

Lab 2 Multiplexer

CS211 Summer 2017

June 19, 2017

Xu Mingzhi

Table of Content

1. Objective	4
2. 2:1 Multiplexer.....	5
2.1 Functionality and Specification	5
2.2 Simulation	6
2.3 Demonstration.....	7
3. 2:1 Multiplexer Using NAND gates only	10
3.1 Functionality and Specifications.....	10
3.2 Simulation	13
3.3 Demonstration.....	13
4. 4:1 Multiplexer Using NAND gates only	16
4.1 Functionality and Specifications.....	16
4.2 Simulation	18
4.3 Demonstration.....	19
5. 4:1 Multiplexer using 2:1 MUX as component.....	21
5.1 Functionality and Specifications.....	21
5.2 Simulation	23
5.3 Demonstration.....	24
6. 2:1 (4-Bit) Multiplexer using 2:1 MUX as component.....	27
6.1 Functionality and Specifications.....	27
6.2 Simulation	28
6.3 Demonstration.....	29
7. 5:1 Multiplexer using 2:1 MUX as components	32
7.1 Functionality and Specification	32
7.2 Simulation	33
7.3 Demonstration.....	34
8. 5:1 (2-bit) Multiplexer	37
8.1 Functionality and Specifications.....	37
8.2 Simulation	38
8.3 Demonstration.....	39
9. 2:4 Decoder	43
9.1 Functionality and Specifications.....	43
9.2 Simulation	45

9.3	Demonstration.....	46
10.	3:8 Decoder.....	48
10.1	Functionality and Specifications	48
10.2	Simulation.....	50
10.3	Demonstration.....	50
11.	8:3 Encoder	53
11.1	Functionality and Specification	53
11.2	Simulation.....	55
11.3	Demonstration.....	56
12.	1:2 Demultiplexer	60
12.1	Functionality and Specifications	60
12.2	Simulation.....	62
12.3	Demonstration.....	63
13.	Conclusion	65
14.	Appendix.....	65

1. Objective

In this lab, we will use everything we have learned so far about Quartus Prime in building circuits, verify designed circuit using waveform simulation, and loading designed circuit onto the FPGA DE1-SoC Board for testing. Multiplexers, Decoder, Encoders, and De-multiplexer will be introduced in this lab.

We will be designing the following circuit in this lab:

- a. 2:1 Multiplexer
- b. 2:1 Multiplexer using NAND gates only
- c. 4:1 Multiplexer using NAND gates only
- d. 4:1 Multiplexer using 2:1 Multiplexers as component
- e. 2:1 (4-Bit) Multiplexer using 1-bit 2:1 Multiplexer as component
- f. 5:1 Multiplexer using 1-bit 2:1 Multiplexer as component
- g. 5:1 (2-Bit) Multiplexer
- h. 2:4 Decoder
- i. 3:8 Decoder
- j. 8:3 Encoder
- k. 1:2 Demultiplexer

2. 2:1 Multiplexer

2.1 Functionality and Specification

This is the truth table of a 2:1 multiplexer, it includes two inputs x and y and the selected input s to the output m. When s is 0 the output m will be x and when is 1 the output m will be y.

Selector s	Output m
0	x
1	y

s	x	y	m
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

$$m = (x \text{ AND } (\text{NOT } s)) \text{ OR } (y \text{ AND } s)$$

The Boolean function can also be written as $m = (x * \sim s) + (y * s)$

When the Boolean function is applied to create the block diagram, a NOT gate, two AND gates, and one OR gate will be required to design the circuit for 2:1 multiplexer. As shown in the diagram below, one of the AND gate will take x and s that goes through the NOT gate as inputs and the second AND gate will take y and s as inputs then the OR gate will the result from both AND gates and outputs m.

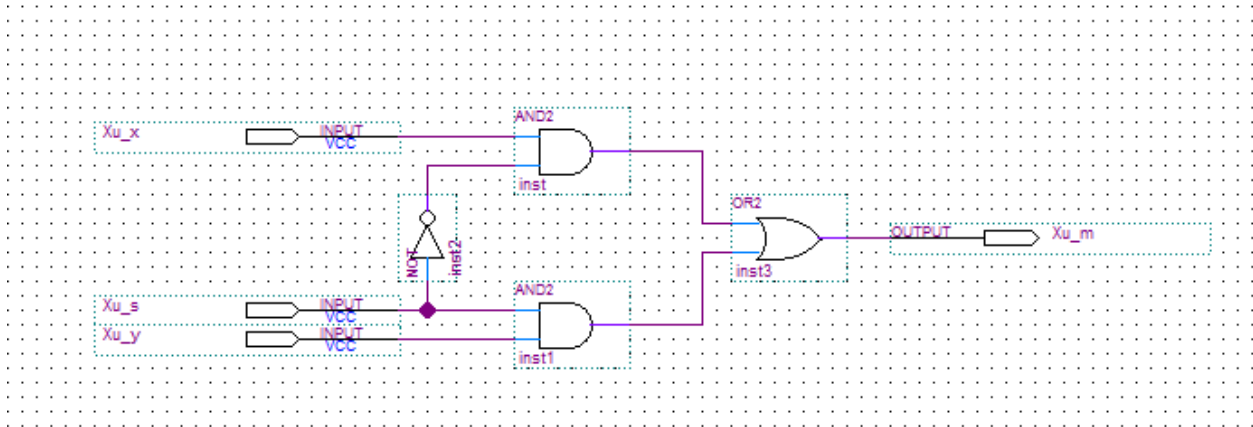


Figure 1: Block diagram of 2:1 multiplexer

The block diagram of 2:1 multiplexer can also be turned into a symbol block.

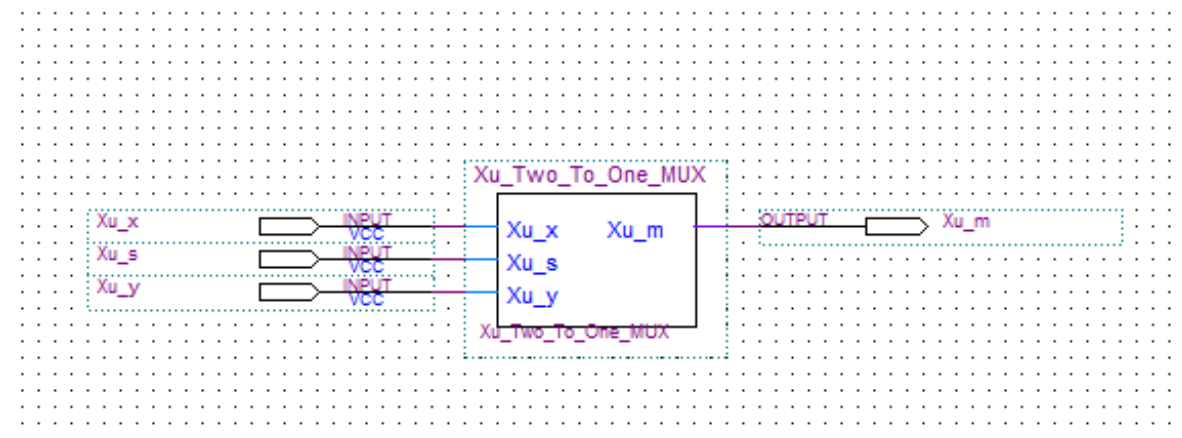


Figure 2: Block diagram of 2:1 multiplexer symbol.

2.2 Simulation

In the simulation, we will give values of 0 and 1 to the inputs at varying intervals of all possible inputs. The selected input *s* will have value of 0 and 1 at each 800-ns interval. Input *x* will have value of 0 and 1 at each 400-ns interval. Input *y* will have value of 0 and 1 at each 200-ns interval.

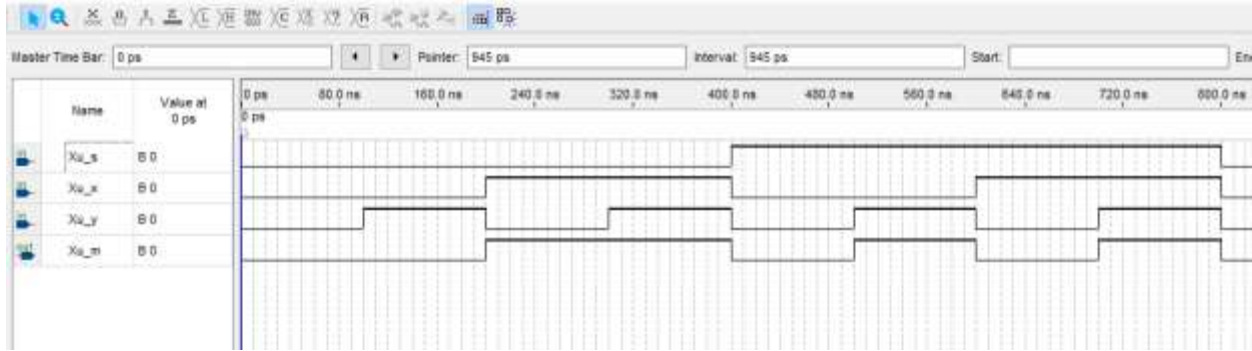


Figure 3: Vector waveform simulation of 2:1 multiplexer.

We can observe that when the selected input s is 0, the output m will have the same as input x .

When the selected input s is 1, the output m will have the same as input y . The results from the simulation corresponds to the truth table of the 2:1 multiplexer and it's Boolean function.

2.3 Demonstration

The PIN assignment of the inputs and outputs on the DE1-SoC Board.

Xu_x is assigned to SW[0] which is PIN_AB12

Xu_y is assigned to SW[1] which is PIN_AC12

Xu_s is assigned to SW[2] which is PIN_AF9

Xu_m is assigned to LEDR[0] which is PIN_V16

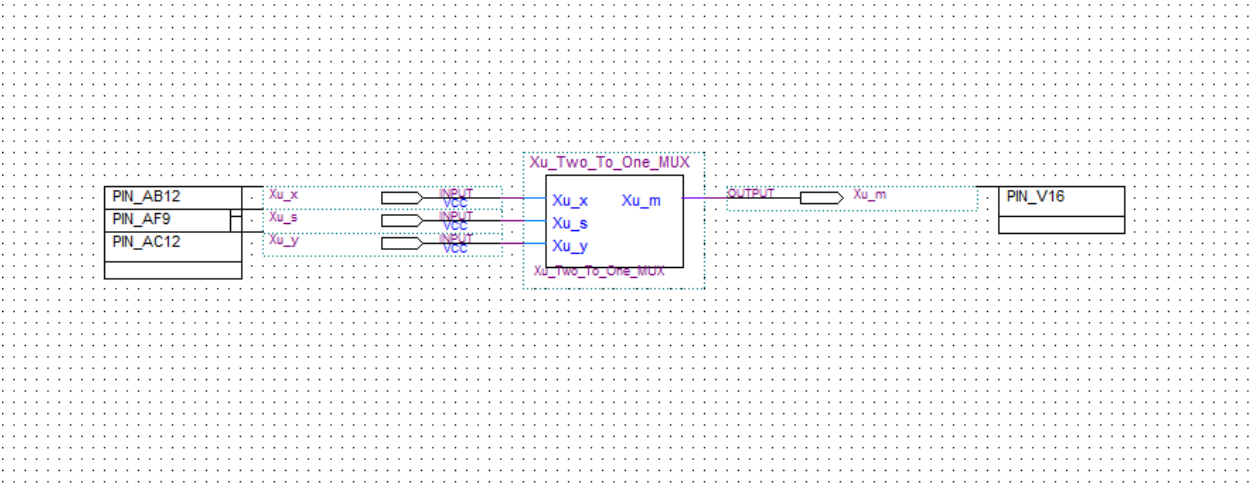


Figure 4: PIN assignments of the circuit to DE1-SoC Board.





Figure 5: PIN Assignments on the DE1-SoC Board.

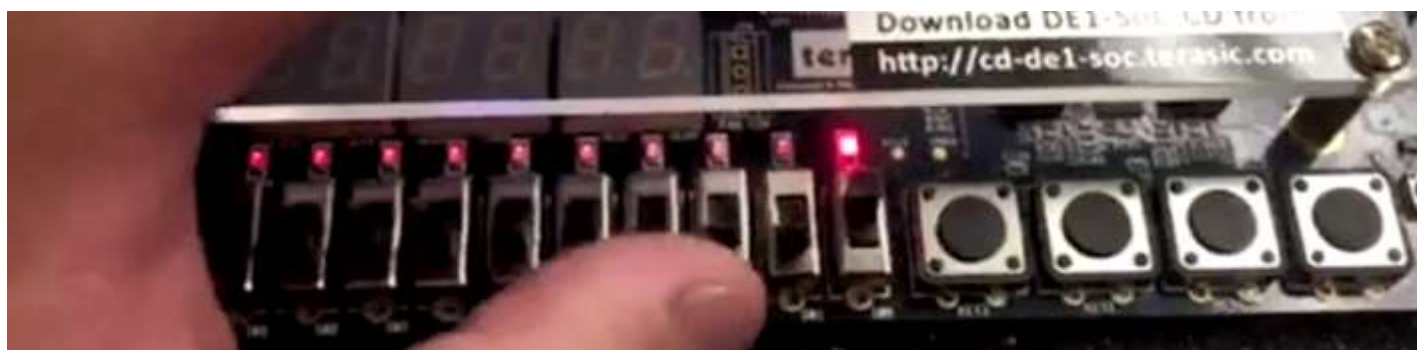


Figure 6: Digital circuit when selector s is 0 (off) input x is 1 (on) outputs m is 1 (on)



Figure 7: Selector s is 1 (on) input y is 1 (on) outputs m is 1 (on)

3. 2:1 Multiplexer Using NAND gates only

3.1 *Functionality and Specifications*

This is a 2:1 multiplexer that only uses NAND logic gates for designing its block diagram, which is different compared to the 2:1 multiplexer that uses NOT, AND, and OR gate to build the block diagram. The block diagram looks different since it uses different gates, but the inputs and outputs should remain the same, therefore the 2:1 multiplexer and 2:1 multiplexer using NAND gates only shares the same truth table and Boolean function. When s is 0 output m will be x , when s is 1 output m will be y .

Selector s	Output m
0	x
1	y

s	x	y	m
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

$$m = (y \text{ NAND } s) \text{ NAND } (x \text{ NAND } (s \text{ NAND } s))$$

1) $y \text{ NAND } s$ Given

2) $(\text{NOT } y) \text{ OR } (\text{NOT } s)$ 1 DeMorgan's Law

3) $s \text{ NAND } s$ Given

4) $(\text{NOT } s) \text{ OR } (\text{NOT } s)$ 3 DeMorgan's Law

5) $\text{NOT } s$ 4 Idempotent Law

6) $x \text{ NAND } (\text{NOT } s)$ Given and 5

7) $(\text{NOT } x) \text{ OR } s$ 6 DeMorgan's Law

8) $((\text{NOT } y) \text{ OR } (\text{NOT } s)) \text{ NAND } ((\text{NOT } x) \text{ OR } s)$ Given and 2, 7

9) $(y \text{ AND } s) \text{ OR } (x \text{ AND } (\text{NOT } s))$ 8 DeMorgan's Law

The Boolean function for 2:1 multiplexer using NAND gates only is

$$m = (y \text{ NAND } s) \text{ NAND } (x \text{ NAND } (s \text{ NAND } s))$$

Since NAND is defined as $\text{NOT}(x \text{ AND } y)$, DeMorgan's Law can be applied which becomes

$$m = \text{NOT}(\text{NOT}(y \text{ AND } s) \text{ AND } \text{NOT}(x \text{ AND } \text{NOT}(s \text{ AND } s)))$$

By following the logical laws shown under the truth table, the Boolean function is manipulated into

$$m = (y \text{ AND } s) \text{ OR } (x \text{ AND } \text{NOT } s)$$

which is the same function that is used for 2:1 multiplexer.

When we apply the original Boolean function of 2:1 multiplexer using NAND gates only, we will need four NAND gates to build the block diagram. There will be one NAND gate that takes selector s as both of its inputs which negates s . There will be one NAND gate that takes y and s as inputs and another NAND gate that takes x and the negation of s as inputs and the last NAND gate takes the results of the two NAND gates and outputs m .

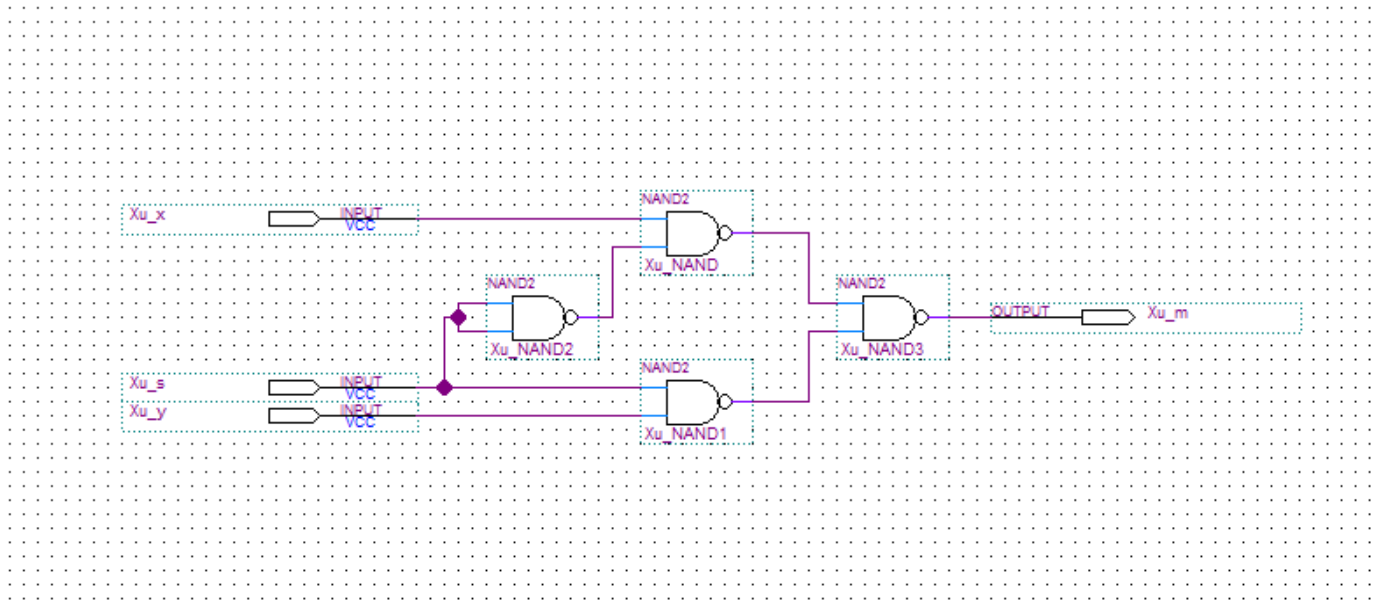


Figure 8: Block diagram of 2:1 multiplexer using NAND gates only.

3.2 Simulation

In the simulation, we will give values of 0 and 1 to the inputs at varying interval, since 2:1 MUX using NAND gates only shares the same truth table and Boolean function as the 2:1 MUX, the inputs and outputs of the simulation should be the same. Selector s will have value of 0 and 1 at each 800-ns interval, input x will have value of 0 and 1 at each 400-ns interval, and input y will have value of 0 and 1 at each 200-ns interval.

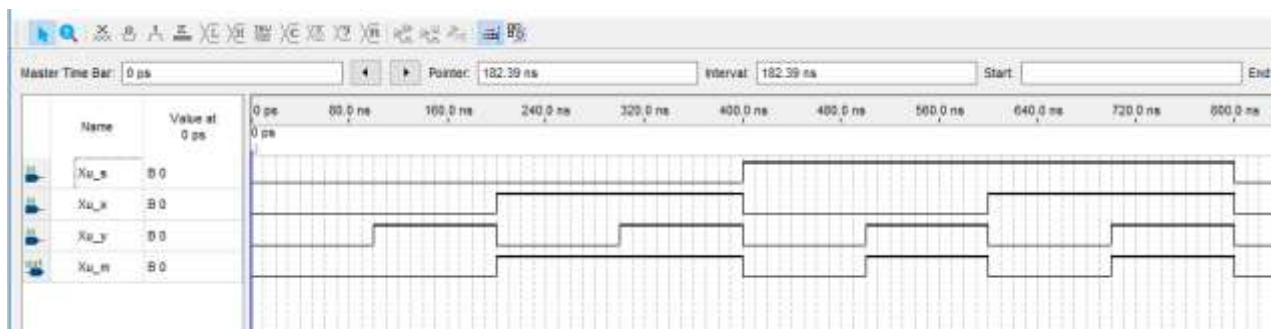


Figure 9: Vector waveform simulation of 2:1 MUX using NAND gates only.

