1708165

by Adam Bedner

A WELL WRITTEN REPORT COVERING MOST ASPECTS OF THE BRIEF. CONCLUSIONS WERE VERY BRIEF.

COMPREHENSIVE FEEDBACK FROM 1ST MARKER.

MARK AGREED

FILE	13694_ADAM_BEDNER_1708165_178073_1800103959.PDF (2.71M)		
TIME SUBMITTED	14-MAY-2021 11:31PM (UTC+0100)	WORD COUNT	4318
SUBMISSION ID	153308904	CHARACTER COUNT	23698

Electrical & Electronic Engineering BSc

SGEE6002 Electronic Production & Test

Pseudo Random Pattern Generators

Name: Adam K Bedner

Student number: 1708165

Date submitted: 14/05/2021

Word Count: 3253

Table of Contents

Table of Figures 3
1. Research 4
1.1 Automatic Machine vision inspection 4
1.2 Automatic Colour Recognition 5
1.3 Histogram Equalisation 6
1.4 Colour extraction 7
2. Design Requirements and Constrains 9
2.1 4-bit 9
2.1.1 CAR90 9
2.1.2 CAR150 11
2.1.3 CAR90/150 Hybrid 13
2.2 6-bit CAR90/150 Hybrid 16
3. Fault simulation on MultiSim using 6-bit CAR90/150 18
3.1 Fault QA SA 0 18
3.2 Fault QD SA 1 19
3.3 Fault QE SA 0 20
4. VHDL implementation 21
4.2 IPs 24
5. Practical experiment with fault simulation. 26
6. Conclusions. 28
References 29

The VHDL implementation

and testing has many parts which should be in the table of contents

Table of Figures

Fig. 1 Pattern matched objects from a stored template. 4
Fig. 2 Pattern matching using template. 4
Fig. 3 HSL Colour spectrum 5
Fig. 4 Colour spectrum array of 16 elements 5
Fig. 5 Histogram of an image 6
Fig. 6 Equalised histogram 7
Fig. 7 Grey scale image 7
Fig. 8 Binary image with only 2 values in the colour plane 8
Fig. 9 Visual representation of the CAR90 rule 9
Fig. 10 CAR90 circuit 10
Fig. 11 CAR90's simulation result 10
Fig. 12 Visual representation of the CAR150 rule 11
Fig. 13 CAR150 Circuit 11
Fig. 14 Simulation results of CAR150 circuit. 12
Fig. 15 Basic schematic for CAR90/150 Hybrid circuit 13
Fig. 16 CAR90/150 Hybrid Circuit 14
Fig. 17 Simulation Results for CAR90/150 Hybrid 14
Fig. 18 6-bit CAR90/150 Circuit 17
Fig. 19 CAR90/150's result 17
Fig. 20 QA SA 0 18
Fig. 21 QA SA 0 Simulation results 18
Fig. 22 QD SA 1 19
Fig. 23 Simulation results for QD SA 1 19
Fig. 24 QE SA 0 20
Fig. 25 QE SA 0 Simulation Results 20
Fig. 26 Simulation of CAR90 in VHDL 21
Fig. 27 Simulation results of the VHDL Code for CAR150 22
Fig. 28 Simulation result of the CAR90/150 4-bit code 23
Fig. 29 Simulation results of the 6-bit CAR90/150 (Broken in four parts) 24
Fig. 30 Clock speed reduction block 24
Fig. 31 6-bit CAR90/150 IP block testing. 25
Fig. 32 practical test of the VHDL IP design 25
Fig. 33 Final IP design 25
Fig. 34 The practical set up 26
Fig. 35 QD SA 0 fault simulation 26
Fig. 36 QD SA 1 simulation result 27
Fig. 37 QB SA 1 Simulation result 27

Good list of figures

1. Research

1.1 Automatic Machine vision inspection

One of the more important and commonly used processes used in machine vision applications is locating an object of interest within a camera's field of view.
Pattern matching software such as LabVIEW can be deployed for fast response times and manipulation of process result data. The results can be used for a variety of post processes such as robotic actions or decisions and pick and place on a conveyor system.
This process makes possible to locate a pattern, including multiple occurrences, which are translated and slightly rotated, with sub pixel accuracy. The process performance efficiency is increased when scene lighting changes uniformly (Wilson, 2016).

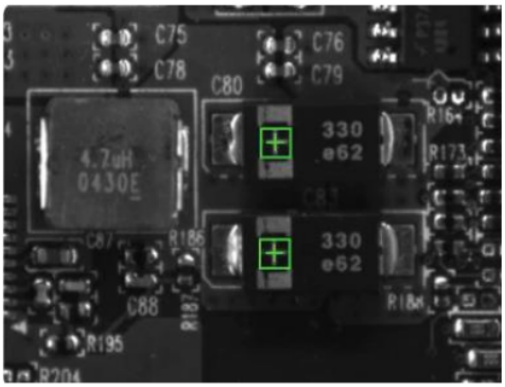


Fig. 1 Pattern matched objects from a stored template.

Good example with supporting source

A pattern matching algorithm needs to locate the reference pattern in an image even if the pattern in the image is rotated or scaled. When a pattern is rotated or scaled in the image, the pattern matching tool can detect the following items in the image:

- The pattern in the image
- The position of the pattern in the image
- The orientation of the pattern
- Multiple instances of the pattern in the image, if applicable

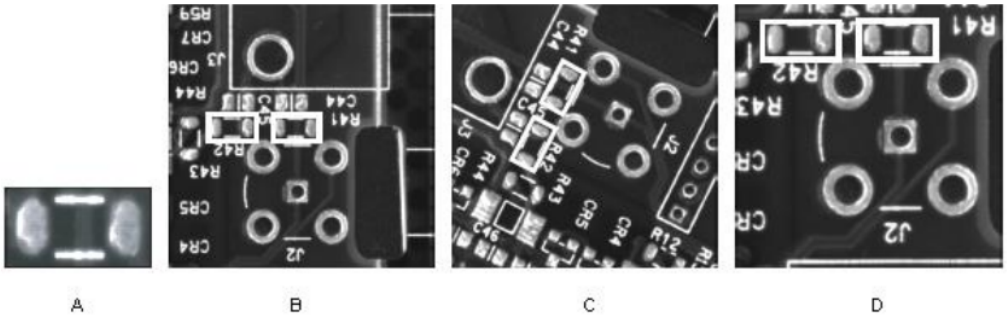


Fig. 2 Pattern matching using template.

This is a good

In Fig. 2, image *a* shows a template image; image *b* shows a template match shifted in the image; image *c* shows a template match rotated in the image; image *d* shows a template match scaled in the image. Images b, c, and d also illustrate multiple instances of the template (NI Zone, nd)

Image processing uses algorithms for the processes and this may include canny edge detection to enhance the pattern matching process accuracy. According to Rasika et al, 2020 a trial and error method appears to be the best approach to get the best test results.

Filters can be used in the frequency domain and special (image) domain to enhance features of interest.

A good significant point made here

1.2 Automatic Colour Recognition

The colour spectrum represents the three dimensional colour information associated with an image that can be used by many of NI IMAQ vision colour processing software. The colour spectrum can be used for colour matching, colour location, colour pattern matching applications using the vision software.

The colour spectrum represents the colour distribution of an image in the HSL space as shown on Fig. 3. If the input image is RGB format, this must be converted to HSL and then the colour spectrum can be computed from the HSL space. This improves the operation speed greatly of the process (NI Vision Concepts, 2000).

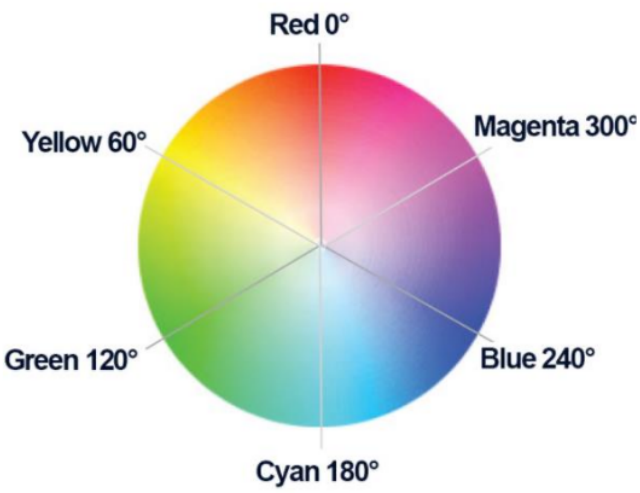


Fig. 3 HSL Colour spectrum

This is the most important point about colour matching and it is

The first element in a colour spectrum array represents the high saturation part in the first sector, the second element represents the low saturation part, the third element the high saturation part of the second sector and so on. If there are n bins in the colour space, the colour spectrum array contains n + 2 elements. The last two components in the colour spectrum represent the black and white colour, respectively (NI Vision Concepts, 2000).

A practical exercise using LabVIEW Machine Vision has be carried out to test this process and the results are shown in the array elements in Fig. 4.

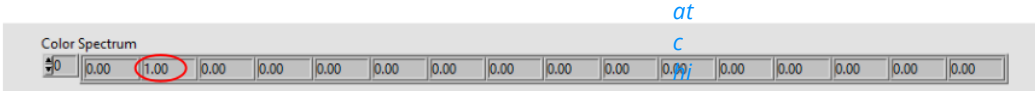


Fig. 4 Colour spectrum array of 16 elements

Very good explanation of this method

The demonstration shows that the element 1 in the array is at maximum score value 1.0 and has detected the saturated colour red in the spectrum. This can be used as a colour matching or detection process.

