CMR
INSTITUTE OF
TECHNOLOGY

USN | | | | | | | | | | |

Internal Assessment Test - 2

Sub:	**Microcontrollers**							Code:	18EE52
Date:	2/12/2022	Duration:	90 mins	Max Marks:	50	Sem:	5th	Branch:	EEE

Answer **Any FIVE FULL** Questions

		Marks	OBE	
			CO	RBT
1	Explain different jump and call instructions of 8051 microcontroller with their jump ranges with the help of figure. Jump explainaton-5 marks Call explaination -5 marks	10	CO2	L2
2	Write an ALP of 8051 to read the content of Port 0 and send it to Port 1 after inversion, continuously. Reading the content -3 Inversion-3 Continuously sending to port1-4	10	CO2	L3
3	Write an assembly language program to toggle the bits of port P1 Alp program-10m	10	CO3	L3
4	Write an ALP & C program to compute 1+2+.....+N (say 15) and save the sum at 70H alp- 5 M C program- 5M	10	CO3	L3
5	Write an assembly language program to exchange the lower nibble of data present in external memory 6000H and 6001H. alp- 8 M expected o/p- 2 M	10	CO3	L3
6	Explain with suitable example, the various conditional instructions available in 8051 in detail. Any five conditional statement with example-2 m each	10	CO2	L4
7	Write an ALP to find the average of ten 8 bit numbers stored in internal RAM location starting from 30 H to onwards store result at 60 H. Alp- 8 M Expected o/p- 2 M	10	CO2	L3
8	Analyze different data types supported by 8051C Microcontroller. 5 data types and its explaination with each example-2 M	10	CO3	L4

All conditional jumps are short jumps

It must be noted that all conditional jumps are short jumps, meaning that the address of the target must be within -128 to +127 bytes of the contents of the program counter (PC). This very important concept is discussed at the end of this section.

Unconditional jump instructions

The unconditional jump is a jump in which control is transferred unconditionally to the target location. In the 8051 there are two unconditional jumps: LJMP (long jump) and SJMP (short jump). Each is discussed below.

LJMP (long jump)

LJMP is an unconditional long jump. It is a 3-byte instruction in which the first byte is the opcode, and the second and third bytes represent the 16-bit address of the target location. The 2-byte target address allows a jump to any memory location from 0000 to FFFFH.

Remember that although the program counter in the 8051 is 16-bit, thereby giving a ROM address space of 64K bytes, not all 8051 family members have that much on-chip program ROM. The original 8051 had only 4K bytes of on-chip ROM for program space; consequently, every byte was precious. For this reason there is also an SJMP (short jump) instruction, which is a 2-byte instruction as opposed to the 3-byte LJMP instruction. This can save some bytes of memory in many applications where memory space is in short supply. SJMP is discussed next.

SJMP (short jump)

In this 2-byte instruction, the first byte is the opcode and the second byte is the relative address of the target location. The relative address range of 00 – FFH

nother control transfer instruction is the CALL instruction, which is used to call a subroutine. Subroutines are often used to perform tasks that need to be performed frequently. This makes a program more structured in addition to saving memory space. In the 8051 there are two instructions for call: LCALL (long call) and ACALL (absolute call). Deciding which one to use depends on the target address. Each instruction is explained next.

LCALL (long call)

In this 3-byte instruction, the first byte is the opcode and the second and third bytes are used for the address of the target subroutine. Therefore, LCALL can be used to call subroutines located anywhere within the 64K-byte address space of the 8051. To make sure that after execution of the called subroutine the 8051 knows where to come back to, the processor automatically saves on the stack the address of the instruction immediately below the LCALL. When a subroutine is called, control is transferred to that subroutine, and the processor saves the PC (program counter) on the stack and begins to fetch instructions from the new location. After finishing execution of the subroutine, the instruction RET (return) transfers control back to the caller. Every subroutine needs RET as the last instruction

ACALL (absolute call)

