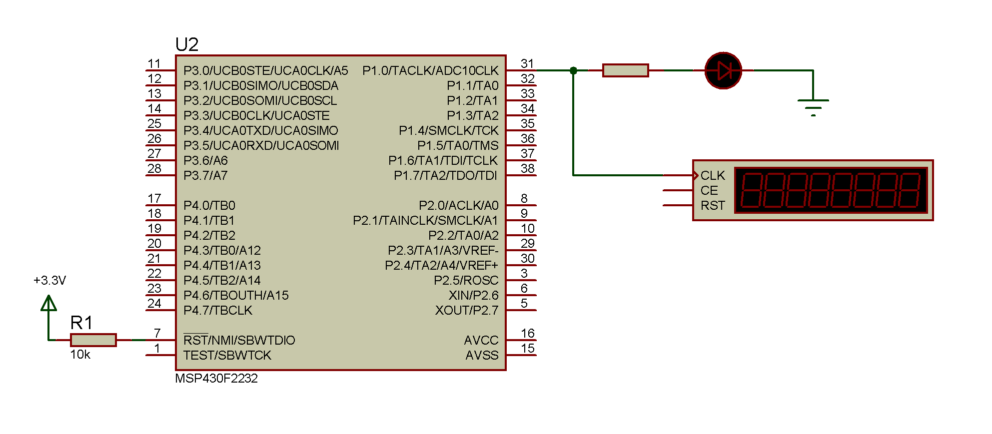
**MSP430 Uyg.15 – TimerA Uygulaması\_1**

Ferudun GÖKCEGÖZ, 06 Ağustos 2011, Cumartesi

Merhaba arkadaşlar. Yeni bir yazımızda yine sizlerle birlikteyiz. Bu uygulamamızda MSP430 un Timer larını kullanmaya başlayacağız. MSP430 larda bir kısmında TimerA bazılarında ise TimerA ve TimerB olmak üzere iki adet 16 bit timer modülü bulunmaktadır. İsterseniz ilk basit timer uygulamamıza devre şemasını vererek başlayalım.

**İşte devre şemamız.**



Devre şemamız oldukça sade. Bu uygulamadaki amacımız, basit olarak MSP430 un timerlarını kullanabilmek için giriş yapmak. İlk başta biraz MSP430 daki timerlardan bahsedelim.

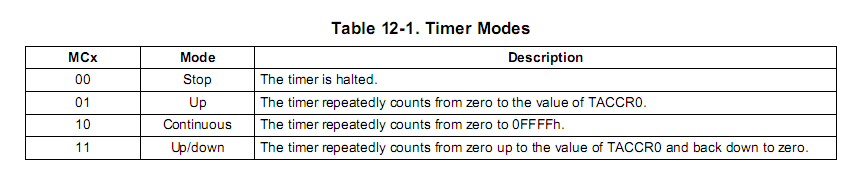
* MSP430 un timerları çeşitli sayma modlarına sahiptir. TimerA control register içerisindeki MC bitleri ile timer ların çalışma modları ayarlanmaktadır.

MC\_0 : Timer Kapalı

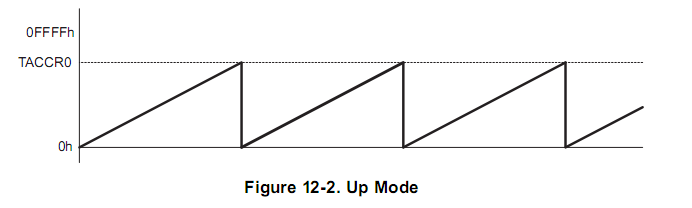
MC\_1 : Timer sürekli olarak sıfırdan CCR değerine kadar sayar.

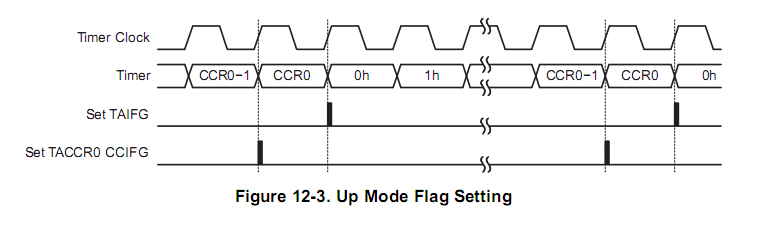
MC\_2 : Timer sürekli olarak 0 dan oxFFFF e kadar sayar.

