

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

:

به این علت که اصول پردازنده‌ها و میکروکنترلرها بر مبنای دیجیتال پایه‌ریزی شده است شما

در کار با میکرو به مباحث دیجیتال برخورد می‌کنید.

دیجیتال تماماً صفرها و یک‌هائی هستند که بوسیله آنها تمامی عملیات‌های ریاضی و

الکترونیکی صورت می‌گیرد و هیچ‌چیز جز صفر و یک برای دیجیتال معنا و مفهومی ندارد.

در شکل زیر چند مترادف دیجیتالی نشان داده شده است:

ریاضیات	الکترونیک	شرط برنامه‌نویسی	صحبت‌های عامیانه
1	+5V	درست	بله
0	0 V	غلط	خیر

:

کوچکترین خانه حافظه را بیت نامگذاری کرده‌اند.

هر چهار بیت را یک نیبل گویند.

هر هشت بیت را یک بایت گویند.

هر دو بایت یک کلمه یا Word است.

:

## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

اگر کسی از شما بپرسد چند سال دارید؟ چه پاسخی خواهید داد؟

۱۸ سال دارم، ۳۶ سال دارم و یا هر پاسخ دیگری. در این پاسخ‌ها از اعدادی استفاده شده

است که از ترکیب ۰، ۱، ۲، ۳، ۴، ۵، ۶، ۷، ۸، ۹ ساخته شده است. به صورت مختصر می‌توانیم بگوئیم ما

برای مشخص کردن تمام اعداد از یکی یا ترکیب چند رقم فوق استفاده می‌کنیم یا به صورت

ساده‌تر از ده رقم برای مشخص کردن تمام اعداد استفاده می‌کنیم.

:

به تعداد رقم‌هایی که برای مشخص کردن اعداد استفاده می‌نمائیم مبنای کاری گوئیم.

**مثال:** مبنای کامپیوتر دو می‌باشد زیرا یک دستگاه دیجیتالی فقط دو رقم صفر و یک را

برای تمام کارهایش استفاده می‌کند.

:

شما وقتی از صفر شروع به شمارش می‌نمائید، بعد از عدد نه دیگر رقمی برای نمایش عدد

بعدی ندارید پس با ترکیب دو رقم یک و صفر عدد جدیدی می‌سازید یعنی بعد از شمارش ده

رقم در مبنای ده به عدد (۱۰) می‌رسید. در مبنای دو شما پس از شمارش دو رقم به عدد (۱۰)

خواهید رسید.

در جدول زیر چهارمبنای اصلی نمایش داده شده است.

## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

مبنای ده Decimal	مبنای هشت Octal	مبنای دو Binary	مبنای شانزده Hexadecimal
0	0	0	0
1	1	1	1
2	2	10	2
3	3	11	3
4	4	100	4
5	5	101	5
6	6	110	6
7	7	111	7
8	10	1000	8
9	11	1001	9
10	12	1010	A
11	13	1011	B
12	14	1100	C
13	15	1101	D
14	16	1110	E
15	17	1111	F
16	18	10000	10

**نکته:** بزرگترین رقم مبنای شانزده برابر بزرگترین عدد چهار رقمی مبنای دو است و بزرگترین رقم مبنای هشت برابر بزرگترین عدد سه رقمی مبنای دو می‌باشد و از این نکته این‌گونه می‌توان استفاده کرد که برای تبدیل اعداد مبنای شانزده می‌توان هر رقم آن‌را با چهار رقم مبنای دو متناظر آن رقم جایگزین کرد و هر رقم مبنای هشت را می‌توان با سه رقم مبنای دو جایگزین کرد.

**مثال:** عدد 5D1A را به مبنای دو تبدیل کنید.

## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

5				D				1				A			
0	1	0	1	1	1	0	1	0	0	0	1	1	0	1	1

به جدول فوق دقت کنید رقمی مانند **1H** برابر یک رقم مبنای دو است در نتیجه سه رقم

سمت چپ را با پر صفر می‌نمائیم.

:

پورت در اصطلاح به معنای درگاه است. یعنی جایگاهی که به‌توان از آن به‌عنوان محل ورود

و خروج داده استفاده کرد. میکروکنترلر خانواده **8x51** دارای ۴ پورت است که به ترتیب

**(P0),(P1),(P2),(P3)** نامگذاری شده است. هر پورت دارای هشت پین است که با همان نام،

یک بایت از حافظه **RAM** را به خود اختصاص داده است. در نتیجه هر بیت متناظر یک پین است

و با یک کردن هر بیت پایه متناظر آن به سطح منطقی (+۵) می‌رود و با صفر کردن هر بیت پایه

نظیر آن **GND** می‌شود. در واقع ۴ پورت میکروکنترلر ۳۲ پین از ۴۰ پین میکرو را اشغال کرده

است.

### : 89C51

:

این پایه‌ها مربوط به پورت یک می‌باشند که به صورت زیر نمایش داده می‌شود:

## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

پورت P1							
P1.7	P1.6	P1.5	P1.4	P1.3	P1.2	P1.1	P1.0
8	7	6	5	4	3	2	1

:

این پایه‌ها مربوط به پورت سه میکروکنترلر است و متناظر با بایت P3 است.

پورت P3							
P3.7	P3.6	P3.5	P3.4	P3.3	P3.2	P3.1	P3.0
17	16	15	14	13	12	11	10

پورت سه علاوه بر اینکه به عنوان درگاه ورودی و خروجی استفاده می‌شود برای کارهای

خاصی نیز استفاده می‌شود.

**نکته:** چگونگی استفاده و نحوه برنامه‌نویسی هر کدام از این کاربردها در درس‌های آینده

گفته خواهد شد. این کاربردها و پایه‌های متناظر آن‌ها در جدول زیر آمده است.

P3.0	P3.1	P3.2	P3.3	P3.4	P3.5	P3.6	P3.7
RXD	TXD	INT0	INT1	T0	T1	WR	RD

:

این پایه‌ها مربوط به پورت ۲ میکروکنترلر و متناظر با بایت P2 است.

پورت P2							
---------	--	--	--	--	--	--	--

## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

P2.7	P2.6	P2.5	P2.4	P2.3	P2.2	P2.1	P2.0
28	27	26	25	24	23	22	21

:

این پایه‌ها مربوط به پورت صفر و متناظر با بایت P0 است این پورت در هنگام استفاده از

ROM و RAM خارجی به صورت مالتی پلکس شده تبدیل می‌شود. و در حالت عادی، مشابه

پورت یک عمل می‌کند. در حالت استفاده از ROM و RAM خارجی شما می‌توانید با استفاده از

پایه‌ی ALE و اتصال آن به یک LATCH آن را جدا سازید.

\* ما در برد آموزشی به دلیل نداشتن RAM خارجی نیازی به دیمالتی پلکس نمودن نداشتیم،

البته باید توجه داشت این پورت می‌بایست با مقاومت‌های ۱۰ کیلو اهم به VCC متصل شوند)

(PULL UP شوند)

ما در برد آموزشی از مقاومت‌های بالاکش ARRAY استفاده می‌نماییم که در شکل زیر

توضیح داده شده است.

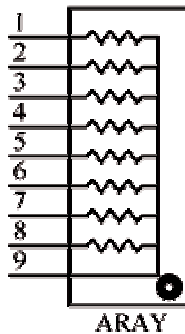
پورت P0							
P3.7	P3.6	P3.5	P3.4	P3.3	P3.2	P3.1	P3.0
32	33	34	35	36	37	38	39

## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

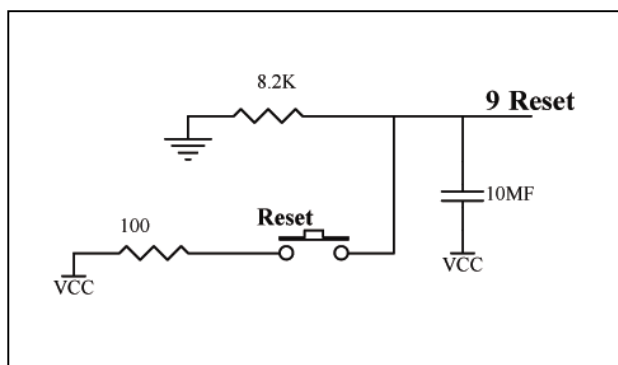
آموزش دوره‌های تخصصی الکترونیک



تا کنون ۳۲ پایه از میکرو را که مربوط به پورت‌های میکرو می‌شد را مورد بررسی قرار دادیم. حال به پایه‌های دیگر می‌پردازیم.

:

این پایه برای Reset کردن میکروکنترلر منظور شده است از آنجایی که هیچ منبع تغذیه‌ای نمی‌تواند به صورت ایده‌آل کار کند و بعد از روشن نمودن زمانی طول کشیده تا تغذیه رگوله شود و تا زمانی که تغذیه ثابت نشود تضمینی در صحت انجام کار میکروکنترلر نیست. پس می‌توانیم با



طراحی مدار زیر مدتی میکروکنترلر را در حالت reset نگه‌داریم و بعد از ثابت شدن تغذیه از حالت Reset خارج می‌شود، مدت Reset شدن

میکروکنترلر توسط R1 و C1 مشخص شده است.

$$C1 = 10\mu F$$

$$R1 = 8.2 K\Omega$$

## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

$$R2 = 100 \Omega$$

مدار بالا می‌تواند به صورت دستی نیز میکروکنترلر را **Reset** نماید.

:

سرعت CPU در انجام کار یکی از مهمترین نکته‌های یک پروژه است حال این سرعت

چگونه و از کجا محاسبه می‌شود؟

پایه‌های ۱۸ و ۱۹ که در شماتیک پایه‌ها به نام‌های XTAL1 , XTAL2 نمایش داده شده‌اند

پایه‌های CPU-Clock می‌باشند. سرعت CPU همواره  $\frac{1}{12}$  سرعت نوسان ساز متصل به پایه‌های

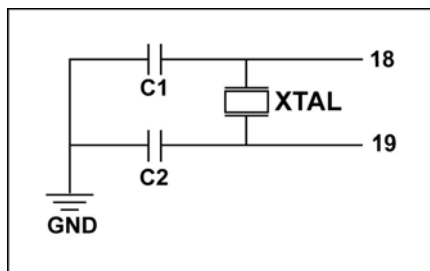
۱۸ و ۱۹ است که به صورت زیر متصل می‌شوند.

**نکته:** حداکثر فرکانس نوسان‌ساز که می‌تواند به میکروکنترلر متصل شود کریستال 24MHz

می‌باشد که در این صورت سرعت CPU برابر 2MHz است. این بدین معناست که CPU قادر

خواهد بود در ۱ ثانیه دو میلیون واحد کاری را انجام دهد. از این پس به هر واحد کاری یک

**سیکل ماشین گفته می‌شود.** لازم به ذکر است زمان هر سیکل از فرمول زیر محاسبه می‌شود.



$$\text{سرعت هر سیکل ماشین} = \frac{1s}{XTAL / 12}$$

$$C1 = C2 = 30 pF \pm \% 10$$



## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

با قرار دادن متغیرها در این فرمول درمی‌یابیم که با یک کریستال ۲۴ مگاهرتز سرعت هر سیکل ماشین برابر با ۵۰۰ نانوثانیه می‌باشد.

**نکته:** از این به بعد کریستال را به صورت قراردادی و بر اساس کریستال برد آموزشی ۱۲ مگاهرتز در نظر می‌گیریم.

توجه داشته باشید که هر دستوری در یک سیکل ماشین اجرا نمی‌شود و بسته به نوع دستور، تعداد سیکل مشخص می‌شود.

:

این پایه جهت اتصال به OE از EEPROM خارجی می‌باشد و در حالت عادی استفاده نمی‌شود.

:

**ALE:** این پایه جهت دیمالتی پلکس نمودن پورت صفر در هنگام استفاده از ROM و RAM خارجی به کار می‌رود.

### پایه ۳۱:

برای دسترسی به حافظه برنامه خارجی است ما در 89C51 معمولاً به علت وجود EEPROM داخلی نیازی به حافظه خارجی نداریم پس این پایه در حالت عادی به VCC متصل است.

## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

:

دو پایه مهم میکروکنترلر است که جهت اتصال به VCC, GND می‌توان طبق پیوست از

آنها استفاده کرد.

:

در جدول زیر این دو میکروکنترلر از نظر حافظه با یکدیگر مقایسه شده‌اند.

تعداد تایمر	ROM خارجی	ROM داخلی	حافظه ROM	میکروکنترلر
2	64K بایت	4K بایت	128 بایت	8051
3	128K بایت	8K بایت	256 بایت	8052

**نکته:** در قسمت ROM خارجی منظور قابلیت پشتیبانی از این مقدار حافظه خارجی است.

## : RAM

RAM میکروکنترلر به قسمت‌های مختلفی تقسیم بندی می‌شود.

الف) بانک‌های ثابت

ب) RAM بیت آدرس پذیر

ج) حافظه RAM همه منظوره

د) ثابت‌های ویژه

آدرس

آدرس بیت

آدرس

آدرس بیت

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

بایت																
7F	حافظه RAM همه منظوره															
30																
2F									7F	7E	7D	7C	7B	7A	79	78
2E									77	76	75	74	73	72	71	70
2D									6F	6E	6D	6C	6B	6A	69	68
2C									67	66	65	64	63	62	61	60
2B									5F	5E	5D	5C	5B	5A	59	58
2A									57	56	55	54	53	52	51	50
29									4F	4E	4D	4C	4B	4A	49	48
28									47	46	45	44	43	42	41	40
27	3F	3E	3D	3C	3B	3A	39	38								
26	37	36	35	34	33	32	31	30								
25	2F	2E	2D	2C	2B	2A	29	28								
24	27	26	25	24	23	22	21	20								
23	1F	1E	1D	1C	1B	1A	19	18								
22	17	16	15	14	13	12	11	10								
21	0F	0E	0D	0C	0B	0A	09	08								
20	07	06	05	04	03	02	01	00								
1F	بانک ۳															
18																
17	بانک ۲															
10																

بایت									
FF									
F0	F7	F6	F5	F4	F3	F2	F1	F0	B
E0	E7	E6	E5	E4	E3	E2	E1	E0	ACC
D0	D7	D6	D5	D4	D3	D2	D1	D0	PSW
B8	----	----	----	BC	BB	BA	B9	B8	IP
B0	B7	B6	B5	B4	B3	B2	B1	B0	P3
A8	AF	----	----	AC	AB	AA	A9	A8	IE
A0	A7	A6	A5	A4	A3	A2	A1	A0	P2
98	بیت آدرس پذیر نیست								SBUF
90	9F	9E	9D	9C	9B	9A	99	98	SCON
8D	بیت آدرس پذیر نیست								TH1
8C	بیت آدرس پذیر نیست								TH0
8B	بیت آدرس پذیر نیست								TL1
8A	بیت آدرس پذیر نیست								TL0
89	بیت آدرس پذیر نیست								TMOD
88	8F	8E	8D	8C	8B	8A	89	88	TCON
87	بیت آدرس پذیر نیست								PCON

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

0F	بانک ۱	83	بیت آدرس پذیر نیست								DPH
08		82	بیت آدرس پذیر نیست								DPL
07		81	بیت آدرس پذیر نیست								SP
00		R0-R7 بانک قراردادی ثابت برای	80	87	86	85	84	83	82	81	80

ثباتهای ویژه

## خلاصه حافظه درون تراشه ۸۰۵۱

توجه داشته باشید خانه های حافظه از نظر نوع دسترسی به ۲ دسته دسترسی بیتی و دسترسی بایتی تقسیم می‌شوند. تمام بایت‌های حافظه به صورت بیتی قابل دسترسی نیستند به عنوان مثال بایتی به نام P1 وجود دارد که می‌توانیم بگوییم بیت صفرم بایت P1(P1.0). اما نمی‌توانیم بگوییم بیت صفرم بایت TMOD زیرا TMOD یک بایت بیت آدرس پذیر نیست شما می‌توانید در جدول خانه‌های حافظه لیست کاملی از بایت‌های بیت آدرس پذیر را مشاهده نمایید. البته به این نکته توجه داشته باشید که از بایت‌های بیت آدرس‌پذیر می‌توان برای کارهای بایتی معمولی نیز استفاده کرد.

