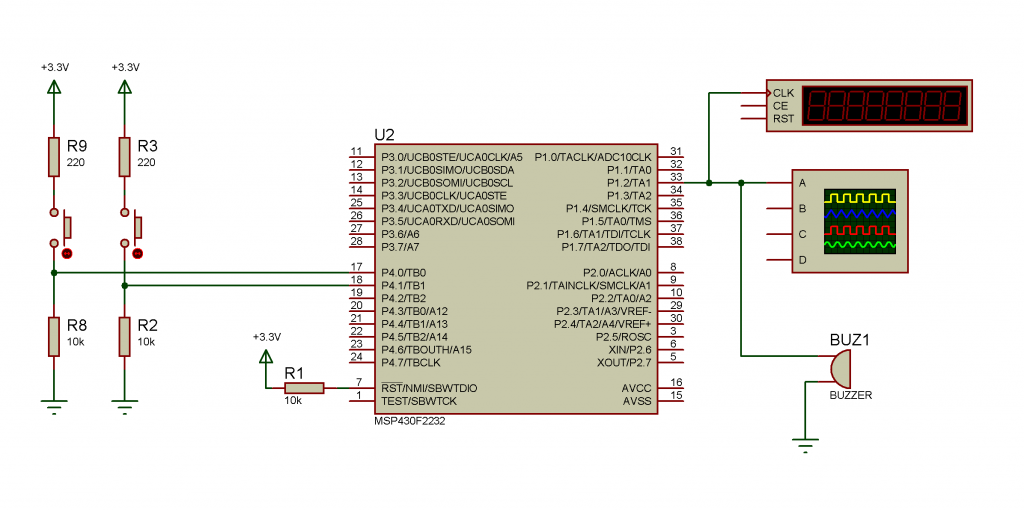
**MSP430 Uyg.22 – PWM Uygulaması\_2**

Ferudun GÖKCEGÖZ, 09 Ağustos 2011, Salı

Merhaba arkadaşlar. Bu dersimizde geçenki derste söylediğimiz gibi bir PWM uygulaması yapacağız. Geçenki derste frekansı ve duty cycle değeri sabit bir pwm sinyali üretmiştik. Bu dersimizde ise MSP430 a bağlı iki adet buton ile duty cycle değerini artırıp azaltacağız. İsterseniz gerekli uyarlamaları yaparak bu uygulamayı launchpad üzerinde de yapabilirsiniz. Eğer hiç bir ek donanımla uğraşmamayım diyorsanız, launchpad üzerindeki butonu sadece artırma veya azaltma butonu olarak kullanabilirsiniz. Ben o şekilde denedim. Yazının sonunda da video yu sizlerle paylaşacağım…

**İlk olarak devre şemamızı verelim hemen…**

****

Devre şemasında da görüldüğü gibi MSP430 a iki adet buton bağlanmış. PWM çıkışında ise bir adet frekansmetre, bir adet osiloskop bir adette buzzer bağlanmış. Buzzer ı neden bağladığımız anlamışsınızdır zaten. PWM sinyalinin duty cycle değerinin değişimini osiloskop üzerinde gözlerken buzzer ın çıkardığı sesleri gözlemlemek için koydum. Bu dersimizde PWM in ne olduğu konusuna değinmeyeceğim. Geçen derste yeteri kadar bahsettiğimizi düşünüyorum.

Gelelim uygulamaya ait yazılımımıza…

|  |  |
| --- | --- |
| **[main.c](http://www.mcu-turkey.com/msp430-uyg-22/" \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-22/#codesyntax_1)** **[http://www.mcu-turkey.com/wp-content/plugins/wp-synhighlight/themes/default/images/printer.png](http://www.mcu-turkey.com/msp430-uyg-22/#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"

void main( void )

{

WDTCTL = WDTPW + WDTHOLD;

DCOCTL=CALDCO\_1MHZ;

BCSCTL1=CALBC1\_1MHZ;

P1DIR = BIT2;

P1SEL = BIT2;

P4SEL = 0x00;

P4DIR = 0x00;

TACCR0 = 20000;

TACCR1 = 10000;

TACCTL1 = OUTMOD\_7;

TACTL = TASSEL\_2 + MC\_1 + ID\_0;

for(;;)

{

if(P4IN\_bit.P0)

{

TACCR1 += 1000;

if(TACCR1 >= 19000) TACCR1 = 19000;

while(P4IN\_bit.P0);

}

else if (P4IN\_bit.P1)

