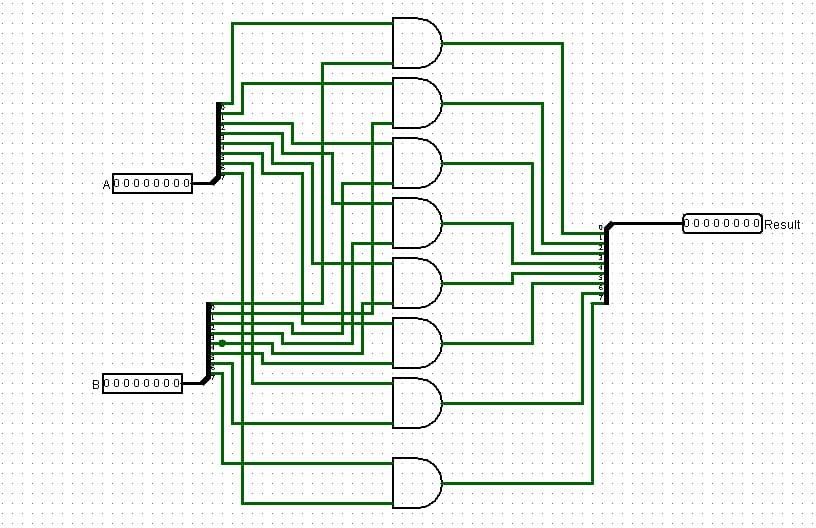
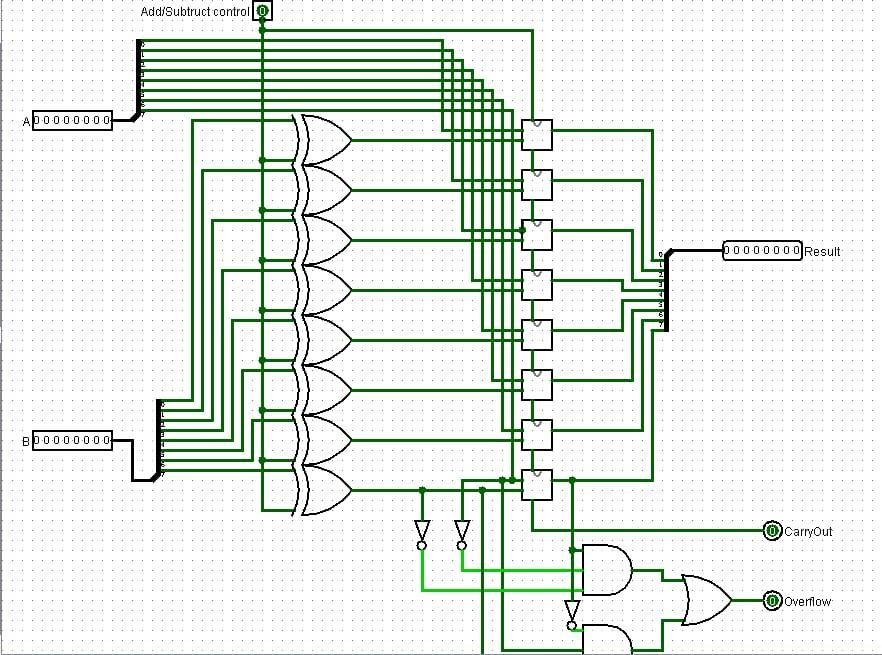
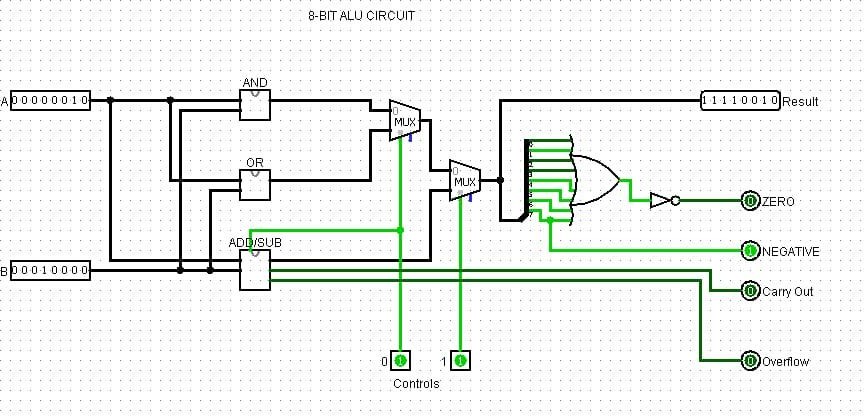
***Phase*** 1



***Phase2***

|  |  |  |
| --- | --- | --- |
| Binary Code | Instruction Name | Description |
| 0000 | NOP | No operation |
| 0001 | ADD | Add the values in the accumulator (register A) and the operand register (register B), storing the result in the accumulator (register A) |
| 0010 | SUB | Subtract the value in the operand register (register B) from the value in the accumulator (register A), storing the result in the accumulator (register A) |
| 0011 | MUL | Multiply the values in the accumulator (register A) and the operand register (register B), storing the result in the accumulator (register A) |
| 0100 | DIV | Divide the value in the accumulator (register A) by the value in the operand register (register B), storing the quotient in the accumulator (register A) |
| 0101 | AND | Perform a bitwise AND operation on the values in the accumulator (register A) and the operand register (register B), storing the result in the accumulator (register A) |
| 0110 | OR | Perform a bitwise OR operation on the values in the accumulator (register A) and the operand register (register B), storing the result in the accumulator (register A) |
| 0111 | XOR | Perform a bitwise XOR operation on the values in the accumulator (register A) and the operand register (register B), storing the result in the accumulator (register A) |
| 1000 | SHL | Shift the value in the accumulator (register A) left by the number of bits specified in the operand register (register B), storing the result in the accumulator (register A) |
| 1001 | SHR | Shift the value in the accumulator (register A) right by the number of bits specified in the operand register (register B), storing the result in the accumulator (register A) |

Phase3



***Assembly Code***

; Add Operands until 0

ADD:

ADD ; Add the value in register B to the accumulator (register A)

CMP B, 0 ; Compare the value in register B with 0

JNE ADD ; If register B is not zero, jump back to the ADD instruction

; The final result is in register A

HLT ; Halt the program

; Shift Right until Least Significant Bit is 0 or the Value is 0xFF

SHIFT:

MOV B, 0 ; Initialize the shift count to 0

LOOP:

SHR ; Shift the value in register A right by the number of bits in register B

INC B ; Increment the shift count

CMP A, 0xFF ; Check if the value in register A is 0xFF

JE END ; If the value is 0xFF, jump to the end

CMP A, 0x01 ; Check if the least significant bit is 0

JNE LOOP ; If the least significant bit is not 0, jump back to the LOOP label

END:

; The final shifted value is in register A

; The total number of shifts is in register B

HLT ; Hal