6 جلسه ششم

آشنایی با مبدل‌های آنالوگ به دیجیتال و دیجیتال به آنالوگ

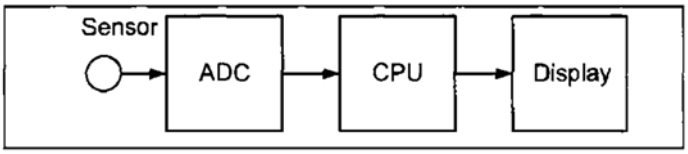
6.1 هدف

در این جلسه، نحوه‌ی تبدیل سیگنال آنالوگ به دیجیتال، استفاده از مبدل ADC داخلی ریزپردازنده، خواندن اطلاعات از حسگرهای آنالوگ و کار با مبدل دیجیتال به آنالوگ بررسی می‌شود.

6.2 مقدمه

در بسیاری از سیستم‌ها، ریزپردازنده لازم است با محیط پیرامون ارتباط برقرار کرده به این صورت که داده‌هایی را از این محیط دریافت نموده و پس از پردازش لازم خروجی را به محیط برگرداند. نکته نخست در این رابطه این است که محیط پیرامون صرفاً شامل سامانه‌های الکتریکی نیست و ساختارهای مکانیکی، نوری و ... نیز در آن وجود دارند. از سوی دیگر در این ارتباط سیگنال‌ها و کمیت‌های محیط پیرامون از جنس آنالوگ بوده، در حالی که در ساختار ریزپردازنده ما با کمیت‌های دیجیتال سر و کار داریم. لذا نیاز به واحدهای سخت‌افزاری برای اندازه گیری کمیت‌های فیزیکی و تبدیل آن‌ها به سیگنال‌های الکتریکی دیجیتال وجود دارد.

به عنوان مثال برای پایش دما یا شدت نور و یا هر کمیت دیگر و نمایش آن بر روی LCD کاراکتری، باید مطابق شکل 6-1 ابتدا کمیت فیزیکی مورد نظر با استفاده از یک حسگر مناسب به یک سیگنال الکتریکی (ولتاژ یا جریان) آنالوگ تبدیل شود. سپس این سیگنال توسط مبدل آنالوگ به دیجیتال به داده‌های دیجیتال تبدیل شده که پردازنده می‌تواند آن‌ها را مورد پردازش قرار داده و مقادیرشان را روی نمایشگر نمایش دهد.



شکل 6-1:اتصال یک حسگر به ریزپردازنده با استفاده از مبدل ADC

معکوس روال فوق برای انتقال یک داده دیجیتال از درون ریزپردازنده به محیط پیرامون با استفاده از مبدل دیجیتال به آنالوگ صورت می پذیرد. در ادامه اجزای مختلف مورد نیاز در ارتباط ریزپردازنده با محیط پیرامون معرفی خواهند شد.

6.3 حسگرها

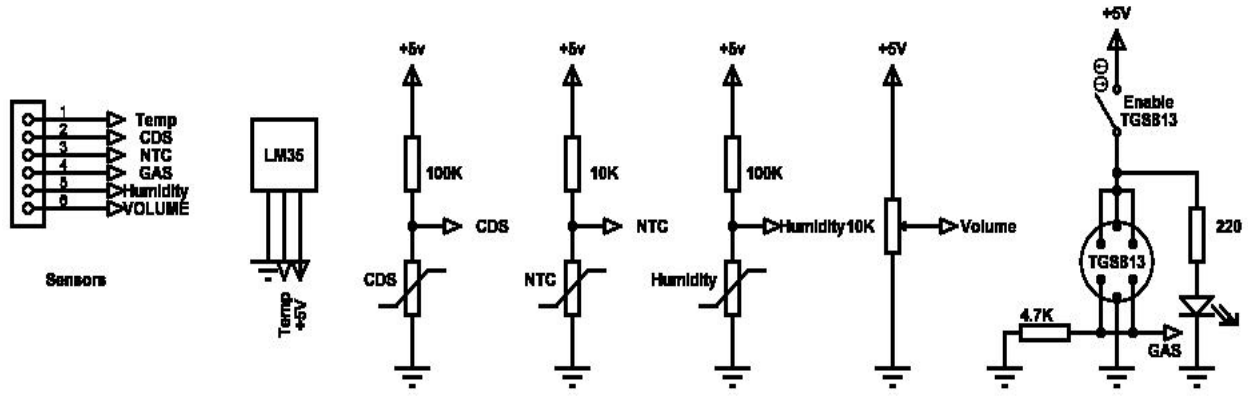
همان گونه که در بخش قبل بیان شد، برای تبدیل کمیت‌های فیزیکی مختلف به کمیت‌های الکتریکی از حسگرهای مختلف استفاده می‌شود. بر روی بورد آموزشی حسگرهای گوناگونی به شرح زیر در بلوکی با عنوان Sensors قرار داده شده است.

- دو عدد حسگر دما (LM35 وNTC)
- یک عدد حسگر نور (CDS)
- یک عدد حسگر رطوبت (HIS06)
- یک عدد حسگر گاز شهری (TGS813)
- یک ولوم 10 کیلو اهم جهت شبیه‌سازی حسگر مولفه‌های محیطی دلخواه

حسگرهای CDS، NTC و HIS06 با تغییر پارامترهای محیطی (به ترتیب نور، دما و رطوبت) تغییر مقاومت می‌دهند. از این رو به منظور ارتباط با ریزپردازنده طبق مدارهای **Error! Reference source not found.**، تغییرات مقاومت در آن‌ها به تغییرات ولتاژ تبدیل شده است.

حسگر TGS813 هم یک حسگر آشکارساز گاز شهری با خروجی ولتاژ می‌باشد که مستقیماً به ریزپردازنده متصل می‌شود. به دلیل افزایش حرارت بدنه حسگر و همچنین جریان کشی بالای این حسگر از منبع تغذیه مطابق شماتیک شکل 6-2 یک کلید کشویی برای قطع و وصل نمودن این حسگر در مسیر تغذیه حسگر تعبیه شده و LED موجود در این بلوک نشانگر وصل یا قطع بودن این حسگر در مدار است.

