

LEVEL 4 ASSESSMENT SPECIFICATION

Student name:	Rhys Stevenson	Student ID number:	
Programme:	HNC Electrical/Electronic Engineering		
Module:	Electronics		
Module code:	PCEE4006	Contribution to Overall Module Assessment (%):	50%
Lecturer:	Jason Pointer	Internal Verifier:	Nigel Richards
Assignment Title:	Digital Systems	Word count (or equivalent):	2000
Submission deadline:	11/05/23	Return date of provisional marks & written feedback:	25/05/23
Submission method:	All written assessments, where practical and possible, must be submitted via Turnitin unless otherwise instructed by the Lecturer. (Please DO NOT put this assessment specification into Turnitin or it will match many similarities with other students' submissions.) Alternative submission method (if applicable): Late submission of the assessment will result in a late penalty mark. Penalties for late submission: Up to one week late, maximum mark of 40%. Over one week late, 0%. Only the Extenuating Circumstances Panel may approve a change to submission dates.		
Academic honesty / referencing:	Academic honesty is required. In the main body of your submission you must give credit to authors on whose research and ideas your work is based. Append to your submission a reference list that indicates the books, articles, etc. that you have used, cited or quoted in order to complete this assessment.		

Module Learning Outcomes (from module syllabus)

1. Demonstrate an understanding of semi-conductor components and characteristics and their application in analogue and **digital** electronic circuits.
2. To design, simulate, build, test and evaluate analogue and **digital** electronic circuits using information gained from datasheets.

STUDENT FEEDBACK FORM

This section details the *extent* to which the assessment criteria are demonstrated by you, which in turn determines your mark. The marks available for each category of skill are shown. Lecturers will use the space provided to comment on the achievement of the task(s), including those areas in which you have performed well and areas that would benefit from development/improvement.

Generic Assessment Criteria	Marks available	Marks awarded
1. Engagement with Literature Skills	10	5.4
Engagement with a good range of literature sources but could have include some more text books, sources had been retrieved independently. Generally sound referencing, with very few inaccuracies.		
2. Knowledge and Understanding Skills	45	24.3
Task 1 & 2 achieved 66%. Task 3 & 4 achieved 42%. Knowledge of the basic underlying concepts and principles is accurate with a good understanding in most areas. Showed excellent understanding of combinational logic. Made some errors with the sequential logic including incorrect count sequence.		
3. Cognitive and Intellectual Skills	10	5.2
Good evaluation of the underlying concepts and principles for most of the tasks.		
4. Practical Application Skills	25	18.4
Task 1 & 2 achieved 82%. Task 3 & 4 achieved 65%. An excellent application of practical work for most tasks, the set up for the counter had some elements that were not connected. Very good use of the test equipment and Multisim to obtain the required test results.		
5. Transferable Skills for Life and Professional Practice	10	7
Work is coherent, very fluent and is presented professionally. Excellent layout making the assignment easy to read.		

Assessment Mark (Assessment marks are subject to ratification at the Exam Board. These comments and marks are to give feedback on module work and are for guidance only until they are confirmed.)	Late Submission Penalties (tick if appropriate)		60%
	Up to 1 week late (40% Max)		
	Over 1 week late (0%)		

PCEE4006 Electronics Assignment:

Digital Systems

By Rhys Stevenson

Table of Contents:

Title page	... 2
Table of contents	... 3
Introduction	... 4
Task 1	
Part A	... 4
Part B	... 4
Part C	... 5
Part D	... 6
Part E	... 7
Part F	... 9
Task 2	...11
Task 3	
Part A	... 13
Part B	... 15
Task 4	... 16
References	... 21
Bibliography	... 22

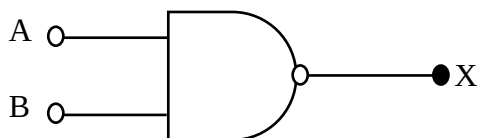
Introduction –

Digital systems use logic, which is a state of being either a 1 or a 0 to determine outputs. Sequential logic is a commonly used design which essentially a combination system (where outputs is only dependant on the inputs) where the outputs are also fed back into the inputs. This allows previously held outputs to be used as a current input data stream. Logic states in these systems are maintained until new inputs are given (Ferdjallah, 2011). ✓

Task 1.

The goal of this task was to answer a number of questions relating to Boolean equations, Demorgans and logic gate circuits. For each question the truth table was determined along with Boolean expression, this was then confirmed using multisim to corroborate the results.

Part (A)



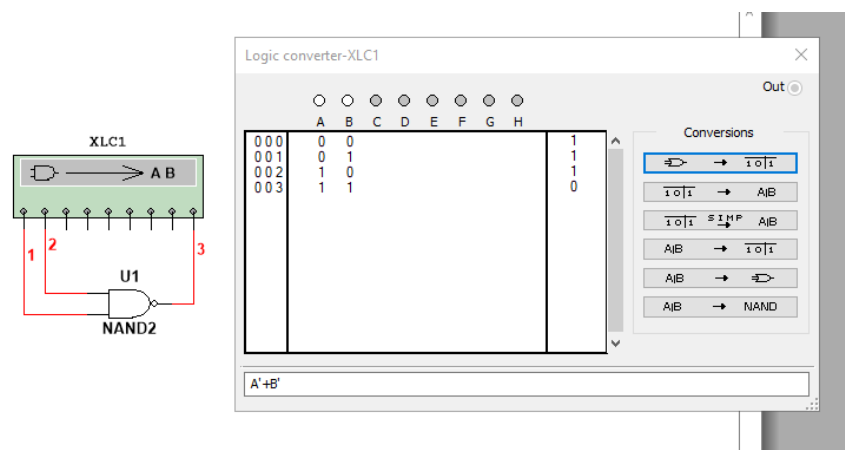
A	B	X
0	0	1
0	1	1
1	0	1
1	1	0

 ✓

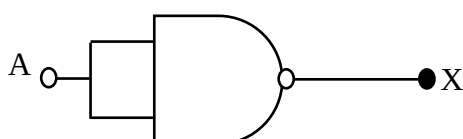
The diagram of the NAND gate shown above was used to solve the truth table for X. This was used to show that the expression for the circuit was:

$$X = \bar{A} \cdot \bar{B}$$
 ✓

This was repeated using Multisim to give the same results as shown below:



