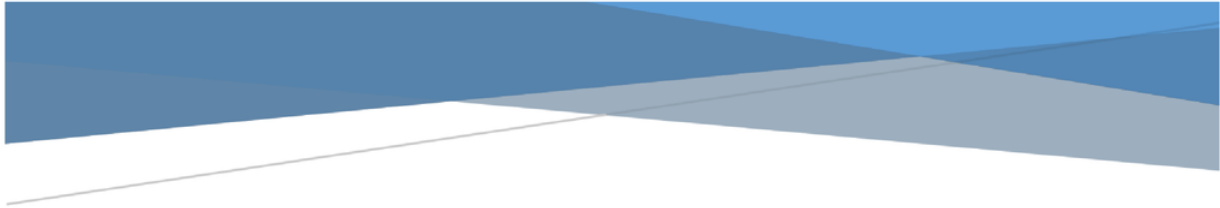


Assignment 2

by Declan Davies

FILE	11279_DECLAN_DAVIES_ASSIGNMENT_2_152849_1592585318.DOCX		
TIME SUBMITTED	24-APR-2018 12:40AM (UTC+0100)	WORD COUNT	3978
SUBMISSION ID	87275495	CHARACTER COUNT	17968



ASSIGNMENT 2 – FINITE STATE MACHINES

PSDE 5006 Electronics II

Declan Davies
[Email address]

Contents

1. Introduction	2
2. Research into Advanced Digital Design Techniques	3
2.1 Implementation of Finite State Machines	3
2.2 Comparison of Moore and Mealy Machines	4
2.3 State Reduction Techniques	7
2.4 Use Of Different Flip Flops	8
3. Design Method, Choices and Calculations	9
3.1 Design 5 Second Delay	9
3.2 15 Second Delay	11
3.3 Main Controller	14
4. Testing and Improving the Design	17
4.1 5 Second Delay	17
4.2 15 Second Delay	19
4.3 System Built In Multi-Sim	19
5. Conclusions	21
6. References	22

1. Introduction

As part of this project the aim is to design a puffin crossing system that will require using the Diligent Nexys 4 DDR FPGA board. As a result it will require preliminary research to be carried out looking into the different ways of implementing the finite state machine, which will mean as a result looking into the positive and negatives of each method. Following deciding upon the best implementation method for the finite state machine additional research needs to be carried out by first looking into a comparison of Moore and Mealy machines along with a comparison between state reduction techniques and the different types of flip flops that could be used for the design of the state machine.

After the preliminary research for the project has been completed the next objective will be to begin the design process that will begin with creating a truth table for each of the different states in the pelican crossing this will need to take into account a delay between the traffic lights changing once the 'Wait' button has been pressed by a pedestrian.

The next objective after successfully creating a truth table will be to begin the design of the karnaugh maps, which is vital as without the completion of the karnaugh maps the circuit cannot be designed within multi-sim. Once the circuit has been designed it will require each of the different circuits to be build in multi-sim and then tested through the use of a logic analyser along with running it on the nexys board.

2. Research into Advanced Digital Design Techniques

In this section it will look into the preliminary research for the following subjects that are relevant to the project as can be seen below:-

- Implementation of Finite State Machines
- Comparison of Moore And Mealy Machines
- State reduction techniques
- Use Of Different Flip Flops

2.1 Implementation of Finite State Machines

According to "Fernando Bevilacqua" a finite state machine is **"a model of computation based on a hypothetical machine made of one or more states."** (2013). Also a finite state machine can only have one active state at any time, which means as a result in order for the machine to perform different actions it must transition from one state to another. A finite state machine is made up of states, events, transitions and actions. The transitions contain a source and a target state, which will be performed once the state machine is in the source state that results in the event associated with the transition occurring.

According to (Gurp, 1999) in **"procedural languages, FSMs are usually implemented using case statements"**. According to (James, 2018) the **"finite state machine is also a useful approach to many problems in software architecture"**. Even though the transition from one state to another depends on the old state and the input it does not produce the same result or behaviour and the reason for this is down to a change of state. This says that the entire history of the machine is summarized in its current state and that is the most important thing not how it reached that particular state.

A way of representing a finite state machine is to draw a circle for each state and an arrow that will show what state follows each of the inputs. As can be seen in figure below:-

