

Portfolio

by Gareth Wilkins

FILE	13876_GARETH_WILKINS_PORTFOLIO_169043_1848432365.PDF (5.05M)		
TIME SUBMITTED	20-MAY-2021 03:44PM (UTC+0100)	WORD COUNT	8180
SUBMISSION ID	154124986	CHARACTER COUNT	51981

Course: HND Electrical and Electronic Engineering
Module: SGEE5007 Virtual Instrumentation and Control Systems
Portfolio: Magnitude Comparator (Element 1)
Name : Gareth Wilkins
University number: 1809804
Date submitted: 20 May 2021

Contents

List of figures.....	3
1. Introduction	5
2. Research.....	6
2.1 Digital design using VHDL.....	6
2.2 Digital design using IP integrator	7
2.3 NI DAQmx data acquisition used in Labview	8
2.4.1 Digital magnitude comparators	11
2.4.2 Multi bit Magnitude comparator	12
2.5 Analogue to digital converters.....	13
2.6 Load cells (namely LCKD-1kg)	14
2.7 Instrumentation and signal conditioning.....	15
3. Design requirements.....	16
3.1 Half adder.....	16
3.2 Full adder	16
3.3 – 4-bit adder.....	17
4 Implementation	18
4.1 Half adder.....	18
4.2 Full adder design.....	18
4.3 – 4-bit adder design	20
4.4 – 4-bit magnitude comparator design	21
4.5 Combination of 4-bit adder and 4-bit comparator	22
4.6 – 8-bit magnitude comparator	23
5 Testing, analysis and implementation	24
5.1 4-bit adder simulation testing.....	24
5.2 4-bit mag comparator testing	26
5.3 – 4-bit adder comparator combinational logic practical testing	28
5.4 - 8-bit magnitude comparator simulation	31
5.5 – 8-Bit magnitude comparator practical testing.....	33
6. Evaluation	36
6. Appendix	37
References	39

*Good
table of
contents
for
element 1*

List of figures

Figure 1: Magnitude comparator to be designed	5
Figure 2: Graphical representation of VHDL structure.	6
Figure 3: Vivado IP catalogue example.	7
Figure 4: IP block example in Vivado	7
Figure 5: DAQmx icons.....	8
Figure 6: DAQmx block diagram	9
Figure 7: DAQmx Create channel icon	9
Figure 8: DAQmx start icon	9
Figure 9: DAQmx Read channel	10
Figure 10: DAQmx clear task icon	10
Figure 11: DAQmx wait next ms multiple function.....	10
Figure 12: Single bit magnitude comparator	11
Figure 13: 4-bit magnitude comparator	12
Figure 14: 8 bit magnitude comparator	12
Figure 15: Comparison of ADCs resolution and sample rate	13
Figure 16: Omega LCKD load cell range	14
Figure 17: On the left potential divider circuit on right Wheatstone bridge circuit.....	15
Figure 19: Half adder truth table	16
Figure 18: Half adder logic diagram	16
Figure 20: Full adder truth table and logic circuit.....	16
Figure 21: 4 bit adder block diagram (Singhal, 2021)	17
Figure 22: Half adder VHDL code	18
Figure 23: Hierarchy structure of full adder	18
Figure 24: VHDL code displaying inherited half adders	19
Figure 25: Hierarchy structure of 4- bit adder	20
Figure 26: 4 bit adder VHDL code	20
Figure 27: 4-bit magnitude comparator VHDL code.....	21
Figure 28: Combination of 4 bit adder and 4 bit magnitude comparator	22
Figure 29: Combinational logic 4-bit adder and magnitude comparator hierarchical structure.....	22
Figure 30: MultiSim 8-bit magnitude comparator constructed in PLD mode.	23
Figure 31: Simulation result for 4-bit adder.	24
Figure 32: Test bench code for 4-bit adder.....	25
Figure 33: 4-bit magnitude comparator result	26
Figure 34: Test bench code for 4-bit magnitude comparator.....	27
Figure 35: 4-bit adder comparator combinational logic test 1 = A=B.....	28
Figure 36: 4-bit adder comparator combinational logic test 2 = A<B.....	29
Figure 37: 4-bit adder comparator combinational logic result test 3 = A>B	30
Figure 38: 8-bit comparator simulation test 1 A=B.....	31
Figure 39: 8-bit comparator simulation test 2 A<B.....	31
Figure 40: 8-bit comparator simulation test 3 A>B.....	32
Figure 41: 8-bit magnitude comparator practical result Test 1 - A=B	33
Figure 42: 8-Bit magnitude comparator practical result test 2 - A<B	34
Figure 43: 8-Bit magnitude comparator practical result test 3 - A>B	35
Table 1: 1 bit magnitude comparator	11
Table 2: Table of results 4-bit adder	24

Good
list of
figures

Table 3: Table of result 4-bit magnitude comparator.....	26
Table 4: 4-bit adder comparator combinational logic result test 1 = A=B.....	28
Table 5: 4-bit adder comparator combinational logic result test 2 = A<B.....	29
Table 6: 4-bit adder comparator combinational logic result test 3 = A>B.....	30
Table 7: 8-Bit magnitude comparator result – A=B	33
Table 8: 8-Bit comparator practical result test 2 – A<B.....	34
Table 9: 8-Bit magnitude comparator practical result test 3- A>B	35

*Good use of
tables*

1. Introduction

This portfolio contains the design of a magnitude comparator where its operation is shown in figure 1. An analogue signal will be obtained from a load cell providing a 0-5V signal which is inputted into the NI Labview Virtual Instrument which converts this into an 8-bit digital signal which outputs to the FPGA via the NI Elvis II board. The second signal will be attained from a set value in Labview between the ranges of 0-1000gms, transformed into an 8-bit digital signal and outputted to the FPGA via the NI ELVIS II board. This portfolio is divided into three elements to enable a systematic approach to the design, testing and implementation.

In element 1 research is conducted into the software and hardware that will be utilised for this project. The programming packages used in this element will be Vivado and NI Multisim. In Vivado using VHDL a 4-bit adder and a 4-bit magnitude comparator are designed and tested individually to begin. Once the operation of each design is proven successful the two designs are combined to produce a 4-bit adder comparator combinational logic circuit. Following this an 8-bit magnitude comparator is constructed in MultiSim by cascading two 4-bit comparators. Both these circuits are simulated and downloaded to the Nexys FPGA board. The outcomes are evaluated to ensure progress to element 2 is achievable.

Good informative introduction

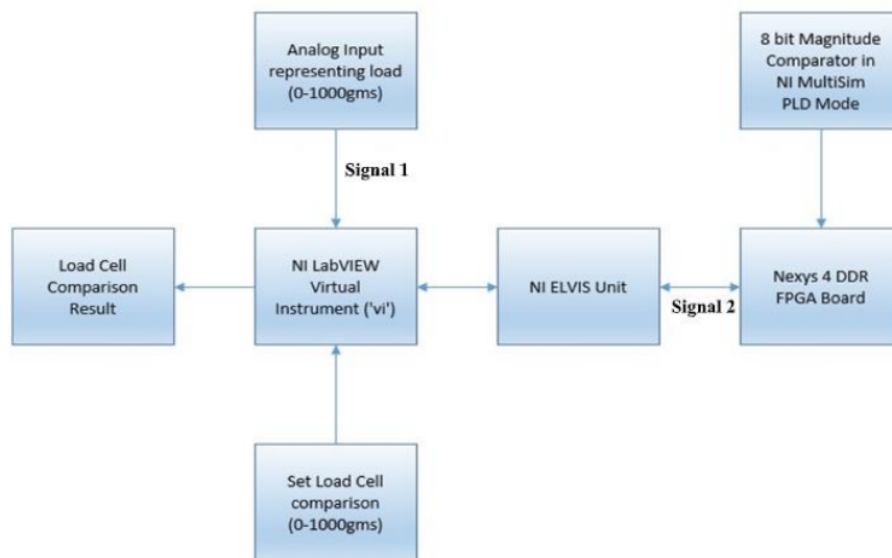


Figure 1: Magnitude comparator to be designed

2. Research

2.1 Digital design using VHDL

VHDL (very high speed integrated circuit hardware description language) is a highly capable hardware descriptive language. Originally VHDL was used as a documentation and modelling language which allowed performance of digital systems designs to be accurately quantified and simulated. (Wakerly, 2006)

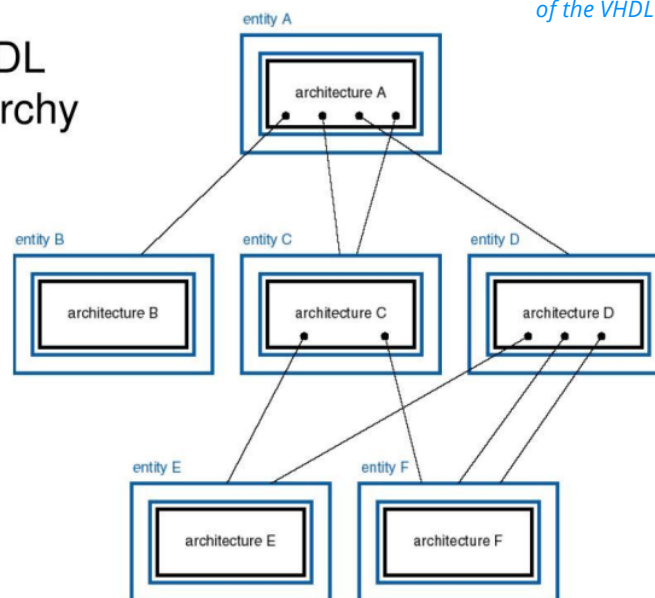
With the development of VHDL synthesis tools popularity in the commercial and enterprise sectors grew stronger. Programs such as Vivado enable logic circuit structures to be written directly from the VHDL behavioural description. Modest combinational circuits to complete microprocessor system on a chip can be designed, simulated, and synthesised using VHDL. (Wakerly, 2006)

VHDL uses the principle of structured programming resembling that of Pascal and Ada programming languages. VHDL code comprises of the entity being the modules input and output declaration and architecture being the description of the module's internal behaviour or structure. In VHDL the entity declaration and architecture definition are separated. The entity declaration is defined as the wrapper for the architecture concealing the information that is inside whilst providing the hooks that the other modules will use. (Wakerly, 2006)

This is the basis of the hierarchy system design where a top level entity can instantiate other entities while concealing the architectural information of the lower level entities from the higher levels. This is shown graphically in figure 2. (Wakerly, 2006)

A good explanation of the structure of the VHDL program.

VHDL Hierarchy



5

Figure 2: Graphical representation of VHDL structure.

2.2 Digital design using IP integrator

Intellectual property (IP) are libraries of hardware descriptive language that are available for users. IPs are available to ease the design process and enable quicker implementation. (Franz, 2021)

The Xilinx Vivado software possess a large IP library in the design suite and IP that can be used with digilent boards. A snapshot of some of the available IP from Vivado's IP catalogue is shown in figure 3.

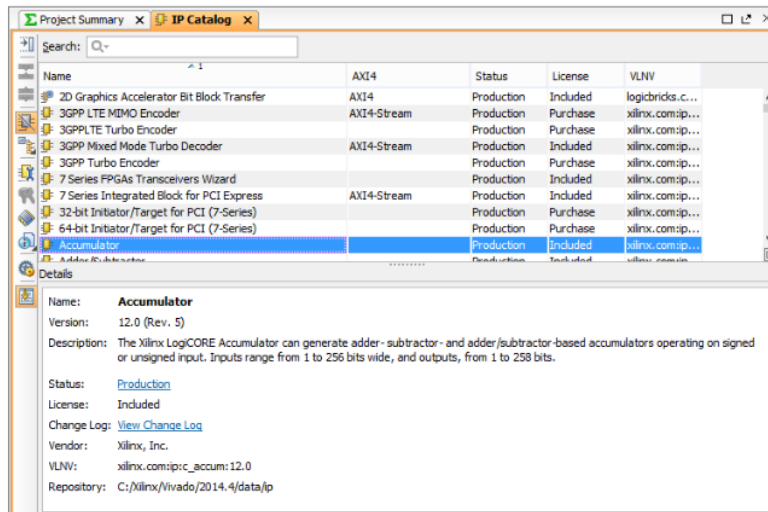


Figure 3: Vivado IP catalogue example.

IPs are not only available in the library but can also be created using Vivado's IP packer tool. The benefit of creating an IP is that you can design schematically by physically joining the pins. This method of implementation is quicker and less tiresome compared to typing each input and output for module instantiation. (Franz, 2021)

This is a good point about the purpose and benefit if IP's in Vivado. Good supporting images.

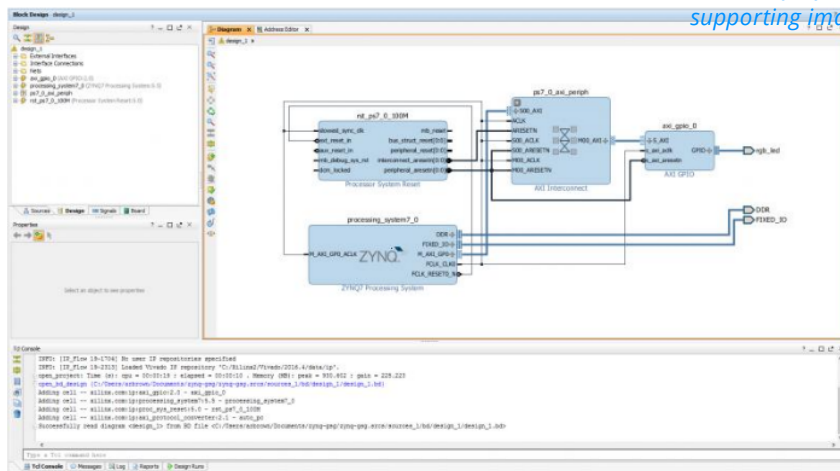


Figure 4: IP block example in Vivado

2.3 NI DAQmx data acquisition used in Labview

The NI DAQmx is the current generation data acquisition driver provided by National Instruments (NI). NI-DAQmx benefits from a complete new driver architecture and application programming interface including new virtual instrumentation functions and improved development tools to control NI DAQ devices. The NI-DAQmx delivers improved performance and ease of use for users over the previous NI-DAQ. Traditional NI-DAQ is still available to support individuals that had development application on NI-DAQ. (National Instruments, 2019)

Multifunction DAQ devices manufactured by NI have the capability to provide an assortment of data acquisition and data generation operations. To program a DAQmx task the following sequential steps are followed: create and configure task, start task, read or write data, stop task, clear task. (Essick, 2019)

Each of the tasks listed above can be carried out by the DAQmx icons shown in figure 5. Several DAQmx icons are polymorphic Vis. Once a polymorphic VI has been placed on the block diagram the associated polymorphic VI selector including possible operational modes. The DAQmx Read.vi for instance can be configured to perform an analogue or digital input by selecting the applicable option on the Vis polymorphic selector. The DAQmx Read.vi can be configured in 50 different modes. By configuring these icons a completed DAQmx task can be coded. (Essick, 2019)

The NI DAQmx function has been identified.

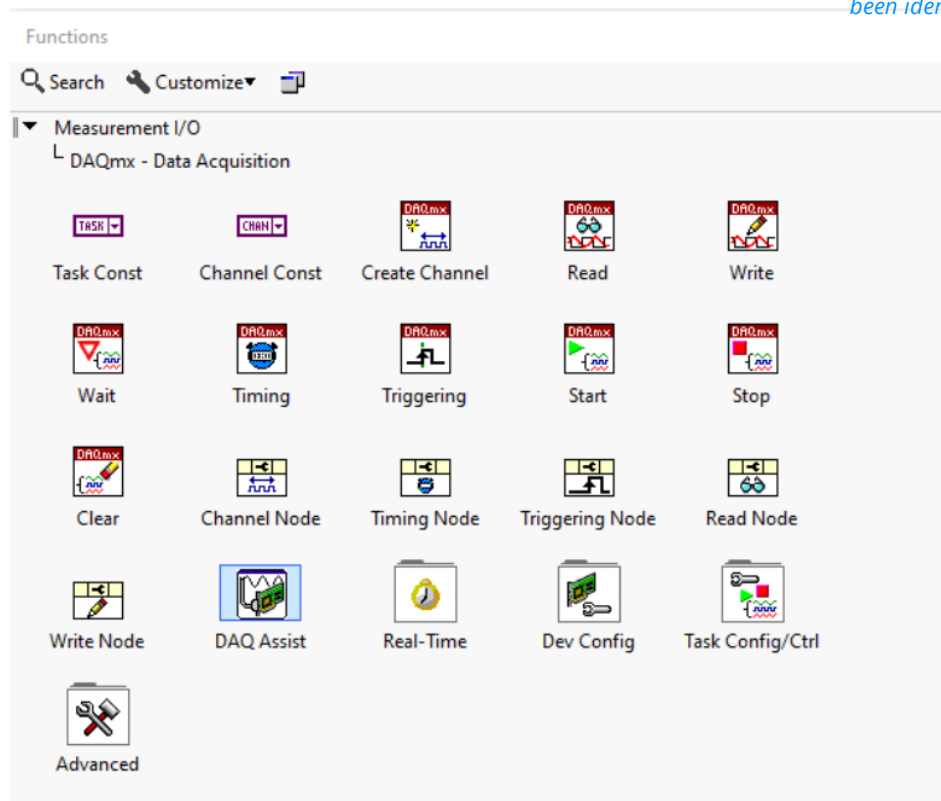


Figure 5: DAQmx icons

A LabView block diagram including some of the DAQmx icons is displayed in figure 6.

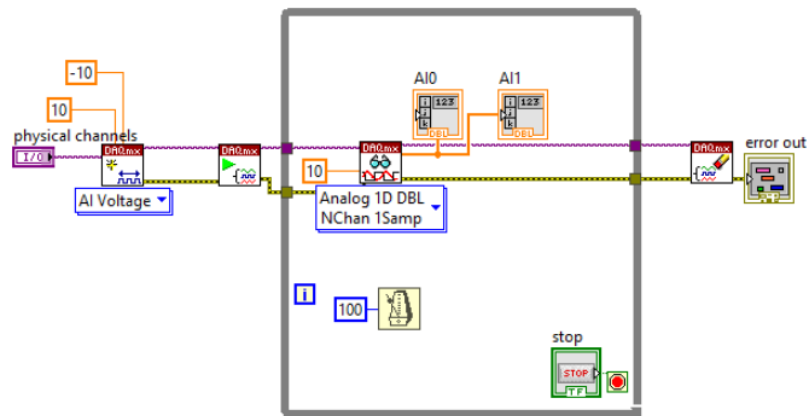
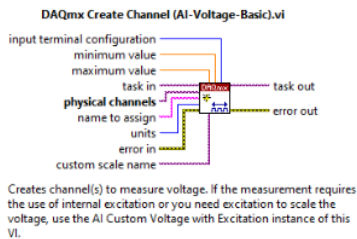


Figure 6: DAQmx block diagram

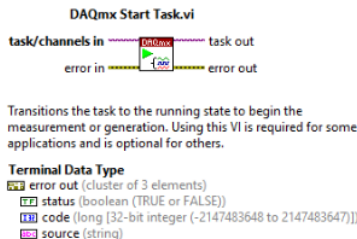
A brief description of the operation of these DAQmx icon operations is defined in the passage. To obtain further detail of the operation of the icon in LabView “detailed help” provides a more in depth resource.



[Detailed help](#)

Figure 7: DAQmx Create channel icon

The “create channel” icon can measure Analogue, Digital and counter IO stipulated by its configuration.



[Detailed help](#)

Figure 8: DAQmx start icon

The “Start task” icon initiates the sequence action needed to perform the required task.

DAQmx Read (Analog 1D DBL NChan 1Samp).vi



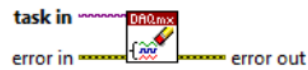
Reads a single floating-point sample from each channel in a task that contains one or more analog input channels.

[Detailed help](#)

Figure 9: DAQmx Read channel

The “read channel” icon reads analogue, digital and counter IO from the input source and determined by its configuratuion.

DAQmx Clear Task.vi



Clears the task. Before clearing, this VI aborts the task, if necessary, and releases any resources the task reserved. You cannot use a task after you clear it unless you recreate the task.

[Detailed help](#)

Figure 10: DAQmx clear task icon

The “clear task” icon clears the current task.

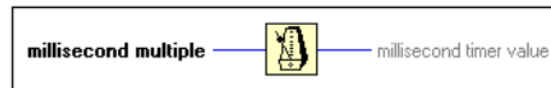


Figure 11: DAQmx wait next ms multiple function

Components of the DAQmx used in the LabVIEW program

have been explained with a good working example.

The “wait until next ms multiple function” synchronises tasks in a sequence within the loop.

2.4.1 Digital magnitude comparators

The digital comparator is a combinational logic circuit which is used extensively in the electronics industry. Comparators are generally constructed with standard AND, NOR and NOT gates which compare the digital inputs of the circuit and produce an output which is determined by the condition of the inputs. (Electronics Post, 2015)

There are two variants of comparator being the identity comparator and the magnitude comparator. The identity comparator compares the input and can only produce a single output. (Electronics Post, 2015)

The Magnitude comparators has three outputs which allows an arithmetic relationship between the input words to be determined: equal to, greater than, less than. (Wakerly, 2006)

A simple single bit magnitude comparator is shown in figure 12.

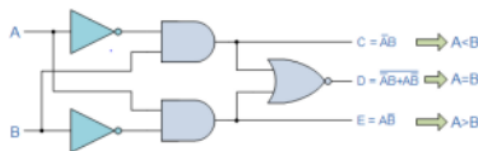


Figure 12: Single bit magnitude comparator

The truth table for this combinational circuit is shown in table 1.

Table 1: 1 bit magnitude comparator

Inputs		Outputs		
B	A	$A > B$	$A = B$	$A < B$
0	0	0	1	0
0	1	1	0	0
1	0	0	0	1
1	1	0	1	0

The comparator cannot distinguish between either the two "0" or two "1" bits as the output $A=B$ is supplied when either $A=B=0$ or $A=B=1$. The output state for $A=B$ resembles that of the EXNOR logic so this is why EXNOR gates are implemented in the design of comparators to compare their respective pair of bits. (Electronics Post, 2015)

2.4.2 Multi bit Magnitude comparator

The 4-bit comparator is a variant of a multi bit magnitude comparator. With this magnitude comparator a 4-bit word on input A and B are compared and an output is obtained.

