

## پودمان ۳

# نصب و راه‌اندازی کنترل‌کننده‌های منطقی



بشر از زمان‌های دور به دنبال روش‌های کنترل دستگاه‌های صنعتی و تکامل بخشیدن به آنها بوده و آنها را برای کنترل دستگاه‌ها به کار گرفته است ولی در چند دههٔ اخیر با پیشرفت فناوری و روی کار آمدن ریزپردازنده‌ها تحوّل چشمگیری در فرایندهای کنترلی به‌وجود آمده است.

یکی از این تحولات، به‌کارگیری علم اتوماسیون است که با استفاده از «PLC» به اجرای پروسه‌های صنعتی و ساختمانی پرداخته است. در اکثر پروسه‌های صنعتی حلقه‌های کنترلی به «PLC» ختم می‌شود که به‌عنوان مغز متفکر سیستم، کنترل پروسه را در اختیار دارد. این سیستم علاوه بر داشتن توانایی بالا در کنترل فرایندها، برای گرایش‌های مختلف علمی، از قابلیت برنامه‌نویسی بسیار ساده‌ای برخوردار است و به‌راحتی به دستگاه‌ها متصل می‌شود.

این توانایی‌ها باعث شده است که کاربرد «PLC» در صنعت توسعهٔ چشمگیری پیدا کند و نیاز به یادگیری آن نیز کاملاً احساس شود.



### واحد یادگیری ۳

## شایستگی انتخاب سخت افزار مناسب برای کنترل فرایند

### هدف‌های این شایستگی عبارت‌اند از:

- توانایی شناخت سخت‌افزار و بررسی انواع مدل‌ها، بررسی رله حالت جامد (SSR) و استفاده جهت راه‌اندازی موتورهای الکتریکی؛
- توانایی انتخاب سخت‌افزار مناسب، با توجه به مزایا و معایب محصولات شرکت‌های مختلف سازنده کنترل‌کننده‌های منطقی؛
- توانایی انتخاب سخت‌افزار، با توجه به تعداد اِلمان‌های کنترل‌کننده و کنترل‌شونده همراه با تعیین مدل و تهیه فهرست خرید قطعات؛
- توانایی سیم‌کشی ورودی و خروجی‌های «PLC» برای اِلمان‌های مختلف؛
- توانایی نصب نرم‌افزار «wpl soft» و «Isp soft» روی رایانه و بررسی تفاوت آنها؛
- توانایی استفاده از دستورات و توابع مختلف در برنامه‌نویسی؛
- توانایی انتقال برنامه نوشته شده در نرم‌افزار به سخت‌افزار؛
- توانایی هماهنگ کردن نرم‌افزار با سخت‌افزار و راه‌اندازی سیستم با «PLC»؛
- توانایی راه‌اندازی یک ربات صنعتی.

### استاندارد عملکرد

پس از اتمام واحد یادگیری و کسب شایستگی، هنرجویان قادر خواهند بود با انتخاب و به‌کارگیری جهت اتوماسیون، برنامه‌ریزی و پیاده‌سازی کنترل فرایند توسط «PLC» اقدام کنند.



### PLC چیست؟

«PLC» حروف اول عبارت «Programmable Logic Controller» است و به معنای کنترل کننده منطقی قابل برنامه ریزی است. منطق کنترل در این کنترل کننده، توسط نرم افزار تعیین می شود، به گونه ای که اطلاعاتی را در قسمت ورودی دریافت می کند و آنها را طبق برنامه ای که در حافظه اش ذخیره شده است پردازش می نماید و نتایج به دست آمده را نیز از طریق واحد خروجی به صورت فرمان هایی به گیرنده ها و اجرا کننده های فرمان مانند عملگرها (Actuators) و رله ها ارسال می کند. به عبارت دیگر «PLC» عبارت است از یک کنترل کننده منطقی به طوری که می توان برایش منطق کنترل را توسط برنامه کاربر (user) تعریف نمود و فرایند کنترل آن، در صورت نیاز، به راحتی قابل تغییر است.

بدانید



فعالیت

مزایای «PLC» را نسبت به مدارهای رله ای، بررسی کنید.



فیلم

معرفی و نمایش سخت افزار «PLC»



## سخت‌افزار «PLC»

این سخت‌افزار شامل قسمت‌های مختلفی است و هر کدام از اجزای آن نیز انواع مختلفی دارند. به همین دلیل لازم است ابتدا شرایط مورد نیاز سیستم تحت کنترل بررسی شود. سپس متناسب با شرایط مورد نیاز، سیستم سخت‌افزار مناسب برای آن انتخاب می‌شود.

اجزا عبارت‌اند از:

- ۱- منبع تغذیه (PS)؛
- ۲- واحد پردازشگر مرکزی (CPU)؛
- ۳- واحد حافظه (memory)؛
- ۴- ترمینال‌های ورودی (input module)؛
- ۵- ترمینال‌های خروجی (output module)؛
- ۶- کارت رابط (interface module)؛
- ۷- کارت شبکه (communication module).

### Teamwork

Programmable logic controllers («PLC») have been an integral part of factor automation and industrial process control for decades. «PLC's» control a wide array of applications from simple lighting functions to environmental systems to chemical processing plants. These systems perform many functions, providing a variety of analog and digital input and output interface; signal processing; data conversion; and various communication protocols.

All of the «PLC's» components and functions are centered around the controller, which is programmed for a specific task.

ترجمه



## منبع تغذیه (Power Supply)

جهت تغذیه قسمت‌های مختلف «PLC» از جمله «CPU» و کارت‌های ورودی و خروجی و به‌طور کلی هر قسمت که به مجموعه «PLC» اضافه می‌شود لازم است حتماً تغذیه آن تأمین شود.

منبع تغذیه، که در انواع مختلف طراحی می‌شود، عبارت‌اند از:

- ۱- منبع تغذیه معمولی با آداپتور، که با ترانس معمولی و پل دیود و خازن صافی ساخته می‌شود.
- ۲- منبع تغذیه سوئیچینگ، که با قطعات الکترونیکی ساخته می‌شود و براساس سوئیچ زنی قطعات الکترونیکی کار می‌کند.

مزایای منبع تغذیه سوئیچینگ را نسبت به منبع تغذیه معمولی بررسی کنید و مزایا و معایب هر کدام را بنویسید.

فعالیت





نکات مهم در انتخاب منبع تغذیه برای یک سیستم کنترل عبارت اند از:

- ۱ توان مصرف کننده ها (میزان جریان مورد نیاز قسمت های مختلف که باید منبع تأمین کند)؛
- ۲ با شبکه شهری هم خوانی داشته باشد (ایران تکفاز حداکثر 230 ولت)؛
- ۳ منبع باید از نوع سوئیچینگ باشد تا راندمان کاری «PLC» افزایش یابد.

چند نمونه از منابع تغذیه «دلتا» شامل «PS05، PS02 و PS01» که به ترتیب از چپ به راست قابل مشاهده هستند منابع با جریان و توان بالاتر نیز وجود دارد که براساس نیاز انتخاب می شود. به عنوان مثال PS02 یک منبع تغذیه با جریان خروجی 2 آمپر است.



- ۱ منابع تغذیه با استاندارد و عرضه شده در بازار دارای چه رنج هایی است؟ (با ذکر کد خرید برای برند دلتا از روی کاتالوگ)
- ۲ در صورتی که یک سیستم کنترل با جریان بالاتر نیاز باشد و منبع مورد نیاز در بازار عرضه نشده است، چه اقدامی باید انجام داد؟

## واحد پردازشگر مرکزی «Central processing unit»

این واحد در واقع مغز سیستم کنترل است و به این صورت عمل می کند که اطلاعات کارت ورودی از طریق «BUS» به آن منتقل می شود و پس از پردازش، طبق برنامه کاربر، نتایج به دست آمده از طریق همان «BUS» به کارت خروجی انتقال می یابد.

یکی از مهم ترین قدم های طراحی انتخاب «CPU» مناسب برای کنترل فرایند است. انتخاب نادرست «CPU» یا انتخابی که مبتنی بر آینده نگری نباشد مشکلات زیادی را در آینده برای سیستم ایجاد خواهد کرد و هزینه های سنگینی را از این بابت تحمیل خواهد نمود.

فاکتورهای اولیه ای که معمولاً در انتخاب «CPU» به آنها توجه می شود عبارت اند از:

- ۱- تعداد ورودی و خروجی (I/O)
- ۲- حجم برنامه نویسی پروسه
- ۳- سرعت پردازش
- ۴- تعداد بیت حافظه مورد نیاز
- ۵- تعداد تایمر مورد نیاز
- ۶- تعداد کانتر مورد نیاز
- ۷- تعداد رجیستر مورد نیاز

و در فرایندهای پیچیده تر ممکن است به فاکتورهای زیر نیز نیاز پیدا کنیم:

- ۱- کارت‌های ورودی و خروجی‌های خاص مثل کارت کنترل موقعیت؛
- ۲- کارت شبکه مورد نیاز سیستم.

اولین فاکتور مهم در انتخاب «CPU» مناسب، تعداد ورودی و خروجی‌های مورد نیاز پروژه است. این تعداد لازم است با احتساب توسعه در آینده، یعنی حداقل ۲۰ درصد اضافه بر تعداد به دست آمده باشد. به طور کلی هر مدل «CPU» می‌تواند تعدادی I/O را پشتیبانی نماید. بنابراین لازم است ابتدا مدل‌های مختلف و قابلیت‌های هر کدام را بشناسیم.

«CPU»‌های دلتا دارای سری‌های S-E-PM-MC است و هر سری نیز دارای زیر مجموعه‌هایی است.

بدانید



- کلیه «CPU»‌های دلتا با تغذیه ۲۴VDC کار می‌کنند (به جز سری E که دارای منبع داخلی هستند). نکته دیگر اینکه می‌توان تغذیه «CPU» را از منبع خودش تأمین نمود.
- نظر به اینکه منبع داخلی توان کمی دارد، لازم است برای توان‌های بالا منبع تغذیه اضافه شود.
- «CPU»‌های مدل S که پسوند E دارند (مثل SE۱۲)، جهت شبکه شدن و برنامه‌ریزی (Program) به پورت اترنت مجهزند.
- «CPU»‌های مدل S و E که پسوند X دارند (مثل SX و EX)، دارای ورودی و خروجی دیجیتال و آنالوگ هستند.
- «CPU»‌های مدل S که پسوند S دارند (مثل SS)، فقط دارای ورودی و خروجی دیجیتال هستند.

فعالیت



جدول زیر را تکمیل کنید و کاتالوگ دو «CPU» دیگر را از اینترنت بگیرید و مشخصات آنها را بررسی کنید.

۱ تعداد کل ورودی و خروجی

۲ مدل «CPU»

۳ نوع تغذیه

۴ نوع خروجی

۵ ورژن «CPU»

۶ تعداد تایمر

۷ تعداد کانتر

Series	1	2	3	4	5
1. Total I/O					
2. Model					
ES / ES2 : ES / ES2 series PLC					
EX / EX2 : EX / EX2 series PLC					
SS / SS2 : SS / SS2 series PLC					
SA / SA2 : SA / SA2 series PLC					
SX / SX2 : SX / SX2 series PLC					
SC : SC series PLC					
SV : SV series P.C					
SE : SE series P.C					
PM : PM series PLC					
MC : MC series PLC					
EH : EH series PLC					
EC : EC series PLC					
3. Power supply					
00 : AC power input					
11 : DC power input					
4. Output type					
R : Relay					
T : Transistor (NPN)					
M : Mixed with differential signal					
S : Transistor (PNP)					
RC : Relay + CANopen					
TC : Transistor + CANopen					
5. Version					

بدانید

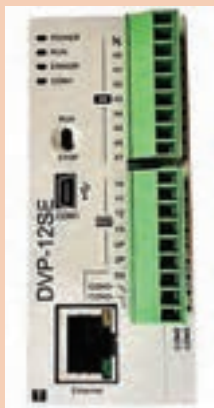
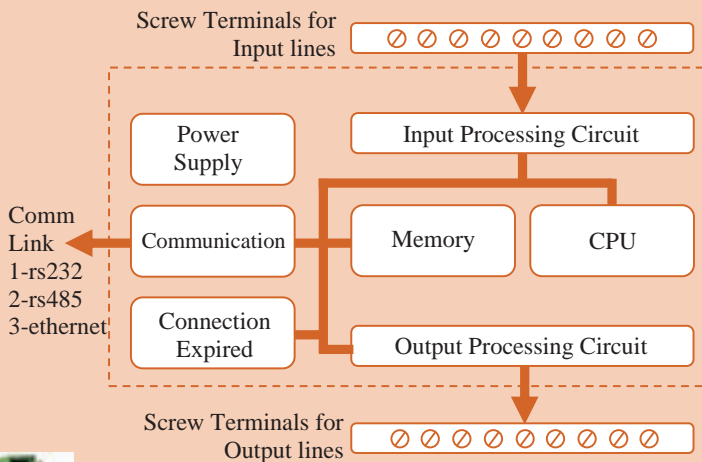


- چراغ سیگنال «ERROR» قرمز رنگ است و وقتی روشن باشد، نشان‌دهنده آن است که در کارت و «CPU» خطا وجود دارد (مثلاً اتصالات درست نیست یا تغذیه کارت وصل نیست).
- چراغ سیگنال «RUN» سبز رنگ است و وقتی روشن باشد علامت آن است که «CPU» در وضعیت اجرای برنامه‌هاست.
- چراغ سیگنال «STOP» نارنجی رنگ است و وقتی روشن باشد نشان‌دهنده آن است که «CPU» روشن است ولی کار پردازش انجام نمی‌شود.
- چراغ سیگنال «RS۲۳۲» نارنجی رنگ است و وقتی در وضعیت چشمک‌زن باشد یعنی «CPU» با یک «DEVICE» دیگر از طریق این پورت در حال تبادل اطلاعات است.
- چراغ سیگنال «RS۴۸۵» نارنجی رنگ است و وقتی چشمک‌زن باشد یعنی «CPU» با یک «DEVICE» دیگر از طریق این پورت در حال تبادل اطلاعات است.
- کلید در وضعیت «RUN»: در این حالت اگر نرم‌افزاری نیز «RUN» باشد، «CPU» در حال پردازش است.
- کلید در وضعیت «STOP»: در این حالت پردازش «CPU» متوقف است.

فعالیت



پروتکل‌های ارتباطی در «دلتا» را برای هر «CPU» بررسی کنید.