Part (B)



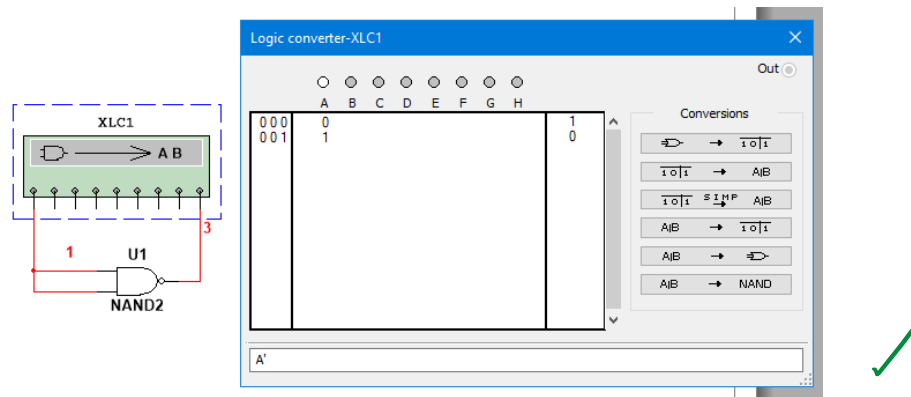
A	X
0	1
1	0

 ✓

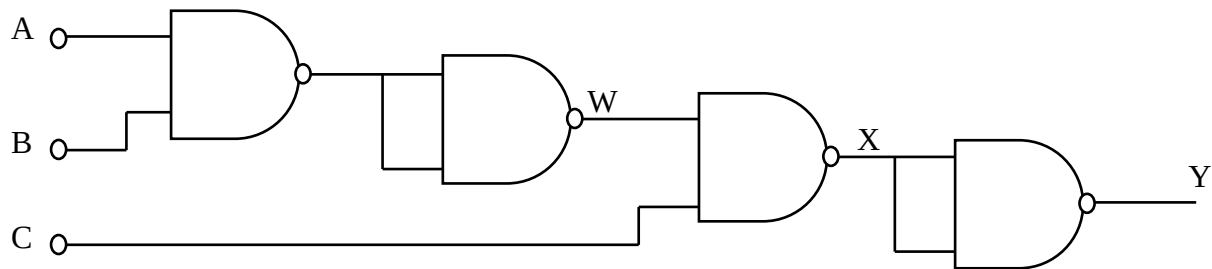
The diagram above shows a NAND gate acting as a NOT gate. The truth table could be solved by inverting the inputs, this gives the flowing formula for the circuit:

$$X = \bar{A}$$
 ✓

The screenshot on the following page confirms this result as shown using multisim:



Part (C)



A	B	C	W	X	Y
0	0	0	0	1	0
0	0	1	0	1	0
0	1	0	0	1	0
0	1	1	0	1	0
1	0	0	0	1	0
1	0	1	0	1	0
1	1	0	1	1	0
1	1	1	1	0	1

With this task the output of multiple parts of the system was needed to be determined to complete the truth table. The diagram above indicates the locations of the multiple outputs. The expression for these values as well as the simplified version are as follows:

$$W = A.B.\bar{C} + A.B.C$$

$$W = A.B$$

$$X = \bar{A}.\bar{B}.\bar{C} + \bar{A}.\bar{B}.C + A.\bar{B}.C + \bar{A}.B.C + A.\bar{B}.\bar{C} + A.\bar{B}.C + A.B.\bar{C}$$

$$X = \overline{A.B.C}$$

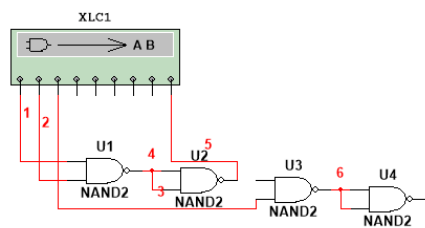
$$Y = A.B.C$$

$$Y = A.B.C$$

Screenshots at each stage were taken as proof of results, these are shown below.

Have not stated what logic gate W, X and Y would represent.

Results for W:



Logic converter-XLC1

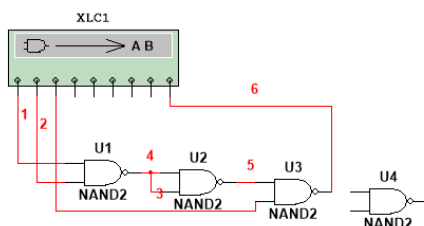
	A	B	C	D	E	F	G	H
000	0	0	0					0
001	0	0	1					0
002	0	1	0					0
003	0	1	1					0
004	1	0	0					0
005	1	0	1					0
006	1	1	0					1
007	1	1	1					1

Conversions

- $A \rightarrow A'$
- $A'B \rightarrow A'B$
- $A'B \rightarrow A'B$
- $A'B \rightarrow A'B$
- $A'B \rightarrow A'B$
- $A'B \rightarrow A'B$
- $A'B \rightarrow A'B$
- $A'B \rightarrow A'B$

AB

Results for X:



