PROPOSED ASSIGNMENT (LAB WORK) ON COMPUTER SYSTEM ARCHITECTURE UNDER CBCS

SUBJECT – CMSA

PAPER – CMSACOR02P

FIRST SEMESTER

ASSEMBLY LANGUAGE PROGRAM BASED ON BASIC COMPUTER ORGANIZATION AND DESIGN CHAPTER -7, BY M. MORRIS MANO REVISED THIRD EDITION

Prepared by Joydeb Das Biswas, Dept. of Computer Science, A.P.C College

PROBLEM-1: Write an assembly language program to add n (n≥1) Integers (positive, negative and zero). The number of elements is stored at the location 150, numbers are stored after the location 150 and the sum is stored at the end.

PROBLEM-2: Write an assembly language program to find number of 1's in a word (16-bit). Word is stored at the memory location 030 and the number of 1's is stored after the memory location 030.

PROBLEM-3: Write an assembly language program to find Greatest Common Divisor (GCD) of Two Positive integers. Numbers are stored memory locations 120 and 121 and GCD is stored after the memory location 121.

PROBLEM-4: Write an assembly language program to perform XOR operation on two binary numbers using OR subroutine.

PROBLEM-5: Write an assembly language program to transfer a block of data from one place to another place (Memory). The number of elements is stored at the location 120, Source data are stored after the location 120 and Destination data are started at the location 140.

PROBLEM-6: Write an assembly language program to complement each word in a block of data (Memory). The number of elements is stored at the location 150, Data are stored after the location 150.

PROBLEM-7: Write an assembly language program to multiply two numbers using shift and add. Multiplicand and Multiplier are stored at memory locations 120 & 121 and Product is stored at memory location 122.

PROBLEM-8: Write an assembly language program to multiply two positive integer numbers using Repeated Addition. Multiplicand and Multiplier are stored at memory locations 120 & 121 and Product is stored at memory location 122**.**

PROBLEM-9: Write an assembly language program to exchange consecutive bits of a memory word using OR subroutine. Memory word is stored at the location 120 and the result is stored at the location 125.

PROBLEM-10: Write an assembly language program to find frequency of a word in an array of n (≥1) memory words. The number of elements is stored at the location 120, Source data are stored after the location 120 and Destination data are started at the location 140.

PROBLEM-11: Write an assembly language program to find maximum of n (n≥1) elements. The number of elements is stored at the location 150, elements are stored after the location 150 and maximum element is stored after the last element.

PROBLEM-12: Write an assembly language program to clear each word in a block of data (Memory). The number of elements is stored at the location 150, elements are stored after the location 150.

PROBLEM-13: Write an assembly language program to accept input two characters, pack such characters in a word and store it in memory location 030.

PROBLEM-14: Write an assembly language program to unpack two characters from a memory word stored at the location 030 and store them in consecutive memory locations 033 and 034.

PROBLEM-15: Write an assembly language program to generate first n (≥1) Fibonacci Numbers and store them in consecutive memory locations. Number n is stored at the location 30 and fibonacci numbers are stored from the location 150.

PROBLEM-16: Write an assembly language program to find number of Positive, Negative and Zero numbers from a set of n numbers. Number n is stored at the location 150, numbers are stored from the location 150 and number of Positive, Negative and Zero numbers are stored at the locations 042, 043 and 044.

PROBLEM-17: Write an assembly language program to verify whether the lower byte and the upper byte of a memory word is Equal Or Not. Memory word is at the location 041 and the status which is 0(unequal) or 1(equal) is stored at the location 044.

PROBLEM-18: Write an assembly language program to add two 32-bit words (double precision addition) with carry. First number is stored at locations 040, 041 and Second number is stored at locations 042,043 and addition & carry are stored next consecutive memory locations.

PROBLEM-19: Write an assembly language program to accept a string of characters end with '#' from Input and display such characters in Output without '#'.

PROBLEM-20: Write an assembly language program to exchange lower byte and upper byte of a memory word. Memory word is at the location 050 and after exchanging bytes, word is stored at the location 054.

Programming Basic Computer

Each line of Assembly language Program has the following format:

LABEL	MNEMONIC/PSEUDO CODE	ADDRESS	INDIRECT ADDRESSING MODE	COMMENTS
(Optional)	(Mandatory)	(Optional)	(Optional)	(Optional)

Labels should be started in **column 1** and Fields are separated by **white space**. Comments are indicated using a // (double slash) or a **;** (semi-colon)

Instruction Types:

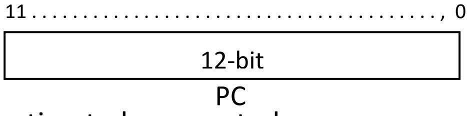
a) Mnemonic Instructions or Machine Instructions

b) Pseudo Instructions

a) Mnemonic Instructions or Machine Instructions:

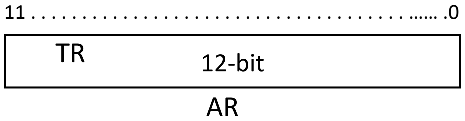
1. CPU Registers: 7 Registers

i) Program Counter(12-bit)



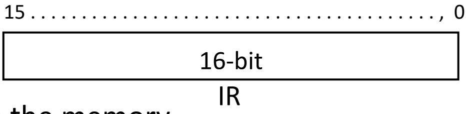
It holds the address of next instruction to be executed.

ii) Address Register(12-bit)



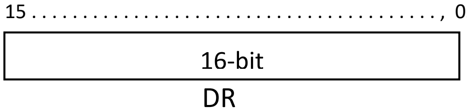
It holds the address of instruction/memory operand. It is directly connected to the Memory Unit.

iii) Instruction Register(16-bit)



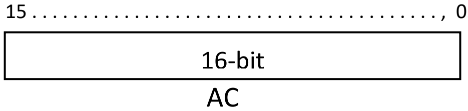
It holds the instruction read from the memory.

iv) Data Register(16-bit)



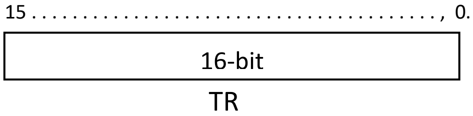
Operand read from memory is placed in data register.

v) Accumulator(16-bit)



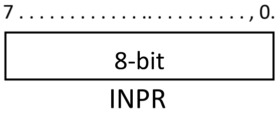
It holds the first operand before any arithmetic or logical operation and it also holds the result after the arithmetic or logical operation.

vi) Temporary Register(16-bit)



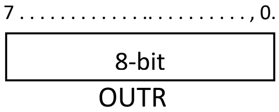
It holds temporary data during instruction execution.

vii) Input Register(8-bit)



It holds 8-bit input character from an Input Device.

viii) Output Register(8-bit)



It holds 8-bit character for an Output Device.

