**Program No:** 2

**Date:** 13/7/2009

**ADDITION AND SUBTRACTION OF 8-BIT NUMBERS**

**PROBLEM DEFINITION**

The aim is to write an assembly language program to

1. add two 8-bit numbers with and without carry.
2. subtract two 8-bit numbers with and without carry.

**THEORETICAL BACKGROUND**

**8 bit addition**

We load the first 8-bit number into the accumulator from a memory location from where it is moved into another register. Similarly, the next 8-bit number is loaded into accumulator. This value and the register content are then added and resultant is stored in the accumulator. Finally, this result is stored in the required memory location.

**8 bit subtraction**

Inorder to perform subtraction in 8085 one of the data should be in accumulator and other data can be in any one of the general purpose registers. After subtraction the result will be in accumulator. 8085 performs 2’s compliment subtraction and then compliments the carry. Therefore if the result is negative, then carry fleg is set and the accumulator will have 2’s compliment of the result. Hence one of the register is used to store the sign of the result. To get the magnitude of the result take it’s 2’s compliment.

**Instructions used in the program**

LDA 16-bit address:

This is a three byte instruction. The contents of a memory location, specified by a 16-bit address in the operand, are copied to the accumulator.

Eg: LDA 2304H

MOV:

This instruction copies the contents of the source register into the destination register; the contents of the source register are not altered. If one of the operands is a memory location, this location is specified by the contents of the HL registers.

Eg: MOV B, C or MOV B, M

ADD:

The contents of operand (register or memory) are added to the contents of the accumulator and the result is stored in the accumulator. If the operand is a memory location, its location is specified by the contents of the HL registers. All flags are modified to reflect the result of the addition.

Eg: ADD B or ADD M

STA 16-bit address:

The contents of the accumulator are copied into the memory location specified by the operand. This is a 3-byte expression; the second byte specifies the low-order address and the third byte specifies the high-order address.

Eg: STA 2001H

HLT:

The CPU finishes executing the current instruction and halts any further execution. An interrupt or reset is necessary to exit from halt state.

Eg: HLT

**ALGORITHM**

**To add two 8-bit numbers without carry**

Step 1: Start

Step 2: load the first data in accumulator and move it to register B

Step 3: load the second data in accumulator

Step 4: add the contents of register B and accumulator

Step 5: store the sum in memory

Step 6: Stop

**To add two 8-bit numbers with carry**

Step 1: Start

Step 2: load the first data in accumulator and move it to register B

Step 3: load the second data in accumulator

Step 4: clear register C for carry

Step 5: add the contents of register B and accumulator

Step 6: if carry flag is set goto Step 7else goto step 8

Step 7: increment C register

Step 8: store the sum in memory

Step 9: move the carry to accumulator and store it in memory

Step 10: Stop

**To subtract two 8-bit numbers without carry**

Step 1: Start

Step 2: load the data to be subtracted in accumulator and move it to register B

Step 3: load the other operand in accumulator

Step 4: subtract the contents of register B from accumulator

Step 5: store the difference in memory

Step 6: Stop

**To subtract two 8-bit numbers with carry**

Step 1: Start

Step 2: load the subtrahend in accumulator and move it to register B

Step 3: load the minuend in accumulator

Step 4: clear register C for sign of the result

Step 5: subtract the contents of register B from accumulator

Step 6: increment C register.

Step 7: Jump on no carry to step 10.

Step 8: Compliment the accumulator and add 01 H

Step 9: Decrement C register.

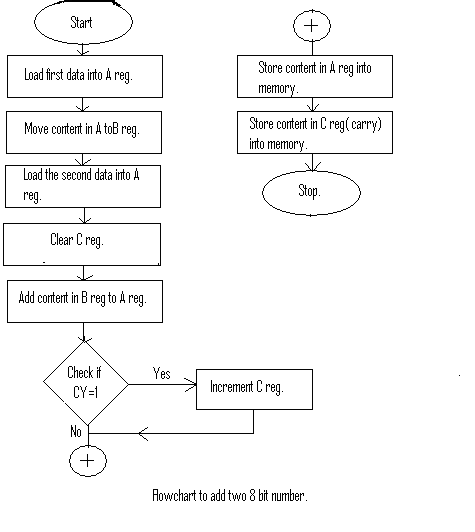
Step 10: store the difference in memory

Step 11: move the content of C register to accumulator and store in memory

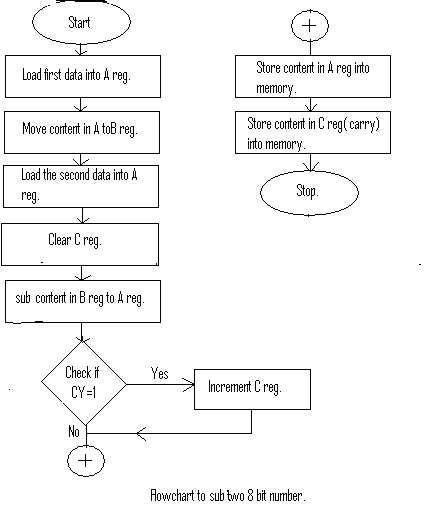
Step 12: Stop

**FLOWCHART**

a. Flowcharts for addition of two 8-bit numbers(with and without carry):

****

b. Flowcharts for subtraction of two 8-bit numbers(with and without carry):

