**EXPERIMENT 1: Store 8-bit data in memory**

|  |
| --- |
| **Statement: Store the data byte 32H into memory location 4000H.**  Program 1:  MVI A, 32H : Store 32H in the accumulator  STA 4000H : Copy accumulator contents at address 4000H  HLT : Terminate program execution  Program 2:  LXI H 4000H : Load HL with 4000H  MVI M, 32H : Store 32H in memory location pointed by HL register pair (4000H)  HLT : Terminate program execution |

Note: The result of both programs will be the same. In program 1 direct addressing instruction is used, whereas in program 2 indirect addressing instructions is used.

**EXPERIMENT 2. Exchange the contents of memory locations**

|  |
| --- |
| **Statement: Exchange the contents of memory locations 2000H and 4000H.**  Program 1:  LDA 2000H : Get the contents of memory location 2000H into accumulator  MOV B, A : Save the contents into B register  LDA 4000H : Get the contents of memory location 4000Hinto accumulator  STA 2000H : Store the contents of accumulator at address 2000H  MOV A, B : Get the saved contents back into A register  STA 4000H : Store the contents of accumulator at address 4000H |
|  |

Program 2:

LXI H 2000H : Initialize HL register pair as a pointer to memory location 2000H.

LXI D 4000H : Initialize DE register pair as a pointer to memory location 4000H.

MOV B, M : Get the contents of memory location 2000H into B register.

LDAX D : Get the contents of memory location 4000H into A register.

MOV M, A : Store the contents of A register into memory location 2000H.

MOV A, B : Copy the contents of B register into accumulator.

STAX D : Store the contents of A register into memory location 4000H.

HLT : Terminate program execution.

Note: In Program 1, direct addressing instructions are used, whereas in Program 2, indirect addressing instructions are used.

**EXPERIMENT 3: Add two 8-bit numbers**

**Statement: Add the contents of memory locations 4000H and 4001H and place the result in memory location 4002H.**

Source program

LXI H 4000H : HL points 4000H

MOV A, M : Get first operand

INX H : HL points 4001H

ADD M : Add second operand

INX H : HL points 4002H

MOV M, A : Store result at 4002H

HLT : Terminate program execution

**EXPERIMENT 4: Add two 16-bit numbers**

**Statement: Add the 16-bit number in memory locations 4000H and 4001H to the 16-bit number in memory locations 4002H and 4003H. Store the result in memory locations 4004H and 4005H.**

Source Program 1:

LHLD 4000H : Get first I6-bit number in HL

XCHG : Save first I6-bit number in DE

LHLD 4002H : Get second I6-bit number in HL

MOV A, E : Get lower byte of the first number

ADD L : Add lower byte of the second number

MOV L, A : Store result in L register

MOV A, D : Get higher byte of the first number

ADD H : Add upper byte

MOV H, A : Store result in H register

SHLD 4004H : Store I6-bit result in memory locations 4004H and 4005H.

HLT :Terminate program execution

**EXPERIMENT 5 : Subtract two 8-bit numbers**

**Statement: Subtract the contents of memory location 4001H from the memory location 2000H and place the result in memory location 4002H.**

Source program:

LXI H, 4000H : HL points 4000H

MOV A, M : Get first operand

INX H : HL points 4001H

SUB M : Subtract second operand

INX H : HL points 4002H

MOV M, A : Store result at 4002H.

HLT : Terminate program execution

**EXPERIMENT 6: Add contents of two memory locations**

**Statement: Add the contents of memory locations 40000H and 4001H and place the result in the memory locations 4002Hand 4003H.**

LXI H, 4000H : HL Points 4000H

MOV A, M : Get first operand

INX H : HL Points 4001H

ADD M : Add second operand

INX H : HL Points 4002H

MOV M, A : Store the lower byte of result at 4002H

MVIA, 00 : Initialize higher byte result with 00H

ADC A : Add carry in the high byte result

INX H : HL Points 4003H

MOV M, A : Store the higher byte of result at 4003H

HLT : Terminate program execution

**EXPERIMENT 7 Subtract two 16-bit numbers**

**Statement: Subtract the 16-bit number in memory locations 4002H and 4003H from the 16-bit number in memory locations 4000H and 4001H. Store the result in memory locations 4004H and 4005H.**

Source program:

LHLD 4000H : Get first 16-bit number in HL

XCHG : Save first 16-bit number in DE

LHLD 4002H : Get second 16-bit number in HL

MOV A, E : Get lower byte of the first number

SUB L : Subtract lower byte of the second number

MOV L, A : Store the result in L register

MOV A, D : Get higher byte of the first number

SU B H : Subtract higher byte of second number with borrow

MOV H, A : Store l6-bit result in memory locations 4004H and 4005H.

SHLD 4004H : Store l6-bit result in memory locations 4004H and 4005H.

