**Homework #3**

**20200639 채우진**

3.13. Multiplicand: 01100010(2), Multiplier: 00010010(2)

Lets denote binary value stored in Multiplicand register A, Product register B for convenience..

|  |  |  |  |
| --- | --- | --- | --- |
| Iteration | Step | A (8 bits) | B (16 bits) |
| 0 | Initial values | 01100010 | 00000000 00010010 |
| 1 | 1. LSB of B=0 → No operation | 01100010 | 00000000 00010010 |
| 2. Shift Right B | 01100010 | 00000000 00001001 |
| 2 | 1 LSB of B=1 → Add A on upper 8 bits of B | 01100010 | 01100010 00001001 |
| 2. Shift Right B | 01100010 | 00110001 00000100 |
| 3 | 1 LSB of B=0 → No operation | 01100010 | 00110001 00000100 |
| 2. Shift Right B | 01100010 | 00011000 10000010 |
| 4 | 1 LSB of B=1 → Add A on left 8 bits of B | 01100010 | 00011000 10000010 |
| 2. Shift Right B | 01100010 | 00001100 01000001 |
| 5 | 1 LSB of B=1 → Add A on left 8 bits of B | 01100010 | 01101110 01000001 |
| 2. Shift Right B | 01100010 | 00110111 00100000 |
| 6 | 1 LSB of B=0 → No operation | 01100010 | 00110111 00100000 |
| 2. Shift Right B | 01100010 | 00011011 10010000 |
| 7 | 1 LSB of B=0 → No operation | 01100010 | 00011011 10010000 |
| 2. Shift Right B | 01100010 | 00001101 11001000 |
| 8 | 1 LSB of B=0 → No operation | 01100010 | 00001101 11001000 |
| 2. Shift Right B | 01100010 | 00000110 11100100 |

The value of the product register after 8th iteration is 00000110 11100100 in binary, which is 1764 in decimal.

3.18 Dividend: 74(111100), Divisor: 21(010001)

Lets denote the value of Quotient Register A, Divisor register to be B, Remainder register to be C for convenience.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Iteration | Step | A (6 bits) | B (12 bits) | C (12 bits) |
| 0 | Initial values | 000000 | 010001 000000 | 000000 111100 |
| 1 | 1. C = C-B | 000000 | 010001 000000 | 101111 111100 |
| 2. C < 0 → C = C+B, A[0] = 0 | 000000 | 010001 000000 | 000000 111100 |
| 3. Shift B right | 000000 | 001000 100000 | 000000 111100 |
| 2 | 1. C = C-B | 000000 | 001000 100000 | 111000 011100 |
| 2. C < 0 → C = C+B, A[0] = 0 | 000000 | 001000 100000 | 000000 111100 |
| 3. Shift B right | 000000 | 000100 010000 | 000000 111100 |
| 3 | 1. C = C-B | 000000 | 000100 010000 | 111100 101100 |
| 2. C < 0 → C = C+B, A[0] = 0 | 000000 | 000100 010000 | 000000 111100 |
| 3. Shift B right | 000000 | 000010 001000 | 000000 111100 |
| 4 | 1. C = C-B | 000000 | 000010 001000 | 111110 110100 |
| 2. C < 0 → C = C+B, A[0] = 0 | 000000 | 000010 001000 | 000000 111100 |
| 3. Shift B right | 000000 | 000001 000100 | 000000 111100 |
| 5 | 1. C = C-B | 000000 | 000001 000100 | 111111 111000 |
| 2. C < 0 → C = C+B, A[0] = 0 | 000000 | 000001 000100 | 000000 111100 |
| 3. Shift B right | 000000 | 000000 100010 | 000000 111100 |
| 6 | 1. C = C-B | 000000 | 000000 100010 | 000000 011010 |
| 2. C > 0 → A[0]=1 | 000001 | 000000 100010 | 000000 011010 |
| 3. Shift B right | 000001 | 000000 010001 | 000000 011010 |
| 7 | 1. C = C-B | 000001 | 000000 010001 | 000000 001001 |
| 2. C > 0 → A[0]=1 | 000011 | 000000 010001 | 000000 001001 |
| 3. Shift B right | 000011 | 000000 001000 | 000000 001001 |

After 7th iteration, the value of quotient register is 000011(2), which is 3 in decimal, and the value of remainder register is 000000 001001(2), which is 9 in decimal.

3.27

Sign bit: 1

Exponent: 12 → 01100 in 5-bit representation

Fraction: 0.25 → 0100000000 in 10-bit representation

|  |  |  |
| --- | --- | --- |
| 1 | 01100 | 0100000000 |

Therefore, is 1011000100000000

4.7

Opcode: 101011(2) = 43 (Store), Rs: 00011(2) = 3, Rt: 00010(2) = 2, Address: 00000000000010100(2) = 20

4.7.1

|  |  |
| --- | --- |
| Output of Sign-Extended | Output of Jump Shift Left 2 |
| 00000000000000000000000000010100 | 00011000100000000000001010000 |

4.7.2

|  |  |
| --- | --- |
| ALUOp[1-0] | Instruction[5-0] |
| 00 | 010100 |

4.7.3

|  |  |
| --- | --- |
| New PC | Path |
| PC + 4 | PC → Adder(PC+4) → branch Mux → jump Mux → PC |

4.7.4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| RegDst Mux  ~ Write Register | ALUSrc Mux  ~ALU | MemtoReg Mux  ~ Write data | Branch Mux  ~ Jump Mux | Jump Mux  ~PC |
| 2 or 20 | 20 | X | PC+4 | PC+4 |

4.7.5

|  |  |  |
| --- | --- | --- |
| ALU | Add(PC+4) | Add(Branch) |
| -3 and 20 | PC and 4 | PC+4 and 80 |

4.7.6

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Read Register 1 | Read Register 2 | Write Register | Write data | RegWrite |
| -3 | 2 | X | X | 0 |