**Department of Electrical Engineering**

**Faculty Member:** Arshad Nazir **Dated:** 12/11/2021

**Semester:** 3rd **Section:** BEE 12C

**Group No.: 7**

**EE-221: Digital Logic Design**

Lab 7: Design a Display System of Rolling Dice

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | **PLO4/CLO4** | **PLO4/CLO4** | **PLO5/CLO5** | **PLO8/CLO6** | **PLO9/CLO7** |  |
| **Name** | **Reg. No** | **Viva / Lab Performance** | **Analysis of data in Lab Report** | **Modern Tool Usage** | **Ethics and Safety** | **Individual and Teamwork** | **Total marks Obtained** |
|  |  | **5 Marks** | **5 Marks** | **5 Marks** | **5 Marks** | **5 Marks** | **25 Marks** |
| **Muhammad Adil Azeem** | **342459** |  |  |  |  |  |  |
| **Muhammad Umer** | **345834** |  |  |  |  |  |  |
| **Raahim Ahmad Waqar** | **342287** |  |  |  |  |  |  |
| **Saad Bakhtiar** | **341150** |  |  |  |  |  |  |
| **Syed Aun Ali Kazmi** | **342384** |  |  |  |  |  |  |

## Lab 7: Design a system that display from 1 to 6

This Lab Activity has been designed to practice the use of basic gates for designing a system

**Objectives:**

* Simplification of Combinational Circuits
* Design and Implementation of a design a system to display dice values.
* Values-Segment Decoder for Selected Digit Display
* There are related questions at the end of this activity. Give complete answers.

Use diagrams if needed for clarity.

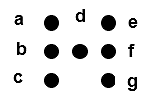
**Lab Instructions**

* This lab activity comprises three parts, namely Pre-lab, Lab tasks, and Post-Lab Viva session.
* The lab report will be uploaded on LMS three days before scheduled lab date. The students will get hard copy of lab report, complete the Pre-lab task before coming to the lab and deposit it with teacher/lab engineer for necessary evaluation. Alternately each group to upload completed lab report on LMS for grading.
* The students will start lab task and demonstrate design steps separately for stepwise evaluation (course instructor/lab engineer will sign each step after ascertaining functional verification)
* Remember that a neat logic diagram with pins numbered coupled with nicely patched circuit will simplify trouble-shooting process.
* After the lab, students are expected to unwire the circuit and deposit back components before leaving.
* The students will complete lab task and submit complete report to Lab Engineer before leaving lab.
* **The Total duration for the lab is 3 hrs. After lab duration, a deduction of 5 marks per day will be done for late submission.**
* **A lab with in-complete lab tasks will not be accepted.**
* There are related questions at the end of this activity. Give complete answers.

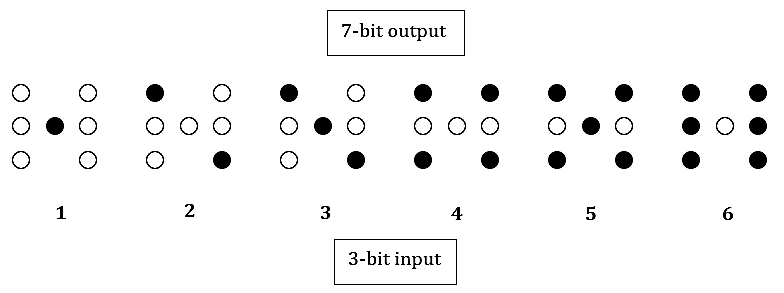
**Pre – Lab Tasks**

**Pre – Lab Task 1**

Design a system that display from 1 to 6 (ONLY)i.e., it shows no output in case of illegal input like 0 and 7. It displays the result on some dice. The dice has seven LEDs a, b, c, d, e, f, g, placed in H shape pattern as shown on the diagram below:



A 1 for each segment (a, b, c, d, e, f, g) indicates that it is lit (on); and a 0 that it is off. The arrangement is the six numbers on the dice are shown below; where the darken circles depicts LED is ON. (**Make sure to switch off all the lights in *Don’t Care* case**)



*Design a driver that produces the seven LED (a, b, c, d, e, f, g) to drive the display.*