A colour spectrum with a larger number of bins (elements) represents the colour information in an image with more detail, such as a higher colour resolution, than a spectrum with fewer bins.

In IMAQ Vision you can choose between three colour sensitivity settings—low, medium, and high. Low divides the hue colour space into seven sectors, giving a total of 2 × 7 + 2 = 16 bins. Medium divides the hue colour space into 14 sectors, giving a total of 2 × 14 + 2 = 30 bins. High divides the hue colour space into 28 sectors, giving a total of 2 × 28 + 2 = 58 bins (NI Vision Concepts, 2000).

Excellent work on this area

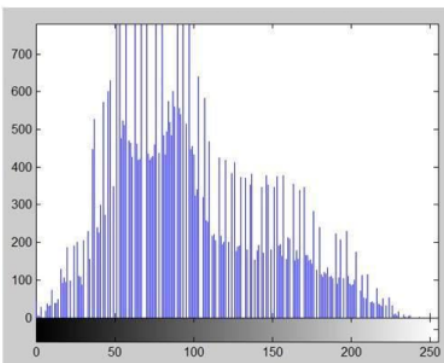
1.3 Histogram Equalisation

A histogram of an image is the graphical interpretation of the image's pixel intensity values. It can be interpreted as the data structure that stores the frequencies of all the pixel intensity levels in the image (Htoon, 2020).

Histogram purpose is defined

A histogram of an image is the graphical interpretation of the image's pixel intensity values. It can be interpreted as the data structure that stores the frequencies of all the pixel intensity levels in the image.

The histogram shown in Fig. 5 is of a typical image where the intensities are gathered in one section of the plane.



Good image to support explanation

Fig. 5 Histogram of an image

Histogram Equalization is an image processing technique that adjusts the contrast of an image by using its histogram. To enhance the image's contrast, it spreads out the most frequent pixel intensity values or stretches out the intensity range of the image. By accomplishing this, histogram equalization allows the image's areas with lower contrast to gain a higher contrast.

The histogram shown in Fig. 6 is the result of the equalisation process on the histogram in Fig. 5.

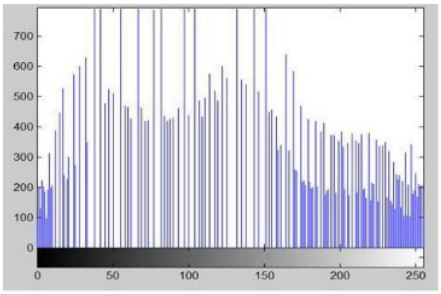


Fig. 6 Equalised histogram

A very good example for this process an its benefits

1.4 Colour extraction

One of the key components of displaying images is the display mode that the video adaptor operates. The display mode indicates how many bits specify the colour of a pixel on the display screen. Generally, the display mode available from a video adaptor ranges from 8 bits to 32 bits per pixel.

If you have an 8-bit display mode, a pixel can be one of 256 different colours. If you have a 16-bit display mode, a pixel can be one of 65,536 colours.

Many processes will only require an 8 bit mode display and this can be implemented in the process by extracting the colour plane and converting to the grey scale as shown in Fig. 7. This will greatly increase the performance of the process in real time (NI Vision Concepts, 2000).

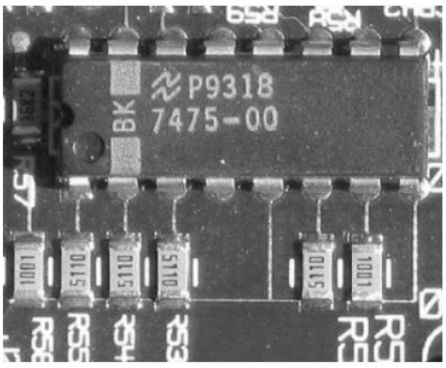


Fig. 7 Grey scale image

Very good example of colour extraction which is a common process in image processing for speed.

Another method of increasing processing performance is to convert the image to the binary scale which will only have two colour values and is used to separate quantities.

The medical industry would use this method to separate bacteria good and bad cells by using a DICOM camera microscope as the sensor device. (NI IMAQ, 2004).

This is a significant point to make as the medical industry uses the same processing metho

Fig. 8 Binary image with only 2 values in the colour plane

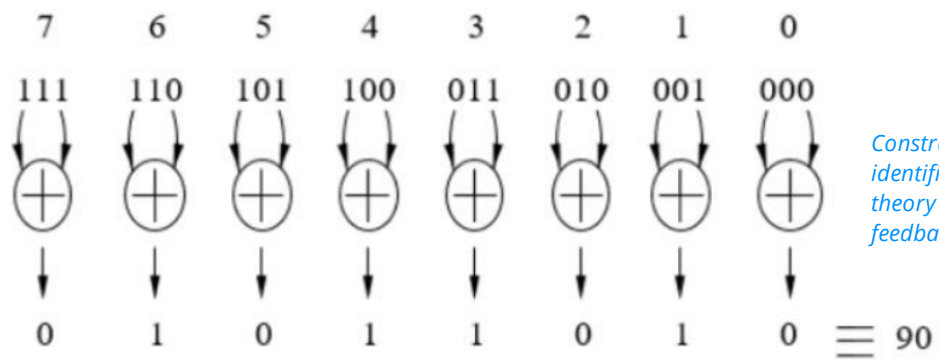
Good image to support research

2. Design Requirements and Constrains

This section is designated to the design of 4 bit CAR90, CAR150, CAR90/150 Hybrid and also 6 bit CAR90/150 Hybrid. The design must not have any repeats in the sequence and the pattern generator to reach maximum possible count for its type.

2.1 4-bit

2.1.1 CAR90



Constraints have been identified using the theory model for feedback

Fig. 9 Visual representation of the CAR90 rule

The output of the CAR90 is determined by flip-flops. The state of these is determined by the output of the flip-flops next to that one. If there is not an adjacent flip-flop, the end is tied to a constant 0, in most cases, ground. The output of the adjacent flip-flops is fed into an XOR gate, and then into the flip-flop's input. The pattern generator is triggered by a pulse to the enable input of the lowest bit; the rest of the flip-flops are constantly enabled.
Based on that information, the sequence generated can be predicted as it is shown in the table below:

QA	QB	QC	QD	Decimal
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

Good explanation with supporting table of expected results

Following, the generator was designed using NI MultiSim simulating program. The circuit is shown in Fig. 10. The circuit has been simulated and tested with a logic analyser. The results from the logic analyser can be seen in Fig. 11.

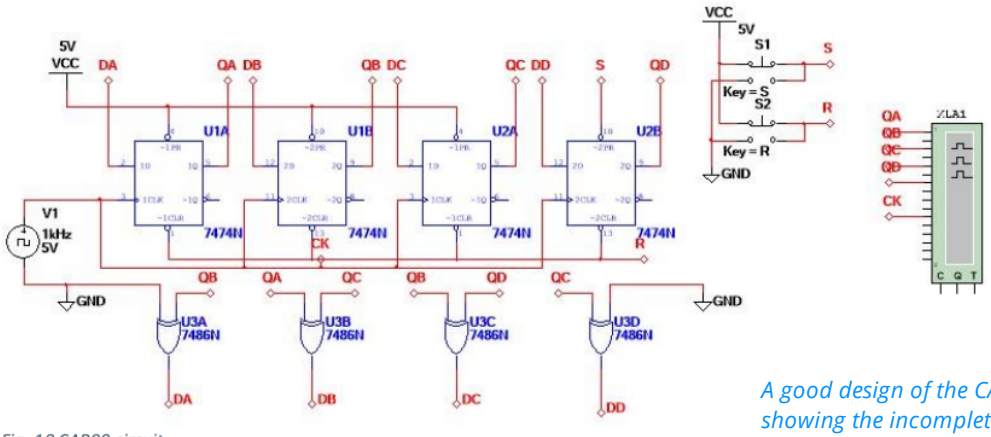


Fig. 10 CAR90 circuit

A good design of the CAR 90 and test showing the incomplete random count

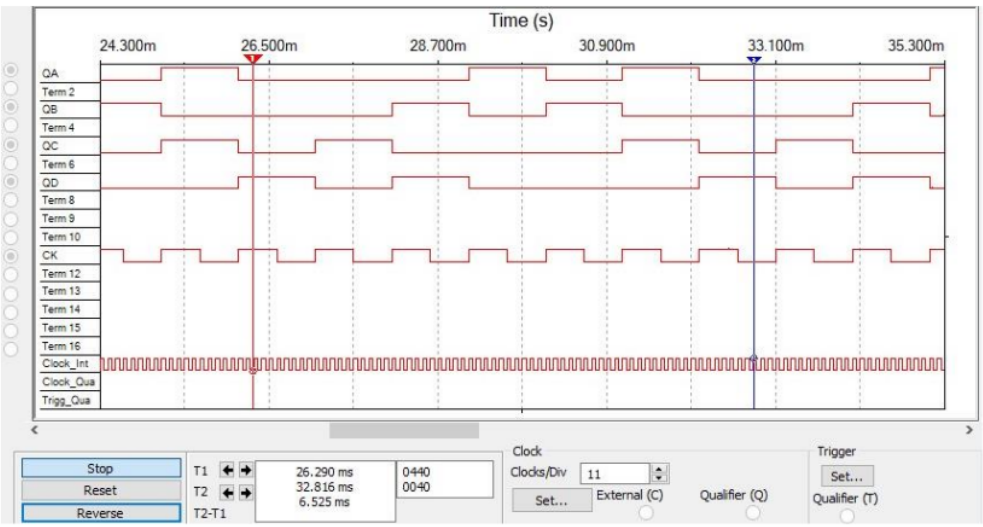


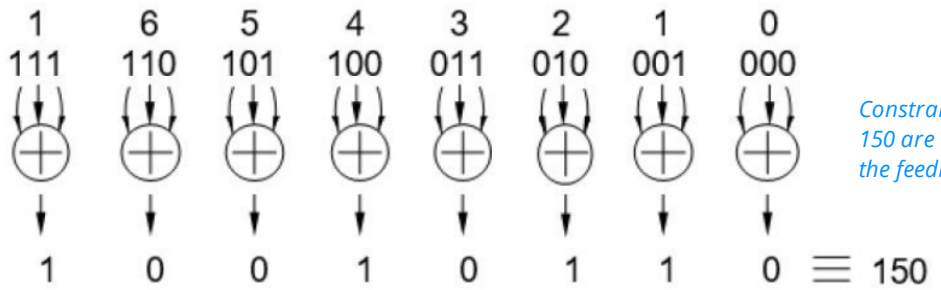
Fig. 11 CAR90's simulation result

The Table below summarizes and compares the expected and simulated results, which proves that the circuit is working properly without faults.

