Digital Electronics & Microprocessors Laboratory (EC2P006)

EXPERIMENT-7

8085 µP Program for Addition and Multiplication Operations

Shorya Sharma
19EE01017

Objectives:

- Write an assembly language program that reads 5-bytes of data from five consecutive memory locations, adds them along with the carry, and stores the sum and carry in next two consecutive memory locations.
- Write an assembly language program that reads two numbers of 16-bit size each from two consecutive memory locations, multiplies them, and stores the result in the next consecutive memory locations.

Specifications:

My roll No is **19EE01017;** X=1 and Y=7

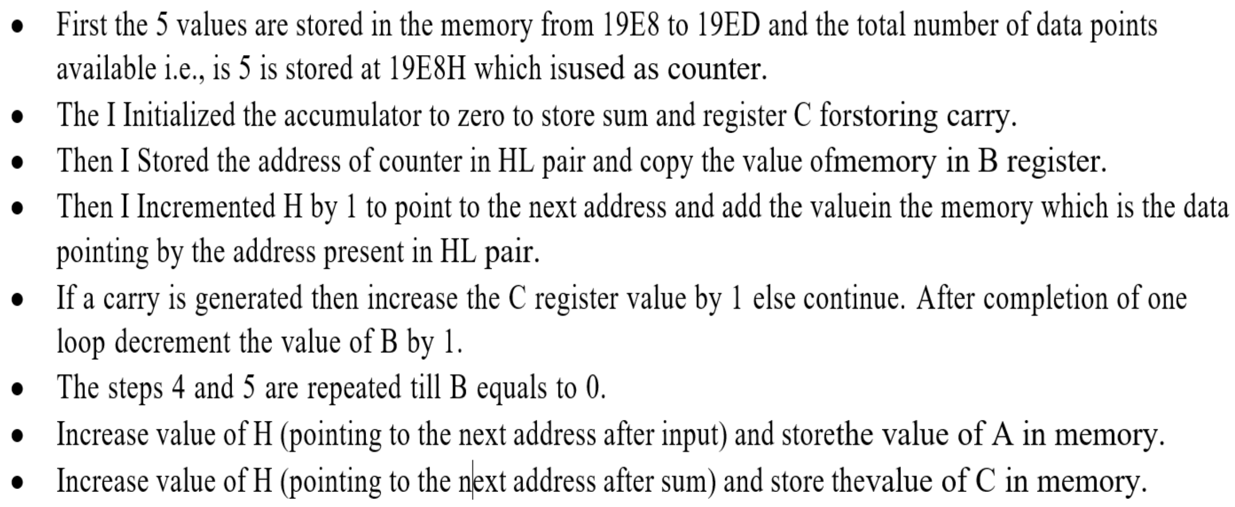
Starting address of memory = (XY00 + 2^{Y}) = (1700+128) = 1828H

