|  |  |  |
| --- | --- | --- |
|  | 8-BIT ARITHMETIC OPERATIONS USING 8051 |  |
| Exp No.: 12 |  | **Name:** S Vishakan |
| Date: 21-10-2020 |  | **Reg. No:** 18 5001 196 |

**AIM:**

To write assembly language programs to perform the following arithmetic operations using an 8051 microcontroller:

1. 8-bit addition
2. 8-bit subtraction
3. 8-bit multiplication
4. 8-bit division

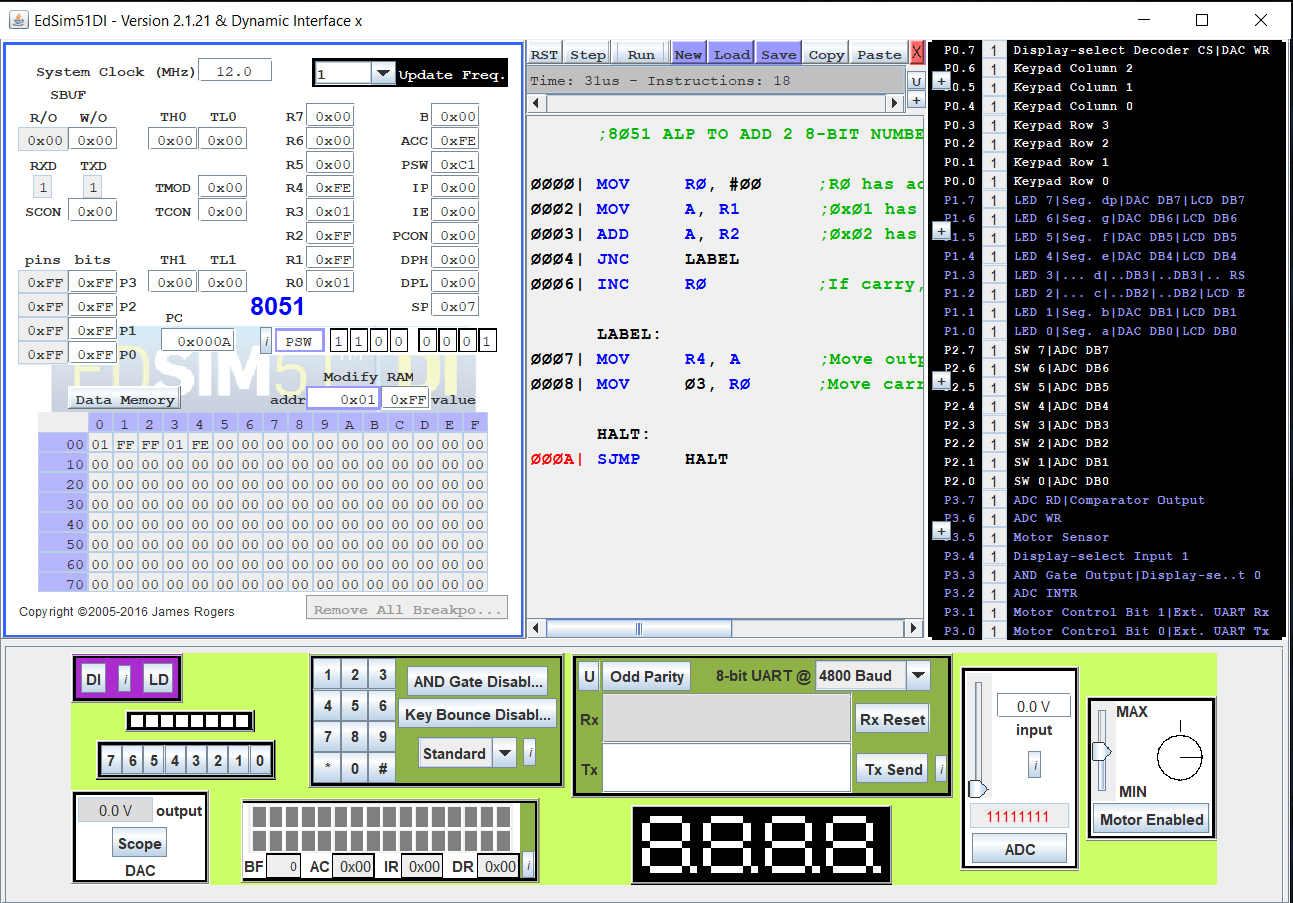
**PROGRAM – 1: 8-BIT ADDITION:**

**ALGORITHM:**

1. Begin
2. Initialize R0 with 00h.
3. Move the value in R1 to A.
4. Add the value in A to with value in R2.
5. Increment R0 if carry is produced.
6. Move R0 to R3 (carry) and A to R4 (sum).
7. End.

|  |  |
| --- | --- |
| **PROGRAM** | **COMMENTS** |
| MOV R0, #00 | R0 has address of 0x00 |
| MOV A, R1 | 0x01 has 1st 8-bit number |
| ADD A, R2 | 0x02 has 2nd 8-bit number. Add it with A |
| JNC LABEL | If no carry, jump to “LABEL”. |
| INC R0 | If carry, increment R0 |
|  |  |
| LABEL: |  |
| MOV R4, A | Move output to R4 from A |
| MOV 03, R0 | Move carry to R3. (MOV R3, R0) is invalid |
|  |  |
| HALT: |  |
| SJMP HALT | Halt the program with a loop. |

