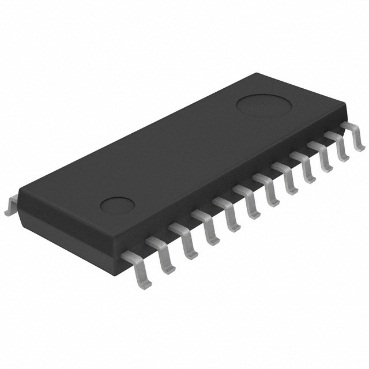
**MSP430 Uyg. 36 – Harici ADC(MCP3204) Kullanımı (3310 LCD ile)**

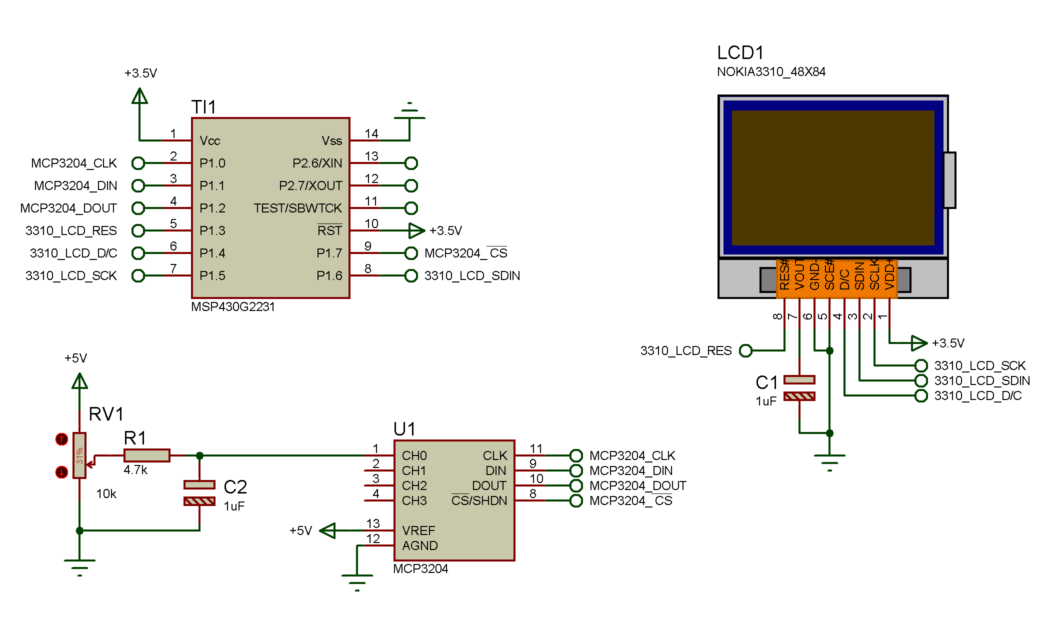
Ferudun GÖKCEGÖZ, 23 Ekim 2011, Pazar



Merhaba arkadaşlar…

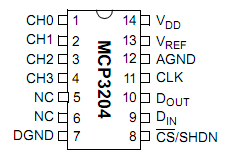
Bu yazımızda MSP430 launchpad ile Microchip firmasının ürettiği MCP3204 kodlu 12 bit ADC entegresini MSP430 ile haberleştirip, 3310 LCD ekranda göstereceğiz.  LCD ekranın içindeki sürücü entegreyle ilgili datasheet e [**buradan**](Dokumanlar/lcd_controller_pcd8544.pdf) ulaşabilirsiniz. MCP3204 ise microchip firmasının ürettiği 12 bitlik adc(analog digital converter-analog digital dönüştürücü) enetegresidir. Bu entegre içinde ilgili datasheet e [**buradan**](Dokumanlar/MCP3204.pdf) ulaşabilirsiniz. 3310 lcd ekran için gerekli C kütüphanelerini Erhan YILMAZ arkadaşımızın hazırladığı bu [**yazıda**](http://www.mcu-turkey.com/?p=15743)  veya bu [**yazıda**](http://www.mcu-turkey.com/?p=8933) bulabilirsiniz. Hadi bakalım başlayalım…

