

آشنایی با تایمر / کانتر و نحوه استفاده از آنها

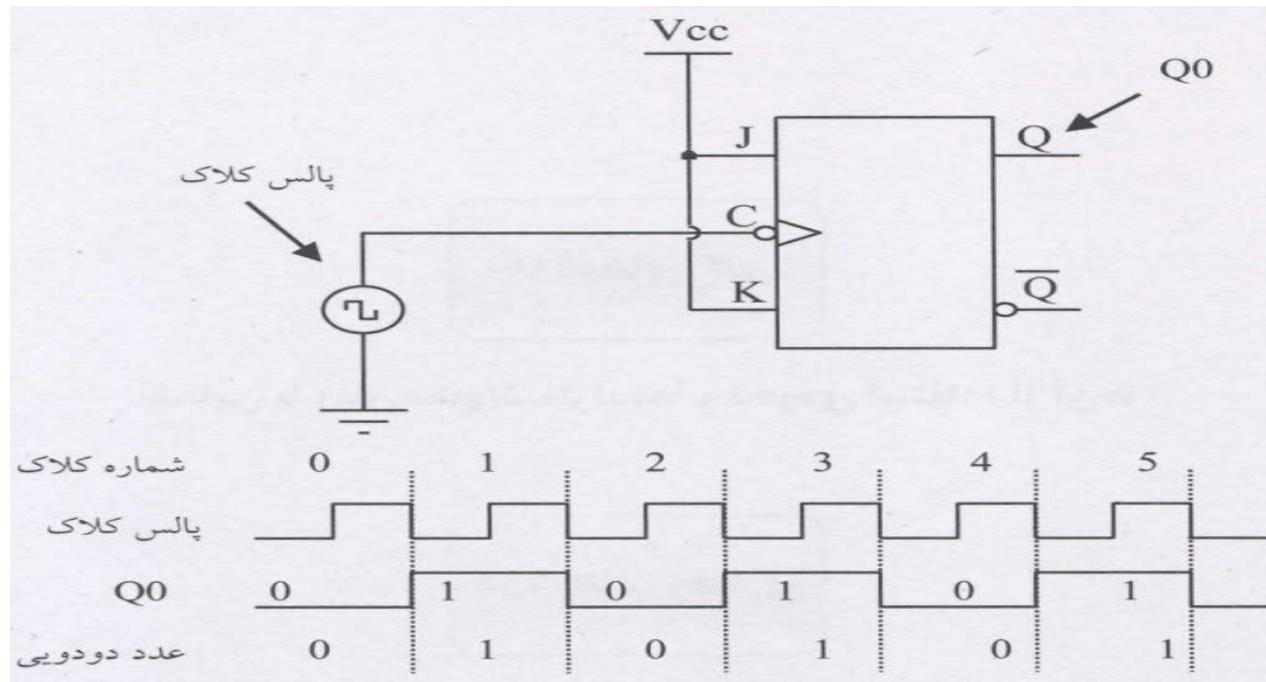
دانشگاه علم و صنعت - دانشکده مهندسی کامپیوتر -

بهمن ۱۴۰۰

هاشم مشحون

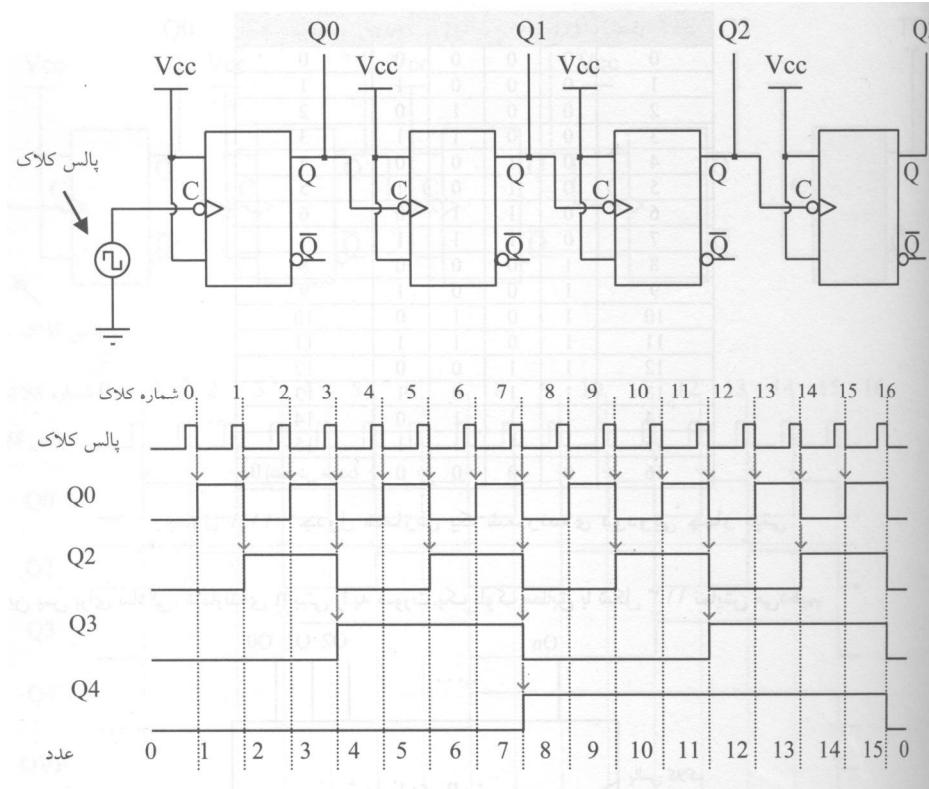
آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

▶ آشنایی با شمارنده های دیجیتال



▶ در فلیپ فلاب فوق که از نوع K-J است اگر هر دو ورودی را به یک منطقی وصل کنیم، لبه های پالس کلاک ورودی باعث تغییر سطح خروجی خواهد شد.

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر



▶ با پشت سرهم بستن چهار فلیپ فلاب طبق شکل فوق میتوان یک شمارنده باینری را ساخت که از صفر تا پانزده را میشمارد.

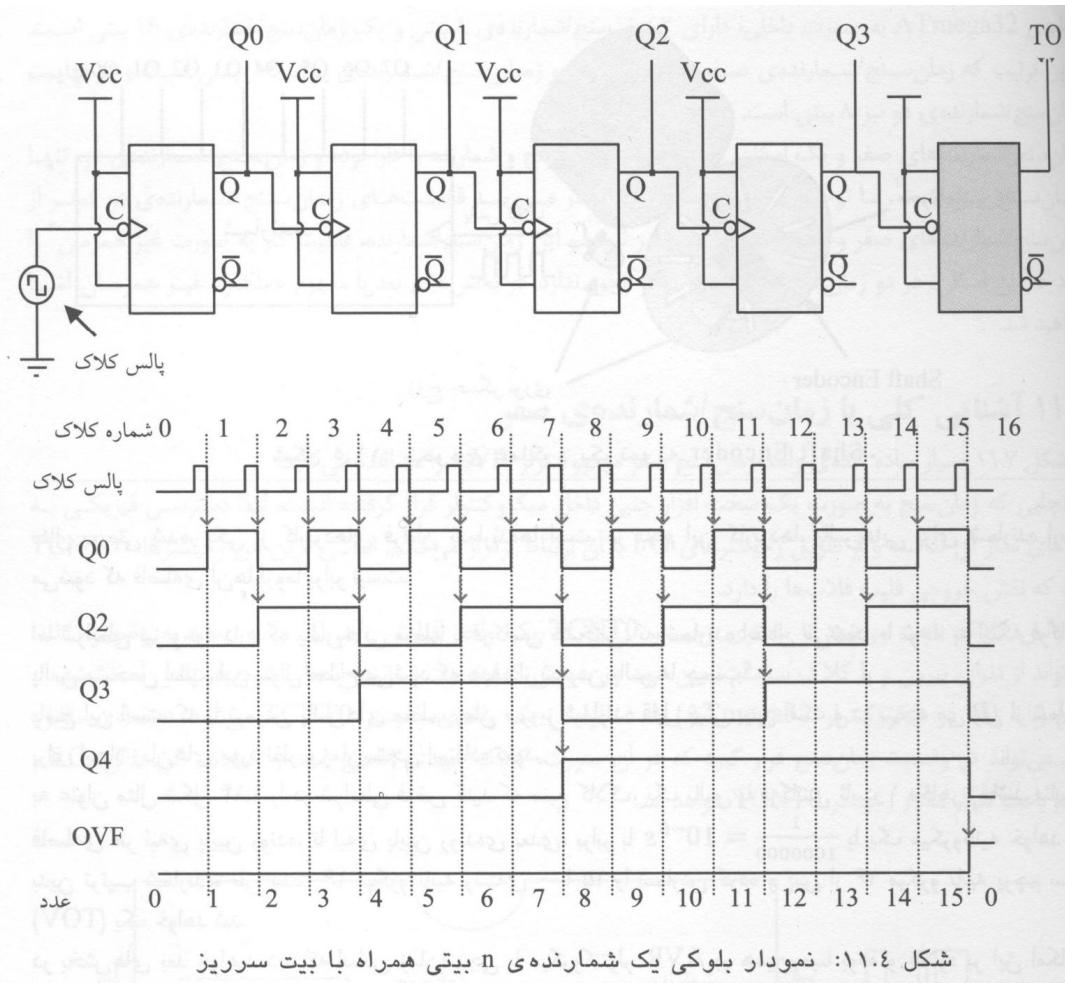
آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

شماره کلاچ	Q3	Q2	Q1	Q0	عدد دودویی
0	0	0	0	0	0
1	0	0	0	1	1
2	0	0	1	0	2
3	0	0	1	1	3
4	0	1	0	0	4
5	0	1	0	1	5
6	0	1	1	0	6
7	0	1	1	1	7
8	1	0	0	0	8
9	1	0	0	1	9
10	1	0	1	0	10
11	1	0	1	1	11
12	1	1	0	0	12
13	1	1	0	1	13
14	1	1	1	0	14
15	1	1	1	1	15
16	0	0	0	0	0 (شمارش مجدد)

در نهایت آنچه در اثر پالس کلاک ورودی، در خروجی فلیپ فلاپها ظاهر میشود مطابق با جدول روبرو است.

- ▶ به همین ترتیب برای ساخت یک شمارنده ۸ بیتی به ۸ فلیپ فلاپ و برای شمارنده n بیتی، n فلیپ فلاپ نوع K-L نیاز خواهد بود.

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

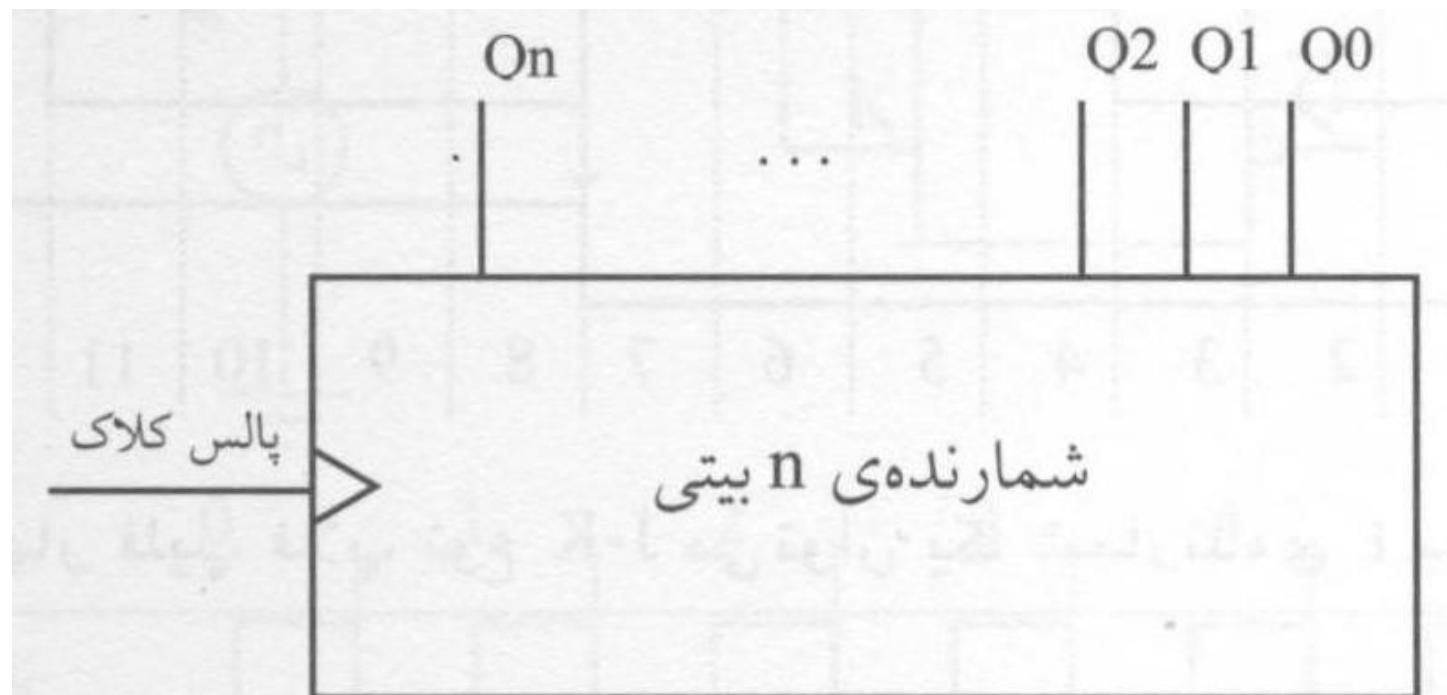


آشنایی با مفهوم سرریز شمارنده

لحظه‌ای که زمان سنج به انتهای رشته‌ی شمارش میرسد و با صفر بازگذاری می‌شود سرریز یا **overflow** نام دارد. چرا که در این لحظه خروجی تمام فلیپ‌فلاپها یک شده و شمارنده‌ی ظرفیت بیشتری ندارد بنابراین با کلاک بعدی خروجی آن صفر می‌شود.

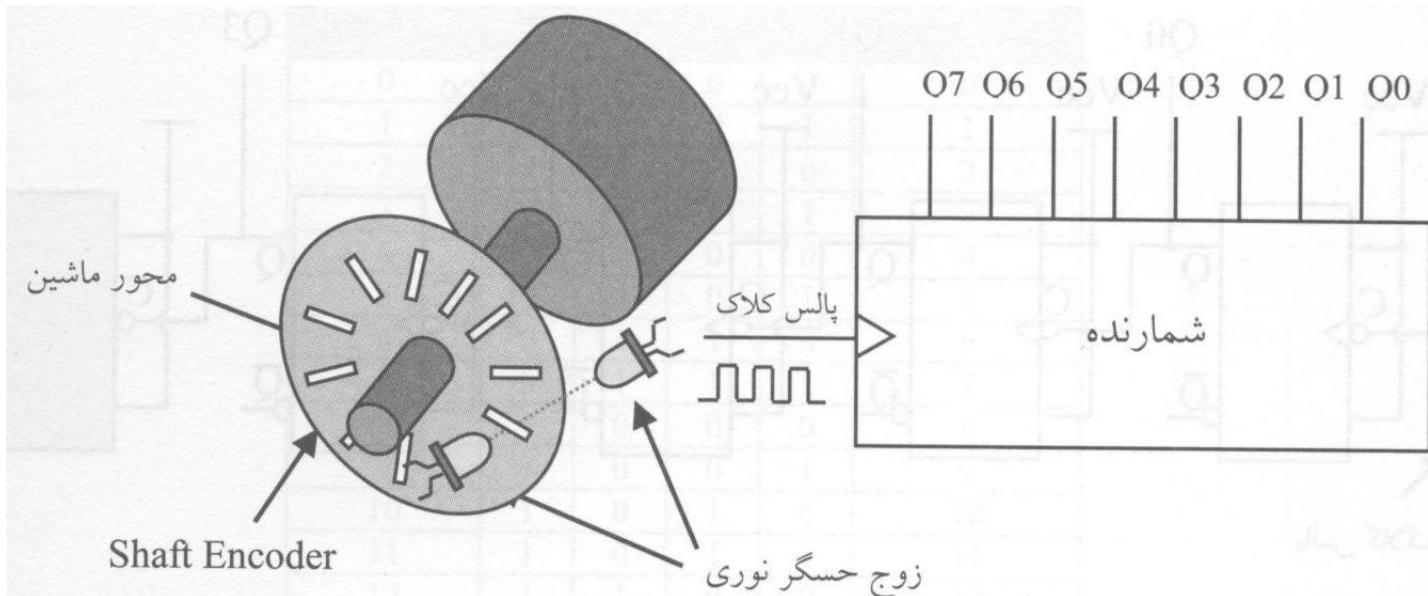
آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

از این پس برای سادگی، شمارنده n بیتی را به صورت یک بلوک مطابق با شکل زیر نمایش میدهیم.



آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

تفاوت تایمر (زمان سنج) و کانتر (شمارنده)
اکنون میدانیم که پالس کلاک، منشاء تغییر در خروجی فلیپ فلاپها و در نتیجه پیشروی در رشته‌ی شمارش است.
این پالس از چه طریق تامین میشود؟



با دریافت هر سیگنال نوری توسط حسگر گیرنده، یک پالس برای شمارنده ارسال میشود و مقدار آن را یک واحد افزایش میدهد. بدین ترتیب شمارنده مقدار حرکت چرخشی محور ماشین را ثبت کرده و اندازه جابجایی آن را به شکل دیجیتالی بیان میکند.

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

تفاوت تایمر (زمان سنج) و کانتر (شمارنده)

اما شرایطی وجود دارد که پالس‌هایی منظم و با فرکانس مشخص، به شمارنده اعمال می‌شود. با توجه به اینکه فرکانس پالس مشخص است، این سوال مطرح می‌شود که هدف از شمردن پالس‌ها چیست؟

پاسخ: با شمردن پالس‌های منظم، زمان سرریز شمارنده قابل پیش‌بینی است و در نتیجه می‌توان از شمارنده برای ایجاد زمانهای مورد نظر و زمان سنجی استفاده نمود.

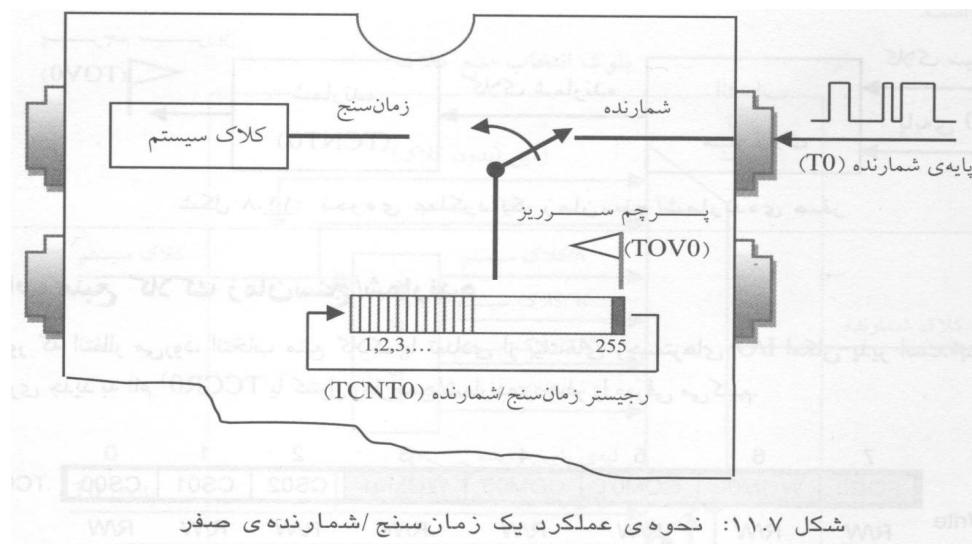
به عنوان مثال فرض کنید که منبع کلاک یک پالس با فرکانس ثابت یک مگاهرتز باشد. بنابراین فاصله‌ی هر لبه پایین رونده تا لبه‌ی پایین رونده بعدی، برابر با یک میکروثانیه خواهد بود. بدین ترتیب شمارنده طی مدت ۱۶ میکروثانیه رشته‌ی صفر تا پانزده را شمارنده کرده و پس از شانزده میکروثانیه پرچم (TOV) یک خواهد شد.

بعدا خواهید دید که اساس زمان سنجی با میکروکنترلر AVR بر همین مبنا بوده و علاوه بر این امکاناتی پیشترفتنه نیز در اختیار کاربر قرار می‌گیرد.

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

آشنایی کلی با تایمر/کانتر صفر

در شکل زیر شکل بسیار ساده واحد زمان سنج صفر میکروکنترلر AVR را مشاهده میکنید.



شکل ۱۱.۷: نحوه عملکرد یک زمان سنج/شمارنده صفر

از آنجا که تایمر به صورت یک سخت افزار جنبی داخل میکروکنترلر قرار گرفته است، لذا دسترسی فیزیکی به پایه های مدار آن نداریم و باید از طریق ثباتهای **I/O** با آن ارتباط برقرار کنیم. اساسی ترین این ثباتها **TCNT0** است که نقش خروجی فلیپ فلاپها را دارد.

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

آشنایی کلی با تایمر/کانتر صفر

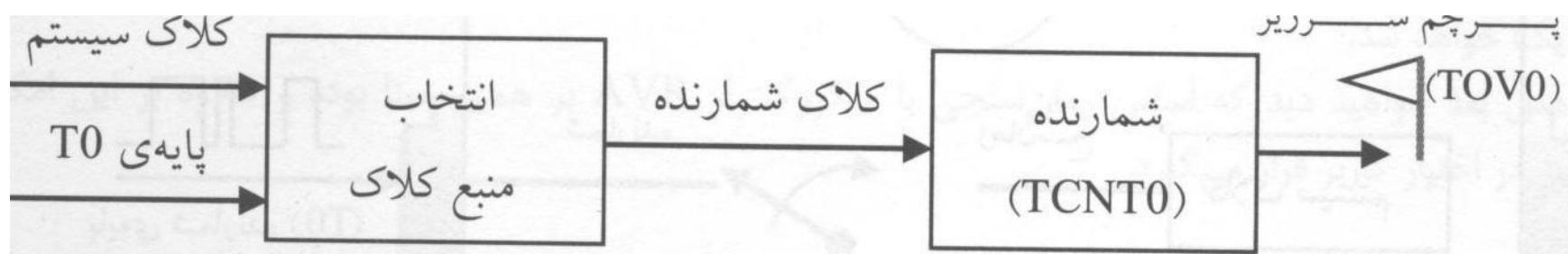
هر پالس کلاک باعث میشود تا محتوى ثبات TCNT0 یک واحد افزایش یابد این پالس بسته به وضعیت کلید، میتواند از دنیای بیرون و یا کلاک سیستم تامین شود. در صورتی که وضعیت شمارنده انتخاب شود. هر پالس اعمال شده از خارج (پایه T0 در میکروکنترلر) باعث افزایش یک واحدی ثبات TCNT0 میشود. در نقطه‌ی مقابل کلید میتواند در وضعیت تایمر قرار گیرد که در آن صورت، کلاک سیستم (تامین شده از اسیلاتور کریستالی، RC و غیره) باعث تغییر مقدار ثبات TCNT0 خواهد شد.

از آنجا که تایمر صفر، هشت بیتی است، ثبات TCNT0 میتواند مقادیر صفر تا ۲۵۵ را اختیار کند. در ابتدا مقدار این ثبات صفر است و با عمل پالس کلاک افزایش پیدا میکند. وقتی مقدار این ثبات ۲۵۵ باشد با اعمال کلاک بعدی، صفر میشود و مجدداً شمارش را از صفر شروع مینماید. و همزمان فلگ سرریز تایمر (TOV0) یک میشود.

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

آشنایی دقیق‌تر با تایمر/کانتر صفر

برای ایجاد زمینه‌ای جهت بررسی سخت افزار تایمر، اجزاء معرفی شده قبلی را در نمودار بلوکی زیر ملاحظه نمایید.



- تایمر صفر شامل یک ثبات شمارنده به نام TCNT0 است که نقش خروجی فلیپ فلامپها را دارد.
- این شمارنده با رسیدن به انتهای رشته شمارش (۲۵۵)، سرریز شده و مجدداً از صفر می‌شمارد، در این لحظه فلگ TOV0 یک می‌شود.
- ثبات TCNT0 برای شمارش نیاز با پالس کلاک دارد. این پالس می‌تواند از دو منبع پایه T0 یا کلاک سیستم تأمین شود که در وضعیت اول، شمارنده یا کانتر نام دارد و در وضعیت دوم تایمر یا زمان سنج نامیده می‌شود.

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

انتخاب منبع کلاک تایمر/کانتر

همانطور که انتظار می‌رود، انتخاب منبع کلاک با تعدادی از بیت‌های ثبات‌آمکان پذیر است. بدین منظور ثبات جدیدی به نام TCCR0 یا کنترل تایمر/کانتر را معرفی می‌کنیم:

Bit	7	6	5	4	3	2	1	0	TCCR0
Read/Write	FOC0	WGM00	COM01	COM00	WGM01	CS02	CS01	CS00	
Initial Value	R/W 0								

بیت‌های [2:0] TCCR0 به نامهای CS0[2:0] وظیفه‌ی انتخاب کلاک را بر عهده دارند که عملکرد این بیت‌ها مطابق با جدول اسلاید بعدی است.

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

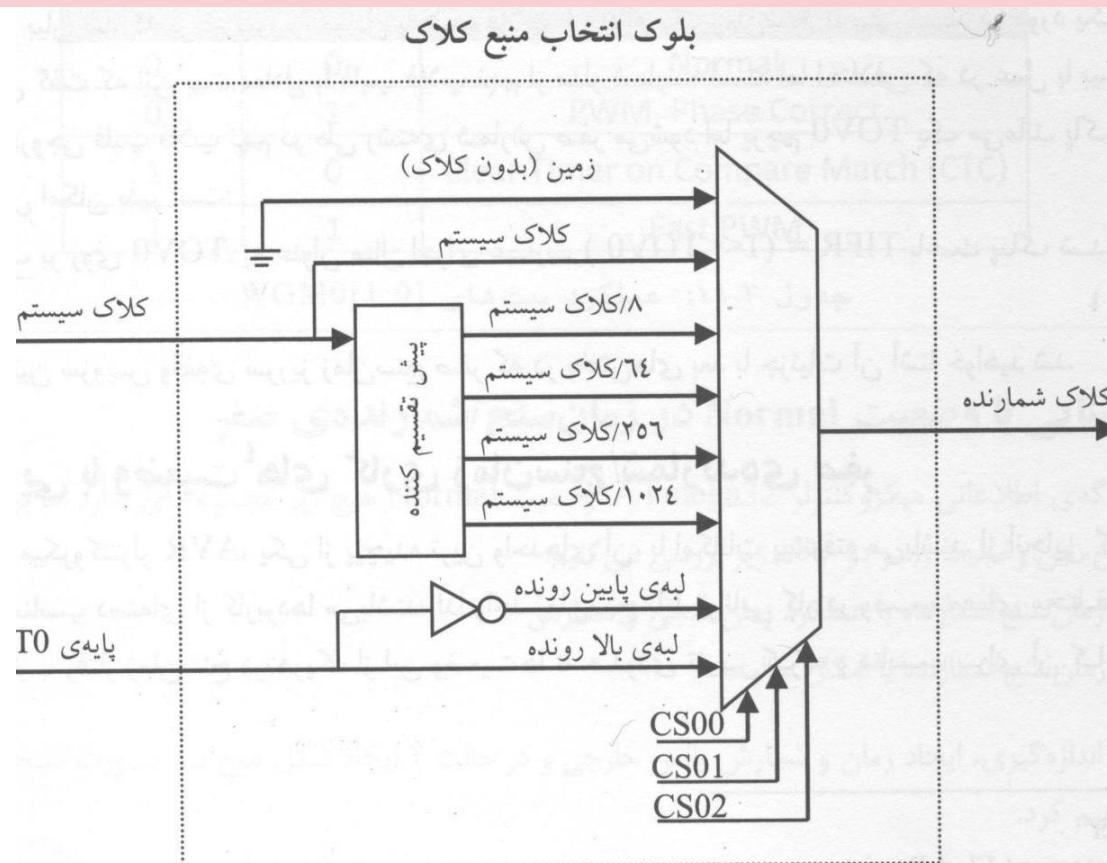
انتخاب منبع کلاک تایمر/ کانتر

CS02	CS01	CS00	منبع کلاک
0	0	0	بدون کلاک (متوقف)
0	0	1	کلاک سیستم (بدون تقسیم)
0	1	0	کلاک سیستم / ۸
0	1	1	کلاک سیستم / ۶۴
1	0	0	کلاک سیستم / ۲۵۶
1	0	1	کلاک سیستم / ۱۰۲۴
1	1	0	لبهی پایین روندهی پالس خارجی (T0)
1	1	1	لبهی بالا روندهی پالس خارجی (T0)

مطابق با جدول فوق اگر تمام بیتهاي CS0[2:0] برابر با صفر باشند، کلاکی به ثبات TCNT0 اعمال نمیشود و شمارنده غیرفعال است. حال اگر مقدار 001 در این سه بیت قرار گیرد، کلاک سیستم به طور مستقیم به شمارنده اعمال خواهد شد.

