

LEVEL 4 ASSESSMENT SPECIFICATION

Student name:	Oliver Evans	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.

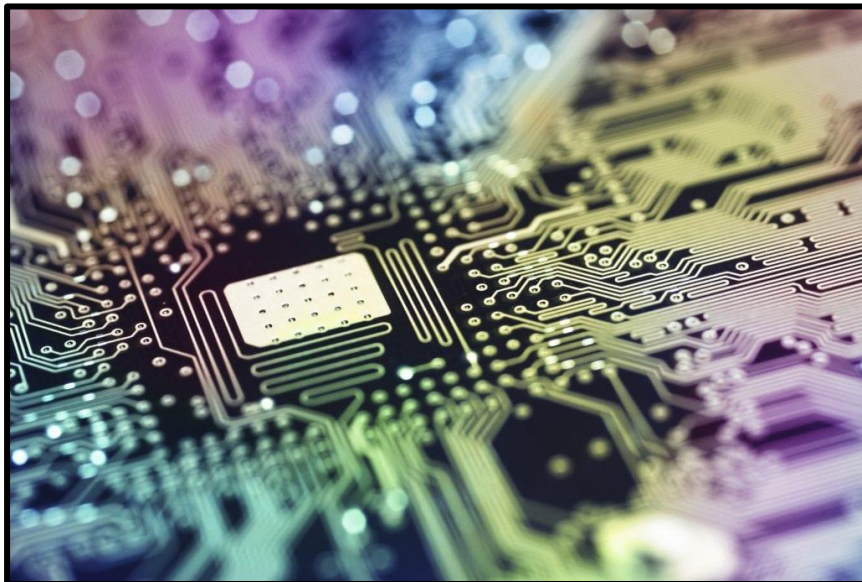
Generic Assessment Criteria	Marks available	Marks awarded
1. Engagement with Literature Skills	10	6.6
Engagement with a wide range of literature sources, including sources retrieved independently. Generally sound referencing, with very few inaccuracies or inconsistencies.		
2. Knowledge and Understanding Skills	45	34.4
Task 1 & 2 achieved 78%. Task 3 & 4 achieved 75%. Excellent knowledge and understanding of the basic underlying concepts and principles of digital systems. Showed excellent understanding the logic gates, Boolean algebra, Karnaugh maps and counter design work.		
3. Cognitive and Intellectual Skills	10	6
Ability to devise arguments using evidence to make mostly appropriate and valid judgements.		
4. Practical Application Skills	25	21
Task 1 & 2 achieved 84%. Task 3 & 4 achieved 84%. An excellent application of practical work for all tasks, very good use of the test equipment and Multisim to obtain the required test results.		
5. Transferable Skills for Life and Professional Practice	10	7.5
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)		76%
	Up to 1 week late (40% Max)		
	Over 1 week late (0%)		



Digital Systems

The design, simulation, build and testing of digital electronic circuits.



Ollie Evans
125854

Assignment 2

PCEE4006
J Pointer

16/02/23

Table of Contents

Assessment Specification.....	1
Front Page.....	2
Table of Contents.....	3
Introduction.....	4
Task 1 – Combinational Logic	4-12
Task 1a.....	4
Task 1b.....	5
Task 1c.....	5-7
Task 1d.....	7-9
Task 1e.....	10
Task 1f.....	11-12
Task 2 – Multiplexer.....	13-14
Task 3 – Sequential Logic.....	15-17
Task 3a.....	15
Task 3b.....	16
Task 3c.....	17
Task 4 – Using Sequential Logic to Solve Real Problems	17-20
Figure Index.....	21
References.....	22
Bibliography.....	22
Acknowledgments.....	22

Introduction

Logic digital systems use exact values in their design and function. Binary code utilises two values to represent data, 0's (off) and 1's (on). These values can be influenced using specific logic gates to get desired outcomes. These systems have a wide range of applications such computers, communication and control systems.

Digital electronics consists of devices such as transistors, relays, diodes and logic gates to process and manipulate digital signals, these signals are represented in binary code as 0's or 1's. Logic gates are the building blocks of logic circuits, such gates include AND, OR, NOT, NAND, NOR and XOR. From these basic gates much more complex circuits can be developed such as counters, multiplexers and comparators.

An extract by Brock J. LaMeres (2017) defines how combining logic gates to create complex circuits operates;

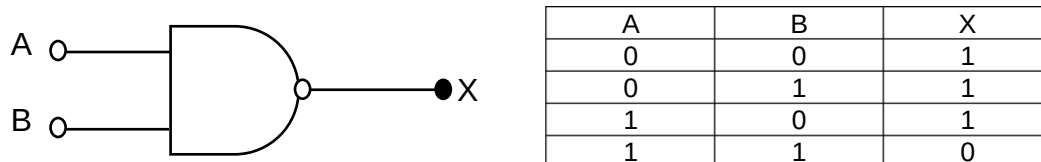
“Combinational logic refers to circuits where the output depends on the present value of the inputs. This simple definition implies that there is no storage capability in the circuitry and a change on the input immediately impacts the output.”

The above extract it explains how instantaneous reaction through a digital system can be to an input change, often delays are implemented to postpone such functions.

A digital system can be designed using a variety of methods such as logic circuit diagram, truth tables, Karnaugh maps and Boolean algebra. These methods clearly show the relationship between the inputs and outputs of the circuit.

Task 1 – Combinational Logic

a)



$$X = \overline{A \cdot B}$$

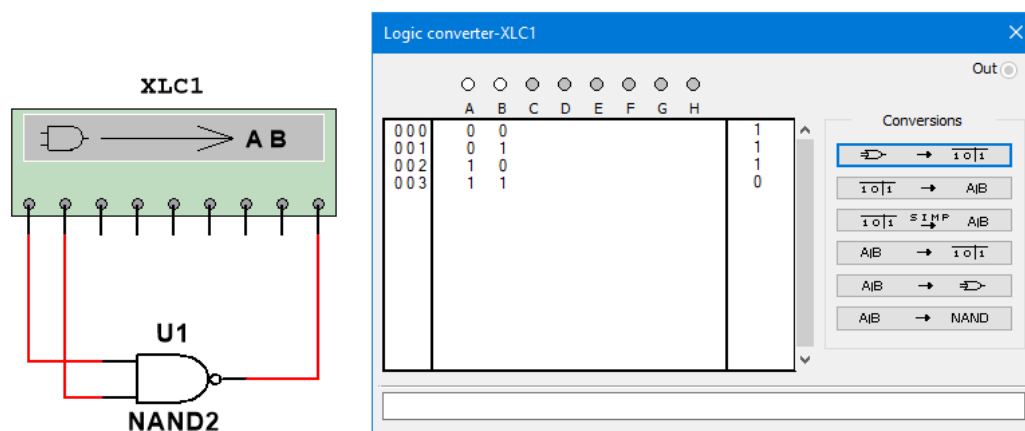


Figure 1.1 – Multisim Task 1a Simulation

Figure 1.1 proves the calculated truth table and Boolean expression for task 1a

b)

