

Assignment 1 - Josh Coles

by Joshua Coles

FILE	8468_JOSHUA_COLES_ASSIGNMENT_1_-_JOSH_COLES_155126_1829976634.DOCX (8.01M)		
TIME SUBMITTED	12-FEB-2018 02:40PM (UTC+0000)	WORD COUNT	4877
SUBMISSION ID	83097384	CHARACTER COUNT	36120

BSc (Hons) Electrical & Electronic Engineering
PSDE6018: Electronic Production & Test

Joshua Coles
Assignment 1

Contents Page

Table of Figures.....	3
1. Introduction.....	5
2. Research.....	6
3. Design of Assignment Requirements & Constraints.....	11
3.1 Development of CAR 90	13
3.2 Development of CAR 150	15
3.3 Development of 4-Bit Hybrid CAR	17
3.4 Development of 6-Bit Hybrid CAR	19
4. Analysis of NI MultiSim Results.....	20
5. Analysis of results using hardware descriptive language FPGA technology	25
6. Analysis of PCB Design	31
7. Practical PCB build and testing of prototype on prototyping board	36
8. Conclusions & Recommendations	38
6. Citation List.....	39
7. Appendix.....	40

Table of Figures

Figure 1 - Vishay Foil Resistor Soldering Profile (Vishay, 2014).....	8
Figure 2 - Mass Differential Example (Surface Mount Process, 2015).....	9
Figure 3 - CAR Register	11
Figure 4 - EXOR CAR 90.....	13
Figure 5 - MultiSim CAR90 Circuit.....	14
Figure 6 - MultiSim CAR90 Results.....	14
Figure 7 - EXOR CAR 150.....	15
Figure 8 - CAR150 Expected Results	15
Figure 9 - MultiSim CAR 150 Circuit.....	16
Figure 10 - MultiSim CAR 150 Results.....	16
Figure 11 - Hybrid CAR Design.....	17
Figure 12 - MultiSim Hybrid CAR.....	18
Figure 13 - MultiSim Hybrid CAR Results	18
Figure 14 - 6-Bit Hybrid CAR.....	19
Figure 15 - 6-bit Hybrid CAR Results.....	19
Figure 16 - Testing Circuit.....	20
Figure 17 - XOR Results.....	20
Figure 18 - Fault Test 1 Circuit.....	21
Figure 19 - Fault Test 1 Results	22
Figure 20 - Fault Test 2 Circuit.....	22
Figure 21 - Fault Test 2 Results	23
Figure 22 - Fault Test 3 Circuit.....	23
Figure 23 - Fault Test 3 Results	24
Figure 24 - 4-bit CAR 90 Xilinx.....	25
Figure 25 - 4-bit CAR 150 Xilinx.....	25
Figure 26 - 4-bit Hybrid CAR Xilinx.....	25
Figure 27 - Test Circuit Results 1	26
Figure 28 - Test Circuit Results 2	26
Figure 29 - 6-bit Hybrid CAR Test Circuit.....	27
Figure 30 - Hybrid CAR Wolfram Logic.....	27
Figure 31 - Fault 1 Diagram.....	28
Figure 32 - Fault 1 Circuit	28
Figure 33 - Fault 2 Diagram.....	29
Figure 34 - Fault 2 Circuit	29
Figure 35 - Fault 3 Diagram.....	30
Figure 36 - Fault 3 Circuit	30
Figure 37 - Eagle Circuit Design	31
Figure 38 - Eagle Circuit Print.....	31
Figure 39 - Solder Stop Mask Solder Side.....	32
Figure 40 - Solder Stop File Component Side	32
Figure 41 - Top Copper File	33
Figure 42 - Bottom Copper Layer.....	33
Figure 43 - Board Outline File.....	34
Figure 44 - Drill Hole File	34
Figure 45 - Complete Manufacturing Design	35
Figure 46 - Manufactured PCB Substrate	36

Figure 47 - PCB Test	36
Figure 48 - PCB Fault Test	37
Table 1 - CAR State Equations.....	12
Table 2 - CAR Outputs Values	12
Table 3 - CAR90 Expected Results	13
Table 4 - Hybrid CAR Output.....	17
Table 5 - MultiSim Results	21
Table 6 - 6-bit Hybrid CAR Xilinx Results	26
Table 7 - Test Board Results.....	27

1. Introduction

The aims of this assignment are to understand the process of designing, constructing and testing PCBs in the industrial environment. This will be achieved by researching the design process of PCBs and the relevant soldering techniques. Research on the testing of the PCBs will cover the automatic optical inspection and the automatic X-Ray inspection processes. Testing methods will be designed, simulated and analysed using the Multisim and LabVIEW software. A PCB will be designed and manufactured using CAD/CAM software and a testing device created to prove the operation of the device. Analysis will be carried out on all results achieved and a conclusion written to summarise the report.

*LabVIEW not used
and intro is a little short
on procedure.*

2. Research

Surface Mount Technology

Surface mount technology (SMT), is made up of surface mount devices (SMD) that are soldered onto a board (Poole, 2012). Before SMT, mass produced electronic circuit boards were made of the traditional leaded electronic components. These components had wires attached that needed preforming before mechanical construction on the board could begin. Automating the soldering process had many issues as the formed cables often missed the intended holes and slowed the production down drastically.

SMD are devices that were made to be directly soldered onto the board without the use of wires. The mechanical construction of electronic boards was made much simpler with SMD and thus the technology was adopted very quickly. Manufacturing costs and times were reduced significantly and major costing was saved. SMD are made up of transistors, diodes, integrated circuits and passive SMDs; typically made up of standardized components such as resistors, capacitors, coils and crystals.

*Images
would
have
helped*

PCB Process

To begin the process of making a printed circuit board (PCB) a design needs to be put together. All projects require a unique PCB so there are no standard models. Designers use computer aided design (CAD) software to design the layout of the PCB. In the design, hole locations for connections and soldering points are marked and then negatively printed on a clear plastic sheet.

The board, also known as the substrate, starts with a base of woven fibre. The woven fibre goes through a process where epoxy resin is either impregnated or sprayed onto the surface. The fibre-resin mix is then fed through a rolling process until the desired thickness of the board is achieved. The flattened fibre is then sent through an oven where the mix is made semi-cured. The mix is then cut into large sheets.

The large sheets are then stacked alternatively with adhesive backed copper and placed in a pressure furnace to complete curing. The cured panels are then placed in a CNC drilling machine, the holes are drilled from the information from the CAD software. Holes that are to be used from conduction are lined with copper, non-conducting holes are plugged to prevent plating.

The pattern of the circuit board can be etched through an 'additive' process where copper plating is applied in the desired area of the board. The surface of the board is first degreased. Positive photoresist material is applied to the entire board, ultraviolet (UV) light is then exposed to the desired circuit path; photoresist is a polymer that becomes soluble when exposed to UV light. The irradiated photoresist is washed off through an alkaline developer, the irradiated area now exposes the copper layer below the original layer of substrate. The panels are then electroplated with copper. The foil on the surface of the panels act as the cathode, any exposed tin is plated with a copper layer of about 0.001 to 0.002 inches. The remaining photoresist cannot act as a cathode and these areas aren't plated. To prevent oxidizing, the copper plating is coated with a protective layer such as tin-lead etc. An acid bath removes the remaining photoresist to expose the machined substrate. Any external connectors are plated on using a similar method, the board then passes through an oven to melt the porous tin-lead plating into a shiny finish that cannot oxidize.

To protect the board from damage, each panel is sealed with epoxy resin. Any information or markings are applied to the board at this point. The boards are then cut into smaller, individual board and edges smoothed. The boards are now ready for the components.

For SMT, the boards first pass through a solder pasting machine that applies a small dab of solder at each connection point. Small components can be mounted using a 'chip shooter' which shoots components into their positions. Larger components are robotically placed on the board. Any components that are oddly shaped or sized may need manual placing on the board. The boards are then passed through another oven where the solder dabs 'reflow', soldering the components into their final positions. Residual flux is cleaned with either water or solvents.

The board is now completed, unless it is immediately used the board will be sealed in plastic bags for future use or shipping (Made How, 2007).

Reflow Soldering

'Reflow' soldering was mentioned in the construction of the PCB board. Reflow soldering is the most common method of attaching SMD to PCBs. The idea is to place components on solder dabs and then heat the solder to its liquefying temperature without damaging the board, solder or components.

A suitable reflow oven is required for the process. The oven must work fast enough to handle the production rate of the pick and place robots. An equation for the speed can be seen below:

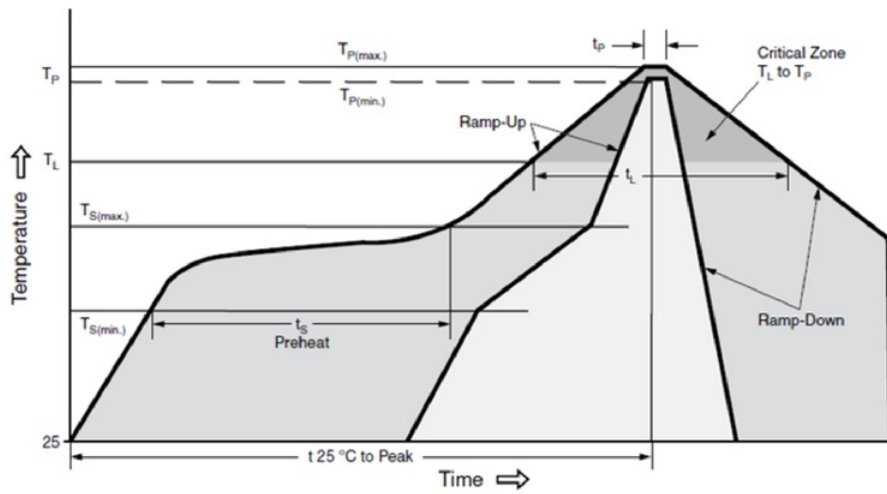
$$\text{Minimum Line Speed} = \frac{\text{Boards per minute} \times \text{Length per board}}{\text{Load Factor}}$$

The load factor is the space between the boards in the process, usually specified by the manufacturer of the reflow oven. For a reflow oven to be appropriate for the process the process speed; defined below, of the oven must be greater than the minimum line speed:

$$\text{Process Speed} = \frac{\text{Oven chamber heated length}}{\text{Process dwell time}}$$

For the oven to work as required a reflow profile needs programming in. To create a reflow profile thermocouples are used to record the range of temperature across a sample assembly. The more thermocouples used the better profile can be developed. There are four stages to a reflow soldering profile: Preheating, Soaking, Reflow and Cooling. Figure 1 shows the reflow soldering profile to be used on the Vishay foil resistors. The graph shows an initial rise in temperature; the preheating, a period of held temperature, the soaking, a period of ramp up to peak temperature; the reflow, and a ramp down on temperature; cooling.

Reflow Soldering Profiles



Classification of Reflow Profiles		
Profile Feature	Sn - Pb Eutectic Assembly	Lead (Pb)-Free Assembly (e.g. SnAgCu)
Average ramp-up rate ($T_{S(max.)}$ to T_P)	3 °C/s maximum	
Preheat		
• Temperature minimum ($T_{S(min.)}$)	100°C	150°C
• Temperature maximum ($T_{S(max.)}$)	150°C	200°C
• Time ($T_{S(min.)}$ to $T_{S(max.)}$) (t_s)	60 s to 120 s	60 s to 180 s
Time maintained above		
• Temperature minimum (T_L)	183°C	217
• Time (T_L)	60 s to 150 s	60 s to 150 s
Minimum peak temperature ($T_{P(min.)}$)	215°C	235°C
Recommended peak temperature (T_P)	235°C	250°C
Maximum peak temperature ($T_{P(max.)}$)	260°C	260°C
Time within 5°C of actual peak temperature (t_p)	10 s to 40 s	20 s to 40 s
Ramp-down rate	6°C/s maximum	6°C/s maximum
Time 25°C to peak temperature	6 min maximum	8 min maximum

Figure 1 - Vishay Foil Resistor Soldering Profile (Vishay, 2014)

Preheating: During this phase the board, PCBs and solder are heated to a specific soak temperature gradually; usually about 2°C a second depending on the solder. The temperature is gradually risen to reduce the expansion stress.