بررسی کنید کاربرد کانکتور مشخص شده در «CPU» چیست؟



### Teamwork

A programmable logic controller is a specialized computer used to control machines and processes. It therefore shares common terms with typical PCs like central processing unit, memory, software and communications. Unlike a personal computer though the «PLC» is designed to survive in a rugged industrial atmosphere and to be very flexible in how it interfaces with inputs and outputs to the real world.



## ترمینال‌های ورودی «Input Module»

اطلاعات از محل سیستم تحت کنترل، دریافت می‌شود. این اطلاعات از طریق ورودی‌های مختلفی قابل دریافت است که عبارت‌اند از:

- ۱- ورودی دیجیتال (DI)
- ۲- ورودی آنالوگ (AI)
- ۳- ورودی خاص (انکودر)



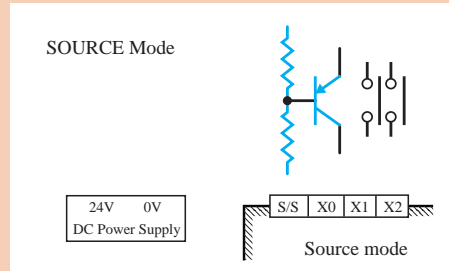
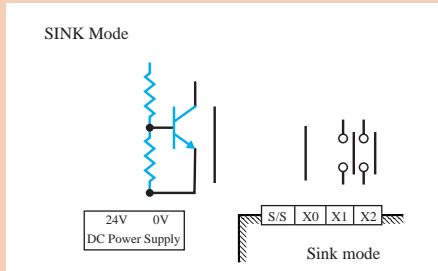
معرفی کارت‌های ورودی و خروجی

۱- **ورودی دیجیتال (Digital Input):** هر ورودی دیجیتال دارای دو حالت صفر یا یک است و معمولاً با ۲۴ ولت تغذیه می‌شود. از جمله ورودی‌های دیجیتال در محیط‌های صنعتی می‌توان استپ، استارت، میکروسوییچ، حسگرهای دیجیتال، و همچنین کنتاکت وسایل حفاظتی مانند بی متال و... را نام برد. در برند «دلتا» (و معمولاً برندهای تایوانی) جهت سیم‌کشی ورودی‌ها ترمینالی به نام S/S وجود دارد. این ترمینال در تمام کارت‌هایی که ورودی هستند اولین ترمینال است و مشخص‌کننده مقدار ولتاژ برای فعال شدن ورودی‌هاست.



بودمان سوم: نصب و راه اندازی کنترل کننده های منطقی

■ اگر می خواهید ورودی ها با ولتاژ منفی فعال شوند باید ولتاژ مثبت را به ترمینال S/S متصل کنید (SINK). بالعکس اگر می خواهید ورودی ها با ولتاژ مثبت فعال شوند باید ولتاژ منفی را به ترمینال S/S متصل کنید (SOURCE).



بنابراین اگر مثلاً بخواهیم با یک حسگر pnp، که سیم برگشت آن در هنگام فعال شدن ۲۴+ می شود، یک ورودی را فعال کنیم، باید به S/S ولتاژ منفی بدهیم. ■ در مسیر هر ورودی دیجیتال داخل کارت، جهت حفاظت «CPU» و مدارهای داخلی در مقابل ولتاژ ناگهانی، از فتو ترانزیستور<sup>۱</sup> استفاده شده است.

بدانید



فعالیت



انواع کارت های ورودی را از نظر مدل و تعداد ورودی بررسی کنید.



فعالیت



با توجه به تأثیر ترمینال S/S، دو نمونه سیم کشی ورودی را از روی کاتالوگ رسم کنید.

۱- Optocoupler

## ترمینال‌های خروجی «Output Module»

از این ترمینال‌ها برای ارسال نتایج به دست آمده پس از پردازش جهت فعال کردن محرک‌ها و رله‌ها استفاده می‌شود و دارای انواع زیر است:

۱- خروجی دیجیتال (DO) ۲- خروجی آنالوگ (AO) ۳- خروجی خاص (فرکانس بالا) هر خروجی دیجیتال دارای دو حالت صفر یا یک است. به طور کلی خروجی دیجیتال در «PLC» به دو صورت در بازار عرضه می‌شود:

- ۱- خروجی ترانزیستوری که در نام‌گذاری، با پسوند T مشخص می‌شود.
- ۲- خروجی رله‌ای که در نام‌گذاری، با پسوند R مشخص می‌شود.

بدانید



- در خروجی از نوع ترانزیستوری اگر لازم باشد به بار متصل شود باید خروجی به رله کمکی متصل شود و از طریق کنتاکت رله به بار فرمان داده شود.
- در نوع رله‌ای، رله در داخل کارت قرار دارد و به راحتی می‌توان از طریق کنتاکت آن به محرک‌ها فرمان داد.
- خروجی ترانزیستوری نسبت به نوع رله‌ای دارای مزایای زیر است:
  - سرعت سوئیچ بالاتر؛
  - عمر مفید بالا، به دلیل نداشتن کنتاکت مکانیکی؛
  - تعداد خروجی بیشتر، به دلیل حجم کم ترانزیستور؛
  - تعمیر راحت‌تر، به دلیل بیرون قرار گرفتن از کارت.
- خروجی رله‌ای نسبت به نوع ترانزیستوری دارای مزایای زیر است:
  - سیم‌کشی آن ساده‌تر است.
  - در ولتاژهای مختلف به راحتی قابل استفاده است و محدود به ۲۴ ولت نیست.
  - جهت تغذیه مصرف‌کننده، جریان بالاتری در رنج ۵ تا ۱۰ آمپر از کنتاکت می‌توان عبور داد ولی در ترانزیستوری محدودیت جریان داریم. در برند دلتا هر خروجی ۳۰۰ میلی‌آمپر است.

فعالیت



انواع کارت‌های خروجی را از نظر مدل و تعداد ورودی بررسی کنید.

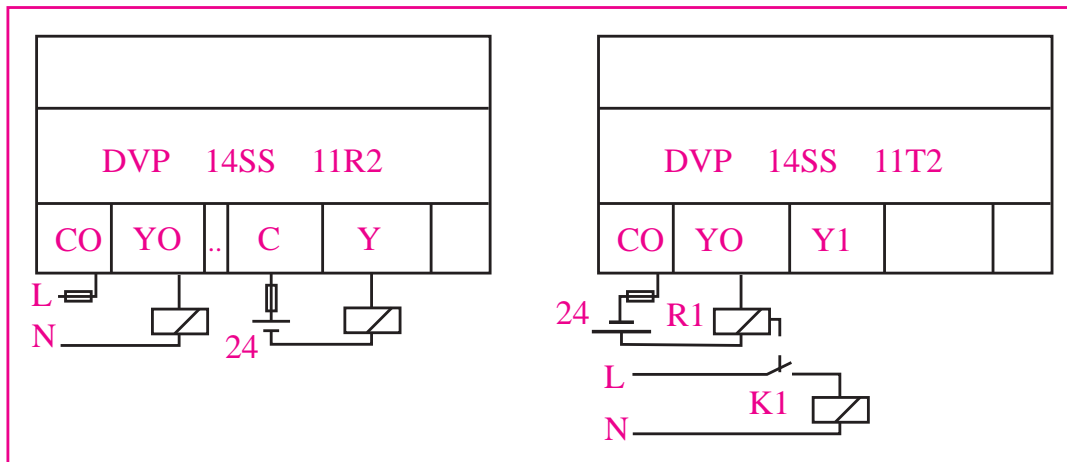


کارت 16SP 11R یک کارت ترکیبی است که ۸ عدد ورودی و ۸ عدد خروجی به مجموعه اضافه می کند. بنابراین در زمانی که لازم باشد تعدادی ورودی و خروجی اضافه شود، به جای دو کارت می توان از یک کارت ترکیبی استفاده نمود.



پسوند M برای کارت ورودی  
پسوند N برای کارت خروجی  
پسوند P برای کارت ترکیبی

شکل زیر نحوه سیم کشی کارت های خروجی را در برند دلتا نشان می دهد.



در سیستم کنترل با «PLC» اطلاعات دستگاه تحت کنترل از طریق ترمینال ها و کارت های ورودی دریافت و از طریق «BUS» به «CPU» منتقل می شود. در «CPU» طبق برنامه کاربر، که در نرم افزار «WPL SOFT» یا «ISP SOFT» نوشته شده از طریق کابل به حافظه «CPU» منتقل می شود و پردازش صورت می گیرد. نتایج به دست آمده پس از پردازش، توسط ترمینال های خروجی، به محرک ها و رله ها ارسال می شود و هیچ گونه ارتباط الکتریکی بین ورودی و خروجی وجود ندارد.

## نصب نرم افزار «WPL SOFT»



نرم افزار «WPL» به راحتی نصب می شود. فقط کافی است که آخرین نسخه نرم افزار را تهیه و گزینه «Setup» را اجرا کنید. پس از نصب، یک آیکون در صفحه ایجاد می شود، و می توانید نرم افزار را باز کنید و براساس نیاز کنترل پروسه، برنامه نویسی کنید.

نصب نرم افزار و معرفی آن

فیلم



برای برنامه نویسی «CPU» های دلتا از یک نرم افزار دیگر به نام «ISP SOFT» نیز می توانیم استفاده کنیم و این نرم افزار برای ارتباط با «CPU» نیاز به یک نرم افزار جانبی با نام «COMMGR» دارد که لازم است متناسب با نیاز، «DRIVER» را تعیین کنیم تا بتوانیم با شبیه ساز یا «CPU» واقعی ارتباط برقرار کنیم.

بدانید



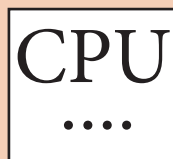
- نرم افزار «WPL SOFT» را بر روی رایانه نصب و منوهای آن را بررسی کنید.
- نرم افزار «ISP SOFT» را نصب و تفاوت آن را با «WPL SOFT» بررسی کنید.

فعالیت



یک سیستم کنترل پیشنهاد کنید که بتواند ۷۸ ورودی و ۴۵ خروجی دیجیتال را پوشش دهد. (مشخص کردن قطعات با کد سفارش الزامی است.)

فعالیت



۱

۲

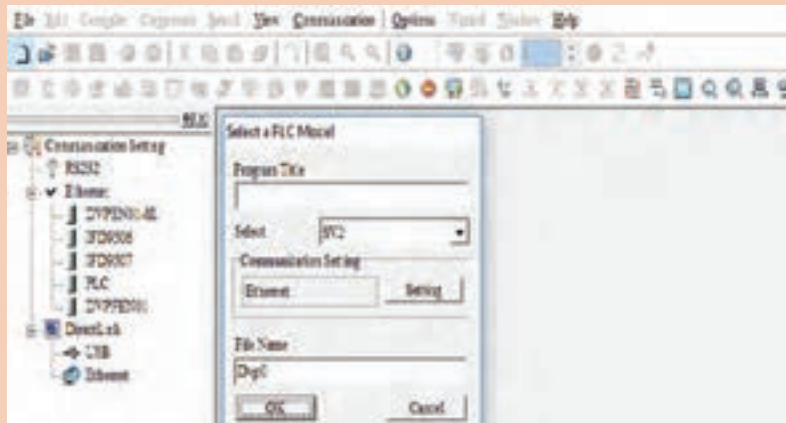
۳

بودمان سوم: نصب و راه‌اندازی کنترل‌کننده‌های منطقی

فعالیت  
کارگاهی



نرم‌افزار «WPL» را اجرا کنید و مطابق دستورالعمل زیر یک مدار ساده را شبیه‌سازی کنید. از منوی فایل یا از نوار ابزار، گزینه «new» را انتخاب کنید، سپس مدل «CPU» مورد نیاز را تعیین کنید. لازم به ذکر است اگر مدل مشخص نباشد یک مدل دلخواه تعیین کنید و پس از مشخص شدن «CPU»، آن را از منوی «options» گزینه «change PLC type» به مدل موجود تغییر دهید.



مدل انتخابی و آدرس «station» و تعداد «step»‌های قابل برنامه‌نویسی برای «CPU» در نوار پایین نرم‌افزار قابل مشاهده است.

بدانید

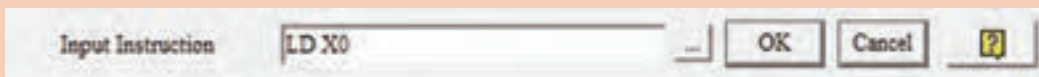


در نوار ابزار بالایی، تمام المان‌های مورد نیاز جهت برنامه‌نویسی وجود دارد.



دستورات برنامه‌نویسی در محیط نرم‌افزار به سه روش قابل استفاده است. برای مثال کنتاکت باز را به روش‌های زیر می‌توانیم وارد محیط برنامه‌نویسی کنیم:

- ۱ کلید میانبر F1؛
- ۲ انتخاب کنتاکت باز در نوار ابزار و تعیین عملوند و آدرس؛
- ۳ تایپ دستور LD X0.





## دستور کنتاکت باز

### LD Load A Contact



Operand: [Device Range](#)

X, Y, M, S, T, C

این دستور برای استفاده از وضعیت عملوند آدرس داده شده در برنامه به کار می‌رود و به صورت یک بیت با دو وضعیت صفر یا یک است. عملوندهای قابل استفاده در این دستور عبارت‌اند از:

- X بیت ورودی؛
- Y بیت خروجی؛
- M بیت حافظه؛
- S بیت STEP؛
- T تایمر؛
- C کانتر.

## دستور کنتاکت بسته

### LDI Load B Contact



Operand: [Device Range](#)

X, Y, M, S, T, C

این دستور برای استفاده از «NOT» وضعیت عملوند آدرس داده شده به کار می‌رود. عملوندهای قابل استفاده در این دستور مانند کنتاکت باز است.