∴

از آدرس 00H تا 1F H حافظه به این بانک‌ها اختصاص داده شده است که در مجموع ۴ بانک به اسامی (بانک صفر، بانک یک، بانک دو، بانک سه)، وجود دارد. هر کدام از این بانک‌ها از هشت بایت تشکیل شده است که به اسامی R0 تا R7 نامگذاری شده‌اند. و ما می‌توانیم به دلخواه

## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

هر کدام از این بانک‌ها را انتخاب کنیم. امکان دسترسی به این بانک‌ها به صورت بایتی است و بیت آدرس پذیر نیست.

شاید از خود سوال کنید که دسترسی بایتی به چه معناست. دسترسی بایتی یعنی اینکه برای انتقال اطلاعات به این ثبات‌ها باید از دستورات بایتی استفاده کرد. و از دستورات بیتی برای تحت تاثیر قرار دادن تنها یک بیت می‌توان استفاده کرد. به بیان ساده‌تر نمی‌توانیم مستقیماً یک بیت از یک بایت را تغییر دهیم و برای تغییر یک بیت باید کل یک بایت را تحت تاثیر قرار دهیم. دستوراتی که از این ثبات‌ها استفاده می‌کنند از دستورالعمل‌هایی هستند که با بقیه روش‌های آدرس‌دهی کار می‌کنند، سریعتر و کوتاه‌تر هستند پس بهتر است برای داده‌هایی که بیشتر به کار برده می‌شوند از این ثبات‌ها استفاده شود.

### : RAM

از آدرس 20H تا 2FH ثبات‌هایی قرار دارند که دسترسی به آن‌ها به صورت بیتی انجام می‌شود یعنی اینکه در این ثبات‌ها علاوه بر اینکه می‌توانیم به این آدرس‌ها به صورت بایتی دیتا انتقال دهیم به صورت بیتی نیز این امر امکان‌پذیر است و ما می‌توانیم یک بیت از یک بایت را مثلاً صفر یا یک کنیم.

### : RAM

## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

از آدرس 30H تا 2FH ثبات‌هایی وجود دارد که دسترسی به آن‌ها به صورت بایتی است. و تنها فرقی با بیت آدرس‌پذیر در نحوه دسترسی به ثبات است. که در بیت آدرس‌پذیر به صورت بیتی و در ثبات‌های همه منظوره به صورت بایتی است.

:

از آدرس 80H تا FFH مخصوص ثبات‌های ویژه‌ای است که هر کدام از این‌ها برای کار خاصی در نظر گرفته شده‌اند که ارتباط CPU با بقیه قسمت‌های میکروکنترلر از طریق این ثبات‌ها انجام می‌پذیرد، که در درس‌های آینده به تمامی آن‌ها خواهیم پرداخت.

:

در کل، دستوراتی که در برنامه‌نویسی میکروکنترلر استفاده می‌شوند به دو دسته تقسیم می‌شوند:

۱- دستورات مربوط به CPU در میکروکنترلرهای ۸۰۵۱

۲- دستورات مربوط به Assembler

تفاوت این دو نوع دستور این است که دستورهای Assembler در هنگام کامپایل شدن اجرا می‌شوند و برای رساندن مفهومی به اسمبلر است ولی دستورهای نوع اول در داخل میکرو اجرا می‌شوند.

:

## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

اگر تاکنون با زبان‌های برنامه‌نویسی مختلف کار کرده باشید مشاهده کرده‌اید که تمامی آن‌ها محیط مخصوص به خود را دارا می‌باشند ولیکن خصوصیت زبان Assembly این است که تنها نیاز به محیطی دارد که اصطلاحاً قابلیت (Text Editor) را داشته باشد.

اگر محیط کار شما ویندوز باشد می‌توانید با اجرا نمودن نرم‌افزار NOTE PAD و چنانچه در سیستم عامل Dos کار می‌کنید می‌توانید با اجرای نرم‌افزار Edit.exe محیط برنامه‌نویسی خود را فعال نمایید.

هم‌اکنون شما می‌توانید دستورات برنامه‌نویسی را در این محیط تایپ نمایید.

### ORG

دستور فوق که مربوط به Assembler می‌باشد به شما امکان می‌دهد، آدرس شروع برنامه در حافظه برنامه EEPROM میکروکنترلر را تعیین نمایید.

مثال:

ORG 0H

این دستور شروع برنامه در حافظه را از خانه صفر آن تعیین می‌کند.

### MOV

این دستور برای کپی کردن یک خانه از حافظه یا داده‌ها در بایت‌های حافظه استفاده می‌شود. توجه کنید که این دستور یک دستور بایستی است.

مثال:

MOV P1,#55H

## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

این دستور مقدار عددی ۵۵ در مبنای شانزده را در بایتی از حافظه به نام P1 می‌ریزد.

```
MOV P1,#85D
```

این دستور مقدار عددی ۸۵ در مبنای ده را در بایتی از حافظه به نام P1 می‌ریزد.

```
MOV P1,#01010011B
```

این دستور مقدار عددی ۰۱۰۱۰۰۱۱ در مبنای دو را در بایتی از حافظه به نام P1 می‌ریزد.

**توجه:** همان‌طور که در دستورات بالا مشاهده نمودید علامت # قبل از هر کدام از اعداد قید

شده بود. این علامت برای مشخص نمودن این مفهوم است که مقدار عددی عبارت بعدی

می‌بایست در بایت معرفی شده از حافظه ریخته شود.

مثال ۲:

```
MOV P1,55H
```

این دستور محتویات داخل بایت 55H حافظه را در خانه P1 حافظه می‌ریزد. دقت کنید که تمام

ثبات‌های موجود را می‌توان به‌جای این بایت استفاده کرد.

مثال ۳:

```
MOV A,P1
```

```
MOV P2,A
```

در مثال بالا اگر بخواهیم اطلاعات روی P1 را روی P2 بریزیم ابتدا باید دیتا را به

آکومولاتور انتقال دهیم و سپس از آکومولاتور به P2 انتقال دهیم.

عبارات B,D,H که بعد از مقدارهای عددی قید شده آمده است مبین مبنای عدد می‌باشد.

B = Binary	D = Decimal	H = Hexadecimal
------------	-------------	-----------------



# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

## EQU

این دستور مربوط به اسمبلر بوده و جهت نامگذاری قسمتی از حافظه داخلی میکروکنترلر

استفاده می‌شود.

**مثال ۴:** برنامه زیر خانه 20H از حافظه را به MEMORY نامگذاری می‌کند.

```
MEMORY EQU 20H
```

:

## JMP

این دستور برای پرش به مکانی از برنامه استفاده می‌شود که این مکان یک برچسب است که قبلاً

نامگذاری شده است دقت کنید که برای پرش‌های کوتاه می‌توان از دستور LJMP و برای پرش‌های

بلند از دستور SJMP استفاده کرد.

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

مثال ۵:

**LOOP:**

.  
. .  
.

**JMP LOOP**

دستور فوق به مکانی از برنامه به نام LOOP پرش می‌کند.

**LOOP:JMP LOOP**

این خط برنامه باعث می‌شود CPU در یک حلقه گیر کند. البته می‌توان به جای برنامه فوق از دستور JMP \$ نیز استفاده کرد.

**END**

این دستور مربوط به ASSEMBLER است که نشان دهنده پایان برنامه و خروج از آن است. لازم به ذکر است این دستور می‌بایست همیشه آخرین خط برنامه باشد.

*نکته:* شما باید به گونه‌ای برنامه‌نویسی نمایید که هیچگاه برنامه به دستور END نرسد.

**( ) :**

با ترکیب دستورات فوق شما می‌توانید یک برنامه ساده بنویسید:

ORG 0

MOV P1,#55H

LOOP: JMP LOOP

## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

END

با پروگرام نمودن IC و قرارداد آن بر روی برد آموزشی این برنامه باعث می‌شود که LED های

روی برد به صورت یک درمیان روشن شوند. چرا؟

:

اولین دستور که نمایان‌گر خانه شروع برنامه است. همان‌طور که در مباحث میکروکنترلر

89C51 مطالعه نموده‌اید، P1 دقیقاً یک بیت از حافظه RAM میکروکنترلر می‌باشد که به ازای هر

بیت خود یک پایه I/O بر روی میکروکنترلر دارد (پایه ۱ تا ۸). حال اگر هر بیت از این حافظه یک

شود شما روی پایه نظیر آن +5V و چنانچه صفر شود شما روی پایه نظیر هر بیت مقدار صفر یا

GND خواهید داشت و بالعکس.

حال عدد 55H را به باینری تبدیل نمایید و در بیت‌های نظیر حافظه قرارداد دهید.

55H = 010101010 B

دقیقاً به ازای عدد باینری بالا پایه‌های میکرو فعال می‌شوند.

**نکته:** البته توجه نمایید LED های روی برد آموزشی به صورت آند مشترک بسته شده‌اند در

نتیجه Active Low هستند پس به ازای هر صفر LED روشن می‌شود.

:

DJNZ COUNT,LOOP

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

این دستور مقدار عددی خانه‌ای از حافظه که قبلاً توسط دستور EQU به نام COUNT

تعریف شده است را یک واحد کاهش می‌دهد و اگر مقدار آن خانه صفر نبود به آدرسی از خانه

حافظه به نام LOOP پرش می‌کند.

## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

در مثال زیر با ایجاد یک تاخیر زمانی دو مقدار مختلف را بر روی پورت یک با فاصله

زمانی مشخص را می‌توانیم روی اسکوپ ببینیم.

:( )

```
COUNT EQU 30H
```

```
ORG 00H
```

```
MAIN:
```

```
MOV COUNT,#200
```

```
MOV P1,#01010101B
```

```
LOOP: DJNZ COUNT,LOOP
```

```
MOV P1,#10101010B
```

```
MOV COUNT,#250
```

```
LOOP1:DJNZ COUNT,LOOP
```

```
JMP MAIN
```

برنامه فوق یک بار پورت یک را با 55H فعال کرده و بعد از 400 $\mu$ s پورت یک را با AAH

بارگذاری می‌کند و بعد از 501 $\mu$ s دوباره پورت یک را با 55H بارگذاری می‌کند و همین کار ادامه

خواهد داشت. شاید از خود بپرسید چگونه متوجه می‌شویم مقدار پورت‌ها، بعد از گذشت چه

زمانی عوض می‌شوند؟

خیلی ساده است انجام هر دستور DJNZ دو سیکل ماشین زمان نیاز دارد و در حلقه اول ما

۲۰۰ بار این دستور را اجرا نمودیم پس در مجموع 400 $\mu$ s و در حلقه بعدی چون قبل از آن یک

## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

دستور MOV وجود دارد و می‌دانیم برای انجام این دستور نیز یک سیکل ماشین زمان نیاز داریم در مجموع  $2 \times 250 + 1$  سیکل ماشین تأخیر ایجاد شده است.

**نکته:** طبق استاندارد که در ابتدای این فصل برای سخت‌افزار میکرو قرار دادیم از نوسان‌ساز 12MHz استفاده می‌کنیم و در نتیجه هر سیکل ماشین یک میکروثانیه زمان نیاز دارد. شما می‌توانید برای تست بصری این تغییرات از دستگاه اسیلوسکوپ استفاده نمایید. زیرا به علت سرعت زیاد LEDها تغییرات آن برای ما ملموس نیست.

برای تنظیم زمان بیشتر می‌توانید از چند حلقه متداخل استفاده نمایید که با یک مثال در زیر نشان داده شده است.

$$X = \text{LOOP1} = 2 * 255$$

$$Y = \text{LOOP2} = 255 * X$$

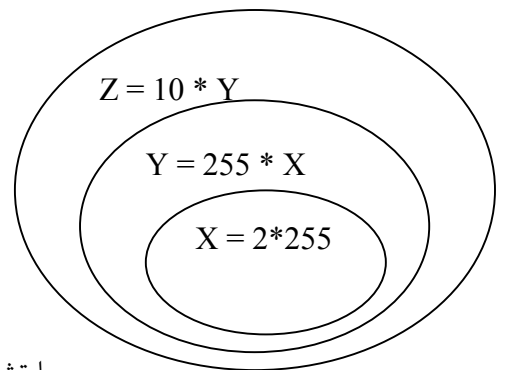
$$Z = \text{LOOP3} = 10 * Y$$

در نتیجه

$$Z = 10 * (255 * (2 * 255)) = 1300500$$

پس با تشکیل حلقه‌های روبرو تقریباً  $1/3$  ثانیه تأخیر

ایجاد نمودیم.



البته توجه نمایید ما از مقدار تأخیر مربوط به خود DJNZ در حلقه دوم و سوم صرف نظر

کردیم زیرا در مقابل عدد به دست آمده مقدار ناچیزی است. و ما از آن چشم‌پوشی می‌کنیم.

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

( ) :

برنامه‌ای بنویسید که LEDهای متصل به پورت یک، در برد آموزشی را با سرعت حدودی

یک تا دو ثانیه خاموش و روشن کند.

```
COUNT1 EQU 20H
```

```
COUNT2 EQU 21H
```

```
COUNT3 EQU 22H
```

```
MAIN:
```

```
MOV P1,#0FFH
```

این دستور باعث می‌شود تمام LED های روی متصل به پورت P1 خاموش شوند (چون این LED

ها به صورت آند مشترک بسته شده‌اند).

```
CALL DELAY
```

حدود یک ثانیه تأخیر

```
MOV P1,#0H
```

روشن کردن LEDها

```
CALL DELAY
```

یک ثانیه تأخیر

```
JMP MAIN
```

پرش به ابتدای MAIN

```
DELAY:
```

```
MOV COUNT2,#255
```

```
MOV COUNT3,#10
```

```
LOOP3:
```

```
MOV COUNT2,#255
```

## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

```
LOOP2:  
MOV COUNT1,#255  
LOOP1:DJNZ COUNT1,LOOP1  
DJNZ COUNT2,LOOP2  
DJNZ COUNT3,LOOP3  
RET  
END
```

### :CALL

در برنامه فوق چون زیر روال DELAY تکرار شده بود ما از تکرار آن خودداری کردیم و از دستور CALL استفاده کردیم در هنگام انجام این دستور برای زیر برنامه DELAY ، CPU به ابتدای برنامه DELAY رفته و آن را اجرا می‌کند تا به دستور RET برسد. پس از انجام کامل این مرحله و رسیدن به دستور RET از زیر برنامه، DELAY برگشته و کار خود را بعد از دستور CALL ادامه می‌دهد. شما با روش‌های مشابه می‌توانید به صورت تقریبی تأخیر مورد نظر خود را در کنترلرها اعمال نمایید، ولی برای ایجاد تأخیرهای دقیق‌تر در اجرای برنامه می‌توانید از TIMER ها (که در ادامه به بحث در مورد آنها می‌پردازیم) استفاده کنید.



## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

:

همان‌طور که ملاحظه نمودید 8051 دستوراتی جهت کار روی بایت‌های حافظه دارد. توجه

داشته باشید شما توسط دستورات بایتی نمی‌توانید روی یک خانه از بیت حافظه کار کنید جهت

کار با بیت‌ها می‌توانید از دستورات بیتی زیر استفاده نمایید.

### SETB

یک بیت را یک کرده

SETB P1.0

بیت صفر از بایت P1 را یک کرده و تأثیری روی بقیه بیت‌ها ندارد.

P1							
X	X	X	X	X	X	X	1
D7	D6	D5	D4	D3	D2	D1	D0

### CLR

یک بیت را صفر کرده

CLR P0.5

بیت پنجم از بایت P0 را صفر کرده و تأثیری روی بقیه بیت‌ها ندارد.

P0							
X	X	0	X	X	X	X	X
D7	D6	D5	D4	D3	D2	D1	D0

### JNB

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

پریش در صورت یک نبودن

**JB**

پریش در صورت یک بودن

JNB P1.2,LOOP

این دستور بیت دوم بایت P1 را بررسی می‌کند، اگر این پایه از بیرون GND شده بود به

LOOP پریش کرده در غیر این صورت برنامه از خط بعدی ادامه خواهد یافت. بیشترین کاربرد این

دو دستور برای چک کردن کلید است که در ادامه یک مثال برای روشن شدن موضوع آورده شده

است.

:( )

برنامه‌ای بنویسید که با زدن کلید P3.3 عدد 55H روی P1 قرار بگیرد و با زدن کلید P3.5

عدد AAH روی P1 قرار بگیرد.

ORG 00H

SCAN:

JNB P3.3,NUM1

JNB P3.5,NUM2

JMP SCAN

NUM1:

MOV P1,#55H

JMP SCAN

NUM2:

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

MOV P1,#0AAH

JMP SCAN

END

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

## TIMER

در میکروکنترلر 89C51 دو TIMER به صورت سخت‌افزاری وجود دارد که ما توسط آن‌ها

می‌توانیم وقایع مختلف را شمارش کنیم مانند شمارش سیکل‌های ماشین که در این صورت

می‌توانیم زمان را محاسبه نماییم.

اصول کار تایمر شمارش رو به بالا در یک سیکل ماشین است. برای شمارش، دوبایت به نام‌های

TH<sub>X</sub> و TL<sub>X</sub> وجود دارد که در مجموع به صورت ۱۶ بیت کنار هم قرار می‌گیرند.

TH								TL							
8	7	6	5	4	3	2	1	8	7	6	5	4	3	2	1

در 89C51 دو تایمر وجود دارد که حالت‌های مختلفی را می‌توانند داشته باشند که در بایت

TMOD انتخاب می‌شوند.

TIMER 1				TIMER 0			
GATE	CT	M1	M0	GATE	CT	M1	M0

M0 و M1 در Tmod، حالت کار تایمر را مطابق جدول زیر مشخص می‌کنند.

M1	M0	نحوه کار شمارش	حالت
0	0	۱۳ بیتی	0
0	1	۱۶ بیتی	1

## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

1	0	۸ بیتی با بارگزاری خودکار	2
1	1	در این حالت تایمر به دو تایمر ۸ بیتی مجزا تبدیل می‌شود	3

در حالت صفر:

سه بیت بالای  $(TH_X, TL_X)$  مورد استفاده قرار نمی‌گیرند و حداکثر تا عدد  $2^{13} = 8192$

می‌تواند شمارش کند.

در حالت یک:

تمام (۱۶ بیت  $TH_X, TL_X$ ) مورد استفاده قرار گرفته و TIMER حداکثر تا عدد  $65535 =$

$2^{16}$  می‌تواند شمارش نماید.

در حالت دو:

فقط ۸ بیت پایین شمارش شده و ۸ بیت بالا به عنوان BACKUP مقدار ۸ بیت پایین مورد

استفاده قرار می‌گیرد. در این حالت تایمر حداکثر تا عدد  $2^8 = 255$  می‌تواند شمارش نماید.

در حالت سه:

تایمر به صورت دو تایمر ۸ بیتی مجزا کار می‌کند.

در حالت‌های فوق چنانچه شمارش ادامه داشته باشد تا به مقدار حداکثر در هر حالت برسد

بعد از یک شمارش دیگر سرریز رخ داده یعنی دوباره تمام بیت‌ها صفر شده و پرچمی به نام  $TF_X$

فعال می‌شود و کار ادامه می‌یابد.

## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

**نکته:** جهت شروع کار تایمر ابتدا باید بیت  $TR_x$  را یک کرد و با صفر کردن آن می‌توان از شمارش تایمر جلوگیری کرد. ما با بارگذاری عددی مناسب می‌توانیم زمان سرریز را کنترل کرده و تأخیر زمانی مورد نظر را ایجاد نماییم.

### :CT

جهت تعیین حالت کار به صورت تایمر یا کنتور است اگر این بیت صفر باشد پالسهای شمارش از نوسان ساز گرفته میشود بصورتی که به ازای هر سیکل یک شمارش خواهیم داشت در حالت می‌گوئیم درحالت تایمر قرار گرفته است زیرا به ازای یک میکرو ثانیه یک شمارش روبه بالا خواهیم داشت (درصورتی که کریستال 12MHZ استفاده شده باشد) که در آینده در مورد آن بحث خواهیم کرد و اگر این بیت در وضعیت یک قرار گیرد دیگر پالسهای شمارش را از کریستال دریافت نمی‌کند بلکه یکی از پایه های پورت سوم ورودی این پالسها از خارج قرار می‌گیرد که به این حالت شمارنده می‌گوئیم و معمولاً برای شمارش یک رویداد خارج از میکرو استفاده می‌شود.

نکته: در حالت شمارنده پالسها از پایه های  $T0$  و  $T1$  دریافت میشوند.

### :GATE

این بیت برای تایمر مشخص می‌کند که در چه زمانی شروع به کار نماید. اگر صفر باشد در صورت یک شدن  $TR$  شروع به کار می‌کند و اگر یک باشد در صورت یک شدن  $TR$  و نیز یک شدن پایه  $INT$  شروع به کار می‌کند و اگر یکی از آنها ( $TR$  و یا  $INT$ ) صفر شود تایمر  $STOP$  خواهد کرد.

## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

:

۱- بارگذاری ثبات TMODE یعنی اینکه کدام تایمر را انتخاب می‌کنیم و در چه حالتی

MOV TMOD,#10H (مثال)

۲- بارگذاری تاخیر مورد نظر در  $TH_X, TL_X$

Mov TH1,#6CH (مثال)

Mov TL1,#0C0H

۳- روشن کردن تایمر با دستور  $SETB TR_X$

SETB TR1 (مثال)

۴- چک کردن بیت  $TF_X$  که چه موقع یک شده است یعنی شمارش به عدد سرریز رسیده است.

LOOP: JNB TF1,LOOP (مثال)

۵- پاک کردن  $TF_X$

CLR TF1 (مثال)

۶- خاموش کردن تایمر  $TR_X$

CLR TR1 (مثال)

حال به ترتیبی که گفته شد برنامه‌ای برای ایجاد ۵۰ میلی ثانیه تأخیر می‌نویسیم.

```
DELAY:  
MOV TMOD,#01H  
MOV TH0,#3CH
```

مقداردهی TIMER0 با عدد 3CB0H یا 15536 دسیمال

```
MOV TL0,#0B0H
```

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

SETB TR0

بررسی پرچم Timer0 جهت یک شدن آن

```
LOOP:JNB TF0,LOOP
CLR TF0
CLR TR0
RET
```

زیر برنامه فوق اگر CALL شود 50ms تأخیر ایجاد می‌کند.

ابتدا TMOD مطابق شکل زیر بارگذاری شده است.

TIMER 1				TIMER 0			
GATE	CT	M1	M0	GATE	CT	M1	MO
0	0	0	0	0	0	0	1

در این بارگذاری TIMER0 در حالت ۱۶ بیتی به صورت تایمر و شروع با یک کردن TR

مشخص شده (شروع نرم‌افزاری) و از آنجایی که در این حالت تایمر حداکثر تا عدد ۶۵۵۳۶

شمارش کرده سپس سرریز می‌کند. ما برای تأخیر 50ms می‌بایست عددی را در TH0 و TL0

(بایت پایینی و بالایی تایمر صفر) بارگذاری کنیم که بعد از ۵۰۰۰۰ سیکل یا ۵۰۰۰۰ میکرو ثانیه

سرریز کند (اگر کریستال 12MHz باشد) پس خواهیم داشت.

$$65536 - 50000 = 15536$$

یعنی اگر TH0 و TL0 را در مجموع با عدد ۱۵۵۳۶ بارگذاری کنیم شمارش به صورت

صعودی انجام می‌شود تا به عدد سرریز ۶۵۵۳۶ می‌رسیم در این مدت ما می‌توانیم پرچم TF۰ را

مرتباً چک کنیم که سرریز رخ داده یا خیر و در صورت پیش آمدن سرریز پرچم TF۰ را پاک کرده

و تایمر را نگاه‌داریم و به ادامه کار بپردازیم.



# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

در برنامه عدد 3C در مبنای شانزده را در TH0 و عدد B0 را در TL0 بارگذاری کردیم پس

در مجموع در تایمر صفر عدد 3CB0 در مبنای شانزده را خواهیم داشت که برابر عدد 15536

دسیمال می‌باشد.

در مرحله بعدی مثال، با یک کردن TR0 تایمر را راه‌اندازی کردیم و در خط بعدی با

سرکشی TF0 منتظر یک شدن TF0 شدیم در این حالت که CPU یک حلقه شرطی را مرتباً اجرا

می‌کند پس از یک شدن TF0 از حلقه خارج می‌شود و با صفر کردن TF0 برنامه DELAY را برای

بار دیگر آماده کردیم و تایمر را نگهداشتیم و با رسیدن به RET به برنامه اصلی باز خواهیم گشت.

**نکته:** اگر بخواهیم زمان تأخیر بیشتر از ۶۵ میلی ثانیه باشد باید از یک حلقه DJNZ

استفاده کنیم.

:( )

با یک تأخیر زمانی یک ثانیه مرتباً LED های روی پورت یک را خاموش و روشن نمایید.

```
MAIN:
MOV TMOD,#01H
MAIN LOOP
MOV P1,#0H
CALL DELAY
MOV P1,#0FFH
CALL DELAY
JMP MAIN
DELAY:
MOV R0,#20
DELAY LOOP:
```

```
MOV TH0,#3CH
MOV TH0,#0B0H
SETB TR0
```

$$X = 50 \text{ ms}$$

$$\begin{aligned} Y &= 20 * X \\ Y &= 20 * 50 \text{ ms} = 1000\text{ms} \\ &= 1 \text{ s} \end{aligned}$$

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

LOOP: JNB TF0,LOOP

CLR TF0

CLR TR0

DJNZ R0,DELAYLOOP

RET

END

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

:( )

برنامه رقص نور دو حالت

ORG 0

MAIN:

MOV P1,#55H

CALL DELAY

MOV P1,#0AAH

CALL DELAY

JMP MAIN

DELAY:

MOV R1,#255

MOV R2,#10

LOOP2:

LOOP1:

MOV R0,#255

LOOP0:DJNZ R0,LOOP0

DJNZ R1,LOOP1

DJNZ R2,LOOP2

RET

END

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

( ) :

رقص نور چند حالتی

MAIN:

MOD1:

در این قسمت با توجه به اینکه LEDها به صورت آند مشترک بسته شده‌اند یک LED خاموش

هشت بار به سمت راست جابجا می‌شود.

```
MOV R0,#8
MOV A,#01H
LOOPMOD1:
RR A
MOV P1,A
CALL DELAY
DJNZ R0,LOOPMOD1
```

MOD2:

در این قسمت همان LED در MOD1 هفت بار به سمت چپ جابجا می‌شود.

```
MOV R0,#7
LOOPMOD2:
RL A
MOV P1,A
CALL DELAY
DJNZ R0,LOOPMOD2
MODONOFF:
MOV R0,#5
CLR A
LOOPMODONOFF:
CPL A
MOV P1,A
CALL DELAY
DJNZ R0,LOOPMODONOFF
```

MOD3:

این مد مانند MOD1 عمل کرده با این تفاوت که LED روشن بین هشت LED هشت بار سمت

راست حرکت می‌کند.

```
MOV R0,#8
```

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

```
MOV A,#0FEH
LOOPMOD3:
RR A
MOV P1,A
CALL DELAY
DJNZ R0,LOOPMOD3
```

MOD4:

این مد یک LED روشن را هشت بار به سمت چپ انتقال می‌دهد.

```
MOV R0,#8
LOOPMOD4:
MOV P1,A
CALL DELAY
RL A
DJNZ R0,LOOPMOD4
CALL DELAY
```

MODCPL:

در این مد LEDها یکی در میان روشن و بقیه خاموش می‌شوند و بالعکس.

```
MOV R0,#8
MOV A,#55H
LOOPMODCPL:
CPL A
MOV P1,A
CALL DELAY
DJNZ R0,LOOPMODCPL
```

JMP MAIN

```
;+++++
;          DELAY   One Second Mili SECOND
DELAY:
MOV R7,#4
LDK:
```

در این قسمت یک تأخیر ۲۰۰ میلی ثانیه ایجاد می‌شود.

```
MOV TH0,#3CH
MOV TL0,#0B0H
MOV TMOD,#01H
SETB TR0
```

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

```
LOOPDELAY:JNB TF0,LOOPDELAY  
CLR TR0  
CLR TF0  
DJNZ R7,LDK  
RET
```

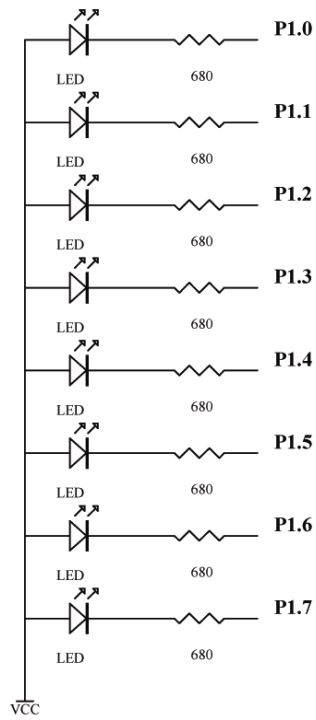
# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

نحوه اتصال LED ها به میکرو کنترلر



## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

:

در برد آموزشی دو کلید فعال به صفر وجود دارد که به P3.3 و P3.5 متصل است شما

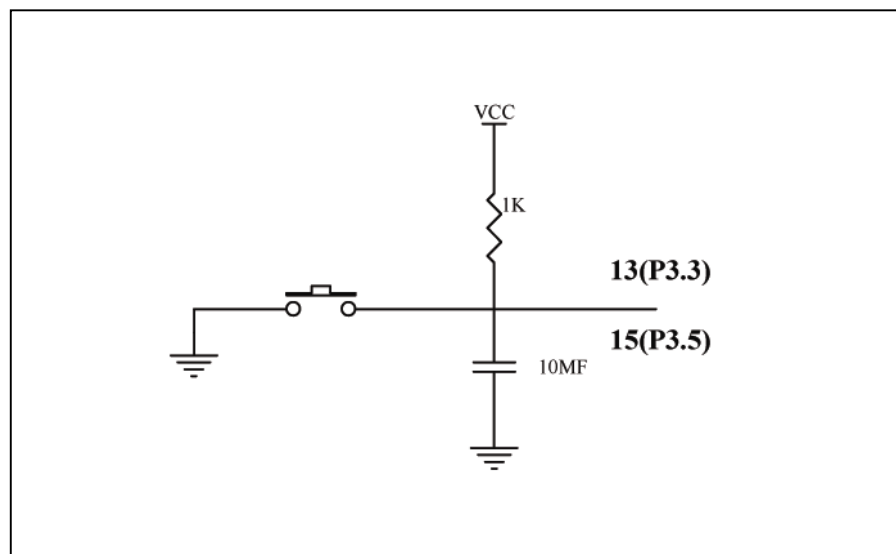
می‌توانید از آنها نیز در طراحی خود استفاده نمایید.

