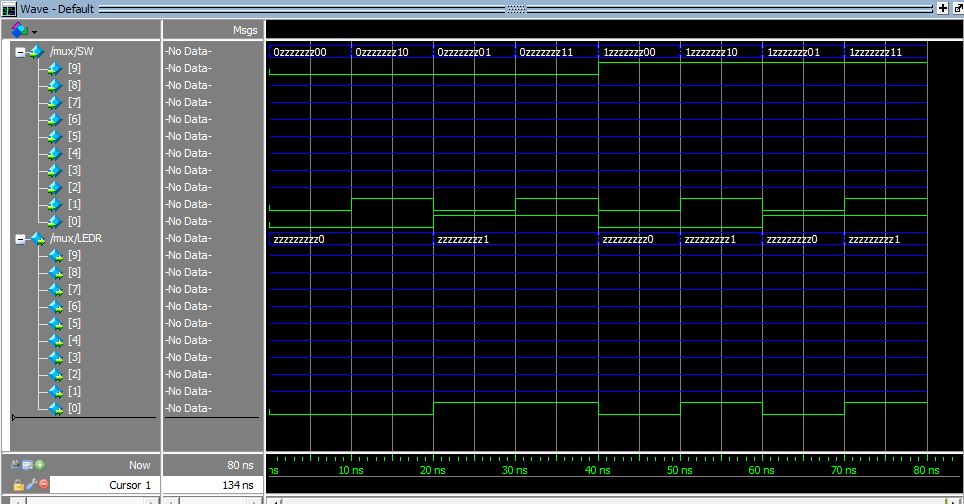
**Lab 2 Pre-lab Report**

Ziyao Chen

1002453409

**Part 1**





There are 8 provided test-cases in total, and inputs are SW[0] (x), SW[1] (y), SW[9] (s); and the output is LEDR[0] (m). From the wave form, we can get the information of inputs and output. And we expect to have the Boolean expression m = x & ~s | y & s, which in our simulation, we expect to have LEDR[0] = SW[0] & ~SW[9] | SW[1] & SW[9].

1. SW[0] = 0, SW[1] = 0, SW[9] = 0

We expect to have output 0 here, and the waveform according to LEDR[0] is 0 as expected.

1. SW[0] = 0, SW[1] = 1, SW[9] = 0

We expect to have output 0 here, and the waveform according to LEDR[0] is 0 as expected.

1. SW[0] = 1, SW[1] = 0, SW[9] = 0

We expect to have output 1 here, and the waveform according to LEDR[0] is 1 as expected.

1. SW[0] = 1, SW[1] = 1, SW[9] = 0

We expect to have output 1 here, and the waveform according to LEDR[0] is 1 as expected.

1. SW[0] = 0, SW[1] = 0, SW[9] = 1

We expect to have output 0 here, and the waveform according to LEDR[0] is 0 as expected.

1. SW[0] = 0, SW[1] = 1, SW[9] = 1

We expect to have output 1 here, and the waveform according to LEDR[0] is 1 as expected.

1. SW[0] = 1, SW[1] = 0, SW[9] = 1

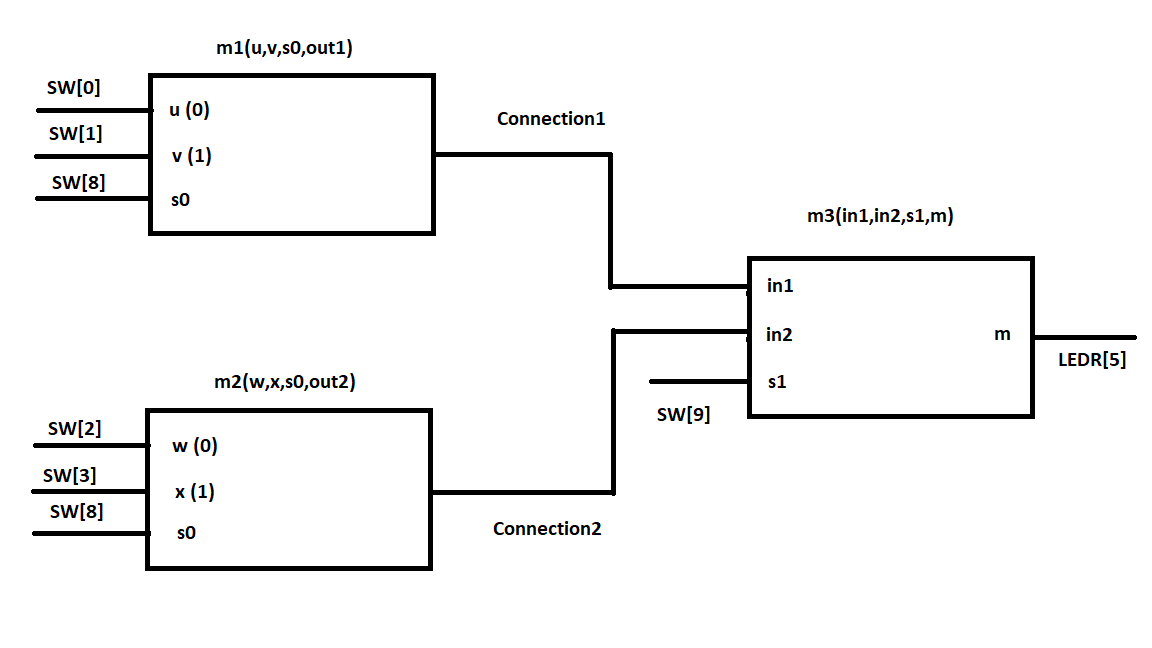
We expect to have output 0 here, and the waveform according to LEDR[0] is 0 as expected.

