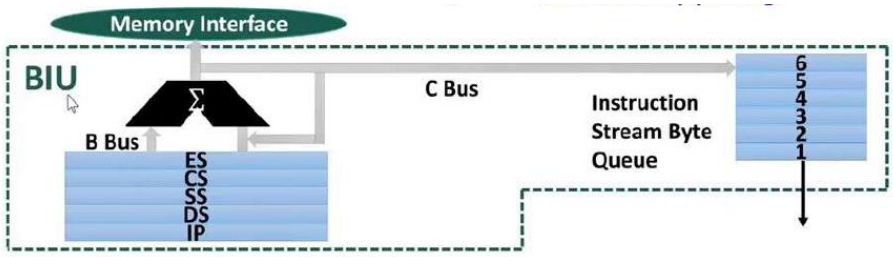
VIT Vellore Institute of Technology (Deemed to be University under section 3 of the UGC Act, 1956)	**Final Assessment Test- KEY – Winter 2023-24 Semester - May 2024**	
	Course code : BECE204L	Slot: C1+TC1
	Course Title : Microprocessors and Microcontrollers	Time: **Three Hours**
Emp Id: shanidul.hoque@vit.ac.in	Course Mode : **CBL**	Max. Marks: **100**
+91-9401408475	Faculty Name : Dr. Shanidul Hoque	School: SENSE

Sl. No:	Class Number(s): VL2023240505477	
1.	**What are the differences between i3, i5, and i7 processors, and how do they compare in terms of performance, features, and suitability for various computing tasks?**	**10**

ANS.

Intel Core i3	**Intel Core i5**	**Intel Core i7**
Intel Core i3 is an entry-level processor design for low-end applications.	Intel Core i5 is a mid-range processor developed for everyday use computer systems.	Intel Core i7 is a high-end processor designed for applications that need high processing power.
Core i3 processor is a dual core processor, i.e. it has two physical cores.	Core i5 is available in quad-core (2 cores) and quad core (4 cores).	Core i7 is available in quad core (4 cores), six cores, and eight-cores.
Core i3 has a cache of size either 3 MB or 4 MB.	Core i5 has a cache of size ranging between 3 MB and 8 MB.	Core i7 has a cache of size ranging between 4 MB and 8 MB.
The clock speed of Core i3 processor is ranging between 1.30 GHz and 3.50 GHz.	The clock speed of Core i5 processor is ranging between 1.90 GHz and 3.80 GHz.	The clock speed of Core i7 processor is ranging between 2.6 GHz and 3.7 GHz.
Core i3 processor supports 4 threads.	Core i5 processor also supports 4 threads.	Core i7 processor supports 8 threads.
Core i3 processor does not support turbo boost feature.	Core i5 supports turbo boost.	Core i7 processor also supports turbo boost.
Core i3 processor does not support virtualization technology.	Core i5 processor supports virtualization technology.	Core i7 processor also supports virtualization technology.
Core i3 processor does not support Intel vPro technology.	Core i5 processor does not support Intel vPro technology.	Core i7 processor supports Intel vPro technology.
Core i3 processor is cheapest among the three.	Core i5 is mid-range processor.	Core i7 is expensive.
Core i3 processor is suitable to use for entry level tasks such as word processing, browsing, documentation, watching videos, etc.	Core i5 processor is suitable for basic work like web browsing, word processing, spreadsheets, basic graphics designing, etc.	Core i7 processor is suitable for high end applications, such as gaming, graphics designing, video editing, simulation, etc.

2.	**(a) What are the key architectural components of the Intel 8086 microprocessor? Explain the function of Bus Interface Unit (BIU) with suitable diagram.** **ANS.** The key architectural components of the Intel 8086 microprocessor are Bus Interface Unit (BIU): Instruction queue, Segmentation register, Instruction pointer Execution Unit (EU): Control Unit, Registers, ALU, Flag Register Explain each block of BIU:  **(b) Rectify errors if any and identify the addressing mode of each of the following instruction in 8086 microprocessor:** **(i) MOV AX, #3FE1H** **(ii) MOV DS, 3000H** **(iii) MOV BL, CX** **(iv) MOV AX, 50H[BX]** **(v) MOV CL, [BX+SI+30H]** **ANS:** (i) Error; MOV AX, 3FE1H ; Immediate addressing mode (ii) Error; MOV AX, 3000H; Immediate addressing MOV DS, AX; Register addressing (iii) Error; MOV BL, [CX]; Register indirect addressing (iv) No error; Register relative addressing **(v)** No error; Base relative indexed addressing	**5+5 =10**

