

PSDE 4004 Assignment 2

by Jac Williams

FILE	11945_JAC_WILLIAMS_PSDE_4004_ASSIGNMENT_2_152847_2095255227.DOCX		
TIME SUBMITTED	30-APR-2018 11:17PM (UTC+0100)	WORD COUNT	5475
SUBMISSION ID	87960818	CHARACTER COUNT	33956



ELECTRONICS ASSIGNMENT 2: DIGITAL CIRCUIT DESIGN PSDE 4004



Jac Williams
TATA STEEL

Table of Contents

Table of Figures:.....	2
Table of Tables:.....	4
Introduction:	5
Research:.....	6
Comparison of combinational and sequential Logic:.....	6
Combinational:.....	6
Sequential:	7
Finite State Machines:	8
Comparison of FPGA's and ASIC's:.....	10
Circuit Design:	11
Decoder:.....	12
Design:.....	12
Non-PLD:	17
PLD:	21
Decade Counter:	26
Design:.....	26
Decade and Decoder combined:.....	35
0-5 Counter:	39
Design:.....	39
0-5 Counter and Decoder combined:	45
Finished circuit	48
Non-PLD:	48
PLD:	51
Conclusions:	58
Works Cited	59

Table of Figures:

Figure 1 - AND Logic Gate	6
Figure 2 - Synchronous and Asynchronous Sequential Logic (Agarwal, 2017)	7
Figure 3 - Mealy FSM Machine (Weatherspoon, 2013)	8
Figure 4 - Moore FSM Machine (Weatherspoon, 2013)	9
Figure 5 - Basic FPGA architecture overview (Zeidman, 2006)	10
Figure 6 - 7 Segment Display	12
Figure 7 - 7 Seg Display "Number 2"	12
Figure 8 - Decoder Non-PLD CCT	17
Figure 9 - Decoder Non-PLD testing inputs	18
Figure 10 - Decoder Testing Binary Count 15	19
Figure 11 - Decoder Testing Binary Count 6	19
Figure 12 - Decoder Testing Binary Count 5	20
Figure 13 - Decoder Testing Binary Count 3	20
Figure 14 - Decoder PLD	21
Figure 15 - Decoder PLD testing 0	22
Figure 16 - Decoder PLD testing 1	23
Figure 17 - Decoder PLD testing 2	23
Figure 18 - Decoder PLD testing 4	24
Figure 19 - Decoder PLD testing 8	24
Figure 20 - Decoder PLD testing 5	25
Figure 21 - Decade Counter State Diagram	27
Figure 22 - Decade Counter Non-PLD CCT	30
Figure 23 - Decade Counter Non-PLD Logic Analyser	31
Figure 24 - Decade Counter PLD CCT	32
Figure 25 - Decade counter PLD 1	33
Figure 26- Decade counter PLD 2	33
Figure 27 - Decade counter PLD 5	34
Figure 28 - Decade counter PLD 9	34
Figure 29 - Decade sub CCT	35
Figure 30 - Decade and Decoder combined PLD CCT	36
Figure 31 - Decade and Decoder PLD test 1	37
Figure 32 - Decade and Decoder PLD test 3	37
Figure 33 - Decade and Decoder PLD test 9	38
Figure 34 - 0-5 Counter State Diagram	39
Figure 35 - 0-5 Counter Non-PLD CCT	42
Figure 36 - 0-5 Counter Logic Analyser	43
Figure 37- 0-5 Counter PLD CCT	44
Figure 38 - 0-5 Counter and Decoder Interfacing	45
Figure 39 - 0-5 Counter and decoder PLD CCT	46
Figure 40 - 0-5 and decoder test max count (5)	47
Figure 41 - Final CCT Non-PLD 01	48
Figure 42 - Final CCT Non-PLD 21	49
Figure 43- Final CCT Non-PLD 38	49
Figure 44 - Final CCT Non-PLD 59	50
Figure 45 - Final CCT Non-PLD 00	50
Figure 46 - Final CCT PLD	51

Figure 47 - Final CCT Clock Sub-CCT	52
Figure 48 - Function Generator and Board	53
Figure 49 - Final CCT Board Count Testing 01	54
Figure 50 - Final CCT Board Count Testing 03	54
Figure 51 - Final CCT Board Count Testing 14	55
Figure 52 - Final CCT Board Count Testing 27	55
Figure 53 - Final CCT Board Count Testing 38	56
Figure 54 - Final CCT Board Count Testing 43	56
Figure 55 - Final CCT Board Count Testing 56	57

Table of Tables:

Table 1 - AND Gate Truth Table	6
Table 2 - Decoder State Table	13
Table 3 - Decoder: Karnaugh Map seg "a"	13
Table 4 - Decoder: Karnaugh Map seg "b"	14
Table 5 - Decoder: Karnaugh Map seg "c"	14
Table 6 - Decoder: Karnaugh Map seg "d"	14
Table 7 - Decoder: Karnaugh Map seg "e"	15
Table 8 - Decoder: Karnaugh Map seg "f"	15
Table 9 - Decoder: Karnaugh Map seg "g"	15
Table 10 - Decade Counter State Allocation Table	26
Table 11 - Decade Counter State Table	27
Table 12 - Decade Counter Karnaugh Map "QA+"	28
Table 13 - Decade Counter Karnaugh Map "QB+"	28
Table 14 - Decade Counter Karnaugh Map "QC+"	28
Table 15 - Decade Counter Karnaugh Map "QD+"	29
Table 16 - 0-5 Counter State Allocation	39
Table 17 - 0-5 Counter State Table	40
Table 18 - 0-5 Counter Karnaugh Map "QA+"	40
Table 19 - 0-5 Counter Karnaugh Map "QB+"	40
Table 20 - 0-5 Counter Karnaugh Map "QC+"	41

Jac Williams

Introduction:

In this assignment we are tasked with creating a digital stopwatch, capable of counting from 0 to 59, and creating the composite sub circuits ourselves using digital design techniques. These circuits are to be simulated on multisim and then, when proven to be working, downloaded to an FPGA which in our case is a Nexys 4 DDR Board.

Research:

Comparison of combinational and sequential Logic:

According to Storey (2009) there are two distinct types of digital systems, the first being combinational, where the output is wholly determined by the states of its present inputs. The second being sequential, wherein the current output is not only determined by the present state of the inputs but also by the succession of inputs leading to the current state.

Combinational:

All basic logic gates are a form of combinational logic, where the output can be discerned solely from the current inputs.

The logic of a combinational circuit can be expressed through the use of truth tables, which can be further condensed into Boolean expressions which encapsulate the logic of the circuit or device in a single Boolean sentence. (A.F.Kana, 2008)

Truth tables are a means of mapping and illustrating the output of a combinational circuit/device to the possible input states.

Using an AND gate as an example, the gate itself can be seen in Figure 1, and the truth table illustrated in Table 1. From the truth table we can discern that an AND gate only outputs a 1 when both of its inputs are 1. Figure 1 - AND Logic Gate

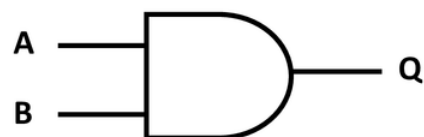


Figure 1 - AND Logic Gate

Inputs		Output
A	B	Q
0	0	0
0	1	0
1	0	0
1	1	1

Table 1 - AND Gate Truth Table

It is advantageous to make the circuit as efficient as possible, typically this means reducing the overall number of gates (utilising circuit reduction techniques such as karnaugh maps, Boolean algebra, and demorgans theorem) or by using only one type of gate (NAND gate replacement techniques) while still meeting the specification of the circuit.

Sequential:

Sequential logic not only takes into account the current state of the inputs when determining the output, but also of previous input states, meaning it contains a form of memory (Storey, 2009).

Sequential circuits can either be synchronous, or asynchronous. Synchronous circuits are governed by a clock signal, whereas asynchronous circuits are determined by changes to the inputs both past and present (Holdsworth, 1991).

Diagrams illustrating these two distinct types of sequential logic can be seen in Figure 2.

With synchronous circuits flip flops are used in order to ensure accurate timing of the state changes, as they must be synchronised. In order to achieve this flip-flops are used. Flip-flops are a form of sequential circuit which takes note of the input and changes the output in accordance with the clock.

Flip Flops are capable of storing a single bit, and therefore multiple flip flops are used in order to create the memory portion of synchronous sequential called shift registers.

(Agarwal, 2017)

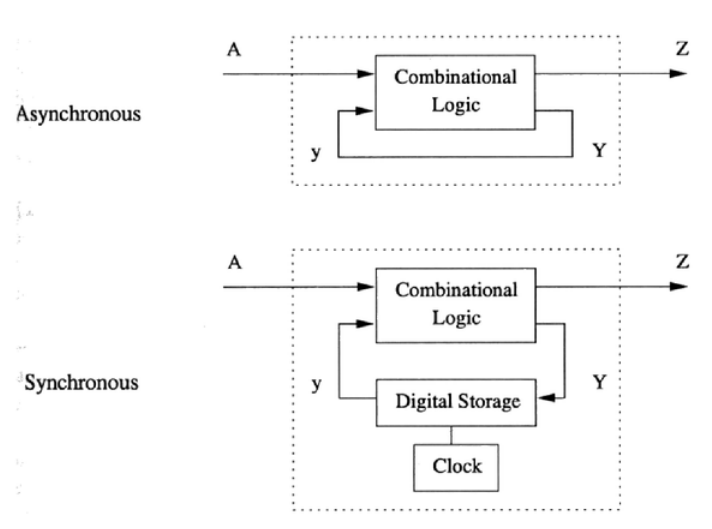


Figure 2 - Synchronous and Asynchronous Sequential Logic (Agarwal, 2017)

Finite State Machines:

Although being purely conceptual the Finite state machines is a useful tool for designing digital circuits, the function of the circuit is condensed into a group of rules and states which determine when the states transition to what next state. They are invaluable in the design process of digital circuits and have many applications.

The formula for designing a digital circuit follows this principle:

1. Look at the specification of the device and what is required of it, and from this ascertain the states transitions needed to achieve this
2. Create a state diagram using the information garnered from step 1 (An example can be seen in Figure 34)
3. Develop a method of interfacing the states and creating them on flip flops (The number of states will be no less than twice the number of flip flops)
4. Tabulate this into a state table (An example can be seen Table 17)
5. Discern the combinational logic required to achieve these states

(Hanson, 2015)

The process in step 5 requires us to draw up karnaugh maps using the state table and identify the Boolean expression that represents the states, from which we can create the combinational logic from logic gates.

There are two variants of finite state machines; a Moore machine, and a Mealy Machine. Both types can be either synchronous or asynchronous depending on whether the state memory is clocked or not.

The Moore machines output is solely dependent upon the current state, whereas the Mealy machine also takes into account the present input. This is illustrated in Figure 3 and Figure 4.

(Weatherspoon, 2013)

General Case: Mealy Machine

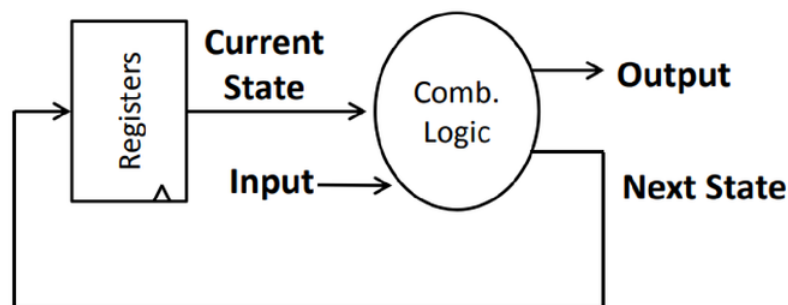


Figure 3 - Mealy FSM Machine (Weatherspoon, 2013)

Special Case: Moore Machine

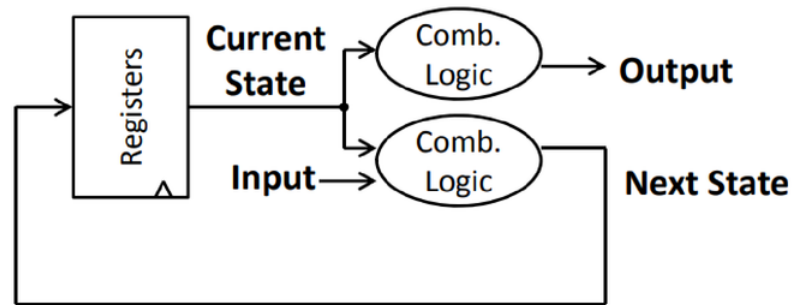


Figure 4 - Moore FSM Machine (Weatherspoon, 2013)

Comparison of FPGA's and ASIC's:

ASIC (Application specific integrated circuit) are integrated circuits (ICs) that are designed for a specific purpose, and come pre-programmed for this purpose. They are ideal for mass-production as they are made to specification with no wastage, containing only what is needed for the final circuit.

Whereas FPGAs (Field programmable gate array) is an IC that doesn't come with a pre-loaded programme, and is designed for testing and prototyping purposes. Conversely to the ASIC the FPGA will likely feature an array of unused transistors as it is created with the design/testing process in mind, where the circuit has not yet been finalised.

Typically when designing a circuit and FPGA will be used for prototyping. Once the circuit has been tested and finalised on an FPGA they will then be converted onto an ASIC where it will be made as efficient and small as possible, ready for cheap mass-production. The ASIC is a small IC chip, which is ideal for mass production, but it is not modifiable once created unlike the FPGA.

(Kilts, 2007)

FPGAs are easily programmable and can be programmed in a short period of time, whilst ASICs take a much longer time to program and is much more complex.

According to Zeidman (2006) In recent years FPGAs have almost superseded the ASIC as more and more often final products remain on FPGA due to increases in processing speed and power consumption.

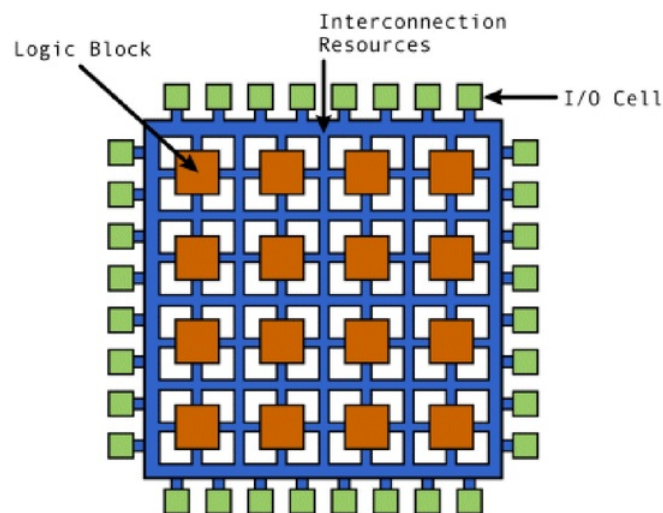


Figure 5 - Basic FPGA architecture overview (Zeidman, 2006)

Circuit Design:

In order to create the final circuit the composite parts which make up the whole must be created. This includes a decade counter for the single unit 0-9 count, a 0-5 counter to make up the count of the tens units, and a decoder in order to convert the binary count of the counters into a decimal count on the 7 segment display.