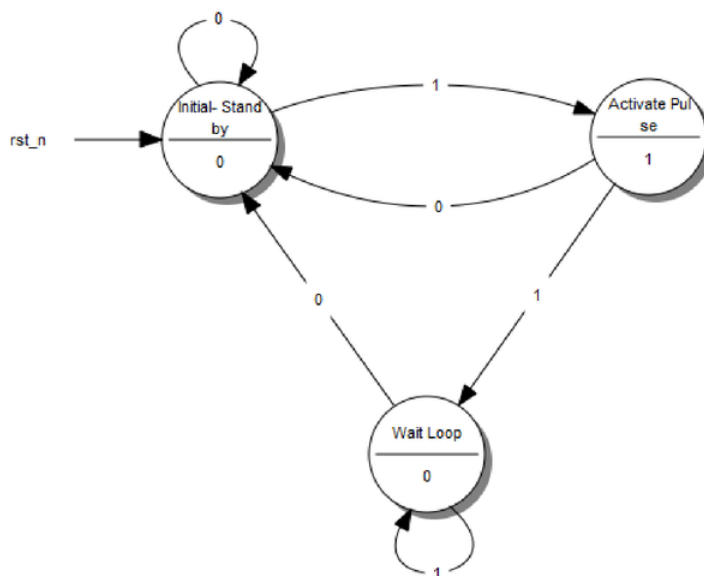


Figure 1 - Three Way Finite State Machine

In figure 1 it displays the picture provided by (Circuits) it shows that when in the first circle called initial-Stand by if the output was 1 it would change state to the second circle called "Activate Pulse" and if this outputs a value of 1 it would move onto the "Wait Loop" but if 0 it would move onto the "Initial Stand By". In the case that the value is 1 and it reaches the wait loop it will run the loop once before outputting value of 1 and moving back to the "Initial – Stand By". Following creating the diagram the next part of the implementation process is to create a "State Table" that can be seen below:-

Current State		Input I	Next State		Outputs Y
A	B		A _{next}	B _{next}	
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	1
0	1	1	1	0	1
1	0	0	0	0	0
1	0	1	1	0	0
1	1	0	X	X	X
1	1	1	X	X	X

Figure 2 - Truth Table

In figure 2 it shows the current state and the input, which will have an impact on the next state and the outputs.

2.2 Comparison of Moore and Mealy Machines

What is a Mealy Machine?

According to (Point, 2018) a "**Mealy Machine is an FSM whose output depends on the present state as well as the present input.**". The way that this can be described is with the following six things in the list below:-

- Q that represents the finite amount of set states.
- Σ that represents the finite amount of set symbols known as the input alphabet.
- O that represents the finite amount of set symbols known as the output alphabet.
- Δ that represents the transition input function for $\delta: Q \times \Sigma \rightarrow Q$
- X that represents the transition output function for $X: Q \times \Sigma \rightarrow O$
- q_0 that represents the initial state for any of the inputs that have been processed ($q_0 \in Q$).

An example of a mealy state table can be seen in figure 2 below:-

Present state	Next state			
	input = 0		input = 1	
	State	Output	State	Output
→ a	b	x ₁	c	x ₁
b	b	x ₂	d	x ₃
c	d	x ₃	c	x ₁
d	d	x ₃	d	x ₂

Figure 3 - Mealy Machine State Table

Once the state machine has been drawn up it next requires designing a state diagram that can be seen in figure 4 below:-

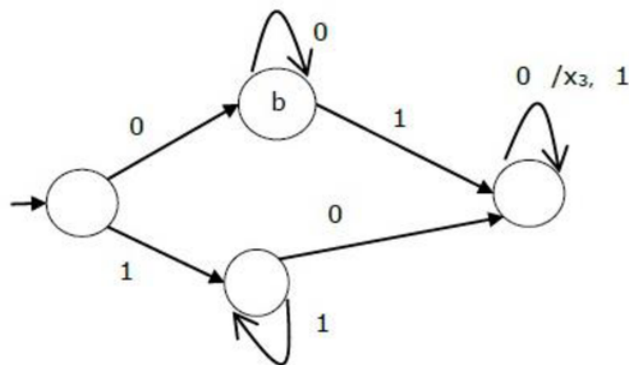


Figure 4 - Mealy Machine State Table

What is a Moore Machine?

According to (Point, 2018) an “FSM whose outputs depend on only the present state” that can be described by the following 6 things that can be seen below:-

- Q that represents the finite amount of set states.
- Σ that represents the finite amount of set symbols known as the input alphabet.
- O that represents the finite amount of set symbols known as the output alphabet.
- Δ that represents the transition input function for $\delta: Q \times \Sigma \rightarrow Q$
- X that represents the transition output function for $X: Q \times \Sigma \rightarrow O$
- q_0 that represents the initial state for any of the inputs that have been processed ($q_0 \in Q$).