1. Complete the following table. (**Make sure to switch off all the lights in *Don’t Care* case**)

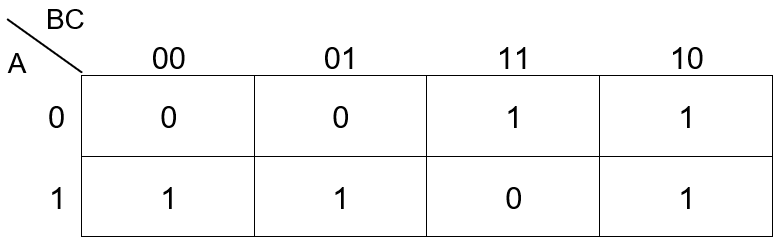
|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Inputs (Binary)** | | | **Outputs (7 LEDs on Dice Display)** | | | | | | |
| **#m** | **A** | **B** | **C** | **a** | **b** | **c** | **d** | **e** | **f** | **g** |
| **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** |
| **1** | **0** | **0** | **1** | **0** | **0** | **0** | **1** | **0** | **0** | **0** |
| **2** | **0** | **1** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **1** |
| **3** | **0** | **1** | **1** | **1** | **0** | **0** | **1** | **0** | **0** | **1** |
| **4** | **1** | **0** | **0** | **1** | **0** | **1** | **0** | **1** | **0** | **1** |
| **5** | **1** | **0** | **1** | **1** | **0** | **1** | **1** | **1** | **0** | **1** |
| **6** | **1** | **1** | **0** | **1** | **1** | **1** | **0** | **1** | **1** | **1** |
| **7** | **1** | **1** | **1** | **0** | **1** | **0** | **0** | **0** | **0** | **0** |

1. Write minimum possible functions to realize outputs (Either using k-mapping/ or minimization of Boolean function)

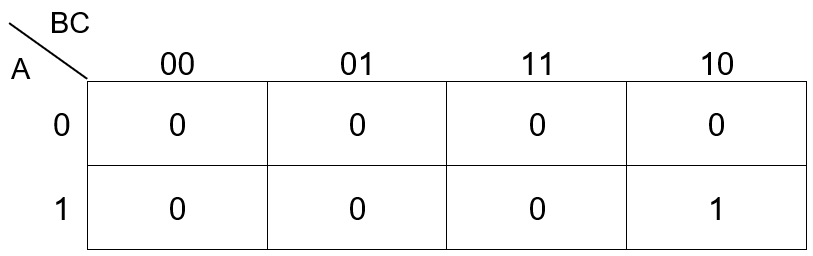
**(Make sure to switch off all the lights in Don’t care case).**

Show and get verified the minimized Boolean Function expressions to Lab Engineer before implementation.