همچنین یک عدد ولوم 10 کیلو اهم به منظور تولید ولتاژ از سطح صفر ولت تا 5 ولت برای شبیه‌سازی حسگرهای پارامترهای محیطی در این بلوک قرار داده شده است.



شکل 6-2- شماتیک مربوط به حسگرهای موجود بر روی بورد آموزشی

در مورد حسگر LM35 در بخش‌های بعدی به تفصیل صحبت خواهد شد.

6.4 مبدل ADC

این مبدل وظیفه تبدیل سیگنال آنالوگ ورودی به یک سیگنال دیجیتال را بر عهده دارد. در ادامه با ویژگی‌های اصلی و ساختار این مبدل آشنا خواهیم شد.

یکی از اصلی‌ترین ویژگی‌های مبدل A/D دقت یا وضوح می‌باشد که بر حسب تعداد بیت بیان می‌شود. در واقع این ویژگی نشان‌دهنده تعداد سطوح قابل تفکیک در بازه تغییرات هر نمونه از سیگنال آنالوگ ورودی است. به عنوان مثال دقت مبدل‌های ADC در ریزپردازنده Atmega16/32، 8 یا 10 بیت می‌باشد که نشان می‌دهد هر نمونه سیگنال ورودی با 256 یا 1024 سطح مختلف قابل تبدیل خواهد بود.

بر این اساس فاصله بین مقدار معادل دو مقدار دیجیتال متوالی را step size می‌نامند. هرچه تعداد بیت‌های مبدل دیجیتال بیشتر باشد، step size کمتر است.

$$step\ size = \frac{V_{ref}}{2^n}$$

V_{ref} یک ولتاژ مرجع است که A/D را قادر می‌سازد تا سیگنال‌های آنالوگ در محدوده‌ی بین 0 تا Vref را مانند شکل 3-6 اندازه بگیرد. V_{ref} می‌تواند یکی از مقادیر AVCC (5 ولت)، ولتاژ مرجع داخلی (2.56 ولت) و یا ولتاژ تعیین شده توسط پایه‌ی خارجی AREF را اتخاذ نماید.

لازم به ذکر است که برای کاهش اثرات نویز در ریزپردازنده از خط تغذیه (AVCC) و زمین (AGND) جداگانه‌ای برای بخش مرتبط با سیگنال‌های آنالوگی استفاده می‌شود که AVCC نباید بیش از ±0.3V نسبت به VCC اختلاف داشته باشد.

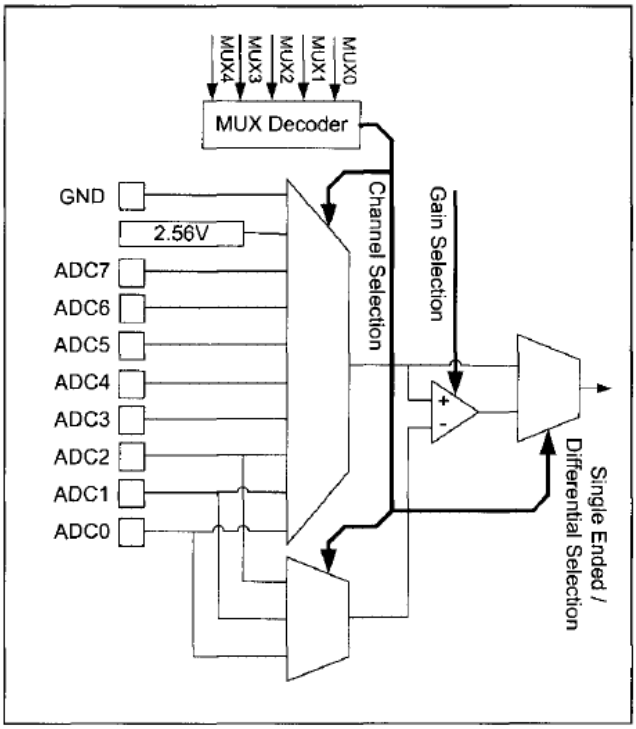
V_{ref} (V)	V_{in} Range (V)	Step Size (mV)
5.00	0 to 5	5/256 = 19.53
4.0	0 to 4	4/256 = 15.62
3.0	0 to 3	3/256 = 11.71
2.56	0 to 2.56	2.56/256 = 10
2.0	0 to 2	2/256 = 7.81
1.28	0 to 1.28	1.28/256 = 5
1	0 to 1	1/256 = 3.90

شکل 3-6 ارتباط میان ولتاژ مرجع، ولتاژ ورودی و step

دقت شود که حداکثر ولتاژ قابل اندازه‌گیری برابر با Vref (=VCC) است و در صورت اعمال ولتاژ بیشتر، مبدل آنالوگ به دیجیتال آسیب می‌بیند. کمترین ولتاژ اعمالی نیز برابر با GND است. بنابراین ADC به ازای ولتاژ 5 ولت در

حالت 10 بیتی عدد 1023 (و در حالت 8 بیتی عدد 255) و به ازای صفر ولت عدد صفر را به عنوان خروجی محاسبه و در ثبات مربوطه قرار می‌دهد.

در Atmega16/32 مطابق شکل 4-6 می‌توان 8 سیگنال آنالوگ ورودی را به دیجیتال تبدیل کرد. در هر لحظه یک یا دو کانال ورودی از طریق مالتی پلکسر انتخاب می‌شوند و پس از اعمال ضریب بهره مناسب، نمونه برداری انجام شده و خروجی به صورت داده‌های دیجیتال ۸ بیتی و یا ۱۰ بیتی در ثبات مربوطه قرار می‌گیرد.