As can be seen above are the same as the ones for the Mealy machine but even though this is the case the state table will have different outputs for the Moore machine as can be seen in figure 5 below:-

Present state	Next State		Output
	Input = 0	Input = 1	
→ a	b	c	x ₂
b	b	d	x ₁
c	c	d	x ₂
d	d	d	x ₃

Figure 5 - Moore Machine State Table

In figure 5 it shows an example of a state table associated with a Moore machine and the state diagram associated with the state table above

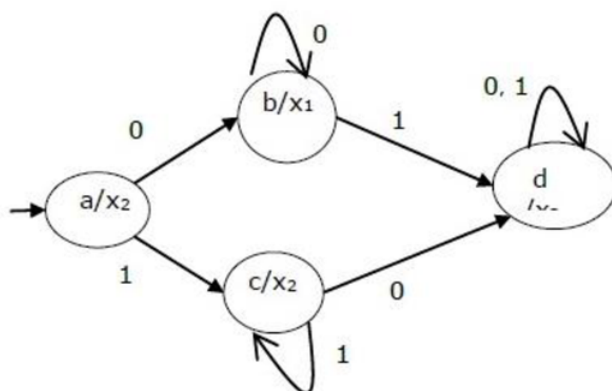


Figure 6 - State Diagram Moore Machine

Comparison Between Mealy Machine vs. Moore Machine

Mealy Machine	Moore Machine
The output for mealy machine will depend on the current state along with the current input	For a moore machine the output is reliant on the current state.
The mealy machine normally has fewer states than a moore machine.	Usually a Moore machine has more states than a mealy machine.
The function of the transition's and the changes and a function of this will be the value of the output function	The function of the current state along with the changes to the clock edges, when the state changes occur and a function of this is the value of the output function.
The mealy machine tends to have a faster reaction to any of the inputs, which generally have a reaction in the clock cycle.	A moore machine requires the use of more logic in order to decode any of the outputs that can result in additional circuit delays and unlike the mealy machine the moore machine ill usually react one clock cycle later.

2.3 State Reduction Techniques

According to (Ehab, 2018) the “Reduction of the number of flip-flops in a sequential circuit is referred to as the state reduction problem. State-reduction algorithms are concerned with procedures for reducing the number of states in a state table, while keeping the external input-output requirements unchanged.”.

The reason that using state reduction techniques is so important is due to the facts that it results in using the least amount of flip flops possible and this will reduce the amount of gates that are needed when it comes to implementation. The goals of state reduction is to first identify and combine the states that have the same behaviour. The equivalent states for all the input combinations, state transition to the same or equivalent states An example of this would be if the equivalent states S0 and S2 both have an output of 0 and they both transition to S1 on a 1 and self-loop on a 0.

According to (Banerjee, 1998) an example of a row matching method can be seen below:-

Example FSM Specification:

Single input X, output Z

Taking inputs grouped four at a time, output 1 if last four inputs were the string 1010 or 0110

Example I/O Behavior:

X = 0010 0110 1100 1010 0011 ...
Z = 0000 0001 0000 0001 0000 ...

Upper bound on FSM complexity:

Fifteen states (1 + 2 + 4 + 8)

Thirty transitions (2 + 4 + 8 + 16)

sufficient to recognize any binary string of length four!

Figure 7 - Row Matching Method

Even though using the row matching technique is a straightforward and easy to implement method of state reduction. The problem that comes with this method is that it does not result in the most reduced state table possible. Another method of state reduction can be performed through the use of implicant charts.

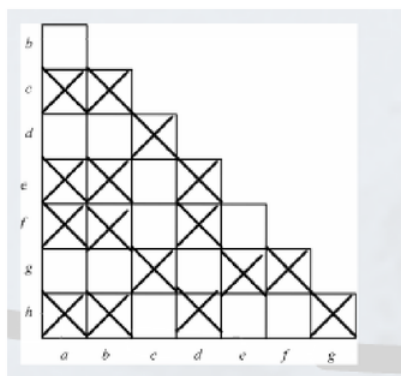


Figure 8 - Implicant Charts 1

In figure 8 it shows the first step of using implicant charts, which is to use a X in the squares to eliminate any outputs that contain incompatible states.

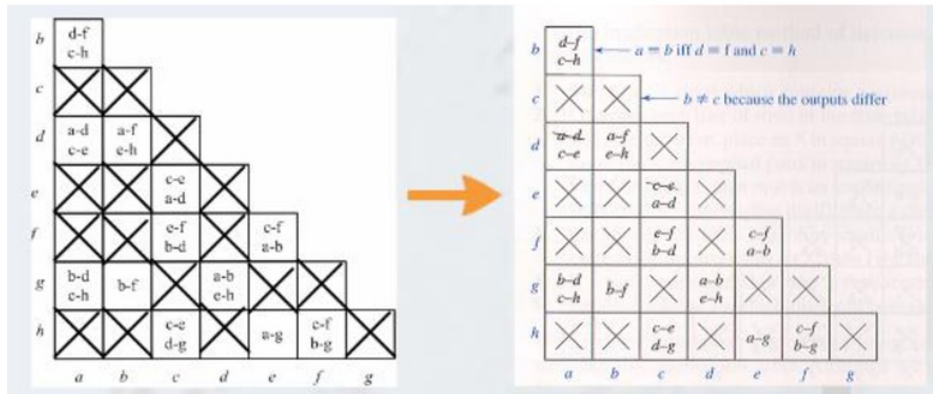


Figure 9 - Implicant Charts Step 2

In figure 9 it shows an example of what will happen in step 2 when looking to remove any self-redundant pairs for example in square a-d it contains a and d. The third step is to take the implied pairs are now entered into each of the non X squares. Then this is repeated till no more X's are added.