We can observe that the result of the 2:1 MUX using NAND gates only simulation have the same output m as the 2:1 MUX which proves that 2:1 MUX can be built using NAND gates only.

3.3 Demonstration

The PIN assignment of the inputs and outputs on the DE1-SoC Board.

Xu_x is assigned to SW[0] which is PIN_AB12

Xu_y is assigned to SW[1] which is PIN_AC12

Xu_s is assigned to SW[2] which is PIN_AF9

Xu_m is assigned to LEDR[0] which is PIN_V16

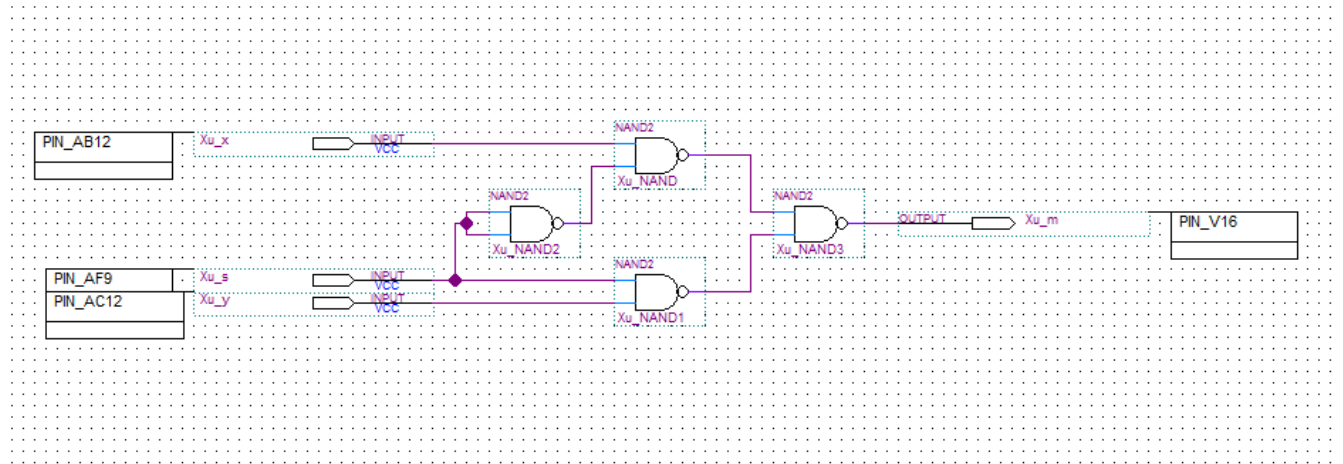


Figure 10: PIN assignment of circuit to DE1-SoC Board.



Figure 11: PIN Assignments on the DE1-SoC Board.

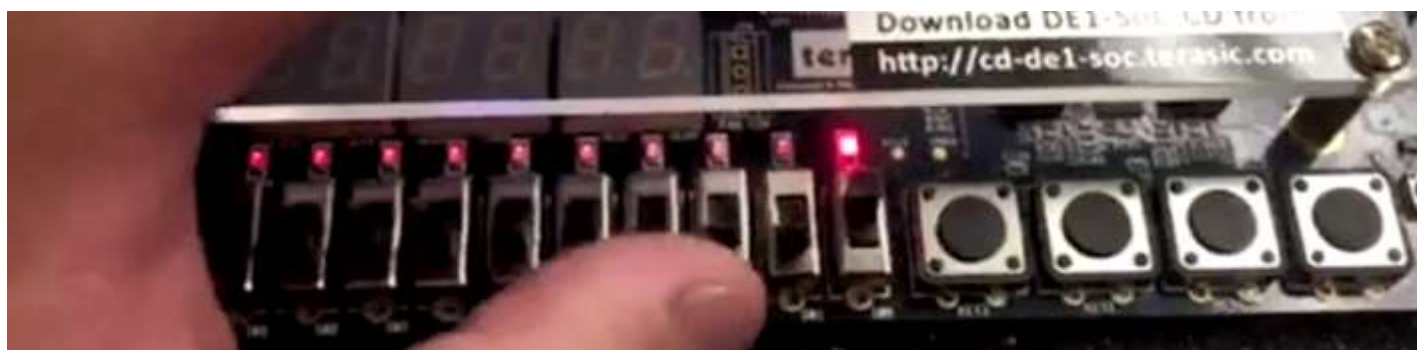


Figure 12: Digital circuit when selector s is 0 (off) input x is 1 (on) outputs m is 1 (on)



Figure 13: Selector s is 1 (on) input y is 1 (on) outputs m is 1 (on)

4. 4:1 Multiplexer Using NAND gates only

4.1 Functionality and Specifications

The 4:1 MUX will have two selectors and four inputs and the selectors will be the ones that determines which input will be output into m. When both selector s1 and s2 is 0 output m will be IN1. When s1 is 0 and s2 is 1, output m will be IN2. When s1 is 1 and s2 is 0, output m will be IN3. When s1 and s2 are 1, output m will be IN4.

s1	s2	m
0	0	IN1
0	1	IN2
1	0	IN3
1	1	IN4

s1	s2	IN1	IN2	IN3	IN4	m
0	0	0	x	x	x	0
0	0	1	x	x	x	1
0	1	x	0	x	x	0
0	1	x	1	x	x	1
1	0	x	x	0	x	0
1	0	x	x	1	x	1
1	1	x	x	x	0	0
1	1	x	x	x	1	1

Since NAND is defined as $\text{NOT}(x \text{ AND } y \dots \text{ AND } z)$ and by DeMorgan's Law it becomes $\text{NOT } x \text{ OR NOT } y \dots \text{ OR NOT } z$

Boolean Function would be

$$m = \text{NOT}(\text{NOT}(\text{IN1 AND NOT}(s1 \text{ AND } s1) \text{ AND NOT}(s2 \text{ AND } s2)) \text{ AND NOT}(\text{IN2 AND NOT}(s1 \text{ AND } s1) \text{ AND } s2) \text{ AND NOT}(\text{IN3 AND } s1 \text{ AND NOT}(s2 \text{ AND } s2)) \text{ AND NOT}(\text{IN4 AND } s1 \text{ AND } s2))$$

By applying the DeMorgan's Law and Idempotent Law the Boolean Function would be

$$m = (\text{IN1 AND NOT } s1 \text{ AND NOT } s2) \text{ OR } (\text{IN2 AND NOT } s1 \text{ AND } s2) \text{ OR } (\text{IN3 AND } s1 \text{ AND NOT } s2) \text{ OR } (\text{IN4 AND } s1 \text{ AND } s2)$$

Which is the same as

$$m = (\text{IN1} * \sim s1 * \sim s2) + (\text{IN2} * \sim s1 * s2) + (\text{IN3} * s1 * \sim s2) + (\text{IN4} * s1 * s2)$$

When the Boolean function is applied to create the block diagram for 4:1 MUX using NAND gates only, seven NAND gates will required to build the circuit. Two of the NAND gate will take s1 and s2 as both inputs which creates the negation of s1 and s2. NAND03 will take IN1, negation s1 and negation s2 as input. NAND04 will take IN2, negation s1 and s1 as inputs. NAND05 will take IN3, s1 and negation of s2 as inputs. NAND06 will take IN4, s1, and s2 as inputs. The last NAND gate NAND07 will take the result from NAND03, NAND04, NAND05, and NAND06 as inputs and outputs m.

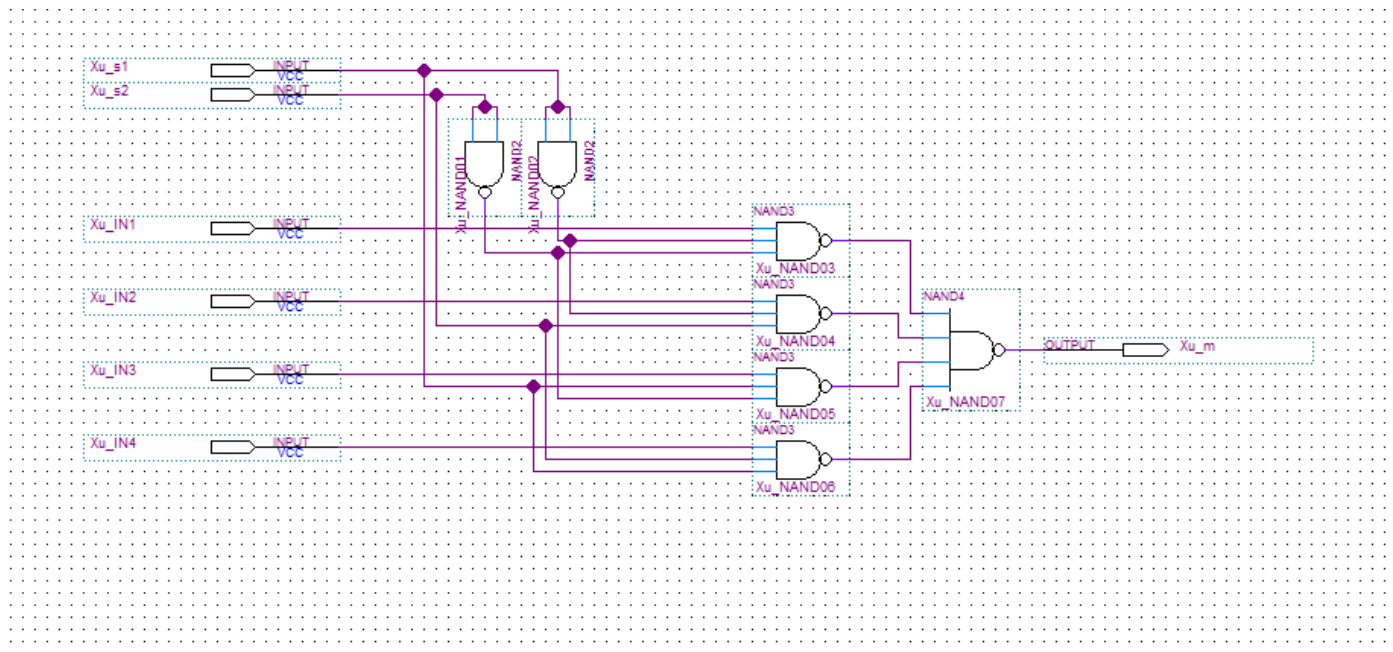


Figure 14: Block diagram of 4:1 multiplexer using NAND gates only.

4.2 Simulation

In the simulation, we will give values of 0 and 1 to the inputs at varying intervals. Selector s1 will have value of 0 and 1 at each 800-ns interval. Selector s2 will have value of 0 and 1 at each 400-ns interval. Input IN1 will have value of 0 and 1 at each 200-ns interval. Input IN2 will have

value of 0 and 1 at each 100-ns interval. Input IN3 will have value of 0 and 1 at each 50-ns interval. Input IN4 will have value of 0 and 1 at each 25-ns interval.

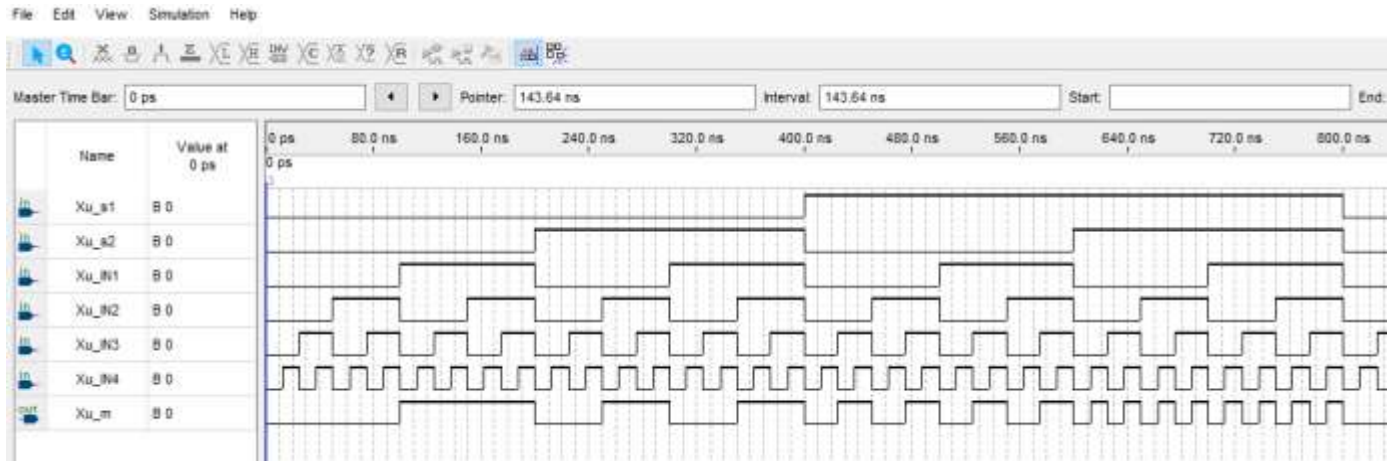


Figure 15: Vector waveform simulation of 4:1 MUX using NAND gates only.

We can observe that when selector s1 and s2 is 0, output m will be IN1. When s1 is 0 and s2 is 1, output m will be IN2. When s1 is 1 and s2 is 0, output m will be IN3. When both s1 and s2 are 1, output m will be IN4. The results correspond to the expected outputs and the truth table of 4:1 MUX.

4.3 Demonstration

The PIN assignment of inputs and outputs on the DE1-SoC Board.

Xu_IN1 is assigned to SW[0] which is PIN_AB12

Xu_IN2 is assigned to SW[1] which is PIN_AC12

Xu_IN3 is assigned to SW[3] which is PIN_AF9

Xu_IN4 is assigned to SW[4] which is PIN_AF10

Xu_s1 is assigned to SW[5] which is PIN_AD11

Xu_s2 is assigned to SW[6] which is PIN_AD12

Xu_m is assigned to LEDR[0] which is PIN_V16

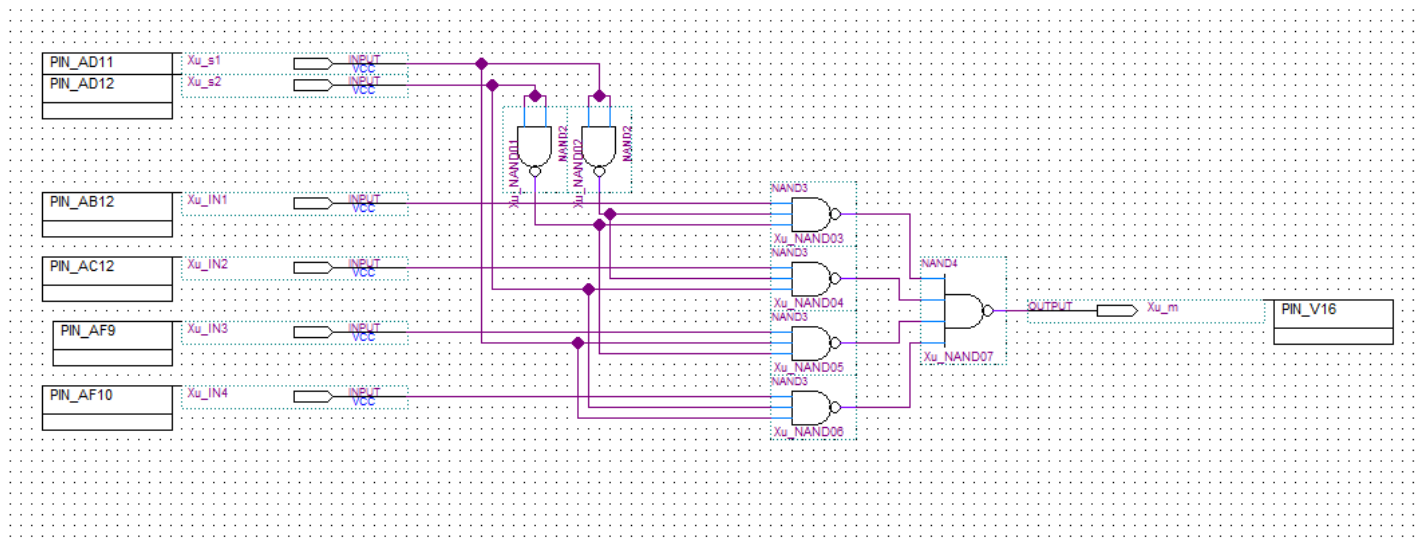


Figure 16: PIN assignment of 4:1 MUX circuit to the DE1-SoC Board.



Figure 17: PIN assignment on the DE1-SoC Board.



Figure 18: s1 and s2 is 0 (off) IN1 is 1 (on) outputs m is 1 (on).

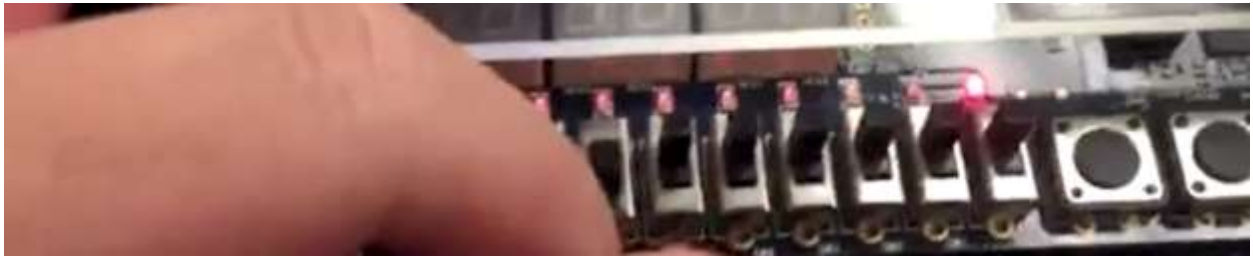


Figure 19: s1 and s2 is 1 (on) IN4 is 1 (on) outputs m is 1 (on).

5. 4:1 Multiplexer using 2:1 MUX as component

5.1 Functionality and Specifications

The functionality and specifications is the same as 4:1 MUX except for this time we will be creating a 4:1 MUX using 2:1 MUX as components. The design of the circuit is different from 4:1 MUX using NAND gates only but if the input IN2 is changed place with input IN3 then the truth table and Boolean function would completely the same as the one in 4:1 MUX. There will

also be two selectors and five inputs and an output m.

s1	s2	m
0	0	IN1
0	1	IN3
1	0	IN2
1	1	IN4

s1	s2	IN1	IN2	IN3	IN4	m
0	0	0	x	x	x	0
0	0	1	x	x	x	1
0	1	x	x	0	x	0
0	1	x	x	1	x	1
1	0	x	0	x	x	0
1	0	x	1	x	x	1
1	1	x	x	x	0	0
1	1	x	x	x	1	1

$$m = (((IN1 \text{ AND NOT } s1) \text{ OR } (IN2 \text{ AND } s1)) \text{ AND NOT } s2) \text{ OR } (((IN3 \text{ AND NOT } s1) \text{ OR } (IN4 \text{ AND } s1)) \text{ AND } s2)$$

When we apply the Boolean function to the block diagram, three 2:1 MUX will require to build the block diagram for 4:1 MUX using 2:1 MUX as components. The 2:1 MUX01 will take IN1, IN2 and s1 as inputs. The 2:1 MUX02 will take IN3, IN4 and s1 as inputs. The 2:1 MUX03 will take results from 2:1 MUX01 and 2:1 MUX02 and s2 as inputs which then outputs m.

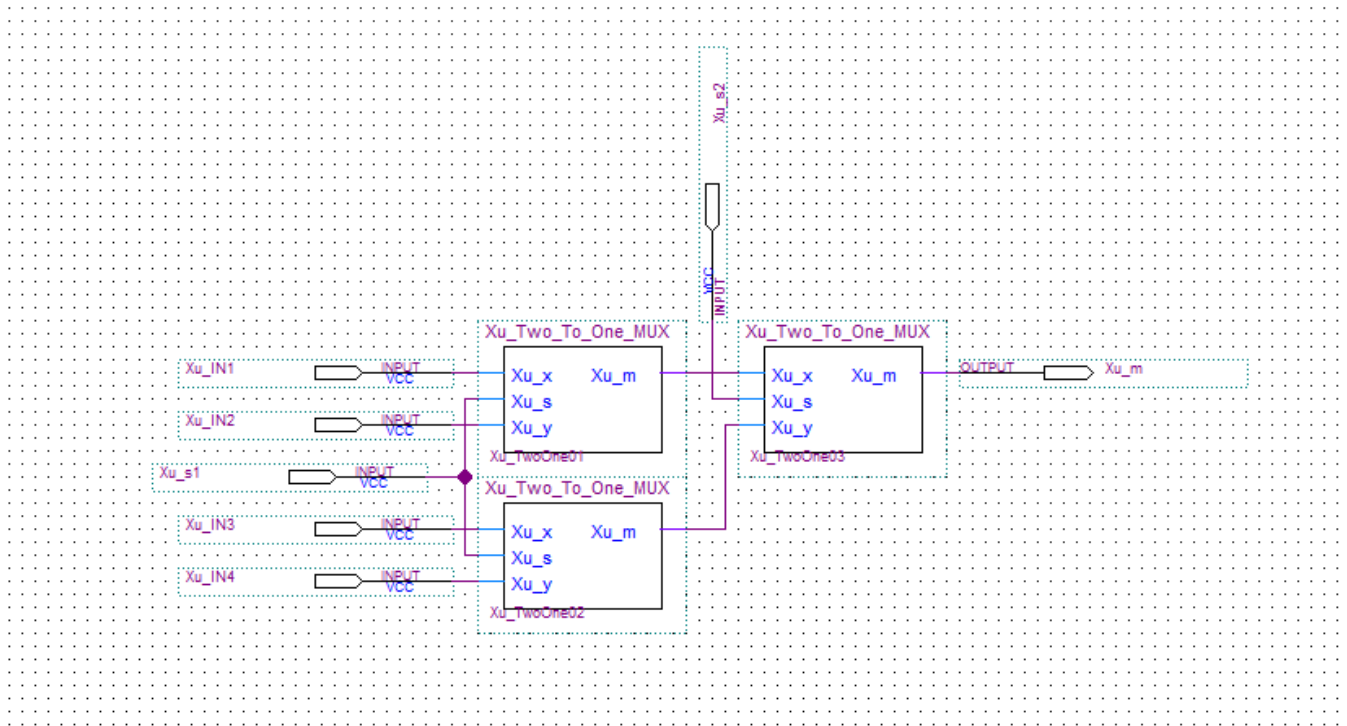


Figure 20: Block diagram of 4:1 MUX using 2:1 MUX as components.