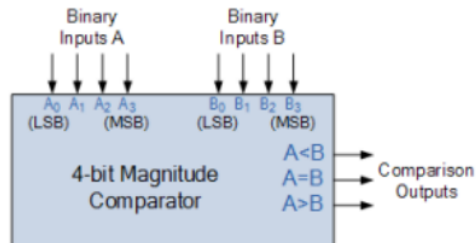


Figure 13: 4-bit magnitude comparator

Cascading comparators enables larger bit comparator to be created. Cascading is achieved by connecting the inputs of a comparator with the output of another comparator. This is shown in figure 14. By cascading two 4-bit comparators an 8-bit comparator is constructed. (Wakerly, 2006)

8-bit magnitude comparator

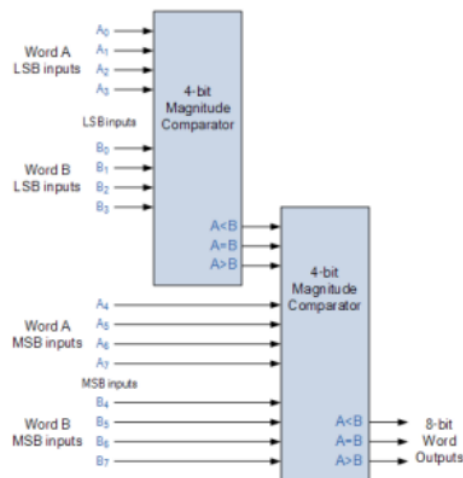


Figure 14: 8 bit magnitude comparator

A comprehensive explanation of the magnitude comparator and a good example of a practical design circuit

The comparator operates by comparing the highest order bit first and moving down to the next lowest bit. It operates in this method to produce the quickest result. Magnitude comparators are utilised significantly in ADC and arithmetic logic to execute diverse arithmetic processes. (Electronics Post, 2015)

2.5 Analogue to digital converters

An analogue signal is that of a continuous variable changing gradually over a variety of values. The analogue value produced by say a sensor is a scaled value of the quantity it represents. The Analogue to digital converter (ADC) converts an analogue signal to a digital format. A digital signal is required for microcontrollers and these signals allow for mathematical operation. The digital signals are also easily transmitted and stored. (Bolton, 2015)

Good explanation with good source

ADCs sample their input at discrete values of time which can cause error if the sample rate is below the Nyquist rate. The Nyquist rate is twice that of the highest frequency of interest. Any values below this can incur some signal loss. (Bolton, 2015)

The resolution of the ADC is determined by the word length. The word length of n bits of full scale analogue input is VFS divided into 2^n bits. ADC experience non-linearity error due to physical imperfections which cause their output to deviate from the linear function of their input. This can typically be 0.5 of the least significant bit. (Bolton, 2015)

ADCs come in several forms with the most commonly available being the successive approximation, ramp and flash. The flash ADC produces the fastest performance compared to the other variants. The settling time specifies the time taken by a converter to produce a word after the inputted analogue signal. A successive approximation converter typically has a conversion time of 1 to 10 μ s for the 8-bit variant. (Bolton, 2015)

The explanation here is very good and there are points that demonstrate a good understanding

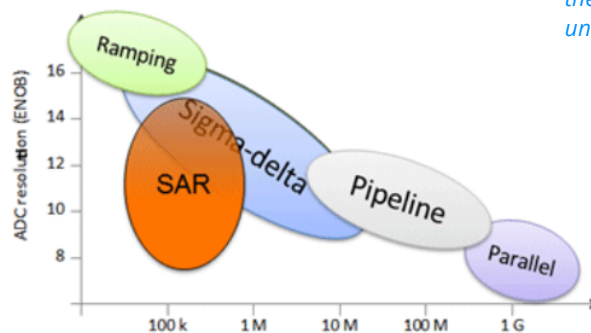


Figure 15: Comparison of ADCs resolution and sample rate

2.6 Load cells (namely LCKD-1kg)

Load cells are a type of sensor which converts force into an electrical signal which changes proportionately with the applied force. The strain gauge load cell is commonplace where the load cell body is a stiff spring that deforms in reaction to the applied force. The strain gauge is the sensing element which is bonded to the load cell body which produces the output signal. (Forge Logic UK, 2021)

The sub-miniature load cell LCKD series feature a rugged all stainless steel construction. The range is shown in figure 16. With a versatile operating range of 1 kg to 1000lbs. The LCKD-1Kg has a maximum capacity of 1 Kg. (Omega, 2021) The full data sheet can be found in Appendix A.

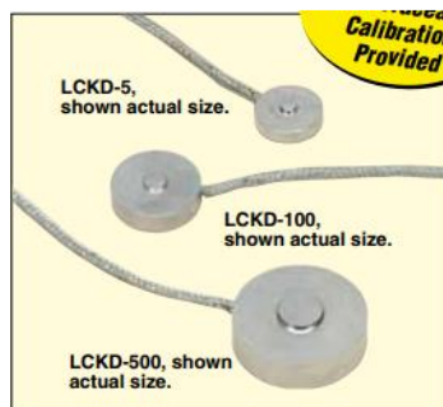


Figure 16: Omega LCKD load cell range

2.7 Instrumentation and signal conditioning

The raw value of a sensor generally must be conditioned for it to be suitable for use in control systems or displays. Signal conditioning is referred to the conversion of the output from the sensor into a suitable form for signal processing. (Bolton, 2015)

The strain gauge is an example where signal conditioning is required. The resistance across the strain gauge varies by the strain exerted. This resistive output is converted to a voltage using a potential divider circuit. A constant voltage is supplied across the strain gauge and another resistor in series. Here the output signal is the voltage across the series resistor. If the resistance of the strain gauge changes the fraction of the supply voltage across the series changes. The simple potential divider circuit provides an output voltage which is proportional to the fraction resistance produced by the strain gauge. (Bolton, 2015) To provide improved accuracy the Wheatstone bridge method can be implemented shown in figure 17.

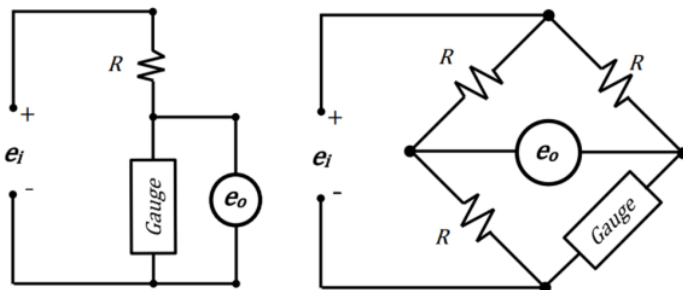


Figure 17: On the left potential divider circuit on right Wheatstone bridge circuit

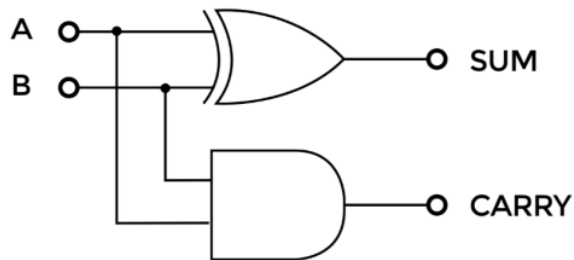
The bridge circuit is correct and these are commonly supplied with a sensor. However this would probably have an instrument amplifier as a stage in the signal condition process.

3. Design requirements

The 4-bit adder will be constructed from 4 full adders which contain two half adders.

3.1 Half adder

The half adder has two inputs being A and B and two outputs being sum and carry. The truth table for the half adder is shown in figure 19. The circuit is an EXOR gate which is in parallel with an AND gate. The EXOR gate relates to the sum output while the AND gate relates to the carry out output. (Coates, 2021)



A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Figure 18: Half adder truth table

Figure 19: Half adder logic diagram

3.2 Full adder

To produce a full adder two half adders are cascaded with the addition of an OR gate. The output of the sum is supplied to the input of the following adder and the carry out bit of both adders are supplied to an OR gate. The full adder's logic diagram and truth table is shown in figure 20.

A	B	Carry In	Sum	Carry out
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

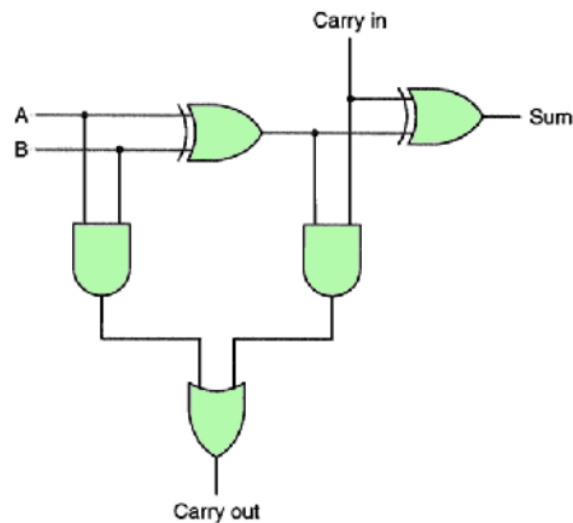
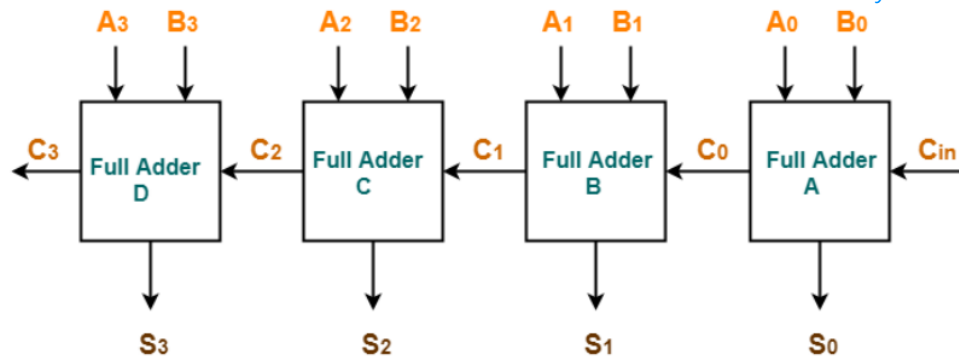


Figure 20: Full adder truth table and logic circuit

3.3 – 4-bit adder

A 4-bit adder involves the cascading of four full adders. The carry out output of the individual full adder is connected to the Cin of the following adder. The S output is the sum value calculated by the adder.

The process for the creation of the 4 bit adder is explained and clearly understood



4-bit Ripple Carry Adder

Figure 21: 4 bit adder block diagram (Singhal, 2021)

4 Implementation

4.1 Half adder

The half adder code which is written in VHDL is shown in figure 22.

```
1  library IEEE;
2  use IEEE.STD_LOGIC_1164.all;
3  use IEEE.STD_LOGIC_unsigned.all;
4
5  entity halfadder is
6  port (a,b : in std_logic;
7        s,c : out std_logic);
8  end halfadder;
9
10 architecture Behavioral of halfadder is
11
12 begin
13     s <= a xor b;
14     c <= a and b;
15
16 end Behavioral;
```

Layout of code is good

Figure 22: Half adder VHDL code

4.2 Full adder design

Using the half adders code twice a full adder is constructed. The hierarchy structure of the full adder is shown in figure 23. The code written to combine the half adders to produce a full adder is shown in figure 23.



Figure 23: Hierarchy structure of full adder

```

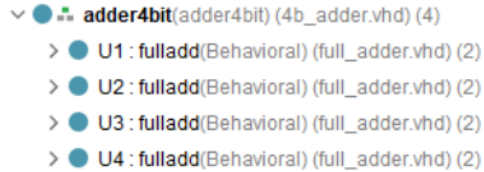
1  library IEEE;
2  use IEEE.std_logic_1164.all;
3  entity fulladd is
4  port(
5    a : in STD_LOGIC;
6    b : in STD_LOGIC;
7    cin : in STD_LOGIC;
8    cout : out STD_LOGIC;
9    s : out STD_LOGIC
10 );
11 end fulladd;
12 architecture Behavioral of fulladd is
13 component halfadder
14 port (
15   a : in STD_LOGIC;
16   b : in STD_LOGIC;
17   c : out STD_LOGIC;
18   s : out STD_LOGIC
19 );
20 end component;
21
22 signal c1 : STD_LOGIC;
23 signal c2 : STD_LOGIC;
24 signal s1 : STD_LOGIC;
25 begin
26 U1 : halfadder
27 port map(
28   a => a, b => b, c => c1, s => s1);
29 U2 : halfadder
30 port map(
31   a => s1, b => cin, c => c2, s => s);
32 cout <= c2 or c1;
33 end Behavioral;
34

```

Figure 24: VHDL code displaying inherited halfadders

4.3 – 4-bit adder design

The 4-bit adder structure includes four full adders. The hierarchy structure is shown in figure 25.



4 bit adder created from 4 full adders which are in turn made up of 2 half adders is explained here

Figure 25: Hierarchy structure of 4- bit adder

The full adders combined to produce a 4-bit adder VHDL code is shown in figure 26.

```
1  library IEEE;
2  use IEEE.std_logic_1164.all;
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
11  b0 : in STD_LOGIC;
12  b1 : in STD_LOGIC;
13  b2 : in STD_LOGIC;
14  b3 : in STD_LOGIC;
15
16  cout : out STD_LOGIC;
17
18  s0 : out STD_LOGIC;
19  s1 : out STD_LOGIC;
20  s2 : out STD_LOGIC;
21  s3 : out STD_LOGIC
22  );
23  end adder4bit;
```

```
1  library IEEE;
2  use IEEE.std_logic_1164.all;
3  entity adder4bit is...
24
25  architecture adder4bit of adder4bit is
26
27  component fulladd...
36
37  signal c1 : STD_LOGIC;
38  signal c2 : STD_LOGIC;
39  signal c3 : STD_LOGIC;
40
41  begin
42
43  U1 : fulladd
44  port map(
45  a => a0, b => b0, cin => cin, cout => c1,
46  s => s0);
47  U2 : fulladd
48  port map(
49  a => a1, b => b1, cin => c1, cout => c2,
50  s => s1);
51  U3 : fulladd
52  port map(
53  a => a2, b => b2, cin => c2, cout => c3,
54  s => s2);
55  U4 : fulladd
56  port map(
57  a => a3, b => b3, cin => c3, cout => cout,
58  s => s3);
59
60  end adder4bit;
61
```

Good layout of code for the 4 bit adder with the architecture made up 4 full adders with sum and carry variables set.

Figure 26: 4 bit adder VHDL code

4.4 – 4-bit magnitude comparator design

The 4-bit magnitude comparator VHDL code is shown in figure 27.

```
6  entity COMP_4_NI is...
24
25  architecture BEHAVIORAL of COMP_4_NI is
26  begin
27  PROCESS (A0, B0, A1, A2, A3, B1, B2, B3, AGTB, AEQB, ALTB)
28  BEGIN
29      if A3>B3 then OAGTB<='1';
30      OAEQB<='0';
31      OALTB<='0';
32      elsif A3<B3 then OAGTB<='0';
33      OAEQB<='0';
34      OALTB<='1';
35      elsif A3=B3 then
36      if A2>B2 then OAGTB<='1';
37      OAEQB<='0';
38      OALTB<='0';
39      elsif A2<B2 then OAGTB<='0';
40      OAEQB<='0';
41      OALTB<='1';
42      elsif A2=B2 then
43      if A1>B1 then OAGTB<='1';
44      OAEQB<='0';
45      OALTB<='0';
46      elsif A1<B1 then OAGTB<='0';
47      OAEQB<='0';
48      OALTB<='1';
49      elsif A1=B1 then
50      if A0>B0 then OAGTB<='1';
51      OAEQB<='0';
52      OALTB<='0';
53      elsif A0<B0 then OAGTB<='0';
54      OAEQB<='0';
55      OALTB<='1';
56      elsif A0=B0 then
57      if AGTB='1' and AEQB='0' and ALTB='0' then OAGTB<='1';
58      OAEQB<='0';
59      OALTB<='0';
60      elsif ALTB='1' and AEQB='0' and AGTB='1' then OAGTB<='0';
61      OAEQB<='0';
62      OALTB<='0';
63      elsif AEQB='1' and AGTB='0' and ALTB='0' then OAGTB<='0';
64      OAEQB<='1';
65      OALTB<='0';
66      elsif ALTB='1' and AEQB='0' and AGTB='0' then OAGTB<='0';
67      OAEQB<='0';
68      OALTB<='1';
69      elsif AEQB='1' then OAGTB<='0';
70      OAEQB<='1';
71      OALTB<='0';
72      elsif ALTB='0' and AEQB='0' and AGTB='0' then OAGTB<='1';
73      OAEQB<='0';
74      OALTB<='1';
75
76  end if;
77      end if;
78      end if;
79  end if;
80  end if;
81  END PROCESS;
82  end BEHAVIORAL;
```

It should be mentioned that the VHDL code here was created by the Multisim circuit design PLD

Figure 27: 4-bit magnitude comparator VHDL code

4.5 Combination of 4-bit adder and 4-bit comparator

The 4-bit adder and 4-bit magnitude comparator were packaged using the IP integrator. The IP designs were combined with the adder outputs being connected to the magnitude comparators inputs.

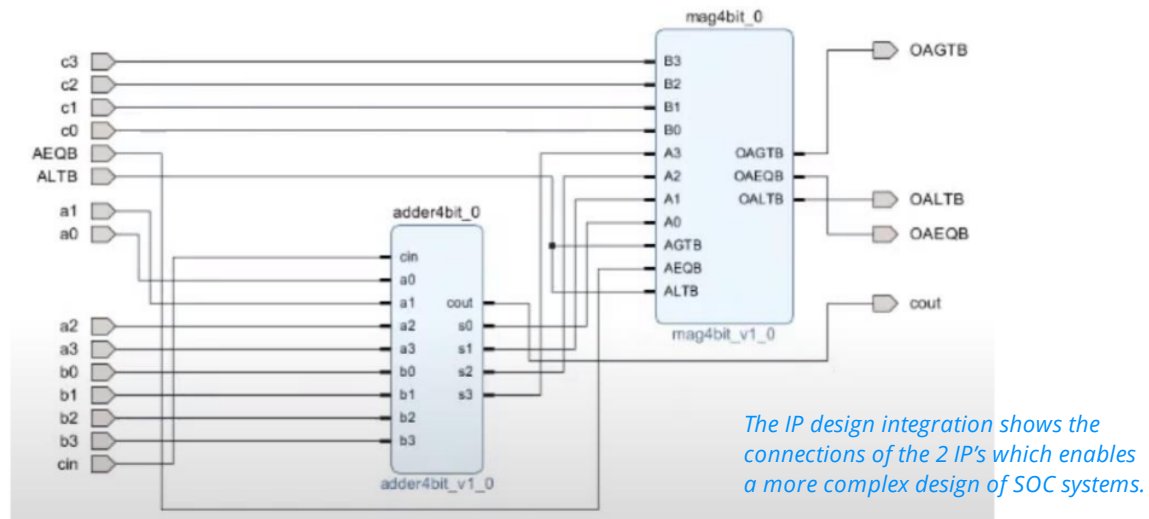


Figure 28: Combination of 4 bit adder and 4 bit magnitude comparator

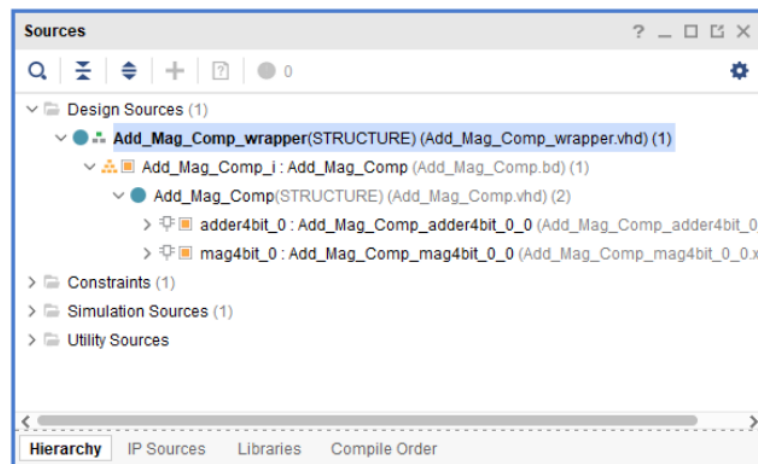


Figure 29: Combinational logic 4-bit adder and magnitude comparator hierarchical structure

The 8-bit magnitude comparator was constructed in the simulation package MultiSim. By cascading the 4-bit mag comparator with another 4-bit mag comparator an 8-bit mag comparator was constructed. The circuit shown in figure 30 is designed in PLD mode to enable downloading to the Nexys DDR board.

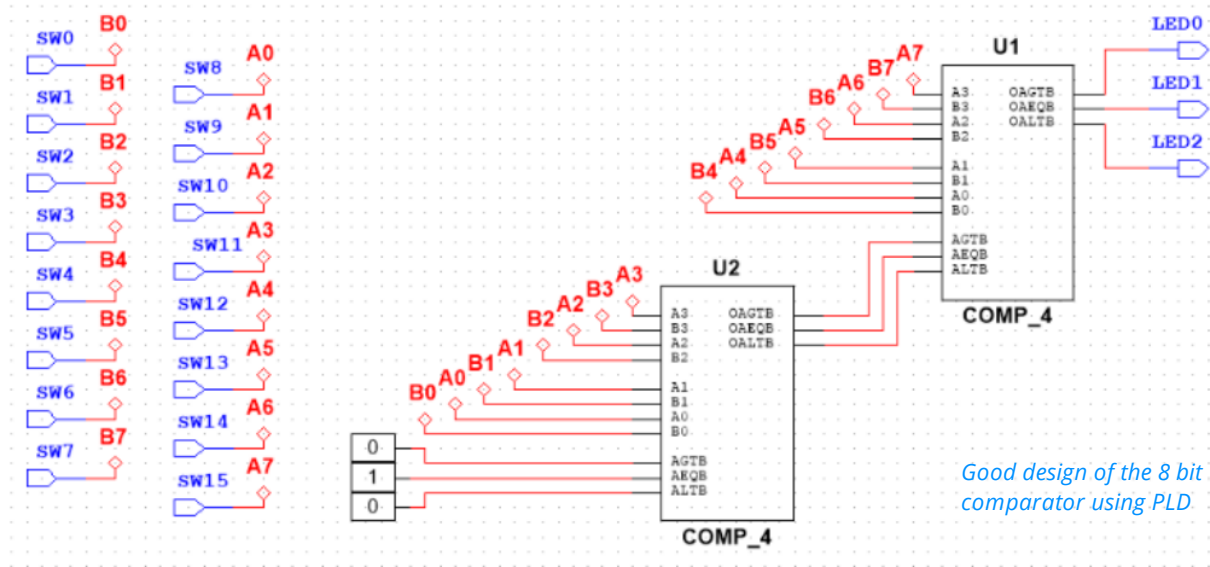


Figure 30: MultiSim 8-bit magnitude comparator constructed in PLD mode.

