

2024421

by Matthew Perry

Submission date: 28-Feb-2023 08:23PM (UTC+0000)

Submission ID: 198176533

File name: 16803_Matthew_Perry_2024421_181628_1241545396.pdf (4.29M)

Word count: 8358

Character count: 50955

Course: HND Electrical and Electronic Engineering

Module: SGEE5007 Virtual Instrumentation and Control

Portfolio: Magnitude Comparator; Element 1

Name: Matthew Perry

University number: 2024421

Date submitted: 28/02/2023

Contents

1. Introduction.....	4
2. Research	5
2.1 Digital design methods using VHDL and IP integration.....	5
2.2 NI DAQmx	6
2.3 Digital Magnitude Comparators	8
2.4 Analogue to Digital Converters	9
2.5 Load Cells	9
2.6 Instrumentation and Signal Conditioning	11
3. Design requirements.....	12
3.1 4-Bit Adder.....	12
3.2 4-bit Magnitude Comparator	14
3.3 4-bit adder with 4-bit Comparator IP design.....	15
3.4 8-bit Magnitude Comparator	16
4. Implementation	17
5. Testing and Analysis	18
5.1 4 -bit Adder.....	18
5.2 4-bit magnitude Comparator	19
5.3 4-bit Adder with 4-bit Comparator IP.....	20
5.4 8-Bit Magnitude Comparator.....	23
6. Evaluation	25
7. Appendix.....	26
8. Works Cited	31

Good detailed table of contents

Figure 1; NI DAQmx Virtual Channel.....	6
Figure 2; Creating a Channel	6
Figure 3; NI DAQ-MX Timing	6
Figure 4; NI-DAQ-mx Read	7
Figure 5; NI DAQ-mx Write.	7
Figure 6; NI-DAQ-mx Formula	7
Figure 7; NI DAQ-mx Clear Task	7
Figure 8; Example of Magnitude Comparator.....	8
Figure 9; Example of ADC Operation.....	9
Figure 10; Successive Approximation ADC Design.....	9
Figure 11; Omega LCKD Example Image	10
Figure 12; 4-Bit Adder Operation	12
Figure 13; Vivado 4-Bit Hierarchy.....	12
Figure 14; 4-Bit IP Integrator Design	13
Figure 15; 4-Bit Magnitude Design PLD	14
Figure 16; 4-Bit Magnitude Comparator IP Design	14
Figure 17; Adder 4-Bit Combined 4-Bit Magnitude IP Design.....	15
Figure 18; PLD 8-Bit Comparator	16
Figure 19; Image of NEXYS 4 DDR	17
Figure 20; 4-Bit Adder Test Results	18
Figure 21; 4-Bit Magnitude Comparator Results.....	19
Figure 22; 4-Bit Magnitude Comparator Simulated Results.....	20
Figure 23; 4-bit Adder with 4-bit Comparator IP Actual Test3 1	21
Figure 24; 4-bit Adder with 4-bit Comparator IP Actual Test 4.....	22
Figure 25; 8-Bit PLD Simulated Test	23
Figure 26; 8-Bit Test Practical.....	24
Figure 27; 4-Bit Adder & 4-Bit Comparator Test 2.....	26
Figure 28; Example of Intellectual Properties	26
Figure 29 ; 4-Bit Adder & 4-Bit Comparator Test 3	27
Figure 30; 8-Bit Comparator Test 2	27
Figure 31; 4-Bit Adder Hierarchy Example	28
Figure 32; 8-Bit Comparator Test 3	28
Figure 33; Constraints File 4-Bit Adder & 4-Bit Mag	29
Figure 34; Constraints File 4-Bit Adder	29
Figure 35; 4-Bit Adder Magnitude Test Bench.....	30
Figure 36; 4-Bit Adder Test Bench Conditions.....	30

Good detailed list of figures

Table 1;Simulation Results 4-Bit Adder.....	18
Table 2;4-Bit Magnitude Results	19
Table 3;Simulated 4-Bit and 4-Bit Magnitude Combined	20
Table 4;4-Bit Adder with 4-Bit Comparator Test	21
Table 5;8-Bit Comparison Table Results	24

Good list of tables

1. Introduction

This portfolio will consist of 3 elements to obtain a comparison result of a 1Kg- Load cell via various methods. There will be two signals, one input representing the load of 0-1000 grams using LabVIEW and another input will be an 8-bit magnitude comparator created from Multisim. For the final system to operate functionally, each stage will be designed and tested as appropriate.

Additionally, research into areas that will be used to support the completion of the system will be carried out.

An appropriate introduction

2. Research

2.1 Digital design methods using VHDL and IP integration.

A digital design method that could be implemented to integrate with the system is an FPGA in addition with LabView.

LabView is a software development environment and language that is created by National Instruments. It uses graphical diagrams which creates an intuitive for engineers and beneficial for parallel programming (McNally, 2022).

The Nexys 4 DDR FPGA software allows the Field Programmable Gate Array (FPGA) to be programmed. An FPGA is the physical circuit that is made up of semiconductor devices that are based around an array of configurable logic blocks that are connected by programmable interconnects (XILINX, 2021). The programming language the software uses is Very High-Speed Integrated Circuit Hardware Description Language (VHDL), this language can be characterised as a text-based language that describes or express logic circuits, additionally it can be used as a general-purpose parallel programming language (CADENCE PCB SOLUTIONS, 2021).

FPGAs are still being developed further to maintain current use, recently IP (Intellectual Properties) design has been implemented. An IP is a separate module that can be used with any FPGA and connected via Wizard in Vivado IDE. Although it is stated as a separate module but can be a large system in their own right (S.R. Ravikiran, 2017). A visual example of an IP looks like is shown in Figure 28.

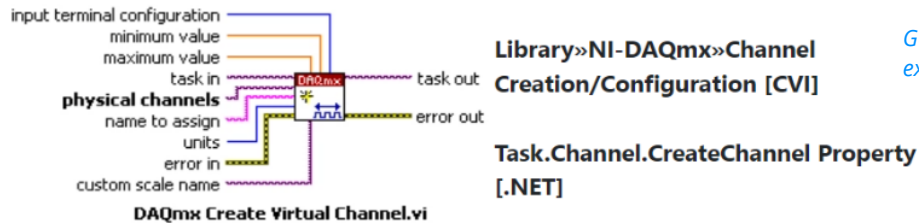
Good significant points with supporting ref sources, and image example from your own design.

2.2 NI DAQmx

NI-DAQmx is the driver software that allows students/engineers to communicate with and control the NI data acquisition devices. This software driver has an extensive library of functions and VIs that can be called from LabView (NI, 2023).

Some minor grammatical errors

Some samples of the main blocks used within the program are investigated below:



Good informative explanation and image

Figure 1; NI DAQmx Virtual Channel

The NI-DAQmx Create Virtual Channel function creates a virtual channel and adds it to a task it can also be used to create multiple virtual channels and compile them to a task. When a task is not specified, the function creates a task. The NI-DAQmx Create Virtual Channel function has numerous instances. These instances correspond to the specific type of measurement or generation the virtual channels perform.

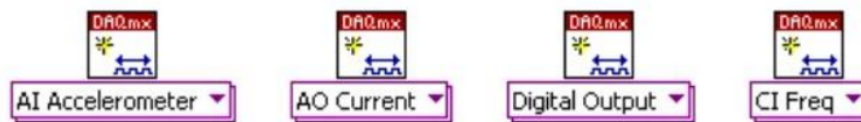


Figure 2; Creating a Channel

The main significant function blocks have been identified here

Creating an input function can differ for each instance of the function. For example:

- Physical Channels – Analog & Digital
- Lines - Digital
- Counter – Virtual Channels

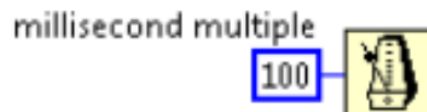


Figure 3; NI DAQ-MX Timing

There are various clocks that can be used, however figure 3 shows a timer that is used to synchronise tasks in a sequence within the while loop.

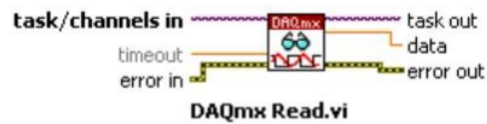


Figure 4; NI-DAQ-mx Read

The DAQmx read function reads a sample from the specified acquisition task. Depending on the instances of the function determines the acquisition such as analog, digital or counter. Once the samples have been completed, it then moves the samples from the DAQ board to the PC buffer and then further from the PLC to the Application Development Environment (ADE) memory.

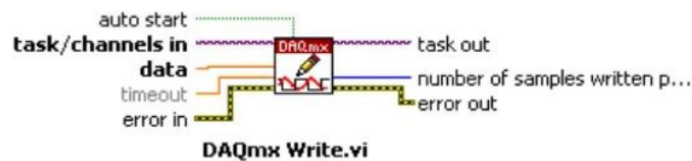


Figure 5; NI DAQ-mx Write.

The NI-DAQmx write function writes samples to the specified generation task. Depending on the instance of the function it allows for the appropriate type of generation (analogue or digital), number of virtual channels, data to be selected and the number of samples.

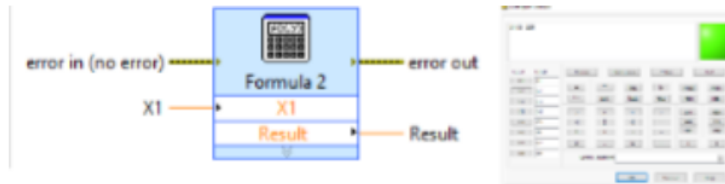


Figure 6; NI-DAQ-mx Formula

The formula blocks allow the signals to be manipulated by using a mathematical model.

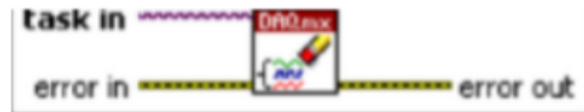


Figure 7; NI DAQ-mx Clear Task

The clear task function will clear the existing task to free up memory. Due to multiple tasks being carried out it is not possible to continue the same task until it is completed and cleared (NI , 2022)

Main blocks have been identified and explained which indicate a good understanding of requirements from LabVIEW and DAQmx

2.3 Digital Magnitude Comparators

A digital magnitude comparator is a combinational circuit that compares two digital or binary numbers. By comparing the numbers, the circuit can then specify whether one binary number is less than, equal to or greater than the compared binary number (GFG, 2022).

Shown below where A & B represent a binary number as an input and the output from the comparator will give a result.



Good example and ref source

Figure 8; Example of Magnitude Comparator

2.4 Analogue to Digital Converters

An analogue to digital converter (ADC) allows micro-processor-controlled circuits to communicate with the real world. Analogue signals can change their value continuously and come from various sources and sensors in particular this portfolio will be using a load cell. Digital circuits only consist of two discrete values which are 0 – low and 1 – high. Due to the difference, it is important for electronic circuits to be able to convert between continuously changing analogue signals and discrete digital signals (Electronic Tutorials , 2022).

An ADC carries out a conversion by some form of quantization essentially mapping the continuous set of analogue values to a set of smaller digital values often by rounding as shown below.

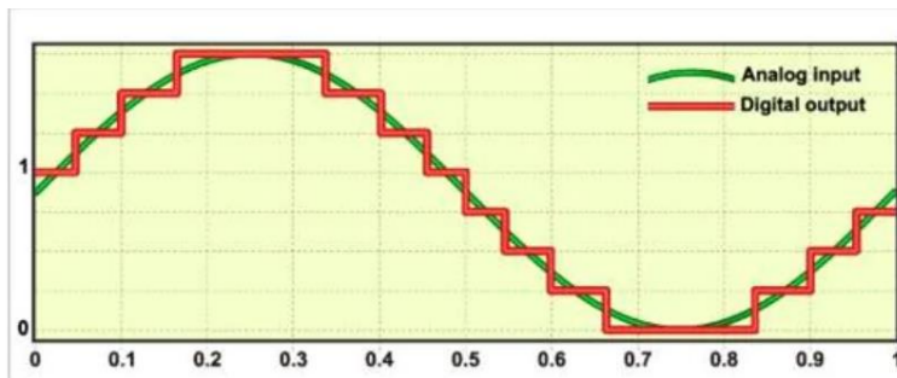


Figure 9; Example of ADC Operation

One of the most widely used ADC is called successive approximation. This type is a low-cost application of medium to high resolution applications (8-18 bits) with a sample speed up to 5 mega samples per second. A design of a successive approximation ADC is shown below (Das, 2020). This diagram shows that the SAR ADC which consists of a comparator, ADC, SAR, and control circuit. When a new conversion starts, the sample and hold circuit samples the input signal which is then compared with the specific output signal of the DAC.

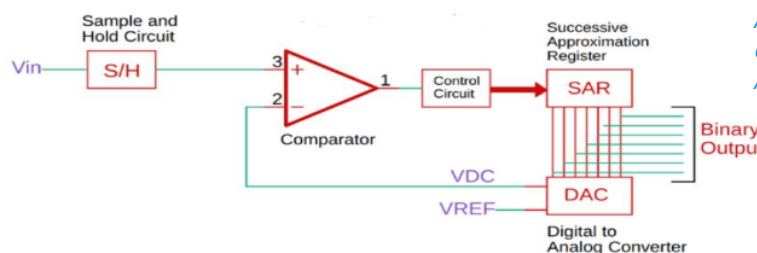


Figure 10; Successive Approximation ADC Design

is a transducer that is used to convert force into a electrical signal that can be highly

accurate device that allows the device to measure weight or force for a number of applications (OLCM Systems, 2020).

There are four main types of load cells, these are pneumatic, hydraulic, capacitance and strain gauge. The strain gauge would be the most suitable for this project due to their size which is usually small with a high frequency bandwidth, financially friendly, low maintenance and precise accuracy (Thorat, 2019). The sensor operates by consisting of a metal foil, when the force is applied the resistance within the foil varies. The resistance is not being measured; however, it is the change in voltage caused by the change in resistance. This is *Good explanation relative to task* proportional and is read from a bridge circuit to produce a voltage output (Morris, 2001)

An example of a specific load cell that could be used for the application is series sub miniature compression load cell which can measure from 1KG to 1000lb with an excitation of 5Vdc (Omega, 2019). A diagram of the sensor is shown below.

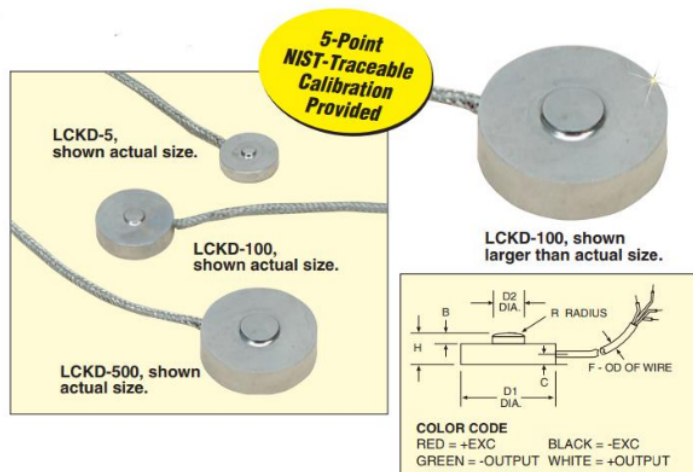


Figure 11; Omega LCKD Example Image

Good practical example

2.6 Instrumentation and Signal Conditioning

An instrumentation system obtains data about a physical system either for the motive of collecting information about a physical system or to be used for the feedback control of the physical system. In the case of this portfolio, the instrumentation system would be made up of an input transducer (Strain gauge load cell) and another component such as a digital processor, computer, or micro-controller (Collins, 2012).

The signal that will be produced by the input would be low voltage, due to the low voltage analogue signal it will have to be amplified and filtered to reduce background noise before digitization (Oxley, 2020). In addition to amplifying and filtering the transducer may need to be excited by an external voltage to produce a signal (Peterson Publications, 2011)

There are many factors that can affect an instrument, however a signal conditioner impacts the characteristics in relation to the selected transducer, such as:

- Sensitivity – This is the minimum amount of force needed to change the load cells output. Due to the excitation and output voltage this can then impact the sensitivity.
- Resolution – This refers to the load cells capability to produce an output relative to the detected change.
- Noise – This can be produced by random non grounded electrical signals produces by components that can impact the accuracy of the signal produced by the load cell. This can be filtered via the signal conditioner (Williams, 2010).
- Span – The minimum and maximum values the input transducer can measure, hence scaling.
- Linearity – Most instruments provide a linear output to change, however some instruments provide a non-linear output such as thermocouples. Signal conditioning can be used to process the values for a better linear response.

Good relevant characteristics identified. This could have included a block diagram of a process involving instrumentation and signal conditioning

A good balanced research carried out on each topic area with good relevant sources used. There are an appropriate number of sources researched for the task.

3. Design requirements

3.1 4-Bit Adder

To be able create the 4-Bit adder, smaller circuits were required to create the final form. The smaller circuits consisted of 2 half adders to combined to make a full adder to cascade a total of 4 full adders. A hierarchy example of this layout can be shown in Figure 31.

Firstly, a half adder was created, this is an electronic circuit that performs the addition of numbers. The half adder can sum two single binary digits and present an output plus a carry value (Techopedia , 2020).

Secondly, the two half adders were combined to create one full adder. The full adder can add three binary digits where one is the carry, for example A, B and C-out or also known as 'sum'.

Shown below is an example of how the 4-bit binary operates. C-in is used to receive the carry from a previous stage, then A & B are summed and carried through to the next adder.

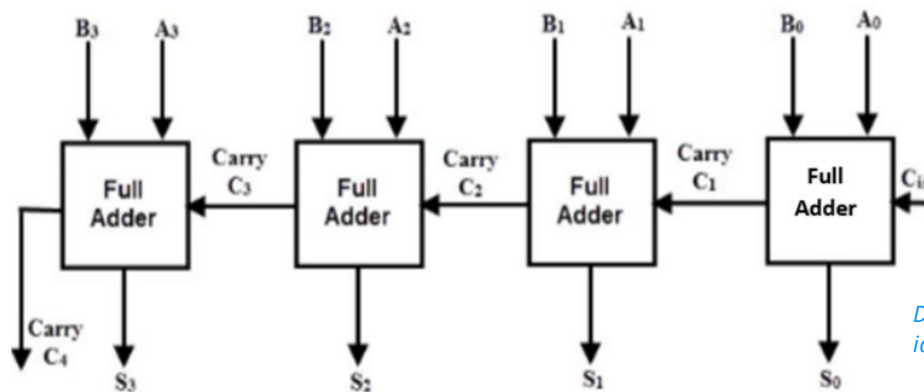


Figure 12; 4-Bit Adder Operation

Design diagram is identified here, and is critical for the VHDL design

Using this logic, the VHDL code was implemented in Vivado to create the half, full and then finally the final 4-Bit adder.

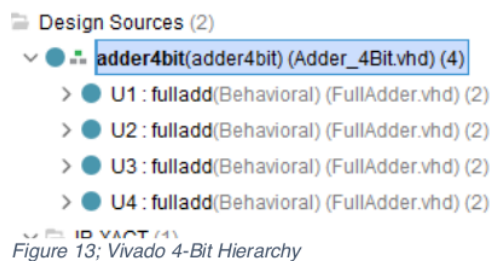


Figure 13; Vivado 4-Bit Hierarchy

VHDL hierarchy is recognised for the system.

The VHDL program explanation is missing

Below shows the IP integrator design for the 4-Bit Adder.