Launchpad üzerinde kuracağımız devre şemasını verelim…

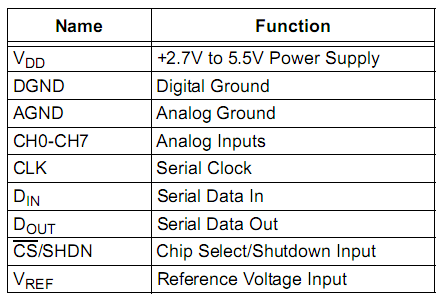


Devre şemasında da görüldüğü gibi MCP3204 entegresi 4 adet analog kanala sahiptir. MCP3208 serisi aynı entegrenin 8 kanala sahip olan modelidir. Bu iki entegrede seri haberleşme (SPI) modunu desteklemektedir. İsterseniz SPI modülü ile haberleştirmeyi gerçekleştirin, isterseniz yazılımsal olarak haberleştirmeyi gerçekleştirin, farketmez.. Her iki şekilde de kolayca entegre ile haberleşebileceksiniz.

Aşağıda MCP3204 entegresine ait  kılıf yapısını görmekteyiz…



Pinlerin özelliklerinide sıralayım…



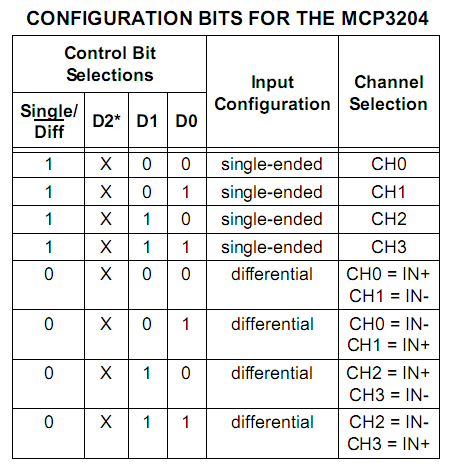
Bu entegreye ait özelliklerden bahsedecek olursak…

1. 4 adet analog kanala sahiptir. (MCP3208 8 analog kanala sahiptir.)
2. Kanallardan tek tek veya diferansiyel modda okuma yapılabilir.
3. SPI haberleşme protokolünü destekler.
4. 2.7V ~5.5V arası besleme gerilimi altında çalışabilir.
5. 100ksps @5V, 50ksps @2.7V
6. Max. 2uA bekleme akımı, çalışma akımı ise max. 400uA
7. -40 C ~+85 C arasında çalışabilme özelliği.
8. PDIP, SOIC, TSSOP kılıflarında üretilebilmektedir.

Günümüzde üretilen modern mikrodenetleyicilerin birçoğunda dahili adc modülü bulunmaktadır. Dahili adc modülleri bazen çözünürlükleri ve ölçme hataları nedeniyle yetersiz gelebilmektedir. Bu şekilde harici adc kullanımı daha kaliteli bir ölçüm sağlayabilir. Ayrıca şunuda belirtmek gerekirki, dahili adc modüllerinin referansı istenilen şekilde analog olarak ayarlananamaktadır. Bu şekilde harici adc kullanımı ölçeceğimiz referans gerilimi aralığını geniş bir seviyede tutabilmemizi sağlar. Örneğin MCP3204/8 entegreleri için 0.25V ~Vdd arasında bir referans aralığına sahiptir.

Bunun önemini şu şekilde açıklayabiliriz. Örneğin dahili adc modülünün referans gerilimi +5V. Ve bizde 0V ~ 1V aralığında hassas bir ölçüm yapmak istiyoruz. Referans gerilimi 5V olan bir adc, 0V~5V arasında (12bit lik ise) 0~4095 arasında digital bir değer üretecektir. Bizim ölçmek istediğimiz alan ise 0~1V aralığında olduğu için, bu aralıkta 0~819 arasında bir değer üretecektir. Eğer referans gerilimi 1V olarak ayarlarsak, adc nin üreteceği 0~4095 arasındaki digital değer 0V~1V aralığına denk gelecektir. Ve biz 5 kat hassas bir ölçüm yapmış olacağız..