Part 1: Assembly program for Addition

Aim: -

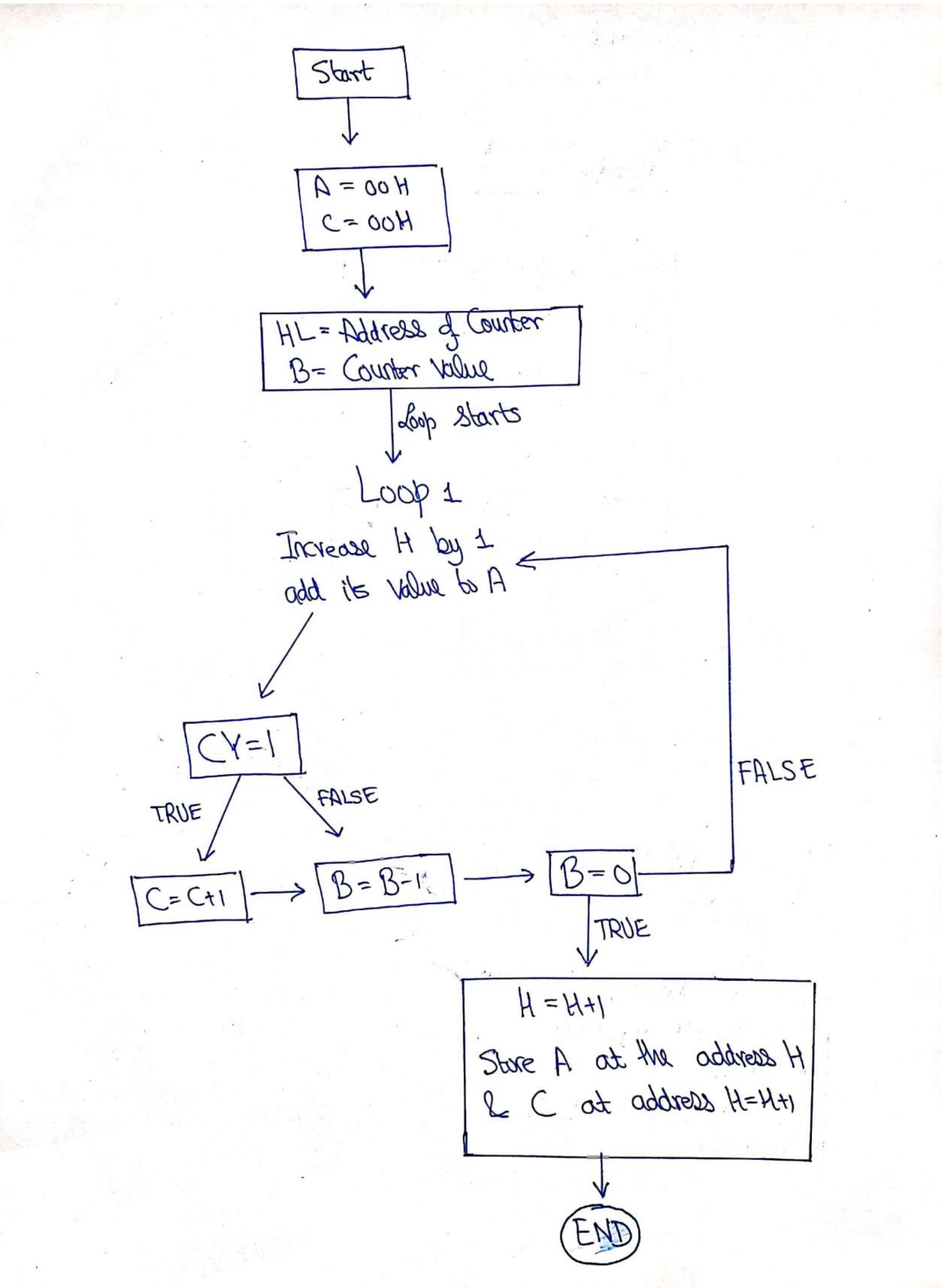
Write an assembly language program that reads 5-bytes of data from five consecutive memory locations, adds them along with the carry, and stores the sum and carry in next two consecutive memory locations.

Algorithm: -



- First the 5 values are stored in the memory from 19E8 to 19ED and the total number of data points available i.e., is 5 is stored at 19E8H which isused as counter.
- The I Initialized the accumulator to zero to store sum and register C forstoring carry.
- Then I Stored the address of counter in HL pair and copy the value ofmemory in B register.
- Then I Incremented H by 1 to point to the next address and add the valuein the memory which is the data pointing by the address present in HL pair.
- If a carry is generated then increase the C register value by 1 else continue. After completion of one loop decrement the value of B by 1.
- The steps 4 and 5 are repeated till B equals to 0.
- Increase value of H (pointing to the next address after input) and storethe value of A in memory.
- Increase value of H (pointing to the next address after sum) and store thevalue of C in memory.

