Microcontrollers in Embedded Systems

Chapter 9

Contents

- Intel 8051 Micro-controller Family, its architecture and instruction sets
- Assembly Language Programming
- Interfacing Seven Segment Display

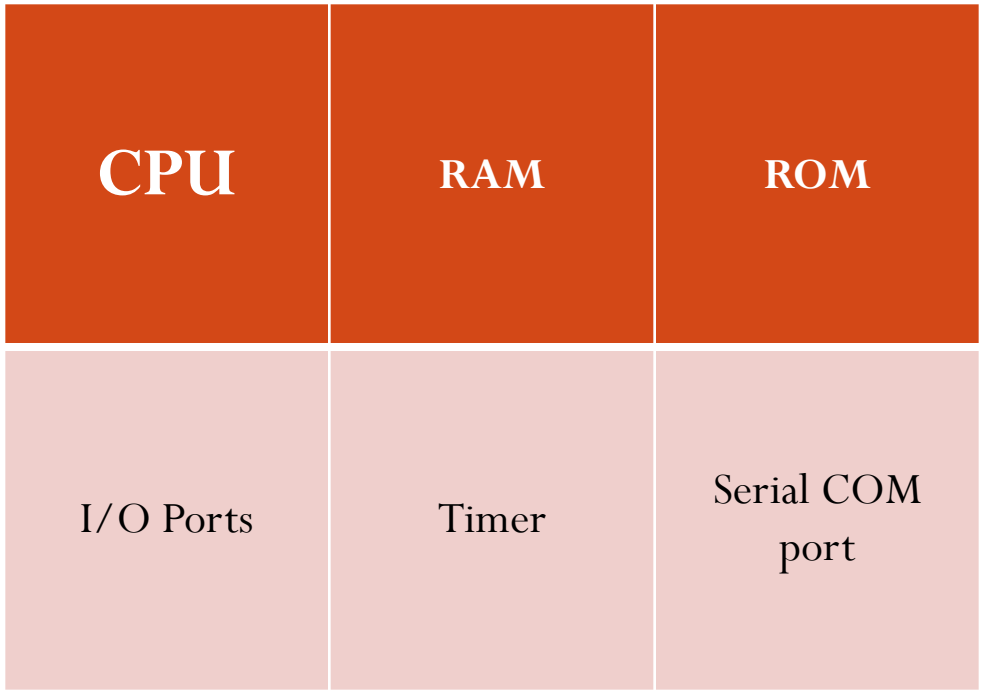
Intel 8051 Micro-Controller Family, its architecture and instruction sets

- Introduction
- General Block Diagram
- Comparison with Microprocessor
- Criteria for choosing a microcontroller
- Comparison of 8051 Family Members
- 8051 Architecture
- 8051 Instruction Set

Introduction

- Small computer (SoC) on a single IC
- Contains processor core, memory, I/O ports and other features
- Used for embedded applications
  - Automatically controlled products

General Block Diagram



Comparison with Microprocessor

Microprocessor	Microcontroller
General Purpose processors	Special Purpose Processors
It contains complete functional CPU only	In addition to functional CPU, it has timers, I/O ports, internal RAM and ROM and other features
Designer can select the size of memory, number of I/O ports, timers etc to be used in the system	Size of memory, number of I/O ports, timers etc will be fixed for a particular microcontroller
Clock speed is very high in GHz range	Clock speed is low in MHz range

...

