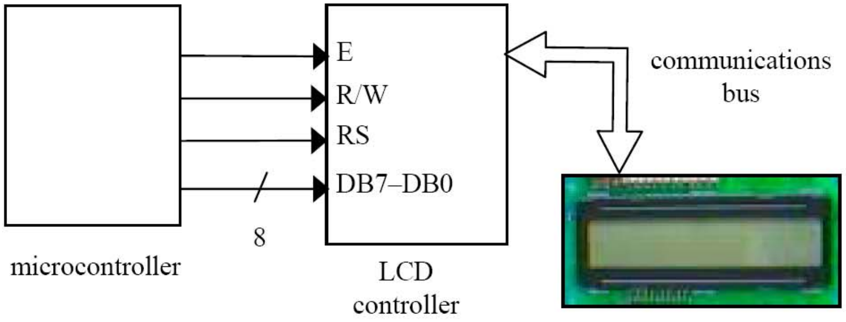
Chương 5
Thiết kế hệ vi xử lý

5.6 Giao tiếp bộ hiển thị (Display)
5.6.2 Giao tiếp với LCD

LCD controller



```
void WriteChar(char c){

  RS = 1;                    /* indicate data being sent */
  DATA_BUS = c;              /* send data to LCD */
  EnableLCD(45);             /* toggle the LCD with appropriate delay */
}
```

CODES	
I/D = 1 cursor moves left	DL = 1 8-bit
I/D = 0 cursor moves right	DL = 0 4-bit
S = 1 with display shift	N = 1 2 rows
S/C =1 display shift	N = 0 1 row
S/C = 0 cursor movement	F = 1 5x10 dots
R/L = 1 shift to right	F = 0 5x7 dots
R/L = 0 shift to left	

RS	R/W	DB$_7$	DB$_6$	DB$_5$	DB$_4$	DB$_3$	DB$_2$	DB$_1$	DB$_0$	Description
0	0	0	0	0	0	0	0	0	1	Clears all display, return cursor home
0	0	0	0	0	0	0	0	1	*	Returns cursor home
0	0	0	0	0	0	0	1	I/D	S	Sets cursor move direction and/or specifies not to shift display
0	0	0	0	0	0	1	D	C	B	ON/OFF of all display(D), cursor ON/OFF (C), and blink position (B)
0	0	0	0	0	1	S/C	R/L	*	*	Move cursor and shifts display
0	0	0	0	1	DL	N	F	*	*	Sets interface data length, number of display lines, and character font
1	0	WRITE DATA								Writes Data

LCD Operation

LCD is gaining popular and replacing LEDs (7-segment …), due to

1. declining price
2. the ability to display numbers, characters, and graphics
3. relieving the CPU task by incorporating a refreshing controller
4. ease of programming for characters and graphics (OLED is the coming display)

LCD Pin Descriptions

14-pin LCD module is discussed here, table 12-1 lists pin’s function, Fig 12-1 shows the pin positions for various LCDs

– Vcc, Vss provide +5V and ground

– Vee is used for contrast controlling

– RS (register select) is used to select the instruction command code register (RS = 0) or data register (RS = 1)

– LCD command codes is listed at table 12-2

– R/W (read/write) allows user to write to (R/W = 0) or read from (R/W = 1) information

– E (enable) latch information at data pins; when data is supplied to data pins, a high-to-low pulse must be applied to this pin

– D0-D7 are the 8-bit data pins; send information to LCD (R/W = 0) and read contents of LCD internal registers (R/W = 1)

– to display letters and numbers, ASCII codes are sent while RS = 1

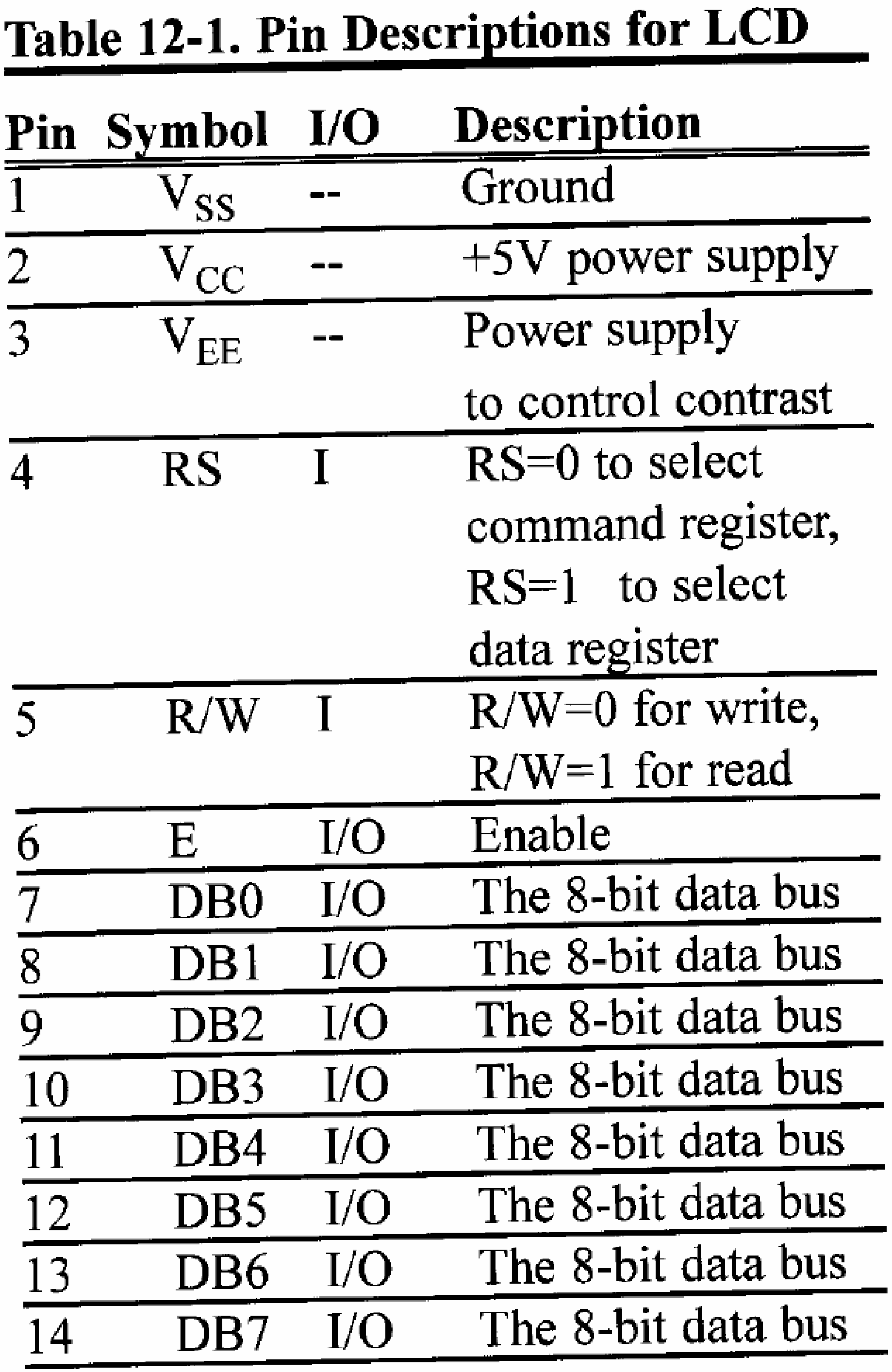


Table 12-1. Pin Descriptions for LCD

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

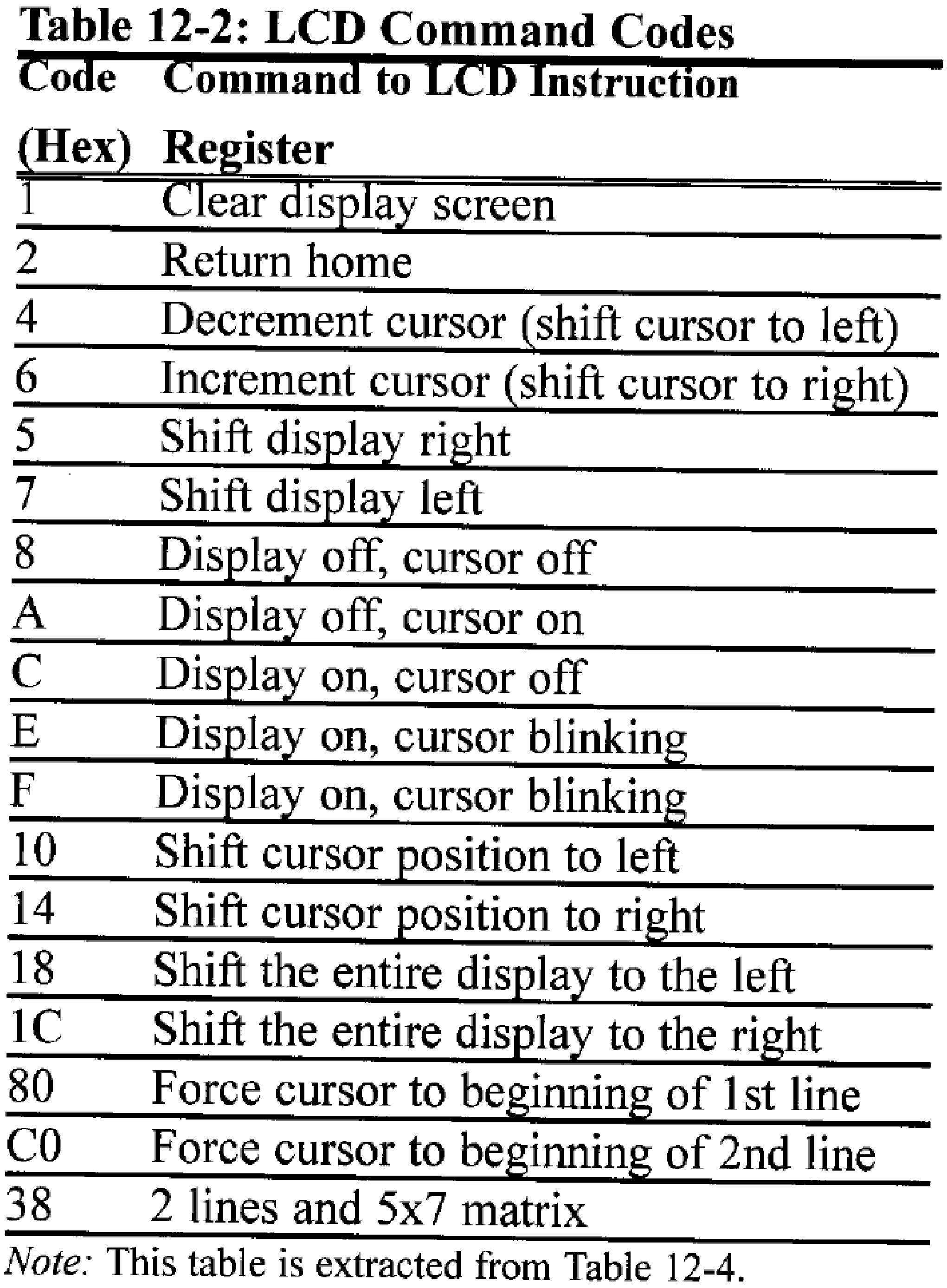
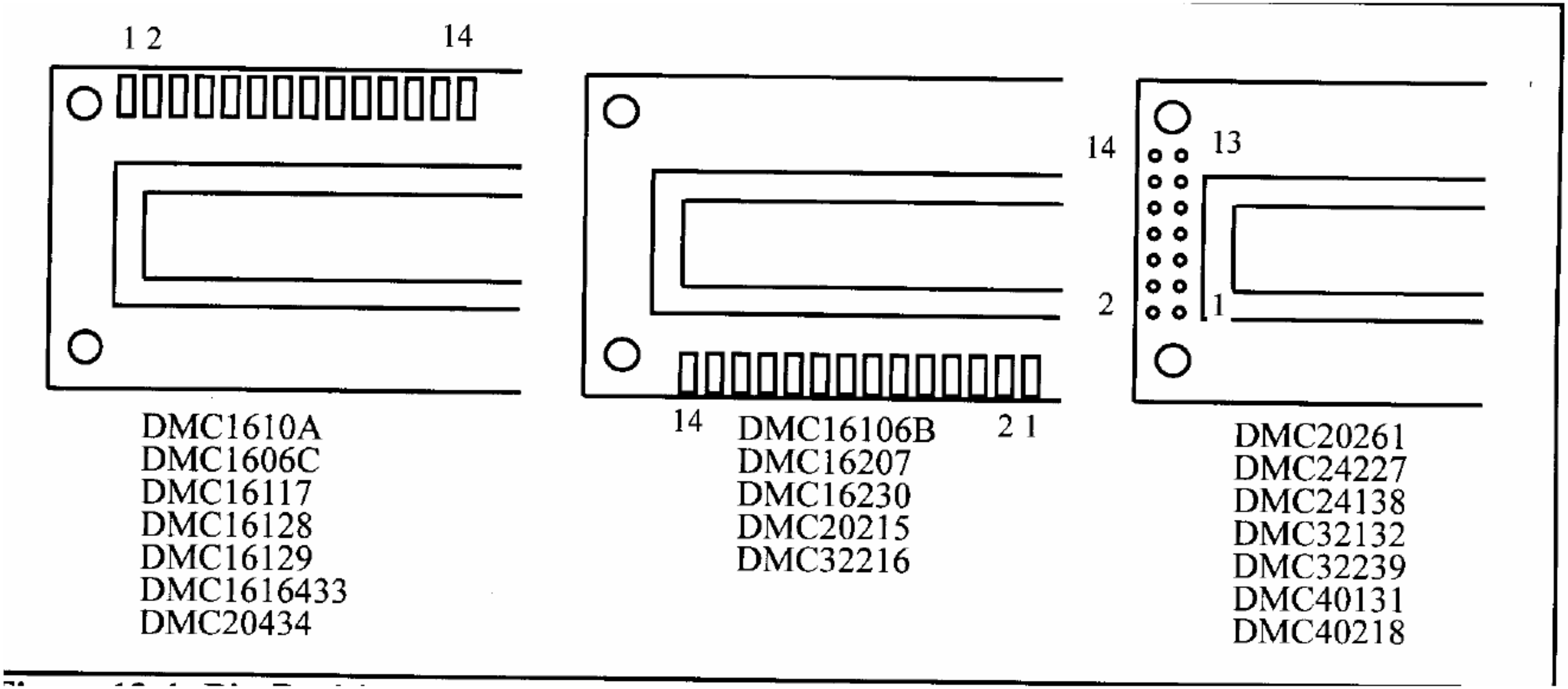


Table 12-2: LCD Command Codes

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 blinking
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 5x7 matrix

Note: This table is extracted from Table 12-4.

Pin diagrams

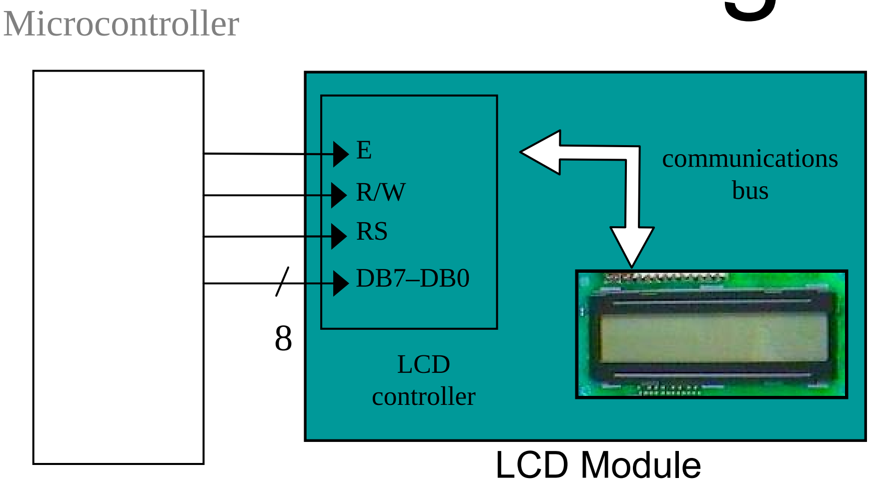


– RS = 0, the command code register is selected, we can send instruction to LCD to perform clear, shift, blink …
– when RS = 0, and R/W = 1, D7 is busy flag, when D7 = 0, LCD is ready to receive new information; it is recommended to check the busy flag before writing any data to the LCD

LCD Interfacing

- Liquid Crystal Displays (LCDs) have become a cheap and easy way to display text for an embedded system
  - Various configurations (1 line by 20 characters upto 8 lines by 80 characters).
- LCD needs a driving circuit to work.
- Driving circuit and LCD are often integrated into a single chip Hitachi LM015 can display one line of 16 characters
- The display has one register into which commands are sent and one register into which data to be displayed are sent
- Two registers are differentiated by the RS input
- Data lines (DB7-DB0) are used to transfer both commands (clearing, cursor positioning, etc) and data (character to be displayed)

Alphanumeric LCD Interfacing



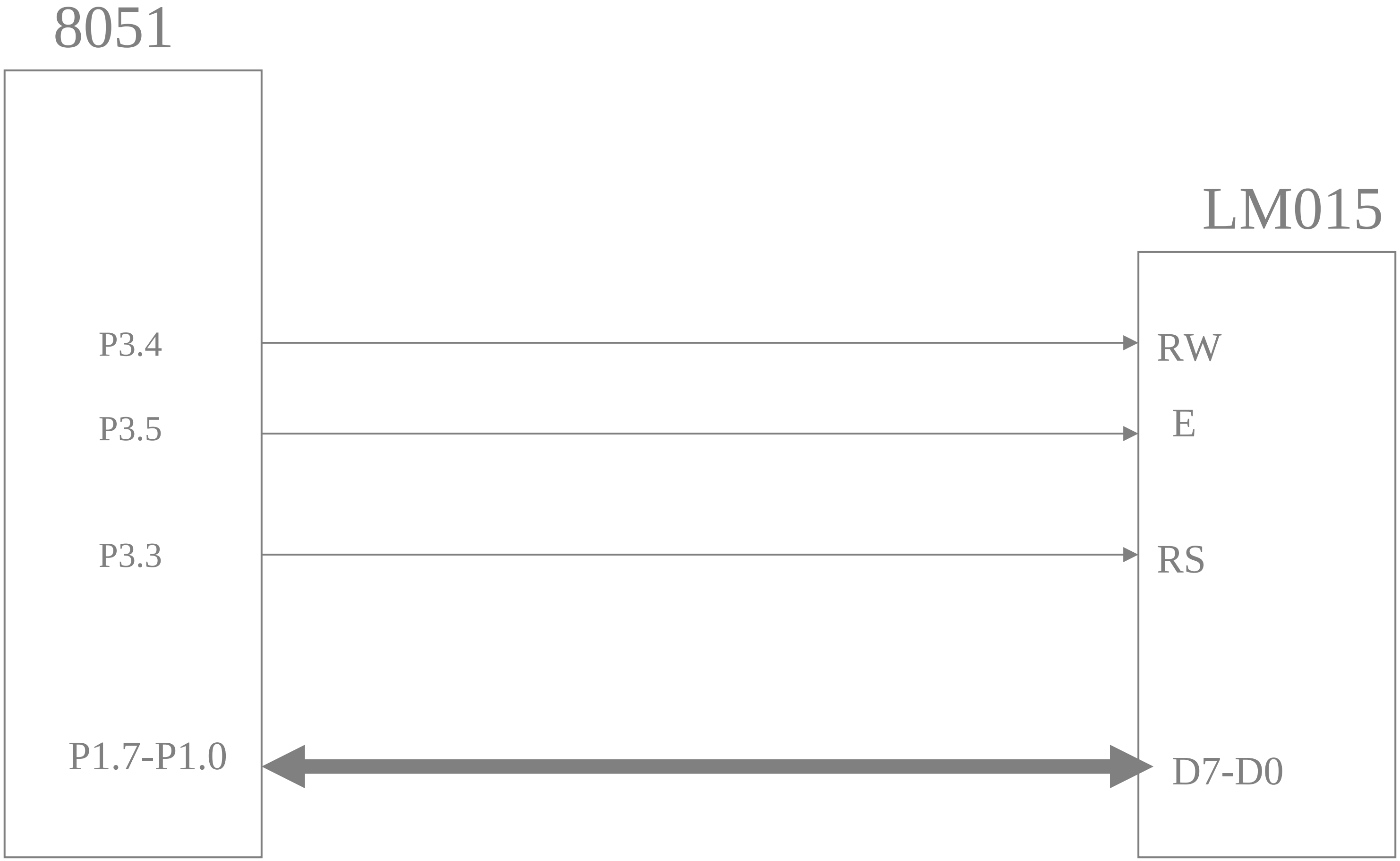
- **Pinout**
  - 8 data pins D7:D0
  - RS: Data or Command Register Select
  - R/W: Read or Write
  - E: Enable (Latch data)
- **RS** – Register Select
  - RS = 0 → Command Register
  - RS = 1 → Data Register
- **R/W** = 0 → Write, R/W = 1 → Read
- **E** – Enable
  - Used to latch the data present on the data pins.
- **D0 – D7**
  - Bi-directional data/command pins.
  - Alphanumeric characters are sent in ASCII format.

LCD Commands

- The LCD’s internal controller can accept several commands and modify the display accordingly. These commands would be things like:
  - Clear screen
  - Return home
  - Decrement/Increment cursor

- After writing to the LCD, it takes some time for it to complete its internal operations. During this time, it will not accept any new commands or data.
  - We need to insert time delay between any two commands or data sent to LCD

Interfacing LCD with 8051



Interfacing LCD with 8051

In main program:

```
. . .
MOV    A, COMMAND
CALL    CMD
CALL DELAY
MOV    A, ANOTHER_CMD
CALL    CMD
CALL DELAY
MOV    A, #’A’
CALL   DATA
CALL DELAY
MOV    A, #’B’
CALL   DATA
CALL  DELAY ….
```

Command and Data Write Routines

```
DATA:  MOV    P1, A  ; A is ascii data
       SETB   P3.3   ; RS=1 data
       CLR    P3.4   ; RW=0 for write
       SETB   P3.5   ; H->L pulse on E
       CLR    P3.5
       RET

CMD:   MOV    P1, A   ; A has the cmd word
       CLR    P3.3    ; RS=0 for cmd
       CLR    P3.4    ; RW=0 for write
       SETB   P3.5    ; H->L pulse on E
       CLR    P3.5
       RET
```

Sending code or data to the LCD with checking busy flag

The above code showed how to send commands to the LCD without checking the busy flag. Notice that we must put a long delay in between issuing data or commands to the LCD. However, a much better way is to monitor the busy flag before issuing a command or data to the LCD. This is shown below.