**SAMPLE I/O SNAPSHOT:**

****

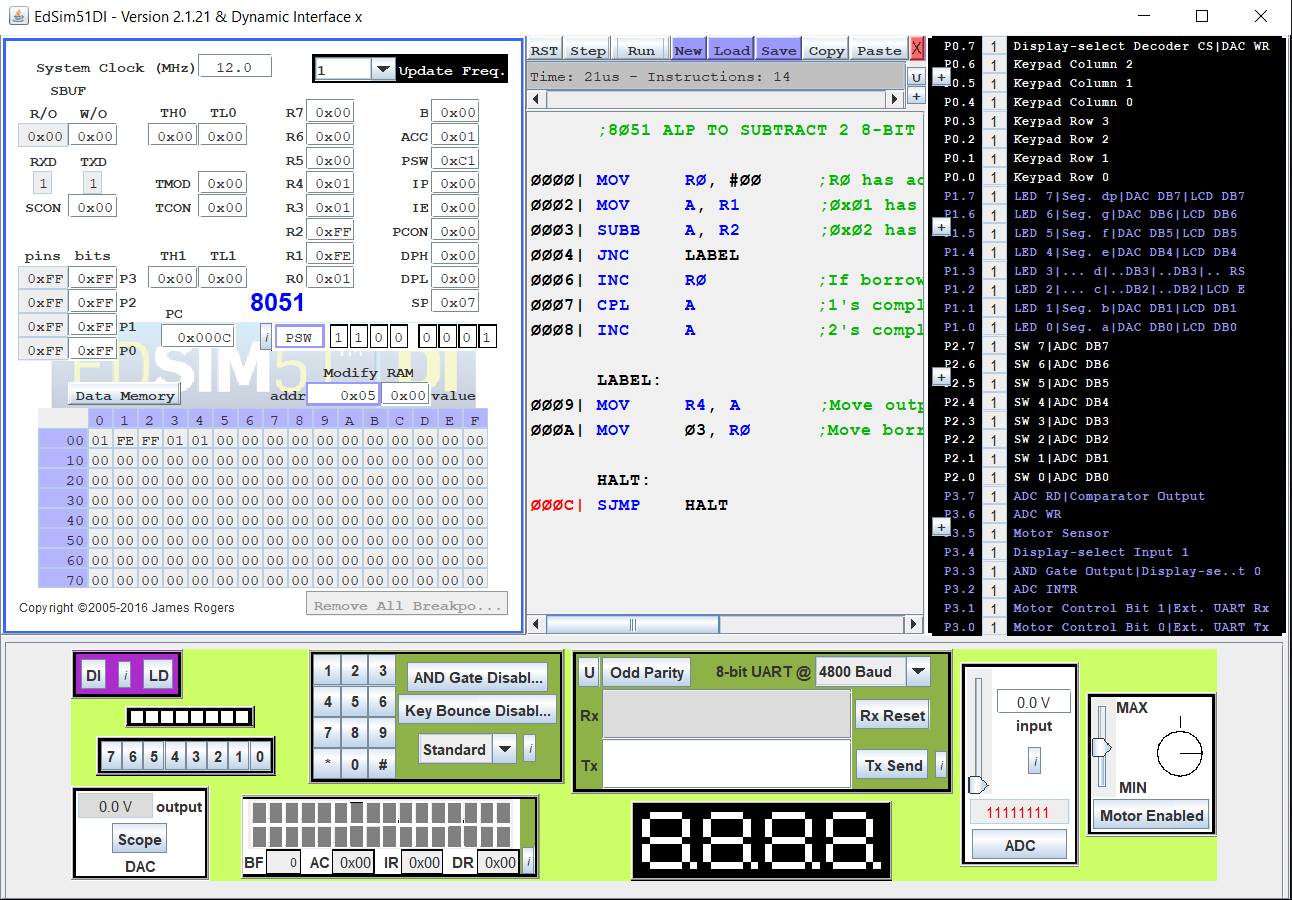
**PROGRAM – 2: 8-BIT SUBTRACTION:**

**ALGORITHM:**

1. Begin.
2. Initialize R0 with 00h
3. Move the value in R1 to A.
4. Subtract the value in A to with value in R2.
5. Increment R0 if carry is produced and take 2's complement of A.
6. Move R0 to R3 (borrow) and A to R4 (difference)
7. End.

|  |  |
| --- | --- |
| **PROGRAM** | **COMMENTS** |
| MOV R0, #00 | R0 has address of 0x00 |
| MOV A, R1 | 0x01 has 1st 8-bit number |
| SUBB A, R2 | ;0x02 has 2nd 8-bit number. Subtract it from A. |
| JNC LABEL | If no carry, jump to “LABEL”. |
| INC R0 | If carry, increment R0 |
| CPL A | 1's complement the difference |
| INC A | 2's complement the difference |
|  |  |
| LABEL: |  |
| MOV R4, A | Move output to R4 from A |
| MOV 03, R0 | Move carry to R3. (MOV R3, R0) is invalid |
|  |  |
| HALT: |  |
| SJMP HALT | Halt the program with a loop. |

