Laboratory Manual

Computer Organization & Architecture
(CS-404)

For

Second Year Students
Department: Computer Science & Engineering

INSTITUTE OF TECHNOLOGY & MANAGEMENT
GWALIOR • MP • INDIA

Department of Computer Science and Engineering

Vision of CSE Department:

The department envisions to nurture students to become technologically proficient,research competent and socially accountable for the welfare of the society.

Mission of the CSE Department:

I. To provide high quality education through effective teaching-learning processemphasizing active participation of students.

II. To build scientifically strong engineers to cater to the needs of industry, higherstudies, research and startups.

III. To awaken young minds ingrained with ethical values and professional behaviorsfor the betterment of the society.

Program Educational Objectives:

Graduates will be able to

I. Our engineers will demonstrate application of comprehensive technical knowledge for innovation and entrepreneurship.

II. Our graduates will employ capabilities of solving complex engineering problemsto succeed in research and/or higher studies.

III. Our graduates will exhibit team-work and leadership qualities to meet stakeholderbusiness objectives in their careers.

IV. Our graduates will evolve in ethical and professional practices and enhancesocioeconomic contributions to the society.

INSTITUTE OF TECHNOLOGY & MANAGEMENT
GWALIOR • MP • INDIA

Program Outcomes (POs):

Engineering Graduates will be able to:

1. **Engineering knowledge**: Apply the knowledge of mathematics, science, engineering Fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis**: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions**: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems**: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society**: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability**: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

INSTITUTE OF TECHNOLOGY & MANAGEMENT
GWALIOR • MP • INDIA

Course Outcomes
COA(CS-404)

CO1:	Able to identify the basic structure of a processor, memory, lnstructions to analyze the working of a system
CO2:	Analyse the working of microprogrammed controller with firmware and hardwired control unit.
CO3:	Interpreting the computer arithmetic operations with structuring the flowchart and hardware algorithms .
CO4:	Classify and analyse the memory structure , input output organization and multiprocessors in a computer system.
CO5:	Able to implement mnemonics using assembler in assembly level language for executing instructions.

Course	Course Outcomes	CO Attainment	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
CO1	Able to identify the basic structure of a processor, memory, lnstructions to analyzethe working of a system..		2	1												2	
CO2	Analyse the working of microprogrammed controller with firmwareand hardwired control.		2									1			2		
CO3	Interpreting the computer arithmetic operations with structuring the flowchart and hardwarealgorithms ..		2	1							1						
CO4	Classify and analyse the memory structure , input output organization and multiprocessors in a computer system		1	1			1								2		
CO5	Able to implement mnemonics using assembler in assemblylevel language for executing instructions.		1				1										2

INSTITUTE OF TECHNOLOGY & MANAGEMENT
GWALIOR • MP • INDIA

List of Program

Program-1

1. Understanding how to implement functions using multiplexers.
2. To study demultiplexer.

Theory:

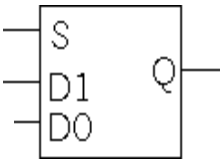
❖ **Multiplexers:**

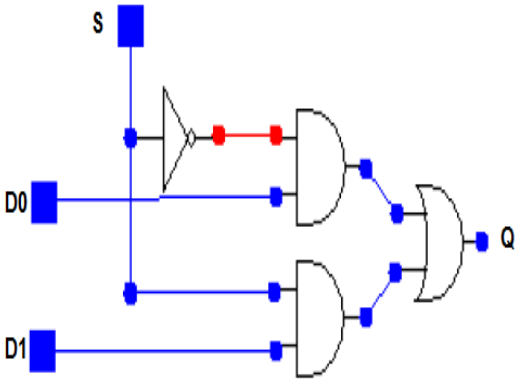
In electronics, a **multiplexer** (or mux) is a device that selects one of several analogor digital input signals and forwards the selected input into a single line. A multiplexer of 2^n inputs has **n** select lines, which are used to select which input lineto send to the output. A 2^n-to-1 multiplexer sends one of 2^n input lines to a single output line.

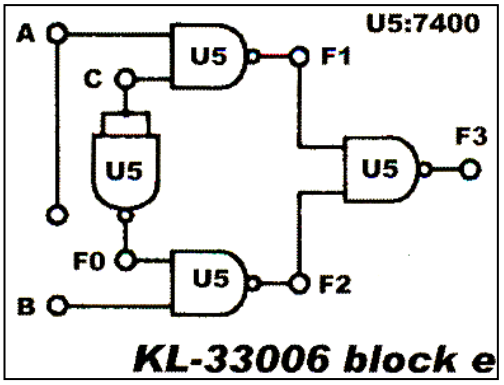
– A multiplexer has two sets of inputs:
• 2^n data input lines
• n select lines, to pick one of the 2^n data inputs
– The mux output is a single bit, which is one of the 2^n data inputs.

- **2-to-1 Mux**

The simplest multiplexer is a 2-to-1 mux







Q = S' D0 + S D1

The select bit S controls which of the data bits D0-D1 is chosen:
– If S=0, then D0 is the output (Q=D0).
– If S=1, then D1 is the output (Q=D1).

Here is a full truth table for this 2-to-1 mux, based on the equation:Q = S'

D0 + S D1

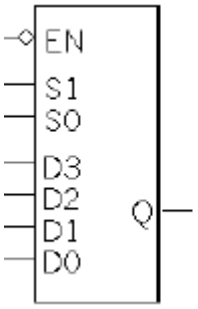
here is another kind of abbreviated truth table.

S	Q
0	D0
1	D1

- **4-to-1 Mux**

Here is a block diagram and abbreviated truth table for a 4-to-1 mux.

- Be careful! In Logic Works the multiplexer has an active-low EN input signal. When EN' = 1, the mux always outputs 1.



EN'	S1	S0	Q
0	0	0	D0
0	0	1	D1
0	1	0	D2
0	1	1	D3
1	x	x	1

Q = S1' S0' D0 + S1' S0 D1 + S1 S0' D2 + S1 S0 D3

Implementing functions with multiplexers:

Muxes can be used to implement arbitrary functions. For a function of n variables follow these steps:

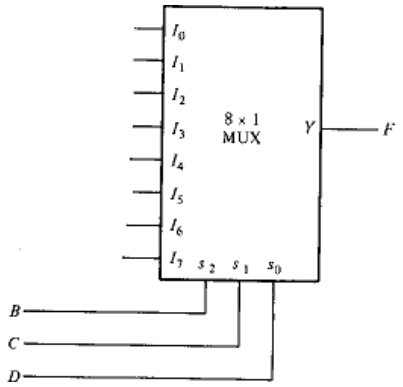
3. Select the type of Mux [2^{n-1}-to-1].
4. Select (n-1) as selection line.
5. The other input connects as input.

