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

Internal Assessment Test 3 – July2022

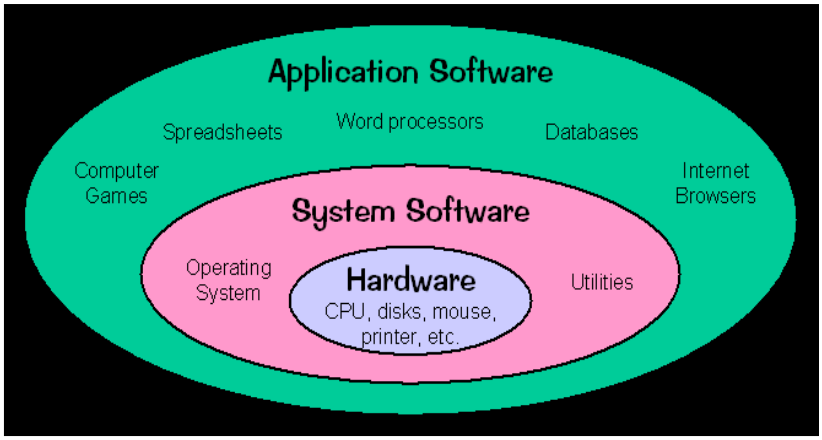
Sub:	System Software and Compiler Design					Sub Code:	18CS61	Branch:	CSE	
Date:	08/07/2022	Duration:	90 min's	Max Marks:	50	Sem/Sec:	6th / A,B,C			OBE

Answer any FIVE FULL Questions

		MARKS	CO	RBT
1 (a)	What is System software? Write the difference between system software and application software	[4]	CO1	L1

Solution:

a. System software consists of a variety of programs that support the operation of a computer.
b. The programs implemented in either software and (or) firmware (permanent softwareprogrammed into a read-only memory.) that makes the computer hardware usable.
c. The software makes it possible for the users to focus on an application or other problem tobe solved, without needing to know the details of how the machine works internally.
d. Bios (basic input output system) It is a type of Firmware used during the booting process(power-on/start up).
e. Example: Text editor, Compiler, Assembler, Loader, Linker, Debugger and Operating system.



Difference between system software and application software

1.System software is used for operating computer hardware.	Application software is used by user to perform specific task
2.System softwarees are installed on the computer when operating system is installed.	Application softwares are installed according to uthe ser's requirements
3.In general, the user does not interact with system software because it works in the	In general, the user interacts with application sofwares

	background.				
	4.System software can run independently. It provides platform for running application softwares.	Application software can't run independently. They can't run without the presence of system software.			
	5.Some examples of system softwares are compiler,assembler, debugger, driver, etc.	Some examples of application softwares are word processor, web browser, media player, etc.			
(b)	Explain SIC/XE Architecture in detail.		[6]	CO1	L2

Answer :

Solution:

Memory :

The 8 bytes are usually contained by the memory. In SIC/XE system, the maximum available memory is 1 megabyte that means 2^{20} bytes. The memory size of standard SIC is very small. Due to these memory size changes, the addressing mode and instruction format are changed in the simplified instructional computer extra expensive (SIC/XE)

Register :

Instead of the registers of SIC (simplified instructional computer), there are 4 additional general-purpose registers in the SIC/XE. That means there are total 9 registers (4 additional registers + 5 registers of SIC).

The four additional registers of SIC/XE are described as follows:

Mnemonic Number Special use

a. A 0 Accumulator (Used for arithmetic operation)
b. X 1 Index register(Used for addressing)
c. L 2 Linkage register (JSUB-jump to subroutine instruction stores the return address in this register)
d. PC 8 Program counter(Contains address of next instruction to be fetched for execution)
e. SW 9 Status word (Contains variety of information including condition code)
f. B 3 Base register; used for addressing
g. S 4 General working register
h. T 5 General working register
1 additional register, 48 bits in length
f . F 6 Floating-point accumulator (48 bits)

Data format:

The data format of a SIC standard version and SIC/XE is almost the same. There are somedifferences in data formats, which are described as follows:

1. Sign bits 0 and 1 are represented by S. Here, 1 is used to show negative, and bit 0 is usedto show positive.
2. With the help of binary numbers, the integers are represented.
3. With the help of ASCII codes, the characters are represented.
4. Exponent is a type of unsigned binary number, which is represented with the help of avalue between 0 and 2047.
5. Fraction is represented with the help of a value between 0 and 1.

