2ch analog input
   2DA : 2ch analog output
3. Particular definition
   S : Slave mode
   (applicable to COM3 coding only)

• AI/AO Module

DVP ○○□□- □○ (Series ❶ ❷ ❸)

1. Total I/O
2. Model
   AD : Analog / digital conversion
   DA : Digital / analog conversion
   PT : PT type temperature module
   TC : Thermocouple type temperature Module
   XA : AD + DA module
   LC : Load cell module
   RC : Resolver module
3. Model
   S or S2 : SS / SA / SX / SC / SV / SS2 SA2 / SX2 / SV2 / SE / MC series PLC
   H2 or H3 : EH2 / EH3 / PM series PLC
   SL : left-side extension for S series PLC
   E2 : ES2 / EX2 series PLC

• Network Module

DVP □□○○- □○ (Series ❶ ❷)

1. Model
   EN01 : MODBUS TCP
   DNET : DeviceNet master
   COPM : CANopen master
   CP02 : CANopen Slave
   DT01 / 02 : DeviceNet Slave
   PF01 / 02 : PROFIBUS DP Slave
2. Model
   S : SS / SA / SX / SC / SV / SS2 / SA2 SX2 / SV2 / SE / MC series PLC
   H2 / H3 : EH2 / EH3 / PM series PLC
   SL : left-side extension for S series PLC

• Accessory: Cable

DVP A CAB ○○○ (Series ❶ ❷ ❸ ❹)

1. Accessory
2. Type
   CAB : Cable
3. Type
   1, 2, 3, 4, ...
4. Length
   15 : 1.5m
   30 : 3.0m

Select A Suitable PLC

Select your desired specifications and locate the most suitable PLC.

What do you need? → Select specifications → Check → Locate PLC

Item	Specifications	Check	Model							
			ES2	EX2	EH3	SS2	SA2	SX2	SV2	SE
Power supply	AC	☐	◎	◎	○					
	DC	☐				○	○	○	○	○
I/O points	< 256	☐	△	△						
	< 512	☐			△	△	△	△	△	△
Program capacity	< 8k	☐				○				
	< 16k	☐	○	○			○	○		○
	< 32k	☐			○				○	
Output type	Transistor (NPN)	☐	◎	◎	◎	◎	◎	◎	◎	◎
	Transistor (PNP)	☐				◎	△	◎	◎	△
	Relay	☐	◎	◎	◎	◎	◎	◎	◎	◎
	Differential signal	☐			◎					
Communication	3 COM ports (RS-232/485)	☐	○	○	△		○	△	△	△
	Ethernet	☐			△		△	△	△	○
	USB	☐						○		○
	DeviceNet	☐			△*1		△*1	△*1	△*1	△*1
	CANopen	☐	◎		△*1		△*1	△*1	△*1	△*1
	PROFIBUS	☐			△*1		△*1	△*1	△*1	△*1
Positioning	2-axis output	☐	○	○	○	○	○	○		○
	4-axis output	☐			◎				○	
	> 4 axes	☐			△	△	△	△	△	△
	2-axis interpolation	☐	○	○	○		○	○	○	○
	100kHz high speed	☐	○	○			○	○		○
	200kHz high speed	☐			○	△	△	△	○	△
High-speed counting	≤ 2 channels	☐	○	○		○	○	○		○
	≥ 3 channels	☐			○*3	△	△	△	○	△
	100kHz high speed	☐	○	○			○	○		○
	200kHz high speed	☐			○	△	△	△	○	△
Analog function	< 4 channels (AD)	☐	△	○	△	△	△	○	△	△
	< 2 channels (DA)	☐	△	○*2	△	△	△	○*2	△	△

Note:
○: With such specification, ◎: Varies upon model, △: With such specification when connected to extension module/function card
*1 : Series that support left-side modules supports master and slave, other series support only slave
*2 : EX/SX2 series have 4 channels of analog input and 2 channels of analog output
*3 : Besides the built-in 4 channels of high-speed counters, EH3 series can be connected to high-speed counter modules

Ordering Information

ES/EX Series PLC

Product name	Power supply	Output method	Inputs	Outputs	Model name	Certificates
ES series standard PLC	100 ~ 240VAC	Relay	8	6	DVP14ES00R2	CE c UL us
	100 ~ 240VAC	Transistor	8	6	DVP14ES00T2	
	100 ~ 240VAC	Relay	16	8	DVP24ES00R2	
	100 ~ 240VAC	Transistor	16	8	DVP24ES00T2	
	100 ~ 240VAC	Relay	18	12	DVP30ES00R2	
	100 ~ 240VAC	Relay	16	16	DVP32ES00R2	
	100 ~ 240VAC	Transistor	16	16	DVP32ES00T2	
	100 ~ 240VAC	Relay	24	16	DVP40ES00R2	
	100 ~ 240VAC	Transistor	24	16	DVP40ES00T2	
	100 ~ 240VAC	Relay	36	24	DVP60ES00R2	
	100 ~ 240VAC	Transistor	36	24	DVP60ES00T2	
EX series analog PLC	100 ~ 240VAC	Relay	8	6	DVP20EX00R2	
		Analog	4	2		
	100 ~ 240VAC	Transistor	8	6	DVP20EX00T2	
		Analog	4	2		

ES/EX Series Digital Module

Product name	Output method	Inputs	Outputs	Model name	Certificates
Digital module	-	8	-	DVP08XM11N	CE c UL us
	Relay	-	8	DVP08XN11R	
	Transistor	-	8	DVP08XN11T	
	-	16	-	DVP16XM11N	
	Relay	-	16	DVP16XN11R	
	Transistor	-	16	DVP16XN11T	
	Relay	-	24	DVP24XN11R	
	Transistor	-	24	DVP24XN11T	
	Relay	4	4	DVP08XP11R	
	Transistor	4	4	DVP08XP11T	
	Relay	16	8	DVP24XP11R	
	Transistor	16	8	DVP24XP11T	
	Relay	16	16	DVP32XP11R	
	Transistor	16	16	DVP32XP11T	

EC3 Series PLC

Product name	Power supply	Output method	Inputs	Outputs	Model name	Certificates
EC3 series standard PLC	100 ~ 240VAC	Relay	6	4	DVP10EC00R3	CE c UL us
	100 ~ 240VAC	Transistor	6	4	DVP10EC00T3	
	100 ~ 240VAC	Relay	8	6	DVP14EC00R3	
	100 ~ 240VAC	Transistor	8	6	DVP14EC00T3	
	100 ~ 240VAC	Relay	8	8	DVP16EC00R3	
	100 ~ 240VAC	Transistor	8	8	DVP16EC00T3	
	100 ~ 240VAC	Relay	12	8	DVP20EC00R3	

EC3 Series PLC

Product name	Power supply	Output method	Inputs	Outputs	Model name	Certificates
EC3 series standard PLC	100 ~ 240VAC	Transistor	12	8	DVP20EC00T3	CE, cULus
	100 ~ 240VAC	Relay	12	12	DVP24EC00R3	
	100 ~ 240VAC	Transistor	12	12	DVP24EC00T3	
	100 ~ 240VAC	Relay	18	12	DVP30EC00R3	
	100 ~ 240VAC	Transistor	18	12	DVP30EC00T3	
	100 ~ 240VAC	Relay	16	16	DVP32EC00R3	
	100 ~ 240VAC	Transistor	16	16	DVP32EC00T3	
	100 ~ 240VAC	Relay	24	16	DVP40EC00R3	
	100 ~ 240VAC	Transistor	24	16	DVP40EC00T3	
	100 ~ 240VAC	Relay	28	20	DVP48EC00R3	
	100 ~ 240VAC	Transistor	28	20	DVP48EC00T3	
	100 ~ 240VAC	Relay	36	24	DVP60EC00R3	
	100 ~ 240VAC	Transistor	36	24	DVP60EC00T3	
Fastest execution time of basic instructions		3.8μs	Execution time of MOV instruction		5.04μs	

ES2/EX2 Series PLC

Product name	Power supply	Output method	Inputs	Outputs	Model name	Certificates
ES2 series standard PLC	100 ~ 240VAC	Relay	8	8	DVP16ES200R	CE, cULus
	100 ~ 240VAC	Transistor	8	8	DVP16ES200T	
	100 ~ 240VAC	Relay	16	8	DVP24ES200R	
	100 ~ 240VAC	Transistor	16	8	DVP24ES200T	
	100 ~ 240VAC	Relay	16	16	DVP32ES200R	
	100 ~ 240VAC	Transistor	16	16	DVP32ES200T	
	240VAC	Relay	16	16	DVP32ES211T	
	100 ~ 240VAC	Transistor	24	16	DVP40ES200R	
	100 ~ 240VAC	Relay	24	16	DVP40ES200T	
	100 ~ 240VAC	Transistor	36	24	DVP60ES200R	
	100 ~ 240VAC	Relay	36	24	DVP60ES200T	
ES2 series built-in CANopen PLC	100 ~ 240VAC	Transistor	36	24	DVP32ES200RC	
	100 ~ 240VAC	Relay	36	24	DVP32ES200TC	
EX2 series analog PLC	100 ~ 240VAC	Relay	8	6	DVP20EX200R	
		Analog	4	2		
	100 ~ 240VAC	Transistor	8	6	DVP20EX200T	
		Analog	4	2		
EX2 series temperature/ analog PLC	100 ~ 240VAC	Relay	16	10	DVP30EX200R	
		Analog	3	1		
	100 ~ 240VAC	Transistor	16	10	DVP30EX200T	
		Analog	3	1		
Fastest execution time of basic instructions		0.35μs	Execution time of MOV instruction		3.4μs	

ES2/EX2 Series Digital I/O Module (AC power supply)

Product name	Power supply	Output method	Inputs	Outputs	Model name	Certificates
ES2/EX2 digital module	100 ~ 240VAC	Relay	-	24	DVP24XN200R	CE, cULus
	100 ~ 240VAC	Transistor	-	24	DVP24XN200T	
	100 ~ 240VAC	Relay	16	8	DVP24XP200R	
	100 ~ 240VAC	Transistor	16	8	DVP24XP200T	
	100 ~ 240VAC	Relay	16	16	DVP32XP200R	
	100 ~ 240VAC	Transistor	16	16	DVP32XP200T	

Ordering Information

ES2/EX2 Series Digital/Analog Module (DC24V)

Product name	Output method	Inputs	Outputs	Model name	Certificates
ES2/EX2 series digital module	-	8	-	DVP08XM211N	CE, cULus
	Relay	-	8	DVP08XN211R	
	Transistor	-	8	DVP08XN211T	
	Relay	4	4	DVP08XP211R	
	Transistor	4	4	DVP08XP211T	
	-	16	-	DVP16XM211N	
	Relay	-	16	DVP16XN211R	
	Transistor	-	16	DVP16XN211T	
	Relay	8	8	DVP16XP211R	
	Transistor	8	8	DVP16XP211T	
ES2/EX2 series analog I/O module	• 4 points of analog voltage (10V, 5V) / current (20mA, 0 ~ 20mA, 4 ~ 20mA) input *1 • Resolution: 14-bit (-32,000 ~ +32,000)			DVP04AD-E2	
	• 4 points of analog voltage (-10V ~ +10V) / current (0 ~ 20mA, 4 ~ 20mA) output*1 • Resolution: 14-bit (-32,000 ~ +32,000) / (0 ~ +32,000)			DVP04DA-E2	
	• 2 points of analog voltage (-10V ~ +10V) / current (0 ~ 20mA, 4 ~ 20mA) output *1 • Resolution: 14-bit (-32,000 ~ +32,000) / (0 ~ +32,000)			DVP02DA-E2	
	• 4 points of analog voltage (10V, 5V) / current (20mA, 0 ~ 20mA, 4 ~ 20mA) input *1 • Input resolution: 14-bit (-32,000 ~ +32,000) • 2 points of analog voltage (-10V ~ +10V) / current (0 ~ 20mA, 4 ~ 20mA) output • Output resolution: 14-bit (-32,000 ~ +32,000) / (0 ~ +32,000)			DVP06XA-E2	
ES2/EX2 series temperature measurement module	• 4 points of platinum RTD (Pt100, Pt1000, Ni100, Ni1000) sensor input/ 0 ~ 300Ω resistance input *1 • Resolution: 16-bit • With PID temperature control			DVP04PT-E2	
	• 4 points of thermocouple (J, K, R, S, T, E, N Type) sensor input/-80mV ~ +80mV voltage input *1 • Resolution: 20-bit • With PID temperature control			DVP04TC-E2	
Absolute resolver module	• Converts 1 set of resolver input signal (angle/speed) into digital signals • Resolution: 12-bit • Supports disconnection detection for distance up to 50m			DVP10RC-E2*2	

*1. Digital/analog photocoupler isolation. No isolation among channels.
*2. Contact Delta sales representative or distributors for the official launch date.

EH2/EH3 Series PLC

Product name	Power supply	Output method	Inputs	Outputs	Model name	Certificates
EH2/EH3 series standard PLC	100 ~ 240VAC	Relay	8	8	DVP16EH00R3	CE, cULus
	100 ~ 240VAC	Transistor	8	8	DVP16EH00T3	
	100 ~ 240VAC	Relay	12	8	DVP20EH00R3	
	100 ~ 240VAC	Transistor	12	8	DVP20EH00T3	
	100 ~ 240VAC	Transistor	16	16	DVP32EH00T3	
	100 ~ 240VAC	Relay	16	16	DVP32EH00R3	
	100 ~ 240VAC	Differential	16	16	DVP32EH00M3	
	100 ~ 240VAC	Relay	16	16	DVP32EH00R3-L	
	100 ~ 240VAC	Transistor	16	16	DVP32EH00T3-L	
	100 ~ 240VAC	Transistor	24	16	DVP40EH00T3	
	100 ~ 240VAC	Relay	24	16	DVP40EH00R3	
	100 ~ 240VAC	Relay	24	24	DVP48EH00R3	
	100 ~ 240VAC	Transistor	24	24	DVP48EH00T3	
	100 ~ 240VAC	Relay	32	32	DVP64EH00R3	
	100 ~ 240VAC	Transistor	32	32	DVP64EH00T3	
	100 ~ 240VAC	Relay	40	40	DVP80EH00R3	
	100 ~ 240VAC	Transistor	40	40	DVP80EH00T3	
Execution time of basic instructions			0.24μs			

EH2/EH3 Series Digital/Analog Module

Product name	Output method	Inputs	Outputs	Model name	Certificates
Digital module	Relay	4	4	DVP08HP11R	CE cULus
	Transistor	4	4	DVP08HP11T	
	Relay	-	8	DVP08HN11R	
	Transistor	-	8	DVP08HN11T	
	-	8	-	DVP08HM11N	
	Relay	8	8	DVP16HP11R	
	Transistor	8	8	DVP16HP11T	
	-	16	-	DVP16HM11N	
	-	32	-	DVP32HM11N	
	Relay	-	32	DVP32HN00R	
	Transistor	-	32	DVP32HN00T	
	Relay	16	16	DVP32HP11R	
	Transistor	16	16	DVP32HP11T	
	Relay	24	24	DVP48HP00R	
	Transistor	24	24	DVP48HP00T	
Analog module	• 4 points of analog voltage (-10V ~ +10V)/current (-20mA ~ +20mA) *1 • Input resolution: 14-bit • Built-in RS-485 interface			DVP04AD-H2	
	• 4 points of analog voltage (0V ~ +10V)/current (0mA ~ +20mA) output *1 • Resolution: 12-bit • Built-in RS-485 interface			DVP04DA-H2	
	• 4 points of analog voltage (-10V ~ +10V)/current (-20mA ~ +20mA) input • 2 points of analog voltage (0V ~ +10V)/current (0mA ~ +20mA) output • Resolution: 12-bit • Built-in RS-485 interface			DVP06XA-H2	
	• 4 points of platinum RTD (PT100) sensor input *1 • Resolution: 0.1°C • Built-in RS-485 interface			DVP04PT-H2	
	• 4 points of thermocouple (J, K, R, S, T type) sensor input *1 • Resolution: 0.1°C • Built-in RS-485 interface			DVP04TC-H2	
	• 8 points of thermocouple (J, K, R, S, T type) sensor input *1 • Resolution: 0.1°C • Built-in RS-485 interface			DVP08TC-H2	
	• 4 channels of differential voltage (-10V ~ +10V) / current (-20mA ~ +20mA) input • Resolution: 16-bit • Built-in RS-485 interface			DVP04AD-H3	
	• 4 channels of voltage (-10V ~ +10V) / current (0 ~ +20mA) output • Resolution: 16-bit • Built-in RS-485 interface			DVP04DA-H3	
	• 4 channels of differential voltage (-10V ~ +10V)/ current (-20mA ~ +20mA) input • 2 channels of voltage (-10V ~ +10V) / current (0 ~ +20mA) output • Resolution: 16-bit • Built-in RS-485 interface			DVP06XA-H3	

*1. Digital/analog photocoupler isolation. No isolation among channels.

EH2/EH3 Series Extension Module/Function Card

Product name	Description	Model name	Certificates
Positioning module	Servo position control module (single axis, 200kHz)	DVP01PU-H2	CE cULus
High-speed counter	High-speed counter module (1CH)	DVP01HC-H2	
Communication module	PROFIBUS DP slave communication module	DVPPF02-H2	
	CANopen slave communication module	DVPCP02-H2	
	DeviceNet slave communication module	DVPDT02-H2	
Function card	RS-232 port conversion (EH2: COM2; EH3: COM3)	DVP-F232	
	RS-422 port conversion (EH2: COM2; EH3: COM3)	DVP-F422	
	RS-485 port extension (COM3), (DVP-EH3 only)	DVP-F485	
	• 2 points of analog voltage (0 ~ 10V)/current (0 ~ 20mA) input • Resolution: 12-bit	DVP-F2AD	
	• 2 points of analog voltage (0 ~ 10V)/current (0 ~ 20mA) output • Resolution: 12-bit	DVP-F2DA	
	Ethernet communication card	DVP-FEN01	
Digital display panel	Displays data in register and real time clock	DVPDU01	

Ordering Information

S Series PLC

Product name	Power supply	Output method	Inputs	Outputs	Model name	Certificates
SV2 series functional PLC	24VDC	Relay	16	12	DVP28SV11R2	CE cULus
	24VDC	Transistor (NPN)	16	12	DVP28SV11T2	
	24VDC	Transistor (PNP)	16	12	DVP28SV11S2	
	24VDC	Transistor (NPN)	10 (2AI)	12	DVP24SV11T2	
Execution time of basic instructions			0.24μs			
SS2 series standard PLC	24VDC	Relay	8	6	DVP14SS211R	CE cULus
	24VDC	Transistor (NPN)	8	6	DVP14SS211T	
	24VDC	Transistor(PNP)	8	4	DVP12SS211S	
SA2 series advanced PLC	24VDC	Relay	8	4	DVP12SA211R	
	24VDC	Transistor	8	4	DVP12SA211T	
SX2 series analog PLC	24VDC	Relay	8 (4AI)	6(2AO)	DVP20SX211R	
	24VDC	Transistor (NPN)	8 (4AI)	6(2AO)	DVP20SX211T	
	24VDC	Transistor (PNP)	8 (4AI)	6(2AO)	DVP20SX211S	
Fastest execution time of basic instructions		0.35μs	Execution time of MOV instruction		3.4μs	
SE network PLC	24VDC	Relay	8	4	DVP12SE11R	CE cULus
	24VDC	Transistor	8	4	DVP12SE11T	
Fastest execution time of basic instructions		0.64μs	Execution time of MOV instruction		2μs	
SX series analog PLC	24VDC	Relay	4 (2AI)	2 (2AO)	DVP10SX11R	CE cULus
	24VDC	Transistor	4 (2AI)	2 (2AO)	DVP10SX11T	
Fastest execution time of basic instructions		3.8μs	Execution time of MOV instruction		5.04μs	

S Series Digital/Analog Module

Product name	Output method	Inputs	Outputs	Model name	Certificates
Digital module	Relay	-	6	DVP06SN11R	CE cULus
	Relay	-	8	DVP08SN11R	
	Transistor	-	8	DVP08SN11T	
	Transistor	-	16	DVP16SN11T	
	Relay	4	4	DVP08SP11R	
	Transistor	4	4	DVP08SP11T	
	-	8	-	DVP08SM11N	
	-	8	-	DVP08SM10N	
	Transistor (PNP)	-	8	DVP08SN11TS	
	Digital switch	8	-	DVP08ST11N	
	Relay	8	8	DVP16SP11R	
	Transistor (PNP)	4	4	DVP08SP11TS	
	Transistor (NPN)	8	8	DVP16SP11T	
	Transistor (PNP)	8	8	DVP16SP11TS	
	Transistor (PNP)	-	16	DVP16SN11TS	
	-	16	-	DVP16SM11N	
	Transistor, Latched connector	-	32	DVP32SN11TN	
	Latched connector	32	-	DVP32SM11N	

Product name	Description		Model name	Certificates
Analog I/O module	• 4 points of analog input voltage (-10V ~ +10V)/ current (-20mA ~ +20mA)*1	• Input resolution: 14-bit (can switch to 16-bit) • Built-in RS-485 interface	DVP04AD-S2	CE cULus
	• 4 points of analog input voltage (0V ~ +10V)/current (0mA ~ +20mA) *1	• Resolution: 14-bit (can switch to 16-bit) • Built-in RS-485 interface	DVP04DA-S2	
	• Analog input+output module (6 points) • 4 points of analog input voltage (-10V ~ +10V)/ current (-20mA ~ +20mA)	• 2 points of analog output voltage (0V ~ +10V)/ current (0mA ~ +20mA) • Input/output resolution: 12-bit (can switch to 16-bit) • Built-in RS-485 interface	DVP06XA-S2	
	• 4 points of analog input voltage (-10V ~ +10V)/ current (-20mA ~ +20mA)*1 • Input resolution: 14-bit	• Built-in RS-485 interface	DVP04AD-S	

S Series Analog Module

Product name	Description	Model name	Certificates
Analog I/O module	• 4 points of analog output voltage (0V ~ +10V)/ current (0mA ~ +20mA)*1 • Output resolution: 14-bit • Built-in RS-485 interface	DVP04DA-S	CE cULus
	• 2 points of analog output voltage (0V ~ +10V)/ current (0mA ~ +20mA) *1 • Output resolution: 12-bit • Built-in RS-485 interface	DVP02DA-S	
	• 6 points of analog input voltage (-10V ~ +10V)/ current (-20mA ~ +20mA) *1 • Input resolution: 14-bit • Built-in RS-485 interface	DVP06AD-S	
	• Analog input+output modules (6 points) • 4 points of analog input voltage (-10V ~ +10V)/ current (-20mA ~ +20mA) • 2 points of analog output voltage (0V ~ +10V)/ current (0mA ~ +20mA) • Input/output resolution: 12-bit • Built-in RS-485 interface	DVP06XA-S	

S Series Extension Module/Left-Side High-Speed Module

Product name	Description	Model name	Certificates
Left-side high-speed analog I/O module	• 4 groups of analog input *1 • Signal range: 1 ~ 5V, 0 ~ 5V, -5 ~ 5V, 0 ~ 10V, -10 ~ 10V, 4 ~ 20mA, 0 ~ 20mA, -20 ~ 20mA • Resolution: 16-bit • Single channel On/Off setup enhances entire conversion efficiency • Conversion time: 250μs/point • Off-line alarm (1 ~ 5 V, 4 ~ 20mA)	DVP04AD-SL	CE cULus
	• 4 groups of analog output *1 • Signal range: 0 ~ 10V, -10 ~ 10V, 4 ~ 20mA, 0 ~ 20mA • Resolution: 16-bit • Offers single channel On/Off setup • Conversion time: 250μs/point	DVP04DA-SL	
Left-side high-speed load cell module	• Supports 2 channels of load cell signal input*1 • Resolution: 20-bit • Connectable to 4-wire/6-wire load cell sensor • Measurable range: 0 ~ 6mV/V	DVP02LC-SL	
	• Supports 1 channel of load cell signal input*1 • Resolution: 20-bit • Connectable to 4-wire/6-wire load cell sensor • Measurable range: 0 ~ 6mV/V	DVP01LC-SL	
Temperature measurement module	• 6 points of platinum RTD (PT100, PT1000, NI100, NI1000) sensor input • Resolution: 0.1°C	DVP06PT-S	
	• 4 points of platinum RTD (PT100, PT1000, NI100, NI1000) sensor input*1 (Version 4.06 and above supports PT1000, NI100, NI1000) • Resolution: 0.1°C • Built-in RS-485 interface	DVP04PT-S	
	• 4 points of thermocouple (J、K、R、S、T type) sensor input*1 • Resolution: 0.1°C • Built-in RS-485 interface	DVP04TC-S	
Positioning module	Servo position control module (single axis, 200kHz)	DVP01PU-S	
Communication module	DeviceNet slave communication module	DVPDT01-S	
	PROFIBUS DP slave communication module	DVPPF01-S	
Left-side high-speed communication module	Ethernet communication module, 10/100Mbps	DVPEN01-SL	
	DeviceNet master communication module, 500kbps	DVPDNET-SL	
	CANopen master communication module, 1Mbps	DVPCOPM-SL	
	PROFIBUS DP slave communication module , 12Mbps	DVPPF02-SL	
	RS-485/RS-422, serial communication module, 460kbps	DVPSCM12-SL	
	BACnet MS/TP Slave communication module, 460kbps	DVPSCM52-SL	
Remote I/O module	RS-485 remote I/O module, connectable to S series I/O modules	RTU-485	
	Ethernet remote I/O module, connectable to S series I/O modules	RTU-EN01	
	DeviceNet remote I/O module, connectable to S series I/O modules	RTU-DNET	
	PROFIBUS remote I/O module, connectable to S series I/O modules	RTU-PD01	

*1. Digital/analog photocoupler isolation. No isolation among channels

Ordering Information

Communication Converter

Product name	Description	Model name	Certificates
Converter	USB to RS-485 converter	IFD6500	CE cULus
	USB to CAN converter	IFD6503	
	USB to RS-485 converter	IFD6530	
	MODBUS TCP to RS-232/485 converter	IFD9506	
	EtherNet/IP to RS-232/485 converter	IFD9507	
	DeviceNet to RS-232/485 converter	IFD9502	
	CANopen to RS-232/485 converter	IFD9503	
	RS-232 to RS-422/485 isolated converter	IFD8500	
	RS-485 to RS-422 isolated repeater	IFD8510	
	RS-422/485 to RS-232 addressable isolated converter	IFD8520	

PM Series

Product name	Power supply	Output method	Inputs	Outputs	Model name	Certificates
General purpose motion controller	100 ~ 240VAC	Differential	16	16	DVP10PM00M	CE cULus
		(Built-in 4-axis of independent 1MHz pulse output)				
Professional motion controller	100 ~ 240VAC	Differential	8	8	DVP20PM00DT	
		(Built-in 2-axis of independent 500kHz pulse output)			DVP20PM00D	
		(Built-in 3-axis of independent 500kHz pulse output)			DVP20PM00M	

PM series extension module	Description	Model name
DVP-PM Communication card	Ethernet/CANopen communication card	DVP-FPMC
DVP-PM Memory card	Data backup memory card (64k words)	PM-PCC01

Execution time of basic instructions	0.13μs	Execution time of MOV instruction	3.74μs

MC Series

Product name	Power supply	Output method	Inputs	Outputs	Model name	Certificates
Network type motion controller	24VDC	CANopen DS402	8	4	DVP10MC11T	CE cULus

TP Series

Product name	Description						Model name	Certificates
TP02	Resolution: 160 x 32, Serial COM ports: RS-232 & RS-485						TP02G-AS1	CE cULus
TP04	Resolution: 128 x 64, Serial COM ports: RS-232 & RS-422/RS-485						TP04G-AS2	
	Resolution: 192 x 64, Serial COM ports: RS-232 & RS-422/RS-485						TP04G-AL2	
	Resolution: 192 x 64, Serial COM ports: RS-232						TP04G-AL-C	CE
	Resolution: 192 x 64, Serial COM ports: RS-232 & RS-422/RS-485, 0 ~ 9 numeric keys available						TP04G-BL-C	
TP04P	Resolution: 192 x 64, Serial COM ports: USB & RS-485	8DI	8DO	-	-	Relay	TP04P-16TP1R	CE cULus
		8DI	16DO	-	-	Relay	TP04P-32TP1R	
		8DI	8DO	4AI	2AO	Relay	TP04P-22XAIR	
		8DI	8DO	2AI	1AO	Relay	TP04P-21EX1R	
TP08	Resolution: 240 x 128, Serial COM ports: RS-232 & RS-422/RS-485,0 ~ 9 numeric keys available						TP08G-BT2	

Peripheral Accessories

Product name	Description	Model name	Certificates
Accessory	Data backup memory card (DVP-EH3 only)	DVP-512FM	CE cULus
	Data backup memory card (64k words)	DVP-PCC01	
	Communication cable for PC (9-pin & 25-pin D-Sub) and PLC, 3m	VCVDVPACAB230	
	Communication cable for PC (9-pin D-Sub) and PLC, 3m	DVPACAB2A30	
	Communication cable for PC (9-pin D-Sub) and PLC (90° bend), 1m	DVPACAB2B10	
	I/O connection cable for DVP-32SM series	DVPACAB7A10	
	I/O connection cable for DVP-32SN series	DVPACAB7B10	
	External terminal module for DVP-32SM series (32 inputs)	DVPAETB-ID32A	
	External terminal module for DVP-32SN series (16 outputs)	DVPAETB-OR16A	
	Supports 4 types of RS-485 connectors	ADP485-01	
	Connection cable for ADP485-01 and ASDA-A series servo	ADPCAB03A	
	Connection cable for ADP485-01 and ASDA-B series servo	ADPCAB03B	
	I/O extension cable for ES/EX series, 0.3m	DVPACAB403	
	Extension cable for EH series PLC and extension module, 0.7m	DVPACAB4A07	
	DeviceNet/CANopen distribution box, 1 for 2	TAP-CN01	
	DeviceNet/CANopen distribution box, 1 for 4	TAP-CN02	
	DeviceNet/CANopen distribution box, 1 for 4 RJ45	TAP-CN03	
	CANopen sub-line, RJ45 connector, 0.3m	TAP-CB03	
	CANopen sub-line, RJ45 connector, 0.5m	TAP-CB05	
	CANopen sub-line, RJ45 connector, 1m	TAP-CB10	
	CANopen sub-line, RJ45 connector, 2m	TAP-CB20	
	CANopen sub-line, RJ45 connector, 3m	TAP-CB30	
	CANopen sub-line, RJ45 connector, 10m	TAP-CB100	
	3.6V lithium battery (unchargeable) for EH/SX series PLC	DVPABT01	
	Terminal resistance for CANopen communication	TAP-TR01	
	TP Programmer Cable	DVPACAB530	

Software

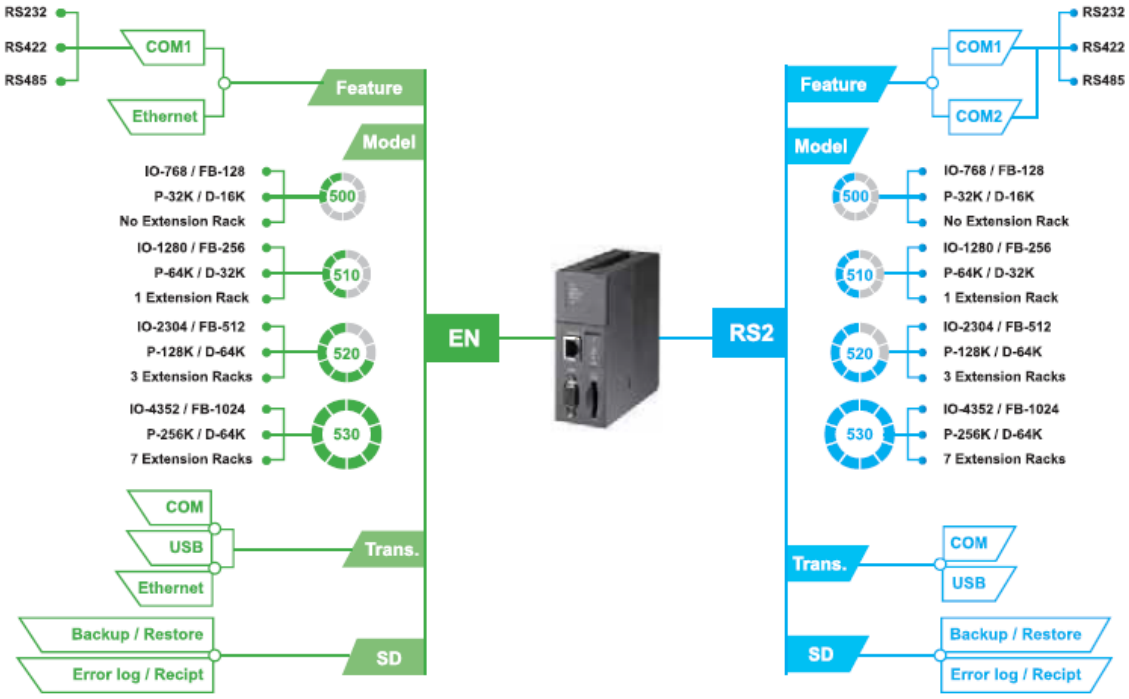
Product name	Description	OS (Windows based software)
ISPSoft	PLC editing software for AH500 and DVP series (supports 5 programming languages: LD, FBD, SFC, ST, IL)	Windows 2000, XP, Vista, Windows 7 (32-bit/64-bit)
WPLSoft	Programming software for DVP-PLC	Windows 98, Me, NT4.0, 2000, XP, Vista, Windows 7 (32-bit/64-bit)
TPEditor	Editing software for TP series text panel	Windows 98, Me, NT4.0, 2000, XP, Vista, Windows 7 (32-bit/64-bit)
PMSoft	Programming software for PM series	Windows 2000, XP, Vista, Windows 7 (32-bit/64-bit)
DCISoft	Delta communication integration software	Windows 2000, XP, Vista, Windows 7 (32-bit/64-bit)
DeviceNet Builder	DeviceNet configuration software	Windows 2000, XP, Vista, Windows 7 (32-bit/64-bit)
CANopen Builder	CANopen configuration software	Windows 2000, XP, Vista, Windows 7 (32-bit/64-bit)
DMT	VB, VC, DLL library for DVP-PLC	Windows 2000, XP, Vista, Windows 7 (32-bit/64-bit)

Industrial Power Supply

Series	Power supply	Inputs	Outputs	Power	Output current	Model name	Certificates
DVP	1-phase	85 ~ 264VAC	DC24V	24W	1A	DVPPS01	CE cULus
				48W	2A	DVPPS02	
				120W	5A	DVPPS05	

* 【Note】For more ordering information, please refer to the catalogue for Delta Industrial Power Supply.

Specification Tree



CPU Selection Table

Item	Specifications	Check	CPU Model							
			AH500-RS2	AH510-RS2	AH520-RS2	AH530-RS2	AH500-EN	AH510-EN	AH520-EN	AH530-EN
Local I/O points	< 768	☐	•				•			
	< 1280	☐		•				•		
	< 2304	☐			•				•	
	< 4352	☐				•				•
Program capacity	< 32k steps	☐	•				•			
	< 64k steps	☐		•				•		
	< 128k steps	☐			•				•	
	< 256k steps	☐				•				•
Expansion capacity	None	☐	•				•			
	< 1 expansion rack	☐		•				•		
	< 3 expansion racks	☐			•				•	
	< 7 expansion racks	☐				•				•
Built-in communication	1 COM port	☐					•	•	•	•
	2 COM ports	☐	•	•	•	•				
	Ethernet	☐					•	•	•	•
	Mini-USB	☐	•	•	•	•	•	•	•	•
SD card	V 1.0	☐	•	•	•	•	•	•	•	•

Model Name Explanation

AH CPU

AHCPU500-RS2

AH	CPU	5	0	0	-	RS2
Series	**Classification** CPU	**Model**	**Function** 0: No expansion rack 1: 1 expansion rack 2: 3 expansion racks 3: 7 expansion racks	**Version**		**Type** RS2: 2 COM ports EN: 1 COM & 1 Ethernet ports

AH Power Supply Module

AHPS05-5A

AH	PS	05	-	5A
Series	**Classification** Power supply	**Function** 05: AC input (100~240V) 15: DC input (24V)		**Type**

AH Backplane

AHBP04M1-5A

AH	BP	04	M1	-	5A
Series	**Classification** Backplane	**Function** 04: 4-slot 06: 6-slot 08: 8-slot 12: 12-slot	**Function** M1: Main backplane E1: Expansion backplane		**Type**

AH Analog I/O Module

AH04AD-5A

AH	04	AD	-	5A
Series	**I/O Channels** 04: 4-channel 06: 6-channel 08: 8-channel	**Classification** AD: Analog input DA: Analog output AX: Analog input / output		**Type** 5A: Voltage / Current 5B: Voltage 5C: Current

AH Digital I/O Module

AH16AM10N-5A

AH	16	AM	1	0	N	-	5A
Series	**I/O points** 16: 16 points 32: 32 points 64: 64 points	**Classification** AM: Digital input AN: Digital output AP: Digital input / output AR: Digital input with interrupt	**Function** 0: No input 1: DC input (24V) 3: AC input (120~240V)	**Function** 0: No output 1: 0.5A transistor / TRIAC output or 2A relay output 2: 0.1A transistor output	**Function** N: No output R: Relay output T: NPN output P: PNP output S: TRIAC output		**Type** 5A: Removable terminal 5B: DB37 connector 5C: Latched connector

AH Temperature Module

AH04PT-5A

AH	04	PT	-	5A
Series	**I/O channels** 04: 4-channel 08: 8-channel	**Classification** PT: Platinum resistance thermometer TC: Thermocouple PTG: Platinum resistance thermometer (channel isolation)		**Type**

AH Network Module

AH10EN-5A

AH	10	EN	-	5A
Series	**Function**	**Classification** EN: Ethernet SCM: Serial communication DNET: DeviceNet PFBM: PROFIBUS master PFBS: PROFIBUS slave COPM: CANopen		**Type**

AH RTU Module

AHRTU-DNET-5A

AH	RTU	-	DNET	-	5A
Series	**Classification** Remote terminal unit		**Function** DNET: DeviceNet PFBS: PROFIBUS		**Type**

AH Motion Module

AH02HC-5A

AH	02	HC	-	5A
Series	**Function** 02: 2-channel 04: 4-channel 05: Simple type (PM) 10: Standard type (PM) 15: Advance type (PM) 20: DMCNET	**Classification** HC: High speed counter PM: Motion controller (Pulse train) MC: Motion controller (Network)		**Type**

Ordering Information

CPU Modules

Model	Local I/O points	Program capacity	Data register D / L / B (note)	Function blocks	Extension backplane	Power consumption (Internal)	Specifications	Certificates
AHCPU500-RS2	768	32K steps (128KB)	16K / 16K / 512K words	128	0	2w	▪ Built-in RS-232 / 422 / 485 multi-modes communication port x 2 (RS-232:115.2kbps / RS-422 / 485: 921.6kbps) ▪ Built-in SD card slot (supports max. 2 GB) ▪ Built-in Mini-USB programming port ▪ Program execution speed: LD instruction @ 0.1μs / 1K steps @ 0.3ms ▪ System diagnose / status light / online editing and debug functions ▪ PLC Link automatic data exchange function ▪ MODBUS RTU / ASCII ▪ LD / SFC / FBD / IL / ST languages ▪ 256 interrupts (Timed / IO / External / Low voltage / Communication) ▪ 2048 timers and counters ▪ No battery required ▪ RTC function (max. 30 days after power off)	CE cULus
AHCPU510-RS2	1280	64K steps (256KB)	32K / 32K / 1024K words	256	1	2w		
AHCPU520-RS2	2304	128K steps (512KB)	64K / 64K / 2048K words	512	3	2w		
AHCPU530-RS2	4352	256K steps (1MB)	64K / 64K / 4096K words	1024	7	2w		
AHCPU500-EN	768	32K steps (128KB)	16K / 16K / 512K words	128	0	2w	▪ Built-in RS-232 / 422 / 485 multi-modes communication port x1 (RS-232:115.2kbps / RS-422 / 485: 921.6kbps) ▪ Built-in Ethernet communication port (100Mbps) ▪ Built-in SD card slot (supports max. 2 GB) ▪ Built-in Mini-USB programming port ▪ Program execution speed: LD instruction @ 0.1μs / 1K steps @ 0.3ms ▪ System diagnose / status light / online editing and debug functions ▪ PLC Link automatic data exchange function ▪ MODBUS RTU / ASCII ▪ LD / SFC / FBD / IL / ST languages ▪ 256 interrupts (Timed / IO / External / Low voltage / Communication) ▪ 2048 timers and counters ▪ No battery required ▪ RTC function (max. 30 days after power off) ▪ NTP network time correction function ▪ WEB / E-mail / IP Filter function	
AHCPU510-EN	1280	64K steps (256KB)	32K / 32K / 1024K words	256	1	2w		
AHCPU520-EN	2304	128K steps (512KB)	64K / 64K / 2048K words	512	3	2w		
AHCPU530-EN	4352	256K steps (1MB)	64K / 64K / 4096K words	1024	7	2w		