**SAMPLE I/O SNAPSHOT:**

****

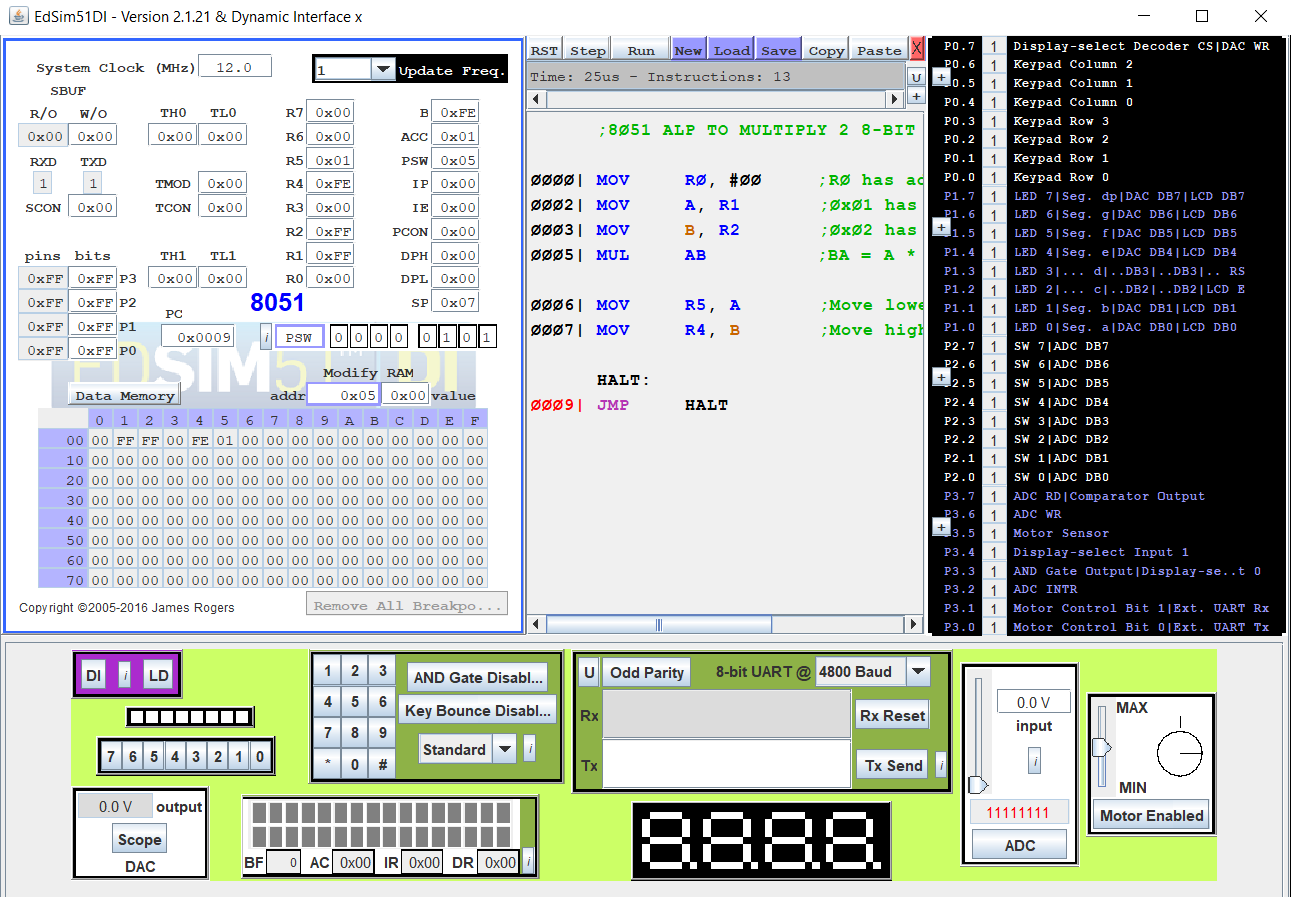
**PROGRAM – 3: 8-BIT MULTIPLICATION:**

**ALGORITHM:**

1. Begin.
2. Initialize R0 with 00h
3. Move the value in R1 to A.
4. Move the value in R2 to B.
5. Multiply A and B.
6. Move B to R4 (MSB of product) and A to R5 (LSB of product)
7. End.

|  |  |
| --- | --- |
| **PROGRAM** | **COMMENTS** |
| MOV R0, #00 | R0 has address of 0x00 |
| MOV A, R1 | 0x01 has 1st 8-bit number |
| MOV B, R2 | 0x02 has 2nd 8-bit number |
| MUL AB | BA = A \* B |
|  |  |
| MOV R5, A | Move lower byte to R5 from A |
| MOV R4, B | Move higher byte to R4 from B |
|  |  |
| HALT: |  |
| SJMP HALT | Halt the program with a loop. |

