

1302736

by Jason Booth

Submission date: 28-Feb-2023 10:22PM (UTC+0000)

Submission ID: 198183831

File name: 19735_Jason_Booth_1302736_181628_483815091.pdf (3.17M)

Word count: 7680

Character count: 44153

Course: HND Electrical and Electronic Engineering

Module: SGEE5007 Virtual Instrumentation and Control

Portfolio: Magnitude Comparator

Name: Jason Booth

University number: 1302736

Date submitted: 28/02/2023

Contents

Element 1	6
1. Introduction	6
2. Research.....	7
2.1 Digital Design Methods using VHDL.....	7
2.2 NI DAQmx.....	8
2.3 Digital Magnitude Comparator	11
2.4 Analogue to Digital Converters	12
2.5 Load Cells (LCKD-1Kg)	13
2.6 Instrumentation and Signal Conditioning	14
3. Design Requirements.....	16
3.1 4-bit Adder.....	16
3.2 VHDL Program.....	16
3.3 Simulation analysis using virtual bus method	17
3.4 I/P design for 4-bit adder	18
3.5 4-bit magnitude Comparator	18
3.6 4-bit adder & 4-bit mag comp combination	19
3.7 4bit mag comparator.....	20
4. Implementation.....	22
5. Test & Analysis.....	24
5.1 4bit Mag Comparator FPGA Testing.....	25
5.2 8bit Adder & Comparator FPGA Testing.....	26
6. Evaluation	28
Element 2.....	29
1. Introduction	29
2. Design Requirements.....	30
2.1 Unity Gain of Instrument Amplifier	30
2.2 Instrument Amplifier with Bridge for Load Cell Sensor.....	30
1.3 Testing.....	32

2. Prototypes.....	33
3.1 10- and 12-bit ADC.....	33
3. Digital Magnitude Comparator.....	41
4.1 Nexys 4 DDR FPGA Development Board	41
4.2 Magnitude Comparator Setup.....	42
3.3 LabView Program	43
3.4 Test Results.....	44
4. Evaluation.....	46
Element 3.....	47
1. Introduction	47
2. Implementation.....	48
3. Testing and Analysis	50
4. Conclusion and Recommendations	51
References	53
Appendix.....	55
A1 – 4bit adder architecture.....	55
A2 – FPGA Input/outputs.....	55
A3 – Load cell tests	56
A4 – 4bit adder testbench simulation.....	58
A5 – 4bit Magnitude Comparator testbench simulation.....	59
A6 – Data logs for load cell graph.....	60

Good detailed table of contents

Figure 1 - VHDL description example.....	7
Figure 2 - Design process & methodology of VHDL	7
Figure 3 - DAQ system.....	8
Figure 4 - While loop	9
Figure 5 - Read data	9
Figure 6 - Write data	9
Figure 7 - DAQmx start task.....	10
Figure 8 - Input channel	10
Figure 9 - Formula block	10
Figure 10 - Digital comparator	11
Figure 11 - 1bit comparator block diagram	11
Figure 12 - 1bit comparator truth table	11
Figure 13 - Flash ADC architecture. If the analog input is between VX4 and VX5, comparators X1 through X4 produce 1s and the remaining comparators produce 0s. (Maxim Integrated Tutorial 810, 2013)	13
Figure 14 - Load Cell.....	14
Figure 15 - Wheatstone bridge (Delta Motion, 2016)	14
Figure 16 - Signal conditioning types (TT Electronics, 2016)	15
Figure 17 - 4bit adder logic diagram.....	16
Figure 18 - 4bit adder hierarchy	16
Figure 19 - 4bit adder simulation	17
Figure 20 - 4bit adder code	18
Figure 21 - 4bit magnitude comparator function block.....	18
Figure 22 - 4bit adder + comparator.....	19
Figure 23 - 8bit mag comparator (equal to)	20
Figure 24 - 8bit mag comparator (less than).....	21
Figure 25 - 8bit mag comparator (greater than).....	21
Figure 26 - IP Design for 4bit adder	22
Figure 27 - 8-bit PLD design	23
Figure 28 - 4bit adder simulation.....	24
Figure 29 - 4bit mag comparator simulation	24
Figure 30 - FPGA 4bit mag comparator test 1	25
Figure 31 - FPGA 4bit mag comparator test 2	25
Figure 32 - FPGA 4bit mag comparator test 3	26
Figure 33 - FPGA 8bit adder & comparator test 1	26
Figure 34 - FPGA 8bit adder & comparator test 2	27
Figure 35 - FPGA 8bit adder & comparator test 3	27

Figure 36 - Instrumentation amplifier with unity gain of 1	30
Figure 37 - Load cell	31
Figure 38 - Instrument amplifier with bridge	32
Figure 39 - Load cell response graph.....	33
Figure 40 - 10 bit ADC system using LabVIEW	34
Figure 41 - Formula block	34
Figure 42 - LabVIEW interface page	35
Figure 43 – 10 bit Value at full scale	35
Figure 44 - 12 bit ADC system using LabVIEW	36
Figure 45 - Formula block	36
Figure 46 - 12 bit Value at full scale	37
Figure 47 - Load cell test 1.....	39
Figure 48 - Load cell test 2.....	39
Figure 49 - Load cell test 3.....	40
Figure 50 - Nexys 4 DDR Connections.....	41
Figure 51 - NI ELVIS II Magnitude Comparator Set Up	42
Figure 52 - FPGA to ELVIS II MYDAQmx connections.....	42
Figure 53 - Block Diagram.....	43
Figure 54 - Panel View.....	44
Figure 55 - Test 1 equals to	45
Figure 56 - Test 2 Less than	45
Figure 57 - Test 3 Greater than.....	46
Figure 58 - Block diagram with added file write blocks	48
Figure 59 - File1 write to file block.....	49
Figure 60 - Data log graph results.....	50
 Table 1 - Load cell results	 33
Table 2 – Test Results	46

Comprehensive list of figures

The use of tables to present test results is a preferred method

Element 1

1. Introduction

The aim of this section is to carry out research on research into specific areas that are relating to each of the elements in the portfolio. The areas researched are:

- Digital design methods using VHDL and IP integration.
- NI DAQmx data acquisition used in LabVIEW.
- Digital Magnitude Comparators.
- Analogue to Digital Converters.
- Load Cells (namely LCKD-1Kg)
- Instrumentation and signal conditioning.

Further into this section will include design and implementation of combinational logic system using VHDL and IP integration and implement using the Nexys 4 DDR FPGA development board. Design and simulation will be carried out on an 8-bit magnitude comparator in MultiSim PLD mode. Analysis of three sets of test results will be made and conclusions drawn.

Introduction indicates an understanding of the requirements of the element

2. Research

2.1 Digital Design Methods using VHDL

VHDL is a hardware description language (HDL). It describes digital systems and circuits. A system can be described in VHDL then simulated of its operation through using a VHDL simulator. The circuit can also be synthesized by using a synthesis program that converts the descriptive language (VHDL) to a gate-level description to map to an ASIC or FPGA. The two HDLs that are widely used today are VHDL and Verilog. Most chips and systems currently in industry are designed by one of the two languages. (Dally, et al., 2015)

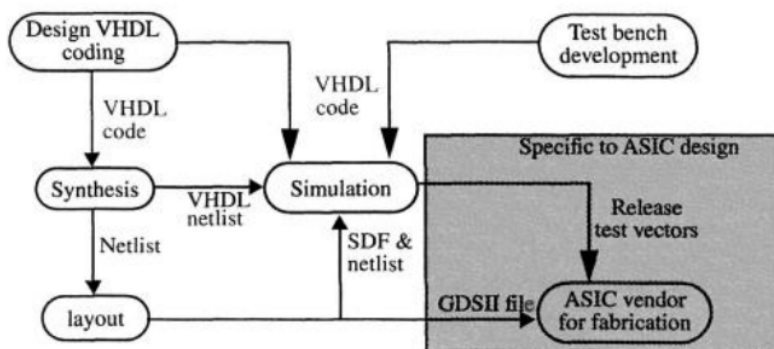
Good points made and supported with relevant source

```
library ieee;  
use ieee.std_logic_1164.all;  
  
entity THERMOSTAT is  
  port ( presetTemp, currentTemp : in std_logic_vector( 2 downto 0 );  
        fanOn : out std_logic );  
end thermostat;  
  
architecture impl of Thermostat is  
begin  
  fanOn <= '1' when currentTemp > presetTemp  
           else '0';  
end impl;
```

Good example. This should be referenced in the text content, and significant parts i.e. entity and architecture should be explained.

Figure 1 - VHDL description example

Figure 1 above shows the description of VHDL coding. In this example, the coding/description is for a thermostat. It should be noted that the VHDL in figure 1 demonstrates the behaviour of the circuit, not how the circuit is implemented. (Dally, et al., 2015)



Good example

Image should have a ref source

Figure 2 - Design process & methodology of VHDL

Figure 2 shows a design process & methodology flow chart for VHDL systems. The 1st stage is the design which is described in VHDL. After each VHDL code is written, preliminary checks

are made by synthesis that checks for any design errors such as insufficient sensitivity list and unintended latches. A test bench is developed to facilitate the design verification. (Chang, 1999)

Chang (1999), says that all VHDL designs can be verified with VHDL simulations and synthesis tools to ensure high fidelity of the VHDL code.

Significant points described about VHDL. However, the IP design has not been explained

According to Charles H. Roth & John (2008), VHDL is very useful in teaching top-down design.

2.2 NI DAQmx

NI DAQmx are software drivers used for data acquisition that can be used in LabView which can be used to measure information coming from analogue inputs on a device such as the NI ELVIS. (MathWorks, 2020)

According to MathWorks (2020), "Data Acquisition Toolbox™ Support Package for National Instruments® NI-DAQmx Devices provides support for CompactDAQ, FieldDAQ™, X-Series, M-Series, E-Series, USB, myDAQ, ELVIS II, and many other types of data acquisition hardware from National Instruments through MATLAB® and Data Acquisition Toolbox."

Data acquisition (DAQ) is the process of using a computer to measure an readings such as voltage, current, temperature, pressure or sound. The make-up of a DAQ system consists of DAQ measurement hardware, sensors, and a computer with programmable software.



Good informative research carried out and relevant supporting image. The image should have a ref source included

Figure 3 - DAQ system

Figure 3 above shows how a DAQ system works. The sensor converts the measurement into a useable electrical signal. This could be voltage or current. The DAQ device acts as an interface between a computer and signals from the outside world. It digitizes the analogue signals coming in so that the computer can understand them.

There are many tools/icons used in LabView which are the DAQ family of communication blocks. These are explained below.

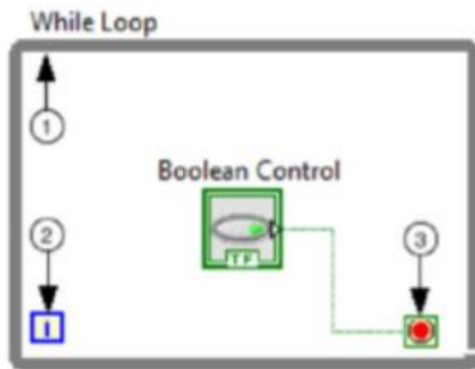


Figure 4 - While loop

While loop enables the program to execute tasks within the box of the loop. It runs continuously until specific conditions are met.

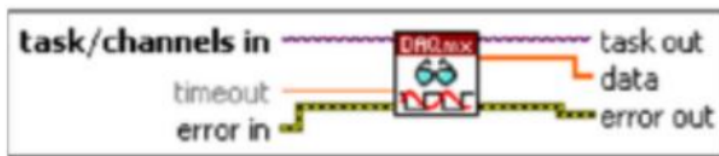


Figure 5 - Read data

The above icon block reads the incoming data and can be set to read either digital or analog signals.

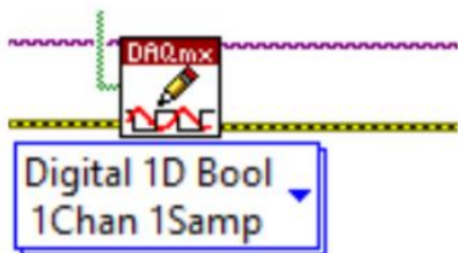


Figure 6 - Write data

The above figure shows the icon for write data. It is used to a digital output to a device that is external.



Diagram illustrating the Formula 2 block in a spreadsheet model. The block is labeled "Formula 2" and contains a calculator icon. It has an input labeled "X1" and an output labeled "Result". The block is annotated with "error in (no error)" and "error out".

The significant LabVIEW blocks are identified and explained

2.3 Digital Magnitude Comparator

Digital magnitude comparator is a combinational circuit that compares two binary numbers and determines their relative magnitudes to establish whether it is greater than, less than or equal to the other binary number. They are needed in digital systems while performing arithmetic or logical operations. (Electronics Hub, 2015)

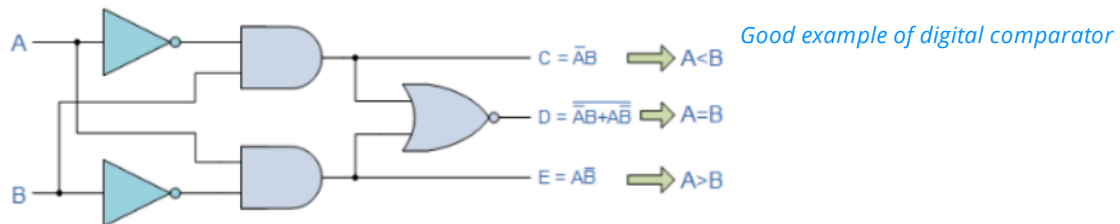


Figure 10 - Digital comparator

Figure 10 above shows the construction of digital comparator using NOR, NOT and AND gates to compare the signals. (Electronics Tutorial, 2014)



Figure 11 - 1bit comparator block diagram

Figure 11 above shows the block diagram for a 1-bit magnitude comparator. It compares the two bits and produces one of three states, less than, greater than, equal to. The truth table for this can be seen in figure 12 below. (Electronics Hub, 2015)

A_0	B_0	L	E	G
0	0	0	1	0
0	1	1	0	0
1	0	0	0	1
1	1	0	1	0

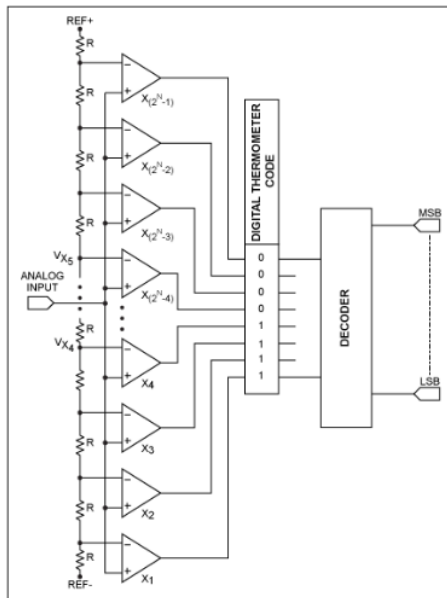
An understanding is presented, however, the expansion possibilities for this could have been explained

Figure 12 - 1bit comparator truth table

2.4 Analogue to Digital Converters

Flash ADCs, which are also known as parallel ADCs, provide the fastest method to convert an analog signal to digital signal. They are suitably used in applications that take up a lot of bandwidth. They have relatively low resolution and consume a lot of power and are quite expensive. They are limited to high-frequency applications that are unable to be addressed by other methods. Examples where they are used are in data acquisition, radar processing, sampling oscilloscopes, satellite comms, and high-density disk drives. (Maxim Integrated Tutorial 810, 2013) [Good explanation with relevant source](#)

Maxim Integrated Tutorial 810 (2013) states that, "flash ADCs are made by cascading high-speed comparators. Figure 13 shows a typical ADC block diagram." They illustrate that for an N-bit converter, the circuit employs $2^N - 1$ comparators. The reference voltage is provided by a resistive-divider with 2^N resistors. Each comparator has a ref voltage that is one least significant bit greater than the ref voltage for the comparator below it. When the comparator's analog input voltage is higher than the ref voltage applied, the comparator produces a 1, if not then it outputs 0. If the analog input is between V_{X4} and V_{X5} , comparators X1 through X4 output 1 and the remaining comparators output 0. The crossover where the code swaps from 1 to 0 is when the input signal is smaller than the respective comparator ref voltage level.



The alternative methods of ADC are not explained

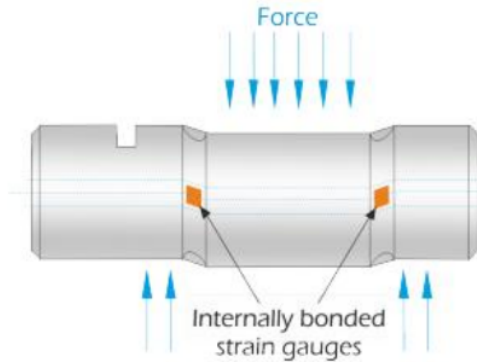
Figure 13 - Flash ADC architecture. If the analog input is between V_{X4} and V_{X5} , comparators X_1 through X_4 produce 1s and the remaining comparators produce 0s. (Maxim Integrated Tutorial 810, 2013)

Figure 13 shows the architecture for thermometer code encoding. The thermometer signal is decoded to the appropriate digital output code. (Maxim Integrated Tutorial 810, 2013)

2.5 Load Cells (LCKD-1Kg)

A load cell is a type of transducer that converts force into an electrical signal. The change in signal is proportional to change in force. (Force Logic, 2016)

Within the load cell is an area that is designed to be stressed when a force is applied. Strain gauges are bonded to these areas that senses the force being applied to the area under load pressure and causes a change in resistance which acts as an electrical output signal. (Anon., 2023)



Good explanation with supporting image

Figure 14 - Load Cell

According to DeltaMotion (2016), “a load cell consists of strain gauges in a Wheatstone configuration that are bonded to areas that are designed to bend as a load is applied.”

