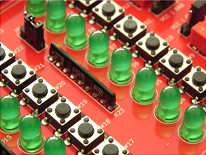
**2-Buton – LED Uygulaması**

Muhammed Fatih İNANÇ, 07 Aralık 2012, Cuma

**[](http://www.mcu-turkey.com/wp-content/uploads/2012/10/led_kucuk.png)**

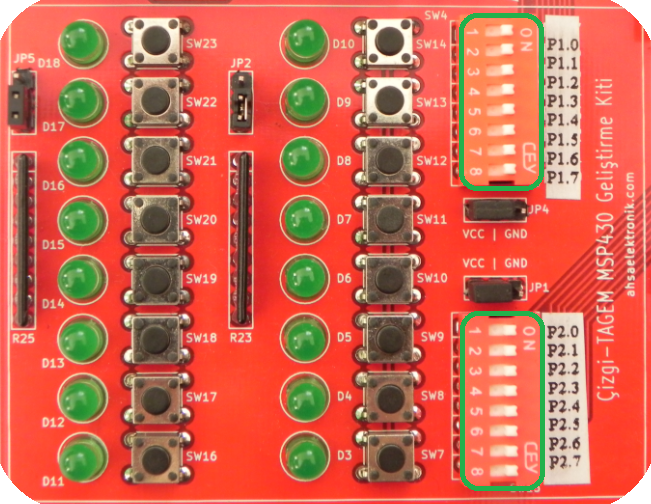
Bu uygulama notunda MSP430 Geliştirme Kiti üzerinde “Buton – LED” uygulamasının nasıl yapılacağını inceleyeceğiz.

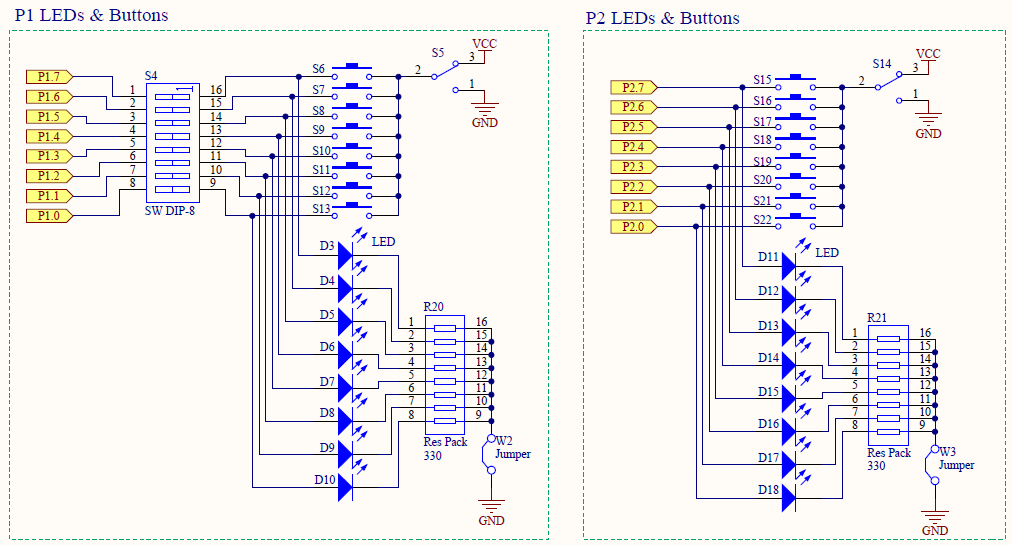
Bu uygulamada gerçekleştirilmesi amaçlanan işlem, kit üzerinde bulunan LED leri, butonların basılma durumuna göre kontrol etmektir.

**Portların Ayarlanması**

Ayarlanması gereken switchler;

* SW4 ve SW15 in tümü **açık**.
* JP1 ve JP4 **GND** konumunda
* JP2 ve JP5 **GND** konumunda
* LaunchPad üzerindeki J3 konnektöründen TXD ve RXD jumperlerı **çıkarılmalıdır**.

****

****

**Uygulamanın Çalışması**

P2 Portundaki herhangi bir butona basıldığında P1 Portunda o pine karşılık gelen LED yanmaktadır. P1 Portundaki değişikliği algılama işlemi yani butona basıldığını kontrol etme olayı *kesmeler* kullanılarak gerçekleştirilmiştir. P1 Kesme bayrağının o anki değeri doğrudan P2OUT saklayıcısına yüklenrek LED yakma işlemi gerçekleştirilmiştir.