5 Testing, analysis and implementation

Following the design process testing and analysis was carried out to ensure the design met the specification. To produce the simulation test bench code was obtained from the testbench website found in appendix B and inputted into the test bench of the project with the resulting bitstreams.

5.1 4-bit adder simulation testing

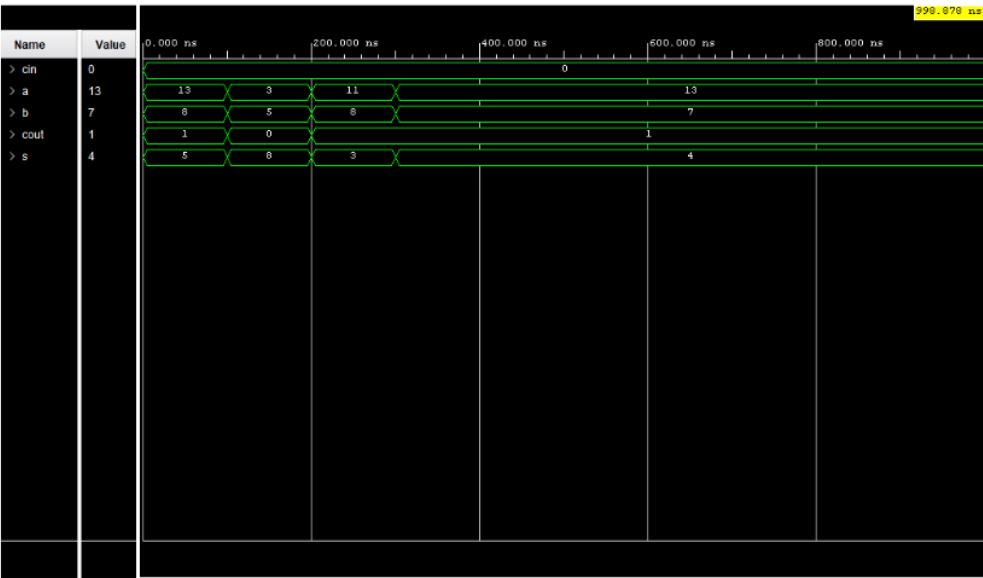


Figure 31: Simulation result for 4-bit adder.

Good result and good use of the virtual bus method for testing arrays

The test bench code can be found in figure 32. The table of result for the expected and simulation results is shown in table 2.

Table 2: Table of results 4-bit adder

Set A			Set B			Expected result			Actual result		
Bin	Dec	Cin	Binary	Dec	Cin	Sum	Cout	Value	Sum	Cout	Value
1101	13	0	1000	8	0	5	1	21	5	1	21
0011	3	0	0101	5	0	5	0	5	5	0	5
1011	11	0	1000	8	0	3	1	19	3	1	19
1101	13	0	0111	7	0	4	1	20	4	1	20

Good use of tables

Under simulation the expected and simulated result produced the expected results.

```

59 stimuli : process
60 begin
61     -- EDIT Adapt initialization as needed
62     B3 <= '1';
63     B2 <= '0';
64     B1 <= '1';
65     B0 <= '0';
66     A3 <= '1';
67     A2 <= '0';
68     A1 <= '1';
69     A0 <= '0';
70     AGTB <= '0';
71     AEQB <= '1';
72     ALTB <= '0';
73     wait for 100 ns;
74     B3 <= '1';
75     B2 <= '0';
76     B1 <= '1';
77     B0 <= '0';
78     A3 <= '0';
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 <= '1';
90     A3 <= '1';
91     A2 <= '0';
92     A1 <= '1';
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;
102
103 end tb;

```

Figure 32: Test bench code for 4-bit adder

5.2 4-bit mag comparator testing

Repeating the process but with the 4-bit comparator the simulation produced the result shown in figure 33.

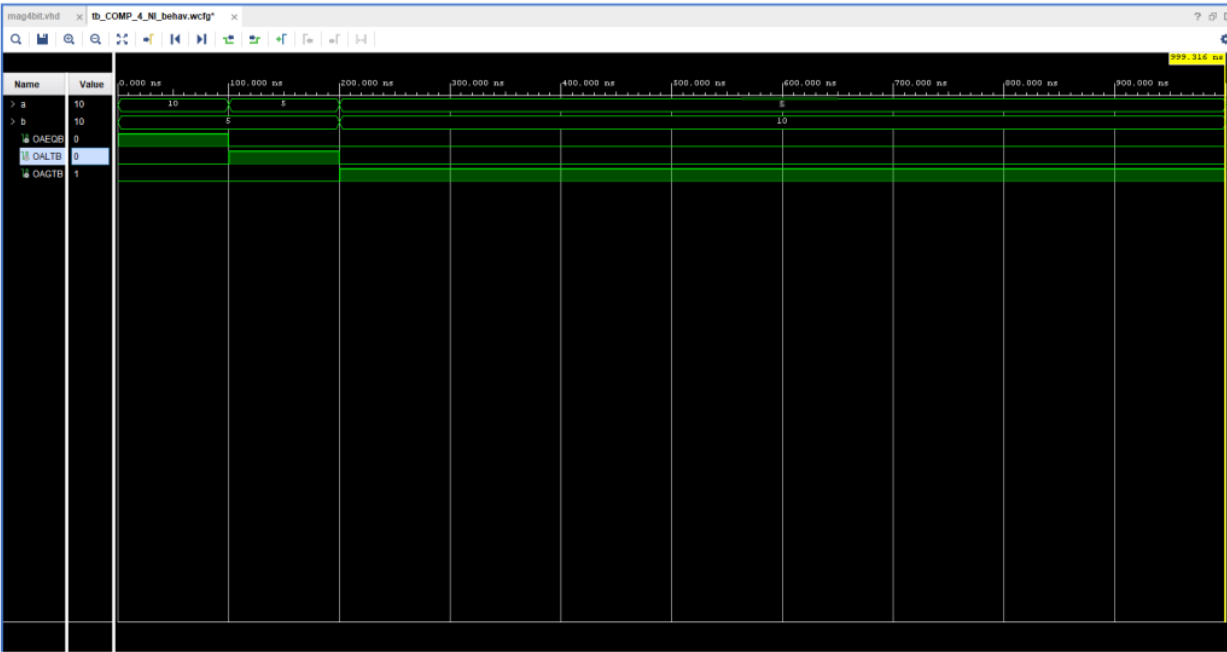


Figure 33: 4-bit magnitude comparator result

The test bench code is shown in figure 34.

Table 3: Table of result 4-bit magnitude comparator

Set A		Set B		Inputs			Output	
Bin	Decimal	Bin	Decimal	AGTB	AETB	ALTB	Expected	Actual
1010	10	0101	5	0	1	0	OAGTB	OAGTB
0101	5	0101	5	0	1	0	OAEQB	OAEQB
0101	5	1010	10	0	1	0	OALTB	OALTB

The simulated produced the same result as the expected result.

```

62  ○ B3 <= '0';
63  ○ B2 <= '1';
64  ○ B1 <= '0';
65  ○ B0 <= '1';
66  ○ A3 <= '1';
67  ○ A2 <= '0';
68  ○ A1 <= '1';
69  ○ A0 <= '0';
70  ○ AGTB <= '0';
71  ○ AEQB <= '1';
72  ○ ALTB <= '0';
73  ○ wait for 100 ns;
74  ○ B3 <= '0';
75  ○ B2 <= '1';
76  ○ B1 <= '1';
77  ○ B0 <= '0';
78  ○ A3 <= '0';
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 <= '1';
90  ○ A3 <= '1';
91  ○ A2 <= '0';
92  ○ A1 <= '1';
93  ○ A0 <= '0';
94  ○ AGTB <= '0';
95  ○ AEQB <= '1';
96  ○ ALTB <= '0';
97  ○ wait for 100 ns;

```

Correct data for test bench

Figure 34: Test bench code for 4-bit magnitude comparator

5.3 – 4-bit adder comparator combinational logic practical testing

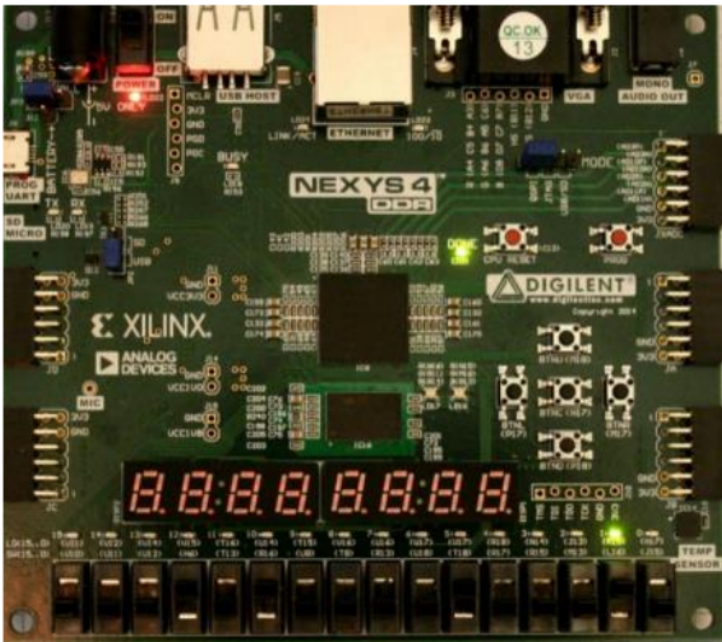


Figure 35: 4-bit adder comparator combinational logic test 1 = A=B

S15	S14	S13	S12	S11	S10	S9	S8	S7	S6	S5	S4	S3	S2	S1	S0			
Cin	ALTB = 1	AEQB = 0		C3	C2	C1	C0	B3	B2	B1	B0	A3	A2	A1	A0	LED0	LED1	LED2
	1			Set value				Sum "B"				Sum "A"				AGTB	AEQB	ALTB
				0	1	0	0	0	0	1	0	0	0	1	0			
				4				2				2						

Table 4: 4-bit adder comparator combinational logic result test 1 = A=B

Result	Sum A + B	4
	Set value C	4
	A=B	

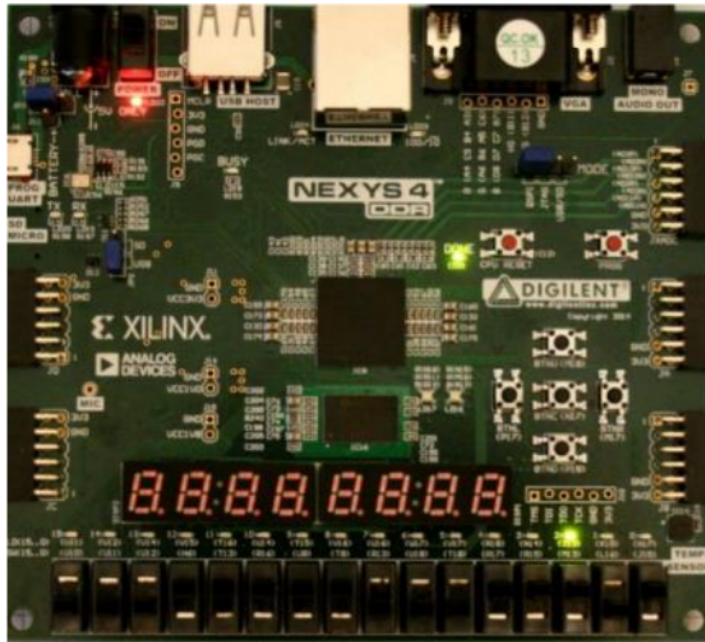


Figure 36: 4-bit adder comparator combinational logic test 2 = $A < B$

S15	S14	S13	S12	S11	S10	S9	S8	S7	S6	S5	S4	S3	S2	S1	S0			
Cin	ALTB=1	AEQB=0		C3	C2	C1	C0	B3	B2	B1	B0	A3	A2	A1	A0	LED0	LED1	LED2
	1			Set value				Sum "B"				Sum "A"				AGTB	AEQB	ALTB
				1	1	1	1	0	0	0	1	1	1	0	1			
				15				1				13						

Table 5: 4-bit adder comparator combinational logic result test 2 = $A < B$

Result	Sum A + B	14
	Set value C	15
		$A < B$

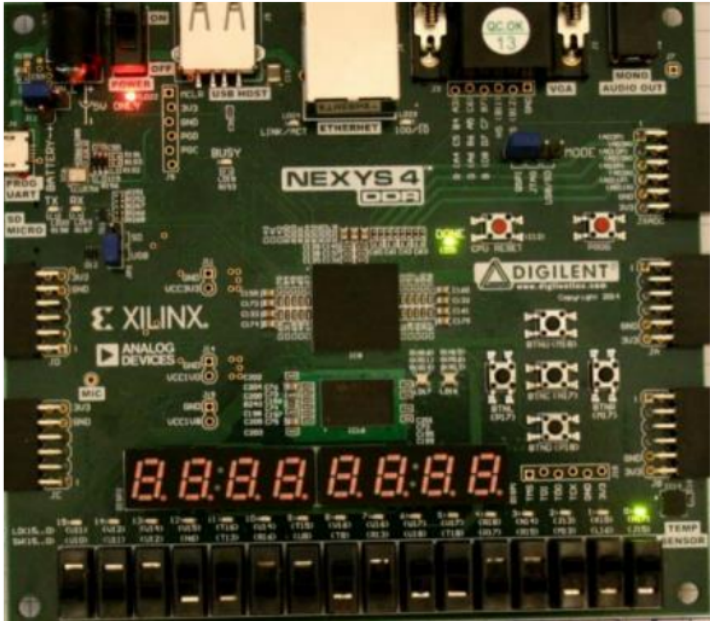


Figure 37: 4-bit adder comparator combinational logic result test 3 = A>B

S15	S14	S13	S12	S11	S10	S9	S8	S7	S6	S5	S4	S3	S2	S1	S0			
Cin	ALTB = 1	AEQB = 0		C3	C2	C1	C0	B3	B2	B1	B0	A3	A2	A1	A0	LED0	LED1	LED2
	1			Set value				Sum "B"				Sum "A"				AGTB	AEQB	ALTB
				1	0	0	1	0	1	1	0	0	1	1	1			
				9				6				7						

Table 6: 4-bit adder comparator combinational logic result test 3 = A>B

Result	Sum A + B	13
	Set value C	9
		A>B

5.4 - 8-bit magnitude comparator simulation

The 8-bit magnitude comparator was simulated in MultiSim by changing input values. A series of test were conducted to prove correct operation.

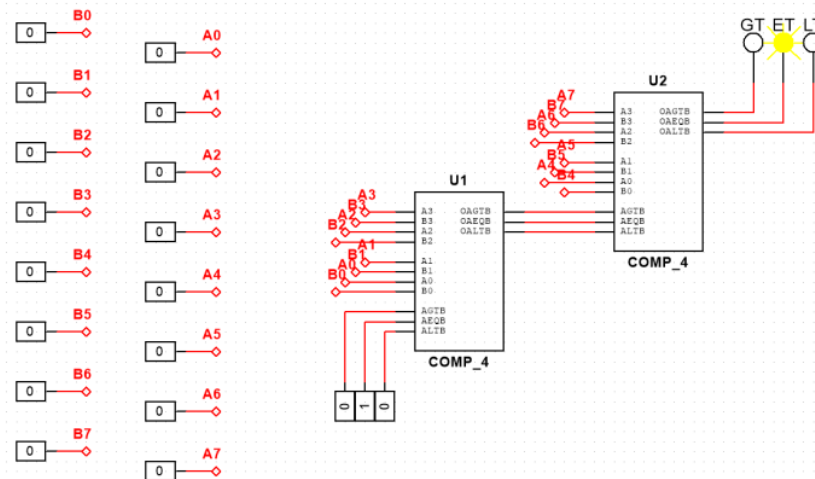


Figure 38: 8-bit comparator simulation test 1 A=B

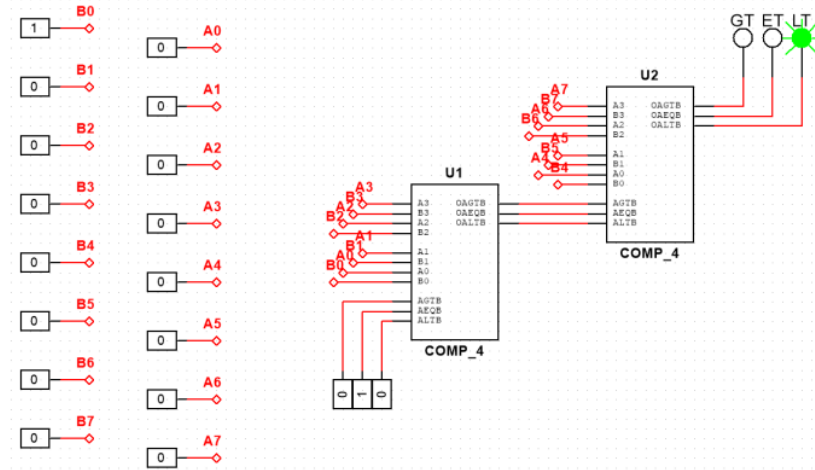
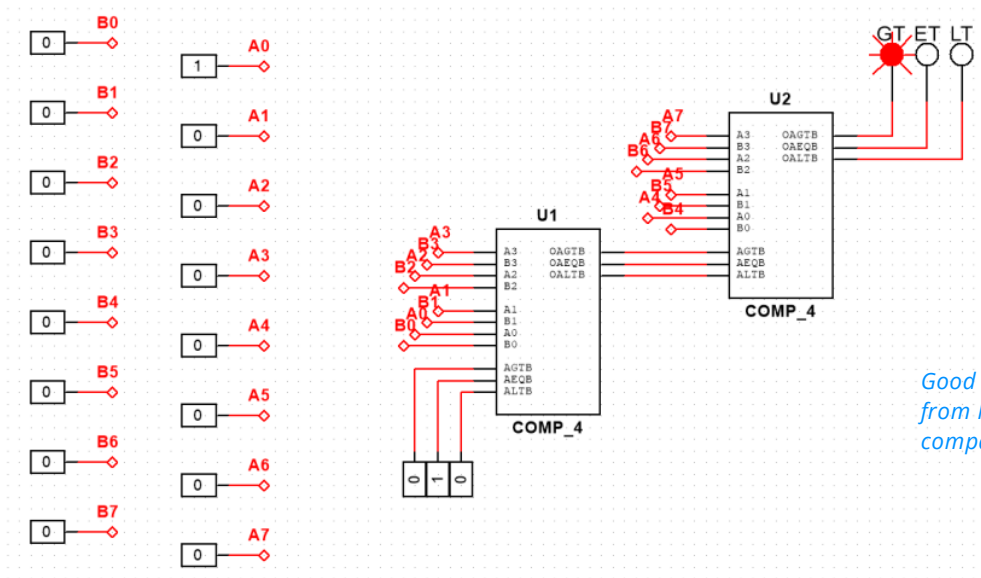


Figure 39: 8-bit comparator simulation test 2 A<B



Good set of results
from Msim 8 bit
comparator

Figure 40: 8-bit comparator simulation test 3 A>B

5.5 – 8-Bit magnitude comparator practical testing

The 8-Bit magnitude comparator design was downloaded to the Nexys FPGA board.

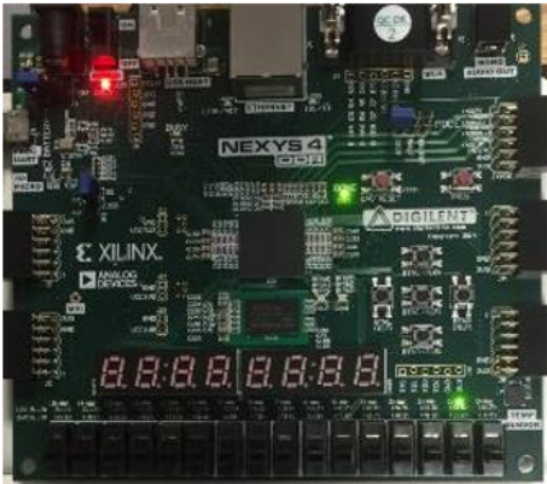


Figure 41: 8-bit magnitude comparator practical result Test 1 - A=B

B								A										
S15	S14	S13	S12	S11	S11	S10	S9	S8	S7	S6	S5	S4	S3	S2	S0			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	LED0	LED1	LED2
0								0								AGTB	AEQB	ALTB

Result	A value	0
	B value	0
	A=B	

Table 7: 8-Bit magnitude comparator result – A=B

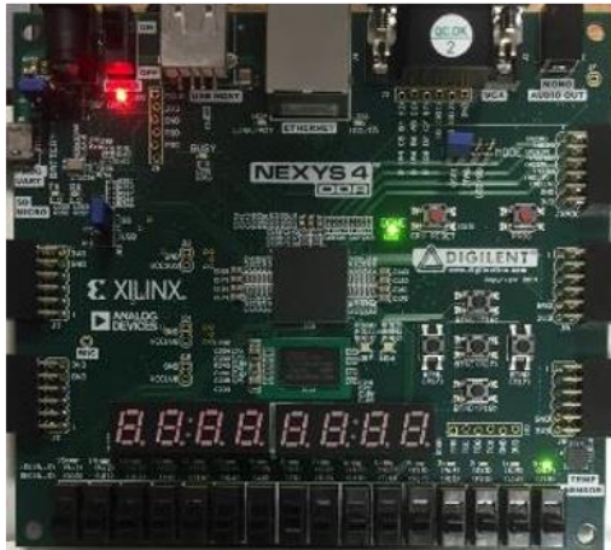


Figure 42: 8-Bit magnitude comparator practical result test 2 - $A < B$

B								A										
S15	S14	S13	S12	S11	S11	S10	S9	S8	S7	S6	S5	S4	S3	S2	S0			
0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	LED0	LED1	LED2
1								0								AGTB	AEQB	ALTB

Result	A value	0
	B value	1
		$A < B$

Table 8: 8-Bit comparator practical result test 2 – $A < B$

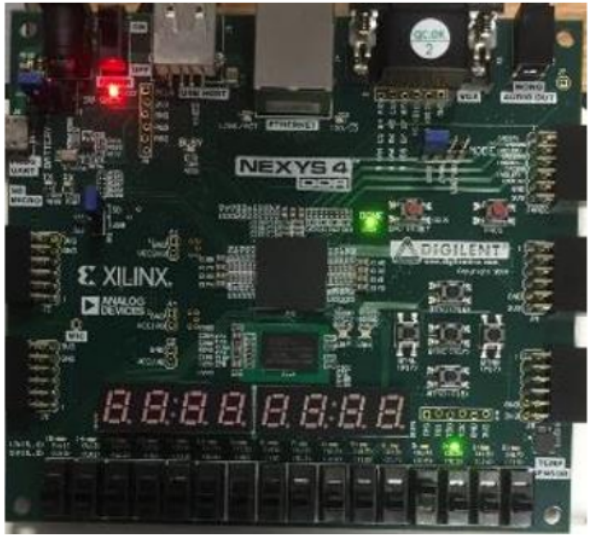


Figure 43: 8-Bit magnitude comparator practical result test 3 - $A > B$

B								A										
S15	S14	S13	S12	S11	S11	S10	S9	S8	S7	S6	S5	S4	S3	S2	S0			
0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	LED0	LED1	LED2
0								1								AGTB	AEOB	ALTB

Result	A value	0
	B value	1
		$A > B$

Table 9: 8-Bit magnitude comparator practical result test 3- $A > B$

Good student

6. Evaluation

The research conducted provided the background knowledge required to perform the tasks of element 1. The VHDL hierarchy structure understanding was imperative for the design of both the 4-bit adder and 4-bit comparator. IP integration unquestionably streamlined the combination of the two designs to produce the 4-bit adder comparator combinational logic design. Further research was performed into NI DAQmx, ADCs, load cells, instrumentation and signal conditioning which will be beneficial for the next elements.