All registers are Special Purpose Registers (SPR).

2. 1-bit Flip-Flop Used.

I is an 1-bit Flip-Flop which contains the left most bit position of the instruction. In case MRI, I is used to specify addressing mode (I=0/1) and otherwise it specifies RRI (I=0) or IOI(I=1).

E is an 1-bit Flip-Flop (Extended Accumulator Bit) which contains the output carry after the execution of ADD instruction. It can be set or complemented and it also contains the shifted bit after the execution of CIR or CIL.

S is an 1-bit Flip-Flop (Start-Stop Bit) which is initialized to 1. It specifies whether the CPU is Halt or Not. When CPU is halt, it set to 0, otherwise it is 1.

FGI (Control Flip-Flop) is a 1-bit Input Flag. It is initially set to 0. It is used for checking whether new information is available in the input device, when it is so, it is set 1 and new information is shifted to INPR and after that, information from INPR is transferred to AC and FGI is set to 0.

FGO (Control Flip-Flop) is a 1-bit Output Flag. It is initially set to 1. It is used for checking whether FGO flag is set or not, if it is set, then information from AC is shifted to OUTR and FGO is cleared to 0. The output device accepts the information and print it and FGO is set to 1.

R is an 1-bit Flip-Flop (Interrupt Flip-Flop) and it is initialized to 0. It is set to 1 when an interrupt is raised and it proceeds with an interrupt cycle in the next instruction cycle and, otherwise it proceeds with an Instruction cycle.

IEN is an 1-bit Flip-Flop (Interrupt Enable Flip-Flop) which is used to take decision whether or not to use interrupt facility(by ION or IOF).

3. Memory: It consists of 4096 (2^{12} memory locations) words; each of size is 16-bit.

INSTRUCTION FORMAT

Instruction Length – 16-bit. All instructions have the same length.

Instruction read from memory is placed in IR (Instruction Register).

Instruction Types

i) Memory Reference Instructions(MRI).
ii) Register Reference Instructions(RRI).
iii) Input-Output Instructions(IOI).

i) Memory Reference Instructions(MRI)

15	14 13 12	11 10 . . . . .1 0
I	OPCODE	12-bit Memorv Address

IR

It is specified by Operation Code ≠ **111** with a **0** or **1** in the left most bit position of the instruction.

a) Bit 15 of IR is transferred to 1-bit Flip-Flop **I**.

IR(15) – 1-bit specifies the **Addressing Mode**

i) IR(15)=0 for **Direct** Addressing Mode
[12-bit memory address contains the operand]

ii) IR(15)=1 for **Indirect** Addressing Mode
[12-bit memory address contains the address of the operand]

b) Next 3 bits IR (14,13,12) specifies **Operation Code** (Opcode). Operation Codes for MRI are 7 operations 000(AND) , 001(ADD) ,010(LDA),011(STA),100(BUN), 101(BSA) , 110(ISZ). 7 basic Instructions and total 14 instructions.

c) Next 12 bits IR (11, 10,………., 1,0) specifies operand address or address of location of the operand.

MRI	Symb ol	Hex code	
		I=0(DA)	I=1(IAD)
1	AND	0XXX	8XXX
2	ADD	1XXX	9XXX
3	LDA	2XXX	AXXX
4	STA	3XXX	BXXX
5	BUN	4XXX	CXXX
6	BSA	5XXX	DXXX
7	ISZ	6XXX	EXXX

<u>MRI for Direct Addressing Mode</u>

1) AND XXX ; AC ←AC ∧ M[XXX] 2) ADD XXX ; AC ←AC + M[XXX]
3) LDA XXX ; AC ← M[XXX] 4) STA XXX ; M[XXX] ← AC
5) BUN XXX ; PC ← XXX 6) BSA XXX ; M[XXX] ← PC, XXX ← XXX + 1,PC ← XXX
7) ISZ XXX ; M[XXX] ← M[XXX] + 1, IF (M[XXX] = 0) THEN PC ← PC + 1

<u>MRI for Indirect Addressing Mode</u>

1) AND XXX I ; AC ←AC ∧ M[M[XXX]] 2) ADD XXX I ; AC ←AC + M[M[XXX]]
3) LDA XXX I ; AC ←M[M[XXX]] 4) STA XXX I ; M[M[XXX]] ← AC
5) BUN XXX I ; PC ← M[XXX] 6) BSA XXX I ; M[M[XXX]] ← PC, M[XXX] ← M[XXX]+1,PC ← M[XXX]
7) ISZ XXX I; M[M[XXX]] ← M[M[XXX]] + 1, IF (M[M[XXX]] = 0) THEN PC ← PC + 1

Maximum Number T-States=7 and Minimum Number T-States=5.
Maximum Number Machine Cycles=4 and Minimum Number Machine Cycles =1.

ii) Register Reference Instructions (RRI)

15 .14 13 12. .	11 . . . . ………………………………….0.
0 1 1 1	Register Operation(12-bit)

IR

Register reference instruction is specified by Operation Code =**111** with a **0** in the left most bit position of the instruction (IR (15) =0).

A register reference instruction specifies an operation on **AC**, **S** (Start-Stop bit) or **E** (Extended accumulator bit) or test on **AC** or **E**.

No operand from the memory is needed in this type of instruction. Each bit from position 11-bit to 0-bit specifies a unique operation. There are 12 instructions.

