

*Alexandria University*

*Faculty of Engineering*

*Computer and Systems Engineering Dept.*

*CS131: Digital Computer Fundamentals*



ALU Assignment

**Designing 4-Bit Arithmetic Logic Unit**

**Group:** B2

**Names:**

1. Esraa Gamal (09)
2. Elsayed Abdelnasser (10)
3. Omnia Magdy (11)
4. Eman Rafik (12)
5. Yomna Gamal (62)
6. **Problem Statement:**

Designing a simple Arithmetic Logic Unit with two inputs 4-bit each and an input operation of 4-bits.

The ALU have an 8-bit output and three more outputs for the arithmetic operations.

ALU provides the following operation:

* Arithmetic Operations:
* Addition.
* Subtraction.
* Multiplication.
* Logic Operations:
* Bitwise AND.
* Bitwise OR.
* Bitwise XOR.
* Bitwise NAND.
* Bitwise Operations:
* 2’s Complement.
* 1’s Complement.
* Logical Shift.
* Arithmetic shift.
* Circular Shift.

1. **Truth Table and Minimization:**

* **Adder:**
* 1-bit Adder:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S** | **Cout** | **B** | **A** | **Cin** |
| 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 | 1 |
| 0 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 |

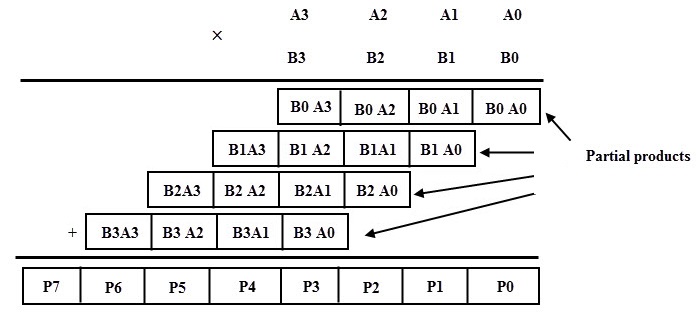
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sum |  |  | A |  |
|  |  | 1 |  | 1 |
| Cin | 1 |  | 1 |  |
|  |  | B |  |  |

Sum = A’B’Cin + AB’Cin’+ A’BCin’ + ABCin

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Cout |  |  | A |  |
|  |  |  | 1 |  |
| Cin |  | 1 | 1 | 1 |
|  |  | B |  |  |

C out = AB + ACin +BCin

* **Subtractor:**
* The numbers of bits in the subtrahend is 5 while that of minuend is 6. We make the number of bits in the subtrahend equal to that of minuend by taking a `0’ in the sixth place of the subtrahend.
* Now, 2’s complement of 010110 is (101101 + 1) i.e.101010. Adding this with the minuend.
* 1     1 0 1 1 0      Minuend  
    
                            1     0 1 0 1 0      2’s complement of subtrahend  
    
     Carry over 1       1     0 0 0 0 0      Result of addition
* After dropping the carry over we get the result of subtraction to be 100000.
* **Multiplier:**
* The first partial product is obtained by the AND gate which is nothing but a least significant bit of the multiplication result. Since the second partial product is shifted to the left position, the first partial second term and second partial product first term is added by half adder and produce the sum output along with the carry out.
* This carry out is added at the next half adder as an input as shown in figure. Likewise, it produces the multiplication result of two binary numbers by using the simple circuit configuration. The multiplication of the two 2 bit number results a 4-bit binary number.
* If we consider two unsigned 4 bit numbers multiplication in which the multiplicand, A is equal to A3A2 A1A0 and the multiplier B is equal to B3B2B1B0. The partial products are produced depending on each multiplier bit multiplied by the multiplicand.



* **Bitwise AND:**
* For 1- bit:

|  |  |  |
| --- | --- | --- |
| **A and B** | **B** | **A** |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 0 | 0 | 1 |
| 1 | 1 | 1 |

* For all a, b, c, d, w, x, y and z:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| d | c | b | a | A |
| z | y | x | w | B |
| d  and  z | c and y | b and  x | a and w | output |

* **Bitwise OR:**
* For 1- bit:

|  |  |  |
| --- | --- | --- |
| **A or B** | **B** | **A** |
| 1 | 0 | 0 |
| 1 | 1 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |

* For all a, b, c, d, w, x, y and z:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| d | c | b | a | A |
| z | y | x | w | B |
| d  or  z | c  or  y | b  or  x | a  or  w | output |

* **Bitwise XOR:**
* For 1- bit:

|  |  |  |
| --- | --- | --- |
| **A or B** | **B** | **A** |
| 0 | 0 | 0 |
| 1 | 1 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |

* For all a, b, c, d, w, x, y and z:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| d | c | b | a | A |
| z | y | x | w | B |
| d  xor  z | c  xor  y | b  xor  x | a  xor  w | output |

* **Bitwise NAND:**
* For 1- bit:

|  |  |  |
| --- | --- | --- |
| **A or B** | **B** | **A** |
| 1 | 0 | 0 |
| 1 | 1 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |

* For all a, b, c, d, w, x, y and z:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| d | c | b | a | A |
| z | y | x | w | B |
| d  nand  z | c  nand  y | b  nand  x | a  nand  w | output |

* **2’s Complement:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **S4** | **S3** | **S2** | **S1** | **D** | **C** | **B** | **A** |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 |
| 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| S1 |  |  | C |  |  |
|  |  | 1 | 1 | 1 |  |
|  | 1 | 1 | 1 | 1 |  |
| A |  |  |  |  | B |
|  | 1 |  |  |  |  |
|  |  |  | D |  |  |

S1 = A’B + A’C + A’D + AB’C’D’

S2 = B’C + B’D + BC’D’

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S2 |  |  | C |  |
|  |  | 1 | 1 | 1 |
|  | 1 |  |  |  |
| A | 1 |  |  |  |
|  |  | 1 | 1 | 1 |
|  |  |  | D |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| S3 |  |  | C |  |  |
|  |  | 1 |  | 1 |  |
|  |  | 1 |  | 1 |  |
| A |  | 1 |  | 1 | B |
|  |  | 1 |  | 1 |  |
|  |  |  | D |  |  |

S3 = C’D + D’C

S4 = D

* **1’s Complement:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **S4** | **S3** | **S2** | **S1** | **D** | **C** | **B** | **A** |
| 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 |
| 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 |
| 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 |
| 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 |
| 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |

S1 = A’

S2 = B’

S3 = C’

S4 = D’

* **Left Logical Shift:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **S4** | **S3** | **S2** | **S1** | **D** | **C** | **B** | **A** |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

S1 = B

S2 = C

S3 = D

S4 = 0

* **Right Logical Shift:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **S4** | **S3** | **S2** | **S1** | **D** | **C** | **B** | **A** |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 | 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 |
| 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 |

S1 = 0

S2 = A

S3 = B

S4 = C

* **Left Arithmetic Shift:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **S4** | **S3** | **S2** | **S1** | **D** | **C** | **B** | **A** |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

S1 = B

S2 = C

S3 = D

S4 = 0

* **Right Arithmetic Shift:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **S4** | **S3** | **S2** | **S1** | **D** | **C** | **B** | **A** |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 | 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 |
| 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

S1 = A

S2 = A

S3 = B

S4 = C

* **Left Circular Shift:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **S4** | **S3** | **S2** | **S1** | **D** | **C** | **B** | **A** |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | 0 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

S1 = B

S2 = C

S3 = D

S4 = A

* **Right Circular Shift:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **S4** | **S3** | **S2** | **S1** | **D** | **C** | **B** | **A** |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

S1 = D

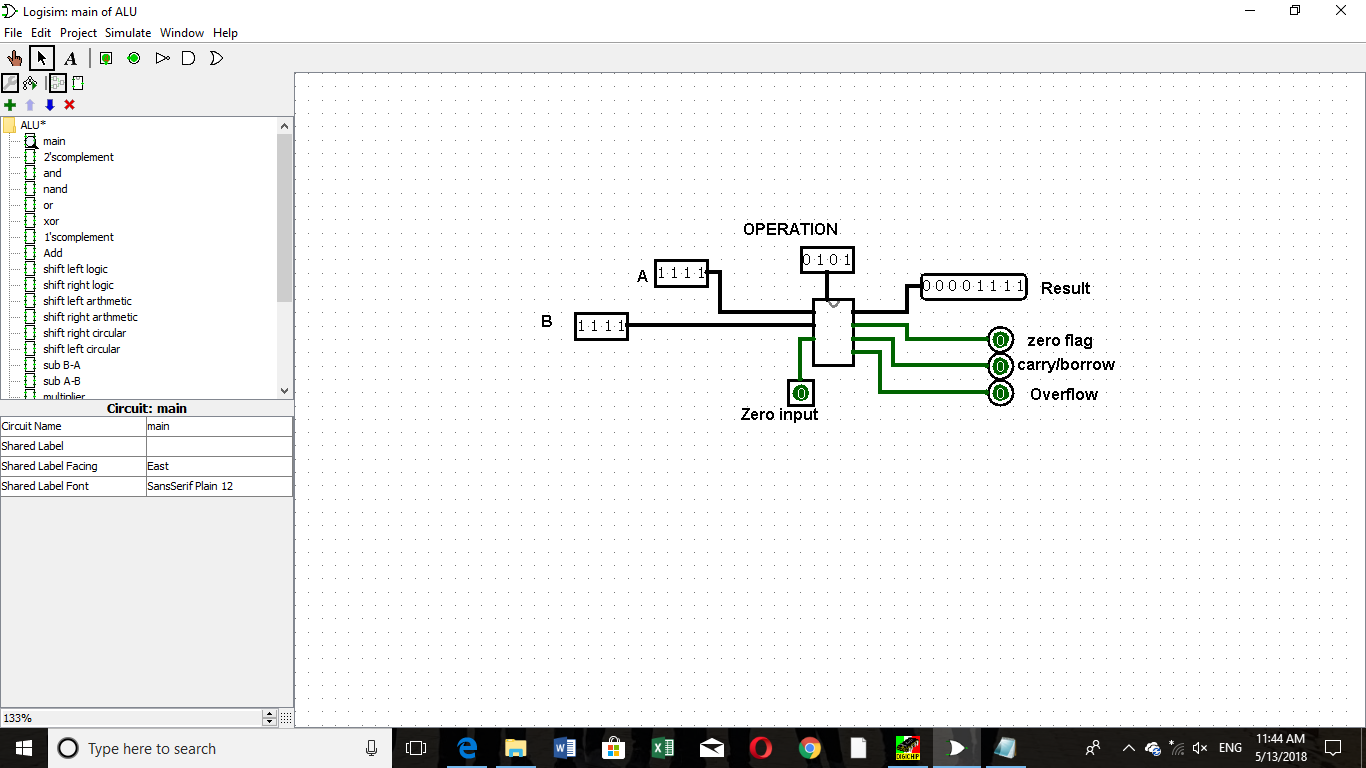
S2 = A

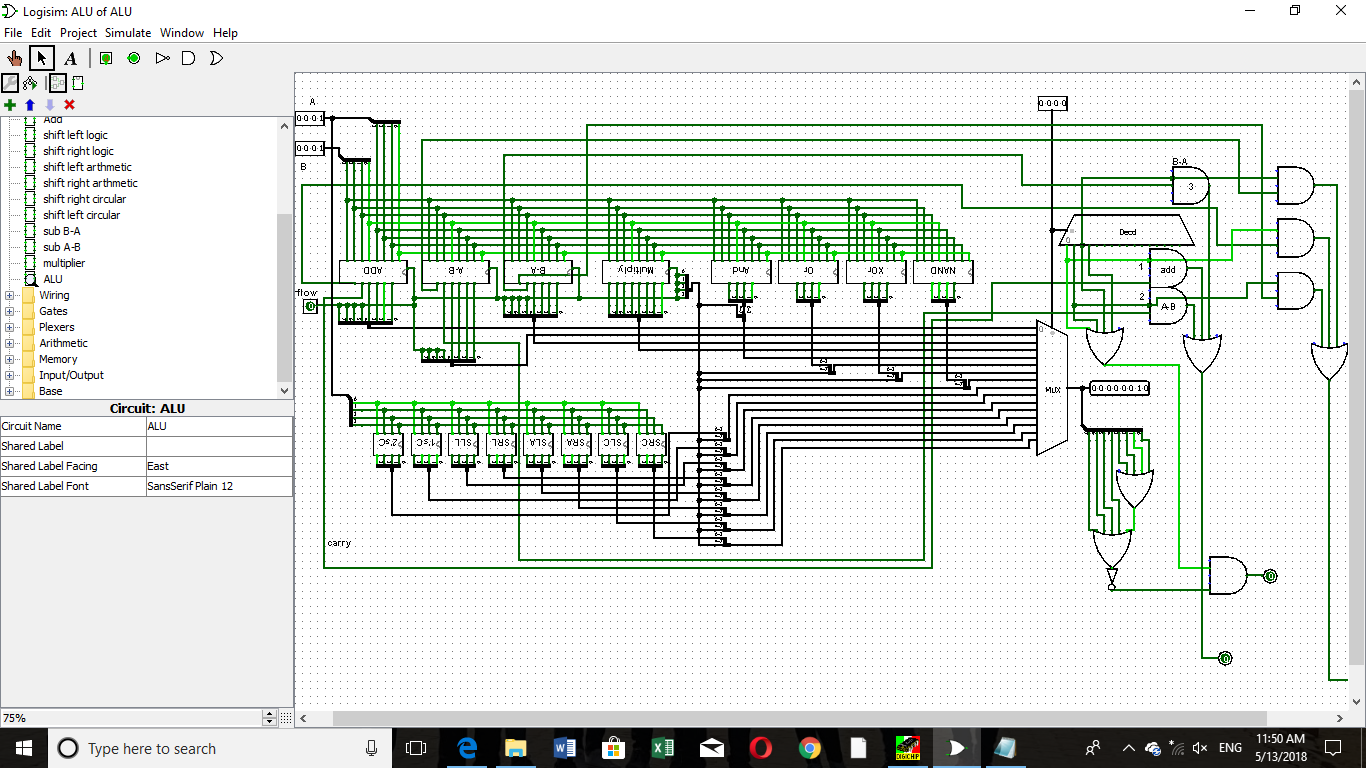
S3 = B

S4 = C

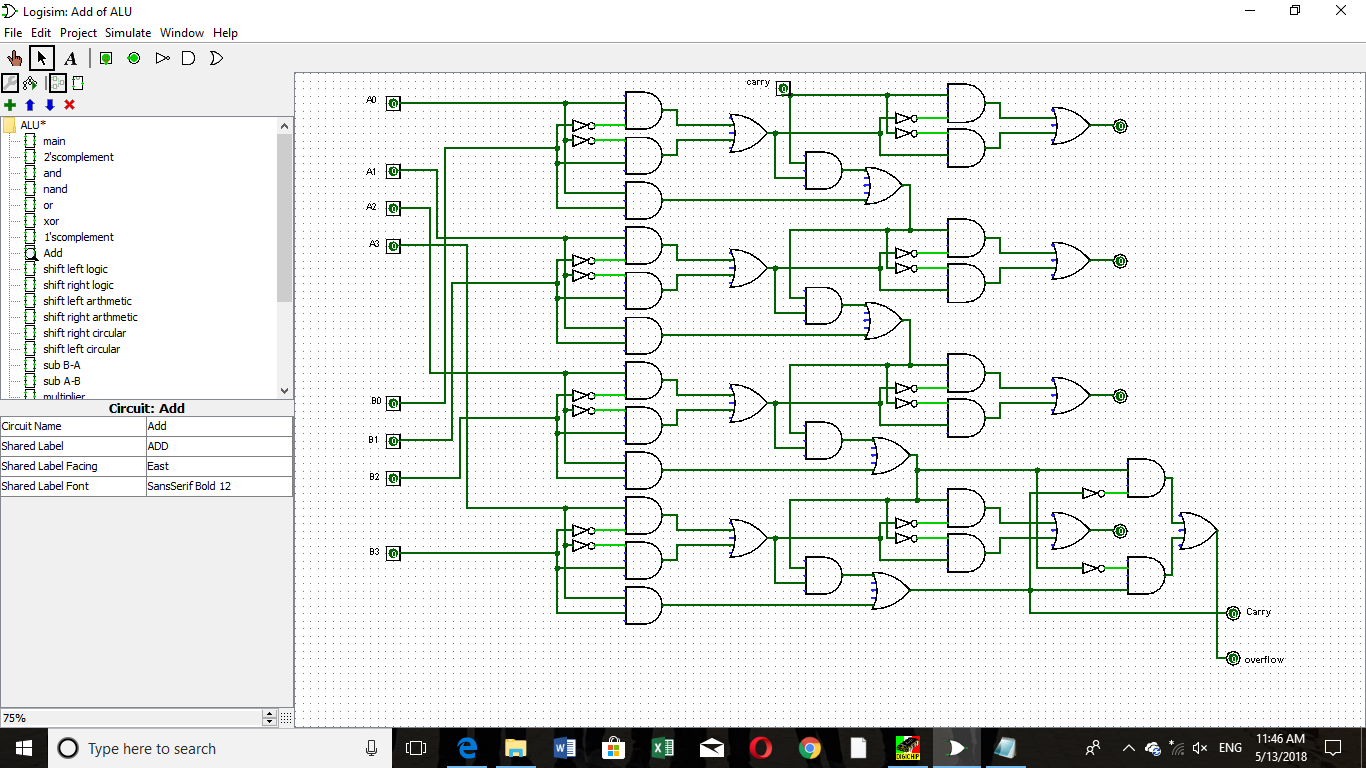
1. **Circuits Diagram:**

* Main Circuit:

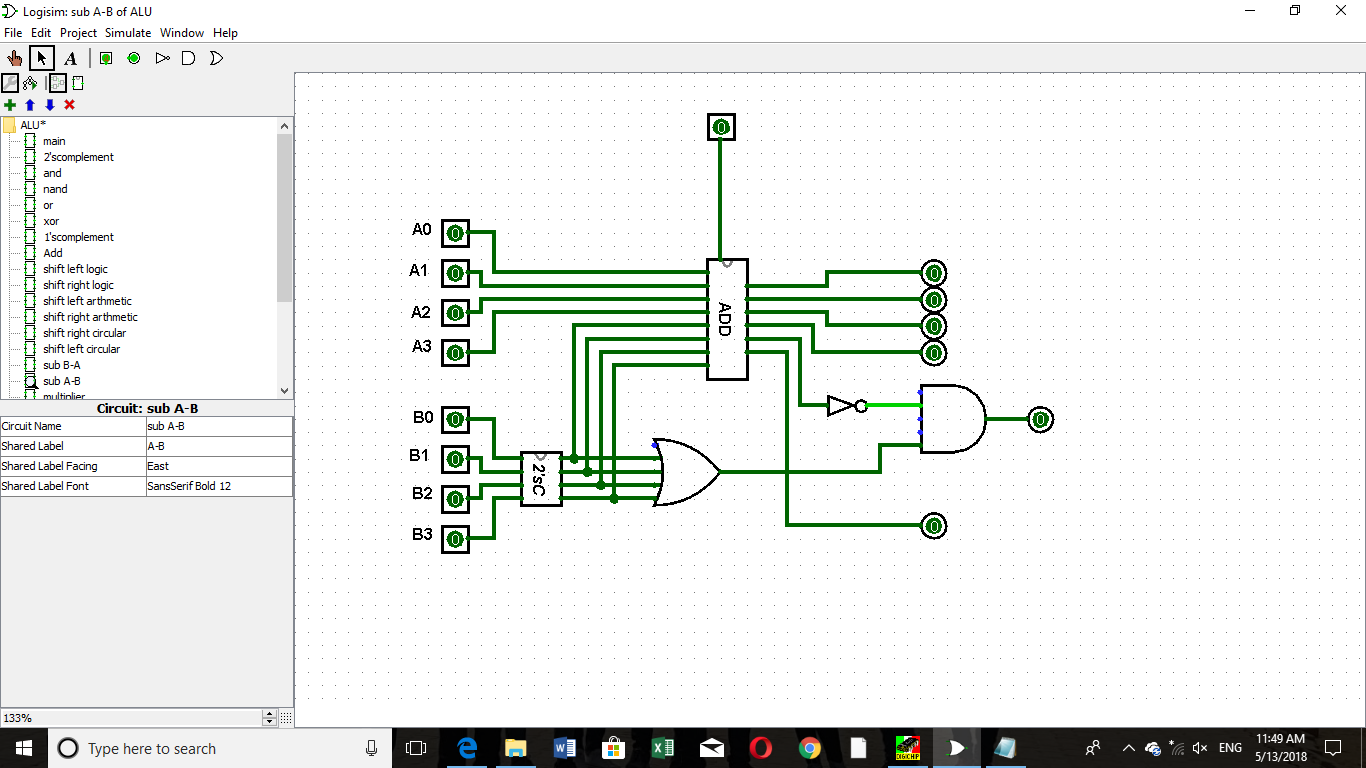




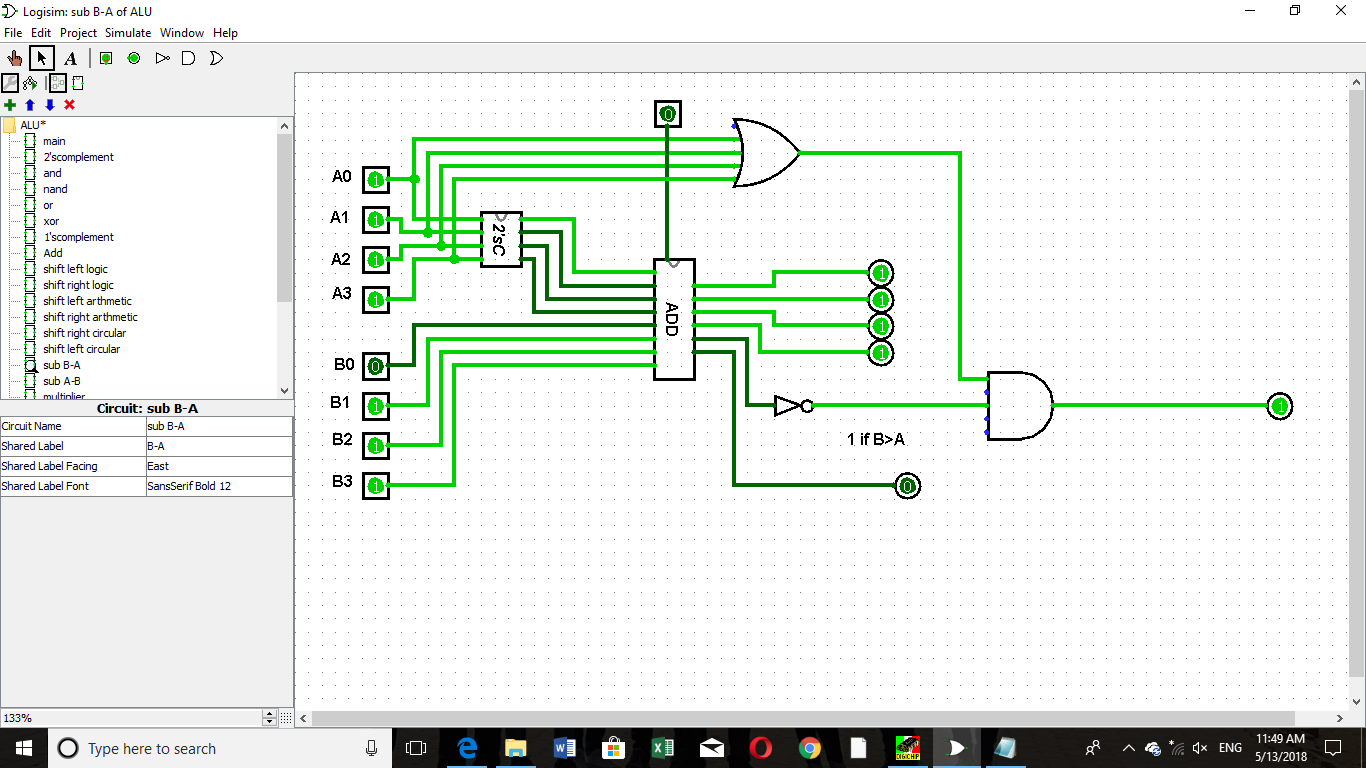
* Full Adder:



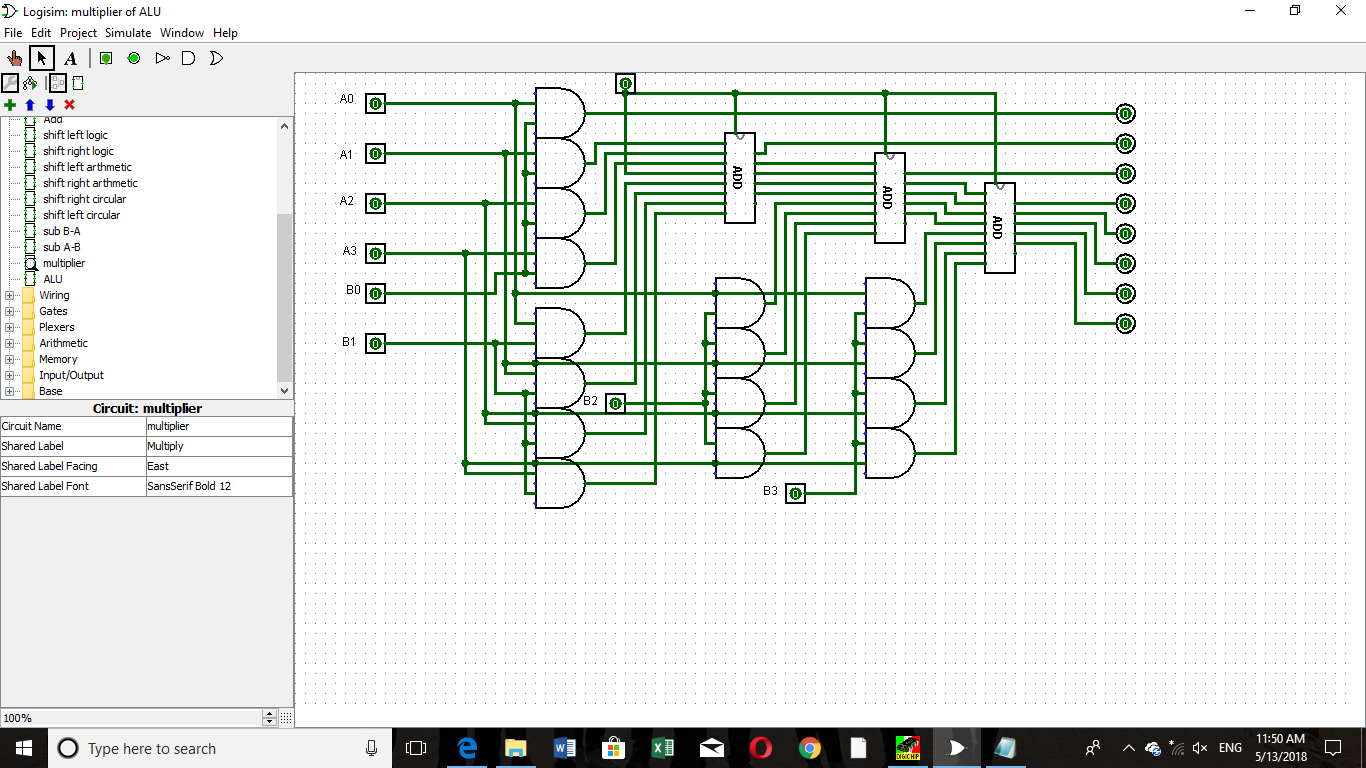
* Full Subtractor (A – B):



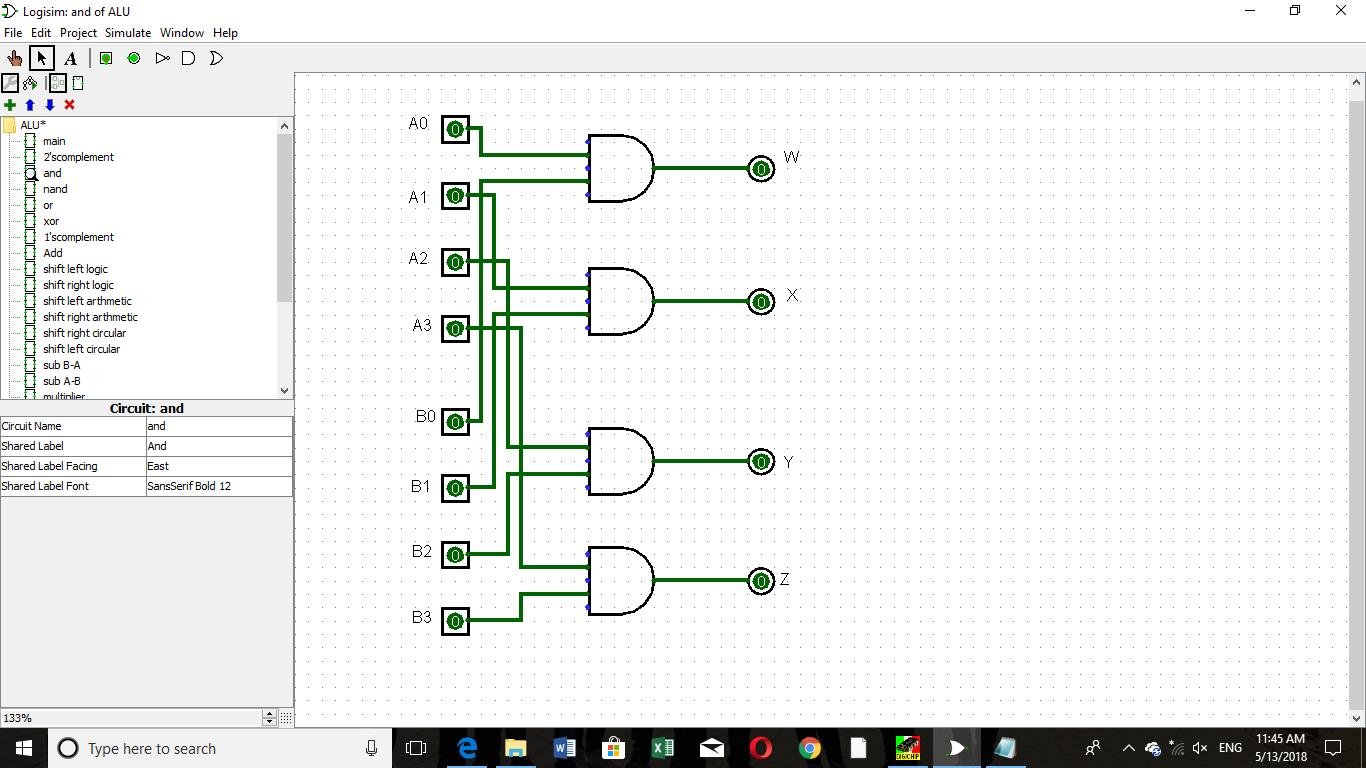
* Full Subtractor (B – A):