Microprocessor	Microcontroller
Microprocessor based systems are expensive and consumes more power	Microcontroller based systems are cheap and consumes less power
Powerful addressing modes and many instructions to move data between memory and CPU	It includes many bit handling instructions along with byte processing instructions
Access times for external memory and I/O devices are more, resulting in slower system	Access times for on-chip memory and I/O devices are les, resulting in a faster system

Criteria for choosing microcontroller

- It must meet computational needs
  - Based on speed, packaging, power consumption, size of RAM and ROM, number of I/O pins and timer
- Flexibility to develop products around it
  - Availability of an assembler, debugger, compiler, emulator, technical support
- Readily available of microcontroller along with its reliable resources

Comparison of 8051 family members

Feature	8051	8052	8031
ROM (KB)	4	8	0
RAM (bytes)	128	256	128
Timers	2	3	2
I/O Pins	32	32	32
Serial Port	1	1	1
Interrupt Sources	6	8	6

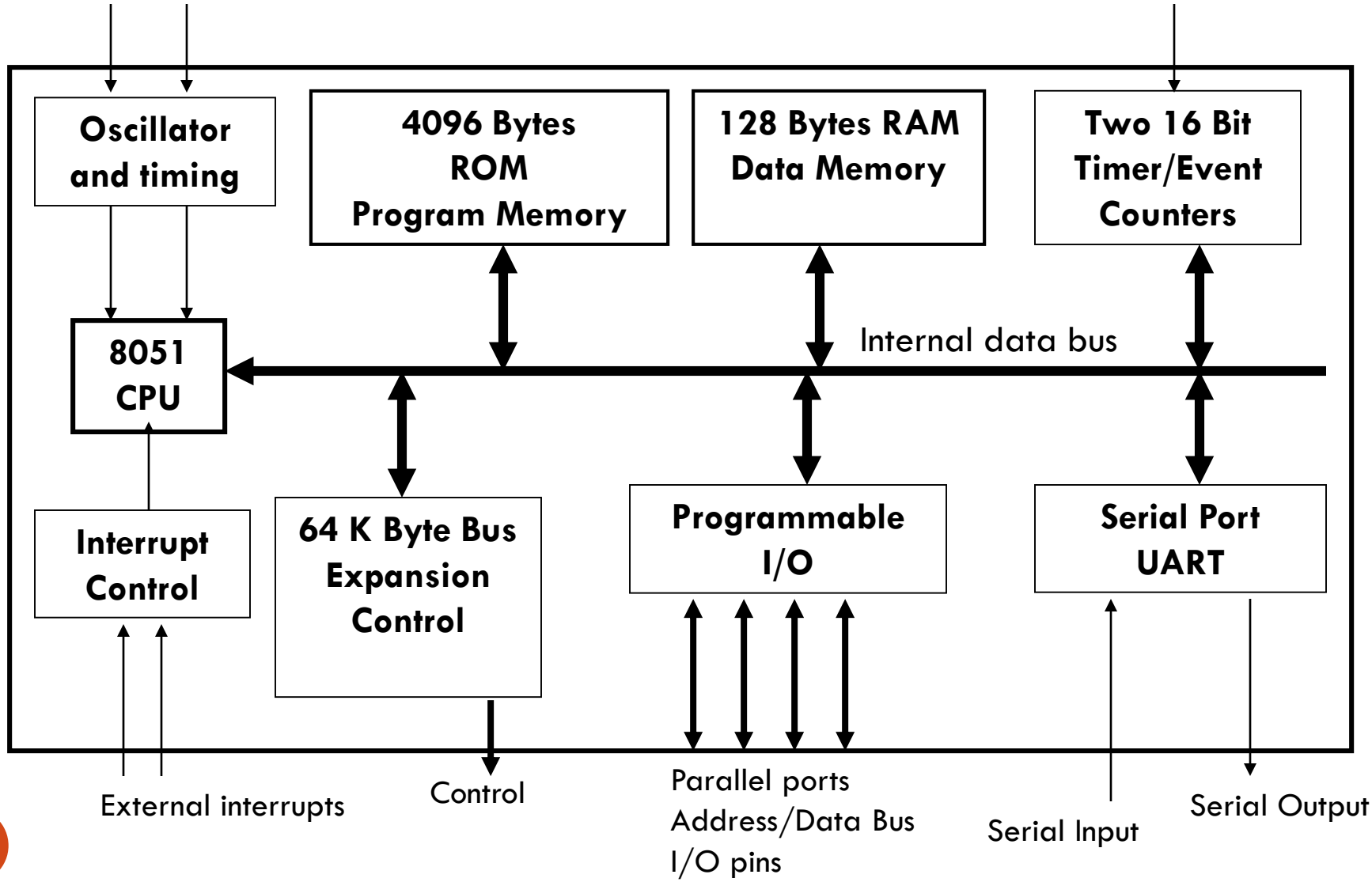
8051 Architecture

- Features of 8051 Architecture
  - Resistors for different operations
    - Eight bit CPU with registers A and B
    - Sixteen bit program counter (PC) and data pointer (DPTR):8bit DPH and DPL.
    - Eight bit program status word (PSW): 4 flags(CY, AC, OV, P) 2register selects bit.
    - Eight bit stack pointer (SP)(LIFO).
  - Internal ROM
    - 4KB as program memory.
    - Look up table can also be stored which can be accessed using appropriate instruction.

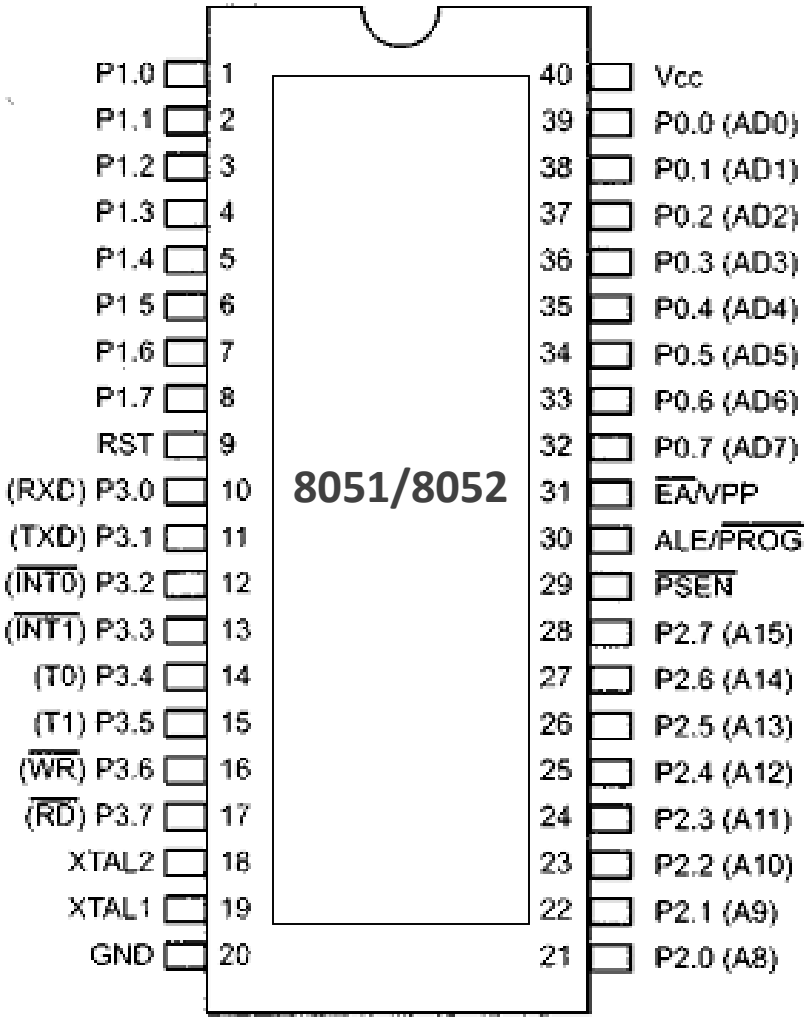
...