Soaking: During this phase the board is held at a desired temperature to ensure all components and solder are heated equally. The soak phase lasts between 1 to 2 minutes depending on the 'mass differential' of the board. The mass differential can be simply defined as the difference in mass between components spread across the board as shown in Figure 2.

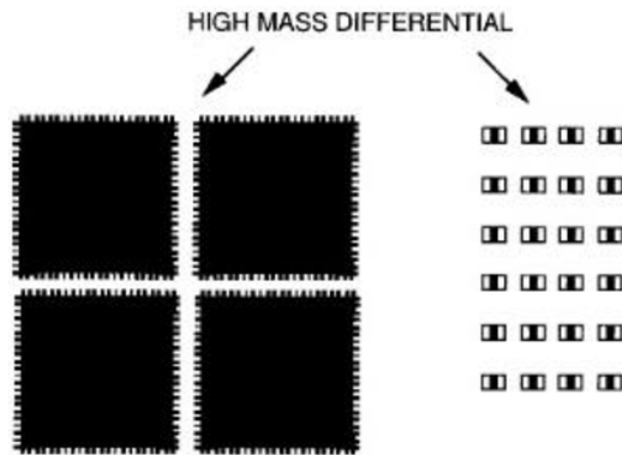


Figure 2 - Mass Differential Example (Surface Mount Process, 2015)

Reflow: The temperature in the reflow oven is increased above the melting point of the solder causing it to liquefy. To ensure the components have been exposed to the liquid solder the temperature is held above melting point for thirty to sixty seconds. This time shouldn't be exceeded, exceeding this time could cause damage to the components or substrate. An environment lacking oxygen is required during the reflow stage, reflowing in an oxygen rich environment will create oxides inside the solder which causes a non-solderable surface. Nitrogen is typically used to purge the oxygen from the reflow area.

Cooling: The rate of cooling must be controlled to prevent deformities and allow for solid fixing of components. The recommended cooling rate is 3°C a second. Once the board has return to atmospheric temperature the reflow soldering process has finished.

(Surface Mount Process, 2015)

Automatic Optical Inspection

After the introduction of SMT and the reduction in size of components, electronic boards began to contain more components and therefore more solder joints. Manually inspecting each board became a near impossible task. Automatic optical inspection (AOI) is a method of inspecting circuit boards using visual methods.

A board is passed through a device that illuminates the board with multiple light sources. One or more high definition camera are used to create a picture of the board. The board is then inspected using various comparison methods, the methods are template matching, pattern matching and statistical pattern matching. Template matching compares the sample photo to a known standard photo of a perfectly working board, any issues or defects are highlighted by the software. Pattern matching uses information stored from faulty and healthy boards and compares the sample photo to both. Statistical pattern matching is very similar to pattern matching, however this method uses a statistical based method of addressing issues, by storing images from several failed boards, acceptable deviations are allowed through (Electronic Notes, 2011).

Automatic X-Ray Inspection

PCBs now use a soldering method called ball grid array (BGA) and chip scale packages (CSP). Using these methods the solder joints are hidden and cannot be visually inspected using AOI. Automatic X-Ray inspection (AXI) is similar to AOI, however AXI uses X-Rays instead of visible light as source.

AXI systems are able to 'see' through the components and analyse the components and solder joints. In more modern AXI systems as well as 2D, 3D models are available for more complicated PCBs. Not only can the AXI system 'see' the solder joints, it is also able to see through the solder and inspect for any voids in the solder that would disrupt the performance of the PCB. Solder thickness, joint size and profile is inspected on the board and the information fed back to the soldering process to alter and improve the results. Any discrepancies or damage is made visible through the use of AXI and an image processing software compares the sample model to a working model (Electronic Notes, 2011).

*Research
of ATE
for
production
industry
should
include
diagrams
for
support.*

3. Design of Assignment Requirements & Constraints

The aim of this assignment is to produce a 6-bit hybrid CAR using rule 90 and rule 150. The objective of the hybrid CAR is for it to be a pseudo-random pattern generator that will be used to test circuits. The pattern generator must have no repeats in the sequence and must reach maximum length.

Maximum length is achieved if the circuit reaches a count of $2^n - 1$ before cycle repetition; where 'n' refers to the number of 'D' flip-flops in the circuit. The pseudo-random pattern must have no repeats in a single cycle; this is because faulty components or connections may result in repeated numbers and therefore repeats will be treated as faults. A seed value is initially loaded into the circuit to begin the sequence.

Figure 3 shows a cellular automata register (CAR). A CAR is a register that comprises of flip-flops whose next state is determined by previous state values and the values of neighbouring flip-flops. The output of the 'D' type flip-flops in Figure 3 depend upon its present value as well as the neighbouring flip-flops.

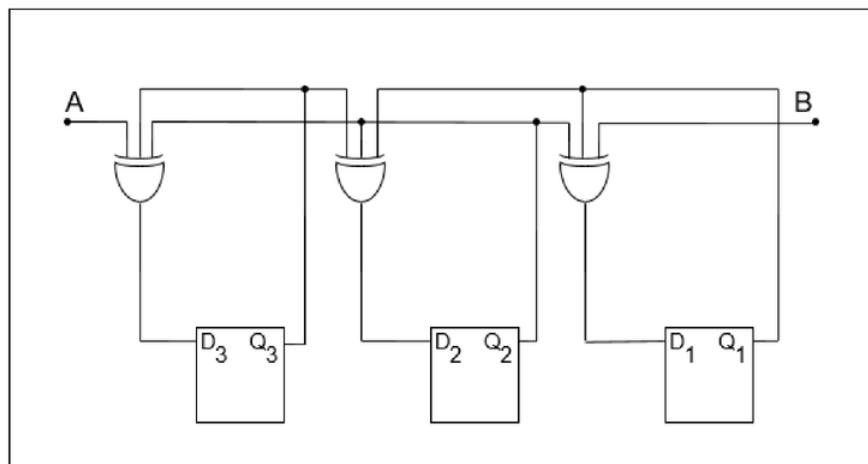


Figure 3 - CAR Register

In Figure 3, points 'A' and 'B' can be either be connected together. In this scenario the CAR is known as a 'Cyclic CAR'. Points 'A' and 'B' can be shorted to ground, in this scenario the CAR is known as a 'Null Boundary CAR'. When using the 'Null Boundary CAR' and initialising to '1' the state equations and output values of the CAR can be seen below in Table 1 and Table 2.

State Equations			
Q3+	=	0 + Q3 + Q2	= Q2 + Q3
Q2+	=	Q3 + Q2 + Q1	= Q1 + Q2 + Q3
Q1+	=	Q2 + Q1 + 0	= Q1 + Q2

Table 1 - CAR State Equations

Output Values			
Q3	Q2	Q1	Value
0	0	1	1
0	1	1	3
1	0	0	4
1	1	0	6
0	0	1	1

Table 2 - CAR Outputs Values

It can be seen from the output values in Table 2 that the maximal length is not achieved, this is because of the way the adjacent cells are connected. The connections have been defined by Wolframs Rule 90 by combining the 3-bit logic through an EXOR.

3.1 Development of CAR 90

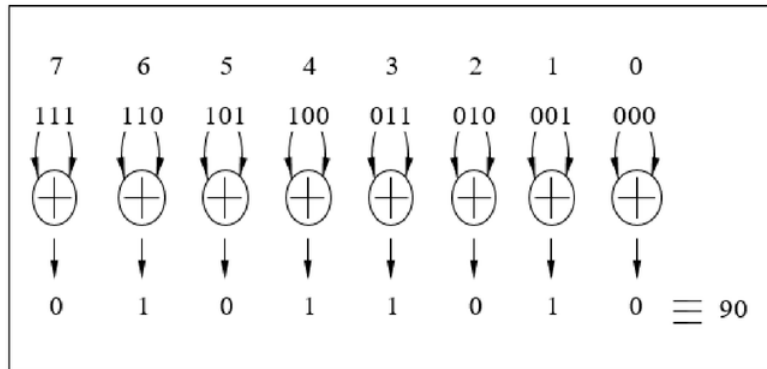


Figure 4 - EXOR CAR 90

Figure 4 shows ‘Rule 90’ in operation where the next state of each ‘D’ type flip-flop; Q_t^+ is defined by combining the outputs of either side flip-flop; Q_{t-1} and Q_{t+1} , through an EXOR. Therefore, Rule 90 can be surmised as:

$$\text{Rule 90: } Q_t^+ = Q_{t-1} \oplus Q_{t+1}$$

The expected results of the CAR90 circuit can be seen in Table 3. The MultiSim circuit for the CAR90 can be seen in Figure 5 and the achieved results can be seen in Figure 6.

A	B	C	D	Binary
0	0	0	1	1
0	0	1	0	2
0	1	0	1	5
1	0	0	0	8
0	1	0	0	4
1	0	1	0	10
0	0	0	1	1

Table 3 - CAR90 Expected Results

Explanation of the theory behind the CAR rules are explained well.

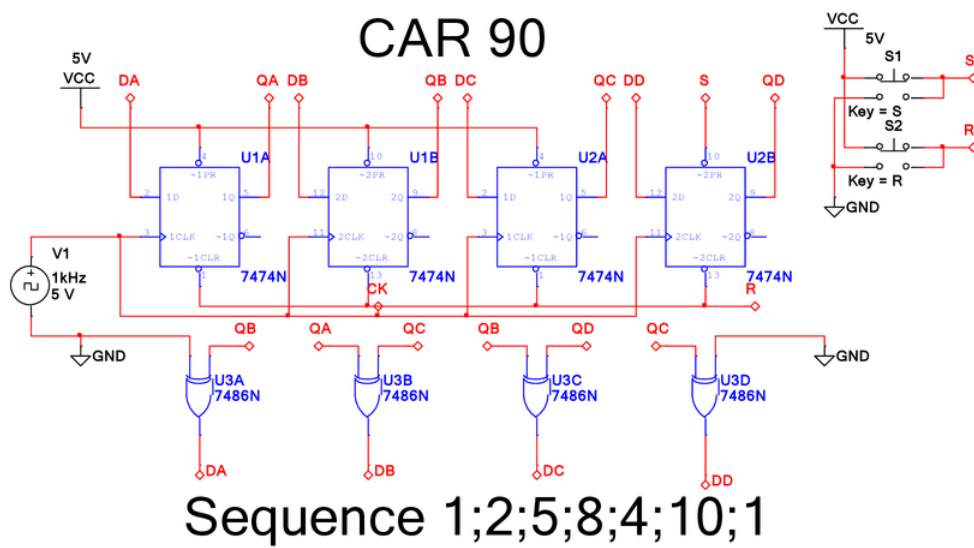


Figure 5 - MultiSim CAR90 Circuit

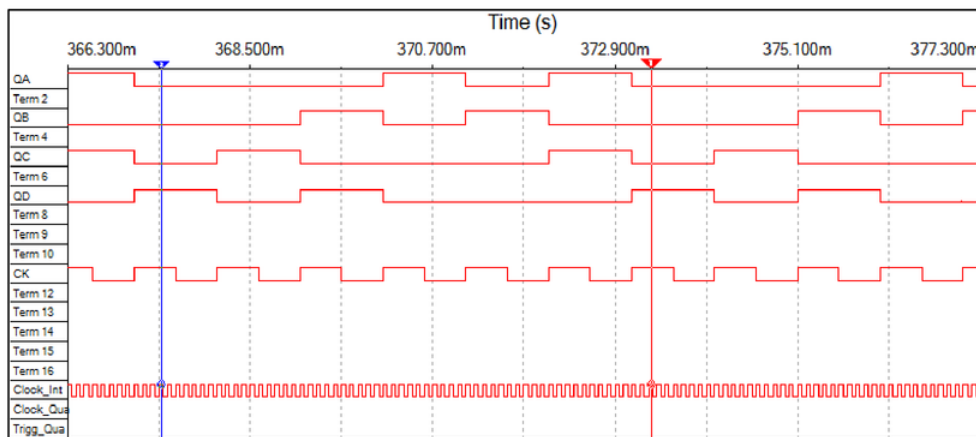


Figure 6 - MultiSim CAR90 Results

From the above figures it can be seen that the maximum length has not been achieved. Expected results were achieved in the MultiSim circuit. This circuit will be used to test later designs from Xilinx and practical circuits.