5.2 Simulation

In the simulation, we will give values of 0 and 1 to the inputs at varying intervals. Selector s1 will have value of 0 and 1 at each 800-ns interval. Selector s2 will have value of 0 and 1 at each 400-ns interval. Input IN1 will have value of 0 and 1 at each 200-ns interval. Input IN2 will have value of 0 and 1 at each 100-ns interval. Input IN3 will have value of 0 and 1 at each 50-ns interval. Input IN4 will have value of 0 and 1 at each 25-ns interval.

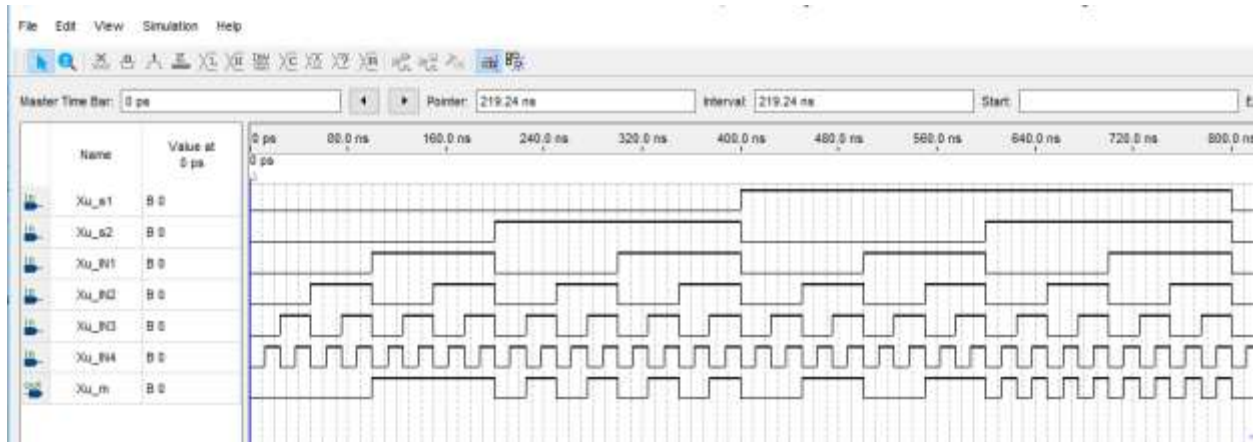


Figure 21: Vector waveform simulation of 4:1 MUX using 2:1 MUX as component.

We can observe that when selector s1 and s2 is 0, output m will be IN1. When s1 is 0 and s2 is 1, output m will be IN3. When s1 is 1 and s2 is 0, output m will be IN2. When both s1 and s2 are 1, output m will be IN4. The results correspond to the expected outputs and the truth table which proves that 4:1 MUX can be designed by using 2:1 MUX as components.

5.3 Demonstration

The PIN assignment of inputs and outputs on the DE1-SoC Board.

Xu_IN1 is assigned to SW[0] which is PIN_AB12

Xu_IN2 is assigned to SW[1] which is PIN_AC12

Xu_IN3 is assigned to SW[3] which is PIN_AF9

Xu_IN4 is assigned to SW[4] which is PIN_AF10

Xu_s1 is assigned to SW[5] which is PIN_AD11

Xu_s2 is assigned to SW[6] which is PIN_AD12

Xu_m is assigned to LEDR[0] which is PIN_V16

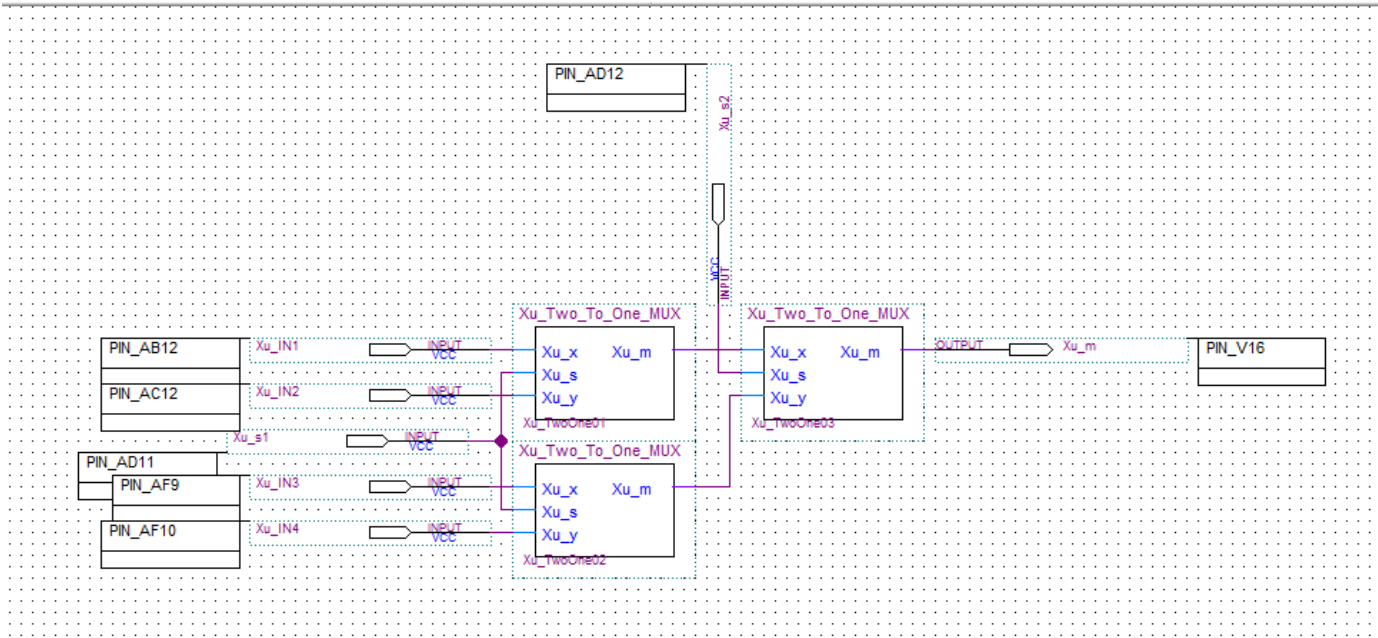


Figure 22: PIN assignment of 4:1 MUX circuit to the DE1-SoC Board.

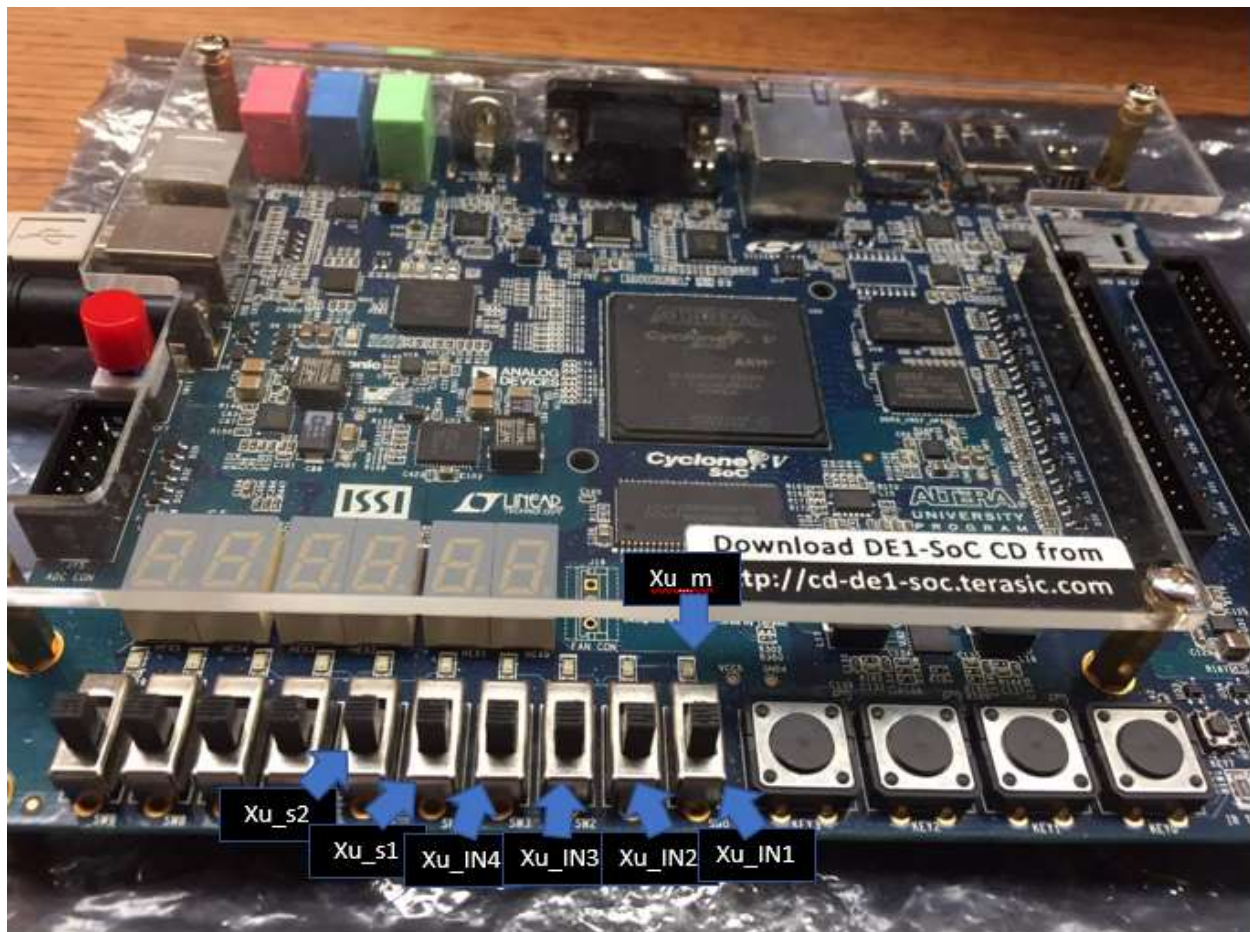


Figure 23: PIN assignment on the DE1-SoC Board.



Figure 24: s1 and s2 is 0 (off) IN1 is 1 (on) outputs m is 1 (on).



Figure 25: s1 and s2 is 1 (on) IN4 is 1 (on) outputs m is 1 (on).

6. 2:1 (4-Bit) Multiplexer using 2:1 MUX as component

6.1 Functionality and Specifications

This time we will be designing a 2:1 MUX that is 4-bit which uses the 1-bit 2:1 MUX as components. There will be a selector input s and total of 8 inputs $x_0, x_1, x_2, x_3, y_0, y_1, y_2, y_3$, and 4 outputs m_0, m_1, m_2, m_3 . It is same as the 1-bit 2:1 MUX except for there is 8 inputs and 4 outputs in this design.

s	m_0	m_1	m_2	m_3
0	x_0	x_1	x_2	x_3
1	y_0	y_1	y_2	y_3

The Boolean functions are:

$$m_0 = (x_0 \text{ AND NOT } s) \text{ OR } (y_0 \text{ AND } s)$$

$$m_1 = (x_1 \text{ AND NOT } s) \text{ OR } (y_1 \text{ AND } s)$$

$$m_2 = (x_2 \text{ AND NOT } s) \text{ OR } (y_2 \text{ AND } s)$$

$$m_3 = (x_3 \text{ AND NOT } s) \text{ OR } (y_3 \text{ AND } s)$$

When we apply the Boolean function to build the block diagram for 4-bit 2:1 MUX, we will need four 1-bit 2:1 MUX as components. Each 1-bit 2:1 MUX component will take the selector s and each of x and y and output its m .

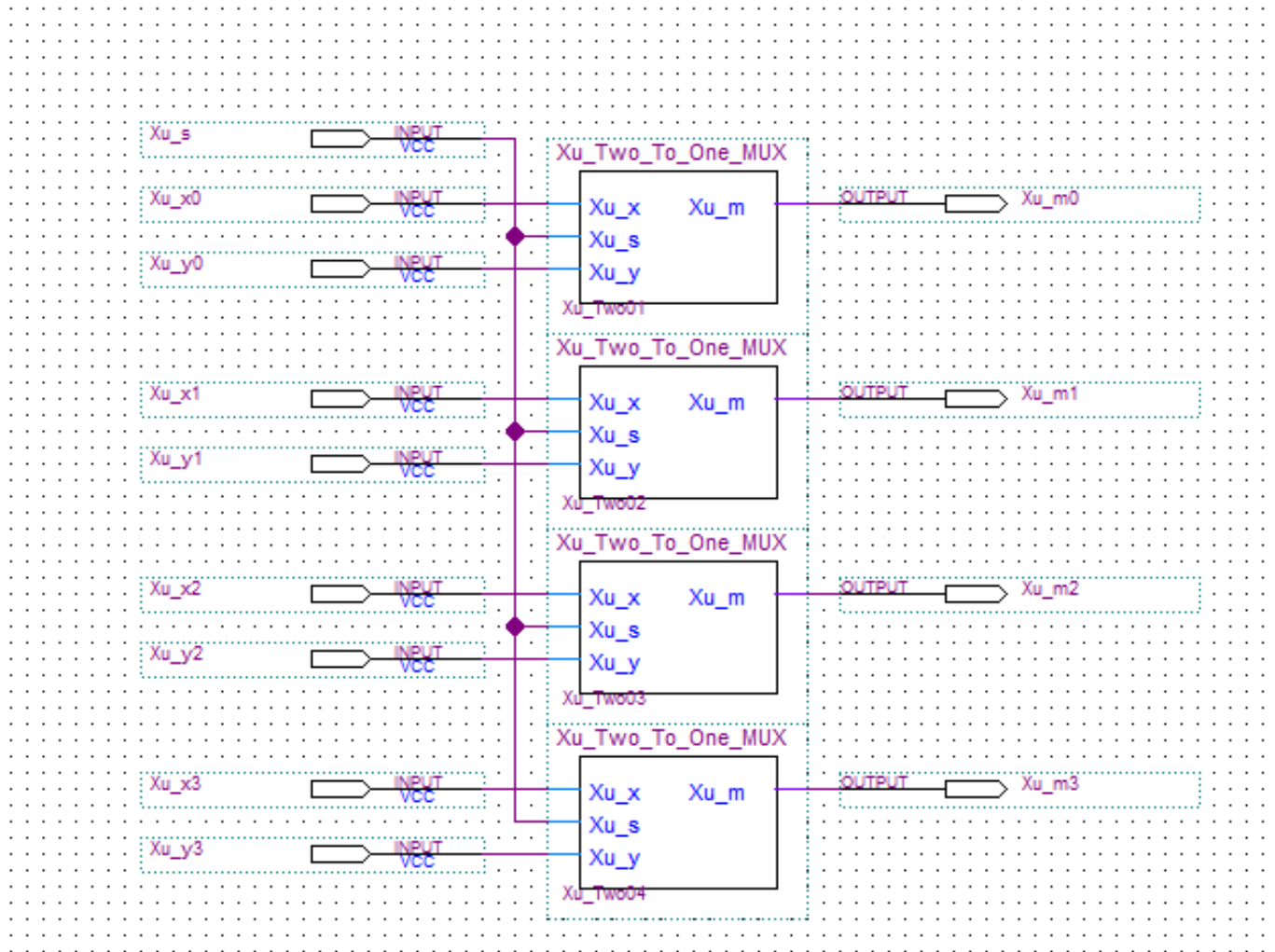


Figure 26: Block diagram of 4-bit 2:1 MUX using 1-bit 2:1 MUX as component.

6.2 Simulation

In the simulation, we will give values of 0 and 1 to the inputs at varying intervals. Selector s will have value of 0 and 1 at each 800-ns interval. Input $x0$, $x1$, $x2$, $x3$ will have value of 0 and 1 at each 400-ns interval. Input $y0$, $y1$, $y2$, $y3$ will have value of 0 and 1 at each 200-ns interval.

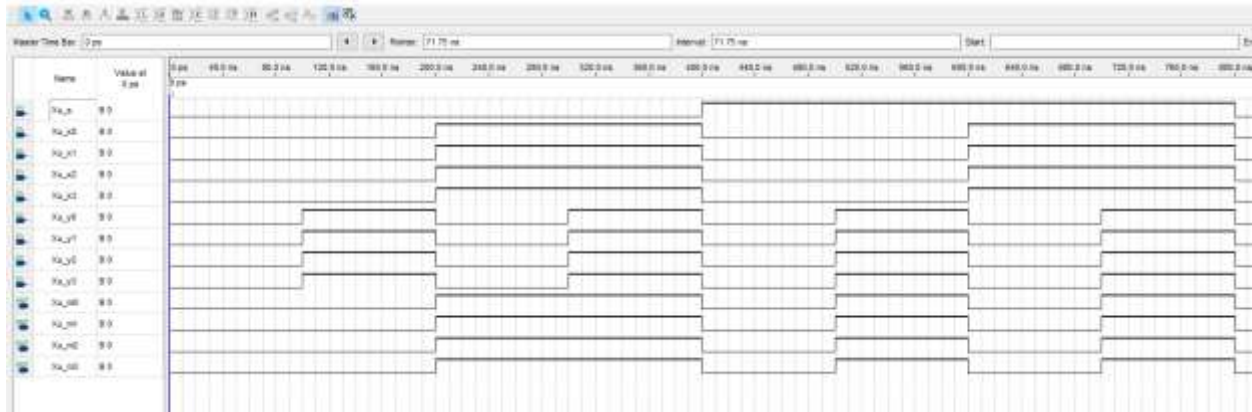


Figure 27: Vector waveform simulation of 4-bit 2:1 MUX

6.3 Demonstration

The PIN assignments of inputs and outputs onto the DE1-SoC Board.

