

assignment 2 5006

by Sean Jones

FILE	10202_SEAN_JONES_ASSIGNMENT_2_5006_152849_1267836435.DOCX		
TIME SUBMITTED	19-APR-2018 12:01PM (UTC+0100)	WORD COUNT	2824
SUBMISSION ID	86861929	CHARACTER COUNT	17 129

Assignment 2

Sean Jones

Finite State Machines

HNC electrical and electronic
engineering

PSDE5006

Submission date: 19/04/2018

Word count: 2994

Contents

1. Introduction, aims and objectives	3
2. Research into advanced digital design techniques	4
3. Design method, choices and calculations	6
4. Testing and improving the design	19
5. Conclusions	22
Reference list and bibliography	23
Bibliography	23

Figure 1 state diagram example	4
Figure 2 kharnaugh map	7
Figure 3 kharnaugh map	8
Figure 4 kharnaugh map	8
Figure 5 controller cct	9
Figure 6 kharnaugh map	9
Figure 7 kharnaugh map	9
Figure 8 output cct	10
Figure 9 controller sub cct	10
Figure 10 kharnaugh map	12
Figure 11 kharnaugh map	12
Figure 12 kharnaugh map	12
Figure 13 5 sec cct	13
Figure 14 cct sub cct	14
Figure 15 5 sec sub cct	14
Figure 16 kharnaugh map	16
Figure 17 15 sec cct	17
Figure 19 complete cct	18
Figure 20 testing	20
Figure 21 fpga board	20
Figure 22 test setup	21

1. Introduction, aims and objectives

This assignment will show the design and testing of a pelican crossing using finite state. It will have a button which when pressed will wait 5 seconds before changing the state of the lights then allowing 15 seconds for pedestrians to cross the road safely. This assignment will need research on similar systems to get the best method. It will also need a final conclusion on how well the design and testing went

2. Research into advanced digital design techniques

(i-programmer, 2018) Says that "The simplest type of computing machine that is worth considering is called a 'finite state machine'." (i-programmer, 2018) also suggests that the finite state machine, changes the current state of something depending on the previous state.

A finite state machine is very simple. it stays on a state until the condition that allows it to move onto the next state is met and so on. Below shows an example of a state diagram. Figure 1

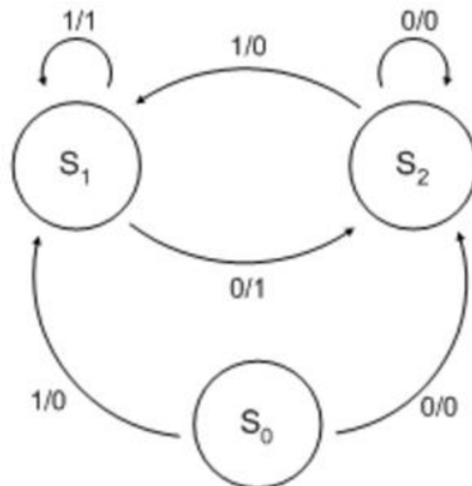


Figure 1 state diagram example

The state diagram is the first stage of implementing a finite state machine. The next stage is to create a truth table from the diagram.

(tutorialspoint, 2018) says that "A Mealy Machine is an FSM whose output depends on the present state as well as the present input." And that "Moore machine is an FSM whose outputs depend on only the present state. "

The Mealy is used in a state where more than one path is desire to give an option of states whereas the Moore machine is used on system where the states tend to loop and follow the same order every time.

The Table below shows a comparison of the mealy and Moore machine given by (tutorialspoint, 2018).

Mealy Machine	Moore Machine
Output depends both upon the present state and the present input	Output depends only upon the present state.
Generally, it has fewer states than	Generally, it has more states than Mealy

Moore Machine.	Machine.
The value of the output function is a function of the transitions and the changes, when the input logic on the present state is done.	The value of the output function is a function of the current state and the changes at the clock edges, whenever state changes occur.
Mealy machines react faster to inputs. They generally react in the same clock cycle.	In Moore machines, more logic is required to decode the outputs resulting in more circuit delays. They generally react one clock cycle later.

The goal of state reduction is to complete the system whilst using as least number of states as possible to cut down on flip flops and complicity. (inst.eecs.berkeley, 2018)“Identify and combine states that have equivalent behaviour”

(inst.eecs.berkeley, 2018) Suggests a equivalent state has all the same input combinations and transitions. It is also the suggested that the best approach to state reduction is the algorithmic approach

(inst.eecs.berkeley, 2018) says this is done by:

- Start with state transition table
- Identify states with same output behaviour
- If such states transition to the same next state, they are equivalent
- Combine into a single new renamed state
- Repeat until no new states are combined

There are 3 main types of flip flops D, JK and T suggests (dummies, 2018)