Implement following function with multiplexer:

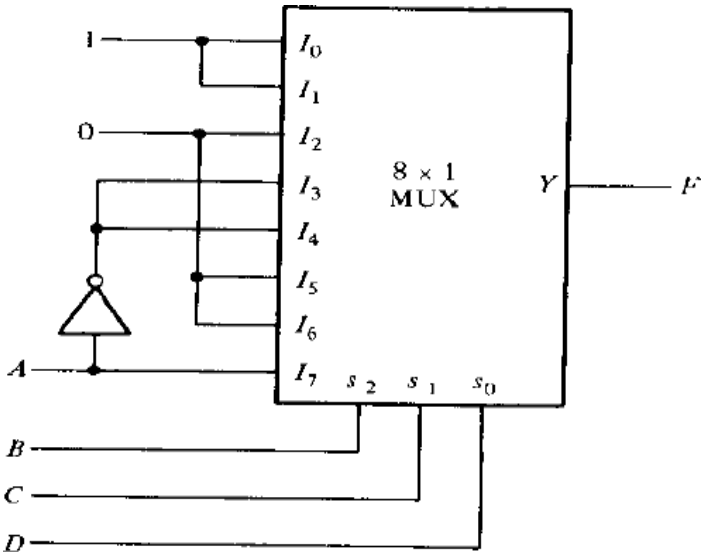
$$F(A, B, C, D) = \Sigma(0, 1, 3, 4, 8, 9, 15)$$

Solution:

1. The type of Mux [2^3-to-1] == 8-to-1 mux
2. Select (3) as selection line. == For example (B, C, and D)
3. The other input connects as input. == (A)



	I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7
A'	⓪	①	2	③	④	5	6	7
A	⑧	⑨	10	11	12	13	14	⑮
	1	1	0	A'	A'	0	0	A

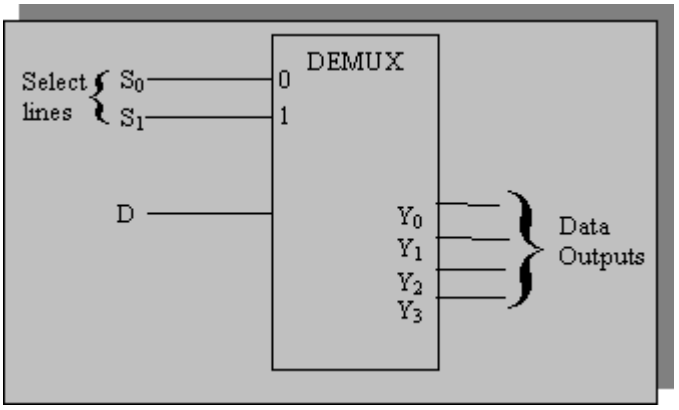


In terms of B

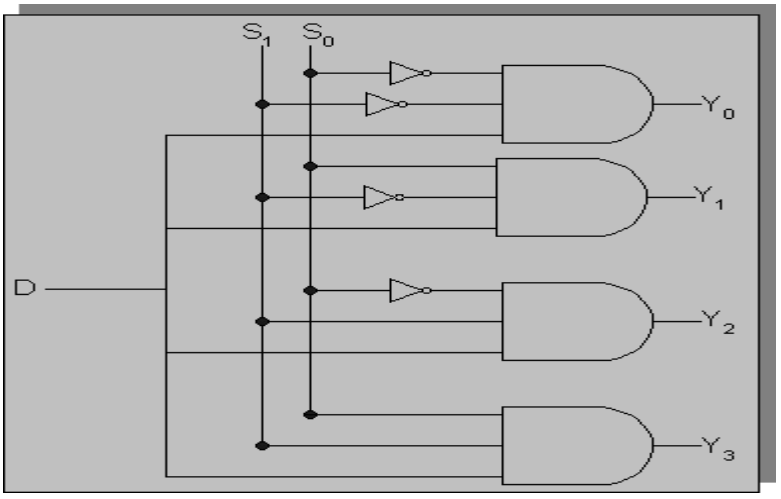
	I0	I1	I2	I3	I4	I5	I6	I7
B'B	0	1	2	3	8	9	10	11
1	4	5 B'	6 0	7 B'	12 B'	13 B'	14 0	15

❖ **Demultiplexers:**

The Demultiplexer is combinational logic circuit that performs the reverse operation of Multiplexer. It has only one input, n selectors and 2^n outputs. Depending on the combination of the select lines, one of the outputs will beselected to take the state of the input.
The following figure shows the block diagram and the truth table for 1x4 Demultiplexer.
By applying logic '1' to the input, the circuit will do the same function of the typical2-to-4 Decoder.



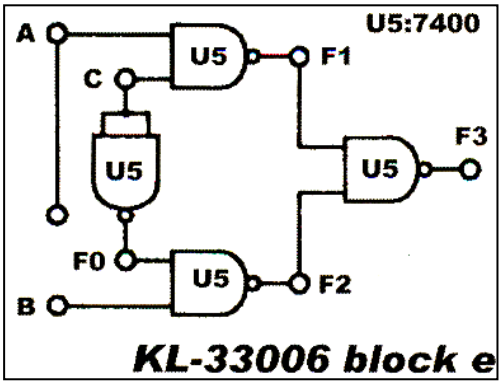
Data	Address		Outputs			
	S_1	S_0	Y_0	Y_1	Y_2	Y_3
D	0	0	D	0	0	0
D	0	1	0	D	0	0
D	1	0	0	0	D	0
D	1	1	0	0	0	D



3. Lab Work:

Part 1: 2-to-1 Mux

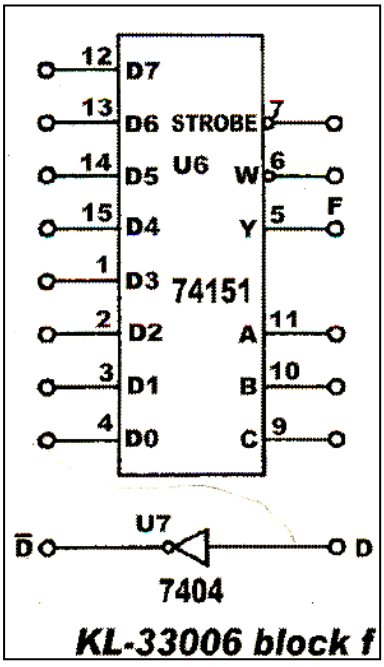
1) Construct 2-to-1 Mux using KL-33006 block e. (D1=A, D0=B, S=C). Connect inputs A, B to SW0 and SW1. Connect input C to SW3.