2.4 Use Of Different Flip Flops

According to (Lowe) in *"electronics, a flip-flop is a special type of gated latch circuit. There are several different types of flip-flops."* The most common types of flip flops are:-

- SR Flip Flop
- D Flip Flop

A SR flip flop is very similar to that of an SR latch it has three inputs that includes a clock input and a set and reset input. The Q output changes depending on the Set and reset values for example if the set value is high the output value is high and if the reset value is high the output value is low. Also it is very important to make sure that both the inputs are not high once the clock has been triggered as that would be considered an invalid input condition.

A D flip flop only has one additional input other than the clock input and that it's called the Data input and once the clock has been triggered it would result in the output matching the input called data. This means that if the data input is high then the output will be high and in the case that the data input is low and the output would be low.

3. Design Method, Choices and Calculations

3.1 Design 5 Second Delay

The first step in designing the 5 second delay for the circuit requires designing a state diagram for the circuit for the following states:-

- S0 – Count 0
- S1 – Count 1
- S2 – Count 2
- S3 – Count 3
- S4 – Count 4
- S5 – Count 5
- S6 – Count 6

The state diagram for this can be seen in figure 10 below:-

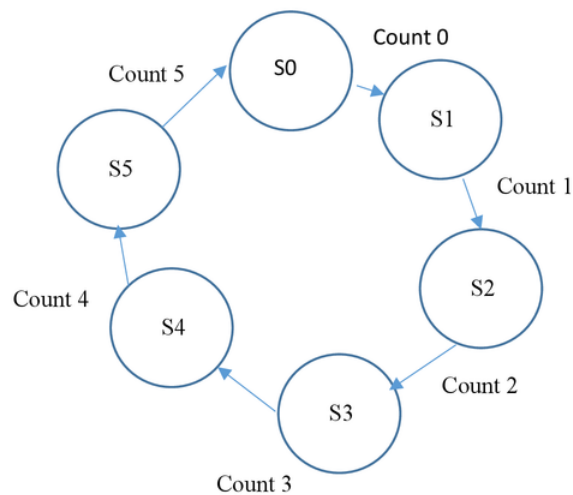


Figure 10 - 5 Second Delay State Diagram

Following the state diagram the next part of the process requires creating a state table for the 5 second delay that can be seen below:-

		Current			Next			Output
Current	Next	Qa	Qb	Qc	Qa	Qb	Qc	
S0	S1	0	0	0	0	0	1	0
S1	S2	0	0	1	0	1	0	0
S2	S3	0	1	0	0	1	1	0
S3	S4	0	1	1	1	0	0	0
S4	S5	1	0	0	1	0	1	0
S5	S6	1	0	1	0	0	0	1
		X	X	X	X	X	X	X
		X	X	X	X	X	X	X

Once the state table has been designed now the karnaugh maps need to be created in order to simulate the circuit in multi-sim and the karnaugh maps for the 5 second delay can be seen in figure 11.

QA+

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

In the table above it shows the karnaugh map for Qa and the result from this karnaugh map gives the following output:-

$$Qa = A \text{Not} C + BC$$

QB+

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

In the table above it shows the karnaugh map for Qb and the result from this karnaugh map can be seen below:-

$$QB = B \text{Not} C + \text{Not} A \text{Not} BC$$

QC+

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

In the table above it shows the karnaugh map for QC and the result from this karnaugh map can be seen below:-

$$QC = \text{Not} C$$

Now that the karnaugh maps have been designed and all the outputs have been completed the circuit in multisim can be built as can be seen below:-

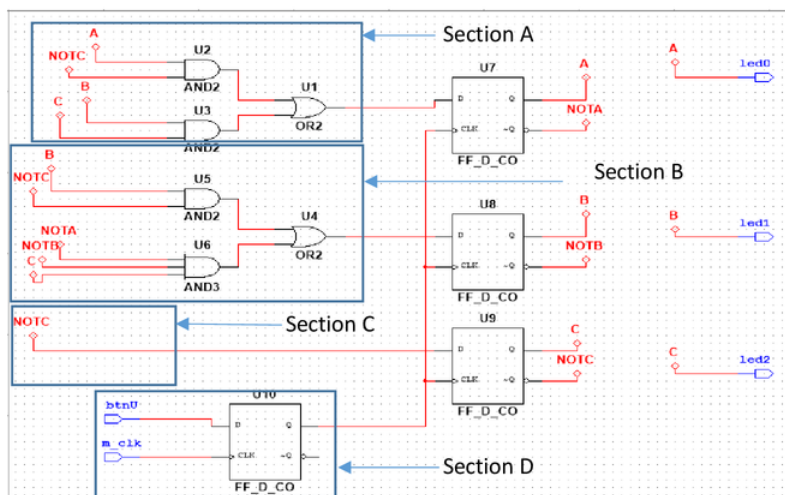


Figure 11 - 5 Second Delay Circuit

In figure 11 it shows section A it shows the results from the first karnaugh map with the initial and gate being an A NotC and the second and gate is BC both and gates are then connected to an OR gate. In section B there are two and gates connected to an OR gate the first of which is a B NOTC and the other is a NOTA, NOTB and C with the OR gate connected to a flip flop. In section C the result of the third karnaugh map NOTC is connected directly to a flip flop. Section D is used to test the circuit by pressing the up button it will change from one state to another.

