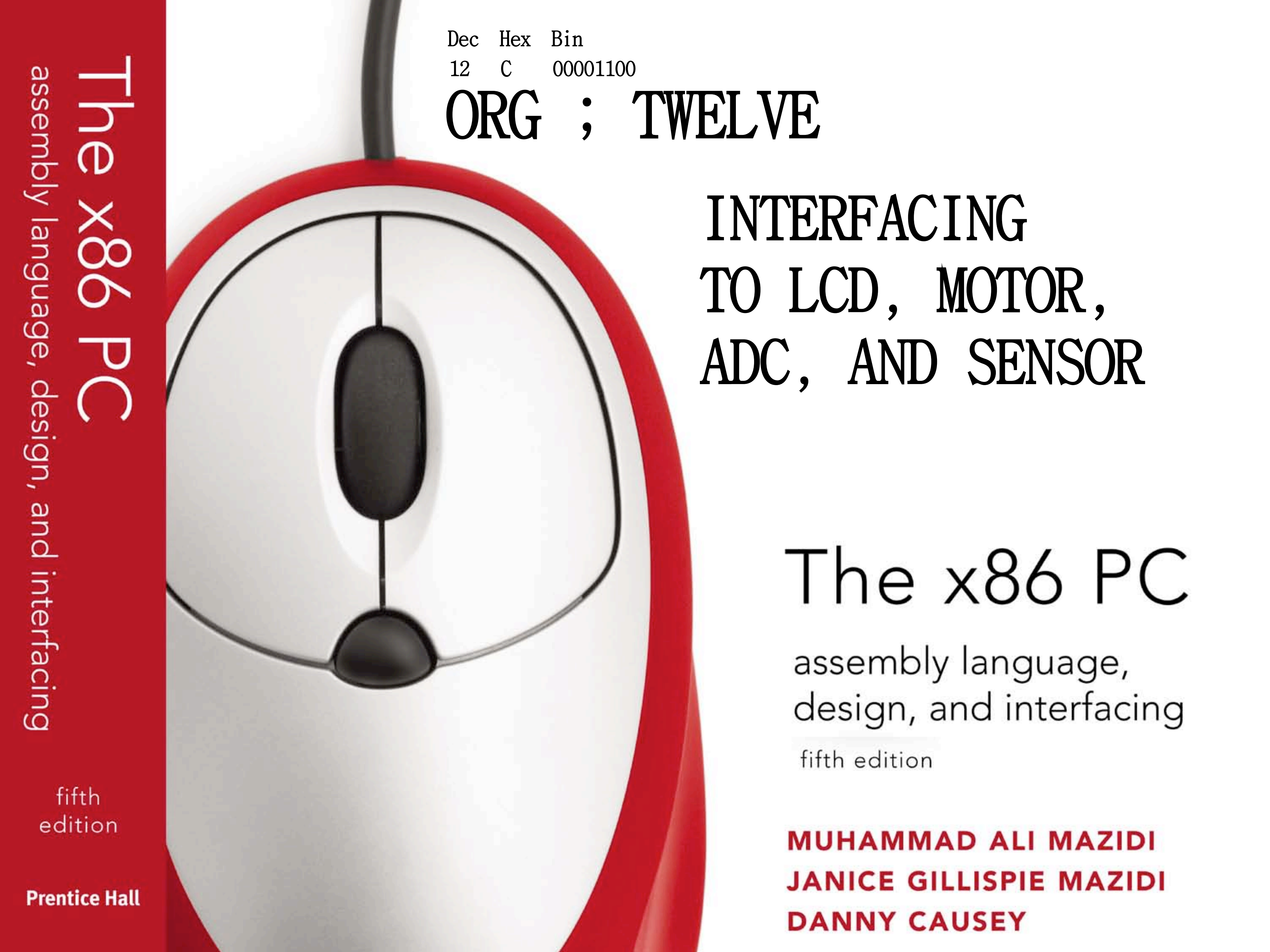
Dec	Hex	Bin
12	C	00001100

ORG ; TWELVE

INTERFACING TO LCD, MOTOR, ADC, AND SENSOR



OBJECTIVES
this chapter enables the student to:

- Diagram and code corresponding Assembly & C/C++ programs for interfacing of a PC to…
  - An LCD.
  - A stepper motor.
  - A DAC (digital-to-analog converter) device.
  - An ADC (analog-to-digital converter) device.
    - And show the interfacing of ADC devices to sensors.

12.1: INTERFACING TO AN LCD
LCD operation

- The LCD is replacing LEDs due to:
  - 1. The declining prices of LCDs.
  - 2. Ability to display numbers, characters, and graphics.
  - 3. Incorporation of the refreshing controller into the LCD itself, relieving the CPU of the task of refreshing the LCD.
  - 4. Ease of programming for both characters and graphics.

12.1: INTERFACING TO AN LCD
LCD pin descriptions

14 pin LCD

Function & pin positions

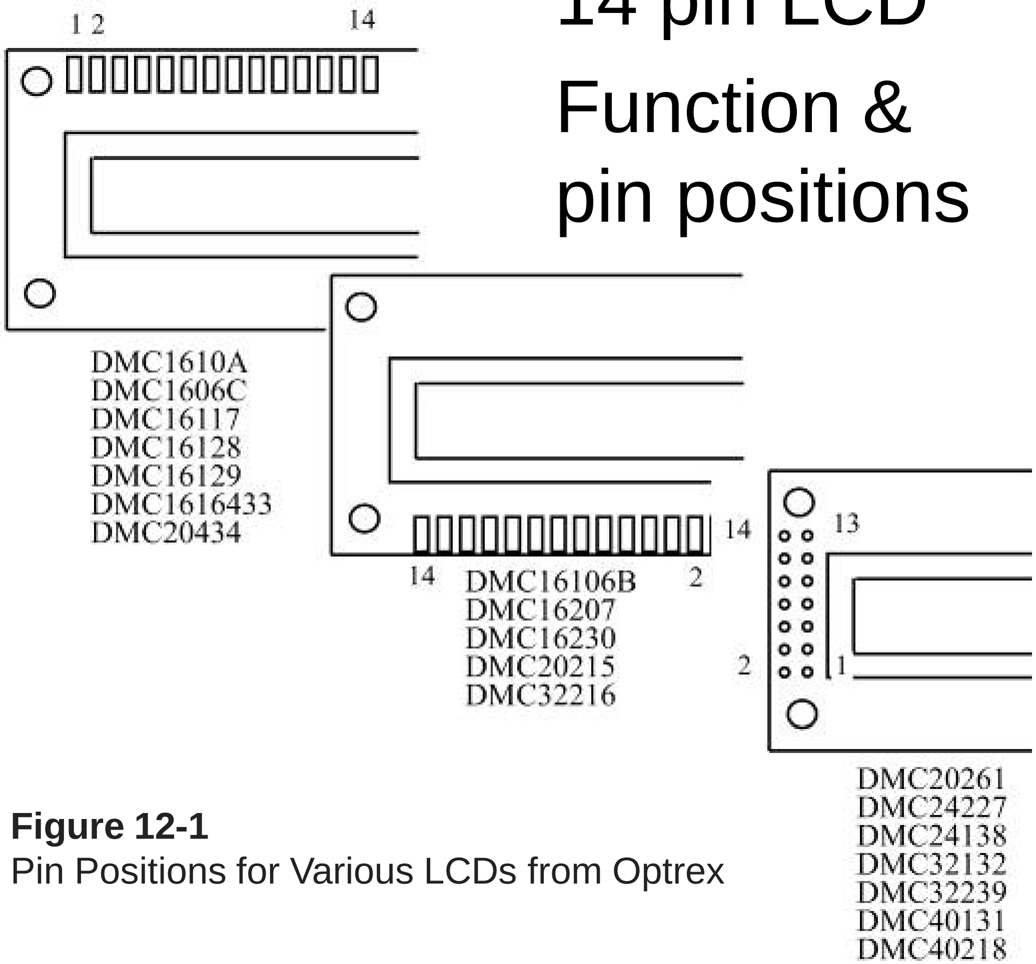


Figure 12-1
Pin Positions for Various LCDs from Optrex

Table 12-1: Pin Descriptions for LCD

Pin	Symbol	I/O	Description
1	V_{SS}	--	Ground
2	V_{CC}	--	+5V power supply
3	V_{EE}	--	Power supply to control contrast
4	RS	I	RS = 0 to select command register, RS = 1 to select data register
5	R/W	I	R/W = 0 for write, R/W = 1 for read
6	E	I/O	Enable
7	DB0	I/O	The 8-bit data bus
8	DB1	I/O	The 8-bit data bus
9	DB2	I/O	The 8-bit data bus
10	DB3	I/O	The 8-bit data bus
11	DB4	I/O	The 8-bit data bus
12	DB5	I/O	The 8-bit data bus
13	DB6	I/O	The 8-bit data bus
14	DB7	I/O	The 8-bit data bus

12.1: INTERFACING TO AN LCD
LCD pin descriptions

- **VCC** - provides +5V.
- **VSS** - ground.
- **VEE** - used for controlling the LCD contrast.
- **RS**, register select - selects one of two registers inside the LCD; RS is used for their selection:
  - If RS = 0, the instruction command code register is selected, allowing the user to send a command such as clear display, cursor at home, and so on.
  - If RS = 1, the data register is selected, allowing the user to send data to be displayed on the LCD. (or data to be retrieved)

12.1: INTERFACING TO AN LCD
LCD pin descriptions

- **R/W**, read/write - input allows the user to write information into the LCD or read information from it.
  – R/W = 1 when reading; R/W = 0 when writing.
- **E**, enable - used by the LCD to latch information presented to its data pins.
- **D0–D7** - data pins used to send information to the LCD or read contents of LCD internal registers.
- Instruction command codes can be sent to the LCD. in order to clear the display, force the cursor to the home position, or blink the cursor.
  – See Table 12-2 on page 317.