شکل 4-6 : کانال‌های ورودی ADC (تعبیه شده بر روی درگاه A)

6.5 ثبات‌های مبدل آنالوگ به دیجیتال

برای کار با ADC یک سری ثبات در دسترس است که در ادامه به معرفی آن‌ها می‌پردازیم.

6.5.1 ثبات کنترلی ADMUX

تنظیمات اولیه مبدل آنالوگ به دیجیتال در ثبات کنترلی **ADMUX** انجام می‌شود. این تنظیمات شامل انتخاب ولتاژ مرجع، ۸ یا ۱۰ بیتی بودن مبدل، انتخاب کانال ورودی و تنظیم حالت‌های عملکردی می‌باشد.

7	6	5	4	3	2	1	0	
REFS1	REFS0	ADLAR	MUX4	MUX3	MUX2	MUX1	MUX0	ADMUX
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

شکل 5-6: ساختار ثبات کنترلی ADMUX

REFS0,1: از این دو بیت برای انتخاب ولتاژ مرجع ADC استفاده می‌کنیم که دارای چهار حالت زیر می‌باشد:

جدول 6-1: تعیین ولتاژهای مرجع مبدل آنالوگ به دیجیتال

REFS1	REFS0	Vref
0	0	AREF
0	1	AVCC
1	0	-
1	1	2.56

تامین ولتاژ مرجع دقیق در فرایند تبدیل آنالوگ به دیجیتال نقش بسیار مهمی‌دارد. دقیق‌ترین ولتاژ مرجع همان ولتاژ 2.56 داخلی می‌باشد. البته می‌توان از تثبیت کننده‌های ولتاژ خارجی برای تولید ولتاژ مرجع دلخواه نیز استفاده نمود که در این حالت باید آن را به پایه‌ی AREF متصل نمود.

ADLAR: از این بیت برای تعیین نحوه پر شدن ثبات ADC به صورت از چپ به راست و یا بالعکس استفاده می‌شود. اگر مقدار این بیت صفر باشد ثبات ADC از سمت چپ پرشده و در غیر این صورت از راست پر می‌شود.

MUX0-4: مقدار این بیت‌ها، مانند شکل 6-6 ترکیب ورودی‌های آنالوگ را برای مبدل تعیین می‌نماید. همچنین در ترکیب‌های تفاضلی مقدار ضریب بهره را نیز مشخص می‌کند. به صورت کلی خروجی از رابطه زیر محاسبه خواهد شد:

$$D_{out} = A * \left(\frac{V_i^+ - V_i^-}{V_{ref}} \right) * 2^n$$

دقت شود که ترکیب تفاضلی فقط در پکیج‌های TQFP و MLF وجود دارد و برای استفاده از آن باید پایه Positive Differential Input را به ولتاژ ورودی آنالوگ و پایه Negative Differential Input را به زمین وصل کنید.

شکل 6-6 : تنظیمات قابل اعمال برای ADC با تغییر پارامترهای MUX4-0 در ثبات ADMUX

MUX4..0	Single Ended Input	Positive Differential Input	Negative Differential Input	Gain
00000	ADC0	N/A		
00001	ADC1			
00010	ADC2			
00011	ADC3			
00100	ADC4			
00101	ADC5			
00110	ADC6			
00111	ADC7			
01000	N/A	ADC0	ADC0	10x
01001		ADC1	ADC0	10x
01010		ADC0	ADC0	200x
01011		ADC1	ADC0	200x
01100		ADC2	ADC2	10x
01101		ADC3	ADC2	10x
01110		ADC2	ADC2	200x
01111		ADC3	ADC2	200x
10000		ADC0	ADC1	1x
10001		ADC1	ADC1	1x
10010		ADC2	ADC1	1x
10011		ADC3	ADC1	1x
10100		ADC4	ADC1	1x
10101		ADC5	ADC1	1x
10110		ADC6	ADC1	1x
10111		ADC7	ADC1	1x
11000		ADC0	ADC2	1x
11001		ADC1	ADC2	1x
11010		ADC2	ADC2	1x
11011		ADC3	ADC2	1x
11100		ADC4	ADC2	1x
11101		ADC5	ADC2	1x
11110	1.22V (V_{BG})	N/A		
11111	0V (GND)			

6.5.2 ثبات ADCSRA

این ثبات نحوه فعال شدن مبدل را تنظیم می‌نماید و هم‌چنین وضعیت پایان یافتن عملیات تبدیل را درون خود نگه می‌دارد.

7	6	5	4	3	2	1	0	
ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0	ADCSRA
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

شکل 6-7: ساختار ثبات ADCSRA

ADEN: با یک کردن این بیت مبدل ADC فعال می‌شود.

ADSC: با نوشتن یک در این بیت، تبدیل شروع می‌شود.

ADATE: با یک کردن این بیت، مبدل ADC می‌تواند به صورت خودکار با لبه بالا رونده منبع تحریک کننده که توسط بیت‌های ADTS از ثبات SFIOR انتخاب می‌شود شروع به کار کند.

ADIF: این بیت بعد از اتمام تبدیل و به روز شدن مقدار درون ثبات داده ADC، برابر با یک می‌شود. در حقیقت یک شدن این بیت نشانه‌ی معتبر بودن داده‌های ثباتADC برای خوانده شدن است.

ADIE: با یک کردن این بیت، پس از اتمام تبدیل وقفه‌ای صادر می‌شود و در زیر روال وقفه، داده ثبات ADC خوانده می‌شود.

ADPS0-3: از این بیت‌ها مطابق جدول 6-2 برای تعیین ضریب تقسیم پالس ساعت ریزپردازنده برای تولید پالس ساعت بخش مبدل آنالوگ به دیجیتال استفاده می‌شود.

