

LEVEL 4 ASSESSMENT SPECIFICATION

Student name:	Dylan John	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	4.8
Engagement with a good range of literature sources, including sources retrieved independently. Generally sound referencing, with very few inaccuracies. The presentation of the reference list is a bit untidy.		
2. Knowledge and Understanding Skills	45	29.5
Task 1 & 2 achieved 56%. Task 3 & 4 achieved 75%. Knowledge of the basic underlying concepts and principles is competent with a very good understanding in most aspects of digital systems. Showed very good understanding the logic gates, Boolean algebra, Karnaugh maps and counter design work.		
3. Cognitive and Intellectual Skills	10	4.2
Some evidence to support emerging arguments and judgements but in some places they are underdeveloped or missing.		
4. Practical Application Skills	25	19.5
Task 1 & 2 achieved 72%. 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	5.8
Mostly coherent, organised and accurate work, in a suitable structure and is for the most part clearly expressed.		

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)

Up to 1 week late (40% Max)

Over 1 week late (0%)

64%

HNC Electrical/Electronic Engineering

Module: Electronics

Module code: PCEE4006



Student name: Dylan John

Student ID number: 157427

Lecturer: Jason pointer

Contents page

Contents page

Introduction	Page 4
Background theory on sequential logic	Page 4
Task one	Page 5
Task two	Page 12
Task three	Page 13
Task four	Page 15
References	Page 19
Bibliography	Page 20

Introduction

Background theory on logic systems

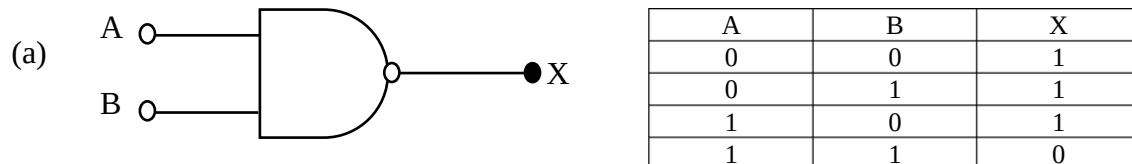
A kind of electronic circuit design for storing and manipulating information is sequential logic. It's different from combinational logic, which is used to create logic gates and circuits that perform basic logic operations such as AND, OR, and NOT. The circuit's output can be determined by the current inputs only in sequential logic, and then also by past states of the circuit. "The answer is that we can't predict the final outcome" – bedford-computing.co.uk, page 6. This means that sequential circuits have memory and can retain information even when the inputs change. Flipflops and registers are two of the most common types of sequential circuits. Furthermore, flipflops are also described as

Not worth referencing "bistables" – Programmable Logic Controllers, Bolton.W. ISBN:9780080961859. Page 61. This is because they've got two different states. Appropriate inputs can switch it from one to the other. The flipflop is an easy two state device to store a single bit of information. It's got two inputs: a clock and a data input. When the flipflop changes its state, the clock input determines the new state, and the data input determines the new state. To describe sequential circuits' behaviour, diagrams or tables which show the different states in which they can be located and how it is shifted from one to another according to its inputs and current state are used. In a wide range of applications, including computer memory, digital signal processing and control systems, sequential logic is used. It is a fundamental concept in digital electronics and it's very important for the design of complex digital circuits. A low-level programming language that computers use to execute instructions is Binary, which also means machine language. the language is called binary. A "binary system is based on just two digits: 0 and 1"- Programmable Logic Controllers, Bolton.W. ISBN:9780080961859. Page 6., which means that there are only two possible digits or states: 0 and 1. The sequence of 0 and 1s, represents every digit with an instruction within. For example, if you insert two digits and store the data in memory or move your records from one register to another, each code corresponds to a

specific operation. The only language that computers can understand directly is binary because it's the language that the computer's processor can execute. However, programming in a binary language is very hard for humans because the codes are complex, and they make mistakes easily.

You have only discussed sequence logic.

Task 1 - Combinational Logic



$$X = A' + B' \quad \checkmark \quad = \overline{A} + \overline{B} = \overline{A \cdot B}$$

In figure one shows the multisim circuit and the truth table this matches the truth table that was filled out