12.1: INTERFACING TO AN LCD
sending commands to LCDs

Table 12-2: LCD Command Codes

Code (Hex)	Command to LCD Instruction Register
1	Clear display screen
2	Return home
4	Decrement cursor (shift cursor to left)
6	Increment cursor (shift cursor to right)
5	Shift display right
7	Shift display left
8	Display off, cursor off
A	Display off, cursor on
C	Display on, cursor off
E	Display on, cursor on
F	Display on, cursor blinking
10	Shift cursor position to left
14	Shift cursor position to right
18	Shift the entire display to the left
1C	Shift the entire display to the right
80	Force cursor to beginning of 1st line
C0	Force cursor to beginning of 2nd line
38	2 lines and 5 × 7 matrix

To send any commands from Table 12-2 to the LCD, make pin **RS** = 0 and send a *high-to-low* pulse to the **E** pin to enable the internal latch of the LCD.

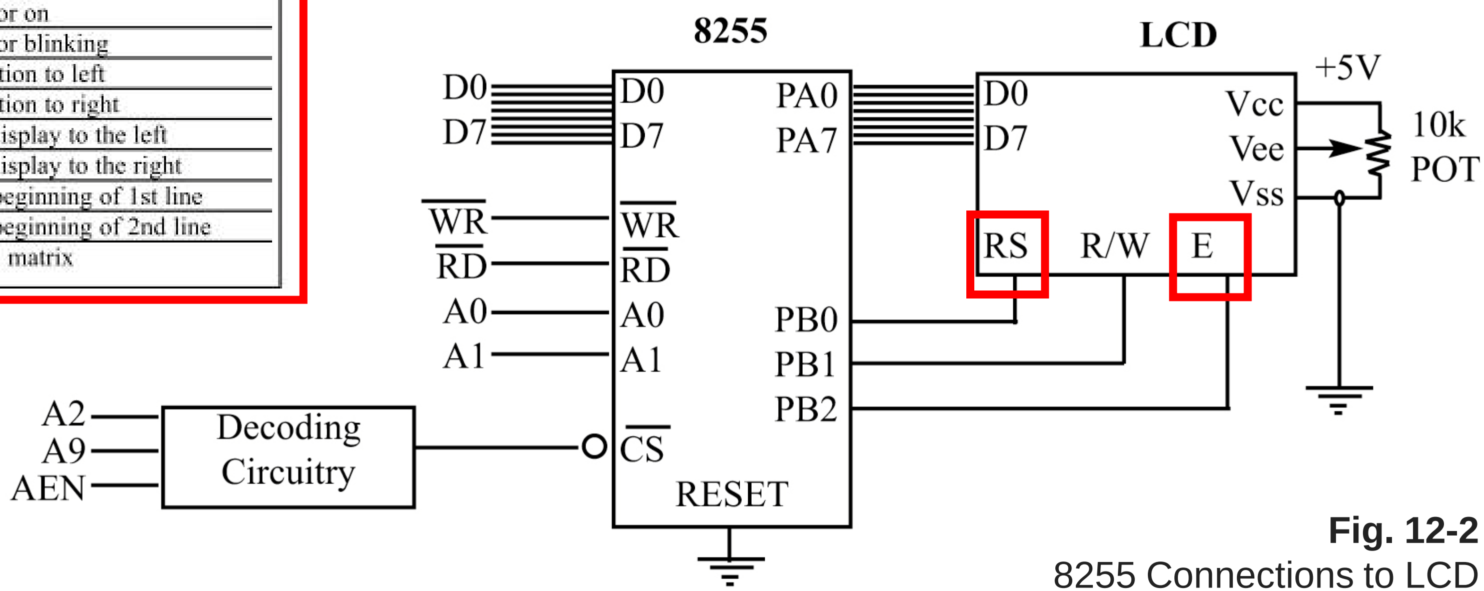


Fig. 12-2
8255 Connections to LCD

- Note the following for the connection in Fig. 12-2:
  - 1. The LCD data pins are connected to 8255 Port **A**.
  - 2. The LCD **RS** pin is connected to PB0 of 8255 Port **B**.
  - 3. The LCD **R/W** pin is connected to PB1 of 8255 Port **B**.
  - 4. The LCD **E** pin is connected to PB2 of 8255 Port **B**.
  - 5. Ports **A** *and* **B** are configured as output ports.

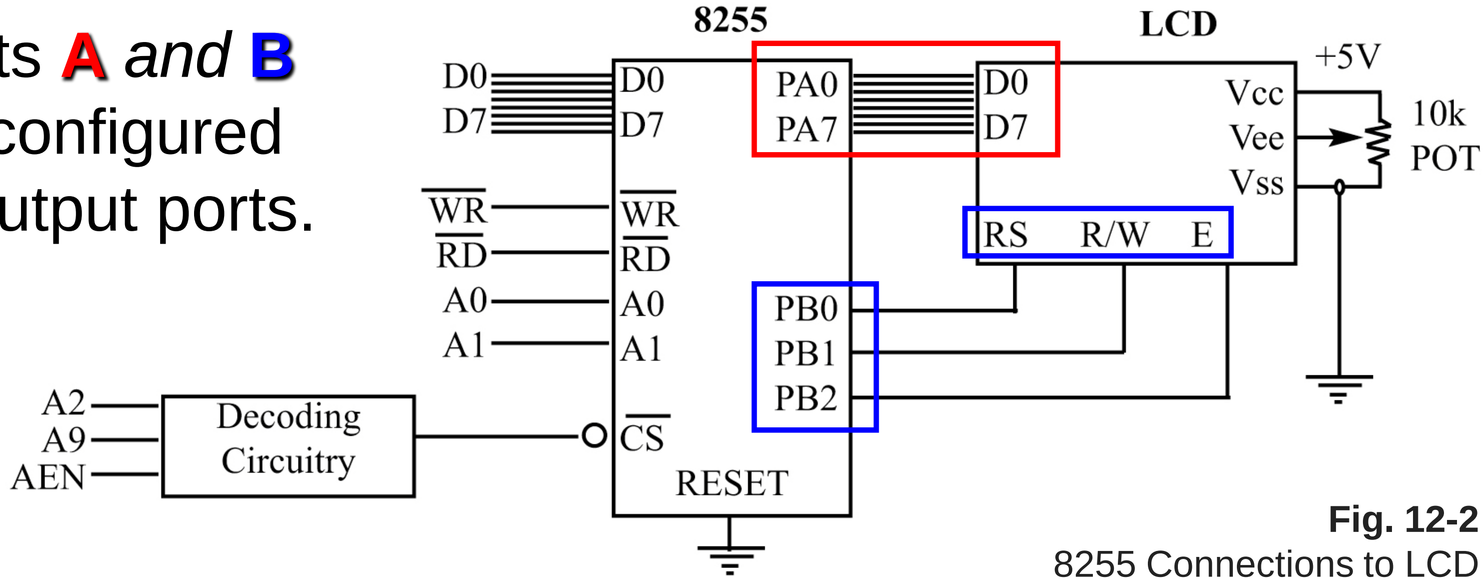


Fig. 12-2
8255 Connections to LCD

sending commands to LCDs

Partial list of code to *send* necessary *commands* to the LCD:

```
;The following sends all the necessary commands to the LCD
 MOV  AL,38H        ;initialize LCD for 2 lines & 5x7 matrix
 CALL COMNDWRT      ;write the command to LCD
 CALL DELAY         ;wait before issuing the next command
 CALL DELAY         ;this command needs lots of delay
 CALL DELAY
 MOV  AL,0EH        ;send command for LCD on, cursor on
 CALL COMNDWRT      ;write the command to LCD
 CALL DELAY         ;wait before issuing the next command
 MOV  AL,01         ;clear LCD
 CALL COMNDWRT
 CALL DELAY         ;wait
 MOV  AL,06         ;command for shifting cursor right
 CALL COMNDWRT
 CALL DELAY         ;wait
```

See the entire program listing on page 318 of your textbook.

12.1: INTERFACING TO AN LCD
sending data to the LCD

- To send *data* to the LCD for display, set RS = 1
  - Send a high-to-low pulse to the E pin to enable the internal latch of the LCD.

