HSTI – ASSEMBLER PROJECT

2019/2020

Theme 16:
"Copy a block of data from one location in memory to another one"

Professor: Victor Lobo

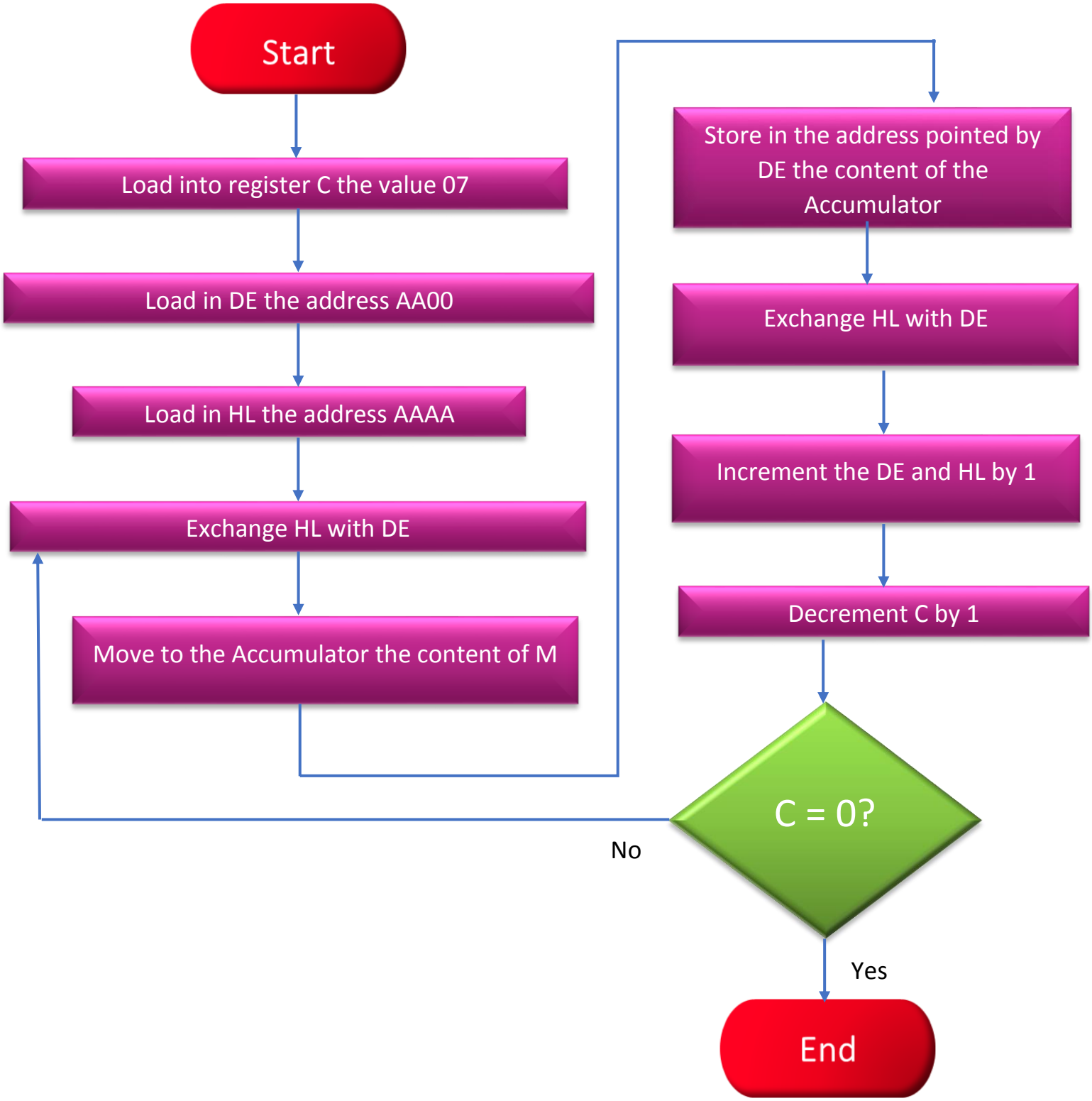
Students:
Ana Lisboa Cotovio
r20181138
Maria Coelho
r20181043

Contents

Objective

The objective of this project is to write a routine that copies a block of data from one location in memory to another one, in Assembly Programming, and to run it in an 8085 microprocessor simulator. The address of the first byte to be copied is going to be passed in register pair DE, and the first destination address is going to be passed in register pair HL. The number of bytes to be copied will be passed in register C. Since the instruction sheet given to us by the professor didn't specify the number of bytes to copy, we decided to copy seven bytes.