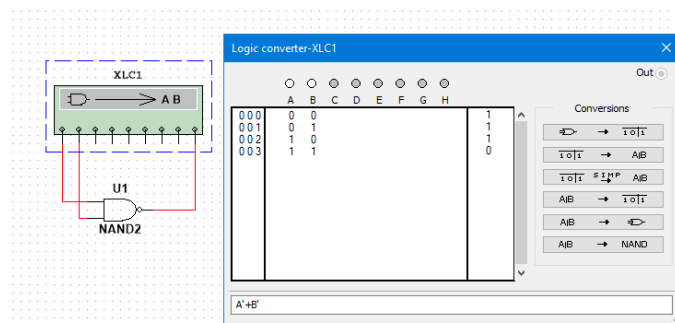
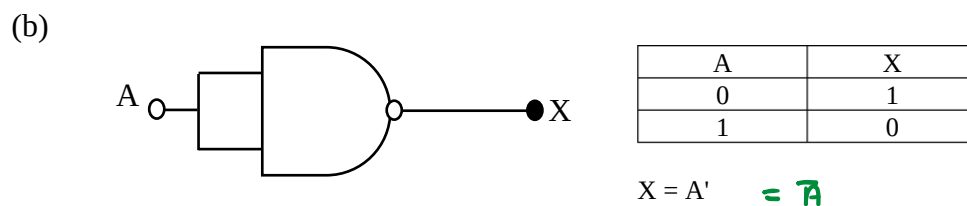


Figure 1



$$X = A' \quad \checkmark \quad = \overline{A}$$

In figure two shows the multisim circuit and the truth table this matches the truth table that was filled out

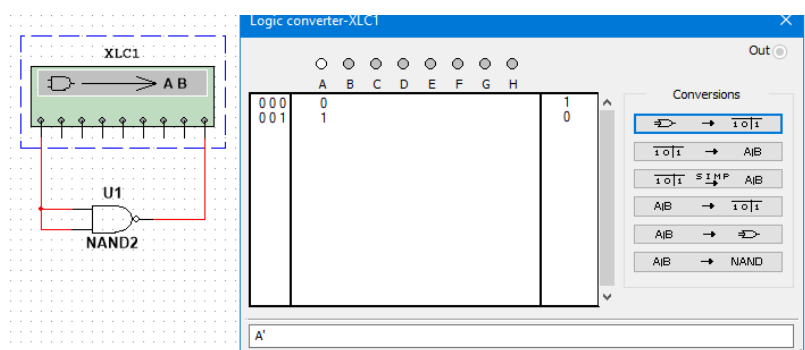
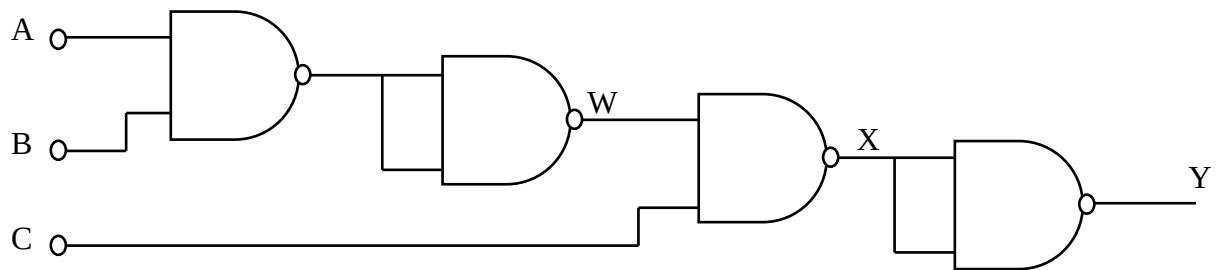


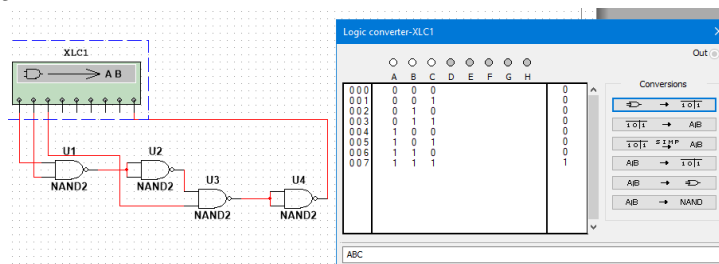
Figure 2

(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

In figure three shows the multisim circuit and the truth table this matches the truth table that was filled out

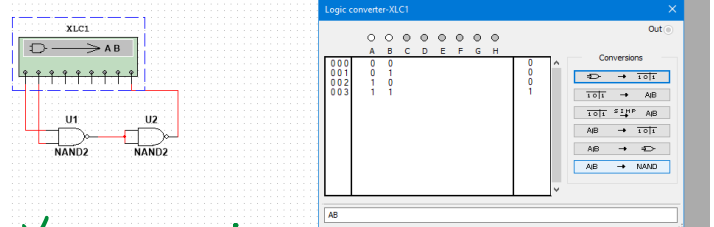


Need to make the image bigger
Figure 3

From the truth table above the logic function of W, X and Y is simplified and stated as .

For W

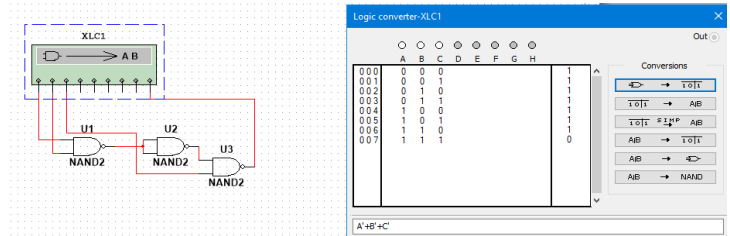
$W = A \cdot B$ 2 input AND gate



Your image is very small.