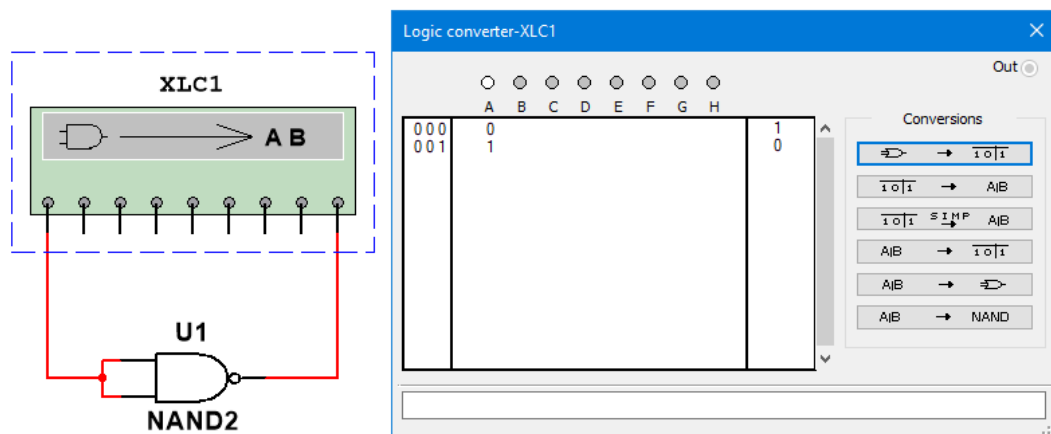
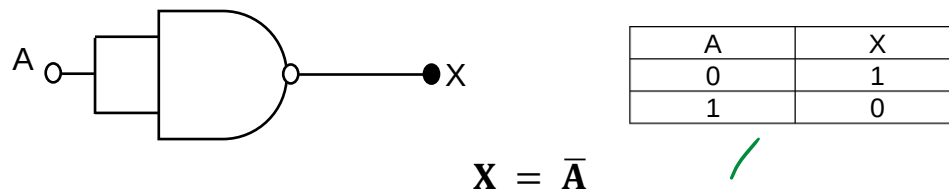
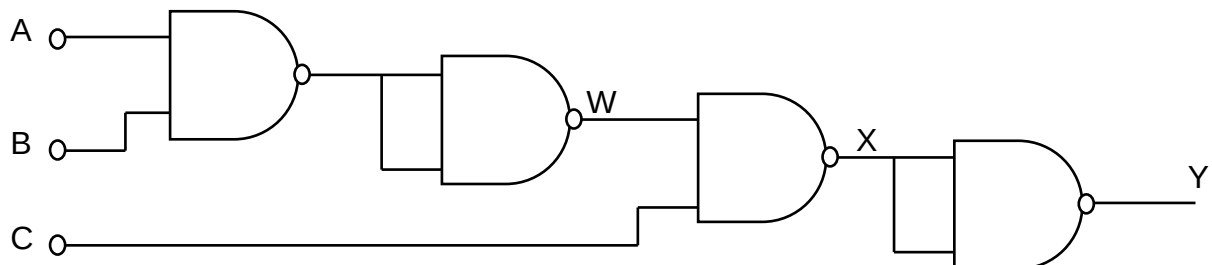


Figure 1.2 proves the calculated truth table and Boolean expression for task 1b

c)

Logic Diagram



Truth Table

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

$Y = ABC$

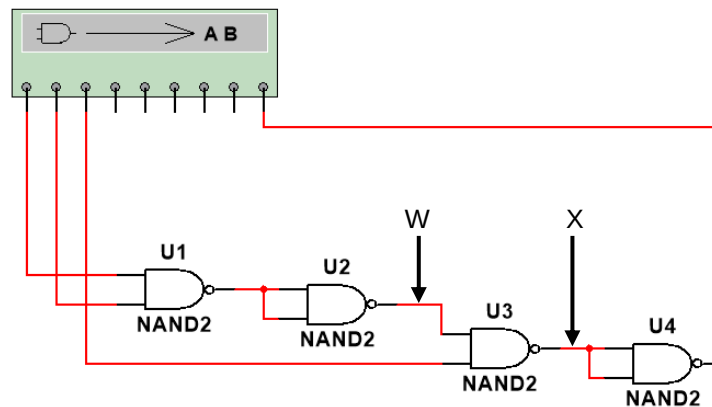


Figure 1.3 – Multisim Task 1c Circuit Diagram

Figure 1.3 shows the circuit layout used with a logic converter to prove the truth table above. It displays the circuit layout for column Y, with the input for the logic converter for W and X column labelled.

	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

Figure 1.4 – Multisim Column W Logic Converter Table

Figure 1.2 proves the calculated truth table for column W for task 1c. The output for W is the same as a 2 input AND gate as A and B had to be positive to get a positive output. Where the C value does not matter as it has no effect.

	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

Figure 1.5 – Multisim Column X Logic Converter Table

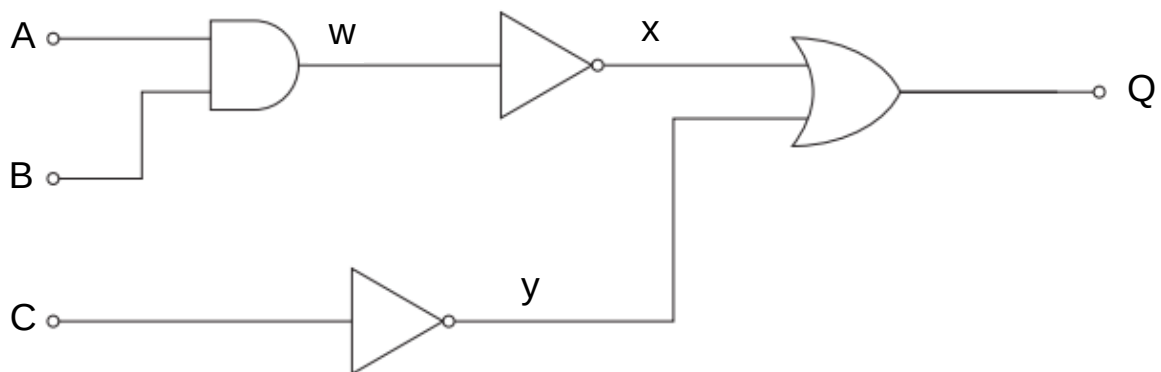
Figure 1.5 proves the calculated truth table for column X for task 1c. The output of X is the same as that of a 3 input NAND gate, as all inputs must be positive to achieve a negative output whereas all other conditions achieve a positive output.

	○	○	○	●	●	●	●	●
	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

Figure 1.6 – Multisim Column Y Logic Convertor Table

Figure 1.6 proves the calculated truth table for column Y for task 1c. The final output of Y is the same as a 3 input AND gate, as all inputs have to be positive to achieve a positive output all other conditions besides this achieves only negative outputs.

d)

Logic Diagram**Truth Table**

A	B	C	w	x	y	Q
0	0	0	0	1	1	1
0	0	1	0	1	0	1
0	1	0	0	1	1	1
0	1	1	0	1	0	1
1	0	0	0	1	1	1
1	0	1	0	1	0	1
1	1	0	1	0	1	1
1	1	1	1	0	0	0

$$Q = \overline{A}\overline{B}\overline{C} + \overline{A}\overline{B}C + \overline{A}B\overline{C} + \overline{A}BC + A\overline{B}\overline{C} + A\overline{B}C + AB\overline{C}$$

Karnaugh Map

A \ BC				
	00	01	11	10
0	1	1	1	1
1	1	1	0	1

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

$$/ = \overline{A \cdot B \cdot C}$$

NAND Gate Circuit

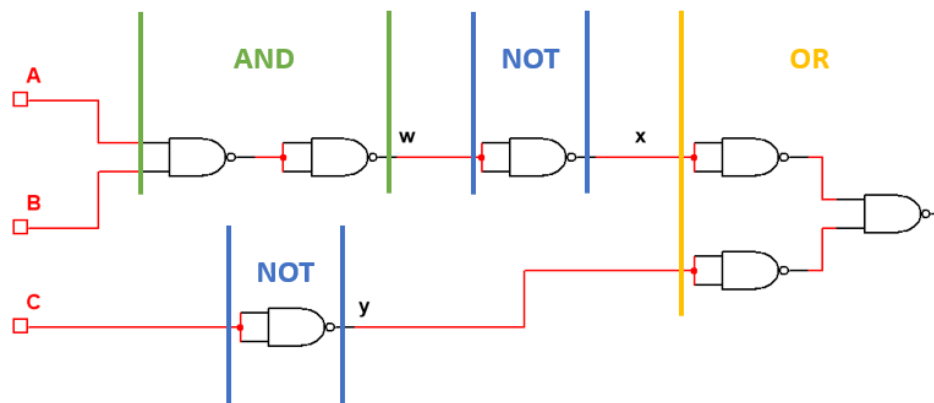


Figure 1.7 – Multisim NAND Gate Equivalent Circuit

Figure 1.7 is the NAND equivalent circuit for task 1d with annotations showing the logic gate conversion into NAND gates.