The design process consisted of designating the state allocations, drawing up a state diagram based off of this, and then finally creating a state table.

From the state table karnaugh maps could be drawn up. Then by grouping the 1's (and x's where necessary) into the largest group of 1, 2, 4, or 8 (and theoretically double 8 and so forth if the map was large enough to warrant it) we could derive Boolean expressions to describe the circuit needed.

By following a Boolean expression a circuit can be created using appropriate logic gates. The circuits were created in Non-PLD mode with a logic analyser on the outputs in order to verify that the circuit was performing according to the specification, and on PLD wherein it was downloaded onto an FPGA (Nexys digital DDR 4 Board) and physically tested.

Once these three circuits were created the counters were separately tested with decoder, and then finally the final circuit was constructed and tested in the same fashion.

Decoder:

Design:

The Decoder is an example of a sequential logic circuit, as it simply derives its output from the current state of its inputs and contains no form of memory. As such a state allocation and state diagram are not applicable.

The outputs of the decoder are the 7 segments of 7 seg display, as depicted in Figure 6. Therefore the state table should contain the states of the 4 bit binary count of the 4 flip flops (A, B, C, and D) needed to activate the segments (a, b, c, d, e, f, g) comprising the desired decimal conversion of the binary. For the binary count, A is the most significant bit and D the least.

I.e. a binary count of 2 would be attained by having C be true, and A, B, and D be false. This would then be converted into a decimal display of "2" by lighting all of the segments except C and F illustrated in Figure 7.

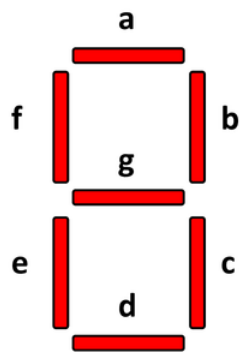


Figure 6 - 7 Segment Display

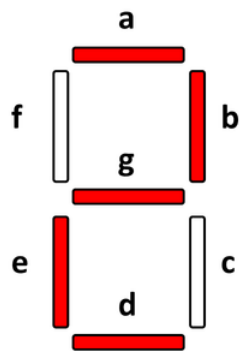


Figure 7 - 7 Seg Display "Number 2"

Table:

Decimal Count	Inputs (Binary)				Outputs						
	A	B	C	D	a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
1	0	0	0	1	0	1	1	0	0	0	0
2	0	0	1	0	1	1	0	1	1	0	1
3	0	0	1	1	1	1	1	1	0	0	1
4	0	1	0	0	0	1	1	0	0	1	1
5	0	1	0	1	1	0	1	1	0	1	1
6	0	1	1	0	1	0	1	1	1	1	1
7	0	1	1	1	1	1	1	0	0	1	0
8	1	0	0	0	1	1	1	1	1	1	1
9	1	0	0	1	1	1	1	1	0	1	1
10	x	x	x	x	x	x	x	x	x	x	X
11	x	x	x	x	x	x	x	x	x	x	x

Table 2 - Decoder State Table

Karnaugh Maps:

By looking at Table 2 karnaugh maps can be created for each of the 7 segments. Care must be taken when creating the maps, as along the rows and columns only 1 digit can change at a time. This leads to a sequence of 00, 01, 11, 10 for the pairings (CD, and AB being the pairings) and must be converted from the 00, 01, 10, 11 sequence in the table.

a:

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

Table 3 - Decoder: Karnaugh Map seg "a"

b:

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

Table 4 - Decoder: Karnaugh Map seg "b"

c:

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

Table 5 - Decoder: Karnaugh Map seg "c"

d:

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

Table 6 - Decoder: Karnaugh Map seg "d"

e:

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

Table 7 - Decoder: Karnaugh Map seg "e"

f:

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

Table 8 - Decoder: Karnaugh Map seg "f"

g:

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

Table 9 - Decoder: Karnaugh Map seg "g"

Boolean Expression:

By examining the groupings made on the karnaugh maps shown in Table 3 to Table 9 we can derive the Boolean expression for each segment.

$$a = A + C + \overline{B}\overline{D} + \overline{B}D$$

$$b = \overline{B} + \overline{C}\overline{D} + CD$$

$$c = B + \overline{C} + D$$

$$d = A + \overline{B}\overline{D} + \overline{C}\overline{D} + \overline{B}C + B\overline{C}D$$

$$e = \overline{B}\overline{D} + \overline{C}\overline{D}$$

$$f = A + B + \overline{C}\overline{D}$$

$$g = A + B\overline{C} + \overline{B}C + \overline{C}\overline{D}$$

Non-PLD:

The purpose of creating a non-PLD prototype is that it allows us to test our theory and ensure that the designed circuit is functioning as intended prior to downloading to the FPGA. If we were to go straight to downloading to the FPGA there would be more potential for errors or issues to occur, and so it's good practice to simulate the circuit first to ensure it is working, that way the circuit is known to be functional and any issues that arise upon downloading are not issues with the basic design of the circuit but rather some error with the PLD side of it.

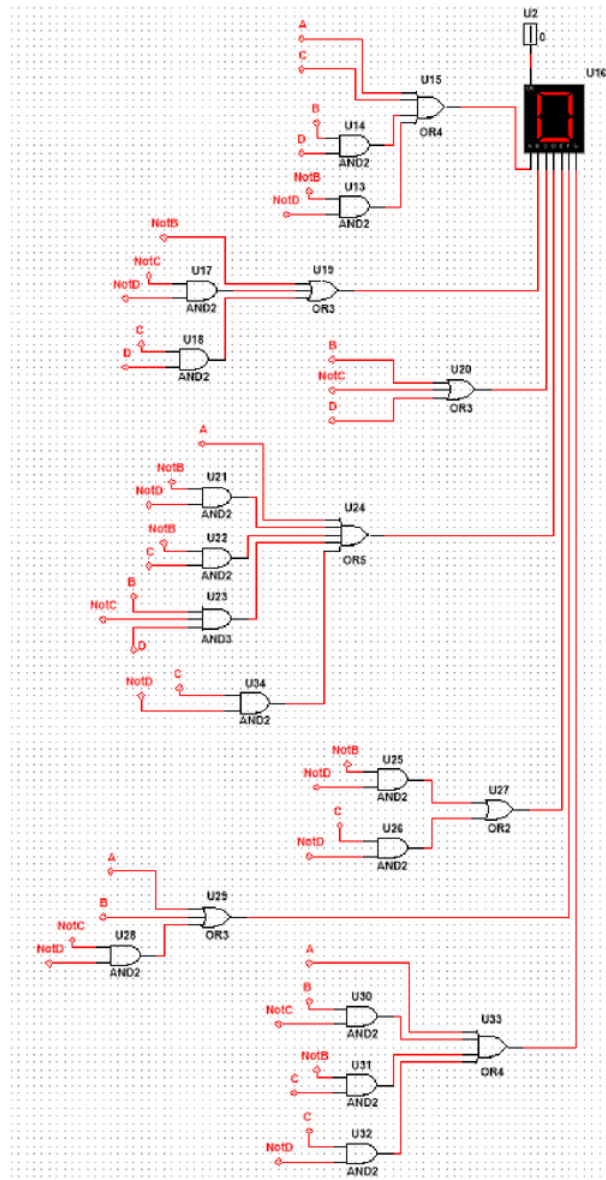


Figure 8 - Decoder Non-PLD CCT

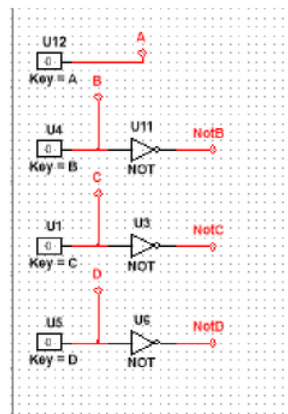


Figure 9 - Decoder Non-PLD testing inputs

The sub circuit displayed in Figure 9 is solely used for testing purposes, as it allows us to manually enter a 4 bit binary count (with D as the least significant bit, and A as the most significant) into the decoder by pressing the relevant key on the keyboard. The reasoning for there being no “NotA” is that in the Boolean expressions “NotA” is never used, and therefore it does not need to be simulated to fully test the decoder.

Testing:

To test the circuit the binary count for 0-9 was manually entered and the display monitored to ensure that it was counting the relevant decimal value of the binary count. Examples of this testing procedure can be seen in Figure 11 to Figure 13. Also in order to ensure the circuit does not pass the maximum count of 9 needed a binary input greater than 9 was entered, in this instance 15 (binary = 1111) and as is evident in Figure 10 the count stops at 9 as desired.

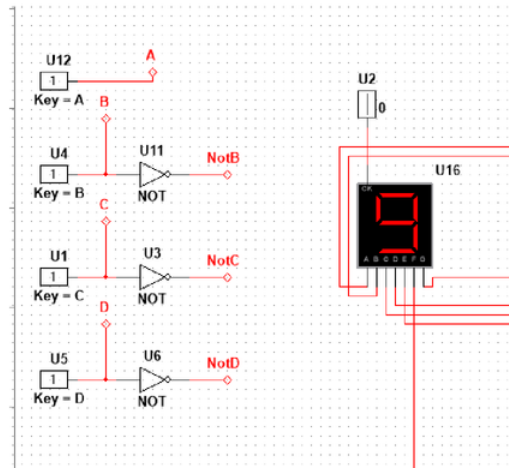


Figure 10 - Decoder Testing Binary Count 15

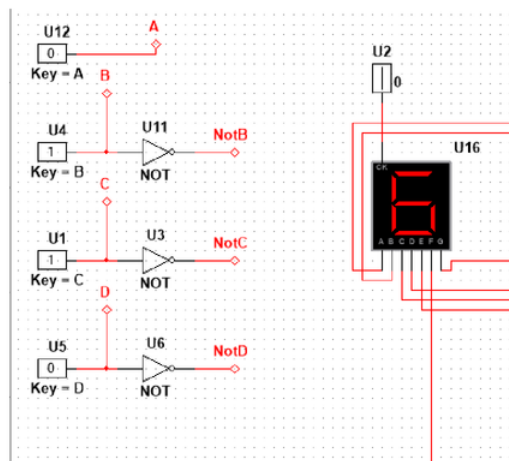


Figure 11 - Decoder Testing Binary Count 6

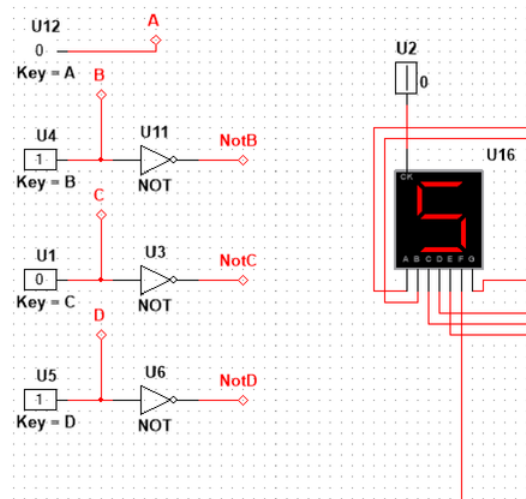


Figure 12 - Decoder Testing Binary Count 5

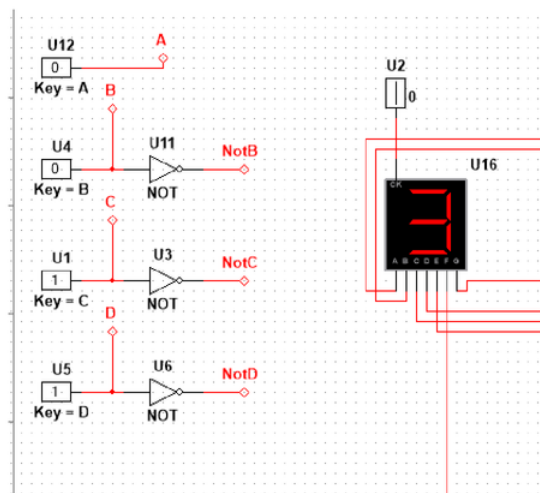


Figure 13 - Decoder Testing Binary Count 3

PLD:

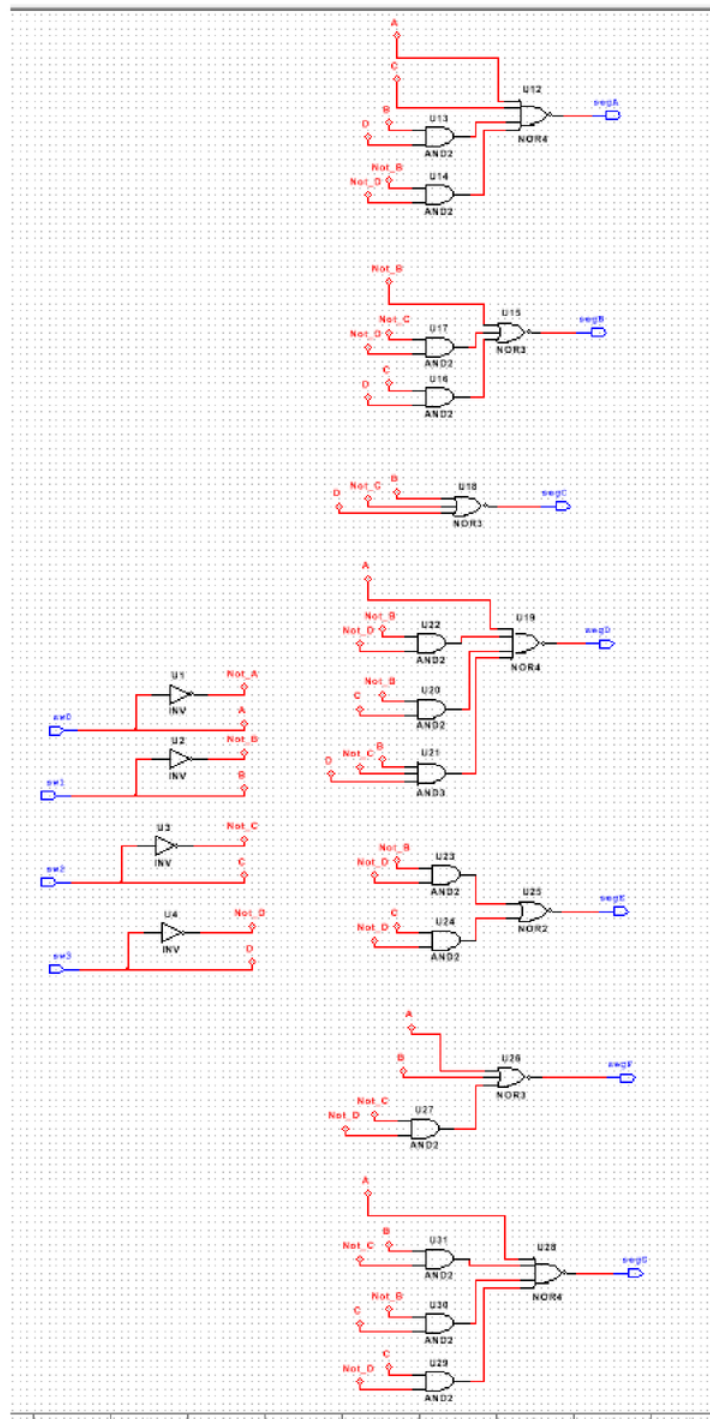


Figure 14 - Decoder PLD

It is important to note that the 7 seg display on the Nexys 4 operates the inverse to the one used on non-PLD and requires a 0 in for the segment to light. In order to achieve this the logic gates going directly to the segments are inverted compared to the ones used in non-PLD. So where a 4 input OR gate was used in the non-PLD a 4 input NOR gate was used in PLD (Figure 14).