2.6 Instrumentation and Signal Conditioning

Strain gauges produce a change in resistance based on the load and displacement applied. A change in resistance is not a useable signal, therefore it needs to be converted into a useable signal by using a signal conditioner. (OMEGA, 2015)

A method to accurately measure a small change in resistance from the strain gauge is to use a Wheatstone bridge. The bridge consists of two voltage dividers wired in parallel, with a common voltage source, as seen in figure 15 below. (Delta Motion, 2016)

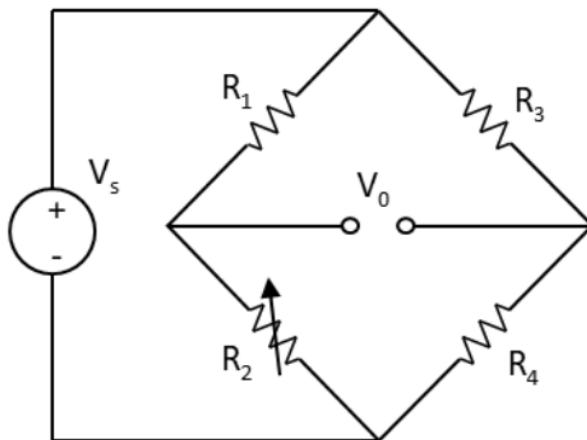


Figure 15 - Wheatstone bridge (Delta Motion, 2016)

According to TT Electronics (2016), a signal conditioner is a device that converts an electrical signal into a different signal (or useable signal). Its primarily to convert a signal that is difficult to read by conventional instruments into an easier readable signal format such as 4-20mA or 0-10 volts.

Signal conditioning applications are idenfified, and indicate an understanding of the practical application

Signal Conditioning Types						
	Amplification	Attenuation	Isolation	Filtering	Excitation	Linearization
Thermocouple	✓		✓	✓		✓
Thermistor	✓		✓	✓	✓	✓
RTD	✓		✓	✓	✓	✓
Strain Gage	✓		✓	✓	✓	✓
Load, Pressure, Torque	✓		✓	✓	✓	✓
Accelerometer	✓		✓	✓	✓	✓
Microphone	✓			✓	✓	✓
LVDT/RVDT	✓		✓	✓	✓	✓
High Voltage		✓	✓			

Good table of examples

Figure 16 - Signal conditioning types (TT Electronics, 2016)

Figure 16 above shows various types of signal conditioning and what type of conditioning is used.

The instrument amplifier purpose could have been explained further due to the relevance in this element

3. Design Requirements

3.1 4-bit Adder

Significant block diagram is identified with all the critical io elements needed for design

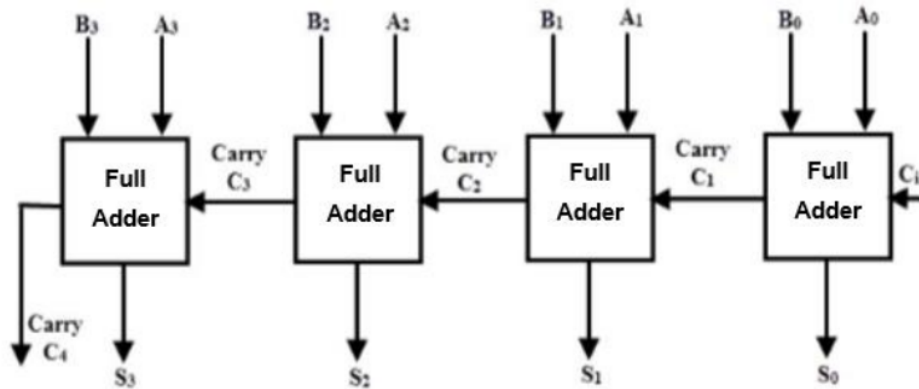
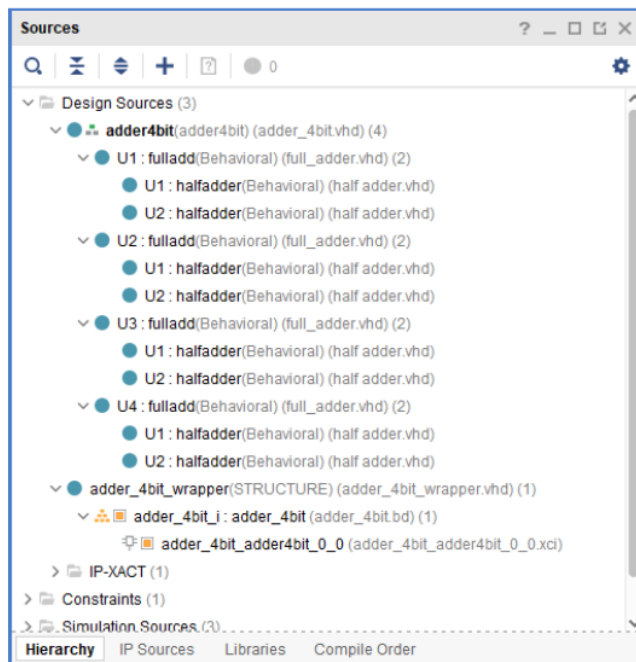


Figure 17 - 4bit adder logic diagram

3.2 VHDL Program



Hierarchy of design elements is identified here

Figure 18 - 4bit adder hierarchy

Figure 18, the hierarchy for the design circuit where the top level inherits the next level down and so on.

3.3 Simulation analysis using virtual bus method

A test bench file was created by using one of the many test bench generators found on google. Synthesis was run to verify syntax and a simulation was generated. The simulation can be seen below.

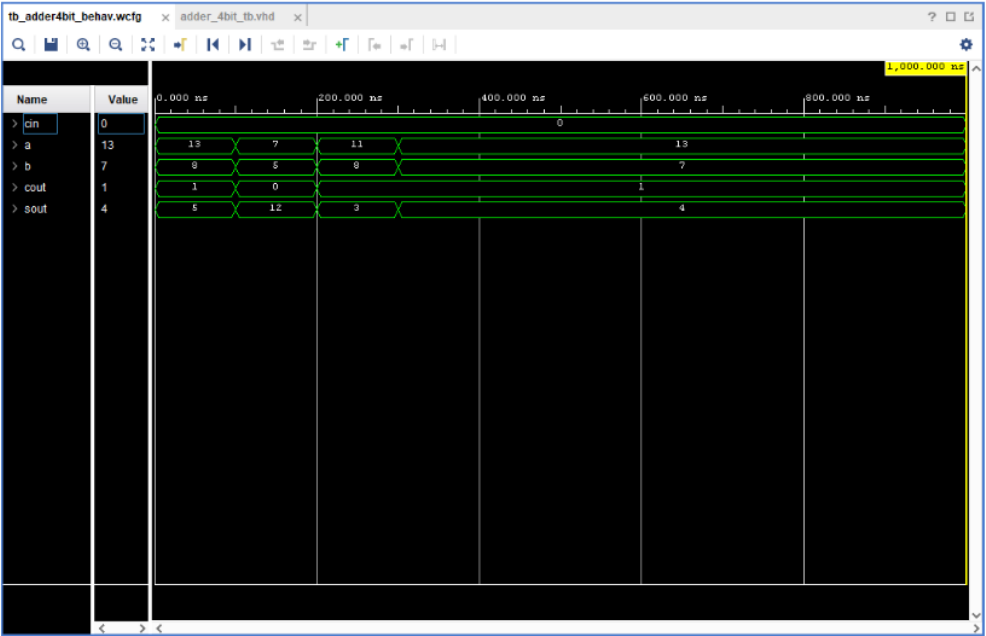


Figure 19 - 4bit adder simulation

This does not belong here. This is the test result from simulation, and you have included this in the test and analysis section.

Looking at the simulation above, when a=13 and b=8 this totals to 21 making cout go high as the total exceeds the max 8bit value of 15. Sout reads 5 as after the 4bit adder totals 16, it starts from new again.

3.4 I/P design for 4-bit adder

```

3  entity adder4bit is
4  port(
5    cin : in STD_LOGIC;
6    a0 : in STD_LOGIC;
7    a1 : in STD_LOGIC;
8    a2 : in STD_LOGIC;
9    a3 : in STD_LOGIC;
10   b0 : in STD_LOGIC;
11   b1 : in STD_LOGIC;
12   b2 : in STD_LOGIC;
13   b3 : in STD_LOGIC;
14   cout : out STD_LOGIC;
15   s0 : out STD_LOGIC;
16   s1 : out STD_LOGIC;
17   s2 : out STD_LOGIC;
18   s3 : out STD_LOGIC
19 );
20 end adder4bit;

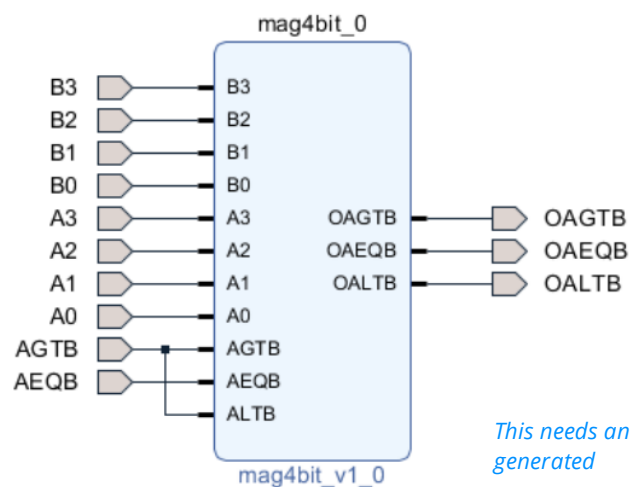
```

The entity is explained here, that specifically describes the io. The architecture that includes the circuit function code is missing

Figure 20 - 4bit adder code

Figure 20 shows the code for the 4-bit adder. Lines 5-13 are the inputs, whereas lines 14-18 are the outputs.

3.5 4-bit magnitude Comparator



This needs an explanation of how it was designed and generated

Figure 21 - 4bit magnitude comparator function block

Above shows the input design for the 4-bit magnitude comparator. If A0 and B0 is 1 then, OAEQB will be 1. If A3 is 1 and B2 is 1 then OAGTB will be 1.

3.6 4-bit adder & 4-bit mag comp combination

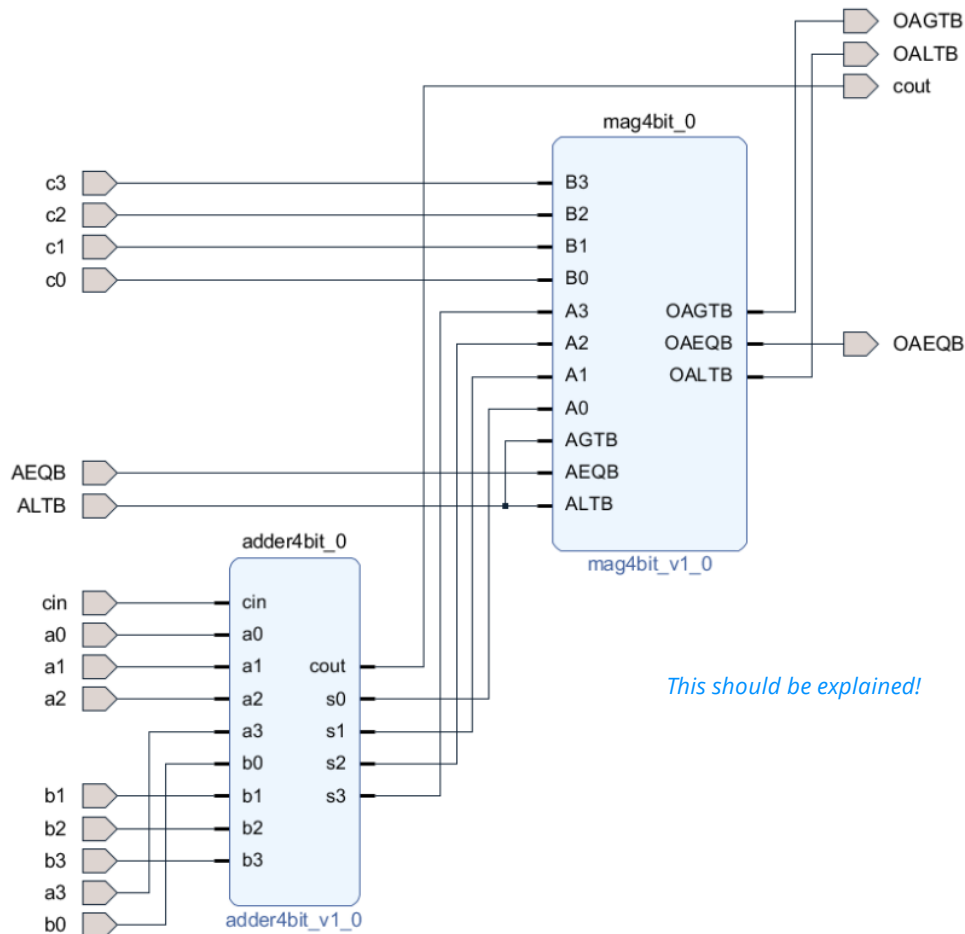


Figure 22 - 4bit adder + comparator

Figure 22 above is the input design of the 4bit magnitude comparator and the 4bit adder joined together.

3.7 4bit mag comparator *Heading here is incorrect!*

The figures below show test results of the 8-bit magnitude comparator design on MultiSim.

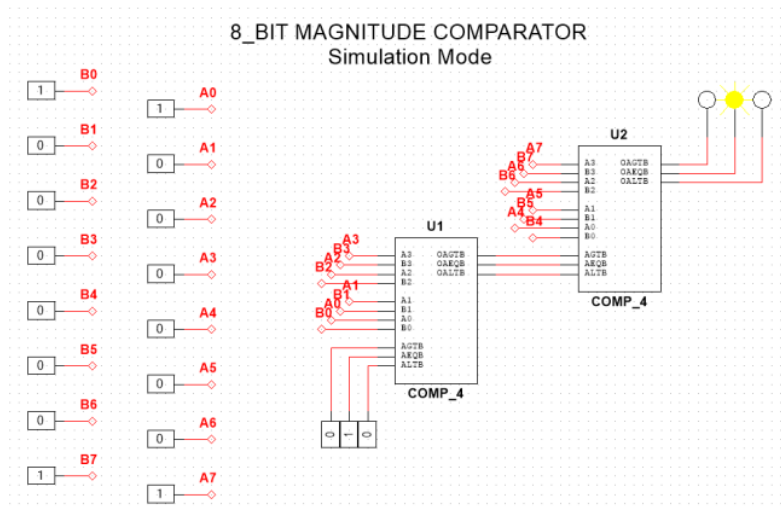


Figure 23 - 8bit mag comparator (equal to)

Above shows that B0, B7 are 1 and A0, A7 are 1, therefore the LED for OAEQB (equals to) is illuminated which is correct.

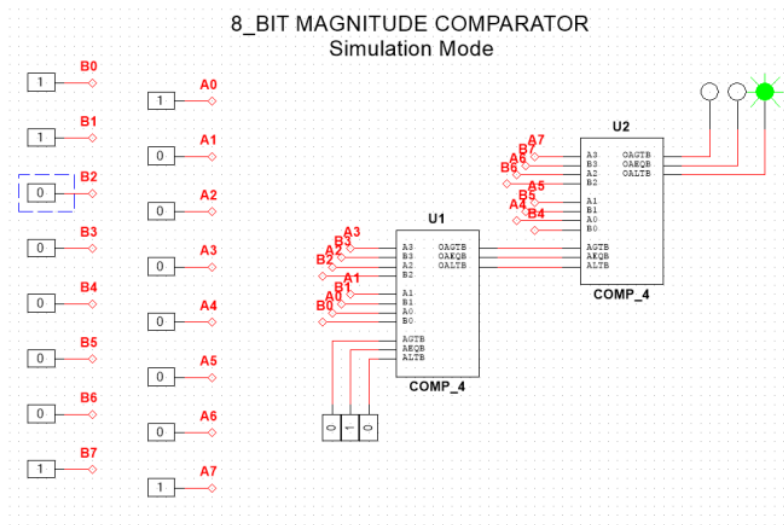


Figure 24 - 8bit mag comparator (less than)

Above shows that B0, B1, B7 are 1 and A0, A7 are 1, therefore the LED for OALTb (less than) is illuminated which is correct.

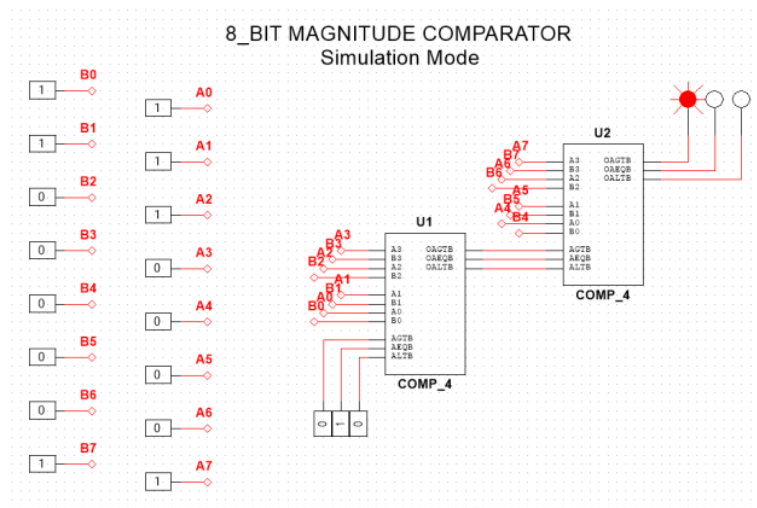


Figure 25 - 8bit mag comparator (greater than)

Above shows that B0, B1, B7 are 1 and A0, A1, A2, A7 are 1, therefore the LED for OAGTB (greater than) is illuminated which is correct.

