**EXP NO: 1 8-BIT ADDITION**

**Date:**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**AIM:**

To write an assembly language program to implement 8-bit addition using 8085 and 8086 processors.

**SOFTWARE: GNUSIM8085, EMU8086**

**ALGORITHM -8085**

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. Add the two register contents with only sum values.
5. Then Check for carry and add with carry
6. Store the value of sum and carry in the memory location.
7. Halt the program.

## **ALGORITHM-8086**

1. Load data from offset 500 to register AL (first number)
2. Load data from offset 501 to register BL (second number)
3. Add these two numbers (contents of register AL and register BL)
4. Apply DAA instruction (decimal adjust)
5. Store the result (content of register AL) to offset 600
6. Set register AL to 00
7. Add contents of register AL to itself with carry
8. Store the result (content of register AL) to offset 601
9. Stop

**PROGRAM:**

|  |  |  |  |
| --- | --- | --- | --- |
| **8085** | | **8086** | |
| **Without Carry**  LDA 8500  MOV B, A  LDA 8501  ADD B  STA 8502  RST 1  **With Carry**  LDA 2050  MOV H, A LDA 2051  ADD H  MOV L, A  MVI A 00  ADC A  MOV H, A  SHLD 3050  HLT | A🡨[8500]  B🡨A  A🡨[8501]  A🡨A+B  A🡨[8502]  A🡨[2050]  H🡨A  A🡨 [2051]  A🡨A+H  L←A  A←00  A←A+A+carry  H←A  **H→3051, L→3050** | MOV AL, [500]  MOV BL, [501] ADD AL, BL  DAA  MOV [600], AL MOV AL, 00  ADC AL, AL  MOV [601],  HLT  END | AL ← [500]  BL ← [501]  AL ← AL+BL  DECIMAL ADJUST  AL → [600]  AL ← 00  AL ← AL+AL+cy(prev)  AL AL → [601] |

**INPUT:**

**OUTPUT:**

**RESULT:** Thus, the program was executed successfully using 8085 and 8086 processors simulator.

**EXP NO: 2 8-BIT SUBTRACTION**

**Date:**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**AIM:** To write an assembly language program to implement 8-bit subtraction using 8085 and 8086 processor.

**SOFTWARE: GNUSIM8085, EMU8086**

**ALGORITHM-8085:**

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.

**ALGORITHM-8085:**

1. Load data from offset 500 to register AL (first number)
2. Load data from offset 501 to register BL (second number)
3. Subtract these two numbers (contents of register AL and register BL)
4. Apply DAS instruction (decimal adjust)
5. Store the result (content of register AL) to offset 600
6. Set register AL to 00
7. Add contents of register AL to itself with carry (borrow)
8. Store the result (content of register AL) to offset 601
9. Stop

**PROGRAM:**

|  |  |  |  |
| --- | --- | --- | --- |
| **8085** | | **8086** | |
| **Without Borrow**  LDA 8000  MOV B, A  LDA 8001  SUB B  STA 8002  RST 1  **With Borrow**  LDA 3052  MOV H, A  LDA 3053  SUB H  MOV L,A  MVI A, 00H  SBB A  MOV H, A  SHLD 3053  HLT | A🡨[8000]  B🡨A  A🡨[8001]  A🡨A-B  A🡨[8502]  A🡨[3052]  H🡨A  A🡨 [3053]  A🡨A-H  L←A  A←00  A←A-borrow  H←A  **H→3051, L→3050** | MOV AL, [500]  MOV BL, [501]  SUB AL, BL  DAS  MOV [600], AL  MOV AL, 00  ADC AL, AL  MOV [601], AL  HLT | AL ← [500]  BL ← [501]  AL ← AL-BL  DECIMAL ADJUST AL  AL → [600]  AL ← 00  AL ← AL+AL+cy(prev)  AL → [601]  END |