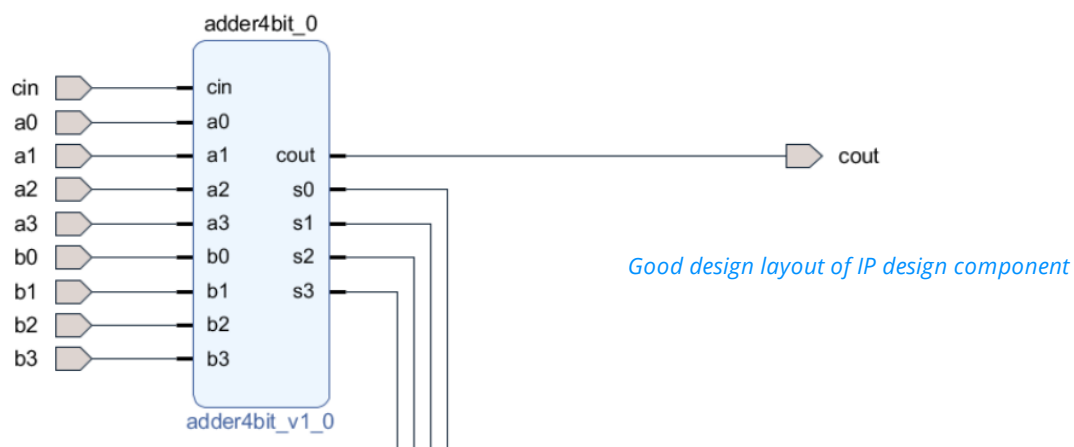


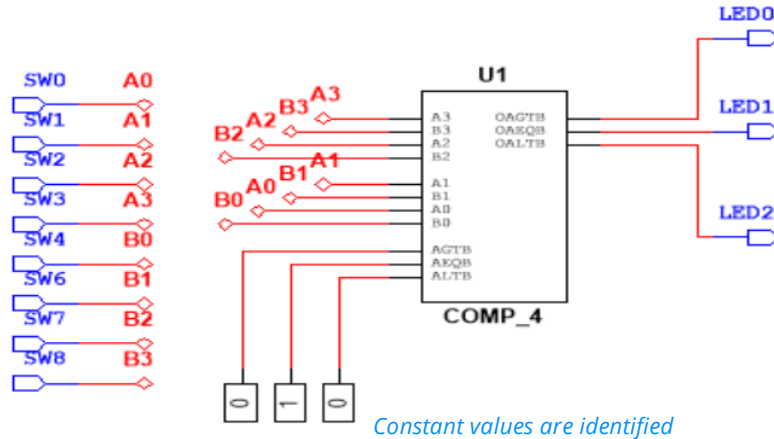
Figure 14; 4-Bit IP Integrator Design

The left side shows the inputs which would be switches on the FPGA and the right side shows the outputs where S0, S1, S2 and S3 are the LEDs. This can be further shown by the [Constraints](#) file used within the software, the relevant code has been highlighted and shown in Figure 34.

3.2 4-bit Magnitude Comparator

By using Multisim PLD mode, the circuit below was created. This shows that the switches are linked to the binary number inputs and then the LEDs are linked to the appropriate output description. Then using this PLD file it was transferred to VHDL code to be implemented into Vivado.

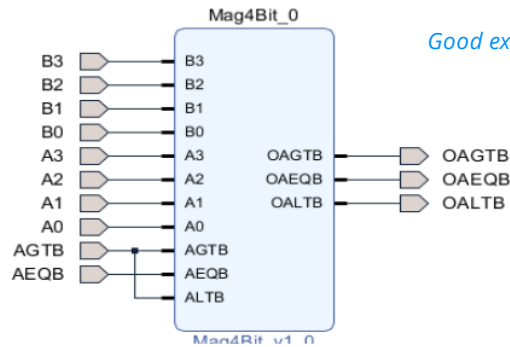
You have identified that the VHDL for this cct has been auto generated and used in Vivado



Constant values are identified

Figure 15; 4-Bit Magnitude Design PLD

Using the code from Multisim, the IP design was created. The switches on the left, versus the LED outputs on the right. Additionally, 'A less than B' is linked with 'A greater than B'.



Good explanation and IP design

Figure 16; 4-Bit Magnitude Comparator IP Design

The constraints file used for this circuit, is the same used in the 4-bit adder design.

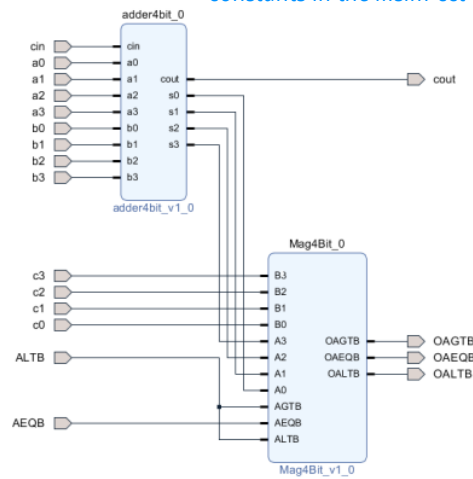
This is true, however, it should be pointed out that the io identifiers in the constraints file must follow the io identifiers in the IP.

3.3 4-bit adder with 4-bit Comparator IP design

The 4-Bit adder was combined with the 4-bit comparator.

Below shows the IP design of the circuit. The 4-Bit adder outputs connect to ports A0 to A3

and ALTB And AGTB are linked together. *This is a good significant point, as this is consistent with the constants in the Msim cct*



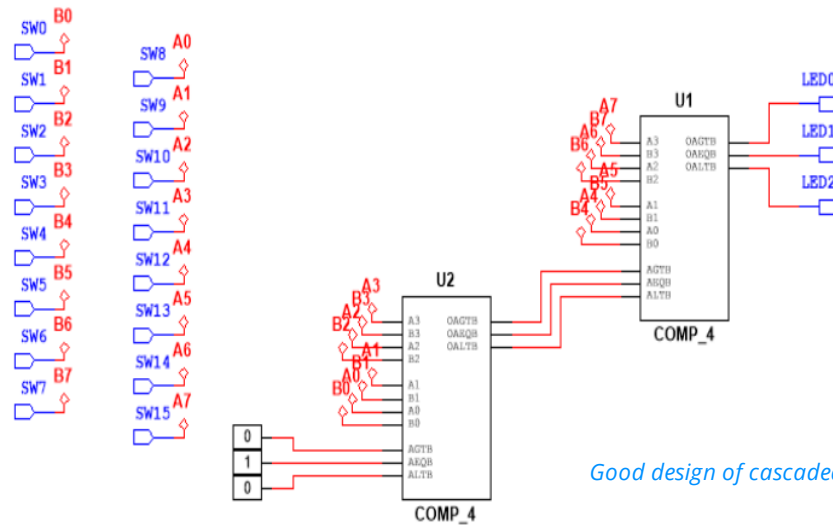
Good combinational system design

Figure 17; Adder 4-Bit Combined 4-Bit Magnitude IP Design

In Figure 33 shows the constraints file, this shows the layout for the pins for the FPGA. The relevant switched and LEDs have been highlighted.

3.4 8-bit Magnitude Comparator

A PLD design was created on Multisim. This shows 2 4-Bit Magnitude Comparators cascaded.



Good design of cascaded 4 bit comparators

Figure 18; PLD 8-Bit Comparator

The constraints file for the 8-bit are the same as the other circuits created.

4. Implementation

As shown from the design stages, the half adder was used to create a full adder and cascaded 4 times. This created the 4-bit adder, using a virtual bus simulator the system was then tested. Additionally using the virtual bus simulator, the 4-Bit Magnitude Comparator was tested.

The 4-Bit Adder and 4-Bit Magnitude Comparator was implemented by downloading to the FPGA NEXYS 4 DDR board. The constraints file shown in the appendix labelled the switches and LEDs addresses.

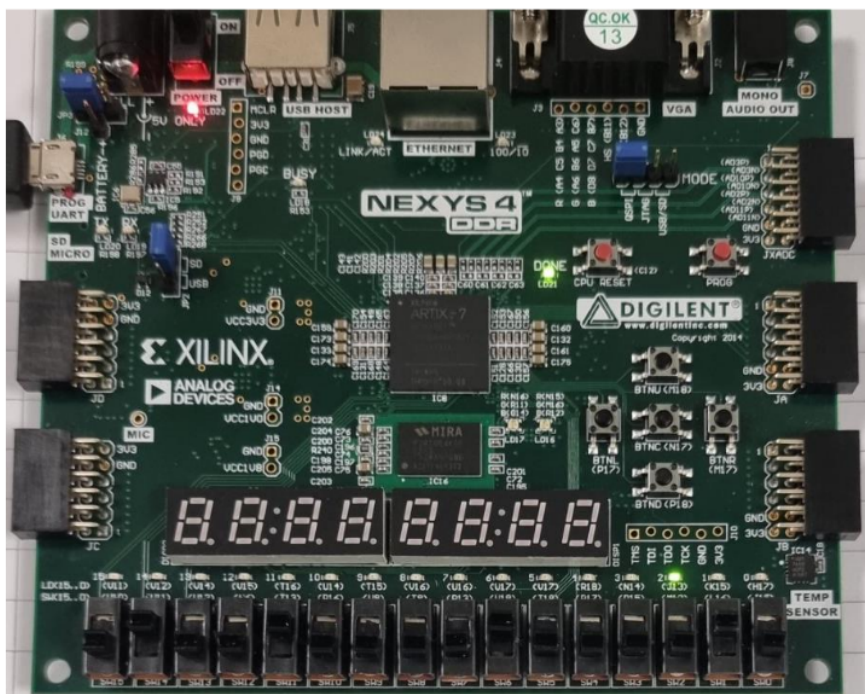


Figure 19; Image of NEXYS 4 DDR

Good clear points for requirements

5. Testing and Analysis

5.1 4-bit Adder

The virtual bus method was used to test the system. Using the code created for the 4-Bit adder it was then used in a online test bench generator. This created a VHDL code, where the conditions could be altered shown in Figure 36.

Below shows the graphical table of results for the tests ran, the tests are shown horizontally for the output for a, b, cout and sout.

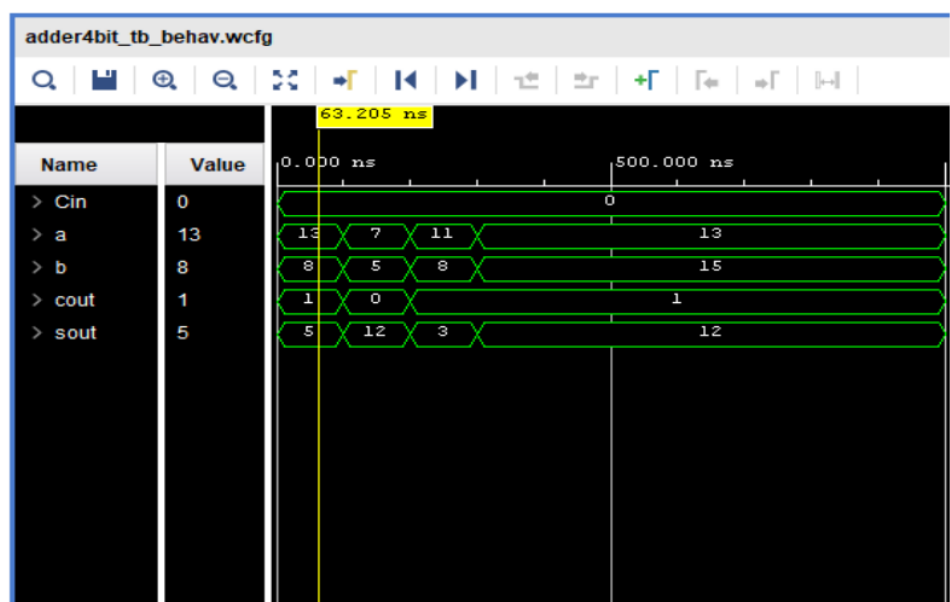


Figure 20; 4-Bit Adder Test Results

Good test results using virtual bus method

From the results, they were sorted into a table which shows each result and an expected result.

Simulation Conditions & Results													Expected Result A	Expected Result B
Conditions	a0	a1	a2	a3	b0	b1	b2	b3	cin	Result A	Result B			
Test 1	1	0	1	1	0	0	0	1	0	13	8		13	8
Test 2	1	1	1	0	1	0	1	0	0	7	5		7	5
Test 3	1	1	0	1	0	0	0	1	0	11	8		11	8
Test 4	1	0	1	1	1	1	1	1	0	13	15		13	15

Table 1;Simulation Results 4-Bit Adder

Cout and sum results should be included. The test results waveforms show the cout carry bit set when the sum exceeds the 4 bit limit

Analysing the results, it shows that the circuit worked as intended by summing the inputs and resulting the correct output. Additionally, the predicted and actual results compared well.

5.2 4-bit magnitude Comparator

Using the virtual bus method, this circuit was tested. There were 3 tests to be carried out which was defined in the test bench VHDL code which was generated online and can be viewed in the appendix under Figure 35. The tests were created to determine 3 states; A less than B, A equal to B and A greater than B. Below displays the results from the Vivado simulation. This shows the decimal values for A & B and where the output goes to state 1, then shows the active output.

Explanation is clear and consistent with results

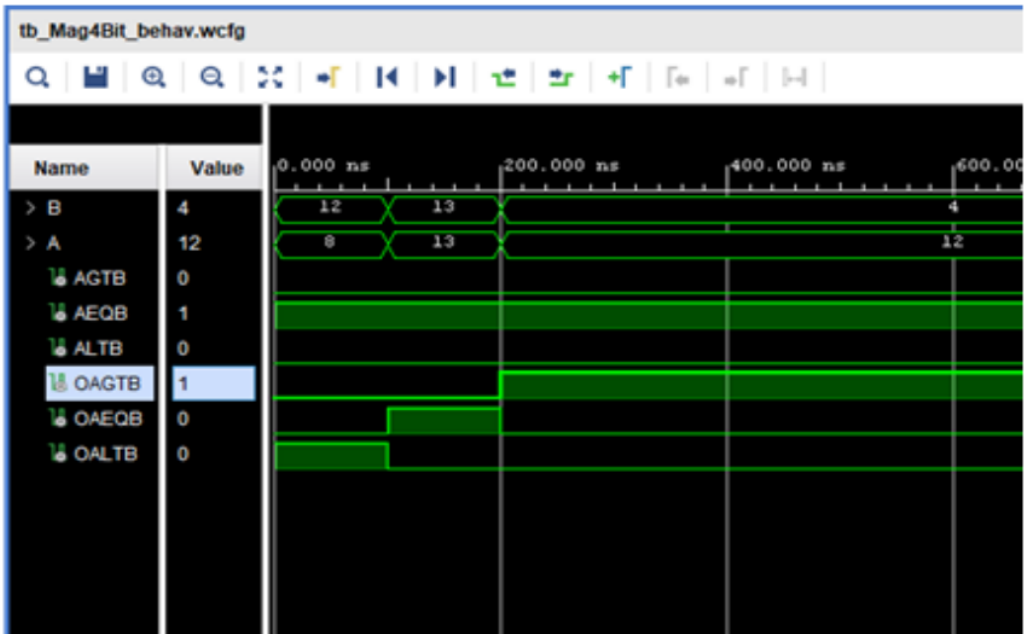


Figure 21; 4-Bit Magnitude Comparator Results

Good tests results and displayed using the virtual bus method

The results were then tabulated for analyses and comparison for predicted and actual values.

Good table of results that follow the actual results, and indicate the 3 conditions required

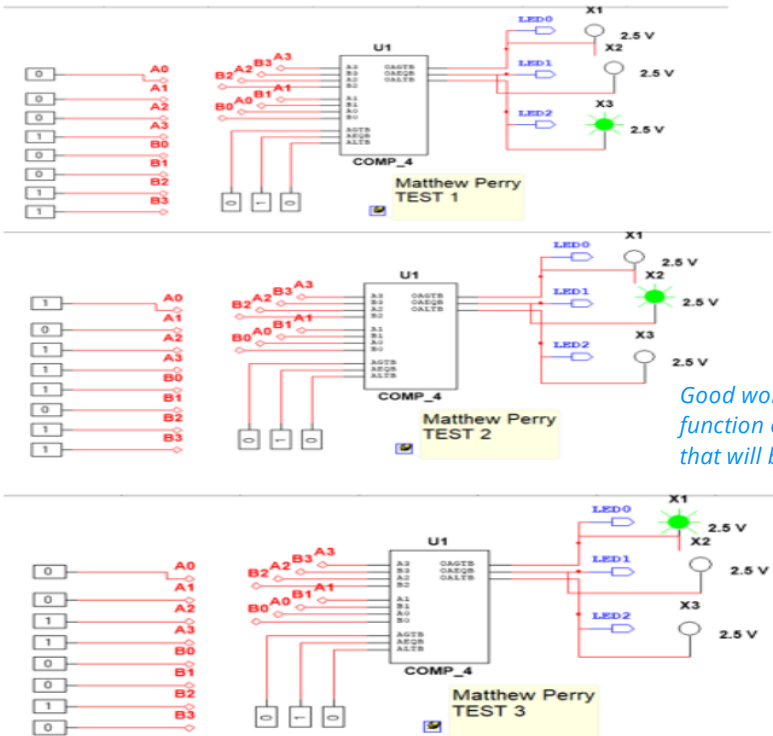
Simulation Conditions & Results 4-Bit Adder Magnitude												
Conditions	a0	a1	a2	a3	b0	b1	b2	b3	Result A	Result B	Output	Expected A Expected B Expected Output
Test 1	0	0	0	1	0	0	1	1	8	12	ALTB	8 12 ALTb
Test 2	1	0	1	1	1	0	1	1	13	13	AEQB	13 13 AEQB
Test 3	0	0	1	1	0	0	1	0	12	4	AGTB	12 4 AGTB

Table 2;4-Bit Magnitude Results

From the results, it shows that the 3 tests were successful. The predicted and actual values were correct which shows that the operation of the circuit worked as intended.

5.3 4-bit Adder with 4-bit Comparator IP

Due to the circuit being created on Multisim PLD mode, the circuit was simulated before implementing the code on the FPGA board. This would ensure that the circuit was operational before further implementation. Using the simulator, the circuit was tested on 3 conditions as shown below.



Good work in identifying the function of the 8 bit comparator that will be used on the FPGA

Figure 22; 4-Bit Magnitude Comparator Simulated Results

Good use of table to show results obtained

Simulation Conditions & Results 4-Bit Adder & 4 Bit Magnitude										
Conditions	a0	a1	a2	a3	b0	b1	b2	b3	Output	Expected Output
Test 1	0	0	0	1	0	0	1	1	ALTB	ALTB
Test 2	1	0	1	1	1	0	1	1	AEQB	AEQB
Test 3	0	0	1	1	0	0	1	0	AGTB	AGTB

Table 3; Simulated 4-Bit and 4-Bit Magnitude Combined

From the tests, the results were tabulated. This shows that the simulated results and predicted results were accurate.

Once the simulation was carried out, the code was downloaded to the FPGA board and tested under similar conditions. Below shows an image displaying the switch position and LED result for test 1. Test 2, and 4 can be shown in the appendix under Figure 30 & Figure 32.

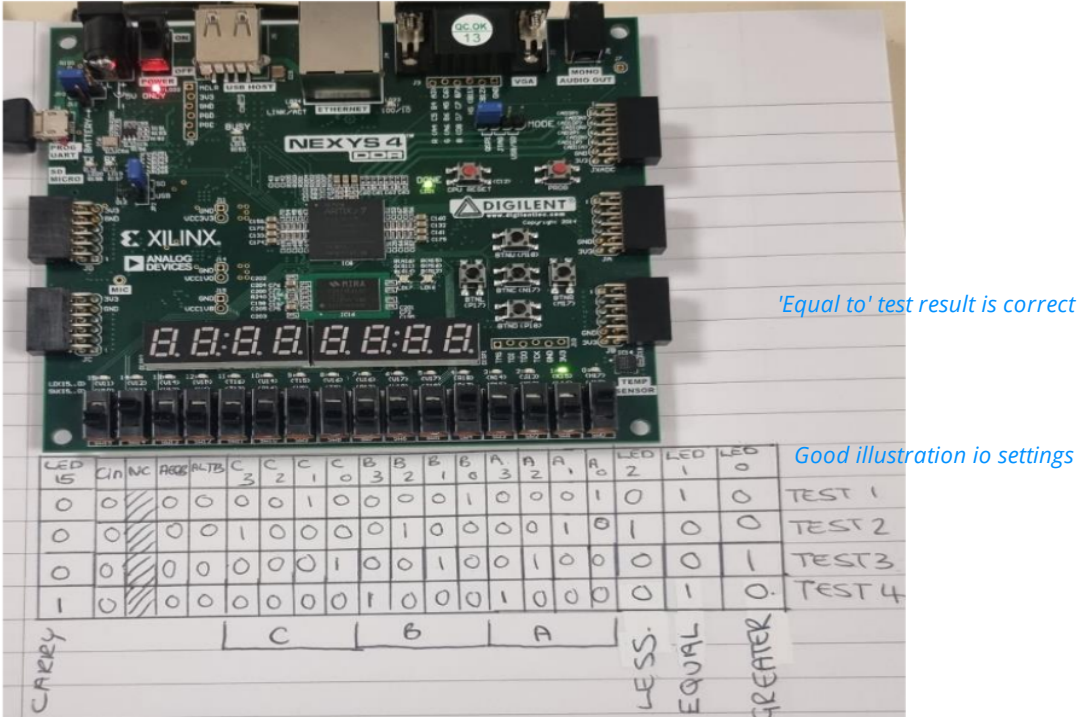


Figure 23; 4-bit Adder with 4-bit Comparator IP Actual Test3 1

From the test, the results were tabulated for comparison. This shows that the circuit operated appropriately.