RRI	Symbol	Hex code
1	CLA	7800
2	CLE	7400
3	CMA	7200
4	CME	7100
5	CIR	7080
6	CIL	7040
7	INC	7020
8	SPA	7010
9	SNA	7008
10	SZA	7004
11	SZE	7002
12	HLT	7001

Clear Instruction - i) CLA ;AC ← 0
ii) CLE ;E ← 0

Complement Instruction - iii) CMA ;AC ← $\overline{AC}$
iv) CME ; E ← $\overline{E}$

Shift Instruction - v) CIR ;AC ←SHR(AC); AC(15) ← E;E ← AC(0)
vi) CIL ;AC ←SHL(AC); AC(0) ← E;E ← AC(15)

Increment Instruction - vii) INC ;AC ← AC + 1

Skip and Test Instruction - viii) SPA ;if (AC(15) = 0) then PC← PC+1
ix) SNA if (AC(15) = 1) then PC← PC+1
x) SZA if (AC= 0) then PC← PC+1
xi) SZE if (E = 0) then PC← PC+1

Halt Instruction - xii) Halt ;S ← 0;

Maximum Number and Minimum Number T-States=4. Maximum Number and Minimum Number Machine Cycle =1.

iii) Input-Output Instructions (IOI)

15 .14 13 12. . 11 . . . .0.

1 1 1 1	I/O operation

IR

Input-Output instruction is specified by Operation Code =**111** with a **1** in the left most bit position of the instruction (IR (15) =1).

An Input-Output Instruction specifies an operation on AC, FGI, FGO, INPR, OUTR or IEN (Interrupt Enable) or test on FGI or FGO.

No operand from the memory is needed in this type of instruction. Each bit from position 11-bit to 6-bit specifies a unique I/O operation. There are only 6 instructions.

IOI	Symbol	Hex code
1	INP	F800
2	OUT	F400
3	SKI	F200
4	SKO	F100
5	ION	F080
6	IOF	F040

Input Character - i) INP ;AC[7…..0] ← INPR; FGI ← 0

Output Character - ii) OUT ;OUTR ← AC[7…..0]; FGO ← 0

Skip on Input Flag - iii) SKI ;IF (FGI = 1) THEN PC ← PC+1;

Skip on Output Flag- iv) SKO IF (FGO = 1) THEN PC ← PC+1

Interrupt Enable On - v) ION IEN ← 1

Interrupt Enable Of - vi) IOF ;IEN ← 0

Maximum Number and Minimum Number T-States=4. Maximum Number and Minimum Number Machine Cycle =1.

b) Pseudo Instructions(Assembler Directives):

i) **ORG** **N ;**It specifies that hexadecimal number N is the memory address of the instruction/operand listed in the following line.

ii) **DEC** **N ;** It converts signed decimal number N into binary and puts into a memory location.

iii) **HEX** **N ;** It converts hexadecimal number N into binary and puts into a memory location either a hex constant or a label.

iv) **END** **;** It specifies the end of assembly language program(assembler does not consider any assembly language after end)

v) **CHR** **;** It puts a character into a memory location in lower byte using single quotes.

PROBLEM-1: Write an assembly language program to add n (n≥1) Integers (positive, negative and zero). The number of elements is stored at the location 150, numbers are stored after the location 150 and the sum is stored at the end.

Answer.

INSTRUCTION NO.	LABEL (Optional)	OPCODE/PSEUDO CODE (Mandatory)	ADDRESS (Optional)	INDIRECT ADDRESSING MODE (Optional)	COMMENTS (Optional)
1		ORG	100		PROGRAM STARTS AT LOC 100
2		LDA	APTR	I	
3		CMA			
4		INC			GET -N
5		STA	NPTR		
6		ISZ	APTR		
7		CLA			CLEAR AC
8		CLE			CLEAR E
9	BACK	ADD	APTR	I	
10		ISZ	APTR		
11		ISZ	NPTR		
12		BUN	BACK		
13		STA	APTR	I	
14		HLT			HALT CPU
15		ORG	120		
16	APTR	HEX	150		
17	NPTR	DEC	0		
18		ORG	150		
19		DEC	5		NUMBER OF ELEMENTS
20		DEC	7		
21		DEC	8		
22		DEC	9		
23		DEC	10		
24		DEC	12		
25		END			PROGRAM END

SYMBOLS

SYMBOL	ADDRESS
APTR	120
NPTR	121
BACK	107

TEST DATA SET

INPUT	OUTPUT
5 7,8,9,10,12	46(2E)
5 -7,8,9,10,12	32(20)
5 -7,8,9,-10,12	12(0C)
1 -7	-7(FFF9)

Program is tested on the Computer Simulator(ComputerSimulator16,Nicholas Duchon) based on Basic Computer design,Chapter-7 , Revised 3rd Edition by Moris M Mano.

PROBLEM-2: Write an assembly language program to find number of 1's in a word (16-bit). Word is stored at the memory location 030 and the number of 1's is stored after the memory location 030.

Answer.

INSTRUCTION NO.	LABEL (Optional)	OPCODE/PSEUDO CODE (Mandatory)	ADDRESS (Optional)	INDIRECT ADDRESSING MODE (Optional)	COMMENTS (Optional)
1		ORG	10		PROGRAM STARTS AT LOC 100
2		CLA			
3		LDA	XXX		MEMORY WORD INTO AC
4	BACK	SZA			
5		BUN	NEXT		
6		BUN	DONE		
7	NEXT	ISZ	CNTR		INCREMENT COUNTER
8		STA	XXX		
9		LDA	ONE		
10		CMA			
11		INC			GET -1
12		ADD	XXX		A-1
13		AND	XXX		A & (A-1)
14		BUN	BACK		
15	DONE	HLT			HALT CPU
16		ORG	30		
17	XXX	HEX	FFFA		
18	CNTR	DEC	0		
19	ONE	DEC	1		
20		END			PROGRAM END

SYMBOLS

SYMBOL	ADDRESS
XXX	30
CNTR	31
ONE	32
DONE	1D
BACK	12
NEXT	15

TEST DATA SET

INPUT	OUTPUT
FFFA	14(0E)
0	0(0)
0101	2(2)

Program is tested on the Computer Simulator(ComputerSimulator16,Nicholas Duchon) based on Basic Computer design,Chapter-7 , Revised 3rd Edition by Moris M Mano.

PROBLEM-3: Write an assembly language program to find Greatest Common Divisor (GCD) of Two Positive integers. Numbers are stored memory locations 120 and 121 and GCD is stored after the memory location 121.

Answer.

