CHAPTER 13

ADC, DAC, AND SENSOR INTERFACING

ADC Devices

- ADCs (analog-to-digital converters) are among the most widely used devices for data acquisition
  - A physical quantity, like temperature, pressure, humidity, and velocity, etc., is converted to electrical (voltage, current) signals using a device called a transducer, or sensor
  - We need an analog-to-digital converter to translate the analog signals to digital numbers, so microcontroller can read them

ADC804 Chip

- ADC804 IC is an analog-to-digital converter
  - It works with +5 volts and has a resolution of 8 bits
  - Conversion time is another major factor in judging an ADC
    - Conversion time is defined as the time it takes the ADC to convert the analog input to a digital (binary) number
    - In ADC804 conversion time varies depending on the clocking signals applied to CLK R and CLK IN pins, but it cannot be faster than 110 µs

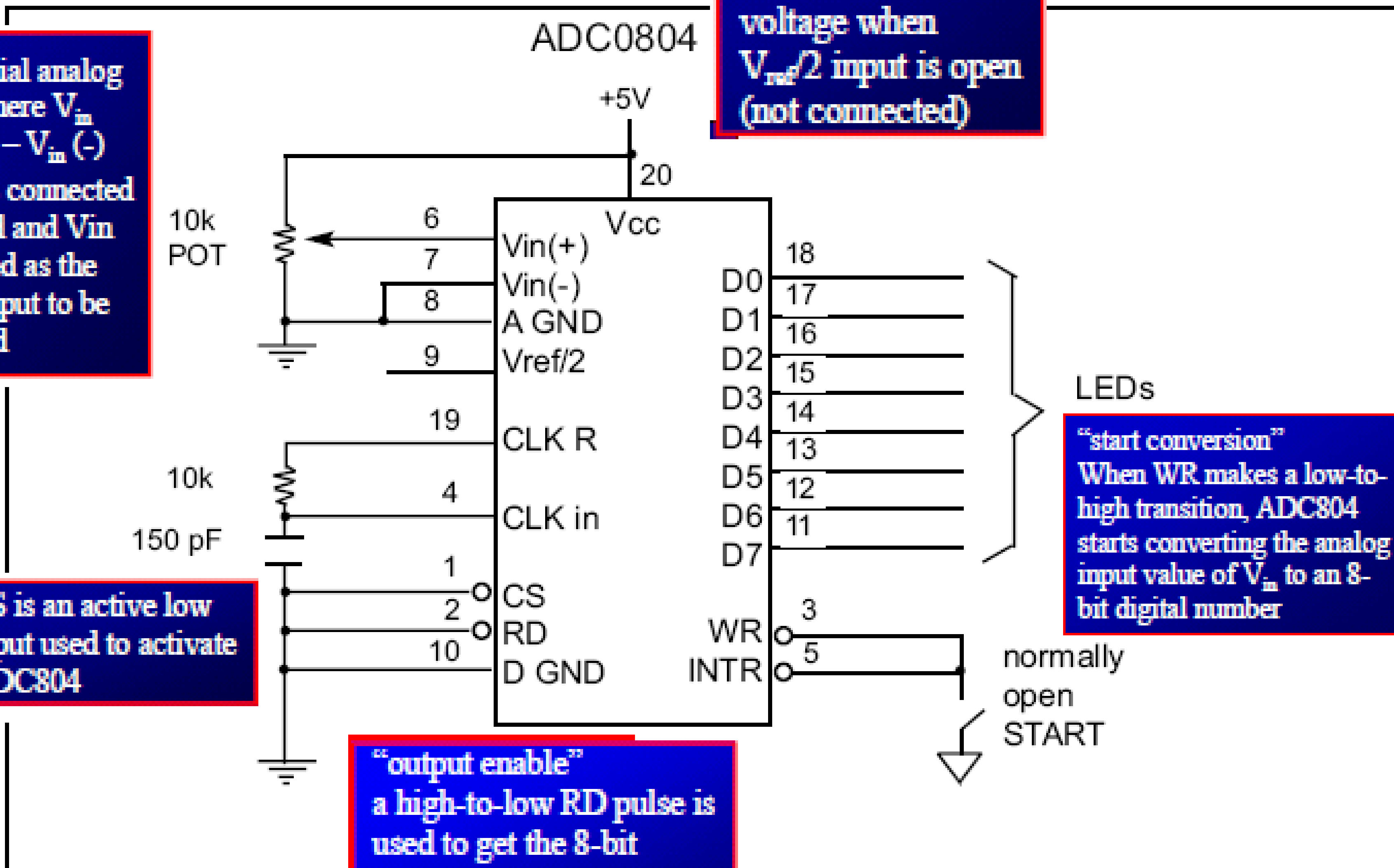


Figure 13-1. ADC0804 … n free running mode)

+5V power supply or a reference voltage when $V_{ref}/2$ input is open (not connected)

Differential analog inputs where $V_{in} = V_{in}(+) - V_{in}(-)$

Vin (-) is connected to ground and Vin (+) is used as the analog input to be converted

"start conversion"

When WR makes a low-to-high transition, ADC804 starts converting the analog input value of V_{in} to an 8-bit digital number

CS is an active low input used to activate ADC804

"output enable"