**INPUT:**

**OUTPUT:**

**RESULT:** Thus, the program was executed successfully using 8085 and 8086 processor simulators.

**EXP NO: 3**

**Date: 8-BIT MULTIPLICATION**

**---------------------------------------------------------------------------------------------------**

**AIM:** To write an assembly language program to implement 8-bit multiplication using 8085 and 8086 processors.

**SOFTWARE: GNUSIM-8085, EMU-8086**

**ALGORITHM:**

1. Start the program by loading a register pair with the address of memory location.
2. Get the first data and load it into the accumulator.
3. Move the data to register B.
4. Get the second data and load it into the accumulator.
5. Move the data to register C.
6. Assign zero to accumulator.
7. Add the Accumulator content with B register contents.
8. Decrement the value of the C register.
9. Check whether the repeated addition is over.
10. Store the value of product and the carry in the memory location.
11. Halt.

**ALGORITHM-8086**

1. Start the program by loading a register pair with the address of memory location.
2. Get the first data from the given offset address and load it into the accumulator.
3. Get the second data from the given offset address and load it into the B register.
4. Multiply the two data and store the result in the given memory location.

A diagram of a algorithm

Description automatically generated

**PROGRAM:**

|  |  |  |  |
| --- | --- | --- | --- |
| **8085** | | **8086** | |
| LDA 8500  MOV B, A  LDA 8501  MOV C, A  XRA A (MVI A,00H) LOOP: ADD B  DCR C  JNZ LOOP  STA 8502  RST 1 | A🡨[8500]  B🡨A  A🡨[8501]  C🡨A  0🡨A  A=A+B  C-1🡨C  JUMP to Loop  if C≠0  A🡨[8502] | MOV AX, [1100]  MOV BX, [1101]  MUL BX  MOV [1102], AX  HLT  MOV SI, [500]  MOV DI, [550]  MOV AX, [SI]  INC [SI]  MOV BX, [SI]  MUL BX  MOV [DI], AX  HLT | AX🡨[1100]  BX🡨[1101]  AX🡨AX\*BX  [1102] 🡨AX  END  SI🡨[500]  DI🡨[550]  AX🡨[SI]  SI=SI+1  BX🡨[SI]  AX🡨AX\*BX  DI 🡨AX  END |

**INPUT:**

**OUTPUT:**

**RESULT:** Thus, the program was executed successfully using 8085 and 8086 processor simulators.

**EXP NO: 4**

**Date: 8-BIT DIVISION**

**---------------------------------------------------------------------------------------------------------------------**

**AIM:** To write an assembly language program to implement 8-bit division using 8085 and 8086 processors.

**SOFTWARE: GNUSIM8085, EMU8086**

**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.

**ALGORITHM-8086**

1. Start the program by loading a register pair with the address of memory location.
2. Get the first data from the given offset address and load it into the accumulator.
3. Get the second data from the given offset address and load it into the B register.
4. Multiply the two data and store the result in the given memory location.

**PROGRAM:**

|  |  |  |  |
| --- | --- | --- | --- |
| **8085** | | **8086** | |
| LDA 8501  MOV B, A  LDA 8500  MVI C,00  LOOP:CMP B  JC LOOP1  SUB B  INR C  JMP LOOP  LOOP1: STA 8502  MOV A, C  STA 8502  RST 1 | A🡨[8501]  B🡨A  A🡨[8500]  C🡨 00  Compare A & B  If carry, store  else subtract.  C=C++  Go to compare.  **Store the remainder**  **Store the Quotient** | MOV AX, [500]  MOV BX, [501]  DIV BX  MOV [600], AX  HLT  MOV SI, 500  MOV DI, 600  MOV AL, [SI]  INC SI  MOV BL, [SI]  DIV BL  MOV [DI], AX  HLT | AX 🡪[500]  BX 🡪 [501]  AX 🡪AX/BX  AX 🡪 [600]  END  SI 🡪[500]  DI 🡪[600]  AL🡨[SI]  SI=SI+1  BL🡨[SI]  AX=AL/BL  AX🡪[DI]  END |

**INPUT:**

**OUTPUT:**

**RESULT:** Thus, the program was executed successfully using 8085 and 8086 processor simulators.