C	B	A	F3
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

Part 2: 8-to-1 Mux

a) Using Mux KL-33006 block f connect inputs DO~D7 to DIP Switch 1.0~1.7; inputs C, B, A to DATA Switches SW2, SW1, SW0 respectively, strobe to 0, Y and W to L0, L1 respectively then complete this table.



D7	D6	D5	D4	D3	D2	D1	D0	C	B	A	Y
0	0	0	0	0	0	1	1	0	0	0	
0	0	0	0	0	0	1	0	0	0	1	
0	0	0	1	0	1	0	1	0	1	0	
0	0	0	0	1	0	0	0	0	1	1	
1	0	0	1	0	0	0	0	1	0	0	
0	0	1	0	0	0	0	0	1	0	1	
0	1	1	0	0	0	0	0	1	1	0	
0	0	0	0	0	0	0	0	1	1	1	

a) Construct the function,

F (A, B, C, D) = Σ (0, 1, 3, 4, 8, 9, 15)

from example 7.1 in terms of A (A isinput) using Mux KL-33006 block f.

4. Exercises:

1) Implement F (A, B, C, D) = ∏ (3, 8, 12) using Mux:
a) In terms of C.
b) In terms of D.

2) Implement 8-to-1 mux using tow 4-to-1 mux and one 2-to-1 mux.

3) Implement 1-to-4 dmux using 1-to-2 dmux.

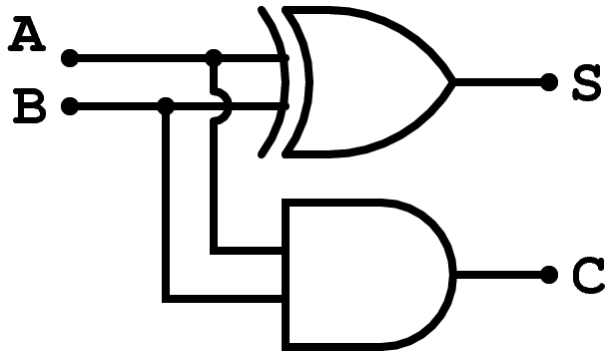
Program-2

To design the circuit of half adder.

IC USED: 7486(X-OR), 7408(AND).

THEORY: A half adder is a logical circuit that performs an additional operation on two binary digits.The half adder produces a sum and a carry value which are both binary digits.

A half adder circuit has two inputs A and B and two outputs – S representing sum and C representing carry.

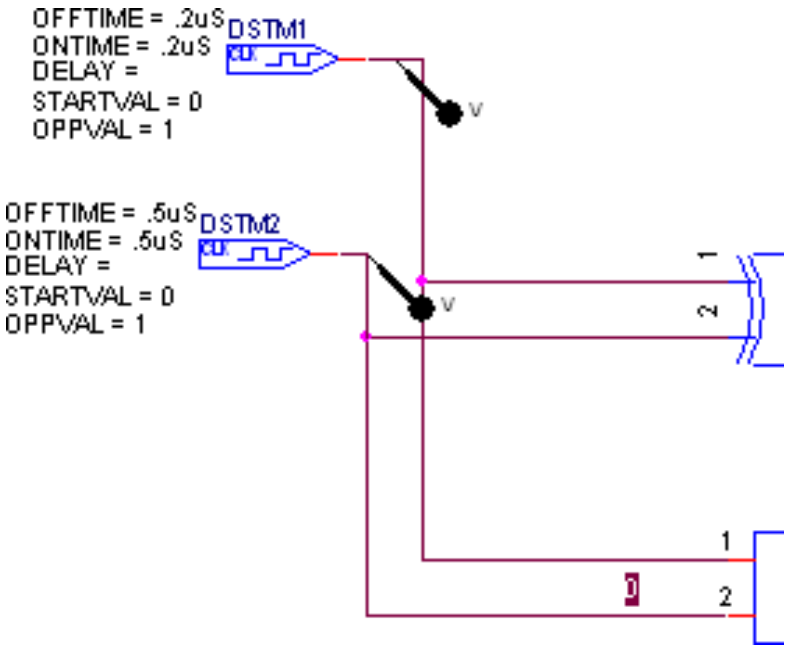


S = A xor B i.e. (A'B + AB')

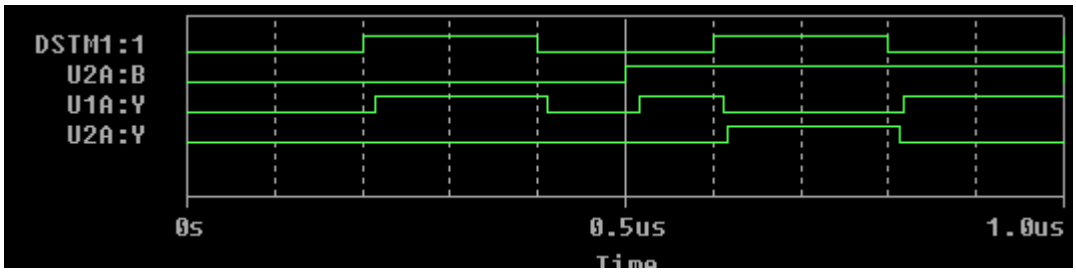
C = A and B i.e. (A.B)

TRUTH TABLE:

SCHEMATIC DIAGRAM:



WAVEFORM:



RESULT: The output waveform of half adder is verified.