a high-to-low RD pulse is used to get the 8-bit converted data out of ADC804

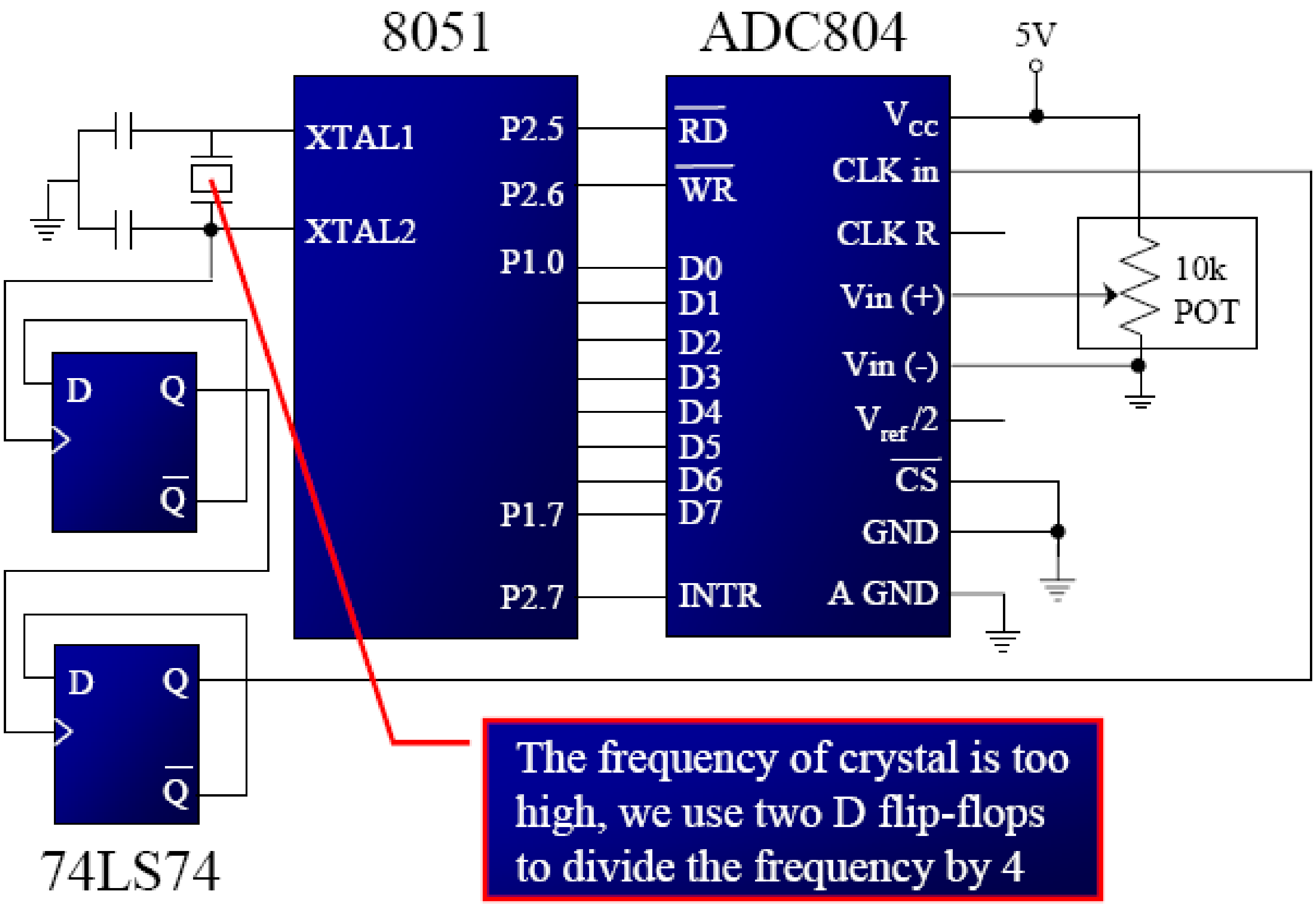
ADC804 Chip (cont.)

- CLK IN and CLK R
  - CLK IN is an input pin connected to an external clock source
  - To use the internal clock generator (also called self-clocking), CLK IN and CLK R pins are connected to a capacitor and a resistor, and the clock frequency is determined by

$$f = \frac{1}{1.1\,RC}$$

    - Typical values are R = 10K ohms and C = 150 pF
      - We get f = 606 kHz and the conversion time is 110 μs

8051 Connection to ADC804 with Clock from XTAL2 of 8051



The frequency of crystal is too high, we use two D flip-flops to divide the frequency by 4

ADC804 Chip (cont.)

- $V_{ref}/2$
  - It is used for the reference voltage
    - If this pin is open (not connected), the analog input voltage is in the range of 0 to 5 volts (the same as the V_{cc} pin)
    - If the analog input range needs to be 0 to 4 volts, $V_{ref}/2$ is connected to 2 volts

$V_{ref}/2$ Relation to V_{in} Range

$V_{ref}/2$(v)	V_{in}(V)	Step Size (mV)
Not connected*	0 to 5	5/256=19.53
2.0	0 to 4	4/255=15.62
1.5	0 to 3	3/256=11.71
1.28	0 to 2.56	2.56/256=10
1.0	0 to 2	2/256=7.81
0.5	0 to 1	1/256=3.90

Step size is the smallest change can be discerned by an ADC

ADC804 Chip (cont.)

- D0-D7
  - The digital data output pins
    - These are tri-state buffered
      - The converted data is accessed only when CS = 0 and RD is forced low
    - To calculate the output voltage, use the following formula

$$D_{out} = \frac{V_{in}}{step\ size}$$

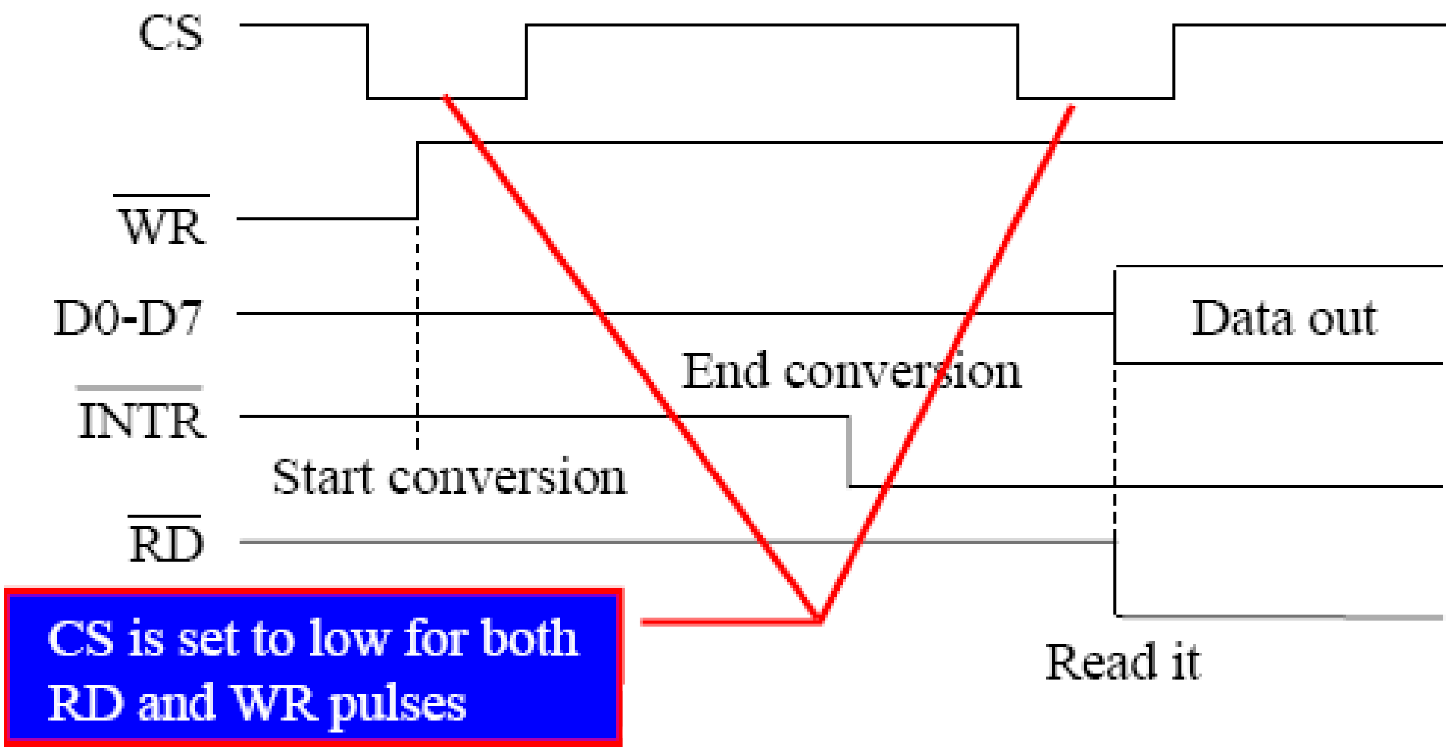
      - D_{out} = digital data output (in decimal),
      - V_{in} = analog voltage, and
      - step size (resolution) is the smallest change