The 4-bit adder contains 4 full adders where the full adders contain two half adders. This is the premise of the hierarchical inheritance of the VHDL structured program. The 4-bit adder functioned as expected in simulation and practical testing on the Nexys FPGA board. Following this the 4-bit magnitude comparator was constructed, simulated, and tested practically. With the systems functioning as intended they were IP packaged and combined in a separate program. This was downloaded to the Nexys FPGA and functioned as expected.

The 8-bit comparator was constructed in Multisim by cascading two 4-bit magnitude comparators. The design was constructed in PLD mode to enable the program to be downloaded to the FPGA board. The logic performed as expected concluding all design operated as required.

The objectives achieved in this element have provided an insight into adder and magnitude comparator design. These designs are the basis of ADC conversion with will be implemented in the system design.

Good evaluation which relates to the next stage in element 2.

6. Appendix

Appendix A

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

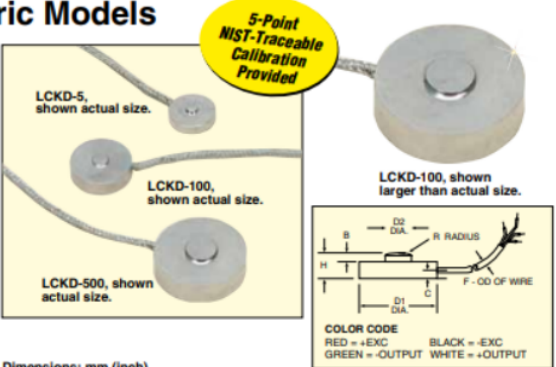
Full Scale Deflection: 0.001 to 0.003"

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 1 kg to 50 lb	9.6 (0.38)	2.2 (0.09)	3.0 (0.12)	0.76 (0.03)	1.0 (0.04)	1.3 (0.05)	6.3 (0.25)
500 to 1000 N 100 to 250 lb	13 (0.50)	3.0 (0.12)	3.8 (0.15)	0.51 (0.02)	1.5 (0.06)	1.3 (0.05)	13 (0.5)
2000 to 5000 N 500 to 1000 lb	19 (0.75)	6.1 (0.24)	6.4 (0.25)	0.76 (0.03)	2.5 (0.10)	1.3 (0.05)	101 (4)

STANDARD MODELS

To Order			
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.

Appendix B



References

Bolton, W., 2015. *Instrumentation and Control Systems*. Second ed. Oxford: Joe Hayton.

Coates, E., 2021. *Raspberrypi*. [Online]

Available at: <https://projects.raspberrypi.org/en/projects/halfadder>
[Accessed 02 1 2021].

Electronics Post, 2015. *Electronics Post*. [Online]

Available at: <https://electronicspost.com/digital-comparator/>
[Accessed 01 02 2021].

Essick, J., 2019. *Hands-on Introduction to LabView for Scientists and Engineers*. Fourth Edition ed. New York : Oxford University Press.

Forge Logic UK, 2021. *Forge Logic UK*. [Online]

Available at: <https://www.force-logic.co.uk/what-is-a-load-cell/>
[Accessed 02 01 2021].

Franz, K., 2021. *Digilent Blog*. [Online]

Available at: <https://blog.digilentinc.com/what-is-an-ip-and-how-do-you-create-one/>
[Accessed 02 01 2021].

National Instruments, 2019. *National Instruments*. [Online]

Available at: <https://knowledge.ni.com/KnowledgeArticleDetails?id=kA00Z000000P8baSAC&l=en-GB>
[Accessed 02 01 2021].

Omega, 2021. *Farnell*. [Online]

Available at: <http://www.farnell.com/datasheets/2340205.pdf>
[Accessed 02 01 2021].

Singhal, A., 2021. *Gatevidyalay*. [Online]

Available at: <https://www.gatevidyalay.com/ripple-carry-adder/>
[Accessed 02 01 2021].

Wakerly, J. F., 2006. *Digital Design Principles and Practices*. Fourth Edition ed. s.l.:Pearson Education.

*Good number of
relevant sources*

Course: HND Electrical and Electronic Engineering
Module: SGEE5007 Virtual Instrumentation and Control Systems
Portfolio: Magnitude Comparator (Element 2)
Name : Gareth Wilkins
University number: 1809804
Date submitted: 20 May 2021

Contents

List of figures.....	3
1. Introduction	4
2. Load cells and Signal Conditioning.....	5
2.1 Unity gain of instrument amplifier.....	5
3. Analogue to digital converter	8
3.1 10-bit ADC.....	8
3.2 12-bit ADC.....	10
3.3 8-bit ADC.....	12
4. Nexys 4 DDR FPGA board.....	16
5. Magnitude comparator practical.....	17
6. LabVIEW	18
7. Testing and Analysis.....	21
8. Evaluation	22
9.1 Appendix.....	23

*Good table of
contents*

List of figures

Figure 1: Schematic of unity gain amplifier 5

Figure 2: Instrument amplifier MultiSim schematic. 6

Figure 3: Simulation results displaying response of load cell 7

Figure 4: 10 bit ADC LabVIEW program 8

Figure 5: Formula block for 10 bit ADC 8

Figure 6: 10-bit ADC with analogue input of 5 showing full scale 9

Figure 7: 10-bit ADC with analogue input of 2.5 constituting half scale 9

Figure 8: Formula block for 12-bit ADC..... 10

Figure 9: 12-bit ADC with analogue input of 5 showing full scale 11

Figure 10: 12-bit ADC with analogue input of 2.5 constituting half scale 11

Figure 11: 8-bit ADC LabVIEW program 12

Figure 12: Formula block for 8-bit ADC..... 12

Figure 13: Analogue value of 1V showing a Boolean value of 51 displayed on panel..... 13

Figure 14: Analogue value of 1V showing a Boolean value of 51 displayed on NI Elvis II board..... 13

Figure 15: Analogue value of 2.5V showing a Boolean value of 128 displayed on panel..... 14

Figure 16: Analogue value of 2.5V showing a Boolean value of 128 displayed on NI Elvis II board..... 14

Figure 17: Analogue value of 5V showing a Boolean value of 255 displayed on panel..... 15

Figure 18: Analogue value of 5V showing a Boolean value of 255 displayed on NI Elvis II board..... 15

Figure 19: Nexys 4 DDR Pin layout 16

Figure 20: Nexys 4 DDR component layout 16

Figure 21: Practical setup comprising NI Elvis II board with Nexys 4 DDR as magnitude comparator. 17

Figure 22: LabVIEW program 18

Figure 23: Formula block 1..... 19

Figure 24: Formula block 2..... 19

Figure 25: Formula block 3..... 20

Figure 26: Formula block 4..... 20

Figure 27: Equal to displayed on panel 21

Figure 28: Less than displayed on panel 21

Figure 29: Greater than displayed on panel 21

Table 1: Results for simulating force on the load cell 7

Good use of table for
test analysis

G
o
o
d
l
i
s
t
o
f
f
i
g
u
r
e
s

1. Introduction

This element evaluates unity gain of instrument amplifiers and an instrument amplifier with bridge for a load sensor. Unity gain was found when the input and output provide an equal relationship. The instrument amplifier with bridge for load cell sensor was tested and the linear response was obtained. Prototypes of the 10- and 12- bit ADC were constructed showing the improvement in accuracy by the increase in resolution. A prototype of an 8-bit ADC with a virtual analogue input was constructed where the results were shown on the panel and on the NI Elvis board. The magnitude comparator which was previously developed in element 1 on the Nexys 4 DDR FPGA board was implemented with the LabVIEW program and a series of test were conducted to establish equal to, greater than and less than. A conclusion was established detailing that the tasks are preliminary prototypes leading to the final element.

*Use the introduction as the
future tense for the element*

2. Load cells and Signal Conditioning

2.1 Unity gain of instrument amplifier

An instrument amplifier with unity gain is shown in figure 1. The concept of unity gain is the notion that the input and output should be equal in relationship. To achieve unity gain, the gain of the amplifier must match the output to attain a magnitude comparable to that of the input.

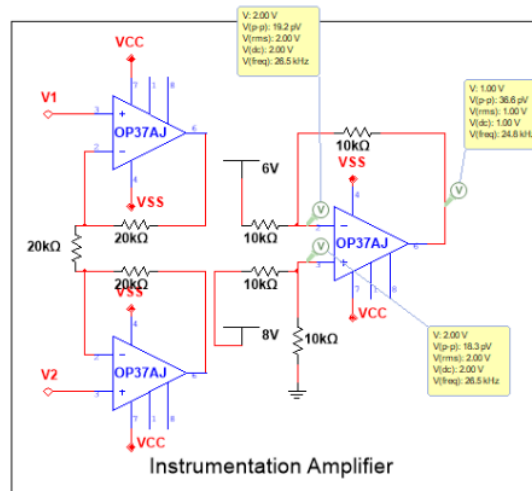


Figure 1: Schematic of unity gain amplifier

Good example of the op amp at unity gain

The amplifier circuit as shown in figure 1 will be simulated to resemble the Omega LCKD load cells. The Omega LCKD load cells were evaluated in element 1. To simulate force on the load cells the potentiometers were adjusted representing the alteration in the load cells value. Loading the amplifier was simulated by increasing opposite resistor values and decreasing the opposing resistor values. With a value of 50% on all potentiometers the load cell will signify zero grams. Conversely with 100% and 0% potentiometer configuration on opposing corners this will represent full load of 1000g.

To accurately condition the signal of the amplifier a gain value of 500 was needed. This was demonstrated with the following calculations:

@1000g and 5V excitation $(V2 - V1) = 10\text{mV}$

@1000g $(V_{out}) = 5\text{V}$

Gain (G) = $5\text{V}/0.01\text{mV} = 500$

To achieve the desired gain value of 500 the resistor values are modified to obtain a gain of 5 for the buffer amplifier and a gain of 100 for the differential amplifier. The calculation with the modified resistor values are shown below.

$$\text{Buffer Gain} = \frac{1 + (2 \times 60k)}{30k} = 5$$

$$\text{Diff Amp Gain} = \frac{30000}{30} = 100$$

Gain of
amplifier
is
identified

An overall gain of 500 is obtained for the instrument amplifier owing to the combination of the buffer and differential amplifier gain.

The instrument amplifier resistor values were modified as per the calculations and illustrated in figure 2.

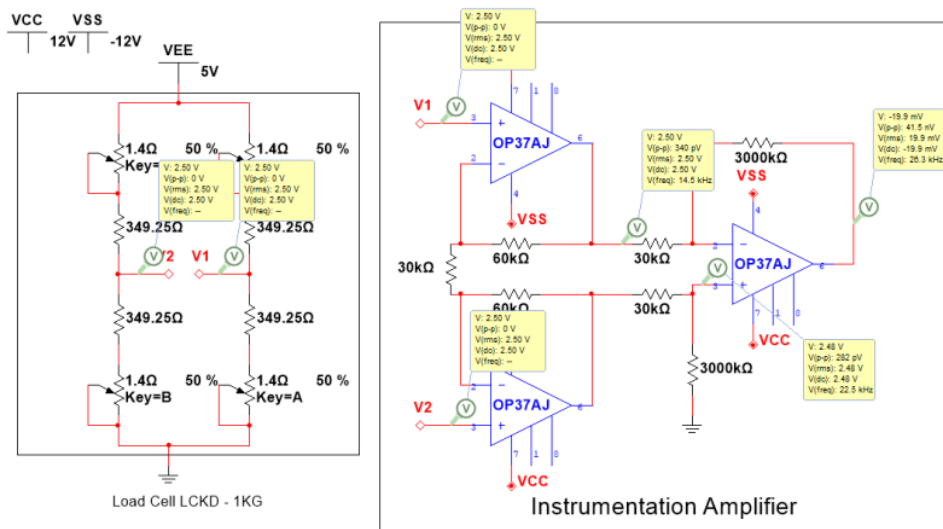


Figure 2: Instrument amplifier MultiSim schematic.

To simulate the loading effect of a 100g increment from 0g to 1000g a 5% increment was applied to the opposing potentiometers and a 5% deduction was applied to the others. The resultant simulations can be reviewed in appendix 9.1 with the results tabulated in table 1.

Table 1: Results for simulating force on the load cell

% Resistance	Grams	Voltage	Linearity error	Voltage removed	error
50	0	0.0199	0.0199	0	
55	100	0.48	0.02	0.5	
60	200	0.98	0.02	1	
65	300	1.48	0.02	1.5	
70	400	1.98	0.02	2	
75	500	2.48	0.02	2.5	
80	600	2.98	0.02	3	
85	700	3.48	0.02	3.5	
90	800	3.98	0.02	4	
95	900	4.48	0.02	4.5	
100	1000	4.98	0.02	5	

Good table of values

The table shows that the increase of load has produced a comparable output for the varying load determining an accurate signal conditioning model. The increase of 5% with the potentiometers simulating 100g of force produces a 0.5V increase in relation with the output. There is negligible error present ranging from 0.0199 – 0.02V which can be attributed to leakage and is exhibited during all setpoint and can consequently be disregarded. The signal conditioning is graphically depicted in figure 3 where the resolution of 100g=0.5V is shown with a linear response to full scale of 1000g=5V.

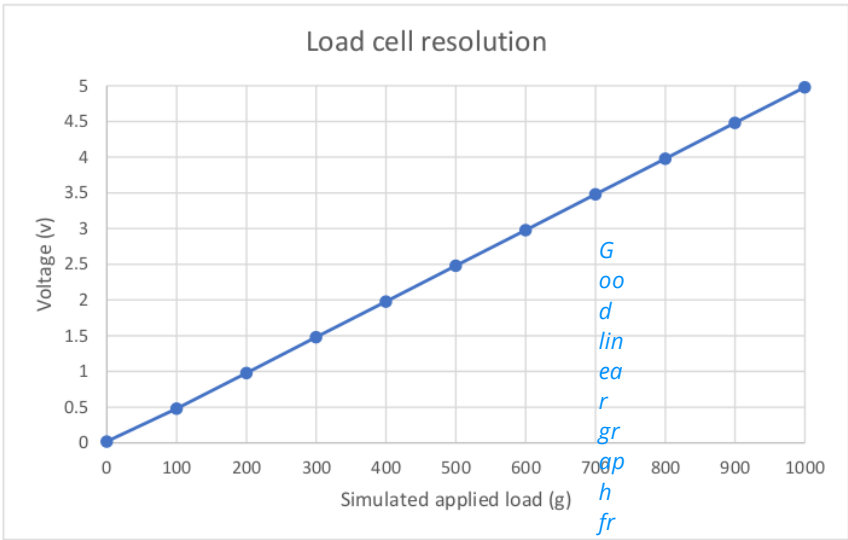


Figure 3: Simulation results displaying response of load cell

Good linear graph from results

3. Analogue to digital converter

3.1 10-bit ADC

Utilising the Labview software an ADC of 10-bit resolution was constructed. The program is shown in figure 4 with the formula block shown in figure 5. The range of the analogue input is 0-5. A 10 bit word contains 1024 bits so a range of 0-1023. To analyse the performance of the 10 bit the slider on the panel was slid to full scale and half scale. With full load of 5V applied all 10 bit digital output indicators illuminate representing a Boolean value of 1023 which is full scale shown in figure 6. Reducing the analogue input to half being 2.5 the resultant Boolean digital output illuminate producing a Boolean value of 511.5 shown in figure 7.

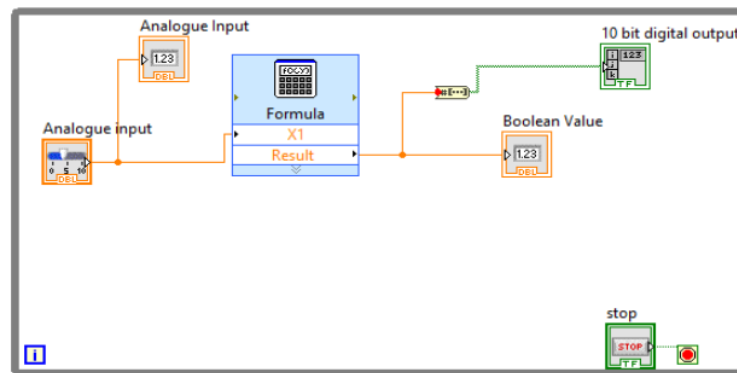


Figure 4: 10 bit ADC LabVIEW program

The formula block for the 10-bit ADC is $(X1/5)*1024$.

1024 – Resolution 10 bit

5 is max scale (Analogue)

X1 – Analogue input signal

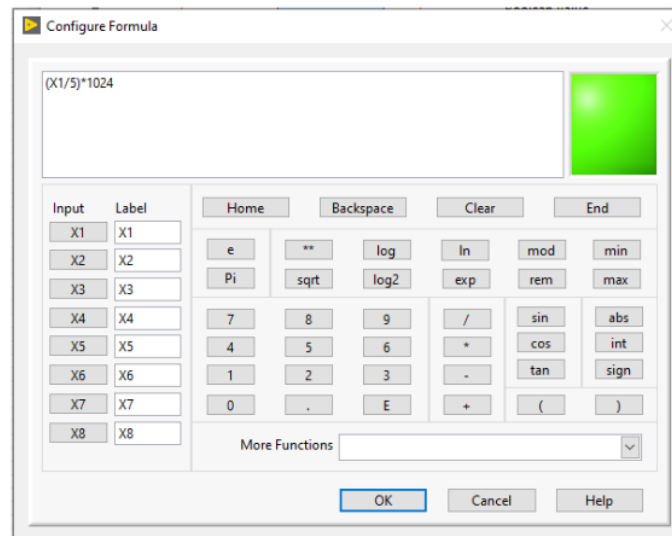


Figure 5: Formula block for 10 bit ADC

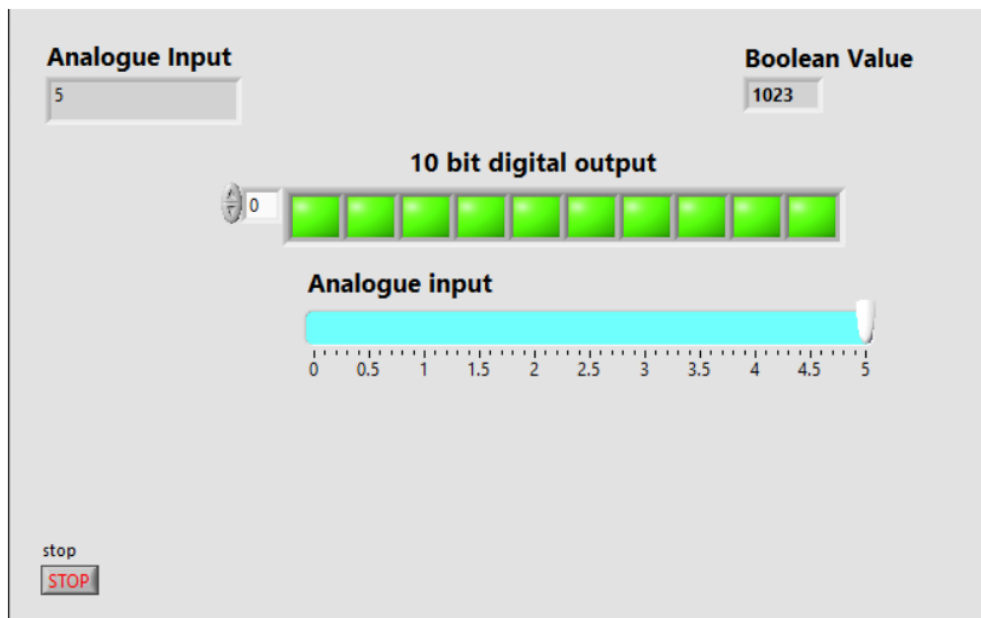


Figure 6: 10-bit ADC with analogue input of 5 showing full scale

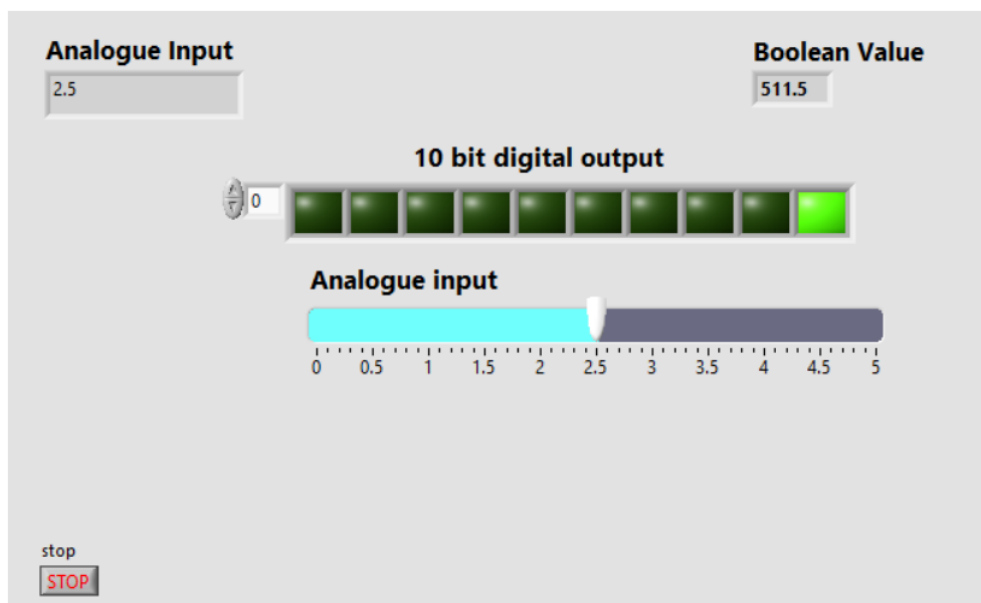


Figure 7: 10-bit ADC with analogue input of 2.5 constituting half scale

Good design method for 10 bit ADC with max value held

3.2 12-bit ADC

The ADC’s resolution was increased to 12 bits to analyse its effects. Two further digital bits have been added to the panel with a resolution of 4096 bits. The modified formula block is illustrated in figure 8.