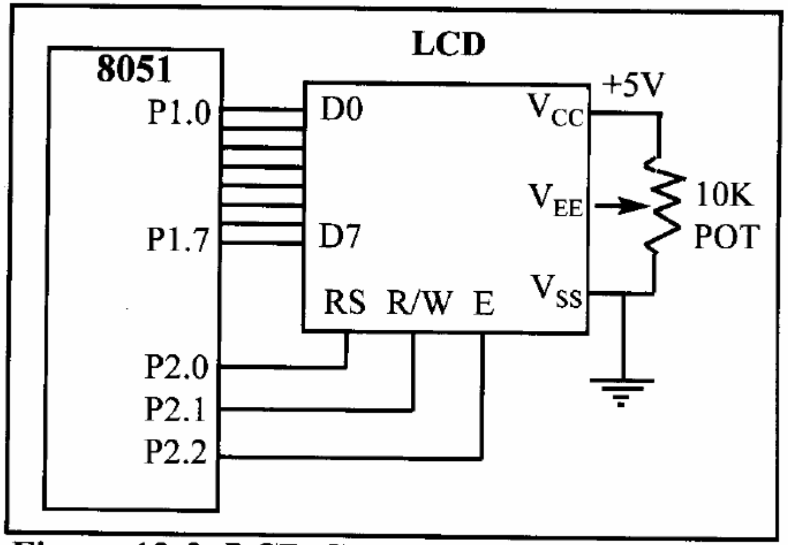
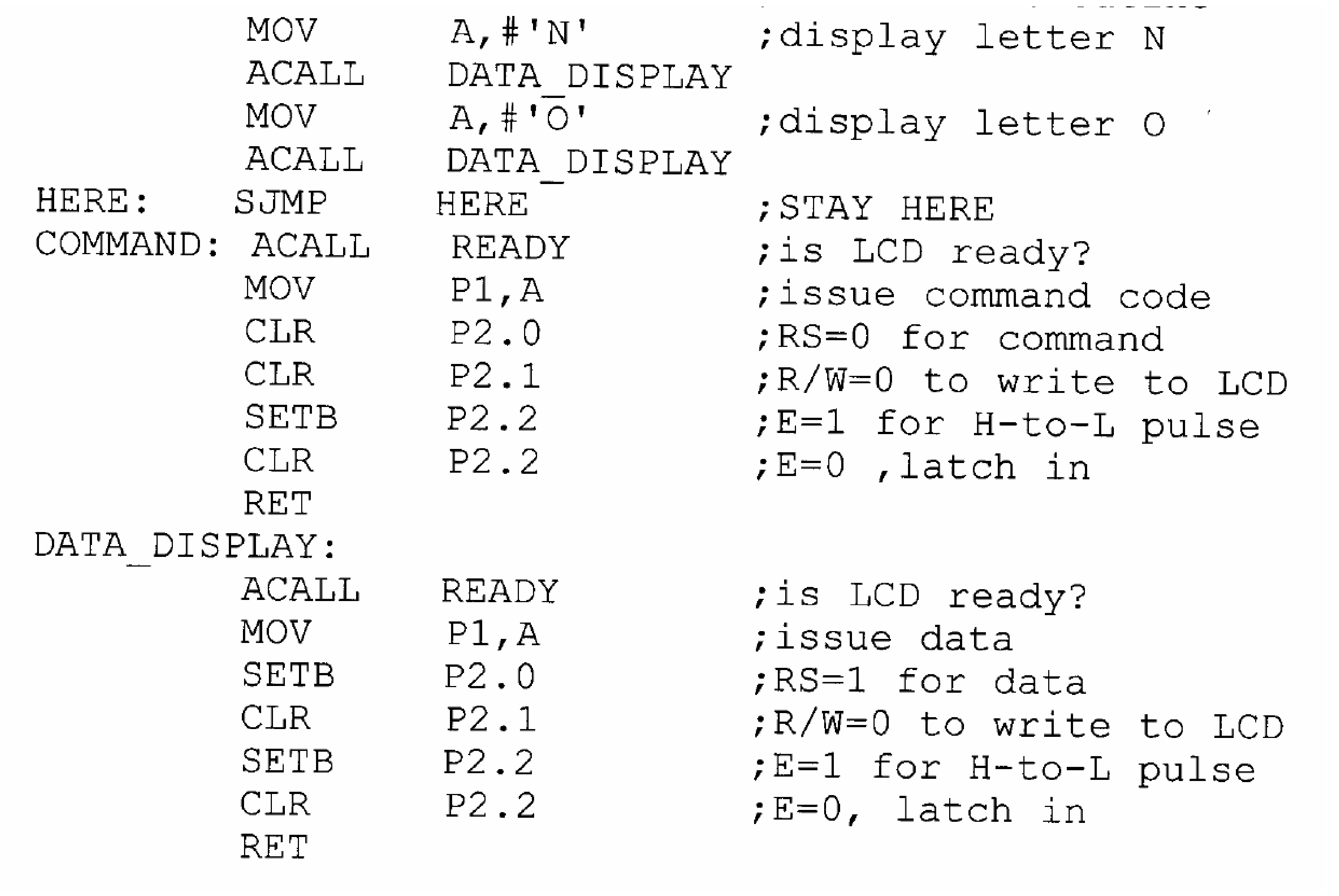
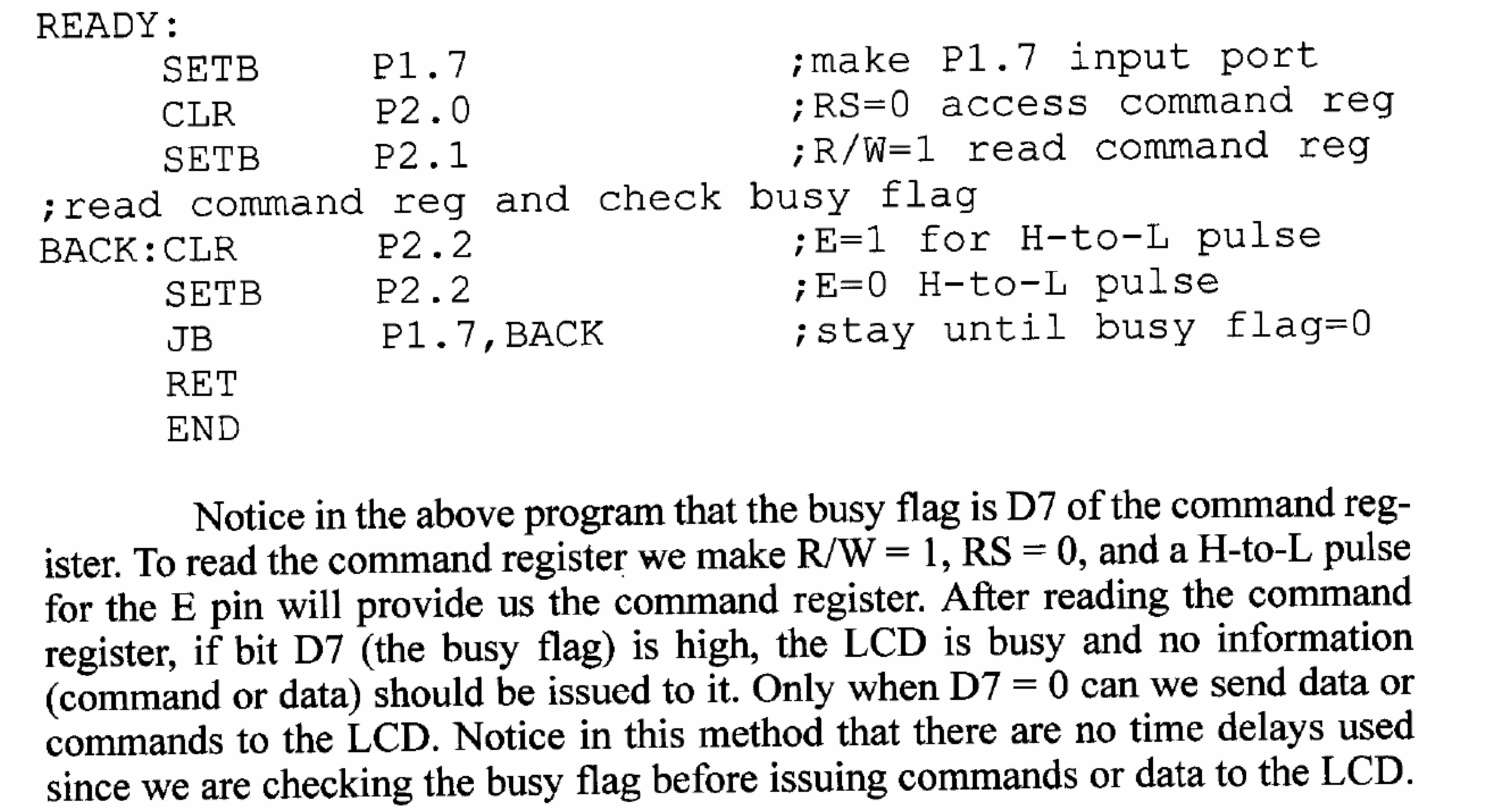


Figure 12-2. LCD Connection

```
;Check busy flag before sending data, command to LCD
;P1=data pin
;P2.0 connected to RS pin
;P2.1 connected to R/W pin
;P2.2 connected to E pin
        ORG
        MOV     A,#38H ;init. LCD 2 lines,5x7 matrix
        ACALL   COMMAND     ;issue command
        MOV     A,#0EH      ;LCD on, cursor on
        ACALL   COMMAND     ;issue command
        MOV     A,#01H      ;clear LCD command
        ACALL   COMMAND     ;issue command
        MOV     A,#06H      ;shift cursor right
        ACALL   COMMAND     ;issue command
        MOV     A,#86H      ;cursor: line 1, pos. 6
        ACALL   COMMAND     ;command subroutine
```



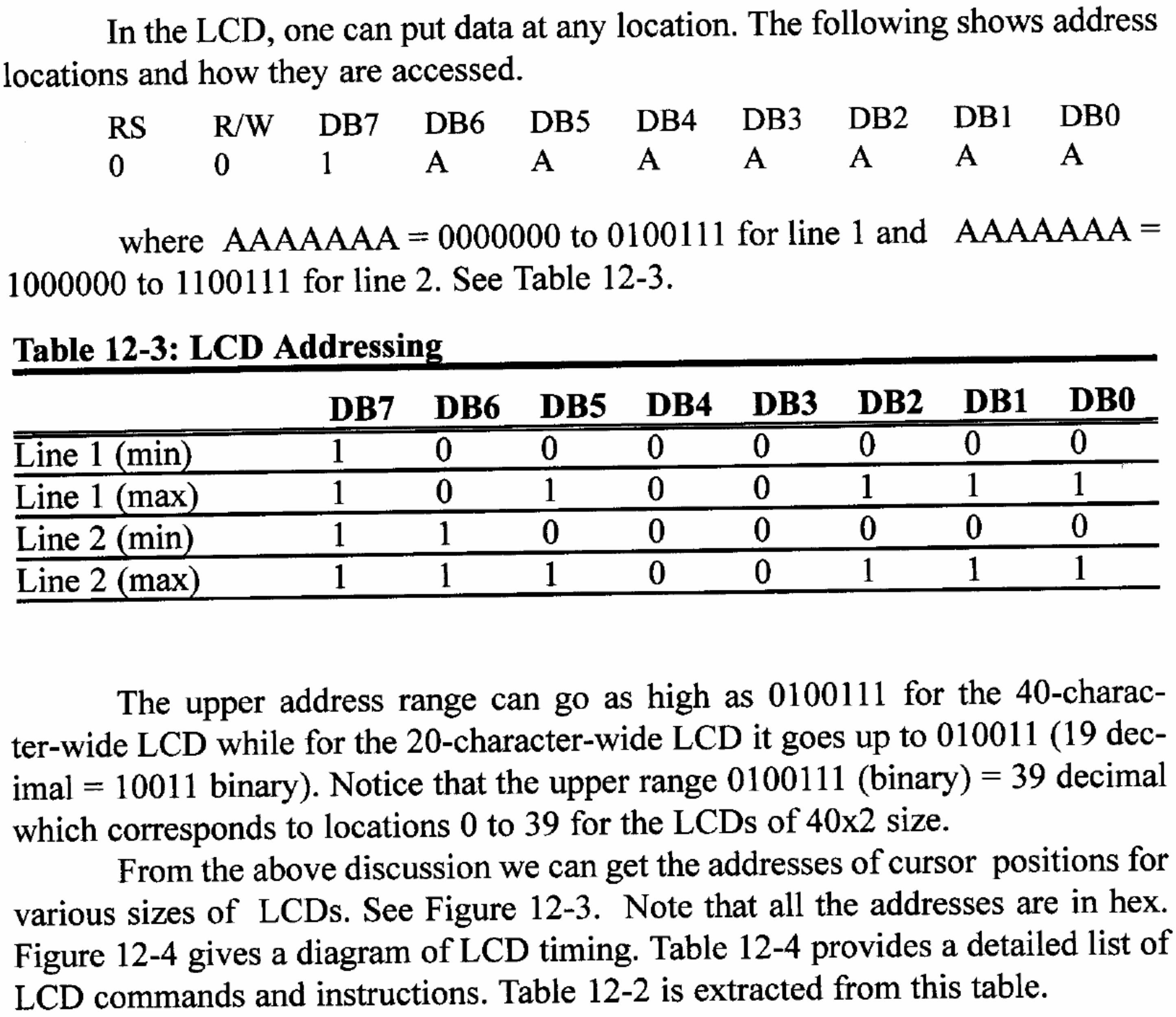
```
          MOV      A,#'N'          ;display letter N
          ACALL    DATA_DISPLAY
          MOV      A,#'O'          ;display letter O
          ACALL    DATA_DISPLAY
HERE:     SJMP     HERE            ;STAY HERE
COMMAND:  ACALL    READY           ;is LCD ready?
          MOV      P1,A            ;issue command code
          CLR      P2.0            ;RS=0 for command
          CLR      P2.1            ;R/W=0 to write to LCD
          SETB     P2.2            ;E=1 for H-to-L pulse
          CLR      P2.2            ;E=0 ,latch in
          RET
DATA_DISPLAY:
          ACALL    READY           ;is LCD ready?
          MOV      P1,A            ;issue data
          SETB     P2.0            ;RS=1 for data
          CLR      P2.1            ;R/W=0 to write to LCD
          SETB     P2.2            ;E=1 for H-to-L pulse
          CLR      P2.2            ;E=0, latch in
          RET
```



```
READY:
      SETB     P1.7              ;make P1.7 input port
      CLR      P2.0              ;RS=0 access command reg
      SETB     P2.1              ;R/W=1 read command reg
;read command reg and check busy flag
BACK:CLR       P2.2              ;E=1 for H-to-L pulse
      SETB     P2.2              ;E=0 H-to-L pulse
      JB       P1.7,BACK         ;stay until busy flag=0
      RET
      END
```

Notice in the above program that the busy flag is D7 of the command register. To read the command register we make R/W = 1, RS = 0, and a H-to-L pulse for the E pin will provide us the command register. After reading the command register, if bit D7 (the busy flag) is high, the LCD is busy and no information (command or data) should be issued to it. Only when D7 = 0 can we send data or commands to the LCD. Notice in this method that there are no time delays used since we are checking the busy flag before issuing commands or data to the LCD.

LCD data sheet



In the LCD, one can put data at any location. The following shows address locations and how they are accessed.

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

where AAAAAAA = 0000000 to 0100111 for line 1 and AAAAAAA = 1000000 to 1100111 for line 2. See Table 12-3.

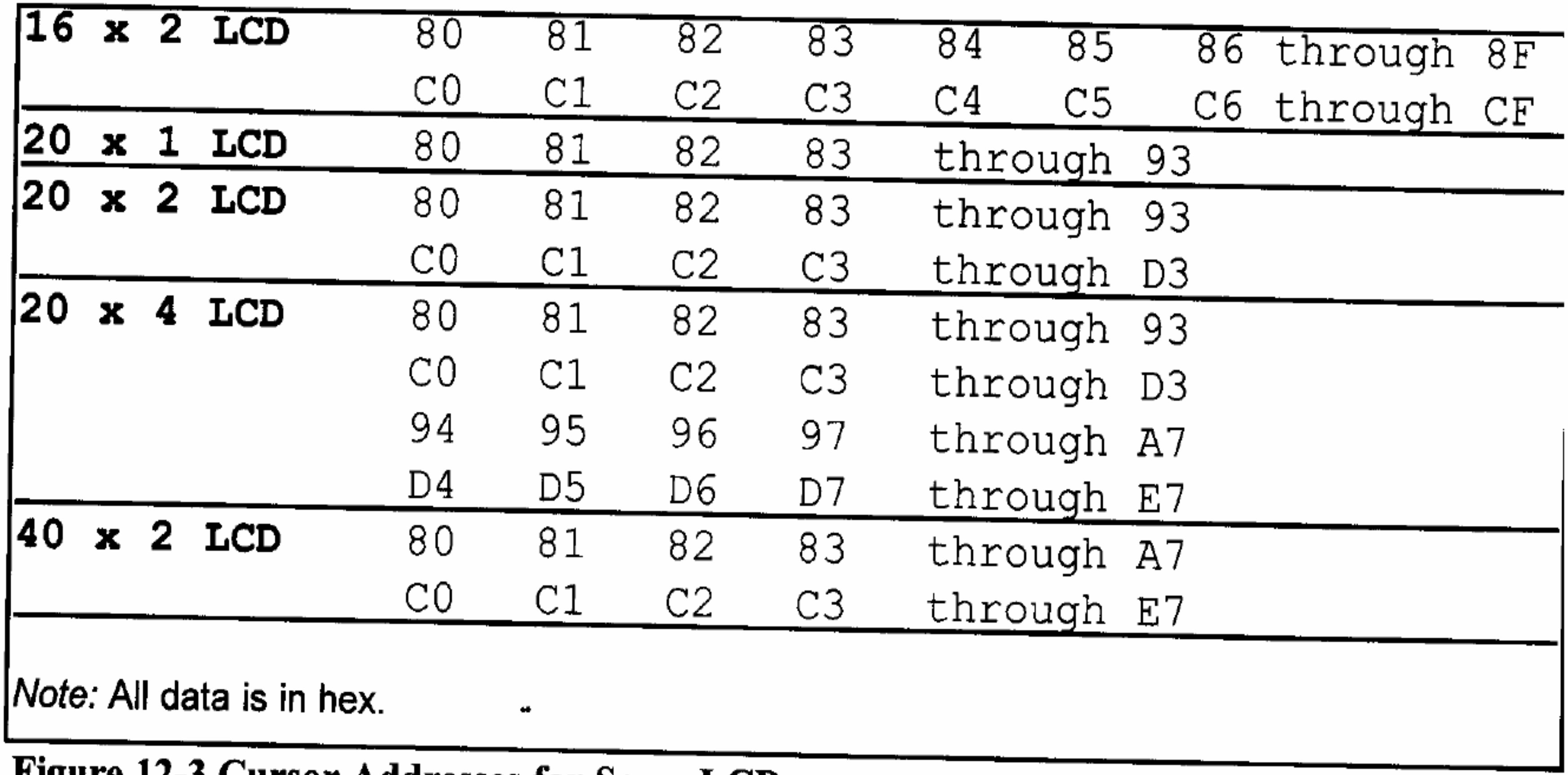
Table 12-3: LCD Addressing

	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

The upper address range can go as high as 0100111 for the 40-character-wide LCD while for the 20-character-wide LCD it goes up to 010011 (19 decimal = 10011 binary). Notice that the upper range 0100111 (binary) = 39 decimal which corresponds to locations 0 to 39 for the LCDs of 40x2 size.

From the above discussion we can get the addresses of cursor positions for various sizes of LCDs. See Figure 12-3. Note that all the addresses are in hex. Figure 12-4 gives a diagram of LCD timing. Table 12-4 provides a detailed list of LCD commands and instructions. Table 12-2 is extracted from this table.

LCD

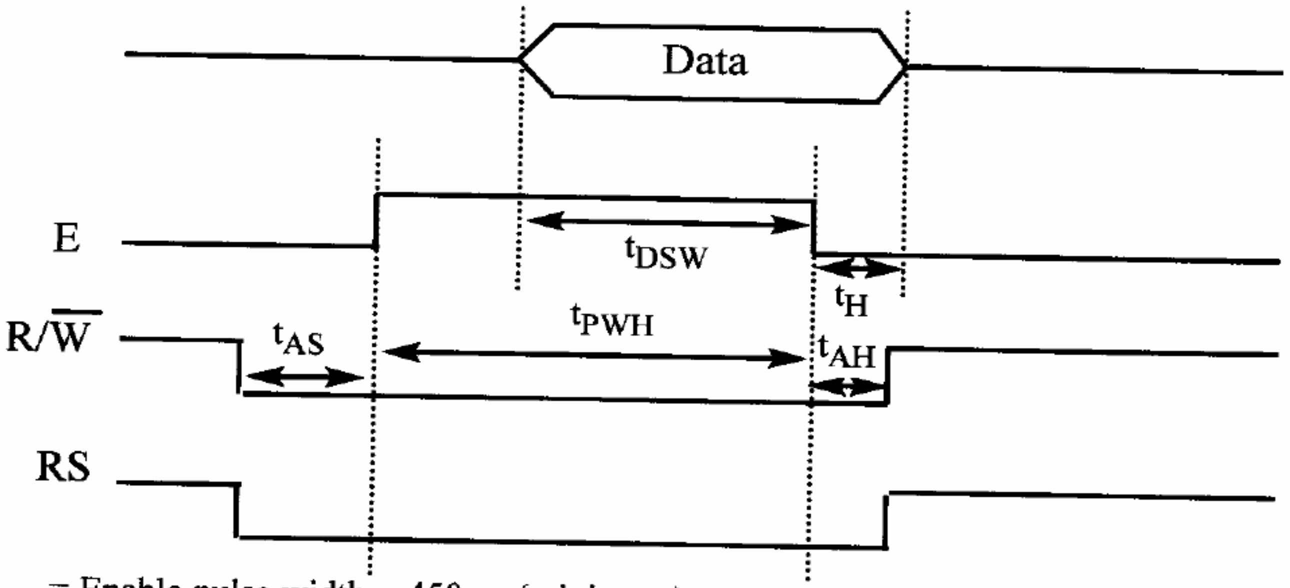


16 x 2 LCD	80	81	82	83	84	85	86 through 8F
	C0	C1	C2	C3	C4	C5	C6 through CF
20 x 1 LCD	80	81	82	83	through 93		
20 x 2 LCD	80	81	82	83	through 93		
	C0	C1	C2	C3	through D3		
20 x 4 LCD	80	81	82	83	through 93		
	C0	C1	C2	C3	through D3		
	94	95	96	97	through A7		
	D4	D5	D6	D7	through E7		
40 x 2 LCD	80	81	82	83	through A7		
	C0	C1	C2	C3	through E7		

Note: All data is in hex.