{

TACCR1 -=1000;

if(TACCR1 <= 1000) TACCR1 = 1000;

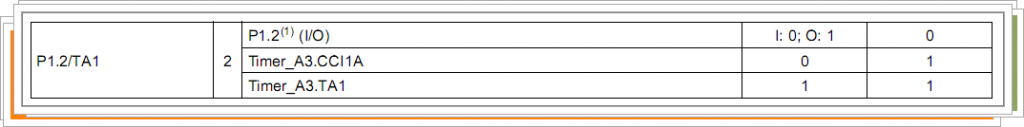
while(P4IN\_bit.P1);

}

}

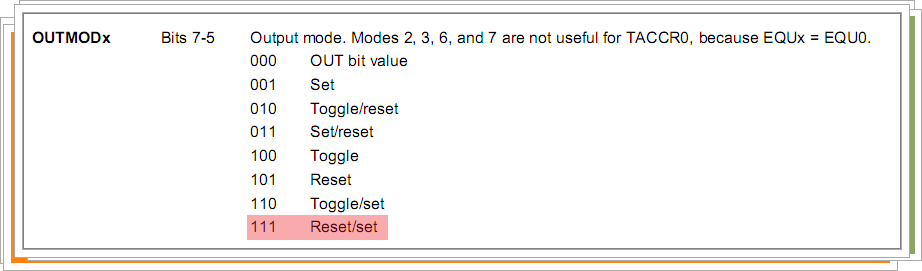
}

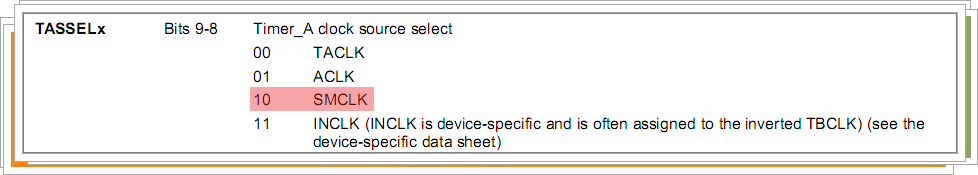
Yazılımı baştan itibaren inceleyecek olursak, ilk olarak main fonksiyonunda WDT ve clock ayarları yapılmış. Daha sonrada port ayarlarına geçilmiş. Port ayarlarında ise ilk olarak PWM sinyalinin çıkışını aldığımız pini ayarları yapılmıştır.

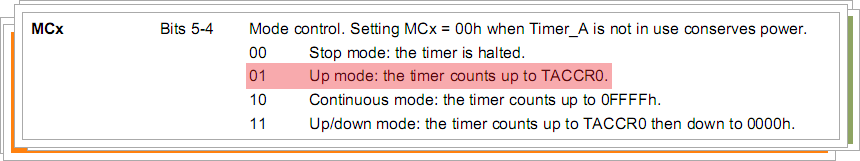
****

Daha sonra ise butonların bağlı olduğu port için gerekli ayarlamalar yapılmıştır. Bu pinler için ise ilk olarak P1SEL kaydedicisi ile bu pinleri Digital I/O olarak kullanacağımız belirtilmiş ardındanda giriş olarak şartlandırılmıştır. Daha sonra ise timer ayarlarına geçilmiştir.

Timer ayarlarında ise ilk olarak TimerA CCR0 ve CCR1 değerleri yüklenmiştir. Bu değerlerle ilgili hesaplamadan bahsetmeden timer konfigurasyonlarından bahsedelim isterseniz.







Görüldüğü gibi TimerA birimimiz UP Mode da kullanılmış, clock kaynağı olarak SMCLK seçilmiştir. Output Mode olarakta PWM sinyali için mode7 seçilmiştir. Daha sonra ise sonsuz döngüye girilmiştir. Sonsuz döngü içerisinde sürekli olarak butonlara basılıp basılmadığı kontrol edilmiştir. Eğer basılmamışsa sonsuz döngü içerisinde yapılacak başka bir işlem yoktur.

