Switching Circuit and Logic Design Laboratory Report-4

by

Group-11

Devendra Palod (*20CS10024*)

Subhajyoti Halder (*20CS10064*)

Anuj Kakde (*20CS30005*)

Deepiha S. (*20CS30015*)

Department of Computer Science and Engineering

Indian Institute of Technology Kharagpur

Spring Semester, 2021-22

Contents

Logic file url:
https://github.com/subhu-darkknight72/SCLD_Lab-4_20CS10064.git

1 Part 1 (≥ 5 Comparator)

1.1 Problem Statement

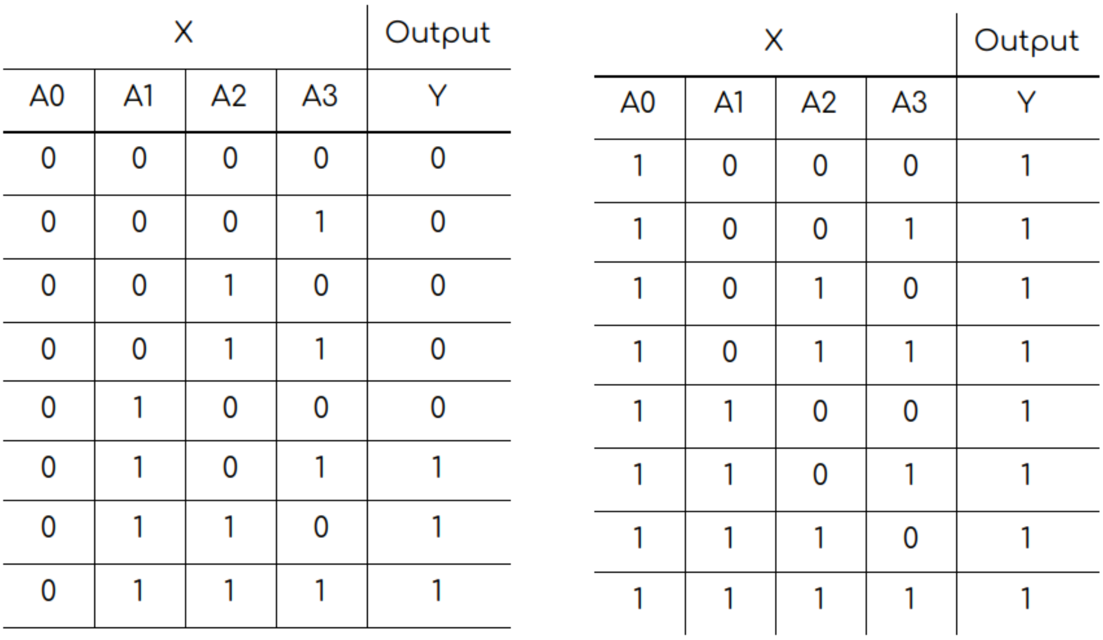
1. Design a comparator to take as input a 4-bit binary number X and output 1 if $X \geq 5$ and 0 otherwise.
2. Test that it works by applying appropriate inputs and checking the outputs.
3. Label the terminals to reflect their roles.
4. Save it as a regular circuit (logic file), reopen and retest.
5. Save it as a component (cmp file), reopen and retest.

1.2 Solution Description

Consider the binary number input to the comparator be X.
Let the binary form of the input X be $A_0A_1A_2A_3$ as expressed in the logic circuit.

1.3 Logic Expression

$$Y = A_0 \vee (A_1 \wedge (A_2 \vee A_3))$$



X				Output
A0	A1	A2	A3	Y
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1

X				Output
A0	A1	A2	A3	Y
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

Figure 1: Truth Table for ≥ 5 Comparator

1.4 Circuit Diagram

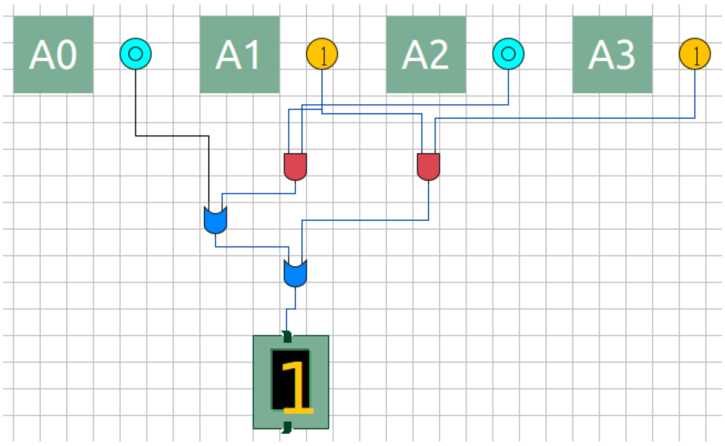


Figure 2: ≥ 5 comparator Circuit

2 Part 2 (≥ 5 3-adder)

2.1 Problem Statement

1. Design a conditional adder to take as input a 4-bit binary number X and output $X + 3$ if $X \geq 5$ and X otherwise; you may use components designed earlier.