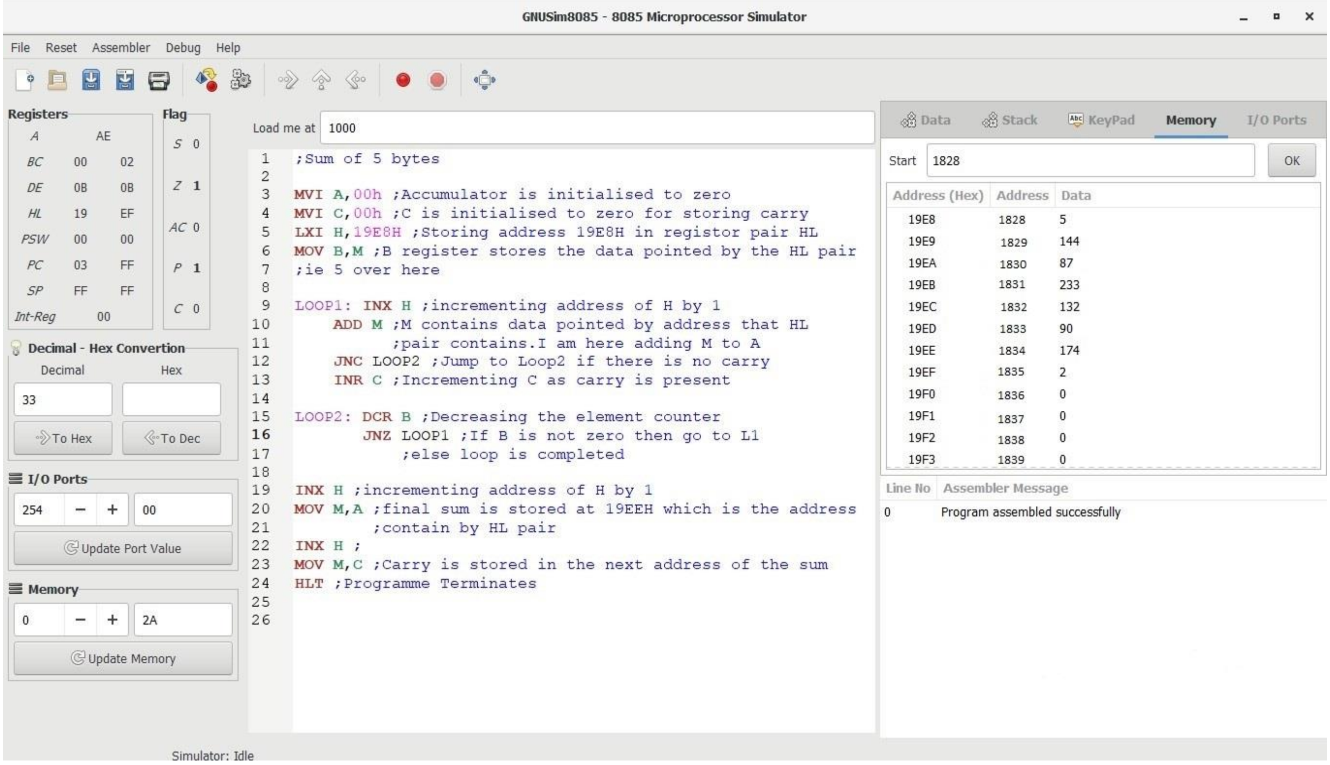
Flowchart: -


Start
A = 00 H
C = 00H
HL = Address of Counter
B = Counter Value
Loop starts
Loop 1
Increase H by 1
add its value to A
CY=1
TRUE
FALSE
C=C+1
B=B-1
B=0
FALSE
TRUE
H = H+1
Store A at the address H
& C at address H=H+1
END

PROGRAM:

Memory	Opcode	Label	Mnemonics	OPERANDS	Comments
07D0	3E		MVI	A,00H	[A]<-00H
07D2	0E		MVI	C,00H	[C]<-00H
07D4	21		LXI	H,19E8H	[H-L]<-[19E8H]
07D7	46		MOV	B,M	[B] <- [M]
07D8	23	LOOP1	INX	H	[H-L] <- [H-L]+1
07D9	8E		ADD	M	A=A+M
07DA	D2		JNC	L2	Jump if Cy=0
07DD	0C		INR	C	[C] = [C]+1
07DE	05	LOOP2	DCR	B	[B] = [B]-1
07DF	C2		JNZ	L1	Jump if not zero
07E2	23		INX	H	[H-L] <- [H-L]+1
07E3	77		MOV	M,A	[M] <- [A]
07E4	23		INX	H	[H-L] <- [H-L]+1
07E5	71		MOV	M,C	[M] <- [C]
07E6	76		HLT		TERMINATE

Code:



Load me at 1000

```
;Sum of 5 bytes

MVI A,00h ;Accumulator is initialised to zero
MVI C,00h ;C is initialised to zero for storing carry
LXI H,19E8H ;Storing address 19E8H in registor pair HL
MOV B,M ;B register stores the data pointed by the HL pair
;ie 5 over here

LOOP1: INX H ;incrementing address of H by 1
    ADD M ;M contains data pointed by address that HL
          ;pair contains.I am here adding M to A
    JNC LOOP2 ;Jump to Loop2 if there is no carry
    INR C ;Incrementing C as carry is present

LOOP2: DCR B ;Decreasing the element counter
       JNZ LOOP1 ;If B is not zero then go to L1
           ;else loop is completed

INX H ;incrementing address of H by 1
MOV M,A ;final sum is stored at 19EEH which is the address
        ;contain by HL pair
INX H ;
MOV M,C ;Carry is stored in the next address of the sum
HLT ;Programme Terminates
```

Registers

A	AE	
BC	00	02
DE	0B	0B
HL	19	EF
PSW	00	00
PC	03	FF
SP	FF	FF
Int-Reg	00	

Flag

S	0
Z	1
AC	0
P	1
C	0

Decimal - Hex Convertion

Decimal: 33 Hex:

I/O Ports: 254 00

Memory: 0 2A

Start 1828

Address (Hex)	Address	Data
19E8	1828	5
19E9	1829	144
19EA	1830	87
19EB	1831	233
19EC	1832	132
19ED	1833	90
19EE	1834	174
19EF	1835	2
19F0	1836	0
19F1	1837	0
19F2	1838	0
19F3	1839	0

Line No	Assembler Message
0	Program assembled successfully

OBSERVATION:

Input values are [144,87,233,132,90]

Sum of input values = 685

My Output Sum = 174 and carry = 2

Here as we know that data in memory is a 8bit value. Sum and carry are both 8bits. To evaluate the value as 16bit :

Sum in HEX = (174)d = AEH

Carry in Hex = (2)d = 2H

Total value is 2AEH and if this value is converted into decimal:

2*(16^2) + 10*(16^1) + 14*(16^0) = 685

Hence both the output sum and input sum are equal

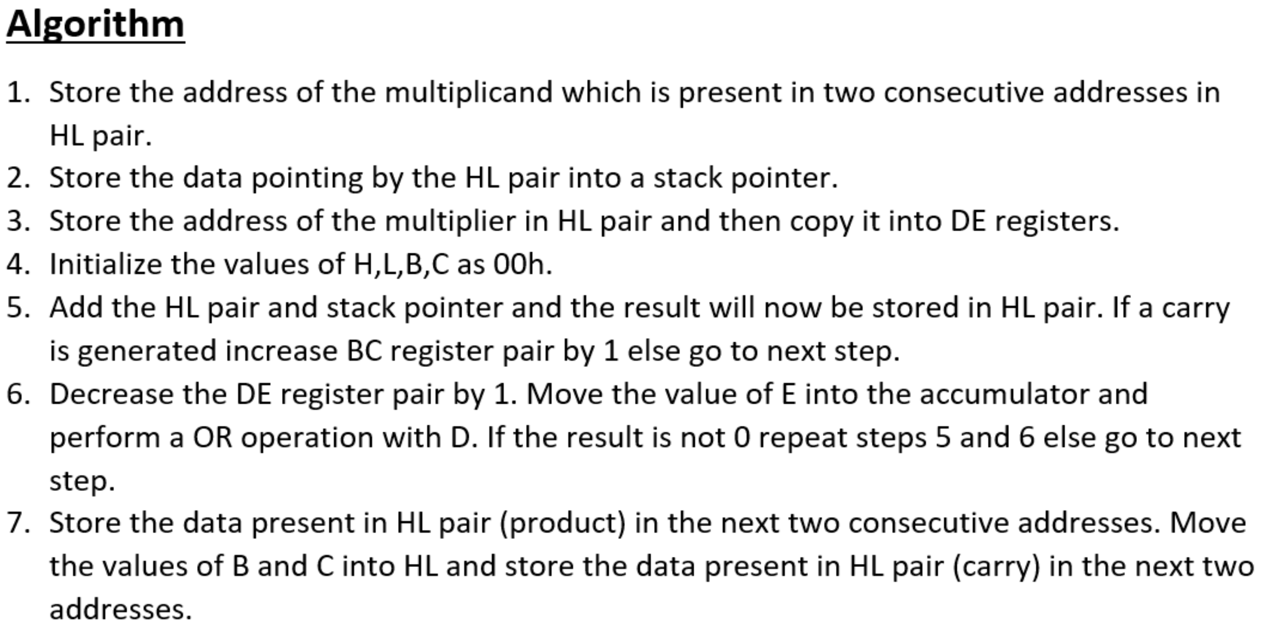
Conclusion

Here we have successfully carried out the addition of five 8-bit numbers which are fetched to microprocessor from memory and whose outputs are verified bytaking two test cases and can be extended for more.

Part 2: 16 Bit Multiplication

Aim:

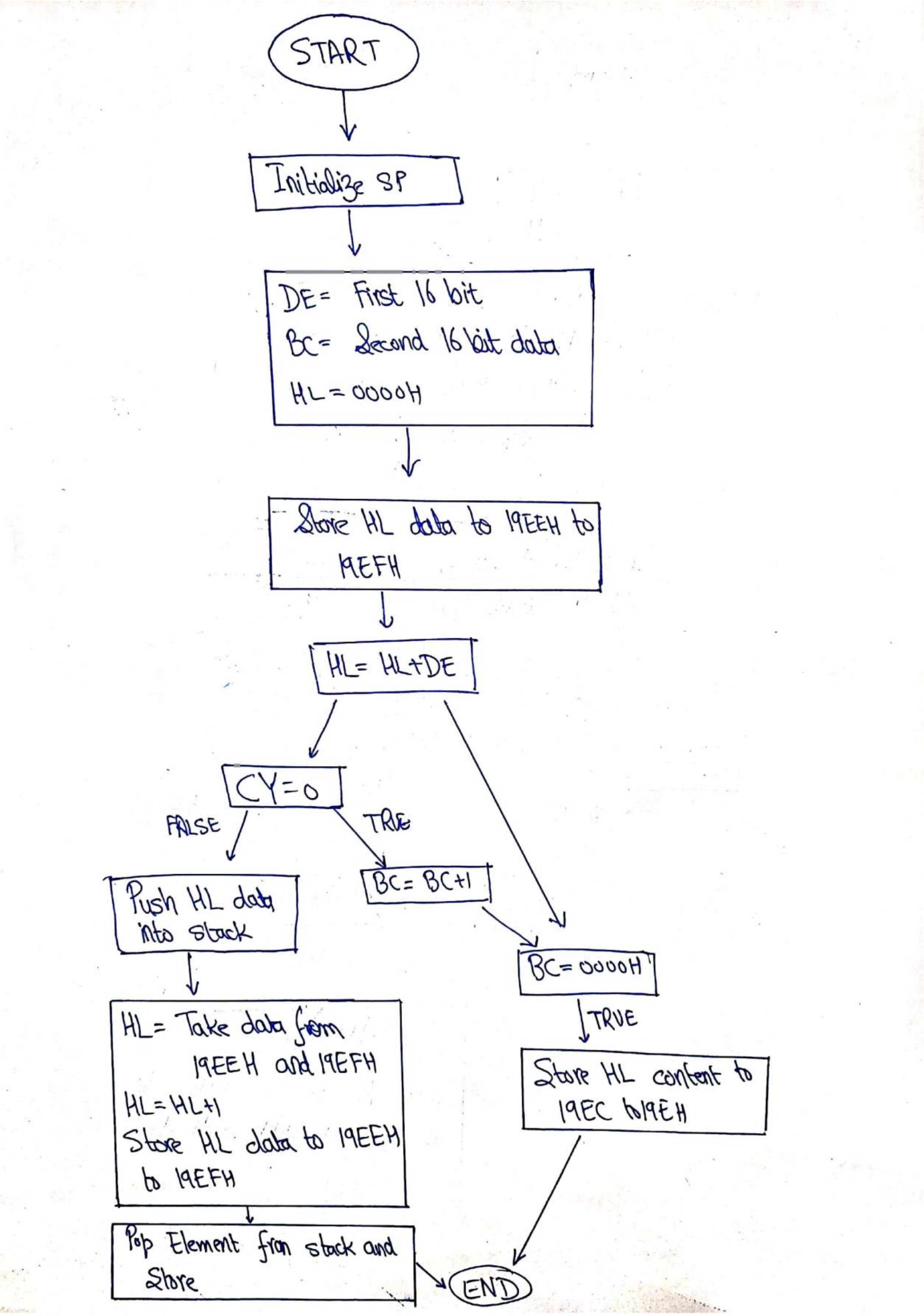
Write an assembly language program that reads two numbers of 16-bit size each from two consecutive memory locations, multiplies them, and stores the result in the next consecutive memory locations.