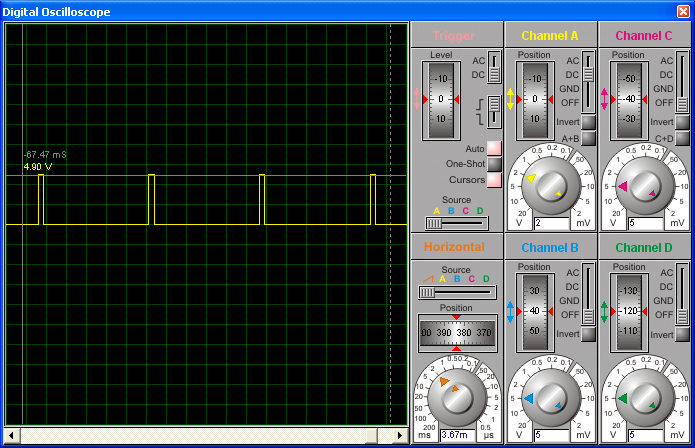
Eğer kullanıcı butonlardan P4.0 pinine bağlı artırma butonuna basarsa, if koşulunun ilk kısmına girilecek ve TimerA CCR1 değeri artırılmıştır.

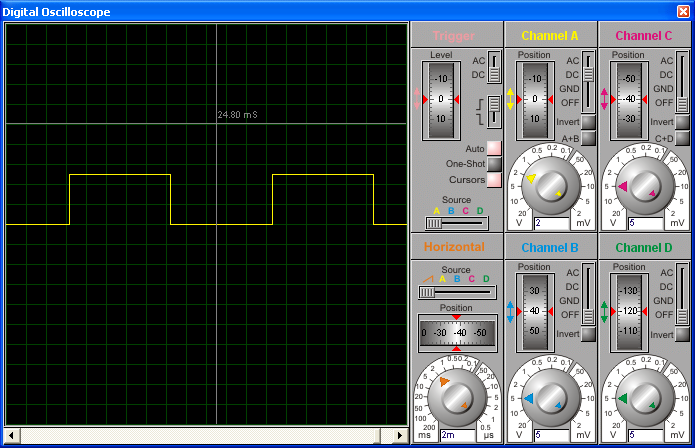
*Hatırlayalım ki, TimerA CCR1 yüklenen değer ile PWM sinyalinin duty cycle değerini belirliyorduk.*

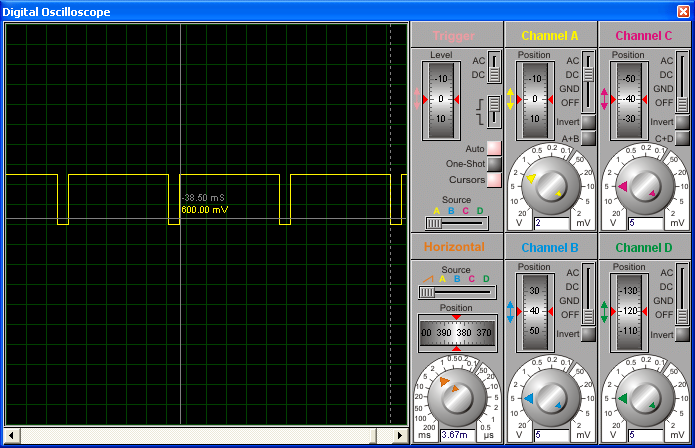
Daha sonra CCR1 değeri için konulan üst sınırın aşılıp aşılmadığı kontrolü yapılmıştır. Eğer aşılmışsa CCR1 değeri için belirlenen üst limit değeri tekrar yüklenecektir. Bu aşamadan sonra kullanıcı hala butona basıyorsa, whille döngüsünün içinden çıkılamayacaktır. Bu şekilde bir ifade kullanılarak, butona her basılışta bir artırma işlemi gerçekleştirilmesi sağlanmıştır.

Eğer kullanıcı P4.1 pinine bağlı olan azaltma butonuna basarsada, else if kısmındaki şart sağlanacak ve içerideki komutlar icra edilecektir. Buradaki satırlardan tekrar bahsetmeye gerek yok zannedersem. Çünkü artırma kısmındaki kodlar ile aynı. Tek fark olarak artırma işlemi yerine azaltma işlemi yapılmakta ve CCR1  değeri için üst limit değilde alt limit kontrolü yapılmaktadır.

Proteus üzerindeki osiloskop ile çıkışı gözlersek, aşağıdaki sinyalleri gözlemleyebiliriz.







Geldik bir yazımızın daha sonuna… Bir dahaki dersimizde MSP430 ile bir KeyPad uygulaması yapacağız. Umarım faydalı olabilmişimdir. Bir dahaki dersimizde görüşmek üzere. Şimdilik Hoşçakalın.

**Ferudun GÖKCEGÖZ**

**fgokcegoz@yahoo.com**