This is consistent with every time the decoder is used in PLD.

To test the circuit the 4 switches represented the 4 bits of the binary count and they were switched in combination and the binary count compared with the decimal shown on the 7 seg, as can be seen in Figure 15 to Figure 20.

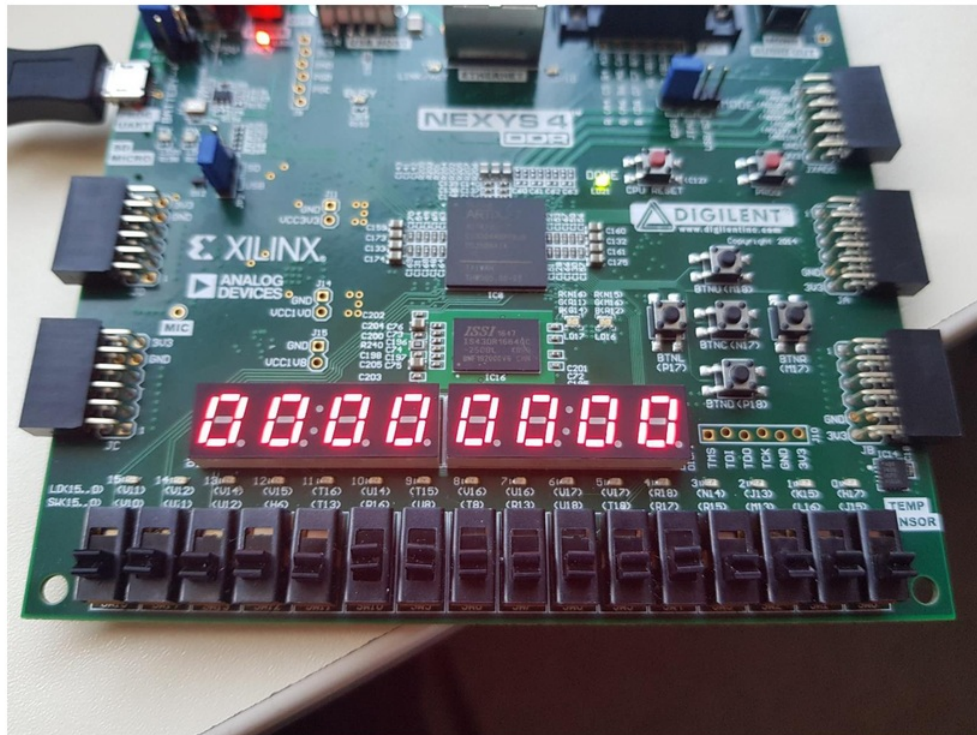


Figure 15 - Decoder PLD testing 0

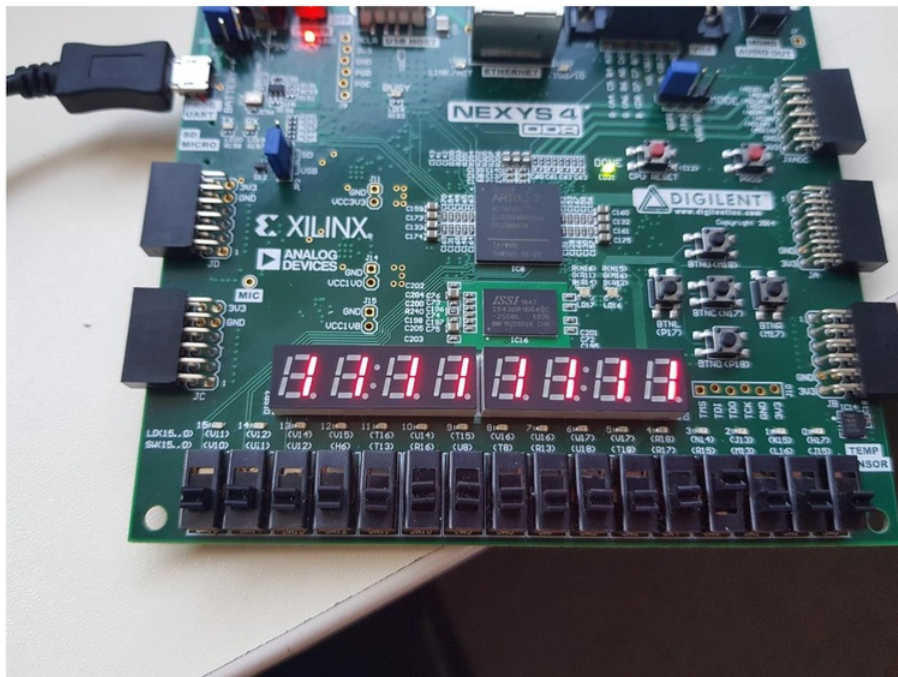


Figure 16 - Decoder PLD testing 1

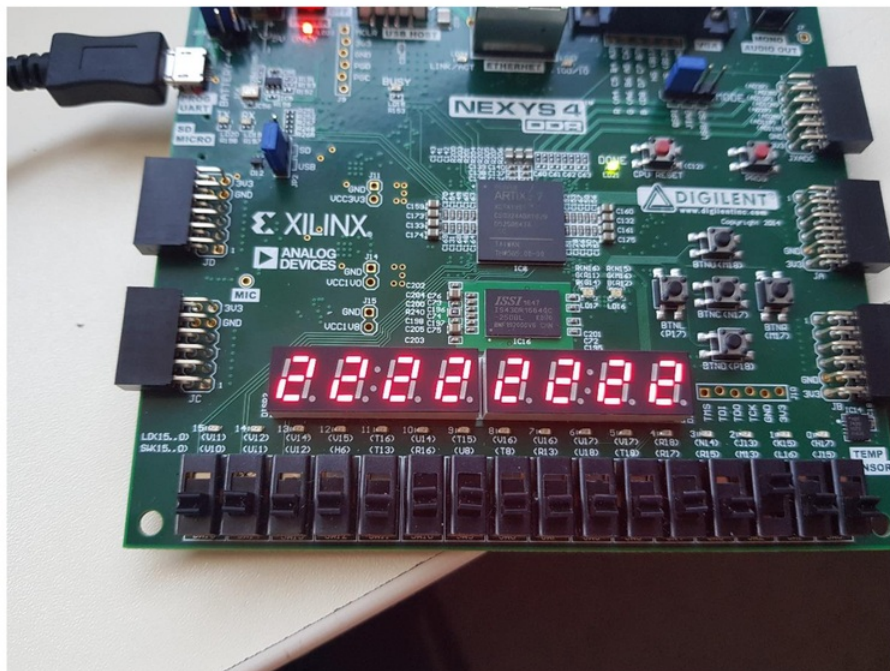


Figure 17 - Decoder PLD testing 2

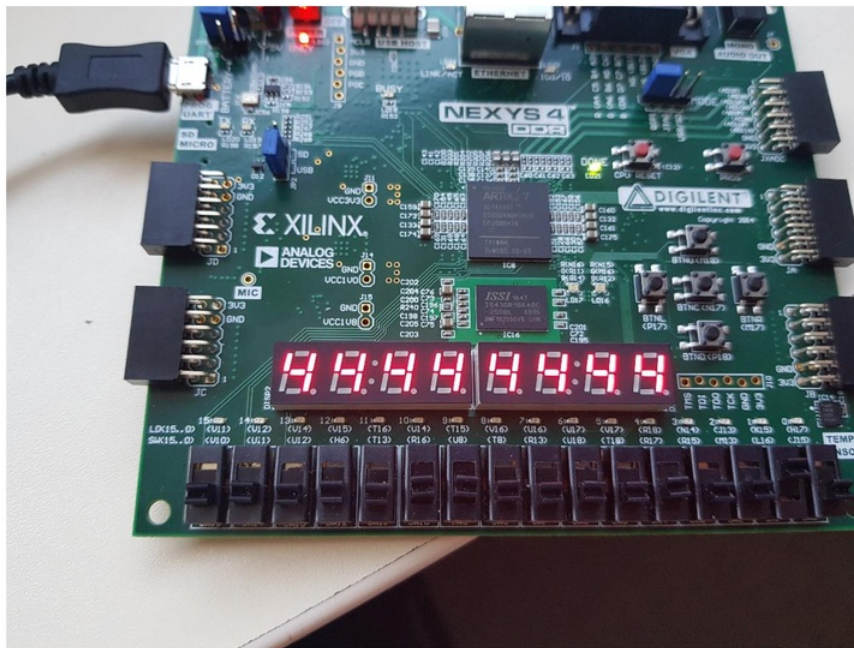


Figure 18 - Decoder PLD testing 4

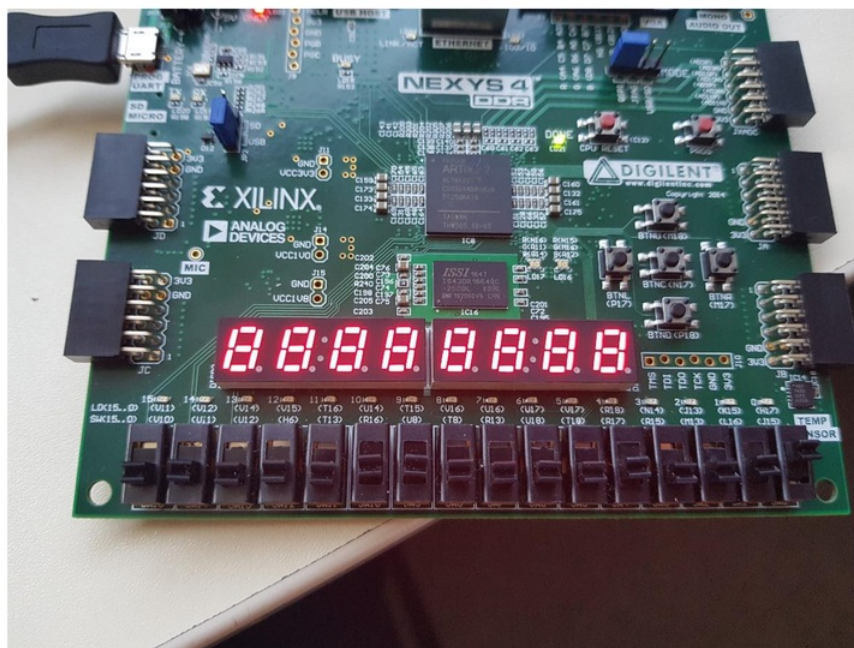


Figure 19 - Decoder PLD testing 8

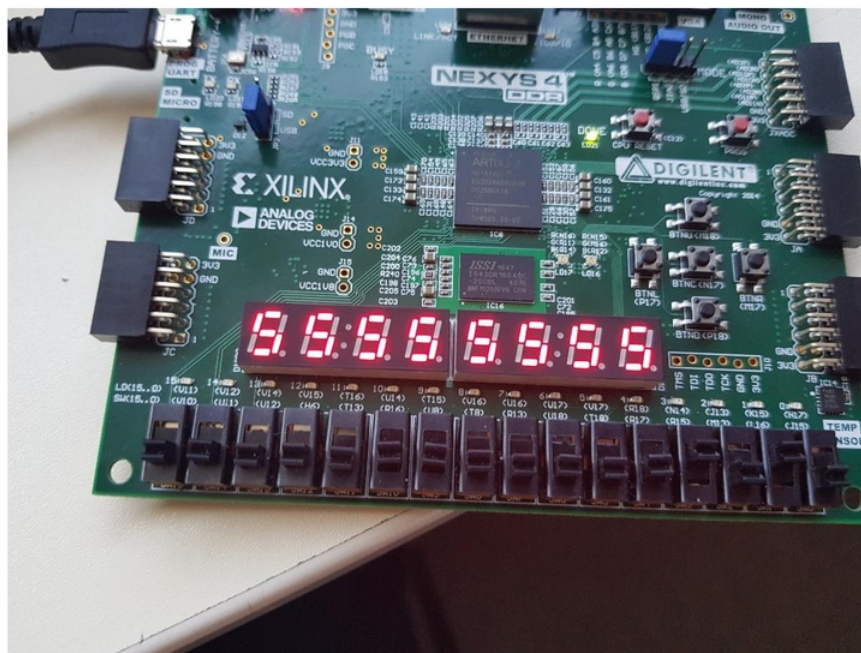


Figure 20 - Decoder PLD testing 5

Decade Counter:

Design:

The decade counter is an example of a synchronous combinational logic circuit. It is required to count from 0 up to 9 in numerical order whenever the clock pulses. Four flip flops must be utilised in order to create the memory needed to count up to 9, as 4 bits are needed to count this high (providing a max count of 15), and 3 flip flops are not enough as they only provide a max count of 7.

The input logic of the four flip flops are taken from the derived Boolean expressions that we ultimately attain after following the design process.