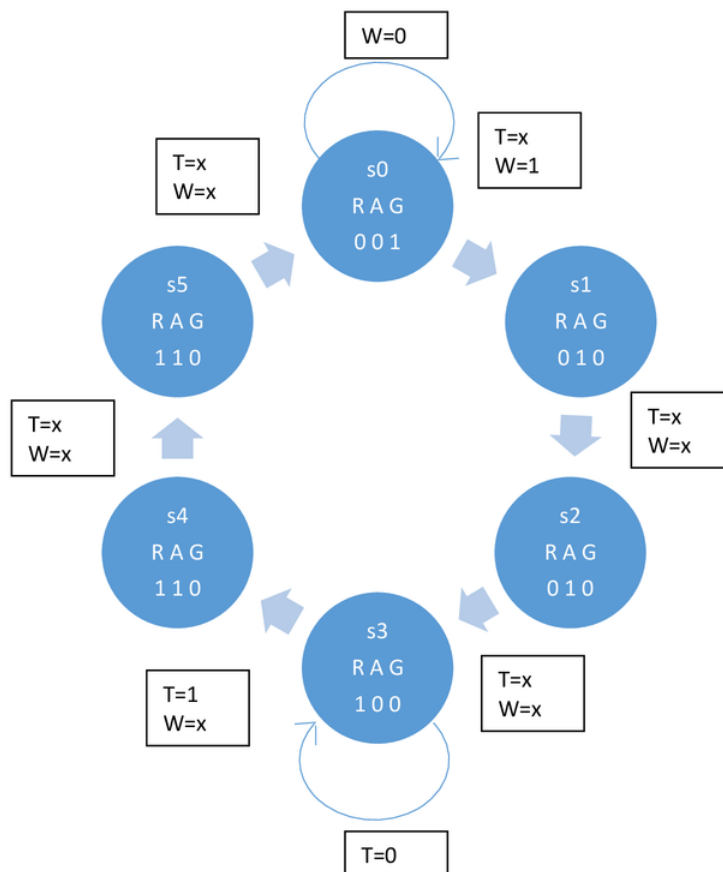
D type: (dummies, 2018) says the D flip-Flop “Has just one input in addition to the CLOCK input. This input is called the DATA input. When the clock is triggered, the Q output is matched to the DATA input “. A D-flip-flop tend to have s and r inputs so the flip flop can be reset ignoring the clock pulse.

(dummies, 2018) says the “JK flip-flop: A common variation of the SR flip-flop. A JK flip-flop has two inputs, labeled J and K. The J input corresponds to the SET input in an SR flip-flop, and the K input corresponds to the RESET input.” The output of a JK will alternate with every pulse if j and k are high.

(dummies, 2018) states “T flip-flop: This is simply a JK flip-flop whose output alternates between HIGH and LOW with each clock pulse. Toggles are widely used in logic circuits because they can be combined to form counting circuits that count the number of clock pulses received. “.a d flip flop can be made into a t flip flop by linking the q and input together.

3. Design method, choices and calculations

When designing a sequential it's good to think about what states the system can be in and how it gets from one state to the other. The best way of visually seeing this is a state diagram. Below shows the state diagram created for the system controller. Where R= Red, A = Amber, G = Green, W = Wait timer complete and T = 15 Sec timer complete. This state diagram also shows the changing of the traffic lights going from green to red and from red to green. On the final system the wait light and walk light will be the inverse of the red and green of the traffic light so if the light is red the walk light will be on and if the light is green the wait light will be on.



Having a state diagram makes it easier to create a truth table. The truth table shows what state all the inputs have to be in to change the state of a certain output and is required to create a Karnaugh map. Below is the controller truth table it shows what outputs the wait and 15 second time effect it also shows the output requirements for the red, amber and green lamps.

	w = 0		w = 1		current			w = 0		w = 1				
Current	t = 0	t = 1	t = 0	t = 1	Qa	Qb	Qc	t = 0	t = 1	t = 0	t = 1	R	A	G
s0	s0	s0	s1	s1	0	0	1	0	0	0	0	0	0	1
s1	s2	s2	s2	s2	0	0	1	0	1	0	0	1	0	0
s2	s3	s3	s3	s3	0	1	0	0	1	1	0	1	1	0
s3	s3	s4	s3	s4	0	1	1	0	1	1	1	0	0	1
s4	s5	s5	s5	s5	1	0	0	1	0	1	1	0	1	0
s5	s0	s0	s0	s0	1	0	1	0	0	0	0	0	0	1

Now the kharnaugh maps can be taken from this truth table. To give a more usable format of the data.

Below shows the kharnaugh maps to create the controller Figure 2

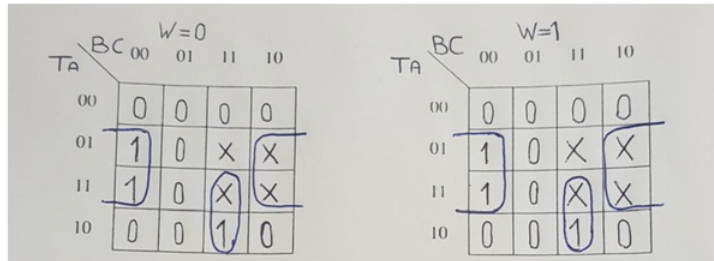
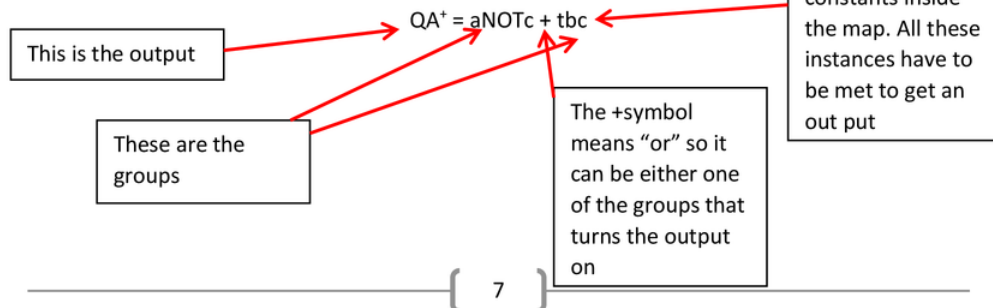


Figure 2 kharnaugh map

The "X" on the map shows a value which can either be 1 or 0 depending on what suits the map best. The idea of the map is to have the least amount of groups and to make the groups as large as possible. The groups can only be the sizes on 2^n otherwise the group will be invalid. The kharnaugh map can only change 1 state at a time so the last to grids are swapped around to allow this. This is important to know when inputting the data into the kharnaugh map. Every "1" must be in a group and groups may only contain "1"s and "X"s, "0" must not be in a group. When the grouping is done then the groups need to be simplified into a Boolean expression. This is a simple process where the group is looked at and where the inputs on the boarder of the map don't change then that letter will be included in the Boolean expression for that group. If the constants for the group is a 1 then it is the letter of that constant that is used in the expression, if it is a 0 then it is classed as "not" that letter. Below shows the Boolean expression for this kharnaugh map.



