**Name: laiba asif**

**Student Number: R00201303**

**Class Group: COMP1DY**

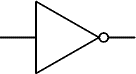
Download and install Logisim: <http://www.cburch.com/logisim/>

**Lab 4 – Basic Logic Gates**

**1. NOT GATE**

A NOT gate gives the opposite value of the input, i.e. a 0 becomes a 1, and a 1 becomes a 0.

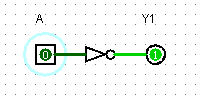
Symbol for NOT =



i) Complete the truth table for an NOT gate:

|  |  |
| --- | --- |
| **A** |  |
| 0 | 1 |
| 1 | 0 |

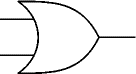
ii) Implement a simple NOT gate using Logisim. Copy and paste the diagram below:



**2. OR GATE**

When using an OR gate, the output is 1 if either/both inputs are 1.

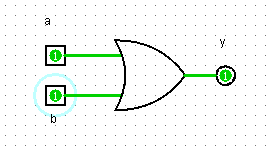
Symbol for OR =



i) Complete the truth table for an OR gate

|  |  |  |
| --- | --- | --- |
| **A** | **B** | **A+B** |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

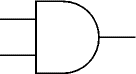
ii) Implement a simple OR gate using Logisim. Copy and paste the diagram below:



**3. AND GATE**

When ANDing 2 bits, the output is 1 (i.e. on / high / true) only when both inputs are 1.

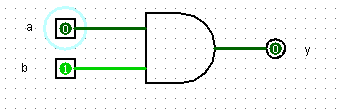
Symbol for AND =



i) Complete the truth table for an AND gate:

|  |  |  |
| --- | --- | --- |
| **A** | **B** | **A.B** |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

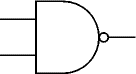
ii) Implement a simple AND gate using Logisim. Copy and paste the diagram below:



**4. NAND GATE**

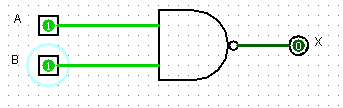
A NAND gate returns the opposite value to an AND gate. When NANDing 2 bits, the output is 0 only when both inputs are 1.

Symbol for NAND =



i) Complete the truth table for an NAND gate:

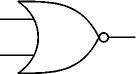
|  |  |  |
| --- | --- | --- |
| **A** | **B** |  |
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

ii) Implement a simple NAND gate using Logisim. Copy and paste the diagram below:

**5. NOR GATE**

A NOR gate returns the opposite value to an OR gate. When using a NOR gate, the output is 0 if either/both inputs are 1.

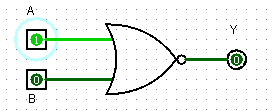
Symbol for NOR =



i) Complete the truth table for a NOR gate:

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

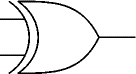
ii) Implement a simple NOR gate using Logisim. Copy and paste the diagram below:



**6. EXOR GATE**

EXOR means exclusive OR. The output is 1 only if either input is 1, excluding the case when both are 1.

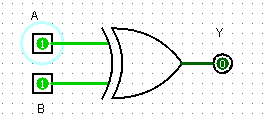
Symbol for EXOR =



i) Complete the truth table for an EXOR gate:

|  |  |  |
| --- | --- | --- |
| **A** | **B** | **A ⊕ B** |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

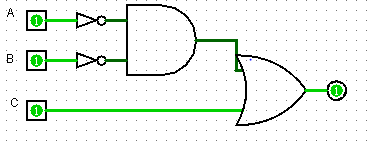
ii) Implement a simple EXOR gate using Logisim. Copy and paste the diagram below:



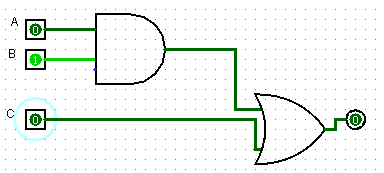
**7. Implement Boolean expressions as circuits**

Draw circuits for the Boolean expressions outlined below.

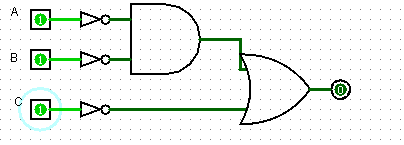
Complete the truth tables for these expressions and verify the truth tables using your circuits.



|  |  |  |  |
| --- | --- | --- | --- |
| A | B | C |  |
| 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 1 |
| 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 1 | 1 | 0 |



|  |  |  |  |
| --- | --- | --- | --- |
| A | B | C |  |
| 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 1 |
| 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 |



|  |  |  |  |
| --- | --- | --- | --- |
| A | B | C |  |
| 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 1 | 1 | 0 |