Expected	Simulated
1	1
2	2
5	5
8	8
4	4
10	10
1	1

Good table of results showing comparison to theory

2.1.2 CAR150



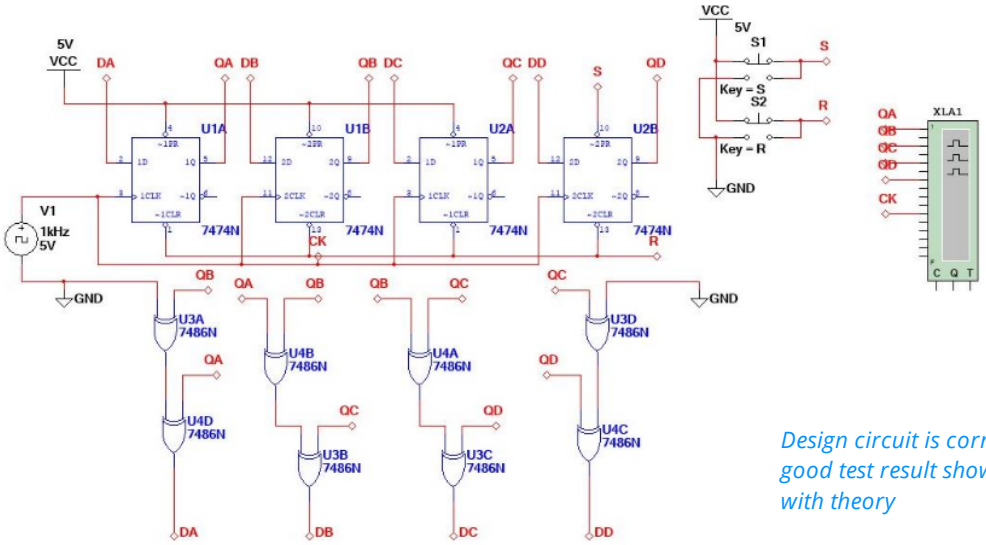
Constraints for the CAR 150 are identified using the feedback model

Fig. 12 Visual representation of the CAR150 rule

Similarly as CAR90, the output of the flip-flop is the CAR150 is determined by the outputs of the adjacent flip-flops, but now also its own output plays a part. 3 gate XOR are now used and the outputs are in a feedback loop. The output sequence can also be predicted and that is shown in the table below.

QA	QB	QC	QD	Decimal
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

Following, the circuit was designed (Fig. 13) and simulated. The circuit was probed with a logic analyser and the results are presented in Fig. 14.



Design circuit is correct and a good test result showing the link with theory

Fig. 13 CAR150 Circuit

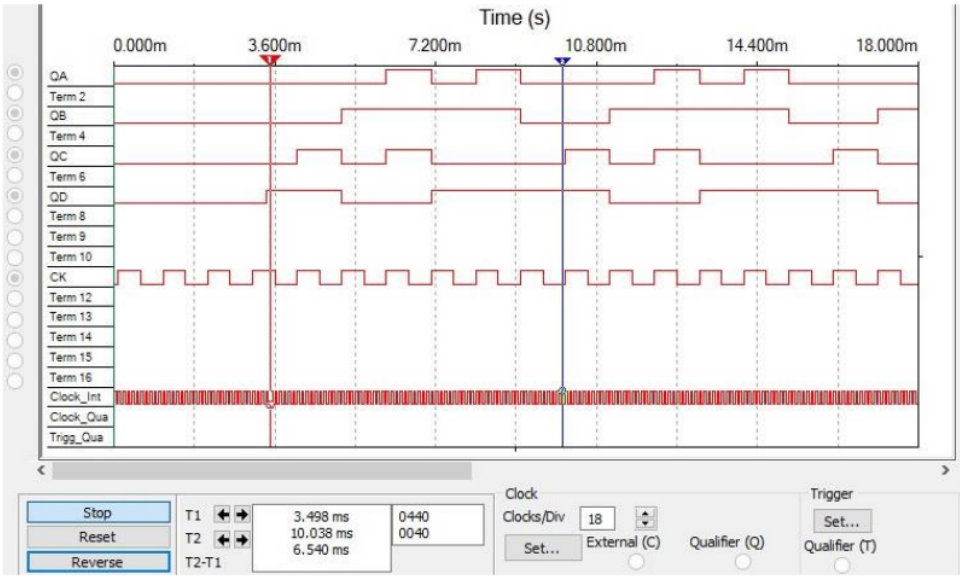


Fig. 14 Simulation results of CAR150 circuit.

The Table below summarizes and compares the expected and simulated results, which proves that the circuit is working properly without faults.

Expected	Simulated
1	1
3	3
4	4
14	14
5	5
13	13
1	1

Comparison showing the link with theory

2.1.3 CAR90/150 Hybrid

As the name suggests, this generator is a combination of the two previously discussed generators. The flow of the generated pattern follows the basic schematic shown on Fig. 15. As before, the generator is triggered by enabling the lowest importance bit's flip-flop.

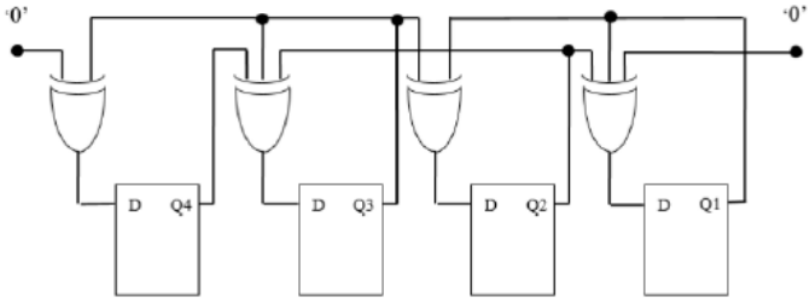


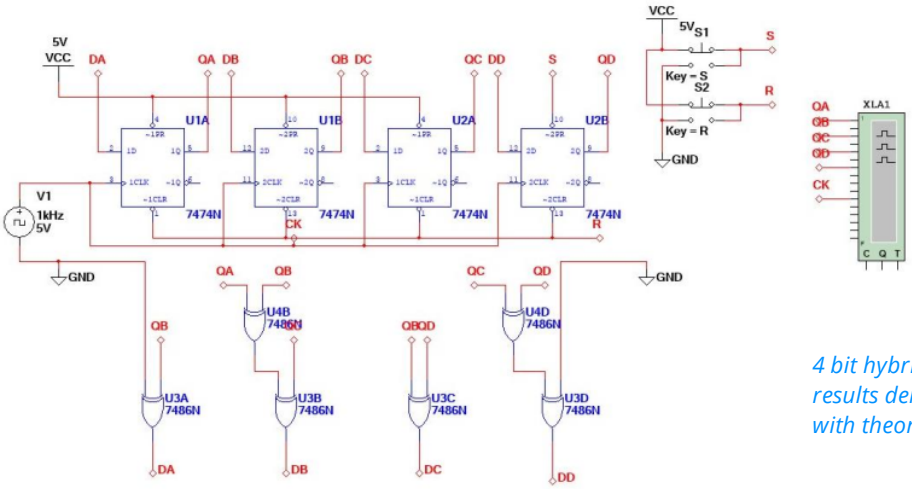
Fig. 15 Basic schematic for CAR90/150 Hybrid circuit

4 bit hybrid design is identified with predicted test results

Based on that, the pattern can be predetermined as shown in the table below:

QA	QB	QC	QD	Decimal
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

Following, the circuit was designed and tested. The graph from the logic analyser is shown in Fig. 17.



