بسمه تعالی

نحوه استفاده و آشنایی با PLC خانواده زیمنس و نرم افزار مربوطه تحت عنوان

Simatic S7 Siemens Industrial Automation

فهرست :

ادامه فهرست :

پيش گفتار مولف :

همانگونه كه تمامي همكاران عزيز مستحضر مي باشيد امروزه با پيشترفت علم و تكنولوژي كاربرد كاميوتر در صنعت نيز نمود پيدا كرده است بدين گونه كه بجاي تابلوهاي فرمان و قدرت دستگاههاي خط توليد كه تماما بصورت رله كنتاكتوري ساخته مي شد و در نتيجه براي تعميرو عيب يابي هر ايراد اولا به تجربه و شناخت كافي از تابلو نياز بود و ثانيا بايستي مرحله به مرحله تمامي تابلو از طريق نقشه كنترل مي گرديد تا ايراد مشخص گردد اما امروزه از دستگاهي بنام plc استفاده مي گردد و بوسيله plc علاوه بر اينكه مي توان هر ايرادي را مونيتور نموده و بر روي صفحه op نمايش داد بدون نياز به نقشه و براحتي مي توان ايرادات دستگاه ها را در كوتاه ترين زمان ممكن مشخص و رفع عيب نمود .

از ديگر مزاياي plc قابليت اتصال انها به يكديگر و برقراري ارتباط شبكه مي باشد كه بدين وسيله مي توان از طريق يک كامپيوتر مركزي در اتاق كنترل تمامي تجهيزات و plc هاي داخل خط توليد را كنترل و مورد بررسي قرار داد كه اين موضوع باعث كاهش نيروي ماهر در خط توليد و افزايش دقت و راندمان تجهيزات مي گردد .

در كارخانه معظم ايران خودرو از سال ۱۳۷۹ تا كنون تقريبا تمامي تجهيزات خطوط توليد سالن هاي مختلف از قبيل مونتاژ ، رنگ ، بدنه سازي ، پرس ، موتورسازي و ريخته گري از حالت قديمي خارج شده و مجهز به سيستم اتوماسيون صنعتي و Plc گرديده است .

از اين رو احتياج به متخصصين plc در اين مدت روز بروز افزايش يافته و در حال گسترش نيز مي باشد .

از انجاییکه در داخل کشور این تکنولوژي و رشته فني بصورت یک رشته تخصصي در دانشگاهها و اموزشگاههاي کشور تدریس نمي گردد و یا بصورت بسیار محدود وجود دارد لذا بندرت شرکت ایران خودرو مي تواند کارشناسان با تجربه را در این رشته تخصصي استخدام نماید

بنابراین بهترین گزینه آموزش افراد در داخل شرکت مي باشد که در سالهاي اخیر مرکز محترم آموزش ایران خودرو در این زمینه گامهاي اساسي برداشته و اقدام به راه اندازي آزمایشگاه plc نموده است .

اگر دوستان و همکاران عزیز جهت خرید کتابهاي مربوط به plc به روبروي دانشگاه تهران مراجعه نموده باشند در مي یابند که اکقریت کتابها ي نوشته شده در این زمینه مربوط به تئوري و مفهوم Plc و شناخت قطعات داخل Plc مي باشد و کتابي که در رابطه با استفاده از نرم افزار plc زیمنس سري خانواده s7 نوشته شده باشد بندرت یافت مي گردد

از انجاییکه در کارخانه ایران خودرو استفاده از زیمنس سري s7 متداول مي باشد از این رو نیاز به یک مرجع اصلي جهت استفاده از نرم افزار simatic s7 بعنوان راهنماي کارشناسان و تکنسین هاي محترم الکترونیک شرکت بیش از پیش احساس مي گردد

در این جزوه فرض بر این مي باشد که همکاران عزیز مفهوم plc و مدارات منطقي و علم لاجیک را که در کتابهاي مختلف به تفسیر آموزش داده شده است دانسته و بجهت استفاده سریع از نرم افزار s7 از این مجموعه استفاده نمایند .

امید است که تلاش چندین ماهه این حقیر بعنوان یک مرجع مورد استفاده همکاران محترم قرار گرفته و مفید واقع گردد

در پايان از همكاران و صاحبنظران اين رشته بابت نواقص موجود عذر خواهي نموده و استدعا دارم تا ضمن مطالعه اين مجموعه ايرادات موجود را مشخص و نظرات خود را براي اينجانب ارسال نمايند تا بتوانم در اسرع وقت مشكلات را برطرف و مجموعه ايي نسبتا كامل را در اختيار همكاران عزيز قرار بدهم .

در اينجا جا دارد از تمامي همكاران عزيز و خصوصا روساي محترم اداره كل و اداره پشتيباني تعميرات مونتاژ كه در تهيه اين مجموعه بنده را مورد لطف و پشتيباني خود قرار داده اند كمال سپاسگزاري را بنماتيم .

يا تشكر

محمد يادگار

كارشناس فني تعميرات مونتاژيک

مقدمه :

- **ساختار PLC**

PLC از عبارت programable logic control به معنای کنترل کننده منطقی قابل برنامه ریزی گرفته شده است . PLC کنترل کننده ای نرم افزاری است که در قسمت ورودی اطلاعاتی را به صورت باینری یا آنالوگ دریافت و آنها را طبق برنامه ای که در حافظه اش ذخیره شده است پردازش می نماید و نتیجه عملیات را نیز از قسمت خروجی به صورت فرمانهایی به گیرنده ها و اجرا کننده های فرمان ارسال می کند. به عبارت دیگر PLC عبارت از یک کنترل کننده منطقی است که می تواند منطق کنترل را توسط برنامه برای آن تعریف نمود ودر صورت نیار براحتی آن را تغییر داد.

وظیفه PLC قبلاً بر عهده مدارات فرمان و رله های کنتاکتوری بود که امروز استفاده از آنها منسوخ شده است.از اشکالات عمده این رله ها این بود که با افزایش این رله ها حجم و وزن مدارات فرمان بسیار بزرگ شده و قیمت آنها نیز افزایش مي یافت و نهایتا عیب یابي اینگونه مدارات بسیار پیچیده و زمان بر مي گردید .

برای رفع این معظل مدارات فرمان الکترونیکی ساخته شدند که آنها نیز به علت اینکه تک کار بودند و برای استفاده در چند مدار می بایستی تغییرات عمده در آنها ایجاد می شد کارایی کمی داشتند.

با استفاده از PLC تغییر در روند تولید یا عملکرد ماشین به راحتی صورت می گیرد زیرا دیگر لازم نیست سیم کشی ها و سخت افزار سیستم کنترل تغییر کند و تنها کافی است چند سطر برنامه نوشت و به PLC ارسال کرد تا کنترل مورد نظر تحقق یابد .

از طرف دیگر قدرتPLC در انجام عملیات منطقی و محاسباتی و مقایسه ای و نگهداری اطلاعات به مراتب بیشتر از تابلوهای فرمان معمولی است .PLC به طراحان این امکان را میدهد که آنچه را که در ذهن دارند در اسرع وقت بیازمایند .

هر کس با مدارات فرمان رله ای کار کرده باشد به خوبی می داند که پس از طراحی تابلو اگر نکته ای از کار افتاده باشد مشکلات بسیاری برای رفع آن پیش روست و زمان زیادی نیز صرف خواهد شد.

اکنون برای توجه بیشتر به تفاوت ها و مزایای PLC نسبت به مدارات کنتاکتوری موارد زیر را بر می شماریم:

۱- استفاده از PLC موجب کاهش حجم تابلوی فرمان می گردد.
۲- استفاده از PLC مخصوصآ در فرایند های عظیم موجب صرفه جویی قابل توجهی در هزینه لوازم و قطعات میشود.
۳- PLC استهلاک مکانیکی ندارد بنا براین علاوه بر عمر بیشتر نیازی به تعمیرات و سرویس های دوره ای نخواهد داشت .
٤- PLC انرژی کمتری مصرف می کند .
٥- PLC ها بر خلاف مدارات رله کنتاکتوری نویز الکتریکی و صوتی ایجاد نمی کند
٦- استفاده از یک PLC منحصر به یک پروسه و فرایند خاصی نیست و با تغییراتی در برنامه می توان به آسانی از آن برای کنترل پروسه های دیگر استفاده کرد.
۷- طراحی و اجرا ی مدارات کنترل و فرمان با استفاده از PLC بسیار سریع و آسان است.

۸- برای عیب یابی مدارات کنتاکتوری الگوریتم و روش خاصی نداریم اما در عیب یابی مدارات PLC براحتی با تغییرات در نرم افزار و*SIMOULATION* کردن آن می توان عیب یابی کرد..

- **کاربرد های PLC در صنایع مختلف :**

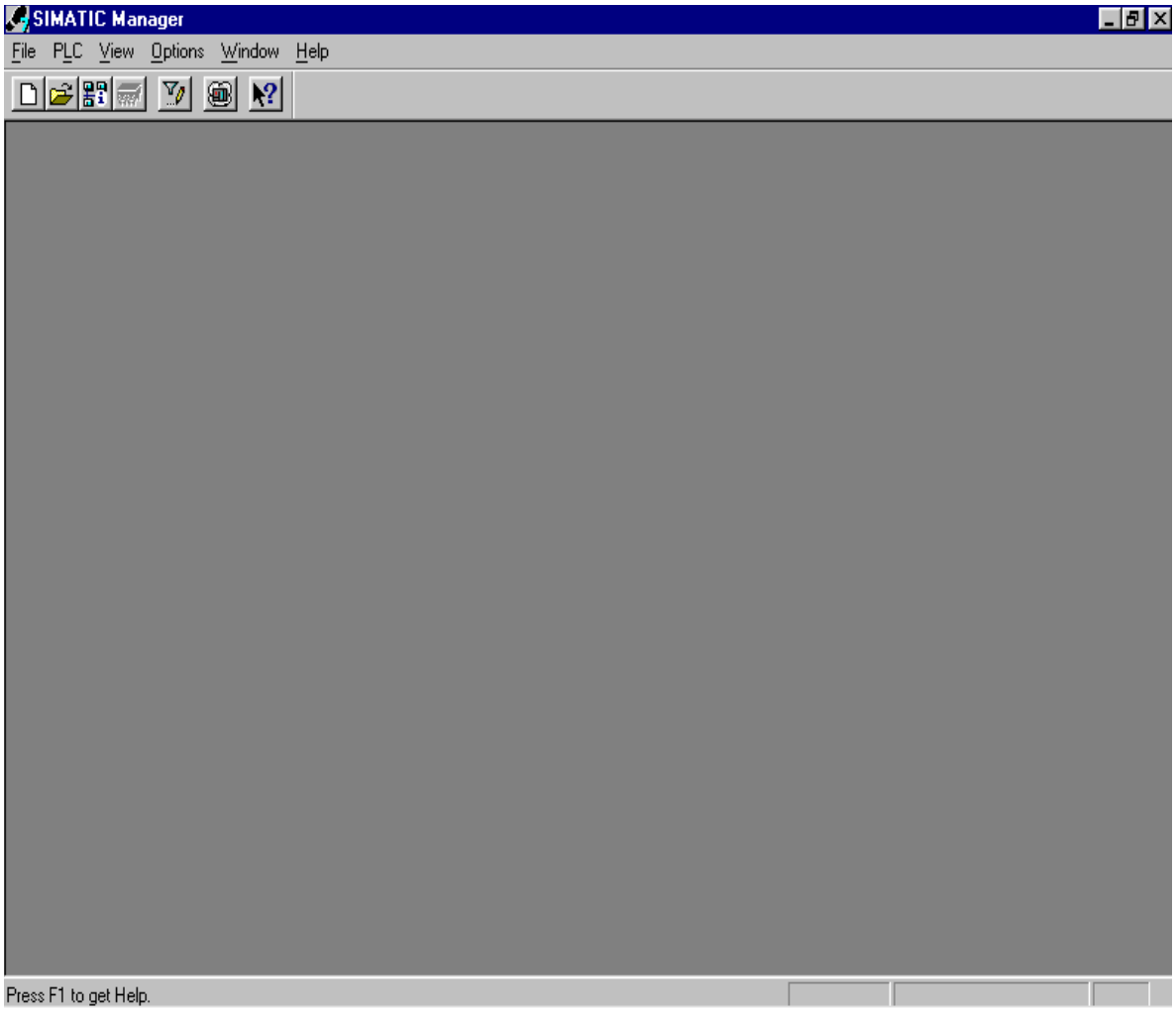
امروزه کاربرد های فراوانی از PLC در پروسه های مختلف صنعتی به چشم می خورد که خود نشانگر اهمیت فراوانPLC در صنعت است .از جمله این استفاده ها می توان به موارد زیر اشاره کرد:

- صنایع اتومبیل سازی شامل سوراخ کاری و پاشش رنگ و حمل موتور LIFT و DROP .
- صنایع پلاستیک سازی شامل ذوب قالبگیری و دمش هوا
- صنایع سنگین شامل کوره های صنعتی کنترل دمای اتوماتیک
- صنایع شیمیایی شامل دستگاه های مخلوت شیمیایی
- خدمات ساختمانی شامل آسانسور تهویه هوا و...
- سیستم های حمل و نقل شامل جرثقیل ها سیستم کانوایر و...
- و..

آشنایی با محیط نرم افزار

همانند نرم افزار های دیگر با کلیک روی آیکون آن راه اندازی می شود . و همانطوریکه در شکل S7 نرم افزار ملاحظه میشود دارای یک باریکه شامل چند قسمت از جمله همانند سیستم عامل ویندوز مي باشد :

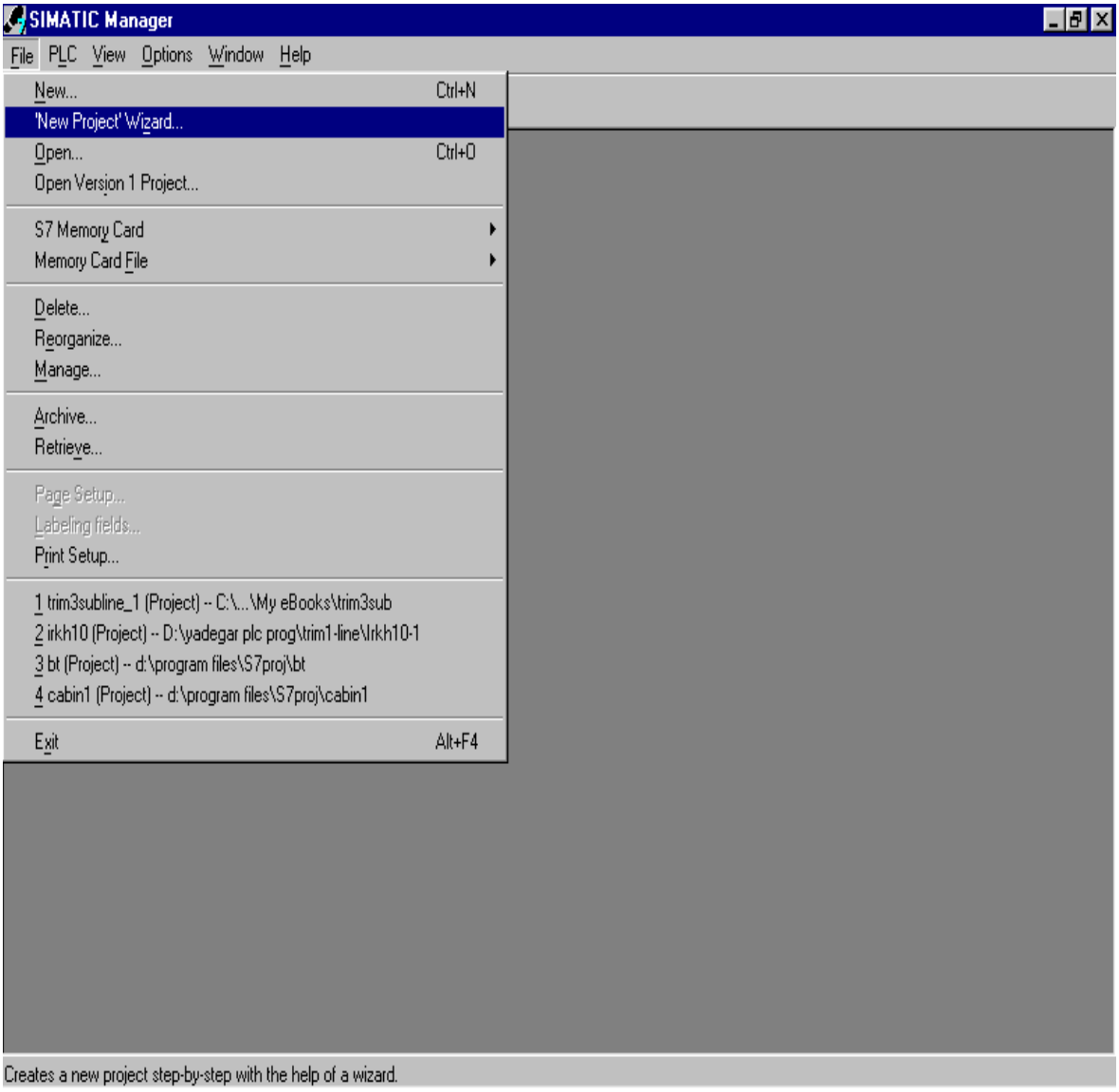
FILE/PLC/VIEW/OPTION/WINDOW/HELP



- **ساختن یک پروژه جدید در برنامه s7**

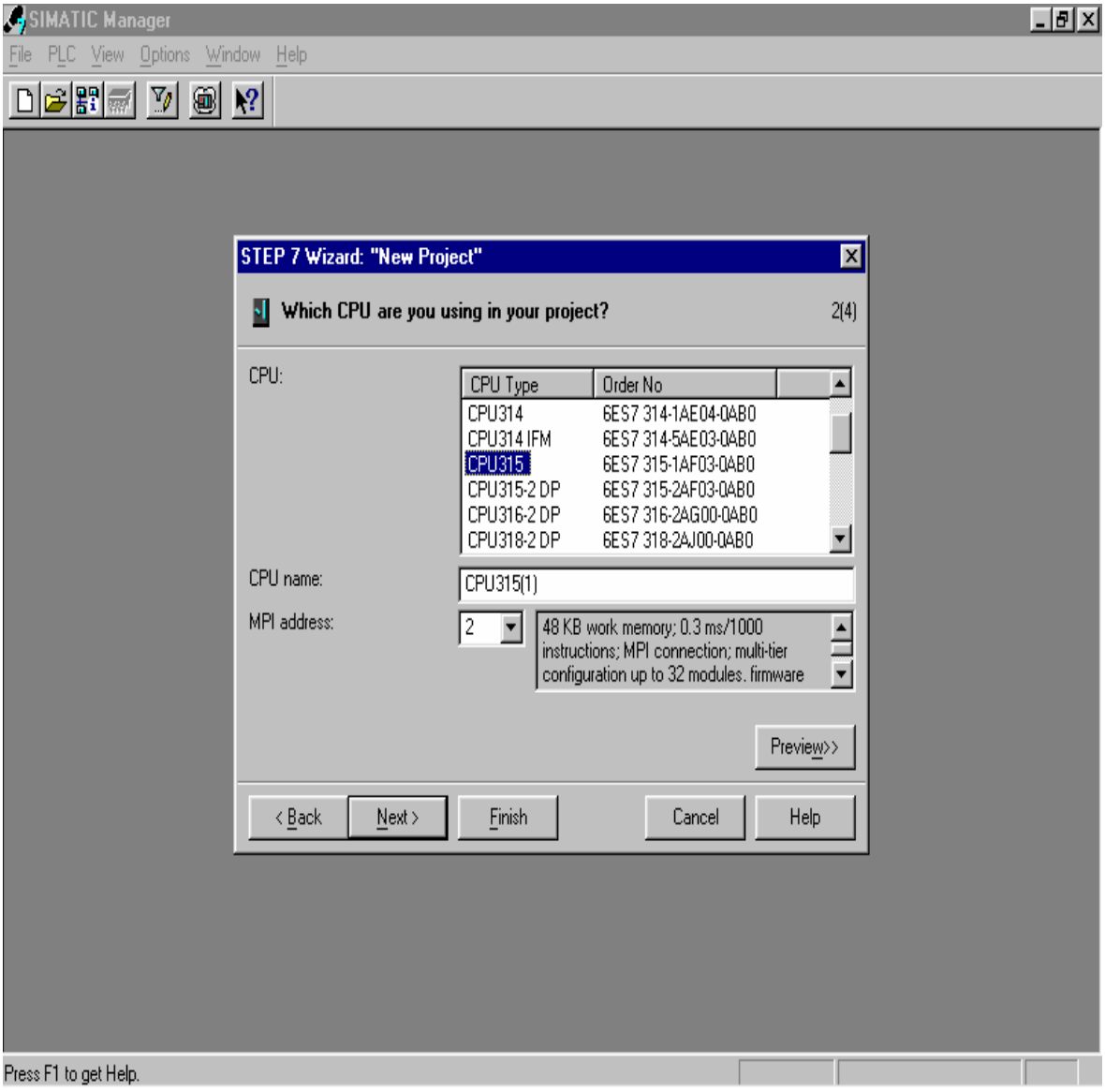
حال با انتخاب منوی FILE یک منوی کرکره ای ظاهر میشود این منو در حقیقت ابتدای کار با نرم افزار S7 مي باشد .