4096 – Resolution 12-bit
5 – Max Scale Analogue
X1 – Analogue input signal

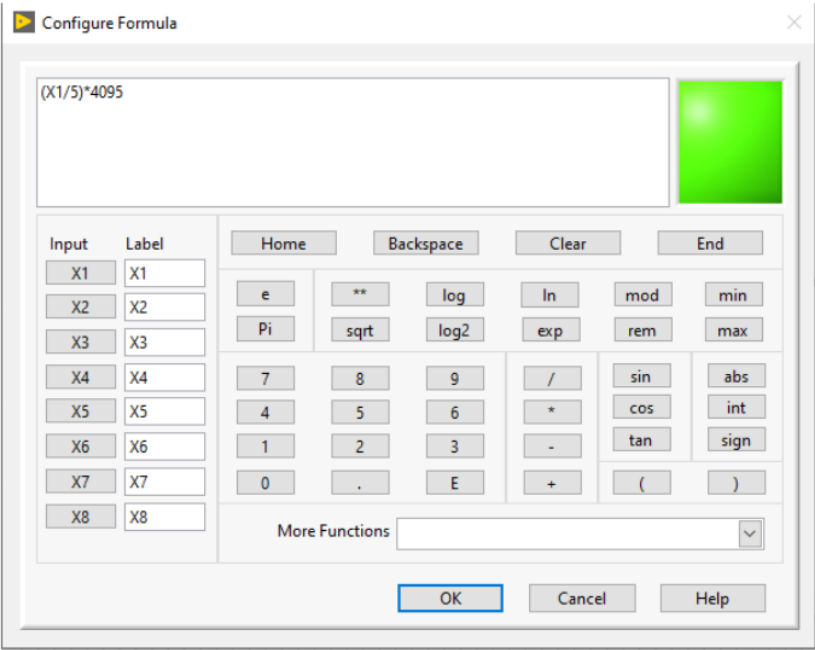


Figure 8: Formula block for 12-bit ADC

Again, full scale and half scale are analysed. With an analogue input of 5 being full scale all 12-bit digital output indicators illuminate with a Boolean value of 4095 this is shown in figure 9. With half analogue input being 2.5 the corresponding digital output indicator illuminate with a Boolean value of 2047.5 this is shown in figure 10.

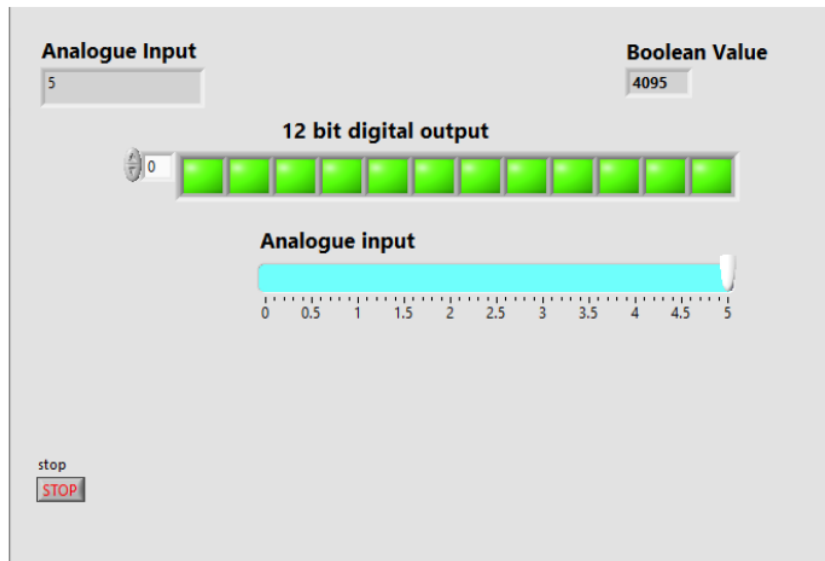


Figure 9: 12-bit ADC with analogue input of 5 showing full scale

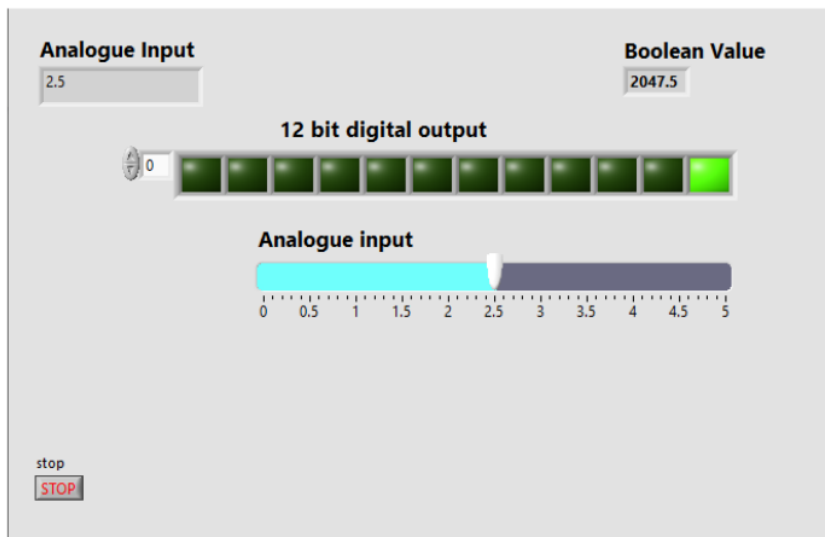


Figure 10: 12-bit ADC with analogue input of 2.5 constituting half scale

Good design of 12 bit ADC with max value held

3.3 8-bit ADC

Illustrated in figure 11 an analogue input, an 8-bit Boolean digital output, and the Boolean value was produced on the panel. A potentiometer is utilised to control the analogue input voltage. A virtual input was configured. The formula configured in the block diagram is shown in figure 12.

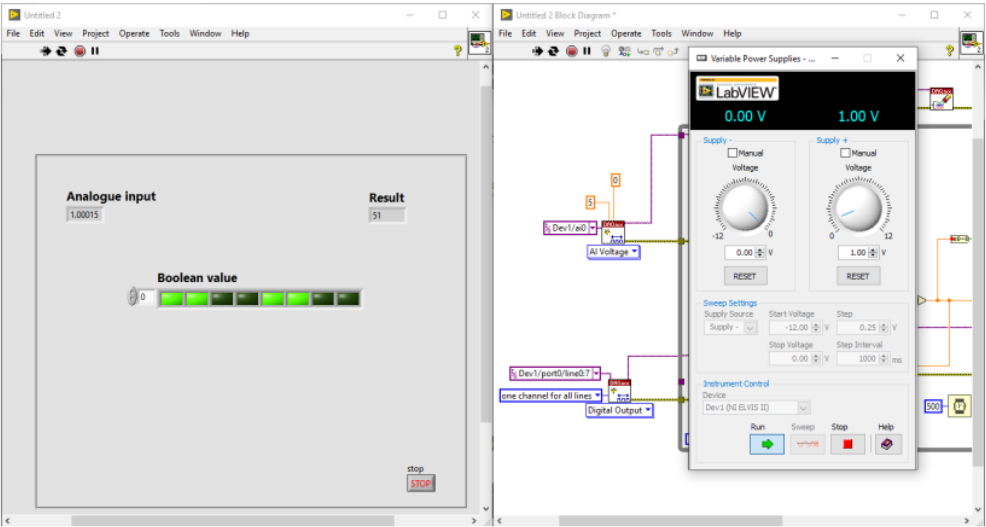
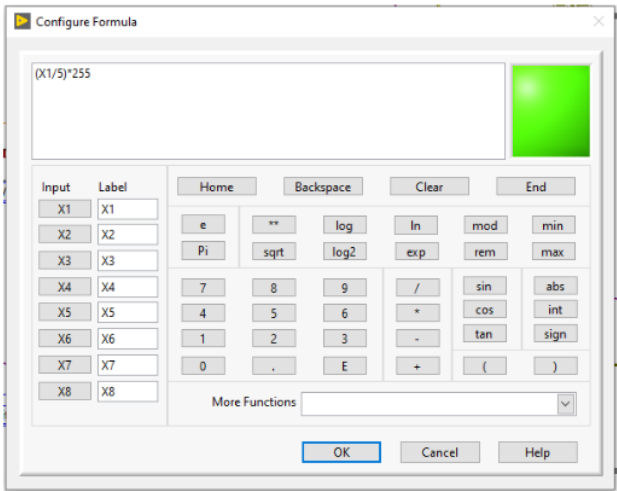


Figure 11: 8-bit ADC LabVIEW program



- X1 – Analogue input signal
- 5 – Max voltage scale
- 255 – 8-bit resolution

Figure 12: Formula block for 8-bit ADC

The LabVIEW program is configured so that the NI Elvis board will display the Boolean values alongside the panel in the software. The values of 1V, 2.5V and 5V were tested shown in figures 13-18 and the digital boolean values on the panel corroborated with the boolean values on the NI Elvis II board.

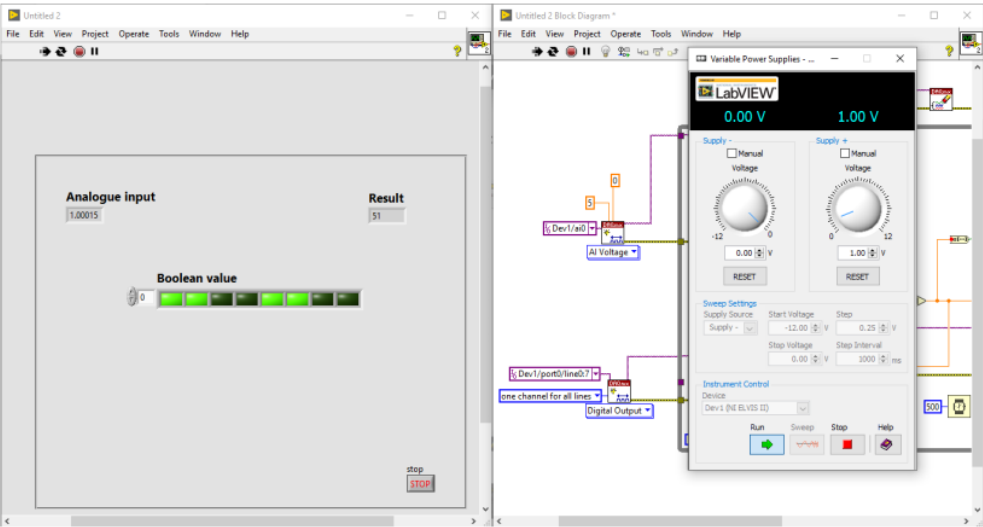


Figure 13: Analogue value of 1V showing a Boolean value of 51 displayed on panel.

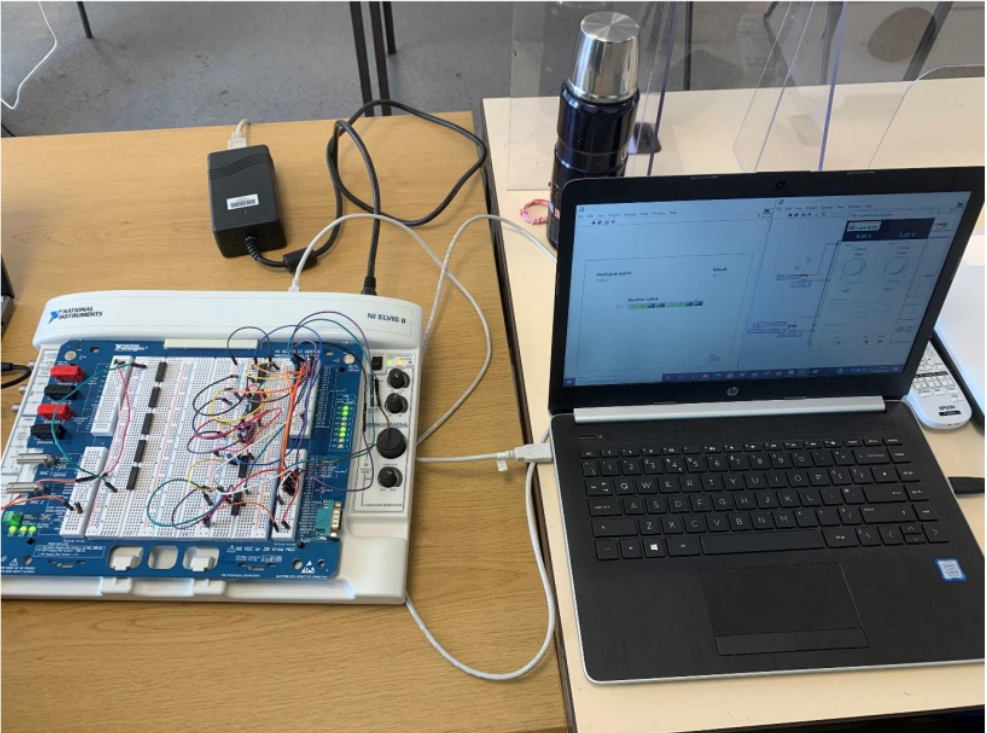


Figure 14: Analogue value of 1V showing a Boolean value of 51 displayed on NI Elvis II board

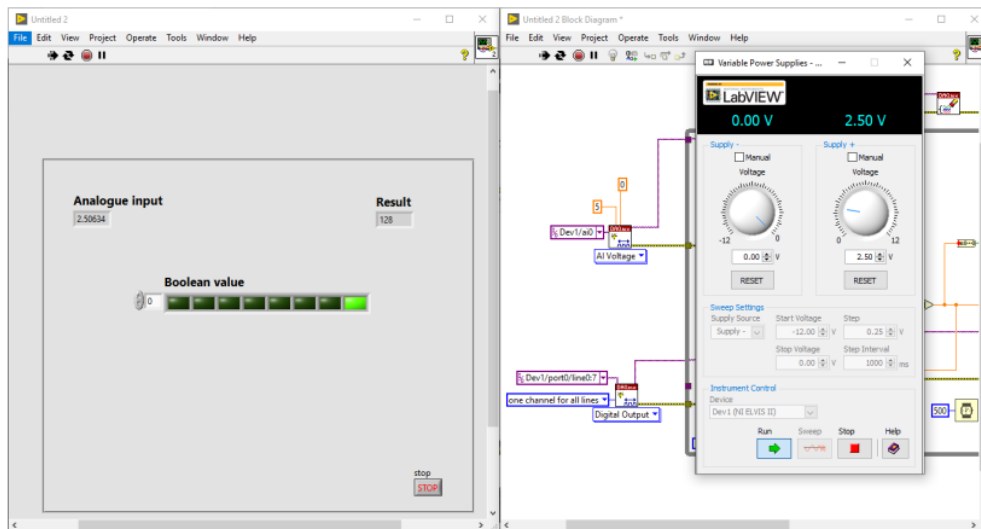


Figure 15: Analogue value of 2.5V showing a Boolean value of 128 displayed on panel.

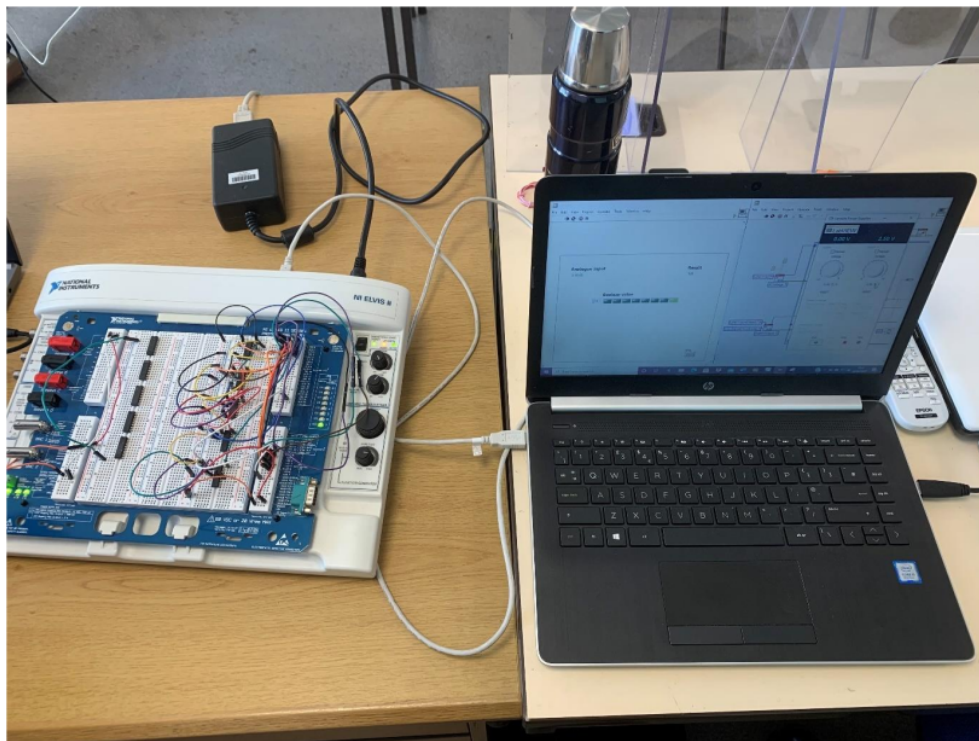


Figure 16: Analogue value of 2.5V showing a Boolean value of 128 displayed on NI Elvis II board

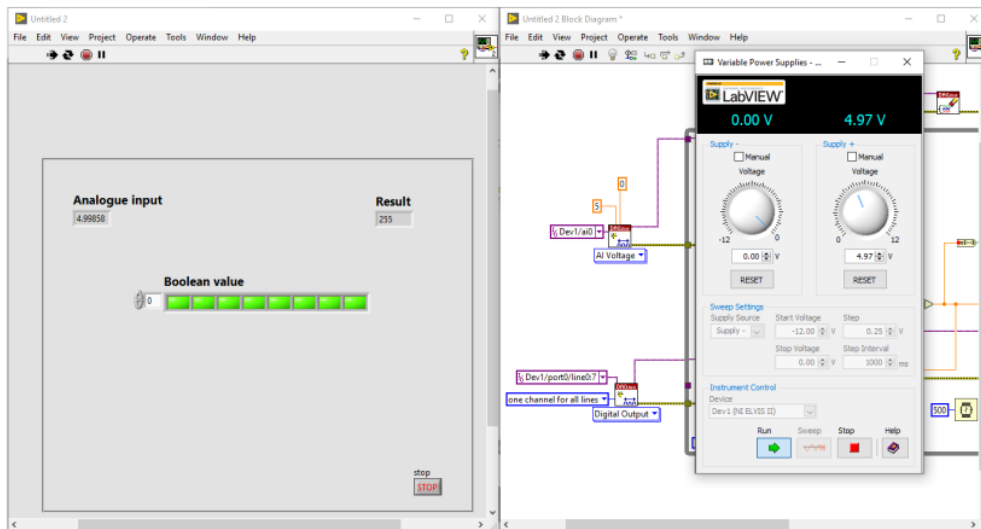


Figure 17: Analogue value of 5V showing a Boolean value of 255 displayed on panel.

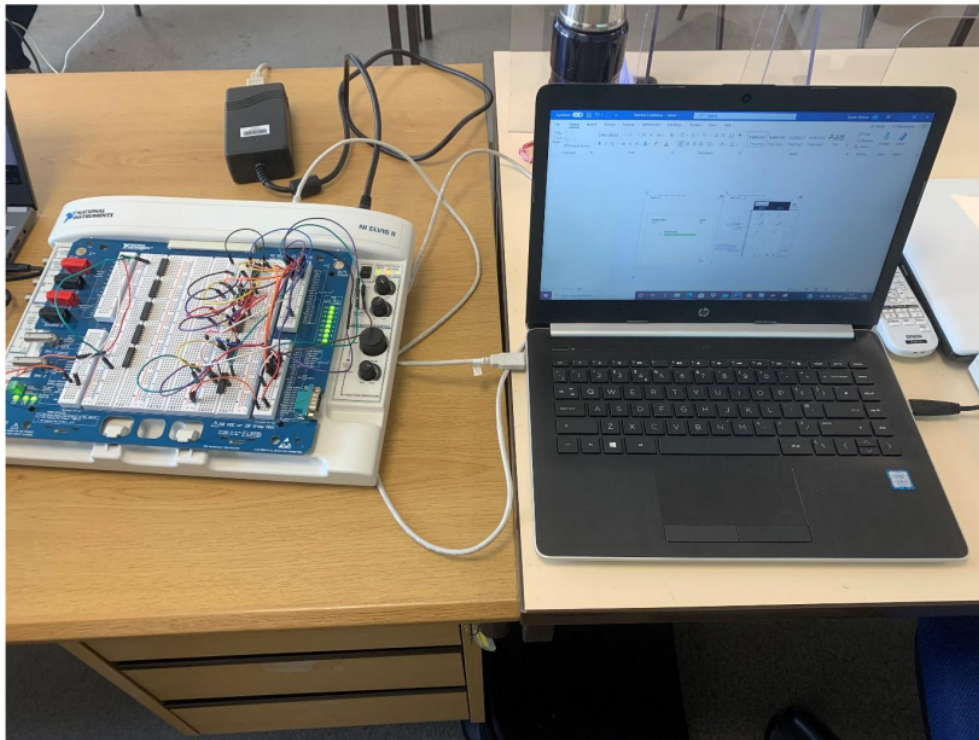


Figure 18: Analogue value of 5V showing a Boolean value of 255 displayed on NI Elvis II board

Good test and analysis of the 8 bit ADC with DAQ board and virtual input.