These are the other kharnaugh maps for this truth table Figure 3 Figure 4

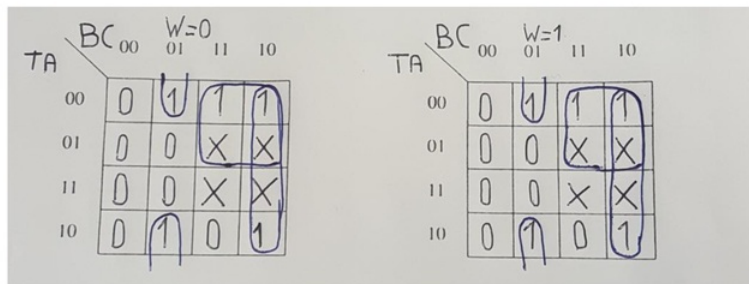


Figure 3 kharnaugh map

$$QB^+ = \text{NOT}a.\text{NOT}b.c + \text{NOT}t.b + b.\text{NOT}c$$

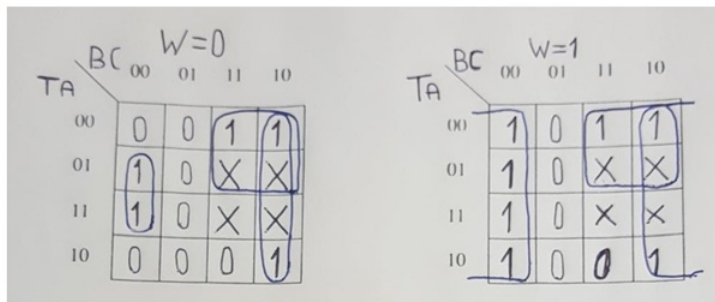


Figure 4 kharnaugh map

$$QC^+ = a.\text{NOT}b.\text{NOT}c + w.\text{NOT}c + b.\text{NOT}c + \text{NOT}t.b$$

When all the Boolean expressions are made they can be converted into logic gate circuit on PLD on multisim. Every output will have its on flip-flop which will connected to the logic gates and a clock every time the clock pulses it will it will change the output state to match the input state coming from the logic gates. Because it is sequential the outputs of the flip-flop will determine the inputs to the logic gates.

Below shows the logic gates connected to the flip-flops for the controller Figure 5

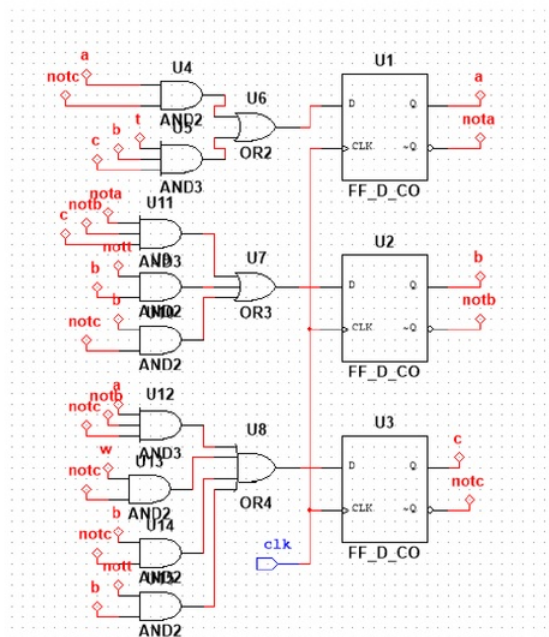


Figure 5 controller cct

Karnaugh maps for the outputs are also made to connect the outputs and controller together
Figure 7 Figure 6