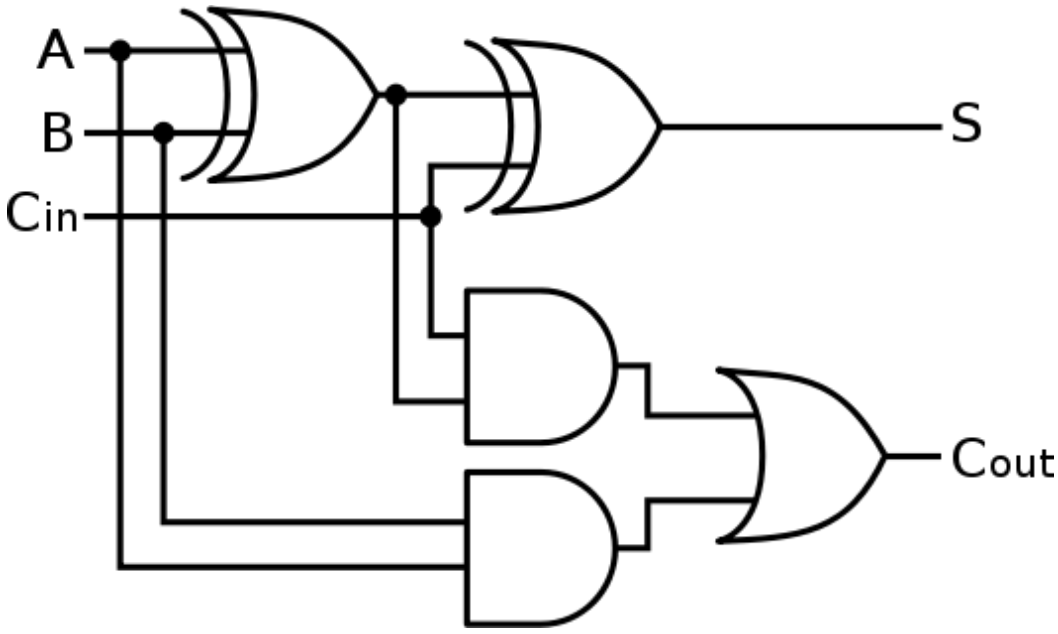
Program-3

To design the circuit of full adder.

IC USED: 7486(X-OR), 7408(AND), 7432(OR).

THEORY: A full adder is a logical circuit that performs an additional operation on three binary digits.The half adder produces a sum and a carry value which are both binary digits.

A full adder circuit has three inputs A,B and Cin and two outputs – S representing sum and Cout representing carry.



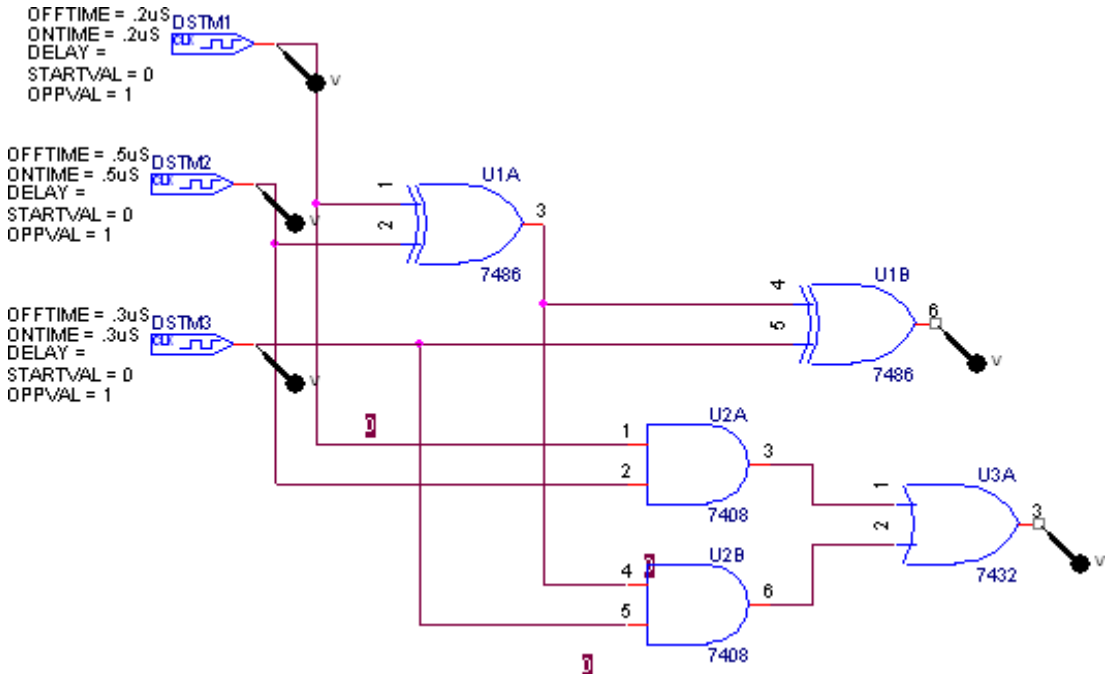
S = A xor B xor C

C = A.B +C(A xor B)

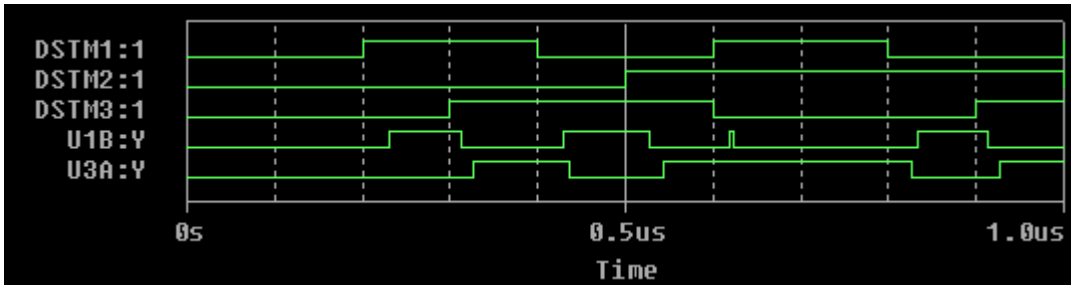
TRUTH TABLE:

0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

SCHEMATIC DIAGRAM:



WAVEFORM:



RESULT: The output waveform of full adder is verified.

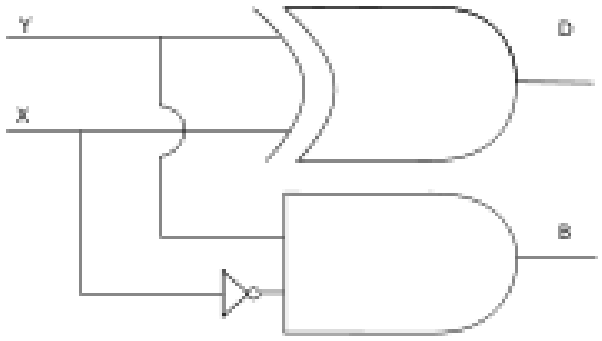
Program-4

To design the circuit of half subtractor

IC USED: 7486(X-OR), 7408(AND), 7404(NOT).

THEORY: A half subtractor is a logical circuit that performs an subtraction operation on two binarydigits. The half subtractor produces a Difference and a borrow value which are both binary digits.

A half subtractor circuit has two inputs X, Y and two outputs – D representing difference and B representing borrow.