## دستور «OUT»

### OUT Output Coil



Operand: [Device Range](#)

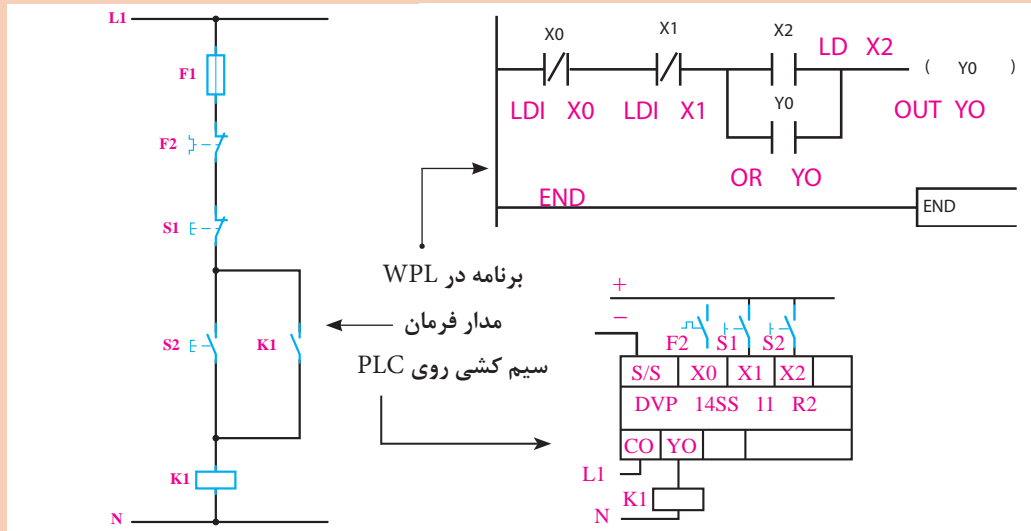
Y, M, S

از این دستور برای اعمال نتیجه به عملوند مورد نظر استفاده می‌شود و همیشه به خطوط برنامه قبل از دستور وابسته است و با تغییر وضعیت صفر به یک، عملوند مورد نظر مثل خروجی یک می‌شود و با صفر شدن وضعیت قبل از دستور عملوند نیز صفر می‌شود.





برنامه کنترل موتور از یک نقطه را بنویسید و آن را با شبیه ساز آزمایش کنید.



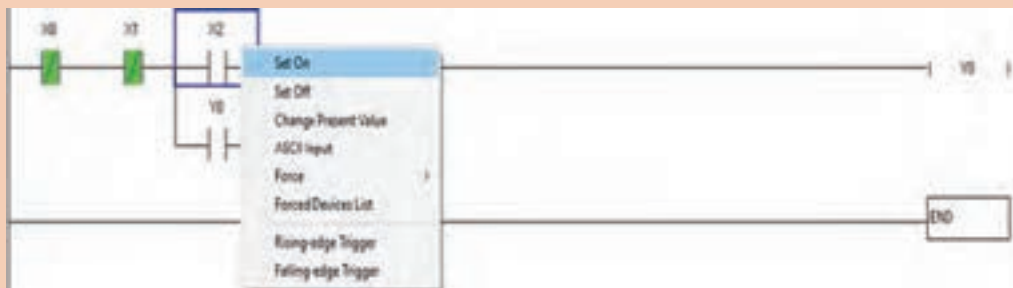
- پس از اتمام برنامه نویسی آن را save کنید و مراحل زیر را به ترتیب اجرا کنید.
- ۱ شبیه ساز را باز کنید؛
  - ۲ برنامه را دانلود کنید؛
  - ۳ تست آنلاین را بزنید.



پس از اجرای آنلاین اگر «CPU» در وضعیت «STOP» بود لازم است به وضعیت «RUN» تبدیل شود.



جهت تست، کافی است هر المانی را که باید تحریک شود انتخاب و موس را روی آن راست کلیک کنید. برای یک شدن گزینه «SET ON» و برای صفر شدن گزینه «SET OFF» را بزنید. لازم است تک به تک المان‌های مورد نیاز جهت تست مانند استپ و استارت صفر و یک شوند.



مدارهای زیر را که در پودمان اول طراحی کردید همانند مثال انجام شده، برنامه‌نویسی و سپس آنلاین تست کنید.

- ۱ کنترل موتور از دو نقطه؛
- ۲ کنترل دو موتور یکی به جای دیگری؛
- ۳ کنترل موتور به صورت چپ‌گرد - راست‌گرد ساده.

فعالیت  
کارگاهی



مدارهای زیر را برنامه‌نویسی و به صورت واقعی اجرا کنید. سپس آنلاین تست کنید.

- ۱ کنترل دو موتور یکی پس از دیگری؛
- ۲ کنترل موتور به صورت چپ‌گرد - راست‌گرد سریع؛
- ۳ کنترل موتور ۵۰ اسب بخار (به صورت ستاره مثلث).

فعالیت  
کارگاهی



زبان‌های برنامه‌نویسی در دلتا سه روش دارد که عبارت‌اند از:



- ۱ Ladder diagram (روش نردبانی)
- ۲ SFC Diagram mode (روش چارتی)
- ۳ Instruction List mode (روش عبارت کوتاه)

لازم به ذکر است در حال حاضر بیشتر از زبان «LAD» استفاده می‌شود ولی وقتی برنامه با یک زبان نوشته شود تبدیل برنامه در نرم‌افزار امکان‌پذیر است (در قسمت ابتدای نوار ابزار تبدیل‌ها وجود دارد).

بدانید





## سمبل‌نویسی «Symbol Table»

در برنامه‌های کنترل، که لازم است کار کنترلی گسترده انجام دهند، پس از الگوریتم‌نویسی برای ورودی‌ها و خروجی‌ها یک نام متناسب با پروژه، در قسمت «symbol table» وارد می‌کنیم و گزینه «show symbol» را فعال می‌کنیم. همچنین می‌توانیم برای هر المان توضیح (comment) بنویسیم. این کار باعث می‌شود حین برنامه‌نویسی در انتخاب آدرس‌ها کمتر خطا شود. در نوار ابزار شکل زیر المان‌های مربوط به فعال‌سازی و ویرایش سمبل و توضیحات مربوط به هر المان برنامه مشخص شده است. لذا لازم است برنامه‌های نوشته شده، از این مبحث به بعد سمبل‌نویسی و توضیح نیز داشته باشند.



### سمبل‌نویسی

فیلم



فعالیت



بدانید



برنامه‌ای برای کنترل موتور ۱۰۰ اسب بخار، به صورت چپ‌گرد - راست‌گرد، اجرا و برای ورودی و خروجی‌ها سمبل‌گذاری کنید. آیا می‌توانید ویژگی سمبل‌گذاری را بیان کنید؟

ساختار کلی دستورات در دلتا با فرمت زیر است و تعداد عملوندها در هر دستور متفاوت است.

عملگر	عملوند ۱ S1	عملوند ۲ S2	
-------	-------------	-------------	--

### عملگر

نوعی عمل منطقی است که باید انجام شود، مانند «SET» و «RESET» و توابع ریاضی مانند «ADD»

### عملوند

آدرسی است که باید عمل منطقی روی آن صورت گیرد و متناسب با مبنای دستور و نوع اطلاعات متفاوت است. در دستورات بیتی از عملوندهای بیتی مثل X و Y و M استفاده می‌شود. در عملوندهای بقیه مبنای از رجیستر D و E و F و برای اعداد ثابت در مبنای دسیمال از K استفاده می‌شود (مانند K10). برای اعداد در مبنای ۱۶ از فرمت H استفاده می‌شود (مانند H10).

هر تابع، عملگری است که بر روی تعدادی عملوند اثرگذار است. برای مثال اطلاعات مربوط به S1 در تابع ADD عبارت‌اند از:

$S_1: K, H, K_N X, K_N Y, K_N M, K_N S, T, C, D, E$

تعیین کنید کاربرد هر کدام از دیتاها چه زمانی است؟

پژوهش



## دستور «SET»

**SET S** s : Set device

Operand: **Device Range**

S : Y, M, S

در برنامه‌هایی که از دستور «OUT» استفاده شده متوجه شدیم جهت فعال ماندن نیاز به نگهدارنده داریم ولی در دستور «SET» وقتی مسیر یک شود محل مورد نظر نیز یک می‌شود و یک می‌ماند.

## دستور «RESET»

**RST S** s : Reset device

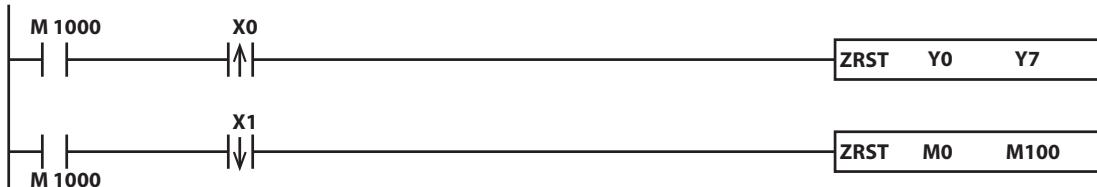
Operand: **Device Range**

S : Y, M, S, T, C, D, E, F

آدرس‌هایی که با دستور «SET» فعال شده‌اند فعال باقی خواهند ماند و جهت غیرفعال شدن نیاز به «RESET» دارند.

## دستور «ZRST»

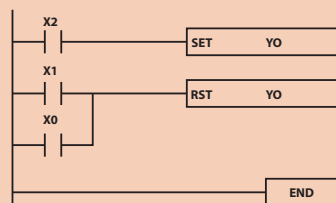
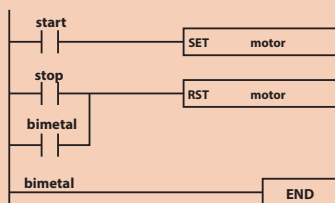
هرگاه بخواهیم تعداد زیادی عملوند Y و M را، که آدرس آنها به ترتیب است غیر فعال کنیم به جای استفاده از دستور «RST» که یک بیت را غیر فعال می‌کند و لازم است تعداد زیادی از این دستور نوشته شود از دستور «ZRST» استفاده می‌کنیم و با یک دستور کلیه عملوندها را غیرفعال می‌کنیم. در دستور زیر با فعال شدن X0 همزمان خروجی Y0 تا Y7 و حافظه M0 تا M100 غیرفعال می‌شود.



■ هر آدرس، مثلاً خروجی در طول برنامه می‌تواند چندین بار «SET» یا «RESET» شود.

در صورتی که خروجی «OUT» فقط یک بار قابل استفاده است و نباید تکرار شود.

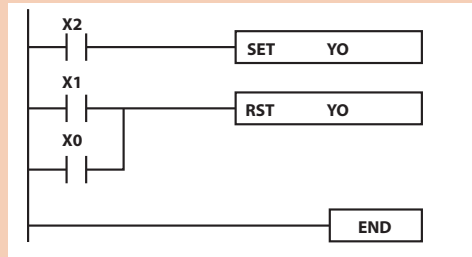
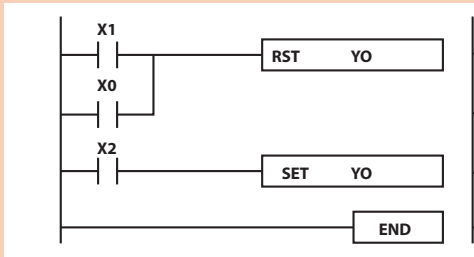
■ برنامه کنترل موتور از یک نقطه توسط دستورات «SET , RESET» نوشته شده که در برنامه سمت چپ سمبل فعال شده و در سمت راست سمبل غیرفعال است.



بدانید



- در این برنامه با زدن استارت X2 موتور روشن می‌شود و روشن می‌ماند و با زدن استپ X1 یا عمل کردن بی‌مثال X0 موتور خاموش می‌شود.
- در دو برنامه زیر، برنامه سمت راست «RESET» مقدم است و برنامه سمت چپ «SET» مقدم است و معمولاً ما از برنامه «RESET» مقدم استفاده می‌کنیم.
- اگر استارت و استپ را به‌طور هم‌زمان بزنیم در برنامه، «RESET» مقدم، خروجی خاموش است ولی در برنامه «SET» مقدم خروجی روشن است زیرا «SET» است.



- اجرای برنامه در «PLC» خط به خط است یعنی خط اول خوانده و اجرا می‌شود، سپس خط دوم و به ترتیب ادامه می‌یابد تا به دستور «END» برسد و سیکل تکرار شود. به همین دلیل دستورات بعد بر دستورات قبل از خود اولویت دارند، زیرا دیرتر اجرا می‌شوند.

فیلم



برنامه‌نویسی با دستورات «SET , RESET»

فعالیت  
کارگاهی



مدارهای زیر را با روش «SET , RESET» برنامه‌نویسی کنید سپس آنلاین تست کنید و در پایان عملی انجام دهید. (حتماً سیم‌کشی انجام شود و کارها واقعی تست شوند)

- ۱ کنترل موتور از دو نقطه؛
- ۲ کنترل دو موتور یکی پس از دیگری؛
- ۳ کنترل موتور به صورت چپ‌گرد - راست‌گرد سریع؛
- ۴ کنترل موتور ۵۰ اسب بخار (ستاره - مثلث).

## دستور (عملگر) «LDP»

**SET S** s : Set device

Operand: [Device Range](#)

S: Y, M, S

این دستور آشکارسازی لبه بالارونده سیگنال است. مانند لحظه وصل کلید که صفر به یک تبدیل می‌شود، در لحظه وصل، یک پالس ۱۰۰ میلی ثانیه تولید و فرمان را به محل مورد نظر صادر می‌کند.

## دستور (عملگر) «LDF»

**RST S S : Reset device**

Operand: Device Range

S : Y, M, S, T, C, D, E, F

این دستور، آشکارسازی لبه پایین رونده سیگنال است. مانند لحظه قطع کلید که یک به صفر تبدیل می‌شود، در لحظه قطع، یک پالس ۱۰۰ میلی‌ثانیه تولید و فرمان را به محل مورد نظر صادر می‌کند.



در این برنامه، لحظه وصل  $X0$ ، خروجی  $Y0$  روشن می‌شود ولی در لحظه تحریک  $X1$  هیچ اتفاقی نمی‌افتد بلکه در لحظه قطع  $X1$ ، خروجی  $Y0$  قطع می‌شود.

بدانید



فیلم



کاربری «LDP» و «LDF»

فعالیت کارگاهی



کاربرد دستور «LDP» و دستور «LDF» را بررسی کنید و در مدار راه‌اندازی موتور ۱۰۰ اسب بخار چپ‌گرد - راست‌گرد، استارت‌های شروع را با دستور LDP اجرا و سپس بررسی کنید کدام روش درست است (استارت با لبه یا بدون لبه)

فعالیت کارگاهی



برنامه‌ای بنویسید که لحظه تحریک استارت، موتور «ستاره» و با قطع شدن استارت، «مثلث» شود.