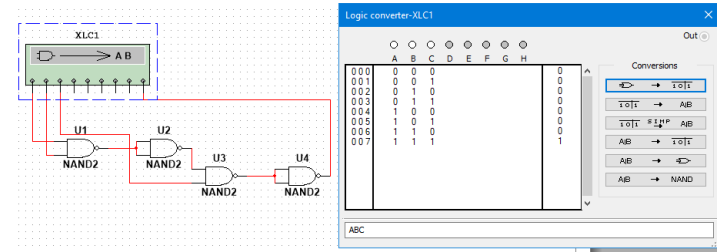
For X
 $X = \overline{A.B.C}$

3 input NAND gate

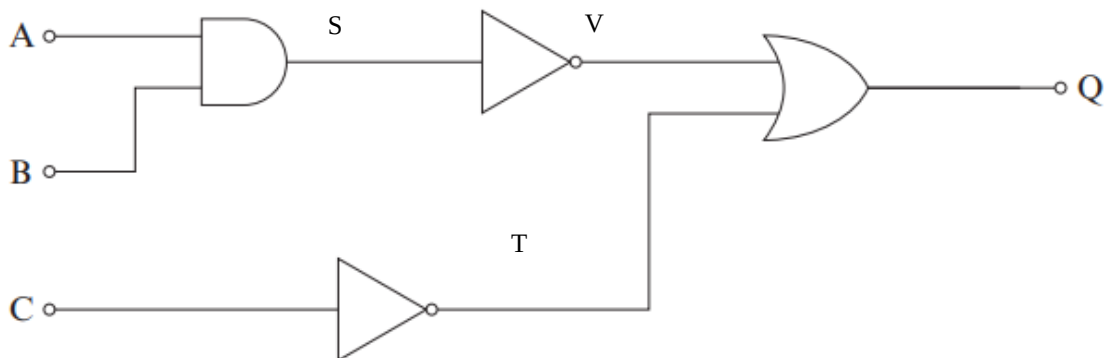


For Y

$Y = A.B.C$ 3 input AND gate



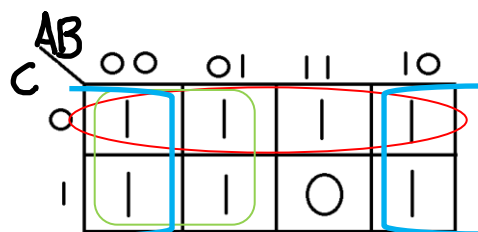
(d)



A	B	C	S	T	V	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

Boolean Expression

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



Looking at the truth table and the Karnaugh map, the Boolean expression can be simplified. By using the Karnaugh map above. We can group the logic ones and simplify the final answer. There are three groups this means there is three parts to the simplified answer. The first group is coloured in red the only value that remains true is $\bar{C}$ this is the first section of the equation. The second section is coloured in green again the only value that doesn't change is $\bar{A}$. The last group is coloured in blue the value that remains is $\bar{B}$. The final equation is $\bar{C} + \bar{A} + \bar{B}$. In figure 4 the first draft of converting the other logic gates by using the NAND gate equivalent. Furthermore, the green lines represent where two NAND gates cancel each other out. In figure 4 this shows the new diagram with all the changes by only using three NAND gates. Furthermore, in figure 4.1 shows the truth table and this matches the truth table above.

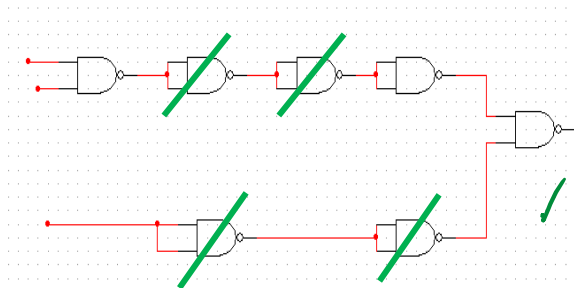


Figure 4

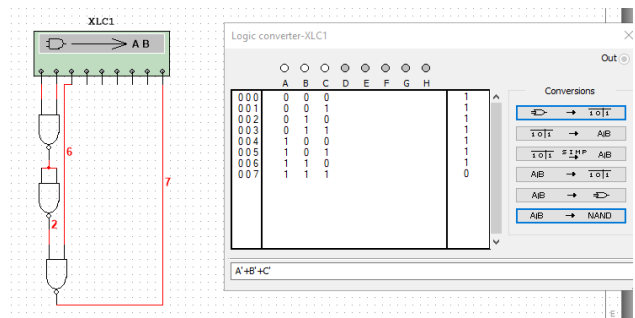


Figure 4.1

There are more steps to aid the simplifications. Reason being is to lower the number of gates used theory this helps to simplify the overall circuit needed to provide the same task with fewer gates needed. This makes the overall costs lower and less chips needed. Using NAND gates instead of using other types of gates has its benefits for example, according to the website analog.com they say that NAND gates are used because they help detect if a signal input is low. Furthermore, it also states that NAND gates are more effective and easier to implement. Also, it is important to simplify the NAND gate as much as possible because the fewer chips used mean less expense.

