

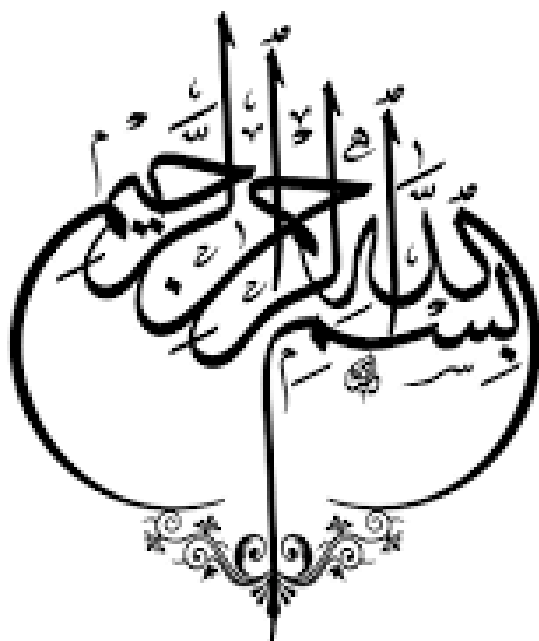
جزوه آزمایشگاه کنترل صنعتی

گروه مهندسی برق-کنترل

دانشگاه بین‌المللی امام خمینی (ره)



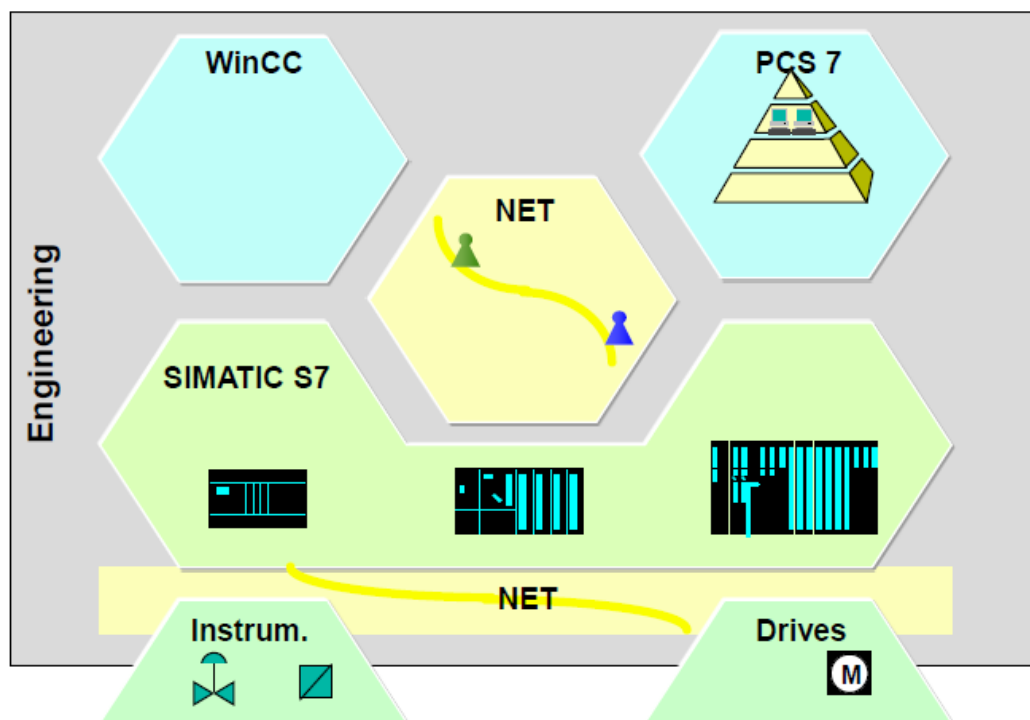
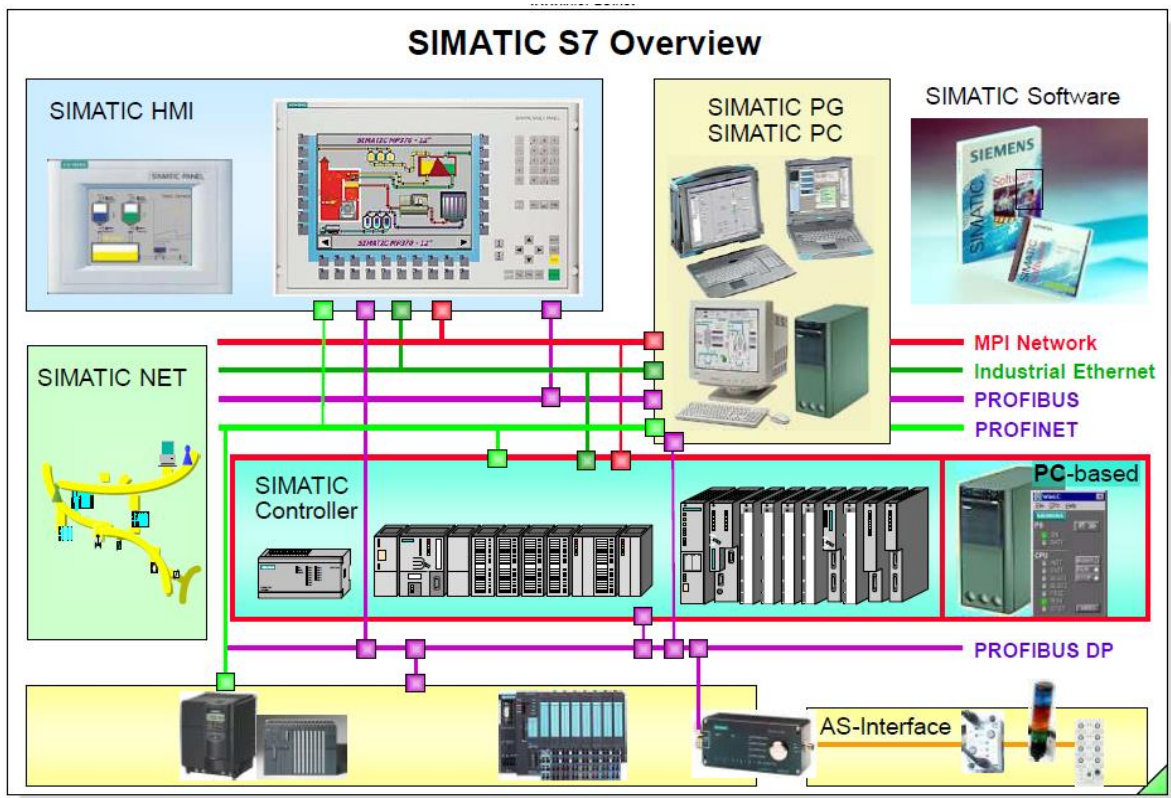
مهندس اکبر اویسی فر



إِنَّ لِكُلِّ شَيْءٍ زَكَاةً ، وَزَكَاةُ الْعِلْمِ أَنْ يُعَلِّمَهُ أَهْلَهُ

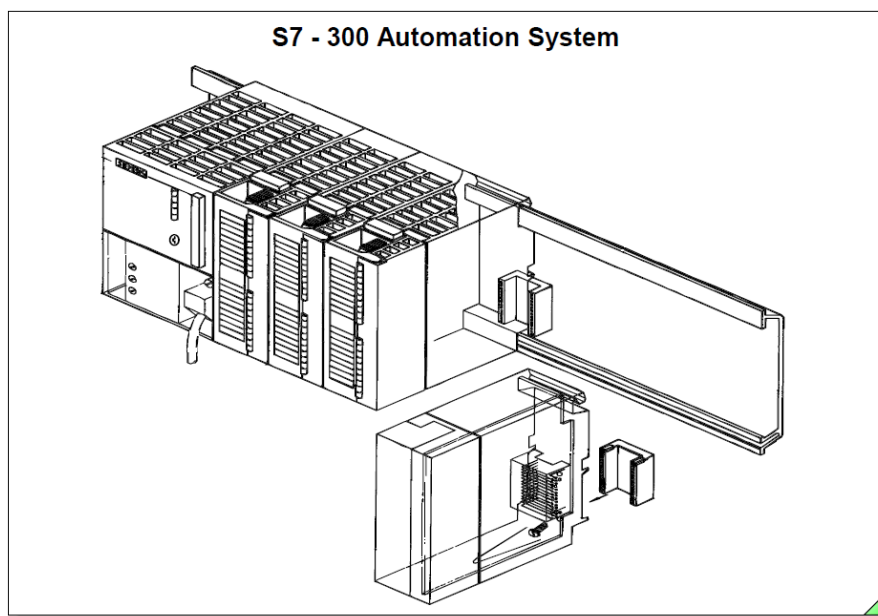
هر چیزی زکاتی دارد و زکات دانش، آموختن آن است به اهلش

محصولات SIMATIC



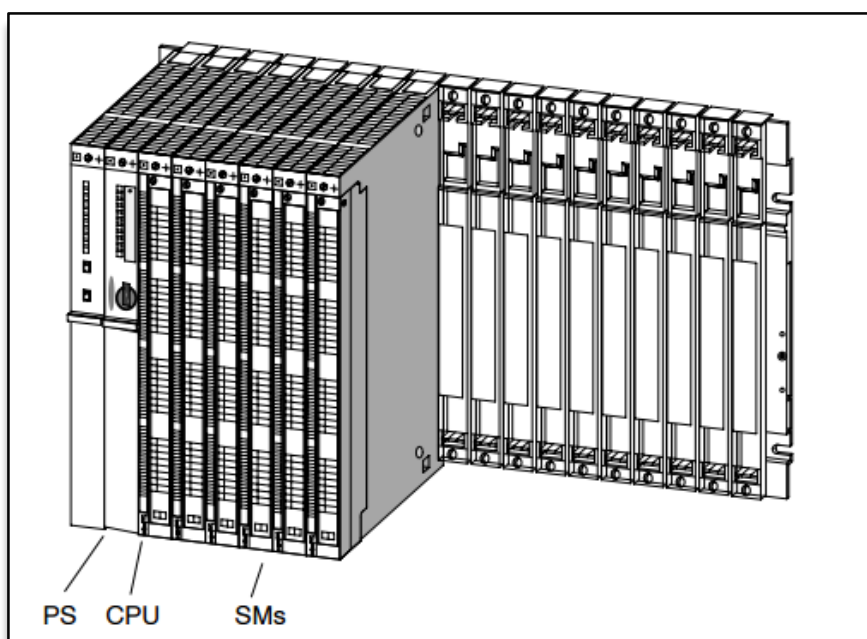
نحوه نصب ماژول های S7-300

در PLC S7-300، ماژول ها توسط Bus Connector به صورت سری به هم متصل می شوند.

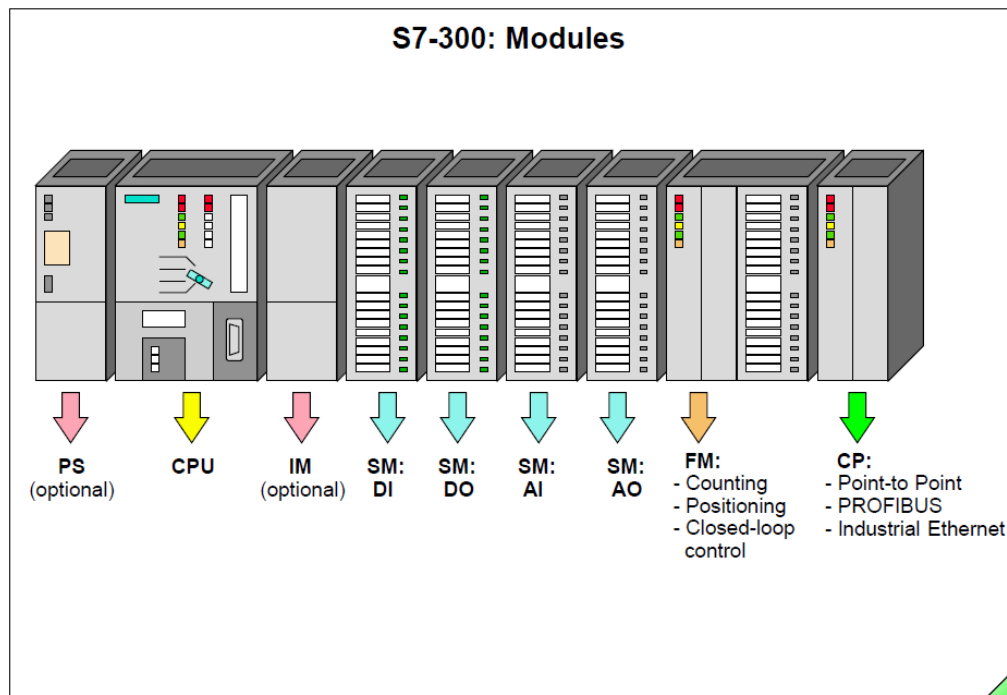


نحوه ی قرار گرفتن ماژول های S7-400

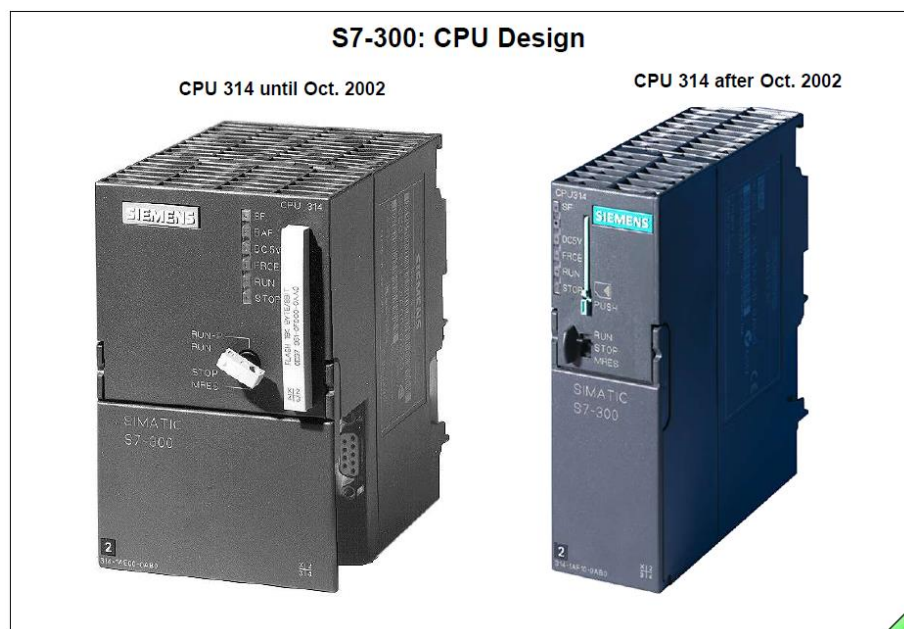
در S7-400 تمامی ماژول ها بر روی رک قرار می گیرند. در واقع ارتباط بین CPU با سایر ماژول ها توسط رک برقرار می شود. در این ساختار تغذیه ماژول ها نیز در رک وجود دارد.



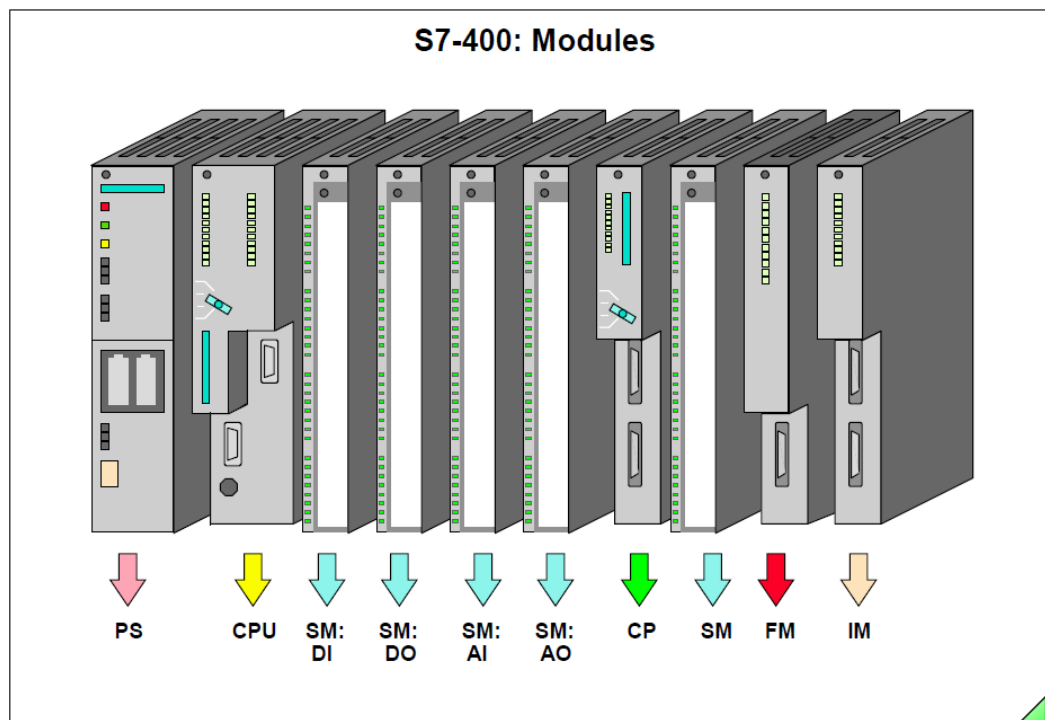
ماژول های S7-300



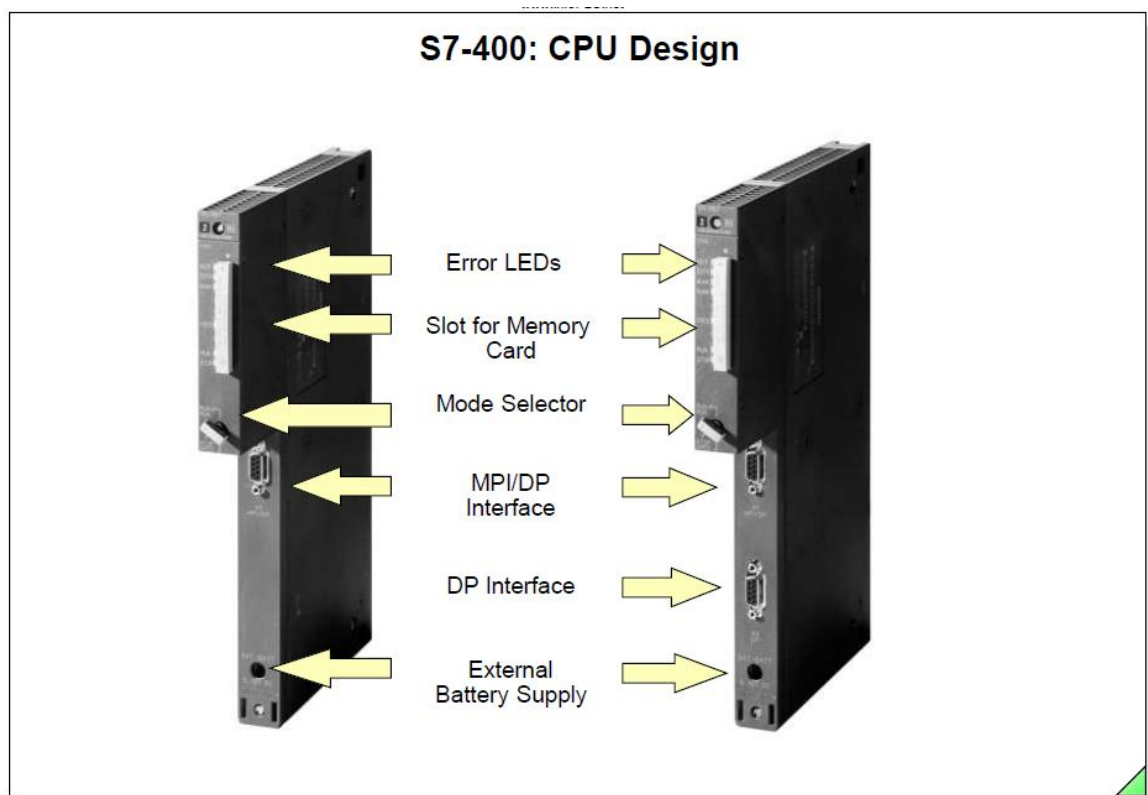
CPU های قدیم و جدید S7-300



ماژول های S7-400

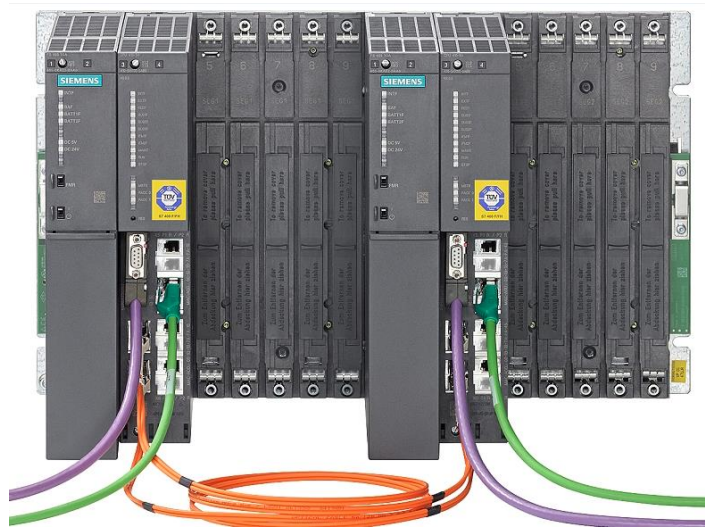


ساختار CPU در S7-400

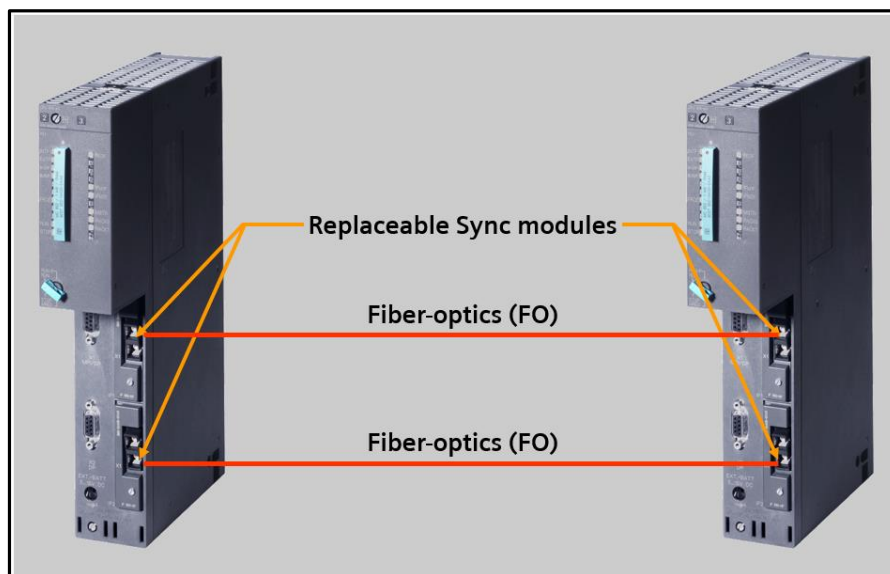


ساختار سیستم H در S7-400

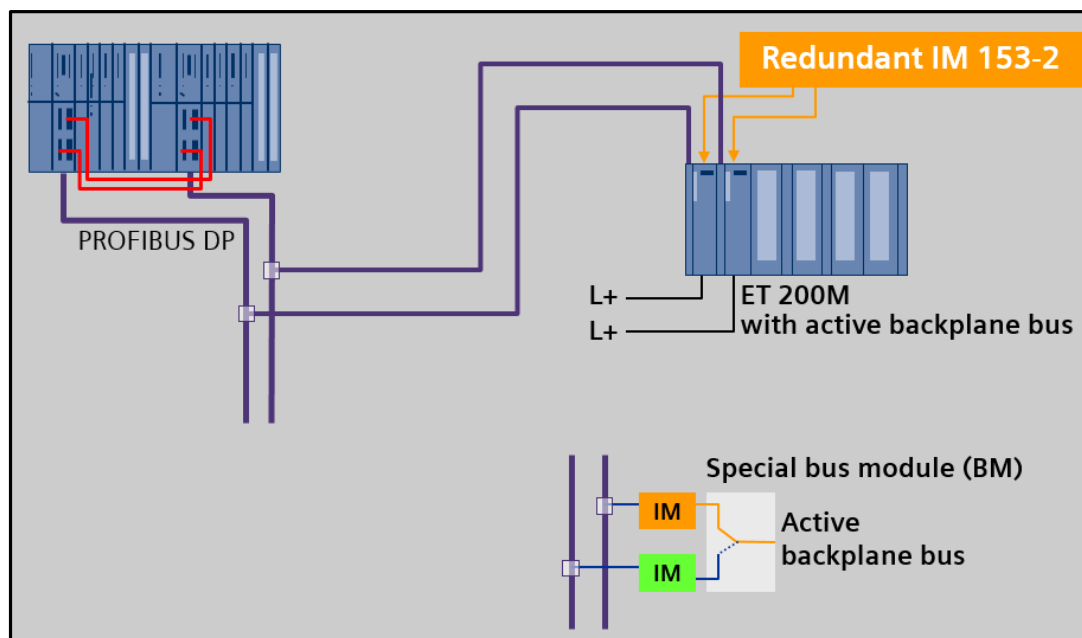
در این ساختار از دو CPU در یک رک استفاده می شود. یک CPU در حال اجرای برنامه و کنترل پروسه می باشد. این کنترلر Master می باشد. زمانی که بر روی این CPU خطایی رخ دهد، سیستم رزرو یا Standby در زمان کوتاهی وارد مدار میشود و کار کنترل را انجام میدهد. این زمان در سیستم Redundant سخت افزاری بسیار کوتاه و در حد چند میلی ثانیه می باشد. سیستم افزونگی سخت افزاری را می توان در برخی از CPU های S7-400 که دارای پسوند H می باشند، پیاده سازی نمود. ایستگاه I/O در این حالت می تواند یک ایستگاه ET200M همانند ET200M با قابلیت اتصال به سیستم Redundant باشد.



در این ساختار دو CPU توسط فیبر به یکدیگر متصل میشوند. عملیات سنکرون سازی بین دو CPU توسط ماژول های سنکرون سازی یا Sync انجام میشود.



اتصال سیستم Redundant به ایستگاه ET200M



آشنایی با ماژول پاور در S7-300

کارت پاور زیمنس تیپ S7-300 دارای تغذیه ی ورودی 220VAC و خروجی 24VDC می باشد. جهت تغذیه ماژول های S7-300 الزامی به استفاده از منبع تغذیه با برند زیمنس نمی باشد و منبع تغذیه با هر برندی با داشتن شرایط لازم برای تغذیه کارت می تواند مورد استفاده قرار گیرد. در صورتی که بخواهیم منبع تغذیه روی ریل استاندارد S7، در کنار سایر ماژول ها قرار گیرد، می توان از ماژول پاور زیمنس و تیپ S7-300 استفاده می کنیم. روی این منبع تغذیه سوئیچ ON/OFF برای روشن و خاموش کردن پاور تعبیه شده است. همچنین LED مربوط به نمایش صحت سطح سیگنال خروجی با رنگ سبز وجود دارد که در شرایط کار عادی روشن می باشد. ترمینال های ورودی و خروجی کارت پاور در شکل زیر نشان داده شده است.



آشنایی با CPU های خانواده S7-300

SIMATIC S7-300 Compact-CPU

S7-300 CPUs module overview

-مشخصات CPU های کامپکت

Overview

- ✓ Description
- ✓ SIMATIC S7-300 CPU 312C
- ✓ SIMATIC S7-300 CPU 313C
- ✓ SIMATIC S7-300 CPU 313C-2 PtP
- ✓ SIMATIC S7-300 CPU 313C-2 DP
- ✓ SIMATIC S7-300 CPU 314C-2 PtP
- ✓ SIMATIC S7-300 CPU 314C-2 DP
- ✓ SIMATIC S7-300 CPU 314C-2 PN/DP

❖ دارای پسوند C می باشند.

❖ دارای تعدادی I/O به صورت Onboard می باشند.

❖ در این CPU ها نیز امکان توسعه وجود دارد.

SIMATIC S7-300 CPU 312C

For small applications with high requirements in terms of processing power

- MPI-Schnittstelle onboard
- Technological functions:
 - Counting
 - Frequency measurement
 - Pulse width modulation
 - Pulse generator
- 10 digital inputs
- 6 digital outputs

**SIMATIC S7-300 CPU 313C**

CPU for installations with high requirements in terms of processing power and response time.

- MPI interface onboard
- Technological functions:
 - Counting
 - Closed loop control
 - Frequency measurement
 - Pulse width modulation
 - Pulse generator
- 24 digital inputs
- 16 digital outputs
- 4 analog inputs
- 2 analog outputs



SIMATIC S7-300 CPU 313C-2 PtP

CPU for installations with high requirements in terms of processing power and response time.

- MPI- interface onboard
- Point to point link interface
- Technological functions:
 - Counting
 - Closed loop control
 - Frequency measurement
 - Pulse width modulation
 - Pulse generator
- 16 digital inputs
- 16 digital outputs

**SIMATIC S7-300 CPU 313C-2 DP**

CPU for installations with high requirements in terms of processing power and response time.

- MPI interface onboard
- PROFIBUS DP master/slave interface
- Technological functions:
 - Counting
 - Closed loop control
 - Frequency measurement
 - Pulse width modulation
 - Pulse generator
- 16 digital inputs
- 16 digital outputs

**SIMATIC S7-300 CPU 314C-2 PtP**

CPU for installations with high requirements in terms of processing power and response time.

- MPI interface onboard
- Point to point link interface
- Technological functions:
 - Counting
 - Closed loop control
 - Frequency measurement
 - Pulse width modulation
 - Pulse generator
 - Positioning
- 24 digital inputs
- 16 digital outputs
- 4 analog inputs
- 2 analog outputs



SIMATIC S7-300 CPU 314C-2 PN/DP

CPU for machines with high demands of computing power and reaction rate

- combined MPI-PROFIBUS DP-interface onboard
- 2 PROFINET Ports for line structures

Technology functions:

- Counting
- Controlling
- Detecting frequencies
- Pulse width modulation
- Pulse generator
- Positioning

- 24 digital inputs
- 16 digital outputs
- 4 analog inputs
- 2 analog outputs

**در CPU های کامپکت:**

Counting : یعنی مجهز بودن CPU به ورودی های HSC (شمارنده های سرعت بالا) برای شمارش

پالس های فرکانس بالا

Detecting Frequency : یعنی CPU مجهز به فانکشن هایی جهت اندازه گیری فرکانس پالس می باشد.

Pulse Width Modulation : یعنی CPU مجهز به فانکشن، جهت تولید پالس PWM در خروجی

می باشد.

Pulse Generator : یعنی CPU مجهز به خروجی پالس با فرکانس بالا برای کنترل موتور پله ای یا سرو

می باشد.

Positioning : یعنی CPU مجهز به فانکشن جهت استفاده در کاربردهای کنترل موقعیت می باشد.

Positioning می تواند به صورت آنالوگ یا دیجیتال باشد.

در S7-300 همانطور که بیان شد، CPU های کامپکت در Order های مختلفی در دسترس می باشد. CPU

314C-2PN/DP یکی از کامل ترین CPU های کامپکت می باشد.

Technical data: Compact CPUs

N = Product innovations

NEW

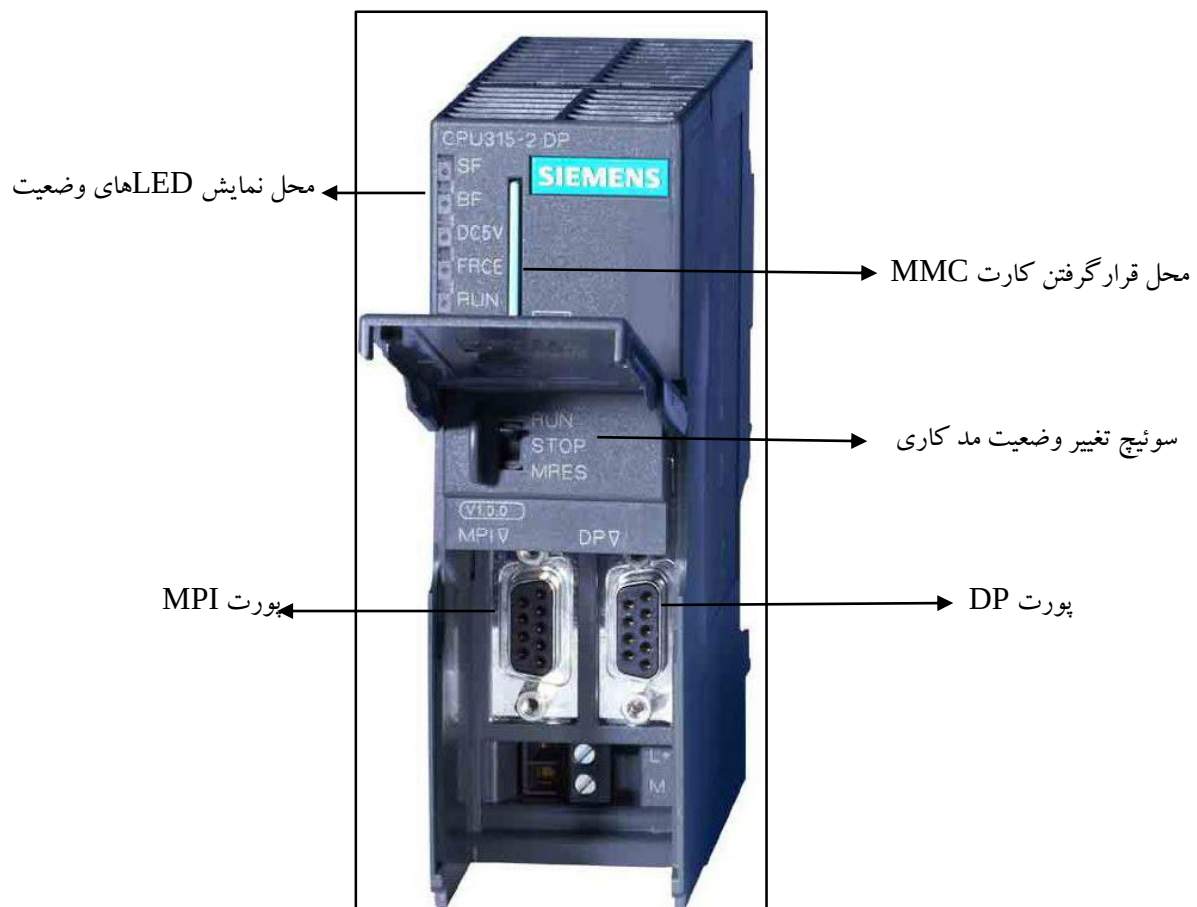
CPU	CPU 312C ¹⁾	CPU 313C ¹⁾	CPU 313C-2 PtP	CPU 313C-2 DP ¹⁾	CPU 314C-2 PtP ¹⁾	CPU 314C-2 DP ¹⁾	CPU 314C-2 PN/DP
Dimensions (mm)	80 x 125 x 130	120 x 125 x 130	80 x 125 x 130 N	80 x 125 x 130 N	120 x 125 x 130		
Required front connector	1 x 40-pin	2 x 40-pin	1 x 40-pin		2 x 40-pin		
Order No. group: 6ES7	312-5BF.	313-5BG.	313-6BG.	313-6CG.	314-6BH.	314-6CH.	314-6EH.
Firmware N	V3.3	V3.3	V3.3	V3.3	V3.3	V3.3	V3.3
Memory							
Work memory N	64 KB	128 KB			192 KB		
Instructions N	21 K	42 K			64 K		
Processing times							
Bit operation N	0.1 µs	0.07 µs			0.06 µs		
Word operations N	0.24 µs	0.15 µs			0.12 µs		
Fixed-point operations N	0.32 µs	0.2 µs			0.16 µs		
Floating-point operations N	1.1 µs	0.72 µs			0.59 µs		
Bit memories/timers/counters							
Bit memory	256 bytes						
S7 timers/counters	256 / 256						
IEC timers/counters	•*)						
Address ranges							
Number of I/Os (bytes)	1 024 / 1 024			2 048 / 2 048	1 024 / 1 024	2 048 / 2 048	
I/O process image	1 024 / 1 024			2 048 / 2 048	1 024 / 1 024	2 048 / 2 048	
Digital channels (central)	266	1 016	1 008		1 016		
Analog channels (central)	64	253	248		253		
Integrated functions							
Counter (incremental enc.)	2, 24 V/10 kHz	3, 24 V/30 kHz			4, 24 V/60 kHz		
Pulse outputs (PCM)	2 channels, max. 2.5 kHz	3 channels, max. 2.5 kHz			4 channels, max. 2.5 kHz		
Frequency measurement	2 channels max. 10 kHz	3 channels, max. 30 kHz			4 channels max. 60 kHz		
Controlled positioning					SFB for positioning, 1 axis via 2 DO, AO		
Integrated "Control" FB		PID controllers					
Integrated inputs/outputs							
Digital inputs	10 x 24 V DC; all channels can be used for process interrupts	24 x 24 V DC; all channels can be used for process interrupts	16 x 24 V DC; all channels can be used for process interrupts		24 x 24 V DC; all channels can be used for process interrupts		
Digital outputs	6 x 24 V DC, 0.5 A	16 x 24 V DC, 0.5 A					
Analog inputs		4 : ± 10 V, 0 ... 10 V, ± 20 mA, 0 / 4 ... 20 mA; 1 : 0 ... 600 Ω, PT100			4 : ± 10 V, 0 ... 10 V, ± 20 mA, 0 / 4 ... 20 mA; 1 : 0 ... 600 Ω, PT100		
Analog outputs		2 : ± 10 V, 0 ... 10 V, ± 20 mA, 0 / 4 ... 20 mA			2 : ± 10 V, 0 ... 10 V, ± 20 mA, 0 / 4 ... 20 mA		
DP interface							
DP master systems int./ CP 342-5	○ / ●			● / ●	○ / ●	● / ●	● / ●
DP slave				●		●	●
Data set gateway ²⁾				●		●	●
PROFINET interface							
PROFINET CBA							●
PROFINET IO							●
PROFINET with IRT							●
Open User Communication (OUC)							
• TCP/IP							●
• UDP							●
• ISO-on-TCP (RFC 1006)							●
Web server							●
PtP interface							
Physics			RS485/422		RS485/422		
Protocol driver			3964 (R), RK512, ASCII		3964 (R), RK512, ASCII		

-مشخصات CPU های ماژولار

- ✓ SIMATIC S7-300 CPU 312
- ✓ SIMATIC S7-300 CPU 314
- ✓ SIMATIC S7-300 CPU 315-2 DP
- ✓ SIMATIC S7-300 CPU 315-2 PN/DP
- ✓ SIMATIC S7-300 CPU 317- 2 DP
- ✓ SIMATIC S7-300 CPU 317-2 PN/DP
- ✓ SIMATIC S7-300 CPU 319-3 PN/DP

Technical data: Standard CPUs N = Product innovations							
CPU	CPU 312	CPU 314 ¹⁾	CPU 315-2 DP ¹⁾	CPU 315-2 PN/DP ¹⁾	CPU 317-2 DP	CPU 317-2 PN/DP ¹⁾	CPU 319-3 PN/DP
Dimensions (mm)	40 x 125 x 130	40 x 125 x 130	40 x 125 x 130	40 x 125 x 130	40 x 125 x 130 N	40 x 125 x 130	120 x 125 x 130
Order No. group: 6ES7	312-1AE.	314-1AG.	315-2AH.	315-2EH.	317-2AK.	317-2EK.	318-3EL.
Firmware	V3.0			V3.2	V3.3 N	V3.2	V3.2
Memory							
Work memory	32 KB	128 KB	256 KB	384 KB	1 MB N	1 MB	2 MB
Instructions	10 K	42 K	85 K	128 K	340 K	340 K	680 K
Processing times							
Bit operation	0.1 µs	0.06 µs	0.05 µs		0.025 µs N	0.025 µs	0.004 µs
Word operation	0.24 µs	0.12 µs	0.09 µs		0.03 µs N	0.03 µs	0.01 µs
Fixed-point operation	0.32 µs	0.16 µs	0.12 µs		0.04 µs N	0.04 µs	0.01 µs
Floating-point operation	1.1 µs	0.59 µs	0.45 µs		0.16 µs N	0.16 µs	0.04 µs
Bit memories/timers/counters							
Bit memory	256 bytes		2 048 bytes		4 096 bytes		8 192 bytes
S7 timers/counters	256 / 256				512 / 512		2 048 / 2 048
IEC timers/counters	• ¹⁾						
Address ranges							
Number of I/Os (bytes)	1 024 / 1 024		2 048 / 2 048		8 192 / 8 192		
Process image I/O (bytes), max.	1 024 / 1 024		2 048 / 2 048		8 192 / 8 192		
Digital channels (central)	256	1 024					
Analog channels (central)	64	256					
DP interfaces							
DP master systems internal / CP 342-5	○ / ●		● / ●				
DP slaves			●				
Data set gateway ²⁾			●	●	●	●	
PROFINET interface							
PROFINET CBA				●		●	
PROFINET IO				●		●	
PROFINET with IRT				● ³⁾		● ³⁾	
Open User Communication (OUC)							
• TCP/IP				●		●	
• UDP				●		●	
• ISO-on-TCP (RFC 1006)				●		●	
Web server				●		●	

آشنایی با ماژول CPU



وضعیت LED های CPU

SF: این LED وضعیت خطاهای سیستمی شامل خطای سخت افزاری و نرم افزاری را با رنگ قرمز گزارش می کند.

BF: این LED وضعیت خطاهای مربوط به شبکه را از لحاظ سخت افزاری و نرم افزاری با رنگ قرمز گزارش می کند.

DC5V: این LED وضعیت صحت و سلامت سطح ولتاژ 5V داخلی باس را با رنگ سبز گزارش می کند.

FRCE: این LED در شرایطی که قابلیت نرم افزاری Force حداقل روی یک بیت اعمال شده باشد با رنگ نارنجی ، روشن می شود.

کارت حافظه S7-300

در سری جدید PLC های S7-300 حافظه ی Load Memory داخلی وجود ندارد و این کارت به عنوان حافظه بارگذاری جهت ذخیره ی برنامه ی کاربر استفاده می شود. بنابراین وجود این کارت در CPU های جدید (از سال 2002 به بعد) الزامی می باشد. با توجه به اینکه این کارت از نوع فلش می باشد، با قطع تغذیه و خارج نمودن آن از اسلات مربوطه بر روی CPU، اطلاعات داخل آن از بین نمی رود.



کارت حافظه MMC در ظرفیت های مختلف ارائه شده است.

Bezeichnung	Order no.
SIMATIC Micro Memory Card 64 KB	6ES7953-8LF20-0AA0
SIMATIC Micro Memory Card 128 KB	6ES7953-8LG20-0AA0
SIMATIC Micro Memory Card 512 KB	6ES7953-8LJ30-0AA0
SIMATIC Micro Memory Card 2 MB	6ES7953-8LL31-0AA0
SIMATIC Micro Memory Card 4 MB	6ES7953-8LM20-0AA0
SIMATIC Micro Memory Card 8 MB	6ES7953-8LP20-0AA0

در CPU های مختلف S7-300 ماکزیمم فضای بارگذاری مقداری مشخص می باشد. در جدول زیر به برخی از این موارد اشاره شده است.

CPU	ORDER NO.	TYPE OF MEMORY CARD
CPU 312	6ES7312-1AD10-0AB0	MMC MAX. 4MB
	6ES7312-1AE13-0AB0	MMC MAX. 4MB
CPU 312 IFM	6ES7312-5AC02-0AB0	No Memory Card
	6ES7312-5AC82-0AB0	No Memory Card
CPU 313	6ES7313-1AD02-0AB0	MC MAX. 512KB
	6ES7313-1AD03-0AB0	MC MAX. 4MB
CPU 314C-2 DP	6ES7314-6CF00-0AB0	MMC MAX.4MB

	6ES7314-6CF01-0AB0	MMC MAX.8MB
CPU 315-2DP	6ES7315-2AF02-0AB0	MC MAX. 512KB
	6ES7315-2AF03-0AB0	MC MAX. 4MB
	6ES7315-2AG10-0AB0	MMC MAX. 8MB
CPU 316	6ES7316-1AG00-0AB0	MC MAX. 512KB
CPU 316-2DP	6ES7316-2AG00-0AB0	MC MAX. 4MB
CPU 317-2PN/DP	6ES7317-2FK13-0AB0	MMC MAX. 8MB
CPU 318-2	6ES7318-2AJ00-0AB0	MC MAX. 4MB
CPU 319-3 PN/DP	6ES7318-3EL00-0AB0	MMC MAX. 8MB
	6ES7318-3EL01-0AB0	MMC MAX. 8MB

بخش های مختلف حافظه

فضای حافظه در CPU به ۳ قسمت تقسیم می شود:

Load Memory (حافظه بارگذاری): فضایی از حافظه می باشد که برنامه کاربر شامل بلوک های منطقی و دیتا بلوک ها، تنظیمات سیستمی و شبکه در آن قرار می گیرد. این حافظه را می توان معادل هارد کامپیوتر در نظر گرفت.

Work Memory (حافظه کاری): بخش های اجرایی برنامه از حافظه بارگذاری به حافظه کاری جهت اجرا منتقل می شوند. این حافظه را می توان معادل RAM کامپیوتر در نظر گرفت.

System Memory (حافظه سیستمی): این فضا شامل بخش های زیر می باشد:

- T (تایمرها)
- C (کانترها)
- M (حافظه)
- PII و PIO (حافظه مربوط به تصاویر ورودی و خروجی)
- L STACK (دیتاهای محلی)

فضاهای حافظه بارگذاری در S7-300 جدید و قدیم و S7-400 طراحی متفاوتی دارند که در بحث های بعدی با جزئیات این موضوع آشنا می شویم.

سوئیچ تعیین وضعیت مد کاری

توسط این سوئیچ حالت کاری CPU از حالت RUN $\rightarrow$ STOP و برعکس تغییر می کند.

مد Startup :

وقتی توسط کلید تعبیه شده بر روی CPU و یا از طریق نرم افزار، CPU به مد RUN سوئیچ می شود، در واقع ابتدا وارد مد Startup می شود. این مد در حالت نرمال، کاملاً گذرا می باشد. در این مد وضعیت سخت افزار چک شده و اگر برنامه ای در بلوکهای Startup نوشته شده باشد، اجرا می شود. در ضمن بسته به تنظیم نوع راه اندازی، ممکن است قسمت هایی یا تمام دیتای حافظه سیستمی پاک شود.

انواع Restart (راه اندازی مجدد CPU)

Warm Restart: در این نوع راه اندازی که در S7-300 و S7-400 قابل اجرایی باشد، اطلاعات فضای حافظه ی سیستمی که به صورت ناپایدار در نظر گرفته شده اند، از بین می روند و بخش های پایدار اطلاعات خود را حفظ می کنند. در ضمن در این نوع راه اندازی، برنامه کاربر از ابتدا آغاز می شود.

Cold Restart: در این نوع راه اندازی کلیه محتویات حافظه سیستمی چه از نوع پایدار و چه از نوع ناپایدار از بین می روند. این نوع راه اندازی در PLC S7-400 و CPU 318-2 در S7-300 قابل اجرا می باشد.

Hot Restart: در این راه اندازی که مختص PLC S7-400 و CPU318 در S7-300 می باشد، کلیه اطلاعات فضای حافظه ی سیستمی چه از نوع ماندگار یا غیرماندگار حفظ می شوند و برنامه کاربر از همان نقطه ای که قطع شده بود ادامه پیدا می کند.

همچنین با ننگه داشتن سوئیچ در مود MRES با در نظر گرفتن شرایطی CPU ریست می شود. در واقع ریست کردن CPU روی هر سه فضای حافظه ی Load Memory، Work Memory و System Memory تاثیر می گذارد. به عبارت دیگر وقتی CPU ریست می شود، مقادیر حافظه های ذکر شده پاک می شوند. اگر کارت حافظه از نوع RAM باشد، با ریست کردن محتوای آن پاک می شود ولی اگر کارت حافظه از نوع FLASH باشد، محتوای آن پاک نمی شود.

پورت MPI

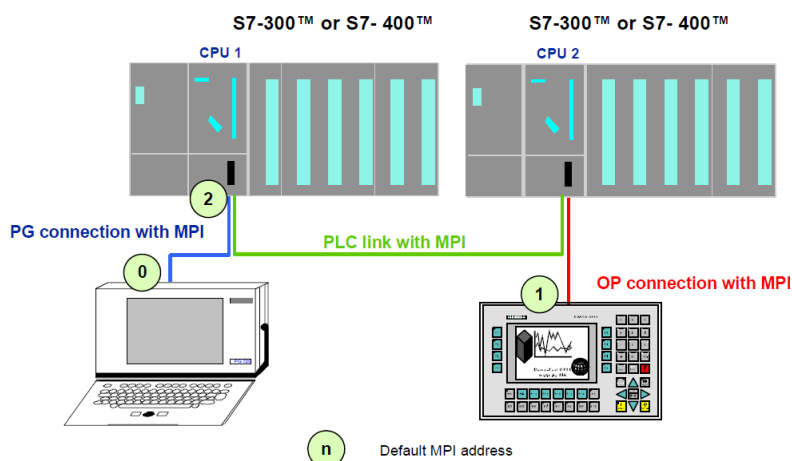
MPI شبکه استاندارد و مختص زیمنس است که روی تمام PLC های سری S7-300 و S7-400 وجود دارد. برای این درگاه ارتباطی سه کاربرد اصلی بیان می شود:

۱- جهت برنامه ریزی برای اتصال کابل PC Adaptor MPI

۲- جهت اتصال به پانل های صنعتی (HMI)

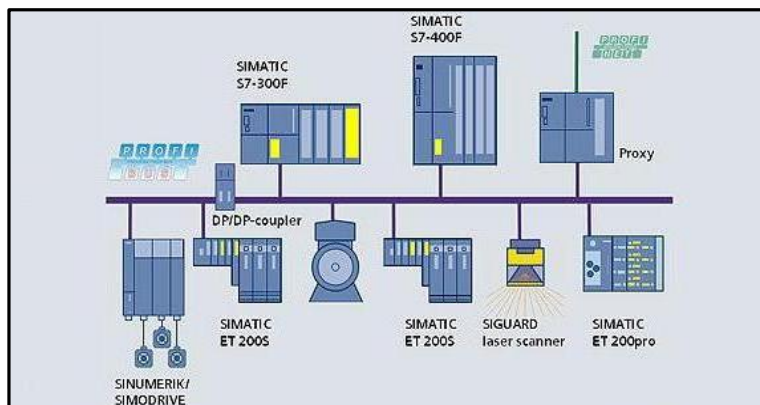
۳- جهت اتصال یک PLC S7 به PLC S7 دیگر

Networking with MPI

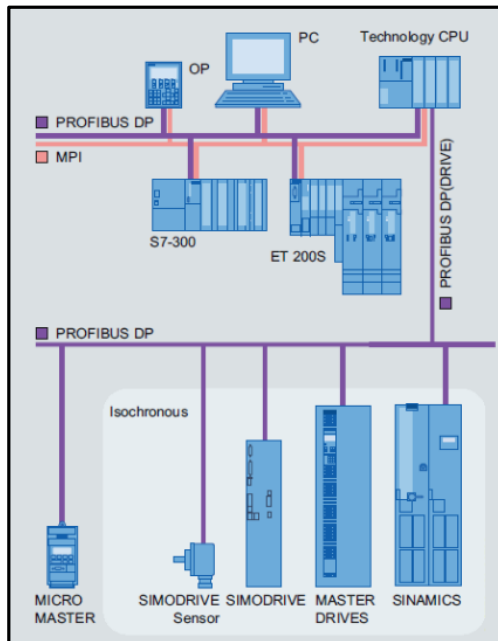


پورت DP

توسط این پورت، PLC S7 می تواند به شبکه پروفیباس (نسخه DP) متصل شود. البته شبکه پروفیباس در ۳ نسخه FMS، DP و PA عرضه شده است.



برخی از مهمترین ایستگاه هایی که در این شبکه قرار می گیرند:



- PLC
- CNC
- HMI
- PC
- DRIVE
- ET

- تجهیزات ابزار دقیق مجهز به اینترفیس شبکه
- تجهیزات قدرت مجهز به اینترفیس شبکه

ET ها یا همان Remote I/O ها یکی از مهمترین ایستگاه هایی هستند که می توانند در شبکه پروفیباس قرار گیرند. وظیفه Remote I/O ها جمع آوری سیگنال از سطح پلنت و انتقال آن به یک CPU می باشد. البته استفاده از ET در پروژه ها می تواند به دلایل مختلفی باشد. شرکت زیمنس ET های متنوعی را طراحی و عرضه نموده است.



کابل شبکه پروفیاس یک کابل دو رشته مسی می باشد.

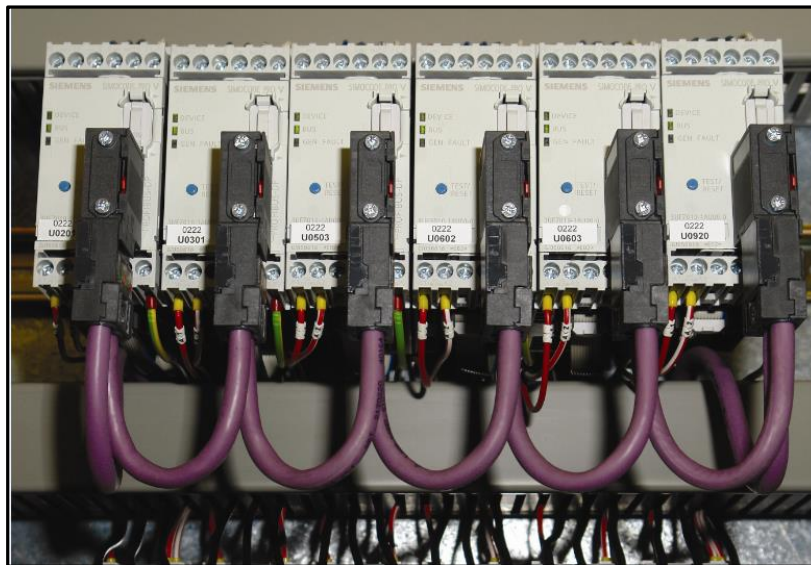


ارتباط این کابل با پورت DP توسط کانکتورهای مطابق شکل زیر برقرار میشود.

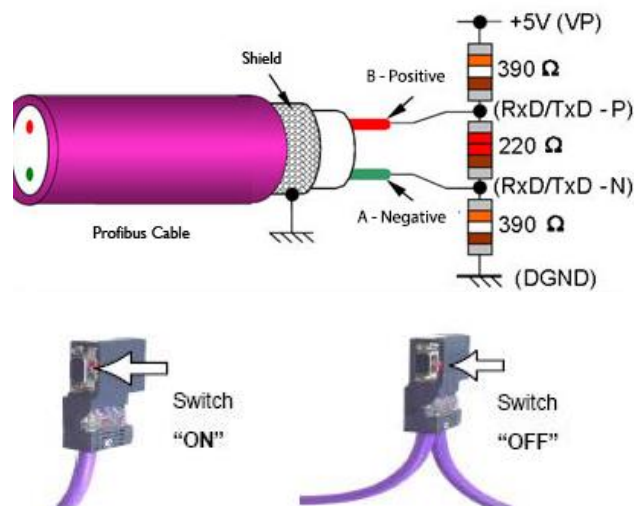


بر روی این کانکتور دو محل ورود و خروج جهت اتصال به ایستگاه قبلی و بعدی تعبیه شده است. توسط کابل مسی می توان تنها توپولوژی باس را بدون استفاده از تجهیز خاصی طراحی و اجرا نمود. در توپولوژی باس، ایستگاه ها به صورت سری به یکدیگر متصل می شوند.

در شکل زیر این توپولوژی را مشاهده می کنید.



بر روی این کانکتور، سوئیچی جهت On و Off کردن ترمیناتور تعبیه شده است. در شکل زیر مدار مقاومتی مربوط به ترمیناتور را مشاهده می کنید. با On کردن سوئیچ، مدار زیر در انتهای باس بر روی مسیر ورودی کانکتور برقرار میشود.



با On کردن این سوئیچ در ایستگاه ابتدا و انتها، باس بسته شده و از بازگشت سیگنال های ناخواسته و بازگشتی جلوگیری می کند. بحث ترمینیت کردن در سرعت های بالا و مسافت های طولانی در کیفیت سیگنال بسیار مهم می باشد.

ماکزیمم سرعت شبکه پروفیباس 12Mbps می باشد. این سرعت با توجه به مسافت محدود می شود.

Baud rate	Maximum segment length
9.6 kbit/s	1 200m
19.2 kbit/s	1 200m
45.45 kbit/s	1 200m
93.75 kbit/s	1 200m
187.5 kbit/s	1 000m
500.0 kbit/s	400m
1.5 Mbit/s	200m
3.0 Mbit/s	100m
6.0 Mbit/s	100m
12.0 Mbit/s	100m

توسط Repeater می توان مسافت کابل را با حفظ سرعت افزایش داد. البته ریپتر کاربردهای دیگری نیز دارد.



افزایش تعداد ایستگاه

افزایش مسافت کابل با حفظ سرعت

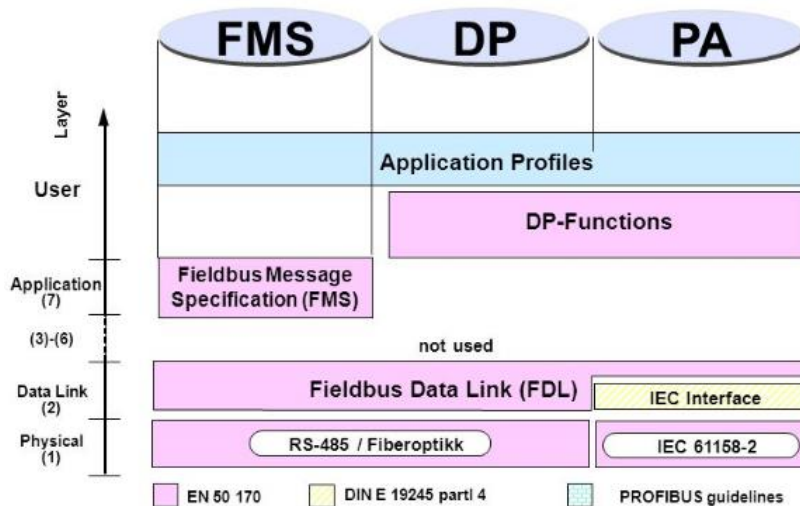
ایزوله کردن دو سگمنت

پیاده سازی توپولوژی Tree

اتصال یک HMI به دو CPU

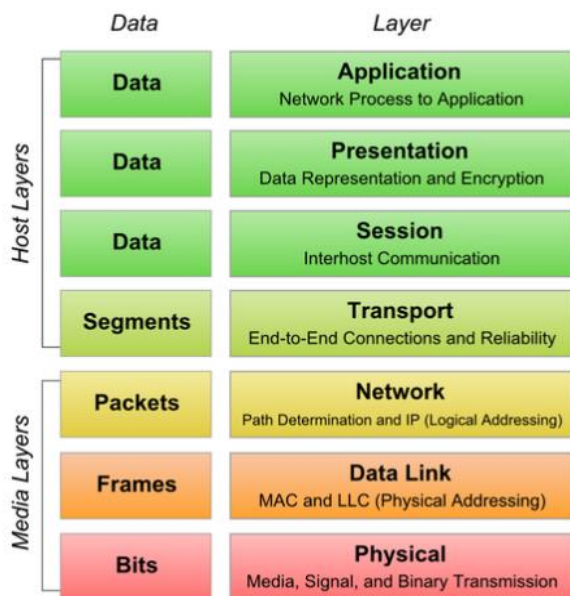
مدل OSI در شبکه پروفیباس:

The PROFIBUS protocols



همانطور که در شکل فوق استفاده می کنید، نسخه DP و PA از لایه های 1 و 2 و در صورت نیاز از یک لایه واسط استفاده می کنند. FMS از لایه های 1، 2 و 7 مدل OSI استفاده می کند. لایه های هفت گانه مدل OSI به صورت زیر می باشند.

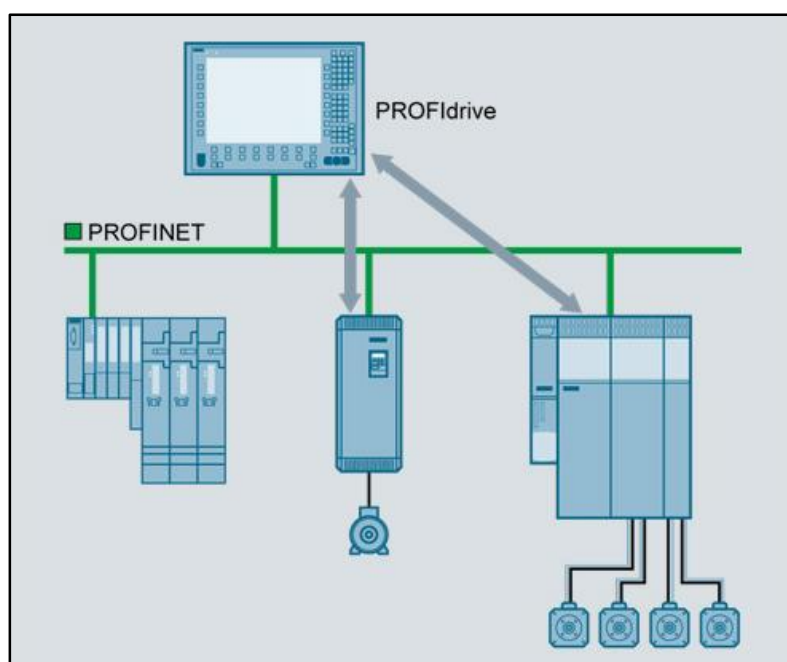
OSI Model



برخی از CPU ها نیز دارای پسوند PN می باشند. این CPU ها مجهز به رابط LAN می باشند. توسط این رابط امکان اتصال به شبکه Profinet امکان پذیر می باشد. این شبکه در بستر Ethernet شبکه ای مناسب برای ارتباط با Remote I/O های جدید می باشد. البته این شبکه به این موضوع محدود نمی شود و روز به روز در حال توسعه می باشد.



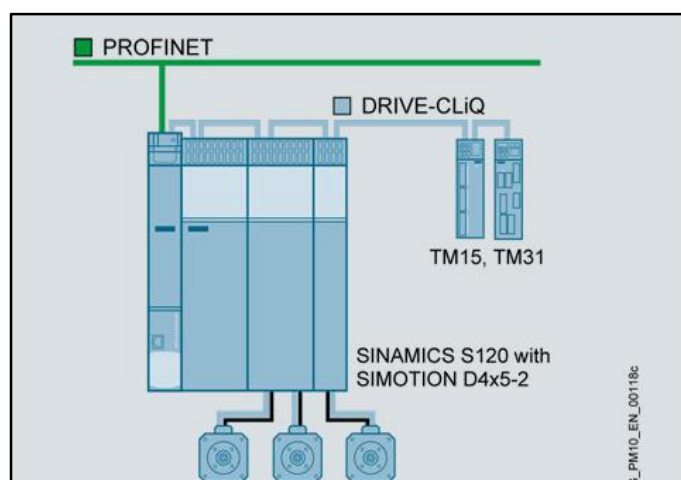
امروزه شرکت زیمنس، ET هایی را با قابلیت اتصال به شبکه Profinet طراحی و عرضه نموده است. همچنین برخی از درایوهای زیمنس نیز قابلیت اتصال به این شبکه را دارا می باشند.



در شکل زیر یک ایستگاه ET200S که مجهز به رابط Profinet می باشد را مشاهده می کنید.



در شکل زیر یک درایو SINAMICS زیمنس که مجهز به رابط Profinet می باشد را مشاهده می کنید.



آشنایی با ماژول های ورودی و خروجی S7-300



کارت های ورودی دیجیتال DI : SM321

کارت های خروجی دیجیتال DO : SM322

کارت های ورودی/خروجی دیجیتال DI/DO : SM323

کارت های ورودی آنالوگ AI : SM331

کارت های خروجی آنالوگ AO : SM332

کارت های ورودی/خروجی آنالوگ AI/AO : SM334

آشنایی با ماژول های ورودی و خروجی S7-400



کارت های ورودی دیجیتال DI : SM421

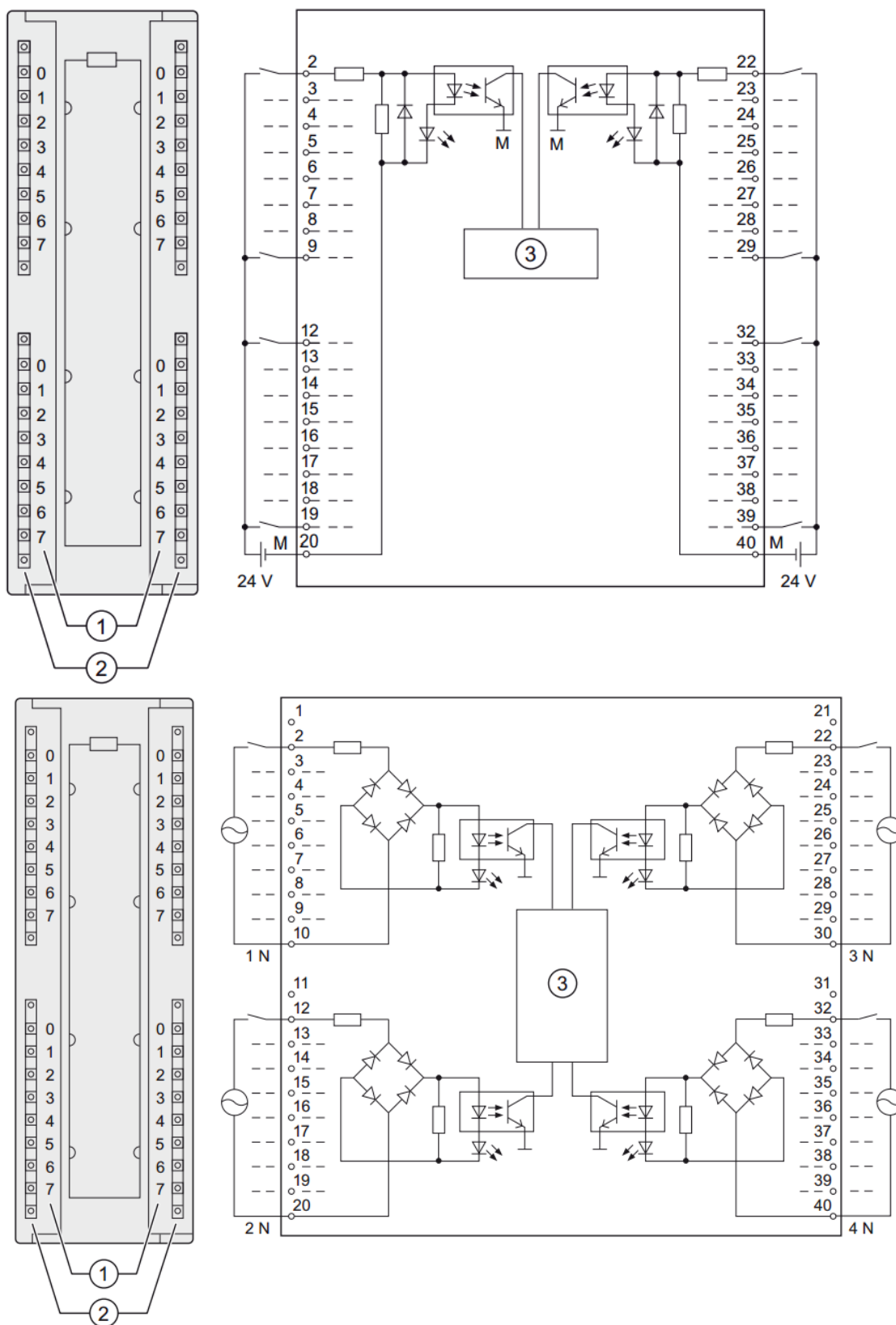
کارت های خروجی دیجیتال DO : SM422

کارت های ورودی آنالوگ AI : SM431

کارت های خروجی آنالوگ AO : SM432

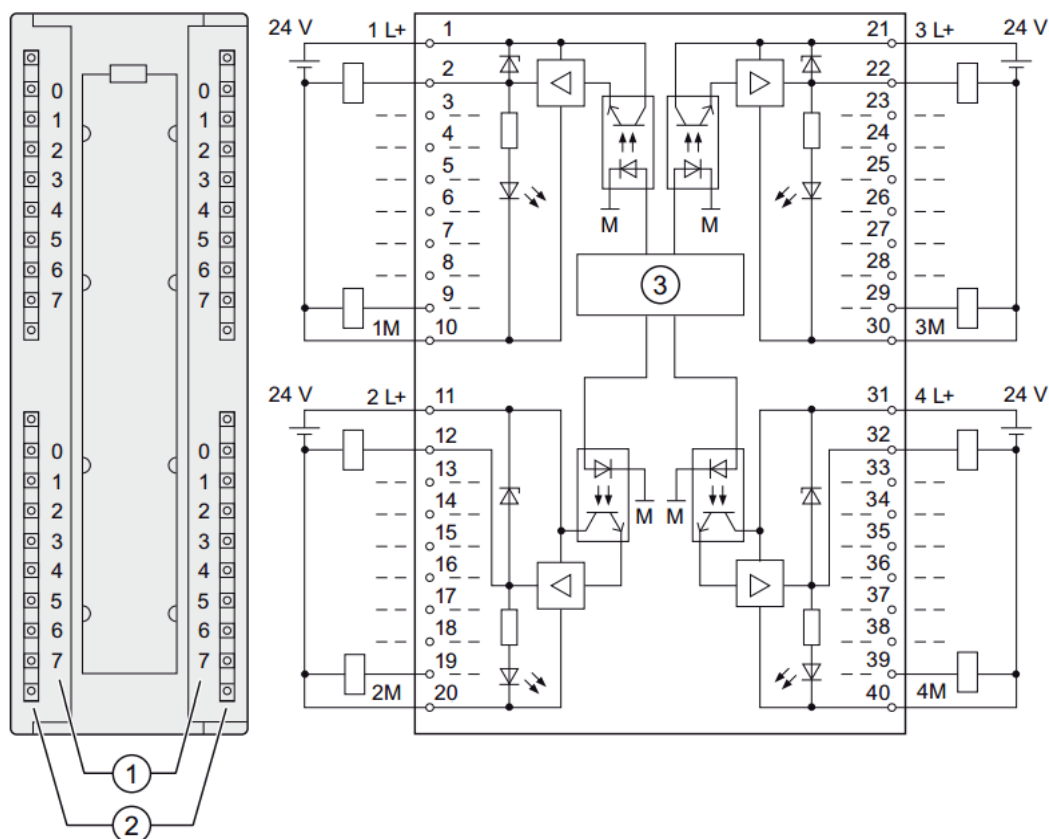
اتصالات در ماژول های DI

Wiring and block diagrams of SM 321; DI 32 x DC 24 V

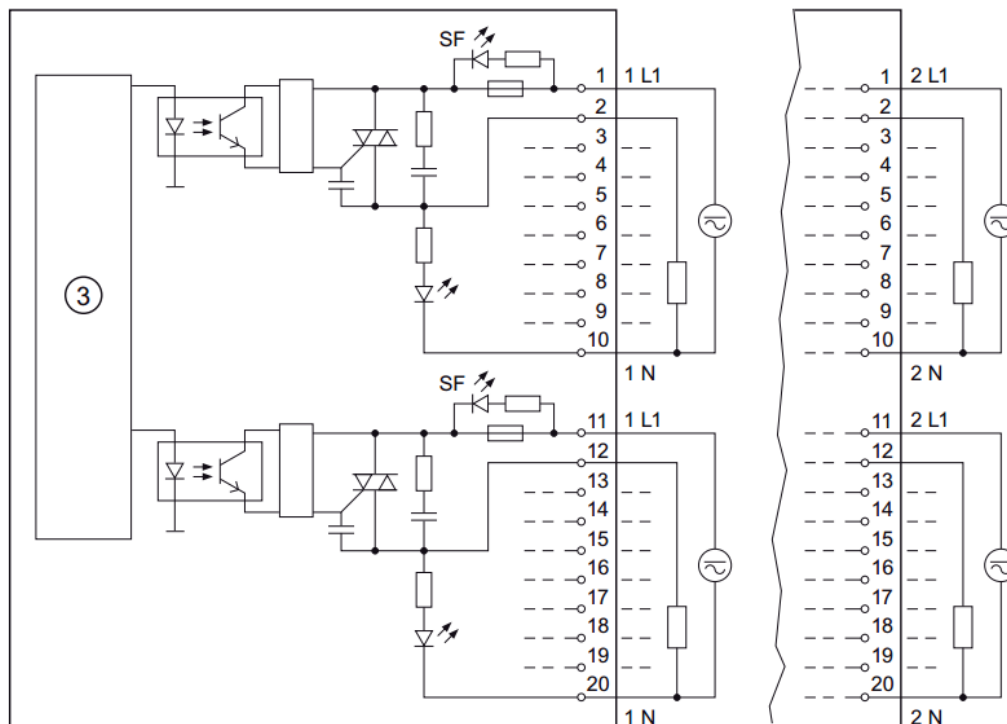


اتصالات در ماژول های DO

Wiring and block diagram of SM 322; DO 32 x DC 24 V/ 0.5 A



Wiring and block diagram of SM 322; DO 32 x AC 120/230 V/1 A

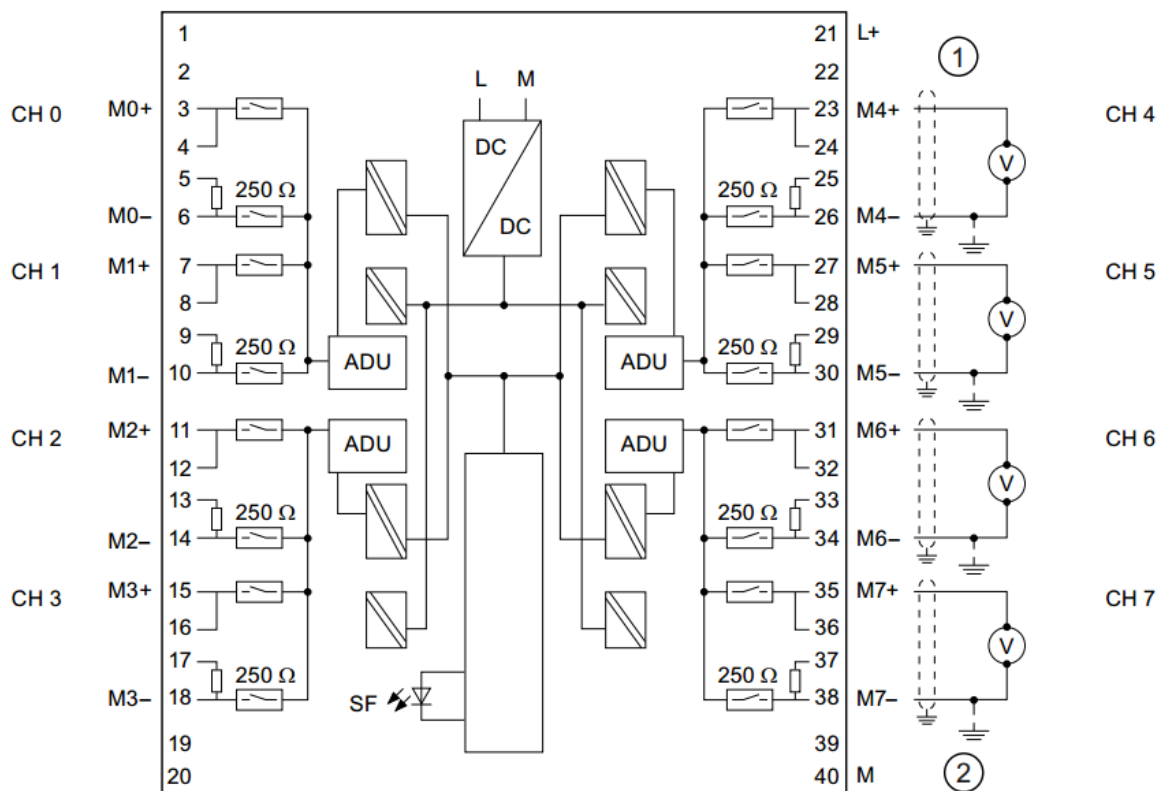


اتصالات در ماژول های AI

نحوه اتصال سیگنال های ولتاژی

Wiring: Voltage and current measurement

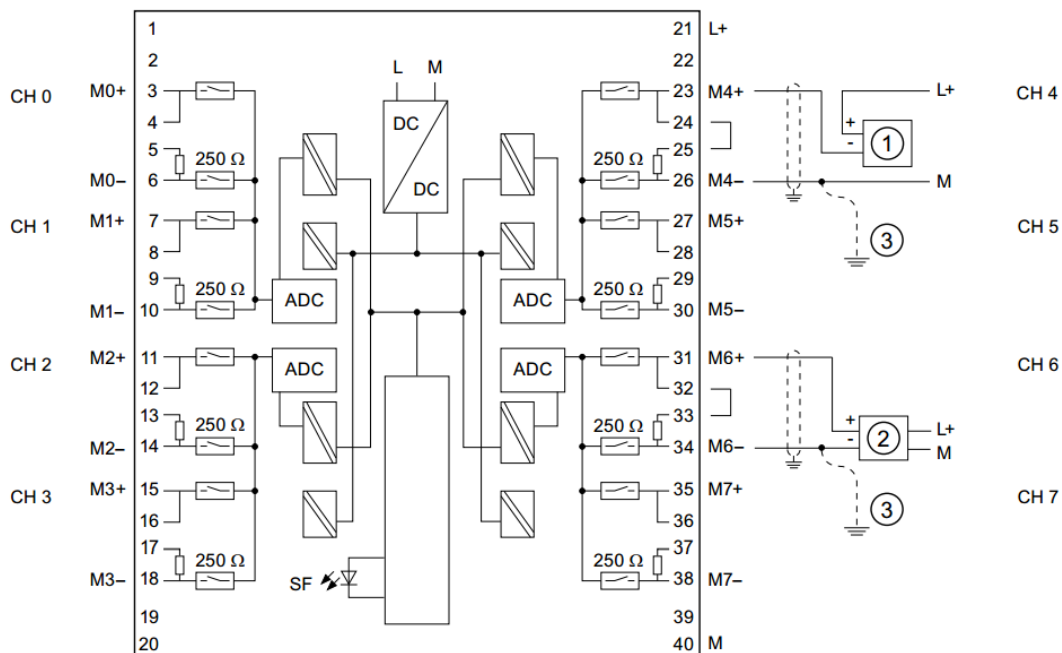
Connection possible at both sides at channels 0 to 7



نحوه اتصال سیگنال های جریانی

Wiring: 2-wire and 4-wire transducer

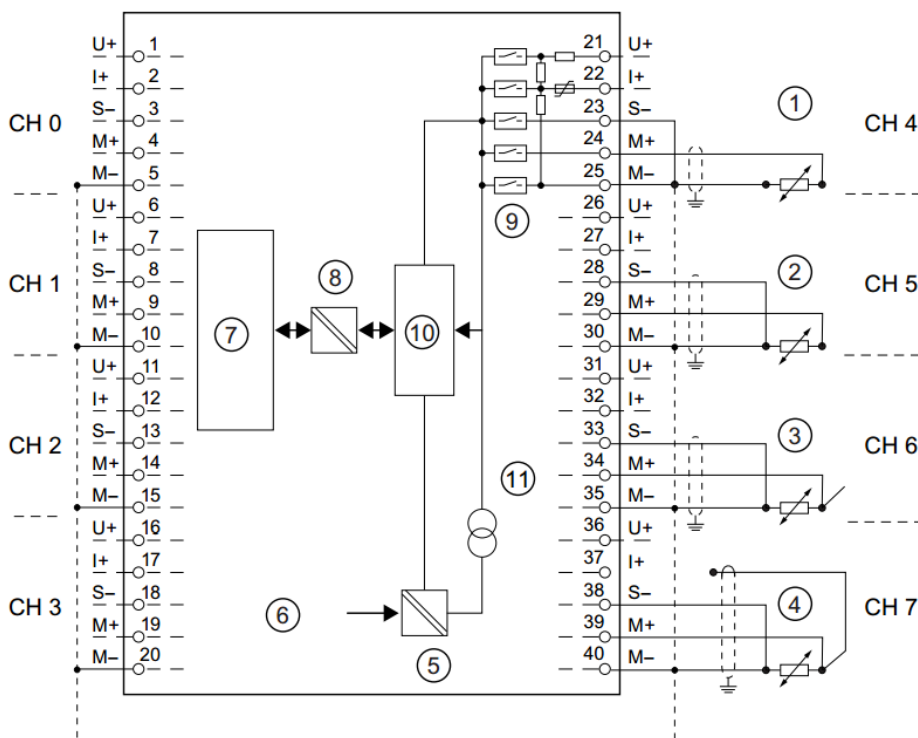
Connection possible at both sides at channels 0 to 7



نحوه اتصال سنسورهای مقاومتی RTD

Wiring: Resistance measurement with 2-, 3- and 4-wire connection

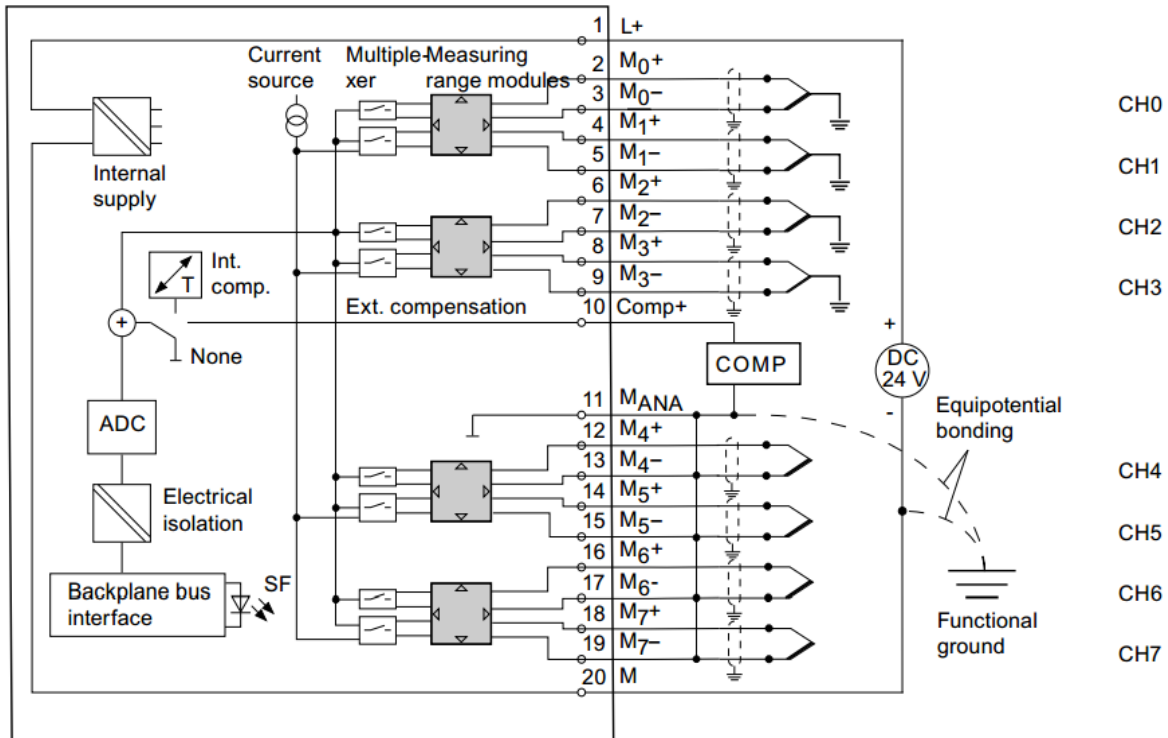
The following connection possibilities also apply to silicon temperature sensors and PTCs.