HLT : Terminate program execution.

**EXPERIMENT 8 : Calculate the sum of series of numbers**

**Statement: Calculate the sum of series of numbers. The length of the series is in memory location 4200H and the series begins from memory location 4201H.**

a. Consider the sum to be 8 bit number. So, ignore carries. Store the sum at memory location 4300H.

Source Program:

LDA 4200H

MOV C,A : Initialize counter

SUB A : sum=0

LXI H, 4201H : Initialize pointer

BACK: ADD M : SUM=SUM+data

INX H : increment pointer

DCR C : decrement counter

JNZ BACK : if counter not 0, repeat

STA 4300H : store sum

HLT : terminate program

**EXPERIMENT 9: Multiply two 8-bit numbers**

**Statement: Multiply two 8-bit numbers stored in memory locations 2200H and 2201H by repetitive addition and store the result in memory locations 2300H and 2301H.**

|  |
| --- |
| Source program : |
|  |

LDA 2200H

MOV E, A

MVI D, 00 : Get the first number in DE register pair

LDA 2201H

MOV C,A: Initialize counter

LXI H, 0000H: result=0

BACK: DAD D : Result= result + first number (DOUBLE ADD)

D CR C: Decrement count

JNZ BACK: If count 0, repeat

SHLD 2300H: Store result

HLT: Terminate

**EXPERIMENT 10: Divide a 16 bit number by a 8-bit number**

***Statement:* Divide 16 bit number stored in memory locations 2200H and 2201H by the 8 bit number stored at memory location 2202H. Store the quotient in memory locations 2300H and 2301H and remainder in memory locations 2302H and 2303H.**

Sample problem 1:

*(2200H) = 60H*

*(2201H) = A0H*

*(2202H) = l2H*

*Result = A060H/12H = 8E8H Quotient and 10H remainder*

*(2300H) = E8H*

*(2301H) = 08H*

*(2302H= 10H*

|  |  |
| --- | --- |
| *(2303H)00H Source program* |  |

LHLD 2200H : Get the dividend

LDA 2202H : Get the divisor

MOV C,A

LXI D, 0000H : Quotient-0

BACK : MOV A,L

SUB C :Subtract divisor

MOV L,A : Save partial result

JNC SKIP : if CY 1 jump

DCR H : Subtract borrow of previous subtraction

SKIP: INX D : Increment quotient

MOV A,H

CPI 00 : Check if dividend < divisor

JNZ BACK : if no repeat

MOV A,L

CMP C

JNC BACK

SHLD 2302H : Store the remainder

XCHG

SHLD 2300H : Store the quotient

HLT : Terminate the prog

**EXPERIMENT 11: Find the largest of given numbers**

***Statement:* Find the largest number in a block of data. The length of the block is in memory location 2200H and the block itself starts from memory location 2201H.**

**Store the maximum number in memory location 2300H. Assume that the numbers in the block are all 8 bit unsigned binary numbers.**

Source Program:

LDA 2200H

MOV C,A : Initialize counter

XRA A : Maximum = Minimum possible value = 0

LXI H, 2201H : Initialize pointer

BACK: CMP M : Is number > maximum

JNC SKIP : Yes, replace maximum

MOV A, M

SKIP: INX H

DCR C

JNZ BACK

STA 2300H : Store maximum number

HLT : Terminate program execution

**EXPERMIENT 12: Arrange in ascending order**

***Statement:* Write a program to sort given 10 numbers from memory location 2200H in the ascending order.**

Source Program:

MVI B, 09 : Initialize counter

START: LXI H, 2200H ; Initialize memory pointer

MVI C, 09 : Initialize counter 2

BACK: MOV A, M : Get the number

INX H : Increment memory pointer

CMP M : Compare number with next number

JC SKIP : If less, don’t interchange

JZ SKIP : If equal, don’t interchange

MOV D, M

MOV M, A

DCX H

MOV M, D

INX H : Interchange two numbers

SKIP: DCR C : Decrement counter 2

JNZ BACK : If not zero, repeat

DCR B : Decrement counter 1

JNZ START

HLT : Terminate program execution

**EXPERIMENT 13: Storing series of data byte to another location**

**Statement: Five data bytes are stored in memory locations at 1050H to 1054 H. Transfer entire block of data to new memory locations starting at 3070H.**

Source Program

START: LXI H, 1050H :Setup HL as pointer for source memory

LXI D, 3070H : Setup DE as pointer for destination

MVI B, 05H : Setup B to count bytes

NEXT: MOV, A M : Get data byte from source memory

STAX D : Store data byte at destination

INX H : Point HL to next source location

INX D : Point DE to next destination

DCR B : One transfer is complete

JNZ NEXT : If counter is not 0, go back to transfer next byte

HLT

**EXPERIMENT 14: Addition with carry**

**Statement: Six bytes of data are stored in memory location starting at 2050H. Add all the data bytes. Use register B to save any carries generated, while adding the data bytes. Display the entire sum at two output ports, or store the sum at two consecutive memory locations, 5070H and 5071H.**

