

AVR Timer/Counter 0

- 8 Bit Up Counter
 - counts from 0 to 255 (0xFF), then loops to 0
 - Internal or External Clock source
- Prescaler
- Output capture through OC0, i.e. PB3, pin 4
- Interrupt on Overflow
 - Transition from 255 to 0 can trigger interrupt if desired

AVR Timer/Counter 0

OC0, Output Compare Match output:

Whenever TCNT0 equals OCR0 (Output Compare Register 0), the comparator signals a match

The PB3 pin can serve as an external output for the Timer/Counter0 Compare Match. The PB3 pin has to be configured as an output

AVR Timer/Counter 1

- 16 Bit
- Dual Comparators A,B (output compares)
- Up Counter
- Interrupt on:
 - Overflow
 - Compare A/B
 - Input Capture of external event on ICP pin.
- Can also act as an 8, 9 or 10 bit PWM Up-Down Counter.

AVR Timer/Counter 1

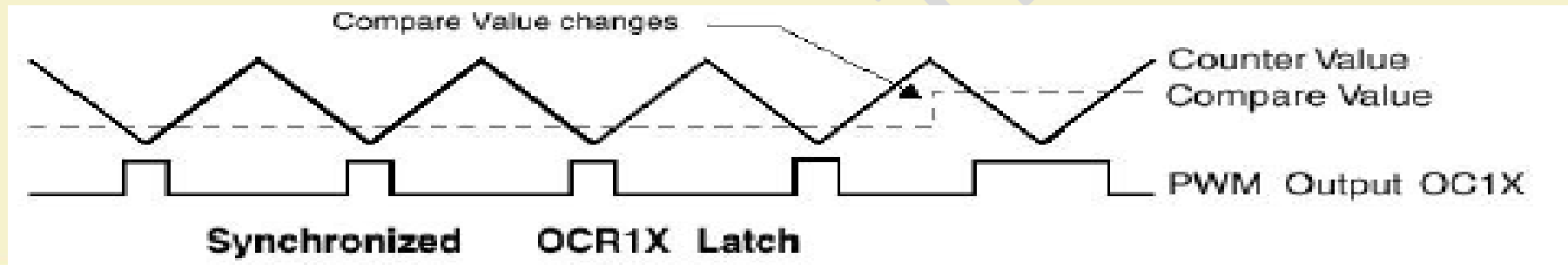
- The Input Capture unit of Timer/Counter captures external events and gives them a time-stamp indicating time of occurrence.
- The external signal indicating an event, or multiple events, can be applied via the ICP1 pin or alternatively, via the Analog Comparator unit.
- The time-stamps can then be used to calculate frequency, duty-cycle, and other features of the signal applied.
- Alternatively the time-stamps can be used for creating a log of the events.

Timer 1 and Output Compare

- The AVR has two *output compares* (OCR1A/B)
 - OCR1A/B are 16-bit registers
 - When the value of OCR1A/OCR1B matches that of Timer1:
 - A user-defined action can take place on the OC1A/OC1B pin (set/clear/inv)
i.e., PD5 / PD4 need to be set as output
 - An interrupt can be triggered
 - Timer1 can be cleared to zero
 - Once set up, output compares operate continuously without software intervention
 - Great for:
 - Precise recurring timing
 - Frequency/Tone generation (maybe sound effects)
 - All kinds of digital signal generation
 - Infrared communications
 - Software-driven serial ports

Timer 1 and PWM

- Pulse-Width Modulation
 - Useful for using digital circuits to achieve analog-like control of motors, LEDs, etc
 - Timer 1 has two channels of PWM output on OCR1A and OCR1B



Timer Control

- Timer 0:
 - Control Register (TCCR0) for clock selection, external clock or internal clock, prescaler etc.
 - Timer/Counter0 (TCNT0) holding counter value
- Timer 1:
 - Control Register A & B (TCCR1A/B)
 - Input Capture Register (ICR1)
 - Timer/Counter1 Output Compare Register A and B (OCR1A/B)
 - Timer/Counter1 (TCNT1)
- Timer Interrupt Registers (Mask and Flag Registers) are Common to Both Timers

AVR Timer/Counter Sources

- Shut Off
- CPU frequency divided by 1,8,64,256,1024
- At 8MHz that's: 1/8us, 1us, 8us, 32us, 128us
- External Input (rising or falling).

CS02	CS01	CS00	Description
0	0	0	Stop, the Timer/Counter0 is stopped.
0	0	1	CK
0	1	0	CK/8
0	1	1	CK/64
1	0	0	CK/256
1	0	1	CK/1024
1	1	0	External Pin T0, falling edge
1	1	1	External Pin T0, rising edge

Interrupts

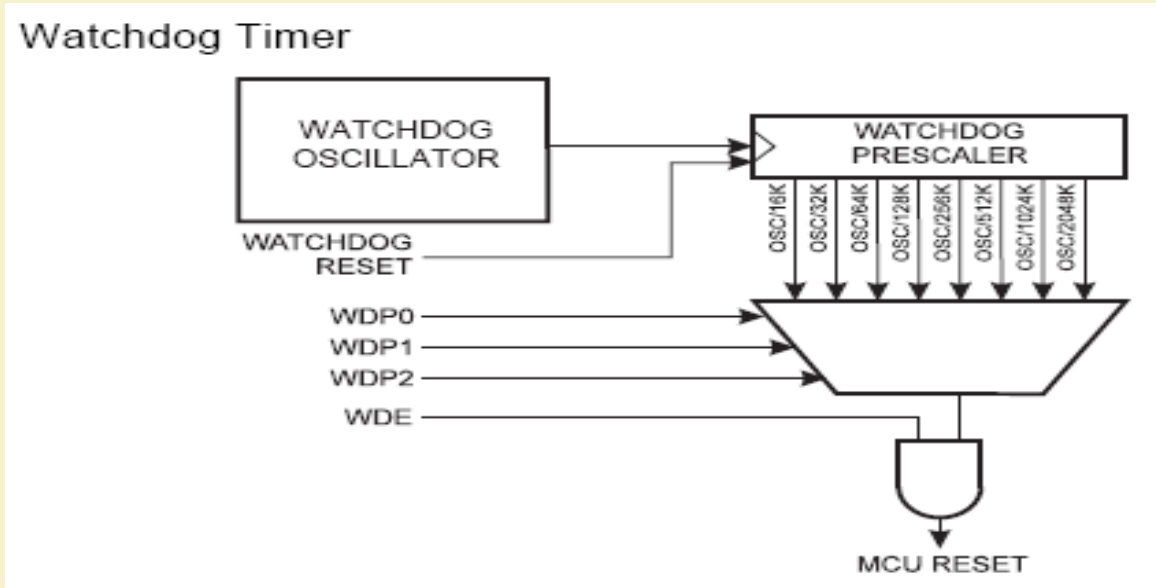
- Interrupts halt normal code execution in order to go do something more important or time sensitive
- Interrupt “Handlers”
 - Using the Interrupt Vectors
- Interrupts are used for:
 - RESET
 - Timers and Time-Critical Code
 - Hardware signaling
 - “I’m done”
 - “Something’s happened that you want to know about”
 - “I have something for you”