نحوه اتصال ترموکوپل

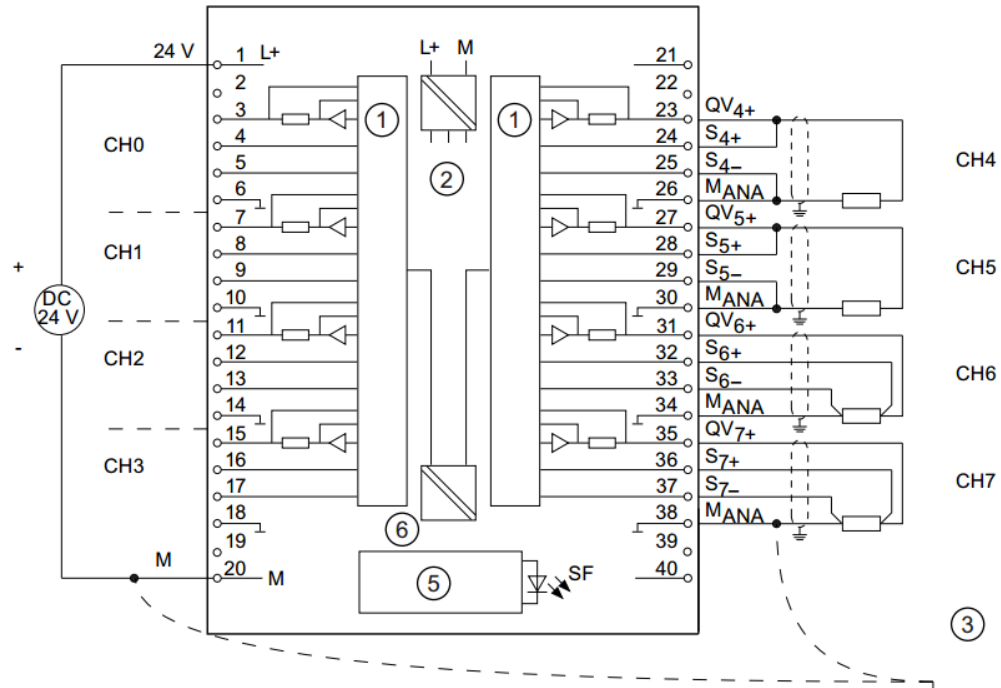
Wiring: Thermocouples with external compensation

Insert a bridge between Comp+ and M_{ANA} when using the internal compensation.



نحوه اتصال بارهای ولتاژی

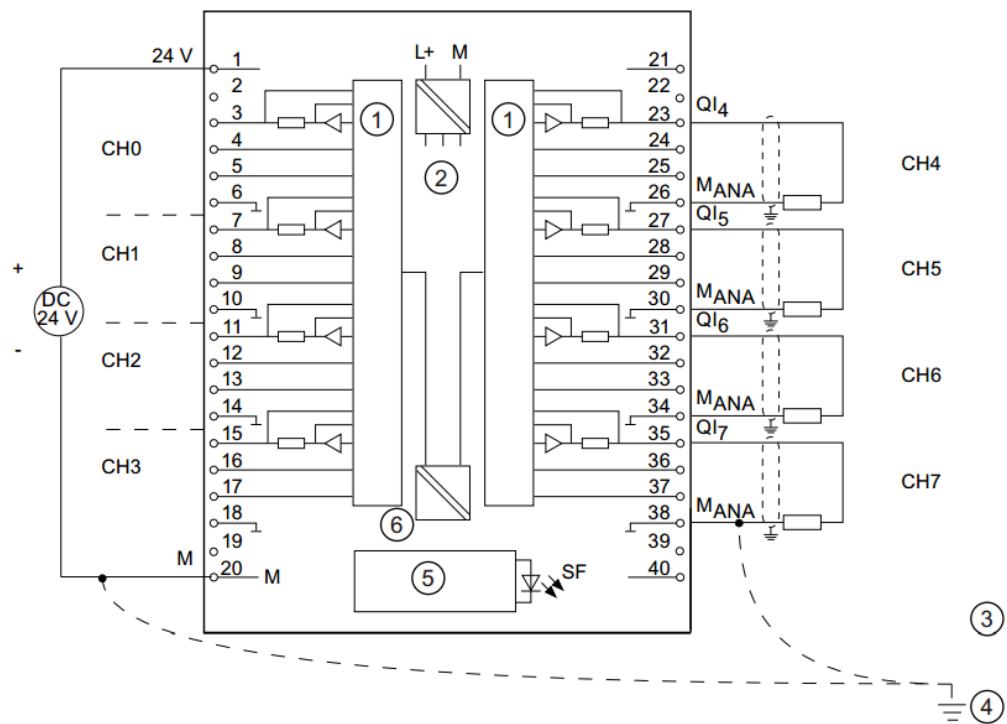
- 2-wire connection, no compensation for line impedance and
- 4-wire connection with compensation for line impedance



اتصالات در ماژول های AO

نحوه اتصال بارهای جریانی

Wiring: Current output



آشنایی با ماژول های ارتباطی IM در S7-300

ماژول IM360S و IM361R جهت توسعه رک تا سه رک افزایشی علاوه بر رک اصلی مورد استفاده قرار می گیرد. محل قرارگیری این کارت در اسلات ۳ روی ریل PLC می باشد.



DI/DO Default Addressing in Multi-Tier Configurations

Rack 3	PS	IM (Receive)	96.0 to 99.7	100.0 to 103.7	104.0 to 107.7	108.0 to 111.7	112.0 to 115.7	116.0 to 119.7	120.0 to 123.7	124.0 to 127.7	
Rack 2	PS	IM (Receive)	64.0 to 67.7	68.0 to 70.7	72.0 to 75.7	76.0 to 79.7	80.0 to 83.7	84.0 to 87.7	88.0 to 91.7	92.0 to 95.7	
Rack 1	PS	IM (Receive)	32.0 to 35.7	36.0 to 39.7	40.0 to 43.7	44.0 to 47.7	48.0 to 51.7	52.0 to 55.7	56.0 to 59.7	60.0 to 63.7	
Rack 0	PS	CPU	IM (Send)	0.0 to 3.7	4.0 to 7.7	8.0 to 11.7	12.0 to 15.7	16.0 to 19.7	20.0 to 23.7	24.0 to 27.7	28.0 to 31.7
Slot 1	2	3	4	5	6	7	8	9	10	11	

IM360S دارای یک رابط و IM361R دارای دو رابط می باشد.

ماژول IM365S/R جهت توسعه رک تنها تا یک رک افزایشی علاوه بر رک اصلی مورد استفاده قرار می گیرد.



مشخصات

S7-300/S7-300F

IM 360/361/365 interface modules**Overview**

- For connecting mounting racks in multi-tier SIMATIC S7-300 configurations
- IM 365: For design of central controller and max. 1 expansion unit.
Limited use of modules in the expansion unit (e.g. no CPs or FMs)
- IM 360/IM 361: For design of central controller and max. 3 expansion units.
No limitation in selection of modules in the expansion unit

Area of application

IM 360/IM 361 and IM 365 modules support a multitier configuration of the S7-300 automation system (CPU 313C, CPU 314 or later), consisting of a central controller (CC) and up to three expansion racks (ERs).

The individual racks are interconnected via the interface modules.

- IM 365:
Central controller and 1 expansion rack each with max. 8 modules; Distance: 1 m
- IM 360/IM 361:
Central controller and max. 3 expansion racks each with max. 8 modules; Distance between two adjacent racks: 4 cm to 10 m

General features

All interfaces have the following common features:

- Compact configuration:
The rugged plastics housing contains the interfaces for the connecting cables.
- Simple mounting:
The interface modules are mounted on the DIN rail (slot 3) and connected to the I/O modules via a bus connector just like any other module.
- Problem-free configuration:
The interface modules are self-configuring. Address assignment is not required.
- Status and fault LEDs

IM 365

The IM 365 is the low-cost expansion for an ER. It contains the following features:

- One of the two IM 365 modules is inserted in the CC, the other into the ER. The modules are connected by a fixed cable 1 m in length.
- Limited use of modules:
The ER is not connected to the C bus (communication bus). Therefore, no C bus nodes (e.g., CPs and FMs, see configuring aids) may be inserted into the ER.
- No separate power supply:
The modules in the ER are powered by the CPU power supply.

IM 360/IM 361

IM 360 and IM 361 are the ideal solution for comprehensive expansions. They contain the following features:

- IM 360 plugs into CC
- IM 361 plugs into each ER
- Separate power supply
Each IM 361 requires an external 24 V DC power supply. This then powers all modules of the respective ER. A PS 307 load power supply can be connected using a power connector.
- No module selection restrictions:

آشنایی با ماژول های FM

از این ماژول ها به منظور انجام مقاصد خاص استفاده می شود. در واقع این ماژول ها می توانند عملیاتی مانند شمارش پالس های فرکانس بالا، تولید پالس های فرکانس بالا و سیگنال های کنترلی جهت کنترل سرو موتور و موتورپله ای ، کنترلرهای PID و... پردازش کنند.

Function modules in detail

- > Counter Module FM 350-1/FM 350-2
- > Positioning Module FM 351
- > Cam Controller FM 352
- > High-Speed Boolean Processor FM 352-5
- > Positioning Module FM 353
- > Positioning Module FM 354
- > Controller Module FM 355
- > Controller Module FM 355-2
- > Positioning Module FM 357-2
- > PROFIBUS-Module IM 174
- > Position decoder module SM 338

SIMATIC S7-300 CPU 312C

For small applications with high requirements in terms of processing power

- MPI-Schnittstelle onboard
- Technological functions:
 - Counting
 - Frequency measurement
 - Pulse width modulation
 - Pulse generator
- 10 digital inputs
- 6 digital outputs



SIMATIC S7-300 CPU 313C

CPU for installations with high requirements in terms of processing power and response time.

- MPI interface onboard
- Technological functions:
 - Counting
 - Closed loop control
 - Frequency measurement
 - Pulse width modulation
 - Pulse generator
- 24 digital inputs
- 16 digital outputs
- 4 analog inputs
- 2 analog outputs



SIMATIC S7-300 CPU 313C-2 PtP

CPU for installations with high requirements in terms of processing power and response time.

- MPI- interface onboard
- Point to point link interface
- Technological functions:
 - Counting
 - Closed loop control
 - Frequency measurement
 - Pulse width modulation
 - Pulse generator
- 16 digital inputs
- 16 digital outputs

**SIMATIC S7-300 CPU 313C-2 DP**

CPU for installations with high requirements in terms of processing power and response time.

- MPI interface onboard
- PROFIBUS DP master/slave interface
- Technological functions:
 - Counting
 - Closed loop control
 - Frequency measurement
 - Pulse width modulation
 - Pulse generator
- 16 digital inputs
- 16 digital outputs

**SIMATIC S7-300 CPU 314C-2 PtP**

CPU for installations with high requirements in terms of processing power and response time.

- MPI interface onboard
- Point to point link interface
- Technological functions:
 - Counting
 - Closed loop control
 - Frequency measurement
 - Pulse width modulation
 - Pulse generator
 - Positioning
- 24 digital inputs
- 16 digital outputs
- 4 analog inputs
- 2 analog outputs



SIMATIC S7-300 CPU 314C-2 PN/DP

CPU for machines with high demands of computing power and reaction rate

- combined MPI-PROFIBUS DP-interface onboard
- 2 PROFINET Ports for line structures
- Technology functions:
 - Counting
 - Controlling
 - Detecting frequencies
 - Pulse width modulation
 - Pulse generator
 - Positioning
- 24 digital inputs
- 16 digital outputs
- 4 analog inputs
- 2 analog outputs

**آشنایی با ماژول های CP**

توسط این ماژول ها، CPU می تواند به شبکه های مختلف متصل شود. در واقع این ماژول های کارت های شبکه برای CPU محسوب می شوند.

Description	Bus system/communication network
CP 340	Point-to-point link
CP 341	Point-to-point link
CP 342-5	PROFIBUS DP
CP 342-5 FO	PROFIBUS DP (fiber optic)
CP 343-1 Lean	PROFINET / Industrial Ethernet
CP 343 1	PROFINET / Industrial Ethernet
CP 343-1 Advanced	Advanced PROFINET / Industrial Ethernet
CP 343-2	AS-Interface (Master)
CP 343-2P	AS-Interface (Master)
CP 343-5	PROFIBUS FMS

Point-to-point link via communication processors

Overview

✓ Description

Description

Point-to-point link via communication processors (CPs) is an extremely powerful and low-cost alternative to bus systems. The advantage of point-to-point links over bus systems is especially significant when only a few (RS 485) devices are to be connected to the SIMATIC S7. The CPs can also easily link third-party systems to the SIMATIC S7. Thanks to the great flexibility of the CPs, different physical transmission media, transmission rates or even customized transmission protocols can be implemented.

The interface modules for the S7-300 are available in three versions, each with one interface for the different physical transmission media.

All point-to-point communications modules use up to 4 protocol drivers which are integrated in the operating system. The CPs, like all other S7 modules, are parameterized from a programming device or PC via the central processing unit (CPU).



SIMATIC S7-300 communication processors

- CP 340 - Low-cost entry version
- CP 341 - Powerful computer connection, loadable protocols

Communication for SIMATIC S7-300

Modular, powerful and with full PROFIBUS functionality: Communication modules for SIMATIC S7-300

CP 342-5

for connecting S7-300, the modular controller for innovative solutions in the manufacturing industry, to PROFIBUS

- Expansion of the process I/O of SIMATIC S7-300 using several PROFIBUS interfaces
- Use in the field as DP-V0 master or DP-V0 slave
- Use in the cell area with cross-network programming device/operator panel communication through S7 routing
- Extensive operator control and monitoring thanks to multiplex function with OP communication
- Flexible use of the process I/O by means of dynamic activation of DP slaves



CP 342-5 FO

Fiber-optic technology is used when

- the environment is severely affected by electromagnetic interference,
- extreme potential differences exist and
- high transmission rates are required.

AS-Interface Master for SIMATIC S7-300

The AS-i master CP343-2 (P) facilitates the connection of AS-i actuators and sensors to the SIMATIC S7-300 or to the ET 200M distributed I/O system. Multiple AS-i masters and AS-i networks can be connected to the SIMATIC S7-300. Up to 62 slaves or up to 992 I/O points can be integrated per AS-i network. Engineering is comfortably realized via drag & drop in the TIA Portal or in SIMATIC STEP 7.

**Communication processors for SIMATIC-S7 300**

With our comprehensive portfolio, we offer the right CP for every requirement, as lean, standard or added value versions for SIMATIC S7-300, the modular controller for innovative system solutions in the manufacturing industry. Connecting controllers via these CPs permits direct integration into Industrial Ethernet networks. The communication processors have multi-protocol capability and support the following protocols:

- PG/OP communication
- S7 communication
- open communication (SEND/RECEIVE)
- PROFINET communication
- IT communication

CP 343-1 Lean: Connects even small applications to Industrial Ethernet

The low-cost connection for programming devices and HMI devices, as well as for communication between S7 programmable controllers.

➤ Product selection and ordering

**CP 343-1: The tried and tested, rugged and secure standard versions**

The efficient, reliable and rugged communication processor that links the S7-300 via Industrial Ethernet, also as a PN IO controller for the connection of PROFINET.

➤ Product selection and ordering



CP 343-1 Advanced: From PROFINET to IT communication

In addition to the functionality of the CP 343-1, the CP 343-1 Advanced offers e-mail functions and user-created web pages and thus optimally supports maintenance and quality assurance. The Internet functions such as FTP even allow connection to the most diverse PC-based systems.



With its integrated security functions of firewall (Stateful Inspection), VPN protocol (IPSec) and protocols for data encryption such as FTPS and SNMPv3, the communications processor protects S7-300 stations and lower-level networks against unauthorized access, as well as protecting data transmission against manipulation and espionage by means of encryption. Therefore, for the S7-300, this CP is the bridge between the field level and the MES level.

➤ Product selection and ordering

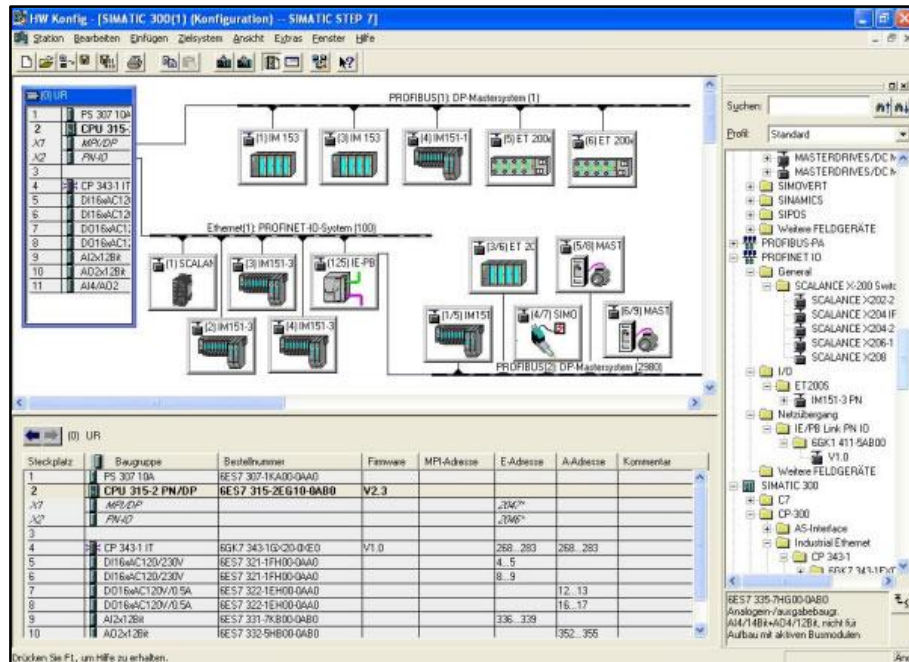
CP 343-1 ERPC: Connects SIMATIC S7-300 controllers with ERP / MES systems

In combination with a loadable firmware expansion, this communication processor offers efficient access via Industrial Ethernet to databases across all corporate levels without laborious programming of the controller.



➤ Product selection and ordering

فصل دوم: آشنایی با نرم افزار SIMATIC MANAGER

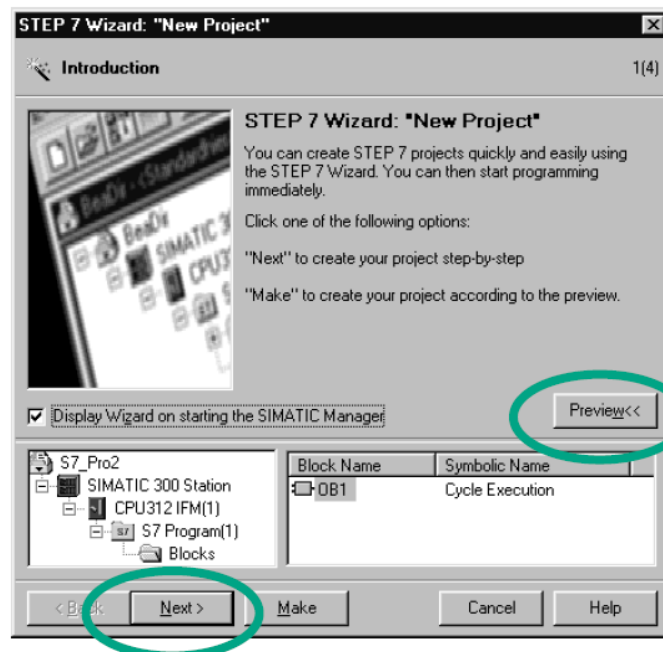


مراحل ساخت پروژه:

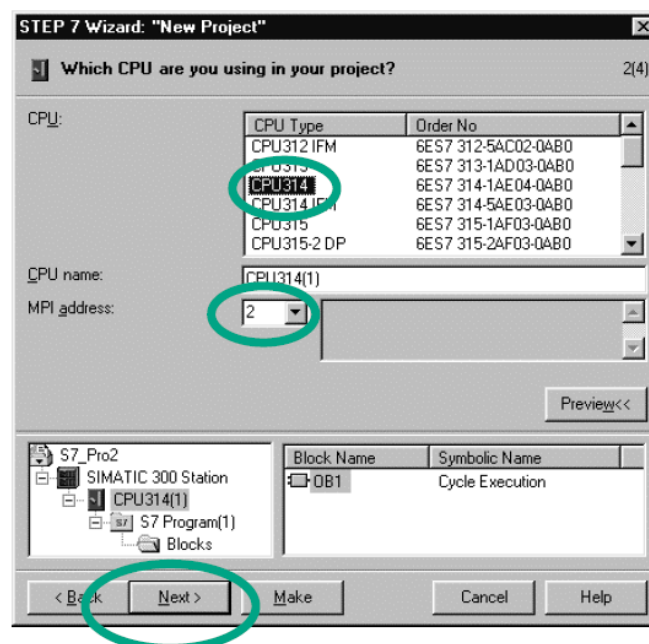
۱- دابل کلیک بر روی آیکون نرم افزار



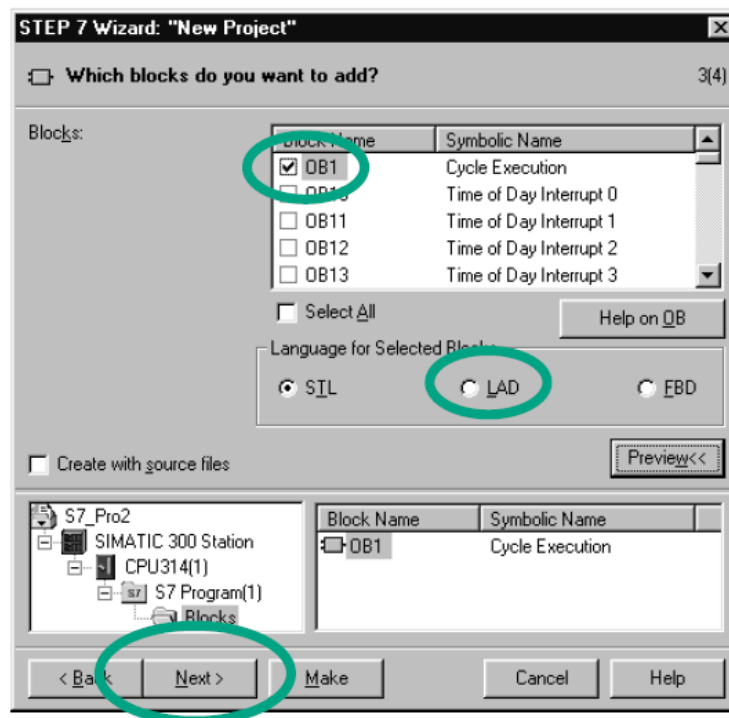
SIMATIC Manager



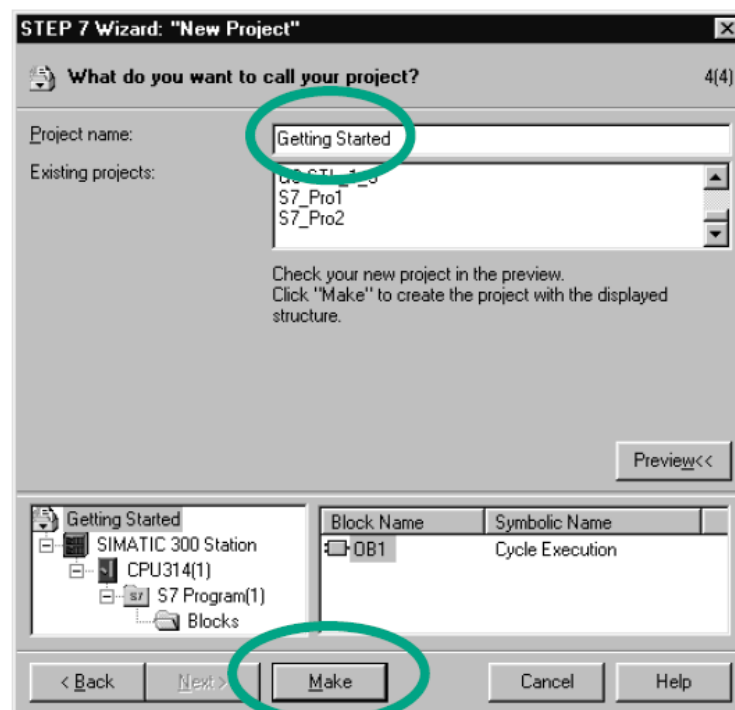
۲- انتخاب نوع CPU و آدرس پورت MPI



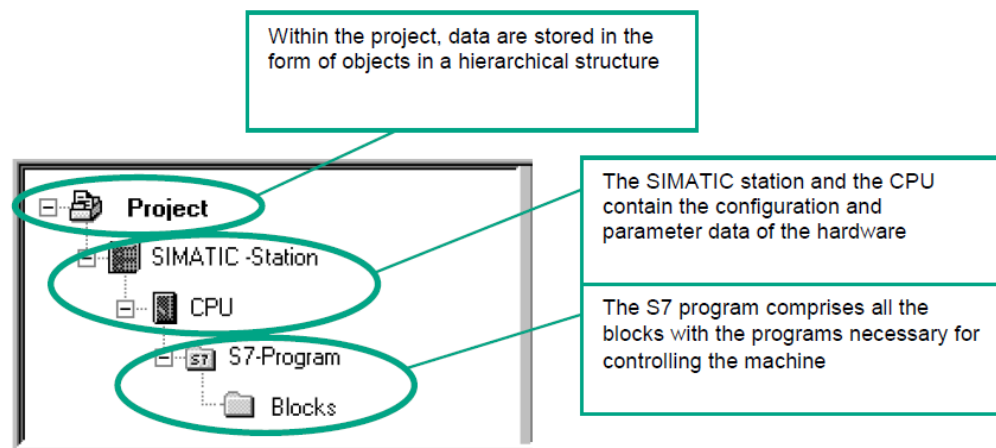
۳- انتخاب بلوک برنامه نویسی و روش برنامه نویسی



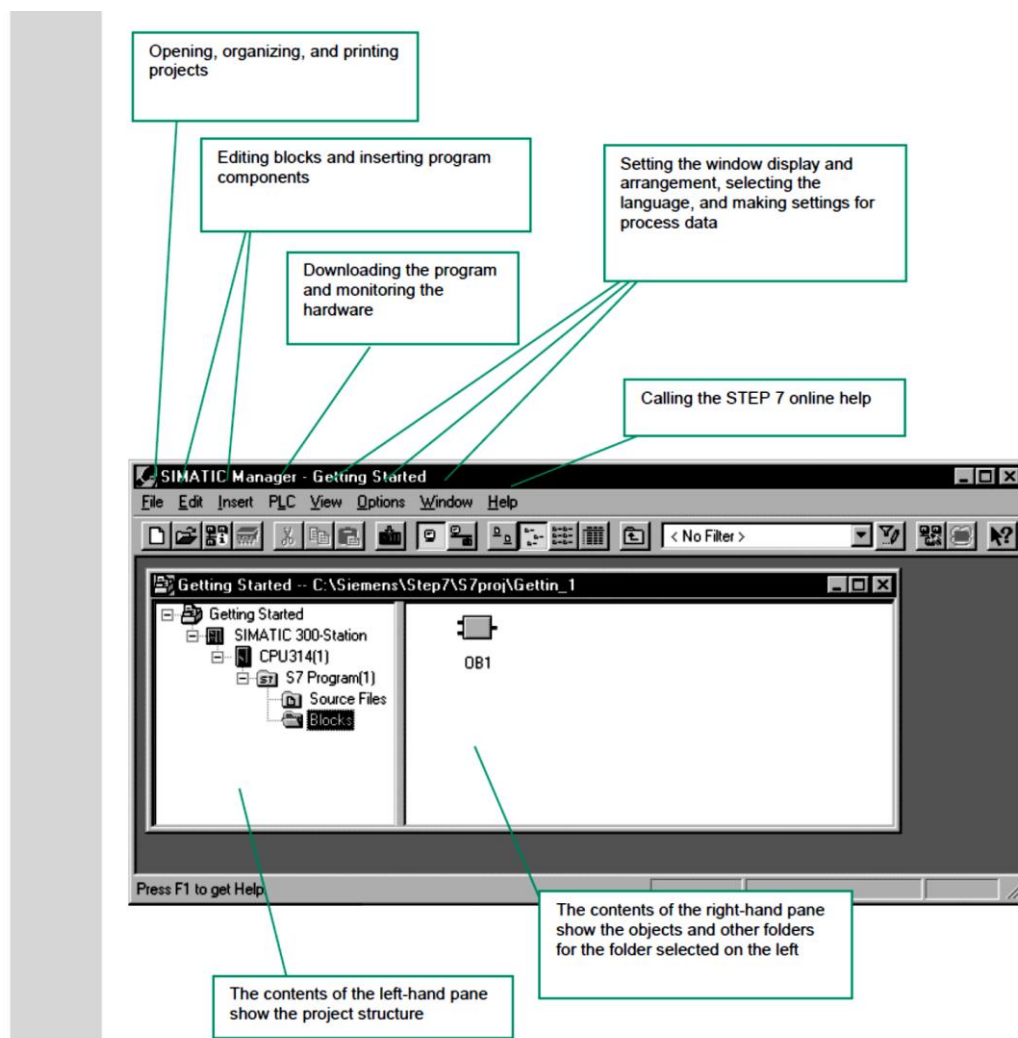
۴- وارد کردن نام برای پروژه و کلیک بر روی گزینه FINISH



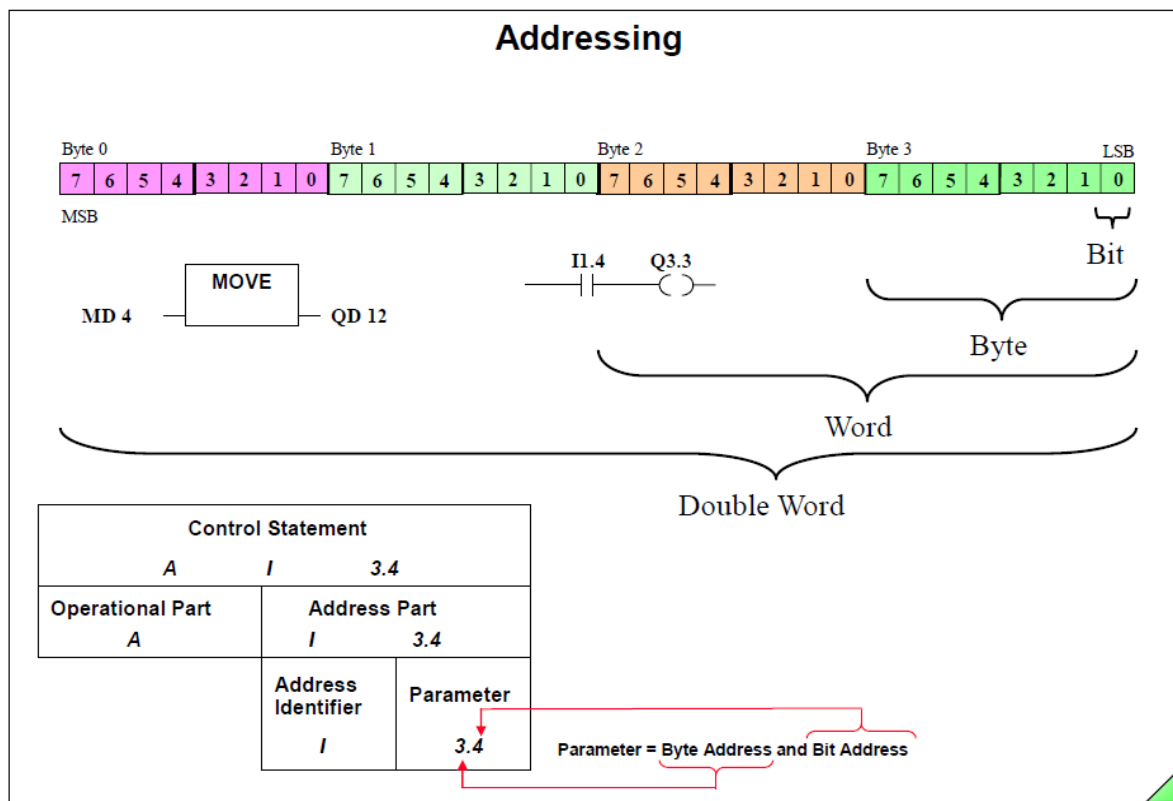
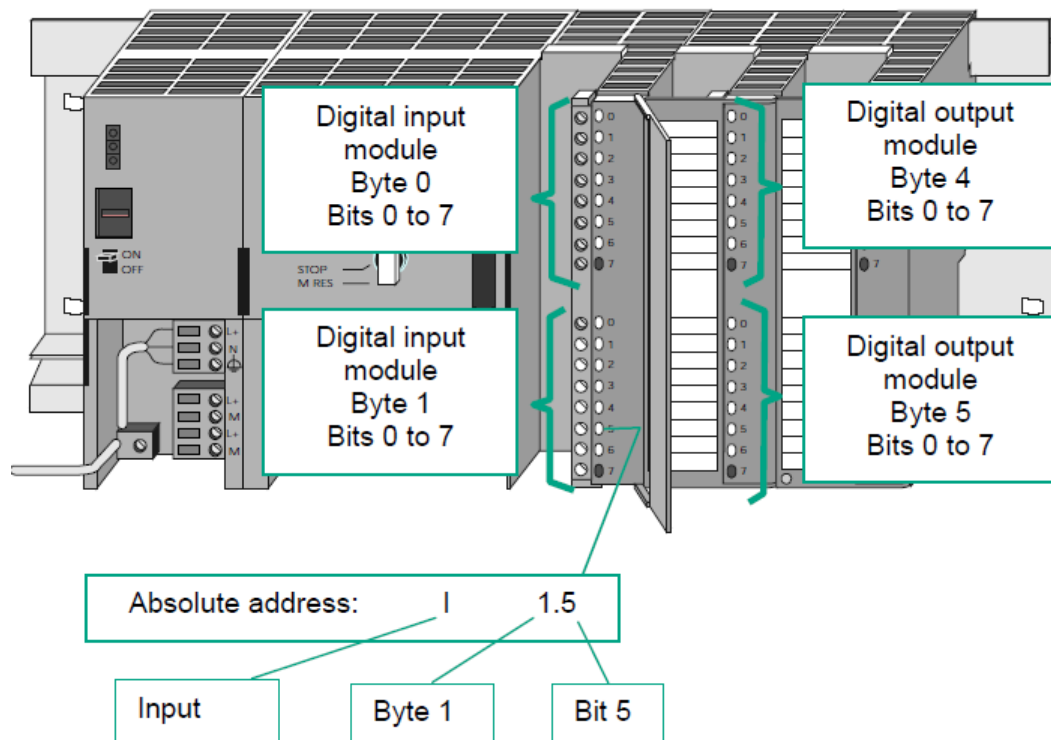
ساختار پروژه ایجاد شده



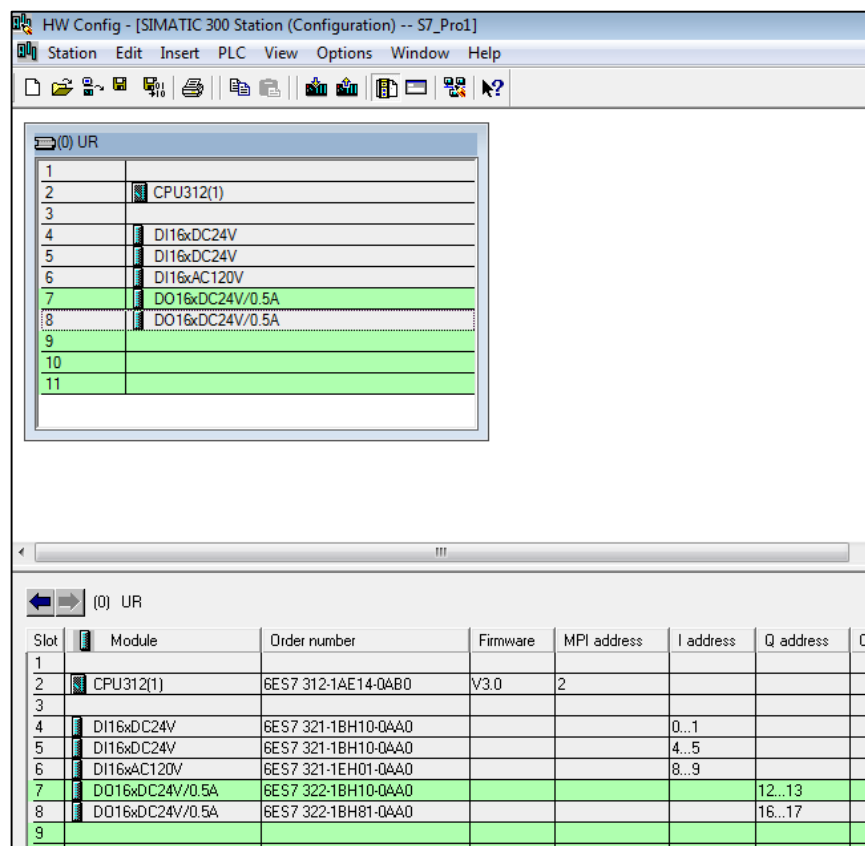
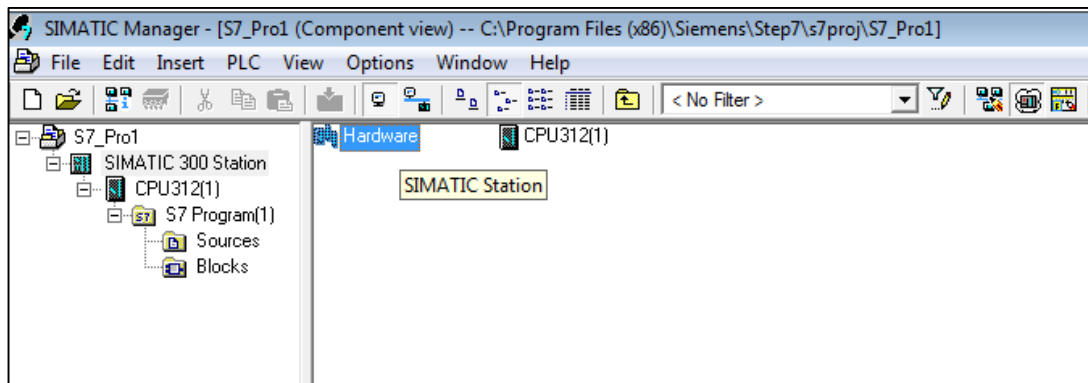
نمای کلی نرم افزار



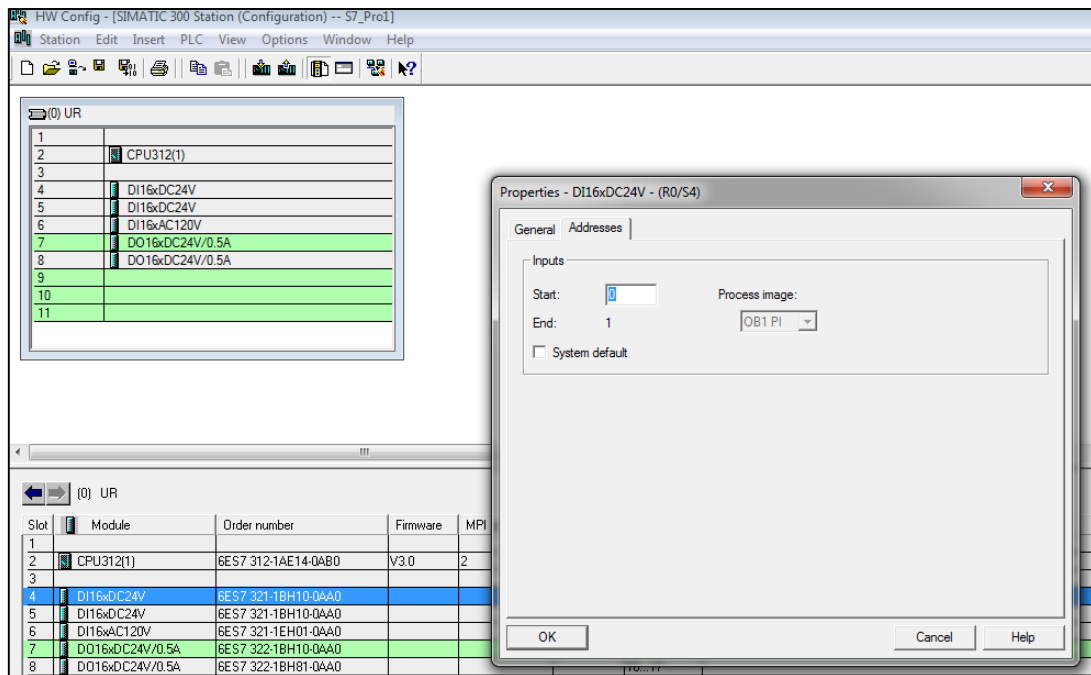
نحوه آدرس دهی در S7



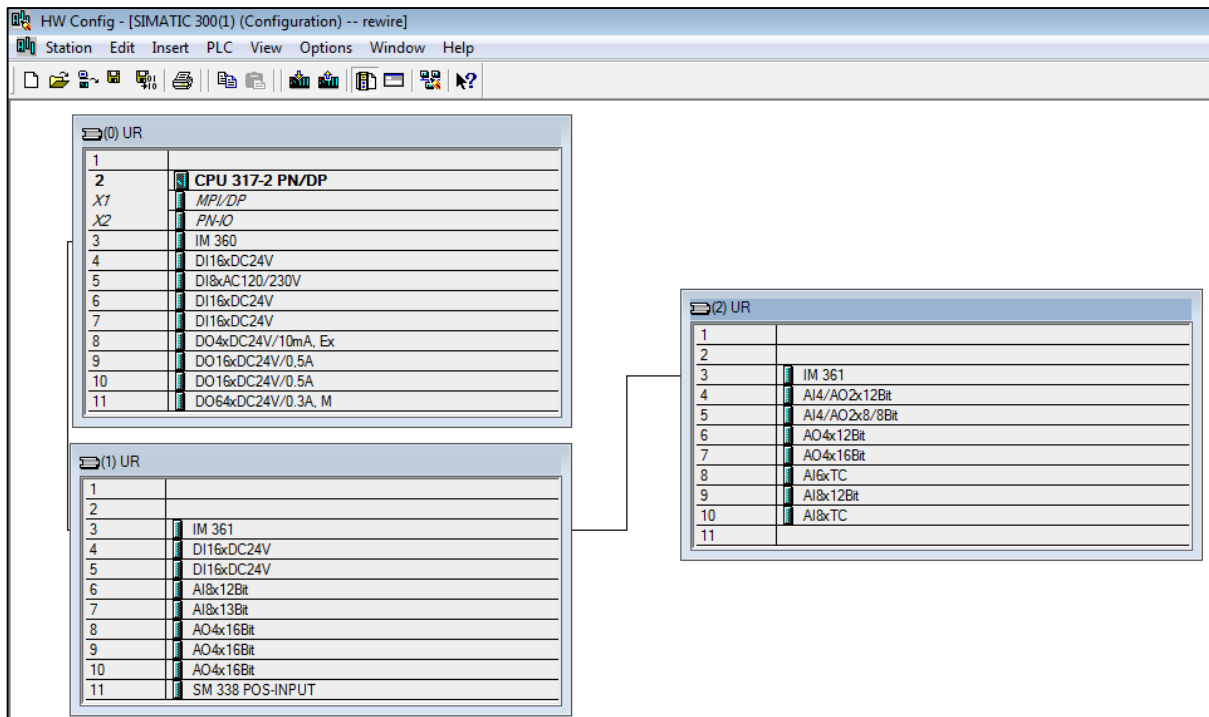
پیکربندی سخت افزار در محیط Hardware



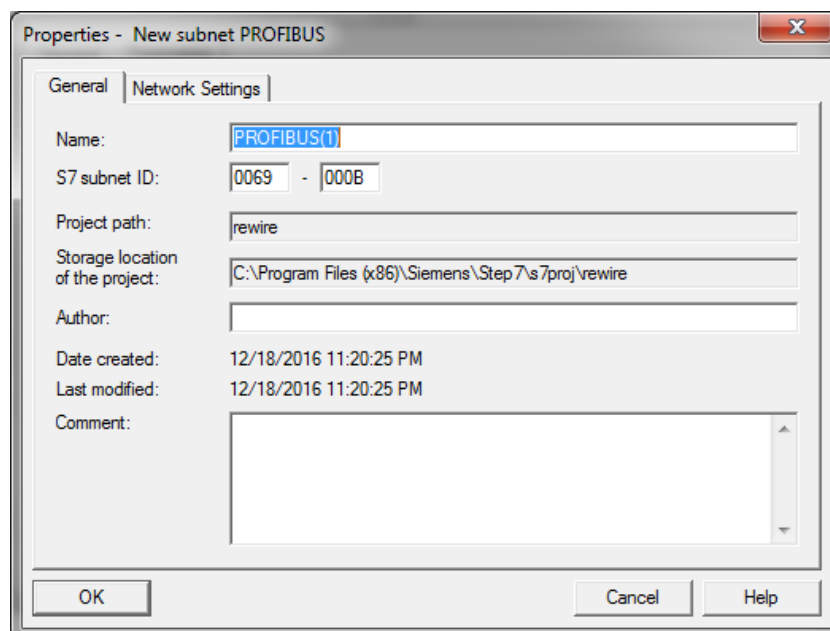
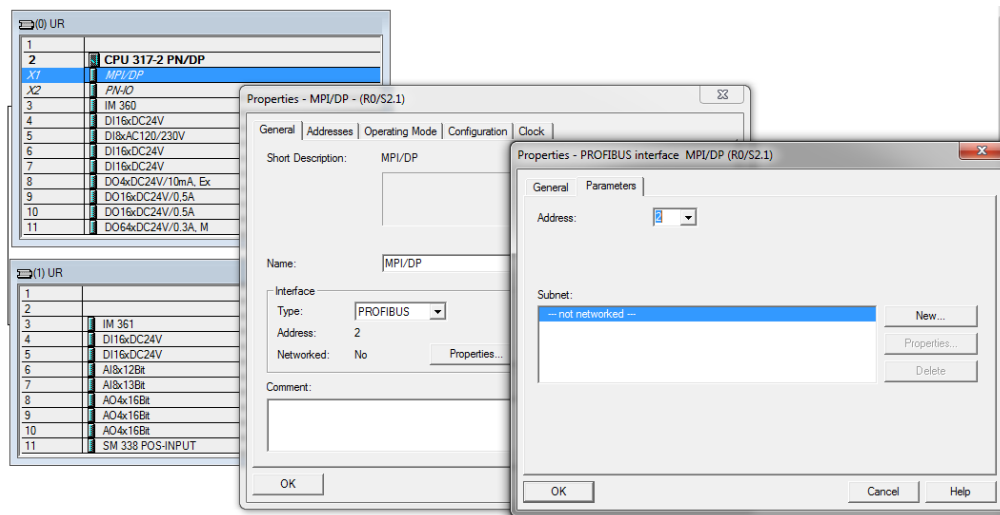
تغییر آدرس های ماژول های SM



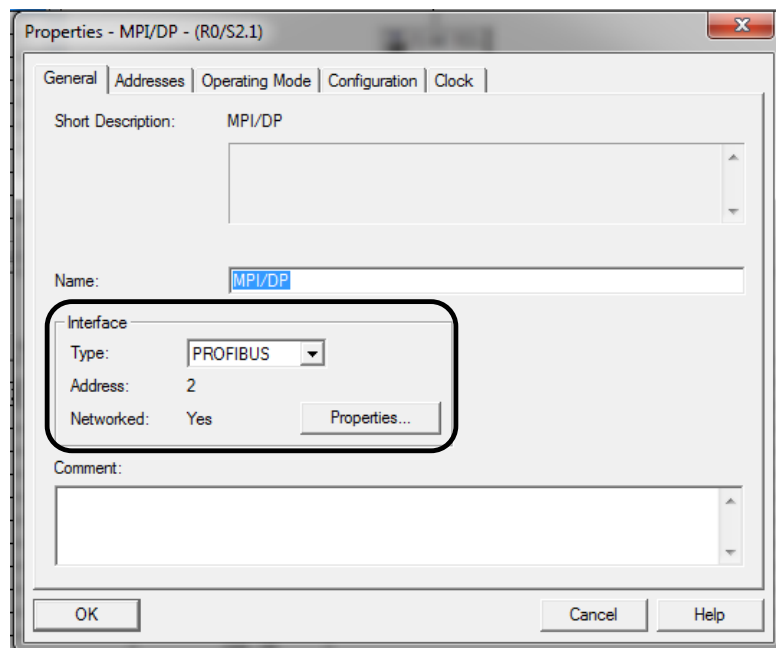
در PLC S7-300 توسط کارت IM361R و IM360S ظرفیت PLC به لحاظ تعداد کارت های SM , CP , FM تا سه رک اضافی در کنار رک اصلی می تواند توسعه پیدا کند . این تعداد ماکزیمم تعداد مجاز برای PLC S7-300 می باشد .



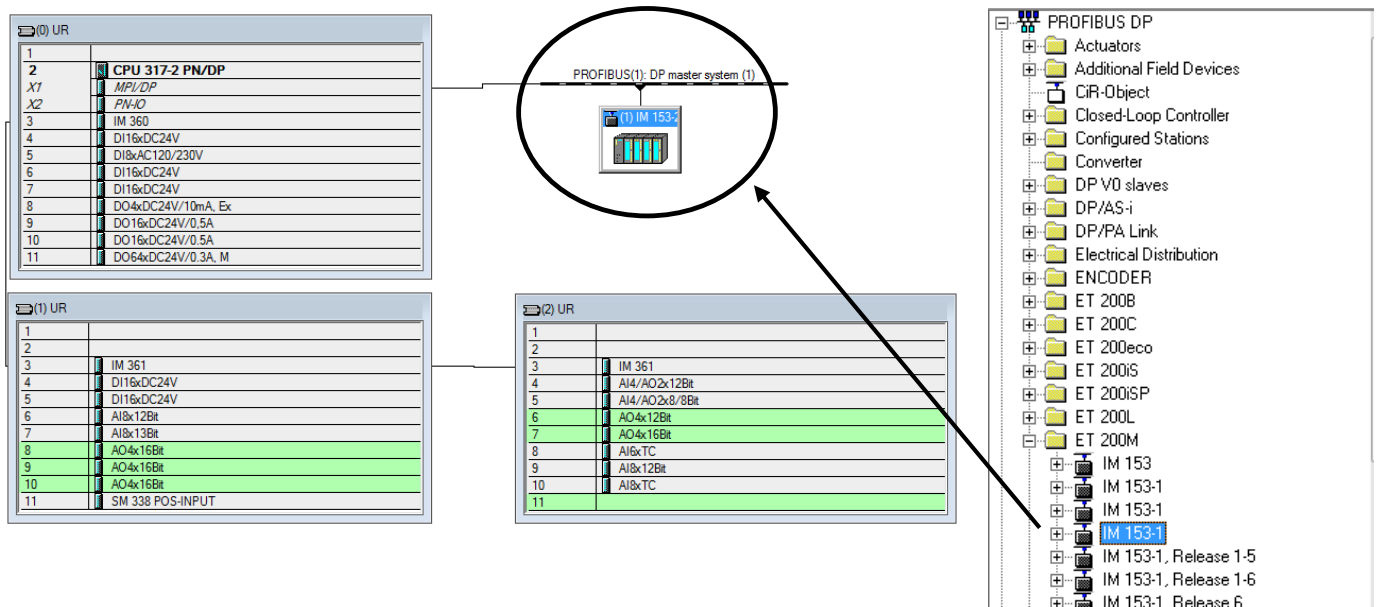
یکی از روش های افزایش تعداد بیشتر I/O و یا متمرکز کردن I/O ها در قسمت های مختلف یک سیستم، استفاده از ایستگاه های Remote I/O می باشد که در لیست تجهیزات زیرمبناس با نام ET معرفی می شوند . معمول ترین راه استفاده از ET ها قرار دادن این ایستگاهها در شبکه ی پروفیباس است و برقراری ارتباط با PLC توسط این شبکه است. جهت اتصال ایستگاه ET به شبکه پروفیباس ابتدا باید بستر این شبکه روی پورت شبکه PLC ایجاد شود.



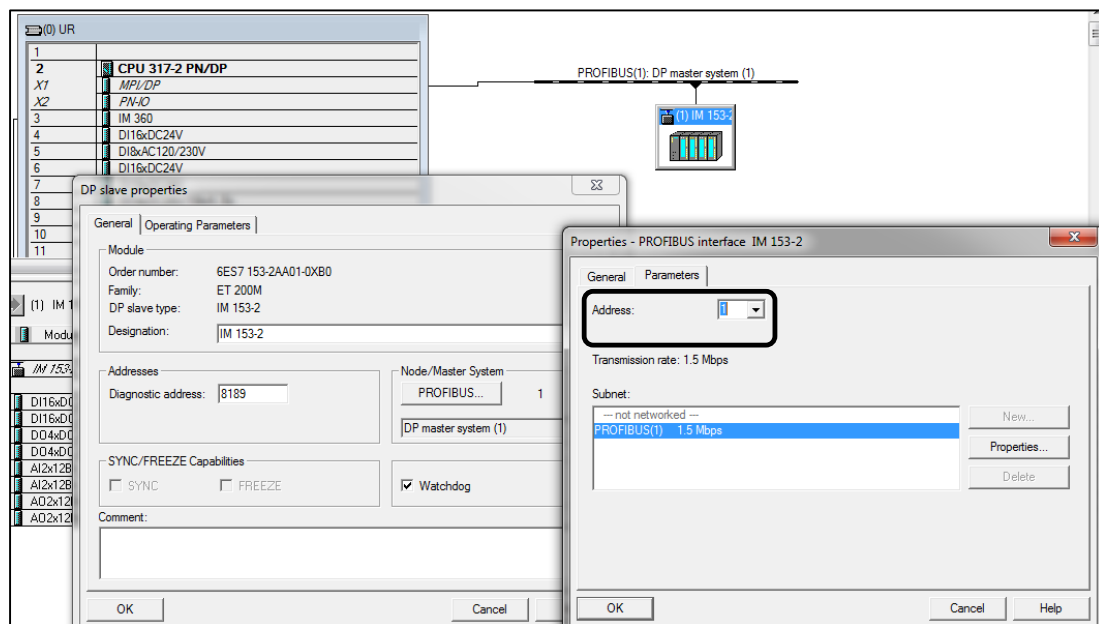
فعال شدن شبکه پروفیباس DP



در تصویر زیر نحوه ی اتصال یک ایستگاه ET200M در شبکه ی پروفیباس نشان داده شده است. جهت چیدمان کارت در ایستگاه ET از قسمت پایین نرم افزار استفاده می کنیم.

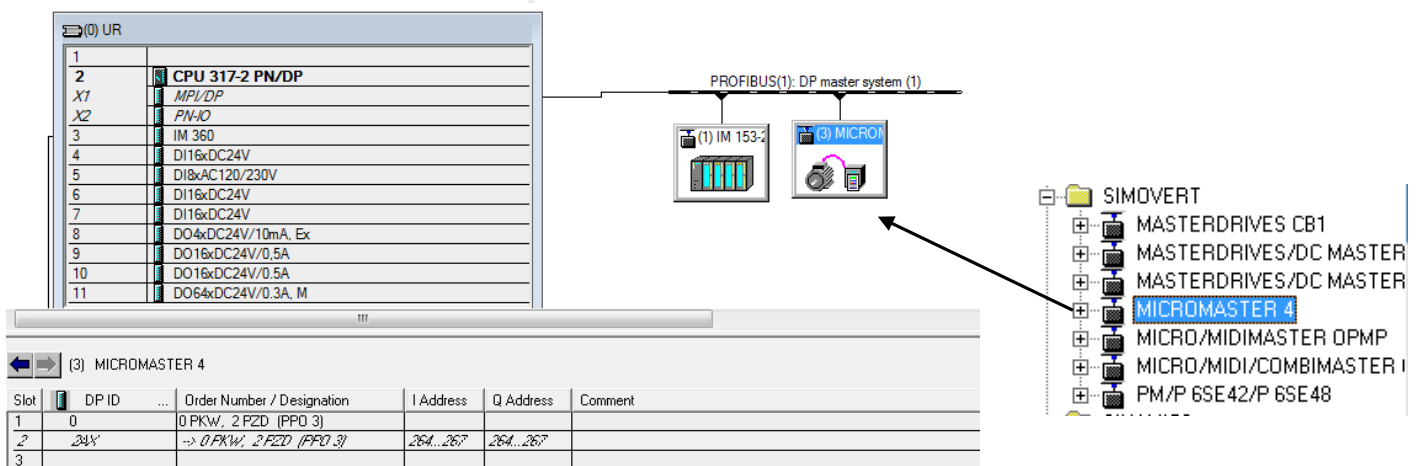


هر یک از ایستگاه های متصل شده به شبکه ی پروفیباس دارای آدرس منحصر به فرد هستند که آدرس Node شبکه نامیده می شود. جهت تعیین این آدرس داریم:



همچنین آدرس کارت های روی ET در ادامه ی کارت های ورودی و خروجی رک اصلی تنظیم می شود.

یکی دیگر از ایستگاه هایی که می تواند به شبکه پروفیباس متصل شود، درایو می باشد. به عنوان مثال جهت اتصال یک درایو میکرومستر زیمنس به شبکه ی پروفیباس از کتابخانه ی نرم افزار و در زیر مجموعه ی PROFIBUS DP گزینه MICROMASTER 4 را انتخاب می کنیم.



جهت تبادل دیتای درایو با PLC ، حتما نیاز به فضای PZD می باشد. از فضای PKW جهت دسترسی و تغییر پارامترهای درایو استفاده می شود.

تعاریف اولیه

نحوه پردازش CPU

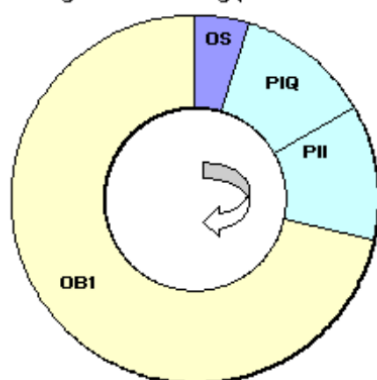
بعد از RUN کردن CPU در صورت وجود بلوک های Start Up ابتدا برنامه راه اندازی خوانده می شود. در ادامه مقادیر تصاویر PIQ به خروجی ها نوشته شده و وضعیت ورودی ها از کارت ها خوانده و در حافظه PII قرار می گیرد. پس از قرارگیری مقدار لحظه ای متغیرها در PII، برنامه نوشته شده در CPU اجرا می شود. در ضمن اجرای برنامه، هر جا که لازم باشد اطلاعات را از جدول PII می خواند. مقدار فضای PII و PIQ برای هر CPU مقداری مشخص می باشد. به عنوان مثال CPU 313C دارای 128/128 بایت فضای Process Image می باشد. در ضمن اجرای برنامه اگر نیاز به ایجاد فرمانی جهت خروجی ها باشد، این فرمان در ناحیه PIQ قرار داده می شود. این فضا نیز بخشی از حافظه CPU می باشد که فرمان های ارسالی در آن ذخیره می شود. پس از اینکه اجرای برنامه تمام شد، CPU در ابتدای سیکل مقادیر موجود در PIQ را به کارت خروجی ارسال می کند.

PII مخفف Process Image Input می باشد و معرف بخشی از حافظه CPU است که ورودی ها در آن ذخیره می شوند.

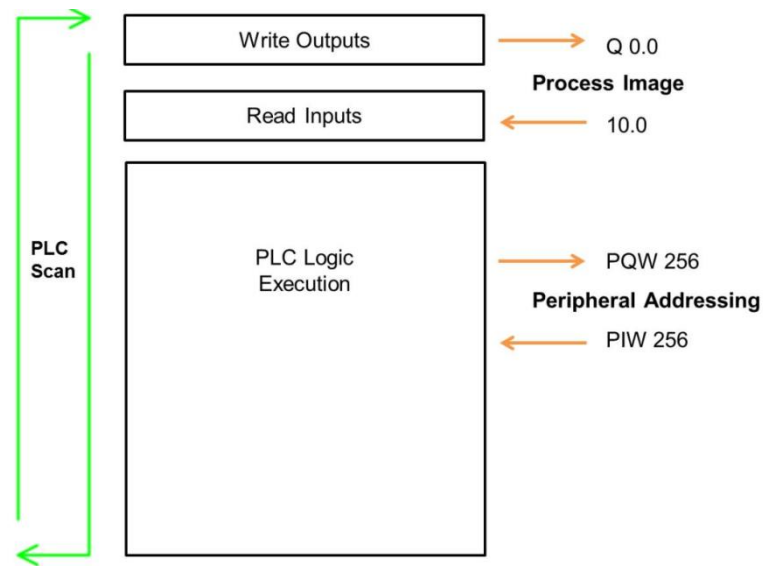
به طور مشابه برخی از خروجی ها بدون اینکه از PIQ فرمان بگیرند به طور مستقیم در صورت لزوم از CPU مقدار می گیرند. به این خروجی ها اصطلاحاً Peripheral output می گویند.

نحوه اسکن CPU بعد از سال ۹۸ به صورت زیر تغییر یافت تا بعد از اینکه CPU از مد Startup عبور کرد، ابتدا مقادیر فضای PIQ را به کارت های خروجی ارسال کند.

Cyclic Program Processing (CPUs as of 10/98)

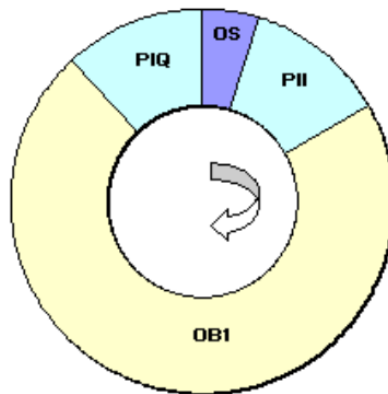


ادامه



قبل از سال ۹۸ پروسه پردازش برنامه به صورت زیر انجام طراحی شده بود.

Cyclic Program Processing (CPUs up to 10/98)

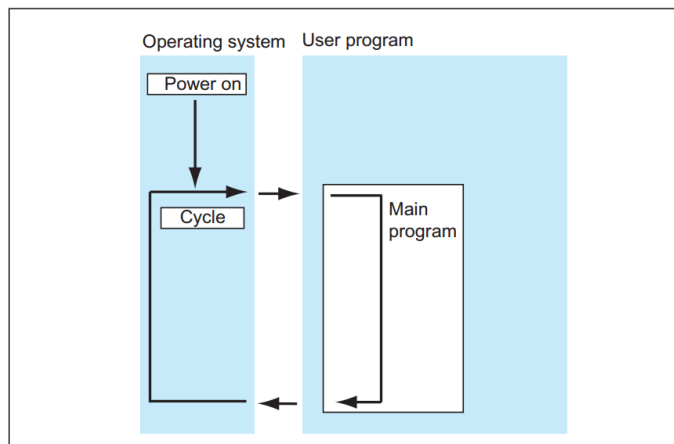


زمان سیکل اسکن

به مدت زمانی که طول می کشد تا یک سیکل کاری توسط CPU اجرا شود، زمان سیکل اسکن گفته می شود. این زمان از جمع زمان های زیر به دست می آید:

۱- زمان مربوط به خواندن ورودی ها و قرار دادن آن در PII

۲- زمان ارسال خروجی ها از PIQ به کارت خروجی

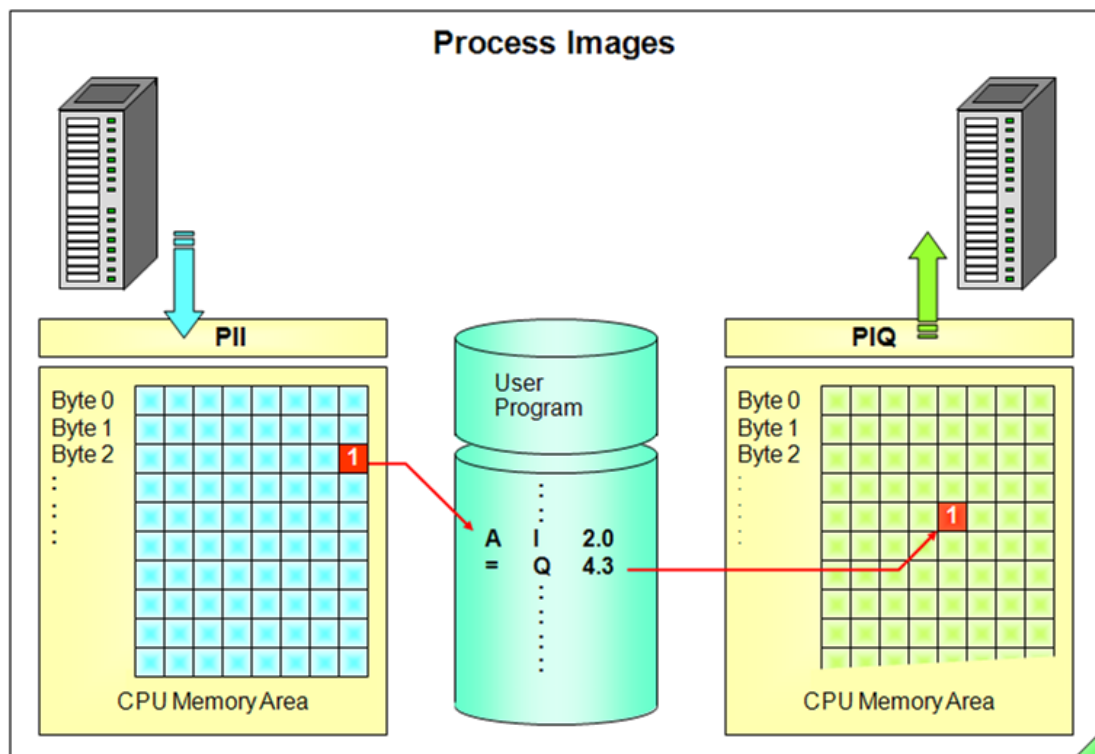


۳- زمان مربوط به اجرای برنامه

۴- زمان مربوط به سیستم عامل

۵- زمان مربوط به تبادل دیتا با شبکه

۶- وقفه های پیش آمده در زمان کار CPU



زمان مجاز سیکل اسکن

از آنجا که در طول زمان اجرای برنامه اصلی توسط CPU، مقادیر موجود در PII به روز رسانی نمی شوند، باید زمان سیکل اسکن زمان کمی باشد تا امکان به روز رسانی سریع PII و PIQ وجود داشته باشد. این زمان با توجه به پروسه های مختلف متفاوت می باشد. در هر صورت کاربر می تواند زمانی را به عنوان حداکثر زمان مجاز سیکل اسکن تعریف کند که در صورت افزایش زمان سیکل اسکن CPU از این زمان، یک وقفه به CPU اعمال شود.

