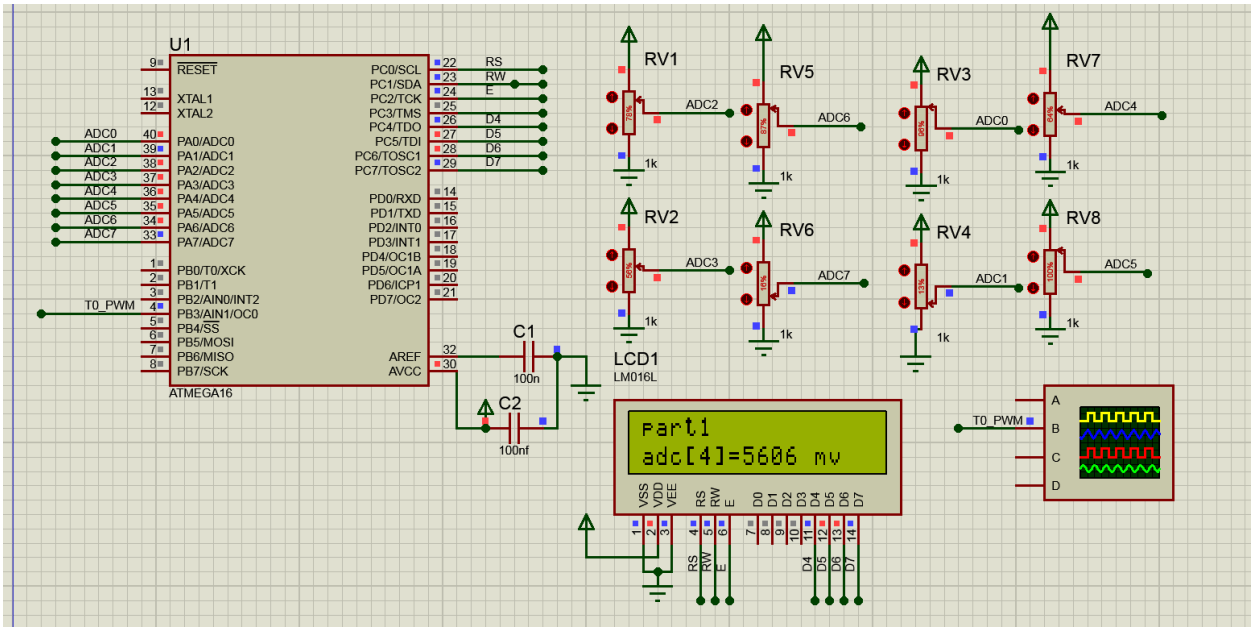
Session6 analog to digital convertor



اتمگا۱۶، فقط یک واحد adc داره که میتونه ۸تاکانال را سرویس دهی کنه که این ۸تاکانال از طریق یه مالتی پلکسر به سمت بخش adc هدایت میشوند یعنی در هرلحظه، ما فقط یکی از این کانال هارو میتوانیم به adc منتقل کنیم.