ADC804 Chip (cont.)

- Analog ground and digital ground
  - Analog ground is connected to the ground of the analog V_{in}
    - To isolate the analog V_{in} signal from transient voltages caused by digital switching of the output D0 – D7
    - This contributes to the accuracy of the digital data output
  - Digital ground is connected to the ground of the V_{cc} pin

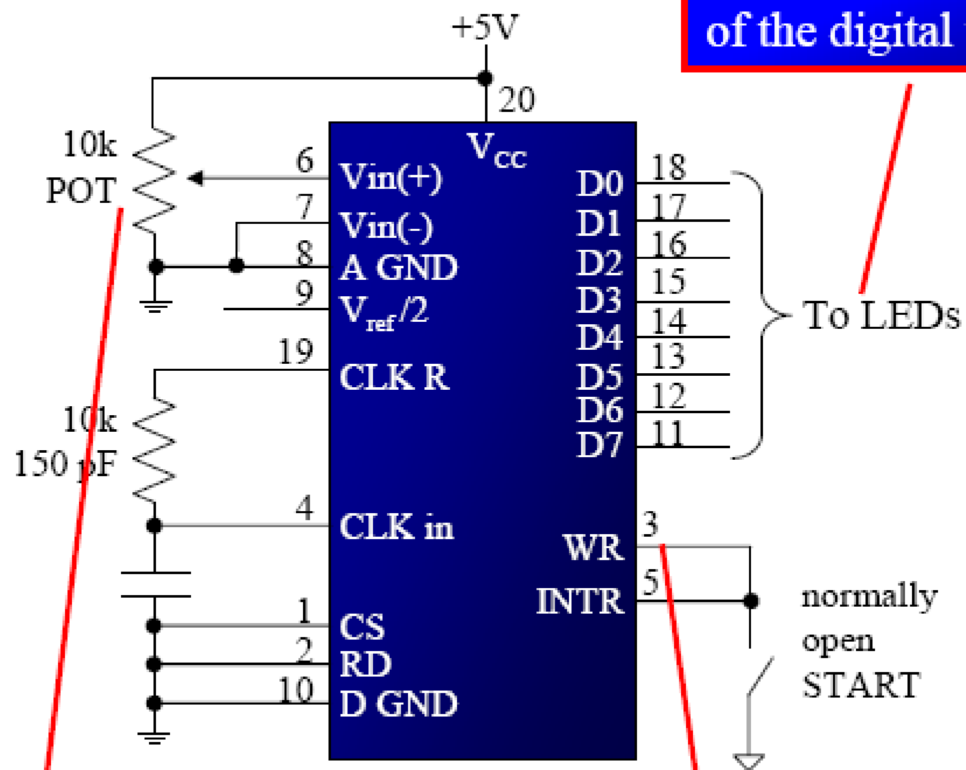
ADC804 Chip (cont.)

- The following steps must be followed for data conversion by the ADC804 chip
  - Make CS = 0 and send a low-to-high pulse to pin WR to start conversion
  - Keep monitoring the INTR pin
    - If INTR is low, the conversion is finished
    - If the INTR is high, keep polling until it goes low
  - After the INTR has become low, we make CS = 0 and send a high-to-low pulse to the RD pin to get the data out of the ADC804


CS
WR
D0-D7
Data out
End conversion
INTR
Start conversion
RD
CS is set to low for both RD and WR pulses
Read it

ADC804 Free Running Test Mode

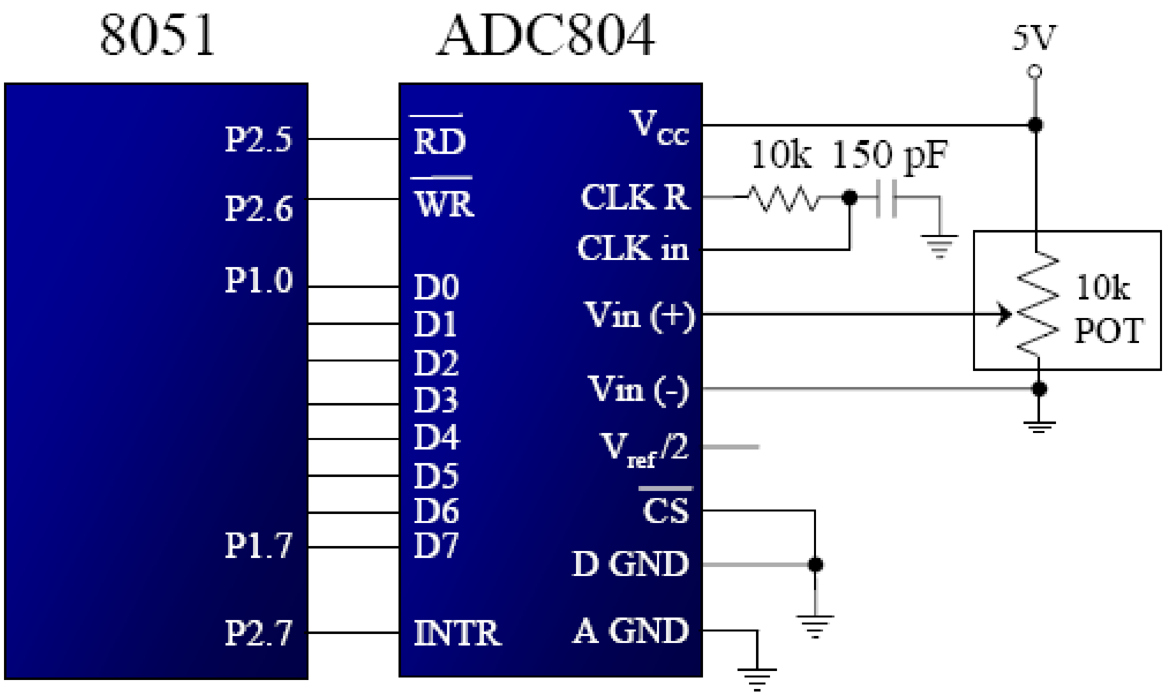
The binary outputs are monitored on the LED of the digital trainer