Xu_x0 is assigned to SW[0] which is PIN_AB12

Xu_x1 is assigned to SW[1] which is PIN_AC12

Xu_x2 is assigned to SW[2] which is PIN_AF9

Xu_x3 is assigned to SW[3] which is PIN_AF10

Xu_y0 is assigned to SW[4] which is PIN_AD11

Xu_y1 is assigned to SW[5] which is PIN_AD12

Xu_y2 is assigned to SW[6] which is PIN_AE11

Xu_y3 is assigned to SW[7] which is PIN_AC9

Xu_s is assigned to SW[8] which is PIN_AD10

Xu_m0 is assigned to LEDR[0] which is PIN_V16

Xu_m1 is assigned to LEDR[1] which is PIN_W16

Xu_m2 is assigned to LEDR[2] which is PIN_V17

Xu_m3 is assigned to LEDR[3] which is PIN_V18

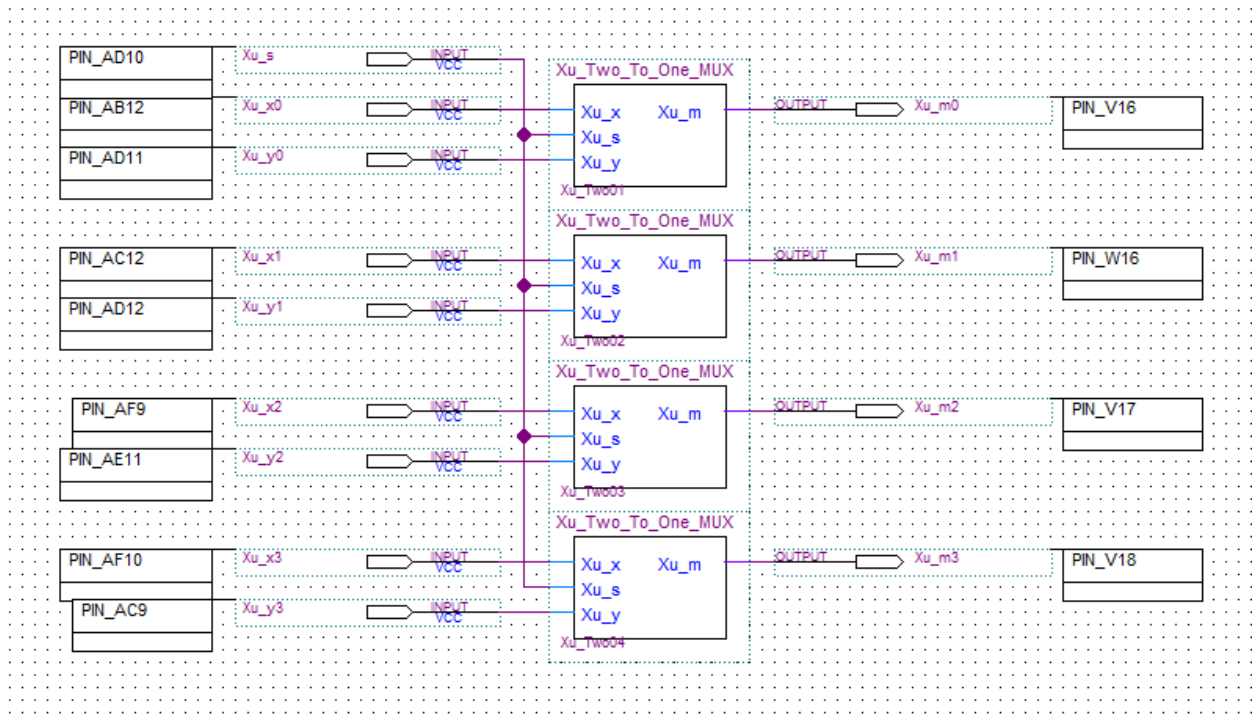


Figure 28: PIN assignment of 4-bit 2: 1 MUX circuit onto the DE1-SoC Board.



Figure 29: PIN assignment on the DE1-SoC Board.

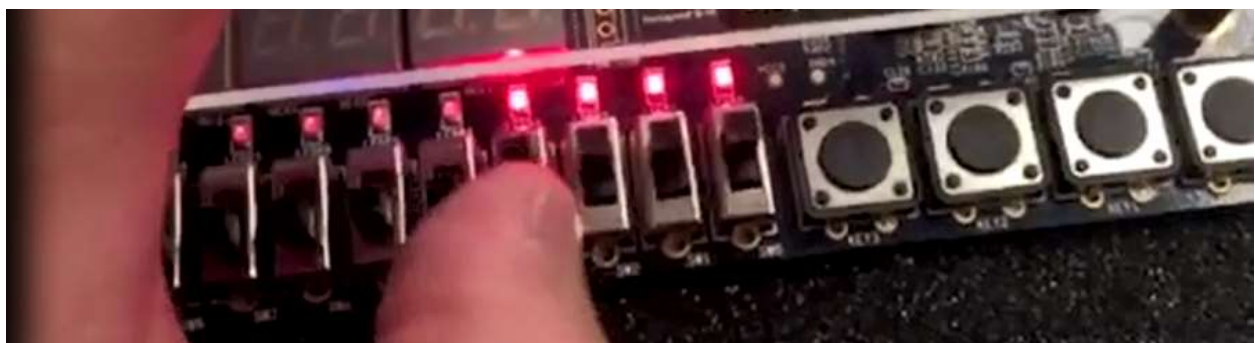


Figure 30: Selector s is 0 (off) all x are 1 (on) output all m are 1 (on).



Figure 31: Selector s is 1 (on) all y are 1 (on) output all m are 1 (on).

7. 5:1 Multiplexer using 2:1 MUX as components

7.1 Functionality and Specification

The functionality and specification of a 5:1 multiplexer is more complex than other multiplexers such as the 2:1 MUX or 4:1 since it will have three selectors and five inputs and an output m.

s1	s2	s3	m
0	0	0	IN1
0	0	1	IN5
0	1	0	IN3
0	1	1	IN5
1	0	0	IN2
1	0	1	IN5
1	1	0	IN4
1	1	1	IN5

The Boolean function is:

$$\text{MUX1} = (\text{IN1 AND NOT } s1) \text{ OR } (\text{IN2 AND } s1)$$

$$\text{MUX2} = (\text{IN3 AND NOT } s1) \text{ OR } (\text{IN4 AND } s1)$$

$$\text{MUX3} = (\text{MUX1 AND NOT } s2) \text{ OR } (\text{MUX2 AND } s2)$$

$$m = (\text{MUX3 AND NOT } s3) \text{ OR } (\text{IN5 AND } s3)$$

When we apply the Boolean function to build the block diagram for 5:1 MUX using 1-bit 2:1 MUX as component, four 1-bit 2:1 MUX will be needed to design the circuit. MUX1 will take IN1, IN2 and s1 as inputs. MUX2 will take IN3, IN4 and s as inputs. MUX3 will s2 and the results of MUX1 and MUX2 as inputs. Lastly MUX4 will take IN5, s3 and result of MUX3 as inputs and produce the output m.

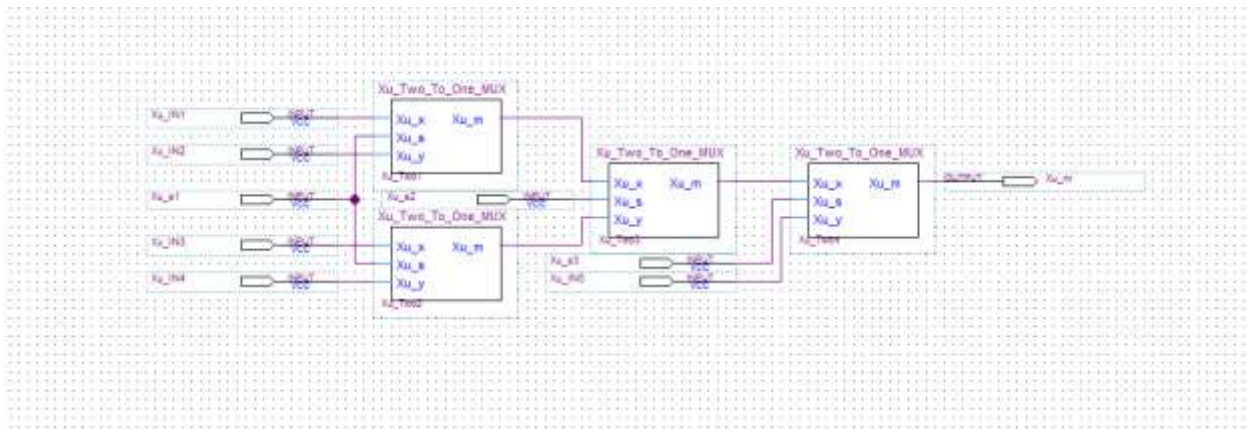


Figure 32: Block diagram of 5:1 MUX using 2:1 MUX as component.

7.2 Simulation

In the simulation, we will give values of 0 and 1 to the inputs at varying intervals. Selector s1 will have value of 0 and 1 at 960-ns interval. Selector s2 will have value of 0 and 1 at 480-ns interval. Selector s3 will have value of 0 and 1 at 240-ns interval. Input IN1 will have value of 0 and 1 at 120-ns interval. Input IN2 will have value of 0 and 1 at 60-ns interval. Input IN3 will have value of 0 and 1 at 30ns-interval. Input IN4 will have value of 0 and 1 at 15-ns interval. Input IN5 will have value of 0 and 1 at 7.5-ns interval.

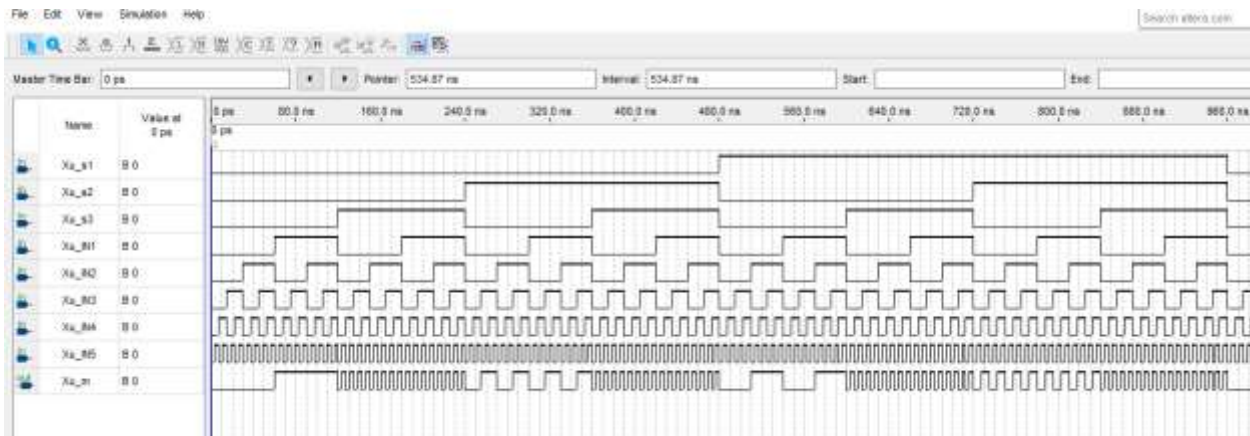


Figure 33: Vector waveform simulation of 5:1 MUX using 2:1 MUX as component.

We can observe that when s1, s2, and s3 are 0, output m will be IN1. When s1 is 1 and both s2 and s3 are 0, output m will be IN2. When s1 and s3 are both 0 and s2 is 1, output m will be IN3. When both s1 and s2 are 1 and s3 is 0, output m will be IN4. When s3 is 1 and whatever s2 or s3 might be, output m will always be IN5.

7.3 Demonstration

The PIN assignments of inputs and outputs onto the DE1-SoC Board.

Xu_IN1 is assigned to SW[0] which is PIN_AB12

Xu_IN2 is assigned to SW[1] which is PIN_AC12

Xu_IN3 is assigned to SW[2] which is PIN_AF9

Xu_IN4 is assigned to SW[3] which is PIN_AF10

Xu_IN5 is assigned to SW[4] which is PIN_AD11

Xu_s1 is assigned to SW[5] which is PIN_AD12

Xu_s2 is assigned to SW[6] which is PIN_AE11

Xu_s3 is assigned to SW[7] which is PIN_AC9

Au_in is assigned to LEDR[0] which is PIN_v10

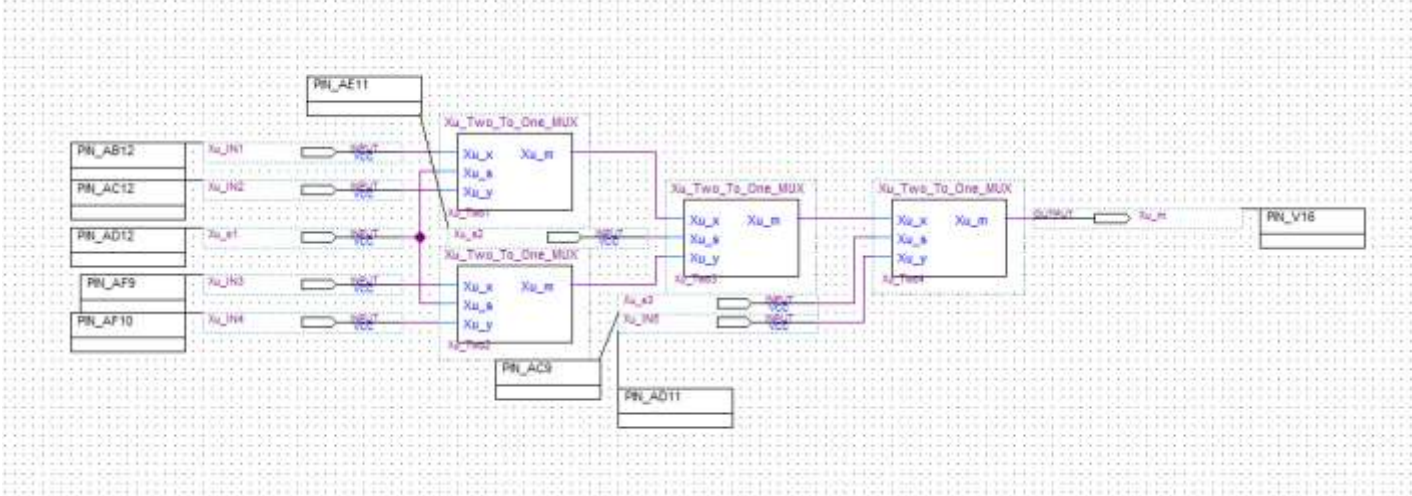


Figure 34: PIN assignment of circuit onto the DE1-SoC Board.

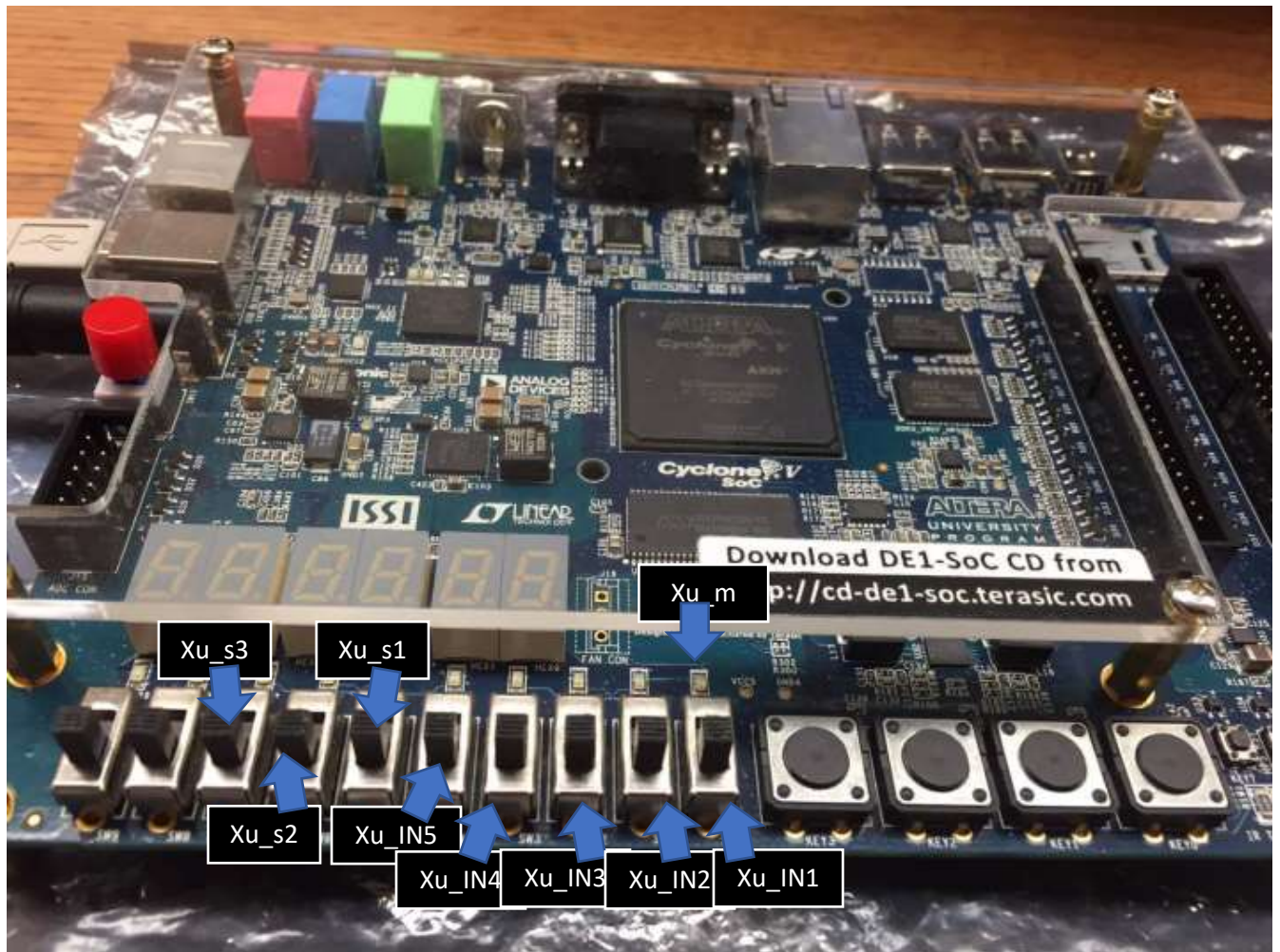


Figure 35: PIN assignments on the DE1-SoC Board.



Figure 36: s1, s2, and s3 is 0 (off) IN1 is 1 (on) outputs m is 1 (on).



Figure 37: s1 and s2 is 0 (off) s3 is 1 (on) IN5 is 1 (on) outputs m is 1 (on).

8. 5:1 (2-bit) Multiplexer

8.1 Functionality and Specifications

The functionality and specification of the 2-bit 5:1 MUX is similar to the 5:1 MUX except this time there will be two outputs meaning there will be ten inputs of x and y and three selector inputs meaning a total of thirteen inputs. The way how this is going to be built is similar to the way in the 4-bit 2:1 MUX.

s1	s2	s3	m1	m2
0	0	0	IN1	IN6
0	0	1	IN5	IN10
0	1	0	IN3	IN8
0	1	1	IN5	IN10
1	0	0	IN2	IN7
1	0	1	IN5	IN10
1	1	0	IN4	IN9
1	1	1	IN5	IN10

The Boolean function are:

$$m1 = (((IN1 \text{ AND NOT } s1) \text{ OR } (IN2 \text{ AND } s1)) \text{ AND NOT } s2) \text{ OR } (((IN3 \text{ AND NOT } s1) \text{ OR } (IN4 \text{ AND } s1)) \text{ AND } s2) \text{ AND NOT } s3) \text{ OR } (IN5 \text{ AND } s3)$$

$$m2 = (((IN6 \text{ AND NOT } s1) \text{ OR } (IN7 \text{ AND } s1)) \text{ AND NOT } s2) \text{ OR } (((IN8 \text{ AND NOT } s1) \text{ OR } (IN9 \text{ AND } s1)) \text{ AND } s2) \text{ AND NOT } s3) \text{ OR } (IN10 \text{ AND } s3)$$

When we apply the Boolean function to build the block diagram for 2-bit 5:1 MUX, we can use two blocks of 1-bit 5:1 MUX as components. The first 1-bit 5:1 MUX takes IN1, IN2, IN3, IN4, IN5, s1, s2, s3 as inputs and outputs m1. The second 1-bit 5:1 MUX takes IN6, IN7, IN8, IN9, IN10, s1, s2, s3 as inputs and outputs m2.