3.	**Develop an Intel 8086 assembly language program to count the odd and even numbers from a given list of 100 numbers stored at starting address of 2000:0300H and save the count values at 3000:0500H and 4000:0700H, respectively.**
	ANS:
	//Comments should be written corresponding to the instructions
	MOV AX,2000H
	MOV DS,AX
	MOV SI,300H
	MOV CX,64H
	MOV BL,00H
	MOV BH,00H
	BACK: MOV AL,[SI]
	ROR AL,1
	JC ODD
	INC BL
	JMP NEXT
	ODD: INC BH
		10

```
NEXT: INC SI
      LOOP BACK
      MOV AX,3000H
      MOV DS,AX
      MOV [500H], BH
      MOV AX,4000H
      MOV DS,AX
      MOV [700H], BL
      HLT
```

4. **(a) Assume a data (x) is present in Accumulator. If x is negative number, write an 8051 assembly language program to transfer a string of data from code space starting at address 200H to RAM location starting at 40H in reverse order. The data is given below:** **10**

0200H: DB "Believe in yourself "

ANS:

//Comments should be written corresponding to the instructions

Assume X is present in Accumulator

```
ORG 0000H
        MOV A, #0F3H; Assume any 8-bit value of X , here e.g X=0F3H is considered.
        RLC A
        JNC BYPASS
        MOV A, #00H
        MOV DPTR, #0213H
        MOV R1, #19
        MOV R0, #40H
 LOOP: CLR A
        MOVC A,@A+DPTR
        MOV @R0, A
        DEC DPL
        INC R0
        DJNZ R1, LOOP
 HERE: SJMP HERE
        ORG 0200H
        DB "VIT UNIVERSITY"
BYPASS: END
```

(b) Develop an 8051 based assembly language program to implement the expressions for half adder and half subtractor circuits. Store the sum/Difference, carry and borrow in PSW.1, PSW.5 and ACC.7, respectively. (Use Basic Logic Gates only) **10**

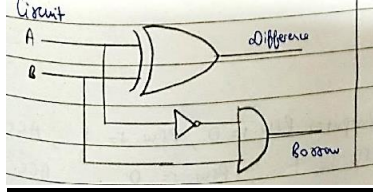
ANS.

Truth table :-

A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

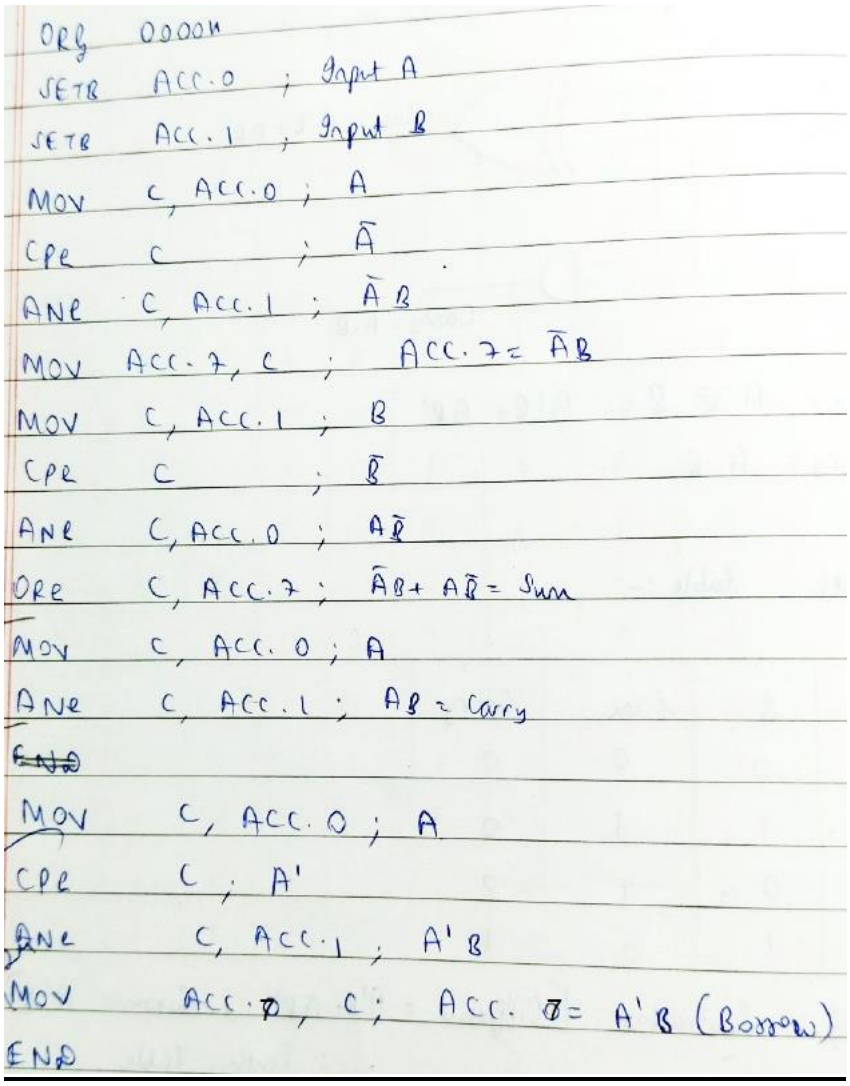
Half Substractor [Difference = A'B + AB' , Borrow = A'B]