Test Conditions & Results 4-Bit Adder & 4 Bit Magnitude										
Conditions	a0	a1	a2	a3	b0	b1	b2	b3	Output	Expected Output
Test 1	1	0	0	0	1	0	0	0	AEQB	AEQB
Test 2	0	1	0	0	0	0	1	0	AELB	AELB
Test 3	0	0	1	0	0	1	0	0	AGTB	AGTB
Test 4 (CARRY)	0	0	0	1	0	0	0	1	Carry	Carry

Table 4;4-Bit Adder with 4-Bit Comparator Test

Good table of results that follow the actual results obtained from practical tests

Good indication of the carry bit here, proving the function of the system when 4 bit limit is exceeded

5.4 8-Bit Magnitude Comparator

The 8-Bit Magnitude Comparator was tested on Multisim PLD mode. As shown below:

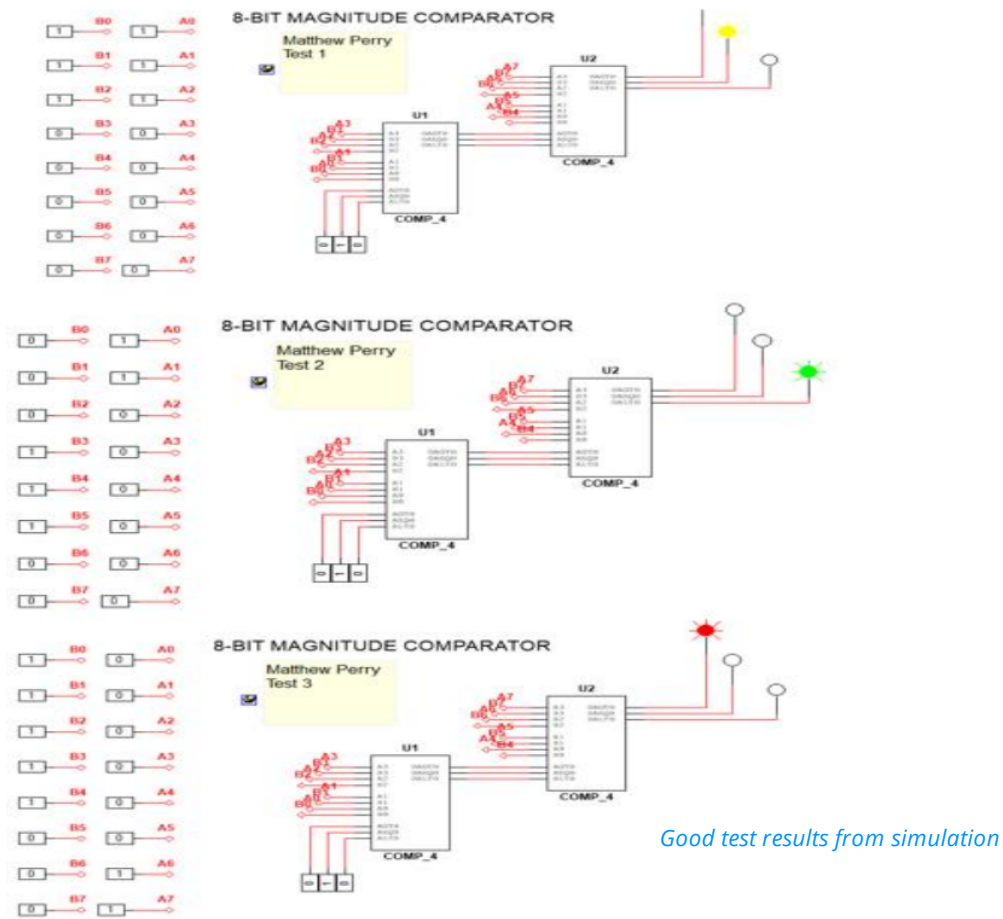


Figure 25; 8-Bit PLD Simulated Test

This shows that the circuit operates appropriately by resulting in three different outputs depending on the inputs.

The code was then downloaded to the FPGA board, and then 3 tests were carried out according to the table below. The test shown in the image is 'Test 1' whereas Test 2 and 3 can be found in the appendix under Figure 30 & Figure 32.

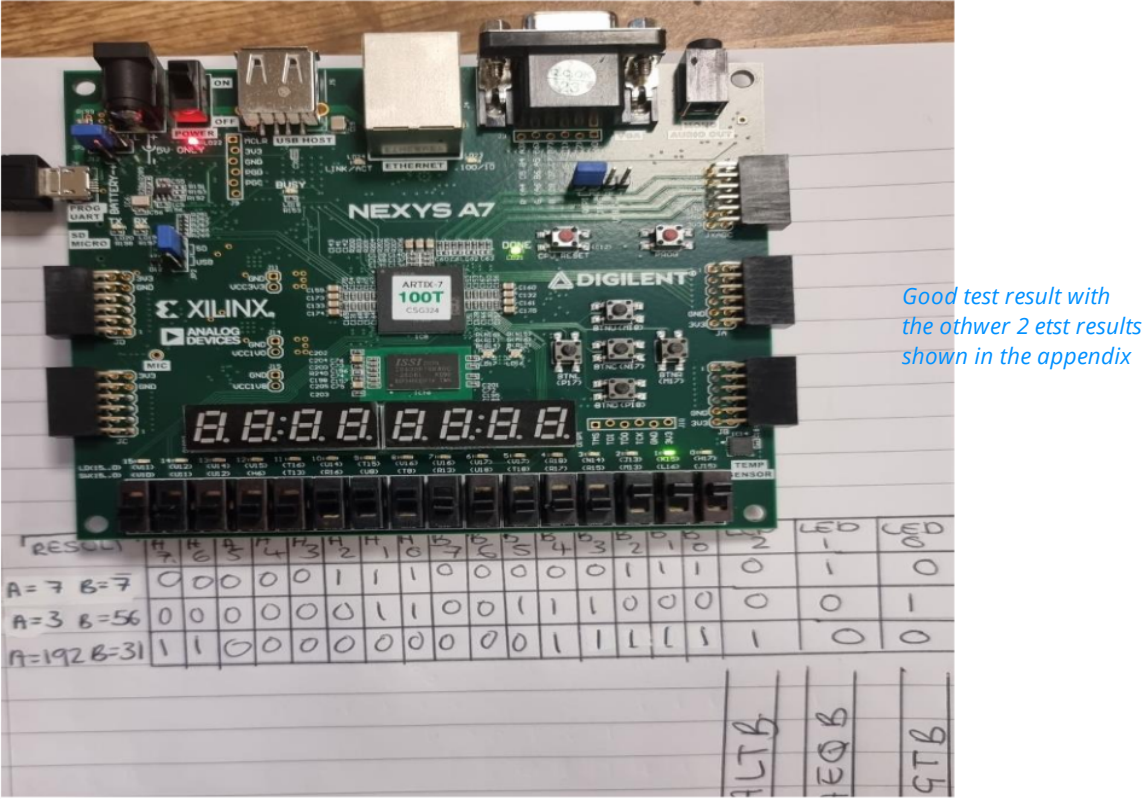


Figure 26; 8-Bit Test Practical

From the tests, it shows that the system works appropriately. A table was compiled of the predicted and actual results to ensure the tests were accurate. Using the 8-bit provided a larger range of inputs and essentially higher decimal numbers.

Results table of 8-Bit Comparator																						
Results	a7	a6	a5	a4	a3	a2	a1	a0	b7	b6	b5	b4	b3	b2	b1	b0	Output Result	Predicted Result	Result A	Result B	Predicted A	Predicted B
Test 1	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1	AEQB	AEQB	7	7	7	7
Test 2	0	0	0	0	0	0	1	1	0	0	1	1	1	0	0	0	AGTB	AGTB	3	56	3	56
Test 3	1	1	0	0	0	0	0	0	0	0	0	1	1	1	1	1	ALTB	ALTB	192	31	192	31

Table 5; 8-Bit Comparison Table Results

Good use of table to organise results

6. Evaluation

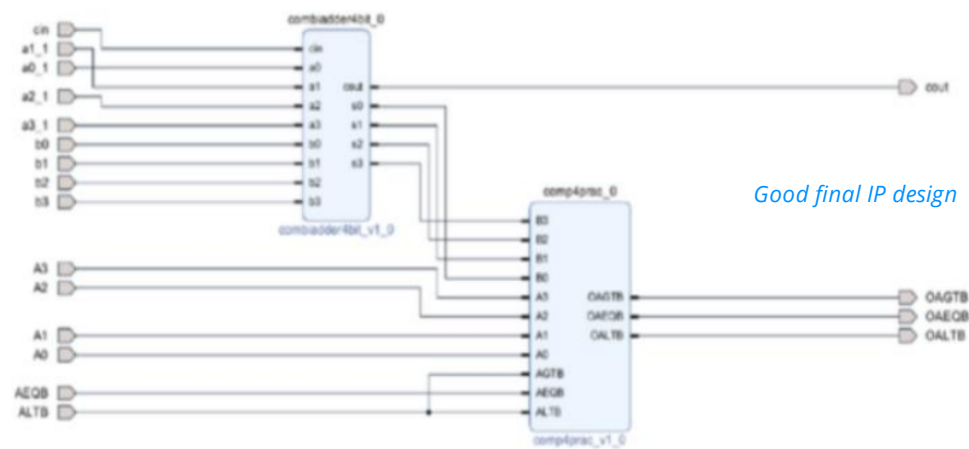
The prototype systems used within the element have been compiled together to build a larger system. It can be seen from the half adder to full and then cascading 4 full bit adders to create a 4-Bit adder.

The 4-bit comparator performed its function appropriately. This was shown through testing the system under the three conditions: equal to, less than and more than. Additionally, the carry function was included, it was shown that due to the system being a 4-Bit it can be shown that $2^4 = 16$. This means that 16 numbers can be represented from 0 to 15. However, the biggest number that can be represented is 15 and the 16 requires a 5th bit as shown $2^5 = 16$.

The 4-Bit system was essentially cascaded in Multisim to create an 8-Bit system while changing some of the input and outputs. For example, for the lower order comparator the A=B cascade input was connected to high while the other two cascading inputs A, B were connected to low. The testing showed that the system was able to accurately determine the three states: equal to, less than and greater to while counting the binary values.

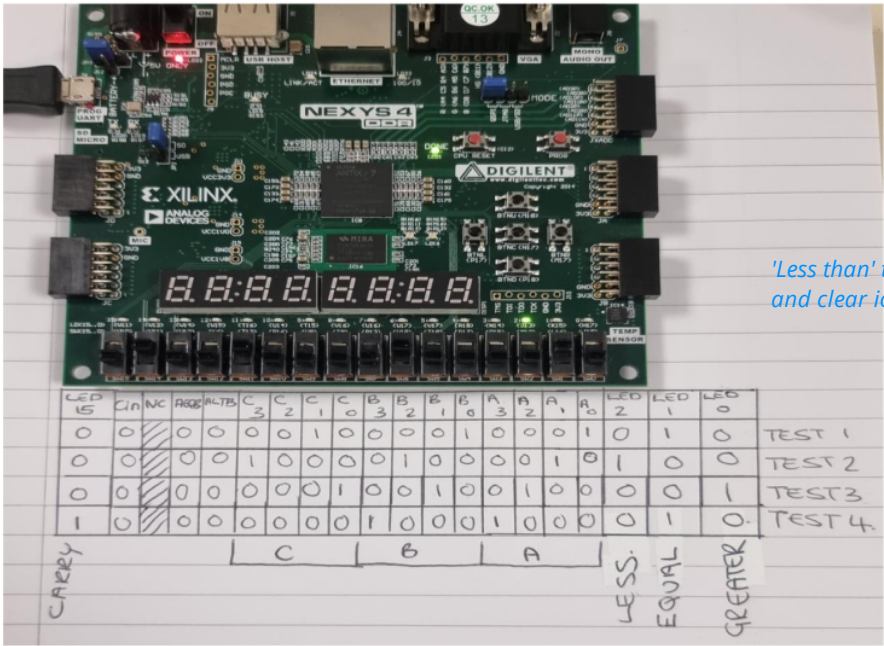
Lastly, for future prototyping if the system included analogue input values an analogue to digital converter would be required. These would include highly efficient comparators and the number of comparators depend on the number of bits the ADC is converting.

7. Appendix



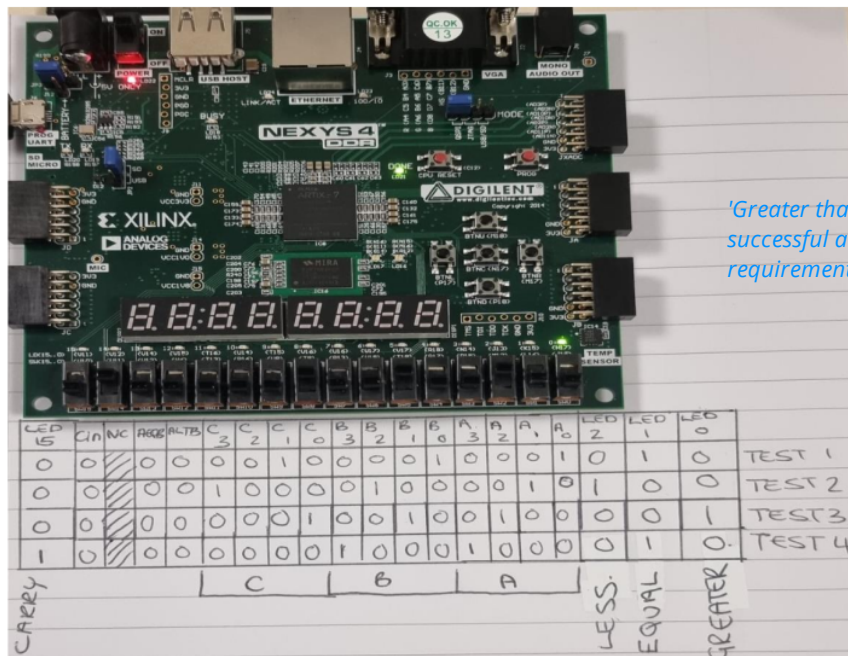
Good final IP design

Figure 28; Example of Intellectual Properties



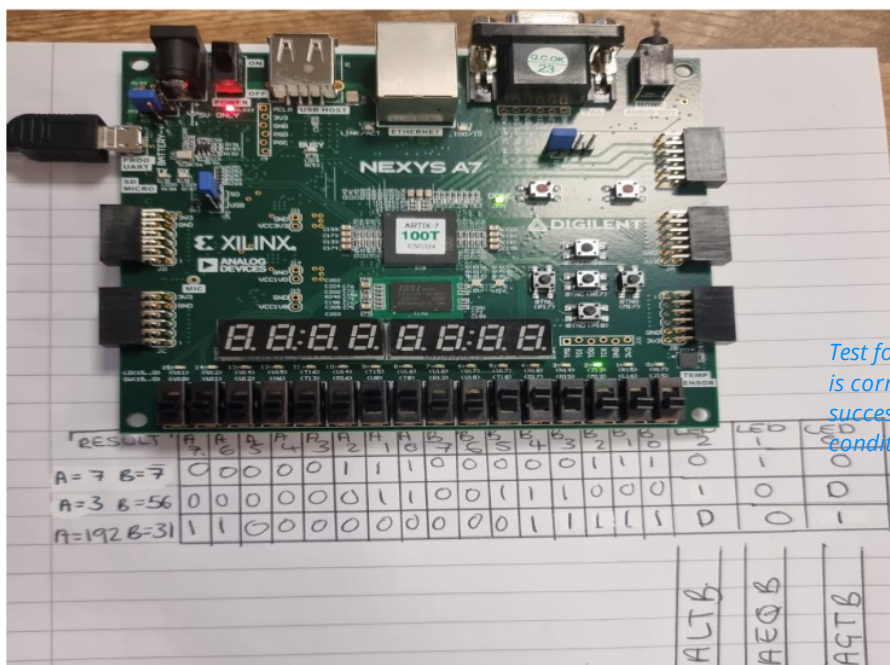
'Less than' test is successful, and clear io configuration

Figure 27; 4-Bit Adder & 4-Bit Comparator Test 2



'Greater than' final test is successful and completes the 3 test requirements

Figure 29 ; 4-Bit Adder & 4-Bit Comparator Test 3



Test for 8 bit comparator is correct. This indicates a successful test for ALT B condition.