a potentiometer used to apply a 0-to-5 V analog voltage to input Vin (+) of the 804 ADC

The CS input is grounded and the WR input is connected to the INTR output

8051 Connection to ADC804 with Self-Clocking



Examine the ADC804 connection to the 8051 in Figure 12-7. Write a program to monitor the INTR pin and bring an analog input into register A. Then call a hex-to ACSII conversion and data display subroutines. Do this continuously.

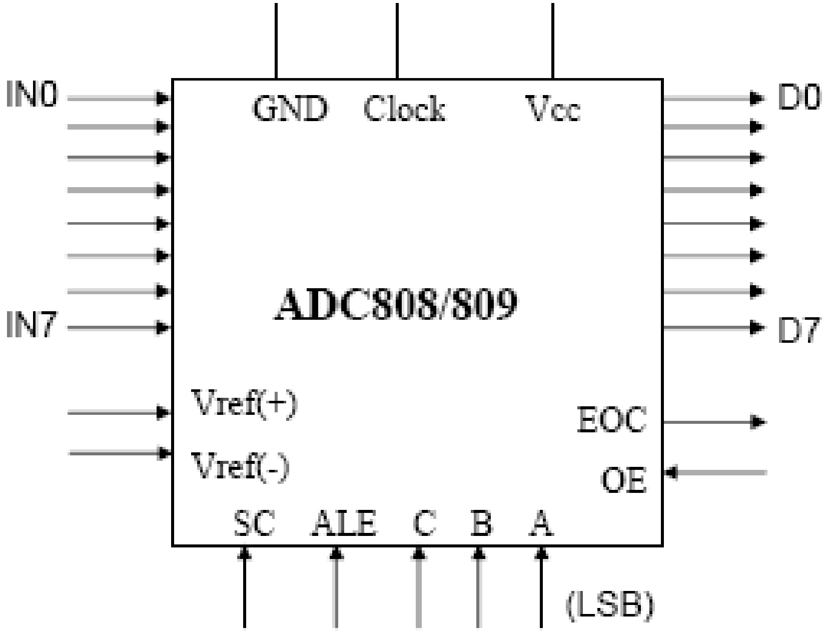
```
;p2.6=WR (start conversion needs to L-to-H pulse)
;p2.7 When low, end-of-conversion)
;p2.5=RD (a H-to-L will read the data from ADC chip)
;p1.0 - P1.7= D0 - D7 of the ADC804
;

        RD      BIT P2.5        ;RD
        WR      BIT P2.6        ;WR (start conversion)
        INTR    BIT P2.7        ;end-of-conversion
        MYDATA  EQU P1          ;P1.0-P1.7=D0-D7 of the ADC804
        MOV     P1,#0FFH        ;make P1 = input
        SETB    INTR
BACK:   CLR     WR              ;WR=0
        SETB    WR              ;WR=1 L-to-H to start conversion
HERE:   JB      INTR,HERE       ;wait for end of conversion
        CLR     RD              ;conversion finished,enable RD
        MOV     A,MYDATA        ;read the data
        ACALL   CONVERSION      ;hex-to-ASCII conversion(Chap 6)
        ACALL   DATA_DISPLAY    ;display the data(Chap 12)
        SETB    RD              ;make RD=1 for next round
        SJMP    BACK
```

ADC808/809 Chip

- ADC808 has 8 analog inputs
  - The chip has 8-bit data output just like the ADC804
  - It allows us to monitor up to 8 different transducers using only a single chip
    - The 8 analog input channels are multiplexed and selected according to table below using three address pins, A, B, and C
- Steps to program ADC808/809
  - Select an analog channel by providing bits to A, B, and C addresses

Selected Analog Channel	C	B	A
IN0	0	0	0
IN1	0	0	1
IN2	0	1	0
IN3	0	1	1
IN4	1	0	0
IN5	1	0	1
IN6	1	1	0
IN7	1	1	1

ADC808/809



ADC808/809 Chip (cont.)

- Activate the ALE pin
  - It needs an L-to-H pulse to latch in the address
- Activate SC (start conversion) by an H-to-L pulse to initiate conversion
- Monitor EOC (end of conversion) to see whether conversion is finished
- Activate OE (output enable) to read data out of the ADC chip
  - An H-to-L pulse to the OE pin will bring digital data out of the chip

ADC0848 interfacing

The ADC0848 IC is another analog-to-digital converter in the family of the ADC0800 series from National Semiconductor Corp.

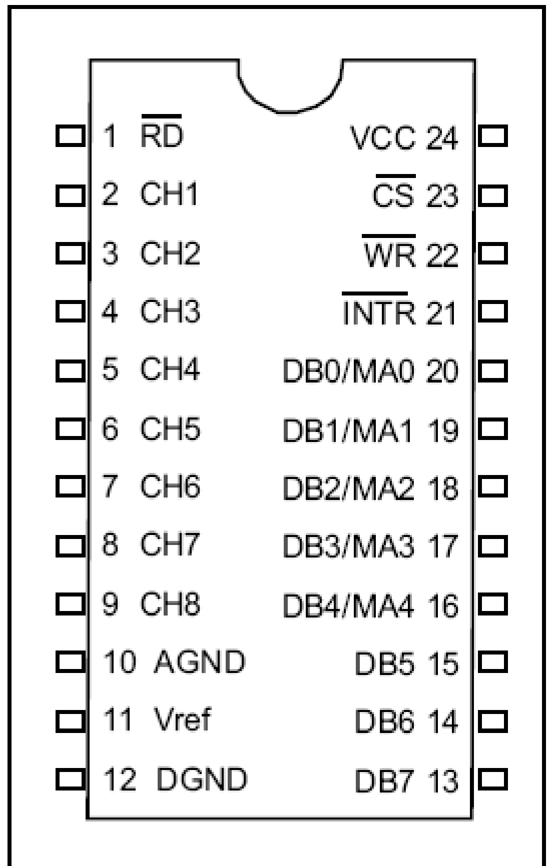


Figure 13-8. ADC0848 Chip

Example 13-1

For a given ADC0848, we have V_{ref} = 2.56 V. Calculate the D0 - D7 output if the analog input is: (a) 1.7 V, and (b) 2.1 V.