MC\_3 : Timer sürekli olarak sıfırdan CCR değerine, CCR değerine ulaşınca tekrar sıfıra geri doğru sayar.



Biz uygulamamızda TimerA yı “Up Mode” da kullanacağız. Yani Timer sıfırdan bizim belirlediğimiz CCR değerine kadar sayacak ve tekrar sıfıra dönecek. Sıfıra dönme yani resetlenme esnasında da, kesme üretecek. Bizim kullanacağımız UP Mode ile alakalı grafiği incelersek, çalışmasını daha iyi kavrayabiliriz.





İsterseniz yazılımı da verip, üzerinde konuşmaya başlayalım…

|  |  |
| --- | --- |
| **[Source code](http://www.mcu-turkey.com/msp430-uyg-15/" \l "codesyntax_1" \o "Click to show/hide code block)** | **[http://www.mcu-turkey.com/wp-content/plugins/wp-synhighlight/themes/default/images/code.png](http://www.mcu-turkey.com/msp430-uyg-15/#codesyntax_1)** **[http://www.mcu-turkey.com/wp-content/plugins/wp-synhighlight/themes/default/images/printer.png](http://www.mcu-turkey.com/msp430-uyg-15/#codesyntax_1)** **[http://www.mcu-turkey.com/wp-content/plugins/wp-synhighlight/themes/default/images/info.gif](http://www.mcu-turkey.com/wp-content/plugins/wp-synhighlight/About.html)** |

*#include "io430.h"*

*#include "in430.h"*

*#define LED P1OUT\_bit.P0*

unsigned char i;

void main( void )

{

WDTCTL = WDTPW + WDTHOLD;

DCOCTL=CALDCO\_1MHZ;

BCSCTL1=CALBC1\_1MHZ;

P1OUT = 0x00;

P1DIR = 0x01;

TACCTL0=CCIE;

TACCR0=50000;

TACTL=TASSEL\_2 + MC\_1 + TAIE;

\_BIS\_SR(GIE);

for(;;);

}

*#pragma vector=TIMERA0\_VECTOR*

\_\_interrupt void Timer\_A0 (void)

{

i++;

if(i==10)

{

LED=~LED;

i=0;

}

TACTL\_bit.TAIFG = 0;

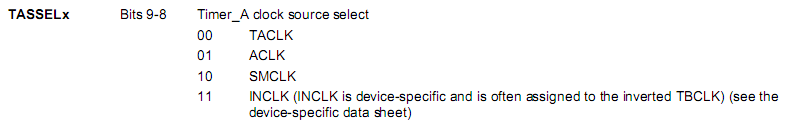
}

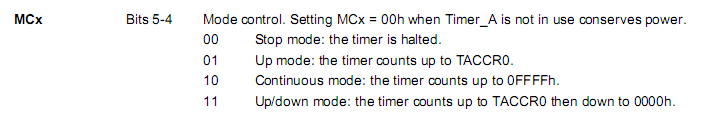
Yazılımda ilk olarak, daha sonra kullanılmak üzeri P1.0 pinine bağlı olan led tanımlanmış. Sonrasında ise unsigned char türünden, “i” isimli bir değişken tanımlanmıştır. Daha sonrada main fonksiyonuna geçilmiştir. Main fonksiyonunda ise ilk olarak her zamanki gibi WDT timer ve Clock ayarları yapılmıştır. Ardından port ayarlarına geçilmiştir. Port ayarlarında ise led imizin bağlı olduğu P1 portu tamamen temizlenmiş ve ardınan P1.0 pini çıkış olarak şartlanmıştır. Daha sonra ise timer ayarlarının yapıldığı kısma gelinmiştir.

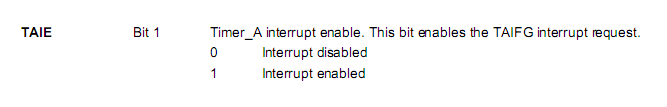
Timer ayarlarının olduğu kısımda ise TACCTL0(TimerA Capture/Compare Register 0) kaydedicisindeki CCIE biti Lojik1 yapılmıştır. Bunun şekilde timer için interrupt kullanımını aktif hale getirilmiştir.