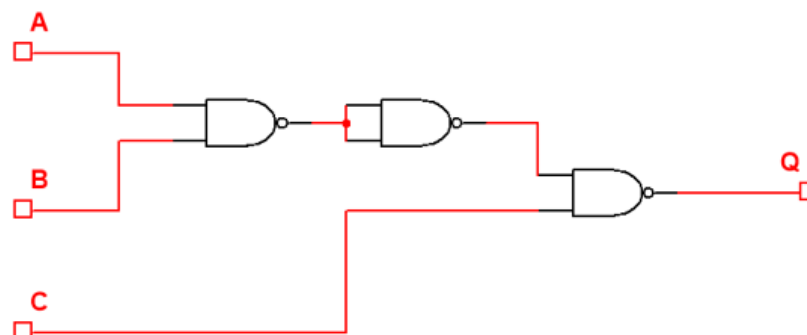


Figure 1.8 – Multisim NAND Gate Simplified Circuit

Figure 1.8 is the simplified circuit of figure 1.7. When one input is shared across both NAND gate inputs it functions as a NOT gate, when there is two of these in series the input inverts through one NAND gate then reverts back through the second NAND gate to the same state prior, when this happens the two NAND gates can be removed having no affect on the system.

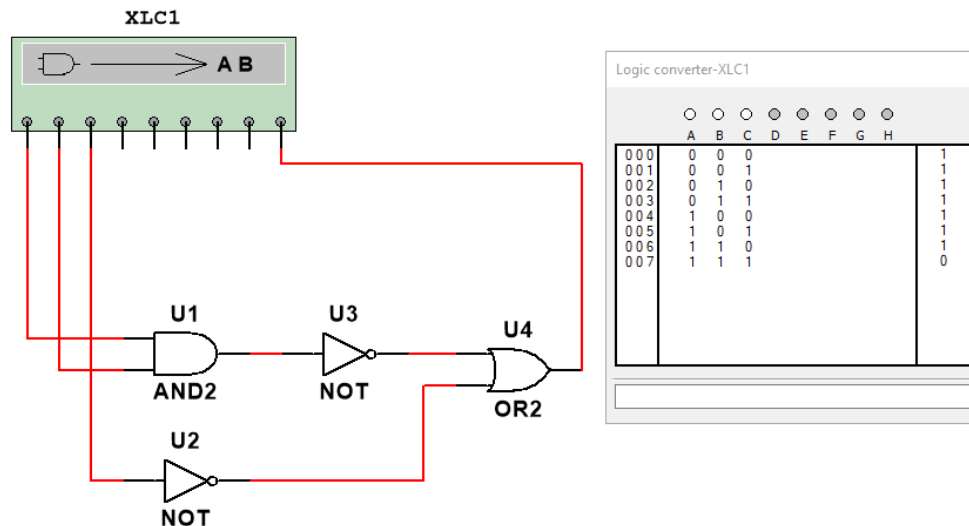


Figure 1.9 – Original Circuit Simulation

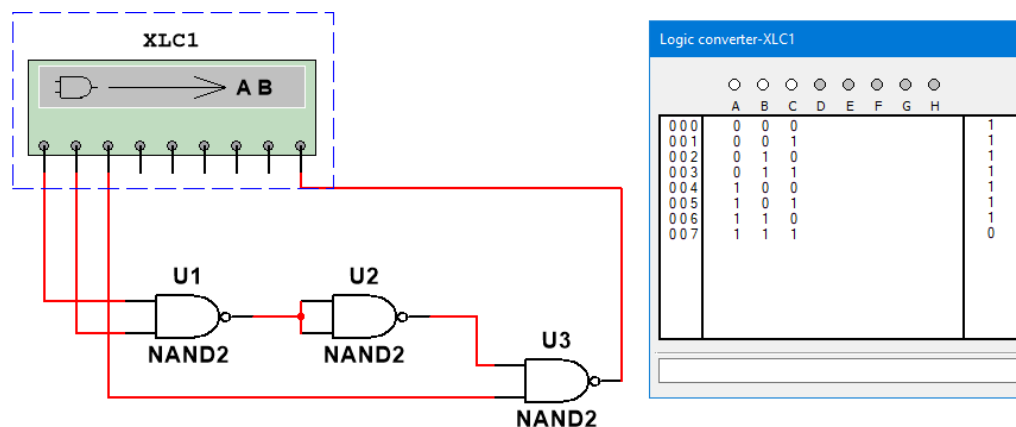


Figure 1.10 – NAND Equivalent Circuit Simulation

When comparing the two truth tables in figure 1.9 and figure 1.10, the function is the same. Not only is there less gates, in the original circuit three separate chips would be needed. Once simplified one chip is needed for the same operation.

A NAND gate performs the logic operation of an inverted AND gate, this is where all inputs must be 1 for a 0 output and all other combinations would generate a 1 output. NAND gates as seen from figure 1.7 can be direct replacement for all other logic gates when placed in various formations, from this initial conversion they can then be simplified as shown in figure 1.8.

There is potential for all gates to be replicated by NOR gates but using NAND gates as a universal gate is more favoured as outlined by M. Morris Mano & Kime, (2004) in the following;

“NAND implementation is less costly. Also, the NAND implementation involves a maximum of three gates in series while the NOR implementation has a maximum of five gates in series. With equal gate delays assumed, the shorter series of gates in the NAND circuit gives a maximum delay from an input change to a corresponding output change about 0.6 times as long as that for the NOR circuit. So, in this particular case, the NAND circuit is superior to the NOR circuit in both cost and delay.”

Implementing NAND gates as a universal gate greatly reduces costs, as less variety of chips would be required and the total numbers of chips would likely be less all gates on the chip is more likely to be used.

e)
i)

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

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

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

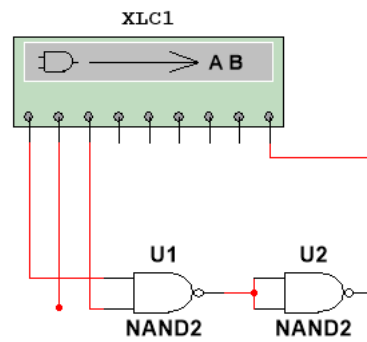
$$Q = (A.(1 + \bar{B} + B)).(\bar{A} + C)$$

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

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

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

$$Q = A.C$$



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					1
006	1	1	0					0
007	1	1	1					1

AC

Figure 1.11 – Boolean Circuit Evidence i

ii)

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

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

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

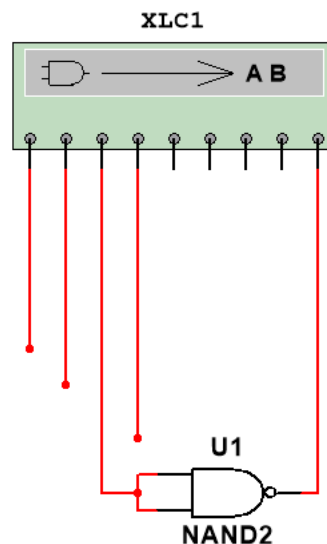
$$Q = \bar{C} (A(B + 1) + \bar{A}.D + \bar{A}.\bar{D})$$

$$Q = \bar{C} (A.1 + \bar{A}.(D + \bar{D}))$$

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

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

$$Q = \bar{C}$$



Logic converter-XLC1

	A	B	C	D	E	F	G	H
000	0	0	0	0				1
001	0	0	0	1				1
002	0	0	1	0				0
003	0	0	1	1				0
004	0	1	0	0				1
005	0	1	0	1				1
006	0	1	1	0				0
007	0	1	1	1				0
008	1	0	0	0				1
009	1	0	0	1				1
010	1	0	1	0				0
011	1	0	1	1				0
012	1	1	0	0				1
013	1	1	0	1				1
014	1	1	1	0				0
015	1	1	1	1				0

C

Figure 1.12 – Boolean Circuit Evidence ii

From figure 1.11 & 1.12 there are values that are terminated to nothing, this was done to show 3-bit and 4-bit tables showing the total function of the circuits. This shows how impactful simplifying logic circuits can be, this can save a company money as total number of chips would be reduced, as well as troubleshooting the system would be much more efficient.

f)

Truth Table

A	B	C	D	Q
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	0
0	1	1	0	0
0	1	1	1	0
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	1
1	1	1	1	1

Above is a truth table for BCD (binary coded decimal) where the numbers ranging from 1010 to 1111 are not used, for the system to detect these invalid codes the Q output will show one to indicate this.

