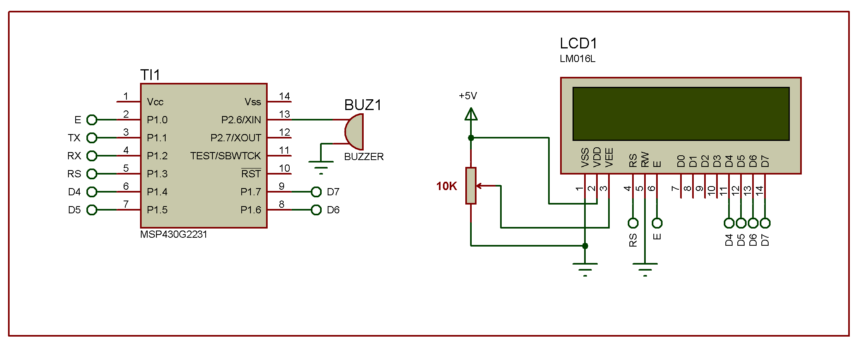
**MSP430 LaunchPad ile İlaçVakti Uygulaması**

Ferudun GÖKCEGÖZ, 02 Ocak 2012, Pazartesi



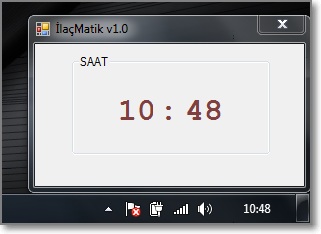
Merhaba arkadaşlar. Bu yazımda launchpad ile ekstradan sadece 2\*16 lcd ve buzzer kullanarak düzenli olarak bir veya birden fazla ilaç kullananların işini kolaylaştıracak, ilaç zamanını hatırlatma amaçlı donanım ve yazılım tasarımı yapacağız. Başta dediğimiz gibi ekstra bir donanıma ihtiyacımız yok. Sadece lcd display ve bir adet buzzer, bir de lcd nin kontrast ını ayarlamak için potansiyometre. Launchpad ile bağlantısı yapılacak olan devrenin genel şeması aşağıdaki şekilde…



Devremizin çalışmasından bahsetmek gerekirse şunları söyleyebiliriz. Launchpad i bilgisayara usb kablosu ile bağladığınızda bilgisayarda sanal bir seri port oluşturulur. Ve bu port üzerinden launchpad üzerindeki mikrodenetleyici ile seri olarak haberleşmeniz mümkün olmaktadır. Bu iş aslında tasarımızın en büyük parçası olacak. Çünkü bu sanal seri port üzerinden gerçek zamanı, bilgisayarın saatinden okuyacağız.

Bildiğimiz gibi gerçek zamanı sürekli güncel olarak takip etmek için rtc entegreleri kullanılır. Bu ürünler hem çok az güç tüketimine sahip olduğundan yıllarca çok az bir güç ile gerçek zamanı sayabiliyor hemde kaliteli bir kristal ile oldukça hassas olarak çalışabiliyorlar.

Biz bu şekilde değilde, launchpad e seri olarak sanal seri porttan belirli aralıklarla (500 msn gibi) bilgisayarın saatini C# üzerinde sa:dk şeklinde okuyup hedef mikroişlemciye göndereceğiz. C# arayüzü de aşağıdaki şekilde…



C# programı ilk olarak çalıştırıldığında 500 msn aralıklarla bilgisayarın saatini okuyup kendi penceresinde yazmaktadır. Aynı andada her yeni okuma yapıldığında seri porttan göndermektedir. Ben c# tarafını çok karışık olmaması için seri port ayarlarını değiştirebileceğimiz bir kısım eklemedim. Açıldığında COM6 – 9600 – N – 8 – 1 ayarlarında başlıyor ve sürekli olarak yapacağı işi koşturuyor. Toplam olarak seri porttan 3 byte gönderiliyor. İlk iki byte saat ve dakika bilgileri, üçüncü byte ise sonlandırma karakteri.  Yani mikroişlemci seri porttan gelen dataları kontrol ediyor ve bu sonlandırma karakteri gelen kadar gelen dataları buffer a yazıyor. Sonlandırma karakteri geldiğinde ise Gelen ilk iki byte ı saat ve dakika nın tutulduğu değişkenlere atıp güncelliyor. Ben sonlandırma karakteri olarak 0xF0 hexadecimal datasını seçtim. C# yazılımı ise aşağıdaki şekilde…

|  |  |
| --- | --- |
| **[Form1.cs](http://www.mcu-turkey.com/msp430-launchpad-ile-ilacvakti-uygulamasi/" \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-launchpad-ile-ilacvakti-uygulamasi/#codesyntax_1)** **[http://www.mcu-turkey.com/wp-content/plugins/wp-synhighlight/themes/default/images/printer.png](http://www.mcu-turkey.com/msp430-launchpad-ile-ilacvakti-uygulamasi/#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)** |