Part (E)

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

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

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

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

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

$$Q = 0 + AC + (A + \bar{B})(\bar{A} + C)B$$

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

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

$$Q = \bar{A}C + (A + C)BA + 0$$

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

$$Q = AC + BA\bar{A} + BAC$$

$$Q = AC + 0 + BAC$$

$$Q = AC + BAC$$

$$Q = AC$$

Equation 2

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

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

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

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

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

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

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

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

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

Not sure what you are doing?

By using multisim the simplified equations can be produced into a NAND gate circuit figure 5 shows the NAND gate version of equation one. Furthermore figure 5.1 is the circuit diagram of equation two by using NAND gates.

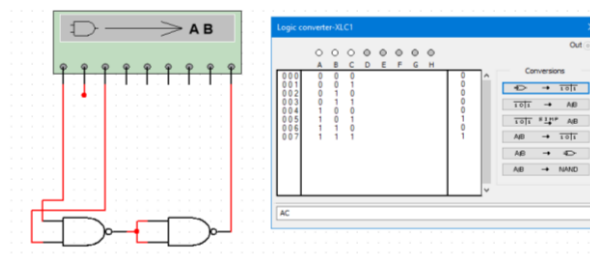


Figure 5

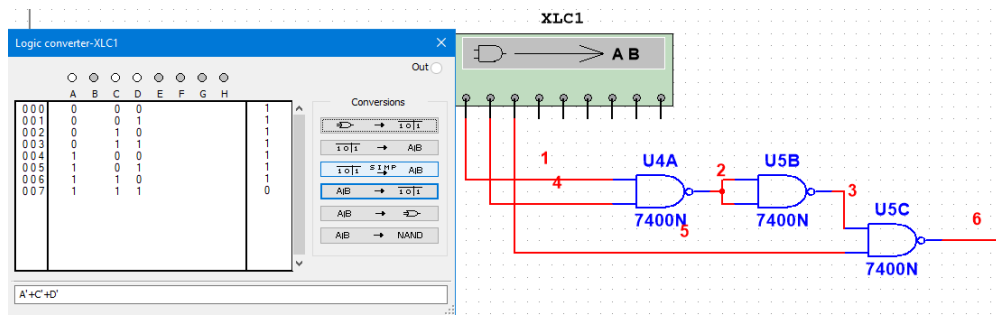


Figure 5.1

The table below shows the truth table of the BCD or Binary Coded Decimal. From this a Karnaugh map has been created from the truth table.

C	B	A	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

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

Looking at the Karnaugh map above. To simplify the final equation the term Grouping is used this involves collecting and evaluating the logic ones to simplify the final answer. From this the final equation is $Q = A.B + A.C$ by using multisim this helps to create a logic gate circuit by only using NAND gates can be seen in figure six this shows the circuit that has been created. ✓

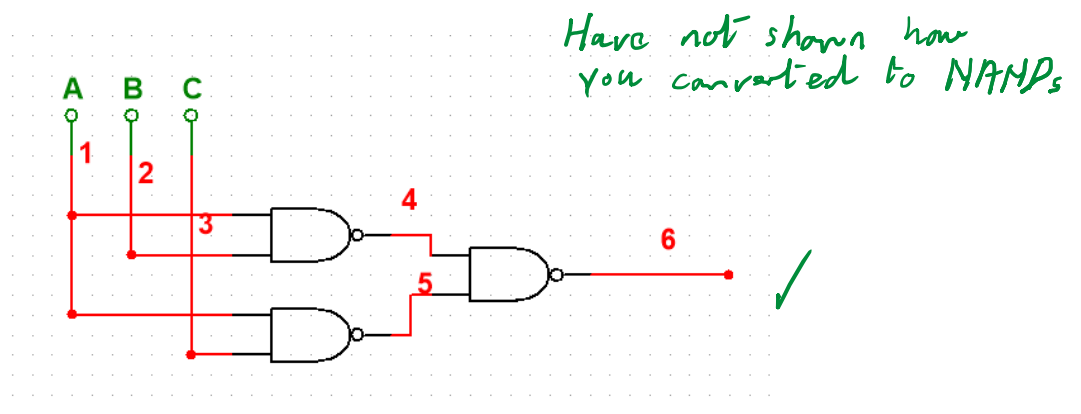


Figure 6

Task 2 - Multiplexer

Input	Selector switch		Output
	A	B	
C0	0	0	C0
C1	0	1	C1
C2	1	0	C2
C3	1	1	C3

Boolean Equation

$$Q = C0.\overline{A}\overline{B} + C1.\overline{A}B + C2.A\overline{B} + C3.AC$$

By using multisim the multiplexer circuit From figure 7 to figure 7.2 shows the multisim circuit and the different outputs showing the different voltages.