8 bit magnitude comparator sim test is explained

4. Implementation

Combined 4-bit adder & 4-bit comparator image of FPGA – with the I/P design

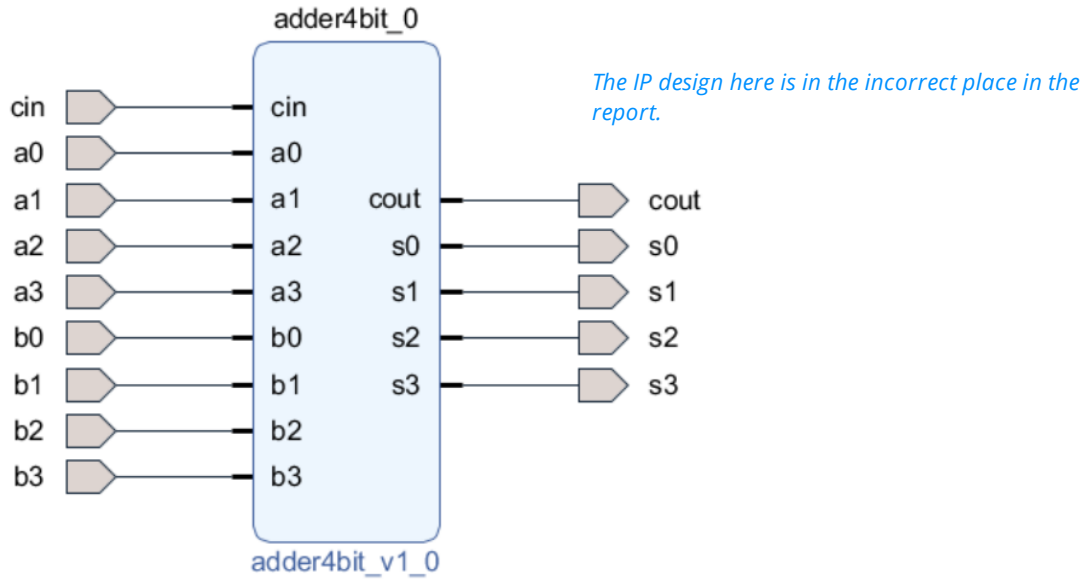


Figure 26 - IP Design for 4bit adder

The coding must be simulated before implementing on an FPGA board to prove the operation is correct. A test bench for 4bit adder and 4bit magna comparator was generated online and adapted to suit the simulation. The test bench script can be seen in appendix A4.

The 8bit magnitude comparator was simulated using a PLD design on MultiSim using Pmod connectors and LED outputs which is seen in figure 27.

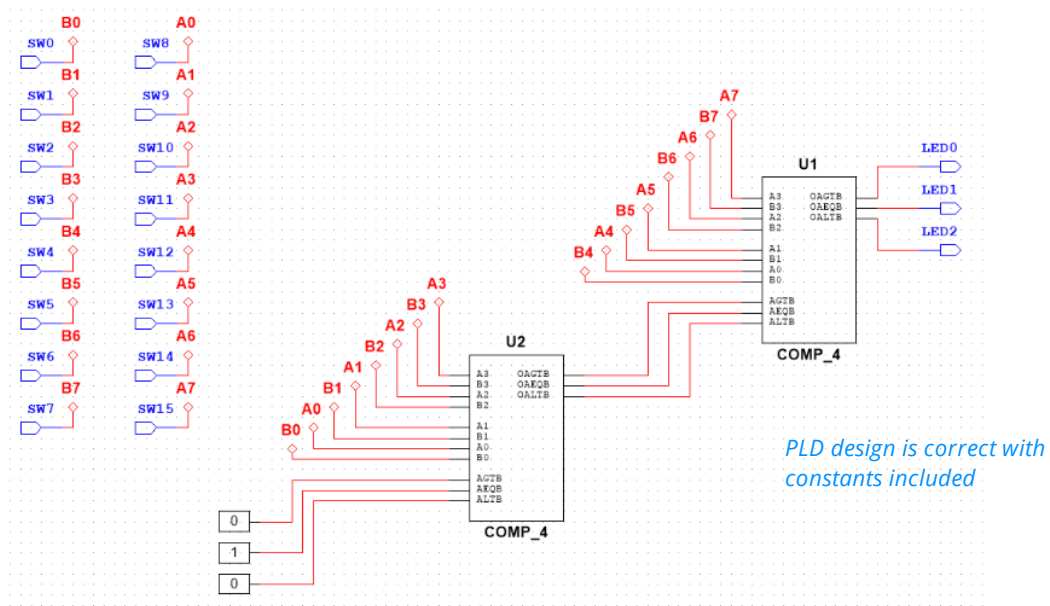


Figure 27 - 8-bit PLD design

5. Test & Analysis

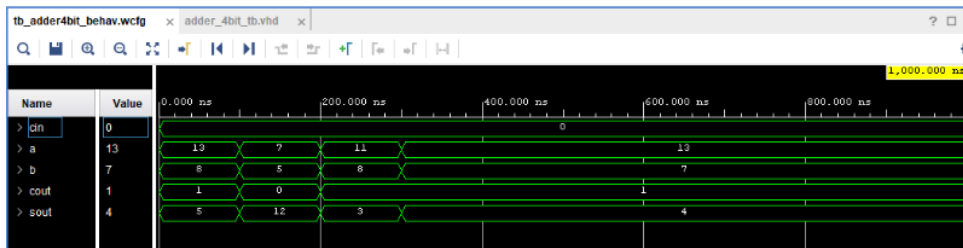


Figure 28 - 4bit adder simulation

Test result is very good with 'cout' demonstrated and explained

Analysing figure 28 above shows three test samples for the 4bit adder, 1st test is a=13 and B=8 which adds up to 21 which make cout=1 because it exceeds the max 4bit value of 15 and sout=5. 2nd test is a=7 and b=5 which makes cout=0 as it does not exceed max 4bit value, sout =12 as it is the total of a+b without it exceeding value of 15. 3rd test is a=11 and b=8 which makes cout=1 as it exceeds max 4bit value and sout = 3 as the count starts over again when it exceeds 15.

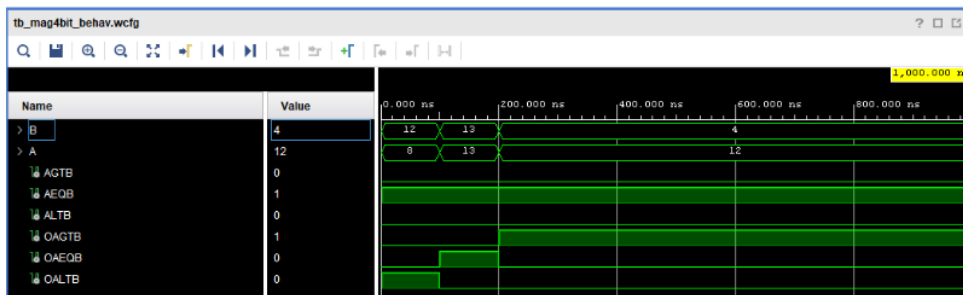


Figure 29 - 4bit mag comparator simulation

Good set of results for all 3 output possibilities

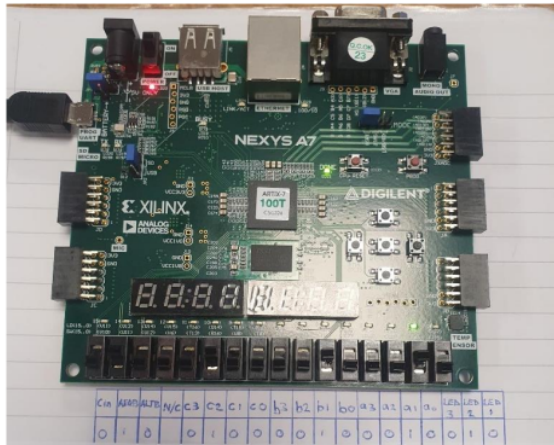
Analysing figure 29 above shows three test samples for the 4bit mag comparator, 1st test is B=12 and A=8 which makes OALTB go high. 2nd test is B=13 and A=13 which makes OAEQB go high. 3rd test is B=4 and A=12 which makes OAGTB go high.

The designs can now be implemented on the Nexys A7 FPGA boards.

Tables could have been used here to present the results more effectively

5.1 4bit Mag Comparator FPGA Testing

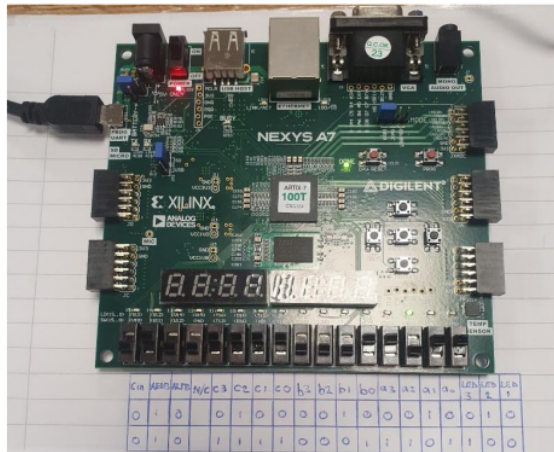
Heading should be '4 bit adder and 4 bit mag comparator combination'.



Test is only part evident in the image, part of the switch setting layout is not in the image.

Figure 30 - FPGA 4bit mag comparator test 1

Looking at test 1 above the dipswitches for a1 and b1 are switched on which brings LED 2 (equal to) on.



Test is only part evident. Part of the switch settings is not evident in the image

Figure 31 - FPGA 4bit mag comparator test 2

Identification is not correct. This should be the 4 bit adder and comparator combination

Looking at test 2 above the dipswitches for a0, a2, a3 and b0 are switched on which brings LED 3 (greater than) on.

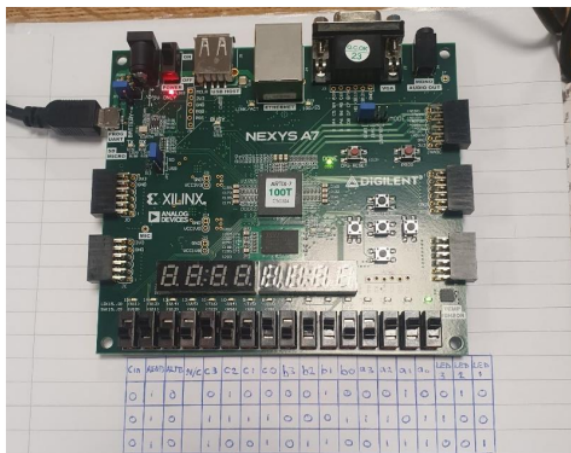


Figure 32 - FPGA 4bit mag comparator test 3

Looking at test 3 above the dipswitches for a0, a1, a2 and b0, b3 are switched on which brings LED 3 (less than) on.

All switch settings are evident and test is correct

The tests carried out are correct. You have confused the identification with the 8 bit comparator. These 3 tests are the 4 bit adder and 4 bit mag comparator combination.

The test needs to be explained further!

5.2 8bit Adder & Comparator FPGA Testing

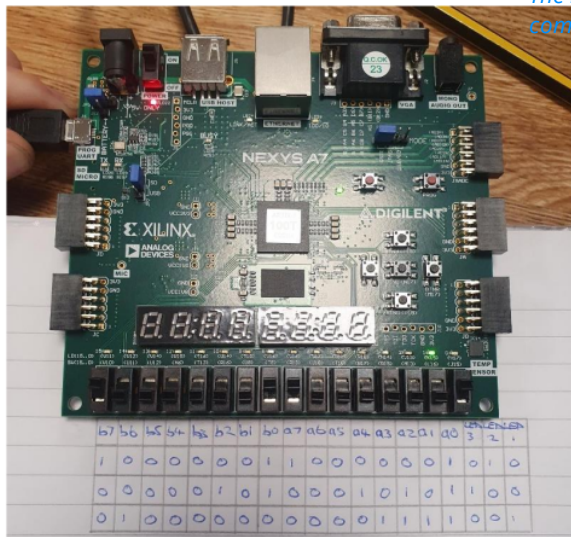


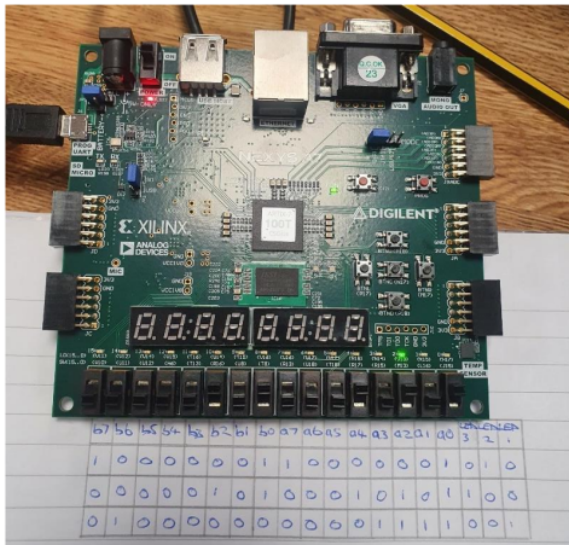
Figure 33 - FPGA 8bit adder & comparator test 1

Looking at test 1 above the dipswitches for a0, a7 and b0, b7 are switched on which brings LED 2 (equals to) on.

The heading here is incorrect. This should be the 8 bit comparator

The test carried out is correct and is evident in the image

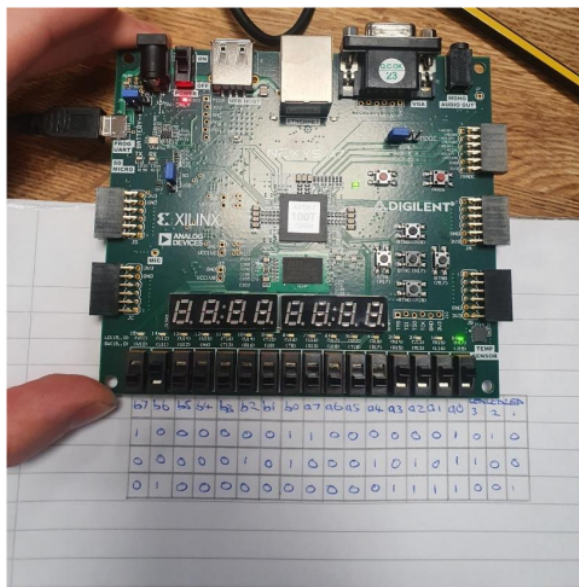
Confusion with image identification. This is the 8 bit comparator only



Test is correct and evident in the image

Figure 34 - FPGA 8bit adder & comparator test 2
Incorrect identification. This is the 8 bit comparator

Looking at test 2 above the dipswitches for a0, a2, a4 and b0, b2 are switched on which brings LED 3 (greater than) on.



Test is correct and evident in the image

Figure 35 - FPGA 8bit adder & comparator test 3
This is the 8 bit comparator

Looking at test 3 above the dipswitches for a0, a1, a2, a3 and b6 are switched on which brings LED 1 (less than) on.

6. Evaluation

From research and experiments the 4-bit comparator can be used to compare two 4-bit words. The two 4-bit numbers are a0, a1, a2, a3 and b0, b1, b2, b3 where a3 and b3 are the most significant bits. The comparator compares the binary numbers from b against a, and the output statements are either less than, greater than, and equal to. Instruments such as load cells need to have their signal conditioned to convert the signal into a useable signal such as 0-10 volts. *This explanation is incomplete!*

From the results gathered from the testing and analysis, it confirms the operation of the circuits designed through vivado and multisim are correct, this was expected as the circuits was simulated prior to testing. Using a testbench program generating from the internet proved useful and was needed to carry out simulations. Also, the synthesis tool was crucial to determine if there were any errors in the circuits.

Evaluation does not point out any significance for the subsequent elements or include a recommendation

Element 2

1. Introduction

Element 2 involves designing a virtual interface through the LabVIEW software on a PC that will interface to an FPGA containing the 8-bit Magnitude Comparator created in Element 1 which is tested on an NI ELVIS II board. This will produce a comparison output between 2⁸ bit digital signals and indicate whether the comparison is less than, equal to or greater than a set value. Also as part of Element 2, a load cell is designed and tested by using the NI ELVIS II board to provide analogue input to LabVIEW in the range of 0-5V which represents a load being applied to an LCKD-1Kg load cell. The weight is then converted to an 8bit digital signal using ADC formulas and outputted to an FPGA. Another signal will be obtained from LabVIEW to represent a set weight value ranging from 0-1000 grams and converted to an 8bit digital signal and outputted to an FPGA and comparisons between actual weight value from the load cell against the setpoint weight in one of three states, less than, equal to and greater than.

*This is an
incorrect
symbol '*' to use
in this context*

2. Design Requirements

2.1 Unity Gain of Instrument Amplifier

Figure 36 below is a circuit of an instrument amplifier with a unity gain of 1.

