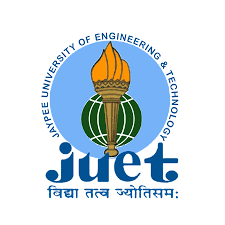
**JAYPEE UNIVERSITY OF ENGINEERING & TECHNOLOGY, GUNA**

**DEPARTMENT OF Computer science & engineering**



**A PRACTICAL WORK BOOK**

**of**

**COMPUTER ORGANIZATION & Architecture LAB**

**(18B17CI414)**

**SUBMITTED**

**TO**

**Dr Rahul pachauri**

**Name : Shourya Pratap Singh Kushwah ENRL NO :** 211B298

**Class :** Btech 3rd Year  **BATCH :** B8

**BRANCH :** CSE  **SESSION :** 2021-2025

***INDEX***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.No.** | **Aim of the experiment** | **Date of submission** | **signature** | **remarks** |
| 1 | Design of basic digital circuits using logic gates. |  |  |  |
| 2 | Design of binary adders and subtractors. |  |  |  |
| 3 | Design of basic combinational logic circuits |  |  |  |
| 4 | Design of combinational logic circuits using MUX, Encoder and seven segment displays. |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**Experiment no. # 01**

**AIM: Design of basic digital circuits using logic gates.**

**Exercise #1:** Design two inputs and five outputs All-in-One logic gate circuit shown in Fig.1 using Logisim simulator with (i) data width 1 (ii) data width 4.

(i)data width 1

|  |  |
| --- | --- |
| **Boolean Expressions** | **Logic Diagram** |
| C=~A  D=A.B  E=A+B  F= ~A+ ~B  G=AB |  |
| **Truth Table** |
|  |

(ii) data width 4.

|  |  |
| --- | --- |
| **Boolean Expressions** | **Logic Diagram** |
| C=~A  D=A.B  E=A+B  F= ~A + ~B  G=AB |  |
| **Truth Table** |
| **Not Applicable** |

**Exercise #2:** Design two inputs and one output All-in-One logic gate diagram shown in Fig.2 using Logisim simulator with (i) data width 1 (ii) data width 8.

(i) data width 1.

|  |  |
| --- | --- |
| **Boolean Expressions** | **Logic Diagram** |
| Y = ~a ~b d + ~a b c + a ~b c + a b ~d |  |
| **Truth Table** |
|  |

(ii) data width 8.

|  |  |
| --- | --- |
| **Boolean Expressions** | **Logic Diagram** |
| Y = ~a ~b d + ~a b c + a ~b c + a b ~d |  |
| **Truth Table** |
| **Not Applicable** |

**Exercise#3:** Design a three-input majority detector combinational digital circuit using Logisim simulator which shows output equal to 1 if the input variables have more 1's than 0's, the output is 0 otherwise.

|  |  |
| --- | --- |
| **Boolean Expressions** | **Logic Diagram** |
| X = b c + a c + a b |  |
| **Truth Table** |
|  |

**Exercise #4**: Design a combinational circuit with three inputs and three outputs. When the input is 0, 1, 2, or 3, the output is one greater than the input and when the input is 4, 5, 6, or 7, the output is one less than the input. Display the input and output digits using Hex digit display with splitter.

|  |  |
| --- | --- |
| **Boolean Expressions** | **Truth Table** |
| X = b c + a c + a b  Y = ~a ~b c + ~a b ~c + a ~b ~c + a b c  Z = ~c |  |
| **Logic Diagram** | |
|  | |

**Experiment no. # 02**

**AIM: : Design of binary adders and subtractors.**

**Exercise #1:** Design half adder and half subtractor shown in Fig. 1 and Fig. 2 using logisim simulator.

(i) Half Adder

|  |  |
| --- | --- |
| **Boolean Expressions** | **Logic Diagram** |
| sum = ~a b + a ~b  carry = a b |  |
| **Truth Table** |
|  |

(ii) Half Subtractor

|  |  |
| --- | --- |
| **Boolean Expressions** | **Logic Diagram** |
| Differnce= ~a b + a ~b  Borrow = a ~b |  |
| **Truth Table** |
|  |