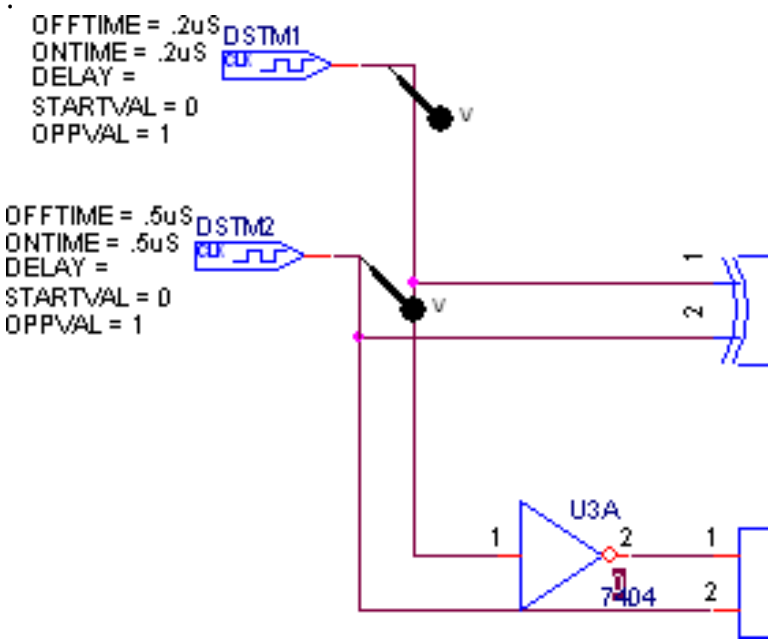
D = A xor B i.e. (A'B

+ AB')B = A'B

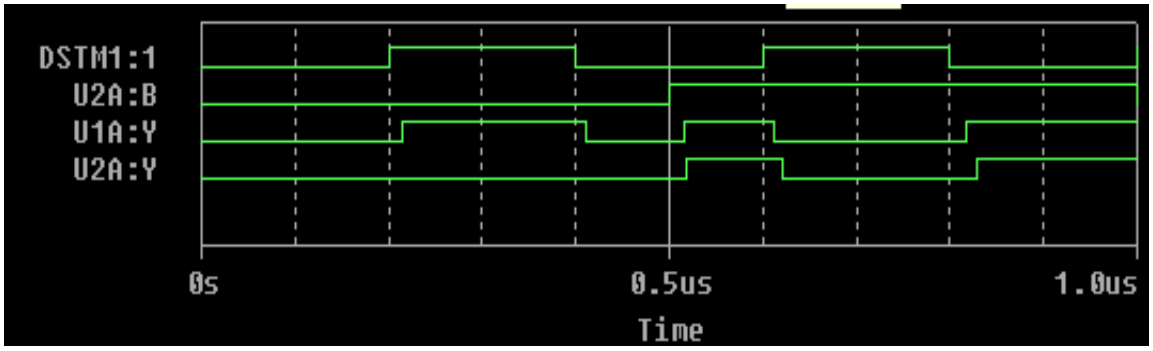
TRUTH TABLE:

SCHEMATIC DIAGRAM

:



WAVEFORM:



RESULT: The output waveform of half subtractor is verified.

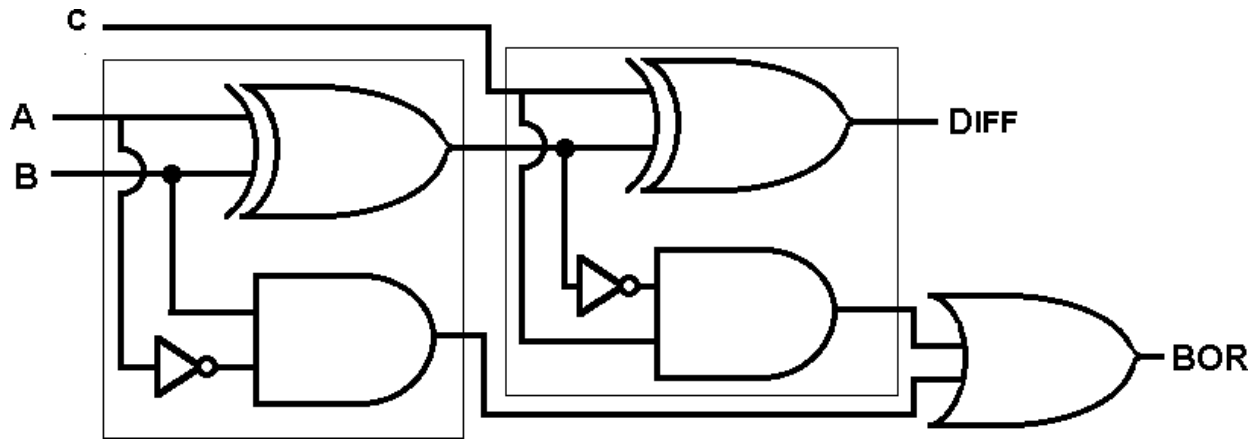
Program-5

To design the circuit of full subtractor.

IC USED: 7486(X-OR), 7408(AND), 7432(OR),7404(NOT).

THEORY: A full subtractor is a logical circuit that performs an subtraction operation on three binarydigits. The full subtractor produces a difference and a borrow value which are both binary digits.

A Full adder circuit has three inputs A,B and C and two outputs – DIFF representing difference andBOR representing borrow.



S = A xor B xor C

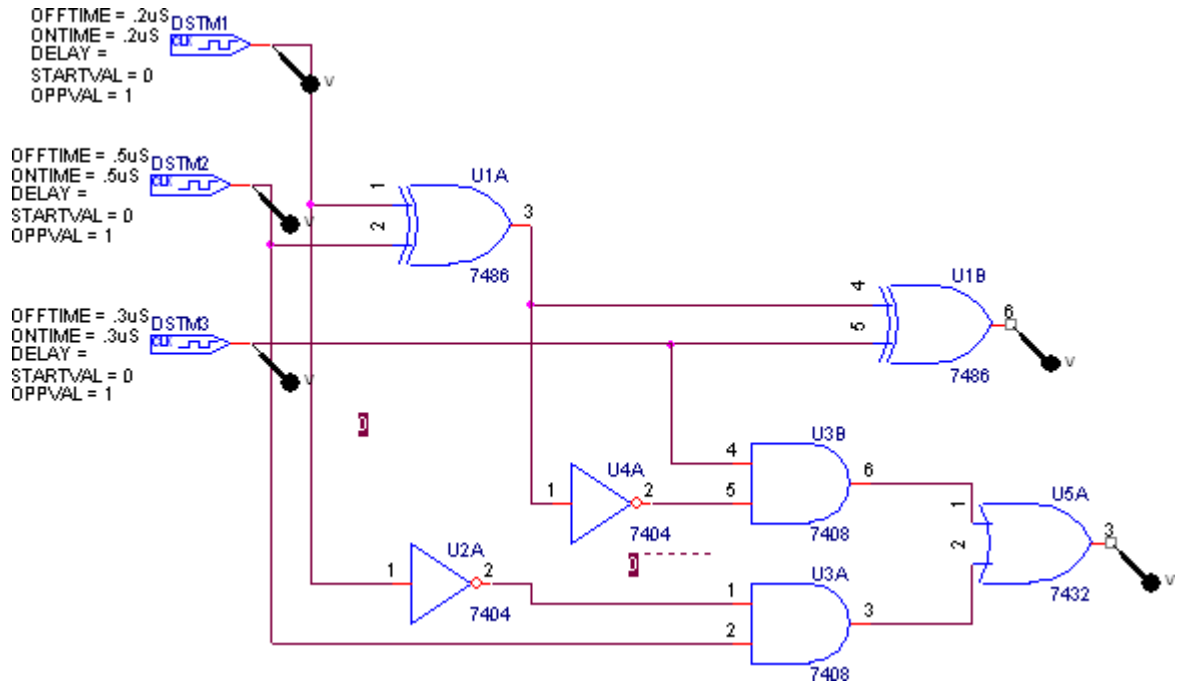
C = A'.B +C(A xnor B)