Figure 30; 8-Bit Comparator Test 2

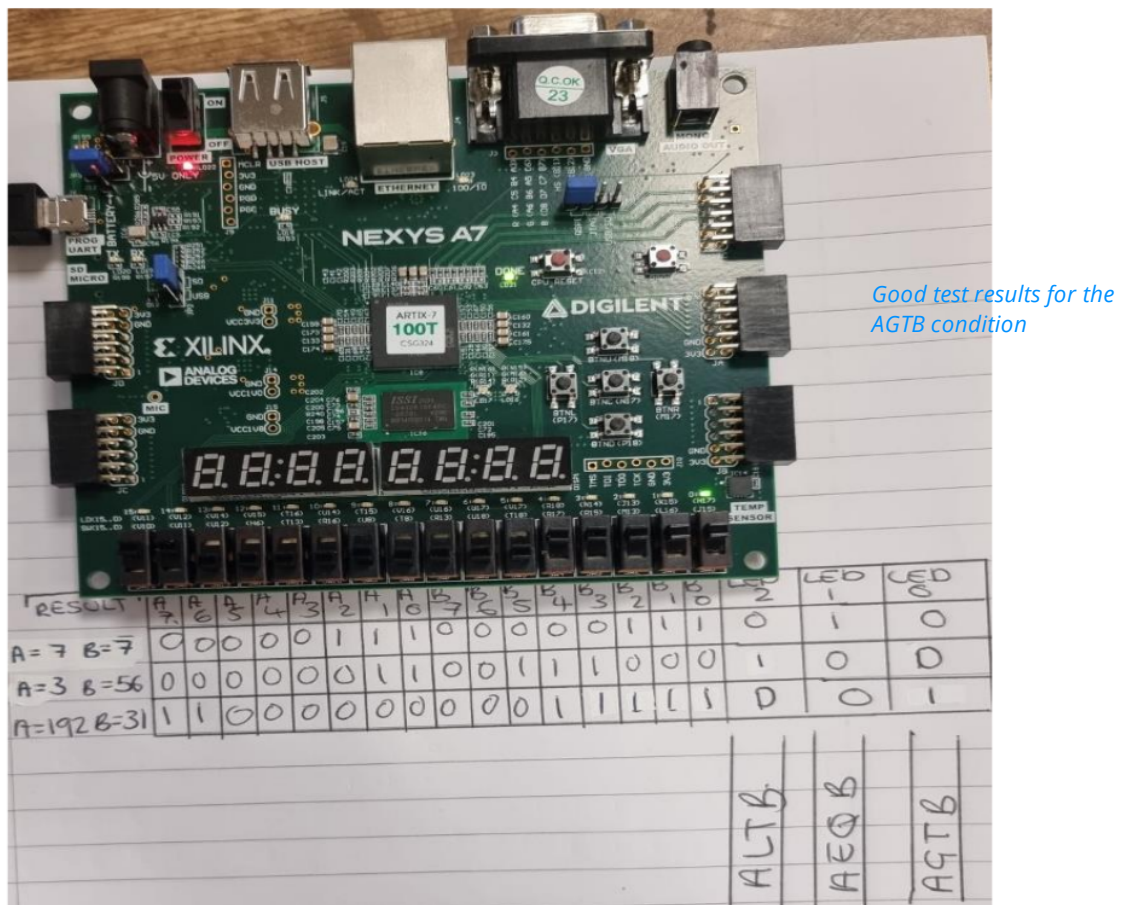


Figure 32; 8-Bit Comparator Test 3

4 bit full adder
 fulladder
 halfadder
 halfadder
 fulladder
 halfadder
 halfadder
 fulladder
 halfadder
 halfadder
 fulladder
 halfadder
 halfadder

Hierarchy as identified in the explanation earlier that shows all the sub-components in the system

Figure 31; 4-Bit Adder Hierarchy Example

```

##Switches

set_property -dict { PACKAGE_PIN J15 IOSTANDARD LVCMOS33 } [get_ports { a0 }]; #IO_L24N_T3_R50_15 Sch=sw[0]
set_property -dict { PACKAGE_PIN L16 IOSTANDARD LVCMOS33 } [get_ports { a1 }]; #IO_L3N_T0_D05_EMCLK_14 Sch=sw[1]
set_property -dict { PACKAGE_PIN M13 IOSTANDARD LVCMOS33 } [get_ports { a2 }]; #IO_L6N_T0_D08_VREF_14 Sch=sw[2]
set_property -dict { PACKAGE_PIN R15 IOSTANDARD LVCMOS33 } [get_ports { a3 }]; #IO_L13N_T2_MRCC_14 Sch=sw[3]
set_property -dict { PACKAGE_PIN R17 IOSTANDARD LVCMOS33 } [get_ports { b0 }]; #IO_L12N_T1_MRCC_14 Sch=sw[4]
set_property -dict { PACKAGE_PIN T18 IOSTANDARD LVCMOS33 } [get_ports { b1 }]; #IO_L7N_T1_D10_14 Sch=sw[5]
set_property -dict { PACKAGE_PIN U18 IOSTANDARD LVCMOS33 } [get_ports { b2 }]; #IO_L17N_T2_A13_D29_14 Sch=sw[6]
set_property -dict { PACKAGE_PIN R13 IOSTANDARD LVCMOS33 } [get_ports { b3 }]; #IO_L5N_T0_D07_14 Sch=sw[7]
set_property -dict { PACKAGE_PIN T8 IOSTANDARD LVCMOS18 } [get_ports { SW8 }]; #IO_L24N_T3_34 Sch=sw[8]
set_property -dict { PACKAGE_PIN U8 IOSTANDARD LVCMOS18 } [get_ports { SW9 }]; #IO_25_34 Sch=sw[9]
set_property -dict { PACKAGE_PIN R16 IOSTANDARD LVCMOS33 } [get_ports { SW10 }]; #IO_L15P_T2_D05_ROWNR_0_14 Sch=sw[10]
set_property -dict { PACKAGE_PIN T13 IOSTANDARD LVCMOS33 } [get_ports { SW11 }]; #IO_L23P_T3_A03_D19_14 Sch=sw[11]
set_property -dict { PACKAGE_PIN H6 IOSTANDARD LVCMOS33 } [get_ports { SW12 }]; #IO_L24P_T3_35 Sch=sw[12]
set_property -dict { PACKAGE_PIN U12 IOSTANDARD LVCMOS33 } [get_ports { SW13 }]; #IO_L20P_T3_A08_D24_14 Sch=sw[13]
set_property -dict { PACKAGE_PIN U11 IOSTANDARD LVCMOS33 } [get_ports { SW14 }]; #IO_L19N_T3_A09_D25_VREF_14 Sch=sw[14]
set_property -dict { PACKAGE_PIN V10 IOSTANDARD LVCMOS33 } [get_ports { cin }]; #IO_L21P_T3_D05_14 Sch=sw[15]

## LEDs

set_property -dict { PACKAGE_PIN H17 IOSTANDARD LVCMOS33 } [get_ports { s0 }]; #IO_L18P_T2_A24_15 Sch=led[0]
set_property -dict { PACKAGE_PIN K15 IOSTANDARD LVCMOS33 } [get_ports { s1 }]; #IO_L24P_T3_R51_15 Sch=led[1]
set_property -dict { PACKAGE_PIN J13 IOSTANDARD LVCMOS33 } [get_ports { s2 }]; #IO_L17N_T2_A25_15 Sch=led[2]
set_property -dict { PACKAGE_PIN N14 IOSTANDARD LVCMOS33 } [get_ports { s3 }]; #IO_L8P_T1_D11_14 Sch=led[3]
set_property -dict { PACKAGE_PIN R18 IOSTANDARD LVCMOS33 } [get_ports { led4 }]; #IO_L7P_T1_D09_14 Sch=led[4]
set_property -dict { PACKAGE_PIN U17 IOSTANDARD LVCMOS33 } [get_ports { led5 }]; #IO_L18N_T2_A11_D27_14 Sch=led[5]
set_property -dict { PACKAGE_PIN V17 IOSTANDARD LVCMOS33 } [get_ports { led6 }]; #IO_L17P_T2_A14_D30_14 Sch=led[6]
set_property -dict { PACKAGE_PIN U16 IOSTANDARD LVCMOS33 } [get_ports { led7 }]; #IO_L18P_T2_A12_D28_14 Sch=led[7]
set_property -dict { PACKAGE_PIN V16 IOSTANDARD LVCMOS33 } [get_ports { LED8 }]; #IO_L16N_T2_A15_D31_14 Sch=led[8]
set_property -dict { PACKAGE_PIN T15 IOSTANDARD LVCMOS33 } [get_ports { LED9 }]; #IO_L14N_T2_SRCC_14 Sch=led[9]
set_property -dict { PACKAGE_PIN U14 IOSTANDARD LVCMOS33 } [get_ports { LED10 }]; #IO_L22P_T3_A05_D21_14 Sch=led[10]
set_property -dict { PACKAGE_PIN T16 IOSTANDARD LVCMOS33 } [get_ports { LED11 }]; #IO_L15N_T2_D05_DOUT_CS0_B_14 Sch=led[11]
set_property -dict { PACKAGE_PIN V15 IOSTANDARD LVCMOS33 } [get_ports { LED12 }]; #IO_L16P_T2_CSI_B_14 Sch=led[12]
set_property -dict { PACKAGE_PIN V14 IOSTANDARD LVCMOS33 } [get_ports { LED13 }]; #IO_L22N_T3_A04_D20_14 Sch=led[13]

```

Good indication of active
io referenced from main
report

Figure 34; Constraints File 4-Bit Adder

```

##Switches

set_property -dict { PACKAGE_PIN J15 IOSTANDARD LVCMOS33 } [get_ports { a0 }]; #IO_L24N_T3_R50_15 Sch=sw[0]
set_property -dict { PACKAGE_PIN L16 IOSTANDARD LVCMOS33 } [get_ports { a1 }]; #IO_L3N_T0_D05_EMCLK_14 Sch=sw[1]
set_property -dict { PACKAGE_PIN M13 IOSTANDARD LVCMOS33 } [get_ports { a2 }]; #IO_L6N_T0_D08_VREF_14 Sch=sw[2]
set_property -dict { PACKAGE_PIN R15 IOSTANDARD LVCMOS33 } [get_ports { a3 }]; #IO_L13N_T2_MRCC_14 Sch=sw[3]
set_property -dict { PACKAGE_PIN R17 IOSTANDARD LVCMOS33 } [get_ports { b0 }]; #IO_L12N_T1_MRCC_14 Sch=sw[4]
set_property -dict { PACKAGE_PIN T18 IOSTANDARD LVCMOS33 } [get_ports { b1 }]; #IO_L7N_T1_D10_14 Sch=sw[5]
set_property -dict { PACKAGE_PIN U18 IOSTANDARD LVCMOS33 } [get_ports { b2 }]; #IO_L17N_T2_A13_D29_14 Sch=sw[6]
set_property -dict { PACKAGE_PIN R13 IOSTANDARD LVCMOS33 } [get_ports { b3 }]; #IO_L5N_T0_D07_14 Sch=sw[7]
set_property -dict { PACKAGE_PIN T8 IOSTANDARD LVCMOS18 } [get_ports { c0 }]; #IO_L24N_T3_34 Sch=sw[8]
set_property -dict { PACKAGE_PIN U8 IOSTANDARD LVCMOS18 } [get_ports { c1 }]; #IO_25_34 Sch=sw[9]
set_property -dict { PACKAGE_PIN R16 IOSTANDARD LVCMOS33 } [get_ports { c2 }]; #IO_L15P_T2_D05_ROWNR_0_14 Sch=sw[10]
set_property -dict { PACKAGE_PIN T13 IOSTANDARD LVCMOS33 } [get_ports { c3 }]; #IO_L23P_T3_A03_D19_14 Sch=sw[11]
set_property -dict { PACKAGE_PIN H6 IOSTANDARD LVCMOS33 } [get_ports { SW12 }]; #IO_L24P_T3_35 Sch=sw[12]
set_property -dict { PACKAGE_PIN U12 IOSTANDARD LVCMOS33 } [get_ports { ALTB }]; #IO_L20P_T3_A08_D24_14 Sch=sw[13]
set_property -dict { PACKAGE_PIN U11 IOSTANDARD LVCMOS33 } [get_ports { AEQB }]; #IO_L19N_T3_A09_D25_VREF_14 Sch=sw[14]
set_property -dict { PACKAGE_PIN V10 IOSTANDARD LVCMOS33 } [get_ports { cin }]; #IO_L21P_T3_D05_14 Sch=sw[15]

## LEDs

set_property -dict { PACKAGE_PIN H17 IOSTANDARD LVCMOS33 } [get_ports { OAGTB }]; #IO_L18P_T2_A24_15 Sch=led[0]
set_property -dict { PACKAGE_PIN K15 IOSTANDARD LVCMOS33 } [get_ports { OAEQB }]; #IO_L24P_T3_R51_15 Sch=led[1]
set_property -dict { PACKAGE_PIN J13 IOSTANDARD LVCMOS33 } [get_ports { OALTB }]; #IO_L17N_T2_A25_15 Sch=led[2]
set_property -dict { PACKAGE_PIN N14 IOSTANDARD LVCMOS33 } [get_ports { led3 }]; #IO_L8P_T1_D11_14 Sch=led[3]
set_property -dict { PACKAGE_PIN R18 IOSTANDARD LVCMOS33 } [get_ports { led4 }]; #IO_L7P_T1_D09_14 Sch=led[4]
set_property -dict { PACKAGE_PIN U17 IOSTANDARD LVCMOS33 } [get_ports { led5 }]; #IO_L18N_T2_A11_D27_14 Sch=led[5]
set_property -dict { PACKAGE_PIN V17 IOSTANDARD LVCMOS33 } [get_ports { led6 }]; #IO_L17P_T2_A14_D30_14 Sch=led[6]
set_property -dict { PACKAGE_PIN U16 IOSTANDARD LVCMOS33 } [get_ports { led7 }]; #IO_L18P_T2_A12_D28_14 Sch=led[7]
set_property -dict { PACKAGE_PIN V16 IOSTANDARD LVCMOS33 } [get_ports { LED8 }]; #IO_L16N_T2_A15_D31_14 Sch=led[8]
set_property -dict { PACKAGE_PIN T15 IOSTANDARD LVCMOS33 } [get_ports { LED9 }]; #IO_L14N_T2_SRCC_14 Sch=led[9]
set_property -dict { PACKAGE_PIN U14 IOSTANDARD LVCMOS33 } [get_ports { LED10 }]; #IO_L22P_T3_A05_D21_14 Sch=led[10]
set_property -dict { PACKAGE_PIN T16 IOSTANDARD LVCMOS33 } [get_ports { LED11 }]; #IO_L15N_T2_D05_DOUT_CS0_B_14 Sch=led[11]
set_property -dict { PACKAGE_PIN V15 IOSTANDARD LVCMOS33 } [get_ports { LED12 }]; #IO_L16P_T2_CSI_B_14 Sch=led[12]
set_property -dict { PACKAGE_PIN V14 IOSTANDARD LVCMOS33 } [get_ports { LED13 }]; #IO_L22N_T3_A04_D20_14 Sch=led[13]
set_property -dict { PACKAGE_PIN V12 IOSTANDARD LVCMOS33 } [get_ports { LED14 }]; #IO_L20N_T3_A07_D23_14 Sch=led[14]
set_property -dict { PACKAGE_PIN V11 IOSTANDARD LVCMOS33 } [get_ports { cout }]; #IO_L21N_T3_D05_A06_D22_14 Sch=led[15]

set_property -dict { PACKAGE_PIN R12 IOSTANDARD LVCMOS33 } [get_ports { LED16_b }]; #IO_L5P_T0_D06_14 Sch=led16_b
set_property -dict { PACKAGE_PIN M16 IOSTANDARD LVCMOS33 } [get_ports { LED16_g }]; #IO_L18P_T1_D14_14 Sch=led16_g
set_property -dict { PACKAGE_PIN N15 IOSTANDARD LVCMOS33 } [get_ports { LED16_r }]; #IO_L11P_T1_SRCC_14 Sch=led16_r
set_property -dict { PACKAGE_PIN G14 IOSTANDARD LVCMOS33 } [get_ports { LED17_b }]; #IO_L15N_T2_D05_ADV_0_15 Sch=led17_b
set_property -dict { PACKAGE_PIN R11 IOSTANDARD LVCMOS33 } [get_ports { LED17_g }]; #IO_0_14 Sch=led17_g
set_property -dict { PACKAGE_PIN N11 IOSTANDARD LVCMOS33 } [get_ports { LED17_r }]; #IO_L11N_T1_SRCC_14 Sch=led17_r

```

Figure 33; Constraints File 4-Bit Adder & 4-Bit Mag

```

1  library IEEE;
2  use IEEE.Std_logic_1164.all;
3  use IEEE.Numeric_std.all;
4
5  entity adder4bit_tb is
6  end;
7
8  architecture bench of adder
9
10 component adder4bit
11 port(
12   cin : in STD_LOGIC;
13   a0 : in STD_LOGIC;
14   a1 : in STD_LOGIC;
15   a2 : in STD_LOGIC;
16   a3 : in STD_LOGIC;
17   b0 : in STD_LOGIC;
18   b1 : in STD_LOGIC;
19   b2 : in STD_LOGIC;
20   b3 : in STD_LOGIC;
21   cout : out STD_LOGIC;
22   a0 : out STD_LOGIC;
23   a1 : out STD_LOGIC;
24   a2 : out STD_LOGIC;
25   a3 : out STD_LOGIC;
26 );
27 end component;
28
29 signal cin: STD_LOGIC;
30 signal a0: STD_LOGIC;
31 signal a1: STD_LOGIC;
32 signal a2: STD_LOGIC;
33 signal a3: STD_LOGIC;
34 signal b0: STD_LOGIC;
35 signal b1: STD_LOGIC;
36 signal b2: STD_LOGIC;
37
38 signal b0: STD_LOGIC;
39 signal cout: STD_LOGIC;
40 signal a0: STD_LOGIC;
41 signal a1: STD_LOGIC;
42 signal a2: STD_LOGIC;
43
44 begin
45   uut: adder4bit port map ( cin => cin,
46                             a0 => a0,
47                             a1 => a1,
48                             a2 => a2,
49                             a3 => a3,
50                             b0 => b0,
51                             b1 => b1,
52                             b2 => b2,
53                             b3 => b3,
54                             cout => cout,
55                             a0 => a0,
56                             a1 => a1,
57                             a2 => a2,
58                             a3 => a3 );
59
60   stimulus: process
61   begin
62
63     -- Put initialisation
64     a0 <= '1';
65     a1 <= '0';
66     a2 <= '1';
67     a3 <= '1';
68     b0 <= '1';
69     b1 <= '0';
70     b2 <= '0';
71     b3 <= '1';
72     cin <= '0';
73     wait for 100 ns;
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     cin <= '0';
83     wait for 100 ns;
84     a0 <= '1';
85     a1 <= '0';
86     a2 <= '0';
87     a3 <= '1';
88     b0 <= '0';
89     b1 <= '0';
90     b2 <= '0';
91     b3 <= '1';
92     cin <= '0';
93     wait for 100 ns;
94     a0 <= '1';
95
96     -- Put test bench stimulus code here
97   end process;
98
99 end;

```

Test data is referenced from report

Figure 36; 4-Bit Adder Test Bench Conditions

```

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

```

Figure 35; 4-Bit Adder Magnitude Test Bench

This is the 4 bit mag comparator test data, it is correctly referenced in the report despite having an incorrect heading identifier here!

Good range of test data used for both test bench applications

8. Works Cited

- CADENCE PCB SOLUTIONS, 2021. *Hardware Description Languages: VHDL vs Verilog, and Their Functional Uses*. [Online]
Available at: <https://resources.pcb.cadence.com/blog/2020-hardware-description-languages-vhdl-vs-verilog-and-their-functional-uses>
[Accessed 6th January 2023].
- Collins, S., 2012. *INSTRUMENTATION AND COMPUTER CONTROL SYSTEMS*. [Online]
Available at: https://www.robots.ox.ac.uk/~gari/teaching/b18/background_lectures/2A2-Signal-Conditioning-L1-Notes-Collins.pdf
[Accessed 6th January 2023].
- Das, D., 2020. *How does Successive Approximation (SAR) ADC Work and Where is it best used?*. [Online]
Available at: <https://circuitdigest.com/article/how-does-successive-approximation-sar-adc-work-and-where-is-it-best-used>
[Accessed 6th January 2023].
- Electronic Tutorials , 2022. *Analogue to Digital Converter*. [Online]
Available at: <https://www.electronics-tutorials.ws/combinational/analogue-to-digital-converter.html>
[Accessed 6th January 2023].
- GFG, 2022. *Magnitude Comparator in Digital Logic*. [Online]
Available at: <https://www.geeksforgeeks.org/magnitude-comparator-in-digital-logic/>
[Accessed 6th January 2023].
- McNally, J., 2022. *What Is LabVIEW? (and why do I care?)*. [Online]
Available at: <https://www.wiresmithtech.com/articles/what-is-labview-and-why-do-i-care/>
[Accessed 6th January 2023].
- Morris, A., 2001. *Global Automation*. [Online]
Available at:
https://people.ifm.liu.se/danfi/TFMT14/TFMT14_files/Measurement%20%26%20Instrumentation%20Principles,3rd%20Edition_Morris.pdf
[Accessed 7th January 2023].
- NI , 2022 . *Learn 10 Functions in NI-DAQmx and Handle 80 Percent of Your Data Acquisition Applications*. [Online]
Available at: <https://www.ni.com/en-gb/support/documentation/supplemental/06/learn-10-functions-in-ni-daqmx-and-handle-80-percent-of-your-dat.html>
[Accessed 12th February 2023].
- NI, 2023. *NI-DAQmx Overview*. [Online]
Available at: <https://www.ni.com/docs/en-US/bundle/ni-daqmx/page/daqhelp/nidagoverview.html#:~:text=What%20is%20DAQmx%3F,CVI%2C%20to%20program%20your%20devices.>
[Accessed 6th February 2023].
- OLCM Systems, 2020. *What is a Load Cell?*. [Online]
Available at: <https://www.lcmsystems.com/resources/load-cells-what-is-a-load-cell>
[Accessed 6th January 2023].

Omega, 2019. *Subminiature Industrial Compression Load Cell*. [Online]

Available at: <https://www.omega.co.uk/pptst/LCKD.html>

[Accessed 7th January 2023].

Oxley, S., 2020. *Signal Conditioning: Methods to Process Analogue Signals to Digital (Infographic)*. [Online]

Available at: <https://blog.ttelectronics.com/signal-conditioning>

[Accessed 9th January 2023].

Peterson Publications, 2011. *Load-cell sensitivity: What it is, how it's measured*. [Online]

Available at:

https://issuu.com/petersonpublications/docs/cq211_web/s/11194600#:~:text=Excitation%20voltage%20is%20the%20voltage,to%2050%20mV%20per%20Volt

[Accessed 9th January 2023].