Source Program:

XRA A :Clear (A) to save sum

MOV B,A :Clear (B) to save carry

MVI C, 06H :Setup register C as a counter

LXI H, 2050H : Setup HL as memory pointer

NXTBYT ADD M :Add byte from memory

JNC NXTMEM :If no carry, don’t increment carry reg

INR B :If carry, save carry bit

NXTMEM INX H :Point to next memory location

DCR C :One addition is completed; decrement counter

JNZ NXTBYT :If all bytes are not yet added, go back to get next byte

LXI H, 5070H :point to the memory location to store answer

MOV M, A :Store low-order byte at 5070H

INX H :Point to location 5071H

MOV M,B : Store carry bits

HLT :Terminate the program

**EXPERIMENT 15: Checking sign with rotate instructions**

**Statement: A set of ten current readings is stored in memory locations starting at 1060H. The readings are expected to be positive (<12710). Write a program to:**

1. Check each reading to determine whether it is positive or negative
2. Reject all negative readings
3. Add all positive readings
4. output FFH to Port 1 at any time when the sum exceeds eight bits to indicate overload; otherwise, display the sum. If no output port is available in the system, go to step 5
5. store FFH in the memory location 2070H when the sum exceeds eight bits; otherwise store the sum.

Source program

MVI B, 00H : Clear B to save sum

MVI C, 0AH : Setup C register as a counter

LXI H, 1060H : Setup HL as memory pointer

NEXT MOV A,M : Get byte

RAL : Shift D7 into CY

JC REJECT : If D7=1, reject byte & go to increment pointer

RAR : If byte is positive, restore it

ADD B : Add previous sum to (A)

JC OVRLOD : If sum > FFH, it is overload

MOV B,A :Save sum

REJECT INX H : Point to next reading

DCR C : One reading is checked; decrement counter

JNZ NEXT : If all readings are not checked, go back to transfer next byte MOV A, B

OUT PORT 1 : Display sum

HLT

OVRLOD: MVI A, FFH : It is overload

OUT PORT 1 : Display overload signal at PORT 1

HLT

**EXPERIMENT 16: Use of Compare Instruction to indicate end of Data String**

**Statement: A set of current readings is stored in memory locations starting at 1050H. The end of the data string is indicated by the data byte 00H. Add the set of readings. The answer may be larger than FFH. Display the entire sum at PORT 1 and PORT 2 .**

DATA : 32, 52, F2, A5, 00

Source Program:

START: LXI H, 1050H : Setup HL as memory pointer

MVI C, OOH : Clear C to save sum

MOV B,C : Clear B to save carry

NXTBYT MOV A,M : Transfer current reading to A

CPI 00H : Is this the last reading?

JZ DSPLAY : If yes, go to display section

ADD C : Add previous sum to accumulator

JNC SAVE : Skip CY register if there is no carry

INR B : Update carry register

SAVE: MOV C,A : Save sum

INX H : Point to next reading

JMP NXTBYT : Go back to get next reading

DSPLAY: MOV A,C

OUT PORT 1 : Display low order byte of sum at PORT 1

MOV A, B : Transfer carry bits to accumulator

OUT PORT 2 : Display high order byte of sum at PORT 2

HLT : End of program

**EXPERIMENT 17: Sorting**

**Statement: A set of three readings is stored in memory starting at 1050 H. Sort the readings in ascending order.**

Data: 87, 56, 42

Source program

START: LXI H, 1050H : Setup HL as a memory pointer for bytes

MVI D, 00 : Clear register D to setup a flag

MVI C, 02 : Set register C for comparison count

CHECK: MOV A,M : Get data byte

INX H : Point to next byte

CMP M : Compare bytes

JC NXTBYTE : If (A) < second byte, do not exchange

MOV B,M : Get second byte for exchange

MOV M,A : Store first byte in second location

DCX H : Point to first location

MOV M,B : Store second byte in first location

INX H : Get ready for next comparison

MVI D, 01 : Load 1 in D as a reminder for exchange

NXTBYTE: DCR C : Decrement comparison count

JNZ CHECK : If comparison count is not 0, go back

MOV A,D : Get flag bit in A

RRC : Place flag bit D0 in Carry

JC START : If flag is 1, exchange occurred; start the next pass

HLT : End of sorting

Execution of Sort Program

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Memory loc. | Initial Bytes | First Pass | Second Pass | Third Pass |
| 1050 | 87 | 56 | 42 | 42 |
| 1051 | 56 | 42 | 56 | 56 |
| 1052 | 42 | 87 | 87 | 87 |
| Reg. D | 0 | 1 | 1 | 0 |