4 bit hybrid design and test results demonstrate the link with theory

Fig. 16 CAR90/150 Hybrid Circuit

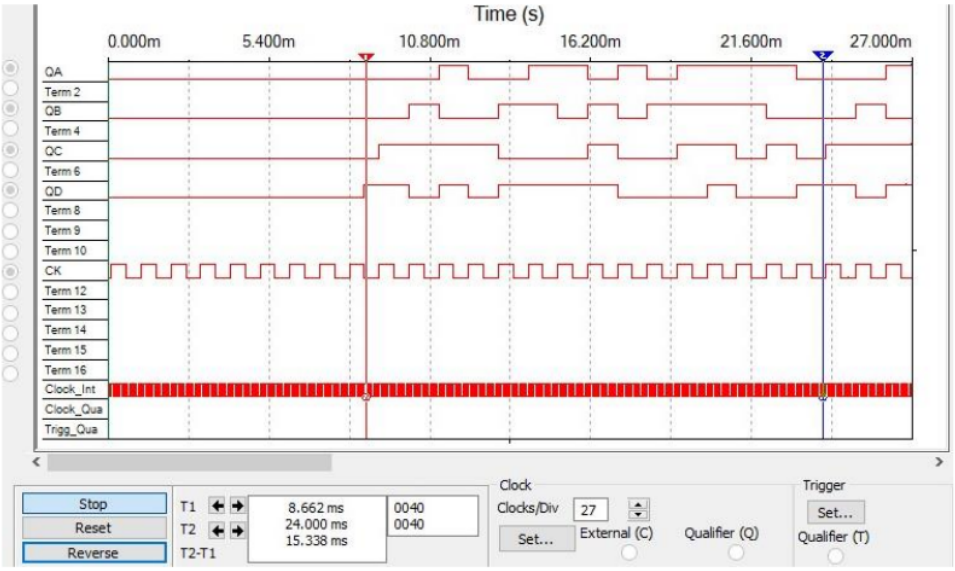


Fig. 17 Simulation Results for CAR90/150 Hybrid

The Table below summarizes and compares the expected and simulated results, which proves that the circuit is working properly without faults.

Expected	Simulated
1	1
3	3
6	6
11	11
2	2
5	5
13	13
9	9
7	7
8	8
4	4
14	14
15	15
12	12
10	10
1	1

Good comparison table of results

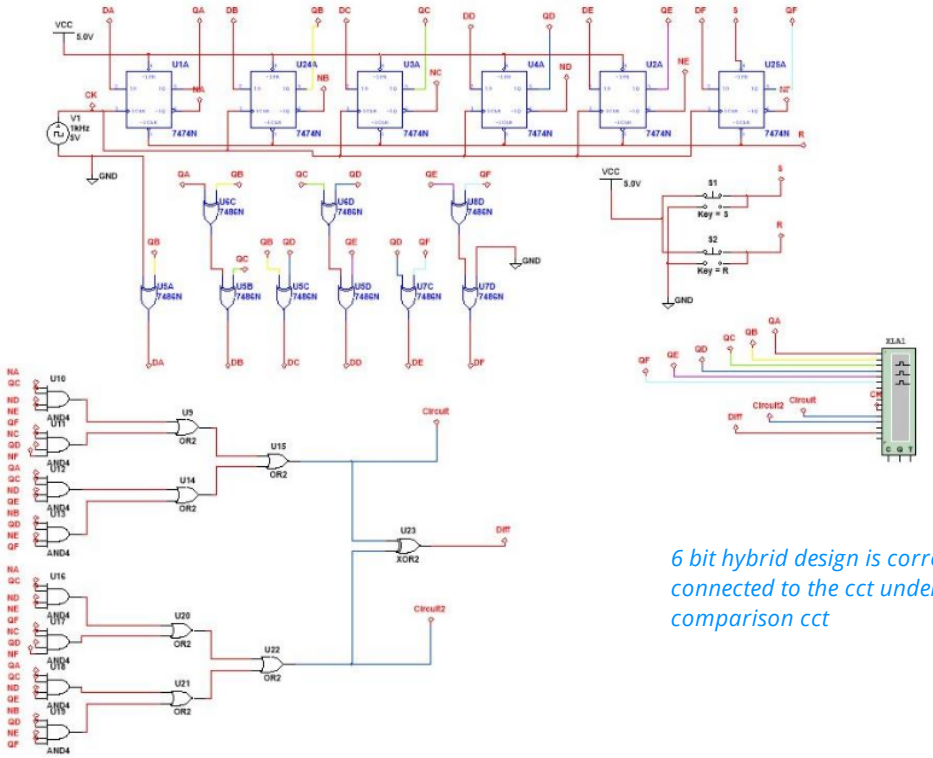
2.2 6-bit CAR90/150 Hybrid

The 6-bit CAR90/150 Hybrid works in the same way as the 4-bit circuit discussed in the previous section. Therefore, the output sequence can be predicted as shown below.

QA	QB	QC	QD	QE	QF	Decimal
0	0	0	0	0	1	1
0	0	0	0	1	1	3
0	0	0	1	1	0	6
0	0	1	0	1	1	11
0	1	0	0	1	0	18
1	1	1	1	0	1	61
1	1	0	0	0	1	49
1	0	1	0	1	1	43
0	0	0	0	1	0	2
0	0	0	1	0	1	5
0	0	1	1	0	1	13
0	1	1	0	0	1	25
1	0	1	1	1	1	47
0	0	0	1	1	1	7
0	0	1	0	0	0	8
0	1	0	1	0	0	20
1	1	0	1	1	0	54
1	0	0	0	1	1	35
0	1	0	1	1	0	22
1	1	0	0	1	1	51
0	0	1	1	1	1	15
0	1	1	1	0	0	28
1	0	0	0	1	0	34
0	1	0	1	0	1	21
1	1	0	1	0	1	53
1	0	0	1	0	1	37
0	1	1	1	0	1	29
1	0	0	0	0	1	33
0	1	0	0	1	1	19
1	1	1	1	1	0	62
1	1	0	1	1	1	55
1	0	0	0	0	0	32
0	1	0	0	0	0	16
1	1	1	0	0	0	56
1	1	1	1	0	0	60
1	1	0	0	1	0	50
1	0	1	1	0	1	45
0	0	1	0	0	1	9
0	1	0	1	1	1	23
1	1	0	0	0	0	48
1	0	1	0	0	0	40
0	0	0	1	0	0	4
0	0	1	1	1	0	14
0	1	1	1	1	1	31
1	0	0	1	0	0	36
0	1	1	1	1	0	30
1	0	0	1	1	1	39
0	1	1	0	0	0	24
1	0	1	1	0	0	44
0	0	1	0	1	0	10
0	1	0	0	0	1	17
1	1	1	0	1	1	59
1	1	1	0	1	0	58
1	1	1	0	0	1	57
1	1	1	1	1	1	63
1	1	0	1	0	0	52
1	0	0	1	1	0	38
0	1	1	0	1	1	27
1	0	1	0	1	0	42
0	0	0	0	0	1	1

Predicted result is identified

The circuit was then designed and tested. For this circuit a comparator has also been added to compare a good and a faulty circuit.



6 bit hybrid design is correct and connected to the cct under test and comparison cct

Fig. 18 6-bit CAR90/150 Circuit

Test showing no fault condition

Fig. 19 CAR90/150's result

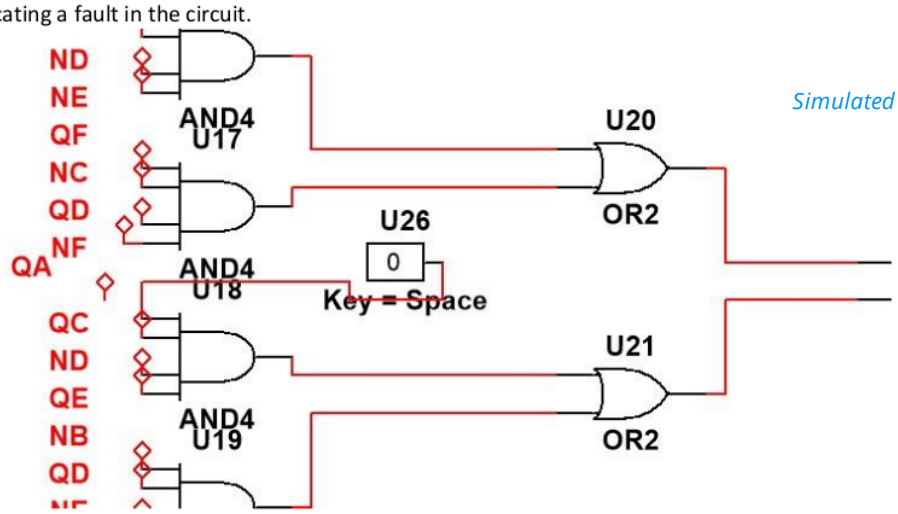
The above graph shows that the test bit remains at constant low, meaning that both circuits are identical. It also shows that the sequence is correct.

3. Fault simulation on MultiSim using 6-bit CAR90/150

For fault simulating, the CAR90/150 (Fig. 18) circuit is being used. The faults are generated by manually substituting a flip-flop's output by a constant value. That is done to only one of the circuits. In total three faults were simulated:

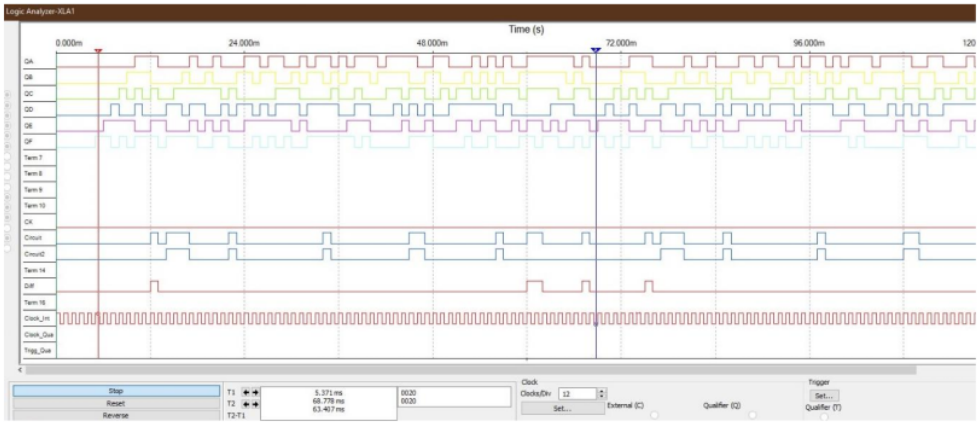
3.1 Fault QA SA 0

To simulate this fault, a constant 0 is connected instead of *QA*. The modified circuit is shown on Fig. 20. The simulation's result is shown Fig. 21 and it can be seen that the test bit does go high, correctly indicating a fault in the circuit.