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

انتخاب منبع کلاک تایمر/کانتر



آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

انتخاب منبع کلاک تایمر/ کانتر

مطابق با جدول اگر تمام بیتهای CS0[2:0] برابر با صفر باشند، کلاکی به ثبات TCNT0 اعمال نمیشود و شمارنده غیرفعال است. حال اگر مقدار 001 در این سه بیت قرار گیرد، کلاک سیستم به طور مستقیم به شمارنده اعمال خواهد شد.

از آنجا که معمولاً کلاک سیستم دارای فرکانس بالایی است، لذا اعمال این کلاک به شمارنده‌ی ۸ بیتی باعث سرریز شدن سریع آن میشود. بنابراین بهتر است تا این کلاک توسط پیش تقسیم کننده بر عددی تقسیم شود. پیش تقسیم کننده میتواند کلاک ورودی را بر اعداد ۸، ۶۴، ۲۵۶ و ۱۰۲۴ تقسیم نماید. برای این منظور باید بیتهای CS0[2:0] را بارگذاری نمود.

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

پرچم سرریز تایمر/کانتر

فلگ TOV0 بیت صفر از ثبات TIFR است.

Bit	7	6	5	4	3	2	1	0	TIFR
Read/Write	OCF2	TOV2	ICF1	OCF1A	OCF1B	TOV1	OCF0	TOV0	
Initial Value	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

قبله بیان شد که فلگ سرریز، معادل با فلیپ فلاپ $n+1$ بیتی است. در مورد یک تایмер ۸ بیتی میتوان گفت که این بیت معادل با فلیپ فلاپ نهم از مدار شمارنده است. اما اختلافی که در عمل با بیت نهم دارد این است که خروجی فلیپ فلاپ در طی رشته‌ی شمارش صفر میشود اما فلگ TOV0 یک میماند. پاک کردن این بیت به دو روش امکان پذیر است:

۲- اجرای روتین سرویس وقفه‌ی سرریز تایمر صفر

۱- نوشتن یک بر روی TOV0

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

آشنایی با وضعیتهای کاری تایمر/کانتر صفر

تایمر/کانتر صفر میکروکنترلر AVR دارای چهار وضعیت کاری است:

1- Normal

2- CTC (Clear Timer on Compare Match)

3- Fast PWM

4- Phase correct PWM

باید بدانید که رفتار تایمر کانتر در وضعیتهای ۱ و ۲ مناسب برای شمارش پالس‌های خارجی، اندازه گیری زمان، ایجاد زمانهای مورد نظر و ایجاد موج مربعی است و وضعیتهای ۲ و ۳ مناسب تولید موج PWM هستند.

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

آشنایی با وضعیتهای کاری تایمر/کانتر صفر

انتخاب یکی از این چهار وضعیت کاری، با بارگذاری عدد مناسب در بیت‌های **WGM0[1:0]** امکان پذیر است.

Bit	7	6	5	4	3	2	1	0	TCCR0
	FOC0	WGM00	COM01	COM00	WGM01	CS02	CS01	CS00	
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

نحوهٔ انتخاب وضعیت مورد نظر با این دو بیت در جدول زیر آمده است:

WGM01	WGM00	وضعیت زمان‌سنج
0	0	Normal
0	1	PWM, Phase Correct
1	0	Clear Timer on Compare Match (CTC)
1	1	Fast PWM

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

آشنایی با وضعیت نرمال در تایمر/کانتر صفر

این موضوع را در دو حالت زیر بررسی میکنیم:

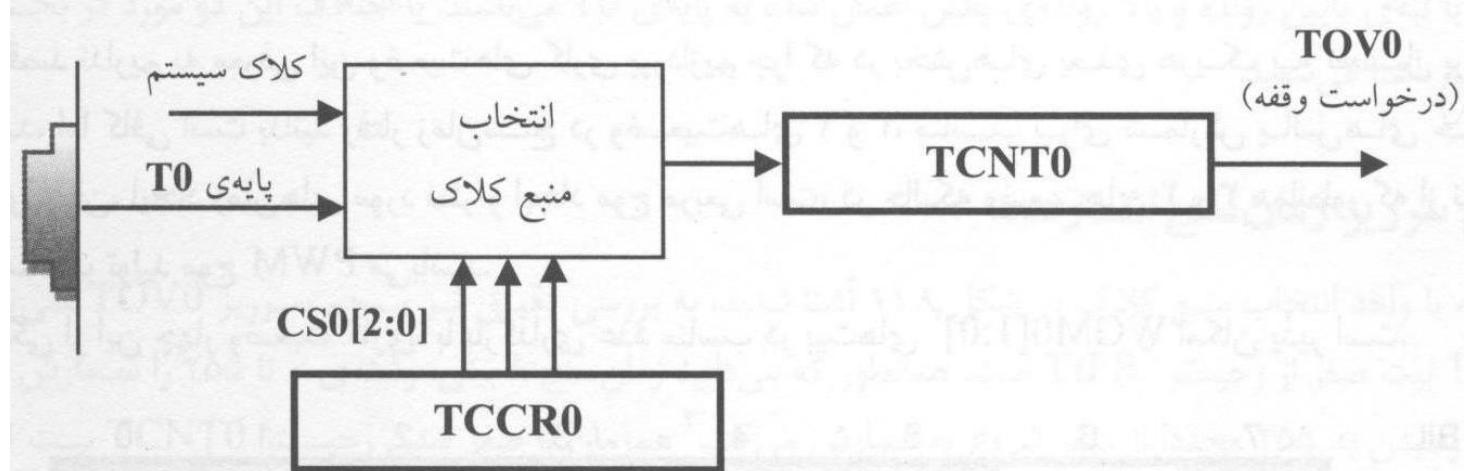
- ۱- تایмер/کانتر با عملکرد زمان سنجی و شمارش
- ۲- تایмер/کانتر با امکان مقایسه

در حالت یک، ایجاد زمان و شمارش پالس خارجی و در حالت دو ایجاد شکل موج به صورت سخت افزاری را بررسی خواهیم کرد.

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

تایمر/کانتر با عملکرد زمان سنجی و شمارش در وضعیت نرمال

نمودار بلوکی تایمر در این حالت مطابق با شکل زیر است. میدانید که شمارش، در ثبات TCNT0 انجام شده و با سرریز شدن آن فلگ TOV0، یک میشود. در این بخش خواهیم دید که در این لحظه میتوانند توسط تایмер، وقفه درخواست شود.



آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

تایمر/کانتر با عملکرد زمان سنجی و شمارش در وضعیت نرمال

مدت زمانی که لازم است سپری شود تا تایمر/کانتر سرریز شود، بستگی به پیش تقسیم کننده دارد. جدول زیر مقادیر مختلف پیش تقسیم کننده و زمان سرریز را در هر یک از حالات نشان میدهد. در این جدول کلاک سیستم ۸ مگاهرتز فرض شده است.

زمان سنج/شمارنده‌ی صفر (کلاک سیستم: ۸ MHz)			
مقدار پیش تقسیم کننده	کلاک شمارنده (MHz)	هر پالس شمارنده (μs)	زمان سرریز شمارنده (μs)
۱	۸	۰.۱۲۵	۳۲
۸	۱	۱	۲۵۶
۶۴	۰.۱۲۵	۸	۲۰۴۸
۲۵۶	۰.۰۳۱۲۵	۳۲	۸۱۹۲
۱۰۲۴	۰.۰۰۷۸۱۲۵	۱۲۸	۳۲۰۰۰

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

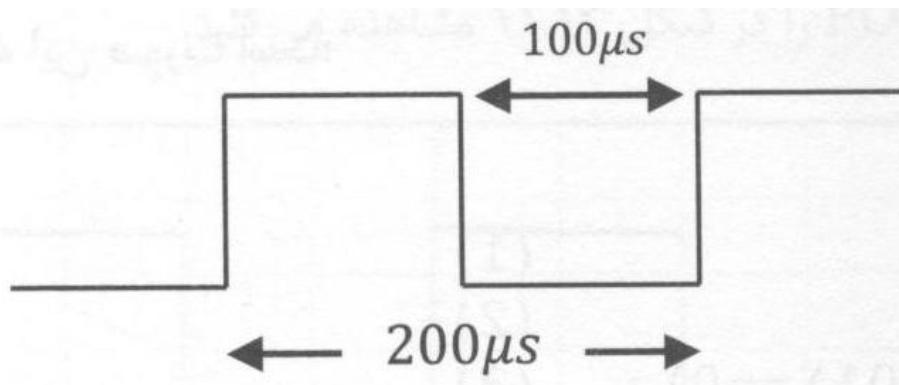
تایمر/کانتر با عملکرد زمان سنجی و شمارش در وضعیت نرمال

مثال : با فرض اینکه منبع کلاک میکروکنترلر ۸ مگاهرتز است، پالس مربعی با فرکانس ۵ کیلوهرتز بر روی پایه PD0 ایجاد نمایید.

پیش از هر چیز باید زمان قطع و وصل موج مورد نظر را محاسبه نماییم:

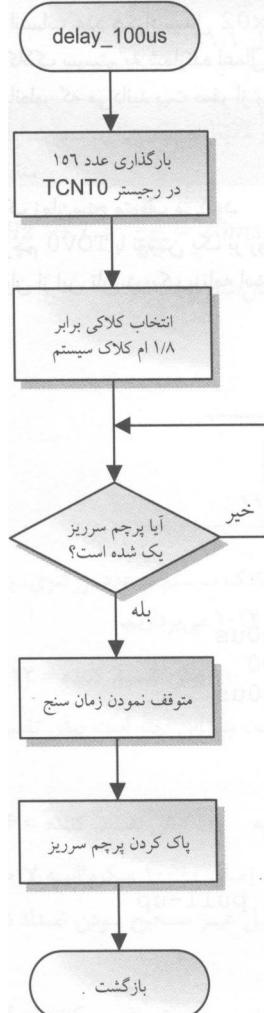
$$f = 5\text{KHz} \rightarrow T = \frac{1}{5000} = 200\mu\text{s}$$

بنابراین مطابق شکل زیر لازم است تا مدت زمان ۱۰۰ میکروثانیه توسط تایмер ایجاد شود



آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

تایمر/کانتر با عملکرد زمان سنجی و شمارش در وضعیت نرمال



در صورتی که کلاک سیستم، توسط پیش تقسیم کننده بر عدد ۸ تقسیم شود، پالسی با فرکانس یک مگاهرتز به TCNT0 اعمال خواهد شد، در نتیجه فاصله ی هر تیک کلاک برابر با یک میکروثانیه است. بدین ترتیب پس از ۲۵۶ میکروثانیه فلگ TOV0 یک شده و برنامه را از بروز سرریز آگاه میکند.

اما همانطور که عنوان شد برای ایجاد شکل موج مطلوب، زمانی برابر با ۱۰۰ میکروثانیه نیاز است، راه حل این مشکل ساده است: TCNT0 یک ثبات قابل خواندن و نوشتן است، پس با بارگذاری عدد ۱۵۶ در این ثبات، رشته شمارش تایмер محدود به بازه ۱۵۶ تا ۲۵۵ شده و با صفر شدن تایمر، فلگ TOV0 یک میشود.

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

تایمر/کانتر با عملکرد زمان سنجی و شمارش در وضعیت نرمال

در ابتدا عدد ۱۵۶ در ثبات TCNT0 بارگذاری شده و با انتخاب پیش تقسیم کننده ۸، زمان سنج شروع به شمارش کرده و پس از آن برنامه منتظر میماند تا تایмер سرریز شود. با یک شدن فلگ TOV0 زمان مورد نظر ایجاد شده است پس میتوان زمان سنج را متوقف نمود.

```
void delay_100us()
{
    TCNT0 = 156;
    TCCR0 = 0x02;0b00000010
    while((TIFR & 0x01) == 0);
    TCCR0 = 0;
    TIFR |= 0x01;
}
```

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

تایمر/کانتر با عملکرد زمان سنجی و شمارش در وضعیت نرمال

اکنون میتوان از این تابع در یک برنامه استفاده کرد و شکل موج مورد نظر مسئله را ایجاد کرد.

```
#include <mega32.h>
void delay_100us();
void init_io();

Void main(void)
{
    init_io();
    while (1)
    {
        PORTD |= (1<<PD0);
        delay_100us();
        PORTD &= ~ (1<<PD0);
        dela_100us();
    }
}
```

```
void init io()
{
    PORTD = 0x00;
    DDRD = 0x01;
}

void delay_100us()
{
    TCNT = 156;
    TCCR0 = 0x02;
    while((TIFR     & 0x01)
          ==0);
    TCCR0 = 0;
    TIFR |= 0x01;
}
```

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

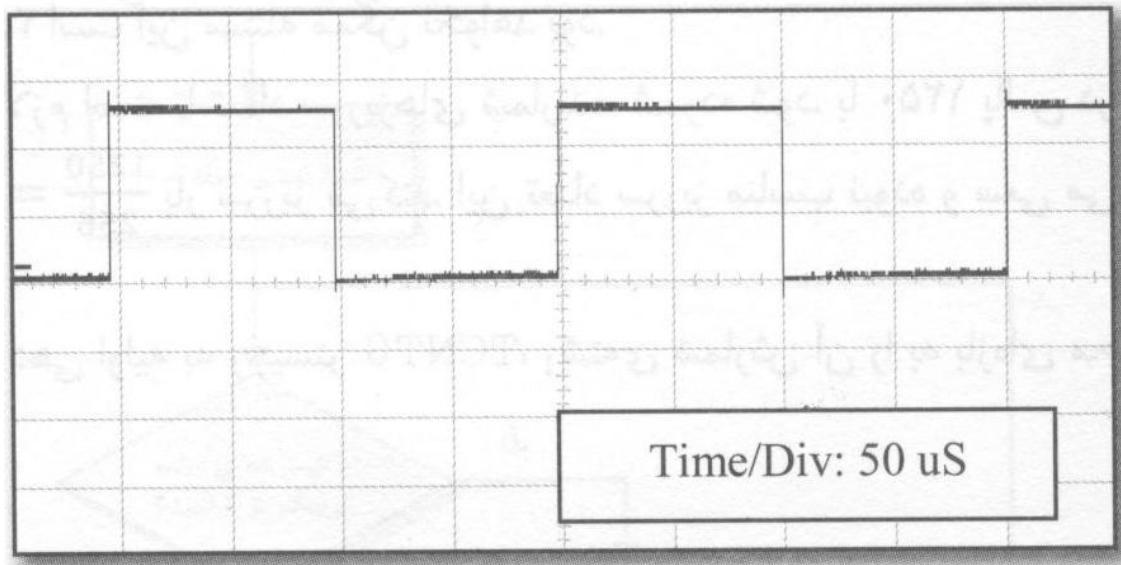
تایمر/کانتر با عملکرد زمان سنجی و شمارش در وضعیت نرمال

```
1 .include "m32def.inc"                                فایل منبع اسambilی این برنامه
2
3     LDI    R16,HIGH(RAMEND)   19
4     OUT    SPH,R16           20  DELAY_100:
5     LDI    R16,LOW(RAMEND)   21      LDI    R16,156
6     OUT    SPL,R16           22      OUT   TCNT0,R16
7
8     LDI    R16,0b00000001   23      LDI    R16,0b00000010
9     OUT    DDRD,R16          24      OUT   TCCR0,R16
10
11 BEGIN:  LDI    R16,01       25  S:      IN    R16,TIFR
12     OUT    PORTD,R16        26      ANDI  R16,0b00000001
13     CALL   DELAY_100       27      BREQ  S
14     LDI    R16,00       28      LDI    R16,0
15     OUT    PORTD,R16        29      OUT   TCCR0,R16
16     CALL   DELAY_100       30      LDI    R16,01
17     RJMP  BEGIN          31      OUT   TIFR,R16
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
```

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

تایمر/کانتر با عملکرد زمان سنجی و شمارش در وضعیت نرمال

شکل موج ایجاد شده روی پایه PD0 در شکل زیر دیده میشود:



آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

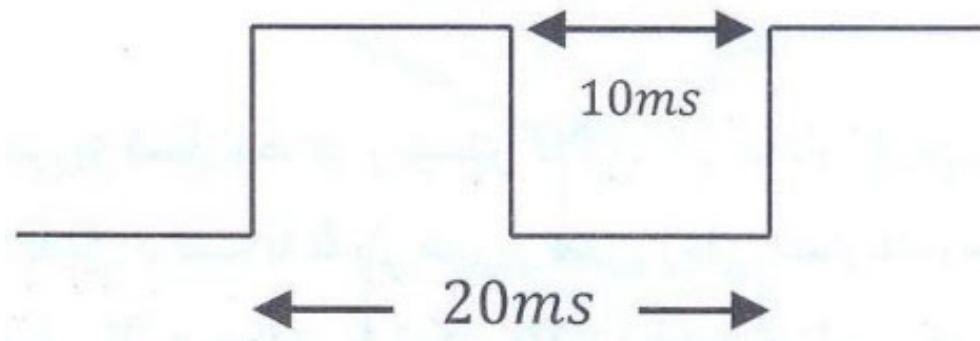
تایمر/کانتر با عملکرد زمان سنجی و شمارش در وضعیت نرمال

مثال : با فرض اینکه منبع کلک میکروکنترلر ۸ مگاهرتز است، پالس مربعی با فرکانس ۵۰ هرتز بر روی پایه PD0 ایجاد نمایید.

پیش از هر پیز باید زمان قطع و وصل موج مورد نظر را محاسبه نماییم:

$$f = 50Hz \rightarrow T = \frac{1}{50} = 20ms$$

بنابراین مطابق شکل زیر لازم است تا مدت زمان ۱۰ میلی ثانیه توسط تایمر ایجاد شود



آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

تایمر/کانتر با عملکرد زمان سنجی و شمارش در وضعیت نرمال

از آنجا که ۱۰ میلی ثانیه زمانی نسبتا طولانی است سعی میکنیم با تقسیم کلاک سیستم بر عدد بزرگتری، پایین ترین فرکانس پالس ممکن را برای شمارنده تامین نماییم. بدین منظور ابتدا به بررسی 10^{24} میپردازیم.

$$Counter_{clock} = \frac{8000000}{1024} = 7812.5Hz$$

با توجه به اینکه نتیجه تقسیم کلاک ۸ مگاهرتز بر 10^{24} مقداری صحیح نیست بنابراین بهتر است پیش تقسیم کننده ۲۵۶ را بررسی کنیم:

$$Counter_{clock} = \frac{8000000}{256} = 31250Hz$$

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

تایمر/کانتر با عملکرد زمان سنجی و شمارش در وضعیت نرمال

با این کلاک هر تیک شمارنده ۳۲ میکروثانیه خواهد بود. در نتیجه برای ایجاد ۱۰۰۰۰ میکروثانیه لازم است تا $10000 / 32 = 312.5$ پالس به شمارنده اعمال شود که این پاسخ نیز به دلیل غیرصحیح بودن تعداد تیکهای شمارنده مناسب نمیباشد. تقسیم بر ۶۴ را آزمایش میکنیم.

$$Counter_{clock} = \frac{8000000}{64} = 125000Hz$$

با این کلاک هر تیک شمارنده ۸ میکروثانیه خواهد بود که در نتیجه برای ایجاد ۱۰۰۰۰ میکروثانیه لازم است تا $10000 / 8 = 1250$ پالس به شمارنده اعمال شود.

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

تایمر/کانتر با عملکرد زمان سنجی و شمارش در وضعیت نرمال

اگر برای تامین کلاک شمارنده از این فرکانس استفاده نماییم برای ایجاد زمان ۱۰ میلی ثانیه یا ۱۰۰۰۰ میکروثانیه، نیاز به شمارش رشته صفر تا ۱۲۵۰ خواهد بود. از آنجا که حداکثر مقدار شمارنده ۸ بیتی عدد ۲۵۵ است این مسئله ممکن نخواهد بود.

برای حل این مشکل نیز لازم است تعداد سرریزهای شمارنده شمرده شود. بدیهی است که شمارنده ۸ بیتی $1250/256 = 4.88$ بار سرریز میکند. این تعداد سرریز مناسب نبوده و سعی میکنیم راه حلی براس صحیح شدن تعداد سرریز بیابیم.

$$\frac{1250}{255} = 4.9$$

$$\frac{1250}{250} = 5.00$$

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

تایمر/کانتر با عملکرد زمان سنجی و شمارش در وضعیت نرمال

```
void delay_10ms()
{
    unsigned char i=0;
    while (i<5)
    {
        TCNT = 6;
        TCCR0 = 0x03;
        while((TIFR & 0x01) ==0);
        TCCR0 = 0;
        TIFR |= 0x01;
        i++;
    }
}
```

بدین ترتیب کافی است تا با هر بار سرریز

شمارنده، در رجیستر TCNT0 مقدار اولیه ۶

بارگذاری شود تا رشته شمارش آن محدود به ۲۵۰

عدد محدود شود. بدین ترتیب با ۵ بار سرریز شدن

زمان شمارنده ۱۰ میلی ثانیه بدست خواهد آمد.

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

تایمر/کانتر با عملکرد زمان سنجی و شمارش در وضعیت نرمال

```
1 .include "m32def.inc"
```

```
2
3     LDI    R16,HIGH(RAMEND)
4     OUT    SPH,R16
5     LDI    R16,LOW(RAMEND)
6     OUT    SPL,R16
7
8     LDI    R16,0b00000001
9     OUT    DDRD,R16
10
11 BEGIN: LDI    R16,01
12     OUT    PORTD,R16
13     CALL   DELAY_100
14     LDI    R16,00
15     OUT    PORTD,R16
16     CALL   DELAY_100
17     RJMP  BEGIN
```

فایل منبع اسembly این مثال

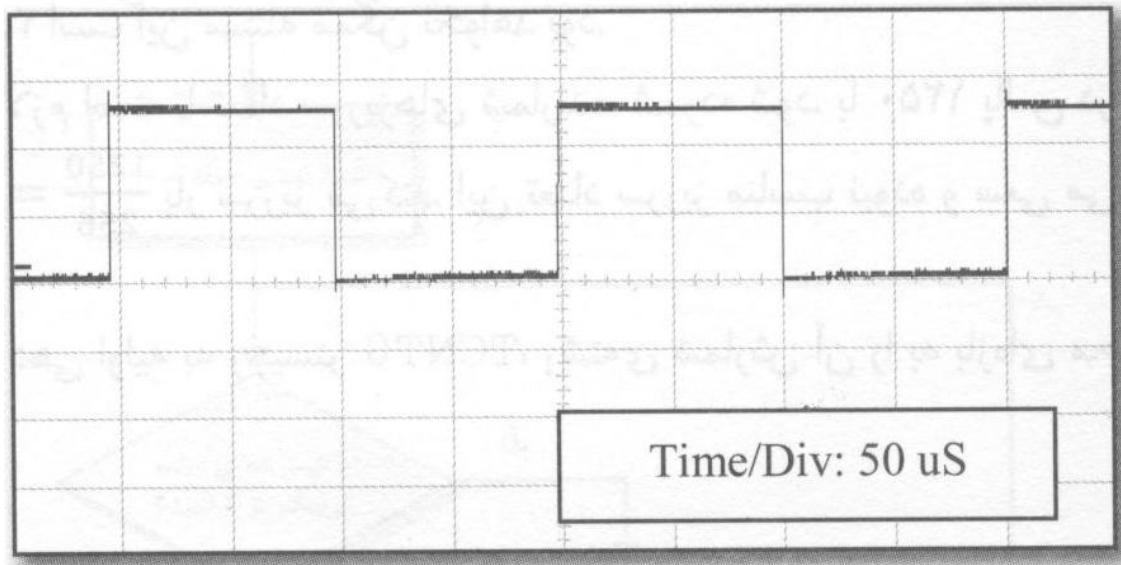
```
19    DELAY_100:
20    :
21 M:   LDI    R17,0
22     LDI    R16,6
23     OUT   TCNT0,R16
24     LDI    R16,0b00000011
25 S:   OUT   TCCRO,R16
26     IN    R16,TIFR
27     ANDI  R16,0b00000001
28     BREQ  S
29     LDI    R16,0
30     OUT   TCCRO,R16
31     LDI    R16,01
32     OUT   TIFR,R16
33     INC   R17
34     CPI   R17,5
35     BRNE  M
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
279
280
281
282
283
284
285
286
287
288
289
289
290
291
292
293
294
295
296
297
298
299
299
300
301
302
303
304
305
306
307
308
309
309
310
311
312
313
314
315
316
317
318
319
319
320
321
322
323
324
325
326
327
328
329
329
330
331
332
333
334
335
336
337
338
339
339
340
341
342
343
344
345
346
347
348
349
349
350
351
352
353
354
355
356
357
358
359
359
360
361
362
363
364
365
366
367
368
369
369
370
371
372
373
374
375
376
377
378
379
379
380
381
382
383
384
385
386
387
388
389
389
390
391
392
393
394
395
396
397
398
399
399
400
401
402
403
404
405
406
407
408
409
409
410
411
412
413
414
415
416
417
418
419
419
420
421
422
423
424
425
426
427
428
429
429
430
431
432
433
434
435
436
437
438
439
439
440
441
442
443
444
445
446
447
448
449
449
450
451
452
453
454
455
456
457
458
459
459
460
461
462
463
464
465
466
467
468
469
469
470
471
472
473
474
475
476
477
478
479
479
480
481
482
483
484
485
486
487
488
489
489
490
491
492
493
494
495
496
497
498
499
499
500
501
502
503
504
505
506
507
508
509
509
510
511
512
513
514
515
516
517
518
519
519
520
521
522
523
524
525
526
527
528
529
529
530
531
532
533
534
535
536
537
538
539
539
540
541
542
543
544
545
546
547
548
549
549
550
551
552
553
554
555
556
557
558
559
559
560
561
562
563
564
565
566
567
568
569
569
570
571
572
573
574
575
576
577
578
579
579
580
581
582
583
584
585
586
587
588
589
589
590
591
592
593
594
595
596
597
597
598
599
599
600
601
602
603
604
605
606
607
608
609
609
610
611
612
613
614
615
616
617
618
619
619
620
621
622
623
624
625
626
627
628
629
629
630
631
632
633
634
635
636
637
638
639
639
640
641
642
643
644
645
646
647
648
649
649
650
651
652
653
654
655
656
657
658
659
659
660
661
662
663
664
665
666
667
668
669
669
670
671
672
673
674
675
676
677
678
679
679
680
681
682
683
684
685
686
687
688
689
689
690
691
692
693
694
695
696
697
697
698
699
699
700
701
702
703
704
705
706
707
708
709
709
710
711
712
713
714
715
716
717
718
719
719
720
721
722
723
724
725
726
727
728
729
729
730
731
732
733
734
735
736
737
738
739
739
740
741
742
743
744
745
746
747
748
749
749
750
751
752
753
754
755
756
757
758
759
759
760
761
762
763
764
765
766
767
768
769
769
770
771
772
773
774
775
776
777
778
779
779
780
781
782
783
784
785
786
787
788
789
789
790
791
792
793
794
795
796
797
797
798
799
799
800
801
802
803
804
805
806
807
808
809
809
810
811
812
813
814
815
816
817
818
819
819
820
821
822
823
824
825
826
827
828
829
829
830
831
832
833
834
835
836
837
838
839
839
840
841
842
843
844
845
846
847
848
849
849
850
851
852
853
854
855
856
857
858
859
859
860
861
862
863
864
865
866
867
868
869
869
870
871
872
873
874
875
876
877
878
879
879
880
881
882
883
884
885
886
887
888
889
889
890
891
892
893
894
895
896
897
897
898
899
899
900
901
902
903
904
905
906
907
908
909
909
910
911
912
913
914
915
916
917
917
918
919
920
921
922
923
924
925
926
927
928
929
929
930
931
932
933
934
935
936
937
938
939
939
940
941
942
943
944
945
946
947
948
948
949
950
951
952
953
954
955
956
957
958
959
959
960
961
962
963
964
965
966
967
968
969
969
970
971
972
973
974
975
976
977
978
979
979
980
981
982
983
984
985
986
987
987
988
989
989
990
991
992
993
994
995
996
997
997
998
999
999
1000
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1009
1010
1011
1012
1013
1014
1015
1016
1017
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1089
1090
1091
1092
1093
1094
1095
1096
1097
1097
1098
1099
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1109
1110
1111
1112
1113
1114
1115
1116
1117
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1189
1190
1191
1192
1193
1194
1195
1196
1196
1197
1198
1199
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1209
1210
1211
1212
1213
1214
1215
1216
1217
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1289
1290
1291
1292
1293
1294
1295
1296
1297
1297
1298
1299
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1309
1310
1311
1312
1313
1314
1315
1316
1317
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1389
1390
1391
1392
1393
1394
1395
1396
1397
1397
1398
1399
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1409
1410
1411
1412
1413
1414
1415
1416
1417
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1489
1490
1491
1492
1493
1494
1495
1496
1497
1497
1498
1499
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1509
1510
1511
1512
1513
1514
1515
1516
1517
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1589
1590
1591
1592
1593
1594
1595
1596
1597
1597
1598
1599
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1609
1610
1611
1612
1613
1614
1615
1616
1617
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1689
1690
1691
1692
1693
1694
1695
1696
1697
1697
1698
1699
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1709
1710
1711
1712
1713
1714
1715
1716
1717
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1789
1790
1791
1792
1793
1794
1795
1796
1797
1797
1798
1799
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1809
1810
1811
1812
1813
1814
1815
1816
1817
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1889
1890
1891
1892
1893
1894
1895
1896
1897
1897
1898
1899
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1909
1910
1911
1912
1913
1914
1915
1916
1917
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1989
1990
1991
1992
1993
1994
1995
1996
1997
1997
1998
1999
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2009
2010
2011
2012
2013
2014
2015
2016
2017
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2089
2090
2091
2092
2093
2094
2095
2096
2097
2097
2098
2099
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2109
2110
2111
2112
2113
2114
2115
2116
2117
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2189
2190
2191
2192
2193
2194
2195
2196
2197
2197
2198
2199
2199