2. Test that it works by applying appropriate inputs and checking the outputs.

3. Label the terminals to reflect their roles.

4. Save it as a regular circuit (logic file), reopen and retest.

5. Save it as a component (cmp file), reopen and retest.

2.2 Solution Description

This is again a comparator that is used to test whether X $\geq$ 5 by changing the nature of output. If (X $\geq$ 5), we add 11 (in binary) to X, else 00 (in binary) to X using Half adders and Full Adders.

2.3 Circuit Diagram

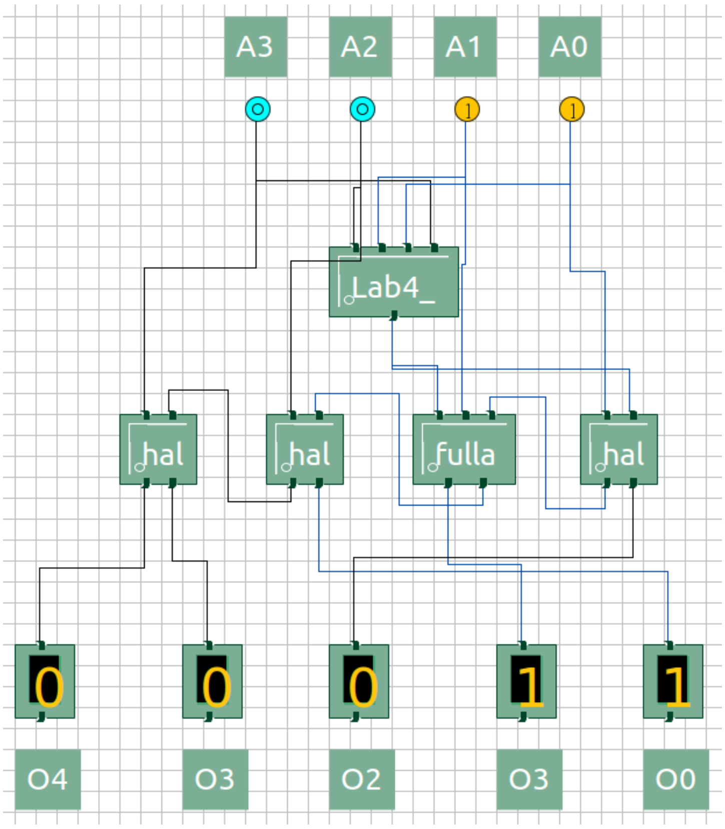


Figure 3: ≥ 5 3adder Circuit

3 Part 3 (4-bit binary to BCD convertor)

3.1 Problem Statement

1. Using the conditinal adder modules, design a combinational 4-bit binary to BCD convertor.
2. Test that it works by applying appropriate inputs and checking the outputs.
3. Label the terminals to reflect their roles.
4. Save it as a regular circuit (logic file), reopen and retest.

3.2 Solution Description

Let the 4-bit binary number X be $A_3A_2A_1A_0$ and the BCD Value of the binary number X be $B_4B_3B_2B_1B_0$. In the 4-bit binary to BCD converter we used one unit of 3 adder the output can't exceed 5 until the three bits $A_2A_1A_0$ are shifted. Therefore we only require 3adder when $A_2A_1A_0$ are shifted in and after the shifting only one bit A_3 remains unchanged.