Note: Data Register B is for the use of function blocks

Main Backplanes

Model	Slot	Power consumption (Internal)	Specifications	Certificates
AHBP04M1-5A	4	0.01w	▪ Supports CPU modules ▪ Supports remote IO communication modules (RTU) ▪ Built-in communication port for extension backplanes ▪ Slot spaces are not occupied by Power / CPU / RTU modules	CE cULus
AHBP06M1-5A	6	0.01w		
AHBP08M1-5A	8	0.01w		
AHBP12M1-5A	12	0.01w		

Extension Backplanes

Model	Slot	Power consumption (Internal)	Specifications	Certificates
AHBP06E1-5A	6	1.41w	▪ For main backplane extension ▪ Built-in communication port for extension backplanes ▪ Slot spaces are not occupied by power modules	CE cULus
AHBP08E1-5A	8	1.41w		

Ordering Information

Power Supply Modules

Model	Power input	Output	Specifications	Certificates
AHPS05-5A	100~240VAC 50 / 60Hz	60W	▪ Power supply for the modules on the racks ▪ LED power indicator ▪ Provides external DC power abnormal signal detection input and triggered interrupt function	CE cULus
AHPS15-5A	24VDC	36W		

Digital I/O Modules (Input)

Model	Points	Signals	Terminal block type	Power consumption (Internal / External)	Accessories (optional)	Specifications	Certificates
AH16AM10N-5A	16	24VDC 5mA	JIS removable terminal block	0.1w / 1.9w	-	▪ PNP / NPN mixed mode design ▪ Supports hot-swapping function ▪ Individual LED status indicator	CE cULus
AH16AM30N-5A	16	120~240VAC 4.5~9mA	JIS removable terminal block	0.1w / -	-		
AH32AM10N-5A	32	24VDC 5mA	EU removable terminal block	0.2w / 3.8w	-		
AH32AM10N-5B	32	24VDC 5mA	DB37	0.2w / 3.8w	DVPACAB7C10 x 1 DVPAETB-ID32B x 1		
AH32AM10N-5C	32	24VDC 5mA	Latched connector	0.2w / 3.8w	DVPACAB7A10 x 1 DVPAETB-ID32A x 1		
AH64AM10N-5C	64	24VDC 3.2mA	Latched connector	0.2w / 4.9w	DVPACAB7A10 x 2 DVPAETB-ID32A x 2	▪ PNP / NPN mixed mode design ▪ Supports hot-swapping function ▪ Individual LED status indicator (32 points)	
New AH16AR10N-5A	16	24VDC 5mA	JIS removable terminal block	0.5w / 1.9w	-	▪ PNP / NPN mixed mode design ▪ Supports hot-swapping function ▪ Individual LED status indicator ▪ Supports I/O interrupts ▪ Supports rising / falling-edge trigger modes ▪ Supports signal time-delay setting for 0.1 / 0.5 / 3 / 15 / 20 ms	

Digital I/O Modules (Output)

Model	Points	Signals	Terminal block type	Power consumption (Internal / External)	Accessories (optional)	Specifications	Certificates
AH16AN01R-5A	16	Relay 240VAC / 24VDC 2A	JIS removable terminal block	2.1w / -	-	▪ Supports hot-swapping function ▪ Individual LED status indicator ▪ Supports keep-last-value function when CPU shuts down	CE cULus
AH16AN01T-5A	16	NPN (Sink) 12~24VDC 0.5A	JIS removable terminal block	0.2w / 0.4w	-		
AH16AN01P-5A	16	PNP (Source) 12~24VDC 0.5A	JIS removable terminal block	0.2w / 0.4w	-		
AH16AN01S-5A	16	TRIAC 120 / 240VAC 0.5A	JIS removable terminal block	0.6w / -	-		
AH32AN02T-5A	32	NPN (Sink) 12~24VDC 0.1A	EU removable terminal block	0.4w / 0.8w	-		
AH32AN02P-5A	32	PNP (Source) 12~24VDC 0.1A	EU removable terminal block	0.4w / 0.8w	-		
AH32AN02T-5B	32	NPN (Sink) 12~24VDC 0.1A	DB37	0.4w / 0.8w	(DVPACAB7C10 x 1 DVPAETB-OR32A x 1) or (DVPACAB7C10 x 1 DVPAETB-OT32B x 1)		
AH32AN02P-5B	32	PNP (Source) 12~24VDC 0.1A	DB37	0.4w / 0.8w	(DVPACAB7C10 x 1 DVPAETB-OR32B x 1) or (DVPACAB7C10 x 1 DVPAETB-OT32B x 1)		
AH32AN02T-5C	32	NPN (Sink) 12~24VDC 0.1A	Latched connector	0.4w / 0.8w	(DVPACAB7B10 x 1 DVPAETB-OR16A x 2) or (DVPACAB7A10 x 1 DVPAETB-OT32A x 1)		
AH32AN02P-5C	32	PNP (Source) 12~24VDC 0.1A	Latched connector	0.4w / 0.8w	(DVPACAB7B10 x 1 DVPAETB-OR16B x 2) or (DVPACAB7A10 x 1 DVPAETB-OT32A x 1)		
AH64AN02T-5C	64	NPN (Sink) 12~24VDC 0.1A	Latched connector	0.6w / 1.5w	(DVPACAB7B10 x 2 DVPAETB-OR16A x 4) or (DVPACAB7A10 x 2 DVPAETB-OT32A x 2)	▪ Supports hot-swapping function ▪ Individual LED status indicator (32 points) ▪ Supports keep-last-value function when CPU shuts down	
AH64AN02P-5C	64	PNP (Source) 12~24VDC 0.1A	Latched connector	0.6w / 1.5w	(DVPACAB7B10 x 2 DVPAETB-OR16B x 4) or (DVPACAB7A10 x 2 DVPAETB-OT32A x 2)		

Digital I/O Modules (Mixed)

Model	Inputs	Outputs	Input signals	Output signals	Terminal block type	Power consumption (Internal / External)	Specifications	Certificates
AH16AP11R-5A	8	8	24VDC 5mA	Relay 240VAC / 24VDC 2A	JIS removable terminal block	1.1w / -	▪ PNP / NPN mixed mode design ▪ Supports hot-swapping function ▪ Individual LED status indicator ▪ Supports keep-last-value function when CPU shuts down	CE cULus
AH16AP11T-5A	8	8	24VDC 5mA	NPN (Sink) 12~24VDC 0.5A	JIS removable terminal block	0.2w / 0.2w		
AH16AP11P-5A	8	8	24VDC 5mA	PNP (Source) 12~24VDC 0.5A	JIS removable terminal block	0.2w / 0.2w		

Ordering Information

Analog I/O Modules (Input)

Model	Channels	Signals	Terminal block type	Power consumption (Internal / External)	Specifications	Certificates
AH04AD-5A	4	0 / 1V~5V, ±5V, 0V~10V, ±10V 0 / 4mA~20mA ±20mA	JIS removable terminal block	0.35w / 1w	▪ Hardware resolution: 16-bit ▪ Conversion time: 150 μs / channel ▪ Base error (ambient temp.): Voltage mode ±0.1%, Current mode ±0.1% ▪ Base error (full temp. range): Voltage mode ±0.45%, Current mode ±0.2% ▪ Linearity error (ambient temp.): Voltage mode ±0.07%, Current mode ±0.05% ▪ Linearity error (full temp. range): Voltage mode ±0.12%, Current mode ±0.23% ▪ Supports hot-swapping function ▪ Isolated signal design ▪ Diagnose function ▪ Module status LED indicator ▪ Supports interrupt function	CE cULus
New AH08AD-5A	8	0 / 1V~5V, ±5V, 0V~10V, ±10V 0 / 4mA~20mA, ±20mA	EU removable terminal block	1.5w / -		
AH08AD-5B	8	0 / 1V~5V, ±5V, 0V~10V, ±10V	JIS removable terminal block	1.9w / -		
AH08AD-5C	8	0 / 4mA~20mA, ±20mA	JIS removable terminal block	1.6w / -		

Analog I/O Modules (Output)

Model	Channels	Signals	Terminal block type	Power consumption (Internal / External)	Specifications	Certificates
AH04DA-5A	4	0 / 1V~5V, ±5V, 0V~10V, ±10V 0 / 4mA~20mA	JIS removable terminal block	0.34w / 2.6w	▪ Hardware resolution: 16-bit ▪ Conversion time: 150 μs / channel ▪ Base error (ambient temp.): Voltage mode ±0.02%, Current mode ±0.06% ▪ Base error (full temp. range): Voltage mode ±0.04%, Current mode ±0.07% ▪ Linearity error (ambient temp.): Voltage mode ±0.004%, Current mode ±0.01% ▪ Linearity error (full temp. range): Voltage mode ±0.004%, Current mode ±0.01% ▪ Supports hot-swapping function ▪ Isolated signal design ▪ Diagnose function ▪ Module status LED indicator ▪ Supports interrupt function ▪ Supports keep-last-value function when CPU shuts down	CE cULus
New AH08DA-5A	8	0 / 1V~5V, ±5V, 0V~10V, ±10V 0 / 4mA~20mA	EU removable terminal block	1w / 5w		
AH08DA-5B	8	0 / 1V~5V, ±5V, 0V~10V, ±10V	JIS removable terminal block	0.25w / 2.2w		
AH08DA-5C	8	0 / 4mA~20mA	JIS removable terminal block	0.25w / 3.7w		

Analog I/O Modules (Mixed)

Model	Channels	Signals	Terminal block type	Power consumption (Internal / External)	Specifications	Certificates
AH06XA-5A	Inputs: 4 Outputs: 2	**Input:** 0 / 1V~5V, ±5V, 0V~10V, ±10V 0 / 4mA~20mA, ±20mA **Output:** 0 / 1V~5V, ±5V, 0V~10V, ±10V 0 / 4mA~20mA	JIS removable terminal block	0.34w / 1.4w	▪ Hardware resolution: 16-bit ▪ Conversion time: 150μs / channel ▪ Input accuracy: same as AH04AD-5A ▪ Output accuracy: same as AH04DA-5A ▪ Supports hot-swapping function ▪ Isolated signal design ▪ Diagnose function ▪ Module status LED indicator ▪ Supports interrupt function ▪ Supports keep-last-value function when CPU shuts down	CE cULus

Temperature Measurement Modules

Model	Channels	Signals	Resolution	Conversion time	Terminal block type	Power consumption (Internal / External)	Specifications	Certificates
AH04PT-5A	4	(2 / 3 / 4-wire RTD input) PT100,PT1000, Ni100,Ni1000, 0Ω~300Ω	0.1°C / 0.1°F 0.1% (0Ω~300Ω)	2 / 4-wire: 150ms / channel 3-wire: 300ms / channel	JIS removable terminal block	2w / -	▪ Effective resolution: 16-bit ▪ Accuracy: ±0.6% (Full Scale) ▪ Supports hot-swapping function ▪ Signal isolated design ▪ Diagnose function ▪ PID function ▪ Module status LED indicator ▪ Supports interrupt function ▪ Supports disconnection detection function ▪ Fully isolated channel design (AH08PTG-5A)	CE cULus
AH04TC-5A	4	Thermocouple input J,K,R,S,T,E,N, ±150mV	0.1°C / 0.1°F	200ms / channel	JIS removable terminal block	1.5w / -		
AH08TC-5A	8	Thermocouple input J,K,R,S,T,E,N, ±150mV	0.1°C / 0.1°F	200ms / channel	JIS removable terminal block	1.5w / -		
New AH08PTG-5A	8	(2 / 3 / 4-wire RTD input) PT100, PT1000, Ni100, Ni1000, 0Ω~300Ω	0.1°C / 0.1°F 0.1% (0Ω~300Ω)	2 / 4-wire: 20ms (fast)~100ms (normal) / channel 3-wire:200ms / channel	EU removable terminal block	0.7w / 4w		

Network Modules

Model	Power consumption (Internal / External)	Specifications			Certificates
AH10EN-5A	1.6w / -	▪ Ethernet communication module (Master / Salve) ▪ 100Mbps communication port x 2 (switch function available) ▪ Ether Link function ▪ MODBUS TCP function	▪ Automatic data exchange function ▪ NTP network time correction function ▪ SNMP / E-mail / IP Filter function	▪ Supports hot-swapping function ▪ Diagnose function ▪ Module status LED indicator	CE cULus
AH10SCM-5A	1.2w / -	▪ Serial communication module (Master / Slave) ▪ Full isolation design in power & signal circuits ▪ Built-in RS-422 / 485 with communication port x 2 (multiple modes, 460.8Kbps) ▪ Supports PLC Link function	▪ Supports user defined communication format (UD Link) ▪ MODBUS RTU / ASCII function available ▪ Supports automatic data exchange function ▪ Supports BACnet Slave function		
AH10DNET-5A	0.9w / 0.72w	▪ DeviceNet communication module (Master / Slave) ▪ Supports max. speed of 1Mbps ▪ Switchable between master and slave modes	▪ Connecting up to 63 slave stations in master mode ▪ RIO capacity in master mode: 490 words IN / OUT		
New AH10PFBM-5A	2w / -	▪ PROFIBUS-DP master module ▪ Supports DPV0 / DPV1 ▪ Max. speed: 12Mbps	▪ Connects up to 124 slaves ▪ Configurable I/O capacity: 2880 words for input / 2880 words for output		
AH10PFBS-5A	1w / -	▪ PROFIBUS-DP slave module ▪ Supports DPV0 / DPV1 ▪ Max. speed: 12Mbps	▪ Configurable I/O capacity: 100 words for input / 100 words for output		
New AH10COPM-5A	1w / -	▪ CANopen module (Master / Slave) ▪ Max. speed: 1Mbps ▪ Connects up to 100 slaves in master mode			

Remote I/O Modules

Model	Power consumption (Internal / External)	Specifications		Certificates
AHRTU-DNET-5A	0.75w / 0.72w	▪ DeviceNet remote I/O module ▪ Supports max. speed of 1Mbps ▪ Supports AH500 DIO modules, AIO modules, temperature measurement modules, and the serial communication module AH10SCM	▪ Installing on the main backplane required ▪ Supports up to 7 extension racks ▪ Diagnose function ▪ Module status LED indicator	CE cULus
New AHRTU-PFBS-5A	1.9w / -	▪ PROFIBUS-DP remote I/O module ▪ Max. speed: 12Mbps ▪ Supports AH500 digital I/O modules, analog I/O modules, temperature measurement modules ▪ Installs on main backplane	▪ Supports 7 extension backplanes ▪ Diagnose function ▪ Module status LED indicator ▪ Configurable I/O capacity: 122 words for input / 122 words for output	

Ordering Information

Motion Control Modules

Model	Terminal block type	Power consumption (Internal)	Accessories (optional)	Specifications		Certificates
AH02HC-5A	EU removable terminal block	2.4w	-	▪ 2 high speed counter channels	▪ 200K Hz input ▪ UD / PD / AB / 4AB modes ▪ Supports interrupt function ▪ Supports hot-swapping function ▪ Diagnose function ▪ Module status LED indicator	CE cULus
AH04HC-5A	HDC	2.4w	DVPACAB7D10 DVPAETB-IO16C	▪ 4 high speed counter channels		
AH05PM-5A	EU removable terminal block	2.7w	-	▪ 2-axis pulse train motion control module ▪ Supports 1MHz output ▪ Supports 2-axis linear interpolation, 2-axis arc interpolation	▪ CPU execution speed:	

Accessories

Products	Descriptions	Models	Specifications	Applicable Modules	Certificates
Cables	Extension cable for connecting extension backplane	AHACAB06-5A	0.6m	AHBP04M1-5A / AHBP06M1-5A / AHBP08M1-5A AHBP12M1-5A / AHBP06E1-5A / AHBP08E1-5A	c UL us
		AHACAB10-5A	1.0m		
		AHACAB15-5A	1.5m		
		AHACAB30-5A	3.0m		
		AHACAB50-5A	5.0m		
		AHACABA0-5A	10.0m		
		AHACABA5-5A	15.0m		
		AHACABB0-5A	20.0m		
		AHACABC0-5A	30.0m		
		AHACABD0-5A	40.0m		
		AHACABE0-5A	50.0m		
		AHACABF0-5A	60.0m		
		AHACABG0-5A	70.0m		
		AHACABH0-5A	80.0m		
		AHACABI0-5A	90.0m		
		AHACABJ0-5A	100.0m		
	I/O extension cable for connecting external terminal modules	DVPACAB7A10	1.0m / Latched connector	AH32AM10N-5C / AH32AN02T-5C / AH32AN02P-5C / AH64AM10N-5C / AH64AN02T-5C / AH64AN02P-5C	
		DVPACAB7B10	1.0m / Latched connector	AH32AN02T-5C / AH32AN02P-5C / AH64AN02T-5C / AH64AN02P-5C	
		DVPACAB7C10	1.0m / DB37	AH32AM10N-5B / AH32AN02T-5B / AH32AN02P-5B	
		DVPACAB7D10	1.0m / HDC	AH04HC-5A / AH20MC-5A	
		DVPACAB7E10	1.0m / HDC	AH10PM-5A / AH15PM-5A	
	CANopen / DeviceNet cables	TAP-CB01	305.0m (Thick / Trunk Cable)	AH10COPM-5A / AH10DNET-5A / AHRTU-DNET-5A / TAP-CN01 / TAP-CN02 / TAP-CN03	
		TAP-CB02	305.0m (Thin / Drop Cable)		
	CANopen / DeviceNet / DMCNET cables	TAP-CB03	0.3m / RJ45	AH20MC-5A / TAP-CN03	
		TAP-CB05	0.5m / RJ45		
		TAP-CB10	1.0m / RJ45		
		TAP-CB20	2.0m / RJ45		
		TAP-CB30	3.0m / RJ45		
		TAP-CB100	10.0m / RJ45		
	PROFIBUS cables	TAP-CBDP	100.0m	AH10PFBM-5A / AH10PFBS-5A / AHRTU-PFBS-5A	
External terminal modules	For digital input modules	DVPAETB-ID32A	Latched connector	AH32AM10N-5C / AH64AM10N-5C	CE
		DVPAETB-ID32B	DB37	AH32AM10N-5B	
	For digital output modules	DVPAETB-OR16A	16 points relay output (240VAC / 24VDC, 2A) Latched connector	AH32AN02T-5C / AH64AN02T-5C	
		DVPAETB-OR16B	16 points relay output (240VAC / 24VDC, 2A) Latched connector	AH32AN02P-5C / AH64AN02P-5C	
		DVPAETB-OR32A	32 points relay output (240VAC / 24VDC, 2A) DB37	AH32AN02T-5B	
		DVPAETB-OR32B	32 points relay output (240VAC / 24VDC, 2A) DB37	AH32AN02P-5B	
		DVPAETB-OT32A	Transistor output latched connector	AH32AN02T-5C / AH32AN02P-5C / AH64AN02T-5C / AH64AN02P-5C	
		DVPAETB-OT32B	Transistor output DB37	AH32AN02T-5B / AH32AN02P-5B	
	For motion control modules	DVPAETB-IO16C	HDC	AH04HC-5A / AH20MC-5A	
		DVPAETB-IO24C	HDC	AH10PM-5A	
		DVPAETB-IO34C	HDC	AH15PM-5A	
Terminal resistors	DMCNET terminal resistors (RJ45)	ASD-TR-DM0008			
	CANopen / DeviceNet terminal resistors (RJ45)	TAP-TR01			
Distribution box	CANopen / DeviceNet distribution Box	TAP-CP01	Power distribution box		CE
		TAP-CN01	1 for 2		
		TAP-CN02	1 for 4		
		TAP-CN03	1 for 4 (RJ45)		
Dummy modules	To protect empty slots	AHASP01-5A			
DIN rail	Used on DIN rail for rack installation	AHADINADP1-5A			
New Fiber optics modules for backplanes	Used for backplane extension via fiber optics (installs at the backplane's lower extension port)	AHAADP01EF-5A	▪ Optical fiber connector: SC ▪ Supported optical fiber types: multi-modes, 62.5/125 μm or 50/125 μm ▪ Optical fiber max. length: 2km	AHBP04M1-5A / AHBP06M1-5A / AHBP08M1-5A / AHBP12M1-5A / AHBP06E1-5A / AHBP08E1-5A	CE
	Used for backplane extension via fiber optics (installs at the backplane's upper extension port)	AHAADP02EF-5A		AHBP06E1-5A / AHBP08E1-5A	c UL us
Memory card	SD card: 1 GB	FMC-SD001G	▪ Capacity: 1 GB ▪ Speed (Read / Write) : Max. 18/15 MB/s	▪ Overwrite: 10,000 times ▪ Operation temperature: -40~85°C	

ضمیمه ب – نمونه مسائل

تعداد زیادی نمونه سئوال مربوط به آزمایشگاه PLC دانشگاه خواجه نصیرالدین طوسی و همچنین منوآل های شرکت دلتا و پروژه‌های عملی اجرا شده در صفحه در نظر گرفته شده برای کتاب به آدرس saba.kntu.ac.ir/eecd/azangoo/research/ISPSoft.html آماده شده است که قابل استفاده است. همچنین شما می‌توانید پروژه‌ها و یا سئوالات و یا راه حل‌های پیشنهادی خود برای مسائل مختلف را به آدرس ایمیل m.azangoo@gmail.com ارسال نمایید تا علاوه بر تکمیل مجموعه در دسترس ، محتویات ارسالی با نام خودتان در سایت قرار گیرد.

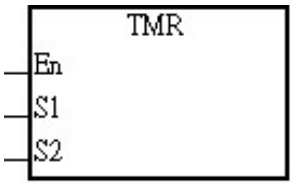
ضمیمه ج – مروری بر دستورالعمل های متداول

۱– نحوه ی به کارگیری دستورالعمل مناسب

ابتدا با استفاده از لیست بخش بعد، دستورالعمل مطلوب خود را پیدا کنید، به عنوان نمونه دربخش مربوط به زمانسنج‌ها مشخص شده است که TMR یک زمانسنج ۱۶ بیتی است. سپس در منوی Help نرم افزار ISPSoft بر روی PLC Instruction and Special Registers Reference کلیک نمایید. در این بخش دستورالعمل TMR را یافته و بر روی لینک آن کلیک نمایید. صفحه‌ای مشابه صفحه زیر ظاهر خواهد شد و اطلاعاتی را در اختیار شما قرار می‌دهد.

API 96 TMR 16-bit timer

S_1 : Timer number
S_2 : Set value



Input/Output	Operands Device Range	Data Type
S_1	T	WORD
S_2	Decimal, D	WORD

Explanations:

When TMR instruction is executed, the specific coil of timer is ON and timer will start to count. When the setting value of timer is attained (counting value >= setting value), the contact will be as following:

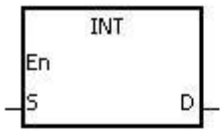
NO(Normally Open) contact	Continuity
NC(Normally Closed) contact	Non-continuity

از توضیحات می‌توان متوجه شد که این زمانسنج دارای یک ورودی نوع T است که پس از فعال شدن بلوک شروع به شمردن زمان می‌کند و تا زمان رسیدن به مقدار ورودی نوع D

به این کار ادامه می دهد و در نهایت نیز وضعیت کانتکت زمانسنج بر اساس NC یا NO بودن آن مطابق جدول انتهایی تعیین می‌شود.

به عنوان نمونه ای دیگر، دستورالعمل INT مقدار اعشاری ممیز شناور را مطابق راهنمای زیر با حذف اعشار به Integer (عدد صحیح) تبدیل می‌کند.

API 129 INT Float point to Integer



S : Source device

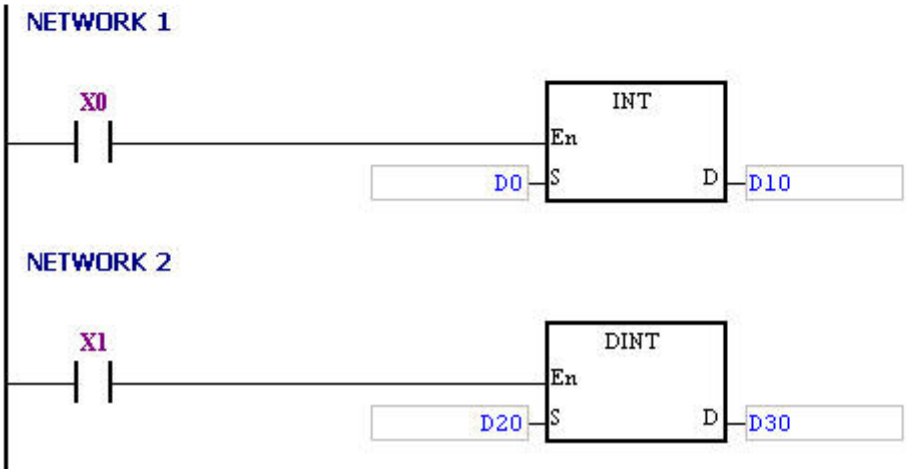
D : Destination device to store the result

Input/Output	Operands Device Range	Data Type
S	D	WORD
D	D	WORD

Explanations:

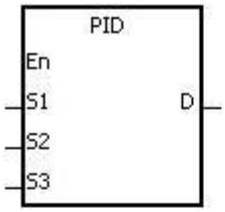
The binary floating point value of the register assigned by **S** is converted to BIN integer and stored in the register assigned by **D**. The decimal of BIN integer is left out.

Sample:



همچنین دستورالعمل PID مطابق توضیحات زیر، مقدار پارامتر فیزیکی مورد نظر ما را با تنظیم ضرایب کنترل کننده یعنی P و I و D به مقدار مطلوب هدایت می‌کند.

API 88 PID PID Calculation



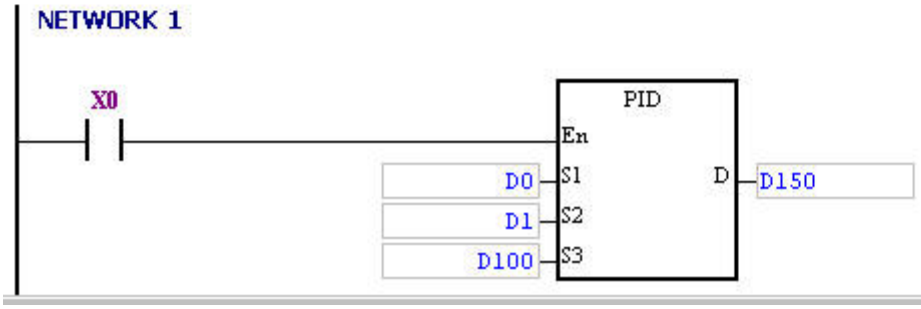
S_1 : Target value (SV)
S_2 : Present value (PV)
S_3 : Parameter (for 16-bit instruction, uses 20 continuous devices, for 32-bit instruction, uses 21 continuous devices)
D : Output value (MV)

Input/Output	Operands Device Range	Data Type
S_1	D	WORD
S_2	D	WORD
S_3	D	WORD
D	D	WORD

Explanations:

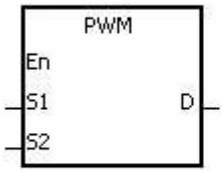
This instruction is specifically for PID control. PID operation will be executed by the scan only when the sampling time is reached. PID refers to "proportion, integration and differential". PID control is widely applied to many machines, pneumatic and electronic equipments.

Sample:



یا دستورالعمل پرکاربرد PWM که می‌توانیم مطابق الگوی زیر مدت زمان دوره[۱] (معکوس فرکانس[۲]) و پالس (مدت زمان یک بودن سیگنال) را تنظیم و از آن در برنامه کنترلی خود استفاده نماییم.

API 58 PWM Pulse Width Modulation



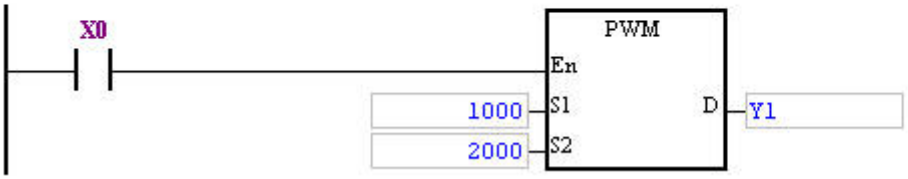
S_1 : Pulse output width
S_2 : Pulse output period
D : Pulse output device (only be specified as Y1)

Input/Output	Operands Device Range	Data Type
S_1	Decimal, 16#, KnX, KnY, KnM, KnS, T, C, D, @E, @F	WORD
S_2	Decimal, 16#, KnX, KnY, KnM, KnS, T, C, D, @E, @F	WORD
D	Y	BIT

Explanations:

1. PWM instruction assigns the pulse output width S_1 and pulse output cycle S_2 from output device D.
2. S_1 and S_2 can be changed when PWM instruction is being executed.

Sample:



[۱] Period

[۲] Frequency

۲-لیست دستورالعمل‌ها

شما با استفاده از لیست پیش رو می‌توانید دستورالعمل‌های مورد نیاز خود را به سادگی در بخش‌های مختلف پیدا کرده و سپس جهت دیدن توضیحات دقیقتر و مشاهده نحوه عملکرد تابع بلوکی آن به راهنمای نرم افزار ISPSoft مراجعه نمایید.

Basic Instruction:

General:

LD	Load A contact
LDI	Load B contact
AND	Series connection- A contact
ANI	Series connection- B contact
OR	Parallel connection- A contact
ORI	Parallel connection- B contact
ANB	Series connection (Multiple Circuits)
ORB	Parallel connection (Multiple circuits)
MPS	Store the current result of the internal PLC operations
MRD	Reads the current result of the internal PLC operations
MPP	Pops (recalls and removes) the currently stored result

Output:

OUT	Output coil
SET	Latch（ON）
RST	Clear the contacts or the registers

Timers, Counters:

96	TMR	16-bit timer
97	CNT	16-bit counter
97	DCNT	32-bit counter

Main control:

MC		Master control Start
MCR		Master control Reset

Rising-edge/falling-edge detection:

90	LDP	Rising-edge detection operation
91	LDF	Falling-edge detection operation
92	ANDP	Rising-edge series connection
93	ANDF	Falling-edge series connection
94	ORP	Rising-edge parallel connection
95	ORF	Falling-edge parallel connection

Rising-edge/falling-edge output:

89	PLS	Rising-edge output
99	PLF	Falling-edge output

End:

END		Program end

Other:

NOP		No operation
INV		Inverting operation
P		Pointer
I		Interrupt program marker

Step ladder:

STL		Step transition ladder start command
RET		Step transition ladder return command

Application Instruction:

Loop Control:

00	CJ	Conditional jump
01	CALL	Call subroutine
02	SRET	Subroutine return
03	IRET	Interrupt return
04	EI	Enable interrupts
05	DI	Disable interrupts

06	FEND	First end
07	WDT	Watchdog timer refresh
08	FOR	Start of FOR-NEXT loop
09	NEXT	End of FOR-NEXT loop

Transmission Comparison:

10	CMP	Compare
11	ZCP	Zone compare
12	MOV	Data Move
13	SMOV	Shift move
14	CML	Compliment
15	BMOV	Block move
16	FMOV	Fill move
17	XCH	Data exchange
18	BCD	Convert BIN data into BCD
19	BIN	Convert BCD data into BIN

Four Fundamental Operations of Arithmetic:

20	ADD	Perform the addition of BIN data
21	SUB	Perform the subtraction of BIN data
22	MUL	Perform the multiplication of BIN data
23	DIV	Perform the division of BIN data
24	INC	Perform the addition of 1
25	DEC	Perform the subtraction of 1
26	WAND / DAND	Perform the logical product (AND) operation
27	WOR / DOR	Perform the logical sum (OR) operation
28	WXOR / DXOR	Perform the exclusive logical add (XOR) operation
29	NEG	Negation

Rotation and Displacement:

30	ROR	Rotate to the right
31	ROL	Rotate to the left
32	RCR	Rotate to the right with the carry flag attached
33	RCL	Rotate to the left with the carry flag attached
34	SFTR	Shift the data of device specified to the right
35	SFTL	Shift the data of device specified to the left
36	WSFR	Shift the register to the right

37	WSFL	Shift the register to the left
38	SFWR	Shift register write
39	SFRD	Shift register read

Data Operation:

40	ZRST	Resets a range of device specified
41	DECO	8 → 256 bits decoder
42	ENCO	256 → 8 bits encoder
43	SUM	Sum of ON bits
44	BON	Check specified bit status
45	MEAN	Mean value
46	ANS	Alarm device output
47	ANR	Alarm device reset
48	SQR	Square root of BIN
49	FLT	Convert BIN integer to binary floating point

High Speed Processing:

50	REF	I/O refresh
51	REFF	Refresh and adjust the response time of input filter
52	MTR	Input matrix
53	DHSCS	High speed counter comparison SET
54	DHSCR	High speed counter comparison RESET
55	DHSZ	Zone comparison (High-speed counter)
56	SPD	Speed detection
57	PLSY	Pulse output
58	PWM	Pulse width modulation output
59	PLSR	Pulse wave output with acceleration/deceleration speed

Convenience Instruction:

60	IST	Manual/Auto control
61	SER	Search a data stack
62	ABSD	Absolute drum sequencer
63	INCD	Increment drum sequencer
64	TTMR	Teaching timer
65	STMR	Special timer
66	ALT	On/Off alternate command
67	RAMP	Ramp signal

69	SORT	Data sort

External I/O Display:

70	TKY	10-key keypad input
71	HKY	16-key keypad input
72	DSW	Digital Switch input
73	SEGD	Decode the 7-step display panel
74	SEGL	7-step display scan output
75	ARWS	Arrow keypad input
76	ASC	ASCII code conversion
77	PR	Output ASCII code
78	FROM	Read special module CR data
79	TO	Special module CR data write in

Serial I/O:

80	RS	Serial data communication
81	PRUN	Octal number system transmission
82	ASCI	Convert HEX to ASCII
83	HEX	Convert ASCII to HEX
84	CCD	Check code
85	VRRD	Potentiometer read
86	VRSC	Potentiometer scale
87	ABS	Absolute value
88	PID	PID calculation

Communication Instruction

100	MODRD	MODBUS data Read
101	MODWR	MODBUS data write in
102	FWD	VFD-A series drive forward instruction
103	REV	VFD-A series drive reverse instruction
104	STOP	VFD-A series drive stop instruction
105	RDST	VFD-A series drive status read
106	RSTEF	VFD-A series drive abnormal reset
107	LRC	LRC error check
108	CRC	CRC error check
150	MODRW	MODBUS data read/write in
206	ASDRW	ASDA servo drive R/W

Floating Operation:

110	DECMP	Binary floating point comparison
111	DEZCP	Binary floating point zone comparison
112	DMOVR	Floating point data Move
116	DRAD	Degree → Radian
117	DDEG	Radian → Degree
118	DEBCD	Convert binary floating point to decimal floating point
119	DEBIN	Convert decimal floating point to binary floating point
120	DEADD	Binary floating point addition
121	DESUB	Binary floating point subtraction
122	DEMUL	Binary floating point multiplication
123	DEDIV	Binary floating point division
124	DEXP	Perform exponent operation of binary floating point
125	DLN	Perform natural logarithm operation of binary floating point
126	DLOG	Perform logarithm operation of binary floating point
127	DESQR	Square root of binary floating point
128	DPOW	Perform power operation of binary floating point
129	INT	Convert binary floating point to BIN integer
130	DSIN	Sine operation of binary floating point
131	DCOS	Cosine operation of binary floating point
132	DTAN	Tangent operation of binary floating point
133	DASIN	Arcsine operation of binary floating point
134	DACOS	Arccosine operation of binary floating point
135	DATAN	Arctangent operation of binary floating point
136	DSINH	Hyperbolic sine operation of binary floating point
137	DCOSH	Hyperbolic cosine operation of binary floating point
138	DTANH	Hyperbolic tangent operation of binary floating point
172	DADDR	Floating Point Number Addition
173	DSUBR	Floating Point Number Subtraction
174	DMULR	Floating Point Number Multiplication
175	DDIVR	Floating Point Number Division

Additional Instruction:

143	DELAY	Command delay
144	GPWM	General pulse width modulation output
145	FTC	Fuzzy temperature control
146	CVM	Valve Control

147	SWAP	Swap high/low byte
148	MEMR	Data backup MEMORY read
149	MEMW	Data backup MEMORY write in
151	PWD	Input pulse width detection
152	RTMU	Start to measure the execution tine of I interrupt
153	RTMD	End to measure the execution tine of I interrupt
154	RAND	Random value
168	MVM	Mask and Combine Designated Bits
196	HST	High speed timer
202	SCAL	Calculation of Proportional Value
203	SCLP	Calculation of Parameter Proportional Value
176	MMOV	Magnifying Transfer with Sign Extension
177	GPS	GPS data receiving
178	DSPA	Solar Cell Positioning
179	WSUM	Sum of multiple devices
196	HST	High Speed Timer
202	SCAL	Proportional calculation
203	SCLP	Parameter proportional calculation
205	CMPT	Compare table
207	CSFO	Catch speed and proportional output

Positioning Control:

155	DABSR	ABS current value read
156	ZRN	Zero point return
157	PLSV	Variable speed pulse output
158	DRVI	Drive to increment
159	DRVA	Drive to absolute
191	DPPMR	2-Axis Relative Position Arc Interpolation
192	DPPMA	2-Axis Absolute Position Arc Interpolation
193	DCIMR	2-Axis Relative Point-to-Point Movement
194	DCIMA	2-Axis Absolute Point-to-Point Movement
195	DPTPO	Single Axis Planned Table Pulse Output
197	DCLLM	Close loop position control
198	DVSPO	Variable speed pulse output
199	DICF	Immediately change frequency

Perpetual Calendar:

160	TCMP	Time compare

161	TZCP	Time zone compare
162	TADD	Time addition
163	TSUB	Time subtraction
166	TRD	Time data read
167	TWR	Time data write in
169	HOUR	Hour meter

Gray Code:

170	GRY	Convert BIN to Gray code
171	GBIN	Convert Gray code to BIN

Matrix Handing:

180	MAND	Matrix AND
181	MOR	Matrix OR
182	MXOR	Matrix XOR
183	MXNR	Matrix NOR
184	MINV	Matrix inverse
185	MCMP	Matrix compare
186	MBRD	Matrix bit read
187	MBWR	Matrix bit write
188	MBS	Matrix bit shift
189	MBR	Matrix bit rotate
190	MBC	Matrix bit state count

Contact Type Logic Operation:

215	LD&	Comparison contact is ON when S1 & S2 is true
216	LD|	Comparison contact is ON when S1 | S2 is true
217	LD^	Comparison contact is ON when S1 ^ S2 is true
218	AND&	Comparison contact is ON when S1 & S2 is true
219	AND|	Comparison contact is ON when S1 | S2 is true
220	AND^	Comparison contact is ON when S1 ^ S2 is true
221	OR&	Comparison contact is ON when S1 & S2 is true
222	OR|	Comparison contact is ON when S1 | S2 is true
223	OR^	Comparison contact is ON when S1 ^ S2 is true

Contact Type Compare Instruction:

224	LD=	Comparison contact is ON when S1 = S2 is true

225	LD>	Comparison contact is ON when S1 > S2 is true
226	LD<	Comparison contact is ON when S1 < S2 is true
228	LD<>	Comparison contact is ON when S1 ≠ S2 is true
229	LD<=	Comparison contact is ON when S1 ≦ S2 is true
230	LD>=	Comparison contact is ON when S1 ≧ S2 is true
232	AND=	Comparison contact is ON when S1 = S2 is true
233	AND>	Comparison contact is ON when S1 > S2 is true
234	AND<	Comparison contact is ON when S1 < S2 is true
236	AND<>	Comparison contact is ON when S1 ≠ S2 is true
237	AND<=	Comparison contact is ON when S1 ≦ S2 is true
238	AND>=	Comparison contact is ON when S1 ≧ S2 is true
240	OR=	Comparison contact is ON when S1 = S2 is true
241	OR>	Comparison contact is ON when S1 > S2 is true
242	OR<	Comparison contact is ON when S1 < S2 is true
244	OR<>	Comparison contact is ON when S1 ≠ S2 is true
245	OR<=	Comparison contact is ON when S1 ≦ S2 is true
246	OR>=	Comparison contact is ON when S1 ≧ S2 is true

Specific Bit Control

266	BOUT	Output Specified Bit of a Word
267	BSET	Set ON Specified Bit of a Word
268	BRST	Reset Specified Bit of a Word
269	BLD	Load NO Contact by Specified Bit
270	BLDI	Load NC Contact by Specified Bit
271	BAND	Connect NO Contact in Series by Specified Bit
272	BANI	Connect NC Contact in Series by Specified Bit
273	BOR	Connect NO Contact in Parallel by Specified Bit
274	BORI	Connect NC Contact in Parallel by Specified Bit

Floating Point Contact Type Comparison

275	FLD=	Floating Point Contact Type Comparison
276	FLD>	Floating Point Contact Type Comparison
277	FLD<	Floating Point Contact Type Comparison
278	FLD<>	Floating Point Contact Type Comparison
279	FLD<=	Floating Point Contact Type Comparison
280	FLD>=	Floating Point Contact Type Comparison
281	FAND=	Floating Point Serial Type Comparison

282	FAND>	Floating Point Serial Type Comparison
283	FAND<	Floating Point Serial Type Comparison
284	FAND<>	Floating Point Serial Type Comparison
285	FAND<=	Floating Point Serial Type Comparison
286	FAND>=	Floating Point Serial Type Comparison
287	FOR=	Floating Point Parallel Type Comparison
288	FOR>	Floating Point Parallel Type Comparison
289	FOR<	Floating Point Parallel Type Comparison
290	FOR<>	Floating Point Parallel Type Comparison
291	FOR<=	Floating Point Parallel Type Comparison
292	FOR>=	Floating Point Parallel Type Comparison

ضمیمه د – مروری بر حافظه های خاص PLCهای دلتا

یکی از نقاط قوت PLCهای دلتا استفاده از حافظه‌ها خاص برای کاربردها خاص است. در PLCهای سری DVP دلتا حافظه‌هایی به صورت بیتی و همچنین داده های نوع D در نظر گرفته شده‌اند که وظایف مشخصی را بر عهده دارند و یا عملکرد خاصی در PLC را مشخص می‌کنند. از این حافظه‌ها نمی‌توان به صورت معمولی مانند دیگر حافظه ها برای ذخیره داده‌ها استفاده کرد و باید از آن‌ها با توجه به عملکردشان در برنامه بهره برد، بسیاری از این حافظه‌ها مانند پرچم وضعیت عمل می‌کنند و داده‌های آن‌ها را که در بازه‌های زمانی گوناگون به روز می‌شوند را تنها می‌توان برای خواندن مورد استفاده قرار داد. مروری بر این حافظه‌ها و استفاده از این حافظه‌های خاص در برنامه‌ها، می‌تواند موجب بهبود کیفیت و راحت‌تر شدن برنامه نویسی شود. همچنین در بسیاری از موارد مانند ارسال و دریافت داده‌ها از طریق شبکه MODBUS استفاده از این حافظه‌ها ضروری است. در ادامه لیست این حافظه‌ها را مرور می‌کنیم.