این زمان به صورت پیش فرض ۱۵۰ میلی ثانیه است که در محیط HW قابل تغییر می باشد.

در S7-400 امکان اختصاص زمانی وجود دارد که به عنوان حداقل زمان سیکل اسکن می باشد. در صورتی که زمان سیکل اسکن از مقدار تعریف شده کمتر باشد، OS سیکل اسکن بعدی را پس از سپری شدن زمان مورد نظر آغاز می کند. مثلاً اگر در یک PLC حداقل زمان سیکل اسکن ۳۰ میلی ثانیه تعریف شده باشد و سیکل اسکن جاری ۲۰ میلی ثانیه طول بکشد، OS به اندازه ۱۰ ثانیه صبر می کند و سپس سیکل اسکن بعدی را اجرا می کند.

بخش های حافظه

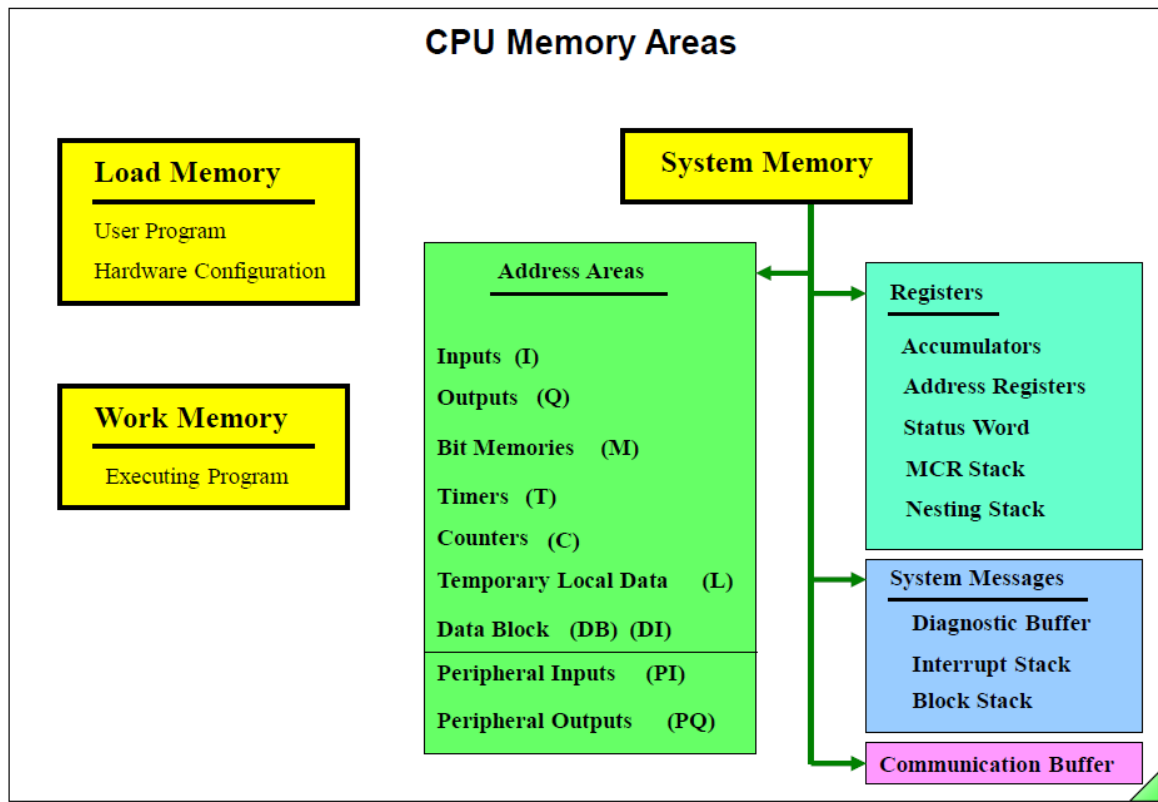
Load Memory

شامل برنامه کاربر، تنظیمات پیکربندی سخت افزار و دستورات برنامه نویسی و سایر جزئیات می باشد.

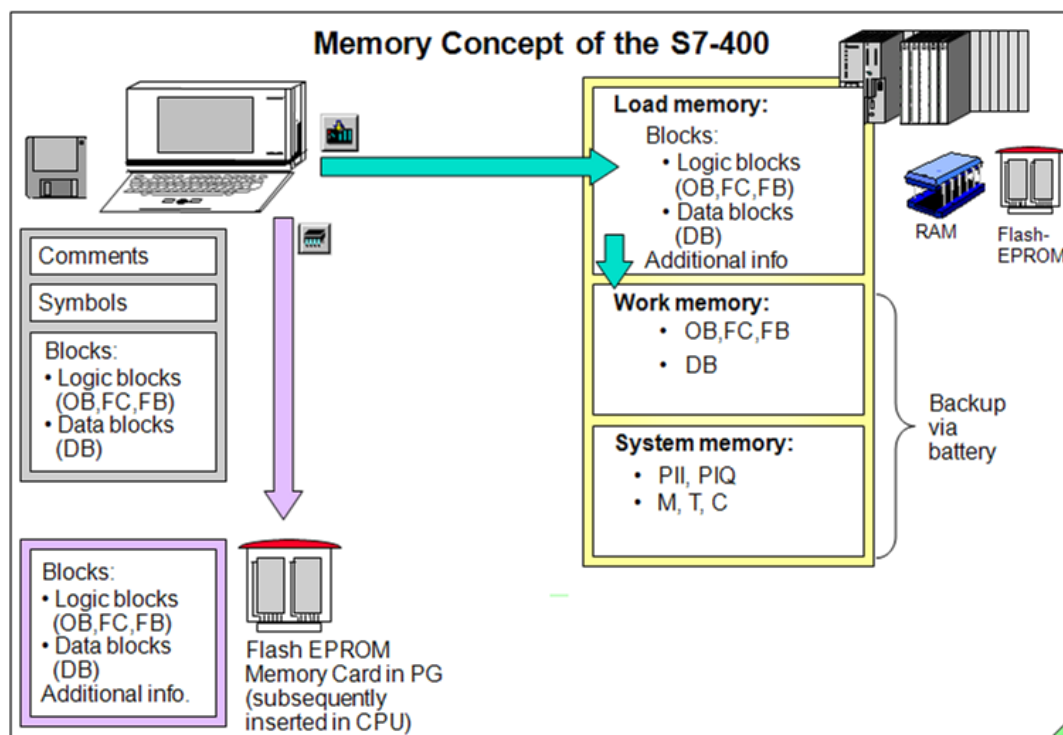
Work Memory

حافظه کاری CPU بوده که برنامه اجرایی به آنجا منتقل می شود. از بین آنچه از برنامه ی کاربر که به Load Memory منتقل شده است، تنها بخش های اجرایی به Work Memory منتقل می گردند.

فضاهای حافظه CPU



مفهوم حافظه در S7-400



بلوک های برنامه

Block Types

There are several different types of blocks you can use within an S7 user program:

Block	Brief Description of Function	See Also
Organization blocks (OB)	OBs determine the structure of the user program.	Organization Blocks and Program Structure
System function blocks (SFB) and system functions (SFC)	SFBs and SFCs are integrated in the S7 CPU and allow you access to some important system functions.	System Function Blocks (SFB) and System Functions (SFC)
Function blocks (FB)	FBs are blocks with a "memory" which you can program yourself.	Function Blocks (FB)
Functions (FC)	FCs contain program routines for frequently used functions.	Functions (FC)
Instance data blocks (instance DB)	Instance DBs are associated with the block when an FB/SFB is called. They are created automatically during compilation.	Instance Data Blocks
Data blocks (DB)	DBs are data areas for storing user data. In addition to the data that are assigned to a function block, shared data can also be defined and used by any blocks.	Shared Data Blocks (DB)

بلوک های Interrupts

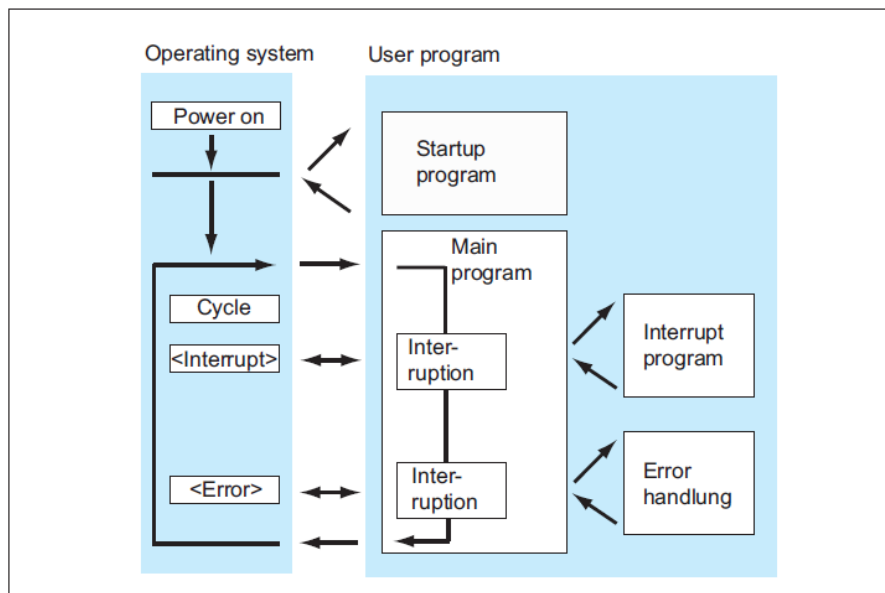
Type of Interrupt	Organization Block	Priority Class (Default)	See also
Main program scan	OB1	1	Organization Block for Cyclic Program Processing (OB1)
Time-of-day interrupts	OB10 to OB17	2	Time-of-Day Interrupt Organization Blocks (OB10 to OB17)
Time-delay interrupts	OB20 OB21 OB22 OB23	3 4 5 6	Time-Delay Interrupt Organization Blocks (OB20 to OB23)
Cyclic interrupts	OB30 OB31 OB32 OB33 OB34 OB35 OB36 OB37 OB38	7 8 9 10 11 12 13 14 15	Cyclic Interrupt Organization Blocks (OB30 to OB38)

Type of Interrupt	Organization Block	Priority Class (Default)	See also
Hardware interrupts	OB40 OB41 OB42 OB43 OB44 OB45 OB46 OB47	16 17 18 19 20 21 22 23	Hardware Interrupt Organization Blocks (OB40 to OB47)
DPV1 interrupts	OB 55 OB 56 OB 57	2 2 2	Programming DPV1 Devices
Multicomputing interrupt	OB60 Multicomputing	25	Multicomputing - Synchronous Operation of Several CPUs
Synchronous cycle interrupt	OB 61 OB 62 OB 63 OB 64	25	Configuring Short and Equal-Length Process Reaction Times on PROFIBUS-DP
Redundancy errors	OB70 I/O Redundancy Error (only in H systems) OB72 CPU Redundancy Error (only in H systems)	25 28	"Error Handling Organization Blocks (OB70 to OB87 / OB121 to OB122)"
Asynchronous errors	OB80 Time Error OB81 Power Supply Error OB82 Diagnostic Interrupt OB83 Insert/Remove Module Interrupt OB84 CPU Hardware Fault OB 85 Program Cycle Error OB86 Rack Failure OB87 Communication Error	25 (or 28 if the asynchronous error OB exists in the startup program)	Error Handling Organization Blocks (OB70 to OB87 / OB121 to OB122)
Background cycle	OB90	29 ¹⁾	Background Organization Block (OB90)
Startup	OB100 Restart (Warm start) OB101 Hot Restart OB102 Cold Restart	27 27 27	Startup Organization Blocks (OB100/OB101/OB102)
Synchronous errors	OB121 Programming Error OB122 Access Error	Priority of the OB that caused the error	Error Handling Organization Blocks (OB70 to OB87 / OB121 to OB122)
¹⁾ The priority class 29 corresponds to priority 0.29. The background cycle has a lower priority than the free cycle.			

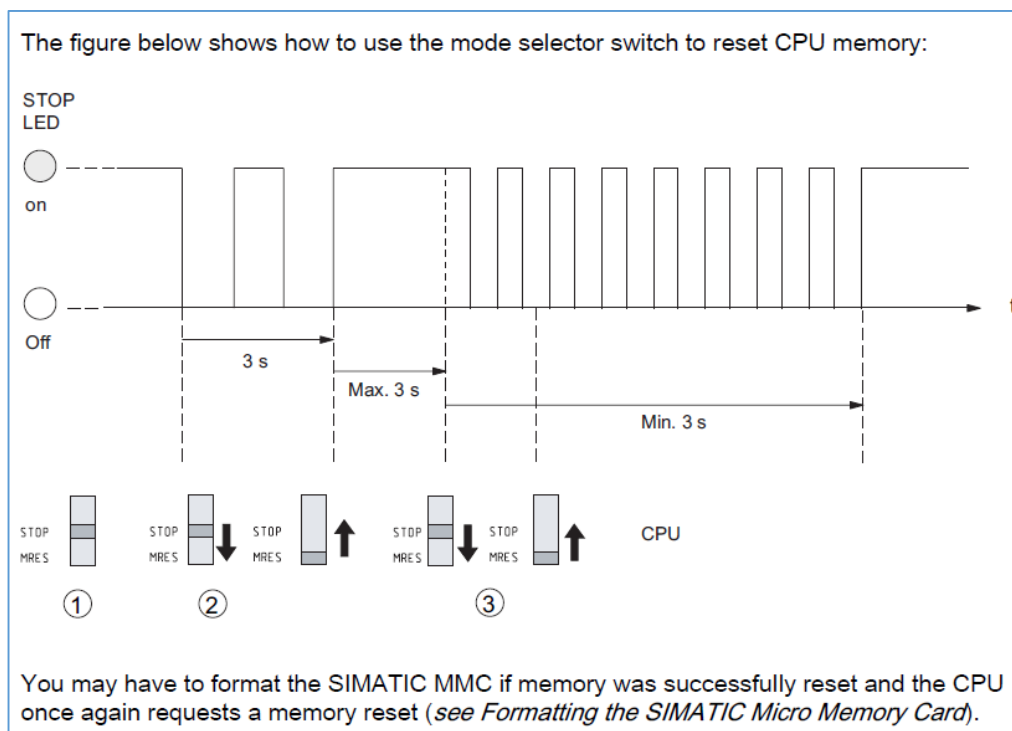
ایجاد وقفه در روال برنامه

Event-Driven Program Processing

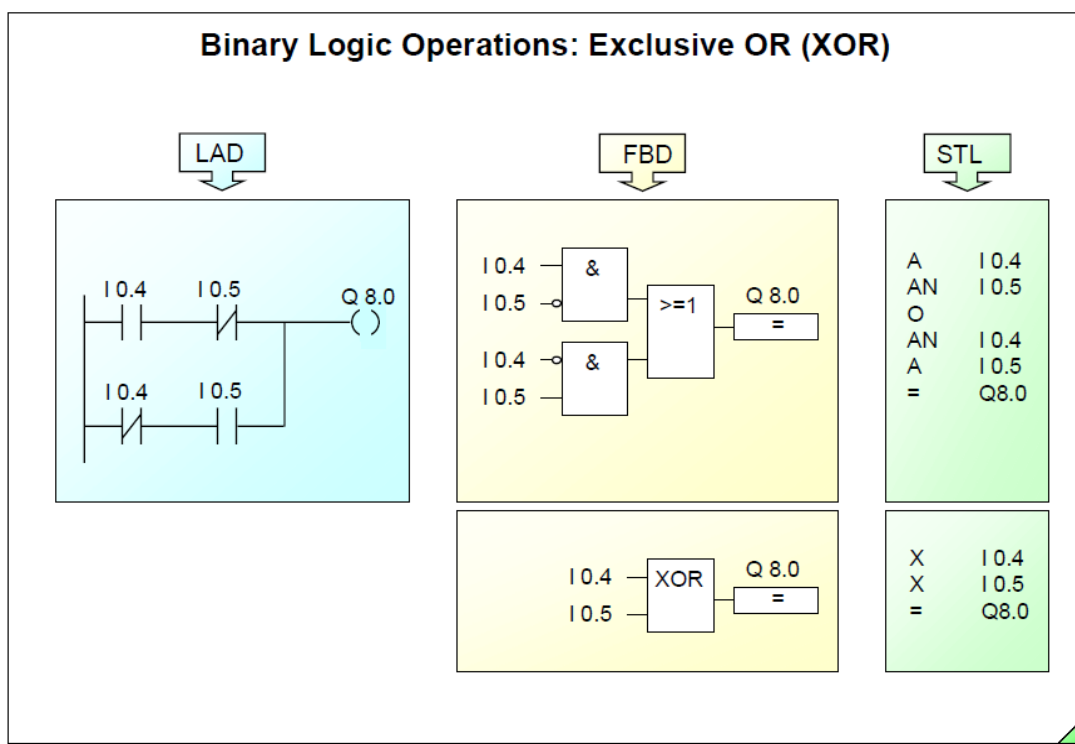
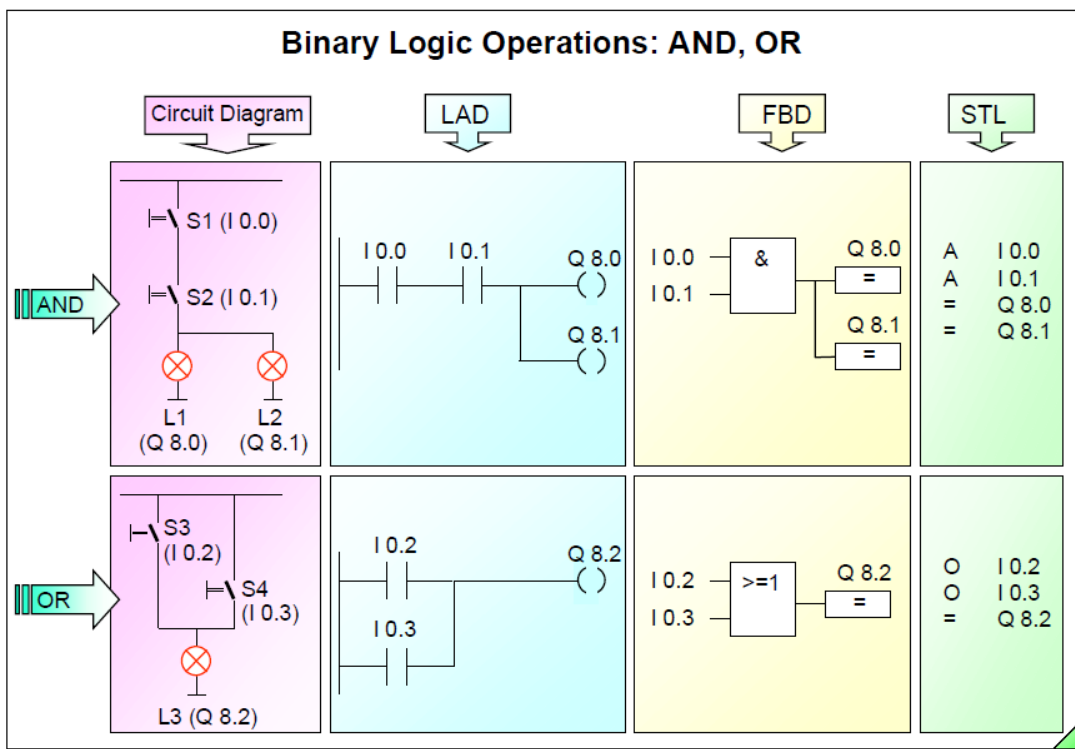
Cyclic program processing can be interrupted by certain events (interrupts). If such an event occurs, the block currently being executed is interrupted at a command boundary and a different organization block that is assigned to the particular event is called. Once the organization block has been executed, the cyclic program is resumed at the point at which it was interrupted.



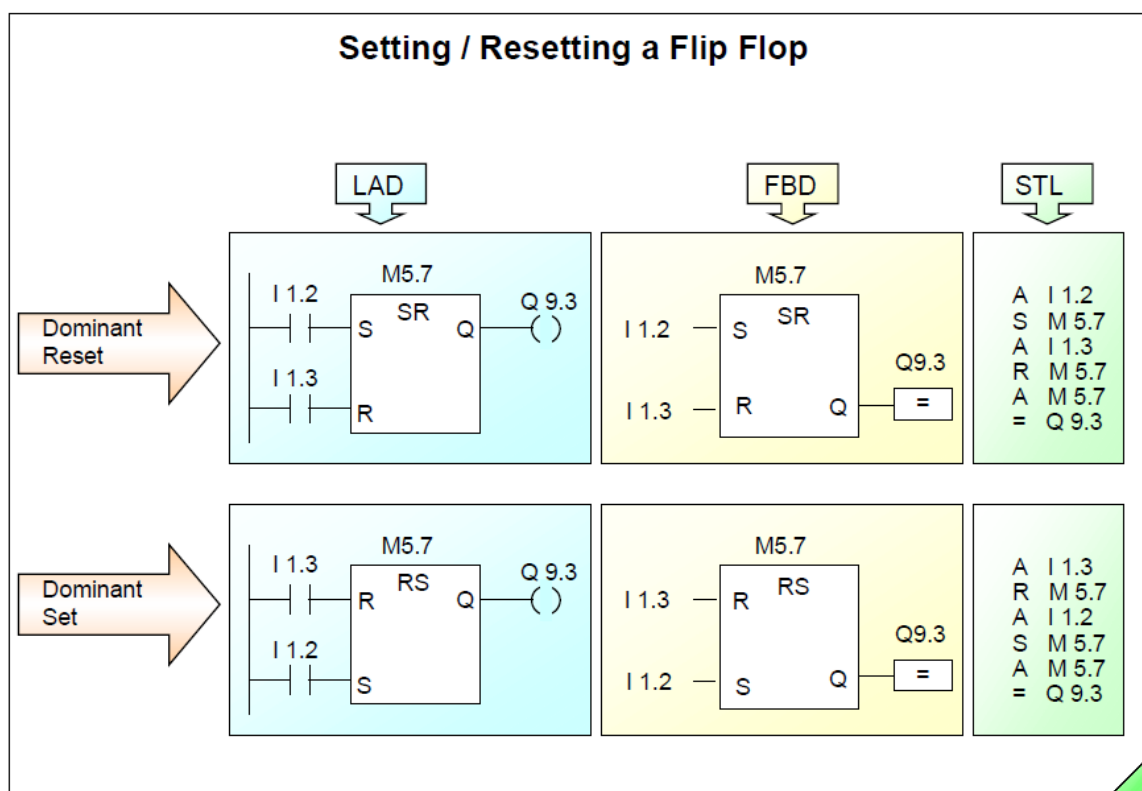
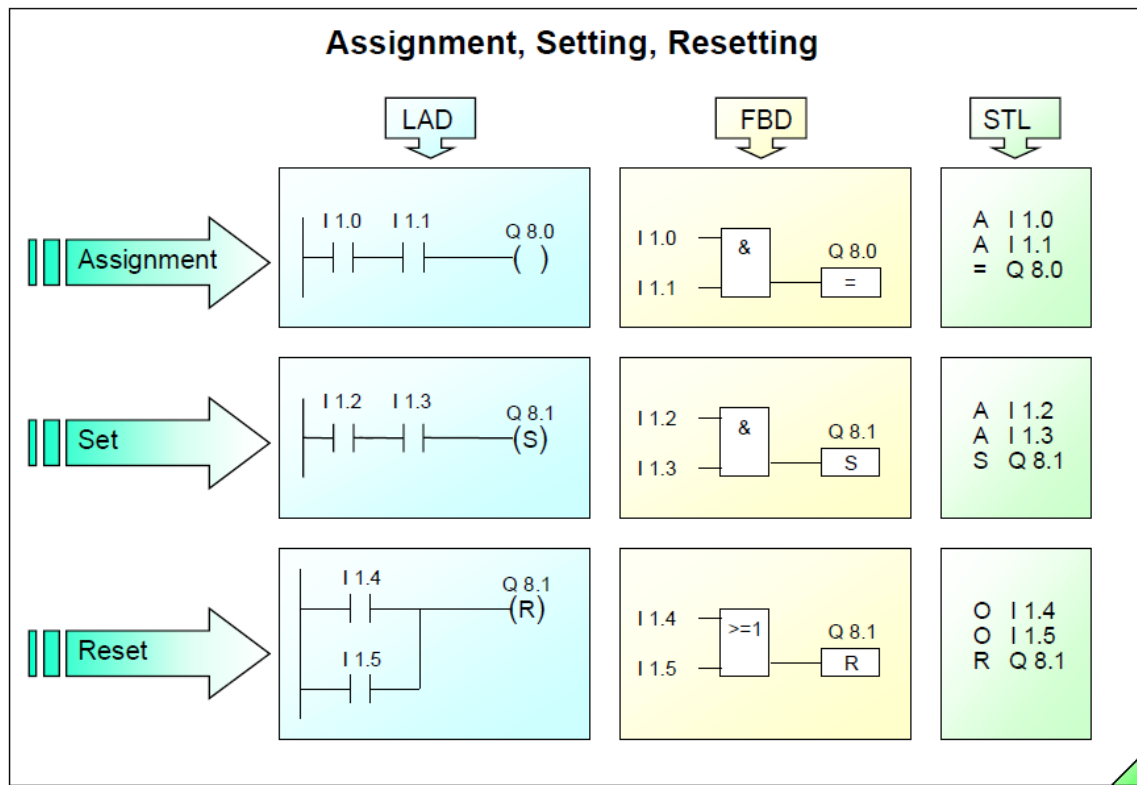
نحوه ریست کردن CPU



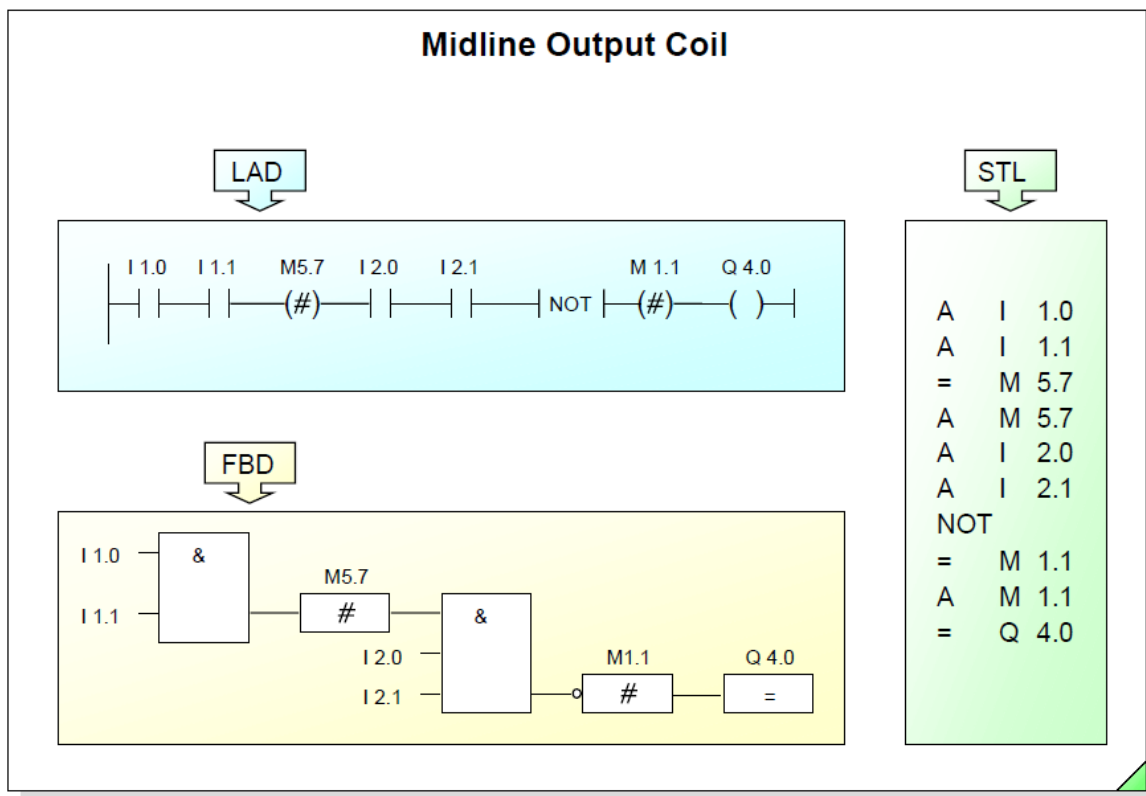
آشنایی با دستورات پرکاربرد در STEP7



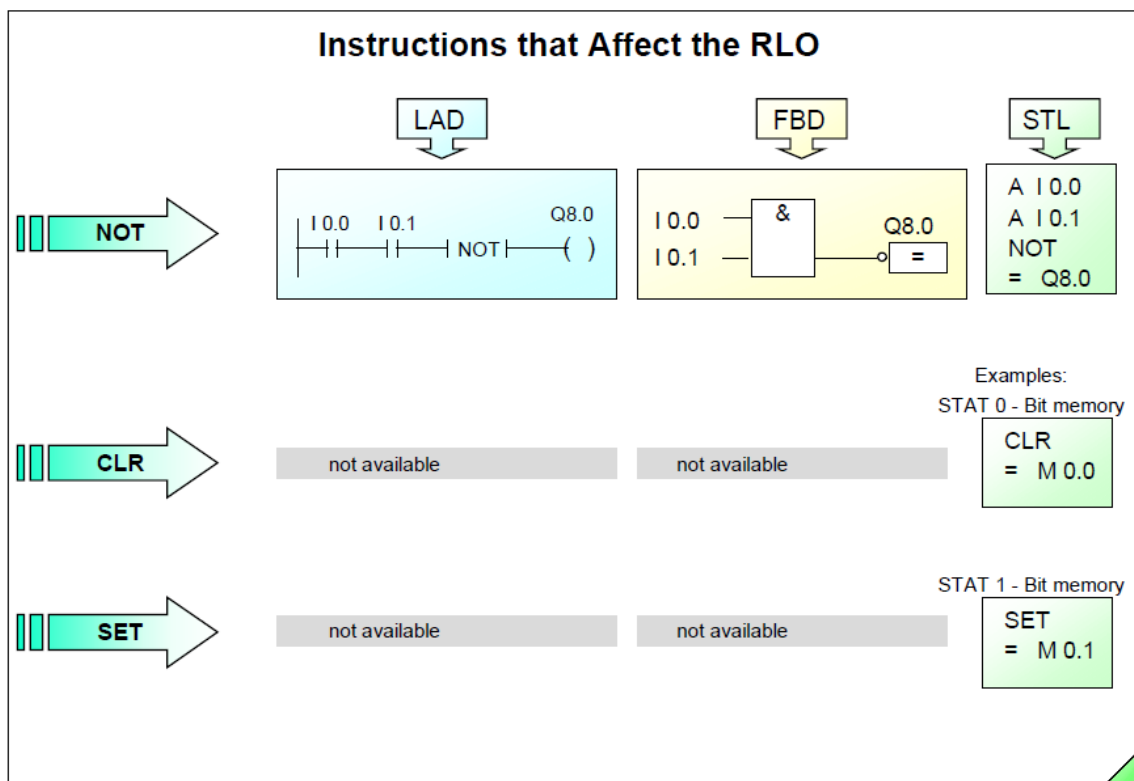
دستورات SET_RESET و فلیپ فلاپ



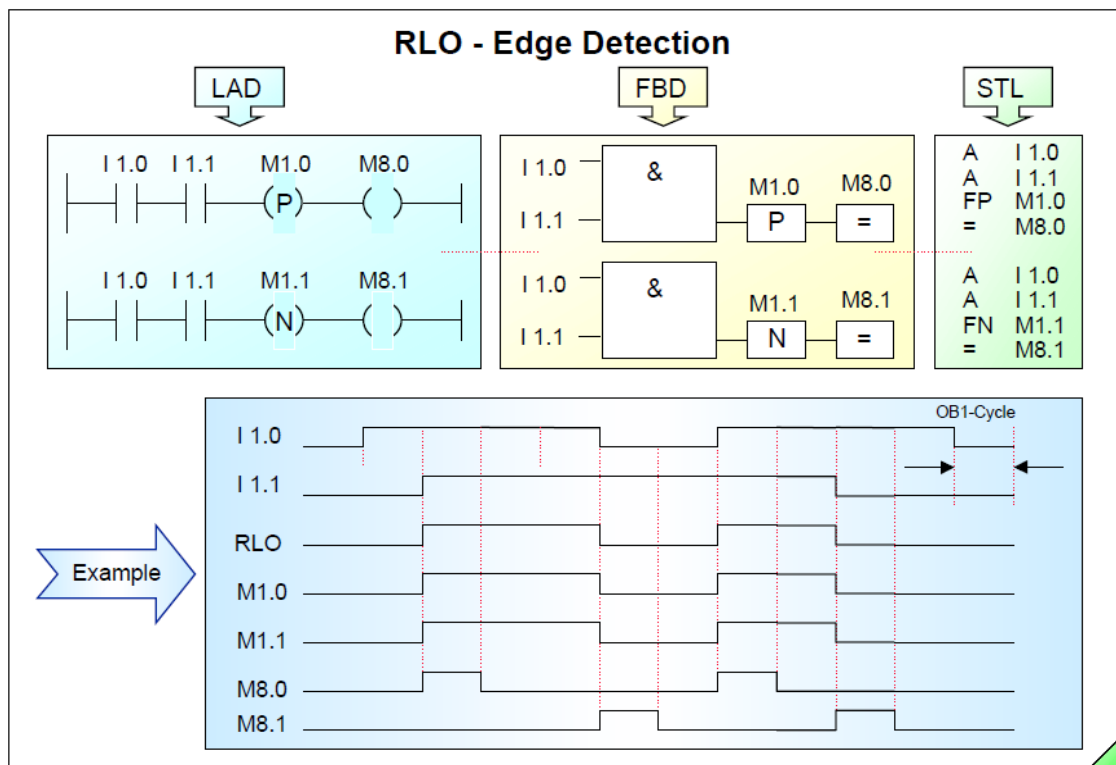
دستور کویل میانی



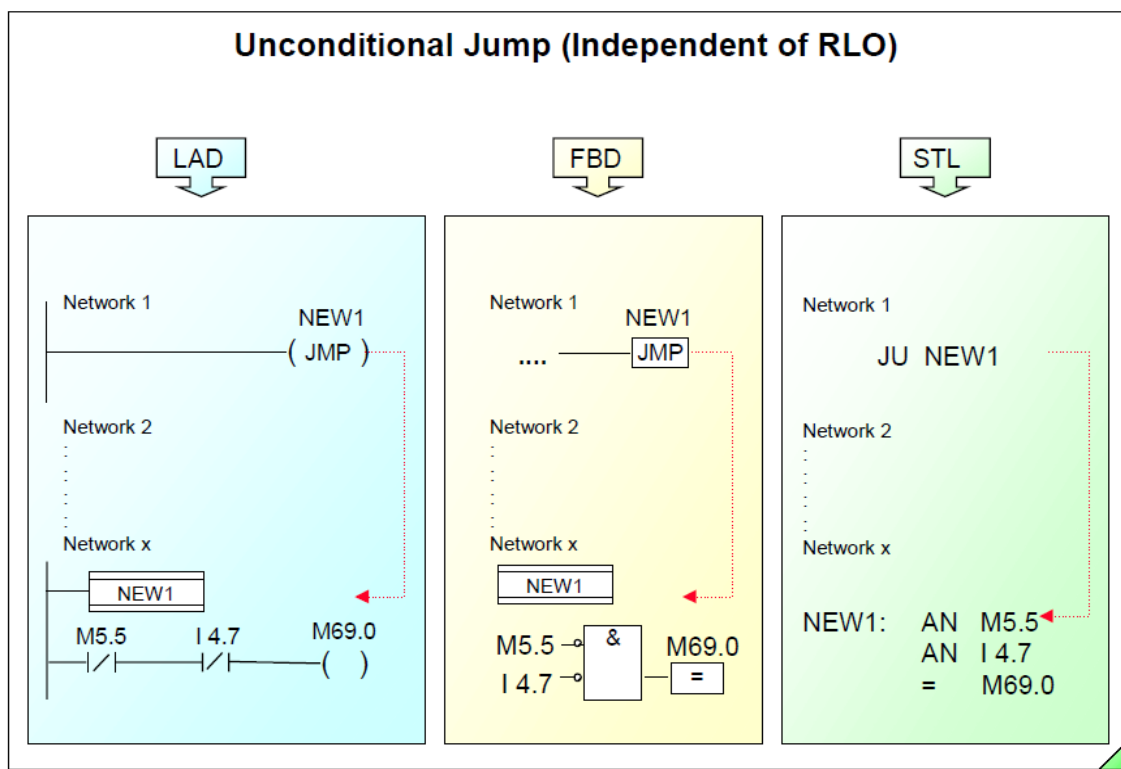
دستورات مرتبط با بیت RLO



دستورات تشخیص لبه



دستورات پرش



آشنایی با تایپ دیتاها

Integer (INT, 16-Bit Integer) Data Type

Value Range -32768 to +32767
(without sign: 0 to 65535)

Arithmetic Operations: such as +, *, /, <, ==

Display Formats:

DEC: +662 **BIN: 2# 0 0 0 0 0 1 0 1 0 0 1 1 0** **HEX: W#160 2 9 6**

Sign positive numbers

without sign

6 x 16⁰ = 6
9 x 16¹ = 144
2 x 16² = 512
662

DEC: -662 **BIN: 2# 1 1 1 1 1 0 1 0 1 0 1 0** **HEX: W#16 F D 6 A**

Sign negative numbers

without sign

10 x 16⁰ = 16
6 x 16¹ = 96
13 x 16² = 3328
15 x 16³ = 61440
64874

Representation as two's complement

-662

Double Integer (DINT, 32-Bit Integer) Data Type

Value Range L# -2147483648 to L#+2147483647
(without sign: 0 to 4294967295)

Arithmetic Operations: such as +, *, /, <, ==

Display Formats:

DEC: L# +540809 **BIN: 2# 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 1 0 0 0 1 0 0 1** **HEX: DW#16# 0 0 0 8 4 0 8 9**

Sign positive numbers

(without sign)

DEC: L# -540809 **BIN: 2# 1 1 1 1 1 1 1 1 1 0 1 1 1 0 1 1 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1** **HEX: DW#16# F F F 7 B F 7 7**

Sign negative numbers

(without sign)

Representation as two's complement

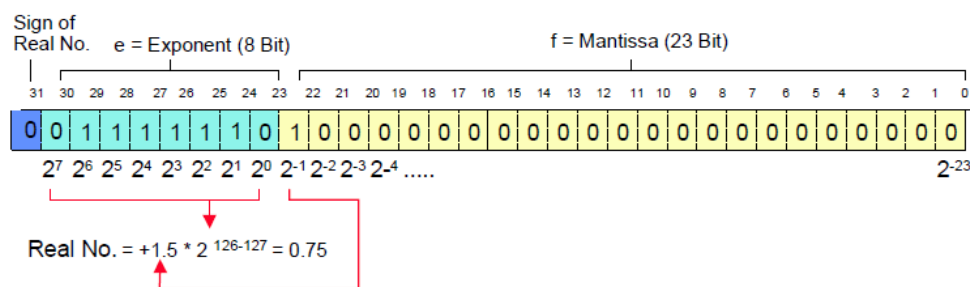
REAL (Floating-point Number, 32 Bit) Data Type

Value Range $-1.175495 \cdot 10^{-38}$ to $3.402823 \cdot 10^{+38}$

Arithmetic Operations: such as + R, * R, <R, ==R
sin, acos, ln, exp, SQR

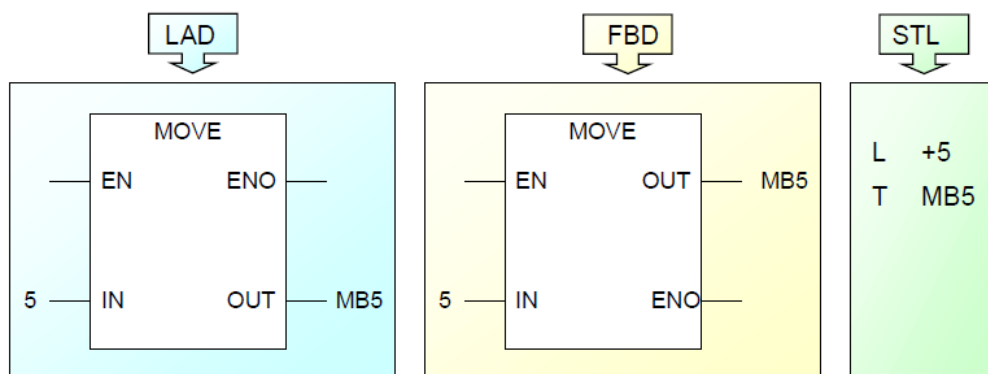
General Format of a Real Number = (Sign) • (1.f) • (2^{e-127})

Example: 7.50000e-001 ($7.5 \cdot 10^{-1} = 0.75$)



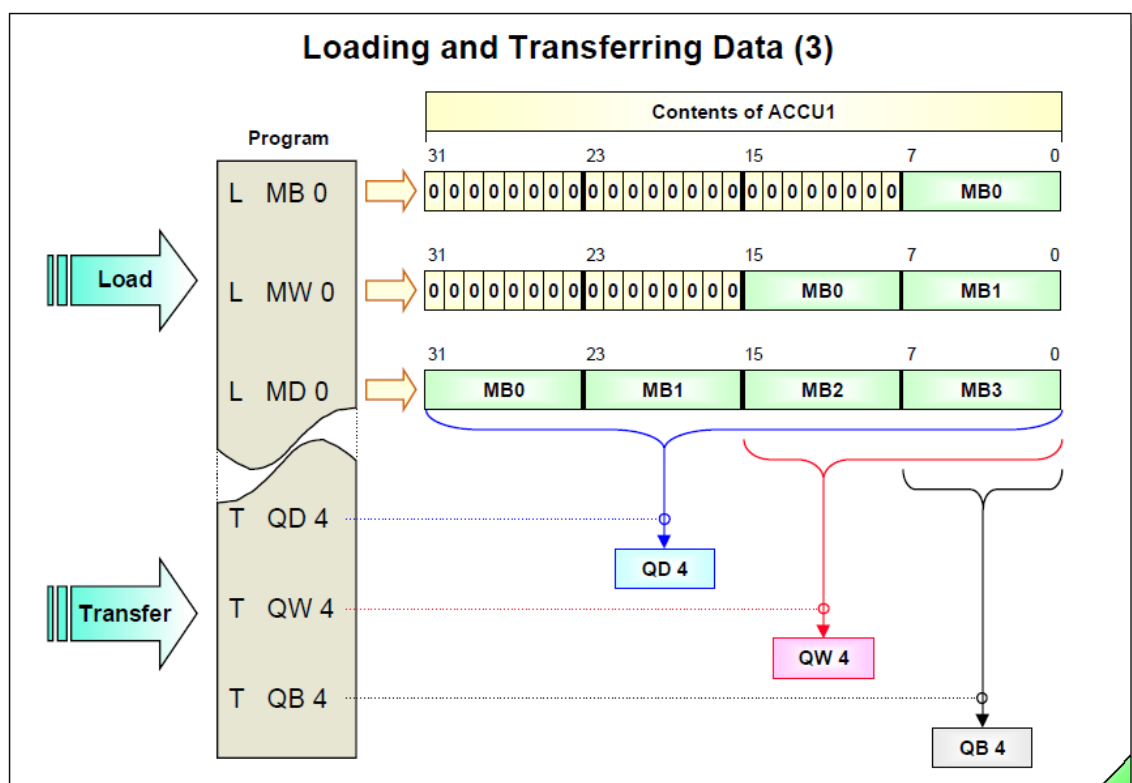
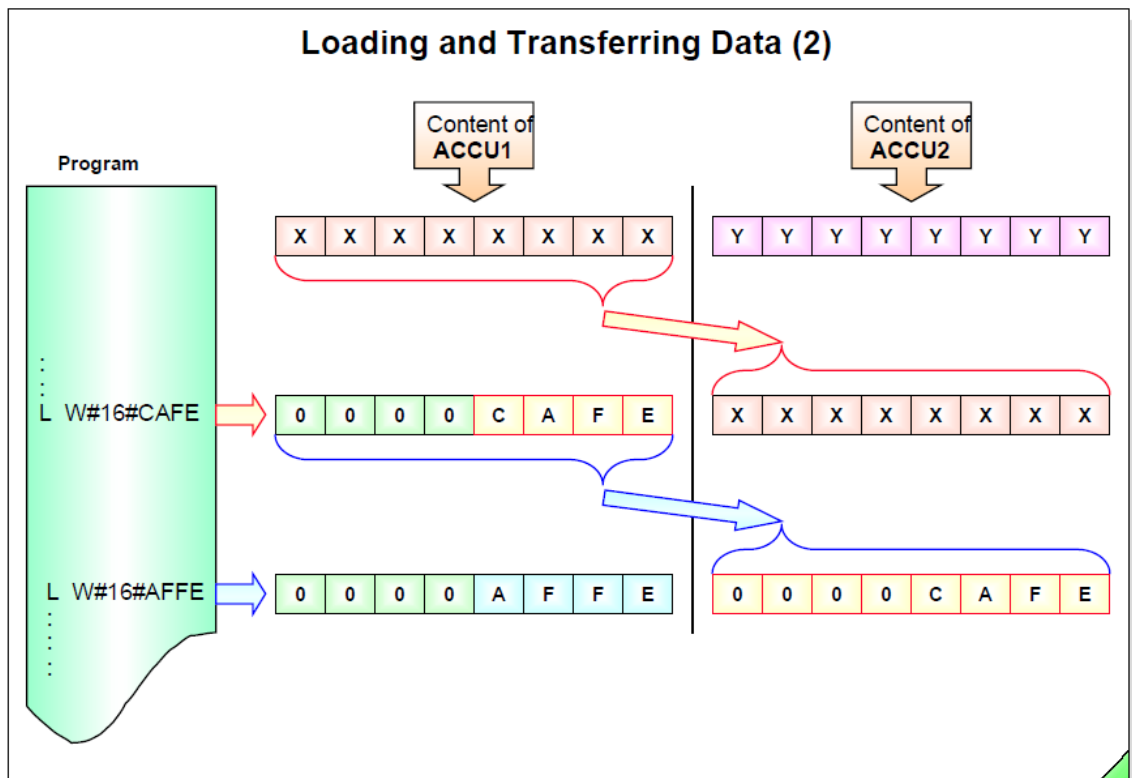
دستور انتقال MOVE

Loading and Transferring Data (1)

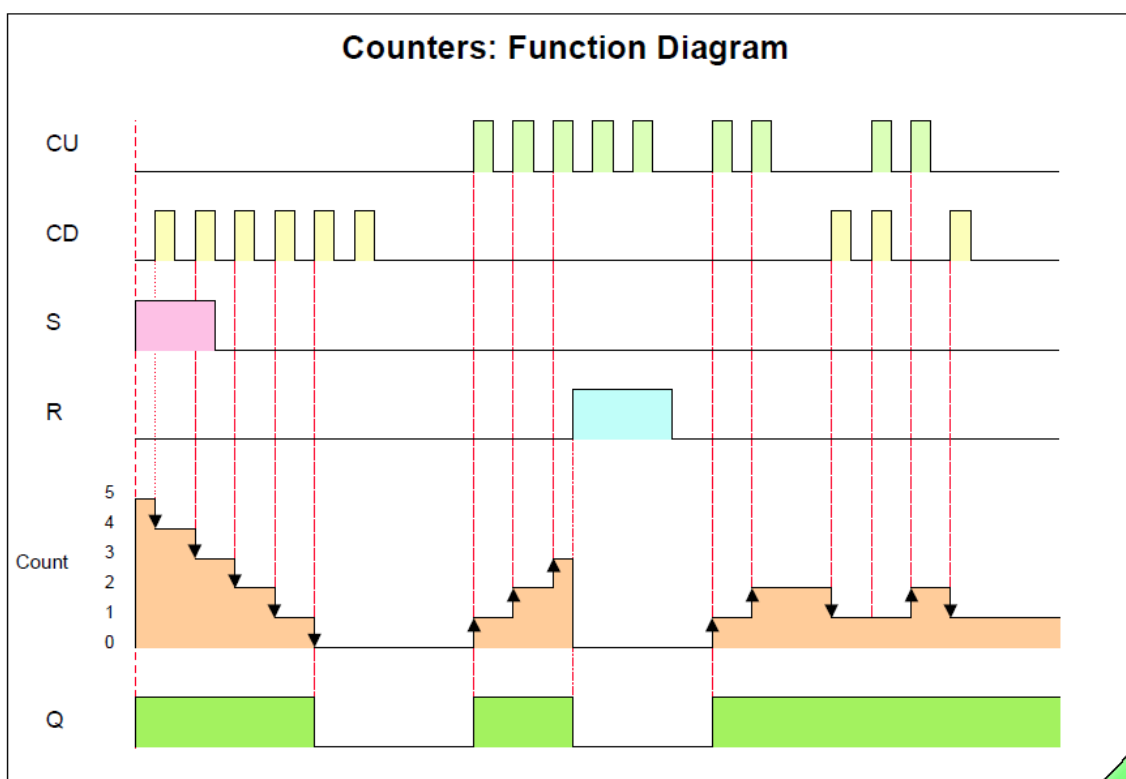
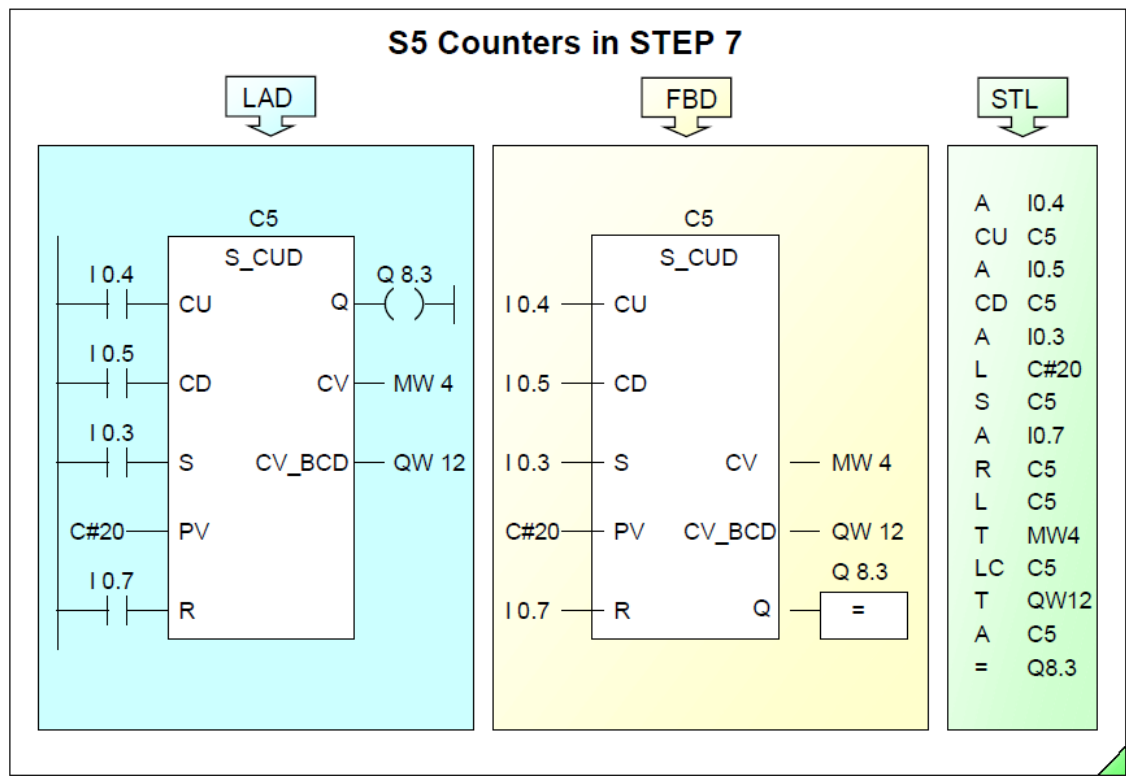


Examples of Load	L +5	//	16-bit constant (Integer)
	L L#523123	//	32-bit constant (Double Integer)
	L B#16#EF	//	byte in hexadecimal form
	L 2#0010 0110 1110 0011	//	16-bit binary value
	L 3.14	//	32-bit constant (Real)

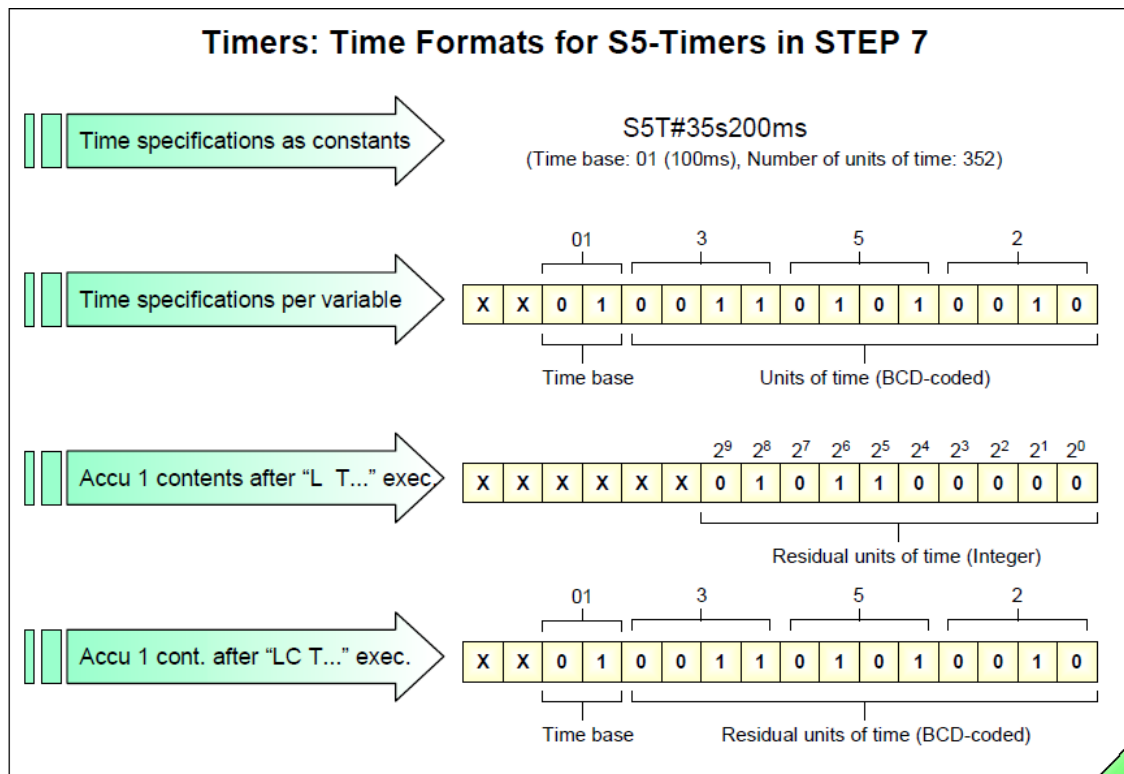
نحوه قرارگیری دیتا در آکومولاتور



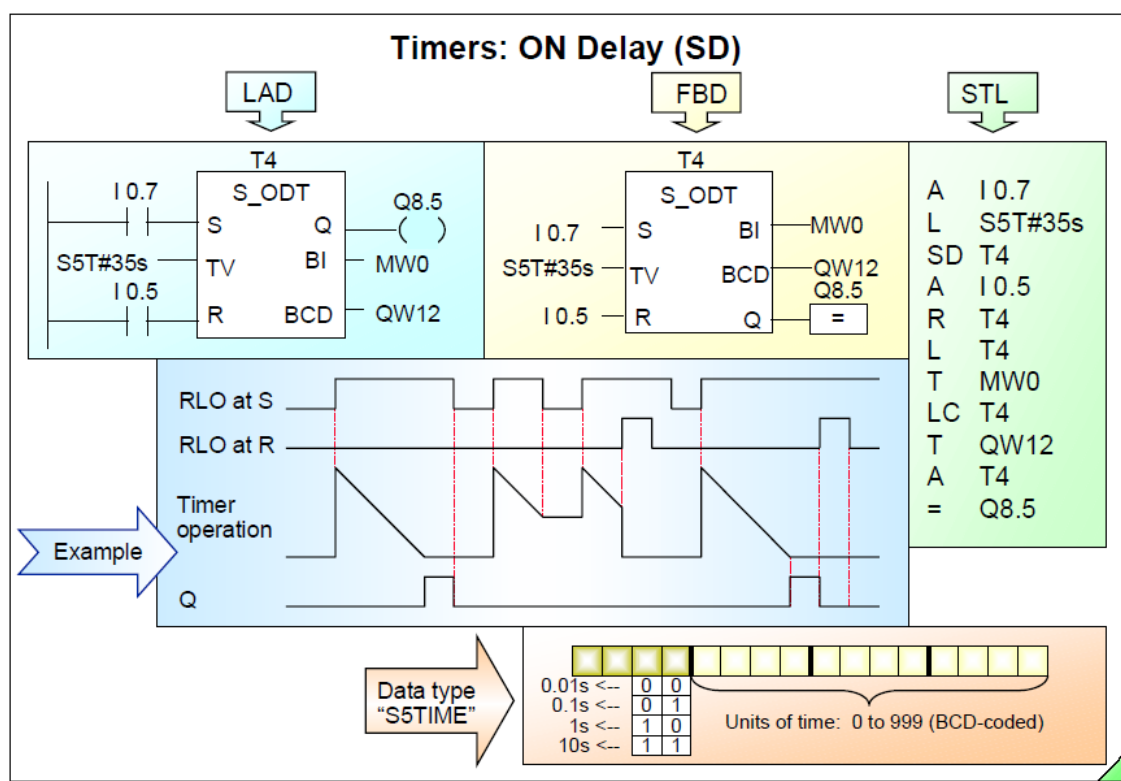
دستورات شمارنده



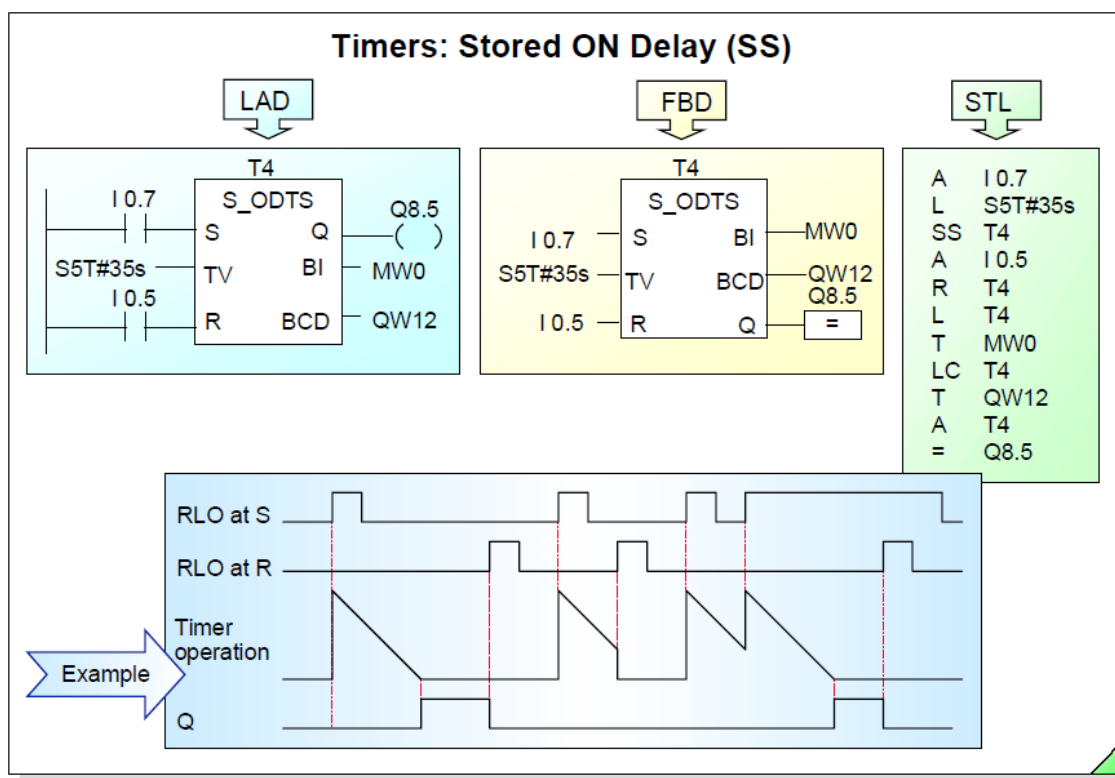
فرمت زمان در S7



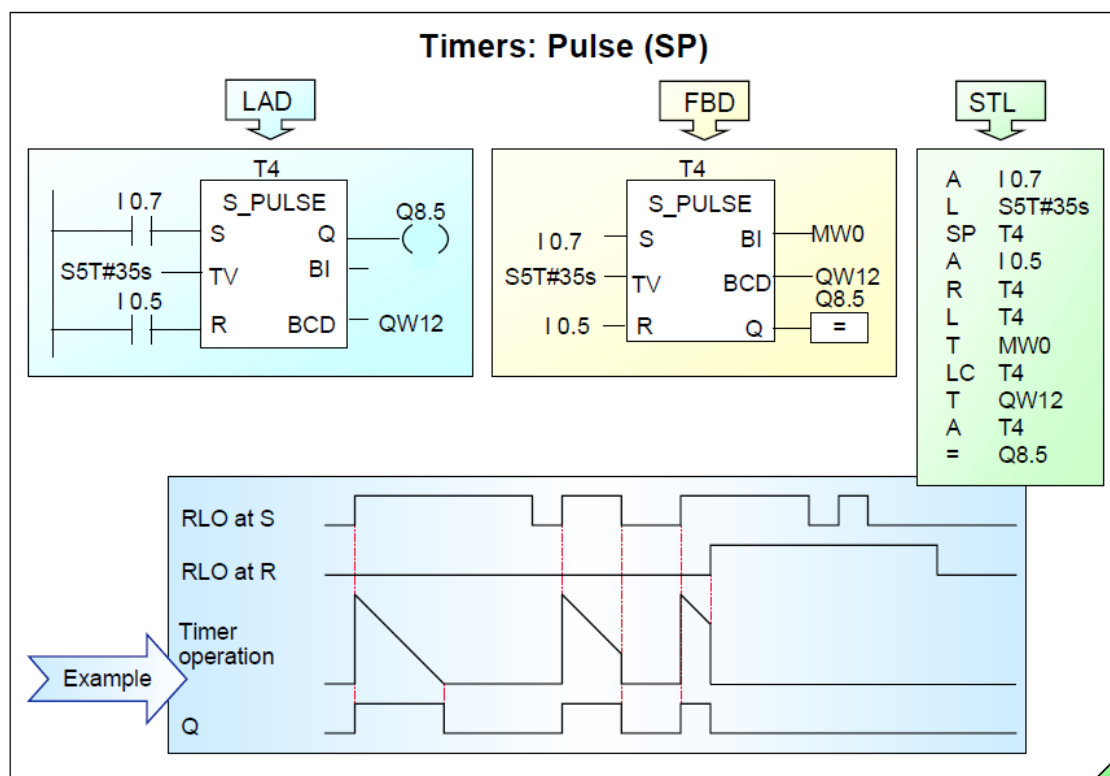
تایمر S-ODT



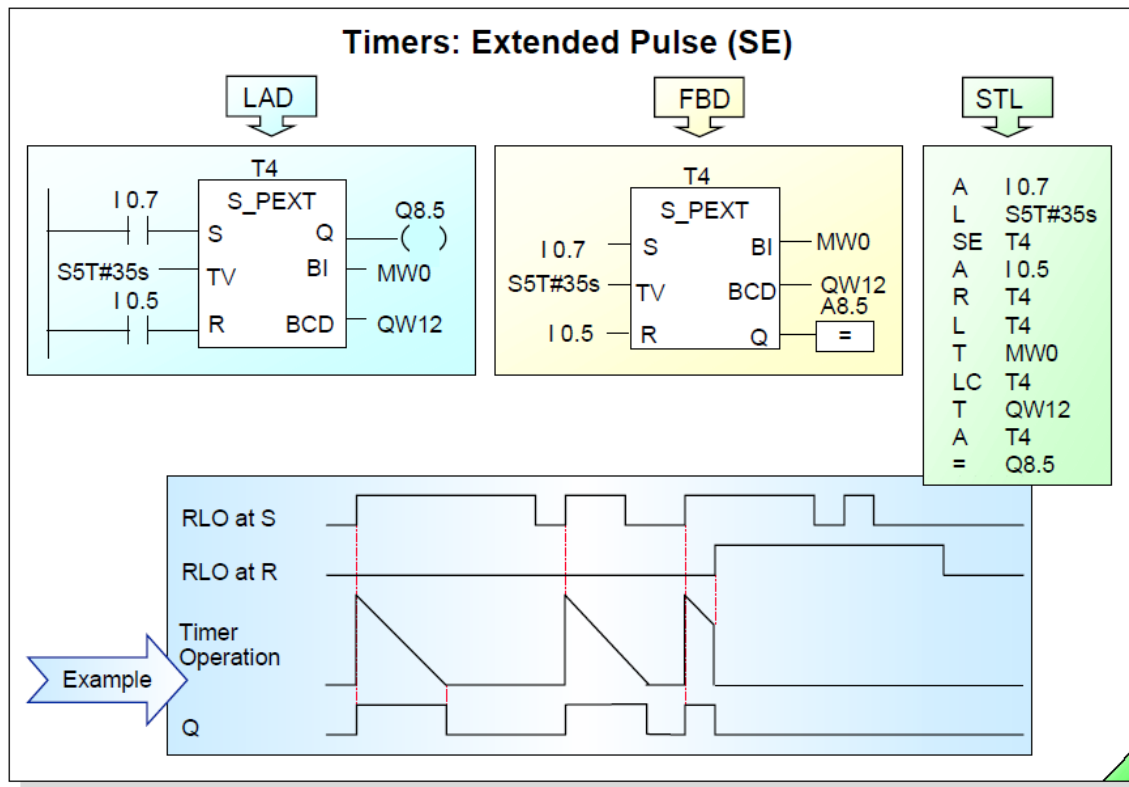
تایمر S-ODTS



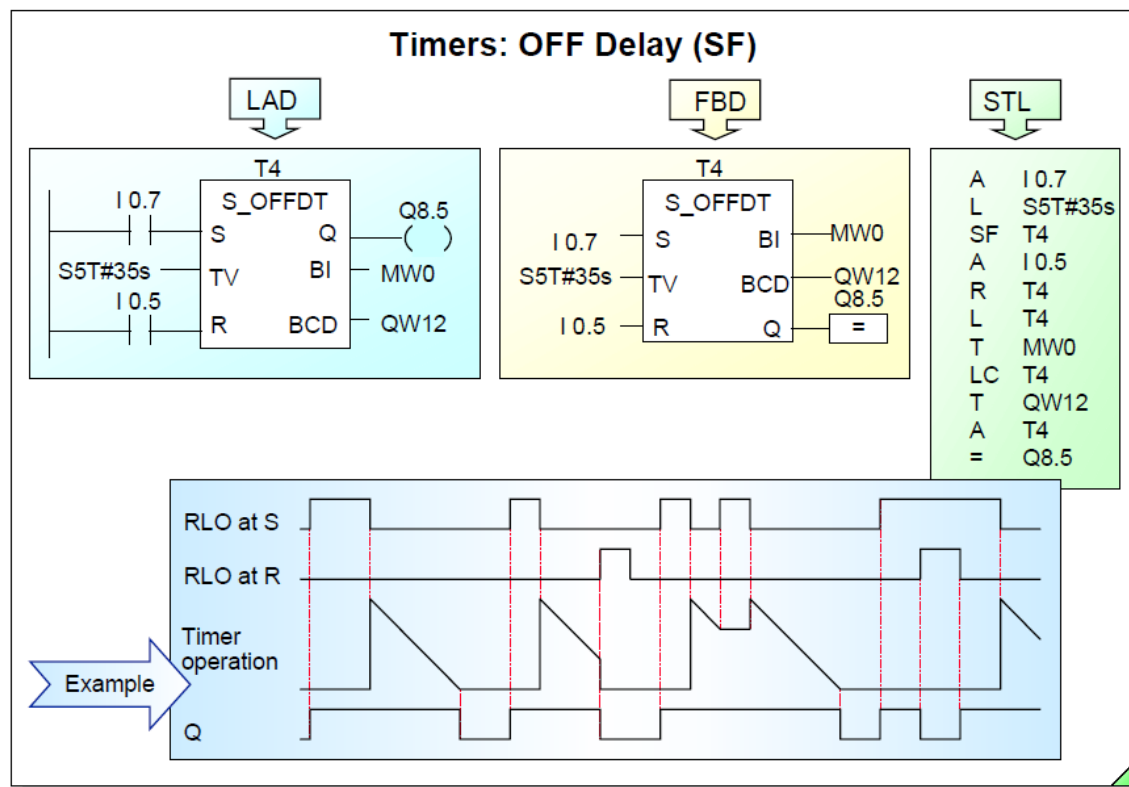
تایمر S-PULSE



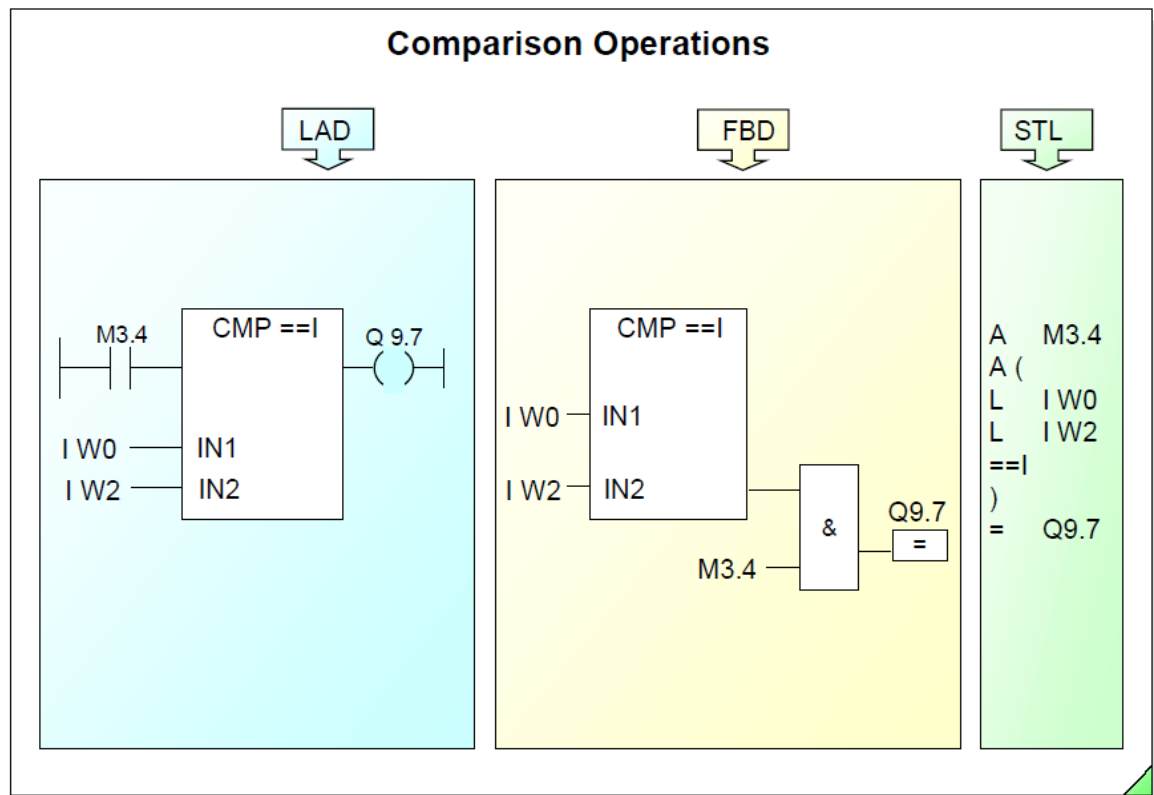
تایمر S-PEXT



تایمر S-OFFDT



دستورات مقایسه کننده

**CMP**

You can use comparison instructions to compare the following pairs of numerical values:

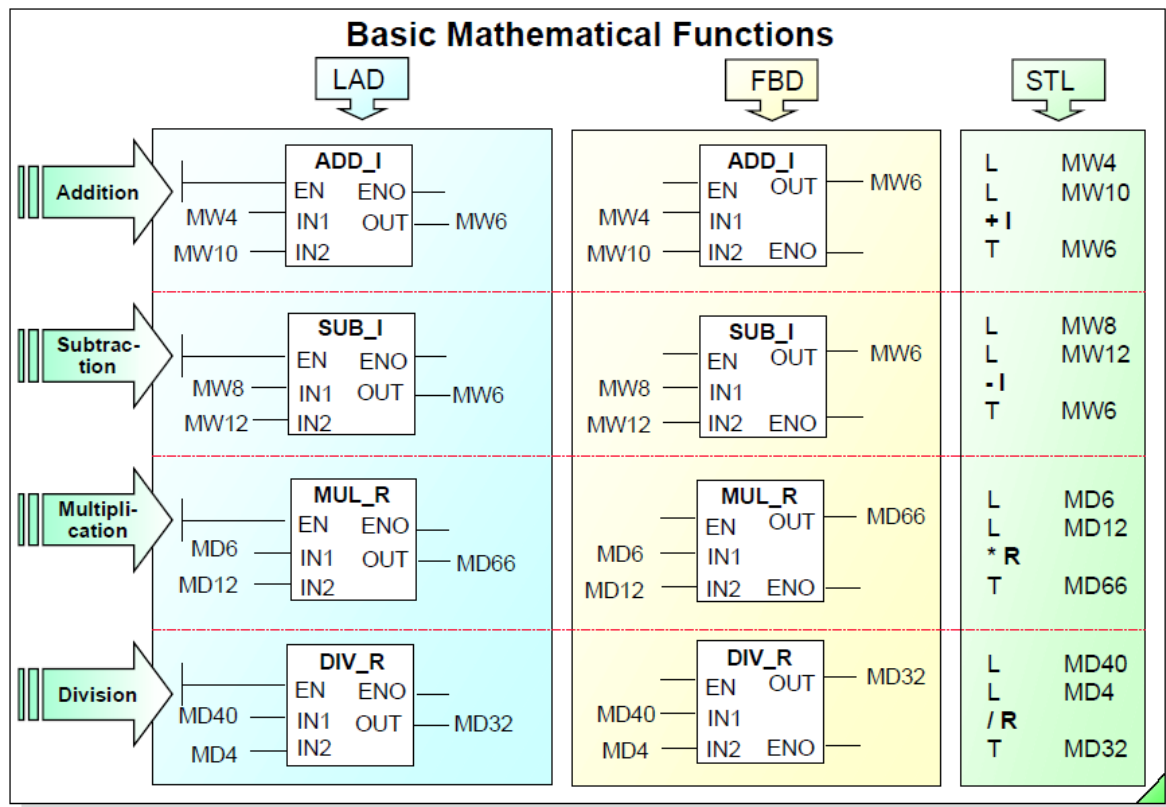
- I** Compare integers (on the basis of 16 bit fixed-point number)
- D** Compare integers (on the basis of 32 bit fixed-point number)
- R** Compare floating-point numbers (on 32 bit real number basis = IEEE floating-point numbers).

If the result of the comparison is "true", then the RLO of the operation is "1", otherwise it is "0".

