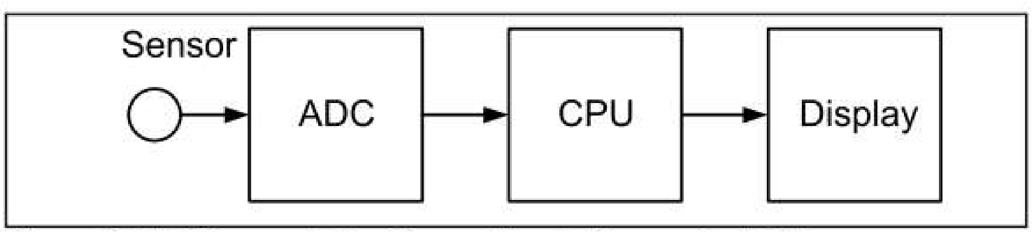
ADC, DAC, AND SENSOR INTERFACING



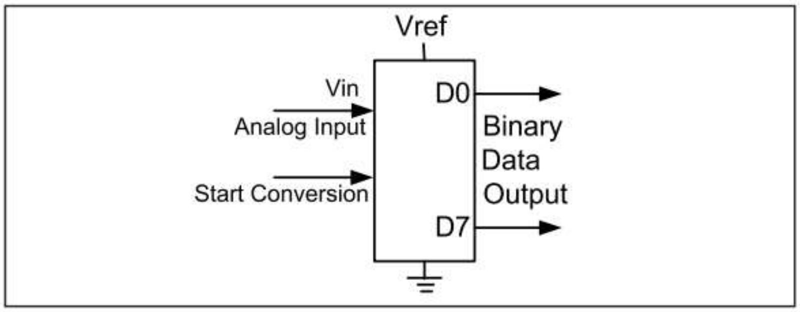


Table 13-1: Resolution versus Step Size for ADC (V_{ref} = 5 V)		
n-bit	Number of steps	Step size (mV)
8	256	5/256 = 19.53
10	1024	5/1024 = 4.88
12	4096	5/4096 = 1.2
16	65,536	5/65,536 = 0.076

Notes: V_{CC} *= 5 V*
Step size (resolution) is the smallest change that can be discerned by an ADC.

Some of the major characteristics of the ADC

- Resolution The ADC has n-bit resolution, where n can be 8, 10, 12, 16, or even 24 bits.
- Higher-resolution ADCs provide a smaller step size, where step size is the smallest change that can be discerned by an ADC. Some widely used resolutions for ADCs are shown in Table 13-1.
- Although the resolution of an ADC chip is decided at the time of its design and cannot be changed, we can control the step size with the help of what is called **Vref.**

Vref

- Vref is an input voltage used for the reference voltage.
- The voltage connected to this pin, along with the resolution of the ADC chip, dictate the step size.
- For an 8-bit ADC, the step size is Vref/256 because it is an 8-bit ADC, and 2 to the power of 8 gives us 256 steps.
- See Table 13-1. For example, if the analog input range needs to be 0 to 4 volts, Vref is connected to 4 volts. That gives 4 V/256 = 15.62 mV for the step size of an 8-bit ADC. In another case, if we need a step size

Table 13-2: V_{ref} Relation to V_{in} Range for an 8-bit ADC

V_{ref} (V)	V_{in} Range (V)	Step Size (mV)
5.00	0 to 5	
4.0	0 to 4	
3.0	0 to 3	
2.56	0 to 2.56	
2.0	0 to 2	
1.28	0 to 1.28	
1	0 to 1	

Step size is V_{ref} / 256

Table 13-3: V_{ref} Relation to V_{in} Range for an 10-bit ADC

V_{ref} (V)	V_{in} (V)	Step Size (mV)
5.00	0 to 5	
4.096	0 to 4.096	
3.0	0 to 3	
2.56	0 to 2.56	
2.048	0 to 2.048	
1.28	0 to 1.28	
1.024	0 to 1.024	

Digital data output

- In an 8-bit ADC we have an 8-bit digital data output of D0–D7, while in the 10-bit ADC the data output is D0–D9.
- To calculate the output voltage, we use the following formula:
- where:
- Dout = digital data output (in decimal),
- Vin = analog input voltage,
- and step size (resolution) is the smallest change, which is Vref/256 for an 8-bit ADC.

