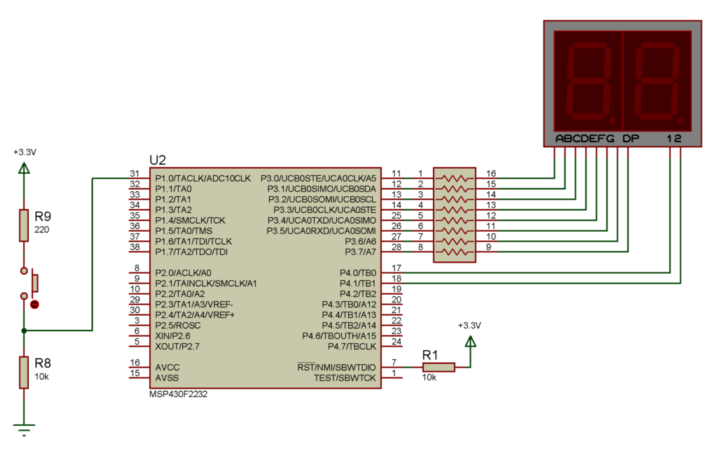
**MSP430 Uyg.8 – Display Uygulaması\_2 (0-99 Sayıcı)**

Ferudun GÖKCEGÖZ, 04 Ağustos 2011, Perşembe

Merhaba arkadaşlar. 8. dersimizle yine sizlerle birlikteyiz. Bir önceki derste söylediğim gibi bu derste, 0-99 sayıcı uygulaması yapacağız. Ve sayıcımız buton kontrollü olacak. Ayrıca iki rakam gösterme mecburiyetinden dolayı iki adet seven segment kullanacağız. İki adet seven segment i de tarama yöntemi ile sürekli yanıyormuş gibi kullanacağız. Hadi bakalım devre şemasını vererek başlayalım.



Görüldüğü gibi devremizde P1.0 pinine bağlı bir adet butonumuz bulunmakta. Ve P3 portuna ortak olarak bağlı iki adet seven segment display imiz bulunmakta. Displaylerin ABCDEFG-Dp uçları ortak. Yani aynı yerden datalarını alıyorlar. Display lerden hangisini kullanacağımızı seçmek için ise iki adet bağlantımız (Display üzerindeki 1-2 pini) bulunmakta. Soldaki displayi seçmek istersek, P4.0 ı, sağdakini seçmek istersek, P4.1 i Lojik 0 seviyesine çekmemiz gerekiyor. Birde unutmadan şunu söyleyelim, display imiz ortak katot. Yani ABCDEFG-Dp uçları anot, katot uçları ise birleştirilmiş şekilde biri “1″ pini, diğeri ise “2″ pinine bağlı. O yüzden 1/2 uçlarını Lojik 0′ a çekerek displayleri aktif hale getiriyoruz.

Yazılıma geçersek;

|  |  |
| --- | --- |
| **[Source code](http://www.mcu-turkey.com/msp430-uyg-8/" \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-8/#codesyntax_1)** **[http://www.mcu-turkey.com/wp-content/plugins/wp-synhighlight/themes/default/images/printer.png](http://www.mcu-turkey.com/msp430-uyg-8/#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 Disp\_1 P4OUT\_bit.P0*

*#define Disp\_2 P4OUT\_bit.P1*

*#define BUTTON P1IN\_bit.P0*

unsigned char d\_0,d\_1;

unsigned char num;

void delay (void)

{

unsigned int j=500;

while(j--);

}

**const** unsigned char Numbers[]={0x3F, 0x06, 0x5B, 0x4F,

0x66, 0x6D, 0x7D, 0x07,

0x7F, 0x6F};

void main( void )

{

WDTCTL = WDTPW + WDTHOLD;

DCOCTL=CALDCO\_1MHZ;

BCSCTL1=CALBC1\_1MHZ;

P1DIR = 0x00;

P3OUT = 0x00;

P3DIR = 0xFF;

P4OUT = 0x00;

P4DIR = 0xFF;

Disp\_1 = 1;

Disp\_2 = 1;

for(;;)

{

d\_0 = num%10;

d\_1 = num/10;

Disp\_2 = 0;

P3OUT = Numbers[d\_0];

delay();

Disp\_2 = 1;

Disp\_1 = 0;

P3OUT = Numbers[d\_1];

delay();

Disp\_1 = 1;

if(BUTTON)

{

num++;

if(num==100) num=0;

d\_0 = num%10;

d\_1 = num/10;

while(BUTTON)

{

Disp\_2 = 0;

P3OUT = Numbers[d\_0];

delay();

Disp\_2 = 1;

Disp\_1 = 0;

P3OUT = Numbers[d\_1];

delay();

Disp\_1 = 1;

}

}

}

}

En baştan incelemeye başlayalım. İlk olarak display lerimizi aktif hale getireceğimiz P4.0 pini ve P4.1 pini ve butonumuzun bağlı olduğu pin define edilmiş.