```

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

تایمر/کانتر با عملکرد زمان سنجی و شمارش در وضعیت نرمال

شکل موج ایجاد شده روی پایه PD0 در شکل زیر دیده میشود:



آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

آشنایی با وقفه‌ی سرریز تایمر/کانتر صفر

همانطور که پیشتر اشاره شد، سرریز شمارنده لحظه مهمی محسوب می‌شود و یکی از منابع وقفه است. بدین ترتیب این امکان فراهم می‌شود تا در فاصله بین دو سرریز، به جای سرکشی پرچم TOV0، بخش‌های دیگری از برنامه اجرا شود.

برای فعال نمودن وقفه سرریز شمارنده، می‌توان بیت TOIE0 از رجیستر TIMSK را یک نمود. در این شرایط اگر فعال ساز عمومی وقفه‌ها (I) یک شده باشد، یک شدن پرچم TOV0 می‌تواند باعث ایجاد وقفه شود. در این شرایط با اجرا شدن ISR به صورت خودکار پرچم سرریز پاک می‌شود.

TIMSK – Timer/Counter Interrupt Mask Register

Bit	7	6	5	4	3	2	1	0	TIMSK
	OCIE2	TOIE2	TICIE1	OCIE1A	OCIE1B	TOIE1	OCIE0	TOIE0	
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

این مثال را خود بررسی و اجرا نمایید

```
#include <mega32.h>
```

```
void init_io();
```

```
interrupt [TIM0_OVF] void timer0_ovf_isr(void)
```

```
{
```

```
    TCNT0 = 156;
```

```
    PORTD ^= (1<<PD0);
```

```
}
```

```
void main(void)
```

```
{
```

```
    init_io();
```

```
    while(1)
```

```
    {}
```

```
}
```

آشنایی با وقفه‌ی سرریز تایمر/کانتر صفر

برنامه قبلی را با استفاده از وقفه سرریز شمارنده بازنویسی کنید.

```
void init_io()
```

```
{
```

```
    PORTD = 0x00;
```

```
    DDRD = 0x01;
```

```
}
```

```
    TCNT = 156;
```

```
    TCCR0 = 0x02;
```

```
    TIMSK = 0x01;
```

```
#asm("sei");
```

```
}
```

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

```
1 .include "m32def.inc"
2 .ORG 0
3     RJMP BEGIN
4 .ORG OVF0addr
5     RJMP ROUTINE
6
7 BEGIN: LDI R16, HIGH(RAMEND)
8     OUT SPH, R16
9     LDI R16, LOW(RAMEND)
10    OUT SPL, R16
11
12 ; Initialize PIN0 PORTD as OUTPUT
13 ;-----
14 LDI R16, 0b00000001
15 OUT DDRD, R16
16
17 ; Initialize Timer 0
18 ;-----
19 LDI R16, 156
20 OUT TCNT0, R16
21 LDI R16, 2
22 OUT TCCR0, R16
23 LDI R16, 1
24 OUT TIMSK, R16
25 SEI
```

آشنایی با وقفه‌ی سرریز تایمر/کانتر صفر

فایل اسembلی منبع این مثال

26	HERE:	LDI	R16, 01
27	OUT	PORTD, R16	
28	RJMP	HERE	
29	ROUTINE:	LDI	R18, 156
30		OUT	TCNT0, R18
31		LDI	R17, 1
32		EOR	R16, R17
33		RETI	
34			
35			
36			
37			

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

آشنایی با وقفه‌ی سرریز تایمر/کانتر صفر

همانطور که ملاحظه میکنید در ISR سرریز شمارنده، TCNT0 با عدد ۱۵۶ بارگذاری شده و سپس عبارتی اجرا میشود که PDO را مکمل میکند. تفاوت مهم این برنامه و برنامه قبلی در این است که در این برنامه زمان ارزشمند پردازنده صرف چک کردن پرچم سرریز TOV0 نشده و بایک شدن پرچم به طور خودکار ISR اجرا میشود.

به عبارت دیگر این امکان وجود دارد از زمان ارزشمند پردازنده بطور مفیدتری استفاده کرد و در حلقه اصلی برنامه وظایف دیگری را به آن سپرد.

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

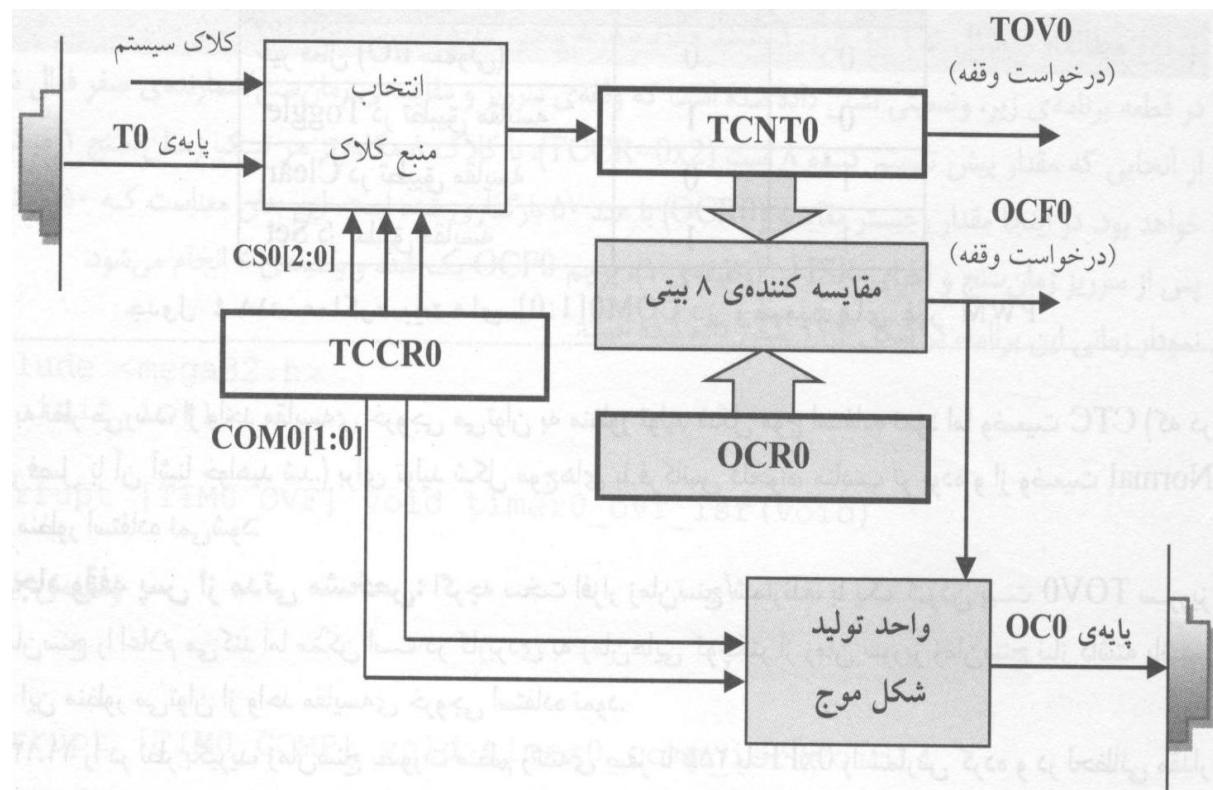
این مثال را خود بررسی و اجرا نمایید

آزمایش عملی

برنامه ای بنویسید که یک LED روشن را از چپ به راست و راست به چپ حرکت دهد. تأخیر باید از پورت C (یا همان دیپ سوئیچها) خوانده شود. یک شمارنده چهار رقمی نیز روی سون سگمنتها روشن شود و از صفر رو به بالا به شمارد. شمارنده در هر ثانیه یکبار عدد خود را افزایش دهد. دقیق کنید که اعداد روی سون سگمنت هر ۲۰ میلی ثانیه یکبار باید سون سگمنت ها را بازنویسی نماید تا اطلاعات بطور صحیح روی آنها دیده شود.

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

آشنایی با زمان تایمر/کانتر با امکان مقایسه



همانطور که از نام این حالت مشخص است، تایmer/کانتر قابلیت مقایسه مقدار TCNT0 را به صورت سخت افزاری دارد. برای آشنایی با این قابلیت، ابتدا لازم است تا با واحد مقایسه خروجی در تایmer/کانتر صفر آشنا شوید.

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

آشنایی با زمان تایمر/کانتر با امکان مقایسه

همانطور که ملاحظه میکنید، ثبات جدیدی به نام OCRO اضافه شده است. محتوی این ثبات در هر سیکل با مقدار TCNT0 مقایسه شده و در صورت برابری فلگ OCF0 در سیکل بعدی یک میشود. این لحظه برابری را تطبیق مقایسه (Compare Match) مینامیم. کاربردهای این تطبیق مقایسه عبارتند از:

۱- تولید شکل موج

۲- ایجاد وقفه پس از مدتی مشخص

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

آشنایی با زمان تایمر/کانتر با امکان مقایسه

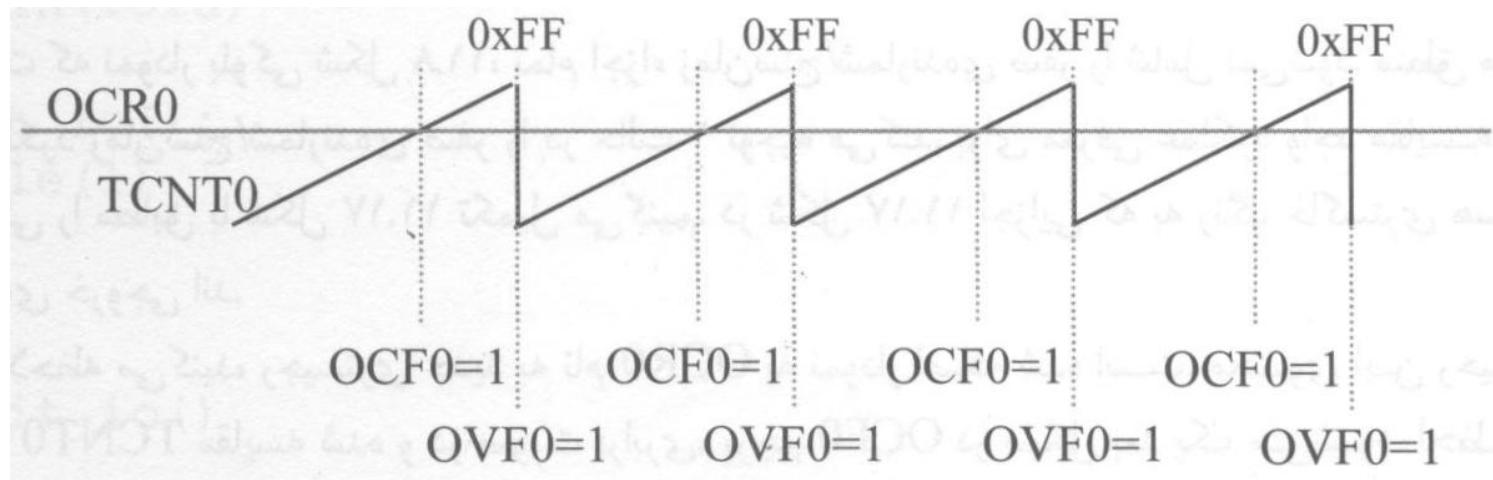
۱- تولید شکل موج - در لحظه یک شدن فلگ OCF0، فرمانی برای واحد تولید شکل موج صادر میشود. این واحد با توجه به تنظیماتی که از بیت‌های COM0[1:0] دریافت میدارد سطح منطقی پایه OC0 را

Bit	7	6	5	4	3	2	1	0	تغییر میدهد.															
	FOC0	WGM00	COM01	COM00	WGM01	CS02	CS01	CS00	TCCR0															
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W																
Initial Value	0	0	0	0	0	0	0	0																
اگرچه به نظر میرسد از واحد مقایسه خروجی میتوان به منظور تولید شکل موج استفاده نمود اما وضعیت CTC برای تولید شکل موجهای با فرکانس دلخواه مناسب تر بوده و از وضعیت نرمال به این منظور استفاده نمیشود.																								
<table border="1"><thead><tr><th>COM01</th><th>COM00</th><th>وضعیت پین 0</th></tr></thead><tbody><tr><td>0</td><td>0</td><td>غیرفعال (I/O معمولی)</td></tr><tr><td>0</td><td>1</td><td>در تطبیق مقایسه Toggle</td></tr><tr><td>1</td><td>0</td><td>در تطبیق مقایسه Clear</td></tr><tr><td>1</td><td>1</td><td>در تطبیق مقایسه Set</td></tr></tbody></table>										COM01	COM00	وضعیت پین 0	0	0	غیرفعال (I/O معمولی)	0	1	در تطبیق مقایسه Toggle	1	0	در تطبیق مقایسه Clear	1	1	در تطبیق مقایسه Set
COM01	COM00	وضعیت پین 0																						
0	0	غیرفعال (I/O معمولی)																						
0	1	در تطبیق مقایسه Toggle																						
1	0	در تطبیق مقایسه Clear																						
1	1	در تطبیق مقایسه Set																						

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

آشنایی با زمان تایمر/کانتر با امکان مقایسه

۲- ایجاد وقفه پس از مدتی مشخص - اگر چه سخت افزار تایمر/کانتر با یک کردن بیت TOV0 سرریز تایмер را اعلام میکند اما ممکن است در کاربردی به زمانهایی کوچکتر از زمان سرریز تایмер نیاز داشته باشیم. به این منظور میتوان از واحد مقایسه خروجی استفاده کرد.



آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

آشنایی با زمان تایمر/کانتر با امکان مقایسه

ممکن است این سوال پیش آید که میتوان با مقدار اولیه دادن به TCNT0 رشته شمارش آن را محدود نمود تا در فاصله زمانی مورد نظر فلگ TOV0 یک شود، پس مزیت واحد مقایسه چیست؟ پاسخ این است که اگر چه دو هر روش نتیجه یکسانی خواهند داشت اما با استفاده از واحد مقایسه تمام عملیات به صورت سخت افزاری انجام شده و در نتیجه برای مقدار اولیه دادن به TCNT0 وقت پردازنده تلف نمیشود. شرط لازم برای درخواست وقفه در لحظه تطبیق مقایسه یک بودن بیت OCIE0 از ثبات TIMSK است. همچنین لازم است بیت فعال ساز عمومی وقفه ها با دستور SEI یک شود.

Bit	7	6	5	4	3	2	1	0	TIMSK
	OCIE2	TOIE2	TICIE1	OCIE1A	OCIE1B	TOIE1	OCIE0	TOIE0	
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

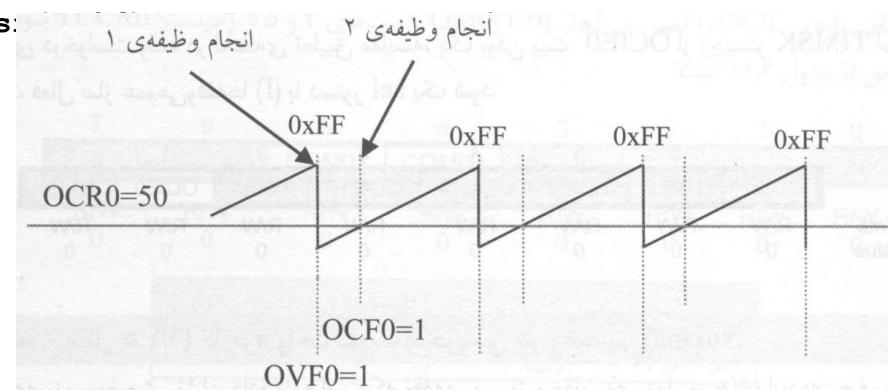
آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

مثال: کاربرد واحد مقایسه خروجی در وضعیت نرمال

در قطعه برنامه زیر وضعیتی نشان داده شده است که وقفه سرریز و مقایسه زمان تایمر/کانتر صفر فعال شده اند. از آنجایی که مقدار پیش تقسیم کننده ۸ است ($TCCR0=0x02$), با کلاک ۸ مگاهرتز هر تیک تایмер یک میکروثانیه خواهد بود در اینجا مقدار ثبات مقایسه ($OCR0$) با عدد ۵۰ بارگذاری شده است.

```
#include <mega32.h>
void init_io();
interrupt [TIM0_OVF] void timer0_ovf_isr(void)
{
    وظیفه یک
}
interrupt [TIM0_COMP] void timer0_comp_isr(void)
{
    وظیفه دو
}
void main(void)
{
    init_io();
    while (1) {};
}
```

```
void init_io();
{
    TCNT0 = 60;
    OCR0 = 50;
    TCCR0 = 0x02;
    TIMSK = 0x03;
    #asm("sei");
```



آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

آشنایی با وضعیت CTC در تایمر/کانتر صفر

اولین نکته در وضعیت جدید این است که مقدار بیتهاي WGM0[1:0] دیگر ۰۰ نبوده و با توجه به

جدول مربوطه برابر ۰۱ است. (اسلاید شماره ۲۰)

Bit	7	6	5	4	3	2	1	0	TCCR0
	FOC0	WGM00	COM01	COM00	WGM01	CS02	CS01	CS00	
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

Initial Value

0	0	0	0	0	0	0	0	0
---	---	---	---	---	---	---	---	---

اصلی ترین کاربرد CTC ایجاد تاخیر و شکل موج به صورت سخت افزاری است. بیاد دارید که در یکی از مثالهای مطرح داده برای ایجاد ۱۰۰ میکروثانیه تاخیر، ثبات TCNT0 را با عدد ۱۵۶ بارگذاری نمودیم، بدین ترتیب رشته شمارش این ثبات، محدود به عدد ۱۵۶ تا ۲۵۵ شد. حال سوال این است که آیا میتوان به جای مقدار اولیه دادن به شمارنده، رشته شمارش آن را از انتهای محدود نمود؟

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

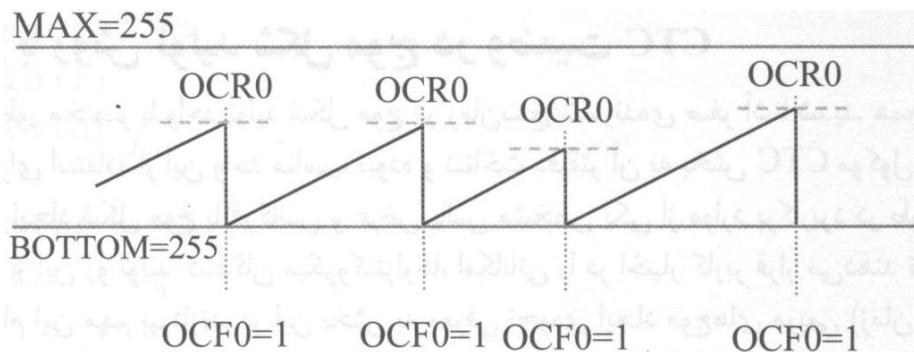
آشنایی با وضعیت CTC در تایمر/کانتر صفر

قبل از پاسخ به این سوال سه واژه کلید که در ادامه مکرراً از آنها استفاده خواهیم کرد را تعریف میکنیم:

- BOTTOM: شمارنده زمانی که 0x00 شود به رسیده است.
- MAX: شمارنده زمانی که 0xFF (یا ۲۵۵ دسیمال) شود به رسیده است.
- TOP: شمارنده زمانی به TOP میرسد که برابر با بالاترین مقدار در رشته شمارش خود باشد. این مقدار بستگی به وضعیت کاری تایمر/کانتر دارد.

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

آشنایی با وضعیت CTC در تایمر/کانتر صفر



حال پاسخ سوال مطرح شده را به این

ترتیب میدهیم:

در وضعیت CTC میتوان TOP شمارنده را به یک عدد دلخواه کوچکتر از ۲۵۶ تغییر داد.

همانطور که میدانید در وضعیت نرمال، TOP تایмер عدد ۲۵۵ بود و این عدد بالاترین مقدار در رشته

شمارش ثبات TCNT است. اکنون با این مفهوم آشنا میشوید که در وضعیت CTC مقدار TOP آن

عددی است که در ثبات OCR0 بارگذاری شده است. به بیان دیگر، بر خلاف وضعیت نرمال،

شمارنده پس از تطبیق مقایسه از عدد OCR0 عبور نمیکند بلکه در همان لحظه مقدار شمارنده صفر

میشود.

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

شمارنده را در وضعیت CTC به شکلی تنظیم نمایید تا هر ۱۰۰ میکروثانیه وظیفه انجام شود

```
#include <mega32.h>
void init_io();

// Timer 0 output compare interrupt service
routine
interrupt [TIM0_COMP] void timer0_comp_isr(void)
{
    وظیفه یک
}
Void main(void)
{
    init_io();
    while (1) {};
}
void init_io()
{
    TCCR0 = 0x0A;
    OCRO = 99;
    TIMSK = 0x02;
    #asm("sei");
}
```

همانطور که مشخص است تایمر/کانتر صفر در وضعیت CTC با TOP=99 تنظیم شده است. بنابراین رشته شمارش TCNT0 محدود به صفر تا ۹۹ خواهد بود. از آنجایی که وقفه مقایسه تایمر/کانتر صفر فعال شده است، پس از ۱۰۰ پالس ساعت، ISR مربوط به آن اجرا شده و در آنجا، وظیفه یک انجام میشود.

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

آشنایی با روش تولید شکل موج در وضعیت CTC

همانطور که میدانید ایجاد شکل موج با فرکانس و عرض پالس مشخص یکی از موارد پرکاربرد در طراحی سیستم‌های میکروکنترلری است. از اینرو تولید کنندگان میکروکنترلرها، امکاناتی را در اختیار کاربر قرار میدهند تا بتوانند به صورت سخت افزاری به انجام این مهم بپردازنند.

برای درک اهمیت مثالی را در نظر بگیرید که وظیفه اصلی آن معکوس نمودن یکی از پایه های I/O بود. در این مثال با هر بار تطبیق مقایسه (پس از گذشت ۱۰۰ میکروثانیه) آن پایه مورد نظر تغییر وضعیت داده میشد و بدین ترتیب پالسی را با دو تناوب ۲۰۰ میکروثانیه ایجاد نمودیم. اما اشکال این روش آن است که اینکار زمانی را از پردازنده اشغال میکند. برای حل این مشکل میتوان از واحد تولید شکل موج استفاده نمود.