Figure 12-3 Cursor Addresses for Some LCDs

LCD Timing



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

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

t_H = Data hold time = 10 ns (minimum)

t_{AS} = Set up 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 Timing

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 display and sets DD RAM address 0 in address counter	1.64 ms
Return Home	0	0	0	0	0	0	0	0	1	-	Sets DD RAM address 0 as address counter. Also returns display being shifted to original position. DD RAM contents remain unchanged.	1.64 ms
Entry Mode Set	0	0	0	0	0	0	0	1	1/D	S	Sets cursor move direction and specifies shift of display. These operations are performed during data write and read.	40 µs
Display On/ Off Control	0	0	0	0	0	0	1	D	C	B	Sets On/Off of entire display (D), cursor On/Off (C), and blink of cursor position character (B).	40 µs
Cursor or Display Shift	0	0	0	0	0	1	S/C	R/L	-	-	Moves cursor and shifts display without changing DD RAM contents.	40 µs
Function Set	0	0	0	0	1	DL	N	F	-	-	Sets interface data length (DL), number of display lines (L) and character font (F).	40 µs

Set CG RAM Address	0 0 0 1	AGC	Sets CG RAM address. CG RAM data is sent and received after this setting.	40 µs
Set DD RAM Address	0 0 1	ADD	Sets DD RAM address. DD RAM data is sent and received after this setting.	40 µs
Read Busy Flag & Address	0 1 BF	AC	Reads Busy flag (BF) indicating internal operation is being performed and reads address counter contents.	40 µs
Write Data CG or DD RAM	1 0	Write Data	Writes data into DD or CG RAM.	40 µs
Read Data CG or DD RAM	1 1	Read Data	Reads data from DD or CG RAM.	40 µs

Notes:

1. Execution times are maximum times when fcp or fosc is 250 kHz.
2. Execution time changes when frequency changes. Ex: When fcp or fosc is 270 kHz: 40 µs × 250 / 270 =37 µs.
3. Abbreviations:

DD RAM	Display data RAM		
CG RAM	Character generator RAM		
ACC	CG RAM address		
ADD	DD RAM address, corresponds to cursor address		
AC	Address counter used for both DD and CG RAM addresses.		
1/D = 1	Increment	1/D = 0	Decrement
S=1	Accompanies display shift		
S/C = 1	Display shift;	S/C = 0	Cursor move
R/L = 1	Shift to the right;	R/L = 0	Shift to the left
DL = 1	8 bits, DL = 0: 4 bits		
N = 1	1line, N = 0 : 1 line		
F = 1	5 x 10 dots, F = 0 : 5 x 7 dots		
BF = 1	Internal operation;	BF = 0	Can accept instruction

Stepper Motors

- more accurately controlled than a normal motor allowing fractional turns or n revolutions to be easily done
- low speed, and lower torque than a comparable D.C. motor
- useful for precise positioning for robotics
- Servomotors require a position feedback signal for control

Stepper Motor Diagram

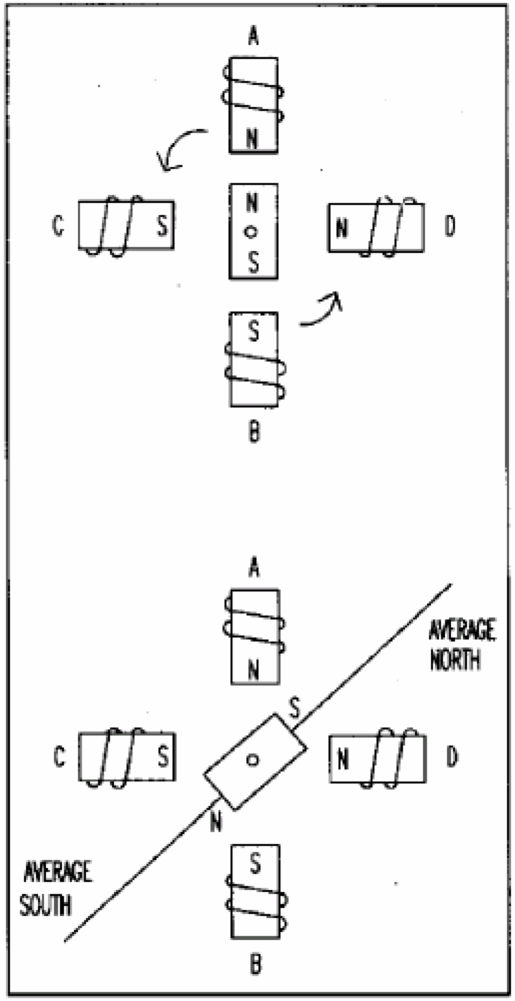


Figure 4-38. Rotor Alignment

(Courtesy of Superior Electric Company)

Stepper Motor Step Angles

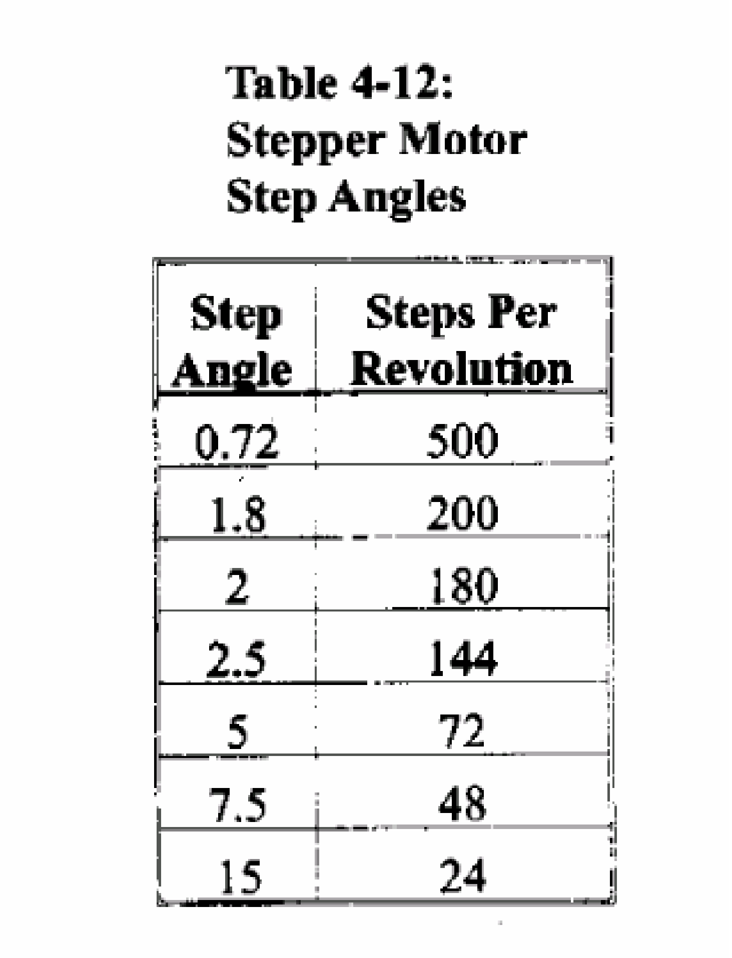


Table 4-12: Stepper Motor Step Angles

Step Angle	Steps Per Revolution
0.72	500
1.8	200
2	180
2.5	144
5	72
7.5	48
15	24

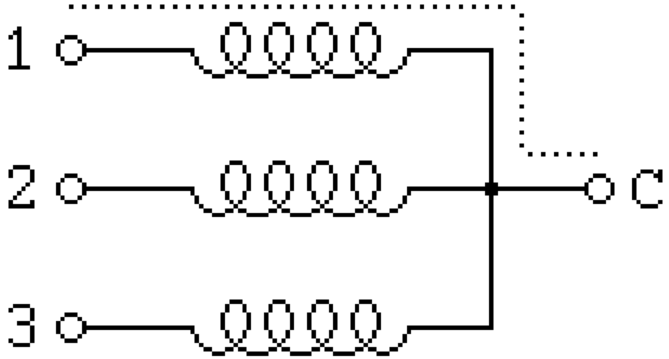
Terminology

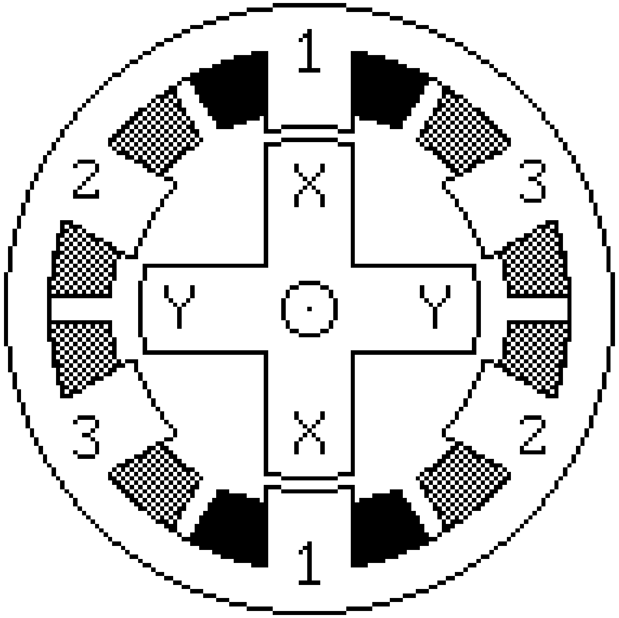
- Steps per second, RPM
  - SPS = (RPM * SPR) /60
- Number of teeth
- 4-step, wave drive 4-step, 8-step
- Motor speed (SPS)
- Holding torque

Stepper Motor Types

- Variable Reluctance
- Permanent Magnet

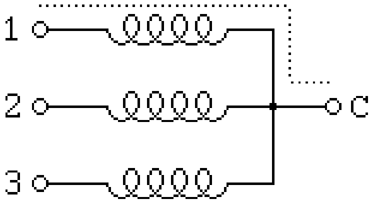
Variable Reluctance Motors

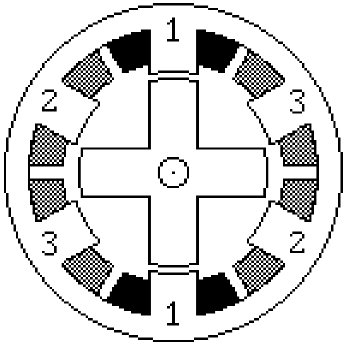




Variable Reluctance Motors

- This is usually a four wire motor – the common wire goes to the +ve supply and the windings are stepped through
- Our example is a 30^o motor
- The rotor has 4 poles and the stator has 6 poles
- Example

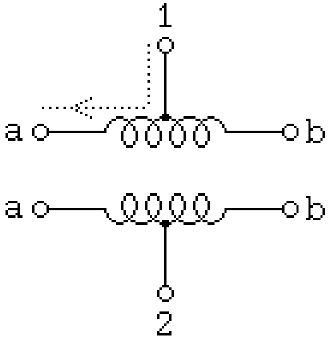


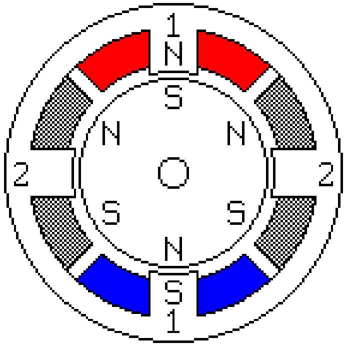


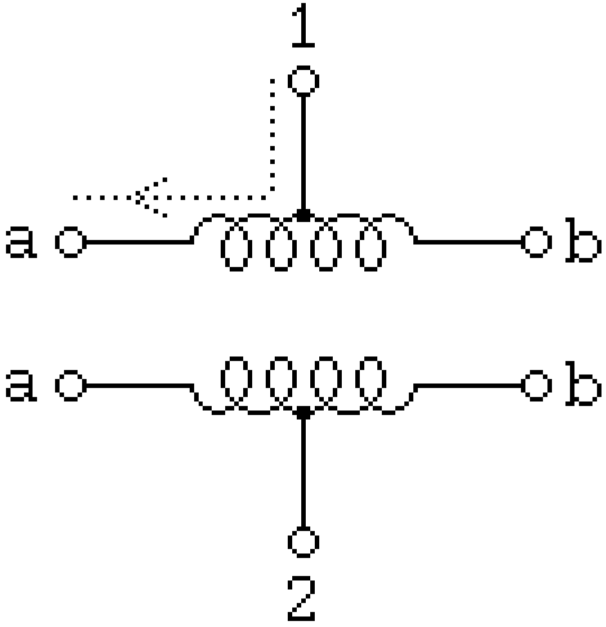
Variable Reluctance Motors

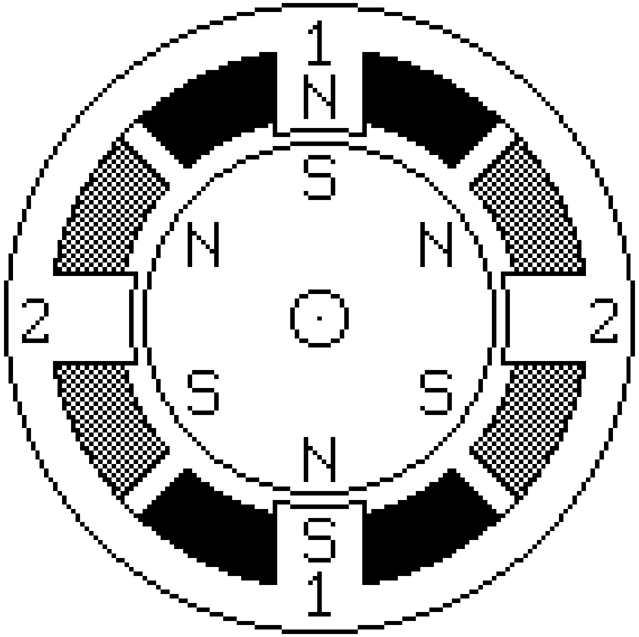
- To rotate we excite the 3 windings in sequence
  - W1 - 1001001001001001001001001
  - W2 - 0100100100100100100100100
  - W3 - 0010010010010010010010010
- This gives two full revolutions

Unipolar Motors









Unipolar Motors

- To rotate we excite the 2 windings in sequence
  - W1a - 1000100010001000100010001
  - W1b - 0010001000100010001000100
  - W2a - 0100010001000100010001000
  - W2b - 0001000100010001000100010
- This gives two full revolutions

Basic Actuation Wave Forms

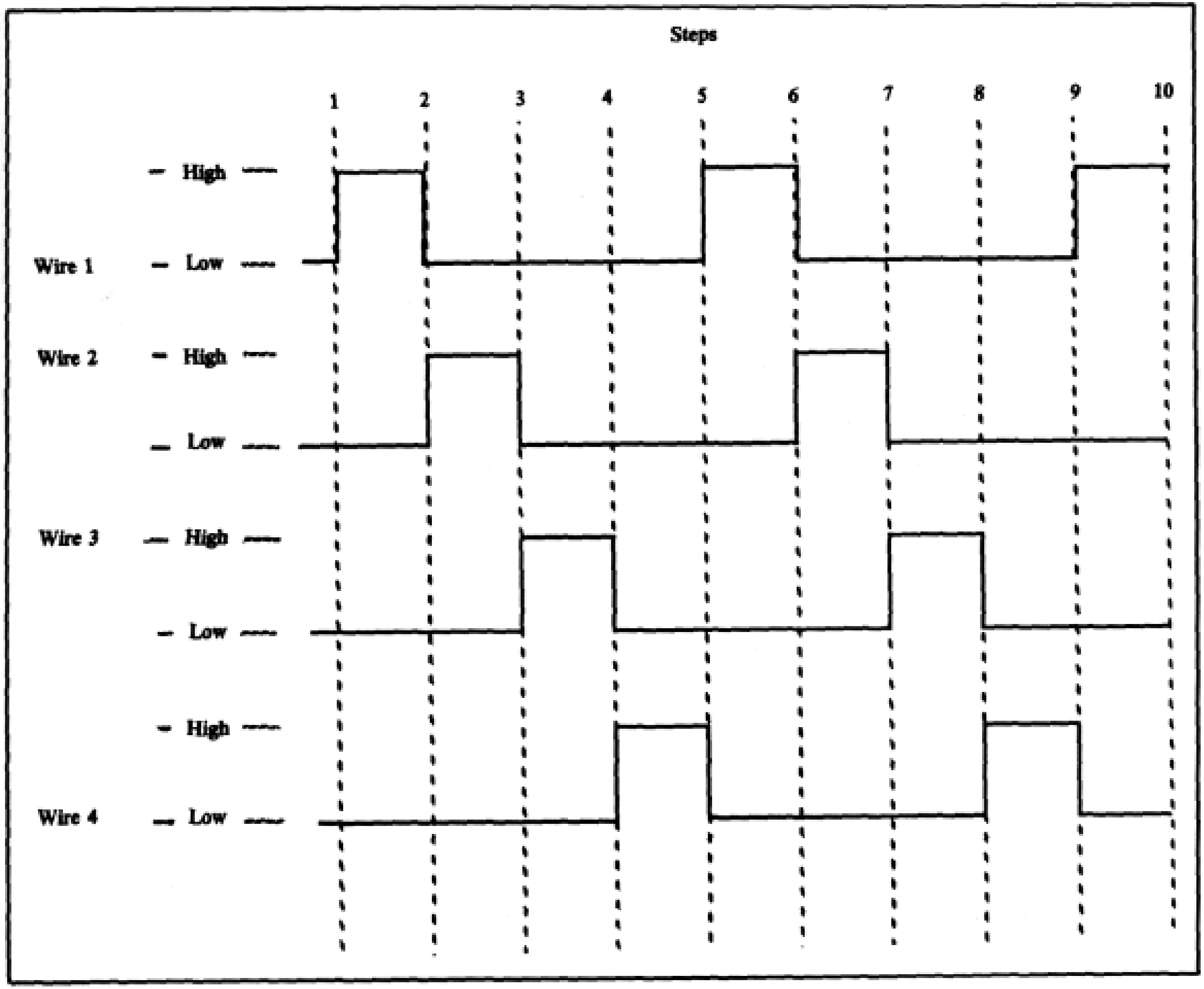


Fig. 14-4. The basic wave-step actuation sequence of a four-phase stepper motor.

Unipolar Motors

- To rotate we excite the 2 windings in sequence
  - W1a - 1100110011001100110011001
  - W1b - 0011001100110011001100110
  - W2a - 0110011001100110011001100
  - W2b - 1001100110011001100110011
- This gives two full revolutions at 1.4 times greater torque but twice the power

Enhanced Waveforms

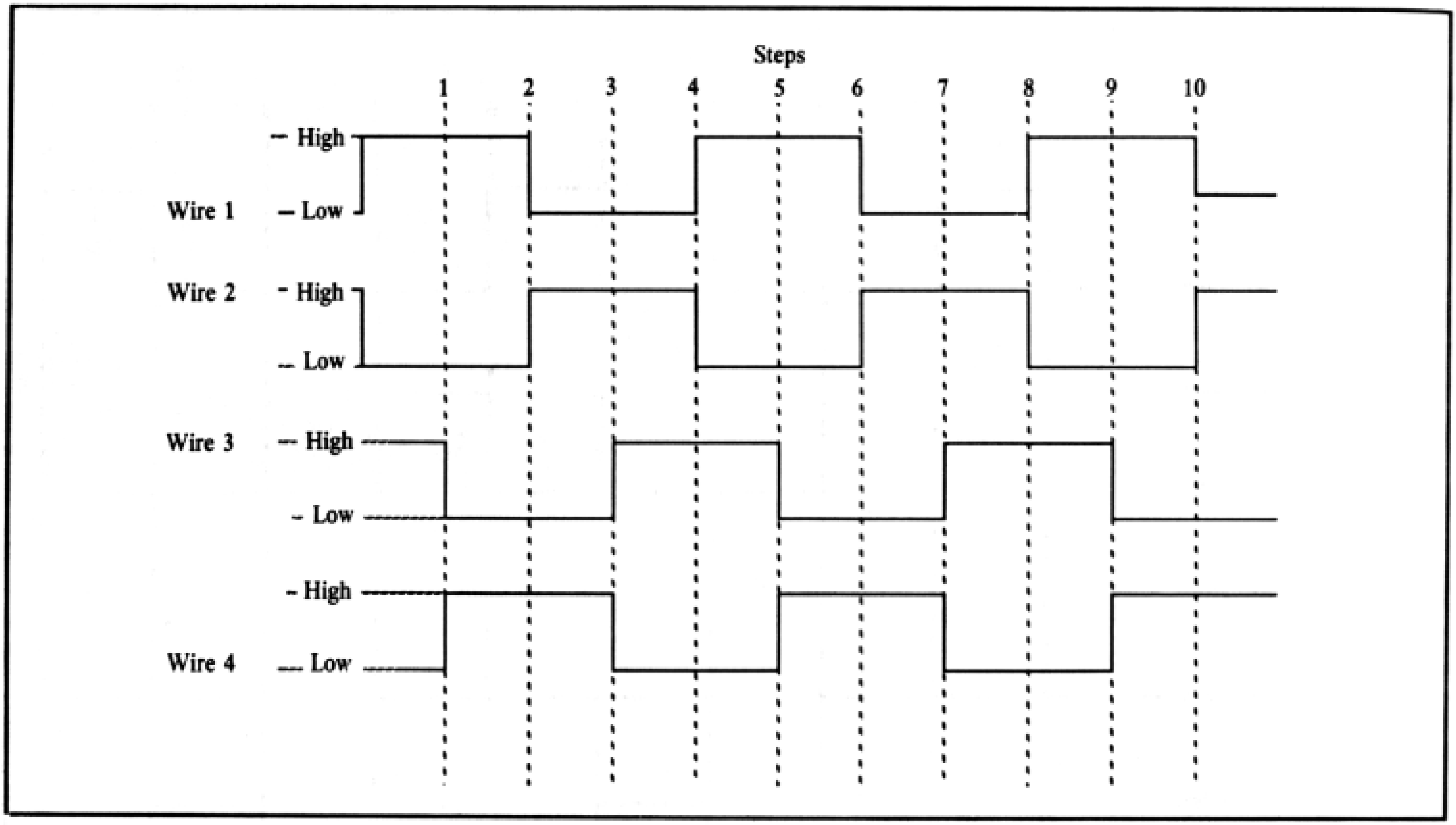


Fig. 14-5. The enhanced on-on/off-off actuation sequence of a four-phase stepper motor.

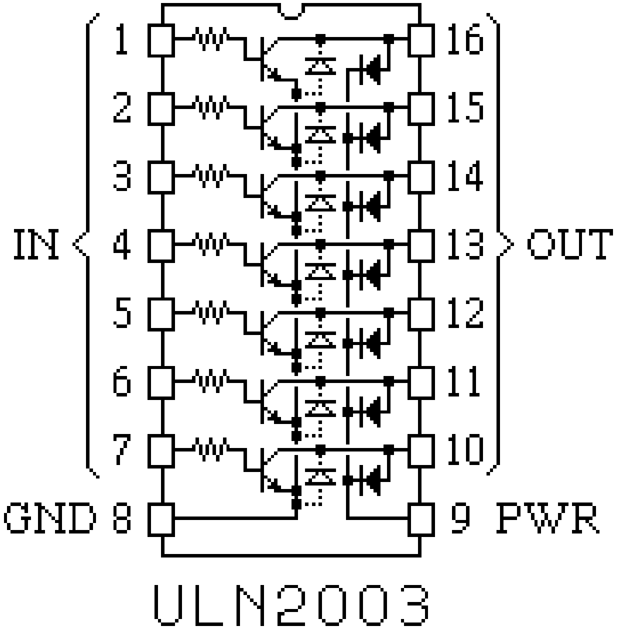
- better torque
- more precise control

Unipolar Motors

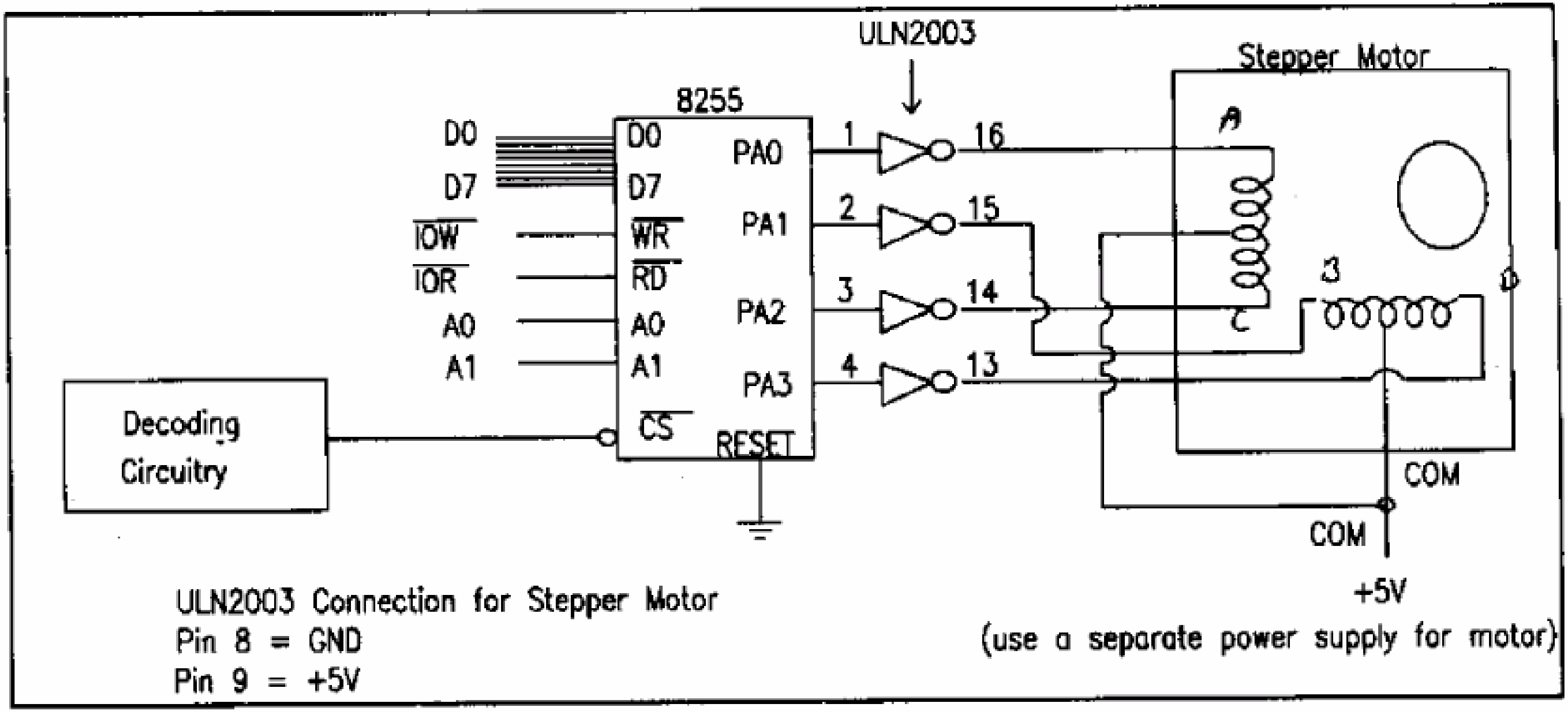
- The two sequences are not the same, so by combining the two you can produce half stepping
  - W1a - 11000001110000011100000111
  - W1b - 00011100000111000001110000
  - W2a - 01110000011100000111000001
  - W2b - 00000111000001110000011100

Motor Control Circuits

- For low current options the ULN200x family of Darlington Arrays will drive the windings direct.