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

namespace WindowsFormsApplication1

{

public partial class Form1 : Form

{

byte[] SendBuffer = [**new**](http://www.google.com/search?q=new+msdn.microsoft.com) byte[5];

int BufferIndex = 0;

public Form1()

{

InitializeComponent();

}

private void timer1\_Tick(object sender, EventArgs e)

{

byte hour = Convert.ToByte(DateTime.Now.Hour);

byte min = Convert.ToByte(DateTime.Now.Minute);

if (hour < 10) label2.Text = "0" + hour.ToString();

else label2.Text = hour.ToString();

if (min < 10) label1.Text = "0" + min.ToString();

else label1.Text = min.ToString();

SendBuffer[0] = min;

SendBuffer[1] = hour;

SendBuffer[2] = 0xF0;

timer2.Enabled = true;

}

private void Form1\_Load(object sender, EventArgs e)

{

serialPort1.Open();

}

private void timer2\_Tick(object sender, EventArgs e)

{ if (BufferIndex == 3)

{

BufferIndex = 0;

timer2.Enabled = false;

}

else

{

serialPort1.Write(SendBuffer, (0 + BufferIndex++), 1);

}

}

}

}

Gelelim mikroişlemci tarafına. Mikroişlemci tarafı ise şu şekilde çalışıyor. C# üzerinden seri olarak gönderilen gerçek zaman bilgisi sürekli olarak takip edilip ayarlanan ilaç saatleri ile karşılaştırılıyor. Eğer ayarlanan ilaç saatleri ile eşleşme durum var ise ekranda 1 dk süre boyunca yani bir sonraki dakikaya kadar buzzer flash yaptırılarak uyarı veriliyor. Ve lcd ekranda ise “! İLAÇ ZAMANI !” uyarısı ile birlikte hangi ilacın vaktinin geldiği yazdırılıyor. Normal standart çalışma durumunda ise lcd ekranda saat bilgisi görüntülenmektedir. Mikroişlemci tarafındaki yazılım ise aşağıdaki şekilde..

|  |  |
| --- | --- |
| **[main.c](http://www.mcu-turkey.com/msp430-launchpad-ile-ilacvakti-uygulamasi/" \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-launchpad-ile-ilacvakti-uygulamasi/#codesyntax_2)** **[http://www.mcu-turkey.com/wp-content/plugins/wp-synhighlight/themes/default/images/printer.png](http://www.mcu-turkey.com/msp430-launchpad-ile-ilacvakti-uygulamasi/#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)** |

#include "io430.h"

#include "lcd\_4bit.h"

#define UART\_TXD 0x02

#define UART\_RXD 0x04

#define UART\_TBIT\_DIV\_2 (1000000 / (9600 \* 2))

#define UART\_TBIT (1000000 / 9600)

unsigned int txData;

unsigned char rxBuffer;

unsigned char birler;

unsigned char onlar;

unsigned char Time[2];

unsigned char Index;

char TEXT[16]="! ILAC ZAMANI !";

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

*/\* İLAÇ SAATLERİ, SA:DK FORMATINDA \*/*

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

unsigned char MControl[3][2]={{23,52},

{20,52},

{20,54}};

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

*//Time[0] minute(dakika)*

*//Time[1] hour(saat)*

#define OUTPUT P2OUT\_bit.P6 //Çıkış Pini

void TimerA\_UART\_init(void);

void TimerA\_UART\_tx(unsigned char byte);

void TimerA\_UART\_print(char \*string);

*//Ekrana Saati Yazan Fonksiyon*

void Lcd\_WR\_Time(void)

{

lcd\_clear();

lcd\_goto(2,3);

birler = Time[1]%10;

onlar = Time[1]/10;

lcd\_puts("SAAT = ");

lcd\_putch(onlar+'0');

lcd\_putch(birler+'0');

lcd\_putch(':');

birler = Time[0]%10;

onlar = Time[0]/10;

lcd\_putch(onlar+'0');

lcd\_putch(birler+'0');