3.2 Development of CAR 150

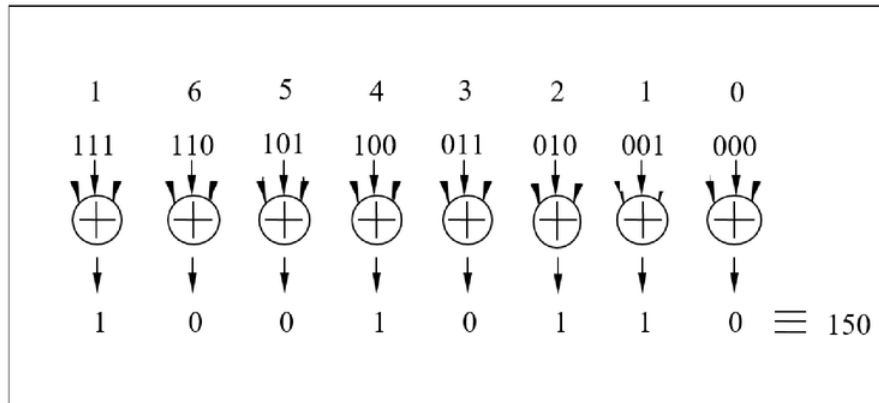


Figure 7 - EXOR CAR 150

Figure 7 shows the 'Rule 150' in operation where the next state of each 'D' type flip-flop; Q_t^+ is defined by combining the outputs of either side flip-flop; Q_{t-1} and Q_{t+1} and its own state, through an EXOR. Therefore, Rule 150 can be summarised as:

$$\text{Rule 150: } Q_t^+ = Q_{t-1} \oplus Q_t \oplus Q_{t+1}$$

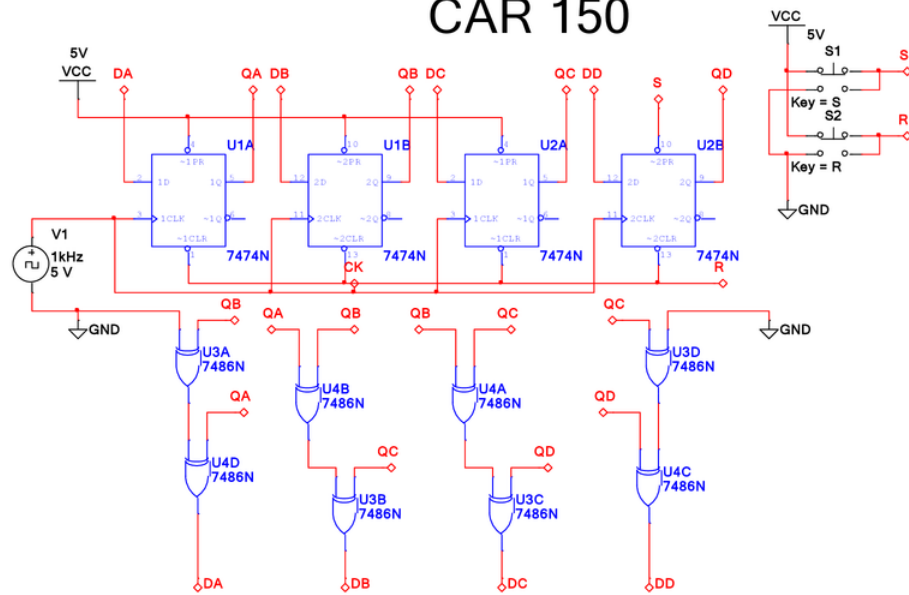
The expected results of this CAR can be seen in Figure 8. The CAR150 circuit can be seen in Figure 9 and the results of the circuit can be seen in Figure 10.

A	B	C	D	Binary
0	0	0	1	1
0	0	1	1	3
0	1	0	0	4
1	1	1	0	14
0	1	0	1	5
1	1	0	1	13
0	0	0	1	1

Figure 8 – CAR150 Expected Results

$$Q_t^+ = Q_{t-1} \oplus Q_t \oplus Q_{t+1}$$

CAR 150



Sequence 1;3;4;14;5;13;1

Figure 9 – MultiSim CAR 150 Circuit

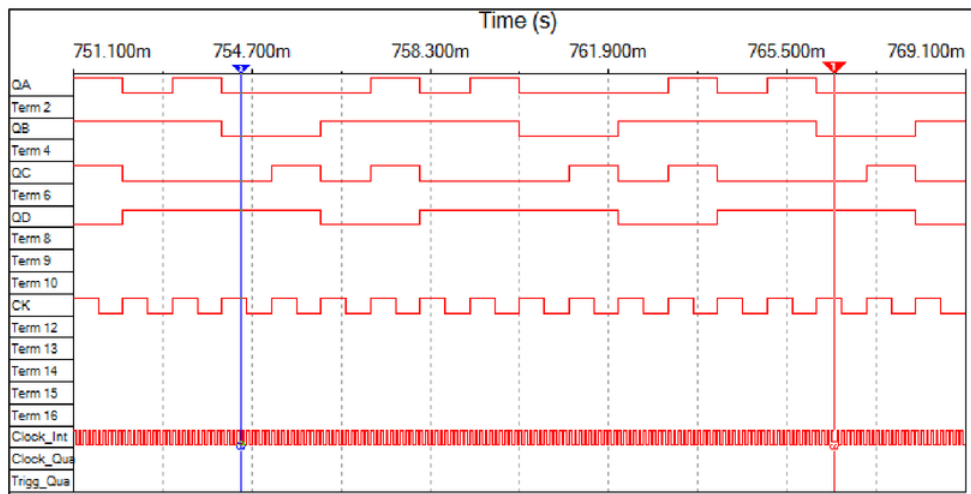


Figure 10 – MultiSim CAR 150 Results

From the above figures it can be seen that again, the maximum length has not been achieved. Expected results were achieved in the MultiSim circuit. This circuit will be used to test later designs from Xilinx and practical circuits. By developing a hybrid circuit between the CAR 90 and CAR 150 the desired maximum length can be achieved.

3.3 Development of 4-Bit Hybrid CAR

In order to achieve maximum length, a hybrid car must be constructed using both Rule 90 and Rule 150. This hybrid will produce a pseudo-random CAR that will alternate between the two rules. The basic design of this is shown in Figure 11. The expected output of the Hybrid CAR can be seen in Table 4.

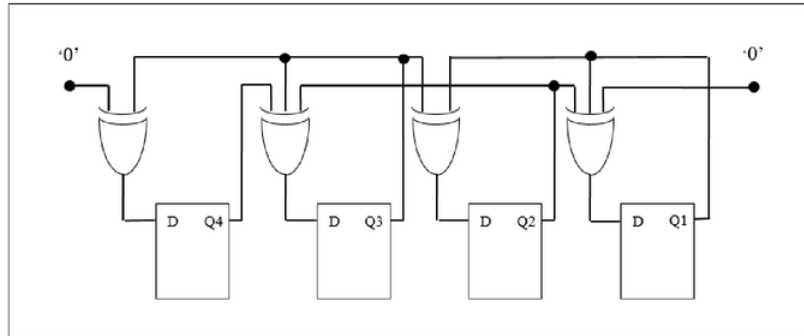
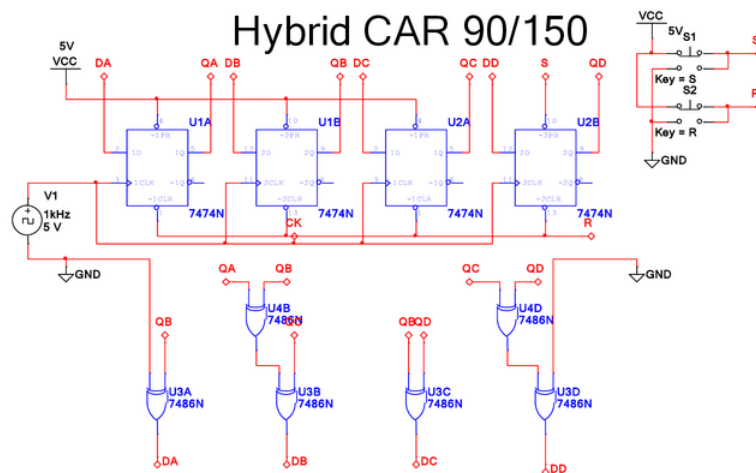


Figure 11 - Hybrid CAR Design

Q4	Q3	Q2	Q1	Decimal Value
0	0	0	1	1
0	0	1	1	3
0	1	1	0	6
1	0	1	1	11
0	0	1	0	2
0	1	0	1	5
1	1	0	1	13
1	0	0	1	9
0	1	1	1	7
1	0	0	0	8
0	1	0	0	4
1	1	1	0	14
1	1	1	1	15
1	1	0	0	12
1	0	1	0	10
0	0	0	1	1

Table 4 - Hybrid CAR Output

The hybrid CAR provides the maximal length pseudo-random pattern generator (PSPG). Alternating between the two rules can be virtually constructed to any length. For example, a 20-bit hybrid CAR with rule 90 set to zero given by 1111 0011 1011 0111 1111 produces a maximal length sequence of 1,048,515 patterns.



Sequence 1;3;6;11;2;5;13;9;7;8;4;14;15;12;10;1

Figure 12 - MultiSim Hybrid CAR

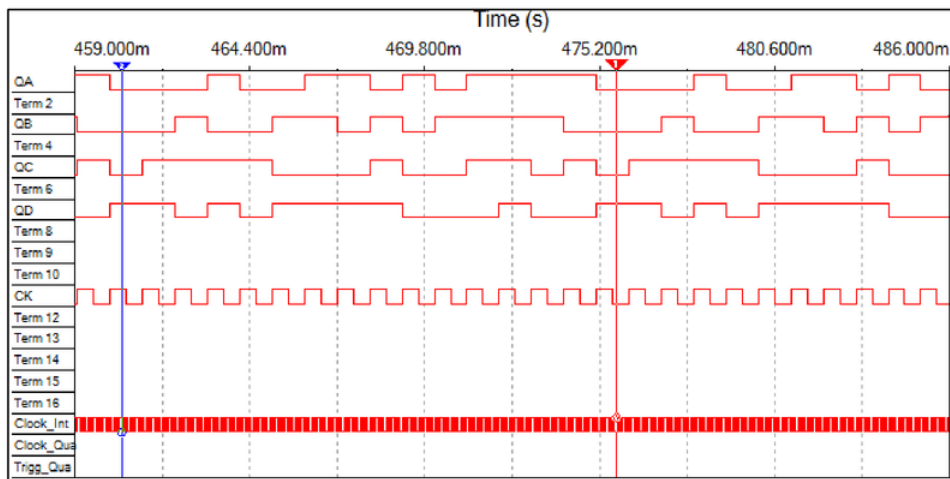


Figure 13 - MultiSim Hybrid CAR Results

Figure 11 shows the MultiSim circuit for the hybrid CAR. Figure 12 shows the achieved results. The maximum length of count 15 was achieved and the expected results match the achieved results.

Prototyping stages and link to theory is explained very well and it shows clearly the individual CAR rules do not give maximum count.