**Uygulama Kodu**

//

//LaunchPad Üzerindeki TXD ve RXD Jumplerlarını Çıkarınız

#include "msp430.h"

void main( void )

{

WDTCTL = WDTPW + WDTHOLD;

BCSCTL1 = CALBC1\_1MHZ;

DCOCTL = CALDCO\_1MHZ;

\_\_delay\_cycles(1000000);

P2DIR = 0x00;

P2IE = 0xFF;

P2IES = 0x00;

P2SEL = 0x00;

P2SEL2= 0x00;

P1DIR = 0xFF;

P1OUT = 0x00;

P2IFG = 0x00;

P2OUT = 0x00;

\_\_bis\_SR\_register(LPM4\_bits + GIE);

}

// Port 2 Kesme Rutini

#pragma vector=PORT2\_VECTOR

\_\_interrupt void P2\_ISR(void)

{

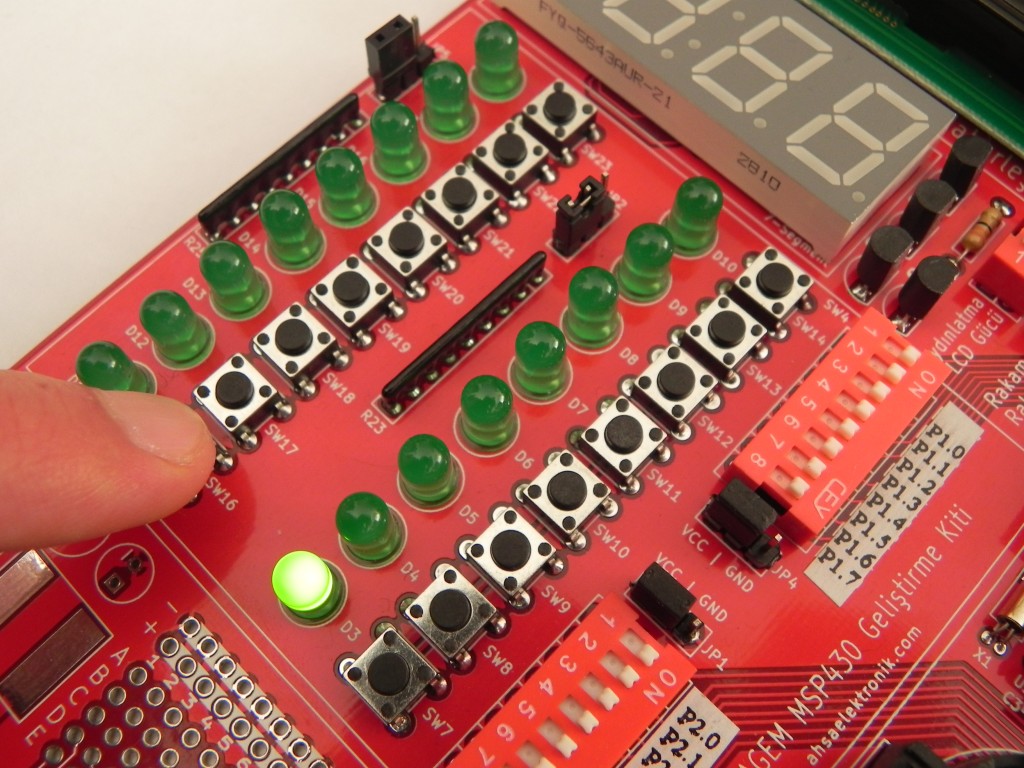
\_\_delay\_cycles(10000);

P1OUT = P2IFG;

P2IFG = 0x00;

}

* 10 ve 11. satırlarda flash hafızada bulunan RC osilatör kalibrasyon değerleri okunarak CPU frekansı 1 MHz’e ayarlanıyor. Ve arkasından(18.satır) kalibre edilmiş frekansın stabilitesi için 100 milisaniye bekleniyor.
* 14 ve 17.satırlar arasında ise Port tanımlamaları yapılıyor. Tüm pinler GPIO olarak seçiliyor ve P2 için tüm pinlerin kesmeleri aktif hale getiriliyor.
* 25.satırda mikrodenetleyici **LPM4** moduna alınıyor ve tüm genel kesmeler aktif hale getiriliyor(*GIE*). LPM4 modunda tüm clock kaynakları kapalı durumdadır ve MCU, yanlızca pinlerden gelen harici kesmelere duyarlıdır. Bu uygulamada harici kesmeye gitme durumu butonlara basılarak oluşturulmaktadır.
* 29.satırda ise PORT2 için kesme vektörü tanımlanıyor. Herhangi bir kesme gelene kadar(yani herhangi bir butona basılana kadar) MCU düşük güç modunda(uyku) da tutuluyor ve kesme geldiğinde *P2\_ISR* olarak tanımlanmış kesme vektörü içerisindeki satırları işleterek tekrar düşük güç moduna geçiyor. Herhangi bir kesme daha geldiğinde yine aynı işlemler tekrarlanıyor.
* 32.satırda P1OUT saklayıcsına P2IFG saklayıcısının içeriği aktarılıyor. Bunun amacı: o an basılan buton durumuna göre P2IFG(*P2 Interrupt Flag*) bir değer alıyor ve bu değer PORT1 in pinlerinde gösterilmek üzere P2OUT saklayıcısına aktarılıyor.  
  Mesela **P2.3** pinindeki butona bastığınızda P2IFG saklayıcısı 0000 1000(0×08) değerini alıyor. Bu değer P1OUT saklayıcısına aktarıldığında PORT1 de bulunan 4. LED yanıyor.
* 33.satırda ise P2 kesme bayraklarının tamamı temizlenerek butona basılması anında tekrar kesme vektörüne dallanılmaya izin veriliyor.

****