Algorithm

1. Store the address of the multiplicand which is present in two consecutive addresses in HL pair.
2. Store the data pointing by the HL pair into a stack pointer.
3. Store the address of the multiplier in HL pair and then copy it into DE registers.
4. Initialize the values of H,L,B,C as 00h.
5. Add the HL pair and stack pointer and the result will now be stored in HL pair. If a carry is generated increase BC register pair by 1 else go to next step.
6. Decrease the DE register pair by 1. Move the value of E into the accumulator and perform a OR operation with D. If the result is not 0 repeat steps 5 and 6 else go to next step.
7. Store the data present in HL pair (product) in the next two consecutive addresses. Move the values of B and C into HL and store the data present in HL pair (carry) in the next two addresses.

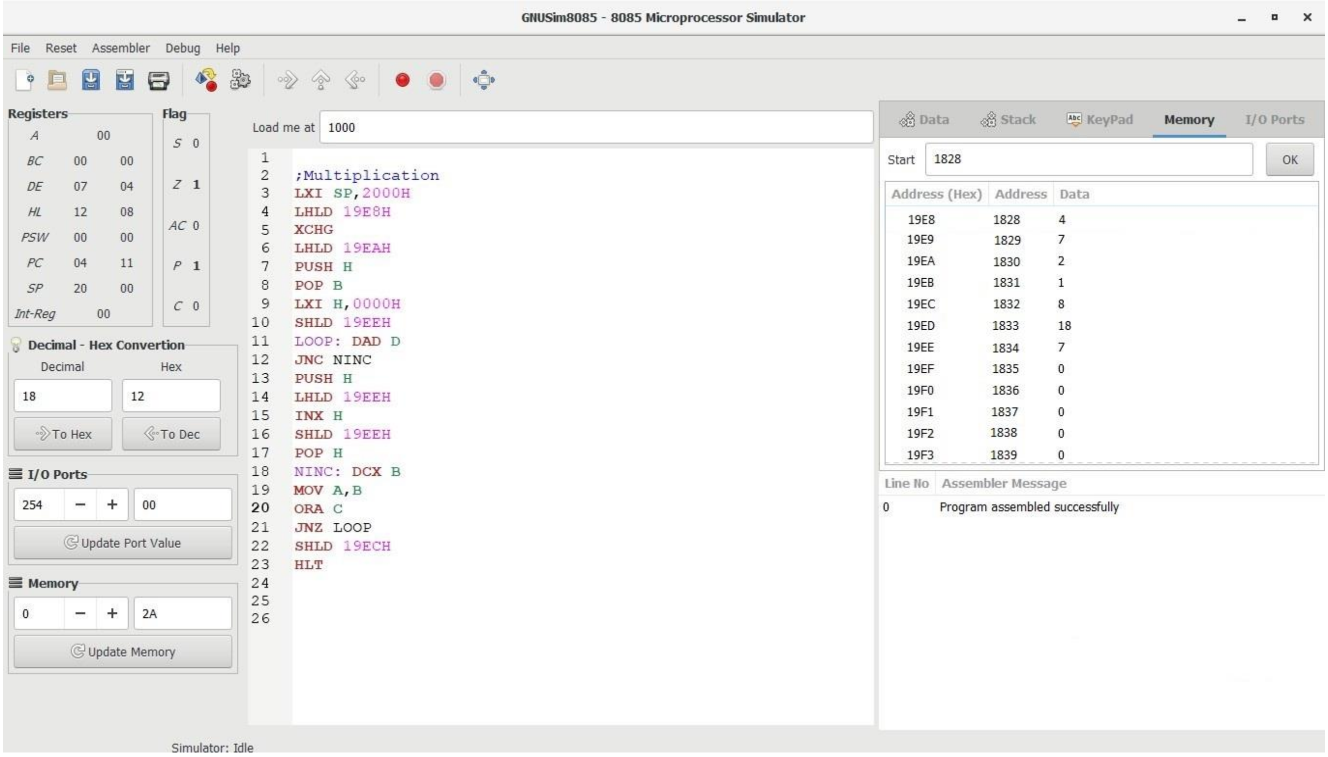
Flowchart:


START
Initialize SP
DE= First 16 bit
BC= Second 16 bit data
HL=0000H
Store HL data to 19EEH to 19EFH
HL= HL+DE
CY=0
FALSE
TRUE
Push HL data into stack
BC= BC+1
BC=0000H
TRUE
HL= Take data from 19EEH and 19EFH
HL=HL+1
Store HL data to 19EEH to 19EFH
Store HL content to 19EC to19EH
Pop Element from stack and Store
END

PROGRAM:

MEMORY	OPCODE	LABELS	MNEMONICS	COMMENTS
1000	31		LXI SP,2000H	Initialize Stack pointer
1003	2A		LHLD 19E8	Load 16-bit data from 19E8H - 19E9H
1006	EB		XCHG	Exchange the data from HL and DE
1007	2A		LHLD 19EA	Load second 16-bit number
100A	E5		PUSH H	Push HL pair into stack
100B	C1		POP B	Load BC with HL pair content from stack
100C	21		LXI H,0000H	Clear HL pai
100F	22		SHLD 19EEH	Store 0000H as LS 2-bytes of the result
1012	19	LOOP	DAD D	Add first numberto HL pair
1013	D2		JNC NINC	if CY = 0, jump to NINC
1016	E5		PUSH H	Push HL into Stack
1017	2A		LHLD 19EEH	Load HL pair from LS 2-bytes of the result
101A	23		INX H	Increase HL pair
101B	22		SHLD 19EEH	Store HL pair as LS 2-bytes of the result
101E	E1		POP H	Pop stack content to HL pair
101F	0B	NINC	DCX B	Decrease BC register pair
1020	78		MOV A,B	*Load B to A*
1021	B1		ORA C	*OR C with A*
1022	C2		JNZ LOOP	*When Z = 0, jump to LOOP*
1025	22		SHLD 19ECH	*Store HL pair to 19ECH*
1028	76		HLD	*Terminate the program*

Code:



Registers

A	00	
BC	00	00
DE	07	04
HL	12	08
PSW	00	00
PC	04	11
SP	20	00
Int-Reg	00	

Flag

S	0
Z	1
AC	0
P	1
C	0

Decimal - Hex Convertion

Decimal	Hex
18	12

I/O Ports: 254 00

Memory: 0 2A

Load me at: 1000

```

;Multiplication
LXI SP,2000H
LHLD 19E8H
XCHG
LHLD 19EAH
PUSH H
POP B
LXI H,0000H
SHLD 19EEH
LOOP: DAD D
JNC NINC
PUSH H
LHLD 19EEH
INX H
SHLD 19EEH
POP H
NINC: DCX B
MOV A,B
ORA C
JNZ LOOP
SHLD 19ECH
HLT
```

Start: 1828

Address (Hex)	Address	Data
19E8	1828	4
19E9	1829	7
19EA	1830	2
19EB	1831	1
19EC	1832	8
19ED	1833	18
19EE	1834	7
19EF	1835	0
19F0	1836	0
19F1	1837	0
19F2	1838	0
19F3	1839	0

Line No	Assembler Message
0	Program assembled successfully

Simulator: Idle

CONCLUSION

Here I used registors B, C, D, E, H, L and the accumulator are used as general proposed registor. I used to stack pointer and push and pop operation in the algorithm. The algorithm is simple that if there are two numbers X and

Y the multiplying X and Y is equivalent to adding X, Y times. So here I used Y as a counter where unless it is zero, we will keep adding X to S (which is initially was zero). I am taking care of the carry if carry is not equal to zero then I am performing S=C+A. At last, when Y=0 I am storing the value of the result in the next two consecutive bits.