Circuit



Truth Table

A	B	Diff	Borrow
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0



5.	**Calculate the amount of delay caused by the delay subroutine if the system has an 8051 with frequency of 16 MHz. The number of machine cycles are provided in parentheses** **DELAY: MOV R2, # 10010110B (1)** **AGAIN: MOV R3, #250 (1)** **HERE: NOP (1)** **NOP (1)** **NOP (1)** **DJNZ R3, HERE (2)** **DJNZ R2, AGAIN (2)** **RET (2)** **ANS:** Time period of machine cycle = 12/16MHz = 0.75 microsec Delay for HERE loop, 5 x 250 x 0.75 =937.5 microsec Delay for Again loop, 3 x 150 x 0.75 + 150 x 937.5 = 140962.5 microsec Total Delay = 3 x 0.75 + 140962.5 = 140964.75 microsec	**10**

6.	**Assume P1 and P2 of the 8051 are connected to LEDs and switches, respectively as shown in Figure 1. Write an 8051 assembly language program to (a) get data on switches connected to P2 and send it to the PC COM serially, (b) receive any data sent by the PC COM and put it on LEDs connected to P1. Perform both the tasks one after another continuously with 2400 baud rate.**	**10**

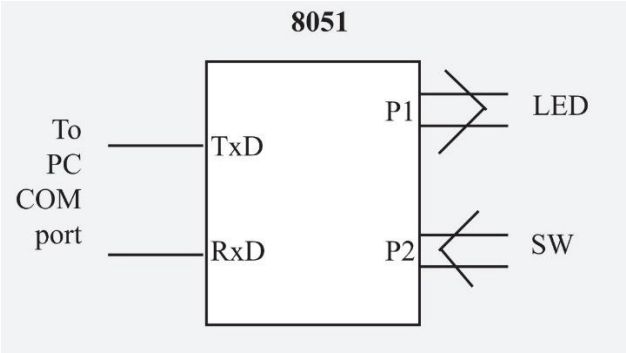


Figure 1

ANS:

//Calculation of TH1 values

//Comments should be written corresponding to the instructions

```
      ORG 0000H
      MOV P2,#0FFH              ;make P2 an input port
      MOV TMOD,#20H             ;Timer 1, mode 2(auto-reload)
      MOV TH1,#-12              ;2400 baud rate (or F4H)
      MOV SCON,#50H             ;8-bit,1 stop, REN enabled
      SETB TR1                  ;start Timer 1
AGN:  MOV  A,P2                 ;read data on P2
      ACALL SEND                ;transfer it serially
      ACALL RECV                ;get the serial data
      MOV P1,A                  ;display it on LEDs
      SJMP AGN                  ;stay in loop indefinitely
;--------------serial data transfer. ACC has the data
SEND:   MOV SBUF,A              ;load the data
HERE:   JNB TI, HERE            ;stay here until last bit gone
          CLR  TI               ;get ready for next char
          RET                   ;return to caller
;----------------receive data serially in ACC
RECV: JNB RI, RECV              ;wait here for char
         MOV A, SBUF            ;save it in ACC
        CLR RI                  ;get ready for next char
        RET                     ;return to caller
```