**MCP3204 ADC Entegresi İçin Konfigürasyon Ayarları**



MCP3204 entegresini kullanıma hazırlamak için öncelikle konfigürasyon ayarlarının yapılması gerekmektedir. Bu ayarlardan bahsedecek olursak, adc entegremiz 2 farklı modda çalışmaktadır. Bu iki farklı moda göre toplamda 8 adet farklı çalışma yöntemi bulunmaktadır. İlk 4 yöntem, CH0-CH3 arasında tüm kanalları kendi içerisinde tek olarak okumak. Son 4 yöntem ise, kanalları diferansiyel modda okumaktır. Yani adc entegremiz çıkış olarak iki farklı kanalı okuyup, arasındaki gerilim farkıyla orantılı olarak bize dijital çıkış üretmektedir. Buda bazı uygulamalarda bize esneklik sağlamaktadır.

MCP3208 entegresi için ise toplamda 16 farklı mod bulunmaktadır. Bunlardan ilk 8 i, her kanalı tek tek okumak. son8 i ise yine diferansiyel modda okuma yapmaktır. MCP320X entegrelerinin modlarına ilişkin konfigürasyon ayarlarını entegreden okuma yapılırken entegreye gönderilmesi gerekir. Entegreyle haberleşmek için okuma yazma timing diagramlarına ilgili datasheet lerden ulaşabilirsiniz… 3310 LCD ekran dan ise bu yazı içerisinde bahsetmeyeceğim. Yazınız başında söylediğim gibi, Erhan Yılmaz arkadaşımızın ilgili yazılarında daha detaylı bilgi edinebilirsiniz..

Gelelim uygulamaya ait yazılıma…

|  |  |
| --- | --- |
| **[MCP3204.c](http://www.mcu-turkey.com/msp430-uyg-36/" \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-36/#codesyntax_1)** **[http://www.mcu-turkey.com/wp-content/plugins/wp-synhighlight/themes/default/images/printer.png](http://www.mcu-turkey.com/msp430-uyg-36/#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 "MCP3204.h"

unsigned char CONFIG\_BYTE;

void MCP3204\_init(unsigned char config)

{

MCP3204\_CS=1;

switch(config)

{

case 1: CONFIG\_BYTE = 0xC0; **break**; *// Single = CH0*

case 2: CONFIG\_BYTE = 0xC8; **break**; *// Single = CH1*

case 3: CONFIG\_BYTE = 0xD0; **break**; *// Single = CH2*

case 4: CONFIG\_BYTE = 0xD8; **break**; *// Single = CH3*

case 5: CONFIG\_BYTE = 0x80; **break**; *// Diff. = CH0-CH1*

case 6: CONFIG\_BYTE = 0x88; **break**; *// Diff. = CH1-CH0*

case 7: CONFIG\_BYTE = 0x90; **break**; *// Diff. = CH2-CH3*

case 8: CONFIG\_BYTE = 0x98; **break**; *// Diff. = CH3-CH2*

}

}

void WriteByte(unsigned char data, unsigned char num)

{

unsigned char i;

\_\_delay\_cycles(2);

for(i=0;i<num;i++)

{

MCP3204\_CLK=0;

if(data & 0x80) MCP3204\_DIN=1;

else MCP3204\_DIN=0;

data<<=1;

\_\_delay\_cycles(50);

MCP3204\_CLK=1;

\_\_delay\_cycles(50);

}

}

unsigned int ReadByte(unsigned char num)

{

unsigned char i;

unsigned int data=0;

for(i=0; i<num; i++)

{

MCP3204\_CLK=0;

\_\_delay\_cycles(50);

if(MCP3204\_DO) data |= 0x01;

else data &= ~0x01;

data<<=1;

MCP3204\_CLK=1;

\_\_delay\_cycles(50);

}

return data;

}

unsigned int Read\_MCP3204(void)

{

unsigned int data;

MCP3204\_CLK=0;

MCP3204\_DIN=1;

MCP3204\_CS =0;

WriteByte(CONFIG\_BYTE,7);

data=ReadByte(12)>>1;

MCP3204\_CS=1;

return(data);

}

|  |  |
| --- | --- |
| **[MCP3204.h](http://www.mcu-turkey.com/msp430-uyg-36/" \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-36/#codesyntax_2)** **[http://www.mcu-turkey.com/wp-content/plugins/wp-synhighlight/themes/default/images/printer.png](http://www.mcu-turkey.com/msp430-uyg-36/#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)** |