در این مرحله با انتخاب گزینه NEW PROJECT WIZARD وارد صفحه جدیدی خواهیم شد.این منو در حقیقت جایی است که درآن میتوانیم قسمت های سخت افزاری مورد نظر را انتخاب کنیم .



- **نحوه انتخاب سخت افزار پروژه**

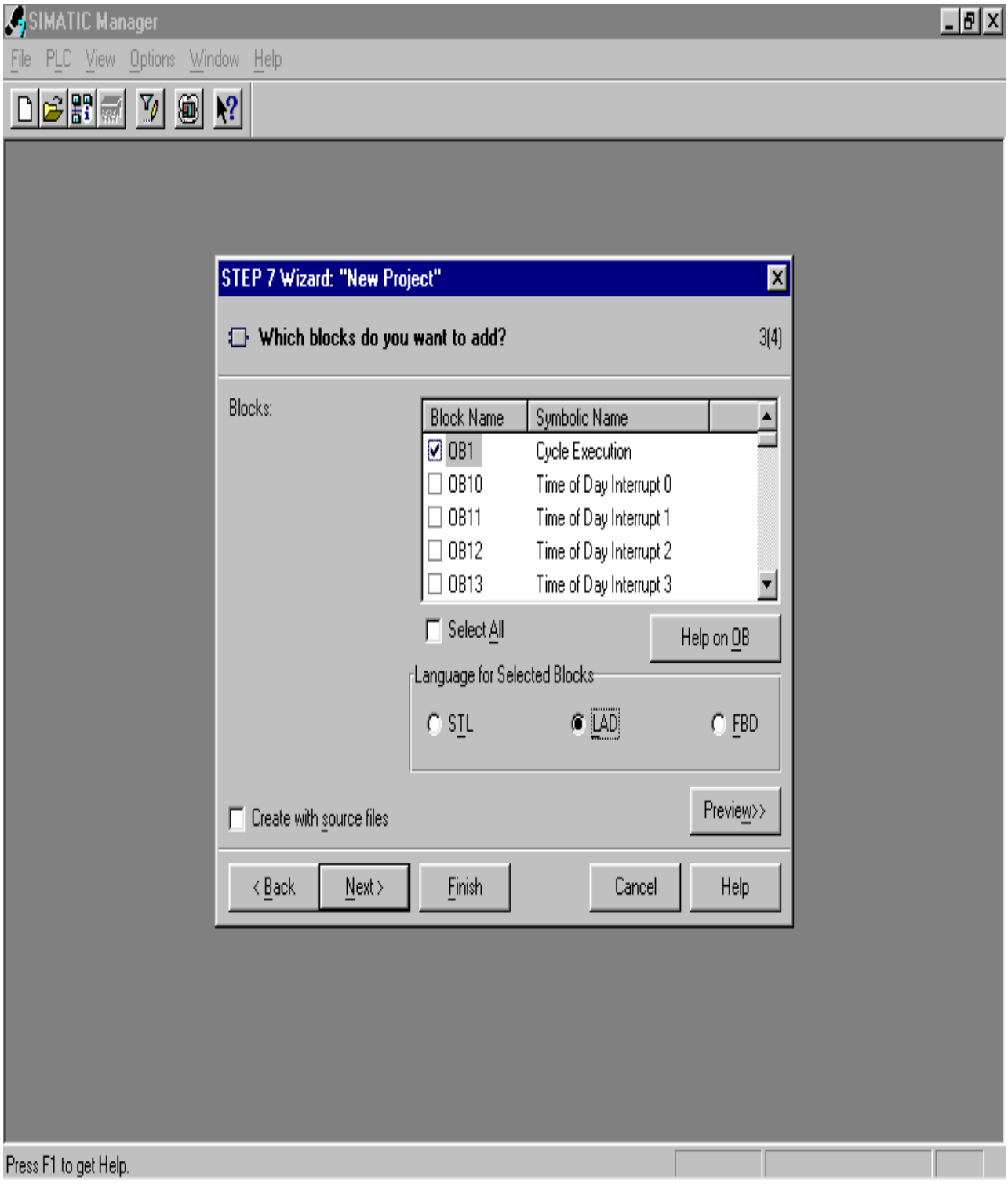
حال میتوانیم CPU مورد نظر را از CPU TYPE های پیشنهادی نرم افزار گزینش کنیم.همچنین در این صفحه به MPI ADDRESS بر میخوریم که مقدار آن را همیشه ۲ در نظر میگیریم. با زدن دکمه NEXT به صفحه بعد مي رويم



در این صفحه ORGANIZATION BLOCK(OB) مورد نظر را گزینش میکنیم.که در این لیست OB های غیر از OB1 نیز تعریف گردیده است که طراح بنا بر احتیاجات خود میتواند از آنها نیز استفاده کند.همچنین در استیل های S7 آمده که کاربر میتواند یکی از سه گزینه LAD,FBD ,STL را بنا به استفاده خود انتخاب کند.

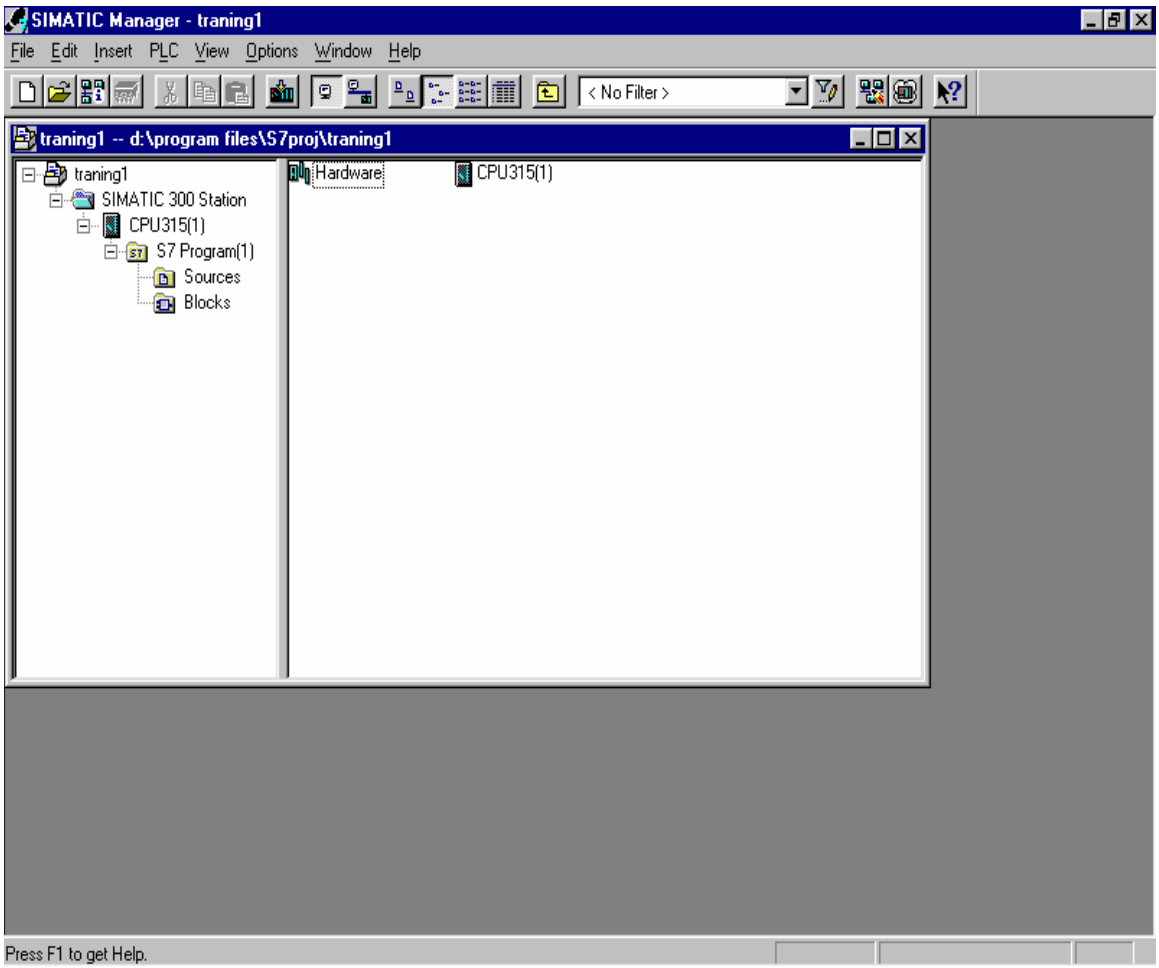
با زدن دکمه FINISH این مرحله به پایان رسیده وصفحه SIMANTIC MANAGE باز میشود.

در صفحه بعد نحوه عملکرد توضیح داده شده در فوق در داخل نرم افزار نمایش داده شده است .


SIMATIC Manager
File PLC View Options Window Help
STEP 7 Wizard: "New Project"
Which blocks do you want to add?
3(4)
Blocks:
Block Name
Symbolic Name
OB1 Cycle Execution
OB10 Time of Day Interrupt 0
OB11 Time of Day Interrupt 1
OB12 Time of Day Interrupt 2
OB13 Time of Day Interrupt 3
Select All
Help on OB
Language for Selected Blocks
STL
LAD
FBD
Create with source files
Preview>>
< Back
Next >
Finish
Cancel
Help
Press F1 to get Help.

- **نحوه کانفیگوریشن و تنظیمات سخت افزار (HARDWARE)**

حال در سمت راست صفحه بلوک های مورد استفاده در برنامه ودر سمت چپ ساختار درختی آنچه در سمت راست انجام شده در پروژه TRAINING 1 دیده می شود. براي رفتن به قسمت SOURCE میتوانیم بقیه سخت افزار های مورد نیاز از جمله کارتهای ورودی و خروجی , کارت های شبکه , منبع تغیه , پردازشگرها و... را از آنچه خود نرم افزار ارایه کرده است گزینش کنیم.برای این منظور باید به HARDWAREرفته والمانها را گزینش کنیم.

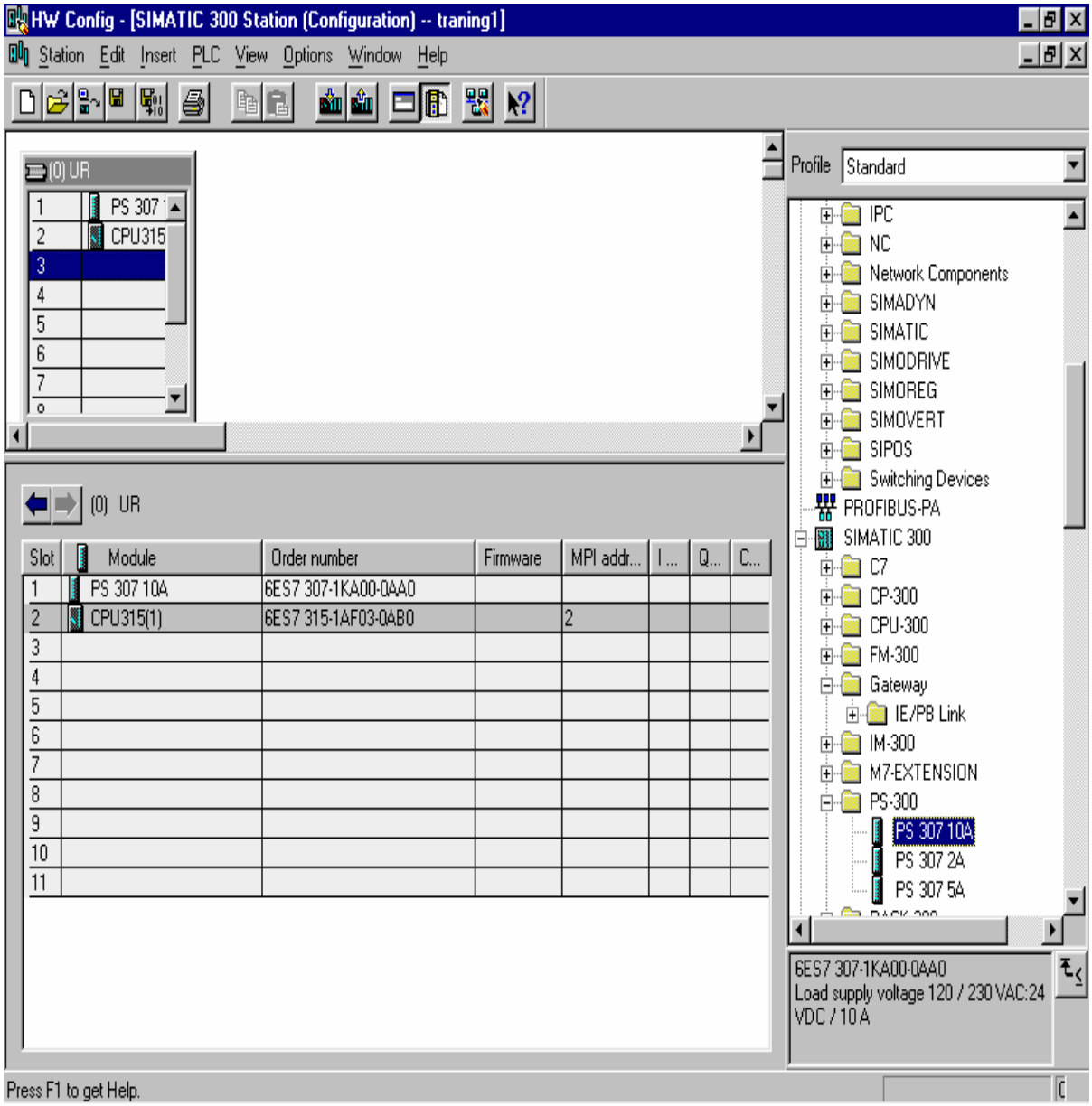


هم اینک صفحه HW CONFIG باز شده است که سمت چپ آن با سخت افزار های انتخابی پر میشود CPU که در ابتدا انتخاب کردیم در SLOT دوم نشسته و ما باید از سمت راست و از کتابخانه آن سخت افزار ها را انتخاب کنیم

.ابتدا باید SIMANTIC300 یا SIMANTIC 400 را انتخاب کنیم.

و سپس المانها را ازآنجا پیدا نماییم .

ما در این مثال محیط SIMANTIC300 را در نظر گرفته ایم.ابتدا PS (POWER SUPPLY) دلخواه را از منوی سمت راست DRAG و در اولین SLOT موجود درجدول DROP میکنیم .یاد آوری این نکته ضروری است که SLOT سوم همیشه خالیست که ما در اینجا307 PS (منبع تغذیه ۵ آمپر) را بر داشتیم



اکنون نوبت انتخاب کارتهای رودی و خروجی است

AI/AO

مربوط به کارتهای آنالوگ است.ANALOG INPUT/OUTPUT .که ما در اینجاکارت AI4/A02 (۴ ورودي آنالوگ, و دو خروجي آنالوگ) را انتخاب کردیم.

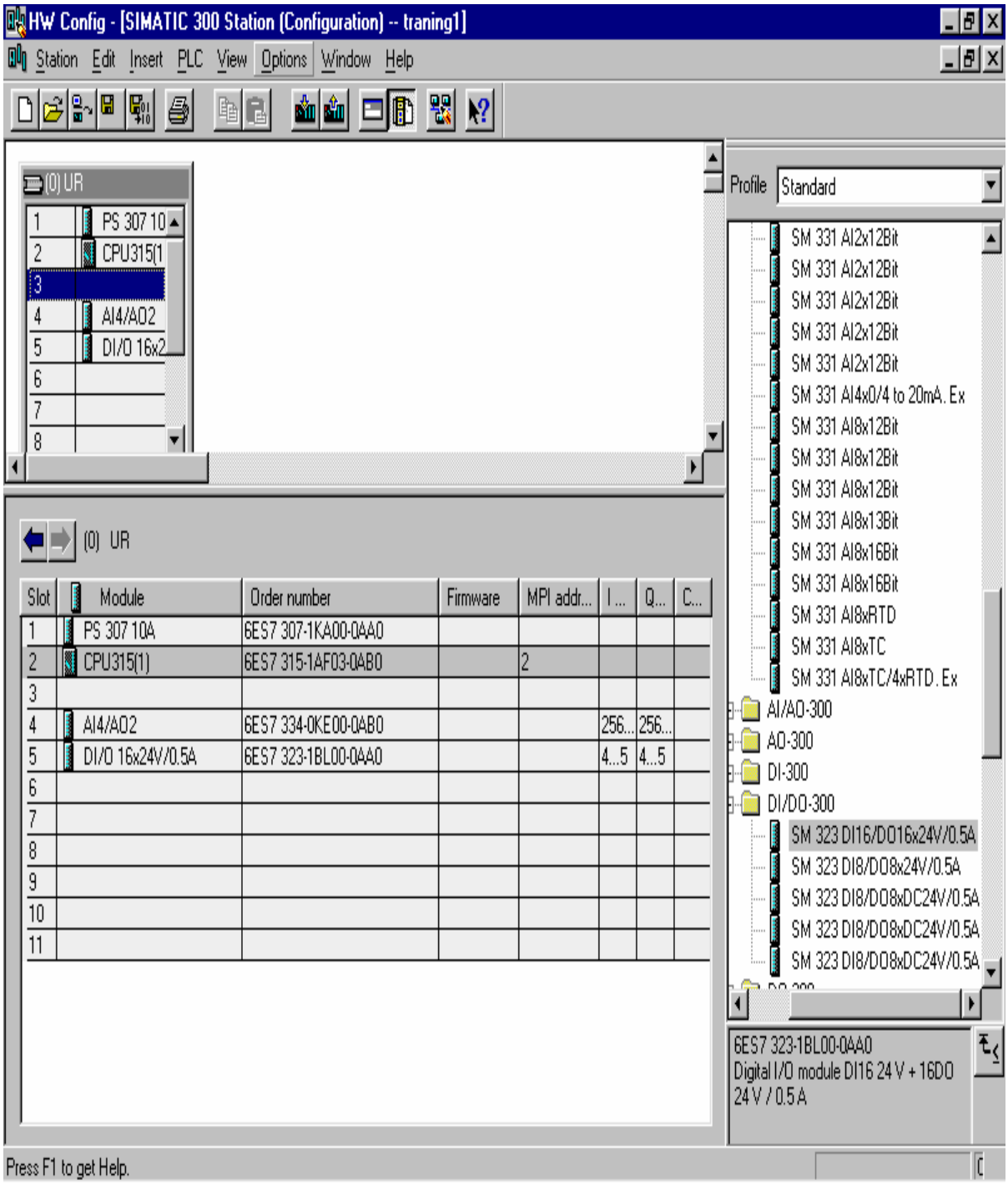
DI/DO

مربوط به کارتهای دیجیتال است. DIGITAL INPUT/OUTPUT .که ما در اینجا کارت 16X12V/.5 A (۱۶ ورودي خروجي دیجیتال ۱۲ ولت نیم آمپر) را انتخاب کردیم.