EXERCISE

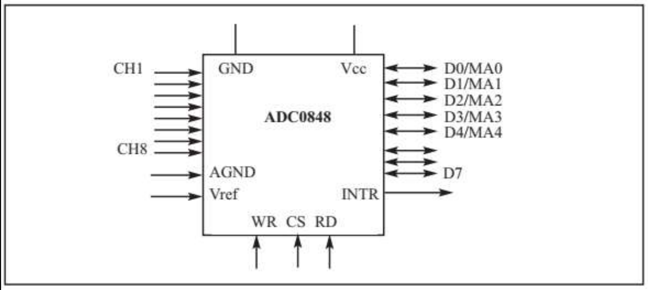
- For an 8-bit ADC, we have Vref = 2.56 V. Calculate the D0–D7 output if the analog input is: (a) 1.7 V, and (b) 2.1 V.

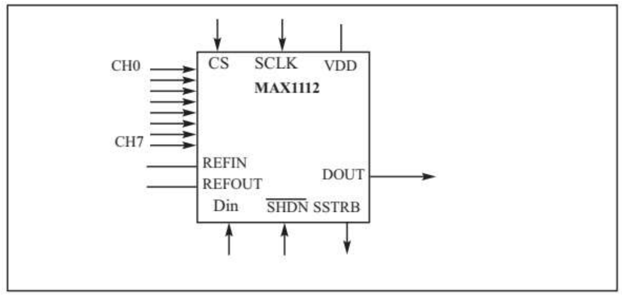
Parallel versus serial ADC

- The ADC chips are either parallel or serial. In parallel ADC, we have 8 or more pins dedicated to bringing out the binary data, but in serial ADC we have only one pin for data out.
- That means that inside the serial ADC, there is a parallel-in-serial-out shift register responsible for sending out the binary data one bit at a time. The D0–D7 data pins of the 8-bit ADC provide an 8-bit parallel data path between the ADC chip and the CPU.
- In the case of the 16-bit parallel ADC chip,

Analog input channels

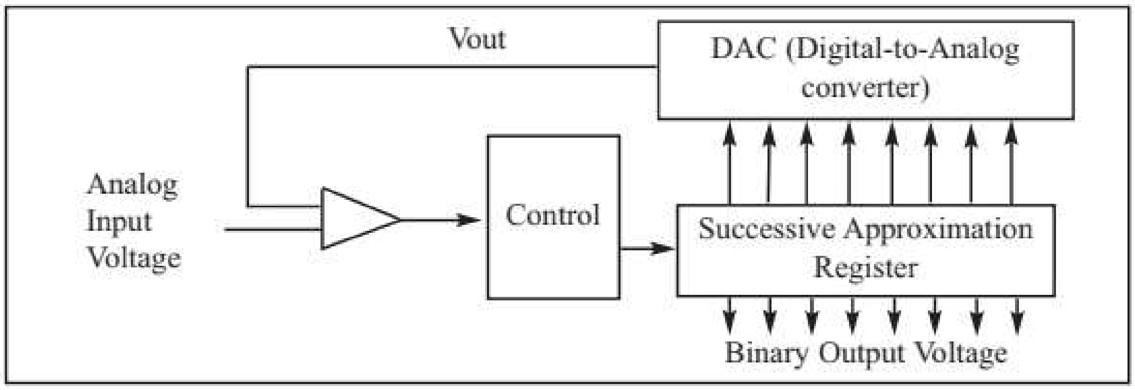
- Many data acquisition applications need more than one ADC.
- For this reason, we see ADC chips with 2, 4, 8, or even 16 channels on a single chip.
- Multiplexing of analog inputs is widely used as shown in the ADC848 and MAX1112.
- In these chips, we have 8 channels of analog inputs, allowing us to monitor multiple quantities such as temperature, pressure, heat, and so on. AVR microcontroller chips come with up to 16 ADC channels.





Start conversion and end-of-conversion signals

- The fact that we have multiple analog input channels and a single digital output register creats the need for start conversion (SC) and end-of-conversion (EOC) signals.
- When SC is activated, the ADC starts converting the analog input value of Vin to an n-bit digital number.
- The amount of time it takes to convert varies depending on the conversion method.
- When the data conversion is complete, the end-of-conversion signal notifies the CPU that the converted data is ready to be picked up. Successive Approximation ADC Successive Approximation is a widely used method of converting an analog input to digital output. It has three main components: (a) successive approximation register (SAR), (b) comparator, and (c) control unit.

