**8-BIT SUBTRACTION**

**EXP NO: 2**

**AIM:** To write an assembly language program to implement 8-bit subtraction using 8085 processor.

**ALGORITHM:**

1) Start the program by loading the first data into the accumulator.

2) Move the data to a register.

3) Get the second data and load it into the accumulator.

4) Subtract the two register contents.

5) Check for borrow.

6) Store the difference and borrow in the memory location.

7) Halt.

**PROGRAM:**

|  |  |
| --- | --- |
| **MNEMONICS** | **EXPLANATION** |
| LDA 8000 | Load accumulator with the first number in the address |
| MOV B, A | Move the data from accumulator to B register |
| LDA 8001 | Load accumulator with the second number in the address |
| SUB B | Subtract the data B register with accumulator |
| STA 8002 | Store the data (Output) of the accumulator in the address |
| RST 1 | HALT |

**INPUT:**

|  |  |
| --- | --- |
| **ADDRESS** | **DATA** |
| 8000 | 4 |
| 8001 | 5 |

**OUTPUT:**

|  |  |
| --- | --- |
| **ADDRESS** | **DATA** |
| 8002 | 1 |

**RESULT:** Thus the program was executed successfully using 8085 processor simulator

**8-BIT MULTIPLICATION**

**EXP NO: 3**

**AIM:** To write an assembly language program to implement 8-bit multiplication using 8085 processor.

**ALGORITHM:**

1) Start the program by loading a register pair with the address of memory location.

2) Move the data to a register.

3) Get the second data and load it into the accumulator.

4) Add the two register contents.

5) Increment the value of the carry.

6) Check whether the repeated addition is over.

7) Store the value of product and the carry in the memory location.

8) Halt.

**PROGRAM:**

|  |  |
| --- | --- |
| **MNEMONICS** | **EXPLANATION** |
| LDA 2200 | Load the accumulator with the first number in the address 8500 |
| MOV E,A | Move the data from accumulator to ‘E’ register |
| MVI D,00 | Move the immediate value 00 into |
| LDA 2201 | Load the accumulator number in the address 2201 |
| MOV C,A | Move the data from a  accumulator to ‘C’ register |
| LXI H,0000 | Load the immediate value 0000 into the HL register pair |
| BACK: DAD D | Back : Label for the loop  D ADD: Add the value in register D |
| DCR C | Decrement register E by 1 |
| JNZ BACK | In register E, is not 0,jump back to the beginning of the loop |
| SHLD 2202 | Store the value on the HL register pair at memory address 2202 |
| HLT | HALT |

**INPUT:**

|  |  |
| --- | --- |
| **ADDRESS** | **DATA** |
| 2200 | 4 |
| 2201 | 2 |

**OUTPUT:**

|  |  |
| --- | --- |
| **ADDRESS** | **DATA** |
| 2202 | 8 |

**RESULT:** Thus the program was executed successfully using 8085 processor simulator.

**8-BIT DIVISION**

**EXP NO: 4**

**AIM:** To write an assembly language program to implement 8-bit division using 8085 processor.

**ALGORITHM:**

1) Start the program by loading a register pair with the address of memory location.

2) Move the data to a register.

3) Get the second data and load it into the accumulator.

4) Subtract the two register contents.

5) Increment the value of the carry.

6) Check whether the repeated subtraction is over.

7) Store the value of quotient and the reminder in the memory location.

8) Halt.

**PROGRAM:**

|  |  |
| --- | --- |
| **MNEMONICS** | **EXPLANATION** |
| START: NOP | It is often used for code alignment |
| LDA 8500 | Load the accumulator with first number in address 8500 |
| MOV B, A | Move data from accumulator to ‘B’ register |
| LDA 8501 | Load the accumulator with second number in the address 8501 |
| MVI C,00 H | Move the immediate value 00 into register ‘C’ |
| LOOP:CMP B | Loop : Label for loop  CMP B: Compare the value in accumulator (A) with (B) |
| JC LOOP1 | If the carrying is (A<B) jump to label loop 1 |
| SUB B | Subtract the value in register (B) from the accumulator (A) |
| INR C | Increment register C by 1 |
| JMP LOOP | Jump back to the beginning of the loop |
| STA 8502 | Store the data in accumulator 8502 |
| MOV A, C | Move the data from `C’ register to accumulator |
| STA 8503 | Store the data in the accumulator in 8503 |
| RST 1 | Typically transfer control to a predefined interrup service routine |
| HLT | HALT |

**INPUT:**

|  |  |
| --- | --- |
| **ADDRESS** | **DATA** |
| 8500 | 2 |
| 8501 | 6 |

**OUTPUT:**

|  |  |
| --- | --- |
| **ADDRESS** | **DATA** |
| 8502 | 0 |
| 8503 | 3 |

**RESULT:** Thus the program was executed successfully using 8085 processor simulator.