جدول 6-2: جدول تنظیمات تقسیم پالس ساعت

ADPS2	ADPS1	ADPS0	Division Factor
0	0	0	2
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

6.5.3 ثبات داده ADC (ADCH, ADCL)

این ثبات شانزده بیتی، حاوی داده‌ی خروجی مبدل است و بنا به مقدار ADLAR، از چپ یا راست پر می‌گردد. در حالت عملکرد 8 بیتی، داده‌ی دیجیتال در بخش با ارزش رجیستر ADC ذخیره می‌شود. و در حالت ده بیتی داده‌ی دیجیتال در رجیستر ADC ذخیره می‌شود.

6.5.4 ثبات SFIOR[1]

با استفاده از این ثبات که ساختار آن در شکل 6-8 نشان داده شده است، منبع تحریک مبدل و نیز حالت عملکرد سرعت بالا تنظیم می‌شود.

7	6	5	4	3	2	1	0	
ADTS2	ADTS1	ADTS0	ADHSM	ACME	PUD	PSR2	PSR10	SFIOR
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

شکل 6-8: ساختار ثبات SFIOR

[1]Special Function IO Register

ADTS0-2: از طریق بیت‌های ADTS0-2 می‌توان منبع تحریک مبدل برای شروع تبدیل را مانند جدول 6-3 تنظیم نمود.

جدول 6-3: تعیین منابع تحریک ADC

منبع تحریک ADC	ADTS0	ADTS1	ADTS2
مدعملکردآزاد	0	0	0
مقایسه کننده آنالوگ	1	0	0
وقفه خارجی صفر	0	1	0
وقفه‌ی (Compare Match) تایمر صفر	1	1	0
سرریز تایمر صفر	0	0	1
وقفه‌ی Compare Match B	1	0	1
سرریز تایمر یک	0	1	1
ضبط رخداد تایمر یک	1	1	1

بیت ADHSM: با فعال شدن این بیت، فرایند نمونه‌برداری در مبدل با سرعت بیشتر و البته با مصرف انرژی بیشتر انجام می‌شود.

6.5.5 معرفی وقفه ADC

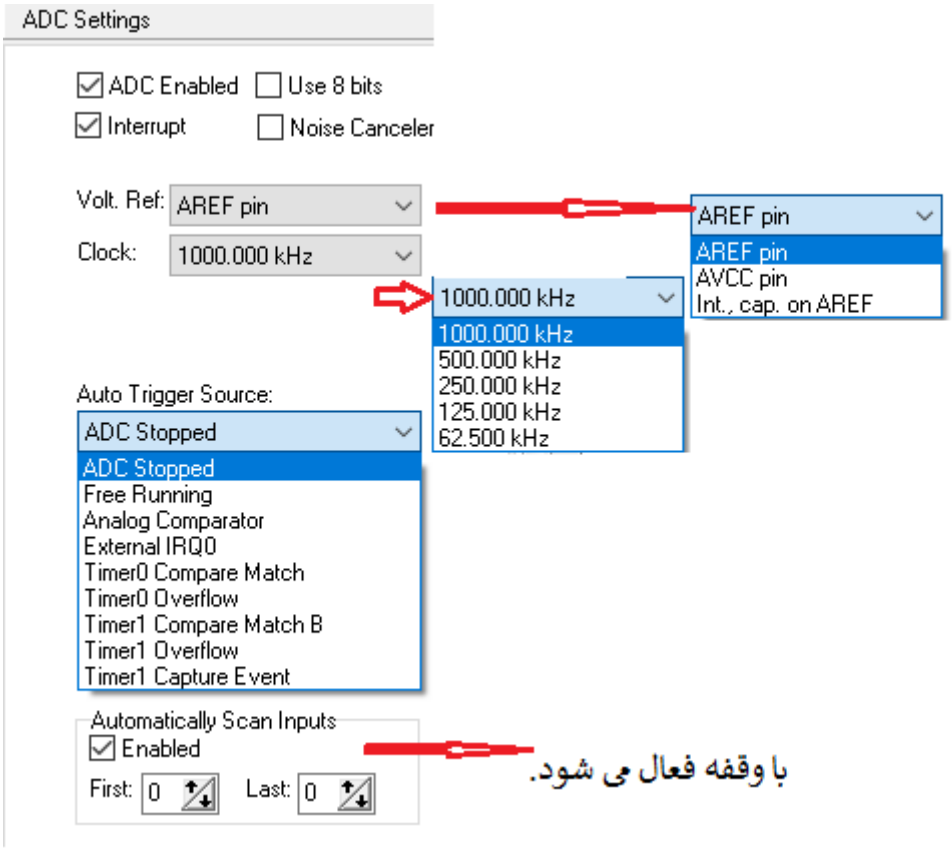
برای جلوگیری از اتلاف زمان ریزپردازنده، به جای بررسی مکرر بیت ADSC در ثبات ADCSRA بهتر است از وقفه استفاده شود (برای این منظور لازم است بیت ADIE از ثبات ADCSRA برابر با یک باشد). در این صورت به محض کامل شدن تبدیل، پرچم ADIF یک می‌شود و CPU به محل اجرای وقفه پرش کرده که در آن‌جا نتیجه ADC قابل استفاده خواهد بود.