آدرس دهی زیر صرفا مربوط به PLCهای سری DVP است، البته حافظه‌های خاص در سری AH500 نیز با آدرس دهی و ساختاری متفاوت وجود دارد، که می‌توانید جهت اطلاعات بیشتر به دستورالعمل برنامه نویسی این PLC مراجعه نمایید.

۱– حافظه‌های بیتی خاص

Function Group	**PLC Operation Flag**
No.	**M1000 ~ M1003**

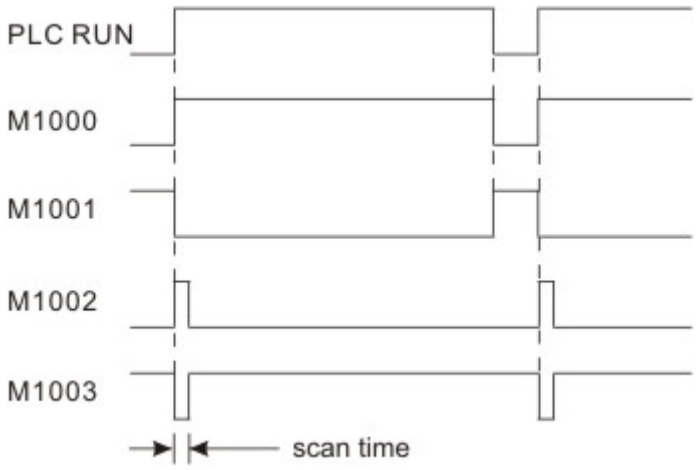
M1000: M1000 (A contact) is constantly “On” during operation and detection. When PLC is in RUN status, M1000 remains “On”.

M1001: M1001 (B contact) is constantly “On” during operation and detection. When PLC is in RUN status, M1001 remains “On”.

M1002: M1002 is “On” during the first scan when PLC starts to RUN and remains “Off” afterward. The pulse width = 1 scan time. Use this contact for all kinds of initial setting.

M1003: M1003 is “Off” during the first scan when PLC starts to RUN and remains “On” afterward. M1003 enables

negative-direction (“Off” immediately when RUN) pulses.



به عنوان نمونه مطابق توضیحات حافظه بیتی M1000 وضعیت Run سیستم را مشخص می-کند و حافظه بیتی M1002 اولین سیکل پس از Run فعال می‌شود که از آن می‌شود برای مقداردهی اولیه PLC استفاده کرد.

Function Group **Grammar Check**
No. **M1004, D1004, D1137**

1. When errors occur in grammer check, ERROR LED indicator flashes and special relay M1004 = On.
2. Timing for PLC grammar check:
   a. When the power goes from “Off” to “On”. b. When the program is written into PLC.
   c. When on-line editing is conducted.
3. Grammar check may start due to illegal use of instruction operands (devices) or incorrect program grammar loop. The error can be detected by the error code in D1004 and error table. The address where the error exists will be stored in D1137. (The address value in D1137 will be invalid if the error is a general loop error.)

Function Group **Data Backup Memory Card**
No. **M1005 ~ M1007**

When the data backup memory card is installed in EH MPU, MPU will operate according to the ON/OFF of switch on the card. If the switch is “On”, the following comparisons will be conducted and the card will be copied to MPU. If the switch is “Off”, MPU will not perform any action. **M1005:** M1005 = On: An error occurs in the comparison between the ciphers of MPU and the data backup memory card and MPU does not perform any action.

M1006: M1006 = On: The data backup memory card has not been initialized.

M1007: M1007 = On: Data in the program area of the data backup memory card do not exist, it means data doesn't exist in the program area of data backup memory card.

Function Group **Scan Time-Out Timer**

No. **M1008, D1008**

---

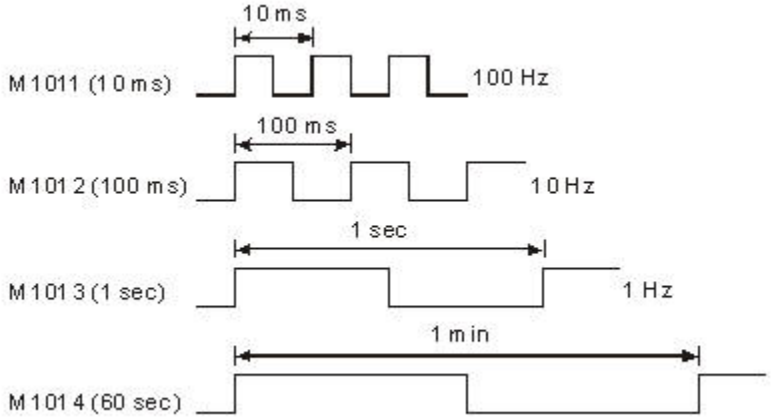
1. M1008 = On: Scan time-out occurs during the execution of the program, and PLC ERROR LED indicator remains beaconinf.
2. Users can use WPLSoft or HPP to monitor the content (STEP address when WDT timer is "On").

Function Group **Internal Clock Pulse**

No. **M1011 ~ M1014**

---

1. All PLC MPUs provide four different clock pulses. When PLC is powered, the four clock pulses will start automatically.
2. The clock pulses also start when PLC is in STOP status. The activation timing of clock pulses and that of RUN will not happen synchronously.



Function Group **High-Speed Timer**

No. **M1015, D1015**

---

1. High-speed counter is valid only when PLC is in RUN status for EH/EH2/SV, but is valid when PLC is in RUN or STOP stauts for SA/SX/SC.
2. M1015 = On: High-speed counter D1015 is enabled only whtn PLC scans to END instruction. (Min. timing unit of D1015: 100us)
3. Timing range of D1015: 0~32,767. When the timing reaches 32,767, the next timing restarts from 0.
4. M1015 = Off: D1015 stops timing immediately.

Function Group **RTC**

No. **M1016, M1017, M1076, D1313 ~ D1319**

1. Special M and special D relevant to RTC

Device	Name	Function
M1016	Year Display	OFF: display the last 2 digits of year in A.D ON: display the last 2 digits of year in A.D. plus 2,000
M1017	± 30 seconds correction	When triggered from “Off” to “On”, the correction is enabled. 0 ~ 29 second: minute intact; second reset to 0 30~ 59 second: mimute + 1; second reset to 0
D1313	Second	0~59
D1314	Minute	0~59
D1315	Hour	0~23
D1316	Day	1~31
D1317	Month	1~12
D1318	Week	1~7
D1319	Year	0 ~ 99 (last 2 digits of Year in A.D.)

2. SA/EH2 series: If the set value in RTC is incorrect, the time will be recovered as “Saturday, 00:00 Jan. 1, 2000” when PLC is powered and restarted.
3. ES2/EX2 series: If set value for RTC is invalid. RTC will display the time as Second: 0, Minute: 0, Hour: 0, Day: 1, Month: 1, Week: 1, Year: 0.
4. ES2/EX2 series: Memory of RTC is latched. RTC will resume the time when power is down. For higher accuracy of RTC, please conduction calibration on RTC when power resumes.
5. Methods of modifying RTC:
   - Apply TWR instruction to modify the built-in real time clock of DVP-ES2. Please refer to TWR for detail.
   - Use peripheral devices or WPLSoft / ISPSoft to set the RTC value.

Function Group **Communication Error Code**

No. **M1025, D1025**

When communication error occurs, M1025 = On and the error code is written into D1025. Error codes are:

01: Illegal instruction code

02: Illegal device address

03: Requested data exceed the range

07: checksum error

Function Group **Execution Complete Flag**

No. **M1029, M1030, M1036, M1039, M1102, M1103**

1. MTR, HKY, DSW, SEGL, PR: M1029 = On whenever the instruction completes one scan period.
2. PLSY, PLSR:
    a. M1029 will be “On” after Y0 pulse output of SA/SX/SC/ES/EX/SS is completed.

       M1030 will be “On” after Y1 pulse output is compeleted.

       When PLSY and PLSR instruction is “Off”, M1029 and M1030 turn “Off”. Users have to reset M1029 and M1030 after the action is completed.
    b. M1029 will be “On” after Y0 and Y1 pulse output of EH/EH2/SV is completed.

       M1030 will be “On” after Y2 and Y3 pulse output is compeleted.

       M1036 will be “On” after Y4 and Y5 pulse output of EH2/SV is completed. M1037 will be “On” after Y6 and Y7 pulse output is completed.

       When PLSY and PLSR instruction is “Off”, M1029, M1030, M1036 and M1037 turn “Off”. When the instruction is re-executed for the next time, M1029, M1030, M1036 and M1037 will turn “Off” and “On” again when the execution is completed.
    c. M1029 will be “ON” after Y0 and Y1 pulse output of ES2 is completed. M1030 will be “ON” after Y2 and Y3 pulse output is completed. When PLSY and PLSR instruction are OFF, M1029 and M1030 turn off as well. When the instruction is re-executed for the next time, M1029 and M1030 will turn off first then ON again when the instruction is completed. Users have to clear M1029 and M1030 manually.
3. INCD: M1029 will be “On” for a scan period when the assigned group number of data are compared.
4. RAMP, SORT:
    a. When the execution of the instruction is completed, M1029= On. Users have to reset M1029.
    b. M1029 turns “Off” when the instruction is “Off”.
5. DABSR:
    a. M1029= ON when instruction is completed.
    b. When the instruction is re-executed for the next time, M1029 will turn off first then ON again when the instruction is completed.
6. ES2/EX2 series ZRN, DRVI, DRVA:
    a. M1029 will be “ON” after Y0 and Y1 pulse output of ES2 is completed. M1102 will be “ON” after Y2 and Y3 pulse output is completed.
    b. When the instruction is re-executed for the next time, M1029 / M1102will turn off first then ON again when the instruction is completed.

Function Group **Clear Instruction**
No. **M1031, M1032**

M1031: Clear non-latched area

Cleared Devices:

Contact status of Y, general-purpose M and general-purpose S General-purpose contact and timing coil of T

General-purpose contact, counting coil reset coil of C General-purpose present value register of D

General-purpose present value register of T General-purpose present value register of C

M1032: Clear latched area

Cleared Devices:

Contact status of M and S for latched

Contact and timing coil of accumulative timer T

Contac and timing coil of high-speed counter C for latched

Present value register of D for latched

Present value register of accumulative timer T

Present value register of high-speed counter C for latched

Function Group **Output Latched During STOP** **No.** **M1033**

When M1033 = On and PLC goes from “RUN” to “STOP” the On/Off status of output is latched.

Function Group **All Y Outputs Prohibited**

No. **M1034**

When M1034 = On, all Y outputs turn “Off”.

Function Group **RUN / STOP Switch**

No. **M1035, D1035**

1. When M1035 = On, EH/EH2/SV determines the content (0 ~ 17) in D1035 to enable the RUN/STOP switch in X0 ~ X17.
2. When M1035 = On, SA/SX/SC enables the RUN/STOP switch in X7 (in SA), X3 (in SX) and X5 (in SC).
3. When M1035 = ON, DVP-ES2 uses input point X7 as the switch of RUN/STOP.

Function Group **Detecting Speed of X0 ~ X5**

No. **M1036**

For SC_V1.4 and above, SPD can detect the speed of X0 ~ X5 at the same time. The total bandwidth is 40KHz.

Function Group **Fixed Scan Time**
No. **M1039, D1039**

1. When M1039 = On, the scan time of program is determined by the content in D1039. When the execution of the program is completed, the next scan will take place when the fixed scan time is reached. If the content in D1039 is less than the actual scan time of the program, the scan time will follow the actual scan time of the program.

2. Instructions related to scan time, RAMP(API 67), HKY(API 71), SEGL(API 74), ARWS(API 75) and PR(API 77) should be used together with “fixed scan time” or “constant interruption”.

3. The scan time in D1010 ~ D1012 also includes constant scan time.

Function Group **Algorithm Error Flag**
No. **M1067, M1068, D1067, D1068**

1. Algorithm error flag:

Device	Description	Latched	STOP → RUN	RUN → STOP
M1067	Algorithm error flag	None	Cleared	Latched
M1068	Algorithm error locked flag	None	Latched	Latched
D1067	Algorithm error code	None	Cleared	Latched
D1068	STEP value when algorithm error	None	Latched	Latched

2. Error code explanation:

D1067 error code	Cause
H’ 0E18	BCD conversion error
H’ 0E19	Divisor is 0
H’ 0E1A	Use of device exceeds the range (including E, F index register modification)
H’ 0E1B	Square root value is negative
H’ 0E1C	FROM/TO instruction communication error

Function Group **X0 Detecting Pulse Width**
No. **M1084, D1023**

ES2 / EX2 series, when M1084 = ON, X6 pulse width detecting function is enabled and the detected pulse width is stored in D1023 (unit: 0.1ms)

M1083 On: detecting width of negative

half cycle (OFF→ON) M1083 Off :

detecting width of positive half cycle (ON→OFF)

Function Group **LV Signal**
No. **M1087, D1100**

1. When PLC detects LV (Low Voltage) signal, it will check if M1087 is “On” or not. If M1087 is “On”, the content in D1100 will be stored in Y0 ~ Y17.

2. Bit0 (LSB) of D1100 corresponds to Y0, bit1 corresponds to Y1, bit8 corresponds to Y10 and so on.

Function Group **File Register**
No. **M1101, D1101 ~ D1103**

When the power of PLC turns from “Off” to “On”, PLC determines whether to automatically send the content in the file register to the assigned data register by checking M1101, D1101 ~ D1103 (for SA/SX/SC/EH/EH2/SV).

M1101 Whether to automatically downland data from file register

D1101 Start No. of file register K0 ~ K1,599 (for SA/SX/SC) Start No. of file register K0 ~ K9,999 (for EH/EH2/SV)

D1102 Number of data read from file register K0 ~ K1,600 (for SA/SX/SC)
Number of data read from file register K0 ~ K8,000 (for EH/EH2/SV)

D1103 Location for storing data read from file register
Start No. of assigned data register D K2,000 ~ K4,999 (for SA/SX/SC) Start No. of assigned data register D K2,000 ~ K9,999 (for EH/EH2/SV)

Function Group **Digital Switch Function Card**
No. **M1104 ~ M1111**

1. When PLC is in RUN status with digital switch function card inserted, the 8 DIP switches amd their status orrespond respectively to M1104 ~ M1111.

2. When PLC is in RUN status with 4DI card inserted into the input AX0 (photocoupler isolation), the status of AX0 ~ AX3 correspond respectively to M1104 ~ M1107.

Function Group **Transistor Output Function Card**
No. **M1112, M1113**

When PLC is in RUN status with 2DO function card inserted, M1112 and M1113 correspond respectively to 2 transistors output points, AY0

and AY1.

Function Group **Pulse Output with Speed Acceleration/Deceleration**
No. **M1115 ~ M1119, D1104**

1. Special D and special M for acceleration/ deceleration of speed pulse output for SA/SX/SC (not applicable to SC_V1.4

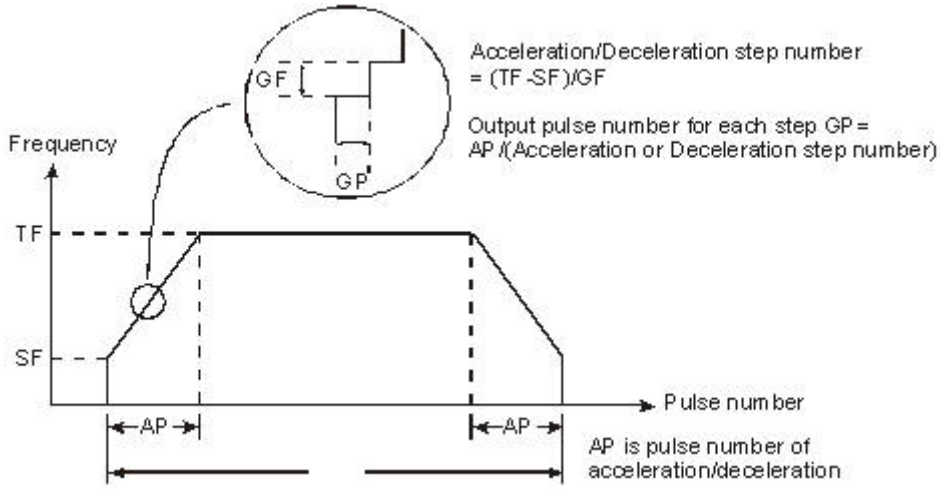
and versions above):

No.	Function
M1115	Activation switch
M1116	“Accelerating” flag
M1117	“Target frequency reached” flag
M1118	“Decelerating” flag
M1119	“Function completed” flag
D1104	Start No. of control register (D)

2. Parameters for D1104 (frequency range: 25Hz ~ 10KHz)

Index	Function	
+ 0	Start frequency (SF)	
+ 1	Gap frequency (GF)	
+ 2	Target frequency (TF)	
+ 3	The lower 16 bits of the 32 bits for the total number of output pulses	(TP)
+ 4	The higher 16 bits of the 32 bits for the total number of output pulses	
+ 5	The lower 16 bits of the 32 bits for the total number of output pulses in accelerating/decelerating section	(AP)
+ 6	The higher 16 bits of the 32 bits for the total number of output pulses in accelerating/decelerating section	

3. No instruction is needed, users need only to fill out the parameter table and enable M1115 (in RUN mode). This functio only supports Y0 output and the timing chart is as below.



4. Note: this function is applicable only when “all” the conditions below are met. a. Start frequency < target frequency.

b. Gap frequency ≤ (target frequency – start frequency)

c. Total number of pulses > (accel/decel number of pulses × 2)

d. For start frequency and target frequency: Min. 25Hz; Max. 10KHz e. Number of accel/decel pulses > number of accel/decel sections

f. When M1115 turns from “On” to “Off”, M1119 will be reset and M1116, M1117 and M1118 remain unchanged.

When PLC goes from “STOP” to “RUN”, M1115 ~ M1119 will be reset as “Off”. D1104 will only be cleared as “0”

when it turns from “Off” to “On”.

g. Either accel/decel pulse output function or PLSY Y0 output can be executed at a time when PLC is operating.

Function Group **Communication Port Function**

No. **M1120, M1136, M1138, M1139, M1143, D1036, D1109, D1120**

Port / Item	COM1	COM2	COM3
Communication format	D1036	D1120	D1109
Communication setting holding	M1138	M1120	M1136
ASCII/RTU mode	M1139	M1143	M1320
Slave communication address	D1121		D1255

1. Supports ES/EX/SS_V6.0/SA/SX_V1.2/SC_V1.0/EH_V1.1/SV_V1.0 and versions above.

2. COM ports (COM1: RS-232, COM2: RS-485, COM3: RS-485) in DVP-ES2 support communication format of MODBUS ASCII/RTU with a transmission speed up to 921kbps. COM1, COM2 and COM3 can be used simultaneously.

COM1 Can be used in master or slave mode. Supports ASCII/RTU communication format, baudrate (115200bps max), and modification on data length (data bits, parity bits, stop bits), D1036: COM1 (RS-232) communication protocol of master/slave PLC. (b8-b15 is not used)

COM2 Can be used in master or slave mode. Supports ASCII/RTU communication format, baudrate (921kbps max), and modification on data length (data bits, parity bits, stop bits), D1120: COM2 (RS-485) communication protocol of master/slave PLC.

COM3 (921kbps Can be used in master or slave mode. Supports ASCII/RTU communication format, baudrate max), and modification on data length (data bits, parity bits, stop bits), D1109: COM3 (RS-485) communication protocol of master/slave PLC. (b8-b15 is not used)

3. Communication Format Settings:

COM1: Communication format is set in D1036. (b8 ~ b15) do not support
Communication setting in M1138 remains
M1139 is set in ASCII/RTU mode

COM2: Communication format is set in D1120
Communication setting in M1120 remains
M1143 is set in ASCII/RTU mode

COM3: Communication format is set in D1109. (b0 ~ b3, b8 ~ b15) do not support
Communication setting in M1136 remains

4. Communication protocols and how to set:

	Content		
b0	Data Length	0: 7 data bits, 1: 8 data bits	
b1 b2	Parity bit	00: None 01: Odd 11: Even	
b3	Stop bits	0: 1 bit, 1: 2bits	
b4 b5 b6 b7	Baud rate	0001(H1): 110 0010(H2): 150 0011(H3): 300 0100(H4): 600 0101(H5): 1200 0110(H6): 2400 0111(H7): 4800 1000(H8): 9600 1001(H9): 19200 1010(HA): 38400 1011(HB): 57600 1100(HC): 115200 1101(HD): 500000 (COM2 / COM3 support)	
b8	Select start bit	0: None	1: D1124
b9	Select the 1st end bit	0: None	1: D1125
b10	Select the 2nd end bit	0: None	1: D1126
b11~b15	No definition		

Function Group **Special High-Speed Pulse Output**
No. **M1133 ~ M1135, D1133**

1. Special D and special M for special high-speed pulse Y0 (50KHz)for SA/SX/SC (not applicable to SC_V1.4 and versions above):

No.	Function
M1133	Output switch for special high-speed pulse Y0 (50KHz) (On = enabled)
M1134	On = Continuous output switch for special high-speed pulse Y0 (50KHz)
M1135	“Number of pulses reached” flag for special high-speed pulse Y0 (50KHz)
D1133	Start No. of control register (D) for special high-speed pulse Y0 (50KHz)

2. Parameters for D1133:

Index	Function
+ 0	The lower 16 bits of the 32 bits for output frequency of special high-speed pulse Y0
+ 1	The higher 16 bits of the 32 bits for output frequency of special high-speed pulse Y0
+ 2	The lower 16 bits of the 32 bits for number of output pulses of special high-speed pulse Y0
+ 3	The higher 16 bits of the 32 bits for number of output pulses of special high-speed pulse Y0
+ 4	The lower 16 bits of the 32 bits of the present number of special high-speed pulses Y0
+ 5	The higher 16 bits of the 32 bits of the present number of special high-speed pulses Y0

3. The function:

All output frequency and number of pulses stated in the table above can be modified when M1133 = On and M1135 = On. The modification will not affect the present output pulses. The present number of output pulses is updated in every scan time. When M1133 turns from “Off” to “on”, the number will be cleared as “0”. When 1133 turns from “On” to “Off”, the last numner of output pulses will be shown.

4. Note:

The special high-speed pulse output function can only be used on specific Y0 output point when PLC is in RUN status. It can coexist with PLSY (Y0) in the program and PLSY (Y1) will not be affected. If PLSY (Y0) instruction is executed prior to this function, the function cannot be used and vice versa. When the function is executed, the general function, general Y0 output will be invalid but Y1 ~ Y7 can be used.

Function Group **Two-axis Synchronous Control**
No. **M1133, M1135, D1133 ~ D1136**

1. Special D and special M for 2-axis synchronous drawing oblique and arc for SC_V1.4 and versions above:

No.	**Function**
M1133	Start flag for Y10 output for two-axis synchronous control
M1135	Start flag for Y11 output for two-axis synchronous control

D1133	Start No. of control register (D) for Y10 output for two-axis synchronous control
D1134	Number of sections for Y10 output for two-axis synchronous control
D1135	Start No. of control register (D) for Y11 output for two-axis synchronous control
D1136	Number of sections for Y11 output for two-axis synchronous control

2. Parameters for D1133, D1135:

Index	Function
+ 0	Y10, Y11 two-axis synchronous control; output frequency of 1st segment = low 16 bits of 32 bits
+ 1	Y10, Y11 two-axis synchronous control; output frequency of 1st segment = high 16 bits of 32 bits
+ 2	Y10, Y11 two-axis synchronous control; output pulse number of 1st segment = low 16 bits of 32 bits
+ 3	Y10, Y11 two-axis synchronous control; output pulse number of 1st segment = high 16 bits of 32 bits

3. The function:

a. Definition of the two axes:

X axis: Y0 (direction output) and Y10 (pulse output) Y axis: Y1 (direction output) and Y11 (pulse output)

b. Define the format of output table:

Assume D1133 = K100 and D1134 = K3 and the output table has to be set as:

Segment No.	Device D	Output frequency	Device D	Number of Output pulses	Description
1	D101,D100	K10,000	D103,D102	K1,000	Segment 1 outputs 1,000 pulses in 10KHz
2	D105,D104	K15,000	D107,D106	K2,000	Segment 2 outputs 2,000 pulses in 15KHz
3	D109,D108	K5,000	D111,D110	K3,000	Segment 3 outputs 3,000 pulses in 5KHz

4. Note:

Note: The frequency and number of output pulses are all in 32-bit. Thus, the 3 segments will continuously occupy 12 D devices (3 × 2 × 2 = 12).

a. Make sure that the output frequency and the number of pulses have been set before using this function. The output frequency and the number of pulses cannot be modified during the execution of the function.

b. When PLC program scans to END instruction, it will auto-check whether this function needs to be enabled. c. When M1133 and M1135 are set in the same scan period, the two axes will output pulses synchronously. d. When the output frequency < 100Hz, the output will be executed in 100Hz. When the output frequency > 100KHz, the output will be executed by 100KHz.

e. Only device D (D0 ~ D999 and D2000 ~ D4999) can be used for this function. DO NOT use other devices or exceed the range of device D.

f. The maximum number of segments for this function is 50. When the number of segments < 1 or > 50, this function will be disabled.

g. After this function is enabled, M1102 = “On” indicates Y10 output is completed and M1103 = “On” indicates Y11 output is completed.

Function Group **Adjustable Pulse Speed Acceleration/Deceleration**
No. **M1144 ~ M1149, M1154, D1030, D1031, D1144, D1154, D1155**

1. Special D and special M of Y0 adjustable pulse speed acceleration/deceleration for SA/SX/SC:

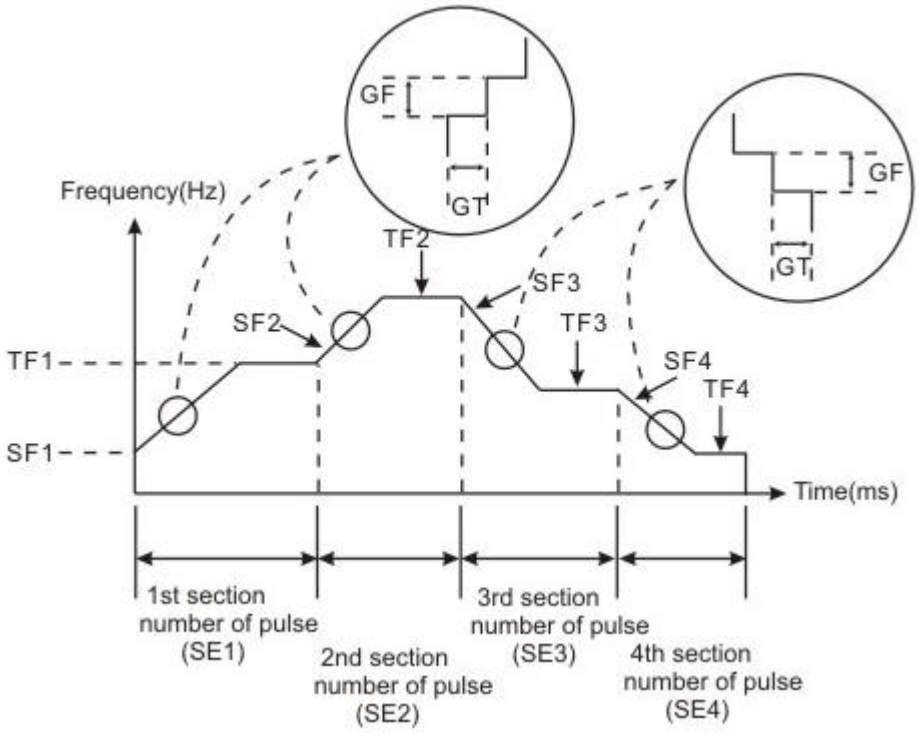
No.	Function
M1144	Activation switch for Y0 adjustable pulse speed acceleration/deceleration
M1145	Accerlerating flag for Y0 adjustable pulse speed acceleration/deceleration
M1146	“Target frequency reached” flag for Y0 adjustable pulse speed acceleration / deceleration
M1147	“Decerlerating” flag for Y0 adjustable pulse speed acceleration / deceleration
M1148	“Function completed” flag for Y0 adjustable pulse speed acceleration/deceleration
M1149	“Counting temporarily stops” flag for Y0 adjustable pulse speed acceleration / deceleration
M1154	“Enabling deceleration” flag for Y0 adjustable pulse speed acceleration/deceleration
D1030	The lower 16 bits in the 32-bit data register for accumulative Y0 output pulses
D1031	The higher 16 bits in the 32-bit data register for accumulative Y0 output pulses
D1144	Starting No. of the register (D) for Y0 adjustable pulse speed acceleration/deceleration
D1154	Recommended value for indicated gap time of deceleration (10 ~ 32,767 ms)
D1155	Recommended value for indicated gap frequency of deceleration (-1 ~ -32,700 Hz)

2. Parameters for D1144:

Index	Function
+ 0	Total number of sections (n) (max. 10)
+ 1	Currently executed section (read only)
+ 2	Start frequency of the 1st section (SF1)
+ 3	Gap time of the 1st section (GT1)
+ 4	Gap frequency of the 1st section (GF1)
+ 5	Target frequency of the 1st section (TF1)
+ 6	The lower 16 bits of the 32 bits for the target number of output pulses in the 1st section (SE1)
+ 7	The higher 16 bits of the 32 bits for the target number of output pulses in the 1st section (SE1)
+ 8	Start frequency of the 2nd section (SF2); Cannot be the same as TF1
+ 9	Gap time of the 2nd section (GT2)
+ 10	Gap frequency of the 2nd section (GF2)
+ 11	Target frequency of the 2nd section (TF2)
+ 12	The lower 16 bits of the 32 bits for the target number of output pulses in the 2nd section (SE2)
+ 13	The higher 16 bits of the 32 bits for the target number of output pulses in the 2nd section (SE2)
:	:
+ n*6 + 2	Start frequency of the nth section (SFn); Cannot be the same as the start frequency of the n-1th section (TFn-1)
+ n*6 + 3	Gap time of the nth section (GTn)
+ n*6 + 4	Gap frequency of the nth section (GFn)
+ n*6 + 5	Target frequency of the nth section (TFn)
+ n*6 + 6	The lower 16 bits of the 32 bits for the target number of output pulses in the nth section (SEn)
+ n*6 + 7	The higher 16 bits of the 32 bits for the target number of output pulses in the nth section (SEn)

3. The function:

This function can only be used on Y1 output point and the timing chart is as follows. After filling out the parameter table, setup M1144 to start the function (should be applied in RUN mode).



4. Usage rule and restriction:

a. The minimum frequency of start frequency and target frequency should be equal to or greater than 200Hz. If it is less than 200Hz, it means finish executing or not to execute.

b. The maximum frequency of start frequency of target frequency is 32700Hz. It will execute in 32700Hz as it is greater than 32700Hz.

c. The interval time range is 1~32767ms and its unit is ms

d. The interval frequency range in acceleration segment is 1Hz~32700Hz and in deceleration segment is -1~-32700Hz. If it is set to 0Hz, the executed segment can't be up to target frequency, but it will transfer to execute next segment after reaching target number.

e. Target number of segment pulse output should be greater than ((GF*GT/1000)* ((TF-SF)/GF). Refer to example 1 for detail. Once Target number of segment pulse output isn't greater than ((GF*GT/1000)* ((TF-SF)/GF), this function can't be used. The improve method is to add interval time or add target number of pulse output.

f. If there is Y0 output designated by high-speed instruction in RUN mode, Y0 output Instruction will be started as high priority.

g. After starting to execute M1144, if M1148 outputs without attaining completed function flag and M1144 is closed, this function will start deceleration function. If designated acceleration function flag M1154 is Off, it will reduce 200Hz per 200ms and stop output pulse till output frequency is less than 200Hz and set M1147 to deceleration flag. But if designated deceleration flag M1154 is On, it will be executed by interval time and

frequency that defined by user. And interval time can't be less than or equal to 0 (if it is less than or equal to 0, factory setting will be set to 200ms). Interval frequency can't be greater than or equal to 0 (factory setting will be set to -1KHz when it is equal to 0 and factory setting will be added negative sign automatically when it is greater than 0.)

h. When M1148 attains completed function flag and M1144 is closed, this function won't start deceleration function and it will clear M1148 flag. Once M1144 is closed, it will clear M1149 flag.

i. The execution segment of this function will execute by total segment number. The maximum segment is 10 segments

j. The acceleration/deceleration of this function will execute by start frequency of the next segment, i.e. when target frequency of execution segment is less than start frequency of the next segment, the next segment is acceleration and the target frequency of the next segment must be greater than start frequency of the next segment. When target frequency of execution segment is greater than the next segment frequency, the next segment is deceleration, therefore, target frequency of the next segment must be less than start frequency of the next segment. If user can't set it by this way, we can't ensure that you can get correct output pulse.

k. When STOP to RUN, M1144~M1149 will be cleared to Off. When RUN to STOP, M1144 will be cleared and M1145~M1149 won't be cleared. D1144 will be cleared to 0 when it is from Off to On and unchanged in other case.

l. The usage parameter range of PC/PA/PH series is D0~D999 and D2000~D4999.
It won't execute this command and close M1144 if parameter is out of range (includes all usage segment parameter).

Function Group **Single Step Execution**

No. **M1170, M1171, D1170**

---

1. Special D and special M for single step execution for EH/EH2/SV:

No.	Function
M1170	Start flag
M1171	Action flag
D1170	STEP No. of the currently executed instruction

2. The function:

a. Execution timing: The flag is valid only when PLC is in RUN status. b. Action Steps:

1) When M1170 is enabled, PLC enters the single step execution mode. PLC stays at a specific instruction, stores the location of STEP in D1170 and executes the instsruction once.

2) When M1171 is forced “On”, PLC executes the next instruction and stops. At the same time, PLC auto-force “Off” M1171 and stops at the next instruction. D1170 stores the present STEP value.

3) When Y output is in single step execution mode, Y outputs immediately without having to wait until END instruction is being executed.

3. Note:

a. Instruction that will be affected by scan time will be executed incorrectly due to the single step execution. For example, when HKY instruction is executed, it takes 8 scan times to obtain a valid input value from a key. Therefore, the single step execution will result in incorrect actions.

b. High-speed pulse input/output and high-speed counter comparison instructions are executed by hardware;

therefore, they will not be affected by the single step execution.

Function Group **Two-Phase Pulse Output**

No. **M1172 ~ M1174, D1172 ~ D1177**

1. Special D and special M for two-phase pulse output for SA/SX/SC:

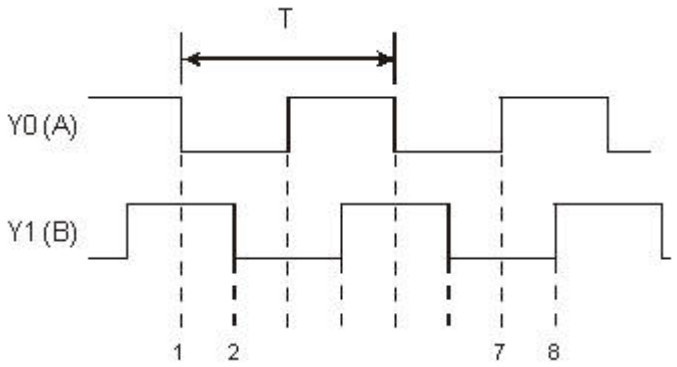
Device	Function Explanation
M1172	Switch for two-phase pulse output (On = enabled)
M1173	On = Continuous output switch
M1174	“Number of pulses reached” flag
D1172	Output frequency (12Hz ~ 20KHz)
D1173	Output mode (K1 and K2)
D1174	The lower 16 bits of the 32 bits for the target number of pulses
D1175	The higher 16 bits of the 32 bits for the target number of pulses
D1176	The lower 16 bits of the 32 bits for the present number of pulses
D1177	The higher 16 bits of the 32 bits for the present number of pulses

2. The function:

Output frequency = 1/1 pulse cycle period (i.e. 1/T; as the figure below)

There are two output modes. K1 refers to “A-phase ahead of B-phase” and K2 refers to “B-phase ahead of A-phase”. The number of pulses accumulates once whenever a phase gap occurs. For example, the number of pulses in the figure

below = 8, and when the number is reached, M1174 turns “On”. To clear the number, simply turn “Off” M1172.



The output frequency, target number of pulses and selection of modes can be modified when M1172 = On and M1174 = Off. Modification on output frequency and target number of pulses will not affect the present number of pulses, but when the mode is modified, the present number of pulses will be cleared as “0”. The present number of output pulses is updated in every scan time. When M1133 turns from “Off” to “on”, the number will be cleared as “0”. When M1172 is cleared as “0” when PLC goes from STOP to RUN. When PLC goes from RUN to STOP, the last number of pulses will be shown.

3. Note:

This function can only be used when PLC is in RUN status and can coexist with PLSY instruction in the program. If PLSY instruction is executed prior to this function, the function cannot be used and vice versa.

Function Group **VR Volume**

No. **M1178, M1179, D1178, D1179**

---

1. Special D and special M for built-in 2-point VR volume for EH/EH2/SV/SA/SC:

No.	Function
M1178	Enable VR0 volume
M1179	Enable VR1 volume
D1178	VR0 value
D1179	VR1 value

2. This fumction should be used when PLC is in RUN status. When M1178 = On, VR0 value will be converted into a value of 0 ~ 255 and stored in D1178. When M1179 = On, VR1 value will be converted into a value of 0 ~ 255 and stored in D1179.

Function Group **MODEM Connection**

No. **M1184 ~ M1188**

---

1. Special M for MODEM connection for EH:

No.	Function	Note
M1184	Enable MODEM	On: The following actions are valid
M1185	Initialize MODEM	Off: Initialization is completed
M1186	Fail to initialize MODEM	Off: M1185 = On
M1187	MODEM initialization is completed	Off: M1185 = On
M1188	Shows if MODEM is connected	On: Connected

2. How to connect:

(Please follow the steps below)

a. Set “On” M1184 (Enable PLC MODEM connection).
b. Set “On” M1185 (Enable initialization of MODEM from PLC).
c. Check if the initialization of MODEM is successful from M1186, M1187.
d. Wait for the connection.

3. Note:

a. When PLC is to be connected with MODEM, a RS-232 extension card is required. If there is no RS-232 extension card, all special M above will be invalid.

b. After enabling MODEM (M1184 = On), PLC has to initialize MODEM first (M1185 = On). If PLC fails to initialize MODEM, the auto-answering function of the MODEM will not be enabled.

c. After MODEM is initialized, it will enter auto-answering mode automatically.

d. If the distant PC is disconnected, MODEM will enter stand-by mode automatically and if the user turns off MODEM now, MODEM will have to be initialized again when it is turned on again.

e. The connection speed is set by PLC as 9,600bps fixed and modification on the speed is not allowed. MODEM has to be able to support the speed of 9,600bps and above.

f. The initialization format from PLC to MODEM are ATZ and ATS0 = 1.

g. If PLC fails to initialize MODEM, use the super terminal in PC to initialize it by the format ATZ and ATS0 = 1.