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

Karnaugh Map

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

$$Q = AB + AC$$

$$Q = AB + AC \quad \checkmark$$

Invert All

$$\overline{Q} = \overline{AB + AC} \quad \checkmark$$

$$\overline{Q} = \overline{AB} \cdot \overline{AC} \quad \checkmark$$

Invert Again

$$Q = \overline{\overline{AB} \cdot \overline{AC}} \quad \checkmark$$

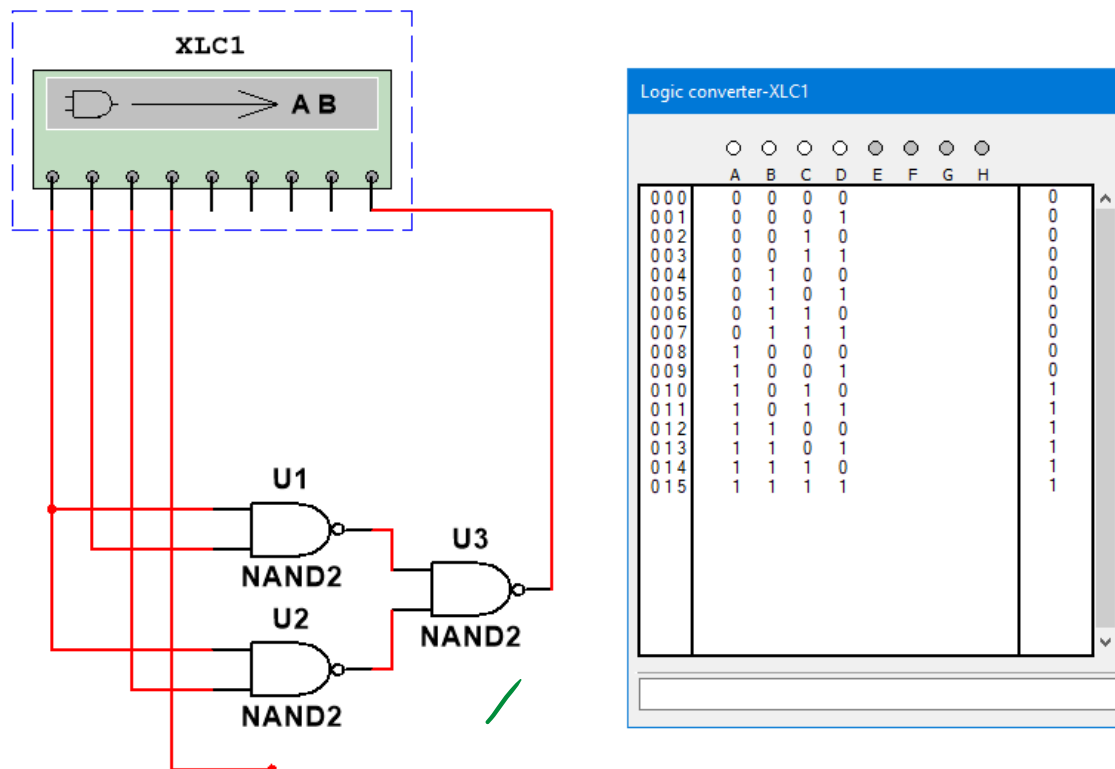


Figure 1.13 – 4-Bit Binary Code Boolean Evidence

When comparing figure 1.13 truth table with the original truth table the outputs and logic functions are identical.

The original Boolean expression was then simplified by means of a Karnaugh map to give the expression $Q = AB + AC$. By using DeMorgan's theorem the expression was further simplified to give $Q = \overline{\overline{AB} \cdot \overline{AC}}$, from this simplification the requirement for 2 AND gates and 1 OR gate has changed to only requiring 3 NAND gates.

In figure 1.13 proves that D inputs have no affect on the output of this logic circuit so when conducting the simulation, the D input was terminated to nothing. The reason for this is to display a 4-bit system to prove figure 1.13 match with the original truth table.

Task 2 - Multiplexer

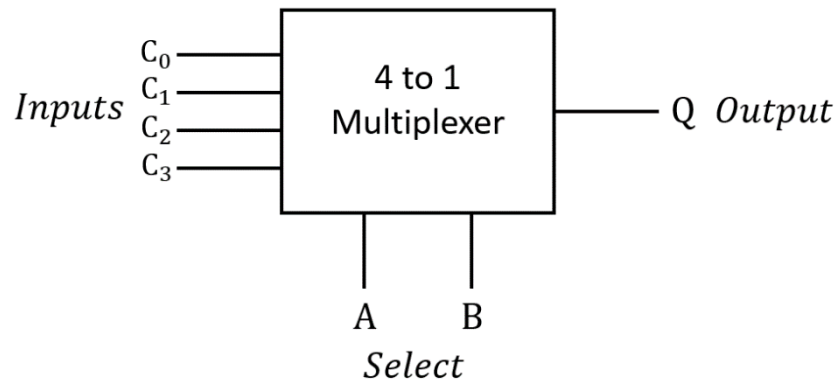


Figure 2.1 – Multiplexer Diagram

Truth Table

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

$$Q = C_0 \cdot \bar{A} \cdot \bar{B} + C_1 \cdot \bar{A} \cdot B + C_2 \cdot A \cdot \bar{B} + C_3 \cdot A \cdot B$$