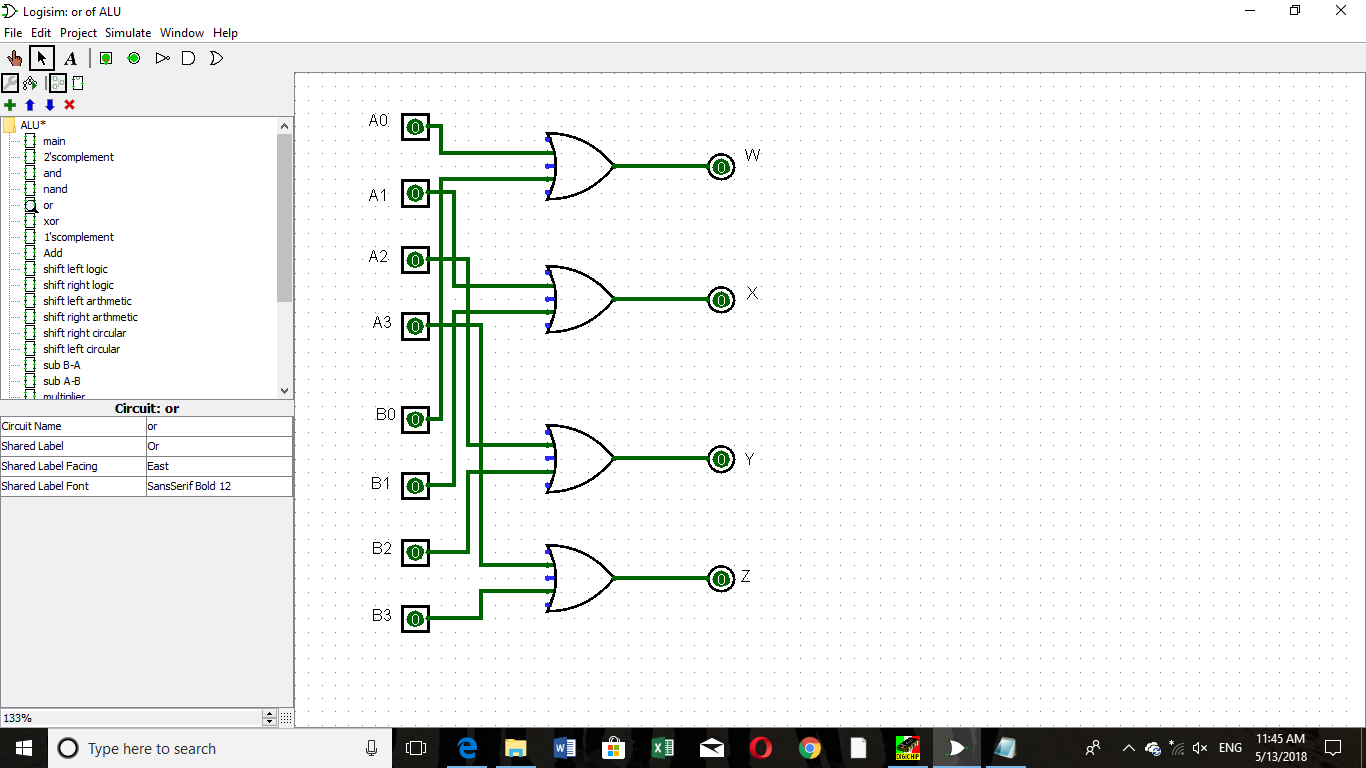
* Multiplier:



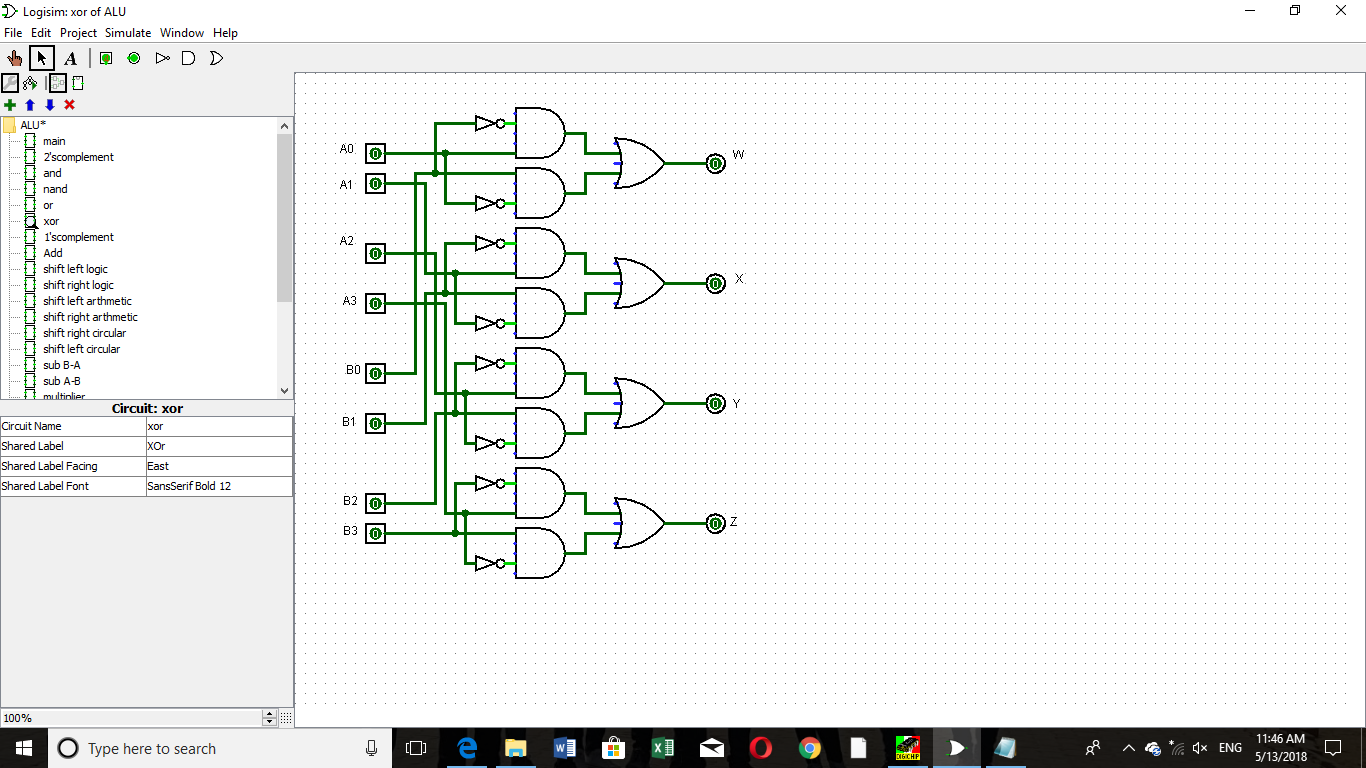
* Bitwise AND:



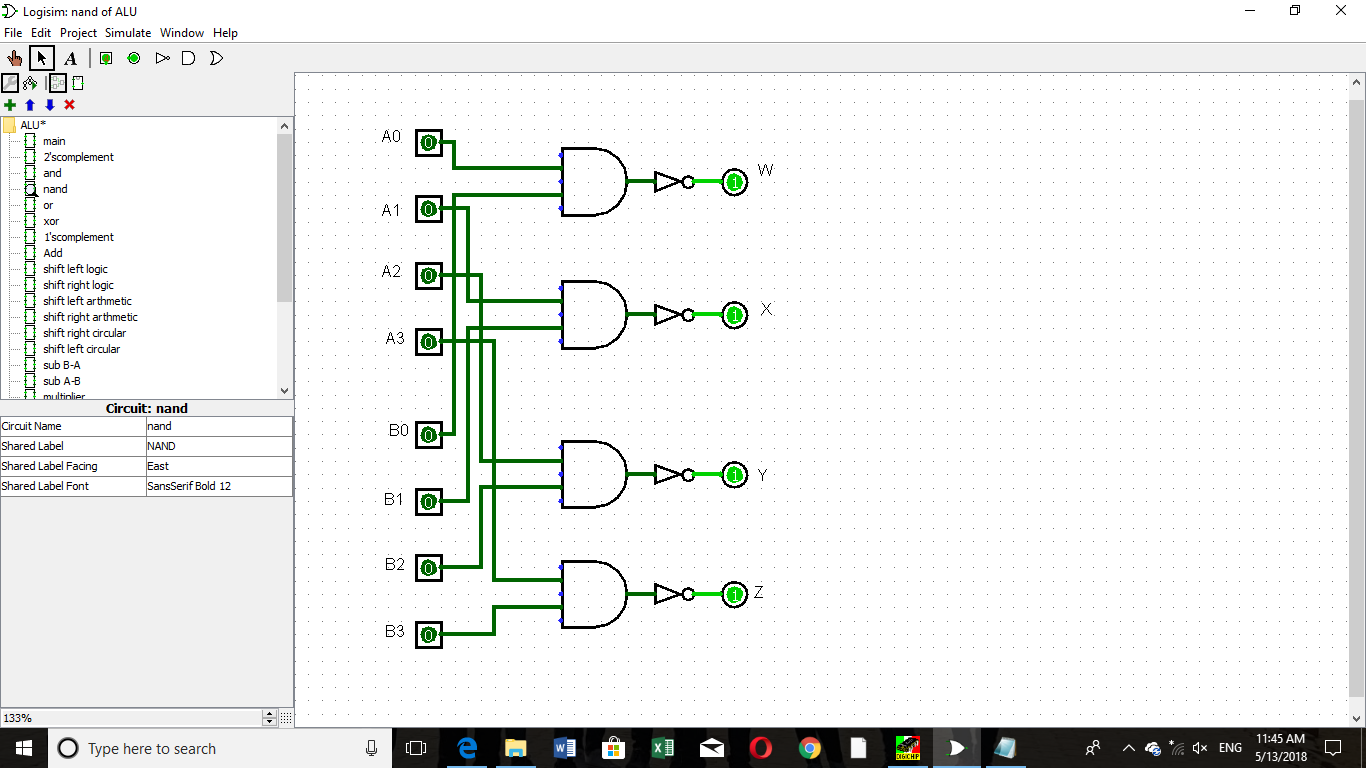
* Bitwise OR:



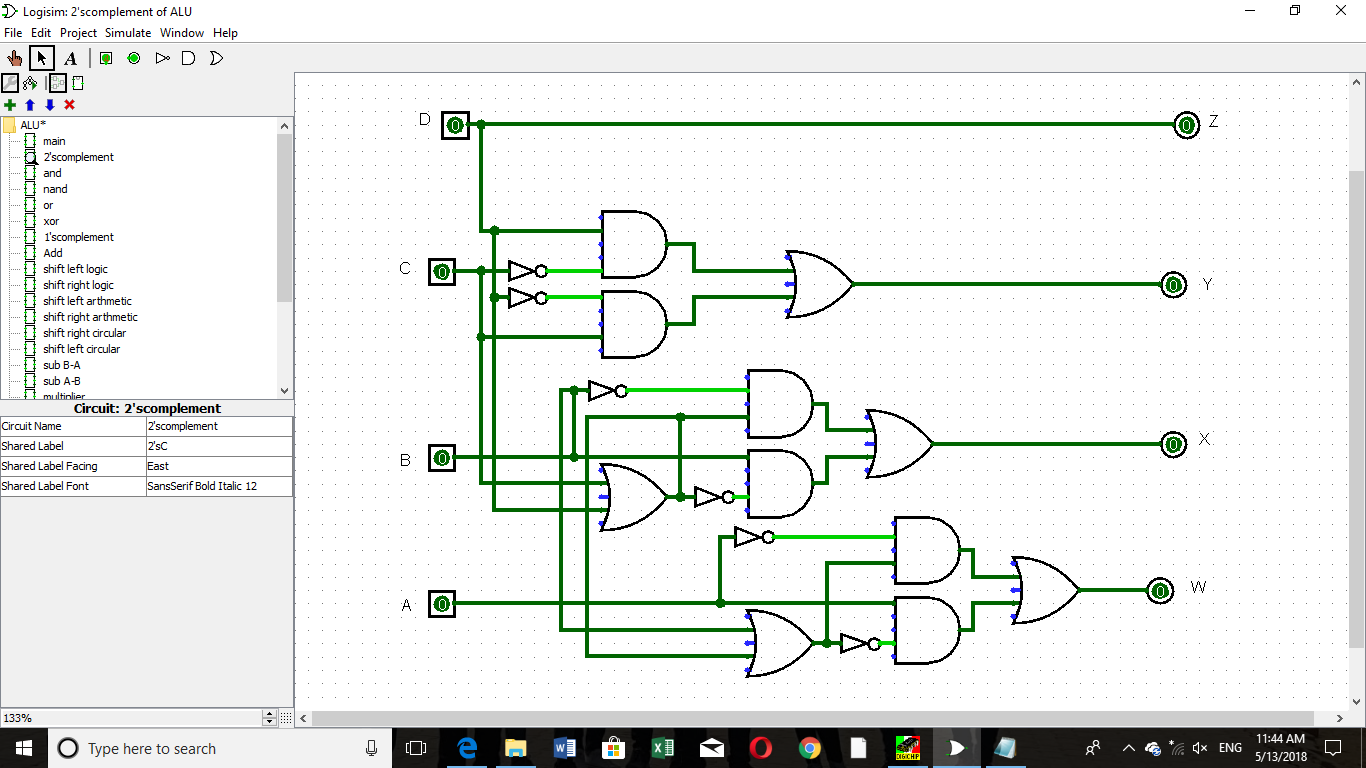
* Bitwise XOR:



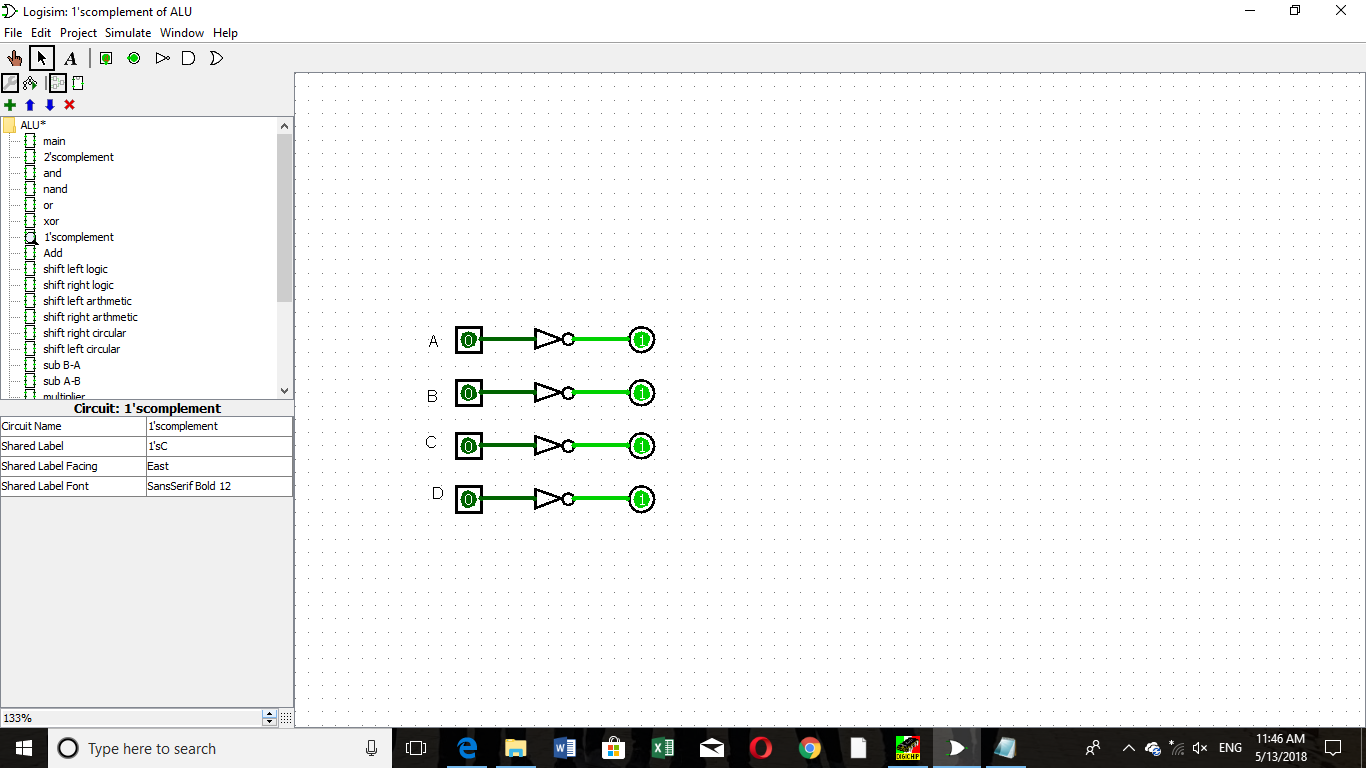
* Bitwise NAND:



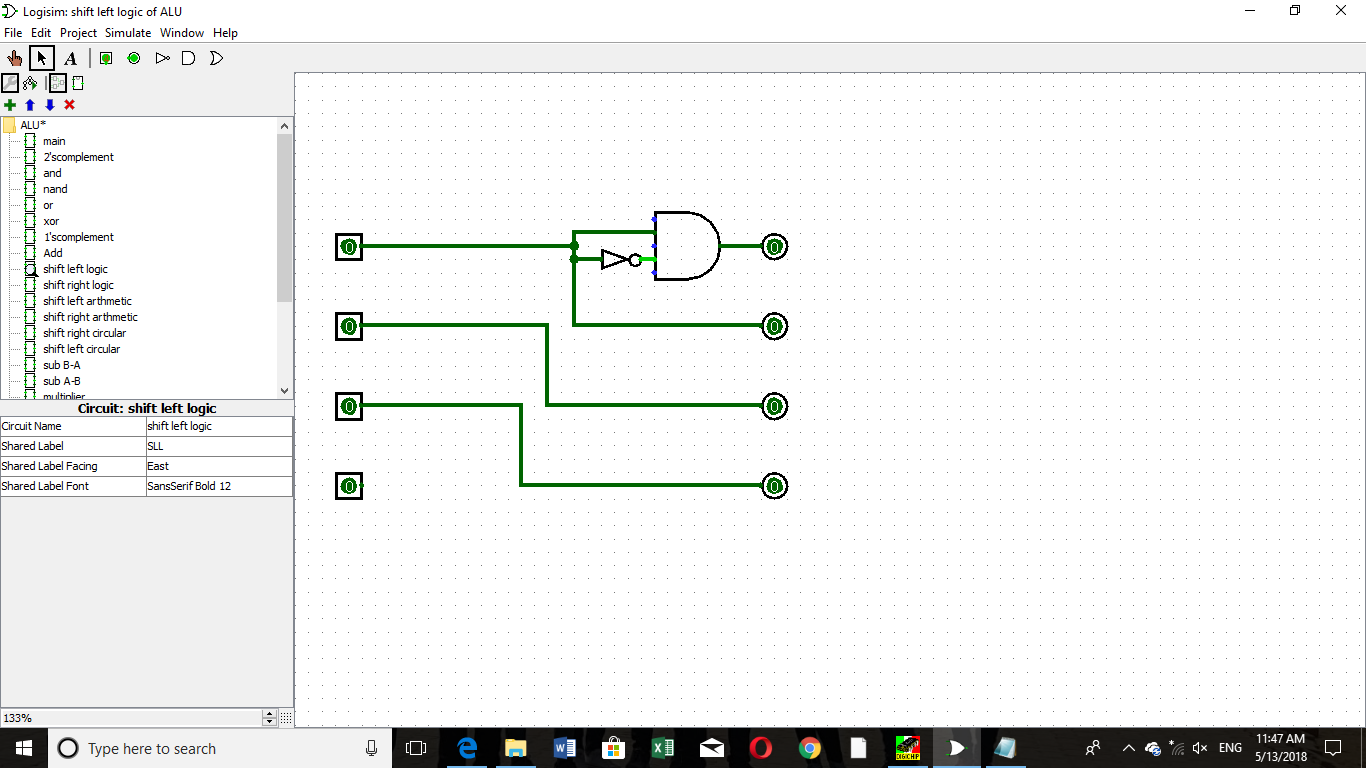
* 2’s Complement:



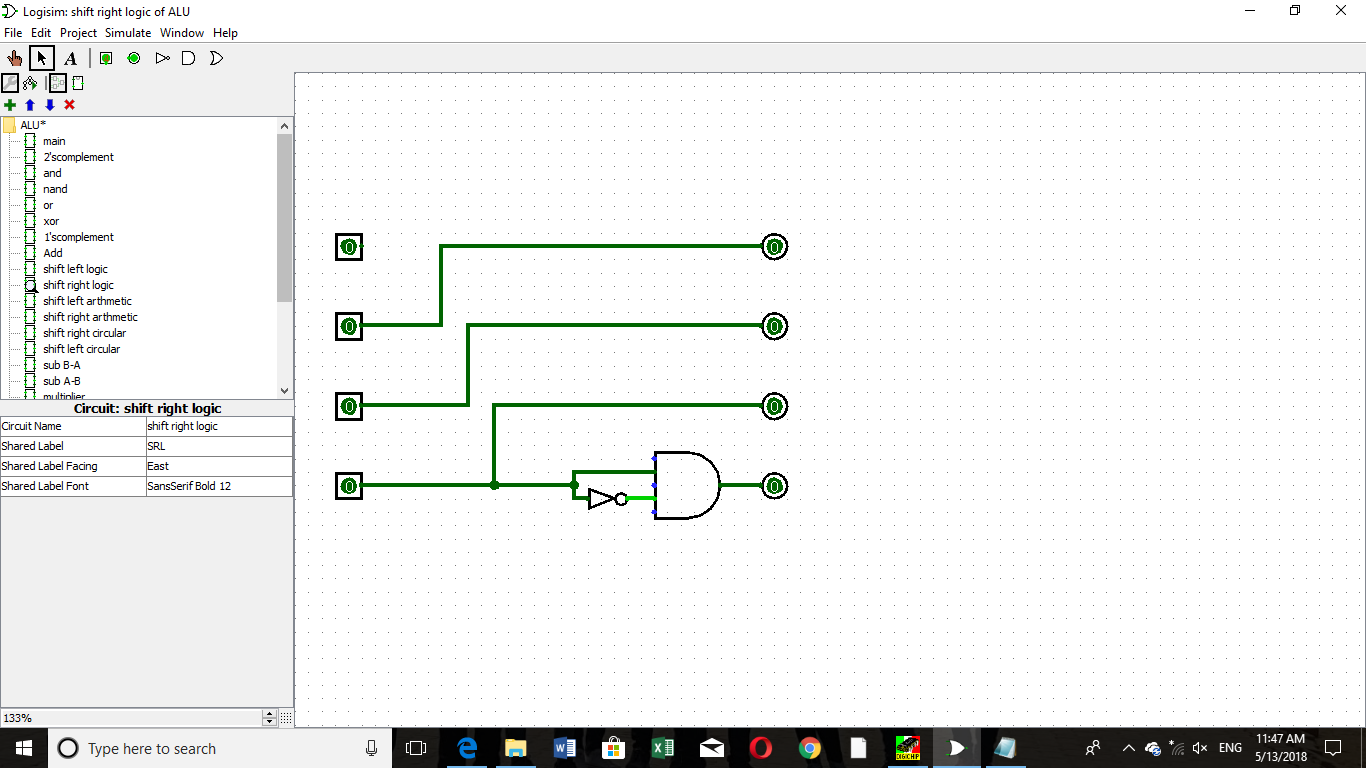
* 1’s Complement:



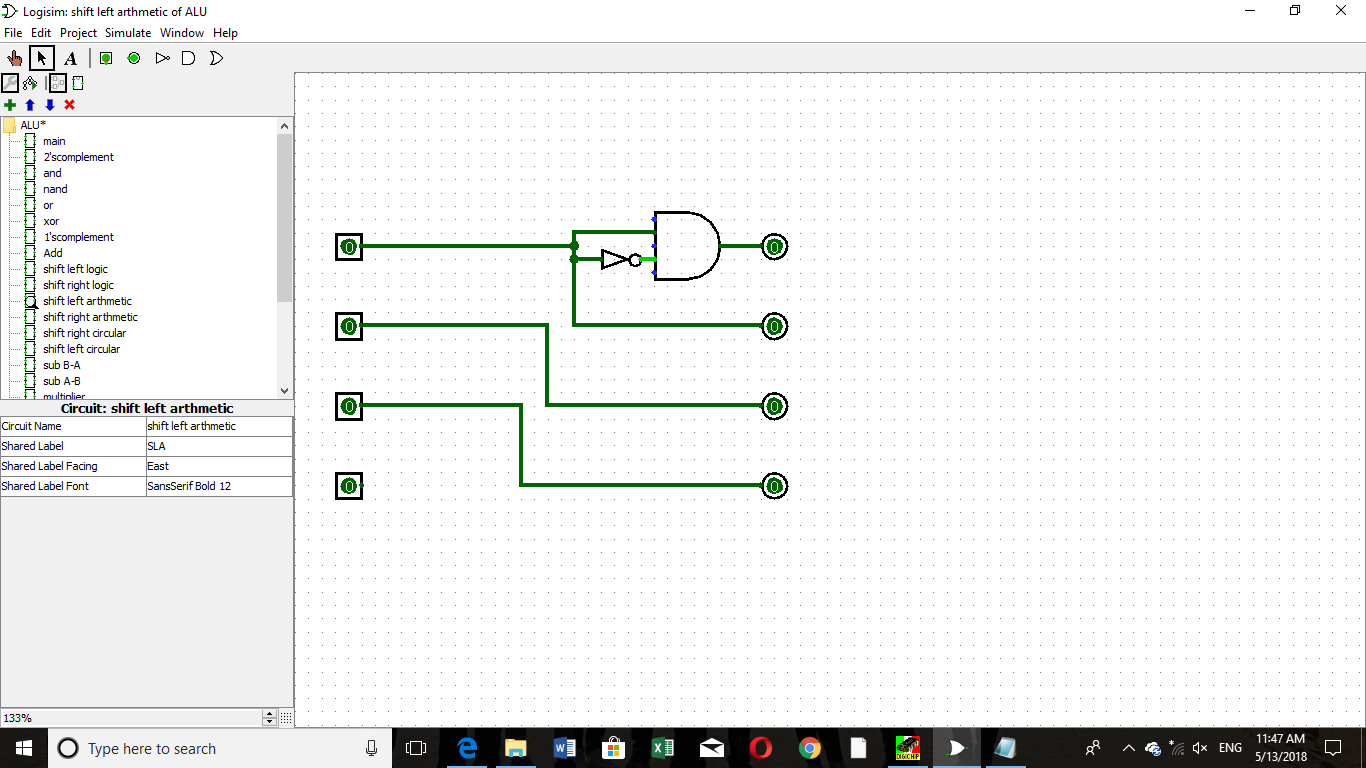
* Left Logical Shift:



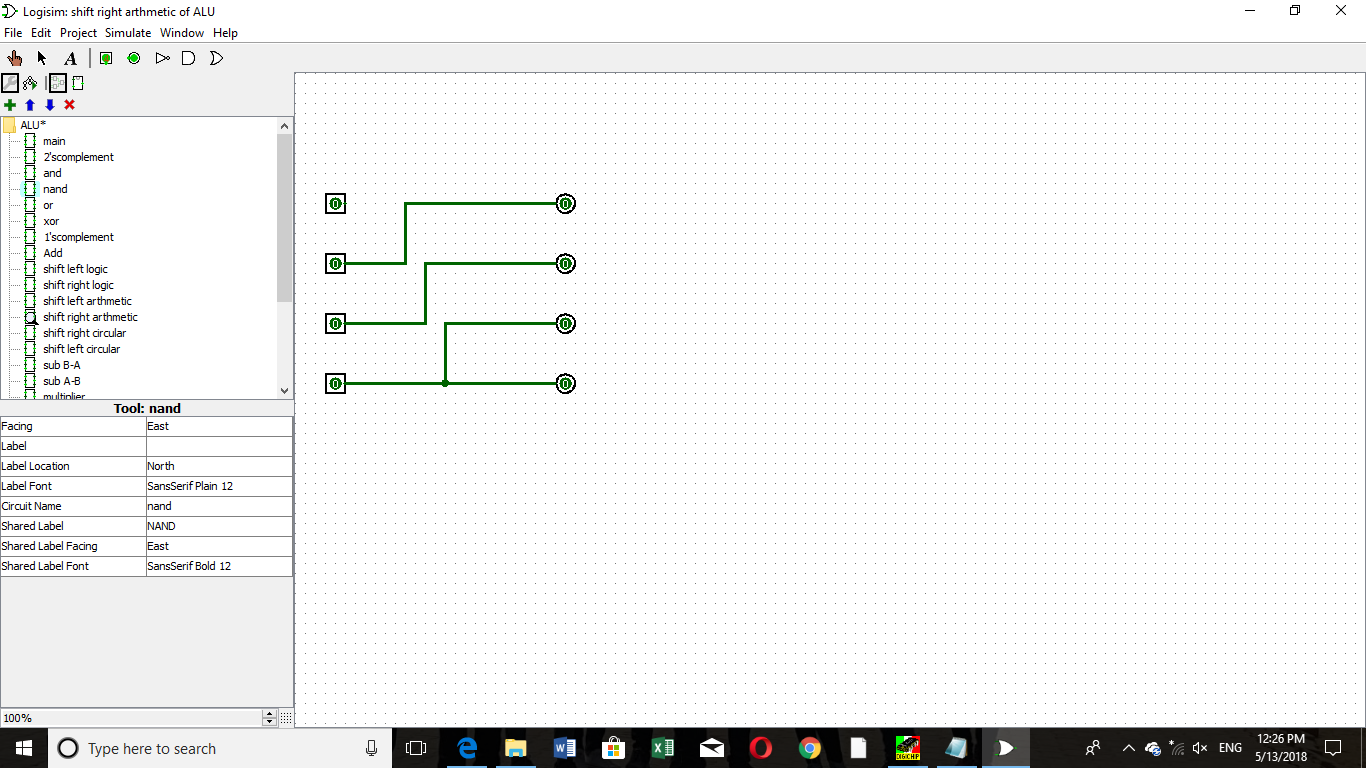
* Right Logical Shift:



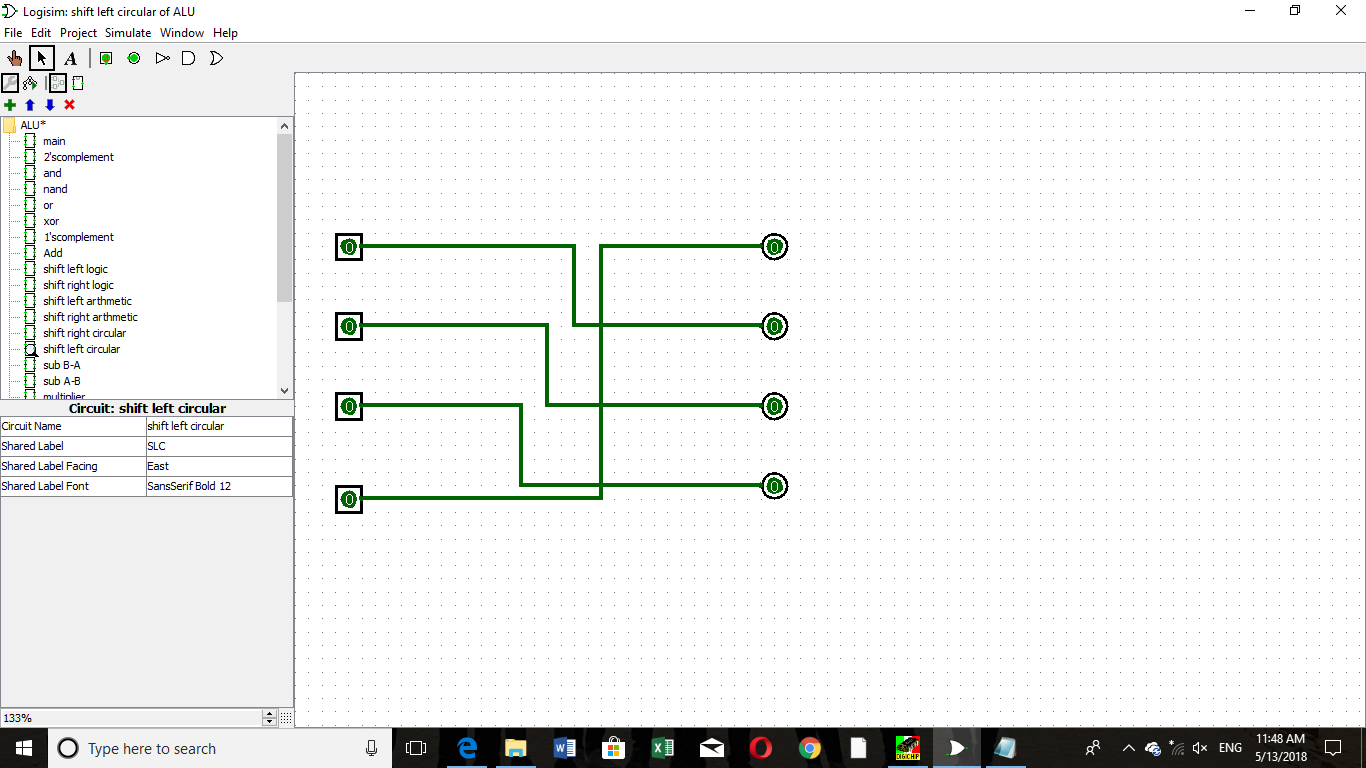
* Left Arithmetic Shift:



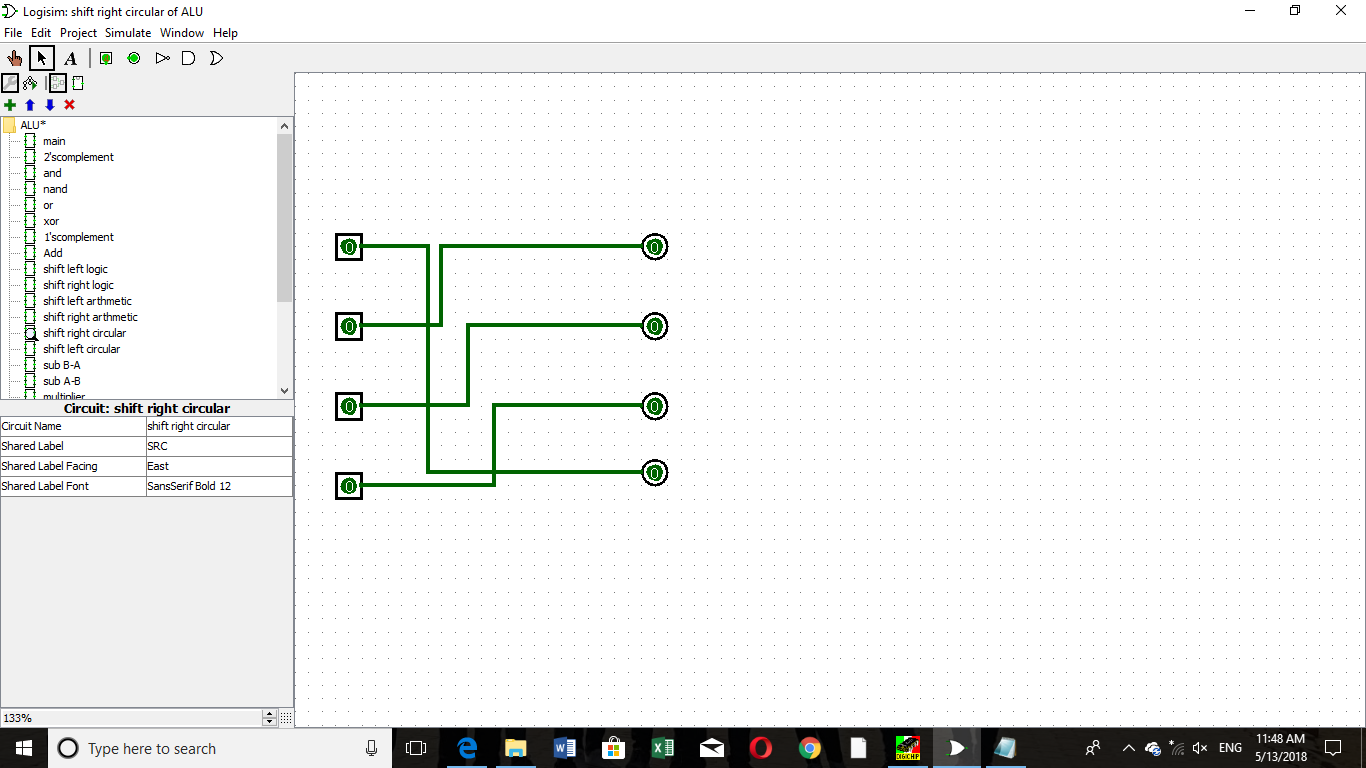
* Right Arithmetic Shift:



* Left Circular Shift:

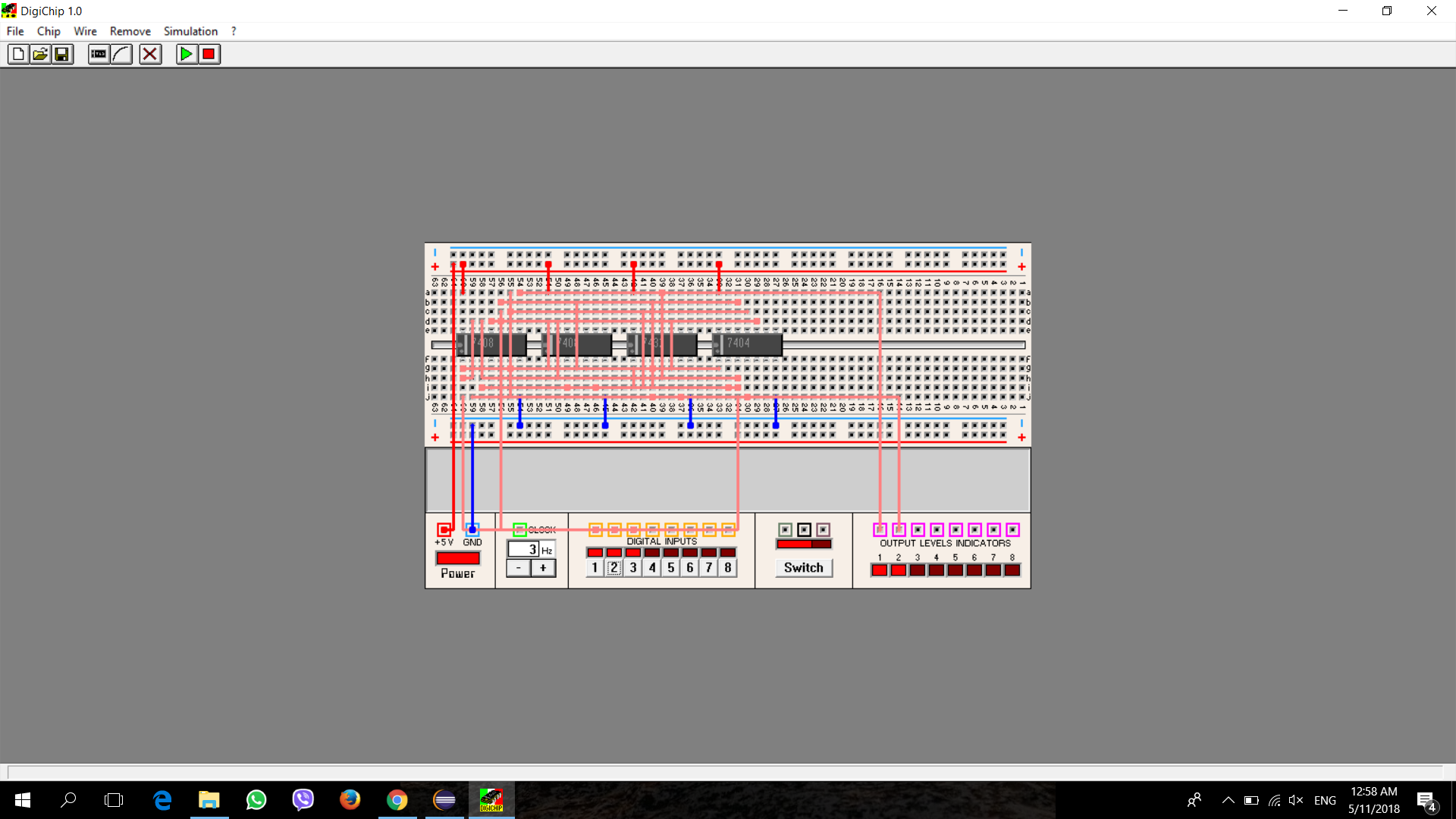


* Right Circular Shift:



1. **Chips Layout and Wiring List:**

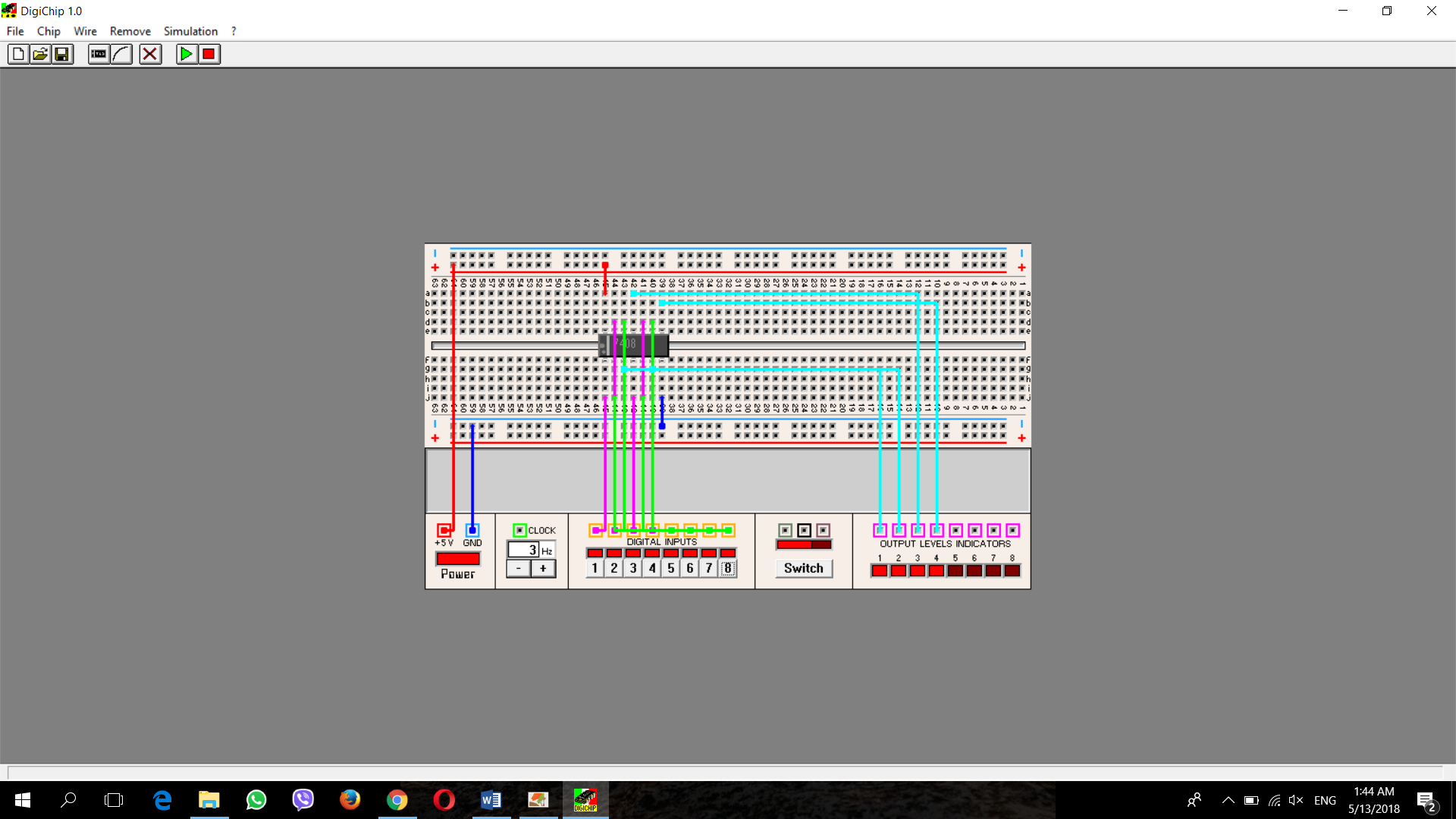
* 1-Bit Adder:



|  |  |  |
| --- | --- | --- |
| Chip | Quantity | Assignment |
| AND(7408) | 2 | A, B |
| OR (7432) | 1 | C |
| Inverter(7404) | 1 | D |

|  |  |
| --- | --- |
| **1st point** | **2nd point** |
| A14 | VCC |
| B14 | VCC |
| C14 | VCC |
| D14 | VCC |
| A7 | GND |
| B7 | GND |
| C7 | GND |
| D7 | GND |
| Input 1 | A1 |
| Input 1 | D1 |
| Input 1 | A13 |
| Input 2 | D3 |
| Input 2 | A5 |
| Input 2 | A12 |
| A3 | C1 |
| A6 | C2 |
| A11 | C5 |
| D4 | A2 |
| D2 | A4 |
| C3 | A9 |
| C3 | D11 |
| C3 | B5 |
| Input 3 | A10 |
| Input 3 | D13 |
| Input 3 | B1 |
| A8 | C4 |
| C6 | Output 2 |
| D10 | B2 |
| D12 | B4 |
| B3 | C13 |
| B6 | C12 |
| C11 | Output 1 |