Simulated fault is set

Fig. 20 QA SA 0

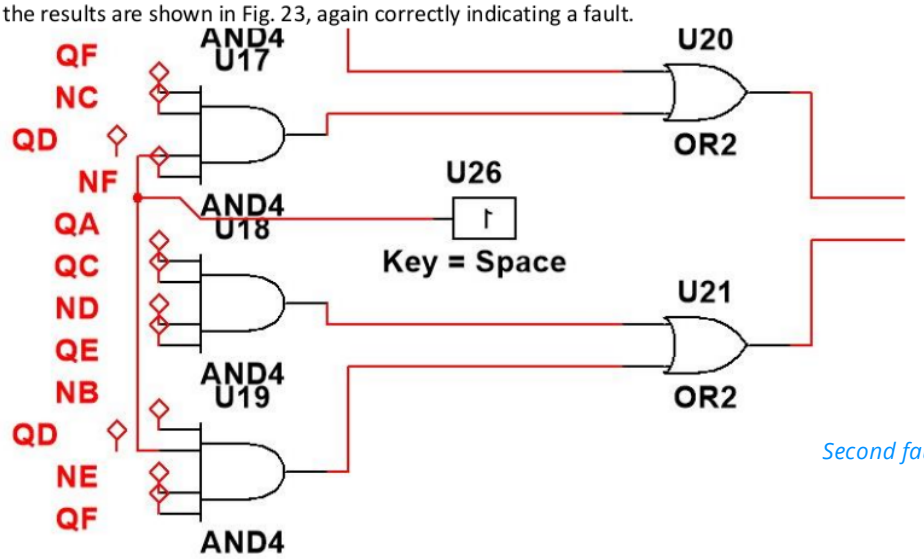


Test shows the fault is detected

Fig. 21 QA SA 0 Simulation results

3.2 Fault QD SA 1

This following fault simulates the bit QD being stuck at 1. The circuit modification is shown in Fig. 22. And the results are shown in Fig. 23, again correctly indicating a fault.



Second fault is set

Fig. 22 QD SA 1

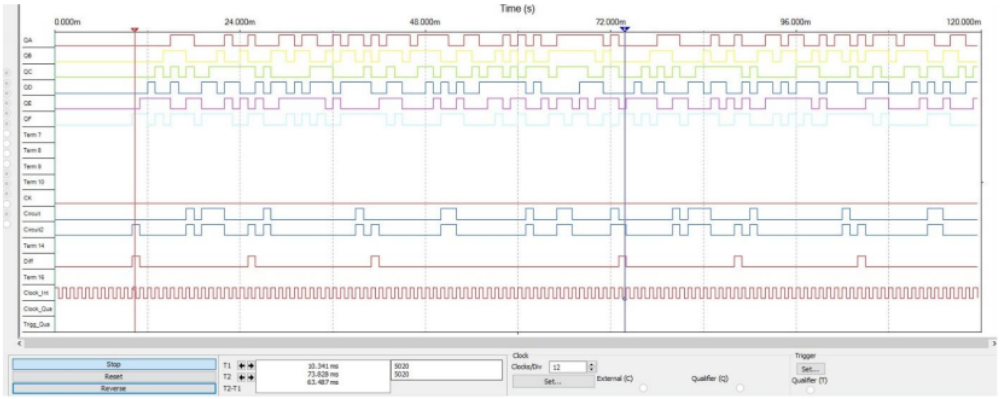
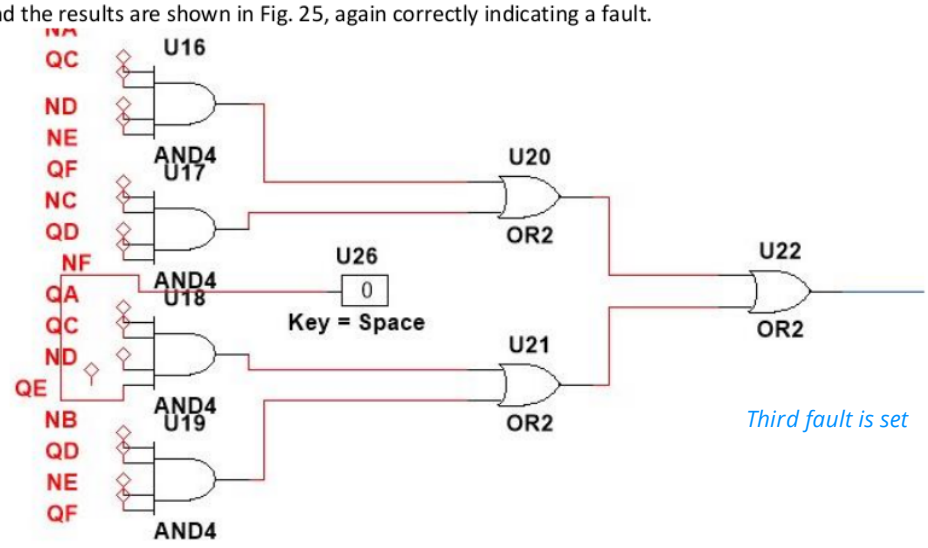


Fig. 23 Simulation results for QD SA 1

Test shows the fault is detected

3.3 Fault QE SA 0

This following fault simulates the bit QE being stuck at 0. The circuit modification is shown in Fig. 24. And the results are shown in Fig. 25, again correctly indicating a fault.



Third fault is set

Fig. 24 QE SA 0

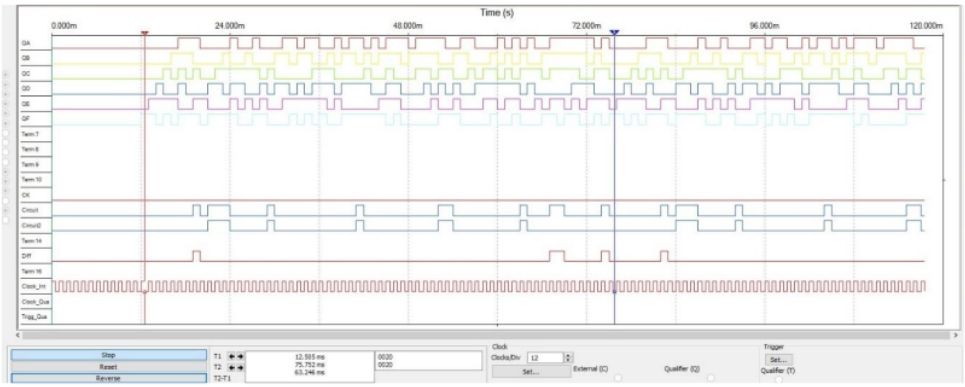


Fig. 25 QE SA 0 Simulation Results

Test has detected fault condition

4. VHDL implementation

Each of the previously designed circuits was now transcribed into a VHDL code. Each of the circuits was simulated to ensure its correct operation. The code required for the adequate inputs and outputs to be defined and its logical behaviour to be coded.

CAR 90

The code for CAR90 is:

```
entity cr90rule is
         port(
                 clr : in STD_LOGIC;
                 clk : in STD_LOGIC;
                 q0,q1,q2,q3 : out std_logic
             );
end cr90rule;

--SET BEHAVIOR SEQUENCE
architecture Behavioral of cr90rule is
signal D, qs: STD_LOGIC_VECTOR(3 downto 0);
begin
        --CAR 90 ALGORITHM
        D(3) <= '0' xor qs(2);
        D(2) <= qs(3) xor qs(1);
        D(1) <= qs(2) xor qs(0);
        D(0) <= qs(1) xor '0';

        -- SET CLK AND START SEQUENCE 0001 FOR 4 BIT
        process(clk, clr)
        begin
                if clr = '1' then
                        qs <= "0001";
                elsif clk'event and clk = '1' then
                        qs <= D;
                end if;
        end process;

--SET OUTPUTS TO ARRAY ELEMENTS
q0 <= qs(0);
q1 <= qs(1);
q2 <= qs(2);
q3 <= qs(3);
```

Entity is identified

Feedback algorithm is identified

Start sequence is set at 0001

Outputs are configured correctly with variables

The simulation is shown in Fig. 26. It shows that the count is correct.