6.6 مراحل برنامه‌نویسی ADC

مراحل برنامه‌نویسی مبدل آنالوگ به دیجیتال در شکل 6-9 نشان داده شده است. بدین منظور لازم است پایه ورودی سیگنال آنالوگ تعریف شود و از منوی CodeWizard سایر پارامترها نیز تنظیم شود که به ترتیب شامل موارد ذیل می‌باشد:

1. فعال کردن ماژول ADC

2. انتخاب 8 یا 10 بیتی بودن تبدیل: در حالت کلی مبدل Atmega16/32 به صورت ده بیتی نمونه‌برداری را انجام می‌دهد، ولی با انتخاب هشت بیتی تنها 8 بیت با ارزش‌تر در نظر گرفته می‌شوند.
3. حذف نویز: بخش حذف نویز با انتخاب وقفه قابل فعال کردن است و این امکان را فراهم می‌کند تا ریزپردازنده در حالت SLEEP هم پروسه‌ی تبدیل به دیجیتال را انجام دهد و بدین وسیله تاثیر نویزهای هسته ریز پردازنده و پایه‌های ورودی و خروجی را کاهش می‌دهد. در این حالت نمونه‌برداری فقط در حالتی که ریزپردازنده حالت بیکار داشته باشد انجام می شود. لذا قبل از انجام این پروسه بایستی پیام Sleep به ریزپردازنده ارسال گردد.
4. تعیین کردن سرعت تبدیل: انتخاب سرعت تبدیل به فرکانس سیگنال آنالوگ ورودی بستگی دارد. اگر فرکانس سیگنال آنالوگ بالا باشد بهتر است از نرخ‌های بالاتر برای تبدیل استفاده نمود و اگر تغییرات ورودی به کندی انجام می‌شود می‌توان از نرخ‌های پایین‌تر تبدیل استفاده نمود تا هم مصرف انرژی ریزپردازنده کمتر شود و هم نویز کمتری از طریق ADC به کل مدار وارد شود. کاهش نویز منتقل شده از بخش ADC به سایر بخش‌های مدار مانند مداراهای حساس و مدارهایی که سیگنال‌های با فرکانس بالایی روی برد منتقل می‌کنند اهمیت خود را بیشتر نشان می‌دهد.
5. انتخاب ولتاژ مرجع: همان گونه که پیش‌تر بیان شد، در مبدل های ADC برای بهبود دقت اندازه‌گیری از ولتاژ مرجع دقیق استفاده می‌شود که می‌توان در صورت نیاز توسط منبع داخلی و یا با استفاده از تراشه‌های خارجی مناسب این ولتاژ را تامین نمود.
6. انتخاب منبع تحریک مبدل: در حالت عادی می‌توان از حالت free running استفاده کرد. همچنین امکان فعال نمودن وقفه در بازه زمانی مشخص هم مانند استفاده از وقفه‌های تایمر وجود دارد.
7. اگر وقفه فعال شده باشد می‌توان به طور خودکار تعدادی از کانال‌های ADC را پردازش نمود.



شکل 6-9 :برنامه نویسی مبدل آنالوگ به دیجیتال با CodeWizard

6.6.1 برنامه خواندن داده‌های مبدل دیجیتال

برای خواندن داده‌های مبدل ADC در حالت بدون وقفه از زیربرنامه‌ی read_adc مانند برنامه 6-1 می‌توان استفاده کرد. هم‌چنین اگر وقفه فعال شده باشد، با استفاده از روتین وقفه‌ی برنامه 6-2، مقدار هر یک از ورودی‌ها خوانده شده و در یک متغیر ذخیره می‌گردد. در برنامه 6-1 و برنامه 6-2 مبدل به صورت ده بیتی فعال بوده و بنابراین کل ثبات ADCW خوانده می‌شود. در صورتی که در حالت هشت بیتی کافی است فقط ADCH خوانده شود.

برنامه 6-1

```
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;
}
```

برنامه 2-6

```
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);
}
```

6.7 اندازه‌گیری دما

برای اندازه‌گیری دما می‌توان از حسگر LM35 که در شکل 6-10 نشان داده شده استفاده نمود. این حسگر دمای بین 55- تا 150 درجه‌ی سانتیگراد را اندازه می‌گیرد و به ازای هر 1 ± سانتی‌گراد، 10 mV± ولتاژ خروجی را تغییر می‌دهد. یعنی به ازای دمای 1 درجه، ولتاژ خروجی حسگر 10mV و به ازای دمای 100 درجه، خروجی حسگر 1V خواهد بود.

شکل 6-10: حسگر اندازه گیری دما LM35

مطابق شکل فوق، این حسگر دارای 3 پایه می‌باشد. در صورتی که حسگر را روبروی خود بگیرید به گونه‌ای که بتوانید نوشته‌هایش را ببینید، پایه سمت چپ تغذیه حسگر (5 ولت)، پایه وسط ولتاژ خروجی (که به ریزپردازنده متصل می‌شود) و پایه سمت راست پایه زمین خواهد بود. خروجی آنالوگ این حسگر را می‌توان توسط مبدل‌های آنالوگ به دیجیتال موجود در تراشه Atmega16/32 به دیجیتال تبدیل کرده و به عنوان مثال ان را روی LCD نمایش داد.

مثال: با استفاده از حسگر LM35، دما را در بازه‌ی 0 تا 50 درجه سانتیگراد اندازه‌گیری و بر روی LCD نمایش دهید.

برای اندازه گیری دما در بازه‌ی 0 تا 50 درجه روابط ذیل برقرار است:

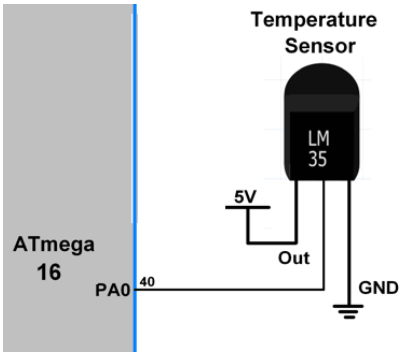
$$D_{OUT} = \frac{V_{LM35}}{V_{REF}} \times 2^{10}$$

$$V_{LM35} = 10mv \times T$$

اگر ولتاژ مرجع برابر با 5 ولت در نظر گرفته شود و مبدل آنالوگ به دیجیتال در حالت 10 بیتی کار کند، دما برابر خواهد بود با:

$$T = \frac{D_{OUT} \times 500}{2^{10}} \cong \frac{D_{OUT}}{2}$$

در ادامه مانند شکل 6-11 خروجی LM35 را به پایه ADC0 متصل می‌کنیم. LCD نیز به درگاه B وصل است.