3.4 Development of 6-Bit Hybrid CAR

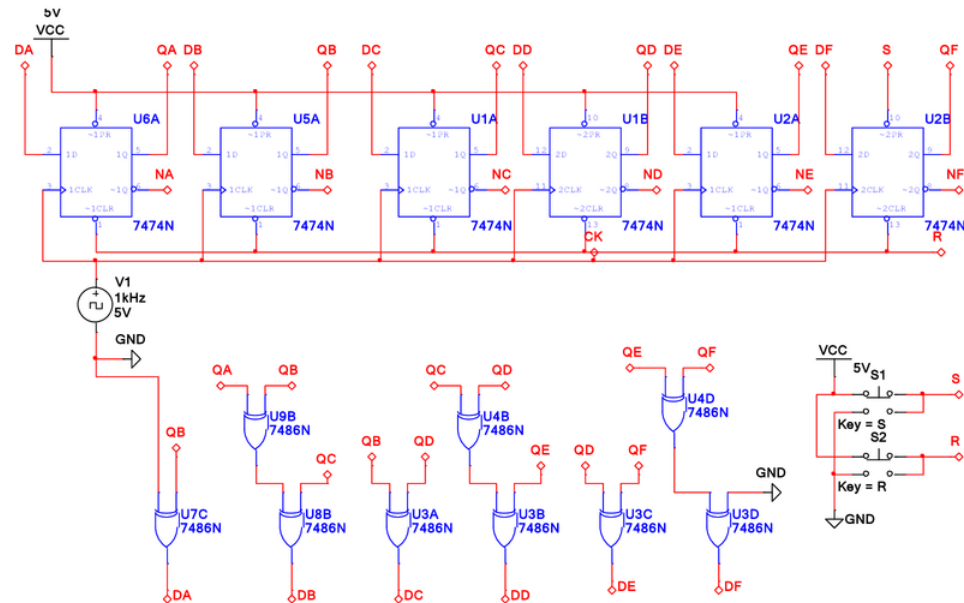


Figure 14 - 6-Bit Hybrid CAR

The maximum length of the 6-bit hybrid CAR shown in Figure 14 is a count of 63.

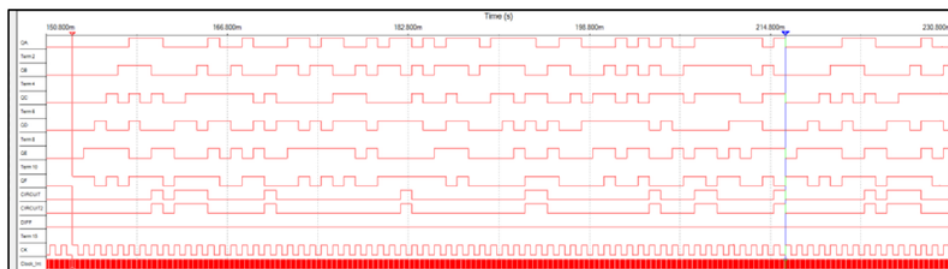


Figure 15 - 6-bit Hybrid CAR Results

Figure 16 shows the results of the MultiSim 6-bit hybrid CAR. The sequence of the MultiSim circuit can be seen below:

1,3,6,11,18,61,49,43,2,5,13,25,46,12,26,41,7,8,20,54,35,22,51,45,15,28,34,21,53,37,29,33,19,62,55,3
2,16,56,60,50,45,9,23,48,40,4,14,31,36,30,39,24,44,10,17,59,58,57,63,52,38,27,42,1

The circuit completes a full cycle of 63 and achieves its maximum length before repetition.

Good
analysis

4. Analysis of NI MultiSim Results

The Boolean expression for the working CAR circuit can be seen below:

$$NA.C.ND.E + B.NC.D.NF + A.C.ND.E + NB.D.NE.F$$

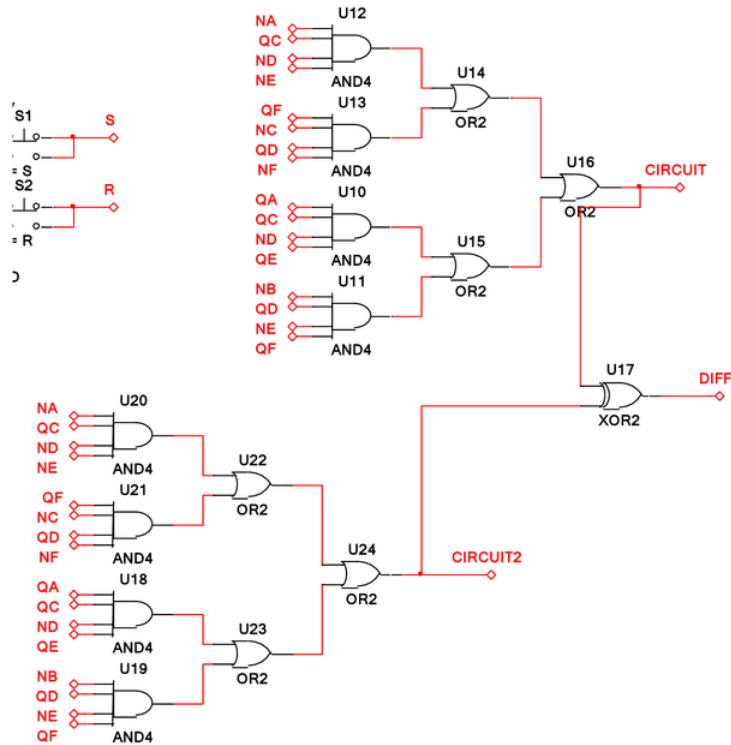


Figure 16 - Testing Circuit

Figure 16 shows the testing circuit used to prove the operation of the hybrid CAR. Fixed logic has been entered that will produce the same pattern as the pseudo-random generator. An XOR gate is being monitored, the pattern between the two inputs will be recorded and any variances highlighted. A variance in the inputs indicate that there is a fault in the circuit.

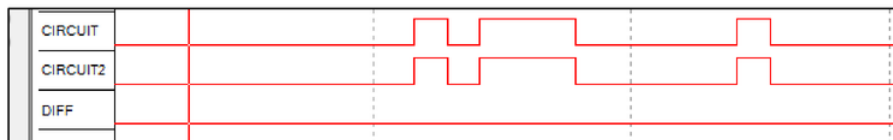


Figure 17 - XOR Results

Figure 17 shows the two inputs and output of the XOR gate from Figure 16. It can be seen that the two inputs changed simultaneously and there were no differences. Therefore the pseudo-pattern generator operation was proven to work. This circuit will be used in further analysis and testing.

Three faults were introduced into the circuit by altering this Boolean equation to prove the operation of the circuit. The Boolean expression for these three faults are:

$$\begin{aligned} & \text{NA.C.ND.E} + \text{B.NC.D.NF} + \text{A.C.ND.E} + \text{NB.1.NE.F (QD SET TO 1)} \\ & \text{NA.C.ND.E} + \text{B.NC.D.NF} + \text{A.C.ND.E} + \text{NB.0.NE.F (QD SET TO 0)} \\ & \text{NA.C.ND.E} + \text{B.NC.D.NF} + \text{A.C.ND.E} + \text{1.D.NE.F (NB SET TO 1)} \end{aligned}$$

Table 5 shows the results achieved from the MultiSim experiment with the three fault scenarios implemented.

Boolean Output from 6-Bit Hybrid CAR	Boolean Required From Good Circuit	Simulated Fault in Test Circuit	Fault Detected? (XOR)
NA.NB.NC.ND.E.F	$\text{NA.C.ND.E} + \text{B.NC.D.NF} + \text{A.C.ND.E} + \text{NB.D.NE.F}$	$\text{NA.C.ND.E} + \text{B.NC.D.NF} + \text{A.C.ND.E} + \text{NB.1.NE.F}$	Y
A.NB.NC.D.NE.F	$\text{NA.C.ND.E} + \text{B.NC.D.NF} + \text{A.C.ND.E} + \text{NB.D.NE.F}$	$\text{NA.C.ND.E} + \text{B.NC.D.NF} + \text{A.C.ND.E} + \text{NB.0.NE.F}$	Y
A.B.C.D.NE.F	$\text{NA.C.ND.E} + \text{B.NC.D.NF} + \text{A.C.ND.E} + \text{NB.D.NE.F}$	$\text{NA.C.ND.E} + \text{B.NC.D.NF} + \text{A.C.ND.E} + \text{1.D.NE.F}$	Y

Table 5 - MultiSim Results

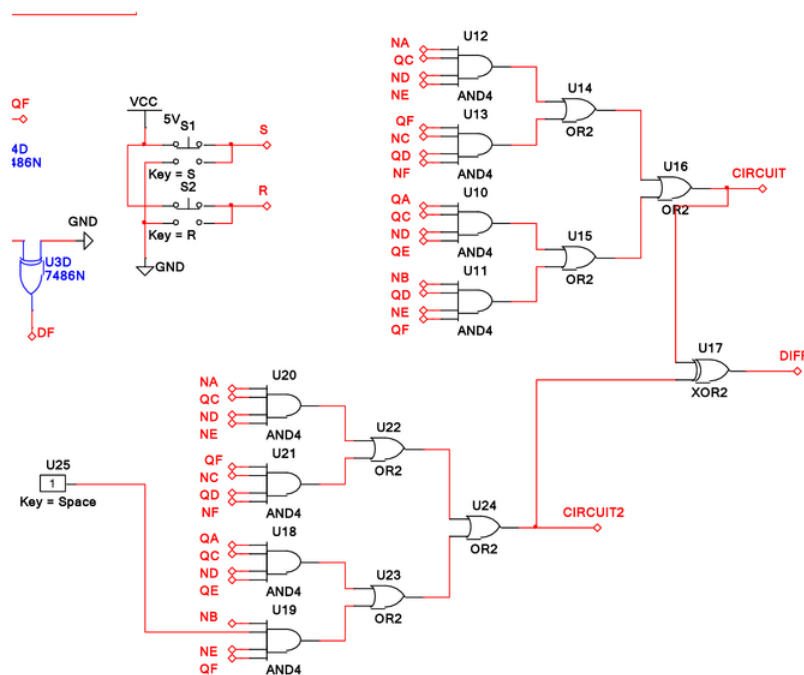


Figure 18 - Fault Test 1 Circuit

Good analysis of fault conditions.

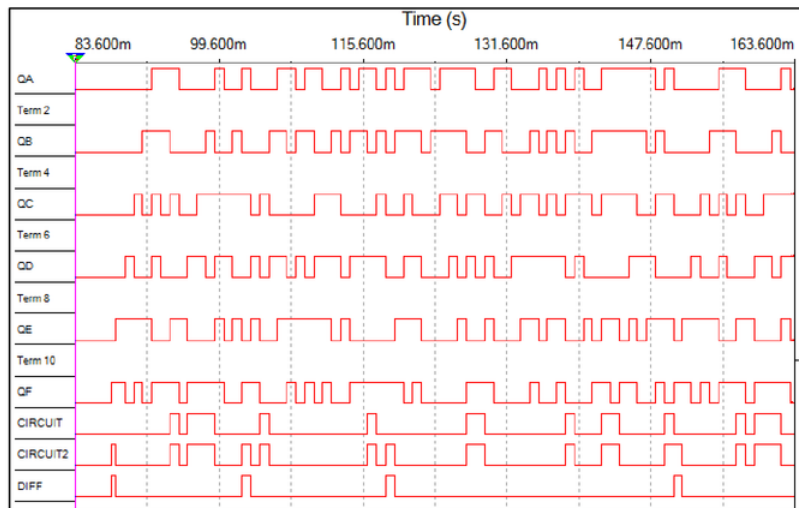


Figure 19 - Fault Test 1 Results

The last signal analysed in Figure 19 shows the output of the XOR signal. The signal goes active when a difference between the two circuits is detected. Figure 20 and Figure 21 shows the test circuit and results for the second fault test. Figure 22 and Figure 23 shows the test circuit and results for the third fault test.