بدانید



M در دلتا حافظه بیتی هستند و با شماره M0 و M1 و... است که جهت ذخیره مقادیر میانی از آنها به صورت بیتی استفاده می‌شود.  
D در دلتا حافظه ۱۶ بیتی هستند که جهت ذخیره اعداد و اطلاعات از ۸ بیت، ۱۶ بیت و ۳۲ بیت استفاده می‌شود و با شماره D0 و D1 و... هستند.  
یادآوری می‌شود که M, D در دلتا سه نوع هستند:

۱ ناپایدار (GENERAL)؛

۲ پایدار (LATCH)؛

۳ مخصوص (SPECIAL).



کاربرد M چیست ؟  
 وضعیت حافظه از M0 تا M4095 را بررسی کنید و کاربرد هر محدوده را بنویسید.  
 کاربرد رجیستر D چیست ؟  
 وضعیت رجیستر از D0 تا D4999 را بررسی کنید و کاربرد هر محدوده را بنویسید.

## تایمر «Timer»

از تایمر جهت زمان سنجی استفاده می شود. در مدارهای اتوماتیک و جاهایی که لازم است پس از سپری شدن زمان، فرمان به یک محل صادر شود از تایمر استفاده می شود.  
 لازم به ذکر است تمام تایمرها در دلتا از نوع «on delay timer» هستند و پله زمانی مشخصی دارند. برای تعریف زمان لازم است پله زمانی را بدانیم و بدین منظور باید به کاتالوگ «CPU» مورد نظر مراجعه کنیم. طبق فرم زیر تایمرهای T0 تا T199 با پله زمانی 100ms و T200 تا T245 با پله زمانی 10ms و T246 تا T249 با پله زمانی 1ms است.

Timer	100 ms	T0~T199, 200 points (*1)	total 256 points	When the timer that set by TMR command reaches the preset value. the T contact with the same number will be on.
		T192~T199 for Subroutine		
	10 ms	T250~T255, 6 points Accumulative (*4)		
		T200~T239, 40 points (*2)		
1 ms	T240~T245, 6 points Accumulative (*4)			
	T246~T249, 6 points Accumulative (*4)			

دستور تایمر، به فرم روبه روست.

TMR	S <sub>1</sub>	S <sub>2</sub>
-----	----------------	----------------

S<sub>1</sub> : Timer number 100ms شماره تایمر است. که در نوع  
 S<sub>2</sub> : set value از T0 تا T199 است. S<sub>2</sub> عدد تنظیم مقدار  
 زمان تایمر است برای محاسبه زمان واقعی  
 تایمر.

Operand: Device Range

S<sub>1</sub> : T

S<sub>2</sub> : K.D

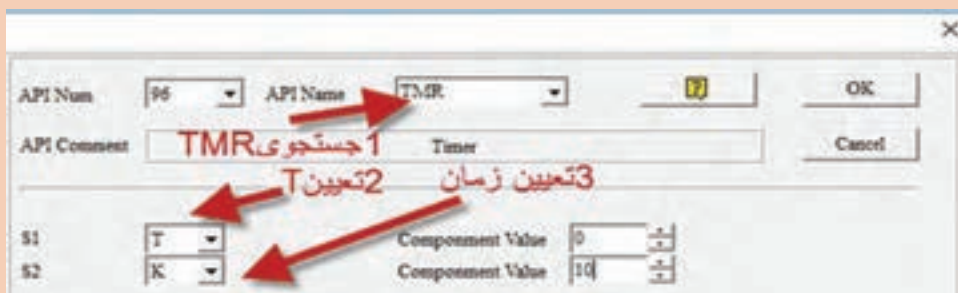
زمان واقعی تایمر = عدد تنظیم (S<sub>2</sub>) × پله  
 زمانی متناسب با شماره تایمر

مثلاً برای تایمر T0، که قرار است ۱۰ ثانیه کار

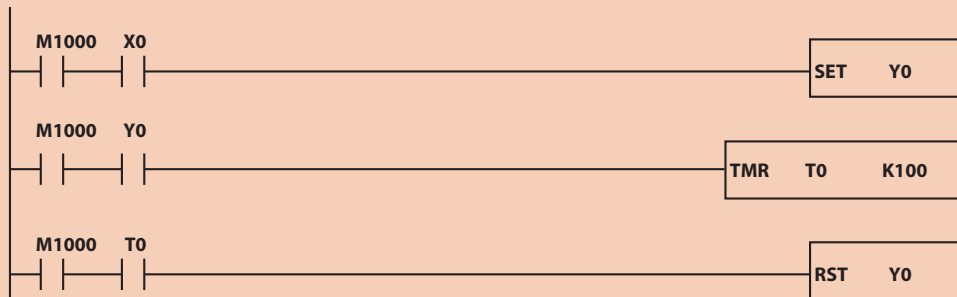
کند، عدد تنظیمی (S<sub>2</sub>) را K100 تعریف می کنیم و در صورت نیاز به زمان متغیر در قسمت عدد تنظیمی، از رجیستر «D» استفاده می شود (مثلاً D0).



دستور تایمر علاوه بر تایپ TMR T0 K10 می تواند از طریق تابع میانبر F6 یا از نوار ابزار اجرا شود. برای استفاده از F6 یا نوار ابزار به شکل زیر عمل می کنیم.



در برنامه زیر وقتی X<sup>o</sup> تحریک شود خروجی Y<sup>o</sup> روشن و تایمر زمان ۱۰ ثانیه را سپری می کند سپس Y<sup>o</sup> خاموش می شود.



آموزش برنامه نویسی با تایمر

فیلم



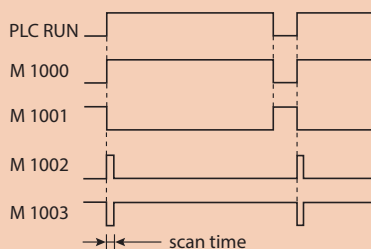
فعالیت کارگاهی



مدارهای زیر را اجرا کنید. زمان های ذکر نشده دلخواه هستند.

- ۱ کنترل دو موتور به صورت یکی پس از دیگری اتوماتیک؛
- ۲ کنترل دو موتور به صورت یکی به جای دیگری اتوماتیک؛
- ۳ کنترل موتور ستاره مثلث اتوماتیک؛
- ۴ برنامه ای بنویسید که با زدن استارت، موتور ۱ روشن و بعد از ۱۰ ثانیه موتور ۲ روشن و بعد از یک ساعت، کل مدار قطع شود.

بدانید



■ M1000 با «RUN» شدن «CPU» یک می شود؛ بنابراین معمولاً در ابتدای هر خط از آن استفاده می شود.

■ M1001 با «STOP» شدن «CPU» یک می شود.

## کانتر یا شمارنده «Counter»

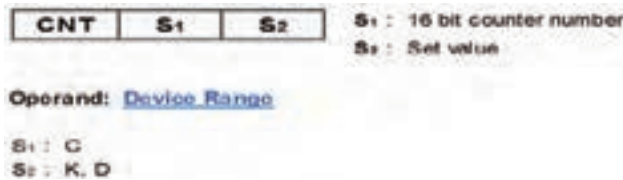
از کانتر جهت شمارش، استفاده می شود. مانند شمارش قطعات در خطوط تولید، شمارش قطعات ورودی و خروجی انبار، شمارش ماشین در پارکینگ ها و...

C	Counter	16-bit count up		total 256 points	When the counter that set by CNT command reaches the preset value, the C contact with the same number will be on.
		C0~C95, 96 points (*1)			
32-bit count up		C200~C215, 16 points (*1)			
		C216~C234, 19 points (*3)			
32-bit high-speed count up/down		C235~C245, 1 phase 1 input, 9 points (*3)			
		C246~C250, 1 phase 2 input, 3 points (*3)			
		C251~C254, 2 phase 2 input, 3 points (*3)			

در مدل دلتا سه نوع کانتر در اختیار شماست:

- ۱- کانتر ۱۶ بیتی که همه از نوع بالا شمار هستند (C0 تا C199)؛
- ۲- کانتر ۳۲ بیتی که براساس نیاز، به کمک حافظه می تواند بالا شمار یا پایین شمار شود (C200 تا C234)؛
- ۳- کانتر ۳۲ بیتی برای شمارش سریع از نوع بالا و پایین شمار که کاربرد فراوانی در خطوط تولید دارد مانند شمارش پالس های انکودر.

### ۱- کانتر ۱۶ بیتی



فقط بالا شمار است و در خطوط تولید برای شمارش قطعات و بسته بندی کاربرد دارد. محدوده قابل شمارش توسط آنها تا ۳۲۷۶۷ است و فرمت دستور آن به شکل روبه رو است:

اگر عدد شمارش شده توسط کانتر برابر با عدد تعریف شده در S2 باشد کانتر فعال می شود و می توانیم از کنتاکت آن برای فرمان استفاده کنیم.

فیلم

آموزش برنامه نویسی با شمارنده



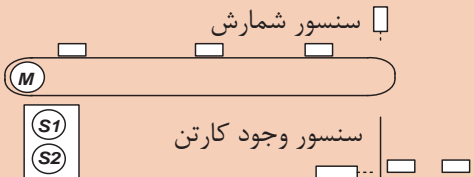
فعالیت

برنامه ای بنویسید که با زدن استارت، موتور روشن و با زدن استپ، موتور خاموش شود. اگر موتور بیش از ده مرتبه خاموش و روشن شد کل مدار خاموش شود و دیگر استارت نشود.



فعالیت

برنامه کنترل بخشی از خط تولید را به نحوی بنویسید که با زدن «S1» با شرط وجود کارتن، تسمه نقاله شروع به کار کند و قطعات را انتقال دهد.



قطعات شمارش شوند و اگر تعداد به ۱۰ رسید تسمه متوقف شود تا کارتن برداشته شود و با گذاشتن کارتن بعدی، مجدد ادامه دهد و این روند ادامه یابد تا وقتی که سیستم استپ (S2) شود.



### ۲- کانتر ۳۲ بیتی

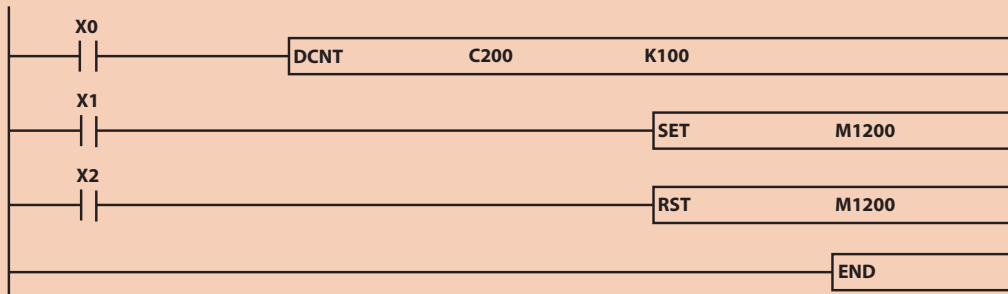
این کانتر قادر است براساس نیاز، بالا یا پایین شمار باشد و به این منظور برای هر کانتر یک حافظه خاص در نظر گرفته شده است، مانند M1200 برای C200 و M1201 برای C201 تا M1234 برای C234. اگر حافظه مربوطه غیرفعال باشد، کانتر بالا شمار و اگر فعال شود کانتر پایین شمار می شود. فرمت برنامه نویسی کانتر ۳۲ بیتی به یکی از دو فرم زیر است.

DCNT C200 K...

DCNT C200 D...

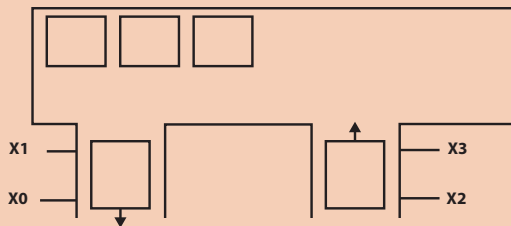


با تحریک X۰ کانتر «C200» شمارش می کند ولی با تحریک X1 و X2 شمارش متفاوت خواهد شد. وقتی X1 تحریک شود حافظه «M1200» فعال می شود و کانتر پایین شمار می شود و با تحریک X2 حافظه «M1200» غیرفعال می گردد و کانتر بالا شمار عمل می کند.



برنامه کنترل پارکینگ را به صورت زیر بنویسید.  
 ■ ظرفیت پارکینگ ۱۰۰ خودرو، تعیین شده است. با ورود خودرو، از ظرفیت کم و با خروج آن، به ظرفیت اضافه شود و ظرفیت موجود مشخص شود.

X2, X3 حسگر مسیر ورودی  
 X0, X1 حسگر مسیر خروجی



## مقایسه گر «comparator»

در برنامه نویسی گاهی نیاز است که مقداری را با مقداری دیگر مقایسه کنیم. برای مثال اگر تعداد محصولی که تولید کردیم (از جلوی حسگر عبور کرده است) بیشتر از یک مقدار مشخص شد، می خواهیم دستگاه خاموش شود و آلام مخصوص سرویس و نگهداری به صدا در آید. یا اگر مقدار دمای محیط از دمایی که ما تعیین کردیم بیشتر یا کمتر یا مساوی بود، می خواهیم خروجی متناسب با آن وضعیت فعال شود. در این مواقع می توان با استفاده از دستورات مقایسه کننده این کار را انجام داد.

توجه داشته باشید در هنگام محاسبات ریاضی لازم است دو مقدار دارای مبنای مشترک باشند. اگر مبنای یکسان نباشند لازم است مقادیر با مبنای متفاوت توسط مبدل ها به مبنای مورد نظر تبدیل شوند.

■ برای برنامه نویسی در «PLC» های دلتا انواع مقایسه کننده هایی که می توان برای راحتی کار از آنها استفاده کرد، موجود است.

■ دستورات مقایسه ای بر پایه «LD»:



این دستور انواع مختلفی دارد و به شکل‌های زیر مورد استفاده قرار می‌گیرد:

**۱- اعداد صحیح ۱۶ بیتی integer**

=LD و <LD و >LD و <>LD و =LD و =>LD

**۲- اعداد صحیح ۳۲ بیتی Doble integer**

=DLD و <DLD و >DLD و <>DLD و =DLD و =>DLD

**۳- اعداد اعشاری Real (float)**

=FLD و <FLD و >FLD و <>FLD و =FLD و =>FLD

فرمت کلی دستور مقایسه‌کننده‌ها به شکل زیر است.

