**Задача 1: При първият модул , времето за закъснение е твърде малък и се получава глич , при увеличаване на времето >#1, гличът се изчиства. При вторият модул, трябва да се опише състояние когато а=0 , тъй като симулаторът, чете стойността b като запомнена предишна , което се характеризира за латч тригерите и автоматично генерира латч.**

**Задача 2:** [**https://www.edaplayground.com/x/8S7v**](https://www.edaplayground.com/x/8S7v) **Задача 3: /\***

**\* Y. Gorbounov 2021**

**\*/**

**`timescale 1ps / 1fs**

**module bin2hex\_gatelevel\_fixed (**

**input x1, x2, x3, x4,**

**output a, b, c, d, e, f, g**

**);**

**wire a1, a2, a3, a4, a5, a6;**

**wire b1, b2, b3, b4, b5;**

**wire c1, c2, c3, c4, c5;**

**wire d1, d2, d3, d4, d5;**

**wire e1, e2, e3, e4;**

**wire f1, f2, f3, f4, f5;**

**wire g1, g2, g3, g4, g5;**

**wire x1, x2, x3\_, x4\_; //direct values**

**wire x1n, x2n, x3n, x4n; //invert values of inputs**

**wire vcc=1'b1;**

**wire gnd=1'b0;**

**//corrective circuit**

**xor #10(x1n, vcc, x1); //for signal x1**

**xor #10(x1\_, gnd, x1);**

**xor #10(x2n, vcc, x2); //for signal x2**

**xor #10(x2\_, gnd, x2);**

**xor #10(x3n, vcc, x3); //for signal x3**

**xor #10(x3\_, gnd, x3);**

**xor #10(x4n, vcc, x4); //for signal x4**

**xor #10(x4\_, gnd, x4);**

**//Instantiation of logic follows.**

**and(a1, x2\_, x3\_);**

**and(a2, x1n, x3\_);**

**and(a3, x1\_, x4n);**

**and(a4, x2n, x4n);**

**and(a5, x1n, x2\_, x4\_);**

**and(a6, x1\_, x2n, x3n);**

**or(a, a1, a2, a3, a4, a5, a6); // f1 <=> a**

**and(b1, x2n, x4n);**

**and(b2, x2n, x3n);**

**and(b3, x1n, x3\_, x4\_);**

**and(b4, x1\_, x3n, x4\_);**

**and(b5, x1n, x3n, x4n);**

**or(b, b1, b2, b3, b4, b5); // f2 <=> b**

**and(c1, x3n, x4\_);**

**and(c2, x1n, x4\_);**

**and(c3, x1\_, x2n);**

**and(c4, x1n, x2\_);**

**and(c5, x2n, x3n);**

**or(c, c1, c2, c3, c4, c5); // f3 <=> c**

**and(d1, x1n, x3\_, x4n);**

**and(d2, x2n, x3\_, x4\_);**

**and(d3, x2\_, x3n, x4\_);**

**and(d4, x1, x2, x4n);**

**and(d5, x2n, x3n, x4n);**

**or(d, d1, d2, d3, d4, d5); // f4 <=> d**

**and(e1, x3\_, x4n);**

**and(e2, x1\_, x3\_);**

**and(e3, x1\_, x2\_);**

**and(e4, x2n, x4n);**

**or(e, e1, e2, e3, e4); // f5 <=> e**

**and(f1, x1\_, x3\_);**

**and(f2, x1\_, x2n);**

**and(f3, x2\_, x4n);**

**and(f4, x3n, x4n);**

**and(f5, x1n, x2\_, x3n);**

**or(f, f1, f2, f3, f4, f5); // f6 <=> f**

**and(g1, x1\_, x3\_);**

**and(g2, x2n, x3\_);**

**and(g3, x1\_, x2n);**

**and(g4, x2\_, x3n, x4\_);**

**and(g5, x1n, x2\_, x4n);**

**or(g, g1, g2, g3, g4, g5); // f7 <=> g**

**endmodule**

