USN :

CMR Institute of Technology, Bangalore
DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING
II - INTERNAL ASSESSMENT

Semester: 4-CBCS 2018
Subject: MICROCONTROLLER (18EC46)
Faculty: Mr Chetan Gowda

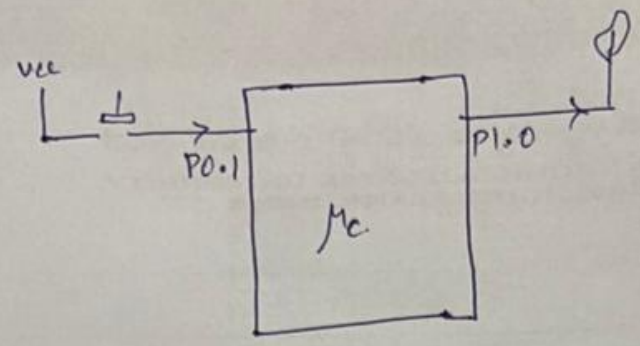
Date: 24 Jun 2021
Time: 01:00 PM - 02:30 PM
Max Marks: 50

Answer any 2 question(s)

Q.No		Marks	CO	PO	BT/CL
1	Consider a case where A)An LED connected to P1.0 is to be turned ON or OFF for every 0.25sec when the switch connected to P0.1 is turned ON, else the LED should be turned off. Write an ALP for executing the same. B) Use a subroutine to set set the flag after every 0.25 sec (can use any bit in bit addressable register as flag), if the flag is set, use the contents of registers used in delay subroutines from case A and check if the value is a palindrome, if yes toggle P1.1 else toggle P1.2 Draw relevant flowchart.	25	CO2,CO3	PO2,PO3	L3
2	In one of the workplaces, a sensor has been attached at the main door which captures the height(8-bit data) of the container entering the building, the sensor is connected to port P0 of 8051. Write ALP for, A) After every 10 entries, the data is sorted in ascending order also the total sum is calculated. B) If the sum of 10 individual height is greater than 200(decimal) value. Toggle P1.0 else, toggle P1.1. Even though the sensor reads digital data, but entered data is in the form of HEX. It's of interest if the data is converted to BCD at all instances for better readability.	25	CO2,CO3	PO2,PO3	L3
3	A) Explain the following instructions with examples: i. DJNZ Rn, Label ii. DA A iii. CJNE A, #10H, Label iv. RLC A v. SETB vi. CPL B) Classify the CALL instructions in 8051 and explain each of them with relevant examples. Explain RET instruction. C) Write an ALP to exchange the lower nibble of data present in external memory 6000H and internal memory 45H.	25	CO2,CO3	PO2,PO3	L2

P0.1 → i/p , P1.0 → o/p

1 (a, b)



(A)

```
        MOV P0, #00H
        MOV P1, #0FFH.
here:   JNB P0.1 here
back:   SETB P1.0.
        CALL delay  // delay program for 0.25 calculation
        JNB P0.1, back
        CLR P1.0.
finish: END.

Delay:  MOV R0, #xx.
here2:  MOV R1, #yy
here1:  MOV R2, #zz.
here:   DJNZ R2, here.
        DJNZ R1, here1
        DJNZ R2, here2
        RET.
```

To check xx yy zz is palindrom

(B)

```
        MOV A, R0.
        SWAP A.
Check   CJNE A, R1, Toggle1
(xx == zz)
        MOV A, R2
Check   ANL A, 0F0H.
y = y   MOV R3, A.
        MOV A, R2
        ANL A, 0FH.
        SWAP A.
        CJNE A, R2, Toggle.
```

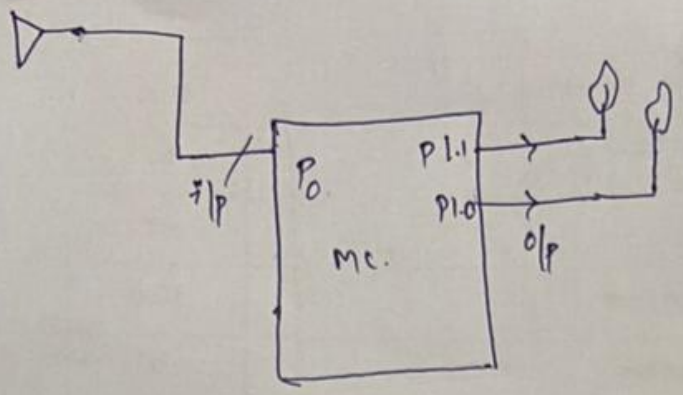
```
Toggle:  setb P1.1
         CALL delay.
         SJMP finish.

Toggle1: setb P1.2
         CALL delay
         SJMP finish.
```

2(a,b).

i/p → FF

o/p → 00h.




```
        MOV   P0, #0FFh.
        MOV   R0, #0Ah.
        MOV   R1, #50h.
loop:   MOV   A, P0.        → can use a delay program.
        MOV   @R1, A.
        INC   R1
        DJNZ  R0, loop.

        /* Write Sorting program */
(A)     CALL  Ascend.   // Ascending program.
        CALL  SUM.      // Sum calculator.
              .         // Store sum. in mem (R2).
        CJNE  R2, #200D, Next.
JC next1 →
next1:  Set b P1.1
        SJMP back.
Next :  Set b P1.0.
        SJMP back.
back. ; CALL BCD    // BCD program.
        SJMP loop.
        End.
```

3 A) Explain the following instructions with examples:

(i) DJNZ Rn, Label:-

Decrement register Rn by 1 & jump to the relative address if the result is not zero; no flags are affected.

example:- DJNZ R0, Label.

ii) DA A (Decimal adjust after addition)

→ When 2 BCD numbers are added, the answer is a non-BCD number

→ To get the result in BCD, DA A instruction is executed (after adding 2 BCD numbers) & storing the result in A.

