****

**National Universityof Computer and Emerging sciences**

**OBJECTIVES:**

* To learn and understand how to design a multiple output combinational circuit
* To learn and understand the working of 2-bit binary comparator
* Adder/Subtractor

**APPARATUS:**Logic trainer, Logic probe

**COMPONENTS:**ICs 74LS08, 74LS32, 74LS04, 74LS86, 74LS02

**THEORY:**

Binarycomparator is a combinational circuit that compares magnitude of two binary data signals A & Band generates the results of comparison in the form of three outputsignals A>B, A=B, A<B.

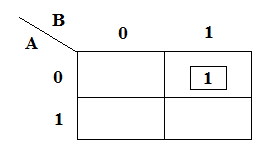
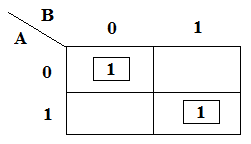
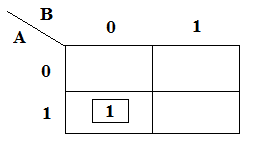
**One-bit comparator:**

One-bit comparator compares magnitude of two numbers A and B, 1 bit each, and generates the comparison result. The result consists of three outputs let us say L, E, G, so that

**Truth Table:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Inputs** | | **Outputs** | | |
| **A** | **B** | **L** | **E** | **G** |
| 0 | 0 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 | 0 |

**K-Maps for Outputs:**

K-Map for Output **L**K-Map for Output **E**K-Map for Output **G**

**Boolean Expressions of Outputs:**

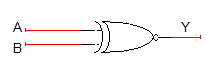
**L:**

**E:**

**G:**

**Exclusive-OR & Exclusive-NOR gates:**

The figure given below shows the symbol of Exclusive-OR (XOR) and Exclusive-NOR (XNOR) gates.

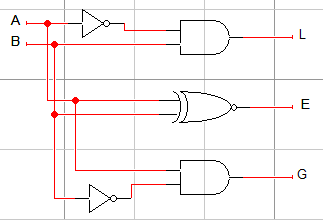


XNOR gate XOR gate

Boolean expression of XNOR gate isand Boolean expression of XOR is. Boolean expression of XNOR gate can be implemented using XOR gate as shown in figure below:



**Circuit Diagram for one-bit comparator:**

****

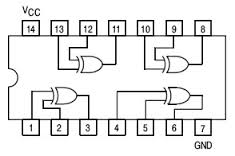
In this experiment 74LS86 IC will be used for implementation of XOR gate function. 74LS86 IC contains four 2-input XOR gates. The function table and connection diagram for this IC are shown below:

**Function Table:**

|  |  |  |
| --- | --- | --- |
| **Inputs** | | **Output** |
| **A** | **B** | **Y** |
| L | L | L |
| L | H | H |
| H | L | H |
| H | H | L |

H= Logic High, L= Logic Low

**Connection Diagram:**



**Adder and subtractor**

**Half Adder:**

Half adder is a logic circuit that performs binary addition of two 1-bit numbers. It generates two outputs namely ‘**Sum**’ and ‘**Carry**’.

**Truth Table:**

|  |  |  |  |
| --- | --- | --- | --- |
| **A** | **B** | **Carry** | **Sum** |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 |

**Boolean Expressions of Outputs:**

**Half Subtractor:**

Half subtractor is a logic circuit that performs binary subtraction of two 1-bit numbers. It generates two outputs namely ‘**Difference**’ and ‘**Borrow**’.

**Truth Table:**

|  |  |  |  |
| --- | --- | --- | --- |
| **A** | **B** | **Borrow** | **Difference** |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 0 |

**Boolean Expressions of Outputs:**

**Full Adder:**

Full adder is a logic circuit that performs binary addition of two 2-bit numbers. It generates two outputs namely ‘**Sum**’ and ‘**Carry**’.

**Truth Table:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **A** | **B** | **C** | **Carry** | **Sum** |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 |

**Boolean Expressions of Outputs:**

or

**Full Subtractor:**

Full subtractor is a logic circuit that performs binary subtraction of two 2-bit numbers. It generates two outputs namely ‘**Difference**’ and ‘**Borrow**’.

**Truth Table:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **A** | **B** | **C** | **Borrow** | **Difference** |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 |

**Boolean Expressions of Outputs:**

or

**Lab Task 1:**

Design a combinational circuit and implement on logic works that compares two 2-bit numbers and generates the comparison result. The result consists of three outputs let us say L, E, G, so that

1. Write truth table
2. Find minimal SOP expressions for the outputs **L**, **E**, and **G** using K-map. Draw separate K-map for each output in the space given below.

**Lab Task 2:**

Implement full adder and full subtractor on logic trainer.

**Lab Task 3:**

Implement 4-bit even parity detectoron logic trainer