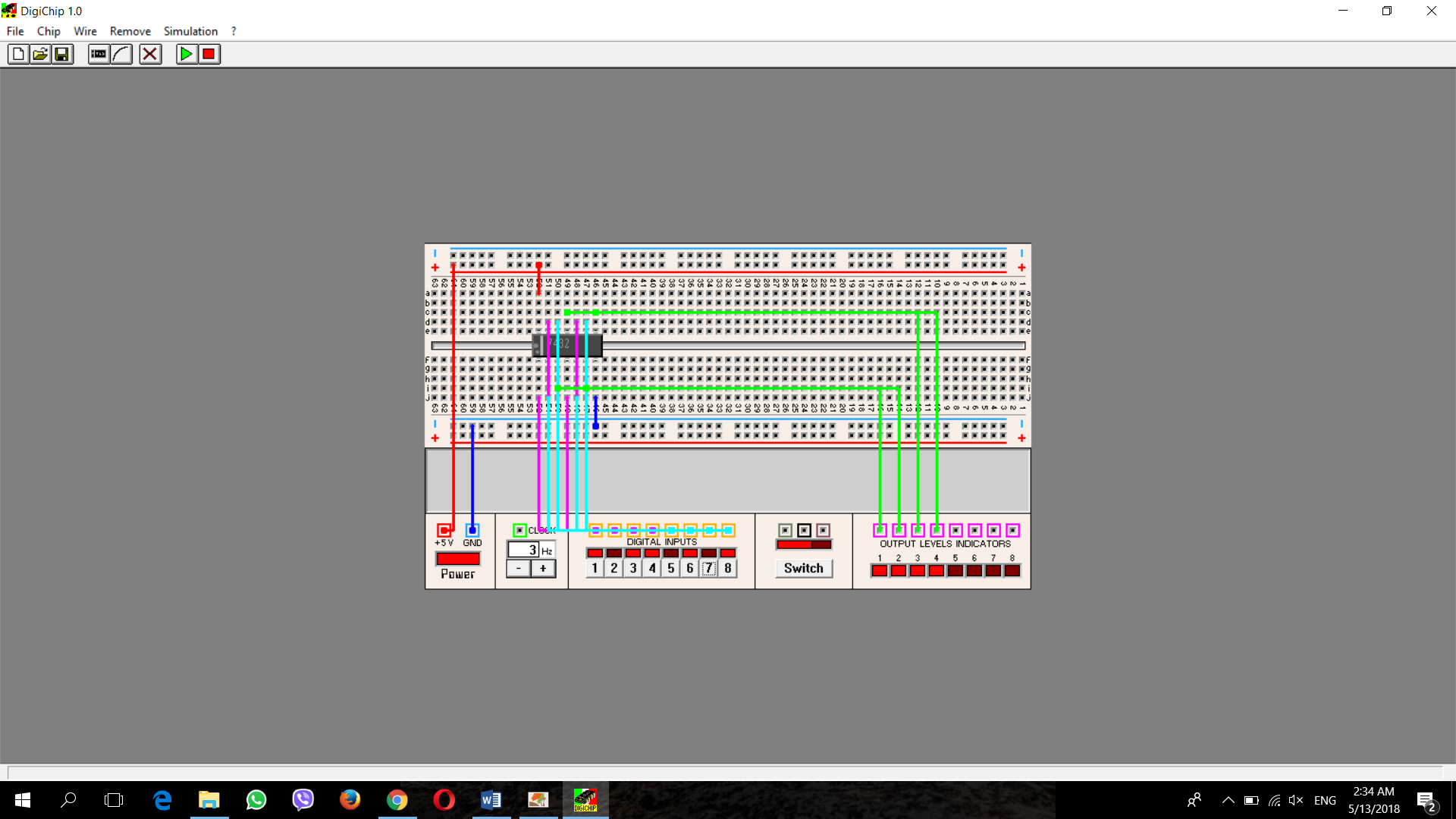
* Bitwise AND:



|  |  |  |
| --- | --- | --- |
| Chip | Quantity | Assignment |
| AND(7408) | 1 | A |

|  |  |
| --- | --- |
| **1st point** | **2nd point** |
| A14 | VCC |
| A7 | GND |
| Input 1 | A1 |
| Input 2 | A4 |
| Input 3 | A13 |
| Input 4 | A10 |
| Input 5 | A2 |
| Input 6 | A5 |
| Input 7 | A12 |
| Input 8 | A9 |
| A3 | Output 1 |
| A6 | Output 2 |
| A11 | Output 3 |
| A8 | Output 4 |

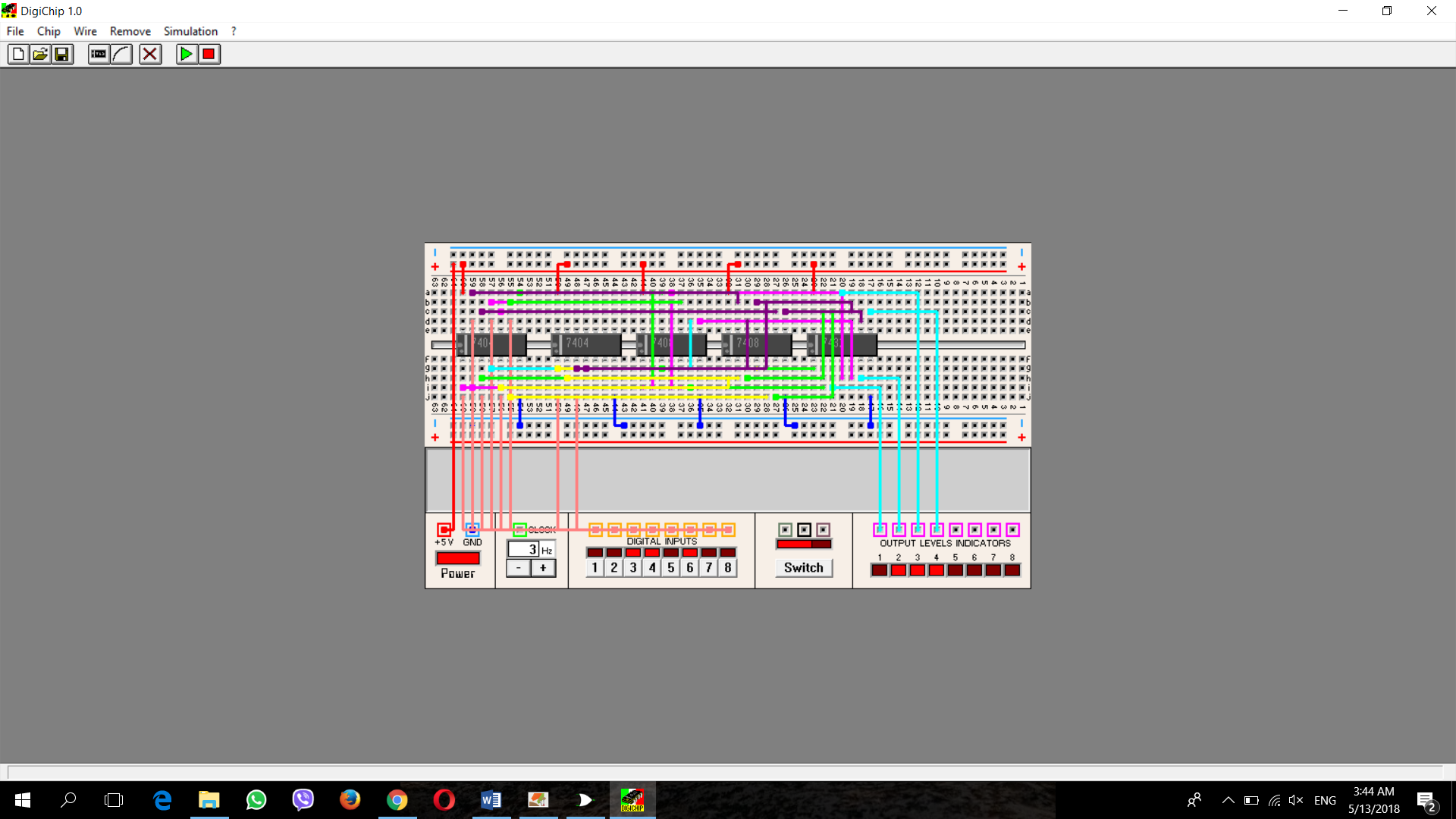
* Bitwise OR:



|  |  |  |
| --- | --- | --- |
| Chip | Quantity | Assignment |
| OR(7432) | 1 | A |

|  |  |
| --- | --- |
| **1st point** | **2nd point** |
| A14 | VCC |
| A7 | GND |
| Input 1 | A1 |
| Input 2 | A4 |
| Input 3 | A13 |
| Input 4 | A10 |
| Input 5 | A2 |
| Input 6 | A5 |
| Input 7 | A12 |
| Input 8 | A9 |
| A3 | Output 1 |
| A6 | Output 2 |
| A11 | Output 3 |
| A8 | Output 4 |

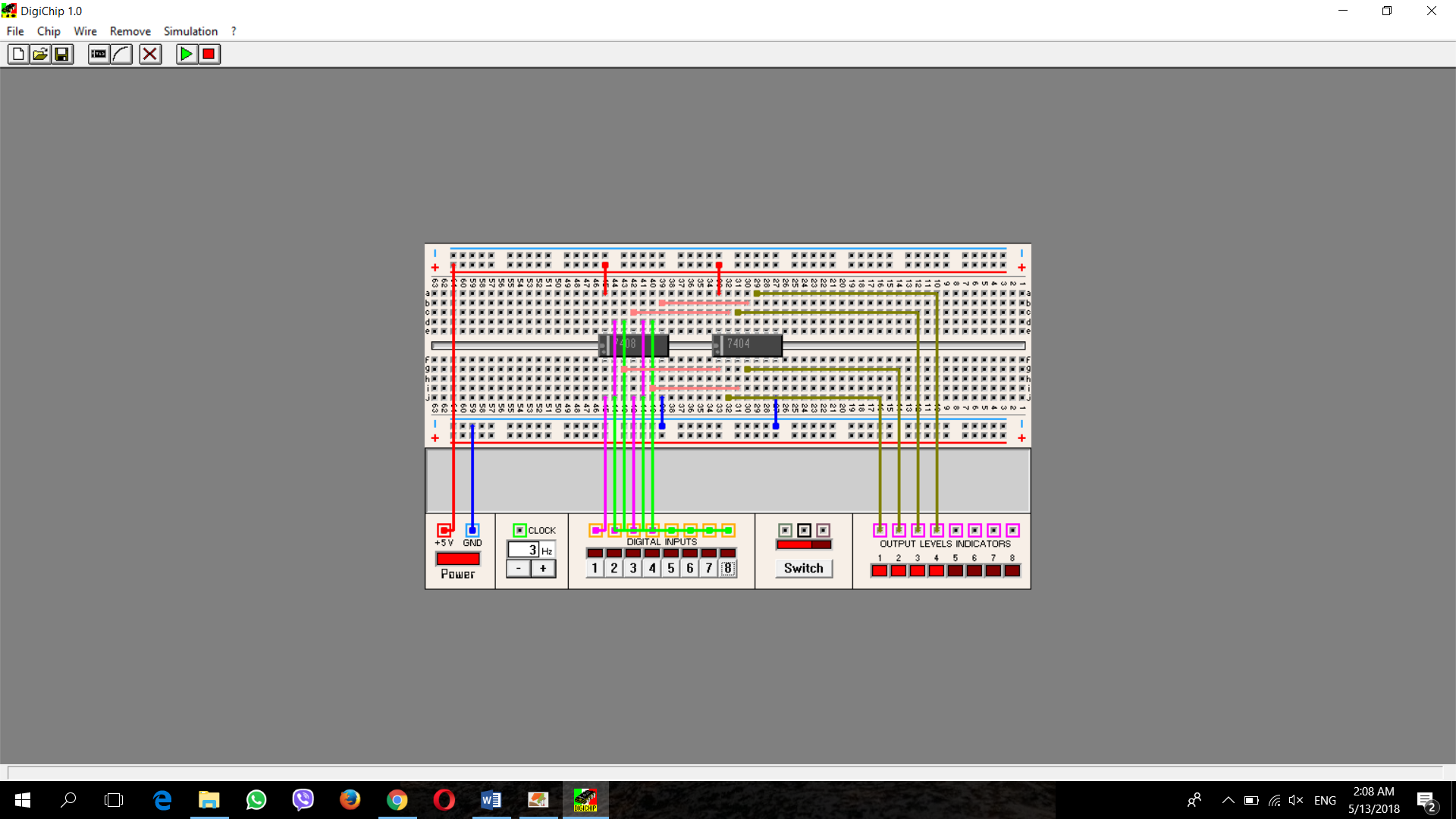
* Bitwise XOR:



|  |  |  |
| --- | --- | --- |
| Chip | Quantity | Assignment |
| Inverter(7404) | 2 | A, B |
| AND(7408) | 2 | C, D |
| OR(7432) | 1 | E |

|  |  |
| --- | --- |
| **1st point** | **2nd point** |
| A14 | VCC |
| B14 | VCC |
| C14 | VCC |
| D14 | VCC |
| E14 | VCC |
| A7 | GND |
| B7 | GND |
| C7 | GND |
| D7 | GND |
| E7 | GND |
| Input 1 | A1 |
| Input 2 | A3 |
| Input 3 | A5 |
| Input 4 | A13 |
| Input 5 | A11 |
| Input 6 | A9 |
| Input 7 | B1 |
| Input 8 | B2 |
| Input 1 | C1 |
| A10 | C2 |
| Input 5 | C4 |
| A2 | C5 |
| C3 | E1 |
| C6 | E2 |
| E3 | Output 1 |
| Input 2 | C13 |
| A8 | C12 |
| Input 6 | C10 |
| A4 | C9 |
| C11 | E4 |
| C8 | E5 |
| E6 | Output 2 |
| Input 3 | D1 |
| B2 | D2 |
| Input 7 | D4 |
| A6 | D5 |
| D3 | E13 |
| D6 | E12 |
| E6 | Output 3 |
| Input 4 | D13 |
| B4 | D12 |
| Input 8 | D10 |
| A12 | D9 |
| D11 | E10 |
| D8 | E9 |
| E8 | Output 3 |