همچنین در جدول پایینی مشخصات کامل و محل آدرس آنها در حافظه مشهود است.

ORDER NUMBER

شماره مشخص هر المان را نشان میدهد که هر شماره مختص یک المان است.


HW Config - [SIMATIC 300 Station (Configuration) -- traning1]
Station Edit Insert PLC View Options Window Help
(0) UR
1 PS 307 10
2 CPU315(1
3
4 AI4/AO2
5 DI/O 16x2
6
7
8
Profile Standard
SM 331 AI2x12Bit
SM 331 AI2x12Bit
SM 331 AI2x12Bit
SM 331 AI2x12Bit
SM 331 AI2x12Bit
SM 331 AI4x0/4 to 20mA. Ex
SM 331 AI8x12Bit
SM 331 AI8x12Bit
SM 331 AI8x12Bit
SM 331 AI8x13Bit
SM 331 AI8x16Bit
SM 331 AI8x16Bit
SM 331 AI8xRTD
SM 331 AI8xTC
SM 331 AI8xTC/4xRTD. Ex
AI/AO-300
AO-300
DI-300
DI/DO-300
SM 323 DI16/DO16x24V/0.5A
SM 323 DI8/DO8x24V/0.5A
SM 323 DI8/DO8xDC24V/0.5A
SM 323 DI8/DO8xDC24V/0.5A
SM 323 DI8/DO8xDC24V/0.5A
(0) UR
Slot Module Order number Firmware MPI addr... I... Q... C...
1 PS 307 10A 6ES7 307-1KA00-0AA0
2 CPU315(1) 6ES7 315-1AF03-0AB0 2
3
4 AI4/AO2 6ES7 334-0KE00-0AB0 256... 256...
5 DI/O 16x24V/0.5A 6ES7 323-1BL00-0AA0 4...5 4...5
6
7
8
9
10
11
6ES7 323-1BL00-0AA0
Digital I/O module DI16 24 V + 16DO 24 V / 0.5 A
Press F1 to get Help.

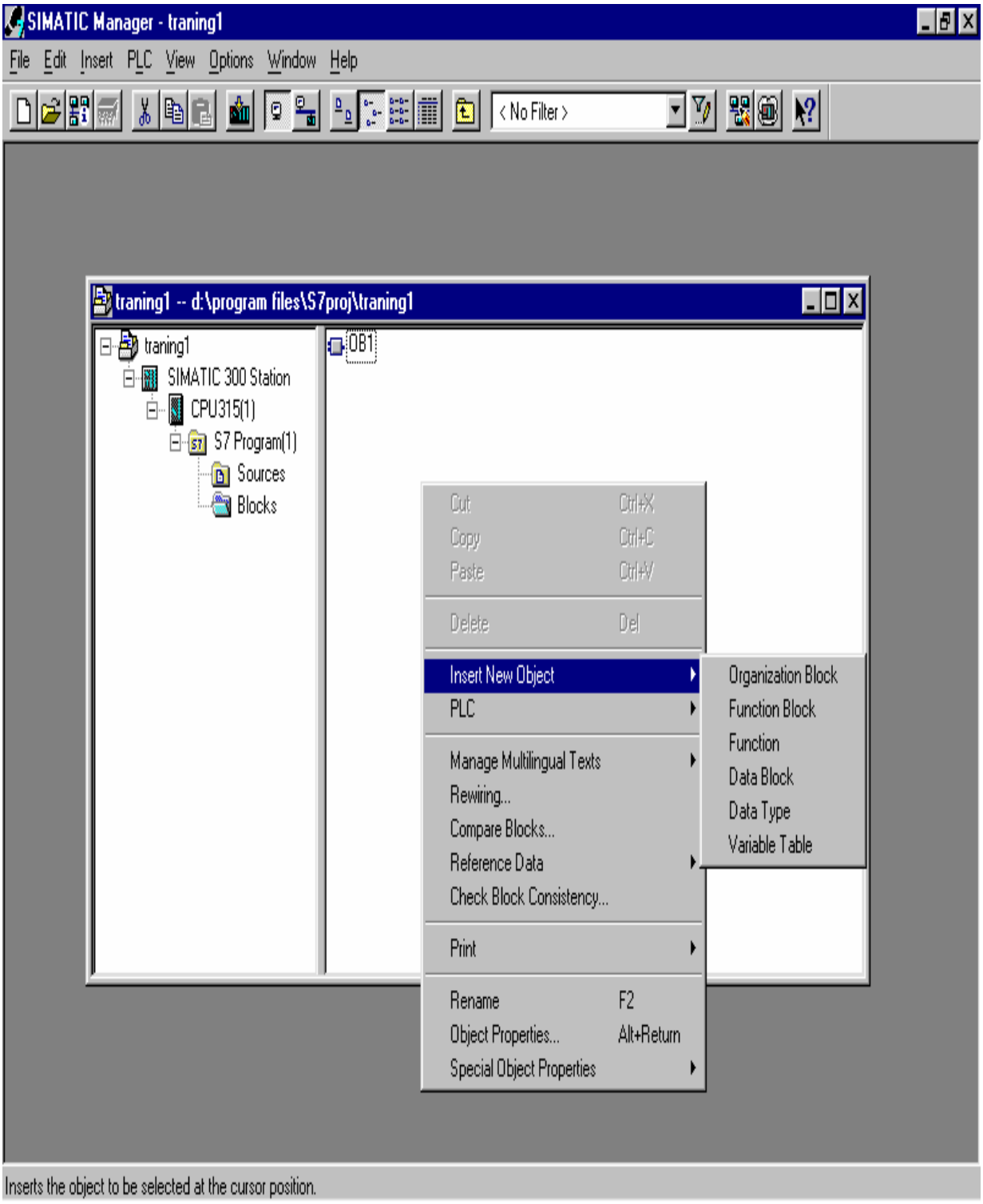
- **ساختن بلوک ها در بر نامه**

همانطور که گفته خواهد شد بلوک های مختلفی با وظایف مشخص در s7 وجود دارد. اکنون می بایست این بلوک ها در بر نامه فراخوانی شوند. برای این منظور ابتدا باید از ساختار درختی سمت چپ BLOCK را انتخاب کنیم. حال با کلیک چپ در سمت راست صفحه ساختار درختی ملاحظه می شود

باانتخاب INSERT NEW OBJECT میتوانیم هر یک از بلوک های :

FUNCTION BLOCK, .DATABLOCK ,
ORGANIZATION BLOCK, VARIABLE TABLE

را براساس احتیاج مطابق شکل زیر بسازیم.


SIMATIC Manager - traning1
File Edit Insert PLC View Options Window Help
< No Filter >
traning1 -- d:\program files\S7proj\traning1
traning1
SIMATIC 300 Station
CPU315(1)
S7 Program(1)
Sources
Blocks
OB1
Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V
Delete Del
Insert New Object
PLC
Manage Multilingual Texts
Rewiring...
Compare Blocks...
Reference Data
Check Block Consistency...
Print
Rename F2
Object Properties... Alt+Return
Special Object Properties
Organization Block
Function Block
Function
Data Block
Data Type
Variable Table
Inserts the object to be selected at the cursor position.

انواع بلوک در S7 :

- **ORGANIZAITION BLOC (OB)**
- **INTEGRATED SPESIAL OBs**
- **SYSTEM FUNCTION BLOCKS(SFBs)**
- **SYSTEM FUNCTIONS(SFC)**
- **FUNCTION BLOCK (FC)**
- **DATA BLOCK (DB)**
- **SYSTEM DATA BLOCK (SDB)**

ORGANIZATION BLOCK –OB :

در حقیقت OB توصیف کننده وظایف هر قسمت است و به عبارت دیگر واسطه بین استفاده کننده از سیستم و نرم افزار است . مهم ترین قسمت در بلوک ها OB است.نکته قابل توجه در OB ها انواع و تقسیم بندی آنها است.

OB1 :

بلوک اصلی موجود است . وOB های دیگر به عنوان وقفه عمل کرده و وارد سیکل اجرای بر نامه شده و برنامه را برای اجرای OB مورد نظرآماده میکند.
انواع OB در برنامه و عملکرد آنهاو شماره آنها در جداول پایین آمده است
سمت راست شماره آنها و تفاوتهایشان در S7, S5 آمده است.
نکته مهم در OB ها PRIORITY آنهاست بدین گونه که هر گاه دو OB همزمان فراخوانی شوند آن OB که دارای PRIORITY بیشتری باشد در اجرا اولویت دارد.
مقادیر آنها در زیر آمده است.

OB10-OB23 که در آنها PRIORITY 2-23 است.
OB70-OB72 که درآنها PRIORITY 23-26 است.

و...

Function	S5 Block	Replacement in S7
Cycle time triggering	OB31	SFC43 RE_TRIGR
Battery failure	OB34	OB81 (Error reaction can be programmed by user)
Access to condition code byte	OB110	STEP 7 instruction: L STW/T STW
Delete ACCU 1 – 4	OB111	STEP 7 instruction sequence: L 0; PUSH; PUSH; PUSH
Roll up ACCU	OB112	Function not identical: STEP 7 instruction: PUSH
Roll down ACCU	OB113	Function not identical STEP 7 instruction: POP
Disable all interrupts on/off	OB120	SFC41 DIS_AIRT SFC42 EN_AIRT
Disable cyclic (timed) interrupts individually on/off	OB121	SFC39 DIS_IRT SFC40 EN_IRT
Delay all interrupts on/off	OB122	SFC41 DIS_AIRT SFC42 EN_AIRT
Delay cyclic (timed) interrupts individually on/off	OB123	SFC39 DIS_IRT SFC40 EN_IRT
Set/read CPU time *(continued on next page)*	OB150	SFC0 SET_CLK SFC1 READ_CLK

که برنامه اصلی در OB ۱ ها و برنامه های که در طول اجراي برنامه دايمآ فراخوانی می شوند در FC (فانکشن) و (فانکشن بلاک) FB ريخته می شوند .

فرمهای نوشتن برنامه در S7 :

۱) فرم نردبانی(LADDER)

۲) فرم نرم افزاری (STATEMENT LIST)

۳) فرم بلوکی (FUNCTION BLOCK DATA)

فرم نردبانی(LAD):

هر گاه در بر نامه المانها به صورت مدارات پاراللل رسم گرددند.یا به عبارت دیگر به صورت قطعات الکترونیکی موجود در کتابخانه قرار گیرند در اینصورت مدار به صورت LADDER بسته شده است.

حال باید در ابتدا با فرمان ها و المانهای موجود در کتابخانه S7 آشنا شویم تا بتوانیم از چگونگی کاربرد آنها در مدارات منطقي آگاه شویم:

بیت هاي لاجیک

: (Normally open contact) ---| |---

کنتاکتور در حالت عادی باز که برای هر کنتاکتور به صورت زیر آدرس دهی میکنیم.

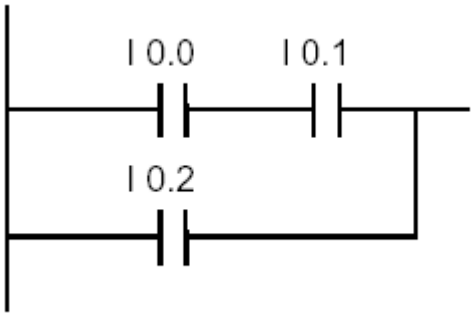
I 0.1 که از سمت چپ میخوانیم عدد اول مربوط به آدرس بایت و عدد دوم مربوط به بیت اشغال شده از حافظه توسط این کنتاکتور است یعنی در انجا آدرس بایت ۰ وبیت ۱ از حافظه اشغال شده است.

برای هر کدام از این المانها جدولی است که حاوی اطلاعاتی در مورد نوع DATA که توسط این آدرس داده میشود و MEMORY AREA آن است .که بعضی از این جداول را در پایین خواهیم آورد.

<address>
---| |---

Parameter	Data Type	Memory Area	Description
<address>	BOOL	I, Q, M, L, D, T, C	Checked bit

مثال :



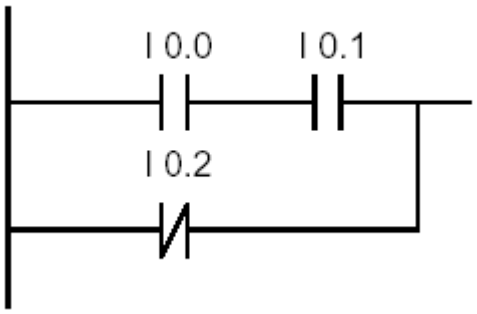
---| / |--- (Normally closed contact) :

کنتاکوری که در حالت عادی بسته است. آدرس دهی آن نیز مانند بالاست .تفاوت این دو کنتاکتور در اینست که کنتاکتور بالایی در زمانیکه بیت RLO یک است فعال میشود اما در پایینی وقتی ۰ است یا به عبارت دیگر کنتاکتور پایینی با ۰ فعال و بالایی با آمدن ۱ فعال میشود.

<address>
---| / |---

Parameter	Data Type	Memory Area	Description
<address>	BOOL	I, Q, M, L, D, T, C	Checked bit

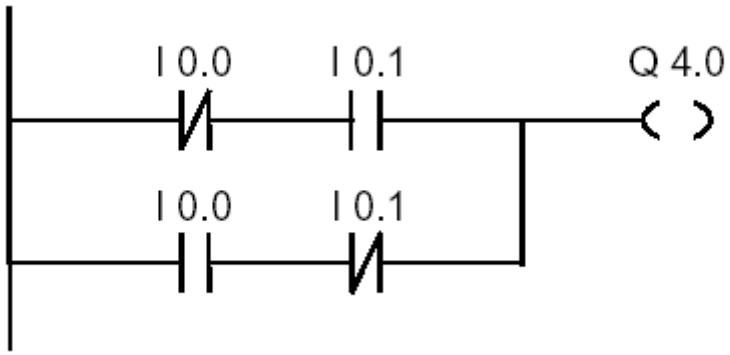
مثال:



XOR (Xclusive OR) :

در مدار XOR جریان یا از شاخه بالایی و یا از پایینی باید برقرار شود.

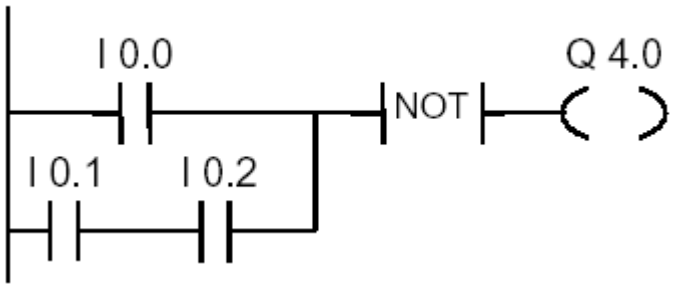
این مدار شامل هر دو کنتاکتور باز و بسته است.



که نتیجه سیگنال در خروجی مورد نظر که در اینجا آدرس Q4.0 است ذخیره میشود.

---|NOT|--- **بیت RLO را منفی میکند.**

مثال:



(OUTPUT COIL) ---( )

به این صورت کار میکند که اگر بیت RLO یک شود ادرس این COIL نیز 1 میشود . و اگر هم RLO صفر باشد بیت آدرس نیز صفر میشود.

مثال:

خروجي Q4.0 در یکی از حالات زیر ۱ میشود:

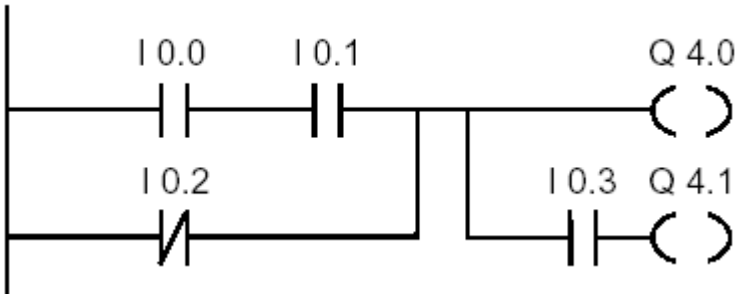
سیگنال ۱ باشد و AND I0.0 , I0.1 رخ دهد.

سیگنال ۰ باشد و جریان I0.2 را عبور دهد.

و یا خروجي Q4.1 زمانی ۱ میشود که:

سیگنال ۰ باشد وI0.2 صفر باشد و I0.3

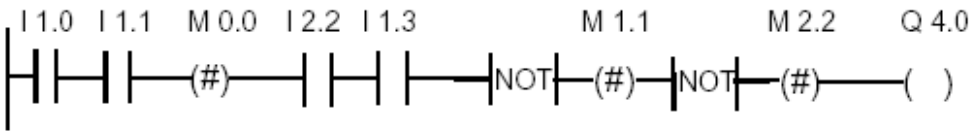
سیگنال ۱ باشد و AND I0.1,I0.0 رخ دهد و یک باشد.

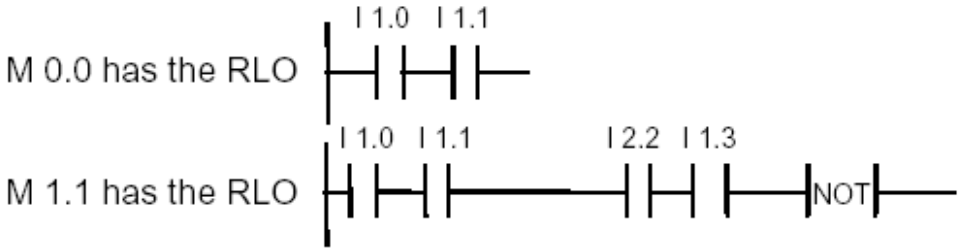


---(#)---

المانی که بیت RLO را در آدرس مورد نظر SAVE مي كند .

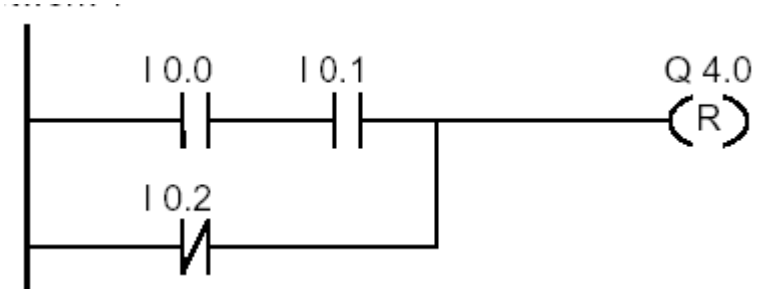
مثال:





---(R)

بیت RLO را RESET میکند یعنی اگر ۱ باشد ۰ و اگرصفر باشد یک میکند.

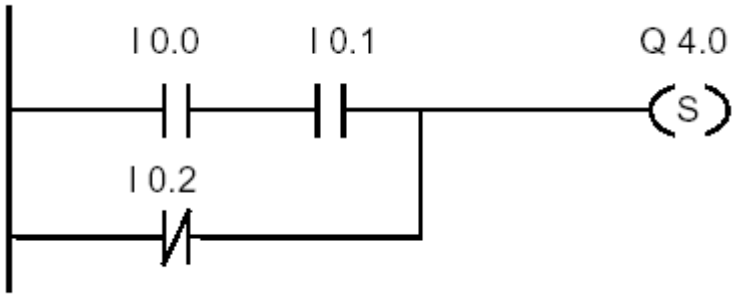


Parameter	Data Type	Memory Area	Description
<address>	BOOL	I, Q, M, L, D, T, C	Reset bit

---(S)

خروجی سیگنال هر چه که باشد SET میکند یعنی ۱ می کند.

مثال:



Parameter	Data Type	Memory Area	Description
<address>	BOOL	I, Q, M, L, D	Set bit

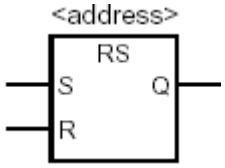
: FLIP FLOP

برای ثابت نگه داشتتن ورودی در حالت صفر و یا یک به فلیپ فلاپ احتیاج داریم. اصطلاحا بعنوان یک تیغه خود نگه دار عمل مي كند .