Function Group **Reverse Interrupt Trigger Pulse Direction**
No. **M1280, M1284, M1286**

1. The flag should be inserted before EI instruction
2. When M0 = OFF, M1280 = OFF. X0 external interrupt will be triggered by rising-edge pulse.
3. When M0 = ON, M1280 = ON. X0 external interrupt will be triggered by falling-edge pulse

Function Group **On / Off of Input Point X**
No. **M1304**

1. For SA/SX/SC, when M1304 = On, peripheral devices, e.g. WPLSoft or HPP, can force On/Off of X0 ~ X17, but the hardware LED will not respond to it.
2. For EH/EH2/SV, when M1304 = On, peripheral devices, e.g. WPLSoft or HPP, can force On/Off of input point X, and the hardware LED will respond to it.

Function Group **High-speed output pulse stop mode**
Number **M1310, M1311, M1334, M1335, D1166, D1167, D1343, D1353**

1. Special D and special M for high-speed pulse output stop mode:

No.	Function
M1334	Select stop mode for Y10 pulse
M1335	Select stop mode for Y11 pulse
M1310	Immediately stop Y10 pulse output
M1311	Immediately stop Y11 pulse output
D1166	X10 rising-edge/falling-edge counting mode switch
D1167	X11 rising-edge/falling-edge counting mode switch
D1343	Acceleration/deceleration time for Y10 pulse output
D1353	Acceleration/deceleration time for Y11 pulse output

2. How do Y10 pulse output stop modes work?

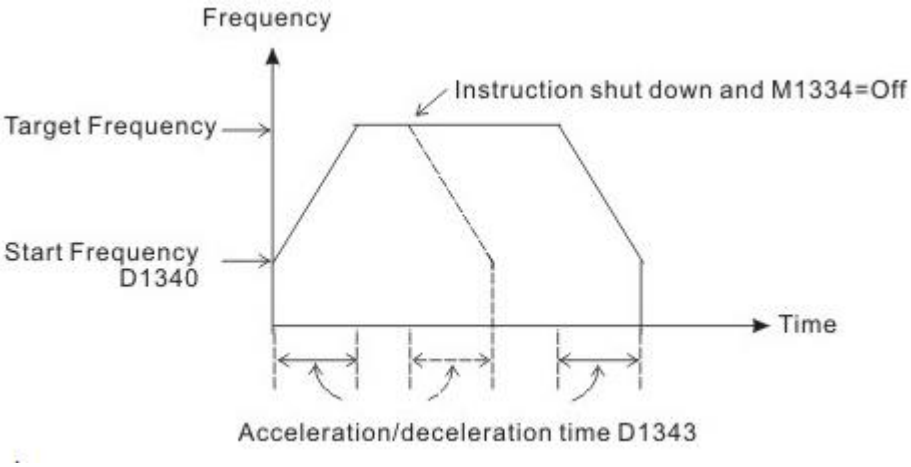
Mode 1 – Planned deceleration

a. Applicable to: DDRVI and DDRVA instructions

b. Criteria for executing planned deceleration: Shut down the criteria contact for pulse output instruction and turn “Off” M1334.

c. The time from executing planned deceleration to the end of pulse output: The time set in D1343 (for acceleration/deceleration)

d. The solid lines in figure 1 are the originally planned routes and the dotted lines refer to the routes after planned deceleration is executed.



(Figure 1)

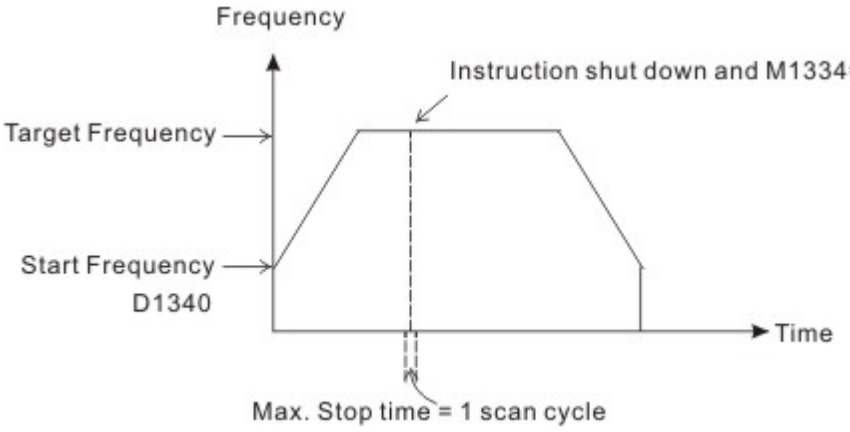
Mode 2 – Output shutdown

a. Applicable to: DDRVI, DDRVA, PLSY instructions

b. Criteria for executing output shutdown: Shut down the criteria contact for pulse output instruction and turn “On” M1334.(Because PLSY does not have acceleration/deceleration setting, M1334 does not need to be set in PLSY)

c. The time from executing output shutdown to the end of pulse output: Max. 1 scan period.

d. The solid lines in figure 2 are the originally planned routes and the dotted lines refer to the routes after output shutdown is executed.



(Figure 2)

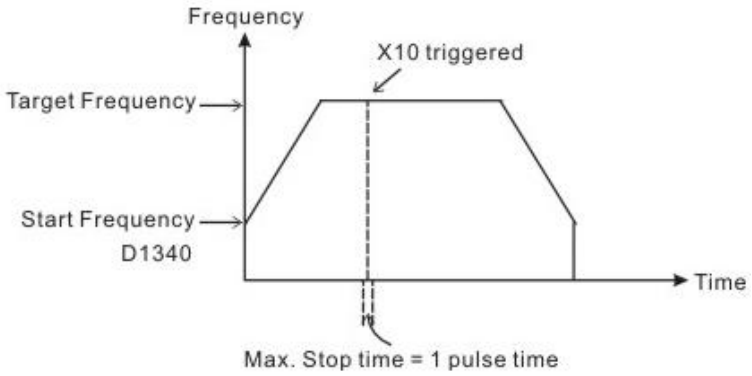
Mode 3 – Immediate output shutdown (supports SC_V1.4 and versions above)

a. Applicable to: DDRVI, DDRVA, PLSY instructions

b. Criteria for executing immediate output shutdown: M1310 = On (set before executing the instruction) and the criteria triggers set in X10 (D1166 = K0 refers to rising-edge; D1166 = K1 refers to falling-edge)

c. The time from executing immediate output shutdown to the end of pulse output: Max. 1 pulse time.

d. The solid lines in figure 3 are the originally planned routes and the dotted lines refer to the routes after X10 is triggered.



(Figure 3)

3. How do Y11 pulse output stop modes work?

Mode 1 – Planned deceleration

a. Applicable to: DDRVI and DDRVA instructions

b. Criteria for executing planned deceleration: Shut down the criteria contact for pulse output instruction and turn “Off” M1335.

c. The time from executing planned deceleration to the end of pulse output: The time set in D1353 (for acceleration/deceleration)

Mode 2 – Output shutdown

a. Applicable to: DDRVI, DDRVA, PLSY instructions

b. Criteria for executing output shutdown: Shut down the criteria contact for pulse output instruction and turn “On” M1335. (Because PLSY does not have acceleration/deceleration setting, M1335 does not need to be set in PLSY)

c. The time from executing output shutdown to the end of pulse output: Max. 1 scan period.

Mode 3 – Immediate output shutdown (supports SC_V1.4 and versions above)

a. Applicable to: DDRVI, DDRVA, PLSY instructions

b. Criteria for executing immediate output shutdown: M1311 = On (set before executing the instruction) and the criteria triggers set in X11 (D1167 = K0 refers to rising-edge; D1167 = K1 refers to falling-edge)

c. The time from executing immediate output shutdown to the end of pulse output: Max. 1 pulse time.

4. Note:

a. The execution criteria M1334 and M1335 for mode 1 and 2 have to be set before executing pulse output shutdown instruction. The execution criteria M1310, M1311 and

trigger criteria D1166, D1167 for mode 3 have to be set before the pulse output instruction is executed.

b. In mode 3 (immediate output shutdown), Y10 can only be used with X10 and Y11 with X11.

c. When using X10 or X11 in mode 3, DO NOT use X10 or X11 as the input high-speed counter.

Function Group	**Easy PLC Link**
Number	**M1350 ~ M1354, M1360 ~ M1519, D1355 ~ D1370, D1399, D1415 ~ D1465, D1480 ~ D1991**

1. Special D and special M for ID1 ~ ID8 of the 16 stations in EASY PLC LINK (M1353 = Off) for SA/SX/SC/EH/EH2/SV:

7. Explanation:

a. EASY PLC LINK is based on MODBUS communication protocol.

b. When Slave PLC is connected through COM1 / COM2 / COM3, baud rate and communication format of all Slaves must be the same (set in D1036). DVP-PLC supports both ASCII and RTU mode.

c. The ID number of the starting slave can be designated by D1399 and should be limited to the range K1~K214.

Slave ID cannot be repeated or the same as Master ID (set in D1121/D1255)

(به علت طولانی بودن این بخش حذف گردید، برای جزئیات بیشتر در مورد حافظه‌های مورد استفاده در ارتباط از طریق شبکه **MODBUS** به راهنمای نرم افزار مراجعه نمایید.)

۲– حافظه‌های داده خاص

Function Group	**Monitor Timer**
No.	**D1000**

1. Monitor timer is used for moitoring PLC scan time. When the scan time exceeds the set time in the moitor timer, the red ERROR LED indicator remains beaconing and all outputs will be “Off”.

2. The initial set value of the time in the monitor timer is 200ms. If the program is long or the operation is too complicated, MOV instruction can be used for changing the set value.

3. The maximum set value in the monitor timer is 32,767ms. Users can add WDT instruction (API 07) into PLC program.

When CPU execution progresses to WDT instruction, the internal monitor timer is cleared as 0 and the scan time will not exceed the set value in the monitor timer.

Function Group	**Program Capacity**
No.	**D1002**

Program capacity differs in different models.

1. ES / EX / SS series: 3,792 Steps (Word)

2. SA / SX / SC series: 7,920 Steps (Word)

3. EH / EH2 / SV series: 15,872 Steps (Word)

4. ES2 / EX2 series: 15,872 Steps (Word)

Function Group **Grammar Check**
No. **M1004, D1004, D1137**

1. When errors occur in grammer check, ERROR LED indicator flashes and special relay M1004 = On.

2. Timing for PLC grammar check:

a. When the power goes from “Off” to “On”. b. When the program is written into PLC.

c. When on-line editing is conducted.

3. Grammar check may start due to illegal use of instruction operands (devices) or incorrect program grammar loop. The error can be detected by the error code in D1004 and error table. The address where the error exists will be stored in D1137. (The address value in D1137 will be invalid if the error is a general loop error.)

Function Group **Scan Time-Out Timer**
No. **M1008, D1008**

1. M1008 = On: Scan time-out occurs during the execution of the program, and PLC ERROR LED indicator remains beaconinf.

2. Users can use WPLSoft or HPP to monitor the content (STEP address when WDT timer is “On”).

Function Group **Scan Time Monitor**
No. **D1010 ~ D1012**

The present value, minimum value and maximum value of scan time are stored in D1010 ~ D1012.

D1010: Present scan time value

D1011: Minimum scan time value

D1012: Maximum scan time value

Function Group **High-Speed Timer**
No. **M1015, D1015**

1. High-speed counter is valid only when PLC is in RUN status for EH/EH2/SV, but is valid when PLC is in RUN or STOP stauts for SA/SX/SC.

2. M1015 = On: High-speed counter D1015 is enabled only whtn PLC scans to END instruction. (Min. timing unit of D1015: 100us)

3. Timing range of D1015: 0~32,767. When the timing reaches 32,767, the next timing restarts from 0.

4. M1015 = Off: D1015 stops timing immediately.

Function Group π (PI)

No. D1018, D1019

1. D1018 and D1019 are combined as 32-bit data register for storing the floating point value of π (PI)
2. Floating point value = H 40490FDB

Function Group Adjustment on Input Terminal Responding Time

No. D1020, D1021

1. D1020 can be used for setting up the response time of receiving pulses at X0 ~X7 for SS / ES / EXi SA / SX / SC/ES2 / EX2 series MPU. Default: 10ms, 0~20ms adjustable.
2. D1021 can be used for setting up the responding time of receiving pulses at X10 ~X17 for ES. (Setup range: 0 ~ 20; Unit: ms)
3. D1021 can be used for setting up the responding time of receiving pulses at X10 ~X11 for SC. (Setup range: 0 ~ 1,000; Unit: time)
4. D1020 can be used for setting up the responding time of receiving pulses at X0 ~X7 for EH/EH2. (Setup range: 0 ~ 60; Unit: ms)
5. D1021 can be used for setting up the responding time of receiving pulses at X10 ~X17 for EH/EH2. (Setup range: 0 ~ 60; Unit: ms)
6. When the power of PLC goes from “Off” to “On”, the content of D1020 and D1021 turn to 10 automatically.
7. If the following programs are executed during the program, the responding time of X0 ~ X7 will be set to 0ms. The fastest responding time of input terminals is 50μs due to that all terminals are connected with RC filter loop.
8. There is no need to make adjustment on responding time when using high-speed counters and interruptions during the program.

Function Group X0 Detecting Pulse Width

No. M1084, D1023

ES2 / EX2 series, when M1084 = ON, X6 pulse width detecting function is enabled and the detected pulse width is stored in D1023 (unit: 0.1ms)

M1083 On: detecting width of negative half cycle (OFF→ON) M1083 Off: detecting width of positive half cycle (ON→OFF)

Function Group Communication Error Code

No. M1025, D1025

When communication error occurs, M1025 = On and the error code is written into D1025. Error codes are:

01: Illegal instruction code

02: Illegal device address

03: Requested data exceed the range

07: checksum error

Function Group **Adjustable Pulse Speed Acceleration/Deceleration**

No. **M1144 ~ M1149, M1154, D1030, D1031, D1144, D1154, D1155**

1. Special D and special M of Y0 adjustable pulse speed acceleration/deceleration for SA/SX/SC:

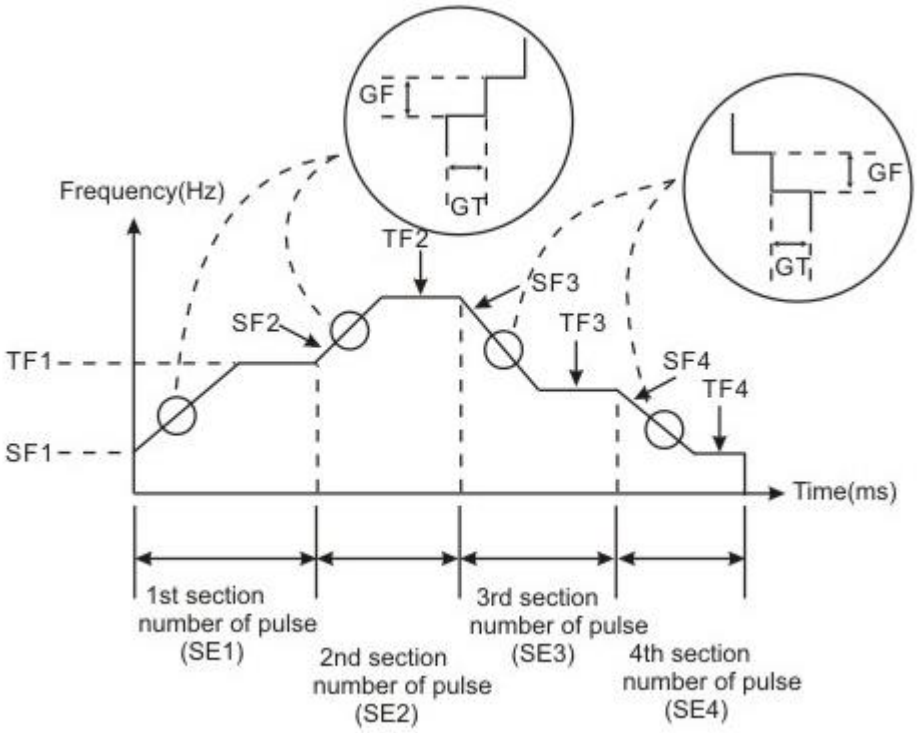
No.	Function
M1144	Activation switch for Y0 adjustable pulse speed acceleration/deceleration
M1145	Accerlerating flag for Y0 adjustable pulse speed acceleration/deceleration
M1146	“Target frequency reached” flag for Y0 adjustable pulse speed acceleration / deceleration
M1147	“Decerlerating” flag for Y0 adjustable pulse speed acceleration / deceleration
M1148	“Function completed” flag for Y0 adjustable pulse speed acceleration/deceleration
M1149	“Counting temporarily stops” flag for Y0 adjustable pulse speed acceleration / deceleration
M1154	“Enabling deceleration” flag for Y0 adjustable pulse speed acceleration/deceleration
D1030	The lower 16 bits in the 32-bit data register for accumulative Y0 output pulses
D1031	The higher 16 bits in the 32-bit data register for accumulative Y0 output pulses
D1144	Starting No. of the register (D) for Y0 adjustable pulse speed acceleration/deceleration
D1154	Recommended value for indicated gap time of deceleration (10 ~ 32,767 ms)
D1155	Recommended value for indicated gap frequency of deceleration (-1 ~ -32,700 Hz)

2. Parameters for D1144:

Index	Function
+ 0	Total number of sections (n) (max. 10)
+ 1	Currently executed section (read only)
+ 2	Start frequency of the 1st section (SF1)
+ 3	Gap time of the 1st section (GT1)
+ 4	Gap frequency of the 1st section (GF1)
+ 5	Target frequency of the 1st section (TF1)
+ 6	The lower 16 bits of the 32 bits for the target number of output pulses in the 1st section (SE1)
+ 7	The higher 16 bits of the 32 bits for the target number of output pulses in the 1st section (SE1)
+ 8	Start frequency of the 2nd section (SF2); Cannot be the same as TF1
+ 9	Gap time of the 2nd section (GT2)
+ 10	Gap frequency of the 2nd section (GF2)
+ 11	Target frequency of the 2nd section (TF2)
+ 12	The lower 16 bits of the 32 bits for the target number of output pulses in the 2nd section (SE2)
+ 13	The higher 16 bits of the 32 bits for the target number of output pulses in the 2nd section (SE2)
:	:
+ n*6 + 2	Start frequency of the nth section (SFn); Cannot be the same as the start frequency of the n-1th section (TFn-1)
+ n*6 + 3	Gap time of the nth section (GTn)
+ n*6 + 4	Gap frequency of the nth section (GFn)
+ n*6 + 5	Target frequency of the nth section (TFn)
+ n*6 + 6	The lower 16 bits of the 32 bits for the target number of output pulses in the nth section (SEn)
+ n*6 + 7	The higher 16 bits of the 32 bits for the target number of output pulses in the nth section (SEn)

3. The function:

This function can only be used on Y1 output point and the timing chart is as follows. After filling out the parameter table, setup M1144 to start the function (should be applied in RUN mode).



4. Usage rule and restriction:
   a. The minimum frequency of start frequency and target frequency should be equal to or greater than 200Hz. If it is less than 200Hz, it means finish executing or not to execute.
   b. The maximum frequency of start frequency of target frequency is 32700Hz. It will execute in 32700Hz as it is greater than 32700Hz.
   c. The interval time range is 1~32767ms and its unit is ms
   d. The interval frequency range in acceleration segment is 1Hz~32700Hz and in deceleration segment is -1~-32700Hz. If it is set to 0Hz, the executed segment can't be up to target frequency, but it will transfer to execute next segment after reaching target number.
   e. Target number of segment pulse output should be greater than ((GF*GT/1000)* ((TF-SF)/GF). Refer to example 1 for detail. Once Target number of segment pulse output isn't greater than ((GF*GT/1000)* ((TF-SF)/GF), this function can't be used. The improve method is to add interval time or add target number of pulse output.
   f. If there is Y0 output designated by high-speed instruction in RUN mode, Y0 output Instruction will be started as high priority.
   g. After starting to execute M1144, if M1148 outputs without attaining completed function flag and M1144 is closed, this function will start deceleration function. If designated acceleration function flag M1154 is Off, it will reduce 200Hz per 200ms and stop output pulse till output frequency is less than 200Hz and set M1147 to deceleration flag. But if designated deceleration flag M1154 is On, it will be executed by interval time and frequency that defined by user. And interval time can't be less than or equal to 0 (if it is less than or equal to 0, factory setting will be set to 200ms). Interval frequency can't be greater than or equal to 0 (factory setting will be set to -1KHz when it is equal to 0 and factory setting will be added negative sign automatically when it is greater

than 0.)

h. When M1148 attains completed function flag and M1144 is closed, this function won't start deceleration function and it will clear M1148 flag. Once M1144 is closed, it will clear M1149 flag.

i. The execution segment of this function will execute by total segment number. The maximum segment is 10 segments

j. The acceleration/deceleration of this function will execute by start frequency of the next segment, i.e. when target frequency of execution segment is less than start frequency of the next segment, the next segment is acceleration and the target frequency of the next segment must be greater than start frequency of the next segment. When target frequency of execution segment is greater than the next segment frequency, the next segment is deceleration, therefore, target frequency of the next segment must be less than start frequency of the next segment. If user can't set it by this way, we can't ensure that you can get correct output pulse.

k. When STOP to RUN, M1144~M1149 will be cleared to Off. When RUN to STOP, M1144 will be cleared and M1145~M1149 won't be cleared. D1144 will be cleared to 0 when it is from Off to On and unchanged in other case.

l. The usage parameter range of PC/PA/PH series is D0~D999 and D2000~D4999. It won't execute this command and close M1144 if parameter is out of range (includes all usage segment parameter).

Function Group **RUN / STOP Switch**
No. **M1035, D1035**

1. When M1035 = On, EH/EH2/SV determines the content (0 ~ 17) in D1035 to enable the RUN/STOP switch in X0 ~ X17.

2. When M1035 = On, SA/SX/SC enables the RUN/STOP switch in X7 (in SA), X3 (in SX) and X5 (in SC).

Function Group **Communication Port Function**
No. **M1120, M1136, M1138, M1139, M1143, D1036, D1109, D1120**

Item \ Port	COM1	COM2	COM3
Communication format	D1036	D1120	D1109
Communication setting holding	M1138	M1120	M1136
ASCII/RTU mode	M1139	M1143	M1320
Slave communication address	D1121		D1255

1. Supports ES/EX/SS_V6.0/SA/SX_V1.2/SC_V1.0/EH_V1.1/SV_V1.0 and versions above.

2. COM ports (COM1: RS-232, COM2: RS-485, COM3: RS-485) in DVP-ES2 support communication format of MODBUS ASCII/RTU with a transmission speed up to 921kbps. COM1, COM2 and COM3 can be used simultaneously.

COM1 Can be used in master or slave mode. Supports ASCII/RTU communication format, baudrate (115200bps max), and modification on data length (data bits, parity bits, stop bits), D1036: COM1 (RS-232) communication protocol of master/slave PLC. (b8-b15 is not used)

COM2 Can be used in master or slave mode. Supports ASCII/RTU communication format, baudrate (921kbps max), and modification on data length (data bits, parity bits, stop bits), D1120: COM2 (RS-485) communication protocol of master/slave PLC.

COM3 Can be used in master or slave mode. Supports ASCII/RTU communication format, baudrate (921kbps max), and modification on data length (data bits, parity bits, stop bits), D1109: COM3 (RS-485) communication protocol of master/slave PLC. (b8-b15 is not used)

3. Communication Format Settings:

COM1: Communication format is set in D1036. (b8 ~ b15) do not support

Communication setting in M1138 remains

M1139 is set in ASCII/RTU mode

COM2: Communication format is set in D1120

Communication setting in M1120 remains

M1143 is set in ASCII/RTU mode

COM3: Communication format is set in D1109. (b0 ~ b3, b8 ~ b15) do not support

Communication setting in M1136 remains

4. Communication protocols and how to set:

	Content		
b0	Data Length	0: 7 data bits, 1: 8 data bits	
b1 b2	Parity bit	00: None 01: Odd 11: Even	
b3	Stop bits	0: 1 bit, 1: 2bits	
b4 b5 b6 b7	Baud rate	0001(H1): 110 0010(H2): 150 0011(H3): 300 0100(H4): 600 0101(H5): 1200 0110(H6): 2400 0111(H7): 4800 1000(H8): 9600 1001(H9): 19200 1010(HA): 38400 1011(HB): 57600 1100(HC): 115200 1101(HD): 500000 (COM2 / COM3 support)	
b8	Select start bit	0: None	1: D1124
b9	Select the 1st end bit	0: None	1: D1125
b10	Select the 2nd end bit	0: None	1: D1126
b11~b15	No definition		

Function Group **Communication Response Delay**
No. **D1038**

1. When PLC is used as slave station, in RS-485 communication interface, users can set up communication response delay time ranging from 0 to 10,000 (0 ~ 1 second). If the time is without the range, D1038 = O (time unit: 0.1ms). The set value of time must be less than that in D1000(scan time-out timer WDT).
2. In PLC-Link, users can set up delayed transmission of the next communication data (unit: 1 scan period) for SA/SX/SC/EH/EH2/SV.
3. COM2, COM3 (RS-485): Data response delay time can be set when PLC is a Slave in RS-485 communication. Unit is 0.1ms. 0~10,000 adjustable.
4. COM2 (RS-485): By using PLC-Link, D1038 can be set to send next communication data with delay. (Unit: 1 scan period)

Function Group **Fixed Scan Time**
No. **M1039, D1039**

1. When M1039 = On, the scan time of program is determined by the content in D1039. When the execution of the program is completed, the next scan will take place when the fixed scan time is reached. If the content in D1039 is less than the actual scan time of the program, the scan time will follow the actual scan time of the program.

2. Instructions related to scan time, RAMP(API 67), HKY(API 71), SEGL(API 74), ARWS(API 75) and PR(API 77) should be used together with “fixed scan time” or “constant interruption”.

3. The scan time in D1010 ~ D1012 also includes constant scan time.

Function Group **Analog Function**
No. **D1056 ~ D1059, D1062, D1110 ~ D1113, D1116 ~ D1118**

1. The function is for EX2 MPU Only
2. If D1118 ≤ 2, it will be regarded as 2ms
3. Resolution of analog input channel : 12 bits.

Voltage: -10V~10V ⇔ -2000~2000(LSB). Current: -20mA~20mA⇔-2000~2000(LSB). Error: ± 2 0LSB

4. Resolution of analog output channel: 12 bits

Voltage: -10V~10V ⇔ -2000~2000(LSB). Current: 0~20mA ⇔ 0~4000(LSB) Error: ±0.05V or ±0.1mA

5. Default of analog input sampling range: (K2). If set value = K1, PLC takes the present value.

Device	Function
D1062	Sampling range of EX2 analog input channels (CH0~CH3): 1~20, Default = K2
D1110	Average value of EX2 analog input channel 0 (AD0)
D1111	Average value of EX2 analog input channel 1 (AD1)
D1112	Average value of EX2 analog input channel 2 (AD2)
D1113	Average value of EX2 analog input channel 3 (AD3)

D1115	Analog mode selection (Voltage / Current) bit0~bit5 indicates CH0~CH5. 0: voltage, 1: current
D1116	Analog output channel 0 (DA0)
D1117	Analog output channel 1 (DA1)
D1118	For EX2 series, sampling time of analog/digital conversion

Function Group **Algorithm Error Flag**

No. **M1067, M1068, D1067, D1068**

1. Algorithm error flag:

Device	Description	Latched	STOP → RUN	RUN → STOP
M1067	Algorithm error flag	None	Cleared	Latched
M1068	Algorithm error locked flag	None	Latched	Latched
D1067	Algorithm error code	None	Cleared	Latched
D1068	STEP value when algorithm error	None	Latched	Latched

2. Error code explanation:

D1067 error code	Cause
H’ 0E18	BCD conversion error
H’ 0E19	Divisor is 0
H’ 0E1A	Use of device exceeds the range (including E, F index register modification)
H’ 0E1B	Square root value is negative
H’ 0E1C	FROM/TO instruction communication error

Function Group **LV Signal**

No. **M1087, D1100**

1. When PLC detects LV (Low Voltage) signal, it will check if M1087 is “On” or not. If M1087 is “On”, the content in D1100 will be stored in Y0 ~ Y17.

2. Bit0 (LSB) of D1100 corresponds to Y0, bit1 corresponds to Y1, bit8 corresponds to Y10 and so on.

Function Group **File Register**

No. **M1101, D1101 ~ D1103**

When the power of PLC turns from “Off” to “On”, PLC determines whether to automatically send the content in the file register to the assigned data register by checking M1101, D1101 ~ D1103 (for SA/SX/SC/EH/EH2/SV).

M1101 Whether to automatically downland data from file register

D1101 Start No. of file register K0 ~ K1,599 (for SA/SX/SC)

Start No. of file register K0 ~ K9,999 (for EH/EH2/SV)

D1102 Number of data read from file register K0 ~ K1,600 (for SA/SX/SC)

Number of data read from file register K0 ~ K8,000 (for EH/EH2/SV)

D1103 Location for storing data read from file register

Start No. of assigned data register D K2,000 ~ K4,999 (for SA/SX/SC) Start No. of assigned data register D K2,000 ~ K9,999 (for EH/EH2/SV)

Function Group No. **Pulse Output with Speed Acceleration/Deceleration M1115 ~ M1119, D1104**

---

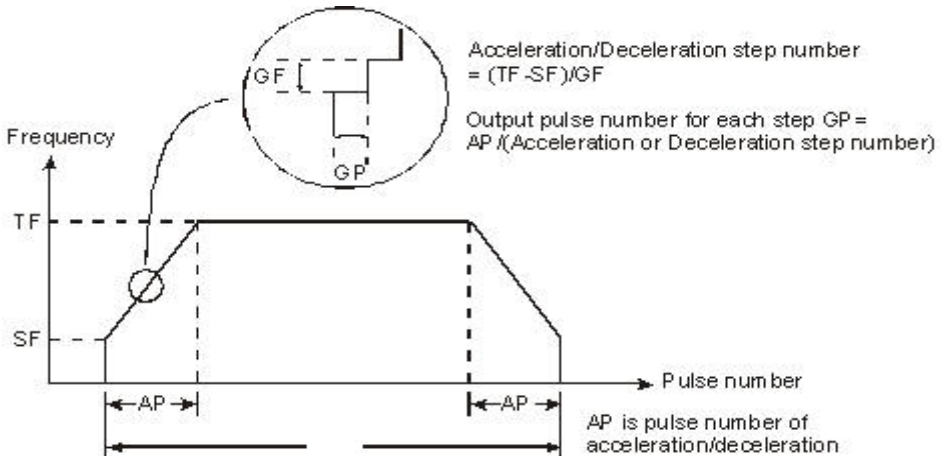
1. Special D and special M for acceleration/ deceleration of speed pulse output for SA/SX/SC (not applicable to SC_V1.4 and versions above):

No.	Function
M1115	Activation switch
M1116	“Accelerating” flag
M1117	“Target frequency reached” flag
M1118	“Decelerating” flag
M1119	“Function completed” flag
D1104	Start No. of control register (D)

2. Parameters for D1104 (frequency range: 25Hz ~ 10KHz)

Index	Function	
+ 0	Start frequency (SF)	
+ 1	Gap frequency (GF)	
+ 2	Target frequency (TF)	
+ 3	The lower 16 bits of the 32 bits for the total number of output pulses	(TP)
+ 4	The higher 16 bits of the 32 bits for the total number of output pulses	
+ 5	The lower 16 bits of the 32 bits for the total number of output pulses in accelerating/decelerating section	(AP)
+ 6	The higher 16 bits of the 32 bits for the total number of output pulses in accelerating/decelerating section	

3. No instruction is needed, users need only to fill out the parameter table and enable M1115 (in RUN mode). This functio only supports Y0 output and the timing chart is as below.



4. Note: this function is applicable only when “all” the conditions below are met. a. Start frequency < target frequency.

b. Gap frequency ≤ (target frequency – start frequency)

c. Total number of pulses > (accel/decel number of pulses × 2)

d. For start frequency and target frequency: Min. 25Hz; Max. 10KHz e. Number of accel/decel pulses > number of accel/decel sections

f. When M1115 turns from “On” to “Off”, M1119 will be reset and M1116, M1117 and M1118 remain unchanged. When PLC goes from “STOP” to “RUN”, M1115 ~ M1119 will be reset as “Off”. D1104 will only be cleared as “0” when it turns from “Off” to “On”.

g. Either accel/decel pulse output function or PLSY Y0 output can be executed at a time when PLC is operating.

Function Group **Special High-Speed Pulse Output**

No. **M1133 ~ M1135, D1133**

---

1. Special D and special M for special high-speed pulse Y0 (50KHz)for SA/SX/SC (not applicable to SC_V1.4 and versions above):

No.	Function
M1133	Output switch for special high-speed pulse Y0 (50KHz) (On = enabled)
M1134	On = Continuous output switch for special high-speed pulse Y0 (50KHz)
M1135	“Number of pulses reached” flag for special high-speed pulse Y0 (50KHz)
D1133	Start No. of control register (D) for special high-speed pulse Y0 (50KHz)

2. Parameters for D1133:

Index	Function
+ 0	The lower 16 bits of the 32 bits for output frequency of special high-speed pulse Y0
+ 1	The higher 16 bits of the 32 bits for output frequency of special high-speed pulse Y0
+ 2	The lower 16 bits of the 32 bits for number of output pulses of special high-speed pulse Y0
+ 3	The higher 16 bits of the 32 bits for number of output pulses of special high-speed pulse Y0
+ 4	The lower 16 bits of the 32 bits of the present number of special high-speed pulses Y0
+ 5	The higher 16 bits of the 32 bits of the present number of special high-speed pulses Y0

3. The function:

All output frequency and number of pulses stated in the table above can be modified when M1133 = On and M1135 = On. The modification will not affect the present output pulses. The present number of output pulses is updated in every scan time. When M1133 turns from “Off” to “on”, the number will be cleared as “0”. When 1133 turns from “On” to “Off”, the last numner of output pulses will be shown.

4. Note:

The special high-speed pulse output function can only be used on specific Y0 output point when PLC is in RUN status. It can coexist with PLSY (Y0) in the program and PLSY (Y1) will not be affected. If PLSY (Y0) instruction is executed prior to this function, the function cannot be used and vice versa. When the function is executed, the general function, general Y0 output will be invalid but Y1 ~ Y7 can be used.

1. Special D and special M for 2-axis synchronous drawing oblique and arc for SC_V1.4 and versions above:

No.	Function
M1133	Start flag for Y10 output for two-axis synchronous control
M1135	Start flag for Y11 output for two-axis synchronous control
D1133	Start No. of control register (D) for Y10 output for two-axis synchronous control
D1134	Number of sections for Y10 output for two-axis synchronous control
D1135	Start No. of control register (D) for Y11 output for two-axis synchronous control
D1136	Number of sections for Y11 output for two-axis synchronous control

2. Parameters for D1133, D1135:

Index	Function
+ 0	Y10, Y11 two-axis synchronous control; output frequency of 1st segment = low 16 bits of 32 bits
+ 1	Y10, Y11 two-axis synchronous control; output frequency of 1st segment = high 16 bits of 32 bits
+ 2	Y10, Y11 two-axis synchronous control; output pulse number of 1st segment = low 16 bits of 32 bits
+ 3	Y10, Y11 two-axis synchronous control; output pulse number of 1st segment = high 16 bits of 32 bits

3. The function:

a. Definition of the two axes:

X axis: Y0 (direction output) and Y10 (pulse output) Y axis: Y1 (direction output) and Y11 (pulse output)

b. Define the format of output table:

Assume D1133 = K100 and D1134 = K3 and the output table has to be set as:

Segment No.	Device D	Output frequency	Device D	Number of Output pulses	Description
1	D101,D100	K10,000	D103,D102	K1,000	Segment 1 outputs 1,000 pulses in 10KHz
2	D105,D104	K15,000	D107,D106	K2,000	Segment 2 outputs 2,000 pulses in 15KHz
3	D109,D108	K5,000	D111,D110	K3,000	Segment 3 outputs 3,000 pulses in 5KHz

4. Note:

Note: The frequency and number of output pulses are all in 32-bit. Thus, the 3 segments will continuously occupy 12 D devices (3 × 2 × 2 = 12).

a. Make sure that the output frequency and the number of pulses have been set before using this function. The output frequency and the number of pulses cannot be modified during the execution of the function.

b. When PLC program scans to END instruction, it will auto-check whether this function needs to be enabled.

c. When M1133 and M1135 are set in the same scan period, the two axes will output pulses synchronously.

d. When the output frequency < 100Hz, the output will be executed in 100Hz. When the output frequency >100KHz, the output will be executed by 100KHz.

e. Only device D (D0 ~ D999 and D2000 ~ D4999) can be used for this function. DO NOT use other devices or exceed the range of device D.

f. The maximum number of segments for this function is 50. When the number of segments < 1 or > 50, this function will be disabled.

g. After this function is enabled, M1102 = “On” indicates Y10 output is completed and M1103 = “On” indicates Y11 output is completed.

Function Group **Detecting Extension**
No. **D1140, D1142, D1143, D1145**

D1140 Number of special right-side extension modules (AD, DA, XA, PT, TC, HC, PU); Max. 8

D1142 Number of X input points on digital extension device

D1143 Number of Y output points on digital extension device

D1145 Number of special left-side extension modules (AD, DA, XA, PT, TC, HC, PU); Max. 8 (applicable to EH2/SV only)

Function Group **High-speed output pulse stop mode**
Number **M1310, M1311, M1334, M1335, D1166, D1167, D1343, D1353**

---

1. Special D and special M for high-speed pulse output stop mode:

No.	Function
M1334	Select stop mode for Y10 pulse
M1335	Select stop mode for Y11 pulse
M1310	Immediately stop Y10 pulse output
M1311	Immediately stop Y11 pulse output
D1166	X10 rising-edge/falling-edge counting mode switch
D1167	X11 rising-edge/falling-edge counting mode switch
D1343	Acceleration/deceleration time for Y10 pulse output
D1353	Acceleration/deceleration time for Y11 pulse output

2. How do Y10 pulse output stop modes work?

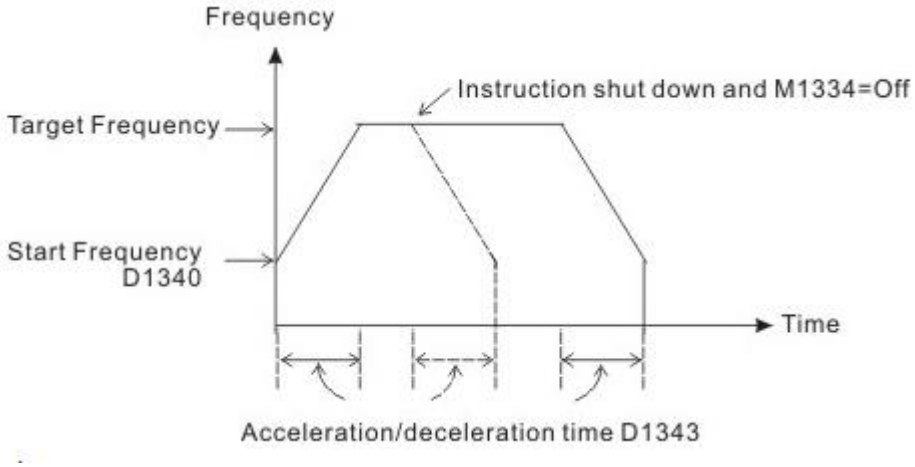
Mode 1 – Planned deceleration

a. Applicable to: DDRVI and DDRVA instructions

b. Criteria for executing planned deceleration: Shut down the criteria contact for pulse output instruction and turn “Off” M1334.

c. The time from executing planned deceleration to the end of pulse output: The time set in D1343 (for acceleration/deceleration)

d. The solid lines in figure 1 are the originally planned routes and the dotted lines refer to the routes after planned deceleration is executed.



(Figure 1)

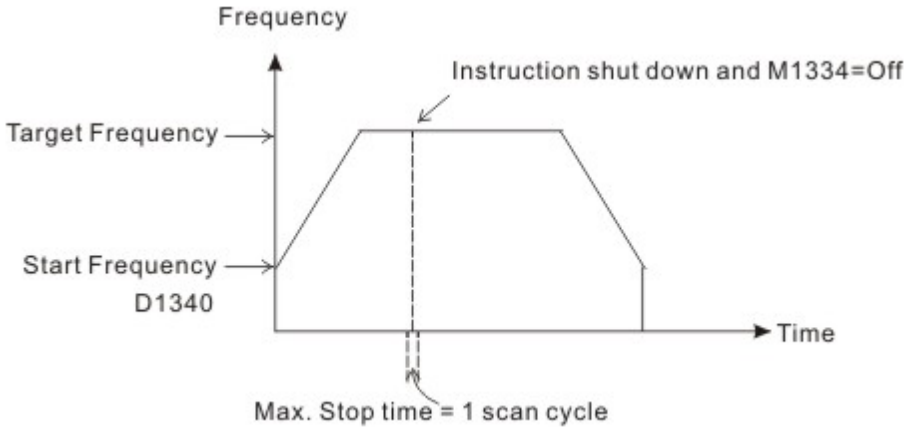
Mode 2 – Output shutdown

a. Applicable to: DDRVI, DDRVA, PLSY instructions

b. Criteria for executing output shutdown: Shut down the criteria contact for pulse output instruction and turn “On” M1334.(Because PLSY does not have acceleration/deceleration setting, M1334 does not need to be set in PLSY)

c. The time from executing output shutdown to the end of pulse output: Max. 1 scan period.

d. The solid lines in figure 2 are the originally planned routes and the dotted lines refer to the routes after output shutdown is executed.