TRUTH TABLE:

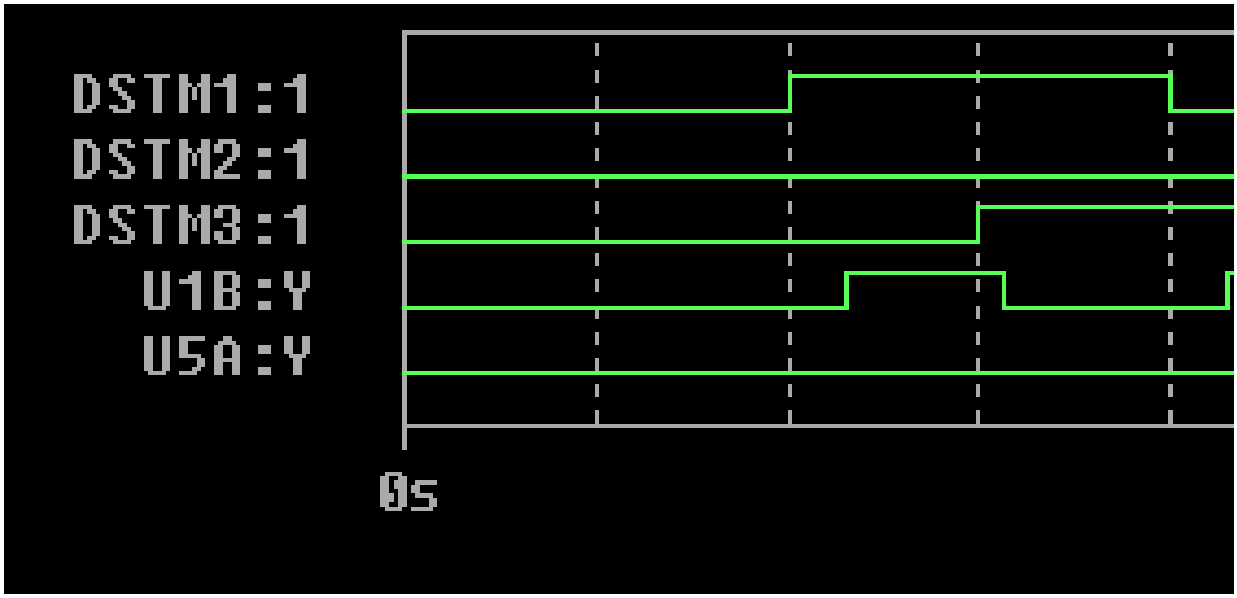
			DIFF	**BOR**

0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0

SCHEMATIC DIAGRAM:



WAVEFORM:



RESULT: The output waveform of full subtractor is verified.

Program-6

Write a program to add two 8 bit numbers and storethe result at memory location 2000

Algorithm-

a. 1.Load the first number from memory location 2050 to accumulator. 2.Move the content of accumulator to register H.

- Load the second number from memory location 2051 to accumulator.
- Then add the content of register H and accumulator using "ADD" instruction and storing result at
- 2000.

2. The carry generated is recovered using "ADC" command and is stored at memory location 2000.

Program –

Memory Address	Mnemonics	Comment
2000	LDA 2050	A<-[2050]
2003	MOV H, A	H<-A
2004	LDA 2051	A<-[2051]
2007	ADD H	A<-A+H
2008	MOV L, A	L←A
2009	MVI A 00	A←00
200B	ADC A	A←A+A+carry
200C	MOV H, A	H←A
200D	SHLD 3050	H→2000, L→3050
2010	HLT	

Explanation-

1. 1.LDA 2050 moves the contents of 2050 memory location to the accumulator.
2. 2.MOV H,A copies contents of Accumulator to register H to A.
3. LDA 2051 moves the contents of 2051 memory location to the accumulator.
4. ADD H adds contents of A (Accumulator) and H register(F9).The result is stored in A itself.For all arithmetic instructions A is by default an operand andA stores the result as well.
5. MOV L,A copies contents of A(34) toL.

   6.MVI A00 moves immediate data (i.e,00)

   to A
6. ADC A adds contents of A(00),contents of register specified (i.eA)and carry (1).As ADC is also an arithmetic operation,A is by default an operand and A stores the result as well.
7. MOV H,A copies contents of A(01) toH.
8. SHLD3050 moves the contents of L register (34) in 3050memory location and contents of H register (01) in 2000 memory location.
9. HLT stops executing the program and halts any further execution.

Program-8

WAP to multiply two 8 bit numbers stored at memory location 2000 and 2001 and stores the result at memorylocation 2000 and 2001

Algorithm-

1. We are taking adding the number 43seven(7) times in this example.
2. As the multiplication of two 8 bit numbers can be maximum of 16bits so we need register pair tostore the result.

Program –

Memory Address	Mnemonics	Comment
2000	LHLD 2050	H←2000, L←2001
2003	XCHG	H↔D, L↔E
2004	MOV C, D	C←D
2005	MVI D 00	D←00
2007	LXI H 0000	H←00, L←00
200A	DAD D	HL←HL+DE
200B	DCR C	C←C-1
200C	JNZ 200A	If Zero Flag=0, goto 200A
200F	SHLD 3050	H→2000, L→2001
2012	HLT	

Explanation-

Register used:A,H,L,C,D,E

a 1.LHLD 2050 loads content of 2000 in H and content of 2001 in L.

b. 2.XCHG exchanges contents of H with D and contents of L with E.

c. 3.MOV C,D copies content of D in C.4.MVI D 00 assigns 00 to D.

