دانشگاه صنعتی اصفهان
دانشکده برق و کامپیوتر

دستورکار آزمایشگاه طراحی مدارهای واسط

آزمایش ششم

تهیه کننده :مهران صفایانی

پاییز۹۵

آشنایی با مبدل قیاسی به رقمی و حسگر دما

❖اهداف

- **آشنایی با نحوه تبدیل سیگنال قیاسی به رقمی**
- **استفاده از مبدل قیاسی به رقمی**
- **استفاده از حسگر دما LM35**

❖مقدمه:

در دنیای خارج از میکرو تمامی مقادیر و سیگنال ها و کمیت ها پیوسته هستند ، اما در دنیای دیجیتال کمیت ها و سیگنال ها شکل گسسته به خود می گیرند . بنابراین برای به کار بردن این مقادیر و کمیت ها در سیستم های دیجیتال ناگریز به استفاده از مبدلی برای تبدیل سیگنال های آنالوگ به دیجیتال می باشیم . در این آزمایش با نحوه تبدیل سیگنالهای آنالوگ به دیجیتال آشنا شده و سپس اطلاعات یک حسگر دما به درون میکروکنترلر خوانده می شود.

سنسور ها مقدار یک کمیت آنالوگ را به ولتاژ یا جریان تبدیل میکند، سپس این ولتاژآنالوگ به مبدل آنالوگ به دیجیتال میکرو داده میشود و مبدل آنالوگ به دیجیتال با استفاده از روش هایی مقدار ولتاژ را به کمیت دیجیتال متناظرتبدیل میکند، سپس این مقدار دیجیتال با اعمال ریاضی به مقدارعددی متناظر تبدیل میشود و روی LCD و 7- segment نمایش داده می شود .

❖ویژگی های ADC میکروکنترلر LPC1768 :

- دقت تبدیل ۱۲ بیت
- ۸ کانال ورودی
- قابلیت کاهش توان
- محدوده اندازه گیری بین VREFN وVREFP
- فرکانس نمونه برداری 200KHz
- دارای وضعیت Burst و نرم افزاری (معمولی)

❖بررسی رجیسترها و نحوه استفاده از ADC :

همانطور که گفته شد از ADC میکرو LPC1768 در دو حالت نرم افزاری و Burst می توان استفاده نمود.در حالت نرم افزاری برای خواندن مقدار مورد نظر هربار میکرو تنظیماتی را انجام و سپس مقدار خوانده شده را به ما می دهد.۸ کانال ADC میکروکنترلر LPC1768 به پایه های زیر متصل هستند:

AD0.0 (A/D converter 0 input 0) [P0.23]
AD0.1 (A/D converter 0 input 1) [P0.24]
AD0.2 (A/D converter 0 input 2) [P0.25]
AD0.3 (A/D converter 0 input 3) [P0.26]
AD0.4 (A/D converter 0 input 4) [P1.30]
AD0.5 (A/D converter 0 input 5) [P1.31]
AD0.6 (A/D converter 0 input 6) [P0.3]
AD0.7 (A/D converter 0 input 7) [P0.2]

برای استفاده از ADC ابتدا با ید دو کار انجام داد :

I. فعال کردن رجیستر توان مربوط به ADC
II. تنظیمات رجیستر مربوط به PINSEL را در وضعیت ADC قرار دهیم.به عنوان مثال برای استفاده از کانال ADC0.0 تنظیمات زیر را انجام می دهیم:

```
LPC_PINCON->PINSEL1 = 1<< 14;      // p0.23 select as ADC0.0
LPC_SC->PCONP |= (1 << 12);        // Enable power to AD block
```

بررسی رجیسترADCR (رجیستر کنترل مبدل A/D) : بیت های صفر تا ۷ این رجیستر جهت انتخاب کانال ADC استفاده می شود و در مد نرم افزاری تنها یکی از این بیت ها می تواند یک باشد.به عنوان مثال جهت انتخاب کانال صفر یعنی AD0.0 از دستور زیر استفاده می نماییم :

```
LPC_ADC->ADCR |= 0x01;         //select AD0.0
```

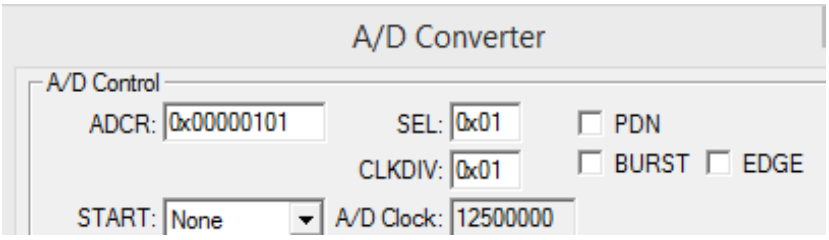
بیت های ۸ تا ۱۵ به نام CLKCIV جهت تنظیم کردن کلاک ADC استفاده می شود. براساس مقداری که به آن می دهیم فرکانس کلاک ADC تقسیم می شود و حداکثرکلاک ADC 13MHz است.رابطه این تقسیم در زیر آمده است :

```
CLKDIV=(PCLK/ADCLK)-1

ADCLK=PCLK/(CLKDIV+1)
```

به عنوان مثال با PCLK=25MHz جهت داشتن فرکانس کلاک ADC ،12.5MHzبیت های ۸ تا ۱۵ به صورت زیر مقدار دهی می شوند:

```
LPC_ADC->ADCR |= 0x0100;       //ADCLK is 12.5 MHz
```