Logic converter-XLC1

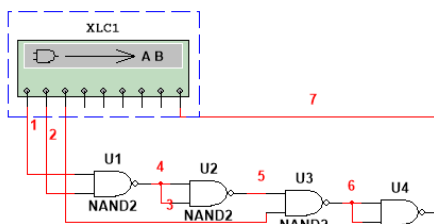
	A	B	C	D	E	F	G	H
000	0	0	0					1
001	0	0	1					1
002	0	1	0					1
003	0	1	1					1
004	1	0	0					1
005	1	0	1					1
006	1	1	0					1
007	1	1	1					0

Conversions

- $A \rightarrow A'$
- $A'B \rightarrow A'B$
- $A'B \rightarrow A'B$
- $A'B \rightarrow A'B$
- $A'B \rightarrow A'B$
- $A'B \rightarrow A'B$
- $A'B \rightarrow A'B$
- $A'B \rightarrow A'B$

$A' + B' + C'$

Results for Y:



Logic converter-XLC1

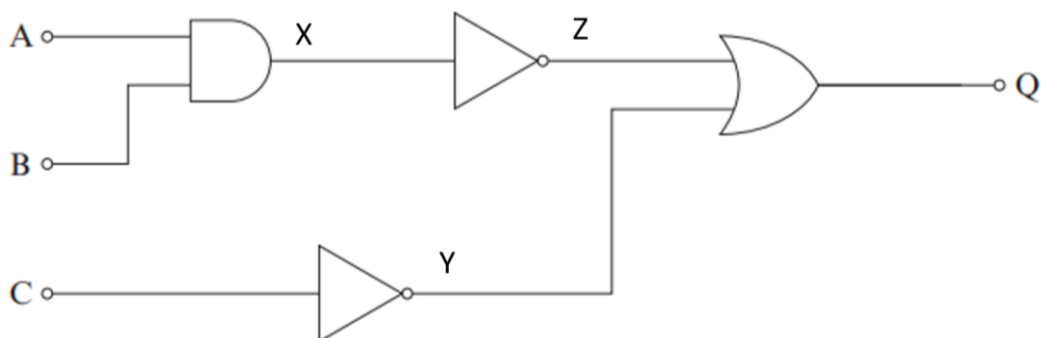
	A	B	C	D	E	F	G	H
000	0	0	0					0
001	0	0	1					0
002	0	1	0					0
003	0	1	1					0
004	1	0	0					0
005	1	0	1					0
006	1	1	0					0
007	1	1	1					1

Conversions

- $A \rightarrow A'$
- $A'B \rightarrow A'B$
- $A'B \rightarrow A'B$
- $A'B \rightarrow A'B$
- $A'B \rightarrow A'B$
- $A'B \rightarrow A'B$
- $A'B \rightarrow A'B$
- $A'B \rightarrow A'B$

ABC

Part (D)



For this part of the task the aim was to produce a truth table and the Boolean expression, along with a redesign the circuit only using two input NAND gates. The truth table and Boolean are shown below.

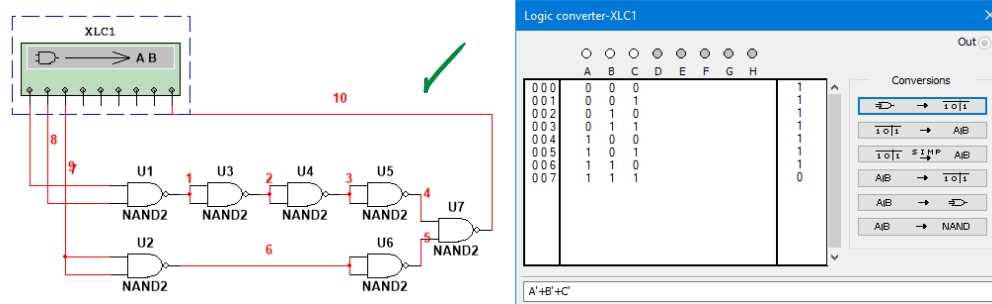
A	B	C	X	Y	Z	Q
0	0	0	0	1	1	1
0	0	1	0	0	1	1
0	1	0	0	1	1	1
0	1	1	0	0	1	1
1	0	0	0	1	1	1
1	0	1	0	0	1	1
1	1	0	1	1	0	1
1	1	1	1	0	0	0

Using the truth table above, the Boolean expression is show below.

$$Q = \bar{A}.\bar{B}.\bar{C} + \bar{A}.\bar{B}.C + \bar{A}.B.\bar{C} + \bar{A}.B.C + A.\bar{B}.\bar{C} + A.\bar{B}.C + A.B.\bar{C}$$

$$Q = \bar{A} + \bar{B} + \bar{C} = \overline{A.B.C}$$

Constructing a circuit using this expression results in an overcomplicated design which can be seen below.



Using Demorgan's theorem it was possible to convert the expression to represent and NAND gate circuit layout using the bars as a guide, the workings out below show this.

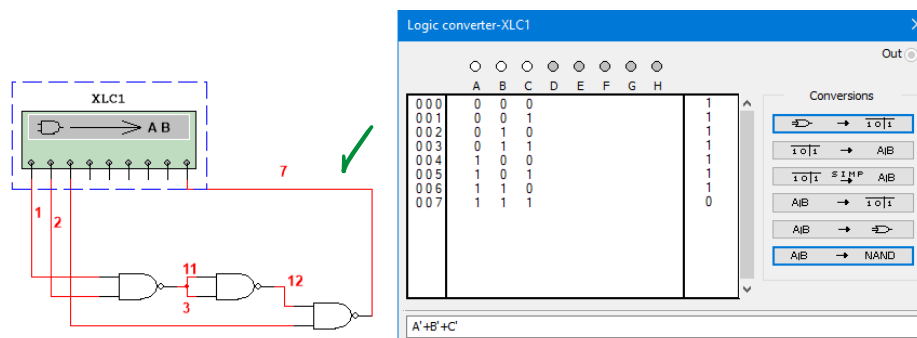
$$Q = \bar{A} + \bar{B} + \bar{C}$$

$$\bar{Q} = \overline{\bar{A} + \bar{B} + \bar{C}}$$

$$\bar{Q} = \bar{\bar{A}}.\bar{\bar{B}}.\bar{\bar{C}}$$

$$Q = \overline{\bar{A}.\bar{B}.\bar{C}}$$

The NAND gate circuit is essentially a 3-input gate, yet it must be constructed out of 2 input NAND components the final design can be seen below.