3.3 Circuit Diagram

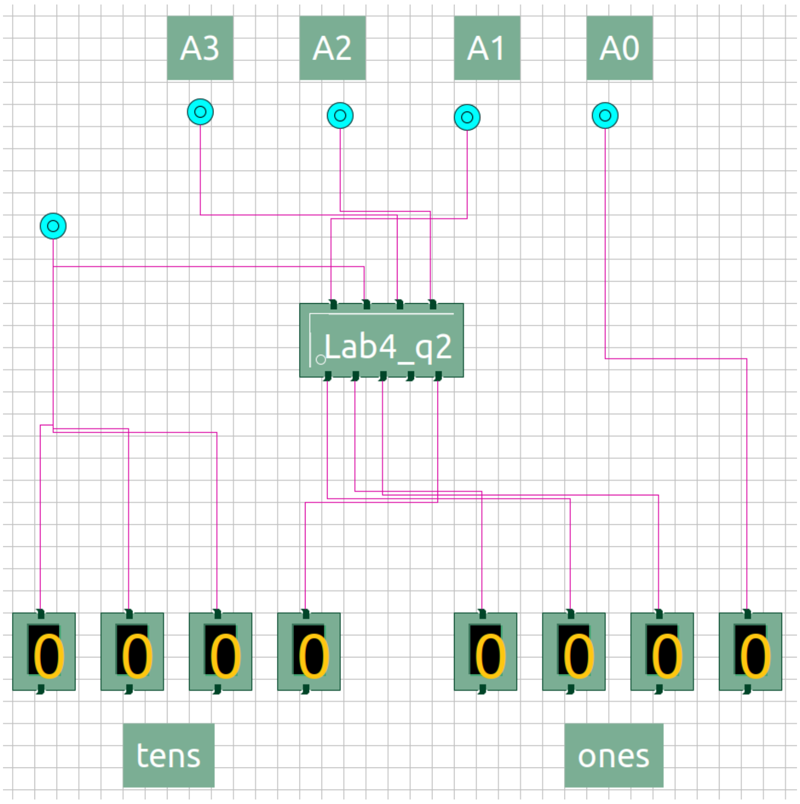


Figure 4: 4-bit binary to BCD converter Circuit

4 Part 4 (6-bit binary to BCD convertor)

4.1 Problem Statement

1. Using the conditinal adder modules, design a combinational 6-bit binary to BCD convertor.
2. Test that it works by applying appropriate inputs and checking the outputs.
3. Label the terminals to reflect their roles.
4. Save it as a regular circuit (logic file), reopen and retest.

4.2 Solution Description

Let the 6-bit binary number X be $A_5A_4A_3A_2A_1A_0$ and it's BCD Value be $B_7B_6B_5B_4B_3B_2B_1B_0$.

In the 6 bit binary to BCD converter we used three units of 3adder

a) First 3-adder takes A_2, A_1, A_0, 0 bit switch(consider it as fixed) as input. and its carry would be ignored, the MSB of sum is redirected to output B_2 as it would not change in the comming steps and hence it would appear in output as it is. and finally the other bits of sum (S_0, S_1, S_2) are redirected as input to the second adder.

b) In the second 3-adder, it takes A_3 and three bits from the first 3 adder as its inputs.This input corresponds to the shift step in the Double Dabble algorithm. Its carry would also be ignored, the MSB of sum is redirected to output B_3 as it would not change in the comming steps and hence it would appear in output as it is and the other bits of sum (S_0, S_1, S_2) are redirected as input to the third adder.

c) Similarly the third 3 adder takes A_4 and three bits from the first 3 adder as its inputs.This input corresponds to the second shift step in the Double Dabble algorithm.Its carrry is also ignored and the sum is redirected to output $B_6B_5B_4$ as no further steps are involved.

d) Finally A_5 is redirected to B_8 as no shift or 3 addition is going to take place.

This completes the Double Dabble Algorithm for conversion of a 6 bit binary number to BCD.