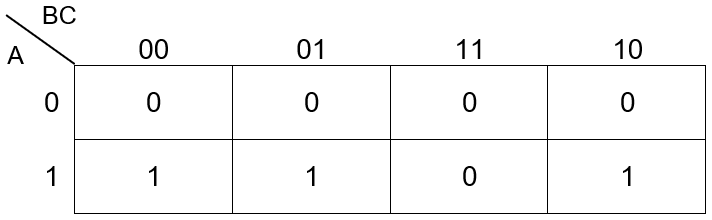
**For a and g**



**For b and f**

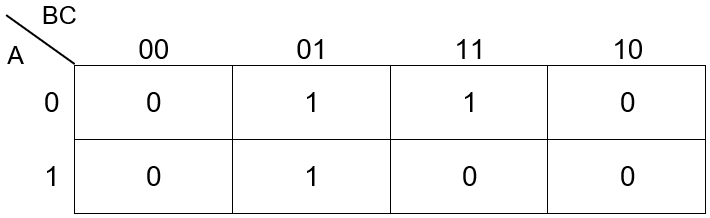


**For c and e**





**For d**

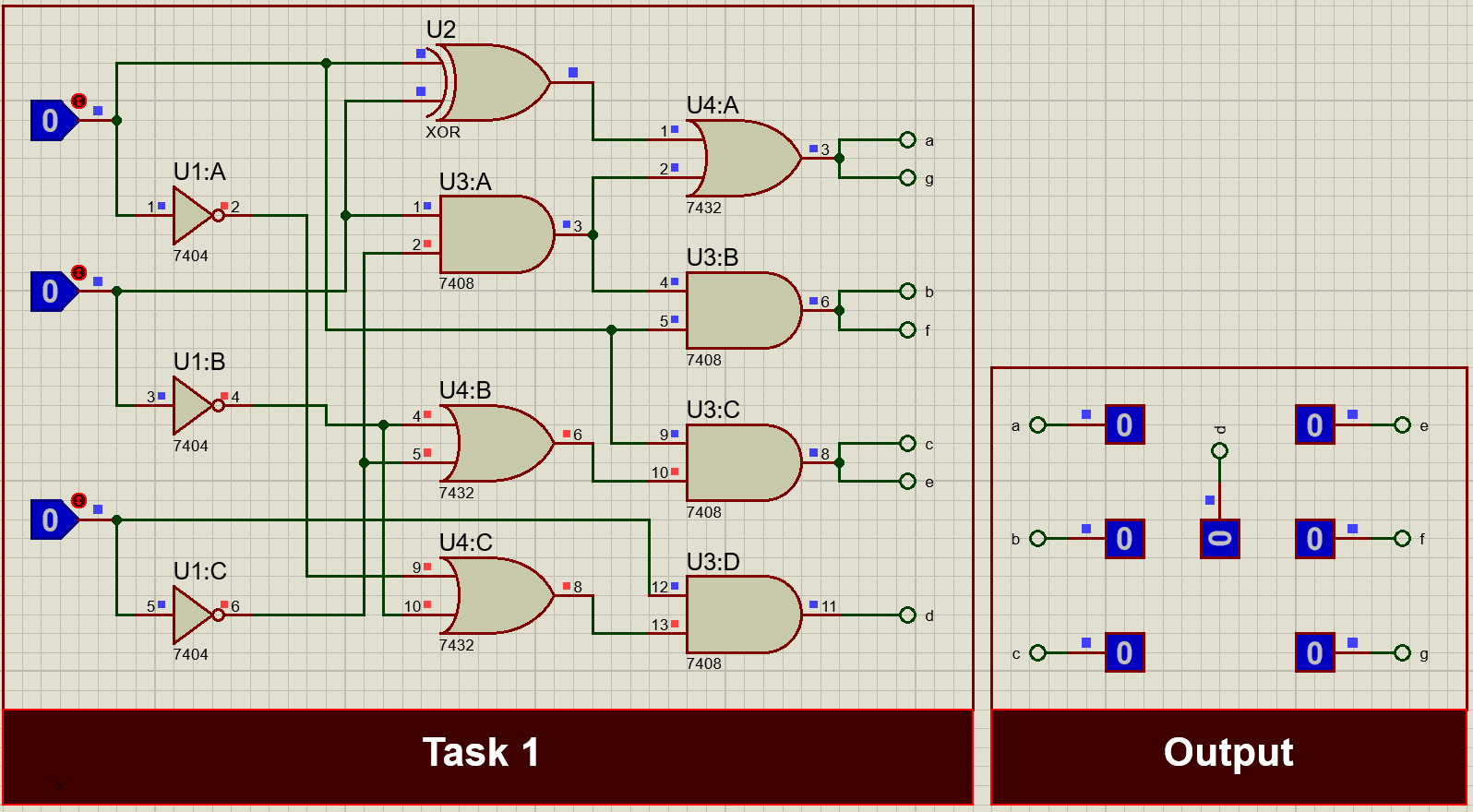


**Simplified Expressions**

|  |  |  |
| --- | --- | --- |
| **Variable** | | **Expression** |
| **a** | **g** | **(A ⊕ B) + BC’** |
| **b** | **f** | **A(BC’)** |
| **c** | **e** | **A(B’+C’)** |
| **d** | **-** | **C(A’+B’)** |