4. Nexys 4 DDR FPGA board

The Nexys 4 DDR FPGA programmable board is operated as a magnitude comparator in this portfolio and collaborates with the Labview program. The Nexys board is configured with the NI Elvis II board adhering to pin allocations shown in figure 19. The Nexys board component layout is illustrated in figure 20.

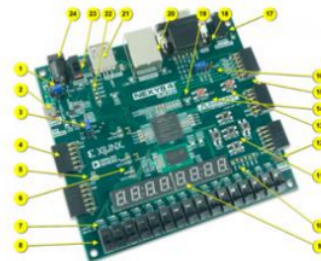


Figure 20. PMOD connectors, front view, as loaded on PCB.

Interface ports identified on
FPGA

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 19: Nexys 4 DDR Pin layout

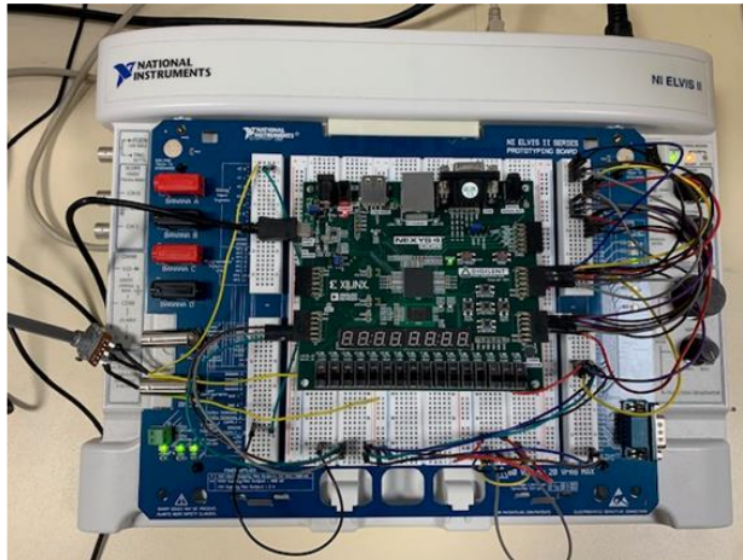


Callout	Component Description	Callout	Component Description
1	Power select jumper and battery header	13	FPGA configuration reset button
2	Shared UART/ JTAG USB port	14	CPU reset button (for soft cores)
3	External configuration jumper (SD / USB)	15	Analog signal Pmod port (XADC)
4	Pmod port(s)	16	Programming mode jumper
5	Microphone	17	Audio connector
6	Power supply test point(s)	18	VGA connector
7	LEDs (146)	19	FPGA programming done LED
8	Slide switches	20	Ethernet connector
9	Eight digit 7-seg display	21	USB host connector
10	JTAG port for (optional) external cable	22	PIC24 programming port (factory use)
11	Five pushbuttons	23	Power switch
12	Temperature sensor	24	Power jack

Figure 20: Nexys 4 DDR component layout

5. Magnitude comparator practical

The resulting connection linking the Nexys 4 DDR FPGA as the magnitude comparator and the NI Elvis II board is shown in figure 21. The potentiometer will enable the analogue signal to be varied to cycle through the magnitude comparator states.



*Good image showing
equipment setup*

Figure 21: Practical setup comprising NI Elvis II board with Nexys 4 DDR as magnitude comparator

6. LabVIEW

The LabVIEW program configured to function with magnitude comparator is shown in figure 22. Four formula blocks are present, and their function will be clarified in the passage.

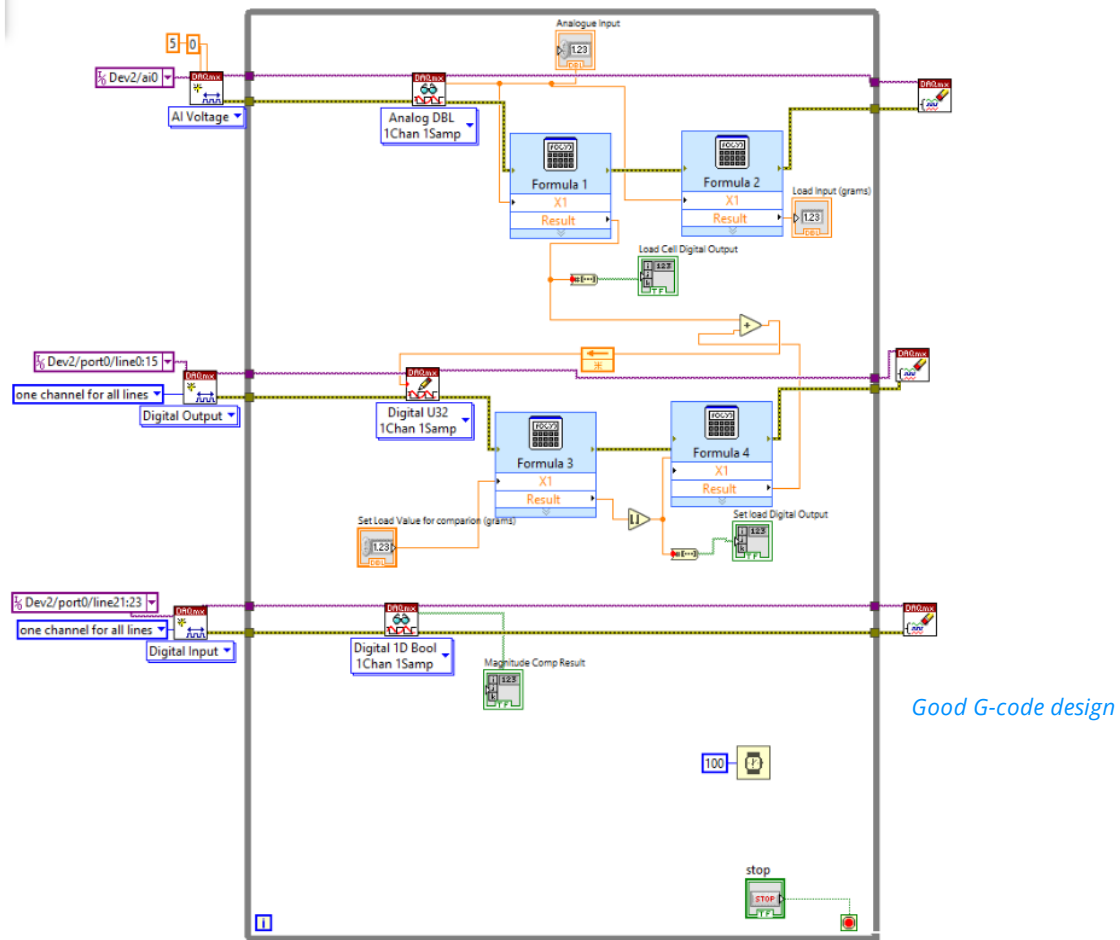


Figure 22: LabVIEW program

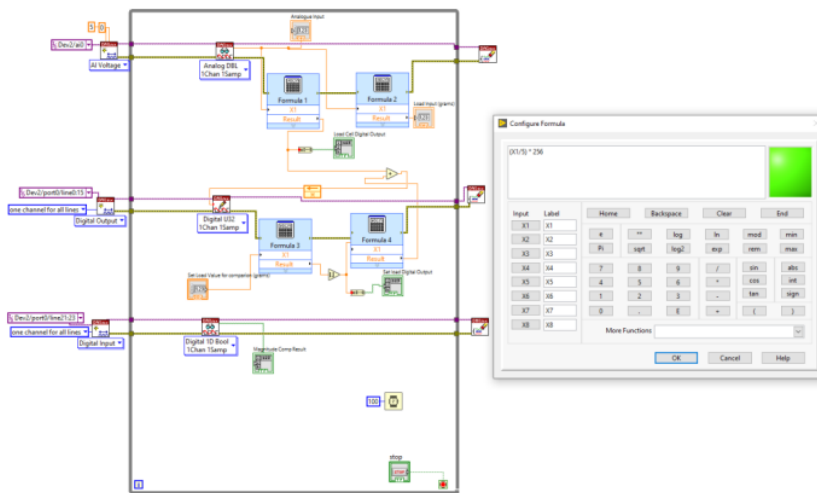


Figure 23: Formula block 1

Converting the analogue input into grams is performed by formula block 1 as shown in figure 23. The analogue input is divided by the maximum value of 5 and multiplied by 256 to produce the binary value. The output of this formula block is supplied to the digital output which signifies the gram value utilising the 8-bit word.

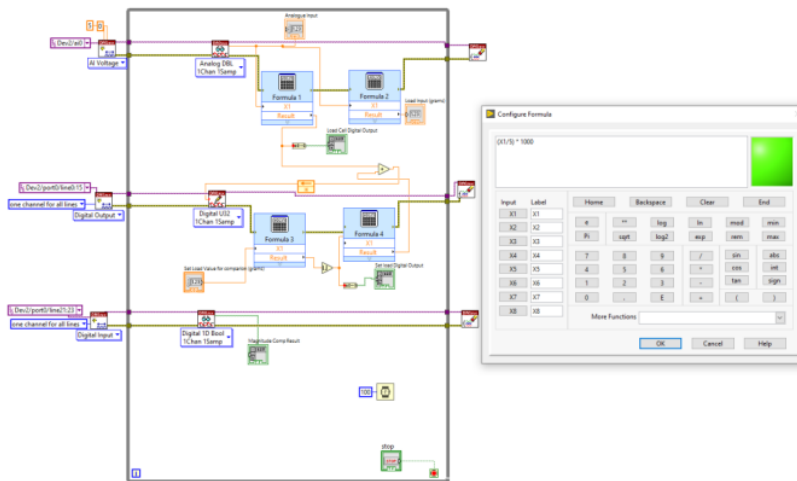


Figure 24: Formula block 2

Converting the analogue input to a gram value is performed by formula block 2. The analogue input value is divided by the maximum analogue input value and subsequently multiplied by 1000 to produce the gram measurement.

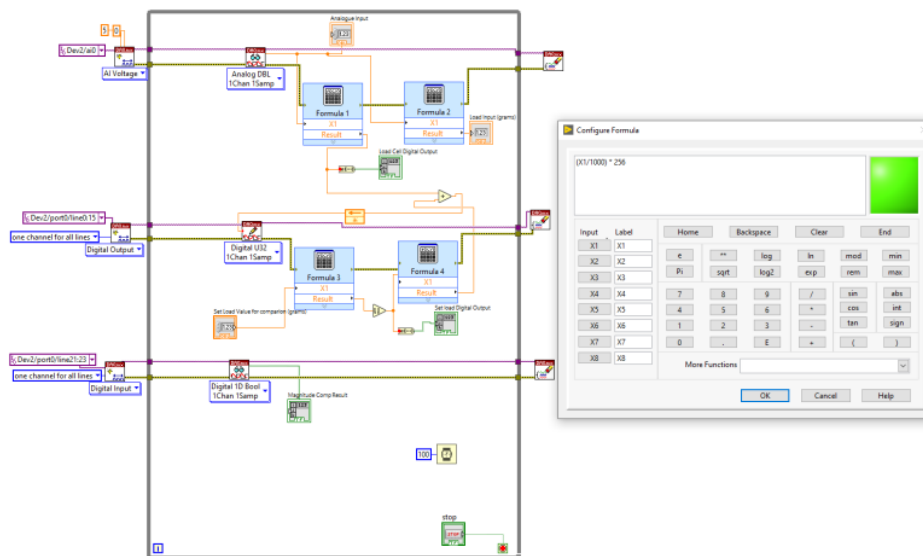


Figure 25: Formula block 3

Converting the set gram value into an 8-bit digital word output is performed by formula block 3 as shown in figure 25. The set gram value is divided by the max gram value and subsequently multiplied by the resolution.

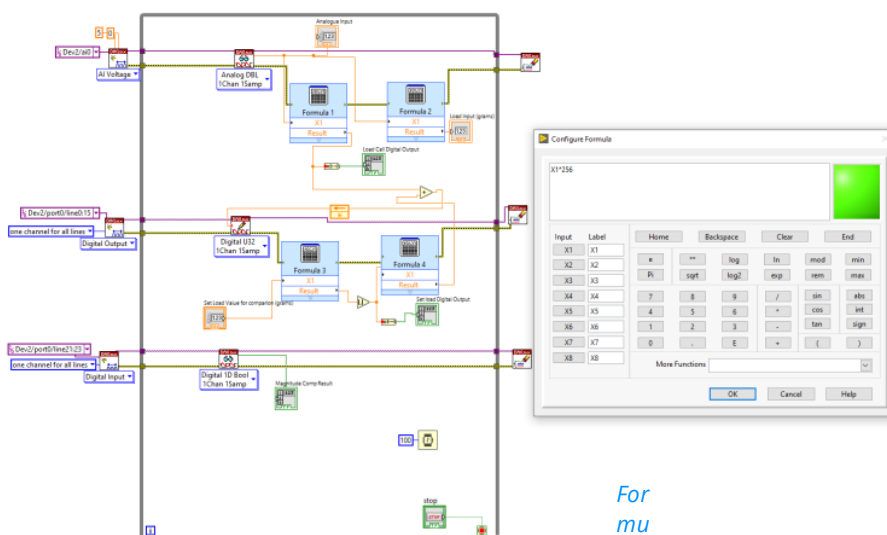


Figure 26: Formula block 4

For
mu
la
blo
ck
4
is
not
exp
lai
ne
d.
Thi
s
fun

7. Testing and Analysis

To evaluate the functionality and performance of the program a potentiometer was utilised to alter the analogue input which replicated the load cell. Three tests were performed on the prototype to assess the performance of the magnitude comparator. The set load was set at 300 and the load input was varied to provide equal to, less than and greater than. All three tests produced the expected results with all states shown in figures 27-29.

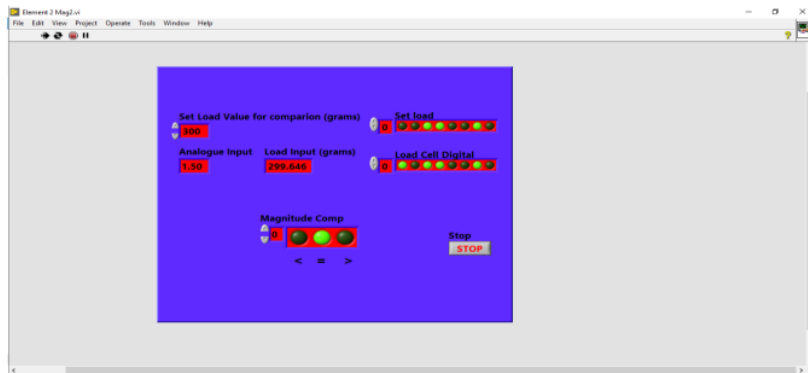


Figure 27: Equal to displayed on panel

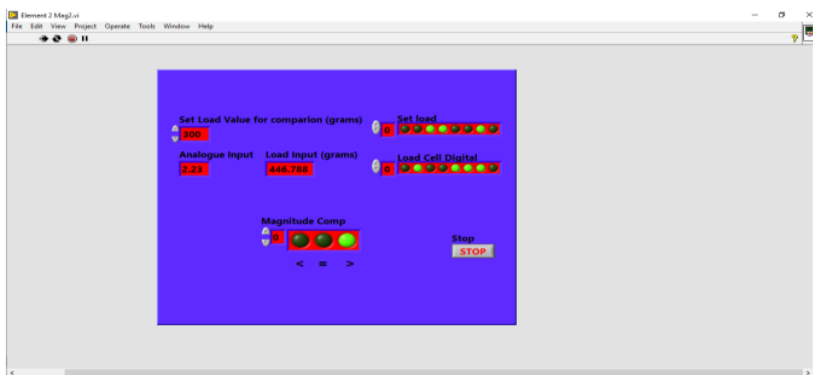


Figure 28: Less than displayed on panel

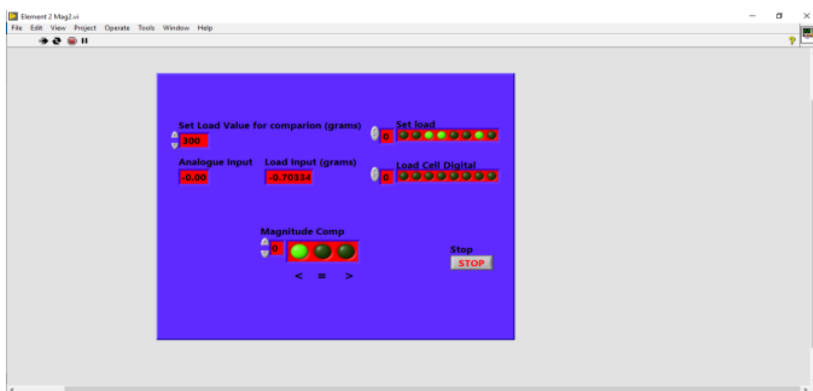


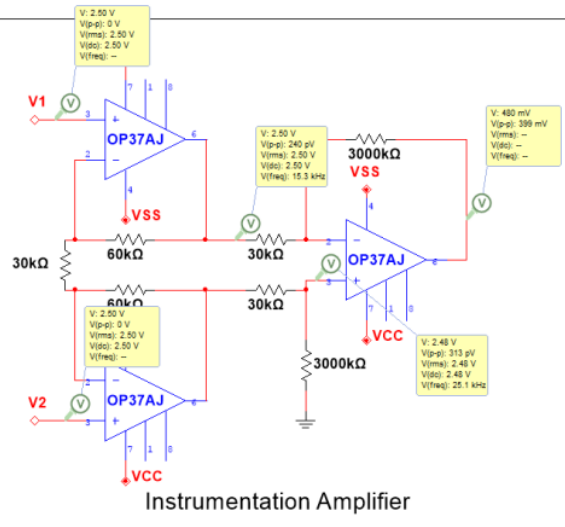
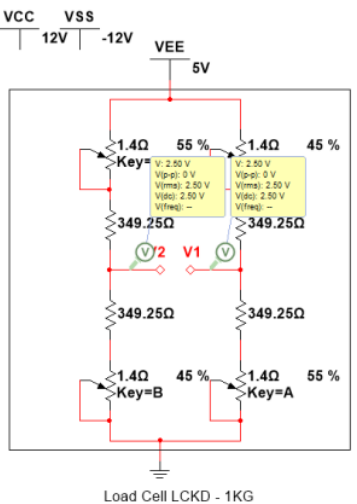
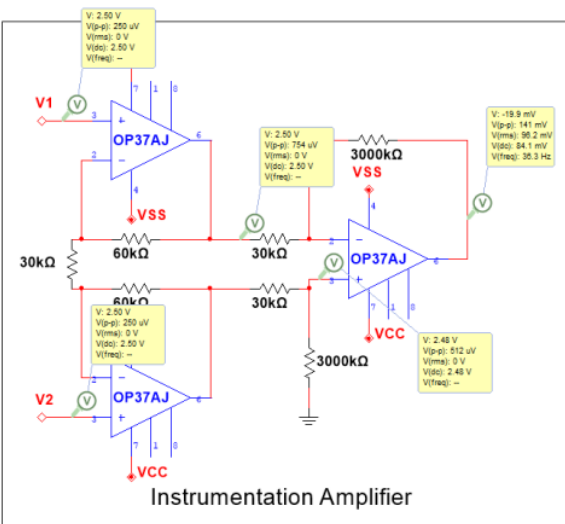
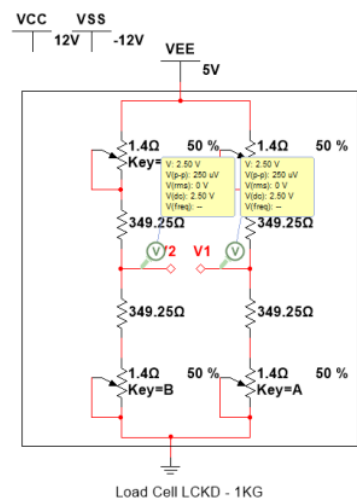
Figure 29: Greater than displayed on panel

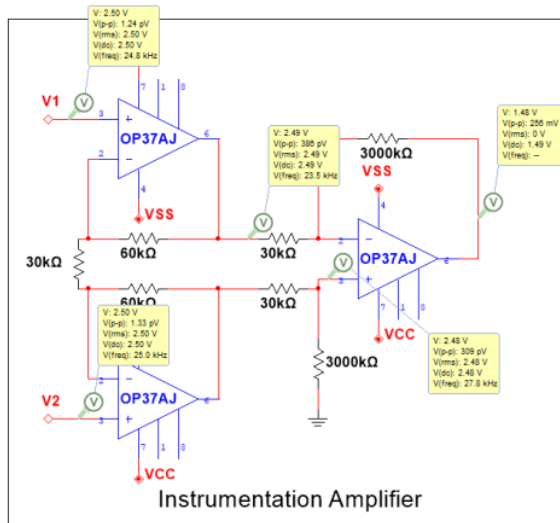
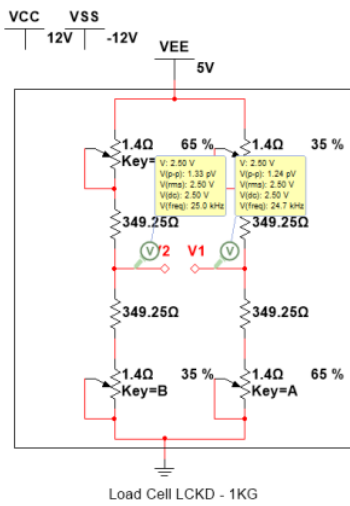
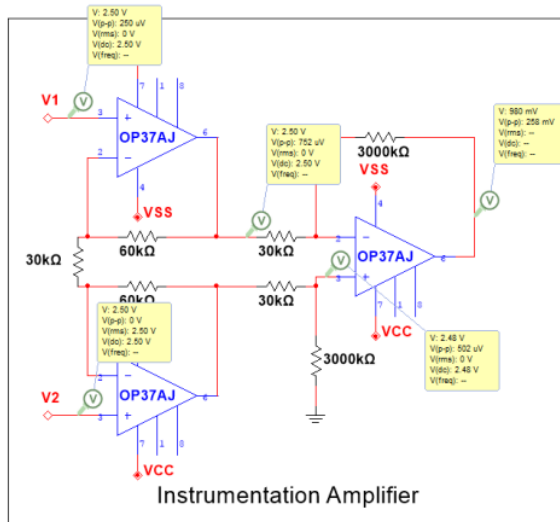
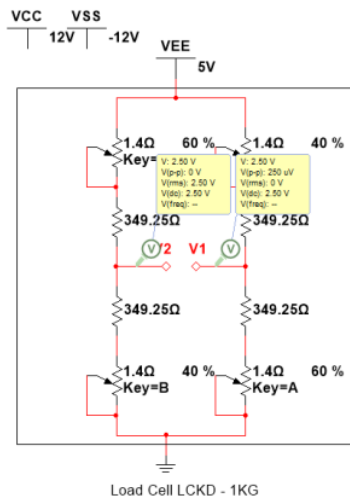
8. Evaluation