S.R. Ravikiran, 2017. *IP Cores For FPGA Designs*. [Online]

Available at: <https://www.electronicsforu.com/electronics-projects/electronics-design-guides/ip-cores-fpga-designs>

[Accessed 6th January 2023].

Techopedia , 2020. *Half Adder*. [Online]

Available at: <https://www.techopedia.com/definition/7509/half-adder>

[Accessed 12th January 2023].

Thorat, S., 2019. *Strain Gauge | Working , Types , Diagram , Applications*. [Online]

Available at: https://learnmech.com/strain-gauge-advantages-and-disadvantages-application/#Advantages_of_Strain_Gauge

[Accessed 6th January 2023].

Williams, B., 2010. *n-Depth Look: Load Cell Sensitivity*. [Online]

Available at: <https://www.montalvo.com/article-library/depth-look-load-cell-sensitivity/>

[Accessed 9th January 2023].

XILINX, 2021. *Field Programmable Gate Array*. [Online]

Available at: [https://www.xilinx.com/products/silicon-devices/fpga/what-is-an-fpga.html#:~:text=Field%20Programmable%20Gate%20Arrays%20\(FPGAs,or%20functionality%20requirements%20after%20manufacturing](https://www.xilinx.com/products/silicon-devices/fpga/what-is-an-fpga.html#:~:text=Field%20Programmable%20Gate%20Arrays%20(FPGAs,or%20functionality%20requirements%20after%20manufacturing)

[Accessed 6th January 2023].

A very good range of sources used in the research

Course: HND Electrical and Electronic Engineering

Module: SGEE5007 Virtual Instrumentation and Control

Portfolio: Magnitude Comparator; Element 2

Name: Matthew Perry

University number: 2024421

Date submitted: 28/02/2023

Contents

1. Introduction.....	4
Prototype 1; 8-bit ADC Comparator.....	5
Prototype 2; 10 & 12-Bit ADC Comparator.....	8
10-Bit.....	8
12-Bit.....	11
2. Design requirements.....	13
2.1 Load cells & Signal Conditioning.....	13
2.1.1 Instrument Amplifier.....	13
2.1.2 Instrument Amplifier with Bridge for Load Cell Sensor.....	14
2.1.3 LabVIEW.....	15
3. Implementation.....	17
3.1 FPGA to Elvis II MYDAQmx Connections.....	17
4. Testing and Analysis.....	20
4.1 Instrument Amplifier.....	20
4.2 Digital Magnitude Comparator.....	21
5. Evaluation.....	23
6. Works Cited.....	24
7. Appendix.....	25

Good detail in the table of contents

Figure 1; 8-Bit Design	5
Figure 2; 8-Bit ADC Test 1	6
Figure 3; 8-Bit ADC Test 2	6
Figure 4; 8-Bit ADC Test 3	7
Figure 5; 10-Bit Block Diagram	8
Figure 6; 10-Bit Test 1	9
Figure 7; 10-Bit Test 2	9
Figure 8; 10-Bit Test 3	10
Figure 9; 12-Bit Test 1	11
Figure 10; 12-Bit Test 2	11
Figure 11; 12-Bit Test 3	12
Figure 12; Visual Representation of Instrument Amplifier	13
Figure 13; LabVIEW Block Diagram	15
Figure 14; Multisim Connections	17
Figure 15; Pmod Connections	18
Figure 16; Nexys DDR Layout	18
Figure 17; Complete Layout DDR	19
Figure 18; Graph showing filtered voltage versus grams output	20
Figure 19; Digital Magnitude Comparator Test 1	21
Figure 20; Digital Magnitude Comparator Test 2	21
Figure 21; Digital Magnitude Comparator Test 3	22

Extensive list of figures

Table 1; 8-Bit Test Table	7
Table 2; Converting Binary to Decimal	7
Table 3; 12-Bit Test Table	12
Table 4; Omega Load Cell Data	14
Table 5; Instrument Amplifier Testing	20
Table 6; Digital Magnitude Comparator Testing	22

Good use of tables for results

1. Introduction

This element will consist of designing the LabVIEW Vi that will interface to the field programmable gate array which contains the 8-Bit Magnitude Comparator that was designed in Element 1. This comparator will compare the output between 2 8-Bit digital signals and indicate three states.

The first signal will be produced by an analogue input into LabVIEW with a voltage range of 0 to 5, this will be the signal that represents the load applied to the 1Kg load cell. This signal will be converted to an 8-Bit digital signal and sent to the FPGA.

The second signal will be produced within LabVIEW using the 'set' function, this will range from 0-1000 grams and converted to an 8-Bit digital signal and sent to the FPGA.

Each step of the process will include the design procedure and testing to ensure the system performs as intended. Additionally, before the full system is constructed, two prototypes will be carried out to assist the completion of the final product by expanding familiarity and knowledge about the system.

A good descriptive introduction that demonstrates a good understanding of the requirements in this element

The system was tested with 3 various voltage inputs between 0 to 5V as shown:

Test 1

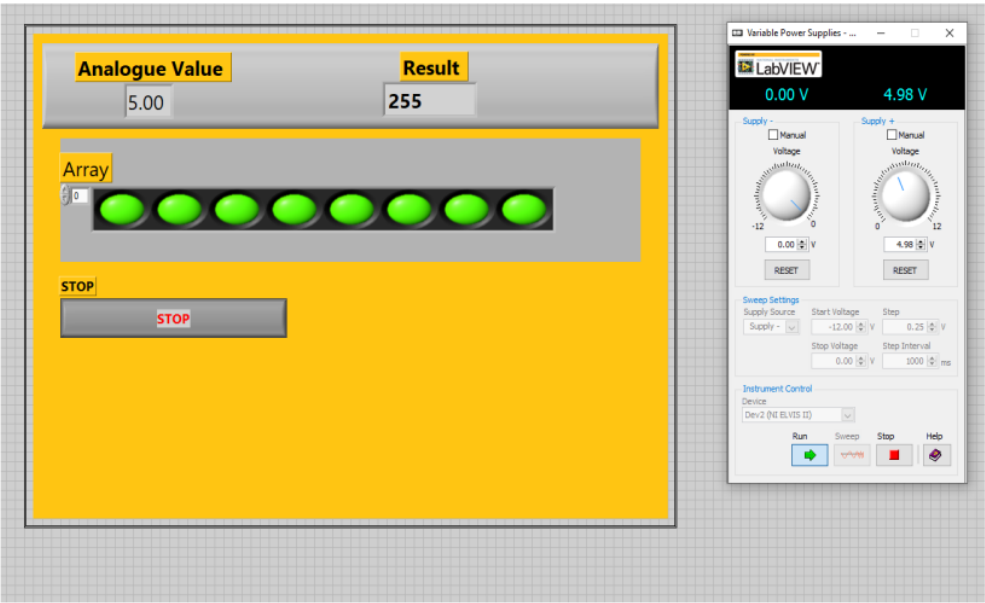


Figure 2; 8-Bit ADC Test 1

Test 2

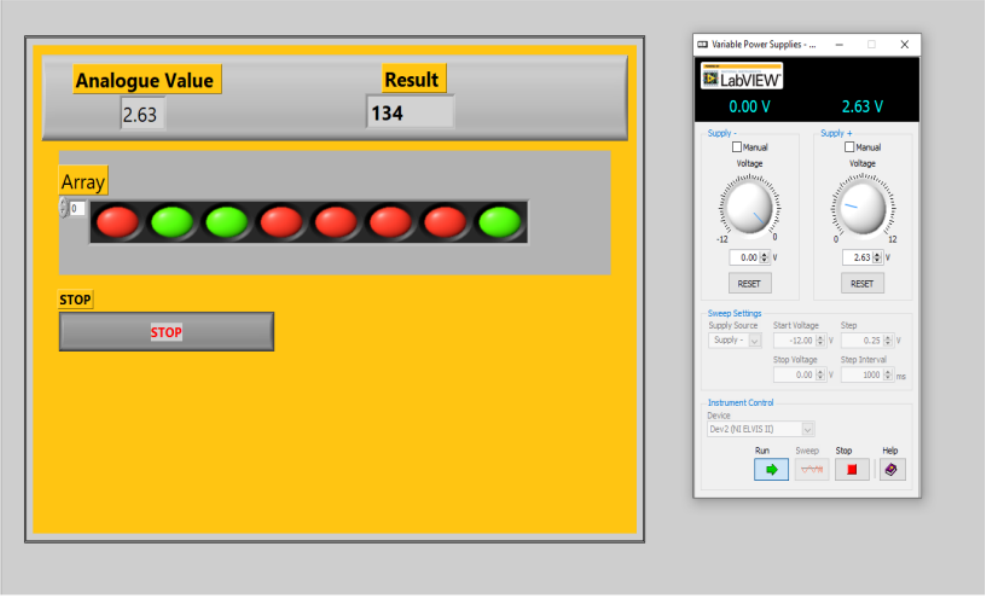


Figure 3; 8-Bit ADC Test 2

Test 3

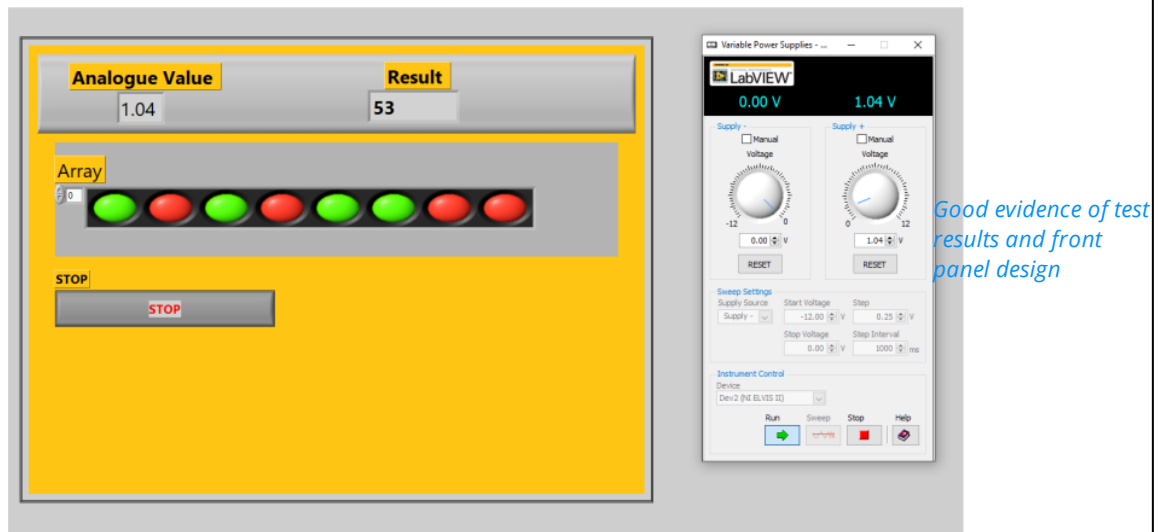


Figure 4; 8-Bit ADC Test 3

From the testing, a table of results was documented as shown:

Results are organised in good order in the table

8-Bit Test Table				
	Analogue Voltage Input	Binary Value	Result	Predicted Result
Test 1	5V	11111111	255	255
Test 2	2.63V	01100001	134	134
Test 3	1.04V	10101100	53	53

Table 1; 8-Bit Test Table

The circuit operated as intended, where the voltage applied was converted to a binary value.

The binary values can also be confirmed by converting the binary numbers to decimal numbers as shown:

How to Calculate Binary to Decimal								
Binary Value	1	1	1	1	1	1	1	1
Decimal	1	2	4	8	16	32	64	128

Table 2; Converting Binary to Decimal

As shown:

Test 1: $1 + 2 + 4 + 8 + 16 + 32 + 64 + 128 = 255$

Test 2: $0 + 2 + 4 + 0 + 0 + 0 + 0 + 128 = 134$

Test 3: $1 + 0 + 4 + 0 + 16 + 32 + 0 + 0 = 53$

Raw values are identified and the significance for further resolution tests

Prototype 2; 10 & 12-Bit ADC Comparator

10-Bit

Firstly, a 10-bit ADC Comparator was designed using LabVIEW Vi, due to it being 10 bits, it has a larger resolution. Shown below is the block diagram for the system.

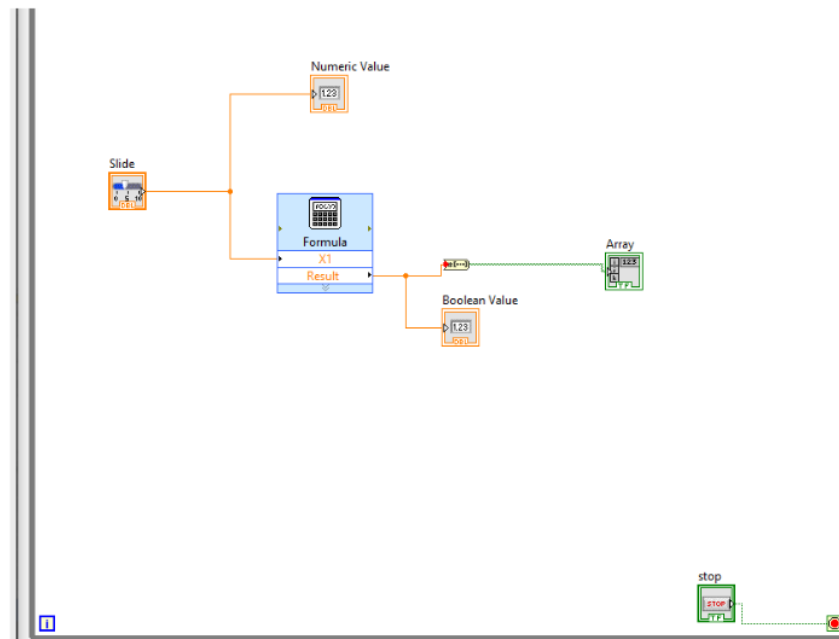


Figure 5; 10-Bit Block Diagram

A slider was used with a range of 0-5 which would simulate the analogue voltage input, this will be read and then processed within a formula block which will use the equation:

$$\left(\frac{X}{5}\right) \times 1023$$

Where:

X – Analogue Input Signal

Good implementation of resolution expansion

5 – Max Voltage

1023 – Resolution 10-Bit

Three tests were conducted on the system that implemented 3 values ranging from 0-5 using the slider.

Test 1

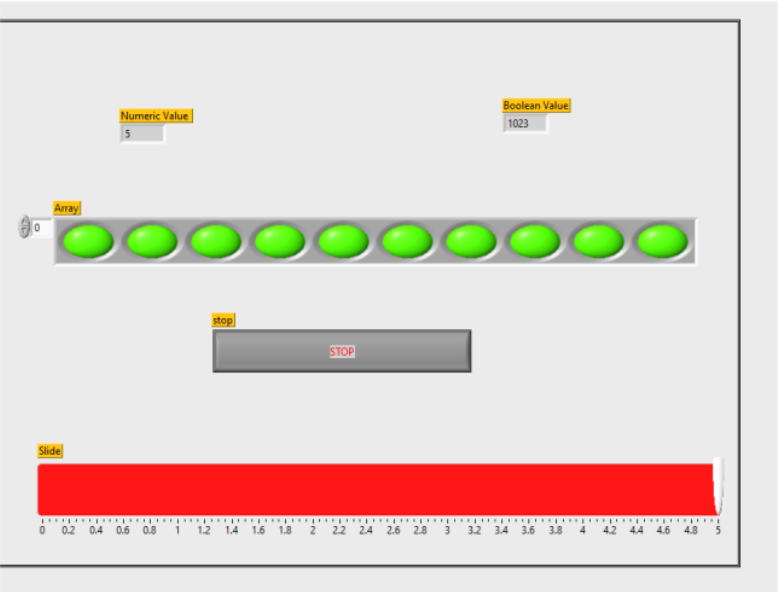


Figure 6; 10-Bit Test 1

Test 2

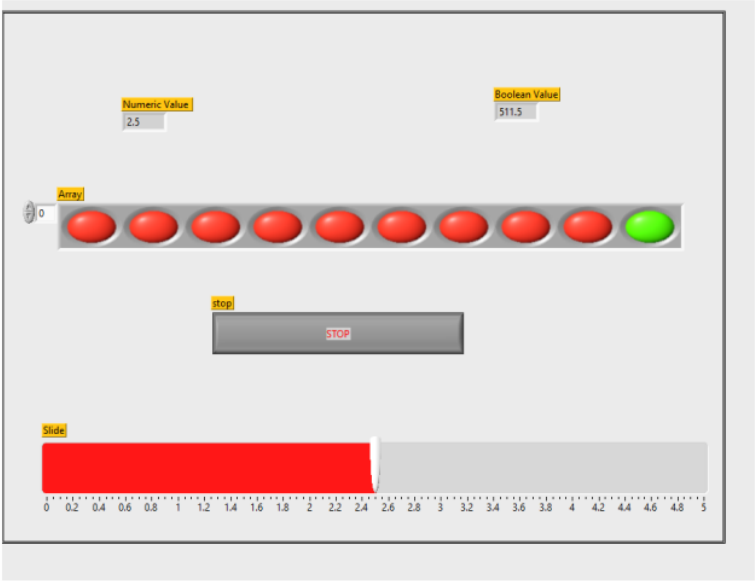
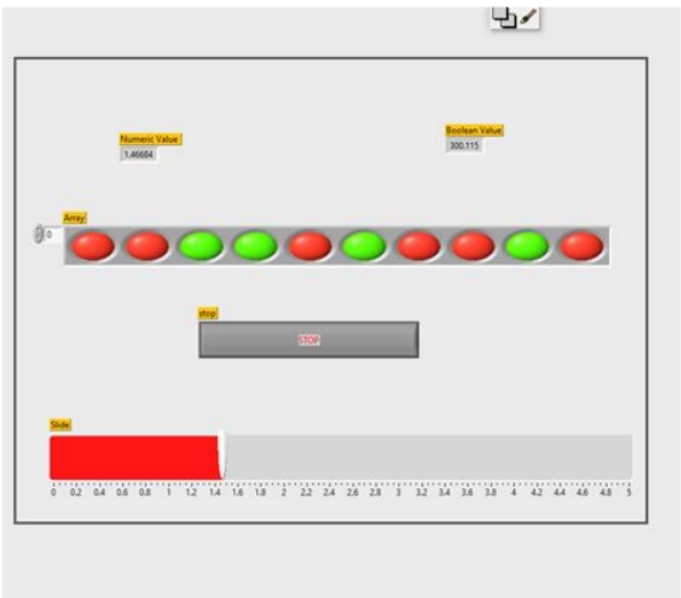


Figure 7; 10-Bit Test 2

Test 3



Good set of test results for 10 bit resolution

Figure 8; 10-Bit Test 3

The results were compiled into a table for analysis shown below.

10-Bit Test Table				
	Analogue Voltage Input	Binary Value	Result	Predicted Result
Test 1	5V	1111111111	1023	1023
Test 2	2.5V	0000000001	511.5	512
Test 3	1.46V	0011010010	300	300

Good use of table to organise results

From the table, it can be shown that the system operated as intended, similarly to the 8-Bit however it has a larger resolution. Test 2 was incorrect by 0.5 due to the scaling error, in the future an amplifier could resolve this issue. Each test can be converted from the binary value to a decimal value to further prove the accuracy of the operation.

As shown:

Test 1: $1 + 2 + 4 + 8 + 16 + 32 + 64 + 128 + 256 + 512 = 1023$

Test 2: $0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 512 = 512$

Test 3: $0 + 0 + 4 + 8 + 0 + 32 + 0 + 0 + 256 + 0 = 300$

Resolution raw data is identified

This shows that the binary value is accurate.

12-Bit

The 10-Bit circuit was modified to process 12-Bits. This created a larger resolution for the system as shown $2^{12} = 4096$ in comparison to the previous 1023. The circuit was the same as the 10-Bit however, the formula block was modified to:

$$\left(\frac{X}{5}\right) \times 4096$$

Where:

X – Analogue Input Signal

5 – Max Voltage

4096 – Resolution 12-Bit

12 bit resolution is implemented and identifies the methods LabVIEW uses to expand resolution

The system was tested under 3 simulated analogue input signals at different scales. This will be a low, medium and high input to see the resolution and processing.