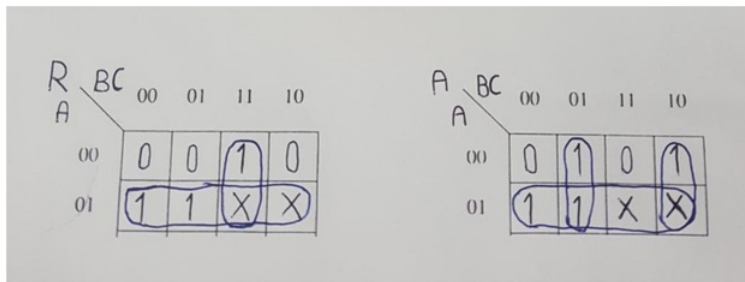


Figure 7 karnaugh map

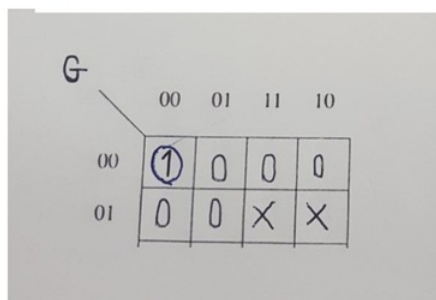


Figure 6karnaugh map

Boolean expressions

$$R = A + B.C$$

$$A = A + \text{NOT}B.C + B.\text{NOT}C$$

$$G = \text{NOT}A.\text{NOT}B.\text{NOT}C$$

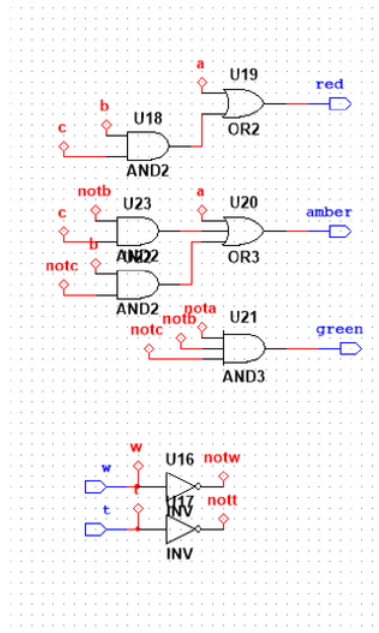


Figure 8 output cct

This converted into logic gates is below Figure 8

These two circuits are then combined and put into a sub circuit so a much simpler final circuit is created when the two timers are added. Another advantage of this is that the variable names don't have to be unique for each of the timers. The sub circuit has connectors that connect the onscreen connectors from inside the sub circuit to allow it to connect to things outside the sub circuit. The final system will have 3 sub circuits which will be connected together with a common clock so that they all work in sync. Below shows what the sub circuit looks like Figure 9

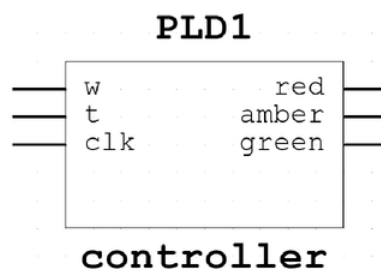
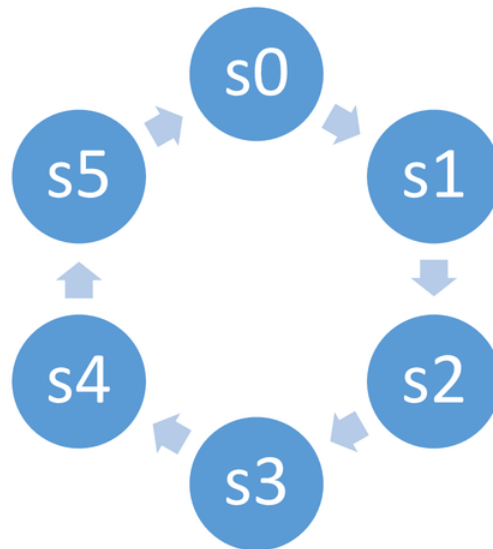


Figure 9 controller sub cct

The next part to be designed is the 5 second timer. This will firstly need a state diagram which is similar to the controller one put a lot simpler as the step number will increase with every second increment instead of other variables. Below is the state diagram.



When the timer has reached 5 it will then go back to 0

The truth table is then made from the state diagram.

		current			next			Output
Current	next	Qa	Qb	Qc	Qa	Qb	Qc	
s0	s1	0	0	0	0	0	0	0
s1	s2	0	0	1	0	0	1	0
s2	s3	0	1	0	0	1	0	0
s3	s4	0	1	1	0	1	1	0
s4	s5	1	0	0	1	0	0	0
s5	s0	1	0	1	1	0	1	1

Below are the kharnaugh maps for the 5 sec counter Figure 10 Figure 11 Figure 12

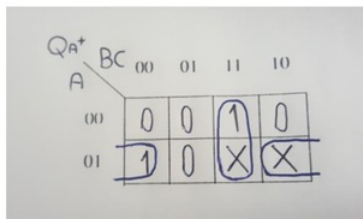


Figure 10 kharnaugh map

$$QA^+ = a.NOTc + bc$$

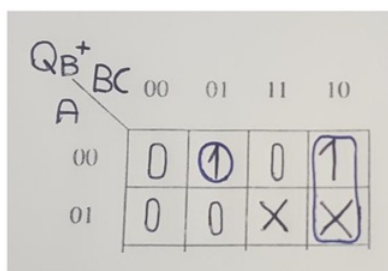


Figure 11 kharnaugh map

$$QB^+ = NOTa.NOTb.c + b.NOTc$$

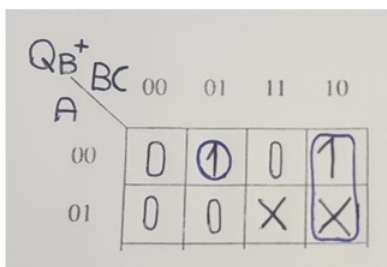


Figure 12 kharnaugh map

$$QC^+ = NOTc$$

Just like the controller all these are converted into logic gates on multisim. This is below the connections on there currently are for testing purposes. The completed circuit will link through the sub circuit and just give a finished count output. Figure 13

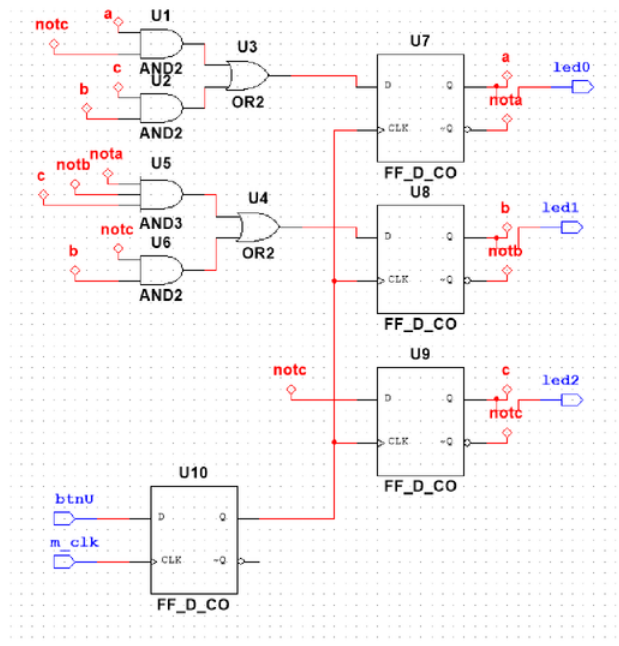


Figure 13 5 sec cct

The circuit then gets compressed into a sub circuit to go into the final circuit Figure 14

