MICROPROCESSOR SYSTEMS

8051 Microcontroller

Zuhaib A. Shaikh,
Asst. Prof., CSE Deptt.,QUEST
Web: zuhaib-shaikh.neocities.org

Intel 8051 Microcontroller

- A microcontroller is a System on Chip
- Contains CPU, ROM, RAM, INPUT, and OUTPUT ports all on a single chip.
- Less capable than the General Purpose microprocessors but cheap and require less connections to function
- Used in portable devices, small scale control systems, and intelligent systems
- Out of many microcontrollers built today, 8051 is very popular
- A typical microcontroller based system operations are defined below:
  - The program (firmware) is written using PC based tools and converted to machine code
  - The program (firmware) is loaded into microcontroller memory using a programmer (flash)
  - Microcontroller is connected to the peripherals and the power is turned ON
  - The microcontroller executes the program (firmware) loaded in its memory

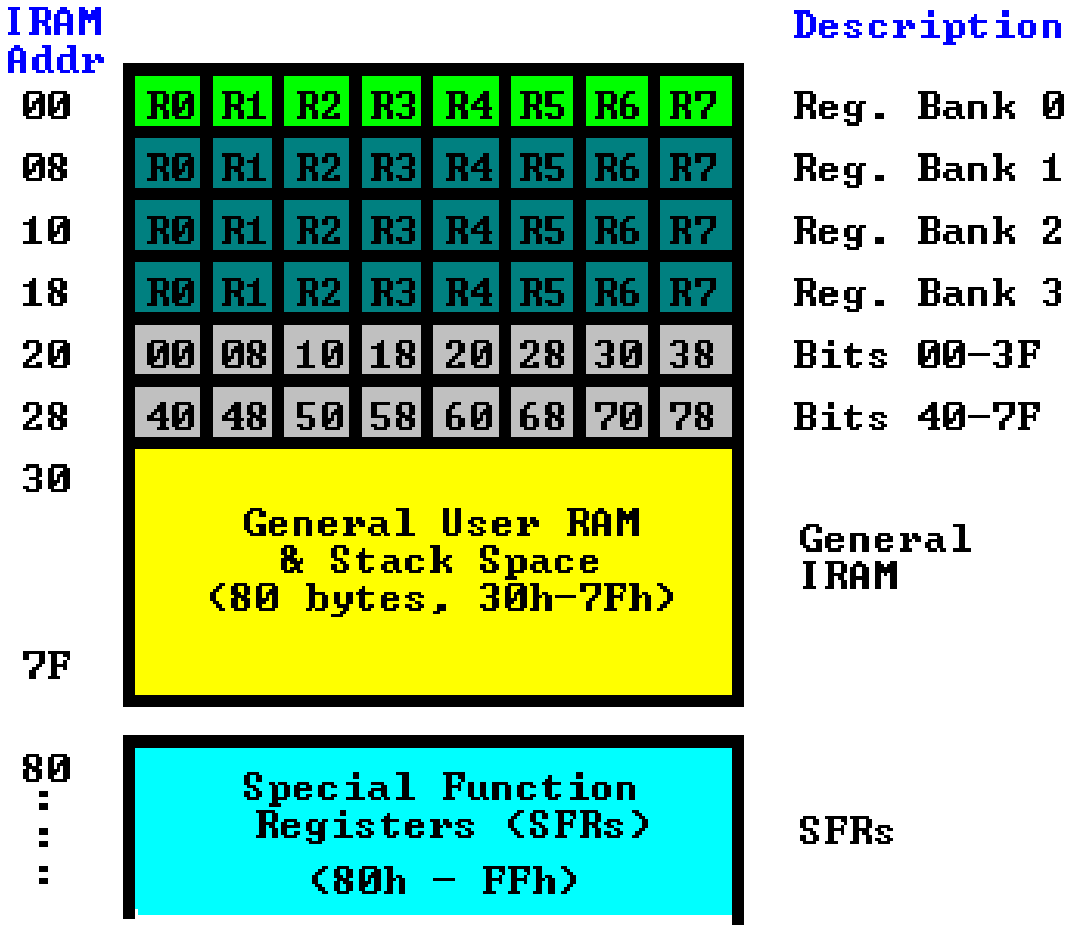
Inside 8051

- Read Only Memory (ROM)
- Random Access Memory (RAM)
- Electrically Erasable Programmable ROM (EEPROM)
- Special Function Registers (SFR)
- Program Counter
- Central Processor Unit (CPU)
- Input/output ports (I/O Ports)
- Oscillator
- Timers/Counters
- Serial communication