:( )

در برنامه زیر با زدن کلیدها می‌توانید منوی مورد نظر جهت نمایش روی LED های برد

آموزشی را انتخاب نمایید.

```
MAIN:  
JNB P3.0,MENU1  
JNB P3.5,MENU2  
JMP MAIN  
MENU1:  
MOV P1,#55H  
JMP MAIN  
MENU2:  
MOV P1,#0AAH  
JMP MAIN  
END
```





# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

## :KEYBOARD

در صفحه کلیدهای ماتریسی اصول کار به صورت زیر است چنانچه کلیدی زده شود سطر

و ستون کلید مذکور به هم وصل می‌شوند ما می‌توانیم با کنترل دقیق سطرها و ستونها کلید زده شده را پیدا کنیم.

برای این کار به ترتیب زیر عمل می‌کنیم.

۱- سطر یک را زمین می‌کنیم

۲- ستونها را چک می‌کنیم

۳- سطر یک را یک می‌کنیم

۴- سطر دو را زمین می‌کنیم

۵- ستونها را چک می‌کنیم

۶- سطر دو را یک می‌کنیم

۷- سطر سه را زمین می‌کنیم

۸- ستونها را چک می‌کنیم

۹- سطر سه را یک می‌کنیم

۱۰- سطر چهار را زمین می‌کنیم

۱۱- ستونها را چک می‌کنیم

۱۲- سطر چهار را یک می‌کنیم

## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

۱۳- برگشت به مکان اولیه

در مراحل بالادرسورت فشرده شدن هرکلید هنگام چک کردن ستونها ستون مربوط به آن صفر خواهد بود.

بعنوان مثال فرض کنید کلیدی در سطر دوم ، ستون سوم زده شده است ، بعداز صفرکردن سطر

دوم وچک کردن ستونها ستون سوم صفر خواهدبود پس به محل مربوط به آن پرش می‌نمائیم.

حال با توجه به مطالب گفته شده به حل یک تمرین می‌پردازیم.

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

:( )

برنامه‌ای بنویسید که ابتدا صفحه کلید چک شود و هر کلیدی که انتخاب شد متناظر باینری

آن کلید بر روی پورت یک مشخص شود.

برای حل این تمرین ابتدا با استفاده از دستور EQU تمامی سطرها و ستون‌ها را با پینی که

به آن متصل هستند مشخص می‌کنیم.

ROW1	EQU	P2.7
ROW2	EQU	P2.6
ROW3	EQU	P2.5
ROW4	EQU	P2.4
COL1	EQU	P2.3
COL2	EQU	P2.2
COL3	EQU	P2.1
KEYBUF	EQU	30H

معرفی یک خانه از حافظه بنام KEYBUF

```
MAIN:  
CALLKEYSCAN  
MOV P1,KEYBUF  
JMP MAIN
```

```
KEYSCAN:  
MOV P2,#0FFH  
CLR ROW1  
JNB COL1,KEY1
```

بررسی ستون اول

```
JNB COL2,KEY2
```

بررسی ستون دوم

```
JNB COL3,KEY3
```

بررسی ستون سوم

```
SETB ROW1
```

برگرداندن سطر صفر شده به حالت اول

```
CLR ROW2
```

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

صفر کردن سطر دوم

JNB COL1,KEY4

اگر سطر اول صفر شد به منزله فشردن کلیدی در سطر دوم و ستون دوم است.

JNB COL2,KEY5

JNB COL3,KEY6

SETB ROW2

CLR ROW3

JNB COL1,KEY7

JNB COL2,KEY8

JNB COL3,KEY9

SETB ROW3

CLR ROW4

JNB COL3,KEYNUMBER

JNB COL2,KEY0

JNB COL1,KEYSTAR

JMP KEYSKAN

KEY1:

MOV KEYBUF,#01H

RET

KEY2:

MOV KEYBUF,#02H

RET

KEY3:

MOV KEYBUF,#03H

RET

KEY4:

MOV KEYBUF,#04H

RET

KEY5:

MOV KEYBUF,#05H

RET

KEY6:

MOV KEYBUF,#06H

RET

KEY7:

MOV KEYBUF,#07H

RET

KEY8:

MOV KEYBUF,#08H

RET

KEY9:

MOV KEYBUF,#09H

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

```
RET  
KEY0:  
MOV KEYBUF,#0H  
RET  
KEYSTAR:  
MOV KEYBUF,#0AAH  
RET  
KEYNUMBER:  
MOV KEYBUF,#55H  
RET  
END
```

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

## نحوه کار با Stepper motor (موتور پله ای)



با مطالعه این بخش شما می‌توانید موتورهای پله‌ای چهار فاز را راه‌اندازی

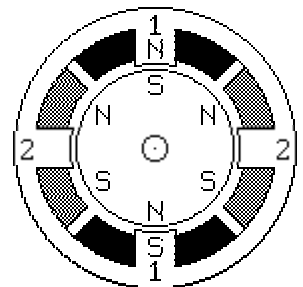
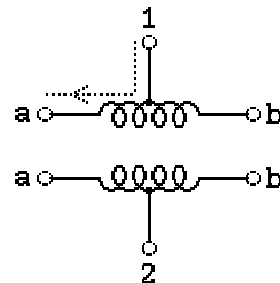
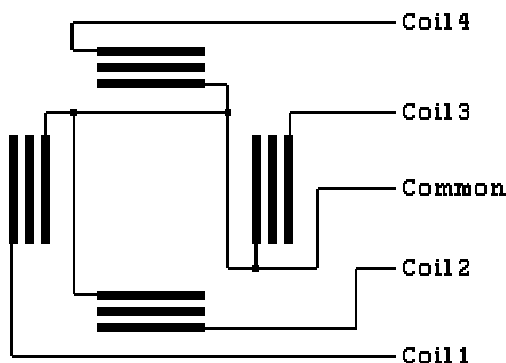
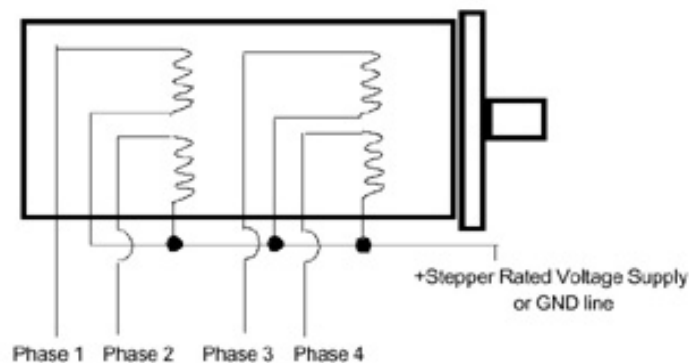
نمائید.

یک موتور پله‌ای چهار فاز از چهار سیم پیچ تشکیل شده است که از یک

سر مشترک شده‌اند.

البته توجه داشته باشید این امکان وجود دارد که سر سیم پیچ‌ها دوبه‌دو باهم مشترک شده

باشند که می‌توان گفت سیم‌های خروجی یک موتور چهار فاز پنج یا شش رشته می‌باشند.



## پیشرو نوآوران کاوش

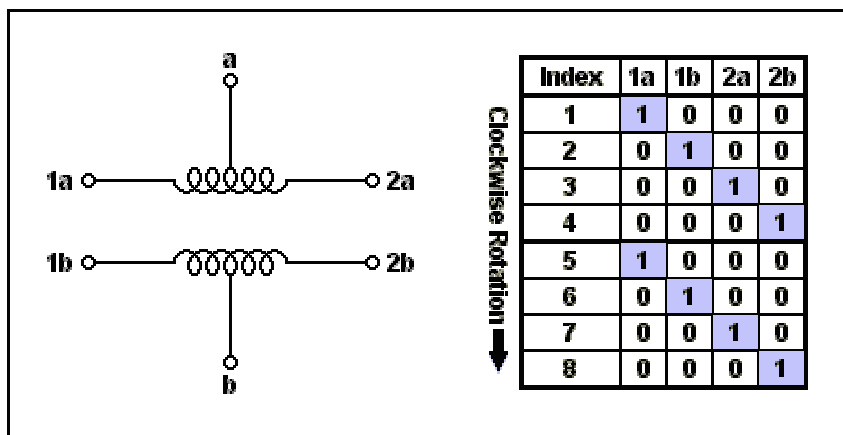
اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

جهت راه‌اندازی یک موتور پله ای شما می‌توانید سر مشترک را به VCC متصل نمایید و با

زمین کردن سر سیم‌پیچ‌ها به ترتیب زیر آن‌را در جهت دلخواه به حرکت در آورید.



به این نکته توجه کنید که میکرو به تنهایی نمی‌تواند جریان و ولتاژ لازم را برای موتور

تأمین نماید به همین دلیل از ترانزیستورهای قدرت استفاده کرده‌ایم که عمل Not را نیز در

خروجی‌ها اعمال می‌نماید، در مجموع با یک کردن بیس هر ترانزیستور مطابق شکل کلکتور آن به

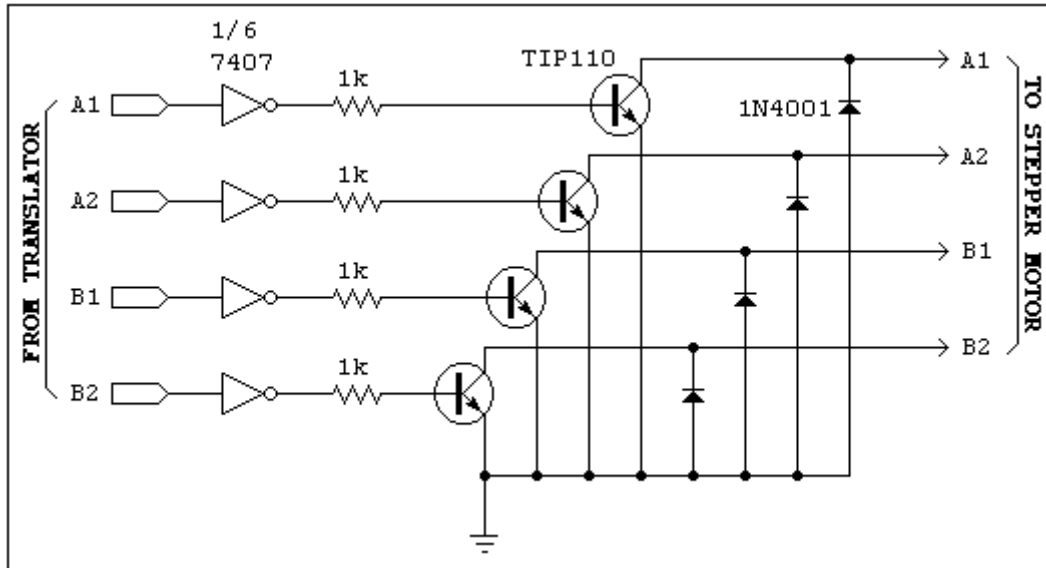
زمین متصل می‌گردد.

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک



چون سر مشترک سیم پیچ‌ها نیز VCC شده‌اند اطراف سیم پیچ انتخاب شده میدان مغناطیسی

ایجاد می‌گردد که باعث جذب روتور می‌شود و به همین طریق می‌توان در یک موتور حرکت ایجاد نمود.

چنانچه شما همواره بترتیب دو فاز کنار هم را در موتور فعال سازید روتور تحت تاثیر

برآیند دو میدان از طرف سیم پیچ‌ها قرار خواهد گرفت در نتیجه موتور با قدرت بیشتری دوران خواهد کرد.

در جدول زیر مقادیری را که سیم پیچ‌های موتور پله‌ای برای به حرکت در آمدن احتیاج

دارد مشاهده می‌کنید.

	سیم پیچ A	سیم پیچ B	سیم پیچ C	سیم پیچ D	
در جهت	پله ۱	1	0	0	1
	پله ۲	1	1	0	0



# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

ساعت	۳	0	1	1	0	ساعت
	۴	0	0	1	1	

توجه داشته باشید که می‌توان از هر یک از پله‌های جدول بالا شروع کرد ولی به محض

شروع باید ترتیب رعایت شود.

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

:( )

برنامه‌ای بنویسید که موتور مرتباً بچرخد.

```
MAIN:  
MOV A,#66H  
BACK:  
RR A  
MOV P1,A  
CALL DELAY  
JNB P3.5,LEFT  
JMP BACK
```

```
DELAY:  
MOV R7,#4  
LDK:
```

در این قسمت یک تأخیر ۲۰۰ میلی ثانیه ایجاد می‌شود.

```
MOV TH0,#3CH  
MOV TL0,#0B0H  
MOV TMOD,#01H  
SETB TR0  
LOOPDELAY:JNB TF0,LOOPDELAY  
CLR TR0  
CLR TF0  
DJNZ R7,LDK  
RET
```

```
END
```

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

:( )

برنامه‌ای بنویسید که با فشار دادن P3.3 موتور راست‌گرد و با فشار دادن کلید P3.5 موتور چپ‌گرد

بچرخد.

```
ORG 00H
KEYSCAN:
JNB P3.3,RIGHT
JNB P3.5,LEFT
JMP KEYSKAN

RIGHT:
MOV A,#66H
BACK:
RR A
MOV P1,A
CALL DELAY
JNB P3.5,LEFT
JMP BACK

LEFT:
MOV A,#66H
BACK:
RL A
MOV P1,A
CALL DELAY
JNB P3.3,RIGHT
JMP BACK

DELAY:
MOV R7,#4
LDK:

MOV TH0,#3CH
MOV TL0,#0B0H
MOV TMOD,#01H
SETB TR0
```

در این قسمت یک تأخیر ۲۰۰ میلی ثانیه ایجاد می‌شود.