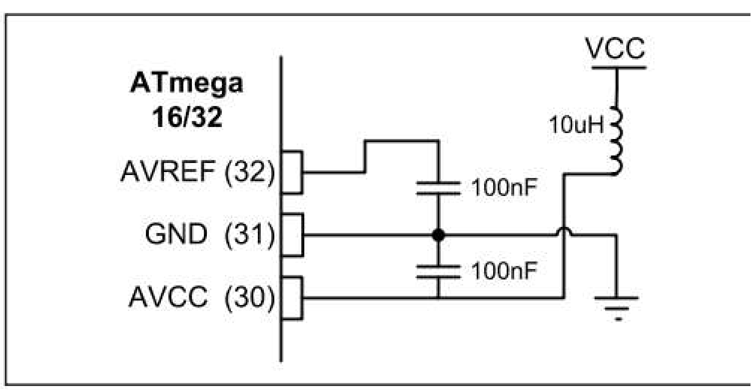


Review Questions

- 1. Give two factors that affect the step size calculation.
- 2. The ADC0848 is a(n) _______-bit converter.
- 3. True or false. While the ADC0848 has 8 pins for Dout, the MAX1112 has only one Dout pin.
- 4. Find the step size for an 8-bit ADC, if Vref = 1.28 V. 5. For question 4, calculate the output if the analog input is: (a) 0.7 V, and (b) 1 V.

ATmega328 ADC features

- The ADC peripheral of the ATmega328 has the following characteristics:
- (a) It is a 10-bit ADC.
- (b) It has 6 analog input channels (8 analog input channels in TQFP and MFL packages) and internal temperature sensor input channel.
- (c) The converted output binary data is held by two special function registers called ADCL (A/D Result Low) and ADCH (A/D Result High).
- (d) Because the ADCH:ADCL registers give us 16 bits and the ADC data out is only 10 bits wide, 6 bits of the 16 are unused. We have the option of making either the upper 6 bits or the lower 6 bits unused.
- (e) We have three options for Vref. Vref can be connected to AVCC (Analog Vcc), internal 1.1 V reference, or external AREF pin. (f) The conversion time is dictated by the crystal frequency connected to the XTAL pins (Fosc) and ADPS0:2 bits. If 10-bit resolalution is needed, 200kHz is the maximum input clock. Otherwise, higher frequencies can also be used.



the AVCC pin provides the supply for analog ADC circuitry.
To get a better accuracy of AVR ADC we must provide a stable voltage source to the AVCC pin
. Figure shows how to use an inductor and a capacitor to achieve this.
Connecting a capacitor between Vref and GND By connecting a capacitor between the AVREF pin and GND you can make the Vref voltage more stable and increase the precision of ADC.

ADMUX register

- Figure shows the bits of ADMUX registers and their usage. In this section we will focus more on the function of these bits.
- Vref source F shows the block diagram of internal circuitry of Vref selection. As you can see we have three options: (a) AREF pin, (b) AVCC pin, or (c) internal 1.1 V. Table 13-4 shows how the REFS1 and REFS0 bits of the ADMUX register can be used to select the Vref source.
- Notice that if you connect the VREF pin to an external fixed voltage you will not be able to use the other reference voltage options in the application, as they will be shorted with the external voltage. Another important point to note is the fact that connecting a 100 nF exter-

REFS1	REFS0	ADLAR	-	MUX3	MUX2	MUX1	MUX0

REFS1:0 Bit 7:6 Reference Selection Bits
These bits select the reference voltage for the ADC.

ADLAR Bit 5 ADC Left Adjust Results
This bit dictates either the left bits or the right bits of the result registers ADCH:ADCL that are used to store the result. If we write a one to ADLAR, the result will be left adjusted; otherwise, the result is right adjusted.

MUX3:0 Bit 3:0 Analog Channel Selection Bits
The value of these bits selects the analog input connected to the ADC.