The values at inputs IN1 and IN2 are compared for conformity with the specified condition:

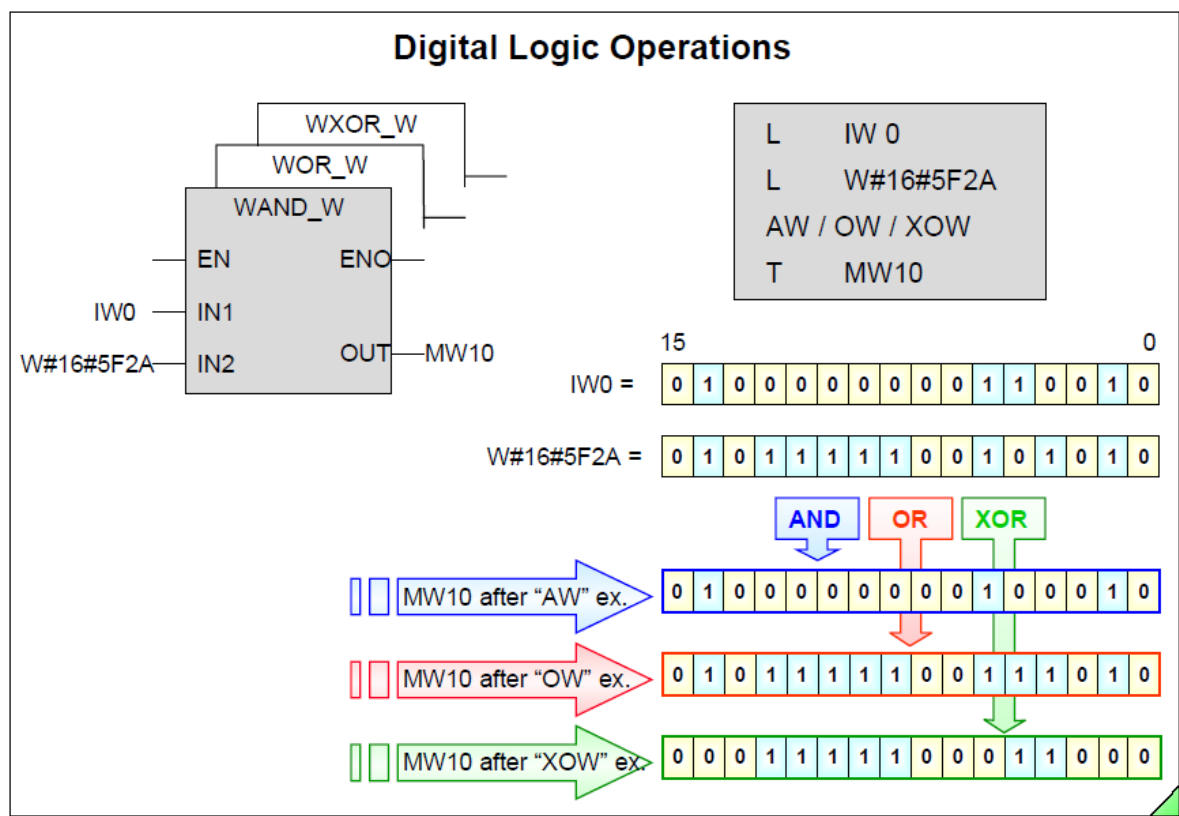
- ==** IN1 is equal to IN2
- <>** IN1 is not equal to IN2
- >** IN1 is greater than IN2
- <** IN1 is less than IN2
- >=** IN1 is greater than or equal to IN2
- <=** IN1 is less than or equal to IN2.

دستورات ریاضی



Addition: ADD_I Add integer
 ADD_DI Add double integer
 ADD_R Add real number
Subtraction: SUB_I Subtract integer
 SUB_DI Subtract double integer
 SUB_R Subtract real number
Multiplication: MUL_I Multiply integer
 MUL_DI Multiply double integer
 MUL_R Multiply real number
Division: DIV_I Divide integer
 DIV_DI Divide double integer
 DIV_R Divide real number

دستورات Logical Operations

**WAND_W**

The "AND Word" operation gates the two digital values at inputs IN1 and IN2 bit by bit in accordance with the AND truth table. The result of the AND operation is stored at the address at output OUT.

The instruction is executed when EN = 1.

Example: Masking out the 4th decade of the thumbwheel buttons :

IW4= = 0100 0100 1100 0100
W#16#0FFF = 0000 1111 1111 1111
MW30 = 0000 0100 1100 0100

WOR_W

The "OR Word" operation gates the two digital values at inputs IN1 and IN2 bit by bit in accordance with the OR truth table. The result of the OR operation is stored at the address at output OUT.

The instruction is executed when EN = 1.

Example: Setting bit 0 in MW32 :

MW32 = 0100 0010 0110 1010
W#16#0001 = 0000 0000 0000 0001
MW32 = 0100 0010 0110 1011

WXOR_W

The "Exclusive OR Word" operation gates the two digital values at inputs IN1 and IN2 bit by bit in accordance with the XOR truth table. The result of the OR operation is stored at the address at output OUT. The result of the XOR operation is stored at the address at output OUT.

The instruction is executed when EN=1.

Example: detecting signal changes in IW0 :

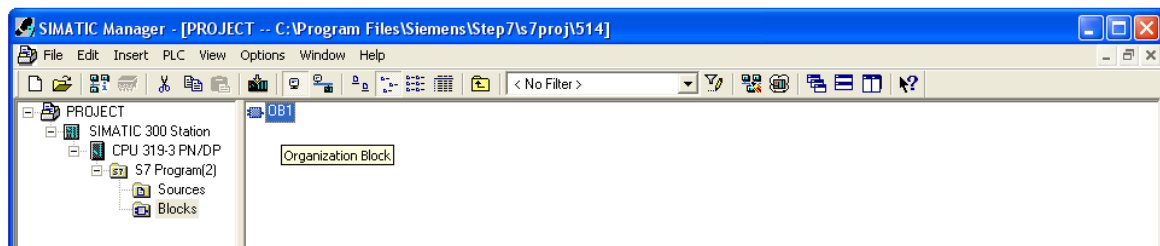
IW0 = 0100 0100 1100 1010
MW28 = 0110 0010 1011 1001
MW24 = 0010 0110 0111 0011

انواع داده در S7**Elementary Data Types in STEP 7**

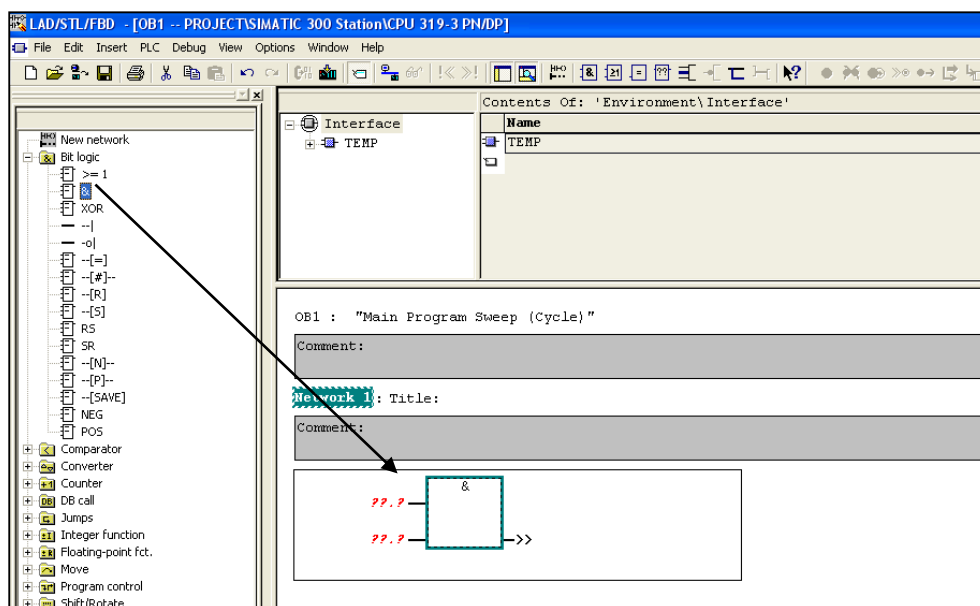
Keyword	Length (in bits)	Example of a constant of this type
BOOL	1	1 or 0
BYTE	8	B#16#A9
WORD	16	W#16#12AF
DWORD	32	DW#16#ADAC1EF5
CHAR	8	'w'
S5TIME	16	S5T#5s_200ms
INT	16	123
DINT	32	65539
REAL	32	1.2 or 34.5E-12
TIME	32	T#2D_1H_3M_45S_12MS
DATE	16	D#1993-01-20
TIME_OF_DAY	32	TOD#12:23:45.12

برنامه نویسی در محیط OB1

همانطور که می دانید بلوک اصلی برنامه کاربر OB1 می باشد. در این قسمت نیز جهت آشنایی هر چه بیشتر با دستورات برنامه نویسی تنها از بلوک OB1 به عنوان بلوک برنامه استفاده می کنیم. برای این منظور لازم است در نرم افزار پس از ساختن یک پروژه بر روی گزینه OB1 دابل کلیک تا وارد محیط برنامه نویسی این بلوک شوید.



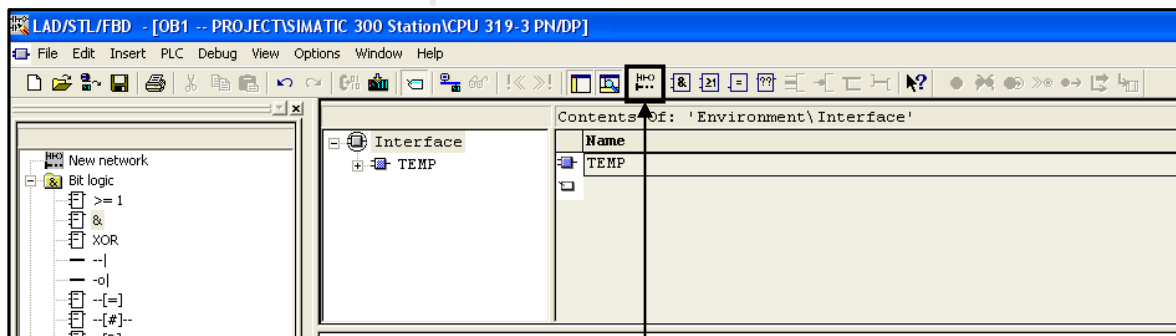
در بلوک OB1 گزینه ای تحت عنوان Bit Logic در پنجره دستورات وجود دارد که در زیر مجموعه این گزینه دستورات بیتی همانند تیغه ها، گیت های منطقی، فلیپ فلاپ ها و دستورات تشخیص لبه قرار گرفته شده اند.



البته لازم به ذکر است که پنجره دستورات در روش برنامه نویسی STL فاقد گزینه Bit Logic و سایر گزینه های مربوط به دستورات می باشد.

نحوه ساختن Network در برنامه

در بلوک های برنامه نویسی برنامه کنترلی می بایست در Network های مختلف وارد شود. در ادامه، برنامه های بررسی شده دارای چندین دستور فلیپ فلاپ می باشند که هر فلیپ فلاپ می بایست در یک Network وارد شود. این موضوع برای سایر دستورات نیز صادق می باشد. ساده ترین راه جهت ساخت Network در بلوک های برنامه استفاده از آیکون مشخص شده در شکل زیر می باشد.



New Network

با هر بار فعال کردن آیکون شکل فوق، یک Network جدید ساخته می شود.

OB1 : "Main Program Sweep (Cycle)"

Comment:

Network 1: Title:

Comment:

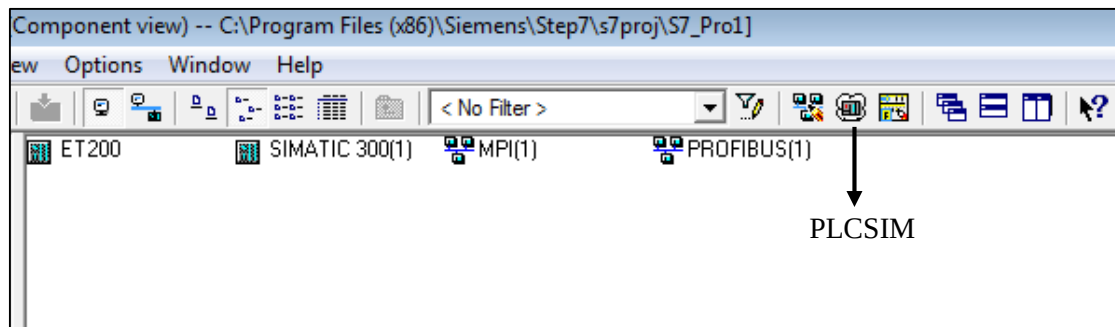
Network 2: Title:

Comment:

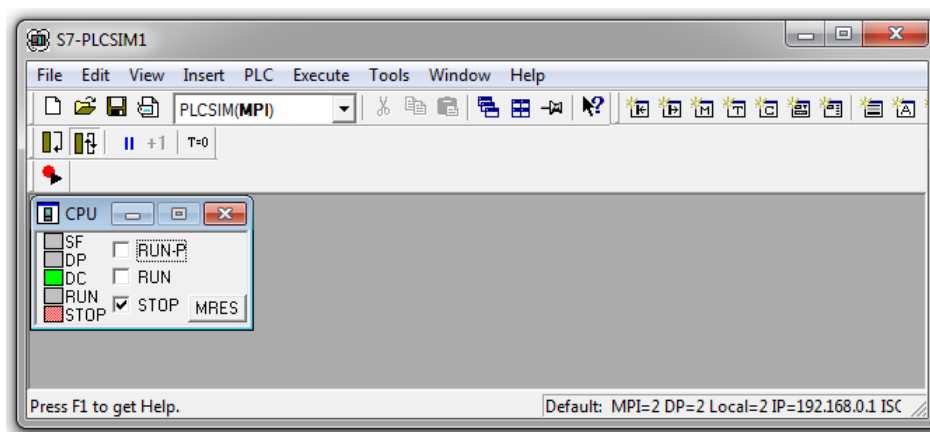
همانطور که در شکل فوق مشاهده می کنید، بلوک OB1 دارای دو Network می باشد که هر Network محل نوشتن دستورات مربوط به یک برنامه می باشد.

تست برنامه

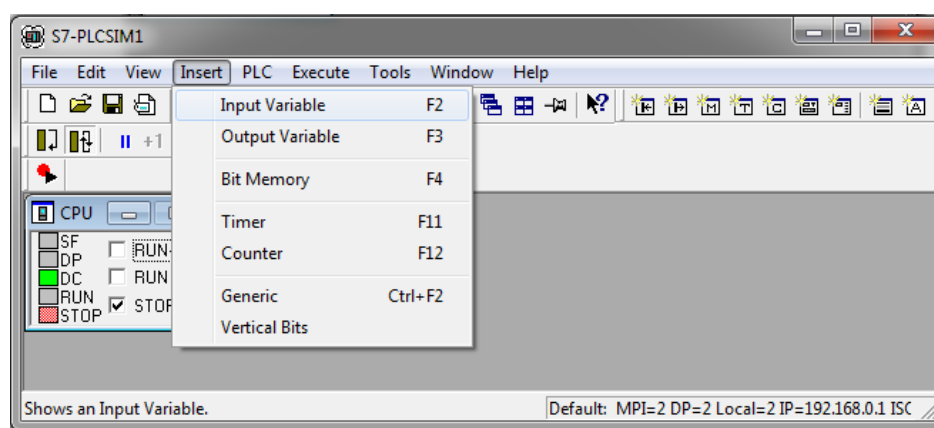
برای تست برنامه در محیط نرم افزار می بایست از نرم افزار PLCSIM استفاده کرد. برای باز کردن نرم افزار PLCSIM بر روی آیکون مشخص شده در شکل زیر کلیک می کنیم.



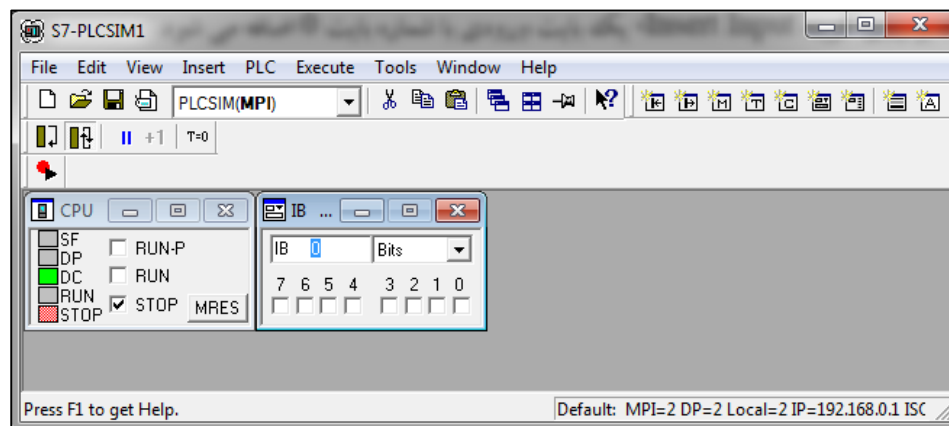
با کلیک بر روی آیکون شکل فوق، محیط شبیه ساز باز می شود.



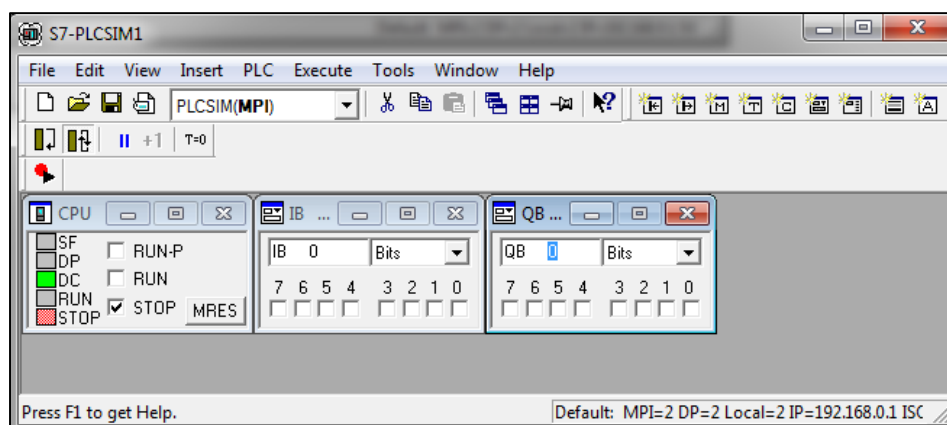
برای وارد کردن بایت های مربوط به ورودی و خروجی در محیط PLCSIM، به مسیر مشخص شده در شکل زیر مراجعه کنید.



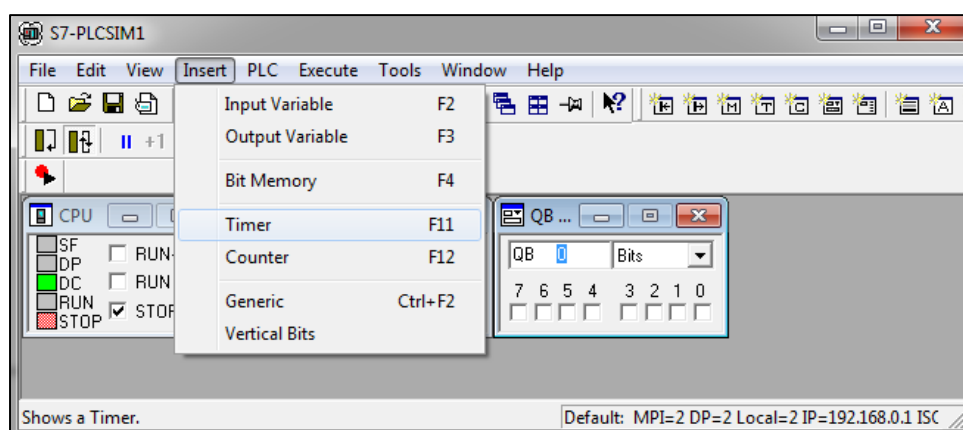
با کلیک بر روی گزینه Insert Input، یک بایت ورودی با شماره بایت 0 اضافه می شود.



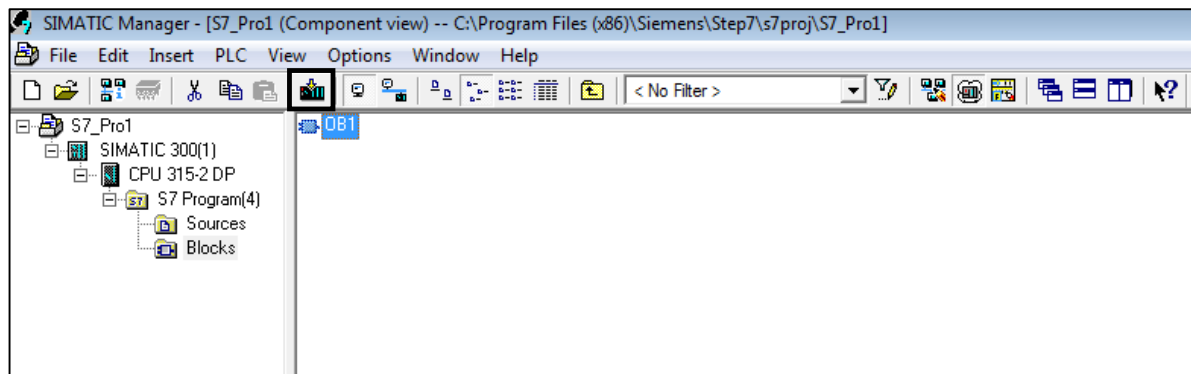
با کلیک بر روی گزینه Insert Output نیز یک بایت خروجی با شماره بایت 0 اضافه می شود.



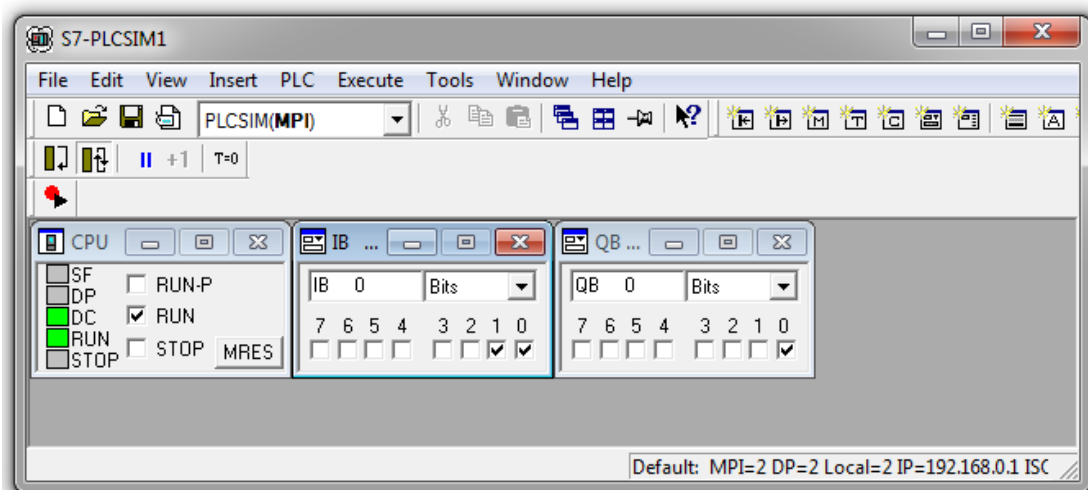
برای دیدن مقادیر تایمرها و کانترها نیز می توان از گزینه های مشخص شده در شکل استفاده نمود.



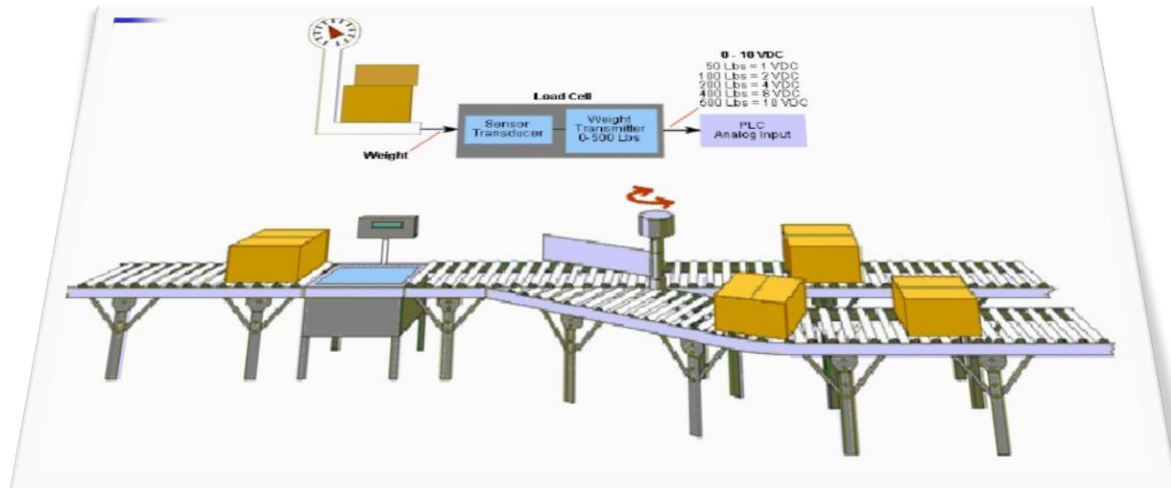
برای دانلود بلوک برنامه، زمانی که شبیه ساز در وضعیت STOP می باشد، بر روی گزینه دانلود کلیک کنید.



با دانلود بلوک برنامه، شبیه ساز را جهت تست در وضعیت RUN قرار می دهیم. در ادامه با فعال کردن بیت های ورودی، مطابق لاجیک نوشته شده برای برنامه، خروجی ها فعال میشوند.

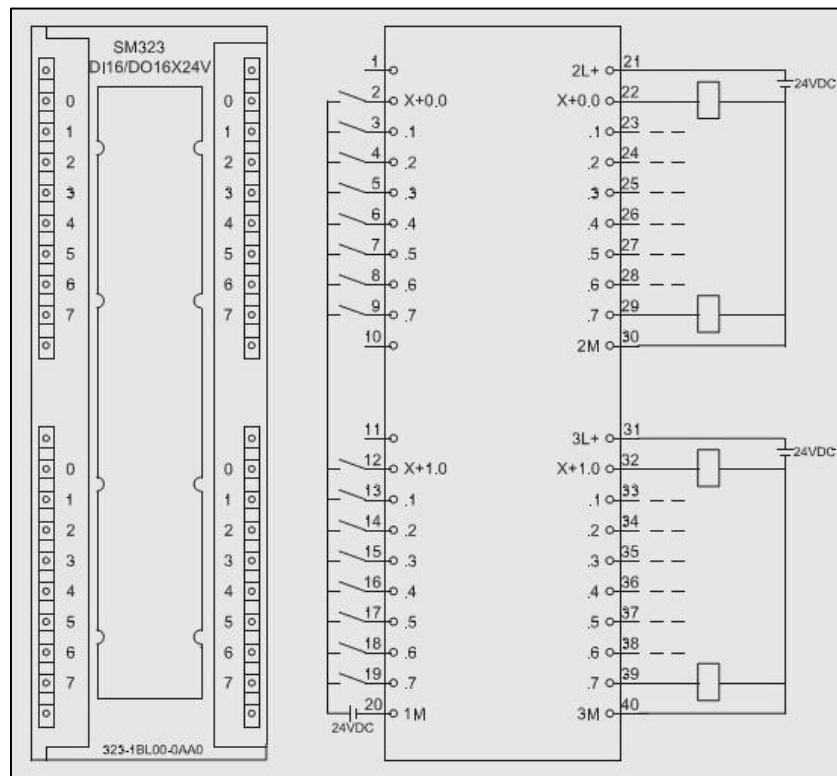


فصل سوم: برنامه نویسی در STEP7



سخت افزار

در تمامی مثال ها فرض بر این است که ورودی های متصل شده به کارت DI تماماً شستی استارت و خروجی های DO نیز به رله ها متصل شده اند.



روش های برنامه نویسی

STL: این زبان برنامه نویسی یک روش نوشتاری است که مجموعه دستورات به صورت کدهایی از پیش تعریف شده نوشته می شود. برخی از دستورات برنامه نویسی تنها در روش STL معتبر می باشند و معادل آنها در روش های گرافیکی تعریف نشده است. اکثر برنامه هایی که به این زبان نوشته می شوند قابل تبدیل به روش های گرافیکی نمی باشند.

LADDER و FBD زبان های برنامه نویسی گرافیکی هستند که لیست دستورات به صورت سمبل ها و یا بلوک هایی در دسترس می باشند. برنامه ای که به یکی از این روش ها نوشته می شود، قابل تبدیل به زبان نوشتاری می باشد.

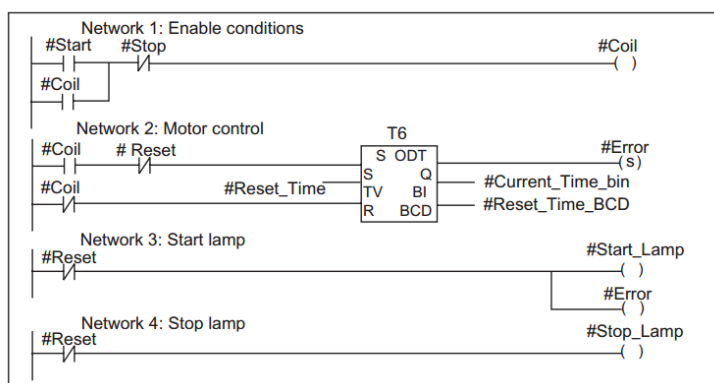
Example of Networks in Statement List

```

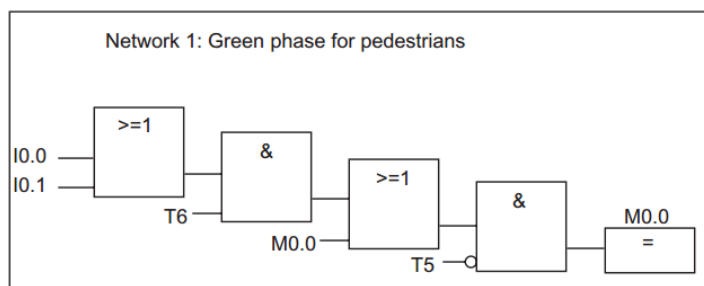
Network 1: Control drain valve
A(
O
O #Coil
)
AN #Close
= #Coil
Network 2: Display "Valve open"
A #Coil
= #Disp_open
Network 3: Display "Valve closed"
AN #Coil
= #Disp_closed

```

Example of Networks in LAD



Example of a Network in FBD



مثال ۱

در این مثال با فعال شدن ورودی $I0.0$ ، خروجی $Q0.0$ به صورت لحظه ای فعال می گردد.

روش LAD

Network 1 : Title:

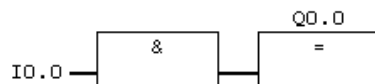
Comment:



روش FBD

Network 1 : Title:

Comment:



روش STL

Network 1 : Title:

Comment:

```

A      I      O.0
=      Q      0.0

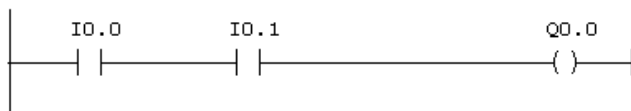
```

مثال ۲

در این مثال زمانی که هر دو ورودی I0.0 و I0.1 با هم فعال شوند، خروجی Q0.0 به صورت لحظه ای فعال می شود.(دستگاه کاتر)

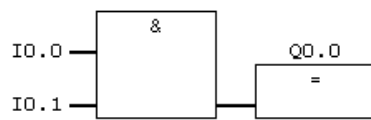
Network 1 : Title:

Comment:



Network 1 : Title:

Comment:



Network 1 : Title:

Comment:

A	I	O.0
A	I	O.1
=	Q	O.0

مثال ۳

در این مثال زمانی که هر کدام از ورودی های I0.0 یا I0.1 فعال شوند، خروجی Q0.0 به صورت لحظه ای فعال می گردد.

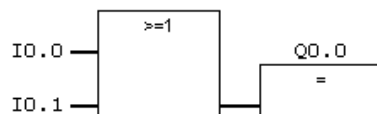
Network 1 : Title:

Comment:



Network 1 : Title:

Comment:



Network 1 : Title:

Comment:

O	I	0.0
O	I	0.1
=	Q	0.0

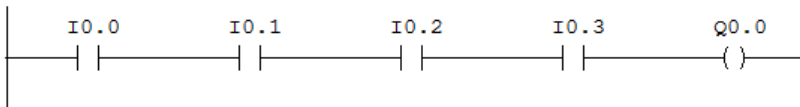
همانطور که می دانید منطق عملکرد AND معادل ضرب و منطق عملکرد OR معادل جمع در مدارات منطقی می باشد.

مثال ۴

پیاده سازی رابطه $(A.B.C.D)=F$

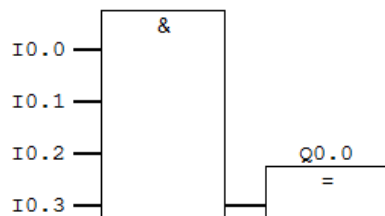
Network 1: Title:

Comment:



Network 1: Title:

Comment:



Network 1: Title:

Comment:

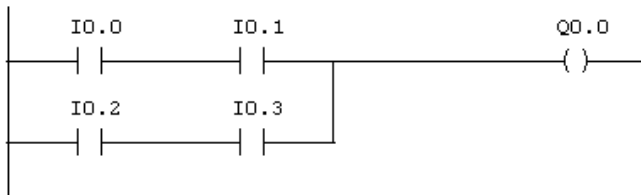
A	I	0.0
A	I	0.1
A	I	0.2
A	I	0.3
=	Q	0.0

مثال ۵

پیاده سازی رابطه $(A.B)+(C.D)=F$

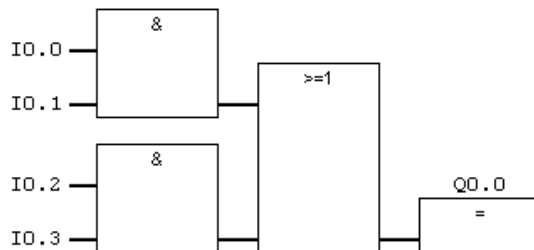
Network 1 : Title:

Comment:



Network 1 : Title:

Comment:



Network 1 : Title:

Comment:

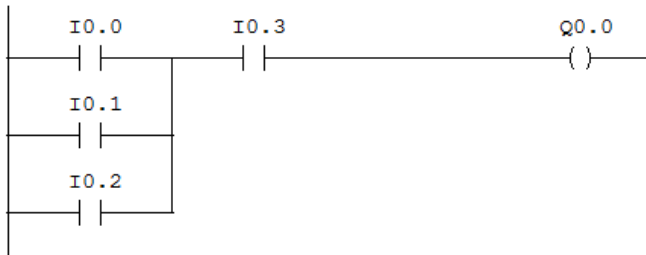
A	I	0.0
A	I	0.1
O		
A	I	0.2
A	I	0.3
=	Q	0.0

مثال ۶

پیاده سازی رابطه $(A+B+C).D=F$

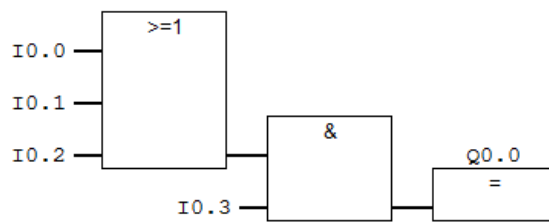
Network 1: Title:

Comment:



Network 1: Title:

Comment:



Network 1: Title:

Comment:

```

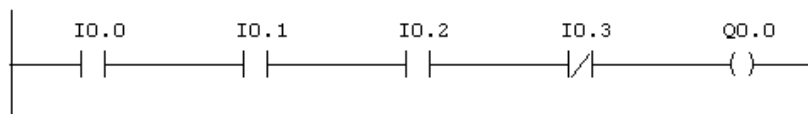
A(
O    I    0.0
O    I    0.1
O    I    0.2
)
A    I    0.3
=    Q    0.0
    
```

مثال ۷

در این مثال با فرض اینکه تمامی ورودی های متصل شده به PLC شاستی استارت می باشند، زمانی که ۳ ورودی اول فعال شوند، خروجی Q0.0 فعال می گردد. در این حالت زمانی که ورودی چهارم فعال شود، خروجی قطع می شود. در واقع در برنامه ورودی I0.3 به صورت NOT در نظر گرفته شده است.

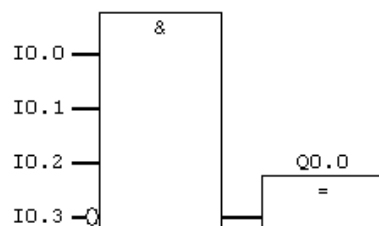
Network 1 : Title:

Comment:



Network 1 : Title:

Comment:



Network 1 : Title:

Comment:

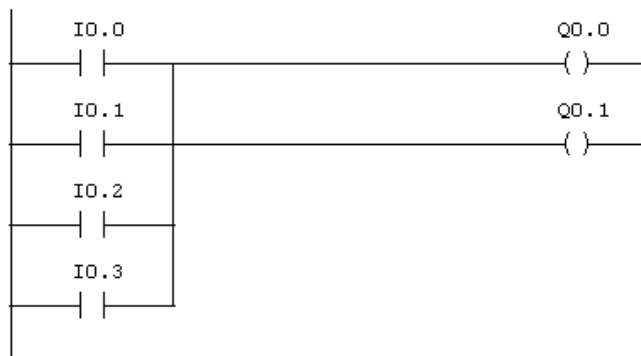
A	I	O.0
A	I	O.1
A	I	O.2
AN	I	O.3
=	Q	O.0

مثال ۸

در این مثال زمانی که هر یک از چهار شاستی متصل شده به ورودی فعال شوند، خروجی های Q0.0 و Q0.1 به صورت لحظه ای فعال می شوند.

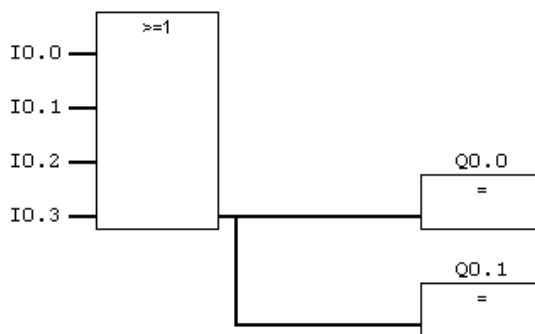
Network 1: Title:

Comment:



Network 1: Title:

Comment:



Network 1: Title:

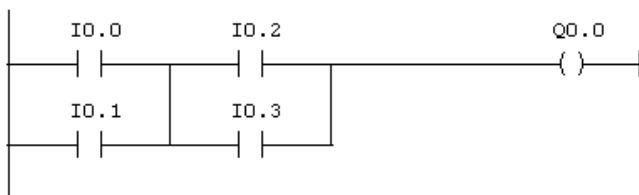
Comment:

O	I	0.0
O	I	0.1
O	I	0.2
O	I	0.3
=	Q	0.0
=	Q	0.1

مثال ۹پیاده سازی رابطه $(A+B).(C+D)=F$

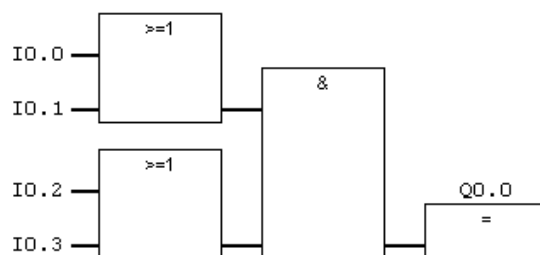
Network 1 : Title:

Comment:



Network 1 : Title:

Comment:



Network 1 : Title:

Comment:

```

A(
O      I      0.0
O      I      0.1
)
A(
O      I      0.2
O      I      0.3
)
=      Q      0.0

```

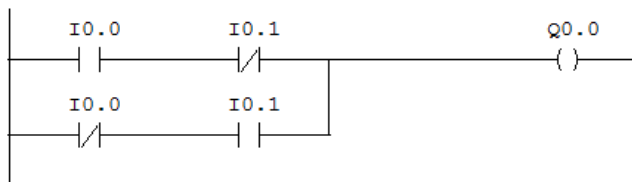
مثال ۱۰

پیاده سازی رابطه $XOR : (A'.B)+(A.B')=F$

کاربرد این منطق مثلاً برای مشخص کردن وضعیت کنترل فاز است. تحت شرایطی که هر سه فاز ورودی وصل و یا قطع باشند، بیزر خاموش است. وقتی برنامه کنترلی، دو فاز شدن و یا تک فاز شدن را تشخیص دهد، خروجی PLC (آلارم) فعال می شود.

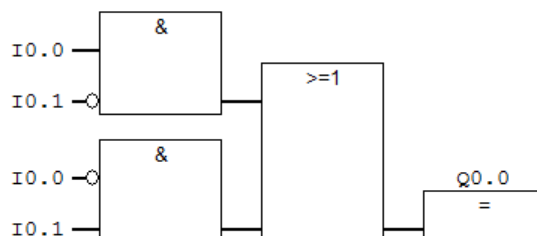
Network 1: Title:

Comment:



Network 1: Title:

Comment:



Network 1: Title:

Comment:

A	I	0.0
AN	I	0.1
O		
AN	I	0.0
A	I	0.1
=	Q	0.0

در مدارات فرمان، برای اجرای لاجیک فوق، می بایست از شاستی استپ استارت دابل استفاده کرد.

مثال ۱۱

در تمامی مثال بررسی شده، برنامه STL نشان داده شده، نتیجه تبدیل برنامه از روش های LAD یا FBD به روش STL بود. به عبارت دیگر این برنامه STL پیش فرض نرم افزار می باشد. در این مثال می خواهیم برنامه مثال ۵ را به روش STL از ابتدا طراحی کنیم. نکته قابل توجه این است که این برنامه از لحاظ عملکرد کاملاً شبیه برنامه مثال ۵ می باشد، ولی به روش های LAD یا FBD تبدیل نمی شود.

Network 1 : Title:

Comment:

A	I	0.0
A	I	0.1
=	M	0.0
A	I	0.2
A	I	0.3
=	M	0.1
O	M	0.0
O	M	0.1
=	Q	0.0

مثال ۱۲

بازنویسی مثال ۹ به روش STL

Network 1 : Title:

Comment:

O	I	0.0
O	I	0.1
=	M	0.0
O	I	0.2
O	I	0.3
=	M	0.1
A	M	0.0
A	M	0.1
=	Q	0.0

مثال ۱۳

برنامه ای به زبان های گرافیکی بنویسید که منطق آن مشابه منطق STL شکل زیر باشد.

Network 1 : Title:

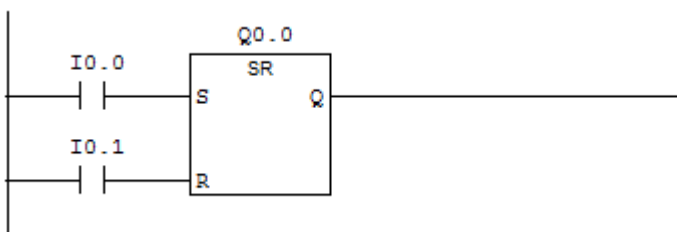
Comment:

O	I	0.0
O	I	0.1
=	M	0.0
A	M	0.0
A	I	0.2
=	M	0.1
O	M	0.1
O	I	0.3
=	Q	0.0

❖ تا اینجا در تمام مثال ها از خروجی لحظه ای استفاده کردیم. در این حالت با برداشتن دست از روی شاستی خروجی قطع میشود و در حالت شبیه سازی وضعیت ورودی ها را به صورت کلید نگه میداشتیم تا بتوانیم نتایج خروجی را ببینیم. اما با توجه به اینکه در عمل، اکثر مدارهای کنترلی به صورت دائم کار می باشند و از آنجایی که تمامی باتن ها به صورت شاستی در نظر گرفته می شوند، می بایست خروجی ها به صورت دائم کنترل شوند.

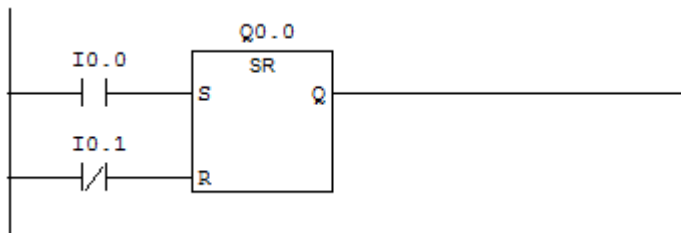
مثال ۱۴

کنترل یک موتور از یک نقطه به صورت دائم



نکته ای که در استفاده از دستور فلیپ فلاپ وجود دارد این است که پایه های فلیپ فلاپ حساس به لبه بالارونده ی ورودی هستند و ما در این مثال نیز از شاستی استارت برای ریست استفاده کردیم. اگر معادل مدار فرمان از شاستی استپ استفاده کنیم که دارای تیغه ی N.C هست، برای اینکه اثر یک بودن را بر روی پایه ی ریست برداریم باید حالت تیغه بسته ی شاستی استپ را روی پایه ی ریست قرار دهیم.

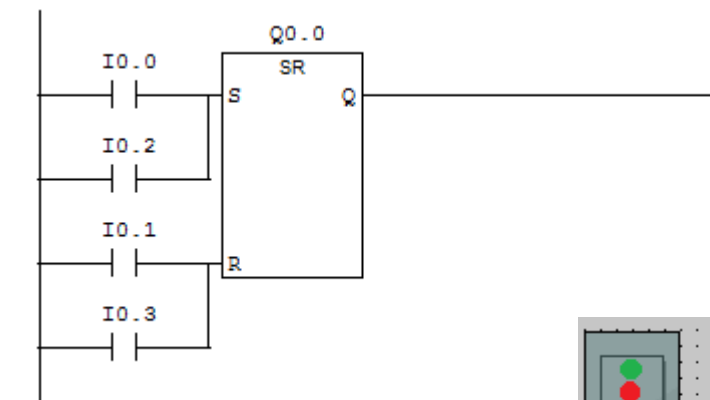
ادامه



در ادامه بحث فرض بر این است که کلیه ی کلید ها و شاستی ها از جنس استارت هستند.

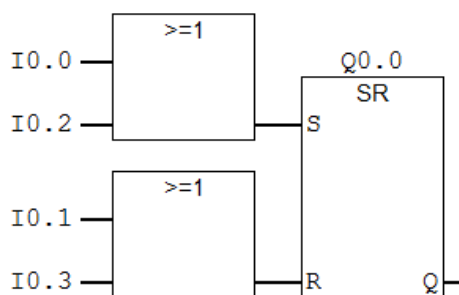
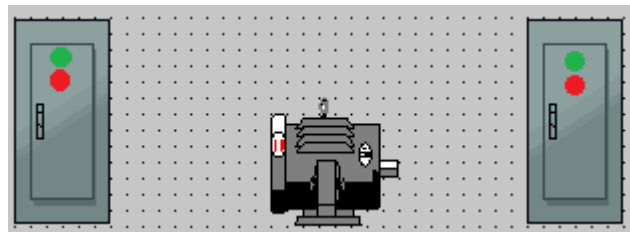
مثال ۱۵

کنترل یک موتور از دو نقطه به صورت دائم



تابلو اول : استارت I0.0 و استپ I0.1

تابلو دوم : استارت I0.2 و استپ I0.3

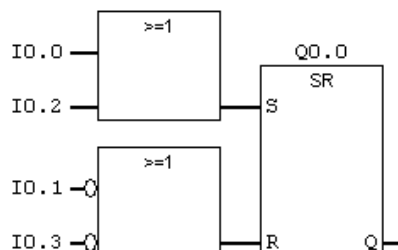


آموزش روش STL در قالب مثال های جداگانه آموزش داده می شود.

فرض کنید در مثال قبل ورودی هایی که به عنوان ریست در نظر گرفته شده اند، از جنس استپ باشند. برنامه به صورت زیر تغییر می کند.

Network 1 : Title:

Comment:

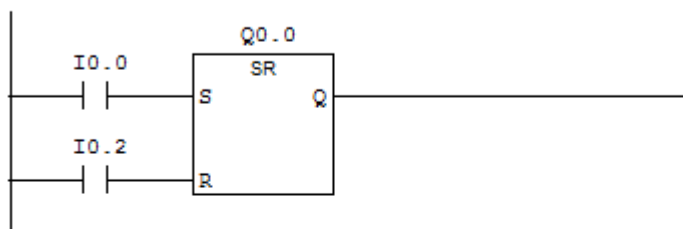


مثال ۱۶

کنترل دو موتور از دو نقطه به صورت مجزا

Network 1 : Title:

Comment:



استارت موتور اول I0.0

استارت موتور دوم I0.1

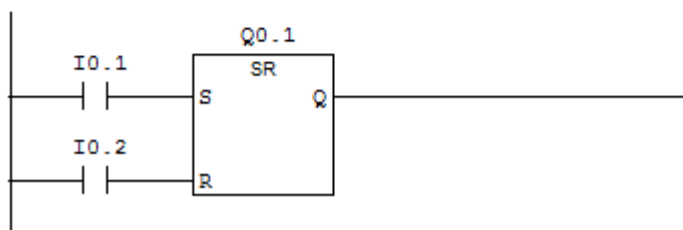
استپ کل I0.2

موتور اول Q0.0

موتور دوم Q0.1

Network 2 : Title:

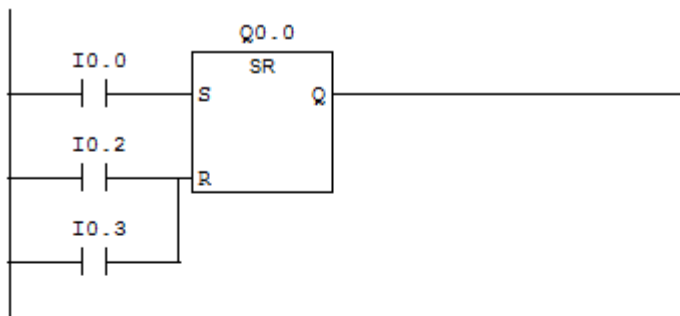
Comment:



در مثال قبل با یک استپ کل هر دو موتور متوقف می شوند. در صورتی که هر یک از موتورها استپ جداگانه داشته باشند، برنامه به صورت زیر خواهد بود.

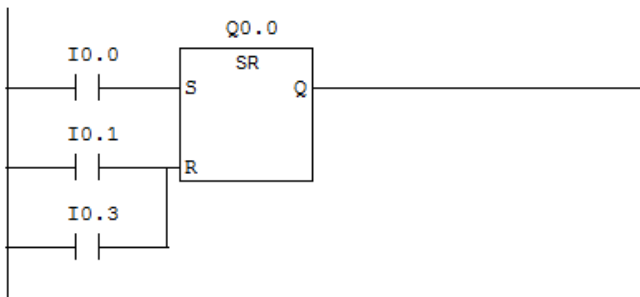
مثال ۱۷

مدار یکی پس از دیگری



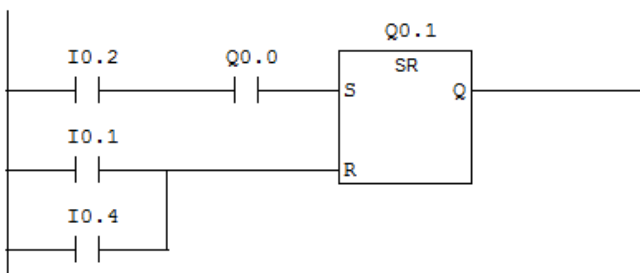
Network 1 : Title:

Comment:



Network 2 : Title:

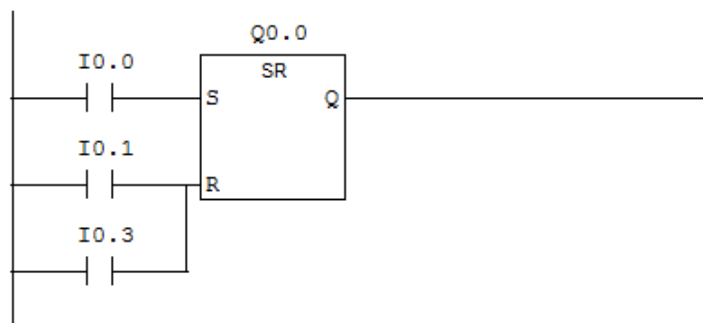
Comment:



در برنامه مثال قبل، با فشردن شاستی I0.3، موتور اول خاموش می شود، ولی موتور دوم به کار خود ادامه می دهد و هیچ گونه وابستگی بعد از روشن شدن به خروجی Q0.0 ندارد.

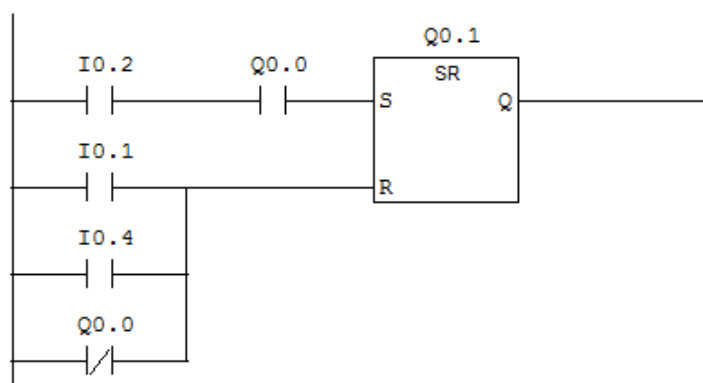
Network 1: Title:

Comment:



Network 2: Title:

Comment:



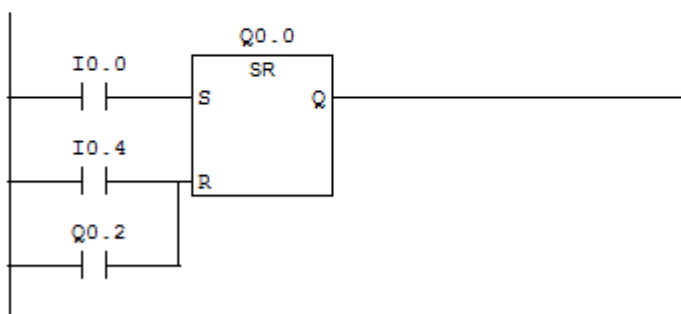
در این مثال اگر پمپ روغن به هر دلیلی خاموش شود، موتور مته نیز خاموش می شود. با زدن استپ I0.3 که مربوط به پمپ روغن می باشد، می توان عملکرد این برنامه را تست نمود.

مثال ۱۸

در یک سیستم صنعتی سه عدد شیر وجود دارد. هر یک از شیرها با شاستی استارت خود باز می شوند مشروط به اینکه شیر دوم حتما بعد شیر اول و شیر سوم بعد شیر دوم باز شود. باز شدن شیر سوم باعث بسته شدن دو شیر قبلی می شود.

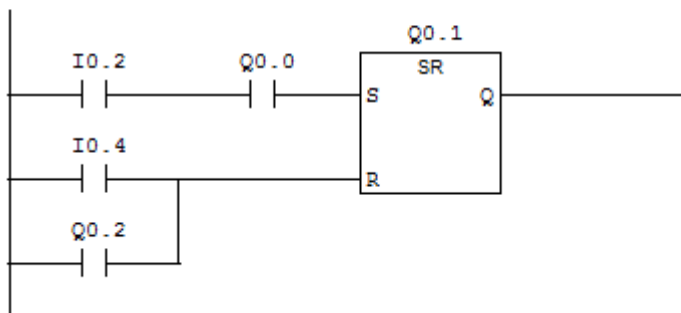
Network 1 : Title:

Comment:



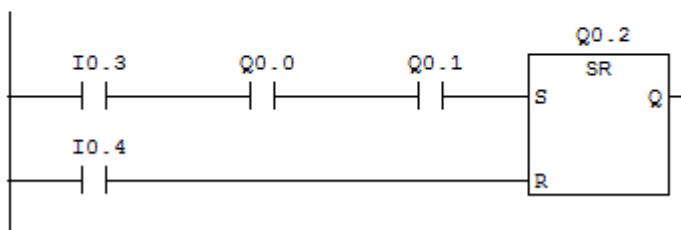
Network 2 : Title:

Comment:



Network 3 : Title:

Comment:

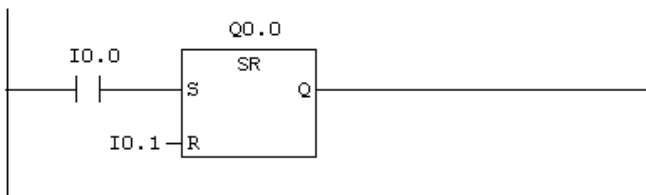


مثال ۱۹

دو موتور داریم که با زدن شاستی اول موتور اول روشن می شود. تا زمانیکه دست روی شاستی هست، موتور اول تنها کار می کند. با برداشتن دست از روی شاستی موتور دوم نیز روشن می شود.

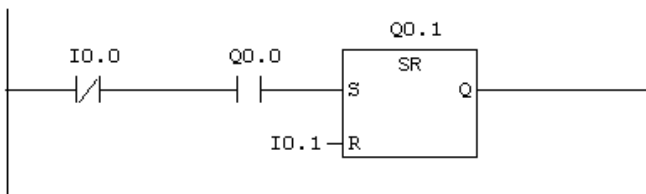
Network 1 : Title:

Comment:



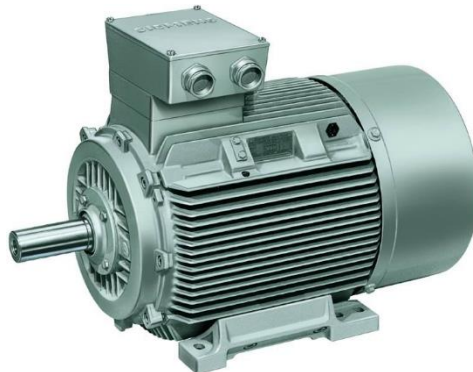
Network 2 : Title:

Comment:



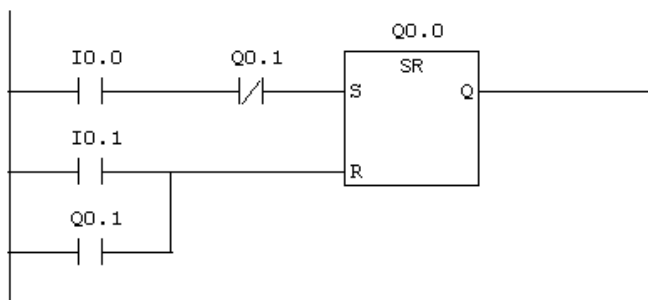
مثال ۲۰

چپگرد راستگرد کند با حفاظت کامل



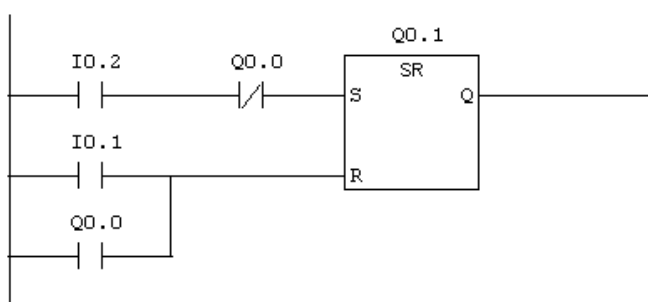
Network 1 : Title:

Comment:

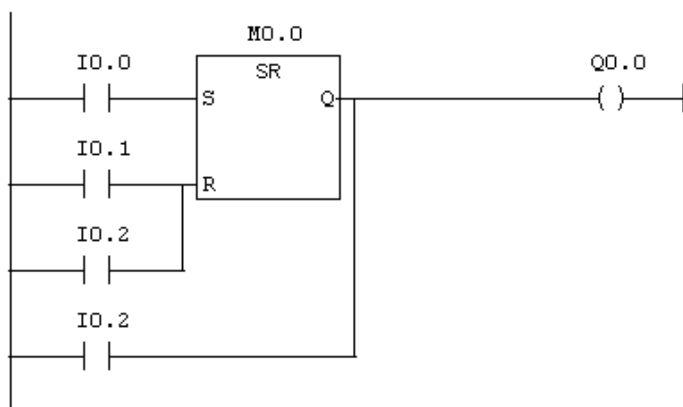


Network 2 : Title:

Comment:



مثال ۲۱



کنترل یک موتور به صورت لحظه ای و دائم

IO.0: شاستی استارت حالت دائم

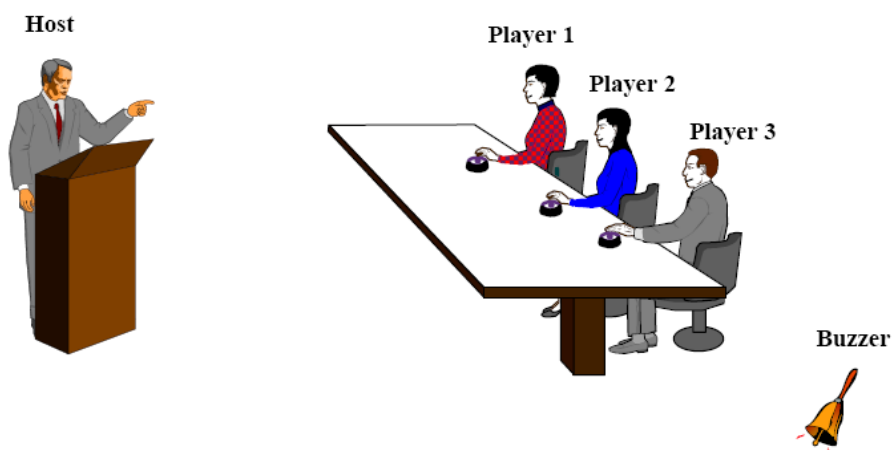
IO.1: شاستی استاپ حالت دائم

IO.2: پدال حالت لحظه ای

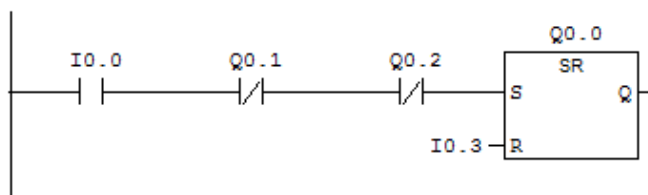
تمرین ۲۲

میز مسابقه برای سه شرکت کننده

می خواهیم برنامه مربوط به سه میز مسابقه را طراحی نماییم. بطوری که با فشردن شاستی میز اول، لامپ آن میز روشن شود و در این لحظه اگر شاستی میز دوم و سوم فشرده شد، در مدار هیچ تغییری حاصل نگردد. این مطلب برای سایر شرکت کننده ها نیز صادق است.

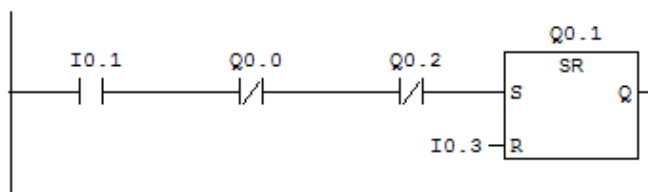


برنامه



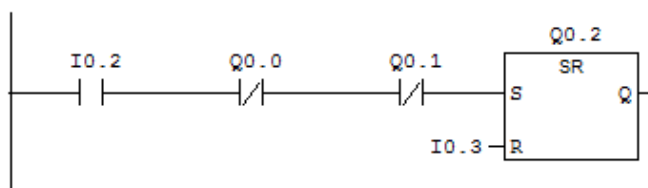
Network 2 : Title:

Comment:



Network 3 : Title:

Comment:

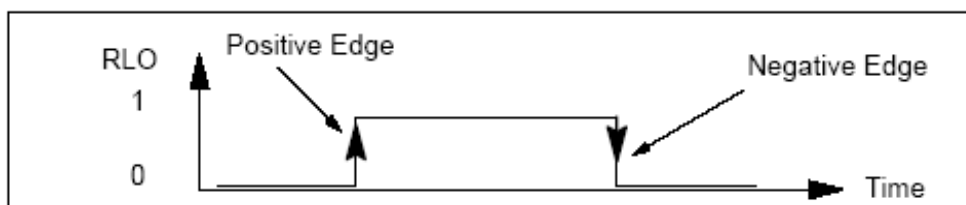


تمرین

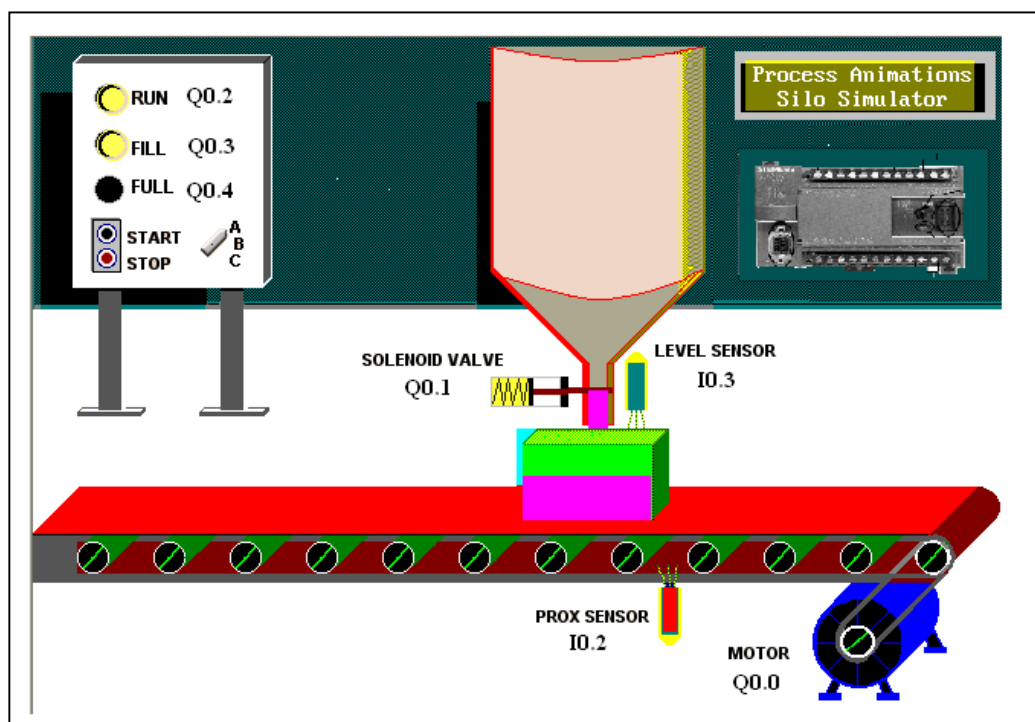
در یک سیستم صنعتی پنج مصرف کننده وجود دارد که از یک منبع، تغذیه می شوند. توان مصرفی هریک تجهیزات به شرح زیر است : P1:5 kw , P2:2 kw , P3:1 kw , P4:7 kw , P5:9 kw

در این پروژه تنها تعدادی از مصرف کننده ها می توانند همزمان روشن شوند که مجموع بار مصرفی آنها کمتر یا برابر 10 kw باشد. هر مصرف کننده دارای یک استارت و استپ مجزا می باشند.

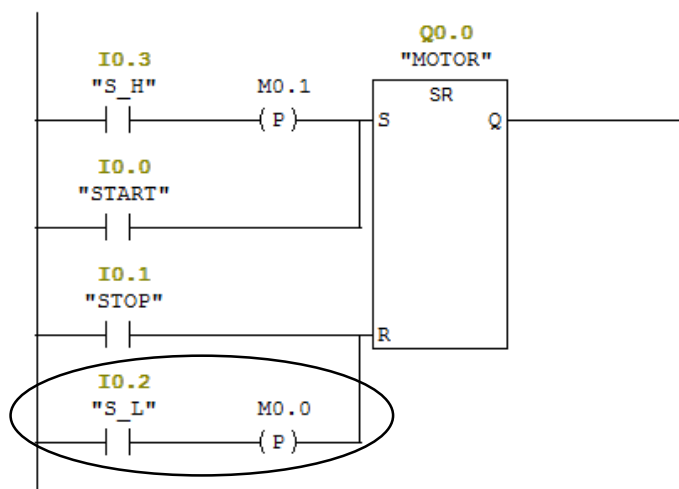
این دستورات یکی دیگر از دستورات پر کاربرد در برنامه نویسی می باشند. توسط این دستورات می توان لبه های ایجاد شده (بالا یا پایین) در یک پالس را آشکار نمود. با استفاده از این دستورات تغییر ورودی از قطع به وصل (خاموش به روشن) یا از وصل به قطع (روشن به خاموش) آشکار می شود. مثلاً هر گاه آشکارسازی قطع به وصل (لبه بالا رونده) یک حسگر مورد نظر باشد، PLC به ازای هر قطع به وصل در یک مرور اجازه عبور برنامه از طریق اتصال مربوطه را می دهد. دستور تغییر از وصل به قطع (روشن به خاموش) نیز به همین ترتیب و در جهت عکس دستور قبل عمل می کند. مثلاً یک کلید را فرض کنیم. زمانی که کلید را فشار می دهیم، اگر از دستور تغییر مثبت استفاده کرده باشیم خروجی فعال می شود. ولی اگر از دستور تغییر منفی استفاده کرده باشیم، زمانی که کلید را رها کنیم خروجی فعال می شود.



این دستورات دارای کاربردهای فراوانی در برنامه نویسی می باشند و در بسیاری از برنامه ها کار برنامه نویسی را برای کاربران ساده و سریع می کنند. در ادامه به چند کاربرد این دستورات با بررسی مثال های مختلف توجه فرمایید.

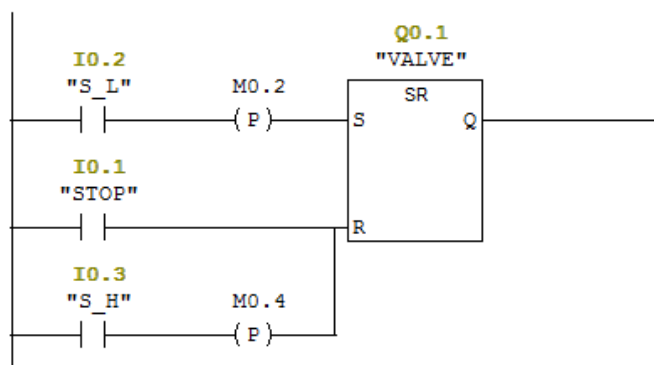


برنامه مثال ۲۳



Network 2: Title:

Comment:



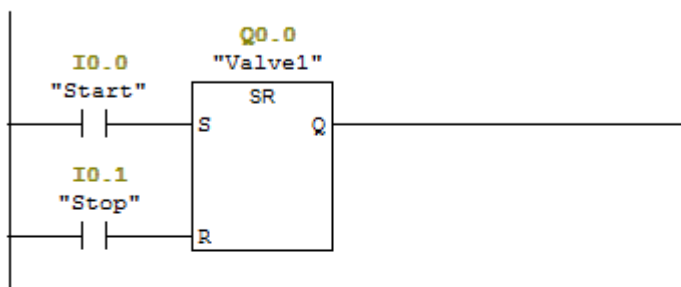
در این مثال قرار دادن دستور P بر روی ورودی I0.2 در NET1 ضروری می باشد. ولی قرار دادن دستور P برای هر دو سنسور در هر دو NET مشکلی ایجاد نمی کند.

آدرس حافظه ای که بالای دستور حساس به لبه ی بالا اختصاص می یابد، جهت انتقال نتیجه به محل دیگر کاربرد دارد.

مثال ۲۴) کاربرد دستور حساس به لبه پایین رونده: در یک سیستم صنعتی دو شیر وجود دارد که بسته شدن یکی منجر به باز شدن دیگری می شود. در این حالت داریم:

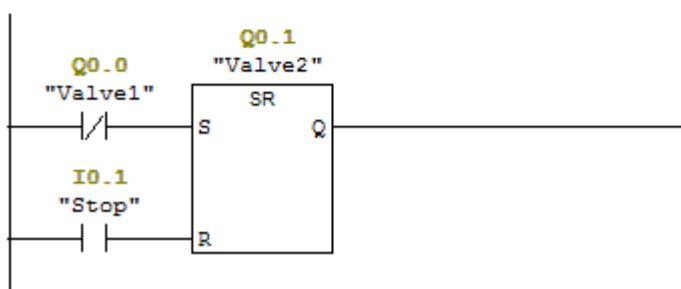
Network 1: Title:

Comment:



Network 2: Title:

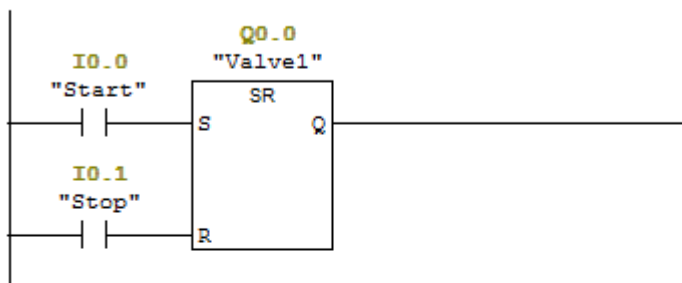
Comment:



در این روش برنامه نویسی از لحظه ی اول Valve2 باز می شود. برای اینکه حتما تغییر حالت بسته شدن شیر ۱ را شرط باز شدن شیر ۲ قرار دهیم، از دستور حساس به لبه ی پایین استفاده میکنیم.

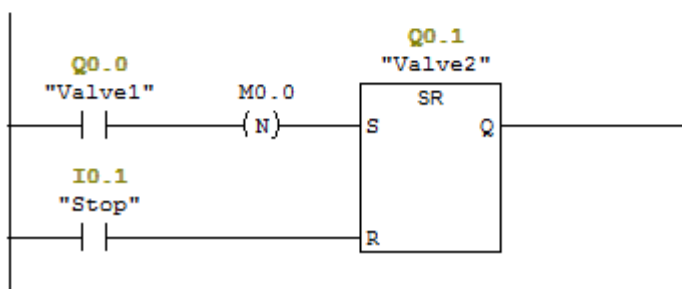
Network 1: Title:

Comment:



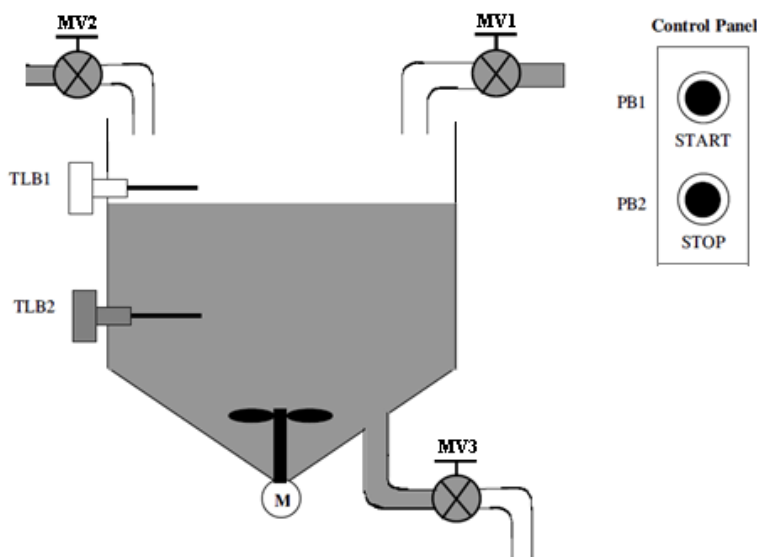
Network 2: Title:

Comment:



مثال ۲۵- کنترل شارژ و تخلیه مخزن

در یک فرآیند صنعتی دو مایع می بایست در یک مخزن با یکدیگر میکس شوند. نحوه کار بدین صورت می باشد که با فشردن شستی استارت PB1 تعبیه شده بر روی پانل، شیرهای ورودی MV1 و MV2 باز می شوند. در همین زمان موتور مربوط به میکسر نیز روشن می گردد. وقتی سطح مایع از سنسور TLB2 عبور و به سنسور TLB1 رسید، شیرهای ورودی بسته و موتور نیز استپ می شود. در مرحله بعد شیر MV3 باز شده و عمل تخلیه صورت می گیرد. وقتی سطح مایع به زیر سنسور TBL2 رسید، شیر تخلیه MV3 و مجددا شیرهای ورودی برای شارژ جدید باز شوند. فشردن شستی استپ نیز در هر مرحله کل سیستم را متوقف می کند.