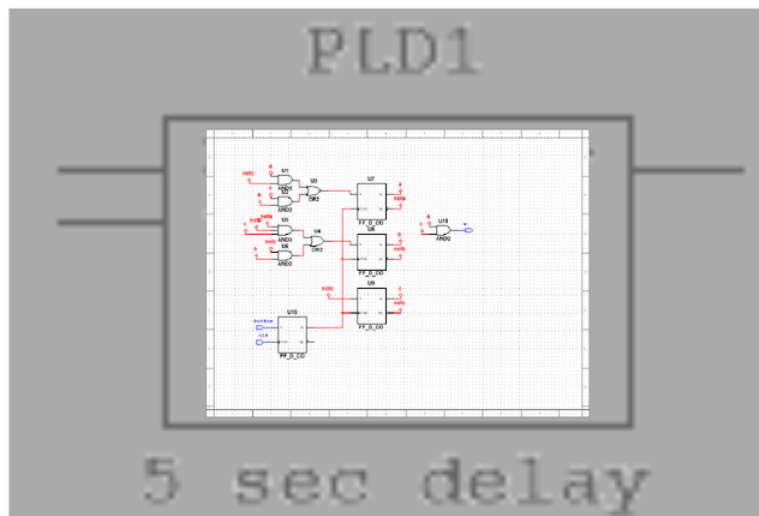


Figure 14 cct sub cct

The sub circuit below shows the input "button" this will trigger the counter to start. "w" is the output that will be high when the count is complete Figure 15

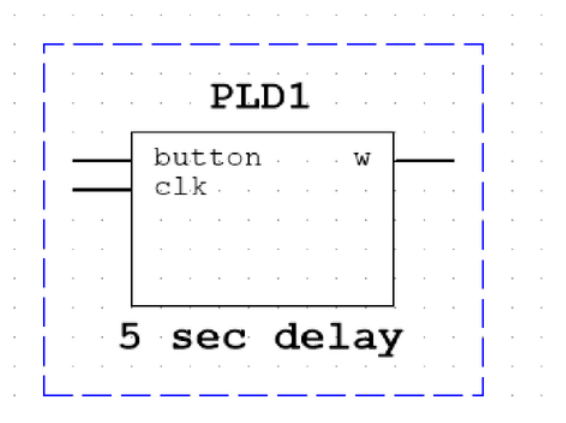
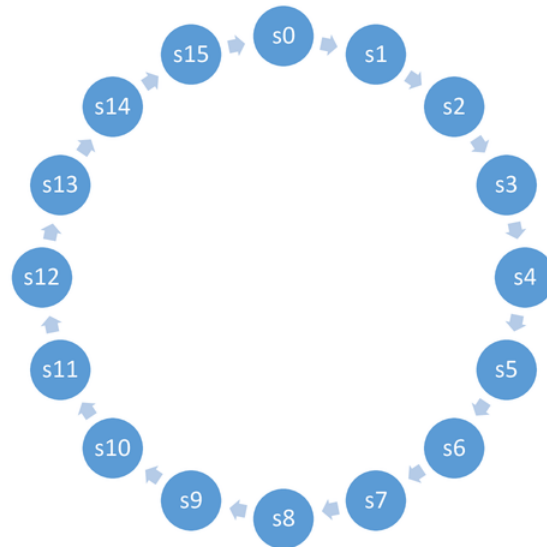


Figure 15 5 sec sub cct

The 15 second timer is the final sub circuit that needs to be made. It is very similar to the 5 second timer but it counts to 15



the truth table can now be made from this

		current				next					
Current	next	Qa	Qb	Qc	Qd	Qa+	Qb+	Qc+	Qd+	Output	
s0	s1	0	0	0	0	0	0	0	1	0	
s1	s2	0	0	0	1	0	0	1	0	0	
s2	s3	0	0	1	0	0	0	1	1	0	
s3	s4	0	0	1	1	0	1	0	0	0	
s4	s5	0	1	0	0	0	1	0	1	0	
s5	s6	0	1	0	1	0	1	1	0	0	
s6	s7	0	1	1	0	0	1	1	1	0	
s7	s8	0	1	1	1	1	0	0	0	0	
s8	s9	1	0	0	0	1	0	0	1	0	
s9	s10	1	0	0	1	1	0	1	0	0	
s10	s11	1	0	1	0	1	0	1	1	0	
s11	s12	1	0	1	1	1	1	0	0	0	
s12	s13	1	1	0	0	1	1	0	1	0	
s13	s14	1	1	0	1	1	1	1	0	0	
s14	s15	1	1	1	0	1	1	1	1	0	
s15	s0	1	1	1	1	0	0	0	0	1	