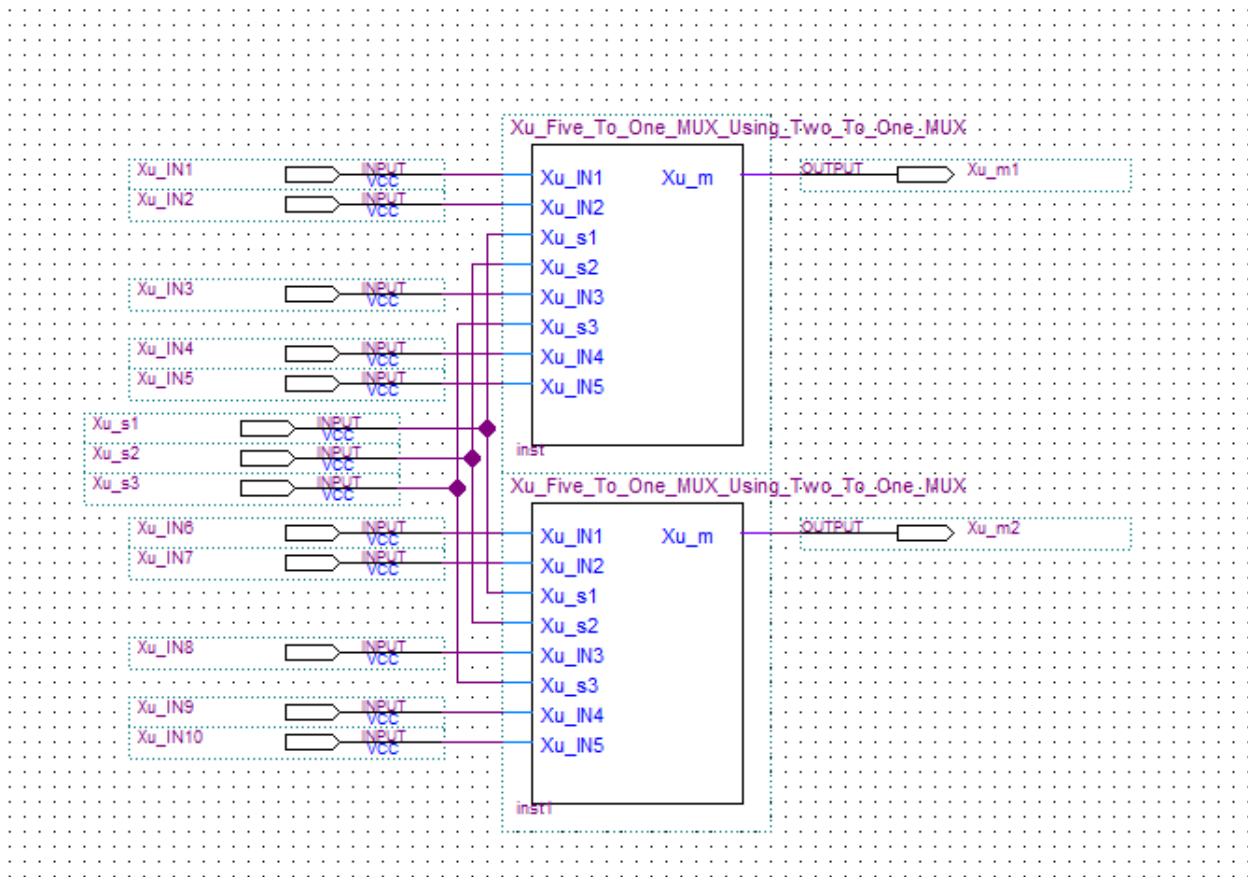


Figure 38: Block diagram of 2-bit 5:1 MUX

8.2 Simulation

In the simulation, we will give values of 0 and 1 to the inputs at varying intervals. Selector s1 will have value of 0 and 1 at 960-ns interval. Selector s2 will have value of 0 and 1 at 480-ns

interval. Selector s3 will have value of 0 and 1 at 240-ns interval. Input IN1 and IN6 will have value of 0 and 1 at 120-ns interval. Input IN2 and IN7 will have value of 0 and 1 at 60-ns interval. Input IN3 and IN8 will have value of 0 and 1 at 30ns-interval. Input IN4 and IN9 will have value of 0 and 1 at 15-ns interval. Input IN5 and IN10 will have value of 0 and 1 at 7.5-ns interval.

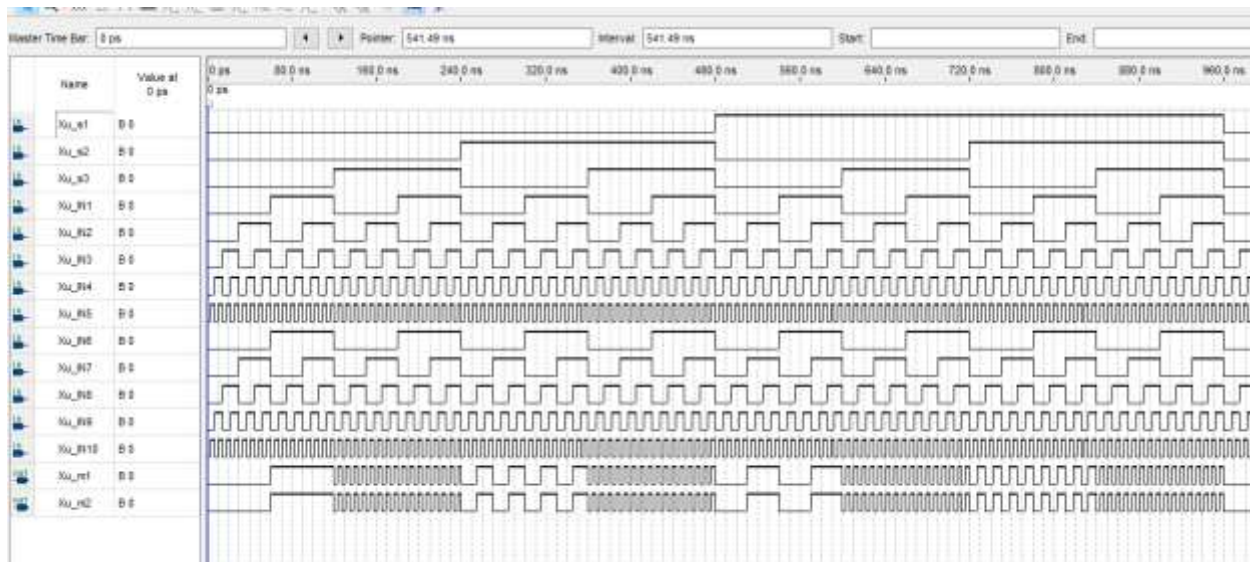


Figure 39: Vector waveform simulation for 2-bit 5:1 MUX.

We can observe that when s1, s2, and s3 are 0, output m1 will be IN1 and m2 will be IN6. When s1 is 1 and both s2 and s3 are 0, output m1 will be IN2 and m2 will be IN7. When s1 and s3 are both 0 and s2 is 1, output m1 will be IN3 and m2 will be IN8. When both s1 and s2 are 1 and s3 is 0, output m1 will be IN4 and m2 will be IN9. When s3 is 1 and whatever s2 or s3 might be, output m1 will always be IN5 and m2 will always be IN10.

8.3 Demonstration

The PIN assignments of inputs and outputs onto the DE1-SoC Board.

Xu_IN1 is assigned to SW[0] which is PIN_AB12

Xu_IN2 is assigned to SW[1] which is PIN_AC12

Xu_IN3 is assigned to SW[2] which is PIN_AF9

Xu_IN4 is assigned to SW[3] which is PIN_AF10

Xu_IN5 is assigned to SW[4] which is PIN_AD11

Xu_IN6 is assigned to SW[5] which is PIN_AD12

Xu_IN7 is assigned to SW[6] which is PIN_AE11

Xu_IN8 is assigned to SW[7] which is PIN_AC9

Xu_IN9 is assigned to SW[8] which is PIN_AD10

Xu_IN10 is assigned to SW[9] which is PIN_AE12

Xu_s1 is assigned to KEY[0] which is PIN_AA14

Xu_s2 is assigned to KEY[1] which is PIN_AA15

Xu_s3 is assigned to KEY[2] which is PIN_W15

Xu_m1 is assigned to LEDR[0] which is PIN_V16

Xu_m2 is assigned to LEDR[1] which is PIN_W16

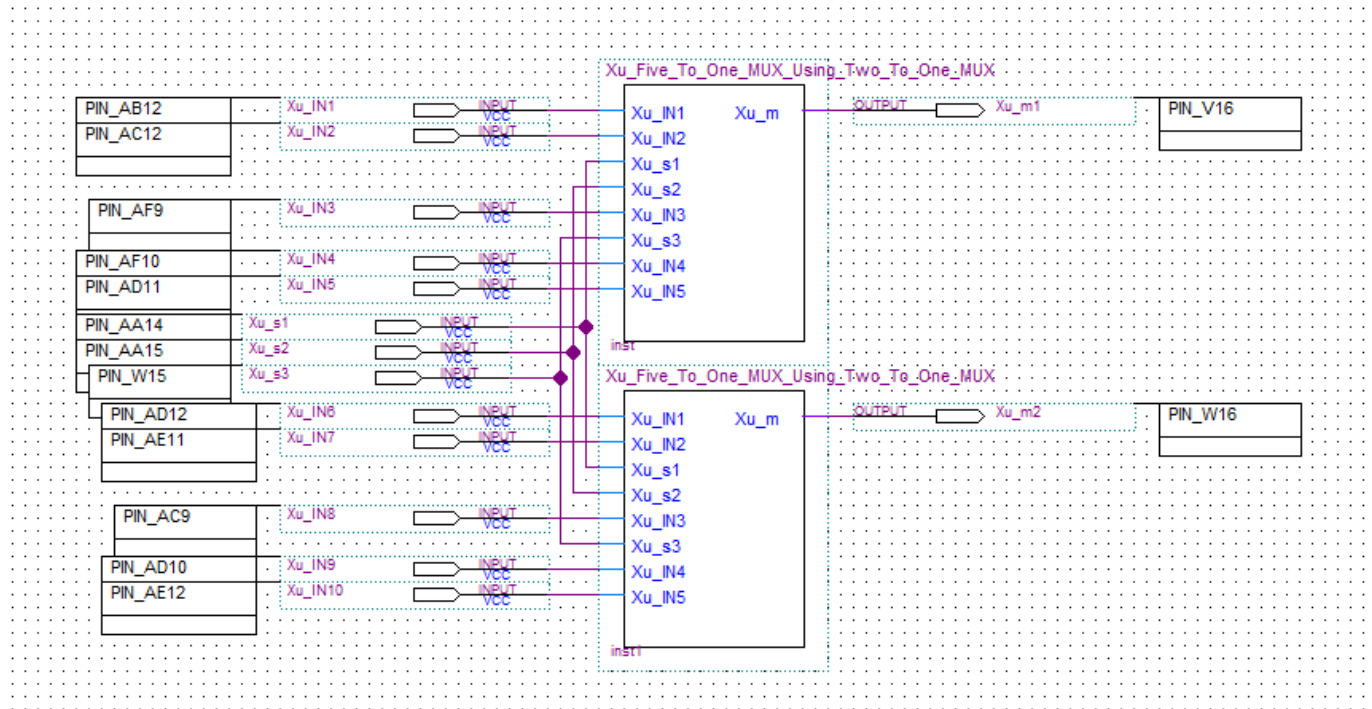


Figure 40: PIN assignments of circuit onto the DE1-SoC Board.

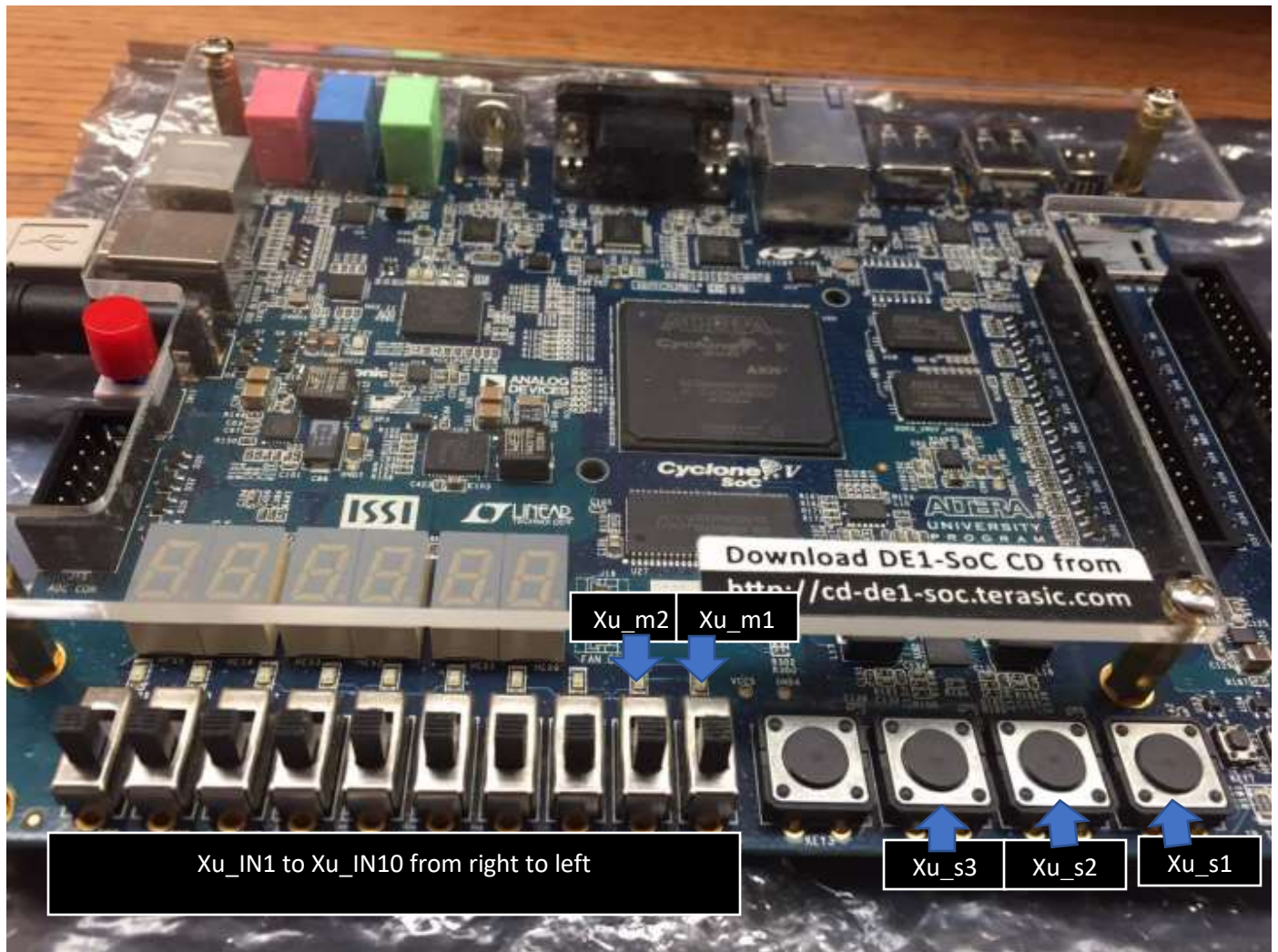


Figure 41: PIN assignments on the DE1-SoC Board.



Figure 41: s1 s2 s3 is 1 (on) IN1 is 1 (on) output m1 is 1 (on).

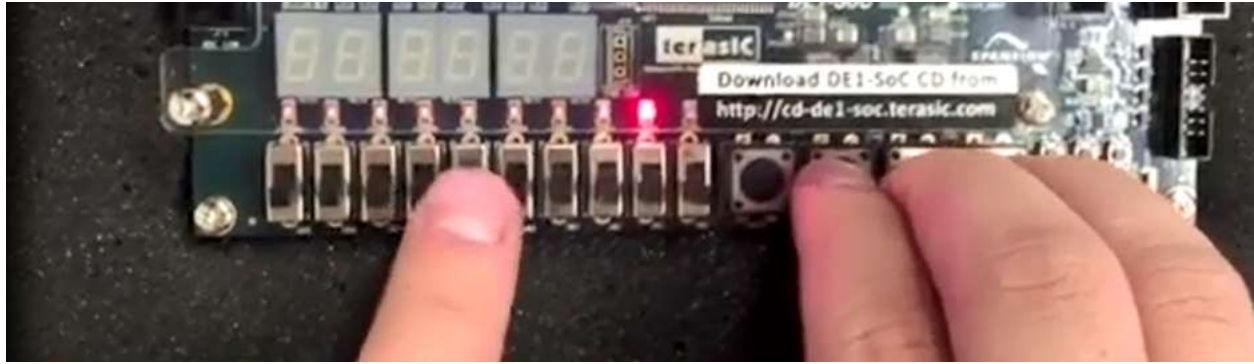


Figure 42: s1 s2 s3 is 1 (on) IN6 is 1 (on) output m2 is 1 (on).

9. 2:4 Decoder

9.1 Functionality and Specifications

A decoder converts modified or compressed data back to its original form which means converting binary information between one to another. We will be designing a 2:4 decoder where it takes two inputs and outputs 4 string of original data. Below is the truth table and Boolean function for 2:4 decoder.

A	B	Q1	Q2	Q3	Q4
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

$Q1 = \text{NOT } A \text{ AND NOT } B$

$Q2 = \text{NOT } A \text{ AND } B$

$Q3 = A \text{ AND NOT } B$

$Q4 = A \text{ AND } B$

When we apply the Boolean functions to build the block diagram for 2:4 decoder, there will be two NOT gates and four AND gates. The two NOT gate will take A and B which creates the negation of A and B. The first AND gate takes NOT A and NOT B as inputs and outputs Q1. The second AND gate takes NOT A and B as inputs and outputs Q2. The third AND gate takes A and NOT B and outputs Q3. The fourth AND gate takes A and B as inputs and outputs Q4.

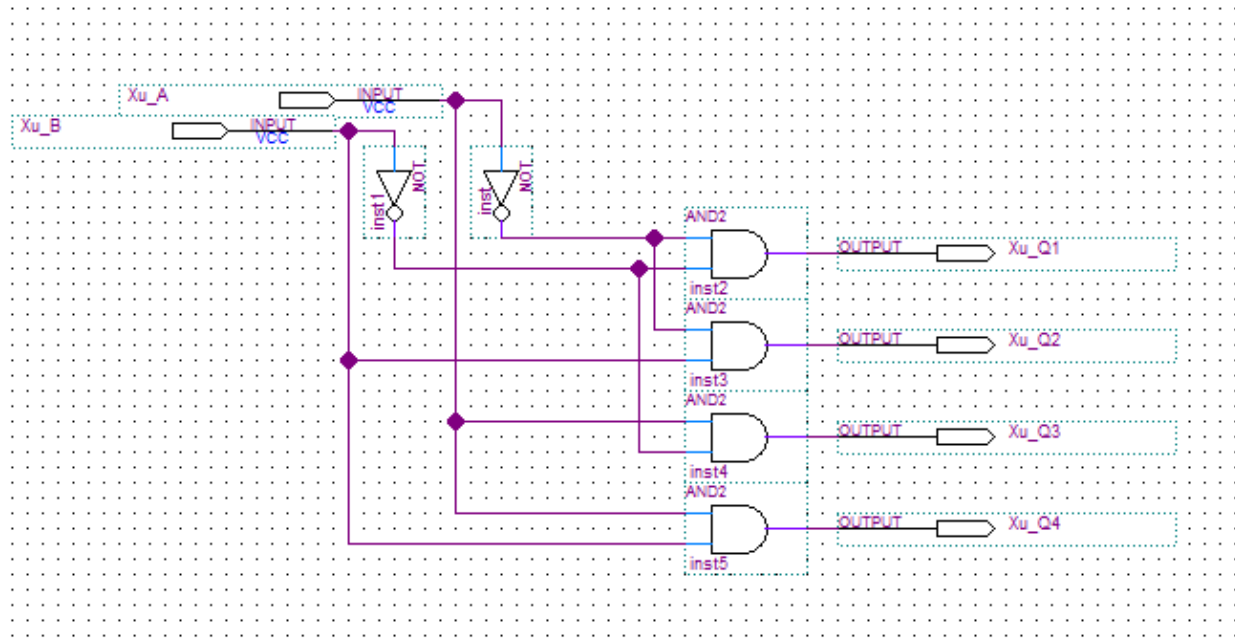


Figure 43: Block diagram of 2:4 Decoder.

9.2 Simulation

In the simulation, we will give values of 0 and 1 to the inputs at varying interval. Input A will have value of 0 and 1 at each 400-ns interval. Input B will have value of 0 and 1 at each 200-ns interval.

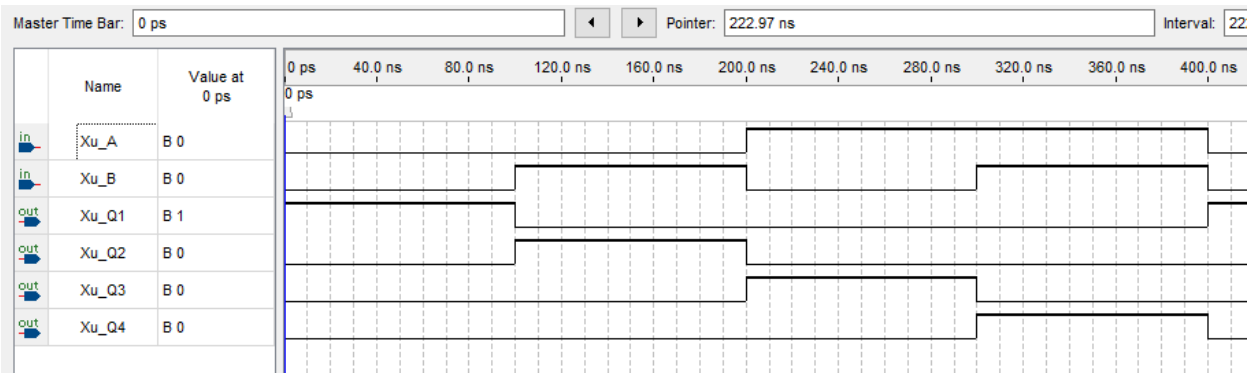


Figure 44: Vector waveform simulation of 2:4 Decoder.

We can observe that when A is 0 and B is 0 output Q1 will be 1. When A is 0 and B is 1 output Q2 will be 1. When A is 1 and B is 0 output Q3 will be 1. When A and B are 1 output Q4 will be 1. The results from the simulation corresponds to the truth table.

9.3 Demonstration

The PIN assignment of inputs and outputs onto the DE1-SoC Board.