INSTRUCTION NO.	LABEL (Optional)	OPCODE/PSEUDO CODE (Mandatory)	ADDRESS (Optional)	INDIRECT ADDRESSING MODE (Optional)	COMMENTS (Optional)
1		ORG	100		
2	BACK	LDA	AVAL		
3		CMA			
4		INC			GET -A
5		ADD	BVAL		GET B-A
6		SZA			B-A=0?
7		BUN	NEXT		B-A≠0
8		BUN	DONE		B-A=0
9	NEXT	SPA			B-A>0?
10		BUN	BIGAVAL		B-A<0
11		STA	BVAL		B-A>0
12		BUN	BACK		
13	BIGAVAL	LDA	BVAL		
14		CMA			
15		INC			GET -B
16		ADD	AVAL		A-B
17		STA	AVAL		
18		BUN	BACK		
19	DONE	LDA	BVAL		
20		STA	GCD		
21		HLT			HALT CPU
22		ORG	120		
23	AVAL	DEC	30		
24	BVAL	DEC	48		
25	GCD	DEC	0		
26		END			PROGRAM END

SYMBOLS

SYMBOL	ADDRESS
AVAL	120
BVAL	121
GCD	122
DONE	111
BACK	100
NEXT	107
BIGAVAL	10B

TEST DATA SET

INPUT	OUTPUT
30,48	6
31,97	1
7,7	7

Program is tested on the Computer Simulator(ComputerSimulator16,Nicholas Duchon) based on Basic Computer design,Chapter-7 , Revised 3rd Edition by Moris M Mano.

PROBLEM-4: Write an assembly language program to perform XOR operation on two binary numbers using OR subroutine.

Answer.

INSTRUCTION NO.	LABEL (Optional)	OP CODE/PSEUDO CODE (Mandatory)	ADDRESS (Optional)	INDIRECT ADDRESSING MODE (Optional)	COMMENTS (Optional)
1		ORG	100		
2		LDA	XXX		
3		CMA			GET A'
4		AND	YYY		A'B
5		STA	RRR		RRR ← A'B
6		LDA	YYY		
7		CMA			GET B'
8		AND	XXX		AB'
9		STA	SSS		SSS ← **AB**'
10		BSA	OR		JUMP TO SUBROUTINE **OR**
11		STA	XOR		XOR ← A'B + AB'
12		HLT			HALT CPU
13		ORG	120		
14	XXX	HEX	1010		
15	YYY	HEX	1011		
16	RRR	HEX	0		
17	SSS	HEX	0		
18	XOR	HEX	0		
19		ORG	130		
20	OR	HEX	0		SUBROUTINE OR
21		LDA	SSS		
22		CMA			GET B'
23		STA	SSS		
24		LDA	RRR		
25		CMA			GET **A'**
26		AND	SSS		
27		CMA			GET (**A'** B')' = A∨B
28		BUN	OR	I	
29		END			PROGRAM END

SYMBOLS

SYMBOL	ADDRESS
XXX	120
YYY	121
RRR	122
SSS	123
XOR	124
OR	130

TEST DATA SET

INPUT	OUTPUT
1010,1011	1
3,7	4
0,0	0

Program is tested on the Computer Simulator(ComputerSimulator16,Nicholas Duchon) based on Basic Computer design,Chapter-7 , Revised 3rd Edition by Moris M Mano.

Problem Design, Solved and Tested by Joydeb Das Biswas, Dept. of Computer Science, APC College.

PROBLEM-5: Write an assembly language program to transfer a block of data from one place to another place (Memory). The number of elements is stored at the location 120, Source data are stored after the location 120 and Destination data are started at the location 140.

Answer.

INSTRUCTION NO.	LABEL (Optional)	OP CODE/PSEUDO CODE (Mandatory)	ADDRESS (Optional)	INDIRECT ADDRESSING MODE (Optional)	COMMENTS (Optional)
1		ORG	100		
2		LDA	APTR	I	
3		CMA			
4		INC			GET –N
5		STA	NPTR		
6		ISZ	APTR		
7	BACK	LDA	APTR	I	
8		STA	BPTR	I	
9		ISZ	APTR		
10		ISZ	BPTR		
11		ISZ	NPTR		
12		BUN	BACK		
13		HLT			HALT CPU
14		ORG	110		
15	APTR	HEX	120		
16	BPTR	HEX	140		
17	NPTR	DEC	0		
18		ORG	120		
19		DEC	5		
20		HEX	1101		
21		HEX	11		
22		HEX	11		
23		HEX	0000		
24		HEX	FF		
25		ORG	140		
26		HEX	0		
27		HEX	0		
28		HEX	0		
29		HEX	0		
30		HEX	0		
31		END			PROGRAM END

SYMBOLS

SYMBOL	ADDRESS
APTR	110
BPTR	111
NPTR	112
BACK	105

TEST DATA SET

INPUT	OUTPUT
SOURCE 5, 1101,11,11,0000,FF	SOURCE 5, 1101,11,11,0000,FF

PROBLEM-6: Write an assembly language program to complement each word in a block of data (Memory). The number of elements is stored at the location 150, Data are stored after the location 150.

Answer.

INSTRUCTION NO.	LABEL (Optional)	OP CODE/PSEUDO CODE (Mandatory)	ADDRESS (Optional)	INDIRECT ADDRESSING MODE (Optional)	COMMENTS (Optional)
1		ORG	100		
2		LDA	APTR	I	
3		CMA			
4		INC			Get –N
5		STA	NPTR		
6	BACK	ISZ	APTR		
7		LDA	APTR	I	
8		CMA			
9		STA	APTR	I	
10		ISZ	NPTR		
11		BUN	BACK		
12		HLT			
13		ORG	120		
14	APTR	HEX	150		
15	NPTR	HEX	0		
16		ORG	150		
17		HEX	5		NO. OF ELEMENTS
18		HEX	4		
19		HEX	0		
20		HEX	3		
21		HEX	6		
22		HEX	110		
23		END			PROGRAM END

SYMBOLS

SYMBOL	ADDRESS
APTR	120
NPTR	121
BACK	104

TEST DATA SET

INPUT	OUTPUT
SOURCE 5, 4,0,3,6,110	SOURCE 5, fffb,ffff,fffc,fff9,feef

Program is tested on the Computer Simulator(ComputerSimulator16,Nicholas Duchon) based on Basic Computer design,Chapter-7 , Revised 3rd Edition by Moris M Mano.

Problem Design, Solved and Tested by Joydeb Das Biswas, Dept. of Computer Science, APC College.