**Exercise #2:** Design full adder using (i) basic gates only (ii) by adding half adder as sub circuit using logisim.

(i) basic gates only

|  |  |
| --- | --- |
| **Boolean Expressions** | **Truth Table** |
| sum = ~a ~b c + ~a b ~c + a ~b ~c + a b c  carry = b c + a c + a b |  |
| **Logic Diagram** | |
|  | |

ii) by adding half adder as sub circuit using Logisim.

|  |  |
| --- | --- |
| **Boolean Expressions** | **Logic Diagram** |
| Sum = ~a ~b c + ~a b ~c + a ~b ~c + a b c  carry = b c + a c + a b |  |
| **Truth Table** |
|  |

**Exercise#3:** : Design 4-bit binary adder using one half adder and 3-full adders as shown in Fig. 5. Use half adder and full adders as sub circuits in the design. Display both the input digits; output digit and end carry digit using Hex digit display with splitter available in logisim simulator.

|  |  |
| --- | --- |
| **Boolean Expressions** | **Truth Table** |
| S0 = ~B0 A0 + B0 ~A0  S1 = ~B1 ~A1 B0 A0 + ~B1 A1 ~B0 + ~B1 A1 ~A0 + B1 ~A1 ~B0 + B1 ~A1 ~A0 + B1 A1 B0 A0.  S2 = ~B2 ~A2 A1 B0 A0 + ~B2 ~A2 B1 B0 A0 + ~B2 ~A2 B1 A1 + ~B2 A2 ~B1 ~A1 + ~B2 A2 ~B1 ~B0 + ~B2 A2 ~B1 ~A0 + ~B2 A2 ~A1 ~B0 + ~B2 A2 ~A1 ~A0 + B2 ~A2 ~B1 ~A1 + B2 ~A2 ~B1 ~B0 + B2 ~A2 ~B1 ~A0 + B2 ~A2 ~A1 ~B0 + B2 ~A2 ~A1 ~A0 + B2 A2 A1 B0 A0 + B2 A2 B1 B0 A0 + B2 A2 B1 A1  S3 = A3 ~B2 ~A1 ~A0 + ~B3 A3 ~A2 ~B1 ~A1 + ~B3 A3 ~A2 ~B1 ~B0 + ~B3 A3 ~A2 ~B1 ~A0 + ~B3 A3 ~A2 ~A1 ~B0 + ~B3 A3 ~A2 ~A1 ~A0 + B3 ~A3 ~B2 ~A2 + B3 ~A3 ~B2 ~B1 ~A1 + B3 ~A3 ~B2 ~B1 ~B0 + B3 ~A3 ~B2 ~B1 ~A0 + B3 ~A3 ~B2 ~A1 ~B0 + B3 ~A3 ~B2 ~A1 ~A0 + B3 ~A3 ~A2 ~B1 ~A1 + B3 ~A3 ~A2 ~B1 ~B0 + B3 ~A3 ~A2 ~B1 ~A0 + B3 ~A3 ~A2 ~A1 ~B0 + B3 ~A3 ~A2 ~A1 ~A0 + B3 A3 A2 A1 B0 A0 + B3 A3 A2 B1 B0 A0 + B3 A3 A2 B1 A1 + B3 A3 B2 A1 B0 A0 + B3 A3 B2 B1 B0 A0 + B3 A3 B2 B1 A1 + B3 A3 B2 A2.  C0 = A3 A2 A1 B0 A0 + A3 A2 B1 B0 A0 + A3 A2 B1 A1 + A3 B2 A1 B0 A0 + A3 B2 B1 B0 A0 + A3 B2 B1 A1 + A3 B2 A2 + B3 A2 A1 B0 A0 + B3 A2 B1 B0 A0 + B3 A2 B1 A1 + B3 B2 A1 B0 A0 + B3 B2 B1 B0 A0 + B3 B2 B1 A1 + B3 B2 A2 + B3 A3 | **Not Applicable**  **(Too Big)** |
| **Logic Diagram** | |
|  | |