Interfacing to Stepper Motors

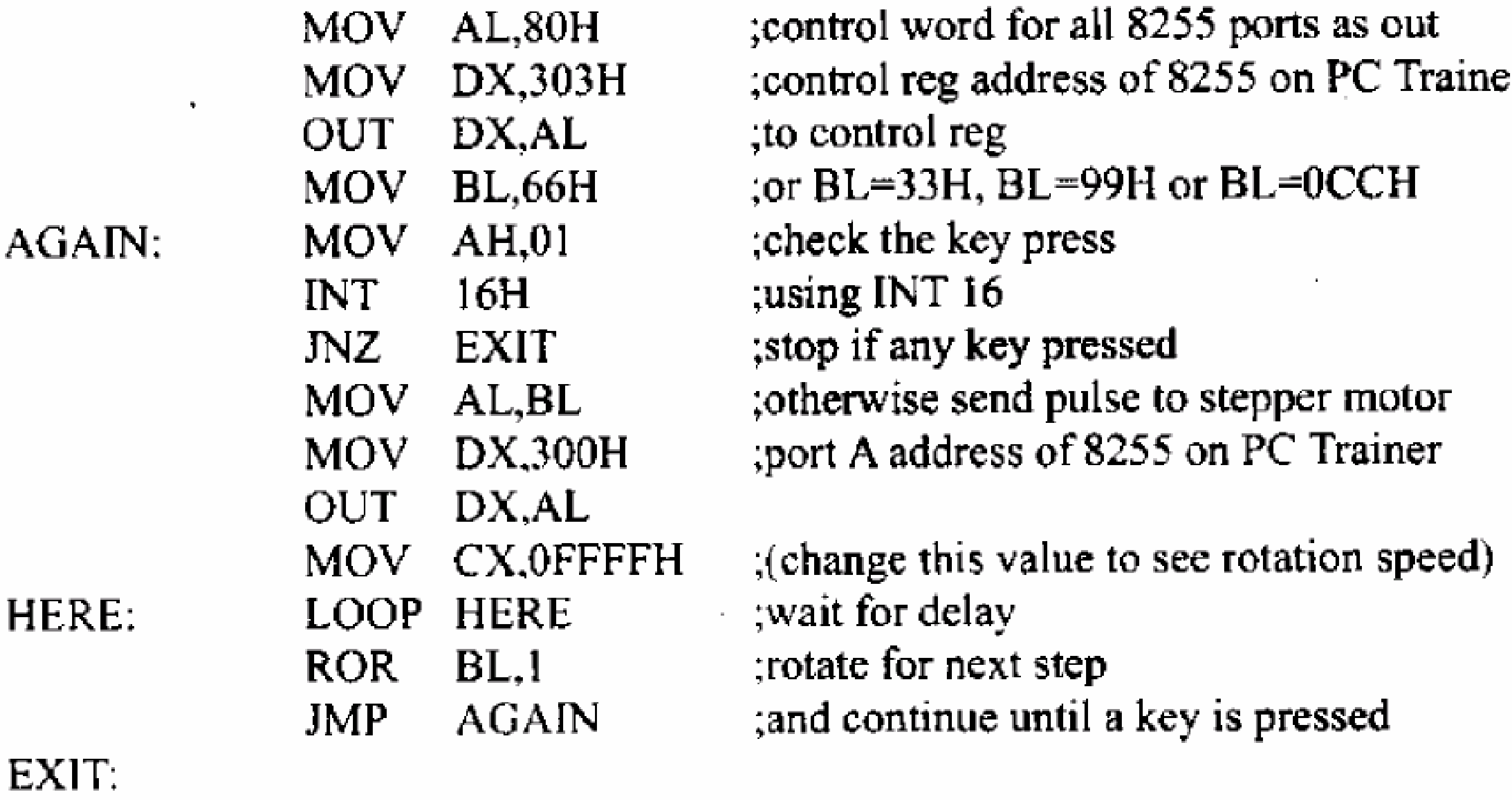


ULN2003 Connection for Stepper Motor
Pin 8 = GND
Pin 9 = +5V

(use a separate power supply for motor)

Figure 4-40. 8255 Connection to Stepper Motor

Example (với 80x86)



```
        MOV   AL,80H       ;control word for all 8255 ports as out
        MOV   DX,303H      ;control reg address of 8255 on PC Trainer
        OUT   DX,AL        ;to control reg
        MOV   BL,66H       ;or BL=33H, BL=99H or BL=0CCH
AGAIN:  MOV   AH,01        ;check the key press
        INT   16H          ;using INT 16
        JNZ   EXIT         ;stop if any key pressed
        MOV   AL,BL        ;otherwise send pulse to stepper motor
        MOV   DX,300H      ;port A address of 8255 on PC Trainer
        OUT   DX,AL
        MOV   CX,0FFFFH    ;(change this value to see rotation speed)
HERE:   LOOP  HERE         ;wait for delay
        ROR   BL,1         ;rotate for next step
        JMP   AGAIN        ;and continue until a key is pressed
EXIT:
```

Giao tiếp với DAC

Digital-to-analog converters

- A digital-to-analogue converter (DAC) translates a digital value to analogue signals.
- The analogue output of an DAC is proportional to its digital input.

- The analogue signals could be voltage or current.
- Digital input can be entered via either a parallel or serial interface.

- Examples of DAC ICs
  - ❑ DAC0800: current mode, 8-bit, parallel interface.
  - ❑ MAX5521: voltage mode, 10-bit, serial interface.

DAC conversion equation

- Let's consider a n-bit voltage-mode DAC.
- Let d = $D_{n-1}D_{n-2}$ …D_0 be the digital input to the DAC.
- Let V_{ref} be the input reference voltage.

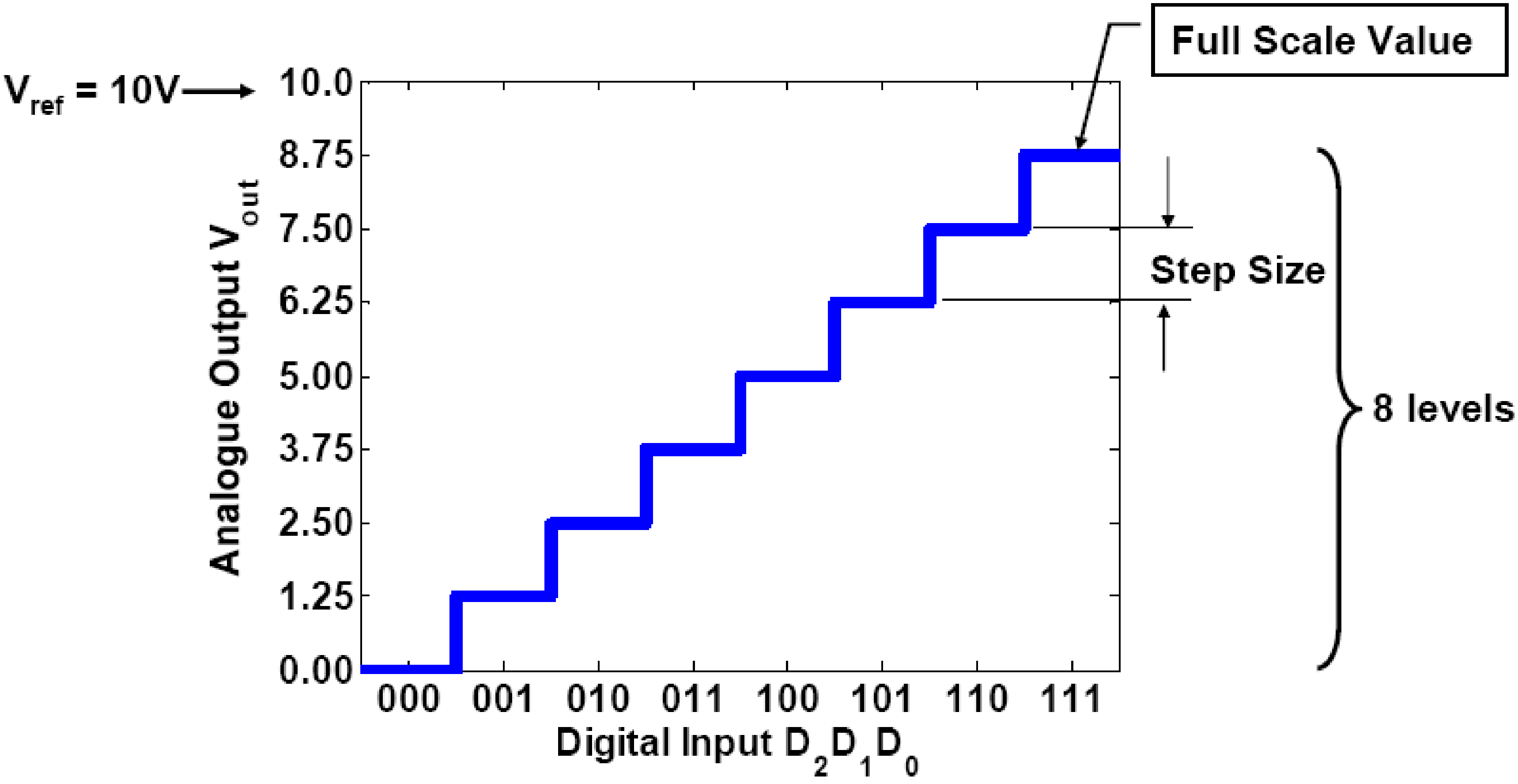
- The output voltage produced by the DAC is

$$V_{out} = V_{ref}\left(\frac{D_{n-1}}{2} + \frac{D_{n-2}}{2^2} + ... + \frac{D_2}{2^{n-2}} + \frac{D_1}{2^{n-1}} + \frac{D_0}{2^n}\right)$$

- In other words,

$$V_{out} = d \times \frac{V_{ref}}{2^n}$$

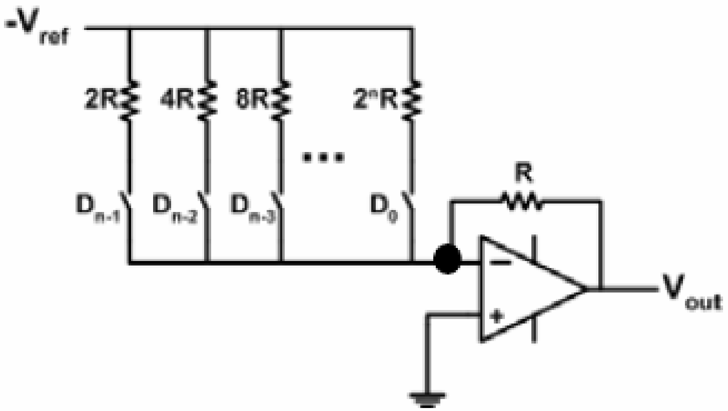
DAC example



Input-output plot of a 3-bit DAC

DAC designs

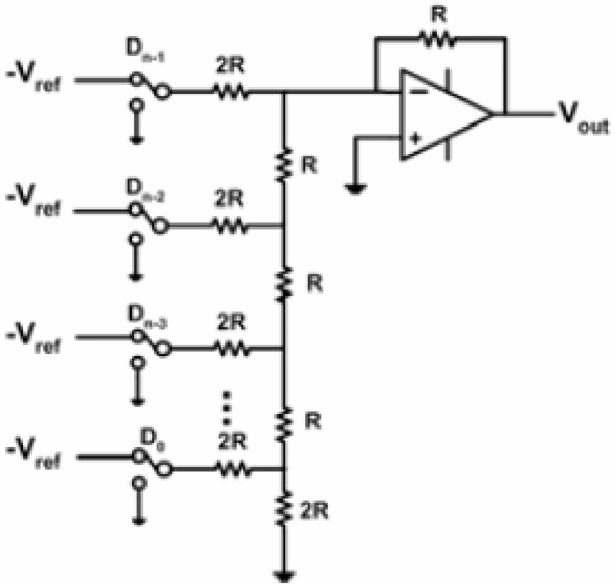
Weighted resistor design



$$-\frac{V_{ref}D_{n-1}}{2R} - \frac{V_{ref}D_{n-2}}{2^2R} + \ldots . - \frac{V_{ref}D_0}{2^nR} + \frac{V_{out}}{R} = 0$$

- The weighted resistor design requires precise resistors over a wide range: very difficult to achieve.

R/2R ladder design



$$V_{out} = V_{ref}\left(\frac{D_{n-1}}{2} + \frac{D_{n-2}}{2^2} + \ldots . + \frac{D_0}{2^n}\right)$$

- The R-2R ladder design requires only two resistor values.

Parameters of a DAC

- **Number of bits**: n = 8, 10, 12, 13, 14, etc.

  A larger n means smoother output.

- **Full scale value**: maximum output (i.e. when d = $2^n - 1$)

$$V_{FS} = V_{ref} \times \frac{2^n - 1}{2^n}$$

- **Step size** (resolution): the smallest change that can be produced

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

- **Quantization error** is half of the step size.

- **Percentage resolution**:

$$percentage\ resolution = \frac{step\ size}{V_{FS}} = \frac{100\%}{2^n - 1}$$

Parameters of a DAC

- **Settling time**: time between the application of input and the output settling to within a specified band of the final value.

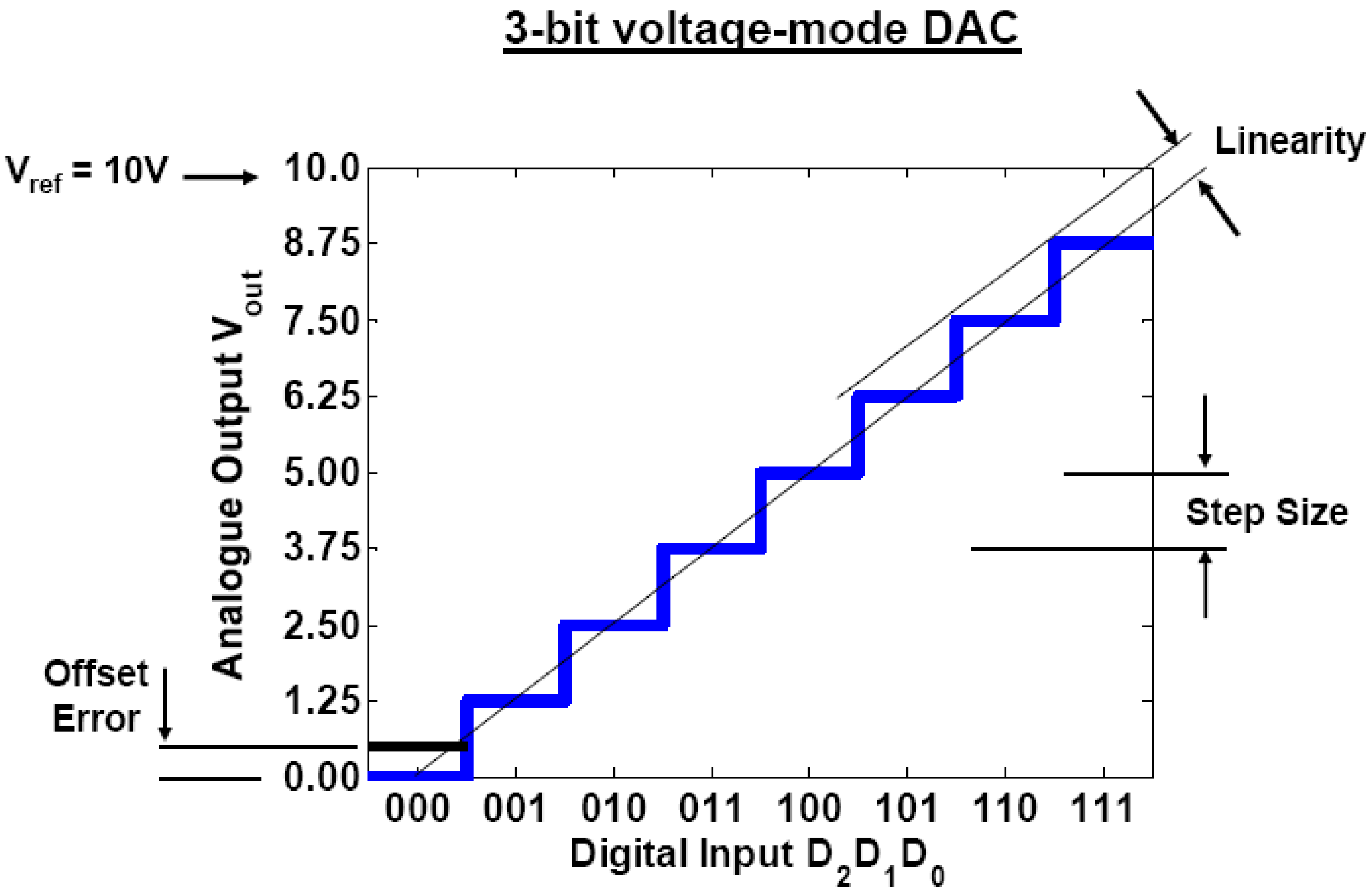
- **Offset error**: the nonzero DAC output when a 0 output is requested (by putting d = 0).

- **Full scale error**: the maximum deviation of the actual FS output from the expected FS output.

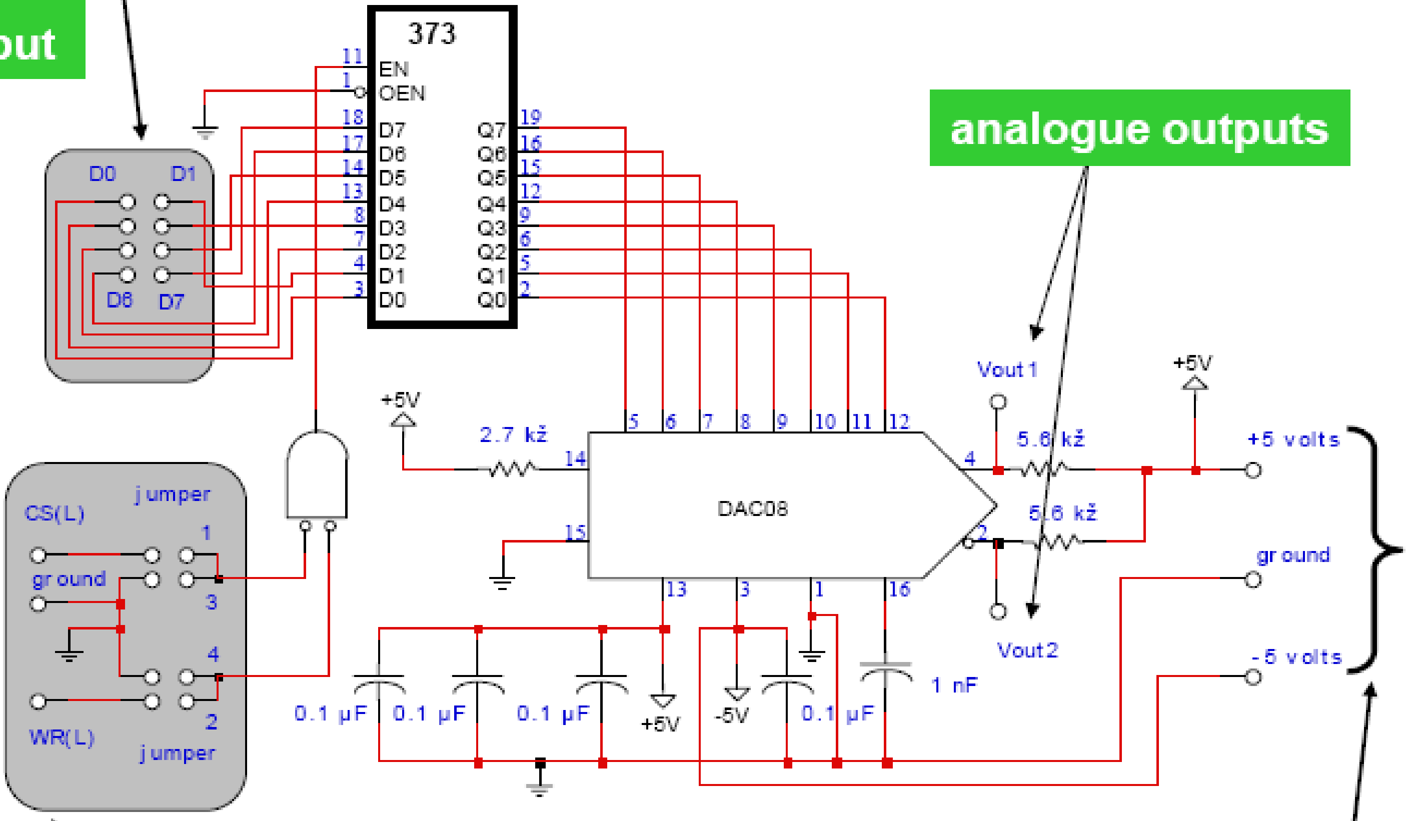
- **Linearity**: maximum deviation from the best-bit line for of input-output plot.

- **Monotonic**: increasing or decreasing input-output function.