Table 18. Reset and Interrupt Vectors

Vector No.	Program Address ⁽²⁾	Source	Interrupt Definition
1	\$000 ⁽¹⁾	RESET	External Pin, Power-on Reset, Brown-out Reset, Watchdog Reset, and JTAG AVR Reset
2	\$002	INT0	External Interrupt Request 0
3	\$004	INT1	External Interrupt Request 1
4	\$006	TIMER2 COMP	Timer/Counter2 Compare Match
5	\$008	TIMER2 OVF	Timer/Counter2 Overflow
6	\$00A	TIMER1 CAPT	Timer/Counter1 Capture Event
7	\$00C	TIMER1 COMPA	Timer/Counter1 Compare Match A
8	\$00E	TIMER1 COMPB	Timer/Counter1 Compare Match B
9	\$010	TIMER1 OVF	Timer/Counter1 Overflow
10	\$012	TIMER0 OVF	Timer/Counter0 Overflow
11	\$014	SPI, STC	Serial Transfer Complete
12	\$016	USART, RXC	USART, Rx Complete
13	\$018	USART, UDRE	USART Data Register Empty
14	\$01A	USART, TXC	USART, Tx Complete
15	\$01C	ADC	ADC Conversion Complete
16	\$01E	EE_RDY	EEPROM Ready
17	\$020	ANA_COMP	Analog Comparator
18	\$022	TWI	Two-wire Serial Interface
19	\$024	INT2	External Interrupt Request 2
20	\$026	TIMER0 COMP	Timer/Counter0 Compare Match
21	\$028	SPM_RDY	Store Program Memory Ready

Watchdog Timer: reset the MCU

The Watchdog Timer is clocked from a separate On-chip Oscillator which runs at 1 MHz



Interfacing

Santanu Chattopadhyay

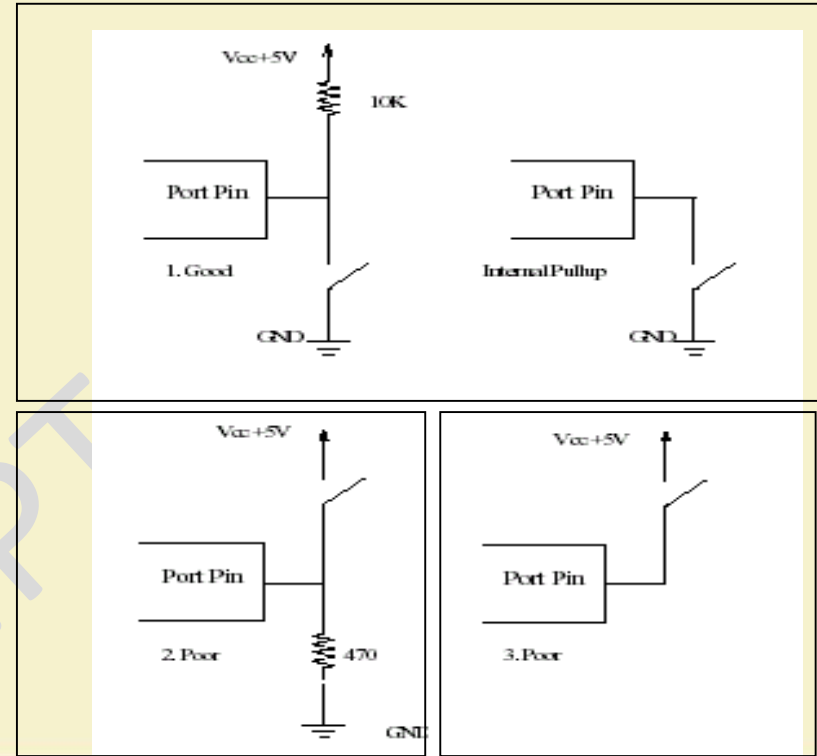
Electronics and Electrical Communication Engineering

I/O devices (Peripherals)

- Examples: switches, LED, LCD, printers, keyboard, keypad
- **Interface** chips
 - are needed to resolve the speed problem
 - **synchronizes** data transfer between CPU and I/O device
- Connection of Interface and CPU
 - Data pins are connected to CPU data bus
 - I/O port pins are connected to I/O device
- CPU may be connected to **multiple** interface
- IO ports are simplest interface

8051 - Switch On I/O Ports

- Case-1:
 - Gives a logic 0 on switch close
 - Current is 0.5ma on switch close
- Case-2:
 - Gives a logic 1 on switch close
 - High current on switch close
- Case-3:
 - Can damage port if 0 is output

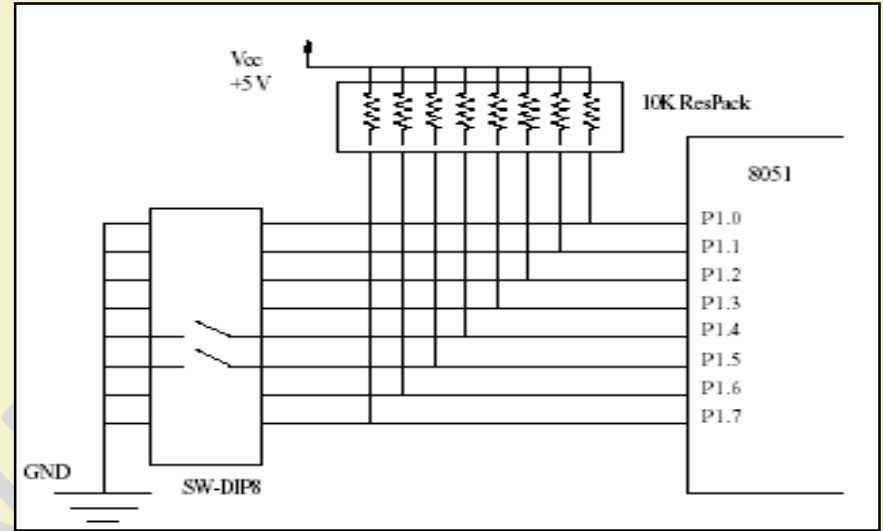


Simple input devices

- DIP switches usually have 8 switches
- Use the case-1 from previous page
- Sequence of instructions to read is:

MOVP1,#FFH

MOVA,P1,



Bouncing contacts

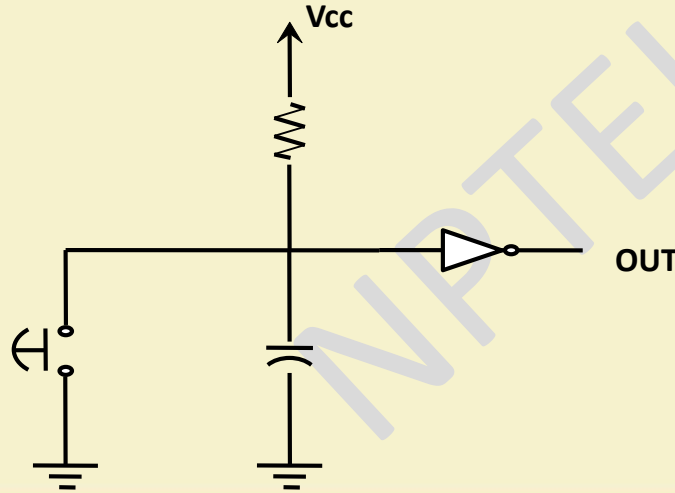
- Contact:
 - Push-button switches
 - Toggle switches
 - Electromechanical relays
- Make and break Contact normally open switch
- The effect is called "contact bounce" or, in a switch, "switch bounce".



- If used as edge-triggered input (as INT0), several interrupt is accorded

Hardware Solution

- An RC time constant to suppress the bounce
- The time constant has to be larger than the switch bounce



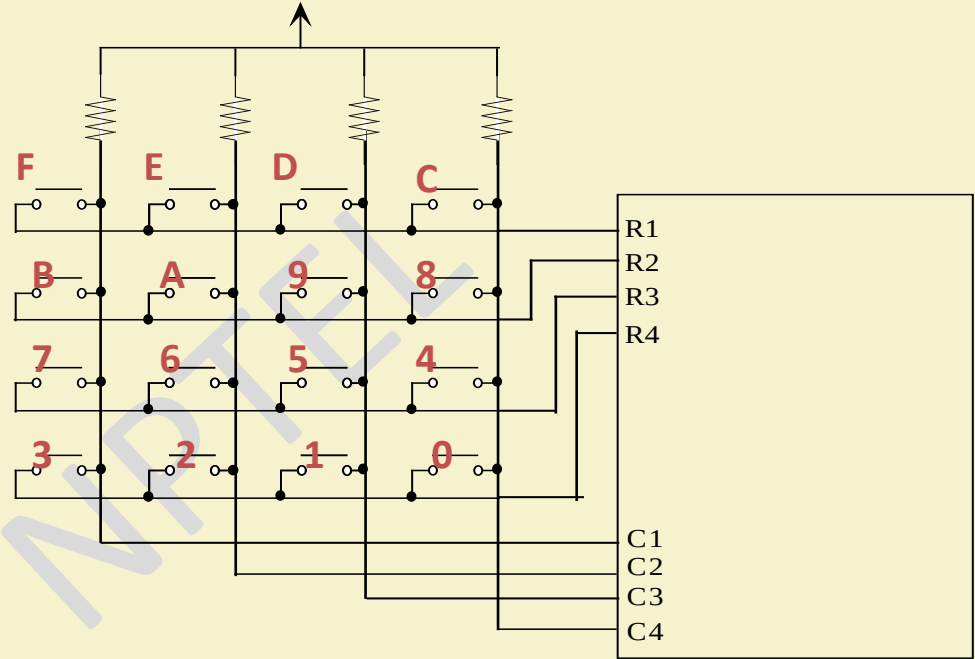
Software Solution

- Read the new state of switch N time
- Wait-and-see technique
 - When the input drops
 - an “appropriate” delay is executed (10 ms)
 - then the value of the line is checked again to make sure the line has stopped bouncing

Interfacing a Keypad

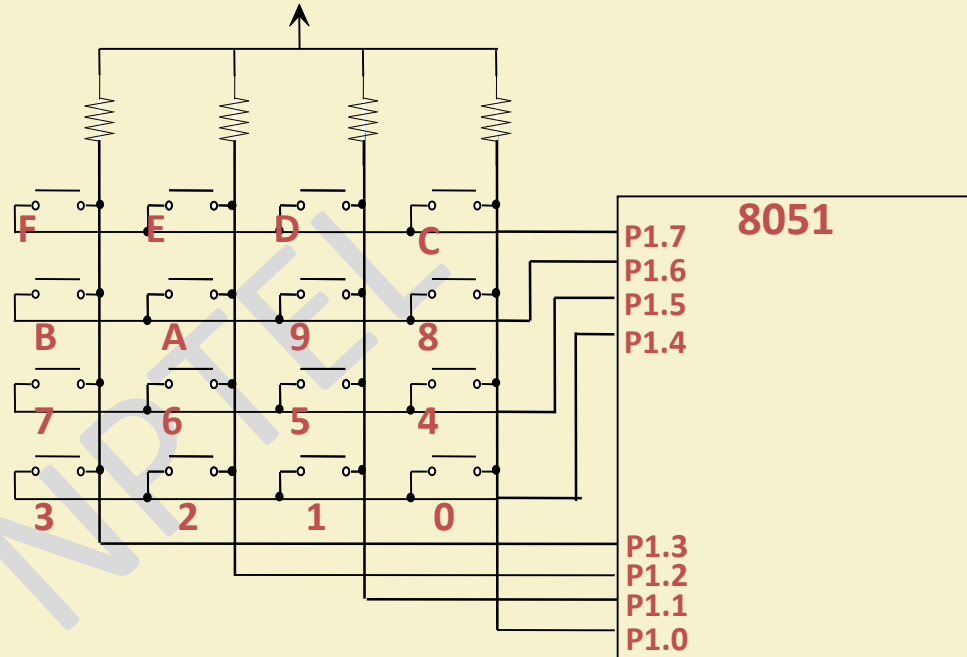
16 keys arranged as a 4X4 matrix

- All key released?
- Place a 0 on R1 port
- Read C port
- If there is a 0 bit then the button at the column/row intersection has been pressed.
- Otherwise, try next row
- Repeat constantly