8051 Memory Types

- On-chip Memory
  - Resided on the microcontroller chip. Contains ROM and RAM
- Off-chip Memory
  - In the form of separate IC
- The On-chip RAM Memory contains the following:
  - Register Banks
  - Stack
  - User variables
  - Bit Memory
  - Special Function Registers (SFR)

Register Banks



Register Banks

- 4 Register Banks each of 8-bytes
- A Register bank is selected with a combination of flags
- Each Register in a Register Bank is named Rn, where n=0-7 for every bank
- If Register Bank 0 is selected then the instruction
  - ADD A, R0
- Will add the contents of memory location 00h to the contents of accumulator
- If Register Bank 2 is selected then the instruction
  - ADD A, R4
- Will add the contents of memory location 14h to the contents of accumulator

Bit Memory

- A number of bit variables capable of storing single bit
- Total 128 bit variables (from 00 to 7Fh)
- Commands SETB and CLR are used to manipulate them
- E.g. SETB 00h will set the first bit variable
- CLR 70h will clear the bit variable at location 70h
- Physically bit variables are part of I-RAM (20-2Fh)
- So MOV 20h, 0FFh is equivalent to:
  - SETB 00h
  - SETB 01h
  - SETB 02h
  - SETB 03h
  - SETB 04h
  - SETB 05h
  - SETB 06h
  - SETB 07h

Addressing Modes

- 8051 supports 8 addressing modes
- Where three (*) modes are related with branch instructions

Addressing Modes	**Instructions**
Register	MOV A,B
Direct	MOV A, 30H
Indirect	MOV A, @R1
Immediate	MOV A, #21H
Relative*	SJMP (within ±128 byte)
Absolute*	AJMP (within 2K code)
Long*	LJMP
Indexed	MOV A, @A+PC

Instruction Set

ARITHMETIC INSTRUCTIONS	
Mnemonic	Description
ADD A,Rn	Adds the register to the accumulator
ADD A,direct	Adds the direct byte to the accumulator
ADD A,@Ri	Adds the indirect RAM to the accumulator
ADD A,#data	Adds the immediate data to the accumulator
ADDC A,Rn	Adds the register to the accumulator with a carry flag
ADDC A,direct	Adds the direct byte to the accumulator with a carry flag
ADDC A,@Ri	Adds the indirect RAM to the accumulator with a carry flag
ADDC A,#data	Adds the immediate data to the accumulator with a carry flag
SUBB A,Rn	Subtracts the register from the accumulator with a borrow
SUBB A,direct	Subtracts the direct byte from the accumulator with a borrow
SUBB A,@Ri	Subtracts the indirect RAM from the accumulator with a borrow
SUBB A,#data	Subtracts the immediate data from the accumulator with a borrow
INC A	Increments the accumulator by 1
INC Rn	Increments the register by 1
INC Rx	Increments the direct byte by 1
INC @Ri	Increments the indirect RAM by 1
DEC A	Decrements the accumulator by 1
DEC Rn	Decrements the register by 1
DEC Rx	Decrements the direct byte by 1
DEC @Ri	Decrements the indirect RAM by 1
INC DPTR	Increments the Data Pointer by 1
MUL AB	Multiplies A and B
DIV AB	Divides A by B
DA A	Decimal adjustment of the accumulator according to BCD code