4.3 Circuit Diagram

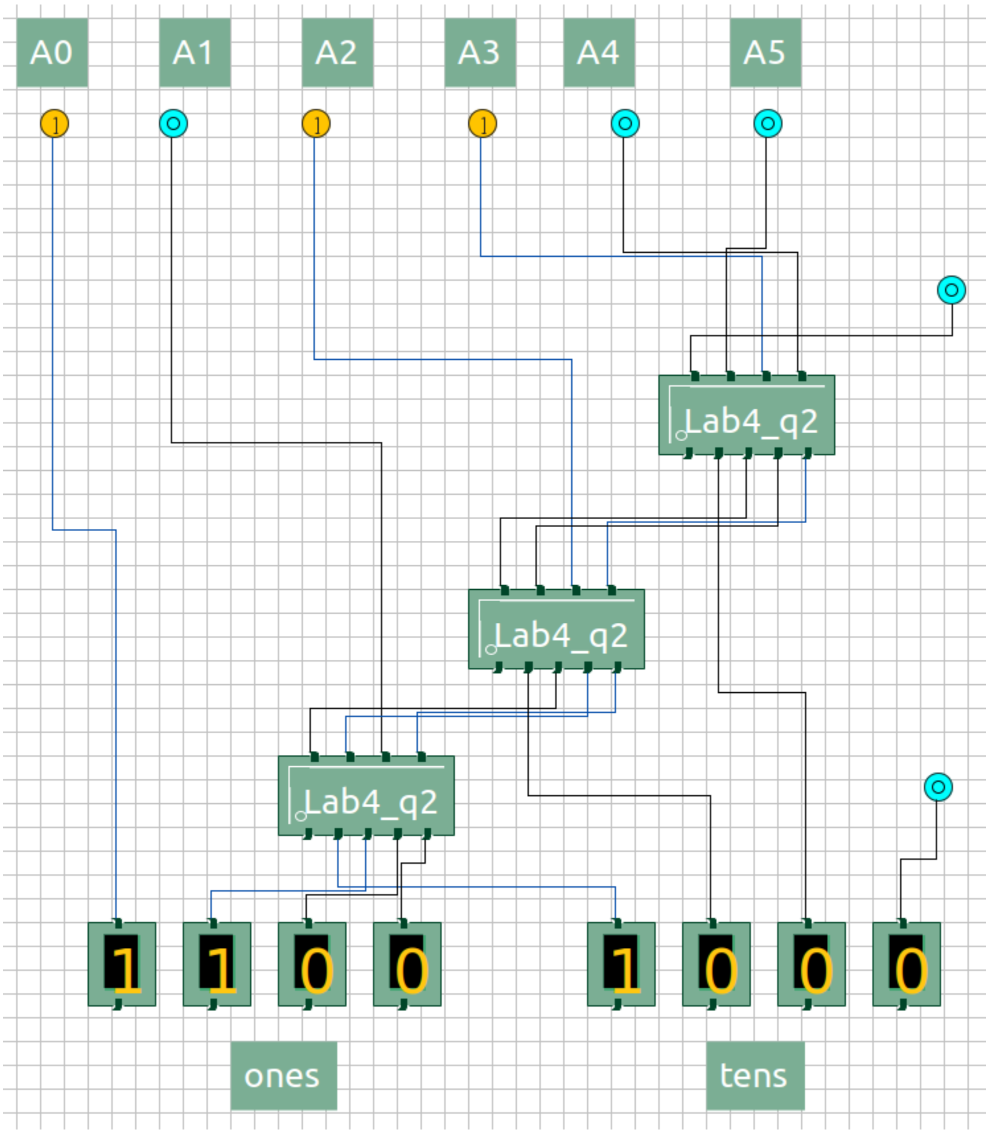


Figure 5: 6-bit binary to BCD convertor Circuit

5 Part 5 (7-bit binary to BCD convertor)

5.1 Problem Statement

1. Using the conditinal adder modules, design a combinational 7-bit binary to BCD convertor.
2. Test that it works by applying appropriate inputs and checking the outputs.
3. Label the terminals to reflect their roles.
4. Save it as a regular circuit (logic file), reopen and retest.

5.2 Solution Description

Let the 7-bit binary number X be $A_6A_5A_4A_3A_2A_1A_0$ and the BCD Value of the binary number X be $B_{11}B_{10}B_9B_8B_7B_6B_5B_4B_3B_2B_1B_0$. In the 7 bit binary to BCD converter we used five units of 3adder.

a) First 3-adder takes A_2, A_1, A_0, 0 bit switch (consider as fixed) as input and its output carry would be ignored as the maximum input value is 7 which becomes 10 after we add 3 to it which is within the range of 4 bits so the carry would remain 0 irrespective of the input, the MSB of sum is redirected to fifth adder which will carry out the add 3 step in tens place if its value exceeds 5 before any shift and the other bits of sum(S_0, S_1, S_2) are redirected as input to the second adder.

b) Second 3 adder takes A_3 and three bits from the first 3 adder as its inputs.This input corresponds to the shift step in the Double Dabble algorithm.Its output carry will be ignored, the MSB of sum is redirected to fifth adder which will carry out the add 3 step in tens place if its value exceeds 5 before any shift and the other bits of sum(S_0, S_1, S_2) are redirected as input to the third adder.

c) Third 3 adder takes A_4 and three bits from the second 3 adder as its inputs.This input corresponds to the second shift step in the Double Dabble algorithm. Its output carry will also be ignored, the MSB of sum is redirected to fifth adder which will carry out athe dd 3 step in tens place if its value exceeds 5 before any shift and the other bits of sum(S_0, S_1, S_2) are redirected as input to the fourth adder.

d) Fourth 3 adder takes A_5 and three bits from the third 3 adder as its inputs.This input corresponds to the second shift step in the Double Dabble algorithm.Its output carry will also be ignored and the sum is redirected to output $B_{10}B_9B_8$ as no further step is involved.

e) Fifth takes the MSBs of output of first, second and third adders along with 0 bit switch (consider as fixed) as input and its output carry will be ignored and the sum is redirected to output $B_7B_6B_5B_4$ as no further step is involved. Finally A_6 is redirected to B_{12} as after this no shift or add 3 step is going to take place.