Xu_A is assigned to SW[0] which is PIN_AB12

Xu_B is assigned to SW[1] which is PIN_AC12

Xu_Q1 is assigned to LEDR[0] which is PIN_V16

Xu_Q2 is assigned to LEDR[1] which is PIN_W16

Xu_Q3 is assigned to LEDR[2] which is PIN_V17

Xu_Q4 is assigned to LEDR[3] which is PIN_V18

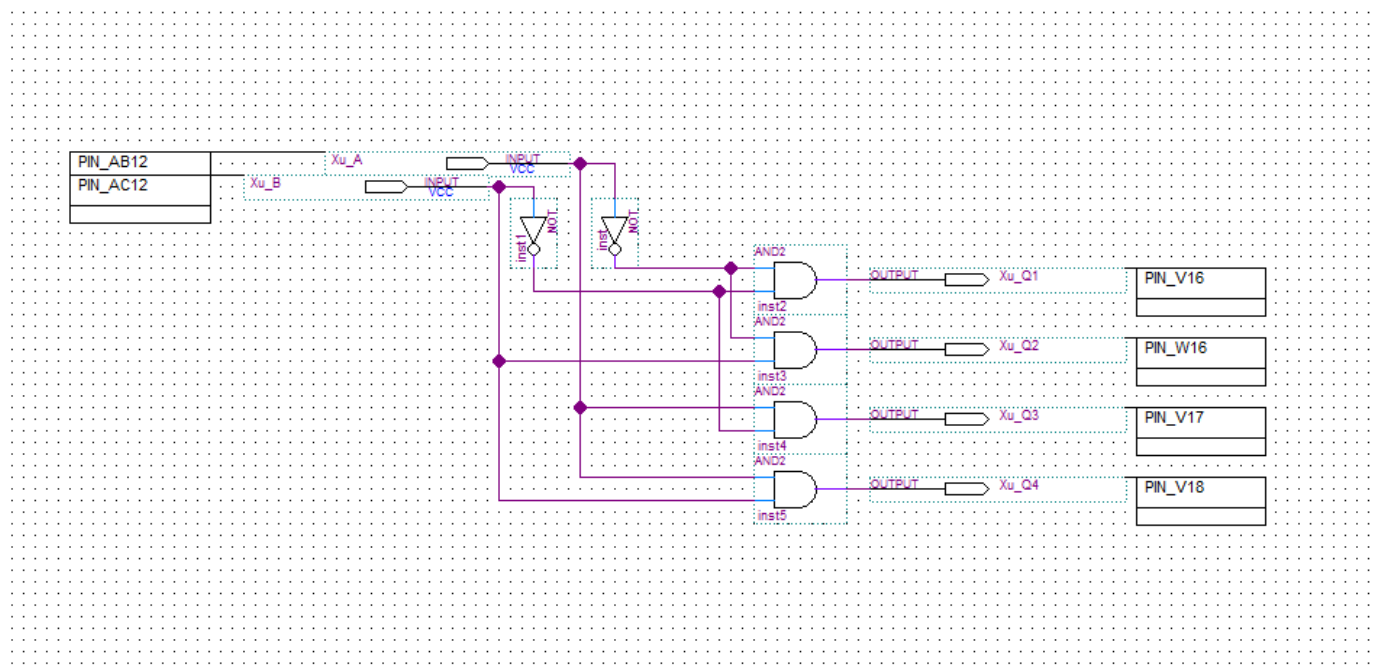


Figure 45: PIN assignment of circuit onto the DE1-SoC Board.

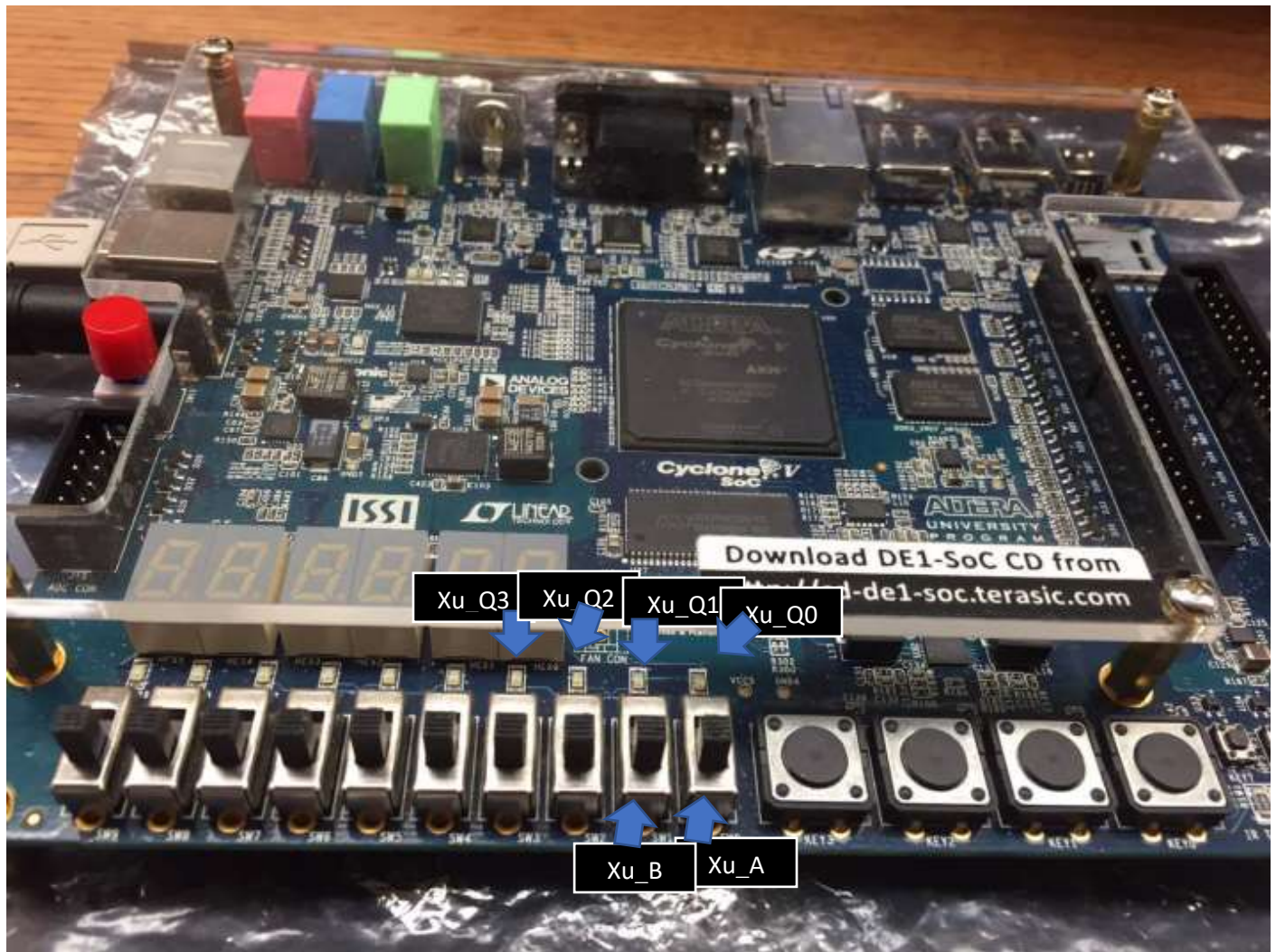


Figure 45: PIN assignment on the DE1-SoC Board.



Figure 46: A is 0 (off) B is 0 (off) outputs Q1 is 1 (on).



Figure 47: A is 1 (on) B is 0 (off) outputs Q3 is 1 (on).

10. 3:8 Decoder

10.1 Functionality and Specifications

The functionality and specifications of the 3:8 decoder is similar to the 2:4 decoder but this time it will have 3 inputs and 8 outputs since there will be 8 possible combinations of the 3 inputs.

Below is the truth table and Boolean function of the 3:8 decoder.

A	B	C	Q1	Q2
0	0	0	1	0
0	0	1	0	1
0	1	0	0	0
0	1	1	0	0
1	0	0	0	0
1	0	1	0	0
1	1	0	0	0
1	1	1	0	0

Q1 = NOT A AND NOT B AND NOT C

Q2 = NOT A AND NOT B AND C

Q3 = NOT A AND B AND NOT C

Q4 = NOT A AND B AND C

Q5 = A AND NOT B AND NOT C

Q6 = A AND NOT B AND C

Q7 = A AND B AND NOT C

Q8 = A AND B AND C

When we apply the Boolean function to create the block diagram for 3:8 decoder, there will be three NOT gates and eight AND gates. The three NOT gates will each take A, B, and C as inputs which creates the negation of A, B, and C. The first AND gate takes NOT A, NOT B, and NOT C as inputs and outputs Q1. The second AND gate takes NOT A, NOT B and C as inputs and outputs Q2. The third AND gate takes NOT A, B and NOT C as inputs and outputs Q3. The fourth AND gate takes NOT A, B and C as inputs and outputs Q4. The fifth AND gate takes A, NOT B and NOT C as inputs and outputs Q5. The sixth AND gate takes A, NOT B and C as inputs and outputs Q6. The seventh AND gate takes A, B and NOT C as inputs and outputs Q7. The eighth AND gate takes A, B and C as inputs and outputs Q8.

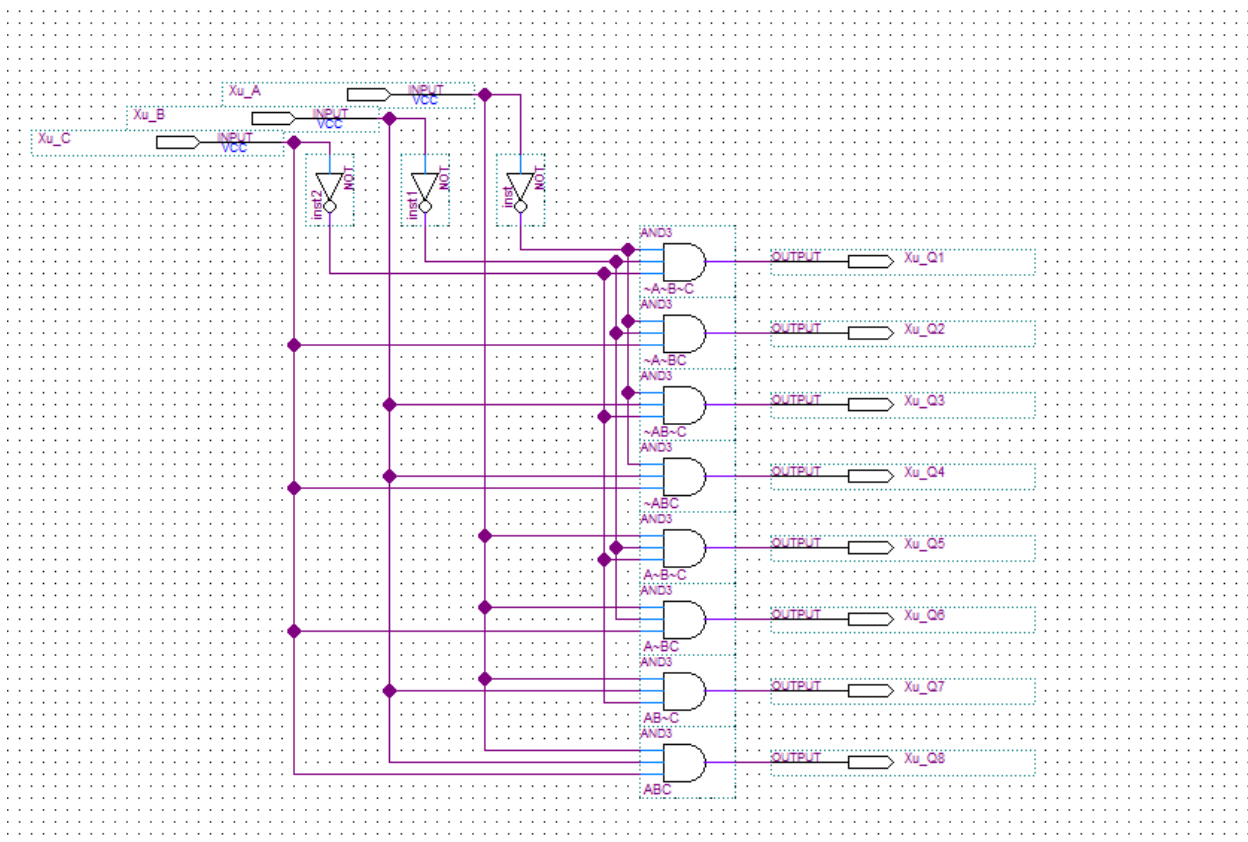


Figure 48: Block diagram of 3:8 decoder.

10.2 Simulation

In the simulation, we will give values of 0 and 1 to the inputs at varying intervals. Input A will have value of 0 and 1 each at 800-ns interval. Input B will have value of 0 and 1 each at 400-ns interval. Input C will have value of 0 and 1 each at 200-ns interval.

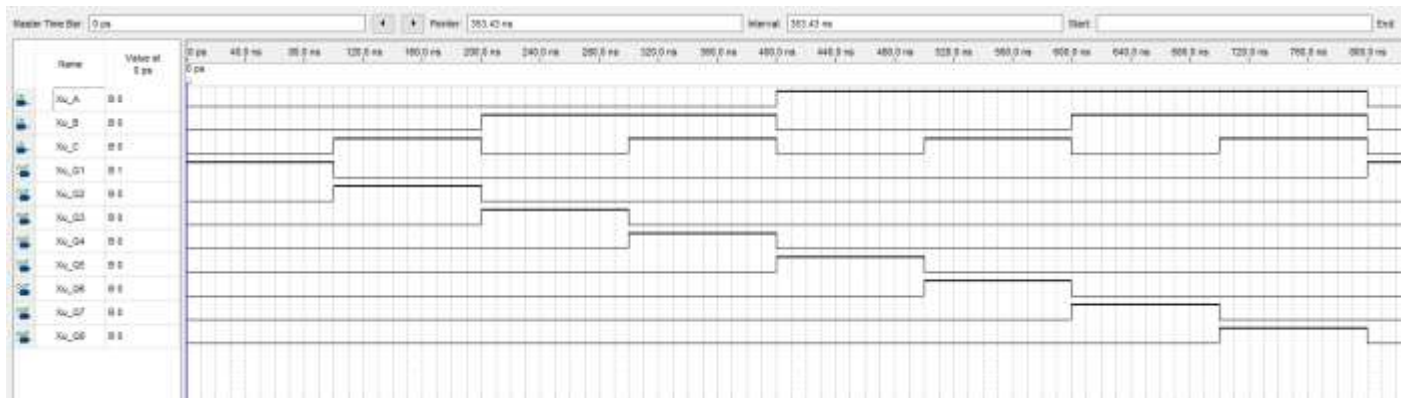


Figure 49: Vector waveform simulation of 3:8 decoder

The result of the simulation corresponds directly to the truth table.

10.3 Demonstration

The PIN assignment of inputs and outputs onto the DE1-SoC Board.

Xu_A is assigned to SW[0] which is PIN_AB12

Xu_B is assigned to SW[1] which is PIN_AC12

Xu_C is assigned to SW[2] which is PIN_AF9

Xu_Q1 is assigned to LEDR[0] which is PIN_V16

Xu_Q2 is assigned to LEDR[1] which is PIN_W16

Xu_Q3 is assigned to LEDR[2] which is PIN_V17

Xu_Q4 is assigned to LEDR[3] which is PIN_V18

Xu_Q5 is assigned to LEDR[4] which is PIN_W17

Xu_Q6 is assigned to LEDR[5] which is PIN_W19

Xu_Q7 is assigned to LEDR[6] which is PIN_Y19

Xu_Q8 is assigned to LEDR[7] which is PIN_W20

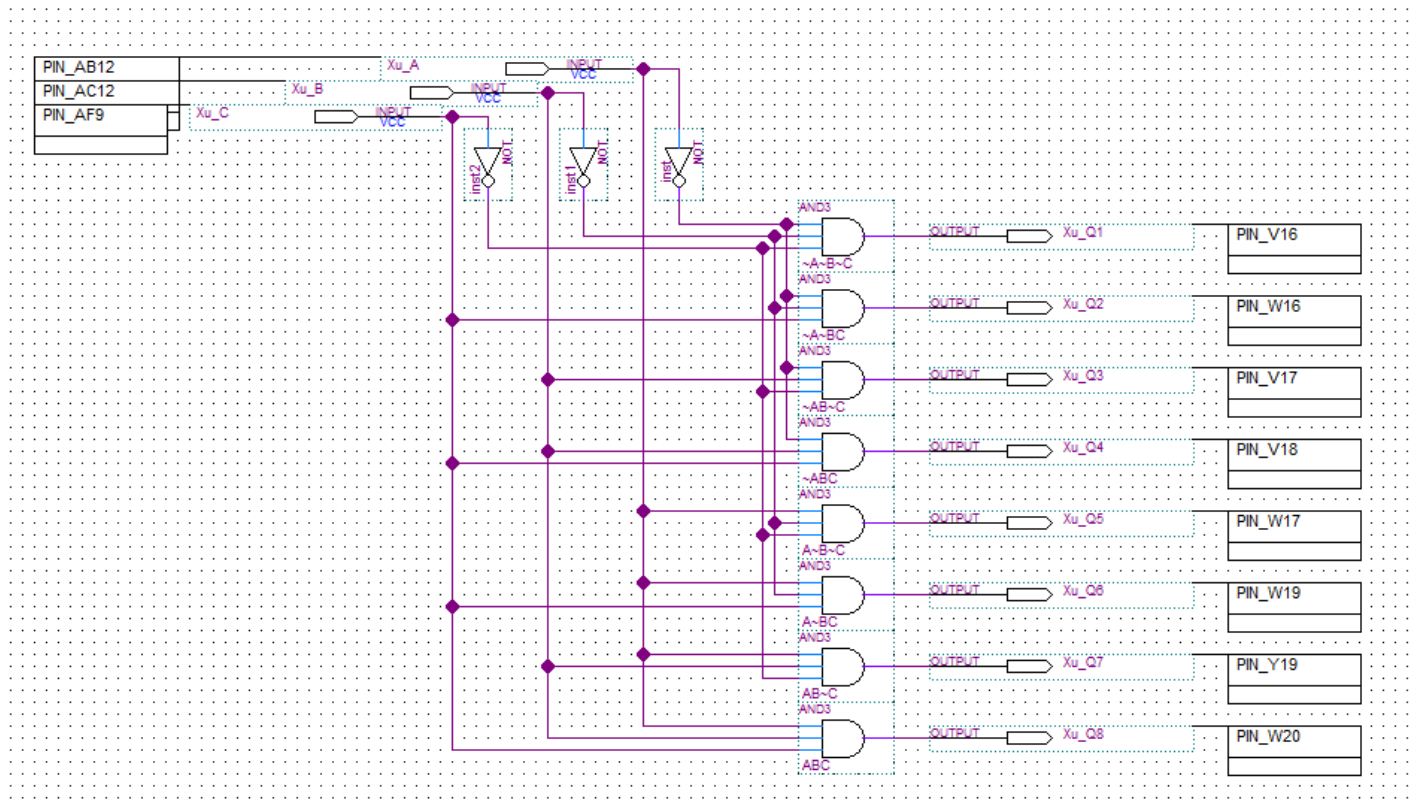


Figure 50: PIN assignments of circuit onto the DE1-SoC Board.