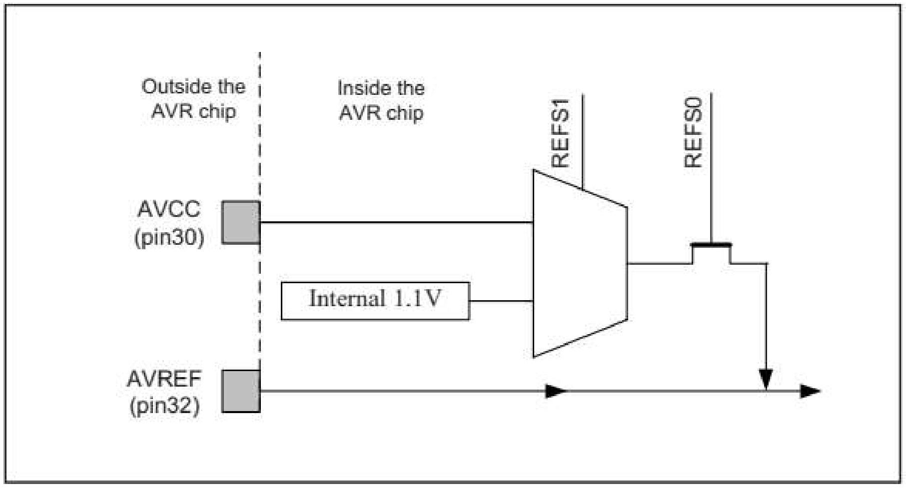
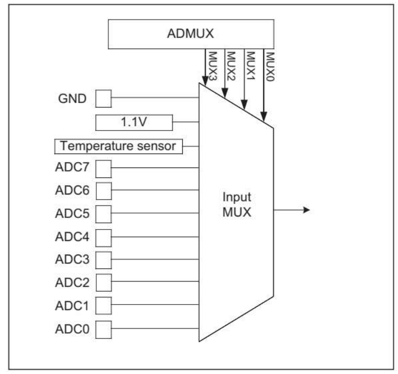


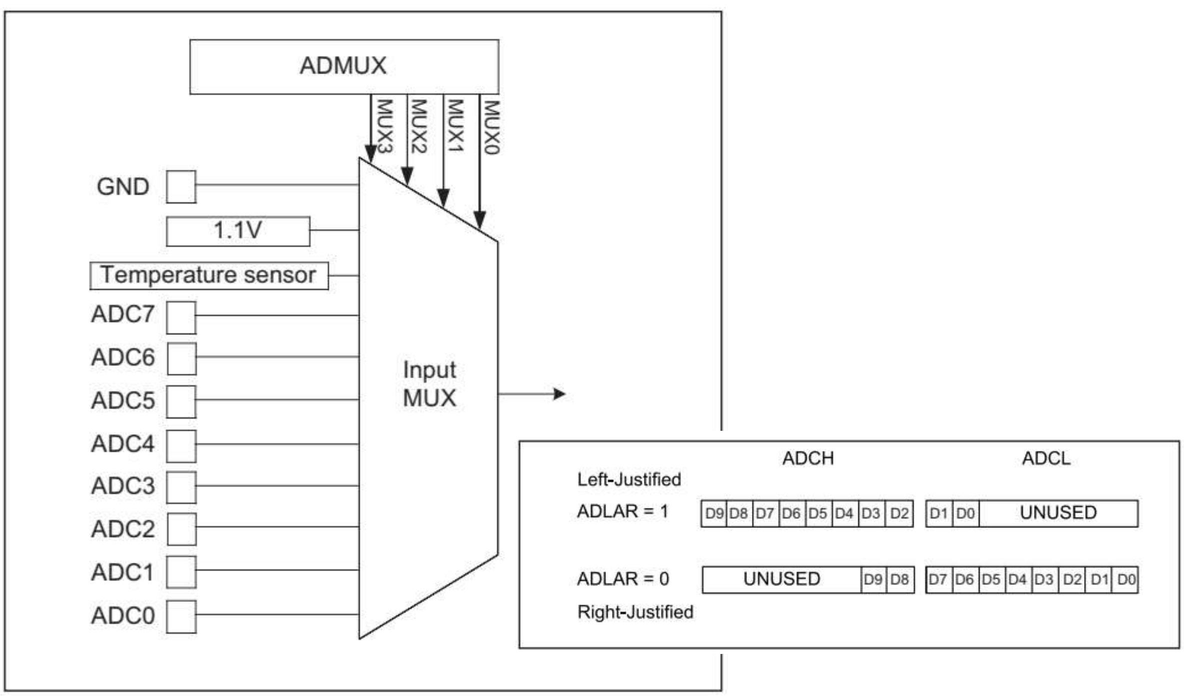
Table 13-4: V_{ref} Source Selection Table for AVR

REFS1	REFS0	V_{ref}	
0	0	AREF pin	Set externally
0	1	AVCC pin	Same as VCC
1	0	Reserved	----
1	1	Internal 1.1 V	Fixed regardless of VCC value

ADC input channel source

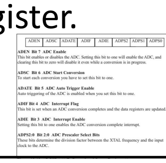
- Figure shows the schematic of the internal circuitry of input channel selection. Table below lists the values of MUX3–MUX0 bits for different inputs.
- The GND of the AVR chip is used as common ground. ADLAR bit operation The AVRs have a 10-bit ADC, which means that the result is 10 bits long and cannot be stored in a single byte.
- In AVR two 8-bit registers are dedicated to the ADC result, but only 10 of the 16 bits are used and 6 bits are unused.
- You can select the position of used bits in the bytes. If you set the ADLAR bit in ADMUX register, the result bits will be left-justified; otherwise, the result bits will be right-justified.
- Notice that changing the ADLAR bit will affect the ADC data register immediately.