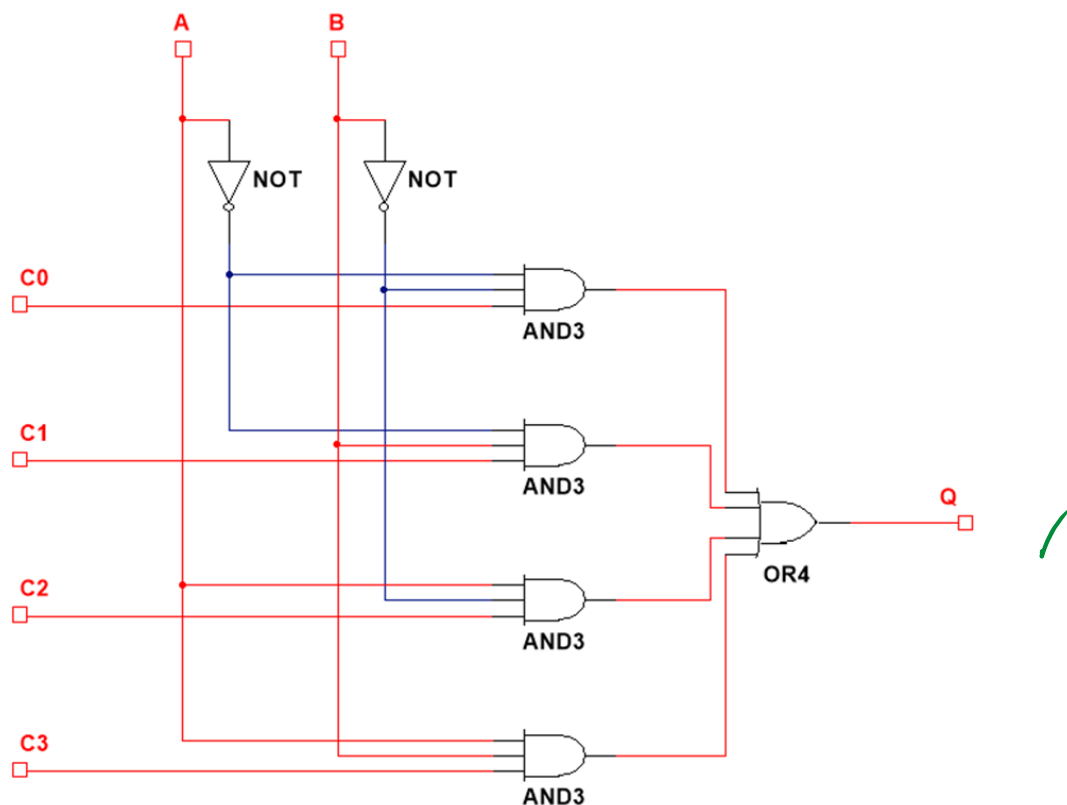


Figure 2.2 – Multiplexer Multisim Circuit

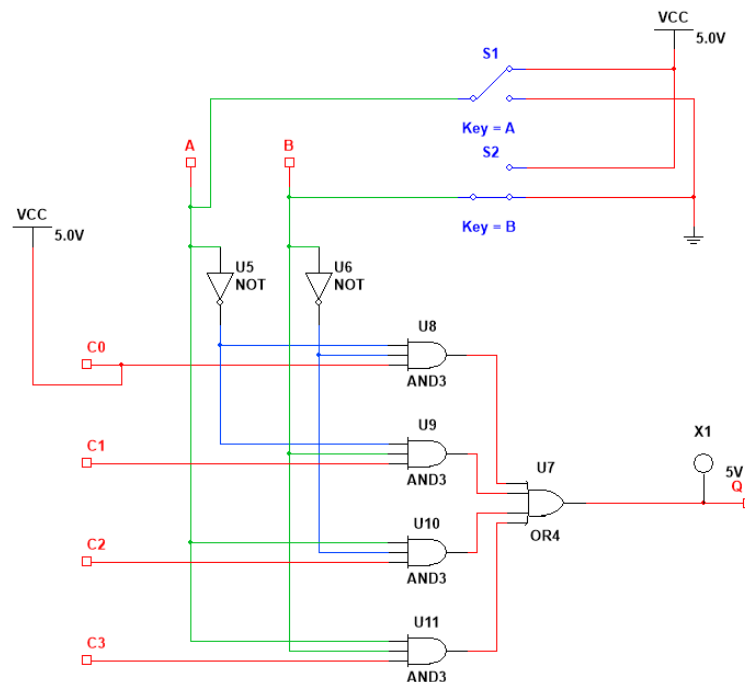


Figure 2.3 – Multiplexer Multisim Simulation 1

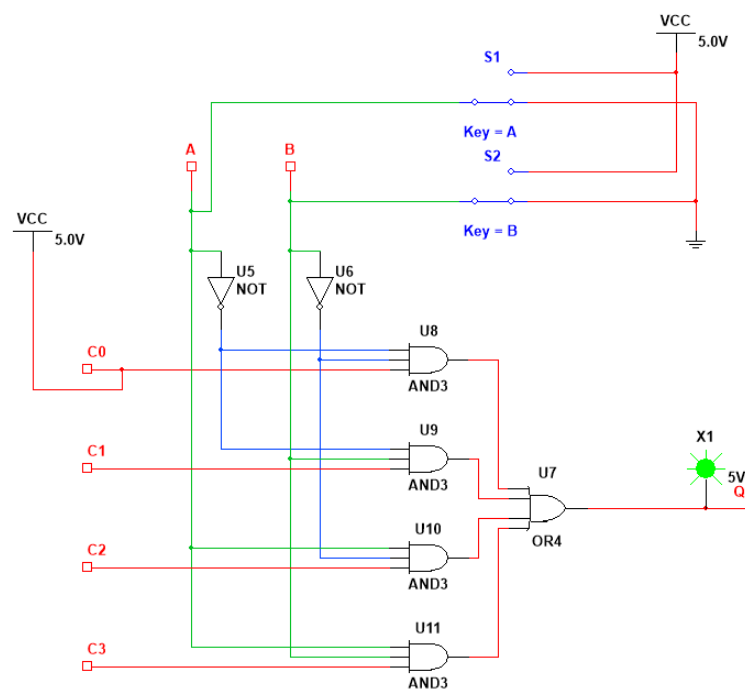


Figure 2.4 – Multiplexer Multisim Simulation 2

Figure 2.4 & 2.5 display proof of circuit function. Figure 2.4 is an example of the input $C_0 \cdot A \cdot \bar{B}$ hence why the output on U7 (4 input OR gate) is 0. Where in figure 2.5 the input is $C_0 \cdot \bar{A} \cdot \bar{B}$ getting a 1 output. Testing was carried out for C_1 , C_2 & C_3 proving circuit function ensuring that it followed the Boolean expression. ✓

Task 3 - Sequential Logic

a)