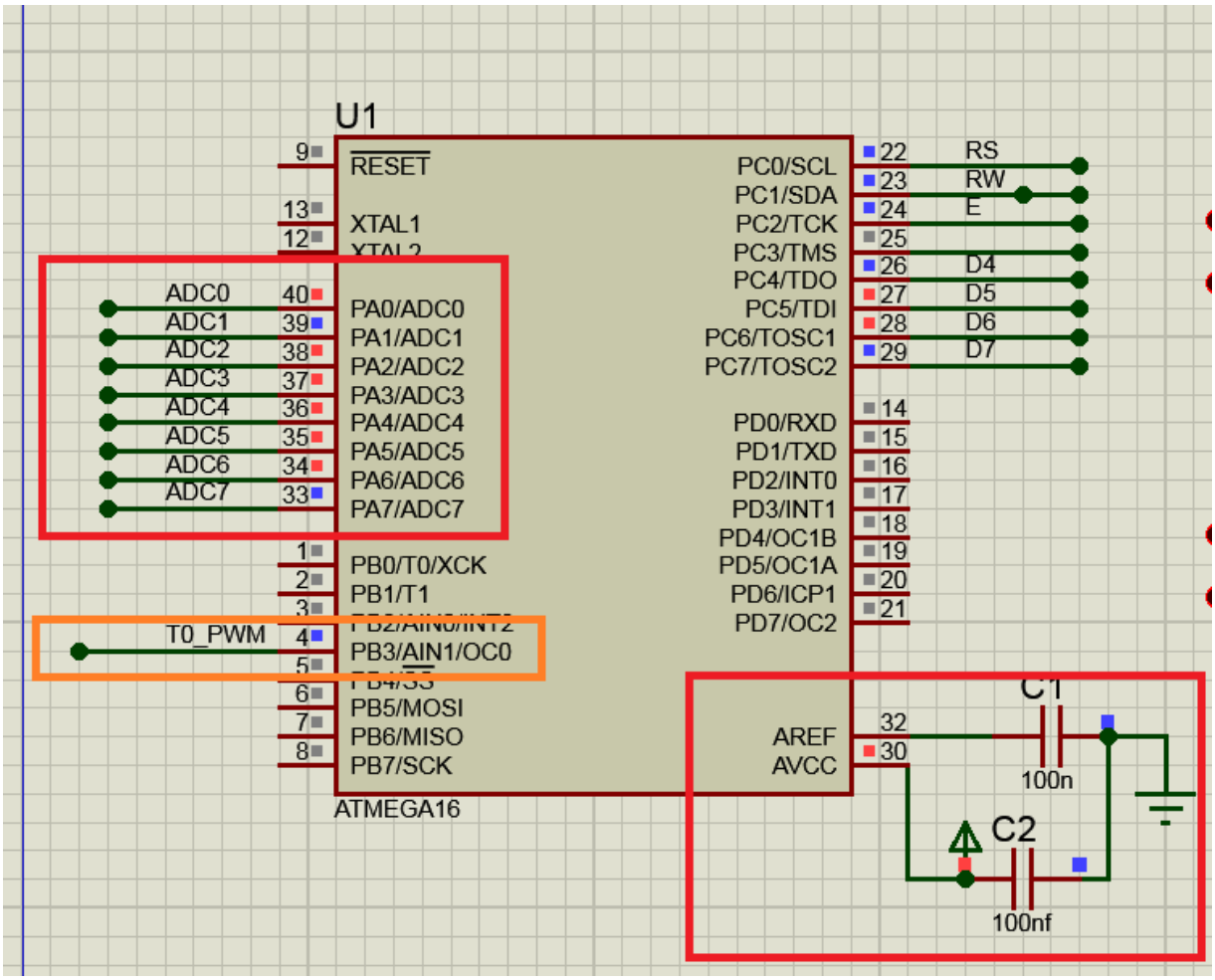
مقدار ۸ تاورودی را به کمک یک واحد مبدل adc میتوانیم بخوانیم. بعدش میتوانیم اطلاعات سنسور دما را روی lcd نمایش بدیم