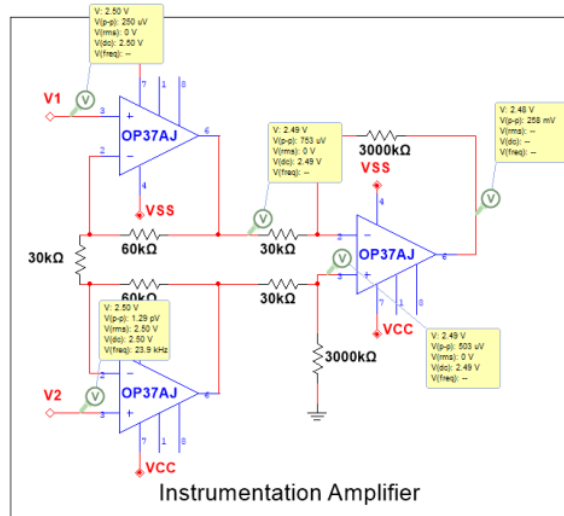
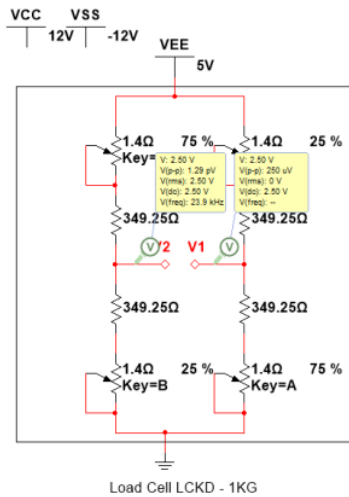
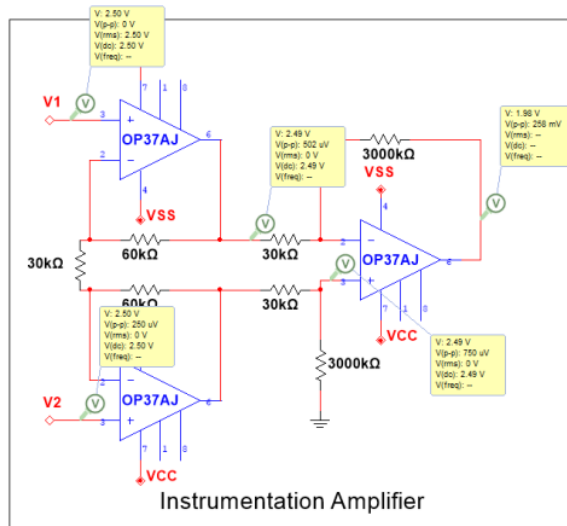
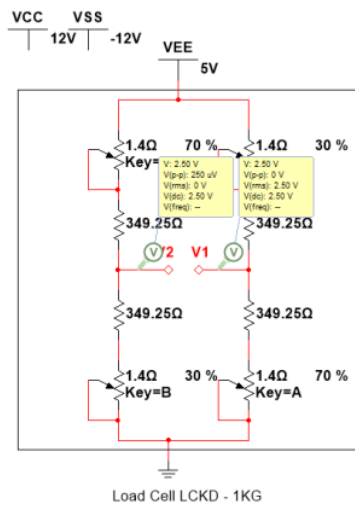
This element has demonstrated the concept of signal conditioning to obtain a digital binary value for an analogue signal. Increasing the ADC bit value increases the accuracy of the converted signal to produce a more faithful representation of the input signal. Converting the analogue signal to a representative digital value enables logical comparison to be performed through a magnitude comparator. This is the fundamental notion of all computer and control systems. The practical implementation of an ADC with a magnitude comparator was achieved through the collaboration of the Nexys 4 DDR FPGA and Labview Elvis II boards. This prototype exhibited a simple system but the fundamental concepts of all control systems.

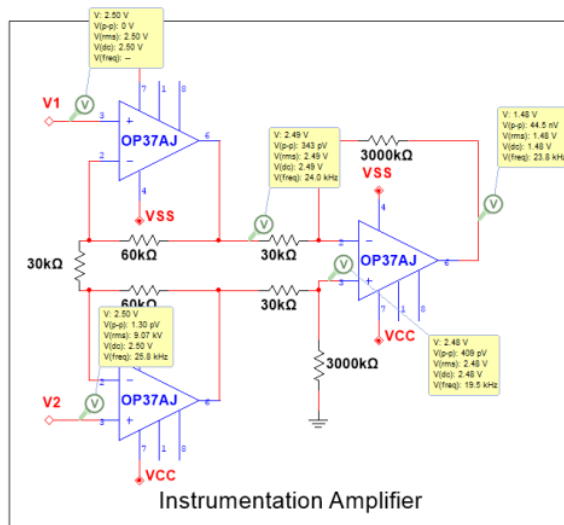
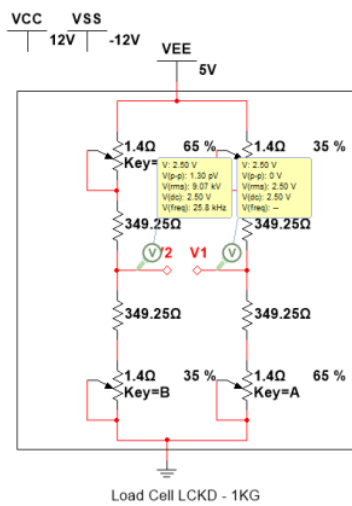
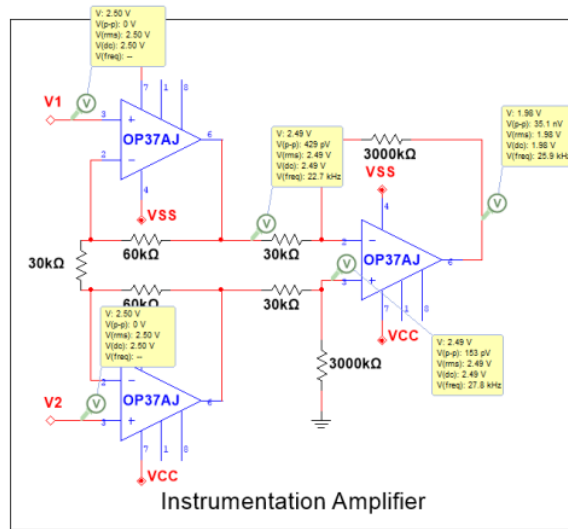
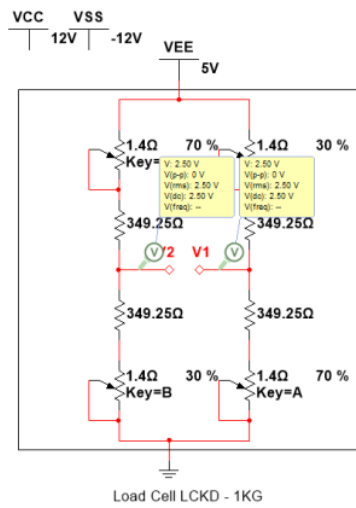
*Good
evaluation*

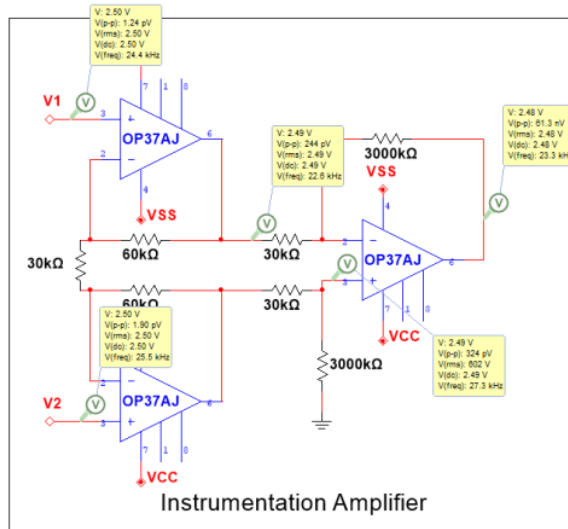
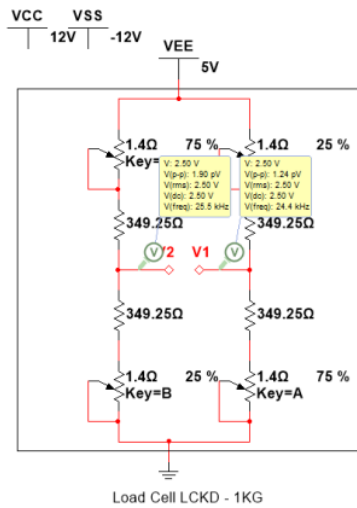
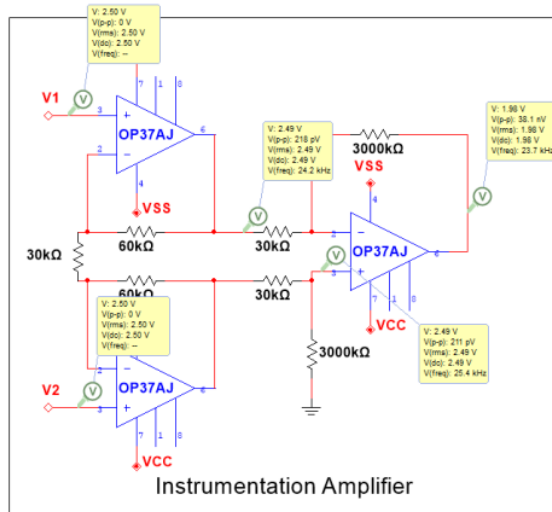
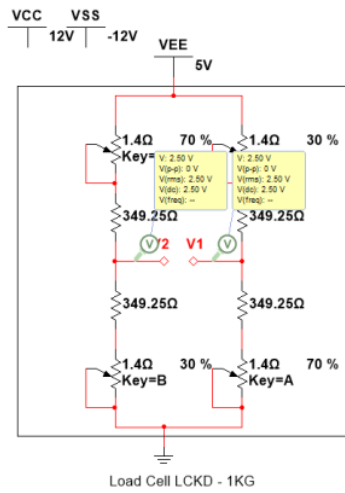
9.1 Appendix

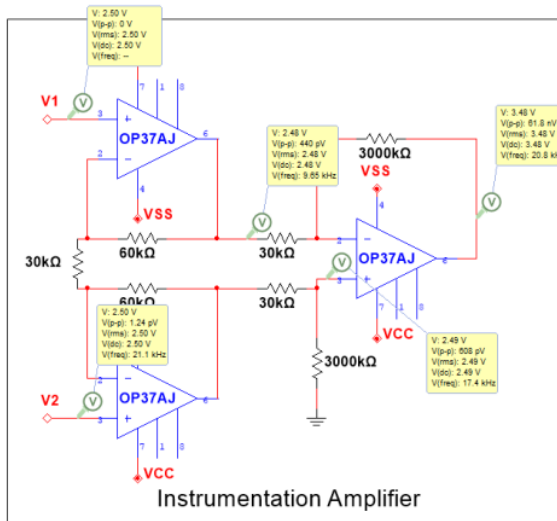
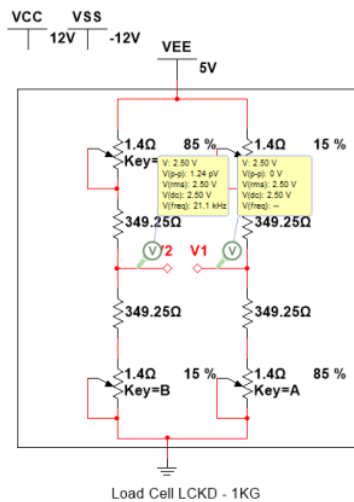
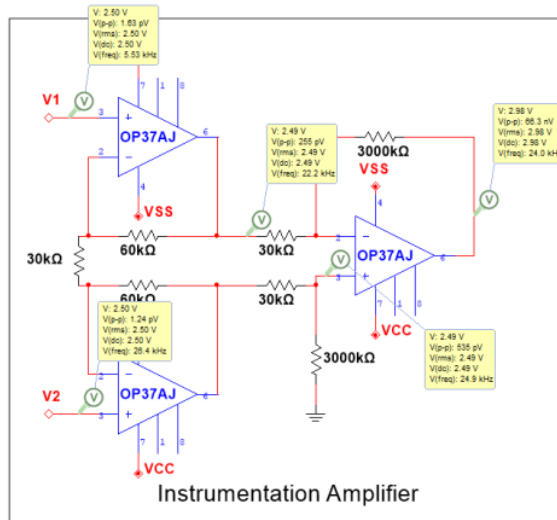
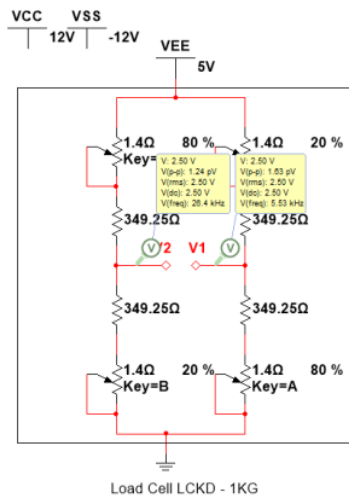


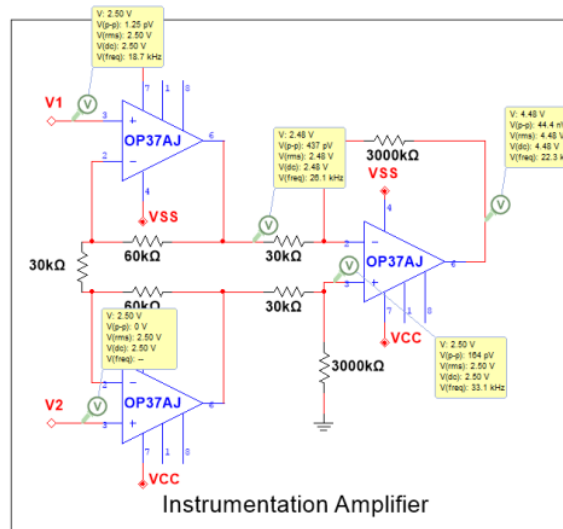
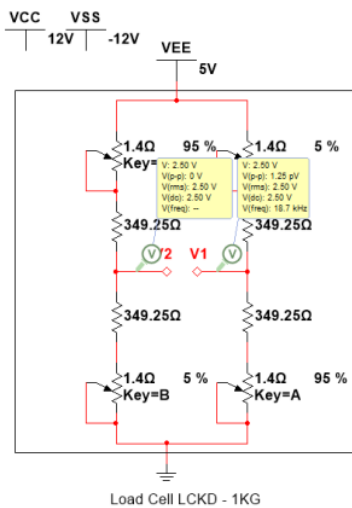
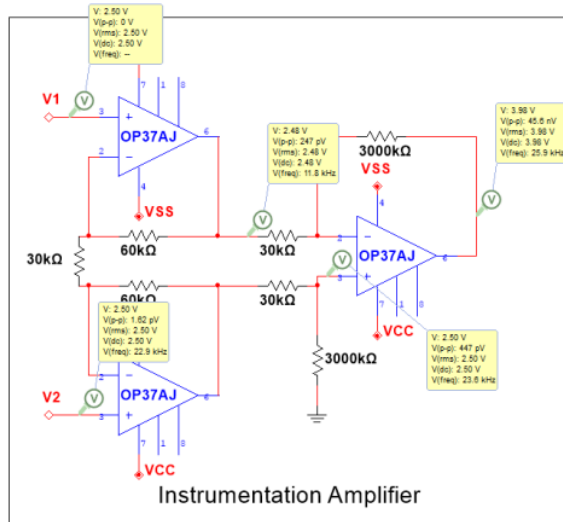
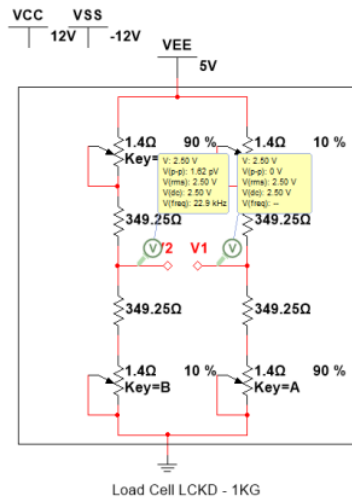












Full range of
instrument amplifier
test

Course: HND Electrical and Electronic Engineering
Module: SGEE5007 Virtual Instrumentation and Control Systems
Portfolio: Magnitude Comparator (Element 3)
Name : Gareth Wilkins
University number: 1809804
Date submitted: 20 May 2021

Contents

List of figures 3

1. Introduction 4

2. Design requirements 5

3. Testing and analysis 9

3.1 Desired value of 200g 9

3.2 Desired value of 500g 10

3.3 Desired value of 700g 11

4. Evaluation 12

List would
be more
informati
ve and
organised
with
subdivide
d
categorie
s with
more
details.

List of figures

Figure 1: LabVIEW program containing data logging elements.....	5
Figure 2: Data log information obtained from File1	6
Figure 3: Configuration of data log data for File1.....	6
Figure 4: Configuration of data log data for File2.....	7
Figure 5: Configuration of data log data for File3.....	7
Figure 6: Configuration of data log data for File4.....	8
Figure 7: Configuration of data log data for File5.....	8
Figure 8: 200g setpoint comparator results	9
Figure 9: 200g setpoint equal to state.....	9
Figure 10: 500g setpoint comparator results	10
Figure 11: 500g setpoint equal to state	10
Figure 12: 700g setpoint comparator results	11
Figure 13: 700g setpoint equal to state	11

[Go
od
list
of
fig
ur
es](#)

1. Introduction

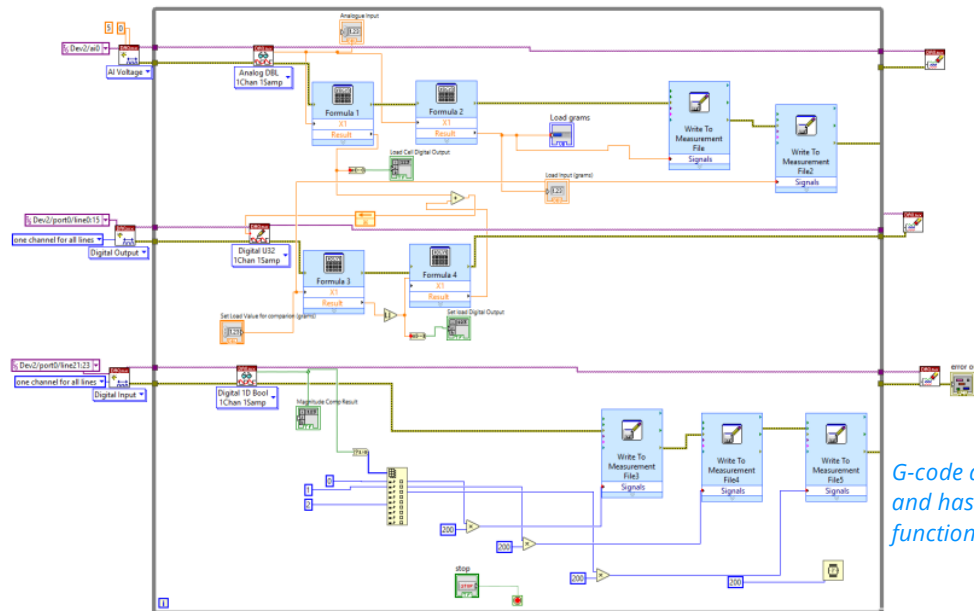
In this element the LabVIEW program that was produced in element 2 was further developed to provide data logging application. This element encompasses the magnitude comparator developed on the FPGA board in element 1 and the LabVIEW program developed in element 2. The datalogging enabled the comprehensive testing of the prototype and a graphical depiction of the states of the comparator in comparison to the raw analogue voltage. A conclusion was produced detailing the success of the magnitude comparator prototype utilising LabVIEW and how the accuracy of the system could have been enhanced by the implementation of an ADC of higher bit construction.

*This should
be in the
future
tense*

2. Design requirements

A Labview program was constructed to enable logging data established on the magnitude comparator. Figure 1 illustrates the program containing the additional file write blocks. These write files permit the logging of data over a specified time into a separate text file within a database. The logging is advantageous as the input signal obtained from the potentiometer and the output of the comparator can be traced beside the desired setpoint and a historic trend can be viewed. The write to measurement file block obtains the signal from their inputs and save the data to the instructed location. The file1 block acquires a signal from the potentiometer which simulates the load; file2 block is constant as the setpoint is its input.

Tetx file is just written to a text file not a database.



G-code design is correct and has all the correct function write blocks

Figure 1: LabVIEW program containing data logging elements

An extract from file1 is shown in figure 2 displaying the saved data within. Being the File1 data the increasing time is captured in the left column with the analogue voltage of the potentiometer variable in the right column. File3, file4 and file5 log the comparator states being less than, equal to and greater than. The configuration of the logged data location is shown in figures 3-7. The format required to save the files was text LVM file. File1 was the only block that required a time column for the x value and was unnecessary for the other blocks. The comparator output was scaled by 200 prior to logging of the data to produce a presentable graphical depiction for evaluation.