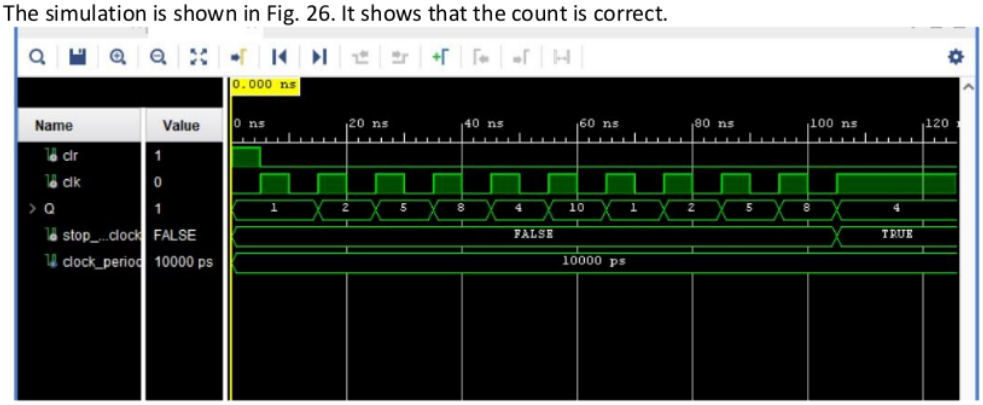


Fig. 26 Simulation of CAR90 in VHDL

Good test method using virtual bus. Test shows correct random sequence

CAR150

The code for CAR150 is as follows:

```
entity cr150_2 is
         port(
                 clr : in STD_LOGIC;
                 clk : in STD_LOGIC;
                 q0,q1,q2,q3 : out std_logic
             );
end cr150_2;
--SET BEHAVIOUR SEQUENCE
architecture Behavioral of cr150_2 is
signal D, qs: STD_LOGIC_VECTOR(3 downto 0);
begin
--SET CAR 150 ALGORITHM SEQUENCE
    D(3) <= '0' xor qs(2) xor qs(3);
        D(2) <= qs(3) xor qs(2) xor qs(1);
        D(1) <= qs(2) xor qs(1) xor qs(0);
        D(0) <= qs(1) xor qs(0) xor '0';

        -- SET CLK AND START SEQUENCE 0001 FOR 4 BIT
        process(clk, clr)
        begin
                if clr = '1' then
                        qs <= "0001";
                elsif clk'event and clk = '1' then
                        qs <= D;
                end if;
        end process;
--SET OUTPUTS TO ARRAY ELEMENTS
q0 <= qs(0);
q1 <= qs(1);
q2 <= qs(2);
q3 <= qs(3);
```

Entity is identified

Feedback algorithm is impelemnted

Start sequence is set

Outputs are configured

The code was simulated and the results are presented on Fig. 27. It proves the correct operation of the circuit.

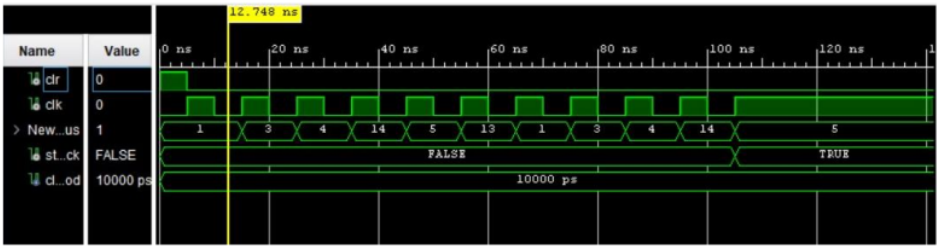


Fig. 27 Simulation results of the VHDL Code for CAR150

CAR90/150 4-bit

Good test method using V-bus. Test shows random count is incomplete

The code for CAR90/150 4-bit is as follows:

```
entity cr90_150rule is
         port(
                 clr : in STD_LOGIC;
                 clk : in STD_LOGIC;
                 q0,q1,q2,q3 : out std_logic
             );
end cr90_150rule;
--SET THE BEHAVIOUR SEQUENCE
architecture Behavioral of cr90_150rule is
signal D, qs: STD_LOGIC_VECTOR(3 downto 0);
begin
--SET THE HYBRID CAR 90/150 SEQUENCE ALGORITHM
        D(3) <= '0' xor qs(2);
        D(2) <= qs(3) xor qs(2) xor qs(1);
        D(1) <= qs(2) xor qs(0);
        D(0) <= qs(1) xor qs(0) xor '0';

        -- SET THE CLK AND START SEQUENCE USING 0001 FOR 4 BIT
        process(clk, clr)
        begin
```

Entity is identified

Feedback algorithm is implemented

```
                if clr = '1' then
                        qs <= "0001";
                elsif clk'event and clk = '1' then
                        qs <= D;
                end if;
        end process;
q0 <= qs(0);
q1 <= qs(1);
q2 <= qs(2);
q3 <= qs(3);

end Behavioral;
```

Start sequence is set at 0001

Outputs are configured

This code was executed and simulated and the results of the simulation are shown in Fig. 28. That proves that the circuit is working properly.

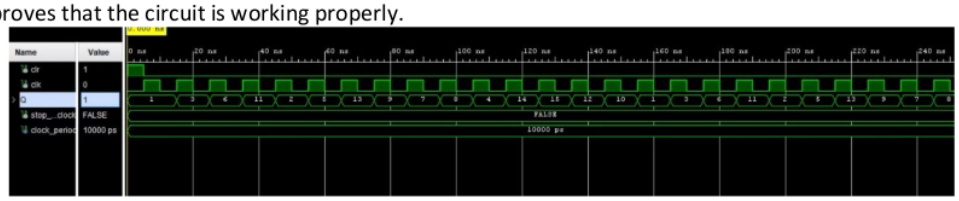


Fig. 28 Simulation result of the CAR90/150 4-bit code

Test showing full random count

CAR90/150 6-bit

The code for CAR90/150 6-bit is as follows:

```
entity hybrid_6_bit is
         port(
                 clr : in STD_LOGIC;
                 clk : in STD_LOGIC;
                 q0,q1,q2,q3,q4,q5 : out std_logic
             );
end hybrid_6_bit;

architecture Behavioral of hybrid_6_bit is
signal D, qs: STD_LOGIC_VECTOR(5 downto 0);
begin

D(5) <= '0' xor qs(4);
D(4) <= qs(5) xor qs(4) xor qs(3);
D(3) <= qs(4) xor qs(2);
D(2) <= qs(3) xor qs(2) xor qs(1);
D(1) <= qs(2) xor qs(0);
D(0) <= qs(1) xor qs(0) xor '0';

        -- Three D flip-flops
        process(clk, clr)
        begin
                if clr = '1' then
                        qs <= "000001";
                elsif clk'event and clk = '1' then
                        qs <= D;
                end if;
        end process;
q0 <= qs(0);
q1 <= qs(1);
q2 <= qs(2);
q3 <= qs(3);
q4 <= qs(4);
q5 <= qs(5);

end Behavioral;
```

Entity is showing the expansion elements

Feedback algorithm shows the extended elements for 6 bit design

Start sequence is set at 000001

Outputs are extended

This code was simulated and tested, the results of the simulation can be seen in Fig. 29 which proves the correct operation of the circuit. The graph is rolled on four times and each following part follows the sequence from top to bottom.

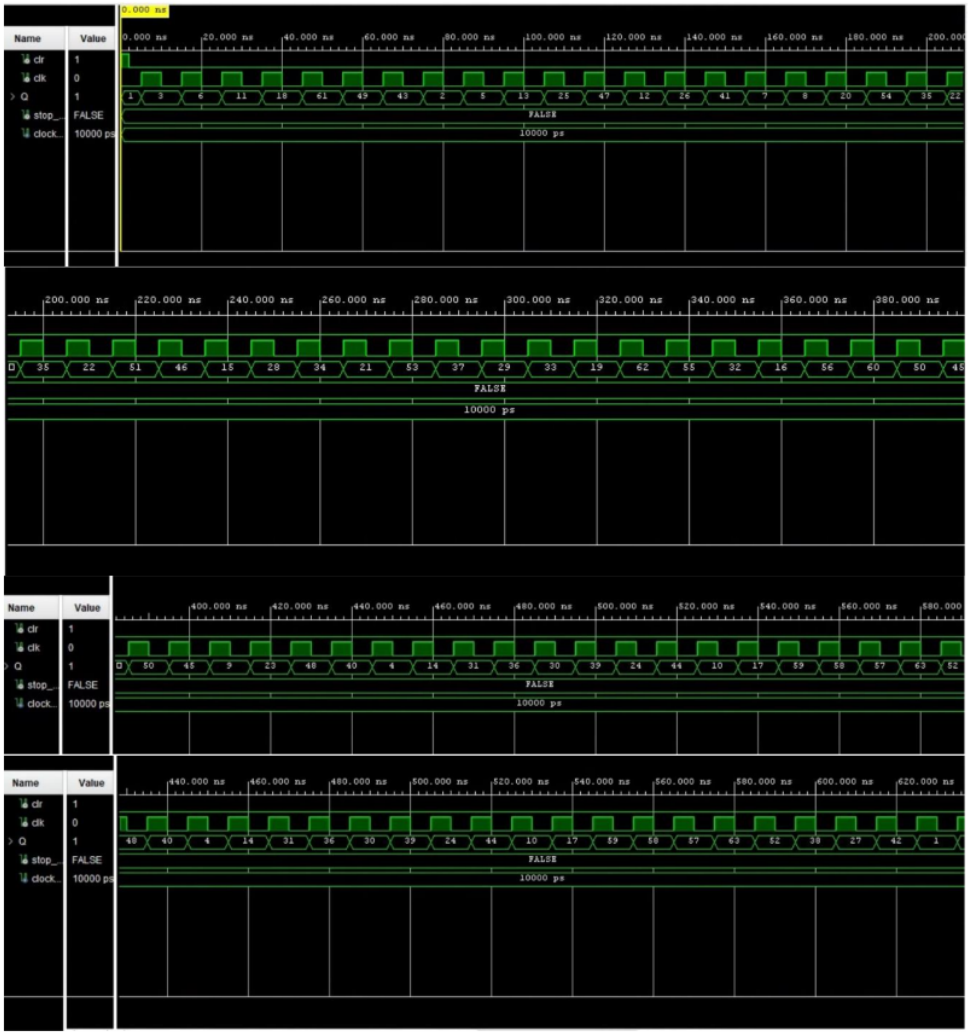


Fig. 29 Simulation results of the 6-bit CAR90/150 (Broken in four parts)

A good set of results showing all count elements

4.2 IPs

Following the IPs were created, tested and combined to build the final circuit.
First, the clock speed was slowed down from 100MHz to 5kHz which is easier to work with. To do so, an IP block has been created and it is shown in Fig. 30.