State Allocation:

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

Table 10 - Decade Counter State Allocation Table

State Diagram:

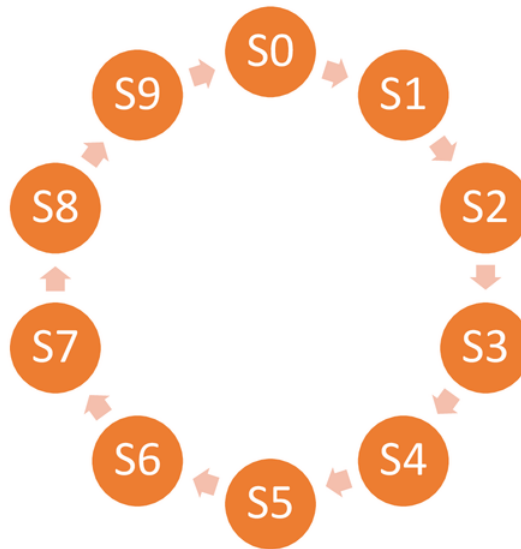


Figure 21 - Decade Counter State Diagram

State Table:

Current	Next	Current				Next			
		Q _A	Q _B	Q _C	Q _D	Q _A ⁺	Q _B ⁺	Q _C ⁺	Q _D ⁺
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	S0	1	0	0	1	0	0	0	0

Table 11 - Decade Counter State Table

Karnaugh Maps:

Q_A^+ :

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

Table 12 - Decade Counter Karnaugh Map "QA+"

Q_B^+ :

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

Table 13 - Decade Counter Karnaugh Map "QB+"

Q_C^+ :

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

Table 14 - Decade Counter Karnaugh Map "QC+"

Q_D^+ :

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

Table 15 - Decade Counter Karnaugh Map "QD+"

Boolean Expression:

$$Q_A^+ = A\bar{D} + BCD$$

$$Q_B^+ = B\bar{D} + B\bar{C} + \bar{B}CD$$

$$Q_C^+ = C\bar{D} + \bar{A}\bar{C}D$$

$$Q_D^+ = \bar{D}$$

Non-PLD:

In order to verify the functionality of the circuit a clock set at 10Hz is supplied to the flip flops and a logic analyser attached to + outputs of each flip flop (shown in Figure 22) so that the binary count can be analysed. The logic analyser was set up according to the clock input and the result can be seen in Figure 23, where it is illustrated that the counter does indeed count from 0 up to 9 and then crucially back to 0.

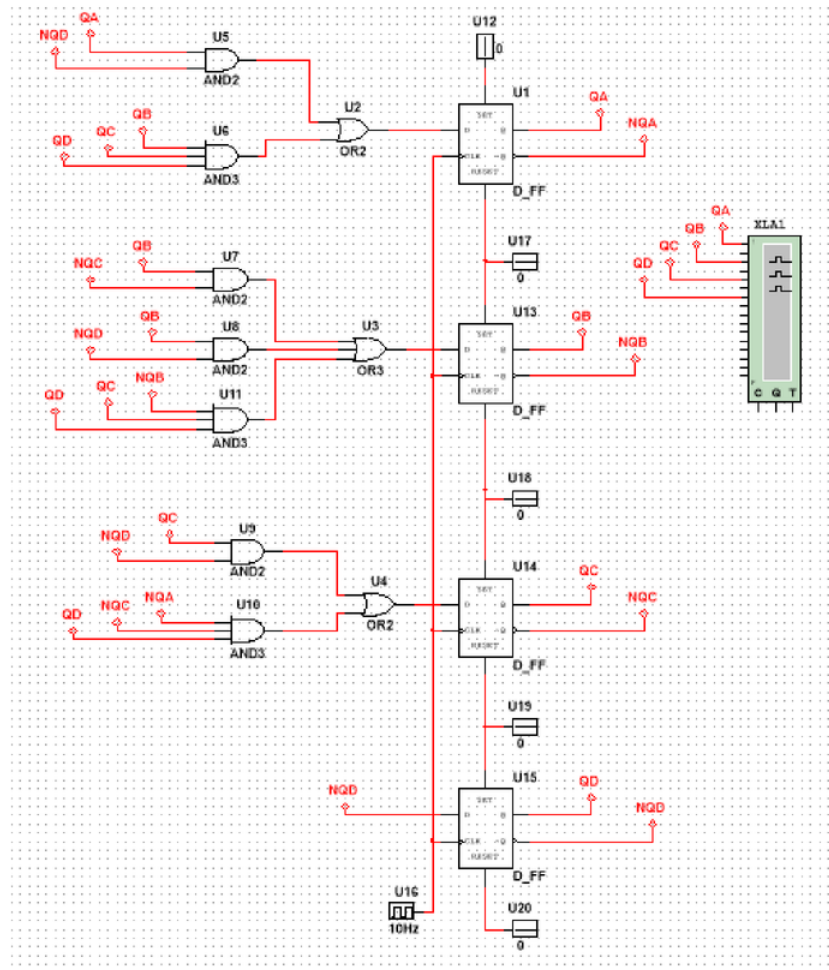


Figure 22 - Decade Counter Non-PLD CCT

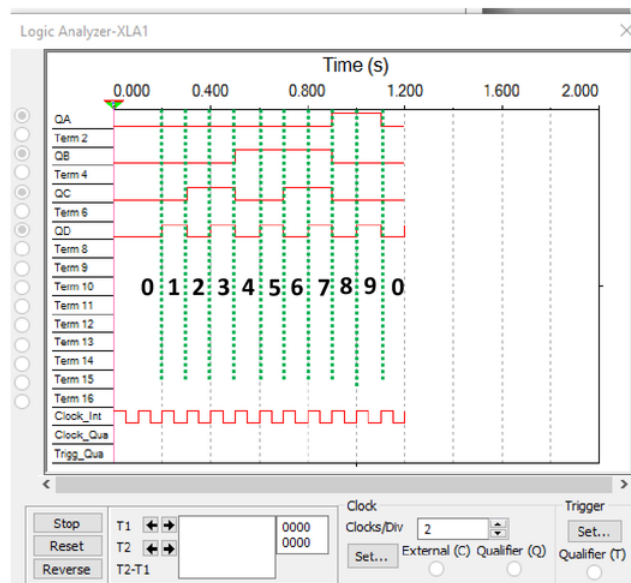


Figure 23 - Decade Counter Non-PLD Logic Analyser

PLD:

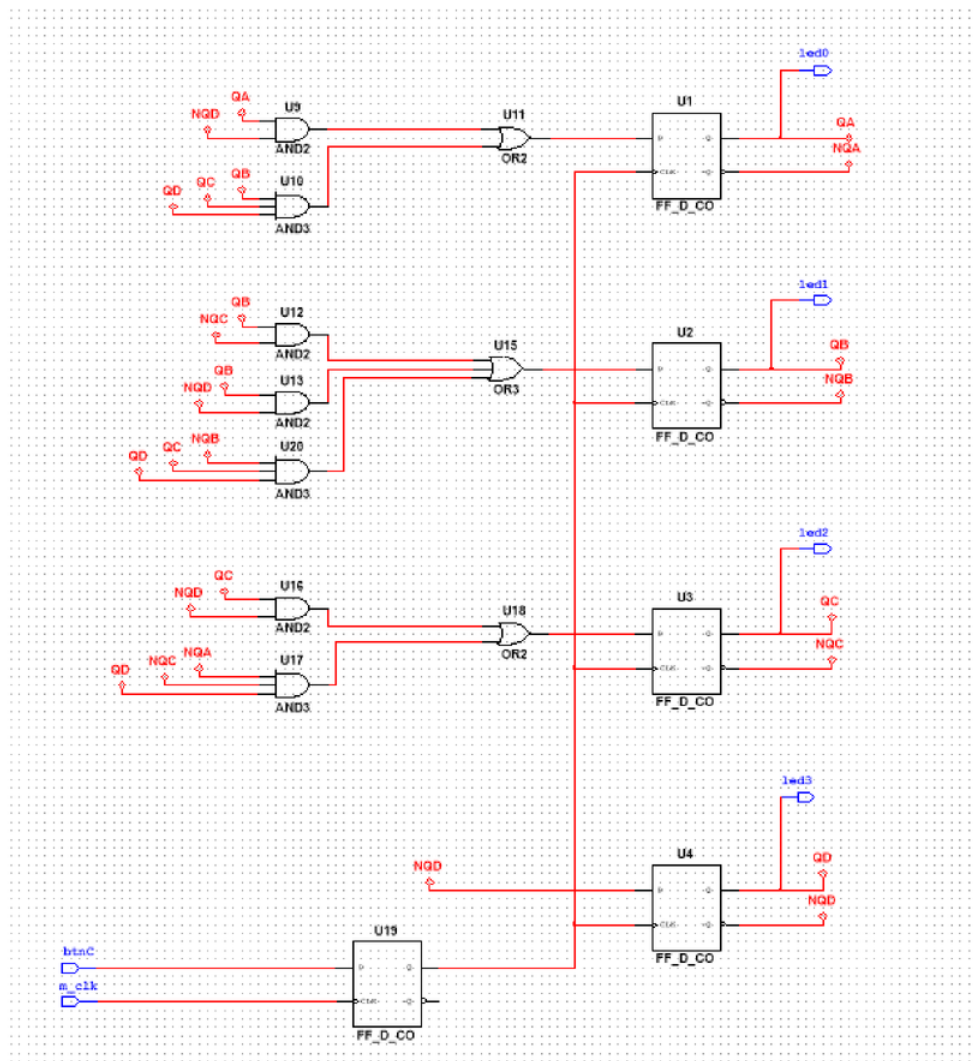


Figure 24 - Decade Counter PLD CCT

The circuit in Figure 24 was downloaded to the board and tested by pressing the centre button to progress the count as displayed in Figure 25 to Figure 28.

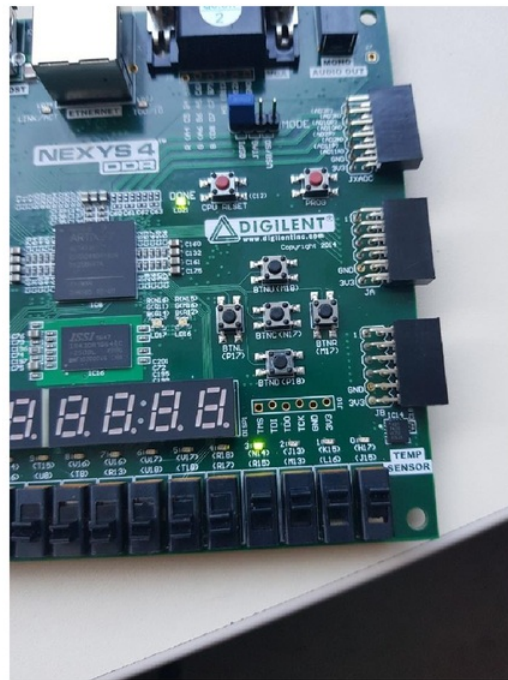


Figure 25 - Decode counter PLD 1

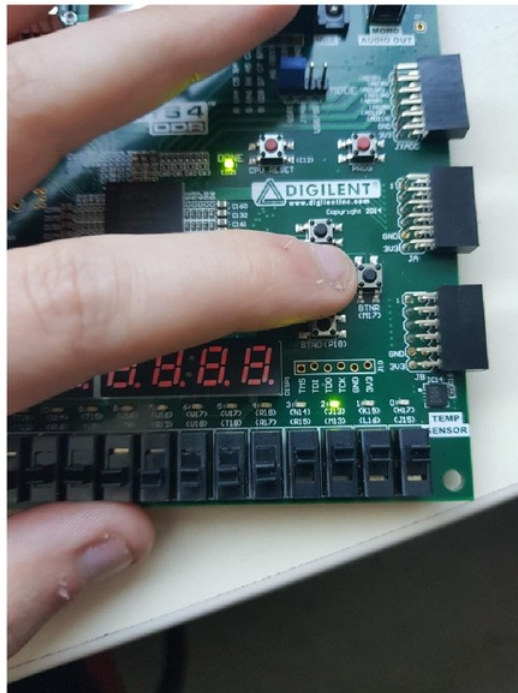


Figure 26- Decode counter PLD 2

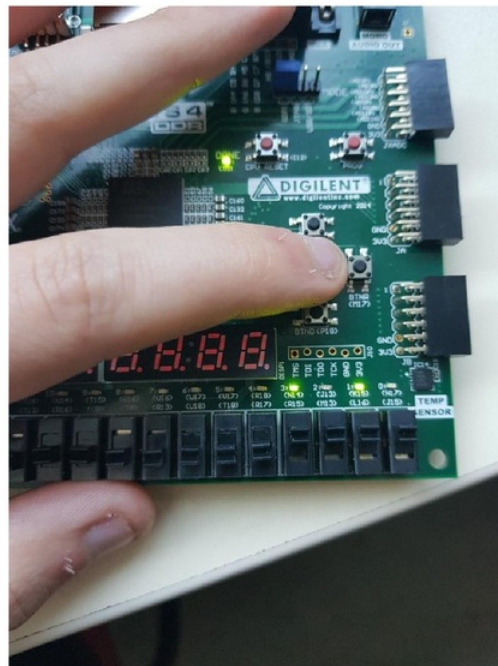


Figure 27 - Decade counter PLD 5

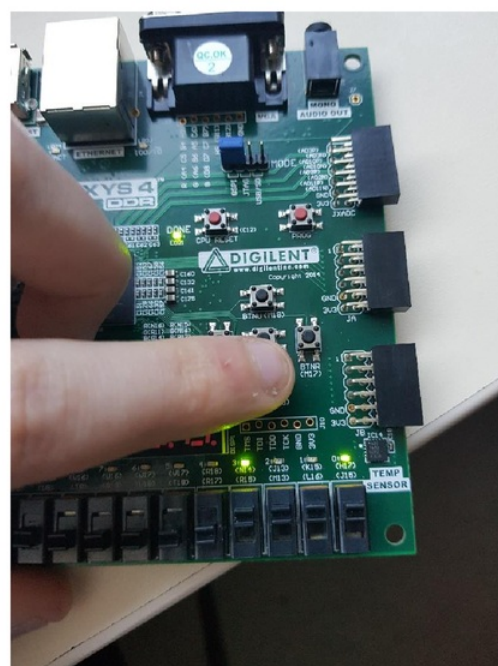


Figure 28 - Decade counter PLD 9

Decade and Decoder combined:

Non-PLD:

To combine the two circuits displayed in Figure 8 and Figure 22 they were simply put into one program, with the decade counter being created as a sub circuit visible in Figure 29. The decoder remains unchanged.