The kharnaugh maps can now be derived from this truth table Figure 16

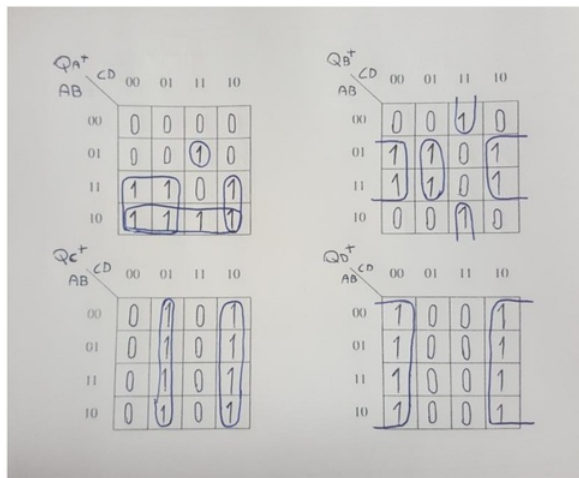


Figure 16 kharnaugh map

The Boolean expressions for these are:

$$Qa+ = a.\text{NOT}c + a.\text{NOT}d + a.\text{NOT}b + \text{NOT}a.b.c.d$$

$$Qb+ = b.\text{NOT}c + b.\text{NOT}d + \text{NOT}b + c.d$$

$$Qc+ = \text{NOT}c.d + c.\text{NOT}d$$

$$Qd+ = \text{NOT}d$$

The Boolean expressions are then converted into logic gates in multisim. Below shows the logic gates Figure 17. The and gate after the flip flops that gives the output T has the binary combination of the number 19 from the flip flops. This will give the count complete output.

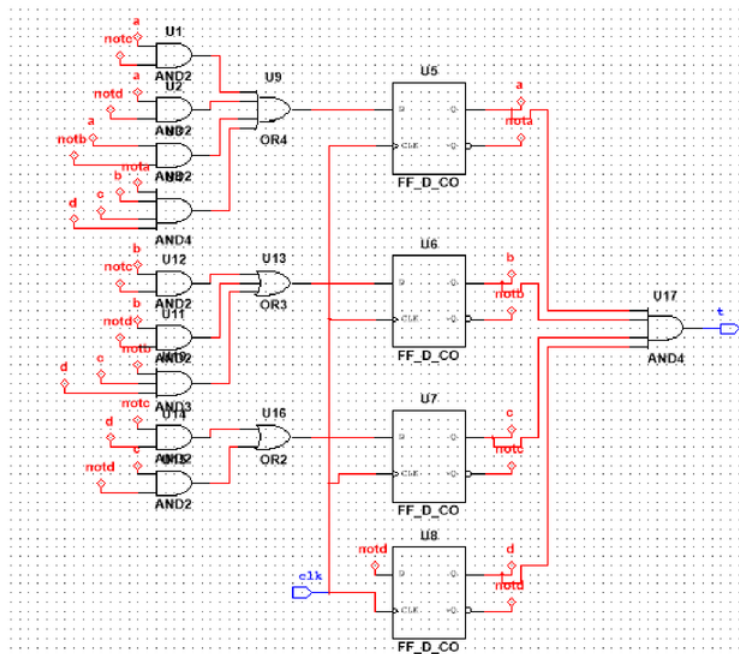


Figure 17 15 sec cct

This is then converted into a sub circuit to be put into the final circuit Figure 17

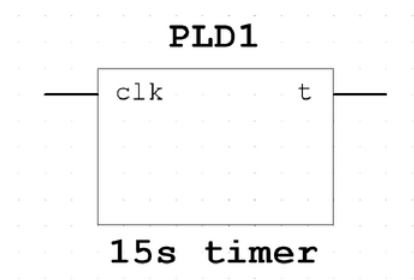


Figure 18 15 sec sub cct

All the sub circuits are put into one circuit where they are linked between each other. They all share the same clock input so that they all work in sync. The 3 led are the traffic lights, the “btnU” is the wait button on the crossing. Led15 is test light to see if the 15 second wait is complete. JA1 is a step frequency input that will work as the clock. Below is the completed circuit Figure 19.

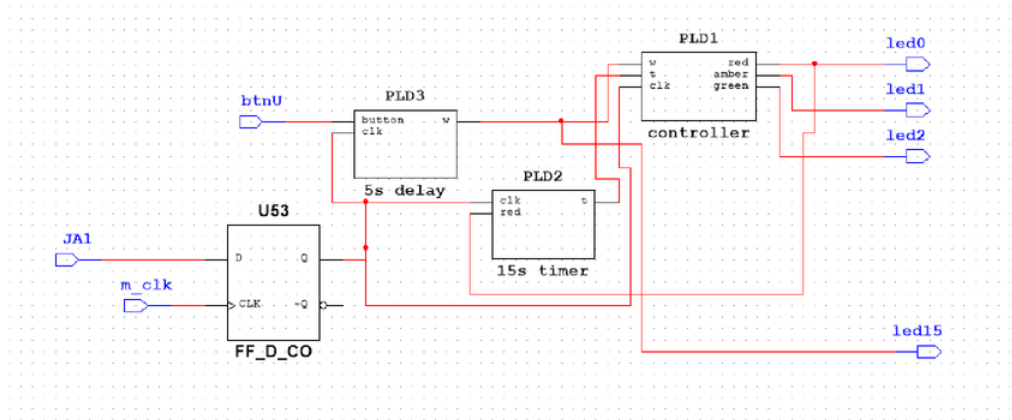


Figure 18 complete cct

4. Testing and improving the design

Once the design and build of the system is complete it is then necessary to test the circuits to ensure they work. The best way to do this is firstly test the individual sub circuits and then test the entire sequence as a whole. The first sub circuit that will be tested is the 5 second timer.