Truth Table

Input	Output	
D	Q	$\bar{Q}$
0	0	1
1	1	0

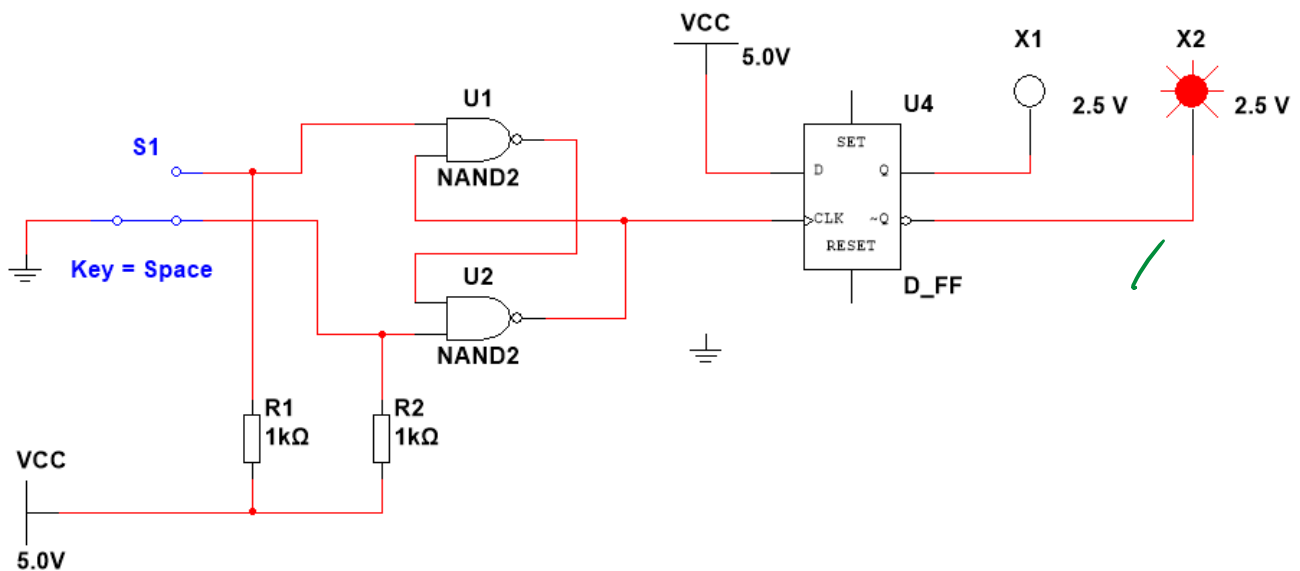


Figure 3.1 – D-Type Circuit Simulation 1

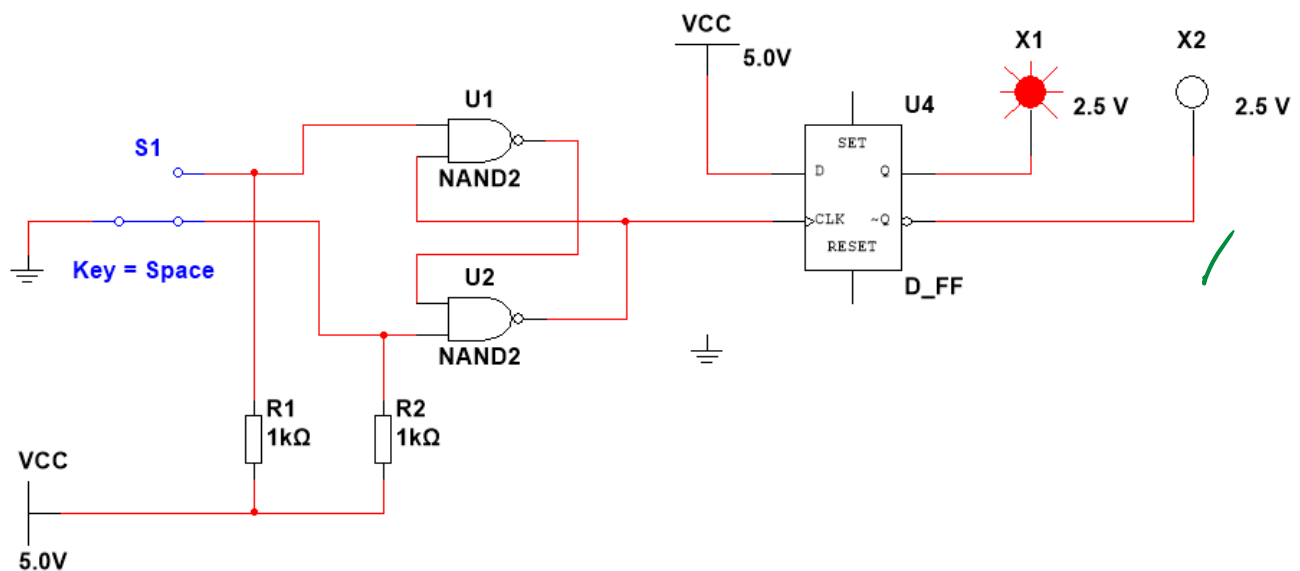


Figure 3.2 – D-Type Circuit Simulation 2

Figure 3.2 shows the beginning of the D type sequence, after one clock pulse figure 3.3 shows the D type flip flop moving the data from D in this case 1 to the output of Q. $\bar{Q}$ is always the invert of Q. A further test was conducted with ground being connected to D to prove the D-type moving a 0 input through to Q after a clock pulse.

b)

Truth Table

J	K	CLK	Q	$\bar{Q}$
0	0	$\uparrow$	Q_0 (No Change)	Q_0 (No Change)
1	0	$\uparrow$	1	0
0	1	$\uparrow$	0	1
1	1	$\uparrow$	$\bar{Q}_0$ (Toggles)	$\bar{Q}_0$ (Toggles)

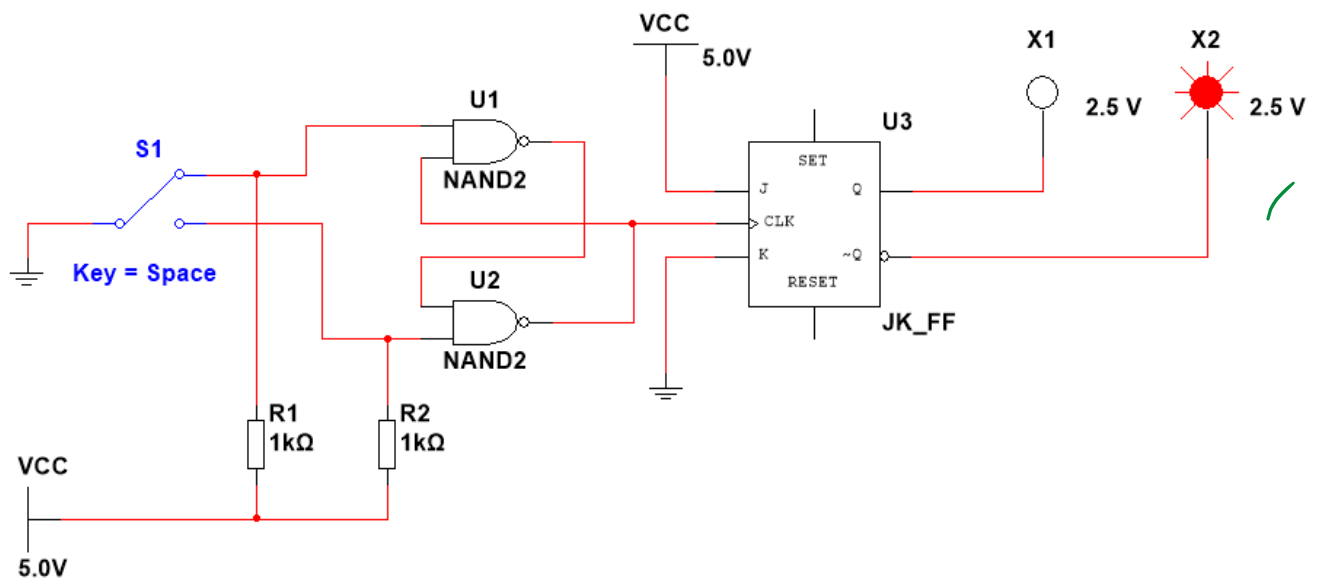


Figure 3.3 – JK-Type Circuit Simulation 1

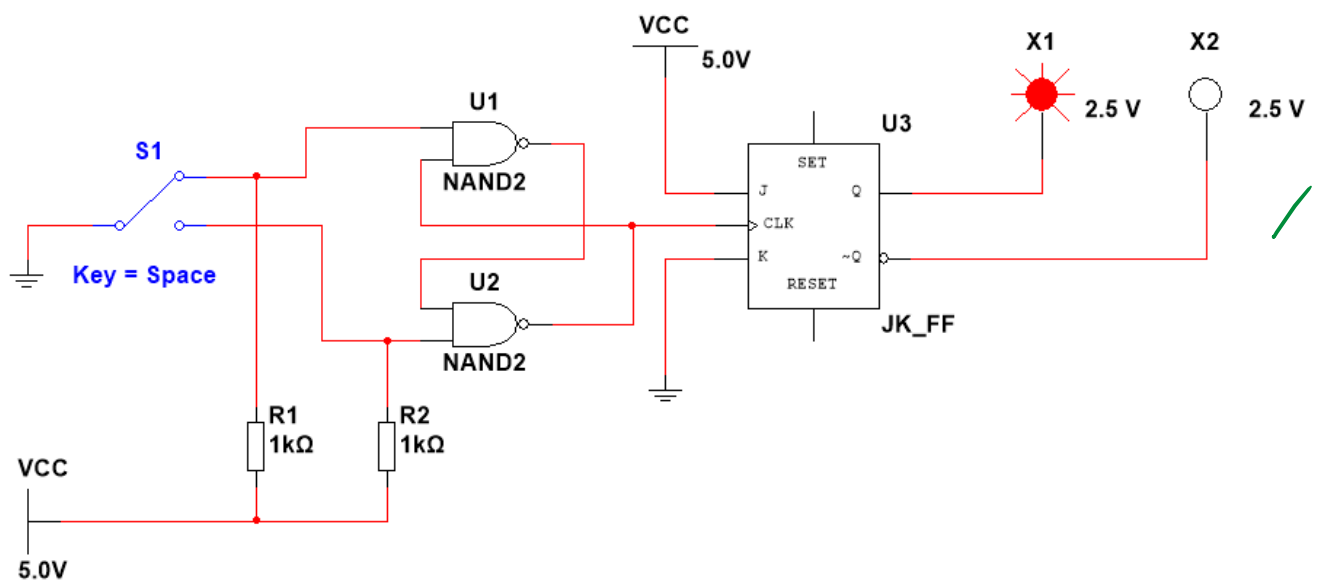


Figure 3.4 – JK-Type Circuit Simulation 2

From the simulation in figure 3.3 and figure 3.4 is an example of highlighted yellow row of the truth table, there is a high input on J and a low input on K resulting in a high output on Q post clock pulse. The set function of the flip flop overrides the J and K input making the output Q 1, the reset function will revert Q back to 0.