Instruction Set

BRANCH INSTRUCTIONS	
Mnemonic	**Description**
ACALL addr11	Absolute subroutine call
LCALL addr16	Long subroutine call
RET	Returns from subroutine
RETI	Returns from interrupt subroutine
AJMP addr11	Absolute jump
LJMP addr16	Long jump
SJMP rel	Short jump (from −128 to +127 locations relative to the following instruction)
JC rel	Jump if carry flag is set. Short jump.
JNC rel	Jump if carry flag is not set. Short jump.
JB bit,rel	Jump if direct bit is set. Short jump.
JBC bit,rel	Jump if direct bit is set and clears bit. Short jump.
JMP @A+DPTR	Jump indirect relative to the DPTR
JZ rel	Jump if the accumulator is zero. Short jump.
JNZ rel	Jump if the accumulator is not zero. Short jump.
CJNE A,direct,rel	Compares direct byte to the accumulator and jumps if not equal. Short jump.
CJNE A,#data,rel	Compares immediate data to the accumulator and jumps if not equal. Short jump.
CJNE Rn,#data,rel	Compares immediate data to the register and jumps if not equal. Short jump.
CJNE @Ri,#data,rel	Compares immediate data to indirect register and jumps if not equal. Short jump.
DJNZ Rn,rel	Decrements register and jumps if not 0. Short jump.
DJNZ Rx,rel	Decrements direct byte and jump if not 0. Short jump.
NOP	No operation

Instruction Set

DATA TRANSFER INSTRUCTIONS	
Mnemonic	Description
MOV A,Rn	Moves the register to the accumulator
MOV A,direct	Moves the direct byte to the accumulator
MOV A,@Ri	Moves the indirect RAM to the accumulator
MOV A,#data	Moves the immediate data to the accumulator
MOV Rn,A	Moves the accumulator to the register
MOV Rn,direct	Moves the direct byte to the register
MOV Rn,#data	Moves the immediate data to the register
MOV direct,A	Moves the accumulator to the direct byte
MOV direct,Rn	Moves the register to the direct byte
MOV direct,direct	Moves the direct byte to the direct byte
MOV direct,@Ri	Moves the indirect RAM to the direct byte
MOV direct,#data	Moves the immediate data to the direct byte
MOV @Ri,A	Moves the accumulator to the indirect RAM
MOV @Ri,direct	Moves the direct byte to the indirect RAM
MOV @Ri,#data	Moves the immediate data to the indirect RAM
MOV DPTR,#data	Moves a 16-bit data to the data pointer
MOVC A,@A+DPTR	Moves the code byte relative to the DPTR to the accumulator (address=A+DPTR)
MOVC A,@A+PC	Moves the code byte relative to the PC to the accumulator (address=A+PC)
MOVX A,@Ri	Moves the external RAM (8-bit address) to the accumulator
MOVX A,@DPTR	Moves the external RAM (16-bit address) to the accumulator
MOVX @Ri,A	Moves the accumulator to the external RAM (8-bit address)
MOVX @DPTR,A	Moves the accumulator to the external RAM (16-bit address)
PUSH direct	Pushes the direct byte onto the stack
POP direct	Pops the direct byte from the stack/td>
XCH A,Rn	Exchanges the register with the accumulator
XCH A,direct	Exchanges the direct byte with the accumulator
XCH A,@Ri	Exchanges the indirect RAM with the accumulator
XCHD A,@Ri	Exchanges the low-order nibble indirect RAM with the