Partial list of code to *send* necessary *data* to the LCD:

```
MOV  AL,'Y'      ;display 'Y' letter
CALL DATWRIT     ;issue it to LCD
CALL DELAY       ;wait before issuing the next character
MOV  AL,'E'      ;display 'E' letter
CALL DATWRIT     ;issue it to LCD
CALL DELAY       ;wait before issuing the next character
MOV  AL,'S'      ;display 'S' letter
CALL DATWRIT     ;issue it to LCD
CALL DELAY       ;wait
```

See the entire program listing on page 319 of your textbook.

12.1: INTERFACING TO AN LCD
checking LCD busy flag

- The above programs used a time delay before issuing the next data or command.
  - To allows the LCD sufficient time to get ready to accept the next data.
- The LCD has a *busy flag* can be monitored, and data issued when it is ready, speeding the process.
  - To check the busy flag, read command register bit D7.
    - D7 = 1 (busy flag = 1) - the LCD is busy, and will not accept any new information.
    - D7 = 0 - the LCD is ready to receive new information.
    - See the code to monitor the busy flag on pages 320 - 321.

12.1: INTERFACING TO AN LCD
LCD cursor position

- One can put data in the LCD at any location.
  - For the 20 x 2 LCD, the address for the first location of line 1 is 80H, and for line 2 it is C0H.
    - AAAAAAA = 0000000 to 0100111 for line 1.
    - AAAAAAA = 1000000 to 1100111 for line 2

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	1	A	A	A	A	A	A	A

UA12-1 Address locations and how they are accessed

Table 12-3: LCD Addressing

	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
Line 1 (min)	1	0	0	0	0	0	0	0
Line 1 (max)	1	0	1	0	0	1	1	1
Line 2 (min)	1	1	0	0	0	0	0	0
Line 2 (max)	1	1	1	0	0	1	1	1

12.1: INTERFACING TO AN LCD
LCD programming in Visual C/C++

- Figs. 12-4 and 12-5 show timing diagrams for LCD write and read timing, respectively.

Table 12-4: List of LCD Instructions

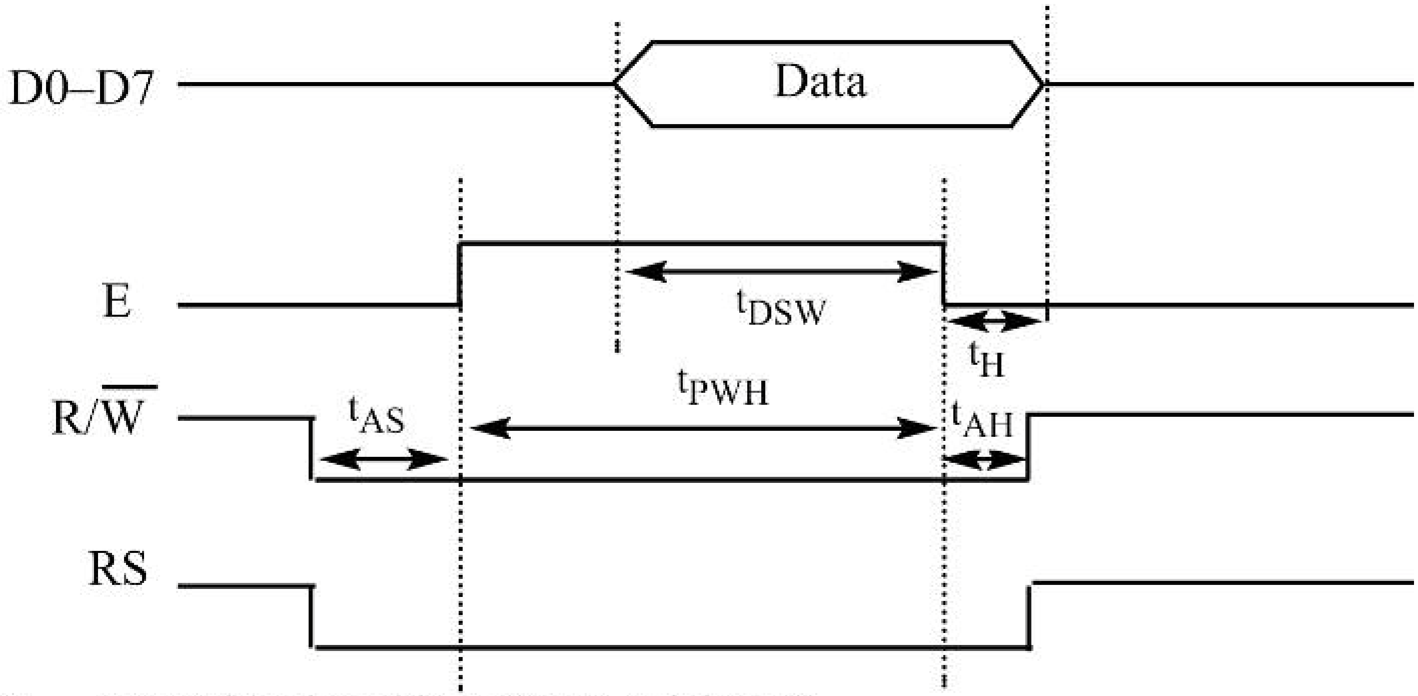
Instruction	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	Description	Execution Time (Max)
Clear Display	0	0	0	0	0	0	0	0	0	1	Clears entire disp… RAM address 0 i…	
Return Home	0	0	0	0	0	0	0	0	1	-	Sets DD RAM a… counter. Also ret… shifted to origina… contents remain unchanged.	
Entry Mode Set	0	0	0	0	0	0	0	1	I/D	S	Sets cursor move direction and specifies shift of display. These operations are	40 µs

The write operation happens on the *H-to-L* pulse of pin **E**.

The read is activated on the *L-to-H* pulse of pin **E**.

See the entire table on page 325 of your textbook.

12.1: INTERFACING TO AN LCD
LCD programming in Visual C/C++



t_{PWH} = Enable pulse width = 450 ns (minimum)

t_{DSW} = Data setup time = 195 ns (minimum)

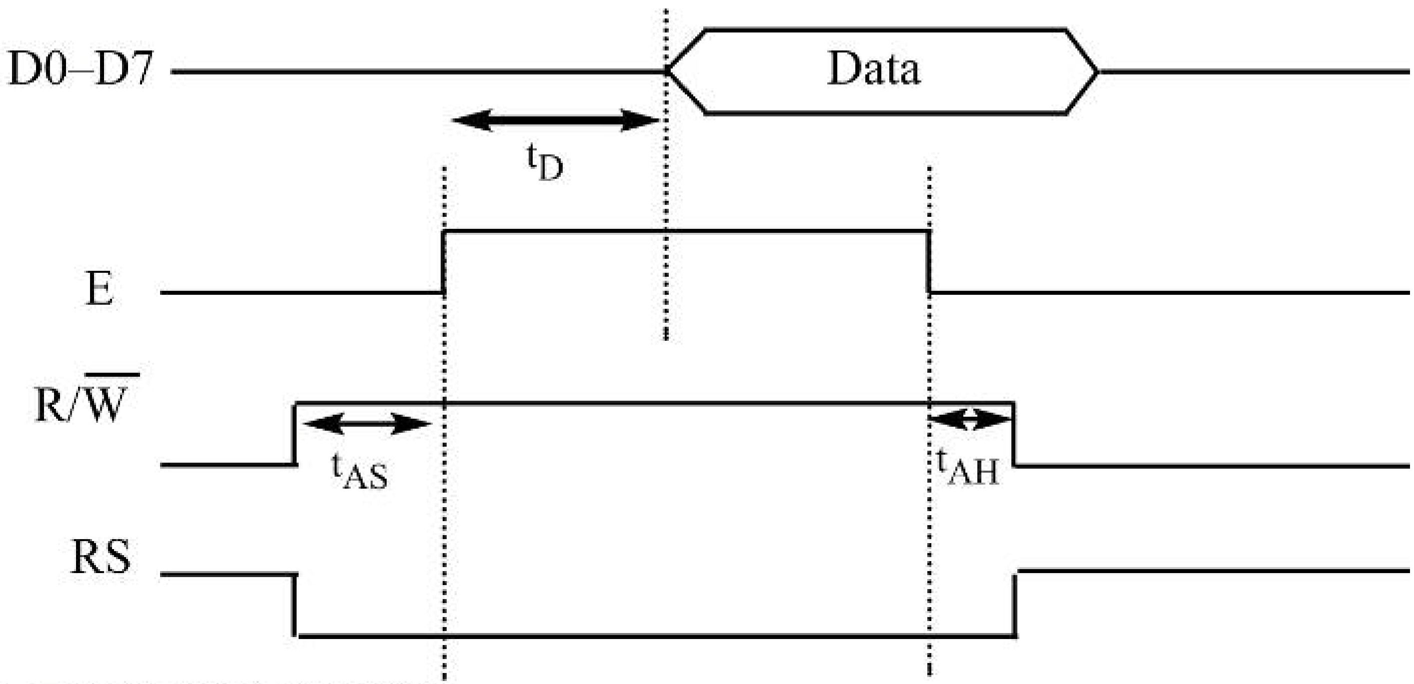
t_H = Data hold time = 10 ns (minimum)

t_{AS} = Setup time prior to E (going high) for both RS and R/W = 140 ns (minimum)

t_{AH} = Hold time after E has come down for both RS and R/W = 10 ns (minimum)

Figure 12-4 LCD Write Timing

12.1: INTERFACING TO AN LCD
LCD programming in Visual C/C++



t_D = Data output delay time

t_{AS} = Setup time prior to E (going high) for both RS and R/W = 140 ns (minimum)

t_{AH} = Hold time after E has come down for both RS and R/W = 10 ns (minimum)

Note: Read requires an L-to-H pulse for the E pin.