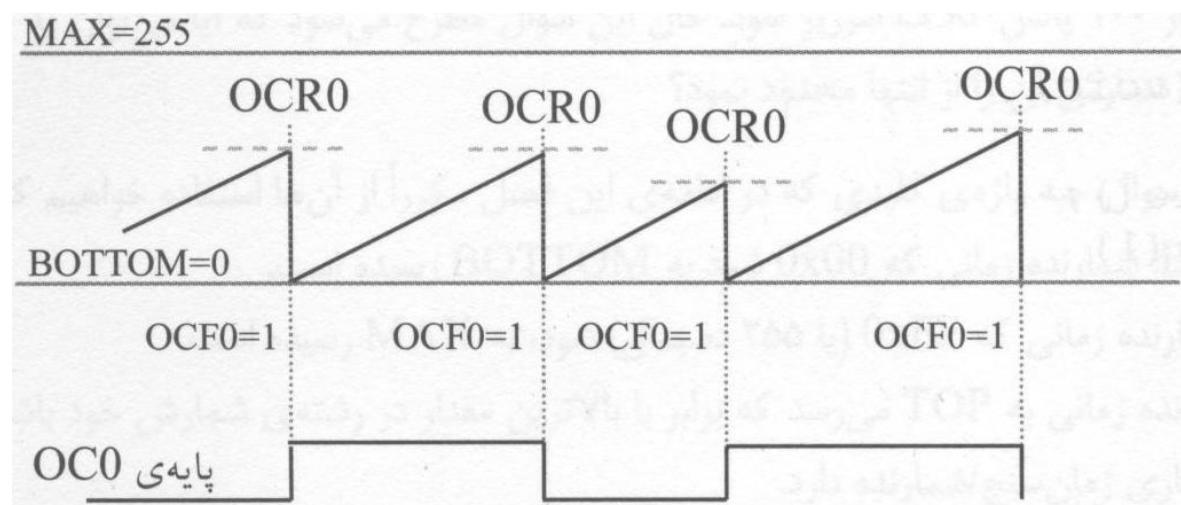
آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

آشنایی با روش تولید شکل موج در وضعیت CTC

میدانید که در لحظه تطبیق مقایسه فلگ OCF0 یک میشود و امکان درخواست وقفه نیز وجود دارد. یک شدن این فلگ به صورت داخلی فرمانی را برای واحد تولید شکل موج صادر کرده و آن واحد نیز با توجه به وضعیت بیتهاي [OC0[1:0] COM0] پایه OC0 را تغییر میدهد. مزیت مهم این روش آن است که تمام عملیات لازم برای تولید شکل موج به صورت خودکار انجام میشود و برنامه درگیر آن نمیگردد و اشکال این خواهد بود که تنها بر روی OC0 (پایه شماره ۴ قطعه ATmega32) میتوان شکل موج ایجاد نمود.

$$F_{OC0} = \frac{F_{CLK}}{2 \times N(1 + OCR0)}$$

فرکانس شکل موج ایجاد شده از
رابطه زیر بدست می آید:



آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

بازنویسی مثال قبلی

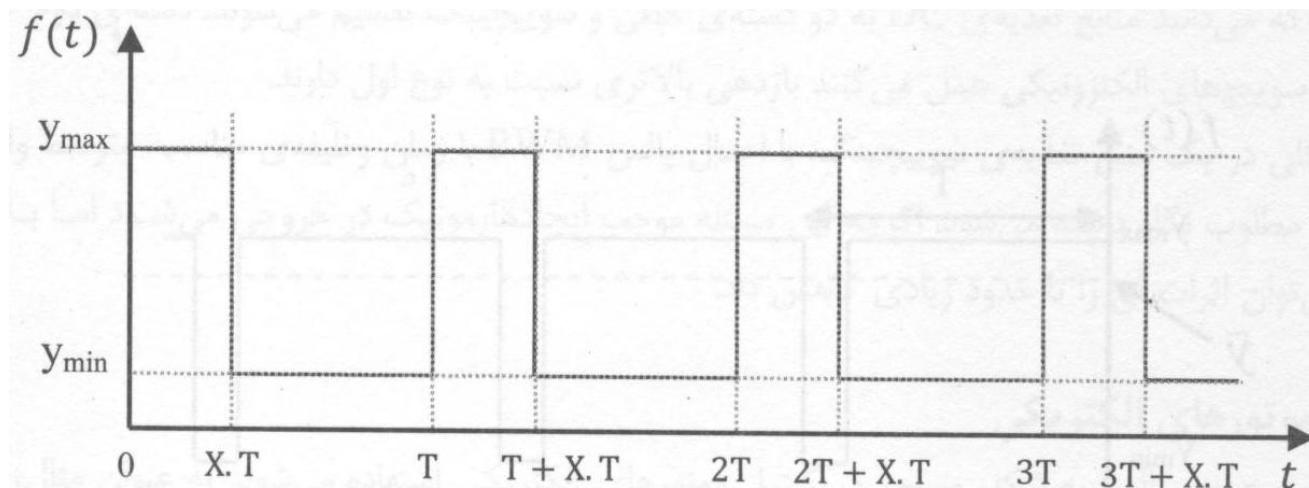
در یکی از مثالهای قبلی برای ایجاد فرکانس ۵۰ کیلوهرتز، برنامه نقش مهمی را دارا بود. اما در وضعیت CTC همانطور که انتظار می‌رود تمام عملیات را سخت افزار انجام داده و برنامه تنها شامل مقداردهی اولیه در تابع `init_io()` است.

```
#include <mega32.h>
void init_io();
Void main(void)
{
    init_io();
    while (1) {};
}
void init_io();
{
    DDRB = 0x08;           //Output OC0
    TCCR0 = 0x1A;          //Prescaler = 1 and Mode: CTC
                           //Toggle on compare match
    OCR0 = 99;             //Set for 100 count
}
```

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

آشنایی با مفهوم مدولاسیون عرض پالس (PWM)

معمولاً هدف از مدوله سازی عرض پالس یک سیگنال یا منبع توان، انتقال اطلاعات و یا کنترل توان تحویلی به بار است. بدین منظور لازم است تا عرض آن پالس، تحت تاثیر یک سیگنال مدوله کننده تغییر نماید. برای روشن شدن موضوع، شکل زیر را در نظر بگیرید:



آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

آشنایی با مفهوم مدولاسیون عرض پالس (PWM)

در اینجا یک پالس به نام $f(t)$ که دامنه آن بین y_{\min} تا y_{\max} و زمان وظیفه آن X است نشان داده شده است. بدیهی است که مقدار متوسط این موج از رابطه زیر بدست می‌آید:

$$\bar{y} = \frac{1}{T} \int_0^T f(t) dt$$

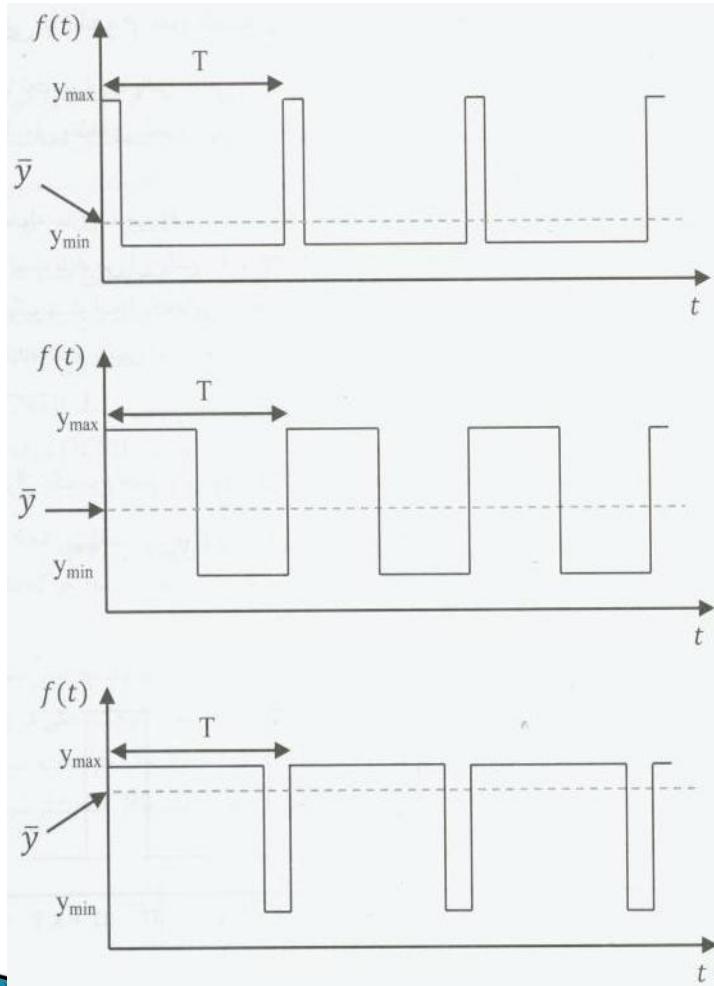
از آنجا که انتگرال یک موج متناوب بر روی یک دوره تناوب برابر سطح زیر آن است \bar{y} آن برابر است با:

$$\bar{y} = \frac{1}{T} (X.T.y_{\max} + (T - X.T).y_{\min}) = X.y_{\max} + (1 - X).y_{\min}$$

با توجه به این رابطه مشخص است که مقدار متوسط سیگنال به طور مستقیم وابسته به زمان وظیفه است.

آشنایی با میکروکنترلر AVR بخشن تایмер/کانتر

آشنایی با مفهوم مدولاسیون عرض پالس (PWM)



در شکل‌های زیر مقدار متوسط یک موج PWM با سه زمان وظیفه متفاوت نشان داده شده است. ملاحظه می‌کنید که با افزایش عرض پالس مقدار \bar{y} افزایش می‌باید.

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

برخی از کاربردهای مدولاسیون عرض پالس

- انتقال اطلاعات
- الکترونیک قدرت
- منابع تغذیه سوئیچینگ
- کنترل موتورهای الکتریکی
- کاربردهای صوتی



آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

آشنایی با وضیت Fast PWM در تایمر/کانتر صفر

اولین نکته این که برای تنظیم نمودن تایمر/کانتر لازم است تا بیت های WGM0[1:0] برابر 11 تنظیم شوند. دومین نکته این است که عملکرد بیتهاي COM0[1:0] در این وضعیت مطابق جدول زیر است:

COM01	COM00	وضعیت پین OC0
0	0	غیر فعال (I/O معمولی)
0	1	رزرو شده
1	0	صفر کردن OC0 در تطبیق مقایسه و یک کردن آن در PWM (TOP غیرمعکوس)
1	1	یک کردن OC0 در تطبیق مقایسه و پاک کردن آن در PWM (TOP معکوس)

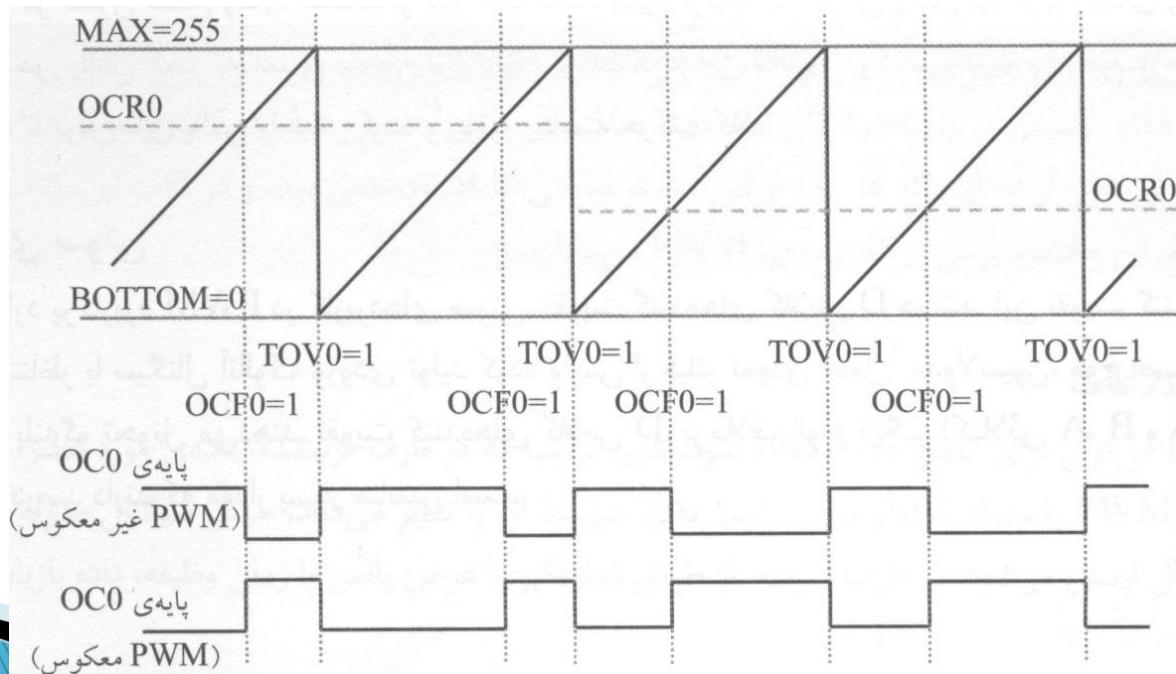
همانگونه که مشاهده میکنید، عملکرد این بیت در وضعیت Fast PWM بسیار متفاوت از وضعیتهاي غیر PWM است.

آشنایی با میکروکنترلر AVR بخشن تایمر/کانتر

آشنایی با وضیت Fast PWM در تایمر/کانتر صفر

در وضعیت PWM غیرمعکوس، شمارنده از BOTTOM تا MAX به صورت صعودی میشمارد، در لحظه تطبیق مقایسه، پایه OC0 پاک شده و با رسیدن به BOTTOM مجدداً یک میشود.

در وضعیت PWM معکوس نیز، شمارنده از MAX تا BOTTOM میشمارد، در لحظه تطبیق مقایسه پایه OC0 یک شده و با رسیدن به BOTTOM صفر خواهد شد.



آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

آشنایی با وضیت Fast PWM در تایمر/کانتر صفر

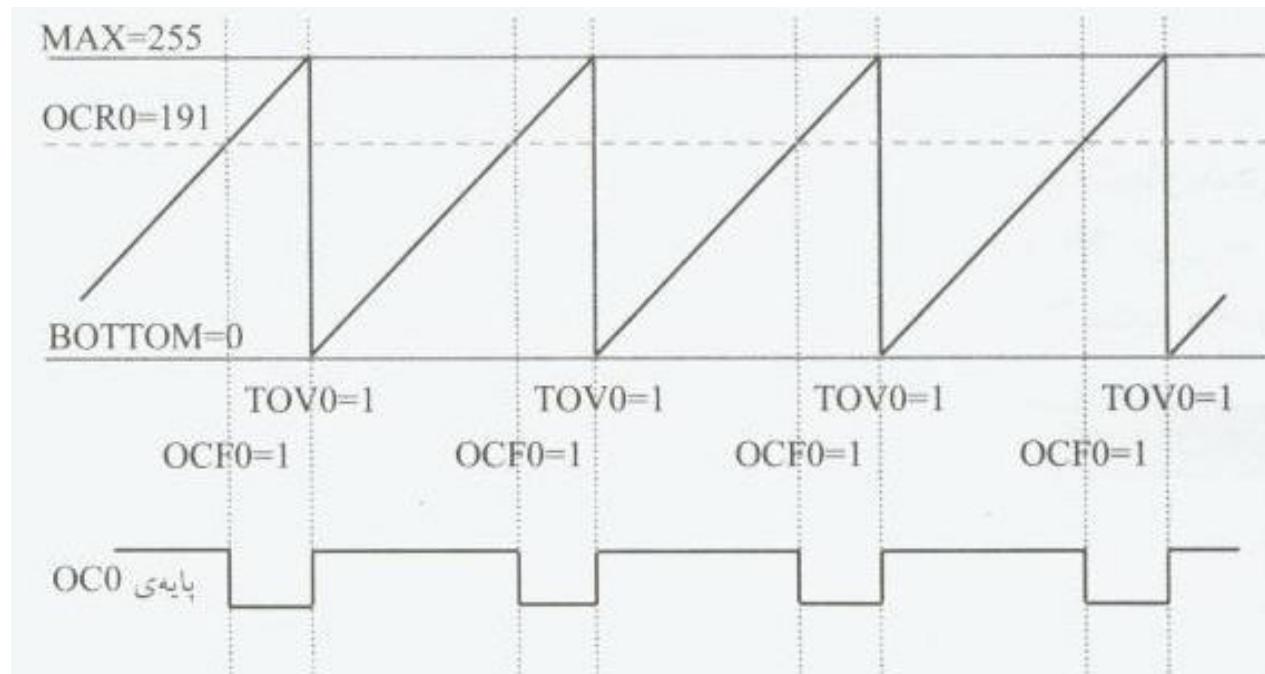
انتساب مقادیر خاص به ثبات OCR0 باعث ایجاد حالت‌های زیر می‌شود:

- در صورتی که مقدار OCR0 برابر با صفر شود، با هر بار سرریز شمارنده یک پالس سوزنی در خروجی ایجاد خواهد شد.
- بارگذاری مقدار ۲۵۵ در OCR0 موجب می‌شود تا بسته به وضعیت خروجی بیتهاي [OC0[1:0] پین OCR0 همواره صفر یا یک باشد. در صورتیکه COM0[1:0] برابر با ۲ باشد. خروجی همواره یک بوده و در صورتی که این دو بیت برابر ۳ باشند (PWM معکوس) خروجی همواره صفر است.
- فرکانس موج خروجی در وضعیت Fast PWM از رابطه زیر بدست می‌آید: (منظور از N مقدار پیش تقسیم کننده است)

$$f = \frac{f_{CLK}}{N \times 256}$$

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

مثال: ایجاد یک موج PWN غیرمعکوس با دوره تناوب ۲۵۶ میکروثانیه و زمان وظیفه ۷۵ روی پایه OC0



آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

مثال: ایجاد یک موج PWN غیرمعکوس با دوره تناوب ۲۵۶ میکروثانیه

```
#include <mega32.h>
void init_io();
Void main(void)
{
    init_io();
    while (1) {};
}
void init_io();
{
    DDRB = 0x08;           //Output OC0
    PORTB = 0x00
    TCCR0 = 0x6A;          //Prescaler = 8
                           //Mode: non-Inverting Fast PWM
    OCR0 = 191;            //Duty cycle : 75%
}
```

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

آشنایی با وضیت Phase Correct PWM در تایمر/کانتر صفر

اولین نکته این که برای تنظیم نمودن تایمر/کانتر لازم است تا بیت های [1:0] WGM0 برابر 01 تنظیم شوند. دومین نکته این است که عملکرد بیتهاي [1:0] COM0 در این وضعیت مطابق جدول زیر است:

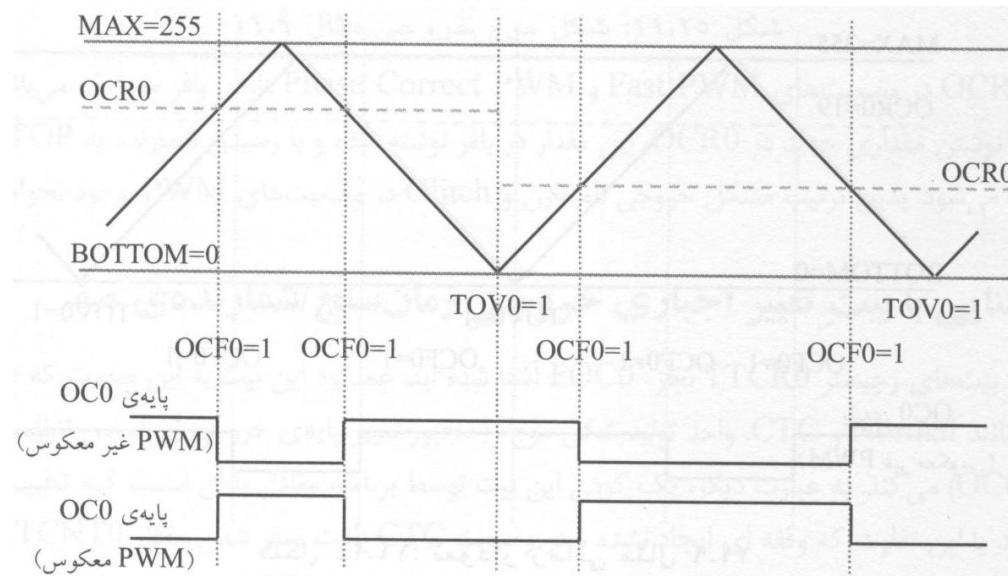
COM01	COM00	وضعیت پین OC0
0	0	غیر فعال (I/O معمولی)
0	1	رزرو شده
1	0	پاک کردن OC0 در تطبیق مقایسه لبه صعودی و یک کردن آن در لبه نزولی (PWM) غیرمعکوس)
1	1	یک کردن OC0 در تطبیق مقایسه لبه صعودی و پاک کردن آن در به نزولی (PWM معکوس)

در این وضعیت، شمارنده مرتبا از BOTTOM تا MAX شمرده و با رسیدن به MAX به صورت نزولی تا BOTTOM میشمارد.

آشنایی با میکروکنترلر AVR بخشن تایمر/کانتر

آشنایی با وضیت Phase Correct PWM در تایمر/کانتر صفر

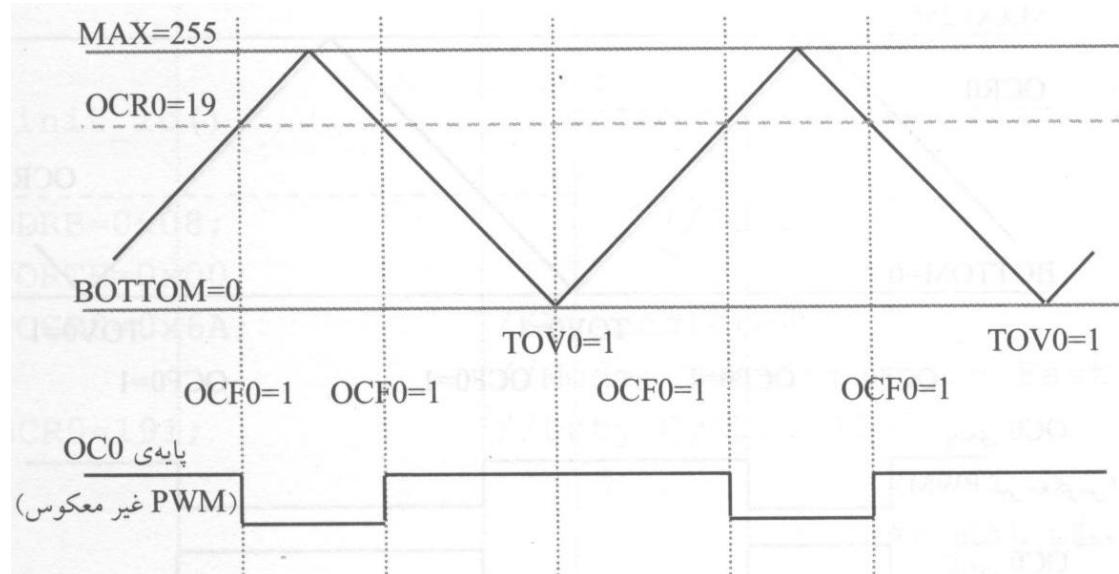
بنابراین طبق شکل زیر مقدار ثبات TCNT0 در دو لبه شمارش با OCR0 برابر میشود. در حالت PWM غیرمعکوس، تطبیق مقایسه روی داده در لبه شمارش صعودی پین را صفر کرده و تطبیق مقایسه در لبه شمارش صعودی پین OC0 را یک میکند. در مورد PWM معکوس، وضعیتی عکس برقرار است.



آشنایی با میکروکنترلر AVR بخش تایмер / کانتر

مثال ایجاد یک موج PWM غیرمعکوس با فرکانس تقریبی دو کیلوهرتز و زمان وظیفه ۷۵٪ بر روی پایه OC0

The figure illustrates the generation of a PWM signal from a triangle wave. The top section shows a triangle wave oscillating between two levels: MAX=255 and BOTTOM=0. The bottom section shows a digital signal labeled OC0 (پایهی), which is high whenever the triangle wave is above the OCF0 level. The label TOV0=1 is placed at the peak of the triangle wave. The labels OCF0=1 are placed at the points where the triangle wave crosses the OCF0 level.



آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

آشنایی با سخت افزار تایمر/کانتر یک

تایмер/کانتر ۱۶ بیتی از دقت بالاتری برخوردار است. بدین ترتیب ثبات TCNT1 شانزده بیتی است و میتواند رشته صفر تا ۶۵۵۳۵ را شمارش کند. مزایای این مسئله از دو دیدگاه قابل بررسی است:

- ۱- در وضعیتهاي غير PWM شمارنده ديرتر سرريز شده و در نتیجه امكان شمارش پالسهاي بيشتر امكان پذير است. امكان ايجاد زمانهاي طولاني تر وجود خواهد داشت.
- ۲- از نقطه نظر وضعیتهاي PWM شكل موج توليد شده، گامهاي کوچکتری در خروجي داشته و به عبارت ديگر رزولوشن بالاتری دارد.

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

آشنایی با سخت افزار تایمر/کانتر یک

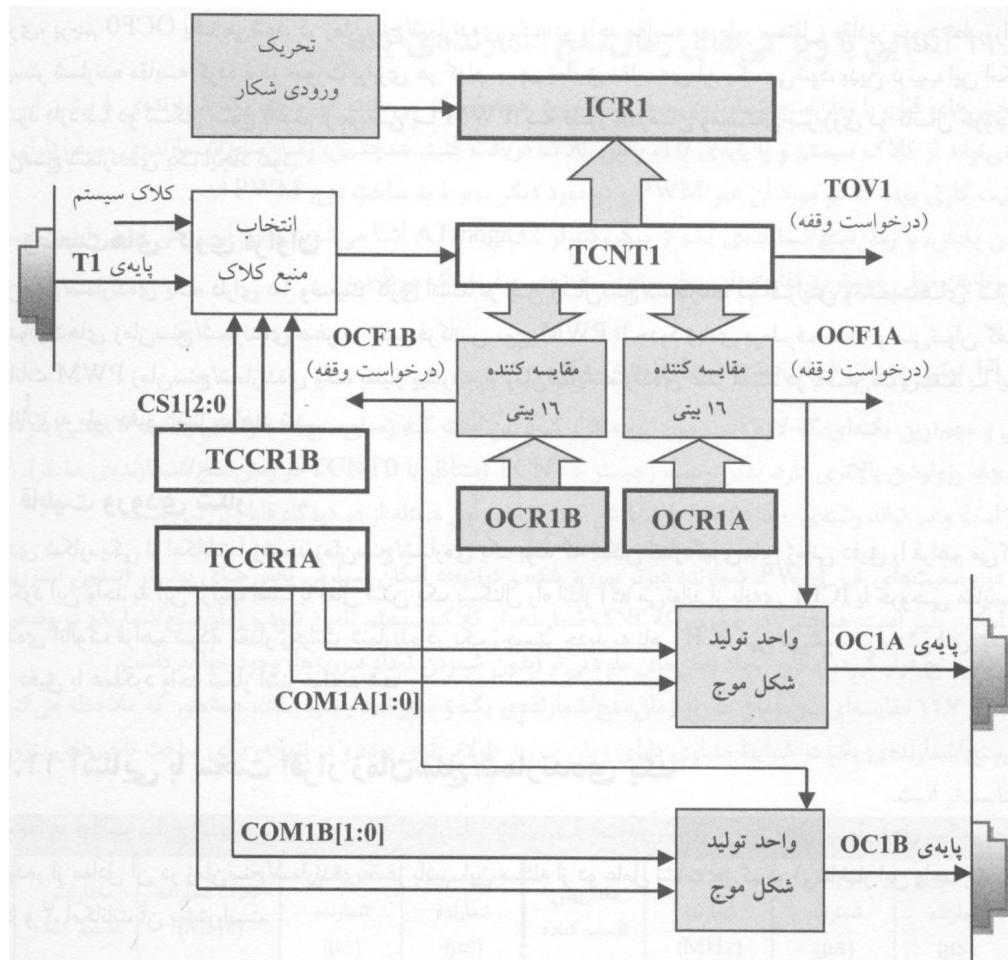
دو واحد مقایسه مستقل

بدین ترتیب این امکان وجود دارد تا دو شکل موج (اعم از مربعی یا PWM) به طور همزمان و مستقل بر روی دو کanal خروجی تایмер/کانتر یک ایجاد شود.

وضعیت های کاری فراوان

تایмер/کانتر یک دارای ۱۵ وضعیت کاری است. میتوان گفت امکانات PWM این تایмер/کانتر بسیار پیشرفته تر از تایмер/کانتر صفر است.