**لیست I/O :**

استارت PB1 : I0.0

استپ PB2 : I0.1

سنسور حد بالای مخزن TBL1 : I0.2

سنسور حد پایین مخزن TBL2 : I0.3

شیر ورودی MV1 : Q0.0

شیر ورودی MV2 : Q0.1

موتور میکسر: Q0.2

شیر تخلیه MV3 : Q0.3

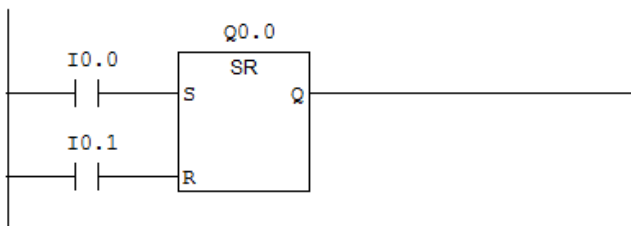
مثال ۲۶

با فشردن شاستی استارت I0.0 پمپ ۱ روشن و پس از سپری شدن ۱۰ ثانیه ، پمپ ۲ نیز روشن شود. با فشردن شاستی استپ I0.1 هر دو پمپ خاموش شوند.



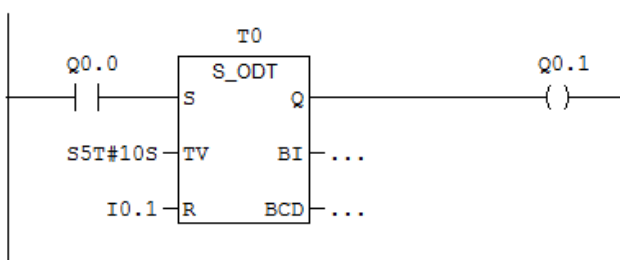
Network 1 : Title:

Comment:



Network 2 : Title:

Comment:

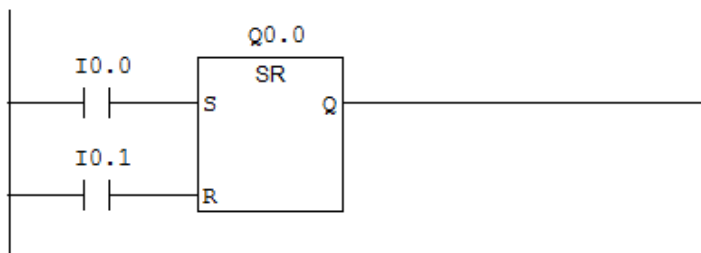


در این مثال با خاموش شدن پمپ اول، پمپ دوم نیز خاموش می شود. چون تایمر به لبه پایین رونده ورودی حساس می باشد.

در این مثال (مثال ۲۷) می خواهیم وابستگی پمپ دوم را به پمپ اول حذف کنیم. در ضمن هدف طراحی مثال ها با تایمر S-ODT می باشد.

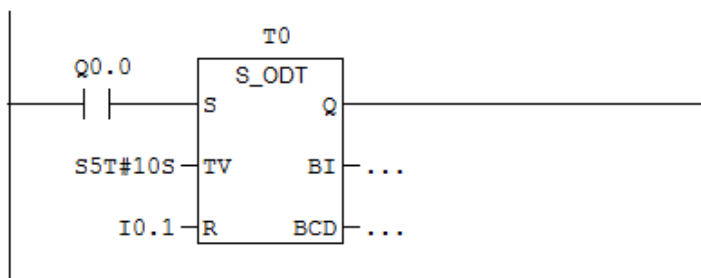
Network 1 : Title:

Comment:



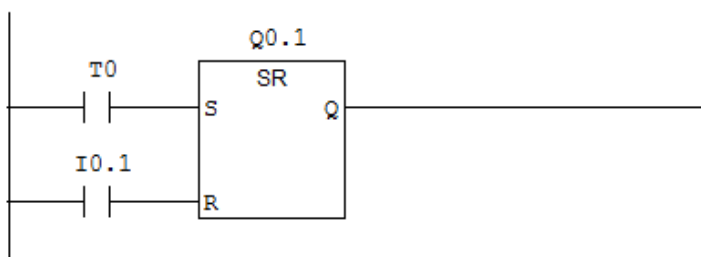
Network 2 : Title:

Comment:



Network 3 : Title:

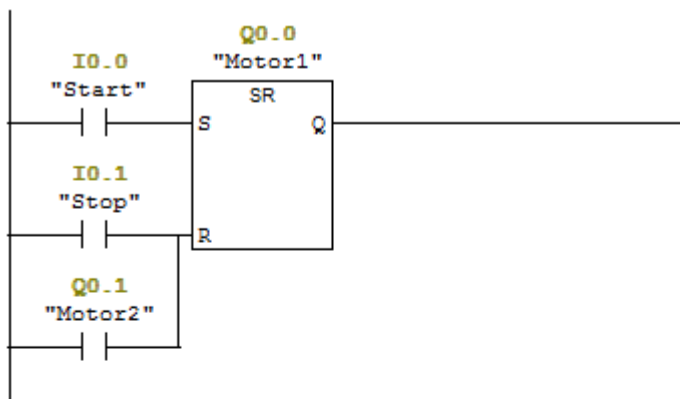
Comment:



در این مثال برای پمپ دوم از فیلیپ فلاپ جداگانه استفاده شده است.

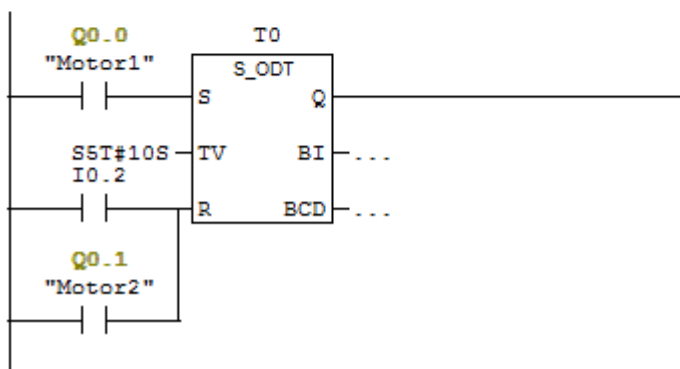
مثال ۲۸

با زدن شاستی استارت موتور اول روشن می شود. موتور دوم با 10s تاخیر زمانی بعد از موتور اول روشن می شود. موتور دوم موتور اول را خاموش می کند.



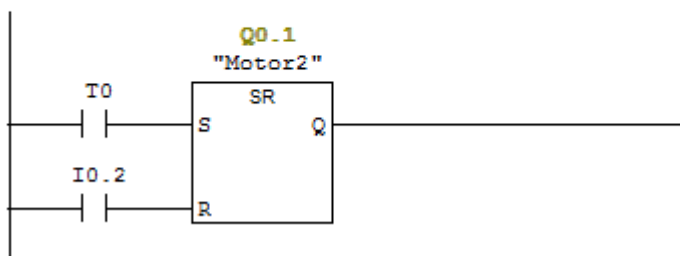
Network 2 : Title:

Comment:



Network 3: Title:

Comment:

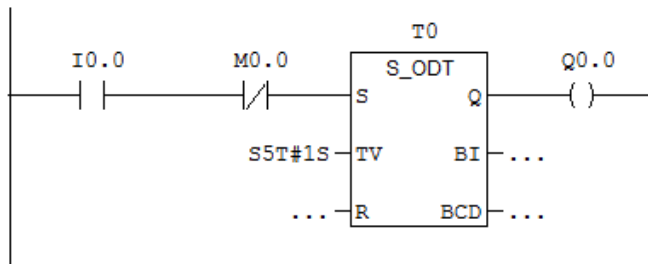


مثال ۲۹- تولید پالس

می خواهیم برنامه‌ای بنویسیم که با فعال شدن ورودی I0.0 در Q0.0 یک قطار پالس (یک ثانیه روشن و یک ثانیه خاموش) تولید گردد.

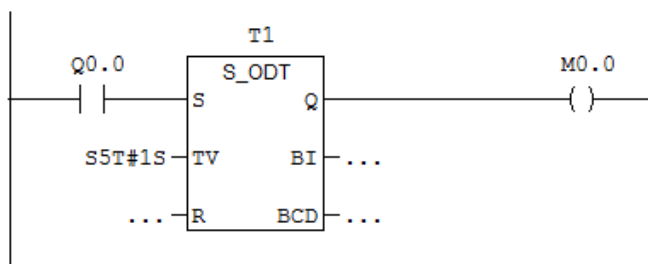
Network 1 : Title:

Comment:



Network 2 : Title:

Comment:

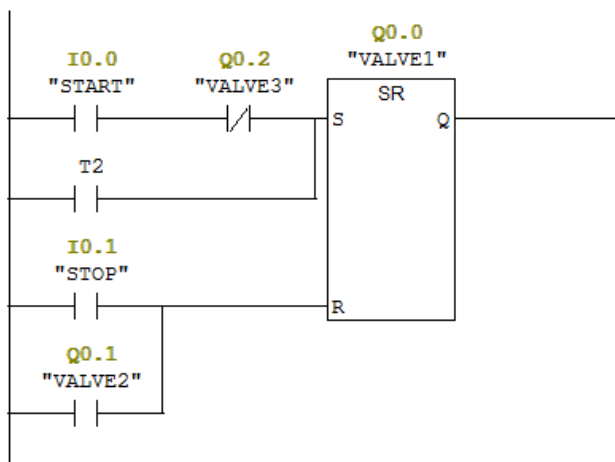


مثال ۳۰

فرض کنید در یک پروژه صنعتی از ۳ ولو در سه مسیر استفاده شده است. نحوه باز و بسته شدن ولوها بدین صورت می باشد که با فشردن شاستی استارت، ولو ۱ باز و ۱ دقیقه بعد ولو ۲ باز و ولو ۱ بسته می شود. در ادامه نیز ۱ دقیقه بعد ولو ۳ باز و ولو ۲ بسته می شود. در مرحله پایانی نیز ۱ دقیقه بعد ولو ۱ مجددا باز و ولو ۳ بسته می گردد. این روند تا زمانی که شاستی استپ فشرده نشده است ادامه می یابد. در ضمن لازم به ذکر است که اگر در وسط سیکل، شاستی استارت مجددا فشرده شد هیچ تاثیری در عملکرد مدار نداشته باشد.

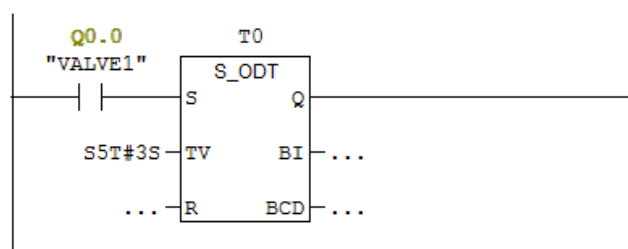
Network 1 : Title:

Comment:



Network 2 : Title:

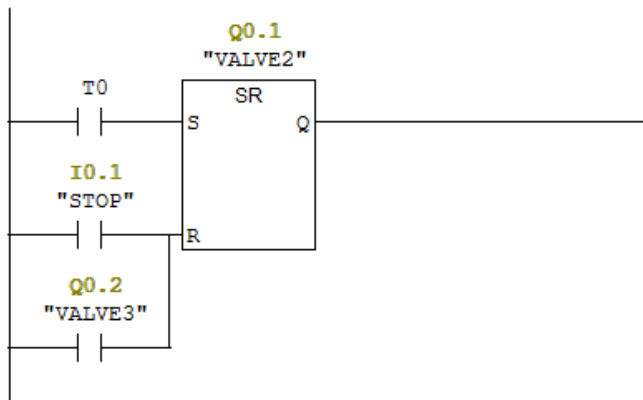
Comment:



ادامه

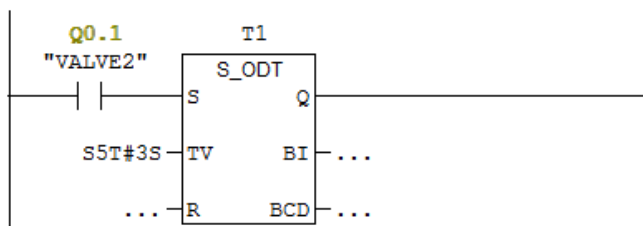
Network 3 : Title:

Comment:



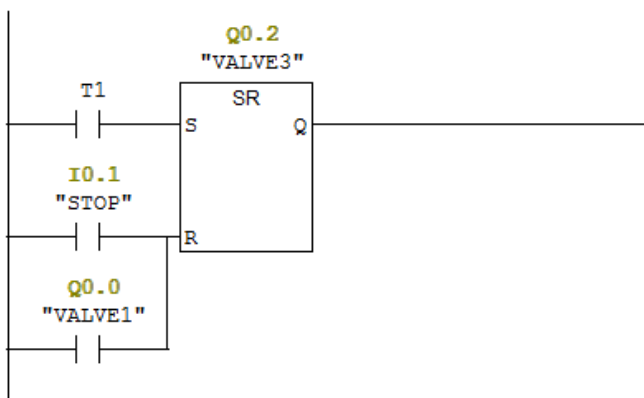
Network 4 : Title:

Comment:



Network 5 : Title:

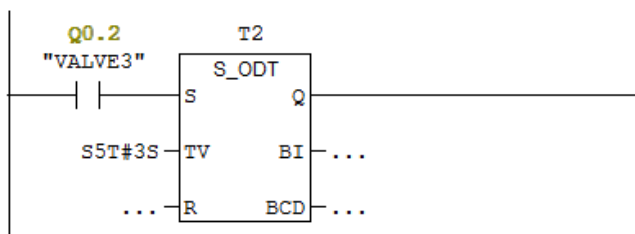
Comment:



ادامه

Network 6 : Title:

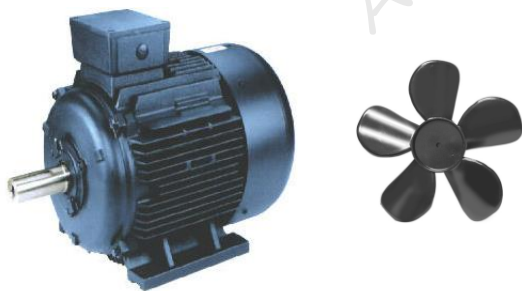
Comment:



مثال ۳۱

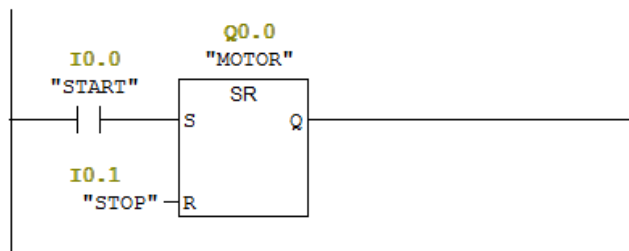
در یک پروژه صنعتی از یک موتور با توان بالا (Q0.0) در یک محیط بسته استفاده و در کنار این موتور نیز یک فن خنک کن (Q0.1) قرار داده شده است. نحوه کار بدین صورت می باشد که با فشردن استارت I0.0 هر دو مصرف کننده وارد مدار می شوند. اما زمانی که فرمان قطع موتور توسط I0.1 صادر گردید، موتور بلافاصله متوقف و فن خنک کن ۳ دقیقه بعد از خاموش شدن موتور خاموش گردد. مطلوب است نوشتن برنامه

این مثال توسط تایمر S-ODT



Network 1 : Title:

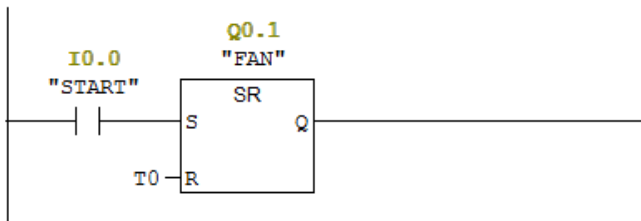
Comment:



ادامه

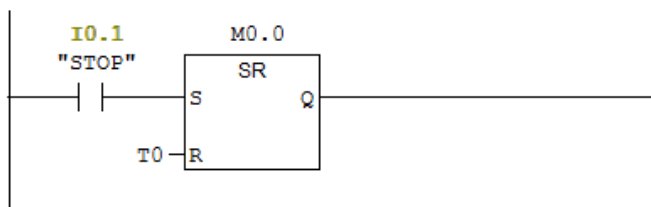
Network 2 : Title:

Comment:



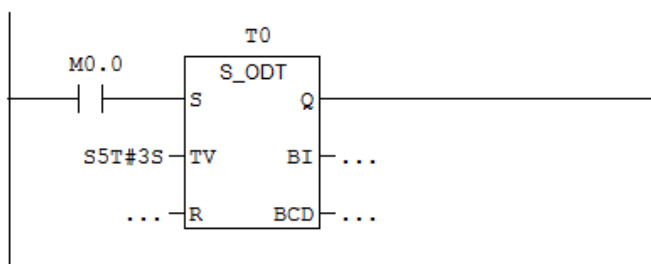
Network 3 : Title:

Comment:



Network 4 : Title:

Comment:



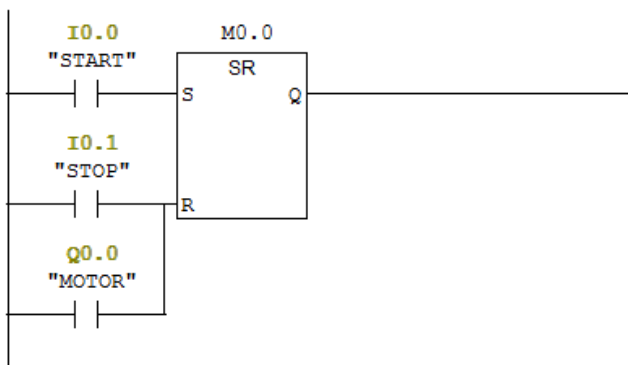
مثال ۳۲

در این مثال می خواهیم برنامه ای طراحی کنیم که با فعال کردن ورودی I0.0 ، یک دقیقه بعد خروجی Q0.0

فعال شود. تایمر S-ODT

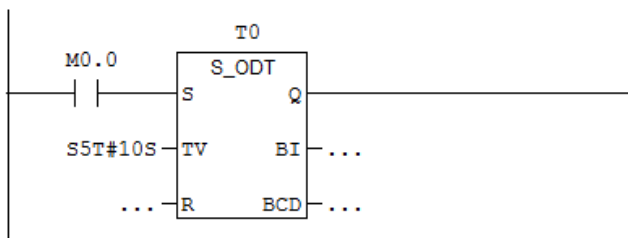
Network 1: Title:

Comment:



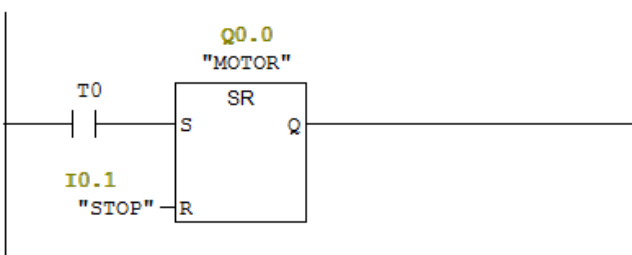
Network 2: Title:

Comment:



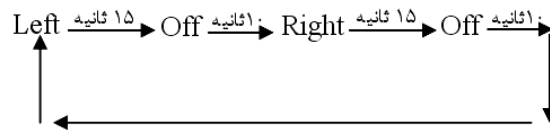
Network 3: Title:

Comment:



تمرین 1- چپگرد / راستگرد اتوماتیک

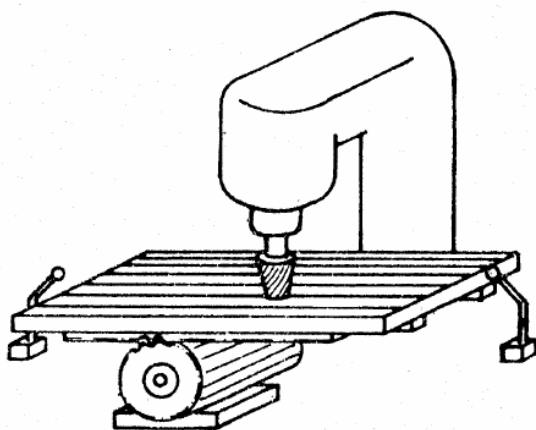
برنامه طراحی کنید که یک موتور سه فاز که به صورت چپگرد راستگرد عمل می کند به صورت زیر کنترل شود.



با توجه به شکل فوق با فرمان استارت I0.1، موتور به صورت چپگرد به مدت ۱۵ ثانیه وارد مدار می شود. پس از سپری شدن ۱۵ ثانیه، موتور به مدت ۱۰ ثانیه خاموش و در ادامه کنتاکتور مربوط به راستگرد وارد مدار می شود. این روند برای حالت راستگرد نیز ادامه پیدا می کند و در پایان سیکل نیز پس از ۱۰ ثانیه توقف مجدداً موتور به صورت چپگرد وارد مدار می شود. این سیکل تا زمانی که شاستی استپ مدار (I0.0) فعال نشود ادامه یابد.

تمرین ۲- کنترل ماشین فرز کاری

یک میز مربوط به ماشین فرز کاری را در نظر بگیرید. نحوه کار بدین صورت می باشد که با روشن شدن موتور، صفحه متحرک ماشین فرز در یک جهت حرکت می کند و در پایان مسیر با برخورد به میکروسوئیچ جهت حرکت موتور عکس می شود و با معکوس شدن جهت حرکت موتور جهت حرکت میز نیز تغییر می کند. این مرحله مجدداً در ابتدای مسیر با برخورد به میکروسوئیچ دیگر تکرار می شود و این روند تا زمانی که شاستی استپ ماشین زده نشود ادامه می یابد.



I0.0 : استپ کل

I0.1 : استارت شروع به کار

I0.2 : میکروسوئیچ ابتدای میز

I0.3 : میکروسوئیچ انتهای میز

Q0.0 : کنتاکتور مربوط به چپگرد

Q0.1 : کنتاکتور مربوط به راستگرد

*هر دو میکروسوئیچ استفاده شده در این ماشین دارای یک تیغه باز می باشند.

مثال ۳۳

در این مثال می خواهیم مثال ۳۰ را به روش STL بازنویسی کنیم. (با استفاده از تایمر S-ODT)

Network 1: Title:

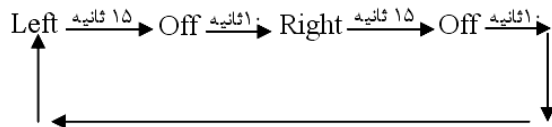
Comment:

A	I	0.0
AN	Q	0.1
AN	Q	0.2
S	Q	0.0
A	Q	0.0
L	S5T#1M	
SD	T	0
A	T	0
R	Q	0.0
S	Q	0.1
A	Q	0.1
L	S5T#1M	
SD	T	1
A	T	1
R	Q	0.1
S	Q	0.2
A	Q	0.2
L	S5T#1M	
SD	T	2
A	T	2
R	Q	0.2
S	Q	0.0
A	I	0.1
R	Q	0.0
R	Q	0.1
R	Q	0.2

مثال ۳۴

در این مثال می خواهیم با زدن شاستی استارت یک موتور به صورت زیر در دو جهت کنترل شود. (طراحی توسط

تایمر S-ODT)

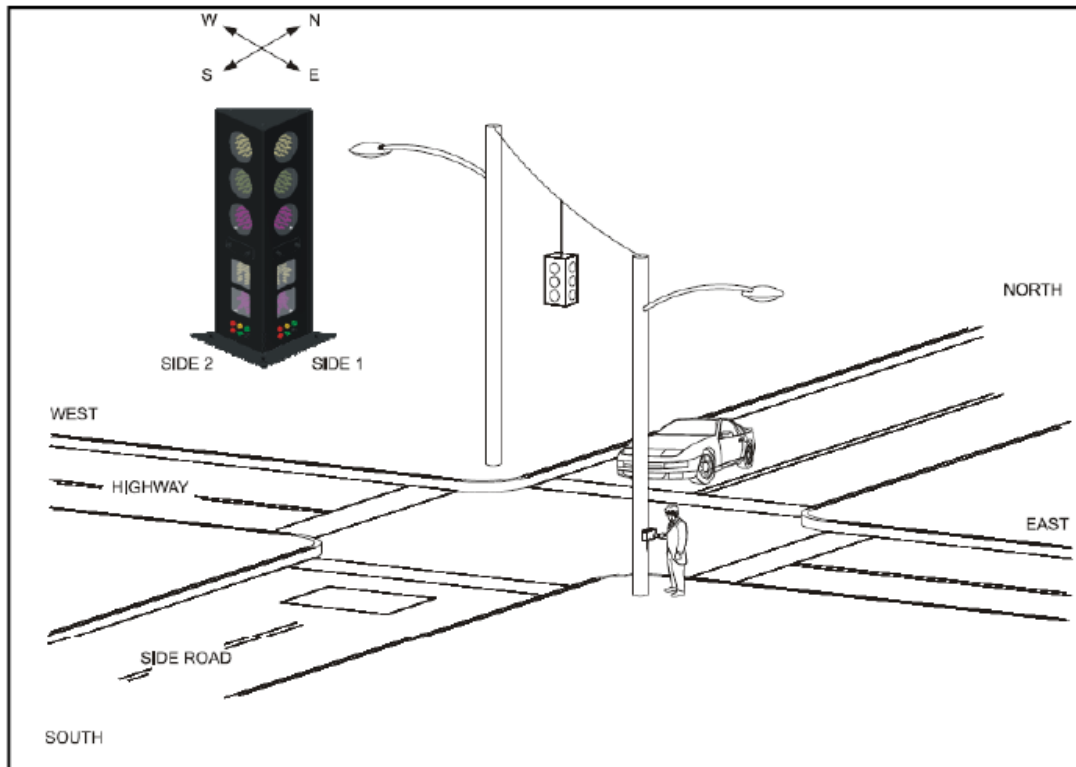


لازم به ذکر است که تا زمانی که شاستی استپ فشرده نشده است، این سیکل تکرار گردد.

A	I	0.0
AN	Q	0.1
S	Q	0.0
A	Q	0.0
L	S5T#10s	
SD	T	0
A	T	0
R	Q	0.0
S	M	0.0
A	M	0.0
L	S5T#5s	
SD	T	1
A	T	1
AN	Q	0.0
S	Q	0.1
R	M	0.0
A	Q	0.1
L	S5T#10s	
SD	T	2
A	T	2
R	Q	0.1
S	M	0.1
A	M	0.1
L	S5T#5s	
SD	T	3
A	T	3
AN	Q	0.1
S	Q	0.0
R	M	0.1
A	I	0.1
R	Q	0.0
R	Q	0.1
R	M	0.0
R	M	0.1

تمرین- در مثال فوق با اعمال شرایطی خاص، عملکرد موتور مختل می شود. مشکل را یافته و برنامه را اصلاح کنید.

تمرین: کنترل چراغ راهنمایی



کنترل چراغ راهنمایی شامل ۴ مرحله می باشد:

۱-مرحله اول: چراغ سبز اصلی به همراه قرمز فرعی

۲-مرحله دوم: چراغ زرد اصلی به همراه قرمز فرعی

۳-مرحله سوم: چراغ قرمز اصلی به همراه سبز فرعی

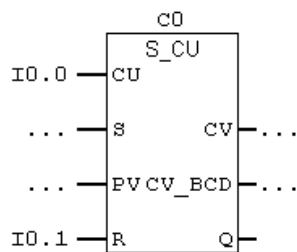
۴-مرحله چهارم: چراغ قرمز اصلی به همراه زرد فرعی

۵- بازگشت به مرحله ۱

در این تمرین جهت شروع به کار از یک کلید استارت و جهت خاموش شدن کلیه لامپ ها از یک کلید استپ استفاده می شود.

مثال ۳۵

شمارش تعداد پالس دریافتی از یک سنسور

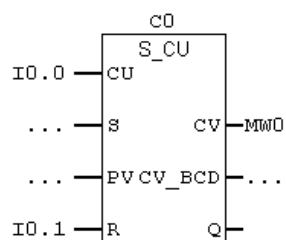


مثال 36

شمارش تعداد پالس و روشن شدن Q0.0 در صورتی که مقدار شمارنده برابر ۵۰ شود.

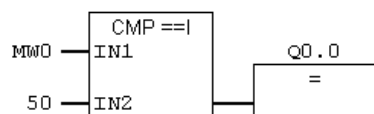
Network 1: Title:

Comment:



Network 2: Title:

Comment:

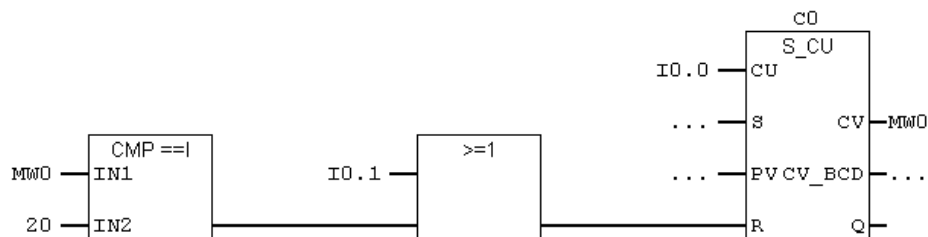


مثال ۳۷

شمارش تعداد پالس های دریافتی از یک سنسور و روشن شدن ۳ خروجی در بازه های مختلف. شمارش تا ۲۰ ادامه یافته و مجدداً شمارنده صفر شده و این سیکل تکرار می شود.

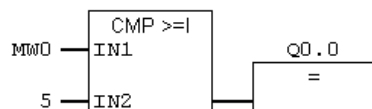
Network 1: Title:

Comment:



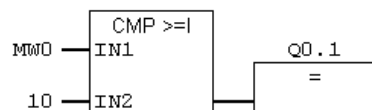
Network 2: Title:

Comment:



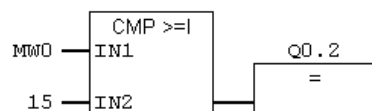
Network 3: Title:

Comment:



Network 4: Title:

Comment:



بازنویسی مثال قبل به روش STL با استفاده از دستورات SET و RESET

Network 1: Title:

Comment:

```

A      I      0.0
CU     C      0
L      C      0
L      5
==I
S      Q      0.0
L      C      0
L      10
==I
S      Q      0.1
L      C      0
L      15
==I
S      Q      0.2
L      C      0
L      20
==I
R      Q      0.0
R      Q      0.1
R      Q      0.2
R      C      0
A      I      0.1
R      C      0
R      Q      0.0
R      Q      0.1
R      Q      0.2

```

مثال ۳۸

شاستی پوش باتن به روش STL

Network 1: Title:

Comment:

```

A      I      0.0
CU     C      0
L      C      0
L      1
==I
S      Q      0.0
L      C      0
L      2
==I
R      Q      0.0
R      C      0

```

مثال ۳۹- خط بسته بندی

با فشردن شاستی PB1 کانوایر مربوط به جعبه ها شروع به حرکت می کند. وقتی جعبه به سنسور SE2 رسید، موتور کانوایر جعبه ها خاموش و موتور کانوایر مربوط به سیب ها روشن می شود. زمانی که تعداد ۱۰ سیب در کارتن قرار گرفت کانوایر سیب ها متوقف و این روند برای جعبه بعدی با روشن شدن مجدد کانوایر جعبه ها تکرار می شود. تعداد سیب ها نیز توسط سنسور SE1 شمرده می شود. شاستی PB2 نیز شاستی ریست کل می باشد.

PB1: ورودی استارت I0.0

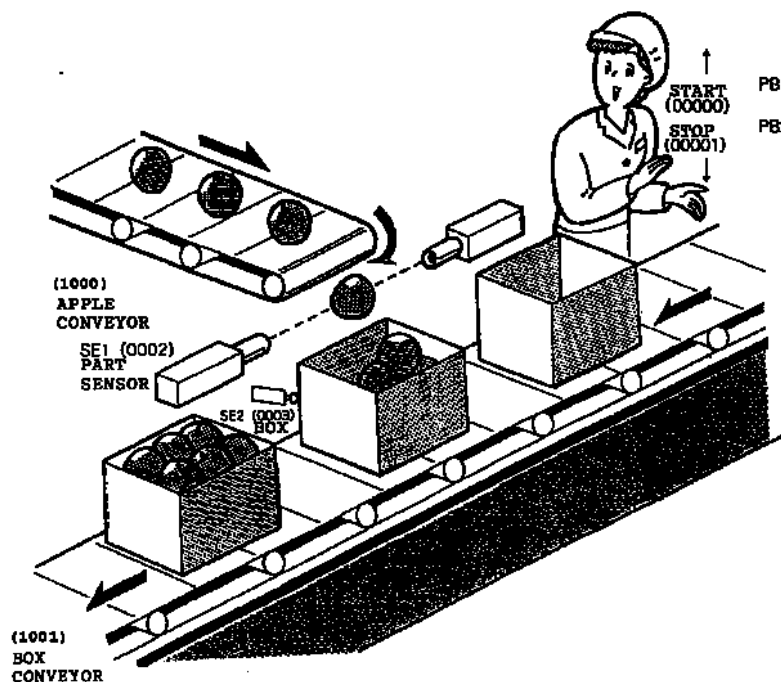
PB2: ورودی استپ I0.1

SE1: ورودی I0.2 (سنسور شمارش سیب ها)

SE2: ورودی I0.3 (سنسور تشخیص وجود جعبه ها)

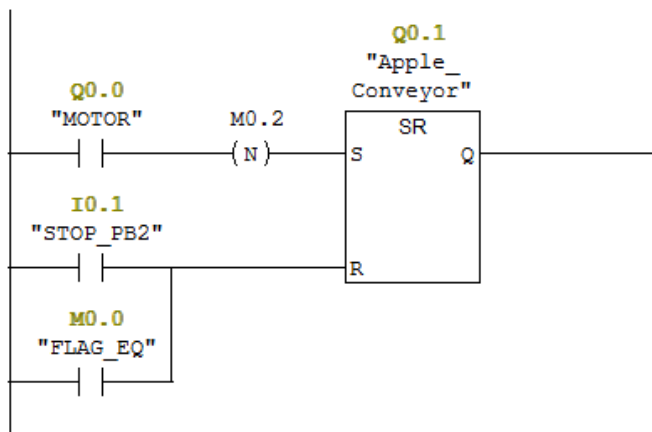
Box Conveyor: خروجی Q0.0

Apple Conveyor: خروجی Q0.1



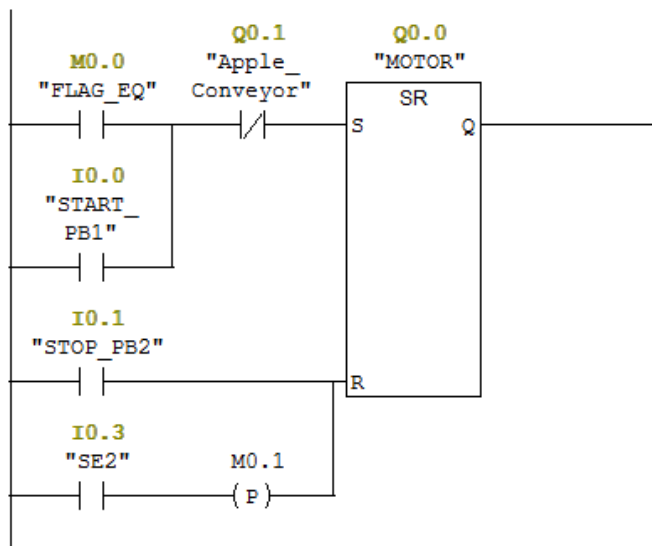
Network 2 : Title:

Comment:



Network 1: Title:

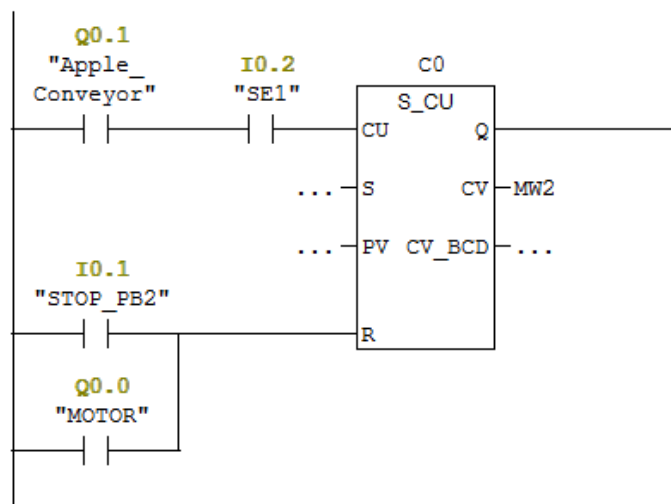
Comment:



ادامه

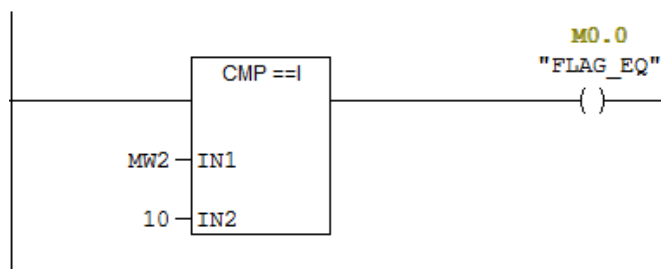
Network 3 : Title:

Comment:



Network 4 : Title:

Comment:



مثال ۴۰

در این مثال می خواهیم تعداد پالس تولید شده از یک سنسور توسط یک شمارنده شمارش و ۳ خروجی PLC در بازه های عددی مشخص شده در قسمت زیر روشن شوند. با رسیدن مقدار شمارنده به ۲۰، هر سه خروجی PLC خاموش و این روند مجدداً با ریست شدن شمارنده تکرار شود.

* با رسیدن مقدار شمارنده به عدد ۵، خروجی Q0.0 روشن شود.

* با رسیدن مقدار شمارنده به عدد ۱۰، خروجی Q0.1 نیز روشن شود.

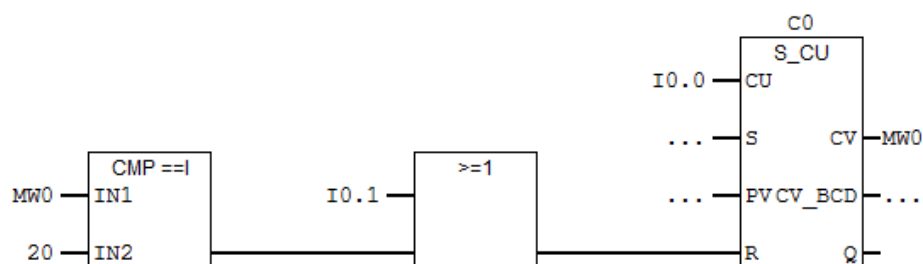
* با رسیدن مقدار شمارنده به عدد ۱۵، خروجی Q0.2 نیز روشن شود.

* با رسیدن مقدار شمارنده به عدد ۲۰، هر سه خروجی خاموش و شمارنده نیز ریست شود.



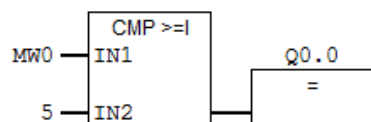
Network 1: Title:

Comment:



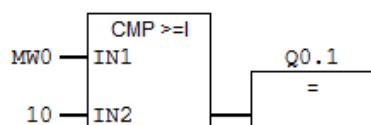
Network 2: Title:

Comment:



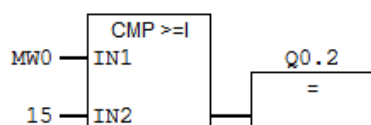
Network 3: Title:

Comment:



Network 4: Title:

Comment:



مثال ۴۱

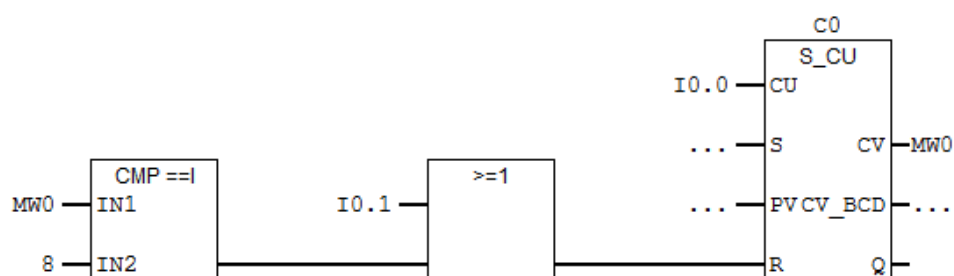
در این مثال قصد داریم برنامه ای طراحی کنیم که توسط یک شاستی بتوان ۴ پمپ را به صورت زیر کنترل کرد. با هر بار فعال شدن شاستی یک پمپ وارد مدار شود. در این صورت پس از ۴ بار فشردن شاستی I0.0 هر ۴ پمپ روشن شوند. اگر شاستی برای بار پنجم فشار داده شد، به همان ترتیب که پمپ ها روشن شده بودند خاموش شوند. به عبارت دیگر با هر دفعه فشردن شاستی از مرحله ۵، پمپ ها خاموش گردند. یعنی در مرحله ۵ پمپ ۴ خاموش و در مرحله ۶ پمپ ۳ خاموش و این روند تا پمپ ۱ ادامه پیدا کند. در ضمن برنامه باید به گونه ای طراحی شود که کل این سیکل مجددا قابل تکرار باشد.



برنامه

Network 1: Title:

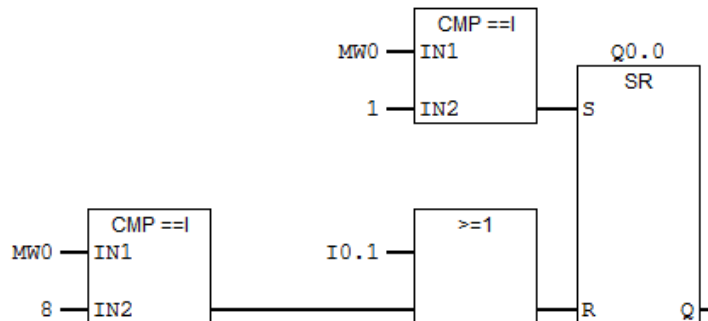
Comment:



ادامه

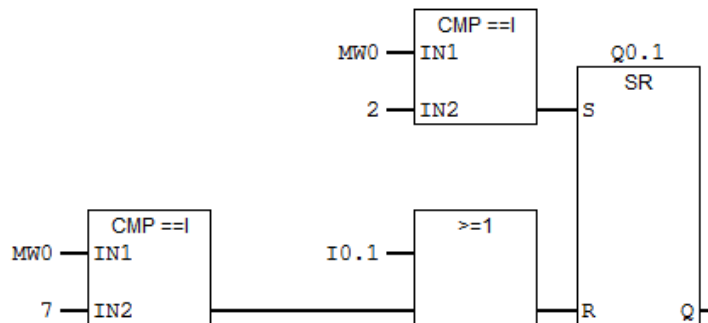
Network 2 : Title:

Comment:



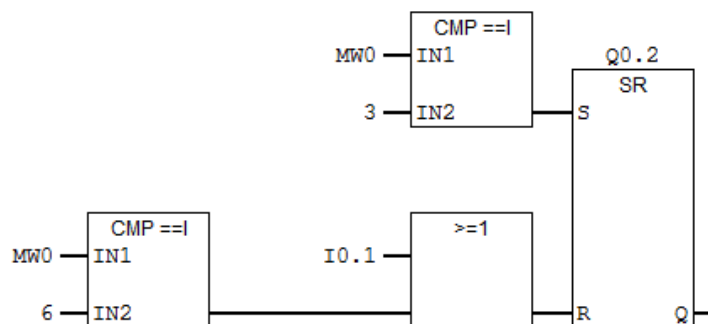
Network 3 : Title:

Comment:



Network 4 : Title:

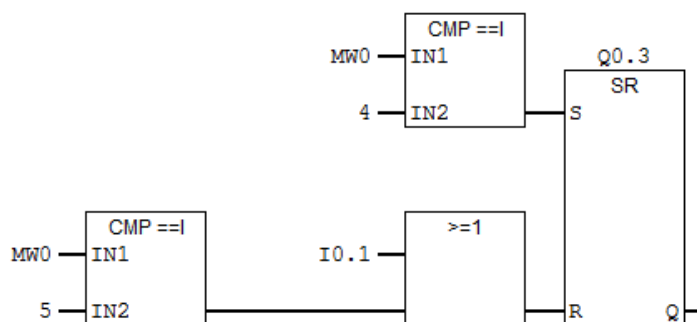
Comment:



ادامه

Network 5 : Title:

Comment:

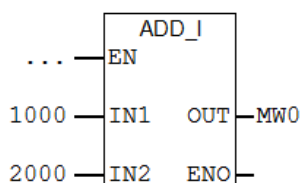


مثال ۴۲

جمع دو مقدار 1000 و 2000 و ذخیره آن در MW0

Network 1: Title:

Comment:

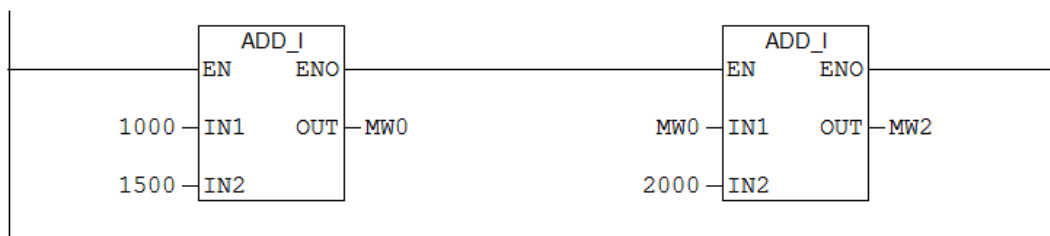


مثال ۴۳

جمع ۳ مقدار 1000 ، 1500 و 2000 و ذخیره آن در MW2

Network 1: Title:

Comment:



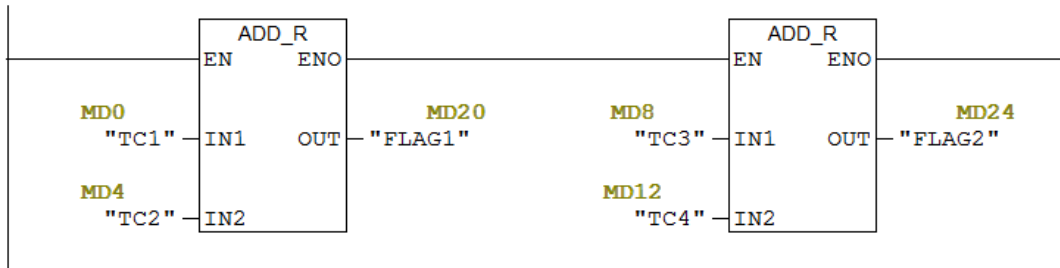
مثال ۴۴

فرض کنیم در ۴ حافظه ۳۲ بیتی ، مقادیر مربوط به ۴ سنسور دما قرار دارند. می خواهیم برنامه ای بنویسیم که میانگین این ۴ دما محاسبه شود.

برنامه

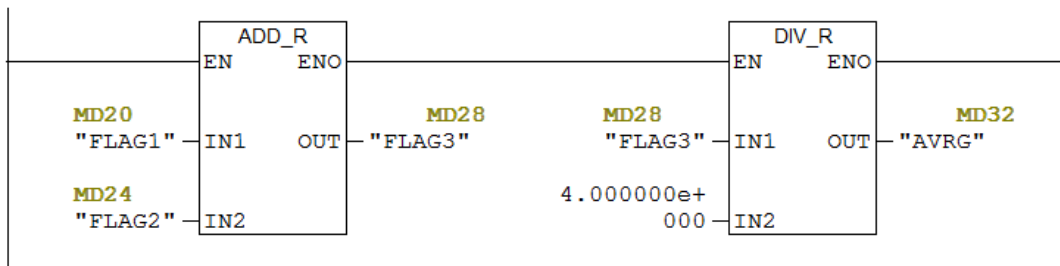
Network 1: Title:

Comment:



Network 2: Title:

Comment:



برای تست این برنامه به راحتی می توان در محیط شبیه ساز به فضاهای مربوط به TC1 تا TC4، مقادیر مورد نظر را به عنوان دما اعمال نمود و نتیجه نهایی را در MD32 مشاهده کرد.

بازنویسی مثال قبل توسط روش STL

Network 1: Title:

Comment:

L	"TC1"	MD0
L	"TC2"	MD4
+R		
T	"FLAG1"	MD20
L	"TC3"	MD8
L	"TC4"	MD12
+R		
T	"FLAG2"	MD24
L	"FLAG1"	MD20
L	"FLAG2"	MD24
+R		
T	"FLAG3"	MD28
L	"FLAG3"	MD28
L	4.000000e+000	
/R		
T	"AVRG"	MD32

برنامه شکل زیر را تحلیل و بحث تعداد آکومولاتور و موضوع جانشینی در آکومولاتور بررسی شود.

Network 1: Title:

Comment:

L	1000	
L	1500	
L	2000	
+I		
T	MW	0

با اجرای برنامه فوق ، مقدار ۳۵۰۰ در MW0 قرار می گیرد.

دلیل؟

مثال ۴۵- کنترل ۳ ولو در بازه های مختلف دمایی

در یک پروسه صنعتی ۴ ترموکوپل جهت اندازه گیری دما در ۴ نقطه یک کوره نصب شده اند. نحوه کار بدین صورت می باشد که در برنامه کنترلی ابتدا می بایست میانگین این چهار دما محاسبه و با توجه به بازه های مختلف دمایی، شیرها باز یا بسته شوند.

در این مثال فرض می گیریم:

دمای نقطه ۱: MD0

دمای نقطه ۲: MD4

دمای نقطه ۳: MD8

دمای نقطه ۴: MD12

میانگین دما (AVG): MD32

$0.0 < AVG < 85.0$ Valve1=Open

$100.0 < AVG < 150.0$ Valve2=Open

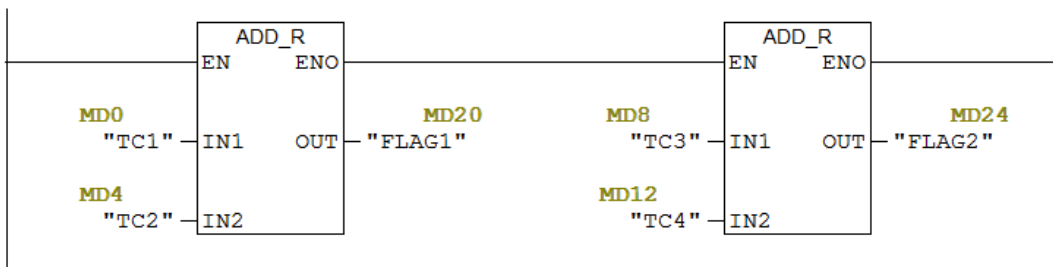
$165.0 < AVG < 250.0$ Valve3=Open

نکته: نحوه پردازش سیگنال های آنالوگ و برنامه نویسی آن در فصل های بعدی به طور کامل مورد بحث قرار خواهد گرفت. در این تمرین هدف آشنایی هر چه بیشتر با دستورات ریاضی می باشد.

برنامه

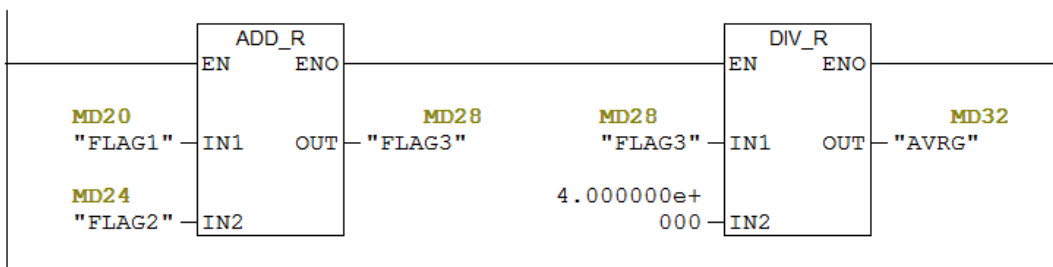
Network 1 : Title:

Comment:



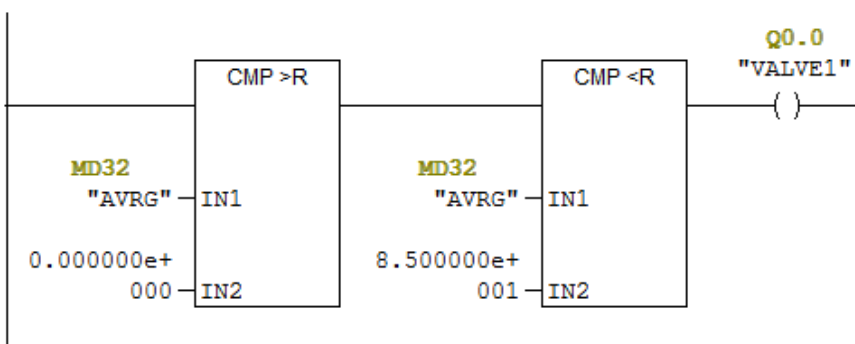
Network 2 : Title:

Comment:



Network 3 : Title:

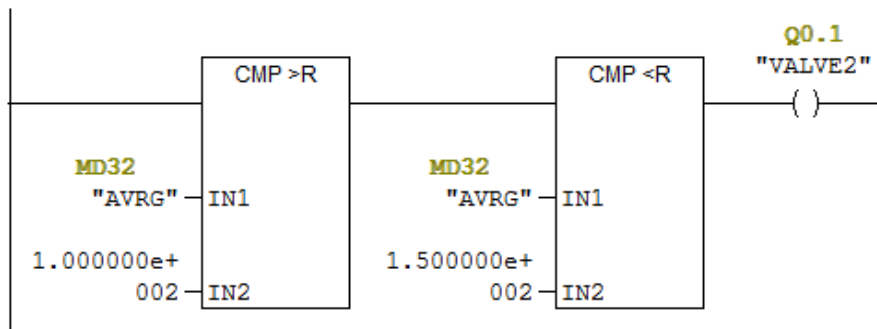
Comment:



ادامه

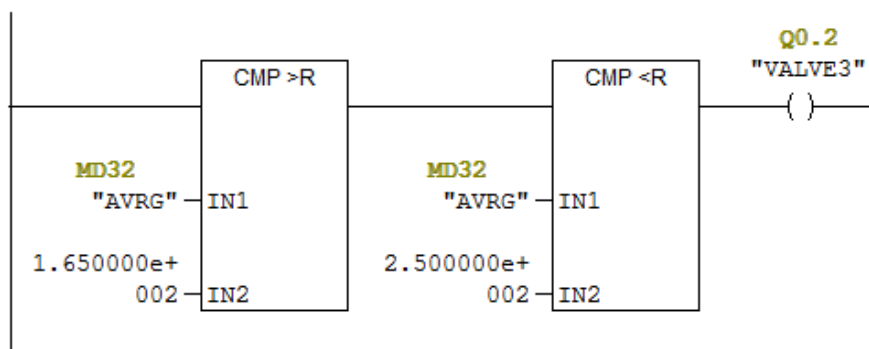
Network 4 : Title:

Comment:



Network 5 : Title:

Comment:

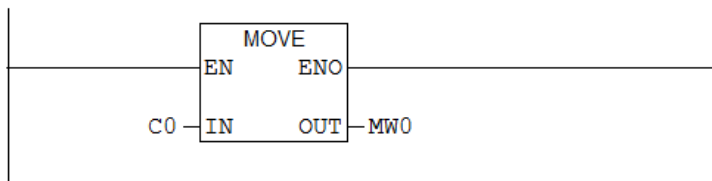


مثال ۴۶

با اجرای برنامه این مثال همیشه و بدون در نظر گرفتن شرطی، مقدار شمارنده C1 به داخل حافظه MW0 منتقل می شود.

Network 1: Title:

Comment:

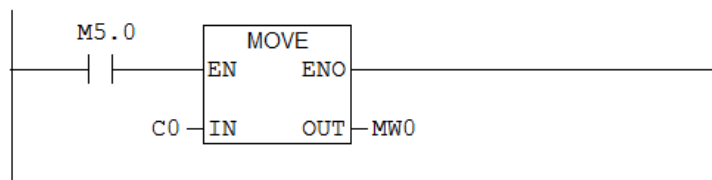


مثال ۴۷

در برنامه این مثال جهت انتقال مقدار شمارنده C1 به MW0 حتما می بایست بیت M5.0 فعال باشد. در غیر این صورت عملیات انتقال انجام نمی شود.

Network 1: Title:

Comment:

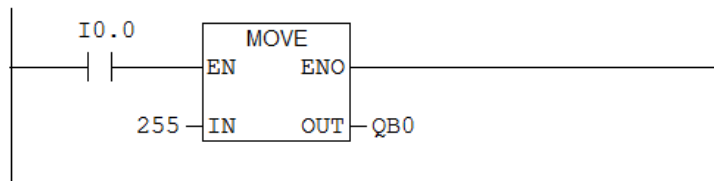


مثال ۴۸

در این مثال با فعال شدن ورودی I0.0 ، بایت صفر خروجی یعنی QB0 روشن و با غیرفعال شدن آن نیز تمامی بیت های QB0 خاموش می شوند.

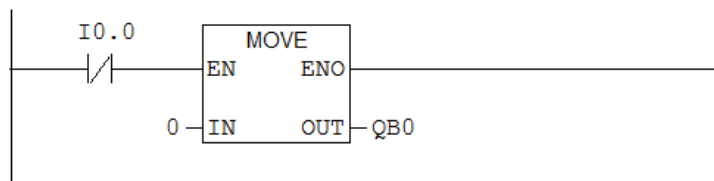
Network 1: Title:

Comment:



Network 2: Title:

Comment:



در این مثال با فعال شدن I0.0 عدد 255 به QB0 منتقل و با ارسال این عدد تمامی بیت های QB0 روشن و همچنین با غیرفعال شدن I0.0 نیز عدد صفر جهت خاموش کردن کل بایت QB0 منتقل می شود.

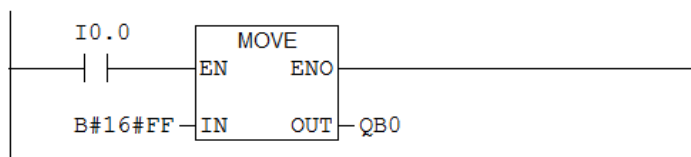
نکته

اعداد در مبنای ۱۶ (Hex)

در S7 جهت نمایش و بارگذاری اعداد در مبنای ۱۶ از فرمت زیر استفاده می شود.

Network 1: Title:

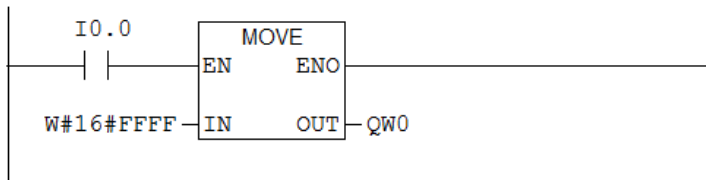
Comment:



اگر مقصد ۱۶ بیتی باشد، از حرف W استفاده می شود.

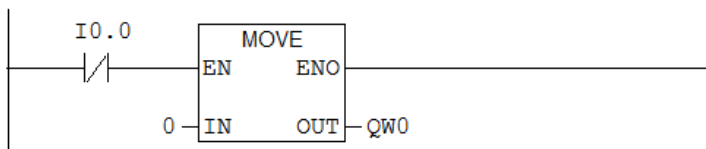
Network 1: Title:

Comment:



Network 2: Title:

Comment:



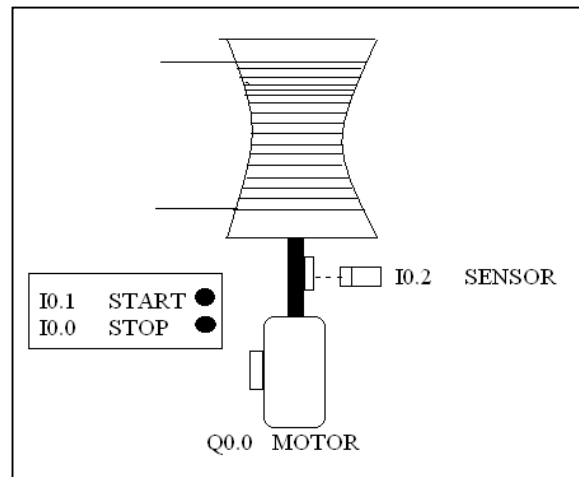
اعداد در مبنای ۲ (Binary)

جهت بارگذاری اعداد در مبنای ۲ از فرمت زیر استفاده می شود. X ها شامل 0 و 1 باینری می باشند.

2#XXXXXXXX

مثال ۴۹

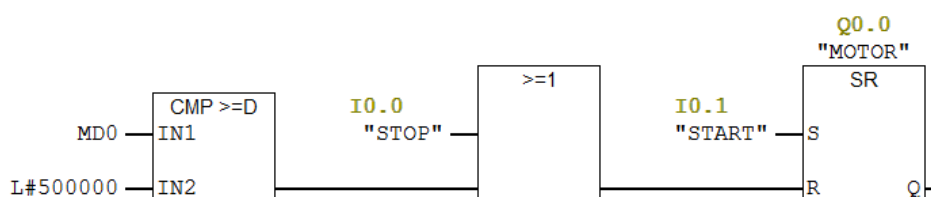
شکل زیر را در نظر بگیرید. طرز کار مدار بدین صورت می باشد که با فشردن کلید استارت موتور با سرعت کمی شروع به حرکت می نماید و عمل پیچیدن کابل شروع می شود. سپس توسط یک سنسور دور موتور شمرده می شود. می خواهیم زمانی که تعداد دور کابل به ۵۰۰۰۰۰ دور رسید، موتور قطع و شمارنده نیز ریست شود تا مدار برای قرقه بعدی آماده گردد.



برنامه این مثال را می خواهیم با استفاده از دستورات ریاضی طراحی کنیم. همانطور که در بحث های قبلی بیان شد، توسط دستور جمع در محدوده ۳۲ بیتی صحیح می توان یک شمارنده ۳۲ بیتی برای شمارش استفاده نمود.

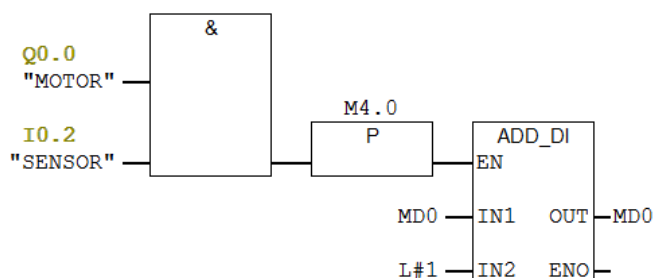
Network 1 : Title:

Comment:



Network 2 : Title:

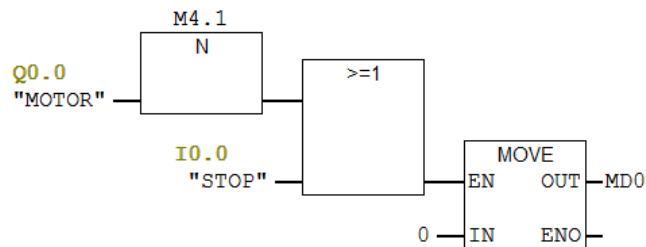
Comment:



ادامه

Network 3: Title:

Comment:



مثال ۵۰

نوشتن لاجیک دستور MOVE شرطی به روش STL

Network 1: Title:

Comment:

```

A      I      0.0
L      255
T      QB      0

```

برنامه فوق اشتباه می باشد. در حالت عادی یک بیت نمی تواند به صورت فوق، شرط انتقال یک مقدار باشد. برای اصلاح برنامه فوق می بایست از دستورات پرش استفاده نمود.

Comment:

```

      A      I      0.0
      JC      M000
      BEU
M000: L      255
      T      QB      0

```

انواع پرش منطقی

- JC Jump if RLO = 1
- JCN Jump if RLO = 0
- JU Jump Unconditional
- JL Jump to Labels

پرش های ریاضی

- JZ Jump if Zero
- JN Jump if Not Zero
- JP Jump if Plus
- JM Jump if Minus
- JPZ Jump if Plus or Zero
- JMZ Jump if Minus or Zero

مثال ۵۱

در این مثال با فعال شدن ورودی I0.0 ، بایت صفر خروجی یعنی QB0 روشن و با غیرفعال شدن آن نیز تمامی بیت های QB0 خاموش می شوند.

Network 1: Title:

Comment:

```

      A      I      0.0
      JC      M000
      L      0
      T      QB      0
      BEU
M000: L      255
      T      QB      0

```

نکته ای که در این برنامه می بایست به آن دقت شود، استفاده از دستور BEU می باشد. استفاده از این دستور باعث اتمام برنامه در خط جاری (BEU) می شود. استفاده از این دستور در برنامه های صنعتی پیشنهاد نمی شود. با تغییر تکنیک برنامه نویسی، می توانیم جایگزین بهتری برای دستور BEU پیدا کنیم.

مثال ۵۲

در این مثال توسط کلید I0.0 می توان منطق بین دو شاستی I0.1 و I0.2 را تعیین کرد. در این مثال مجدد از دستور BEU استفاده شده است.

```

A      I      0.0
JC     M000
O      I      0.1
O      I      0.2
=      Q      0.0
BEU
A      I      0.1
A      I      0.2
=      Q      0.0
BE

```

مثال ۵۳

این مثال بدون استفاده از دستور BEU با استفاده از دستورات JC ، JCN و JU بازنویسی شده است. هر دو برنامه ارائه شده دارای عملکرد یکسان می باشند. در این مثال اگر بعد از دستور NOP 0 برنامه ای وجود داشته باشد، مستقل از برنامه این مثال، پردازش می شود. دستور NOP 0 دستور بدون کار می باشد و صرفاً جهت ادامه پردازش در این مثال استفاده شده است.

A	I	0.0	A	I	0.0
JCN	M000		JC	M000	
A	I	0.1	O	I	0.1
A	I	0.2	O	I	0.2
=	Q	0.0	=	Q	0.0
JU	M001		JU	M001	
M000: O	I	0.1	M000: A	I	0.1
O	I	0.2	A	I	0.2
=	Q	0.0	=	Q	0.0
M001: NOP	0		M001: NOP	0	

دستور JCN زمانی پرش را انجام می دهد که بیت RLO در وضعیت 0 باشد.

مثال ۵۳

در این مثال زمانی که مقدار عملیات تفریق منفی شود، کاراکتر E در MB10 قرار می گیرد.

```

L      MW      0
L      MW      2
-I
JM     M000
JU     M001
M000: L      'E'
      T      MB      10
M001: NOP     0

```

مثال ۵۴

در این مثال منفی شدن مقدار تفریق، باعث روشن شدن Q0.0 می شود. در غیر اینصورت خروجی Q0.0 خاموش می باشد.

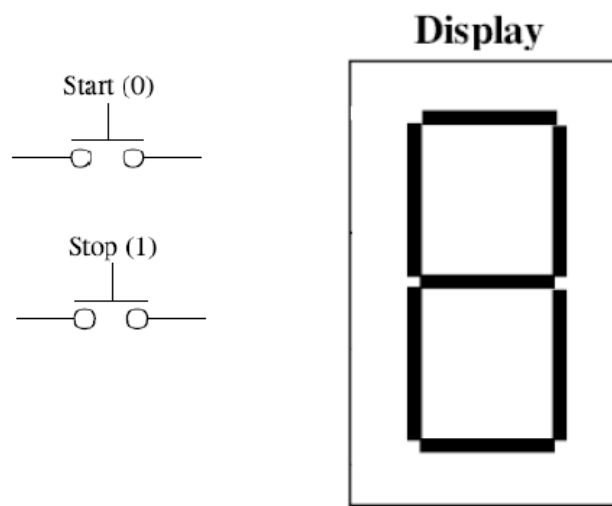
```

L      MW      0
L      MW      2
-I
JM     M000
SET
R      Q      0.0
JU     M001
M000: SET
      S      Q      0.0
M001: NOP     0

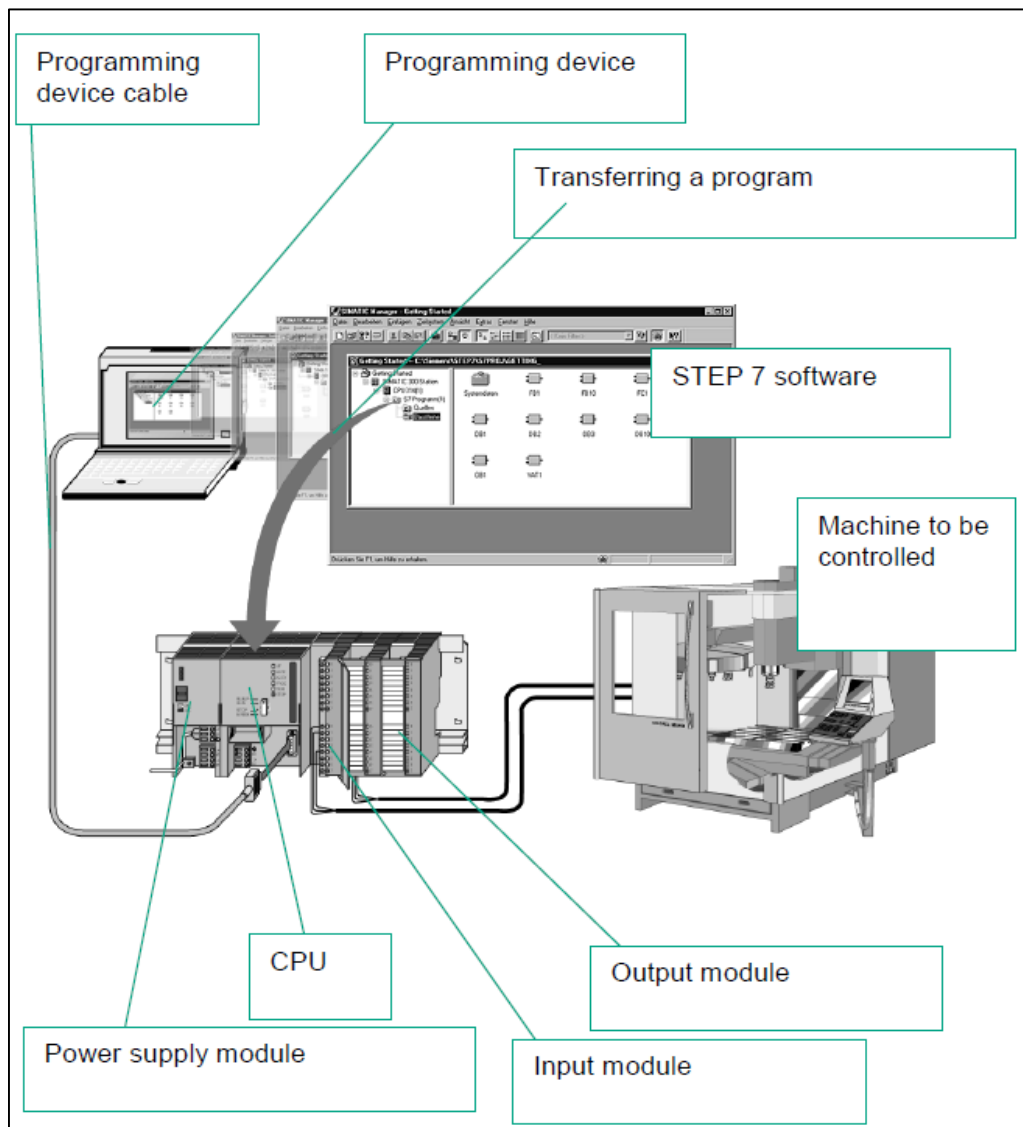
```

تمرین

مطلوب است طراحی برنامه مربوط به یک سون سگمنت که با فشردن شاستی استارت از عدد 0 شروع به شمارش کرده و هر یک ثانیه یک واحد به آن افزوده شود. با رسیدن به عدد 9 مجدداً صفر شده و این روند تا فشردن شاستی استپ تکرار شود. در ضمن جهت کنترل سون سگمنت از یک بایت خروجی PLC استفاده شده است. (طراحی برنامه به روش STL)



فصل چهارم: آشنایی با امکانات نرم افزار SIMATIC MANAGER



۱- وضعیت Monitor

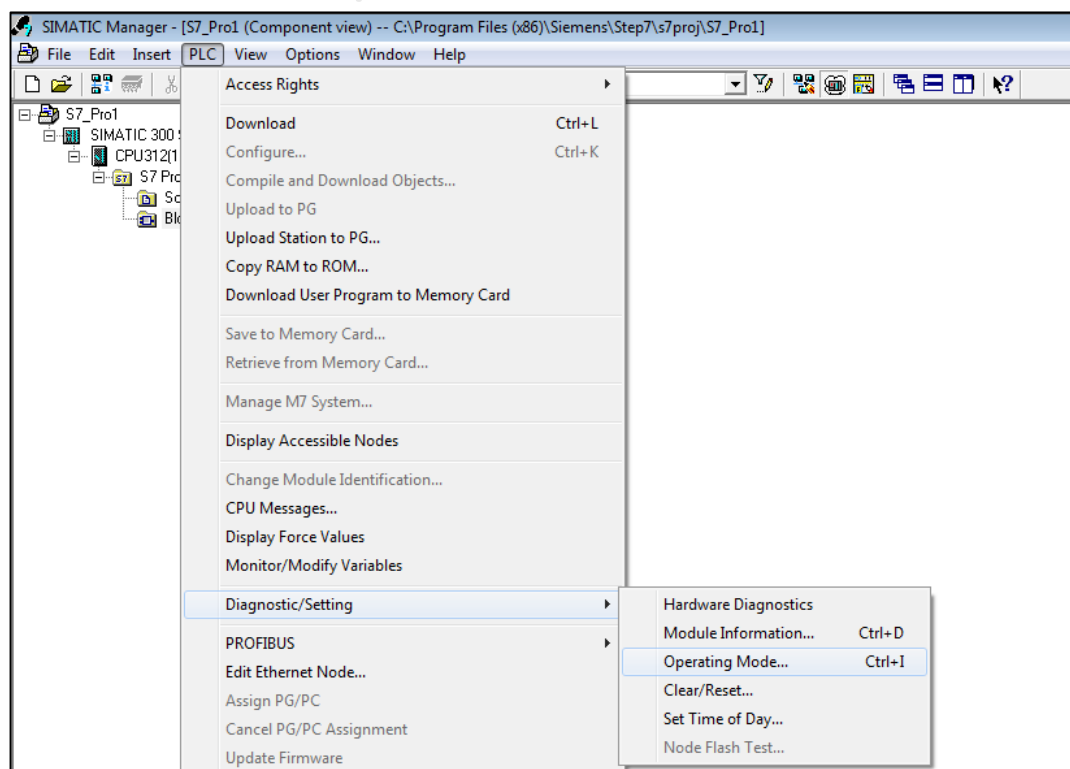
اکثر نرم افزارهای PLC دارای ابزار Monitor یا Status می باشند. ابزار مانیتور یکی از قابلیت های مهم و کاربردی در یک نرم افزار می باشد. با استفاده از این ابزار می توان وضعیت I/O ها و همچنین سایر دستورات و توابع کنترلی را در محیط نرم افزار مشاهده نمود.