****

**PROGRAM DEVELOPMENT**

**To add two 8-bit numbers without carry**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| MEMORY STATUS | OPCODE | LABEL | MNEMONIC | COMMENTS |
| 2000 | 3A 00 25 | START | *LDA 2500* | Load first data into A reg. |
| 2003 | 47 |  | *MOV B,A* | Store the data in B reg |
| 2004 | 3A 01 25 |  | *LDA 2501* | Load second data into A reg |
| 2007 | 80 |  | *ADD B* | Add contents of reg A and reg B |
| 2008 | 32 03 25 |  | *STA 2502* | Store the result in memory |
| 200B | 76 | HALT | *HLT* | Stop |

**To add two 8-bit numbers with carry**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| MEMORY STATUS | OPCODE | LABEL | MNEMONIC | COMMENTS |
| 2000 | 3A 00 25 | START | *LDA 2500* | Load first data into A reg. |
| 2003 | 47 |  | *MOV B,A* | Store the data in B reg |
| 2004 | 3A 01 25 |  | *LDA 2501* | Load second data into A reg |
| 2007 | 80 |  | *ADD B* | Add contents of B reg and A reg |
| 2008 | 32 02 25 |  | *STA 2502* | Store the result to loc 2502 |
| 200B | 0E 00 |  | *MVI C,00H* | Move to reg c 00H |
| 200D | D2 |  | *JNC NEXT* | If no carry goto NEXT |
| 200E | 0E 01 |  | *MVI C,01H* | Store the result in memory |
| 2010 | 79 | *NEXT* | *MOV A,C* | Move contents of reg C to reg A |
| 2011 | 32 05 25 |  | *STA 2503* | Store the carry in memory |
| 2014 | 76 | HALT | *HLT* | Stop |

**To subtract two 8-bit numbers without carry**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| MEMORY STATUS | OPCODE | LABEL | MNEMONIC | COMMENTS |
| 2000 | 3A 00 25 | START | *LDA 2500* | Load first data into A reg. |
| 2003 | 47 |  | *MOV B,A* | Store the data in B reg |
| 2004 | 3A 01 25 |  | *LDA 2501* | Load second data into A reg |
| 2007 | 90 |  | *SUB B* | Subtract the contents of reg A and reg B |
| 2008 | 32 03 25 |  | *STA 2502* | Store the result in memory |
| 200B | 76 | HALT | *HLT* | Stop |

**To subtract two 8-bit numbers with carry**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| MEMORY STATUS | OPCODE | LABEL | MNEMONIC | COMMENTS |
| 2000 | 3A 00 25 | START | *LDA 2500* | Load first data into A reg. |
| 2003 | 47 |  | *MOV B,A* | Store the data in B reg |
| 2004 | 3A 01 25 |  | *LDA 2501* | Load second data into B reg |
| 2007 | 90 |  | *SUB B* | Subtract the contents of reg A and reg B |
| 2008 | 0E 01 |  | *MVI C,01H* | Load reg c with 00H |
| 200A | DA 11 20 |  | *JC NEXT* | Jump on borrow |
| 200D | 2F |  | *CMA* | Compliment A |
| 200E | 3C |  | *INR A* | Increment A reg |
| 200F | 0E 00 |  | *MVI C,00H* | Load reg c with 01H |
| 2011 | 32 02 25 | NEXT | *STA 2502* | Store difference in memory |
| 2014 | 79 |  | *MOV A,C* | Move contents of reg C to reg A |
| 2015 | 32 03 25 |  | *STA 2503* | Store the borrow in memory |
| 2018 | 76 | HALT | *HLT* | stop |

**TESTING STRATEGIES**

**Test Case 1:**

|  |  |  |
| --- | --- | --- |
| Memory Address | Data | Comments |
| 2500H | 02H | Operand 1 |
| 2501H | 05H | Operand 2 |

**Test Case 2:**

|  |  |  |
| --- | --- | --- |
| Memory Address | Data | Comments |
| 2500H | 07H | Operand 1 |
| 2501H | 08H | Operand 2 |

**Test Case 3:**

|  |  |  |
| --- | --- | --- |
| Memory Address | Data | Comments |
| 2500H | 02H | Operand 1 |
| 2501H | 05H | Operand 2 |

**Test Case 4:**

|  |  |  |
| --- | --- | --- |
| Memory Address | Data | Comments |
| 2500H | 05H | Operand 1 |
| 2501H | 03H | Operand 2 |

**SUMMARY OF RESULTS**

Case 1:

|  |  |  |
| --- | --- | --- |
| MEMORY ADDRESS | DATA | COMMENTS |
| 2502 | 07H | 07H is the sum stored in 2502H. |

Case 2:

|  |  |  |
| --- | --- | --- |
| MEMORY ADDRESS | DATA | COMMENTS |
| 2502 | 0FH | 0F is the sum stored in 2502H |
| 2503 | 00H | 01 is the carry stored in 2503.H |

Case 3:

|  |  |  |
| --- | --- | --- |
| MEMORY ADDRESS | DATA | COMMENTS |
| 2502 | 03H | 04 is the difference stored in 2502H |

Case 4:

|  |  |  |
| --- | --- | --- |
| MEMORY ADDRESS | DATA | COMMENTS |
| 2502 | 02H | 02 is the difference stored in 2502H. |
| 2503 | 01H | 01 is the borrow stored in 2503H. |

**CONCLUSION**

The program to add and subtract two 8-bit numbers with and without carry was written and test run successfully.