Instruction Set

LOGIC INSTRUCTIONS	
Mnemonic	**Description**
ANL A,Rn	AND register to accumulator
ANL A,direct	AND direct byte to accumulator
ANL A,@Ri	AND indirect RAM to accumulator
ANL A,#data	AND immediate data to accumulator
ANL direct,A	AND accumulator to direct byte
ANL direct,#data	AND immediae data to direct register
ORL A,Rn	OR register to accumulator
ORL A,direct	OR direct byte to accumulator
ORL A,@Ri	OR indirect RAM to accumulator
ORL direct,A	OR accumulator to direct byte
ORL direct,#data	OR immediate data to direct byte
XRL A,Rn	Exclusive OR register to accumulator
XRL A,direct	Exclusive OR direct byte to accumulator
XRL A,@Ri	Exclusive OR indirect RAM to accumulator
XRL A,#data	Exclusive OR immediate data to accumulator
XRL direct,A	Exclusive OR accumulator to direct byte
XORL direct,#data	Exclusive OR immediate data to direct byte
CLR A	Clears the accumulator
CPL A	Complements the accumulator (1=0, 0=1)
SWAP A	Swaps nibbles within the accumulator
RL A	Rotates bits in the accumulator left
RLC A	Rotates bits in the accumulator left through carry
RR A	Rotates bits in the accumulator right
RRC A	Rotates bits in the accumulator right through carry

Instruction Set

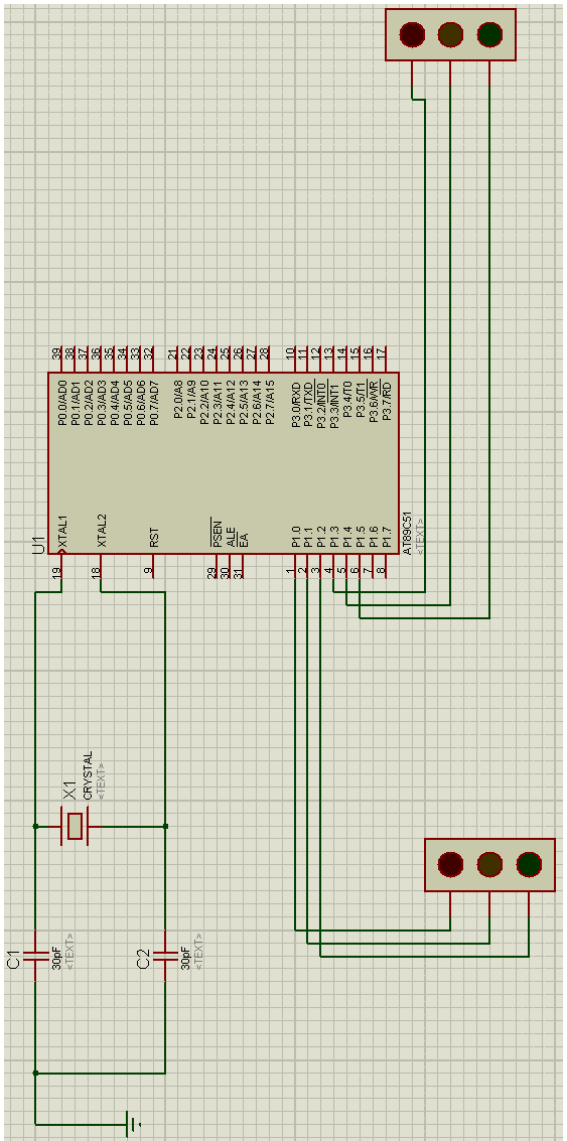
BIT-ORIENTED INSTRUCTIONS	
Mnemonic	**Description**
CLR C	Clears the carry flag
CLR bit	Clears the direct bit
SETB C	Sets the carry flag
SETB bit	Sets the direct bit
CPL C	Complements the carry flag
CPL bit	Complements the direct bit
ANL C,bit	AND direct bit to the carry flag
ANL C,/bit	AND complements of direct bit to the carry flag
ORL C,bit	OR direct bit to the carry flag
ORL C,/bit	OR complements of direct bit to the carry flag
MOV C,bit	Moves the direct bit to the carry flag
MOV bit,C	Moves the carry flag to the direct bit

8051 based control systems

- Traffic Light Control System

```
;INITILIZE PINS
RED1        EQU   P1.0
YELLOW1     EQU   P1.1
GREEN1      EQU   P1.2
RED2        EQU   P1.3
YELLOW2     EQU   P1.4
GREEN2      EQU   P1.5
MOV P1,#00H
START:
   SETB RED1
   SETB GREEN2
   ACALL DELAY_L
   CLR RED1
   CLR GREEN2
   SETB YELLOW1
   SETB YELLOW2
   ACALL DELAY_S
   CLR YELLOW1
   CLR YELLOW2
   SETB RED2
   SETB GREEN1
   ACALL DELAY_L
   CLR RED2
   CLR GREEN1
```