آیکون مربوط به فعال و یا غیر فعال کردن وضعیت مانیتور

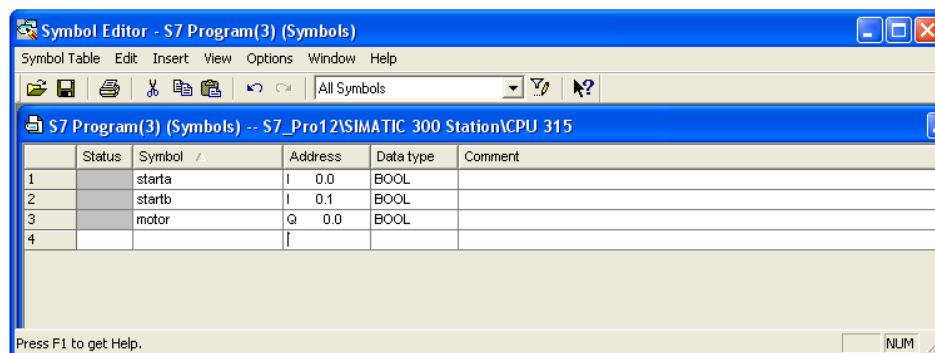
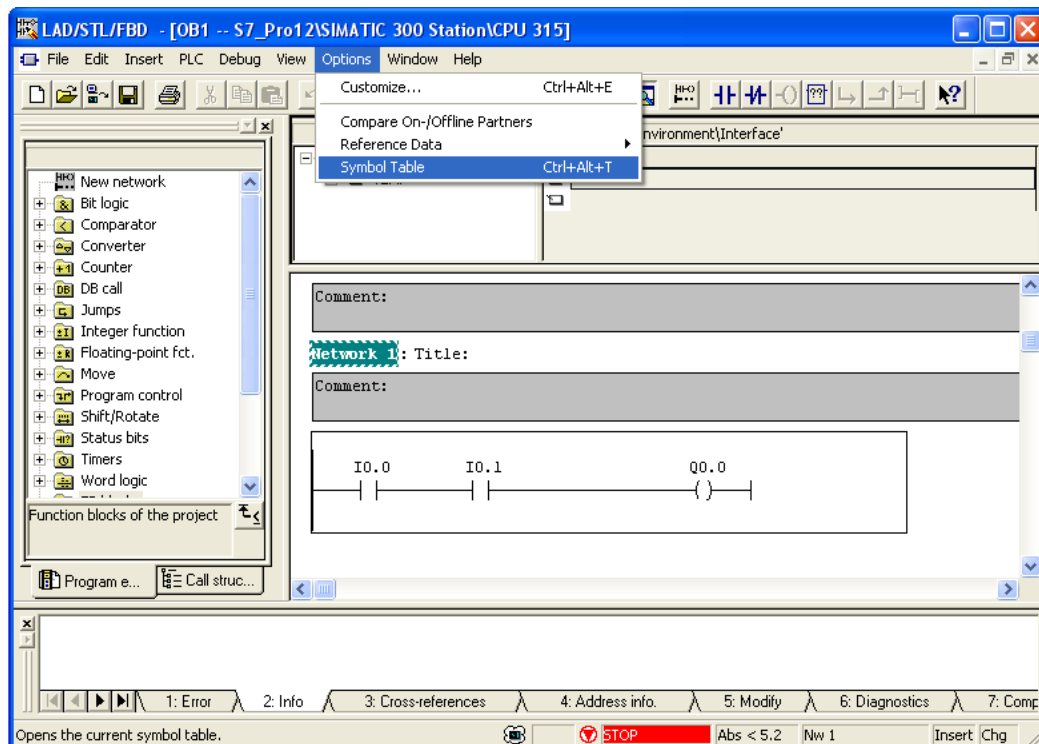
۲- نحوه Run و Stop کردن CPU

در نرم افزار مورد بحث نیز امکان تغییر مد کاری CPU وجود دارد. شرط اینکه بتوان مد کاری CPU را از طریق PC تغییر داد این است که PLC در حالت سخت افزاری در وضعیت Run باشد، یعنی کلید را در مد Run قرار داده باشیم.



۳- جدول سمبل ها Symbolic Table

توسط این جدول می توان بر روی آدرس های برنامه برچسب های دلخواه قرار داد.



بعد از تکمیل جدول فوق، گزینه SAVE را انتخاب تا برچسب های در نظر گرفته شده به برنامه اعمال شوند.

۴- کار با جدول VAT

جدول VAT یکی از قابلیت‌های مهم در نرم افزار مورد بحث می باشد. در قسمت زیر به برخی از قابلیت‌های این جدول اشاره شده است:

۱- مانیتور کردن متغیرهای I/O/M/T/C/DB

۲- انجام عملیات Force

۳- اعمال مقادیر جدید به متغیرها (Modify)

در این جدول تمامی متغیرها می توانند تعریف شوند.

منظور از متغیرها موارد زیر می باشند:

* ورودی

* خروجی

* فضاهای حافظه

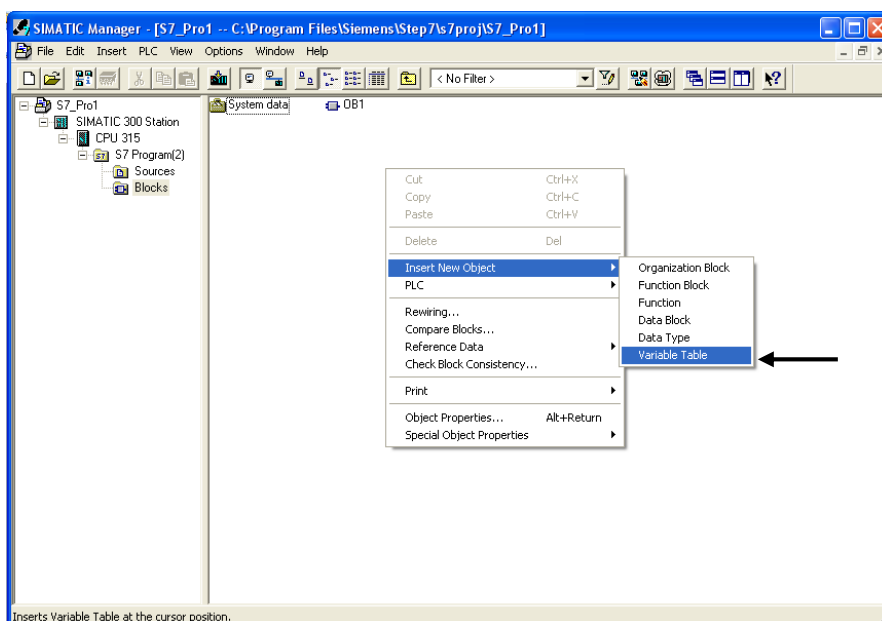
* شمارنده ها

* تایمرها

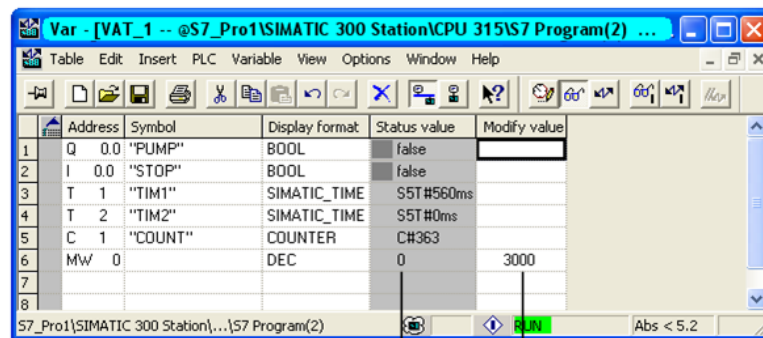
* ورودی و خروجی های آنالوگ

* فضاهای بلوک های اطلاعاتی DB

* و...



این جدول دارای یک ستون دیگر با نام Modify Value می باشد که توسط این قسمت می توان مقادیر جدیدی که به عنوان Set Point می باشند را به PLC ارسال نمود. فرض کنید عدد موجود در حافظه MW0 به عنوان S.P (نقطه مطلوب) یک پروسه می باشد. این عدد متغیر بوده و این قابلیت را دارد که محتویاتش توسط نرم افزار و بدون STOP کردن CPU توسط کاربر تغییر کند. در این صورت می توان از این جدول استفاده نمود. جهت انجام این کار ابتدا آدرس مورد نظر را در ستون های جدول وارد و عدد مورد نظر را نیز در قسمت Modify Value وارد کنید.



مقدار جدید مقدار جاری

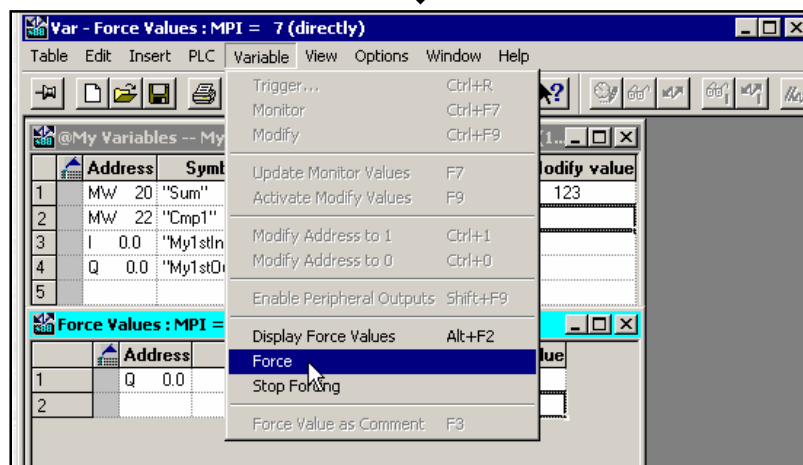
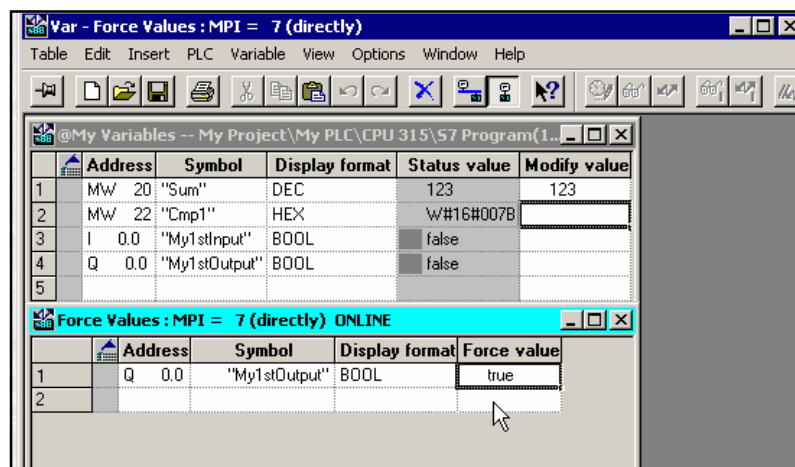
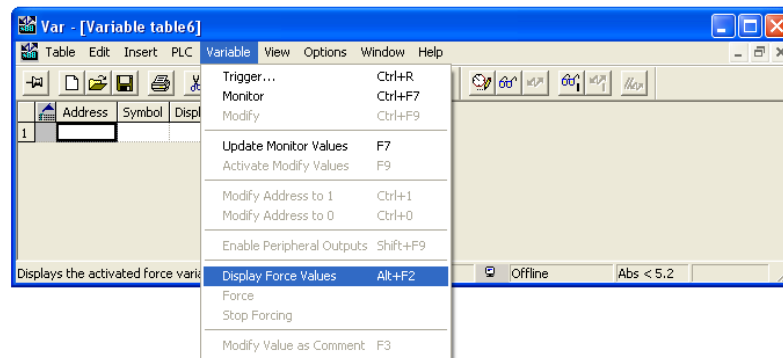
بعد از اینکه عدد مورد نظر را وارد نمودید، نوبت به ارسال این عدد به فضای مورد نظر در CPU می باشد. جهت انجام این کار کافیست در حالت مانیتور بر روی آیکون شکل زیر کلیک کنید.



آیکون Modify Value

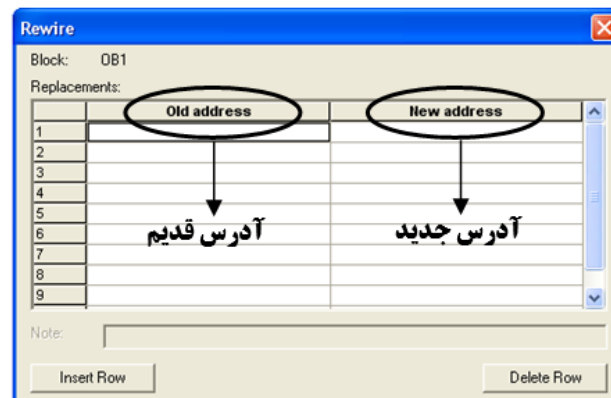
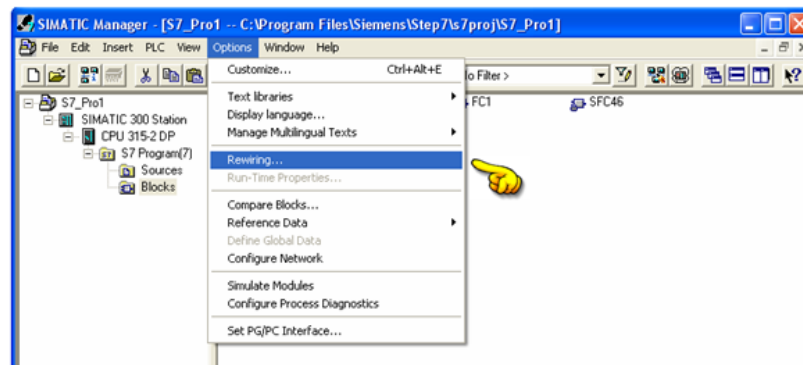
۵- نحوه Force کردن متغیرها

ابتدا لازم است در مورد Force توضیح مختصری داده شود. حالت Force یکی از امکانات کاربردی در اکثر نرم افزارهای اتوماسیون صنعتی می باشد. توسط این قابلیت کاربران متخصص می توانند بر روی I/O های PLC، مخصوصا خروجی های آن از طریق نرم افزار تاثیر مناسب را بگذارند. منظور از تاثیر همان فعال و یا غیر فعال کردن آنها می باشد.



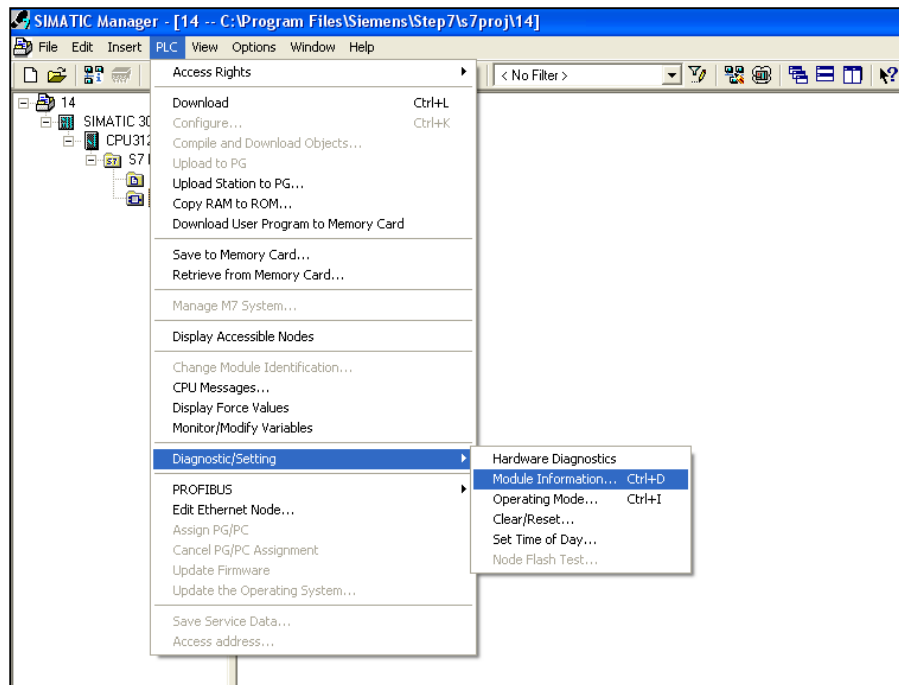
۶- آشنایی با قابلیت Rewiring

توسط این قابلیت این امکان برای کاربر وجود دارد که به راحتی و در کوتاه ترین زمان بتواند در کل پروژه خود یک یا چند آدرس را به آدرس جدید تغییر دهد.



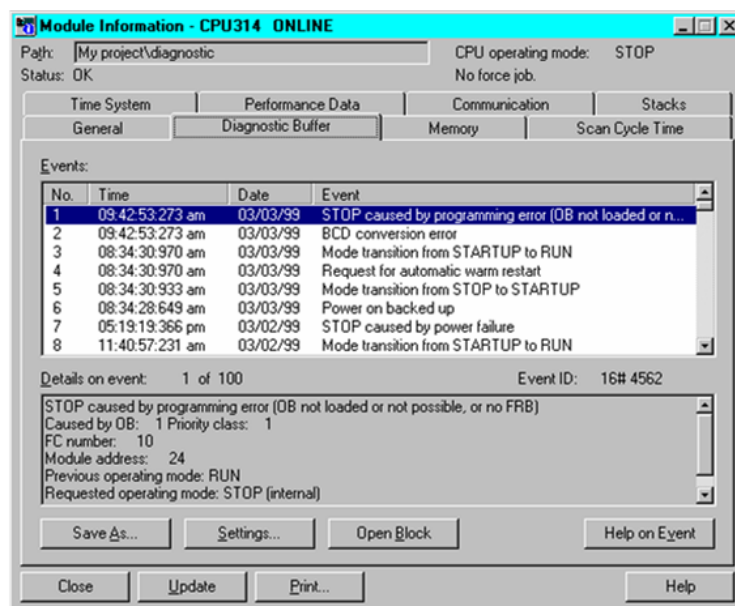
۷-آشنایی با پنجره Module Information

توسط این گزینه، اطلاعات جامعی از CPU در حالت آنلاین در دسترس قرار می گیرد. این اطلاعات شامل پیغام های CPU، زمان سیکل اسکن، مشاهده فضاهای حافظه و می باشد



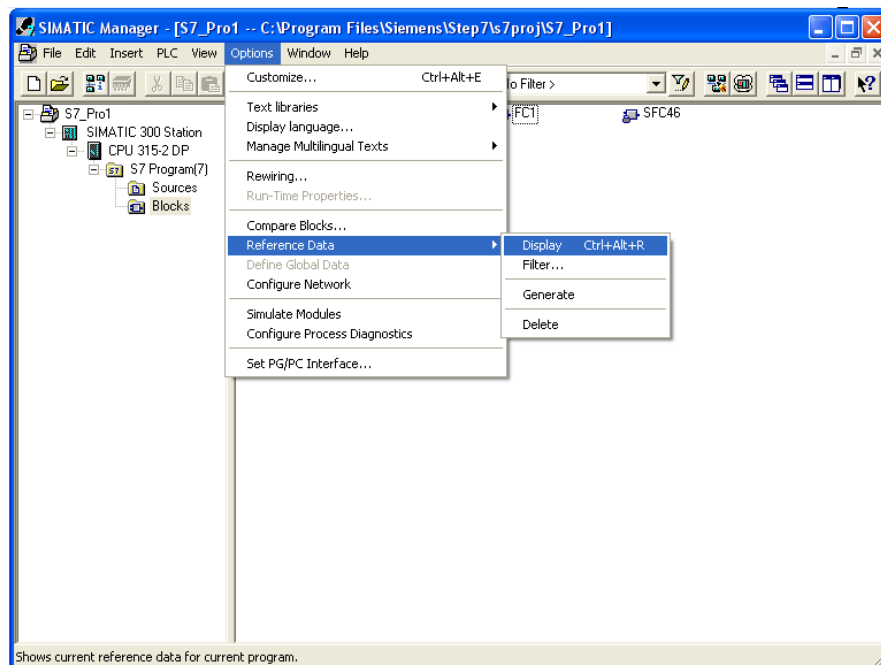
با مراجعه به Diagnostic Buffer می توان به پنجره مربوط به پیغام های CPU دسترسی پیدا کرد. تمامی فالت ها و سایر موارد به همراه ساعت و تاریخ در این بافر ثبت می شود. این بافر در بحث عیب یابی بسیار مفید

می باشد.

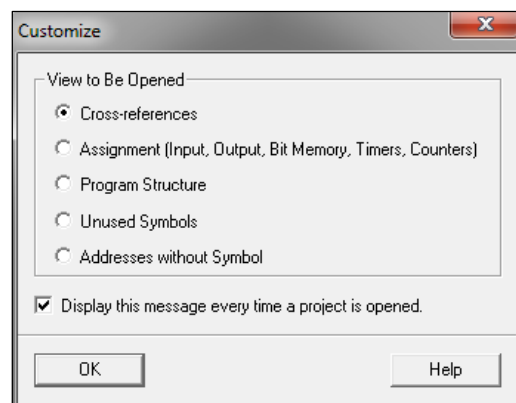


۸- قابلیت Reference Data

این گزینه اطلاعات مفیدی در رابطه با برنامه در اختیار کاربر قرار میدهد. این اطلاعات شامل آدرس های استفاده شده در برنامه، ساختار بلوک های برنامه و می باشد.



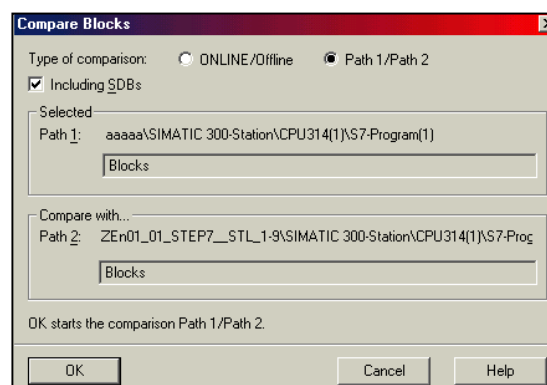
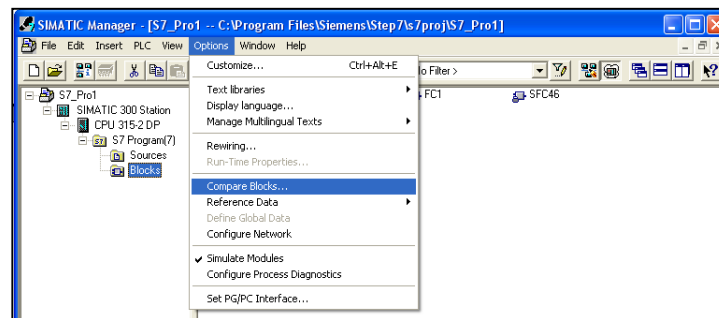
بعد از اینکه بر روی گزینه Display کلیک نمودید، پنجره ای مطابق شکل زیر نمایان می شود.



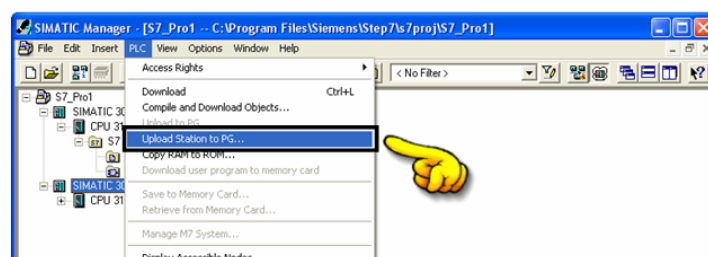
با انتخاب هر یک از گزینه های فوق، به صفحه ای جهت مشاهده اطلاعات مختلف در رابطه با بخش های مختلف پروژه مراجعه می شود.

۹- مقایسه بلوک ها Compare Blocks

توسط این ابزار می توان یک پروژه یا یک یا چند بلوک از یک پروژه در PC را با بلوکهای داخل PLC مقایسه نمود. عملیات مقایسه می تواند در حالت آنلاین با PLC و یا در حالت آفلاین بین دو پروژه داخل PC باشد.

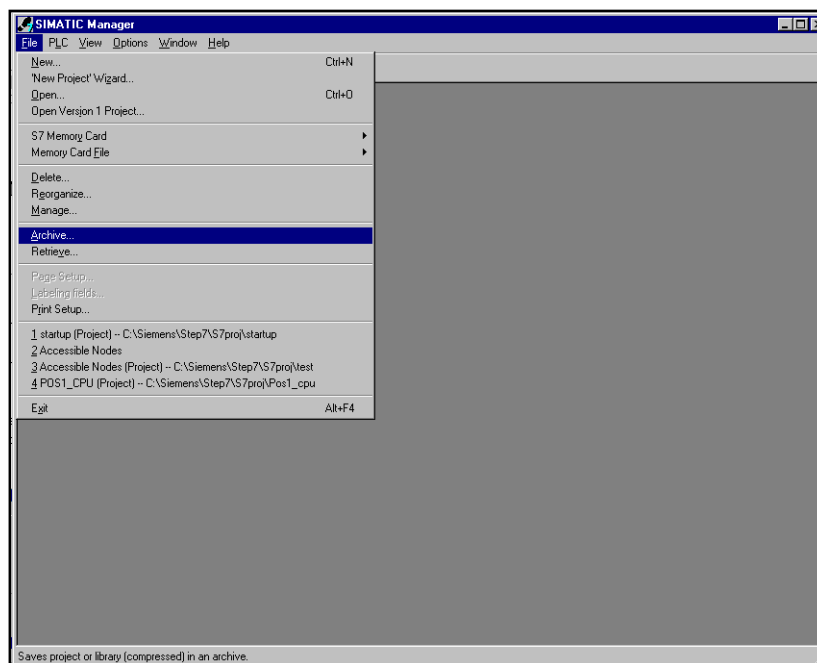


۱۰- گرفتن Backup از کل پروژه داخل PLC



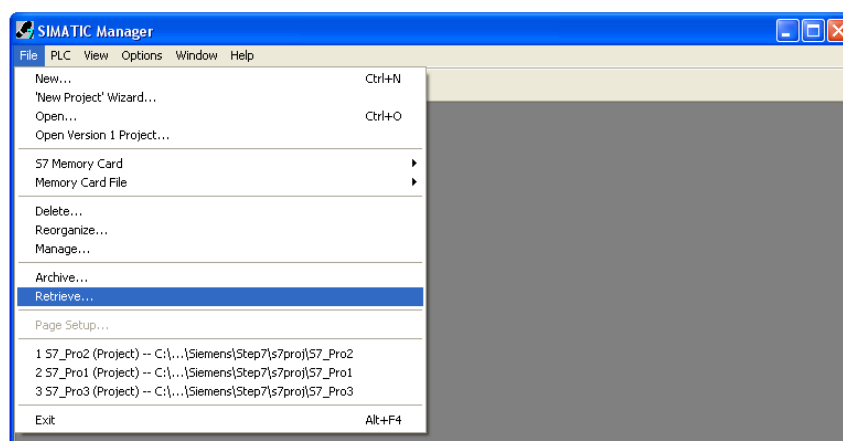
۱۱- بایگانی پروژه Archive

توسط این قابلیت در نرم افزار کاربران می توانند پروژه خود را با حجمی بسیار کم بر روی هارد و یا یک دیسکت بایگانی کنند. این موضوع یکی از امکانات مهم در اکثر نرم افزارهای مربوط به PLC می باشد.



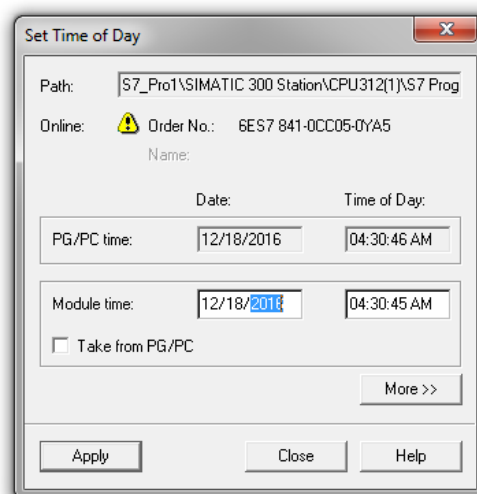
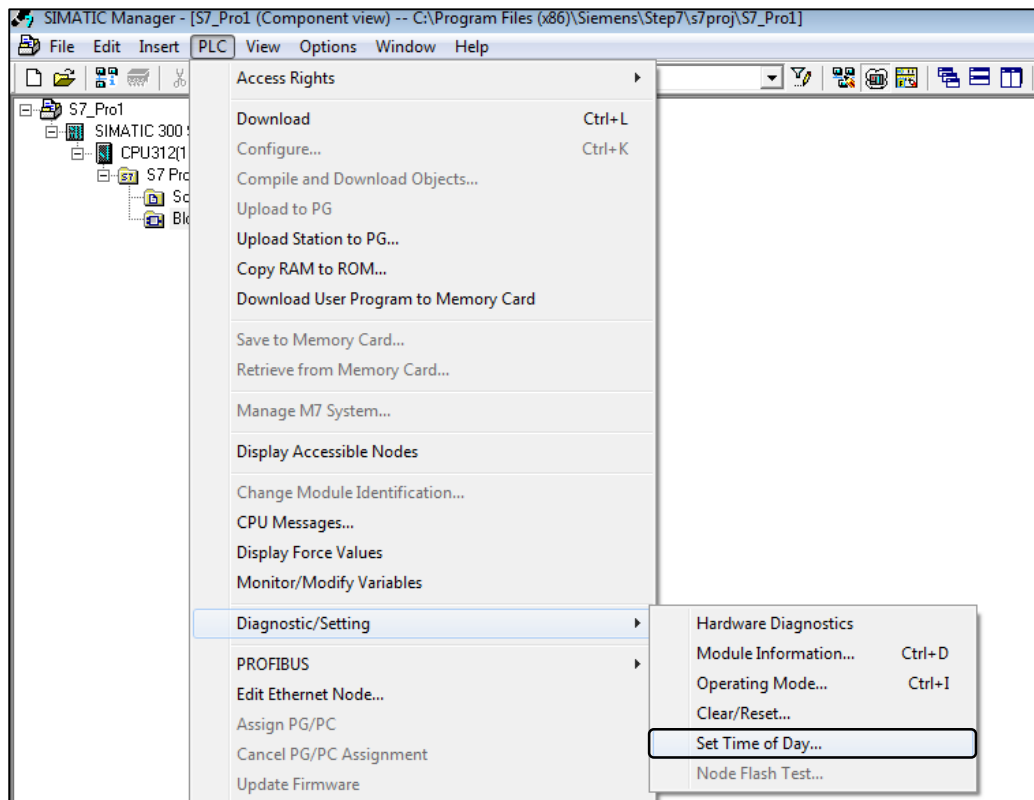
۱۲- بازیابی پروژه بایگانی شده (Retrieve)

توسط گزینه Retrieve که در منوی File قرار گرفته شده است می توانید فایل آرشیو شده را مجدداً بازیابی کنید. پروژه ای که با قابلیت Archive ذخیره شده باشد حتماً می بایست با Retrieve باز گردد.



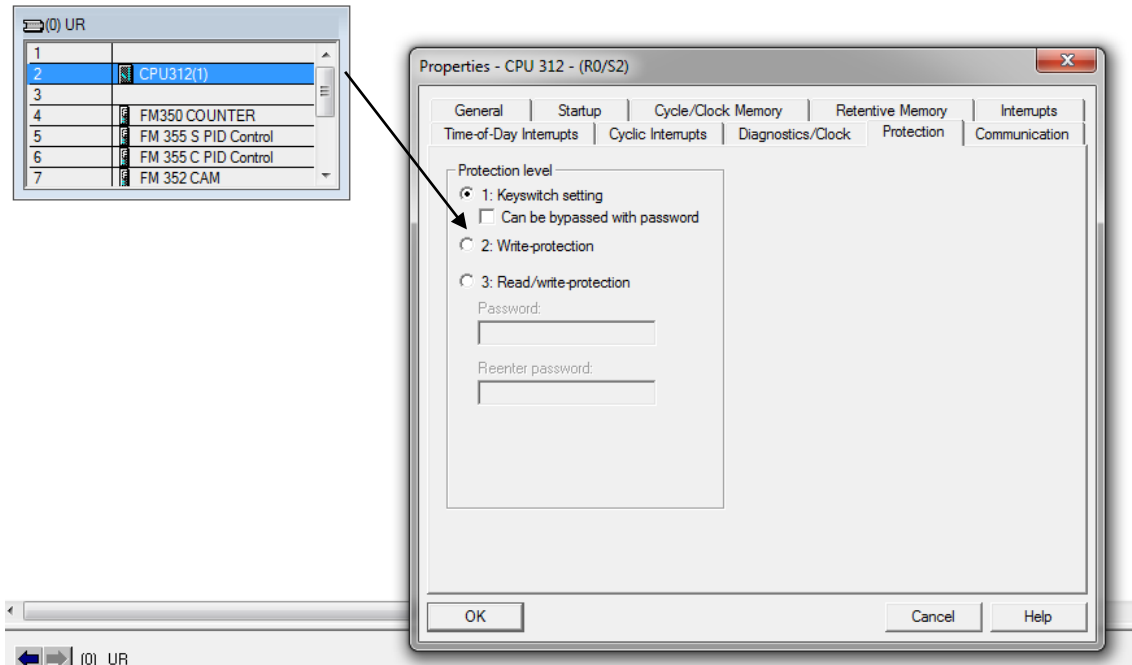
۱۳-تنظیم ساعت و تاریخ CPU

توسط این گزینه می توان ساعت و تاریخ CPU را تنظیم و با کلیک بر روی گزینه Apply به CPU اعمال نمود.



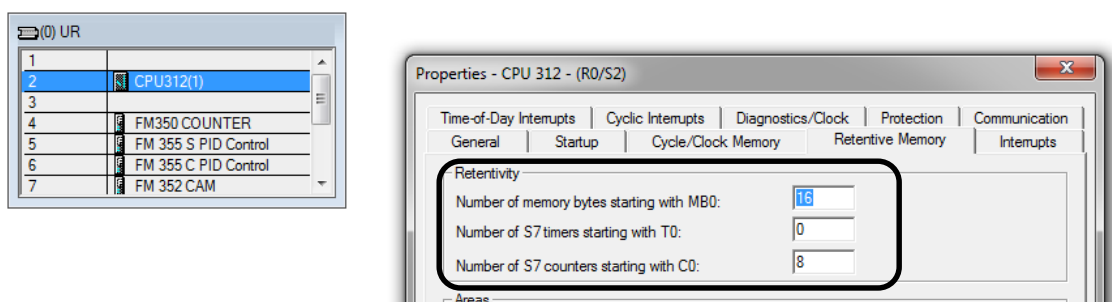
۱۴-محافظت از برنامه CPU

توسط این گزینه می توان CPU را در مقابل نوشتن و یا خواندن/نوشتن محافظت کرد.



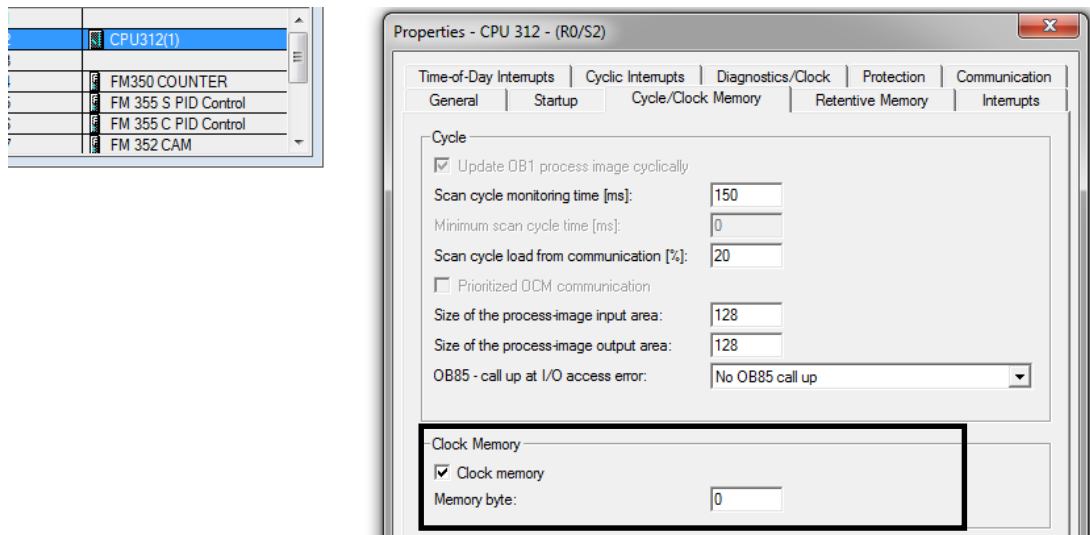
۱۵-سازماندهی فضای حافظه

توسط این گزینه می توان فضاهای حافظه را در دو گروه پایدار و ناپایدار مدیریت کرد.



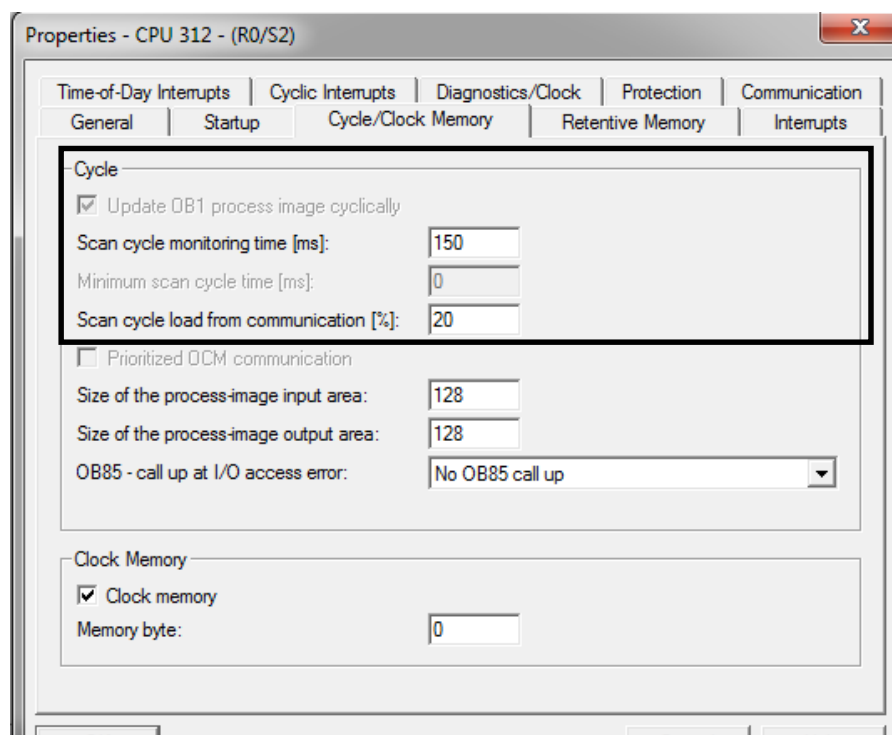
۱۶- استفاده از پالس CPU

با استفاده از این گزینه می توان یک بایت را جهت دریافت پالس های آماده CPU اختصاص داد.

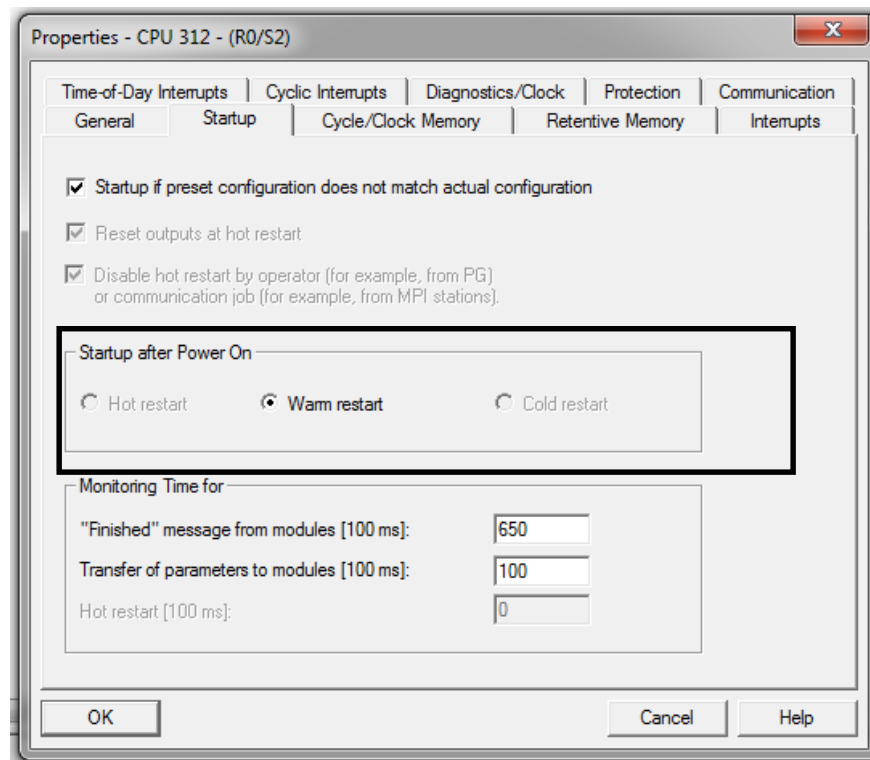


۱۷- تنظیمات مربوط به حداقل و حداکثر زمان سیکل اسکن

توسط این سربرگ می توان حداکثر و حداقل زمان مجاز برای سیکل اسکن را مشخص کرد. تعیین حداقل زمان در CPU های 400 امکان پذیر می باشد.

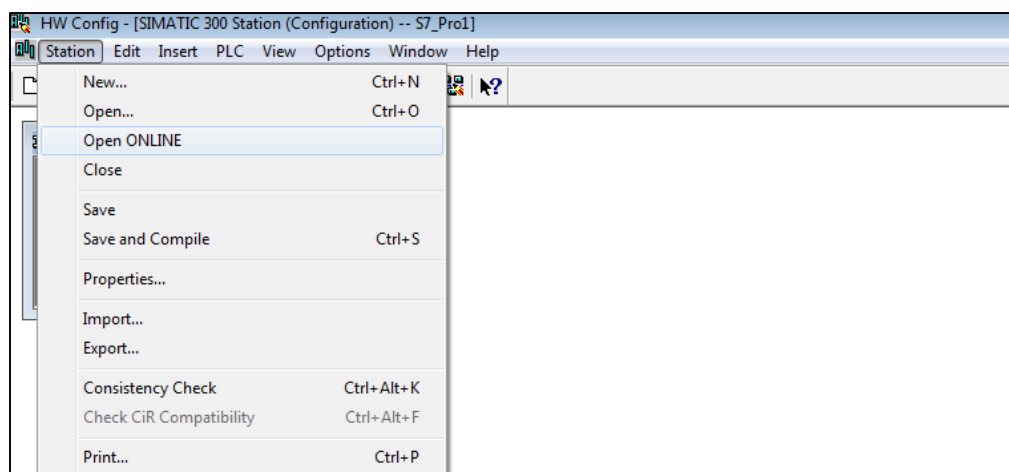


۱۸- تعیین نوع ری استارت CPU



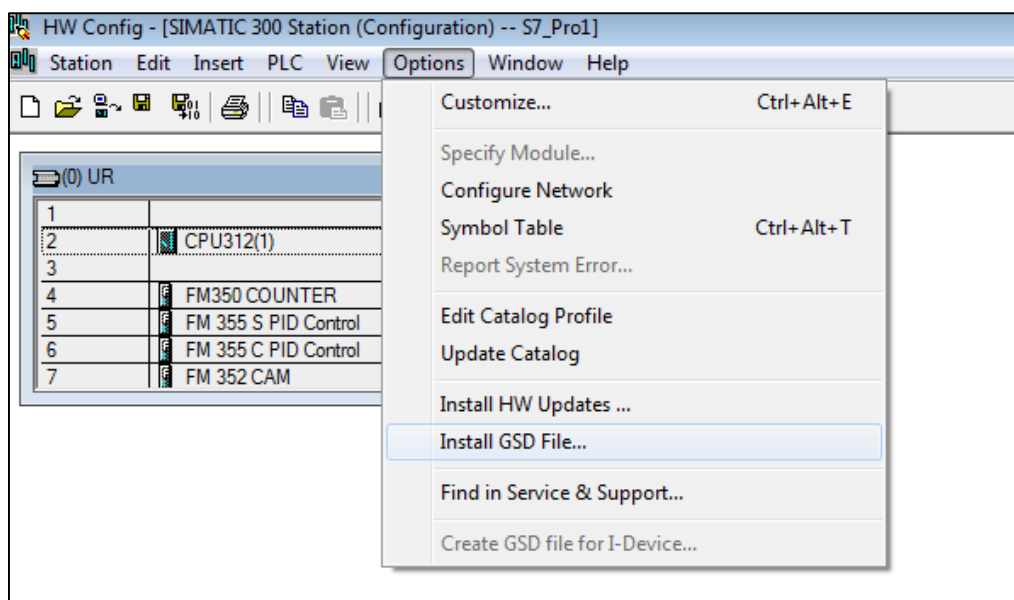
۱۹- مشاهده آنلاین سخت افزار

توسط این گزینه می توان سخت افزار پروژه را در حالت آنلاین مشاهده نمود.



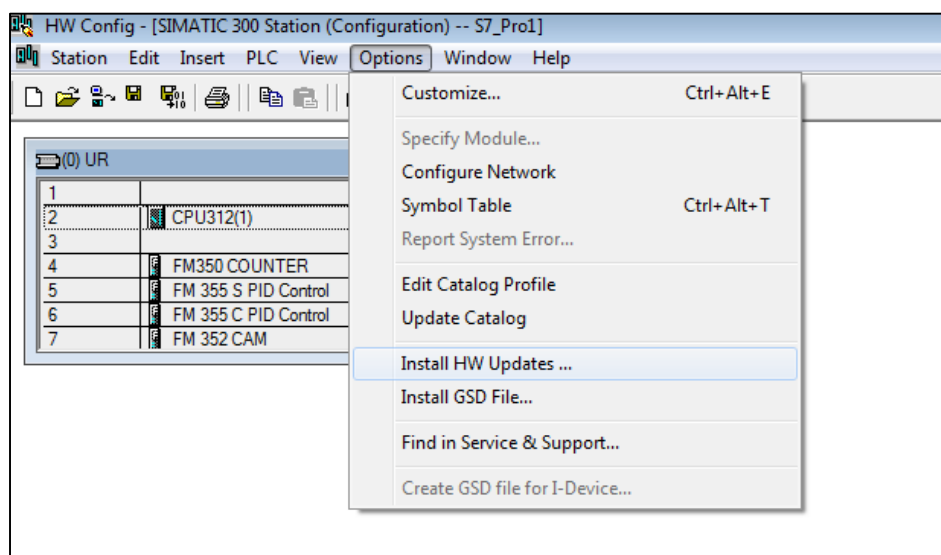
۲۰- نصب فایل GSD

توسط این گزینه می توان تجهیزات برندهای دیگر و یا سایر محصولات زیمنس را به لیست کاتالوگ اضافه نمود. این ایستگاه ها ، توسط فایل هایی با نام GSD به لیست اضافه می شوند.



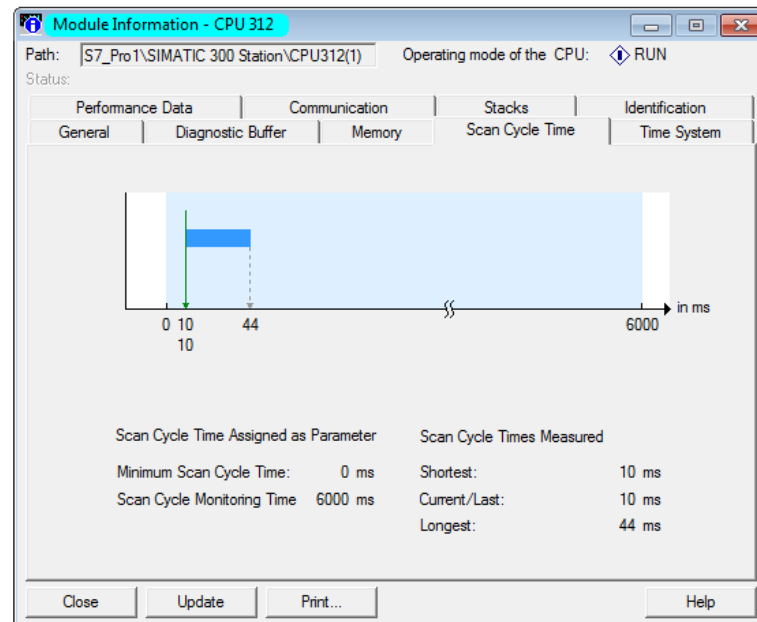
۲۱- آپدیت کردن لیست کاتالوگ

با انتخاب این گزینه، نرم افزار با اتصال به سایت زیمنس و دانلود مازول های جدید، لیست کاتالوگ را آپدیت می کند.



۲۲- مشاهده زمان سیکل اسکن CPU

توسط این زبانه می توان مدت زمان جاری سیکل اسکن، بیشترین زمان و کمترین زمان سیکل را در حالت آنلاین مشاهده کرد.



۲۳- مشاهده مشخصات CPU به صورت آنلاین

توسط این زبانه می توان به مواردی از قبیل بلوک های سیستمی، تعداد تایمر، کانتر و بلوکهای وقفه که یک CPU می تواند ساپورت می کند.

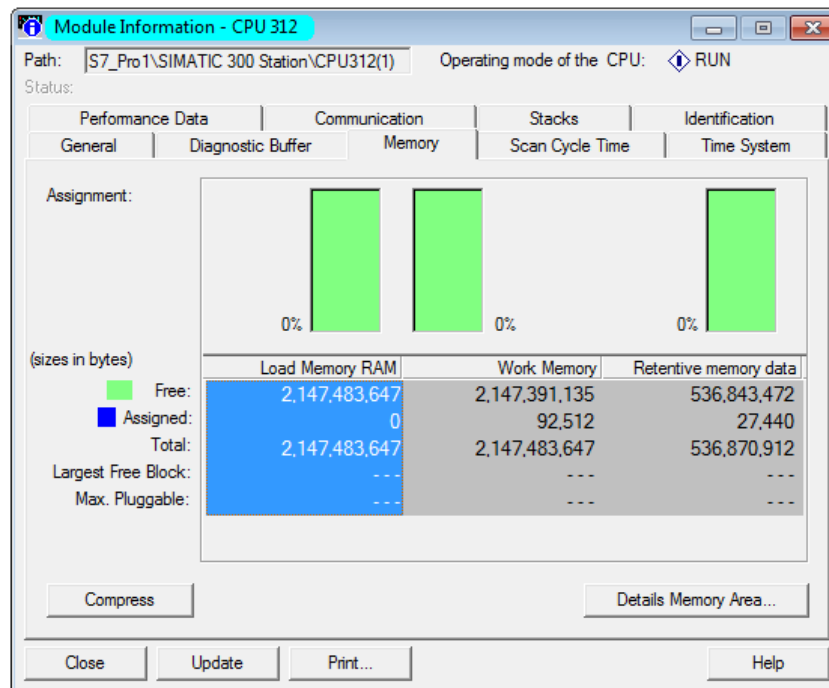
The screenshot shows the 'Module Information - CPU 312' window with the 'System Blocks' and 'Address Areas' sections expanded.

Organization Blocks:			System Blocks:		
No.	Function		No.	Name	Symbol Comment
OB1	Free scan cycle	- start event: sta...	SFB0	CTU	Count Up
OB10	Time-of-day interrupt	- start eve...	SFB1	CTD	Count Down
OB11	Time-of-day interrupt	- start eve...	SFB2	CTUD	Count Up/Down
OB12	Time-of-day interrupt	- start eve...	SFB3	TP	Generate a Pulse
OB13	Time-of-day interrupt	- start eve...	SFB4	TON	Generate an On ...
OB14	Time-of-day interrupt	- start eve...	SFB5	TOF	Generate an Off ...
OB15	Time-of-day interrupt	- start eve...	SFB8	USEND	Uncoordinated S...
OB16	Time-of-day interrupt	- start eve...	SFB9	URCV	Uncoordinated ...

Address Areas:			
Address type	Quantity	Area from	to / max. length
Process Image Inputs	131072 (Bits)	I0.0	I16383.7
Process Image Outputs	131072 (Bits)	Q0.0	Q16383.7
Bit Memory	131072 (Bits)	M0.0	M16383.7
Timers	2048	T0	T2047
Counters	2048	C0	C2047
Local Data	32768 (Bytes)		

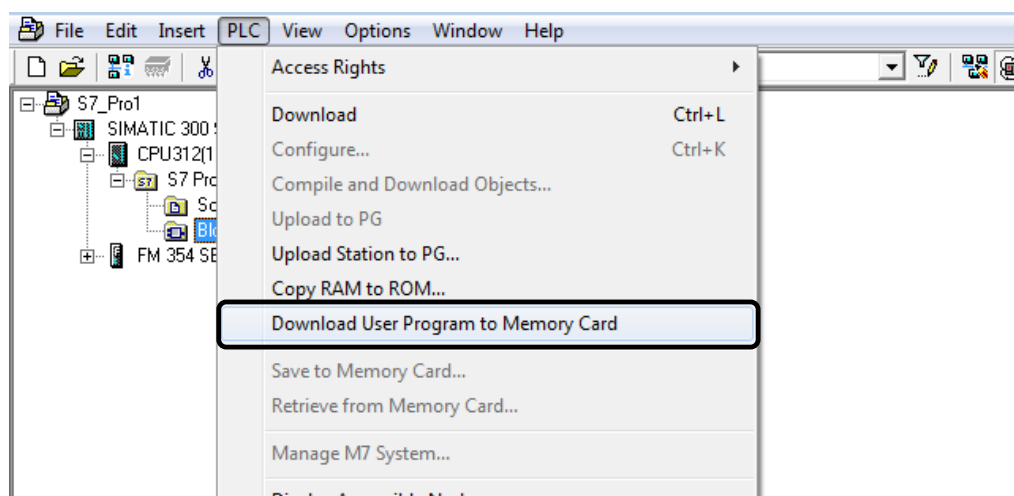
۲۴- مشاهده حجم فضای حافظه استفاده شده

توسط این ابزار می توان مقدار فضای حافظه Load Memory و System Memory مربوط به CPU را مشاهده و در رابطه با فضای اشغال شده اطلاعات لازم را کسب نمود.

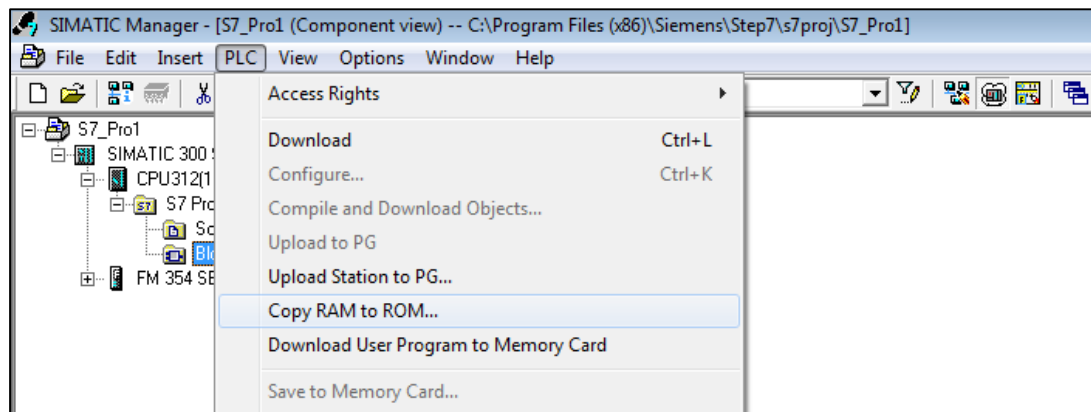


۲۵- انتقال برنامه PC به روی کارت MC از نوع Flash

توسط این گزینه در حالت STOP می توان پروژه را به صورت کلی بر روی کارت حافظه MC دانلود نمود.

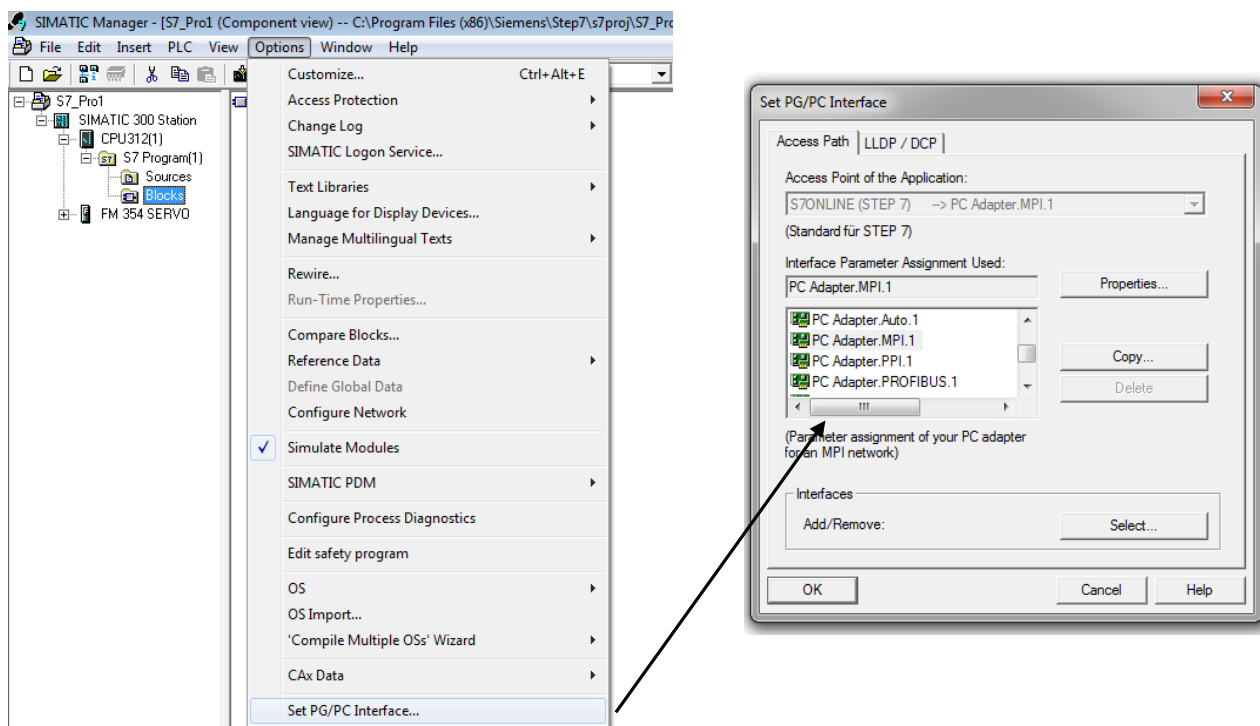


۲۶- انتقال برنامه از روی RAM داخلی به روی ROM داخلی (CPU 314 IFM)

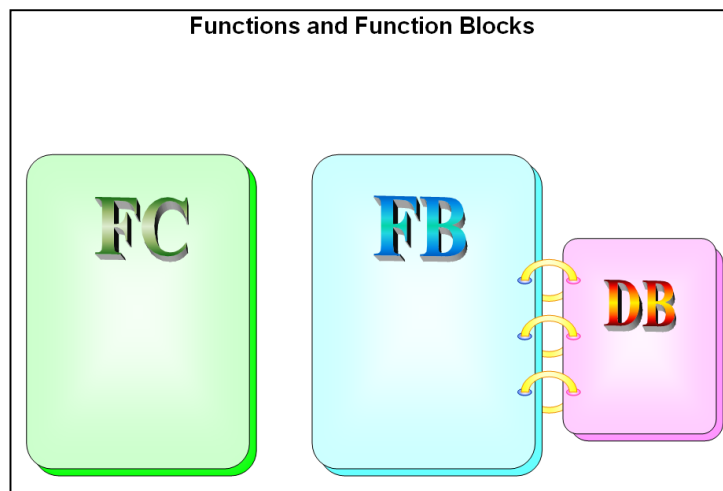


۲۷- تنظیمات درایورها و کابل های ارتباطی

توسط این گزینه می توان تنظیمات مربوط به کابل های ارتباطی نصب شده را انجام داد.



آشنایی با بلوک های برنامه



آشنایی با روش های برنامه نویسی

در اکثر نرم افزارهای برنامه نویسی معمولاً برنامه نویس می تواند برنامه خود را با توجه به حجم و نوع پروژه کنترلی به روش های مختلفی طراحی نماید. منظور از روش های مختلف برنامه نویسی روش های STL، FBD و LAD نمی باشد. منظور از روش های برنامه نویسی در این قسمت ساختاری است که برنامه نویس برای برنامه خود در نظر می گیرد. معمولاً در برنامه کنترلی می توان از ۳ ساختار استفاده نمود. البته لازم به ذکر است که انتخاب ساختار مناسب برای برنامه، بستگی به حجم و نوع برنامه دارد.

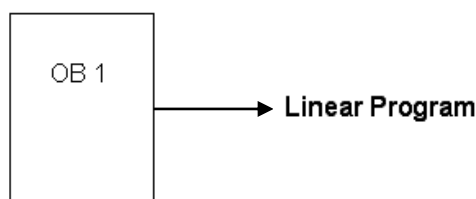
- Linear program
- Divided program
- Structured Program

Linear program •

این روش برنامه نویسی تحت عنوان برنامه نویسی خطی شناخته می شود. در این روش تمامی برنامه مربوط به یک پروژه در یک بلوک و به صورت متوالی نوشته می شود. منظور از بلوک همان بلوک OB1 در نرم افزار می باشد. در ابتدا لازم است که ویژگی و نحوه اجرای بلوک OB1 مورد بررسی قرار گیرد. در اکثر PLC های شرکت زیمنس بلوک هایی با نام های مختلف جهت کاربردهای مختلف تعبیه شده است.

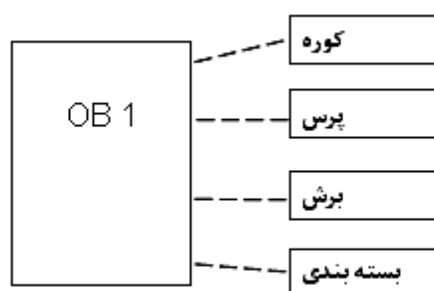
اما وجه اشتراک بین تمامی PLC های زیمنس استفاده از بلوک OB1 به عنوان بلوک اصلی در برنامه می باشد. زمانی که CPU به حالت RUN سوئیچ می شود، CPU در ابتدا در صورت وجود OB های راه اندازی به آنها مراجعه و سپس نیاز به بلوک OB1 دارد و در واقع سیستم عامل بلوک OB1 را فراخوانی می کند. البته اجرای OB1 را نیز می توان توسط سایر OB ها در هر لحظه قطع نمود. پس در نتیجه بلوک OB1 به عنوان ساختار اصلی برنامه کاربر بوده که اکثر بلوک های برنامه جهت اجرا حتما می بایست در این بلوک فراخوانی شوند. در برنامه نویسی خطی برنامه نویس کل برنامه خود را در بلوک OB1 در Network های متوالی وارد می کند. در پروژه های صنعتی که حجم برنامه کنترلی زیاد می باشد، نوشتن تمامی برنامه در بلوک OB1 کار صحیح و منطقی نمی باشد. به دلیل اینکه تمامی برنامه می بایست پشت سرهم و در یک بلوک طراحی شود.

یکی از معایب برنامه نویسی خطی عیب یابی مشکل در برنامه نوشته شده می باشد. همچنین در بسیاری از مواقع نیاز می باشد که قسمتی از برنامه توسط CPU پردازش نشود. در برنامه نویسی خطی به دلیل اینکه تمامی برنامه در یک بلوک می باشد، برنامه نویس معمولا کمتر می تواند بر روی بخش های مختلف برنامه کنترل لازم را داشته باشد. از طرفی برخی از ماشین آلات صنعتی دارای چند مد کاری می باشند. به عنوان مثال فرض کنید که یک دستگاه دارای ۲ مد دستی و اتوماتیک می باشد. برنامه این دو مد کاری با یکدیگر متفاوت می باشند. از طرفی هر دو مد مربوط به یک ماشین بوده و اگر از برنامه نویسی خطی استفاده گردد امکان تداخل آدرس و المانهای موجود در برنامه با یکدیگر وجود دارد. پس در نتیجه استفاده از این روش برنامه نویسی در پروژه هایی که حجم برنامه زیاد می باشد توصیه نمی شود.



Divided program

در این روش برنامه نویسی از بلوک های دیگری جهت نوشتن برنامه استفاده می شود. در واقع در این روش قسمت های مختلف مربوط به یک پروژه در بلوک های مجزا از یکدیگر نوشته می شوند. این بلوک ها به عنوان زیربرنامه مورد استفاده قرار می گیرند. به عنوان مثال یک خط تولید را در نظر بگیرید که شامل بخش های مختلفی از جمله کوره، پرس، برش و بسته بندی می باشد. در این حالت با استفاده از بلوک های زیربرنامه می توان برنامه مربوط به هر بخش را از بخش دیگر جدا نمود.

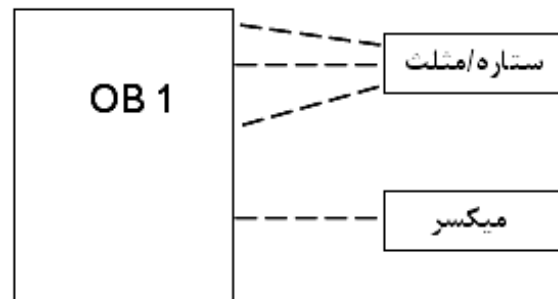


این نوع برنامه نویسی که کاربرد فراوانی در پروژه های بزرگ دارد، تحت عنوان برنامه نویسی تقسیم بندی شده شناخته می شود و یکی از مزایای این روش عیب یابی آسان و تحت کنترل درآوردن هر چه بیشتر اجزای مختلف برنامه می باشد. در PLC مختلف شرکت زیمنس نام های مختلفی برای این بلوک ها در نظر گرفته شده است که در ادامه در نرم افزار مورد بحث با این بلوک ها آشنا می شویم.

Structured Program

روش دیگری که در پروژه های بزرگ مورد استفاده قرار می گیرد روش برنامه نویسی ساختار یافته می باشد. از این روش در پروژه هایی که یک منطق بین چندین بخش از پروژه مشابه می باشد استفاده می شود. در واقع استفاده از این روش زمان برنامه نویسی را بسیار کاهش می دهد. به عنوان مثال فرض کنید که قرار است ۱۰ موتور توسط یک PLC کنترل شوند. راه اندازی تمامی موتورها به صورت ستاره/ مثلث دستی می باشد. تنها نقطه تفاوت در این پروژه آدرس های I/O مربوط به هر موتور می باشد. در این گونه موارد نیازی به تکرار برنامه برای ۱۰ موتور به صورت مجزا نمی باشد.

در این صورت برنامه نویس می تواند یک تابع با منطق ستاره/ مثلث طراحی و این تابع را برای تمامی موتورها استفاده کند. در نتیجه استفاده از روش ساختار یافته در بخش هایی از برنامه که قرار است تکرار شود بسیار مفید می باشد.



آشنایی با بلوک های FC

همانطوریکه بیان شد بلوک OB1 به عنوان بلوک اصلی برنامه می باشد که بصورت سیکلی توسط سیستم عامل پردازش می شود. اما در اکثر PLC های زیمنس بلوک هایی نیز به عنوان زیربرنامه می توانند در برنامه مورد استفاده قرار گیرند. کسانی که با PLC های سری S5 آشنایی دارند به خوبی می دانند که در PLC های سری S5 علاوه بر بلوک OB1 امکان نوشتن برنامه در بلوک های PB نیز وجود دارد. در نرم افزار STEP7 نیز بلوک هایی با نام FC جهت نوشتن برنامه کنترلی تعبیه شده است. همچنین از مزایای مهم بلوک های FC می توان به قابلیت تابع سازی آنها اشاره نمود.

مثال

از بلوک های FC می توان بصورت جداگانه به عنوان زیربرنامه های OB1 استفاده نمود. در این مثال هدف نوشتن دو برنامه در دو FC بصورت مجزا و فراخوانی آنها در OB1 می باشد. (برنامه نویسی تقسیم شده)

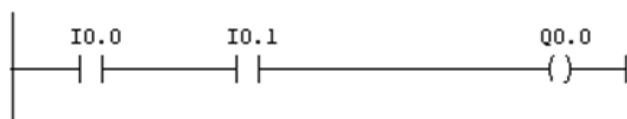
فرض کنید می خواهیم یک موتور را به دو صورت کنترل کنیم. جهت راه اندازی این موتور از دو کلید استارت استفاده می شود. به غیر از این دو کلید از یک کلید سلکتوری نیز برای تعیین منطق بین استارت ها استفاده شده است. در واقع این کلیدها می توانند هم بصورت سری و هم بصورت موازی فرمان وصل موتور را صادر کنند.

سری یا موازی بودن کلیدها توسط کلید سلکتوری مشخص می گردد. این کلید دارای دو حالت A و B می باشد.

اگر کلید در وضعیت A قرار داشت، باید کلیدهای استارت سری و اگر کلید در وضعیت B بود همان دو کلید استارت بصورت موازی وارد مدار شوند. حالت A به ورودی I 2.0 و حالت B به ورودی I 2.1 متصل می باشند.

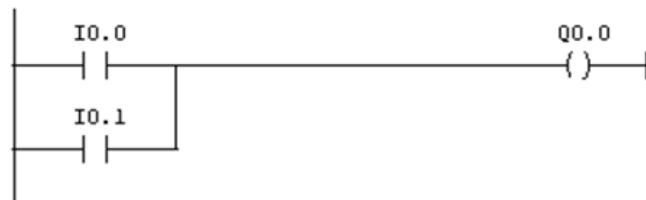
برنامه FC 1 (حالت سری)

Network1



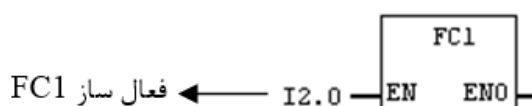
برنامه FC2 (حالت موازی)

Network1

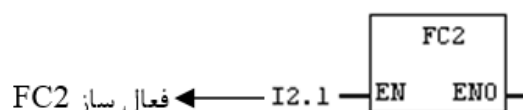


فراخوانی در OB1

Network1



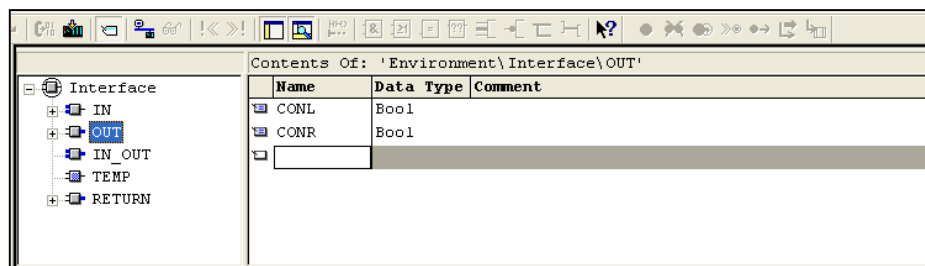
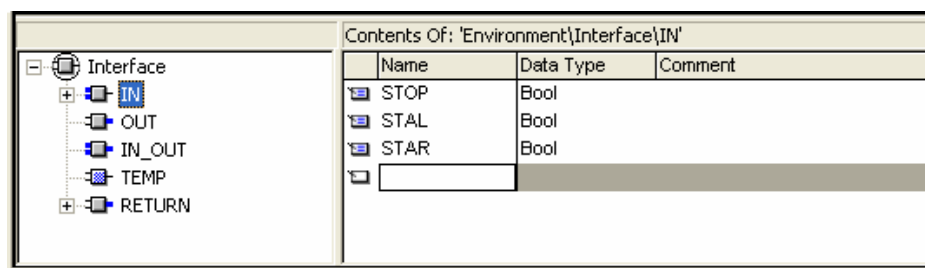
Network2



همانطور که در برنامه OB 1 ملاحظه می کنید، از دو کلید I 2.0 و I 2.1 جهت شروط اجرا شدن FC ها استفاده شده است. در این برنامه به عنوان مثال برای اجرا شدن FC1 و راه اندازی موتور به صورت سری ، حتما باید ورودی I 2.0 فعال باشد. با غیر فعال شدن I 2.0 مرور برنامه توسط CPU در FC1 قطع می گردد.

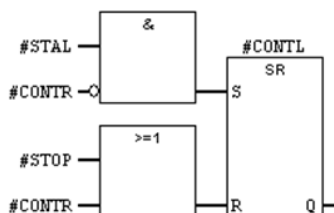
مثال

فرض کنید در یک فرآیند صنعتی از ۴ موتور استفاده شده است. عملکرد هر ۴ موتور به صورت چپگرد راستگرد می باشد. در این مثال هدف ساخت یک تابع چپگرد راستگرد برای ۴ موتور در یک بلوک FC می باشد.



در ادامه نوبت به نوشتن برنامه کنترلی می باشد. جهت نوشتن یک تابع حتما می بایست از پارامترهای جدول با در نظر گرفتن نام مشابه استفاده نمود.

Network1

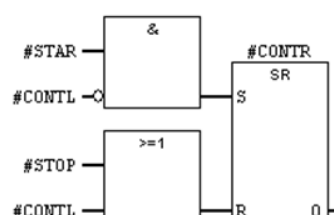


```

A      #STAL
AN     #CONTR
S      #CONTR
A(
O      #STOP
O      #CONTR
)
R      #CONTR

```

Network2



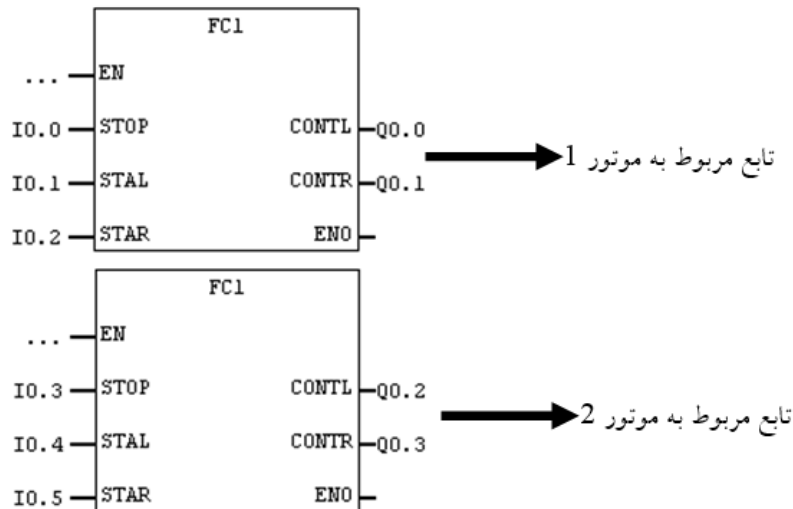
```

A      #STAR
AN     #CONTR
S      #CONTR
A(
O      #STOP
O      #CONTR
)
R      #CONTR

```

بعد از اینکه نوشتن برنامه کنترلی در FC1 به پایان رسید، نوبت به فراخوانی تابع FC1 در OB1 می باشد. برای این منظور بلوک FC1 را ذخیره و وارد محیط OB1 شوید. در این محیط به قسمت فراخوانی بلوک FC مراجعه و عملیات فراخوانی این بلوک را برای ۴ موتور تکرار کنید.