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر



آشنایی با سخت افزار تایمر/کانتر یک

ثبتات TCNT1 با دریافت پالس کلاک، رشته صفر تا ۶۵۵۳۵ را شمارش کرده و پس از سرریز شدن، فلگ TOV1 یک میشود.

تایمر/کانتر یک نیز میتواند در دو وضعیت کانتر و تایmer عمل کند. در وضعیت اول منبع کلاک، پالسهایی خواهد بود که از دنیای بیرون تامین میشود و در وضعیت دوم پالس کلاک سیستم منبع شمارش ثبات TCNT1 است.

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

انتخاب منبع کلاک

انتخاب منبع کلاک مطابق با جدول زیر از طریق بیت‌های $CS1[1:0]$ انجام می‌شود.

Bit	7	6	5	4	3	2	1	0	TCCR1B
	ICNC1	ICES1	-	WGM13	WGM12	CS12	CS11	CS10	
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	
منبع کلاک									
0	0	0	0	0	0	0	0	0	بدون کلاک (متوقف)
0	0	0	1	0	0	0	0	0	کلاک سیستم (بدون تقسیم)
0	1	0	0	0	0	0	0	0	کلاک سیستم / ۸
0	1	0	1	0	0	0	0	0	کلاک سیستم / ۶۴
1	0	0	0	0	0	0	0	0	کلاک سیستم / ۲۵۶
1	0	0	1	0	0	0	0	0	کلاک سیستم / ۱۰۲۴
1	1	0	0	0	0	0	0	0	لبه‌ی پایین رونده‌ی پالس خارجی (T1)
1	1	0	1	0	0	0	0	0	لبه‌ی بالا رونده‌ی پالس خارجی (T1)

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

مقایسه کننده های تایمر/کانتر یک

تایмер/کانتر دارای دو ثبات مقایسه، دو مقایسه کننده دیجیتال، دو فلگ تطبیق مقایسه و دو پایه خروجی مقایسه است. محتوای ثبات ۱۶ بیتی TCNT1 به طور پیوسته و مستقل با دو ثبات OCR1A و OCR1B مقایسه میشود، در صورتی که تطبیق مقایسه در واحد A روی دهد فلگ OCF1A و در صورتی که در واحد B تطبیق مقایسه ایجاد شده باشد، فلگ OCF1B یک خواهد شد.

Bit	7	6	5	4	3	2	1	0	TIFR
	OCF2	TOV2	ICF1	OCF1A	OCF1B	TOV1	OCF0	TOV0	
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

Initial Value

0	0	0	0	0	0	0	0	0
---	---	---	---	---	---	---	---	---

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

فعال کننده های مقایسه کننده تایمر/کانتر یک

در صورتی که بیت فعال ساز وقفه آن فلگ و فعال ساز عمومی وقفه ها یک باشند، یک شدن هر کدام از فلگهای OCF1A و OCF1B میتواند باعث درخواست وقفه شود. این دو بیت فعال ساز به نام های OCIE1A و OCIE1B از ثبات TIMSK میباشند. همچنین بیت ۲ از این ثبات به نام TOIE1 فعال ساز وقفه سرریز کانتر یک است.

Bit	7	6	5	4	3	2	1	0	TIMSK
	OCIE2	TOIE2	TICIE1	OCIE1A	OCIE1B	TOIE1	OCIE0	TOIE0	
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

کلمات کلیدی

سه واژه کلید که در ادامه مکرراً از آنها استفاده خواهیم کرد را تعریف میکنیم:

- BOTTOM: شمارنده زمانی که 0x0000 شود به رسیده است.
- MAX: زمانی که شمارنده 0xFFFF (دسیمال ۶۵۵۳۵) شود به خود رسیده است.
- TOP: زمانی که شمارنده برابر با بالاترین عدد در رشته شمارش خود میشود به رسیده است.

نکته مهمی که در مورد TOP وجود دارد این است که در تایمر/کانتر صفر، تغییر دادن TOP به عددی کمتر از MAX تنها در وضعیت CTC ممکن بود اما در تایمر/کانتر یک در وضعیت های PWM نیز میتوان مقدار TOP را به عددی کمتر از MAX تغییر داد.

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

وضعیت های کاری تایمر/کانتر یک

تایمر/کانتر یک دارای ۱۵ وضعیت کاری است که یک وضعیت **NORMAL**، دو وضعیت **CTC** و ۱۲ وضعیت نیز مربوط به PWM میباشند.
این وضعیت ها را در جدول مشاهده میکنید.

شماره	WGM13	WGM12	WGM11	WGM10	وضعیت کاری	TOP	بروزرسانی OCR1x	لحظه‌ی یک شدن TOV1
0	0	0	0	0	Normal	0xFFFF	فوری	MAX
1	0	0	0	1	PWM, Phase Correct, 8-bit	0x00FF	TOP	BOTTOM
2	0	0	1	0	PWM, Phase Correct, 9-bit	0x01FF	TOP	BOTTOM
3	0	0	1	1	PWM, Phase Correct, 10-bit	0x03FF	TOP	BOTTOM
4	0	1	0	0	CTC	OCR1A	فوری	MAX
5	0	1	0	1	Fast PWM, 8-bit	0x00FF	BOTTOM	TOP
6	0	1	1	0	Fast PWM, 9-bit	0x01FF	BOTTOM	TOP
7	0	1	1	1	Fast PWM, 10-bit	0x03FF	BOTTOM	TOP
8	1	0	0	0	PWM, Phase and Frequency Correct	ICR1	BOTTOM	BOTTOM
9	1	0	0	1	PWM, Phase and Frequency Correct	OCR1A	BOTTOM	BOTTOM
10	1	0	1	0	PWM, Phase Correct	ICR1	TOP	BOTTOM
11	1	0	1	1	PWM, Phase Correct	OCR1A	TOP	BOTTOM
12	1	1	0	0	CTC	ICR1	فوری	MAX
13	1	1	0	1	رزرو شده	-	-	-
14	1	1	1	0	Fast PWM	ICR1	BOTTOM	TOP
15	1	1	1	1	Fast PWM	OCR1A	BOTTOM	TOP

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

وضعیت های کاری تایمر/کانتر یک

انتخاب هر یک از این وضعیتها از طریق بیت‌های WGM1[3:0] از ثباتهای TCCR1A و TCCR1B انجام می‌شود.

Bit	7	6	5	4	3	2	1	0	
	COM1A1	COM1A0	COM1B1	COM1B0	FOC1A	FOC1B	WGM11	WGM10	TCCR1A
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	.
Initial Value	0	0	0	0	0	0	0	0	
Bit	7	6	5	4	3	2	1	0	
	ICNC1	ICES1	-	WGM13	WGM12	CS12	CS11	CS10	TCCR1B
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

همانطور که ملاحظه می‌کنید در جدول تنها یک وضعیت NORMAL وجود دارد که در این حالت TOP کانتر یک عدد MAX یا همان 0xFFFF است.

دو وضعیت CTC (۴ و ۱۲) در TOP با یکدیگر اختلاف دارند، بدین معنا که در وضعیت ۴، کانتر با رسیدن به عددی که در ثبات OCR1A است، پاک شده و در وضعیت ۱۲، حداکثر مقدار ثبات TCNT1، برابر با مقدار ICR1 است (بدیهی است که حالات شکار این حالت غیرفعال است).

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

آشنایی با نحوه دسترسی به ثباتهای ۱۶ بیتی

ثباتهای ICR1، OCR1A، OCR1B و TCNT1 شانزده بیتی هستند و میکروکنترلر از طریق باس داده ۸ بیتی به آنها دسترسی پیدا میکند، بنابراین نوشتن یا خواندن یکی از آنها باید طی دو سیکل انجام شود. مشکلی که به این ترتیب به وجود میآید این است که مقدار دهی متوالی به دو بایت این ثباتها، در برخی موارد ممکن است باعث ایجاد مقادیر میانی ناخواسته ای شود.

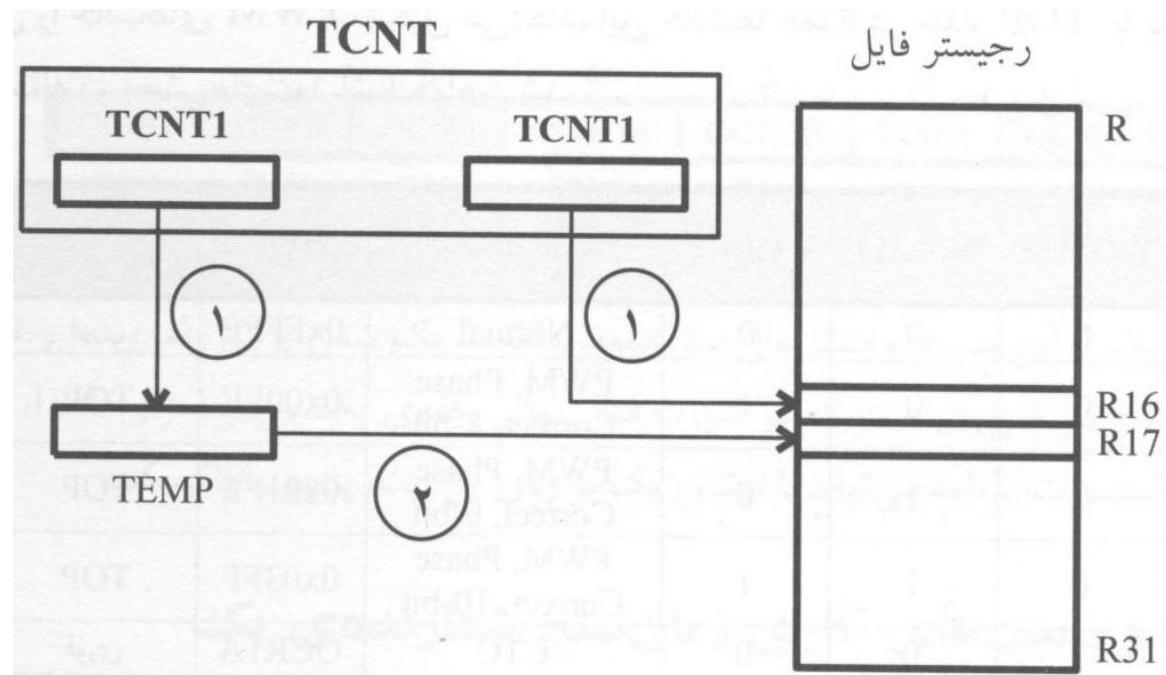
برای حل این مشکل از یک ثبات موقتی به نام TEMP استفاده میشود. این ثبات آدرسی در فضای ثباتهای I/O ندارد و تنها برای عملیات داخلی از آن استفاده میشود. این ثبات بین تمام ثباتهای ۱۶ بیتی مشترک است و موقتا بایت بالاتر داده ۱۶ بیتی را نگهداری میکند.

آشنایی با میکروکنترلر AVR بخشن تایمر/کانتر

آشنایی با نحوه دسترسی به ثباتهای ۱۶ بیتی

برای روشن شدن نحوه عملکرد ثبات TEMP شکل زیر را در نظر بگیرید. این شکل به طور نمونه نحوه خواندن از ثبات TCNT1 را نشان میدهد. اعداد داخل دایره نمایشگر شماره سیکل میباشند.

بنابراین برای استفاده از این قابلیت لازم است تا برای خواندن یک ثبات ۱۶ بیتی ابتدا بایت با ارزش کمتر و سپس بایت با ارزش بالاتر خوانده شود.

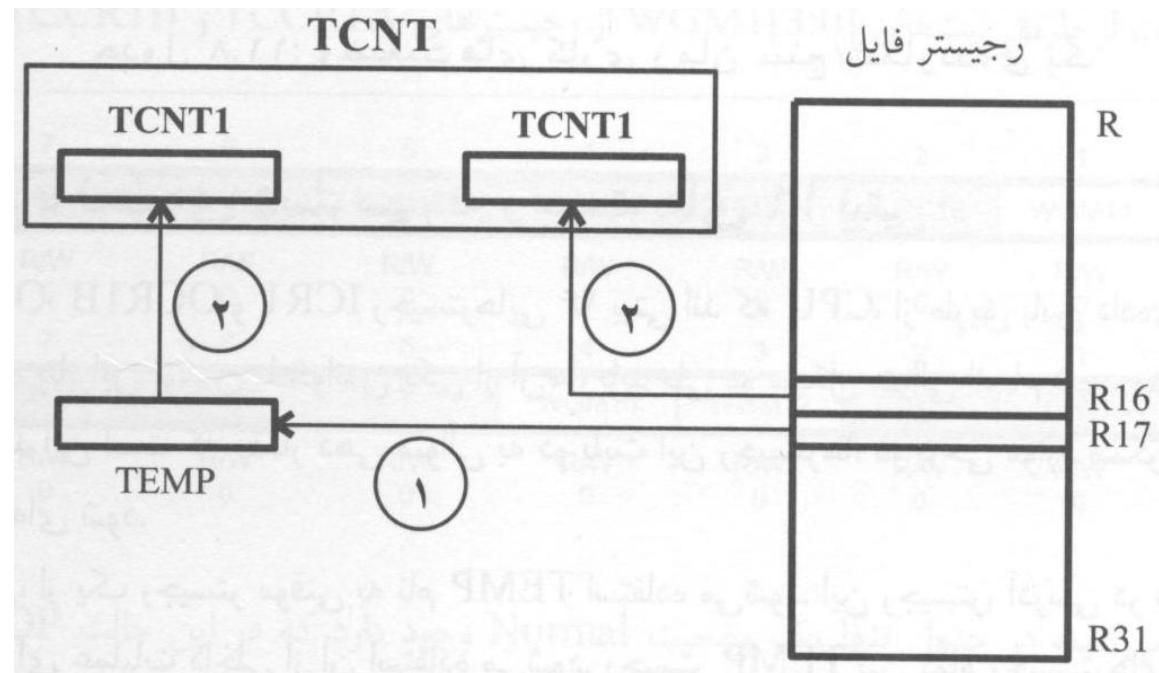


آشنایی با میکروکنترلر AVR بخشن تایمر/کانتر

آشنایی با نحوه دسترسی به ثباتهای ۱۶ بیتی

مطابق با شکل زیر، نوشتمن یک بایت در بالاتر موجب ذخیره شدن آن در ثبات TEMP میشود، حال نوشتمن بایت پایین تر موجب میشود تا دو بایت بالا و پایین به طور همزمان به ثبات ۱۶ بیتی منتقل شود.

در بروز رسانی یک ثبات ۱۶ بیتی ابتدا بایت با ارزش بیشتر نوشته شود.



آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

آشنایی با نحوه دسترسی به ثباتهای ۱۶ بیتی

نکته مهمی که در مورد **TEMP** لازم است مد نظر قرار گیرد این است که این ثبات، بین تمام ثباتهای ۱۶ بیتی مشترک است، بنابراین بین دو سیکل بروز رسانی یک ثبات ۱۶ بیتی، ممکن است وقفه‌ای درخواست شود و در **ISR** آن وقفه، مقدار **TEMP** تغییر کند. این مسئله باعث خطا در نوشتن یا خواندن ثبات میشود. برای جلوگیری از آن میتوان قبل از سیکل اول، فعال ساز عمومی وقفه‌ها را غیر فعال و پس از سیکل دوم آن را فعال نمود:

```
#asm("cli")
// Writing 16 bit Register
#asm("sei")
```

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

آشنایی با نحوه دسترسی به ثباتهای ۱۶ بیتی

در محیط CodeVisionAVR مانند بیشتر کامپایلرهای سطح بالا، عملیات دسترسی به ثباتهای ۱۶ بیتی توسط کامپایلر انجام میشود. به عنوان مثال با نوشتن دستور **TCNT1=0x4455**، کامپایلر ابتدا بایت با ارزش بیشتر و سپس بایت با ارزش کمتر را در محل ثباتهای **TCNT1H** و **TCNT1L** مینویسید.

تنها استثنا این قضیه، ثبات **ICR1** است که در دسترسی به آن، کامپایلر کمکی نکرده و باید بایت بالاتر و پایین تر آن به ترتیب ذکر شده، نوشته و یا خوانده شوند. به عنوان مثال برای نوشتن عدد **0x4455** در ثبات **ICR1** نمیتوان از دستور **ICR1=0x4455** استفاده کرد و باید بایت بالاتر و پایین تر را به این ترتیب مقداردهی نمود:

ICR1H=0X44 ;

ICR1L=0X55 ;

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

شناسایی وضعیت NORMAL در تایمر/کانتر یک

برای انتخاب این وضعیت باید مقدار 0b0000 را به بیتهاي WGM[3:0] مناسب کرد. وضعیت نرمال در تایمر/کانتر یک مشابه نرمال در تایمر/کانتر صفر است با این تفاوت که رشته شمارش ثبات TCNT1 وسیعتر از TCNT0 میباشد. بدین ترتیب که شمارنده از عدد صفر تا ۶۵۵۳۵ شمرده و در لحظه صفر شدن، فلگ TOV1 یک میشود.

COM1A1/COM1B1	COM1A0/COM1B0	وضعیت پین OC1A/OC1B
0	0	غیر فعال (I/O معمولی)
0	1	در تطبیق مقایسه Toggle
1	0	در تطبیق مقایسه Clear
1	1	در تطبیق مقایسه Set

در وضعیت نرمال میتوان از دو واحد مقایسه A و B برای ایجاد وقفه در فواصل مشخص استفاده نمود. در صورت نیاز میتوان با تنظیم بیتهاي COM1A[1:0] و COM1B[1:0] مطابق با جدول بالا به صورت سخت افزاری بر روی پایه های OC1A و OC1B شکل موج تولید کرد.

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

شناسایی اصول استفاده از تایمر/کانتر یک در وضعیت NORMAL

```
#include <mega32.h>
#include <lcd.h>
#include <stdlib.h>
//Alphanumeric LCD Module functions
#asm
    .equ __lcd_port=0x12 ;PORTD
#endifasm

void ini_io();
unsigned char cnt;
unsigned char str_cnt[3];

Interrupt [TIM1_OVF] void timer1_ovf_isr(void)
{
    TCNT1=34286; //Set for 31250 count
    if(cnt<99)
        cnt++;
    else
        cnt=0;
    itoa(cnt,str_cnt); //convert integer to string
    lcd_clear();
    lcd_puts(str_cnt);
}
```

مزیت تایمر/کانتر یک در مقایسه با صفر این

است که شمارنده میتواند رشته طولانی‌تری را

بدون سریز شمارش کند و بدین ترتیب امکان

ایجاد زمانهای طولانی به صورت سخت افزاری

وجود خواهد داشت. اینک میخواهیم یک حلقه

تایmer یک ثانیه طراحی نماییم.

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

شناسایی اصول استفاده از تایمر/کانتر یک در وضعیت NORMAL

```
void main(void)
{
    init_io(0);
    while(1)
    {
    };
}

Void init_io()
{
    // LCD module initialization
    lcd_init(16);
    lcd_putchar('0');

    // Timer/Counter 1 initialization
    TCNT1=34286;    //Set for 31250 count
    TCCR1A=0X00;
    TCCR1B=0X04;    //Prescaler=256
    TIMSK=0X04;      //Enable TOV1 interrupt
    #asm("sei");
}
```

از آنجا که با پیش تقسیم کننده ۲۵۶، هر تیک شمارنده ۳۲ میکروثانیه طول میکشد، بدینهی است که برای سپری شدن یک ثانیه یک میلیون میکروثانیه لازم است شمارنده ۳۱۲۵۰ با افزایش یابد ($1000000/32=31250$). حال کافی است تا ثبات **TCNT1** در هر بار سرریز، با $31250=34286$ بارگذاری شود ($34286-31250=65536$).

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

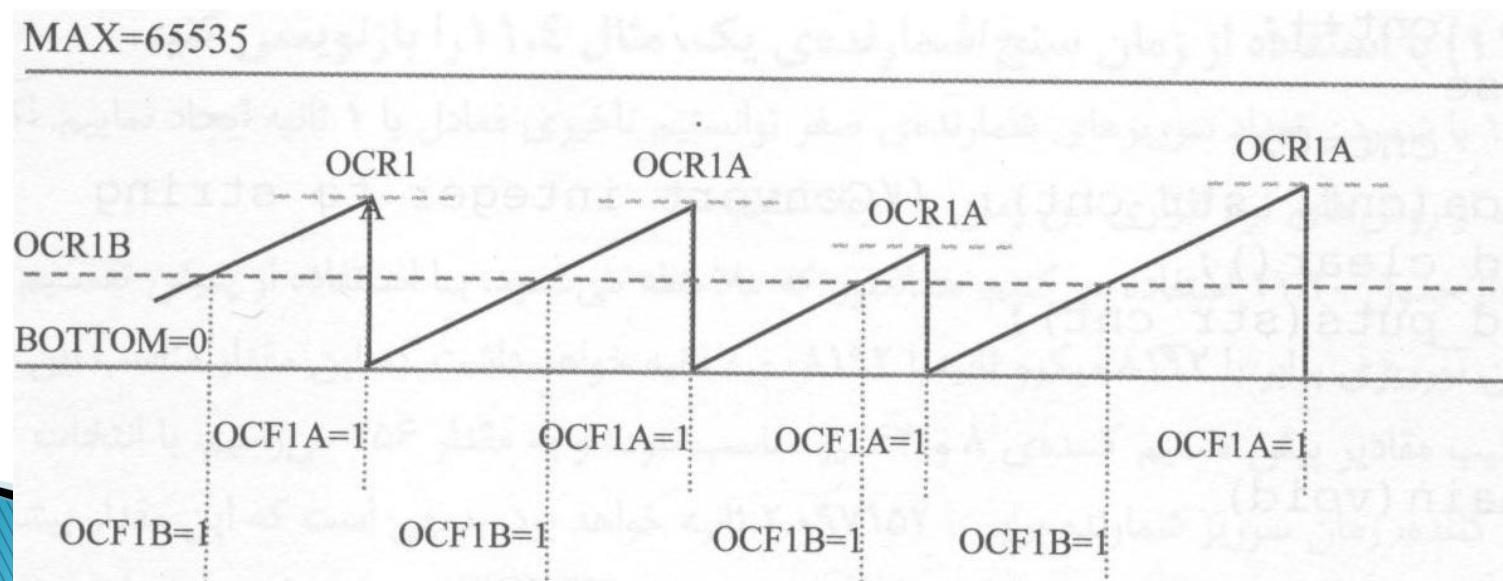
آشنایی با وضعیت های CTC و تفاوت آنها

عملکرد CTC در تایمر/کانتر یک مشابه حالت صفر است با این تفاوت که در اینجا TOP شمارنده توسط ثبات ICR1 تعیین میشود.

در وضعیت ۴ از آنجایی که TOP شمارنده برابر مقدار OCR1A است، مطابق با شکل زیر با رسیدن شمارنده به مقدار OCR1A، ثبات TCNT1 پاک شده و همزمان در سیکل بعدی فلگ OCF1A نیز یک خواهد شد.

در عین حال، مطابق شکل به طور همزمان محتوی ثبات OCR1B نیز با TCNT1 مقایسه شده و در صورت برابری، فلگ OCF1B یک میشود. با یک شدن هر یک از دو فلگ OCF1A و OCF1B در صورتی که فعال ساز وقهه آن فلگ (در ثبات TIMSK) و فعال ساز عمومی وقهه ها یک باشند، وقهه رخ خواهد داد.

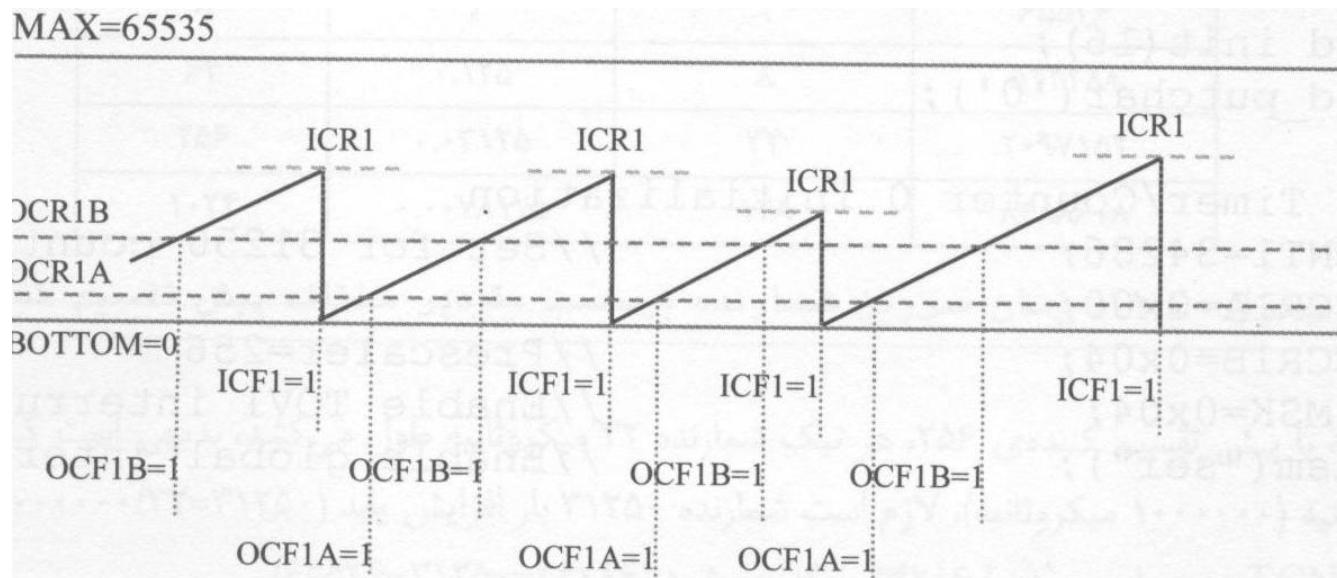
MAX=65535



آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

آشنایی با وضعیت های CTC و تفاوت آنها

در وضعیت ۱۲ شمارنده برابر مقدار ICR1 است، بدین ترتیب مطابق با شکل زیر شمارنده از صفر شروع به شمارش کرده و با رسیدن به مقدار ICR1 مقدار آن صفر شده و همزمان فلگ ICF1 یک میشود. بدین ترتیب مقدار ثباتهای OCR1B و OCR1A در مقدار TOP شمارنده نداشته و تنها لحظه‌ی یک شدن فلگهای OCF1A و OCF1B را تعیین میکنند.



آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

بازنویسی مثال قبلی

```
#include <mega32.h>
#include <lcd.h>
#include <stdlib.h>
//Alphanumeric LCD Module functions
#asm
    .equ __lcd_port=0x12 ;PORTD
#endifasm

void ini_io();
unsigned char cnt;
unsigned char str_cnt[3];

Interrupt [TIM1_COMPA] void timer1_compa_isr(void)
{
    if(cnt<99)
        cnt++;
    else
        cnt=0;
    itoa(cnt,str_cnt); //convert integer to string
    lcd_clear();
    lcd_puts(str_cnt);
}
```

در نمونه قبلی برای ایجاد زمان یک ثانیه، در روتین سرویس وقفه، مقدار اولیه ۳۴۲۸۶ را در ثبات TCNT1 بارگذاری کردیم. اکنون با استفاده از CTC میتوانیم TOP شمارنده را به ۳۱۲۴۹ تغییر دهیم تا شمارنده ۳۱۲۵۰ گام را طی کند. بدین منظور در وضعیت CTC تایмер را با TOP=OCR1A پیکربندی کرده ایم.