Solution:

Since the step size is 2.56/256 = 10 mV, we have the following.

(a) D_{out} = 1.7 V/10 mV = 170 in decimal, which gives us 10101011 in binary for D7 - D0.

(b) D_{out} = 2.1 V/10 mV = 210 in decimal, which gives us 11010010 in binary for D7 - D0.

DAC INTERFACING

This section will show how to interface a DAC (digital-to-analog converter) to the 8051.

Digital-to-analog (DAC) converter

The digital-to-analog converter (DAC) is a device widely used to convert digital pulses to analog signals.

MC1408 DAC (or DAC808)

In the MC1408 (DAC0808), the digital inputs are converted to current (I_{out}), and by connecting a resistor to the I_{out} pin, we convert the result to voltage. The total current provided by the I_{out} pin is a function of the binary numbers at the D0-D7 inputs of the DAC0808 and the reference current (I_{ref}), and is as follows:

$$I_{out} = I_{ref}\left(\frac{D7}{2} + \frac{D6}{4} + \frac{D5}{8} + \frac{D4}{16} + \frac{D3}{32} + \frac{D2}{64} + \frac{D1}{128} + \frac{D0}{256}\right)$$

where D0 is the LSB, D7 is the MSB for the inputs, and I_{ref} is the input current that must be applied to pin 14.

Example 13-3

Assuming that R = 5K and I_{ref} = 2 mA, calculate V_{out} for the following binary inputs:
(a) 10011001 binary (99H) (b) 11001000 (C8H)

Solution:

(a) I_{out} = 2 mA (153/256) = 1.195 mA and V_{out} = 1.195 mA × 5K = 5.975 V
(b) I_{out} = 2 mA (200/256) = 1.562 mA and V_{out} = 1.562 mA × 5K = 7.8125 V

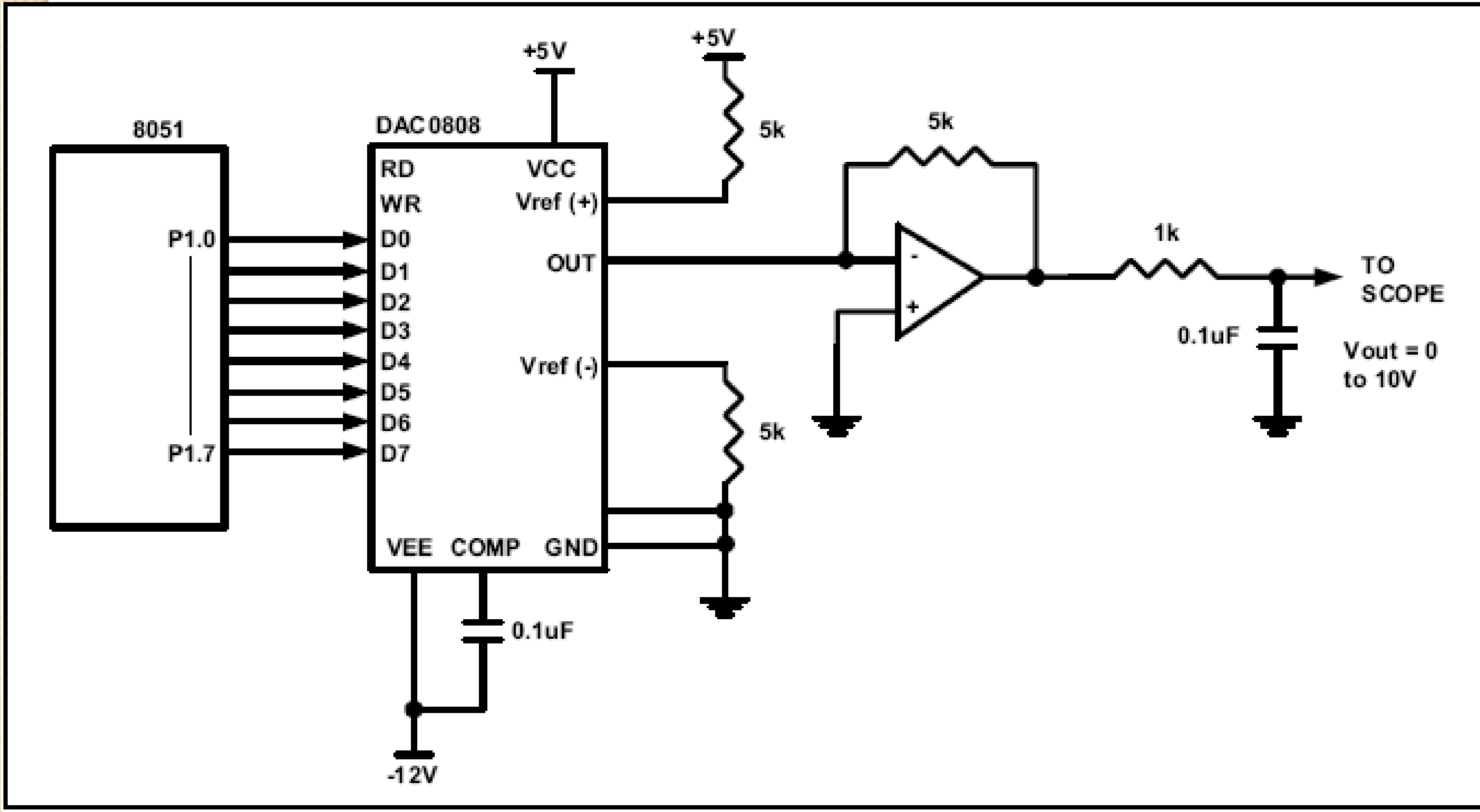


Figure 13-18. 8051 Connection to DAC808

Generating a sine wave

To generate a sine wave, we first need a table whose values represent the magnitude of the sine of angles between 0 and 360 degrees. The values for the sine function vary from -1.0 to +1.0 for 0 – to 360- ° angles.