ACALL is a 2-byte instruction in contrast to LCALL, which is 3 bytes. Since ACALL is a 2-byte instruction, the target address of the subroutine must be within 2K bytes because only 11 bits of the 2 bytes are used for the address. There is no difference between ACALL and LCALL in terms of saving the program counter on the stack or the function of the RET instruction. The only difference is that the target address for LCALL can be anywhere within the 64K-byte address space of the 8051 while the target address of ACALL must be within a 2K-byte range. In many variations of the 8051 marketed by different companies, on-chip ROM is as low as IK byte. In such cases, the use of ACALL instead of LCALL can save a number of bytes of program ROM space.

2. ORG 00h
MOV A, #0FFH;A=FF hex
MOV P0, A;make P0 an i/p port ;by writing it all 1s
BACK: MOV A, P0 ;get data from P0
CPL A : TAKE COMPLEMENT OF A
MOV P1, A ;send it to port 1
SJMP BACK ;keep doing it
END

3. ORG 00h
MOV A, #0FFH;A=FF hex
MOV P1, A;make P0 an i/p port ;by writing it all 1s
MOV A,#55H
BACK: MOV A, P1 ;get data from P0
CPL A : TAKE COMPLEMENT OF A
MOV P1, A ;send it to port 1
SJMP BACK ;keep doing it
END

4.

```
ORG 0000h
MOV R0,#30h     ; N value
   MOV B,@R0
   INC R0
   MOV A,R0
   MUL AB         ; n(n+1)
   MOV B,#02h
   DIV AB         ; n(n+1)/2
   MOV 70h,A       ; Final result is stored in register R4
end
```

```
include <reg51.h>
void main(void)
 {
    Unsigned int i;
    Unsigned char sum;
        for (i = 1; i <= 15; i++)
   {
        sum =(i*(i+2))/2;
    }

    }
```

6. org 0000h
Mov r2,#01
Mov dptr,#6000h
Movx a,@dptr
Swap a
Movx @dptr,a
Inc dptr
Djnz r2,l1
end

7. org 00h
Mov r1,#0ah
Mov r0,#30h
Mov a,@r0
L1:Inc r0
Add a,@r0
Djnz r1,l1
Mov b,#0ah
Div ab
Mov 60h,a
End

8. explaination
Unsigned char
Signed char
Unsigned int
Signed int
Sbit (single bit)
Bit and sfr

- The bit data type allows access to single bits of bit-addressable memory spaces 20 – 2FH
- To access the byte-size SFR registers, we use the sfr data type

Data Type	Size in Bits	Data Range/Usage
unsigned char	8-bit	0 to 255
(signed) char	8-bit	-128 to +127
unsigned int	16-bit	0 to 65535
(signed) int	16-bit	-32768 to +32767
sbit	1-bit	SFR bit-addressable only
bit	1-bit	RAM bit-addressable only
sfr	8-bit	RAM addresses 80 – FFH only

- The unsigned int is a 16-bit data type
  - Takes a value in the range of 0 to 65535 (0000 – FFFFH)
  - Define 16-bit variables such as memory addresses
  - Set counter values of more than 256
  - Since registers and memory accesses are in 8-bit chunks, the misuse of int variables will result in a larger hex file
- Signed int is a 16-bit data type
  - Use the MSB D15 to represent – or +
  - We have 15 bits for the magnitude of the number from –32768 to +32767
- The signed char is an 8-bit data type
  - Use the MSB D7 to represent – or +
  - Give us values from –128 to +127
- We should stick with the unsigned char unless the data needs to be represented as signed numbers
  - temperature

- The character data type is the most natural choice
  - 8051 is an 8-bit microcontroller
- Unsigned char is an 8-bit data type in the range of 0 – 255 (00 – FFH)
  - One of the most widely used data types for the 8051
    - Counter value
    - ASCII characters
- C compilers use the signed char as the default if we do not put the keyword *unsigned*