**API 224~230 LD\* Contact Comparison**

LD*	S <sub>1</sub>	S <sub>2</sub>	S <sub>1</sub> : Data source device 1
			S <sub>2</sub> : Data source device 2

Operand: [Device Range](#)

S<sub>1</sub> : K, H, KnX, KnY, KnM, KnS, T, C, D, E, F

S<sub>2</sub> : K, H, KnX, KnY, KnM, KnS, T, C, D, E, F

S1 مقدار خوانده شده از اِلمان اول است، مثل کانتر یا D و...

S2 مقدار خوانده شده از اِلمان دوم است، مثل کانتر یا D یا عدد ثابت و...

در مقایسه‌کننده مقدار S1 با S2 مقایسه شده و در صورتی که شرط مقایسه برقرار باشد خروجی مقایسه‌کننده، فعال می‌شود و به محل عملوند مورد نظر بعد از فرمان اعمال می‌گردد.

■ برای مثال، اِلمان اول می‌تواند مقدار خوانده شده از حسگر دما و اِلمان دوم یک عدد ثابت باشد یعنی دما با یک مقدار ثابت مقایسه شود.

■ یا اِلمان اول می‌تواند مقدار حسگر دما و اِلمان دوم دمای تعیین شده از «HMI» باشد که با رجیستر D از «HMI» به «CPU» اعمال می‌شود و...

در خط برنامه، با فعال شدن شرط اجرای دستور مقایسه X0 زیر اگر مقدار D0 از عدد ۱۰ بیشتر شود خروجی Y0 فعال می‌گردد تا زمانی که مقدار D0 از ۱۰ کمتر و Y0 غیرفعال شود.

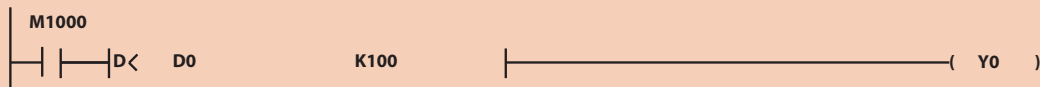


بدانید

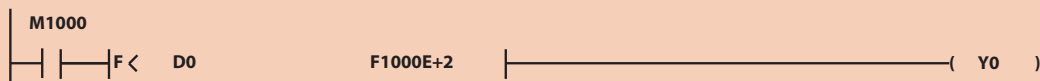




این خط دستور مقایسه ۱۶ بیتی اعداد صحیح است که اطلاعات D0 را با عدد 100 مقایسه می کند و اگر D0 از 100 کمتر شد Y0 روشن می شود.



این خط دستور مقایسه ۳۲ بیتی اعداد صحیح است که اطلاعات D0 را با عدد 100 مقایسه می کند و اگر D0 از 100 کمتر شد Y0 روشن می شود.



این خط دستور مقایسه ۳۲ بیتی اعداد اعشاری است که اطلاعات D0 را با عدد 100.0 که عدد اعشاری است مقایسه می کند و اگر D0 از 100 کمتر شد Y0 روشن می شود.



با دستور «INC» و «DEC» می توانیم کار شمارش را انجام دهیم. برای درک بهتر این دستور به مثال ترکیبی زیر توجه کنید:

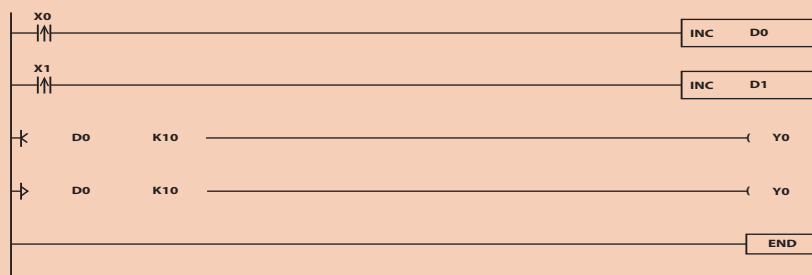
پارکینگی را در نظر بگیرید، که برای ۱۰ خودرو ظرفیت دارد و در ورودی پارکینگ حسگر X0 تعبیه شده است تا تعداد خودروهای ورودی را بشمارد.

در خروجی پارکینگ نیز حسگر X1 تعبیه شده است تا تعداد خودروهای خروجی را بشمارد. تعداد خودروی باقی مانده در پارکینگ توسط دستورات مقایسه کننده مقایسه می شود و خروجی متناسب فعال می گردد.

الف) در صورتی که تعداد خودرو در پارکینگ، از ۱۰ خودرو کمتر باشد خروجی Y0، که به تابلوی «وارد شوید» متصل است فعال می شود.

ب) در صورتی که تعداد خودرو در پارکینگ، بیشتر یا مساوی ۱۰ خودرو باشد خروجی Y1 که به تابلوی «ظرفیت تکمیل است» متصل است فعال می شود.

در این مثال، D0 ظرفیت پارکینگ است.



فیلم

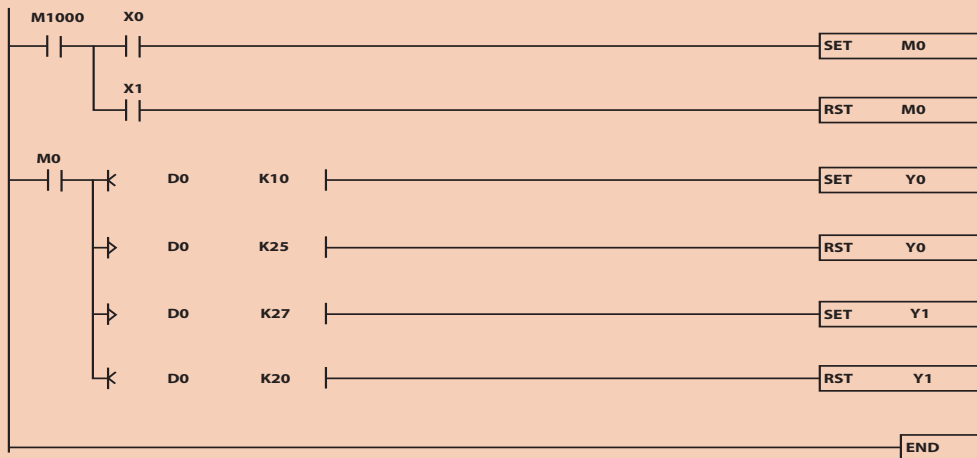
آموزش برنامه نویسی با مقایسه گر



فعالیت



عملکرد برنامه زیر را با کمک هنرآموز بررسی و آن را در شبیه ساز تست و کاربرد آن را نیز مشخص کنید.



برای تست شبیه ساز در محیط برنامه، لازم است به رجیستر مقداردهی کنیم. نحوه مقداردهی پس از فعال کردن سیمولاتور، به صورت زیر است.



فعالیت

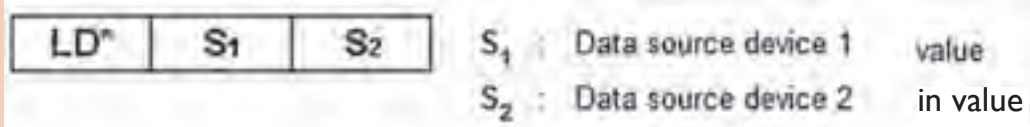


فرض کنید دمای یک کوره در رجیستر D۰ ذخیره می شود. برنامه ای بنویسید که متناسب با تغییر دمای کوره به صورت زیر عمل کند:

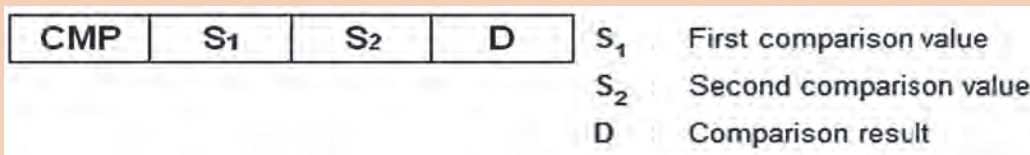
- ۱ اگر دما کوچک تر از ۱۰۰ شد ۳ مشعل روشن شود؛
- ۲ اگر دما بزرگ تر از ۱۰۰ و کوچک تر از ۲۰۰ شد ۲ مشعل روشن شود؛
- ۳ اگر دما بزرگ تر از ۲۰۰ و کوچک تر از ۳۰۰ شد ۱ مشعل روشن شود؛
- ۴ اگر دما بزرگ تر از ۳۰۰ و کوچک تر از ۳۵۰ شد ۳ مشعل خاموش شود؛
- ۵ و اگر دما بزرگ تر از ۴۰۰ شد سیستم خنک کننده روشن شود.



علاوه بر دستور «LD» برای مقایسه‌کننده‌ها دستور «CMP» نیز وجود دارد. نتیجه دستور «LD» به صورت بیتی است که بعد از دستور می‌تواند فرمان لازم را صادر کند.



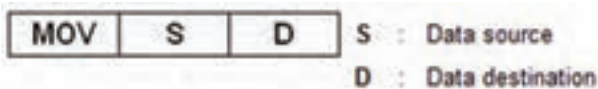
ولی دستور «CMP» دو مقدار عملوند S<sub>1</sub> و S<sub>2</sub> را با هم مقایسه می‌کند و نتیجه آن در عملوند D، سه حالت ایجاد می‌کند. بنابراین هر آدرسی برای آن تعیین شود دو بیت بعدی نیز استفاده می‌شود. مثلاً اگر M0 معرفی شود M1 و M2 نیز اشغال می‌شود. بنابراین در ادامه از M3 به بعد می‌توانیم استفاده کنیم.



اگر  $D0 < K100$  بود، M0 فعال می‌شود؛  
اگر  $D0 = K100$  بود، M1 فعال می‌شود؛  
اگر  $D0 > K100$  بود، M2 فعال می‌شود.



## دستور «MOV»



Operand: [Device Range](#)

S : K, H, KnX, KnY, KnM, KnS, T, C, D, E, F

D : KnY, KnM, KnS, T, C, D, E, F

از این تابع برای انتقال اطلاعات (بارگذاری و انتقال) استفاده می‌شود و زمانی که لازم باشد اطلاعات یک رجیستر یا یک عدد ثابت را به رجیستر دیگری منتقل کنیم کاربرد دارد. فرمت تابع به شکل روبه‌رو است:





با تحریک X0 در چهار خط برنامه زیر چه اتفاقی می‌افتد و تفاوت آنها در چیست؟

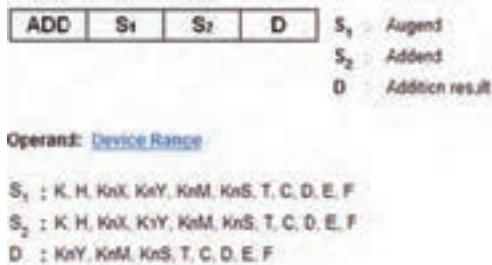


## توابع ریاضی

توابع و دستورات پر کاربرد محاسبه ریاضی مانند ADD و SUB و Div و Mul و... نیز همانند دستور Mov و دستور مقایسه‌کننده برای هر سه مینا وجود دارد و هرگاه لازم باشد روی دو دیتا (data) عملیات ریاضی انجام شود باید دقت شود که دارای مینای مشترک باشند. برای مثال هر دو دیتا از نوع اعشاری باشند.

### ۱- دستور «ADD»

برای جمع دو دیتای اعداد صحیح، از نوع ۱۶ بیتی استفاده می‌شود و حاصل را می‌تواند در رجیستر ذخیره کند:



S<sub>1</sub> مقدار دیتای اول

S<sub>2</sub> مقدار دیتای دوم

D حاصل جمع ذخیره شده در رجیستر

$$D = S_1 + S_2$$

### ۲- دستور «SUB»

برای تفریق دو دیتای اعداد صحیح، از نوع ۱۶ بیتی استفاده می‌شود و حاصل را در رجیستر می‌تواند ذخیره کند.



$$D = S_1 - S_2$$

### ۳- دستور «DIV»

برای تقسیم دو دیتای اعداد صحیح، از نوع ۱۶ بیتی استفاده می‌شود و حاصل را در رجیستر می‌تواند ذخیره کند.



$$D = S_1 / S_2$$

#### ۴- دستور «MUL»

MUL	S1	S2	D
-----	----	----	---

برای ضرب دو دیتای اعداد صحیح، از نوع ۱۶ بیتی استفاده می‌شود و حاصل را در رجیستر می‌تواند ذخیره کند.

$$D = S_1 \times S_2$$

فیلم

آموزش توابع ریاضی



فعالیت کارگاهی



تفاوت دستور «Add» با دستور «Dadd» و دستور «Daddr» در چیست؟ این تفاوت در بقیه دستورات ریاضی نیز بررسی شود. در محیط نرم‌افزار، توابع ریاضی زیر را برنامه‌نویسی و جواب به دست آمده را تعیین کنید. آیا جواب به دست آمده با جواب ماشین حساب مطابقت دارد؟

$$F_1 = \frac{45 \times 14}{2}$$

$$F_2 = \frac{45 \times 14}{3} + 3 - 4/2$$

برنامه‌ای بنویسید که با تحریک X0 حالت‌های زیر ایجاد شود؟

در برنامه‌نویسی فقط از اعداد هگزا دسیمال استفاده شود:

- ۱ با تحریک X0 تعداد ۸ خروجی هم‌زمان روشن شود؛
- ۲ با تحریک X1 از ۸ خروجی به صورت یک در میان بیت‌های زوج روشن شود؛
- ۳ با تحریک X1 از ۸ خروجی به صورت یک در میان بیت‌های فرد روشن شود؛
- ۴ با تحریک X2 تعداد ۸ خروجی غیر فعال شوند.