Interfacing a Keypad

```
RelS:    mov     P1, #0FH
          mov     A, P1
          ani     A, #0FH
          cpi     A, #0FH
          jne     RelS
scan:     mov     P1, #EFH
          jnb     P1.0, db_0
scan1:    jnb     P1.1, db_1
scan2:    jnb     P1.2, db_2
scan3:    jnb     P1.3, db_3
scan4:    mov     P1, #DFH
          jnb     P1.0, db_4
... .
... .
... .
```



Interfacing a Keypad

```
db_0:      lcall    wt_10ms
           jb       P1.0, scan1
           mov      A, #0
           ljmp     get_code

db_1:      lcall    wt_10ms
           jb       P1.1, scan2
           mov      A, #1
           ljmp     get_code

... .
...
... .

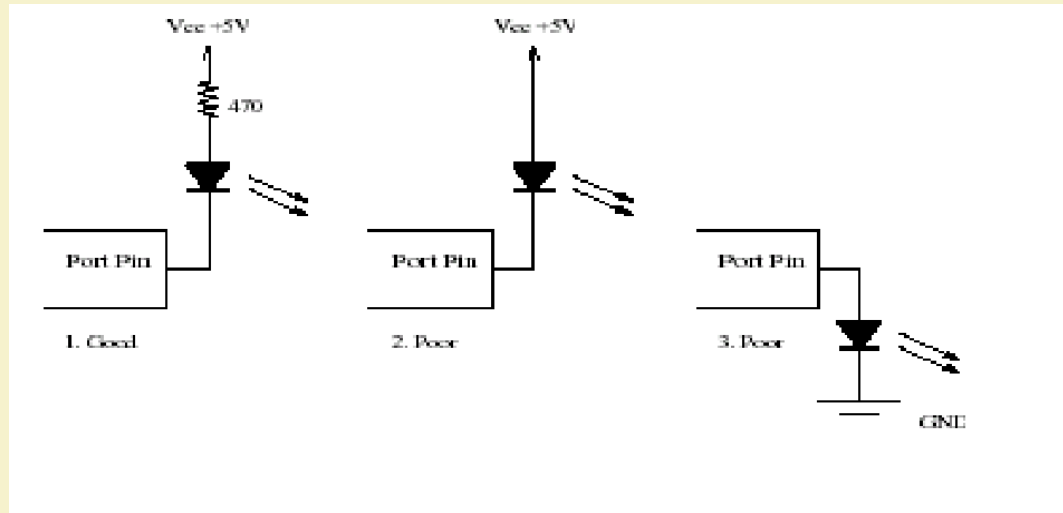
get_code:  mov      DPTR, #key_tab
           movc     A, @A+DPTR
           ljmp     Rel

key_tab:   db       '0123456789ABCDEF'

END
```

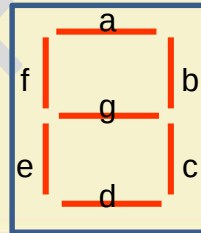
Simple output devices

- Case-1
 - LED is ON if output=zero
 - Most LEDs drop 1.7 volts and need about 10ma
 - Current is $(5-1.7)/470$
- Case-2
 - Too much current
 - Failure of Port or LED
- Case-3
 - Not enough drive (1ma)
 - LED is too dim



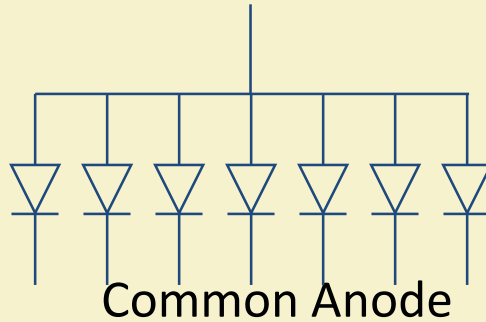
The 7-Segment Display

- 7 LEDs arranged to form the number 8.
 - By turning on and off (LEDs), different combinations can be produced.
 - useful for displaying the digits 0 through 9, and some characters.

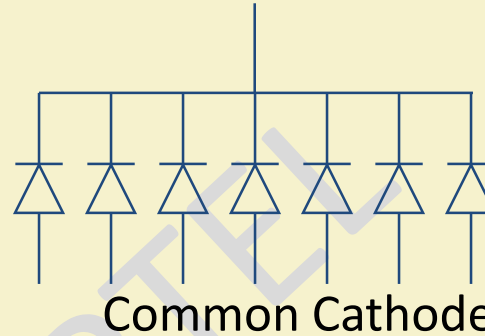


The 7-segment Display (Cont.)

- 7-segment displays come in 2 configurations:



Connect **cathode** to the **output**

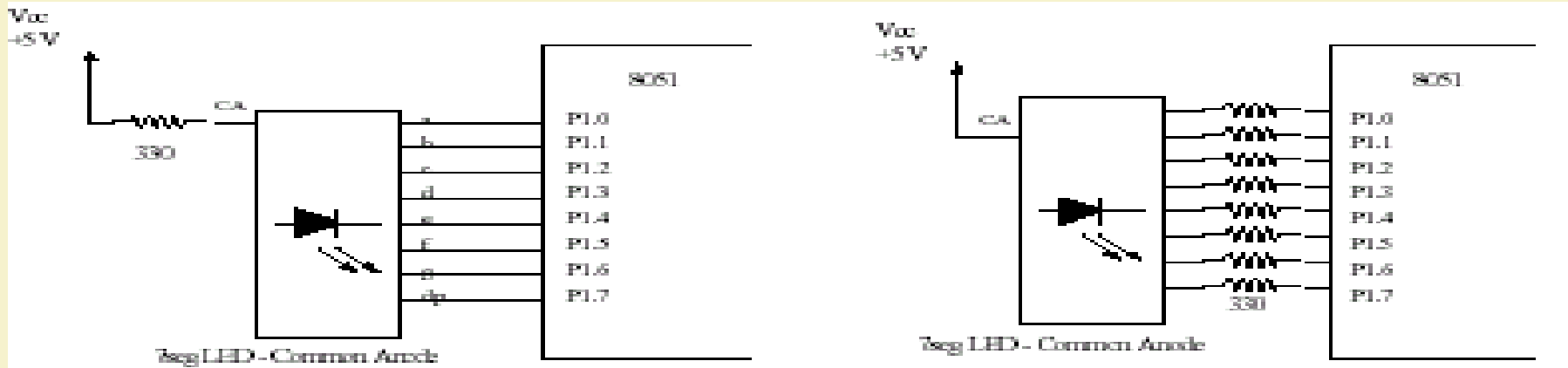


Connect **cathode** to the **output**

- Therefore, the **common anode** variety would be **better** for our interfacing needs.