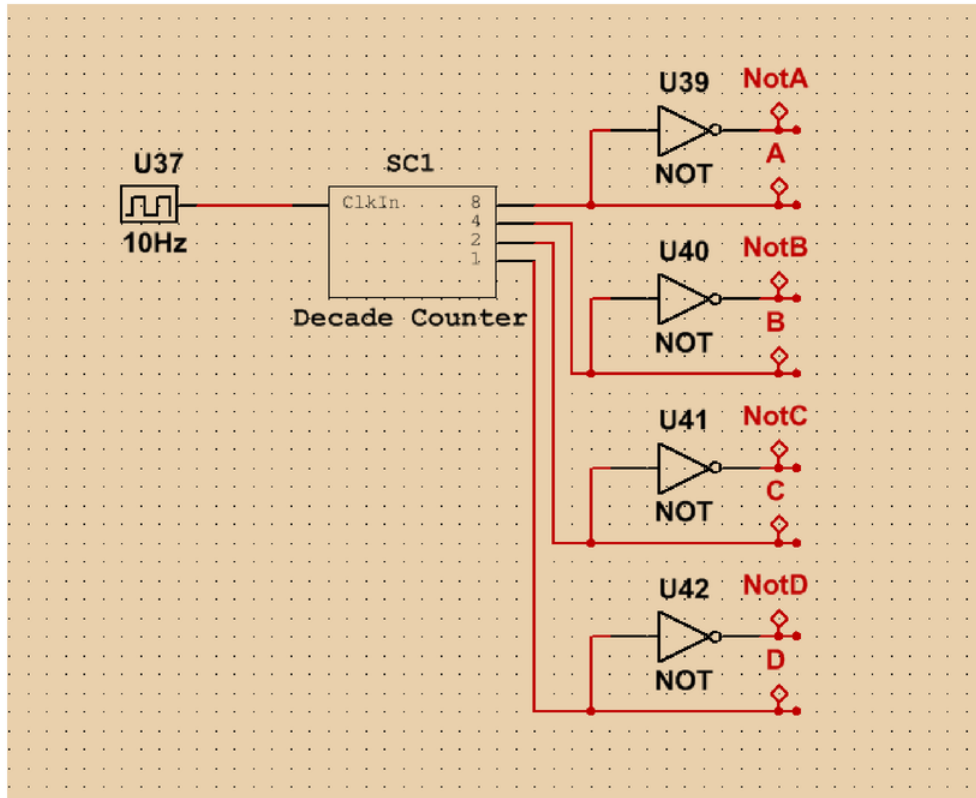


Figure 29 - Decade sub CCT

PLD:

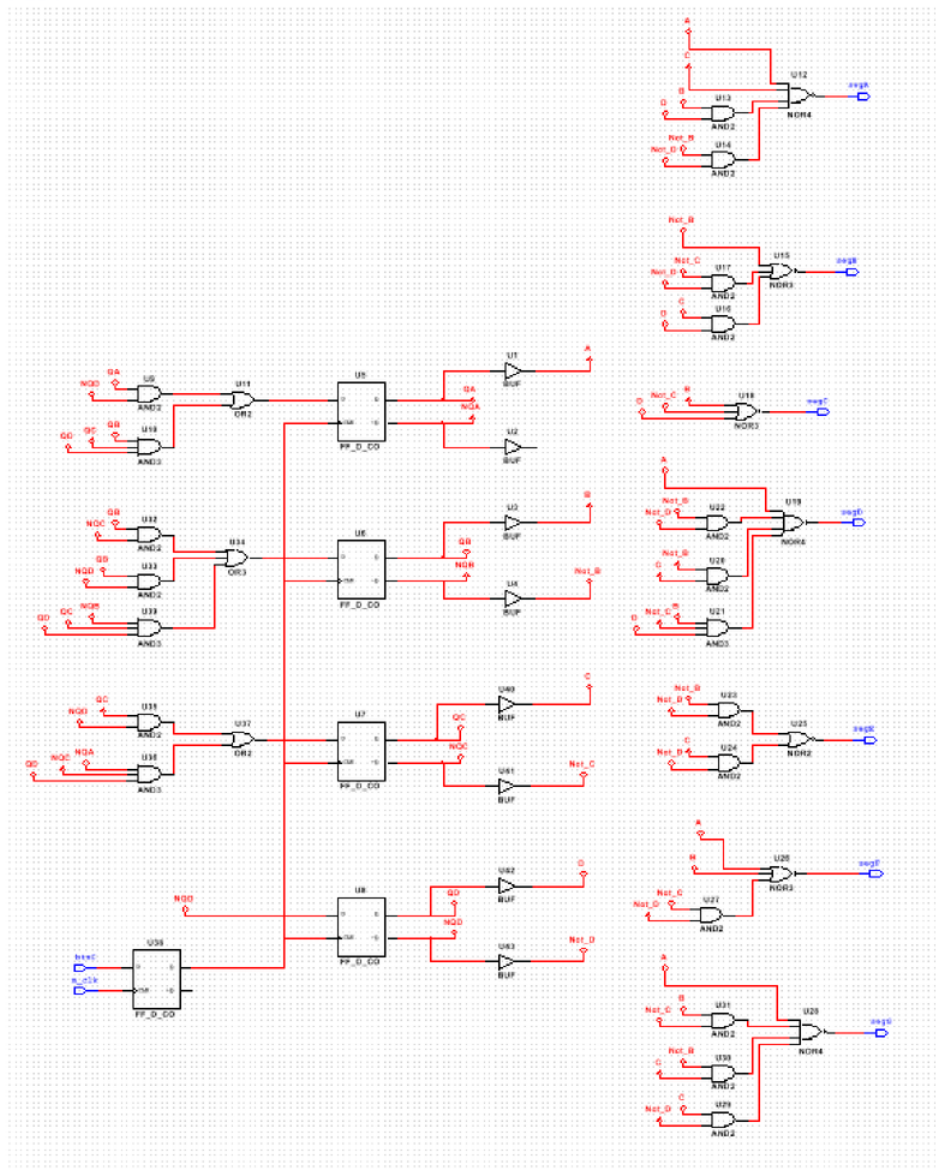


Figure 30 - Decade and Decoder combined PLD CCT

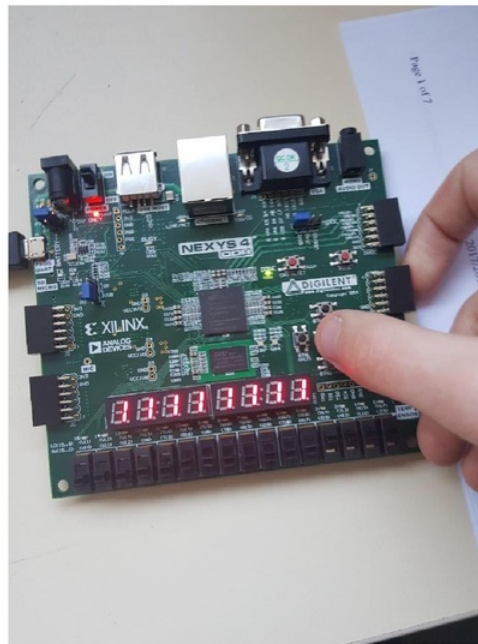


Figure 31 - Decode and Decoder PLD test 1

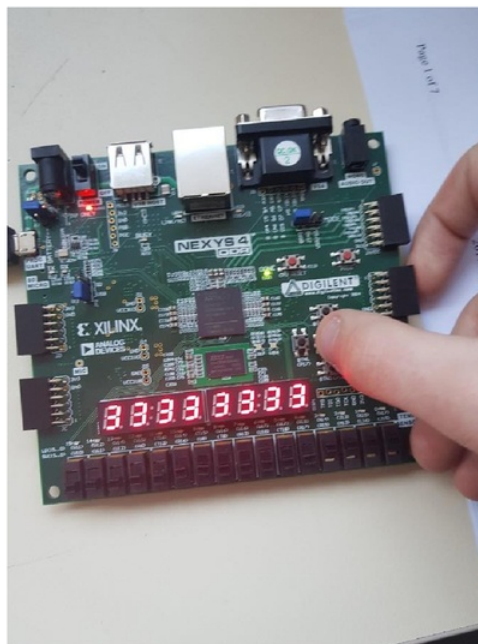


Figure 32 - Decode and Decoder PLD test 3

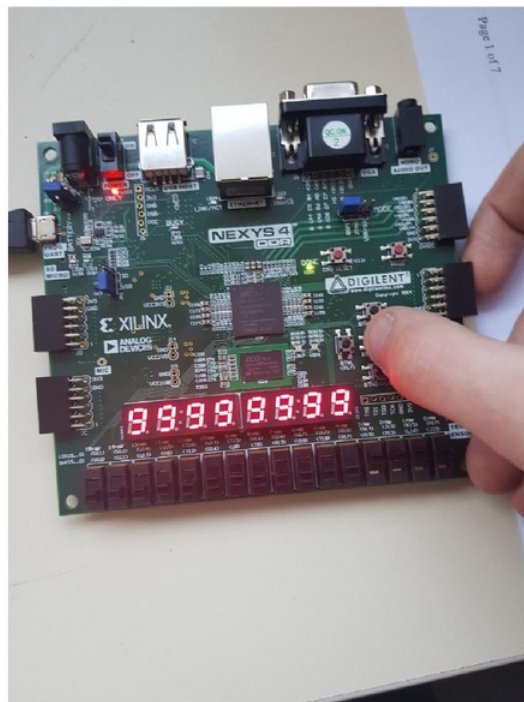


Figure 33 - Decode and Decoder PLD test 9

The circuit Figure 30 is tested with the results show in Figure 31, Figure 32, and Figure 33.

0-5 Counter:

Design:

Similarly to the decade counter the 0-5 counter is also a synchronous sequential logic circuit.

But unlike the decade counter, the 0-5 counter only requires 3 bit binary (and therefore only 3 flip flops) as the maximum it is required to count to is a decimal 5. 2 bit binary provides us with a max count of 3, which is too low, and 3 bit binary a maximum count of 7 which allows us to reach the max 5 (and must be capped at this to prevent counting to 7).

State Allocation:

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

Table 16 - 0-5 Counter State Allocation

State Diagram:

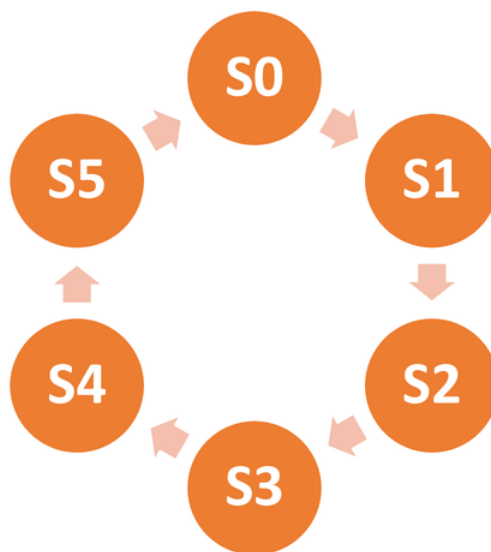


Figure 34 - 0-5 Counter State Diagram

State Table:

		Current			Next		
Current	Next	Q_A	Q_B	Q_C	Q_A^+	Q_B^+	Q_C^+
S0	S1	0	0	0	0	0	1
S1	S2	0	0	1	0	1	0
S2	S3	0	1	0	0	1	1
S3	S4	0	1	1	1	0	0
S4	S5	1	0	0	1	0	1
S5	S0	1	0	1	0	0	0

Table 17 - 0-5 Counter State Table

Karnaugh Maps:

Q_A^+ :

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

Table 18 - 0-5 Counter Karnaugh Map " Q_A^+ "

Q_B^+ :

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

Table 19 - 0-5 Counter Karnaugh Map " Q_B^+ "

Q_C^+ :

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

Table 20 - 0-5 Counter Karnaugh Map " Q_C^+ "

Boolean Expression:

$$Q_A^+ = A\bar{C} + BC$$

$$Q_B^+ = B\bar{C} + ABC$$

$$Q_C^+ = \bar{C}$$

Non-PLD:

Created in Non-PLD mode in order to test the circuit and ensure it is functioning as required prior to downloading to an FPGA, as it is easier to monitor the program and remedy any potential faults.

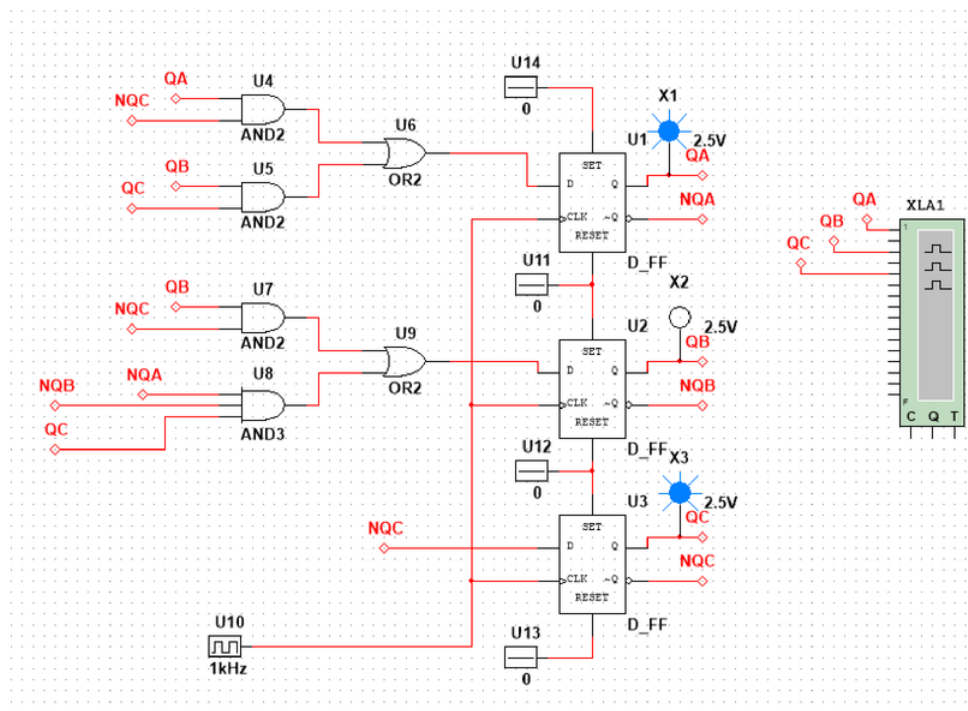


Figure 35 - 0-5 Counter Non-PLD CCT

By using a logic analyser an overview of the digital pulses can be seen, and as can be seen in Figure 36 the circuit counts from 0 up to 5 and returns to 0.