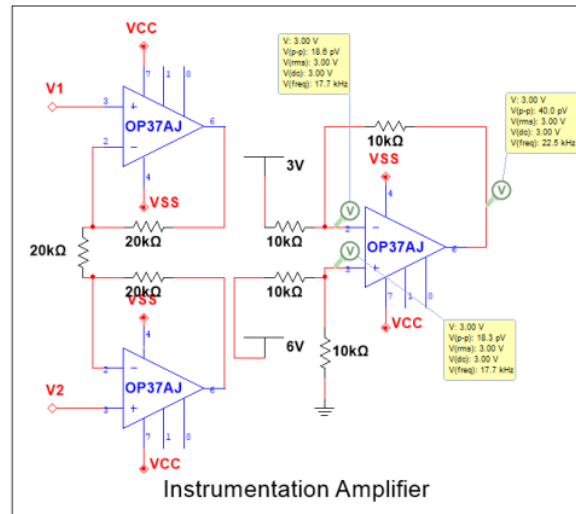


Figure 36 - Instrumentation amplifier with unity gain of 1

The output is equal to the difference between the two input levels and the gain is set to 1 by the feedback resistor and the input resistor being equal.

Load cell instrument amplifier

2.2 Instrument Amplifier with Bridge for Load Cell Sensor

An LCKD-1Kg is used to carry out the design requirements in this portfolio. This load cell has the following specifications:

Excitation: 5Vdc, 7Vdc max

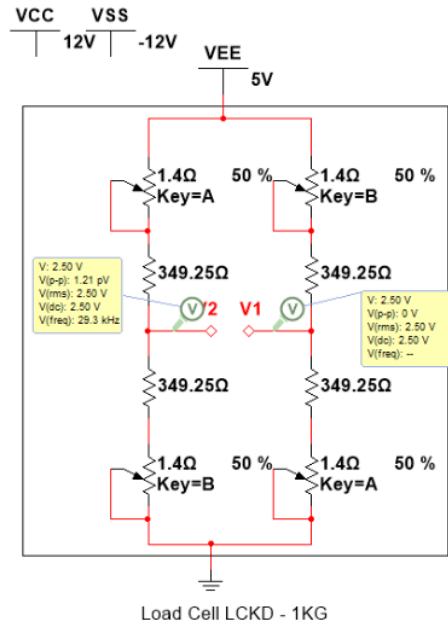
Output: 2mV/V nominal

Unstressed resistance: 350Ω

Good information about load cell spec

The weight is simulated by varying the value of the potentiometers (see in figure 37). Top-left and bottom-right potentiometers are increased whilst the remaining potentiometers are decreased. This will simulate weight being placed on the load cell in 5% stages.

At 50% across all potentiometers, the load cell measures 0V (zero grams). The load cell will read 5V (1000 grams) at 100% across top-left and bottom-right potentiometers and opposing potentiometers at 0%.



Good explanation and supporting image with initial conditions

Figure 37 - Load cell

A gain of 500 is required from the amplifier to correctly condition the signal. The calculations are presented below:

$$\text{@ 1KG and 5V excitation } V2 - V1 = 10\text{mV}$$

$$\text{@ 1KG } V_{\text{out}} = 5\text{V}$$

$$\text{Gain } G = 5 / 0.01 = 500$$

Good identification of spec requirements

To achieve the correct gain, the values of the resistors are changed so that the buffer has a gain of 5 and the differential amplifier has a gain of 100. The workings for these are shown below:

$$\text{Buffer gain} = (1 + (2 * 20\text{k}) / 10\text{k}) = 5$$

$$\text{Diff amp gain} = 1000 / 10 = 100$$

Incorrect value here. You have used 500k and 5k resistor values

The two gains are multiplied together to provide an overall gain of 500 for the instrument amplifier.

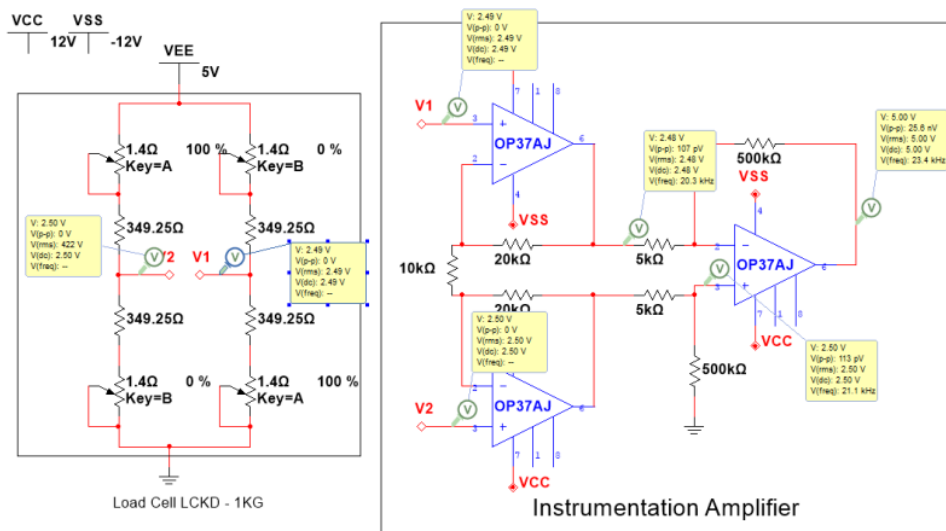


Figure 38 - Instrument amplifier with bridge

The above figure shows the circuit for the instrument amplifier with the bridge for the load cell with the updated values as per the calculations.

1.3 Testing

The circuits in figure 38 was tested by increasing the top-left and bottom-right potentiometers by 5% stages and decreasing the opposing potentiometers by 5% stages. This replicated 100 grams being placed on the load cell. The results can be seen in the below. Images of each stage can be seen in the appendix A3.

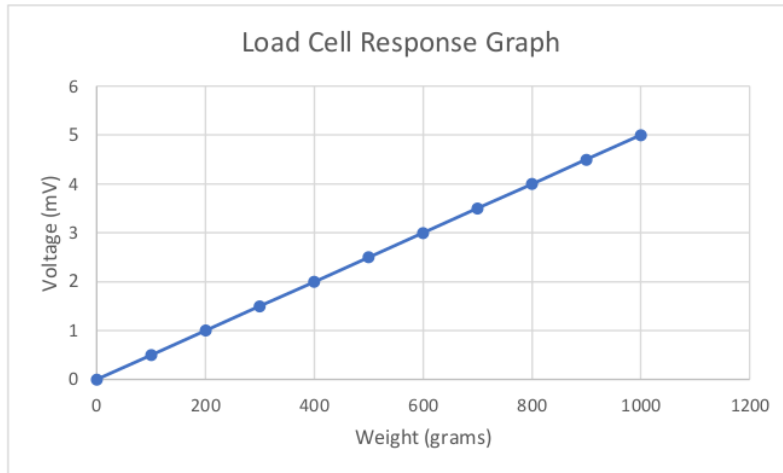
Resistance	Grams (g)	Voltage (V)
50%	0	0
55%	100	0.5
60%	200	1
65%	300	1.5
70%	400	2
75%	500	2.5
80%	600	3
85%	700	3.5
90%	800	4
95%	900	4.5

Good set of test results

100%	1000	5
------	------	---

Table 1 - Load cell results

Analysing the results in the table above, the signal conditioning is correct by providing the correct output for each load. For every load increase of 100 grams, the output voltage is increased by 0.5 volts. A line graph is produced from the results.



Good linear graph produced from results

Figure 39 - Load cell response graph

As expected from the results, the graph shows a linear response, and the output is proportional to the weight.

2. Prototypes

3.1 10- and 12-bit ADC

Prototypes were created using 10-bit and 12-bit ADC using LabView. LabView allows simple human interface with little programming needed.

10 and 12 bit ADC using LabVIEW

10 bit ADC system using LabVIEW

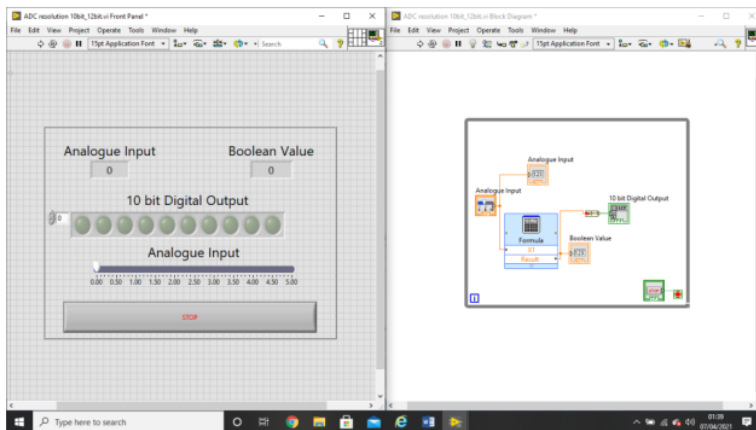


Figure 40 - 10 bit ADC system using LabVIEW

The above image shows the interface and circuit of the 10-bit ADC system using LabVIEW

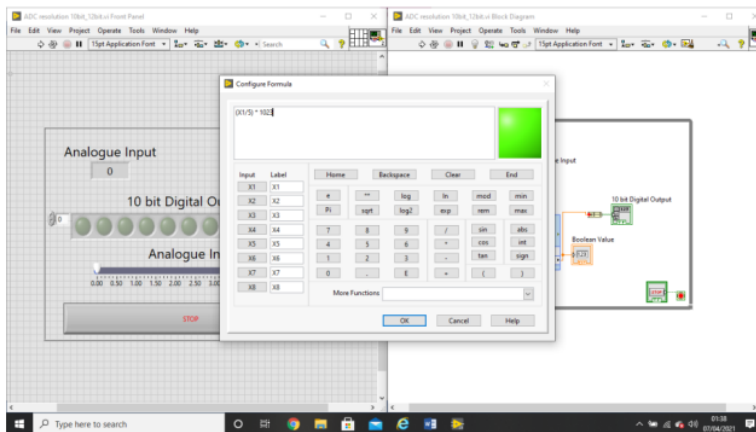


Figure 41 - Formula block

The above image shows the formula block and values used.

1023 – Resolution 10 bit

5V is max scale

X – Analogue input signal

Correct block program for 10 bit resolution

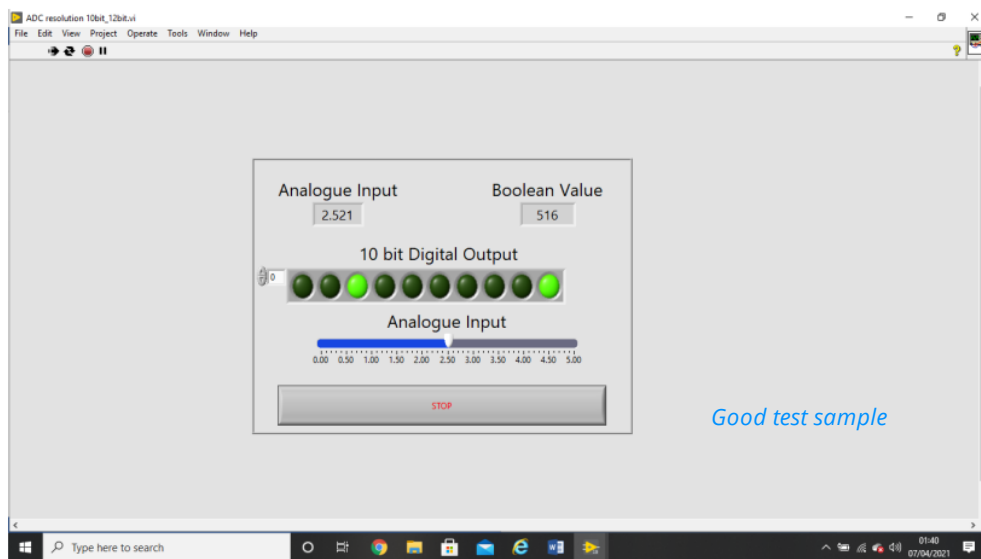


Figure 42 - LabVIEW interface page

LED's are equal to the Boolean value

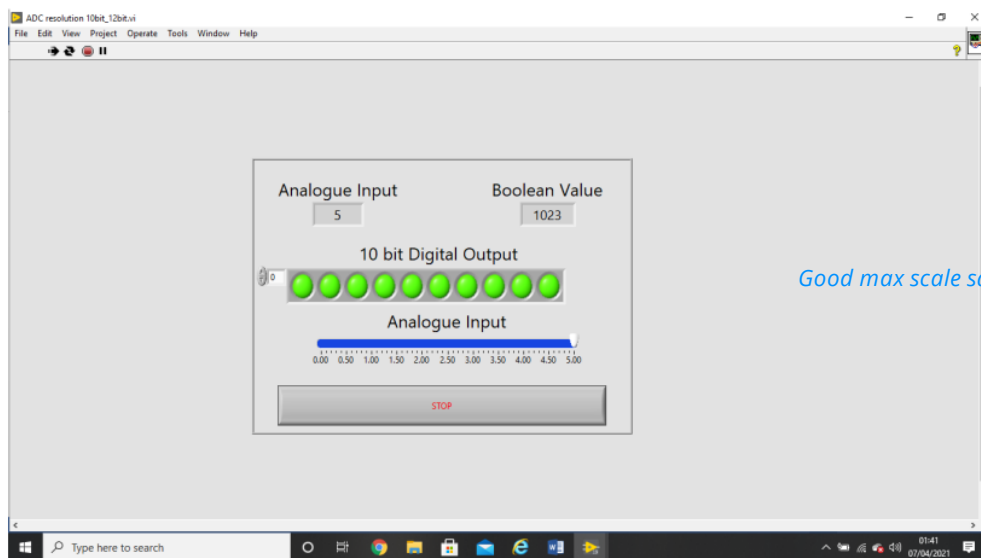
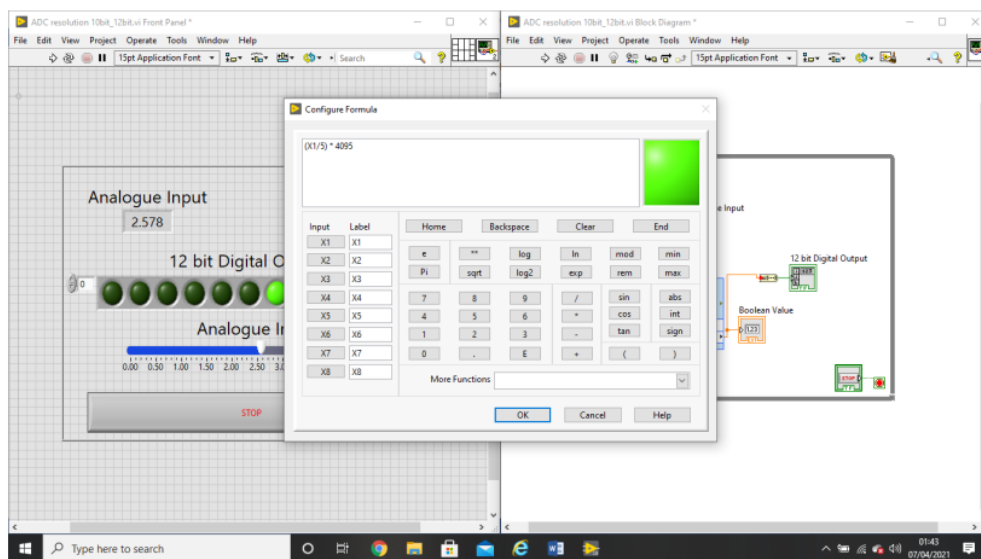
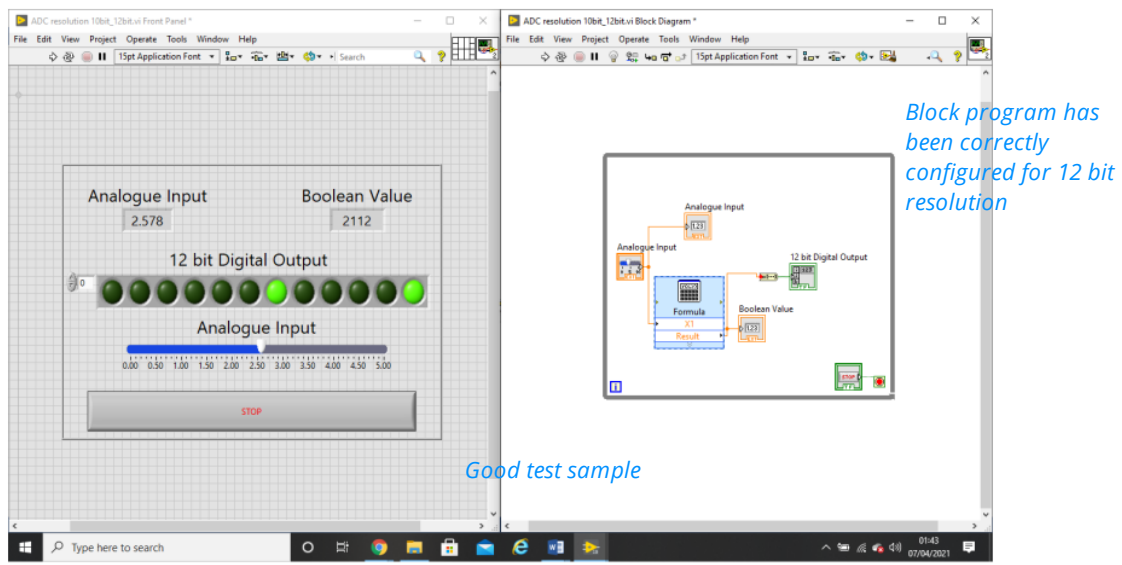


Figure 43 – 10 bit Value at full scale

Image above shows the value at full scale which is 1023.

12 bit resolution



The above image shows the formula block and values used.