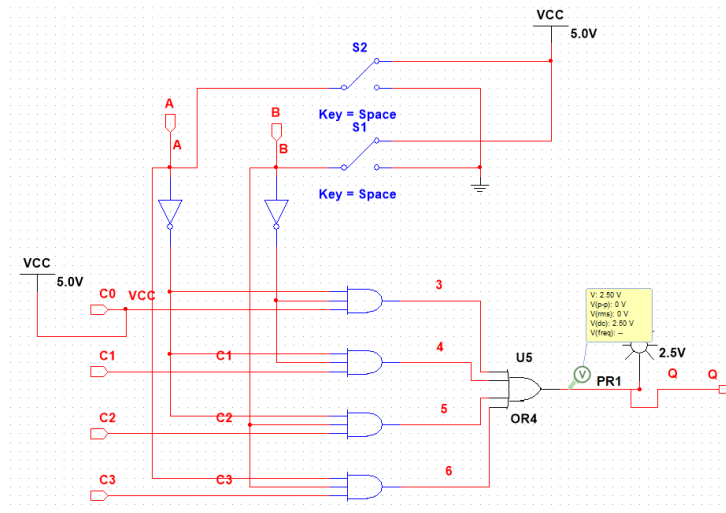


Figure 7

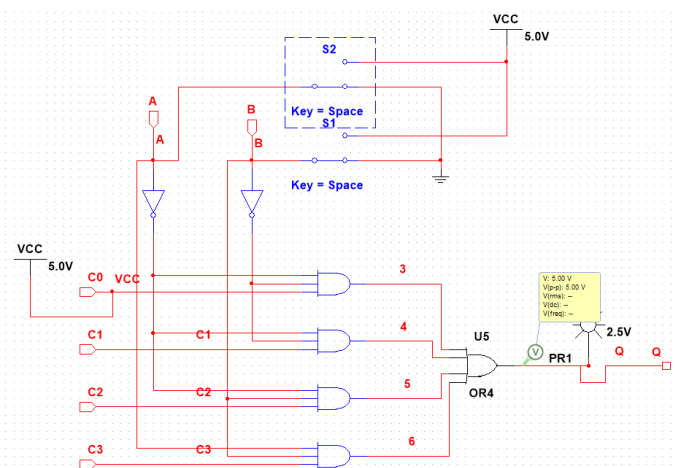


Figure 7.1

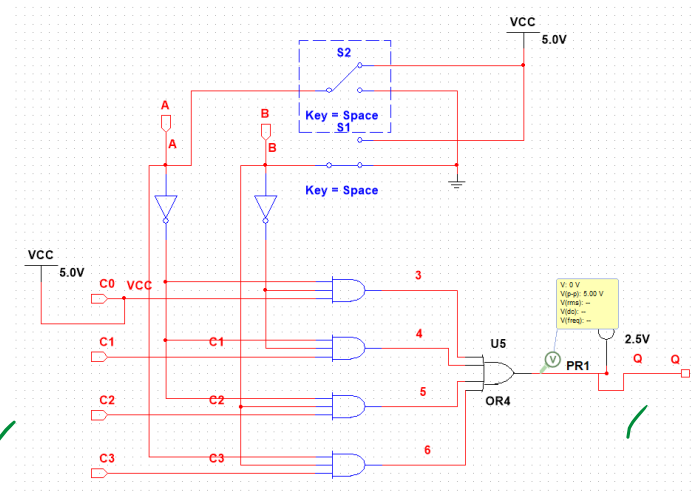


Figure 7.2

Need a statement about testing.

Task 3 - Sequential Logic

(a)

Table 1 - D-type truth table

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

Figures 8 to 8.1 shows the d type flip-flop in both states this is created by using multisim.