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

بازنویسی مثال قبلی

```
void main(void)
{
    init_io(0);
    while(1)
    {
    };
}

Void init_io()
{
    // LCD module initialization
    lcd_init(16);
    lcd_putchar('0');

    // Timer/Counter 1 initialization
    OCR1A=31249;      //Set for 31250 count
    TCCR1A=0X00;      //CTC: TOP=OCR1A
    TCCR1B=0X0C;      //Prescaler=256
    TIMSK=0X10;        //Enable OCF1A interrupt
    #asm("sei");
}
```

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

آشنایی با رو تولید شکل موج در وضعیت های CTC

مشاهده کردید که یک شدن فلگ OCF1A یا OCF1B، فرمانی برای واحد تولید شکل موج صادر میکند. در این لحظه این واحد با توجه به مقادیر بیتهاي [COM1A[1:0] و COM1B[1:0] وضعیت پایه OC1A یا OC1B را تغییر میدهد.

حالات مختلفی که بیتهاي [COM1A[1:0] و COM1B[1:0] میتوانند اختیار کنند مطابق با جدول زیر است:

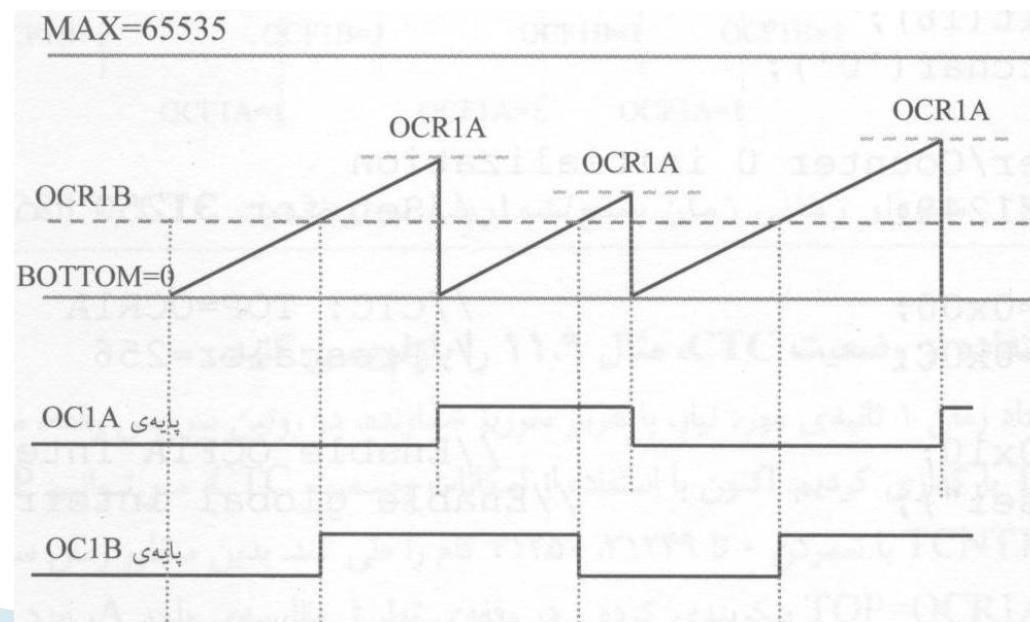
وضعیت پین	OC1A/OC1B	COM1A0/COM1B0	COM1A1/COM1B1
غیر فعال (I/O معمولی)		0	0
در تطبیق مقایسه Toggle		1	0
در تطبیق مقایسه Clear		0	1
در تطبیق مقایسه Set		1	1

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

آشنایی با رو تولید شکل موج در وضعیت های CTC

با توجه به مطالب بیان شده، میتوان نحوه ایجاد شکل موج در وضعیت‌های CTC را به طور مجزا بررسی نمود.

در وضعیت ۴، TOP شمارنده برابر با مقدار OCR1A بوده و در نتیجه فعال نمودن واحد تولید شکل موج، باعث تغییر وضعیت پایه OC1A یا OC1B میشود. فرض کنید بیتهای COM1A[1:0] در وضعیتی باشند که در لحظه تطبیق مقایسه، پایه های OC1A و OC1B، تاگل (معکوس) شوند. همانند شکل زیر



آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

فرکانس شکل موج ایجاد شده

فرکانس شکل موج ایجاد شده از رابطه زیر بدست می‌آید:

$$f_{OC1A/OC1B} = \frac{f_{CLK}}{2N(1 + OCR1A)}$$

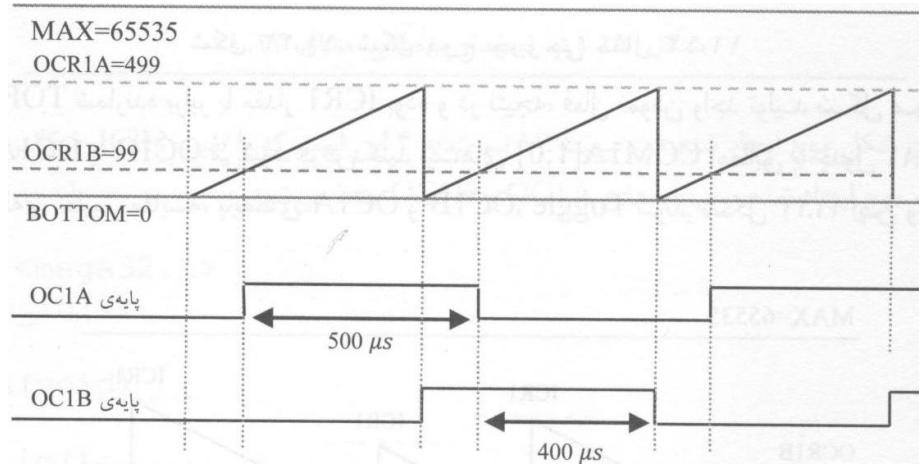
با توجه به این رابطه، بالاترین فرکانس پایه $OCR1A$ یا $OCR1B$ زمانی خواهد بود که مقدار $OCR1A$ صفر بوده و پیش تقسیم کننده نیز برابر یک باشد. بدین ترتیب شکل موجی با نصف فرکانس کلاک سیستم ایجاد می‌شود.

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

مثال ایجاد شکل موج

با استفاده از وضعیت CTC، دو موج مربعی با فرکانس یک کیلوهرتز و اختلاف زمانی ۴۰۰ میکروثانیه بر روی پایه های OC1A و OC1B ایجاد نمایید (کلاک سیستم ۸ مگاهرتز است).

اگر مقدار پیش تقسیم کننده ۸ باشد، کلاک شمارنده



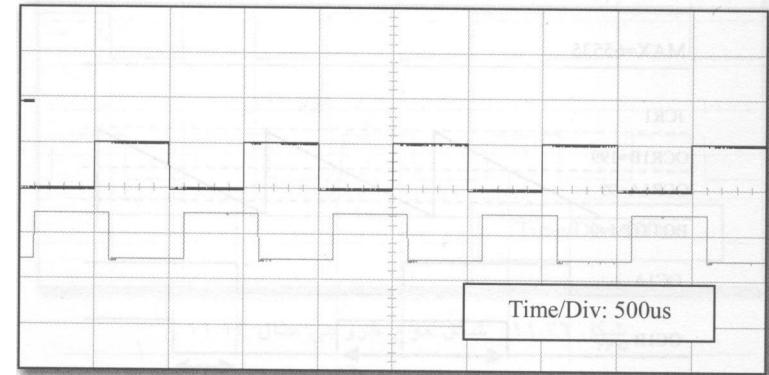
برابر یک مگاهرتز بوده و هر تیک آن یک میکروثانیه خواهد بود. بدین ترتیب در صورتی که TOP شمارنده برابر ۴۹۹ باشد، دوره تناوب شکل موجهای ایجاد شده بر روی OC1A و OC1B یک میلی ثانیه و فرکانس آنها یک کیلوهرتز است. همچنین مطابق با شکل زیر، برای ایجاد اختلاف زمانی ۴۰۰ میکروثانیه کافی است تا OCR1B با ۹۹ بارگذاری شود.

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

مثال ایجاد شکل موج

```
#include <mega32.h>
void init_io();
void main(void)
{
    init_io(0);
    while(1)
    {
    };
}

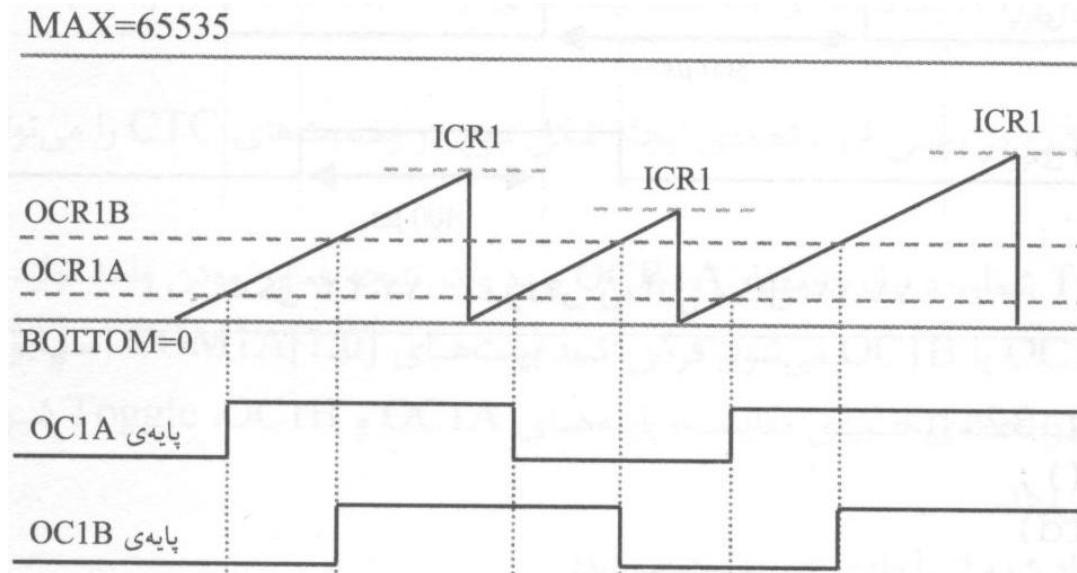
Void init_io()
{
    DDRD=0x30;          //Output OC1A and OC1B
    OCR1A=499;          //Set for 1000 count
    OCR1B=99;           //TO 100us delay
    TCCR1A=0X50;         //CTC: TOP=OCR1A
    TCCR1B=0X0A;         //Prescaler=8
                           //OC1A output: Toggle
                           //OC1B output: Toggle
}
```



آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

آشنایی با رو تولید شکل موج در وضعیت های CTC

در وضعیت ۱۲، TOP شمارنده برابر با مقدار ICR1 بوده و در نتیجه فعال نمودن واحد تولید شکل موج، باعث تغییر وضعیت پایه OC1B یا OC1A میشود. فرض کنید بیتهای COM1A[1:0] در وضعیتی باشند که در لحظه تطبیق مقایسه، پایه های OC1A و OC1B، تاگل (معکوس) شوند. همانند شکل زیر



آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

فرکانس شکل موج ایجاد شده

فرکانس شکل موج ایجاد شده از رابطه زیر بدست می آید:

$$f_{OC1A/OC1B} = \frac{f_{CLK}}{2N(1 + ICR1)}$$

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

آشنایی با وضعیت های PWM در تایмер/کانتر یک

قابلیت های عمدۀ تایمر/کانتر یک، ناشی از وضعیت های PWM قابل انعطاف آن است. همانطور که میدانید یک سیگنال PWM دو مشخصه اساسی دارد: عرض پالس و فرکانس.

- در مورد عرض پالس، به لحاظ ۱۶ بیتی بودن تایمر/کانتر یک، موج PWM ایجاد شده رزولوشن نسبتا بالایی دارد.
- فرکانس سیگنال PWM تولید شده توسط تایمر/کانتر یک را میتوان به سادگی و به شکلی پیوسته تغییر داد

تایمر/کانتر یک دارای ۱۲ وضعیت PWM است. از این ۱۲ وضعیت، ۵ مورد مربوط به Fast PWM، ۵ مورد مربوط به Phase & Frequency PWM و دو مورد آن مربوط به یک وضعیت جدید به نام Phase Correct PWM است.

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

شناسایی وضعيت های Fast PWM در تایمر/کانتر یک

در صورتی که پنج وضعیت مربوط به Fast PWM را از جدول کلی قبلی استخراج کنیم جدول زیر نتیجه میگردد.

شماره	WGM13	WGM12	WGM11	WGM10	وضعیت کاری	TOP	بروزرسانی OCR1x	لحظه‌ی یک شدن پرچم TOV1
5	0	1	0	1	Fast PWM, 8-bit	0x00FF	BOTTOM	TOP
6	0	1	1	0	Fast PWM, 9-bit	0x01FF	BOTTOM	TOP
7	0	1	1	1	Fast PWM, 10-bit	0x03FF	BOTTOM	TOP
14	1	1	1	0	Fast PWM	ICR1	BOTTOM	TOP
15	1	1	1	1	Fast PWM	OCR1A	BOTTOM	TOP

با کمی دقت در این جدول میتوان دریافت که اختلاف این پنج وضعیت کاری، ناشی از مقدار TOP آنهاست که هر یک را مورد بررسی قرار میدهیم.

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

Fast PWM, 8-bit

در این حالت TOP شمارنده برابر با 0x00FF خواهد بود و مشابه با تایمر/کانتر صفر عمل میکند. بدین ترتیب رشته شمارش ثبات TCNT1 محدود به اعداد صفر تا ۲۵۵ شده و در لحظه سرریز شدن شمارنده، فلگ TOV1 یک میشود. فرکانس سیگنال ایجاد شده از رابطه زیر قابل محاسبه است:

$$Frequency = \frac{f_{CLK}}{N \times 256}$$

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

Fast PWM, 9-bit

در این حالت TOP شمارنده برابر با 0x01FF خواهد بود. بدین ترتیب رشته شمارش ثبات TCNT1 محدود به اعداد صفر تا ۵۱۱ شده و در لحظه سرریز شدن شمارنده، فلگ TOV1 یک میشود. فرکانس سیگنال ایجاد شده از رابطه زیر قابل محاسبه است:

$$Frequency = \frac{f_{CLK}}{N \times 512}$$

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

Fast PWM, 10-bit

در این حالت TOP شمارنده برابر با $0x03FF$ خواهد. بدین ترتیب رشته شمارش ثبات TCNT1 محدود به اعداد صفر تا 1023 شده و در لحظه سرریز شدن شمارنده، فلگ TOV1 یک میشود. فرکانس سیگنال ایجاد شده از رابطه زیر قابل محاسبه است:

$$Frequency = \frac{f_{CLK}}{N \times 1024}$$

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

Fast PWM, TOP=ICR1

در این حالت TOP شمارنده برابر با مقدار ثبات ICR1 خواهد بود و رزولوشن سیگنال ایجاد شده از رابطه زیر بدست می‌آید:

$$resolution = \frac{\log(ICR1 + 1)}{\log(2)}$$

بدین ترتیب، رشته شمارش ثبات TCNT1 محدود به اعداد صفر تا ICR1 شده و در لحظه سرریز شدن شمارنده، فلگ TOV1 یک میشود. در این وضعیت فرکانس سیگنال ایجاد شده از رابطه زیر قابل محاسبه است:

$$Frequency = \frac{f_{CLK}}{N \times (ICR1 + 1)}$$

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

Fast PWM, TOP=OCR1A

در این حالت **TOP** شمارنده برابر با مقدار ثبات **OCR1A** خواهد بود و رزولوشن سیگنال ایجاد شده از رابطه زیر بدست می‌آید:

$$resolution = \frac{\log(OCR1A + 1)}{\log(2)}$$

بدین ترتیب، رشته شمارش ثبات **TCNT1** محدود به اعداد صفر تا **OCR1A** شده و در لحظه سرریز شدن شمارنده، فلگ **TOV1** یک میشود. در این وضعیت فرکانس سیگنال ایجاد شده از رابطه زیر قابل محاسبه است:

$$Frequency = \frac{f_{CLK}}{N \times (OCR1A + 1)}$$

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

شناسایی وضعیت های Fast PWM در تایمر/کانتر یک

حال ممکن است با توجه به جامعیت دو وضعیت آخر، لزوم سه وضعیت نخست زیر سوال برود. به عبارت دیگر ممکن است این سوال مطرح شود:

«با توجه به اینک میتوان در دو وضعیت Fast PWM, TOP=OCR1A و Fast PWM, TOP=ICR1 میتوان شمارند را اعداد 0xFF، 0x1FF و 0x3FF تعیین کرد، چه لزومی دارد تا برای این سه مورد، وضعیتهای جداگانه وجود داشته باشد؟»

پاسخ این است که استفاده از وضعیت Fast PWM, TOP=ICR1 موجب غیرفعال شدن، قابلیت شکار میشود. همچنین استفاده از وضعیت Fast PWM, TOP=OCR1A موجب غیرفعال شدن خروجی PWM واحد A شده به بیان دیگر نمیتوان موج PWM بر روی پایه OCR1A تولید نمود.

همانطور که از روابط مشخص است وضعیت کاری Fast PWM هشت، نه و ده بیتی در رزولوشن و فرکانس با هم متفاوتند.

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

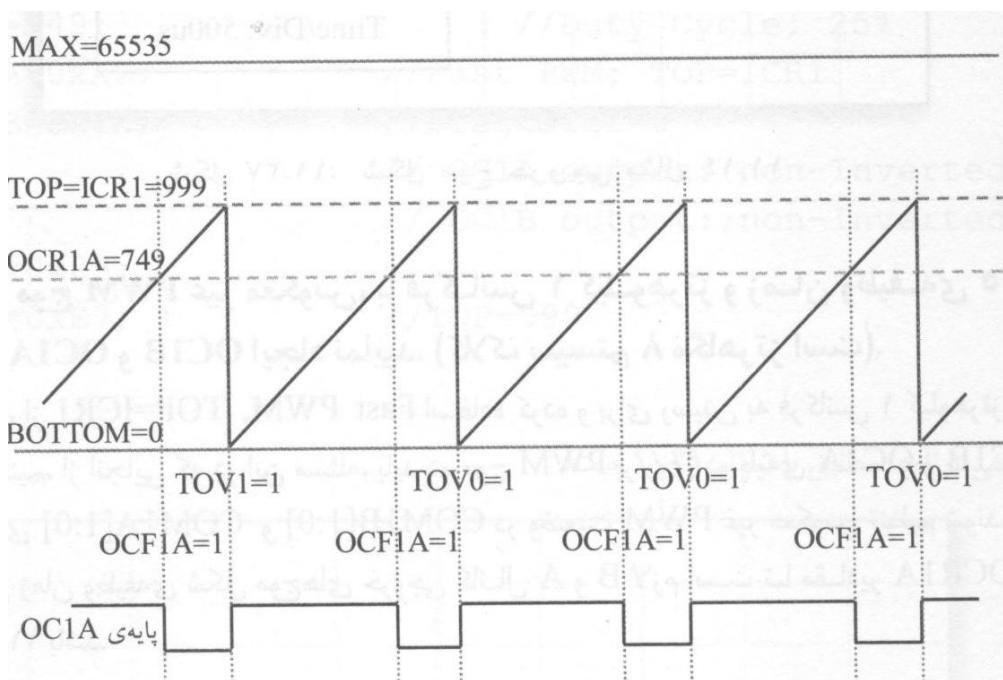
آشنایی با روش تنظیم واحد تولید شکل موج

مشابه با وضعیت CTC، چنانچه بخواهیم واحد تولید موج را فعال نماییم تا به صورت خودکار شکل موجی را بر روی پایه های OC1A یا OC1B ایجاد نماید، لازم است تا بیتهاي [COM1A[1:0] و [COM1B[1:0] تنظیم شوند. حالتهاي مختلفی را که این بیتها شامل میشوند در جدول زیر مشاهده مینمایید.

COM1A1/ COM1B1	COM1A0/ COM1B0	وضعیت پایه های [COM1B[1:0] یا [COM1A[1:0]
0	0	غیر فعال (I/O معمولی)
0	1	در صورتی که [WGM1[3:0] برابر با ۱۵ باشد: پایه OC1A در لحظه‌ی تطبیق مقایسه، OC1B می‌شود و پایه OC1B غیر فعال است. برای سایر وضعیت‌ها، پایه‌های OC1A و OC1B، عملکرد تولید شکل موج نداشته و I/O معمولی‌اند.
1	0	پاک کردن OC1B یا OC1A در تطبیق مقایسه و یک کردن آن در TOP (غیر PWM معکوس)
1	1	یک کردن OC1B یا OC1A در تطبیق مقایسه و پاک کردن آن در TOP (PWM معکوس)

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

مثال: ایجاد شکل موج غیرمعکوس با فرکانس ۱KHz و زمان وظیفه ۷۵٪ بر روی پایه OC1A



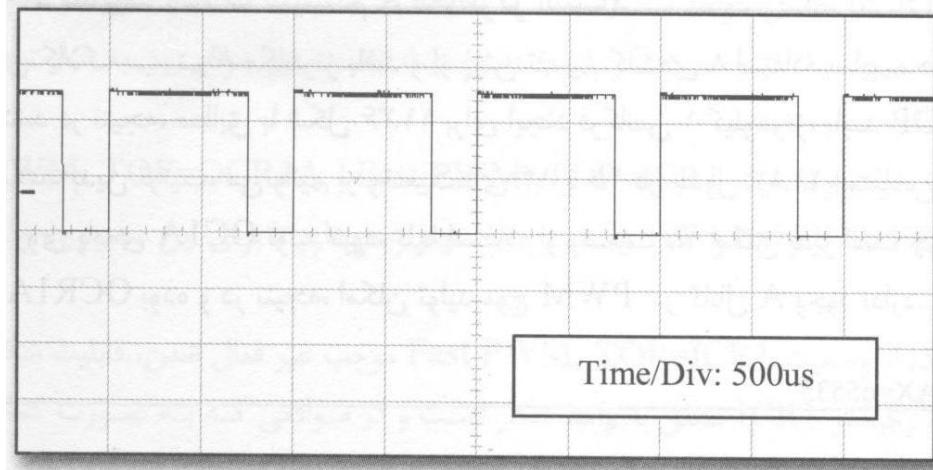
در صورتی که فرکانس کلاک سیستم ۸ مگاهرتز باشد و از پیش تقسیم کننده ۸ استفاده نماییم، هر تیک شمارنده یک میکروثانیه طول میکشد. در نتیجه برای ایجاد فرکانس یک کیلوهرتز باید TOP شمارنده برابر با ۹۹۹ (یا 0x03E7) باشد. بدین ترتیب میتوانیم از وضعیتهای ۱۴ یا ۱۵ استفاده نماییم. اما از آنجا که شکل موج باید بر روی پایه OC1A تولید شود، تنها استفاده از وضعیت ۱۵ امکان پذیر است چرا که در وضعیت ۱۴، TOP شمارنده برابر OCR1A بوده و در نتیجه تولید موج PWM در کanal A وجود ندارد.

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

مثال: ایجاد شکل موج غیرمعکوس با فرکانس 1KHz و زمان وظیفه ۷۵٪ بر روی پایه OC1A

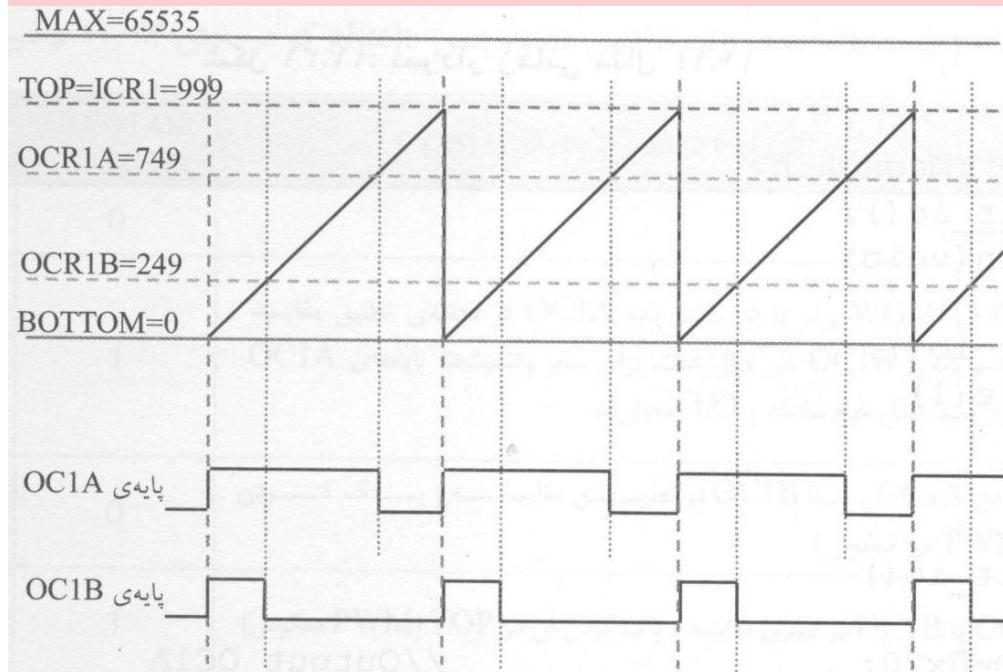
```
#include <mega32.h>
void init_io();
void main(void)
{
    init_io();
    while(1)
    {
    };
}

Void init_io()
{
    DDRD=0x20;          //Output OC1A
    // Timer/counter 1 initialization
    OCR1A=749;          //Duty Cycle: 75%
    TCCR1A=0X82;        //Fast PWM:CTC: TOP=ICR1
    TCCR1B=0X1A;        //Prescaler=8
                           //OC1A output: Non-Inverted
    ICR1H=0x03;
    ICR1L=0xE7;        // TOP=999
}
```



آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

مثال: ایجاد دو سیگنال PWM غیرمعکوس، با فرکانس ۱KHz و زمان وظیفه ۷۵٪ و ۲۵٪ بر روی پایه های OC1B و OC1A



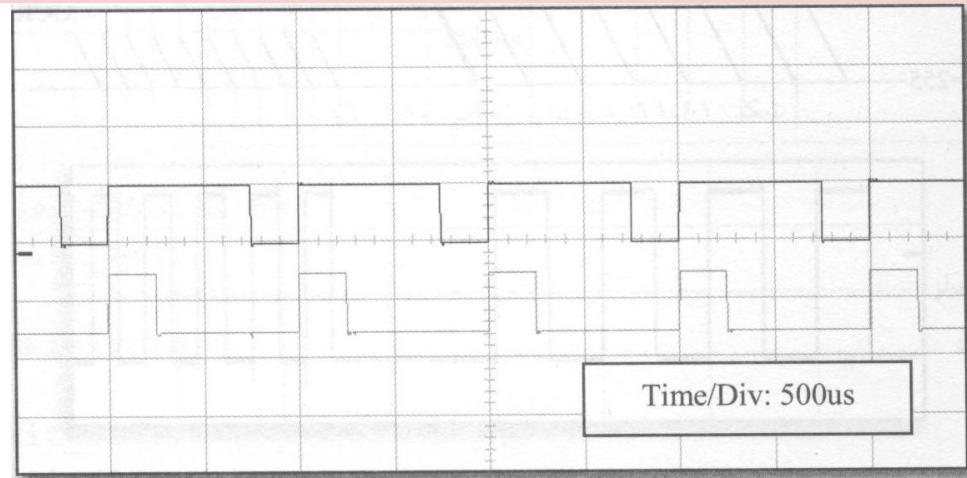
مشابه مثال قبلی، از وضعیت Fast PWM, Top=ICR1 استفاده کرده و برای رسیدن به فرکانس یک کیلوهرتز، ICR1 را با ۹۹۹ بارگذاری میکنیم. از آنجا که در این مثال OC1B دو سیگنال PWM بر روی دو پایه OC1A و OC1B تولید شود، لازم است تا بیتهاي COM1A[1:0] و COM1B[1:0] در وضعیت PWM غیرمعکوس تنظیم گردند.