### برنامه‌نویسی سازمان یافته

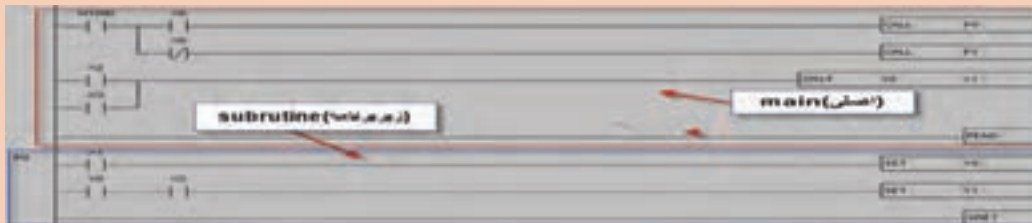
برنامه‌هایی که تا قبل از این مبحث نوشته شده از نوع برنامه‌نویسی خطی<sup>۱</sup> است. لازم است کل برنامه کنترل در «Main» نوشته شود و در حین اجرا، برنامه خط به خط اجرا گردد. حتی اگر دستور یا خطی در یک سیکل نیاز نباشد، «CPU» مجبور به اجرای دستور است. این کار باعث می‌شود تعداد خط‌های برنامه برای «CPU» حین اجرا زیاد باشد که در برنامه‌های سنگین ممکن است «CPU» از نظر «CYCLE TIME» دچار مشکل شود. بنابراین گاهی اوقات مجبور به تعویض و استفاده از «CPU» با سرعت پردازش بالاتر می‌شویم. این راه اصولی نیست و از نظر اقتصادی نیز مقرون به صرفه نیست. به همین دلیل از برنامه‌نویسی سازمان یافته استفاده می‌کنیم. در این روش، برنامه‌نویسی شامل یک Main و یک سری زیر برنامه<sup>۲</sup> است و می‌توان هر زیر برنامه را (Subroutine) به صورت شرطی یا غیرشرطی اجرا کرد.

۱- Linear Program

۲- Subroutine

این کار باعث می شود همیشه «CPU» فقط با قسمتی از برنامه درگیر باشد و زمان «scan time» کاهش یابد. همچنین می توانیم از «CPU»هایی که سرعت پردازش کمتری دارند برای کنترل های وسیع نیز استفاده نماییم، به همین دلیل از نظر اقتصادی به صرفه تر خواهد شد. تعداد زیربرنامه ها بستگی به مدل «CPU» دارد بنابراین لازم است قبل از برنامه نویسی سازمان یافته، کاتالوگ «CPU» بررسی شود و تعداد زیربرنامه های قابل اجرا برای «CPU» شناسایی گردد.

زیربرنامه ها (Subroutine) با P نام گذاری می شوند و از آنها برای فراخوانی از دستور «CALL» استفاده می شود و لازم است پایان هر زیر برنامه و برنامه اصلی تعیین شود.



- با فعال شدن X0 زیر برنامه P0 اجرا می شود و با غیرفعال بودن X0 زیر برنامه P1 اجرا می شود.
- «RESET» اصلی بهتر است در برنامه «MAIN» نوشته شود.
- پایان «MAIN» با «FEND» تعیین می شود.
- پایان هر زیربرنامه باید با «SRET» تعیین شود.

نحوه استفاده از برنامه نویسی سازمان یافته

- با استفاده از برنامه نویسی سازمان یافته برنامه کنترل به شرح زیر بنویسید (تمام نکات ایمنی لحاظ شود). دستگاه دارای یک کلید انتخاب ۲-۱ است.
- اگر کلید انتخاب در وضعیت ۱ باشد، با زدن استارت ۱، موتور ۱ روشن و بعد از ۱۰ ثانیه موتور ۲ روشن و بعد از ۲۰ ثانیه مدار خاموش شود.
- اگر کلید انتخاب در وضعیت ۲ باشد، با زدن استارت ۱، موتور ۱ روشن و با زدن استارت ۲ موتور ۲ روشن شود و با زدن استپ کل مدار قطع شود.

بدانید



فیلم

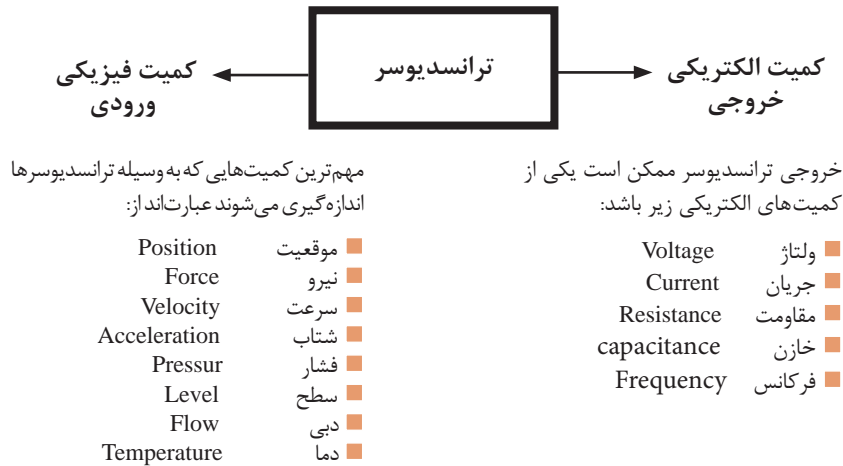


فعالیت  
کارگاهی



## ورودی آنالوگ

هرگاه لازم باشد کمیت‌ها مانند نیرو، دما، فشار، وزن و... به صورت پیوسته کنترل شوند، از ورودی آنالوگ استفاده می‌شود، به این صورت که توسط حسگرها کمیت اندازه‌گیری شده به یک کمیت الکتریکی استاندارد تبدیل می‌شود و به ورودی آنالوگ «PLC» ارسال می‌شود.



ورودی آنالوگ در دلتا به دو صورت است:

- 1- ترمینال‌های ورودی آنالوگ موجود در «CPU» که از نوع ولتاژی و جریانی است.
- 2- کارت ورودی آنالوگ که به مجموعه «CPU» اضافه می‌شود و متناسب با نیاز، جهت سیستم کنترل می‌توان انواع مختلفی را انتخاب و به مجموعه اضافه کرد. مانند:

DVP 04AD - DVP06AD - DVP 04PT - DVP 04TC

فیلم

نحوه نوشتن برنامه به صورت آنالوگ



### 1- ترمینال‌های ورودی آنالوگ موجود در «CPU» :

هر کانال ورودی آنالوگ روی «CPU» دارای انواع ولتاژ و جریان است و بر اساس نیاز می‌توانیم از هر کانال، نوع ولتاژ یا جریان آن را مورد استفاده قرار دهیم. برای هر کانال یک رجیستر مخصوص وجود دارد که مقدار تغییرات حسگر متصل شده به آن در رجیستر مربوطه ذخیره می‌شود. رجیستر مربوط به هر کانال مشخص است:

- D1110 مربوط به مقدار میانگین کانال اول؛
- D1111 مربوط به مقدار میانگین کانال دوم؛
- D1118 زمان نمونه برداری بر حسب ms؛
- D1056 مقدار لحظه به لحظه کانال اول؛
- D1057 مقدار لحظه به لحظه کانال دوم.



بدانید



- «CPU 10sx» دارای دو کانال ورودی و دو کانال خروجی آنالوگ روی «CPU» است.
- «CPU 20ex» دارای چهار کانال ورودی و دو کانال خروجی آنالوگ روی «CPU» است.
- مقادیر میانگین در مواردی که تغییرات زیاد است به کار می‌رود، مانند سطح آب که ممکن است در حین پر شدن، نوسان زیادی داشته باشد، می‌توانیم در یک زمان مشخص میانگین بگیریم.
- مقدار لحظه به لحظه را در مواقعی که خواهیم همه تغییرات در هر لحظه اندازه‌گیری شود، مورد استفاده قرار می‌دهیم.
- اگر سیم‌کشی کانال درست انجام شود در این صورت مثلاً در کانال 1 چنانچه مقدار میانگین لازم باشد کافی است D1110 و چنانچه مقدار لحظه به لحظه مورد نیاز باشد D1056 را در برنامه استفاده کنیم.

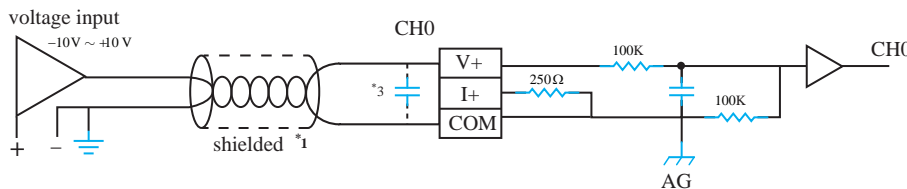
فعالیت



کارت‌های آنالوگ را از نظر مشخصات بررسی کنید. همچنین روش سیم‌کشی یک کانال در هر کارت را متناسب با نوع آن معلوم کنید.

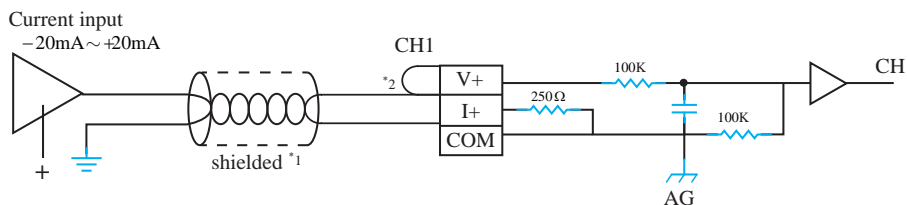
### الف) ورودی آنالوگ از نوع ولتاژ

معمولاً حسگرهایی که خروجی استاندارد 0-10VDC یا  $\pm 10VDC$  و... دارند به ترمینال  $V_0+$  و COM وصل می‌کنیم. (COM به منفی و  $V_0+$  به برگشت حسگر)



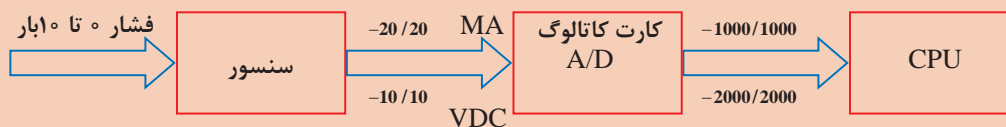
### ب) ورودی آنالوگ از نوع جریان

معمولاً حسگرهایی که خروجی استاندارد 0-20mA یا  $\pm 20mA$  و 4-20mA دارند به ترمینال « $V_0+$ ،  $I_0+$ ، COM» وصل می‌کنیم. (COM به منفی،  $I_0+$  و  $V_0+$  به هم متصل شده و به برگشت حسگر متصل می‌شود). معمولاً سیگنال جریان 4-20mA در صنعت بیشترین کاربرد را دارد.





خروجی حسگر که مثلاً  $10\text{ VDC} \pm$  است، در ورودی آنالوگ به دیجیتال تبدیل می‌شود، که با بررسی کاتالوگ، عدد  $-10$  تا  $+10$  ولت در کارت به عدد  $-2000$  تا  $+2000$  تبدیل و در «CPU» اعمال می‌شود که برنامه‌نویسی و معادل‌سازی آن کار نسبتاً طاقت فرسایی است. برای مثال فشار یک مخزن که بین  $0$  تا  $10$  بار است، عدد  $-2000$  در رجیستر تعیین شده «CPU» قرار می‌گیرد که باید بتوان عدد خوانده شده در رجیستر را به رنج  $0-10$  تبدیل نمود. اصطلاحاً به این عمل «scale» گفته می‌شود. بنابراین لازم است ابتدا ورودی آنالوگ خوانده شده با دستور «SCLP» به مقیاس مورد نظر تبدیل شود.



## دستور «SCLP»

از این دستور برای مقیاس و تبدیل مقادیر و اطلاعات استفاده می‌شود. برای مثال حسگر سطح با محدوده اندازه‌گیری صفر تا  $20$  متر طبق نمودار پایین به  $0$  تا  $10$  ولت و در «CPU» به عدد صفر تا  $2000$  تبدیل می‌شود که خیلی قابل فهم نیست. بنابراین می‌توانیم با این دستور عدد صفر تا  $2000$  را به همان رنج حسگر یعنی صفر تا  $20$  تبدیل کنیم.

SCLP	S <sub>1</sub>	S <sub>2</sub>	D
------	----------------	----------------	---

S<sub>1</sub> : Source value  
 S<sub>2</sub> : Parameters  
 D : Operation result

Operand: Device Range

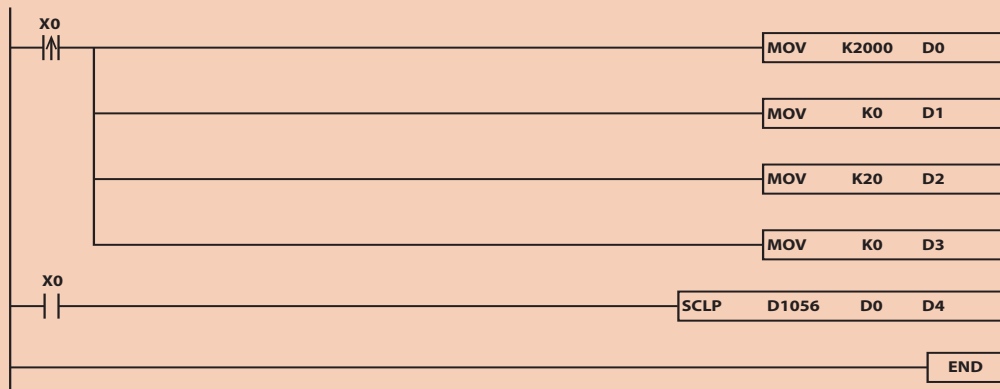
S<sub>1</sub> : K, H, D  
 S<sub>2</sub> : D  
 D : D

S<sub>1</sub> : مقدار خوانده شده از حسگر یعنی ورودی آنالوگ است.  
 S<sub>2</sub> : مقادیر «MIN» و «MAX» برای تبدیل هستند.  
 D : حاصل تبدیل به مقیاس را در محل آدرس داده شده ذخیره می‌کند و برای برنامه‌نویسی از آن استفاده می‌کنیم.

بدانید



لحظه‌ی یک شدن X0 عدد ۲۰۰۰ به عنوان حد بالا در D0 و عدد صفر در D1 به عنوان حد پایین ذخیره می‌شود. در دستور بعدی عدد ۲۰ به عنوان حد بالای مقیاس‌دهی در D2 و عدد صفر به عنوان حد پایین مقیاس‌دهی در D3 ذخیره می‌شود. سپس با دستور «SCLP» مقدار اندازه‌گیری شده از ورودی آنالوگ D۱۰۵۶ توسط پارامترهای تعیین شده، که شروع آنها از D0 بود، در D4 ذخیره می‌شود، که عددی بین صفر تا ۲۰ است. در ادامه از D4 جهت بررسی و پردازش استفاده می‌شود.



فیلم



نحوه برنامه‌نویسی آنالوگ و SCALE کردن

فعالیت



برنامه‌ای بنویسید که دمای اتاق را از طریق کانال صفر بخواند و به صورت زیر عمل کند:

- اگر دما کمتر از ۱۵ درجه بود گرمکن روشن شود؛
- اگر دما بیش از ۲۳ درجه شد گرمکن خاموش شود؛
- اگر دما بیش از ۲۷ درجه شد کولر روشن شود؛
- اگر دما کمتر از ۱۹ درجه بود کولر خاموش شود.