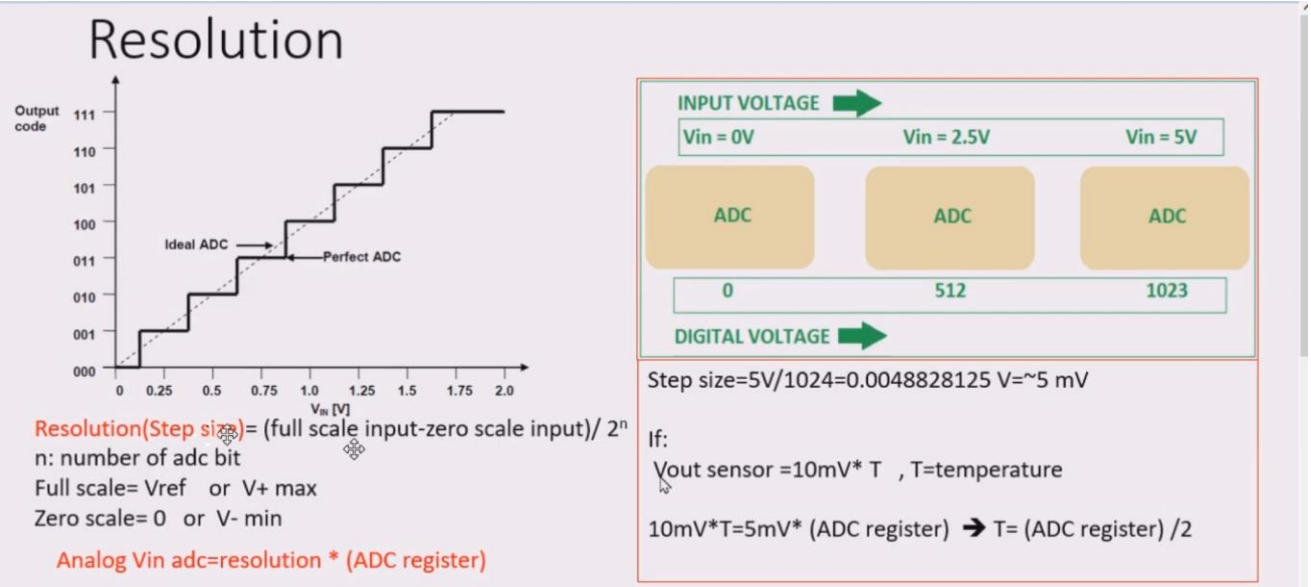
تغذیه ی بخش adc متفاوت از تغذیه ی بخش دیجیتال است و دوتا پایه ی زیر را داریم براش.

AREF,AVCC

خصوصیات اصلی ADC: ۱. دقت اندازه گیری برحسب بیت که ۸بیت یا ۱۰ بیت است.

حداقل ولتاژ قابل اندازه گیری را STEP SIZE میگویند. هرچه تعداد بیت مبدل بیشتر باشه استپ سایز، کمتر میشه.


U1
9
RESET
13
XTAL1
12
ADC0
ADC1
ADC2
ADC3
ADC4
ADC5
ADC6
ADC7
40
39
38
37
36
35
34
33
PA0/ADC0
PA1/ADC1
PA2/ADC2
PA3/ADC3
PA4/ADC4
PA5/ADC5
PA6/ADC6
PA7/ADC7
1
2
3
T0_PWM
4
5
6
7
8
PB0/T0/XCK
PB1/T1
PB3/AIN1/OC0
PB5/MOSI
PB6/MISO
PB7/SCK
ATMEGA16
PC0/SCL
PC1/SDA
PC2/TCK
PC3/TMS
PC4/TDO
PC5/TDI
PC6/TOSC1
PC7/TOSC2
22
23
24
25
26
27
28
29
RS
RW
E
D4
D5
D6
D7
PD0/RXD
PD1/TXD
PD2/INT0
PD3/INT1
PD4/OC1B
PD5/OC1A
PD6/ICP1
PD7/OC2
14
15
16
17
18
19
20
21
AREF
AVCC
32
30
C1
100n
C2
100nf



Resolution(step size) = (full scale – zero scale)/(2^n)

N = number of adc bit تعداد بیت های نمونه برداری

Full scale = Vref or V+ max

دامنه ی مثبت سیگنال یا حداکثر دامنه ای که اندازه میگیره.

Zero scale = 0 or V- min

در یک سیگنال سینوسی، حداقل دامنه ی نوسان

Analog Vin adc = resolution * (ADC register)

محاسبه ی مقدار انالوگ یه سیگنالی که به adc متصل کردیم. بعدش میتوانیم این عدد را روی lcd نمایش بدیم.