3.2 15 Second Delay

The first step in the process of designing a 15 second delay is to identify all of the states for the 15 second delay that can be seen below:-

- S0 – Count 0
- S1 – Count 1
- S2 – Count 2
- S3 – Count 3
- S4 – Count 4
- S5 – Count 5
- S6 – Count 6
- S7 – Count 7
- S8 – Count 8
- S9 – Count 9
- S10 – Count 10
- S11 – Count 11
- S12 – Count 12
- S13 – Count 13
- S14 – Count 14
- S15 – Count 15

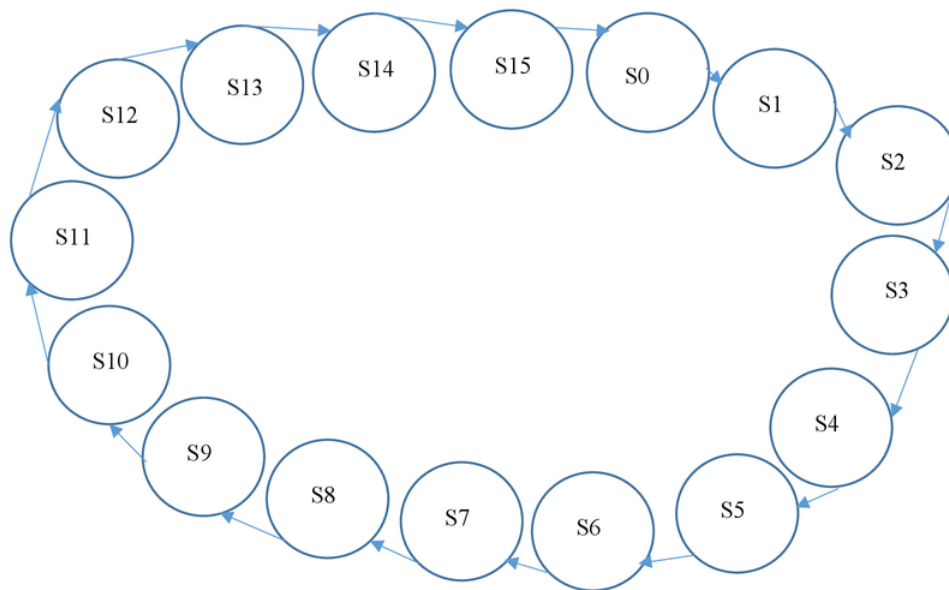


Figure 12 - State Diagram 15 Second Delay

Now that the state diagram for the 15 second delay has been created this means that the next step in the process is to create a truth table for the 15 second delay.

Current	Next	Current				Next			
		Qa	Qb	Qc	Qd	Qa	Qb	Qc	Qd
S0	S1	0	0	0	0	0	0	0	1
S1	S2	0	0	0	1	0	0	1	0
S2	S3	0	0	1	0	0	0	1	1
S3	S4	0	0	1	1	0	1	0	0
S4	S5	0	1	0	0	0	1	0	1
S5	S6	0	1	0	1	0	1	1	0
S6	S7	0	1	1	0	0	1	1	1
S7	S8	0	1	1	1	1	0	0	0
S8	S9	1	0	0	0	1	0	0	1
S9	S10	1	0	0	1	1	0	1	0
S10	S11	1	0	1	0	1	0	1	1
S11	S12	1	0	1	1	1	1	0	0
S12	S13	1	1	0	0	1	1	0	1
S13	S14	1	1	0	1	1	1	1	0
S14	S15	1	1	1	0	1	1	1	0
S15	S0	1	1	1	1	0	0	0	0
		X	X	X	X	X	X	X	X
		X	X	X	X	X	X	X	X

Now that the truth table for the 15 second delay has been designed the karnaugh maps can be created so the simulation of the circuit can begin.