Good.

Part (E)

For this section the aim was to simplify the expressions given only using long form Boolean algebra and Demorgan's, using multisim to verify the results, then design a NAND logic circuit. The first of the expressions can be seen below.

$$Q = (A + B)(A + \bar{B})(\bar{A} + C) \quad \checkmark$$

$$Q = (A.A + A.\bar{B} + A.B + B.\bar{B})(\bar{A} + C) \quad \checkmark$$

$$Q = (A + A.\bar{B} + A.B)(\bar{A} + C)$$

$$Q = (A + (1 + \bar{B}) + A.B)(\bar{A} + C) \quad \checkmark$$

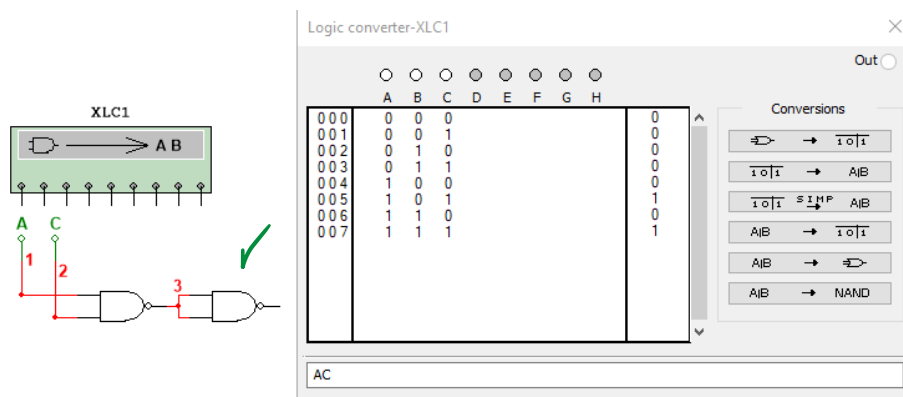
$$Q = (A + A.B)(\bar{A} + C) \quad \checkmark$$

$$Q = A(\bar{A} + C)$$

$$Q = A.\bar{A} + A.C \quad \checkmark$$

$$Q = A.C \quad \checkmark$$

The simplification of Q showed that no Demorgan's was necessary as the NAND circuit was equal to a AND gate as shown below.



For the second half of part E, the Boolean expression below was simplified.

$$Q = A.B.\bar{C} + A.\bar{C} + \bar{A}.\bar{C}.D + \bar{A}.\bar{C}.\bar{D} \quad \checkmark$$

$$Q = A.\bar{C} + \bar{A}.\bar{C}.D + A.\bar{C}.\bar{D} \quad \checkmark$$

$$Q = A.\bar{C} + \bar{A}.\bar{C}.D \quad \checkmark$$

$$Q = \bar{C}(\bar{A}.D + A) \quad \checkmark$$

$$Q = \bar{C}(D + A)$$

$$Q = \bar{C}.D + \bar{C}.A \quad \checkmark$$

You have written down the wrong equation.

Demorgans was then used to determine the NAND gate setup as shown below.

$$Q = \bar{C}.D + \bar{C}.A \quad \checkmark$$

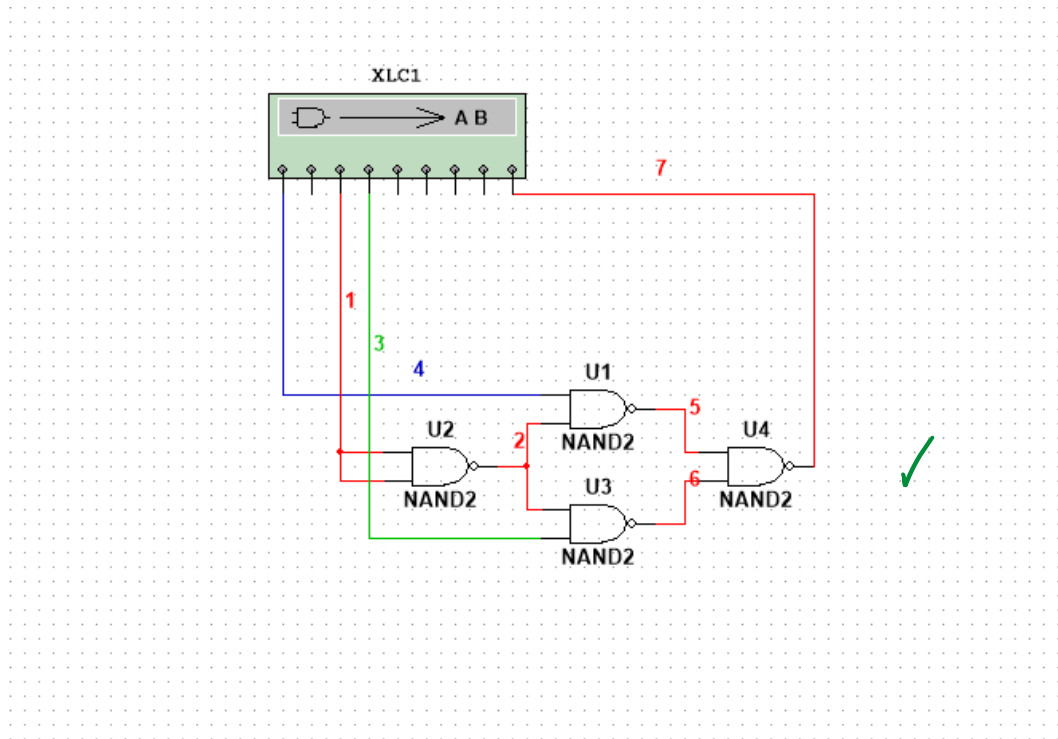
$$\bar{Q} = \overline{\bar{C}.D + \bar{C}.A} \quad \checkmark$$