The SIC/XE contains an additional floating-point data type with 48 bit, which is shown asfollows:

The value will be represented with the help of following formula:

Value = (s) * f * 2(exponent-1024)

Instruction formats:

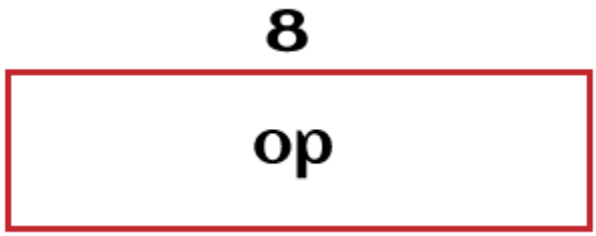
The instruction format of simplified instructive format is not enough for SIC/XE because the available memory size of SIC/XE is 2^{20} bytes. That means an address of SIC/XE cannot fit into the field of 15 bit.

There are two ways to solve the memory-related problem, which is described as follows:

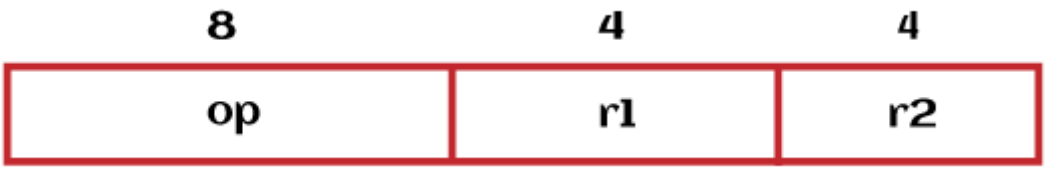
- It can be solved with the help of using **relative addressing,** which is shown by instruction format 3.
- It can be solved by **extending the address field** to 20 bits, which is shown by instructive format 4.

The SIC/XE contains four types of format. Where, format 1 and format 2 cannot be used to reference the memory. The bit 'e' is used to distinguish between format 3, and format 4.

Format 1: It is a 1-byte format. For example: HIO, NORM, SIO, TIO.

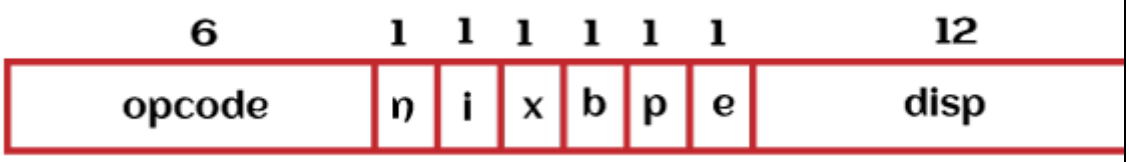


Format 2: It is a 2-byte format.



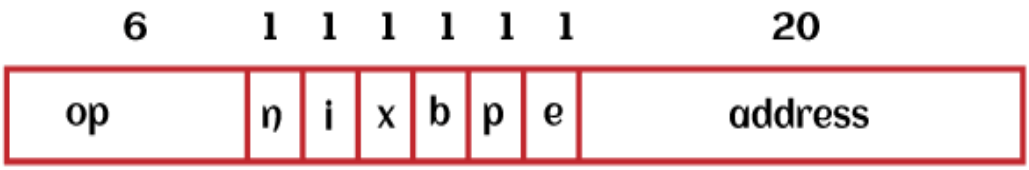
The registers are represented with the help of above two addresses. So we don't require to access the memory for execution.

Format 3: It is a 3-byte format.



The instruction will be interpreted in the form of simple SIC instruction if and only if "n bit" and "i bit" both bits are 0.

Format 4: It is a 4-byte format.