1. SW[0] = 1, SW[1] = 1, SW[9] = 1

We expect to have output 1 here, and the waveform according to LEDR[0] is 1 as expected.

**Part 2**

1. It would have 26 = 64 rows.

2.

3.

module mux4to1(u, v, w, x, s1, s0, m);

input u; //selected when s1 is 0 and s0 is 0

input v; //selected when s1 is 0 and s0 is 1

input w; //selected when s1 is 1 and s0 is 0

input x; //selected when s1 is 1 and s0 is 1

input s1,s0; //select signal

output m;

wire Connection1, Connection2;

mux2to1 m1(u,v,s0,Connection1);

mux2to1 m2(w,x,s0,Connection2);

mux2to1 m3(Connection1,Connection2,s1,m);

endmodule

module mux2to1(x, y, s, m);

input x; //selected when s is 0

input y; //selected when s is 1

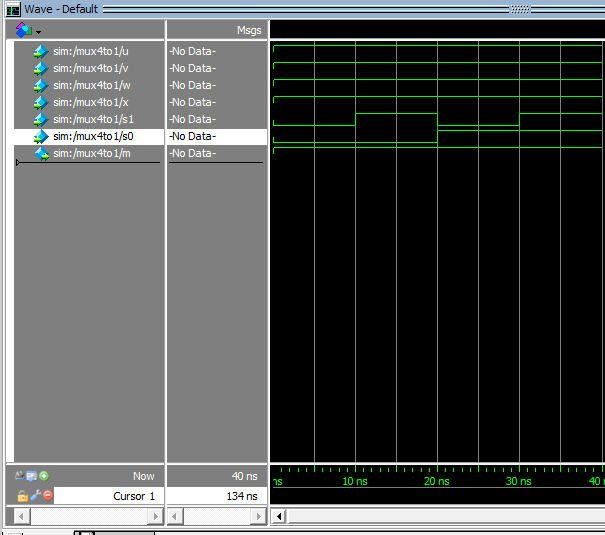
input s; //select signal

output m; //output

assign m = s & y | ~s & x;

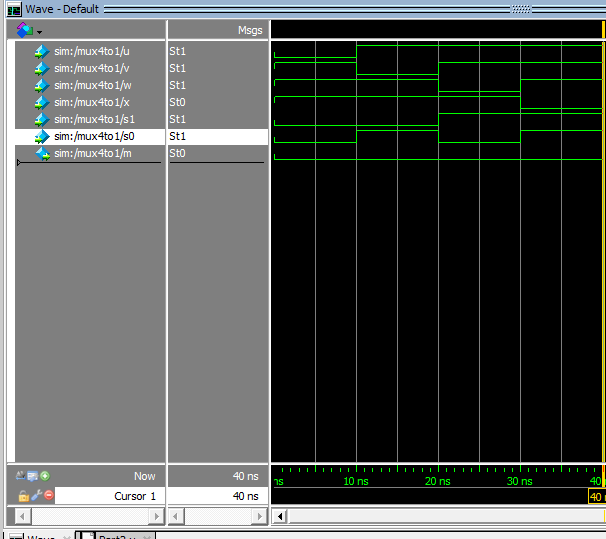
endmodule

5. First, we set the value on u, v, w, x all on 1. Thus, we now expect to have ouput 1, no matter what value we set for s0 and s1, and there are only four possible combination of s0 and s1 (00, 01, 10, 11 respectively).



In these four simulation, we get the expected output.

Now, we simulate another 4 test cases. Make u = 0 when s1 is 0 and s0 is 0; v = 0 when s1 is 0 ad s0 is 1; w = 0 when s1 is 1 and s0 is 0; x is 0 when s1 is 1 and s0 is 1. We don’t care about the value of other not mentioned inputs in each test-case, and we expect to get ouput 0 in all 4 test cases.



As the waveforms demonstrated, we got our expected outputs as 0 in all four test-cases.