(Figure 2)

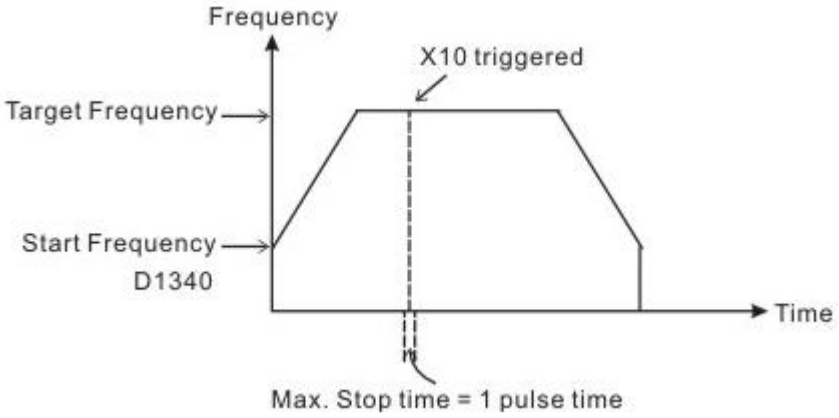
Mode 3 – Immediate output shutdown (supports SC_V1.4 and versions above)

a. Applicable to: DDRVI, DDRVA, PLSY instructions

b. Criteria for executing immediate output shutdown: M1310 = On (set before executing the instruction) and the criteria triggers set in X10 (D1166 = K0 refers to rising-edge; D1166 = K1 refers to falling-edge)

c. The time from executing immediate output shutdown to the end of pulse output: Max. 1 pulse time.

d. The solid lines in figure 3 are the originally planned routes and the dotted lines refer to the routes after X10 is triggered.



(Figure 3)

3. How do Y11 pulse output stop modes work?

Mode 1 – Planned deceleration

a. Applicable to: DDRVI and DDRVA instructions

b. Criteria for executing planned deceleration: Shut down the criteria contact for pulse output instruction and turn "Off" M1335.

c. The time from executing planned deceleration to the end of pulse output: The time set in D1353 (for acceleration/deceleration)

Mode 2 – Output shutdown

a. Applicable to: DDRVI, DDRVA, PLSY instructions

b. Criteria for executing output shutdown: Shut down the criteria contact for pulse output instruction and turn "On" M1335. (Because PLSY does not have acceleration/deceleration setting, M1335 does not need to be set in PLSY)

c. The time from executing output shutdown to the end of pulse output: Max. 1 scan period. Mode 3 – Immediate output shutdown (supports SC_V1.4 and versions above)

a. Applicable to: DDRVI, DDRVA, PLSY instructions

b. Criteria for executing immediate output shutdown: M1311 = On (set before executing the instruction) and the criteria triggers set in X11 (D1167 = K0 refers to rising-edge; D1167 = K1 refers to falling-edge)

c. The time from executing immediate output shutdown to the end of pulse output: Max. 1 pulse time.

4. Note:

a. The execution criteria M1334 and M1335 for mode 1 and 2 have to be set before executing pulse output shutdown instruction. The execution criteria M1310, M1311 and trigger criteria D1166, D1167 for mode 3 have to be set before the pulse output instruction is executed.

b. In mode 3 (immediate output shutdown), Y10 can only be used with X10 and Y11 with X11.

c. When using X10 or X11 in mode 3, DO NOT use X10 or X11 as the input high-speed counter.

Function Group **Single Step Execution**

No. **M1170, M1171, D1170**

1. Special D and special M for single step execution for EH/EH2/SV:

No.	Function
M1170	Start flag
M1171	Action flag
D1170	STEP No. of the currently executed instruction

2. The function:

a. Execution timing: The flag is valid only when PLC is in RUN status. b. Action Steps:

1) When M1170 is enabled, PLC enters the single step execution mode. PLC stays at a specific instruction, stores the location of STEP in D1170 and executes the instsruction once.

2) When M1171 is forced “On”, PLC executes the next instruction and stops. At the same time, PLC auto-force “Off” M1171 and stops at the next instruction. D1170 stores the present STEP value.

3) When Y output is in single step execution mode, Y outputs immediately without having to wait until END instruction is being executed.

3. Note:

a. Instruction that will be affected by scan time will be executed incorrectly due to the single step execution. For example, when HKY instruction is executed, it takes 8 scan times to obtain a valid input value from a key. Therefore, the single step execution will result in incorrect actions.

b. High-speed pulse input/output and high-speed counter comparison instructions are executed by hardware; therefore, they will not be affected by the single step execution.

Function Group **Two-Phase Pulse Output**

No. **M1172 ~ M1174, D1172 ~ D1177**

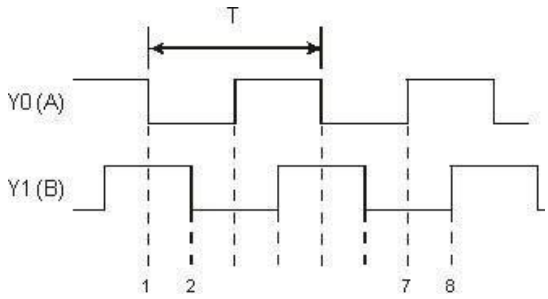
1. Special D and special M for two-phase pulse output for SA/SX/SC:

Device	Function Explanation
M1172	Switch for two-phase pulse output (On = enabled)
M1173	On = Continuous output switch
M1174	“Number of pulses reached” flag
D1172	Output frequency (12Hz ~ 20KHz)
D1173	Output mode (K1 and K2)
D1174	The lower 16 bits of the 32 bits for the target number of pulses
D1175	The higher 16 bits of the 32 bits for the target number of pulses
D1176	The lower 16 bits of the 32 bits for the present number of pulses
D1177	The higher 16 bits of the 32 bits for the present number of pulses

2. The function:

Output frequency = 1/1 pulse cycle period (i.e. 1/T; as the figure below)

There are two output modes. K1 refers to “A-phase ahead of B-phase” and K2 refers to “B-phase ahead of A-phase”. The number of pulses accumulates once whenever a phase gap occurs. For example, the number of pulses in the figure below = 8, and when the number is reached, M1174 turns “On”. To clear the number, simply turn “Off” M1172.



The output frequency, target number of pulses and selection of modes can be modified when M1172 = On and M1174 = Off. Modification on output frequency and target number of pulses will not affect the present number of pulses, but when the mode is modified, the present number of pulses will be cleared as “0”. The present number of output pulses is updated in every scan time. When M1133 turns from “Off” to “on”, the number will be cleared as “0”. When M1172 is cleared as “0” when PLC goes from STOP to RUN. When PLC goes from RUN to STOP, the last number of pulses will be shown.

3. Note:

This function can only be used when PLC is in RUN status and can coexist with PLSY instruction in the program. If PLSY instruction is executed prior to this function, the function cannot be used and vice versa.

Function Group **VR Volume**

No. **M1178, M1179, D1178, D1179**

1. Special D and special M for built-in 2-point VR volume for EH/EH2/SV/SA/SC:

No.	Function
M1178	Enable VR0 volume
M1179	Enable VR1 volume
D1178	VR0 value
D1179	VR1 value

2. This fumction should be used when PLC is in RUN status. When M1178 = On, VR0 value will be converted into a value of 0 ~ 255 and stored in D1178. When M1179 = On, VR1 value will be converted into a value of 0 ~ 255 and stored in D1179.

Function Group **Interruption Instruction for Reading the Number of Pulses**

No. **D1180, D1181, D1198, D1199**

1. SA/SX/SC can use external interruption to store the present value in the middle-high-speed counter into D1180 ~ D1181, D1198 ~ D1199.

2. The function:

a. For SA/SX, X0 (pulse input point) has to work with X4 (external interruption point), C235/C251/C253 (high- speed counter) and I401 (interruption No.). D1180 and D1181 are the

registers to store the 32-bit values. X1 (pulse input point) has to work with X5 (external interruption point), C236 and I501. D1198 and D1199 are the registers to store the 32-bit values.

b. For SC, X10 (pulse input point) has to work with X4 (external interruption point), C243/C255 (high-speed counter) and I401 (interruption No.). D1180 and D1181 are the registers to store the 32-bit values.

X11 (pulse input point) has to work with X5 (external interruption point), C245 and I501. D1198 and D1199 are the registers to store the 32-bit values.

Function Group **Latched Area**
N0. **D1200 ~ D1219**

The latched area for EH / EH2 / SV / SA / SX / SC is from the start address No. to the end address No.

Function Group **Stores Value of High-speed Counter when Interrupt Occurs**
N0. **D1240 ~ D1241, D1242~D1243**

---

1. If extertal interrupts are applied on input points for Reset, the interrupt instructions have the priority in using the input points. In addition, PLC will move the current data in the counters to the associated data registers below then reset the counters.

2. Function:

a. X0 (counter input) and X4 (external Interrupt) will correspondingly work together with C246, C248, C252, and I400/I401. Use D1240 and D1241 as a 32 bit register to set X0 and X4.

b. X0 (counter input) and X1 (external Interrupt) will correspondingly work together with C243, and I100/I101. Use D1240 and D1241 as a 32 bit register to set X0 and X1.

c. X2 (counter input) and X5 (external Interrupt) will correspondingly work together with C250, C254 and I500/I501. Use D1242 and D1243 as a 32 bit register to set X2 and X5.

Special D	D1241, D1240				D1243, D1242		
Counter	C243	C246	C248	C252	C244	C25	C25

Function Group **RTC**
No. **M1016, M1017, M1076, D1313 ~ D1319**

---

1. Special M and special D relevant to RTC

Devic	Na	Functi
M1016	Year Display	OFF: display the last 2 digits of year in A.D ON: display the last 2 digits of year in A.D. plus 2,000
M1017	± 30 seconds correction	When triggered from “Off” to “On”, the correction is enabled. 0 ~ 29 second: minute intact; second reset to 0 30~ 59 second: mimute + 1; second reset to 0
D1313	Second	0~59

D1314	Minute	0~59
D1315	Ho	0~23
D1316	D	1~31
D1317	Month	1~12
D1318	Week	1~7
D1319	Ye	0 ~ 99 (last 2 digits of Year in A.D.)

2. SA/EH2 series: If the set value in RTC is incorrect, the time will be recovered as “Saturday, 00:00 Jan. 1, 2000” when PLC is powered and restarted.

3. ES2/EX2 series: If set value for RTC is invalid. RTC will display the time as Second: 0, Minute: 0, Hour: 0, Day: 1, Month: 1, Week: 1, Year: 0.

4. ES2/EX2 series: Memory of RTC is latched. RTC will resume the time when power is down. For higher accuracy of RTC, please conduction calibration on RTC when power resumes.

5. Methods of modifying RTC:

- Apply TWR instruction to modify the built-in real time clock of DVP-ES2. Please refer to TWR for detail.
- Use peripheral devices or WPLSoft / ISPSoft to set the RTC value.

Function Group N0. **Right-Side Special Extension Module ID D1320 ~ D1327**

1. The ID of special extension module, if any, connected to EH/EH2/SV are stored in D1320 ~ D1327 in sequence.

2. Special extension module ID for EH:

Module Name	Module ID (hex)	Module Name	Module ID (hex)
DVP04AD-H	H’0400	DVP01PU-H	H’0110
DVP04DA-H	H’0401	DVP01HC-H	H’0120
DVP04PT-H	H’0402	DVP02HC-H	H’0220
DVP04TC-H	H’0403	DVP01DT-H	H’0130
DVP06XA-H	H’0604	DVP02DT-H	H’0230

3. Special extension module ID for EH2:

Module Name	Module ID (hex)	Module Name	Module ID (hex)
DVP04AD-H2	H’6400	DVP01PU-H2	H’6110
DVP04DA-H2	H’6401	DVP01HC-H2	H’6120
DVP04PT-H2	H’6402	DVP02HC-H2	H’6220
DVP04TC-H2	H’6403	DVP01DT-H2	H’6130
DVP06XA-H2	H’6604	DVP02DT-H2	H’6230

4. When I/O modules are connected, the ID of each I/O module will be stored in D1320~D1327 in connection order.

5. ID of each AIO module:

Module Name	Module ID (hex)	Module Name	Module ID (hex)
DVP04AD-E2	H'0080	DVP06XA-E2	H'00C4
DVP02DA-E2	H'0041	DVP04PT-E2	H'0082
DVP04DA-E2	H'0081	DVP04TC-E2	H'0083

Function Group **Easy PLC Link**

Number **M1350 ~ M1354, M1360 ~ M1519, D1355 ~ D1370, D1399, D1415 ~ D1465, D1480 ~ D1991**

---

1. Special D and special M for ID1 ~ ID8 of the 16 stations in EASY PLC LINK (M1353 = Off) for SA/SX/SC/EH/EH2/SV:

7. Explanation:

a. EASY PLC LINK is based on MODBUS communication protocol.

b. When Slave PLC is connected through COM1 / COM2 / COM3, baud rate and communication format of all Slaves must be the same (set in D1036). DVP-PLC supports both ASCII and RTU mode.

c. The ID number of the starting slave can be designated by D1399 and should be limited to the range K1~K214. Slave ID cannot be repeated or the same as Master ID (set in D1121/D1255)

(به علت طولانی بودن این بخش حذف گردید، برای جزئیات بیشتر در مورد حافظه‌های مورد استفاده در ارتباط از طریق شبکه **MODBUS** به راهنمای نرم افزار مراجعه نمایید.)

Function Group **Left-Side High-Speed Special Extenstion Module ID**

No. **D1386 ~ D1393**

---

1. The ID of left-side high-speed special extension module, if any, connected to SV are stored in D1386 ~ D1393 in sequence.

2. Left-side high-speed special extension module ID for SV:

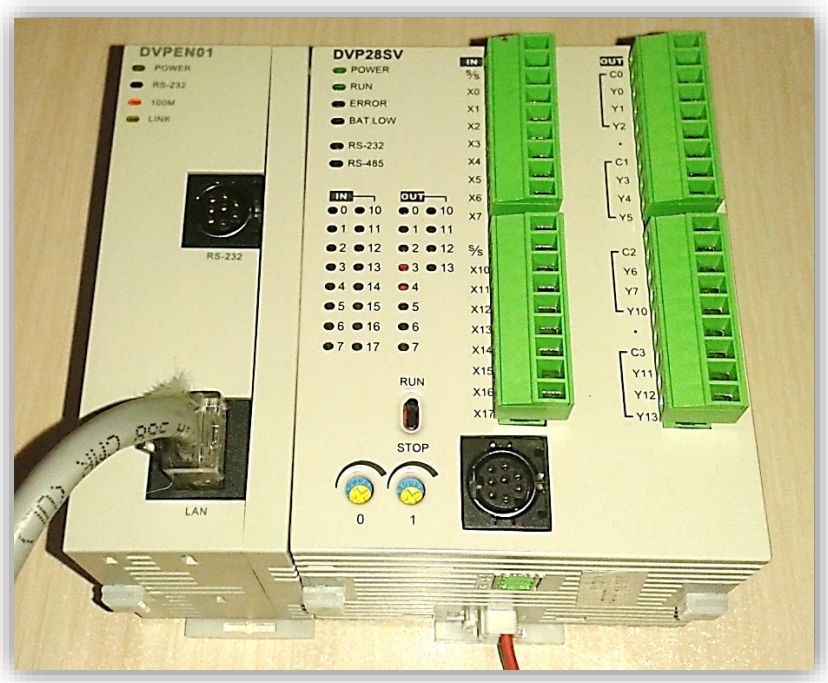
Module Name	Module ID (hex)	Module Name	Module ID (hex)
DVP04AD-SL	H'4400	DVP01HC-SL	H'4120
DVP04DA-SL	H'4401	DVP02HC-SL	H'4220
DVP04PT-SL	H'4402	DVPDNET-SL	H'4130
DVP04TC-SL	H'4403	DVPEN01-SL	H'4050
DVP06XA-SL	H'6404	DVPMDM-SL	H'4040
DVP01PU-SL	H'4110		

ضمیمه ه – پیکربندی ماژول DVPEN01

[*** این ضمیمه به عنوان نمونه‌ای جهت آشنایی بیشتر با نحوه پیکربندی یکی از ماژول‌های پرکاربرد کمپانی دلتا در شبکه‌های صنعتی آورده شده و در تشریح آن فقط بر روی ISPSoft تمرکز نشده است. در کتابی که در آینده در مورد شبکه های صنعتی محصولات کمپانی دلتا منتشر خواهد شده، جزئیات راه‌اندازی و به-کارگیری ماژول‌ها مختلف شبکه مورد بررسی قرار خواهد گرفت.]

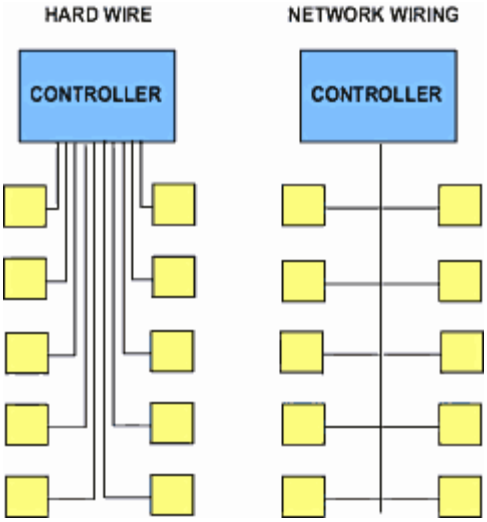
۱– مقدمه

DVPEN01 ماژول شبکه Ethernet محصولی از کمپانی دلتا برای استفاده در اتوماسیون صنعتی است. از این ماژول می‌توان برای ارتباط نرم افزارهای کمپانی دلتا مانند ISPSoft و WPLSoft با PLCهای سری DVP استفاده کرد. قابلیت ارسال ایمیل، تنظیم اتوماتیک زمان، تبادل اطلاعات، ارسال ایمیل و مانیتورینگ از طریق وب، اسکادا و HMI از دیگر امکاناتی است که این ماژول برای کاربر فراهم می‌کند. دارا بودن پورت RS-232 و همچنین تشخیص دهنده اتوماتیک MDI/MDI-X که کاربر را از معکوس کردن سیم بندی کابل‌های شبکه بی-نیاز می‌کند از دیگر جذابیت‌های این ماژول می‌باشد.

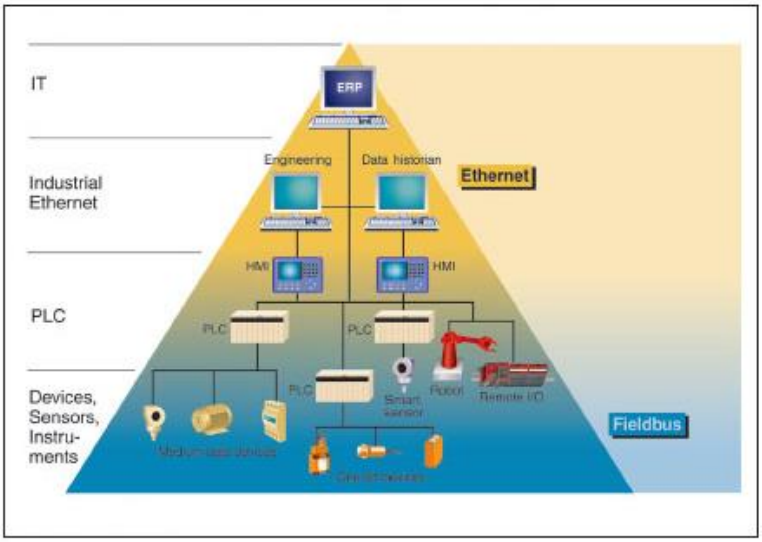


شبکه‌های صنعتی

دلیل اولیه و اصلی استفاده از شبکه‌های صنعتی کاهش پیچیدگی سیستم در اثر کاهش سیم-کشی می‌باشد که موجب صرفه‌جویی در هزینه و زمان می‌شود. این موضوع همچنین باعث می‌شود که بتوانیم راحت‌تر به عیب‌یابی سیستم بپردازیم، دستگاه‌های متصل به شبکه را از طریق رایانه تنظیم کنیم، داده‌ها را به راحتی جمع‌آوری کرده و با استفاده از اینترنت به صورت غیرحضوری به کنترل سیستم بپردازیم.



پروتکل‌های کوناگونی برای شبکه‌های صنعتی وجود دارد که هر یک کاربرد و ویژگی‌های خاصی دارند. تفاوت عمده این پروتکل‌های ارتباطی در قیمت، سرعت انتقال داده، نحوه انتقال داده، تعداد گره‌های شبکه و ... می‌باشد.

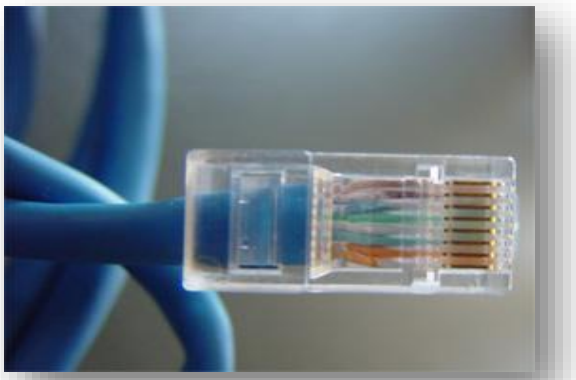


شبکه‌ی Ethernet

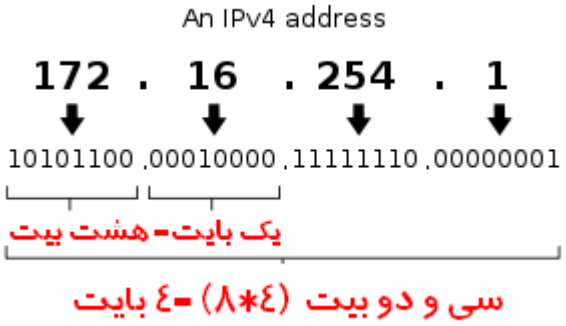
Ethernet پر استفاده ترین شبکه محلی (LAN[1]) مورد استفاده در سرتاسر دنیا است. توسعه شبکه‌های بی‌سیم بر اساس استانداردهای مرتبط با پروتکل Ethernet موجب گسترش روزافزون این شبکه محلی شده است. همچنین فراگیر بودن استفاده از این پروتکل موجب کاهش قیمت سخت افزارهای مرتبط با آن، آشنایی گسترده کاربران و گسترش هرچه بیشتر آن شده است. تجهیزات نصب شده بر روی شبکه Ethernet با ارسال بسته‌های داده از طریق کابل شبکه با یکدیگر ارتباط برقرار می‌کنند هر دستگاه (گره های درون شبکه Ethernet) دارای یک آدرس ٤٨ بیتی (MAC) می‌باشد که بصورت سخت‌افزاری در کارت شبکه (NIC) قرار دارد و کارت شبکه در هر گره (دستگاه یا کامپیوتر) صرفا داده‌هایی بر

[1] local area network

روی شبکه را دریافت می‌کند که دارای آدرس آن دستگاه باشد و بسته‌های اطلاعاتی که آدرس گره‌های دیگر را دارند قبول نمی‌کند.

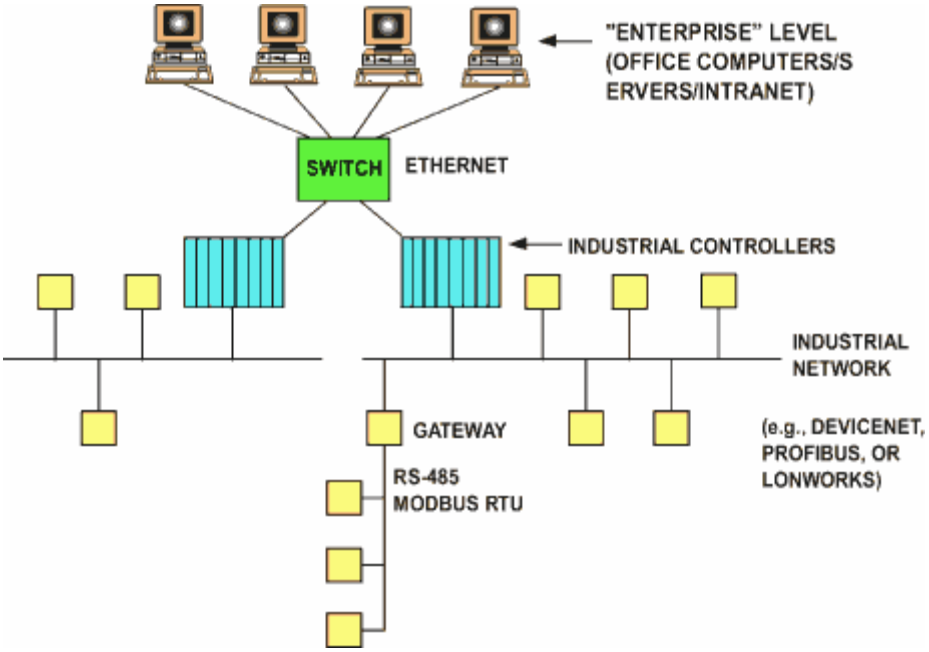


نشانی پروتکل اینترنت (Internet Protocol Address=IP Address) نشانی عددی است که به هریک از دستگاه ها و رایانه‌های متصل به شبکه‌ای که بر مبنای نمایه TCP/IP (از جمله اینترنت و Ethernet) کار می‌کند، اختصاص داده می‌شوند. پیام‌هایی که دیگر رایانه‌ها و دستگاه‌ها (گره‌ها) برای گره دیگری می‌فرستند با این نشان عددی همراه است و می‌توان نقش آن در شبکه را مانند «نشانی گیرنده» در نامه‌های پستی تعبیر کرد، تا در نهایت پیام به گره مورد نظر برسد. نشان IP شامل ٤ بخش ٨ بیتی است (معادل چهار عدد بین صفر الی ۲۵۵)



IP گره‌های شبکه را می‌توان به صورت اتوماتیک (دینامیک) و یا دستی (استاتیک) تعیین کرد. IP دینامیک با هر بار وصل‌شدن به Ethernet تغییر می‌کند. اما IP استاتیک به صورت ثابت بر روی دستگاه تنظیم می‌شود. IP دینامیک در هر شبکه توسط سرور DHCP به رایانه‌ها و

گره‌های شبکه اختصاص داده می‌شود. یعنی وقتی گره‌ای به Ethernet وصل می‌شود، سرور به صورت اتوماتیک به آن یک نشانی IP اختصاص می‌دهد و نیازی به تنظیمات دستی نیست. پروتکل Ethernet استفاده شده در محیط های صنعتی که به "Ethernet صنعتی" معروف هست از نظر نرم‌افزاری دقیقا مشابه پروتکل استاندارد Ethernet است با این تفاوت که سخت افزار آن باید مطابق با شرایط نامساعد صنعتی طراحی شود، شرایطی شامل نویزهای الکترومغناطیسی، دمای بالا، لرزش شدید، رطوبت و غیره. Ethernet در صنعت بیشتر در لایه‌های بالای معماری (لایه‌های نزدیک به کامپیوتر و کنترل کننده‌های مرکزی) مورد استفاده قرار می‌گیرد.



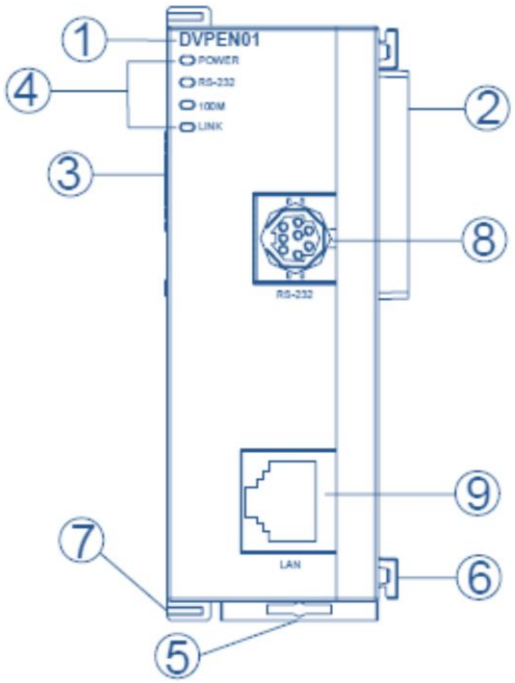
۲–ماژول DVPEN01

ویژگی‌های اصلی ماژول DVPEN01 به صورت زیر است:

- تشخیص اتوماتیک سرعت شبکه (۱۰ الی ۱۰۰ مگابیت بر ثانی)
- امکان استفاده از هر دو نوع اتصال مستقیم و معکوس پورت‌های شبکه RJ-45
- پشتیبانی از پروتکل Modbus TCP
- پشتیبانی از ارتباط سریال RS-232

- قابلیت ارسال ایمیل
- تصحیح زمان PLC از طریق اینترنت
- پشتیبانی از انتقال داده

مشخصات فیزیکی:



۱- نام مدل

۲- پورت برای اتصال به دستگاه‌های دیگر (CPU)

۳- پورت برای اتصال ماژول‌های ورودی/خروجی

٤- نمایشگرهای اتصال تغذیه، RS-232، تشخیص دهنده سرعت 100M، اتصال شبکه

۵- پین اتصال به ریل

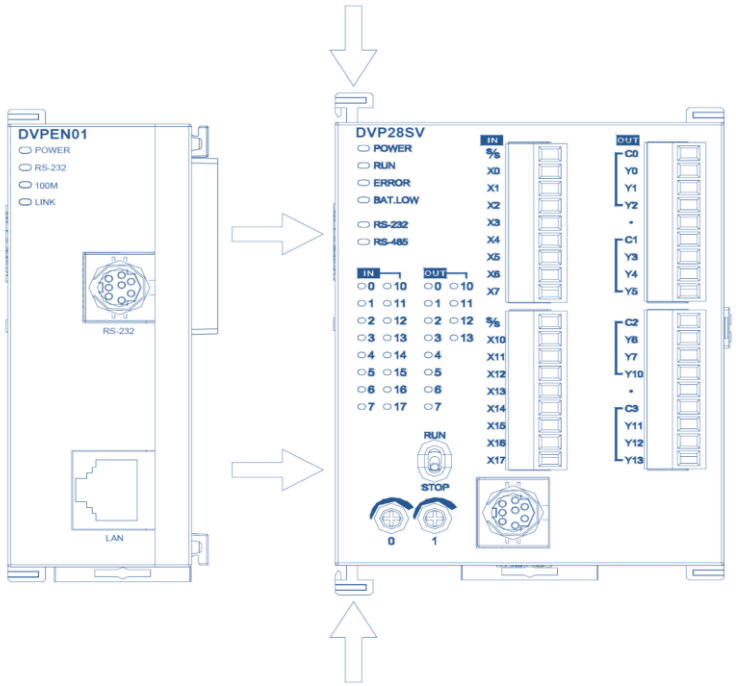
۶- پین اتصال ماژول

۷- پین اتصال ماژول

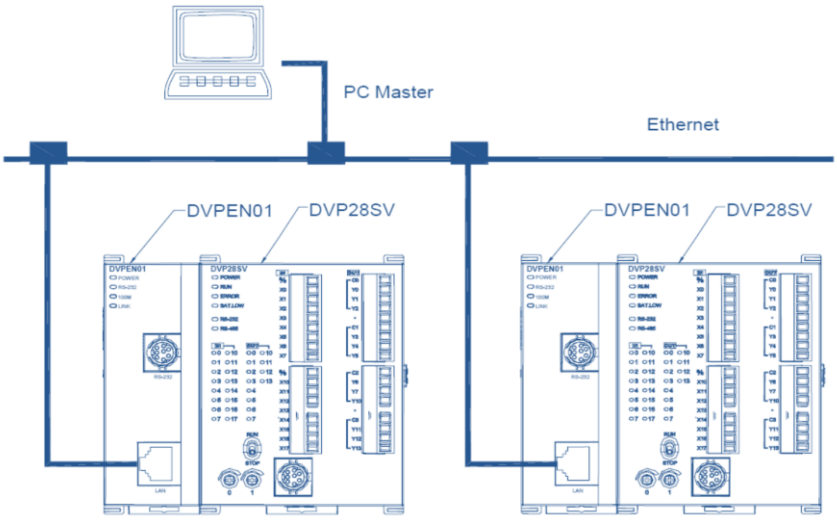
۸- پورت RS-232

۹- پورت RJ-45 برای Ethernet

برای اتصال ماژول فوق به دیگر ماژول‌ها مانند RJ-45 مشابه شکل زیر عمل می‌کنیم:

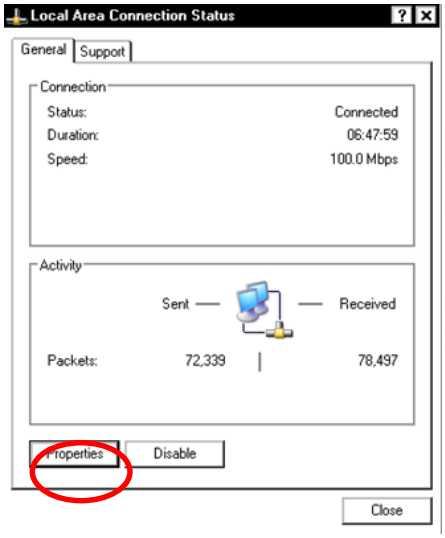


نحوه اتصال DVPEN01 در شبکه Ethernet:

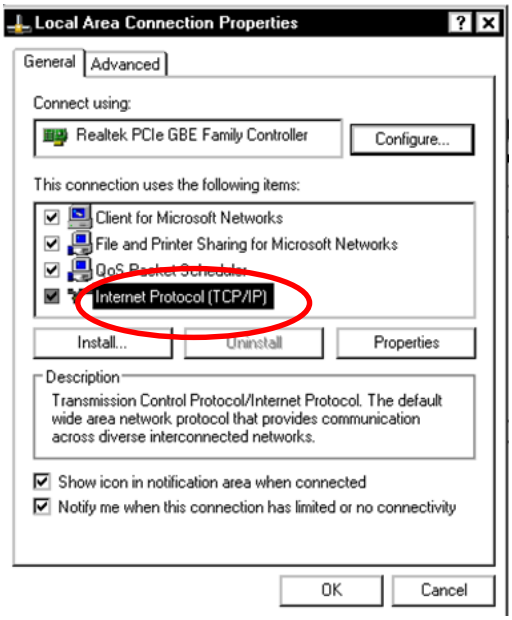


۳– تنظیم IP کامپیوتر به صورت Static

می‌بایست ابتدا IP کامپیوتری که با ISPSoft یا WPLSoft در ارتباط است را به صورت استاتیک[1] تعریف کنیم[2]. برای اینکار بر روی آیکون شبکه بر روی Taskbar یعنی  دوبار کلیک می –کنیم تا صفحه Local Area Connection Status باز شود. گزینه Properties را در این صفحه انتخاب می‌کنیم.



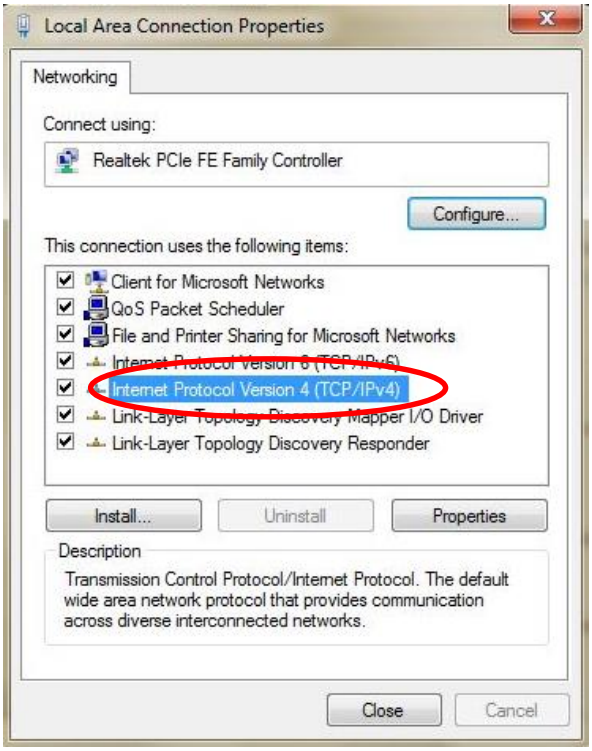
در صفحه‌ی باز شده بر روی Internet Protocol (TCP/IP) دوبار کلیک می‌کنیم.



[1] Static

۲ البته می‌توان IP دینامیک نیز تعریف کرد، ولی در این قسمت IP را استاتیک در نظر می‌گیریم. برای تنظیم IP به صورت DCHP به بخش "تنظیم IPماژول DVPEN01 به صورت DHCP" مراجعه کنید.

(توجه شود که در ویندوز Vista و 7 باید گزینه Internet Protocol Version 4 (TCP/IPV4) انتخاب شود)

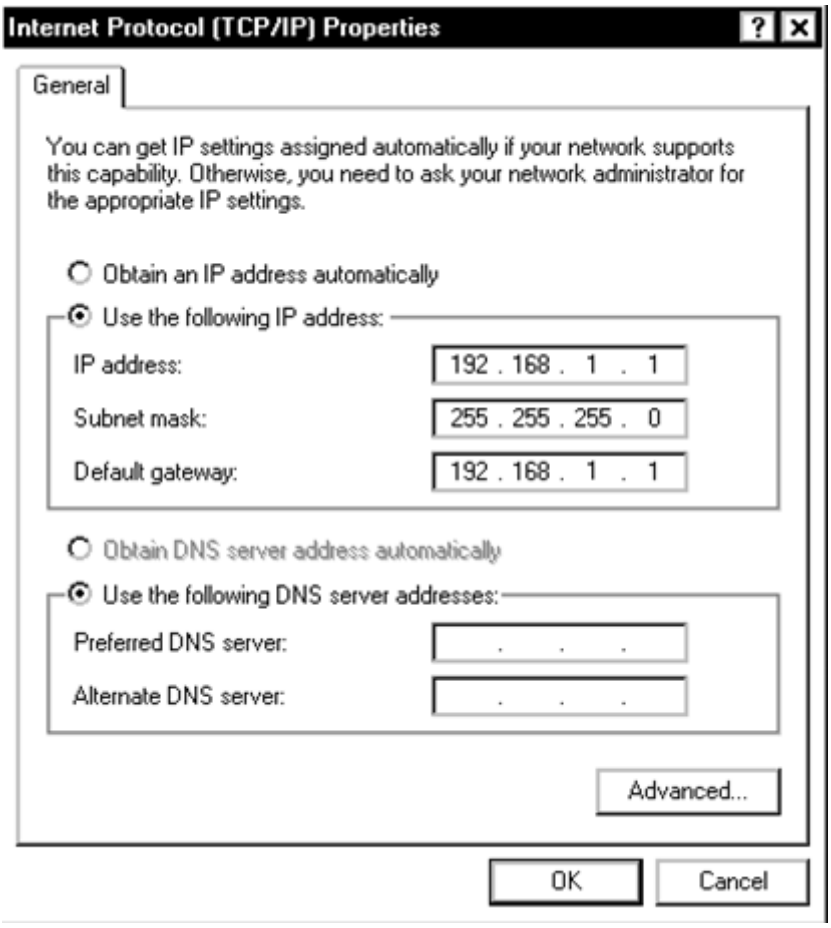


در صفحه باز شده IP و Subnet و Gateway را مشابه پنجره زیر تنظیم کنید. همچنین توجه به نکات زیر ضروری است:

۱- توجه کنید که Subnet باید به صورت 255.255.255.0 تنظیم شود.

۲- IP و Gateway نیز باید به صورت 192.168.x.y تعریف شود، **در این حالت x و y نباید برابر 0 یا 255 باشند**. اکثر مشکلاتی که کاربران در ارتباط با پیکربندی این ماژول دارند عدم تنظیم مناسب IP کامپیوتر می‌باشد. به عنوان مثال تنظیم IP به صورت 192.168.0.4 نادرست ولی تنظیم آن به صورت 192.168.1.4 مناسب می-باشد. (این بخش با توجه به تجربیات اعضای بخش فنی شرکت کامیاب مرام گفته می‌شود و در راهنمای رسمی ماژول به صورتی دیگر مطرح شده است. می‌توانید برای اطمینان بیشتر آن را مورد بررسی قرار دهید)

۳- IP کامپیوتر مشابه هیچکدام از دستگاه‌های درون شبکه Ethernet نباشد، در غیر این صورت تداخل رخ می‌دهد و عملکرد شبکه مختل خواهد شد.