Therefore, to achieve the full-scale 10V output, we use the following equation.

$$V_{out} = 5\text{ V} + (5 \times \sin\theta)$$

V_{out} of DAC for various angles is calculated and shown:

Table 13-7: Angle vs. Voltage Magnitude for Sine Wave

Angle θ (degrees)	Sin θ	V_{out} (Voltage Magnitude) 5 V + (5 V × sin θ)	Values Sent to DAC (decimal) (Voltage Mag. × 25.6)
0	0	5	128
30	0.5	7.5	192
60	0.866	9.33	238
90	1.0	10	255
120	0.866	9.33	238
150	0.5	7.5	192
180	0	5	128
210	–0.5	2.5	64
240	–0.866	0.669	17
270	–1.0	0	0
300	–0.866	0.669	17
330	–0.5	2.5	64
360	0	5	128

Example 13-5

Verify the values given for the following angles: (a) 30° (b) 60°.

Solution:

(a) $V_{out} = 5\ V + (5\ V \times \sin\theta) = 5\ V + 5 \times \sin 30° = 5\ V + 5 \times 0.5 = 7.5\ V$

DAC input values = $7.5\ V \times 25.6 = 192$ (decimal)

(b) $V_{out} = 5\ V + (5\ V \times \sin\theta) = 5\ V + 5 \times \sin 60° = 5\ V + 5 \times 0.866 = 9.33\ V$

DAC input values = $9.33\ V \times 25.6 = 238$ (decimal)

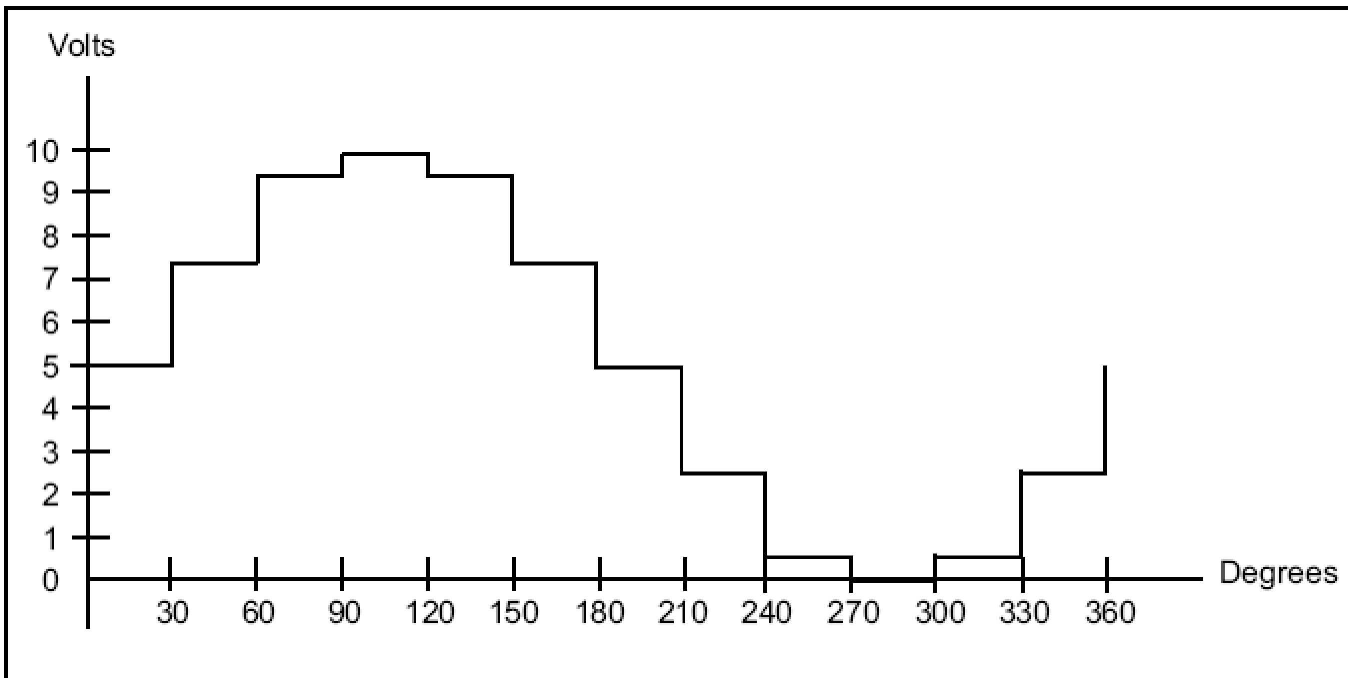


Figure 13-19. Angle vs. Voltage Magnitude for Sine Wave

This program sends the values to the DAC continuously (in an infinite loop) to produce a crude sine wave.

```
AGAIN:      MOV DPTR,#TABLE
            MOV R2,#COUNT
BACK:       CLR A
            MOVC A,@A+DPTR
            MOV P1,A
            INC DPTR
            DJNZ R2,BACK
            SJMP AGAIN
            ORG 300
TABLE:      DB 128,192,238,255,238,192 ;see Table 13-7
            DB 128,64,17,0,17,64,128
```

```
;To get a better looking sine wave, regenerate
;Table 13-7 for 2-degree angles
```

Interfacing Temperature Sensor

- A thermistor responds to temperature change by changing resistance
  - Its response is not linear
  - The complexity associated with writing software for such nonlinear devices has led many manufacturers to market the linear temperature sensor

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

LM34 and LM35 Temperature Sensors

- The sensors of the LM34/LM35 series are precision integrated-circuit temperature sensors
  - The output voltage is linearly proportional to the Fahrenheit/Celsius temperature
  - The LM34/LM35 requires no external calibration since it is inherently calibrated
  - It outputs 10 mV for each degree of Fahrenheit/Celsius temperature

Table 13-9: LM34 Temperature Sensor Series Selection Guide

Part Scale	Temperature Range	Accuracy	Output
LM34A	–50 F to +300 F	+2.0 F	10 mV/F
LM34	–50 F to +300 F	+3.0 F	10 mV/F
LM34CA	–40 F to +230 F	+2.0 F	10 mV/F
LM34C	–40 F to +230 F	+3.0 F	10 mV/F
LM34D	–32 F to +212 F	+4.0 F	10 mV/F

Note: Temperature range is in degrees Fahrenheit.