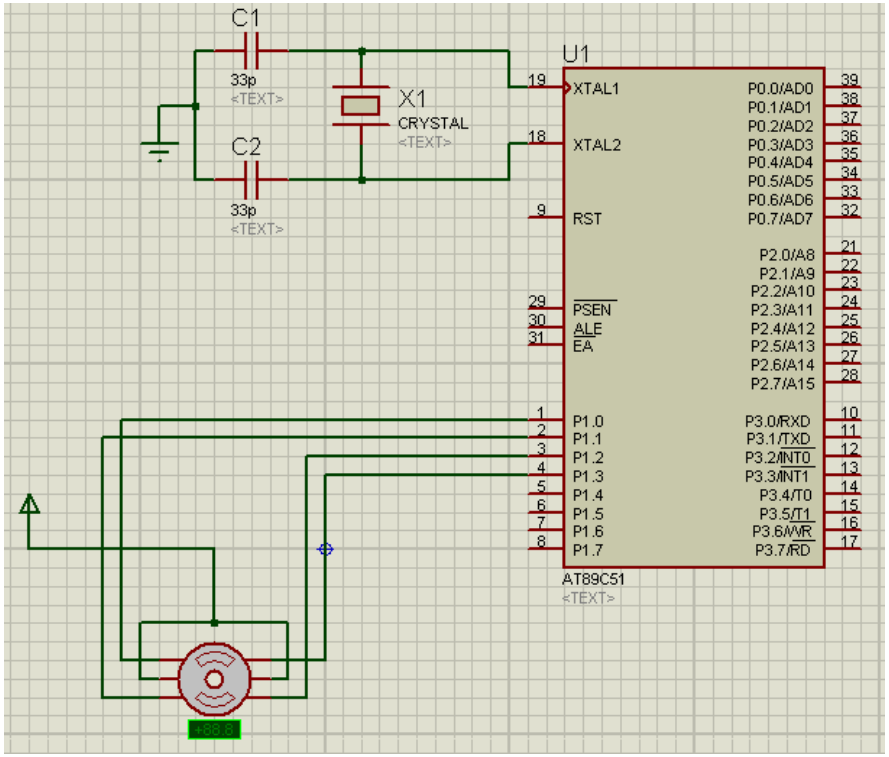
```
SETB YELLOW1
  SETB YELLOW2
  ACALL DELAY_S
  CLR YELLOW1
  CLR YELLOW2
  JMP START
DELAY_L:
              MOV R1,#80H
              MOV R2,#20H
              MOV R3,#10H
LOOP1: DJNZ R1,LOOP1
                DJNZ R2,LOOP1
                DJNZ R3,LOOP1
RET
DELAY_S:
              MOV R1,#30H
              MOV R2,#20H
              MOV R3,#10H
LOOP2: DJNZ R1,LOOP2
                DJNZ R2,LOOP2
                DJNZ R3,LOOP2
RET
END
```



8051 based control systems

- Stepper Motor Control System

```
STEPPER EQU P1
START: MOV STEPPER,#08H
                ACALL DELAY1
                MOV STEPPER,#0CH
                ACALL DELAY1
                MOV STEPPER,#04H
                ACALL DELAY1
                MOV STEPPER,#06H
                ACALL DELAY1
                MOV STEPPER,#02H
                ACALL DELAY1
                MOV STEPPER,#03H
                ACALL DELAY1
                MOV STEPPER,#01H
                ACALL DELAY1
                MOV STEPPER,#09H
                ACALL DELAY1
                JMP START
DELAY1: MOV R3,#8
                MOV R2,#32
                MOV R1,#117
TT2: DJNZ R1,TT2
                DJNZ R2,TT2
                DJNZ R3,TT2
RET
END
```