Daha sonra display üzerinde göstereceğimiz sayının tutulduğu değişken ve o sayının birler ve onlar hanesini tutan değişkenlerimiz tanımlanmaktadır. Bu değişkenlerimiz “unsigned char” türündendir. Yani bellekte 1 byte lık yer kaplar. 0-255 arası değer alabilir. Sizde şunu alışkanlık hale getirebilirsiniz ki, tanımladığınız değişkenler için gereğinden büyük boyutlu değişkenler tanımlamayın. Yani “unsigned char” türünden bir değişken işinizi görüyorsa, boş yere “int veya unsigned int” gibi bellekte 2 byte yer kaplayan değişkenler tanımlama gerek yoktur. Çünkü bu yazılımınızı gereksiz yere yavaşlatacaktır.

Yazılımda daha sonra, display leri tararken kullanacağımız gecikme fonksiyonumuz bulunmakta, ardındanda, Display de rakamları görüntülerken kullanacağımız seven segment rakam kodları tanımlanmıştır. Daha sonrada main fonksiyonuna yani ana programa geçilmiştir.

Ana programımızda ise, ilk olarak her zamanki gibi WDT, Clock ayarlamaları yapılmış ve Port ayarlarına geçilmiştir. Port ayarlarında ise ilk olarak P1 portu tümüyle giriş yapılmış, P3 portu temizlenmiş ardındanda tümüyle çıkış yapılmış, P4 portu temizlenmiş ve tümüyle çıkış olarak şartlanmış. Daha sonrasında ise Displaylerin her ikisinin aktif etme uçları Lojik1 seviyesine çekilmiş, yani her iki displayde pasif duruma getirilmiş. Sonrasında ise sonsuz döngüye girilmiştir.

|  |  |
| --- | --- |
| **[Source code](http://www.mcu-turkey.com/msp430-uyg-8/" \l "codesyntax_2" \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-8/#codesyntax_2)** **[http://www.mcu-turkey.com/wp-content/plugins/wp-synhighlight/themes/default/images/printer.png](http://www.mcu-turkey.com/msp430-uyg-8/#codesyntax_2)** **[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)** |

d\_0 = num%10;

d\_1 = num/10;

Disp\_2 = 0;

P3OUT = Numbers[d\_0];

delay();

Disp\_2 = 1;

Disp\_1 = 0;

P3OUT = Numbers[d\_1];

delay();

Disp\_1 = 1;

kısmından bahsedecek olursak, ilk olarak sayının birler hanesi sonrada onlar hanesi hesaplanmıştır. Sonraki satırda ise, birler hanesinin gösterileceği display aktif hale getirilmiş ve P3 portuna ilgili seven segment kodu gönderilmiştir. Sonra kısa bir süre gecikmenin ardından, birler hanesinin gösterileceği display pasif duruma getirilip, onlar hanesinin gösterileceği display aktif duruma getirilmiş. Yine aynı şekilde sayının onlar hanesine ilişkin rakamın seven segment kodu P3 portuna gönderilmiş ve ardındanda kısa bir gecikme verilmiş ve onlar hanesinin gösterildiği displayde pasif duruma getirilmiştir. Sonsuz for döngüsünün içi incelendiğinde, butona basılmadığı taktirde, sürekli az önce anlatılan işlemler sürekli yapılacaktır.

Butona basıldığı durumda ise,

|  |  |
| --- | --- |
| **[Source code](http://www.mcu-turkey.com/msp430-uyg-8/" \l "codesyntax_3" \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-8/#codesyntax_3)** **[http://www.mcu-turkey.com/wp-content/plugins/wp-synhighlight/themes/default/images/printer.png](http://www.mcu-turkey.com/msp430-uyg-8/#codesyntax_3)** **[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)** |

num++;

if(num==100) num=0;

d\_0 = num%10;

d\_1 = num/10;

while(BUTTON)

{

Disp\_2 = 0;

P3OUT = Numbers[d\_0];

delay();

Disp\_2 = 1;

Disp\_1 = 0;

P3OUT = Numbers[d\_1];

delay();

Disp\_1 = 1;

}

yukarıdaki kodlar icra edilecektir. Bu kodlarıda satır satır anlatacak olursak, ilk olarak displayde gösterilen sayı bir artırılmış ve alt satırda sayının 100′e ulaşıp ulaşmadığı kontrol edilmiştir. Eğer ulaşmışsa, sayı sıfıra eşitlenmiştir. Ondan sonra ise yeni sayının birler ve onlar hanesi sürekli hesaplanmıştır. Ardından da while kısmında ise buton hala basılı durumda ise ekranın sönük kalmaması için, yukarıda anlatıldığı gibi Display1 ve Display2 belirli aralıklarla taranmıştır. Eğer buton elimiz çekmişsek, mikrodenetleyici bu kısımdan çıkarak, sonsuz for döngüsünün başına gelerek yine tarama işlemine devam edecektir. Butona basılmadığı taktirde de bu işlem devam edecektir.

Geldik bir yazımızın daha sonuna… Bir sonraki dersimizde, 7448 entegresini kullanarak, 0-99 arası yukarı ve aşağı sayıcı uygulaması yapacağız. Bir dahaki yazımızda görüşmek üzere, şimdilik hoşçakalın…

***Ferudun GÖKCEGÖZ***

***fgokcegoz@yahoo.com***