شکل 6-11 نحوه اتصال LM35 به ریزپردازنده

سپس در قسمت CodeWizard تنظیمات مربوط به LCD و ADC انجام و قطعه کد زیر به برنامه اضافه می‌گردد.

برنامه 6-3

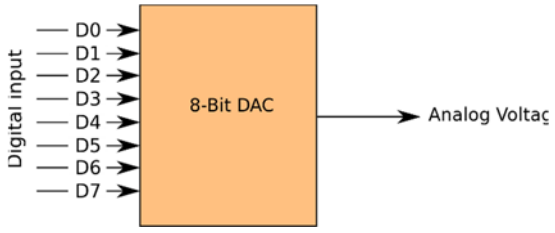
```
#include <mega16.h>

#include <delay.h>
#include <stdio.h>

// Alphanumeric LCD functions
#include <alcd.h>

// Declare your global variables here

// 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;
}

void main(void)
{
// Declare your local variables here
char str[20];
// Input/Output Ports initialization
// Port A initialization
// Function: Bit7=In Bit6=In Bit5=In Bit4=In Bit3=In Bit2=In Bit1=In
Bit0=In
DDRA=(0<<DDA7) | (0<<DDA6) | (0<<DDA5) | (0<<DDA4) | (0<<DDA3) |
(0<<DDA2) | (0<<DDA1) | (0<<DDA0);
// State: Bit7=T Bit6=T Bit5=T Bit4=T Bit3=T Bit2=T Bit1=T Bit0=T
PORTA=(0<<PORTA7) | (0<<PORTA6) | (0<<PORTA5) | (0<<PORTA4) |
(0<<PORTA3) | (0<<PORTA2) | (0<<PORTA1) | (0<<PORTA0);

// Port B initialization
// Function: Bit7=In Bit6=In Bit5=In Bit4=In Bit3=In Bit2=In Bit1=In
Bit0=In
DDRB=(0<<DDB7) | (0<<DDB6) | (0<<DDB5) | (0<<DDB4) | (0<<DDB3) |
(0<<DDB2) | (0<<DDB1) | (0<<DDB0);
// State: Bit7=T Bit6=T Bit5=T Bit4=T Bit3=T Bit2=T Bit1=T Bit0=T
PORTB=(0<<PORTB7) | (0<<PORTB6) | (0<<PORTB5) | (0<<PORTB4) |
(0<<PORTB3) | (0<<PORTB2) | (0<<PORTB1) | (0<<PORTB0);

// Port C initialization
// Function: Bit7=In Bit6=In Bit5=In Bit4=In Bit3=In Bit2=In Bit1=In
Bit0=In
DDRC=(0<<DDC7) | (0<<DDC6) | (0<<DDC5) | (0<<DDC4) | (0<<DDC3) |
(0<<DDC2) | (0<<DDC1) | (0<<DDC0);
// State: Bit7=T Bit6=T Bit5=T Bit4=T Bit3=T Bit2=T Bit1=T Bit0=T
PORTC=(0<<PORTC7) | (0<<PORTC6) | (0<<PORTC5) | (0<<PORTC4) |
(0<<PORTC3) | (0<<PORTC2) | (0<<PORTC1) | (0<<PORTC0);

// Port D initialization
// Function: Bit7=In Bit6=In Bit5=In Bit4=In Bit3=In Bit2=In Bit1=In
Bit0=In
DDRD=(0<<DDD7) | (0<<DDD6) | (0<<DDD5) | (0<<DDD4) | (0<<DDD3) |
(0<<DDD2) | (0<<DDD1) | (0<<DDD0);
// State: Bit7=T Bit6=T Bit5=T Bit4=T Bit3=T Bit2=T Bit1=T Bit0=T
```

```
PORTD=(0<<PORTD7) | (0<<PORTD6) | (0<<PORTD5) | (0<<PORTD4) |
(0<<PORTD3) | (0<<PORTD2) | (0<<PORTD1) | (0<<PORTD0);

// 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);

// Alphanumeric LCD initialization
// Connections are specified in the
// Project|Configure|C Compiler|Libraries|Alphanumeric LCD menu:
// RS - PORTB Bit 0
// RD - PORTB Bit 1
// EN - PORTB Bit 2
// D4 - PORTB Bit 4
// D5 - PORTB Bit 5
// D6 - PORTB Bit 6
// D7 - PORTB Bit 7
// Characters/line: 16
lcd_init(16);
while (1)
      {
 //       Place your code here
          read_adc(0);
          sprintf(str,"%d.%d",(ADCH*500)/1023,((
ADCH*500)/1023)%10);
          lcd_puts(str);
          delay_ms(1000);

      }
}
```

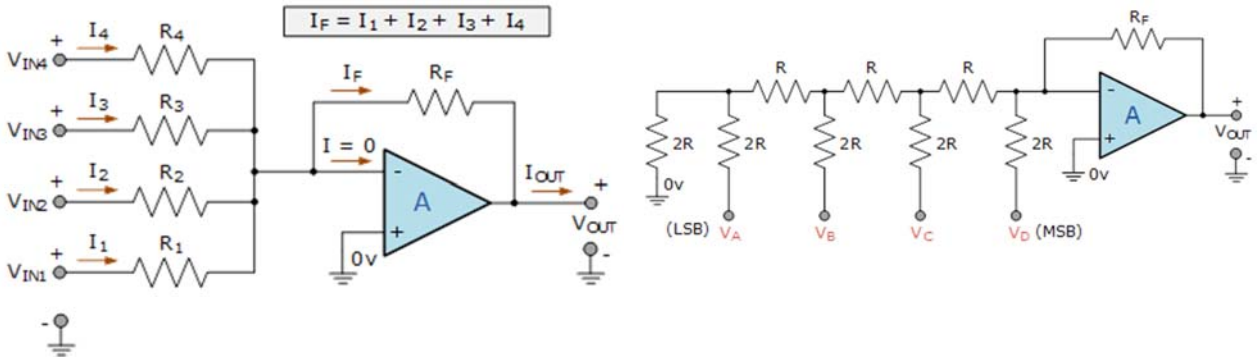
6.8 آشنایی با تبدیل سیگنال دیجیتال به آنالوگ

در تبدیل‌کننده‌های دیجیتال به آنالوگ مانند شکل 6-12، به ازای داده‌های دیجیتال ورودی، خروجی معادل آنالوگ به صورت ولتاژ یا جریان تولید می‌شود.