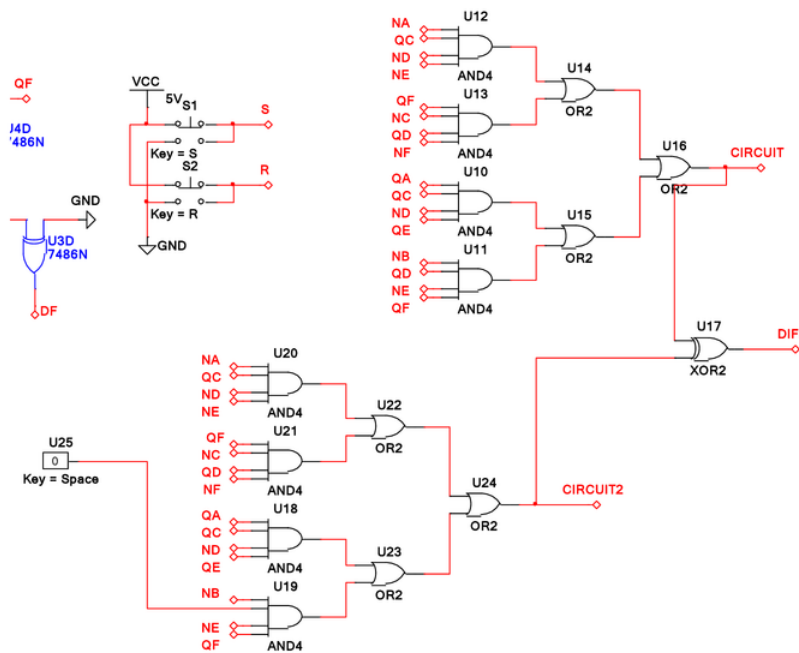


Figure 20 - Fault Test 2 Circuit

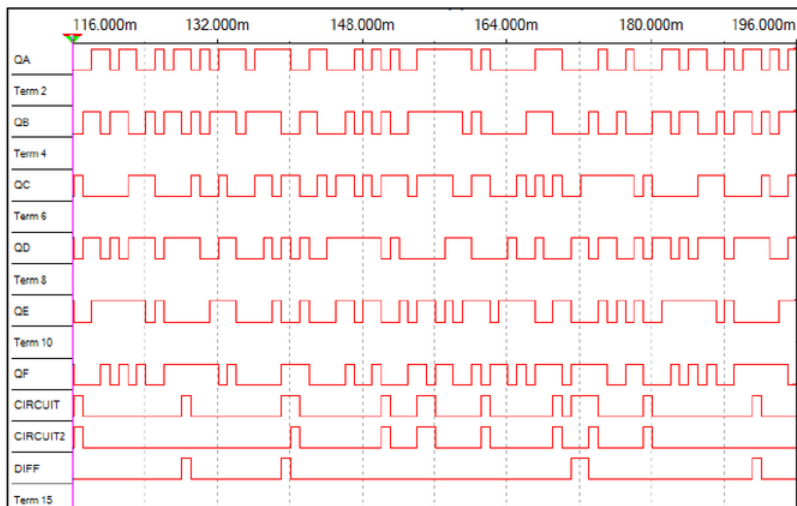


Figure 21 - Fault Test 2 Results

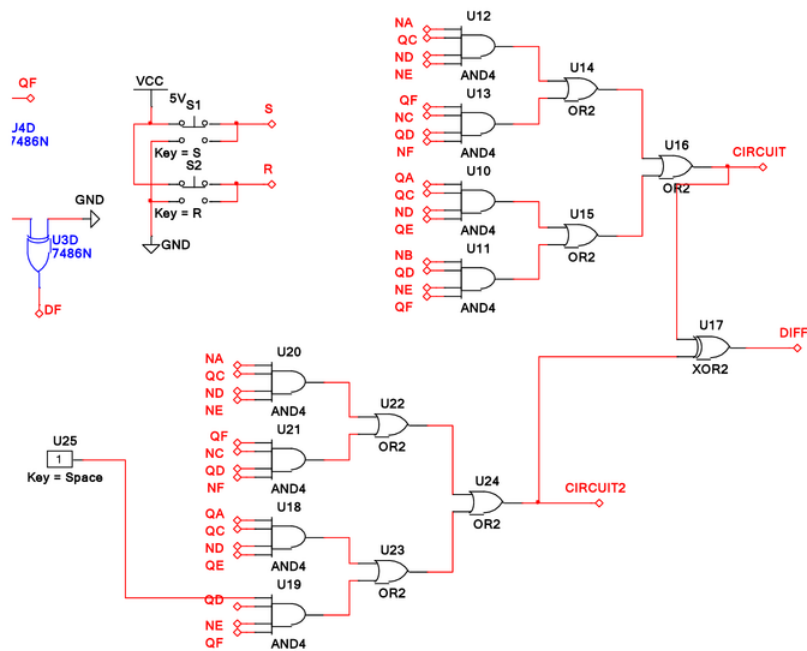


Figure 22 - Fault Test 3 Circuit

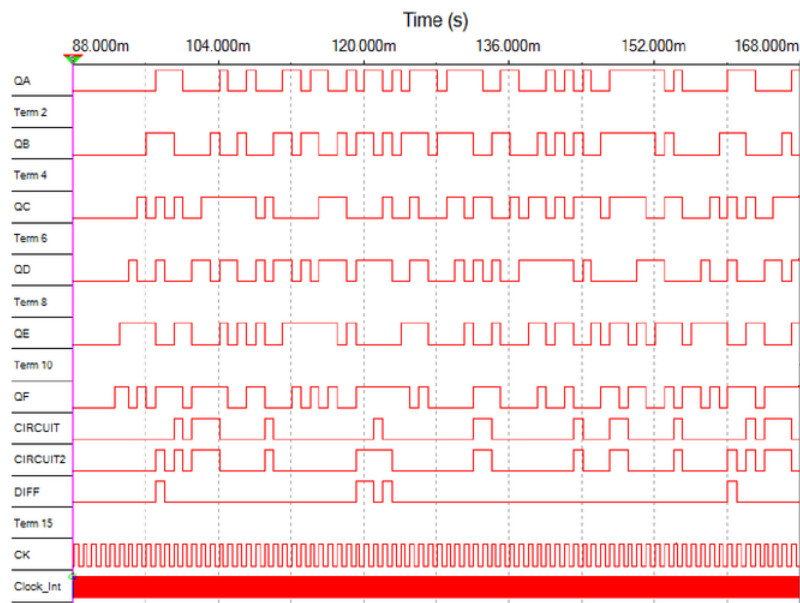


Figure 23 - Fault Test 3 Results

5. Analysis of results using hardware descriptive language FPGA technology

4 Bit CAR 90

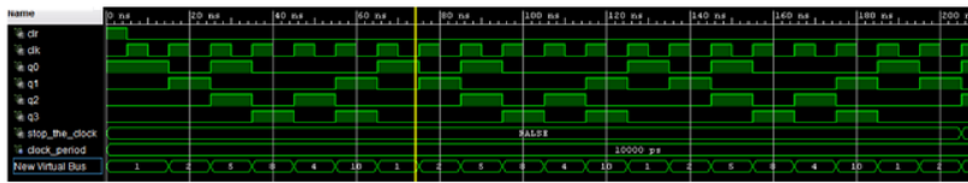


Figure 24 - 4-bit CAR 90 Xilinx

Figure 24 shows the results achieved from the 4-bit CAR 90 created in Xilinx. The results achieved from the Xilinx model are the same as the MultiSim model and thus further prove that the CAR 90 cannot achieve maximum length of count $15; 2^n-1$.

4 Bit CAR 150

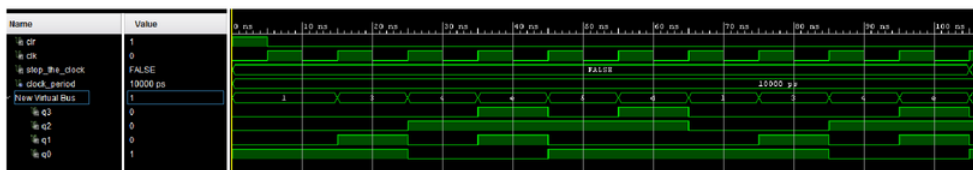


Figure 25 - 4-bit CAR 150 Xilinx

Figure 25 shows the results achieved from the 4-bit CAR 150 created in Xilinx. The results achieved are the same as the MultiSim model, this the CAR 150 cannot achieve maximum length.

4 Bit Hybrid CAR

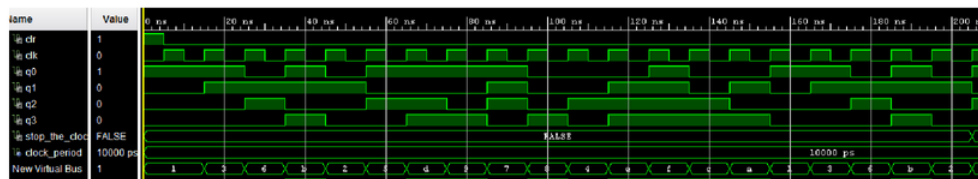


Figure 26 - 4-bit Hybrid CAR Xilinx

Figure 26 shows the results achieved from the 4-bit hybrid CAR. The results achieved match the results from the MultiSim model and maximum length is achieved as the count reaches 15 before cycle repetition.

*Prototype of 4 bit CAR
shows capability of
maximum count.*

6 Bit Hybrid CAR

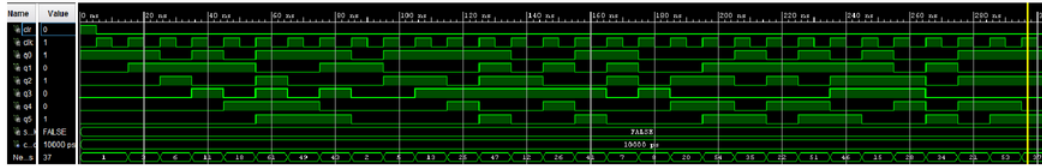


Figure 27 - Test Circuit Results 1

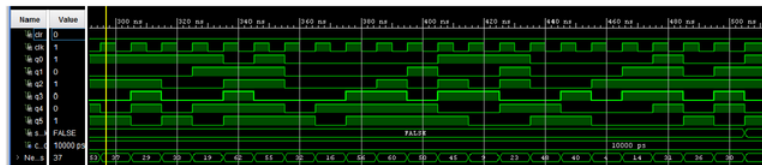


Figure 28 - Test Circuit Results 2

Figure 27 and Figure 28 show the results achieved from simulating the 6-bit hybrid CAR in Xilinx. The results from this test prove the hybrid CAR can achieve maximum count before repetition. Table 6 shows the order of count achieved from 6-bit hybrid CAR.

1st Half			2nd Half		
Binary Output	Decimal Value	Repeats	Binary Output	Decimal Value	Repeats
000001	1	No	010011	19	No
000011	3	No	111110	62	No
000110	6	No	110111	55	No
001011	11	No	100000	32	No
010010	18	No	010000	16	No
111101	61	No	111000	56	No
110001	49	No	111100	60	No
101011	43	No	110010	50	No
000010	2	No	101101	45	No
000101	5	No	001001	9	No
001101	13	No	010111	23	No
011001	25	No	110000	48	No
101111	47	No	101000	40	No
001100	12	No	000100	4	No
011010	26	No	001110	14	No
101001	41	No	011111	31	No
000111	7	No	100100	36	No
001000	8	No	011110	30	No
010100	20	No	100111	39	No
110110	54	No	011000	24	No
100011	35	No	101100	44	No
010110	22	No	001010	10	No
110011	51	No	010001	17	No
101111	47	No	111011	59	No
001111	15	No	111010	58	No
011100	28	No	111001	57	No
100010	34	No	111111	63	No
010101	21	No	110100	52	No
110101	53	No	100110	38	No
100101	37	No	011011	27	No
011101	29	No	101010	42	No
100001	33	No	000001	1	No

Table 6 - 6-bit Hybrid CAR Xilinx Results

Analysis shows 6 bit capability of maximum count for final design.