This completes the Double Dabble Algorithm for conversion of a 7 bit binary to BCD.

5.3 Circuit Diagram

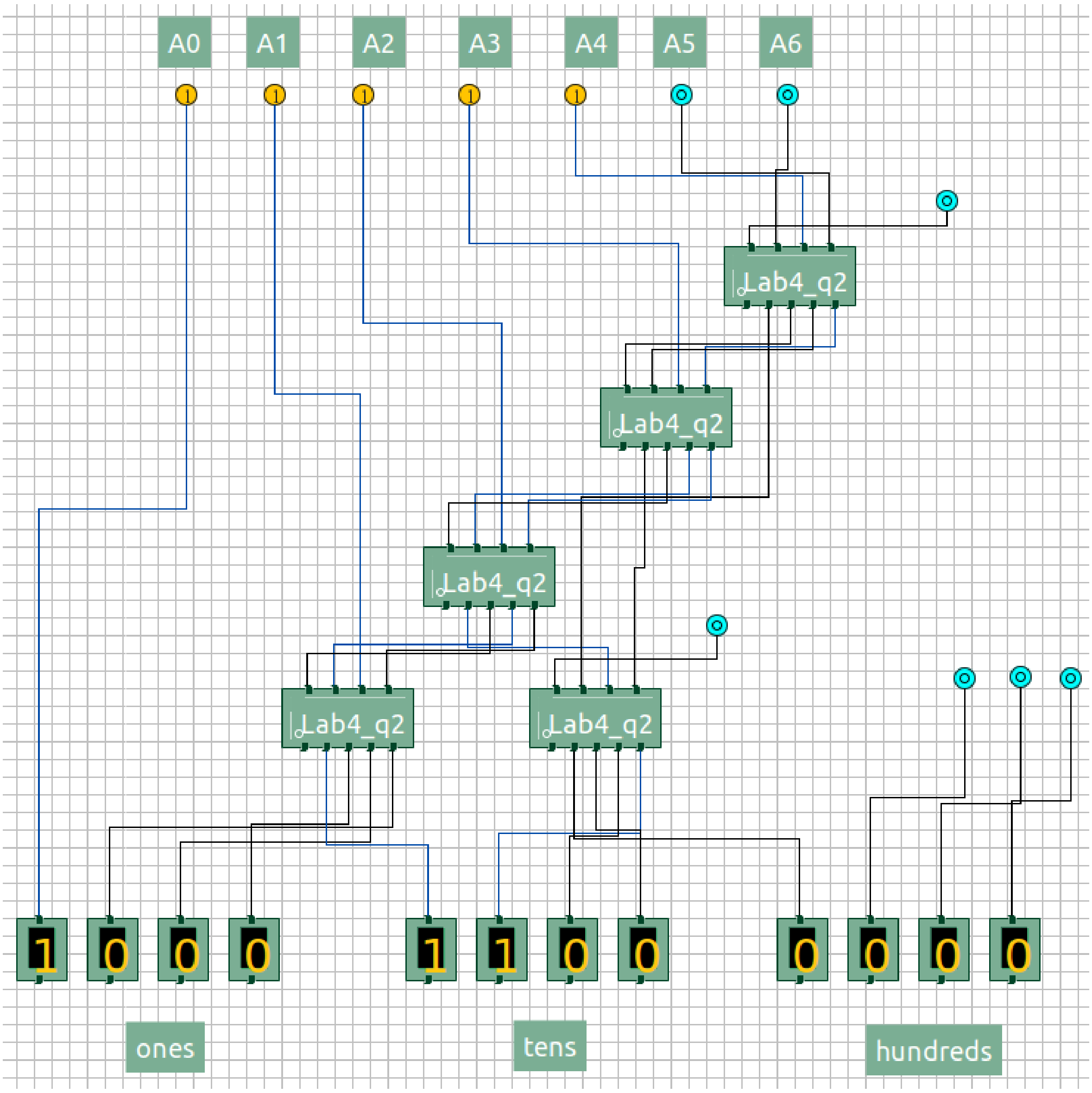


Figure 6: 7-bit binary to BCD convertor Circuit