Qa+

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

The result from the karnaugh maps can be seen below:-

$$Qa = ANOTC + ANOTD + ANOTB + NOTABCD$$

Qb+

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

The result from the karnaugh map above can be seen below:-

$$Qb = BNOTC + BNOTD + NOTBCD$$

QC+

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

The result from the karnaugh map above can be seen below:-

$$QC = \text{NOTCD} + \text{CNOTD}$$

QD+

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

The result from the karnaugh map above can be seen below:-

$$QD = \text{NOTD}$$

Once all of the karnaugh maps have been completed the circuit can be built in multi-sim as can be seen below:-

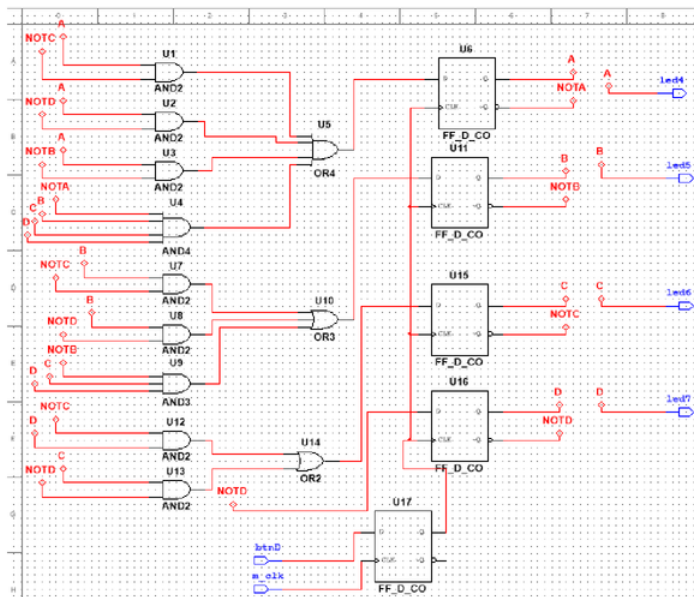


Figure 13 - Multi-Sim 15 Second Delay

In figure 13 the circuit for multi-sim circuit each of the OR gates are an output from the karnaugh maps for the 15 second delay.

3.3 Main Controller

The first step in designing the main controller is to identify all of the states associated with the main controller as can be seen below:-

- S0 – Count 0
- S1 – Count 1
- S2 – Count 2
- S3 – Count 3
- S4 – Count 4
- S5 – Count 5

The next step is to make up a truth table for the main controller as can be seen below:-

Current	W = 0		W = 1		Current			If w= 0 T=0			If w= 0 T=1		
	T=0	T=1	T=0	T=1	Qa	Qb	Qc	Qa	Qb	Qc	Qa	Qb	Qc
S0	S0	S0	S1	S1	0	0	0	0	0	0	0	0	0
S1	S2	S2	S2	S2	0	0	1	0	1	0	0	1	0
S2	S0	S3	S3	S3	0	1	0	0	1	1	0	1	1
S3	S3	S4	S3	S4	0	1	1	0	1	1	1	0	0
S4	S5	S5	S5	S5	1	0	0	1	0	1	1	0	1
S5	S0	S0	S0	S0	1	0	1	0	0	0	0	0	0

If W=1 T=0			If W=1 T=1			Output		
Qa	Qb	Qc	Qa	Qb	Qc	R	A	G
0	0	1	0	0	1	0	0	1
0	1	0	0	1	0	0	1	0
0	1	1	0	1	1	0	1	0
1	1	1	1	0	0	1	0	0
0	0	1	1	0	1	1	1	0
0	0	0	0	0	0	1	1	0

Following building the truth table the next step will be to design the karnaugh maps for the circuit that can be seen below:-

Qa+

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

Qa = A NOTC + TBC

Qb+

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

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

Qc+

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

$$Q_c = A \text{ NOT } B \text{ NOT } C + W \text{ NOT } C + B \text{ NOT } C + \text{NOT } B$$

R

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

$$R = A + BC$$

A

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

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

G

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

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

Once the karnaugh maps have been completed the multi-sim circuit can then be designed and simulated as can be seen below:-