Interfacing a 7-segment display

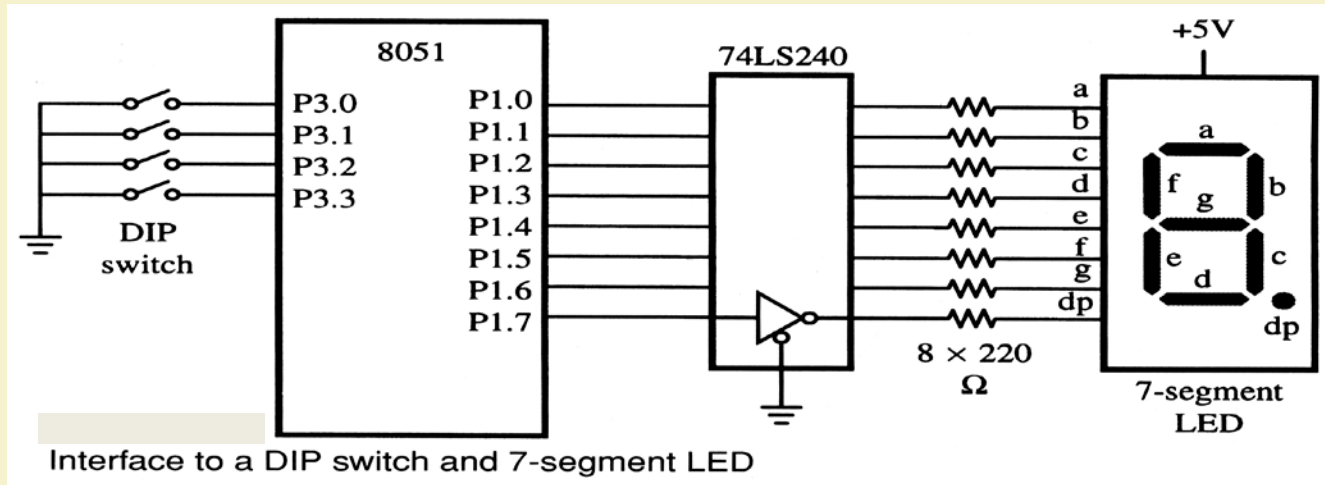
- A **resistor** will be **needed** to control the current
- This leaves two possibilities:



- Case 2 would be more appropriate
- Case 1 will produce different **brightness** depending on the number of LEDs turned on.

Use of current buffer

- ❑ Interfacing to a DIP switch and 7-segment display
- ❑ Output a '1' to ON a segment
- ❑ We can use 74244 to common cathode 7_seg



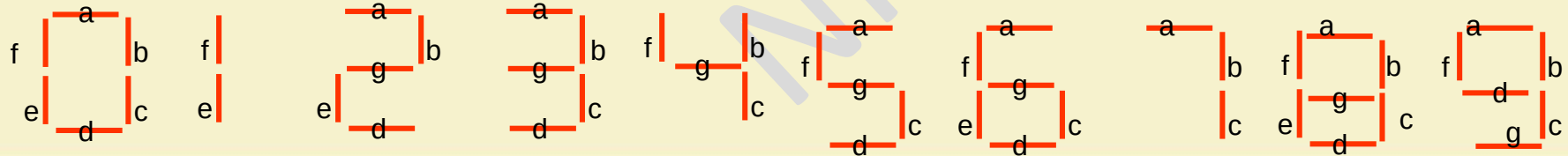
BCD to 7_Seg lookup table

```

mov     a, p3
anl     a, 0fh
get_code: mov   DPTR, #7s_tab
movc    A, @A+DPTR
mov     p1, a

7s_tab: db      3fh, 30h, 5bh, 4fh, 66h
               db      6dh, 7dh, 07h, 7fh, 6fh
               END
    
```

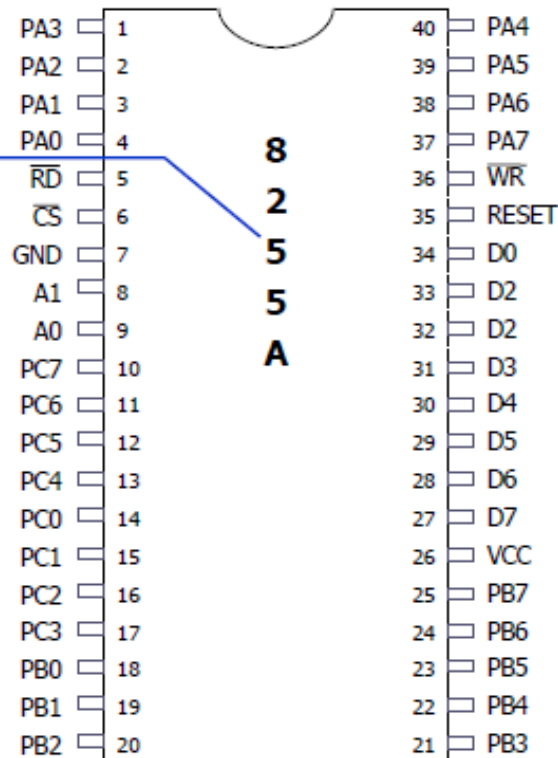
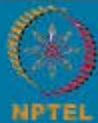
BCD	p g f e d c b a 7_seg	hex
0000	0 0 1 1 1 1 1 1	3f
0001	0 0 1 1 0 0 0 0	30
0010	0 1 0 1 1 0 1 1	5b
0011	0 1 0 0 1 1 1 1	4f
0100	0 1 1 0 0 1 1 0	66
0101	0 1 1 0 1 1 0 1	6d
0110	0 1 1 1 1 1 0 1	7d
0111	0 0 0 0 0 1 1 1	07
1000	0 1 1 1 1 1 1 1	7f
1001	0 1 1 0 1 1 1 1	6f



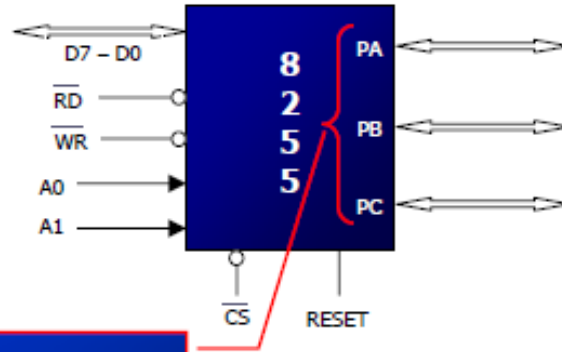
8255 – Programmable Peripheral Interface (PPI)