c)

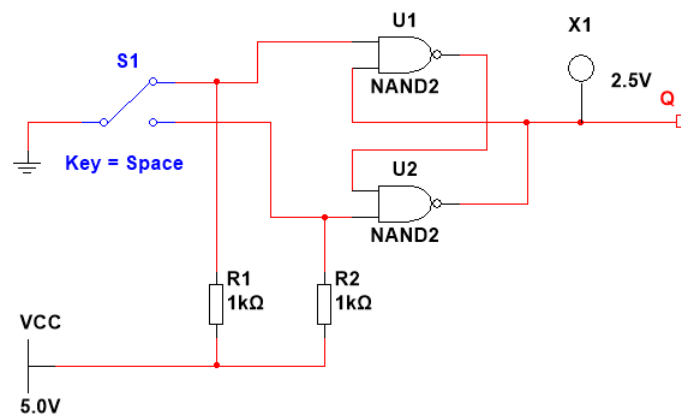


Figure 3.5 – Debouncing Multisim Circuit

When the electrical contacts within a mechanical switch meet there is often a series of ripple signals from the elasticity of the contact material making and breaking. Figure 3.5 is an example of debouncing this is done by use of a S-R latch circuit, this circuit minimises the ripple effect from the switch. Research by Gay, W. (2015) investigates the signal produced by a non debounced switch and a corrected debounced switch seen in figure 3.6. When looking at the bouncing signal the system would see it as multiple high's and low's instead of the intended single on.



Figure 3.6 – Bouncing Waveform Versus Ideal Debounced Switch

This circuit seen in figure 3.5 consists of two NANDs, a SPDT switch and two pull up resistors. The resistors provide a logic high for the NAND gates, the switch pulls one of the inputs to ground. In the circuit diagram one of the inputs for U1 would be pulled to ground making a logic low, regardless of the other input a NAND gate would output a logic high. U2 is connected to VCC though R2 this NAND gate has the input high from U1 and high from VCC. U2 will continuously give out a logic low to U1 so in the event of bouncing U1 would have one input bouncing but because it maintains a logic low from U2 it holds the value eliminating bouncing at the Q point.

Task 4 - Using Sequential Logic to Solve Real Problems

Excitation Table

Present Qa	Next Qa	Ja	Ka
0 →	0	0	x
0 →	1	1	x
1 →	1	x	0
1 →	0	x	1

Above is an excitation table which shows the minimum required inputs to achieve the next state. This had to be completed for all Q values in the table below. The x value stands for a don't care condition this could have the value of 0 or 1, it does not affect the result as it does not matter.

Truth Table

	Present			Next			JK-a		JK-b		JK-c	
	Qa	Qb	Qc	Qa	Qb	Qc	Ja	Ka	Jb	Kb	Jc	Kc
0	0	0	0	0	1	0	0	x	1	x	0	x
2	0	1	0	1	0	0	1	x	x	1	0	x
4	1	0	0	1	1	0	x	0	1	x	0	x
6	1	1	0	0	0	0	x	1	x	1	0	x

Karnaugh Maps

JK-a

Qa \ Qb Qc	00	01	11	10
0	0	X	X	1
1	X	X	X	X

$$J_a = Q_b$$

Qa \ Qb Qc	00	01	11	10
0	X	X	X	X
1	0	X	X	1

$$K_a = Q_b$$

JK-b

Qa \ Qb Qc	00	01	11	10
0	1	X	X	X
1	1	X	X	X

$$J_b = 1$$

Qa \ Qb Qc	00	01	11	10
0	X	X	X	1
1	X	X	X	1

$$K_b = 1$$

JK-c

Qa \ Qb Qc	00	01	11	10
0	0	X	X	0
1	0	X	X	0

$$J_c = 0$$

Qa \ Qb Qc	00	01	11	10
0	X	X	X	X
1	X	X	X	X

$$K_c = 0$$

The green don't care in the Karnaugh maps relate the figures that would be used for numbers 1, 3, 5 & 7. As this counter is not using these numbers at all they were highlighted don't cares prior to building the k maps.

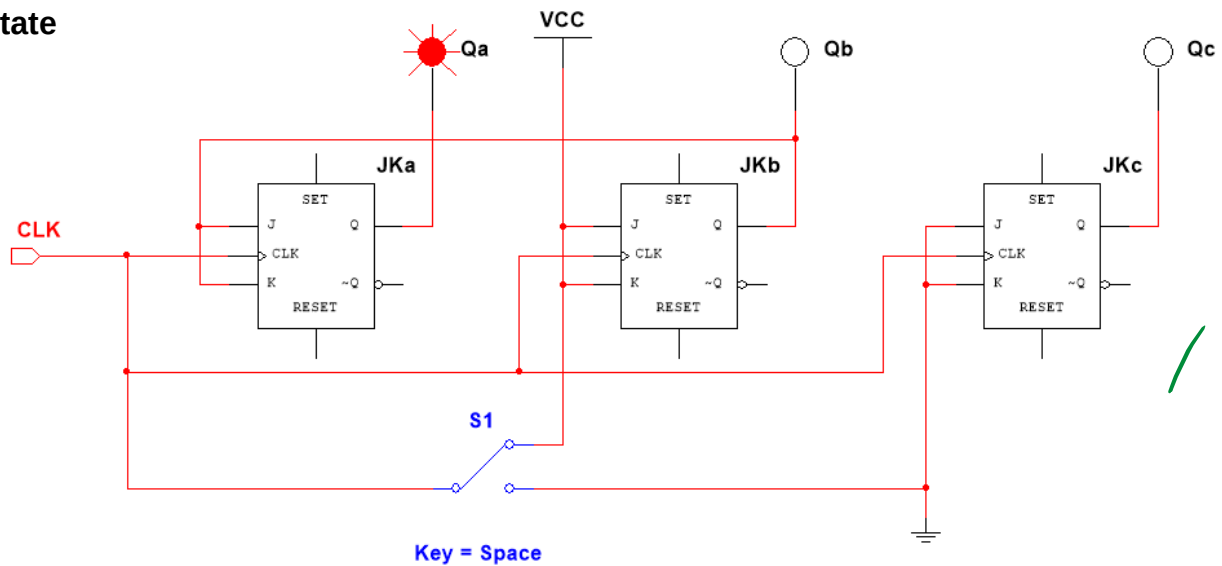
100 State

Figure 4.4 – Multisim Sequential Logic Circuit – 100 State (4)