- Internal RAM (128 Bytes)
  - Four Register Banks, each of eight registers
  - Sixteen bytes of bit addressable memory
  - Eighty bytes of general purpose data memory
- Thirty two I/O pins arranged as four 8 bit ports
- Two 16 bit timer/counters: T0 and T1
- Full Duplex serial data receiver/transmitter: SBUF
- Control Register: TCON, TMOD, SCON, PCON, IP, IE
- Two external interrupts and three internal interrupt sources
- Oscillator and clock circuits

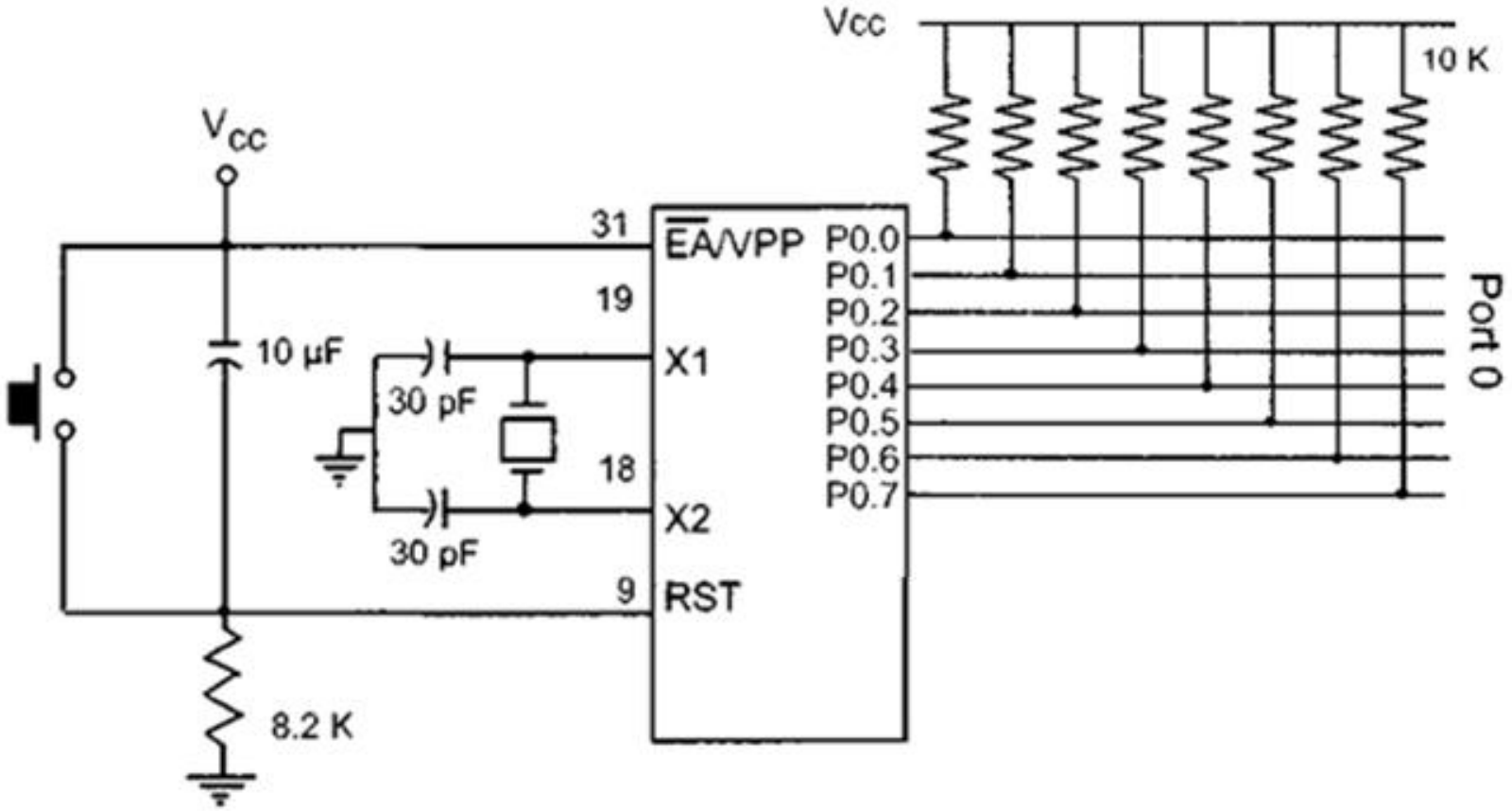
Internal Block Diagram of 8051



PIN CONFIGURATION



MINIMUM HARDWARE CONFIGURATION



8051 Instruction Sets

- **Data transfer instructions**
- **Arithmetic Instructions**
- **Logical Instructions**
- **Bit Manipulation Instructions**
- **Program Control Instructions**