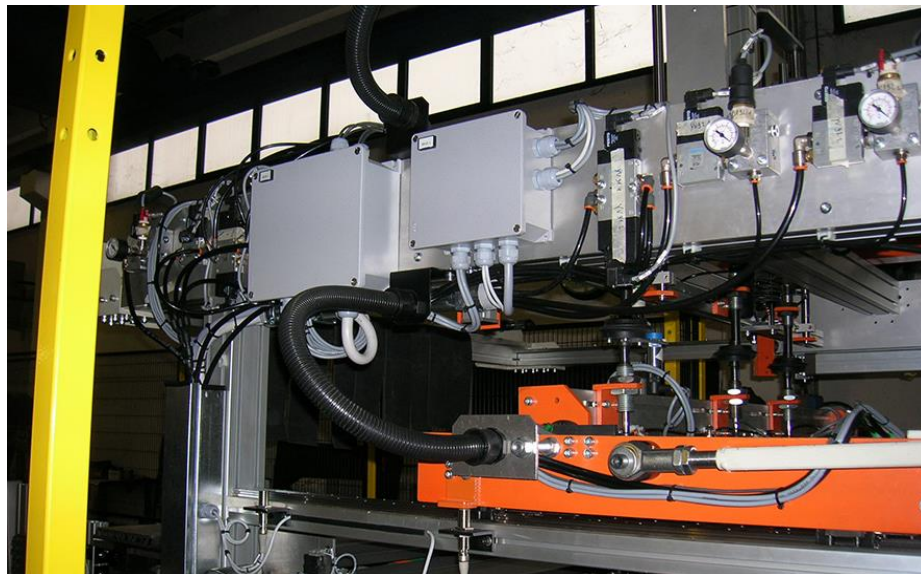
بلوک FC1 را در چهار Network برای چهار موتور فراخوانی کنید. بعد از اینکه عملیات فراخوانی به پایان رسید در نهایت نوبت به آدرس دهی I/O های تابع می باشد.



فراخوانی را برای ۲ موتور دیگر ادامه می دهیم

فصل پنجم: مدارهای فرمان در سیستم‌های پنوماتیک

پنیوماتیک از کلمه پنیوما که یک کلمه یونانی به معنای تنفس یا باد می باشد گرفته شده است. در واقع پنیوماتیک علمی هست که در مورد حرکت باد می باشد. یا به عبارت دیگر دانش استفاده از انرژی هوای فشرده برای انجام دادن کار را پنیوماتیک می گویند. امروزه کمتر کارخانه یا کارگاهی را می توان یافت که از تجهیزات پنیوماتیک استفاده نکرده باشند.



کمپرسور (Compressor)

کمپرسور یا متراکم کننده، برای فشرده کردن سیالات به کار می رود. در برخی از دستگاه‌ها و ماشین‌آلات، کمپرسورها وسایلی هستند که توسط آنها هوا فشرده شده و سپس به سمت قسمت‌های مختلف سیستم فرستاده می شود.

در سیستم های پنیوماتیک فراهم کردن هوای فشرده (متراکم کردن) به عهده کمپرسور می باشد. کمپرسورها در انواع پیستونی، مارپیچی و تیغه ای در صنعت موجود می باشند.

به عنوان نمونه کمپرسور پیستونی قدیمی ترین و متداولترین انواع کمپرسور است. این کمپرسور در شکل های دو طرفه یا یکطرفه موجود می باشند که قطعات آنها روغنکاری شونده یا روغن کاری نشونده می باشند که به اشکال متفاوتی دارای تنوع تعداد سیلندر هستند. به استثنای کمپرسورهای واقعاً کوچک که دارای سیلندرهای عمودی هستند. شکل بندی V برای کمپرسورهای کوچک رایج ترین است.

در کمپرسورهای بزرگ دو طرفه، شکل بندی نوع L با سیلندر عمودی فشار پایین و سیلندر افقی فشار بالا مزایای بسیار زیادی داشته و به همین علت است که این نوع طرح نسبت به طرح‌های دیگر رایج تر است. کمپرسورهایی که روغنکاری می‌شوند، بطور معمول با سیستم روغن کاری پاششی یا روغن کاری فشاری کار می‌کنند. بیشتر کمپرسورها دارای سوپاپ‌های خودکار هستند. یک سوپاپ خودکار به هنگام بروز اختلافات فشار در دو سمت بشقابک سوپاپ باز و بسته می‌شود.



سمبل‌ها

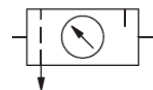
- | | |
|---|--|
| ➤ کمپرسور با دبی ثابت. | |
| ➤ مخزن هوای فشرده با انشعاب T. | |
| ➤ منبع فشار | |
| ➤ فیلتر | |
| ➤ جدا کردن و فیلتر کردن ذرات | |
| ➤ آب گیر (جدا کننده آب)، کارانداز دستی. | |
| ➤ آب گیر با تخلیه اتوماتیک. | |

واحد مراقبت (Unit Service)

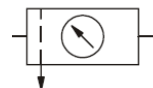
مجموعه واحد مراقبت را که به طور معمول از سه عنصر فیلتر، رگولاتور و چرب کننده هوای فشرده تشکیل می شود را FRL نیز می نامند. اگر چه رطوبت و ناخالصی های هوای فشرده تا حدودی در بخش تولید آن گرفته و جدا می شود، اما برای استفاده نهایی از هوای فشرده در تجهیزات گوناگون پنوماتیک به هوای فشرده با کیفیت بالاتر نیاز می باشد. از این رو در نزدیکی مصرف کننده مجموعه ای به نام واحد مراقبت جهت بالا بردن کیفیت هوا نیاز داریم. وظیفه فیلتر گرفتن ذرات غبار آب و روغن کمپرسور می باشد. همچنین روغن باش یا همان چرب کننده نیز وظیفه روانکاری هوا را بر عهده دارد. روتکار مورد نیاز دستگاه را می توان با مخلوط کردن در هوای فشرده (چرب کردن هوا) به قسمت های گوناگون رسانید. در این قسمت با چکاندن قطرات روغن بر روی جریان هوای فشرده، هوای گذرنده از روغن پاش چرب می شود.

وظیفه رگولاتور به عنوان یکی دیگر از بخش های واحد مراقبت تنظیم هوای فشرده برای مصرف کننده ها می باشد. گاهی ممکن است فشار هوای فشرده مورد نیاز یک مصرف کننده کمتر از فشار هوایی باشد که کمپرسور ایجاد کرده است. در این صورت جهت جلوگیری از آسیب دیدن مصرف کننده ها یا صرفه جویی در مصرف انرژی می توان فشار خروجی را توسط رگولاتور تنظیم نمود.

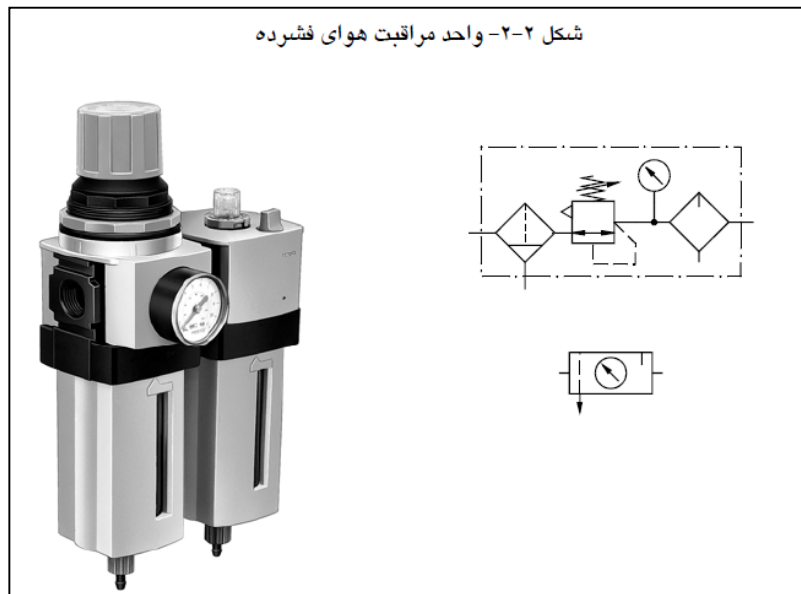
سمبول ساده شده واحد مراقبت هوای فشرده



سمبول ساده شده واحد مراقبت هوای فشرده بدون روغن زن



بر روی واحد مراقبت یک گیج فشار جهت مشاهده مقدار فشار خروجی رگولاتور نیز وجود دارد.



شیرهای پنوماتیکی (Valve)

عملکرد شیرها، کنترل فشار یا کنترل میزان جریان می باشد و براساس نوع طراحی، شیرها دارای دسته بندی زیر می باشند:

- ❖ شیرهای راه دهنده (کنترل جهت جریان)
- ❖ عناصر سیگنال دهنده (ورودی)
- ❖ عناصر پردازشگر
- ❖ عناصر کنترل
- ❖ شیرهای یکسوکننده
- ❖ شیرهای کنترل جریان
- ❖ شیرهای کنترل فشار
- ❖ شیرهای قطع و وصل

شیرهای راه دهنده:

شیر راه دهنده برای قطع و وصل انرژی هوای فشرده مورد استفاده قرار می گیرد.

هر شیر به فرمی که در زیر بیان شده است شناخته می شود:

تعداد دهانه ها: (2 راهه، 3 راهه، 4 راهه و غیره)

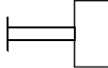
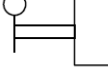
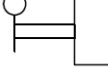
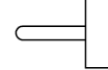
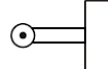
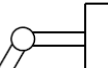
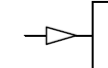
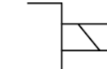
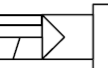
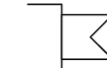
تعداد وضعیت ها: (2 وضعیته، 3 وضعیته و غیره)

روشهای تحریک شیرها (کارانداز)

برای کنترل و تغییر حالت یک شیر، باید آن را تحریک کرد. عامل تحریک می تواند به صورت دستی، مکانیکی و الکتریکی باشد.

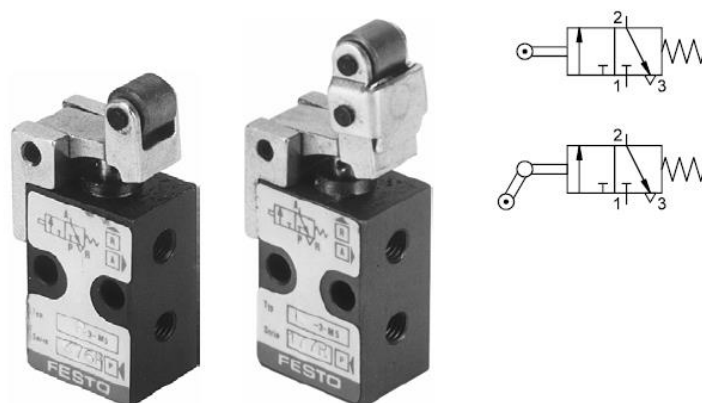
روشهای تحریک برگشت:

برگشت یک شیر نیز می تواند به صورت مکانیکی، دستی و یا الکتریکی باشد.

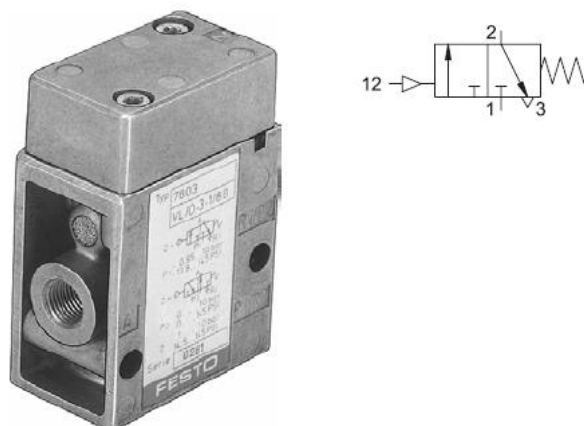
عمومی	دستی		
دکمه ای			
اهرمی			
اهرمی خارجی			
پدالی			
شاخکی	مکانیکی		
غلطکی			
غلطکی برگشت خلاص			
تحریک مستقیم پنیوماتیکی	پنیوماتیکی		
تحریک غیر مستقیم پنیوماتیکی (پیلوتی)			
تحریک تک سولونوئیدی	الکتریکی		
تحریک دو سولونوئیدی			
تحریک سلونوئیدی و پیلوتی در هر دو طرف به همراه تحریک دستی	ترکیبی		

مثال

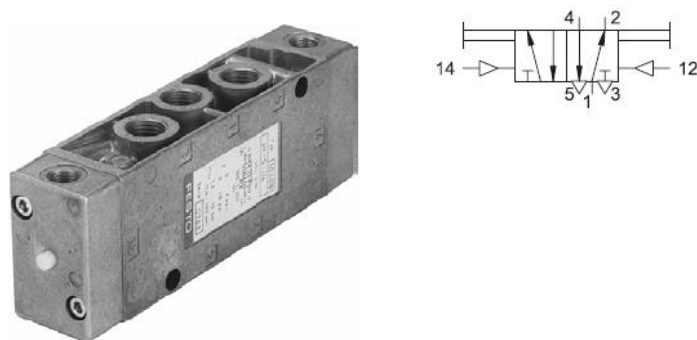
شکل ۳-۲- شیر ۳/۲ با تحریک اهرم غلطکی (بدون برگشت خلاص و با برگشت خلاص)



شکل ۴-۲- شیر ۳/۲ با تحریک هوای فشرده با برگشت فنری



شکل ۵-۲- شیر ۵/۲ برای کنترل سیلندر با دو تحریک



شیرهای راه دهنده و مراحل تکمیل سمبل

وضعیت های سوئیچی شیر با مربع نمایش داده میشود.



تعداد مربع هایی که در کنار هم قرار می گیرند نمایشگر تعداد موضع سوئیچی شیر میباشد.



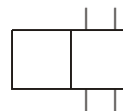
خطوط بیانگر مسیر جریان ، جهت فلش نشانگر جهت مسیر عبور جریان میباشد.



قطع بودن عبور جریان در داخل مربع با T مشخص میگردد.



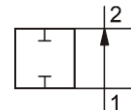
دهانه های اتصالی (تغذیه و تخلیه) با خطوط ترمیمی در قسمت چهارگوش موضع سکون و نیز موضع شروع نمایش داده میشود.



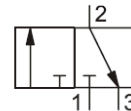
شیرهای راه دهنده

شیر راه دهنده بوسیله تعداد اتصالات کنترل شونده، تعداد وضعیت ها و مسیرهای جریان مشخص می شود. برای جلوگیری از بروز اشتباه تمام دهانه ها در حالت سکون شیر نامگذاری می شوند.

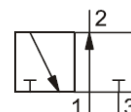
شیر راه دهنده ۲/۲ ، موضع نرمال باز.



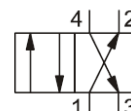
شیر راه دهنده ۳/۲ ، موضع نرمال بسته.



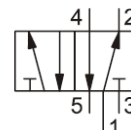
شیر راه دهنده ۳/۲ ، موضع نرمال باز.



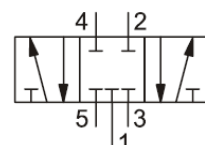
شیر راه دهنده ۴/۲ ،



شیر راه دهنده ۵/۲ ،



شیر راه دهنده ۵/۳ موضع وسط بسته .



برای معرفی شیرهای کنترل راه دهنده از یک سیستم عدد گذاری که مطابق استاندارد 3-5599 DIN ISO می باشد، استفاده می شود. پیش از این ، سیستم حروف گذاری مورد استفاده قرار می گرفت. در اینجا هر دو سیستم معرفی و طبق جدول زیر نمایش داده می شوند.

ISO 5599-3	سیستم حروف گذاری	دهانه ها یا اتصالات
۱	P	دهانه فشار (دهانه تغذیه)
۴،۲	A,B	خطوط کاری
۵،۳	R,S	دهانه های تخلیه

خطوط پیلوت

۱۰	Z	حضور سیگنال ،سبب قطع جریان از دهانه ۱ به ۲ میشود
۱۲	Y,Z	حضور سیگنال ، سبب اتصال دهانه ۱ به ۲ میشود
۱۴	Z	حضور سیگنال ، سبب اتصال دهانه ۱ به ۴ میشود
۸۱،۹۱	PZ	پیلوت کنکی

علائم شیرهای راه دهنده

Symbol	Description
	Directional control valve 2/2-way valve, closed normal position
	Directional control valve 2/2-way valve, open normal position
	Directional control valve 3/3-way valve, closed neutral position
	Directional control valve 4/2-way valve
	Directional control valve 3/2-way valve, closed normal position
	Directional control valve 3/2-way valve, open normal position
	Directional control valve 4/3-way valve, closed neutral position
	Directional control valve 4/3-way valve, exhaust neutral position
	Directional control valve 5/2-way valve
	Directional control valve 5/3-way valve, closed neutral position

Symbol	Description
	Directional control valve 5/3-way valve, exhaust neutral position
	Directional control valve 5/3-way valve, open neutral position
	Manual Control General
	Manual Control Button
	Manual Control Lever
	Manual Control Pedal
	Mechanical Control Plunger
	Mechanical Control Spring
	Mechanical Control Roller
	Mechanical Control Roller with idle return
	Solenoid with one effective winding
	Solenoid with two windings acting in opposition
	Combined Control by solenoid and pilot valve
	Pressure Control

شیرهای یکسو کننده

شیرهای یکسو کننده (Check Valve) اجازه عبور سیگنال در یک جهت را می دهند و در جهت دیگر جلوی جریان را می بندند. این عملکرد در میان انواع دیگر ، در شیرهای تعویض کننده و تخلیه سریع بیشتر به چشم می خورد . شیر یک سو کننده تحت عنوان یک عنصر پایه، بوسیله نقطه چین در درون انواع دیگر شیرها در تصاویر زیر مشخص شده است.

شیرهای یکسو کننده و مشتقات آنها

شیر با مانع برگشت بدون فنر.



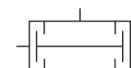
شیر با مانع برگشت با فنر.



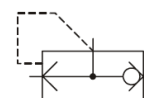
شیر تعویض کننده (یا).



شیر دو فشاره (و).



شیر تخلیه سریع.



شیرهای کنترل جریان

شیر کنترل جریان از طریق ایجاد گلویی و محدود کردن گذرگاه هوای فشرده در یک جهت خاص ، سبب کم شدن جریان میشود و نتیجتاً جریان سیال را کنترل می کند . به طور ایده ال ، باید بین دو حالت کاملاً باز و بسته محدود کننده جریان، امکان بینهایت انتخاب مختلف وجود داشته باشد .

کنترل کننده جریان باید تا حد ممکن به عنصر کاری نزدیک تر نصب شود و به گونه ای تنظیم شود که با احتیاجات سیستم در کاربردهای مختلف، مطابقت داشته باشد.

اگر کنترل کننده جریان با یک شیر یکسو کننده ترکیب شود، عملکرد آن بدین صورت خواهد بود که در یک جهت جریان کنترل می شود و در جهت دیگر جریان کامل را عبور خواهد داد.

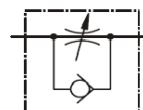


شیر کنترل جریان

شیر کنترل جریان ، قابل تنظیم.



شیر کنترل جریان یک طرفه.



شیرهای کنترل فشار

شیرهای کنترل فشار، فشار کل سیستم پنوماتیکی و یا قسمتی از آنرا تحت نفوذ و تأثیر خود قرار می دهند. شیرهای رگولاتور فشار به طور کلی بوسیله نیروی تراکمی فنرشان قابل تنظیم می باشند. سمبولهای مشهور برای این شیرها در انواع زیر خلاصه می شود.

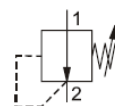
➤ شیر تنظیم کننده فشار بدون دهانه تخلیه

➤ شیر تنظیم کننده فشار با دهانه تخلیه

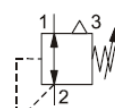
➤ شیر تابع فشار



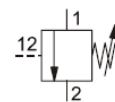
شیر تنظیم کننده فشار بدون دهانه تخلیه



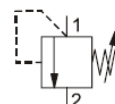
شیر تنظیم کننده فشار با دهانه تخلیه



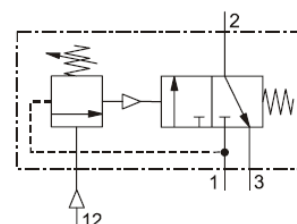
شیر تابع فشار با منبع خارجی



شیر تابع فشار درون خط



شیر تابع فشار ترکیبی

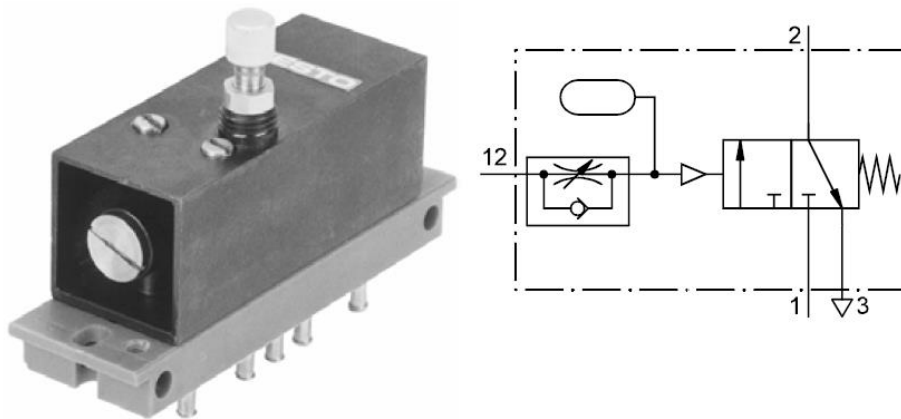


شیرهای ترکیبی

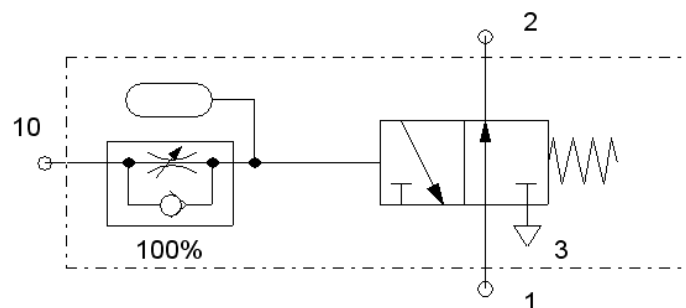
ترکیب عملکردهای عناصر مختلف، می تواند یک عملکرد جدید بوجود آورد. یک مثال مناسب، شیر تاخیر زمانی می باشد که در واقع ترکیبی از یک شیر کنترل کننده جریان یک طرفه، یک مخزن کوچک هوا و یک شیر راه دهنده 3/2 می باشد.

با تنظیم پیچ محدود کننده جریان، مقدار زیاد و یا کمی از جریان هوای فشرده در هر واحد زمانی وارد مخزن می شود. هنگامیکه فشار کنترلی لازم ایجاد شود، شیر سوئیچ می شود و جریان را عبور می دهد. تا زمانیکه سیگنال کنترل وجود داشته باشد، شیر در همین موضع باقی خواهد ماند.

شیر تاخیر زمانی (تایمر تاخیر در وصل)



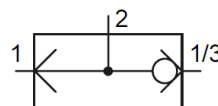
این تایمر می تواند به صورت تاخیر در قطع نیز باشد. در این صورت بعد از سپری شدن مدت زمان مورد نظر مسیر عبور هوای فشرده باز می شود.



انواع دیگر شیرهای ترکیبی نیز وجود دارند، از جمله:

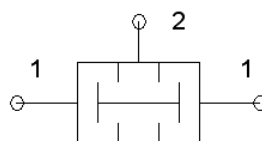
شیر OR

در این شیر سیگنال خروجی هنگامیکه که در یکی از ورودی ها فشار موجود باشد برقرار می شود.



شیر AND

در این شیر سیگنال خروجی زمانی که در هر دو ورودی فشار موجود باشد برقرار می شود.



عناصر قدرت

بخش قدرت از عناصر کنترلی و اجزاء قدرت یا عمل کننده ها، تشکیل یافته است. گروه عمل کننده شامل انواع مختلف عمل کننده های خطی و دورانی در ابعاد و با ساختمان های مختلف، می باشد. عناصر کنترلی، مکمل عمل کننده ها می باشند که هوای فشرده مورد نیاز جهت به حرکت در آمدن عمل کننده ها را به آنها منتقل می کنند. به طور معمول این شیر (شیر کنترل کننده) مستقیماً به منبع هوای فشرده اصلی وصل می شود و به منظور حداقل کردن افت فشار ناشی از مقاومت مسیر، در نزدیکی عمل کننده نصب می شود.

شکل ۱۲-۲- عمل کننده با عنصر کنترلی



عمل کننده ها را باز هم می توان به گروههای کوچکتری تقسیم بندی نمود:

عمل کننده های خطی

سیلندر یک طرفه

سیلندر دو طرفه

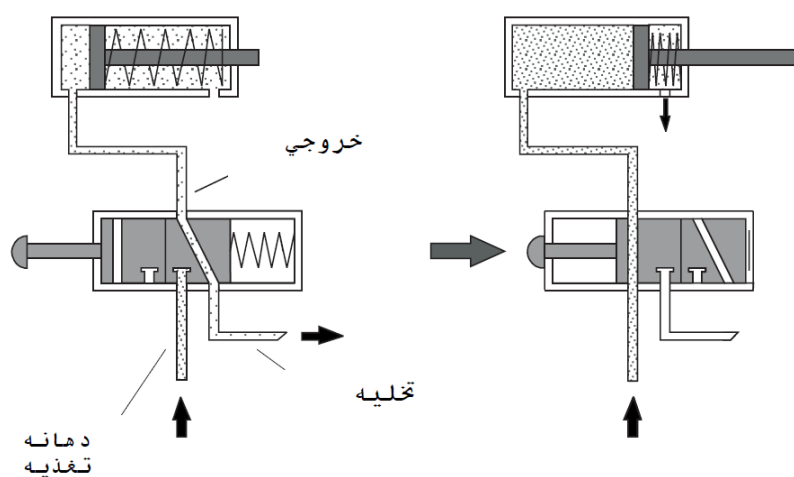
عمل کننده های دورانی

موتورهای پنیوماتیکی

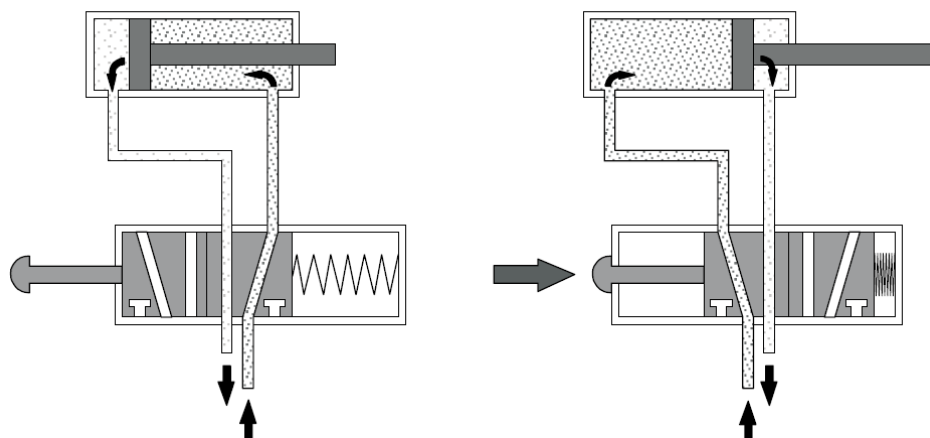
عمل کننده های دورانی



کنترل یک سیلندر یک طرفه



کنترل یک سیلندر دوطرفه



علائم عمل کننده ها

Symbol	Description
	Single-acting cylinder, single piston rod, return stroke by external force
	Double-acting cylinder, single piston rod
	Double-acting cylinder, non-rotating single piston rod
	Double-acting cylinder, rear boss mount, single piston rod
	Double-acting air-hydro cylinder, single piston rod
	Double-acting cylinder with double piston rod
	Double-acting cylinder with double non rotating piston rod
	Double-acting air-hydro cylinder with double piston rod
	Single-acting cylinder, single piston rod, return stroke by spring
	Single-acting cylinder, non rotating single piston rod, return stroke by spring

Symbol	Description
	Single-acting cylinder, single piston rod, stroke by spring, return stroke by air pressure
	Single-acting cylinder, non rotating single piston rod, stroke by spring, return stroke by air pressure
	Double-acting cylinder with cushioning adjustable at both ends, single piston rod
	Double-acting cylinder with cushioning adjustable at both ends, double piston rod
	Magnetically coupled rodless cylinder
	Double-acting cylinder, single piston rod, with built in speed controller
	Double-acting cylinder, double piston rod, with built in speed controller
	Stroke reading cylinder, single piston rod
	Stroke reading cylinder with brake, single piston rod
	Double-acting cylinder with lock, single piston rod

خلاصه مطالب

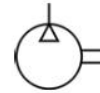
کمپرسور: تامین هوای فشرده مورد نیاز



(a) Compressor used in schools



(b) Compressor used in laboratories

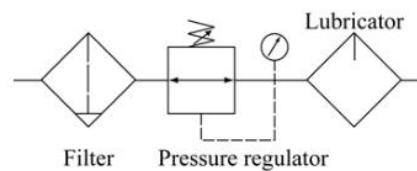


(c) Pneumatic symbol of a compressor

واحد مراقبت: تولید هوای فشرده با کیفیت بالا، تنظیم فشار خروجی و روغن پاشی



(a) Pressure regulating component

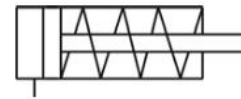


(b) Pneumatic symbols of the pneumatic components within a pressure regulating component

سیلندر یکطرفه: با ورودی هوای فشرده سیلندر حرکت کرده و در صورت قطع هوای فشرده، سیلندر با نیروی فنر به موضع اولیه خود باز می گردد.



Fig. 5 (a) Single acting cylinder

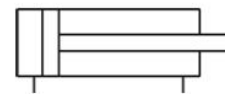


(b) Pneumatic symbol of a single acting cylinder

سیلندر دوطرفه: در این سیلندر برای بازگرداندن سیلندر به مکان اولیه به جای نیروی فنر از نیروی فشار هوای فشرده بر سطح دیگر استفاده می شود.



Fig. 7 (a) Double acting cylinder



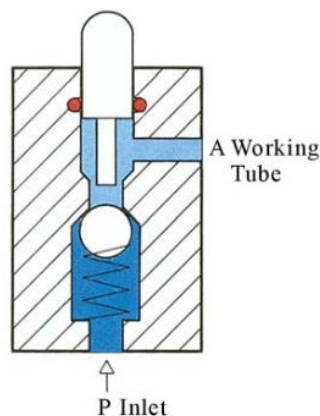
(b) Pneumatic symbol of a double acting cylinder

انواع شیرهای راه دهنده:

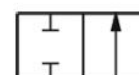
2/2 Directional control valve



Fig. 9 (a) 2/2 directional control valve



(b) Cross section



(c) Pneumatic symbol of a 2/2 directional control

The structure of a 2/2 directional control valve is very simple. It uses the thrust from the spring to open and close the valve, stopping compressed air from flowing towards working tube 'A' from air inlet 'P'. When a force is applied to the control axis, the valve will be pushed open, connecting 'P' with 'A' (Fig. 9). The force applied to the control axis has to overcome both air pressure and the repulsive force of the spring. The control valve can be driven manually or mechanically, and restored to its original position by the spring.

(ii) Flow control valve

A flow control valve is formed by a non-return valve and a variable throttle (Fig. 14).



(ii) 3/2 Directional control valve

A 3/2 directional control valve can be used to control a single acting cylinder (Fig. 10). The open valves in the middle will close until 'P' and 'A' are connected together. Then another valve will open the sealed base between 'A' and 'R' (exhaust). The valves can be driven manually, mechanically, electrically or pneumatically. 3/2 directional control valves can further be divided into two classes: Normally open type (N.O.) and normally closed type (N.C.) (Fig. 11).

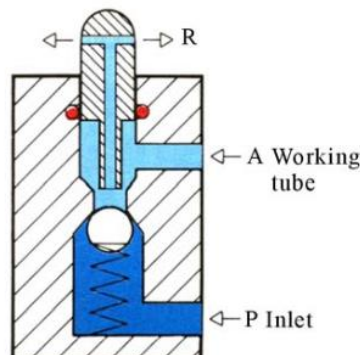


Fig. 10 (a) 3/2 directional control valve

(b) Cross section



(a) Normally closed type



(b) Normally open type

Fig. 11 Pneumatic symbols

(iii) 5/2 Directional control valve

When a pressure pulse is input into the pressure control port 'P', the spool will move to the left, connecting inlet 'P' and work passage 'B'. Work passage 'A' will then make a release of air through 'R1' and 'R2'. The directional valves will remain in this operational position until signals of the contrary are received. Therefore, this type of directional control valves is said to have the function of 'memory'.

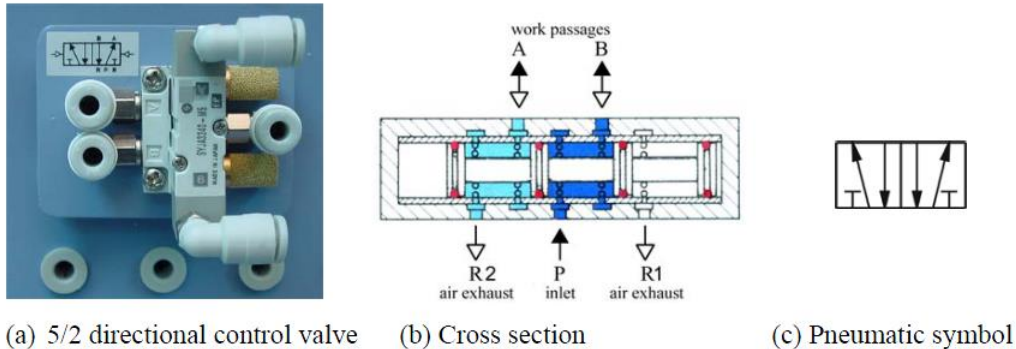


Fig. 12 5/2 directional control valve

(c) Control valve

A control valve is a valve that controls the flow of air. Examples include non-return valves, flow control valves, shuttle valves, etc.

(i) Non-return valve

A non-return valve allows air to flow in one direction only. When air flows in the opposite direction, the valve will close. Another name for non-return valve is poppet valve (Fig. 13).

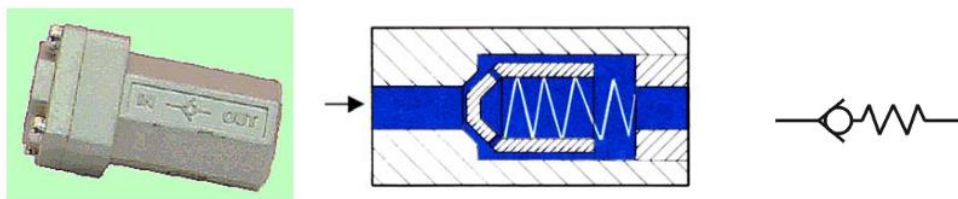


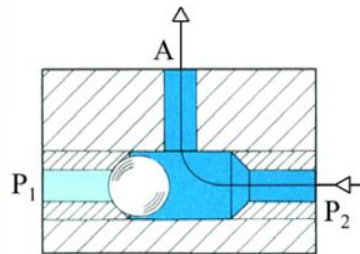
Fig. 13 (a) Non-return valve (b) Cross section (c) Pneumatic symbol

(iii) Shuttle valve

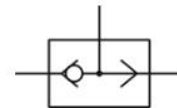
Shuttle valves are also known as double control or single control non-return valves. A shuttle valve has two air inlets ' P_1 ' and ' P_2 ' and one air outlet ' A '. When compressed air enters through ' P_1 ', the sphere will seal and block the other inlet ' P_2 '. Air can then flow from ' P_1 ' to ' A '. When the contrary happens, the sphere will block inlet ' P_1 ', allowing air to flow from ' P_2 ' to ' A ' only.



Fig. 15 (a) Shuttle valve

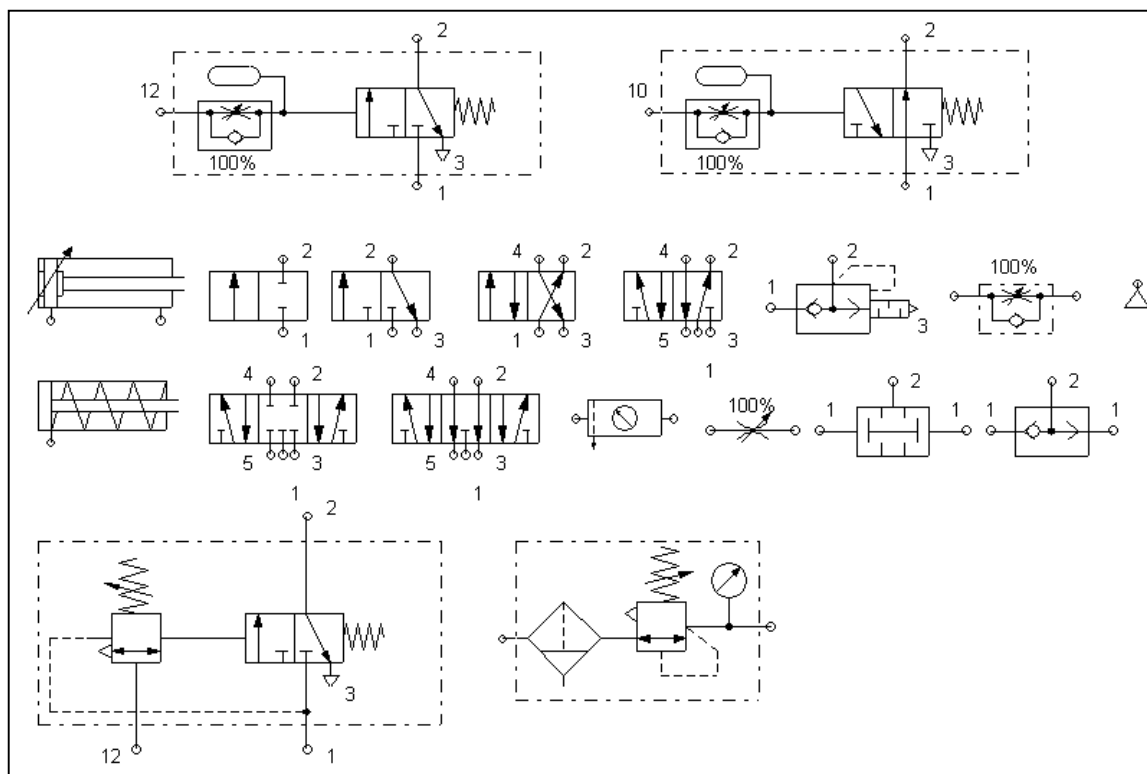


(b) Cross section

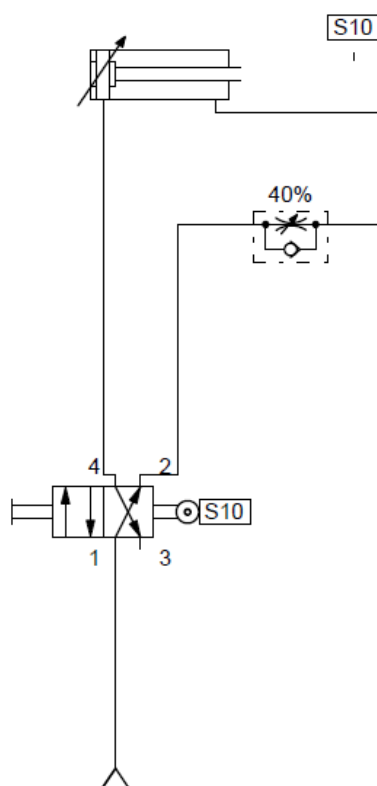


(c) Pneumatic symbol

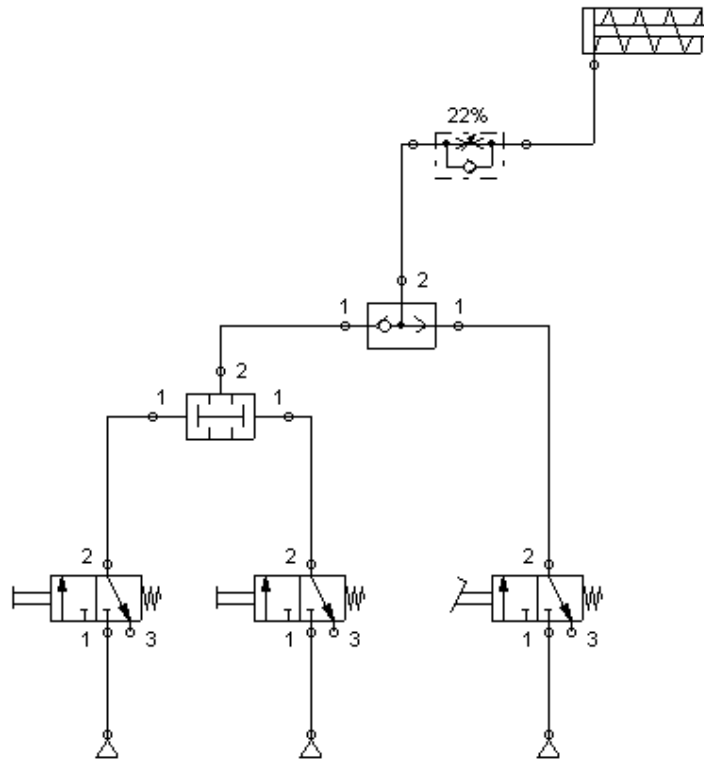
علائم مهم در پنوماتیک



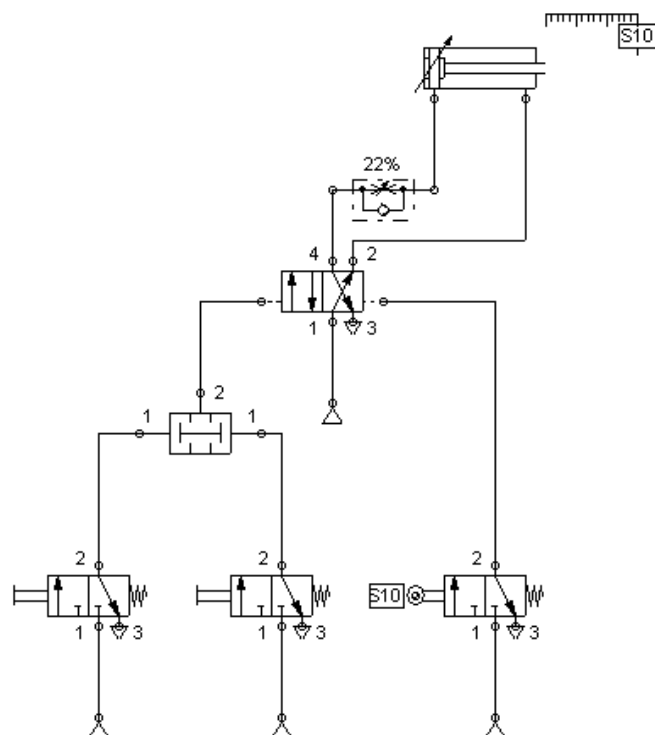
مثال ۱: با زدن باتن شیر، جک با فشار کمپرسور به جلو حرکت کرده و بعد از برخورد با میکروسوییچ S10 به آرامی به مکان اولیه خود بازگردد.



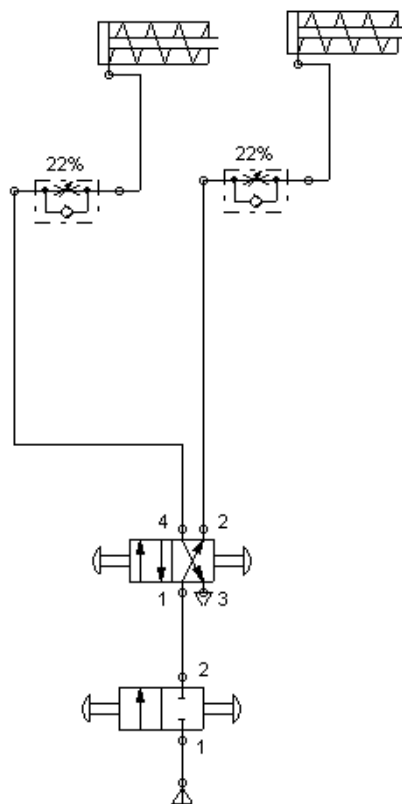
مثال ۲: یک جک توسط دو باتن سری و یک پدال به جلو رفته و برگشت جک به صورت فنر می باشد.



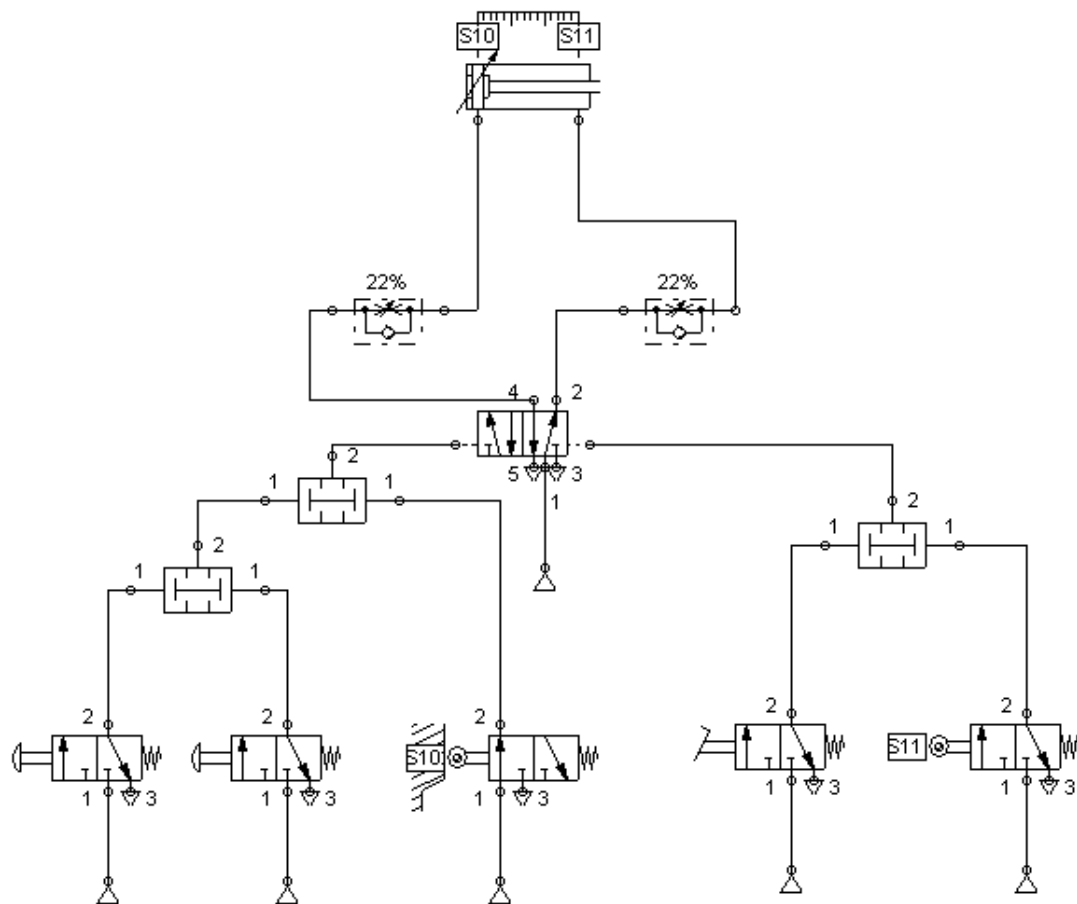
مثال ۳: در مثال ۱ تحریک را توسط دو باتن انجام دهید. در واقع جهت حفاظت، اپراتور می بایست برای یک لحظه هر دو باتن را فشار دهد تا جک به آرامی به جلو حرکت کند. بعد از برخورد به میکروسوییچ S10 به صورت سریع به مکان اولیه بازگردد.



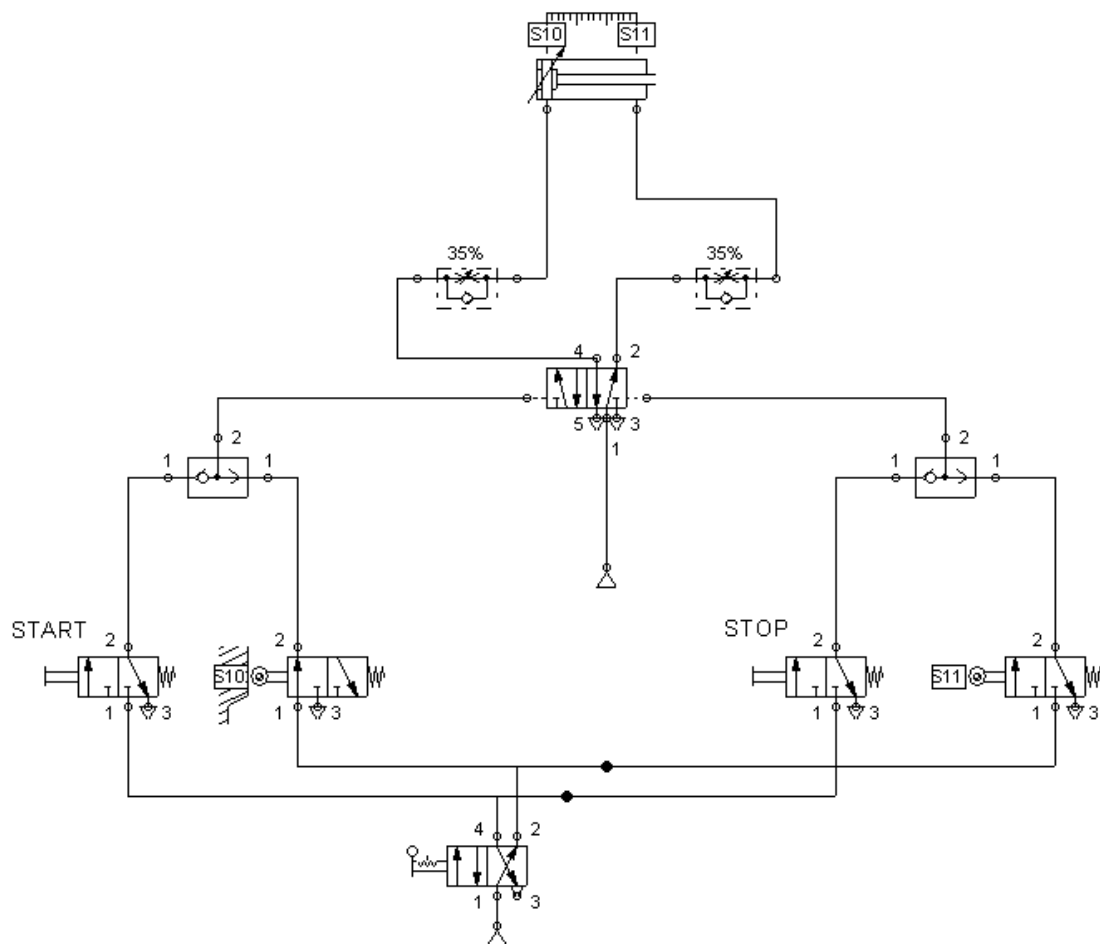
مثال ۴: کنترل دو سیلندر با عملکردی معکوس نسبت به یکدیگر



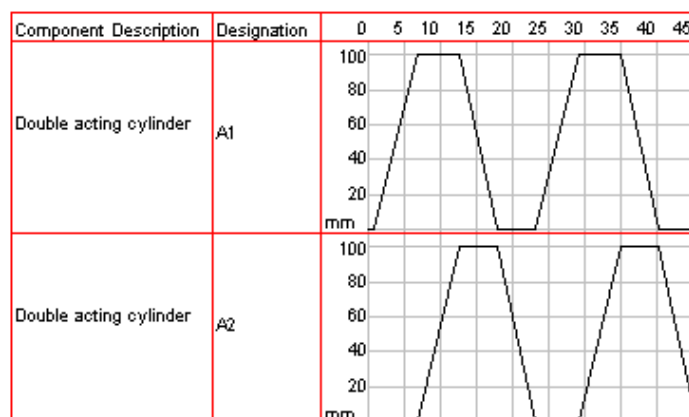
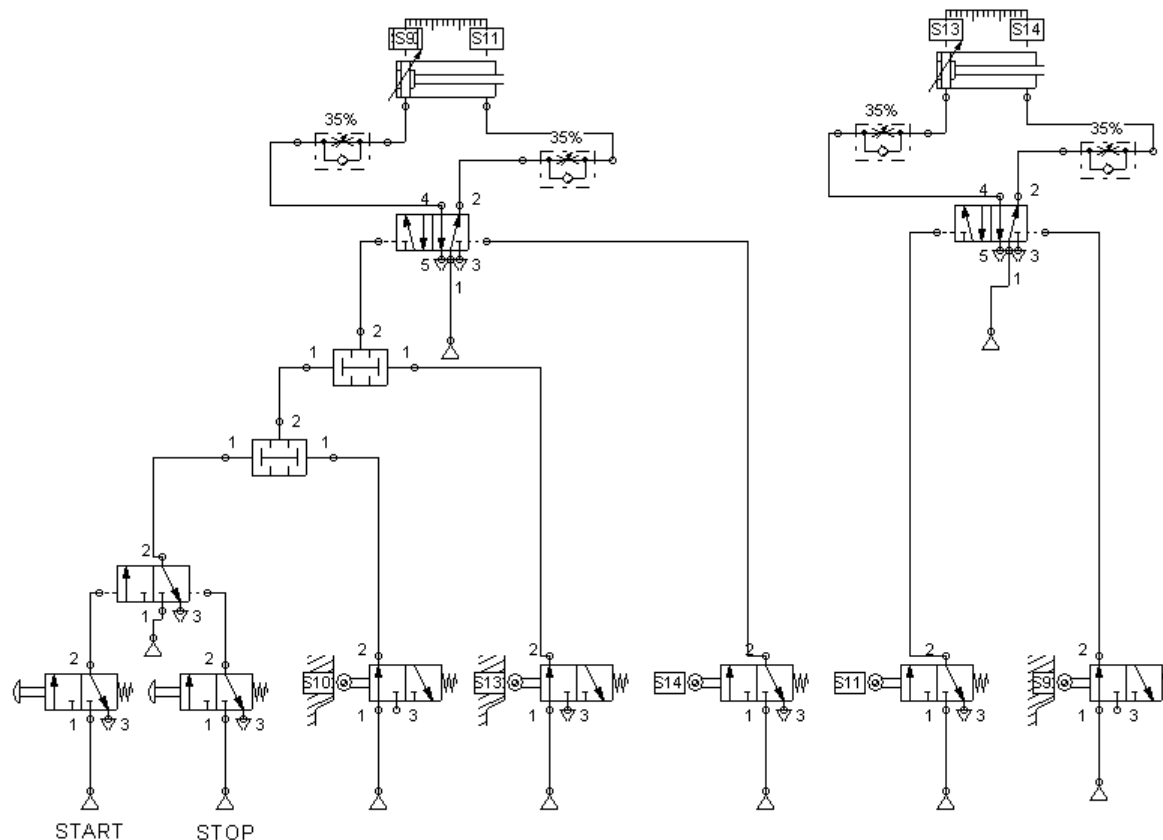
مثال ۵: در این مثال با بودن سیلندر در موضع اولیه، با فشردن دو باتن استارت به صورت لحظه ای، جک شروع به حرکت می کند. با رسیدن جک به موضع انتهایی، تا مادامی که پدال فعال نشود، جک به عقب باز نگردد. اگر در حین حرکت جک دو باتن استارت فعال شوند، هیچ تاثیری در مدار نداشته باشد.



مثال ۶: در این مثال توسط شیر با تحریک اهرمی وضعیت اتومات و یا دستی کار کردن سیلندر مشخص می شود. در حالت ابتدایی سیلندر به صورت اتومات دارای حرکت رفت و برگشت پیوسته می باشد. با رفتن به حالت دستی فرامین جلو و عقب رفتن جک به صورت دستی توسط باتن های استارت و استپ می باشد.

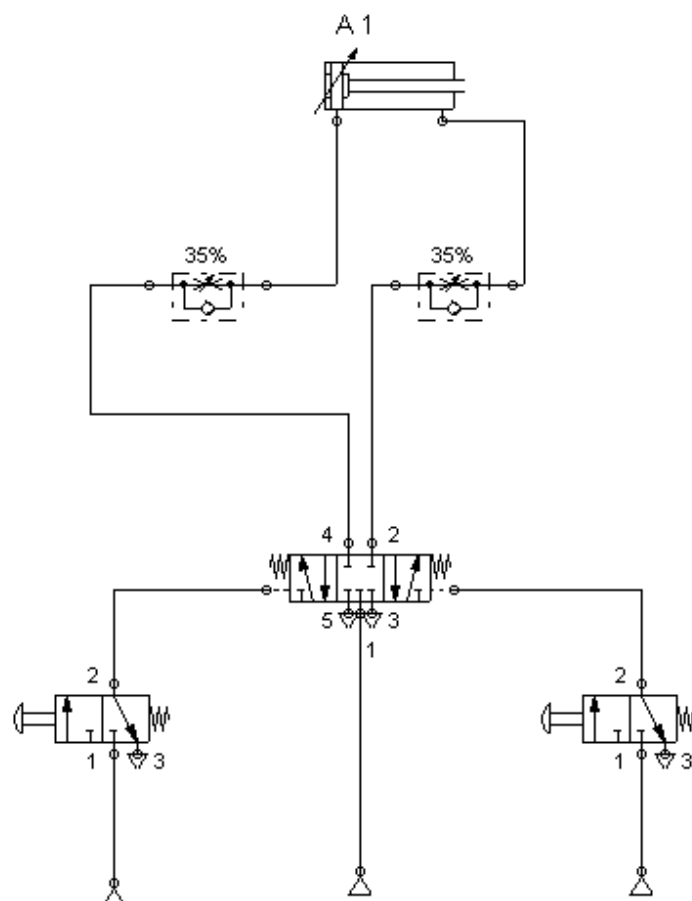


مثال ۷: در این مثال با فشردن باتن استارت جک اول شروع به حرکت نموده و با رسیدن به موضع انتهایی، جک دوم نیز حرکت رو به جلوی خود را آغاز می کند. با رسیدن جک دوم به موضع انتهایی، جک اول به مکان اولیه بازگشته و سپس جک دوم به موضع اولیه باز می گردد. با بازگشت جک دوم به مکان ابتدایی این روند مجدداً با حرکت جک اول تکرار می شود. با فشردن باتن استپ، هر دو جک به مکان اولیه باز می گردند و عملکرد پروسه متوقف می شود. در موضع ابتدایی جک ۱ دو میکروسوییچ S9 و S10 وجود دارد که عامل های تحریک دو شیر برقی می باشند.



مثال ۸: کنترل یک جک در مسیر رفت و برگشت

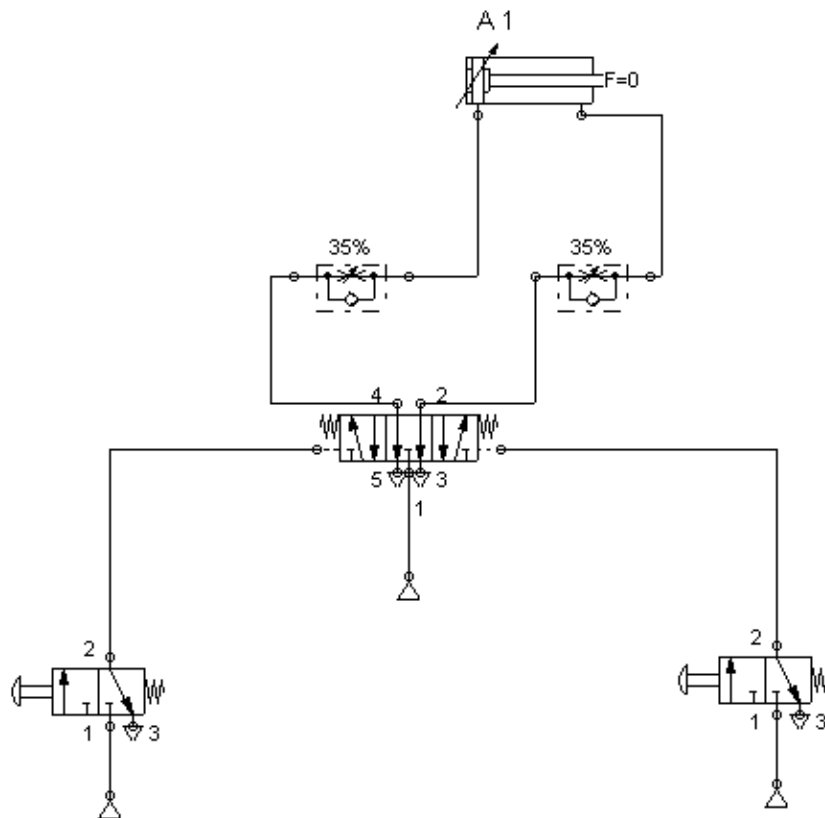
توسط یک شیر ۵/۳ با موضع وسط بسته می خواهیم جک را در هر نقطه ای در میانه نگه داریم . موضع وسط این شیر بسته می باشد.



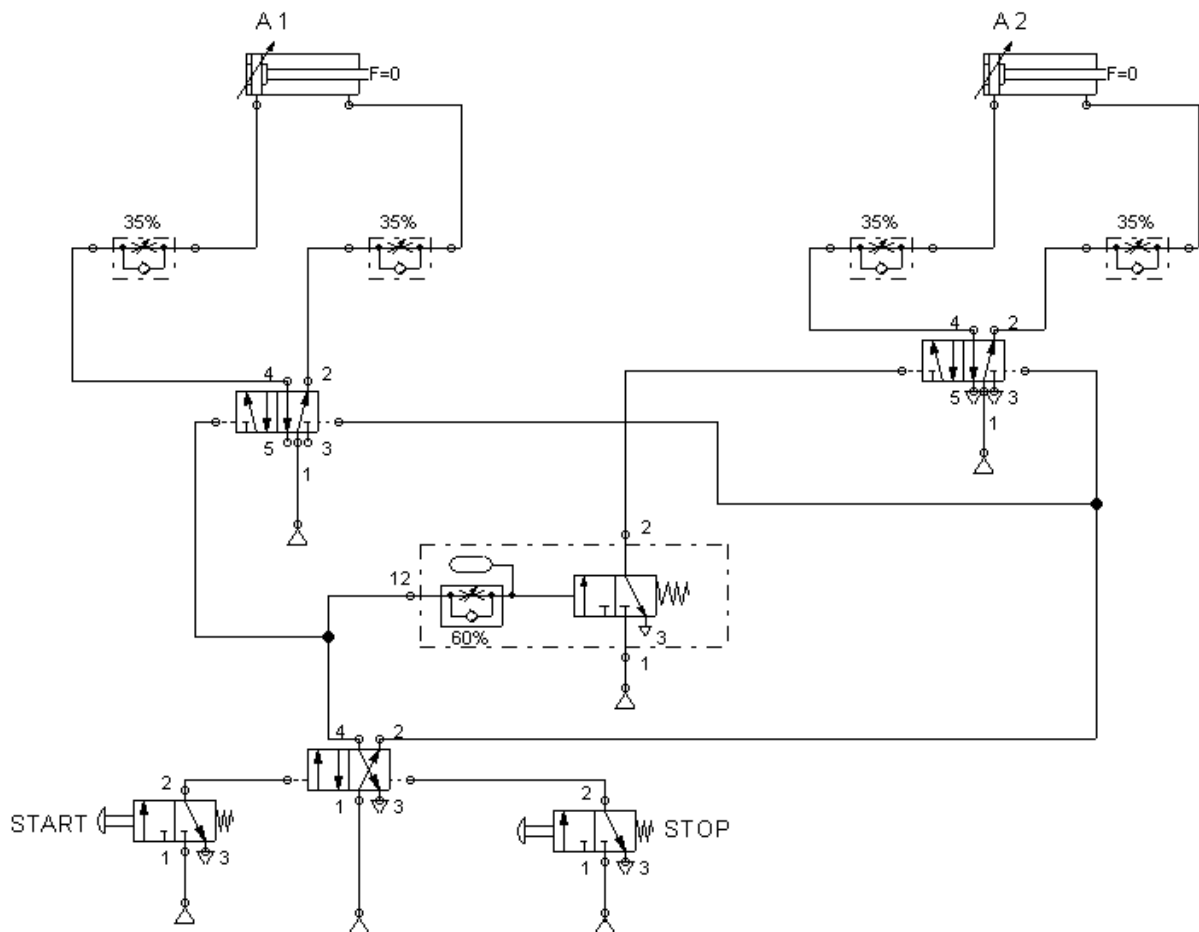
[illegible]

مثال ۱۰: کنترل یک سیلندر با شیر موضع وسط شناور

در این مثال توسط یک شیر ۵/۳ می خواهیم کنترل یک سیلندر را در کل مسیر ابتدا تا انتها انجام دهیم. تفاوت این مثال با مثال ۸ در این است که در این پروسه زمانی که شیر در موضع وسط می باشد، هیچ فشاری در ورودی و خروجی جک نبوده و جک را به راحتی می توان به دست به جلو و عقب برد.

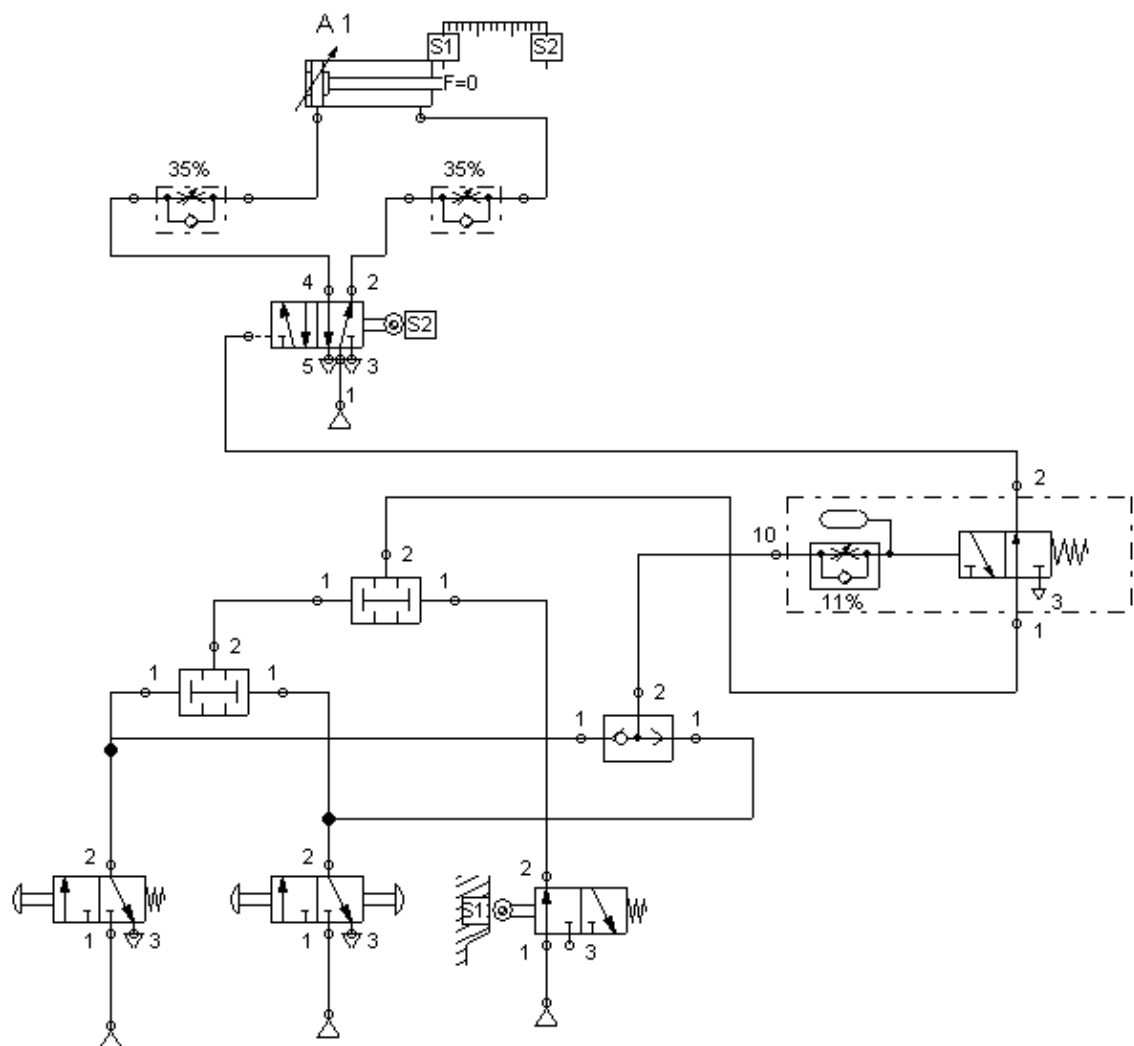


در این مثال با فشردن باتن استارت جک اول بلافاصله شروع به حرکت نموده و با اندکی تاخیر جک دوم نیز حرکت می کند. با فشردن شاستی استپ هر دو جک به موقعیت اولیه باز گردند.

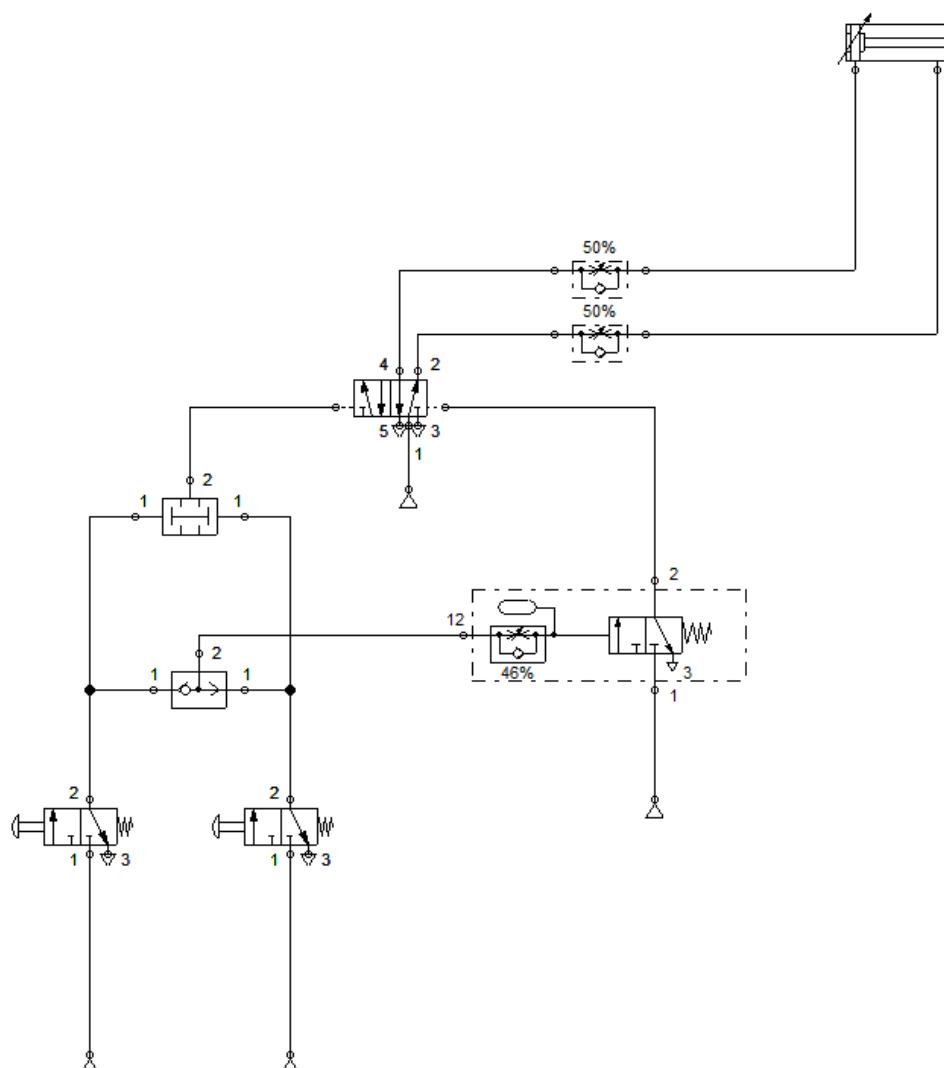


مثال ۱۳

در این مثال جهت کنترل یک پرس ضربه ای می بایست هر دو دست اپراتور در گیر فشردن باتن ها تعبیه شده باشد. برای جلوگیری از یکسره کردن یکی از باتن ها جهت راحتی کار می بایست سیستم را به گونه ای طراحی کنیم که در صورتی که یکی از باتن ها توسط یک چسب برای همیشه یکسره شد سیستم توسط باتن بعدی کار نکند. در واقع اپراتور مجبور می باشد که باتن را به حالت اولیه خود بازگرداند. در این مثال زمانی که یک لحظه هر دو باتن فشرده شوند جک پایین آمده و پس از رسیدن به میکروسوییچ در انتهای مسیر به صورت اتوماتیک به موقعیت اولیه خود باز می گردد. در این مثال فشردن یک لحظه باتن ها کافی می باشد. تایمر استفاده شده در این مثال تایمر تاخیر در قطع می باشد.