2. LXI H 0000 assigns 00 to H and 00 to L.

3. DAD D adds HL and DE and assigns the result to HL.7.DRC C decrements C by 1.

a. 8.JNZ 200A jumps program counter to 2000A if zero flag=0. 9.SHLD stores value of H at memory location 2000 and L at 2001

4. HLT stops executing the program and halts any further execution.

Program-9

Addition of 16 bit number

Algorithm –

1. **Load both the lower and the higher bits of first number at once**
2. **Copy the first number to another register pair**
3. **Load both the lower and the higher bits of second number at once**
4. **Add both the register pairs and store the result in a memory location**

Program –

MEMORY ADDRESS	MNEMONICS	COMMENTS
2000	LHLD 2050	H-L ← 2050
2003	XCHG	D H & E L
2004	LHLD 2052	H-L ← 2052
2007	DAD D	H ← H+D & L ← L+E
2008	SHLD 3050	A → 3050
200B	HLT	Stops execution

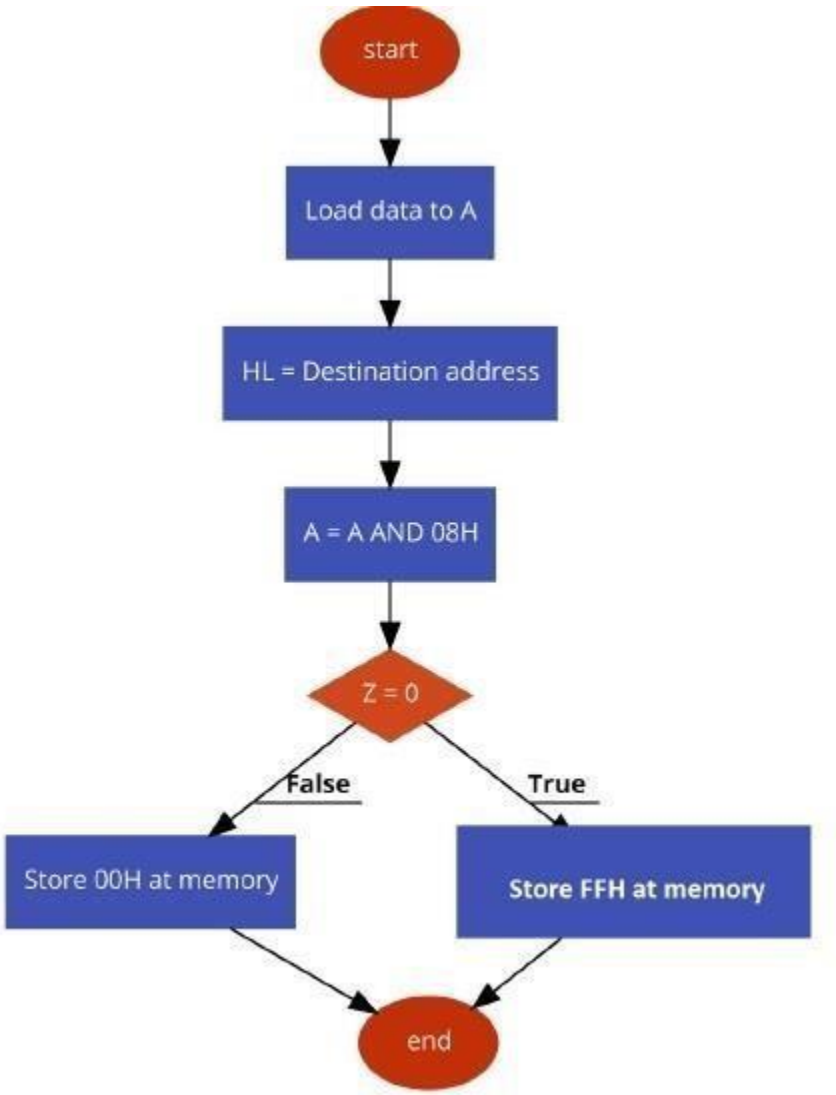
Explanation –

1. **LHLD 2050** loads the value at 2050 in L register and that in 2051 in H register (firstnumber)
2. **XCHG** copies the content of H to D register and L to S register
3. **LHLD 2052** loads the value at 2052 in L register and that in 2053 in H register(second number)
4. **DAD D** adds the value of H with D and L with E and stores the result in H and L
5. **SHLD 3050** stores the result at memory location 3050
6. **HLT** stops execution.

Program-10

WAP which tests if any bit is '0' in a data byte specified at an address 2000. If itis so, 00 would be stored at address 20 Program-601 and if not so then FF should be storedat the same address

Flow Diagram



Program

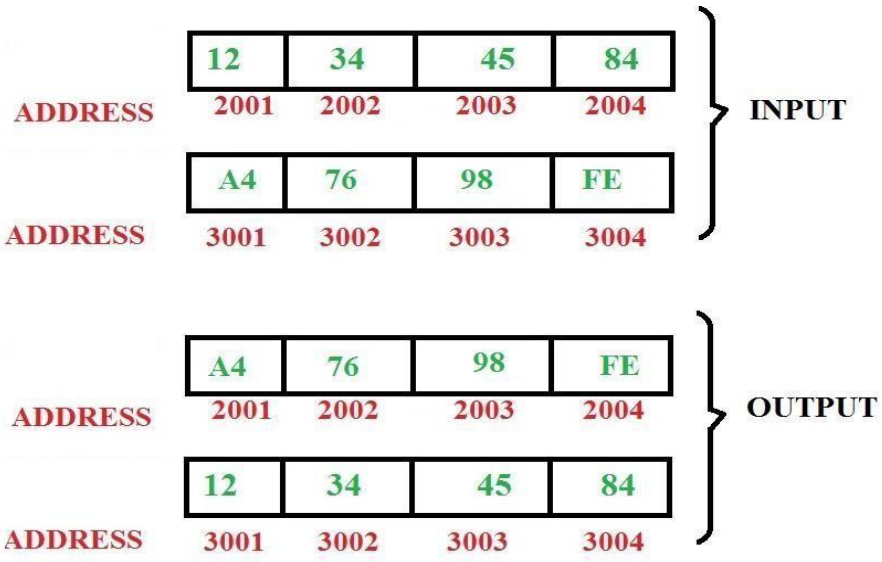
Address	HEX Codes		Mnemonics	Comments

F000	21, 00, 80		LXIH,8000H	Load address to get data

Address	HEX Codes	Labels	Mnemonics	Comments
F003	7E		MOV A, M	Load memory content to Acc
F004	21, 50, 80		LXI H,8050H	Load the destination address
F007	E6, 08		ANI 08H	AND acc with 0000 1000
F009	C2, 11, F0		JNZ NONZ	When Z flag is set, save 00H
F00C	36,00		MVI M, 00H	Save FFH when Z is not set
F00E	C3, 13, F0		JMP END	Jump to stop the program

F011	36, FF	NONZ	MVIM, FFH	Save 00H
F013	76	END	HLT	Terminate the program

Assume that 3 bytes of data are stored at consecutive memory addresses of thedata memory starting at 2000. Write a program which loads register C with (2000), i.e. with data contained at memory address2000, D with (2001), E with (2002) and A with (2001).