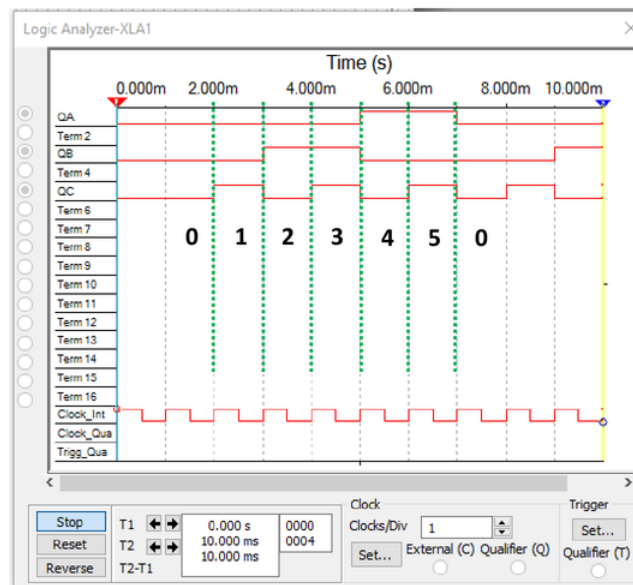


Figure 36 - 0-5 Counter Logic Analyser

PLD:

Once the circuit has been tested on non-PLD mode then the PLD circuit (Figure 37) can be downloaded onto the nexys digital DDR 4 board for physical testing. 4 LED's were used to represent the bit binary, and in order to test manually the centre button "BtnC" was input into the clock so that the count would progress whenever it was pressed.

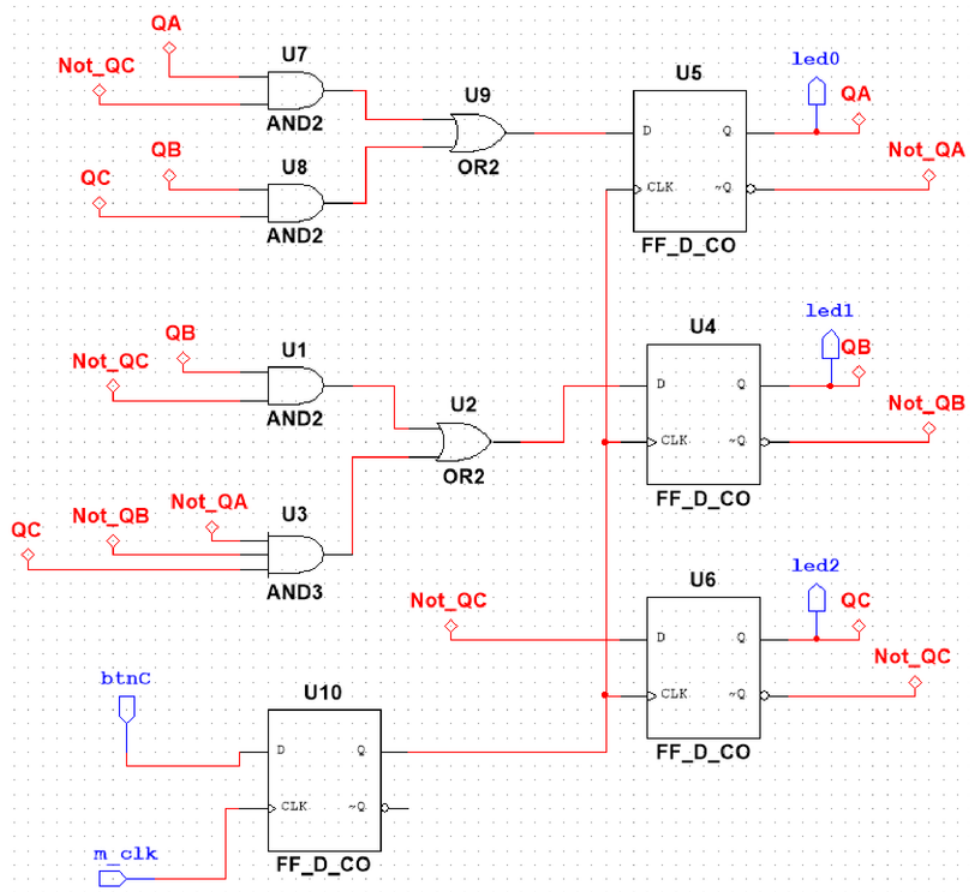


Figure 37- 0-5 Counter PLD CCT

0-5 Counter and Decoder combined:

Non-PLD:

To ensure the 0-5 counter and decoder (Figure 35 and Figure 8 respectively) were properly interfaced they were tested together. The only alteration made was that buffers were implemented on the outputs of the flip flops in order to convert these into the appropriate inputs for the decoder without changing the net names. Since the decoder was designed for a 4 bit binary input, and the 0-5 only outputs 3 bit binary, the most significant bit of the decoder "A" is set to 0 as it is not used. The Least significant bit of the counter (QC) is addressed to the least significant bit of the decoder(D), and so forth. The LEDs were implemented solely for testing purposes to clearly illustrate the current states of the flip flop outputs. This can all be seen in Figure 38.

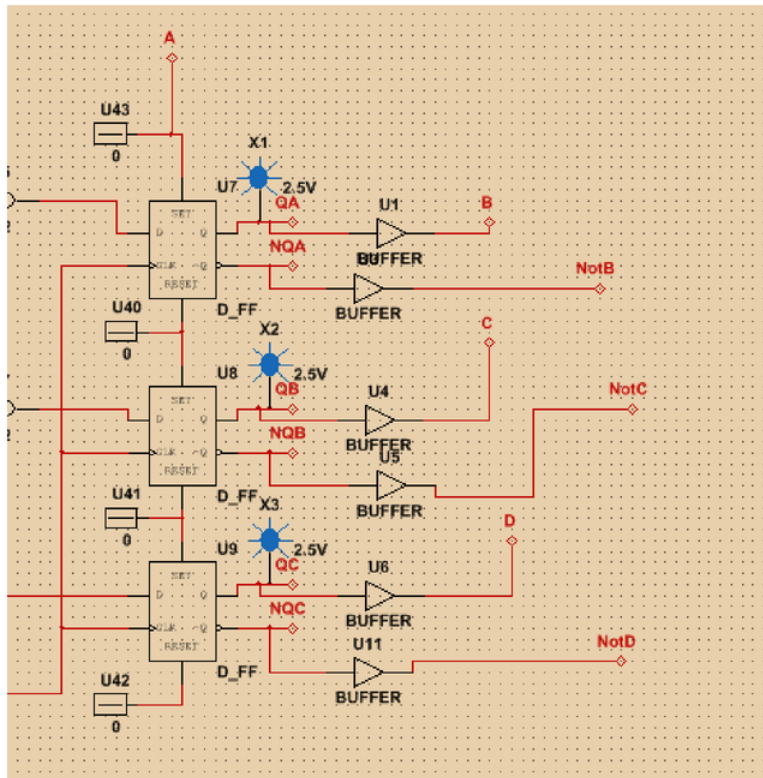


Figure 38 - 0-5 Counter and Decoder Interfacing



This circuit operated exactly like the decade and decoder combined seen in Figure 30 except the max count was 5 as can be seen in Figure 40.

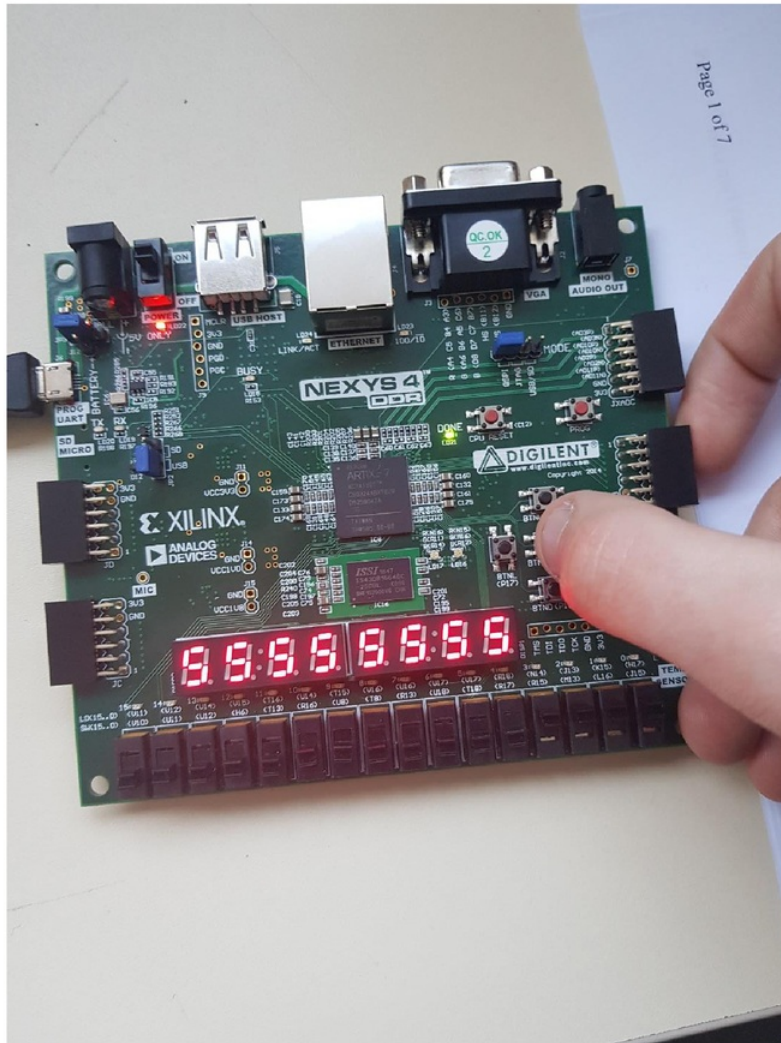


Figure 40 - 0-5 and decoder test max count (5)

Finished circuit

Non-PLD:

The circuits were combined in non-PLD to ensure correct functionality, the results can be seen in Figure 41 to Figure 45. The circuit counts from 0 up to the cap at 59, whereupon it resets the count to 0 and begins again.

