CSE 369 Lab 3

Digital Design using FPGAs

Isaac Wu
2360957
October 23, 2024

1. Karnaugh Maps

(a) Discounted

U	P	C	Mark	Discounted
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	X
0	1	0	1	X
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	X
1	1	1	1	X

CM \ UP	00	01	11	10
00	0	X	1	0
01	0	X	1	0
11	0	1	X	1
10	0	1	X	1

$$\text{Discounted} = \mathbf{P} + \mathbf{UC}$$

(b) Stolen

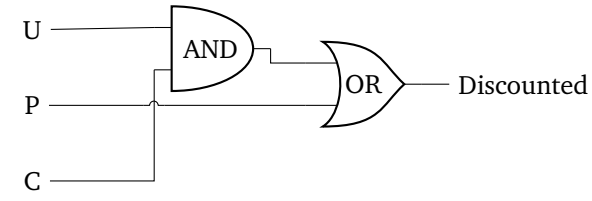
U	P	C	Mark	Stolen
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	X
0	1	0	0	X
0	1	0	1	X
0	1	1	0	0
0	1	1	1	X
1	0	0	0	1
1	0	0	1	0
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	X
1	1	1	0	X
1	1	1	1	X

CM \ UP	00	01	11	10
00	1	X	0	1
01	0	X	X	0
11	X	X	X	0
10	0	0	X	1

Stolen $= \overline{\mathbf{U}}\,\overline{\mathbf{C}}\,\overline{\mathbf{M}} + \mathbf{U}\,\overline{\mathbf{P}}\,\overline{\mathbf{M}}$

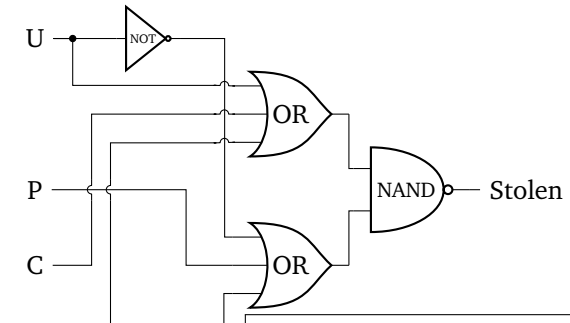
2. Circuit Diagrams

(a) Discounted



Mark $\boxed{P + UC}$

(b) Stolen



$$\boxed{\overline{U}\,\overline{C}\,\overline{M} + U\,\overline{P}\,\overline{M} \rightarrow \overline{(U + C + M)(\overline{U} + P + M)}}$$

3. ModelSim Simulation

Here is the code for my recognizer circuit. I defined logic variables for U, P, C, and Mark to make it easier to translate the inputs. Then, using the logic calculated using the Karnaugh Maps above, I assigned values to Discounted and Stolen, two logic variables defined above. Then, I assigned the values of Discounted and Stolen to two LEDs to display the output of the calculation.

```
/* Top-level module that defines the I/Os for the DE1-SoC board
 * and the circuit behavior.
 */
module lab3 (
  output logic [6:0] HEX0, HEX1, HEX2, HEX3, HEX4, HEX5,
  output logic [9:0] LEDR,
  input  logic [3:0] KEY,
  input  logic [9:0] SW
  );

  // Default values, turns off the HEX displays
  assign HEX0 = 7'b1111111;
  assign HEX1 = 7'b1111111;
  assign HEX2 = 7'b1111111;
  assign HEX3 = 7'b1111111;
  assign HEX4 = 7'b1111111;
  assign HEX5 = 7'b1111111;

  logic U, P, C, Mark, Discounted, Stolen;
  assign U = SW[3];
  assign P = SW[2];
  assign C = SW[1];
  assign Mark = SW[0];

  assign Discounted = P | (U & C);
  assign Stolen = (~U & ~C & ~Mark) | (U & ~P & ~Mark);

  assign LEDR[0] = Discounted;
  assign LEDR[1] = Stolen;

endmodule  // lab3

```

This is the code for my test benchmark used to ensure correct functionality from my circuit. This code will run my lab3 module with all 16 combinations of switch inputs.

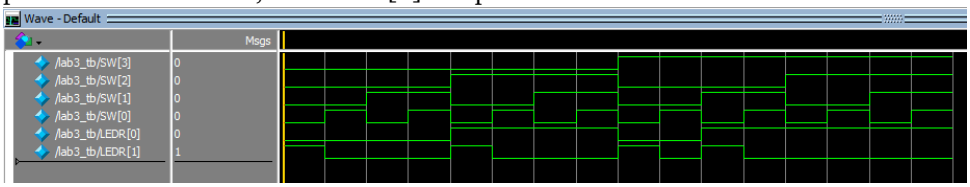
```
module lab3_tb();
  logic [6:0] HEX0, HEX1, HEX2, HEX3, HEX4, HEX5;
  logic [9:0] LEDR;
  logic [3:0] KEY;
  logic [9:0] SW;

  // instantiate device under test
  lab3 dut (.HEX0, .HEX1, .HEX2, .HEX3, .HEX4, .HEX5, .KEY, .LEDR, .SW);

  // test input sequence - try all combinations of inputs
  integer i;
  initial begin
    SW[9:4] = 1'b0;
    for(i = 0; i < 16; i++) begin
      SW[3:0] = i; #10;
    end
  end
endmodule  // lab3_tb

```

This is what my wave diagram looks like. The top 4 waves show the 4 switches alternating to test all 16 combinations. The bottom two waves are the LED output, with LEDR[0] corresponding to if a product is Discounted, and LEDR[1] if a product was stolen.



4. Misc.

How many hours (estimated) it took to complete this lab in total, including reading, planning, designing, coding, debugging, and testing.

It took around 7 hours to complete this lab.