Symbols and its meaning

- #data – represents 8 bit immediate value
- Rn – represents one of registers R0 – R7
- @Rp – represents address pointed by value of Rp. Rp may be either R0 or R1.
- Direct: represents direct byte addressable memory
- Bit: represents direct bit addressable memory
- C: Carry, A: Accumulator, B: Register B
- Addr11: 11 bit address, Addr16: 16 bit address, rel: relative address

Data Transfer Instructions

- MOV: Data movement between various registers
- MOVX: Moves data to and from external RAM
  - Registers R0, R1 and DPTR used to hold the address of data byte in external RAM
- MOVC: Code memory read only data move
  - Access is done using A along with DPTR or PC
- PUSH and POP: Stack operation
- XCH: Data exchange

Arithmetic Instructions

- ADD: Used to add values of register, memory
- ADDC: add two values with carry
- SUB: subtract one value from another
- SUBB: subtract one value from another with borrow
- INC: increments value of register/memory by one
- DEC: decrements value of register/memory by one
- MUL: Multiply accumulator and register B
- DIV: Divide accumulator by B register
- DA: Decimal Adjust Accumulator

Logical Instructions

- ANL: AND operation of two operand
- ORL: OR operation of two operand
- XRL: XOR operation of two operand
- CLR A: Clear Accumulator
- CPL A: Complement Accumulator

...

- RL A: Rotate Accumulator Left
- RLC A: Rotate Accumulator Left through Carry
- RR A: Rotate Accumulator Right
- RRC A: Rotate Accumulator Right through Carry
- SWAP A: Swap Nibbles of Accumulator

Bit Manipulation Instructions

- CLR C/bit: clears Carry/bit
- SETB C/bit: Sets Carry/bit
- CPL C/bit: Complements Carry/bit
- ANL C, bit: AND operation of C and bit
- ORL C, /bit: OR operation of C and NOT of bit
- MOV C/bit, bit/C: moves C/bit to bit/C

*bit represents bit addressable memory location

...

- JC rel: Jump if carry is set
- JNC rel: Jump if carry is not set
- JB bit, rel: jump if specified bit is set
- JNB bit, rel: Jump if specified bit is not set
- JBC bit, rel: if specified bit is set then Jump by clearing that bit

Program Control Instructions

- ACALL addr11: absolute subroutine call
- LCALL addr16: Long subroutine call
- RET: Return from subroutine
- RETI: Return from interrupt
- AJMP addr11: Absolute jump
- LJMP addr16: Long Jump
- SJMP rel: Short Jump
- JMP @A + DPTR: Indirect jump

...

- JZ rel: Jump if A is zero
- JNZ rel: Jump if A is not zero
- CJNE: Compare and Jump if not equal
  - CJNE A, direct/#data, rel
  - CJNE Rn/@Rp, #data, rel
- DJNE Rn/direct, rel: decrement and jump if not zero
- NOP: No operation

Assembly Language Programming

1. WAP to generate a 50% duty cycle pulse using a PIN 2.3 of microcontroller

   OR

   Blink a LED, with equal on and off time, connected to PIN 2.3

➔

```
        ORG 00H
        CLR P2.3
BACK:
        CLR P2.3
        LCALL DELAY

        SETB P2.3
        LCALL DELAY

        SJMP BACK
```