Here

- **n** is used to show the **indirect bit**
- **i** is used to show the **immediate bit**
- **x** is used to show the **index bit**
- **b** is used to show the **base bit**
- **p** is used to show the **PC relative bit**
- **e** is used to show the **extended bit**

The variation of the instruction can be interpreted with the help of the above-described bits alone or in combination, shown as follows:

Bit representation	Use
x = 1	It is used to show the indexed addressing
e = 0	It is used to show the 3-byte format
e = 1	It is used to show the 4-byte format
b = 0 and p = 1	It is used to show the PC relative addressing
b = 1 and p = 0	It is used to show the base or displacement addressing
n = 1 and i = 1	It is used to show the direct addressing

n = 1 and i = 0	It is used to show the indirect addressing
n = 0 and i = 1	It is used to show the immediate addressing
n = 0 and i = 0	It is used to show the simple SIC interpretation.

So the last 15 bits, as well as the bpe bits, are treated as an address.

2. (a)Generate object program for given program (sic)Given LDX= 04 LDA=00 ADD=18 LDCH=50JLT=38 STA=0C RSUB=4C STCH=54 [10] CO1 L3

Label	**Opcode**	**Operand**
ADDITION	START	105D
FIRST	LDX	ZERO
	LDA	FIVE
LOOP	ADD	TABLE,X
	LDCH	STR1,X
	STCH	STR2,X
	JLT	LOOP
	STA	TOTAL
	RSUB	
TABLE	RESW	1000
STR1	BYTE	C'EOF'
STR2	RESB	6
ZERO	WORD	0
FIVE	WORD	5
TOTAL	RESW	1
	END	FIRST

Solution:

address		**Label**	**Opcode**	**Operand**	
		ADDITION	START	105D	
105D	3	FIRST	LDX	ZERO	041C36
1060	3		LDA	FIVE	001C39
1063	3	LOOP	ADD	TABLE,X	189075
1066	3		LDCH	STR1,X	509C2D
1069	3		STCH	STR2,X	549C30
106C	3		JLT	LOOP	381063
106F	3		STA	TOTAL	0C1C3C
1072	3		RSUB		4C0000
1075	BB8	TABLE	RESW	1000 1000*3=3000) Hex value BB8	
1C2D	3	STR1	BYTE	C'EOF'	454F46
1C30	6	STR2	RESB	6	
1C36	3	ZERO	WORD	0	000000

1C39	3	FIVE	WORD	5	000005	
1C3C	3	TOTAL	RESW	1		
			END	FIRST		

3. (a) Write the target address for the following machine instructions [4] CO2 L2
a. 032600h b. 03C300h c. 022030h d. 010030h where PC=003000h B=006000h and X=000090

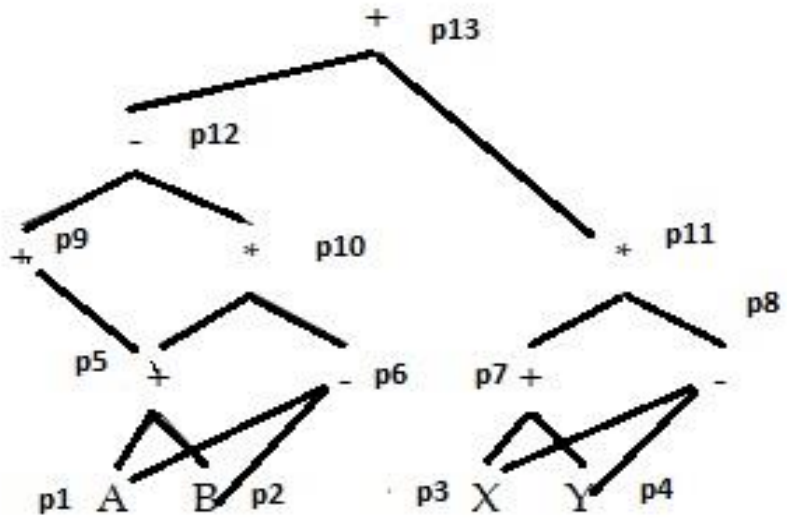
SOLUTION:

Hex	OP	n	I	x	b	P	e	disp/address	Target Addres
032600	000000	1	1	0	0	1	0	0110 0000 0000	3600
03C300	000000	1	1	1	1	0	0	0011 0000 0000	6390
022030	000000	1	0	0	0	1	0	0000 0011 0000	3030
010030	000000	0	1	0	0	0	0	0000 0011 0000	30

(b) Draw a directed acyclic graph for the given expression and show the steps of construction. [6] CO2 L3
((A+B)-((A+B)*(A-B)))+((X+Y)*(X-Y))

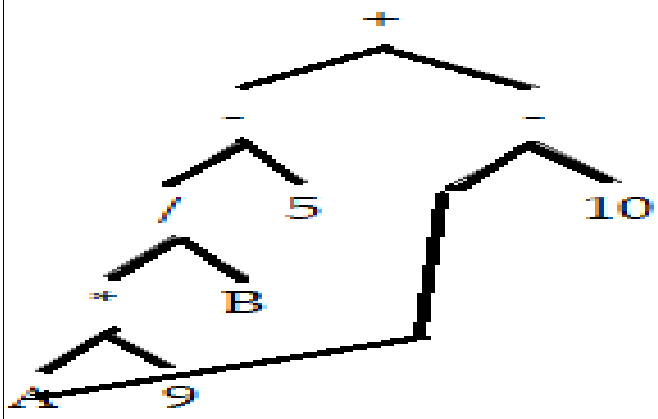
Solution:

SNO	PRODUCTION	SEMANTIC RULE
1.	E -> E + T	E.Node = new Node (+, E.Node, T.Node)
2.	E -> E - T	E.Node = new Node (-, E.Node, T.Node)
3.	E -> T	E.Node = T.Node
4.	T->T*F	T.Node = new Node (*, T.Node, F.Node)
5.	T->T/F	T.Node = new Node (/, T.Node, F.Node)
6.	T->F	T.Node = F.Node
7.	F ->(E)	F.Node = E.Node
8.	F->id	F.Node = new Lean(id, id.entry)
9.	F-> digit	FNode = new Leaf(digit, digit.val)



Steps for constructing DAG are

1. **P1=new leaf(id,entry A)**

	2. **P2=new leaf(id,entry B)** 3. **P3=new leaf(id,entry X)** 4. **P4=new leaf(id,entry Y)** 5. **P5=new node (+,P1,P2)=P9** 6. **P6=new node (-,P1,P2)** 7. **P7= new node (+,P3,P4)** 8. **P8= new node (-,P3,P4)** 9. **P10= new node (*,P5,P6)** 10. **P11= new node (*,P7,P8)** 11. **P12=new node (*,P9,P10)** 12. **P13=new node (+,P11,P12)**			
4. (a)	Draw the syntax tree and write the 3-address code for the following expression: A*9/B-5+(A-10) **Solution:** **3 address code:** **T1= A-10** **T2=A*9** **T3=T2/B** **T4=T3-5** **T5=T4+T1** **Syntax tree:**  	[4]	CO2	L1
(b)	Generate assembly-level language code (target code) for the following three addresses that p and q are in memory location y=*q q=q+4 *p=y p=p+4 **Solution:** Assembly code:	[6]	CO2	L3

```
y = *q
q = q + 4
*p = y
p = p + 4
```

answer

```
LD R1, q
LD R2, 0(R1)
ADD R1, R1, #4
ST q, R1
LD R1, p
ST 0(R1), R2
ADD R1, R1, #4
ST p, R1
```

5.	Generate intermediate code for the following statement (given w=8 bytes)
	For I from 0 to 10 do
	For j from 0 to 10 do
	a[i][j]= 0.0
	For I from 0 to 10 do
	a[i,i]=1.0
		[10]	CO2	L3

Solution:

```
1)i=1 //Leader 1 (First statement)
2) j=1 //Leader 2 (Target of 11th statement)
3) t1 = 10 * i //Leader 3 (Target of 9th statement)
4) t2 = t1 + j
5) t3 = 8 * t2
6) t4 = t3 – 88
7) a[t4] = 0.0
8) j = j + 1
9) if j <=10 goto (3)
10) i = i + 1 //Leader 4 (Immediately following
Conditional goto statement)
11) if i <= 10 goto (2)
12) i = 1 //Leader 5 (Immediately following
Conditional goto statement)
13) t5 = i - 1 //Leader 6 (Target of 17th statement)
14) t6 = 88 * t5
15) a[t6] = 1.0
16) i = i + 1 17) if i <= 10 goto (13)
```