7.	**Write an 8051 assembly language program that continuously get 8-bit data from P0 and sends it to P1 while the P3.3 (INT1) pin is connected to a switch which is normally high. Whenever it goes low, it should turn on an LED for 100 microsecond which is connected to P2.5 pin.**	**10**

ANS:

//Calculation of TH0 values

//Comments should be written corresponding to the instructions

```
ORG 0000H
  LJMP MAIN
;…….ISR……
ORG 0013H
  LJMP NEXT
```

```
;……..MAIN Program……..
ORG 0030H
MAIN: MOV P0,#0FFH
      MOV P1,#00H
      CLR P2.5
      MOV IE, #10000100B
HERE: MOV A,P0
      MOV P1,A
      SJMP HERE
; ……..
NEXT:SETB P2.5
     MOV TMOD, #02H
     MOV TH0, #-92
     SETB TR0
L1: JNB TF0, L1;
    CLR TR0
    CLR TF0
    CLR P2.5
    RETI
    END
```

8. **(a) Write an 8051 assembly language program to interface an 16x2 LCD by sending Hex code commands "38H, 0EH, 01H, 06H" and display "VIT Vellore" on it in line 2 position 3. Assume D0-D7, RS, R/W', and E pins of LCD are connected with P1.0-P1.7, P2.0, P2.1, and P2.2 pins of 8051 microcontroller, respectively.** **10**

ANS:

```
ORG 0000H
MOV A,#38H
ACALL COMNWRT
ACALL DELAY
MOV A,#0EH
ACALL COMNWRT
ACALL DELAY
MOV A,#01H
ACALL COMNWRT
ACALL DELAY
MOV A,#06H
ACALL COMNWRT
ACALL DELAY
MOV A,#C3H
ACALL COMNWRT
ACALL DELAY
```

```
        MOV   DPTR,#MYDATA
D1:     CLR   A
        MOVC  A,@A+DPTR
        ACALL DATAWRT  ;call command subroutine
        ACALL DELAY    ;give LCD some time
        INC   DPTR
        JZ    AGAIN
        SJMP  D1
AGAIN:  SJMP  AGAIN    ;stay here
COMNWRT:               ;send command to LCD
        MOV   P1,A     ;copy reg A to P1
        CLR   P2.0     ;RS=0 for command
        CLR   P2.1     ;R/W=0 for write
        SETB  P2.2     ;E=1 for high pulse
        ACALL DELAY    ;give LCD some time
        CLR   P2.2     ;E=0 for H-to-L pulse
        RET
DATAWRT:               ;write data to LCD
        MOV   P1,A     ;copy reg A to port 1
        SETB  P2.0     ;RS=1 for data
        CLR   P2.1     ;R/W=0 for write
        SETB  P2.2     ;E=1 for high pulse
        ACALL DELAY    ;give LCD some time
        CLR   P2.2     ;E=0 for H-to-L pulse
        RET
DELAY:  MOV   R3,#250  ;50 or higher for fast CPUs
HERE2:  MOV   R4,#255  ;R4 = 255
HERE:   DJNZ  R4,HERE  ;stay until R4 becomes 0
        DJNZ  R3,HERE2
        RET
        ORG   300H

MYDATA: DB "VIT Vellore", 0
        END
```

(b) Develop an assembly language program to interface a 4x4 matrix keyboard with 8051 microcontroller. **10**

ANS.

```
MOV P2, #0FFH ; make P2 input (column)
L1: MOV P1, #00H ; make P2 output (row)
    MOV A, P2 ; read all column
    ANL A, #0FH;
    CJNE A, #0FH, L2
    SJMP L1

L2: ACALL Delay ; call 20 ms delay

    MOV A, P2 ;
   ANL A, #0FH ; check if any key is pressed
   CJNE A, #0FH, ScanRow ; key pressed
    SJMP L1                    scan row
```

```
ScanRow: MOV P1, #11111110B ; ground row 0
         MOV A, P2; read all columns
         ANL A, #0FH
         CJNE A, #0FH, Row_0; key row 0, find column
         MOV P1, #11111101B ; ground row 1
         MOV A, P2
         ANL A, #0FH
         CJNE A, #0FH, Row_1; key row 1, find column
         MOV P1, #11111011B ; ground row 2
         MOV A, P2
         ANL A, #0FH
         CJNE A, #0FH, Row_2; key row 2, find column
         MOV P1, #11110111B ; ground row 2
         MOV A, P2
         ANL A, #0FH
         CJNE A, #0FH, Row_3 ; key row 3, find column
         LJMP L1
```