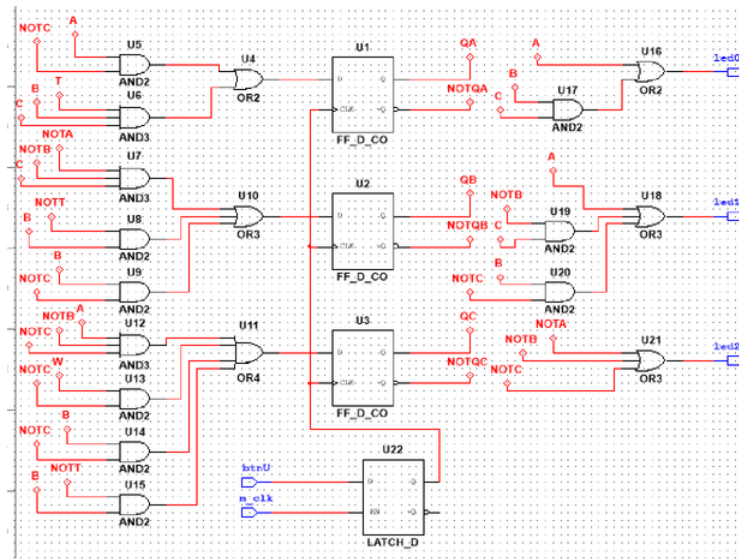


Figure 14 - Main Controller Circuit

4. Testing and Improving the Design

4.1 5 Second Delay

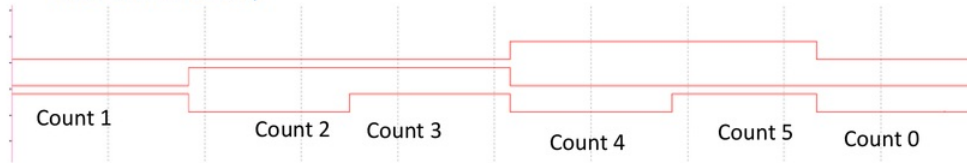


Figure 15 - 5 Second Delay Logic Analyser

In figure 15 it shows the states for the 5 second delays logic analyser along with all of the states and below screenshots for each of the states example on the test board.

State 0

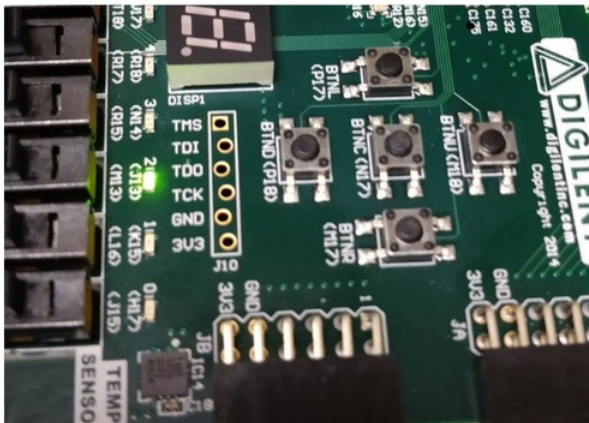


Figure 16 - 5 Second Delay State 0

State 1



Figure 17 - 5 Second Delay State 1

State 2



Figure 18 - 5 Second Delay State 2

State 3

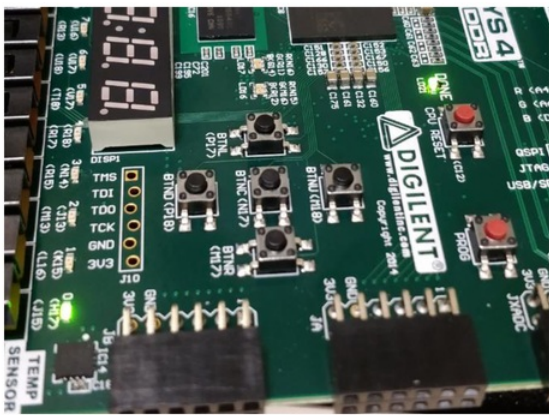


Figure 19 - 5 Second Delay State 3

State 4

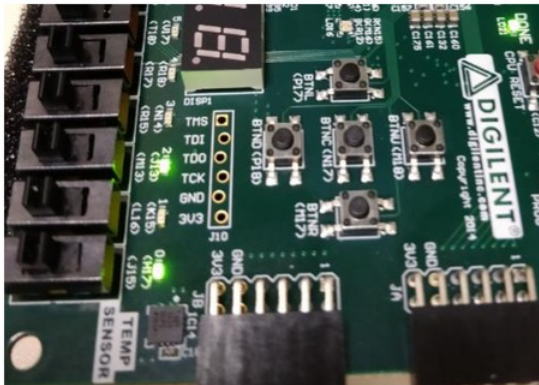


Figure 20 - 5 Second Delay State 4

State 5



Figure 21 - State 5 5 Second Delay

4.2 15 Second Delay

The screenshot below shows the results of a logic analyser on the 15 second delay circuit with all of the states associated with the 15 second delay timer.