ADCH: ADCL registers

- After the A/D conversion is complete, the result sits in registers ADCL (A/D Result Low Byte) and ACDH (A/D Result High Byte).
- As we mentioned before, the ADLAR bit of the ADMUX is used for making it right-justified or leftjustified because we need only 10 of the 16 bits.
- ADCSRA register The ADCSRA register is the status and control register of ADC.
- Bits of this register control or monitor the operation of the ADC. In Figure you can see a description of each bit of the ADCSRA register. We will examine some of these bits in more detail.



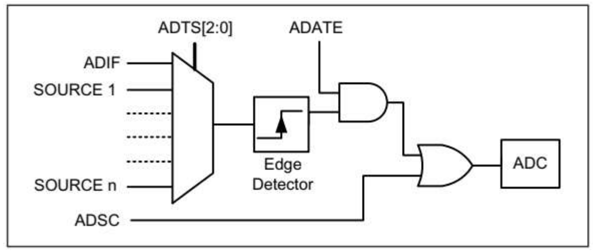
ADCSRA

ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0

- **ADEN Bit 7 ADC Enable**
- This bit enables or disables the ADC. Setting this bit to one will enable the ADC, and clearing this bit to zero will disable it even while a conversion is in progress.
- **ADSC Bit 6 ADC Start Conversion**
- To start each conversion you have to set this bit to one.
- **ADATE Bit 5 ADC Auto Trigger Enable** Auto triggering of the ADC is enabled when you set this bit to one.
- **ADIF Bit 4 ADC Interrupt Flag** This bit is set when an ADC conversion completes and the data registers are updated.
- **ADIE Bit 3 ADC Interrupt Enable** Setting this bit to one enables the ADC conversion complete interrupt.
- **ADPS2:0 Bit 2:0 ADC Prescaler Select Bits**
- These bits determine the division factor between the XTAL frequency and the input clock to the ADC.

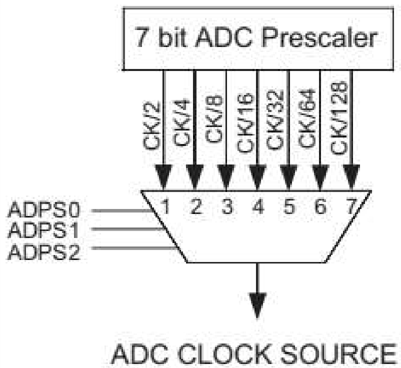
ADC Start Conversion bit

- As we stated before, an ADC has a Start Conversion input. The AVR chip has a special circuit to trigger start conversion. As you see in Figure 13-11, in addition to the ADCSC bit of ADCSRA there are other sources to trigger start of conversion. If you set the ADATE bit of ADCSRA to high, you can select auto trigger source by updating ADTS2:0 in the ADCSRB register. If ADATE is cleared, the ADTS2:0 settings will have no effect.



A/D conversion time

- As you see in Figure 13-12, by using the ADPS2:0 bits of the ADCSRA register we can set the A/D conversion time. To select the conversion time, we can select any of Fosc/2, Fosc/4, Fosc/8, Fosc/16, Fosc/32, Fosc/64, or Fosc/128 for ADC clock, where Fosc is the speed of the crystal frequency connected to the AVR chip.
- Notice that the multiplexer has 7 inputs since the option ADPS2:0 = 000 is reserved. For the AVR, the ADC requires an input clock frequency less than 200 kHz for the maximum accuracy. Look at Example 13-3 for clarification.



ADPS2	ADPS1	ADPS0	ADC Clock
0	0	0	Clk/2
0	0	1	CK/2
0	1	0	CK/4
0	1	1	CK/8
1	0	0	CK/16
1	0	1	CK/32
1	1	0	CK/64
1	1	1	CK/128

Example

- An AVR is connected to the 16 MHz crystal oscillator. Calculate the ADC frequency for (a) ADPS2:0 = 001 (b) ADPS2:0 = 100 (c) ADPS2:0 = 111

Sample-and-hold time in ADC

- A timing factor that we should know about is the acquisition time. After an ADC channel is selected, the ADC allows some time for the sample-and-hold capacitor (C hold) to charge fully to the input voltage level present at the channel.
- In the AVR, the first conversion takes 25 ADC clock cycles in order to initialize the analog circuitry and pass the sample-and-hold time. Then each consecutive conversion takes 13 ADC clock cycles.
- Table 13-6 lists the conversion times for some different conditions. Notice that sample-and-hold time is the first part of each conversion. If the conversion time is not critical in your application and you do not want to deal with calculation of ADPS2:0 you can use ADPS2:0 = 111 to get the maximum accuracy of ADC.