Fig. 30 Clock speed reduction block

Good hybrid design of clk

Following, the 6-bit CAR90/150 IP block was created. It was configured according to the schematic on
.

IP for 6 bit hybrid is good and shows all io connections

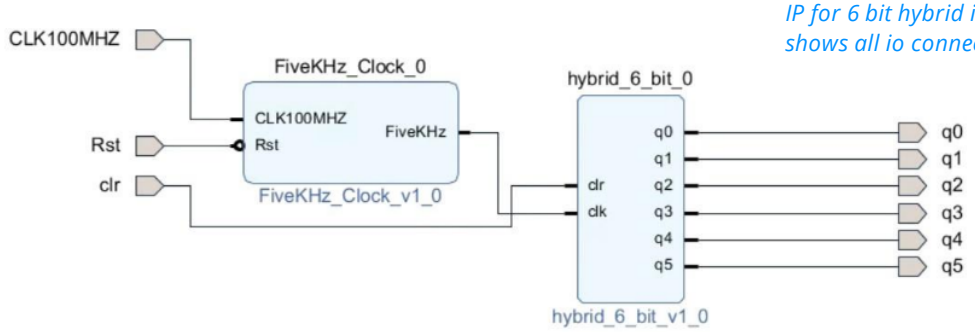
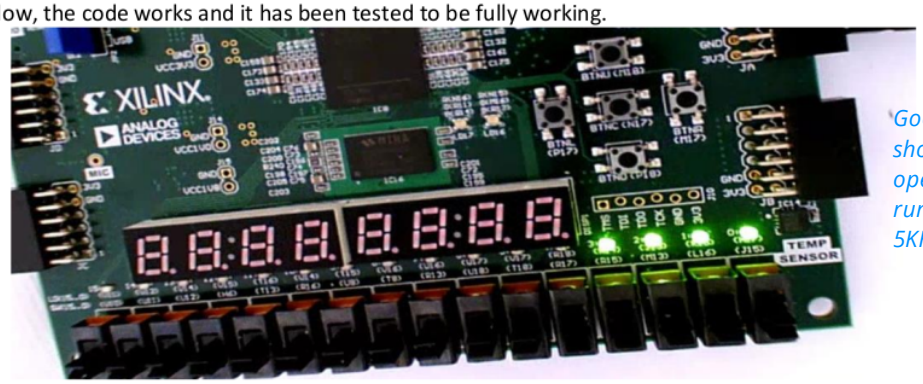


Fig. 31 6-bit CAR90/150 IP block testing.

To ensure its correct operation, the test circuit was downloaded on to a prototype board. As shown below, the code works and it has been tested to be fully working.



Good practical test showing the operation of the 6 bit running with the 5KHZ clk

Fig. 32 practical test of the VHDL IP design

Confirmed that it was working, the IP design was developed to be able to compare circuits for fault detection. The schematic from Fig. 33 was followed.

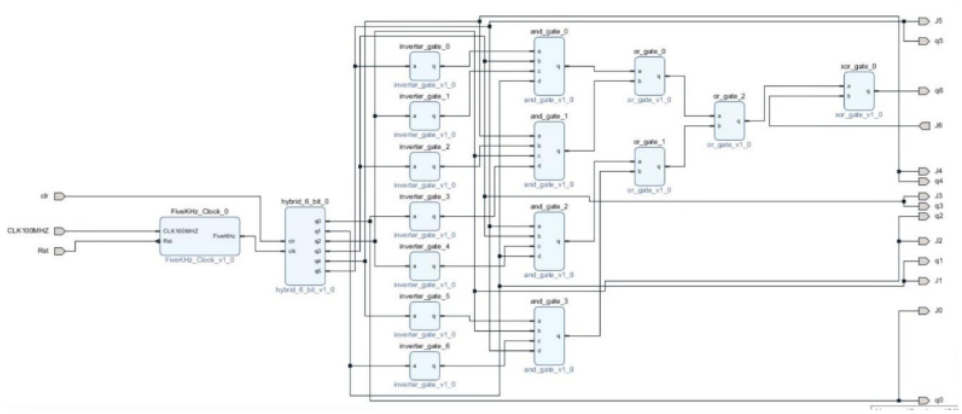


Fig. 33 Final IP design

Final design is correct and all components are in place.

5. Practical experiment with fault simulation.

Now the IP design was downloaded to the prototype board and fault-finding test board (designed in previous assessment) as shown in Fig. 34. The faults were simulated on the hardware with physical cables. The faults simulated are: QD SA 0, QD SA 1, QB SA 1Fig. 33