$$\bar{Q} = \overline{\bar{C}.D}.\overline{\bar{C}.A} \quad \checkmark$$

$$Q = \overline{\overline{\bar{C}.D}.\overline{\bar{C}.A}} \quad \checkmark$$

You have shown the correct method but use wrong equation.

With the final equation done the following NAND gate circuit was constructed.



Part (F)

The final part of task 1 was to design a Binary Code Decimal (BCD) to detect a range between 1010 to 1111 of numbers that are not used. For this a Karnaugh map was constructed to determine the expression output of the device that would result from an unused number. The Truth table for the device can be seen below.

	A	B	C	D	Q
1	0	0	0	0	0
2	0	0	0	1	0
3	0	0	1	0	0
4	0	0	1	1	0
5	0	1	0	0	0
6	0	1	0	1	0
7	0	1	1	0	0
8	0	1	1	1	0
9	1	0	0	0	0
10	1	0	0	1	0
11	1	0	1	0	1
12	1	0	1	1	1
13	1	1	0	0	1
14	1	1	0	1	1
15	1	1	1	0	1
16	1	1	1	1	1

For the truth table the output would equal 1 when any of the unused BCD's are detected, this gave the expression:

$$Q = A.\bar{B}.C.\bar{D} + A.\bar{B}.C.D + A.B.\bar{C}.\bar{D} + A.B.\bar{C}.D + A.B.C.\bar{D} + A.B.C.D$$

This allowed the construction of a Karnaugh map to give a simplified version.

	CD				
AB		00	01	11	10
	00	0	0	0	0
	01	0	0	0	0
	11	1	1	1	1
	10	0	0	1	1

✓

From the map and Demorgans the expression could be used to show the NAND gate configuration for the circuit. It should also be noted that input D had no effect of the outcome.

$$Q = A.B + A.C$$

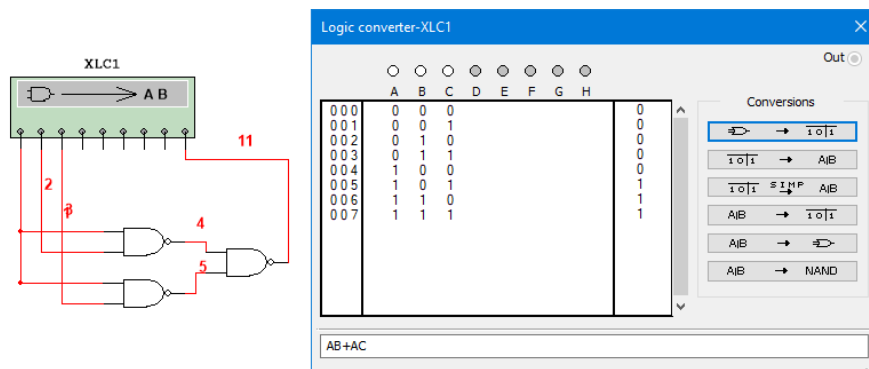
$$\bar{Q} = \overline{A.B + A.C}$$

$$\bar{Q} = \overline{A.B} . \overline{A.C}$$

$$Q = \overline{\overline{A.B} . \overline{A.C}}$$

✓
✓
✓
✓

The final expression showed that there would be a shared NAND gate between A and B, another between A and C and both outputs would pass through a final gate for the output. This was shown by using this design in multisim to confirm the output results as shown below.



✓

very well done.

Task 2 –

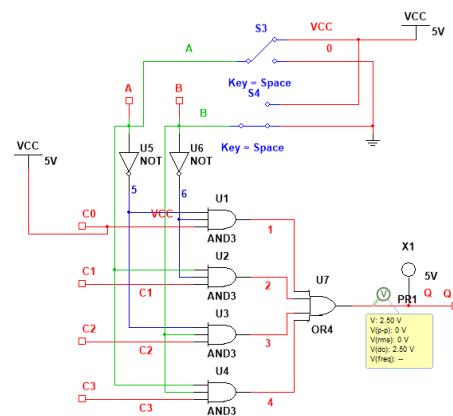
The second part of this assignment will involve the use of a 4 to 1 multiplexer to solve the problem of having multiple inputs connected to a single output. A truth table and Boolean equation shown below will represent the multiplexer being designed.

INPUT	SELECTOR SWITCH		OUTPUT (Q)
	A	B	
C_0	0	0	C_0
C_1	0	1	C_1
C_2	1	0	C_2
C_3	1	1	C_3

From the truth table the Boolean equation is:

$$Q = C_0\bar{A}\bar{B} + C_1\bar{A}B + C_2A\bar{B} + C_3AB$$

These pieces of information allow the design of the multiplexer in multisim, using a probe (X1) to indicate the correct output Q.