در انتها نیز با کلیک بر روی OK در هر دو صفحه‌ی باز شده، تنظیم IP کامپیوتر را نهایی می‌کنیم.

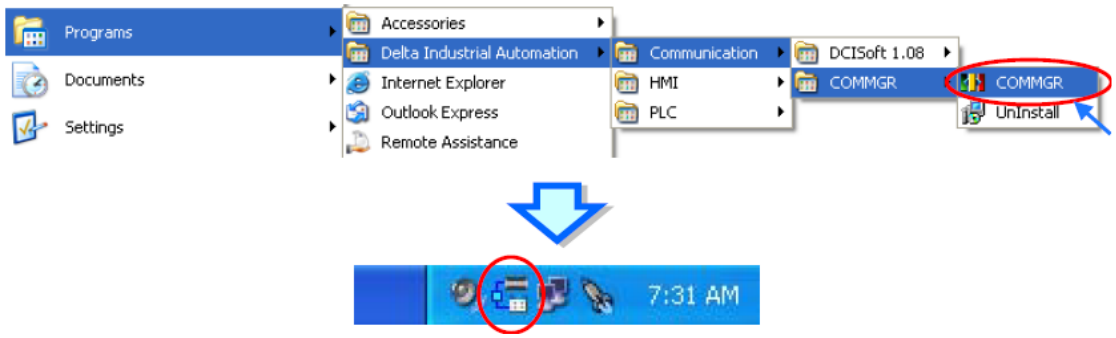
۴– راه اندازی ماژول DVPEN01 با استفاده از نرم افزار ISPSoft

ارتباط بین ISPSoft و PLC-های کمپانی دلتا به صورت بلوک دیاگرام زیر از طریق نرم افزار COMMGR برقرار می‌شود[۱].

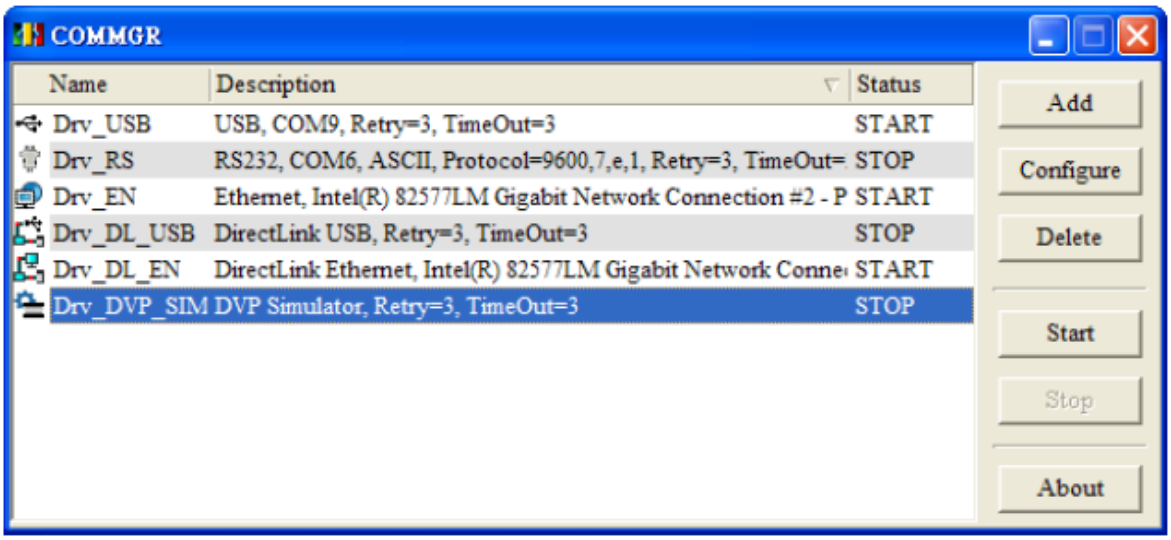
پس از نصب نرم افزار COMMGR آیکون آن () در نوار Taskbar ویندوز ظاهر می-شود:  . همچنین پس از هربار شروع مجدد ویندوز، این نرم افزار در ابتدای بالا آمدن ویندوز خود به خود فعال می‌شود. در صورتی هم که به هر دلیلی این نرم افزار غیرفعال باشد و بخواهیم آن را فعال کنیم، در منوی Start، در قسمت نرم افزارها[۲] طبق مسیر زیر می‌توانیم COMMGR را فعال کنیم. پس از اجرای COMMGR شما می‌توانید با دوبار کلیک کردن بر روی آیکون آن پنجره COMMGR را فعال نمایید.

۱ توجه شود که COMMGR تنها برای نرم افزار ISPSoft نسخه ۲ به بالا کاربرد دارد.

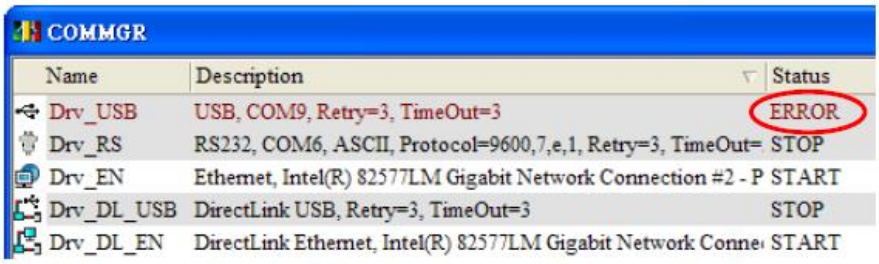
۲ Programs



پنجره COMMGR مطابق شکل زیر باز خواهد شد که با گزینه‌های سمت راست می‌توان درایورهای آن را مدیریت کرد. درایورهای لیست شده در COMMGR ارتباط نرم افزار و پورت های ارتباطی کامپیوتر را برقرار می‌کند. (درایور در واقع یک سری دستورالعمل است که کامپیوتر از انها پیروی می کند تا اطلاعات را برای انتقال به دستگاه جانبی خاص یا بازیابی از ان دوباره قالب بندی کند و بدین وسیله امکان ارتباط نرم افزار و سخت افزار را برقرار می‌کند)

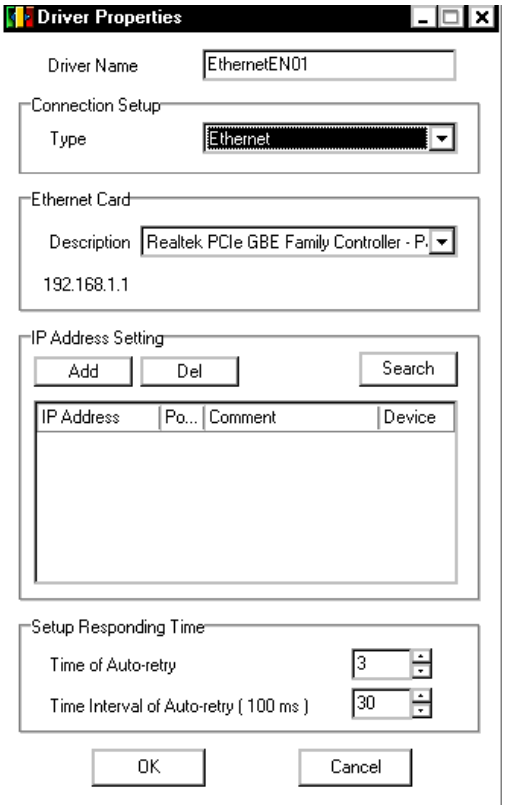


در صورتی که ارتباط نرم افزار با پورت‌های مشخص شده به وسیله COMMGR برقرار باشد، در ستون Status وضعیت ارتباط به صورت START مشخص خواهد شد، با این حال اگر COMMGR نتواند به هر دلیلی با پورت مورد نظر ارتباط برقرار کند (این حالت ممکن است به علت اشغال بودن پورت در اثر استفاده نرم افزاری دیگر باشد)، درایور متوقف خواهد شد و وضعیت ERROR به نمایش خواهد آمد.

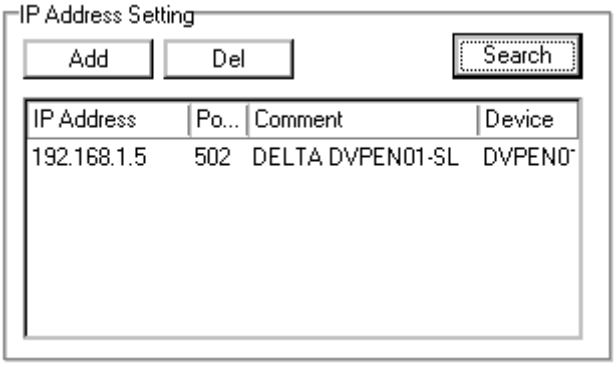


ساخت درایور در COMMGR برای شبکه Ethernet

ابتدا بر روی Add در COMMGR کلیک کنیم تا صفحه Driver Properties باز شود. در این قسمت می‌توانیم اسم درایور را به صورت دلخواه در قسمت Driver Name تعیین نماییم. توجه شود که از این نام در آینده برای ارتباط نرم افزارهای کمپانی دلتا با پورت مورد نظر باید استفاده شود به همین سبب پیشنهاد می‌شود مکانیزم مشخصی برای انتخاب نام درایور شامل در نظر گرفتن نام پورت و ویژگی‌های آن لحاظ تا از سردرگمی جلوگیری شود. نوع پورت (پروتکل) ارتباطی مورد نظر خود را نیز می‌توانیم در قسمت Connection Setup تنظیم نماییم. در اینجا پورت مورد نیاز Ethernet است.



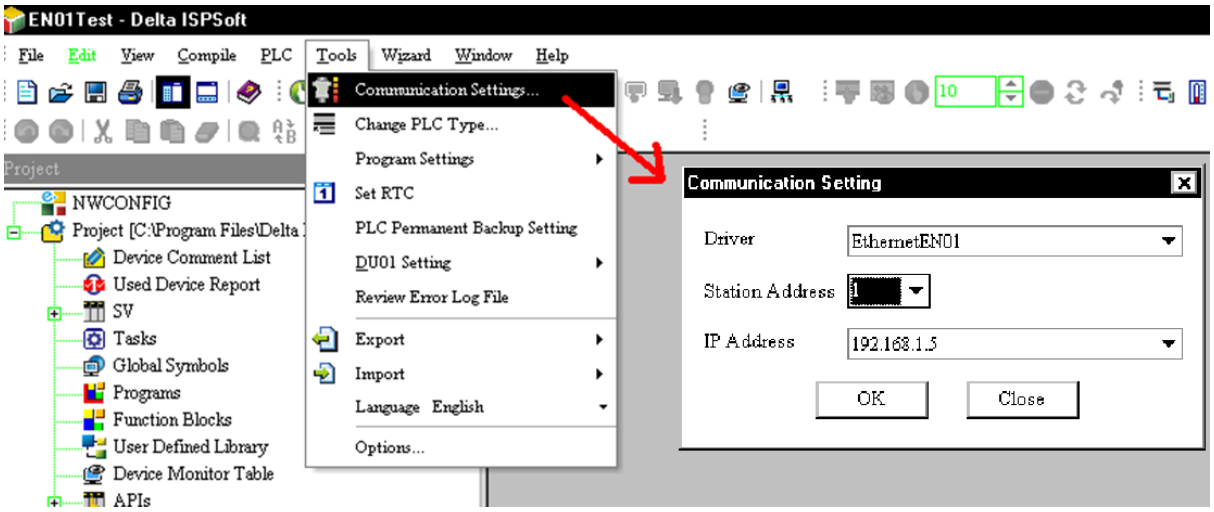
در قسمت Ethernet Card سخت افزار کارت شبکه کامپیوتر لیست شده است و IP آن نیز در سمت چپ آن (در شکل بالا ۱۹۲,۱۶۸,۱,۱) مشخص شده است. در قسمت بعدی یعنی IP Address Setting باید IP ماژول‌هایی که از طریق Ethernet می‌خواهیم با آن‌ها در ارتباط باشیم را بیاوریم. البته در صورتی که ماژول‌ها متصل به شبکه باشند، می‌توانیم با استفاده کلیک بر روی Search به صورت اتوماتیک به لیست IP آن‌ها دسترسی و آن‌ها را به درایور اضافه کنیم. در شکل زیر می‌بینیم که پس از Search کردن، نرم افزار DVPEN01 را تشخیص و IP آن را به درایور اضافه کرده است.



در قسمت Setup Responding Time نیز کاربر می‌تواند حداکثر تعداد تلاش برای ارتباط و حداکثر مدت زمان انتظار برای اینکار را به ترتیب در دو قسمت Time of Auto-retry و Time Interval of Auto-retry مشخص نماید. پس از تایید و ساخت درایور می‌توان از طریق گزینه‌های سمت راست COMMGR آن ها را فعال و یا غیرفعال کرد و یا اینکه توسط گزینه Configure آن‌ها را دوباره تنظیم کرد.

تنظیم ارتباط بین نرم افزار ISPSoft و COMMGR

پس از تنظیم داریورها در COMMGR کاربر می‌تواند در ISPSoft نیز درایور را در هر پروژه برای ارتباط با PLC تعیین نماید. برای اینکار اگر پروژه گروهی است ابتدا باید پروژه داخلی مورد نظر خود را فعال نمایید. سپس در سربرگ Tools گزینه Communication Settings را انتخاب نمایید.



در پنجره جدید در قسمت Driver، باید نوع درایور را مشخص کرد، همچنین Station Address متناظر با PLC که با PC در ارتباط است باید تعیین شود. اگر کاربر Station Address را نمی داند می‌تواند به جای آن صفر را انتخاب نماید. اگر نوع ارتباط درایور به صورت Ethernet باشد، آنگاه کاربر باید IP تنظیم شده در COMMGR مربوط به دستگاهی که می‌خواهد با آن ارتباط برقرار کند را انتخاب نماید. (توجه شود که با آنکه در راهنمای رسمی کمپانی دلتا گفته شده در صورتی که از Station Address اطلاعاتی ندارید آن را برابر صفر قرار دهید ولی مشاهده شده که گاهی این موضوع باعث ایجاد مشکلاتی در ارتباط نرم افزار با شبکه شده است – گروه فنی کامیاب مرام)

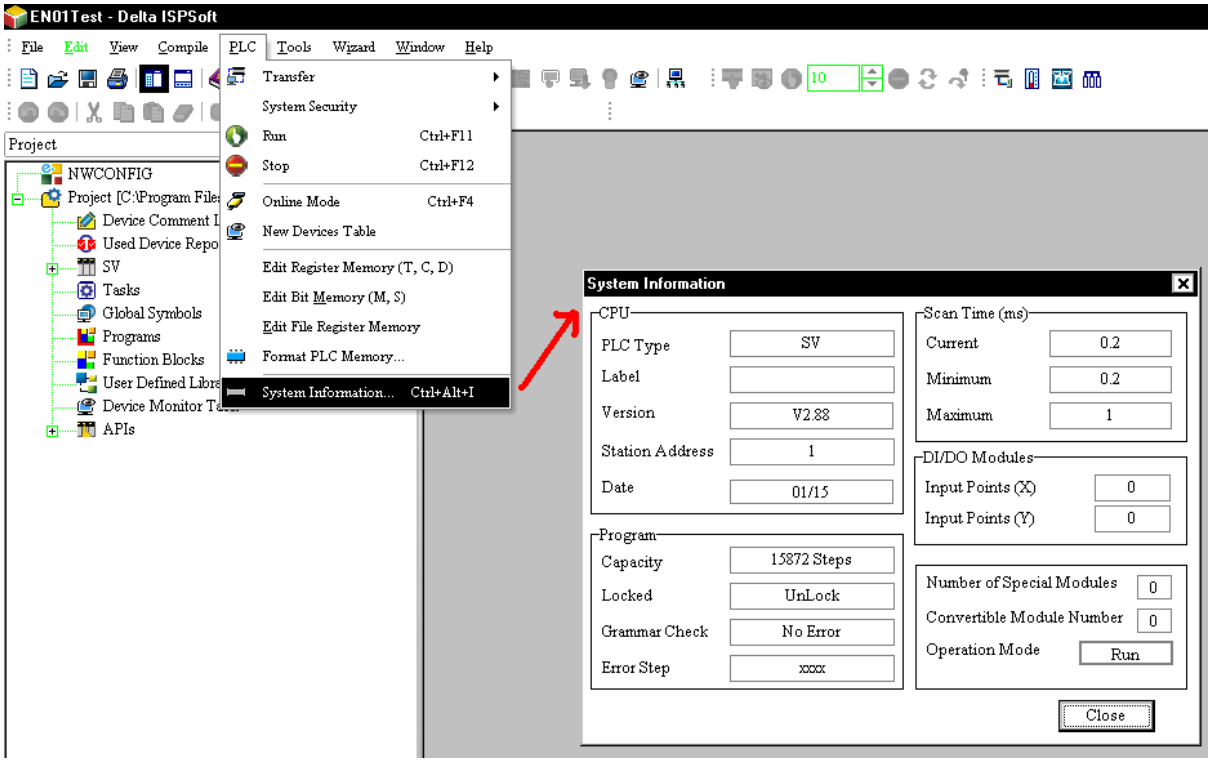
پس از اتمام تنظیمات، اطلاعات در مورد درایور متصل شده در نوار وضعیت نمایش داده می شود.

پس از طی شدن مراحل فوق کاربران برای اطمینان از اتصال کامپیوتر به PLC می‌توانند از تستی ساده استفاده کنند. در ابتدا موارد زیر را بررسی کنید:

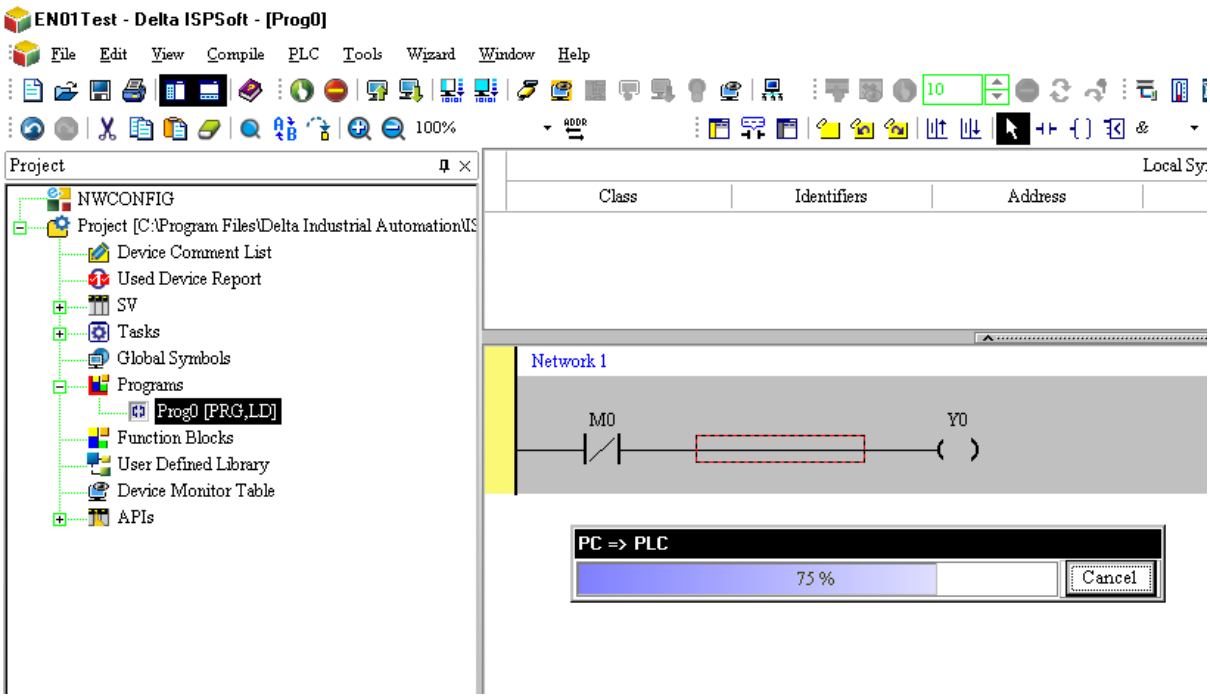
- وضعیت درایور در حالت Start باشد. و کابل شبکه RJ-45 هم به ماژول DVPEN01 و هم کامپیوتر متصل باشد.

- وضعیت کانال ارتباطی شامل کارت شبکه کامپیوتر، Hub و پورت سریال عادی باشد.
- درایور، آدرس Station و IP در Communication Setting درست تنظیم شده باشد.
- PLC به صورت درستی به DVPEN01 متصل باشد، تغذیه PLC متصل و وضعیت آن عادی باشد.

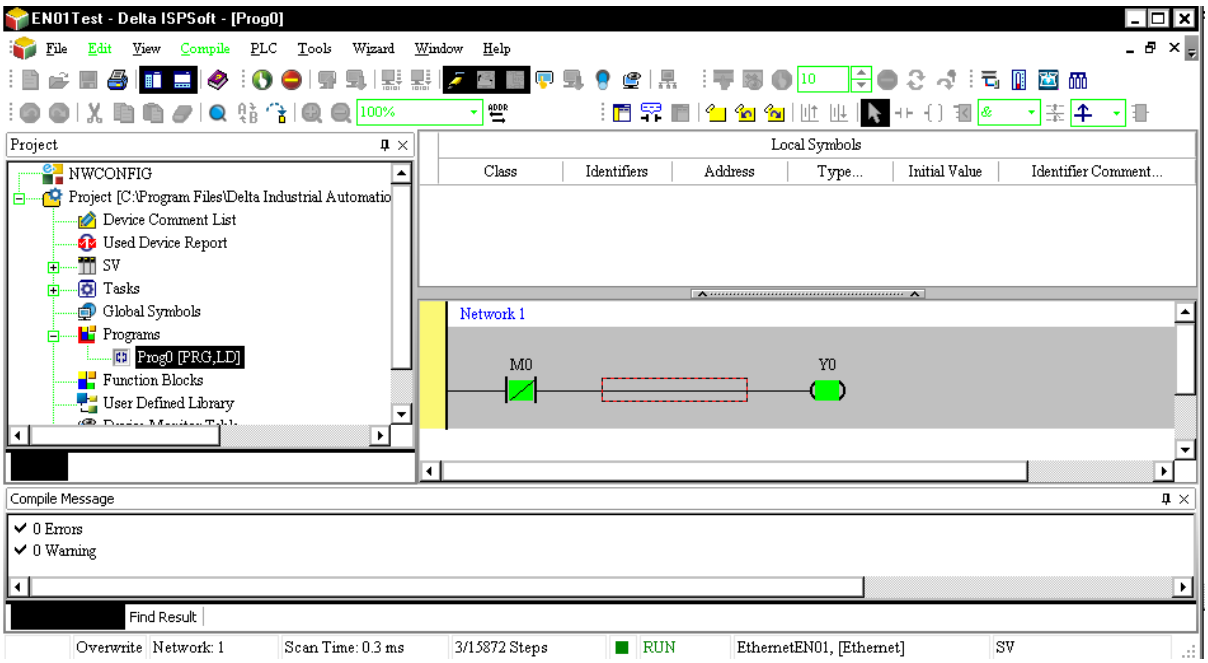
حال می توان در سربرگ PLC گزینه System Information را انتخاب کرد، **اگر ارتباط PLC با کامپیوتر به صورت نرمال برقرار شود ، آنگاه صفحه System Information ظاهر شده و اطلاعات رسیده از PLC نمایش داده می‌شود.**



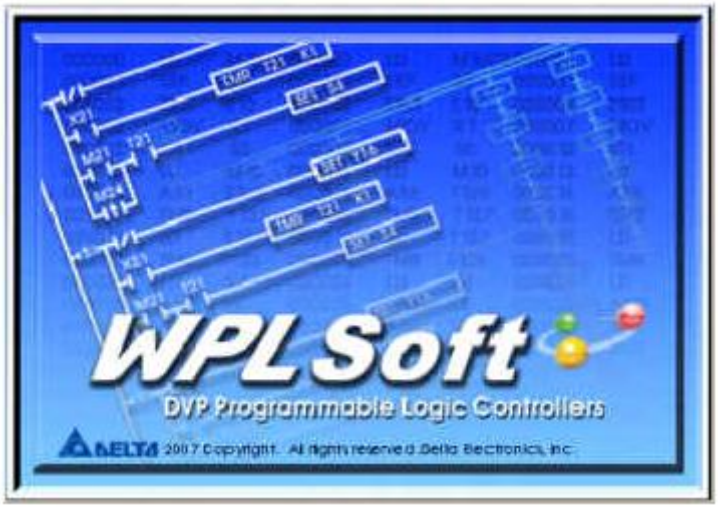
حال با خیال راحت می‌توانیم با استفاده از شبکه Ethernet برنامه را بر روی PLC بارگزاری و یا مانیتور کرد، همچنین در NWCONFIG می‌توان به تنظیم شبکه پرداخت. (برای اطلاعات بیشتر در مورد کار با نرم افزار ISPSoft به راهنمای آن- کاری از شرکت کامیاب مرام- مراجعه کنید.) پس از برقراری ارتباط در زیر بارگزاری برنامه در PLC از طریق Ethernrt را مشاهده می کنید:



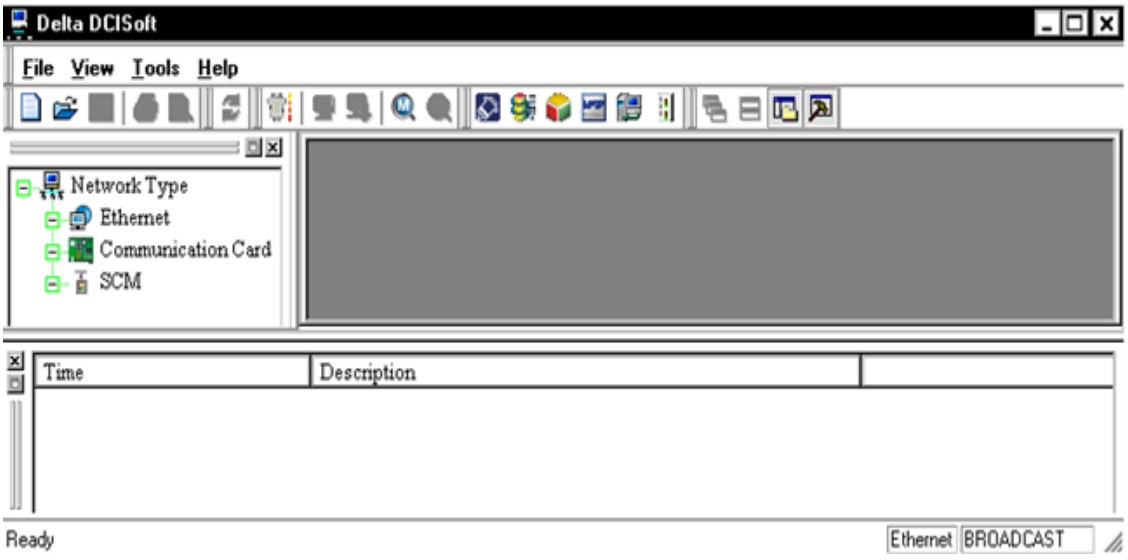
همچنین مانیتورینگ آنلاین با استفاده از Ethernet در ISPSoft:



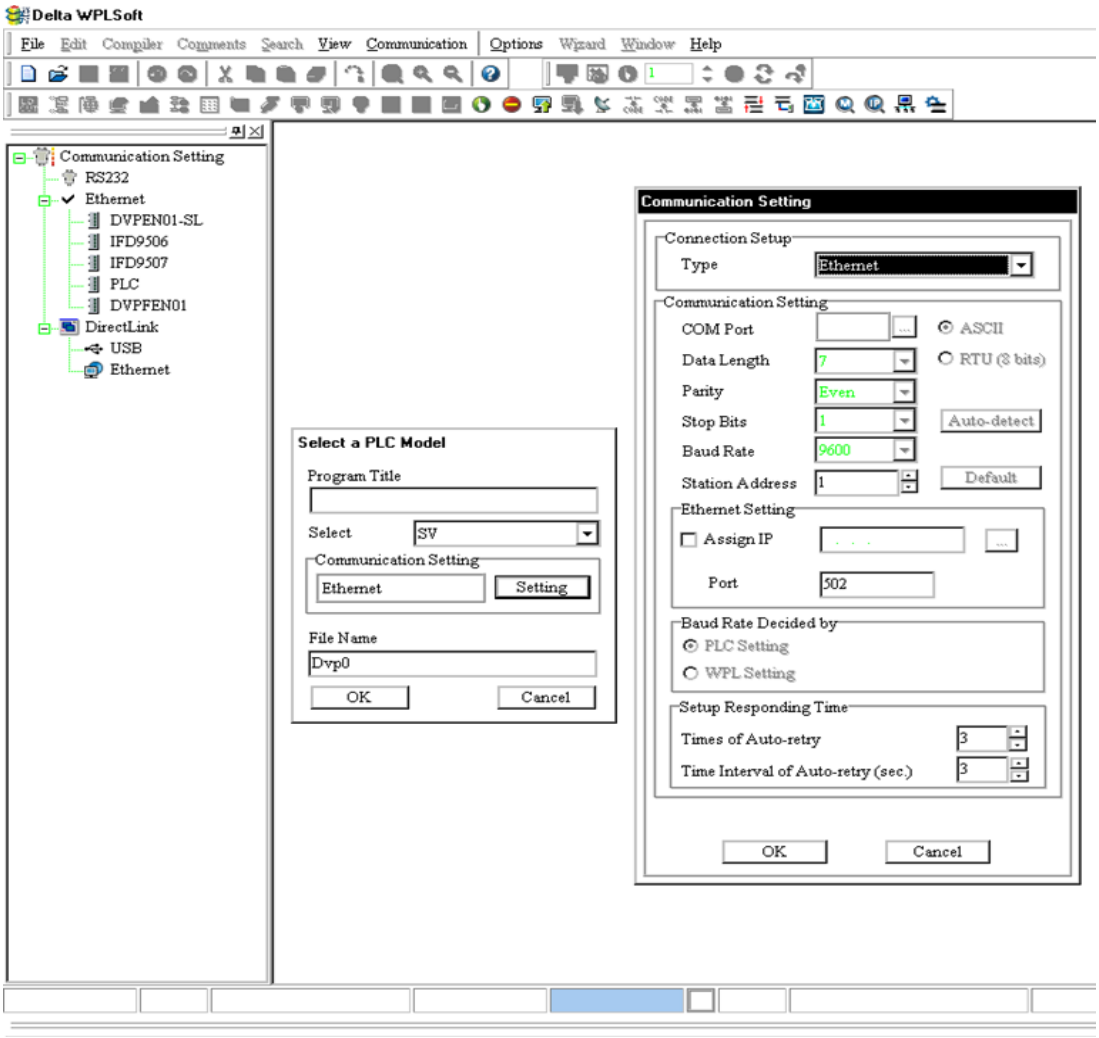
۵– راه اندازی ماژول DVPEN01 با استفاده از نرم افزار WPLSoft



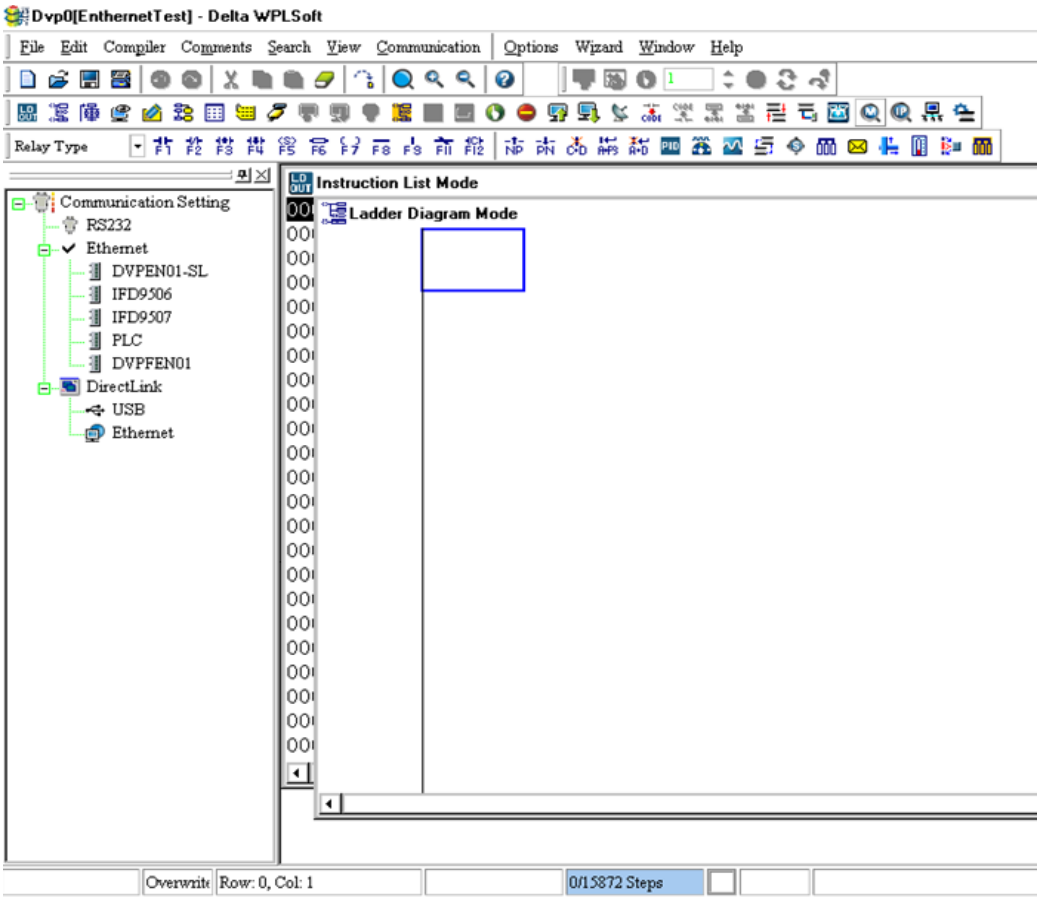
برای اینکه بتوانیم از طریق WPLSoft با ماژول DVPEN01 ارتباط برقرار کنیم. همزمان با این نرم افزار باید نرم افزار DCISoft را نیز اجرا کنیم. (باز نبودن همزمان این نرم افزار دلیل بسیاری از خطاها و هشدارها مانند "SCMSoft Parameter Error!" در WPLSoft می‌باشد)



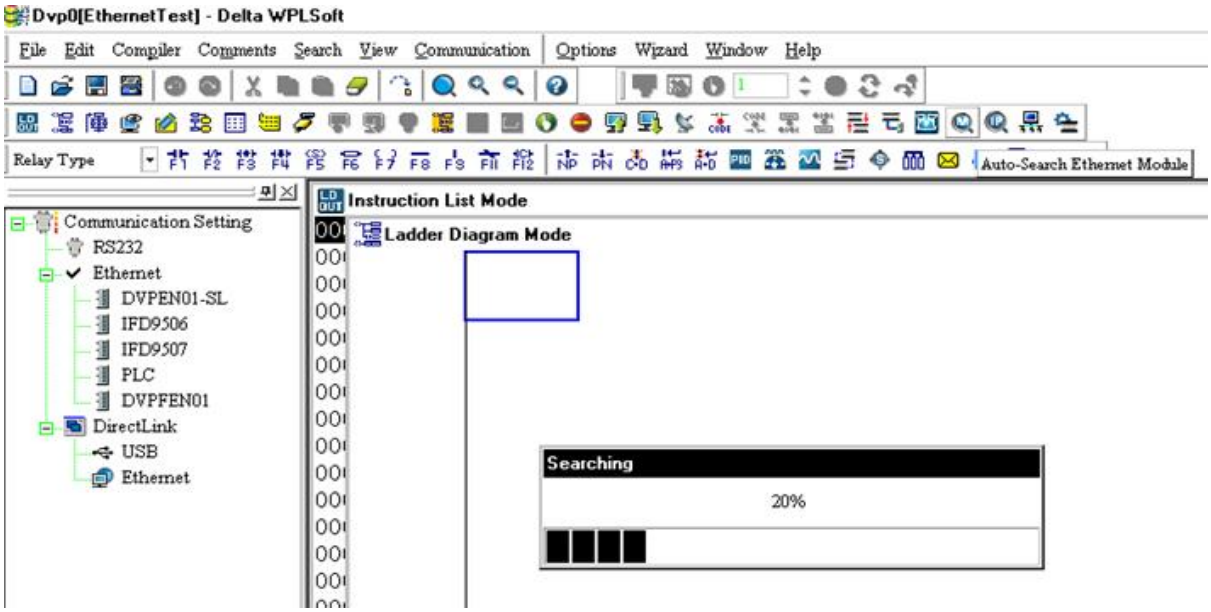
حال زمانی که می‌خواهیم پروژه جدیدی در WPLSoft تعریف کنیم، در قسمت Communication Setting نوع یا Type ارتباط را به صورت Ethernet انتخاب میکنیم. (البته تنظیم نوع ارتباط بعد از ساخت پروژه نیز امکان پذیر است، در ادامه این مورد شرح داده خواهد شد)



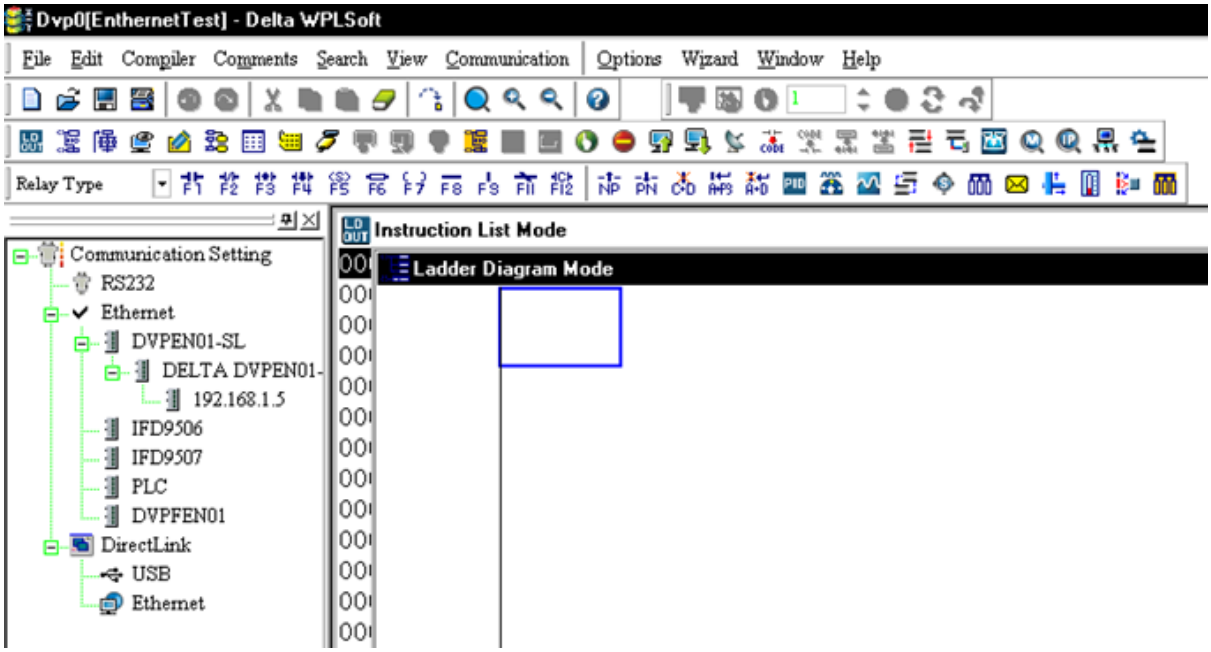
پس از ایجاد پروژه ، پنجره آن مانند شکل زیر خواهد شد که در سمت چپ آن بخش تنظیمات ارتباطی قرار دارد. مشخص است هنوز هیچ روشی برای ارتباط از طریق Ethernet تنظیم نشده است چرا که IP دستگاهی که نرم افزار از طریق آن بتواند ارتباط داشته باشد، تعیین نشده است.