: RESET SET FLIP FLOP(RS)

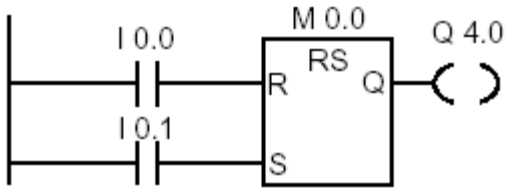
در این فلیپ فلاپ هنگامی تغییر رخ میدهد که بیت RLO ۱ باشد و زمانیکه ۰ باشد هیچ تغییری روی مقادیر SET , RESET رخ نمی دهد.

در جدول پایین دیاگرام و مقادیر یک فلیپ فلاپ را مشاهده مي كنيم



Parameter	Data Type	Memory Area	Description
<address>	BOOL	I, Q, M, L, D	Set or reset bit
S	BOOL	I, Q, M, L, D	Enabled reset instruction
R	BOOL	I, Q, M, L, D	Enabled reset instruction
Q	BOOL	I, Q, M, L, D	Signal state of <address>

مثال:



اگر I0.0 یک باشد و I0.1 صفر باشد بیت حافظه M0.0 SET می شود.و Q4.0 مقدار ۰ را به خود می گیرد.

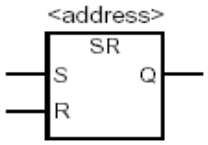
نکته :

(در نرم افزار S7 براي اختصاص فضاي حافظه داخلي از اصطلاح M و آدرس دهي مربوطه استفاده مي گردد و در نوع از Cpu ها تعداد آدرس هاي حافظه M که مي توان استفاده نمود مشخص گرديده و از طريق گرفتن REFERENCE در نرم افزار مي توانيم آنرا بفهميم)

اگر I0.0 صفر باشد و I0.1 یک باشد بیت M0.0 RESET میشود و Q4.0 مقدار ۱ را به خود می گیرد.

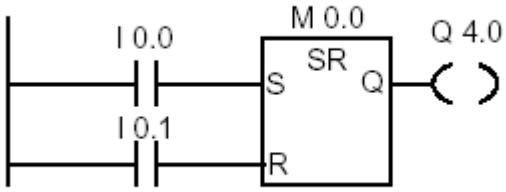
اگر هر ۲ صفر باشند هیچ تغییری رخ نمی دهد واگر هر دو ۱ باشند. مقدار SET بعنوان مقدم در نظر گرفته مي شود .

: SET RESET FLIP FLOP(SR)



Parameter	Data Type	Memory Area	Description
<address>	BOOL	I, Q, M, L, D	Set or reset bit
S	BOOL	I, Q, M, L, D	Enable set instruction
R	BOOL	I, Q, M, L, D	Enable reset instruction
Q	BOOL	I, Q, M, L, D	Signal state of <address>

مثال:



اگر I0.0 یک باشد و I0.1 صفر باشد بیت M0.0 SET می شود.و Q4.0 مقدار ۱ را به خود می گیرد.

اگر I0.0 صفر باشد و I0.1 یک باشد بیت M0.0 RESET میشود و Q4.0 مقدار ۰ را به خود می گیرد.

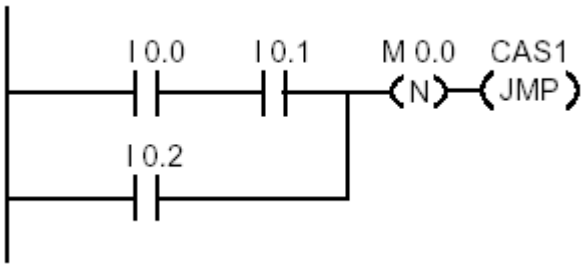
اگر هر ۲ صفر باشند هیچ تغییری رخ نمی دهد واگر هر دو ۱ باشند. مقدار RESET ارجح قرار مي گيرد .

---(N)---

زمانیکه آدرس از ۱ به ۰ تغییر می یابد بیت RLO را یک می کند.

Parameter	Data Type	Memory Area	Description
<address>	BOOL	I, Q, M, L, D	Edge memory bit, storing the previous signal state of RLO

مثال:



بیت حافظه M0.0 در RLO قدیم SAVE شده است هنگامی که سیگنال تغییر کند و سیگنال از ۱ به ۰ تغییر کند

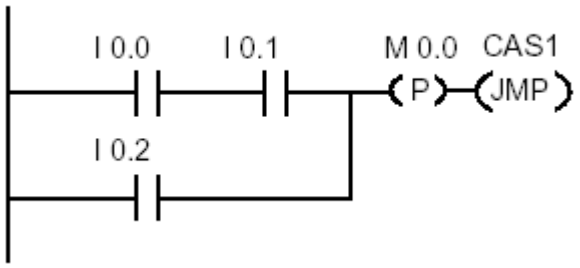
برنامه به آدرس CAS1 پرش می کند.

---(P)---

زمانیکه آدرس از ۰ به ۱ تغییر می یابد بیت RLO را یک می کند.

Parameter	Data Type	Memory Area	Description
<address>	BOOL	I, Q, M, L, D	Edge memory bit, storing the previous signal state of RLO

مثال:

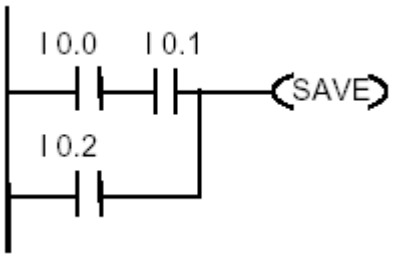


بیت M0.0 در RLO قدیم SAVE شده است هنگامی که سیگنال از ۰ به ۱ تغییر کند برنامه به آدرس CAS1 پرش می کند.

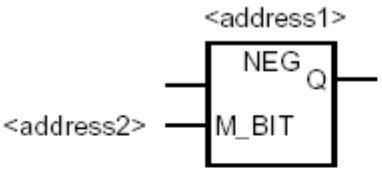
---(SAVE)

اطلاعات RLO را در BR MEMORY ذخیره می کند.

مثال:



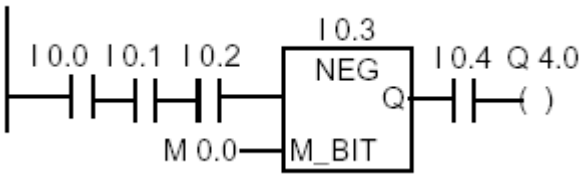
: NEG (NEGATIV EDG DETECTION)



Parameter	Data Type	Memory Area	Description
<address1>	BOOL	I, Q, M, L, D	Scanned signal
<address2>	BOOL	I, Q, M, L, D	M_BIT edge memory bit, storing the previous signal state of <address1>
Q	BOOL	I, Q, M, L, D	One shot output

محتویات RLO قدیم و جدید را مقایسه میکند و در صورتیکه RLO جدید ۱ و RLO قدیمی ۰ باشد ، بیت RLO را ۱ میکند.

مثال:



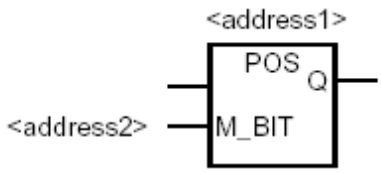
در I0.3 لبه پایین رونده داریم.

I0.0 و I0.1 و I0.2 سیگنال هاي ۱ مي باشند

با لبه پایین رونده در I0.4 سیگنال ۱ گردیده و خروجي Q4.0 یک مي گردد .

---(P)---

(POSITIVE EDGE DETECTION)

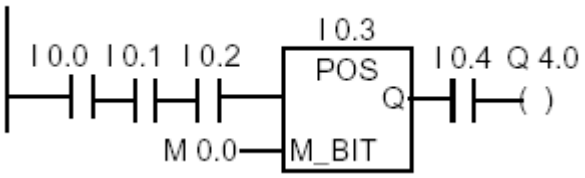


Parameter	Data Type	Memory Area	Description
<address1>	BOOL	I, Q, M, L, D	Scanned signal
<address2>	BOOL	I, Q, M, L, D	M_BIT edge memory bit, storing the previous signal state of <address1>
Q	BOOL	I, Q, M, L, D	One shot output

محتویات RLO قدیم و جدید را مقایسه میکند و در صورتیکه RLO جدید 1 و RLO قدیمی 0 باشد ، بیت RLO را 0 میکند.

با لبه بالا رونده عمل مي نمايد و حساس به سطح سيگنال نمي باشد و زماني كه سيگنالي از حال صفر به حالت يک تغيير حالت يابد با اولين لبه بالارونده اين بلوک عمل مي نمايد

مثال:



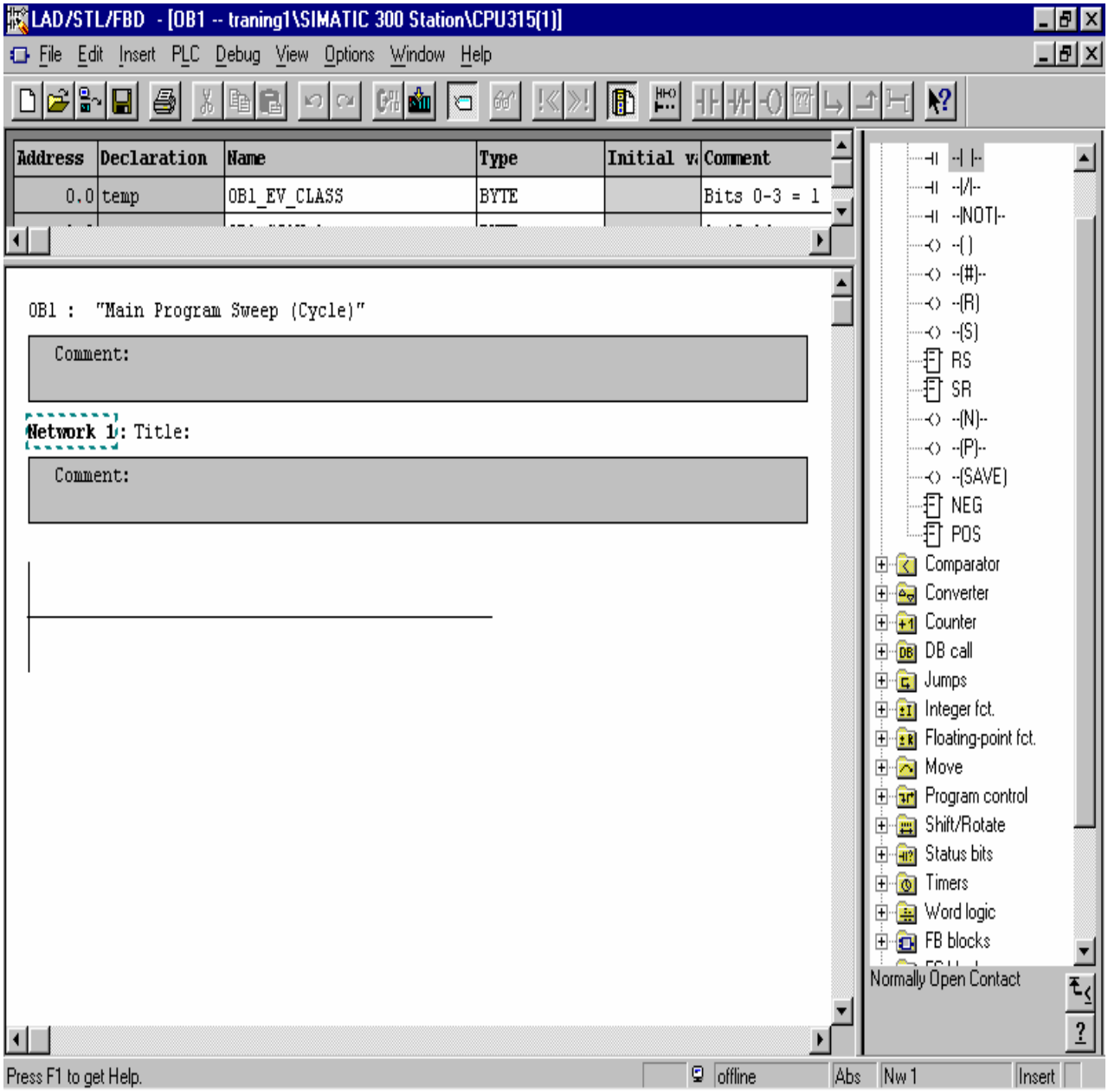
در I0.3 لبه بالا رونده داريم.

در صورتيكه I0.0 و I0.1 و I0.2 به سيگنال ۱ تغيير حالت بدهند .

دراين حالت در I0.4 و خروجي Q4.0 سيگنال يک مي گردد .

نكته :

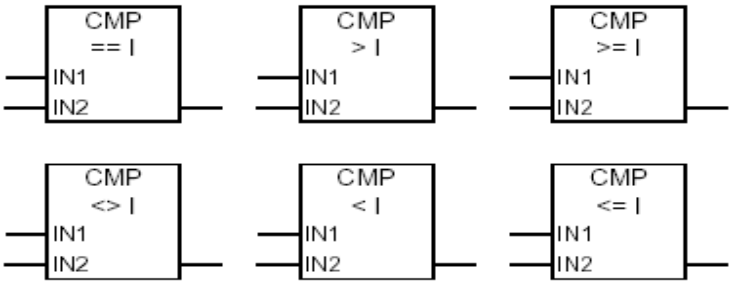
در شكل زير در منوي سمت چپ كه كتابخانه S7 مي باشد تمامي دستورات فوق بصورت گرافيكي نمايش داده شده است كه مي توان براحتي بوسيله موس و با روش DRAG AND DROP آنها را به داخل برنامه و در هر NETWORK دلخواه كه با باز كردن يک FC بعنوان مثال بصورت اتوماتيک ساخته مي شود قرار داد .


LAD/STL/FBD - [OB1 -- traning1\SIMATIC 300 Station\CPU315(1)]
File Edit Insert PLC Debug View Options Window Help
Address Declaration Name Type Initial v Comment
0.0 temp OB1_EV_CLASS BYTE Bits 0-3 = 1
OB1 : "Main Program Sweep (Cycle)"
Comment:
Network 1: Title:
Comment:
--| |--
--|/|--
--|NOT|--
--( )
--(#)--
--(R)
--(S)
RS
SR
--(N)--
--(P)--
--(SAVE)
NEG
POS
Comparator
Converter
Counter
DB call
Jumps
Integer fct.
Floating-point fct.
Move
Program control
Shift/Rotate
Status bits
Timers
Word logic
FB blocks
Normally Open Contact
Press F1 to get Help.
offline
Abs
Nw 1
Insert

مقایسه کننده ها (COMPARATOR) :

(CMPI) COMPARE INTEGER

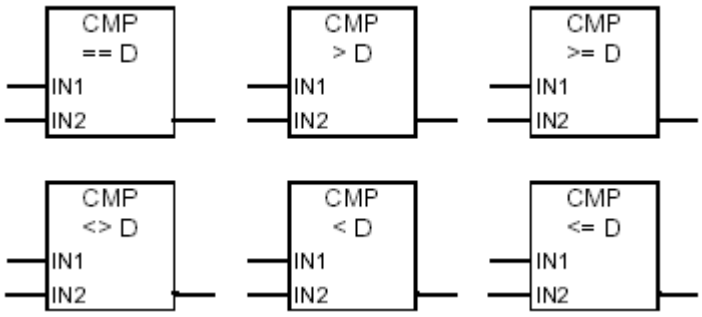
مقایسه کننده فوق براي مقایسه دو عدد INTEGER (اعداد صحیح بدون اعشار) استفاده مي گردد که انواع مختلف کوچکتر مساوي ، بزرگتر مساوي و مساوي را مي توان انتخاب نمود



Parameter	Data Type	Memory Area	Description
box input	BOOL	I, Q, M, L, D	Result of the previous logic operation
box output	BOOL	I, Q, M, L, D	Result of the comparison, is only processed further if the RLO at the box input = 1
IN1	INT	I, Q, M, L, D or constant	First value to compare
IN2	INT	I, Q, M, L, D or constant	Second value to compare

:(CMP D) COMPARE DOUBLE INTEGER

براي مقايسه اعداد INTEGER كه بزرگ بوده و بيش از دو WORD براي ذخيره آنها در حافظه مورد نياز مي باشد



Parameter	Data Type	Memory Area	Description
box input	BOOL	I, Q, M, L, D	Result of the previous logic operation
box output	BOOL	I, Q, M, L, D	Result of the comparison, is only processed further if the RLO at the box input = 1
IN1	DINT	I, Q, M, L, D or constant	First value to compare
IN2	DINT	I, Q, M, L, D or constant	Second value to compare

- *يادآوري مدارات منطقي*

0 AND 1 =BIT

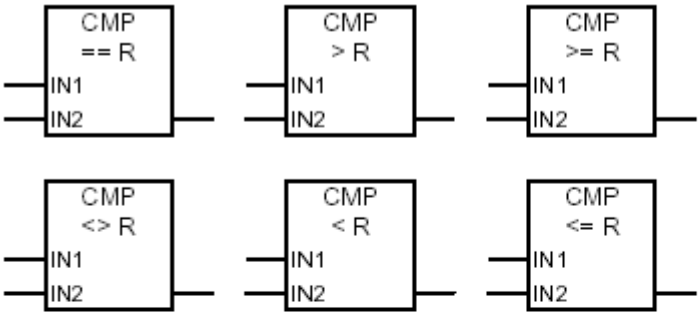
8BIT =BYTE

2 BYTE =WORD

4BYTE OR 2 WORD =DOUBLE WORD

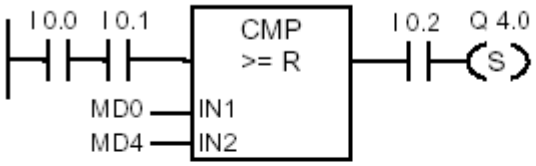
COMPARE REAL(CMP R)

براي مقايسه اعداد اعشاري استفاده مي گردد



Parameter	Data Type	Memory Area	Description
box input	BOOL	I, Q, M, L, D	Result of the previous logic operation
box output	BOOL	I, Q, M, L, D	Result of the comparison, is only processed further if the RLO at the box input = 1
IN1	REAL	I, Q, M, L, D or constant	First value to compare
IN2	REAL	I, Q, M, L, D or constant	Second value to compare

مثال :



Output Q4.0 is set if the following conditions exist:

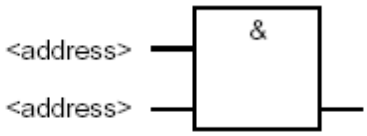
- There is a signal state of "1" at inputs I0.0 and at I0.1
- And MD0 >= MD4
- And there is a signal state of"1" at input I0.2

از آنجایی که در برنامه نویسی با s7 بعلت نزدیک بودن روش به مدارات منطقي معمولاً از FBD (FUNCTION BLOCK DIAGRAM) استفاده می کنیم و کاربرد المانها در تمام فرمها یکی است.لذا ادامه LIBRARY را در FBD ادامه می دهیم.

: FUNCTION BLOCK DIAGRAM (FBD)

: AND(&)

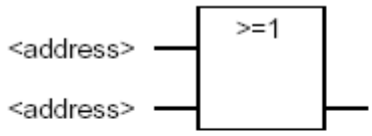
آدرس ها را در محل نشان داده شده می نویسیم و از بلوک AND استفاده می کنیم.



Parameter	Data Type	Memory Area	Description
<address>	BOOL	I, Q, M, T, C, D, L	The address indicates the bit whose signal state will be checked.

OR(>=1) :

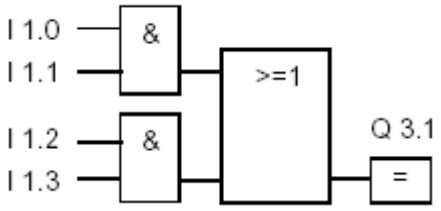
آدرس ها را در محل نشان داده شده می نویسیم و از بلوک AND استفاده می کنیم.



Parameter	Data Type	Memory Area	Description
<address>	BOOL	I, Q, M, T, C, D, L	The address indicates the bit whose signal state will be checked.