PROBLEM-7: Write an assembly language program to multiply two numbers using shift and add. Multiplicand and Multiplier are stored at memory locations 120 & 121 and Product is stored at memory location 122.

Answer.

INSTRUCTION NO.	LABEL (Optional)	OP CODE/PSEUDO CODE (Mandatory)	ADDRESS (Optional)	INDIRECT ADDRESSING MODE (Optional)	COMMENTS (Optional)
1		ORG	100		
2	BACK	CLE			
3		LDA	BVAL		
4		CIR			RIGHT SHIFT MULTIPLIER
5		STA	BVAL		
6		SZE			
7		BUN	EONE		
8		BUN	EZERO		
9	EONE	LDA	AVAL		
10		ADD	PARSUM		ADD SHIFTED MULTIPLICAND WITH PARTIAL SUM
11		STA	PARSUM		
12		CLE			
13	EZERO	LDA	AVAL		
14		CIL			LEFT SHIFT MULTIPLICAND
15		STA	AVAL		
16		ISZ	NCOUNT		
17		BUN	BACK		
18		HLT			HALT CPU
19		ORG	120		
20	AVAL	DEC	7		MULTIPLICAND
21	BVAL	DEC	5		MULTIPLIER
22	NCOUNT	DEC	-16		
23	PPROD	DEC	0		
24		END			PROGRAM END

SYMBOLS

SYMBOL	ADDRESS
AVAL	120
BVAL	121
NCOUNT	122
PPROD	123
BACK	100
EONE	107
EZERO	10B

TEST DATA SET

INPUT	OUTPUT
7 , 5	35(23)
8,5	40(28)

Program is tested on the Computer Simulator(ComputerSimulator16,Nicholas Duchon) based on Basic Computer design,Chapter-7 , Revised 3rd Edition by Moris M Mano.

PROBLEM-8: Write an assembly language program to multiply two positive integer numbers using Repeated Addition. Multiplicand and Multiplier are stored at memory locations 120 & 121 and Product is stored at memory location 122.

Answer.

INSTRUCTION NO.	LABEL (Optional)	OP CODE/PSEUDO CODE (Mandatory)	ADDRESS (Optional)	INDIRECT ADDRESSING MODE (Optional)	COMMENTS (Optional)
1		ORG	100		
2		LDA	BVAL		
3		CMA			
4		INC			
5		STA	BVAL		
6		CLA			
7	AGAIN	ADD	AVAL		
8		ISZ	BVAL		
9		BUN	AGAIN		
10		STA	PROD		
11		HLT			HALT CPU
12		ORG	120		
13	AVAL	DEC	8		MULTIPLICAND
14	BVAL	DEC	5		MULTIPLIER
15	PROD	DEC	0		
16		END			PROGRAM END

SYMBOLS

SYMBOL	ADDRESS
AVAL	120
BVAL	121
PROD	122
AGAIN	105

TEST DATA SET

INPUT	**OUTPUT**
8 , 5	40(28)
1 3	3(3)

Program is tested on the Computer Simulator(ComputerSimulator16,Nicholas Duchon) based on Basic Computer design,Chapter-7 , Revised 3rd Edition by Moris M Mano.

Problem Design, Solved and Tested by Joydeb Das Biswas, Dept. of Computer Science, APC College.

PROBLEM-9: Write an assembly language program to exchange consecutive bits of a memory word using OR subroutine. Memory word is stored at the location 120 and the result is stored at the location 125.

Answer.

INSTRUCTION NO.	LABEL (Optional)	OP CODE/PSEUDO CODE (Mandatory)	ADDRESS (Optional)	INDIRECT ADDRESSING MODE (Optional)	COMMENTS (Optional)
1		ORG	100		
2		CLE			
3		LDA	XXX		
4		AND	EPONE		
5		CIR			
6		STA	TTT		
7		CLE			
8		LDA	XXX		
9		AND	OPONE		
10		CIL			
11		STA	SSS		
12		BSA	OR		JUMP TO SUBROUTINE OR
13		STA	RESULT		
14		HLT			HALT CPU
15		ORG	120		
16	XXX	HEX	9999		
17	EPONE	HEX	AAAA		
18	OPONE	HEX	5555		
19	TTT	HEX	0		
20	SSS	HEX	0		
21	RESULT	HEX	0		
22		ORG	130		
23	OR	HEX	0		SUBROUTINE OR
24		LDA	SSS		
25		CMA			
26		STA	SSS		
27		LDA	TTT		
28		CMA			
29		AND	SSS		
30		CMA			
31		BUN	OR	I	
32		END			PROGRAM END

SYMBOLS

SYMBOL	ADDRESS
XXX	120
EPONE	121
OPONE	122
TTT	123
SSS	124
RESULT	125
OR	130

TEST DATA SET

INPUT	**OUTPUT**
1001 1001 1001 1001	0110 0110 0110 0110
1010 1010 0101 1111	0101 0101 1010 1111

Program is tested on the Computer Simulator(ComputerSimulator16,Nicholas Duchon) based on Basic Computer design,Chapter-7 , Revised 3rd Edition by Moris M Mano. Problem Design, Solved and Tested by Joydeb Das Biswas, Dept. of Computer Science, APC College.

PROBLEM-10: Write an assembly language program to find frequency of a word in an array of n (≥1) memory words. The number of elements is stored at the location 120, Source data are stored after the location 120 and Destination data are started at the location 140.

Answer.

INSTRUCTION NO.	LABEL (Optional)	OP CODE/PSEUDO CODE (Mandatory)	ADDRESS (Optional)	INDIRECT ADDRESSING MODE (Optional)	COMMENTS (Optional)
1		ORG	100		
2		LDA	APTR	I	
3		CMA			
4		INC			
5		STA	NPTR		
6		ISZ	APTR		
7	BACK	LDA	APTR		
8		CMA			
9		INC			
10		ADD	KVAL		
11		SZA			
12		BUN	NEXT		
13		ISZ	FCOUNT		
14	NEXT	ISZ	APTR		
15		ISZ	NPTR		
16		BUN	BACK		
17		HLT			HALT CPU
18		ORG	120		
19	APTR	HEX	150		
20	NPTR	DEC	0		
21	KVAL	HEX	A		
22	FCOUNT	DEC	0		
23		ORG	150		
24		DEC	5		
25		DEC	7		
26		DEC	A		
27		DEC	3		
28		DEC	A		
29		DEC	C		
30		END			PROGRAM END

SYMBOLS

SYMBOL	ADDRESS
AGAIN	17
APTR	40
NPTR	41
NEXT	1A
MAX	156
DONE	24

TEST DATA SET

INPUT	OUTPUT
5 ,KEY VALUE(A) 0007 000A 0003 000A 000C	2
5 ,KEY VALUE(6) 0007 0009 0003 0004 000C	0

Program is tested on the Computer Simulator(ComputerSimulator16,Nicholas Duchon) based on Basic Computer design,Chapter-7 , Revised 3rd Edition by Moris M Mano.