Parameters of a DAC



DAC board



8-bit digital input

analogue outputs

2 jumpers to select memory-mapped or ioport-mapped

reference voltages

Digital to Analog Converter

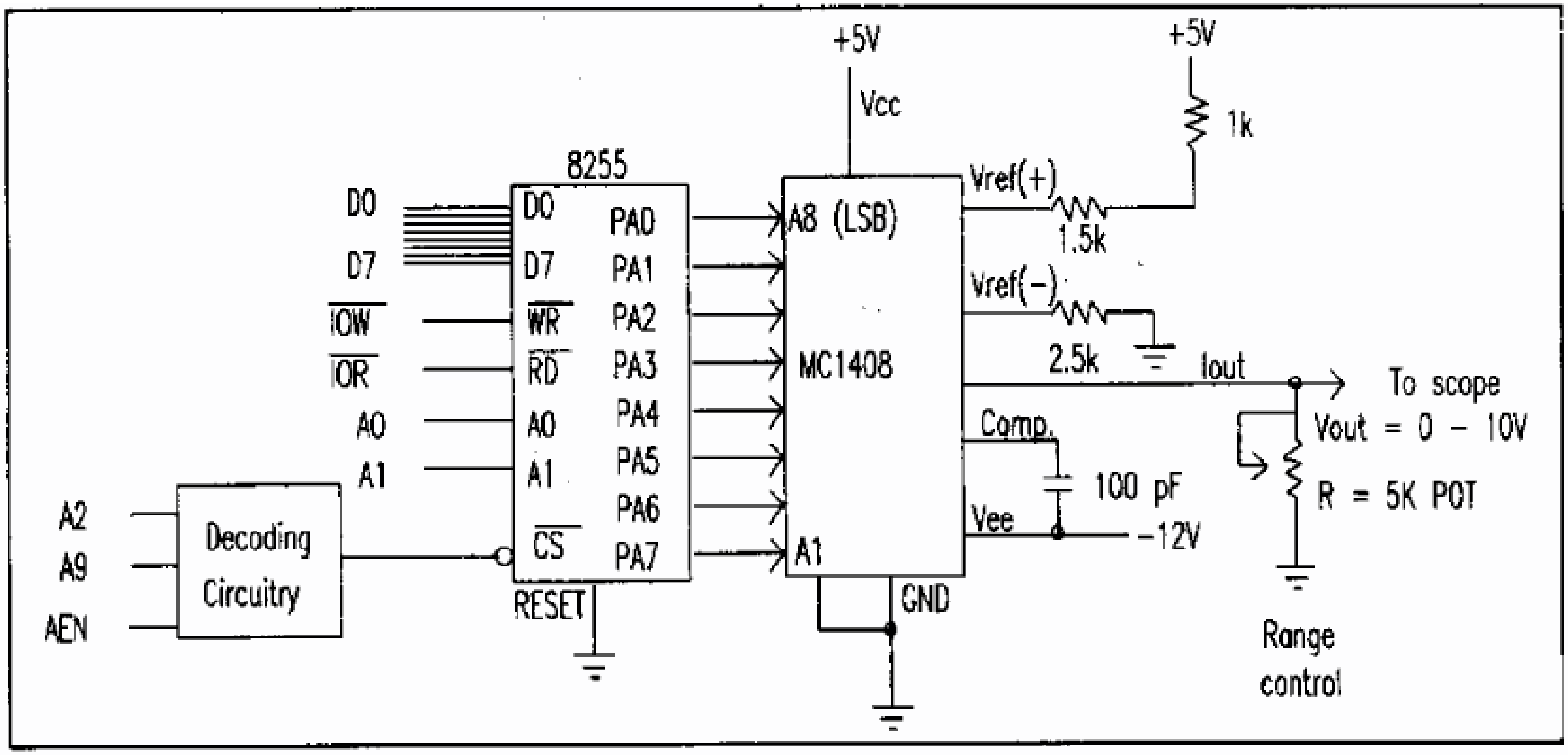
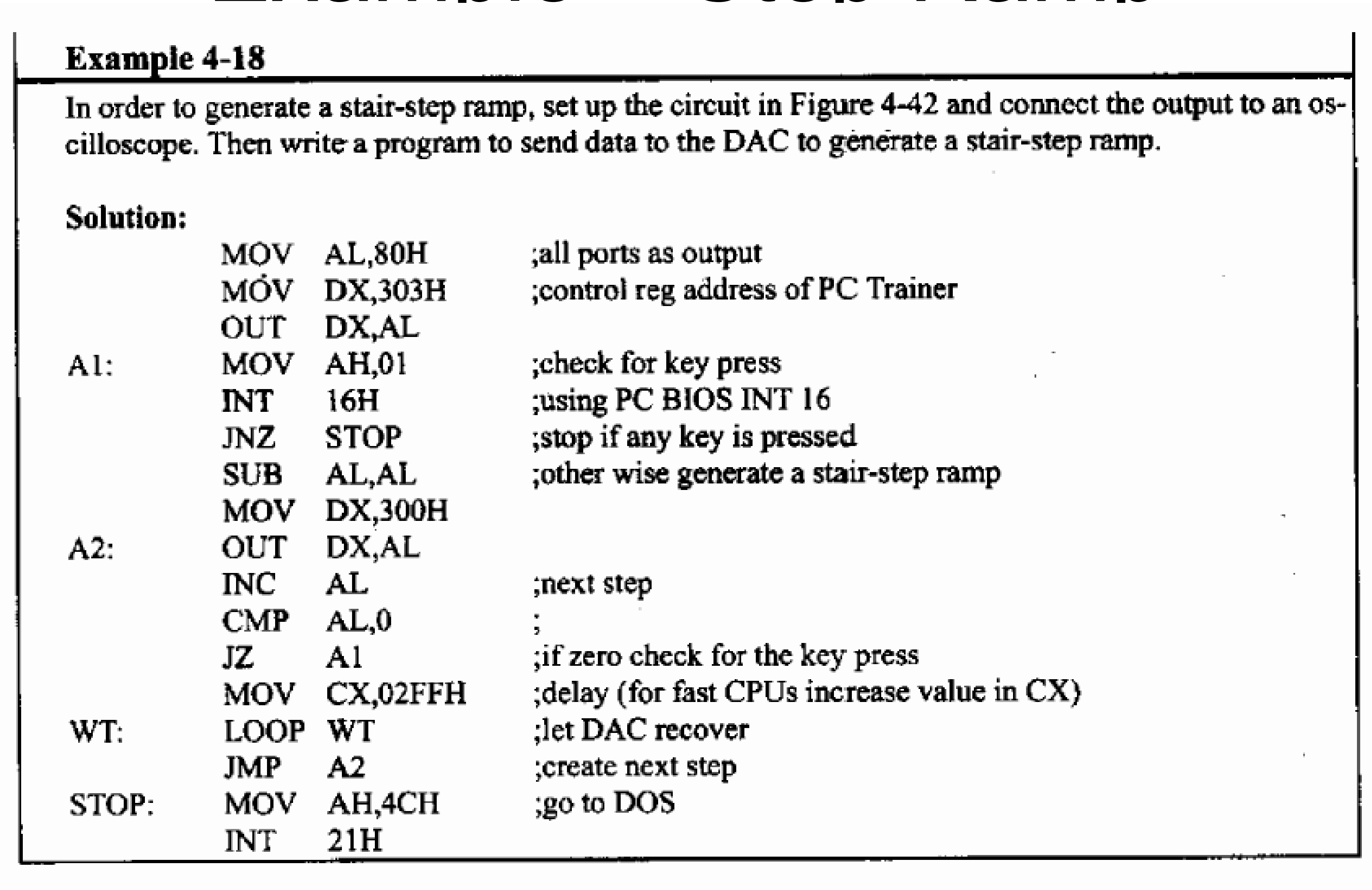


Figure 4-42. DAC Connection to 8255

Resolution: 8 bit

Example – Step Ramp



Example 4-18

In order to generate a stair-step ramp, set up the circuit in Figure 4-42 and connect the output to an oscilloscope. Then write a program to send data to the DAC to generate a stair-step ramp.

Solution:

```
            MOV   AL,80H        ;all ports as output
            MOV   DX,303H       ;control reg address of PC Trainer
            OUT   DX,AL
A1:         MOV   AH,01         ;check for key press
            INT   16H           ;using PC BIOS INT 16
            JNZ   STOP          ;stop if any key is pressed
            SUB   AL,AL         ;other wise generate a stair-step ramp
            MOV   DX,300H
A2:         OUT   DX,AL
            INC   AL            ;next step
            CMP   AL,0          ;
            JZ    A1            ;if zero check for the key press
            MOV   CX,02FFH      ;delay (for fast CPUs increase value in CX)
WT:         LOOP  WT            ;let DAC recover
            JMP   A2            ;create next step
STOP:       MOV   AH,4CH        ;go to DOS
            INT   21H
```

Giao tiếp với ADC

Analogue to digital converter

- An ADC samples an analogue signal at discrete times and converts the sampled signal to digital form.

- Used with transducers, ADCs allow us to monitor real-world inputs and perform control based on the inputs.

- Examples of ADC ICs
  - ❑ ADC0804: 8-bit, successive approximation
  - ❑ Maxim104: 8-bit, flash type

ADC applications

- Local positioning sensor for object tracking
  - The Receiver Signal Strength Indicator (RSSI) output of an FM receiver is measured.
  - The RSSI output voltage is inversely proportional to the distance.
- Temperature sensor for controlling shower water
  - We can measure the output voltage of a thermistor.
  - The thermistor's resistance depends on its temperature.
- Electric fence monitoring
  - Measure the voltage level of an electric fence.
  - Determine if the fence is being tampered.

ADC conversion equation

- Let’s consider an n-bit ADC.
- Let V_{ref} be the reference voltage.
- Let V_{in} be the analogue input voltage.

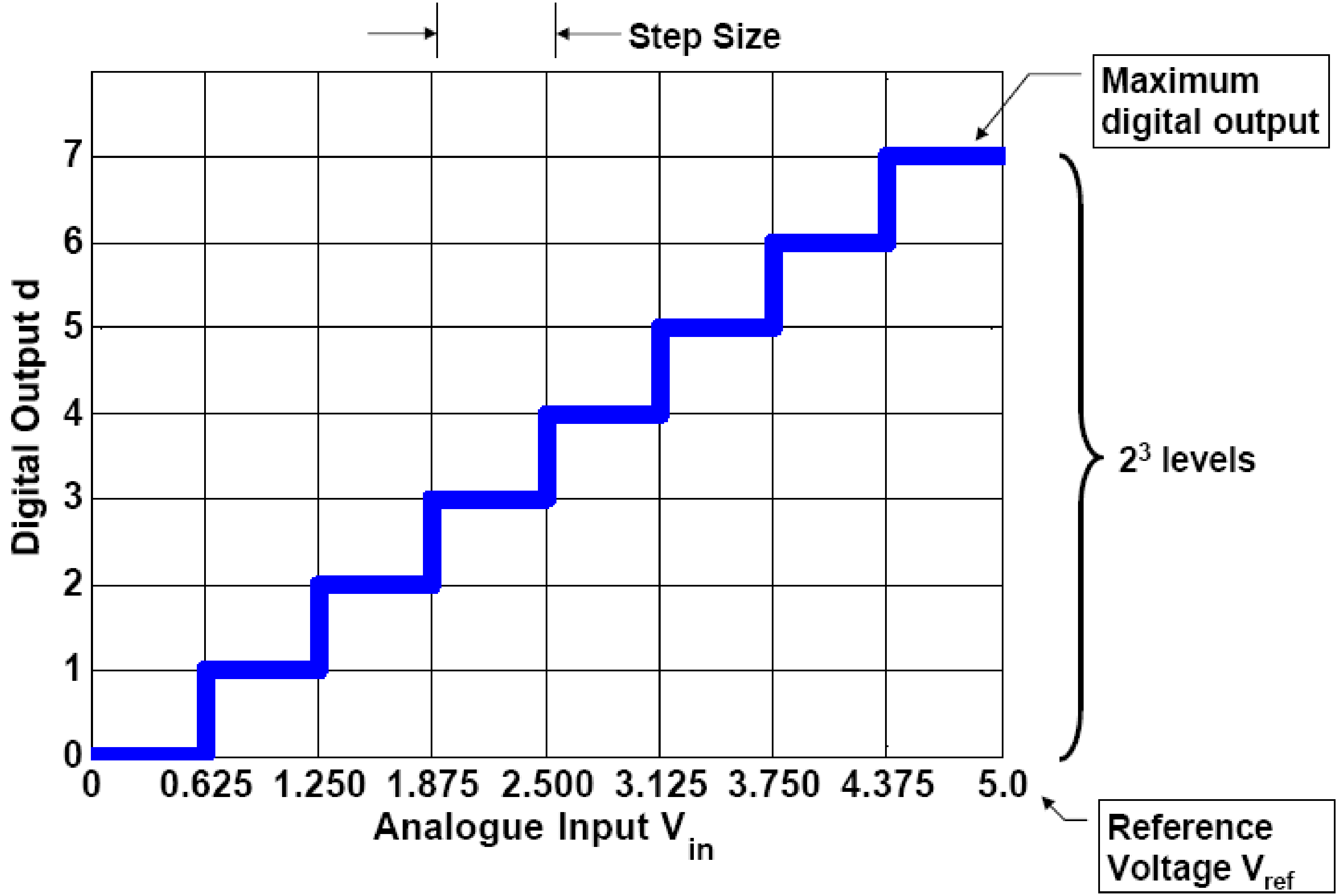
- The ADC’s digital output, d = $D_{n-1}D_{n-2} \ldots D_0$, is given as

$$d = floor[\frac{V_{in}}{step\ size}]$$

- The **step size** (**resolution**) is the smallest change in input that can be discerned by the ADC:

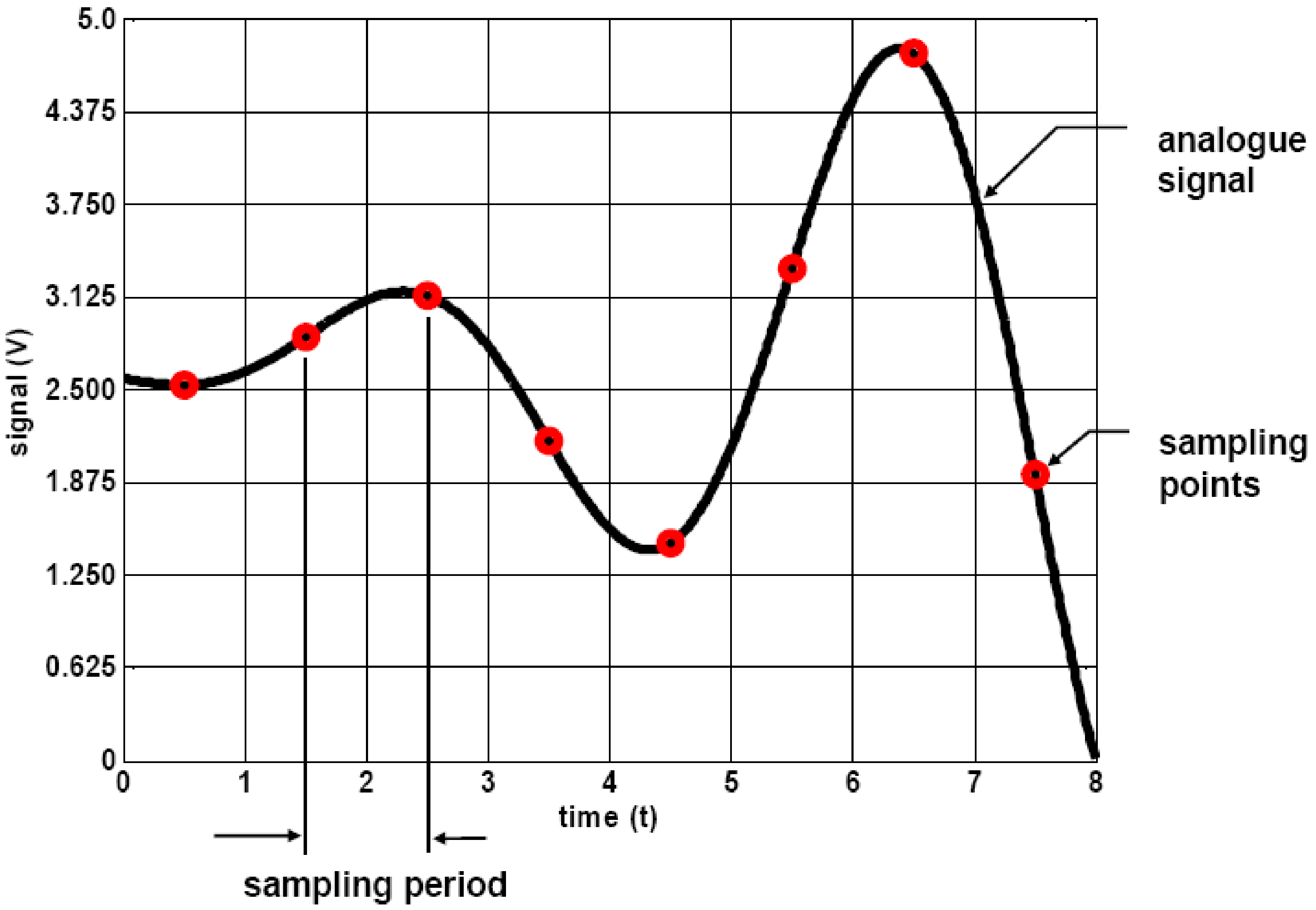
$$step\ size = \frac{V_{ref}}{2^n}$$

ADC input-output characteristics

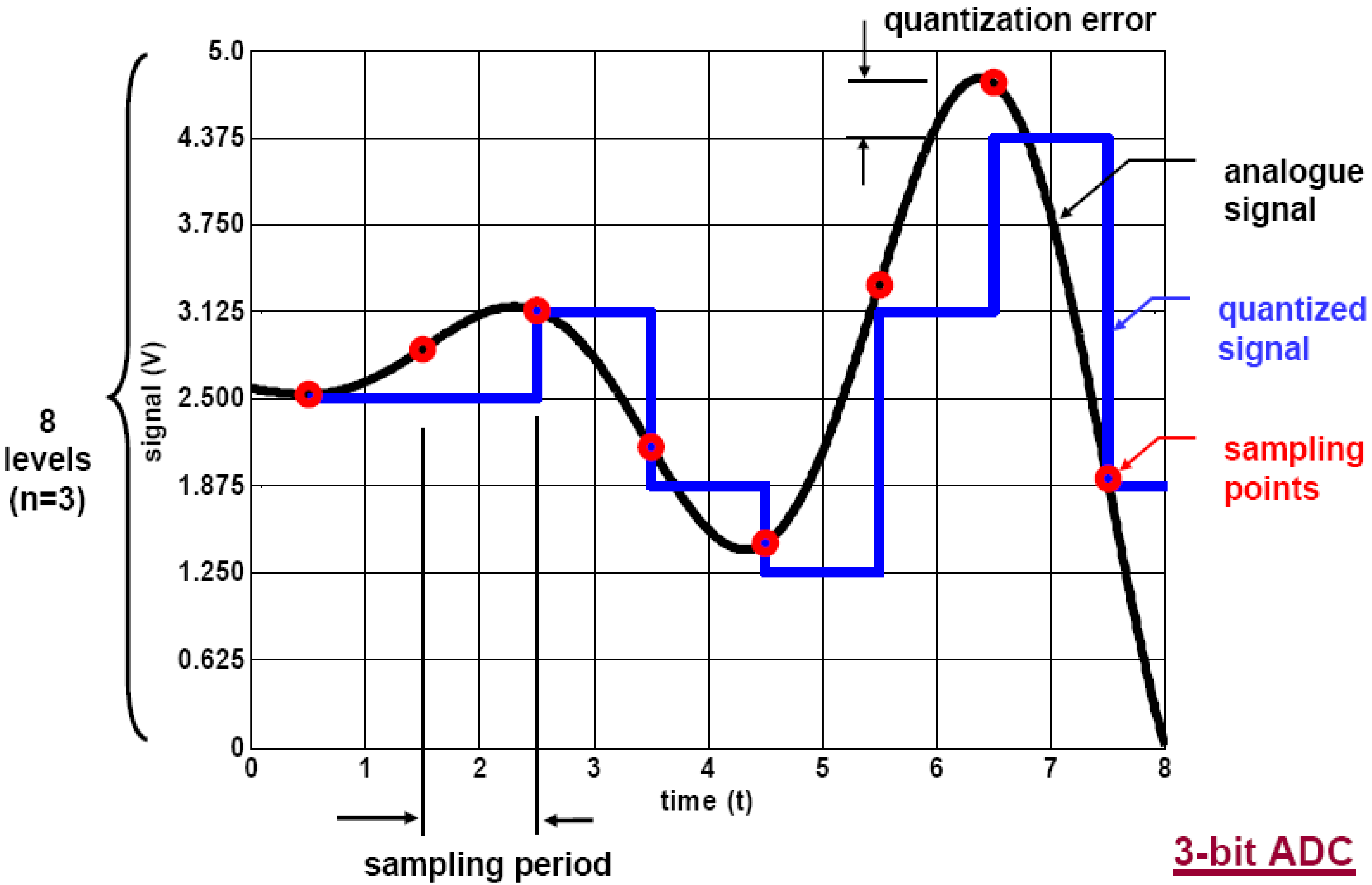


3-bit analogue to digital converter

Sampling an analogue signal



Quantizing a sampled signal



3-bit ADC

Parameters of an ADC

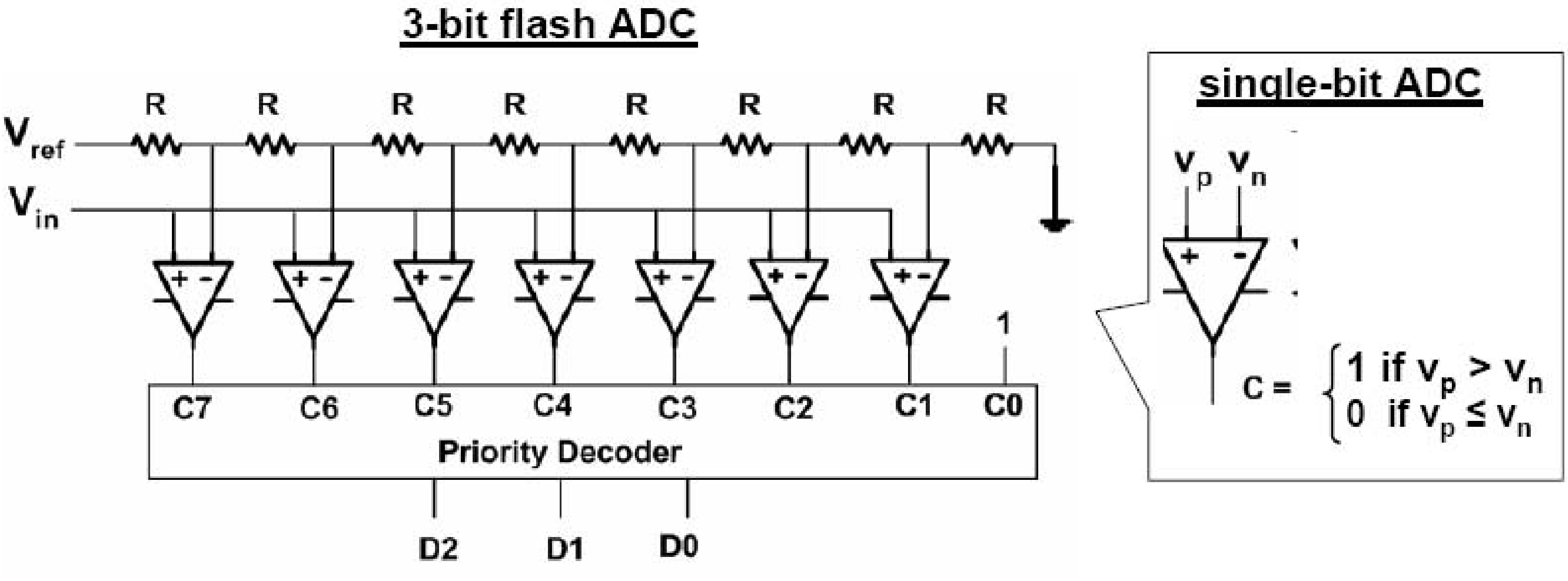
- Number of bits n: The higher the number of bits, the more precise the digital output.

- Quantisation error E_q: the average difference between the analogue input and the quantized value. The quantization error of an ideal ADC is half of the step size.

- Sample time T_{ASM}: a sampling capacitor must be charged for a duration of t_{ASM} before conversion taking place.