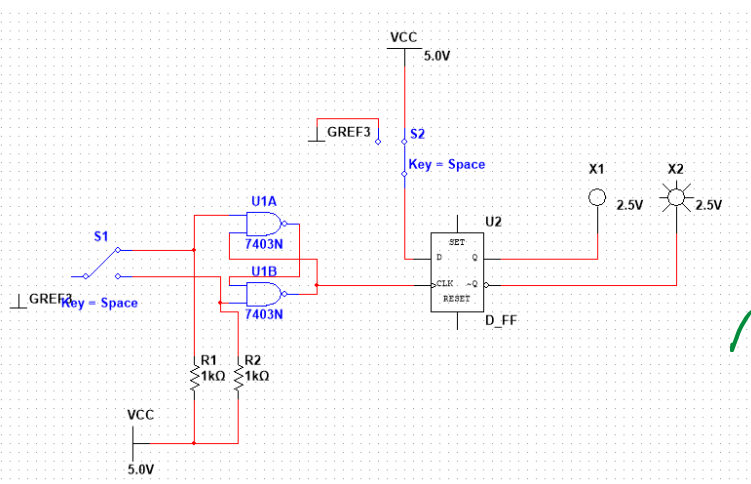


Figure 8

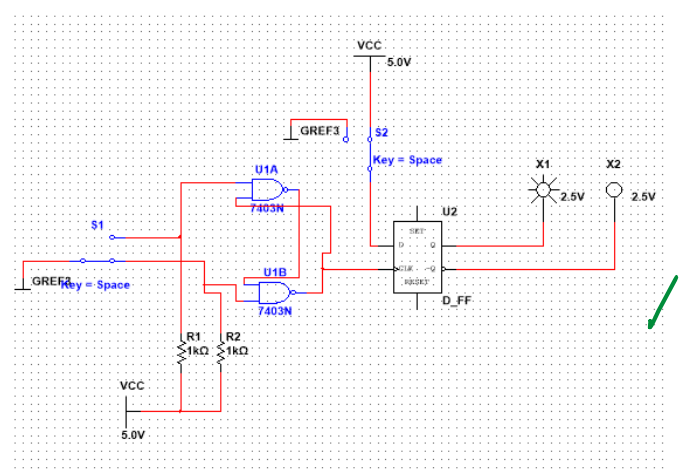


Figure 8.1

(b)

Table 2 – J-K truth table

Input		Outputs	
J	K	Q	$\bar{Q}$
0	0	No change	
0	1	0	1
1	0	1	0
1	1	Toggle	

figure 8.2 to 8.3 shows the JK flip-flops in two different states is shown by using multisim.

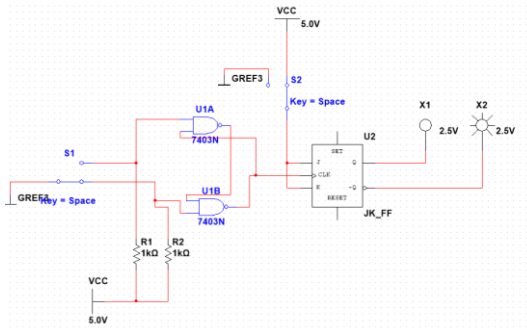


Figure 8.2

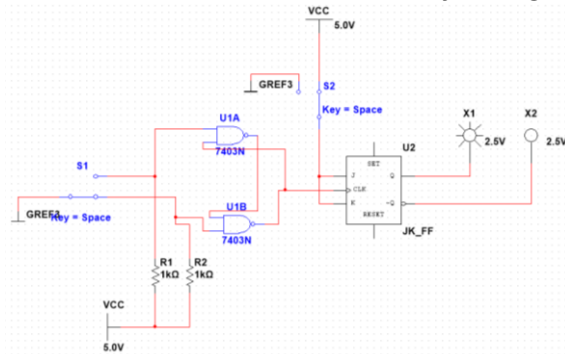


Figure 8.3

(C.)

An electrical signal from the switches and buttons may be loud and unpredictable, which can sometimes cause several changing events like. The term “bounce” is used.” bounce can occur, and the contacts do not make or open cleanly. Because they involve an elastic member, they bounce back and forth like an oscillating spring” – Programmable Logic Controllers, Bolton.W. ISBN:9780080961859page 30 when pressing or releasing a button. A simple switching circuit is seen in figure 9. In addition, a graph that helps to see the fluctuations in the switch speed is displayed next to the circuit. By using the bouncing signal to drive a feedback loop which controls the output of the circuit until the latch is shut down, cross coupled NAND gates work. Two NAND gates are connected in a feedback loop and their output is linked to another NAND gate's input. With the NAND gates as inputs, this is shown in figures 8 to 8.3. When the switch is first pressed, the input of one NAND gate goes high, causing the output of the other NAND gate to go low. This low output is feeding back into the first NAND gate, which is causing the output to increase. This results in a positive feedback loop, which stabilizes the output of the circuit. The graphs of the debounced widow are presented in figure 9.1. It will start to detect the change, and once that state is changed, it will be checked.

