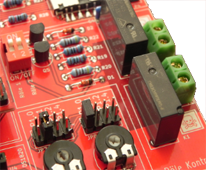
**3-Röle Kontrol Uygulaması**

Muhammed Fatih İNANÇ, 08 Aralık 2012, Cumartesi



Bu uygulama notunda MSP430 Geliştirme Kiti üzerinde röle kontrol uygulamasının nasıl yapılacağını inceleyeceğiz.

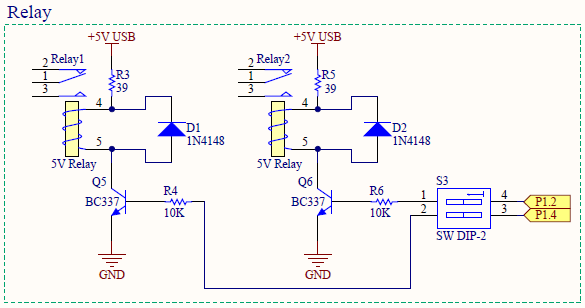
Bu uygulamada gerçekleştirilmesi amaçlanan işlem, kit üzerinde bulunan “Röle Kontrol” kısmındaki 2 adet röleyi *timer* kullanarak belli bir zaman periyodunda kontrol etmektir.

**Portların Ayarlanması**

Ayarlanması gereken switchler;

* SW3 tümü **açık** konumda
* SW4 P1.2 ve P1.4 **açık** konumda
* Diğer tüm switchler **kapalı** konumda
* LaunchPad üzerindeki J3 konnektöründen TXD ve RXD jumperlerı **çıkarılmalıdır.**