}

void main(void)

{

unsigned char GeneralFlag;

unsigned char GeneralFlag2;

WDTCTL = WDTPW + WDTHOLD;

DCOCTL = 0x00;

BCSCTL1 = CALBC1\_1MHZ;

DCOCTL = CALDCO\_1MHZ;

P1OUT = 0x00;

P1SEL = UART\_TXD + UART\_RXD;

P1DIR = 0xFF & ~UART\_RXD;

P2OUT = 0x00;

P2SEL = 0x00;

P2DIR = 0xFF;

\_\_enable\_interrupt();

TimerA\_UART\_init();

lcd\_init();

lcd\_clear();

lcd\_goto(2,1);

lcd\_puts(" ilacMatik v1.0 ");

OUTPUT = 1;

\_\_delay\_cycles(1000000);

OUTPUT = 0;

lcd\_clear();

for (;;) *// Sonsuz Döngüye Giriliyor.*

{

\_\_bis\_SR\_register(LPM0\_bits);

if(rxBuffer==0xF0)

{

if(GeneralFlag!=1)

Lcd\_WR\_Time();

Index = 0;

OUTPUT = 0;

\_\_delay\_cycles(400000);

*//İlk ilaç saati kontrol ediliyor.*

if(Time[1]==MControl[0][0] && Time[0]==MControl[0][1])

{

if(GeneralFlag2!=1)

{

GeneralFlag = 1;

GeneralFlag2 = 1;

lcd\_clear();

lcd\_goto(1,1);

lcd\_puts(TEXT);

lcd\_goto(2,1);

lcd\_puts(" --> AFERIN ");

}

OUTPUT = 1;

}

*//ikinci ilaç saati kontrol ediliyor.*

else if (Time[1]==MControl[1][0] && Time[0]==MControl[1][1])

{

if(GeneralFlag2!=1)

{

GeneralFlag = 1;

GeneralFlag2 = 1;

lcd\_clear();

lcd\_goto(1,1);

lcd\_puts(TEXT);

lcd\_goto(2,1);

lcd\_puts(" --> MAJEZIK ");

}

OUTPUT = 1;

}

*//Üçüncü ilaç saati kontrol ediliyor.*

else if (Time[1]==MControl[2][0] && Time[0]==MControl[2][1])

{

if(GeneralFlag2!=1)

{

GeneralFlag = 1;

GeneralFlag2 = 1;

lcd\_clear();

lcd\_goto(1,1);

lcd\_puts(TEXT);

lcd\_goto(2,1);

lcd\_puts(" --> MINOSET ");

}

OUTPUT = 1;

}

else *// Uyarı durumu bitmişse bayraklar temizlenip çıkılıyor.*

{

GeneralFlag = 0;

GeneralFlag2 = 0;

}

}

else

{

Time[Index++]=rxBuffer;

}

}

}

*//------------------------------------------------------------------------------*

*// Timer A yı İki Yönlü haberleşme için hazırlayan alt program*

*//------------------------------------------------------------------------------*

void TimerA\_UART\_init(void)

{

TACCTL0 = OUT;

TACCTL1 = SCS + CM1 + CAP + CCIE;

TACTL = TASSEL\_2 + MC\_2;

}

*//------------------------------------------------------------------------------*

*// Bir byte data gönderimini sağlayan alt program*

*//------------------------------------------------------------------------------*

void TimerA\_UART\_tx(unsigned char byte)

{

while (TACCTL0 & CCIE);

TACCR0 = TAR;

TACCR0 += UART\_TBIT;

TACCTL0 = OUTMOD0 + CCIE;

txData = byte;

txData |= 0x100;

txData <<= 1;

}

*//------------------------------------------------------------------------------*

*// UART ile bir string datasını göndermeye yarayan alt program*

*//------------------------------------------------------------------------------*

void TimerA\_UART\_print(char \*string)

{

while (\*string) {

TimerA\_UART\_tx(\*string++);

}

}

*//------------------------------------------------------------------------------*

*// UART Data Göndermeye ilişkin kesme alt programı*

*//------------------------------------------------------------------------------*

#pragma vector = TIMERA0\_VECTOR

\_\_interrupt void Timer\_A0\_ISR(void)