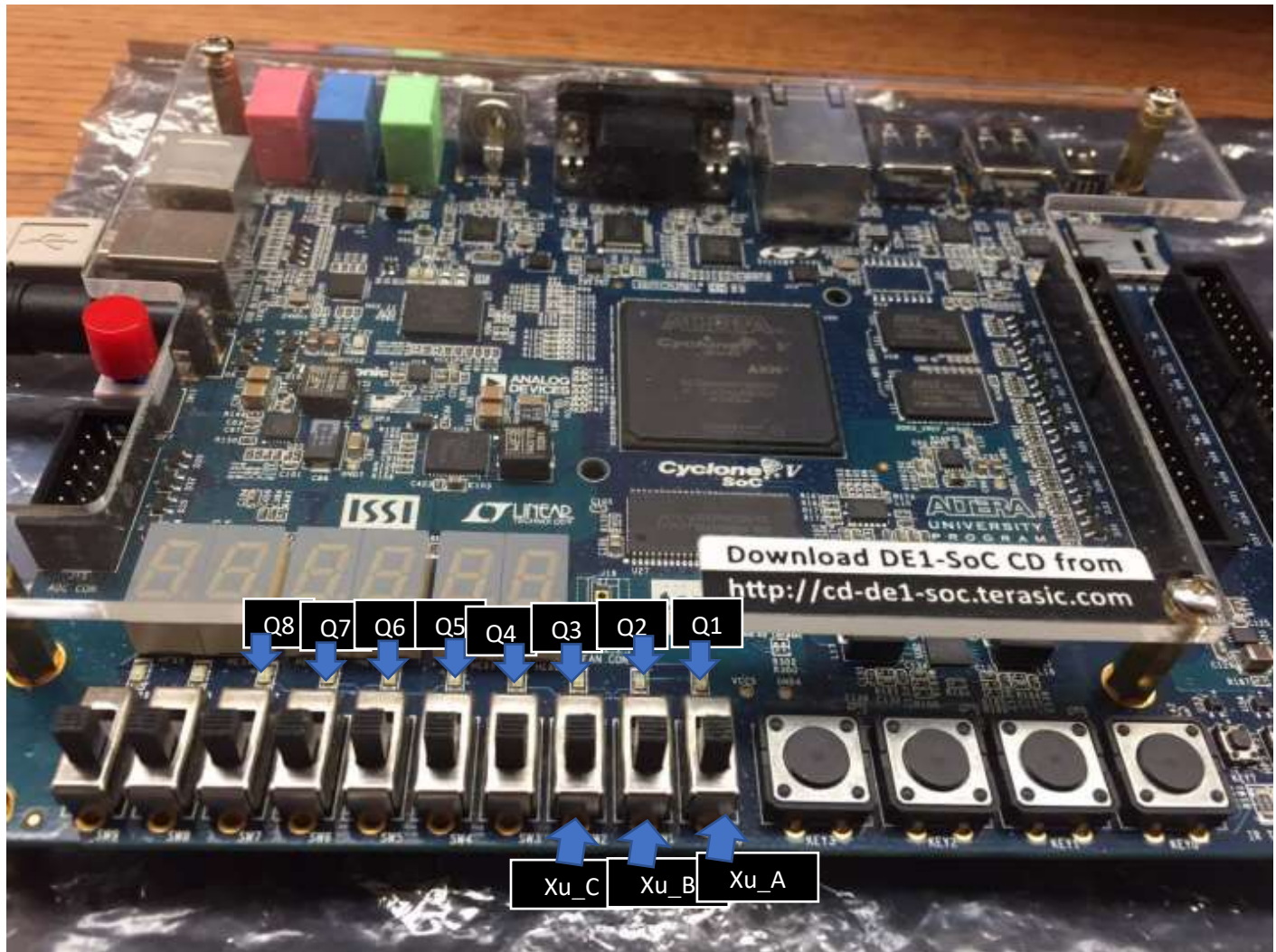


Figure 51: PIN assignment on the DE1-SoC Board.



Figure 52: A is 0 (off) B is 0 (off) C is 0 (off) output Q1 is 1 (on).



Figure 53: A is 1 (on) B is 0 (off) C is 0 (off) output Q5 is 1 (on).

11. 8:3 Encoder

11.1 Functionality and Specification

An encoder converts an original set of data to a modified version, which is similar to decoder in that it changes one format to another. Encoders changes a large set of information into a smaller set. The 8:3 encoder will have 8 inputs and 3 outputs. Below is the truth table and Boolean function. The x's in the graph can represent either 0 or 1 which it doesn't matter since it will not affect the output.

Q1	Q2	Q3
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

D0	D1	D2	D3	D4	D5	D6
1	0	0	0	0	0	0
x	1	0	0	0	0	0
x	x	1	0	0	0	0
x	x	x	1	0	0	0
x	x	x	x	1	0	0
x	x	x	x	x	1	0
x	x	x	x	x	x	1
x	x	x	x	x	x	x

$$Q1 = D1 \text{ OR } D3 \text{ OR } D5 \text{ OR } D7$$

$$Q2 = D2 \text{ OR } D3 \text{ OR } D6 \text{ OR } D7$$

$$Q3 = D4 \text{ OR } D5 \text{ OR } D6 \text{ OR } D7$$

When we apply the Boolean function to create the block diagram for 8:3 encoder, we will need to use three OR gates. The first OR gate will take D1, D3, D5, and D7 as inputs and outputs Q1. The second OR gate will take D2, D3, D6, D7 as inputs and outputs Q2. The third OR gate will take D4, D5, D6, D7 as inputs and outputs Q3.

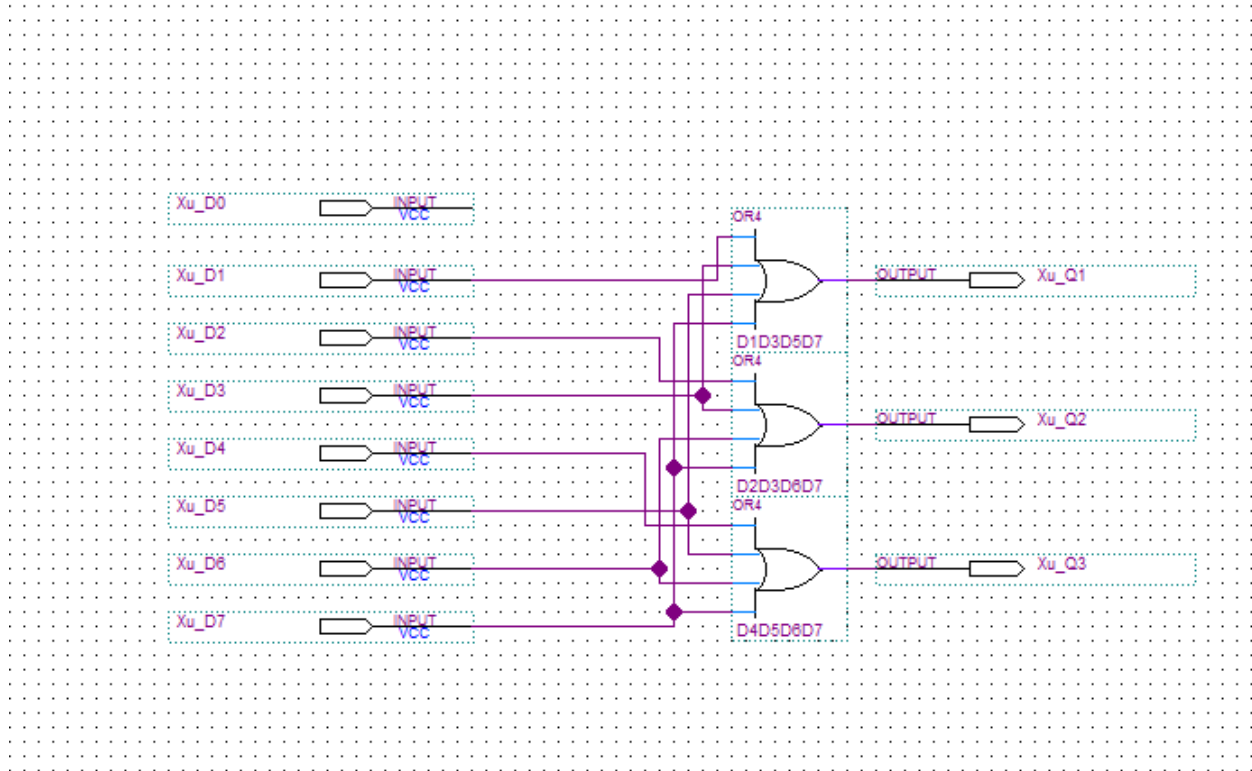


Figure 54: Block diagram of 8:3 encoder.

11.2 Simulation

In the simulation, we will give values of 0 and 1 to the inputs at varying intervals. Input D0 will have value of 0 and 1 at each 960-ns interval. Input D1 will have value of 0 and 1 at each 480-ns interval. Input D2 will have value of 0 and 1 at each 240-ns interval. Input D3 will have value of 0 and 1 at each 120-ns interval. Input D4 will have value of 0 and 1 at each 60-ns interval. Input D5 will have value of 0 and 1 at each 30-ns interval. Input D6 will have value of 0 and 1 at each 15-ns interval. Input D7 will have value of 0 and 1 at each 7.5-ns interval.

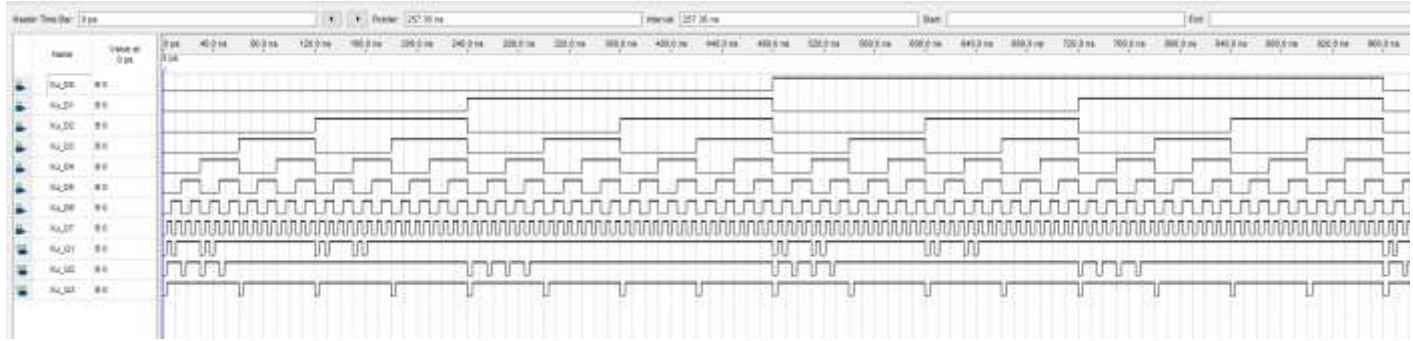


Figure 55: Vector waveform simulation of 8:3 encoder.

The results of the simulation correspond to the expected output and the truth table of the 8:3 encoder. There is also a priority order in this encoder, D7 has the highest priority whereas D0 has the lowest. When D7 is 1, all the other lower priority inputs will be ignored.

11.3 Demonstration

The PIN assignment of inputs and outputs onto the DE1-SoC Board.

Xu_D0 is assigned to SW[0] which is PIN_AB12

Xu_D1 is assigned to SW[1] which is PIN_AC12

Xu_D2 is assigned to SW[2] which is PIN_AF9

Xu_D3 is assigned to SW[3] which is PIN_AF10

Xu_D4 is assigned to SW[4] which is PIN_AD11

Xu_D5 is assigned to SW[5] which is PIN_AD12

Xu_D6 is assigned to SW[6] which is PIN_AE11

Xu_D7 is assigned to SW[7] which is PIN_AC9

Xu_Q1 is assigned to LEDR[0] which is PIN_V16

Xu_Q2 is assigned to LEDR[1] which is PIN_W16

Xu_Q3 is assigned to LEDR[2] which is PIN_V17

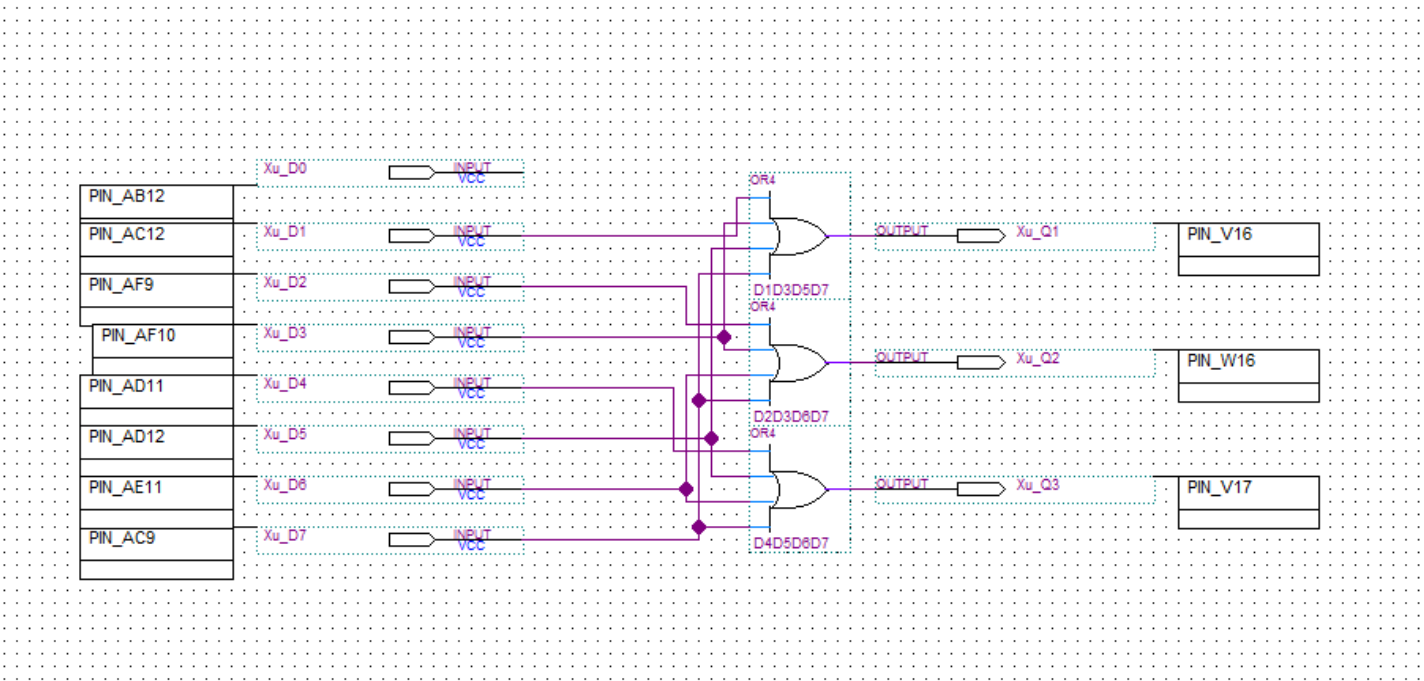


Figure 56: PIN assignment of the circuit onto the DE1-SoC Board.

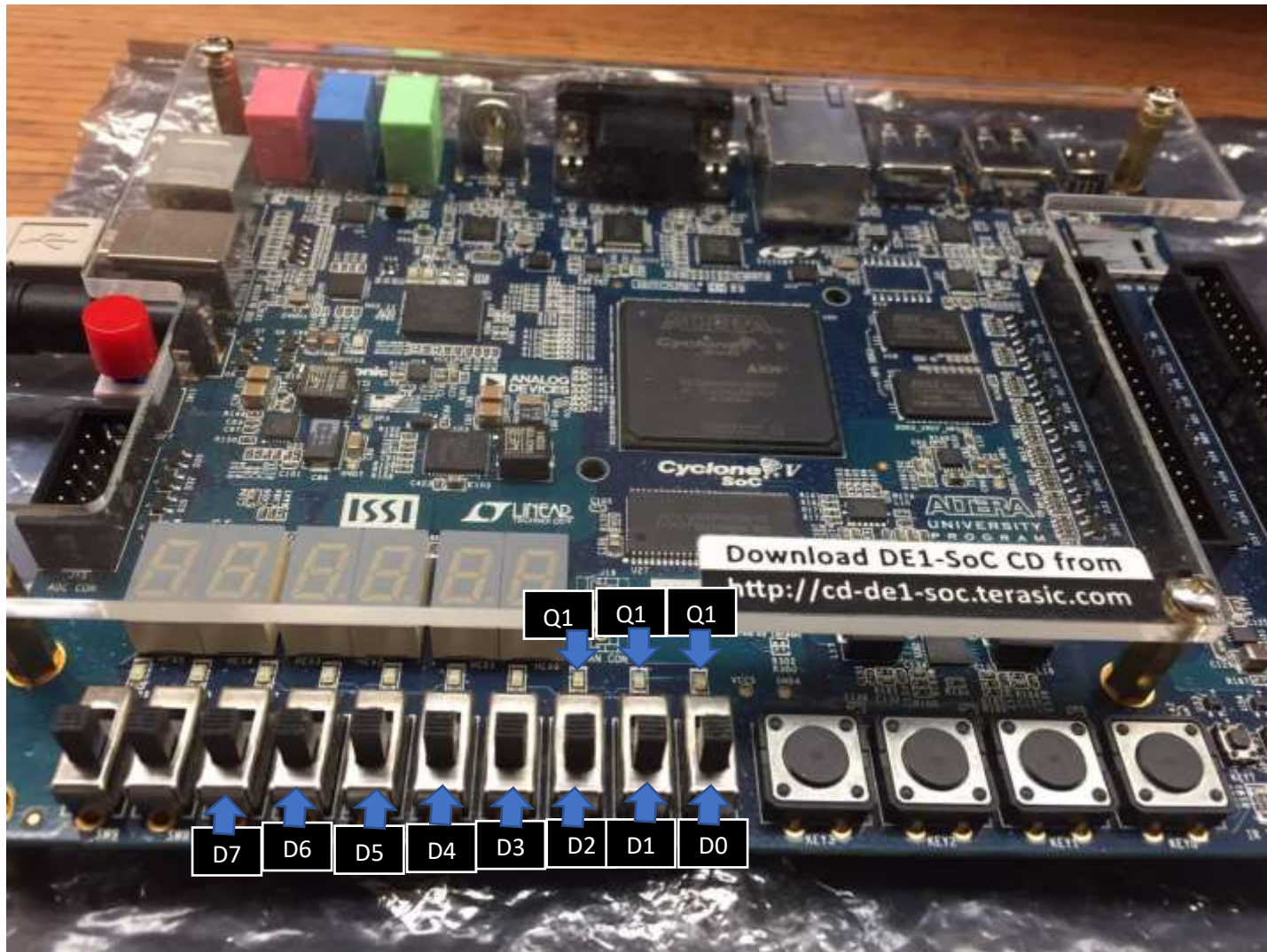


Figure 57: PIN assignment on the DE1-SoC Board.



Figure 58: D1 is 1 (on) outputs Q1 is 1 (on).



Figure 59: D7 is 1 (on) outputs Q1 Q2 Q3 are 1 (on).

12. 1:2 Demultiplexer

12.1 *Functionality and Specifications*

A demultiplexer has one input and more than one output, the purpose of this is to send a signal to one out of the many output it is connected to. A 1:2 demultiplexer will have selector input s and an input IN which produces two outputs. Below is the truth table and Boolean function for 1:2 DeMUX.

s	IN	OUT1	OUT2
0	0	0	0
0	1	0	0
1	0	1	0
1	1	0	1

OUT1 = NOT s AND IN

OUT2 = s and IN

When we apply the Boolean function to create the block diagram for 1:2 DeMUX, we need a NOT gate and two AND gates. The NOT gate will take selector s as input which creates the negation of s. The first AND gate will take s and IN as inputs and outputs Out1. The second AND gate will take NOT s and IN as inputs and outputs Out2.

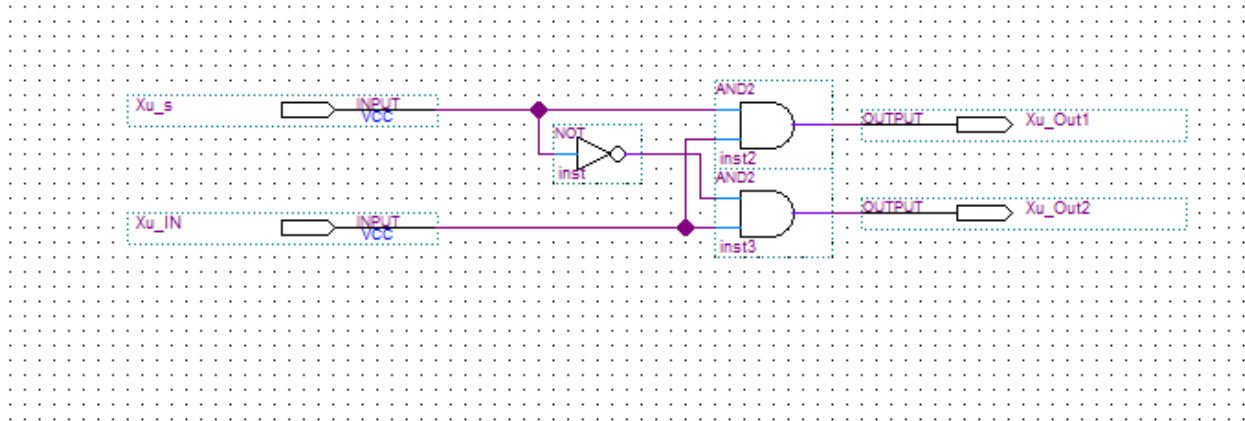


Figure 60: Block diagram of 1:2 DeMUX.

12.2 Simulation

In the simulation, we will give values off 0 and 1 to the inputs at varying intervals. Selector *s* will have value of 0 and 1 at each 400-ns interval. Input *IN* will have value of 0 and 1 at each 200-ns interval.