قسمت مهم بعدی این رجیستر بیت شماره ۲۱ یعنی PDN است که پاور ADC را فعال می نماید:

```
LPC_ADC->ADCR |= 0x200000;
```

بیت های شماره ۲۴ تا ۲۶ این رجیستر نیز جهت START نمونه برداری استفاده می شوند ، وقتی مقدار 1 را به آن ها دهیم، عملیات تبدیل شروع به اجرا می شود. البته شروع نمونه برداری را توسط قسمت های دیگر مثل پایه P2.10 و یا تایمرها نیز می توان کنترل نمود ، که در تصویرهای زیر لیست قسمت هایی که می توانند عملیات کنترل شروع نمونه برداری را انجام دهند آمده است :

26:24	START	When the BURST bit is 0, these bits control whether and when an A/D conversion is started:	0
	000	No start (this value should be used when clearing PDN to 0).	
	001	Start conversion now.	
	010	Start conversion when the edge selected by bit 27 occurs on the P2.10 / EINT0 / NMI pin.	
	011	Start conversion when the edge selected by bit 27 occurs on the P1.27 / CLKOUT / USB_OVRCRn / CAP0.1 pin.	
	100	Start conversion when the edge selected by bit 27 occurs on MAT0.1. Note that this does not require that the MAT0.1 function appear on a device pin.	
	101	Start conversion when the edge selected by bit 27 occurs on MAT0.3. Note that it is not possible to cause the MAT0.3 function to appear on a device pin.	
	110	Start conversion when the edge selected by bit 27 occurs on MAT1.0. Note that this does not require that the MAT1.0 function appear on a device pin.	
	111	Start conversion when the edge selected by bit 27 occurs on MAT1.1. Note that this does not require that the MAT1.1 function appear on a device pin.	

```
LPC_ADC->ADCR |= 0x01000000; // Start A/D Conversion
```

رجیستر ADGDR : این رجیستر وضعیت مبدل آنالوگ به دیجیتال را مشخص می نماید و یکی از بیتهای مهم این رجیستر برای ما بیت شماره 31 می باشد که به وسیله این بیت متوجه می شویم که تبدیل انجام شده است یا نه و زمانی که تبدیل کامل شد این بیت یک می شود.

رجیستر AD0DR0 تا AD0DR8 : در این رجیستر مقدار دیجیتال سیگنال تبدیل شده در بیتهای ۴ تا ۱۵ قرار می گیرد و برای خواندن ADC از این رجیستر استفاده می نماییم.

```
i = LPC_ADC->ADDR0;
```

برنامه نمونه اول: در این برنامه قصد داریم تا کانال AD0 را راه اندازی کرده و سپس مقدار تبدیل شده را برگرداند.

```
unsigned short ADC_Read(void)
{
      unsigned int i;
      LPC_PINCON -> PINSEL1 = 0x00004000; // p0.23 select as ADC0.0
      LPC_SC->PCONP |= (1 << 12);        // Enable power to AD block
      LPC_ADC->ADCR |= 0x01;             //select AD0.0
      LPC_ADC->ADCR |= 0x0100;           //ADCLK is 12.5 MHz
      LPC_ADC->ADCR |= 0x200000;         //Power up,1 << 21
      LPC_ADC->ADCR |= 0x01000000;       // Start A/D Conversion
      while ((LPC_ADC->ADGDR & 0x80000000) == 0); // Wait for end of
      A/D Conversion
      i = LPC_ADC->ADDR0;                // Read A/D Data Register
      return (i >> 4) & 0x0FFF;          // bit 4:15 is 12 bit AD value
}
```

برنامه نمونه دوم: برنامه ای بنوسید که به وسیله سنسور دما lm35 دما را خوانده و بر روی LCD نمایش دهد.

```
#include <lpc17xx.h>
#define LCD_LPC1768
#define LCD_PORT_2
#define LCD_RS     0
#define LCD_RW     1
#define LCD_E      2
#define LCD_DB4    4
#define LCD_DB5    5
#define LCD_DB6    6
#define LCD_DB7    7
#include "lcd.h"
void delay(int w){
      while(w--);
}
int main(){
      int i = 0;
      LPC_GPIO2 -> FIODIR = 0xffffffff;
      LPC_PINCON -> PINSEL1 = 1<<14;   // Select P23.0 as ADC
      LPC_SC -> PCONP |= (1<<12);      // Enable power to AD block
      LPC_ADC -> ADCR |= 0x01;         // Select ADC0.0
      LPC_ADC -> ADCR |= 0x0400;       // ADC Clock is 25MHz / 5
      LPC_ADC -> ADCR |= 0x200000;     // Power up,1 << 21
```

```
while(1){
            LPC_ADC -> ADCR |= (0<<24); // Stop ADC
            LPC_ADC -> ADCR |= (1<<24); // Start ADC
            while((LPC_ADC -> ADGDR & 0x80000000) == 0); // Wait for
            end of A/D Conversion
            i = (LPC_ADC -> ADGDR & 0xfff0);
            i = i / 12.41; //(3.3V / 2^12)=0.0806 mV   10mV / 0.0806
            = 12.41 unit
            lcd_clear();
            lcd_gotoxy(1,1);
            lcd_puts(i);
            delay(2000000);
      }
}
```

❖دستورکار

I. برنامه ای بنویسید که هر ده ثانیه یکبار اطلاعات دما را خوانده و بروی نمایشگر نشان دهد

II. برنامه ای بنویسید که در صورت فشردن یک کلید اطلاعات دما بروی نمایشگر بروز رسانی گردد