Test 1

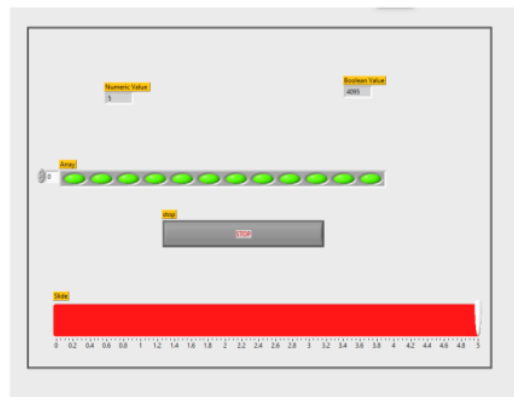


Figure 9; 12-Bit Test 1

Test 2

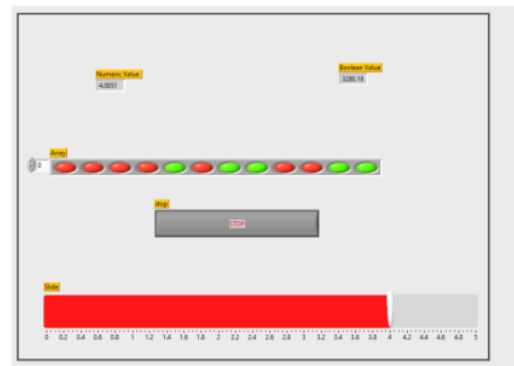


Figure 10; 12-Bit Test 2

Test 3



Good test results and use of front panel instrumentation

Figure 11; 12-Bit Test 3

The results of the tests were documented in a table below for analysis.

Good table of results

12-Bit Testing Table					
	Simulated Analogue Value	Binary Result	Result	Expected Result	Are the LED's Correct?
Test 1	5V	111111111111	4095	4095	Yes
Test 2	4.0051V	000010110011	3280	3280	Yes
Test 3	1.99617V	110001100110	1634.87	1635	Yes

Table 3; 12-Bit Test Table

This shows that the system operated as intended with accurate results, the only discrepancy was test 3 with a 0.13 error between the decimal result and expected result.

The binary value can be proved to be correct by

Test 1: $1 + 2 + 4 + 8 + 16 + 32 + 64 + 128 + 256 + 512 + 1024 + 2048 = 4095$

Test 2: $0 + 0 + 0 + 0 + 16 + 0 + 64 + 128 + 0 + 0 + 1024 + 2048 = 3280$

Test 3: $1 + 2 + 0 + 0 + 0 + 32 + 64 + 0 + 0 + 512 + 1024 + 0 = 1635$

Raw data is identified

2. Design requirements

2.1 Load cells & Signal Conditioning

2.1.1 Instrument Amplifier

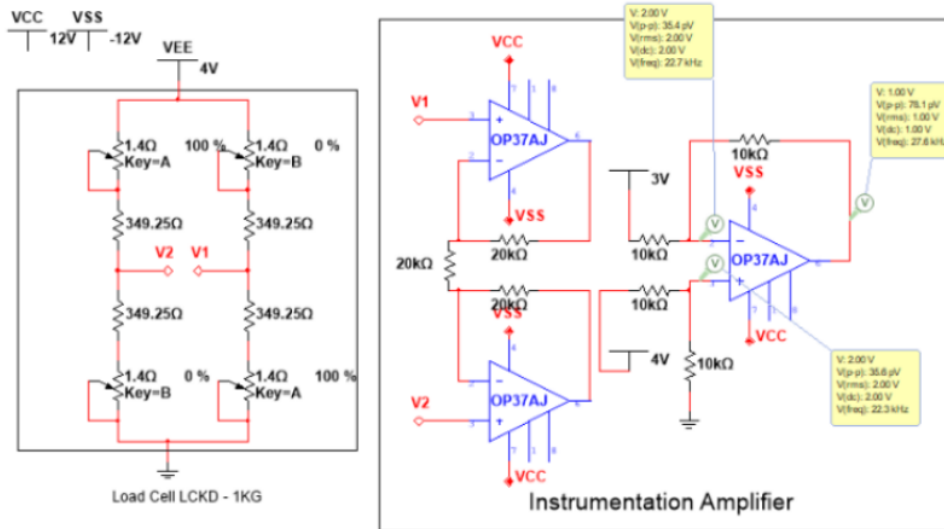


Figure 12; Visual Representation of Instrument Amplifier

Above displays an instrumentation amplifier with unity gain. This circuit is made up of a buffered differential amplifier with three resistors linking the two buffer circuits together. It can be noted that all resistors are of equal value apart from the 20KΩ (R_{gain}). It can be shown that the instrumentation amplifier is disconnected from the amplifier part of the circuit. The circuit has unity gain, meaning the input will be the same level as the produced output. The inputs have an applied voltage of 3 and 4V and the amplifier output with unity is 1V.

This circuit has the advantage of having high input impedances on the V1 & V2 inputs and an adjustable gain that can be set by a single resistor (All About Circuits , 2021).

Good clear explanation

2.1.2 Instrument Amplifier with Bridge for Load Cell Sensor

The instrument amplifier previously describe will be coupled with the Omega LCKD load cell. This load cell will be used to determine how much force is being applied on a given point. This will produce a change in resistance that enables the calculation of how much weight is applied.

Using the data obtained from the Omega Data sheet shown in Data Sheet 1 the following table was created.

Omega 1Kg Load Cell Data	
Excitation	5Vdc, 7Vdc Max
Output	2mV/V Nominal
Max Output	5V x 2mV = 10mV

Table 4; Omega Load Cell Data

The load cell will be zero grams (0V) at 50% on all resistors. The cell will be loaded completely 1000 grams (5v) at 100% and 0% resistor settings in opposite corners.

Instrument amplifier:

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

$$\text{Diff amp gain} = 6000K / 60K = 100$$

Good practical resistor values for gain calculation

For the amplifier to correctly condition the signal a gain of 500 is required. This is proven below.

Gain required:

$$\text{At 1KG and 5V excitation } V_2 - V_1 = 10mV$$

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

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

You have identified the link with the instrument and instrument amplifier. The resistor values chosen have given the correct gain required for signal conditioning

The test result evidence from the amplifier is missing. However, the results from the test are recorded in the table of values, and the linear graph is produced from the results.

2.1.3 LabVIEW

LabVIEW will be used along side the magnitude comparator for the interfacing of the system. Below shows the block diagram that was designed for the system. Each function has been commented on in black that provides an explanation of its purpose within the system.

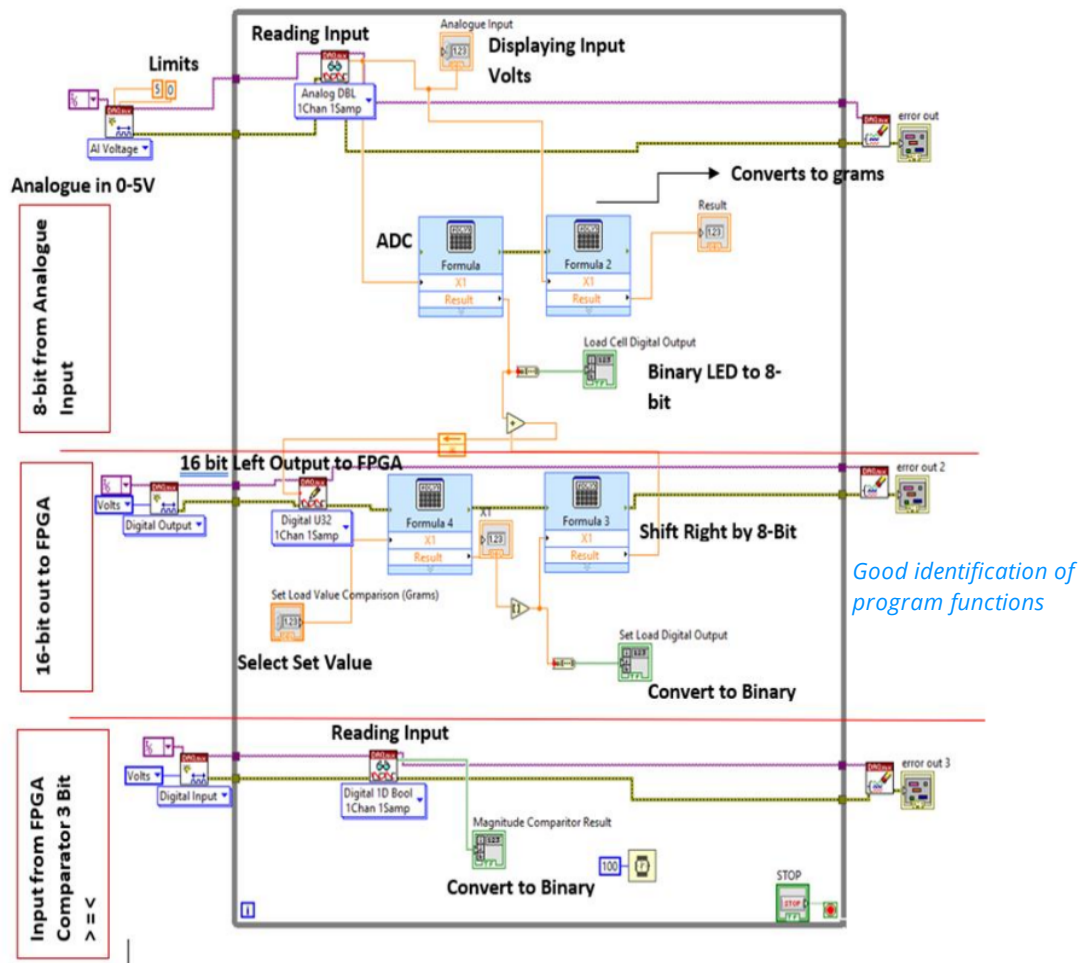


Figure 13; LabVIEW Block Diagram

The block has been divided into 3 sections that handle different parts of the system, the upper layer is the 8-Bit input from the analogue signal, the middle sends the 16-bits to the FPGA and the bottom is the input comparison received from the FPGA.

There are 4 formula blocks also used within the system, there functions are described below:

- Formula – This is used to convert the analogue input into a digital output. This is done by dividing the analogue input by the maximum analogue input and then multiplying the figure by 256 which represents the resolution to convert it to binary. This value is then fed into the digital output that represents the grams value using an 8-Bit word.

$$\left(\frac{\text{Analogue Input}}{5}\right) * 256 = \text{Grams}$$

- Formula 1 - This is used to convert the analogue input into a gram value (ADC). This is carried out by dividing the analogue input by the maximum input and multiplying by 1000.

$$\left(\frac{\text{Analogue Input}}{5}\right) * 1000 = \text{Grams}$$

- Formula 2 – This is used to convert the set gram value into a digital 8-Bit word output. This is done by dividing the set gram value by the mac gram value and then multiplying it by the resolution of the word.

$$\left(\frac{\text{Set Gram Value}}{1000}\right) * 256 = \text{Digital 8 Bit word}$$

- Formula 3 – This is used to shift bits 8 places.

Additionally, the blocks that have been created have also been added to the main panel which displays the input (set) and outputs (read).

Good explanation of program functions with the 8 bit data shift left, which is critical for the function of the 8 bit mag comp

3. Implementation

3.1 FPGA to Elvis || MYDAQmx Connections

Below shows the Multisim program of the circuit which has labelled connections from the FPGA to the Elvis II MYDAQmx connections.

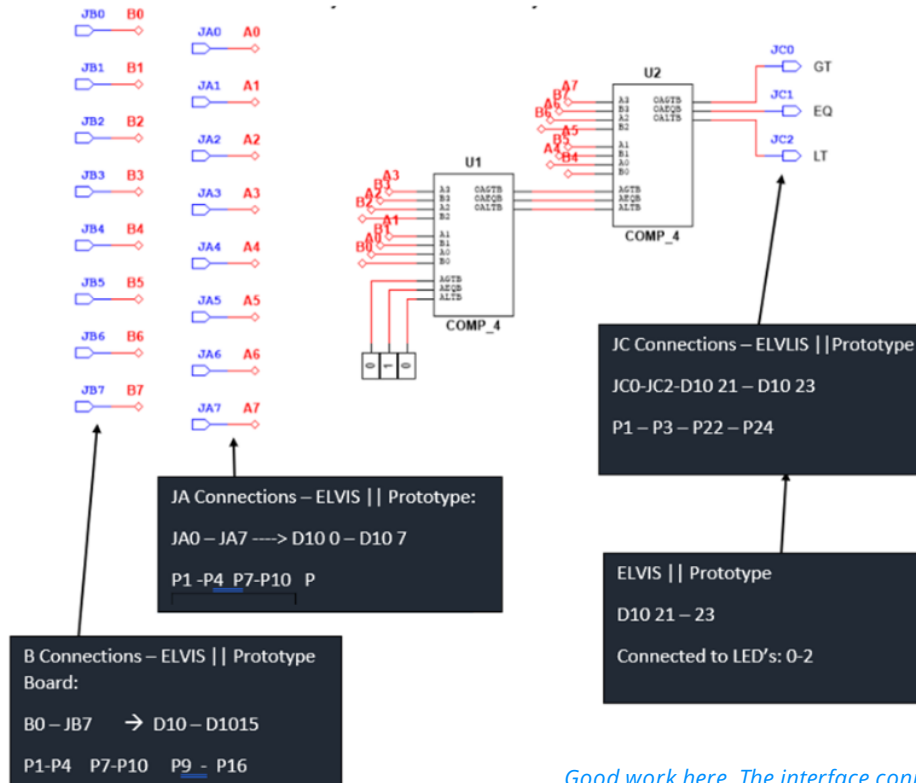


Figure 14; Multisim Connections

Good work here. The interface connections are identified for the DAQ and FPGA board. This shows that you have read and understood the tech data of the FPGA and DAQ development boards

Below shows the PMod port connections with a table displaying in assignments. These connected are arranged in a 2x6 right angle and are 100mm female connectors that fit a standard 2x6 male pin header. Each port can provide two 3.3VCC signals (pin 6 & 12) and two ground signals (pin 5 & 11) and lastly 8 logic signals.



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

Pmod port configuration is identified for use with interface connections

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 15;Pmod Connections

Below shows an image that described the layout of the components on the Nexys DDR board.

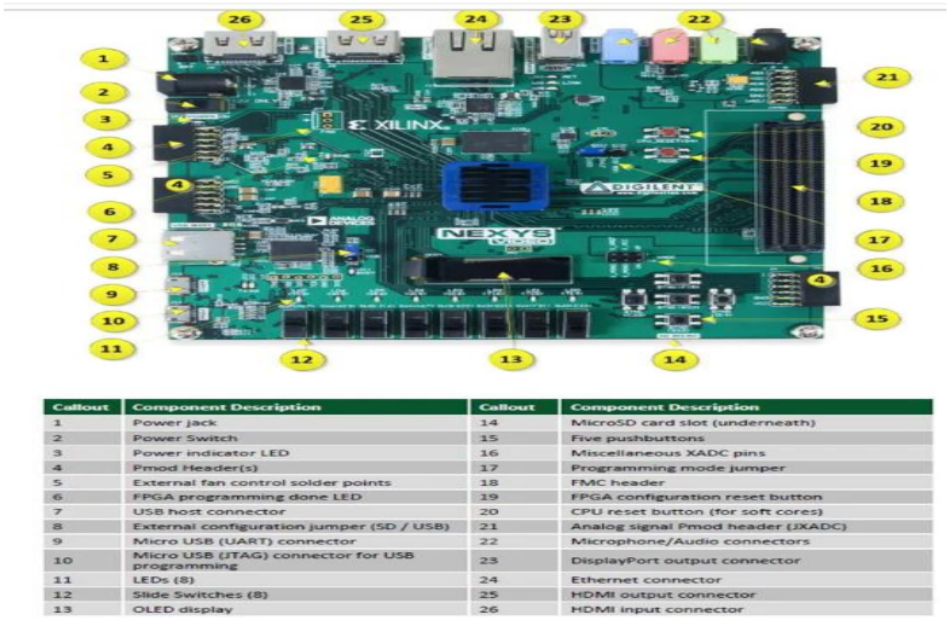
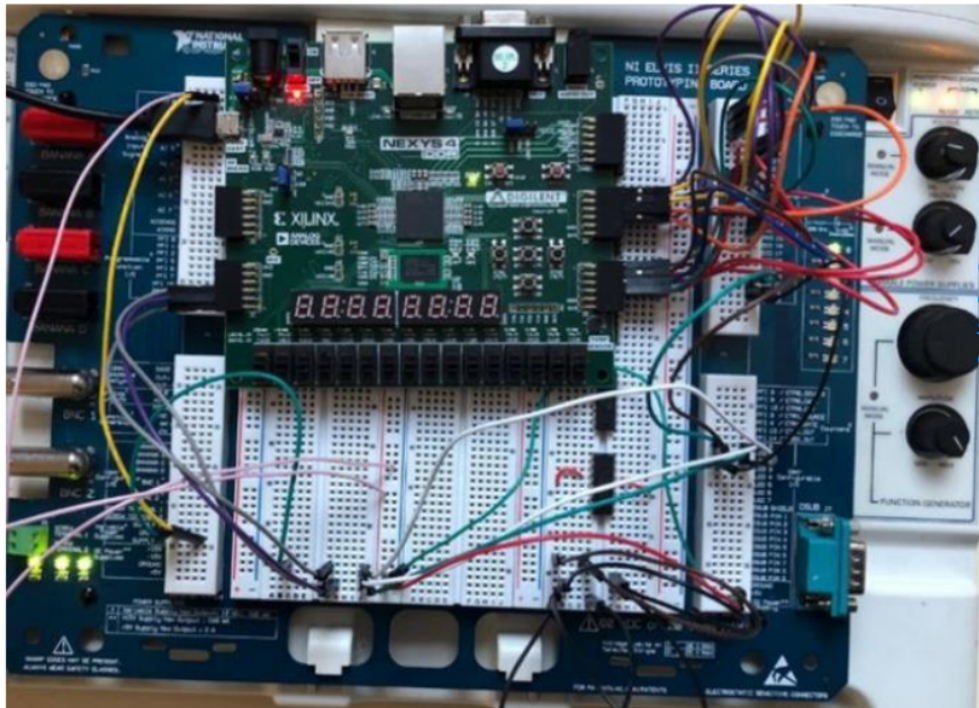


Figure 16; Nexys DDR Layout
(Digilent , n.d.)

Below shows the configuration of the Nexys 4 DDR board connected to the Elvis board. On the Elvis board, the potentiometers are shown that will be used to change the input which will give the appropriate output dependent on the magnitude.



*Good clear
evidence of the
system
configuration*

Figure 17; Complete Layout DDR

4. Testing and Analysis

4.1 Instrument Amplifier

The system will be tested to show that the signal conditioning has worked appropriately and provided an accurate output. The table below shows the testing which was carried out at increased intervals of 0.5V which was done by altering the resistance values.

Instrument Amplifier and Signal Conditioner Testing			
Resistance	Voltage (Included Error)	Filtered Voltage	Grams
50	0.041	0	0
55	0.541	0.5	100
60	1.04	1	200
65	1.54	1.5	300
70	1.92	2	400
75	2.54	2.5	500
80	3.04	3	600
85	3.54	3.5	700
90	4.04	4	800
95	4.54	4.5	900
100	5.04	5	1000

Table 5; Instrument Amplifier Testing

Good results obtained from test procedure. The test evidence of the inst amp is missing

From the results, it can be shown that the circuit operated as intended. It can be shown that there was an approximate 0.04V error before the signal was conditioned, however once it was filtered the voltage was clean providing exact increments. Additionally, from the results a graph was created for visual representation.