→ The working of DA A is as follows

If [(A3-0) > 9 or AC = 1]

then (A3-0) ← (A3-0) + 6, A(7-4) ← (A7-4) + AC

→ DA adjusts the contents of accumulator to correspond to BCD numbers have been added by the ADD or ADDC instruction

→ If the carry bit is set or if the value of bits 0-3 exceed 0x06 is added to accumulator

Example:- DA A.

(iii) CJNE A, #10H, Label

Compare the contents of the A register with 10H; if they are not equal then jump to the relative address; set the carry flag to 1, if A is less than the number, otherwise, set the carry flag to 0.

Example :-

CJNE A, #10H, Label.

(IV) RLC A.

The RLC instruction rotates the eight bits in the accumulator & the one bit in carry flag left one bit position. Bit 7 of the accumulator is rotated into the carry flag while the orignd value of carry flag is rotated into bit 0 of the accumulator Bit 0 of the accumulator is rotated into bit 1, bit 1 into bit 2 & so on. No other flags are affected by this operation

example :- RLC A.

V) SET B

The SETB instruction sets the bit operand to a value of 1. This instruction can operate on carry flag or any other directly addressable bit. No flags are affected by this instruction.

operation :- SETB
(bit) = 1

Example :- SETB 63H.

(VI) CPL.

The CPL instruction logically complements the value of the specified destination operand & stores the result back in the destination operand. Bits that previously contained a 1 will be changed to 0 & bits that previous contained 0 will be changed to a 1.

Example :- CPL A

3B) Classify the CALL instructions in 8051 & explain each of them with relevant examples. Explain RET instruction.

1) LCALL address [16 bit]

Call the subroutine located anywhere in program memory space; push the address of the instruction immediately following the call on stack

LCALL instruction work as.

1) [PC] = [PC]+3
2) [SP] = [SP] +1 (if SP contains default value 07, Then SP increments & [SP]= 08
3) [SP] = [PC 7-0]
4) [SP] = [SP]+1
5) [ ]
6) PC = address (16 bit).

ACALL Address (11 bit)

This is absolute call instruction which unconditionally calls the subroutine located at the indicated 11 bit address. This is a 2 byte instruction. The ACALL instruction works as.

a) During execution of SCALL, [PC] = [PC+2]; during execution of this instruction

PC = 8549 + 2 = 854Bh.

b) [SP] = [SP]+1 (If SP contains default value as 7 it increments to [SP] = 08

c) [[SP]] = [PC7-4]: (lower byte of PC contents 4B will be stored in memory loc 08.

d) [SP] = [SP]+1 (SP increments from 08 → 09)

e) [[SP]] = [PC15-8]; Higher byte of PC content, ie 85 will be stored in memory loc 09.

(0x854B) will be stored in stack

f) [PC10-0] = address (11 bit)

RET instruction

RET instructions pops two contents from stack & load it to PC

g) $[PC_{15-8}] = [[SP]]$; contents of current top of the stack will be moved to higher byte.

h) $[SP] = [SP] - 1$ (sp decrements)

(i) $[PC_{7-0}] = [[SP]]$; contents of bottom of the stack will be moved to lower byte of PC

(j) $[SP] = [SP] - 1$ (SP decrements again.

3C) Write an ALP to exchange the lower nibble of data present in external memory 6000H & internal memory 45H.

1) ORG 0000H (Set program counter 00h)
2) MOV DPTR, #6000H (copy address 6000H to dptr)
3) MOVX A, @ DPTR (Copy contents of 6000h to A)
4) MOV R0, #45h (load pointer R0=45H)
5). MOV @R0, A ;(copy cont of A to RAM pointed by R0)
6) Increment DPL ; (Increment ~~DPTR~~ pointer)
7) MOVX A, @DPTR ; (copy contents of 6001H to A)

8) XCHD A, @R0 : (exchange lower nibble of A with RAM pointed by R0)

9) MOVX @DPTR, A ; (Copy contents of A to 6001)

10) DEC DPL ; (Decrement pointer)

11) MOV A, @R0 ; (copy contents of RAM pointed by R0 to A)

12) MOVX @DPTR, A ; (copy contents of A to RAM pointed by DPTR)

13) END.

---

1) Consider a case where

A) An LED connected to P1.0 is to be turned ON & OFF every 0.25 sec. when the switch connected to P0.1 is turned on. else the LED should be turned off write an ALP for executing the same

Soln:-

1) SETB P1.0
2) AGAIN : MOV C, P1.0
3) JNC AGAIN
4) SETB P1.0
5) ACALL DELAY
6) CLC
7) MOV P1.0, C.