4095 – Resolution 12 bit

5V is max scale

X – Analogue input signal

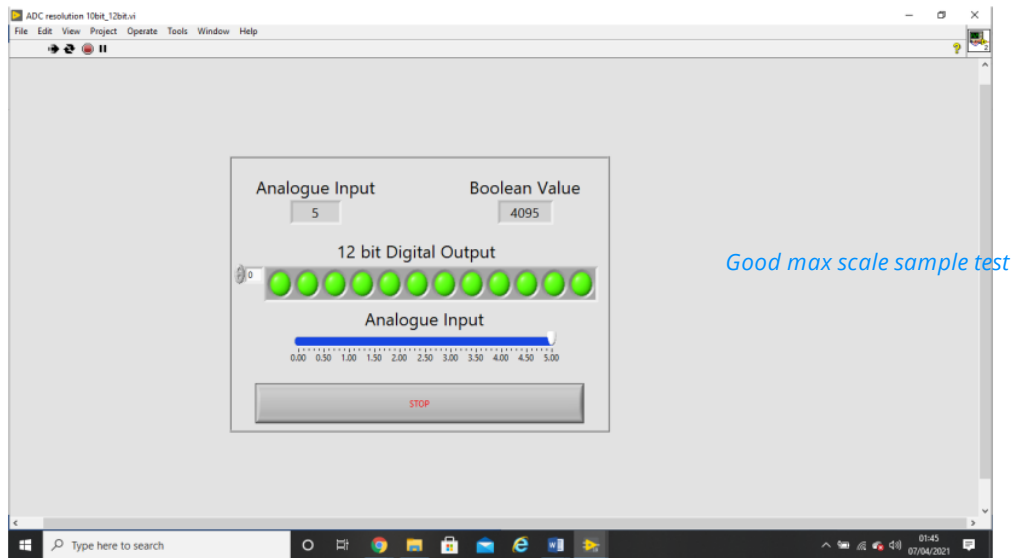


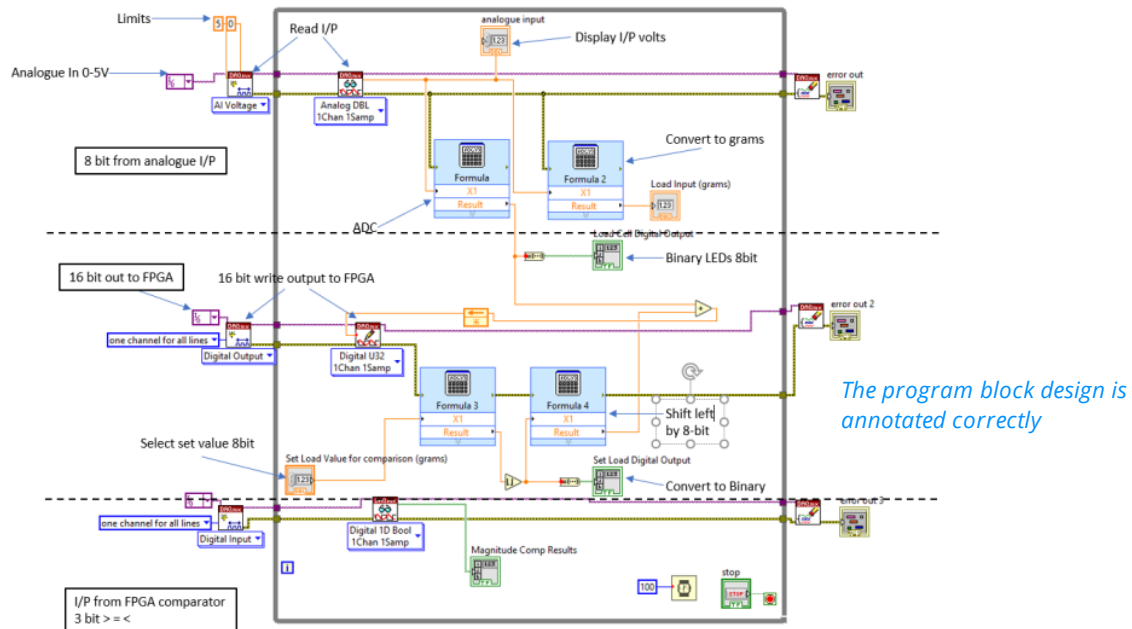
Figure 46 - 12 bit Value at full scale

Image above shows the value at full scale which is 4095.

8 bit ADC prototype with virtual input is missing

8 Bit

This heading is incorrect



Note: The 8-bit shift left (increase number value) is multiplied by 256 which is 8 bit value.

This moves the 8 bits over a value of 256 for all 8 bits.

The other 8 bits are then added to the 8-bit space left.

Example: 16 bits need to be sent to FPGA.

00000000 11101111 - 16 bits (Values are just for the example)

□ Shift the left 8 bits to the right 8 bits by multiply each bit by 256.

So:

11101111 00000000 - The left manual set value is shifted *256 to the right and the eight on the left is a space to add the other 8 bits from the analogue input.

Now the program adds the next 8 bits (random values for example) form analogue input.

11101111 00110011

The shifted 8 bits are the manual set value for comparison. The added 8 bits are the set value from the sensor (potentiometer) analogue input.

Good explanation of the shifted 8 bit value

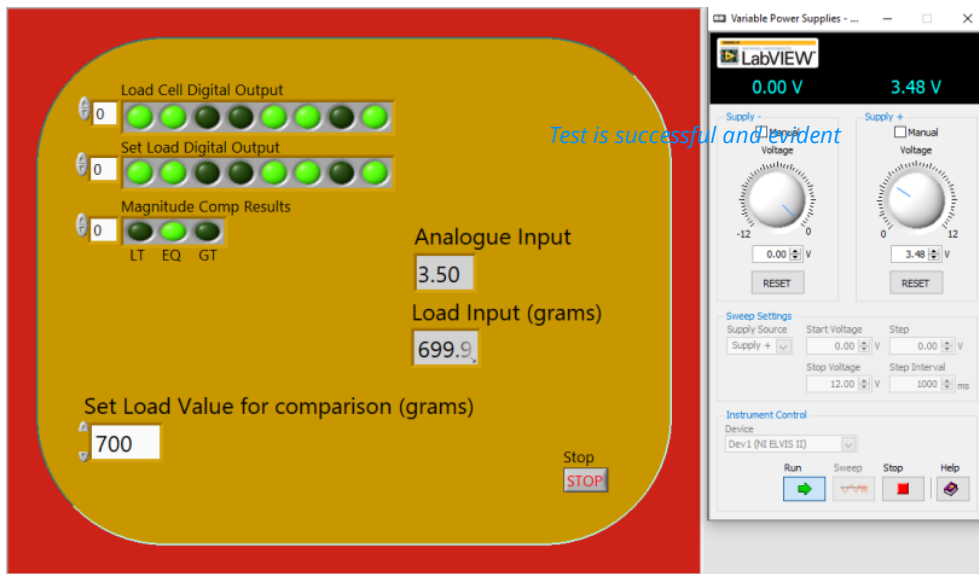


Figure 47 - Load cell test 1

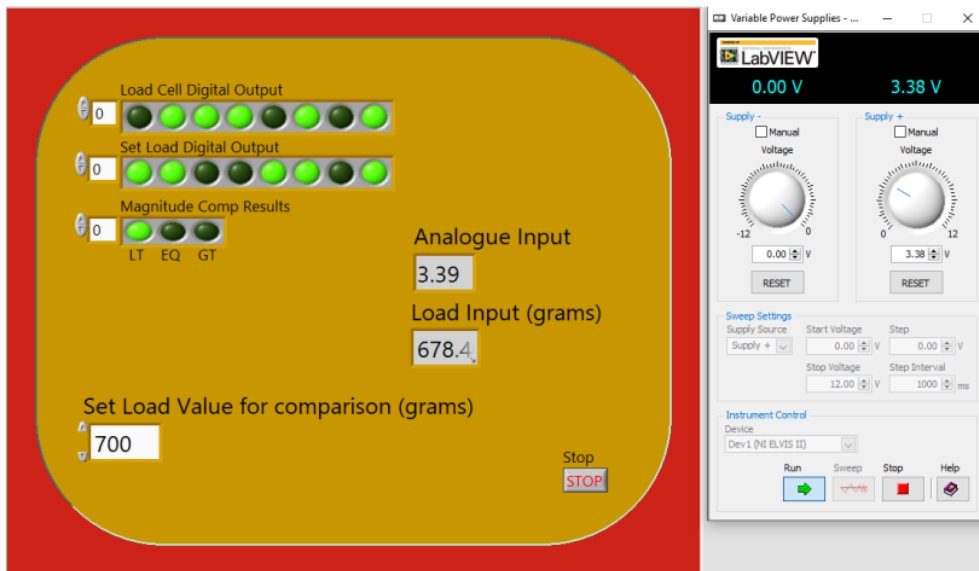


Figure 48 - Load cell test 2

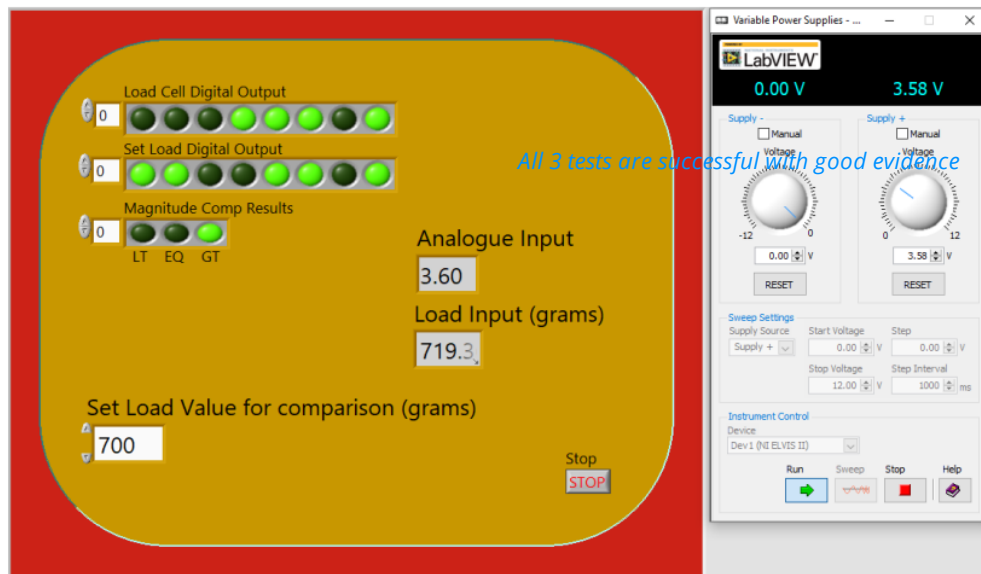


Figure 49 - Load cell test 3

Evaluation

Test Table Good use of table of results					
Test No.	State	LabVIEW Result	FPGA 8bit Result	Comp	Pass/Fail
1	<	Sensor value < SP	SV < SP		Pass
2	=	Sensor value = SP	SV = SP		Pass
3	>	Sensor Value > SP	SV > SP		Pass

- 3 test states were carried out with 8-bit mag comparator (< = >)
- Test results indicated in LabVIEW were the same as result feedback from FPGA
- This proves the operation of each state in real time was reliable
- The same test procedure can be implemented in element 3

3. Digital Magnitude Comparator

4.1 Nexys 4 DDR FPGA Development Board

The Nexys 4 DDR FPGA is a programmable board that works well on MultiSim. The coding for the magnitude comparator is burnt onto the board to be used as a magnitude comparator. It also works in conjunction with LabView.



Figure 20. PMOD connectors; front view, as loaded on PCB.

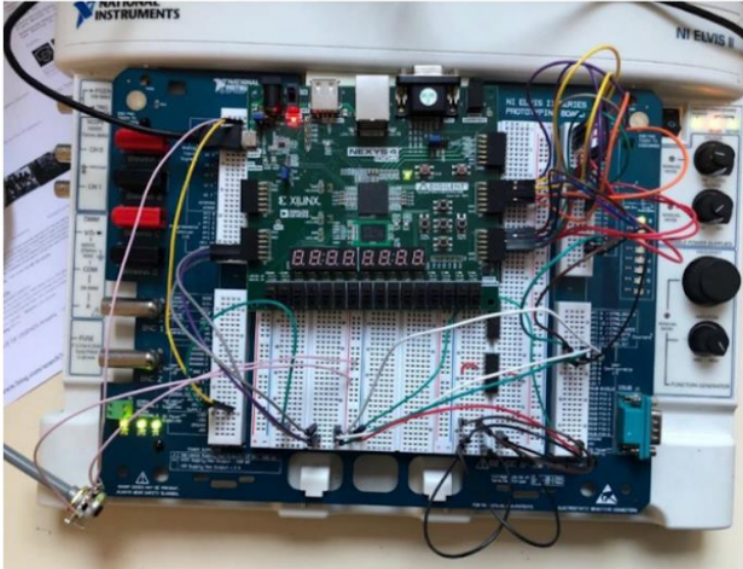
Table 5. Nexys4 DDR Pmod pin assignments.

Pmod JA	Pmod JB	Pmod JC	Pmod JD	Pmod XDAC
JA1: C17	JB1: D14	JC1: K1	JD1: H4	JXADC1: A13 (AD3P)
JA2: D18	JB2: F16	JC2: F6	JD2: H1	JXADC2: A15 (AD10P)
JA3: E18	JB3: G16	JC3: J2	JD3: G1	JXADC3: B16 (AD2P)
JA4: G17	JB4: H14	JC4: G6	JD4: G3	JXADC4: B18 (AD11P)
JA7: D17	JB7: E16	JC7: E7	JD7: H2	JXADC7: A14 (AD3N)
JA8: E17	JB8: F13	JC8: J3	JD8: G4	JXADC8: A16 (AD10N)
JA9: F18	JB9: G13	JC9: J4	JD9: G2	JXADC9: B17 (AD2N)
JA10: G18	JB10: H16	JC10: E6	JD10: F3	JXADC10: A18 (AD11N)

Figure 50 - Nexys 4 DDR Connections

Figure 50 above shows the connections for the Nexys 4 DDR board. This was used to understand the pin layout and how to connect the Nexys 4 DDR board to the NI ELVIS II.

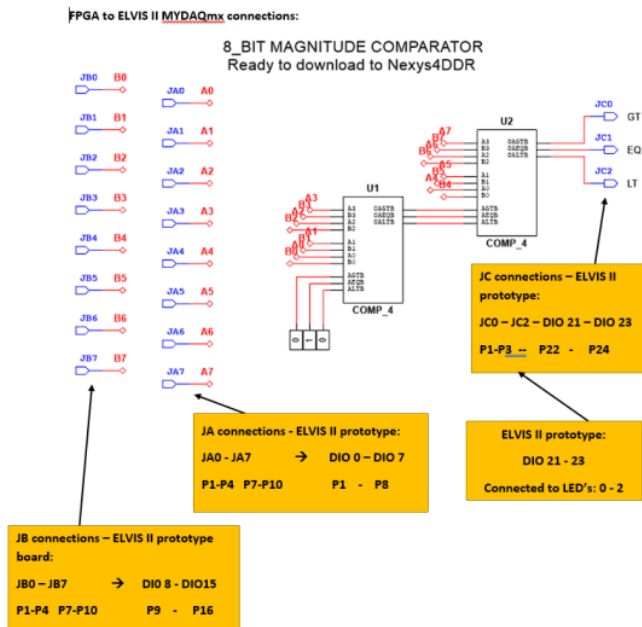
4.2 Magnitude Comparator Setup



Good image showing practical setup using PMOD ports of FPGA

Figure 51 - NI ELVIS II Magnitude Comparator Set Up

The image above is the final set up of the magnitude comparator (Nexys DDR FPGA) connected to the NI ELVIS II.



Good connection key that demonstrates a good practical understanding of the interface connections for FPGA and LabVIEW with DAQmx

Figure 52 - FPGA to ELVIS II MYDAQmx connections

3.3 LabView Program

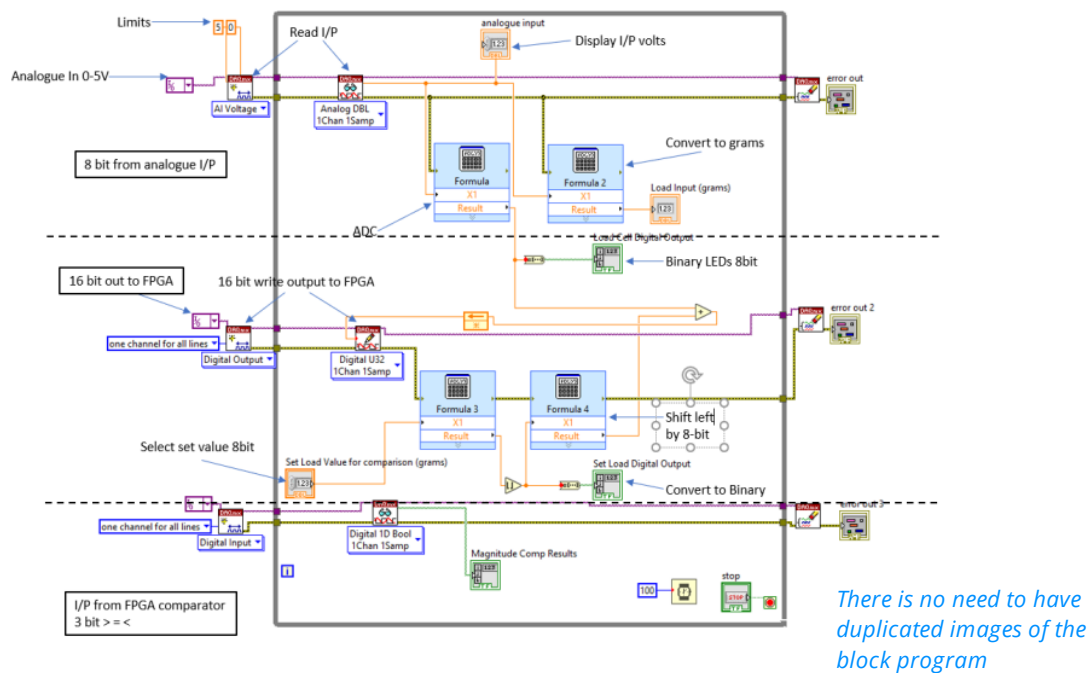


Figure 53 - Block Diagram

Formula 1 in figure 53 is used to convert the analogue input to grams. This is achieved by dividing the analogue input value by the maximum analogue input of 5, and then multiplying by 1000. For example, analogue input value of 1 will give a weight value of 200 grams $((1/5)*1000=200)$.