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

```
LOOPDELAY:JNB TF0,LOOPDELAY  
CLR TR0  
CLR TF0  
DJNZ R7,LDK  
RET  
END
```

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

:

# پیشرو نوآوران کاوش

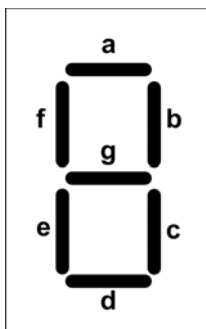
اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

## 7SEGMENT

7SEGMENT از ترکیب هفت LED ساخته شده است که LEDها به صورت منظم چیده



شده‌اند که آن‌ها را با حروف لاتین مشخص کرده‌ایم. جهت نمایش عدد

مورد نظر قسمت‌های مختلف می‌بایست انتخاب شوند.

مثال: برای نمایش عدد 5 می‌بایست حروف a, f, g, c, d

همگی روشن شوند.

توجه: البته 7segmentها آند و یا کاتد LED آن‌ها بسته به نوع segment مشترک شده‌اند که در برد

آموزشی 7segment از نوع آند مشترک می‌باشد.

جهت سادگی کار و حداکثر استفاده از میکروکنترلر ما از یک مبدل BCD به 7segment

استفاده کردیم که شما با قراردادن عدد خود به صورت BCD بر روی P2 مبدل فوق که از IC

7447 استفاده شده است آن را روی 7SEGMENT خواهید دید. برای داشتن تعداد رقم‌های بیشتر

در 7SEGMENT می‌توان چند 7SEGMENT را به صورت ماتری پلکس شده استفاده کرد، بدین

صورت که تمام قسمت‌ها یک 7SEGMENT به 7SEGMENT بعدی نیز متصل است مثلاً a1 به

a2 و به a3 و...

حال اگر ما پایه مشترک آن‌ها را (در برد آموزشی آند) نیز وصل کنیم به صورت همزمان

همه segmentها یک عدد را نمایش می‌دهند اگر به صورت منظم و برنامه‌ریزی دقیق آن‌ها

انتخاب شوند شما خواهید توانست رقم دلخواه خود را نمایش دهید.

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

در برد آموزشی آنها به P1.4 و P1.5 متصل شده‌اند پس اگر عدد 15H را روی P1 بریزیم

مشاهده خواهید کرد. عدد ۵ نمایش داده می‌شود و عدد یک روشن کننده آند 7segment سمت

راستی است.

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

( ) :

برنامه زیر عدد ۳ را در رقم اول نمایش می‌دهد.

```
MAIN:  
MOV P1,#13H  
LOOP:JMP LOOP  
END
```

نمونه برنامه (۱۳): برنامه زیر عدد ۴ را در رقم دوم نمایش می‌دهد.

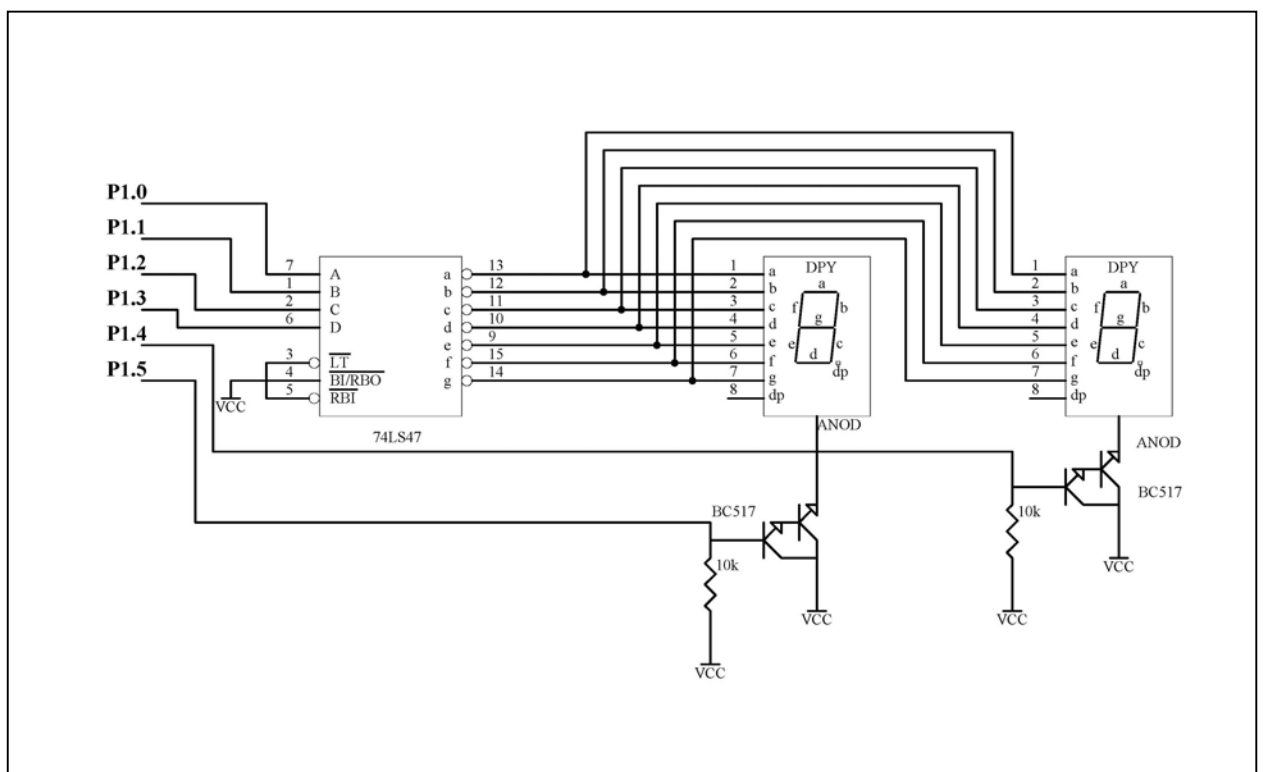
```
MAIN:  
MOV P1,#24H  
LOOP: JMP LOOP  
END
```

نمونه برنامه (۱۴): برنامه زیر عدد 34 را روی 7SEGMENT نمایش می‌دهد.

```
MAIN:  
MOV P1,#13H  
NOP  
MOV P1,#24H  
JMP MAIN  
END
```

در مثال بالا به این دلیل که سریعاً عدد ۳ و ۴ روی 7SEGMENT نمایش داده می‌شود و

همچنین با در نظر گرفتن خطای دید، چشم، بیننده عدد 34 را مشاهده می‌کند.





## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

شاید این سوال برای شما پیش آید که اگر عدد ما یک عدد ثابت نبود و به طور مداوم تغییر

کند چگونه می‌توان این عدد را نمایش داد. در این صورت می‌بایست هر رقم را داخل یک بایت

ریخته و تک تک آن‌ها را روی 7SEGMENT انتقال دهیم.

SHOW:

MOV A,VALUE

MOV B,10

DIV AB

ADD A,#10H

MOV P1,A

CALL DELAY

ADD B,#20H

MOV P1,B

CALL DELAY

JMP SHOW

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

وقفه یکی از مباحث اساسی در میکروکنترلرها است که احتیاج به توجه ویژه‌ای دارد.

در ابتدا باید وقفه را تعریف کرد. می‌توان گفت که وقفه یعنی درخواست از CPU. با توجه

به این تعریف می‌توان فهمید که وقفه برای انجام عملی وارد می‌شود. هنگام وارد شدن یک وقفه

cpu چهار کار را انجام میدهد که به ترتیب زیر است.

۱- منبع وقفه برای CPU مشخص است.

۲- آدرس برنامه‌ای را که باید انجام دهد، را می‌داند و به آنجا می‌رود.

۳- برنامه مورد نظر را در آنجا انجام می‌دهد.

۴- بعد از انجام برنامه مورد نظر به مکان اولیه قبل از وارد شدن به وقفه برمی‌گردد و کار را

ادامه می‌دهد.

در 89C51 در مجموع پنج منبع وقفه وجود دارد:

Timer0 در هنگام سرریز.

Timer1 در هنگام سرریز.

External0 (خارجی) از طریق پایه ۱۲

External1 (خارجی) از طریق پایه ۱۳

## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

Serial در هنگام کامل شدن دریافت یا ارسال.

در بالا شرایطی را که باعث وارد شدن وقفه به CPU می‌شود مشخص شده‌است، حال باید

بدانیم هنگام وارد شدن هر وقفه چه اتفاقی می‌افتد.

در هنگام وارد شدن هر وقفه CPU به خانه خاصی از حافظه (در جدول زیر آدرس این

نقاط گفته شده است) پرش می‌کند و برنامه از آن خانه ادامه می‌یابد و تا زمانی که به دستور

RETI نرسیده، CPU برنامه را اجرا می‌کند به محض رسیدن به دستور RETI، CPU به محلی

که وقفه وارد شده بود بازگشته و ادامه کار خود را انجام می‌دهد حال کافی است شما با در نظر

گرفتن جدول زیر برنامه‌ای را که باید در هنگام وارد شدن وقفه اجرا شود در محل خاص حافظه

بنویسید.

IE							
EA	NU	ET2	ES	ET1	EX1	ET0	EX0
فعال ساز کلکی	استفاده نشده	2BH	23H	1BH	13H	0BH	03H
		تایمر ۲ در 89C52	درگاه سریال	تایمر ۱	خارجی یک	تایمر صفر	خارجی صفر

بایت IE جهت مشخص نمودن و فعال کردن وقفه مورد نظر به کار می‌رود کافی است شما

بیت نظیر وقفه مورد استفاده در برنامه را همراه بیت EA فعال کنید.

مثال: چه عددی را می‌بایست در IE بارگذاری کرد که وقفه صفر خارجی فعال شود؟

## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

IE							
EA	NU	ET2	ES	ET1	EX1	ET0	EX0
1	0	0	0	0	0	0	1
8				1			

MOV IE,#81H

برای اولین مثال از وقفه خارجی استفاده می‌کنیم. پایه‌های ۱۲ و ۱۳ در روی پورت سه قرار

دارند که همان گونه که قبلاً گفته شد برای کارهای خاص در نظر گرفته شده‌اند.

برای اینکه یک وقفه خارجی رخ دهد باید دو شرط رعایت شود که با داشتن این دو شرط

وقفه خارجی صادر می‌شود. این دو شرط به قرار زیر هستند:

الف: ثبات IE را طوری مقداردهی کنیم که حداقل یکی از وقفه‌های خارجی قابلیت فعال شدن را

داشته باشند.

ب: شرط دوم وارد آمدن یک پالس به یکی از پایه‌های ۱۲ یا ۱۳ بترتیب برای وقفه خارجه صفر یا

وقفه خارجی یک است.

## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

( ) :

برنامه‌ای بنویسید که در صورت وارد شدن یک پالس به پایه ۱۲ همه پایه‌های پورت یک

صفر شود.

```
ORG 0
JMP MAIN
ORG 03H
INTETERNAL:
MOV P1,#00H
RETI
MAIN:
MOV IE,#81H
SETB ITO
SJMP $
END
```

در برنامه فوق ابتدا با پرش به MAIN خانه 3H که مربوط به وقفه خارجی یک بوده را پرش

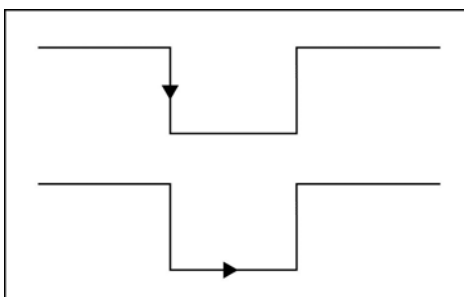
کردیم تا فضای برنامه وقفه را خالی نگه داشته باشیم در MAIN وقفه مورد نظر فعال شده و ITO

یک شده است.

بیت ITO مشخص کننده فعال‌سازی وقفه خارجی صفر در لبه پالس یا سطح صفر پالس

است.

شما می‌توانید با زدن کلید وقفه روی برد آموزشی یک پالس روی پایه ۱۳ ایجاد نمایید.



وارد شدن وقفه در حالت لبه

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

## وارد شدن وقفه در حالت سطح

IT0 نیز مربوط به وقفه خارجی صفر است.

اگر  $IT=0$  وقفه حساس به سطح خواهد بود.

اگر  $IT=1$  وقفه حساس به لبه

حال اگر شما با کلید بر روی برد آموزشی یک پالس ایجاد نمایید LED های روی برد

آموزشی روشن می‌شود.

:

حال به بررسی چگونگی رخ دادن وقفه‌های تایمر می‌پردازیم.

برای وقوع یک وقفه تایمری هم دو شرط لازم است. گاهی اوقات احتیاج داریم که اتفاق

خاصی در فواصل زمانی معینی رخ دهد برای اینکار می‌توان از وقفه تایمری استفاده کرد.

:

الف: ثبات IE را طوری مقداردهی کنیم که حداقل یکی از وقفه‌های تایمری قابلیت فعال شدن

را داشته باشند.

ب: سرریز شدن تایمر ( $TF=1$ ). با توجه به مدت زمانی که در تایمر بارگذاری شده با تمام

شدن این زمان پرچم TF برابر یک می‌شود.

برای روشن شدن این موضوع به حل یک تمرین می‌پردازیم.

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

( ) :

برنامه‌ای بنویسید که موج مربعی 8KHz روی پایه P1.6 تولید نماید.

```
ORG 0H
JMP MAIN
ORG 0BH
CPL P1.6
RETI
MAIN:
MOV IE,#82H
MOV TMOD,#02H
MOV TH0,#62
SETB TR0
SJMP $
END
```

در برنامه فوق ابتدا با پرش به MAIN خانه‌های حافظه را برای نوشتن برنامه وقفه خالی

نگه‌داشتیم.

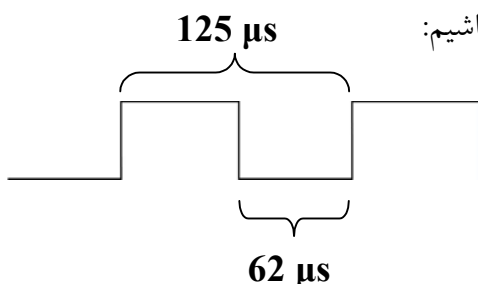
ثبات کنترل وقفه (IE) را با عدد ۸۲ جهت فعال‌سازی وقفه تایمر صفر و EA بارگذاری

نمودیم.

جهت ایجاد یک شکل موج مربعی کافی است آن پایه معکوس شود و سرعت معکوس

شدن همان فرکانس موج ایجاد شده خواهد بود.

اگر بخواهیم یک شکل موج مربعی 8KHz داشته باشیم:



# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

$$1000000/8000 = \text{مقدار بارگذاری در تایمر}$$

$$125 \mu\text{s} / 2 = 62.5 \sim 62 \mu\text{s} = \text{مقدار بارگذاری در تایمر}$$

نمونه دیگری از کاربرد وقفه‌های تایمری را در تمرین زیر مشاهده می‌فرمایید.