فعالیت



برنامه‌ای بنویسید که دمای اتاق را از طریق کانال یک بخواند و به صورت زیر عمل کند:

- اگر دما کمتر از ۱۰۰ درجه بود ۳ مشعل روشن شود؛
- اگر دما بین ۱۰۰ تا ۲۰۰ درجه بود ۲ مشعل روشن شود؛
- اگر دما بین ۲۰۰ تا ۳۰۰ درجه بود ۱ مشعل روشن شود؛
- اگر دما بین ۳۰۰ تا ۳۵۰ درجه بود همه مشعل‌ها خاموش شود؛
- اگر دما بالای ۳۵۰ درجه بود سیستم خنک‌کننده روشن شود.

## ۲- ترمینال‌های خروجی آنالوگ موجود در «CPU»

از خروجی آنالوگ جهت کنترل مقادیر پیوسته (مانند کنترل سرعت موتور، کنترل میزان باز و بسته شدن ولوها، کنترل میزان نور و...) استفاده می‌شود.

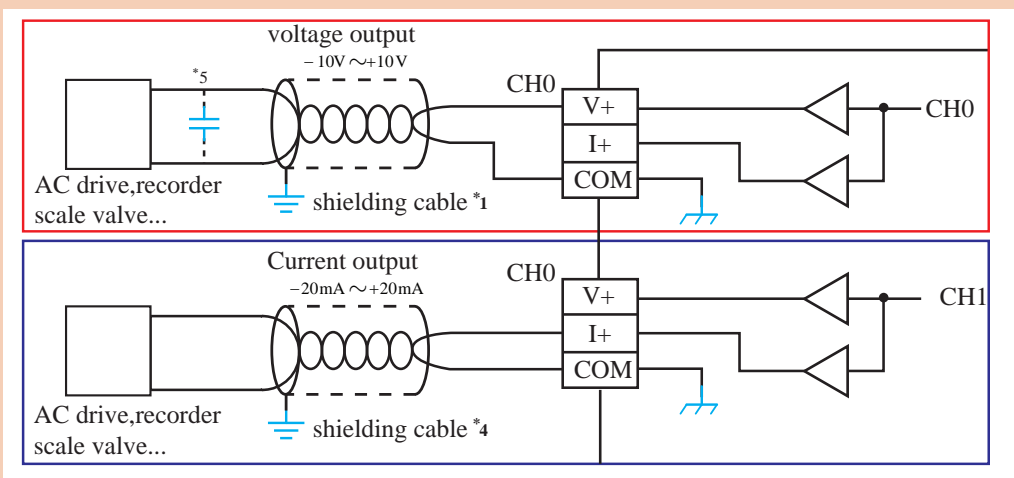
هر کانال خروجی آنالوگ روی «CPU» فقط دارای سیگنال نوع ولتاژ و جریان است و بر اساس نیاز می‌توان از هر کانال، نوع ولتاژ یا جریان آن را مورد استفاده قرار داد.

برای هر کانال یک رجیستر مخصوص وجود دارد که مقدار تغییرات حاصل از برنامه باید در آن ذخیره شود تا در کانال خروجی اعمال شود. رجیستر مربوط به هر کانال مشخص است و عبارت‌اند از:

■ D1116 خروجی کانال اول؛

■ D1117 خروجی کانال دوم.

- فرایند ارسال اطلاعات به خروجی آنالوگ برعکس ورودی آنالوگ است. اگر عدد را از مقیاس برگردانیم باید عدد 2000- تا 2000 را در رجیستر مثلاً D1116 اعمال کنیم تا در خروجی 4-20mA یا 0-20mA یا 10mA± برای جریانی‌ها و 10VDC- تا 10VDC± برای ولتاژی‌ها اعمال شود.
- کارت توسعه آنالوگ است و دارای چهار کانال خروجی است.



برنامه‌ای بنویسید که سیستم کنترل فشارآب مربوط به یک مجتمع ۲۰ واحدی را به صورت پیوسته انجام دهد. فشار توسط حسگر صفر تا ۲۰ بار از طریق کانال صفر خوانده شود و به صورت زیر عمل کند:

- اگر فشار کمتر از ۲ بار بود موتور با فرکانس ۵۰ هرتز کار کند؛
- اگر فشار بین ۲ تا ۳ بار بود موتور با فرکانس ۴۰ هرتز کار کند؛
- اگر فشار بین ۳ تا ۴ بار بود موتور با فرکانس ۲۵ هرتز کار کند؛
- اگر فشار بین ۴ تا ۵ بار بود موتور با فرکانس ۱۵ هرتز کار کند؛
- اگر فشار بالای ۶ بار بود موتور خاموش شود.

بدانید



فعالیت  
کارگاهی





سیستم کنترل اجرا شده در فعالیت کلاسی صفحه قبل یک سیستم پله ای است و ممکن است نوسانی شود. بنابراین در چنین مواقعی باید از سیستم حلقه بسته و سیستم کنترل «PID» استفاده شود تا بتوانیم یک سیستم پایدار طراحی کنیم. البته در صورتی که تعداد تقسیمات ریز شود، می توانیم به روش سعی و خطا، یک سیستم کنترل تقریباً متعادل طراحی کنیم ولی قطعاً ایده آل نخواهد بود.

## کارت های توسعه آنالوگ

کارت های توسعه آنالوگ نیز به دو گروه کارت های توسعه ورودی و خروجی تقسیم می شوند.

### ۱- کارت های توسعه آنالوگ ورودی

این کارت ها بستگی به نوع اندازه گیری دارند و بر اساس نیاز کنترل، انتخاب می شوند و عبارت اند از:

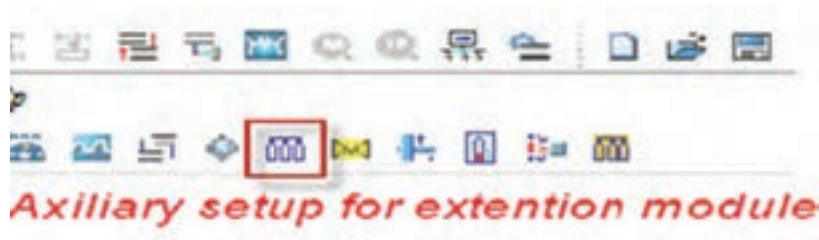
- Dvp04AD کارت آنالوگ با ۴ ورودی از نوع ولتاژ و جریان؛
- Dvp06AD کارت آنالوگ با ۶ ورودی از نوع ولتاژ و جریان؛
- Dvp06XA کارت آنالوگ با ۴ ورودی و ۲ خروجی از نوع ولتاژ و جریان؛
- Dvp04PT کارت آنالوگ با ۴ ورودی از نوع PT100 برای اندازه گیری دما؛
- Dvp04TC کارت آنالوگ با ۴ ورودی از نوع ترموکوپل.

در «CPU» های سری S و H و E جمعاً تا ۸ کارت، قابل توسعه است که از K0 که اولین کارت آنالوگ تا K7 که هشتمین کارت است، می توان به سیستم اضافه نمود. جهت استفاده از کانال های کارت آنالوگ و خواندن مقادیر، می توان به کمک دستور «FROM» مقدار هر کانال از هر کارت را خواند و در یک رجیستر ذخیره نمود یا می توان تنظیمات مربوط به هر کانال آنالوگ را از طریق «WIZARD» نرم افزار انجام داد. مراحل تنظیم «WIZARD» به دو صورت قابل دسترسی است:

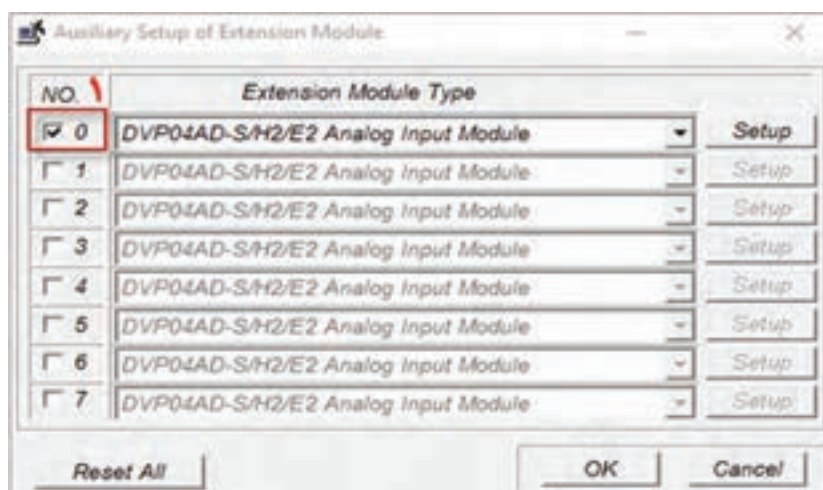
الف) نوار منو - گزینه «Wizard»



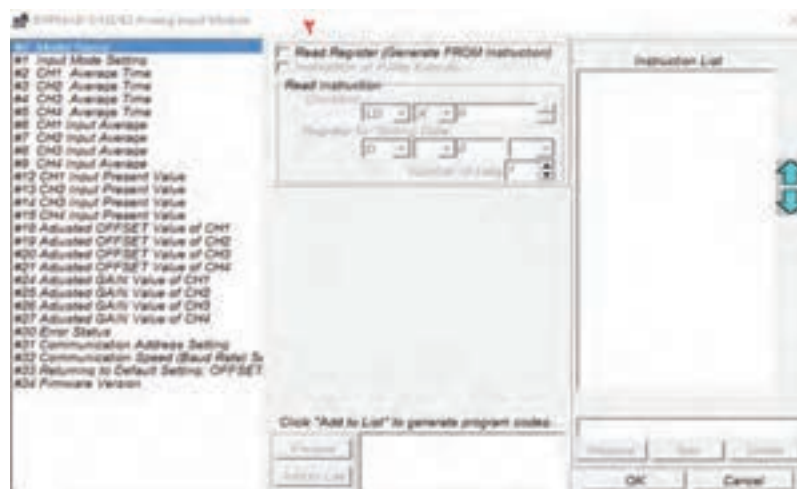
ب) نوار ابزار



مراحل تنظیم به صورت زیر است :  
 ۱- Auxiliary setup of extension module را باز کرده؛ اولین کارت را انتخاب می کنیم.



۲- زبانه سمت راست را کلیک می کنیم تا کارت ها باز شوند و کارت مورد نظر را انتخاب کنیم. مثلاً DVP04AD، سپس گزینه «setup» را می زنیم تا وارد تنظیمات کارت شویم.



۳- CR0 مربوط به مدل کارت است که با آن کاری نداریم. CR1 مربوط به «input mode setting» است و از آن برای تنظیم هر کانال از نظر ولتاژ یا جریان یا هر نوع دیگر استفاده می‌شود. مثلاً اگر خروجی حسگر متصل به کانال اول ولتاژ است باید در این قسمت نوع سیگنال کانال اول را ولتاژ انتخاب کنیم. سپس گزینه «write» را انتخاب و شرط دستور را M1000 یا هر شرطی که برای اجرا نیاز داریم اعمال می‌کنیم و در نهایت کانال اول را از نظر نوع سیگنال انتخاب می‌کنیم. سپس «preview» را می‌زنیم تا دستور در پایین ظاهر شود و در نهایت «add to list» را می‌زنیم تا در خانه سمت راست اضافه شود.

■ اگر همه کانال‌ها را بخواهیم برای هر چهار کانال باید سیگنال را انتخاب و مراحل «preview» و «add to list» را تکرار کنیم تا در سمت راست، برای همه کانال‌ها سیگنال تعیین شده باشد.



لازم به ذکر است برای میانگین لازم است زمان تعیین شود که زمان برای کانال اول CR2 است.

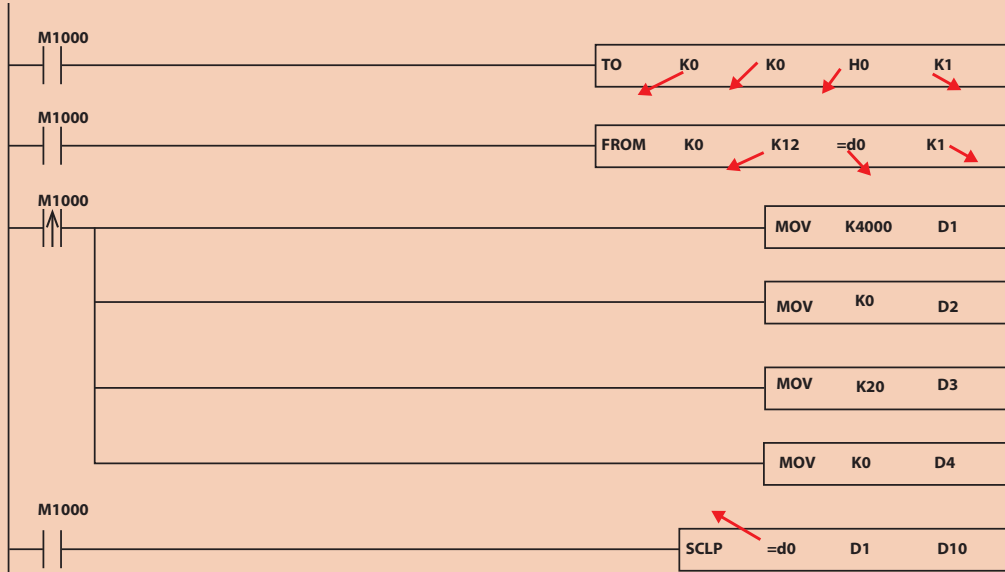


باید دقت شود آدرس حافظه‌های اختصاص یافته تداخل نداشته باشد.

بدانید



برنامه‌نویسی کارت‌های آنالوگ همانند ورودی آنالوگ در «CPU» است فقط در برنامه‌ها به جای «d1056» باید مثلاً d0 که آنالوگ در آن ذخیره شده است جای‌گذاری شود و صرفاً دستور «to» و یک دستور «from» برای هر کانال نسبت به قبل اضافه می‌شود و فرایند مقیاس‌دهی و برنامه کنترل همانند قبل است.