Fluxogram


Start
Load into register C the value 07
Load in DE the address AA00
Load in HL the address AAAA
Exchange HL with DE
Move to the Accumulator the content of M
Store in the address pointed by DE the content of the Accumulator
Exchange HL with DE
Increment the DE and HL by 1
Decrement C by 1
C = 0?
No
Yes
End

Memory Map

Table with the information of all the code, mnemonics, addresses of each instruction and number of bytes for each instruction.

Address	Mnemonics	Code	Number of Bytes
0800	MVI C, 05h	0E 07	2
0802	LXI D, AA00h	11 00 AA	3
0805	LXI H, AAAAh	21 AA AA	3
0808	Copy: XCHG	EB	1
0809	MOV A, M	7E	1
080A	STAX D	12	1
080B	XCHG	EB	1
080C	INX D	13	1
080D	INX H	23	1
080E	DCR C	0D	1
080F	JNZ Copy	C2 08 08	3
0812	HLT	76	1

Listing of Code with Comments

Before applying the code, we had to load values into memory so that It may be copied. Our decision in this matter was to load them manually. In order to copy the seven bytes defined before we loaded values in memory from address AA00h until address AA06.

Routine

- ✓ **0E 07:** Load into register C the number of bytes that will be copied, which in this case is seven bytes ([C] <- 07)
- ✓ **11 00 AA:** Load in DE the address of the first byte to be copied ([DE] <- AA00)
- ✓ **21 AA AA:** Load in HL the first destination address ([HL] <- AAAA)
- ✓ **Copy:** Loop that will copy the bytes from the given address in DE to the given address in HL. This Loop will be made as long as C is above 0. In this case it will be executed seven times, since our number of bytes to be copied is seven.
- ✓ **76:** Finish the program (Stop)

Copy

- ✓ **EB**: Exchange the address in HL with the address in DE ([HL] <- [DE] & [DE] <- [HL])
- ✓ **7E**: Move to the Accumulator the content of the address pointed by HL, this is, the content of M ([A] <- M)
- ✓ **12**: Store in the address pointed by DE the content of the Accumulator ([[DE]] <- A)
- ✓ **EB**: Exchange the address in HL with the address in DE ([HL] <- [DE] & [DE] <- [HL])
- ✓ **13**: Increment the register pair DE by 1 in order to move to the next address in memory, which is the next byte's location ([DE] <- [DE] + 1)
- ✓ **23**: Increment the register pair HL by 1 in order to move to the next address in memory, in order not to paste different numbers in the same memory address. If this was made the early bytes would disappear since the would be over written ([HL] <- [HL] + 1)
- ✓ **0D**: Decrement the register C by 1 in order to keep track of how many bytes are left to be copies. ([C] <- [C] – 1)
- ✓ **C2 08 08:** Until the las instruction becomes 0 this instruction will go back to the beginning of cop and perform a loop. The last instruction is the decrementation of the register C. As long as this last operation doesn't load a 1 in the Zero flag, the loop will be repeated. The loop will be performed as many times as the number initially loaded in C. In our case, this loop will be performed seven times. (Jump if not 0 to 0808)

Conclusion

Creating this code, we got to understand better that even the simplest things have more instructions and thought in it than we imagined. The loop depending of the number of bytes to copy was extremely important in this code, and we got to learn it better. Like is was said in the beginning of the project we decided the number of bytes to be copies, but we made a code that would be easily adaptable to a different set number of bytes to copy.

Following all the instructions given to us through the guidance sheet, we think we presented here a short and versatile code which is also efficient and has no errors and a fluxogram that is simple and understandable.

Annexes

0E 07:	uses adresses 0800 and 0801.
	Number 07 hexadecimal is loaded in C
11 00 AA:	uses addresses from 0802 to 0804
	Makes DE point to address AA00
21 AA AA:	uses addresses from 0805 to 0807
	Makes HL point to address AAAA
Copy:	
EB:	uses address 0808
	Swaps HL with DE pointing adresses
7E:	uses address 0809
	Loads Accumulator with memory value correspondent with HL pointing adress
12:	uses address 080A
	Loads address pointed by DE with Accumulator value
EB:	uses address 080B
	Swaps HL with DE pointing adresses
13:	uses address 080C
	Increments DE by 1 each time the loop performs
	Will read and copy addresses from AA00 to AA06
23:	uses address 080D
	Increments HL by 1 each time the loop performs
	Will paste the values to addresses from AAAA to AAB0
0D:	uses address 080E
	Decements C by 1 each time the loop performs
C2 08 08:	uses addresses from 080F to 0811
	Will jump to the beginning of Copy in the C is no 0 yet. Will not perform if the C is 0 and let the program continue
76:	uses address0812
	Finishes the program