Formula 2 is used to convert the analogue input to a digital signal. This is achieved by dividing the analogue input by its maximum value of 5, and then multiplying by 256 (8-bit) to convert to a binary number.

Formula 3 is used to convert the set weight value into 8-bit word output. This is achieved by dividing the set weight value by maximum weight value (1000) and then multiplying by the 8-bit resolution which is 256.

Formula blocks are explained

Formula 4 shifts the bits over 8 places by multiplying by 256.

Note: The 8-bit shift left (increase number value) is multiplied by 256 which is 8 bit value. This moves the 8 bits over a value of 256 for all 8 bits. The other 8 bits are then added to the 8-bit space left. For example, 16 bits need to be sent to FPGA:

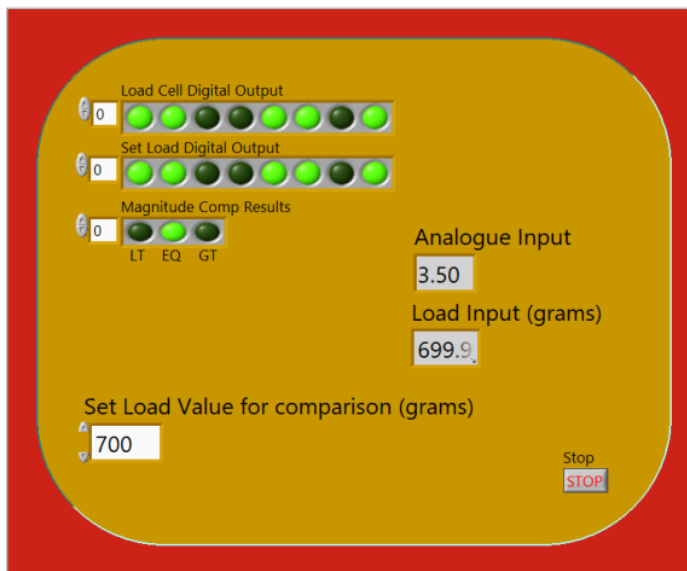
00000000 11101111 - 16 bits

Shift the left 8 bits to the right 8 bits by multiply each bit by 256.

So:

11101111 00000000 - The left manual set value is shifted *256 to the right and the eight on the left is a space to add the other 8 bits from the analogue input. Now the program adds the next 8 bits (random values for example) from analogue input.

11101111 00110011 - The shifted 8 bits are the manual set value for comparison. The added 8 bits are the set value from the sensor (potentiometer) analogue input. *Good example that demonstrates an understanding*



Good front panel design

Figure 54 - Panel View

3.4 Test Results

The program was tested by adjusting the potentiometer on the NI ELIS II. This varies the analogue input which is converted by the ADC formula block. Three tests are carried out to ensure the magnitude comparator is working correctly. Each tests compares the set load value of 700 grams to the load input to measure less than, greater than & equals to.

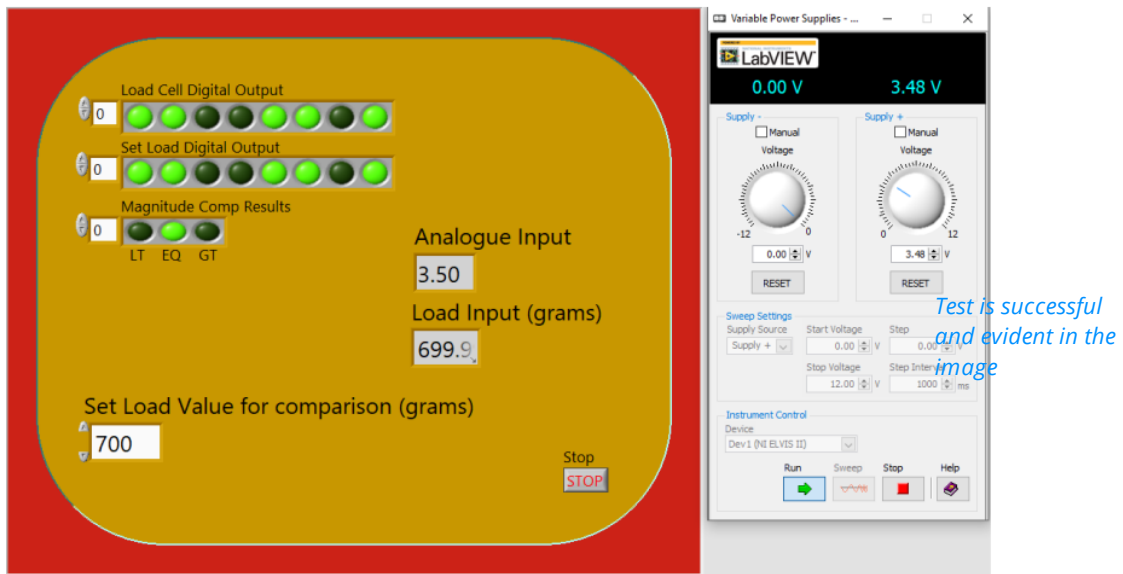


Figure 55 - Test 1 equals to

Figure 55 above shows that the load input equals the set load. This is confirmed by the LED displays for load cell digital output and set load digital output representing an 8-bit binary number, which both words have the same bits illuminated.

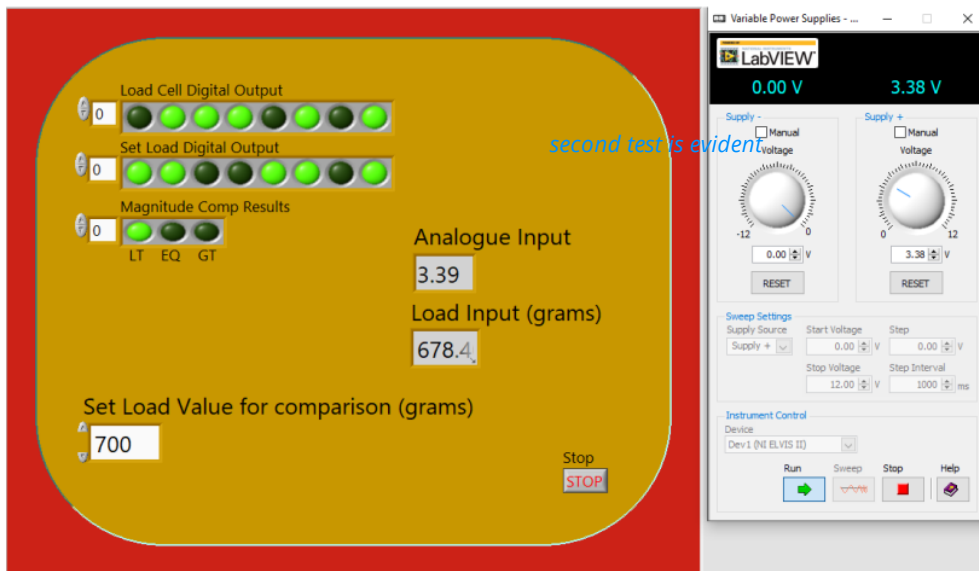


Figure 56 - Test 2 Less than

Figure 56 above shows the load input is less than the set load. Again, looking at the binary word represented by the LED displays the binary number for load cell digital output is less than binary number for set load digital output.

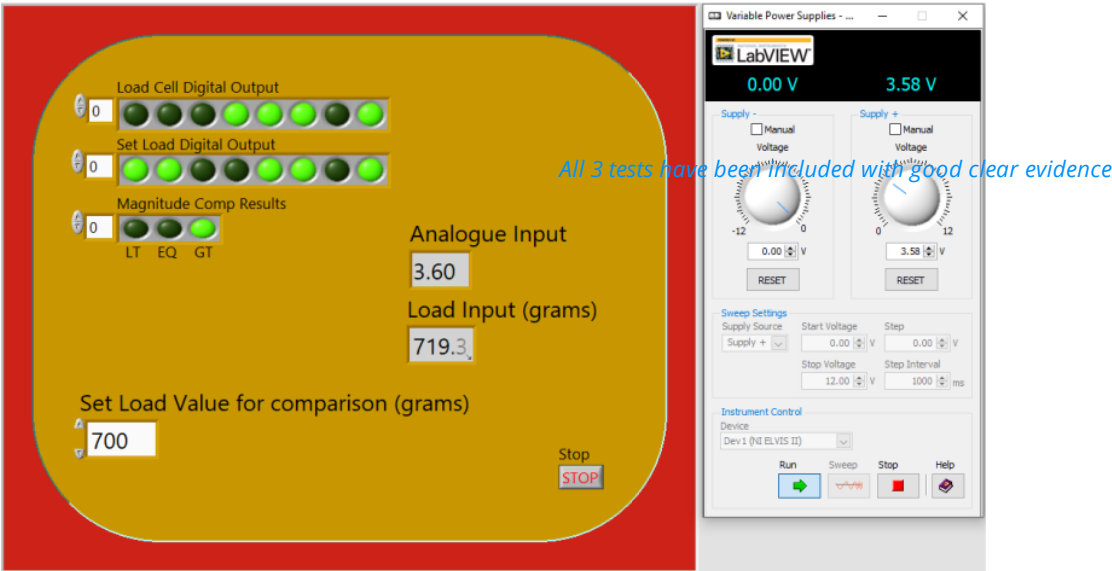


Figure 57 - Test 3 Greater than

Figure 57 above shows the load input is greater than 700 grams. All three tests conclude that the program is working correctly.

4. Evaluation

Test Table						Pass/Fail
Test No.	State	LabVIEW Result	FPGA Result	8bit	Comp	
1	<	Sensor value < SP	SV < SP			Pass
2	=	Sensor value = SP	SV = SP			Pass
3	>	Sensor Value > SP	SV > SP			Pass

Table 2 – Test Results

There is no need for the duplicates of the test results

Analysing the table above indicate the three tests carried out passed, confirming the program is working correctly. The results indicated in LabView were the same as results feedback from the FPGA in portfolio 1 which was as expected. This proves the operation of each state in real time was reliable. The successful results indicate that the same test procedure can be implemented in portfolio 3.

Element 3

A poor evaluation

Element 3

1. Introduction

This section of the portfolio involves adapting the NI LabVIEW design to output the actual load cell value (in grams) over its full range and the comparison values in the form of data logging files. Microsoft EXCEL is used to produce a graph of actual load over the range for a period of time ensuring that all comparison of results is covered. This element uses the same program built in element 2 with the main difference is that the data is recorded over a specific time and is saved into different files depending on the data. The benefit of this is that the data is logged and stored which can be used as a history record. These are used in many SCADA systems to pull up past history data for an instrument or an actuator.

The intro includes some good significant points relevant to the extended element 2 block program design

2. Implementation

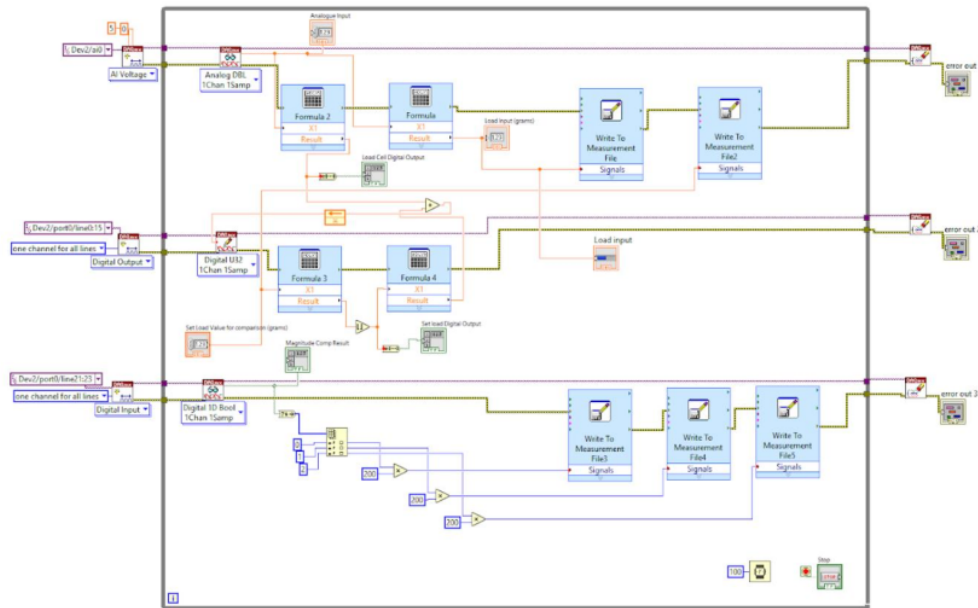


Figure 58 - Block diagram with added file write blocks

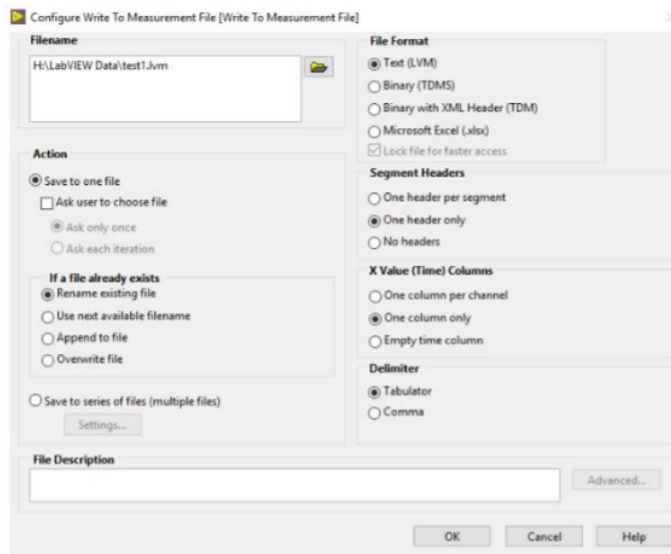
There are 5 write to file blocks added to the program which three can be seen at the bottom-right, and two at the top-right of figure 58, these takes the inputs and saves them to a specific folder to keep as record. *This is not a good explanation!*

Figures need to be identified specifically

Figure XXX shows file 1 write to file block which takes the inputs from the potentiometer that simulates the load weight. Here also displays the configuration format, how much segment headers, the X-value comment and can chose the desired location to save the file. File 2 reads the inputs from the setpoint so that it stays constant. Files 3, 4 & 5 logs the data for the three comparator outputs which are less than, greater than and equal to.

This is a poor explanation

File 1



This is a good sample of the write file configuration

Figure 59 - File1 write to file block

It should be noted that the output values to the comparator write to file blocks are scaled just in order to produce clear graphical results when it comes to producing a graph. The scaling can be seen in figure 58 with the multiplication icons and blue lines connecting to write to measurement files 3,4 and 5.

Scaling is explained with a ref to evidence

3. Testing and Analysis

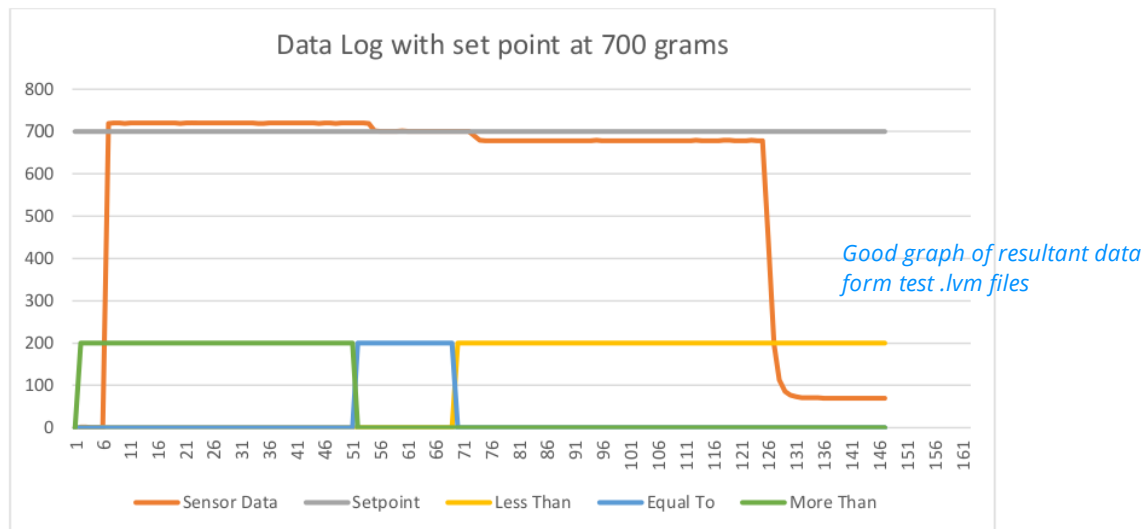


Figure 60 - Data log graph results

The graph above is produced by the data recorded from the log files which can be seen in appendix A6. The legend is displayed at the bottom of the graph, the setpoint indicated in grey reads 700 grams across the whole timeline. The sensor data (red line) reads above 700 grams which is above the setpoint and in result of this the more than (green line) trend is reading 200 meaning it's on. At roughly 56 on the x-axis, sensor data value reads 700 meeting the setpoint dead-on. This brings the more than trend line back to 0 switching it off and the equal to (blue line) trend reads 200 indicating it is switched on. Shortly after at roughly 76 on x-axis the sensor data reads below 700 dropping it below the setpoint and in result equal to reduces to 0 switching it off and the less than (yellow line) trend reads 200 meaning it is switched on. The results from this test determine that the comparator is working correctly as intended and also the data from the log files are correct and accurate.