To test the design, the A and B circuits were switched between a 5V source or a grounded circuit to imitate wither a logic 1 or 0 respectively. The input options chosen corresponds to the signal input directed to the output, for example selecting A1 and B0 would output C_2 to the Q. This is shown in figure 1 when both inputs are logic 0, Q would equal C_0 Which is the only input that has a voltage source connected to it, this is indicated by the active probe. However, when C_1 is selected the probe is inactive, thus proving its consistency as shown in figure 2.

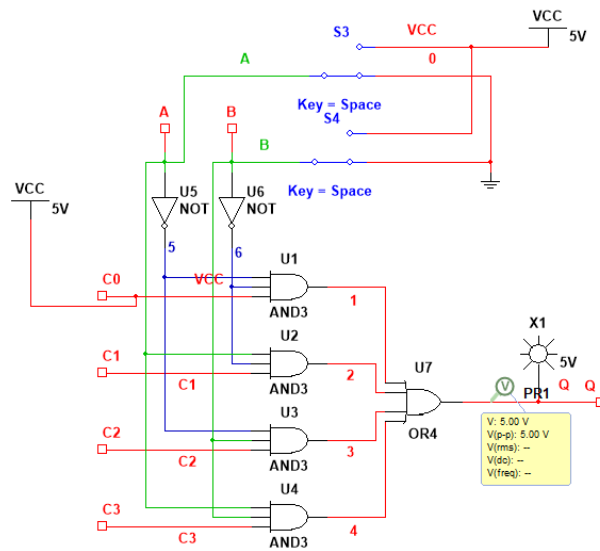


Figure 1 - Multiplexer channel C_0 connected

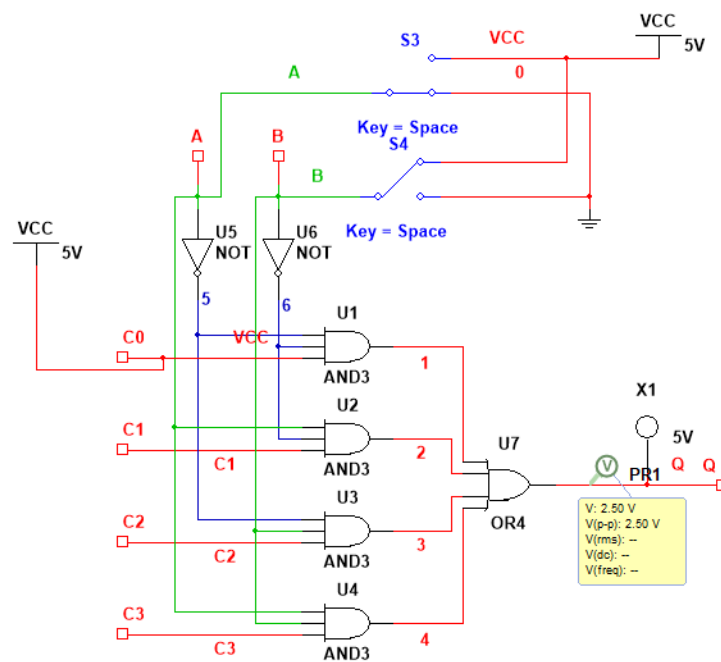


Figure 2 - Multiplexer channel C_1 connected

Task 3 –

Part A -

The aims of this part were to construct a debouncing circuit in multisim and to connect a D- type flip-flop as well as a JK flip-flop to the output to record the Q and Q bar outputs of both the circuits. Using the information gained a truth table for both types of circuit was formed.

The final circuit designed for the D-type shown in figure 3 below represents the debouncing circuit shown in the assignment brief combined with the flip-flop chip.

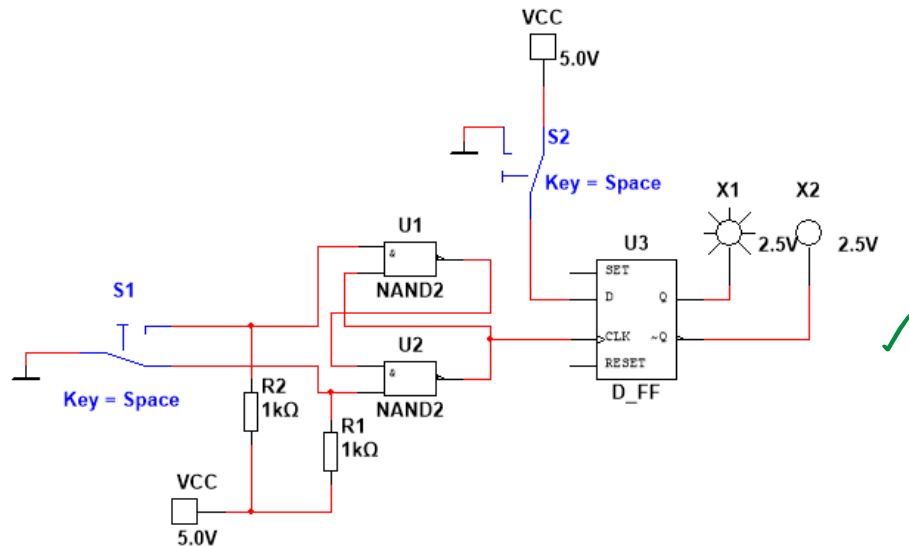


Figure 3 – D – type Flip-flop debouncing circuit

Using the circuit, a truth table was made as shown below.

Inputs		Outputs	
D		Q	$\bar{Q}$
0		0	1
1		1	0

For the second half of task 3 the same was done but with using a JK flip-flop instead of D – type. This would much more likely be used in real world scenarios as the JK is a far more commonly used chip since it is a more adaptable component that utilises more complex logic, most importantly allowing a toggle feature when both inputs are 1 instead of creating an error. Figure 4 below shows the circuit used to complete the truth table.