شکل 6-12: بلوک دیاگرام مبدل دیجیتال به آنالوگ

دقت یا وضوح خروجی بر حسب تعداد بیت‌های ورودی تعیین می‌شود به این صورت که اگر تعداد ورودی‌های DAC به تعداد n باشد، تعداد سطوح خروجی 2^n خواهد بود. پروسه‌ی تبدیل سیگنال دیجیتال به آنالوگ مانند شکل 13-6 به دو روش نردبانی[1] و باینری وزن‌دار قابل انجام است.



ب الف

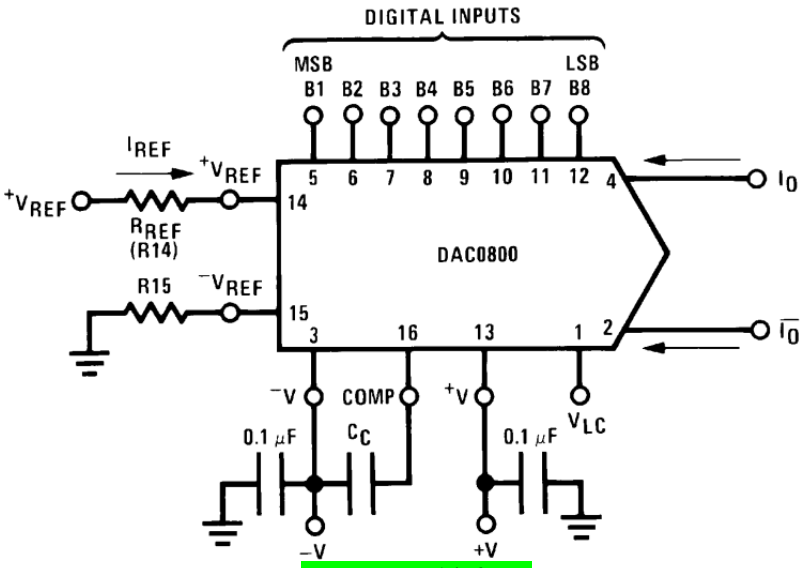
شکل 13-6 : پروسه‌ی تبدیل سیگنال دیجیتال به آنالوگ، الف: نردبانی، ب: باینری وزن‌دار

در بورد آموزشی آزمایشگاه از ماژول DAC0800 به عنوان یک تبدیل‌کننده دیجیتال به آنالوگ هشت بیتی به روش باینری وزن‌دار استفاده شده است که دارای 256 سطح در خروجی می‌باشد. در شکل 14-6 نمایی از آن نشان داده شده است. ورودی‌های دیجیتال از LSB تا MSB به ترتیب به پایه‌های B8 تا B1 متصل می‌گردد. ولتاژ مرجع برای تبدیل به آنالوگ از تغذیه تراشه و یا ولتاژی که به پایه‌های (+)Vref و (-)Vref به ترتیب در پایه‌های 14 و 15 اعمال می‌شود، تامین می‌گردد. همچنین برای تقویت خروجی میتوان مانند

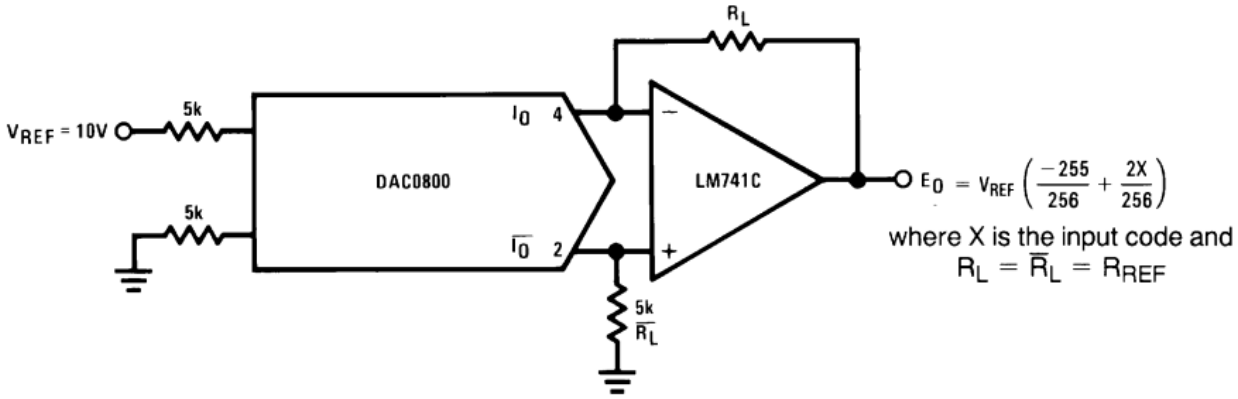
شکل 15-6: تقویت کننده در خروجی مبدل دیجیتال به آنالوگ

از یک تقویت کننده عملیاتی استفاده نمود.

[1] ladder



شکل 6-14 مبدل DAC0800



شکل 6-15: تقویت کننده در خروجی مبدل دیجیتال به آنالوگ

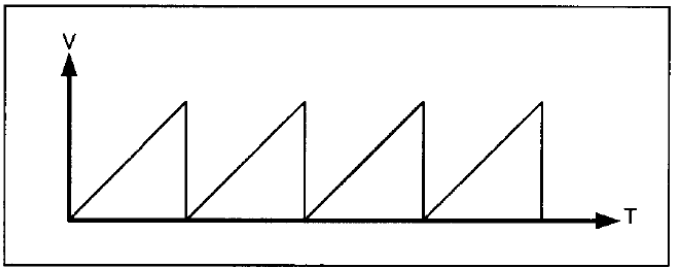
6.9 نمونه برنامه مبدل آنالوگ به دیجیتال

به عنوان نمونه با استفاده از برنامه 6-4 و با به کارگیری مبدل دیجیتال به آنالوگ می‌توان یک موج دندانه اره‌ای مانند شکل 6-16 طراحی نمود.