Good explanation of resultant graph

4. Conclusion and Recommendations

Carrying out research into the topic area related to this portfolio provided good knowledge and understanding in developing logic circuits along with simulation and implementing onto FPGA boards. From research its gathered that most chips and systems in industry are designed by writing descriptions in VHDL and Verilog languages. NI DAQmx is a powerful software tool used for data acquisition that is adopted by many big-name programming software such as LabVIEW.

From the test results from the pictures of the FPGA, the 4-bit magnitude comparator worked as intended and compared with values from b against a which output one of three states, less than, greater than and equal to. However, as this is only 4-bits the limitation is that the maximum value that can be compared is 15.

The 8-bit magnitude comparator behaved exactly like the 4-bit magnitude comparator however as it is 8bits, therefore a higher number can be compared and the maximum value that can be compared is 255.

Using an instrument amplifier along with a bridge was needed for correct operation of the load cell. By changing the values of the resistors, a gain of 500 was made which provided good signal conditioning for the load cell to work well.

As the signals coming in are analogue, ADCs had to be used to convert the analog signal to digital which allowed for comparisons between digital values which is achieved through the magnitude comparators. This allowed it to draw conclusions whether one signal was equal to, greater than or less than.

Modifying the virtual interface on LabVIEW to accommodate the write to file blocks provided a very extensive amount of data to be saved in text files which is needed to produce a graph to graphically demonstrate the results. This is a massive benefit to the system as it provides historical records which can be used to check the weight of the load cell in the past. This information can be useful for users to determine if the sensor's accuracy is good or if it is starting to deteriorate and to establish if the instrument needs calibration to provide better accuracy. Along with this data, graphs can be generated to form trend lines which are widely used in SCADA systems.

The system used is 8-bits which doesn't provide a very good resolution, therefore the comparison of the raw analogue is subject to quantization error. For example, the setpoint is 700 grams but the load cell is reading 703 grams but the magnitude comparator is still indicating this is equal to the setpoint. This can be improved by incorporating 16-bits to the

A good point made here

system as this will double the resolution and reduce the quantization error making the system twice as accurate.

This portfolio has provided good knowledge and understandability on virtual programming on various software packages that are applicable to industry as many instruments will need signal conditioning and converted to digital. The knowledge is also transferable to other programming techniques as having prior knowledge and experience in one programming language will help in understanding a new one.

You only need to evaluate element 3 here

References

Anon., 2023. *What is a Load Cell?*. [Online]

Available at: <https://www.lcmssystems.com/resources/load-cells-what-is-a-load-cell>

Chang, K.-c., 1999. *Digital Systems Design with VHDL and Synthesis: An Integrated Approach*. s.l.:Wiley-IEEE Computer Society Press.

Charles H.Roth, J. & John, L. K., 2008. *Digital Systems Design Using VHDL*. 2nd ed. s.l.:Chris Carson.

Dally, W. J., Harting, R. C. & Aamodt, T. M., 2015. *Digital Design Using VHDL*. Cambridge: Cambridge University Press.

Delta Motion, 2016. *Load Cell and Wheatstone Bridge Fundamentals*. [Online]

Available at:

https://deltamotion.com/support/webhelp/rmctools/Controller_Features/Transducer_Basics/Load_Cell_Fundamentals.htm

[Accessed 27 02 2023].

Electronics Hub, 2015. *Digital Comparator and Magnitude Comparator*. [Online]

Available at: <https://www.electronicshub.org/digital-comparator-and-magnitude-comparator/>

[Accessed 27 02 2023].

Electronics Tutorial, 2014. *Digital Comparator*. [Online]

Available at: https://www.electronics-tutorials.ws/combination/comb_8.html

[Accessed 02 27 2023].

Force Logic, 2016. *WHAT IS A LOAD CELL? FAQ*. [Online]

Available at: <https://www.force-logic.co.uk/what-is-a-load-cell/>

MathWorks, 2020. *NI-DAQmx Support from Data Acquisition Toolbox*. [Online]

Available at: <https://uk.mathworks.com/hardware-support/nidaqmx.html>

[Accessed 27 02 2023].

Maxim Integrated Tutorial 810, 2013. *Virtual Instrumentation & Control Systems*. [Online]

Available at: <https://pdfserv.maximintegrated.com/en/an/AN810.pdf>

OMEGA, 2015. *Strain Gauge, Introduction to Strain gauges*. [Online]

Available at: <https://www.omega.co.uk/prodinfo/straingauges.html>

Shambro, J., 2018. *What Is Unity Gain?*. [Online]

Available at: <https://www.liveabout.com/what-is-unity-gain-1817673>

TT Electronics, 2016. *Signal Conditioning: Methods to Process Analogue Signals to Digital*.
[Online]
Available at: <https://blog.ttelectronics.com/signal-conditioning>
[Accessed 27 02 2023].

Good number of relevant reference sources

Appendix

A1 – 4bit adder architecture

```
21 ⊞ architecture adder4bit of adder4bit is
22 ⊞ component fulladd
23 ⊞ port (
24 ⊞ a : in STD_LOGIC;
25 ⊞ b : in STD_LOGIC;
26 ⊞ cin : in STD_LOGIC;
27 ⊞ cout : out STD_LOGIC;
28 ⊞ s : out STD_LOGIC
29 ⊞ );
30 ⊞ end component;
31
32 ⊞ signal c1 : STD_LOGIC;
33 ⊞ signal c2 : STD_LOGIC;
34 ⊞ signal c3 : STD_LOGIC;
35 ⊞ begin
36 ⊞ U1 : fulladd
37 ⊞ port map(
38 ⊞ a => a0, b => b0, cin => cin, cout => c1,
39 ⊞ s => s0);
40 ⊞ U2 : fulladd
41 ⊞ port map(
42 ⊞ a => a1, b => b1, cin => c1, cout => c2,
43 ⊞ s => s1);
44 ⊞ U3 : fulladd
45 ⊞ port map(
46 ⊞ a => a2, b => b2, cin => c2, cout => c3,
47 ⊞ s => s2);
48 ⊞ U4 : fulladd
49 ⊞ port map(
50 ⊞ a => a3, b => b3, cin => c3, cout => cout,
51 ⊞ s => s3);
52 ⊞ end adder4bit;
```

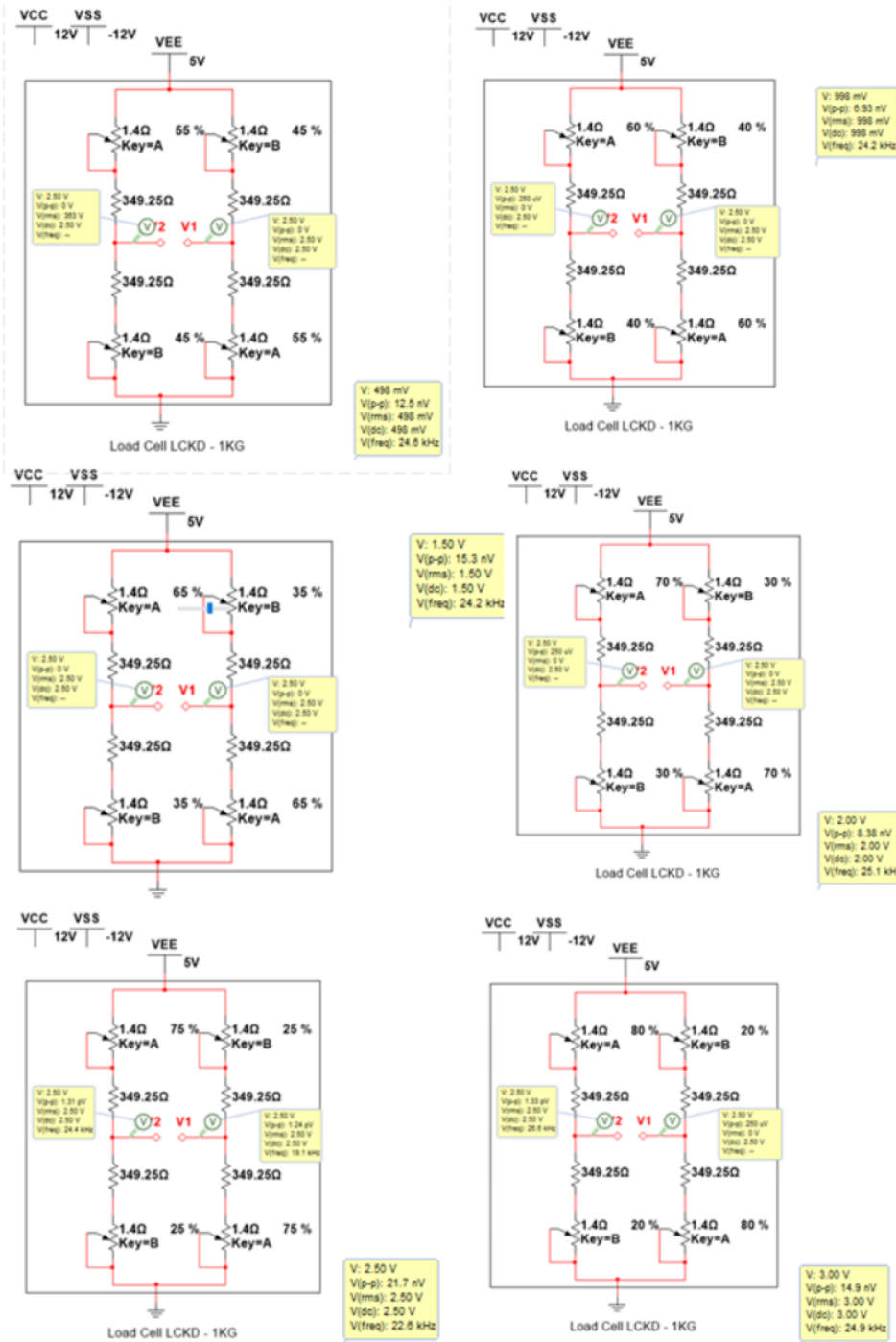
The architecture should be explained and referenced from the main report. The architecture is not explained

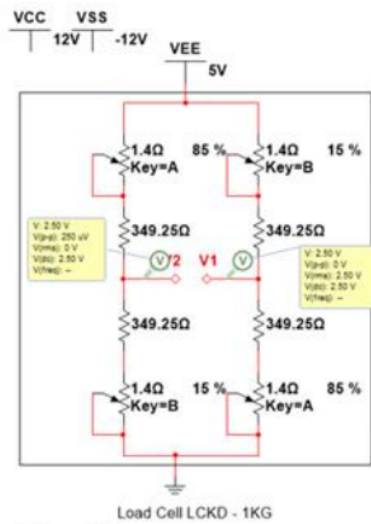
A2 – FPGA Input/outputs

```
11 ##Switches
12
13 set_property -dict { PACKAGE_PIN J15 IOSTANDARD LVCMOS33 } [get_ports { a0 }]; #IO_L24N_T3_R50_15 Sch=sw[0]
14 set_property -dict { PACKAGE_PIN L16 IOSTANDARD LVCMOS33 } [get_ports { a1 }]; #IO_L3N_T0_DQS_EMCCLK_14 Sch=sw[1]
15 set_property -dict { PACKAGE_PIN M13 IOSTANDARD LVCMOS33 } [get_ports { a2 }]; #IO_L6N_T0_D08_VREF_14 Sch=sw[2]
16 set_property -dict { PACKAGE_PIN R15 IOSTANDARD LVCMOS33 } [get_ports { a3 }]; #IO_L13N_T2_MRCC_14 Sch=sw[3]
17 set_property -dict { PACKAGE_PIN R17 IOSTANDARD LVCMOS33 } [get_ports { b0 }]; #IO_L12N_T1_MRCC_14 Sch=sw[4]
18 set_property -dict { PACKAGE_PIN T18 IOSTANDARD LVCMOS33 } [get_ports { b1 }]; #IO_L7N_T1_D10_14 Sch=sw[5]
19 set_property -dict { PACKAGE_PIN U18 IOSTANDARD LVCMOS33 } [get_ports { b2 }]; #IO_L17N_T2_A13_D29_14 Sch=sw[6]
20 set_property -dict { PACKAGE_PIN R13 IOSTANDARD LVCMOS33 } [get_ports { b3 }]; #IO_L5N_T0_D07_14 Sch=sw[7]
21 set_property -dict { PACKAGE_PIN T8 IOSTANDARD LVCMOS18 } [get_ports { c0 }]; #IO_L24N_T3_34 Sch=sw[8]
22 set_property -dict { PACKAGE_PIN U8 IOSTANDARD LVCMOS18 } [get_ports { c1 }]; #IO_25_34 Sch=sw[9]
23 set_property -dict { PACKAGE_PIN R16 IOSTANDARD LVCMOS33 } [get_ports { c2 }]; #IO_L15P_T2_DQS_RDWR_B_14 Sch=sw[10]
24 set_property -dict { PACKAGE_PIN T13 IOSTANDARD LVCMOS33 } [get_ports { c3 }]; #IO_L23P_T3_A03_D19_14 Sch=sw[11]
25 #set_property -dict { PACKAGE_PIN H6 IOSTANDARD LVCMOS33 } [get_ports { SW12 }]; #IO_L24P_T3_35 Sch=sw[12]
26 set_property -dict { PACKAGE_PIN U12 IOSTANDARD LVCMOS33 } [get_ports { ALTB }]; #IO_L20P_T3_A08_D24_14 Sch=sw[13]
27 set_property -dict { PACKAGE_PIN U11 IOSTANDARD LVCMOS33 } [get_ports { AEQB }]; #IO_L19N_T3_A09_D25_VREF_14 Sch=sw[14]
28 set_property -dict { PACKAGE_PIN V10 IOSTANDARD LVCMOS33 } [get_ports { cin }]; #IO_L21P_T3_DQS_14 Sch=sw[15]
29
30
31 ## LEDs
32
33 set_property -dict { PACKAGE_PIN H17 IOSTANDARD LVCMOS33 } [get_ports { OAGTB }]; #IO_L18P_T2_A24_15 Sch=led[0]
34 set_property -dict { PACKAGE_PIN K15 IOSTANDARD LVCMOS33 } [get_ports { OAEQB }]; #IO_L24P_T3_RS1_15 Sch=led[1]
35 set_property -dict { PACKAGE_PIN J13 IOSTANDARD LVCMOS33 } [get_ports { OALTB }]; #IO_L17N_T2_A25_15 Sch=led[2]
```

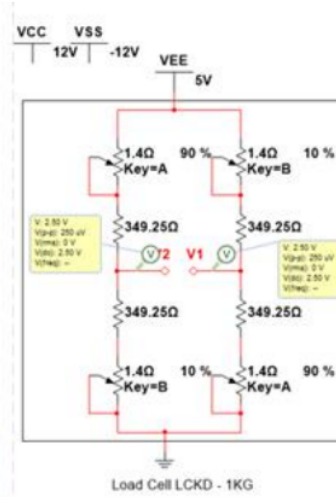
Constraints file is listed, however, there is no explanation or reference from the main report to the io constraints

A3 – Load cell tests

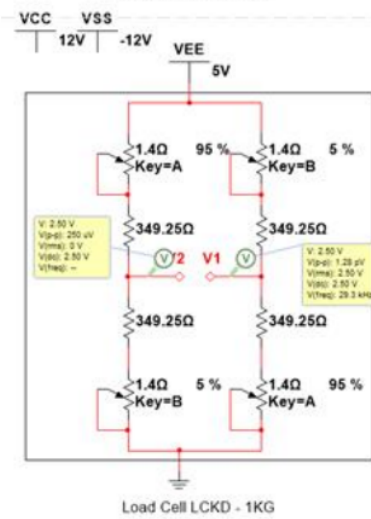




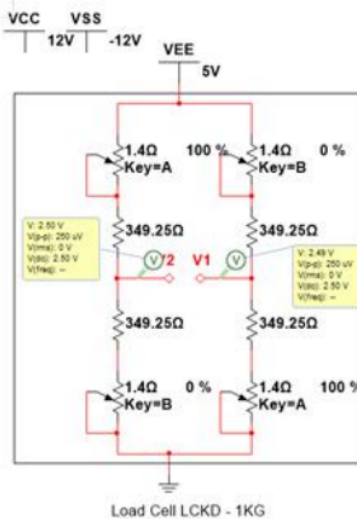
V: 3.50 V
V(p-p): 15.4 nV
V(rms): 3.50 V
V(dc): 3.50 V
V(freq): 23.1 kHz