Figure 12-5 LCD Read Timing

12.2: INTERFACING TO A STEPPER MOTOR
stepper motors

- A stepper motor translates electrical pulses into mechanical movement.
  - Used for position control in disk drives, printers & robotics.

Every stepper motor has a permanent magnet rotor (shaft) surrounded by a stator.

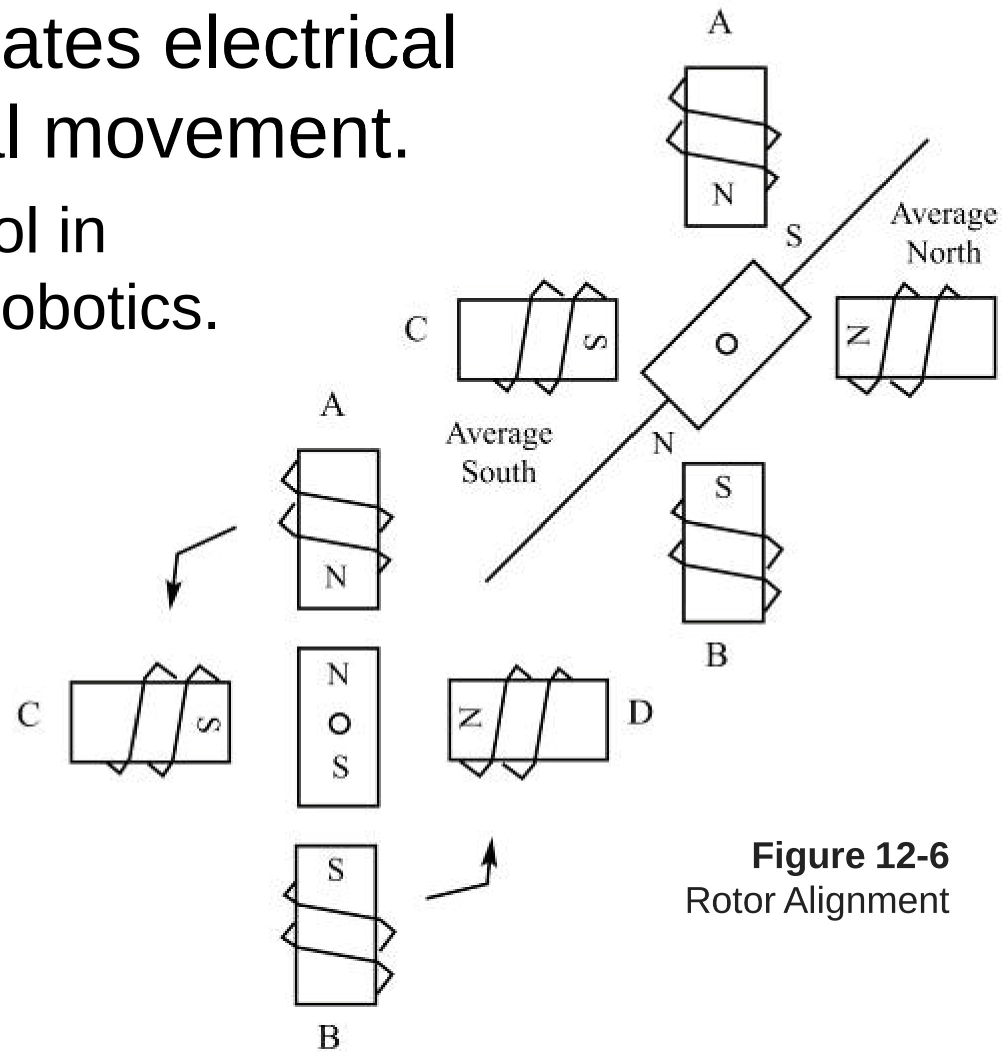


Figure 12-6
Rotor Alignment

12.2: INTERFACING TO A STEPPER MOTOR

stepper motors

- Most common stepper motors have four stator windings, paired with a center-tapped common.
  - Commonly referred to as a *four-phase* stepper motor.

The center tap allows the change of current direction in each of two coils when a winding is grounded, causing a polarity change of the stator.

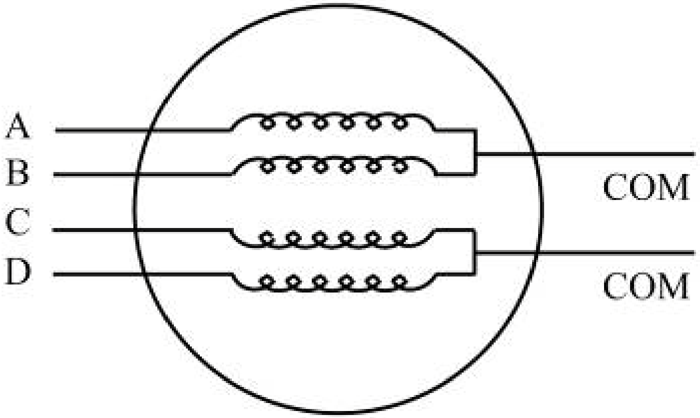


Figure 12-7 Stator Windings Configuration

The shaft moves in a fixed repeatable increment, allowing precise positioning, with direction of rotation dictated by the stator poles.

12.2: INTERFACING TO A STEPPER MOTOR
stepper motors

- This stepper motor has four leads representing the four stator windings, and two commons for the center tapped leads.
- As the sequence of power is applied to each stator winding, the rotor will rotate.
  - Once started, sequence must continue in the proper order.

Table 12-5: Normal 4-Step Sequence

Clockwise	Step #	Winding A	Winding B	Winding C	Winding D	Counter-clockwise
↓	1	1	0	0	1	↑
	2	1	1	0	0	
	3	0	1	1	0	
	4	0	0	1	1	

12.2: INTERFACING TO A STEPPER MOTOR step angle

- Movement is associated with a single step depends on internal construction of the motor.
  - In particular the number of teeth on the stator & rotor.

The *step angle* is the minimum degree of rotation associated with a single step.

Various motors have different step angles.

Table 12-6: Stepper Motor Step Angles

Step Angle	Steps per Revolution
0.72	500
1.8	200
2.0	180
2.5	144
5.0	72
7.5	48
15	24

12.2: INTERFACING TO A STEPPER MOTOR step angle

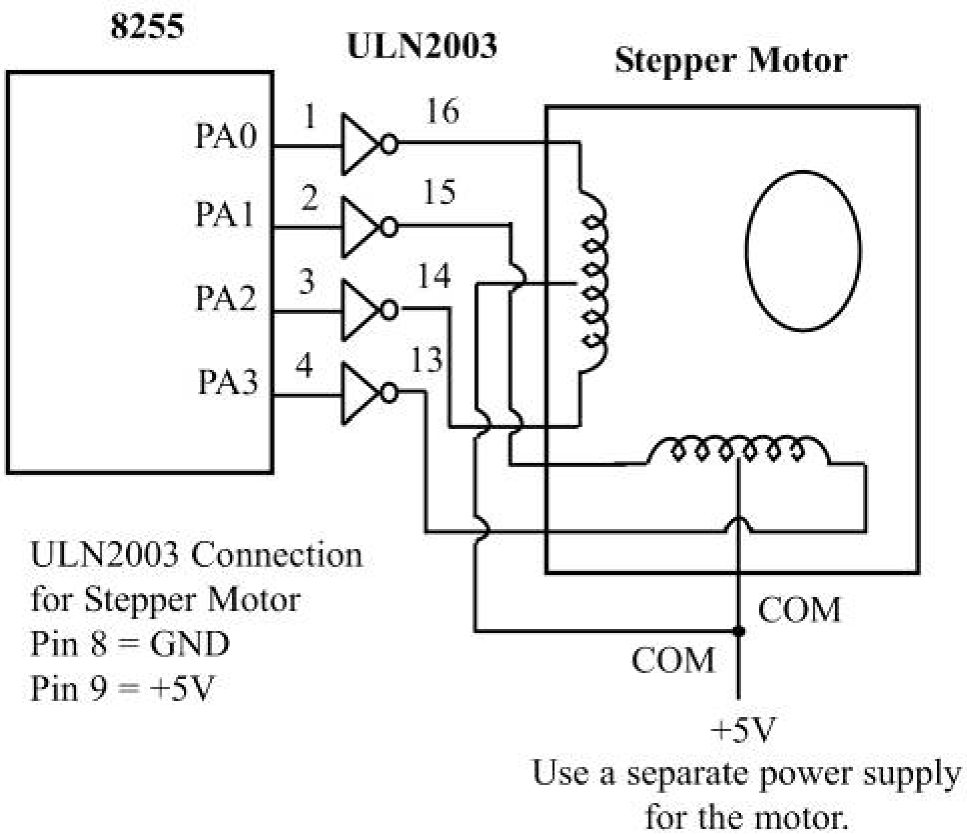
- Steps per revolution is the total number of steps needed to rotate one complete rotation, or 360 degrees (e.g., 180 steps x 2 degrees = 360).
- A stepper motor does not need to have more terminal leads for the stator to achieve smaller steps.
  - While some manufacturers have set aside only one lead for the common signal instead of two, they always have four leads for the stators.