****

****

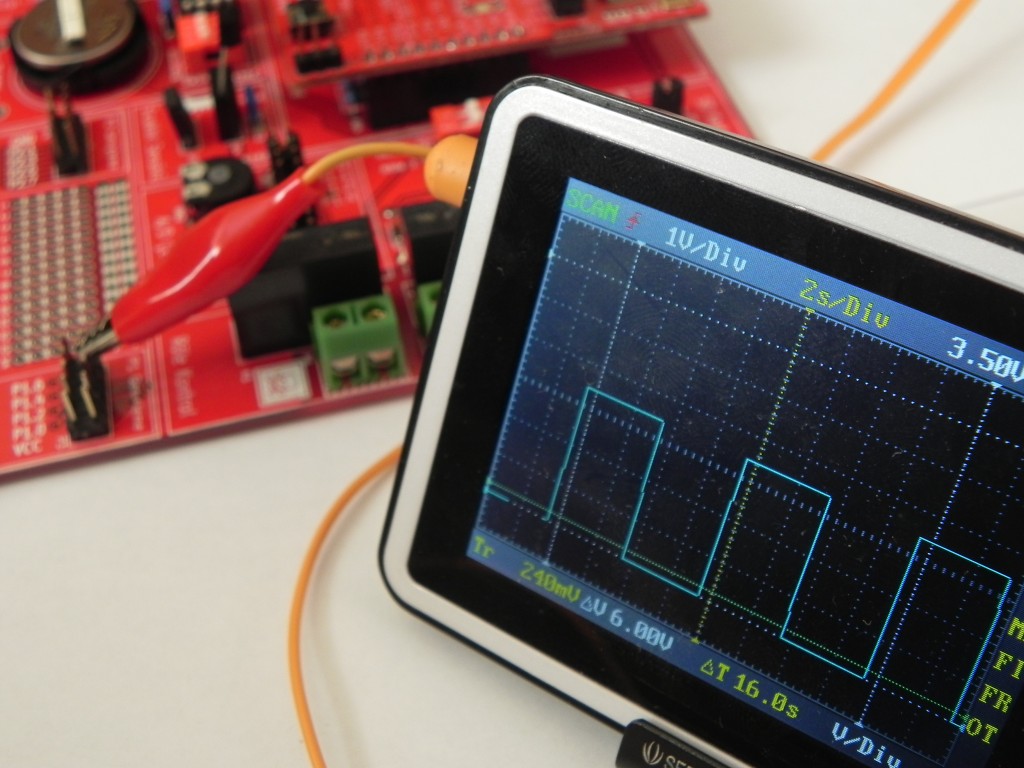
**Uygulamanın Çalışması**

Kit üzerinde bulunan 2 adet röle bir timer(zamanlayıcı) vasıtası ile kontrol edilmektedir. Ayarlanan timer 50ms lik periyotlarla kesme vektörüne dallanmaktadır. Bu kesme vektörü içerisindeki bir sayaç(*tick* değişkeni) 40 kez 50ms leri saydığınıda “Röle 1” toggle(1 ise 0, 0 ise 1) olur, 80 kez 50ms leri saydığında ise “Röle 2” toggle olur. Yani her 2 saniyede (50ms \* 40) bir kez “Röle 1”, her 4 saniyede(50ms \* 80) 1 kez de “Röle 2” toggle olur. “Röle 2” toggle olduktan sonra sayaç sıfırlanır.

**Uygulama Kodu**

|  |  |
| --- | --- |
| main.c | |
| 01  02  03  04  05  06  07  08  09  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50 | //  //  // LaunchPad Üzerindeki TXD ve RXD Jumplerlarını Çıkarınız    #include "msp430.h"    #define ROLE1(x)       ( (x) ? (P1OUT |= BIT2) : (P1OUT &= ~BIT2) )  #define ROLE2(x)       ( (x) ? (P1OUT |= BIT4) : (P1OUT &= ~BIT4) )  #define ROLE1\_TERSLE   (P1OUT ^= BIT2)  #define ROLE2\_TERSLE   (P1OUT ^= BIT4)    unsigned int  tick  = 0;    void main( void )  {    WDTCTL = WDTPW + WDTHOLD;      BCSCTL1 = CALBC1\_1MHZ;    DCOCTL  = CALDCO\_1MHZ;    \_\_delay\_cycles(1000000);      P1DIR = BIT2 + BIT4;    P1OUT = 0x00;      ROLE1(0);    ROLE2(1);      TA0CTL   = TASSEL\_2 + MC\_1 + TACLR;    TA0CCR0  = 50000 - 1;    TA0CCTL0 = CCIE;      //LPM1 Düşük güç moduna giriliyor ve kesmeler aktif    \_\_bis\_SR\_register(LPM1\_bits + GIE);  }    //TimerA0 Kesme Rutini  #pragma vector=TIMER0\_A0\_VECTOR  \_\_interrupt void TIMERA0\_A0\_ISR(void)  {    tick++;      if(tick == 40)        //2 saniye      ROLE1\_TERSLE;      if(tick == 80)    {      ROLE2\_TERSLE;      tick = 0;    }  } |

* 22.satırda P1.2 ve P1.4 çıkış diğer pinler giriş olarak seçiliyor.
* 23.satırda ise P1 portunun tüm çıkışları sıfırlanıyor.
* 28.satırda **Timer\_A0** biriminin ayarlamaları yapılıyor.
* 29.satırda **TA0CCR0** saklayıcısına *50000-1* değeri yüklenerek kesmeye gitme periyodu **50mS** olarak ayarlanıyor.
* 30.satırda ise **TA0CCR0** için kesmeler aktif hale getiriliyor.
* 33.satırda MCU düşük güç moduna alınarak kesmeler aktif hale getiriliyor.
* Bir timer kesmesi oluştuğunda yani her 50mS de bir defa program *TIMERA0\_A0\_ISR* etiketi ile tanımlanmış kesme vektörüne dallanıyor.
* 40. satırda *tick* değişkeni 1 kez arttırılıyor. Bu değişken aslında **50mS** leri sayarak 2 saniyelik süreyi tesbit etmeye yarıyor. 2 saniye geçtiğinde **ROLE1** toggle yapılıyor. 4 saniye geçtşiğinde ise **ROLE2** toggle yapılıyor. 4 saniye geçtiğinde *tick* değişkeni sıfırlanıyor.

****