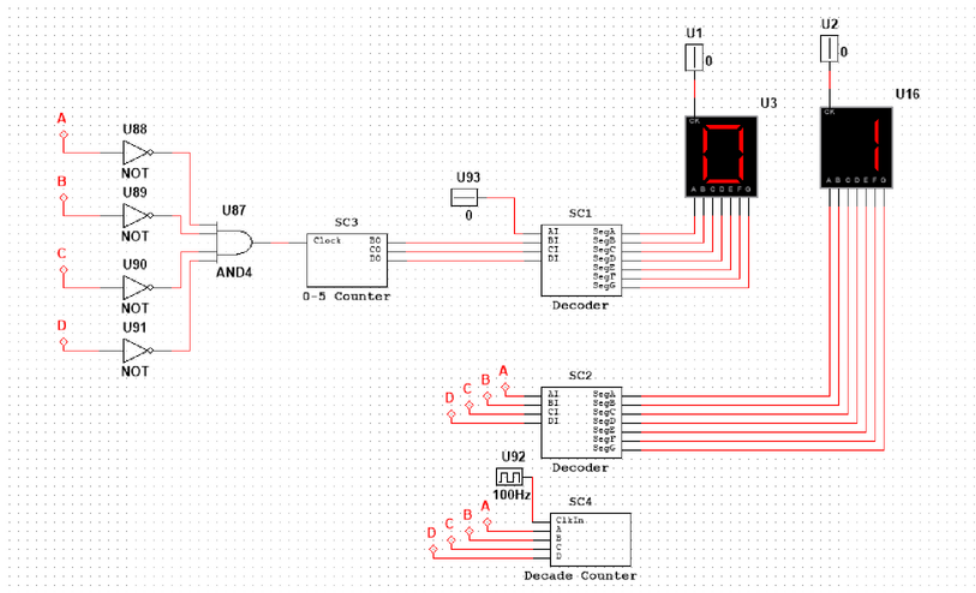


Figure 41 - Final CCT Non-PLD 01

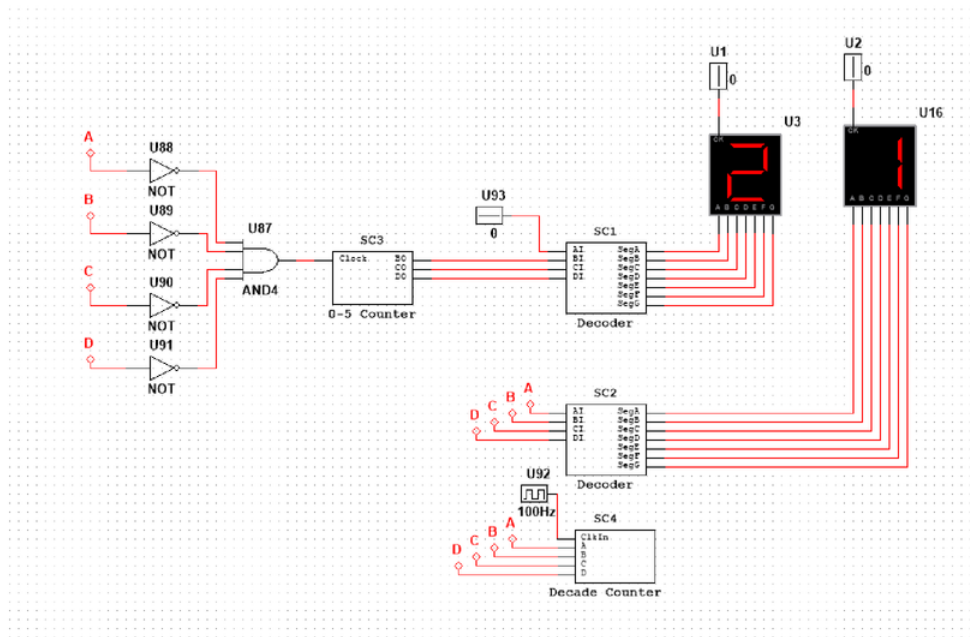


Figure 42 - Final CCT Non-PLD 21

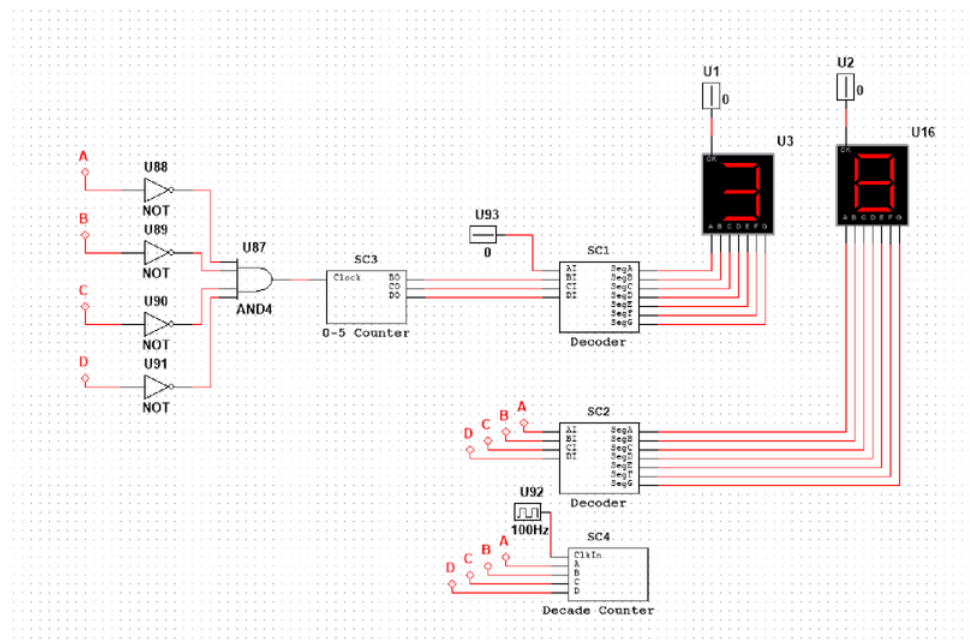


Figure 43- Final CCT Non-PLD 38

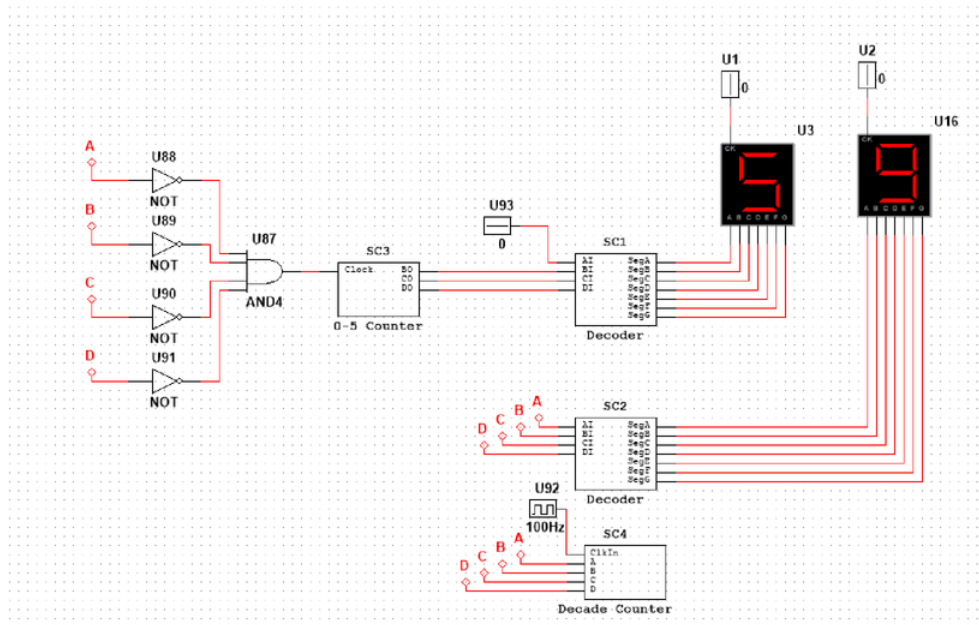


Figure 44 - Final CCT Non-PLD 59

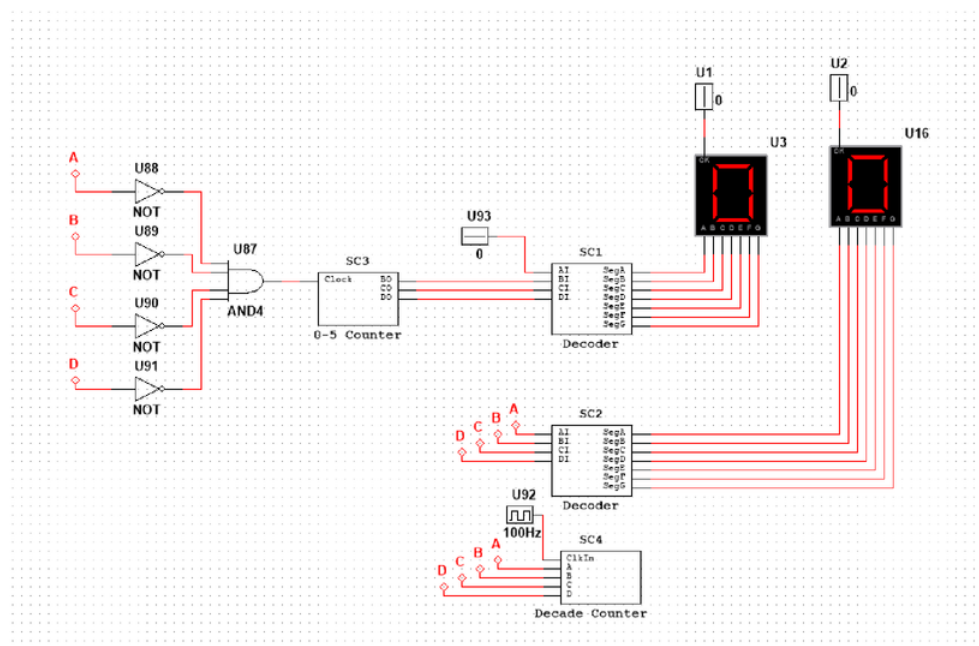


Figure 45 - Final CCT Non-PLD 00

Jac Williams

PLD:

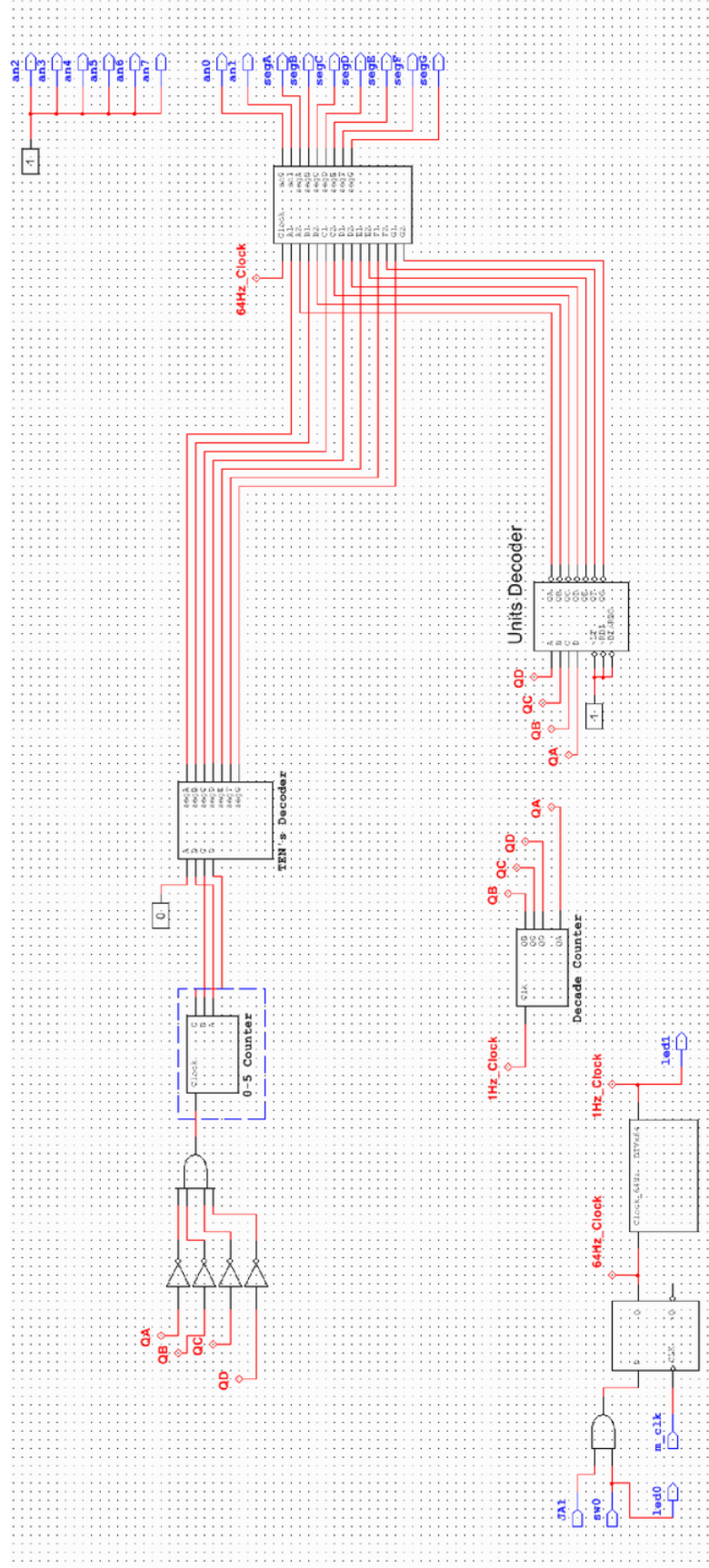


Figure 46 - Final CCT PLD

The Circuit depicted in Figure 46 is the final circuit which is a stopwatch that counts from 0-59 as stated by the brief.

The circuit is supplied by a 1Hz clock input, which is attained by supplying a 64Hz input from a physical function generator connected the board through the input "JA1". This 64Hz is conditioned into providing a 1Hz clock through the sub circuit displayed in Figure 46. Wherein the 64Hz clock is input into a series of T type flip flops, which essentially half the clock pulse with each successive flip-flop (going from 64, 32, 16, 8, 4, 2, and finally 1Hz). The reasoning for providing 64Hz from the function generator instead of 1Hz is that the 64Hz is used by the 7 segment multiplexer.

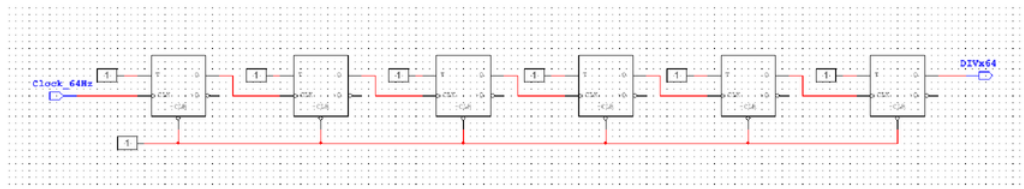


Figure 47 - Final CCT Clock Sub-CCT

In order to create a stop/start function the function generator input "JA1" was put into an and gate along with a physical switch "SW0" on the Board. This means that the only time the generated 64Hz signal will be input into the multisim clock is when the switch SW0 is made (an LED has been added for clarity, indicating when this switch is made or not).

Every second the unit count will increase by 1 (as the input clock is 1Hz i.e. once a second) inside the decade counter. The decade counter then outputs its 4 bit binary to the units decoder, which converts into a readable signal for the 7 seg display. In order to keep the program looking neat and not overcomplicating things this decoded signal is sent to the multiplexer, which in turn governs the 7 segment display outputs.

The 0-5 counter has a unit weighting of ten, meaning each count is a value of 10. This only increases when all of the outputs of the decade counter are 0 (initially in place of inverters the "NotA" outputs were used but this gave a false trigger every unit count), meaning that decade counter has counted up to 9 and has returned to 0. The output of the 0-5 counter is then sent to the TEN's decoder, with the weighting of the bits being matched and a digital constant of 0 is supplied to A (the M.S.B) as it is not used. This too is then sent to the 7 seg multiplexer.

Testing:

The circuit was tested for errors, and then downloaded onto the Nexys 4 board ready for testing.

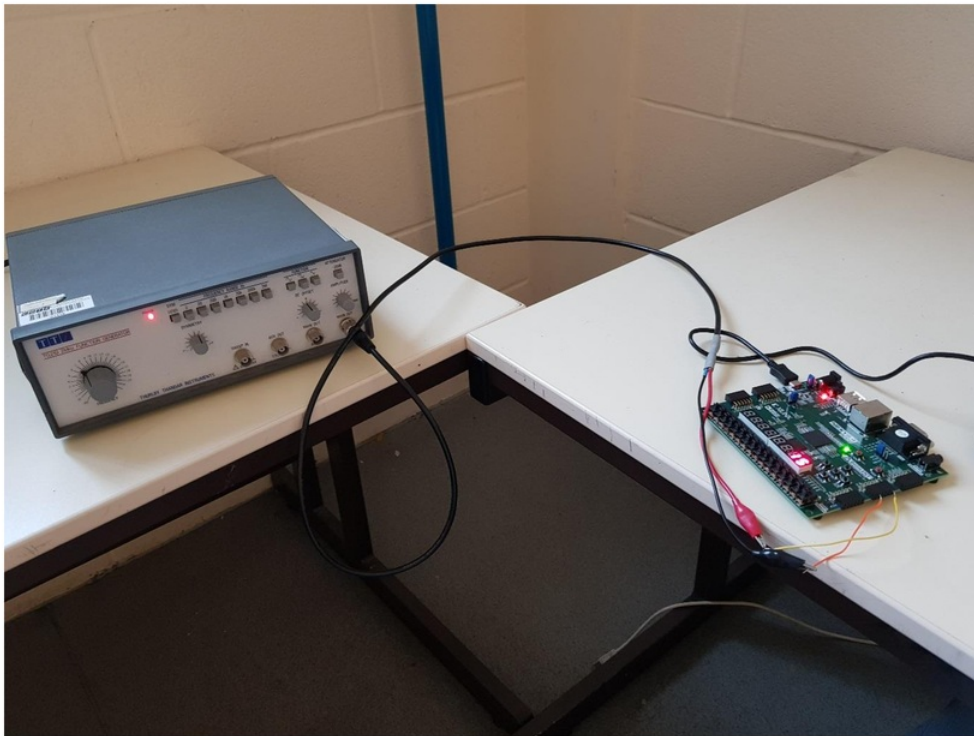


Figure 48 - Function Generator and Board

The function generator was set to a 64Hz square wave output, which was fed into the JA1 and GND terminals of the Nexys board, shown in Figure 46Figure 48.

Once the clock was supplied the board began counting on the 7 segment display, starting from 0 and counting up to 59 wherein it reset as evidenced in Figure 49 to Figure 55.