Table 13-6: Conversion Time Table

Condition	Sample and Hold Time (Cycles)	Total Conversion Time (Cycles)
First Conversion	14.5	25
Normal Conversion	1.5	13
Auto trigger conversion	1.5 / 2.5	13/14

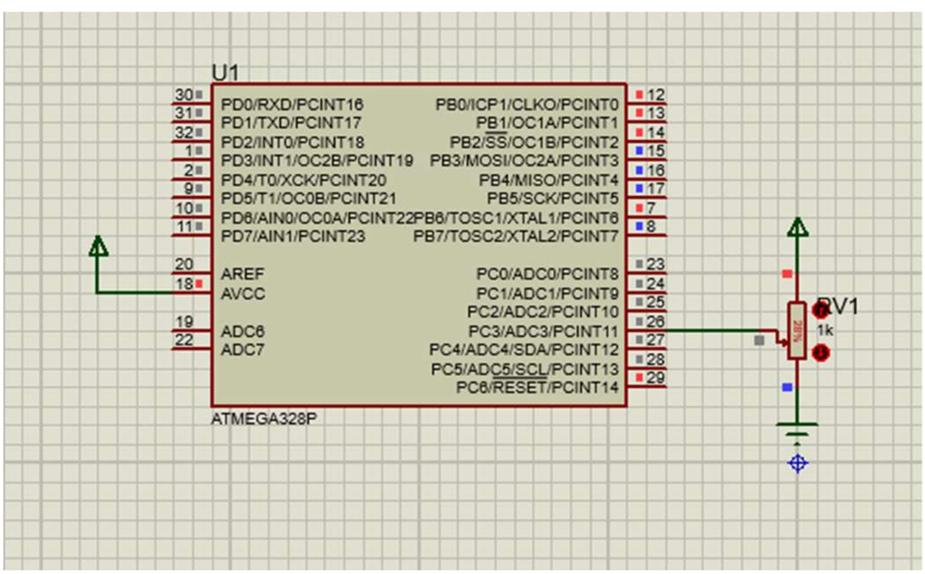
Steps in programming the A/D converter using polling

- To program the A/D converter of the AVR, the following steps must be taken:
- 1. Make the pin for the selected ADC channel an input pin.
- 2. Turn on the ADC module of the AVR because it is disabled upon power-on reset to save power.
- 3. Select the conversion speed. We use registers ADPS2:0 to select the conversion speed.
- 4. Select voltage reference and ADC input channels. We use the REFS0 and REFS1 bits in the ADMUX register to select voltage reference and the MUX3:0 bits in ADMUX to select the ADC input channel.
- 5. Activate the start conversion bit by writing a one to the ADSC bit of ADCSRA.

Steps in programming the A/D converter using polling 2

- 6. Wait for the conversion to be completed by polling the ADIF bit in the ADCSRA register.
- 7. After the ADIF bit has gone HIGH, read the ADCL and ADCH registers to get the digital data output. Notice that you have to read ADCL before ADCH; otherwise, the result will not be valid.
- 8. If you want to read the selected channel again, go back to step 5. 9. If you want to select another Vref source or input channel, go back to step 4.