**Part3**



|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Character** | **0** | **1** | **2** | **3** | **4** | **5** | **6** |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| 2 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 3 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| 4 | 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| 5 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| 6 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 7 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 9 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| A | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| B | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| C | 0 | 1 | 1 | 0 | 0 | 0 | 1 |
| D | 1 | 0 | 0 | 0 | 0 | 1 | 0 |
| E | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| F | 0 | 1 | 1 | 1 | 0 | 0 | 0 |

0 = m1 + m4 + m11 + m13

= A̅B̅C̅D + A̅BC̅D̅ + AB̅CD + ABC̅D

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | C̅D̅ | C̅D | CD | CD̅ |
| A̅B̅ | 0 | 1 | 0 | 0 |
| A̅B | 1 | 0 | 0 | 0 |
| AB | 0 | 1 | 0 | 0 |
| AB̅ | 0 | 0 | 1 | 0 |

1 = m5 + m6 + m11 + m12 + m14 + m15

= A̅BC̅D + A̅BCD̅ + AB̅CD + ABC̅D̅ + ABCD̅ + ABCD

= ACD + BCD̅ + A̅BC̅D + ABD̅

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | C̅D̅ | C̅D | CD | CD̅ |
| A̅B̅ | 0 | 0 | 0 | 0 |
| A̅B | 0 | 1 | 0 | 1 |
| AB | 1 | 0 | 1 | 1 |
| AB̅ | 0 | 0 | 1 | 0 |

2 = m2 + m12 + m14 + m15

= A̅B̅CD̅ + ABC̅D̅ + ABCD̅ +ABCD

= A̅B̅CD̅ + ABD̅ + ABC

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | C̅D̅ | C̅D | CD | CD̅ |
| A̅B̅ | 0 | 0 | 0 | 1 |
| A̅B | 0 | 0 | 0 | 0 |
| AB | 1 | 0 | 1 | 1 |
| AB̅ | 0 | 0 | 0 | 0 |

3 = m1 + m4 + m7 + m10 + m15

= A̅B̅C̅D + A̅BC̅D̅ + A̅BCD + AB̅CD̅ + ABCD

= A̅B̅C̅D + A̅BC̅D̅ + BCD + AB̅CD̅

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | C̅D̅ | C̅D | CD | CD̅ |
| A̅B̅ | 0 | 1 | 0 | 0 |
| A̅B | 1 | 0 | 1 | 0 |
| AB | 0 | 0 | 1 | 0 |
| AB̅ | 0 | 0 | 0 | 1 |

4 = m1 + m3 + m4 + m5 + m7 + m9

= A̅B̅C̅D + A̅B̅CD +A̅BC̅D̅ + A̅BC̅D + A̅BCD + AB̅C̅D

= A̅D + A̅BC̅ + B̅C̅D

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | C̅D̅ | C̅D | CD | CD̅ |
| A̅B̅ | 0 | 1 | 1 | 0 |
| A̅B | 1 | 1 | 1 | 0 |
| AB | 0 | 0 | 0 | 0 |
| AB̅ | 0 | 1 | 0 | 0 |

5 = m1 + m2 + m3 + m7 + m13

= A̅B̅C̅D + A̅B̅CD̅ + A̅B̅CD + A̅BCD + ABC̅D

= A̅B̅D + A̅B̅C + A̅CD + ABC̅D

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | C̅D̅ | C̅D | CD | CD̅ |
| A̅B̅ | 0 | 1 | 1 | 1 |
| A̅B | 0 | 0 | 1 | 0 |
| AB | 0 | 1 | 0 | 0 |
| AB̅ | 0 | 0 | 0 | 0 |

6 = m0 + m1 + m7 + m12

= A̅B̅C̅D̅ + A̅B̅C̅D + A̅BCD + ABC̅D̅

= A̅B̅C̅ + A̅BCD + ABC̅D̅

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | C̅D̅ | C̅D | CD | CD̅ |
| A̅B̅ | 1 | 1 | 0 | 0 |
| A̅B | 0 | 0 | 1 | 0 |
| AB | 1 | 0 | 0 | 0 |
| AB̅ | 0 | 0 | 0 | 0 |



module SevenSegmentDecoder(c3,c2,c1,c0,ss\_out);

input c3,c2,c1,c0; //the bcd input

output [6:0] ss\_out; //the seven segements

assign ss\_out[0] = ~c3 & ~c2 & ~c1 & c0 | ~c3 & c2 & ~c1 & ~c0 | c3 & ~c2 & c1 & c0 | c3 & c2 & ~c1 & c0;

assign ss\_out[1] = c3 & c1 & c0 | c2 & c1 & ~c0 | ~c3 & c2 & ~c1 & c0 | c3 & c2 & ~c0;

assign ss\_out[2] = ~c3 & ~c2 & c1 & ~c0 | c3 & c2 & ~c0 | c3 & c2 & c1;

assign ss\_out[3] = ~c3 & ~c2 & ~c1 & c0 | ~c3 & c2 & ~c1 & ~c0 | c2 & c1 & c0 | c3 & ~c2 & c1 & ~c0;

assign ss\_out[4] = ~c3 & c0 | ~c3 & c2 & ~c1 | ~c2 & ~c1 & c0;

assign ss\_out[5] = ~c3 & ~c2 & c0 | ~c3 & ~c2 & c1 | ~c3 & c1 & c0 | c3 & c2 & ~c1 & c0;

assign ss\_out[6] = ~c3 & ~c2 & ~c1 | ~c3 & c2 & c1 & c0 | c3 & c2 & ~c1 & ~c0;

endmodule