#ifndef \_MCP3204\_H\_

#define \_MCP3204\_H\_

#define MCP3204\_CLK P1OUT\_bit.P0

#define MCP3204\_DIN P1OUT\_bit.P1

#define MCP3204\_DO P1IN\_bit.P2

#define MCP3204\_CS P1OUT\_bit.P7

**extern** void MCP3204\_init(unsigned char);

**extern** void WriteByte(unsigned char,unsigned char);

**extern** unsigned int ReadByte(unsigned char);

**extern** unsigned int Read\_MCP3204(void);

#endif //\_MCP3204\_H\_

void MCP3204\_init(unsigned char) isimli alt programımızla, entegremizi kullanıma hazırlıyoruz. Bu fonksiyonumuzun parametresi olarak çalışma moduna ilişkin bir byte lık data almaktadır.

void WriteByte(unsigned char,unsigned char) alt programımız ise entegreye bir byte lık bilgi yazmaya yarar.

unsigned int ReadByte(unsigned char) altprogramımız ise entegreden iki byte lık bilgi okumamıza yarar.

unsigned int Read\_MCP3204(void) alt programımız ise adc den digital değer okumamızı sağlar. Bu fonksiyonumuz geri dönüş değeri olarak 12 bitlik adc sonucunu içerir.

|  |  |
| --- | --- |
| **[main.c](http://www.mcu-turkey.com/msp430-uyg-36/" \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-36/#codesyntax_3)** **[http://www.mcu-turkey.com/wp-content/plugins/wp-synhighlight/themes/default/images/printer.png](http://www.mcu-turkey.com/msp430-uyg-36/#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)** |

#include "io430.h"

#include "MCP3204.h"

#include "Nokia3310.h"

unsigned int value;

unsigned int voltage;

unsigned char String[6];

void main(void)

{

WDTCTL = WDTPW + WDTHOLD;

DCOCTL=CALDCO\_1MHZ;

BCSCTL1=CALBC1\_1MHZ;

P1DIR = 0xFB;

Nokia3310\_Init();

MCP3204\_init(1);

\_\_delay\_cycles(500000); *//~500 ms gecikme*

Nokia3310\_gotoxy(0,0); Nokia3310\_puts(" MCP3204 ");

Nokia3310\_gotoxy(0,1); Nokia3310\_puts(" 12 Bit ADC ");

Nokia3310\_gotoxy(0,2); Nokia3310\_puts(" UYGULAMASI ");

for(;;)

{

value = Read\_MCP3204();

voltage = (unsigned int)(value\*0.1221);

Nokia3310\_gotoxy(24,4);

String[0]=(voltage/100%10)+'0';

String[1]='.';

String[2]=(voltage/10%10)+'0';

String[3]=(voltage%10)+'0';

String[4]='V';

String[5]='**\0**';

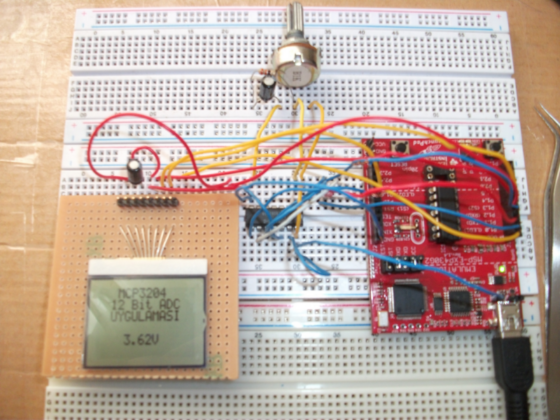
Nokia3310\_puts(String);

\_\_delay\_cycles(500000);

}

}

Uygulamaya ait bir fotoğraf paylaşalım…



Görüldüğü gibi MSP430 Launchpad ile Harici adc den doğru bir şekilde gerilim değerlerimizi okuduk. Bu arada 3310 LCD ekran kullanımına ilişkin yazıları hazırlayan Erhan Yıldız arkadaşıma çok teşekkür ederim. Bir başka uygulamamızda görüşmek üzere. Şimdilik Hoşçakalın…

**Ferudun GÖKCEGÖZ**

**fgokcegoz@yahoo.com**