Table 13-10: LM35 Temperature Sensor Series Selection Guide

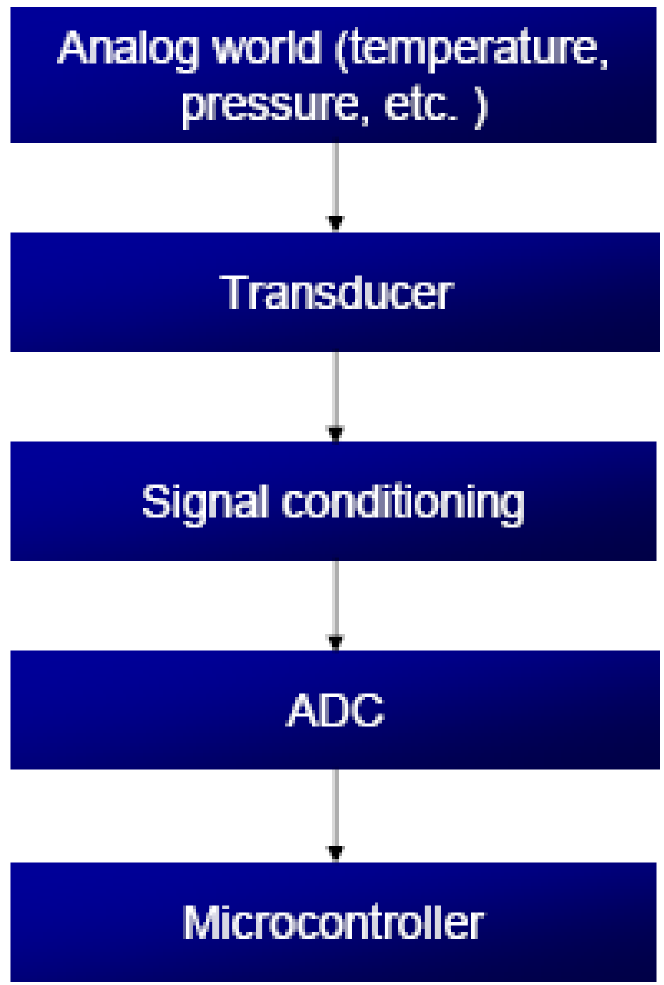
Part	Temperature Range	Accuracy	Output Scale
LM35A	–55 C to +150 C	+1.0 C	10 mV/C
LM35	–55 C to +150 C	+1.5 C	10 mV/C
LM35CA	–40 C to +110 C	+1.0 C	10 mV/C
LM35C	–40 C to +110 C	+1.5 C	10 mV/C
LM35D	0 C to +100 C	+2.0 C	10 mV/C

Note: Temperature range is in degrees Celsius.

Signal Conditioning and Interfacing LM35

- Signal conditioning is a widely used term in the world of data acquisition
  - It is the conversion of the signals (voltage, current, charge, capacitance, and resistance) produced by transducers to voltage, which is sent to the input of an A-to-D converter
    - Signal conditioning can be a current-to-voltage conversion or a signal amplification
    - The thermistor changes resistance with temperature
      - The change of resistance must be translated into voltage in order to be of any use to an ADC

Getting Data From the Analog World



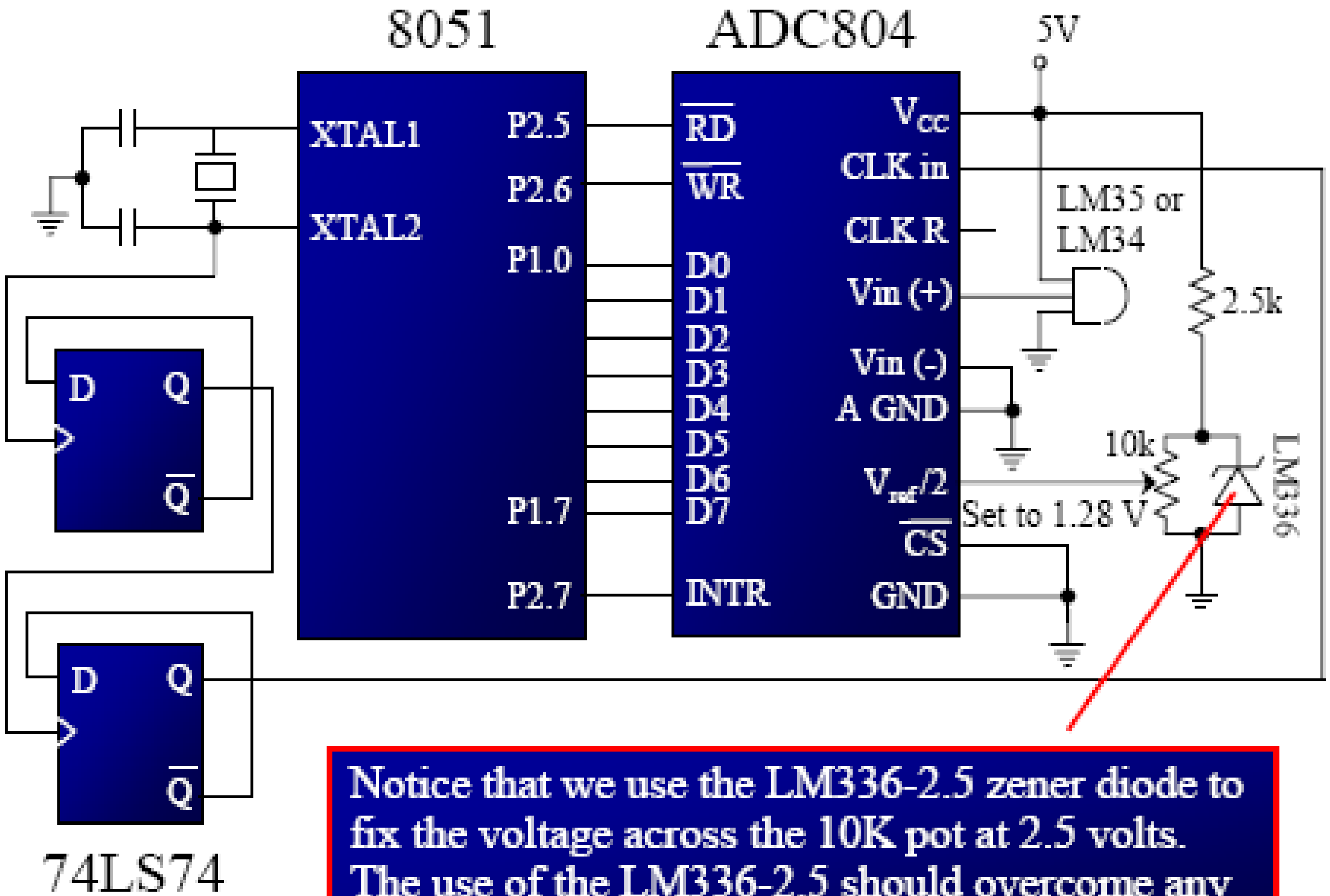
Example:

Look at the case of connecting an LM35 to an ADC804. Since the ADC804 has 8-bit resolution with a maximum of 256 steps and the LM35 (or LM34) produces 10 mV for every degree of temperature change, we can condition Vin of the ADC804 to produce a Vout of 2560 mV full-scale output. Therefore, in order to produce the full-scale Vout of 2.56 V for the ADC804, We need to set Vref/2 = 1.28. This makes Vout of the ADC804 correspond directly to the temperature as monitored by the LM35.