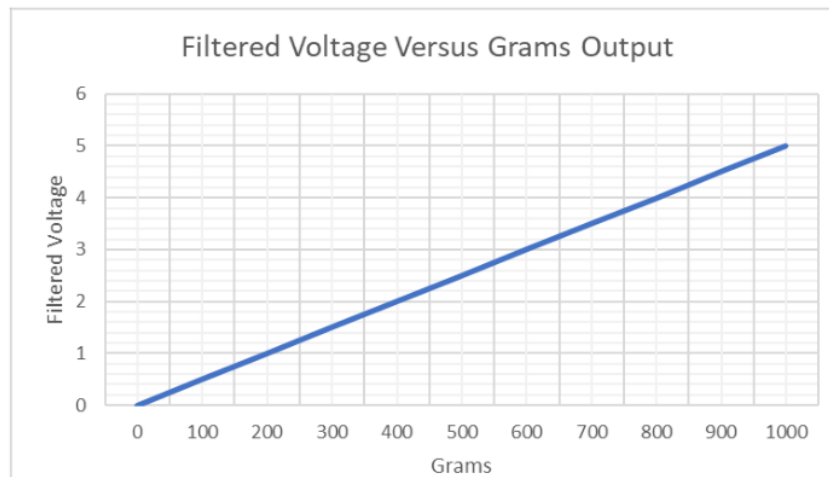


Figure 18; Graph showing filtered voltage versus grams output.

Good linear graph that indicates the correct operation of inst amp

This shows a linear response between voltage (load applied) and grams (weight).

4.2 Digital Magnitude Comparator

The digital magnitude comparator will be tested for the 3 output conditions which are equal to, less than and greater than. This will be done by setting a value of 845 grams on the LabVIEW Vi and comparing it to the analogue input which is provided by manually adjusting the potentiometer.

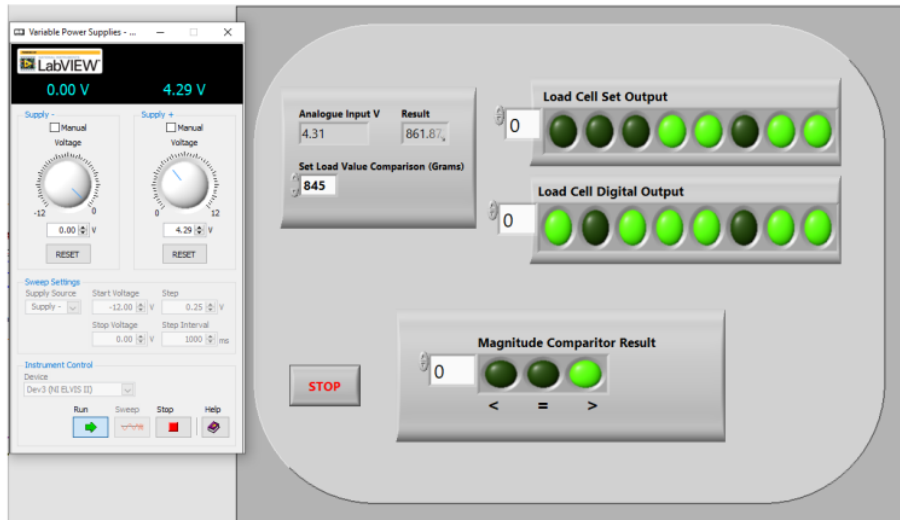


Figure 19; Digital Magnitude Comparator Test 1

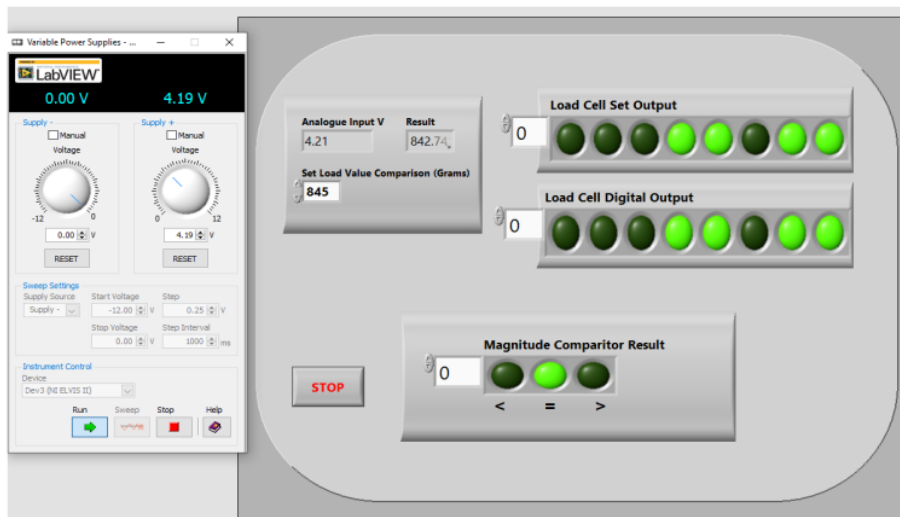


Figure 20; Digital Magnitude Comparator Test 2

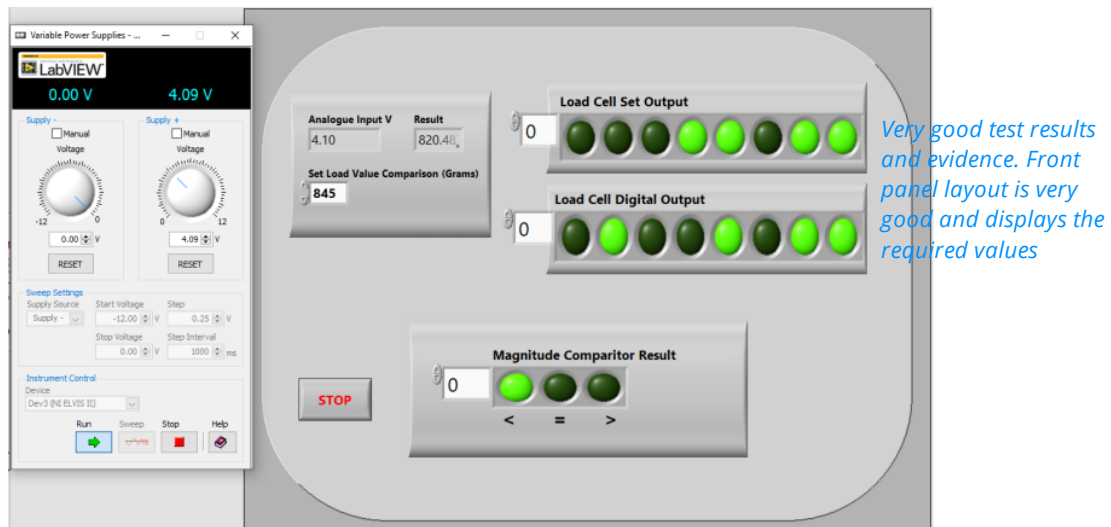


Figure 21; Digital Magnitude Comparator Test 3

Test 1 a voltage of 4.31V was applied to the system to be compared to the set value of 845 grams. This gave the output condition of greater than, because when converted the result was 861. Test 2 a voltage of 4.21V was applied which resulted in the equal to and lastly test 3 a voltage of 4.1V was applied that resulted in a reading of 820 which was then less than.

While conducting the testing, the results were documented as shown below:

Good explanation and results

Digital Magnitude Test Table					
	Set Value	Analogue Input V	Converted Input	Result	Are LED's Operational?
Test 1	845	4.31V	861	Greater Than	Yes
Test 2	845	4.21V	842	Equal To	Yes
Test 3	845	4.1V	820	Less Than	Yes

Table 6; Digital Magnitude Comparator Testing

Good use of table of values

This shows that the system is operational and accurate for its intended purpose.

5. Evaluation

*Good informative reasons
given here*

During this element, three prototypes have been designed, tested and analysed. By completing the prototypes, it gained an understanding of how the LabVIEW Vi and Elvis board works. Additionally, each prototype had a larger bit which increased the resolution of the system and was converted by using the formula block and modifying the equation.

By converting the analogue input to digital data it enabled the system to compare the data via the digital magnitude comparator. Additionally, the signal produced by the analogue input was processed via a signal conditioner. Then the comparator was able to determine 3 states which were less than, greater than and equal to dependant on the values.

From the testing completed in Table 6, it showed that the system was successfully operational.

6. Works Cited

All About Circuits , 2021. *The Instrumentation Amplifier*. [Online]

Available at: <https://www.allaboutcircuits.com/textbook/semiconductors/chpt-8/the-instrumentation-amplifier/>

[Accessed 15th January 2023].

Digilent , n.d. *Nexys 4 DDR Reference Manual*. [Online]

Available at: <https://digilent.com/reference/programmable-logic/nexys-4-ddr/reference-manual>

[Accessed 12th February 2023].

Omega Engineering , 2020. *Subminiature Load Cell*. [Online]

Available at: <https://www.omega.co.uk/pptst/LCKD.html>

[Accessed 2nd February 2023].

Good relevant sources used

7. Appendix

Subminiature Industrial Compression Load Cell

Standard and Metric Models



Compression
0-2.2 lb to 0-1000 lb
0-10 to 0-5000 N

1 Newton = 0.2248 lb
1 daNewton = 10 Newtons
1 lb = 454 g
1 t = 1000 kg = 2204 lb

LCKD/LCMKD Series



Small in size but not in performance, OMEGA's LCKD/LCMKD Series compression load cells are designed to measure load ranges from 1 kg to 1000 lb. Rugged all stainless steel construction and high-performance strain gages ensure superior linearity and stability. Temperature compensation is achieved through a miniature circuit board in the cable. These cells are designed to be mounted on a smooth, flat surface.

SPECIFICATIONS

Excitation: 5 Vdc, 7 Vdc maximum

Output: 2 mV/V nominal

5-Point Calibration:

0%, 50%, 100%, 50%, 0%

Linearity: $\pm 0.25\%$ FSO

Hysteresis: $\pm 0.25\%$ FSO

Repeatability: $\pm 0.10\%$ FSO

Zero Balance: $\pm 2\%$ FSO

Operating Temp Range:

-54 to 121°C (-65 to 250°F)

Compensated Temp Range:

16 to 71°C (60 to 160°F)

Thermal Effects:

Span: $\pm 0.018\%$ FSO/°C

Zero: $\pm 0.009\%$ FSO/°C

Safe Overload: 150% of capacity

Ultimate Overload: 300% of capacity

Bridge Resistance: 350 Ω minimum

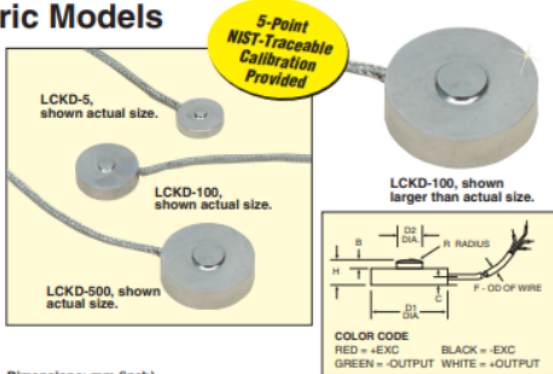
Full Scale Deflection: 0.001 to 0.003"

Electrical Connection: 1.5 m (5')

4-conductor insulated cable with temperature compensation board

Weight: <14 g (<0.5 oz)

Protection Class: IP54



Dimensions: mm (inch)

CAPACITY	D1	D2	H	B	C	F	R
10 to 200 N	9.5	2.2	3.0	0.76	1.0	1.3	6.3
1 kg to 50 lb	(0.38)	(0.09)	(0.12)	(0.03)	(0.04)	(0.05)	(0.25)
500 to 1000 N	13	3.0	3.8	0.51	1.5	1.3	13
100 to 250 lb	(0.50)	(0.12)	(0.15)	(0.02)	(0.06)	(0.05)	(0.5)
2000 to 5000 N	19	6.1	6.4	0.76	2.5	1.3	101
500 to 1000 lb	(0.75)	(0.24)	(0.25)	(0.03)	(0.10)	(0.05)	(4)

STANDARD MODELS

CAPACITY		MODEL NO.	COMPATIBLE METERS
lb	N		
2.2	9.8	LCKD-1KG	DPIS, DP41-S, DP25B-S
5	22.2	LCKD-5	DPIS, DP41-S, DP25B-S
10	46	LCKD-10	DPIS, DP41-S, DP25B-S
25	111	LCKD-25	DPIS, DP41-S, DP25B-S
50	222	LCKD-50	DPIS, DP41-S, DP25B-S
100	445	LCKD-100	DPIS, DP41-S, DP25B-S
250	1112	LCKD-250	DPIS, DP41-S, DP25B-S
500	2224	LCKD-500	DPIS, DP41-S, DP25B-S
1000	4448	LCKD-1000	DPIS, DP41-S, DP25B-S

METRIC MODELS

CAPACITY		MODEL NO.	COMPATIBLE METERS
N	lb		
10	2.25	LCMKD-10N	DP41-S, DP25B-S, iSeries
20	4.5	LCMKD-20N	DP41-S, DP25B-S, iSeries
50	11.2	LCMKD-50N	DP41-S, DP25B-S, iSeries
100	22.5	LCMKD-100N	DP41-S, DP25B-S, iSeries
200	45	LCMKD-200N	DP41-S, DP25B-S, iSeries
500	112	LCMKD-500N	DP41-S, DP25B-S, iSeries
1000	225	LCMKD-1KN	DP41-S, DP25B-S, iSeries
2000	450	LCMKD-2KN	DP41-S, DP25B-S, iSeries
5000	1124	LCMKD-5KN	DP41-S, DP25B-S, iSeries

Comes complete with 5-point NIST-traceable calibration and 59 k Ω shunt data.

Ordering Examples: LCKD-1KG, 1 kg capacity load cell.

LCKD-25, 25 lb capacity load cell.

LCMKD-10N, 10 Newton capacity load cell.

LCMKD-2KN, 2000 Newton capacity load cell.

1

Data Sheet 1; Omega Load Cell

Good relevant source evidence

(Omega Engineering , 2020)

Course: HND Electrical and Electronic Engineering

Module: SGEE5007 Virtual Instrumentation and Control

Portfolio: Magnitude Comparator; Element 3

Name: Matthew Perry

University number: 2024421

Date submitted: 28/02/2023

Contents

1. Introduction 4

1. Design requirements.....5

2. Implementation 7

3. Testing and Analysis.....8

5. Conclusions and Recommendations 9

7. Appendix.....10

There could be more detail with the contents

Figure 1; NI LabVIEW Design 5

Figure 2; Write to Measurement File Configuration..... 6

Figure 3; Directory for saved write files..... 6

Figure 4; Graph of Logged Data 8

Figure 5; Text File Greater Than..... 10

Figure 6; Text File Equal To 10

Figure 7; Text File Less Than 11

Figure 8; Text File Set Load 12

Figure 9; Text File Input Load & Time 12

Figure 10; Panel Image..... 14

Good list of figures

Table 1; Test Table 8

Table 2; Logged Data in Excel..... 13

Good use of tables

1. Introduction

This element will consist of adapting the NI LabVIEW design to output the load cell value displayed in grams over its full range, while also including the comparison outputs in the form of data logging files. The data logging files will be transferred to Microsoft Excel to create a graph to analyse the response and outputs over a period of time.

The introduction demonstrates a clear understanding of requirements

1. Design requirements

The NI LabVIEW Design used in Element 2 was adapted to be able to log the data. The load input in grams will be sent to a write file and also the set load value will be sent to a write file. From the magnitude comparator result an array is used to send a result to the write measurement file. Dependent on the magnitude, the system will use the number value of:

300 – Greater than

This is a good idea for scaling the output log values

200 – Equal to

100 – Less than

This means that the system will be able to log historical data of the inputs and outputs over a period.

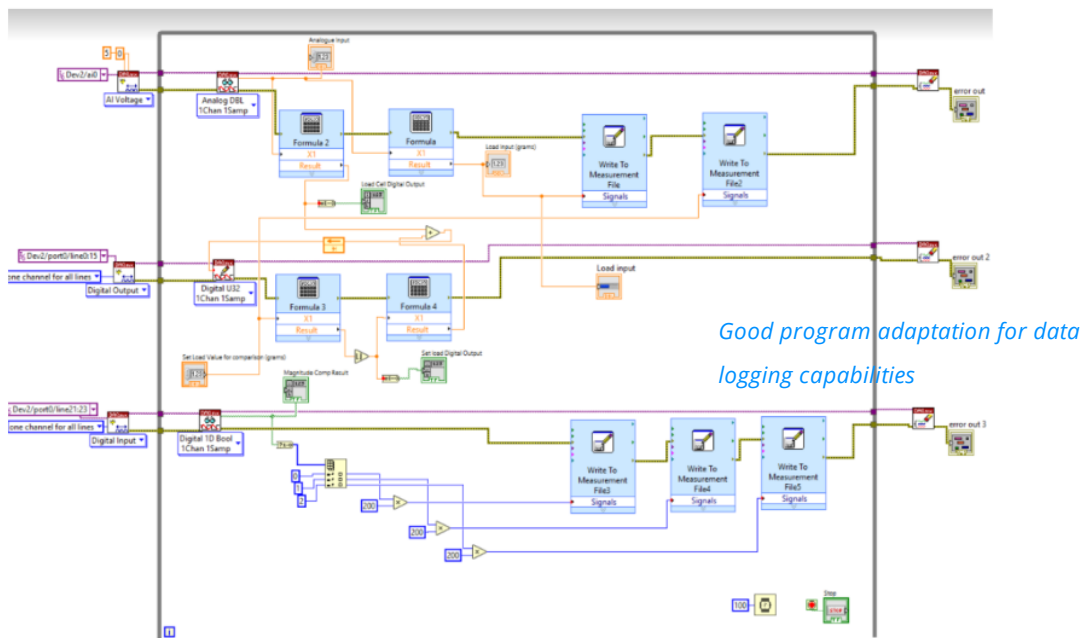
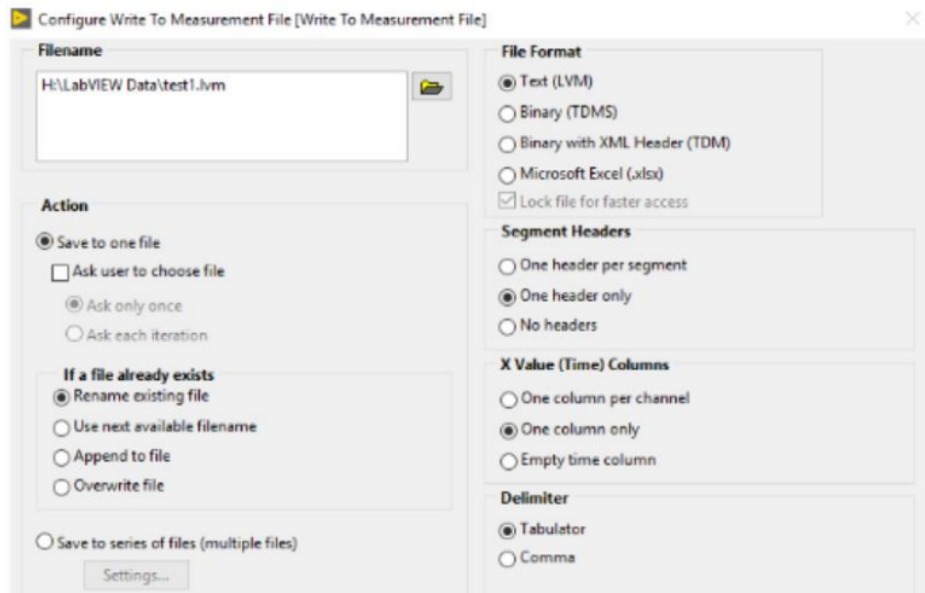
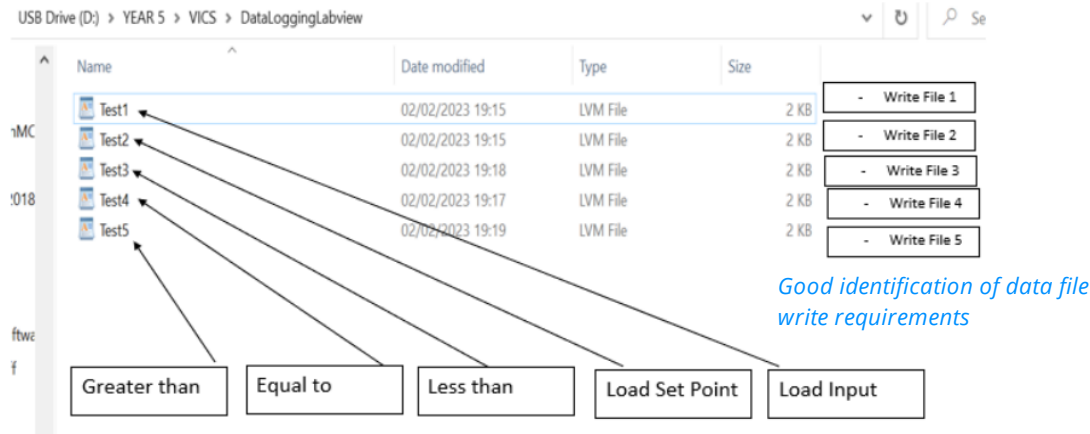


Figure 1; NI LabVIEW Design

Below shows the write to measurement file configuration, this selects where the data will be saved and how the data will be arranged.