V: 4.00 V
V(p-p): 3.72 nV
V(rms): 4.00 V
V(dc): 4.00 V
V(freq): 25.2 kHz



V: 4.50 V
V(p-p): 8.69 nV
V(rms): 4.50 V
V(dc): 4.50 V
V(freq): 20.3 kHz



V: 5.00 V
V(p-p): 13.7 nV
V(rms): 5.00 V
V(dc): 5.00 V
V(freq): 21.4 kHz

Good test evidence

A4 – 4bit adder testbench simulation

```

59 stimuli : process
60 begin
61     -- EDIT Adapt initialization as needed
62     cin <= '0';
63     a0 <= '1';
64     a1 <= '0';
65     a2 <= '1';
66     a3 <= '1';
67     b0 <= '0';
68     b1 <= '0';
69     b2 <= '0';
70     b3 <= '1';
71     wait for 100 ns;
72
73     cin <= '0';
74     a0 <= '1';
75     a1 <= '1';
76     a2 <= '1';
77     a3 <= '0';
78     b0 <= '1';
79     b1 <= '0';
80     b2 <= '1';
81     b3 <= '0';
82     wait for 100 ns;
83
84     cin <= '0';
85     a0 <= '1';
86     a1 <= '1';
87     a2 <= '0';
88     a3 <= '1';
89     b0 <= '0';
90     b1 <= '0';
91     b2 <= '0';
92     b3 <= '1';
93     wait for 100 ns;
94
95     cin <= '0';
96     a0 <= '1';
97     a1 <= '0';
98     a2 <= '1';
99     a3 <= '1';
100    b0 <= '1';
101    b1 <= '1';
102    b2 <= '1';
103    b3 <= '0';
104    wait for 100 ns;
105
106    -- EDIT Add stimuli here
107
108    wait;
109 end process;
110
111 end tb;

```

Good range of test data

4-bit adder

A5 – 4bit Magnitude Comparator testbench simulation

```
60  stimuli : process
61  begin
62      -- EDIT Adapt initialization as needed
63      B3 <= '1';
64      B2 <= '1';
65      B1 <= '0';
66      B0 <= '0';
67      A3 <= '1';
68      A2 <= '0';
69      A1 <= '0';
70      A0 <= '0';
71      AGTB <= '0';
72      AEQB <= '1';
73      ALTB <= '0';
74  wait for 100ns;
75      -- EDIT Add stimuli here
76
77      B3 <= '1';
78      B2 <= '1';
79      B1 <= '0';
80      B0 <= '1';
81      A3 <= '1';
82      A2 <= '1';
83      A1 <= '0';
84      A0 <= '1';
85      AGTB <= '0';
86      AEQB <= '1';
87      ALTB <= '0';
88  wait for 100ns;
89
90      B3 <= '0';
91      B2 <= '1';
92      B1 <= '0';
93      B0 <= '0';
94      A3 <= '1';
95      A2 <= '1';
96      A1 <= '0';
97      A0 <= '0';
98      AGTB <= '0';
99      AEQB <= '1';
100     ALTB <= '0';
101  wait for 100ns;
102     wait;
103  end process;
104
105 end tb;
```

Good range of test data

A6 – Data logs for load cell graph

LabVIEW Measurement		700	0	0	0
Writer_Version	2	700	0	0	200
Reader_Version	2	700	0	0	200
Separator	Tab	700	0	0	200
Decimal_Separator	.	700	0	0	200
Multi_Headings	No	700	0	0	200
0	719.588127	700	0	0	200
0.101853	719.877217	700	0	0	200
0.202846	719.812975	700	0	0	200
0.303987	719.652369	700	0	0	200
0.404714	719.812975	700	0	0	200
0.503782	719.97358	700	0	0	200
0.604557	719.97358	700	0	0	200
0.70381	720.005701	700	0	0	200
0.804582	720.26267	700	0	0	200
0.904527	719.877217	700	0	0	200
1.004384	720.326912	700	0	0	200
1.103884	720.134185	700	0	0	200
1.203839	720.326912	700	0	0	200
1.304469	719.652369	700	0	0	200
1.405008	720.359033	700	0	0	200
1.506625	720.519638	700	0	0	200
1.60749	720.037822	700	0	0	200
1.70553	720.069943	700	0	0	200
1.806318	720.069943	700	0	0	200
1.906769	719.780854	700	0	0	200
2.008081	719.97358	700	0	0	200
2.107551	720.166306	700	0	0	200
2.207386	719.845096	700	0	0	200
2.30873	719.812975	700	0	0	200
2.40947	719.780854	700	0	0	200
2.507967	719.877217	700	0	0	200
2.609486	719.716611	700	0	0	200
2.709698	719.588127	700	0	0	200
2.810257	719.556006	700	0	0	200
2.910323	720.134185	700	0	0	200
3.010571	719.877217	700	0	0	200
3.109719	720.037822	700	0	0	200
3.210257	720.230548	700	0	0	200
3.310122	720.037822	700	0	0	200
7.422461	678.698119	700	200	0	0
7.522798	678.794481	700	200	0	0
7.624549	678.569635	700	200	0	0
7.723823	678.569635	700	200	0	0
7.824277	678.73024	700	200	0	0
7.924506	678.569635	700	200	0	0
8.024218	678.826602	700	200	0	0
8.123487	678.794481	700	200	0	0
8.224376	678.505393	700	200	0	0
8.322965	678.794481	700	200	0	0
8.424245	678.505393	700	200	0	0
8.523338	678.665998	700	200	0	0
8.624337	678.698119	700	200	0	0
8.723319	678.633877	700	200	0	0
8.824844	678.922965	700	200	0	0
8.925374	678.794481	700	200	0	0
9.025214	678.73024	700	200	0	0
9.125	678.601756	700	200	0	0
9.225505	678.633877	700	200	0	0
9.324081	678.569635	700	200	0	0
9.425025	678.826602	700	200	0	0
9.525585	678.505393	700	200	0	0
9.626588	678.762361	700	200	0	0
9.72663	678.473272	700	200	0	0
9.826314	678.633877	700	200	0	0
9.926984	678.73024	700	200	0	0
10.026922	678.73024	700	200	0	0
10.127712	678.473272	700	200	0	0
10.226546	678.441151	700	200	0	0
10.328818	678.569635	700	200	0	0
10.428476	678.537514	700	200	0	0
10.527792	678.665998	700	200	0	0
10.628572	678.890844	700	200	0	0
10.727793	678.633877	700	200	0	0
10.828516	678.665998	700	200	0	0
10.930959	678.569635	700	200	0	0
11.030822	678.473272	700	200	0	0
11.12973	679.11569	700	200	0	0
11.23182	678.890844	700	200	0	0
11.328937	678.665998	700	200	0	0

3.410111	719.812975	700	0	0	200
3.510382	719.941459	700	0	0	200
3.609796	719.845096	700	0	0	200
3.70999	719.748733	700	0	0	200
3.810147	719.652369	700	0	0	200
3.910829	719.877217	700	0	0	200
4.011827	720.294791	700	0	0	200
4.11281	719.620248	700	0	0	200
4.212821	719.845096	700	0	0	200
4.311732	719.812975	700	0	0	200
4.414909	720.005701	700	0	0	200
4.515486	720.069943	700	0	200	0
4.614898	719.941459	700	0	200	0
4.715405	719.68449	700	0	200	0
4.813955	701.375507	700	0	200	0
4.914789	700.540361	700	0	200	0
5.012733	700.283393	700	0	200	0
5.11432	700.861571	700	0	200	0
5.214305	700.861571	700	0	200	0
5.314826	701.15066	700	0	200	0
5.414196	700.925813	700	0	200	0
5.514053	700.957934	700	0	200	0
5.615189	700.797329	700	0	200	0
5.7154	700.636724	700	0	200	0
5.816147	700.572482	700	0	200	0
5.916821	700.700966	700	0	200	0
6.016731	700.82945	700	0	200	0
6.117413	700.379756	700	0	200	0
6.217765	700.797329	700	0	200	0
6.317237	700.572482	700	200	0	0
6.416758	700.797329	700	200	0	0
6.519423	700.604603	700	200	0	0
6.62232	690.004684	700	200	0	0
6.718557	679.051449	700	200	0	0
6.82005	678.601756	700	200	0	0
6.920279	678.698119	700	200	0	0
7.02139	678.473272	700	200	0	0
7.122142	678.601756	700	200	0	0
7.222382	678.826602	700	200	0	0
7.322236	678.633877	700	200	0	0

11.430336	678.826602	700	200	0	0
11.530855	678.698119	700	200	0	0
11.630817	679.11569	700	200	0	0
11.731817	678.633877	700	200	0	0
11.832102	678.537514	700	200	0	0
11.931654	444.828712	700	200	0	0
12.031986	197.505847	700	200	0	0
12.132692	113.095712	700	200	0	0
12.233481	86.147489	700	200	0	0
12.3344	76.832853	700	200	0	0
12.432935	73.042761	700	200	0	0
12.534024	71.404671	700	200	0	0
12.633823	70.89076	700	200	0	0
12.73457	70.505327	700	200	0	0
12.833341	70.376849	700	200	0	0
12.933924	70.184133	700	200	0	0
13.034144	70.184133	700	200	0	0
13.132818	70.023536	700	200	0	0
13.233159	69.638103	700	200	0	0
13.333862	69.7987	700	200	0	0
13.435945	69.895058	700	200	0	0
13.534386	69.895058	700	200	0	0
13.635377	69.670222	700	200	0	0
13.734357	69.573864	700	200	0	0
13.834319	69.7987	700	200	0	0
13.933981	69.76658	700	200	0	0
14.035343	69.477506	700	200	0	0

Good evidence of data logs

There should be evidence of the .lvm text files generated in the appendix

1302736

GRADEMARK REPORT

FINAL GRADE

GENERAL COMMENTS

Instructor

53 / 100

PAGE 1

PAGE 2

PAGE 3

Text Comment. Good detailed table of contents

PAGE 4

PAGE 5

Text Comment. Comprehensive list of figures

Text Comment. The use of tables to present test results is a preferred method

PAGE 6

Text Comment. Introduction indicates an understanding of the requirements of the element

PAGE 7

Text Comment. Good points made and supported with relevant source

Text Comment. Good example. This should be referenced in the text content, and significant parts i.e. entity and architecture should be explained.

Text Comment. Good example

Text Comment. Image should have a ref source

PAGE 8

Text Comment. Significant points described about VHDL. However, the IP design has not been explained

Text Comment. Good informative research carried out and relevant supporting image. The image should have a ref source included

PAGE 9

PAGE 10

Text Comment. The significant LabVIEW blocks are identified and explained

PAGE 11

Text Comment. Good example of digital comparator

Text Comment. An understanding is presented, however, the expansion possibilities for this could have been explained

PAGE 12

Text Comment. Good explanation with relevant source

PAGE 13

Text Comment. The alternative methods of ADC are not explained

PAGE 14

Text Comment. Good explanation with supporting image

PAGE 15

Text Comment. Signal conditioning applications are identified, and indicate an understanding of the practical application

Text Comment. Good table of examples

Text Comment. The instrument amplifier purpose could have been explained further due to the relevance in this element

Text Comment. Significant block diagram is identified with all the critical io elements needed for design

Text Comment. Hierarchy of design elements is identified here

Text Comment. This does not belong here. This is the test result from simulation, and you have included this in the test and analysis section.

Text Comment. The entity is explained here, that specifically describes the io. The architecture that includes the circuit function code is missing

Text Comment. This needs an explanation of how it was designed and generated

Text Comment. This should be explained!

Text Comment. Heading here is incorrect!

Text Comment. 8 bit magnitude comparator sim test is explained

Text Comment. The IP design here is in the incorrect place in the report.

Text Comment. PLD design is correct with constants included

Text Comment. Test result is very good with 'cout' demonstrated and explained

Text Comment. Good set of results for all 3 output possibilities

Text Comment. Tables could have been used here to present the results more effectively

PAGE 25

Text Comment. Heading should be '4 bit adder and 4 bit mag comparator combination'.

Text Comment. Test is only part evident in the image, part of the switch setting layout is not in the image.

Text Comment. Test is only part evident. Part of the switch settings is not evident in the image

Text Comment. Identification is not correct. This should be the 4 bit adder and comparator combination

PAGE 26

Text Comment. All switch settings are evident and test is correct

Text Comment. The tests carried out are correct. You have confused the identification with the 8 bit comparator. These 3 tests are the 4 bit adder and 4 bit mag comparator combination.

Text Comment. The test needs to be explained further!

Text Comment. The heading here is incorrect. This should be the 8 bit comparator

Text Comment. The test carried out is correct and is evident in the image

Text Comment. Confusion with image identification. This is the 8 bit comparator only

PAGE 27

Text Comment. Test is correct and evident in the image

Text Comment. Incorrect identification. This is the 8 bit comparator

Text Comment. Test is correct and evident in the image

Text Comment. This is the 8 bit comparator

PAGE 28

Text Comment. This explanation is incomplete!

Text Comment. Evaluation does not point out any significance for the subsequent elements or include a recommendation

PAGE 29

Text Comment. This is an incorrect symbol '*' to use in this context

PAGE 30

Text Comment. Good image indicating purpose of amplifier

Text Comment. Good information about load cell spec

PAGE 31

Text Comment. Good explanation and supporting image with initial conditions

Text Comment. Good identification of spec requirements

Text Comment. Incorrect value here. You have used 500k and 5k resistor values

PAGE 32

Text Comment. Good set of test results

PAGE 33

Text Comment. Good linear graph produced from results

PAGE 34

Text Comment. Correct block program for 10 bit resolution

PAGE 35

Text Comment. Good test sample

Text Comment. Good max scale sample test

PAGE 36

Text Comment. Block program has been correctly configured for 12 bit resolution

Text Comment. Good test sample

PAGE 37

Text Comment. Good max scale sample test

Text Comment. 8 bit ADC prototype with virtual input is missing

PAGE 38

Text Comment. This heading is incorrect

Text Comment. The program block design is annotated correctly

Text Comment. Good explanation of the shifted 8 bit value

PAGE 39

Text Comment. Test is successful and evident

PAGE 40

Text Comment. All 3 tests are successful with good evidence

Text Comment. Good use of table of results

PAGE 41

PAGE 42

Text Comment. Good image showing practical setup using PMOD ports of FPGA

Text Comment. Good connection key that demonstrates a good practical understanding of the interface connections for FPGA and LabVIEW with DAQmx

PAGE 43

Text Comment. There is no need to have duplicated images of the block program

Text Comment. Formula blocks are explained

Text Comment. Good example that demonstrates an understanding

Text Comment. Good front panel design

Text Comment. Test is successful and evident in the image

Text Comment. second test is evident

Text Comment. All 3 tests have been included with good clear evidence

Text Comment. This is a duplicate of the earlier table

Text Comment. There is no need for the duplicates of the test results

Text Comment. A poor evaluation

Text Comment. Element 3

Text Comment. The intro includes some good significant points relevant to the extended element 2 block program design

Text Comment. This is not a good explanation!

Text Comment. Figures need to be identified specifically

Text Comment. This is a poor explanation

Text Comment. This is a good sample of the write file configuration

Text Comment. Scaling is explained with a ref to evidence

Text Comment. Good graph of resultant data form test .lvm files

Text Comment. Good explanation of resultant graph

Text Comment. A good point made here

Text Comment. You only need to evaluate element 3 here

Text Comment. Good number of relevant reference sources

Text Comment. The architecture should be explained and referenced from the main report. The architecture is not explained

Text Comment. Constraints file is listed, however, there is no explanation or reference from the main report to the io constraints

Text Comment. Good test evidence

Text Comment. Good range of test data

Text Comment. Good range of test data

Text Comment. Good evidence of data logs

Text Comment. There should be evidence of the .lvm text files generated in the appendix

GRADING FORM: CSG-RUBIC

JASON BOOTH

0

RESEARCH-INFORMED LITERATURE



Research covers good significant points in most areas. The IP design and mag comparator could have been researched further, and the ADC is restricted to one method. There are a good number of sources used in the research. 6/10

6/10

KNOWLEDGE AND UNDERSTANDING



Element 1 requirements have been mostly explained. However, some sections have misplaced and disjointed content, VHDL is only part explained, and only basic explanations given to practical test results. There are complete test results with parts of the image evidence missing. 5/10 Element 2 instrument amplifier has incorrect resistor values in the calculation. The 8 bit ADC with virtual input is missing. Some content is disjointed with incorrect image identification. There is a sequence of events and all 3 test are included. There is duplicated content in this section. 5.5/10 Element 3 explanations are poor in

15/30

most areas . The write files are poorly explained and the .lvm text file evidence is missing. The test results are present and a graph has been produced from the data. 4.5/10

ANALYSIS



Element 1 test results have been achieved and indicate a good method using the virtual bus method and range of test data. Parts of the practical test evidence is disjointed and missing. Evaluation covers significant points. 2.5/5 Element 2 test results are explained and include a good range of values. 8 bit ADC is missing. Good test results from the instrument amplifier and load cell sim. The evaluation is poor. 5.5/10 Element 3 test results are successful and a good graph is produced from the test data. The explanations are poor and evidence of the .lvm files is missing. Evaluation includes some good relevant points and a recommendation. 5/10

13/25

PRACTICAL APPLICATION AND DEP.



Element 1 design and build has been achieved, however the explanation of the ip design is only partly covered. Ther is disjointed content and explanations are far too basic in some areas. There is evidence of confusion in the identification of the FPGA parctical tasks. 2.5/5 Element 2 interfacing methods used indicate an understnding of the system functions. The 8 bit should have been included as part of the process. 3/5 Element 3 data logging methods have been implemented with a

11/20

scaler value for the output, however, the explanations are poor in this section. 5.5/10

SKILLS FOR PROFESSIONAL PRAC.



Element 1 has most of the requirements, there are some minor grammatical errors and there is disjointed content. 3/5 Element 2 has misleading headings and the sequence is disjointed in parts. There is duplicated content in this section. 2.5/5 Element 3 has test results in good order, however there is evidence missing and poor reference to existing evidence in appendix. 2.5/5

8/15