{

static unsigned char txBitCnt = 10;

TACCR0 += UART\_TBIT;

if (txBitCnt == 0) {

TACCTL0 &= ~CCIE;

txBitCnt = 10;

}

else {

if (txData & 0x01) {

TACCTL0 &= ~OUTMOD2;

}

else {

TACCTL0 |= OUTMOD2;

}

txData >>= 1;

txBitCnt--;

}

}

*//------------------------------------------------------------------------------*

*// UART Data Almaya ilişkin kesme alt programı*

*//------------------------------------------------------------------------------*

#pragma vector = TIMERA1\_VECTOR

\_\_interrupt void Timer\_A1\_ISR(void)

{

static unsigned char rxBitCnt = 8;

static unsigned char rxData = 0;

switch (\_\_even\_in\_range(TAIV, TAIV\_TAIFG)) {

case TAIV\_TACCR1:

TACCR1 += UART\_TBIT;

if (TACCTL1 & CAP) {

TACCTL1 &= ~CAP;

TACCR1 += UART\_TBIT\_DIV\_2;

}

else {

rxData >>= 1;

if (TACCTL1 & SCCI) {

rxData |= 0x80;

}

rxBitCnt--;

if (rxBitCnt == 0) {

rxBuffer = rxData;

rxBitCnt = 8;

TACCTL1 |= CAP;

\_\_bic\_SR\_register\_on\_exit(LPM0\_bits);

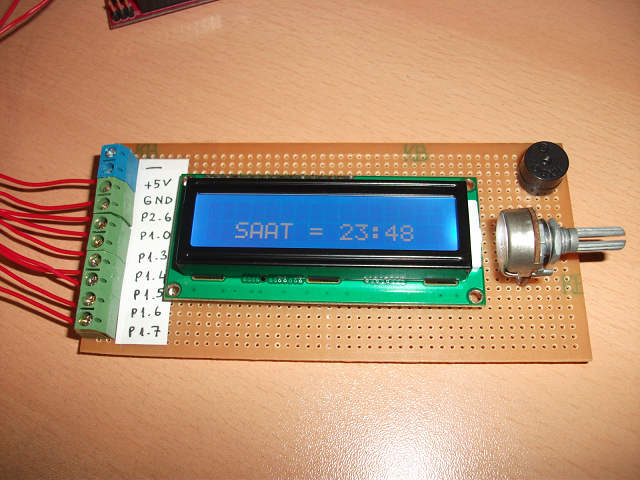
}

}

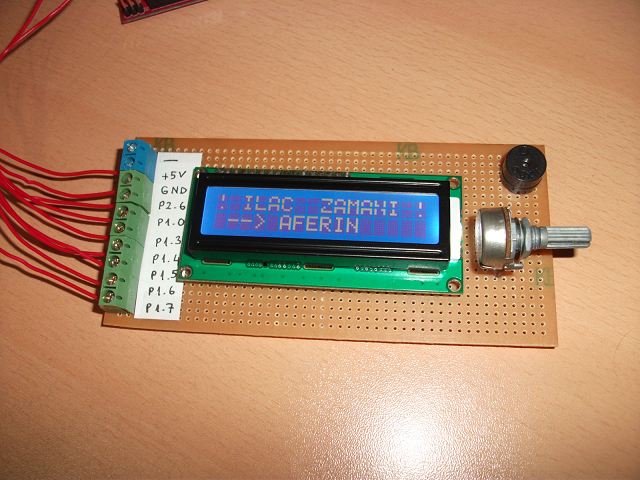
**break**;

}

}



Devrenin Normal Çalışma Durumu



Uyarı Verilirken Devrenin Çalışma Durumu

Devrenin hem yazılım tarafı hemde donanım tarafı dahada geliştirmeye müsaittir. Örneğin ilaç zamanları ve isimleri program üzerinden mikroişlemci yazılımına müdahale edilmeden değiştirilebilecek şekilde düzenlenebilir. Veya devrenin bilgisayardan bağımsız olarak çalışabilmesi adına RTC entegresi eklenip gerçek zaman o şekilde takip edilebilir.

Vakit ayırıp yazıyı okuyan herkese teşekkür ederim. Bir dahaki yazımda görüşmek dileğiyle.

Hoşçakalın…

**Ferudun GÖKCEGÖZ**

**fgokcegoz@yahoo.com**