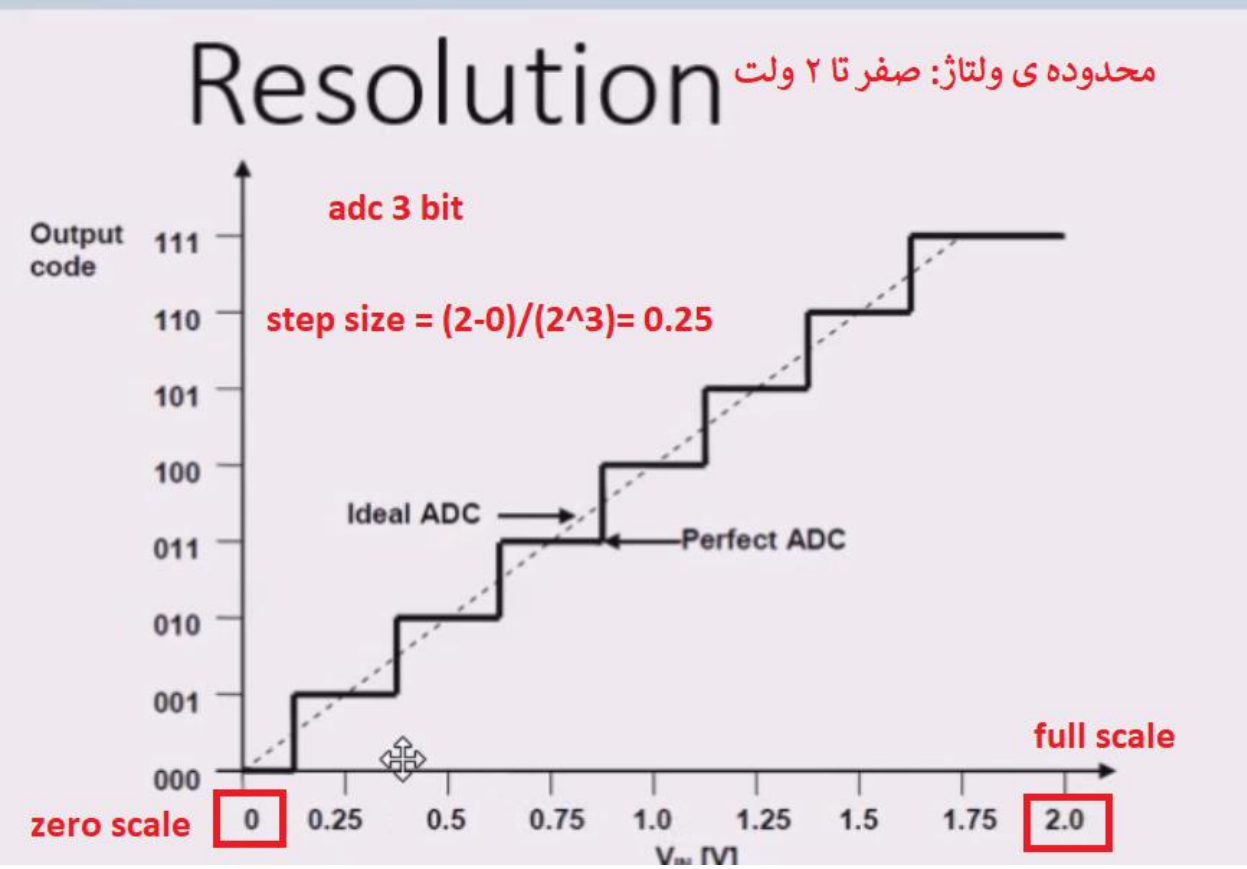
هرچه تعداد بیت ها بیشتر باشه(n بزرگتر باشه) بازه های کوچکتری را میتوانیم اندازه بگیریم و باعث میشه اندازه گیری ما، دقیق تر بشه.

در انتخاب یک adc به دوتا چیز باید دقت کنیم:

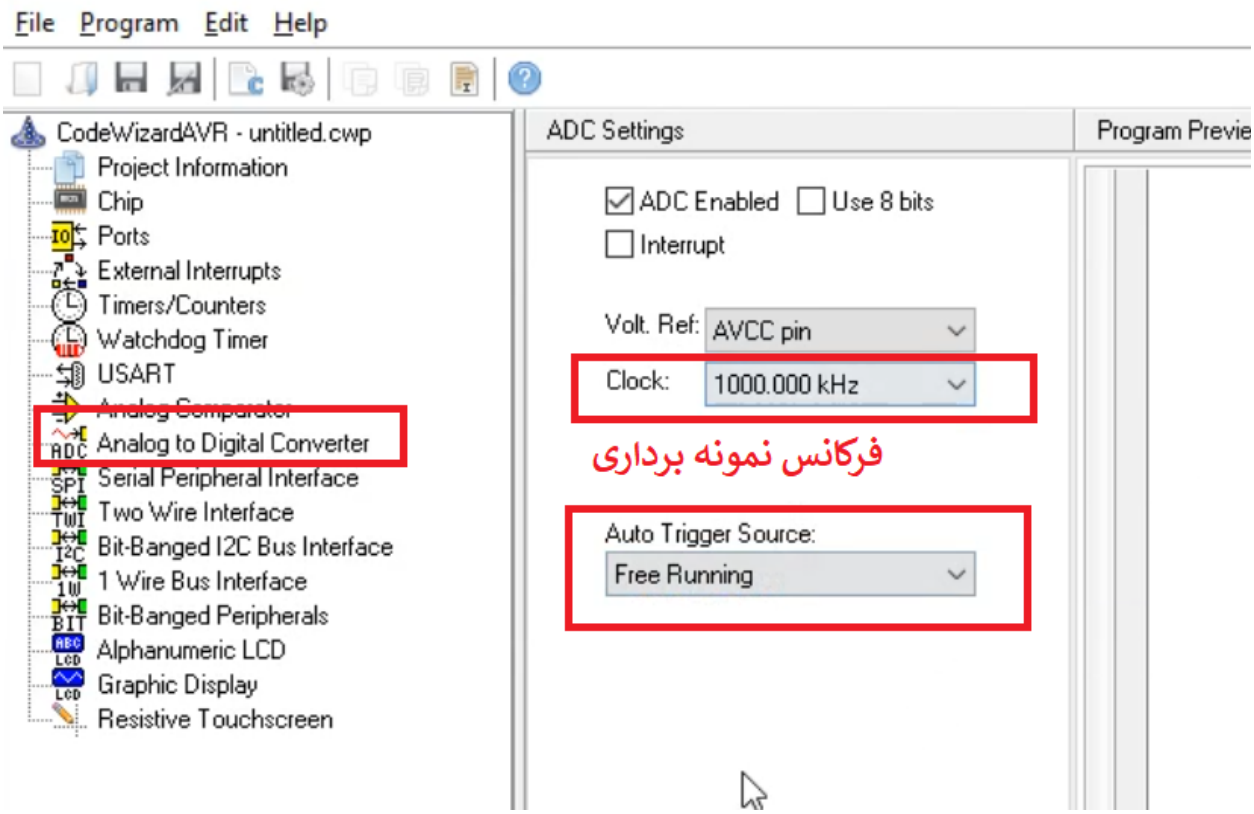
۱. تعداد بیت های نمونه برداری

۲. فرکانسی که میتواند نمونه برداری ها را انجام بده و سیگنالی که قراره به adc متصل کنیم.

اگر بخاهیم استپ سایز های خیلی کوچکتر را اندازه بگیریم، باید سراغ مبدل هایی بریم که دارای n بیشتری هستند(تعداد بیت بیشتری دارند)



اگر فرکانس سیگنال ورودی خیلی زیاد باشه باید از adc هایی با سرعت بالاتر استفاده کنیم تا بتواند تغییرات سیگنال را برای ما شناسایی کنه.



استفاده از حالت ۱۰ بیتی بدون وقفه

```
// Voltage Reference: AVCC pin
#define ADC_VREF_TYPE ((0<<REFS1) | (1<<REFS0) | (0<<ADLAR))
// Read the AD conversion result
unsigned int read_adc(unsigned char adc_input)
{
ADMUX=adc_input | ADC_VREF_TYPE;
// Delay needed for the stabilization of the ADC input voltage
delay_us(10);
// Start the AD conversion
ADCSRA|=(1<<ADSC);
// Wait for the AD conversion to complete
while ((ADCSRA & (1<<ADIF))==0);
ADCSRA|=(1<<ADIF);
return ADCW;
}
```