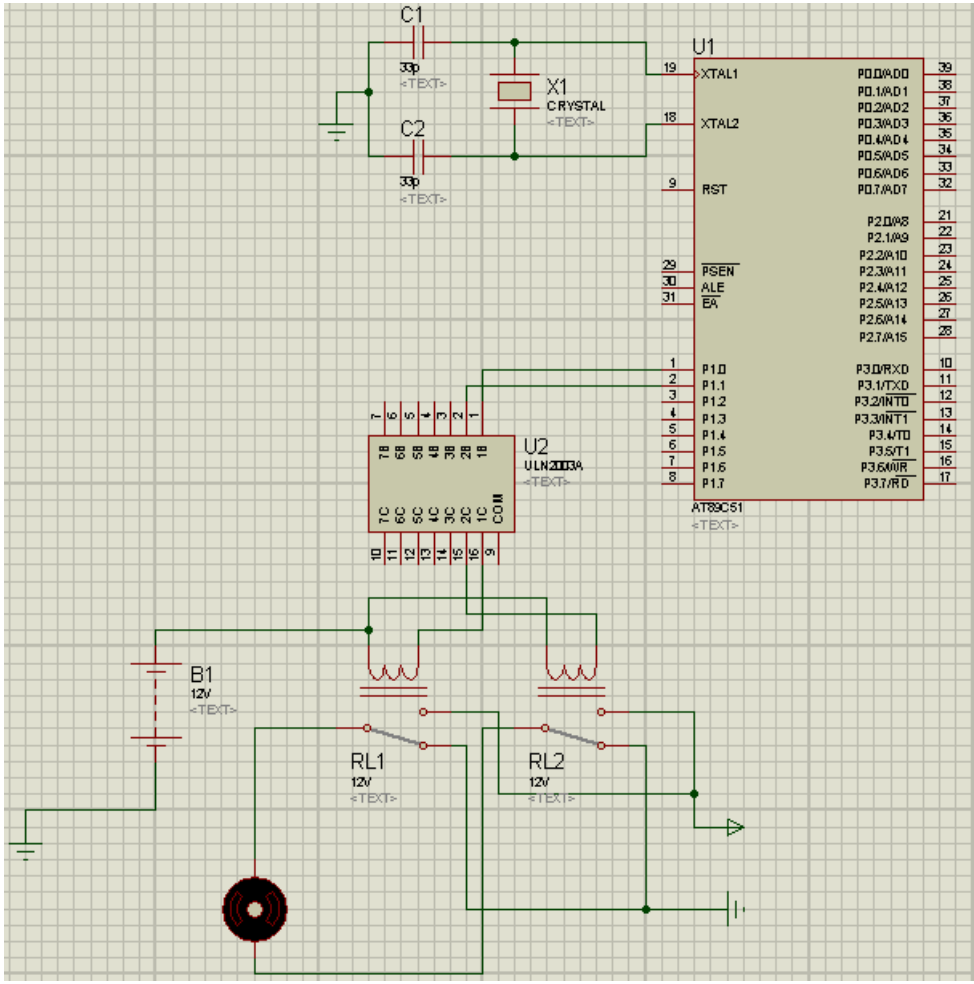
8051 based control systems

- Washing Machine Control System

```
PR1 EQU P1.0
PR2 EQU P1.1
CLR PR1
CLR PR2
START:      SETB PR1
            CLR PR2
            ACALL DELAY10
            CLR PR1
            CLR PR2
            ACALL DELAY2
            CLR PR1
            SETB PR2
            ACALL DELAY10
            CLR PR1
            CLR PR2
            ACALL DELAY2
            JMP START
DELAY10: MOV R3,#77
            MOV R2,#75
            MOV R1,#164
TT1: DJNZ R1,TT1
            DJNZ R2,TT1
            DJNZ R3,TT1
RET
```

```
DELAY2: MOV R3,#16
            MOV R2,#66
            MOV R1,#234
TT2: DJNZ R1,TT2
            DJNZ R2,TT2
            DJNZ R3,TT2

RET
END
```



Questions