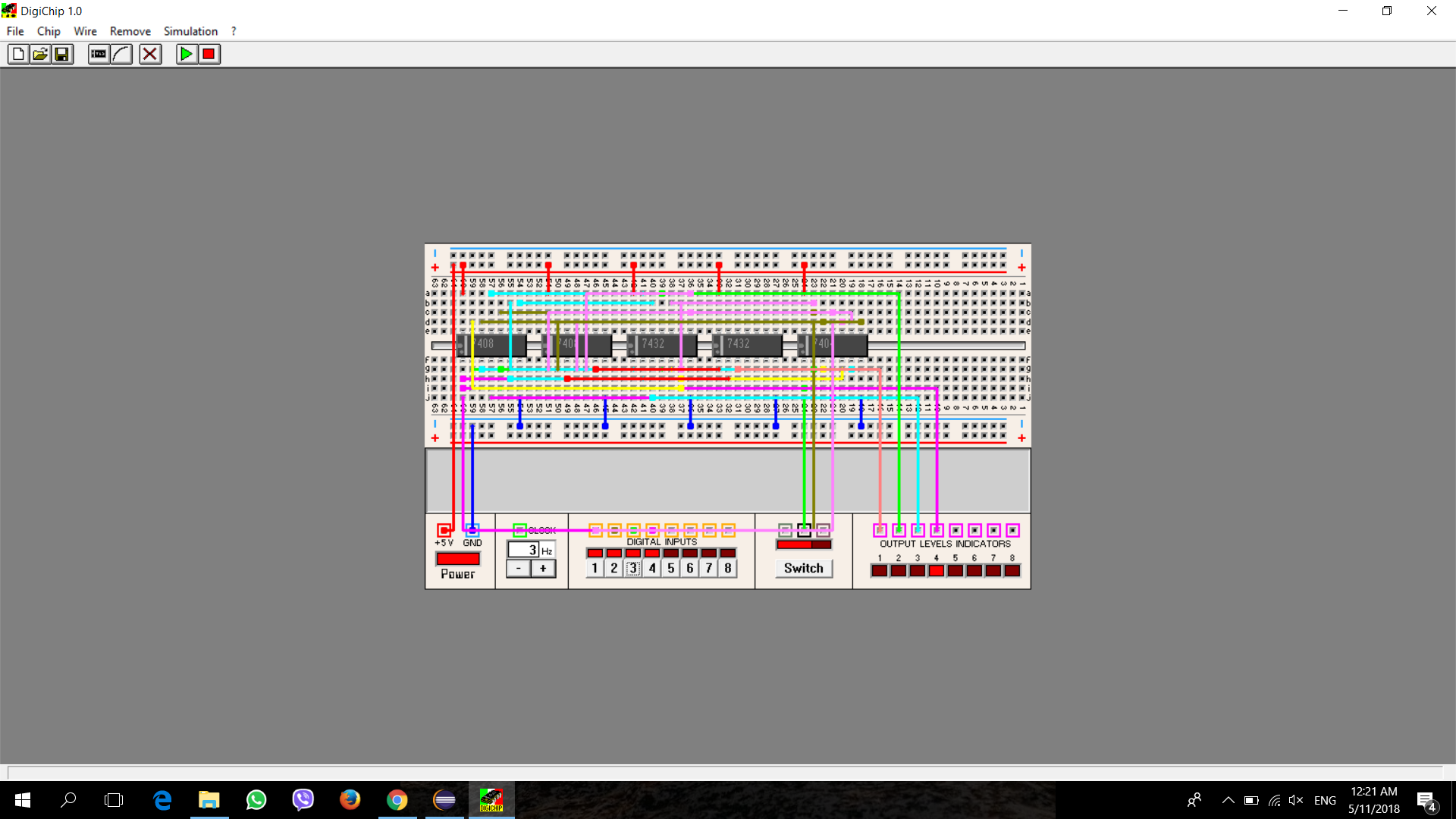
* Bitwise NAND:



|  |  |  |
| --- | --- | --- |
| Chip | Quantity | Assignment |
| AND(7408) | 1 | A |
| Invertor(7404) | 1 | B |

|  |  |
| --- | --- |
| **1st point** | **2nd point** |
| A14 | VCC |
| B14 | VCC |
| A7 | GND |
| B7 | GND |
| Input 1 | A1 |
| Input 2 | A4 |
| Input 3 | A13 |
| Input 4 | A10 |
| Input 5 | A2 |
| Input 6 | A5 |
| Input 7 | A12 |
| Input 8 | A9 |
| A3 | B1 |
| A6 | B3 |
| A11 | B13 |
| A8 | B11 |
| B2 | Output 1 |
| B4 | Output 2 |
| B12 | Output 3 |
| B10 | Output 4 |

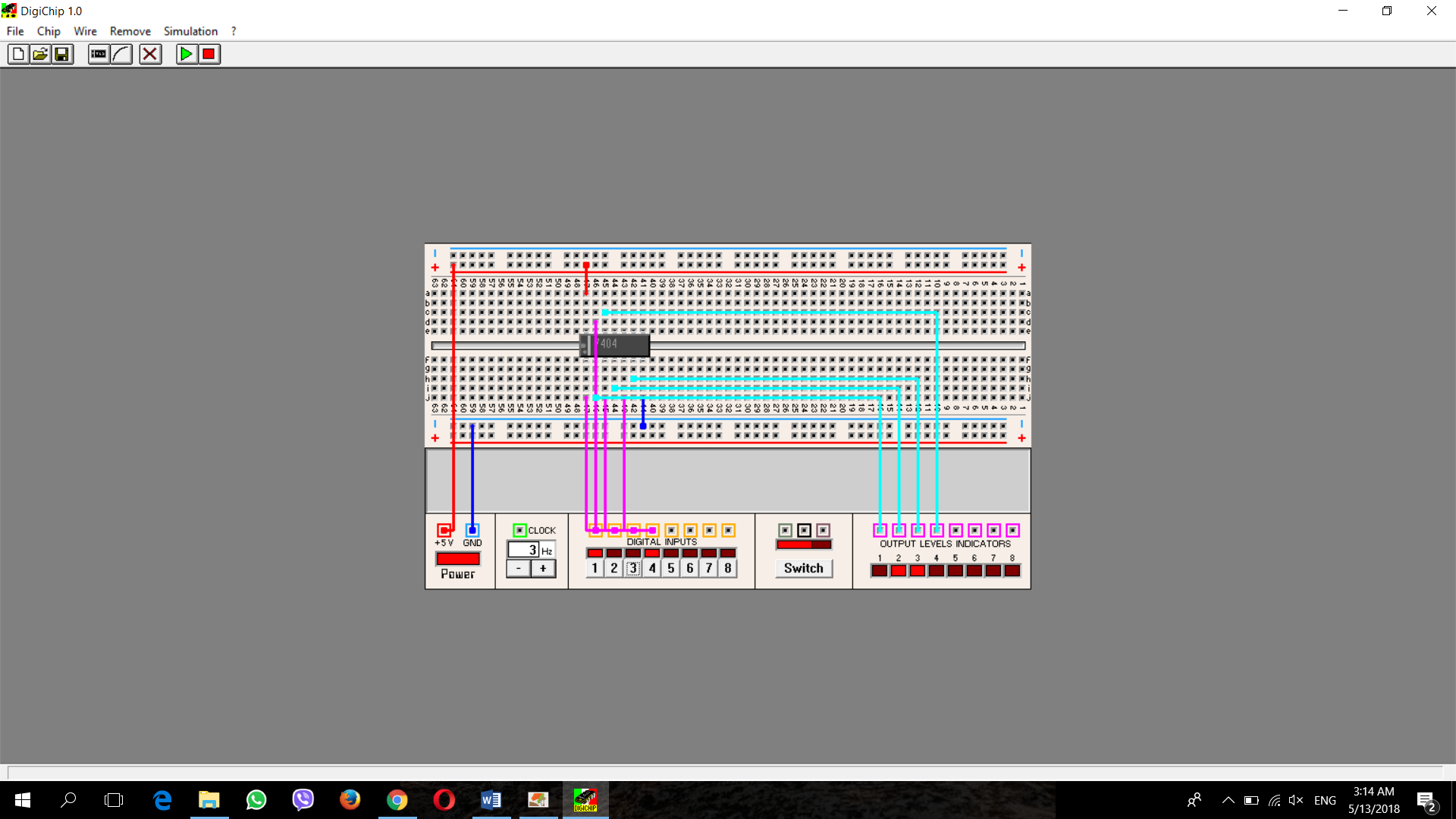
* 2’s Complement:



|  |  |  |
| --- | --- | --- |
| Chip | Quantity | Assignment |
| AND(7408) | 2 | A, B |
| OR (7432) | 2 | C, D |
| Inverter(7404) | 1 | E |

|  |  |
| --- | --- |
| **1st point** | **2nd point** |
| A14 | VCC |
| B14 | VCC |
| C14 | VCC |
| D14 | VCC |
| E14 | VCC |
| A7 | GND |
| B7 | GND |
| C7 | GND |
| D7 | GND |
| E7 | GND |
| Input 4 | Output 4 |
| Input 4 | A1 |
| Input 4 | E3 |
| Input 4 | C5 |
| Input 3 | E1 |
| Input 3 | A5 |
| Input 3 | C4 |
| E2 | A2 |
| E4 | A4 |
| A3 | C1 |
| A6 | C2 |
| C3 | Output 3 |
| Input 2 | E12 |
| Input 2 | C10 |
| Input 2 | A10 |
| E13 | A12 |
| C6 | E5 |
| E6 | A9 |
| A8 | C12 |
| A11 | C13 |
| C11 | Output 2 |
| Input 1 | E11 |
| Input 1 | B1 |
| C6 | C9 |
| C8 | B5 |
| C8 | E9 |
| E8 | B2 |
| B3 | D1 |
| B6 | D2 |
| B4 | E10 |
| D3 | Output 1 |

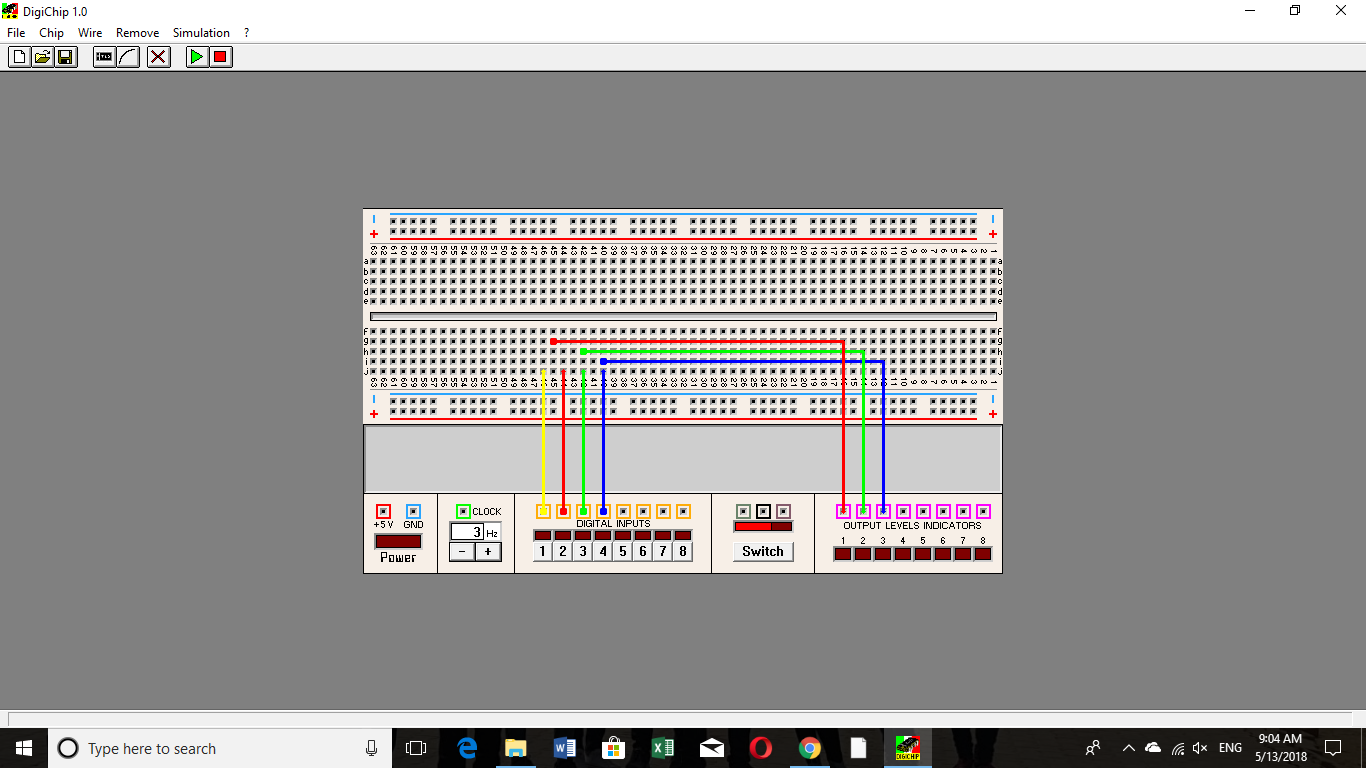
* 1’s Complement:



|  |  |  |
| --- | --- | --- |
| Chip | Quantity | Assignment |
| Inverter(7404) | 1 | A |

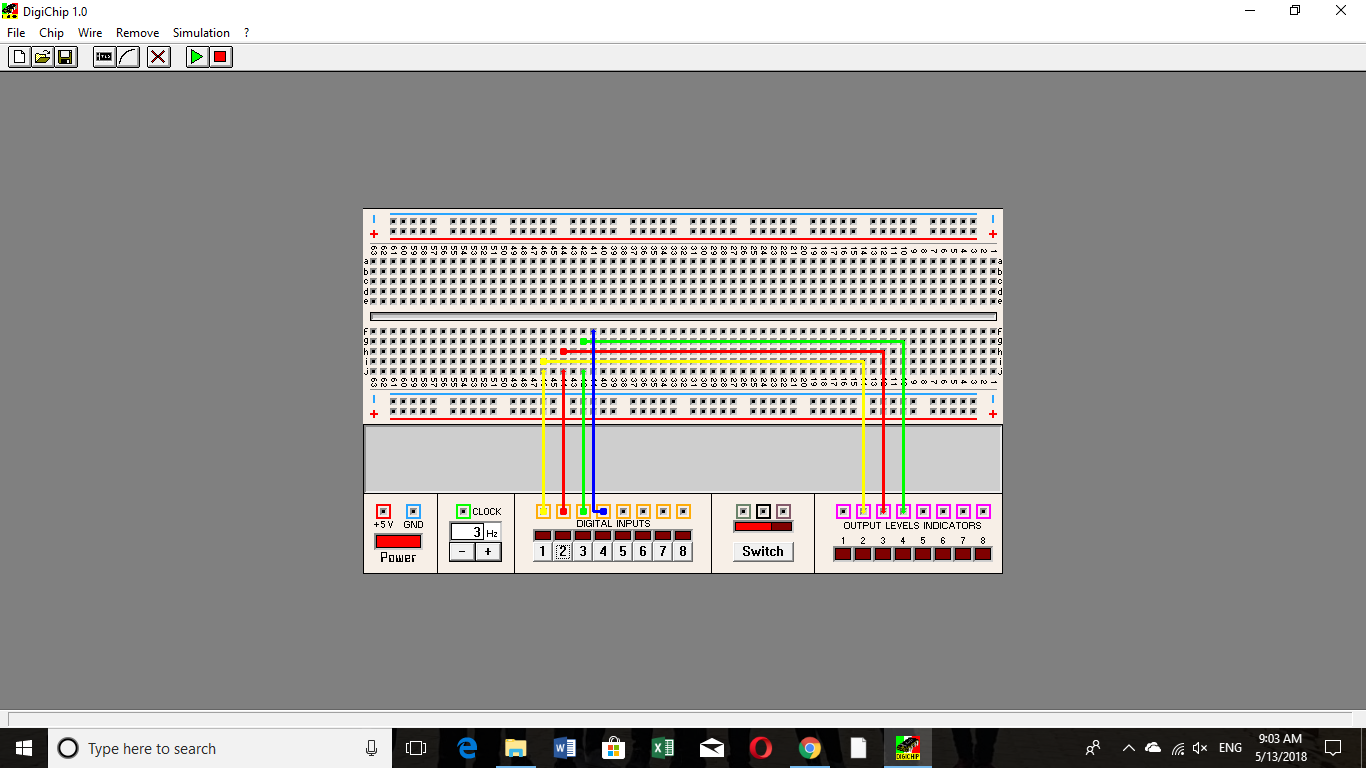
|  |  |
| --- | --- |
| **1st point** | **2nd point** |
| A14 | VCC |
| A7 | GND |
| Input 1 | A1 |
| Input 2 | A3 |
| Input 3 | A5 |
| Input 4 | A13 |
| A2 | Output 1 |
| A4 | Output 2 |
| A6 | Output 3 |
| A12 | Output 4 |