Problem Design, Solved and Tested by Joydeb Das Biswas, Dept. of Computer Science, APC College.

PROBLEM-11: Write an assembly language program to find maximum of n (n≥1) elements. The number of elements is stored at the location 150, elements are stored after the location 150 and maximum element is stored after the last element.

Answer.

INSTRUCTION NO.	LABEL (Optional)	OP CODE/PSEUDO CODE (Mandatory)	ADDRESS (Optional)	INDIRECT ADDRESSING MODE (Optional)	COMMENTS (Optional)
1		ORG	10		
2		LDA	APTR	I	
3		CMA			
4		INC			
5		STA	NPTR		
6		ISZ	APTR		
7		LDA	APTR	I	
8		STA	MAX		
9	AGAIN	ISZ	NPTR		
10		BUN	NEXT		
11		BUN	DONE		
12	NEXT	ISZ	APTR		
13		LDA	APTR	I	
14		CMA			
15		INC			
16		ADD MAX			
17		SNA			
18		BUN AGAIN			
19		LDA	APTR	I	
20		STA	MAX		
21		BUN	AGAIN		
22	DONE	HLT			HALT CPU
23		ORG	40		
24	APTR	HEX	150		
25	NPTR	DEC	0		
26		ORG	150		
27		DEC	5		NO. OF ELEMENTS
28		DEC	7		
29		DEC	2		
30		DEC	5		
31		DEC	9		
32		DEC	3		
33	MAX	DEC	0		
34		END			PROGRAM END

SYMBOLS

SYMBOL	ADDRESS
AGAIN	17
APTR	40
NPTR	41
NEXT	1A
MAX	156
DONE	24

TEST DATA SET

INPUT	OUTPUT
5 0007 ,0002 0009 0005 0003	9
1 0011	11(B)

Program is tested on the Computer Simulator(ComputerSimulator16,Nicholas Duchon) based on Basic Computer design,Chapter-7 , Revised 3rd Edition by Moris M Mano.

Problem Design, Solved and Tested by Joydeb Das Biswas, Dept. of Computer Science, APC College.

PROBLEM-12: Write an assembly language program to clear each word in a block of data (Memory). The number of elements is stored at the location 150, elements are stored after the location 150.

Answer.

INSTRUCTION NO.	LABEL (Optional)	OP CODE/PSEUDO CODE (Mandatory)	ADDRESS (Optional)	INDIRECT ADDRESSING MODE (Optional)	COMMENTS (Optional)
1		ORG	100		
2		LDA	APTR	I	
3		CMA			
4		INC			Get –N
5		STA	NPTR		
6	BACK	ISZ	APTR		
7		LDA	APTR	I	
8		CLA			
9		STA	APTR	I	
10		ISZ	NPTR		
11		BUN	BACK		
12		HLT			HALT CPU
13		ORG	120		
14	APTR	HEX	150		
15	NPTR	HEX	0		
16		ORG	150		
17		DEC	5		
18		DEC	4		
19		DEC	0		
20		DEC	3		
21		DEC	6		
22		DEC	110		
23		END			PROGRAM END

SYMBOLS

SYMBOL	ADDRESS
APTR	120
NPTR	121
BACK	104

TEST DATA SET

INPUT	OUTPUT
5 0004 0000 0003 0006 0110	5 0000 0000 0000 0000 0000

Program is tested on the Computer Simulator(ComputerSimulator16,Nicholas Duchon) based on Basic Computer design,Chapter-7 , Revised 3rd Edition by Moris M Mano.

PROBLEM-13: Write an assembly language program to accept input two characters, pack such characters in a word and store it in memory location 030.

Answer.

INSTRUCTION NO.	LABEL (Optional)	OP CODE/PSEUDO CODE (Mandatory)	ADDRESS (Optional)	INDIRECT ADDRESSING MODE (Optional)	COMMENTS (Optional)
1		ORG	10		
2	BACK	SKI			
3		BUN	BACK		
4		CLA			
5		INP			
6		CIL			
7		CIL			
8		CIL			
9		CIL			
10		CIL			
11		CIL			
12		CIL			
13		CIL			
14	AGAIN	SKI			
15		BUN	AGAIN		
16		INP			
17		STA	PCHAR		
18		HLT			HALT CPU
19		ORG	30		
20	PCHAR	HEX	0		
21		END			PROGRAM END

SYMBOLS

SYMBOL	ADDRESS
BACK	10
AGAIN	1B
PCHAR	30

TEST DATA SET

INPUT	OUTPUT
A B	PCHAR – AB(4142)

Program is tested on the Computer Simulator(ComputerSimulator16,Nicholas Duchon) based on Basic Computer design,Chapter-7 , Revised 3rd Edition by Moris M Mano.

Problem Design, Solved and Tested by Joydeb Das Biswas, Dept. of Computer Science, APC College.

PROBLEM-14: Write an assembly language program to unpack two characters from a memory word stored at the location 030 and store them in consecutive memory locations 033 and 034.

Answer.