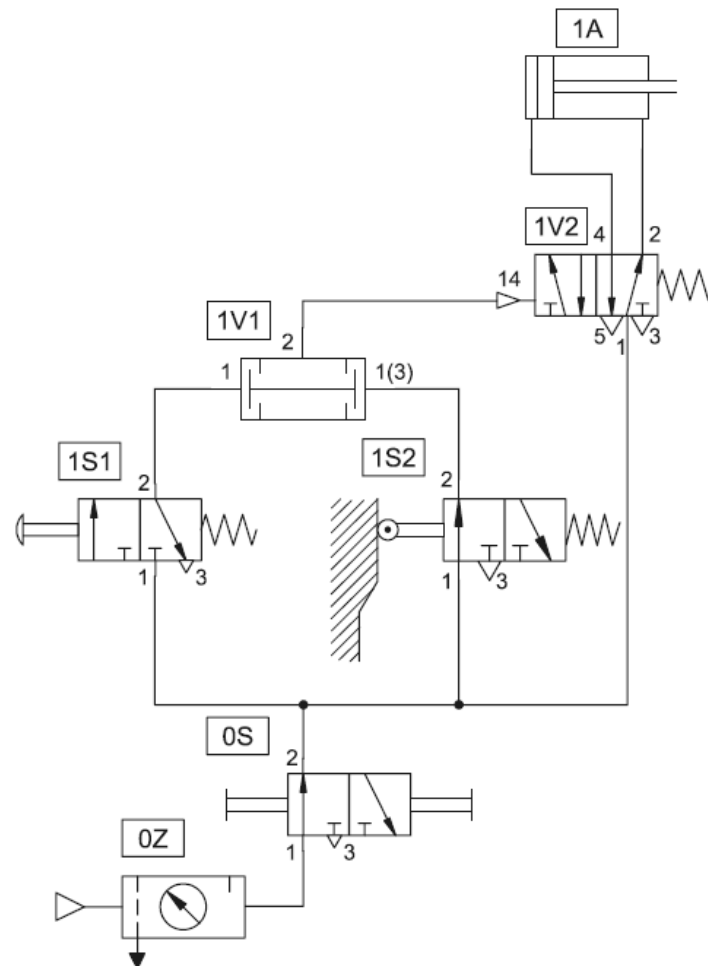


مثال ۱۴: مدار ایمنی استارت دو دستی

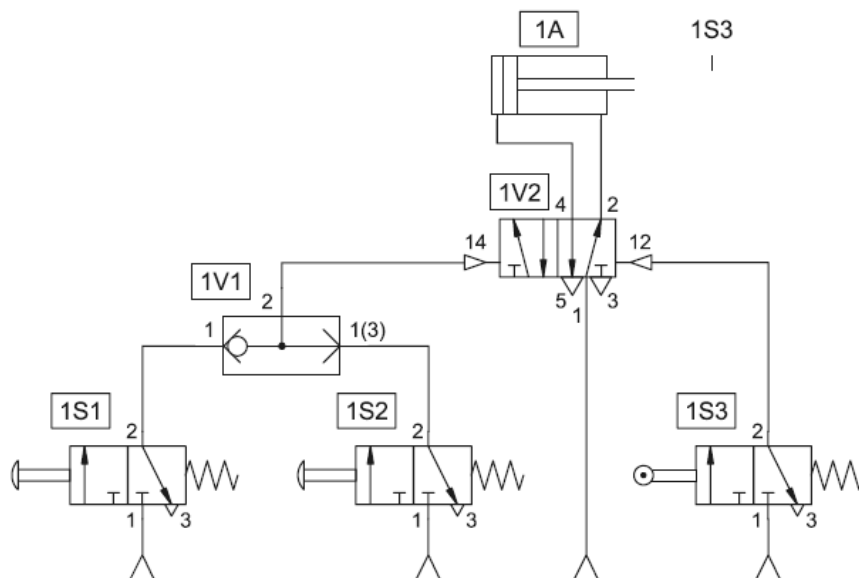


تمرین ۱

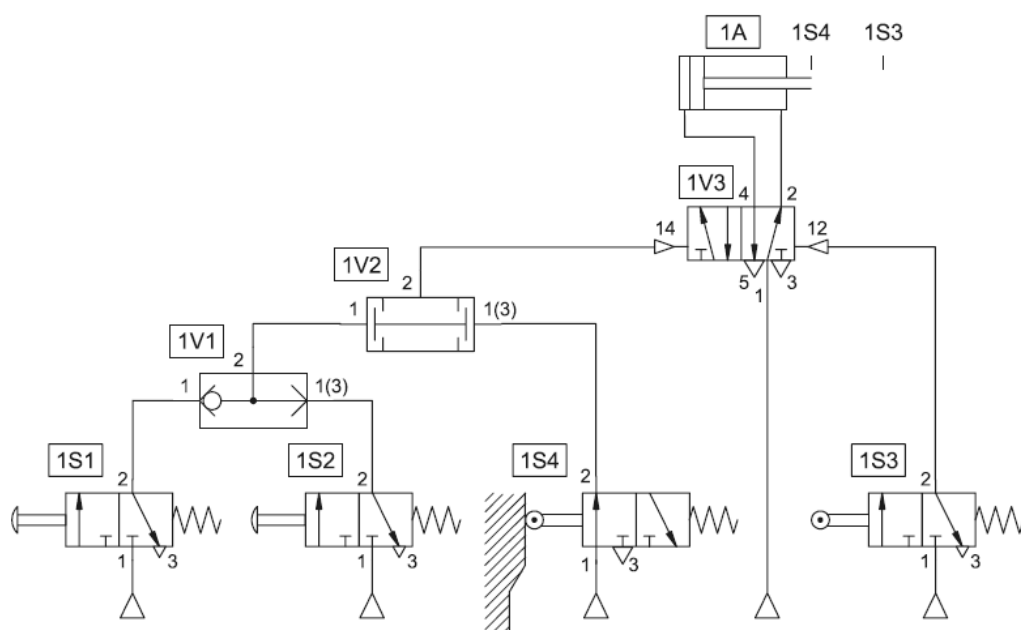
عملکرد مدار زیر را بررسی کنید.



عملکرد مدار زیر را بررسی کنید.

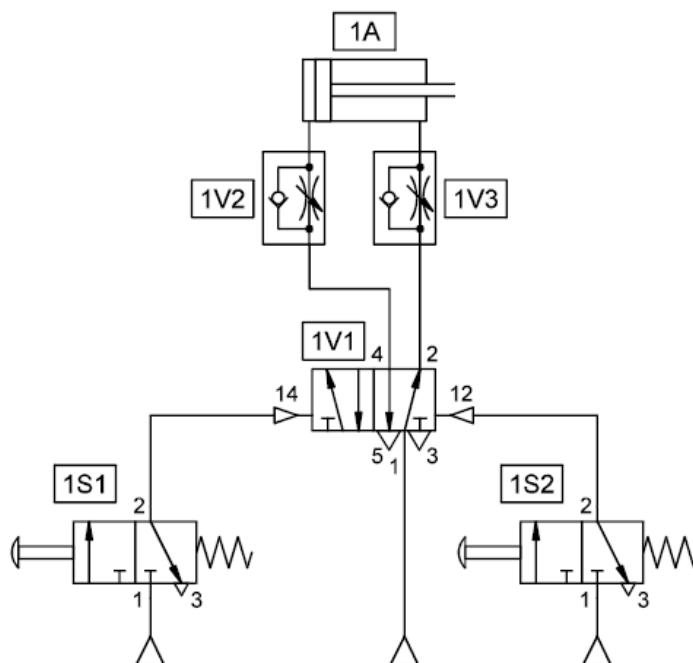


عملکرد مدار زیر را بررسی کنید.



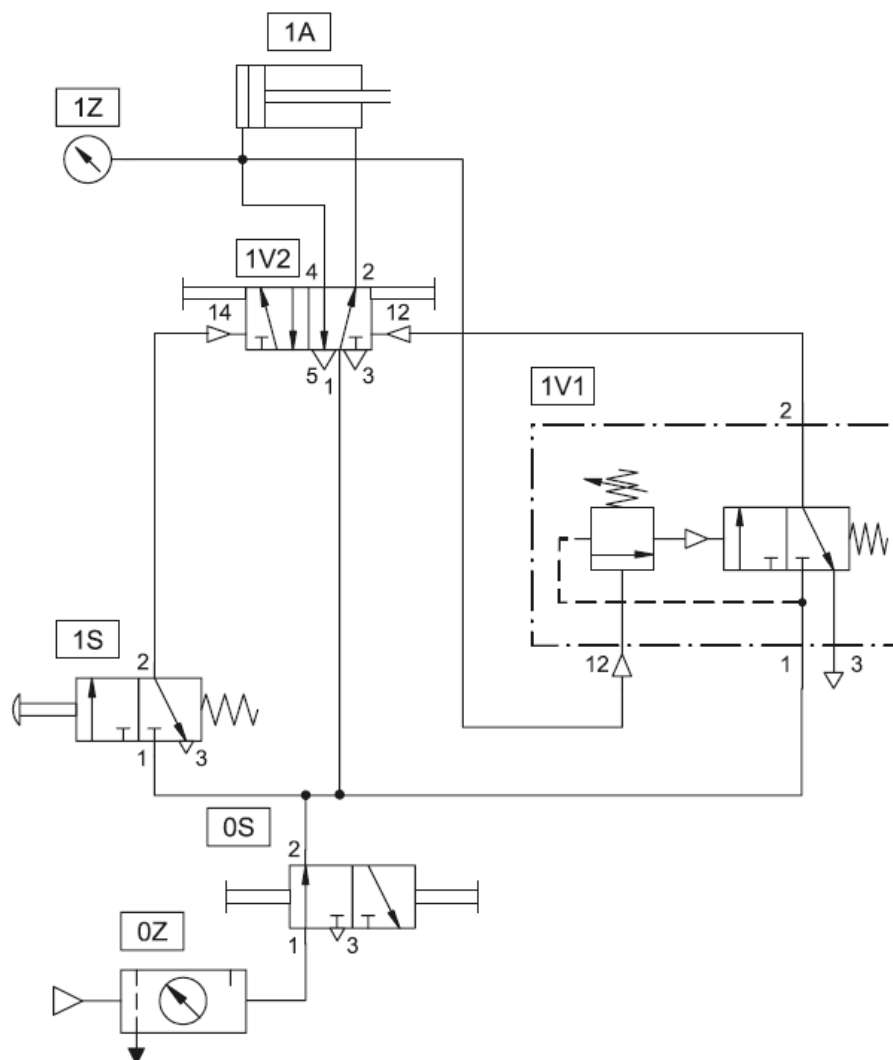
تمرین ۵

عملکرد مدار زیر را بررسی کنید.



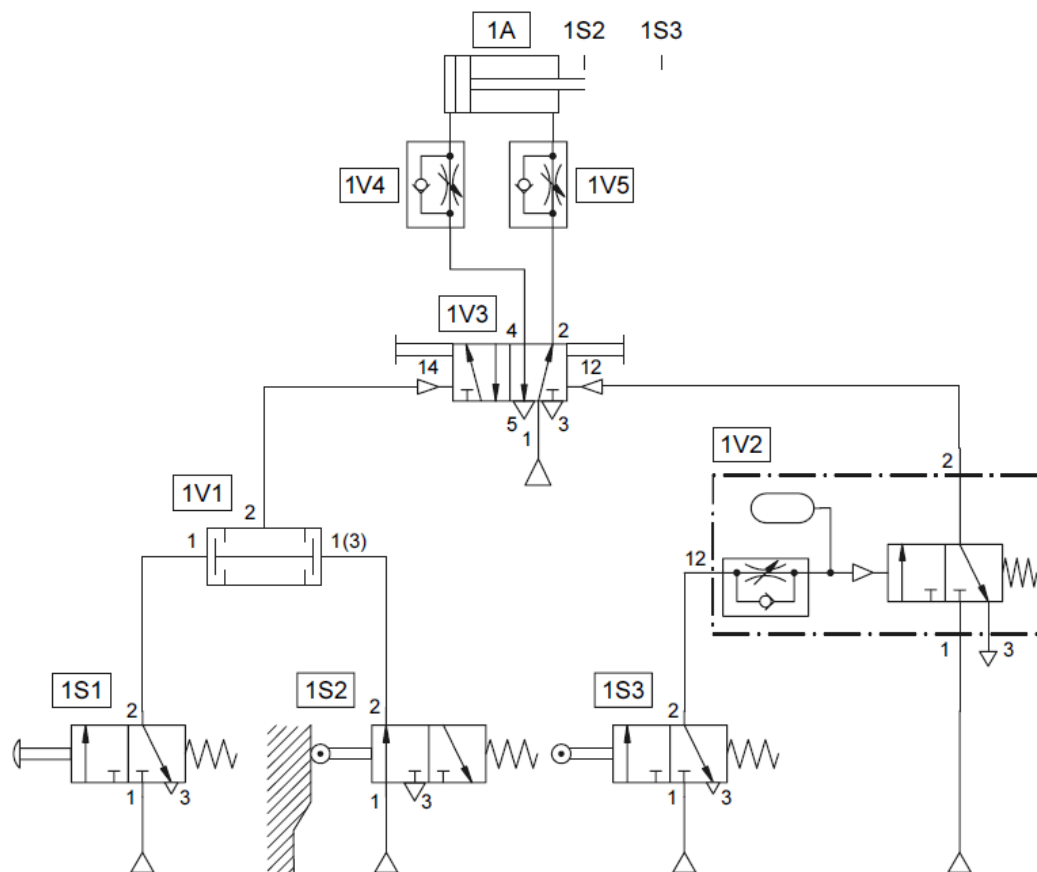
تمرین ۶

عملکرد مدار زیر را بررسی کنید.



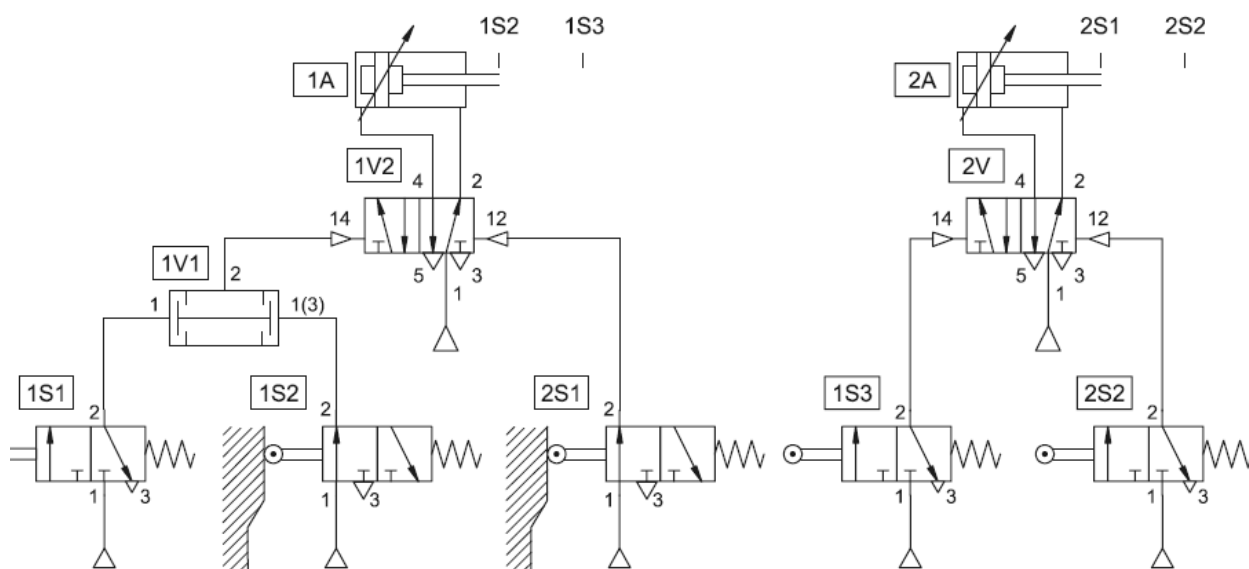
تمرین ۷

عملکرد مدار زیر را بررسی کنید.



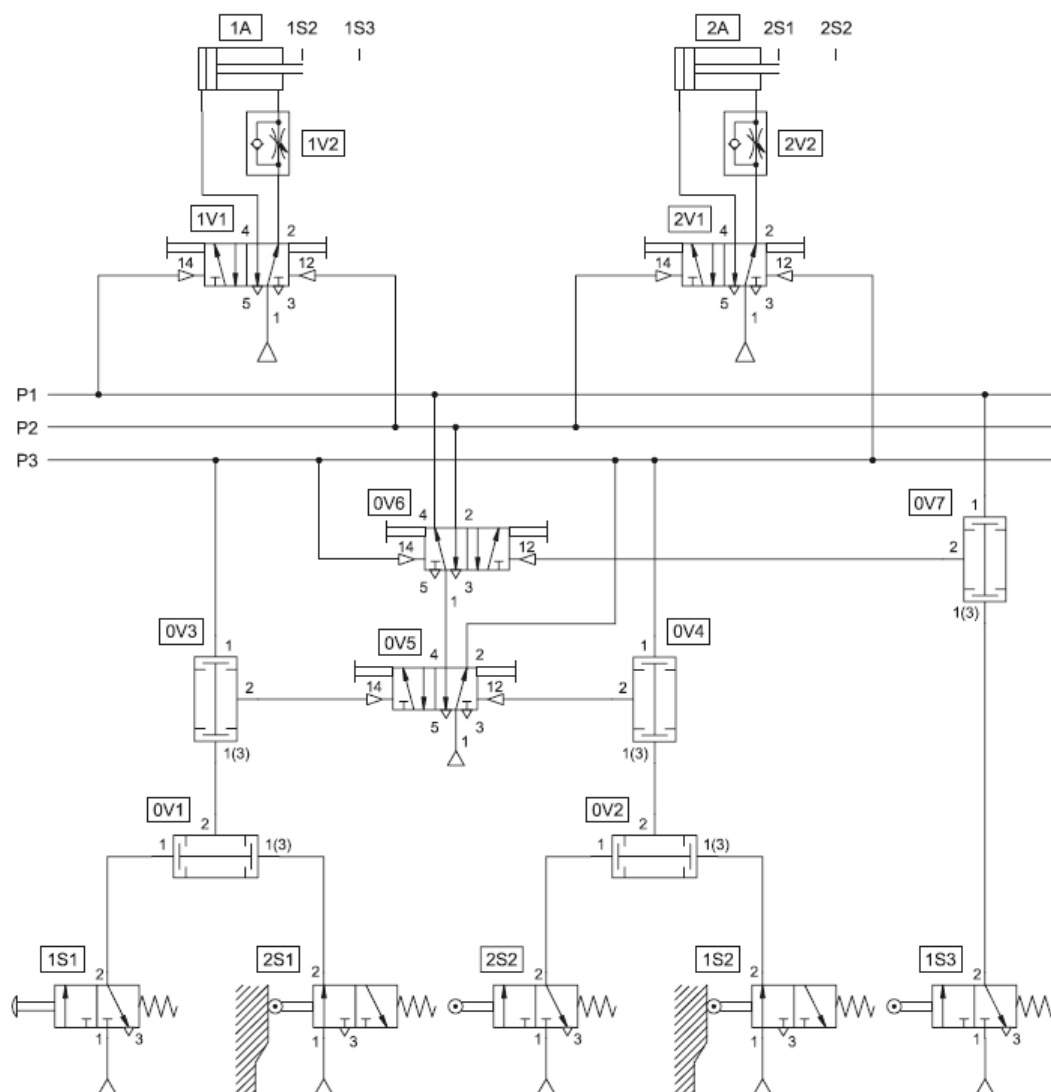
تمرین ۸

عملکرد مدار زیر را بررسی کنید.



تمرین ۹

عملکرد مدار زیر را بررسی کنید.



آشنایی با درایوهای صنعتی

امروزه استفاده از اینورترها در مراکز صنعتی بیش از پیش ضرورت یافته است. از آنجایی که درایوها نیز می توانند به عنوان یک تجهیز در خروجی آنالوگ PLC به کار برده شوند، لازم است که اطلاعاتی کلی در رابطه با اجزای یک درایو و نحوه عملکرد آن در اختیار شما دوستان عزیز قرار گیرد. اینورترها همان کنترل دور موتورهای می باشند که توسط آنها می توان بسیاری از پارامترهای موتور را تحت کنترل گرفت.

کاربردهای اینورتر

*انجام تنظیمات مورد نیاز جهت تنظیم سرعت یک موتور

*انجام تنظیمات مورد نیاز جهت اعمال گشتاور به یک موتور

*تغییر جهت گردش موتور بدون نیاز به مدارات کنتاکتوری

*افزایش راندمان و صرفه جویی در انرژی

در گذشته نه چندان دور به جای استفاده از اینورتر از موتورهای چند سرعتی استفاده می شد که این موتورها نیز دارای معایبی بودند. از جمله این معایب می توان به جریان راه اندازی بالا، کنترل در سرعت های محدود و مدت زمان طولانی جهت افزایش یا کاهش سرعت.

درایوهای AC برق ورودی را گرفته و به ولتاژ و فرکانس قابل تنظیم برای کنترل موتورهای تبدیل می کنند.

به طور کلی سه نوع اینورتر وجود دارد:

۱- مبدل ولتاژ متغیر VVI

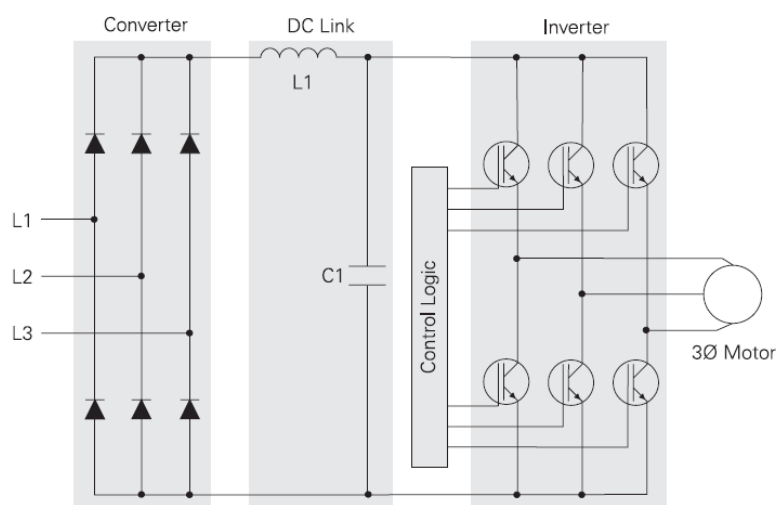
۲- مبدل منبع جریان CSI

۳- مدولاسیون پهنای پالس PWM

تمامی درایوهای AC، ولتاژ AC را به DC تبدیل کرده و با روش های کلیدزنی خاص، ولتاژ DC را به ولتاژ و فرکانس متغیر AC تبدیل می کنند.

اینورترهای مدولاسیون عرض پالس PWM

درایوهای مدولاسیون عرض پالس همانند میکرومستر و مستردرایو زیمنس شکل موج جریان سیسنوسی تری را نسبت به سایر اینورترها فراهم می کنند و بازده و کارایی بالاتری بیشتری نیز دارند. یک درایو PWM پایه ای شامل کانورتر، لینک DC، لاجیک کنترل و اینورتر می باشد. بخش مبدل شامل دیودهای یکسوکننده سه فاز ثابت برای تبدیل ولتاژ AC به DC. سلف L1 و خازن C1 نیز به عنوان صافی خط DC استفاده می شوند. مقدار جبران DC در حدود 1.35 برابر ولتاژ خط به خط ولتاژ تغذیه است. این مقدار در حدود 650VAC می باشد.



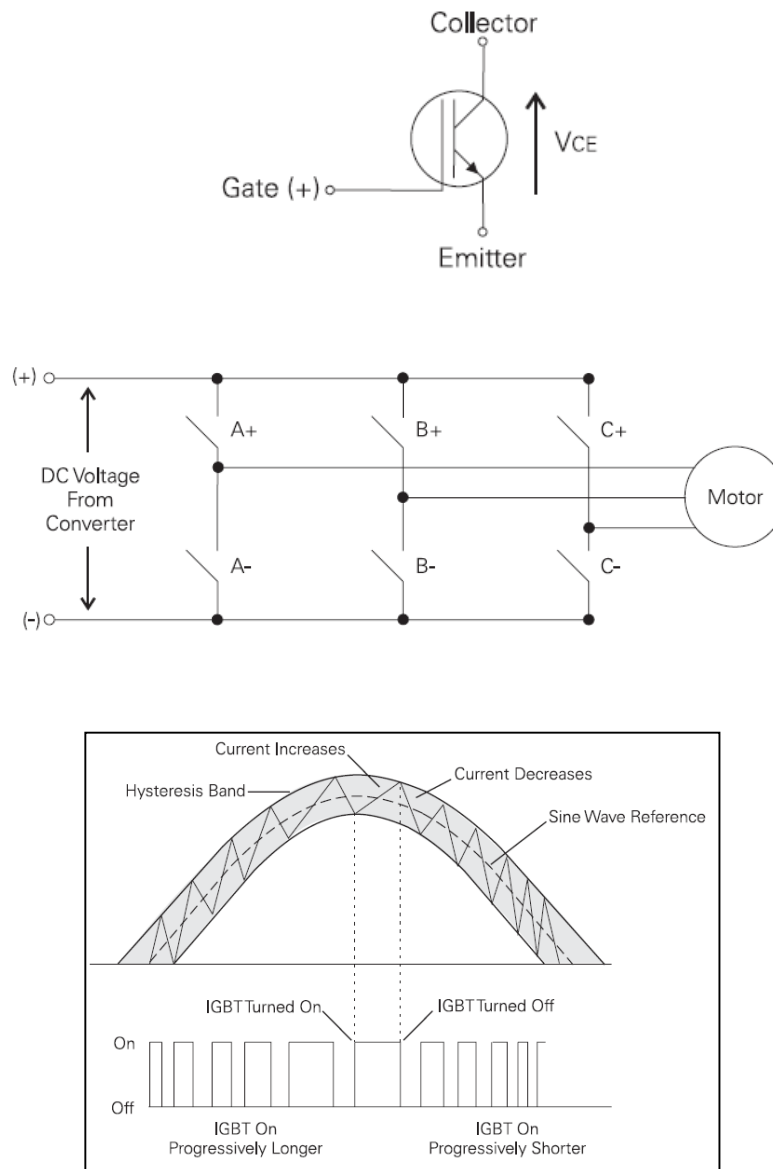
لاجیک کنترل و اینورتر

خروجی ولتاژ و فرکانس موتور توسط واحد کنترل و اینورتر کنترل می شود. واحد اینورتر شامل ۶ المان سوئیچ می باشد. این المان ها می توانند ترانزیستورها، MOSFET و IGBT ها باشند. واحد کنترل نیز جهت کنترل IGBT ها برای روشن و خاموش کردن آنها استفاده تا ولتاژ و فرکانس متغیر برای موتور تولید شود.

IGBT ها:

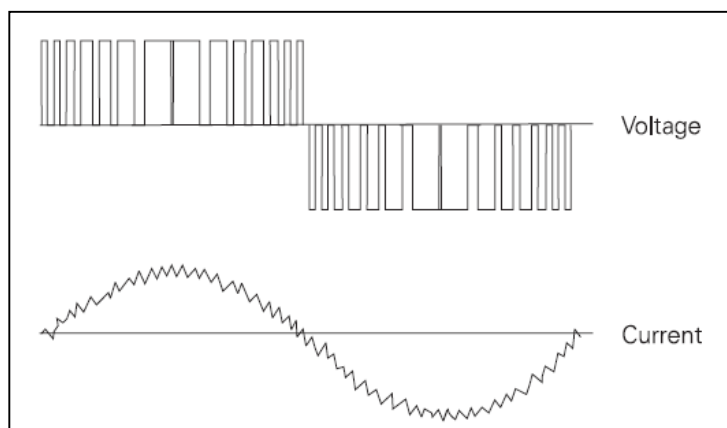
IGBT ها دارای سرعت کلید زنی بالایی می باشند و می توانند در هر ثانیه چند هزار بار باز و بسته شوند. در زمان ۴۰۰ نانو ثانیه روشن و در زمان کمتر از ۵۰۰ نانو ثانیه خاموش می شوند.

یک IGBT شامل پایه های کلکتور ، گیت و امیتر می باشد. وقتی به پایه گیت ولتاژ مثبت اعمال می شود IGBT روشن و یا در واقع کلید بسته می گردد. این کلید با قطع شدن ولتاژ باز می شود. اگر به پایه گیت ولتاژ منفی اعمال شود از خاموش بودن IGBT اطمینان حاصل می گردد. بنابراین عملیات کلید زنی با اعمال ولتاژهای $+15V$ و $-15V$ انجام می شود.



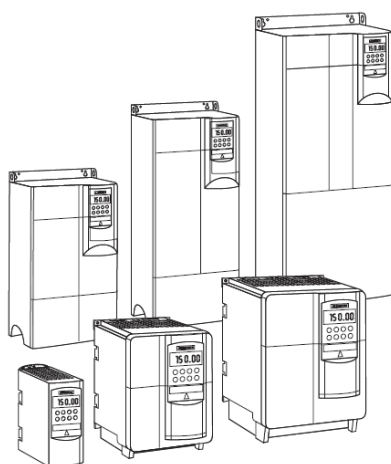
چندین تکنیک برای مدولاسیون عرض پالس PWM وجود دارد. شکل فوق روش PWM را شرح می دهد. در ابتدا IGBT طوری کلید زنی می شود که بتواند برای موتور ولتاژ مثبت تولید کند. (جریان موتور مثبت می باشد).

IGBT تنها برای مدت کوتاهی روشن بوده و جریان کوچکی را برای موتور تولید می کند. سپس برای مدت طولانی تری روشن شده و جریان بالاتری را برای موتور تامین می کند تا اینکه به مقدار پیک برسد. سپس IGBT در زمان کمتری روشن شده و جریان موتور را کاهش می دهد. نیم سیکل منفی نیز به همین ترتیب ساخته می شود.



درايوهای میکرومستر زیمنس MICROMASTER :

شرکت زیمنس طیف وسیعی از درایوهای AC را ارائه می دهد. در گذشته، جهت راه اندازی درایوها نیاز به کار شنا سان متخصص بود. میکرومسترهای زیمنس درایوهایی را با قابلیت تنظیم خودکار، کنترل شار جریان، کنترل برداری و حلقه تنظیم PID ارائه می دهند. میکرومستر دارای یک پروسوسور برنامه ریزی شده می باشد. نصب و راه اندازی این درایوها بسیار ساده و سریع می باشد.



سری های میکرومستر:

MM 410

MM 420

MM 430

MM 440

اجزای درایوهای Micromaster

پانل های اپراتوری

میکرومسترها دارای ۲ پانل اپراتوری مبتدی و پیشرفته می باشند.



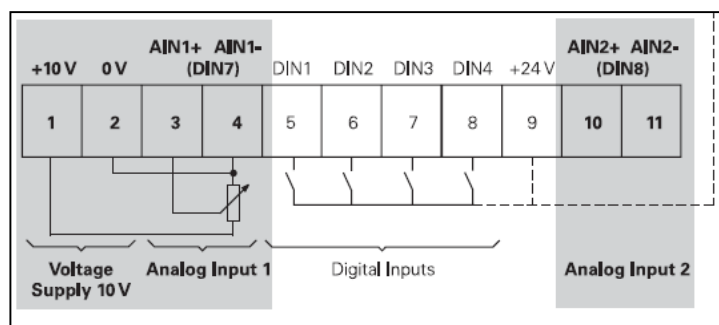
BOP

AOP

ورودی آنالوگ AI

درایوهای MM440 دارای دو ورودی آنالوگ (AIN1 و AIN2) PID می باشند. توسط این ورودیها می توان توسط یک پتانسیومتر سرعت موتور را کنترل نمود. در کنترل به صورت PID کنترل سرعت بسیار مطلوب می باشد. به عنوان مثال در کنترل دما و فشار. سوئیچ های S1 و S2 جهت تعیین نوع سیگنال ورودی (جریان یا ولتاژ) استفاده می شوند. همچنین ورودی های آنالوگ می توانند به عنوان ورودی دیجیتال نیز استفاده شوند. البته تعداد ورودی ها و سایر قسمتهایی که در ادامه بحث می شود برای سایر مدل های میکرومستر متفاوت می باشد. این سیگنال می تواند توسط کارت خروجی آنالوگ PLC تامین شود.

در مثال زیر از ورودی آنالوگ AIN1 به عنوان ورودی آنالوگ مرجع جهت کنترل سرعت موتور بین 0 تا 100% استفاده می شود. ترمینال ۱ دارای ولتاژ داخلی 10VDC ثابت می باشد. ترمینال ۲ ترمینال برگشتی و یا زمین برای تغذیه ۱۰ ولت می باشد. ترمینال ۳ نیز به عنوان ترمینال مثبت ورودی آنالوگ می باشد. بین ترمینال ۲ و ۴ یک جامپر قرار داده شده است.

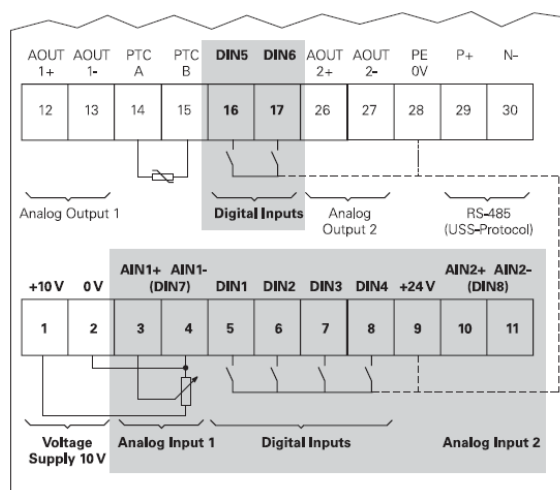


سیگنال ورودی هم می تواند در بازه 0 تا 10V و یا بین 0 تا 20 میلی آمپر باشد.

ورودی های دیجیتال DI

MM440 دارای ۶ ورودی دیجیتال می باشد. همچنین ورودی های آنالوگ نیز می تواند در حالت دیجیتال پیکربندی شوند. سوئیچ ها و کنتاکت ها می توانند بین ورودی 24VDC (ترمینال ۹) و ورودی های دیجیتال قرار گیرند.

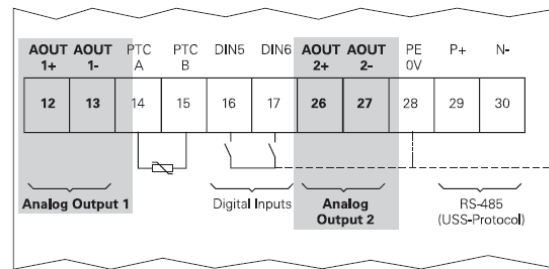
استاندارد کارخانه ورودی DIN1 را به عنوان ورودی استارت/ استپ در نظر گرفته است. DIN2 به عنوان سوئیچ تغییر جهت و DIN3 به عنوان ورودی ری ست خطاها می باشد. سایر توابع مانند PRESET سرعت و JOG نیز می توانند برنامه ریزی شوند.



خروجی آنالوگ AO

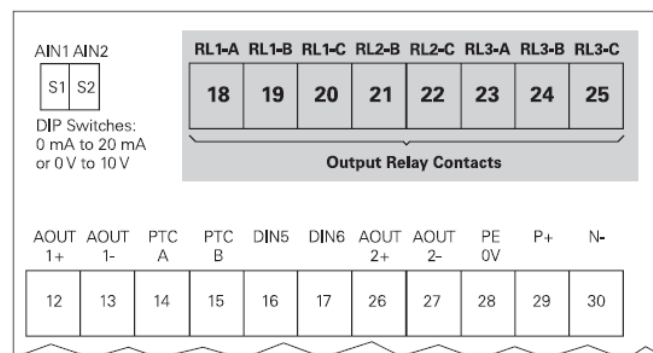
از خروجی آنالوگ جهت نظارت بر فرکانس، SetPoint فرکانس، ولتاژ خط DC، جریان موتور، گشتاور و دور موتور استفاده می شود. خروجی آنالوگ درایو می تواند به عنوان ورودی آنالوگ یک PLC باشد. در این صورت به راحتی می توان یک سیستم حلقه بسته طراحی نمود.

MM440 دارای دو خروجی AOUT1 و AOUT2 می باشد.



خروجی دیجیتال DO

در MM440 سه خروجی دیجیتال قابل برنامه ریزی وجود دارد. (RL1، RL2 و RL3). توسط خروجی های رله ای می توان شرایط مختلف کاری را از جمله فعال بودن درایو، خطاهای رخ داده برای درایو، فرکانس صفر یا نقطه مینیمم برای درایو را نشان داد.



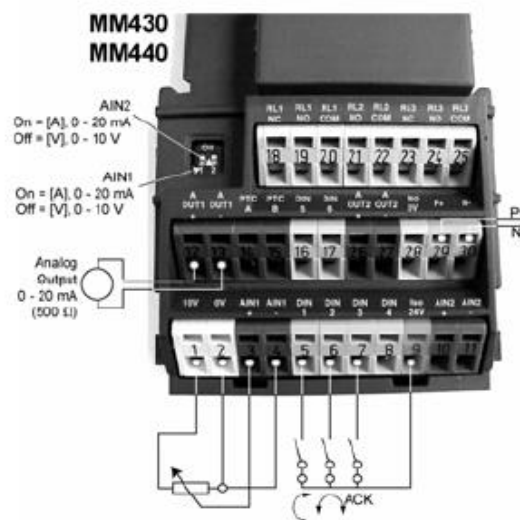
نکته

تعیین منبع فرمان و همچنین منبع تنظیم SetPoint برای درایو توسط پارامترهای درایو تعیین می شود. اگر منبع تنظیم SetPoint، ورودی های آنالوگ تعیین شود در این صورت توسط یک پتانسیومتر خارجی و یا کارت خروجی آنالوگ PLC می توان دور موتور را کنترل نمود.

یادآوری

ورودی آنالوگ در اینورترهای زیمنس

اکثر اینورترهای شرکت های مختلف دارای ورودی آنالوگ می باشند. توسط ورودی آنالوگ می توان یک ولتاژ 0 تا 10 ولت DC به اینورتر تزریق نمود که اینورتر نیز با دریافت این ولتاژ می تواند سرعت یک موتور را تغییر دهد. این ولتاژ می تواند از یک پتانسیومتر و یا از کارت خروجی آنالوگ یک PLC ارسال شود.

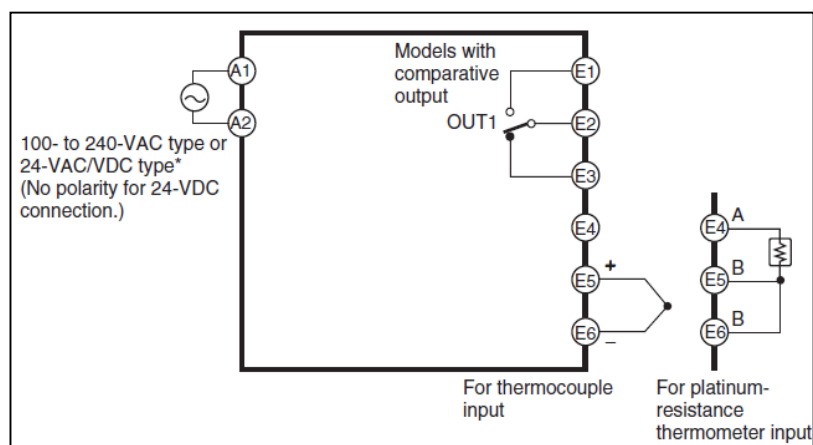


آشنایی با سیگنال های آنالوگ

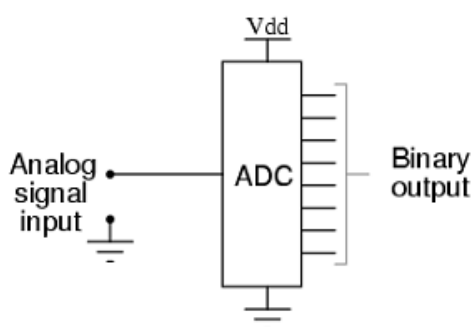
در صنعت امروز کمیت هایی مانند دما ، فشار ، وزن و سرعت به عنوان کمیت های آنالوگ به حساب می آیند. تجهیزاتی که قادر به اندازه گیری این کمیت ها می باشند نیز به عنوان تجهیزات آنالوگ محسوب می شوند. در پروسه های صنعتی معمولاً برای هر یک از موارد فوق یک کنترلر به عنوان اندازه گیر تعبیه شده است. این کنترلرها به صورت مستقیم قادر به اندازه گیری کمیت توسط سنسور مربوطه می باشند. به عنوان مثال کنترلرهای دما به عنوان یکی از تجهیزات پرکاربرد در مراکز صنعتی می باشند. در این کنترلرها این امکان وجود دارد که یک PT100 و یا یک ترموکوپل به صورت مستقیم جهت اندازه گیری حرارت متصل شود. در این صورت عددی که بر روی نمایشگر این کنترلر نشان داده می شود دمای اندازه گیری شده می باشد.



بسیاری از این کنترلرها دارای خروجی آنالوگ نیز می باشند که از این خروجی آنالوگ به راحتی می توان در کنترل کننده های دیگر به عنوان ورودی آنالوگ استفاده نمود. همچنین اکثر کنترلرها مجهز به تیغه های فرمان می باشند که این فرمان یک فرمان دیجیتال با توجه به پارامتر S.P می باشد. از این خروجی نیز می توان در ورودی دیجیتال PLC استفاده نمود.



اما زمانی که در یک پروسه هدف اندازه گیری کمیت های بیان شده توسط PLC می باشد، این عمل توسط کارتهای ورودی آنالوگ انجام می شود. در سیگنال های آنالوگ تغییرات جزئی در سیگنال نیز دارای مفهوم بوده که می بایست در سیستم در نظر گرفته شود. زمانی که جهت اندازه گیری یک سیگنال آنالوگ از PLC استفاده می شود قبل از اینکه سیگنال مورد نظر توسط CPU پردازش شود می بایست به فرم دیجیتال تبدیل گردد. این عملیات تبدیل توسط مبدل های ADC کارتهای ورودی آنالوگ انجام می شود. نتیجه مقادیر تبدیل شده را می توان در یک حافظه ذخیره نمود.



در قسمت زیر به برخی از تجهیزات ورودی آنالوگ اشاره شده است:

۱- ترموکوپل ها (TC)

۲- ترمومترها (RTD)

۳- سنسورهای وزن (Load Cell)

۴- سنسورهای فشار

۵- سنسورهای سطح

۶- سنسورهای فاصله

۷- سنسورهای فلومتر

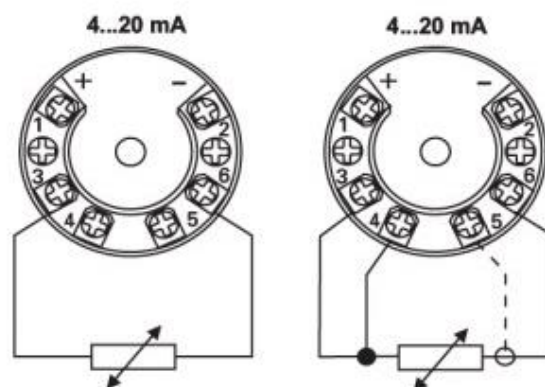
ترانسمیتر چیست ؟

ترانسمیتر وسیله ای است که یک سیگنال الکتریکی ضعیف را دریافت کرده و به سطوح قابل قبول برای کنترلرها و مدارهای الکترونیکی تبدیل می کند ، مثلاً یک حلقه فیدبک سیگنالی در سطح میکروولت یا میلی ولت یا میلی آمپر تولید می کند و این سیگنال ضعیف می تواند با عبور از ترانسمیتر به سیگنالی در سطوح صفر تا ده ولت و یا ۴ تا ۲۰ میلی آمپر تبدیل شود. ترانسمیترها عموماً از قطعاتی مثل op-amp برای تقویت و خطی کردن این سطوح ضعیف سیگنال استفاده می کنند . سنسورها و ملحقات آنها مثل ترانسدیوسرها را در گروه های بزرگی تحت عنوان ابزار دقیق قرار داده و آنها را بر اساس نوع انرژی قابل استفاده و روشهای تبدیل ، دسته بندی می کنند.

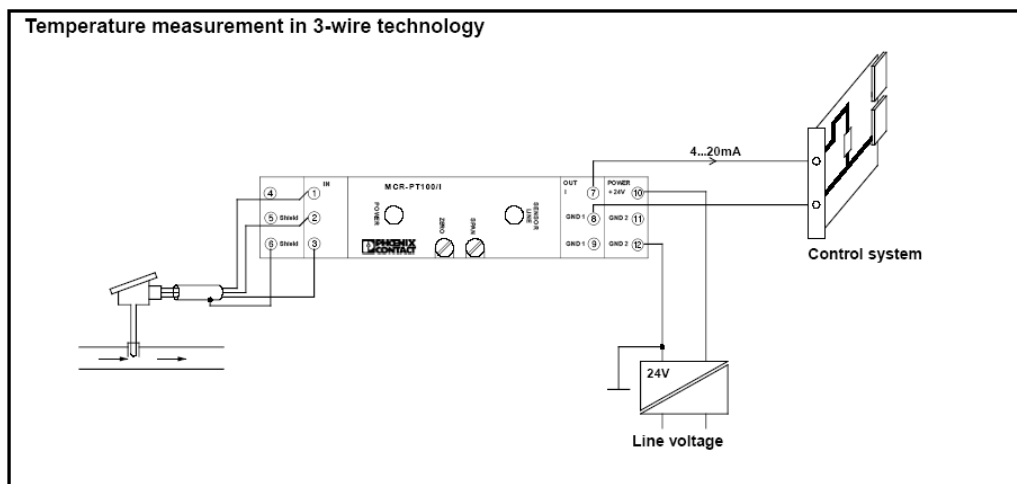
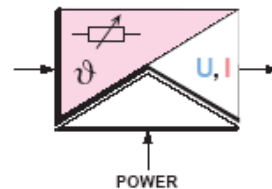


آشنایی با مشخصات چند مدل ترانسمیتر

در بسیاری از کاربردها جهت اتصال سنسور حرارتی PT100 به ورودی آنالوگ یک PLC از ترانسمیتر استفاده می شود. در واقع این ورودی آنالوگ مختص PT100 نبوده و قابلیت دریافت سیگنال های استاندارد را دارا می باشد. در این صورت می بایست از ترانسمیتر با خروجی استاندارد استفاده نمود. برای این منظور شرکت های سازنده بسیاری اقدام به ساخت ترانسمیتر مختص PT100 می نمایند. با استفاده از این تجهیز خروجی PT100 تبدیل به یک سیگنال استاندارد از جنس جریان می شود. البته لازم به ذکر است که ترانسمیتر با یک ولتاژ ثابت DC می بایست تغذیه شود که این ولتاژ معمولاً بین 10 الی 30 ولت DC می باشد.

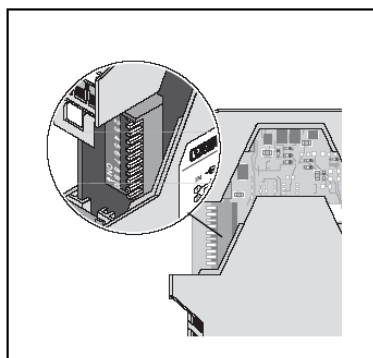


همچنین شرکت فونیکس نیز دارای مبدل های متنوع برای استفاده در سیگنال های آنالوگ می باشد. یک نوع از این مبدل ها، مبدل MCR-PT100 می باشد. توسط این MCR خروجی یک PT100 را می توان به یک سیگنال استاندارد تبدیل نمود.



همانطور که در شکل فوق ملاحظه می کنید این مبدل نیاز به یک تغذیه ثابت دارد. خروجی این مبدل نیز یک سیگنال استاندارد از نوع جریان بوده که جهت ارسال به بردهای کنترلی استفاده می شود. یکی از مزایای این مبدل ها اتصال انواع RTD به ورودی می باشد.

توسط سه سوئیچ اول DIP Switch می توان نوع PT100 را از لحاظ تعداد سیم خروجی جهت اتصال مشخص نمود.

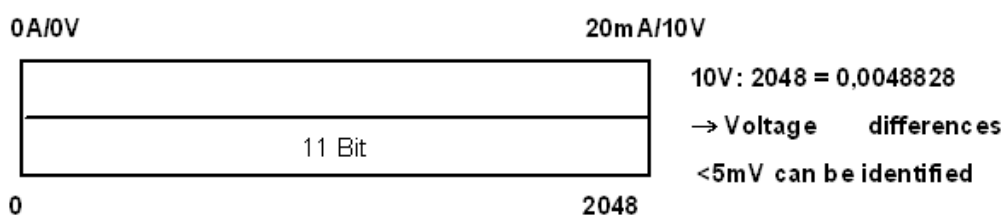


دقت در مبدل آنالوگ به دیجیتال

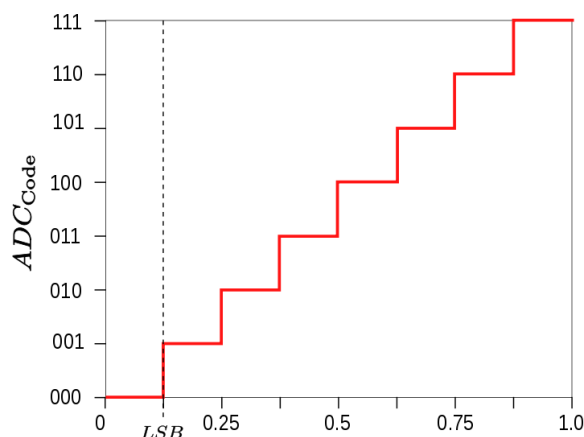
یکی از مشخصات مهم در انتخاب یک کارت ورودی آنالوگ چند بیتی بودن آن می باشد. معمولاً مبدل های ADC دارای قدرت تفکیک ۸ الی ۱۶ بیت می باشند. منظور از قدرت تفکیک یک کارت دقت آشکار سازی کوچکترین تغییر در سیگنال می باشد. مبدل های ۱۶ بیتی دارای بیشترین دقت بوده که این کارت ها معمولاً در PLC های سری 300 و 400 مورد استفاده قرار می گیرند. برای اینکه بتوان حداقل سیگنال قابل تشخیص برای یک کارت را تعیین نمود به شکل زیر دقت کنید. در این شکل مبدل مورد بحث ۱۱ بیتی می باشد.

فرض کنیم به این کارت که دارای قدرت تفکیک ۱۱ بیت می باشد ولتاژ 0 تا 10 ولت اعمال می شود. در این صورت با تقسیم ولتاژ ۱۰ ولت بر ۲ به توان حد تفکیک کارت می توان حداقل تغییر قابل تشخیص توسط کارت را مشخص نمود.

همانطور که در شکل زیر مشاهده می کنید حداقل این سیگنال در حدود 5 میلی ولت می باشد. این عدد بدین معنا می باشد که در این کارت تغییرات کمتر از 5 میلی ولت ثبت نمی شوند.



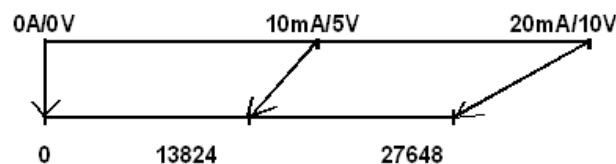
اگر این موضوع را برای یک کارت با حد تفکیک ۱۶ بیت بررسی کنید ملاحظه می شود که عدد به دست آمده بسیار کوچکتر از 5 میلی ولت می باشد. نتیجه این بحث این است که دقت در کارت هایی که دارای قدرت تفکیک بالاتر می باشند بیشتر است.



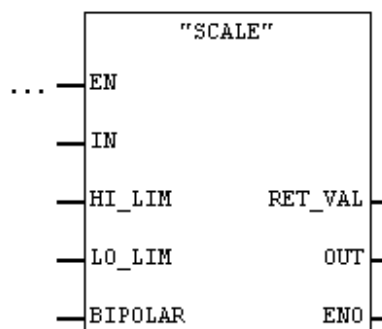
مقیاس گذاری یا SCALE چیست؟

زمانی که یک ورودی آنالوگ در PLC خوانده می شود، حاصل، یک عدد در بازه مشخصی می باشد. این عدد را نمی توان بصورت مستقیم در برنامه نویسی به کار برد. دلیل این مطلب این است که این عدد یک عدد خام بوده و هنوز معادل به واحد نشده است. در نتیجه قبل از شروع به برنامه نویسی می بایست عدد حاصل از دریافت کانال ورودی را معادل به واحد نمود که این عمل توسط برنامه نویس انجام می گیرد.

SCALE یک رابطه و یا یک معادله خط می باشد که توسط شرکت سازنده مشخص شده است. در اکثر مواقع این فرمول بصورت تابعی آماده در اختیار برنامه نویس قرار می گیرد.



تابع آماده در نرم افزار برای استفاده در سیگنالهای ورودی آنالوگ بلوک FC105 می باشد. در واقع توسط این بلوک می توان ورودی آنالوگ را به مقدار کمیت در بازه دلخواه تبدیل نمود.



Range	Voltage e.g.:		Current e.g.:		Resistance e.g.:		Temperature e.g. Pt100	
	Meas.range $\pm 10V$	Units	Meas.range 4 ... 20mA	Units	Meas.range 0...300Ohm	Units	Meas.range -200...+850°C	Units
Overflow	≥ 11.759	32767	≥ 22.815	32767	≥ 352.778	32767	≥ 1000.1	32767
Overrange	11.7589 ... 10.0004	32511 ... 27649	22.810 ... 20.0005	32511 ... 27649	352.767 ... 300.011	32511 ... 27649	1000.0 ... 850.1	10000 ... 8501
Rated range	10.00 7.50 ... -7.5 -10.00	27648 20736 ... -20736 -27648	20.000 16.000 ... 4.000 0	27648 20736 ... 0	300.000 225.000 ... 0.000 0	27648 20736 ... 0	850.0 ... -200.0	8500 ... -2000
Underrange	- 10.0004 ... - 11.759	- 27649 ... - 32512	3.9995 ... 1.1852	- 1 ... - 4864	Negative values not possible	- 1 ... - 4864	- 200.1 ... - 243.0	- 2001 ... - 2430
Underflow	$\leq - 11.76$	- 32768	≤ 1.1845	- 32768		- 32768	$\leq - 243.1$	- 32768

تبدیلات در کارت خروجی آنالوگ

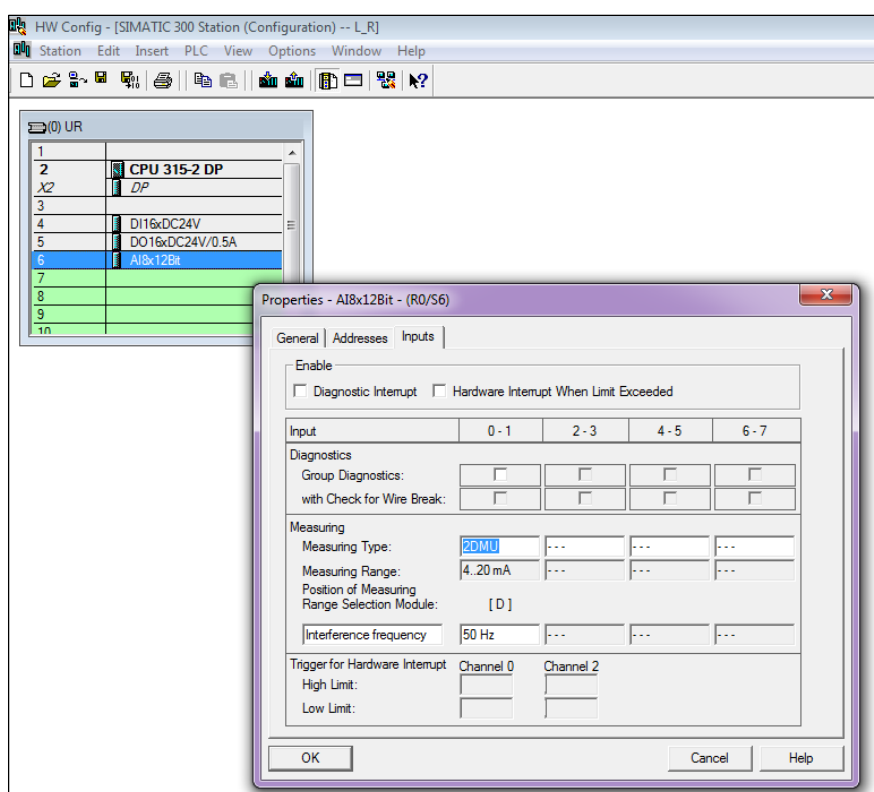
Range	Units	Voltage			Current		
		Output ranges: 0 to 10V 1 to 5V $\pm 10V$			Output ranges: 0 to 20mA 4 to 20mA $\pm 20mA$		
Overflow	≥ 32767	0	0	0	0	0	0
Overrange	32511 ... 27649	11.7589 ... 10.0004	5.8794 ... 5.0002	11.7589 ... 10.0004	23.515 ... 20.0007	22.81 ... 20.005	23.515 ... 20.0007
Rated range	27648 ... 0 ... - 6912 ... - 6913 ... - 27648	10.0000 ... 0 ... 0	5.0000 ... 1.0000 ... 0.9999 ... 0	10.0000 ... 0 ... -10.0000	20.000 ... 0 ... 0	20.000 ... 4.000 ... 3.9995 ... 0	20.000 ... 0 ... -20.000
Underrange	- 27649 ... - 32512			- 10.0004 ... - 11.7589		- 20.007 ... - 23.515	
Underflow	$\leq - 32513$			0		0	

همانطور که در جدول شکل فوق ملاحظه می کنید در مازول های خروجی آنالوگ بر خلاف ورودی آنالوگ، عدد فرستاده شده به کارت تبدیل به یک سیگنال الکتریکی می شود. به عنوان مثال اگر عدد ۲۷۶۴۸ به کارت فرستاده شود، ولتاژ 10V و همچنین اگر عدد 0 فرستاده شود، کارت ولتاژ 0V را تولید می کند. البته لازم به ذکر است در صورتی که عدد منفی به کارت رسال شود، کارت نیز سیگنال منفی تولید می کند.

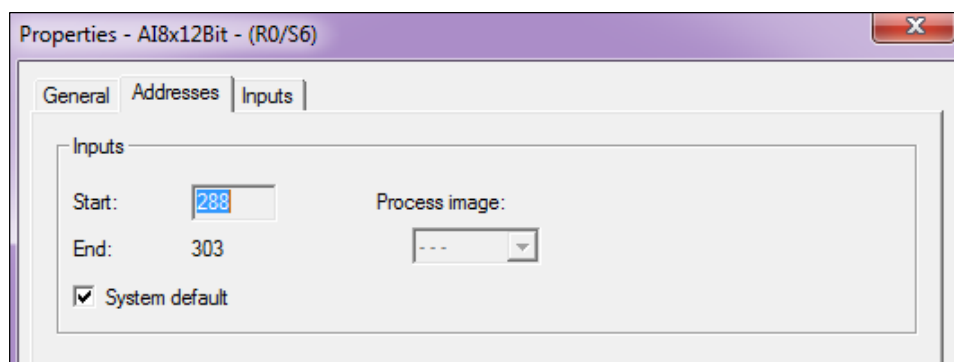
مثال

فرض کنید یک ترانسمیتر فشار با خروجی 4 تا 20 میلی آمپر به کارت AI کنترلر S7-300 متصل شده است. ترانسمیتر بر روی بازه 0 تا 30 بار کالیبره شده است. می خواهیم مقدار فشار را بر روی HMI متصل شده به PLC نمایش دهیم.

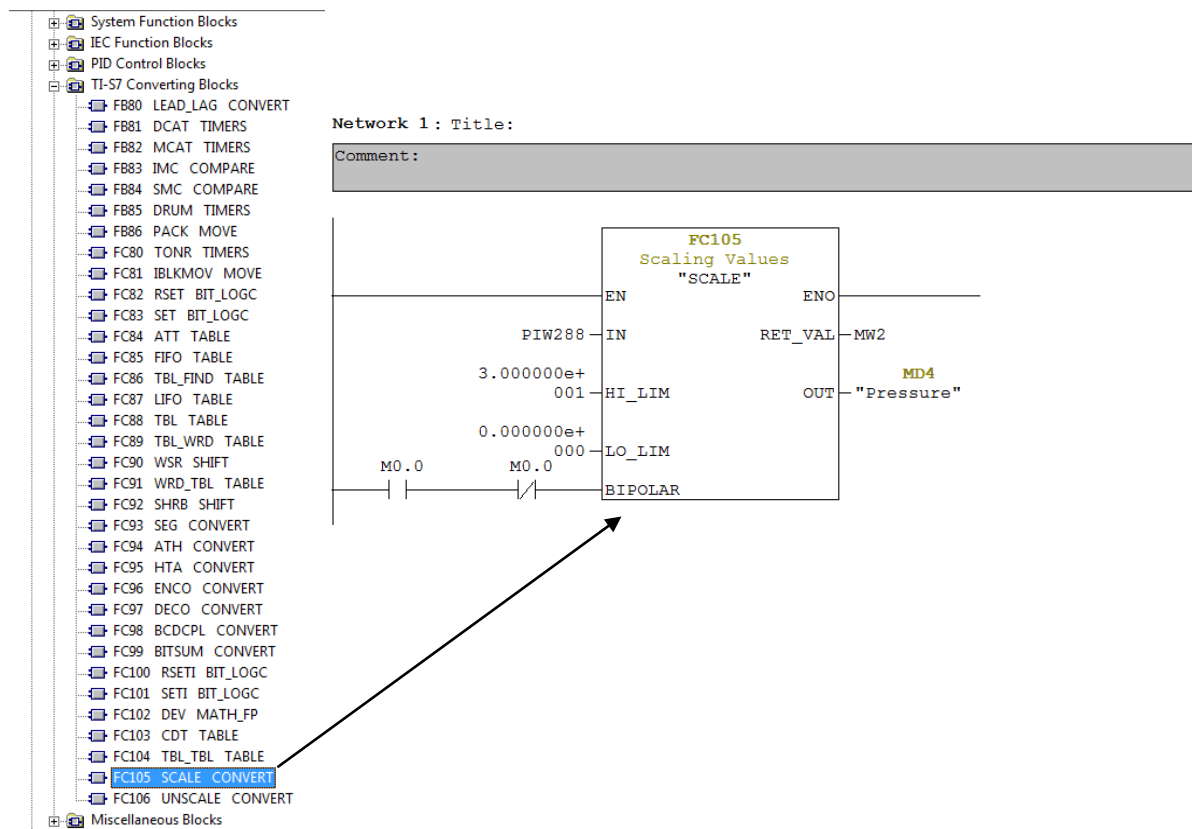
ابتدا در محیط HW یک کارت AI وارد و تنظیمات کانال را بر روی 4 تا 20 میلی آمپر ۲ سیم قرار می دهیم.



آدرس اختصاص داده شده به کانال اول کارت AI، آدرس 288 می باشد. پس ترانسمیتر به کانال اول به آدرس PIW288 متصل شده است.



در ادامه بلوک FC105 را از کتابخانه نرم افزار فراخوانی می کنیم.



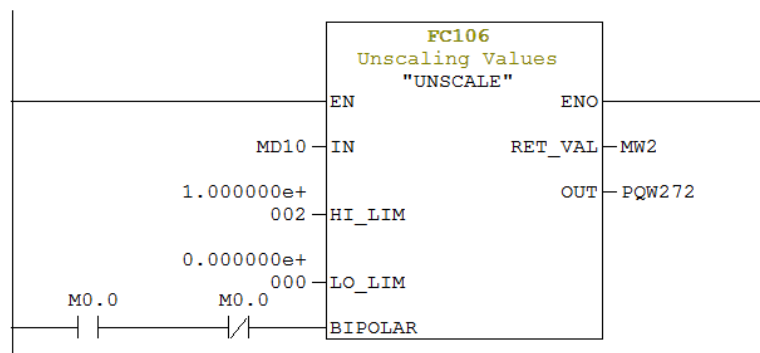
مقدار فشار در حافظه MD4 به صورت Real قابل دریافت می باشد.

Unscale توسط تابع FC106

زمانی که از کارتهای خروجی آنالوگ جهت کنترل شیرهای تدریجی و یا کنترل سرعت موتور توسط درایوها استفاده می شود، حتما نیاز می باشد که جهت برگرداندن ورودی Real به عدد صحیح جهت ارسال به کارت AO از تابع Unscale استفاده شود. عملکرد در کارتهای AO بدین صورت می باشد که کارت با دریافت عدد صحیح ۱۶ بیتی (0 تا 27648) آن را به توجه به حد بالا و حد پایین مشخص شده، تبدیل به سیگنال آنالوگ می کند. عدد ارسالی به کارت همانند آنچه که در مورد AI بیان شد می تواند بازه منفی را نیز شامل شود.

Network 2: Title:

Comment:



در برنامه فوق مقدار مقدار درصد باز شدن یک کنترل ولو در حافظه MD10 قرار می گیرد. تابع نیز به ازای بازه 0 تا 100 دریافتی، مقدار 0 تا 27648 را به PQW272 منتقل می کند. کارت AO نیز به ازای این بازه خام، سیگنال 4 تا 20 میلی آمپر را به کنترل ولو ارسال می کند.

تمرین

۱- در یک مسیر عبور روغن از یک PT100 برای اندازه گیری دمای روغن استفاده شده است. خروجی PT100 به یک ترانسمیتر با خروجی 4 تا 20 میلی آمپر متصل شده است. ترانسمیتر نیز بر روی 0 تا 350 درجه کالیبره شده است. برنامه کنترلی می بایست به گونه ای باشد که زمانی دما بیش از 220 درجه شد، آلارم High Temperature بر روی HMI نمایان شود (بیت M0.0 ست شود). اگر دما کمتر از 220 درجه بود، امکان تایید آلارم توسط یک باتن (M0.1) وجود داشته باشد. در غیر اینصورت آلارم ریست نشود. در ضمن مقدار دما نیز می بایست بر روی HMI نمایش داده شود (DB1.DB0).



۲- در یک مخزن جهت اندازه گیری سطح مخزن از یک سنسور Level Detector که می تواند یک سنسور راداری باشد استفاده شده است. ارتفاع مخزن 3 متر می باشد. خروجی ترانسمیتر رادار نیز به ازای 3 متر سیگنال 20 میلی آمپر و به ازای 0 متر سیگنال 4 میلی آمپر به کارت ارسال می کند. می خواهیم برنامه به گونه ای باشد که:



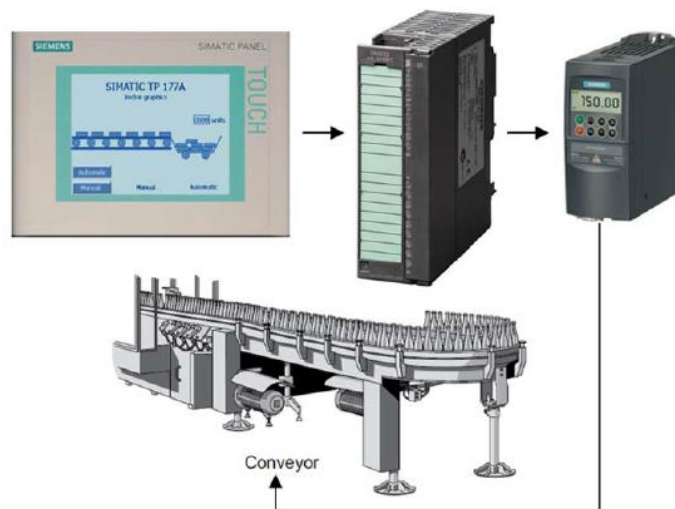
* زمانی که سطح مخزن کمتر از 1 متر بود، ولو 1 باز باشد.

* زمانی که سطح مخزن بین 1 تا 2 متر بود، ولو 2 باز باشد.

* زمانی که سطح مخزن بیشتر از 2.5 متر بود، ولو 3 باز باشد.

سطح مخزن بر حسب متر نیز می بایست بر روی HMI نمایش داده شود (DB1.DB0)

۳- در یک پروسه صنعتی از یک HMI جهت کنترل سرعت یک موتور سه فاز توسط یک اینورتر که مربوط به یک کانوایر می باشد استفاده می گردد. در این حالت از یک ابزار در HMI جهت کنترل سرعت موتور بین فرکانس کاری 0 تا 50 هرتز استفاده شده است. ورودی آنالوگ اینورتر نیز به یکی از کانالهای خروجی آنالوگ PLC متصل شده است. در این صورت کارت خروجی آنالوگ با تولید ولتاژ 0 تا 10V برای اینورتر موجب کنترل سرعت موتور در فرکانس مشخص شده می گردد. مقدار وارد شده بر روی HMI به عنوان فرکانس کاری درایو، در حافظه DB1.DBD0 قرار می گیرد. در ضمن زمانی که فرکانس کاری بیش از 45HZ شد، آلارم High Speed بر روی HMI ظاهر شود(بیت M0.0)



۴- در یک ماشین پرس جهت کنترل سرعت بالا و پایین آمدن پرس از یک شیر پروپورشنال استفاده می شود. این شیر نیز توسط یک برد الکترونیکی کنترل می شود. تنظیم نقطه SetPoint برای برد الکترونیکی توسط خروجی آنالوگ PLC صورت می گیرد. سیگنال SetPoint یک ولتاژ در بازه $-10V \dots +10V$ می باشد که می بایست توسط ماژول AO تولید شود. این شیر در $-10V$ ولت کاملاً بسته و در $+10V$ ولت کاملاً باز می باشد. کنترل مقدار باز یا بسته شدن شیر توسط یک HMI صورت می گیرد.

عدد ارسالی از HMI یک عدد در بازه 0% تا 100% بوده که نمایانگر میزان باز و بسته بودن شیر می باشد.



۵- در یک بخش از یک ماشین تراش از یک سنسور فاصله آنالوگ با مشخصات زیر استفاده شده است.

2...6mm (رنج مجاز اندازه گیری)

0...10v (خروجی)

در واقع این سنسور یک سنسور فاصله می باشد که حساسیت آن به فلزات بوده و بازه اندازه گیری آن 2 تا 6 میلی متر می باشد. خروجی آن نیز به ازای این بازه، یک سیگنال آنالوگ 0 تا 10 ولت می باشد. همچنین بر روی این ماشین تراش یک موتور نیز وجود دارد که توسط یک درایو کنترل می شود. در این موتور، دور متناسب با قرارگرفتن قطعه در فواصل مختلف در مقابل سنسور می باشد. حداکثر فرکانس موتور نیز 10HZ می باشد. برنامه کنترلی می بایست به گونه ای باشد که هر چه قدر قطعه به سنسور نزدیک تر می گردد دور موتور بیشتر شود. در واقع فرمان درایو به صورت معکوس می باشد. به عبارت دیگر هر چه قدر ولتاژ در ورودی آنالوگ به ازای نزدیک شدن قطعه کمتر می شود، دور موتور بیشتر شود.

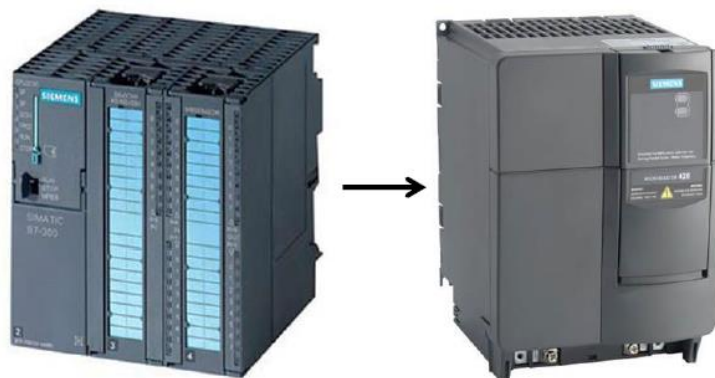
۶-فرکانس یک درایو توسط خروجی آنالوگ PLC کنترل می شود. بر روی HMI دو کلید جهت افزایش (Increment=M0.0) و یا کاهش (Decrement=M0.1) سرعت وجود دارد. نحوه کار بدین صورت می باشد که با هر بار فشردن باتن INC روی HMI، 5 هرتز به فرکانس درایو اضافه و با هر بار فشردن باتن DEC نیز 5 هرتز از فرکانس کاری کم می گردد.

در ضمن از خروجی های دیجیتال PLC نیز برای استارت/ استپ درایو (Q0.0) و همچنین تغییر جهت چرخش (Q0.1) استفاده می کنیم. فرامین استارت و استپ (M0.2) و تغییر جهت (M0.3) نیز از HMI صادر می گردد.

بر روی HMI یک باتن RESET (M0.4) نیز تعبیه شده است که با فشردن آن هم موتور استپ می شود و هم SP فرکانس صفر می گردد.

❖ خروجی آنالوگ 0 تا 10 ولت PLC به ورودی آنالوگ درایو متصل شده است.

❖ دقت کنید که بازه مقدار ورودی، زمانی که از ابتدا بر روی باتن DEC کلیک می کنیم منفی نشود.

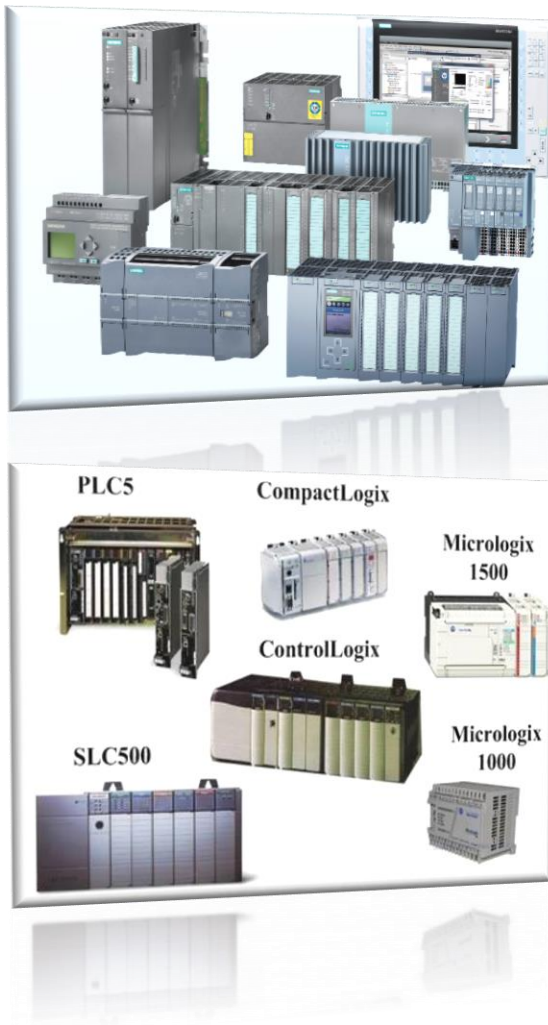


۷- یک PT100 به یک ترانسمیتر با خروجی جریانی متصل شده است. ترانسمیتر بر روی دمای 0 تا 350 درجه کالبره شده است. در واقع خروجی ترانسمیتر به ازای 0 تا 350 درجه، 4 تا 20 میلی آمپر می باشد که به ورودی آنالوگ PLC متصل شده است. خروجی آنالوگ PLC نیز به یک شیر تناسبی (آنالوگ) متصل شده است که به ازای 4 تا 20 میلی آمپر، 0% تا 100% تغییر وضعیت می دهد. برنامه کنترلی می بایست به گونه ای باشد که بین دمای 130 تا 150، شیر بین 0 تا 100 درصد کنترل شود. زیر دمای 130 شیر کاملاً باز (20 میلی آمپر)، بالای 150، شیر کاملاً بسته (4 میلی آمپر) و بین این دو رنج بین 0 تا 100 درصد تغییر کند. در واقع هر چه قدر دما بالاتر می رود، شیر بسته شود. هدف از اجرای این پروژه نگه داشتن دمای سیستم بین دو رنج داده شده توسط کاربر می باشد که در این مثال 130 و 150 در نظر گرفته شده است.

در ضمن بر روی HMI می بایست دما و مقدار باز شدن شیر برحسب درصد نمایش داده شود.



دوره های تخصصی اتوماسیون صنعتی SIEMENS و Allen Bradley



Studio 5000-ControlLogix

Studio 5000_PanelView 5000

RSLogix500_SLC500

Factory Talk View Studio

PLC- S5

S7-400F/FH

PCS7

S7-300,400

S7-400F/FH

S7-1200(TIA)

S7-300,400(TIA)

WinCC

Protool

WinCC Flexible

Ethernet - Profibus -Profinet

SINAMICS Servo Drive

Micromaster Drive

Simotion Scout

S7 PID