12.2: INTERFACING TO A STEPPER MOTOR
stepper motor connection and programming

Fig. 12-8 8255 Connection to Stepper Motor

Example 12-2, on page 328, shows programming of the stepper motor as connected in Fig. 12-8.

This example contains some very important points on motor interfacing.



12.2: INTERFACING TO A STEPPER MOTOR steps per second and RPM relation

- The relationship between the RPM (revolutions per minute), steps per revolution, and steps per second is intuitive, and is as follows:

$$\text{Steps per second} = \frac{\text{RPM} \times \text{Steps per revolution}}{60}$$

12.2: INTERFACING TO A STEPPER MOTOR four-step sequence/number of rotor teeth

- The switching sequence in Table 12-5 is called the 4-step switching sequence since after four steps the same two windings will be "ON".
  - The minimum step angle is always a function of the number of teeth on the rotor.

Example 12-3

Give the number of times the 4-step sequence in Table 12-5 must be applied to a stepper motor to make an 80-degree move if the motor has a 2-degree step angle.

Solution:

A motor with a 2-degree step angle has the following characteristics:

Step angle:	2 degrees	Steps per revolution:	180
Number of rotor teeth:	45	Movement per 4-step sequence:	8 degrees

To move the rotor 80 degrees, we need to send 10 four-step sequences consecutively, since 10 × 4 steps × 2 degrees = 80 degrees.

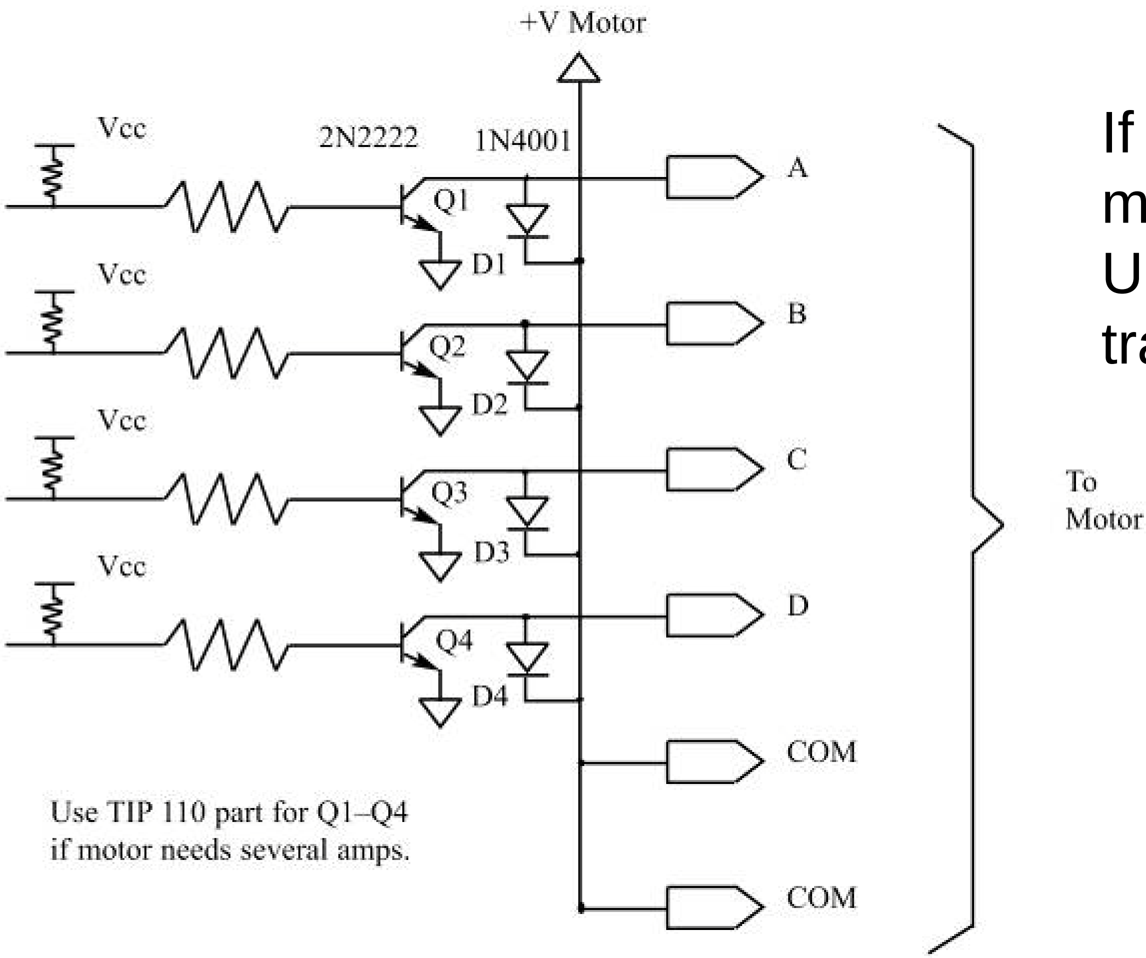
12.2: INTERFACING TO A STEPPER MOTOR
motor speed/holding torque/wave drive

- The motor speed, measured in steps per second (steps/s), is a function of the switching rate.
  - Torque is measured with rated voltage and current applied to the motor, in ounce-inch (or kg-cm) units.
- In addition to the 8- & 4-step sequences, there is also the *wave drive* 4-step sequence.

Table 12-8: Wave Drive 4-Step Sequence

Clockwise	Step #	Winding A	Winding B	Winding C	Winding D	Counter-clockwise
↓	1	1	0	0	0	↑
	2	0	1	0	0	
	3	0	0	1	0	
	4	0	0	0	1	

12.2: INTERFACING TO A STEPPER MOTOR
motor speed/holding torque/wave drive



If a given motor requires more current than the ULN2003 can provide, transistors can be used.

Fig. 12-9 Using Transistors for Stepper Motor Driver

12.3: INTERFACING TO A DAC
digital-to-analog (DAC) converter

- A DAC (digital-to-analog converter) is a widely device used to convert digital pulses to analog signals.
- The number of data bit inputs determines resolution of the DAC, as the number of analog output levels is equal to 2^n. (n is the number of data bit inputs)

12.3: INTERFACING TO A DAC
DAC 808

- In the DAC808, digital inputs are converted to current (I_{out}), a resistor connected to the I_{out} pin, converts the result to voltage.
  - The total current is a function of the binary numbers at the **D0–D7** inputs of the 1408 and the reference current (I_{ref}).
    - The I_{ref} current is generally set to 2.0 mA.

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

  - 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.

12.3: INTERFACING TO A DAC
DAC 808

Generation of current reference (setting I_{ref} = 2 mA) by using the standard 5-V power supply and 1K, 1.5K ohm standard resistors.

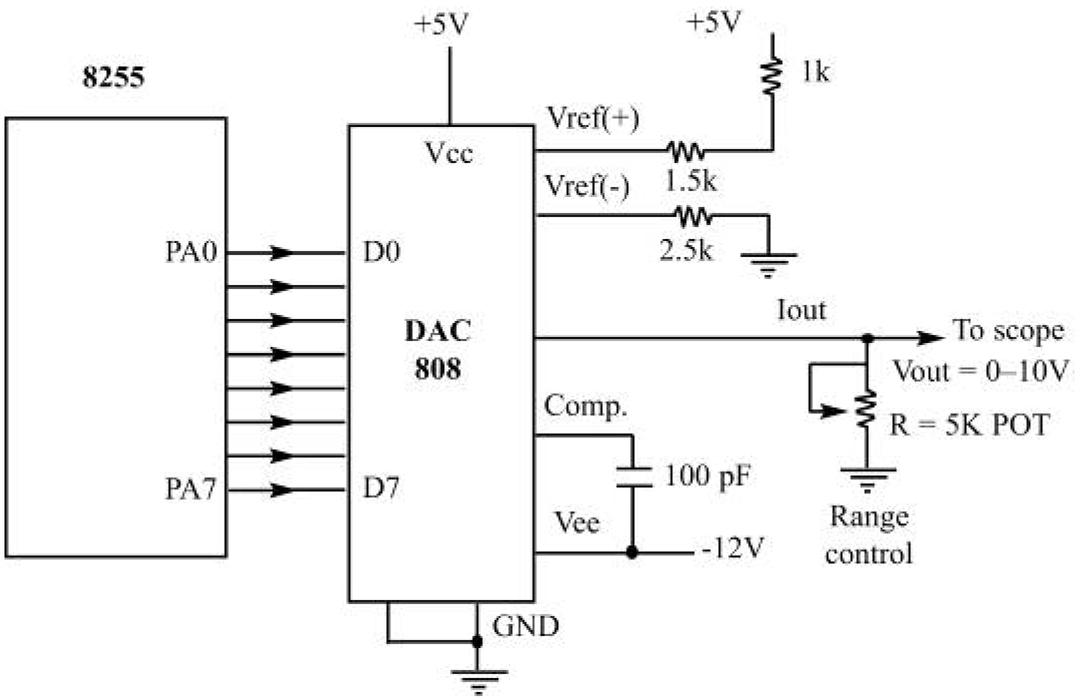


Figure 12-10 8255 Connection to DAC 88

Some also use a zener diode (LM336), which overcomes fluctuation associated with the power supply voltage.