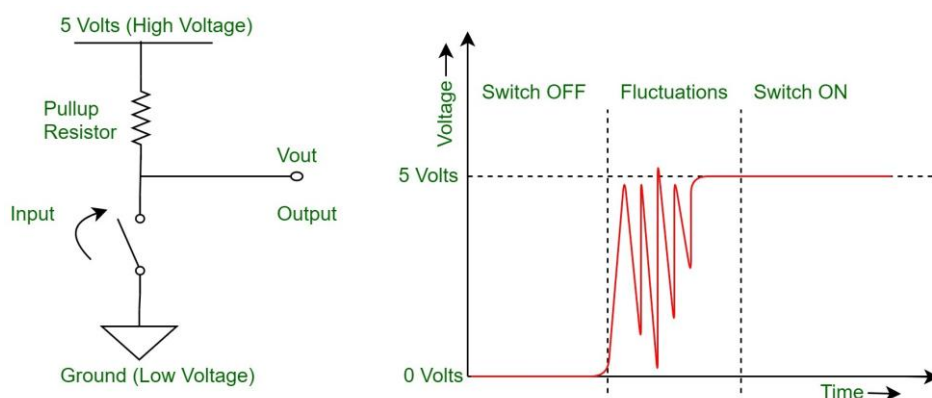


Figure 9

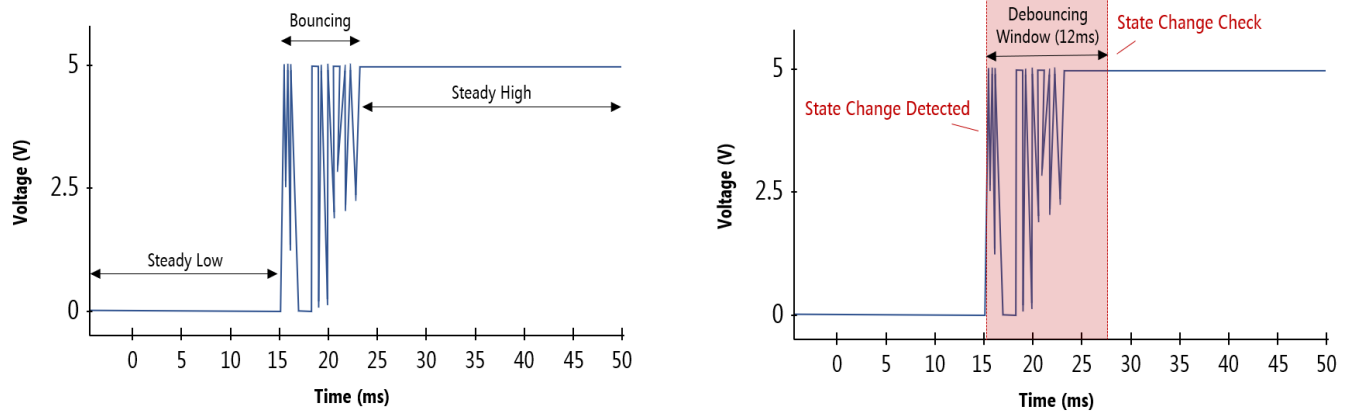


Figure 9.1

Task 4

Using Sequential Logic to Solve Real Problems

Table one shows the truth table of a JK flip-flop.
X = don't care

Table – 1

Present Q	Next Q	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

Table two shows the truth table created for the counter.

Table - 2

	Present			Next								
	Qa	Qb	Qc	Qa	Qb	Qc	Ja	Ka	Jb	Kb	Jc	Kc
0	0	0	0	0	0	1	0	X	0	X	1	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	1	X	1	X	1	1	X

The following Karnaugh maps were created with the use of Table 2. The final circuit could be made easier through the creation of these Karnaugh diagrams. Below you will see all Karnaugh maps which have been prepared for this task.