**Exercise #4:** Design 4-bit binary adder-subtractor using full adders as shown in Fig. 6. Use full adders as sub circuits in the design. Display both the input digits, initial carry digit; output digit, and end carry digit using Hex digit display with splitter available in logisim simulator.

|  |  |
| --- | --- |
| **Boolean Expressions** | **Truth Table** |
| S0 = ~B0 A0 + B0 ~A0  S1 = ~C0 ~B1 ~A1 B0 A0 + ~C0 ~B1 A1 ~A0 + ~B1 A1 ~B0 + ~C0 B1 ~A1 ~A0 + B1 ~A1 ~B0 + ~C0 B1 A1 B0 A0 + C0 ~B1 ~A1 B0 ~A0 + C0 ~B1 A1 A0 + C0 B1 ~A1 A0 + C0 B1 A1 B0 ~A0  S2 = ~B3 A2 ~B1 ~A1 A0 + ~B3 A2 ~B1 A1 ~A0 + ~C0 A3 A2 B1 A1 + ~C0 B3 ~A2 ~A1 ~B0 + ~C0 B3 ~A2 ~A1 ~A0 + B3 ~A2 ~B1 ~B0 + B3 ~A2 ~B1 ~A1 A0 + B3 ~A2 ~B1 A1 ~A0 + ~C0 B3 A2 A1 B0 A0 + B3 A2 B1 ~A1 B0 A0 + C0 ~B3 ~A2 ~A1 B0 ~A0 + C0 ~B3 ~A2 B1 ~A1 + C0 ~B3 ~A2 B1 B0 ~A0 + C0 ~B3 A2 A1 ~B0 + C0 ~B3 A2 A1 A0 + C0 A3 A2 B1 ~A1 + C0 A3 A2 B1 B0 ~A0 + C0 B3 ~A2 A1 ~B0 + C0 B3 ~A2 A1 A0 + C0 B3 A2 ~A1 B0 ~A0  S3 = ~C0 A2 A1 B0 A0 + ~C0 A2 B1 B0 A0 + ~C0 A2 B1 A1 + B2 ~A2 A1 B0 A0 + ~C0 B2 A2 + B2 A2 ~A1 B0 ~A0 + A3 ~A2 B1 B0 A0 + A3 ~A2 B1 A1 + C0 ~A2 ~A1 B0 ~A0 + C0 ~A2 B1 ~A1 + C0 ~A2 B1 B0 ~A0 + C0 B2 ~A2 + C0 B2 B1 ~A1 + C0 B2 B1 B0 ~A0  C3 = B2 A2 ~B1 A0 + ~C0 A3 + B3 A2 ~B1 ~B0 + B3 A2 A1 ~B0 + B3 A2 A1 A0 + C0 ~B2 ~B1 ~B0 + C0 ~B2 ~B1 A0 + C0 ~B2 ~B1 A1 + C0 ~B2 A1 ~B0 + C0 ~B2 A1 A0 + C0 ~B2 A2 + C0 A2 ~B1 A1 | **Not Applicable**  **(Too Big)** |
| **Logic Diagram** | |
|  | |

**Experiment no. # 03**

**AIM: Design of basic combinational logic circuits**

**Exercise #1:** Design an 8:1 multiplexer using two 4:1 multiplexers and one 2:1 multiplexer (shown in Fig.1) Use both types of multiplexers as sub circuits in the design.