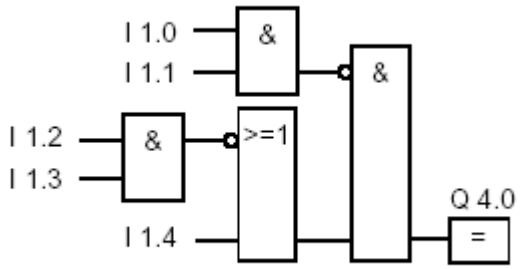
ترکیب AND ,OR :

I1.0,I1.1 با هم AND میشوند و I1.2,I1.3 نیز با هم AND میشوند نتایج با هم OR و در Q3.1 ریخته می شود.



: NEGATE BINARY INPUT(⊣/⊢)

هر گاه انجام نشدن یک رخداد برای ما مطرح باشد در این صورت از این فرم استفاده می کنیم. يعني اينكه بخواهيم معكوس يک ورودي (NOT) را به سيستم بدهيم .



مفقوم بالا اينكه (I2.3 , I1.2 AND) يک نشود با I1.4 OR شوند . NOT

جواب AND بلوک اول (بالايي) و بلوک دوم AND(پاييني) ميشوند

و جواب در Q4.0 ريخته ميشود.

: ADRESSING

آدرس دهي در s7

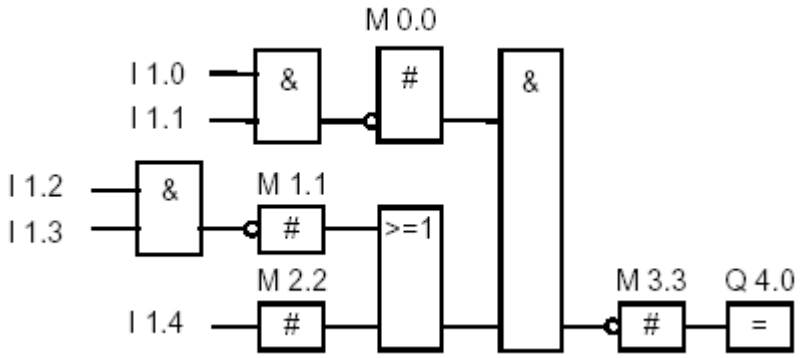
Parameter	Data Type	Memory Area	Description
<address>	BOOL	I, Q, M, D, L	The address specifies the bit to which the signal state of the string of logic operations is assigned.

در s7 مقداری به ADDRESS یک بلوک تخصیص داده می شود.
که در بالا این آدرس Q4.0 است.

: MIDLINE OUTPUT (#)

هدف آن SAVE کردن بیت RLO است.

مثال:



M0.0 منفی(NEGATIVE) RLO بلوک اول را SAVE میکند.
M1.1 منفی(NEGATIVE) RLO بلوک دوم را SAVE میکند.
M2.2 : RLO بیت I1.4 را SAVE میکند.
M3.3 منفی(NEGATIVE) RLO بلوک آخر را SAVE میکند.

: RESET (R)

همانند آنچه درروش LADD گفته شد.

: SET(S)

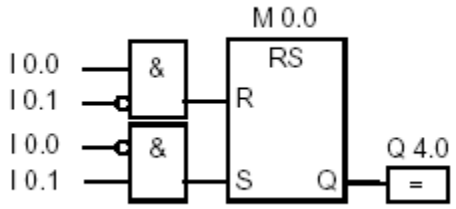
همانند آنچه درروش LADD گفته شد.

فليپ فلاپ :

: RESET_SET FIP FLOP

توضيحات همانند گذشته كه در روش ladder توضيح داده شده مي باشد .

مثال:



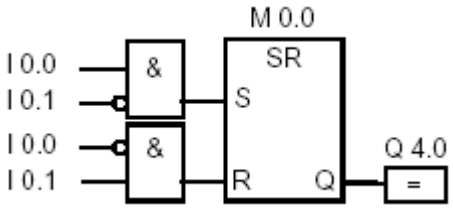
اگر I0.0 یک و I0.1 صفر باشد بیت M0.0 RESET و Q4.0 صفر است.

اگر I0.0 یک و I0.1 صفر باشد بیت M0.0 SET و Q4.0 یک است.

اگر هر دو صفر باشند اتفاقی نمی افتد.

اگر هر دو یک باشند SET رخ می دهد.Q4.0 یک میشود.

: SET_RESET FLIP FLOP

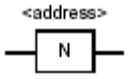


اگر I0.0 یک و I0.1 صفر باشد بیت M0.0 SET و Q4.0 یک است.

اگر I0.0 یک و I0.1 صفر باشد بیت M0.0 RESET و Q4.0 صفر است.

اگر هر دو صفر باشند اتفاقی نمی افتد.

اگر هر دو یک باشند RESET رخ می دهد.Q4.0 یک میشود.

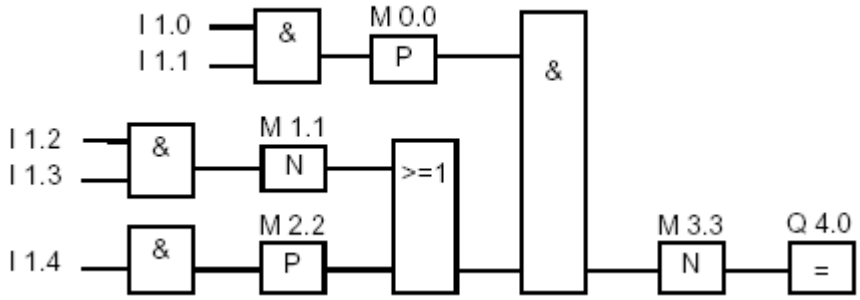


NEGETIVE RLO EDG

POSITIVE RLO EDG

آشکار سازهاي لبه بالارونده و پايين رونده بوده که توضيحات در روش قبلي ارسال گرديده است

مثال :

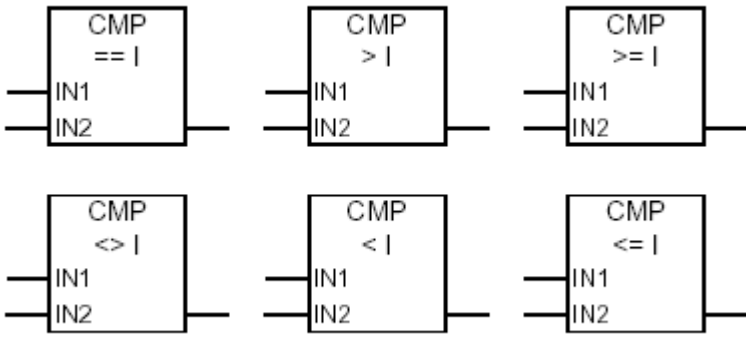


در N اگر از ۱ به ۰ برویم RLO را ۱ میکند.
در P اگر از ۰ به ۱ برویم RLO را ۰ میکند.

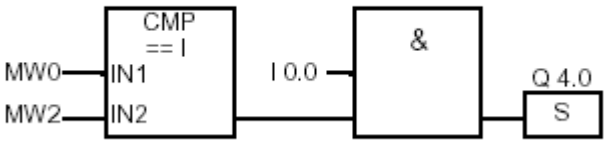
مقایسه کننده ها در FBD :

: (COMPARE INTEGER) CMPI

توضیحات همانند قبل .



مثال:

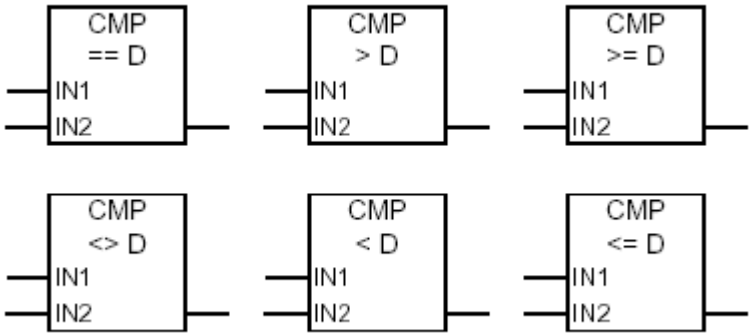


SET Q4.0است زمانی که MWO,MW1 برابر باشند وبا I0.0 AND شوند.

نکته :

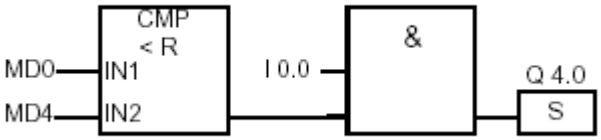
MW (MEMORY WORD) به حافظه هاي بلاتر از يک BIT حتي يک BYTE اطلاق مي گردد که داراي دو بايت مي باشد و براي ذخيره مقادير بزرگ و يا اعداد اعشاري کاربرد دارد

: CMPD



توضیحات همانند قبل.

: CMPR



MD0 از MD4 کوچکتر باشد I0.0 یک باشد

در این صورت Q4.0 SET است.

نکته :

MD (MEMORY DOUBLE) به حافظه داراي 2WORD اطلاق مي گردد

تبدیل کننده ها:

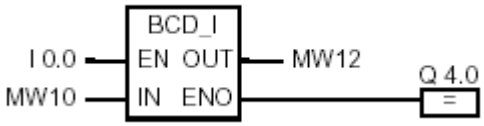
CONVERSION INSTROCTIONS

BCD_I :

مبدل تبدیل اعداد باینري به فرمت INTEGER BCD مي باشد

Parameter	Data Type	Memory Area	Description
EN	BOOL	I, Q, M, D, L, T, C	Enable input
IN	WORD	I, Q, M, D, L or constant	Number in BCD format
OUT	INT	I, Q, M, D, L	Integer value of the BCD number
ENO	BOOL	I, Q, M, D, L	Enable output

مثال:

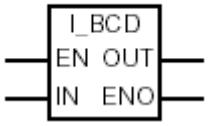


در مثال فوق

MW10 به صورت یک عدد BCDخوانده میشود وتبدیل به INTEGER می شود. و نتیجه در MW12 ریخته می شود.در صورت انجام کار Q4.0 یک میشود. (ENO=EN).

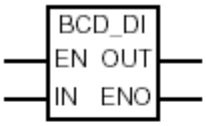
: I_BCD

Parameter	Data Type	Memory Area	Description
EN	BOOL	I, Q, M, D, L, T, C	Enable input
IN	INT	I, Q, M, D, L or constant	Integer
OUT	WORD	I, Q, M, D, L	BCD value of the integer
ENO	BOOL	I, Q, M, D, L	Enable output



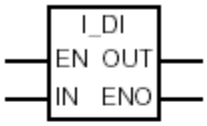
اين بلوک ورودي بصورت INTEGER بوده و خروجي بصورت BCD INTEGER مي باشد

: BCD TO DOUBLE INTEGER(BCD_DI)



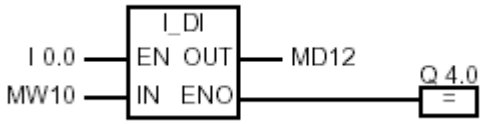
Parameter	Data Type	Memory Area	Description
EN	BOOL	I, Q, M, D, L, T, C	Enable input
IN	DWORD	I, Q, M, D, L or constant	Number in BCD format
OUT	DINT	I, Q, M, D, L	Double integer value of the BCD number
ENO	BOOL	I, Q, M, D, L	Enable output

: INTEGER TO DOUBLE INTEGER(I_DI)



Parameter	Data Type	Memory Area	Description
EN	BOOL	I, Q, M, D, L, T, C	Enable input
IN	INT	I, Q, M, D, L or constant	Value to be converted
OUT	DINT	I, Q, M, D, L	Result
ENO	BOOL	I, Q, M, D, L	Enable output

مثال :



I0.0 را از MW10 به صورت INTEGER می گیرد تبدیل میکند به DOUBLE INTEGER و جواب را در MW12 میریزد.

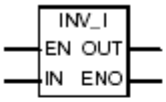
: DOUBLE INTEGER TO BCD(DI_BCD)

Parameter	Data Type	Memory Area	Description
EN	BOOL	I, Q, M, D, L, T, C	Enable input
IN	DINT	I, Q, M, D, L or constant	Double Integer
OUT	DWORD	I, Q, M, D, L	BCD value of the double integer
ENO	BOOL	I, Q, M, D, L	Enable output

: DOUBLE INTEGER TOREAL(DI_R)

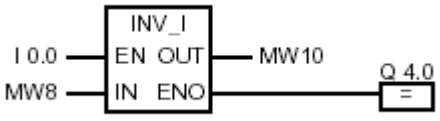
Parameter	Data Type	Memory Area	Description
EN	BOOL	I, Q, M, D, L, T, C	Enable input
IN	DINT	I, Q, M, D, L or constant	Value to be converted
OUT	REAL	I, Q, M, D, L	Result
ENO	BOOL	I, Q, M, D, L	Enable output

: ONES COMPELET INTEGER(INV_I)



Parameter	Data Type	Memory Area	Description
EN	BOOL	I, Q, M, D, L, T, C	Enable input
IN	INT	I, Q, M, D, L or constant	Input value
OUT	INT	I, Q, M, D, L	Ones complement of the integer
ENO	BOOL	I, Q, M, D, L	Enable output

مثال:



در مثال بالا داریم :10000001 MW8=01000001 که فرم INTEGER است.و در خروجی مکمل ۱ آن یعنی:

10111110 011111110 در MW8 داریم.

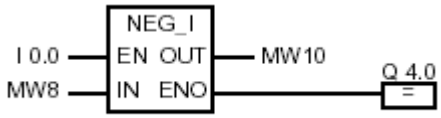
: ONES COMPELEMENT DOUBLE INTEGER (INV_DI)

Parameter	Data Type	Memory Area	Description
EN	BOOL	I, Q, M, D, L, T, C	Enable input
IN	DINT	I, Q, M, D, L or constant	Input value
OUT	DINT	I, Q, M, D, L	Ones complement of the double integer
ENO	BOOL	I, Q, M, D, L	Enable output

: TWOS COMPELEMENT INTEGER(NEG_I)

Parameter	Data Type	Memory Area	Description
EN	BOOL	I, Q, M, D, L, T, C	Enable input
IN	INT	I, Q, M, D, L or constant	Input value
OUT	INT	I, Q, M, D, L	Twos complement of the integer
ENO	BOOL	I, Q, M, D, L	Enable output

مثال:



در مثال بالا MW8=+10 و MW10= -10 است.

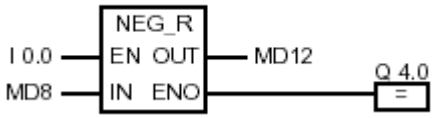
: TWOS COMPELEMENT DOUBLE INTEGER(NEG_DI)

Parameter	Data Type	Memory Area	Description
EN	BOOL	I, Q, M, D, L, T, C	Enable input
IN	DINT	I, Q, M, D, L or constant	Input value
OUT	DINT	I, Q, M, D, L	Twos complement of the double integer
ENO	BOOL	I, Q, M, D, L	Enable output

: NEGATIV REAL NUMBER(NEG_R)

Parameter	Data Type	Memory Area	Description
EN	BOOL	I, Q, M, D, L, T, C	Enable input
IN	REAL	I, Q, M, D, L or constant	Input value
OUT	REAL	I, Q, M, D, L	The result is the negated input value.
ENO	BOOL	I, Q, M, D, L	Enable output

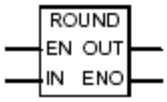
مثال:



در بالا داریم MD12= -6.234 , MD8=+6.234 قسمت حقیقی را مثبت کرده است.

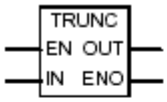
در زیر به نام بردن باقی المابها بسنده میکنیم:

: ROUND

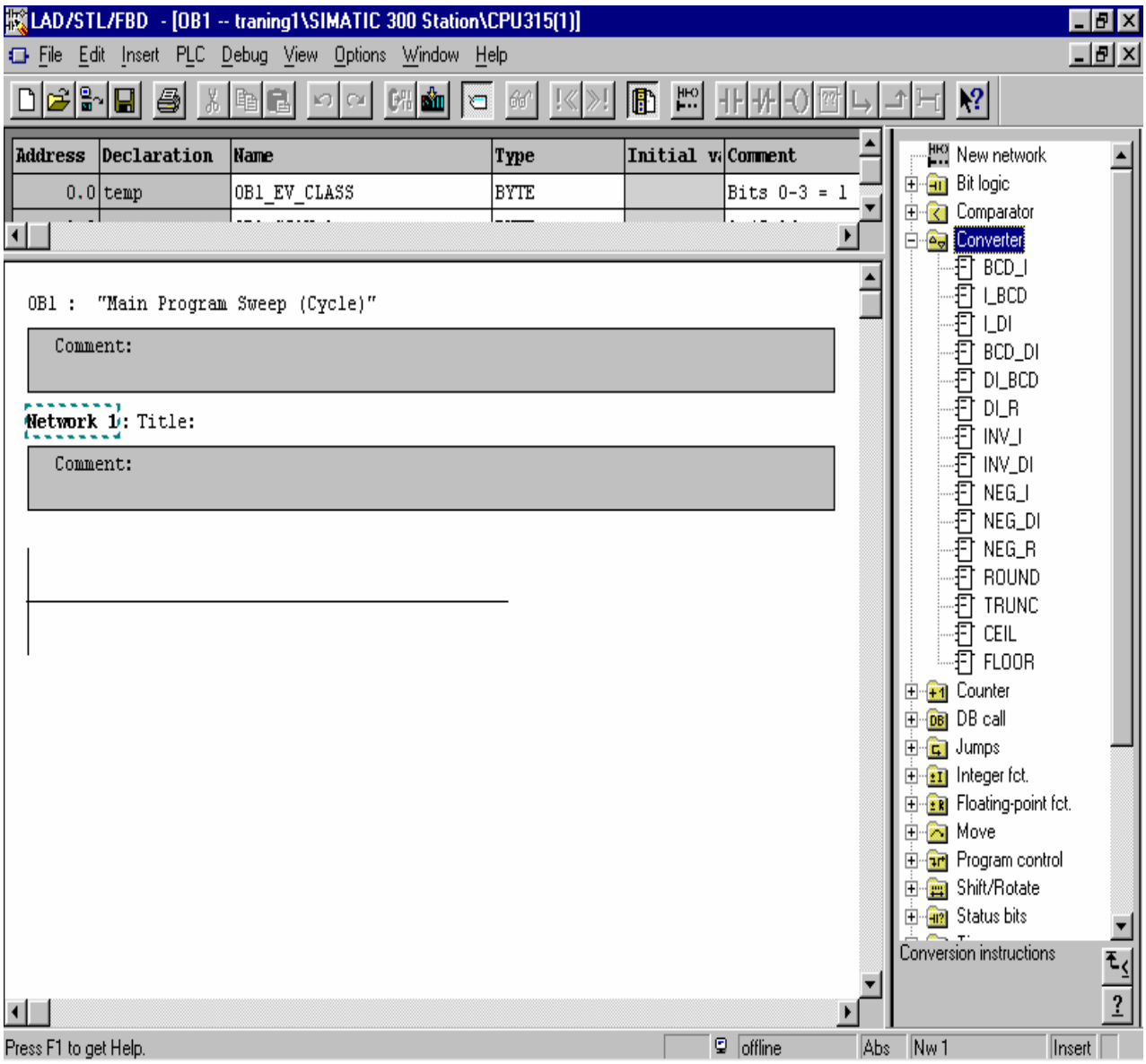


Parameter	Data Type	Memory Area	Description
EN	BOOL	I, Q, M, D, L, T, C	Enable input
IN	REAL	I, Q, M, D, L or constant	Value to be rounded
OUT	DINT	I, Q, M, D, L	IN rounded to the next double integer
ENO	BOOL	I, Q, M, D, L	Enable output

: TRUNC

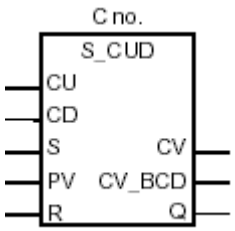


Parameter	Data Type	Memory Area	Description
EN	BOOL	I, Q, M, D, L, T, C	Enable input
IN	REAL	I, Q, M, D, L or constant	Value to be truncated
OUT	DINT	I, Q, M, D, L	Integer component of IN
ENO	BOOL	I, Q, M, D, L	Enable output


LAD/STL/FBD - [OB1 -- traning1\SIMATIC 300 Station\CPU315(1)]
File Edit Insert PLC Debug View Options Window Help
Address Declaration Name Type Initial v Comment
0.0 temp OB1_EV_CLASS BYTE Bits 0-3 = 1
OB1 : "Main Program Sweep (Cycle)"
Comment:
Network 1: Title:
Comment:
New network
Bit logic
Comparator
Converter
BCD_I
I_BCD
I_DI
BCD_DI
DI_BCD
DI_R
INV_I
INV_DI
NEG_I
NEG_DI
NEG_R
ROUND
TRUNC
CEIL
FLOOR
Counter
DB call
Jumps
Integer fct.
Floating-point fct.
Move
Program control
Shift/Rotate
Status bits
Conversion instructions
Press F1 to get Help.
offline
Abs
Nw 1
Insert

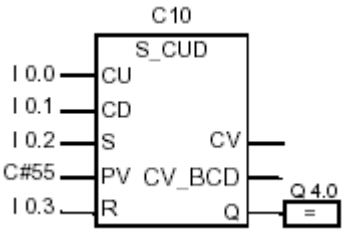
شمارنده ها (COUNTER) :

: ASSIGN PARAMETER AND COUNT UP/DOWN (S_CUD)



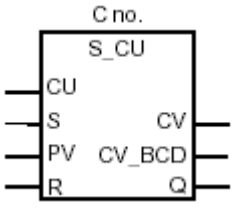
Parameter English	Parameter German	Data Type	Memory Area	Description
no.	Nr.	COUNTER	C	Counter identification number. The range depends on the CPU.
CU	ZV	BOOL	I, Q, M, D, L	ZV input: Up Counter
CD	ZR	BOOL	I, Q, M, D, L	ZR input: Down Counter
S	S	BOOL	I, Q, M, D, L, T, C	Input for presetting the counter
PV	ZW	WORD	I, Q, M, D, L or constant	Count value in the range between 0 and 999 or Count value entered as C#<value> in BCD format
R	R	BOOL	I, Q, M, D, L, T, C	Reset input
CV	DUAL	WORD	I, Q, M, D, L	Current count value (hexadecimal number)
CV_BCD	DEZ	WORD	I, Q, M, D, L	Current count value (BCD format)
Q	Q	BOOL	I, Q, M, D, L	Status of the counter

مثال:



در شمارنده فوق با تغییر سیگنال *از ۰ به ۱ در I0.2* SET می شود.C#55 یعنی مقدار شمارش تا ۵۵ دفعه مي باشد .هنگامی که I0.0 از ۰ به ۱ تغییر کند.شمارنده به صورت افزایشی تا ۹۹۹ به بالا می شمارد.واگر I0.1 از ۰ به ۱ تغییر کند به صورت کاهشی تا ۰ می شمارد.هنگامی که I0.3 از ۰ به ۱ تغییر کند مقدار COUNTER در ۰ ست می شود.