Daha sonra ise TimerA CCR0 değerine “50000″ sayısı yüklenmiştir. Bir sonraki satırda göreceğiz ki, timer için gerekli saat darbelerini SMCLK tan sağlayacağız. SMCLK yı başta 1MHz olarak ayarlamıştık. Yani Timer a gelen darbelerin frekansı 1MHz, dolayısıyla periyodu ise, 1μsn dir. Yani Timer ımız 50000\*1μsn = 50 msn de bir kesme üretecektir. Gelelim bir sonraki satıra..

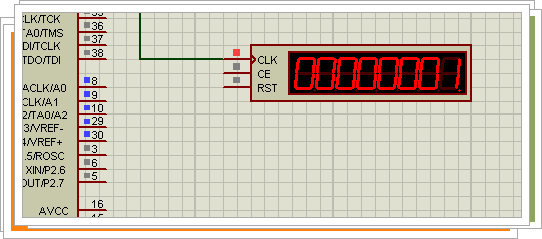
Bir sonraki satırda ise TimerA control register için gerekli ayarlar yapılmıştır. İlk olarak TASSEL(TimerA Clock Source Select) bitleri TASSEL\_2 olarak ayarlanmıştır. Yani timer a gelen saat darbeleri SMCLK tarafından sağlanacaktır. Daha MC bitleri ise MC\_1 olarak seçilmiştir. Bunun anlamı ise TimerA UP Mode da çalıştırılacaktır. Ve bu register da son olarak TAIE(TimerA Interrrupt Enable )bitide Lojik1 yapılarak kesmeler aktif hale getirilmiştir. Ardından da tüm kesmelere izin verilerek sonsuz döngüye girilmiştir.



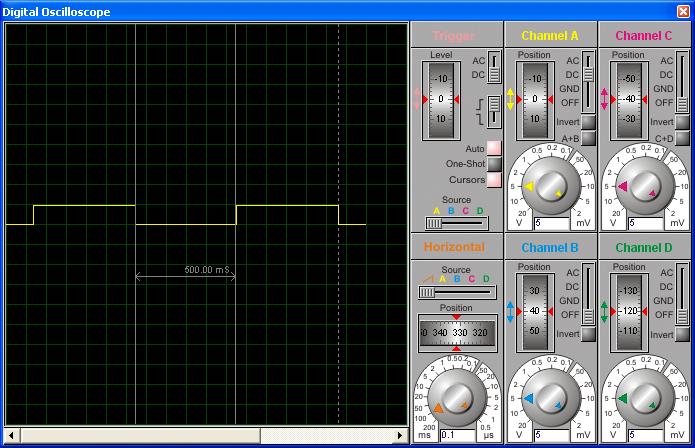




Gelelim kesme alt programında icra edilecek işlemlere… Kesme fonksiyonuna ilk girildiğinde main fonksiyonundan önce tanımlanan i değişkeni 1 artırılmıştır. Daha sonra ise bu değişkenin “10″ değerine eşit olup olmadığı kontrol edilmiştir. Eğer eşit değilse, kesme bayrağı temizlenip kesme alt programında çıkılmıştır. Eşit olduğu taktirde de, P1.0 pinine bağlı olan led terslenmiştir. Kesme alt programında bu şekilde bir yöntem kullanılmasının sebebi şudur. Yukarıda hesapladığımız gibi TimerA modülü 50 msn de bir kesme üretecektir. Kesme alt programına 10 kere girildiğinde, toplam geçen süre 500 msn olacaktır. Bu şekilde, 500 msn de bir led in durumu terslenerek led in 500 msn yanık, 500 msn sönük durumda olması sağlanmıştır. Böylelikle P1.0 pininde frekansı 1 Hz olan bir sinyal elde edilmiştir. Bunu o pine bağlanan bir frekansmetre ilede görebiliriz.



İstersenin P1.0 pinindeki değişkleri birde osiloskop üzerinde inceleyelim.



Geldik bir yazımızın daha sonuna. Bir sonraki dersimizde yine TimerA ile ilgili bir uygulama yapacağız. Bir dahaki dersimizde görüşmek üzere. Şimdilik Hoşçakalın..

***Ferudun GÖKCEGÖZ***

***fgokcegoz@yahoo.com***