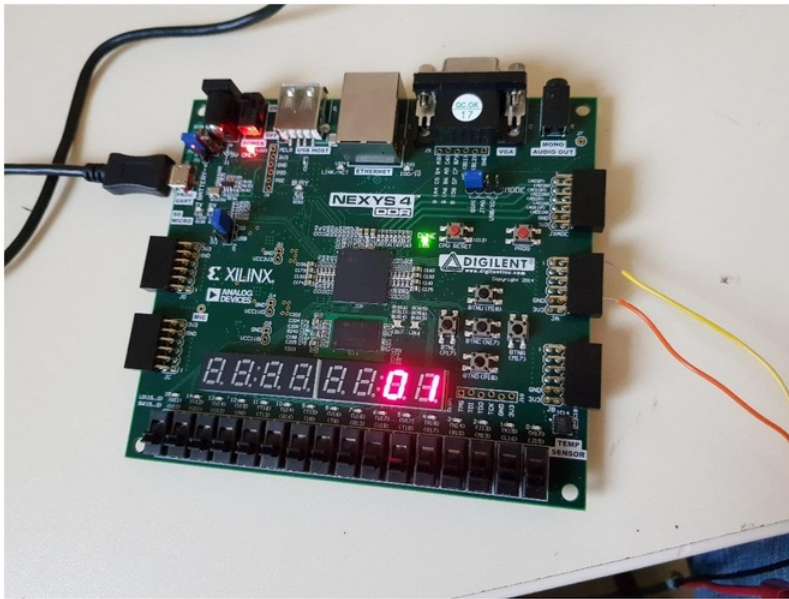


Figure 49 - Final CCT Board Count Testing 01

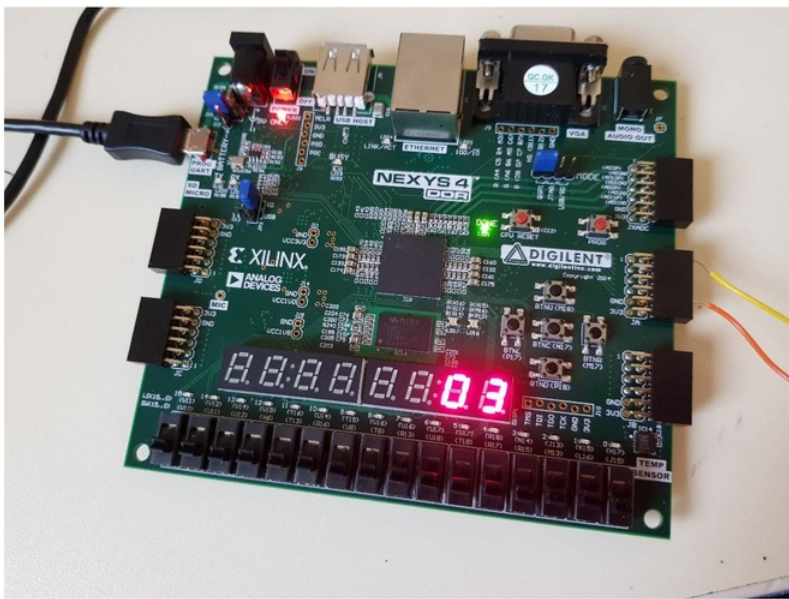


Figure 50 - Final CCT Board Count Testing 03

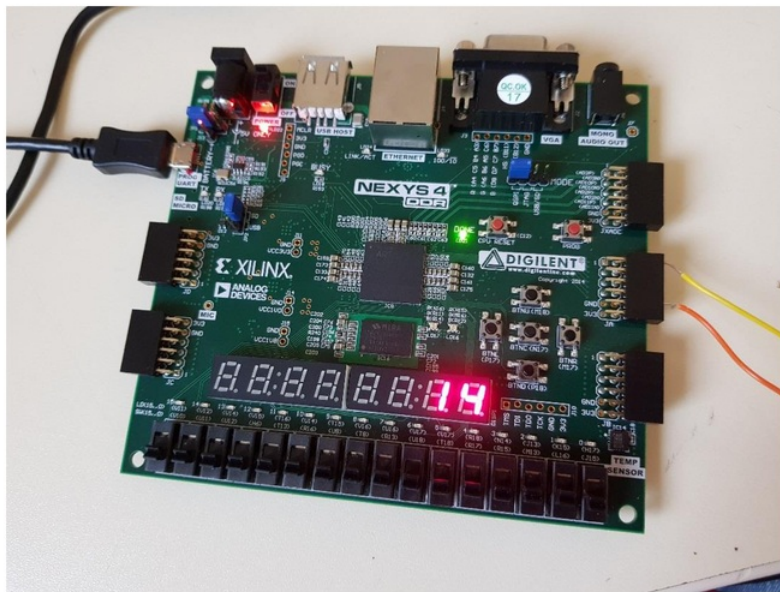


Figure 51 - Final CCT Board Count Testing 14

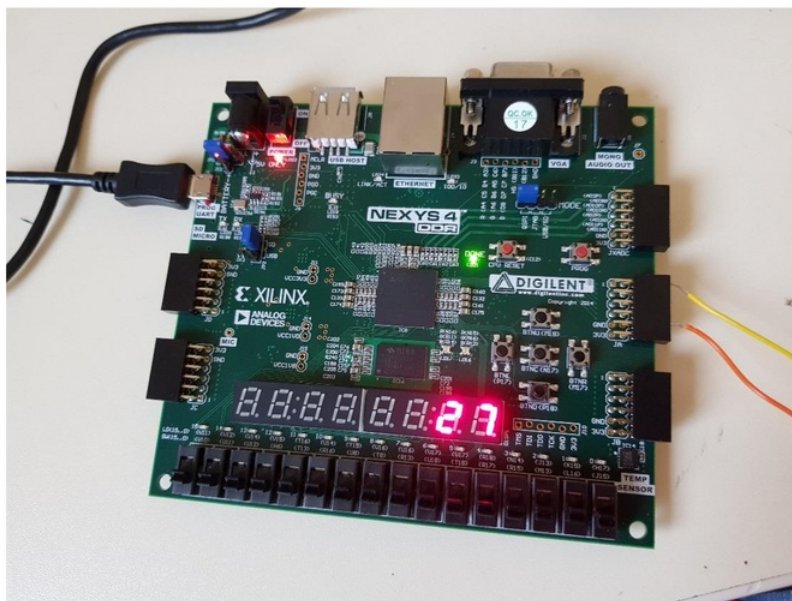


Figure 52 - Final CCT Board Count Testing 27

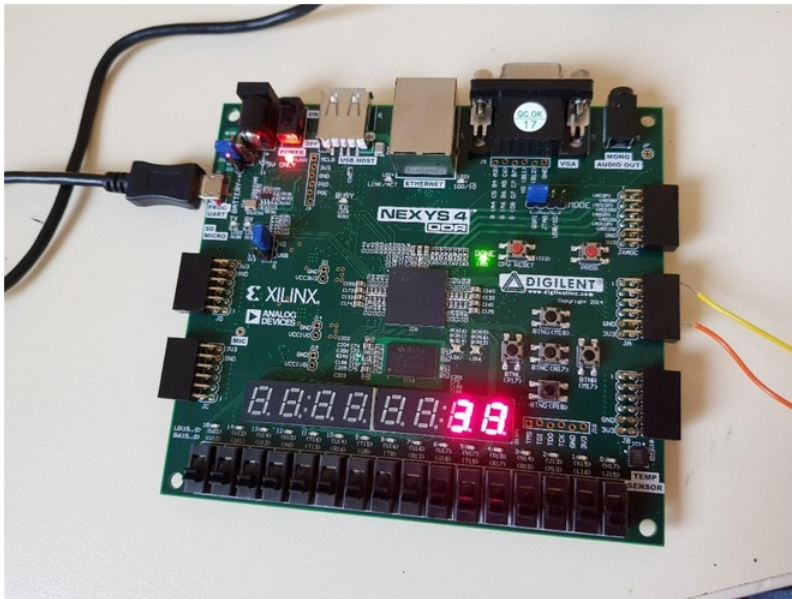


Figure 53 - Final CCT Board Count Testing 38

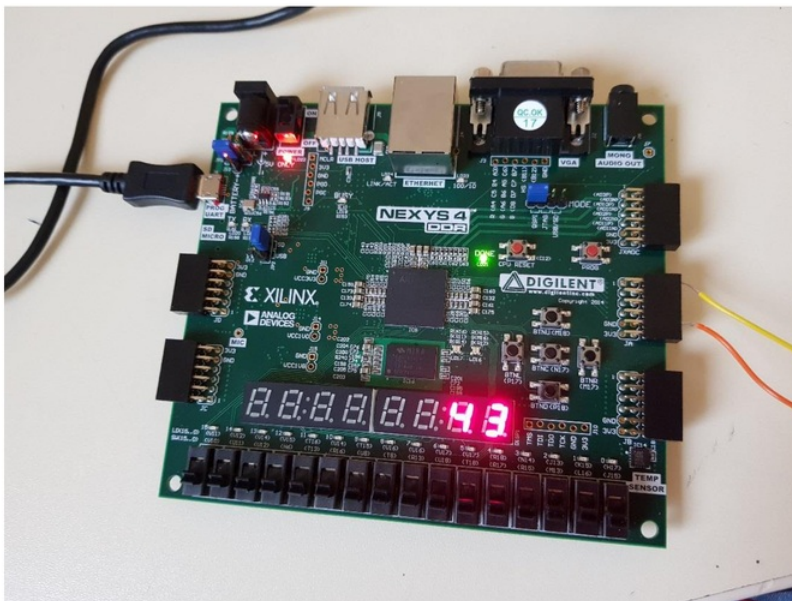


Figure 54 - Final CCT Board Count Testing 43

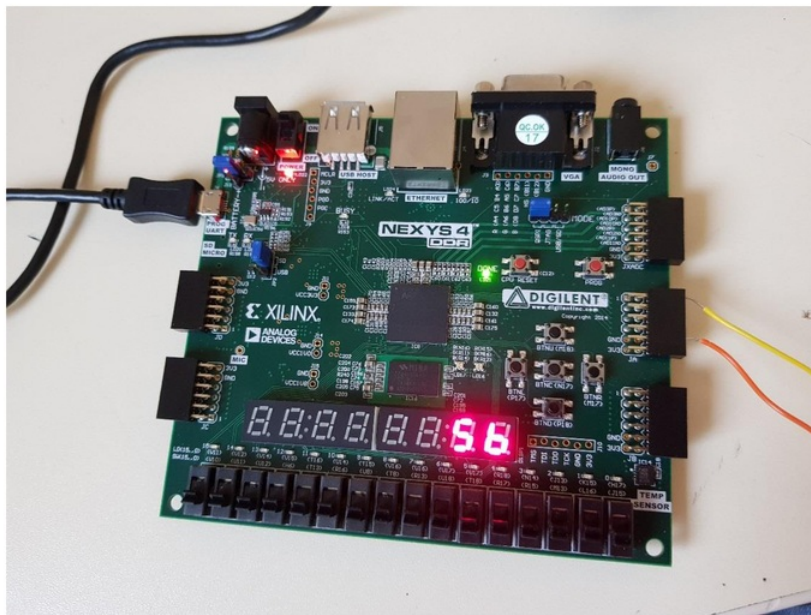


Figure 55 - Final CCT Board Count Testing 56

Conclusions:

By following tried and tested circuit design techniques mentioned in the research the final circuit functioned to specification laid out in the brief. The circuit was capable of a cyclical count from 0-59 whereupon it restarted the cycle at 0 again, and the count could be stopped and started at any point.

The circuit was initially simulated on multisim to ensure that the core design principles were sound, and then exported onto an FPGA for physical testing.

Creating the circuits in non-PLD initially proved to be a useful choice, as it allows us to hone in on any errors in the fundamental design of the circuit prior to exporting to a board where other factors could come into play and hamper our ability to fault-find.

Now that the circuit has been finalised, if mass production was the goal then they would be converted onto ASICs, as only a small portion of the FPGA was used and it would be wasteful to use it past the prototyping phase.

Works Cited

A.F.Kana, 2008. *Digital Logic Design*. [Online]

Available at: https://drive.google.com/file/d/0B_w12ctQbwTDT2oxaHNiQzk3Zk0/edit
[Accessed 25 April 2018].

Agarwal, T., 2017. *What Are The Different Types of Sequential Circuits?*. [Online]

Available at: <https://www.elprocus.com/different-types-of-sequential-circuits/>
[Accessed 20 April 2018].

allaboutcircuits, 18/10/2010. *finite-state-machines*. [Online]

Available at: <https://www.allaboutcircuits.com/textbook/digital/chpt-11/finite-state-machines/>
[Accessed 23 10 2017].

Hanson, K., 2015. *Digital Electronics Part III : Finite State Machines*. [Online]

Available at:
http://www.physics.wisc.edu/undergrads/courses/fall2017/623/lecture_notes/digital/digital-03.html
[Accessed 26 April 2018].

Holdsworth, B., 1991. *Digital Logic Design*. 2nd ed. Delhi: East West.

Kilts, S., 2007. *Advanced FPGA Design: Architecture, Implementation, and Optimization*. 2nd ed. Hoboken: John Wiley & Sons.

Storey, N., 2009. *Electronics: A Systems Approach*. 4th ed. Gosport: Pearson Education Limited.

Weatherspoon, H., 2013. *State and Finite State Machines*. [Online]

Available at: <https://www.cs.cornell.edu/courses/cs3410/2013sp/lecture/04-state-w-g.pdf>
[Accessed 25 April 2018].

Zeidman, B., 2006. All about FPGAs. *EE Times*, 1(1), p. 3.

PSDE 4004 Assignment 2

GRADEMARK REPORT

FINAL GRADE

GENERAL COMMENTS

Instructor

90/100

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

PAGE 22

PAGE 23

PAGE 24

PAGE 25

PAGE 26

PAGE 27

PAGE 28

PAGE 29

PAGE 30

PAGE 31

PAGE 32

PAGE 33

PAGE 34

PAGE 35

PAGE 36

PAGE 37

PAGE 38

PAGE 39

PAGE 40

PAGE 41

PAGE 42

PAGE 43

PAGE 44

PAGE 45

PAGE 46

PAGE 47

PAGE 48

PAGE 49

PAGE 50

PAGE 51

PAGE 52

PAGE 53

PAGE 54

PAGE 55

PAGE 56

PAGE 57

PAGE 58

PAGE 59

PAGE 60

GRADING FORM: CSG-RUBIC

JAC WILLIAMS

90

RESEARCH-INFORMED LITERATURE



There was evidence of extensive further reading and research from highly credible and reliable sources. Your interpretation and understanding of the additional research was excellent 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.

9

KNOWLEDGE AND UNDERSTANDING



The knowledge and understanding of the subject demonstrated by this assignment was generally excellent. There were very few gaps in your knowledge of the basic principles and the higher level concepts were very well explained. Your explanation of the design process and calculations was also excellent and it did extend your knowledge well beyond that contained in the course notes.

27

ANALYSIS



Your analysis of the task was excellent, both in class and as shown in this assignment. You have demonstrated very good analytical skills and your approach to the task was very logical. You carried out a detailed evaluation of the results and included an excellent conclusion relating your practical results to accepted theory.

18

PRACTICAL APPLICATION AND DEP.



Your application of practical skills for this task was generally excellent. You used the resources available in MultiSim appropriately when designing the circuit and you used appropriate options for testing and evaluating the circuit performance. You managed to design a very efficient circuit that functioned well within the specification and achieved with very little assistance. Your results were very detailed, well presented and allowed for an excellent conclusion to be attempted.

27

SKILLS FOR PROFESSIONAL PRAC.



In this assignment you have demonstrated skills in

9

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