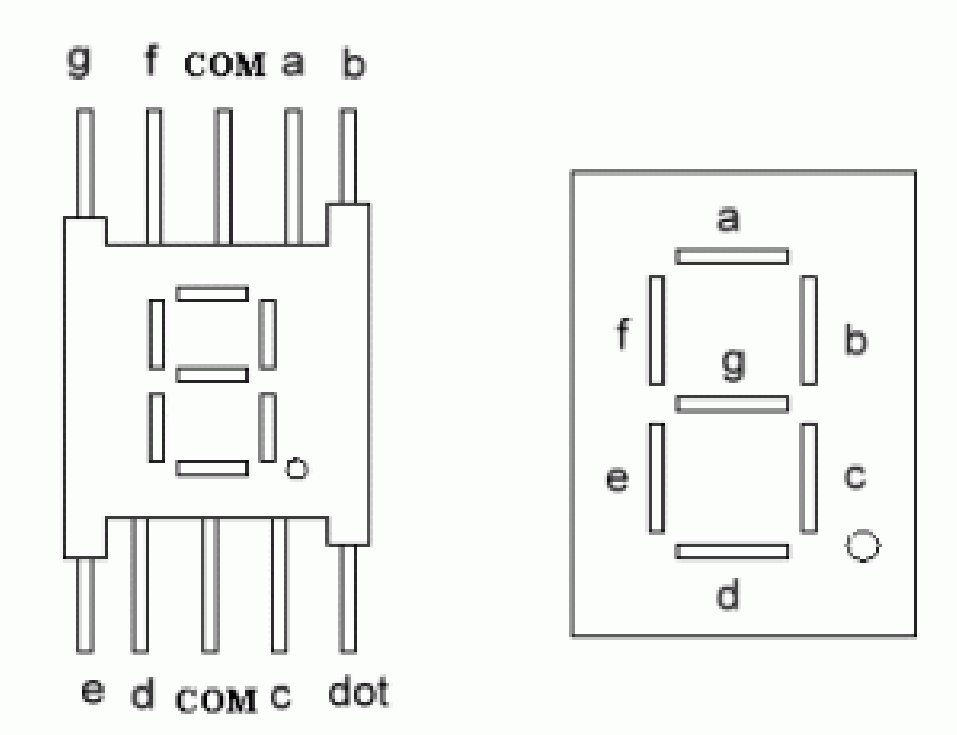
```
        ORG 300H
DELAY:
        MOV R5, #64H
AGAIN:  MOV R4, #0FFH
AGAN:   MOV R3, #08H
AGA:    DJNZ R3, AGA
        DJNZ R4, AGAN
        DJNZ R5, AGAIN
        RET
END
```

...

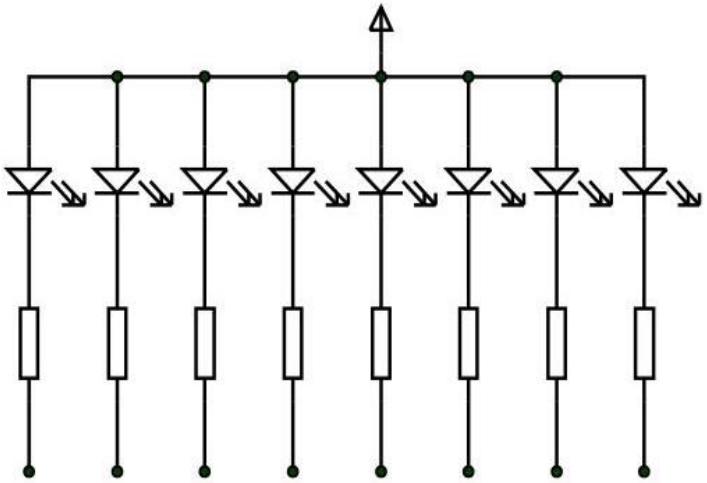
```
# include<at89x52.h>

void delay(unsigned int x)
{
    unsigned int i,j;
    for(i=0;i<x;i++)
        for(j=0;j<1275;j++);
}

void main()
{
        P2.3 = 0;
        while(1)
        {
            P2.3 = 0;
            delay(100);
            P2.3 = 1;
            delay(100);
        }
}
```