برای اینکه سیستم بتواند ماژول‌های متصل به Ethernet را به صورت اتوماتیک جستجو و شناسایی نماید، بر روی آیکون (Auto-Search Ethernet Module) کلیک می‌کنیم (اگر این آیکون غیر فعال است- به صورت - ابتدا بر روی Ethernet در سمت چپ صفحه کلیک کنید. همچنین توجه کنید در این مرحله DCISoft باید باز باشد)

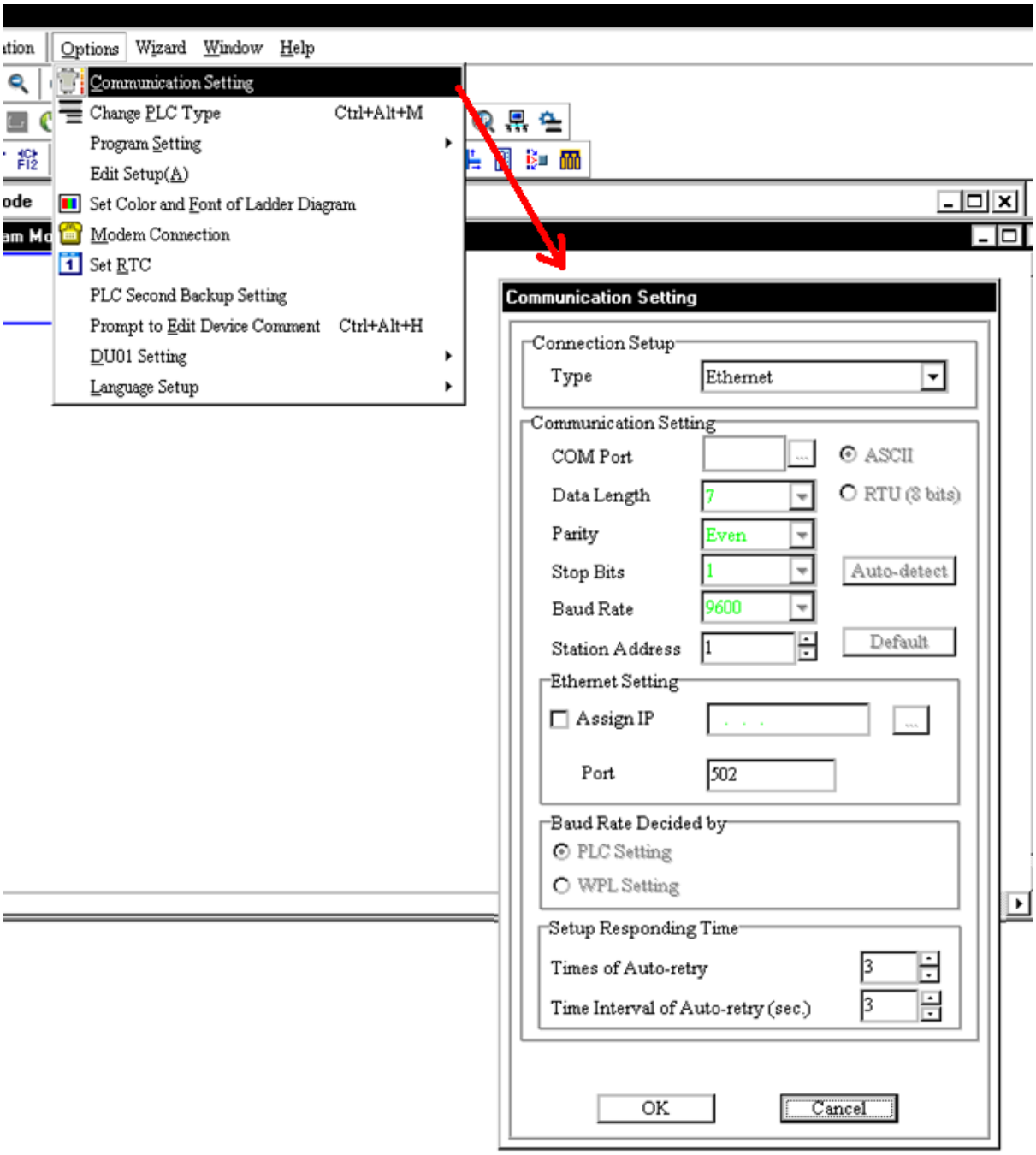


در انتهای جستجو ، IP ماژول‌های یافت شده در بخش Communication Setting لیست می‌شوند.



حال اگر در ابتدای ساخت پروژه تنظیمات شبکه Ethernet را انجام نداده اید، می‌توانید با انتخاب Communication Setting از منوی Option نوع (Type) شبکه را به صورت Ethernet انتخاب کنید. دیگر بخش ها را به صورت پیش فرض قرار دهید و بر روی OK کلیک کنید.

(توجه شود که با آنکه در راهنمای رسمی کمپانی دلتا گفته شده در صورتی که از Station Address اطلاعاتی ندارید آن را برابر صفر قرار دهید ولی مشاهده شده که گاهی این موضوع باعث ایجاد مشکلاتی در ارتباط نرم افزار با شبکه شده است)

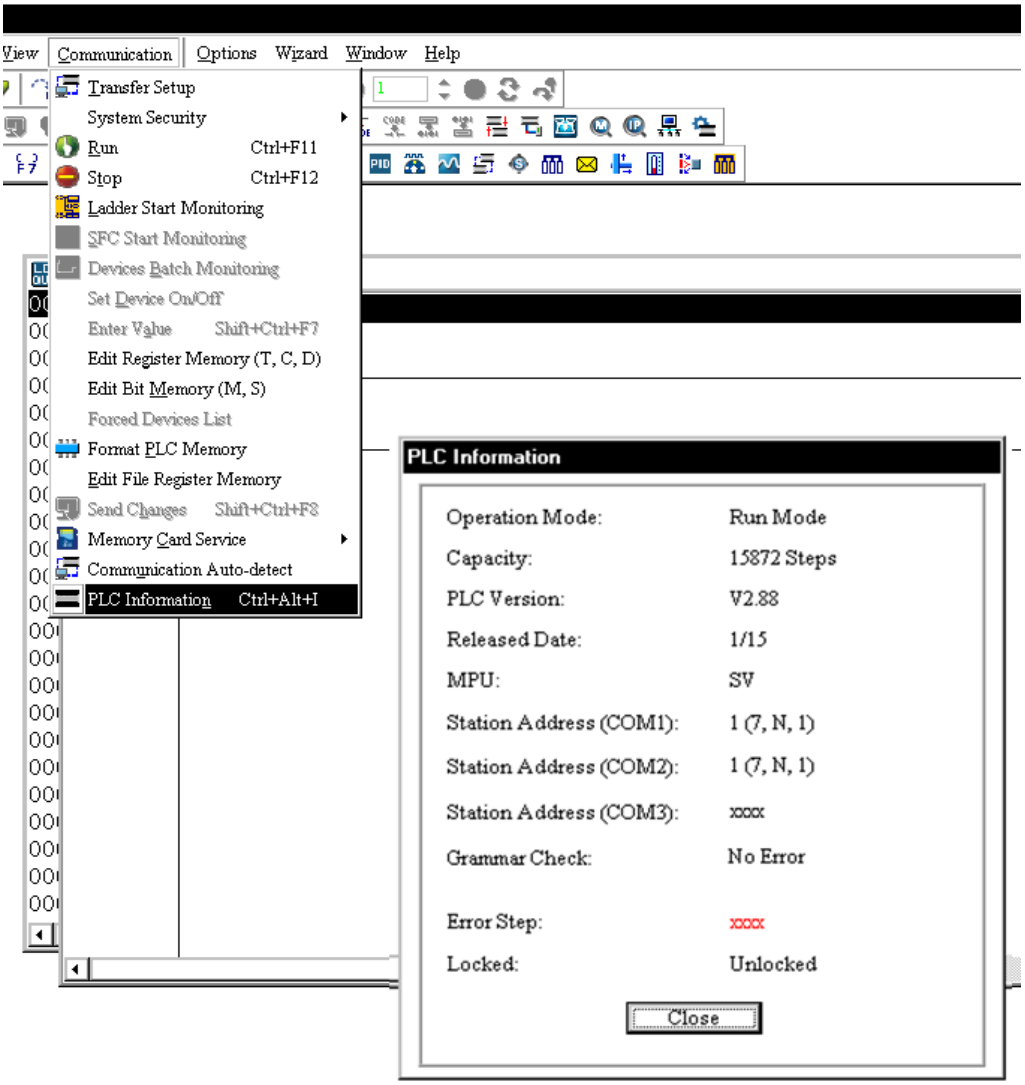


پس از طی شدن مراحل فوق کاربران برای اطمینان از اتصال کامپیوتر به PLC می‌توانند از تستی ساده استفاده کنند. در ابتدا موارد زیر را بررسی کنید:

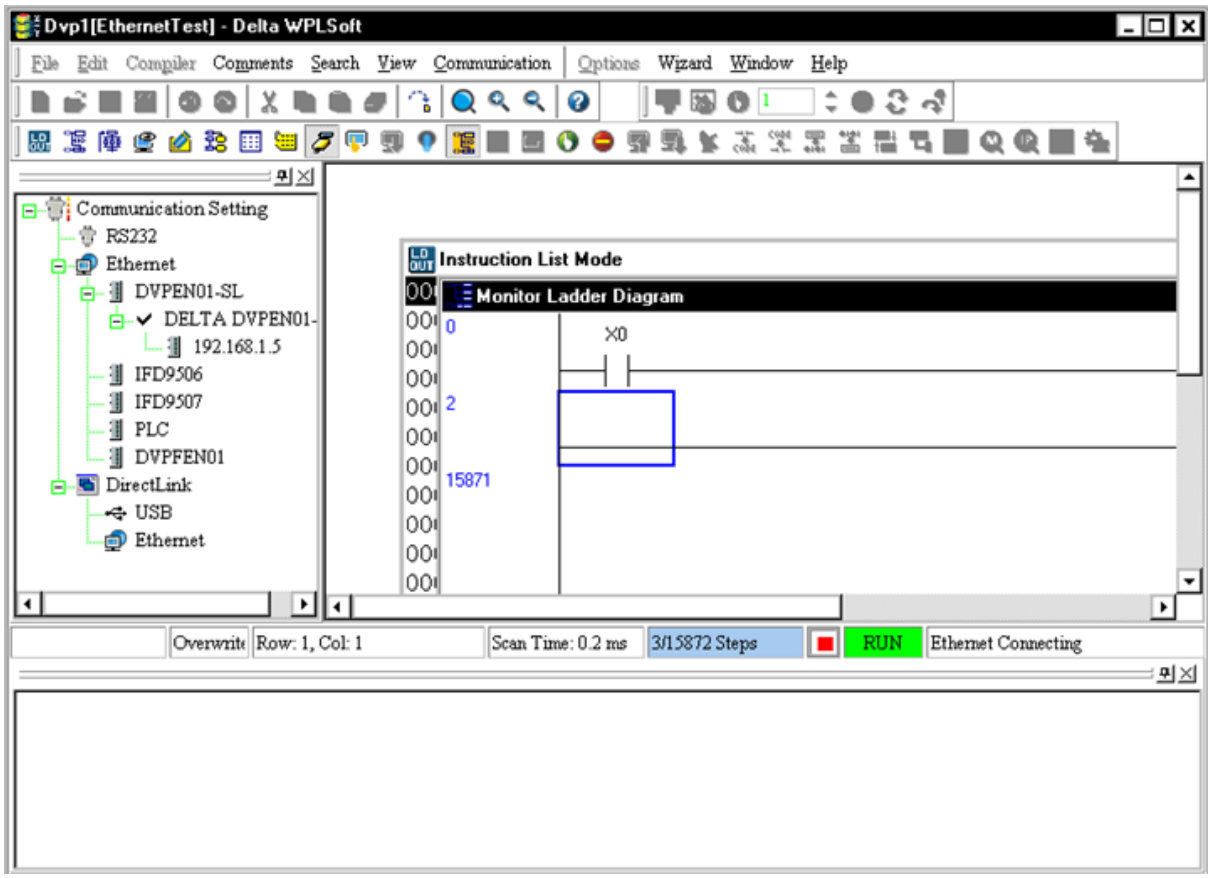
- وضعیت درایور در حالت Start باشد. و کابل شبکه RJ-45 هم به ماژول DVPEN01 و هم کامپیوتر متصل باشد.

- وضعیت کانال ارتباطی شامل کارت شبکه کامپیوتر، Hub و پورت سریال عادی باشد.
- درایور، آدرس Station و IP در Communication Setting درست تنظیم شده باشد.
- PLC به درستی به DVPEN01 متصل باشد، تغذیه PLC متصل و وضعیت آن عادی باشد.

حال می توان در سربرگ Communication گزینه System Information را انتخاب کرد، **اگر ارتباط PLC با کامپیوتر به صورت نرمال برقرار شود ، آنگاه صفحه System Information ظاهر شده و اطلاعات رسیده از PLC نمایش داده می‌شود.**

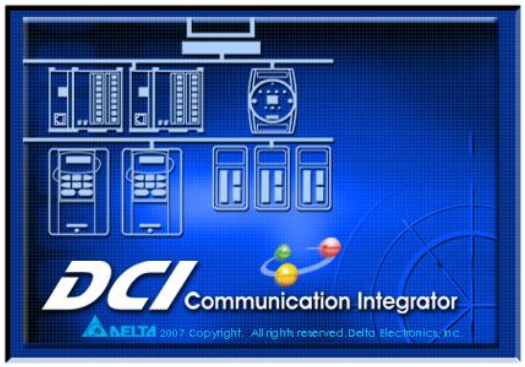


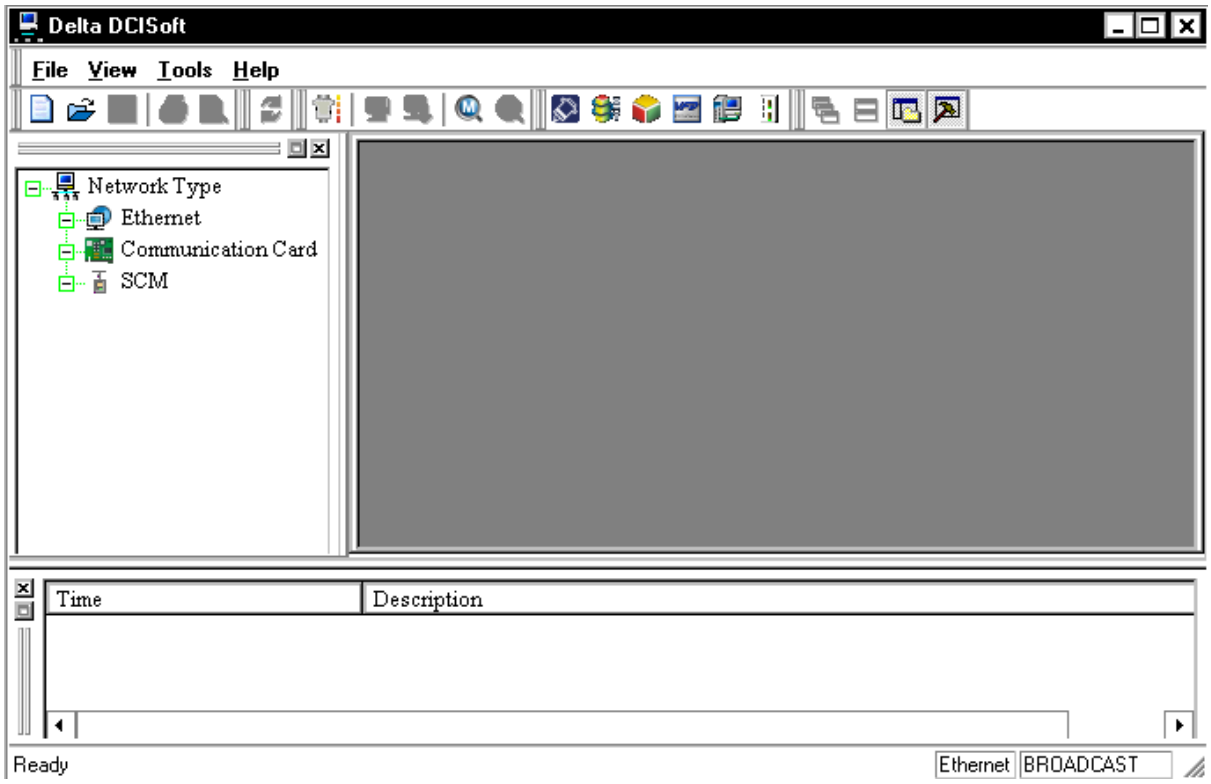
حال با خیال راحت می‌توانیم با استفاده از Ethernet برنامه را بر روی PLC بارگزاری و یا مانیتور کرد. در زیر بارگزاری برنامه در PLC و مانیتورینگ آنلاین آن از طریق Ethernet را مشاهده می‌کنید:



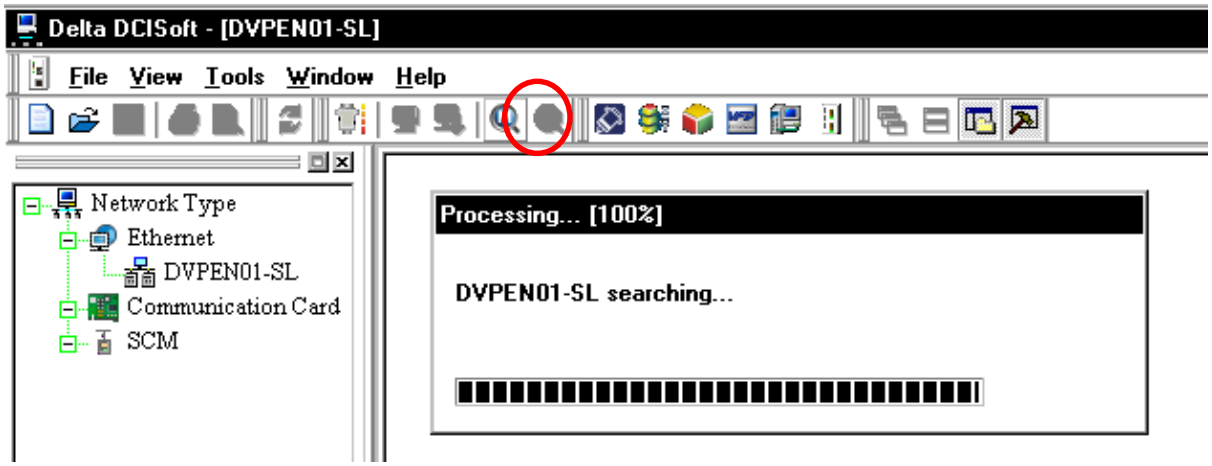
۶– تنظیمات DVPEN01 با استفاده از نرم افزار DCISoft

برای آنکه بتوانیم پارامترهای داخلی DVPEN01 را تنظیم کنیم می‌توان از نرم افزار DCISoft استفاده کرد. برای اینکار ابتدا DCISoft را باز می‌کنیم.

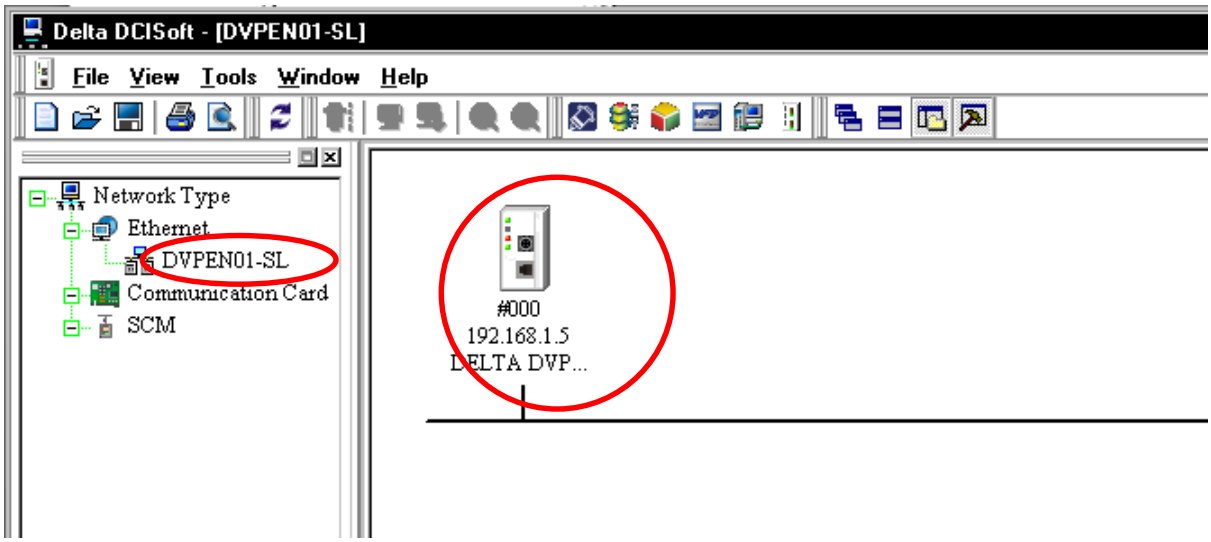




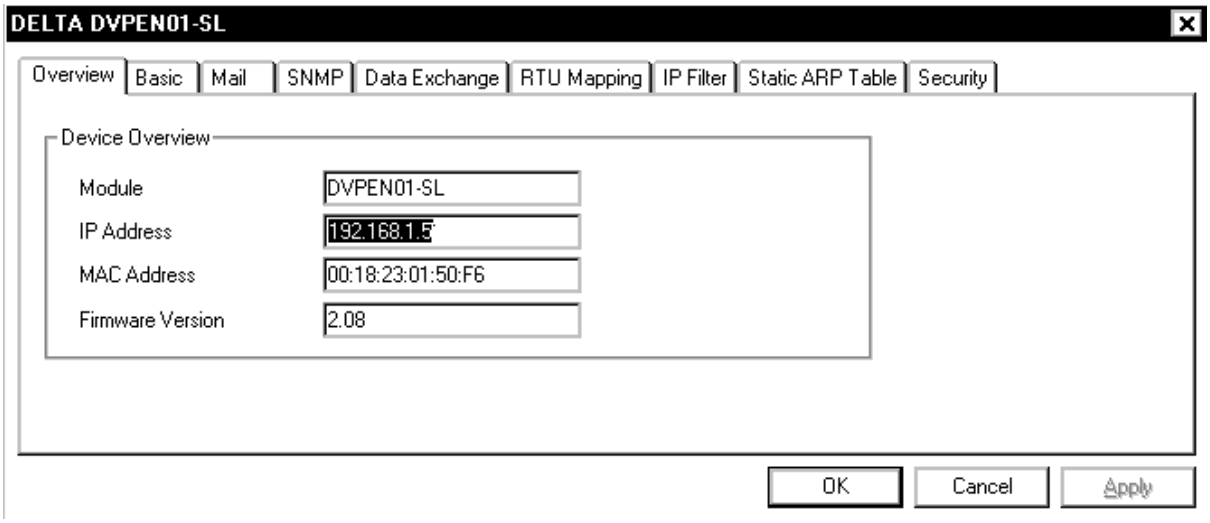
حال برای آنکه نرم افزار بتواند ماژول‌های درون شبکه Ethernet را پیدا کند، ابتدا بر روی Ethernet در قسمت Network Type کلیک کرده تا آیکون Search یعنی فعال شود، سپس بر روی آیکون کلیک می‌کنیم.



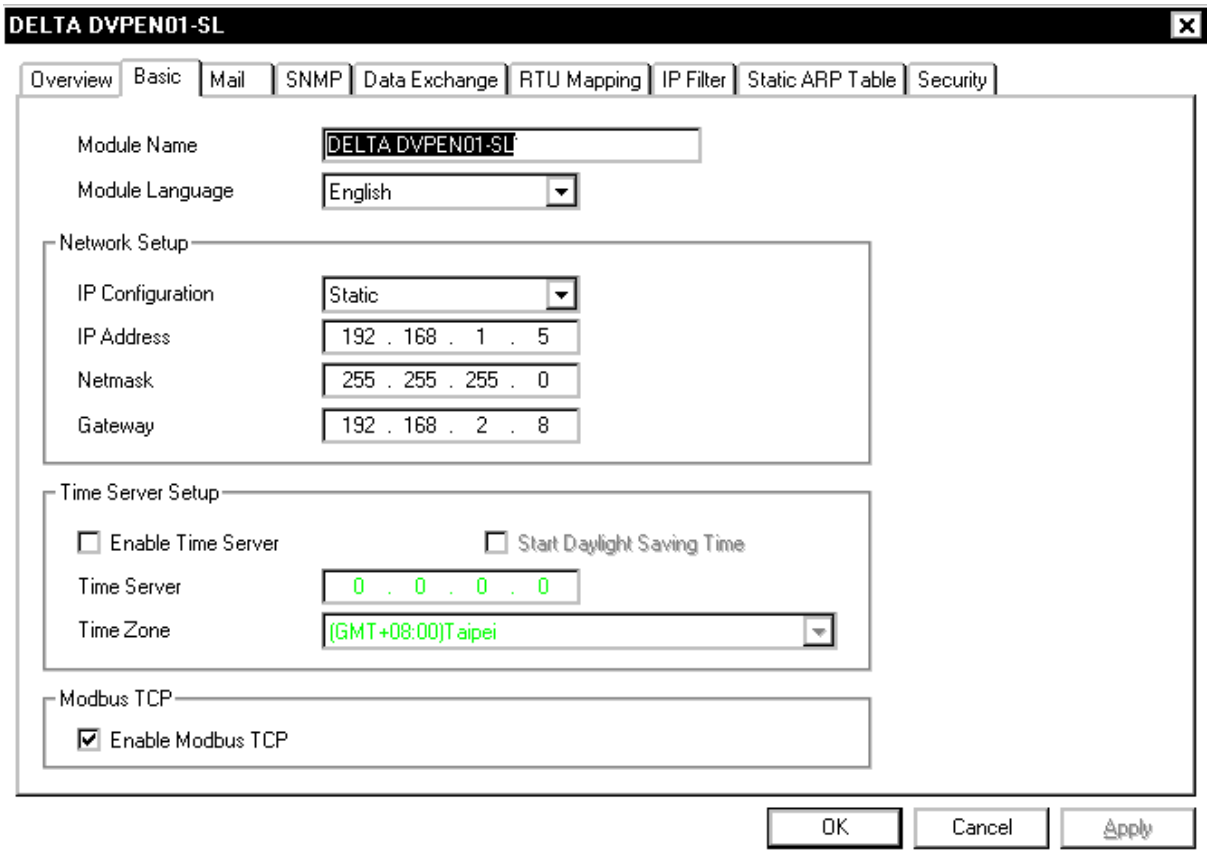
پس از طی شدن مراحل جستجو، DCISoft شبکه و ماژول‌های متصل به آن را برای ما لیست می‌کند. در شکل زیر می‌بینید که نرم افزار ماژول DVPEN01 را یافته و IP آن را برای ما نمایش می‌دهد.



با دوبار کلیک بر روی ماژول مورد نظر، صفحه تنظیمات آن باز خواهد شد. در سربرگ اول آن یعنی Overview می‌توانیم اطلاعات کلی پیرامون ماژول از جمله نام،IP و Mac Address را مشاهده کنیم.



در سربرگ Basic نیز می‌توان تنظیمات اولیه مربوط به ماژول را انجام دهیم. مهمترین بخش تنظیمات، انتخاب IP غیرتکراری برای ماژول می‌باشد. IP ماژول‌های DVPEN01 به صورت پیش فرض 192.168.1.5 می‌باشد. در صورتی که ما بیش از یک دستگاه DVPEN01 در شبکه داشته باشیم به علت تداخل نمی‌توانیم از IP پیش فرض برای همه ماژول‌ها استفاده کنیم و باید حتما IP آن‌ها را تغییر دهیم.



شرح پارامترهای سربرگ Basic

- Module Name: ممکن است تعداد زیادی ماژول DVPEN01 در شبکه موجود باشد، می‌توانیم با اختصاص نام منحصر به فرد به هرکدام، تشخیص آن‌ها را راحت تر کرد.
- Module Language: زبانی که برای نام ماژول در نظر گرفته‌اید.
- Enable Modbus TCP: برای فعال کردن Modbus TCP. در صورتی که این گزینه غیر فعال باشد، نرم افزار قادر به بارگزاری و استخراج برنامه نخواهد بود.
- Enable Time Correction: با فعال کردن این گزینه می‌توان از طریق سرور در زمان‌های مشخص، ساعت داخلی PLC را اصلاح کرد.
- Start Daylight Saving Time: در نظر گرفتن تغییر زمان رسمی کشور در ابتدای بهار و پاییز
- Time Server: آدرس IP سروری که می‌خواهیم از طریق آن ساعت PLC را اصلاح کنیم.

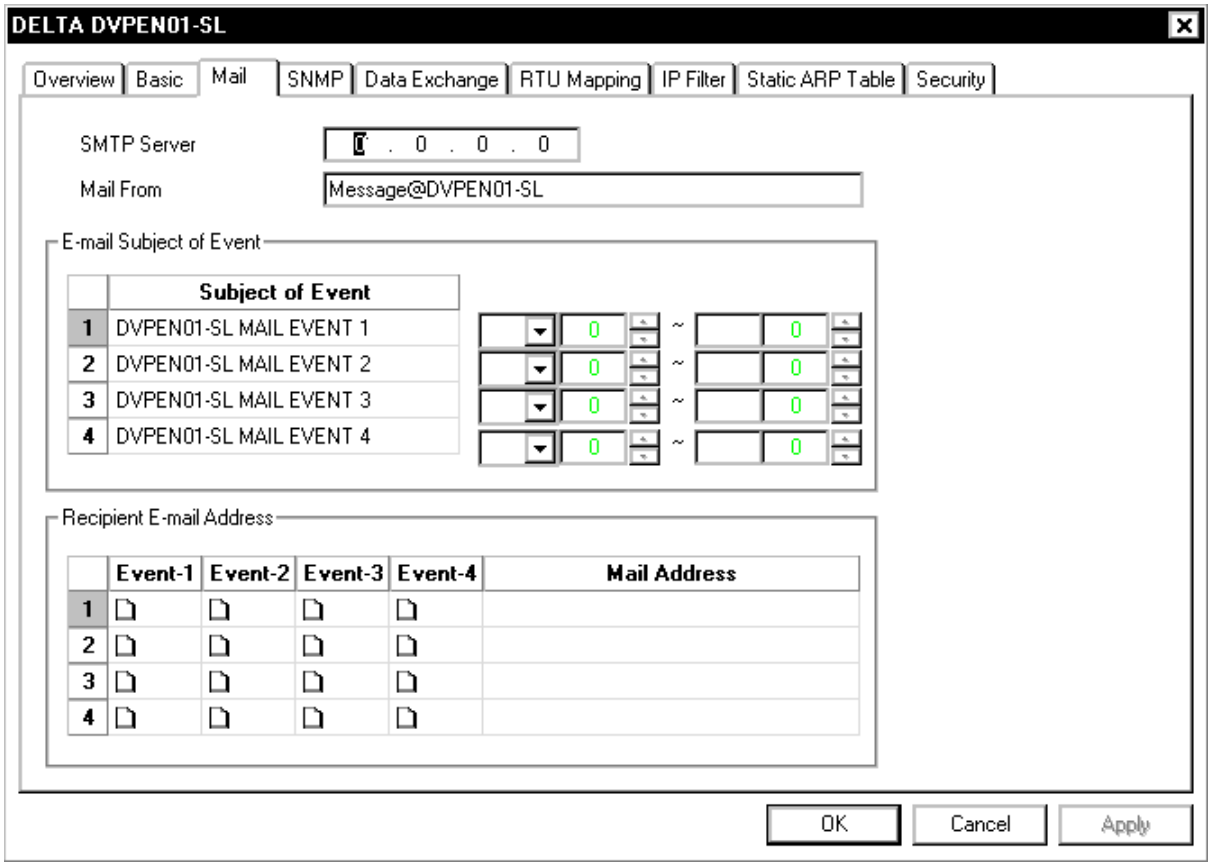
- Time Zone: ساعت منطقه جغرافیایی
- Network Setup:
  - IP Configuration: دو نوع روش اختصاص IP[1] دارد: Static که در آن کاربر به صورت دلخواه و دستی تنظیمات IP و Subnet mask و Gateway هر دستگاه در شبکه را تنظیم می‌کند و DHCP که در آن سرور بصورت اتوماتیک به تجهیزات متصل به شبکه IP و Subnet mask و Gateway اختصاص می‌دهد.
  - IP address: در واقع IP آدرس تجهیزات در شبکه است و تمام تجهیزات درون شبکه باید دارای IP باشند. در حالت پیش فرض IP ماژول های DVPEN01 به صورت 192.168.1.5 است. در صورتی که بیش از یک DVPEN01 در شبکه داشته باشیم، حتما باید IP آن ها را تغییر دهیم، چرا که هر ماژول باید دارای یک آدرس IP منحصر به فرد باشد. (در حالت DHCP سرور به صورت اتوماتیک به هر کدام از تجهیزات در شبکه یک IP منحصر به فرد اختصاص می‌دهد.)
  - Subnet mask و Gateway: Subnet mask برای تنظیم Subnet به کار می‌رود. در صورتی که IP مبدا و مقصد در یک Subnet نباشند، شبکه از طریق Gateway ارتباط بین مبدا و مقصد را برقرار می‌کند. به صورت پیشفرض ماژول DVPEH01 دارای Subnet mask اولیه 255.255.255.0 و Gateway اولیه 192.168.1.1 می‌باشد.

تنظیمات مربوط به ایمیل

DVPEN01 را می‌توان برای ارسال ایمیل‌های شرح وضعیت و یا خطا تنظیم کرد. این ماژول امکان ارسال ٤ مجموعه ایمیل را فراهم می‌کند. به همراه هر ایمیل نیز می‌توان وضعیت

[1] IP در واقع آدرس دستگاه‌های متصل به شبکه است و از طریق آن می‌توان مشخص کرد که در هر لحظه می‌خواهیم با کدام یک از تجهیزات متصل به شبکه ارتباط برقرار کنیم.

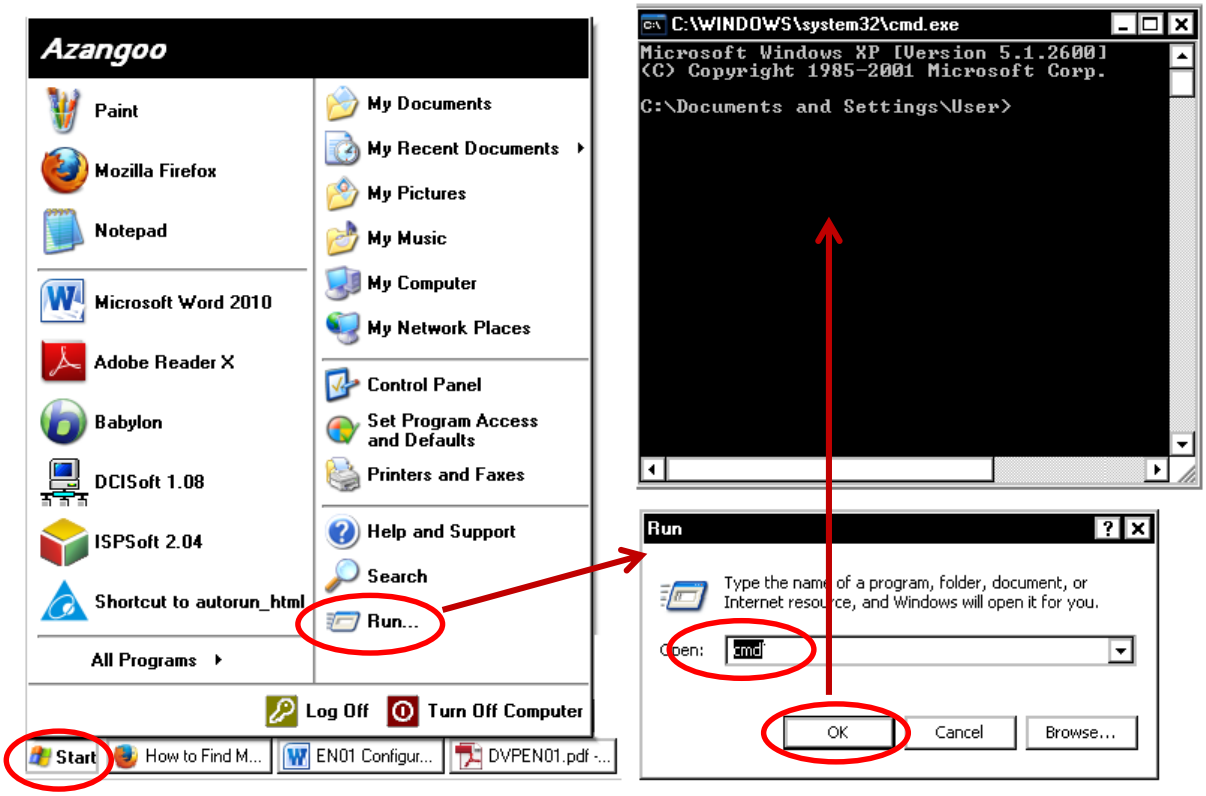
حافظه یا داده‌های تعریف شده را به کاربر ارسال کنیم. هرگاه تغییری در داده‌های مشخص شده به وجود آید آن را از طریق ایمیل می‌توانیم به کاربر اطلاع دهیم.



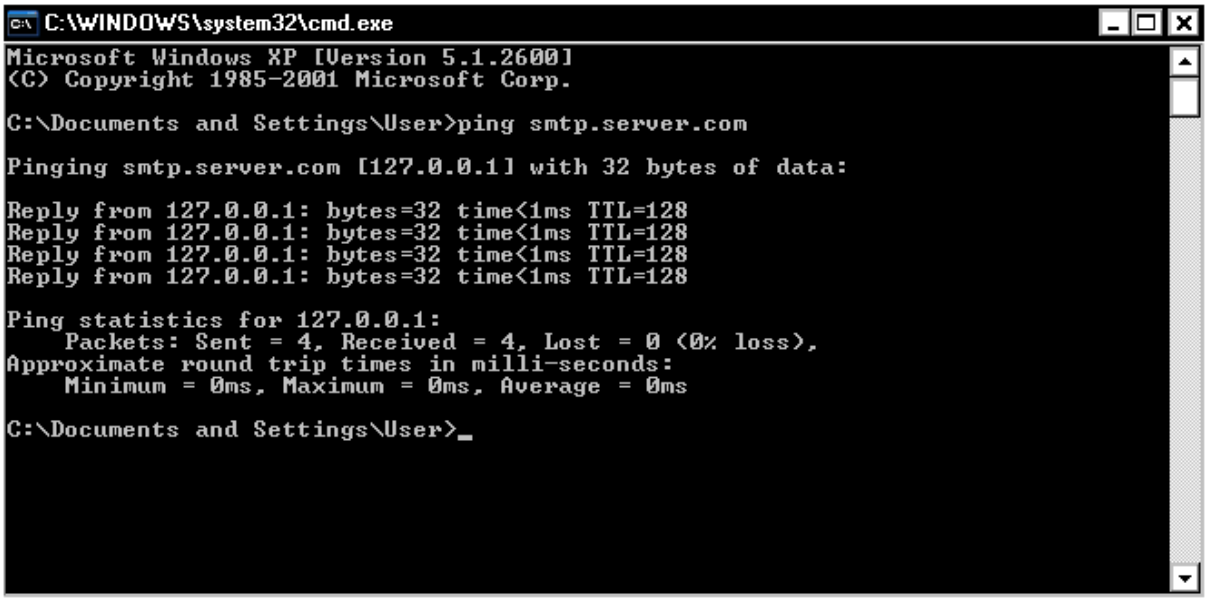
۱- SMTP Server: برای اینکه بتوان ایمیلی ارسال کرد، باید در شبکه سرور SMTP[۱] وجود داشته باشد بتوان ایمیل را ابتدا به آن ارسال و این سرور ایمیل را به آدرس مقصد ارسال کند. در قسمت SMTP باید آدرس IP سرور SMTP مشخص شود.

در صورتی که آدرس IP سرور SMTP خود را نمی‌دانید می‌توانید به صورت زیر عمل کنید.در منوی Start گزینه Run را انتخاب و در آن cmd را تایپ و بر روی OK کلیک کنید.

[۱] Simple Mail Transfer Protocol



در صفحه cmd.exe باز شده، Ping یک فاصله و نام سرور خود را تایپ کنید، مانند "ping smtp.server.com" (در صورتی که نام سرور خود را نمی دانید، احتمال دارد این نام به صورت پیشفرض smtp.server.com در نظر گرفته شده باشد.)