8255 Chip

8255 is a 40-pin DIP chip



8255 Block Diagram



It has three separately accessible 8-bit ports, A, B, and C

- They can be programmed to input or output and can be changed dynamically
- They have handshaking capability

⑨ PA0 - PA7 (8-bit port A)

- Can be programmed as all input or output, or all bits as bidirectional input/output

⑨ PB0 - PB7 (8-bit port B)

- Can be programmed as all input or output, but cannot be used as a bidirectional port

⑨ PC0 - PC7 (8-bit port C)

- Can be all input or output
- Can also be split into two parts:
 - CU (upper bits PC4 - PC7)
 - CL (lower bits PC0 - PC3)each can be used for input or output
- Any of bits PC0 to PC7 can be programmed individually

⑨ $\overline{\text{RD}}$ and $\overline{\text{WR}}$

- These two active-low control signals are inputs to the 8255
- The $\overline{\text{RD}}$ and $\overline{\text{WR}}$ signals from the 8031/51 are connected to these inputs

⑨ D0 – D7

- are connected to the data pins of the microcontroller
- allowing it to send data back and forth between the controller and the 8255 chip

⑨ RESET

- An active-high signal input
- Used to clear the control register
 - When RESET is activated, all ports are initialized as input ports

⑨ A0, A1, and $\overline{\text{CS}}$ (chip select)

- $\overline{\text{CS}}$ is active-low
- While $\overline{\text{CS}}$ selects the entire chip, it is A0 and A1 that select specific ports
- These 3 pins are used to access port A, B, C, or the control register

8255 Port Selection

CS	A1	A0	Selection
0	0	0	Port A
0	0	1	Port B
0	1	0	Port C
0	1	1	Control register
1	X	X	8255 is not selected

- ⑨ While ports A, B and C are used to input or output data, the control register must be programmed to select operation mode of three ports
- ⑨ The ports of the 8255 can be programmed in any of the following modes:
 1. Mode 0, simple I/O
 - Any of the ports A, B, CL, and CU can be programmed as input out output
 - All bits are out or all are in
 - There is no single-bit control as in P0-P3 of 8051

2. Mode 1

- Port A and B can be used as input or output ports with handshaking capabilities
- Handshaking signals are provided by the bits of port C

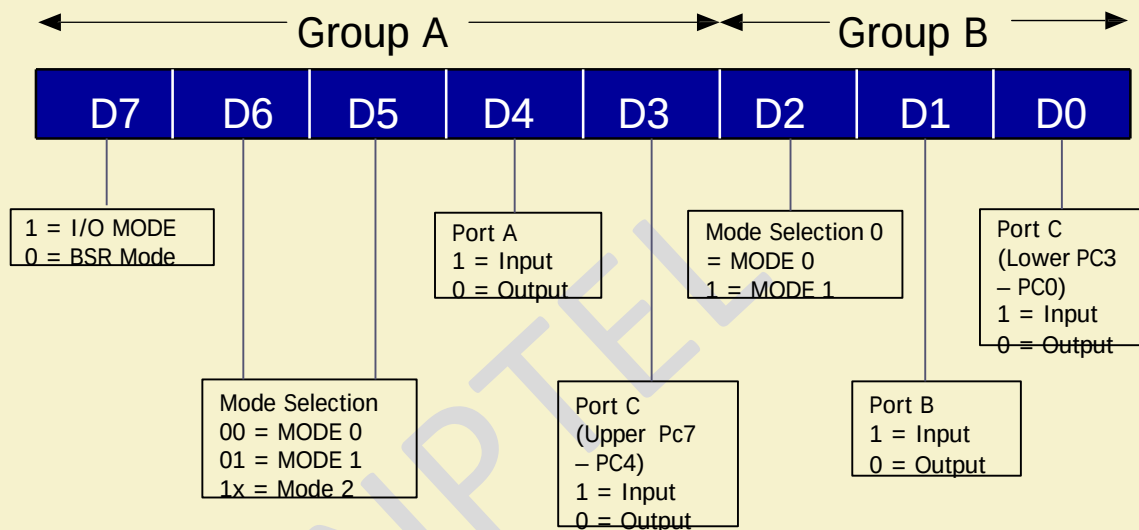
3. Mode 2

- Port A can be used as a bidirectional I/O port with handshaking capabilities provided by port C
- Port B can be used either in mode 0 or mode 1

4. BSR (bit set/reset) mode

- Only the individual bits of port C can be programmed

8255 Control Word Format (I/O Mode)



- ⑨ The more commonly used term is I/O Mode 0
- Intel calls it the basic input/output mode
 - In this mode, any ports of A, B, or C can be programmed as input or output
 - A given port cannot be both input and output at the same time

Example

Find the control word of the 8255 for the following configurations:

- (a) All the ports of A, B and C are output ports (mode 0)
- (b) PA = in, PB = out, PCL = out, and PCH = out

Solution:

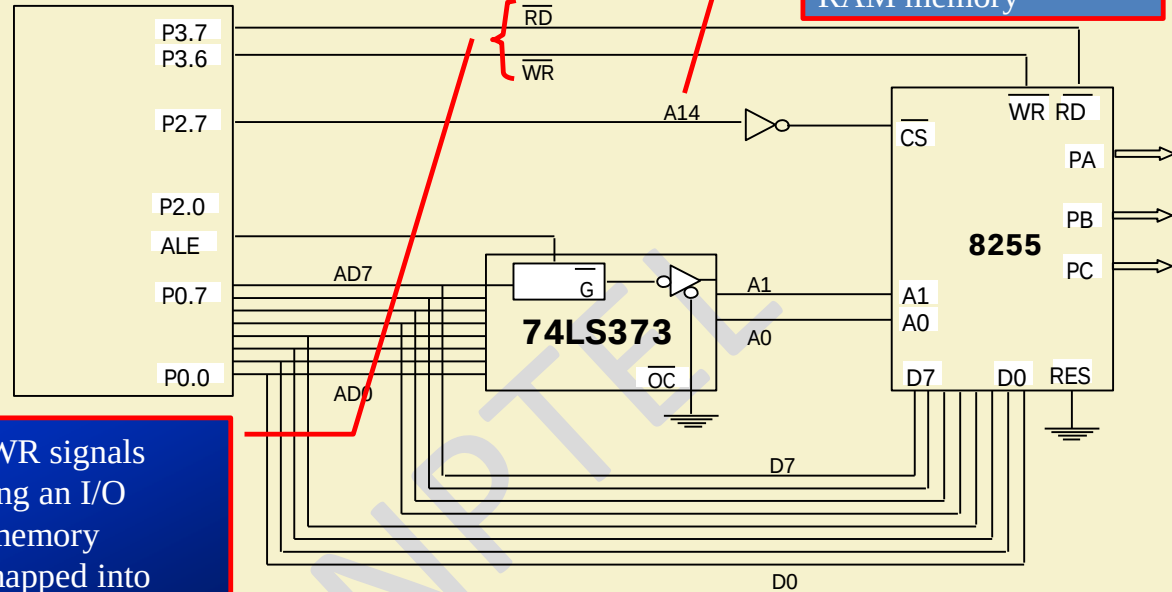
From Figure 15-3 we have:

- (a) 1000 0000 = 80H (b) 1001 0000 = 90H

- ⑨ The 8255 chip is programmed in any of the 3 modes
 - mentioned earlier by sending a byte (Intel calls it a control word) to the control register of 8255
- ⑨ We must first find the port address assigned to each of ports A, B ,C and the control register
 - called mapping the I/O port

8051 Connection to the 8255

8255 is connected to an 8031/51 as if it is a RAM memory



Notice the use of RD and WR signals

- This method of connecting an I/O chip to a CPU is called memory mapped I/O, since it is mapped into memory space
- use memory space to access I/O
- use instructions such as MOVX to access 8255

Example: Write a program to send 55H and AA to all ports continuously.

The base address for the 8255 is as follows:

A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	
X	1	X	x	X	x	x	x	x	x	x	x	x	x	0	0	= 4000H PA
X	1	X	X	x	x	x	x	x	x	x	x	x	X	0	1	= 4001H PB
X	1	X	X	x	x	x	x	x	x	x	x	X	X	1	0	= 4002H PC
x	1	x	X	x	x	x	x	x	x	x	x	x	x	1	1	= 4003H CR

The control byte (word) for all ports as output is 80H.

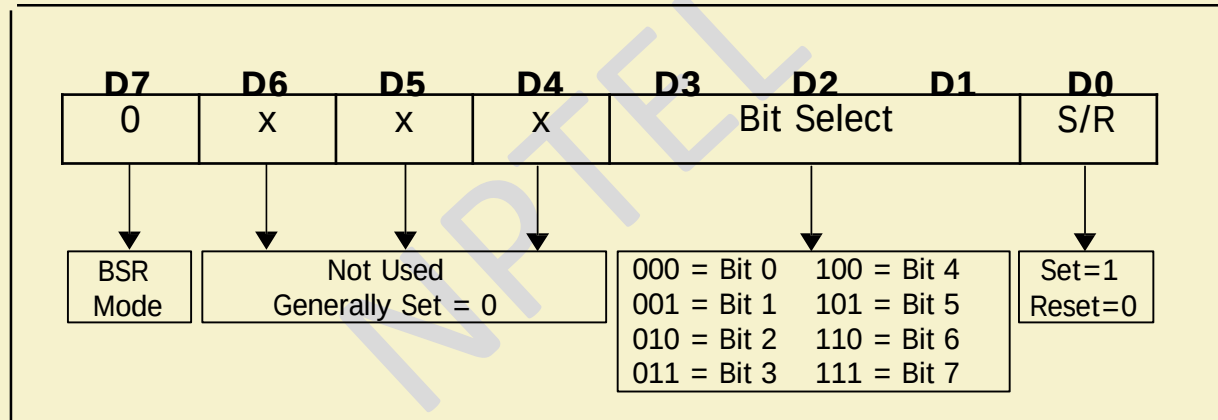
```

MOV      A, #80H           ;control word
                                ;(ports output)
MOV      DPTR, #4003H      ;load control reg
                                ;port address
MOVX     @DPTR, A          ;issue control word
MOV      A, #55H           ;A = 55H
AGAIN:   MOV      DPTR, #4000H ;PA address
MOVX     @DPTR, A          ;toggle PA bits
INC      DPTR              ;PB address
MOVX     @DPTR, A          ;toggle PB bits
INC      DPTR              ;PC address
MOVX     @DPTR, A          ;toggle PC bits
CPL      A                 ;toggle bit in reg A
ACALL    DELAY             ;wait
SJMP     AGAIN             ;continue

```

Bit Set Reset (BSR) Mode

- ⑨ A unique feature of port C
 - The bits can be controlled individually
- ⑨ BSR mode allows one to set to high or low any of the PC0 to PC7



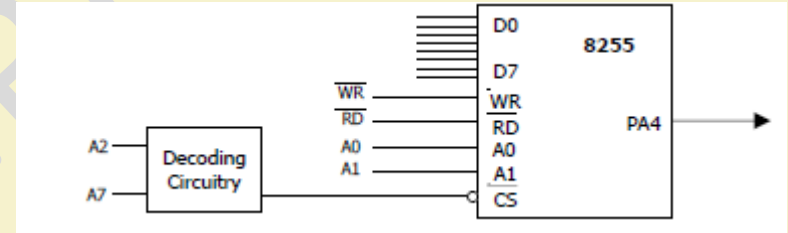
Program PC4 of the 8255 to generate a pulse of 50 ms with 50% duty cycle.

Solution:

To program the 8255 in BSR mode, bit D7 of the control word must be low. For PC4 to be high, we need a control word of “0xxx1001”.

Likewise, for low we would need “0xxx1000” as the control word. The x’s are for “don’t care” and generally are set to zero.

```
MOV    a, #00001001B    ;control byte for PC4=1
MOV    R1, #CNTPORT      ;load control reg port
MOVX   @R1, A             ;make PC4=1
ACALL  DELAY              ;time delay for high pulse
MOV    A, 00001000B      ;control byte for PC4=0
MOVX   @R1, A             ;make PC4=0
ACALL  DELAY
```



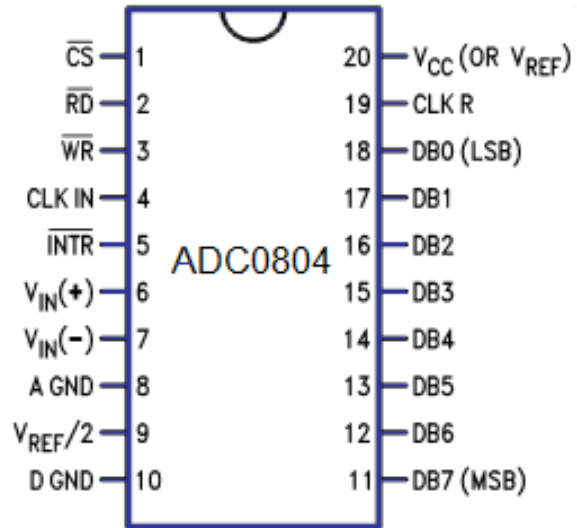
8255 Mode 1: I/O with Handshake

- ⑨ Handshaking refers to the process of two intelligent devices communicating back and forth
 - Example--printer
- ⑨ Mode 1: outputting data with handshaking signals
 - A and B can be used to send data to device with handshaking signals
 - Handshaking signals are provided by port C