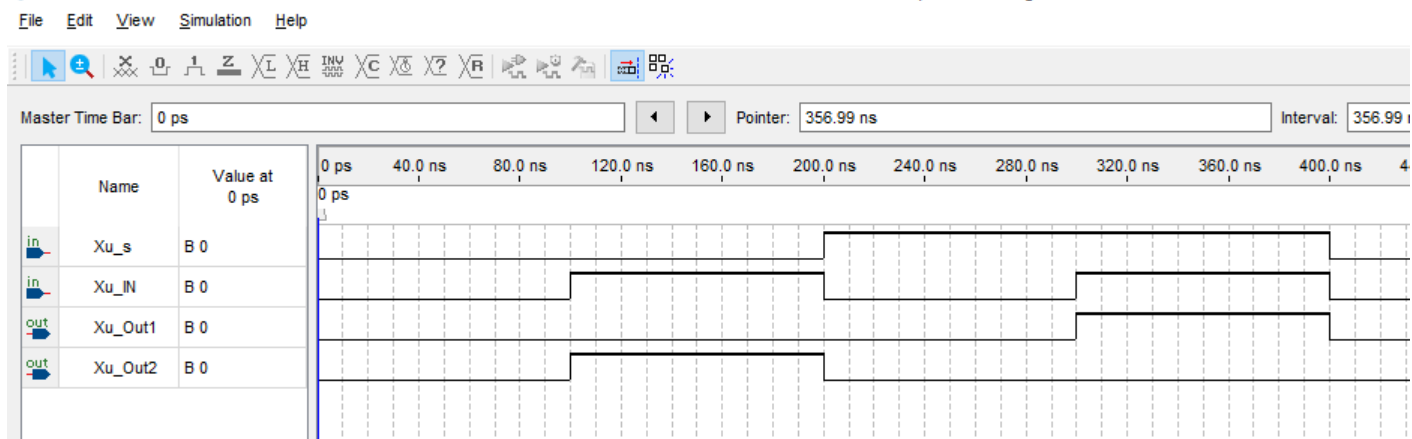


Figure 61: Vector waveform simulation of 1:2 DeMUX.

We can observe that when the selector is 0 and input *IN* is 1, the output *Out2* will be 1. When the selector is 1 and input *IN* is 1, the output *Out1* will be 1. The results correspond to the expected output and the truth table.

12.3 Demonstration

The PIN assignment of inputs and outputs onto the DE1-SoC Board.

Xu_IN is assigned to SW[0] which is PIN_AB12

Xu_s is assigned to SW[1] which is PIN_AC12

Xu_Out1 is assigned to LEDR[0] which is PIN_V16

Xu_Out2 is assigned to LEDR[1] which is PIN_W16

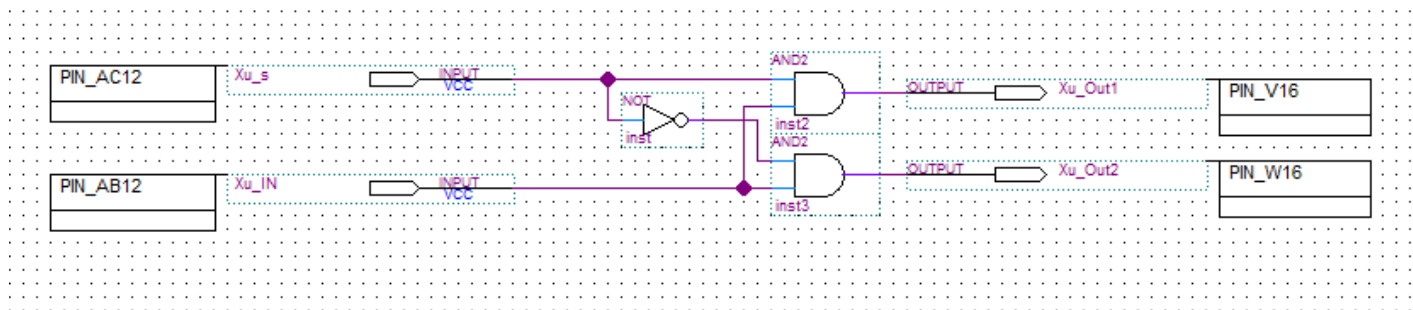


Figure 62: PIN assignment of the circuit onto the DE1-SoC Board.

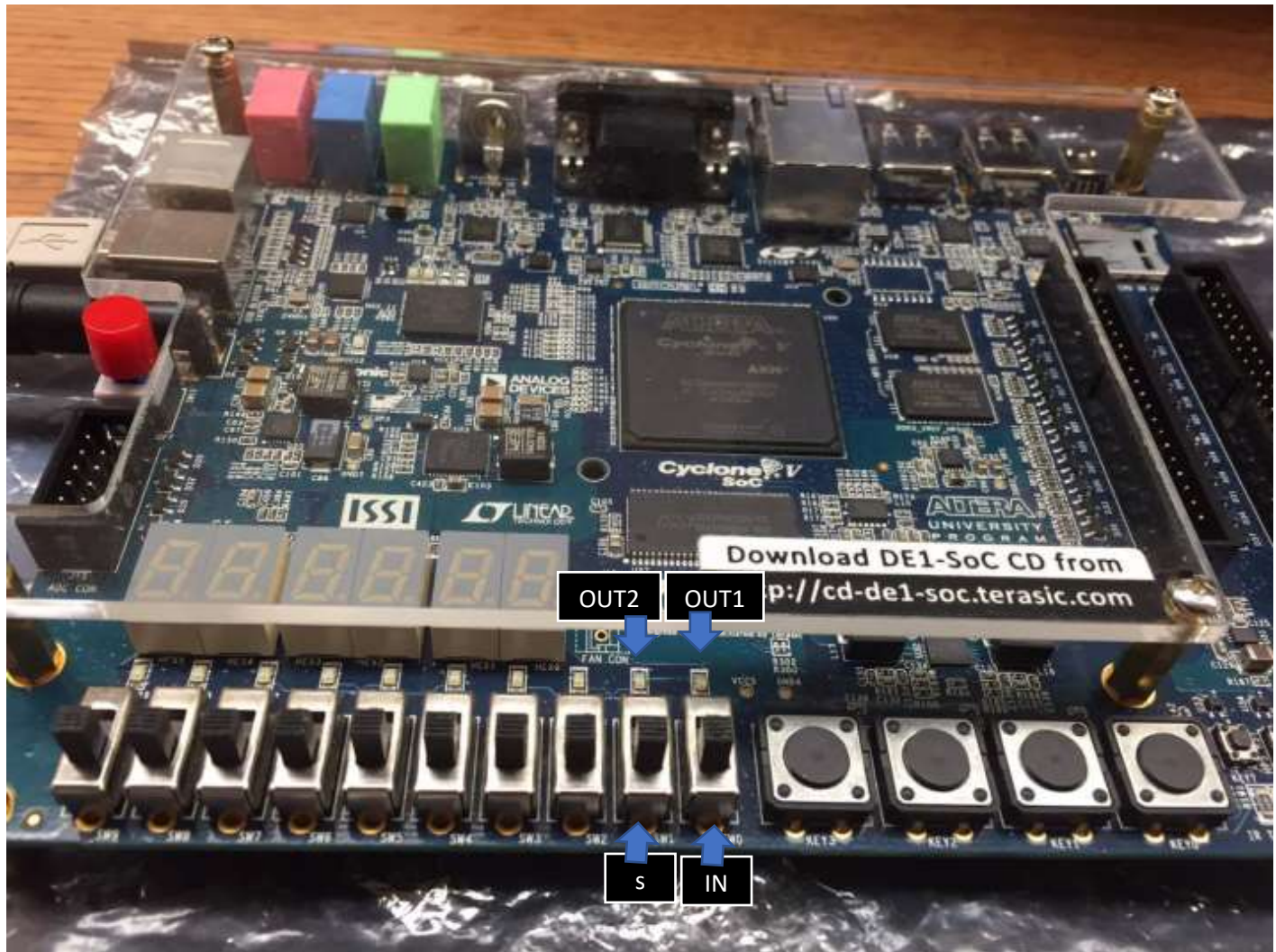


Figure 63: PIN assignment on the DE1-SoC Board.



Figure 64: Selector s is 0 (off) IN is 1 (on) output Out2 is 1 (on)



Figure 64: Selector s is 1 (on) IN is 1 (on) output Out1 is 1 (on)

13. Conclusion

In this lab, I've learned how to build from a simple 2:1 multiplexer and use that knowledge to design a more complex multiplexer such as the 4:1 MUX and 5:1 MUX. There are also many ways to design a circuit which I've learned from the previous lab but there was more of it in this lab. Decoders, Encoders, and DeMultiplexer were also introduced in this lab. The idea of decoders and encoders are to convert sets of data from one format to another. DeMultiplexer lets one input to go to different outputs which is determined by the selector input.

14. Appendix

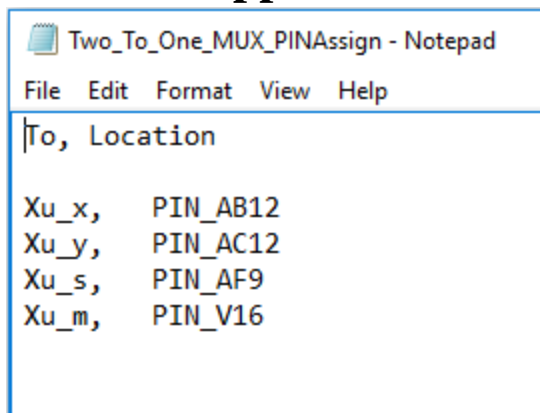


Figure 65: PIN assignment for 2:1 MUX

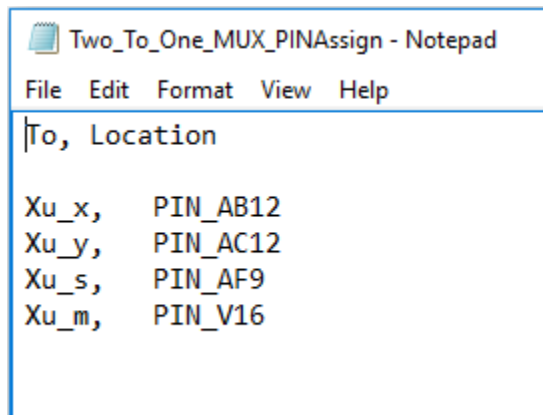


Figure 66: PIN assignment for 2:1 MUX using NAND gates only

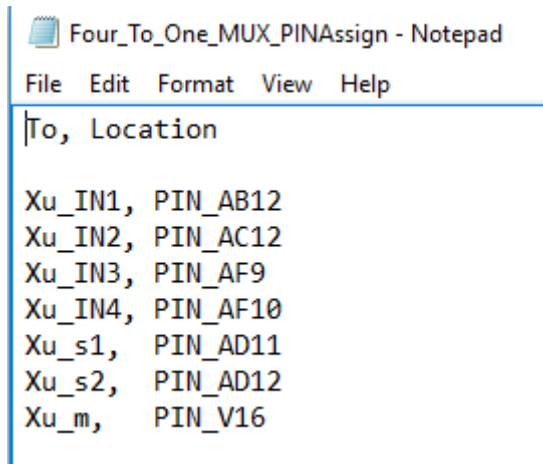


Figure 67: PIN assignment for 4:1 MUX using NAND gates only

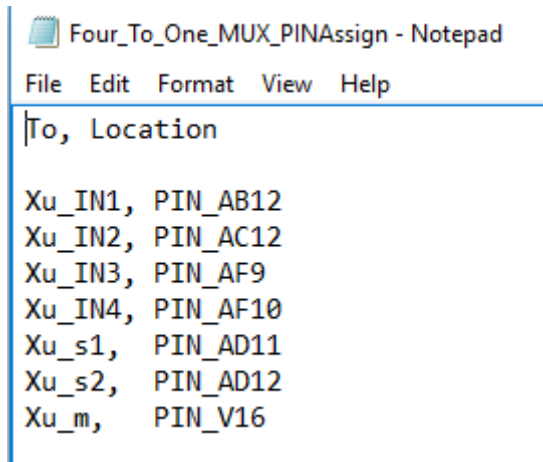


Figure 68: PIN assignment for 4:1 MUX using 2:1 MUX as component

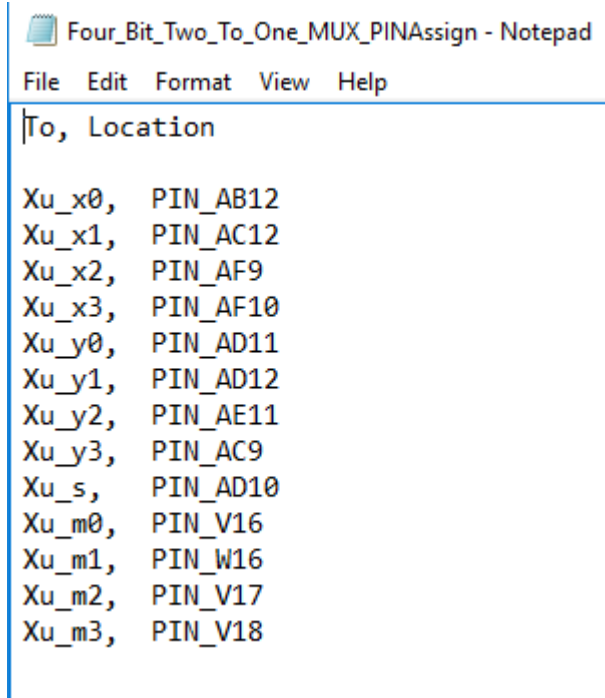


Figure 69: PIN assignment for 4-bit 2:1 MUX

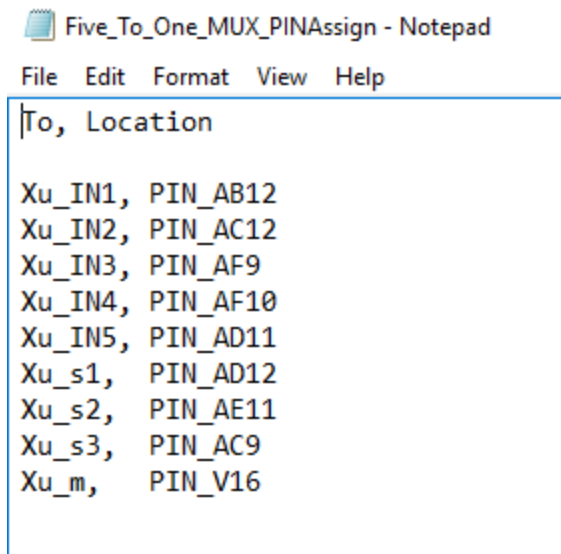
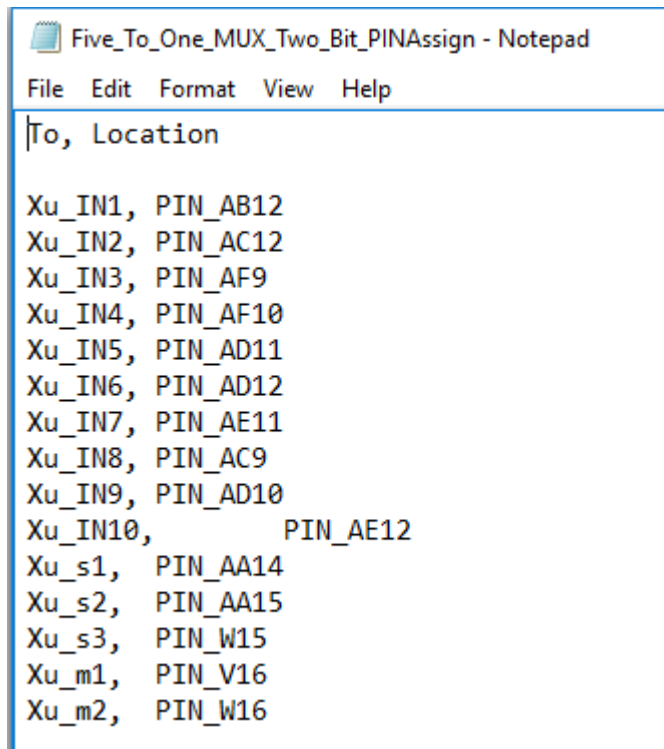
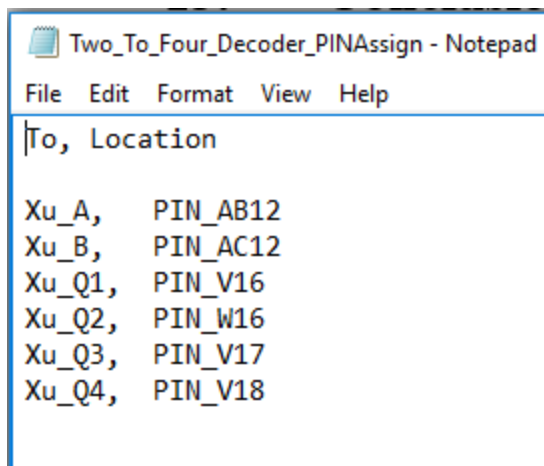


Figure 70: PIN assignment for 2:1 MUX



```
Five_To_One_MUX_Two_Bit_PINAssign - Notepad
File Edit Format View Help
To, Location
Xu_IN1, PIN_AB12
Xu_IN2, PIN_AC12
Xu_IN3, PIN_AF9
Xu_IN4, PIN_AF10
Xu_IN5, PIN_AD11
Xu_IN6, PIN_AD12
Xu_IN7, PIN_AE11
Xu_IN8, PIN_AC9
Xu_IN9, PIN_AD10
Xu_IN10, PIN_AE12
Xu_s1, PIN_AA14
Xu_s2, PIN_AA15
Xu_s3, PIN_W15
Xu_m1, PIN_V16
Xu_m2, PIN_W16
```

Figure 71: PIN assignment for 2-bit 5:1 MUX



```
Two_To_Four_Decoder_PINAssign - Notepad
File Edit Format View Help
To, Location
Xu_A, PIN_AB12
Xu_B, PIN_AC12
Xu_Q1, PIN_V16
Xu_Q2, PIN_W16
Xu_Q3, PIN_V17
Xu_Q4, PIN_V18
```

Figure 72: PIN assignment for 2:4 Decoder

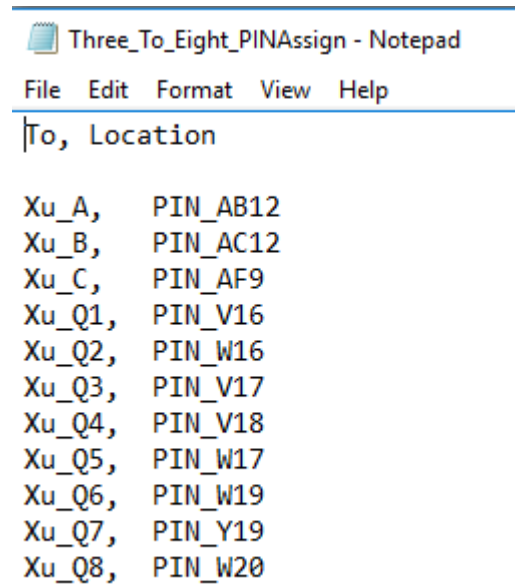


Figure 73: PIN assignment for 3:8 Decoder

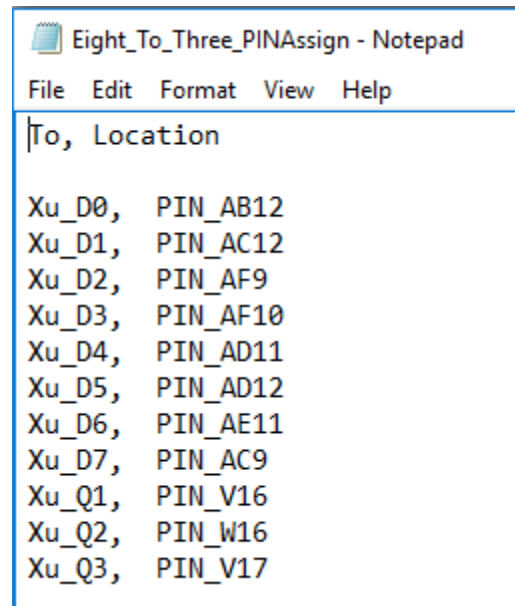


Figure 74: PIN assignment for 8:3 Encoder

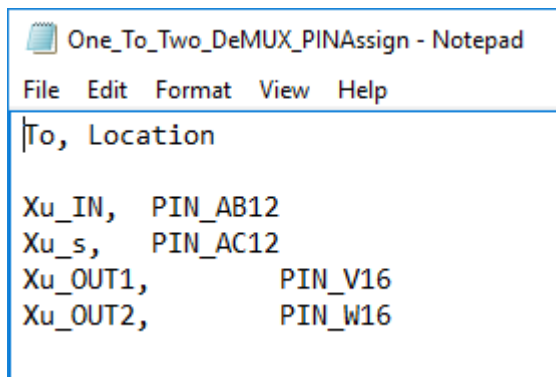


Figure 75: PIN assignment for 1:2 DeMUX.