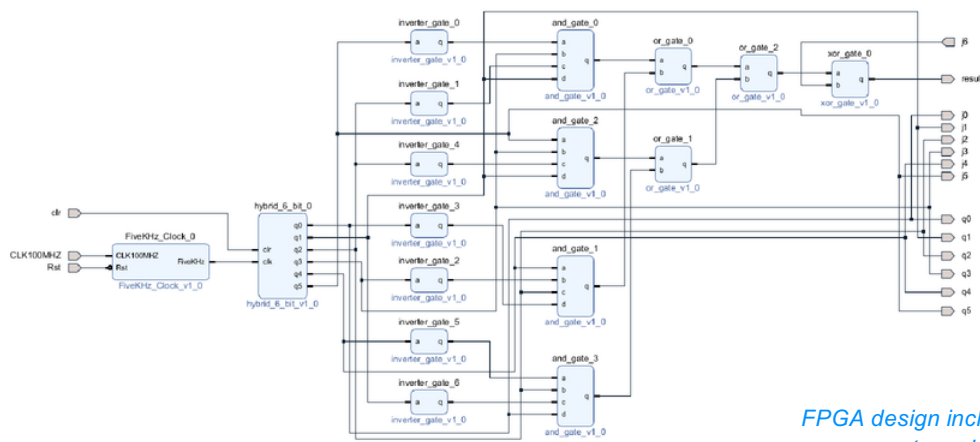


Figure 29 - 6-bit Hybrid CAR Test Circuit

FPGA design includes all components and out from x-or gate which is needed for comparison.

```

16 D(5) <= '0' xor qs(4);
17 D(4) <= qs(5) xor qs(4) xor qs(3);
18 D(3) <= qs(4) xor qs(2);
19 D(2) <= qs(3) xor qs(2) xor qs(1);
20 D(1) <= qs(2) xor qs(0);
21 D(0) <= qs(1) xor qs(0) xor '0';

```

Figure 30 - Hybrid CAR Wolfram Logic

Figure 29 shows the test circuit built in Xilinx, a description for each block can be found in the appendix. Figure 30 shows the logic of the 6-bit hybrid CAR. Bits 0, 2 and 4 use the 150 rule, therefore the output is dependent on the state of its two neighbouring bits as well as its own value. Bits 1, 3 and 5 use the 90 rule, therefore the output of those bits depend on the state of its two neighbouring bits.

The Xilinx program will be downloaded to a NEXYS4 test board, the XOR in the circuit will have an external input that will connect to a breadboard. The breadboard will run a circuit identical to the circuit in Figure 29, faults will be introduced into the breadboard circuit to prove the operation of the CAR program. The results recorded from the simulated fault test can be seen in Table 7.

Boolean Output from 6-bit Hybrid CAR	Good Test Circuit Boolean Output	Simulated Fault Test Circuit Boolean Output	Fault Detected? (XOR)
NA.B.NC.D.E.F	NA.C.ND.E + B.NC.D.NF + A.C.ND.E + NB.D.NE.F	NA.C.ND.E + B.NC.D. 1 + A.C.ND.E + NB.D.NE.F	Y
A.NB.C.ND.E.F	NA.C.ND.E + B.NC.D.NF + A.C.ND.E + NB.D.NE.F	NA.C.ND.E + B.NC.D.NF + A.C.ND. 0 + NB.D.NE.F	Y
NA.NB.NC.D.NE.NF	NA.C.ND.E + B.NC.D.NF + A.C.ND.E + NB.D.NE.F	NA.C.ND.'1' + B.NC.D.NF + A.C.ND.E + NB.D.NE. 1	Y

Table 7 - Test Board Results

Good analysis of fault conditions.

For the first fault test, bit 'F' on the input of second AND gate was held at 1. This made the AND gate output a 1, contrary to the output of good test circuit. Therefore the output on the connected XOR gate went high. Figure 31 shows the illustration of the fault and Figure 32 shows the fault present on the circuit. LED15 was connected to the XOR gate and can be seen as illuminated here, indicating fault.

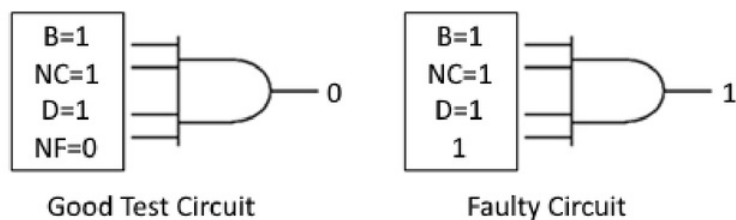


Figure 31 - Fault 1 Diagram



Figure 32 - Fault 1 Circuit

On the second fault test, bit 'E' on AND gate 3 was held at zero. This resulted in the AND gates output dropping to 0. The XOR indicator illuminated from this fault and the result can be seen and illustrated in Figure 33 and Figure 34.

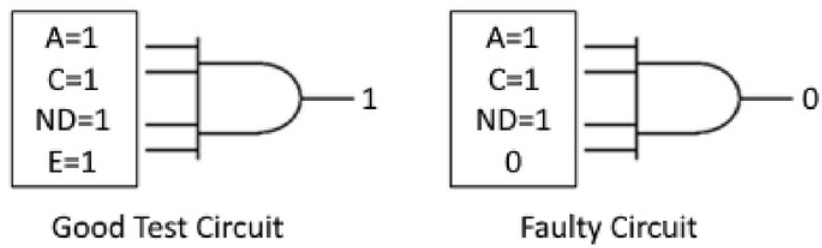


Figure 33 - Fault 2 Diagram

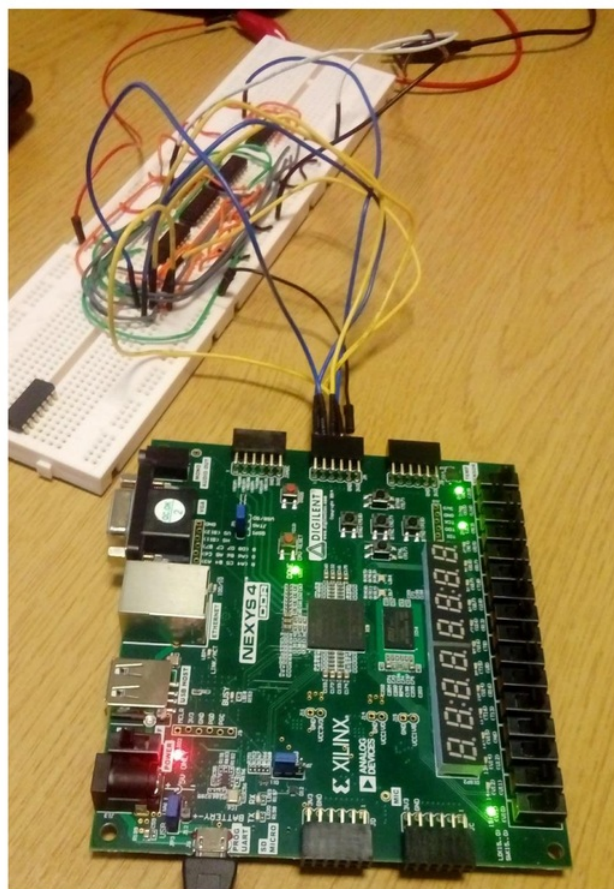


Figure 34 - Fault 2 Circuit

On the third fault test, bit 'F' on AND gate 4 was held at 1. This resulted in the AND gates output rising to 1. The XOR indicator illuminated from this fault and the result can be seen and illustrated in Figure 35 and Figure 36

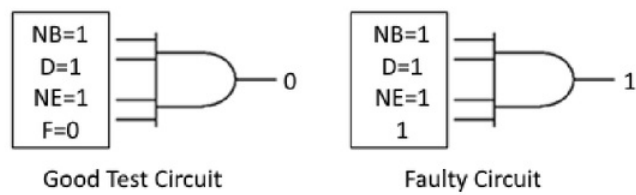


Figure 35 - Fault 3 Diagram

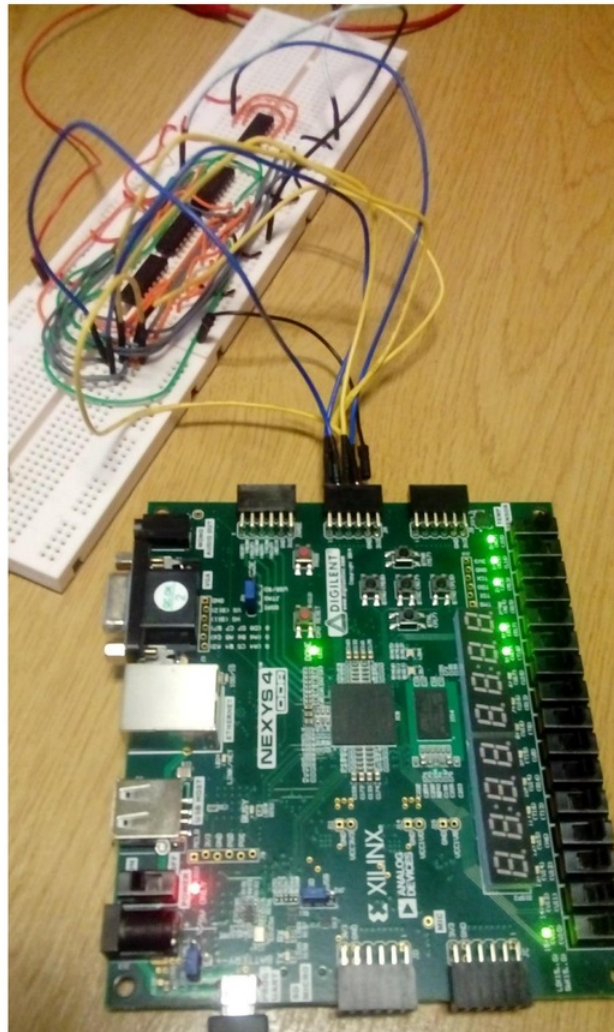


Figure 36 - Fault 3 Circuit

FPGA CAR
shows a test
for the CUT
practical and a
fault condition.

6. Analysis of PCB Design

Figure 37 and Figure 38 shows the circuit design developed in AutoDESK EAGLE. The circuit design shows the components, inputs and outputs. The circuit print shows the layout of the board, this design will be used to manufacture the board. The AutoDESK EAGLE software converts this document into several files that are used for specific stages during the manufacturing process.

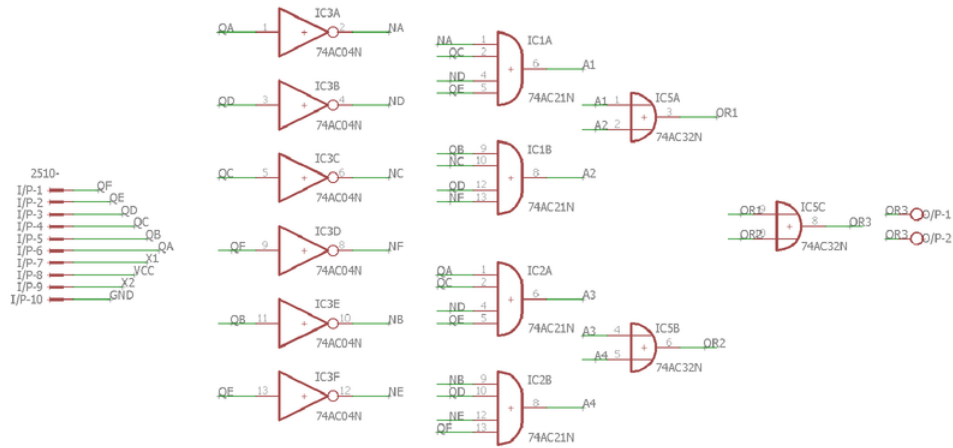


Figure 37 - Eagle Circuit Design

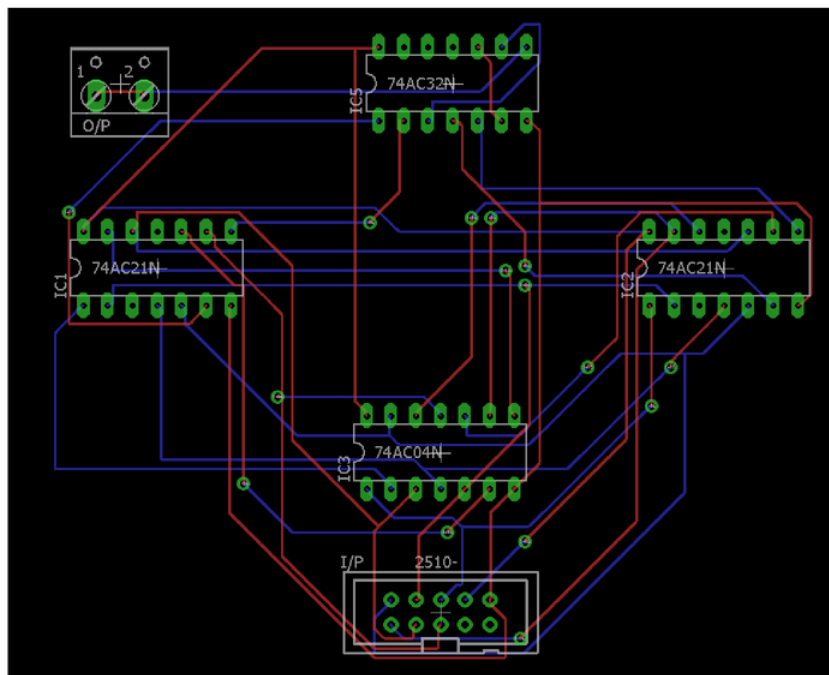


Figure 38 - Eagle Circuit Print



Figure 39 - Solder Stop Mask Solder Side

Figure 39 shows the first solder stop mask (.sts) file layout. This file is responsible for telling the system where the solder resin is layered onto copper traces on the board, this is to prevent unintended solder bridges.