JK - A

$Q_B Q_C$	00	10	11	01
0	0	X	X	1
1	X	X	X	X

$$J-A = Q_b$$



$Q_B Q_C$	00	10	11	01
0	X	X	X	X
1	0	X	X	1

$$K-A = Q_b$$



$Q_B Q_C$	00	10	11	01
0	1	X	X	X
1	1	X	X	X

$$J-B = 1$$



$Q_B Q_C$	00	10	11	01
0	X	X	X	1
1	X	X	X	1

$$K-B = Q_b$$



$Q_B Q_C$	00	10	11	01
0	0	X	X	0
1	0	X	X	0

$$J-C = 0$$



$Q_B Q_C$	00	10	11	01
0	X	X	X	X
1	X	X	X	X

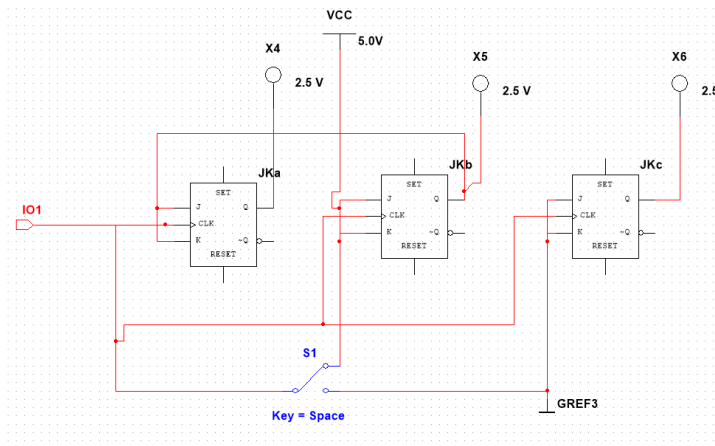
$$K-C = 0$$



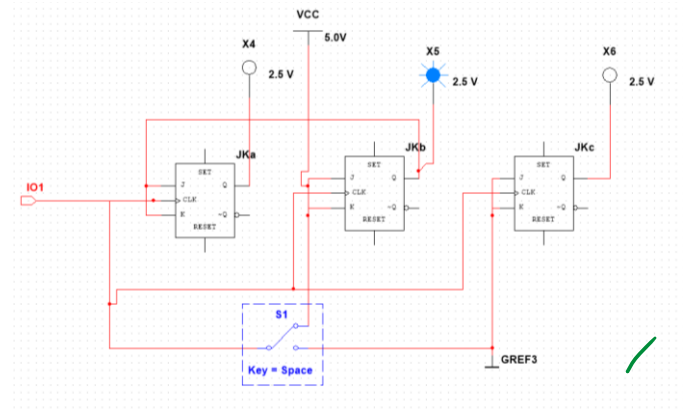
The use of these Karnaugh maps has helped to create a sequence logic counter with JK flipflops. In each of these states, the counter is shown below. The counter is at 000 as shown in Figure 10 and this indicates the counter has not been activated. The system's status changes to 010 once the switch is switched twice. The blue light, which indicates the second number, is shown in Figure 10.1. The state changed to 100 again after a second flicker of the switch and is now number four

in figure 10.2. Finally, the final switch that flickers the state to 110 is now the final state change this number is now saying 2 and 4 with this information, the operator simply adds 4 and 2 equalling 6.

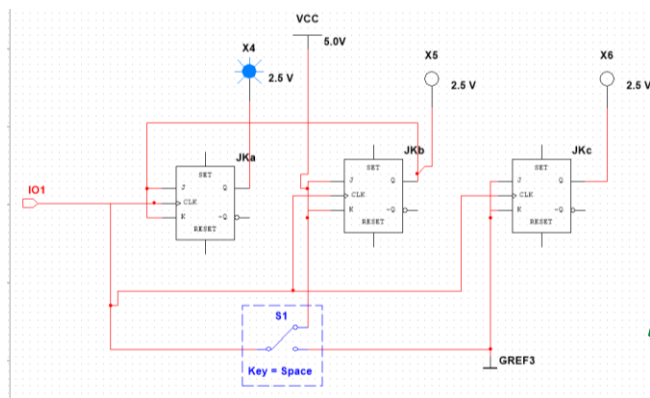
000 - state



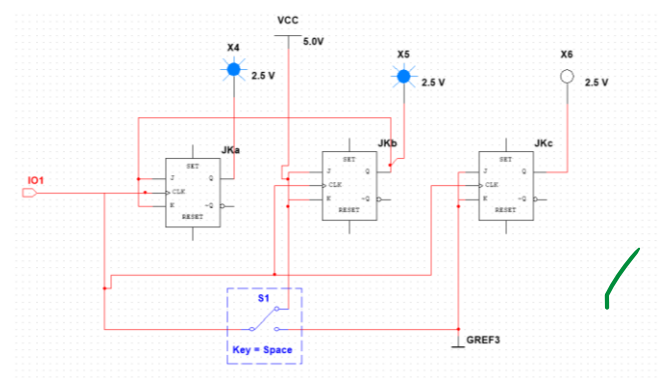
010 - state



100 - state



110 - state



Well done.

Reference

Bolton W. *Programmable Logic Controllers, 6th Edition*. Elsevier Science & Technology. ISBN: 9780128029299 (2015)

analog.com

bedford-computing.co.uk,



This is not how you listed your references.

bibliography

Bolton W. *Programmable Logic Controllers, 6th Edition*. Elsevier Science & Technology. ISBN: 9780128029299 (2015)

Petruszella F D. *Programmable Logic Controllers, 5th Edition*. McGraw Hill Education. ISBN: 9781259254734 (2016)

Kamel K, Kamel E. *Programmable Logic Controllers: Industrial Control*. McGraw Hill Education. ISBN: 9780071810456 (2013)

Green C. *Learn How To Program And Troubleshoot Ladder Logic*. CreateSpace Independent Publishing Platform. ISBN: 9781508474920 (2015)

"Digital Design: Principles and Practices" by John F. Wakerly.

"Fundamentals of Digital Logic with Verilog Design" by Stephen Brown and Zvonko Vranesic.

"Sequential Logic: Analysis and Synthesis" by Joseph Cavanagh.

"Introduction to Digital Systems: Modeling, Synthesis, and Simulation Using VHDL" by Mohammed Ferdjallah.

"Sequential Logic Design" by Les Lawton.