1. Draw the complete logic circuit diagram of the system from simplified equations.

**Schematic / Proteus Simulation**



**Lab Tasks**

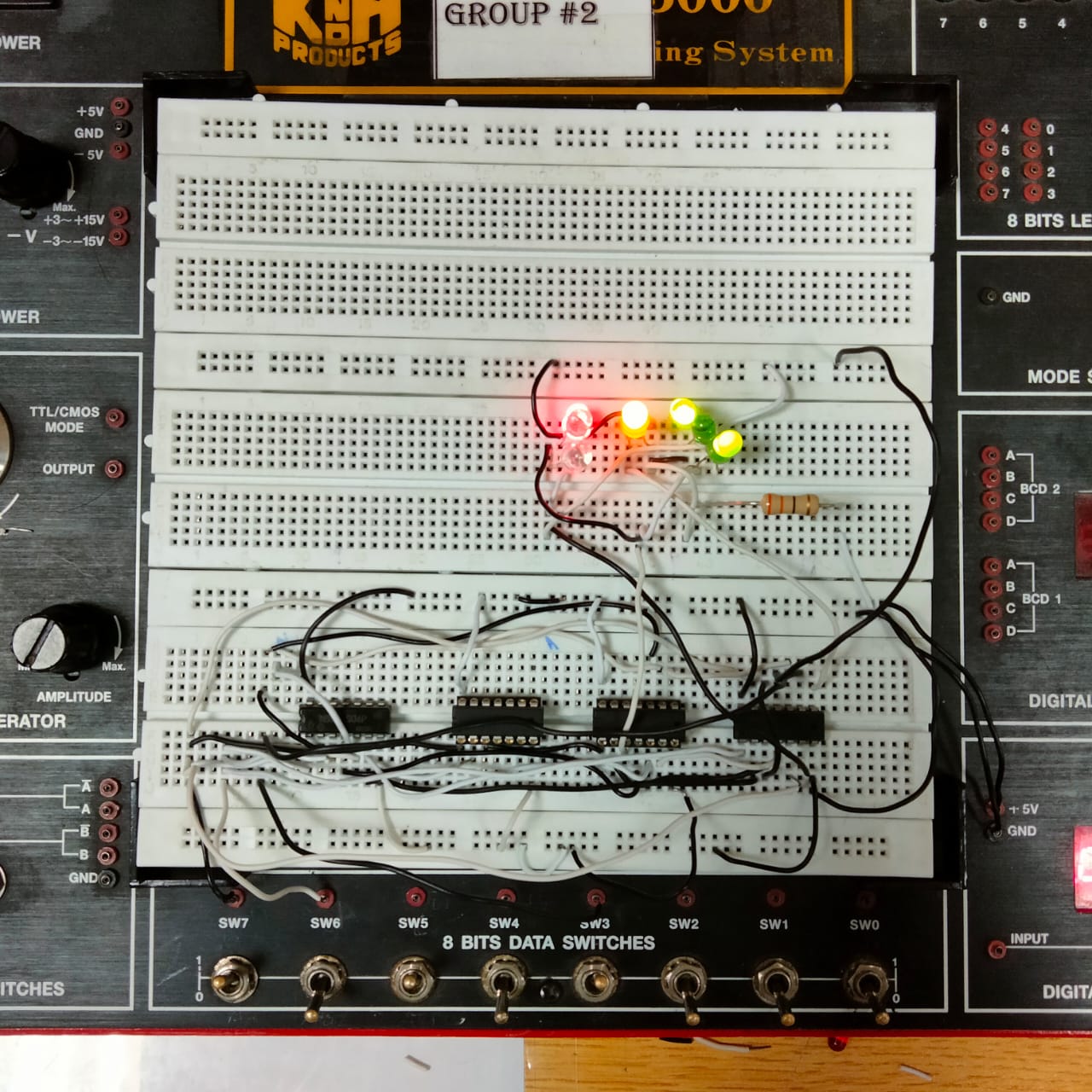
**Lab Task 1**

1. Implement the designed logic circuit on hardware. Utilize your creativity to make the Dice display **model.**

**Truth Table**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Inputs (Binary)** | | | **Outputs (7 LEDs on Dice Display)** | | | | | | |
| **#m** | **A** | **B** | **C** | **a** | **b** | **c** | **d** | **e** | **f** | **g** |
| **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** |
| **1** | **0** | **0** | **1** | **0** | **0** | **0** | **1** | **0** | **0** | **0** |
| **2** | **0** | **1** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **1** |
| **3** | **0** | **1** | **1** | **1** | **0** | **0** | **1** | **0** | **0** | **1** |
| **4** | **1** | **0** | **0** | **1** | **0** | **1** | **0** | **1** | **0** | **1** |
| **5** | **1** | **0** | **1** | **1** | **0** | **1** | **1** | **1** | **0** | **1** |
| **6** | **1** | **1** | **0** | **1** | **1** | **1** | **0** | **1** | **1** | **1** |
| **7** | **1** | **1** | **1** | **0** | **1** | **0** | **0** | **0** | **0** | **0** |