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

مثال: ایجاد دو سیگنال PWM غیرمعکوس، با فرکانس ۱KHz و زمان وظیفه ۷۵٪ و ۲۵٪ بر روی پایه های OC1B و OC1A

```
#include <mega32.h>
void init_io();
void main(void)
{
    init_io();
    while(1)
    {
    };
}

Void init_io()
{
    DDRD=0x30;           //Output OC1A and OC1B
    // Timer/counter 1 initialization
    OCR1A=749;          //Duty Cycle: 75%
    OCR1B=249;          // Duty Cycle: 25%
    TCCR1A=0X82;        //Fast PWM:CTC: TOP=ICR1
    TCCR1B=0X1A;         //Prescaler=8
                    //OC1A output: Non-Inverted
                    //OC1B output: Non-Inverted
    ICR1H=0x03;
    ICR1L=0xE7;         // TOP=999
}
```

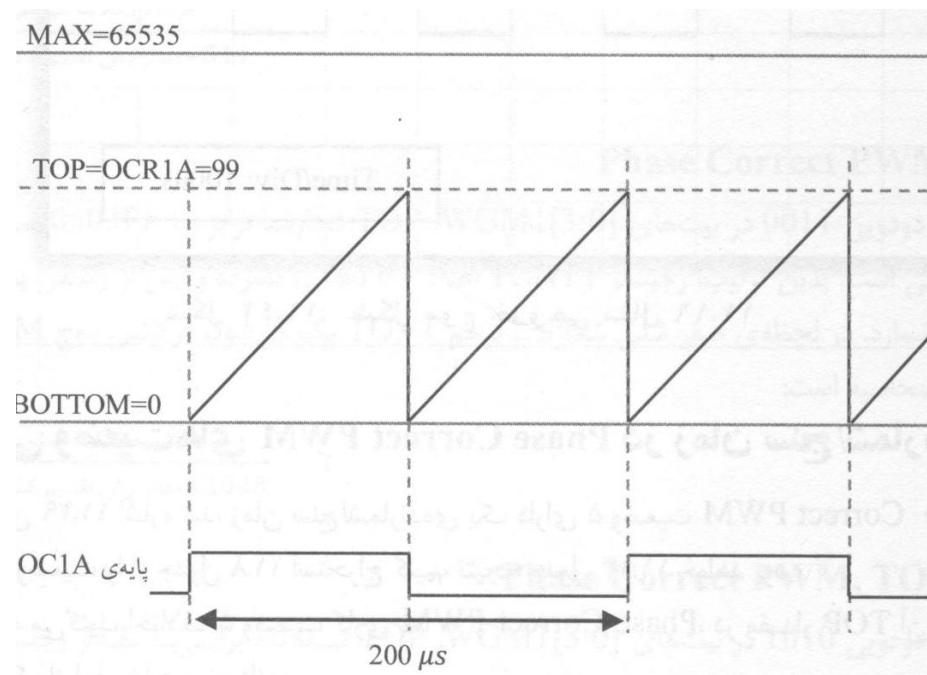


آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

مثال: ایجاد یک موج مربعی با فرکانس ۵ کیلوهرتز با شمارنده یک در وضعیت Fast PWM

برای ایجاد این شکل موج، مطابق با دستورالعمل ارائه شده، شمارنده را در وضعیت ۱۵ تنظیم کرده ($WGM1[3:0]=15$) و خروجی واحد مقایسه را به صورت Toggle تنظیم میکنیم. حال کافی است مقدار OCR1A برابر

با ۹۹ باشد.

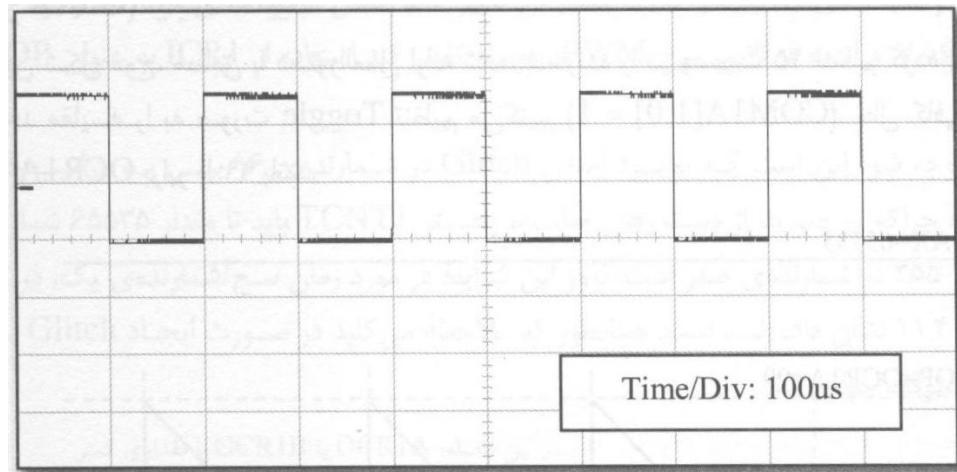


آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

مثال: ایجاد یک موج مربعی با فرکانس ۵ کیلوهرتز با شمارنده یک در وضعیت Fast PWM

```
#include <mega32.h>
void init_io();
void main(void)
{
    init_io(0);
    while(1)
    {
    };
}

Void init_io()
{
    DDRD=0x20;      //Output OC1A
    // Timer/counter 1 initialization
    OCR1A=99;
    TCCR1A=0X43;   //Fast PWM: TOP=OCR1A
    TCCR1B=0X1A;   //Prescaler=8
                    //OC1A output: Non-Inverted
}
```



آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

شناسایی وضعیتهای Phase Correct PWM در تایمر/کانتر یک

همانطور که قبلاً اشاره شد، تایمر/کانتر یک دارای ۵ وضعیت Phase Correct PWM است. در جدول زیر این پنج وضعیت را مشاهده میکنید.

شماره	WGM1 3	WGM1 2	WGM1 1	WGM 10	وضعیت کاری	TOP	بروزرسانی OCR1x	لحظه‌ی یک شدن TOV1 پرچم
1	0	0	0	1	PWM, Phase Correct, 8-bit	0x00FF	TOP	BOTTOM
2	0	0	1	0	PWM, Phase Correct, 9-bit	0x01FF	TOP	BOTTOM
3	0	0	1	1	PWM, Phase Correct, 10-bit	0x03FF	TOP	BOTTOM
10	1	0	1	0	PWM, Phase Correct	ICR1	TOP	BOTTOM
11	1	0	1	1	PWM, Phase Correct	OCR1A	TOP	BOTTOM

همانطور که ملاحظه میکند، اختلاف ۵ وضعیت کاری، در مقدار TOP آنهاست که در ادامه به بررسی هر یک از این موارد میپردازیم.

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

Phase Correct PWM, 8-bit

در این حالت TOP شمارنده برابر با 0x00FF خواهد بود و مشابه با تایمر/کانتر صفر عمل میکند. بدین ترتیب ثبات اعداد صفر تا ۲۵۵ را شمرده و پس از رسیدن به TOP به صورت نزولی تا صفر میشمارد. در لحظه صفر شدن شمارنده، شده و فلگ TOV1 یک میشود. فرکانس سیگنال ایجاد شده از رابطه زیر قابل محاسبه است:

$$Frequency = \frac{f_{CLK}}{2 \times N \times 255}$$

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

Phase Correct PWM, 9-bit

در این حالت TOP شمارنده برابر با 0x01FF خواهد بود و مشابه با تایمر/کانتر صفر عمل میکند. بدین ترتیب ثبات اعداد صفر تا ۵۱۱ را شمرده و پس از رسیدن به TOP به صورت نزولی تا صفر میشمارد. در لحظه صفر شدن شمارنده، شده و فلگ TOV1 یک میشود. فرکانس سیگنال ایجاد شده از رابطه زیر قابل محاسبه است:

$$Frequency = \frac{f_{CLK}}{2 \times N \times 511}$$

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

Phase Correct PWM, 10-bit

در این حالت TOP شمارنده برابر با 0x03FF خواهد بود و مشابه با تایمر/کانتر صفر عمل میکند. بدین ترتیب ثبات اعداد صفر تا ۱۰۲۳ را شمرده و پس از رسیدن به TOP به صورت نزولی تا صفر میشمارد. در لحظه صفر شدن شمارنده، شده و فلگ TOV1 یک میشود. فرکانس سیگنال ایجاد شده از رابطه زیر قابل محاسبه است:

$$Frequency = \frac{f_{CLK}}{2 \times N \times 1023}$$

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

Phase Correct PWM, TOP=ICR1

در این حالت TOP شمارنده برابر با مقدار ثبات ICR1 خواهد بود و رزولوشن سیگنال ایجاد شده از رابطه زیر بدست می‌آید:

$$resolution = \frac{\log(ICR1 + 1)}{\log(2)}$$

بدین ترتیب، رشته شمارش ثبات ICR1 اعداد صفر تا TCNT1 را شمرده و پس از رسیدن به مقدار ICR1 به صورت نزولی تا صفر می‌شمارد. در لحظه صفر شدن شمارنده، فلگ TOV1 یک می‌شود. فرکانس سیگنال ایجاد شده از رابطه زیر قابل محاسبه است:

$$Frequency = \frac{f_{CLK}}{2 \times N \times ICR1}$$

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

Phase Correct PWM, TOP=OCR1A

در این حالت TOP شمارنده برابر با مقدار ثبات OCR1A شده و رزولوشن سیگنال ایجاد شده از رابطه زیر بدست می‌آید:

$$resolution = \frac{\log(OCR1A + 1)}{\log(2)}$$

بدین ترتیب، رشته شمارش ثبات OCR1A اعداد صفر تا TCNT1 را شمرده و پس از رسیدن به مقدار OCR1A، به صورت نزولی تا صفر می‌شمارد. در لحظه صفر شدن شمارنده، فلگ TOV1 یک می‌شود. فرکанс سیگنال ایجاد شده از رابطه زیر قابل محاسبه است:

$$Frequency = \frac{f_{CLK}}{2 \times N \times OCR1A}$$

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

آشنایی با روش تنظیم واحد تولید شکل موج

مشابه با وضعیت Fast PWM، چنانچه بخواهیم واحد تولید موج را فعال نماییم تا به صورت خودکار شکل موجی را بر روی پایه های OC1B یا OC1A ایجاد نماید، لازم است تا بیتهاي COM1A[1:0] و COM1B[1:0] تنظیم شوند. حالتهاي مختلفی را که این بیتها شامل میشوند در جدول زیر مشاهده مینمایید.

COM1A1/ COM1B1	COM1A0/ COM1B0	وضعیت پایه های COM1B[1:0] یا COM1A[1:0]
0	0	غیر فعال (I/O معمولی)
0	1	در صورتی که WGM1[3:0] برابر با ۱۱ باشد: پایه OC1A در لحظه‌ی تطبیق مقایسه، OC1B می‌شود و پایه‌ی OC1B غیر فعال است. برای سایر وضعیت‌ها، پایه‌های OC1A، عملکرد تولید شکل موج نداشته و I/O معمولی آند. (حالت خاص)
1	0	پاک کردن OC1A یا OC1B در تطبیق مقایسه در شمارش صعودی و یک کردن آن در تطبیق مقایسه در شمارش نزولی (PWM غیر معکوس)
1	1	یک کردن OC1A یا OC1B در تطبیق مقایسه در شمارش صعودی و پاک کردن آن در تطبیق مقایسه در شمارش نزولی (PWM معکوس)

آشنایی با میکروکنترلر AVR بخشن تایмер/کانتر

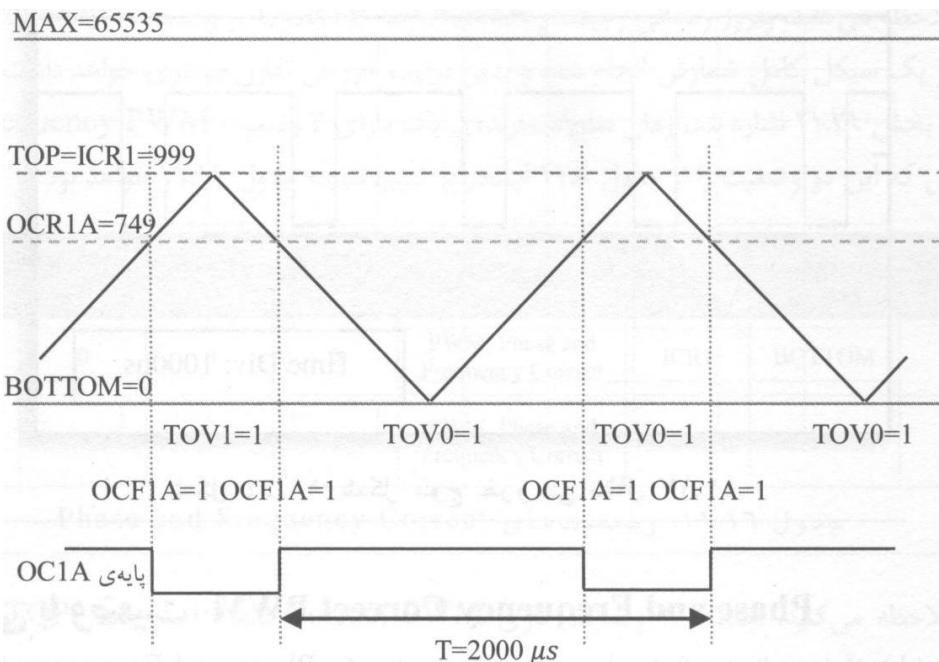
آشنایی با روش تولید شکل موج در وضیت Phase Correct PWM

برای تولید شکل موج، در اولین گام لازم است تا وضعیت مناسب برای سیگنال PWM مورد نظر را انتخاب نمود. پس از آن میتوان با تنظیم بیت‌های [COM1B[1:0] و [COM1A[1:0]] نوع شکل سیگنال خروجی (معکوس، غیرمعکوس و یا حالت خاص را انتخاب نمود.

در اسلاید بعد مثالی را بررسی می‌کنیم.

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

مثال: ایجاد سیگنال PWM غیرمعکوس با فرکانس ۵۰۰ هرتز و زمان وظیفه ۷۵٪ بروی پایه OC1A



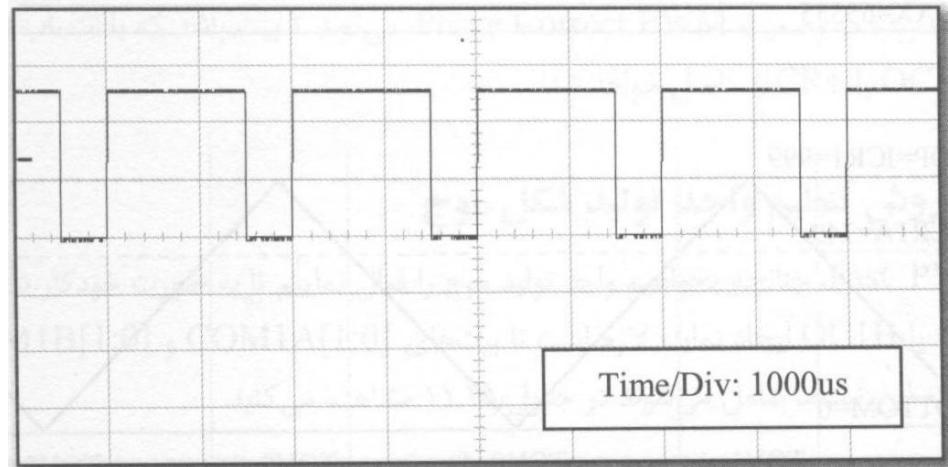
در صورتی که فرکانس کلاک سیستم ۸ مگاهرتز باشد و از پیش تقسیم کننده ۸ استفاده نماییم، هر تیک شمارنده یک میکروثانیه طول میکشد. در نتیجه برای ایجاد فرکانس ۵۰۰ هرتز باید TOP شمارنده برابر با ۹۹۹ (یا ۰x03E7) باشد. بدین ترتیب میتوانیم از وضعیتهای ۱۰ یا ۱۱ استفاده نماییم. اما از آنجا که شکل موج باید بر روی پایه OC1A تولید شود، تنها استفاده از وضعیت ۱۰ امکان پذیر است چرا که در وضعیت ۱۱، TOP شمارنده برابر OCR1A بوده و در نتیجه تولید موج PWM در کanal A وجود ندارد.

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

مثال: ایجاد سیگنال PWM غیرمعکوس با فرکانس ۵۰۰ هرتز و زمان وظیفه ۷۵٪ بروی پایه OC1A

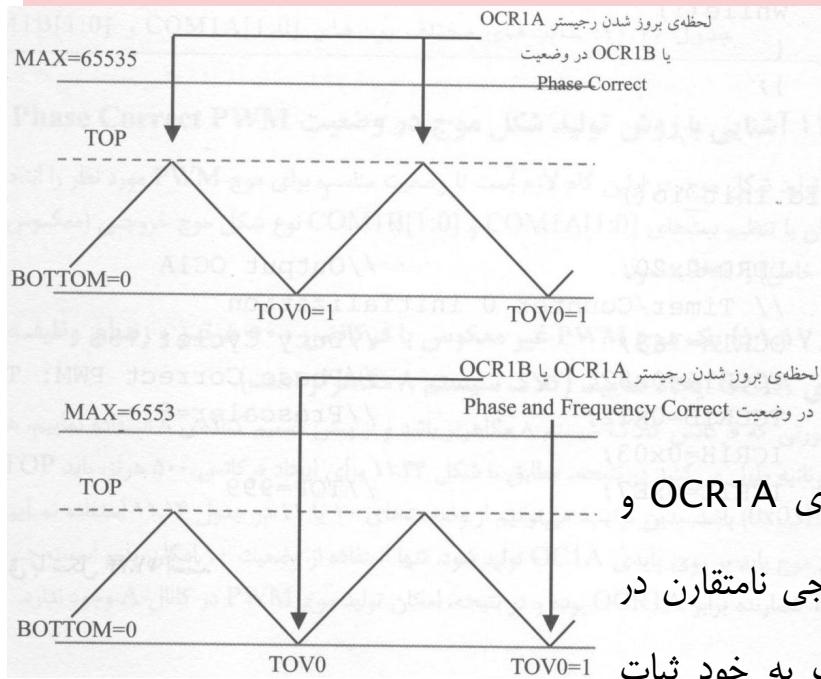
```
#include <mega32.h>
void init_io();
void main(void)
{
    init_io();
    while(1)
    {
    };
}

Void init_io()
{
    DDRD=0x20;          //Output OC1A
    // Timer/counter 1 initialization
    OCR1A=749;         //Duty Cycle: 75%
    TCCR1A=0X82;        //Phase Correct PWM:CTC: TOP=ICR1
    TCCR1B=0X12;        //Prescaler=8
    ICR1H=0x03;
    ICR1L=0xE7;        // TOP=999
}
```



آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

آشنایی با وضعیت Phase and Frequency Correct PWM



این یک وضعیت جدید است که تنها مخصوص تایمر ۱۶ بیتی است و بسیار شبیه به اس Phase Correct PWM این تفاوت که لحظه انتقال مقدار بافر مضاعف به ثبات بافر متفاوت است.

همانطور که قبلاً گفته شد در وضعیتهاي PWM، ثباتهای OCR1A و OCR1B دارای بافر مضاعف میباشند، این مسئله از ایجاد خروجی نامتقارن در خروجی جلوگیری میکند. اما لحظه ای که مقدار بافر مضاعف به خود ثبات منتقل میشود در وضعیتهاي Phase and Phase Correct PWM با یکدیگر متفاوت است. این مسئله در شکل نشان داده شده است.

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

آشنایی با وضعیت PWM

تایмер/کانتر یک دارای دو وضعیت Phase and Frequency PWM است که در جدول زیر آمده اند.

شماره	WGM1 3	WGM1 2	WGM1 1	WGM1 0	وضعیت کاری	TOP	بروزرسانی OCR1x	لحظه‌ی یک شدن TOV1 پرچم
8	1	0	0	0	PWM, Phase and Frequency Correct	ICR1	BOTTOM	BOTTOM
9	1	0	0	1	PWM, Phase and Frequency Correct	OCR1A	BOTTOM	BOTTOM

همانطور که ملاحظه میکنید، اختلاف دو وضعیت کاری در مقدار TOP آنهاست که در ادامه به بررسی هر یک از این موارد میپردازیم.

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

Phase and Frequency Correct PWM, TOP=ICR1

در این حالت TOP شمارنده برابر با مقدار ثبات ICR1 شده و رزولوشن سیگنال PWM از رابطه زیر به دست می‌آید:

$$resolution = \frac{\log(ICR1+1)}{\log(2)}$$

بدین ترتیب، ثبات TCNT1 اعداد صفر تا ICR1 را شمرده و پس از رسیدن به مقدار ICR1، به صورت نزولی تا صفر می‌شمارد. در لحظه صفر شدن شمارنده، فلگ TOV1 یک می‌شود. فرکانس سیگنال PWM ایجاد شده از رابطه زیر قابل محاسبه است:

$$Frequency = \frac{f_{CLK}}{2 \times N \times ICR1}$$

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

Phase and Frequency Correct PWM, TOP=OCR1A

در این حالت TOP شمارنده برابر با مقدار ثبات OCR1A شده و رزولوشن سیگنال PWM از رابطه زیر به دست

می‌آید:

$$resolution = \frac{\log(OCR1A + 1)}{\log(2)}$$

بدین ترتیب، ثبات TCNT1 اعداد صفر تا OCR1A را شمرده و پس از رسیدن به مقدار OCR1A، به صورت نزولی تا صفر می‌شمارد. در لحظه صفر شدن شمارنده، فلگ TOV1 یک می‌شود. فرکانس سیگنال PWM ایجاد شده از رابطه زیر قابل محاسبه است:

$$Frequency = \frac{f_{CLK}}{2 \times N \times OCR1A}$$

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

آشنایی با روش تنظیم واحد تولید شکل موج

چنانچه بخواهیم واحد تولید موج را فعال نماییم تا به صورت خودکار شکل موجی را بر روی پایه های OC1A یا OC1B ایجاد نماید، لازم است تا بیتهاي [COM1A[1:0] و COM1B[1:0]] تنظیم شوند. حالتهاي مختلفی را که این بیتها شامل میشوند در جدول زیر مشاهده مینمایید.

WGM1[3:0] OC1A/OC1B	COM1A[0] COM1B[0]	وضعیت پایه های [COM1B[1:0] یا COM1A[1:0]]
0	0	غیر فعال (I/O معمولی)
0	1	در صورتی که WGM1[3:0] برابر با ۹ باشد: پایه OC1A در لحظه‌ی تطبیق مقایسه، Toggle می‌شود و پایه‌ی OC1B غیر فعال است. برای سایر وضعیت‌ها، پایه‌های OC1A و OC1B، عملکرد تولید شکل موج نداشته و I/O معمولی‌اند.
1	0	پاک کردن OC1B یا OC1A در تطبیق مقایسه در لبه‌ی شمارش صعودی و یک کردن آن در لبه‌ی شمارش نزولی (PWM غیر معکوس)
1	1	یک کردن OC1B یا OC1A در تطبیق مقایسه در لبه‌ی شمارش صعودی و پاک کردن آن در لبه‌ی شمارش نزولی (PWM معکوس)

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

مثال: ایجاد سیگنال PWM غیرمعکوس با فرکانس ۵۰۰ هرتز و زمان وظیفه ۷۵٪ بروی پایه OC1A

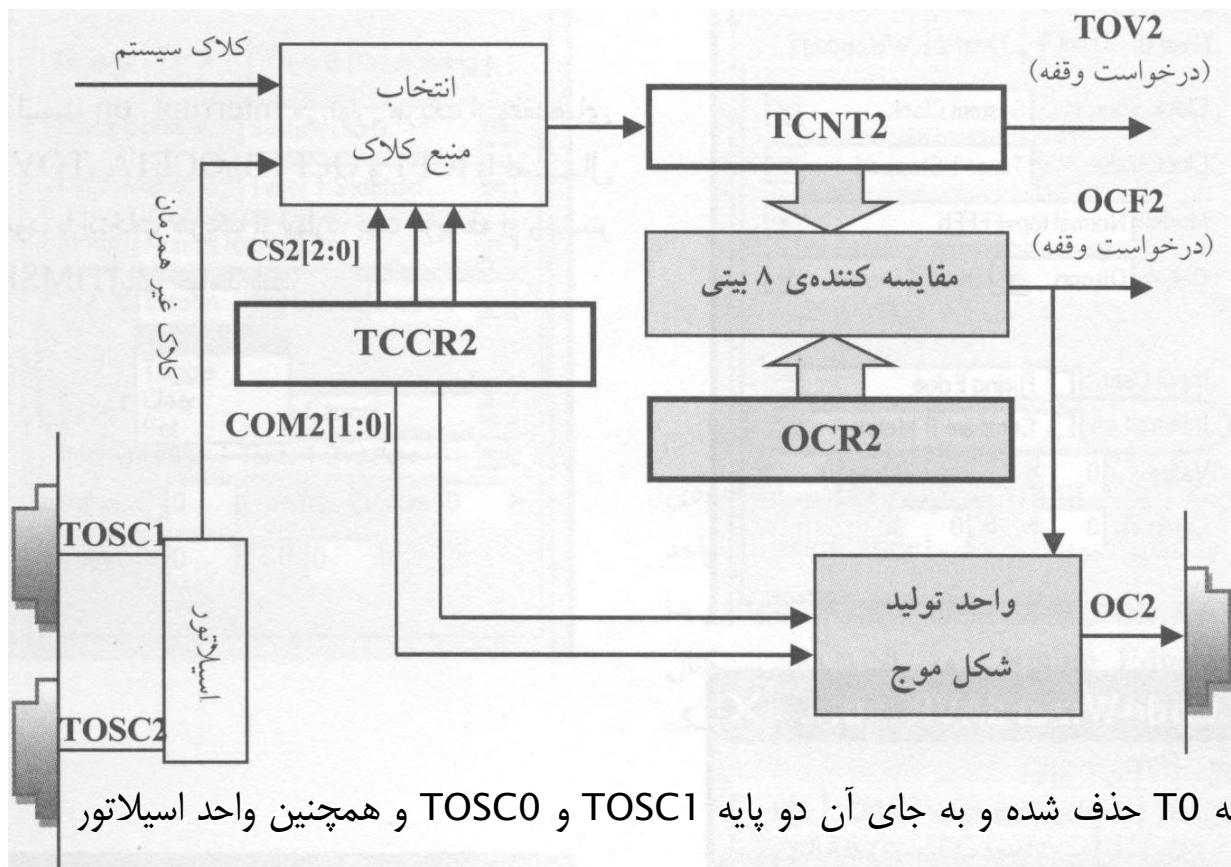
```
#include <mega32.h>
void init_io();
void main(void)
{
    init_io();
    while(1)
    {
    };
}

Void init_io()
{
    DDRD=0x20;          //Output OC1A
    // Timer/counter 1 initialization
    OCR1A=749;         //Duty Cycle: 75%
    TCCR1A=0X80;        //Phase & Frequency Correct PWM: TOP=ICR1
    TCCR1B=0X12;        //Prescaler=8
    ICR1H=0x03;
    ICR1L=0xE7;        // TOP=999
}
```

پاسخ این مثال همانند مثال قبلی است و شکل موج خروجی آنها نیز یکسان است، با این تفاوت که در صورت تغییر مقدار OCR1A و OCR1B، لحظه بروز رسانی این ثباتها متفاوت است.
تفاوت کد این برنامه و مثال قبلی در این است که تایمر/کانتر یک در وضعیت Phase and Frequency Correct تنظیم شده است.