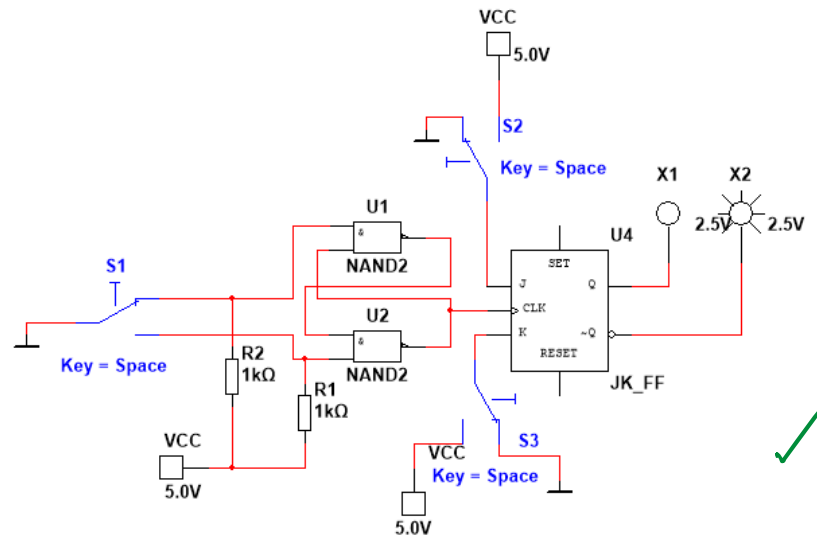


Figure 4 - JK flip – flop Debouncing circuit

The truth table below was formed from the results taken from the JK circuit.

Inputs		Outputs	
J	K	Q	$\bar{Q}$
0	0	0	1
0	1	0	1
1	0	1	0
1	1	0	1

The JK chip operates by having set operations depending on the inputs given. For JK 00, the outputs remain the same with no change, 01 is set, setting Q to 0 with JK 10 is reset putting Q to 1, with Q bar being the opposite. The last input of JK 11 will toggle the outputs depending on the last set of outputs recorded.

for $J = K = 0$ there would be no change.
 $J = K = 1$ the outputs would toggle.

Part B –

Ripple occurs in most forms of mechanical electrical switches. This phenomenon refers to the momentary connections a switch will make with an opposing contact as it moves from one position to another, as it makes contact it will bounce off it multiple times. This produces a ripple in the electrical signal being transmitted. These ripple signals are problematic in digital systems as they can be interpreted as using the switch a number of times each time the switch is used. This is due to the system having an number of 0 and 1 inputs within nano seconds, that would be read as a data stream (A.Ellis, 2014). ✓

To counteract this problem, using a mechanical switch that limits bouncing can be implemented such as a mercury wetted switch, although these are not commonly used due to the toxic nature of the metal (A.Ellis, 2014). ✓

Presently the use of SR latches is a far safer means to eliminate switch bouncing. This can be done by using a number of different designs but for this assignment a NAND SR latch will be used as an example.

An SR latch will store a binary state while there is power to the system. These latches are built by having two NAND gates with the outputs of each gate feeding back into the input of the opposite gate, this is called a cross coupled NAND gate configuration. When the set state is activated the non-inverted Q output is put to a logic 1, alternatively a high reset input results in a inverted Q output. If both set and reset inputs are set to low, the inverted and noninverted Q outputs remain the same.

An issue with this design of this type of latch is that the Q outputs can become out of sync if both inputs are set to high simultaneously and can result in a race condition (Lipiansky, 2012).

A race condition has two possible outcomes, either both gates activate at the same speed which will give an indeterminate output. The other possibility is that one gate operates faster than the other, becoming logic 1 and overriding the other gate and takes control of the state of the latch (Lipiansky, 2012). ✓

Good.

Task 4 –

The final goal of this assignment is to construct a 3-bit synchronous binary counter which incorporates the JK flip flop chip from the previous task. The counter will need to be able to count from 0 to 7 only using even numbers.

To determine the states at which the counter needs to be in to count up evenly, a truth table would need to be devised as shown in figure 5 using the excitation table for JK flip flop chips shown in figure _.

Q	Q bar	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0



Figure 5- JK flip flop excitation table

The Xs within figures 5 and 6 represent don't cares, where the output of either a logic 1 or 0 and that may be used as either for the completion of K maps as necessary.

CLK	Q			Q Bar			C		B		A	
	C	B	A	C	B	B	J	K	J	K	J	K
0	0	0	0	0	0	1	0	X	0	X	1	X
1	0	0	1	0	1	0	0	X	1	X	X	1
2	0	1	0	0	1	1	0	X	X	0	1	X
3	0	1	1	1	0	0	1	X	X	1	X	1
4	1	0	0	1	0	1	X	0	0	X	1	X
5	1	0	1	1	1	0	X	0	1	X	X	1
6	1	1	0	1	1	1	X	0	X	0	1	X
7	1	1	1	0	0	0	X	1	X	1	X	1

Figure 6 - 3-bit binary counter truth table

Using this true table, each state the counter must be in can be uncovered by using a K map as shown below.

The table is incorrect it should only cover the even number and zero.

JC K map

		BA			
		00	01	11	10
C	0	0	0	1	0
	1	X	X	X	X

JC = BA ✓

KC K map

		BA			
		00	01	11	10
C	0	X	X	X	X
	1	0	0	1	0

KC = BA ✓

JB K map

		BA			
		00	01	11	10
C	0	0	1	X	X
	1	0	1	X	X

JB = A ✓

KB K map

		BA			
		00	01	11	10
C	0	X	X	1	0
	1	X	X	1	0

KB = A

JA K map

		BA			
		00	01	11	10
C	0	1	X	X	1
	1	1	X	X	1

JA = 1

KA K map

		BA			
		00	01	11	10
C	0	X	1	1	X
	1	X	1	1	X

KA = 1

it is now possible to construct a 3-bit synchronous circuit using the input results from the K maps. For this all the CLK inputs must share a common source and the probes used must light up to imitate the count of a 3-bit binary counter. Figure 7 shows the concept of the circuit that counts in a standard sequence of 1 to 7.