INSTRUCTION NO.	LABEL (Optional)	OP CODE/PSEUDO CODE (Mandatory)	ADDRESS (Optional)	INDIRECT ADDRESSING MODE (Optional)	COMMENTS (Optional)
1		ORG	10		
2		LDA	XXX		
3		AND	YYY		
4		STA	ZZZ		
5		LDA	XXX		
6		AND	TTT		
7		CLE			
8		CIR			
9		CIR			
10		CIR			
11		CIR			
12		CIR			
13		CIR			
14		CIR			
15		CIR			
16		STA	SSS		
17		HLT			HALT CPU
18		ORG	30		
19	XXX	HEX	4241		
20	YYY	HEX	FF		
21	TTT	HEX	FF00		
22	ZZZ	CHR	'0'		
23	SSS	CHR	'0'		
24		END			PROGRAM END

SYMBOLS

SYMBOL	ADDRESS
XXX	30
YYY	31
TTT	32
ZZZ	33
SSS	34

TEST DATA SET

INPUT	OUTPUT

M[030]=AB	M[033]=A(0041)
	M[034]=B(0042)

Program is tested on the Computer Simulator(ComputerSimulator16,Nicholas Duchon) based on Basic Computer design,Chapter-7 , Revised 3rd Edition by Moris M Mano.

Problem Design, Solved and Tested by Joydeb Das Biswas, Dept. of Computer Science, APC College.

PROBLEM-15: Write an assembly language program to generate first n (≥1) Fibonacci Numbers and store them in consecutive memory locations. Number n is stored at the location 30 and Fibonacci numbers are stored from the location 150.

Answer.

INSTRUCTIO N NO.	LABEL (Optional)	OP CODE/PSEUDO CODE (Mandatory)	ADDRESS (Optional)	INDIRECT ADDRESSING MODE (Optional)	COMMENTS (Optional)
1		ORG	10		
2		LDA	NUMADD		
3		CMA			
4		INC			
5		STA	NPTR		
6		LDA	FFIB		
7		STA	APTR	I	
8		ISZ	NPTR		
9		BUN	NEXT		
10		BUN	DONE		
11	NEXT	ISZ	APTR		
12		LDA	SFIB		
13		STA	APTR	I	
14	BACK	ISZ	NPTR		
15		BUN	AFTER		
16		BUN	DONE		
17	AFTER	ISZ	APTR		
18		LDA	FFIB		
19		ADD	SFIB		
20		STA	APTR	I	
21		LDA	SFIB		
22		STA	FFIB		
23		LDA	APTR	I	
24		STA	SFIB		
25		BUN	BACK		
26	DONE	HLT			HALT CPU
27		ORG	30		
28	NUMADD	DEC	9		
29	FFIB	DEC	0		
30	SFIB	DEC	1		
31	APTR	HEX	150		
32	NPTR	DEC	0		
33		END			PROGRAM END

SYMBOLS

SYMBOL	ADDRESS
NUMADD	30
FFIB	31
SFIB	32
APTR	33
NPTR	34
NEXT	19
BACK	1C
AFTER	1F
DONE	28

TEST DATA SET

INPUT	OUTPUT
9	0000 0001 0001 0002 0003 0005 0008 000d 0015
1	0000

Program is tested on the Computer Simulator(ComputerSimulator16,Nicholas Duchon) based on Basic Computer design,Chapter-7 , Revised 3rd Edition by Moris M Mano. Problem Design, Solved and Tested by Joydeb Das Biswas, Dept. of Computer Science, APC College.

PROBLEM-16: Write an assembly language program to find number of Positive, Negative and Zero numbers from a set of n numbers. Number n is stored at the location 150, numbers are stored from the location 150 and number of Positive, Negative and Zero numbers are stored at the locations 42, 43 and 44.

Answer.

INSTRUCTIO N NO.	LABEL (Optional)	OP CODE/PSEUDO CODE (Mandatory)	ADDRESS (Optional)	INDIRECT ADDRESSING MODE (Optional)	COMMENTS (Optional)
1		ORG	10		
2		LDA	APTR	I	
3		CMA			
4		INC			
5		STA	NPTR		
6	BACK	ISZ	APTR		
7		CLA			
8		ADD	APTR	I	
9		SNA			
10		BUN	PORZ		
11		ISZ	NVAL		
12		BUN	NEXTNUM		
13	PORZ	SZA			
14		BUN	POSIT		
15		ISZ	ZVAL		
16		BUN	NEXTNUM		
17	POSIT	ISZ	PVAL		
18	NEXTNUM	ISZ	NPTR		
19		BUN	BACK		
20		HLT			HALT CPU
21		ORG	40		
22	APTR	HEX	150		
23	NPTR	DEC	0		
24	PVAL	DEC	0		
25	NVAL	DEC	0		
26	ZVAL	DEC	0		
27		ORG	150		
28		DEC	6		
29		DEC	7		
30		DEC	-4		
31		DEC	0		
33		DEC	-7		
34		DEC	8		
35		DEC	0		
36		END			PROGRAM END

SYMBOLS

SYMBOL	ADDRESS
APTR	40
NPTR	41
PVAL	42
NVAL	43
ZVAL	44
BACK	14
POSIT	1f
PORZ	1b
NEXTNUM	20

TEST DATA SET

INPUT	OUTPUT
6 7 , -4 ,0 ,-7 ,8 , 0	0002 0002 0002
1 3	0001

Program is tested on the Computer Simulator(ComputerSimulator16,Nicholas Duchon) based on Basic Computer design,Chapter-7 , Revised 3rd Edition by Moris M Mano. Problem Design, Solved and Tested by Joydeb Das Biswas, Dept. of Computer Science, APC College.

PROBLEM-17: Write an assembly language program to verify whether the lower byte and the upper byte of a memory word is Equal Or Not. Memory word is at the location 041 and the status which is 0(unequal) or 1(equal) is stored at the location 044.

Answer.

INSTRUCTION NO.	LABEL (Optional)	OP CODE/PSEUDO CODE (Mandatory)	ADDRESS (Optional)	INDIRECT ADDRESSING MODE (Optional)	COMMENTS (Optional)
1		ORG	10		
2		LDA	XXX		
3		AND	YYY		MASK LOWER BYTE
4		CMA			
5		INC			GET –LOWER BYTE
6		STA	ZZZ		
7		LDA	XXX		
8		AND	TTT		MASK UPPER BYTE
9		CLE			
10		CIR			
11		CIR			
12		CIR			
13		CIR			
14		CIR			
15		CIR			
16		CIR			
17		CIR			GET UPPER BYTE INTO
18		ADD	ZZZ		UPPER BYTE - LOWER
19		SZA			CHECK FOR EQUAL
20		BUN	NEXT		
21		INC			
22		BUN	DONE		
23	NEXT	CLA			
24	DONE	STA	SSS		
25		HLT			HALT CPU
26		ORG	40		
27	XXX	HEX	AABB		
28	YYY	HEX	FF		
29	ZZZ	HEX	0000		
30	TTT	HEX	FF00		
31	SSS	DEC	0		
33		END			PROGRAM END

SYMBOLS

SYMBOL	ADDRESS
XXX	40
YYY	41
ZZZ	42
TTT	43
SSS	44
NEXT	25
DONE	26

TEST DATA SET

INPUT	OUTPUT
AABB	0
DADA	1

Program is tested on the Computer Simulator(ComputerSimulator16,Nicholas Duchon) based on Basic Computer design,Chapter-7 , Revised 3rd Edition by Moris M Mano. Problem Design, Solved and Tested by Joydeb Das Biswas, Dept. of Computer Science, APC College.