Explanation is correct

```

text1 - Notepad
File Edit Format View Help
LabVIEW Measurement
Writer_Version 2
Reader_Version 2
Separator Tab
Decimal_Separator .
Multi_Headings No
X_Columns One
Time_Pref Absolute
Operator Gareth
Date 2021/04/29
Time 11:36:20.4970297813415527344
***End_of_Header***

Channels 1
Samples 1
Date 2021/04/29
Time 11:36:20.4970297813415527344
X_Dimension Time
X0 0.000000000000000E+0
Delta_X 1.000000
***End_of_Header***
X_Value Untitled Comment
0.000000 858.223844
0.299254 855.236532
0.600914 859.733561
0.902974 857.228073
1.202471 857.806263
1.502768 858.320209

```

Figure 2: Data log information obtained from File1

Figure 3: Configuration of data log data for File1

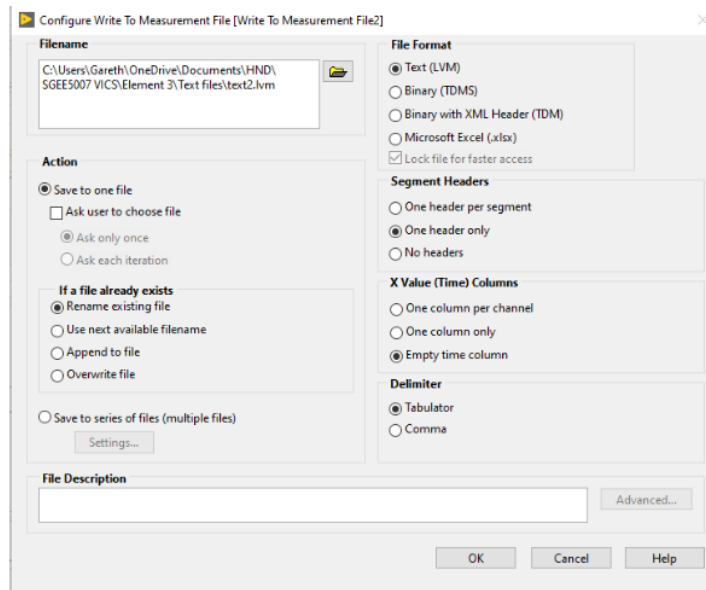


Figure 4: Configuration of data log data for File2

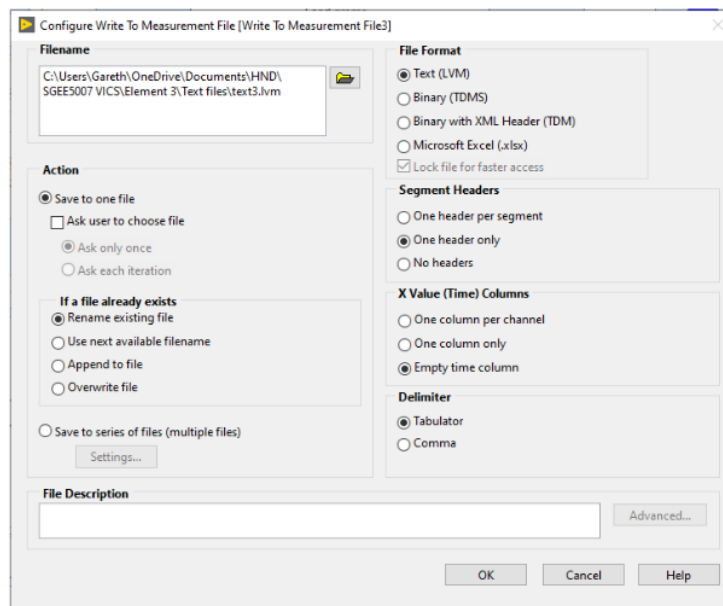


Figure 5: Configuration of data log data for File3

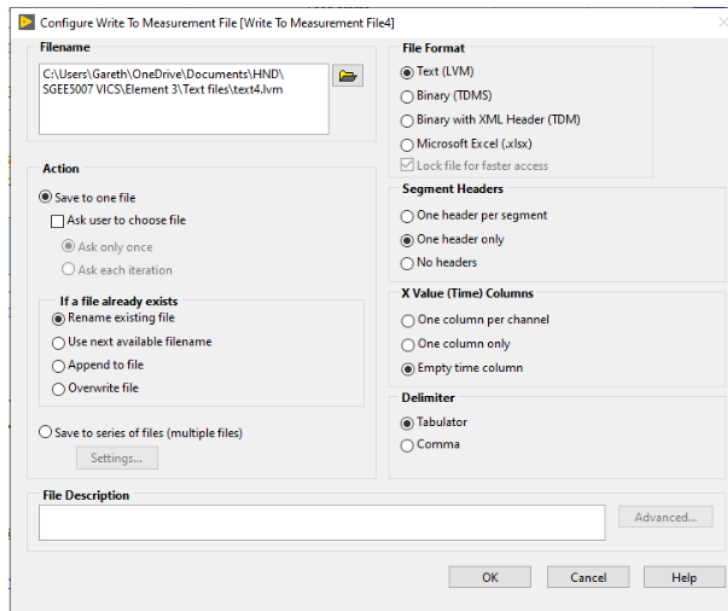


Figure 6: Configuration of data log data for File4

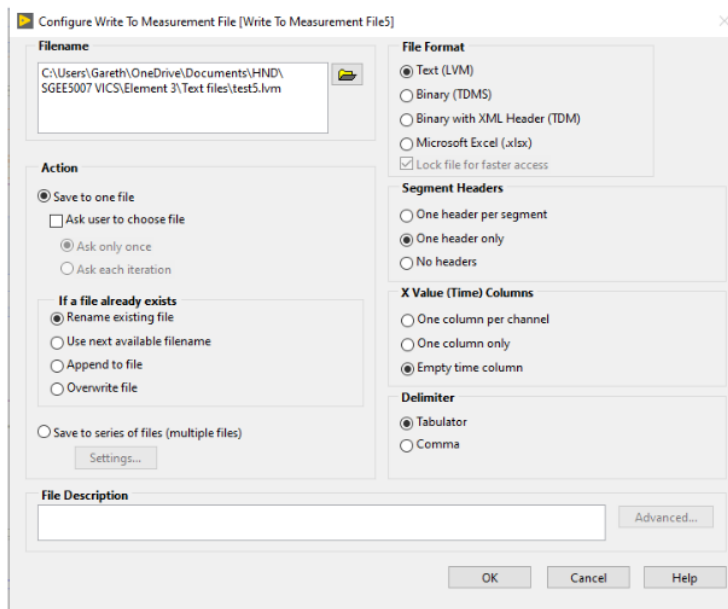


Figure 7: Configuration of data log data for File5

3. Testing and analysis

To test functionality and correct operation the potentiometer was varied to obtain different analogue inputs values which were compared with the setpoint by the magnitude comparator. Three setpoint were selected being 200, 500 and 700.

3.1 Desired value of 200g

The data log was obtained, and a graphical depiction of the states was produced and shown in figure 8. The graph illustrates that the analogue input was increased from zero to below 200 with the yellow trend showing less than from the setpoint of 200. Increasing the analogue voltage to 199.979 produced the equal to state indicated by the blue trend. Increasing the analogue voltage further produced the greater than state shown by the green trend. The equal to panel view is shown in figure 9 where the set load and load cell are the same.

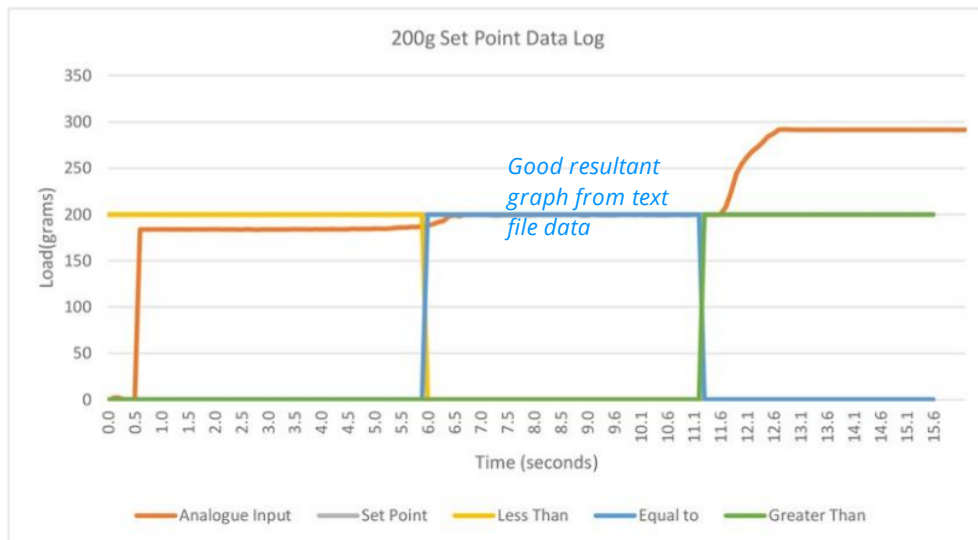


Figure 8: 200g setpoint comparator results

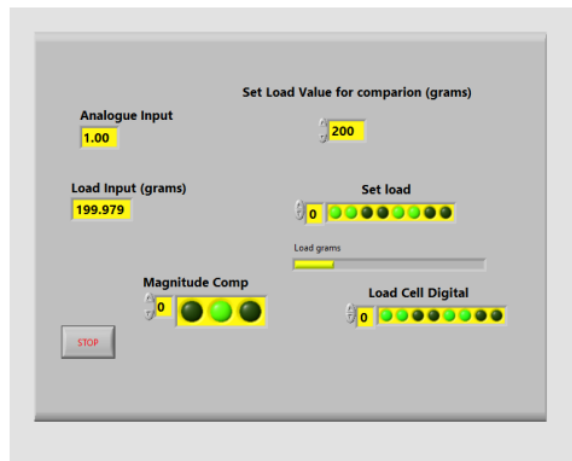


Figure 9: 200g setpoint equal to state

Good front panel design

3.2 Desired value of 500g

Again, the data was obtained following the same method as previously discussed and the graphical illustration of the states are shown in figure 10. The analogue input was set below the desired value and the less than state was obtained from the magnitude comparator illustrated by the yellow trend. Increasing the analogue voltage to the desired set point of 500 the equal to state was acquired as shown by the blue trend. Increasing the analogue voltage beyond the desired value of 500 resulted in the greater than state illustrated by the green trend. The equal to panel view where the set load and load cell are of the same value as shown in figure 11.

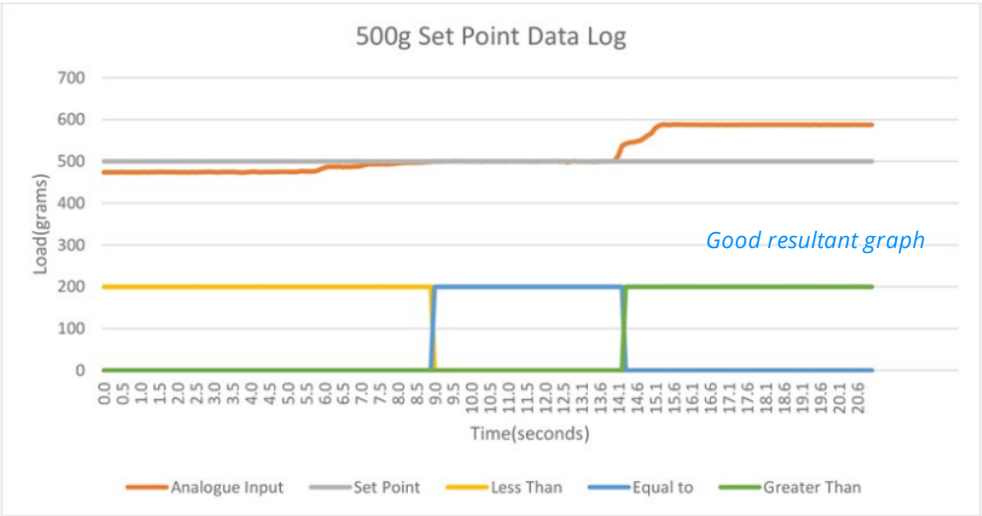


Figure 10: 500g setpoint comparator results

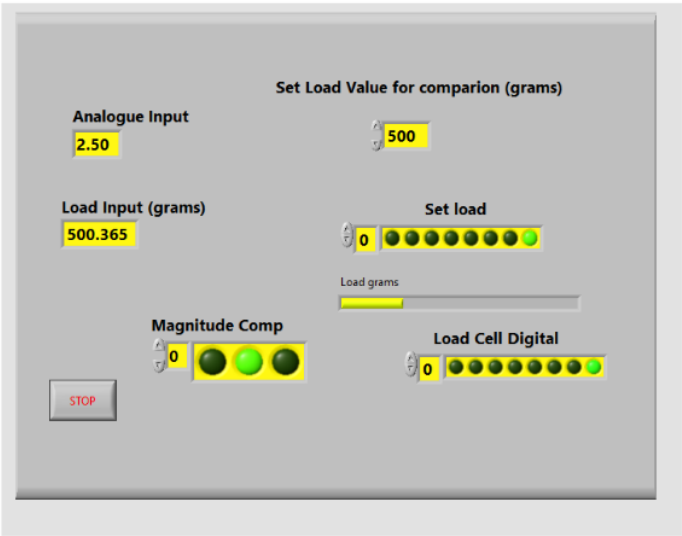


Figure 11: 500g setpoint equal to state

3.3 Desired value of 700g

The data was obtained following the same method as previously discussed and the graphical illustration of the states are shown in figure 12. The analogue voltage was set below the desired value of 700 producing the less than state from the magnitude comparator shown by the yellow trend. The analogue input voltage was increased to 700 which produced the equal to state from the magnitude comparator shown by the blue trend. By increasing the analogue voltage beyond the 700 setpoint the greater than state was produced as illustrated by the green trend. The equal to panel view where the set load and load cell are the same is shown in figure 13.

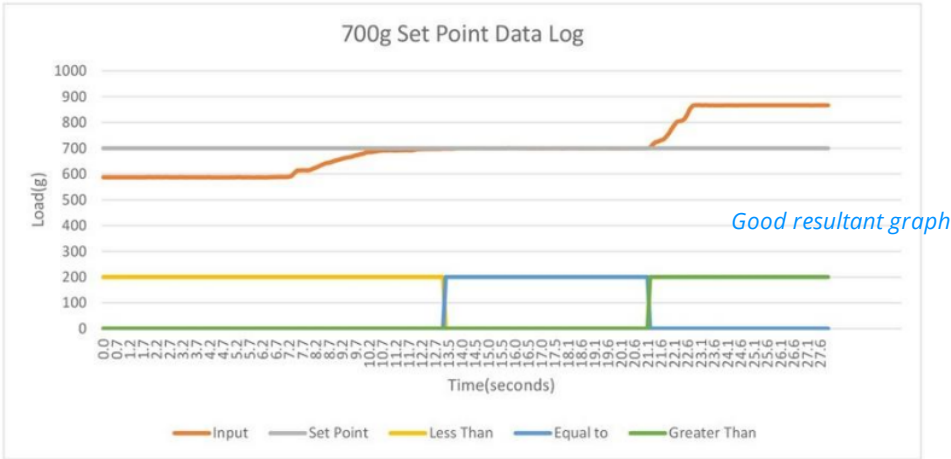


Figure 12: 700g setpoint comparator results

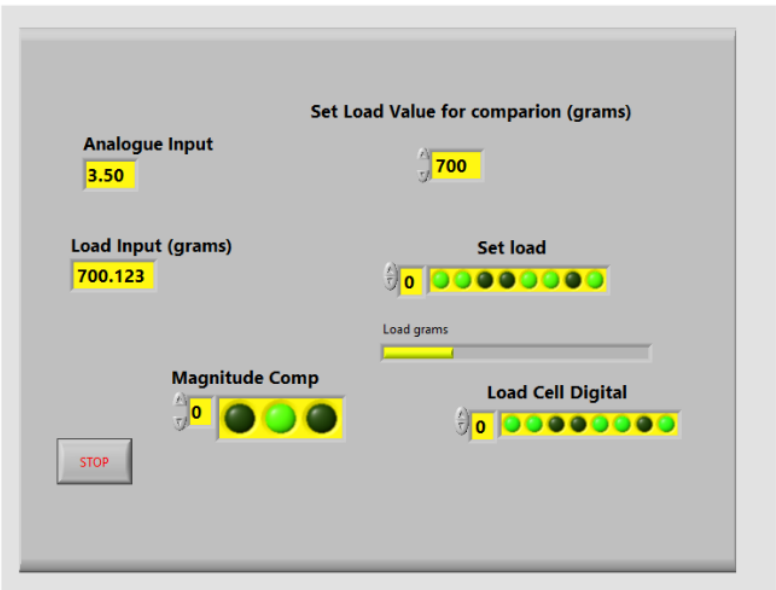


Figure 13: 700g setpoint equal to state

4. Evaluation

This element practically demonstrated the conversion of an analogue signal to digital by means of an ADC and comparing this digital data with a magnitude comparator which was developed in element 1 of this portfolio. It was found that varying the analogue data achieved the three states set in the magnitude comparator being equal to, less than and greater than. It was observed that the magnitude comparator would signal equal to state when the setpoint and input value were similar but not exact. The prototype would benefit from using an ADC of higher bit value. As the ADC was an 8-bit model in the program the raw data was exposed to quantization error which produced suboptimal accuracy due to the lower resolution. As this was a prototype the accuracy was not of great importance however with application requiring a higher degree of accuracy an enhanced ADC bit level would be required.

Good evaluation

Portfolio

GRADEMARK REPORT

FINAL GRADE

GENERAL COMMENTS

74/100

Instructor

PAGE 1

PAGE 2

Text Comment. Good table of contents for element 1

PAGE 3

Text Comment. Good list of figures

PAGE 4

Text Comment. Good use of tables

PAGE 5

Text Comment. Good informative introduction

PAGE 6

Text Comment. A good explanation of the structure of the VHDL program.

PAGE 7

Text Comment. This is a good point about the purpose and benefit if IP's in Vivado. Good supporting images.

PAGE 8

Text Comment. The NI DAQmx function has been identified.

PAGE 9

Text Comment. Components of the DAQmx used in the LabVIEW program have been explained with a good working example.

Text Comment. A comprehensive explanation of the magnitude comparator and a good example of a practical design circuit

Text Comment. Good explanation with good source

Text Comment. The explanation here is very good and there are points that demonstrate a good understanding

Text Comment. The bridge circuit is correct and these are commonly supplied with a sensor. However this would probably have an instrument amplifier as a stage in the signal condition process.

Text Comment. The process for the creation of the 4 bit adder is explained and clearly understood

Text Comment. Layout of code is good

Text Comment. The initial hierarchy for the system starts here and is identified

Text Comment. 4 bit adder created from 4 full adders which are in turn made up of 2 half adders is explained here

Text Comment. Good layout of code for the 4 bit adder with the architecture made up 4 full adders with sum and carry variables set.

PAGE 21

Text Comment. It should be mentioned that the VHDL code here was created by the Multisim circuit design PLD

PAGE 22

Text Comment. The IP design integration shows the connections of the 2 IP's which enables a more complex design of SOC systems.

PAGE 23

Text Comment. Good design of the 8 bit comparator using PLD

PAGE 24

Text Comment. Good result and good use of the virtual bus method for testing arrays

Text Comment. Good use of tables

PAGE 25

Text Comment. This is the test bench data for the comparator not the adder

PAGE 26

Text Comment. Good test showing all 3 states

Text Comment. Good use of tables

PAGE 27

Text Comment. Correct data for test bench

PAGE 28

PAGE 29

PAGE 30

Text Comment. Very good set of test results

Text Comment. Good set of results from Msim 8 bit comparator

Text Comment. Good results for the 8 bit comparator

Text Comment. Good evaluation which relates to the next stage in element 2.

Text Comment. Good number of relevant sources

Text Comment. Good table of contents

Text Comment. Good list of figures

Text Comment. Good use of table for test analysis

Text Comment. Use the introduction as the future tense for the element

Text Comment. Good example of the op amp at unity gain

Text Comment. Gain of amplifier is identified

Text Comment. Good table of values

Text Comment. Good linear graph from results

Text Comment. Good design method for 10 bit ADC with max value held

Text Comment. Good design of 12 bit ADC with max value held

Text Comment. Good test and analysis of the 8 bit ADC with DAQ board and virtual input.

Text Comment. Interface ports identified on FPGA

Text Comment. Good image showing equipment setup

Text Comment. Good G-code design

Text Comment. Formula block 4 is not explained. This function is to multiply the first 8 bits by 256 which converts it to 16 bit. The next 8 bits are added using the + block.

Text Comment. Good tests shown on front panel

Text Comment. Good evaluation

Text Comment. Full range of instrument amplifier test

Text Comment. List would be more informative and organised with subdivided categories with more details.

Text Comment. Good list of figures

Text Comment. This should be in the future tense

Text Comment. Tetxt file is just written to a text file not a database.

Text Comment. G-code design is correct and has all the correct function write blocks

Text Comment. Explanation is correct

PAGE 75

PAGE 76

PAGE 77

Text Comment. Text file configurations are correct

PAGE 78

Text Comment. Good resultant graph from text file data

Text Comment. Good front panel design

PAGE 79

Text Comment. Good resultant graph

Text Comment. Good test on front panel

PAGE 80

Text Comment. Good resultant graph

Text Comment. Good front panel test

PAGE 81

Text Comment. Good evaluation

GRADING FORM: CSG-RUBIC

GARETH WILKINS

0

RESEARCH-INFORMED LITERATURE



Research section has covered all requirements in good detail. Good relevant sources used.

8/10

KNOWLEDGE AND UNDERSTANDING



E1-Design and use of magnitude comparator with digital inputs has been created with good explanations. There is a minor error with test data for the 4 bit adder where the 4 bit comparator test data is referred to. Test methods used are good and followed a logical sequence. 7.5/10 E2- The LabVIEW design for the analogue load cell input is explained and the design is a good effort. The 16 bit word operation for the fpga is not explained. The shift right and add parts are missing and this is the key to how the data bits are organised for the fpga. The design is operational but needs to be explained further. 6.5/10 E3-Dat logging has ben fully explained and a good knowledge of this section is demonstrated. 8/10

22/30

ANALYSIS



E1-Test and results are very comprehensive . The evaluation covers the main issues. Apart from an

18/25

error with incorrect data for test bench for 4 bit adder. 3/5 E2-The results from LabVIEW are comprehensive and detailed. The evaluation needs to be in future tense.. 7/10 E3-Results from the data logging section are very detailed. The evaluation covers good points but needs to be in future tense. 8/10

PRACTICAL APPLICATION AND DEP.



E1-Prototypes for the comparator have been designed using logical methods. PLD design methods used were successfully implemented using the FPGA development board. 3.5/5 E2-Design and build of LabVIEW interface for fpga followed a good development sequence. The development sections were tested at specific stages to prove design prototypes. The data was shifted right and added successfully. 3.5/5 E3-Data logging using LabVIEW methods has been designed and tested for operational requirements. 8/10

15/20

SKILLS FOR PROFESSIONAL PRAC.



Report was good

11/15