* Left Logical Shift:



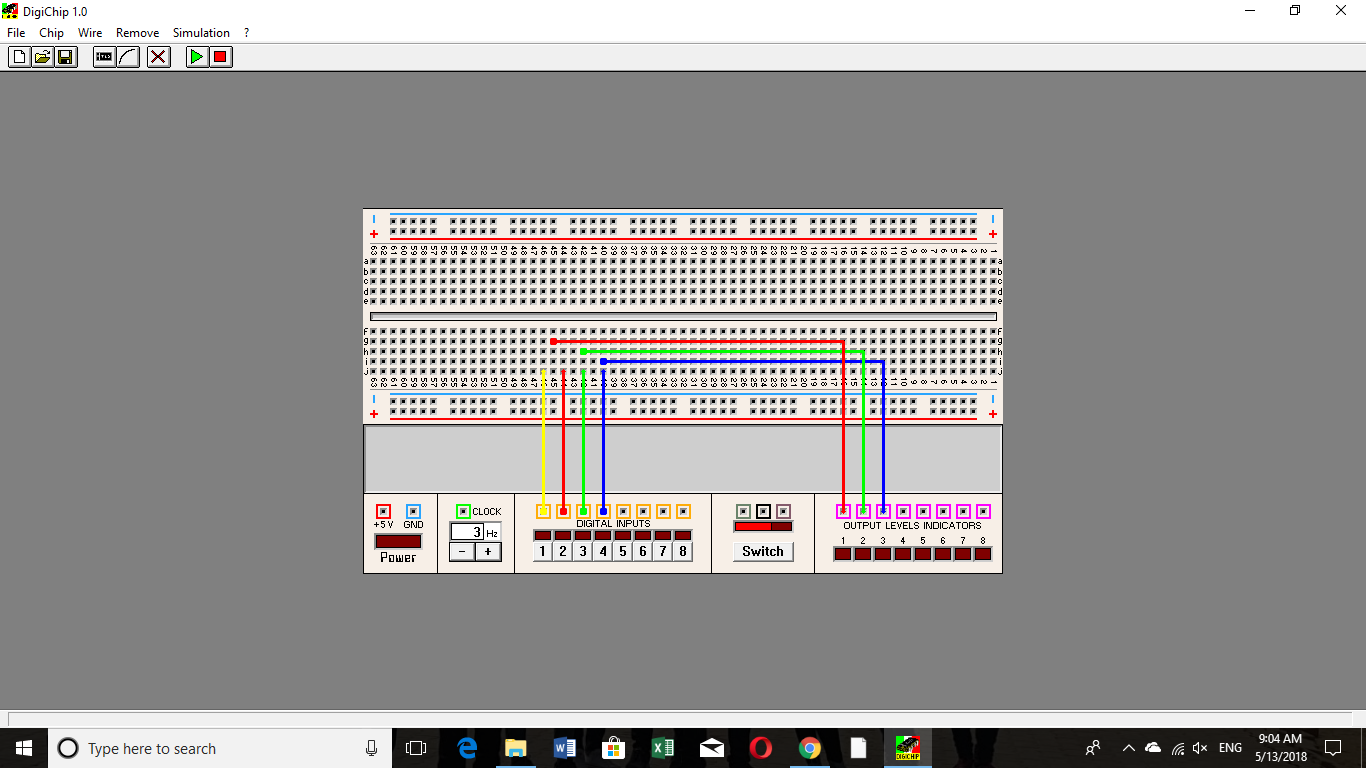
|  |  |
| --- | --- |
| **1st point** | **2nd point** |
| Input 2 | Output 1 |
| Input 3 | Output 2 |
| Input 4 | Output 3 |

* Right Logical Shift:



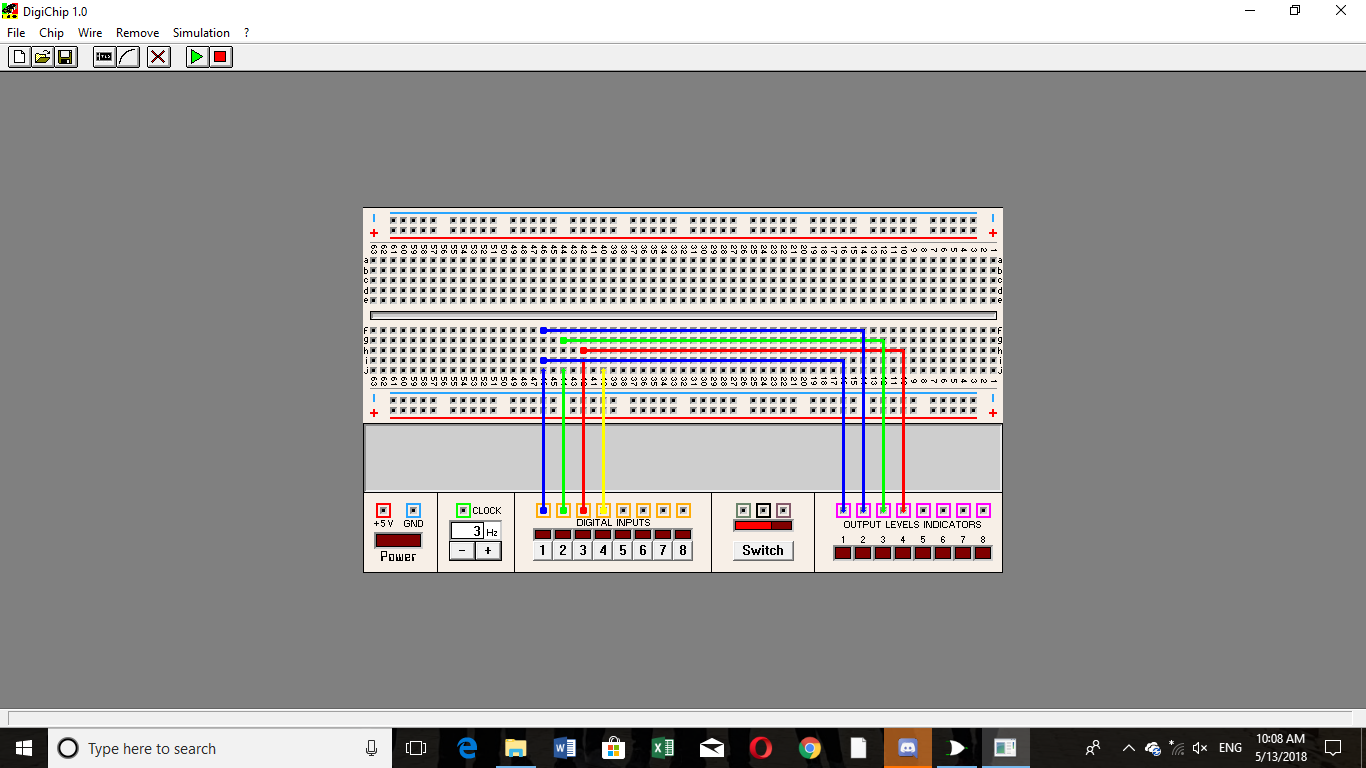
|  |  |
| --- | --- |
| **1st point** | **2nd point** |
| Input 1 | Output 2 |
| Input 2 | Output 3 |
| Input 3 | Output 4 |

* Left Arithmetic Shift:



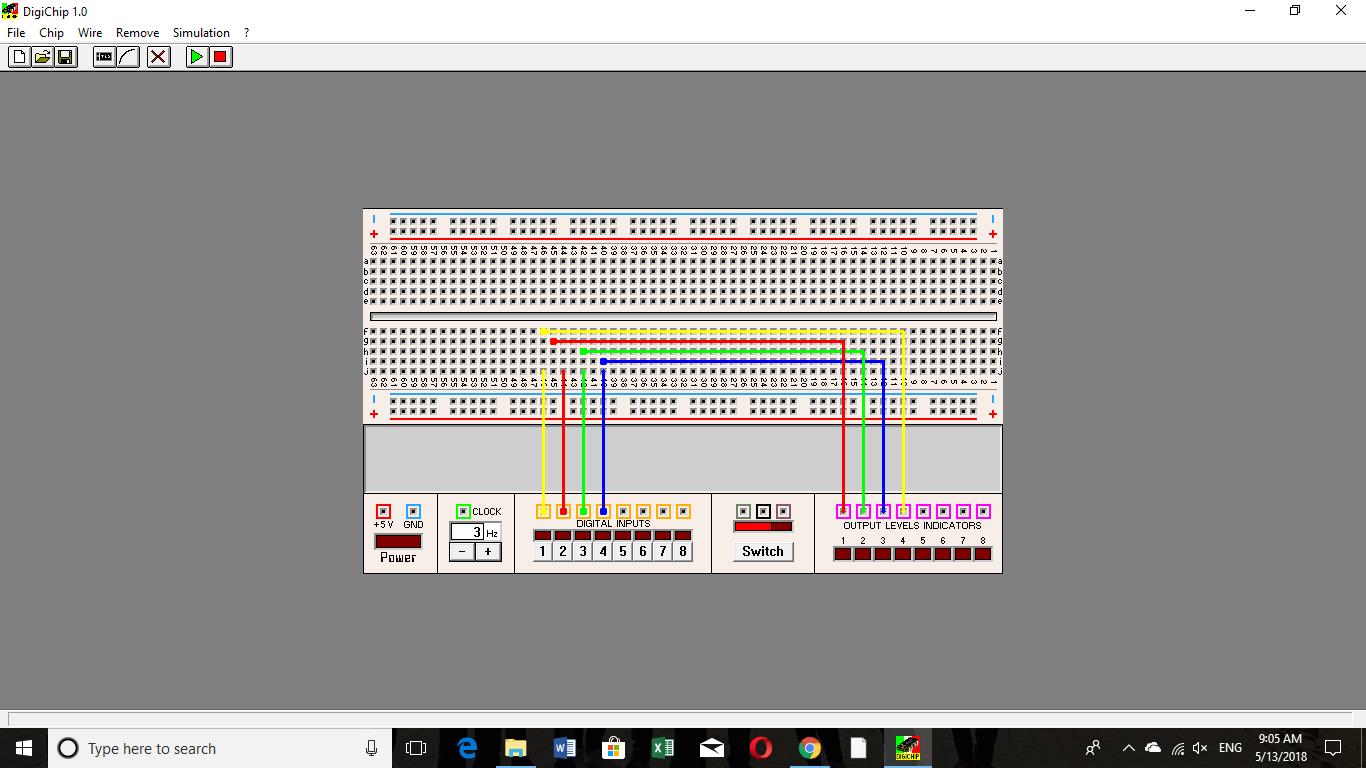
|  |  |
| --- | --- |
| **1st point** | **2nd point** |
| Input 2 | Output 1 |
| Input 3 | Output 2 |
| Input 4 | Output 3 |

* Right Arithmetic Shift:



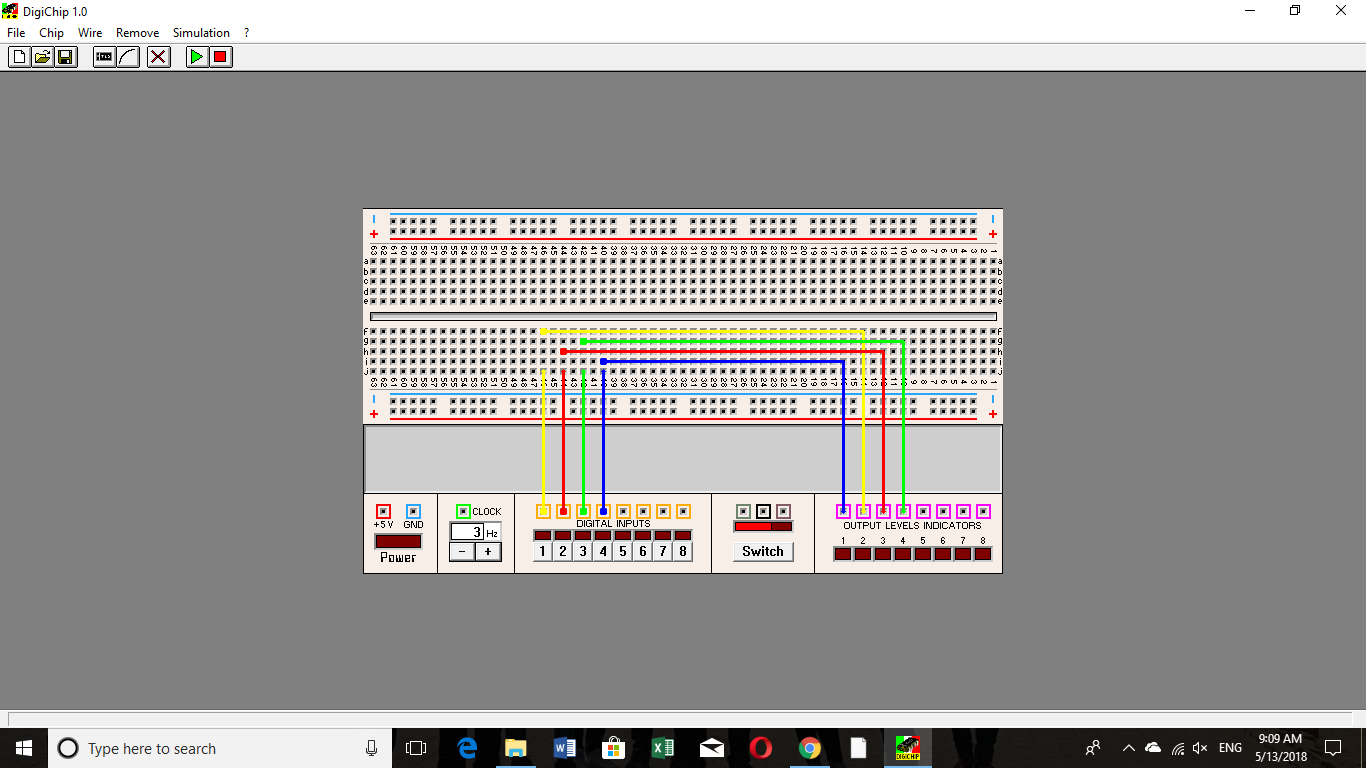
|  |  |
| --- | --- |
| **1st point** | **2nd point** |
| Input 1 | Output 1 |
| Input 1 | Output 2 |
| Input 2 | Output 3 |
| Input 3 | Output 4 |

* Left Circular Shift:



|  |  |
| --- | --- |
| **1st point** | **2nd point** |
| Input 1 | Output 4 |
| Input 2 | Output 1 |
| Input 3 | Output 2 |
| Input 4 | Output 3 |

* Right Circular Shift:



|  |  |
| --- | --- |
| **1st point** | **2nd point** |
| Input 1 | Output 2 |
| Input 2 | Output 3 |
| Input 3 | Output 4 |
| Input 4 | Output 1 |