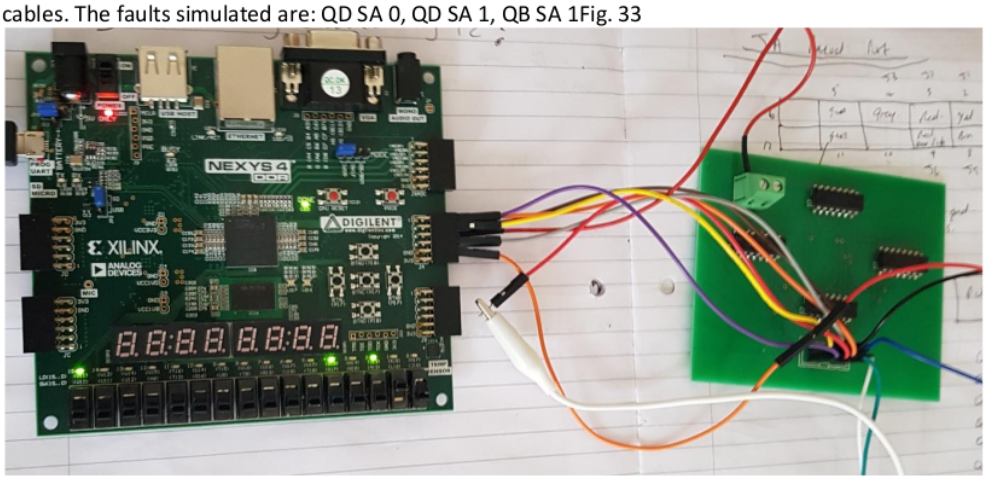


Fig. 34 The practical set up

Good evidence of practical setup

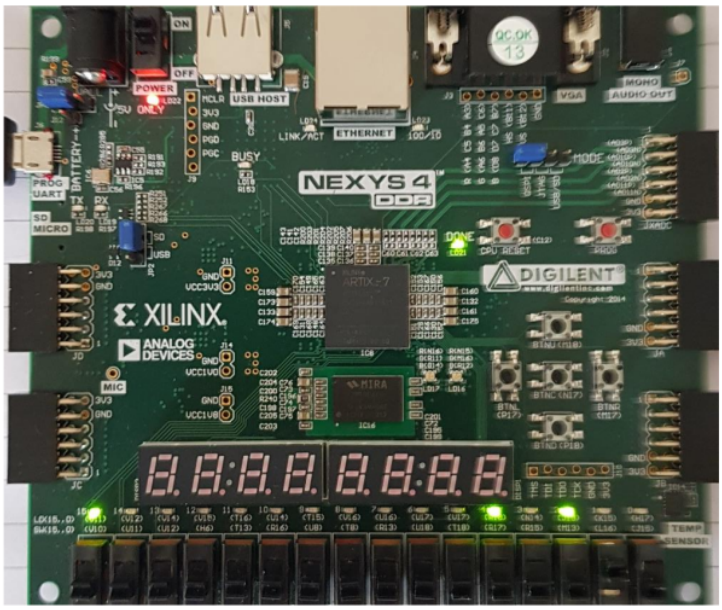


Fig. 35 QD SA 0 fault simulation

Test shows fault condition is detected

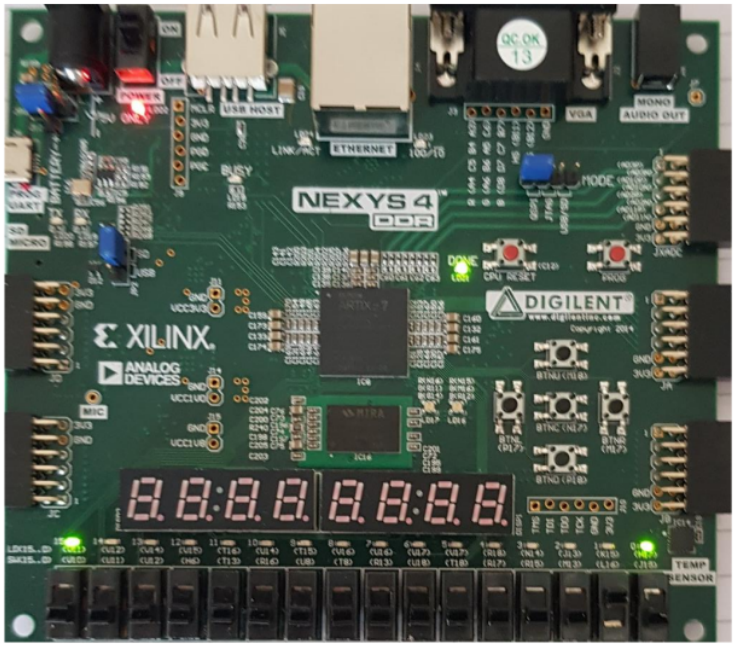


Fig. 36 QD SA 1 simulation result

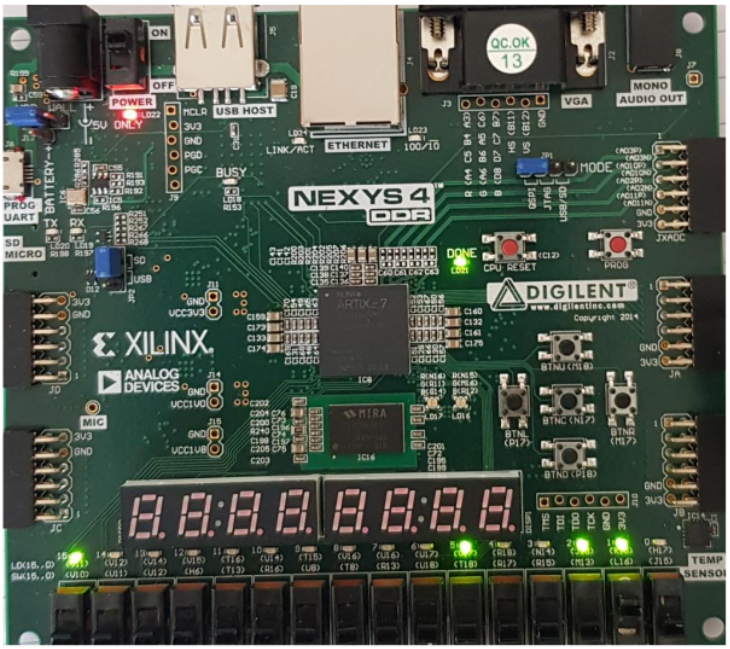


Fig. 37 QB SA 1 Simulation result

All fault conditions are detected

As the test results shown in Fig. 35Fig. 37 prove that the circuit designed is capable of detecting a fault, means that the circuit is operational.

A fault cleared as a condition would have been useful as a test

6. Conclusions.

The research into image processing shows how capable the contemporary systems are. It reflects the benefits of image processing to the industry and with the constantly advancing technology it can be used for process and quality controls.
The simulations and practical experiments of the CAR circuits have proven the theory touched. They also proven that the sequences are the same in MultiSim, VHDL and in practice.

Conclusion is a little brief and there are many parts that should be concluded with recommendations if possible.

References

A. Wilson, 2016. Applying Pattern Matching Software in Machine Vision Applications
https://www.vision-systems.com/embedded/article/16736836/applying-pattern-matching-software-in-machine-vision-applications
[Accessed on: 18th March 2021]

NI Zone, nd. What to Expect from a Pattern Matching Tool.
https://zone.ni.com/reference/en-XX/help/372916T-01/nivisionconcepts/what_to_expect_from_a_pattern_matching_tool/
[Accessed on: 18th March 2021]

Rasika R Swati A, G. D. (2020, 04 03). *quality control of PCB using image processing*. https://www.researchgate.net/publication/303318951_Quality_Control_of_PCB_using_Image_Processing
[Accessed on: 18th March 2021]

NI Vision Concepts, 2000. IMAQ Vision Concepts Manual.
Archived: IMAQ Vision Concepts Manual - National Instruments (ni.com)
[Accessed on: 19th March 2021]

K.S.Htoon, 2020. A Tutorial o Histogram Equalisation.
https://medium.com/@kyawsawhtoon/a-tutorial-to-histogram-equalization-497600f270e2
[Accessed on: 19th March 2021]

NI IMAQ, 2004. Vision Assistant Tutorial
https://neurophysics.ucsd.edu/Manuals/National%20Instruments/NI%20Vision%20Assistant%20Tutorial.pdf
[Accessed on: 19th March 2021]

A good number of relevant sources

1708165

GRADEMARK REPORT

FINAL GRADE

71 /100

GENERAL COMMENTS

Instructor

PAGE 1

PAGE 2

Text Comment. The VHDL implementation and testing has many parts which should be in the table of contents

PAGE 3

Text Comment. Good list of figures

PAGE 4

Text Comment. Good example with supporting source

Text Comment. This is a good method identified here. Probably the most useful for PCB testing

PAGE 5

Text Comment. A good significant point made here

Text Comment. This is the most important point about colour matching and it is identified

Text Comment. Very good explanation of this method

PAGE 6

Text Comment. Excellent work on this area

Text Comment. Histogram purpose is defined

Text Comment. Good image to support explanation

PAGE 7

Text Comment. A very good example for this process an its benefits

Text Comment. Very good example of colour extraction which is a common process in image processing for speed.

Text Comment. This is a significant point to make as the medical industry uses the same processing methods only with much more sophisticated sensor equipment.

PAGE 8

Text Comment. Good image to support research

PAGE 9

Text Comment. Constraints have been identified using the theory model for feedback

Text Comment. Good explanation with supporting table of expected results

PAGE 10

Text Comment. A good design of the CAR 90 and test showing the incomplete random count

Text Comment. Good table of results showing comparison to theory

PAGE 11

Text Comment. Constraints for the CAR 150 are identified using the feedback model

Text Comment. Design circuit is correct and a good test result showing the link with theory

PAGE 12

Text Comment. Comparison showing the link with theory

PAGE 13

Text Comment. 4 bit hybrid design is identified with predicted test results

PAGE 14

Text Comment. 4 bit hybrid design and test results demonstrate the link with theory

PAGE 15

Text Comment. Good comparison table of results

PAGE 16

Text Comment. Predicted result is identified

PAGE 17

Text Comment. 6 bit hybrid design is correct and connected to the cct under test and comparison cct

Text Comment. Test showing no fault condition

PAGE 18

Text Comment. Simulated fault is set

Text Comment. Test shows the fault is detected

PAGE 19

Text Comment. Second fault is set

Text Comment. Test shows the fault is detected

PAGE 20

Text Comment. Third fault is set

Text Comment. Test has detected fault condition

PAGE 21

Text Comment. Entity is identified

Text Comment. Feedback algorithm is identified

Text Comment. Start sequence is set at 0001

Text Comment. Outputs are configured correctly with variables

Text Comment. Good test method using virtual bus. Test shows correct random sequence

PAGE 22

Text Comment. Entity is identified

Text Comment. Feedback algorithm is impelemnted

Text Comment. Start sequence is set

Text Comment. Outputs are configured

Text Comment. Good test method using V-bus. Test shows random count is incomplete

Text Comment. Entity is identified

Text Comment. Feedback algorithm is implemented

PAGE 23

Text Comment. Start sequence is set at 0001

Text Comment. Outputs are configured

Text Comment. Test showing full random count

Text Comment. Entity is showing the expansion elements

Text Comment. Feedback algorithm shows the extended elements for 6 bit design

Text Comment. Start sequence is set at 000001

Text Comment. Outputs are extended

PAGE 24

Text Comment. A good set of results showing all count elements

Text Comment. Good hybrid design of clk

PAGE 25

Text Comment. IP for 6 bit hybrid is good and shows all io connections

Text Comment. Good practical test showing the operation of the 6 bit running with the 5KHZ clk

Text Comment. Final design is correct and all components are in place.

PAGE 26

Text Comment. Good evidence of practical setup

Text Comment. Test shows fault condition is detected

PAGE 27

Text Comment. All fault conditions are detected

Text Comment. A fault cleared as a condition would have been useful as a test

PAGE 28

Text Comment. Conclusion is a little brief and there are many parts that should be concluded with recommendations if possible.

PAGE 29

Text Comment. A good number of relevant sources

GRADING FORM: CSG-RUBIC

ADAM BEDNER	**0**

RESEARCH-INFORMED LITERATURE

Machine vision and pattern matching have been covered in good detail with good relevant sources used in the text. 8/10 Colour recognition researched the possibilities of colour depth recognition and use a good example to support this. The array has been explained in good detail. Good relevant sources used. 8/10

16/20

KNOWLEDGE AND UNDERSTANDING

Design of Msim 6 bit CAR using the prototype method with each stage tested proved to be a good method and the resulting design was proven in the test stage. The VHDL design followed the same pattern to create the final 6 bit design. The system design tests proved the system. 7/10 IP design using the FPGA development board implemented the IP components derived from the VHDL stage of development. The IP gate designs were created using the same method. These were not covered in report due to simplicity, but there should be an example in the appendix. Also the constraints file should have been explained 7/10

14/20

ANALYSIS

Test results analysis of 6 bit hybrid CAR showed that the random and maximum count could be achieved only by combining the two rules 90 and 150. The theory showed that this was the case and

15/20

was proved through testing carried out. The explanations are fairly accurate about this. 6.5/8 The analysis of the FPGA results show operational conditions conditions. 6/8 Analysis of PCB test results has covered the fault set and cleared conditions as required. 2.5/4

PRACTICAL APPLICATION AND DEP.

CAD implementation and testing of prototype designs has been carried out using a good sequential method and enabled the final hybrid design to be implemented. The test procedures showed how the single rule systems could be implemented as a hybrid and used to drive a test cct. 8/10 FPGA implementation required the design of SOC components for an IP system design. This has been carried out firstly with errors and through test and analysis the errors were resolved to give a good working end product that was used with the practical testing. 10/15

18/25

SKILLS FOR PROFESSIONAL PRAC.

Deployment of the software and on line resources has been carried out to a very good standard. 4/5 Report layout is good and all parts of the report are in the correct sequence. 3/5 Conclusion is poor. 1/5

8/15