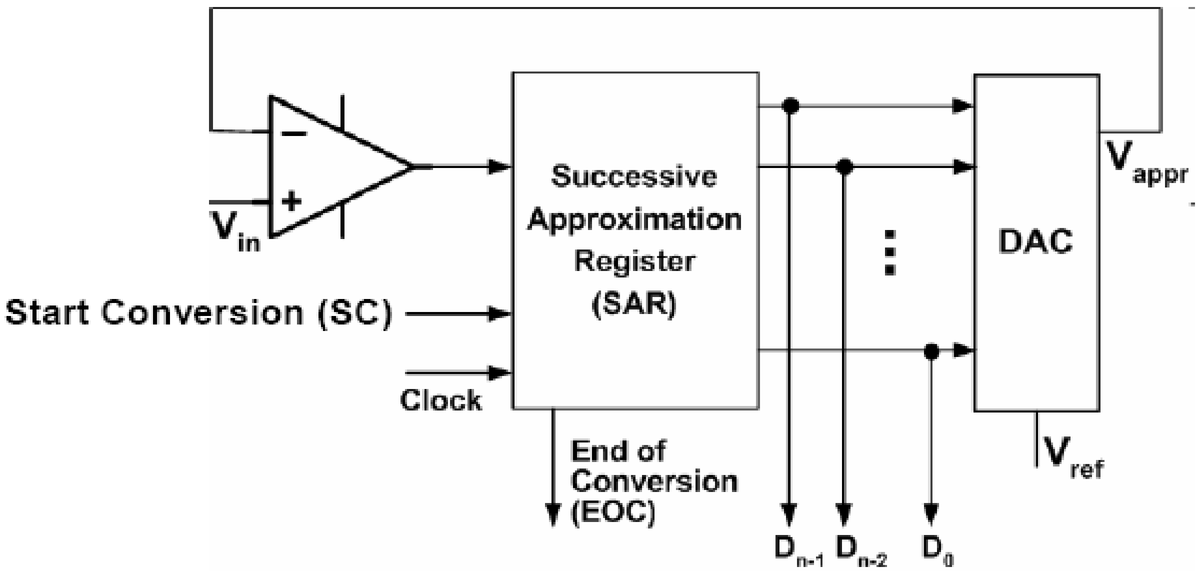
- Conversion time T_{CONV}: time taken to convert the voltage on the sampling capacitor to a digital output.

ADC designs: flash ADC



- A n-bit flash ADC uses 2^n-1 comparators and a priority decoder.
- Advantage: the fastest type of ADC.
- Disadvantages: limited resolution, expensive, and large power consumption.

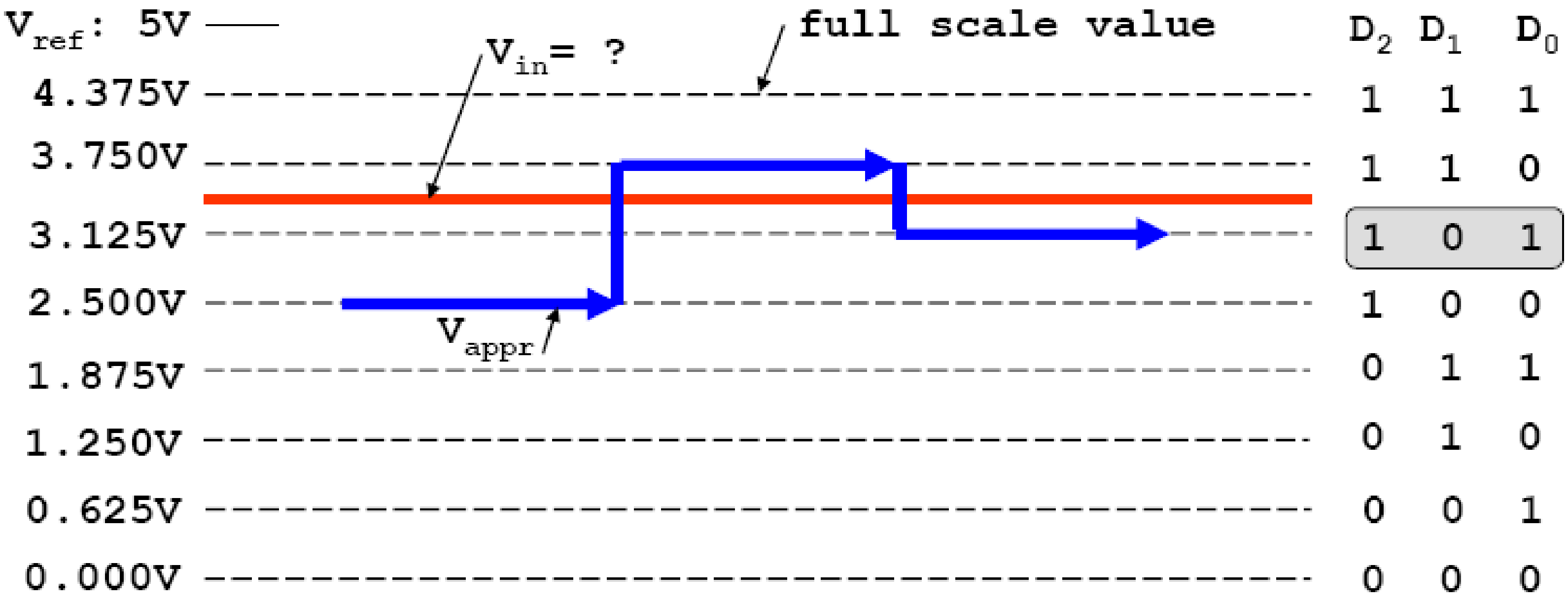
ADC designs: successive-approximation ADC



- Quite fast, one of the most widely used design for ADCs.
- A DAC is used to generate approximations of the input voltage.
- A comparator is used to compare V_{in} and V_{appr}.
- In each cycle, SAR finds one output bit using comparator’s output.
- To start conversion, set SC = 1. When conversion ends, EOC = 1.

Successive-approximation ADC

Binary search for a 3-bit ADC



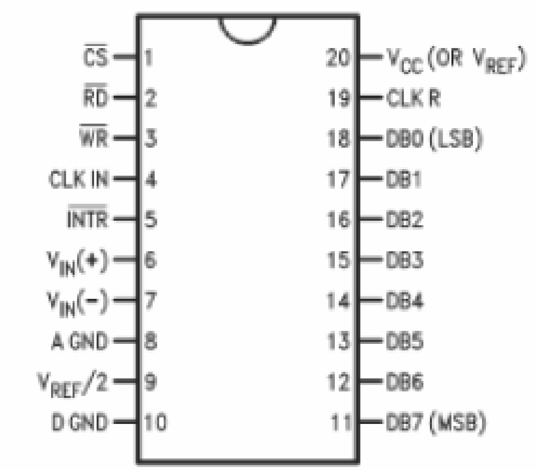
	D_2	D_1	D_0
V_{ref}: 5V			
4.375V	1	1	1
3.750V	1	1	0
3.125V	1	0	1
2.500V	1	0	0
1.875V	0	1	1
1.250V	0	1	0
0.625V	0	0	1
0.000V	0	0	0

clock cycle 1 D_2 = 1 [$V_{in} > V_{appr}$]	clock cycle 2 D_1 = 0 [$V_{in} < V_{appr}$]	clock cycle 3 D_0 = 1 [$V_{in} > V_{appr}$]

Example of ADC ICs

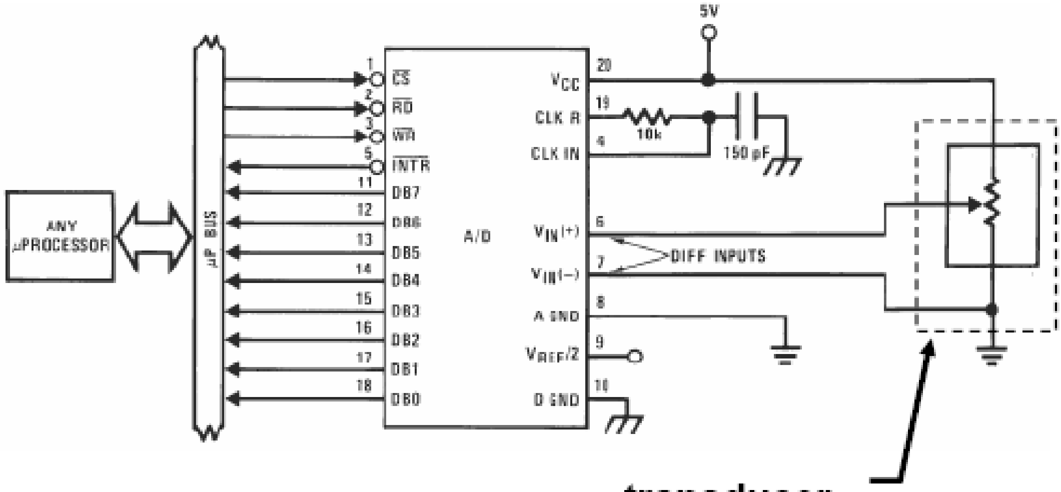
ADC0804

(National Semiconductor)



- V_{CC}: reference voltage
- RD: to read digital output
- WR: to start a new conversion
- INTR: when conversion completes
- $V_{IN}(+)$, $V_{IN}(-)$: analogue input
- DB0-DB7: 8-bit output
- CLK IN, CLK R: clock signal

A typical application of ADC0804



transducer

Analog to Digital

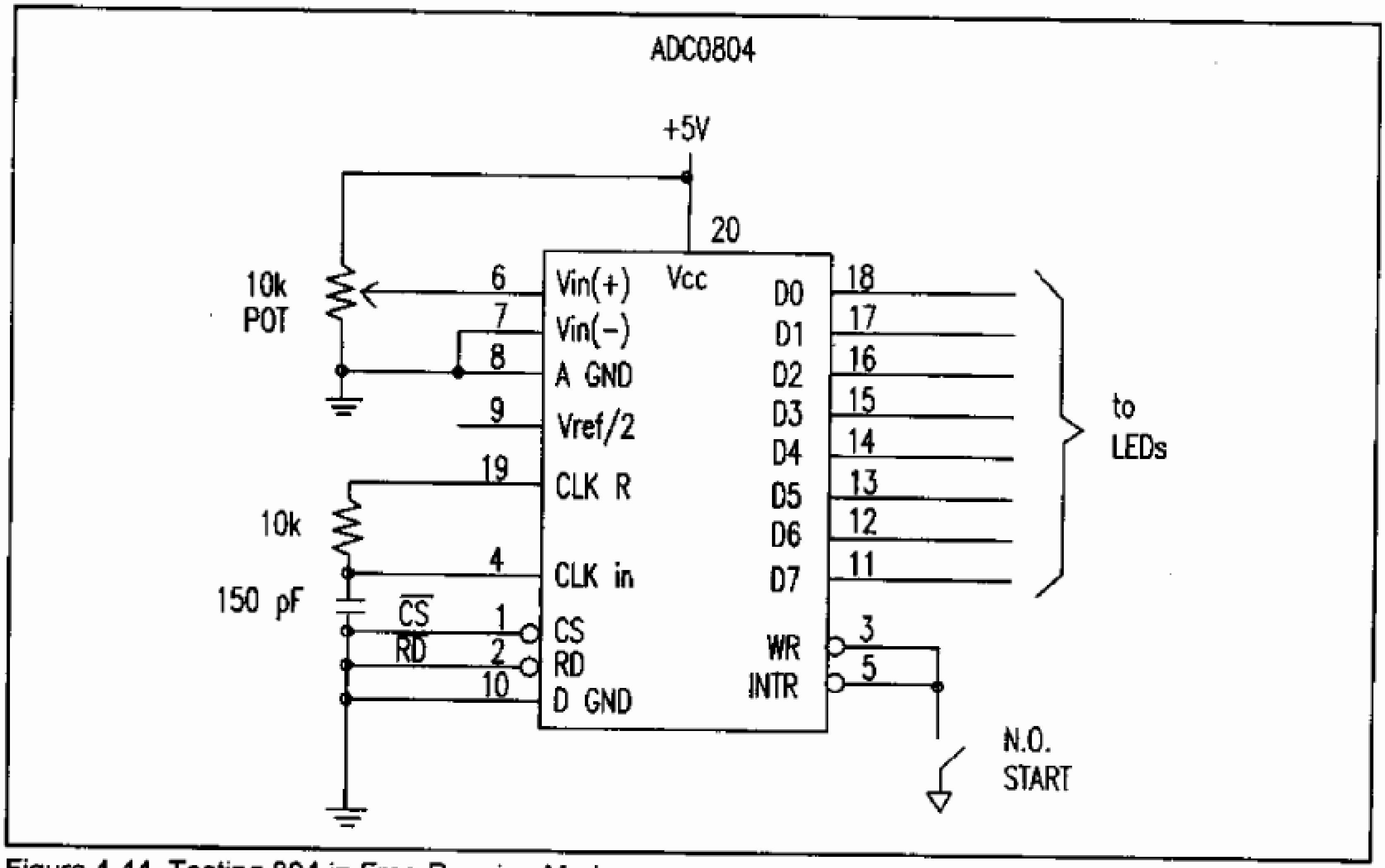


Figure 4-44. Testing 804 in Free Running Mode

V_{in} Range

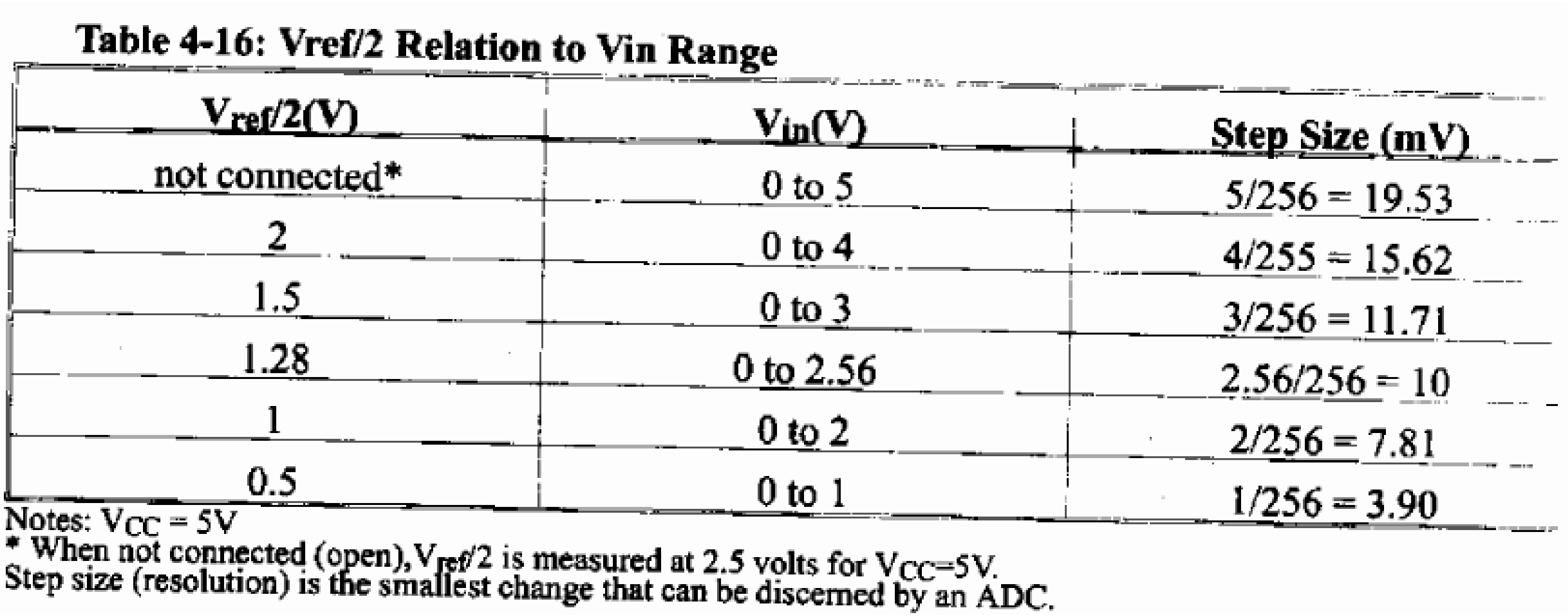


Table 4-16: Vref/2 Relation to Vin Range

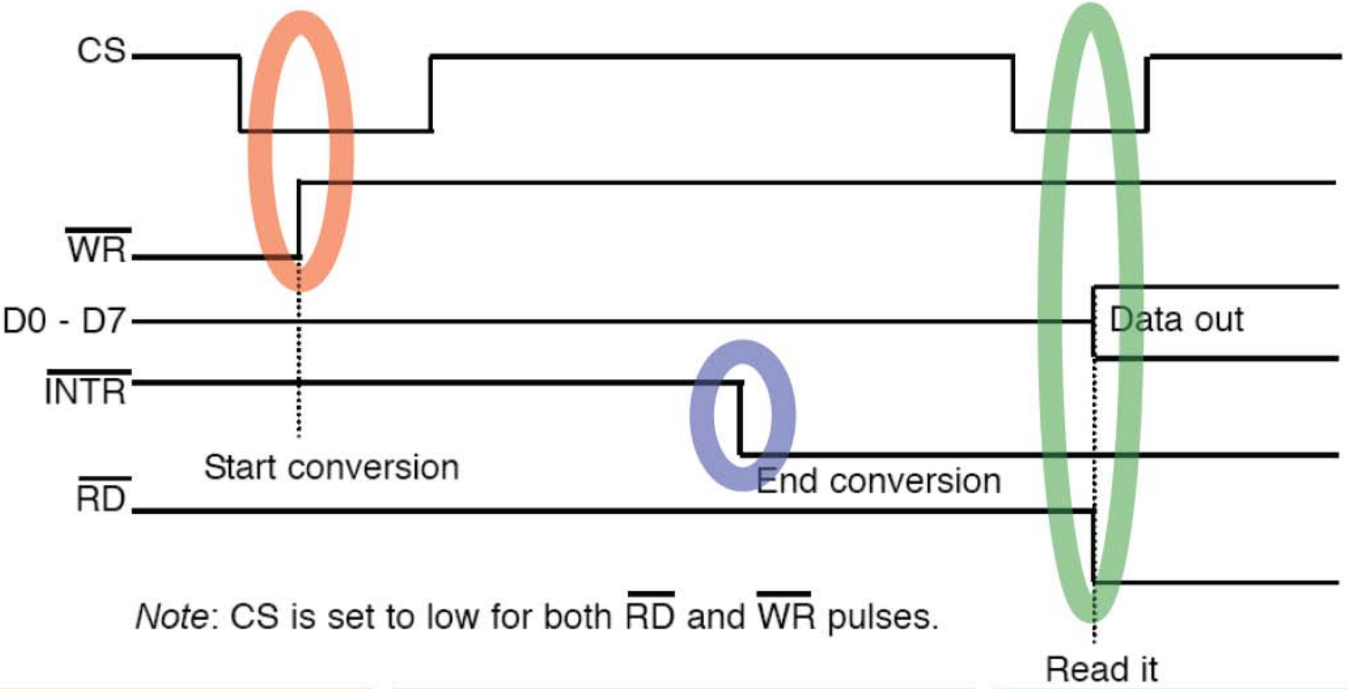
$V_{ref}/2$(V)	V_{in}(V)	Step Size (mV)
not connected*	0 to 5	5/256 = 19.53
2	0 to 4	4/255 = 15.62
1.5	0 to 3	3/256 = 11.71
1.28	0 to 2.56	2.56/256 = 10
1	0 to 2	2/256 = 7.81
0.5	0 to 1	1/256 = 3.90

Notes: V_{CC} = 5V

* When not connected (open), $V_{ref}/2$ is measured at 2.5 volts for V_{CC}=5V.

Step size (resolution) is the smallest change that can be discerned by an ADC.

Timing Diagram for ADC transaction



Note: CS is set to low for both $\overline{RD}$ and $\overline{WR}$ pulses.

1. rising /WR during /CS

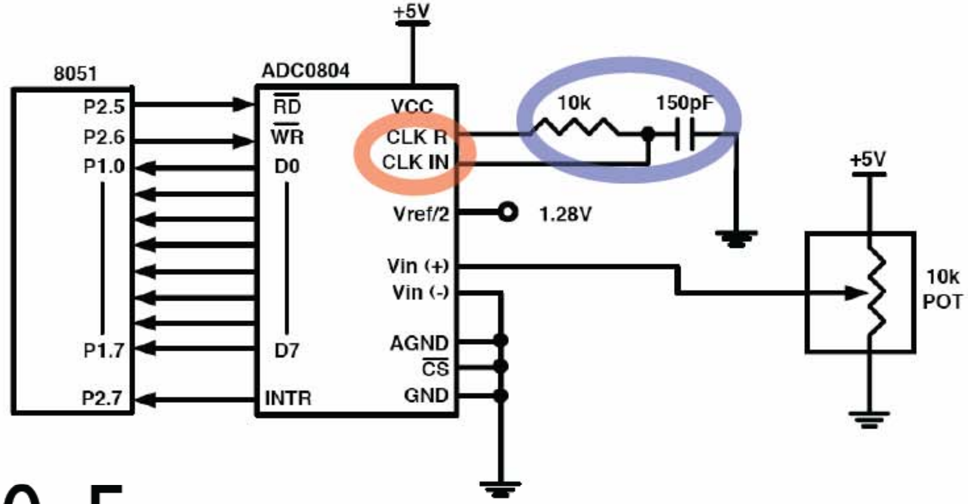
2. ADC finishes conversion

3. assert /RD during /CS

Could also assert /CS the whole time w/out pulsing!

CLK IN and CLK R

- As internal clock
  - Set the R, C value
    $f = 1/(1.1\ RC)$
  - R = 10KΩ, C = 150pF
    $f = 1/(1.1*10^4*150*10^{-12})$
    = 606060.6 => 606KHz,
    or 1.65μs cycle time
- Can also take external clock



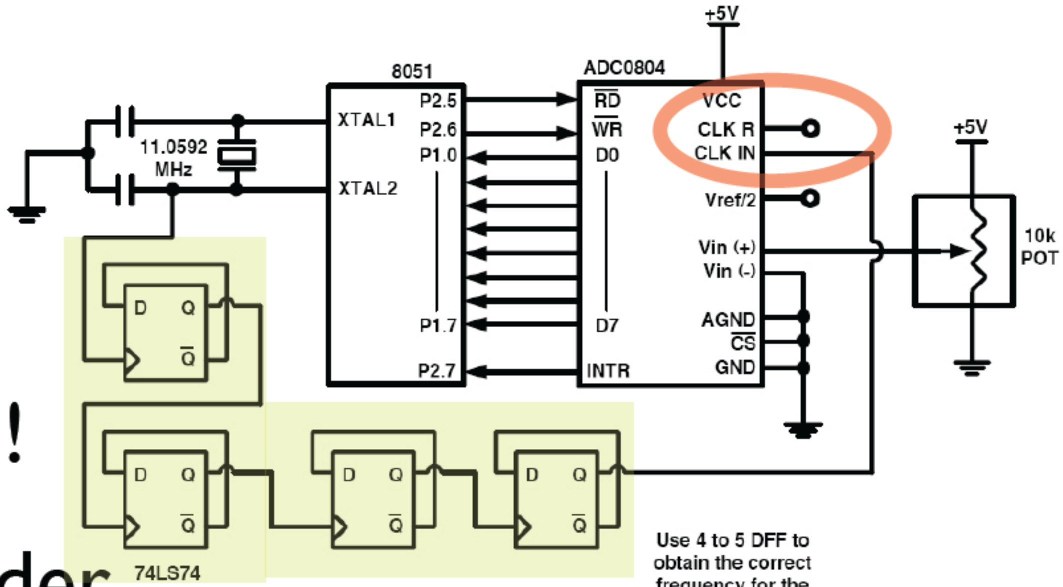
External clocking scheme for ADC0804

- Could use the same crystal as for 8051

- Issue: freq too high!