**Hardware Implementation**



1. Write Verilog code to realize the design using dataflow model. Design also test bench to check the valid outputs. Include all the timing diagram snap shots and Verilog code in the report.

**Dataflow Model**

module task1(a, b, c, d, e, f, g, A, B, C);

    input A, B, C;

    output a, b, c, d, e, f, g;

    assign a = (~A & B) | (A & ~B) | (B & ~C);

    assign g = (~A & B) | (A & ~B) | (B & ~C);

    assign b = A & (B & ~C);

    assign f = A & (B & ~C);

    assign c = A & (~B | ~C);

    assign e = A & (~B | ~C);

    assign d = C & (~A | ~B);

endmodule

module testbench;

    reg A, B, C;

    wire a, b, c, d, e, f, g;

    task1 t1(a, b, c, d, e, f, g, A, B, C);

    initial

        begin

        #100 A = 0; B = 0; C = 0;

        #100 A = 0; B = 0; C = 1;

        #100 A = 0; B = 1; C = 0;

        #100 A = 0; B = 1; C = 1;

        #100 A = 1; B = 0; C = 0;

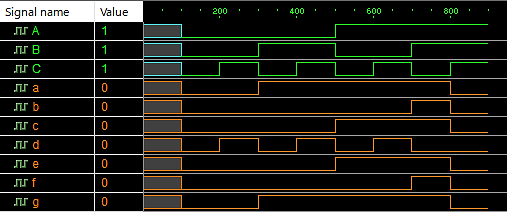
        #100 A = 1; B = 0; C = 1;

        #100 A = 1; B = 1; C = 0;

        #100 A = 1; B = 1; C = 1;

    end

endmodule



**Gate – Level Model**

module task1(a, b, c, d, e, f, g, A, B, C);

    input A, B, C;

    output a, b, c, d, e, f, g;

    wire w1, w2, w3, w4, w5, w6, w7;

    not n1(w1, A);

    not n2(w2, B);

    not n3(w3, C);

    xor x1(w4, A, B);

    and a1(w5, B, w3);

    or o1(a, w4, w5);

    or o2(g, w4, w5);

    and a2(b, w5, A);

    and a3(f, w5, A);

    or o3(w6, w2, w3);

    and a4(c, w6, A);

    and a5(e, w6, A);

    or o4(w7, w1, w2);

    and a6(d, w7, C);

endmodule

module testbench;

    reg A, B, C;

    wire a, b, c, d, e, f, g;

    task1 t1(a, b, c, d, e, f, g, A, B, C);

    initial

        begin

        #100 A = 0; B = 0; C = 0;

        #100 A = 0; B = 0; C = 1;

        #100 A = 0; B = 1; C = 0;

        #100 A = 0; B = 1; C = 1;

        #100 A = 1; B = 0; C = 0;

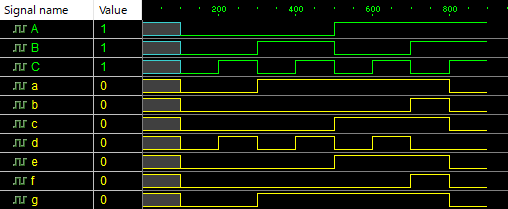
        #100 A = 1; B = 0; C = 1;

        #100 A = 1; B = 1; C = 0;

        #100 A = 1; B = 1; C = 1;

    end

endmodule

****

**Observations/Comments:**

In this lab we implemented the logic to design a display system of a rolling dice using:

1. Basic gates (AND, NOT and OR)
2. Using XOR

We implemented the LEDs (placed in an H - shape) at the receier end of the circuit and wrote the Verilog code for the circuit and verified the outputs.