Algorithm –

1. Take a count equal to 4
2. Store the starting address of both blocks in 2 different register pairs
3. Now exchange the contents at the addresses in both register pairs
4. Increment the values of both register pairs
5. Decrements count by 1
6. If count is not equal to 0 repeat steps 3 to 5

MEMORY ADDRESS	MNEMONICS	COMMENTS
2500	LXI D 2001	D <= 20, E <= 01
2503	LXI H 3001	H <= 20, L <= 01
2506	MVI C 04	C <= 04
2508	MOV B, M	B <= M[H-L]

MEMORY ADDRESS	MNEMONICS	COMMENTS
2509	LDAX D	A <= M[D-E]
250A	MOV M, A	M[H-L] <= A
250B	MOV A, B	A <= B
250C	STAX D	M[D-E] <= A
250D	INX H	[H-L] <= [H-L] + 1
250E	INX D	[D-E] <= [D-E] + 1
250F	DCR C	C <= C – 1
2510	JNZ 2508	JUMP TO 2508 IF C NOT EQUAL TO 0
2513	HLT	STOP THE PROGRAM

Explanation –

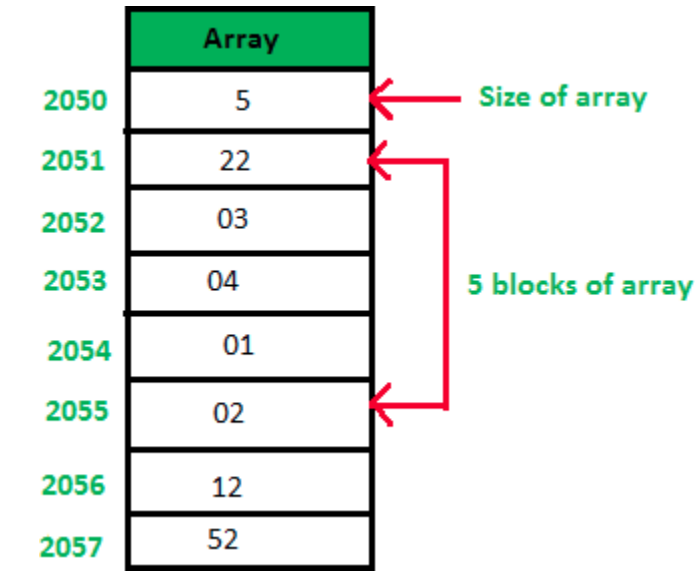
1. **LXI D 2001 –** Loads register pair, that is in this case, D=20 and E=01 LXI H 3001 – H=30 and L=01
2. **MVI C 04 –** Assigns immediate data, eg.- here C=04
   MVI A 45 – assigns A(accumulator) with 45, A=45
3. **MOV B, M –** Here M is the data in H – L register pair and it serves as an address. Copies content at address stored in M to register B
4. **LDAX D –** Here Accumulator is loaded with the data stored at address formed by register pair D – E
5. **MOV M, A –** Here A's content is copied to address which is stored in M. MOV A, B – Copies content of register B to A
6. **STAX D –** Stores the content of A (accumulator) in the address formed by register pair D – E.
7. **INX H –** Increment the content of register pair H – L
8. **INX H –** Increment the content of register pair D – E
9. **DCR C –** Decrements the content of register C
10. **JNZ 2508 –** If value of register C is not equal to 0 then jump to address 2508
11. **HLT –** Stop execution of program

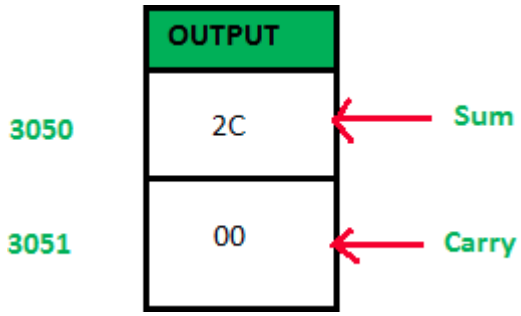
Sixteen bytes of data are specified at consecutive data-memory locations startingat 2000. Write a program which increments the value of all sixteen bytes by 01

.

Program-12

WAP to add t 10 bytes stored at memory location starting from 3000. Store theresult at memory location 300A





Algorithm –

1. Load the base address of the array in HL register pair.
2. Use the size of the array as a counter.
3. Initialise accumulator to 00.
4. Add content of accumulator with the content stored at memory location given in HL pair.
5. Decrease counter on each addition.

Program –

Address	Mnemonics	Comments
2000	LDA 2050	A <- [2050]

Explanation –

Address	Mnemonics	Comments
2003	MOV B, A	B <- A
2004	LXI H, 2051	H <- 20 and L <- 51
2007	MVI A, 00	A <- 00
2009	MVI C, 00	C <- 00
200B	ADD M	A <- A+M
200C	INR L	M <- M+1
200D	JNC 2011	
2010	INR C	C <- C+1
2011	DCR B	B <- B-1
2012	JNZ 200B	
2015	STA 3050	3050 <- A
2018	MOV A, C	A <- C
2019	STA 3051	3051 <- A
201C	HLT	Terminates the program

1. **LDA 2050:** load accumulator with content of location 2050
2. **MOV B, A:** copy contents of accumulator to register B
3. **LXI H, 2051:** store 20 to H register and 51 to L register
4. **MVI A, 00:** store 00 to accumulator
5. **MVI C, 00:** store 00 to register C
6. **ADD M:** add accumulator with the contents of memory location given in HL register pair

7. **INR L:** increase address by 1
8. **JNC 2011:** if not carry, jump to location 2011 otherwise to the location given in PC
9. **INR C:** increase content of register C by 1
10. **DCR B:** decrease content of register B by 1
11. **JNZ 200B:** if not zero, jump to location 200B otherwise to the location given in PC
12. **STA 3050:** store contents of accumulator to memory location 3050
13. **MOV A, C:** copy contents of register C to accumulator
14. **STA 3051:** store contents of accumulator to memory location 3051
15. **HLT:** terminates the program