Temperature vs. Vout of the ADC804

Temp. (C)	Vin (mV)	Vout (D7 – D0)
0	0	0000 0000
1	10	0000 0001
2	20	0000 0010
3	30	0000 0011
10	100	0000 1010
30	300	0001 1110

8051 Connection to ADC804 and Temperature Sensor



Notice that we use the LM336-2.5 zener diode to fix the voltage across the 10K pot at 2.5 volts. The use of the LM336-2.5 should overcome any fluctuations in the power supply

Signal conditioning and interfacing the LM35 to the 8051

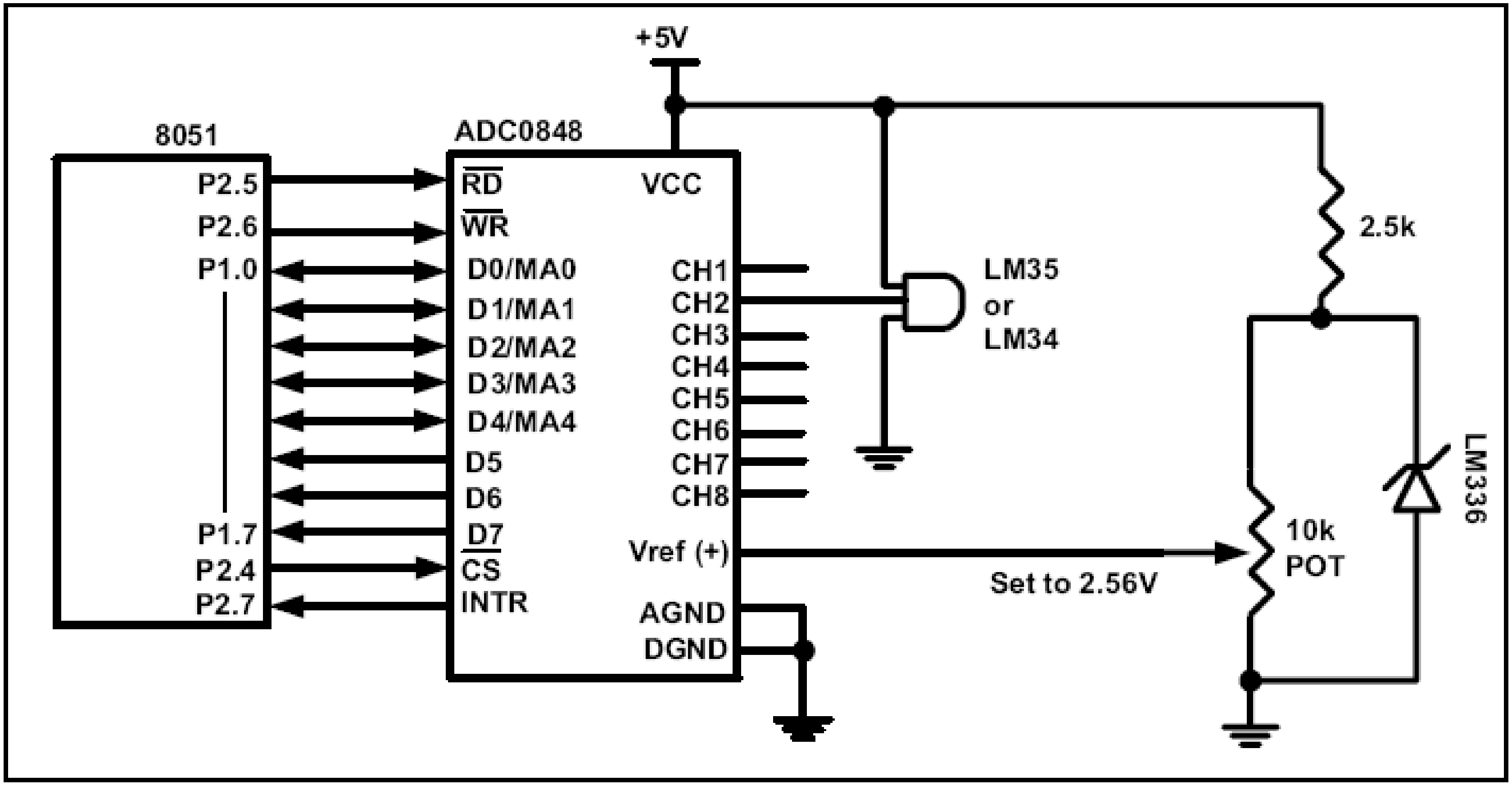


Figure 13-21. 8051 Connection to ADC0848 and Temperature Sensor

Reading and displaying temperature

```
;Program 13-1
;Assembly code to read temperature, convert it,
;and put it on P0 with some delay
      RD     BIT P2.5           ;RD
      WR     BIT P2.6           ;WR (start conversion)
      INTR   BIT P2.7           ;end-of-conversion
      MYDATA      EQU P1        ;P1.0-P1.7=D0-D7 of the ADC0848
      MOV    P1,#0FFH           ;make P1 = input
      SETB   INTR
BACK: CLR    WR                 ;WR=0
      SETB   WR                 ;WR=1 L-to-H to start conversion
HERE: JB     INTR,HERE          ;wait for end of conversion
      CLR    RD                 ;conversion finished,enable RD
      MOV    A,MYDATA           ;read the data from ADC0848
      ACALL CONVERSION                 ;hex-to-ASCII conversion
      ACALL DATA_DISPLAY        ;display the data
      SETB   RD                 ;make RD=1 for next round
      SJMP   BACK
```

```
CONVERSION:
      MOV   B,#10
      DIV   AB
      MOV   R7,B         ;least significant byte
      MOV   B,#10
      DIV   AB
      MOV   R6,B
      MOV   R5,A         ;most significant byte
      RET

DATA_DISPLAY
      MOV   P0,R7
      ACALL DELAY
      MOV   P0,R6
      ACALL DELAY
      MOV   P0,R5
      ACALL DELAY
      RET
```