12.3: INTERFACING TO A DAC
converting I_{out} to voltage in 1408 DAC

- Input resistance of the load where it is connected will also affect the output voltage.
  - The $\mathbf{I_{ref}}$ current output is isolated by connecting it to an op amp such as the 741, with Rf = 5 kilohms for the feedback resistor.

Example 12-5
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 (153/255) = 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

12.3: INTERFACING TO A DAC
generating a sine wave

- To generate a sine wave, requires a table whose values represent the magnitude of the sine of angles between 0 and 360 degrees.
  - Values for the sine function vary from -1.0 to +1.0.
    - Table values are integer numbers representing the voltage magnitude for the sine of *theta*.
  - This ensures that only integer numbers are output to the DAC by the x86 processor.
- Table 12-9 shows angles, sine values, voltage magnitude, and integer values representing the voltage magnitude for each angle.
  - With 30-degree increments.

12.3: INTERFACING TO A DAC
generating a sine wave – table 12-9

Table 12-9: Angle v. Voltage Magnitude for Sine Wave

Angle θ (degrees)	Sin θ	Vout (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	
300	–0.866	0.669	
330	–0.5	2.5	
360	0	5	

See the entire sine wave table on page 334 of your textbook.

12.4: INTERFACING TO ADC CHIPS & SENSORS ADC devices

- The physical world is *analog* (continuous).
  - Temperature, pressure (wind or liquid), humidity, and velocity are examples of physical quantities.
  - Digital computers use *binary* (discrete) values,
- A physical quantity is converted to electrical (voltage, current) signals using a device called a *transducer*. (Also referred to as sensors)
- Analog-to-digital converters are widely used devices for translating analog sensor signals to digital numbers so the PC can read them.

12.4: INTERFACING TO ADC CHIPS & SENSORS
ADC0848 chip

- The ADC0848 IC is an 8-bit resolution analog-to-digital converter from National Semiconductor Corp.

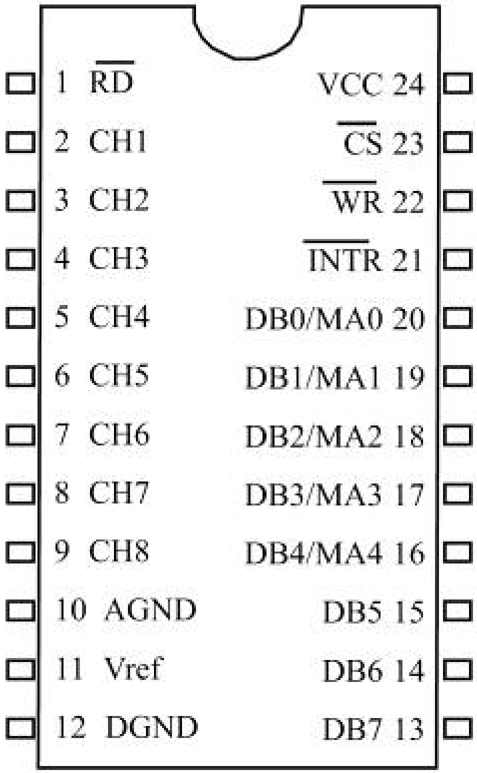


Figure 12-12 ADC0848 Chip

Chip data sheets can be found at *www.national.com*

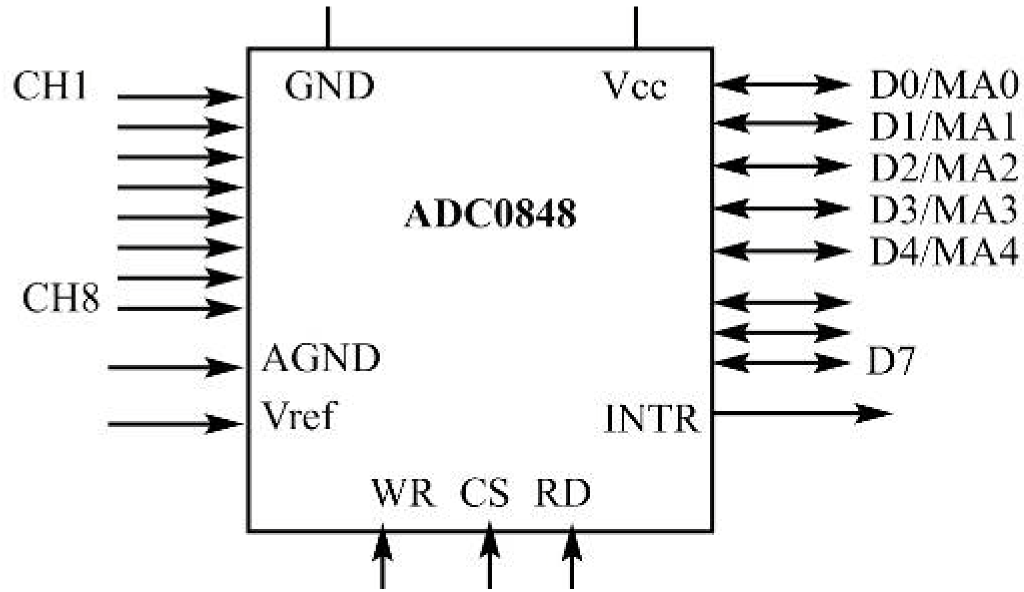


Figure 12-13 ADC0848 Block Diagram

ADC0848 chip pins

- **CS**, chip select - *active-low* input used to activate the 848 chip. To access 848, this pin must be *low*.
- **RD**, read - an *active-low* input signal.
  - **RD** gets converted data out of the 848 chip.
  - **RD** is referred to as output enable. (**OE**)
- $\mathbf{V_{ref}}$ - input voltage for reference voltage.
  - Voltage connected to this pin dictates the step size.

Table 12-10: ADC0848 V_{ref} vs. Step Size

V_{ref}(V)	Step size (mV)
5	19.53
4	15.62
2.56	10
1.26	5
0.64	2.5

Note: Step size = $V_{ref}/256$.

12.4: INTERFACING TO ADC CHIPS & SENSORS
ADC0848 chip pins

- **DB0–DB7** - digital data output pins.
  - Tri-state buffered, and converted data is accessed only when CS = 0, and a low pulse is applied to the **RD** pin.
- **MA0–MA4**, multiplexed address - ADC0848 uses multiplexed address/data pins to select the channel.
- **WR**, write - an input to the ADC0848 with two roles:
  - It latches the address of the selected channel present on the **D0–D7** pins.
  - It informs the ADC0848 to start the conversion of analog input at that channel.
- **VCC** - the +5 volt power supply.

12.4: INTERFACING TO ADC CHIPS & SENSORS
ADC0848 chip pins

- **CH1–CH8** - 8 channels of the $\mathbf{V_{in}}$ analog inputs.
  - In what *single-ended* mode, each channel can be used for analog $\mathbf{V_{in}}$ where **AGND** (analog ground) pin is used as a ground reference for all the not all at the same time since there is only a single channels.
  - In *differential mode*, two channels, such as CH1 and CH2, are paired together for the $\mathbf{V_{in}}$(+) and $\mathbf{V_{in}}$(–) differential analog inputs.

12.4: INTERFACING TO ADC CHIPS & SENSORS
ADC0848 chip pins

- **AGND**, **DGND** (analog ground, digital ground) - input pins providing the ground for both the analog and the digital signal.
  - Analog ground is connected to the ground of the analog $\mathbf{V_{in}}$.
  - Digital ground is connected to the ground of the **VCC** pin.

12.4: INTERFACING TO ADC CHIPS & SENSORS
ADC0848 chip pins

- **INTR**, interrupt - an *active-low* output pin.
  - Normally high, when conversion is finished, it goes low to signal the CPU that converted data is ready for pick up.