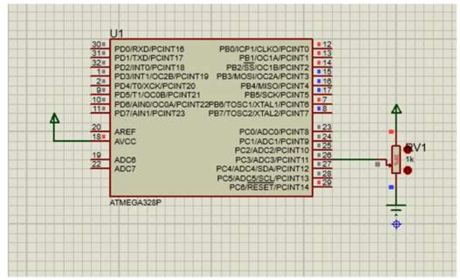
```
      LDI   R16,0xFF
      OUT   DDRB, R16       ;make Port B an output
      OUT   DDRD, R16       ;make Port D an output
      LDI   R16,0
      OUT   DDRC, R16       ;make Port C an input for ADC
      LDI   R16,0x87        ;enable ADC and select ck/128
      STS   ADCSRA, R16
      LDI   R16,0xC0        ;1.1V Vref, ADC0 single ended
      STS   ADMUX, R16      ;input, right-justified data
READ_ADC:
      LDI   R16,0x87|(1<<ADSC)
      STS   ADCSRA,R16      ;start conversion
KEEP_POLLING:               ;wait for end of conversion
      LDS   R16,ADCSRA
      SBRS  R16,ADIF        ;is it end of conversion yet?
      RJMP  KEEP_POLLING    ;keep polling end of conversion
      LDI   R16,(1<<ADIF)
      STS   ADCSRA,R16      ;write 1 to clear ADIF flag
      LDS   R16,ADCL        ;YOU HAVE TO READ ADCL FIRST
      OUT   PORTD,R16       ;give the low byte to PORTD
      LDS   R16,ADCH        ;READ ADCH AFTER ADCL
      OUT   PORTB,R16       ;give the high byte to PORTB
      RJMP  READ_ADC        ;keep repeating it
```



```
• int main(void)
• {
• DDRB =     ;

• ADMUX |=

• ADCSRA |=
• DIDR0 =
•     while(1)
•     {
•         ADCSRA |=           //hABILITO CONVERSION
• while                     ;//ESPERAR QUE ACABE CONVERSION A/D
• PORTB=
• _delay_ms(   );
•     }
• }
```



Temperature sensors

- Sensors convert physical data such as temperature, light intensity, flow, and speed to electrical signals. Depending on the transducer, the output produced is in the form of voltage, current, resistance, or capacitance. For example, temperature is converted to electrical signals using a sensor called a thermistor. A thermistor responds to temperature change by changing resistance, but its response is not linear, as seen in Table 1 and Figure 1.
- The complexity associated with writing software for such nonlinear devices has led many manufacturers to market a linear temperature sensor. Simple and widely used linear temperature sensors include the LM34 and LM35 series from National Semiconductor Corp. They are discussed next.

Temperature (C)	Tf (K ohms)
0	29.490
25	10.000
50	3.893
75	1.700
100	0.817

From William Kleitz, Digital Electronics

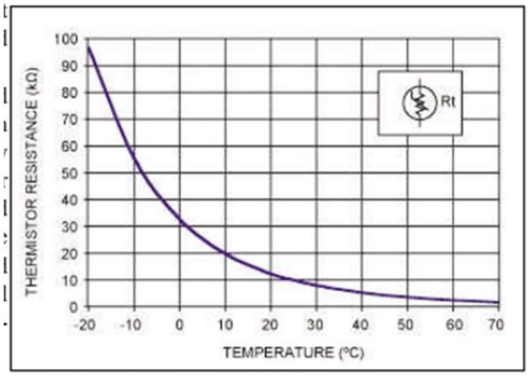
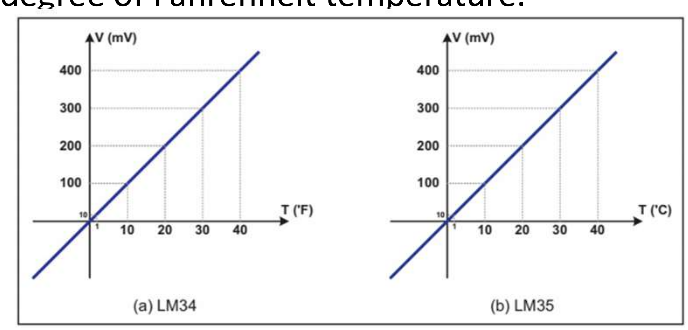


Figure 13-14. Thermistor (Copied from http://www.maximintegrated.com)

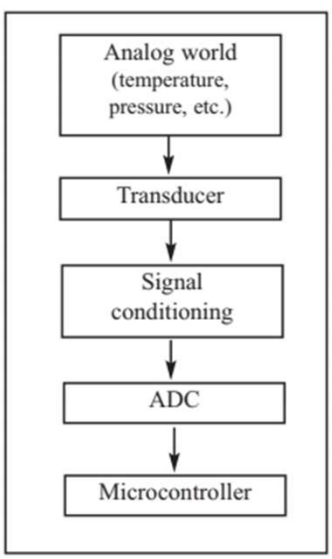
LM34 and LM35 temperature sensors

- The sensors of the LM34 series are precision integrated-circuit temperature sensors whose output voltage is linearly proportional to the Fahrenheit temperature. See Figure 13-15. The LM34 requires no external calibration because it is internally calibrated. It outputs 10 mV for each degree of Fahrenheit temperature.