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

آشنایی با سخت افزار تایمر/کانتر دو

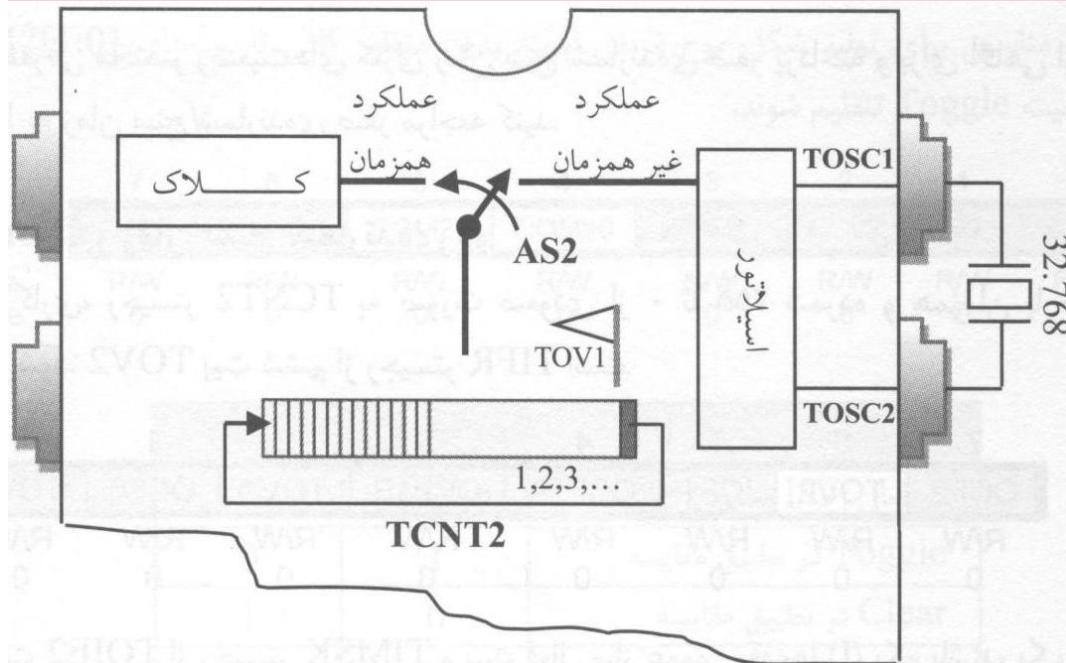


این واحد جنبی همانند تایمر/کانتر صفر، ۸بیتی است با این تفاوت که بر خلاف آن دارای ورودی شمارنده نبوده و به جای آن، قابلیت کار در وضعیت آسنکرون را دارد. بلوك منطقی آن را در شکل روی رو مشاهده می‌نمایید.

همانطور که ملاحظه می‌کنید، پایه T0 حذف شده و به جای آن دو پایه TOSC0 و TOSC1 و همچنین واحد اسیلاتور اضافه شده است.

آشنایی با میکروکنترلر AVR بخشن تایمر/کانتر

آشنایی با مفهوم عملکرد غیرهمزمان در تایمر/کانتر دو



قبلا مشاهده کردید که با اعمال یک پالس ساعت منظم به شمارنده، میتوانیم از آن به عنوان تایمر استفاده نماییم. در تمام مواردی که تا کنون از تایمر/کانتر به عنوان تایمر استفاده کرده ایم این پالس ساعت منظم از کلاک سیستم تامین شده است. حال با وضعیتی آشنا خواهیم شد که پالس تایمر از یک منع غیرهمزمان با کلاک سیستم تامین نمیشود.

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

آشنایی با مفهوم عملکرد غیرهمزمان در تایمر/کانتر دو

همانطور که در شکل مشخص است در صورتی که بیت AS2 از ثبات ASSR، یک شود منبع کلاک تایмер، از کلاک سیستم به یک اسیلاتور با فرکانس ۳۲.۷۶۸ تغییر خواهد کرد.

Bit	7	6	5	4	3	2	1	0	ASSR
Read/Write	R/W								
Initial Value	0	0	0	0	0	0	0	0	

زمانی که از عملکرد غیرهمزمان استفاده میشود، کلاک سیستم باید حداقل ۴ برابر کلاک غیرهمزمان باشد.

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

آشنایی با وضعیت های کاری تایمر/کانتر دو

وضعیت های کاری این تایمر/کانتر مشابه با تایمر/کانتر صفر بوده و بدیهی است که توسط بیتهاي WGM2[1:0] انتخاب میشود.

Bit	7	6	5	4	3	2	1	0	TCCR2
	FOC2	WGM20	COM21	COM20	WGM21	CS22	CS21	CS20	
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

نحوه انتخاب وضعیت مورد نظر با این دو بیت، در جدول زیر آمده است

WGM01	WGM00	وضعیت زمان سنج
0	0	Normal
0	1	PWM, Phase Correct
1	0	Clear Timer on Compare Match (CTC)
1	1	Fast PWM

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

وضعیت Normal در تایمر دو

در این وضعیت، ثبات TCNT2 به صورت صعودی از صفر تا ۲۵۵ شمرده و همزمان با صفر شدن آن، فلگ TOV2 یک میشود. TOV2 بیت ششم از ثبات TIFR است.

Bit	7	6	5	4	3	2	1	0	TIFR
Read/Write	R/W								
Initial Value	0	0	0	0	0	0	0	0	

در صورتی که بیت TOIE2 از ثبات TIMSK و بیت فعال ساز عمومی وقفه ها (I) یک باشد، یک شدن فلگ TOV2 میتواند باعث درخواست وقفه گردد.

Bit	7	6	5	4	3	2	1	0	TIMSK
Read/Write	R/W								
Initial Value	0	0	0	0	0	0	0	0	

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

وضعیت Normal در تایمر دو

همچنین از واحد مقایسه تایمر/کانتر دو میتوان برای ایجاد وقفه های تطبیق مقایسه با اختلاف زمانی مشخص از رخدادهای سرریز استفاده کرد. بدین منظور کافی است تا مقدار مقایسه، در ثبات OCR2 بارگذاری شود. همانطور که میدانید، محتوى ثبات TCNT2 به طور پیوسته با مقدار OCR2 مقایسه شده و در لحظه برابری (تطبیق مقایسه) فلگ OCF2 یک میشود.

Bit	7	6	5	4	3	2	1	0	
	OCF2	TOV2	ICF1	OCF1A	OCF1B	TOV1	OCF0	TOV0	TIFR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

در صورتی که بیت OCIE2 از ثبات TIMSK و بیت فعال ساز عمومی وقفه ها (I) یک باشند، یک شدن فلگ OCF2 میتواند باعث درخواست وقفه شود.

Bit	7	6	5	4	3	2	1	0	
	OCIE2	TOIE2	TICIE1	OCIE1A	OCIE1B	TOIE1	OCIE0	TOIE0	TIMSK
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

اگر چه میتوان با تنظیم بیتهاي COM2[1:0] با استفاده از واحد تولید موج، موج مربعی ایجاد نمود اما به دلیل اینکه برای مقدار اولیه دادن به شمارنده، زمان زیادی از CPU گرفته میشود، از وضعیت CTC برای تولید شکل موج استفاده میکنیم.

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

وضعیت CTC در تایمر دو

در این وضعیت با رسیدن شمارنده به مقدار OCR2، ثبات TCNT2 پاک شده و مجدداً از صفر شروع به شمارش می‌کند. بدین ترتیب رشته شمارش، محدود به اعداد صفر تا OCR2 شده و همزمان با رسیدن شمارنده به مقدار TOP، فلگ OCF2 یک می‌شود.

این وضعیت برای تولید شکل موج مربعی بسیار مناسب است. بدین منظور کافی است است تا بیتهاي COM2[1:0] مطابق با جدول زیر تنظیم شوند.

Bit	7	6	5	4	3	2	1	0	TCCR2
FOC2	WGM20	COM21	COM20	WGM21	CS22	CS21	CS20		
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

COM21	COM20	وضعیت پین OC2
0	0	غیر فعال (I/O معمولی)
0	1	در تطبیق مقایسه Toggle
1	0	در تطبیق مقایسه Clear
1	1	در تطبیق مقایسه Set

عملکرد بیتهاي COM2[1:0] در وضعیتهای غیر PWM

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

وضعیت Fast PWM در تایمر دو

وضعیت Fast PWM یکی از دو وضعیت PWM تایمر/کانتر دو است. در این حالت، ثبات TCNT2 به طور افزایشی تا ۲۵۵ شمرده و با رسیدن به TOP، مجدداً از صفر شروع به شمارش می‌کند. در لحظه‌ی تطبیق مقایسه، بسته به وضعیت بیت‌های [1:0] COM2 مطابق با جدول زیر تغییر وضعیت میدهد.

COM21	COM20	وضعیت پین OC2
0	0	غیر فعال (I/O معمولی)
0	1	رزرو شده
1	0	پاک کردن OC2 در تطبیق مقایسه و یک کردن آن در TOP (Fast PWM غیر معکوس)
1	1	یک کردن OC2 در تطبیق مقایسه و پاک کردن آن در TOP (Fast PWM معکوس)

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

وضعیت Phase Correct PWM در تایمر دو

مشابه با تایمر صفر، ثبات TCNT2 به طور صعودی از صفر تا ۲۵۵ شمرده و پس از آن به صورت نزولی تا صفر شمارش میکند. از آنجایی که در این حالت، حامل مدولاسیون دو شیبه است در هر سیکل شمارش، دو تطبیق مقایسه وجود خواهد داشت. در هر یک از دو تطبیق مقایسه، مطابق با تنظیمات بیت های COM2[1:0] وضعیت پایه ی OC2 تغییر میکند.

COM21	COM20	وضعیت پین OC2
0	0	غیر فعال (I/O معمولی)
0	1	رزرو شده
1	0	پاک کردن OC2 در تطبیق مقایسه‌ی شمارش صعودی و یک کردن آن در شمارش نزولی (PWM غیر معکوس)
1	1	یک کردن OC2 در تطبیق مقایسه‌ی شمارش صعودی و پاک کردن آن در شمارش نزولی (PWM معکوس)

آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

آشنایی با واحد انتخاب منبع کلاک در تایمر/کانتر دو

در این جدول حالت‌های لبه‌ی بالارونده و پایین رونده وجود نداشته و به جای آنها، دو مقدار تقسیم بر ۳۲ و ۱۲۸ اضافه شده است.

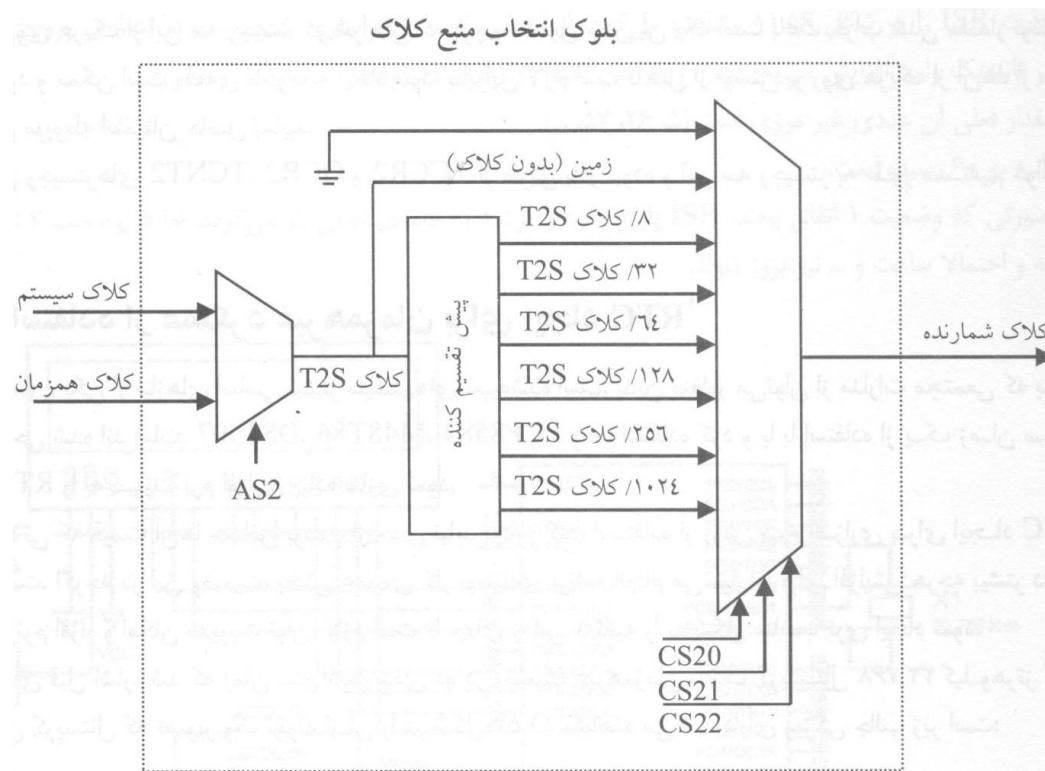
Bit	7	6	5	4	3	2	1	0	TCCR2
	FOC2	WGM20	COM21	COM20	WGM21	CS22	CS21	CS20	
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

منبع کلاک	CS22	CS21	CS20
بدون کلاک (متوقف)	0	0	0
کلاک سیستم (بدون تقسیم)	0	0	1
۸/کلاک T2S	0	1	0
۳۲/کلاک T2S	0	1	1
۶۴/کلاک T2S	1	0	0
۱۲۸/کلاک T2S	1	0	1
۲۵۶/کلاک T2S	1	1	0
۱۰۲۴/کلاک T2S	1	1	1

آشنایی با میکروکنترلر AVR بخش تایмер / کانتر

آشنایی با واحد انتخاب منبع کلاک در تایمر/کانتر دو

ممکن است این سوال مطرح شود که منظور از AS2 چیست؟ در پاسخ باید گفت که منبع کلاک تایمر دو میتواند از کلاک سیستم و یا کلاک غیرهمزمان باشد که توسط بیت AS2 تعیین میشود.



آشنایی با میکروکنترلر AVR بخش تایمر/کانتر

آشنایی با سایر بیت‌های ثبات وضعیت غیرهمزمان

با بیت AS2 از ثبات ASSR آشنا شدید و میدانید که با یک کردن این بیت، منبع کلاک شمارنده از کلاک غیرهمزمان تامین میشود. اکنون به بررسی سایر بیت‌های ثبات ASSR میپردازیم.

Bit	7	6	5	4	3	2	1	0	ASSR
Read/Write	R/W	R/W	R/W	R/W	R/W	TCNT2UB	OCR2UB	TCR2UB	
Initial Value	0	0	0	0	0	0	0	0	

زمانی که بر روی یکی از سه ثبات TCNT2، OCR2 و TCCR2 مقداری نوشته میشود، آن مقدار به یک ثبات موقتی منتقل شده و پس از لبه‌ی بالارونده‌ی TOSC1 مقدار مورد نظر به مقصد منتقل میشود.

از آنجا که کلاک غیرهمزمان (۳۷۶۸۴هرتز) بسیار پایین‌تر از کلاک سیستم است، لذا اعمال این دو لبه‌ی بالارونده، ممکن است زمان نسبتاً زیادی به طول انجامد. برای جلوگیری از بازنویسی مقدار ثبات موقت قبل از انتقال به خود ثبات، سه بافر موقت در نظر گرفته شده است. این بافرها تنها در عملکرد غیرهمزمان فعال بوده و در شرایطی که بیت AS2 صفر است عیرفعال میشوند. با نوشتن یک مقدار جدید در هر یک از این سه ثبات، فلگ مشغول بودن آن یک شده و تا اتمام بروز رسانی یک باقی میماند. زمانی که محتوى بافر موقت به خود ثبات منتقل شد، این فلگ صفر میشود. فلگ مشغول بودن ثبات‌های TCNT2، OCR2 و TCCR2 به ترتیب OCR2UB، TCNT2UB و TCR2UB میباشند. بنابراین لازم است تا قبل از نوشتن بر روی هر یک از آنها، از صفر بودن فلگ مربوطه اطمینان حاصل نمایید.

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

استفاده از عملکرد غیرهمزمان برای ایجاد RTC

آگاهی از زمان، یکی از نیازهای اساسی بیشتر سیستم‌های تعبیه شده است. بدین منظور میتواند از مدارات مجتمعی که بدین منظور طراحی شده اند (مانند DS1307، M48T86، PCF8583 و) استفاده کرد و یا با استفاده از یک تایمر، عملکرد RTC را به صورت نرم افزاری پیاده سازی نمود.

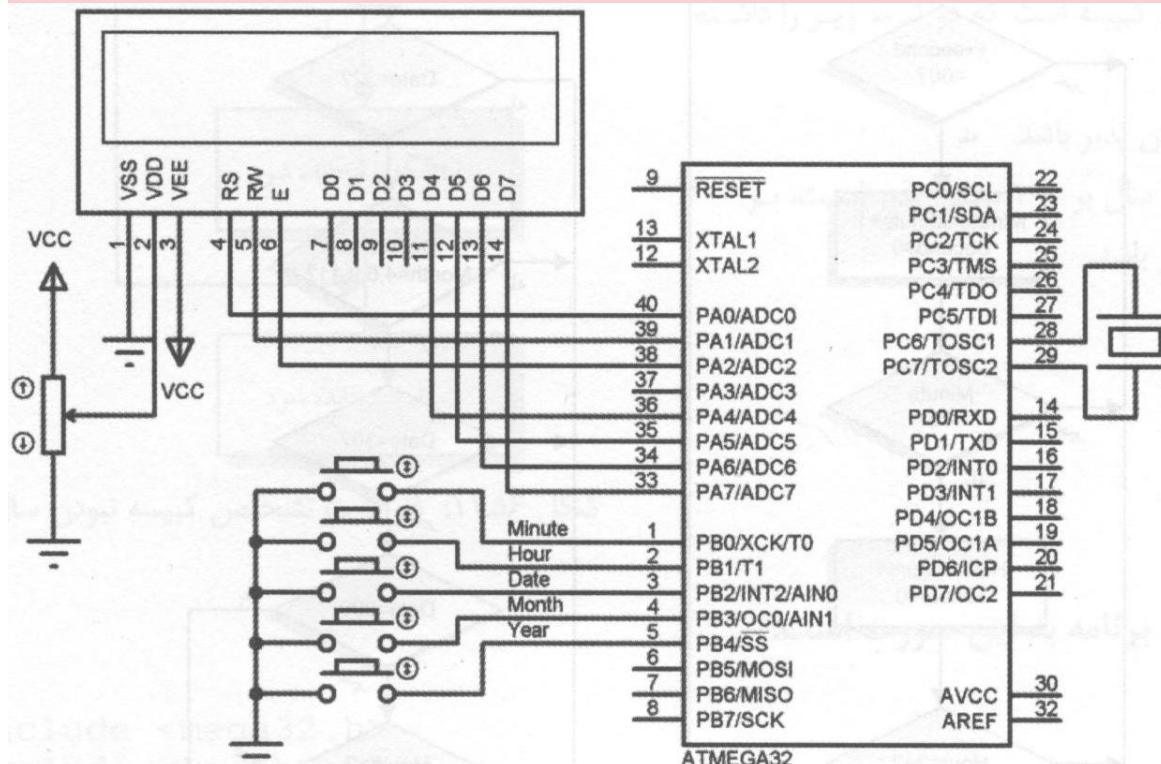
قبل اشاره شد که تایمر/کانتر دو در وضعیت غیرهمزمان با یک کریستال ۳۲۷۶۸ کیلوهرتز کار میکند. این کریستال دارای ویژگی جالب زیر است:

$$\frac{32768}{128} = 256$$

این بدان معناست که اگر فرکانس کریستال ساعت توسط پیش تقسیم کننده بر عدد ۱۲۸ تقسیم شود و به تایمر/کانتر هشت بیتی امال گردد، زمان سرریزی معادل با یک ثانیه خواهد داشت. بدین ترتیب بدون هیچ گونه بار نرم افزاری میتوانیم به زمان سرریز یک ثانیه دست پیدا کنیم. این نتیجه‌ای ایده آل برای پیاده سازی عملکرد RTC است.

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

مثال: طرح یک ساعت و تقویم با نمایش بر روی LCD کاراکتری با استفاده از وضعیت غیرهمزمان تایмер/کانتر دو

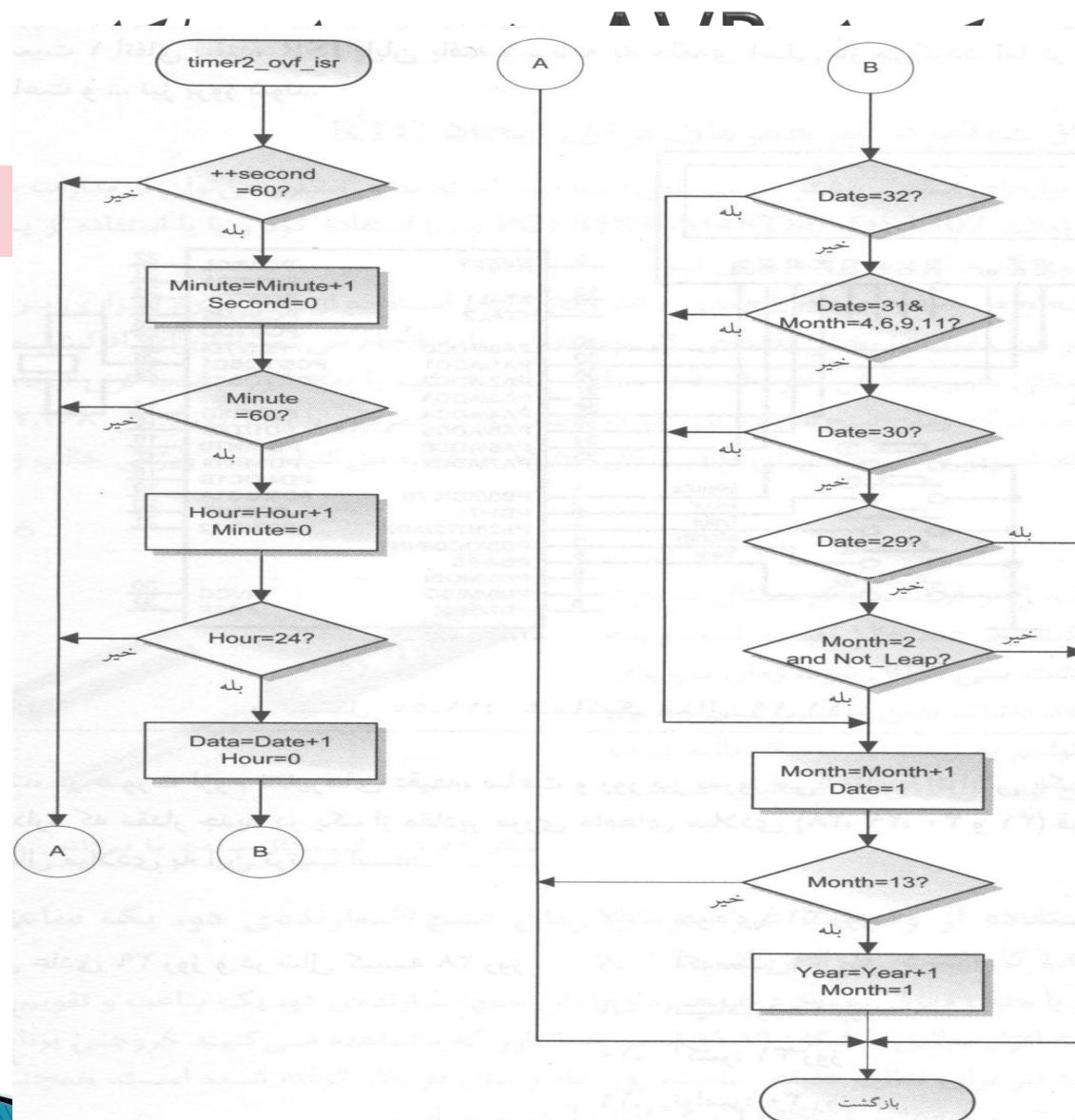


شماییک اتصالات سخت افزار مطابق با شکل زیر است. همانطور که مشاهده میکنید خروجی برنامه یک LCD کاراکتری است و ۵ کلید نیز برای تنظیم دقیقه، ساعت، روز، ماه و سال در نظر گرفته شده است. همچنین اتصال یک کریستال ۳۲۷۶۸ هرتزی به پایه های TOSC2 و TOSC1 ضروری است.

آشنایی با مید

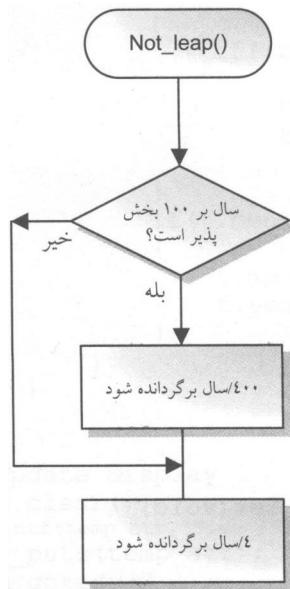
مثال: طرح یک ساعت و ته

مهمترین بخش برنامه، روتین وقفه تایمر است. با فرض اینکه تایمر/کانتر دو در وضعیت غیرهمزمان و پیش تقسیم کننده ۱۲۸ تنظیم شده باشد، پس زا هر یک ثانیه ثبات TCNT2 سرریز شده و به این ترتیب برنامه به ISR منشعب میشود.



آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

مثال: طرح یک ساعت و تقویم



در صورتی که سال کبیسه نباشد، مقدار **True** و در صورت کبیسه بودن **False** را برمیگرداند.
یک سال میلادی در صورتی کبیسه است که دو شرط زیر را داشته باشد:
- سال بر ۴ بخش پذیر باشد.
- در صورتی که سال بر ۱۰۰ بخش پذیر است، بر ۴۰۰ نیز بخش پذیر باشد.

کد برنامه در فایل main.c آمده است. آنرا بررسی کنید.

آشنایی با میکروکنترلر AVR بخش تایмер/کانتر

مثال: طرح یک ساعت و تقویم

همانطور که ملاحظه میکنندی این برنامه دارای دو زیربرنامه دیگر نیز میباشد:

تابع **(init_io)**: در این تابع، تمام تنظیمات مربوط به ورودی و خروجیها و همچنین تایمر/کانتر دو انجام میشود. نکته ای که در این تابع وجود دارد این است که در آن تاخیری به اندازه یک ثانیه برای پایدار شدن اسیلاتور در نظر گرفته شده است. علت این مسئله این است که کریستال فرکانس پایین، زمان شروع به کار نسبتاً زیادی دارد.

تابع **(check_keys)**: بوسیله این تابع، کلیدها ببررسی شده و در صورت لزوم متغیرهای مربوط به زمان بروز میشوند. در این تابع از متغیر Flag برای جلوگیری از افزایش بی مورد متغیر مورد نظر استفاده شده است. تاخیر ۲۰۰ میلی ثانیه ای نیز برای Debounce نمودن کلید بوده که بسته به نوع کلید استفاده شده میتوان آن را تغییر داد و یا حذف نمود.

در این برنامه برای نگهداری متغیرهای مربوطه به زمان از یک ساختار به نام time استفاده شده است.