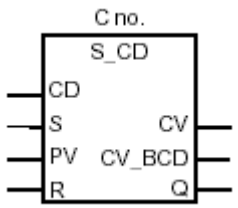
:ASSIGN PARAMETER AND COUNT UP (S_CU)



Parameter English	Parameter German	Data Type	Memory Area	Description
no.	Nr.	COUNTER	C	Counter identification number. The range depends on the CPU.
CU	ZV	BOOL	I, Q, M, D, L	ZV input: Up Counter
S	S	BOOL	I, Q, M, D, L, T, C	Input for presetting the counter
PV	ZW	WORD	I, Q, M, D, L or constant	Count value in the range between 0 and 999 or Count value entered as C#<value> in BCD format
R	R	BOOL	I, Q, M, D, L, T, C	Reset input
CV	DUAL	WORD	I, Q, M, D, L	Current count value (hexadecimal number)
CV_BCD	DEZ	WORD	I, Q, M, D, L	Current count value (BCD format)
Q	Q	BOOL	I, Q, M, D, L	Status of the counter

: ASSIGN PARAMETER AND COUNT DOWN (S_CD)

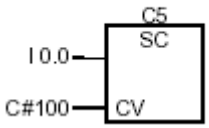
با جدول مشابه بالا



: SET COUNTER VALUE

Parameter English	Parameter German	Data Type	Memory Area	Description
Counter no.	Counter no	COUNTER	C	The address1 specifies the number of the counter which is to be preset with a value.
CV	ZW	WORD	I, Q. M, D, L or constant	The value to be preset (address2) can lie between 0 and 999. When you enter a constant, C# must stand in front of the value specified by the BCD format, for example, C#100.

مثال:



شمارنده فوق مقدار ۱۰۰ به COUNTER VALUE داده شده و با تحریک I0.0 مقدار کانتر به ۱۰۰ می رود در غیر اینصورت اتفاقی نمی افتد.(از ۱ به ۰ برویم)

: UP COUNTER(CU)

Parameter	Data Type	Memory Area	Description
Counter no.	COUNTER	C	The address specifies the number of the counter which is to be incremented.

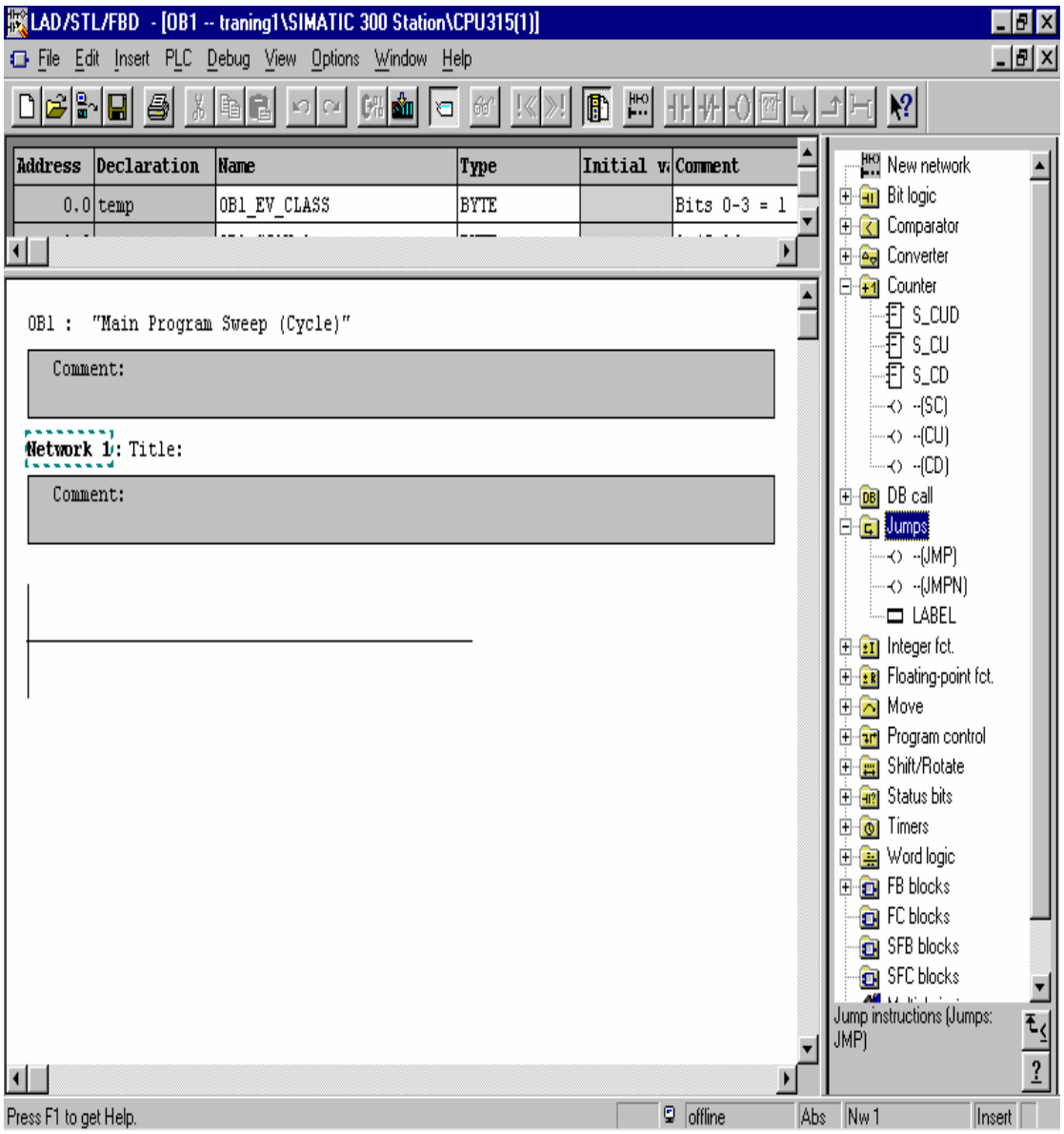
شمارنده فوق به صورت صعودی تا ۹۹۹ می شمارد.

DOWN COUNTER(CD)

شمارنده معکوس

Parameter	Data Type	Memory Area	Description
Counter no.	COUNTER	C	The address specifies the number of the counter which is to be decremented.

در شکل زیر در قسمت چپ روش انتخاب انواع شمارنده ها از داخل کنابخانه S7 مشخص گردیده است

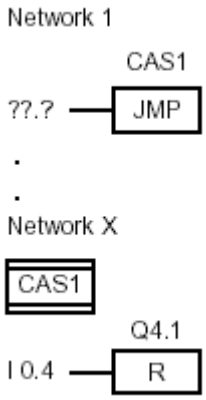


دستورات پرش (JUMP) :

: UN CONDITIONAL JUMP TO BLOCK

Parameter	Data Type	Memory Area	Description
Name of a jump label	-	-	The address specifies the label to which the program will jump unconditionally.

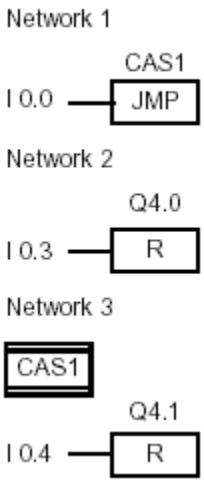
مثال:



: CONDITIONAL JUMPIN A BLOCK

Parameter	Data Type	Memory Area	Description
Name of a jump label	-	-	The address specifies the label to which the program will jump if the RLO is 1.

مثال:



زمانیکه I0.0 فعال شود (یک شود) CAS1 اجرا می شود.

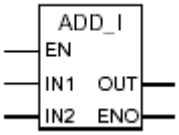
زمانیکه I0.3 فعال شود(یک شود) فرمان RESET می آید.

:INTEGER MATH INSTRUCTION

کاربرد دستورات رياضي براي اعداد صحيح در نرم افزار

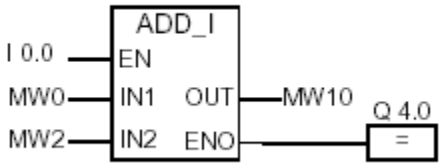
ADD INTEGER(ADD_I)

دستور جمع اعداد صحیح



Parameter	Data Type	Memory Area	Description
EN	BOOL	I, Q, M, D, L, T, C	Enable input
IN1	INT	I, Q, M, D, L or constant	First value for addition
IN2	INT	I, Q, M, D, L or constant	Second value for addition
OUT	INT	I, Q, M, D, L	Result of addition
ENO	BOOL	I, Q, M, D, L	Enable output

مثال:



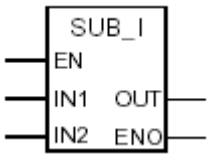
با فعال شدن این بلوک بوسیله ورودي I0.0 شروع به جمع می کند.

۲ ورودی MW0 وMW2 را می گیرد و با هم جمع میکند و خروجی را در MW10 میریزد

.اگر حاصل در رنج مورد نظر نباشد یا به هر دلیل غیر قابل قبول باشد

I0.0 صفر میشود و Q4.0 را صفر می کند.

SUBTRACT INTEGER(SUB_I)



Parameter	Data Type	Memory Area	Description
EN	BOOL	I, Q, M, D, L, T, C	Enable input
IN1	INT	I, Q, M, D, L or constant	Minuend (value from which second value is subtracted)
IN2	INT	I, Q, M, D, L or constant	Subtrahend (value subtracted from the first value)
OUT	INT	I, Q, M, D, L	Result of subtraction
ENO	BOOL	I, Q, M, D, L	Enable output

تایمرها (TIMER INSTRUCTIONS) :

انواع تایمر عبارتند از:

۱)

(PULSE TIMER) S_PULSE

۲)

(EXTENDED PULSETIMER) S_PEXT

۳)

(ON-DELAY TIMER) S_ODT

۴)

(RETENTIVE ON –DELAY TIMER) S_ODTS

۵)

(OF DELAY TIMER) S_OFFDT

۶)

SP

۷)

SD

۸)

SE

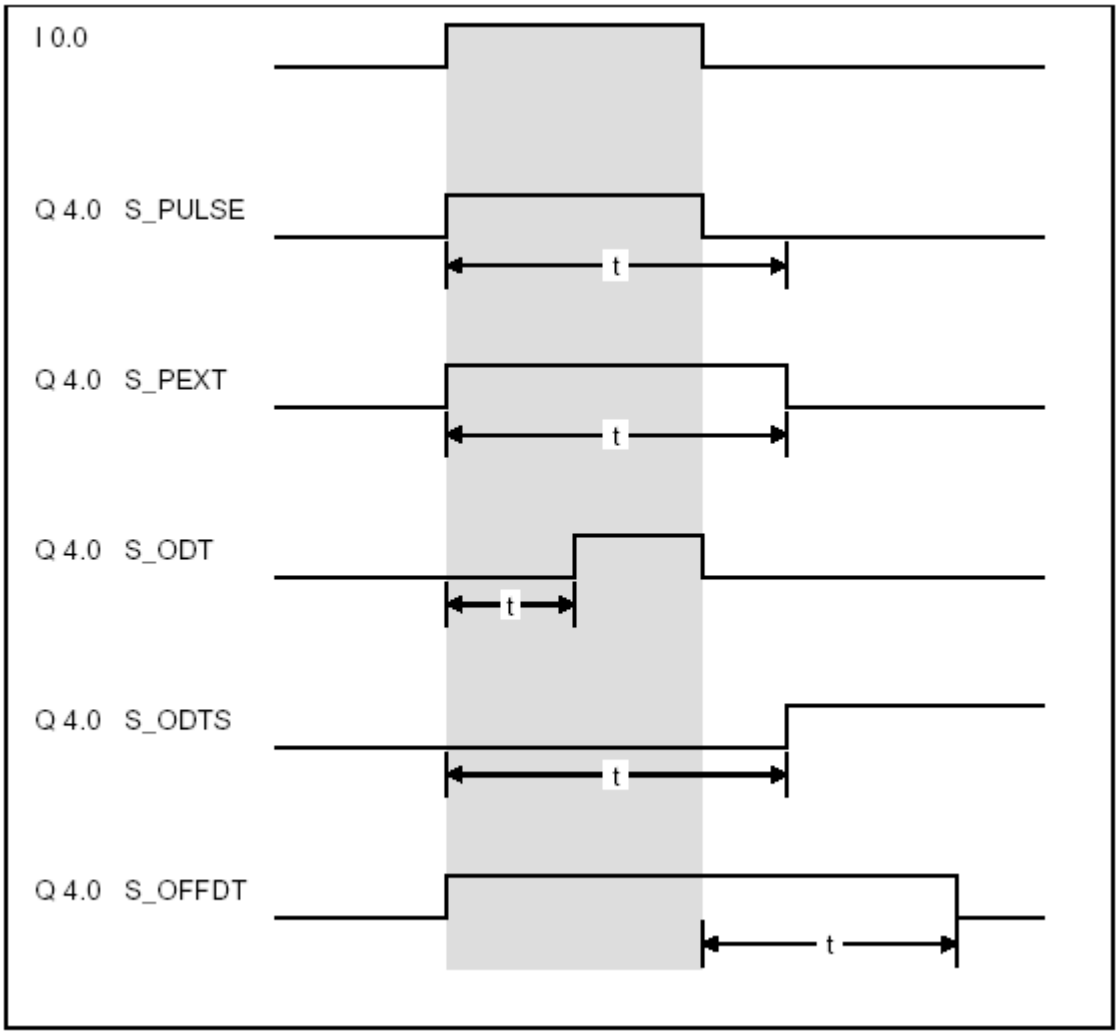
۹)

SS

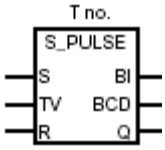
(۱۰

SF

ابتدا مقایسه ای کلی در مورد تایمر ها و تفاوت های بین آنها از روی شکل در زیر ارایه میدهیم:

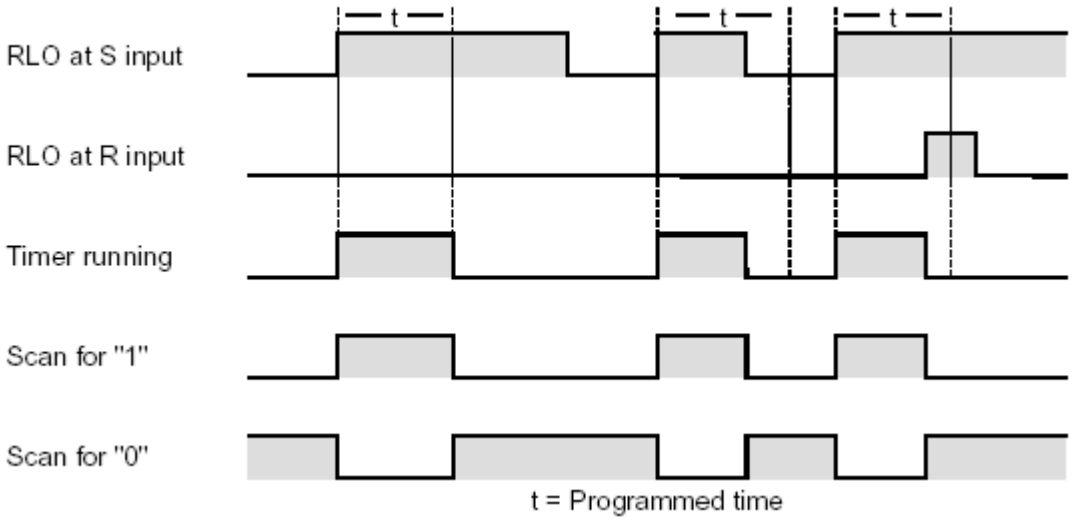


S_PULSE



Parameter English	Parameter German	Data Type	Memory Area	Description
no.	Nr.	TIMER	T	Timer identification number. The range depends on the CPU.
S	S	BOOL	I, Q, M, D, L, T, C	Start input
TV	TW	S5TIME	I, Q, M, D, L or constant	Preset time value (range 0-9999)
R	R	BOOL	I, Q, M, D, L, T, C	Reset input
BI	DUAL	WORD	I, Q, M, D, L	Time remaining (value in integer format)
BCD	DEZ	WORD	I, Q, M, D, L	Time remaining (value in BCD format)
Q	Q	BOOL	I, Q, M, D, L	Status of the timer

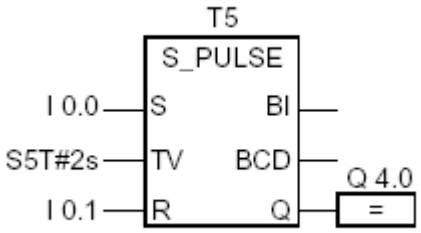
پاسخ این تایمر را به ورودی های مختلف SET ,RESET را ملاحظه میکنیم:



که پالس RLO می آید وبا توجه به RESET پاسخ را ملاحظه می کنیم:

TIMERبدون توجه به پالس به مقدار زمان خود روشن است و پس از پایان یافتن زمانش به پایان می رسد

مثال



در ابتدا باید در مورد نامگذاری تایمر ها و چگونگی زمان دادن به آن به نکات زیر توجه کرد:

Time Base	Binary Code for the Time Base
10 ms	00
100 ms	01
1 s	10
10 s	11

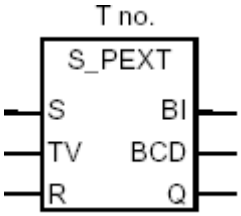
شکل باینری زمان ها به فرم بالاست.