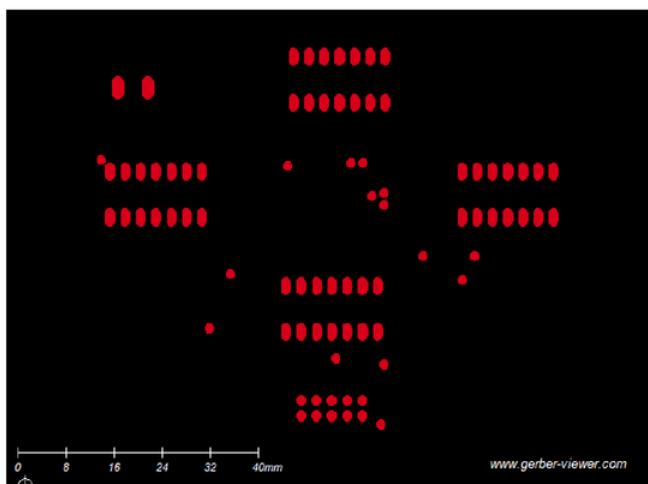


Figure 40 - Solder Stop File Component Side

Figure 40 shows the solder stop file for the component side of the board. This file is responsible for identifying where silicone needs placing onto solder connections on the components to prevent unintended solder bridges from forming.

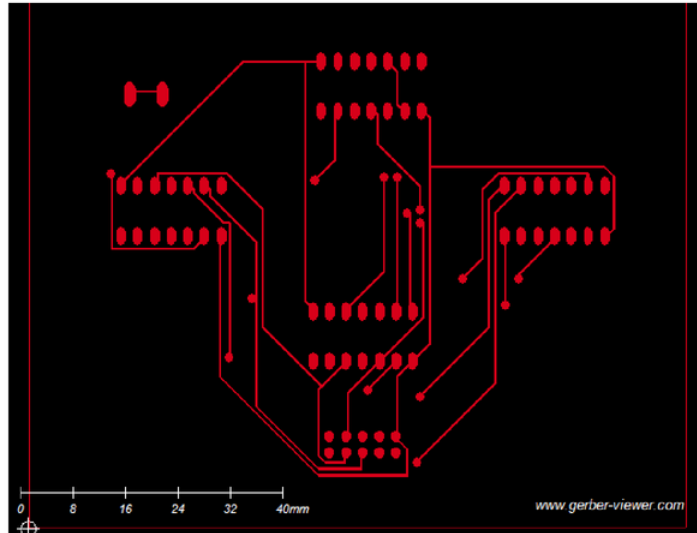


Figure 41 - Top Copper File

Figure 41 shows the top copper layer. This file identifies the copper path for the circuit. Figure 42 shows the bottom copper layer for the circuit.

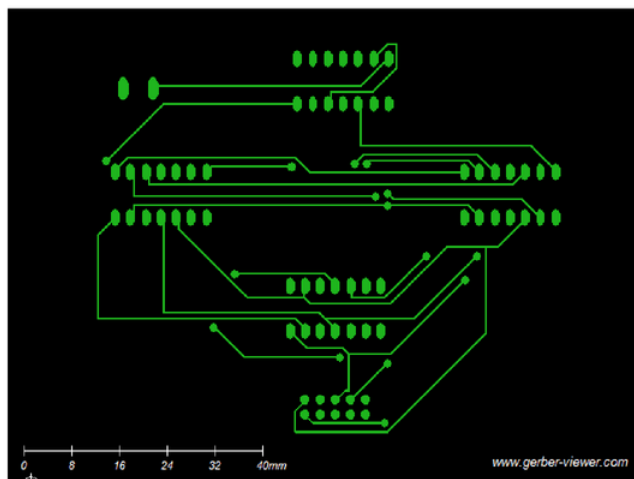


Figure 42 - Bottom Copper Layer

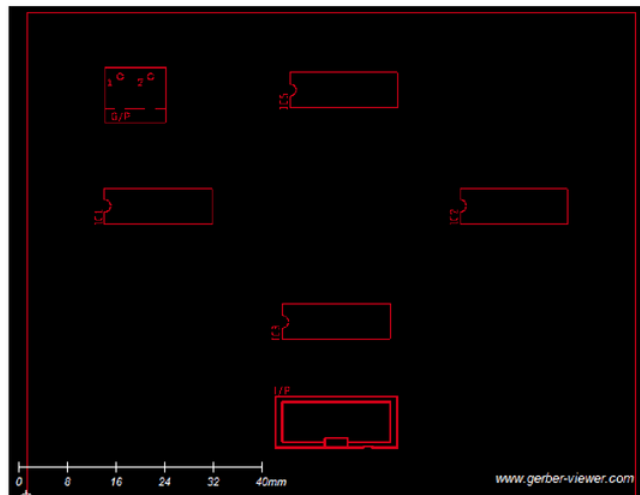


Figure 43 - Board Outline File

Figure 43 shows the outline of the board. Each component position is identified on the board.

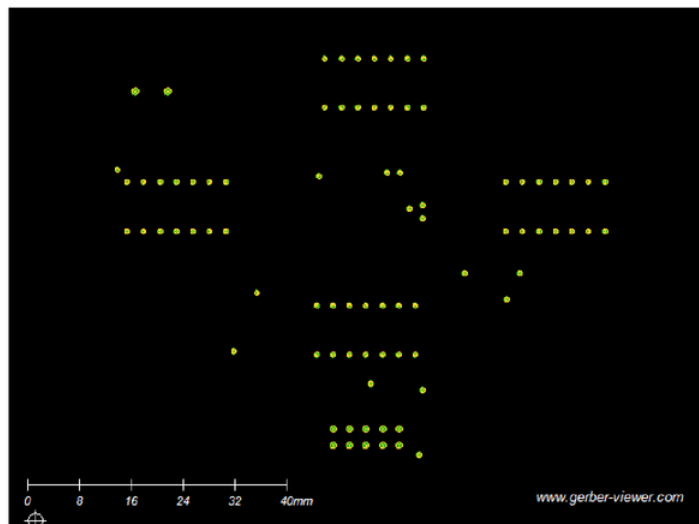


Figure 44 - Drill Hole File

Figure 44 shows the drill hole file. The yellow marks identify where manufacturing needs to drill holes in the board to be able to mount the components.

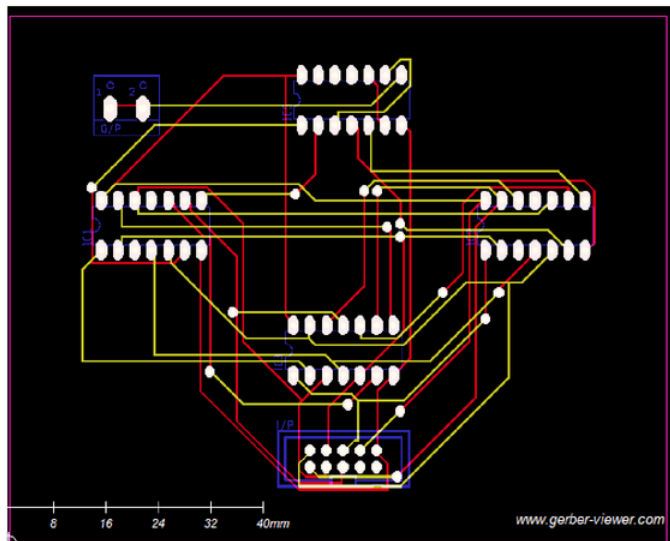


Figure 45 - Complete Manufacturing Design

Figure 45 shows the complete design of the PCB to be manufactured. The files will be sent off to an external company for manufacturing.

7. Practical PCB build and testing of prototype on prototyping board

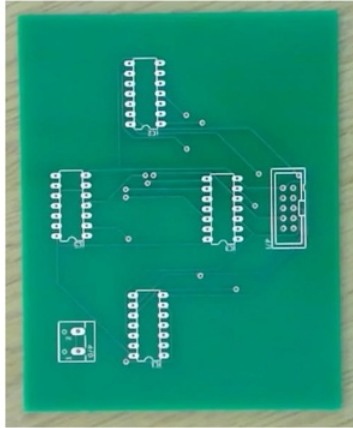


Figure 46 - Manufactured PCB Substrate

Figure 46 shows the manufactured substrate, created externally by the use of the files created through AUTODESK EAGLE. Components will be soldered onto the board in the designated area.

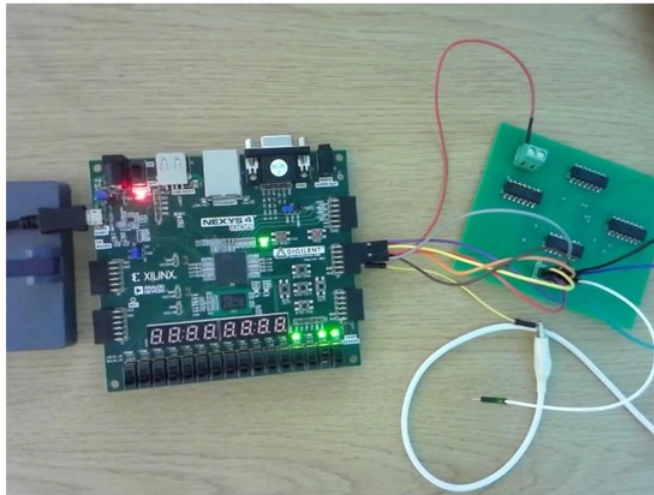


Figure 47 - PCB Test

Figure 47 shows the manufactured board being tested against a good test circuit inside the chip of the NEXUS board. A full cycle was completed and maximum count was achieved without LED15 indicating any errors.

PCB final test is explained well and good use of diagrams as evidence.