( ) :

با استفاده از وقفه تایمر برنامه‌ای بنویسید که بایت SEG1 و SEG2 را روی خانه اول و دوم

7SEGMENT برد آموزشی نمایش دهد.

```
SEG1 EQU 30H
SEG2 EQU 31H
```

```
ORG 0H
JMP MAIN
ORG 0BH
```

```
CLR TR0
MOV TH0,#0ECH
MOV TL0,#078H
MOV P1,SEG1
SETB P1.4
CALL DELAY
CLR P1.4
MOV P1,SEG2
SETB P1.4
CALL DELAY
CLR P1.4
SETB TR0
RETI
```



# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

MAIN:

```
MOV IE,#82H  
MOV TMOD,#11H  
MOV TH0,#0ECH  
MOV TL0,#78H  
SETB TR0  
SJMP $
```

DELAY:

```
MOV TH1,#0FCH  
MOV TL1,#18H  
SETB TR1  
LOOPDELAY:JNB TF1,LOOPDELAY  
CLR TR1  
CLR TF1  
RET  
END
```

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

( ) :

در نمونه بر نامه ۱۵ چگونگی تولید کردن یک شکل موج را با استفاده از وقفه تایمر مشاهده

کردید در تمرین زیر چگونگی تولید دو شکل موج مختلف را به صورت همزمان مشاهده می‌کنید.

با استفاده از وقفه تایمر دو شکل موج 500H و 7KH تولید کنید.

```
ORG 00H
JMP MAIN
ORG 0BH
JMP TS0
ORG 1BH
JMP TS1

MAIN:
MOV TMOD,#12H
MOV TH0,#-71
SETB TR0
SETB TF1
MOV IE,#8AH
SJMP $

TS0:
CPL P1.7
RETI

TS1:
CLR TR1
MOV TH1,#HIGH(-1000)
MOV TL1,#LOW(-1000)
SETB TR1
CPL P1.6
RETI
END
```

## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

**نکته:** فرکانس‌های زیر 1KHZ را با استفاده از حالت یک تایمر می‌توان تولید کرد. پس برای تولید شکل موج 7KHZ از حالت دو تایمر صفر و برای تولید شکل موج 500HZ از حالت یک، تایمر یک استفاده می‌کنیم.

همان‌گونه که می‌دانیم وقفه تایمر وقتی فعال می‌شود که دو شرط بارگذاری صحیح IE و سرریز شدن تایمر به وجود آید بنابراین برای تولید فرکانس 7KHZ به راحتی با استفاده از وقفه تایمر صفر و حالت یک آن می‌توان این کار را انجام داد.

برای تولید فرکانس 500HZ ابتدا در آدرس 1BH که محل پرش وقفه تایمر یک است برنامه‌ای برای تولید این موج می‌نویسیم و حالا برای اجرا شدن این برنامه کافی است که وقفه تایمر یک دو شرط فعال شدن وقفه‌ها را داشته باشد. شرط بارگذاری ثبات وقفه را که قبلاً انجام داده‌ایم و برای اینکه شرط دوم  $TF=1$  را فراهم کنیم در قسمت MAIN برنامه این کار انجام می‌دهیم. بنابراین بعد از هر وقفه که تایمر صفر ایجاد می‌کند خود به خود برنامه به  $TF1=1$  می‌رسد و به آدرس 1BH می‌رود و در آنجا شکل موج 500HZ را همزمان با فرکانس 7KHZ ایجاد می‌کنیم.

:

(MSB)				(LSB)			
-	-	PT2	PS	PT1	PX1	PTO	PXO

عملکرد	موقعیت	نماد
رزرو شده	IP.7	-

## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

رزرو شده	IP.6	-
برای تعیین سطح تقدّم وقفه تایمر 2 بکار می رود.	IP.5	PT2
اگر $PT2 = 1$ باشد، این وقفه در سطح تقدم بالاتر قرار می گیرد.		
برای تعیین سطح تقدم وقفه درگاه سریال بکار می رود.	IP.4	PS
اگر $PS = 1$ باشد، این وقفه در سطح تقدم بالاتر قرار می گیرد.		
برای تعیین سطح تقدم وقفه تایمر 1 بکار می رود.	IP.3	PT1
اگر $PT1 = 1$ باشد، این وقفه در سطح تقدم بالاتر قرار می گیرد.		
برای تعیین سطح تقدم وقفه تایمر 0 بکار می رود.	IP.1	PT0
اگر $PT0 = 1$ باشد، این وقفه در سطح تقدم بالاتر قرار می گیرد.		
برای تعیین سطح تقدم وقفه خارجی 0 بکار می رود.	IP.0	PX0
اگر $PX0 = 1$ باشد، این وقفه در سطح تقدم بالاتر قرار می گیرد.		
کاربر نرم‌افزار نباید بیت‌های تعریف نشده را یک کند، تا وقتی که از آن‌ها در محصولات آینده MCS-51 استفاده شود.		

اگر یکی از شرایط زیر برقرار باشد. سخت‌افزار عملیات فراخوانی وقفه را انجام نخواهد داد:

۱. وقفه‌ای با حق تقدم برابر یا بالاتر، از قبل در حال اجرا باشد. اجرای وقفه با حق تقدم

پائین‌تر توسط وقفه با حق تقدم بالاتر متوقف می‌شود؛ البته اگر وقفه با حق تقدم بالاتر

فعال شده باشد.

۲. سیکل ماشین فعلی، آخرین سیکل دستورالعمل در حال اجرا نباشد. اجرای

دستورالعمل باید قبل از پرش به دستورالعمل‌های سرویس‌دهی وقفه به اتمام برسد.

۳. دستورالعمل در حال اجرا، دستورالعمل RETI یا هر دستورالعملی که عمل نوشتن در

رجیسترهای IE یا IP را انجام می‌دهد، باشد. حداقل باید یک دستورالعمل اضافی بعد

## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

از دستورالعمل‌هایی که پیش از پرش به آدرس بردار وقفه قرار داشتند، اجرا شود. دلیل

استفاده از این شرط، جلوگیری از رخ دادن یک وقفه در میان اجرای یک روال است

که بنام فرآیند پیکربندی دوباره وقفه‌ها نیز مشهور است. کاربردهای راه‌اندازی شونده

با وقفه بدون در نظر گرفتن بی‌نظمی حاصل از دسترسی نداشتن به رجیسترهای کنترل

وقفه، که وظیفه تکرار وقفه‌ها را به‌عهده دارند، نیز به اندازه کافی پیچیده هستند.

مدت زمانی را که بین فعال شدن یک وقفه و شروع اجرای روال سرویس‌دهی طول

می‌کشد، زمان پاسخ می‌نامند. در 8051 کوتاهترین زمان پاسخ برابر ۳ سیکل ماشین و طولانی‌ترین

(بدترین حالت) آن برابر ۹ سیکل ماشین می‌باشد. در کاربردهای با محدودیت زمانی، باید فرض را

بر آن بگذاریم که بدترین حالت اتفاق خواهد افتاد.

:

در حالت کلی، وقفه‌ها می‌توانند حساس به سطح یا حساس به تغییر حالت (تریگر شونده با

سطح یا تریگر شونده با لبه) باشند. رویداد حساس به لبه را گذرا می‌نامند، زیرا وقتی بیت پرچم

مربوط به وقفه آن پاک می‌شود، خود رویداد وقفه دهنده نیز پاک می‌گردد (متوقف می‌شود).

به عبارت دیگر، پاک کردن بیت پرچم وقفه حساس به سطح به شرط اینکه سطح منطقی خارجی که

وقفه را ایجاد کرده است فعال بماند، هیچ تأثیری بر روند اجرای وقفه نخواهد گذاشت.

در 8051، می‌توانیم وقفه‌های خارجی INT0 و INT1 را به‌طور مجزا به یکی از دو حالت

حساس به سطح و حساس به لبه پیکربندی کنیم. بدین منظور از بیت‌های IT0 و IT1 رجیستر

## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

TCON استفاده می‌کنیم. هنگام رخ دادن وقفه، بیت پرچم یک می‌شود و وقتی فراخوانی از بردار

وقفه صورت می‌پذیرد، بیت پرچم به‌طور خودکار صفر می‌شود.

در مورد تایمرها، پرچم‌های TF0 و TF1 وقتی یک می‌شوند که عمل شمارش در رجیستر

شمارندهٔ مربوطه یک دور بزند، یعنی همهٔ یک‌ها به صفر تبدیل شوند، وقفه‌های شمارنده حساس

به لبه هستند. همچنین، TF0 و TF1 در طول سرویس‌دهی وقفه به‌صورت خودکار پاک می‌شوند.

وقفهٔ درگاه سریال به‌هنگام یک شدن بیت RI یا TI، فعال می‌شود. البته، RI و TI به‌طور

خودکار پاک نمی‌شوند. روال سرویس‌دهی وقفه، خود تشخیص می‌دهد که کدام بیت، وقفه را

ایجاد کرده است و سپس به‌عنوان اجرای قسمتی از روال، آن بیت را پاک می‌کند.

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

:

SMO	SM1	SM2	REN	TB8	RB8	TI	RI
-----	-----	-----	-----	-----	-----	----	----

SM0	SM1	
0	0	حالت صفر
0	1	حالت یک
1	0	حالت دو
1	1	حالت سه

## SM0, SM1

جهت مشخص نمودن حالت درگاه سریال طبق جدول بالا به کار می‌رود.

:

حالت صفر:

حالت صفر از عملیات نیمه دو طرفه همزمان تشکیل یافته است. داده‌ها از طریق پایه RXD و در دسته‌های ۸ بیتی ارسال و دریافت می‌شوند (البته نه به‌طور همزمان). به‌طوری‌که ابتدا کم ارزش‌ترین بیت ارسال یا دریافت می‌شود. نرخ ارسال / دریافت داده برابر با یک دوازدهم فرکانس اسپیلاتور است. حداکثر سرعت انتقال که هم فرکانس با نرخ ارسال / دریافت داده است، در طول هر دو عملیات ارسال و دریافت برای همزمان کردن گیرنده و فرستنده بکار می‌رود و در پایه TXD قبل دسترسی می‌باشد. لبه ساعت انتقال در طول حالت معتبر هر بیت داده اتفاق می‌افتد. توجه

## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

داشته باشید که TXD و RXD کاربردهای دوم پایه‌های درگاه 3 هستند. در حالت صفر، عملیات دریافتی وقتی آغاز می‌شود که بیت REN و RI در رجیستر SCON به ترتیب 1 و 0 شده باشند.

### حالت یک:

حالت یک، به صورت عملیات غیر همزمان دو طرفه کامل است. داده از طریق پایه TXD ارسال و از طریق پایه RXD دریافت می‌شود. قالب کامل داده در ارسال/دریافت سریال شامل یک بیت شروع (همیشه صفر است)، به دنبال آن ۸ بیت داده (ابتدا کم ارزش‌ترین بیت)، و در انتها یک بیت توقف (همیشه حداقل یک است)، ارسال می‌شود. بیت‌های شروع و توقف توسط سخت‌افزار به ۸ بیت داده که از طریق نرم‌افزار به رجیستر SBUF نوشته شده یا از آن خوانده شده است، اضافه می‌گردند. نرخ ارسال بیت متغیر است و می‌توان با استفاده از تایمر 1 به عنوان مولد نرخ ارسال بیت، آن را تعیین کرد.

### حالت دو:

حالت دو شبیه حالت یک است، به غیر از دو استثناء که وجود دارد. اولاً، قالب داده ۱۱ بیتی است؛ یک بیت داده به عنوان نهمین بیت داده، قبل از بیت توقف اضافه شده است. موقع ارسال بیت نهم را می‌توان از بیت TB8 رجیستر SCON خواند. فرض بر این است که محتوی TB8 قبل از آغاز عملیات ارسال نوشته شده است. در عملیات دریافت، بیت نهم را می‌توان از بیت RB8 رجیستر SCON خواند. از بیت نهم معمولاً به عنوان بیت توازن ۸ بیت داده استفاده می‌شود. ثانیاً، نرخ ارسال بیت برابر یک سی و دوم یا یک شصت و چهارم فرکانس اسیلاتور می‌باشد، که انتخاب



## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

یکی از این دو نرخ ارسال توسط بیت SMOD (بیت هفتم) رجیستر PCON صورت می‌گیرد. اگر SMOD یک شود، 1/32 و اگر صفر شود، 1/64 فرکانس سیلاتور مورد استفاده قرار می‌گیرد.

### حالت سه:

حالت سه شبیه حالت دو است، به استثنای اینکه نرخ ارسال بیت در این حالت متغیر است و طبق روشی که در حالت یک گفته شد، تعیین می‌شود. عملیات دریافت در حالت‌های 1، 2 و 3 با یک کردن بیت REN رجیستر SCON فعال می‌شود. دریافت واقعی وقتی شروع می‌شود که بیت شروع ورودی، سبب ایجاد یک تغییر حالت یک به 0 در پایه RXD شود. همان‌طور که گفته شد، در هر حالت کاری عملیات ارسال با نوشتن در SBUF آغاز می‌گردد.

توجه کنید در حالت‌های یک و سه که نرخ ارسال و دریافت توسط **TIMER** مشخص می‌شوند

بهتر است **TIMER1** در حالت دو انتخاب شود (**8 bit auto reload**) در این دو حالت نرخ

ارسال و دریافت از شکل زیر پیروی می‌کند.

استفاده از کریستال (نوسان‌ساز) 11.0592MHz به این علت است تا عدد خروجی یک عدد

صحیح و بدون اعشار ۲۸۸۰۰ باشد سپس می‌توانیم با بارگذاری **TIMER1** نرخ ارسال را طبق

جدول زیر تغییر دهیم.

x	Baud Rate
-2	14400
-3	9600
-6	4800

## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

-12	2400
-24	1200
-48	600

**:SM2**

بیشتر در ارتباط چند ریزپردازنده مورد استفاده قرار می‌گیرد. چنانچه SM2 انتخاب شده

باشد شرط وارد شدن وقفه سریال دریافت بیت نهم است. این بیت در حالت عادی صفر است.

REN اگر این بیت یک شده باشد پایه RXD روی 8051 فعال می‌شود.

**:RB8**

بیت نهم ارسال است اگر یک باشد بیت نهم یک ارسال شده و اگر صفر باشد بیت نهم صفر

ارسال می‌شود در حالت عادی این بیت صفر است.

**:TB8**

بیت دریافت شده نهم در این حافظه قرار می‌گیرد و در حالت عادی صفر است. شاید از

خود سوال کنید که چرا این دو بیت وجود دارند بدین دلیل که SBUF 8 بیت بی‌تی است و

نمی‌تواند P یا بیت نهم را هم در خود جای دهد به همین دلیل این دو بیت را جهت دریافت یا

ارسال بیت نهم قرار دادند.

جهت ارسال اطلاعات به درگاه سریال RS232 به طور مثال درگاه سریال کامپیوتر باید به

نکات زیر توجه کنید.

## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

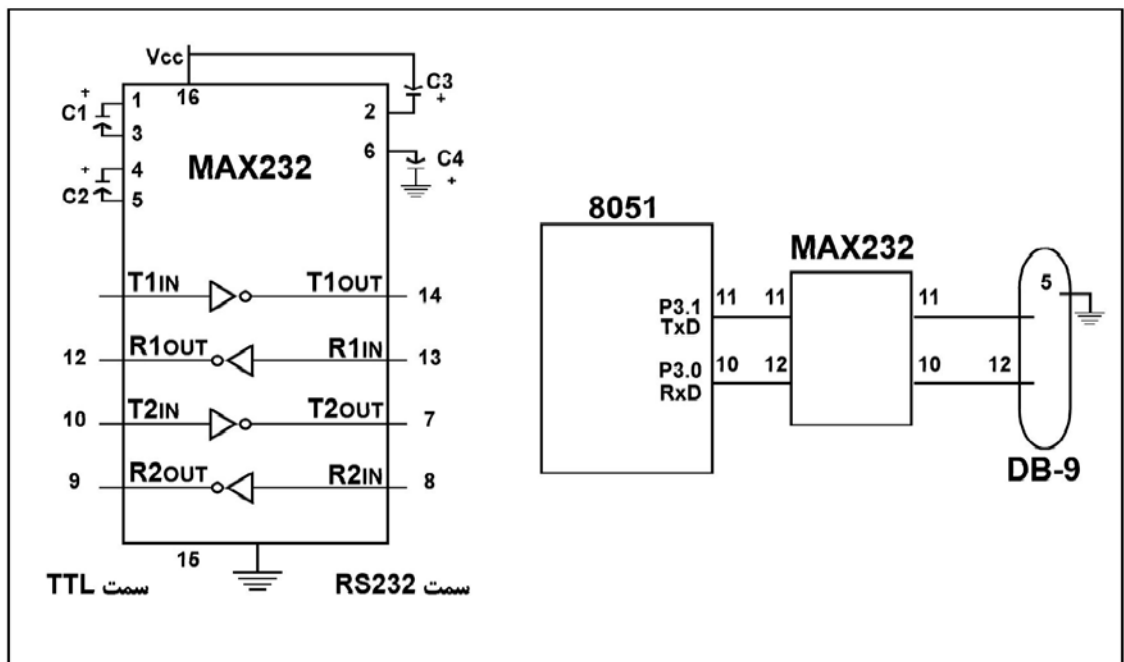
آموزش دوره‌های تخصصی الکترونیک

۱- از آنجاییکه در میکروکنترلر سطح منطقی یک +5 و صفر GND است و در استاندارد

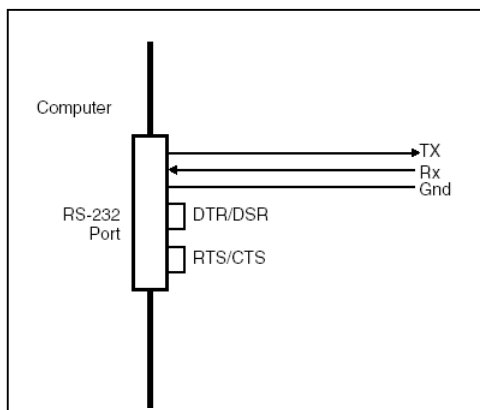
RS232 ، -15 تا -3 سطح یک و +15 تا +3 سطح منطقی صفر است. باید این تبدیل از

سطح TTL به RS232 صورت بگیرد این تبدیل می‌تواند توسط آی‌سی Max232 به

صورت شکل زیر صورت بگیرد.



۲- دومین نکته تهیه یک کابل رابط از کامپیوتر به 89C51 می‌باشد که شما می‌توانید مانند



شکل زیر عمل نمایید.

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

در ارسال داده از طریق سریال باید مراحل زیر را طی کرد:

۱- بارگذاری SCON جهت انتخاب حالت ارسال

۲- بارگذاری TMOD برای حالت ۱ و ۳

۳- بارگذاری TIMER1 برای حالت ۱ و ۳

۴- انتقال داده در SBUF

۵- سرکشی پرچم ارسال در این قسمت TI را سرکشی کرده تا ارسال تمام شود بعد از اتمام

کامل ارسال TI به صورت سخت‌افزاری یک می‌شود.

۶- پاک کردن پرچم TI

:( )

برنامه‌ای بنویسید که مقادیر روی پورت دو را بر روی درگاه سریال با نرخ 9600bps ارسال

کند.

```
MAIN:
MOV SCON,#01010000B
MOV TMOD,#20H
MOV TH1,# -3
SETB TR1
NEXT:
MOV A,P2
MOV SBUF,A
LOOP:JNB TI,LOOP
CLR TI
JMP NEXT
END
```

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

در دریافت داده از طریق سریال باید مراحل زیر را طی کرد:

۱- بارگذاری SCON جهت انتخاب حالت دریافت

۲- بارگذاری TMOD برای حالت ۱ و ۳

۳- بارگذاری TIMER1 برای حالت ۱ و ۳

۴- انتقال داده در SBUF

۵- سرکشی پرچم ارسال در این قسمت RI را سرکشی کرده تا ارسال تمام شود بعد از اتمام

کامل ارسال RI به صورت سخت‌افزاری یک می‌شود.

۶- پاک کردن پرچم RI

۷- انتقال داده دریافتی از SBUF

:( )

برنامه‌ای بنویسید که اطلاعاتی را از درگاه سریال با نرخ 9600bps دریافت کند.

```
MAIN:
MOV SCON,#01010000B
MOV TMOD,#20H
MOV TH1,#-3 ;IF XTALL CHANG TO 11.0592 SET BAUD RATE -3=9600|-6=4800|-
12=2400|-24=1200 .....
SETB TR1
RESEVER:
LOOP:JNB RI,LOOP
MOV A,SBUF
CLR RI
MOV P1,A
JMP RESEVER
END
```

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

## LCD

تمایل به استفاده از نمایشگر کریستال مایع (LCD) به عنوان صفحه نمایش در پروژه‌های مختلف روز به روز افزایش می‌یابد. در جایی که 7SEGMENT نمی‌تواند کاراکترها را نمایش دهد و یا اینکه فضای اضافی برای نمایش تعداد رقم‌ها می‌گیرد از LCD استفاده می‌کنند.

## :LCD

پایه	نماد	I/O	توصیف
1	VSS	---	زمین
2	VCC	---	منبع تغذیه
3	VEE	---	منبع تغذیه کنترل درخشندگی
4	RS	I	انتخاب ثبات داده یا دستورالعمل
5	R/W	I	گذر گاه داده
6	E	I	گذر گاه داده
7	DB0	I/O	گذر گاه داده
8	DB1	I/O	گذر گاه داده
9	DB2	I/O	گذر گاه داده
10	DB3	I/O	گذر گاه داده
11	DB4	I/O	گذر گاه داده
12	DB5	I/O	گذر گاه داده
13	DB6	I/O	گذر گاه داده

## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

گذر گاه داده	I/O	DB7	14
زمین	----	VSS	15
با یک مقاومت 10 OHM به VCC	----	VCC	16

## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

### :RS

در داخل LCD دو ثبات وجود دارد و این پایه برای انتخاب این دو ثبات به ترتیب زیر عمل

می‌کند.

الف: اگر  $RS=0$  باشد ثبات دستورالعمل فرمان انتخاب می‌شود و باعث می‌شود فرمان‌هایی که در

جدول زیر آورده شده صادر گردد.

ب: اگر  $RS=1$  باشد ثبات داده انتخاب می‌گردد و به ما اجازه نوشتن روی LCD را می‌دهد.

### :R/W

این پایه به کاربر اجازه نوشتن اطلاعات در LCD و یا خواندن از آن را فراهم می‌کند. برای

نوشتن  $R/W=1$  است.

### :E

LCD از این پایه برای لچ کردن اطلاعات ارایه شده به پایه‌های داده‌اش استفاده می‌کند. وقتی

داده‌های اعمال شد، یک پالس بالا به پایین به این پایه اعمال می‌گردد. تا به این وسیله LCD

اطلاعات موجود در پایه‌های داده را لچ کند.

### :LCD

برای نوشتن برنامه‌ها بر روی LCD به نکات زیر توجه شود.

(۱) LCD برای عملیات درونی خود برای ثبات دستورالعمل و داده به مدت معینی زمان نیاز دارد، در

صورتی که ما اطلاعات جدیدی را در حین عملیات وارد کنیم LCD آن را نمی‌پذیرد.



## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

بنابراین به دو روش می‌توانیم از پایان عملیات مطمئن شویم.

الف: چک کردن بیت پرچم

ب: ایجاد یک تأخیر زمانی بین اطلاعات ورودی

۲) ابتدا یک زیر برنامه COMMAND و یک زیر برنامه DATA می‌نویسیم.

برای زیر برنامه DATA باید توجه کرد که پایه‌های کنترلی به صورت زیر مقداردهی می‌شوند.

الف:  $RS = 1$

ب:  $R/W = 0$

ج: به پایه E ابتدا یک پالس HIGH وبعد یک پالس LOW وارد می‌کنیم.

برای زیر برنامه DATA باید توجه کرد که پایه‌های کنترلی به صورت زیر مقداردهی می‌شوند.

الف:  $RS = 0$

ب:  $R/W = 0$

ج: وبه پایه E ابتدا یک پالس HIGH وبعد یک پالس LOW وارد می‌کنیم.

۳) برای اطمینان از عدم مشغول بودن LCD همان‌طور که گفته شد می‌توانیم از چک کردن

بیت پرچم استفاده کنیم. برای این کار باید توجه کرد که در حالت  $R/W = 1$  و  $RS = 1$  وقتی

که بیت پرچم (D7) برابر یک باشد LCD مشغول بوده و از عملیات درونی مراقبت می‌کند

و هیچ اطلاعات جدیدی را نمی‌پذیرد.

## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

وقتی  $D7 = 0$  باشد LCD برای دریافت اطلاعات جدید آماده است. برنامه چک کردن بیت

پرچم به قرار زیر است.

```
READY:  
SETB D7  
CLR RS  
SETB R/W  
BACK:  
CLR E  
SETB E  
JB D7,BACK  
RET
```

۴) برای استفاده از ثبات دستورالعمل باید توجه داشت که بعضی از COMMAND ها با

بعضی دیگر از COMMAND ها نمی‌توانند در یک برنامه اجرا شوند و باعث می‌شوند که برنامه

اجرا نشود. پس متذکر می‌شویم که حتی ترتیب بعضی از COMMAND نیز باید رعایت شود.

در جدول زیر فرمان‌ها و کدهای ثبات دستورالعمل آمده است.

کد	فرمان به ثبات دستورالعمل
1	پاک کردن صفحه نمایش
2	بازگشت به مکان اولیه
4	جابجایی مکان نما به چپ
6	جابجایی مکان نما به راست
5	جابجایی نمایش به راست
7	جابجایی نمایش به چپ
8	نمایش خاموش مکان نما خاموش
A	نمایش خاموش مکان نما روشن

## پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

نمایش روشن مکان نما خاموش	C
نمایش روشن مکان نما روشن	E
نمایش روشن مکان چشمک بزند	F
جابجایی محل مکان نما به چپ	10
جابجایی محل مکان نما به راست	14
کل صفحه نمایش به چپ جابجا شود	18
کل صفحه نمایش به راست جابجا شود	1C
مکان نما به آغاز خط دوم برود	C0
ساز مان دهی ۲ خط ماتریس ۷×۵	38

( ) :

برنامه‌ای بنویسید که کلمه NO بر روی LCD نمایش داده شود. (از ایجاد تأخیر زمانی برای

اطمینان از عدم مشغول بودن LCD استفاده کنید).

```
PORTLCD EQU P0
RS EQU P2.5
R/W EQU P2.6
EN EQU P2.7
ORG 00H
MOV A,#38H
CALL COMMAND
MOV A,#0E
CALL COMMAND
MOV A,#01H
CALL COMMAND
MOV A,#06H
CALL COMMAND
MOV A,#89H
CALL COMMAND
```

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

MOV A,#' N '

CALL DAT

MOV A,#' O '

CALL DAT

SJMP \$

COMMAND:

MOV PORTLCD,A

CALL DELAY

CLR RS

CLR R/W

SETB EN

CLR EN

RET

DAT:

MOV PORTLCD,A

CALL DELAY

SETB RS

CLR R/W

SETB EN

CLR EN

RET

DELAY:

MOV R0,#50

K1:MOV R1,#255

K2:DJNZ R1,K2

DJNZ R0,K1

RET

END

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

این برنامه از صفحه کلید یک عدد دو رقمی دریافت می‌کند و برنامه‌اش می‌دهد که مشخص کننده

تعداد و دور موتور است (موتور پله‌ای  $1/8$  درجه).

در ادامه برنامه عدد دورقمی دیگری را جهت تنظیم سرعت موتور از شما دریافت می‌نماید و آن

عدد را نمایش می‌دهد که در عمل همان میزان تأخیر در بین هر گام موتور است و بعد از آن موتور

را با مشخصا وارد شده به حرکت در می‌آورد.

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

BUSER	EQU	P2.0
ROW1	EQU	P2.7
ROW2	EQU	P2.6
ROW3	EQU	P2.5
ROW4	EQU	P2.4
COL1	EQU	P2.3
COL2	EQU	P2.2
COL3	EQU	P2.1
STEPPERPORT	EQU	P0
DELAYCOUNTER	EQU	19H
NUMBERLOAD	EQU	20H
TEMPKEYSCAN	EQU	21H
DAHGANKEYSCAN	EQU	22H
NUMBERDISPLAY	EQU	23H
SEGMENT1	EQU	24H
SEGMENT2	EQU	25H
TEMPDOR	EQU	26H
DORNUMBER	EQU	27H
SPEEDNUMBER	EQU	28H

```
ORG 0
JMP MAIN
ORG 1BH
INTERRUPTTIMER1:
MOV TH1,#HIGH(64000)
MOV TL1,#LOW(64000)
CLR TF1
MOV P1,SEGMENT1
CALL DELAYDISPLAY
MOV P1,SEGMENT2
CALL DELAYDISPLAY
MOV P1,#0FFH
RETI
```

```
MAIN:
MOV TMOD,#11H
MOV IE,#88H
MOV SP,#50H
MOV TH1,#HIGH(64000)
MOV TL1,#LOW(64000)
```

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

```
SETB TR1
MOV NUMBERDISPLAY,#0H
CALL DISPLAY
LOOPMAIN:
```

```
CALL KEYSKAN
MOV DORNUMBER,NUMBERLOAD
CALL DELAY
MOV NUMBERDISPLAY,#0
CALL DISPLAY
```

```
CALL KEYSKAN
MOV SPEEDNUMBER,NUMBERLOAD
SCANLEFTRIGHT:
CLR ROW4
JNB COL3,RIGHT
JNB COL1,LEFT
JMP SCANLEFTRIGHT
```

```
;+++++
+
;          -----RIGHT
```

ایجاد حرکت موتور به سمت راست

```
RIGHT:
MOV TEMPDOR,#50
MOV NUMBERDISPLAY,DORNUMBER
CALL DISPLAY
LOOPRIGHT:
MOV STEPPERPORT,#00001001B
CALL SPEED
MOV STEPPERPORT,#00001100B
CALL SPEED
MOV STEPPERPORT,#00000110B
CALL SPEED
MOV STEPPERPORT,#00000011B
CALL SPEED
DJNZ TEMPDOR,LOOPRIGHT
DJNZ DORNUMBER,RIGHT
MOV NUMBERDISPLAY,DORNUMBER
CALL DISPLAY
CLR BUSER
```