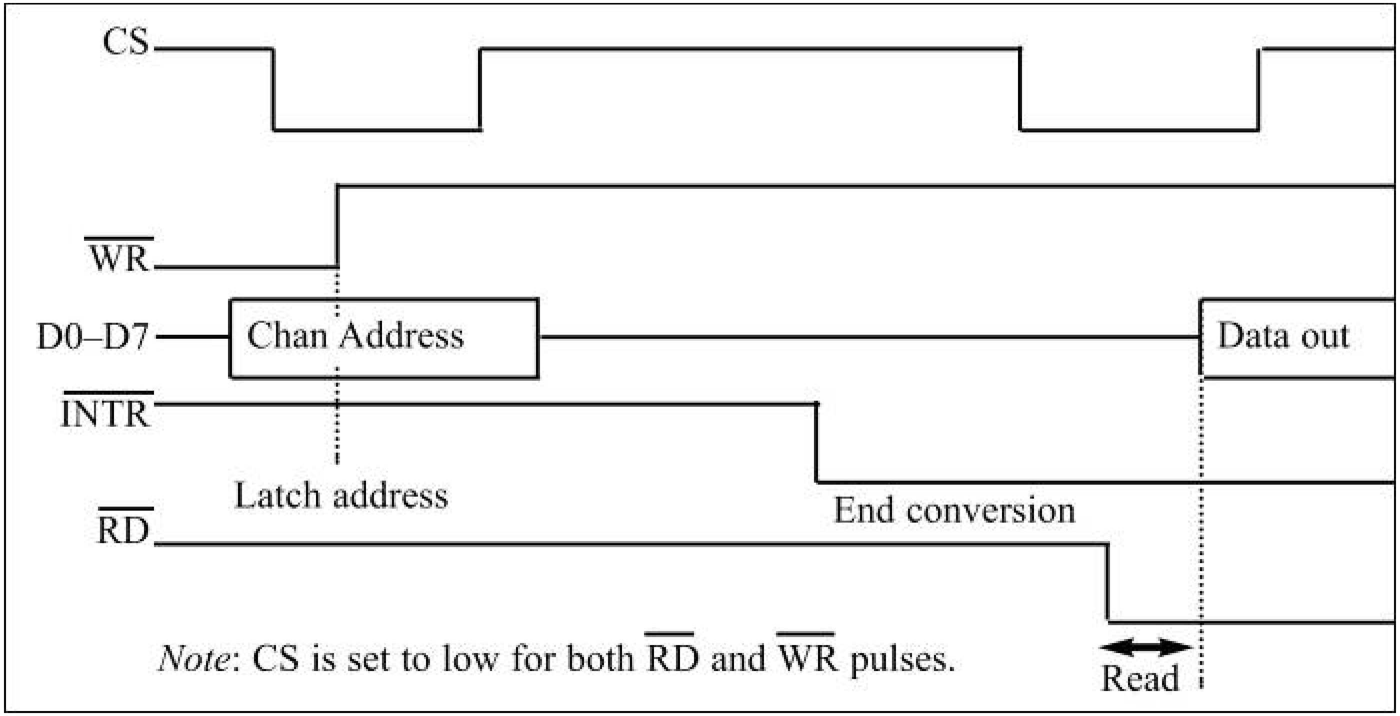


Figure 12-14 Selecting a Channel and Read Timing for ADC0848

12.4: INTERFACING TO ADC CHIPS & SENSORS
selecting an input channel

- Steps for data conversion by the ADC0848 chip:
  - 1. While **CS** = 0 and **RD** = 1, provide the address of the selected channel to the **DB0–DB7** pins
    - Apply a *low-to-high* pulse to the **WR** pin to latch in the address and start the conversion.
  - The channel's addresses are 08H for CH1, 09H for CH2, 0AH for CH3, and so on, as shown in Table 12-11.
  - 2. While **WR** = 1 and **RD** = 1, keep monitoring **INTR**.
    - When **INTR** goes low, the conversion is finished.
    - If **INTR** is high, keep polling until it goes low.
  - 3. After **INTR** has become low, make **CS** = 0, **WR** = 1, and apply a *low* pulse to **RD** to get the data from the 848.

12.4: INTERFACING TO ADC CHIPS & SENSORS
ADC0848 connection to 8255

- A summary of the connection between the 8255 and the ADC0848.

PA0–PA7 to D0–D7 of ADC:	Channel selection (out), data read (in)
PB0 to INTR	Port B as input
PC0 to WR	Port C as output
PC1 to RD	Port C as output

- Port **A** is an output when a channel is selected, and an input when we read the converted data.
- The **INTR** pin of the ADC must be monitored for end-of-conversion; configure **PB** as input.
  - Since both **WR** & **RD** are inputs into ADC, Port **C** is configured as an output port.

12.4: INTERFACING TO ADC CHIPS & SENSORS
interfacing a temperature sensor to PC

- Transducers convert physical quantities to electrical signals.
  - Depending on the transducer, the output produced is in the form of voltage, current, resistance, or capacitance.
- Temperature is converted to electrical signals using a *thermistor*, a transducer which responds to temperature change by changing its resistance.
  - Response is not linear, as shown.

Table 12-12: Thermistor Resistance vs. Temperature

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

12.4: INTERFACING TO ADC CHIPS & SENSORS
interfacing a temperature sensor to PC

- Complexity with writing software for such nonlinear devices has to the linear temperature sensor.
  - Including the LM34 and LM35 series from National Semiconductor Corp.

12.4: INTERFACING TO ADC CHIPS & SENSORS
LM34 and LM35 temperature sensors

- LM**34** series sensors are precision integrated-circuit temperature sensors with output voltage linearly proportional to the **Fahrenheit** temperature.
  - Requires no external calibration, as it is inherently calibrated, outputting 10 mV for each degree *Fahrenheit*.

Table 12-13: 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

LM34 and LM35 temperature sensors

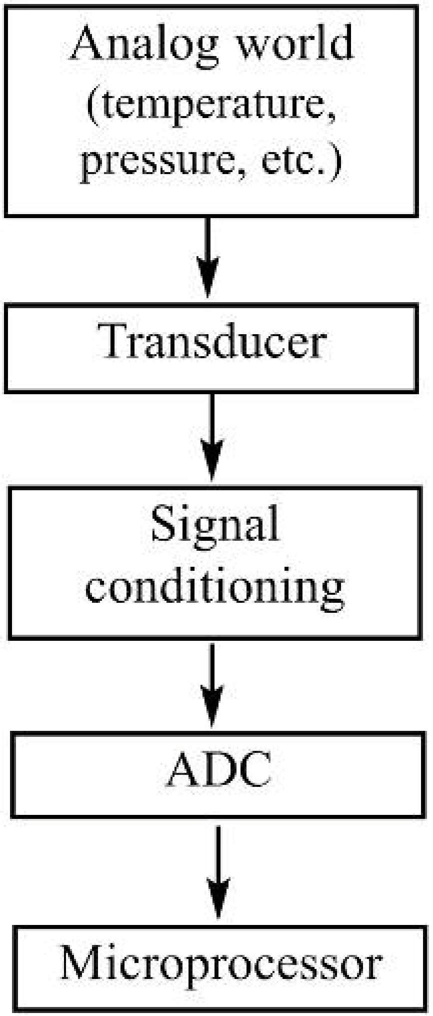
- LM**35** series sensors are precision integrated-circuit temperature sensors with output voltage linearly proportional to **Celsius** (centigrade) temperature.
  - Requires no external calibration, as it is inherently calibrated, outputting 10 mV for each degree *centigrade*.

Table 12-14: LM35 Temperature Sensor Series Selection Guide

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

12.4: INTERFACING TO ADC CHIPS & SENSORS
signal conditioning/interfacing LM35/PC

- Most common transducers produce output as voltage, current, charge, capacitance & resistance.
  - *Signal conditioning* is a widely used term for the conversion of these signals to voltage to send to an A-to-D converter.
    - Current-to-voltage conversion or a signal amplification.
- The change of resistance in a thermistor must be translated to voltage to be of use to an ADC.
  - As in connecting an LM35 to an ADC0848



12.4: INTERFACING TO ADC CHIPS & SENSORS
signal conditioning/interfacing LM35/PC

- Connection of the temperature sensor to CH2 of the ADC0848.