There are **6 Basic Blocks** in the above code :

B1) Statement 1
B2) Statement 2
B3) Statement 3-9
B4) Statement 10-11
B5) Statement 12
B6) Statement 13-17

6.	Translate the assignment into 1.Three Address Code 2. Quadruple 3. Triple 4. Indirect Triple. 1. a= b* -c +b* -c 2. a = b[i] + c[j]	[6]	CO2	L3

solution:

1.Three address code

t1 = minus c
t2 = b * t1
t3 = minus c
t4 = b * t3
t5 = t2 + t4
a = t5

Quadruple

	op	arg1	arg2	result
	minus	c		t1
	*	b	t1	t2
	minus	c		t3
	*	b	t3	t4
	+	t2	t4	t5
	=	t5		a

Triple

	op	arg1	arg2
0	minus	c	
1	*	b	(0)
2	minus	c	
3	*	b	(2)
4	+	(1)	(3)
5	=	a	(4)

	op		op	arg1	arg2
35	(0)	0	minus	c	
36	(1)	1	*	b	(0)
37	(2)	2	minus	c	
38	(3)	3	*	b	(2)
39	(4)	4	+	(1)	(3)
40	(5)	5	=	a	(4)

Quadruple and triple

0) =[]	b	i	t1
1) =[]	c	j	t2
2) +	t1	t2	t3
3) =	t3		a

0) =[]	b	i	
1) =[]	c	j	
2) +	(0)	(1)	
3) =	a	(2)	

3 address code :

T1=i*4
T2=b[T1]
T3=j*4
T4=C[T3
T5=T2+T4
a=T5

indirect triple

11 →	(0)
12 →	(1)
13 →	(2)
14 →	(3)

Course Outcomes		Modules covered	PO1	PO2	PO3	PO4	PO5	PO6		PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3	PSO4
CO1	Explain system software	**1**	3	3	2	2	-	-	-	-	-	-	-	-	-	-	-	-	CO1
CO2	Design and develop lexical analyzers, parsers and code generators	**2,3,4,5**	3	3	3	3	3	2	-	-	-	-	-	-	-	-	2	-	CO2
CO3	Utilize lex and yacc tools for implementing different concepts of system software	**4**	3	3	3	3	3	2	-	-	-	-	-	-	-	-	3	-	CO3

COGNITIVE LEVEL	REVISED BLOOMS TAXONOMY KEYWORDS
L1	List, define, tell, describe, identify, show, label, collect, examine, tabulate, quote, name, who, when, where, etc.
L2	summarize, describe, interpret, contrast, predict, associate, distinguish, estimate, differentiate, discuss, extend
L3	Apply, demonstrate, calculate, complete, illustrate, show, solve, examine, modify, relate, change, classify, experiment, discover.
L4	Analyze, separate, order, explain, connect, classify, arrange, divide, compare, select, explain, infer.
L5	Assess, decide, rank, grade, test, measure, recommend, convince, select, judge, explain, discriminate, support, conclude, compare, summarize.

PROGRAM OUTCOMES (PO), PROGRAM SPECIFIC OUTCOMES (PSO)				CORRELATION LEVELS	
PO1	Engineering knowledge	PO7	Environment and sustainability	0	No Correlation
PO2	Problem analysis	PO8	Ethics	1	Slight/Low
PO3	Design/development of solutions	PO9	Individual and team work	2	Moderate/ Medium
PO4	Conduct investigations of complex problems	PO10	Communication	3	Substantial/ High
PO5	Modern tool usage	PO11	Project management and finance		
PO6	The Engineer and society	PO12	Life-long learning		
PSO1	Develop applications using different stacks of web and programming technologies				

PSO2	Design and develop secure, parallel, distributed, networked, and digital systems
PSO3	Apply software engineering methods to design, develop, test and manage software systems.
PSO4	Develop intelligent applications for business and industry