**SAMPLE I/O SNAPSHOT:**

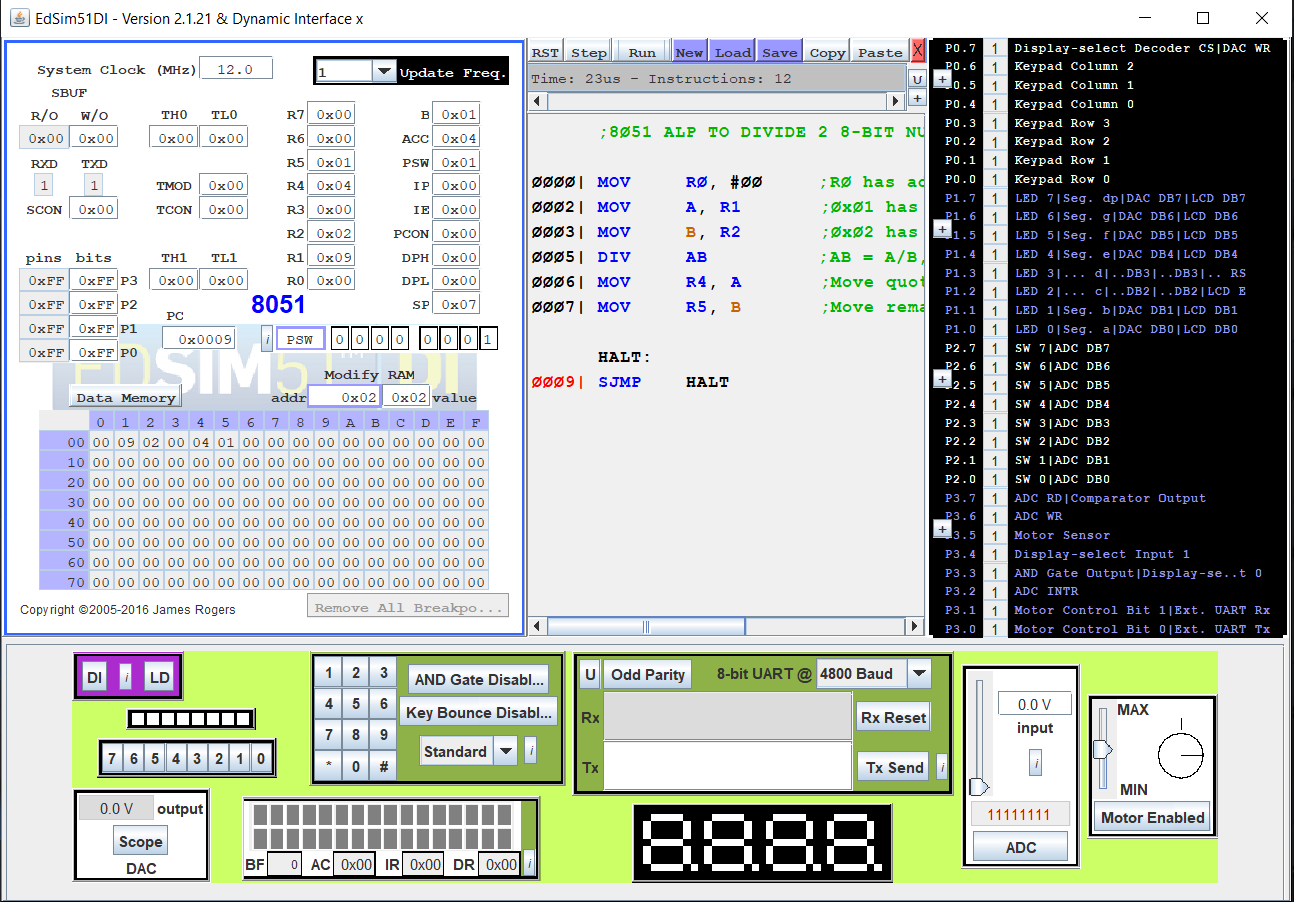
****

**PROGRAM – 4: 8-BIT DIVISION:**

**ALGORITHM:**

1. Begin.
2. Initialize R0 with 00h.
3. Move the value in R1 to A.
4. Move the value in R2 to B.
5. Divide A by B.
6. Move A to R4 (quotient) and B to R5 (remainder)
7. End.

|  |  |
| --- | --- |
| **PROGRAM** | **COMMENTS** |
| MOV R0, #00 | R0 has address of 0x00 |
| MOV A, R1 | 0x01 has 1st 8-bit number |
| MOV B, R2 | 0x02 has 2nd 8-bit number |
| DIV AB | BA = A / B, A: Quotient, B: Remainder |
|  |  |
| MOV R5, A | Move quotient to R4 from A |
| MOV R4, B | Move remainder to R5 from B |
|  |  |
| HALT: |  |
| SJMP HALT | Halt the program with a loop. |

**SAMPLE I/O SNAPSHOT:**

**RESULT:**

The assembly level programs were written to perform the above specified 8-bit arithmetic operations using an 8051 microcontroller and the outputs were verified.