The maps are correct for your truth table.

using the reset command has left no room for a manual reset switch, which can only be done by completing the counting sequence.

Figures 9 to 11 shows the final in sequence counting performed by figure 8.

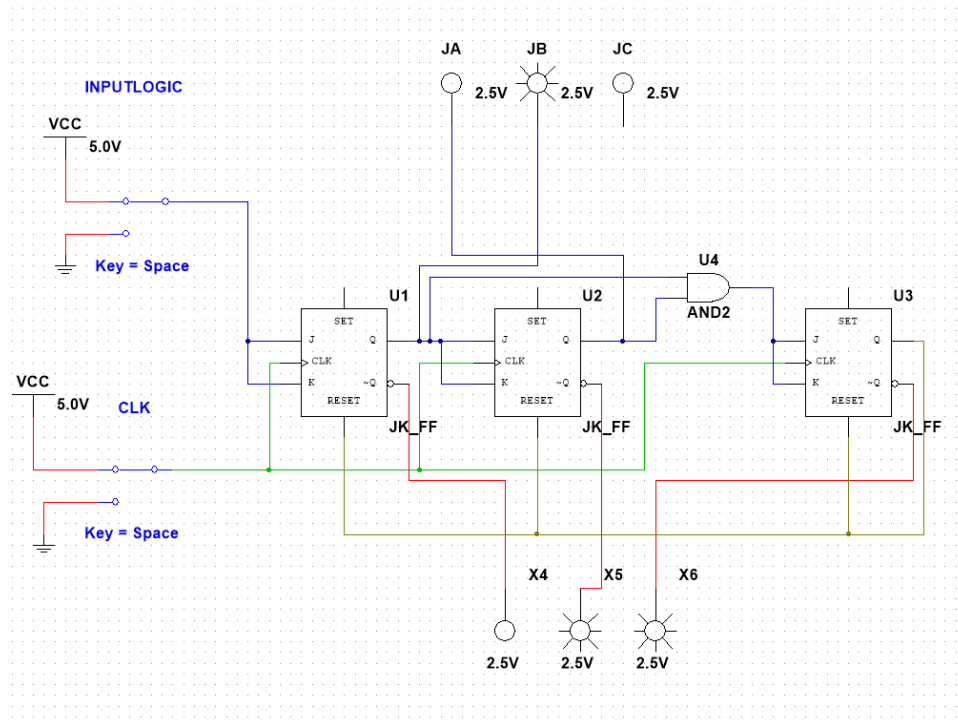


Figure 9 – counter in state 2

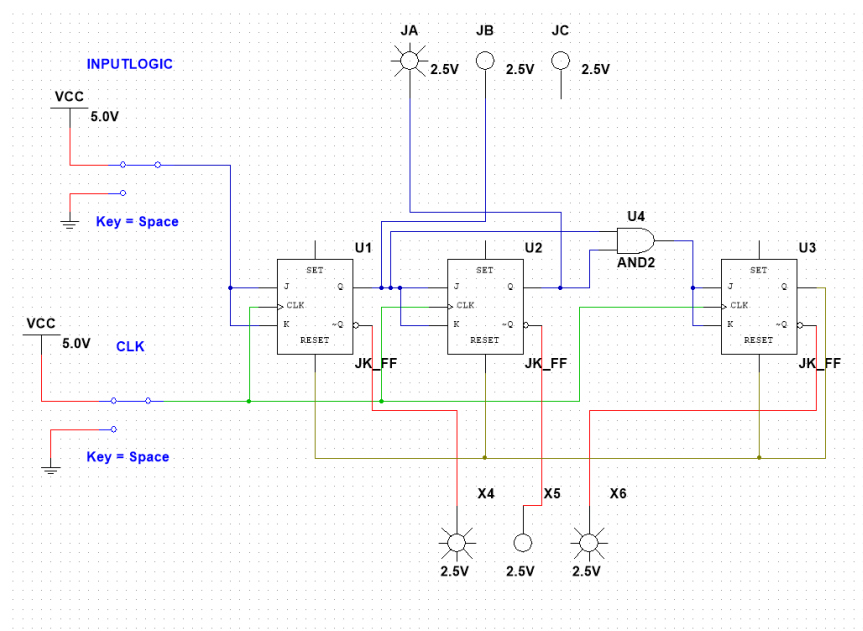


Figure 10 – counter in state 4

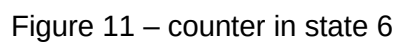


Figure 11 – counter in state 6

References –

Figures 1 to 11 – All figures produced from own work

A.Ellis (2014) Handbook of Electrical Switches. 1st edn. World Technologies.

Lipiansky, E. (2012) Electrical, Electronics, and Digital Hardware Essentials for Scientists and Engineers. 1st edn. Wiley.

Ferdjallah, M. (2011) Introduction to Digital Systems. 1st edn. Wiley.

Would like to see more sources being used.

Bibliography –

A.Ellis (2014) Handbook of Electrical Switches. 1st edn. World Technologies.

Lipiansky, E. (2012) Electrical, Electronics, and Digital Hardware Essentials for Scientists and Engineers. 1st edn. Wiley.

Ferdjallah, M. (2011) Introduction to Digital Systems. 1st edn. Wiley.