ما از حالت ۱۰ بیتی استفاده کردیم ⇦ پس خروجی این تابع ⇦ integer

چون میخاد یه عدد ۱۶ بیتی را برای ما برگشت بزنه

انتخاب کانال موردنظر

انتظار برای تمام شدن فرایند نمونه برداری

مبدل ۱۰ بیتی adc بدون وقفه

به کمک این زیربرنامه میشه هرکدام از کانال های موردنظر در adc را خواند

ADC WORD: یک رجیستر ۱۶ بیتی است

```
// ADC initialization
// ADC Clock frequency: 1000.000 kHz
// ADC Voltage Reference: AVCC pin
// ADC Auto Trigger Source: Free Running
ADMUX=ADC_VREF_TYPE;
ADCSRA=(1<<ADEN) | (0<<ADSC) | (1<<ADATE) | (0<<ADIF) | (0<<ADIE) | (0<<ADPS2) | (1<<ADPS1) | (1<<ADPS0);
SFIOR=(0<<ADTS2) | (0<<ADTS1) | (0<<ADTS0);
```

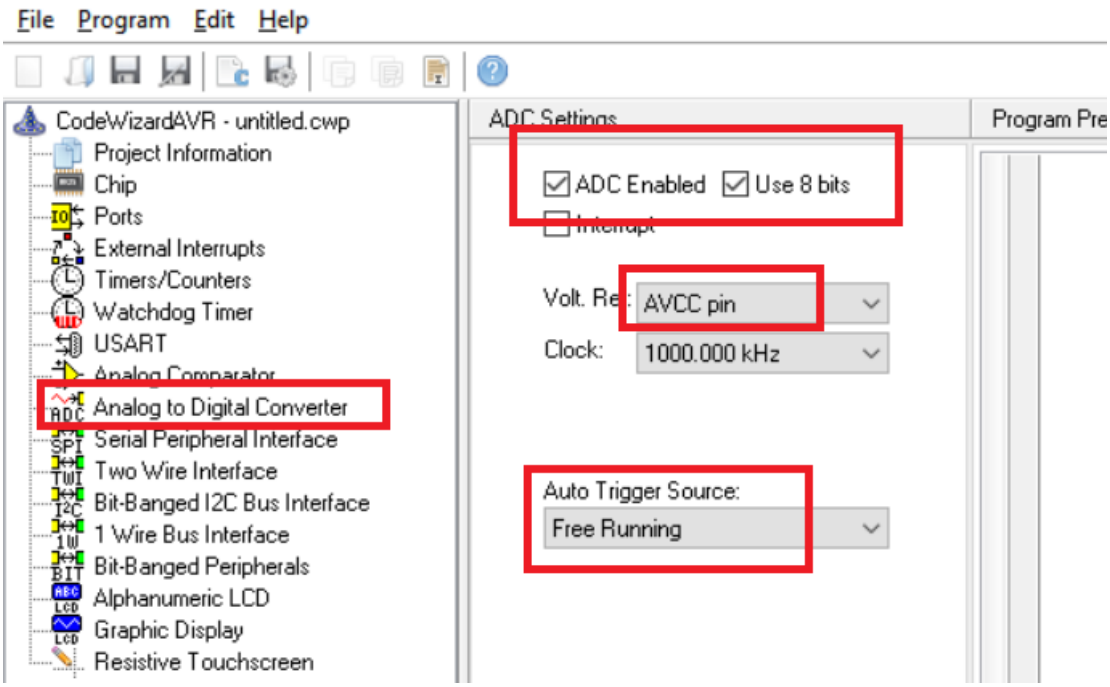
خواندن یکی از پورت ها و نمایش روی LCD

```
void main(void)
{
int data;
```

خواندن کانال ۵

```
while (1)
      {
      // Place your code here
       data = read_adc(5);
       //display data on LCD
      }
}
```

استفاده از حالت۸ بیتی بدون وقفه



```
// Voltage Reference: AVCC pin
#define ADC_VREF_TYPE ((0<<REFS1) | (1<<REFS0) | (1<<ADLAR))

// Read the 8 most significant bits
// of the AD conversion result
unsigned char read_adc(unsigned char adc_input)
{
ADMUX=adc_input | ADC_VREF_TYPE;
// Delay needed for the stabilization of the ADC input voltage
delay_us(10);
// Start the AD conversion
ADCSRA|=(1<<ADSC);
// Wait for the AD conversion to complete
while ((ADCSRA & (1<<ADIF))==0);
ADCSRA|=(1<<ADIF);
return ADCH;
}
```