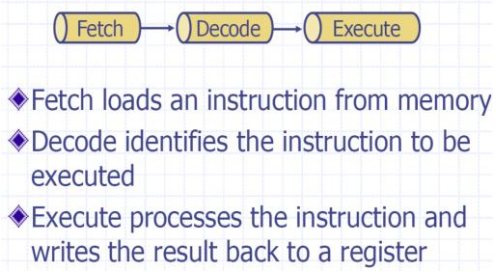
```
Row_0: MOV DPTR, #KCODE0 ; set DPTR = start of Row 0
       SJMP FIND ;          Find column key belongs to
Row_1: MOV DPTR, #KCODE1
       SJMP FIND
Row_2: MOV DPTR, #KCODE2
       SJMP FIND
Row_3: MOV DPTR, #KCODE3
       SJMP FIND

FIND: RRC A ;        Check any carry bit is low
      JNC MATCH ; if zero get ASCII code
      INC DPTR
      SJMP FIND ; keep searching

MATCH: CLR A
       MOVC A, @A+DPTR ; get ASCII from table
       MOV P0, A ; display pressed key
       LJMP L1
```

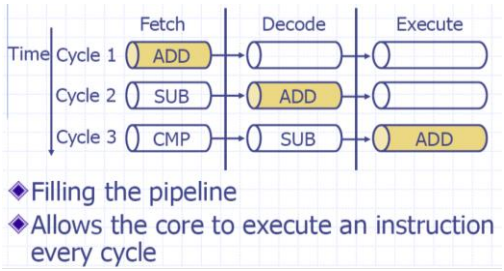
```
ORG 300H

KCODE0: DB '0','1','2','3'          ;ROW 0
KCODE1: DB '4','5','6','7'          ;ROW 1
KCODE2: DB '8','9','A','B'          ;ROW 2
KCODE3: DB 'C','D','E','F'          ;ROW 3
```

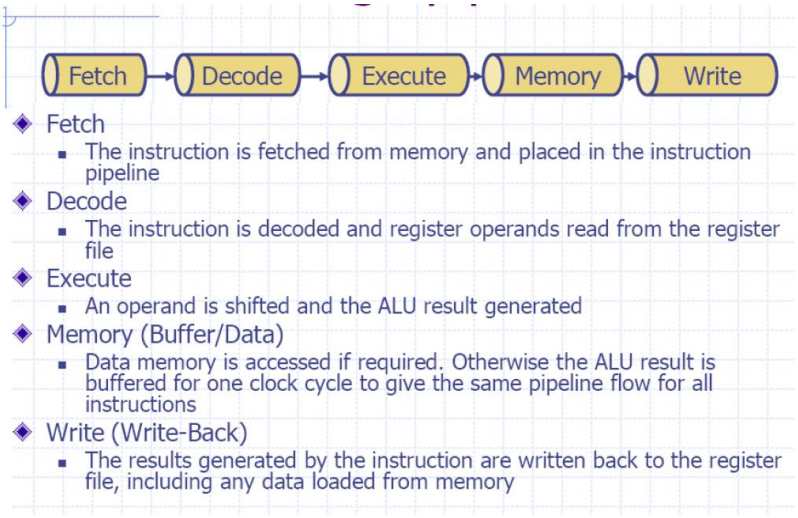
9. **Explain the 3-stage pipelining architecture of ARM7 and the 5-stage pipelining architecture of ARM9 processors with relevant diagrams depicting their pipeline structures.** **10**

ANS:

3-stage pipelining architecture of ARM7 processor:





	5-stage pipelining architecture of ARM9 processor:  //Explanation for both the architecture should also be included	
10.	**Provide a concise overview of the data processing instructions available for ARM processors.** **ANS:** Overview of the data processing instructions should contain the following topics: Data transfer instruction: Various MOV instructions; MOV instruction with Barrel shifter) Arithmetic Instructions: ADD, SUB, MUL etc. with and without Barrel shifter Logical Instructions: AND, OR, EX-OR etc. Compare Instructions: CMP, TEQ, TST etc.	**10**
	⇔⇔⇔	