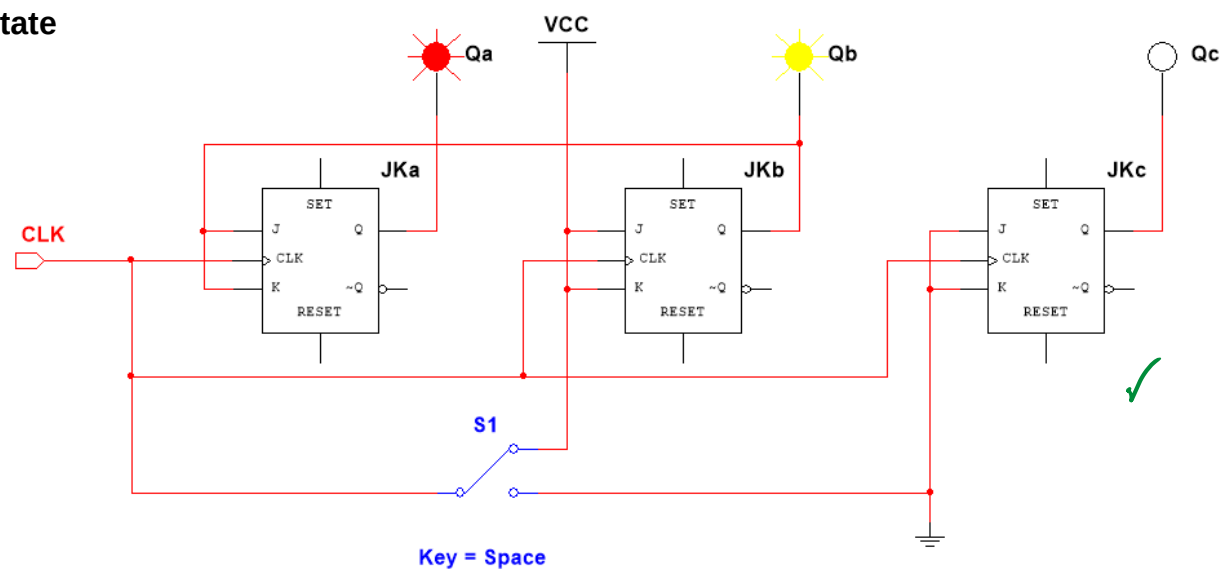
110 State

Figure 4.5 – Multisim Sequential Logic Circuit – 110 State (6)

Figure 4.2 through to figure 4.5 shows the sequence of counting in line with the brief, the brief stated to have a 3-stage binary counter to produce only even numbers in the range 0 to 7. From previous figures the brief has been met. This counter will restart as in the truth table the next Q values after 6 would bring the counter back to 0.

Excellent work.

Figure Index

Figure 1.1 – Multisim Task 1a Simulation – Circuit and table simulated in Multisim by Ollie Evans

Figure 1.2 – Multisim Task 1b Simulation – Circuit and table simulated in Multisim by Ollie Evans

Figure 1.3 – Multisim Task 1c Circuit Diagram – Circuit simulated in Multisim by Ollie Evans

Figure 1.4 – Multisim Column W Logic Convertor Table – Table simulated in Multisim by Ollie Evans

Figure 1.5 – Multisim Column X Logic Convertor Table – Table simulated in Multisim by Ollie Evans

Figure 1.6 – Multisim Column Y Logic Convertor Table – Table simulated in Multisim by Ollie Evans

Figure 1.7 – Multisim NAND Gate Equivalent Circuit – Circuit simulated in Multisim by Ollie Evans

Figure 1.8 – Multisim NAND Gate Simplified Circuit – Circuit simulated in Multisim by Ollie Evans

Figure 1.9 – Original Circuit Simulation – Circuit and table simulated in Multisim by Ollie Evans

Figure 1.10 – NAND Equivalent Circuit Simulation – Circuit and table simulated in Multisim by Ollie Evans

Figure 1.11 – Boolean Circuit Evidence i – Circuit and table simulated in Multisim by Ollie Evans

Figure 1.12 – Boolean Circuit Evidence ii – Circuit and table simulated in Multisim by Ollie Evans

Figure 1.13 – 4-Bit Binary Code Boolean Evidence – Circuit and table simulated in Multisim by Ollie Evans

Figure 2.1 – Multiplexer Diagram – Diagram created in Powerpoint by Ollie Evans

Figure 2.2 – Multiplexer Multisim Circuit – Circuit simulated in Multisim by Ollie Evans

Figure 2.3 – Multiplexer Multisim Simulation 1 – Circuit simulated in Multisim by Ollie Evans

Figure 2.4 – Multiplexer Multisim Simulation 2 – Circuit simulated in Multisim by Ollie Evans

Figure 3.1 – D-Type Circuit Simulation 1 – Circuit simulated in Multisim by Ollie Evans

Figure 3.2 – D-Type Circuit Simulation 2 – Circuit simulated in Multisim by Ollie Evans

Figure 3.3 – JK-Type Circuit Simulation 1 – Circuit simulated in Multisim by Ollie Evans

Figure 3.4 – JK-Type Circuit Simulation 2 – Circuit simulated in Multisim by Ollie Evans

Figure 3.5 – Debouncing Multisim Circuit – Circuit simulated in Multisim by Ollie Evans

Figure 3.6 – Bouncing Waveform Versus Ideal Debounced Switch – Gay, W. (2015). Exploring the Raspberry Pi 2 with C++. Berkeley, CA: Apress : Imprint : Apress.

Figure 4.1 – Sequential Logic Circuit Multisim – Circuit simulated in Multisim by Ollie Evans

Figure 4.2 – Multisim Sequential Logic Circuit – 000 State (0) – Circuit simulated in Multisim by Ollie Evans

Figure 4.3 – Multisim Sequential Logic Circuit – 010 State (2) – Circuit simulated in Multisim by Ollie Evans

Figure 4.4 – Multisim Sequential Logic Circuit – 100 State (4) – Circuit simulated in Multisim by Ollie Evans

Figure 4.5 – Multisim Sequential Logic Circuit – 110 State (6) – Circuit simulated in Multisim by Ollie Evans

References

Catalyst. (2018). Digital System Design – Crash the Course [online] Available at: <https://wp.wpi.edu/catalyst/2018/01/24/crash-the-course-digital-system-design/>

LaMeres, B.J. (2017). Introduction to Logic Circuits & Logic Design with Verilog. Springer.

M. Morris Mano and Kime, C.R. (2004). Logic and Computer Design Fundamentals. Prentice Hall.

Gay, W. (2015). Exploring the Raspberry Pi 2 with C++. Berkeley, CA: Apress : Imprint : Apress.

Bibliography

Wakerly J F. Digital Design Principles and Practices. Prentice Hall. ISBN: 8120330218 (2005)

Storey N. Electronics, A Systems Approach, 6th Edition. Pearson. ISBN: 0273719181 (2017)

Maddock R J, Calcutt D M. Electronics for Engineering. Prentice Hall. ISBN: 9780582215832 (1994)

Tooley M. Electronic Circuits: Fundamentals and applications, 4th Edition. Routledge. ISBN: 9781138828926 (2015)

Green D C. Digital Electronics. Longman. ISBN: 9780582317369 (1998)

National Instruments Corp (2021). Product Documentation - NI. [online] Available at: <https://www.ni.com/docs/en-US/bundle/Multisim/resource/374483d.pdf>.

Floyd, T.L. and Pearson (2018). Electronic devices : conventional current version. Harlow, England I Pozostale: Pearson Education Limited.

Hambley, A.R. (2005). Electrical Engineering. Prentice Hall.

Katz, R.H. (1994). Contemporary Logic Design. Prentice Hall.

Karnaugh, M., 1953. The map method for synthesis of combinational logic circuits. *Transactions of the American Institute of Electrical Engineers, Part I: Communication and Electronics*, 72(5), pp.593-599.

Holdsworth, B. and Woods, C., 2002. Digital logic design. Elsevier.

Catalyst. (2018). Digital System Design – Crash the Course [online] Available at: <https://wp.wpi.edu/catalyst/2018/01/24/crash-the-course-digital-system-design/>

LaMeres, B.J. (2017). Introduction to Logic Circuits & Logic Design with Verilog. Springer.

M. Morris Mano and Kime, C.R. (2004). Logic and Computer Design Fundamentals. Prentice Hall.

Gay, W. (2015). Exploring the Raspberry Pi 2 with C++. Berkeley, CA: Apress : Imprint : Apress. ✓