Interfacing Seven Segment Display

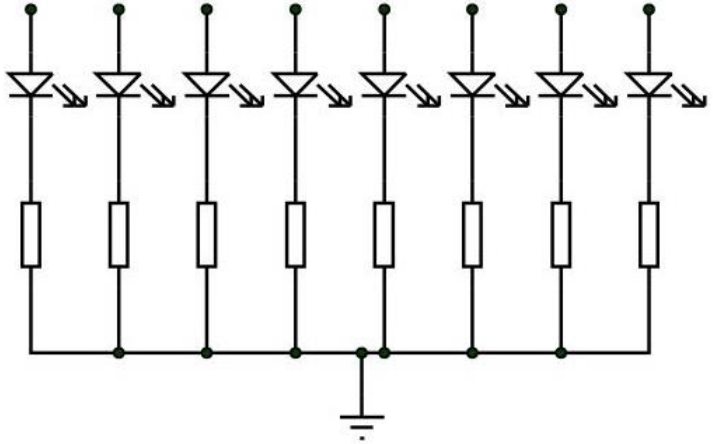
- Pin Configurations



Modes of Configuration



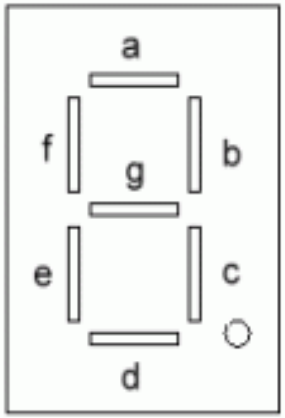
Common Anode Configurations



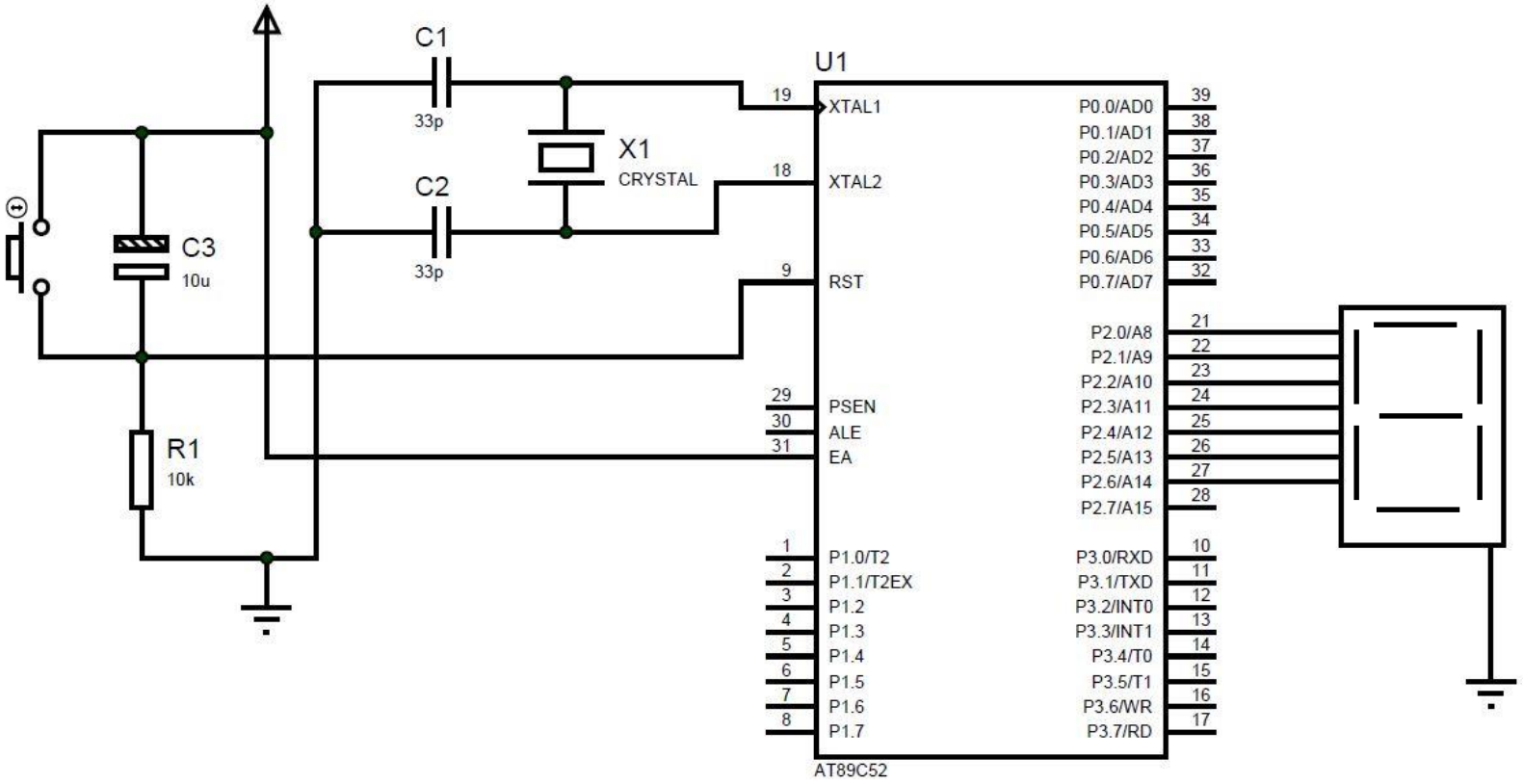
Common Cathode Configurations

Equivalent HEX Table

Common Anode Configurations										For Common Cathode mode
Digits	Individual LEDs Illuminated								HEX	
	dp	g	f	e	d	c	b	a		HEX
0	1	1	0	0	0	0	0	0	0xC0	0x3F
1	1	1	1	1	1	0	0	1	0xF9	0x06
2	1	0	1	0	0	1	0	0	0xA4	0x5B
3	1	0	1	1	0	0	0	0	0xB0	0x4F
4	1	0	0	1	1	0	0	1	0x99	0x66
5	1	0	0	1	0	0	1	0	0x92	0x6D
6	1	0	0	0	0	0	1	0	0x82	0x7D
7	1	1	1	1	1	0	0	0	0xF8	0x07
8	1	0	0	0	0	0	0	0	0x80	0x7F
9	1	0	0	1	0	0	0	0	0x90	0x6F



Connection with 8051/52



- Driver Circuit is not shown in the diagram
- Common cathode 7 segment is used

Programming in C

```
# include<at89x52.h>

void delay(unsigned int x)
{
      unsigned int i,j;
      for(i=0;i<x;i++)