1. 2:1 MUX

|  |  |
| --- | --- |
| **Boolean Expressions** | **Logic Diagram** |
| Y = ~S0 I0 + S0 I1 |  |
| **Truth Table** |
| A screenshot of a computer code  Description automatically generated |

1. 4:1 MUX

|  |  |
| --- | --- |
| **Boolean Expressions** | **Logic Diagram** |
| Y = ~s0 ~s1 I0 + ~s0 s1 I1 + s0 ~s1 I2 + s0 s1 I3 |  |
| **Truth Table** |
| A black text on a white background  Description automatically generated |

1. 8:1 MUX using two 4:1 MUX and 2:1 MUX.

|  |  |
| --- | --- |
| **Boolean Expressions** | **Logic Diagram** |
| Y = ~S0 ~S1 ~S3 I0 + ~S0 ~S1 S3 I4 + ~S0 S1 ~S3 I2 + ~S0 S1 S3 I6 + S0 ~S1 ~S3 I1 + S0 ~S1 S3 I5 + S0 S1 ~S3 I3 + S0 S1 S3 I7 |  |
| **Karnaugh (K) Map** |
|  |

**Exercise #2:** Design a 1:8 de-multiplexer using three 1:4 de-multiplexers. Use 1:4 de-multiplexers as sub circuits in the design.

|  |  |
| --- | --- |
| **Boolean Expressions** | **Logic Diagram** |
| **Y0=~S0~S1~S2.A Y1=S0~S1~S2A Y2=~S0S1~S2A Y3=S0S1~S2A Y4=~S0~S1S2 A Y5=S0 ~S1S2 A Y6=~S0S1S2 A Y7=S0S1S3A** |  |
| **Truth Table** |
|  |

**Exercise #3:** Design quad to binary (4-to-2) encoder using logic gates. Display all four input digits using seven segment displays and two output binary bits using hex displays available in logisim simulator.

|  |  |
| --- | --- |
| **Boolean Expressions** | **Logic Diagram** |
| **O0 = I3 + I1**  **O1 = I3 + I2** |  |
| **Truth Table** |
|  |

**Exercise #4:** Design 3-to-8 decoder using two 2-to-4 decoders with enable (E) line shown in Fig. 2. Use 2:4 decoder as sub circuits in the design.

|  |  |
| --- | --- |
| **Boolean Expressions** | **Logic Diagram** |
| **I0=** **~A0 ~A1 ~A2 I1=** **~A0 A1 ~A2 I2**=**A0 ~A1 ~A2 I3=** **A0 A1 ~A2 I4=** **~A0 ~A1 A2 I5=** **~A0 A1 A2 I6=** **A0 ~A1 A2 I7=** **A0 A1 A2** |  |
| **Truth Table** |
|  |

**Experiment no. # 04**

**AIM: Design of combinational logic circuits using MUX, Encoder and seven segment displays.**

**Exercise#1**: Design full adder using in-built block of (i) 4:1 multiplexer with initial carry as input line (ii) 8:1 multiplexer.

|  |  |
| --- | --- |
| **Boolean Expressions** | **Logic Diagram** |
| Sum = ~a ~b c + ~a b ~c + a ~b ~c + a b c  carry = b c + a c + a b |  |
| **Truth Table** |
|  |

**Exercise#2**: Design a logic circuit (using in-built block of priority encoder, hex display and seven segment display) which display ‘E’, ‘O’ and ‘P’ if input octal number is even, odd and prime respectively.

|  |  |
| --- | --- |
| **Boolean Expressions** | **Logic Diagram** |
| a = b = g = 1  e = ~A2 ~A1 A0  d = A0 + ~A2 A1  f = ~A2 ~A1 + A2 ~A0 |  |
| **Truth Table** |
|  |

**Exercise#3**: Design calendar logic circuit to show the number of days in a given month. Display input month and leap year using two hex displays and number of days in outputs using eight seven segment displays.

|  |  |
| --- | --- |
| **Boolean Expressions** | **Logic Diagram** |
| Days28 = ~L ~M1 M2 ~M3 ~M4  Days29 = ~M4 ~M3 M2 ~M1 L  Days30 = ~M4 M3 ~M1 + M4 ~M3 M1  Days31 = ~M4 M1 + M4 ~M3 ~M1 + M4 ~M2 ~M1 |  |
| **Implementation Table** |
| **Not Applicable**  **(Too Big)** |

**Exercise#4:** Design logic circuit to determine whether the input year is leap year or not.

Use four hex displays to show four decimal digits of input year and one hex display to show output.

|  |
| --- |
| **Logic Diagram** |
|  |