S5T شکل استاندارد نمایش تایمر هاست.

زمان مورد نظر را به فرم #5 نشان داده میشود.که در پایه TV نوشته می شود.

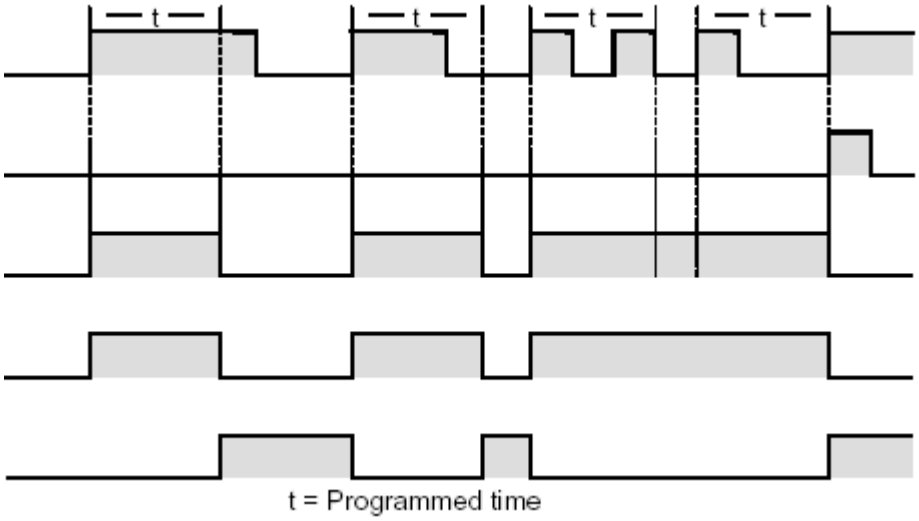
در مثال بالا با تحریک I0.0 یعنی SET تایمر فعال و ۲ ثانیه می شمارد و با فعال شدن RESET دیگر تایمر نمی شمارد.خروجی در Q4.0 ریخته می شود و به ۲ فرم باینری و BCD نمایش می دهد.

S_PEXT

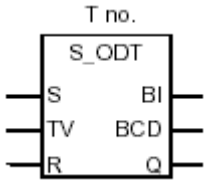


Parameter English	Parameter German	Data Type	Memory Area	Description
no.	Nr.	TIMER	T	Timer identification number. The range depends on the CPU.
S	S	BOOL	I, Q, M, D, L, T, C	Start input
TV	TW	S5TIME	I, Q, M, D, L or constant	Preset time value (range 0-9999)
R	R	BOOL	I, Q, M, D, L, T, C	Reset input
BI	DUAL	WORD	I, Q, M, D, L	Time remaining (value in integer format)
BCD	DEZ	WORD	I, Q, M, D, L	Time remaining (value in BCD format)
Q	Q	BOOL	I, Q, M, D, L	Status of the timer

تا انتهایTIME میرود و منتظر لبه بالا رونده برای تحریک بعدی می مانند و اگر لبه نیاید ادامه مي يابد

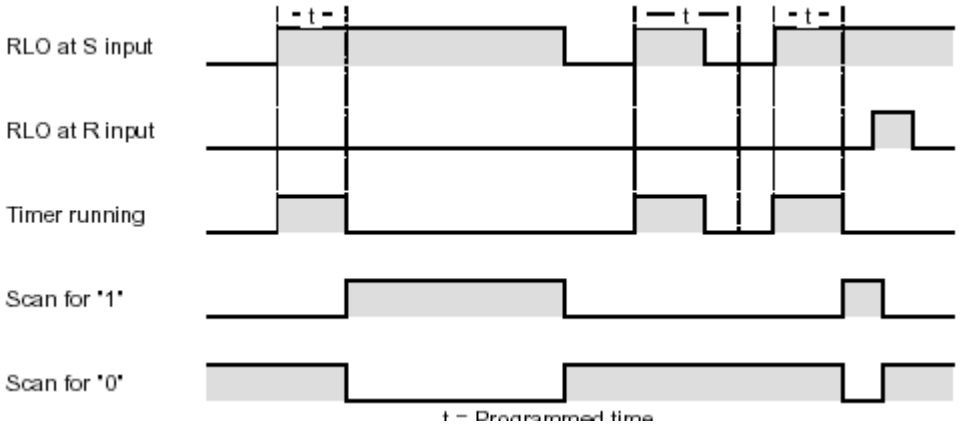


: S_ODT

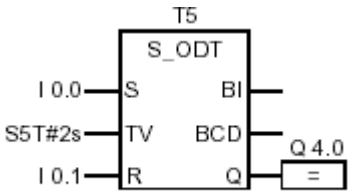


Parameter English	Parameter German	Data Type	Memory Area	Description
no.	Nr.	TIMER	T	Timer identification number. The range depends on the CPU.
S	S	BOOL	I, Q, M, D, L, T, C	Start input
TV	TW	S5TIME	I, Q, M, D, L or constant	Preset time value (range 0-9999)
R	R	BOOL	I, Q, M, D, L, T, C	Reset input
BI	DUAL	WORD	I, Q, M, D, L	Time remaining (value in integer format)
BCD	DEZ	WORD	I, Q, M, D, L	Time remaining (value in BCD format)
Q	Q	BOOL	I, Q, M, D, L	Status of the timer

با تحریک TIMER پالس به اندازه زمان خود می آید ومنتظر لبه بالا رونده بعدی می ماند و TIMER بدون توجه به اندازه پالس و با توجه به لبه تحریک می شود.

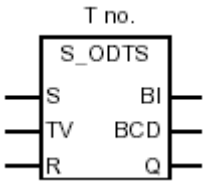


مثال:



با تحریکI0.0 تایمر به کار می افتد واگر ۲ ثانیه به پایان رسد و I0.0 همچنان فعال باشد Q4.0 همچنان ۱ است واگر I0.0 تغییر حالت دهد Q4.0 صفر می شوداگر I0.1تغییر حالت دهداز ۰ به ۱ در زمانیکه TIMER کار می کند تایمر RESTART می شود.

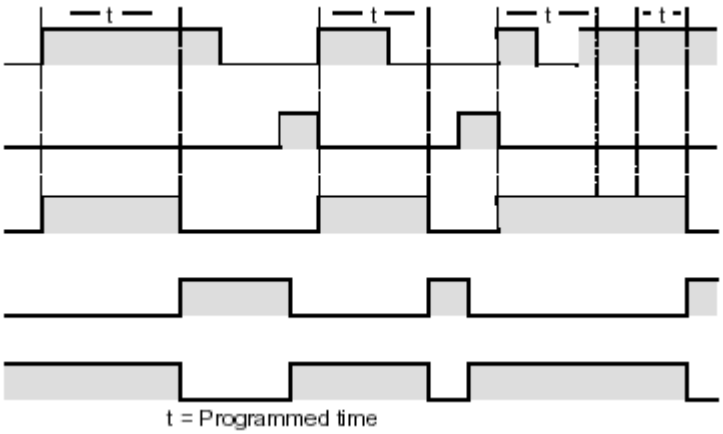
S_ODTS :



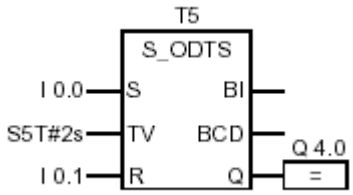
Parameter English	Parameter German	Data Type	Memory Area	Description
no.	Nr.	TIMER	T	Timer identification number. The range depends on the CPU.
S	S	BOOL	I, Q, M, D, L, T, C	Start input
TV	TW	S5TIME	I, Q, M, D, L or constant	Preset time value (range 0-9999)
R	R	BOOL	I, Q, M, D, L, T, C	Reset input
BI	DUAL	WORD	I, Q, M, D, L	Time remaining (value in integer format)
BCD	DEZ	WORD	I, Q, M, D, L	Time remaining (value in BCD format)
Q	Q	BOOL	I, Q, M, D, L	Status of the timer

نکته :

در این تایمر بعد از آمدن فرمان تایمر زمان مورد نظر را سپری می کند و تا آمدن لبه بالا رونده ادامه می یابد واگر لبه نیاید پالس هم نداریم.

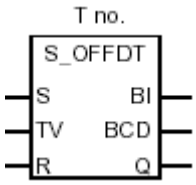


مثال:



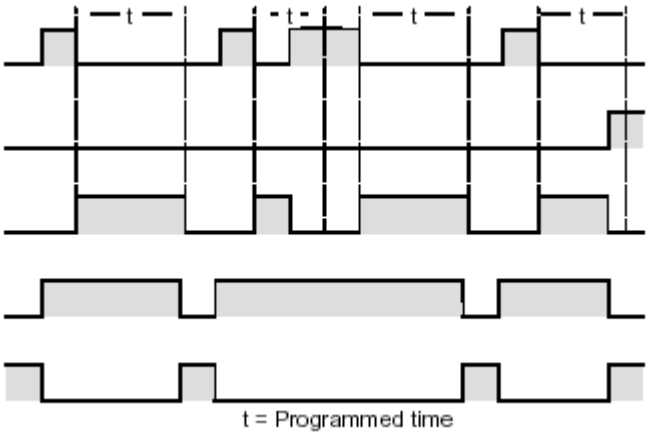
وقتیکهI0.0 از از ۰ به ۱ (لبه بالا رونده) تغییر حالت دهد تایمر بر خلاف تغییر سیگنال عمل میکند یعنی اگر I0.0 از ۰ به ۱ تغییر حالت پیدا کند بعد از گذشت زمان T تایمر RESTART می شود و اگر در زمانیکه تایمر کار میکند I0.1 ز ۰ به ۱ تغییر وضعیت دهد تایمر RESTART می شود.وQ4.0 در زمانیکه زمان تایمر تمام شده و و I0.1 صفر است روشن است.

: S_OFFDT

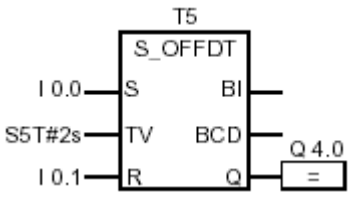


Parameter English	Parameter German	Data Type	Memory Area	Description
no.	Nr.	TIMER	T	Timer identification number. The range depends on the CPU.
S	S	BOOL	I, Q, M, D, L, T, C	Start input
TV	TW	S5TIME	I, Q, M, D, L or constant	Preset time value (range 0-9999)
R	R	BOOL	I, Q, M, D, L, T, C	Reset input
BI	DUAL	WORD	I, Q, M, D, L	Time remaining (value in integer format)
BCD	DEZ	WORD	I, Q, M, D, L	Time remaining (value in BCD format)
Q	Q	BOOL	I, Q, M, D, L	Status of the timer

این تایمر با لبه پایین رونده فعال میشود وتا پایان TIME میرود اگر در این زمان پالس پایین رونده دیگری بیاید پالس به پایان میرود و اگر نه تا انتهای پالس میرود.



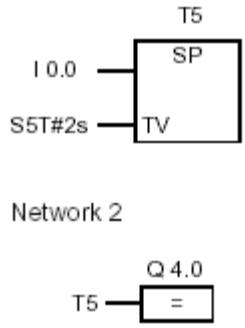
مثال:



اگر I0.0 از ۱ به ۰ تغییر کند تایمر بکار می افتد و Q4.0 زمانی ۱ است که I0.0 یک باشد یا تایمر در حال کار باشد و I0.1 اگر در زمان کار کردن تایمر فعال شود تایمر RESTART میشود.

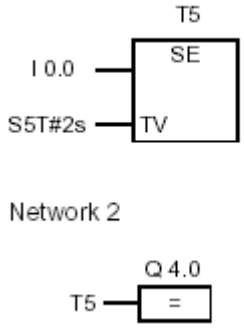
: SP(START PULS TIMER)

مثال:



اگر I0.0 از ۱ به ۰ تغییر کند تایمر بکار می افتد و تا زمانیکهSIGNAL STATE یک است تایمر می شمارد و تایمر مرتبآ پالس مربعی با فواصل معین ایجاد می کند.

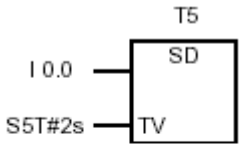
: SE (START EXTENDED PULSE TIMER)

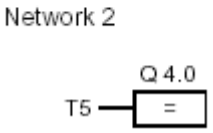


اگر I0.0 از ۱ به ۰ تغییر کند تایمر بکار می افتد و بدون تاثیر از لبه پایین رونده ادامه می یابد و اگر I0.0بعد از اتمام TIME فعال شود تایمر RESTART می شود.

: SD(START ON DELAY TIMER)

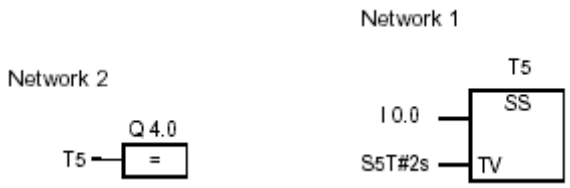
مثال:



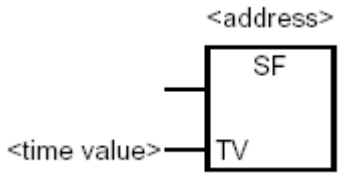


اگر I0.0 از ۱ به ۰ تغییر کند تایمر بکار می افتد واگر I0.0 یک باشد و تایمر نیز از شمارش خارج Q4.0 یک است و اگر SIGNAL STATE از ۱ به ۰ تغییر کند TIMER ری استارت می شود.

: SS (START RETENTIVE ON DELAY)

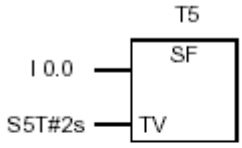


: SF (STSRT OFF DELAY TIMER)

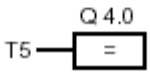


Parameter English	Parameter German	Data Type	Memory Area	Description
Timer no.	Timer no.	TIMER	T	The address specifies the number of the timer which is to be started.
TV	TW	S5TIME	E, A, M, D, L or constant	Timer value (S5TIME format)

مثال:

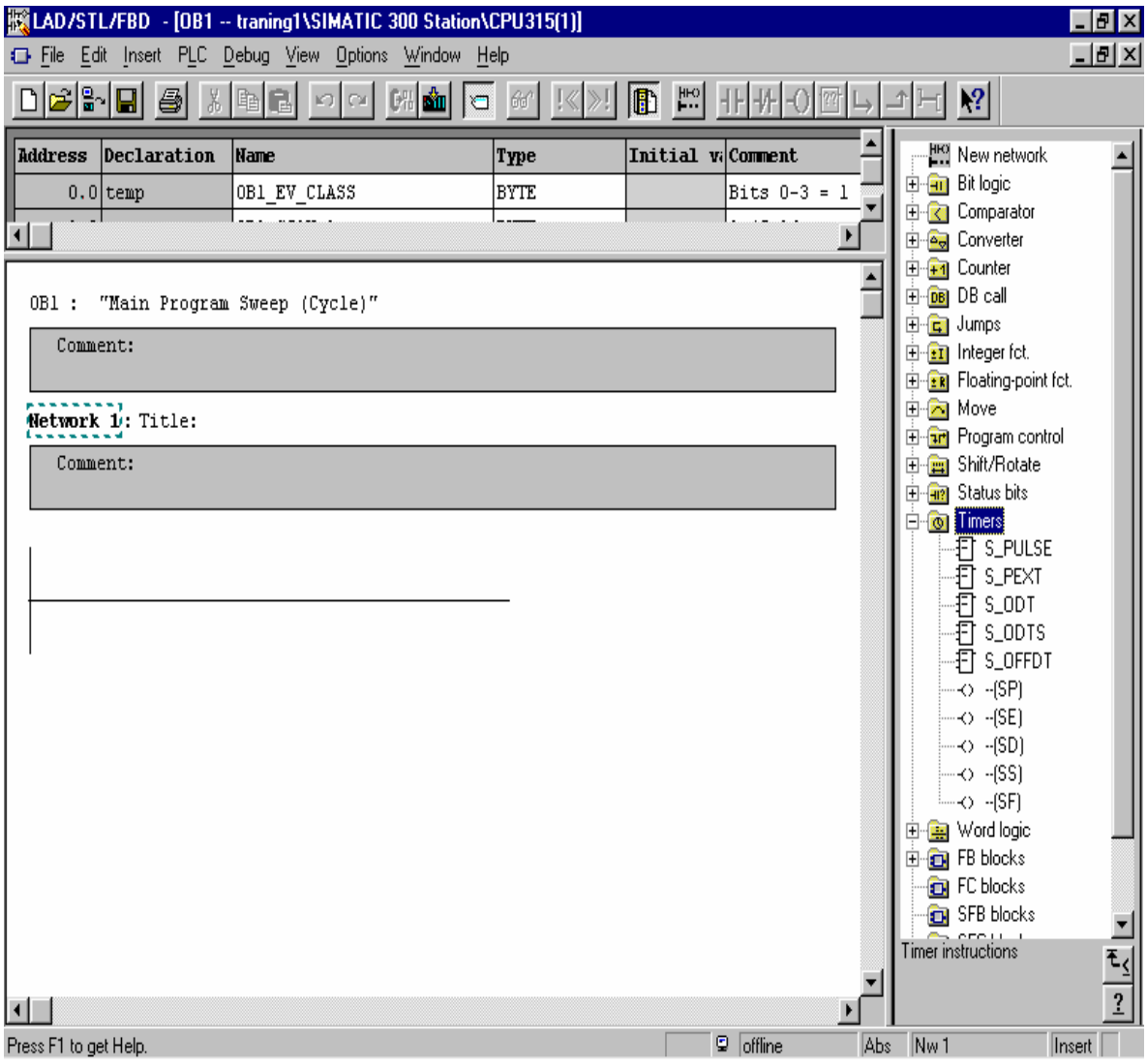


Network 2



زمانیکه SIGNSL STSTE از ۱ به ۰ تغییر کند تایمر بکار می افتد واگر از ۰ به ۱ برود TIMER ری استارت می شود Q4.0 زمانی ۱ است که یا IO.O یک باشد یا تایمر در حال کار باشد.

در شکل زیر نمای تایمر ها را در برنامه S7 در قسمت library ملاحظه می کنیم:



شیفترها(SHIFT INSTRUCTION)

انواع شیفتر عبارتند از:

SHR_I (SHIFT RIGH INTEGER)

SHR_DI (SHIFT RIGH DOUBLE INTEGER)

SHL_W (SHIFT LEFT WORD)

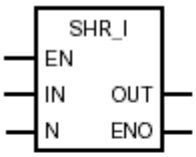
***SH*R_W (SHIFT RIGHT WORD)**

SHL_DW (SHIFT LEFT DOUBLE WORD)

SHR_DW (SHIFTRIGH DOUBLE WORD)

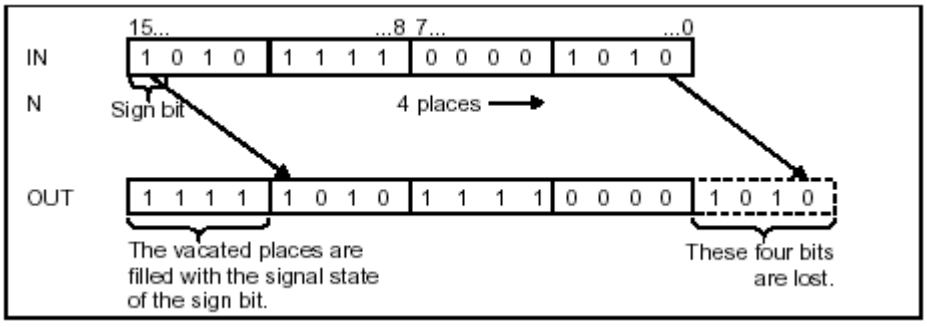
که در زیر به معرفی چندی از آنها می پردازیم:

: SHR_I

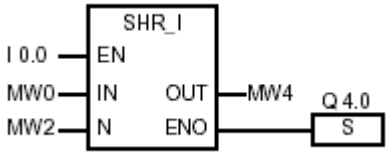


Parameter	Data Type	Memory Area	Description
EN	BOOL	I, Q, M, L, D, T, C	Enable input
IN	INT	I, Q, M, L, D	Value to be shifted
N	WORD	I, Q, M, L, D	Number of bit positions by which the value will be shifted
OUT	INT	I, Q, M, L, D	Result of the shift instruction
ENO	BOOL	I, Q, M, L, D	Enable output

که در شکل نحوه شیفت دادن آن را ملاحظه میکنیم:

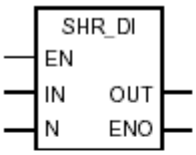


مثال:



با فعال شدن I0.0 شیفتر فعال شده و آنچه در MW0 است را به تعداد بیتی که MW2 اعلام می کند به راست شیفت می دهد و نتیجه را در MW4 می ریزد.