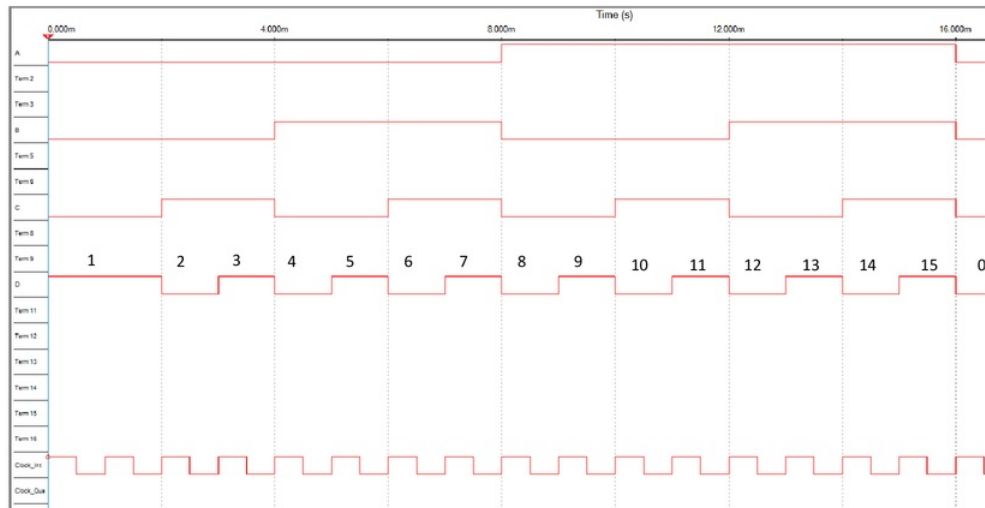


Figure 22 - 15 Second Delay Logic Analyser

4.3 System Built In Multi-Sim

In figure 23 it shows the complete circuit that was built within multi-sim:-

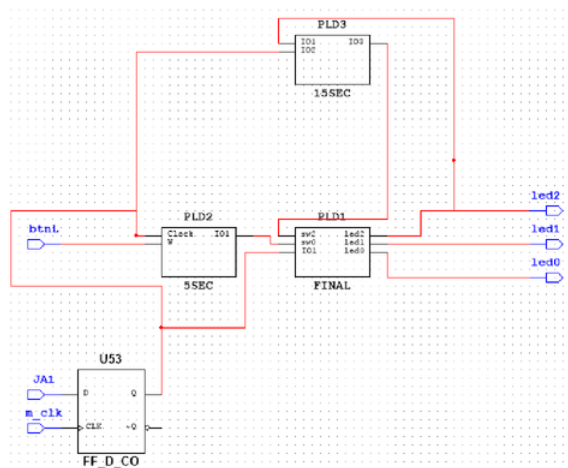


Figure 23 - Complete Multi-SIm Circuit

5. Conclusions

The first part of the project was to design a puffin crossing system that required the use of a Digilent Nexys 4 DDR FPGA board. The result of this I needed to carry out some preliminary research into the different ways of implementation that could be carried out for a finite state machine this meant that I had to look into both the positive and negatives of using the different methods. The next step was to decide on the best method for implementing the finite state machine so I needed to carry out research into the moore and mealy machines. Following deciding on the best method of implementating additional research then needed to be carried out on state reduction techniques to limit the amount of states used and the different types of flip flops that could be used in the design of a puffin crossing.

Then following carrying out the preliminary research I then needed to look into creating the truth tables for all of the different states for that pelican crossing this required the need to take into account a delay between the traffic and the wait button that would be pressed by the pedestrian.

Once I had created the truth tables successfully it meant I needed to begin to design the karnaugh maps for the circuit that is vital as if I did not I would not have been able to build the circuit. Once I had built the circuit in multi-sim I had to run a logic analyzer to test the circuit was running as expected.

6. References

Banerjee, P., 1998. *Finite State Machine*, N/A: N/A.

Bevilacqua, F., 2013. *Game Development*. [Online]
Available at: gamedevelopment.tutsplus.com
[Accessed 20 March 2018].

Circuits, A. A., N/A. *All About Circuits*. [Online]
Available at: <https://www.allaboutcircuits.com/textbook/digital/chpt-11/finite-state-machines/>
[Accessed 23 April 2018].

Ehab, D., 2018. *UO Babylon*. [Online]
Available at: http://repository.uobabylon.edu.iq/2010_2011/3_30940_163.pdf
[Accessed 23 April 2018].

Gurp, J. V., 1999. *On The Implementation Of Finite State MACHines*, s.l.: University Of Karlskrona.

James, M., 2018. *I-Programmer*. [Online]
Available at: <http://www.i-programmer.info/babbages-bag/223-finite-state-machines.html>
[Accessed 23 April 2018].

Kun, J., 2015. *Prezi*. [Online]
Available at: <https://prezi.com/dd7okw6szrqj/state-reduction-implication-chart-method/>
[Accessed 23 April 2018].

Lowe, D., N/A. *Dummies*. [Online]
Available at: <http://www.dummies.com/programming/electronics/diy-projects/digital-electronics-types-of-flip-flop-circuits/>
[Accessed 23 April 2018].

Point, T., 2018. *Tutorials Point*. [Online]
Available at: https://www.tutorialspoint.com/automata_theory/moore_and_mealy_machines.htm
[Accessed 23 April 2018].

Assignment 2

GRADEMARK REPORT

FINAL GRADE

GENERAL COMMENTS

Instructor

70/100

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

DECLAN DAVIES**70****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 good. There were very few gaps in your knowledge of the basic principles and your understanding of the higher level concepts was demonstrated. You could have explained in more detail how the circuit evolved and the design process for how the overall system evolved.

22**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.

11

PRACTICAL APPLICATION AND DEP.



Your application of practical skills for this task were generally OK, 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.



In this assignment you have demonstrated skills in working individually and also collaboratively. Your written report was very well structured and written in a coherent and logical way. The report was presented in a professional manner using appropriate language and formatting, good diagrams in appropriate places, good sentence/paragraph structure and clear calculations

8