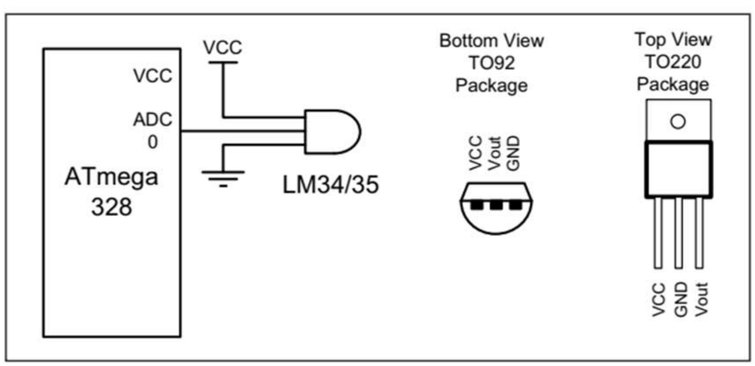


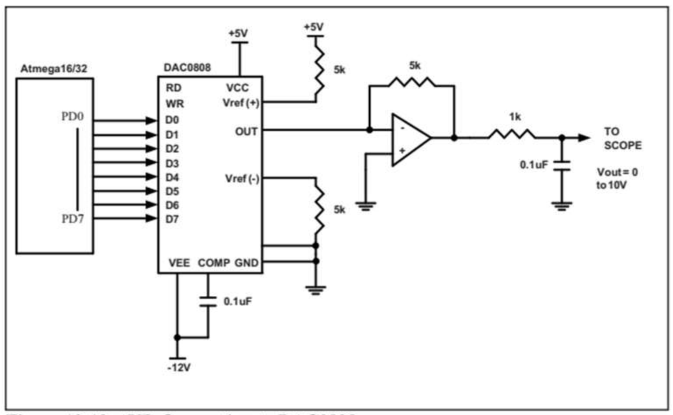
Signal conditioning

- Signal conditioning is widely used in the world of data acquisition. The most common transducers produce an output in the form of voltage, current, charge, capacitance, and resistance. We need to convert these signals to voltage, however, in order to send input to an ADC. This conversion (modification) is commonly called signal conditioning. See Figure 13-16. Signal conditioning can be current-to-voltage conversion or signal amplification. For example, the thermistor changes resistance with temperature. The change of resistance must be translated into voltages to be of any use to an ADC. We now look at the case of connecting an LM34 (or LM35) to an ADC of the ATmega328.



- for every degree of temperature change. Now, if we use the step size of 10 mV, the Vout will be 10,240 mV (10.24 V) for full-scale output.
- This is not acceptable even though the maximum temperature sensed by the LM34 is 300 degrees F, and the highest output we will get for the A/D is 3000 mV (3.00 V). Now if we use the internal 1.1 V reference voltage, the step size would be 1.1 V/1024 = 1.07 mV.
- This makes the binary output number for the ADC around nine times the real temperature because the sensor produces 10 mV for each degree of temperature change and the step size is 1.07 mV (10 mV/1.07 mV = 9.3). We can scale it by dividing it by 9.3 to get the real number for temperature. Figure 13-17 shows the pin configuration of the LM34/LM35 temperature





- True or false. The ATmega328 has an on-chip A/D converter.
- True or false. A/D of the ATmega328 is an 8-bit ADC.
- True or false. ATmega328 has 4 channels of analog input.
- True or false. The unused ADC pins of the ATmega328 can be used for I/O pins.
- True or false. The ADC conversion speed in the ATmega328 depends on the crystal frequency.
- True or false. Upon power-on reset, the ADC module of the ATmega328 is turned on and ready to go.
- True or false. The ADC module of the ATmega328 has an external pin for the start-conversion signal.
- True or false. The ADC module of the ATmega328 can convert only one channel at a time.
- True or false. The ADC module of the ATmega328 can have multiple external Vref+ at any given time.
- True or false. The ADC module of the ATmega328 can use the AVCC for Vref.
- In the ADC of ATmega328, what happens to the converted analog data? How do we know that the ADC is ready to provide us the data?
- In the ADC of ATmega328, what happens to the old data if we start conversion again before we pick up the last data?
- For the ADC of ATmega328, find the step size for each of the following Vref: (a) Vref = 1.024 V (b) Vref = 2.048 V (c) Vref = 2.56 V
- In the ATmega328, what should the Vref value be if we want a step size of 2 mV?
- In the ATmega328, what should the Vref value be if we want a step size of 3 mV?