The 5 second timer will be tested by connecting the Digilent Nexys 4 DDR FPGA leds to the circuit and a button to step the system through the sequence. The leds will show the current count of the counter. This is the testing results from the 5 second counter.

number	Led	321	Test results
0		000	000
1		001	001
2		010	010
3		011	011
4		100	100
5		101	101

The 5 second counter counted as it should to 5 seconds and then reset itself back to 0 which was as planned.

15 second counter is the next thing to be tested. This will be tested the same as the 5 second counter but there will be 4 led outputs as it's a 4 bit output. This will need to count from 0 to 15 and reset itself back to 0.

number	Led	4321	Test results
0		0000	0000
1		0001	0001
2		0010	0010
3		0011	0011
4		0100	0100
5		0101	0101
6		0110	0110
7		0111	0111
8		1000	1000
9		1001	1001
10		1010	1010
11		1011	1011
12		1100	1100
13		1101	1101
14		1110	1110
15		1111	1111

The test table shows that the counter performed exactly as it should do. The advantage of a testing table is that differences between expected and what happened can clearly be spotted.

The controller sub circuit will have to be tested using a logic analyser to determine whether it is changing states correctly or not. The logic analyser shows the clock input against the out puts of the 3 led. The states show that the colours are of a traffic light sequence. The advantage of a logic analyser is that the timing scale is visible also so if there was a fault it would be easier to pin point it. Figure 20

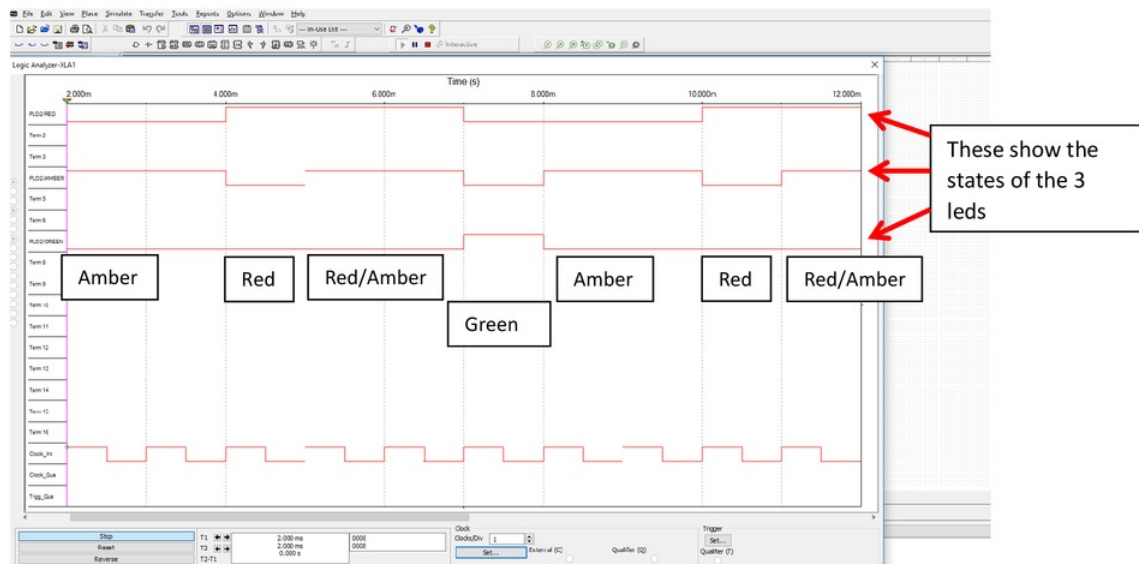


Figure 19 testing

The complete final circuit is then tested to see if the system functions correctly when built all in one. This will be done by transferring the final circuit to the fpga board. There will need to be a function generator attached to the pin JA1. This will need to have a square step wave form with a frequency of 1Hz so that the clock pulse will be 1 every second. This is what the function generator connected to the board looks like Figure 21 Figure 22.