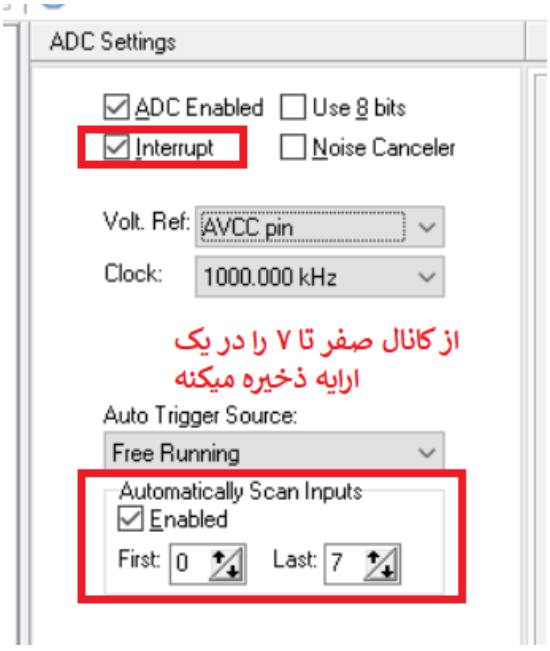
خروجی اینجا از نوع **char** است

چون خروجی ۸ بیت است نیازی به استفاده از **int** نیست

۸ بیت پرارزش را برمیگردانه

```
// ADC initialization
// ADC Clock frequency: 1000.000 kHz
// ADC Voltage Reference: AVCC pin
// ADC Auto Trigger Source: Free Running
// Only the 8 most significant bits of
// the AD conversion result are used
ADMUX=ADC_VREF_TYPE;
ADCSRA=(1<<ADEN) | (0<<ADSC) | (1<<ADATE) | (0<<ADIF) | (0<<ADIE) | (0<<ADPS2) | (1<<ADPS1) | (1<<ADPS0);
SFIOR=(0<<ADTS2) | (0<<ADTS1) | (0<<ADTS0);
```

استفاده از حالت ۱۰ بیتی با وقفه



```
// Declare your global variables here

#define FIRST_ADC_INPUT 0
#define LAST_ADC_INPUT 7
unsigned int adc_data[LAST_ADC_INPUT-FIRST_ADC_INPUT+1];
// Voltage Reference: AVCC pin
#define ADC_VREF_TYPE ((0<<REFS1) | (1<<REFS0) | (0<<ADLAR))

// ADC interrupt service routine
// with auto input scanning
interrupt [ADC_INT] void adc_isr(void)
{
static unsigned char input_index=0;
// Read the AD conversion result
adc_data[input_index]=ADCW;
// Select next ADC input
if (++input_index > (LAST_ADC_INPUT-FIRST_ADC_INPUT))
   input_index=0;
ADMUX=(FIRST_ADC_INPUT | ADC_VREF_TYPE)+input_index;
// Delay needed for the stabilization of the ADC input voltage
delay_us(10);
// Start the AD conversion
ADCSRA|=(1<<ADSC);
}
```

ارایه ۸تایی که تایپش int است

اولین کانالی که خوانده میشه

اخرین کانالی که خوانده میشه

اندازه ی ارایه به تعداد کانال هایی است که قراره خوانده شه

ما داریم به صورت ۱۰ بیتی کار میکنیم پس تایپ ارایه integer

چه زمانی اینتراپت اتفاق میفتد؟

زمانی که فرایند نمونه برداری انجام شده باشه و داده ی موردنظر، داخل رجیستر قرار بگیره

پس ما باید مقدار رجیستر را بخوانیم و داخل ارایه بریزیم.

پس هربار که نمونه برداری تمام شد وارد این زیربرنامه وقفه میشه و مقدار رجیستر را داخل ارایه میریزه.