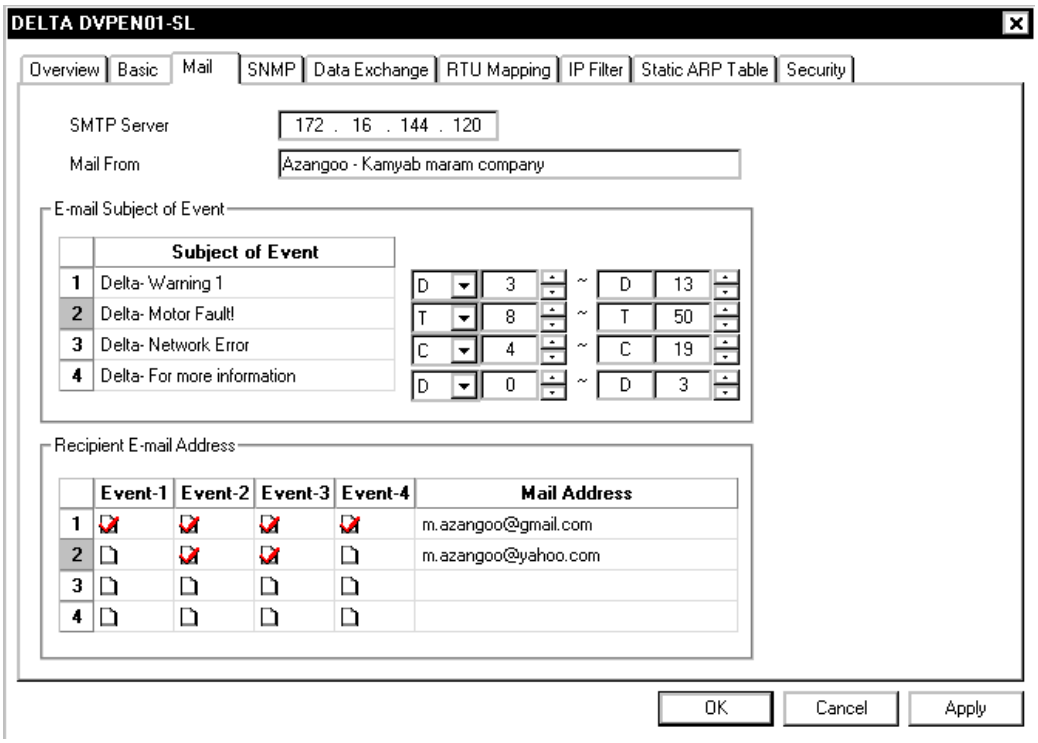
در این حالت ویندوز تلاش می‌کند با سرور SMTP ارتباط برقرار کند. و در نهایت پیغامی به صورت "Pinging x.x.x.x with 32 bytes of data." می‌دهد که در آن "x.x.x.x" همان آدرس IP سرور می‌باشد[1].

۲- Mail From: در این قسمت می‌توانیم نام فرستنده را مشخص کنیم.

۳- E-mail Subject of Event: در این قسمت می‌توانیم عنوان چهار ایمیل و البته (محدوده حداکثر ۱۰۰) رجیستری که داده‌های آن‌ها در ایمیل فرستاده می‌شود را تعیین کنیم.

٤- Recipient E-mail Address: آدرس گیرنده‌ها و اینکه کدام یک از چهار نوع ایمیل برای آن‌ها ارسال شود.

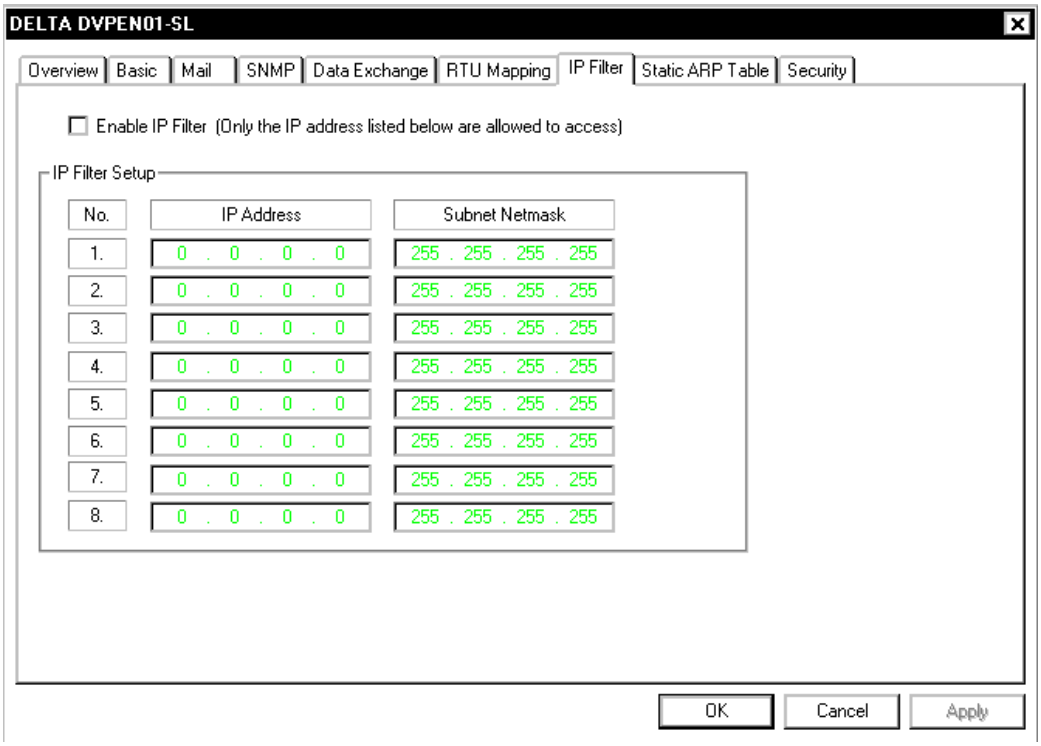
مثالی از این تنظیمات را در زیر می‌بینید.



محدود کردن دسترسی IPها

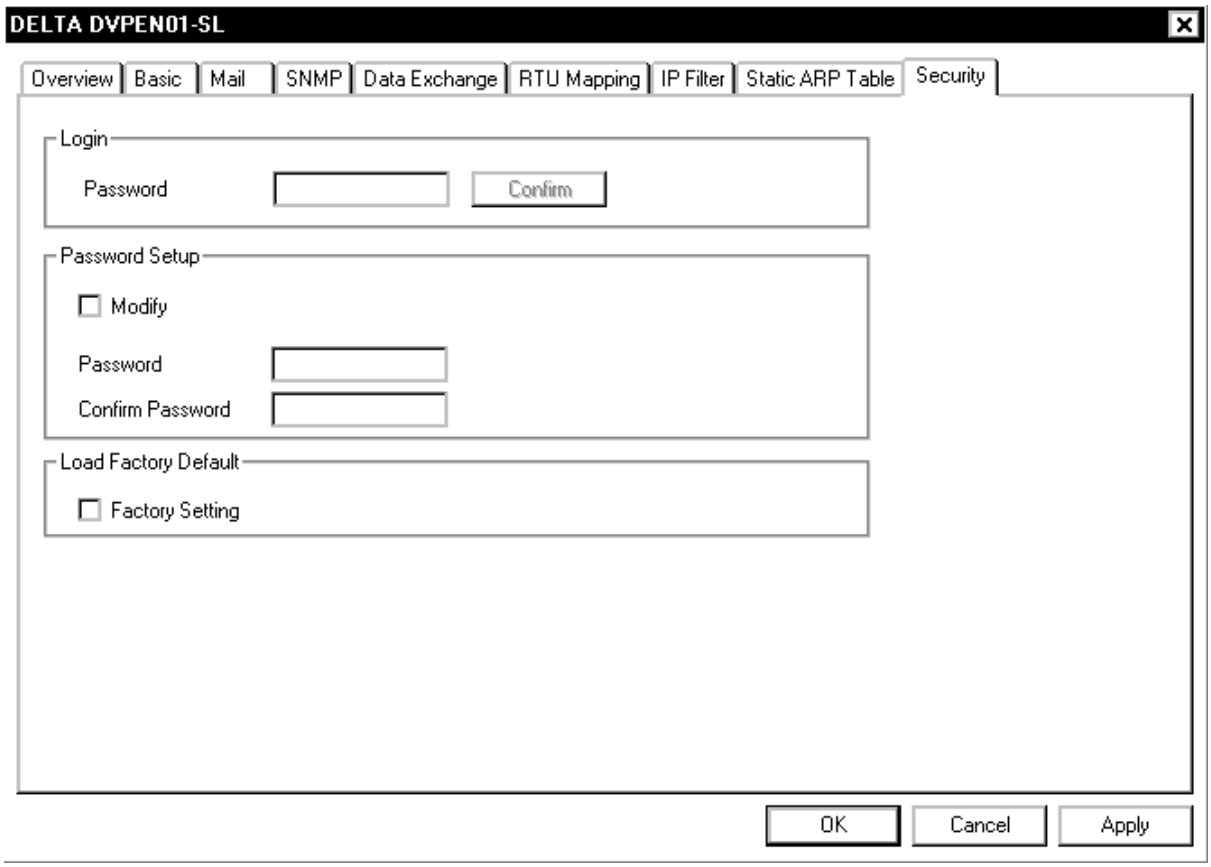
[1] http://www.ehow.com/how_5810894_smtp-server-ip-address.html

در سربرگ IP می‌توانیم IPهایی که سیستم می‌تواند با آن‌ها ارتباط برقرار کند را مشخص کنیم. برقراری ارتباط با دیگر IPها برای امنیت بیشتر و جلوگیری از خطاهای احتمالی مسدود می‌شود.



رمزعبور

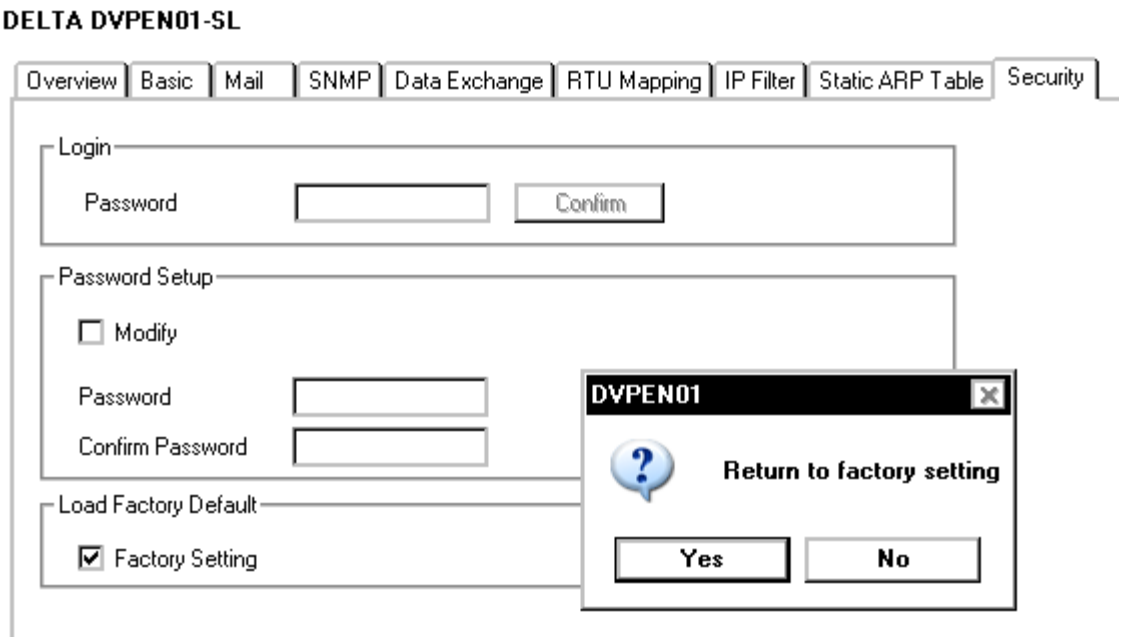
برای حفظ امنیت و اینکه هرکسی در شبکه نتواند تنظیمات DVPEN01 را تغییر دهد می‌توانیم بر روی آن رمزعبور تعبیه کنیم.



برای ایجاد رمز عبور، می‌توان Modify را انتخاب و یک رمز حداکثر ٤ کاراکتری در New Password تعیین کرد. رمز جدید را برای جلوگیری از اشتباه در قسمت Confirm Password نیز باید تکرار کنید.

برای تغییر رمز عبور پس از وارد کردن رمز قبلی و کلیک بر روی Confirm می‌توان Modify را انتخاب و یک رمز حداکثر ٤ کاراکتری در New Password تعیین کرد (اگر این قسمت را خالی بگزارید رمز عبور حذف خواهد شد). رمز جدید را برای جلوگیری از اشتباه در قسمت Confirm Password نیز باید تکرار کنید.

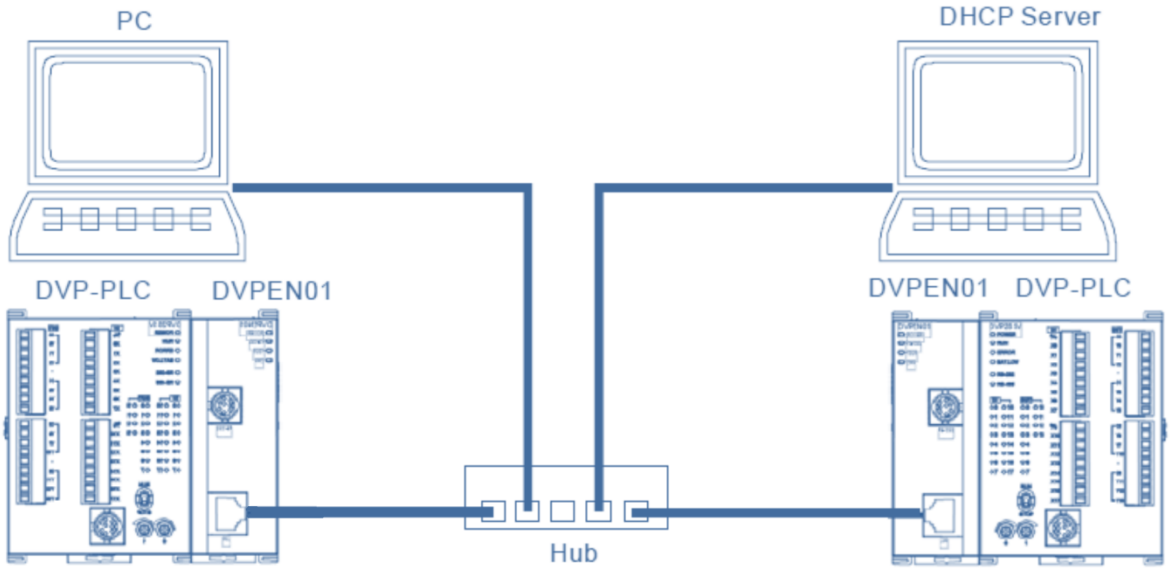
زمانی که رمز برای سیستم تعبیه شده باشد، برای هر تغییری در DVPEN01 ابتدا باید در سربرگ Security رمز را وارد و Confirm کرد. در صورتی که رمز عبور فراموش شده باشد، با استفاده از RS-232 می‌توان به DVPEN01 متصل و آن را ریست کرد. این کار با انتخاب Factory Setting در قسمت Load Factory Default امکان‌پذیر است.



پس از تایید، پروسه ریست شروع و حدود ۱۰ ثانیه طول می‌کشد و در این حین **به هیچ وجه نباید** تغذیه قطع شود.

۷– تنظیم IPماژول DVPEN01 به صورت DHCP

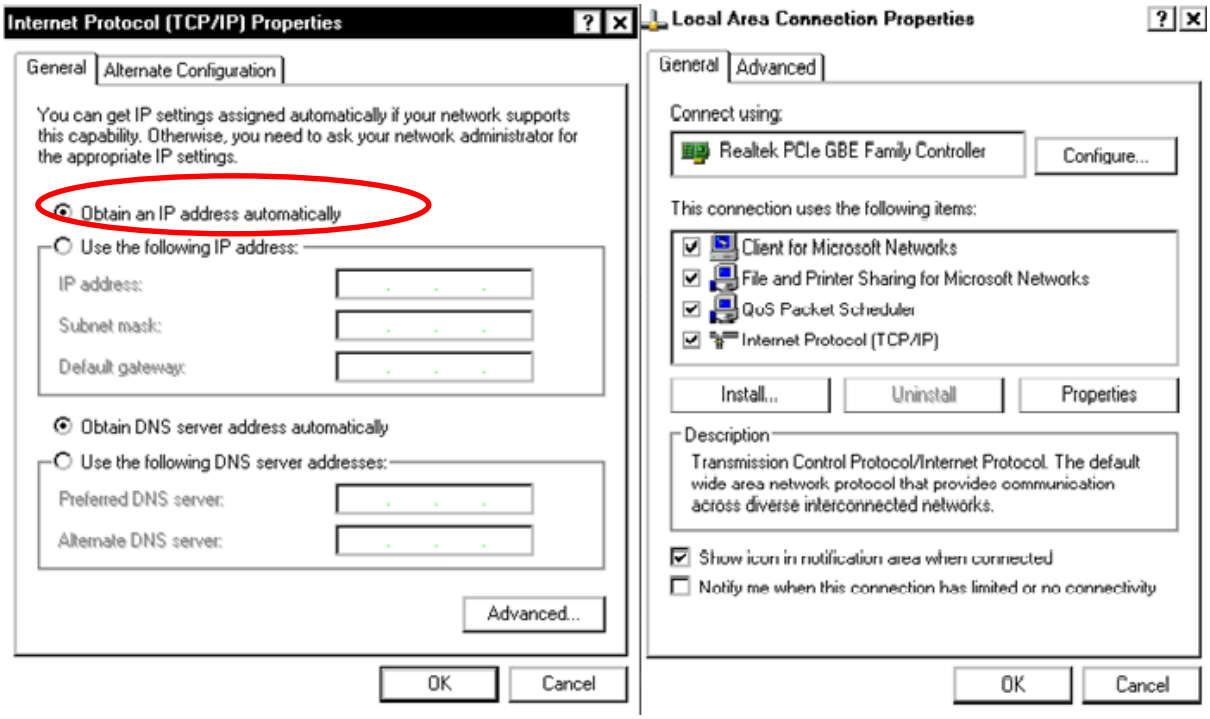
در بخش‌های قبلی به اختصاص IP سیستم به صورت استاتیک پرداختیم. در این قسمت می-خواهیم چگونگی تنظیم IP سیستم به صورت دینامیک در DHCP بپردازیم. در این حالت سرور به صورت اتوماتیک یک IP آزاد را به DVPEN01 اختصاص خواهد داد. (در این حالت برای اتصال کامپیوتر به DVPEN01 نیاز به سرور وHub است). اتصالات کامپیوتر به DVPEN01 در این حالت به صورت زیر در خواهد آمد:



توجه شود که در این حالت برای اینکه تخصیص IP تغییر نکند و تنظیمات سرور عوض نشود، نباید سرور هیچگاه خاموش و یا قطع شود.

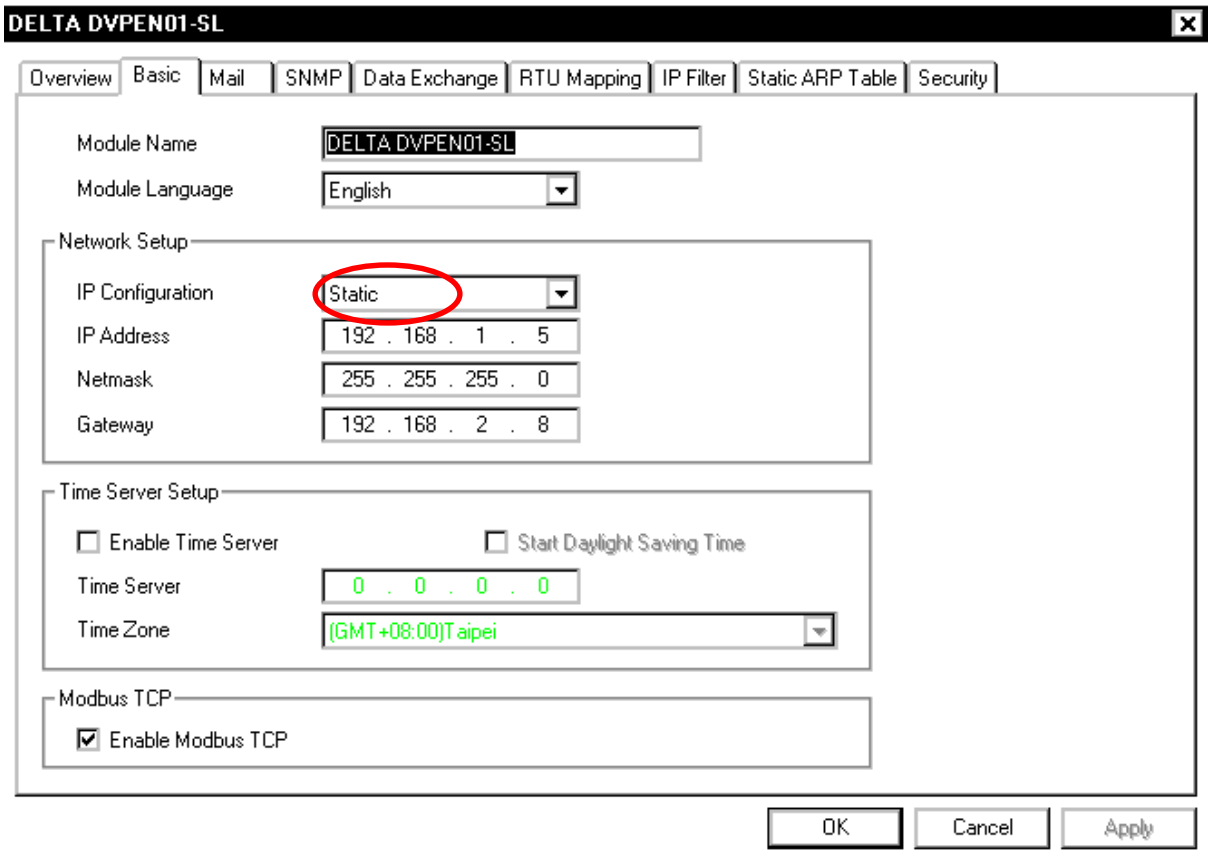
تنظیم IP کامپیوتر به صورت DHCP

در این حالت باید در صفحه Internet Protocol (TCP/IP) گزینه Obtain an IP Address automatically را انتخاب کنید. و صفحات باز شده را تایید کنید.

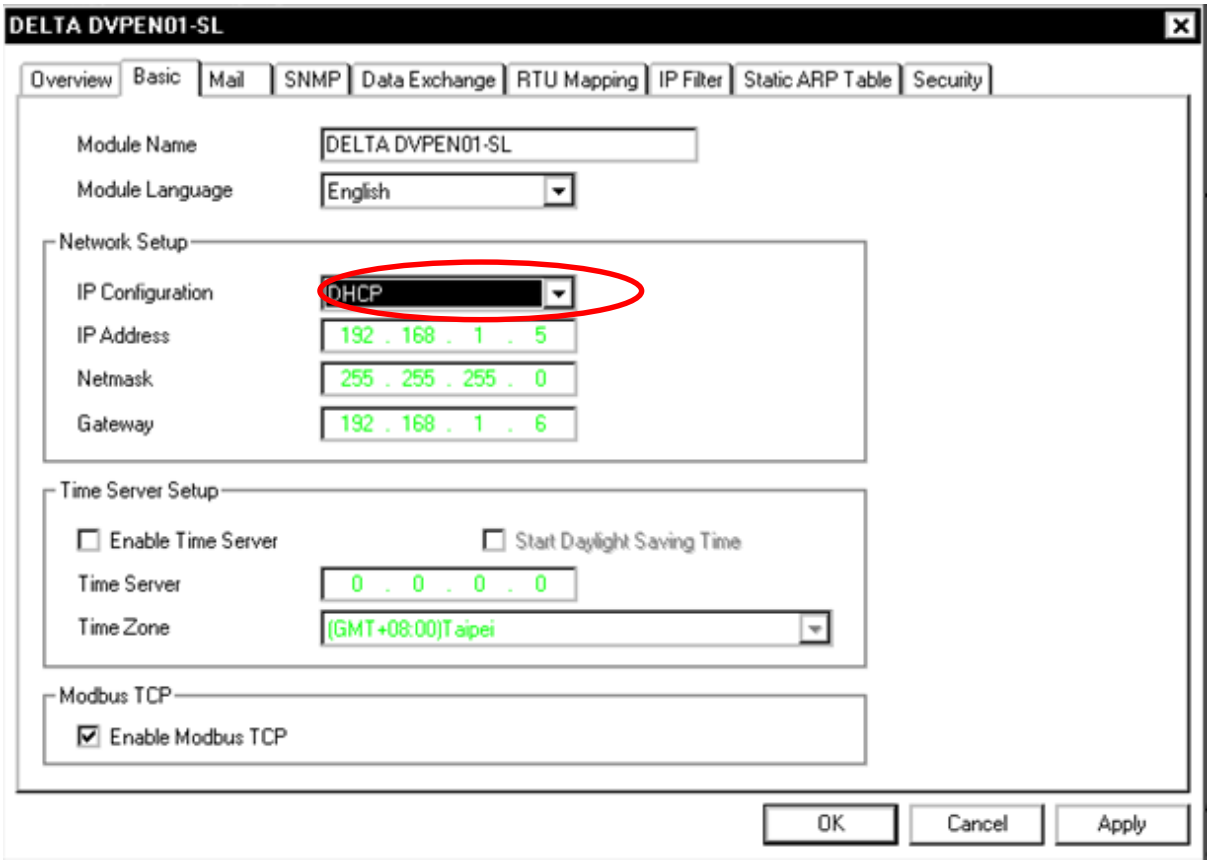


تنظیم IP ماژول DVPEN01 به صورت DHCP

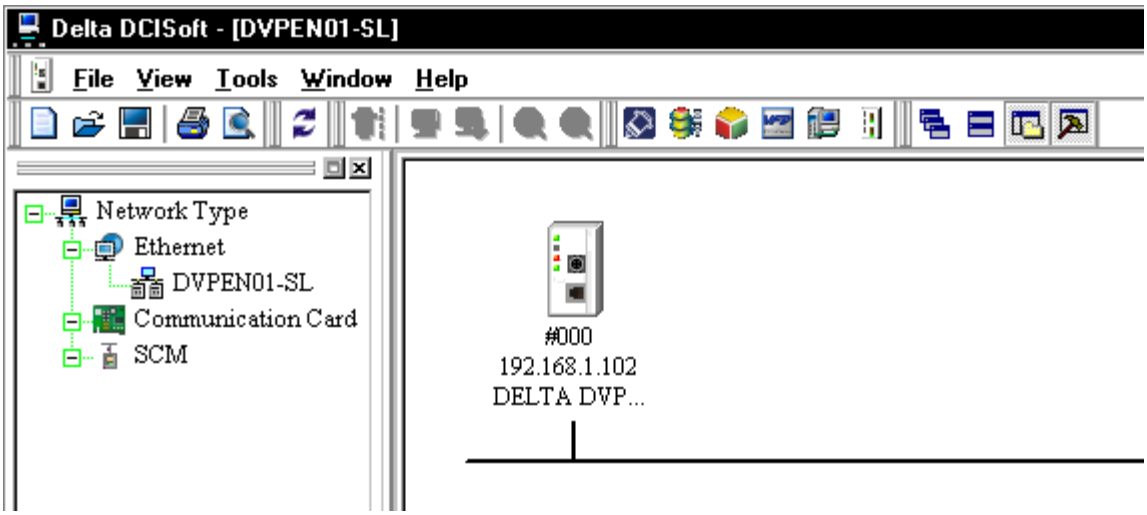
به صورت پیش فرض نوع تخصیص IP برای DVPEN01 به صورت Static است.



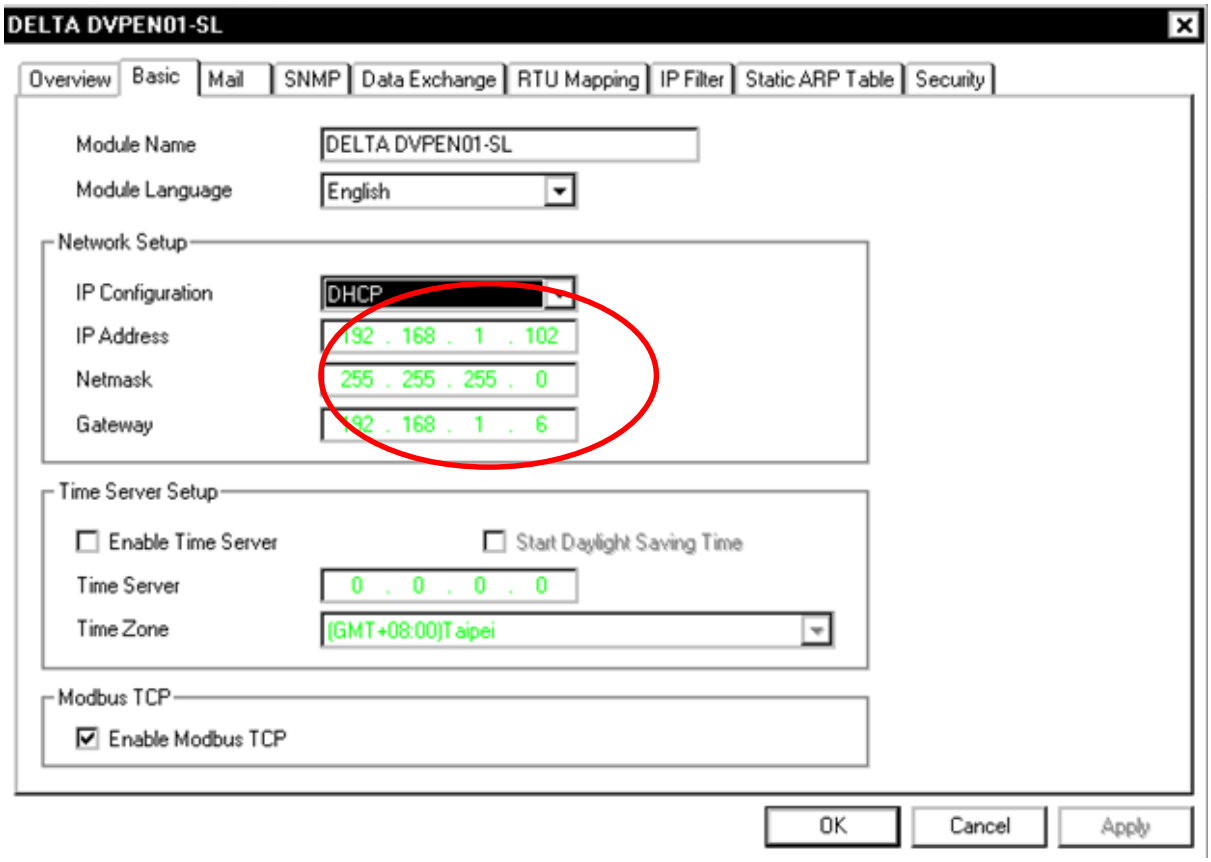
برای تغییر آن، در DCISoft در سربرگ Basic می‌توانیم Type سیستم را به صورت DHCP انتخاب کنیم.



مشخص است که در این حالت IP و دیگر گزینه ها در این صفحه به صورت غیرفعال در آمده‌اند و کاربر قادر به تغییر آن‌ها نمی‌باشد. حال برای اینکه IP جدید تخصیص داده شده به سیستم را ببینیم، می‌توانیم بر روی OK کلیک کرده و نرم افزار DCISoft را بسته و دوباره باز کنیم و سپس بر روی آیکون کلیک می‌کنیم تا نرم‌افزار IP جدید ماژول را شناسایی کند.

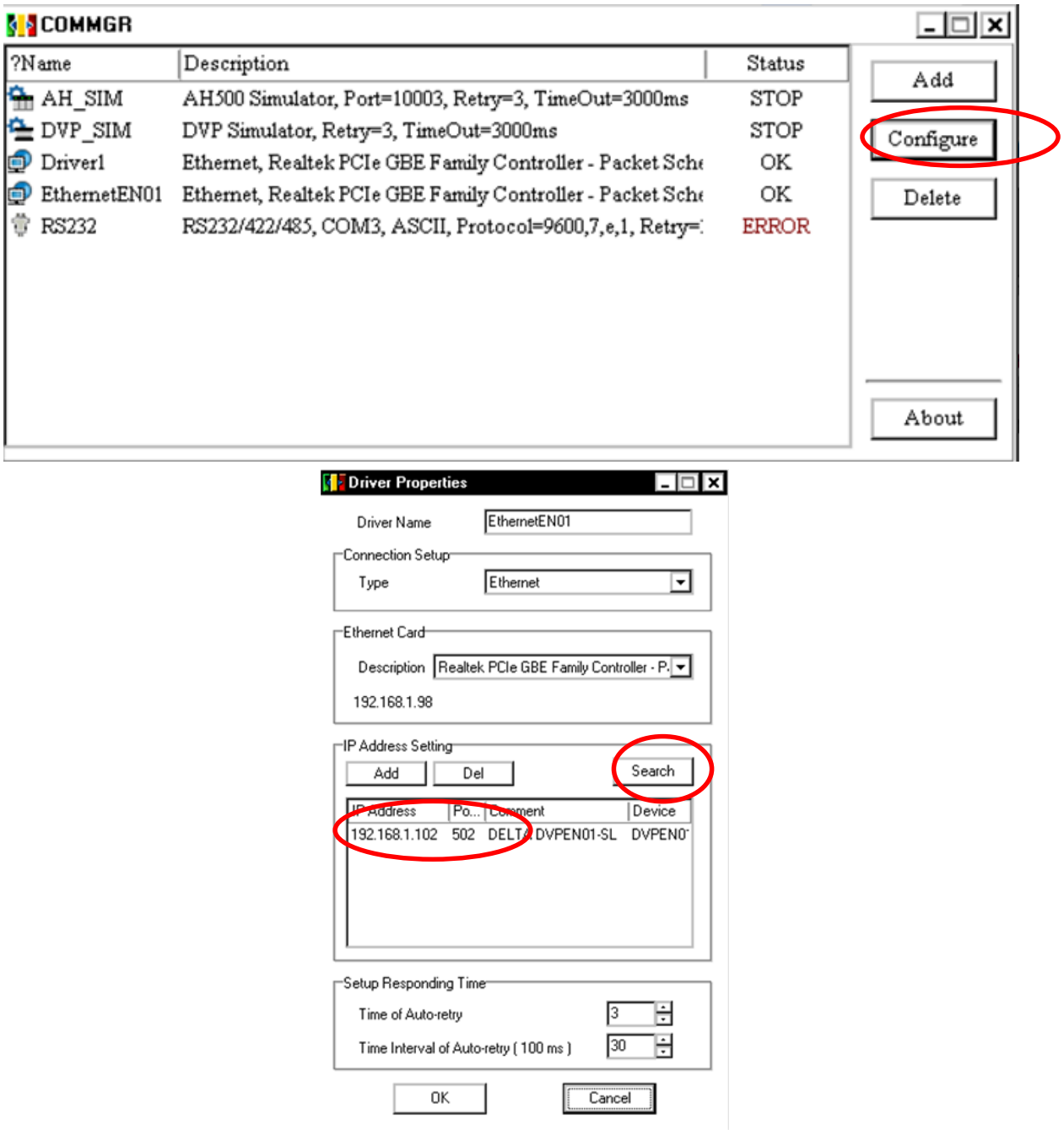


مشخص است که IP اختصاص داده شده به سیستم به صورت 192.169.1.102 است. با دوبار کلیک بر روی ماژول می‌توان به بقیه تنظیمات تخصیص داده شده دسترسی داشته باشیم.



شناسایی IP جدید در COMMGR

برای ارتباط ISPSoft با DVPEN01 نیاز است IP آن را در COMMGR لیست کنیم. برای این کار کافی است پس از تنظیم IP به صورت DHCP یکبار دیگر درایور مورد نظر را انتخاب و با کلیک بر روی Configure یکبار دیگر IP ها را جستجو کنیم.



IP تنظیم شده برابر 192.168.1.12 به وسیله COMMGR پیدا شده.

فهرست مراجع

- ISPSoft User Manual, 18/01/2013
- ISPSoft User Manual, 18/03/2015
- AH500 Programming Manual, 09/11/2012
- DVP-PLC Application Manual (Programming), 30/05/2013
- Delta AH500 PLC series, DVP PLC series, DOPsoft, WPLsoft, DCISoft and DVPEN01 Module Documents

- محمدرضا ماهر، "پیکر بندی و برنامه نویسی شبکه اترنت صنعتی با نرم افزار STEP7"، انتشارات قدیس، ۱۳۹۰

- محمد صمدیان، "دستور کار آزمایشگاه PLC دانشگاه صنعتی خواجه نصیرالدین طوسی"، ۱۳۹۰

- برای تکمیل این کتاب از الگوی پایان نامه دانشجویان دانشگاه خواجه نصیر، آماده شده به وسیله دکتر تقی راد استفاده شده است.

واژه نامه فارسی به انگلیسی

Ethernet	اِترنت
Execut	اجرا
Static	استاتیک
Upload	استخراج
Octal Value	اکتال (مبنای هشت)
Past	الحاق
Station Address	آدرس ایستگاه
Download	بارگزاری
Feedback	بازخورد
Message display area	بخش پیام
Project management area	بخش مدیریت پروژه
Qualifier	بررسی کننده
Flag	پرچم
Jump	پرش
STEP	پله
Configuration	پیکربندی
Function blocks	توابع بلوکی
Comment	توضیحات
Table of local symbols	جدول نمادها (سیمبول-های) محلی
Force	حالت اجبار

Communication Commands	دستورات ارتباطی
Instructions	دستورالعمل
Decimal Value	دسیمال (ده دهی)
Double Click	دوبار کلیک چپ
Hard Disk Drive	دیسک سخت
Index Register	رجیستر میانی
Assembly	زبان اسمبلی
Period	زمان دوره
Hierarchical Tree Structure	ساختار سلسله مراتبی درختی
Overflow	سرریز
Global symbols	سیمبول های سراسری
local symbols	سیمبول های محلی
local area network (LAN)	شبکه محلی
High-speed counter	شمارنده های پر سرعت
Task	عملکرد
Action	عملکرد
Operator	عملگرها
Operand	عملوند
Frequency	فرکانس

Drag and Drop	فشردن و نگه داشتن کلیک چپ حین جابجا کردن آن
Portable	قابل انتقال
Compile	کامپایل کردن
Contact	کانتکت
Rising edge-triggered contact	کانتکت تشخیص دهنده لبه بالا رونده
Falling edge-triggered contact	کانتکت تشخیص دهنده لبه پایین رونده
Normally-open contact	کانتکت در حالت عادی باز
Normally-close contact	کانتکت در حالت عادی بسته
Keywords	کلمات کلیدی
Left Click	کلیک چپ موس
Right Click	کلیک راست موس
Delta	کمپانی دلتا
Coil	کویل
Transition	گذار

Node	گره
Monitoring	مانیتورینگ
Function Block Definition	مرجع تابع بلوکی
Routing	مسیریابی
Shortcut	میان بر
Simulink	نرم افزار سیمولینک
MATLAB	نرم افزار متلب
Programs	نرم افزارها
Function Block Instance	نسخه تابع بلوکی
Bookmark	نشانه گزاری
Symbols	نمادها
Hexadecimal	هگزادسیمال (مبنای ۱۶)
Program organization units (POU)	واحدهای سازماندهی برنامه
Conditional interrupt	وقفه شرطی

واژه نامه انگلیسی به فارسی

Action	عملکرد
Assembly	زبان اسمبلی
Bookmark	نشانه گزاری
Coil	کویل
Comment	توضیحات
Communication Commands	دستورات ارتباطی
Compile	کامپایل کردن
Conditional interrupt	وقفه شرطی
Configuration	پیکربندی
Contact	کانتکت
Decimal Value	دسیمال (ده دهی)
Delta	کمپانی دلتا
Double Click	دوبار کلیک چپ
Download	بارگزاری
Drag and Drop	فشردن و نگه داشتن کلیک چپ حین جابجا کردن آن
Ethernet	اِترنت
Execut	اجرا
Falling edge-triggered contact	کانتکت تشخیص دهنده لبه پایین رونده
Feedback	بازخورد
Flag	پرچم
Force	حالت اجبار
Frequency	فرکانس
Function Block Definition	مرجع تابع بلوکی
Function Block Instance	نسخه تابع بلوکی
Function blocks	توابع بلوکی
Global symbols	سیمبول های سراسری
Hard Disk Drive	دیسک سخت
Hexadecimal	هگزادسیمال (مبنای ۱۶)
Hierarchical Tree Structure	ساختار سلسله مراتبی درختی
High-speed counter	شمارنده های پر سرعت
Index Register	رجیستر میانی
Instructions	دستورالعمل
Jump	پرش
Keywords	کلمات کلیدی
Left Click	کلیک چپ موس
local area network (LAN)	شبکه محلی

Project management area	بخش مدیریت پروژه
Qualifier	بررسی کننده
Right Click	کلیک راست موس
Rising edge-triggered contact	کانتکت تشخیص دهنده لبه بالا رونده
Routing	مسیریابی
Shortcut	میان بر
Simulink	نرم افزار سیمولینک
Static	استاتیک
Station Address	آدرس ایستگاه
STEP	پله
Symbols	نمادها
Table of local symbols	جدول نمادها (سیمبول-های) محلی
Task	عملکرد
Transition	گذار
Upload	استخراج

local symbols	سیمبول های محلی
MATLAB	نرم افزار متلب
Message display area	بخش پیام
Monitoring	مانیتورینگ
Node	گره
Normally-close contact	کانتکت در حالت عادی بسته
Normally-open contact	کانتکت در حالت عادی باز
Octal Value	اکتال (مبنای هشت)
Operand	عملوند
Operator	عملگرها
Overflow	سرریز
Past	الحاق
Period	زمان دوره
Portable	قابل انتقال
Program organization units (POU)	واحدهای سازماندهی برنامه
Programs	نرم افزارها

Delta Electronics

Persian ISPSoft User Manual

and Introduction to Delta AH500 and DVP PLC Series

& Industrial Networks

Mohammad Azangoo

Spring 2015