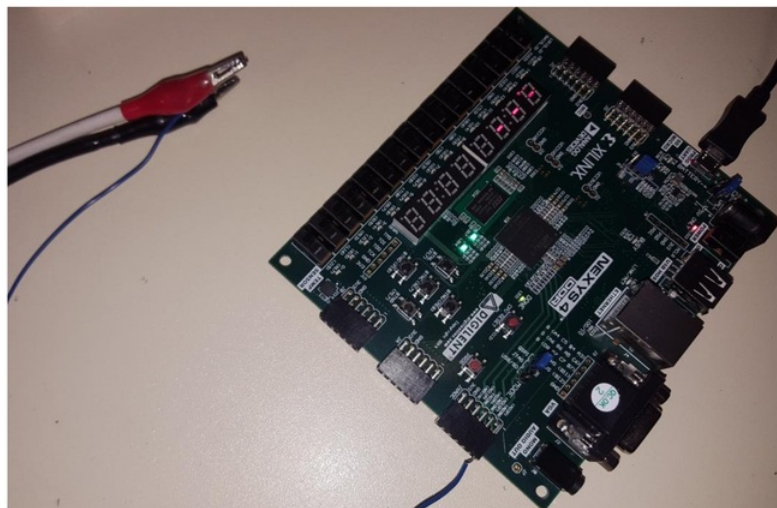


Figure 20 fpga board

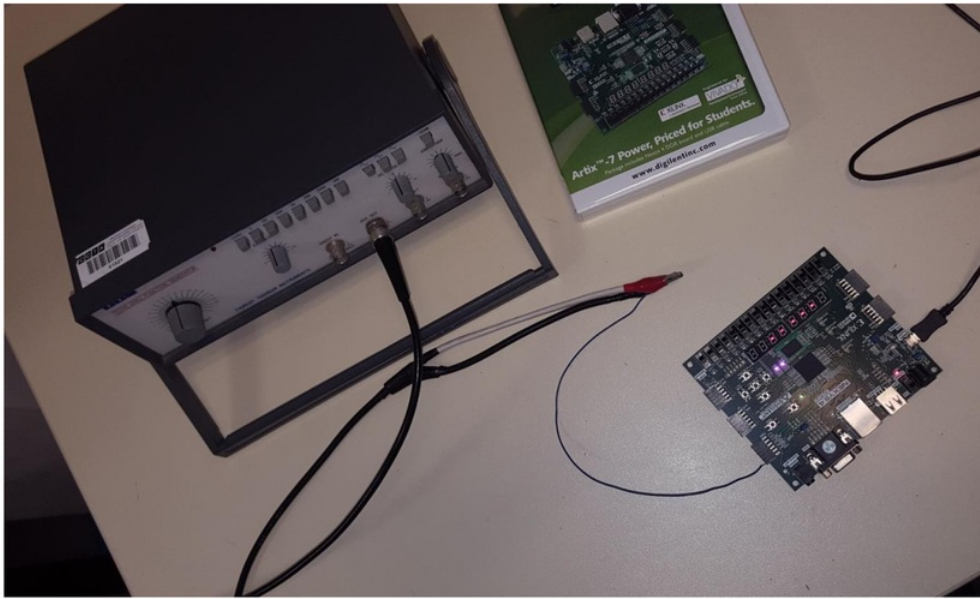


Figure 21 test setup

A test table was produced to show the testing of the system.

Test	What should happen?	What happend
run	Green led illuminated	Green led illuminated
Push wait button	5 sec delay	5 sec delay
Do nothing	Lights change from green to red	Lights changed from green to red
Do nothing	15 sec delay	15 sec delay
Do nothing	Lights change from red to green	Lights changed from red to green

The test table shows that the system run as it was expected and all the time delays were fairly accurate.

5. Conclusions

The project went very well to design and test a pelican crossing. The design was very good and creating a strong initial design improved the time of testing and implementing of the circuit. The research gave a head state on the state designing and boosted knowledge in this subject.

Creating all the truth tables were very time consuming as the controller had a very complex one with many variables to consider.

The testing could have been improved to use the logic analyser on all the circuits but it was only until the last sub circuit that it was made aware that this feature was available. Time would have been saved on testing if this feature was recognised sooner.

The crossing could have been improved if it was a puffin crossing instead of a pelican crossing. This would prevent people coming up to the crossing, pressing wait and walking off causing the lights to change even though nobody is crossing. The puffin crossing would have sensor by the wait button to see if the person is still standing there. This could be implemented by just having an extra input on the wait sub circuit causing the count to rest to 0 if the person present sensor is not healthy.

Reference list and bibliography

Bibliography

- dummies. (2018, 04 19). *www.dummies.com*. Retrieved from *www.dummies.com*:
<http://www.dummies.com/programming/electronics/diy-projects/digital-electronics-types-of-flip-flop-circuits/>
- inst.eecs.berkeley. (2018, 04 19). <http://www-inst.eecs.berkeley.edu/~cs150/sp00/classnotes/katz-ch9-mod.pdf>. Retrieved from *www-inst.eecs.berkeley.edu*: <http://www-inst.eecs.berkeley.edu/~cs150/sp00/classnotes/katz-ch9-mod.pdf>
- i-programmer. (2018, 4 19). *i-programmer*. Retrieved from *www.i-programmer.info*: <http://www.i-programmer.info/babbages-bag/223-finite-state-machines.html>
- tutorialspoint. (2018, 4 19).
https://www.tutorialspoint.com/automata_theory/moore_and_mealy_machines.htm.
Retrieved from *www.tutorialspoint.com*:
https://www.tutorialspoint.com/automata_theory/moore_and_mealy_machines.htm

assignment 2 5006

GRADEMARK REPORT

FINAL GRADE

GENERAL COMMENTS

71 /100

Instructor

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8

PAGE 9

PAGE 10

PAGE 11

PAGE 12

PAGE 13

PAGE 14

PAGE 15

PAGE 16

PAGE 17

PAGE 18

PAGE 19

PAGE 20

PAGE 21

GRADING FORM: CSG-RUBIC

SEAN JONES**71****RESEARCH-INFORMED LITERATURE**

There was evidence of extensive further reading and research from credible and reliable sources. Your interpretation and understanding of the additional research was very good and demonstrated a deep underlying appreciation of the subject. Your references were appropriate to the task and your reference list was of an appropriate length for this assignment.

8**KNOWLEDGE AND UNDERSTANDING**

The knowledge and understanding of the subject demonstrated by this assignment was generally very good. There were very few gaps in your knowledge of the basic principles and your understanding of the higher level concepts was very good. Your research and explanation was particularly good and extended your knowledge well beyond that contained in the course notes.

25**ANALYSIS**



Your explanation of the design process was good, however the conclusions lacked some detail. You could have evaluated the use of the various components, specified whether the solution met the specification and suggested ways to improve the solution. You have however demonstrated good analytical skills and your approach to the task was logical. Further work – in your conclusion, you should relate the results to the original specification and to accepted theory.

12

PRACTICAL APPLICATION AND DEP.



Your application of practical skills for this task were generally good, you used the resources appropriately when carrying out the experiment and you used appropriate options for testing and evaluating the circuit performance - albeit to a limited fashion. Testing generally lacked a lot of the required detail however and you should have included a comprehensive set of results for each stage.

21

SKILLS FOR PROFESSIONAL PRAC.



Your written report was generally good and contained most of the required detail, although it lacked the depth of explanation required. Avoid the use of photographs and/or scans of hand-written notes.

5