پس در این ارایه، همیشه اخرین مقدار نمونه برداری شده را داریم.

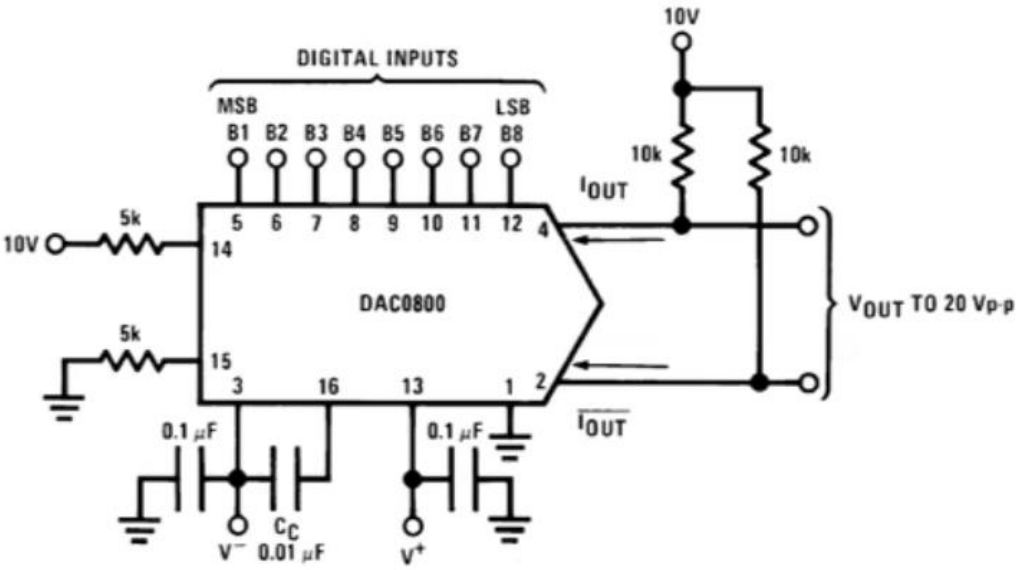
چون داریم از اینتراپت ها استفاده میکنیم => اینتراپت گلوبال هم باید فعال باشه.

```
// Global enable interrupts
#asm("sei")
```

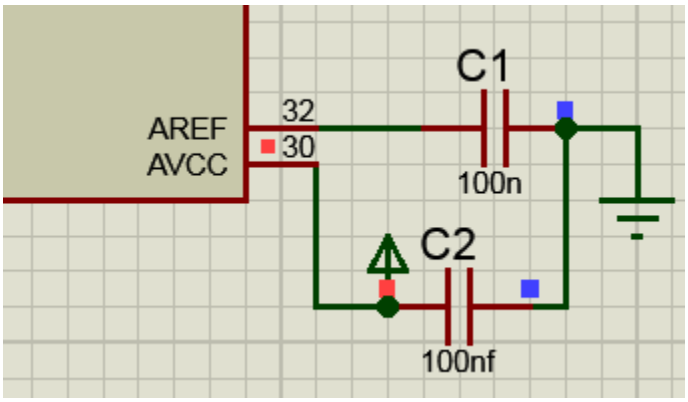
مبدل دیجیتال به انالوگ ADC

DAC0800

مبدل ۸ بیتی که به میکرو میتوانیم متصل کنیم و عددی که توسط رجیستر ADC تهیه میشه را به DAC0800 ارسال کنیم و خروجی اون را روی خازن ببینیم.



شکل 7-14 DAC0800



اگه این دوتا پورت را (power بخش adc و انالوگ vref) تنظیم نکنیم و یه مقداری براش تعیین نکنیم، این مدار کار نمیکنه چرا؟

در میکرو کنترلرها، تغذیه ی بخش انالوگ و تغذیه ی بخش دیجیتال از هم جداست برای اینکه روی هم تاثیر نگذارند و باعث نشوند که نویز این دوتا روی هم اثر بگذاره => در اکثر میکرو کنترلرها برای بخش انالوگ یک تغذیه ی جداگانه ای را باید تهیه کنیم.