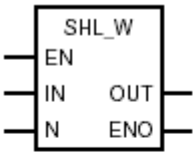
SHR_DI



Parameter	Data Type	Memory Area	Description
EN	BOOL	I, Q, M, L, D, T, C	Enable input
IN	DINT	I, Q, M, L, D	Value to be shifted
N	WORD	I, Q, M, L, D	Number of bit positions by which the value will be shifted
OUT	DINT	I, Q, M, L, D	Result of the shift instruction
ENO	BOOL	I, Q, M, L, D	Enable output

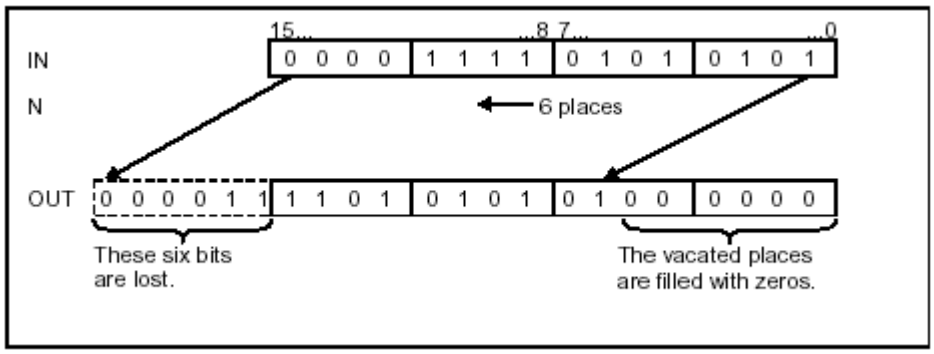
همانند قبل است اما این بار DOUBLE INTEGER است.

: SHL_W (SHIFT LEFT WORD)



Parameter	Data Type	Memory Area	Description
EN	BOOL	I, Q, M, L, D, T, C	Enable input
IN	WORD	I, Q, M, L, D	Value to be shifted
N	WORD	I, Q, M, L, D	Number of bit positions by which the value will be shifted
OUT	WORD	I, Q, M, L, D	Result of the shift instruction
ENO	BOOL	I, Q, M, L, D	Enable output

که فرم عددی که میگیرد WORD است و همانند شکل زیر به چپ شیفت می دهد.

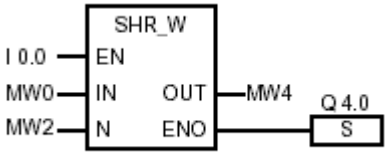


SHR_W(SHIFT RIGHT WORD)

همانند بالاست با این تفاوت که به سمت راست شیفت می دهد.

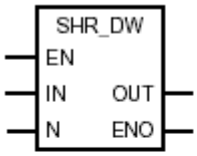
Parameter	Data Type	Memory Area	Description
EN	BOOL	I, Q, M, L, D, T, C	Enable input
IN	WORD	I, Q, M, L, D	Value to be shifted
N	WORD	I, Q, M, L, D	Number of bit positions by which the value will be shifted
OUT	WORD	I, Q, M, L, D	Result of the shift instruction
ENO	BOOL	I, Q, M, L, D	Enable output

مثال:

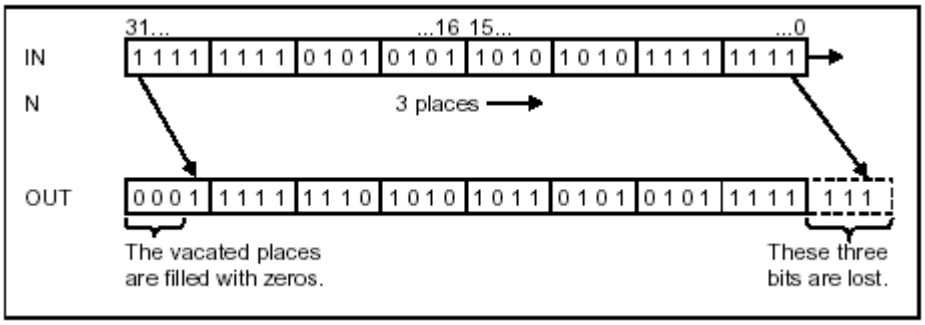


با ۱ شدن I0.0 فعال می شود و محتویات MW0 را به تعداد بیت موجود در MW2 به صورت WORD به سمت راست شیفت می دهد و نتیجه را در MW4 میریزد.

SHR_DW (SHIFT RIGHT DOUBLE WORD)

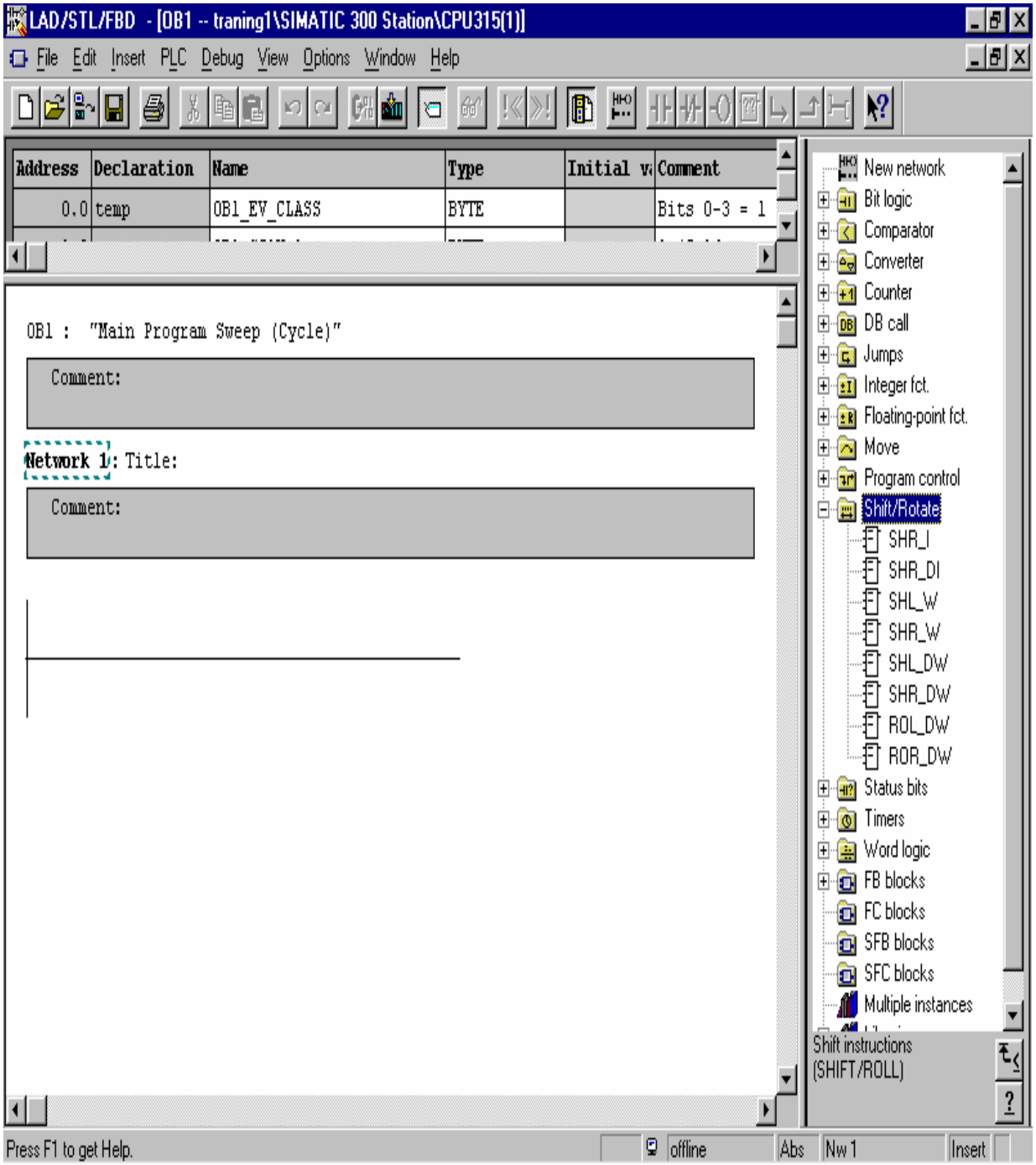


Parameter	Data Type	Memory Area	Description
EN	BOOL	I, Q, M, L, D, T, C	Enable input
IN	DWORD	I, Q, M, L, D	Value to be shifted
N	WORD	I, Q, M, L, D	Number of bit positions by which the value will be shifted
OUT	DWORD	I, Q, M, L, D	Result of the shift instruction
ENO	BOOL	I, Q, M, L, D	Enable output

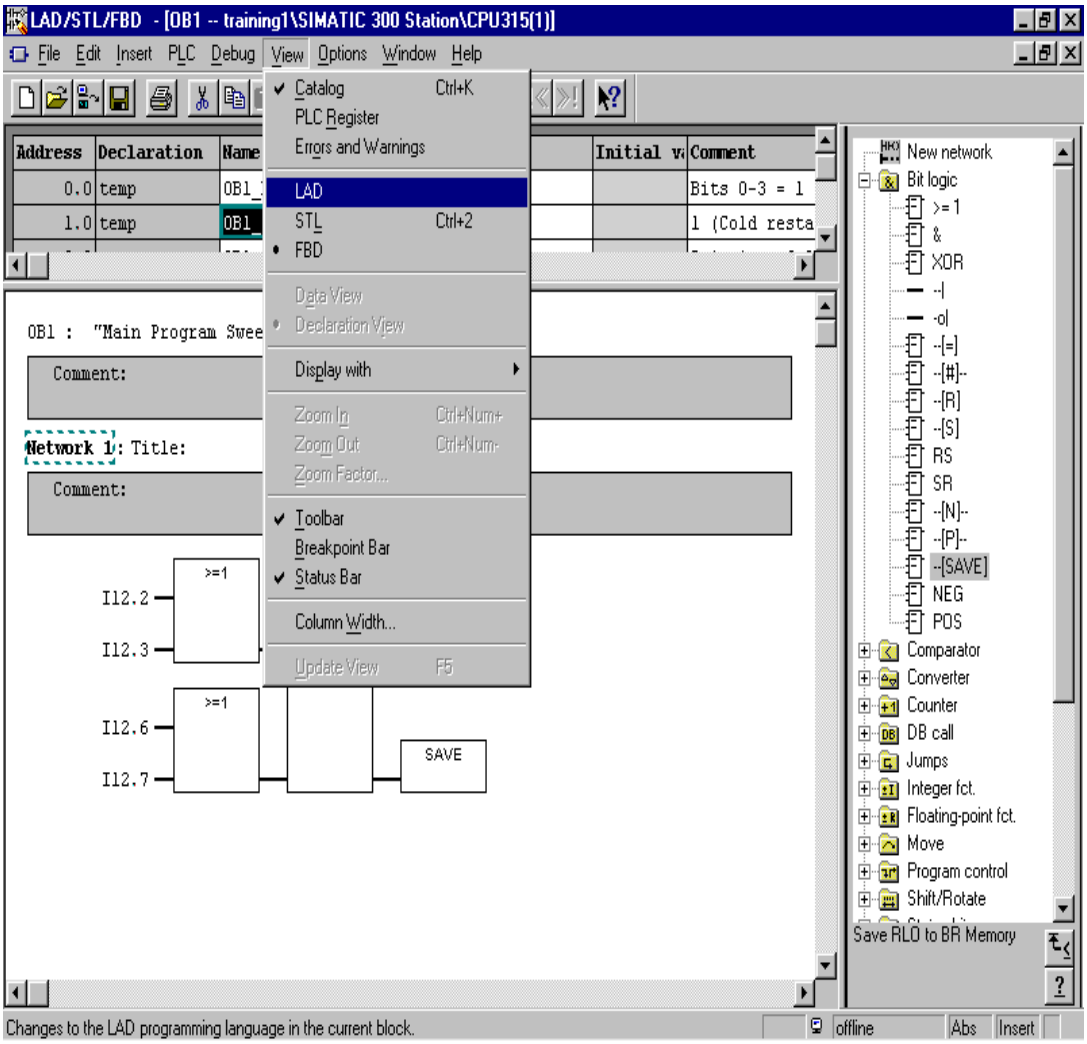


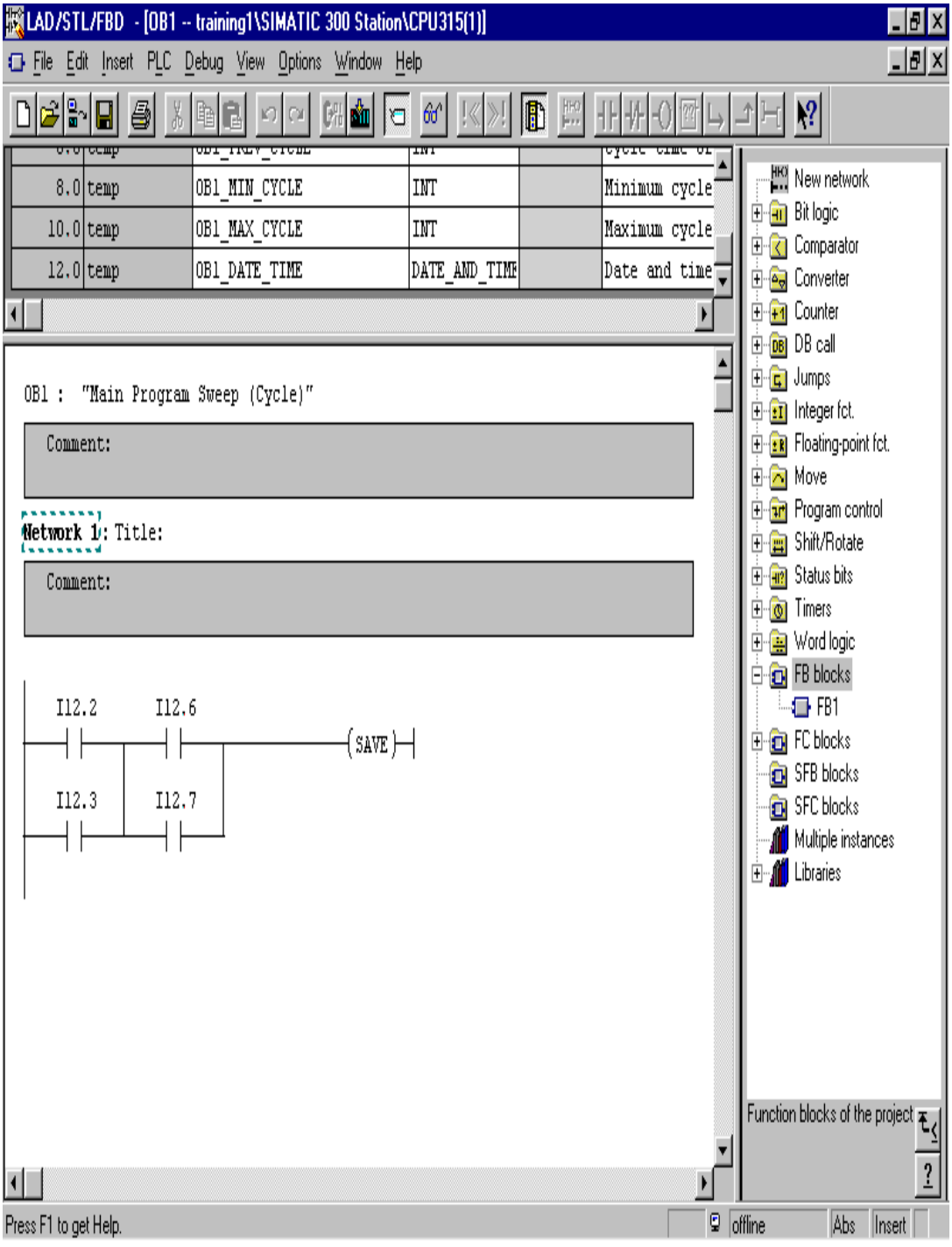
که فرم اطلاعات به صورت DOUBLE WORD را می گیرد و به سمت راست شیفت میدهد

در این شکل نحوه انتخاب دستور شیفت را از داخل library برنامه s7 مشاهده مي كنيم



در نهایت گزارش را با طرح بر نامه ای ساده در محیط FBD و LADD به پایان می بریم:




LAD/STL/FBD - [OB1 -- training1\SIMATIC 300 Station\CPU315(1)]
File Edit Insert PLC Debug View Options Window Help
8.0 temp OB1_MIN_CYCLE INT Minimum cycle
10.0 temp OB1_MAX_CYCLE INT Maximum cycle
12.0 temp OB1_DATE_TIME DATE_AND_TIME Date and time
OB1 : "Main Program Sweep (Cycle)"
Comment:
Network 1: Title:
Comment:
I12.2
I12.6
SAVE
I12.3
I12.7
New network
Bit logic
Comparator
Converter
Counter
DB call
Jumps
Integer fct.
Floating-point fct.
Move
Program control
Shift/Rotate
Status bits
Timers
Word logic
FB blocks
FB1
FC blocks
SFB blocks
SFC blocks
Multiple instances
Libraries
Function blocks of the project
Press F1 to get Help.
offline
Abs
Insert

تقدير و تشكر :

در نهايت جا دارد از تمام دوستان بخاطر قصوراتي كه در اين جزوه ملاحظه نموده اند عذر خواهي نمايم و از ايشان تمنا دارم كه بنده را براي رفع مشكلات و ايرادات موجود راهنمايي و ياري بفرمايند .

همانطور كه كارشناسان محترم اين رشته اطلاع دارند مطالب موجود در مراجع زيمنس بسيار حجيم و زياد بوده و نمي توان تمامي مطالب را بصورت يک كتابچه تدوين نمود .

مطالب ذكر شده در فوق در جقيقت قسمت اول شروع كار با نرم افزار s7 و Plc زيمنس مي باشد كه اميدوارم براي تمامي همكاران عزيز بعنوان يک reference مفيد واقع گردد

انشاالله در آينده نزديک سعي در ارائه قسمت دوم اين كتابچه كه كاربرد برنامه نويسي با نرم افزار s7 بوده و مثالهاي عملي و مفيد در آن بكار برده شده باشد ادامه مي دهيم و همچنين مطالبي درباره نحوه network configuration و ارتباط شبكه ايي plc با قطعات داخلي و همچنين plc ها با يكديگر ,و همچنين نحوه setting پانل هاي اپراتوري زيمنس op3-7-27-37-270 و كار با نرم افزار هاي graph & simulation را به استحضار خوانندگان عزيز مي رسانم .

امیدوارم که خداوند در این راه فرصت کافي را در اختیار اینجانب قرار دهد تا بتوانیم خدمتي را براي کشور عزیزمان و جامعه صنعتي و اتوماسیون بنامیم .

چکیده:

آشنایی با PLC و نرم افزار s7 , انواع بلوک در s7 ,
فرم های نمایش در s7 , تایمر ها , شمارنده ها,
مقایسه کننده ها , تبدیل کننده ها و شیفت دهنده ها و سخت افزار در s7

منابع بکار گرفته شده در این مجموعه :

- **Working with step 7 (siemens manuals)**
- **Configuring hardware with step 7 (siemens manuals)**
- **Getting started with step 7 (siemens manuals)**
- **Ladder logic for s7-300 and s7-400 programming (siemens manuals)**
- **Functiom block diagram for s7-300 and s7-400 programming (siemens manuals)**

ومن الله توفیق

محمد یادگار
کارشناس فني برق و الکترونیک
اداره کل تعمیرات مونتاژ یک
اداره پشتیباني مهندسي