- Solution: clock divider

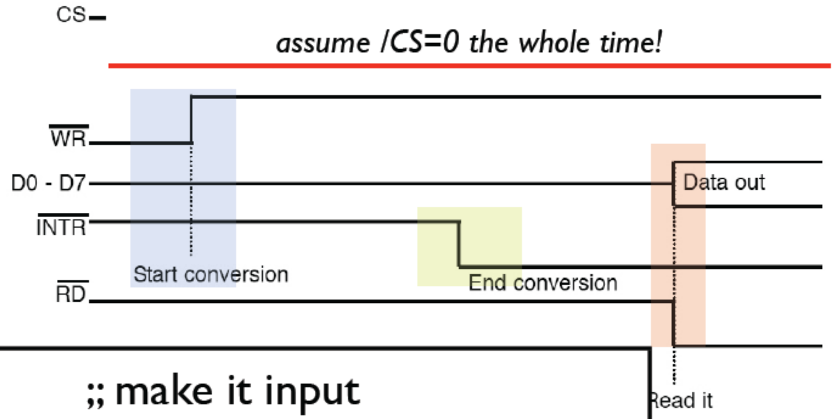
  - cascaded D-flipflops:
    next one is clocked by the prev's Q,
    each feeds /Q to its own D



Use 4 to 5 DFF to obtain the correct frequency for the ADC chip

Assembly for ADC0804

```
RD          BIT P2.5
WR          BIT P2.6
INTR        BIT P2.7
MYDATA      EQU  P1
```



```
        SETB    INTR            ;; make it input
BACK:   CLR     WR              ;; pulse WR by lowering first
        SETB    WR              ;; conv starts on rising edge
HERE:   JB      INTR, HERE      ;; poll for /INR to go low
        CLR     RD              ;; assert output enable
        MOV     A, MYDATA       ;; read conv value from port
        ACALL   CONVERSION      ;; Chap.6 routine to ASCII
        ACALL   DATA_DISPLAY    ;; Chap. 12 routine to display
        SETB    RD              ;; de-assert output enable
        SJMP    BACK            ;; do it again
```

Interfacing ADC

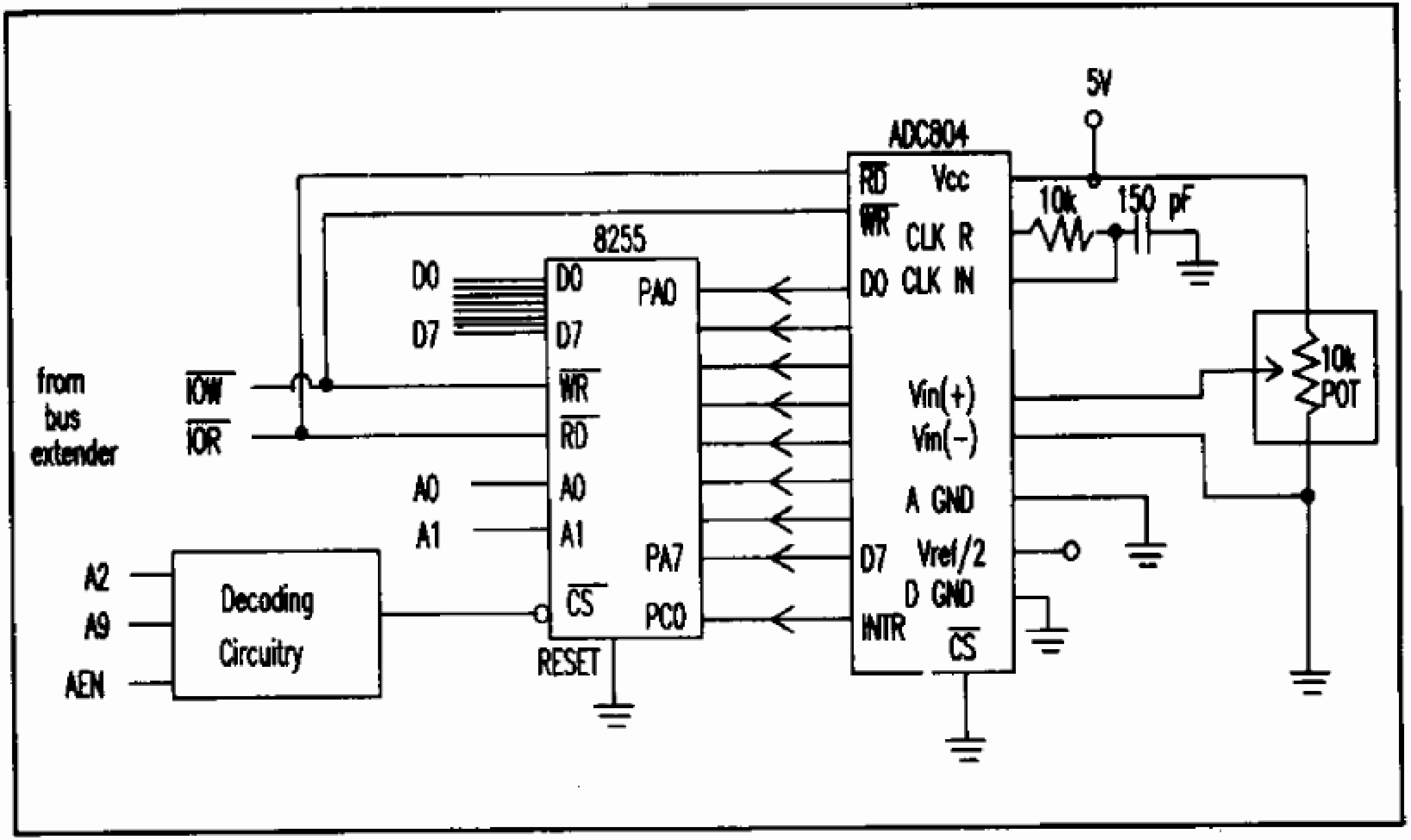
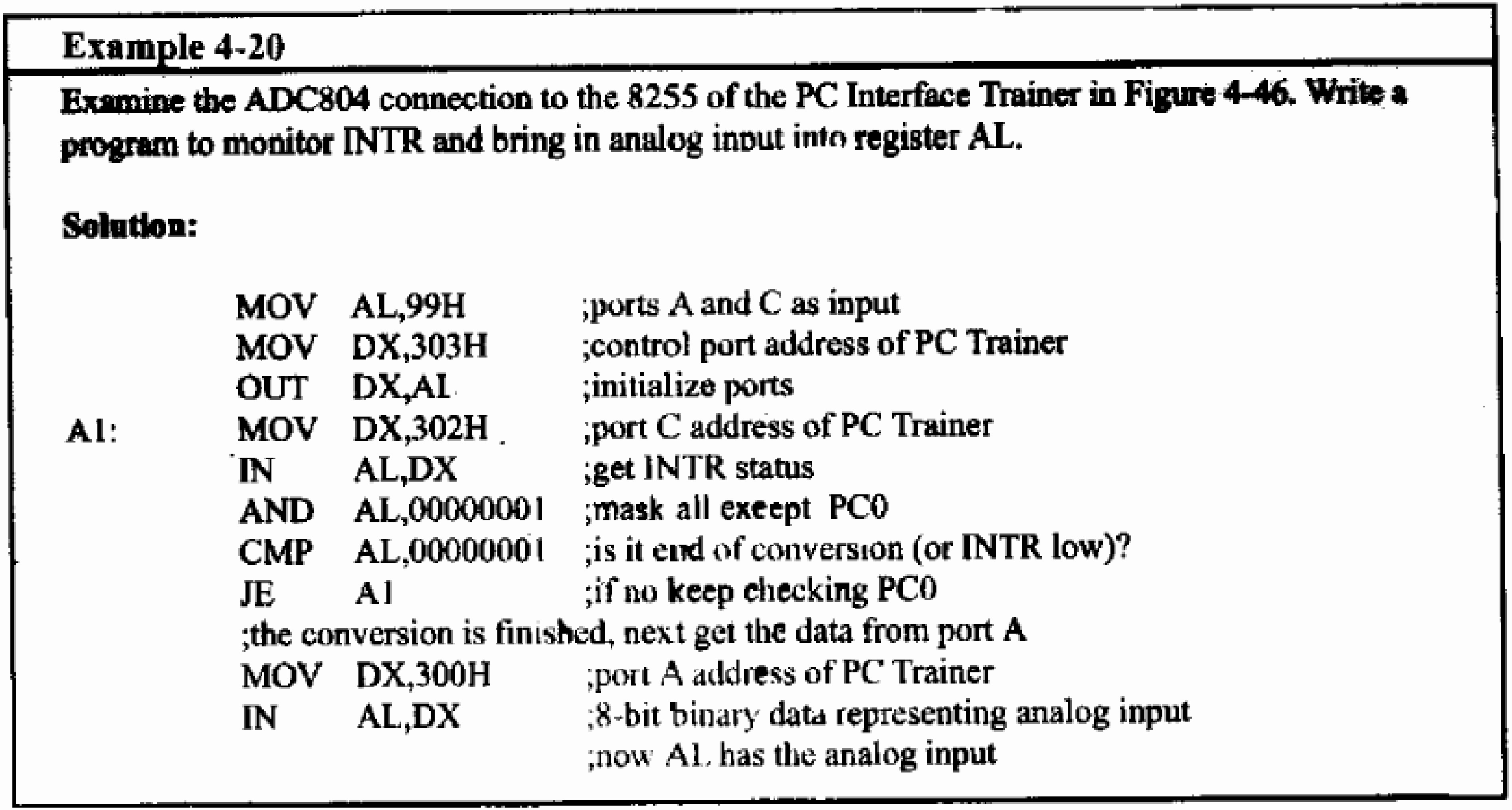


Figure 4-46. 8255 Connection to ADC804

Example (với 80x86)



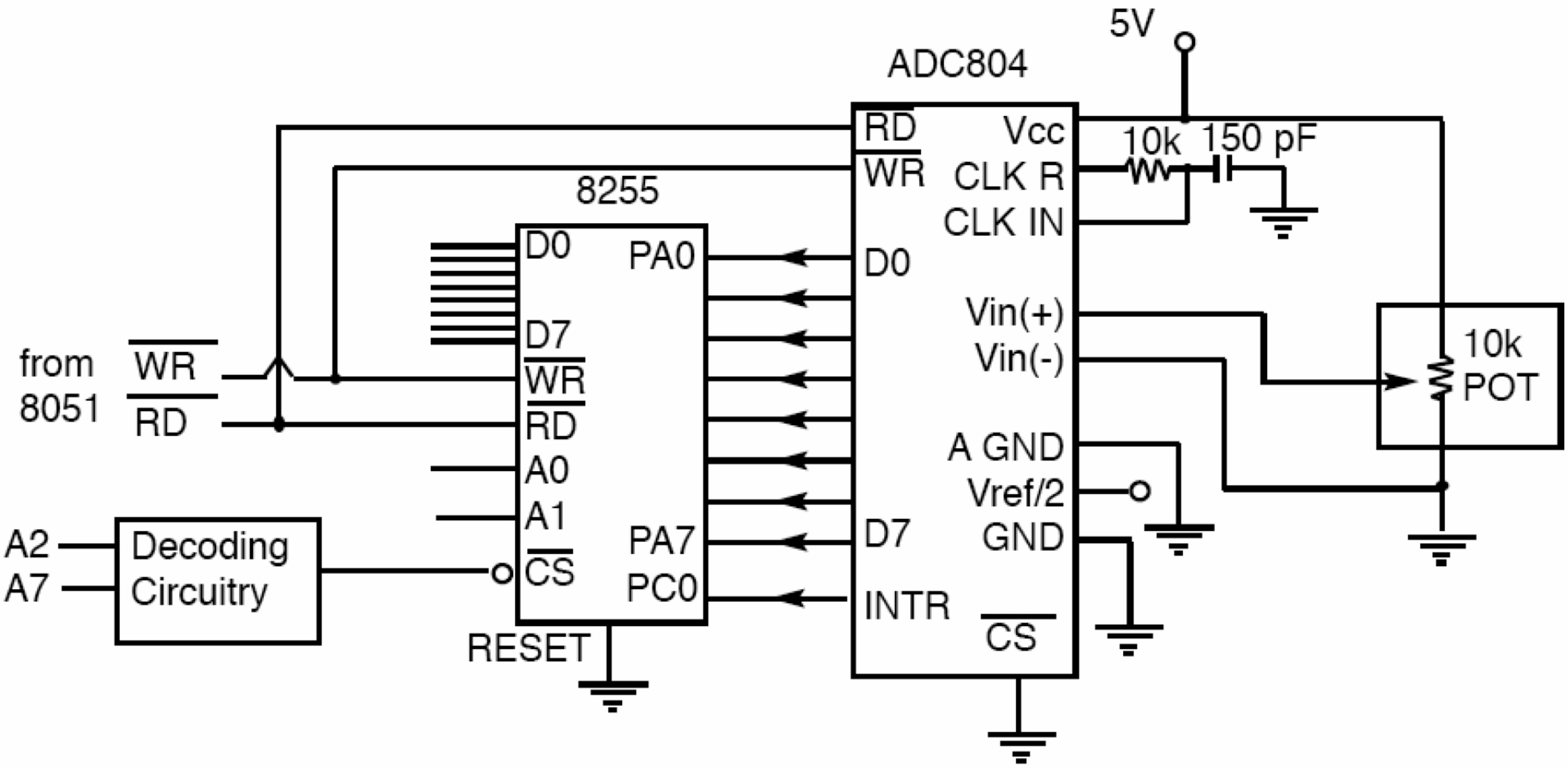
Example 4-20

Examine the ADC804 connection to the 8255 of the PC Interface Trainer in Figure 4-46. Write a program to monitor INTR and bring in analog input into register AL.

Solution:

```
        MOV   AL,99H         ;ports A and C as input
        MOV   DX,303H        ;control port address of PC Trainer
        OUT   DX,AL          ;initialize ports
A1:     MOV   DX,302H        ;port C address of PC Trainer
        IN    AL,DX          ;get INTR status
        AND   AL,00000001    ;mask all exeept  PC0
        CMP   AL,00000001    ;is it end of conversion (or INTR low)?
        JE    A1             ;if no keep checking PC0
        ;the conversion is finished, next get the data from port A
        MOV   DX,300H        ;port A address of PC Trainer
        IN    AL,DX          ;8-bit binary data representing analog input
                             ;now AL has the analog input
```

8051 giao tiếp với ADC



Temperature Sensor

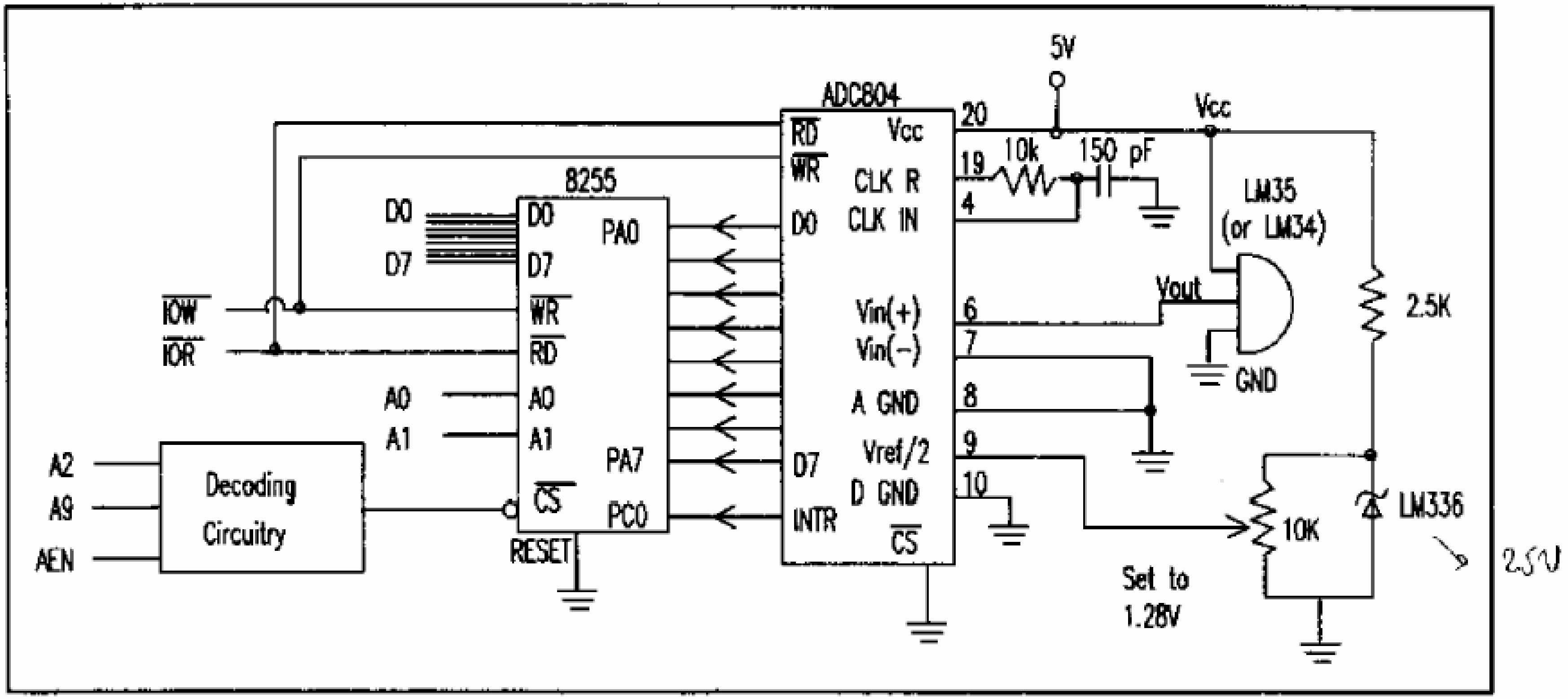
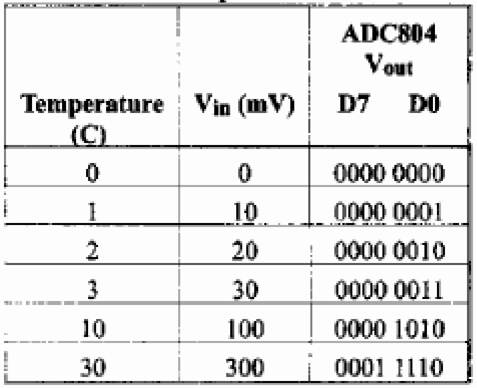


Figure 4-47. Temperature Sensor Connection to ADC804

Table 4-20: Temp. vs. VOUT of the 804



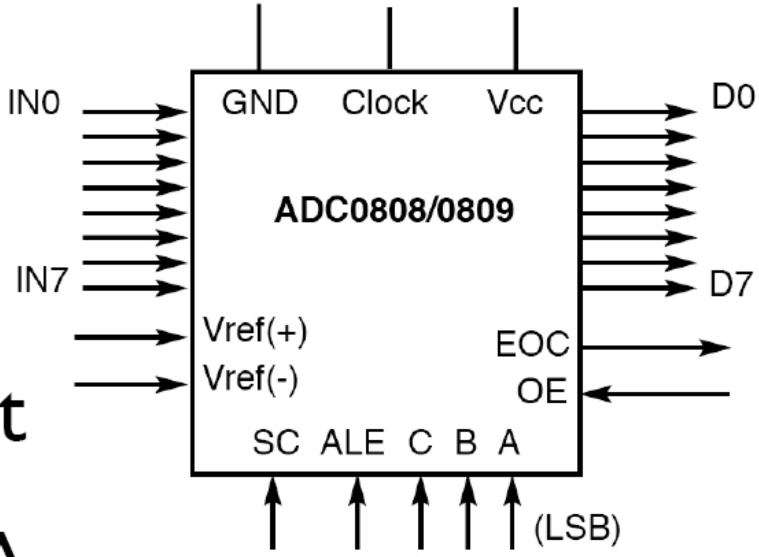
Temperature (C)	V_{in} (mV)	ADC804 V_{out} D7 D0
0	0	0000 0000
1	10	0000 0001
2	20	0000 0010
3	30	0000 0011
10	100	0000 1010
30	300	0001 !110

ADC0808/0809: multi-(analog)-channel

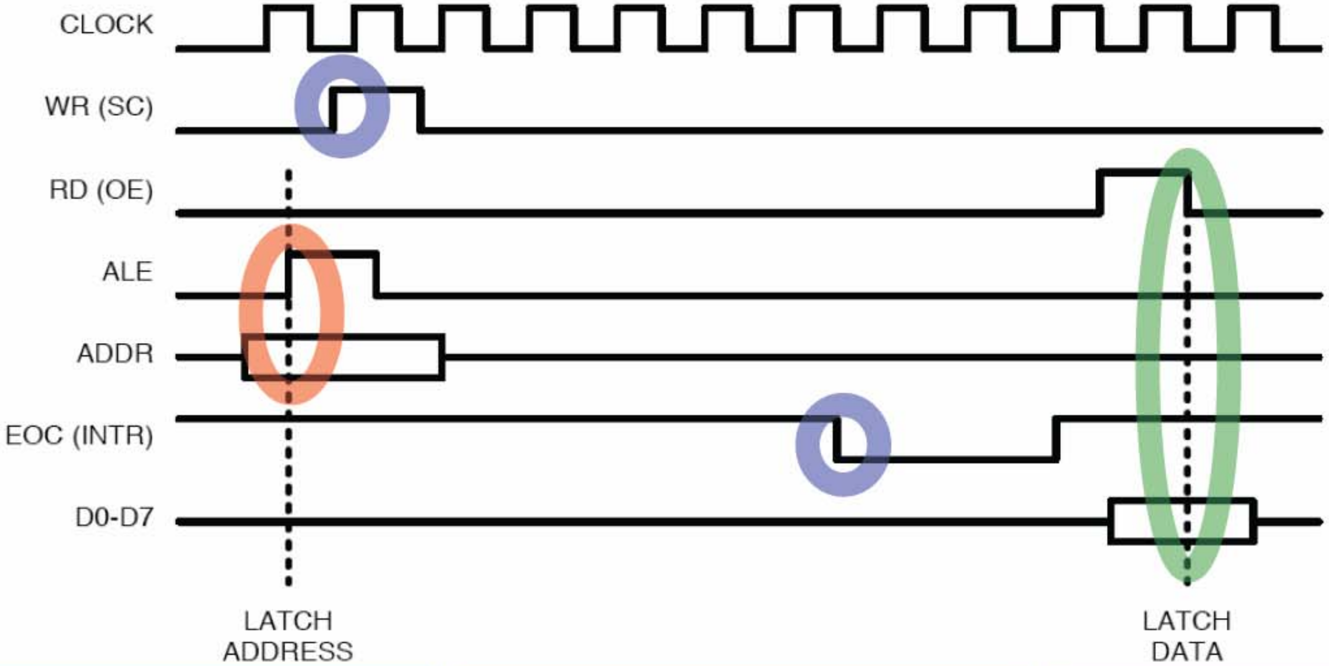
- 8 analog input lines
  - Selected by 3-bit, internally share 1 ADC
  - Conceptually, analog multiplexor on input
  - ALE latches the address
- 8-bit output port
  - similar to the single-channel ADC

Pin interface on ADC0808/0809

- IN0..IN7: analog input channels
- SC, EOC: (=WR, INTR)
  start conv, end-of-conv
- OE: (=RD) output enable
- C B A : 3-bit channel select
- ALE: clock for latching CBA
- $V_{ref}(+), V_{ref}(-)$: max and "gnd"