Analog-to-Digital Converters (ADCs)

- Used for data acquisition
- Physical quantity converted to electrical signals using *transducers*
- An ADC has n-bit resolution, $1/2^n$ is the step-size
- Conversion time is the time needed for a conversion

ADC0804



FEATURES OF ADC 0804

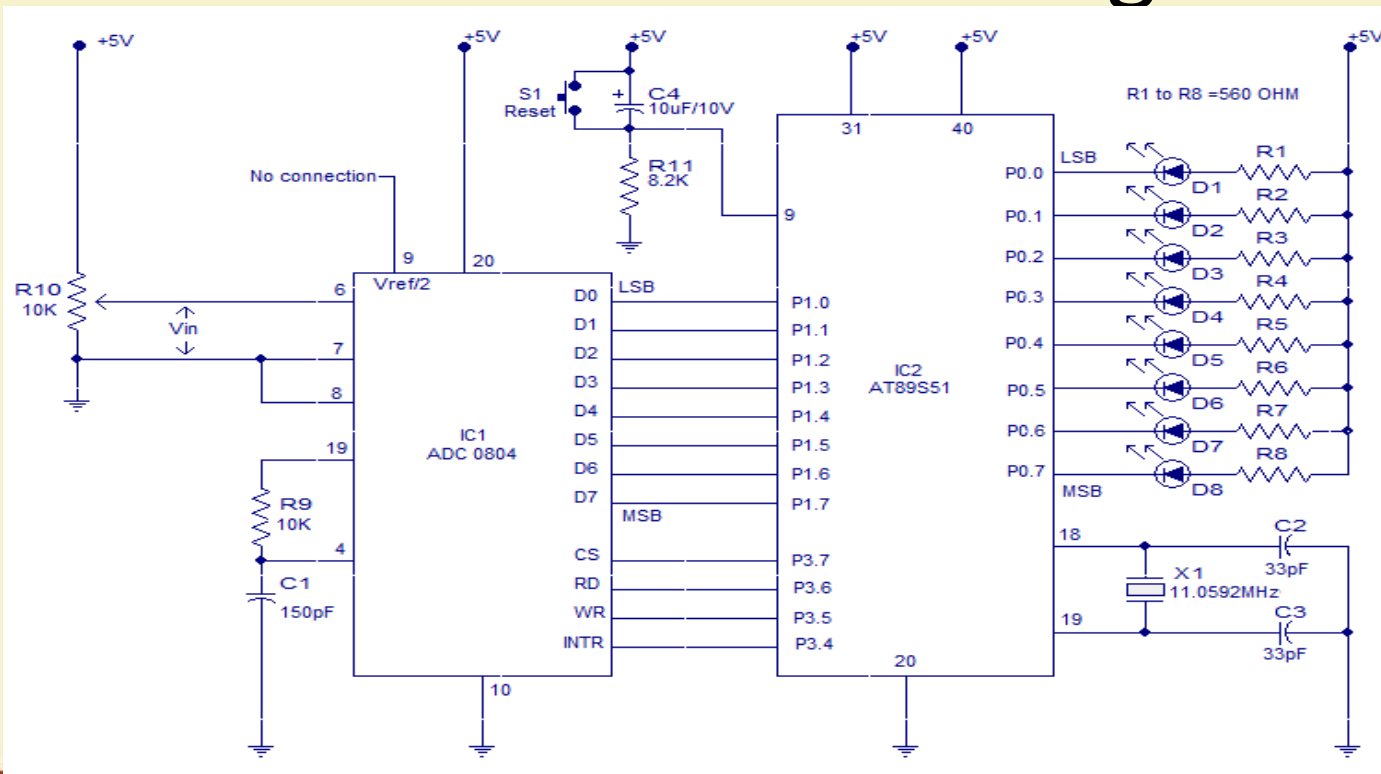
- 8 bit resolution
- Differential analogue voltage inputs
- 0-5V input voltage range
- No zero adjustment
- Built-in clock generator
- Voltage at $V_{ref}/2$ (pin9) can be externally adjusted to convert smaller input voltage spans to full 8 bit resolution.

Pin Description

- **CS: Chip Select** It is an active low pin and is used to activate ADC0804
- **RD: Read** It is an input pin and is active low. ADC stores the result in an internal register after conversion of analog data. This pin helps to get the data out of the ADC0804.
When CS=0, high to low pulse is given to RD pin, then digital output comes on the pins D0-D7
- **WR: Write** It is an input pin and is active low which is used to initiate the ADC to start the conversion process. When CS=0, WR makes a low to high transition, then ADC starts the conversion process.
- **CLK IN: Clock IN** This is an input pin which is connected to an external clock source.
- **INTR: Interrupt** This is an output pin and is active low. When the conversion is over, this pin goes low.

- **Vin+:** **Analog Input** Analog input to ADC.
- **Vin-:** **Analog Input** Analog input connected to ground.
- **AGND:** **Analog Ground** Connected to ground.
- **Vref/2:** **Reference Voltage** Used to set the reference voltage. Default reference voltage is 5V when not connected. Step size can be reduced by using this pin.
- **DGND:** **Digital Ground** Connected to ground.
- **D7-D0:** **Output Data Bits** Output bits of binary data.
- **CLKR:** **Clock Reset** To reset the clock.
- **Vcc:** **Positive Supply** Power supply of ADC.

8051 Interfacing



8051 Program

ORG 00H

MOV P1,#11111111B // initiates P1 as the input port

MAIN: CLR P3.7 // makes CS=0

SETB P3.6 // makes RD high

CLR P3.5 // makes WR low

SETB P3.5 // low to high pulse to WR for starting conversion

WAIT: JB P3.4,WAIT // polls until INTR=0

CLR P3.7 // ensures CS=0

CLR P3.6 // high to low pulse to RD for reading the data from ADC

MOV A,P1 // moves the digital data to accumulator

CPL A // complements the digital data

MOV P0,A // outputs the data to P0 for the LEDs

SJMP MAIN // jumps back to the MAIN program END