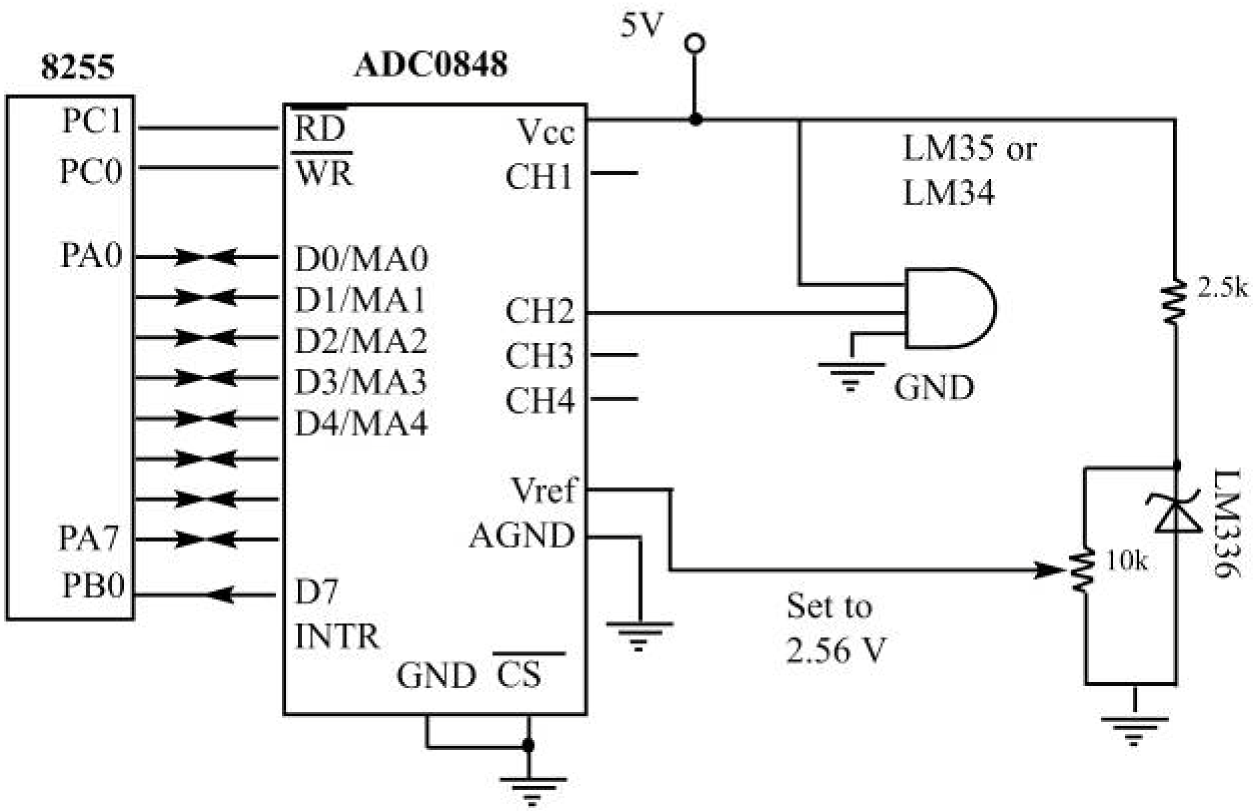


Figure 12-17 8255 Connection to ADC0848 and Temperature Sensor

The LM336-2.5 zener diode is used to fix the voltage across the 10k POT at 2.5 volts.

The LM336-2.5 should overcome fluctuations in the power supply.

12.4: INTERFACING TO ADC CHIPS & SENSORS signal conditioning/interfacing LM35/PC

- Connection of the temperature sensor to CH2 of the ADC0848.

Table 12-15: Temperature v. Vout of the ADC0848

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

12.4: INTERFACING TO ADC CHIPS & SENSORS
ADC808/809

- ADC808/809 has eight input channels.
  - To convert 8 different analog inputs.
    - An 8-bit ADC.

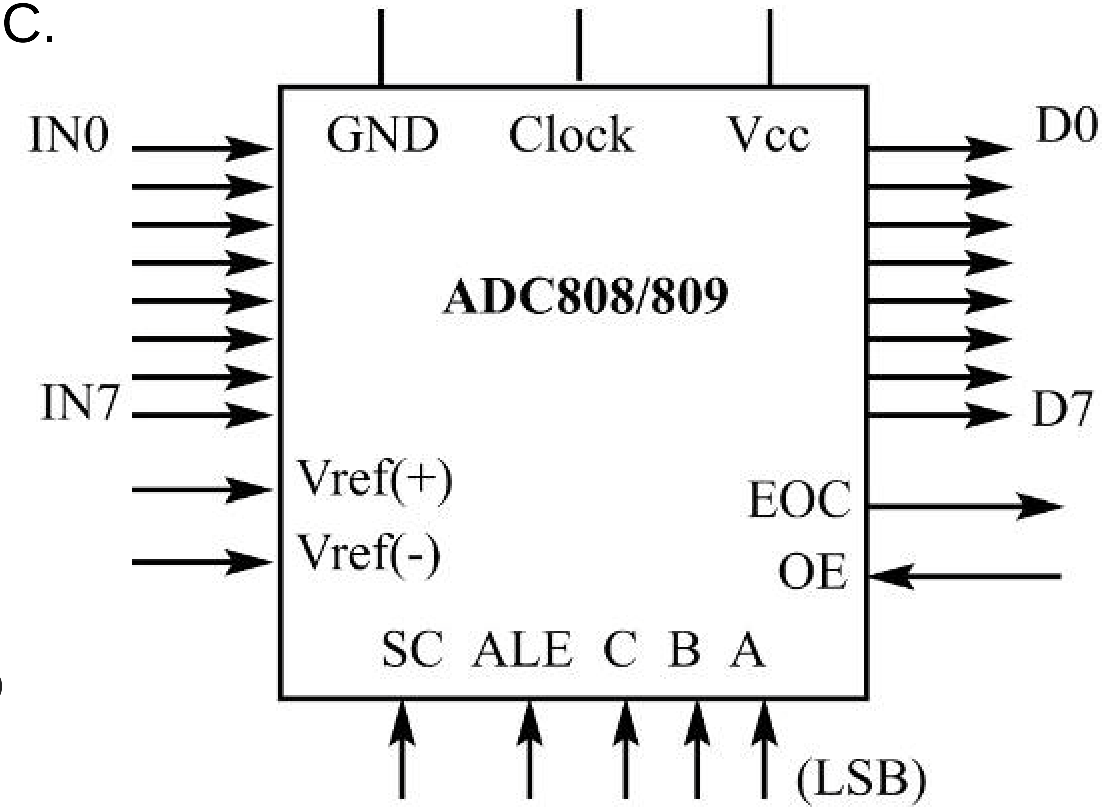


Figure 12-18 ADC 808/809

12.4: INTERFACING TO ADC CHIPS & SENSORS
ADC808/809 pins

- **OE**, output enable - an *active-high* input signal.
  - ADC converts analog input to binary equivalent and holds it in an internal register.
    - **OE** is used to get the converted data out of the ADC808 chip.
- **SC**, start conversion - an input pin used to inform ADC808 to start the conversion process.
  - The amount of time it takes to convert varies depending on the **CLK** value.
- **CLK** - is an input pin, connected to an external clock source.
  - ADC0848 uses an internal clock; ADC808, external.

12.4: INTERFACING TO ADC CHIPS & SENSORS
ADC808/809 pins

- **EOC**, end of conversion - an *active-low* output pin.
  - Normally high, after **EOC** goes low, a low-to-high pulse is sent to the **OE** pin to get the data out of the ADC808.
- $\mathbf{V_{ref}}$(+) & $\mathbf{V_{ref}}$(–) - input voltages for reference voltage, which dictates the step size.
- **D0–D7** - tri-state buffered digital data output pins.
  - Converted data is accessed only when OE is forced *high*.
- **IN0–IN7** - 8 channels of the $\mathbf{V_{in}}$ analog inputs.
  - Allows the read of 8 different analog signals.
    - Not all at the same time, as there is only a single D0–D7.

12.4: INTERFACING TO ADC CHIPS & SENSORS
ADC808/809 pins

- **A**, **B**, **C**, and **ALE** - input signals to ADC808/809.
  - The channel is selected according to Table 12-16.

To select a channel, provide the channel address to the A, B, and C pins according to Table 12-16.

Apply an *L-to-H* pulse to the ALE pin to latch in the address.

Table 12-16: ADC808/809 Analog Channel Selection

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

12.4: INTERFACING TO ADC CHIPS & SENSORS
how to read ADC808/809 data

- ADC808/809 has a clock pin, which requires an external clock source.
  - Conversion speed varies according to the speed of the external clock source.
  - If you use a time delay to wait before you read the data, the size of the delay varies depending on the speed of the clock connected to the clock pin.

12.4: INTERFACING TO ADC CHIPS & SENSORS
how to read ADC808/809 data - steps

- Steps to select a channel & read ADC808/809 data:
  - 1. Provide the channel address to pins **A**, **B**, and **C**. (see Table 12-16)
  - 2. Apply an *L-to-H* pulse to **ALE** to latch in the channel address.
  - 3. Apply an *L-to-H* pulse to **SC** pin to start the conversion of analog input to digital data.
  - 4. After 8 clocks, **EOC** will go *low* to indicate the data is converted and ready to be picked up.
    - Use a small time delay, or monitor the EOC pin, then read the data out after it goes *low*.
  - 5. Apply an *L-to-H* pulse to **OE** pin & read the data.

Dec Hex Bin
12 C 00001100

ENDS ; TWELVE

The x86 PC

assembly language, design, and interfacing

fifth edition

MUHAMMAD ALI MAZIDI
JANICE GILLISPIE MAZIDI
DANNY CAUSEY

The x86 PC
assembly language, design, and interfacing
fifth edition

Prentice Hall