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

```
CALL DELAY
SETB BUSER
JMP LOOPMAIN
;+++++
+
;          -----LEFT
```

ایجاد حرکت موتور به سمت چپ

```
LEFT:
MOV TEMPDOR,#50
MOV NUMBERDISPLAY,DORNUMBER
CALL DISPLAY
LOOPLEFT:
MOV STEPPERPORT,#00000011B
CALL SPEED
MOV STEPPERPORT,#00000110B
CALL SPEED
MOV STEPPERPORT,#00001100B
CALL SPEED
MOV STEPPERPORT,#00001001B
CALL SPEED
DJNZ TEMPDOR,LOOPLEFT
DJNZ DORNUMBER,LEFT
MOV NUMBERDISPLAY,DORNUMBER
CALL DISPLAY
CLR BUSER
CALL DELAY
SETB BUSER
JMP LOOPMAIN
;
+++++
;          -----SPEED
```

این زیربرنامه تأخیر لازم را بین هر گام موتور ایجاد می‌نماید.

```
SPEED:
MOV R6,SPEEDNUMBER
LOOPSPD:
MOV TH0,#0FCH
MOV TL0,#18H
SETB TR0
LOOP1S:JNB TF0,LOOP1S
CLR TR0
CLR TF0
;          -----STOP TEST
```



# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

```
CLR ROW4
;JNB COL2,PAUSE
SETB ROW4
;
DJNZ R6,LOOPSPPEED
RET
```

```
; -----SCAN KEYBOARD -----
KEYSCAN:
SETB ROW4
CLR ROW1
JNB COL1,KEY1
JNB COL2,KEY2
JNB COL3,KEY3
SETB ROW1
```

```
CLR ROW2
JNB COL1,KEY4
JNB COL2,KEY5
JNB COL3,KEY6
SETB ROW2
```

```
CLR ROW3
JNB COL1,KEY7
JNB COL2,KEY8
JNB COL3,KEY9
SETB ROW3
```

```
CLR ROW4
JNB COL2,KEYZERO
SETB ROW4
JMP KEYSCAN
```

```
; -----FIND KEYBOARD PERESS -----
KEY1:
SETB ROW1
MOV TEMPKEYSCAN,#1
JMP KEYLOAD
```

```
KEY2:
SETB ROW1
MOV TEMPKEYSCAN,#2
JMP KEYLOAD
```

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

```
KEY3:  
SETB ROW1  
MOV TEMPKEYSCAN,#3  
JMP KEYLOAD
```

```
KEY4:  
SETB ROW2  
MOV TEMPKEYSCAN,#4  
JMP KEYLOAD
```

```
KEY5:  
SETB ROW2  
MOV TEMPKEYSCAN,#5  
JMP KEYLOAD
```

```
KEY6:  
SETB ROW2  
MOV TEMPKEYSCAN,#6  
JMP KEYLOAD
```

```
KEY7:  
SETB ROW3  
MOV TEMPKEYSCAN,#7  
JMP KEYLOAD
```

```
KEY8:  
SETB ROW3  
MOV TEMPKEYSCAN,#8  
JMP KEYLOAD
```

```
KEY9:  
SETB ROW3  
MOV TEMPKEYSCAN,#9  
JMP KEYLOAD
```

```
KEYZERO:  
SETB ROW4  
MOV TEMPKEYSCAN,#0  
JMP KEYLOAD
```

```
;  
----- MERG FIRST NUMBER & SECOND -----
```

ایجاد یک عدد دورقمی برای 7seg و محاسبات

```
KEYLOAD:  
JNB 07H,DAHGAN
```

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

```
MOV A,TEMPKEYSCAN
MOV B,DAHGANKEYSCAN
ADD A,B
MOV NUMBERLOAD,A
MOV NUMBERDISPLAY,A
CALL DISPLAY
CALL DELAY
CLR 07H
RET
```

```
DAHGAN:
MOV A,TEMPKEYSCAN
MOV B,#10
MUL AB
MOV DAHGANKEYSCAN,A
SETB 07H
MOV NUMBERDISPLAY,A
CALL DISPLAY
CALL DELAY
JMP KEYSCAN
```

```
DELAYDISPLAY:
MOV R7,#255
LOOPDISPLAY:
DJNZ R7,LOOPDISPLAY
RET
```

```
; -----SPELLET ONE NUMBER TO 2 DIGIT -----
DISPLAY:
MOV A,NUMBERDISPLAY
MOV B,#10
DIV AB
ORL B,#10H
ORL A,#20H
MOV SEGMENT1,B
MOV SEGMENT2,A
RET
```

```
; -----DELAY 50 MIL SECOND -----
DELAY:
MOV DELAYCOUNTER,#10
LDK:
MOV TH0,#3CH
```

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

```
MOV TL0,#0B0H
SETB TR0
LOOPDELAY:JNB TF0,LOOPDELAY
CLR TR0
CLR
TF0
DJN
Z
DEL
AYC
OUN
TER,
LDK
RET
END
```

شماتی  
ک برد  
آموزش  
ی

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

آدرس  
بایت

آدرس بیت

آدرس  
بایت

آدرس بیت

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

7F	همه منظوره RAM حافظه							
30								
2F	7F	7E	7D	7C	7B	7A	79	78
2E	77	76	75	74	73	72	71	70
2D	6F	6E	6D	6C	6B	6A	69	68
2C	67	66	65	64	63	62	61	60
2B	5F	5E	5D	5C	5B	5A	59	58
2A	57	56	55	54	53	52	51	50
29	4F	4E	4D	4C	4B	4A	49	48
28	47	46	45	44	43	42	41	40
27	3F	3E	3D	3C	3B	3A	39	38
26	37	36	35	34	33	32	31	30
25	2F	2E	2D	2C	2B	2A	29	28
24	27	26	25	24	23	22	21	20
23	1F	1E	1D	1C	1B	1A	19	18
22	17	16	15	14	13	12	11	10
21	0F	0E	0D	0C	0B	0A	09	08
20	07	06	05	04	03	02	01	00
1F	بانک ۳							
18								
17	بانک ۲							
10								
0F	بانک ۱							
08								
07	R0-R7 بانک قراردادی ثابت برای							
00								

FF									
F0	F7	F6	F5	F4	F3	F2	F1	F0	B
E0	E7	E6	E5	E4	E3	E2	E1	E0	ACC
D0	D7	D6	D5	D4	D3	D2	D1	D0	PSW
B8	----	----	----	BC	BB	BA	B9	B8	IP
B0	B7	B6	B5	B4	B3	B2	B1	B0	P3
A8	AF	----	----	AC	AB	AA	A9	A8	IE
A0	A7	A6	A5	A4	A3	A2	A1	A0	P2
98	بیت آدرس پذیر نیست								SBUF
98	9F	9E	9D	9C	9B	9A	99	98	SCON
90	97	96	95	94	93	92	91	90	P1
8D	بیت آدرس پذیر نیست								TH1
8C	بیت آدرس پذیر نیست								TH0
8B	بیت آدرس پذیر نیست								TL1
8A	بیت آدرس پذیر نیست								TL0
89	بیت آدرس پذیر نیست								TMOD
88	8F	8E	8D	8C	8B	8A	89	88	TCON
87	بیت آدرس پذیر نیست								PCON
83	بیت آدرس پذیر نیست								DPH
82	بیت آدرس پذیر نیست								DPL
81	بیت آدرس پذیر نیست								SP
80	87	86	85	84	83	82	81	80	P0

ثباتهای ویژه

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

خلاصه حافظه درون تراشه ۸۰۵۱

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

## آدرسهای بی‌تی از قبل تعیین شده در 89C51

مفهوم	آدرس بیت	موقعیت بیت	نماد
پرچم نقلی	D7H	PSW.7	CY
پرچم نقلی کمکی	D6H	PSW.6	AC
پرچم 0	D5H	PSW.5	F0
بیت انتخاب رجیستر بانک 1	D4H	PSW.4	RS1
بیت انتخاب رجیستر بانک 0	D3H	PSW.3	RS0
پرچم سرریز	D2H	PSW.2	OV
پرچم توازن	D0H	PSW.0	P
پرچم سرریز تایمر 1	8FH	TCON.7	TF1
بیت کنترل راه‌اندازی تایمر 1	8EH	TCON.6	TR1
پرچم سرریز تایمر 0	8DH	TCON.5	TF0
بیت کنترل راه‌اندازی تایمر 0	8CH	TCON.4	TR0
پرچم لبه وقفه 1	8BH	TCON.3	IE1
بیت کنترل نوع وقفه 1	8AH	TCON.2	IT1
پرچم لبه وقفه 0	89H	TCON.1	IE0
بیت کنترل نوع وقفه 0	88H	TCON.0	IT0
بیت کنترل حالت سریال 0	9FH	SCON.7	SM0
بیت کنترل حالت سریال 1	9EH	SCON.6	SM1
بیت کنترل حالت سریال 2	9DH	SCON.5	SM2
تواناساز دریافت کننده	9CH	SCON.4	REN



# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

بیت 8 ارسال	9BH	SCON.3	TB8
بیت 8 دریافت	9AH	SCON.2	RB8

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

ادامه از صفحه قبل

مفهوم	آدرس بیت	موقعیت بیت	نماد
پرچم وقفه ارسال	99H	SCON.1	TI
پرچم وقفه دریافت	98H	SCON.0	RI
تواناساز همه وقفه ها	AFH	IE.7	EA
تواناساز وقفه درگاه سریال	ACH	IE.4	ES
تواناساز وقفه تایمر 1	ABH	IE.3	ETI
تواناساز وقفه خارجی 1	AAH	IE.2	EX1
تواناساز وقفه تایمر 0	A9H	IE.1	ET0
تواناساز وقفه خارجی 0	A8H	IE.0	EX0
خواندن داده از حافظه خارجی	B7H	P3.7	RD
نوشتن داده در حافظه خارجی	B6H	P3.6	WR
پرچم خارجی تایمر / شمارنده 1	B5H	P3.5	T1
پرچم خارجی تایمر / شمارنده 0	B4H	P3.4	T0
پایه ورودی وقفه 1	B3H	P3.3	INT1
پایه ورودی وقفه 0	B2H	P3.2	INT0
پایه ارسال درگاه سریال	B1H	P3.1	TXD
پایه دریافت درگاه سریع	B0H	P3.0	RXD
حق تقدم وقفه درگاه سریال	BCH	IP.4	PS
حق تقدم وقفه تایمر 1	BBH	IP.3	PT1
حق تقدم وقفه خارجی 1	BAH	IP.2	PX1

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

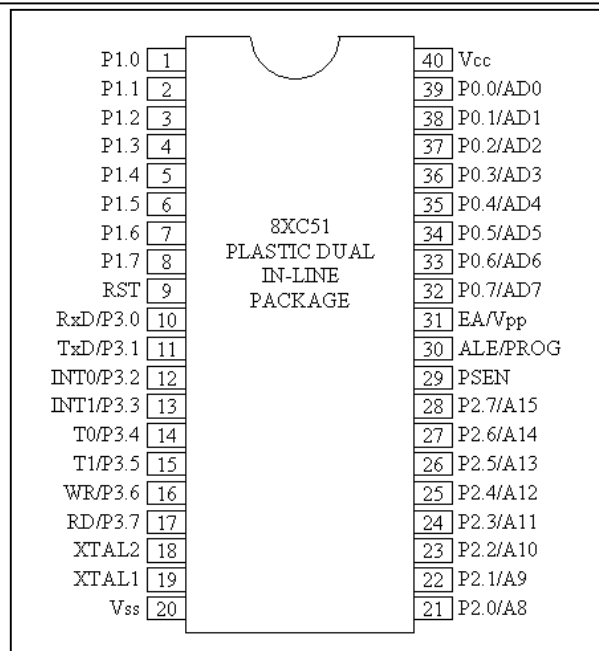
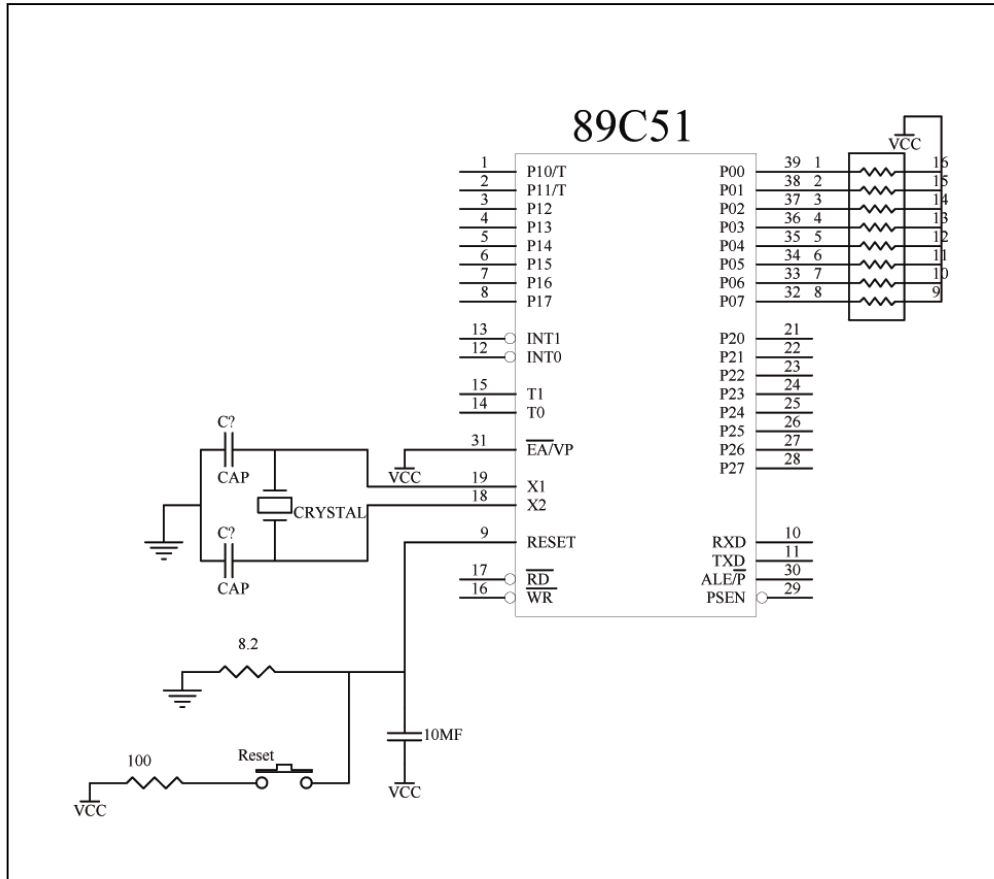
حق تقدم وقفه تایمر 0	B9H	IP.1	PT0
حق تقدم وقفه خارجی 0	B8H	IP.0	PX0

# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک



# پیشرو نوآوران کاوش

اتوماسیون صنعتی

طراحی، تولید، اجرای پروژه‌های الکترونیکی

آموزش دوره‌های تخصصی الکترونیک

---

نمایی از پایه‌های میکروکنترلر 89C51