برنامه 6-4

```
#include <mega16.h>
unsigned char i;        //define a counter
int main (void)
{
DDRB = OxFF;
while (1)
{
PORTB = i;
```

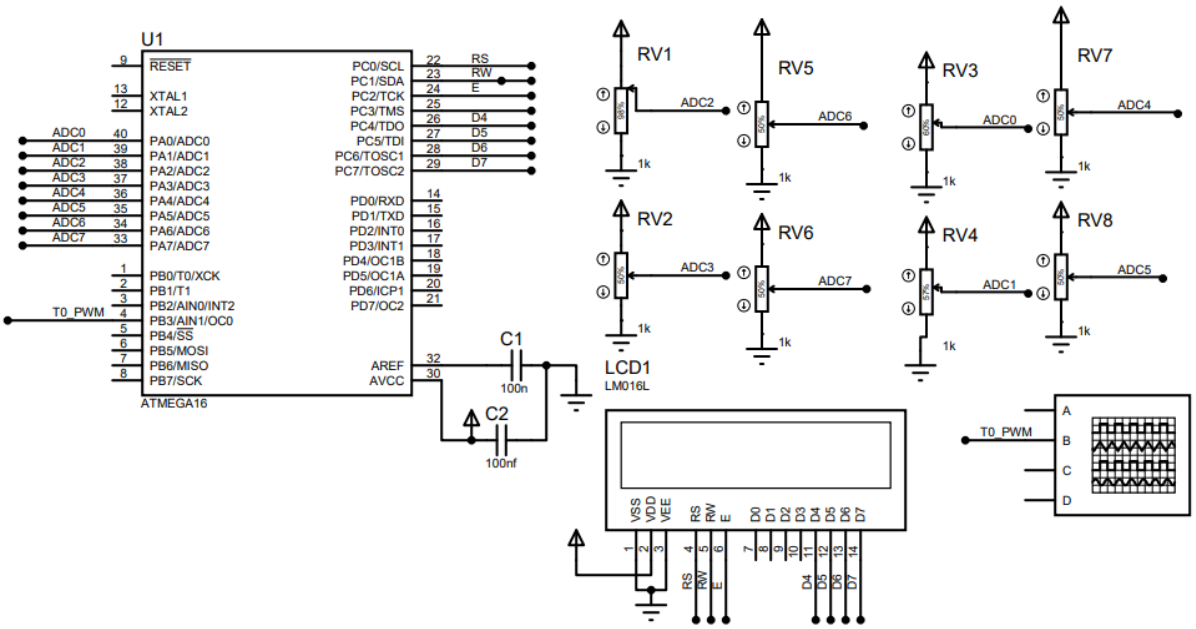
```
i++;
}
return 0;
}
```



شکل 6-16: Step Ramp Output

6.10 برنامه‌های اجرایی مبدل‌های آنالوگ به دیجیتال

سخت‌افزار زیر را در نظر بگیرید:



1- زیربرنامه‌ای بنویسید که ولتاژ هر یک از ورودی‌های آنالوگ را بخواند و بر حسب میلی‌ولت روی LCD نشان دهد. (راهنمایی: تنظیمات CodeWizard را برای ADC بدون وقفه فعال نمایید. در کدهای ایجاد شده برای خواندن مقدار دیجیتال هر یک از کانال‌ها، از تابع read_adc استفاده می‌شود. پس از اجرای صحیح این بند، پیکربندی بخش ADC را در قالب یک زیر برنامه و تابع read_adc را نیز به فایل جانبی منتقل نمایید این فایل در بخش نهایی استفاده خواهد شد.)

2- در یک پروژه جدید، زیر برنامه ای بنویسید که از طریق وقفه مقدار هر یک از ورودی‌های آنالوگ را بخواند و در صورتی که تغییرات بیش از 5 درصد باشد، آن را روی LCD نشان دهد. (آخرین مقدار نشان داده شده روی LCD را با مقدار جدید هر ورودی آنالوگ مقایسه نمایید و چنانچه بیش از 5٪ تغییر داشت، جهت نمایش بر روی LCD اقدام نمایید. بدین وسیله بار پردازشی ریزپردازنده کمتر شده و نوشته های روی LCD پرش نخواهد داشت. پس از اجرای صحیح این بند، پیکربندی بخش ADC را در قالب یک زیر برنامه و روال سرویس دهی وقفه و تعاریف اولیه را نیز به فایل جانبی منتقل نمایید این فایل در بخش نهایی استفاده خواهد شد.)

3- زیربرنامه‌ای بنویسید که با استفاده از تایمر صفر، یک پالس PWM تولید نماید به صورتی که چرخه‌ی کار این پالس بر اساس سیگنال آنالوگ متصل به ADC0 تنظیم گردد. (راهنمایی: در این بخش مانند بند قبلی بایستی وقفه‌ی ADC فعال باشد.)

4- بندهای فوق را در قالب یک پروژه در بیاورید. (راهنمایی: از فایل‌های جانبی تهیه شده در بندهای 1 و 2 استفاده نمایید. می‌توان ابتدا ADC را بدون وقفه فعال کرد و پس از نمایش ورودی‌های آنالوگ، مجدداً ADC را پیکربندی نمود و وقفه‌ی ADC را فعال کرد. سپس بند 2 و 3 را به طور همزمان اجرا نموده، به گونه‌ای که در حین اجرای برنامه تغییرات هر یک از ورودی‌ها نیز لحاظ گردد.)