            for(j=0;j<1275;j++);
}

char digits[] = {0x3F, 0x06,   0x5B,
0x4F, 0x66,            0x6D, 0x7D,
0x07,      0x7F, 0x6F};
```

```
void main()
{
    unsigned char i;
    P2 = 0x00;
    while(1)
    {
       for(i=0;i<10;i++)
       {
            P2 = digits[i];
            delay(100);
       }
    }
}
```

Programming in Assembly

```
MOV P2, #00H
BACK:
    MOV P2, #3FH
    LCALL DELAY
    MOV P2, #06H
    LCALL DELAY
    MOV P2, #5BH
    LCALL DELAY
    MOV P2, #4FH
    LCALL DELAY
    MOV P2, #66H
    LCALL DELAY
    MOV P2, #6DH
    LCALL DELAY
    MOV P2, #7DH
    LCALL DELAY
    MOV P2, #07H
    LCALL DELAY
    MOV P2, #7FH
    LCALL DELAY
    MOV P2, #6FH
    LCALL DELAY
SJMP BACK
```

...

```
        ORG 300H
DELAY:
                MOV R5, #64H
AGAIN:          MOV R4, #0FFH

AGAN:           MOV R3, #08H
AGA:DJNZ R3, AGA
                DJNZ R4, AGAN
                DJNZ R5, AGAIN
                RET
        END
```