Timing Diagram for the ADC0809



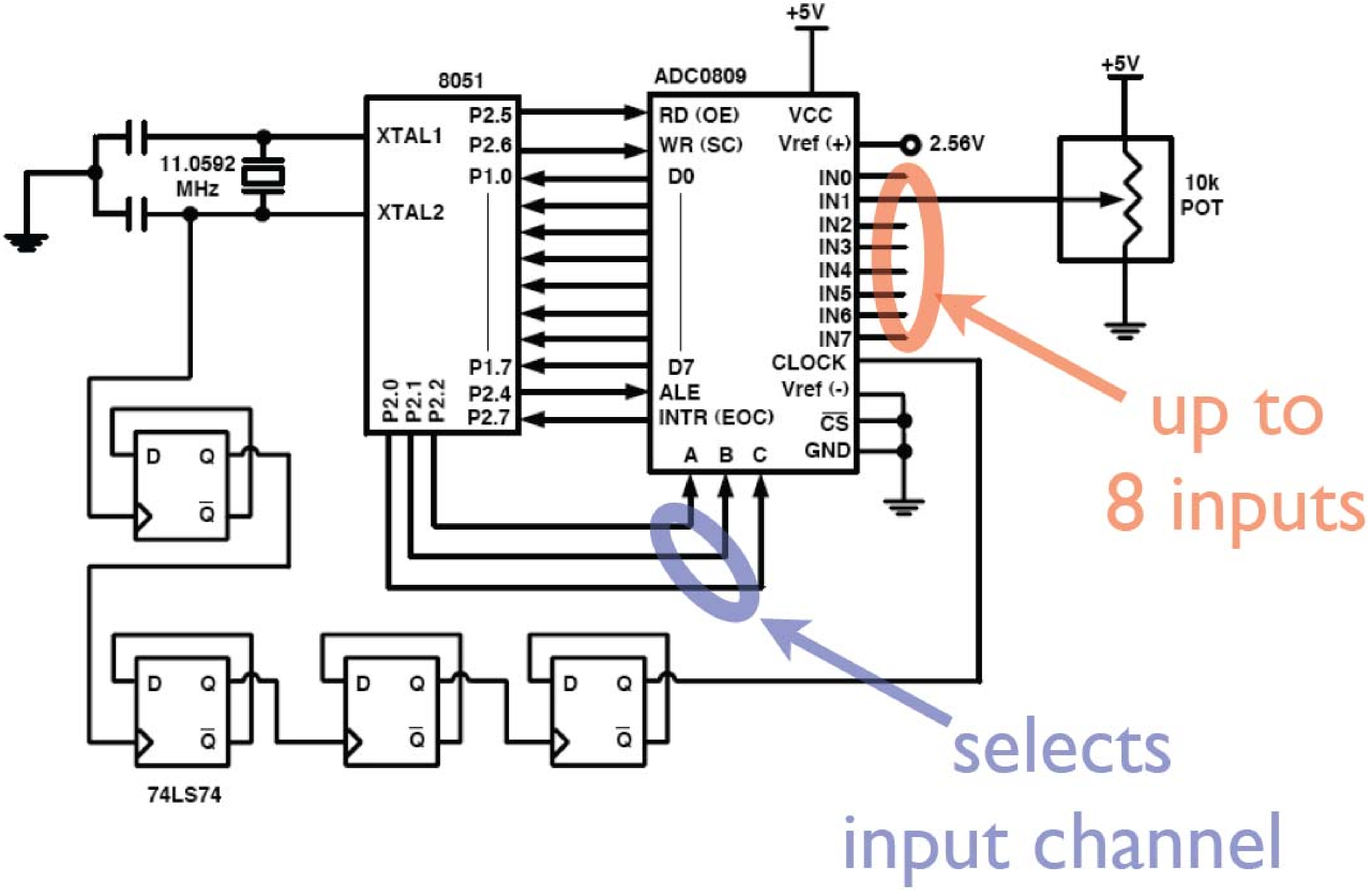
1. set ADDR, pulse ALE

2. pulse SC, wait for EOC

3. read data during OE

ADDR is formed by C B A

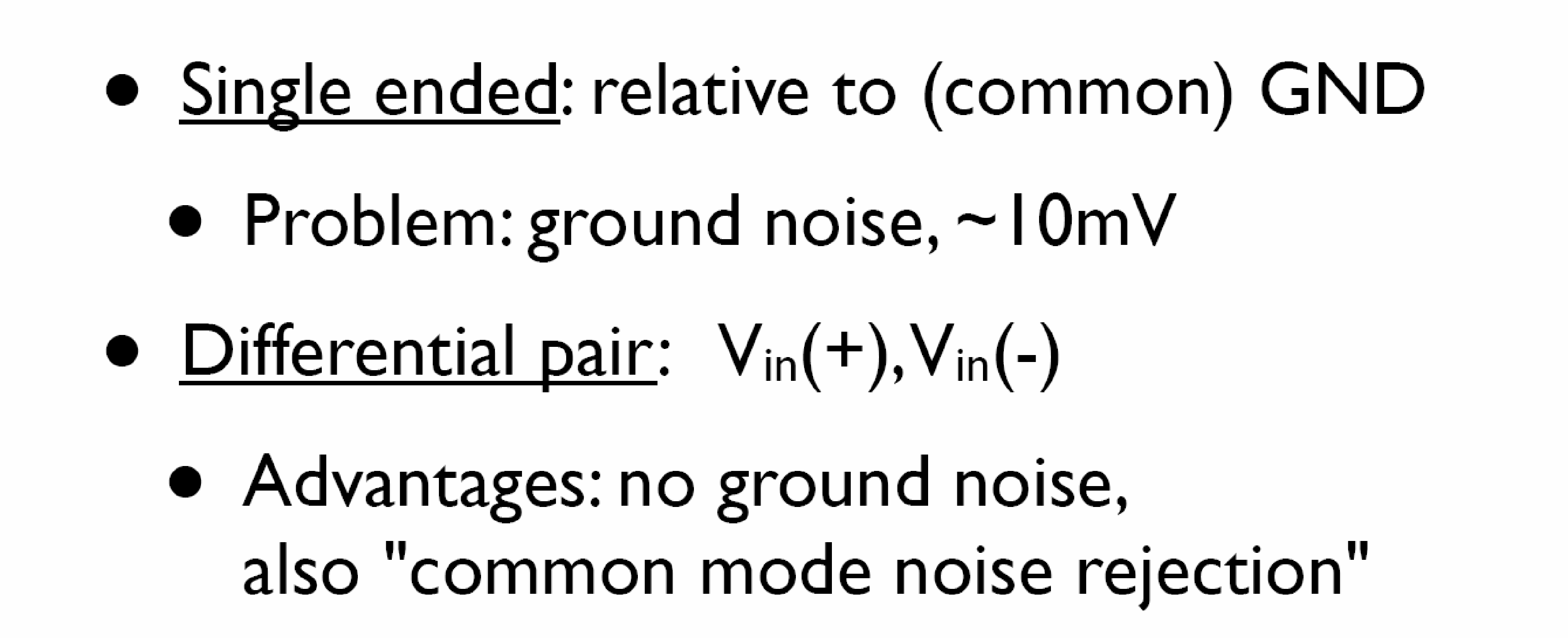
Schematic for 8051 connected to ADC0809 up to 8 inputs selects input



Reference voltages

- $V_{ref}/2$ (e.g., *ADC0804*): half the max voltage
  e.g., $V_{ref}/2=2.5V$ => max voltage = 5.0V
  => each step is 5.0V / 2^8 = 19.53mV
  - Assumes same GND as signal source
- $V_{ref}(+)$, $V_{ref}(-)$ (e.g., *ADC0808/0809*)
  - $V_{ref}(-)$ serves as GND for signal source
    possible to tie $V_{ref}(-)$ to same GND
  - voltage step = $(V_{ref}(+)-V_{ref}(-))$ / 2^8

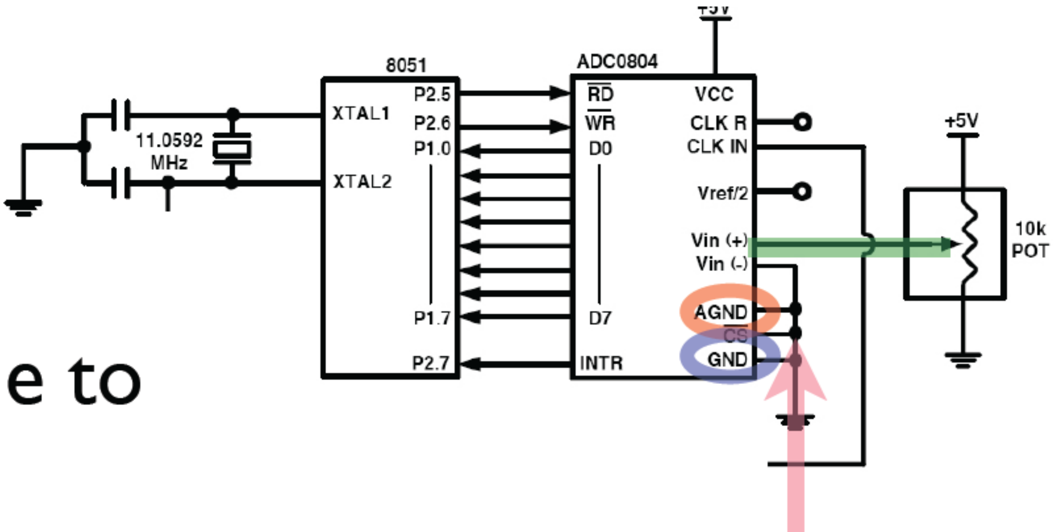
Single-ended vs Differential Pair input



- Single ended: relative to (common) GND
  - Problem: ground noise, ~10mV
- Differential pair: $V_{in}(+), V_{in}(-)$
  - Advantages: no ground noise,
    also "common mode noise rejection"

Digital vs Analog Ground

- AGND: analog
- GND: digital
- Digital => noisy due to frequent switching
- Analog: sensitive to noise
- Solution: separate AGND & GND
- Keep analog signal lines short

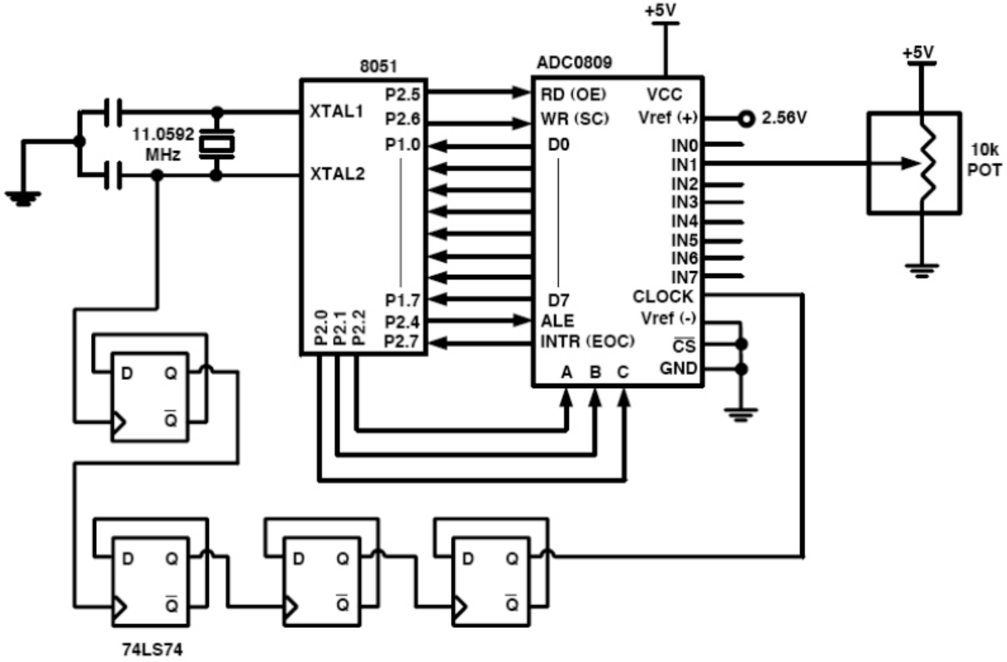


bad idea to tie them together!

Assembly for ADC0809

```
ALE        BIT P2.4
OE         BIT P2.5
SC         BIT P2.6
EOC        BIT P2.7
ADDR_A     BIT P2.0
ADDR_B     BIT P2.1
ADDR_C     BIT P2.2
MYDATA     EQU P1

ORG        0H
SETB       EOC ;; inp
CLR        ALE
CLR        SC
CLR        OE
```



Assembly for ADC0809 (2/2)

```
BACK:   CLR     ADDR_C
        CLR     ADDR_B      ;; sel ch=1
        SETB    ADDR_A
        ACALL   DELAY
        SETB    ALE         ;; latch addr
        ACALL   DELAY
        SETB    SC          ;; start
        ACALL   DELAY
        CLR     ALE
        CLR     SC
```

```
HERE:   JB      EOC, HERE
HERE1:  JNB     EOC, HERE1
        SETB    OE
        ACALL   DELAY
        MOV     A, MYDATA
        CLR     OE
        ACALL   CONVERSION
        ACALL   DATA_DISPLAY
        SJMP    BACK
```

Printer Connection

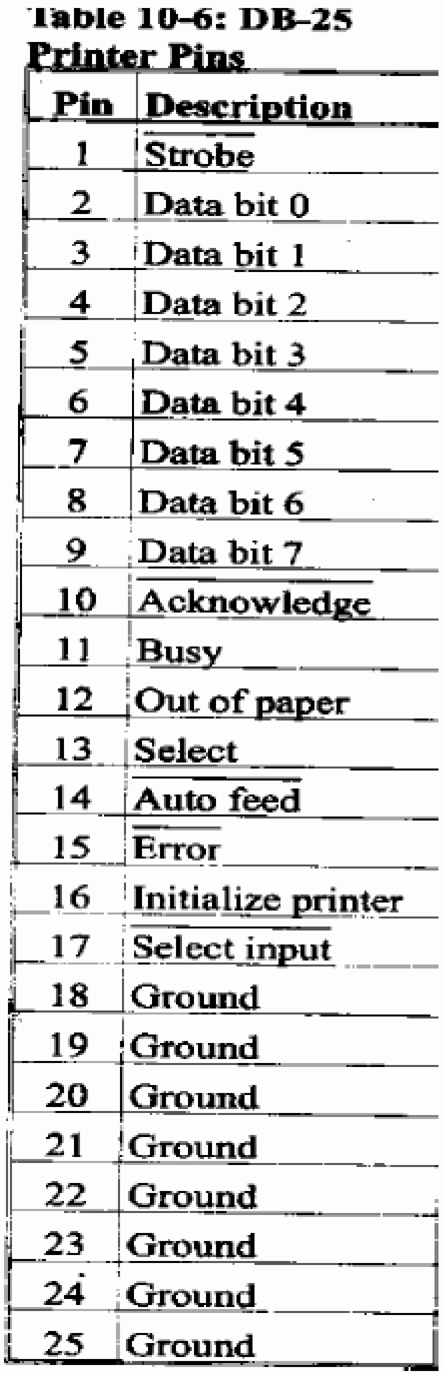


Table 10-6: DB-25 Printer Pins

Pin	Description
1	Strobe
2	Data bit 0
3	Data bit 1
4	Data bit 2
5	Data bit 3
6	Data bit 4
7	Data bit 5
8	Data bit 6
9	Data bit 7
10	Acknowledge
11	Busy
12	Out of paper
13	Select
14	Auto feed
15	Error
16	Initialize printer
17	Select input
18	Ground
19	Ground
20	Ground
21	Ground
22	Ground
23	Ground
24	Ground
25	Ground

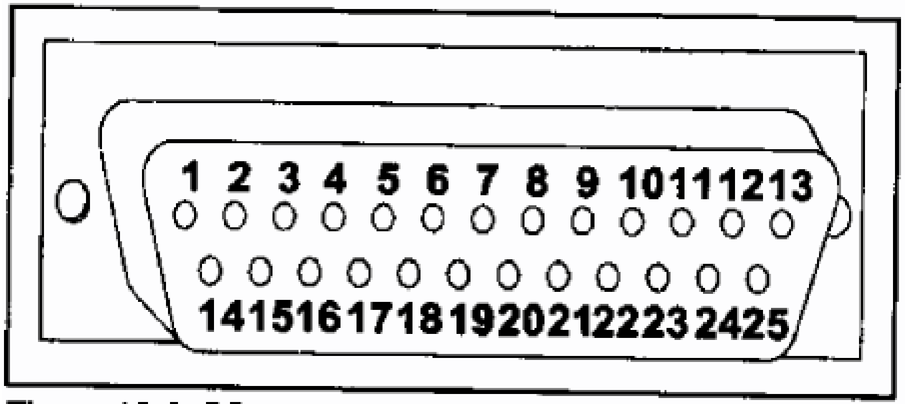


Figure 10-8. DB-25P (Male) Printer Conhector
(Reprinted by permission from "IBM Technical Reference" c. 1988
by International Business Machines Corporation)

IO Base Address for LPT

- -d 0040:0008
- LPT: 03BC, 0378, 0278
- 3 I/O ports
  - LPT data lines
  - LPT status lines
  - LPT control lines

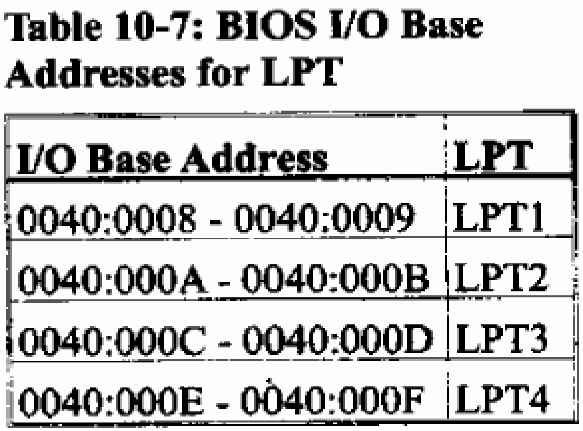


Table 10-7: BIOS I/O Base Addresses for LPT

I/O Base Address	LPT
0040:0008 - 0040:0009	LPT1
0040:000A - 0040:000B	LPT2
0040:000C - 0040:000D	LPT3
0040:000E - 0040:000F	LPT4

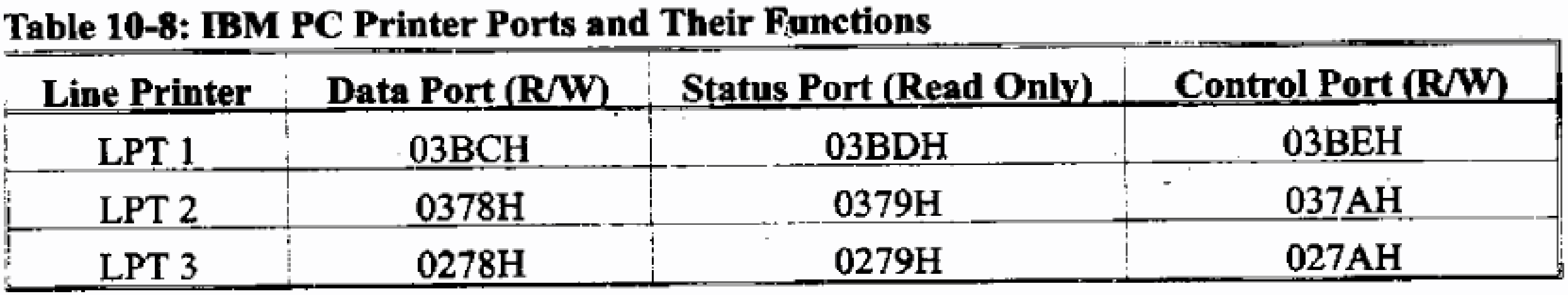
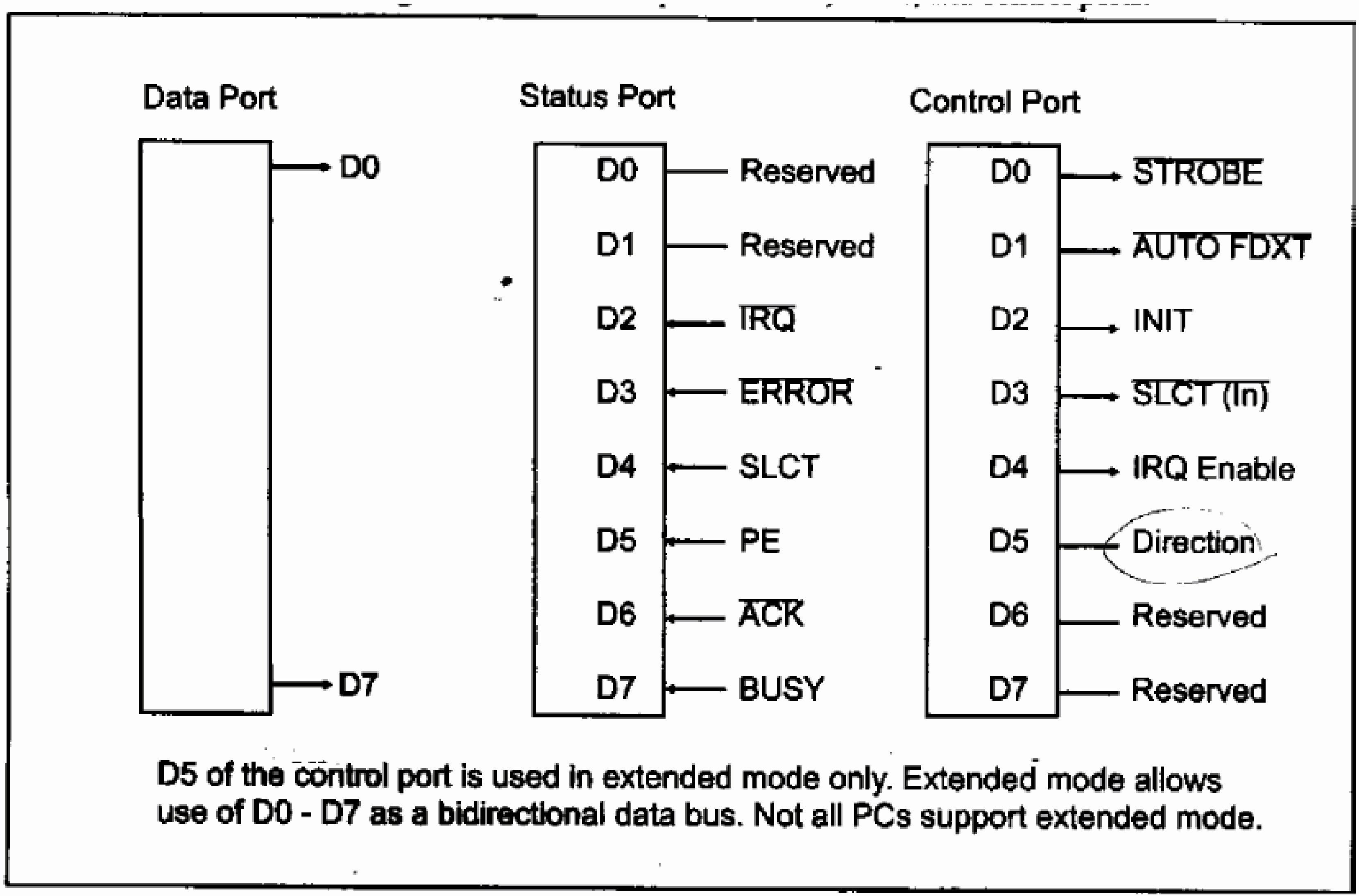


Table 10-8: IBM PC Printer Ports and Their Functions

Line Printer	Data Port (R/W)	Status Port (Read Only)	Control Port (R/W)
LPT 1	03BCH	03BDH	03BEH
LPT 2	0378H	0379H	037AH
LPT 3	0278H	0279H	027AH

Printer’s Ports


Data Port
D0
D7
Status Port
D0 Reserved
D1 Reserved
D2 $\overline{\text{IRQ}}$
D3 $\overline{\text{ERROR}}$
D4 SLCT
D5 PE
D6 $\overline{\text{ACK}}$
D7 BUSY
Control Port
D0 $\overline{\text{STROBE}}$
D1 $\overline{\text{AUTO FDXT}}$
D2 INIT
D3 $\overline{\text{SLCT (In)}}$
D4 IRQ Enable
D5 Direction
D6 Reserved
D7 Reserved

D5 of the control port is used in extended mode only. Extended mode allows use of D0 - D7 as a bidirectional data bus. Not all PCs support extended mode.

Figure 10-9. Printer's Data, Status, and Control Ports

Useful Links

- http://www.kmitl.ac.th/~kswichit/
- http://www.8052.com/codelib.phtml
- http://www.circuits-city.com/
- http://www.ikalogic.com/cat_microcontrollers.php