PROBLEM-18: Write an assembly language program to add two 32-bit words (double precision addition) with carry. First number is stored at locations 40, 41 and Second number is stored at locations 42,43 and addition & carry are stored next consecutive memory locations.

Answer.

INSTRUCTION NO.	LABEL (Optional)	OP CODE/PSEUDO CODE (Mandatory)	ADDRESS (Optional)	INDIRECT ADDRESSING MODE (Optional)	COMMENTS (Optional)
1		ORG	10		
2		CLE			
3		LDA	LAWORD		
4		ADD	LBWORD		
5		STA	LCWORD		
6		CLA			
7		CIL			
8		ADD	HAWORD		
9		SZE			
10		BUN	NEXT		
11		BUN	NSUM		
12	NEXT	ISZ	CARRY		
13		CLE			
14	NSUM	ADD	HBWORD		
15		STA	HCWORD		
16		SZE			
17		ISZ	CARRY		
18		HLT			HALT CPU
19		ORG	40		
20	LAWORD	HEX	FFFF		
21	HAWORD	HEX	FFFF		
22	LBWORD	HEX	EEEE		
23	HBWORD	HEX	FFFF		
24	LCWORD	HEX	0		
25	HCWORD	HEX	0		
26	CARRY	HEX	0		
27		END			PROGRAM END

SYMBOLS

SYMBOL	ADDRESS
LAWORD	40
HAWORD	41
LBWORD	42
HBWORD	43
LCWORD	44
HCWORD	45
CARRY	46
NEXT	1A
NSUM	1C

TEST DATA SET

INPUT		OUTPUT	
		SUM	CARRY
HAWORD LAWORD	FFFF FFFF	FFFF EEED	1
HBWORD LBWORD	FFFF EEEE		
HAWORD LAWORD	1 FFFF	1 EEED	1
HBWORD LBWORD	FFFF EEEE		
HAWORD LAWORD	11 10	24 22	0
HBWORD LBWORD	13 12		

The above program will be run also for decimal data set.

Program is tested on the Computer Simulator(ComputerSimulator16,Nicholas Duchon) based on Basic Computer design,Chapter-7 , Revised 3rd Edition by Moris M Mano. Problem Design, Solved and Tested by Joydeb Das Biswas, Dept. of Computer Science, APC College.

PROBLEM-19: Write an assembly language program to accept a string of characters end with '#' from Input and display such characters in Output without '#'.

Answer.

INSTRUCTION NO.	LABEL (Optional)	OP CODE/PSEUDO CODE (Mandatory)	ADDRESS (Optional)	INDIRECT ADDRESSING MODE (Optional)	COMMENTS (Optional)
1		ORG	10		
2		LDA	ENDCH		
3		CMA			
4		INC			
5		STA	ENDCH		
6	AGAIN	SKI			
7		BUN	AGAIN		
8		CLA			
9		INP			
10		STA	TEMP		
11		ADD	ENDCH		
12		SZA			
13		BUN	NEXT		
14		BUN	DONE		
15	NEXT	LDA	TEMP		
16		OUT			
17		BUN	AGAIN		
18	DONE	HLT			HALT CPU
19		ORG	40		
20	ENDCH	CHR	'#'		
21	TEMP	HEX	0		
22		END			PROGRAM END

SYMBOLS

SYMBOL	ADDRESS
ENDCH	40
TEMP	41
DONE	20
AGAIN	14
NEXT	1D

TEST DATA SET

INPUT	OUTPUT
ABCD#	ABCD
#	
12345#	12345

Program is tested on the Computer Simulator(ComputerSimulator16,Nicholas Duchon) based on Basic Computer design,Chapter-7 , Revised 3rd Edition by Moris M Mano.

Problem Design, Solved and Tested by Joydeb Das Biswas, Dept. of Computer Science, APC College.

PROBLEM-20: Write an assembly language program to exchange lower byte and upper byte of a memory word. Memory word is at the location 050 and after exchanging bytes, word is stored at the location 054.
Ans.

INSTRUCTION NO.	LABEL (Optional)	OP CODE/PSEUDO CODE (Mandatory)	ADDRESS (Optional)	INDIRECT ADDRESSING MODE (Optional)	COMMENTS (Optional)
1		ORG	10		
2		LDA	XXX		
3		AND	UMASK		
4		CLE			
5		CIR			
6		CIR			
7		CIR			
8		CIR			
9		CIR			
10		CIR			
11		CIR			
12		CIR			UPPER BYTE IN LOWER
13		STA	TEMP		
14		LDA	XXX		
15		AND	LMASK		
16		CLE			
17		CIL			
18		CIL			
19		CIL			
20		CIL			
21		CIL			
22		CIL			
23		CIL			
24		CIL			LOWER BYTE IN UPPER
25		ADD	TEMP		
26		STA	RESULT		
27		HLT			HALT CPU
28		ORG	50		
29	XXX	HEX	A245		
30	UMASK	HEX	FF00		
31	LMASK	HEX	FF		
32	TEMP	HEX	0		
33	RESULT	HEX	0		
34		END			PROGRAM END

SYMBOLS

SYMBOL	ADDRESS
XXX	50
UMASK	51
LMASK	52
TEMP	53
RESULT	54

TEST DATA SET

INPUT	OUTPUT
A245	45A2
0011	1100

Program is tested on the Computer Simulator(ComputerSimulator16,Nicholas Duchon) based on Basic Computer design,Chapter-7 , Revised 3rd Edition by Moris M Mano. Problem Design, Solved and Tested by Joydeb Das Biswas, Dept. of Computer Science, APC College.