بدانید



در حالت شبیه‌ساز جهت، باید به جای حسگر، که قرار است اطلاعات را از ورودی آنالوگ اندازه‌گیری کند، در نرم‌افزار به صورت دستی مقداردهی شود و فرایند آن به شکل زیر است:



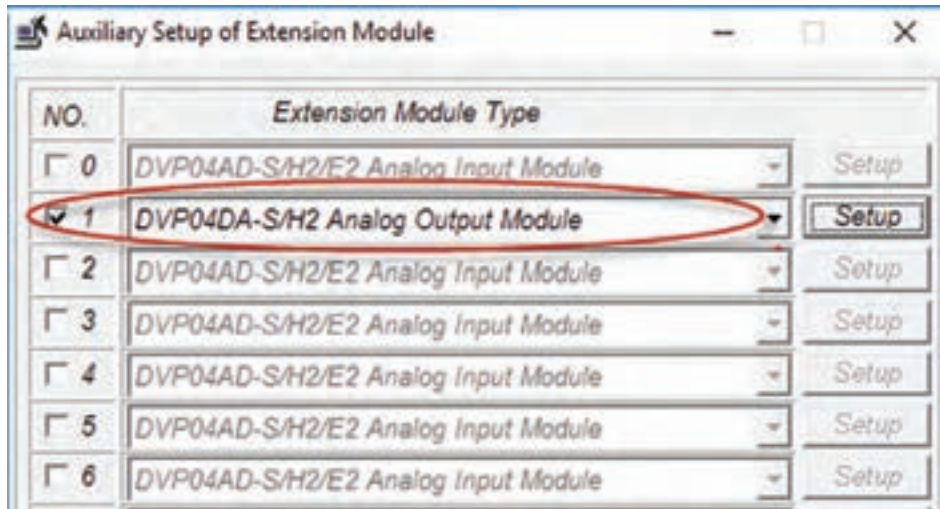


بودمان سوم: نصب و راه اندازی کنترل کننده های منطقی

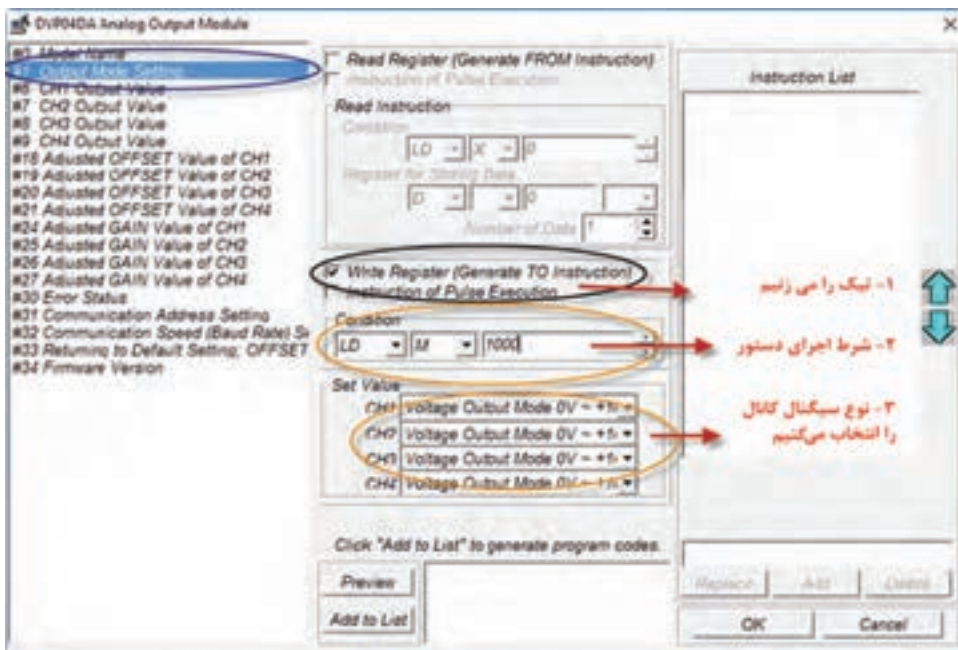
## ۲- کارت های توسعه آنالوگ خروجی

Dvp02DA کارت با دو خروجی آنالوگ از نوع ولتاژ و جریان است. تمامی مراحل تنظیم کارت خروجی آنالوگ همانند کارت ورودی آنالوگ است.

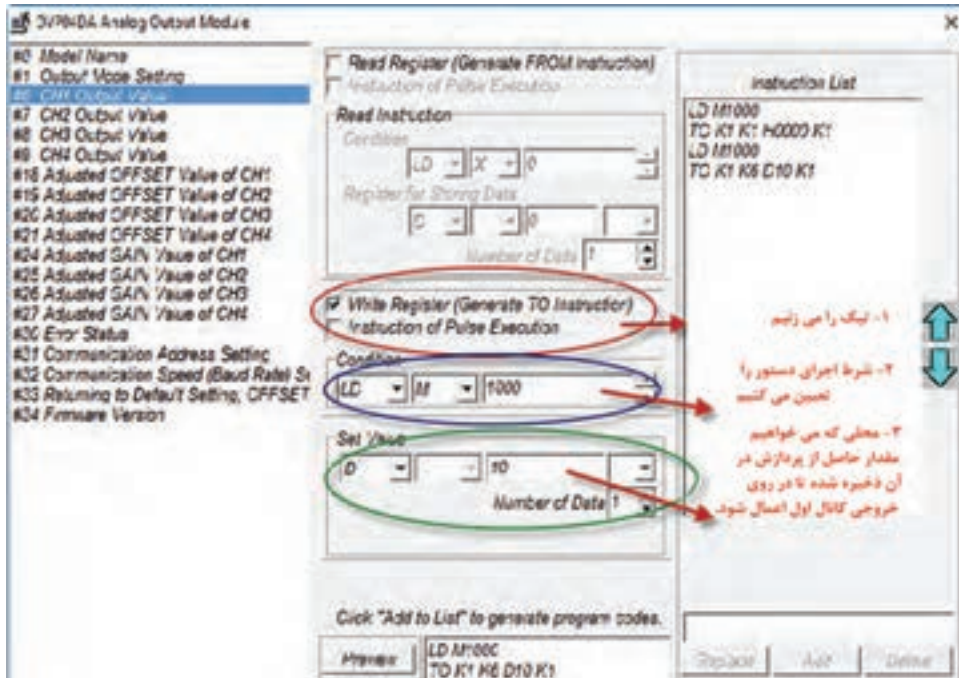
۱- کارت خروجی را انتخاب می کنیم و وارد تنظیمات می شویم.



## ۲- تنظیم سیگنال کانال ها



۳- انتخاب یک رجیستر، برای اینکه مقدار حاصل از پردازش را که باید در کانال خروجی اعمال شود، در آن ذخیره کنیم تا مقدار لحظه به لحظه، از طریق رجیستر به کانال اعمال گردد.



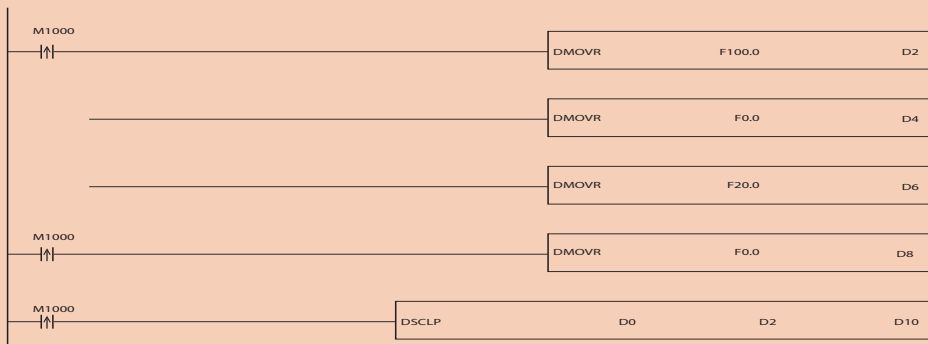
کارت آنالوگ ورودی و خروجی روی «PLC» اضافه کنید و فعالیت صفحه ۱۱۷ و دو فعالیت اول صفحه ۱۱۹ را روی کارت‌های آنالوگ پیاده سازی کنید.

فعالیت



مقیاس‌بندی با دستور «SCLP» از نوع اعداد صحیح است. بنابراین مقادیر خوانده شده به صورت پله‌ای هستند (۱، ۲، ۳ و...) و دقت خوبی ندارد. با اضافه شدن کارت آنالوگ می‌توانیم مقیاس‌دهی در مبنای اعشاری انجام دهیم که باعث می‌شود دقت بالایی داشته باشیم و مقادیر پیوسته خواهند بود. لذا در کارت‌های آنالوگ مقیاس‌بندی اعشاری خواهد بود (همانند برنامه زیر)

بدانید



مبنای d0 باید اعشاری باشد اگر نبود باید در برنامه دستور FLT D0 D12 اجرا شده و به جای d0 در دستور مقیاس d12 قرار بگیرد.

نکته



برنامه ای بنویسید که سیستم کنترل فشار آب مربوط به یک مجتمع ۲۰ واحدی را به صورت پیوسته انجام دهد. فشار توسط حسگر صفر تا ۲۰ بار از طریق کانال صفر بخواند و به صورت زیر عمل کند.

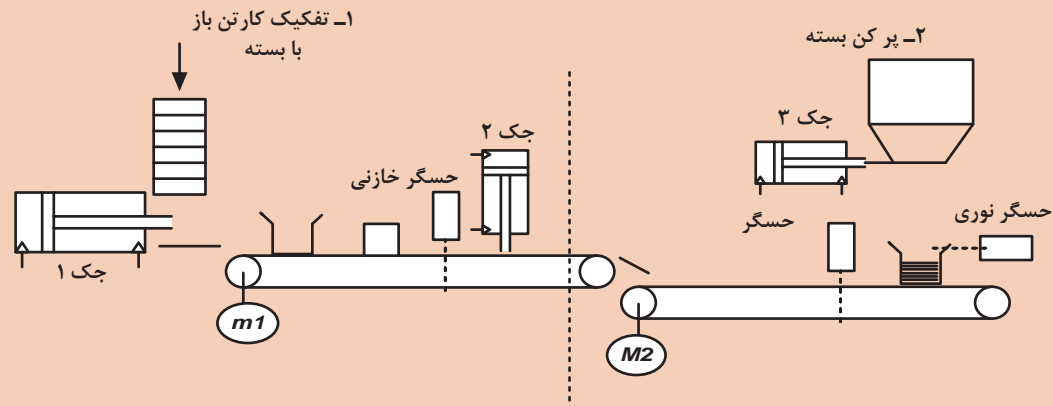
فعالیت



- اگر فشار کمتر از ۲ بار بود موتور با فرکانس ۵۰ هرتز کار کند؛
- اگر فشار بین ۲ تا ۳ بار بود موتور با فرکانس ۴۰/۵ هرتز کار کند؛
- اگر فشار بین ۳ تا ۴ بار بود موتور با فرکانس ۲۵/۵ هرتز کار کند؛
- اگر فشار بین ۴ تا ۵ بار بود موتور با فرکانس ۱۵/۵ هرتز کار کند؛
- اگر فشار بالای ۶ بار بود موتور خاموش شود.

با توجه به مطالب ارائه شده در پنوماتیک از نظر انتخاب لیست شیرها و جک ها، سیستم کنترل خط زیر را برای یک ایستگاه یا دو ایستگاه طراحی کنید و برنامه کنترل آن را بنویسید. لازم به ذکر است موتور ۱ باید قابلیت کنترل دور داشته باشد.

پروژه



### شرح کار:

تعیین تعداد I/O، سیم‌کشی و اجرای کامل پروژه و برنامه‌نویسی کنترل فرایند با رعایت کامل نکات ایمنی و همراه با راه‌اندازی نهایی پروژه



### استاندارد عملکرد:

پس از اتمام واحد یادگیری و کسب شایستگی «PLC»، هنرجویان قادر خواهند بود یک فرایند صنعتی را بررسی و تعداد ورودی و خروجی مورد نیاز را تعیین کنند و برنامه کنترل مربوطه را بنویسند و آن را به‌طور کامل راه‌اندازی کنند.

### شاخص‌ها:

صحت تعیین تعداد I/O - نصب صحیح سیم‌کشی I/O - برنامه‌نویسی صحیح پروژه - عملکرد صحیح فرایند در تست شبیه‌ساز - کنترل صحت سیم‌کشی در حالت آنلاین - کنترل عملکرد صحیح هر قسمت پروژه - تست کامل و راه‌اندازی نهایی پروژه.

### شرایط انجام کار و ابزار و تجهیزات:

#### الف) شرایط

- ۱- اجرا در کارگاه «PLC»
- ۲- نور یکنواخت با شدت ۴۵۰ لوکس
- ۳- تهویه استاندارد و دمای ۲۰°C
- ۴- تجهیزات استاندارد و آماده به کار
- ۵- وسایل ایمنی استاندارد
- ۶- زمان ۱۸۰ دقیقه

#### ب) ابزار و تجهیزات

- ۱- فیوز سیلندری سه فاز ۲- کلید مینیاتوری تک‌فاز ۳- کنترل فاز برای موتور تسمه نقاله ۴- کنترل بار برای موتور تسمه نقاله - شستی استپ و استارت ۵- حسگر نوری و القایی - رله 24VDC ۶- کنتاکتور ۷- موتور ۸- سیم ۹- داکت ۱۰- ریل

### معیار شایستگی:

ردیف	مرحله کار	حداقل نمره قبولی از ۳	نمره هنرجو
۱	تعیین تعداد I/O	۱	
۲	سیم‌کشی I/O	۱	
۳	برنامه‌نویسی	۳	
۴	تست شبیه ساز	۲	
۵	کنترل صحت ورودی/خروجی در حالت آنلاین و تست و کنترل هر قسمت فرایند	۳	
۶	تست نهایی و راه‌اندازی کامل پروژه	۳	
	<b>شایستگی‌های غیر فنی، ایمنی، بهداشت، توجهات زیست محیطی و نگرش:</b> <ol style="list-style-type: none"> <li>۱ رعایت قواعد و اصول در مراحل کار؛</li> <li>۲ استفاده از لباس کار و کفش ایمنی؛</li> <li>۳ تمیز کردن گیره و محیط کار؛</li> <li>۴ رعایت دقت و نظم.</li> </ol>		
	<b>میانگین نمرات</b>		
	* حداقل میانگین نمرات هنرجو برای قبولی و کسب شایستگی، «۲» است.		