The write to measurement files record the data in specific text files, below shows the directory of the files.



This shows what measurement is saved in which file and what measurement write file block is being implemented for each document. Each file can be opened which displays the appropriate data, this can be seen under the appendix under Figure 5, Figure 6, Figure 7, Figure 8 & Figure 9.

2. Implementation

The system was ran using LabVIEW vi, the set point of the load was set to 900 on the front panel. The potentiometer was varied which varies the load cell input value in grams that created an output of 3 conditions.

There are minor grammatical errors in the text content

3. Testing and Analysis

The system was set to a set point of 900, then using the potentiometer the input voltage was varied 3 times. The results were manually documented into a table meanwhile the data was being logged to the text files.

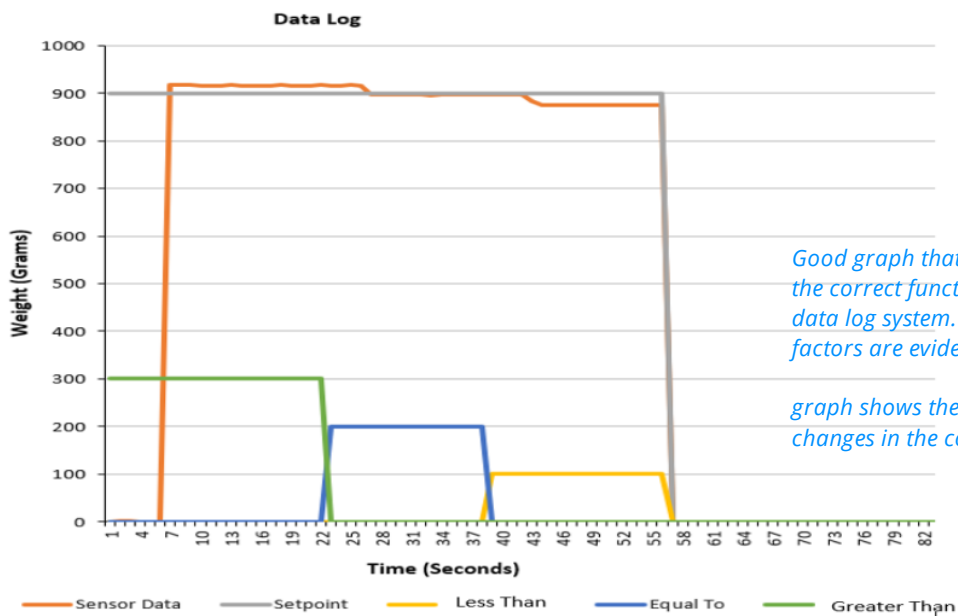
Magnitude Comparator					
	Setpoint	Load Cell Input	Magnitude Result	LED Magnitude Result	Correct Displays on front panel?
Change 1	900	916	300	>	Yes
Change 2	900	900	200	=	Yes
Change 3	900	875	100	<	Yes

Table 1; Test Table

Good table of results

The table displays that the front panel was correctly comparing the magnitudes and displaying the correct read values.

Using the data from the logged files, it was arranged appropriately in Microsoft Excel to produce a graph.



Good graph that demonstrates the correct functioning of the data log system. The scaling factors are evident and the graph shows the timing of the changes in the conditions.

Figure 4; Graph of Logged Data

This shows the input data in orange against the setpoint in grey. At the bottom of the graph the corresponding outputs go high during the period that is appropriate.

5. Conclusions and Recommendations

In conclusion, the system was successful. The system was able to log 5 types of data which were the setpoint, load cell input and the 3 comparison descriptions. Meanwhile the system was accurately converting the input and comparing it to the setpoint and outputting the correct description.

An improvement that could be implemented in the future would be to change the write to measurement configuration. At present, the data is written to a text file and then manually transferred to Microsoft Excel. However, the LabVIEW Vi software allows the block to write directly to Microsoft Excel which would reduce the time taken to record and display the data conveniently.

A good recommendation that can be achieved using available functions in the LabVIEW ATE.

7. Appendix

```
LabVIEW Measurement
Writer_Version 2
Reader_Version 2
Separator Tab
Decimal_Separator .
Multi_Headings No
X_Columns No
Time_Pref Absolute
Operator MP210867
Date 2023/02/02
Time 19:15:10.3017392158508300781
***End_of_Header***

Channels 1
Samples 1
Date 2023/02/02
Time 19:15:10.3017392158508300781
X_Dimension Time
X0 0.0000000000000000E+0
Delta_X 1.000000
***End_of_Header***
X_Value Untitled Comment
300.000000
300.000000
300.000000
```

Figure 5; Text File Greater Than

```
Channels      1  
Samples       1  
Date    2023/02/02  
Time   19:15:10.3017392158508300781  
N_Dimension         Time  
X0     0.0000000000000000E+0  
Delta_X           1.000000  
***End_of_Header***  
X_Value        Untitled          Comment  
  
0.000000  
0.000000  
0.000000  
0.000000  
0.000000  
0.000000  
0.000000  
0.000000  
0.000000  
0.000000  
0.000000  
0.000000  
0.000000  
0.000000  
0.000000  
0.000000  
0.000000  
0.000000  
0.000000  
0.000000  
0.000000  
200.000000  
200.000000  
200.000000  
200.000000
```

Figure 6; Text File Equal To

Figure 7; Text File Less Than

```

LabVIEW Measurement
Writer_Version 2
Reader_Version 2
Separator Tab
Decimal_Separator .
Multi_Headings No
X_Columns No
Time_Pref Absolute
Operator MP210867
Date 2023/02/02
Time 19:15:10.2954974174499511719
***End_of_Header***

Channels 1
Samples 1
Date 2023/02/02
Time 19:15:10.2954974174499511719
X_Dimension Time
X0 0.0000000000000000E+0
Delta_X 1.000000
***End_of_Header***
X_Value Untitled Comment
900.000000
900.000000
900.000000
900.000000
900.000000
900.000000
900.000000
900.000000
900.000000
---

```

Figure 8; Text File Set Load

```

Separator Tab
Decimal_Separator .
Multi_Headings No
X_Columns One
Time_Pref Absolute
Operator MP210867
Date 2023/02/02
Time 19:15:10.32122802734375
***End_of_Header***

Channels 1
Samples 1
Date 2023/02/02
Time 19:15:10.32122802734375
X_Dimension Time
X0 0.0000000000000000E+0
Delta_X 1.000000
***End_of_Header***
X_Value Untitled Comment
0.000000 916.974570
0.098511 917.006692
0.199805 916.813960
0.299905 915.721815
0.399302 916.139400
0.499673 915.625450
0.600171 916.846082
0.698602 916.010912
0.799094 916.621229
0.898836 916.460619
1.000716 916.043034
1.099655 917.167301

```

Data from og files is evident here and the scaling factors can be identified

Figure 9; Text File Input Load & Time

Table 2; Logged Data in Excel

LabVIEW Measurement		900	0	0	300
Writer_Version	2	900	0	0	300
Reader_Version	2	900	0	0	300
Separator	Tab	900	0	0	300
Decimal_Separator	.	900	0	0	300
Multi_Headings	No	900	0	0	300
	0	916.97457	900	0	300
	0.098511	917.006692	900	0	300
	0.199805	916.81396	900	0	300
	0.299905	915.721815	900	0	300
	0.399302	916.1394	900	0	300
	0.499673	915.62545	900	0	300
	0.600171	916.846082	900	0	300
	0.698602	916.010912	900	0	300
	0.799094	916.621229	900	0	300
	0.898836	916.460619	900	0	300
	1.000716	916.043034	900	0	300
	1.099655	917.167301	900	0	300
	1.199853	916.653351	900	0	300
	1.301092	916.556985	900	0	300
	1.401631	916.396375	900	0	300
	1.502362	916.97457	900	0	300
	1.601516	915.689693	900	0	200
	1.701779	916.235766	900	0	200
	1.802065	916.97457	900	0	200
	1.900849	916.524863	900	0	200
	2.00074	897.829937	900	0	200
	2.10066	897.187501	900	0	200
	2.204838	897.444476	900	0	200
	2.301727	896.705674	900	0	200
	2.402902	896.930527	900	0	200
	2.503068	897.476597	900	0	200
	2.602191	896.255968	900	0	200
	2.703782	897.315988	900	0	200
	2.801901	897.476597	900	0	200
	2.90289	897.605085	900	0	200
	3.003366	896.930527	900	0	200
	3.103738	897.412354	900	0	200
	3.203078	896.962648	900	100	0
	3.302722	897.315988	900	100	0
	3.401942	897.315988	900	100	0
	3.504073	897.155379	900	100	0
	3.605906	882.893307	900	100	0
	3.703349	874.638021	900	100	0
	3.802963	875.18409	900	100	0
	3.904984	875.537429	900	100	0
	4.00666	874.79863	900	100	0
	4.104623	876.051377	900	100	0
	4.209475	874.573778	900	100	0
	4.306302	875.569551	900	100	0
	4.405931	875.216212	900	100	0
	4.507679	874.477413	900	100	0
	4.605118	874.638021	900	100	0
	4.705316	875.633794	900	100	0
	4.805022	875.18409	900	100	0
	4.907153	875.023482	900	100	0

Good table of data values

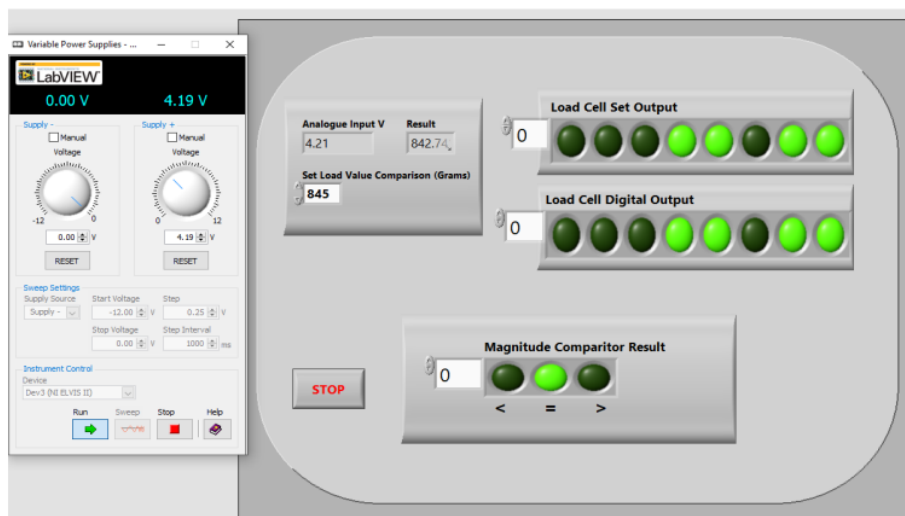


Figure 10; Panel Image

Good design for test

FINAL GRADE

GENERAL COMMENTS

Instructor

70/100

PAGE 1

PAGE 2

Text Comment. Good detailed table of contents

PAGE 3

Text Comment. Good detailed list of figures

Text Comment. Good list of tables

PAGE 4

Text Comment. An appropriate introduction

PAGE 5

Text Comment. Good significant points with suporting ref sources, and image example from your own design.

PAGE 6

Text Comment. Some minor grammatical errors

Text Comment. Good informative explanation and image

Text Comment. The main significant function blocks have been identified here

Text Comment. Main blocks have been identified and explained which indicate a good understanding of requirements from LabVIEW and DAQmx

Text Comment. Good example and ref source

Text Comment. Grammatical error

Text Comment. Good example.

Text Comment. A very good example of a commonly used method of ADC

Text Comment. This title is offset a little here

Text Comment. Good explanation relative to task

Text Comment. Good practical example

Text Comment. Good relevant characteristics identified. This could have included a block diagram of a process involving instrumentation and signal conditioning

Text Comment. A good balanced research carried out on each topic area with good relevant sources used. There are an appropriate number of sources researched for the task.

Text Comment. Design diagram is identified here, and is critical for the VHDL design

Text Comment. VHDL hierarchy is recognised for the system.

Text Comment. The VHDL program explanation is missing

Text Comment. Good design layout of IP design component

Text Comment. Constraints

PAGE 14

Text Comment. You have identified that the VHDL for this cct has been auto generated and used in Vivado

Text Comment. Constant values are identified

Text Comment. Good explanation and IP design

Text Comment. This is true, however, it should be pointed out that the io identifiers in the constraints file must follow the io identifiers in the IP.

PAGE 15

Text Comment. This is a good significant point, as this is consistent with the constants in the Msim cct

Text Comment. Good combinational system design

PAGE 16

Text Comment. Good design of cascaded 4 bit comparators

PAGE 17

Text Comment. Good clear points for requirements

PAGE 18

Text Comment. Good test results using virtual bus method

Text Comment. Cout and sum results should be included. The test results waveforms show the cout carry bit set when the sum exceeds the 4 bit limit

PAGE 19

Text Comment. Explanation is clear and consistent with results

Text Comment. Good test results and displayed using the virtual bus method

Text Comment. Good table of results that follow the actual results, and indicate the 3 conditions required

Text Comment. Good work in identifying the function of the 8 bit comparator that will be used on the FPGA

Text Comment. Good use of table to show results obtained

Text Comment. 'Equal to' test result is correct

Text Comment. Good illustration io settings

Text Comment. Good table of results that follow the actual results obtained from practical tests

Text Comment. Good indication of the carry bit here, proving the function of the system when 4 bit limit is exceeded

Text Comment. Good explanation

Text Comment. Very good test example, proving the carry operation

Text Comment. Good test results from simulation

Text Comment. Good test result with the othwer 2 etst results shown in the appendix

Text Comment. Good use of table to organise results

Text Comment. Good final IP design

Text Comment. 'Less than' test is successful, and clear io configuration

Text Comment. 'Greater than' final test is successful and completes the 3 test requirements

Text Comment. Test for 8 bit comparator is correct. This indicates a successful test for ALTB condition.

PAGE 28

Text Comment. Good test results for the AGTB condition

Text Comment. Hierarchy as identified in the explanation earlier that shows all the sub-components in the system

PAGE 29

Text Comment. Good indication of active io referenced from main report

PAGE 30

Text Comment. Test data is referenced from report

Text Comment. This is the 4 bit mag comparator test data, it is correctly referenced in the report despite having an incorrect heading identifier here!

Text Comment. Good range of test data used for both test bench applications

Text Comment. The 4 bit adder VHDL listing is missing and there is no explanation of the program in this section!

PAGE 31

PAGE 32

Text Comment. A very good range of sources used in the research

PAGE 33

PAGE 34

Text Comment. Good detail in the table of contents

PAGE 35

Text Comment. Extensive list of figures

Text Comment. Good use of tables for results

PAGE 36

Text Comment. A good descriptive introduction that demonstrates a good understanding of the requirements in this element

PAGE 37

Text Comment. Good effort with this prototype using virtual instruments

Text Comment. Good identification of function with recognition of resolution abilities

PAGE 38

PAGE 39

Text Comment. Good evidence of test results and front panel design

Text Comment. Results are organised in good order in the table

Text Comment. Raw values are identified and the significance for further resolution tests

PAGE 40

Text Comment. Good implementation of resolution expansion

PAGE 41

PAGE 42

Text Comment. Good set of test results for 10 bit resolution

Text Comment. Good use of table to organise results

Text Comment. Resolution raw data is identified

PAGE 43

Text Comment. 12 bit resolution is implemented and identifies the methods LabVIEW uses to expand resolution

PAGE 44

Text Comment. Good test results and use of front panel instrumentation

Text Comment. Good table of results

Text Comment. Raw data is identified

PAGE 45

Text Comment. Good clear explanation

PAGE 46

Text Comment. Good practical resistor values for gain calculation

Text Comment. You have identified the link with the instrument and instrument amplifier. The resistor values chosen have given the correct gain required for signal conditioning

Text Comment. The test result evidence from the amplifier is missing. However, the results from the test are recorded in the table of values, and the linear graph is produced from the results.

PAGE 47

Text Comment. Good identification of program functions

PAGE 48

Text Comment. Good explanation of program functions with the 8 bit data shift left, which is critical for the function of the 8 bit mag comp

PAGE 49

Text Comment. Good work here. The interface connections are identified for the DAQ and FPGA board. This shows that you have read and understood the tech data of the FPGA and DAQ development boards

PAGE 50

Text Comment. Pmod port configuration is identified for use with interface connections

PAGE 51

Text Comment. Good clear evidence of the system configuration

Text Comment. Good results obtained from test procedure. The test evidence of the inst amp is missing

Text Comment. Good linear graph that indicates the correct operation of inst amp

Text Comment. Very good test results and evidence. Front panel layout is very good and displays the required values

Text Comment. Good explanation and results

Text Comment. Good use of table of values

Text Comment. Good informative reasons given here

Text Comment. Good relevant sources used

Text Comment. Good relevant source evidence

Text Comment. There could be more detail with the contents

Text Comment. Good list of figures

Text Comment. Good use of tables

Text Comment. The introduction demonstrates a clear understanding of requirements

Text Comment. This is a good idea for scaling the output log values

Text Comment. Good program adaptation for data logging capabilities

Text Comment. Good identification of data file write requirements

Text Comment. There are minor grammatical errors in the text content

Text Comment. Good table of results

Text Comment. Good graph that demonstrates the correct functioning of the data log system. The scaling factors are evident and the graph shows the timing of the changes in the conditions.

Text Comment. A good recommendation that can be achieved using available functions in the LabVIEW ATE.

Text Comment. Data from log files is evident here and the scaling factors can be identified

Text Comment. Good table of data values

Text Comment. Good design for test

MATTHEW PERRY**0****RESEARCH-INFORMED LITERATURE**

Research content includes good relevant details and significant points of interest. Good relevant sources used.

8/10**KNOWLEDGE AND UNDERSTANDING**

Element 1 design requirements have been explained with supporting evidence. The 4 bit adder program is missing and not explained. There are significant points included that demonstrate an understanding. 5/10 Element 2 does not have the evidence of the tested instrument amp included, however, the components and parameters are explained. All other requirements are included and explained in good detail. The LabVIEW program is explained and key parts are annotated. The program design and front panel layout is very good. All ADC's are explained, and good interface methods have been implemented. 6/10 Element 3 content demonstrates a good understanding of the requirements and application. 7/10

18/30**ANALYSIS**



Element 1 test results are all included and well documented. Good use of table of values, however the sum and carry bit should be included and explained in the table for the 4 bit adder test. The results in Vivado are displayed using the virtual bus method. The evaluation includes good relevant detail. 3.5/5 Element 2 test results are explained and include a good range of values. The evidence of the inst amp is missing, however, the parameters and results are all explained. Good quality results with explanations. 7.5/10 Element 3 test results are consistent with the graph. All the required evidence is included and the write file operation is tested with evidence. 8/10

19/25

PRACTICAL APPLICATION AND DEP.



Element 1 design and build is partly explained, however, the VHDL 4 bit adder program section is missing. The implementation is correct and operation was successful. 2.5/5 Element 2 interfacing methods used indicate a good working knowledge of the functions. The only part missing is the inst amp test evidence that would have shown the test methods. 3.5/5 Element 3 is fully documented with all relevant evidence of the task explained. Good initiative has been demonstrated in the task. 8/10

14/20

SKILLS FOR PROFESSIONAL PRAC.



Element 1 is consistent with presentation of requirements, and in a good sequential order. There are minor grammatical errors. 3.5/5 Element

11/15

2 includes good quality methods for presenting content, and in a good sequential order. There are minor grammatical errors. 3.5/5 Element 3 has all the requirements in the content and in good order. 4/5