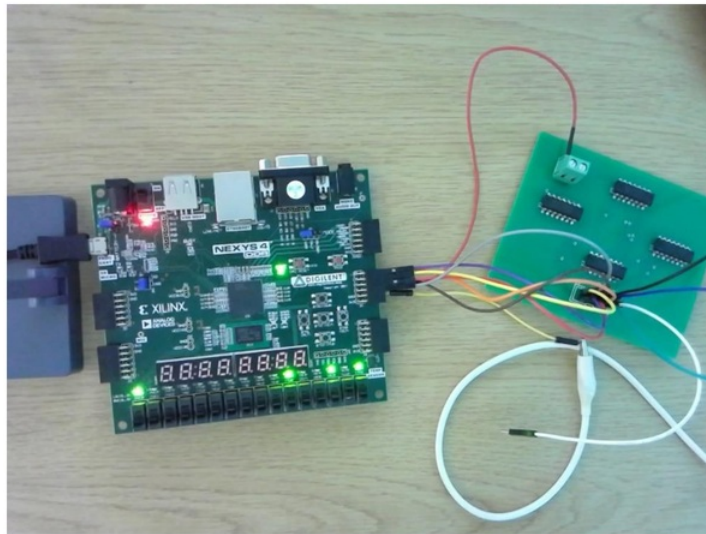


Figure 48 - PCB Fault Test

Figure 48 shows LED15 illuminating, indicating a fault. A purpose fault was placed in the circuit inside the NEXUS board to prove the operation of the fault indicator. This test proves that the fault logic works.

8. Conclusions & Recommendations

The aims of this assignment where:

1. Research into the designing and soldering of PCBs
2. Research AOI and AXI
3. Investigate and design pattern generators to test prototype designs
4. Use CAD/CAM to design and produce a PCB
5. Design and test a 'Pass/Fail' tester for the produced PCB.

Research was carried out on the designing and manufacturing process of creating a PCB from start to finish. Research was also carried out on automatic optical and x-ray inspection. Using MultiSim and Xilinx, prototype designs where created and tested against expected outputs. The Autodesk Eagle software was used to create CAD designs that where sent to an external manufacturer to produce the substrate. The components where then soldered onto the board to complete the PCB. The produced PCB was then tested against the 'Pass/Fail' circuit created in Xilinx and downloaded to the Nexus board. The PCB passed the test and the desired results were achieved, a fault was simulated to prove the operation of the fault logic.

All objectives of the assignment were achieved and an operating, tested PCB was produced.

The FPGA design shows that the virtual test cct can be created and integrated into a real world ATE design application. This is the theme for the conclusion.

6. Citation List

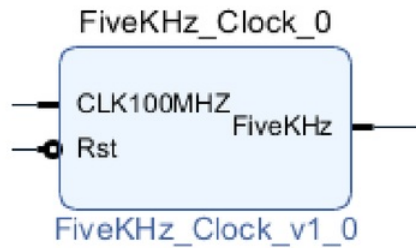
- Electronic Notes. (2011). *Automated X-Ray Inspection AXI for PCB & BGA*. Retrieved from Electronic Notes: <https://www.electronics-notes.com/articles/test-methods/automatic-automated-test-ate/axi-x-ray-inspection.php>
- Electronic Notes. (2011). *Automatic optical inspection, AOI systems*. Retrieved from Electronic Notes: <https://www.electronics-notes.com/articles/test-methods/automatic-automated-test-ate/aoi-optical-inspection.php>
- Made How. (2007, 05). *Printed Circuit Board*. Retrieved from How Products are Made: <http://www.madehow.com/Volume-2/Printed-Circuit-Board.html>
- Poole, I. (2012). *What is SMT?* Retrieved from Radio Electronics: <http://www.radio-electronics.com/info/data/smt/what-is-surface-mount-technology-tutorial.php>
- Surface Mount Process. (2015). *REFLOW SOLDERING PROCESS*. Retrieved from Surface Mount Process : <http://www.surfacemountprocess.com/reflow-soldering-process.html>
- Vishay. (2014). *Vishay Resistor Soldering Profile*. Retrieved from Vishay: <http://www.vishaypg.com/docs/63133/63133.pdf>

*This is
appropriate
for this report.*

7. Appendix

Clock

The clock is used for the program to be able to move from one state to the next. The clock is setup to convert a 100MHz input signal into a 5KHz pulse output.



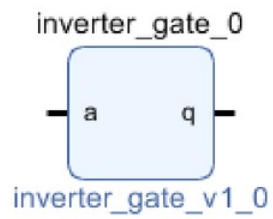
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
library UNISIM;
use UNISIM.VCOMPONENTS.ALL;
entity clk_hybrid_wrapper is
    port (
        CLK100MHZ : in STD_LOGIC;
        Rst : in STD_LOGIC;
        clr : in STD_LOGIC;
        q0 : out STD_LOGIC;
        q1 : out STD_LOGIC;
        q2 : out STD_LOGIC;
        q3 : out STD_LOGIC
    );
end clk_hybrid_wrapper;

architecture STRUCTURE of clk_hybrid_wrapper is
    component clk_hybrid is
        port (
            q0 : out STD_LOGIC;
            q1 : out STD_LOGIC;
            q2 : out STD_LOGIC;
            q3 : out STD_LOGIC;
            CLK100MHZ : in STD_LOGIC;
            Rst : in STD_LOGIC;
            clr : in STD_LOGIC
        );
    end component clk_hybrid;
begin
    clk_hybrid_i: component clk_hybrid
        port map (
            CLK100MHZ => CLK100MHZ,
            Rst => Rst,
            clr => clr,
            q0 => q0,
            q1 => q1,
            q2 => q2,
            q3 => q3
        );
end STRUCTURE;
```

*This is the wrapper file.
This should be the
program file to show
the clk division.*

Inverter Gate

The inverter gate simply converts the input on port A to the opposing voltage level on port Q.



```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

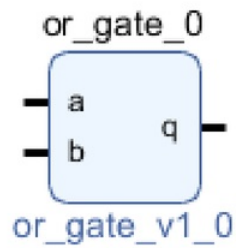
entity inverter_gate is
    -- Port ( );
    Port ( a : in std_logic;
          q : out std_logic);
end inverter_gate;
architecture Behavioral of inverter_gate is

    begin
        q <= not a;

end Behavioral;
```

OR Gate

The output on the OR gate goes high when an input signal has been detected on input port A or B.



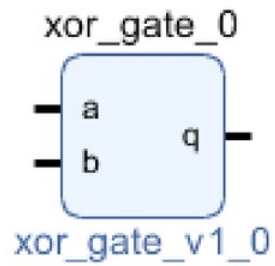
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity or_gate is
    -- Port ( );
    Port ( a : in std_logic;
           b : in std_logic;
           q : out std_logic);
end or_gate;
architecture Behavioral of or_gate is

begin
    q <= a or b;
end Behavioral;
```

XOR Gate

The output of the XOR gate goes high when a signal is detected on one of the inputs. If both inputs goes high the output switches off.



```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity xor_gate is
    -- Port ( );
    Port ( a : in std_logic;
           b : in std_logic;
           q : out std_logic);
end xor_gate;

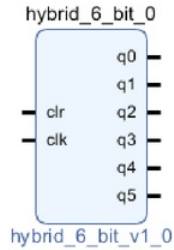
architecture Behavioral of xor_gate is

begin
    q <= a xor b;

end Behavioral;
```

Hybrid 6-bit CAR

The hybrid 6-bit CAR operates on wolframs rule 90 and rule 150. Bit 1,3 and 5 follow the 90 rule, therefore the outputs of those bits depend on the state of the neighbouring signals. Bit 0, 2 and 4 follow the 150 rule, therefore the output of those bit depend on its own current state as well as its neighbours state.



```
1  library IEEE;
2  use IEEE.STD_LOGIC_1164.all;
3
4  entity hybrid_6_bit is
5      port(
6          clr : in STD_LOGIC;
7          clk : in STD_LOGIC;
8          q0,q1,q2,q3,q4,q5 : out std_logic
9      );
10 end hybrid_6_bit;
1
2  architecture Behavioral of hybrid_6_bit is
3      signal D, qs: STD_LOGIC_VECTOR(5 downto 0);
4      begin
5
6      D(5) <= '0' xor qs(4);
7      D(4) <= qs(5) xor qs(4) xor qs(3);
8      D(3) <= qs(4) xor qs(2);
9      D(2) <= qs(3) xor qs(2) xor qs(1);
10     D(1) <= qs(2) xor qs(0);
11     D(0) <= qs(1) xor qs(0) xor '0';
12
13     -- Three D flip-flops
14     process(clk, clr)
15     begin
16         if clr = '1' then
17             qs <= "000001";
18         elsif clk'event and clk = '1' then
19             qs <= D;
20         end if;
21     end process;
22
23     q0 <= qs(0);
24     q1 <= qs(1);
25     q2 <= qs(2);
26     q3 <= qs(3);
27     q4 <= qs(4);
28     q5 <= qs(5);
29
30 end Behavioral;
```

Good appendix of VHDL components needed. This appendix could be referenced from within the main report.

Assignment 1 - Josh Coles

GRADEMARK REPORT

FINAL GRADE

GENERAL COMMENTS

Instructor

69/100

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

Text Comment. LabVIEW not used and intro is a little short on procedure.

PAGE 6

Text Comment. Images would have helped

PAGE 7

PAGE 8

PAGE 9

PAGE 10

Text Comment. Research of ATE for production industry should include diagrams for support.

PAGE 11

PAGE 12

PAGE 13

Text Comment. Explanation of the theory behind the CAR rules are explained well.

PAGE 14

PAGE 15

Text Comment. Prototyping stages and link to theory is explained very well and it shows clearly the individual CAR rules do not give maximum count.

Text Comment. Good analysis

Text Comment. This is the test cct and the DUT for comparison

Text Comment. Good analysis of fault conditions.

Text Comment. Prototype of 4 bit CAR shows capability of maximum count.

Text Comment. Analysis shows 6 bit capability of maximum count for final design.

Text Comment. FPGA design includes all components and out from x-or gate which is needed for comparison.

Text Comment. Good analysis of fault conditions.

Text Comment. FPGA CAR shows a test for the CUT practical and a fault condition. Good illustration.

PAGE 31

PAGE 32

PAGE 33

PAGE 34

PAGE 35

PAGE 36

Text Comment. PCB final test is explained well and good use of diagrams as evidence.

PAGE 37

PAGE 38

Text Comment. The FPGA design shows that the virtual test cct can be created and integrated into a real world ATE design application. This is the theme for the conclusion.

PAGE 39

Text Comment. This is appropriate for this report.

PAGE 40

Text Comment. This is the wrapper file. This should be the program file to show the clk division.

PAGE 41

PAGE 42

PAGE 43

PAGE 44

Text Comment. Good appendix of VHDL components needed. This appendix could be referenced from within the main report.

GRADING FORM: CSG-RUBIC

JOSHUA COLES

0

RESEARCH-INFORMED LITERATURE



SMT is covered and explained well. It is difficult to explain this without an illustration. 3/5 SMT technology is explained and includes the solder process using heat profiles. The PTH technology is not covered and in most cases PCB's will consist of both technologies. 3/5 Optical and x-ray inspection has been explained. Again here it would benefit from an illustration. 6/10 Intro is a little weak. No Labview is used. The research would have benefitted with diagrams. Refs could have been more academic, but there are relevant sources.

12/20

KNOWLEDGE AND UNDERSTANDING



Design of CAR using CAD and VHDL is shown clearly and demonstrates good understanding. All the processes are there showing links to theory. 5/7 PCB design process is explained using illustrated examples of the actual design used. 5/7 The design of the CAR using FPGA technology is demonstrated with practical prototypes. The explanations are clear and follow a logical sequence. 4/6

14/20

ANALYSIS



The test results proved the design and the comparison to the theory is demonstrated. The tests are explained and reasons given for errors. 8/10 The analysis of the FPGA results is linked with the theory which includes the appropriate illustrations. 3.,5/5 The PCB design process is logical and each layer is tested separately to the overall test. 3.5/5 This area is covered in detail.

15/20

PRACTICAL APPLICATION AND DEP.



The manufacturing process of the PCB and soldering is explained well with knowledge applied from research about heat profiles. The PCB test software was located on line and a professional test software was used to test and isolate each layer. 12/15 FPGA testing using prototypes is illustrated and explained in a clear logical manner. The sequence of events makes sense of the process. 7/10

19/25

SKILLS FOR PROFESSIONAL PRAC.



Use of appropriate software which includes software downloaded on line for PCB layer testing was professional. 3.5/5 The report structure and presentation is very good. 3.5/5 Conclusion is a little weak and does not include any of the significant findings from this task. 2/5

9/15