Rapid Controller Prototyping

Introduction to LabVIEW data acquisition Laboratory

Overview

This laboratory is split into a number of exercises aimed at providing an introduction to LabVIEW programming and data acquisition, based around the 3-degrees of freedom, (3-DOF), helicopter training system.

The first two exercises are provided to lead you through the basic steps of building a simple LabVIEW program, and to introduce you to a number of the aspects of the LabVIEW programming environment.

The remaining exercises will guide you through the development of a data acquisition program, based around the plug in learning board and the 3-DOF control module. During these exercises, you use the NI DAQmx libraries to develop a data acquisition program to achieve:

- Multichannel analogue input
- Multichannel digital input
- Multichannel digital output
- Single channel analogue output

After you have completed each exercise, please bring your work to the attention of the laboratory demonstrators, so your progress can be assessed.

Before attending these laboratory sessions, you should to read the introductory LabVIEW reference document to provide you with some background understanding of the LabVIEW environment and programming elements.

Exercise 1 – Create a Basic LabVIEW VI

Goal: This exercise will guide you through the first stages of building a LabVIEW program, and demonstrate the use of a while loop to allow the code to iteratively operate

Part 1

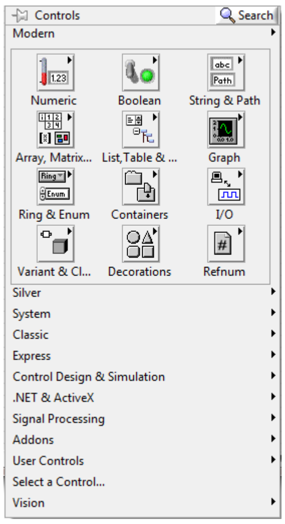
Description: This part of the exercise will guide you through the process of creating a LabVIEW VI, and adding a while loop with a termination button

Procedure:

1. Launch LabVIEW
2. In the getting started window, select **File»New VI** to create a new VI.

   You will see two windows appear; the first will be a grey window with a grid pattern. This is the Front Panel windows and serves as the graphical user interface, (GUI). The second window, usually appearing behind the Front Panel, is the Block Diagram where the LabVIEW graphical code is placed.

3. Open the Block Diagram by selecting **Window»Show Block Diagram**, or by pressing<Ctrl-E>.
4. Switch back to the front panel by selecting **Window»Show Front Panel**, or by pressing<Ctrl-E>.
5. Open the controls palette by right-clicking on the block diagram. This can be pinned to the desktop, when the block diagram is in focus, by clicking on the pin symbol in the top left of the functions palette.



6. Select a vertical point slider by navigating to **Modern»Numeric»Vertical Point Slider**, from the controls palette, and add this to the front panel.
7. Select a tank by navigating to **Modern»Numeric»Tank**, and add this to the front panel.
8. Switch back to the block diagram, (see above for method)

   You will notice that two icons have appeared on the block diagram. The orange icon is linked to the vertical point slider, and the blue icon is linked to the tank. These icons are intrinsically linked to the controls on the front panel. If they are deleted from either window they will be removed from the other, furthermore, if they are relabelled in one window they are relabelled in the other.

9. Switch to the front panel view.
10. Hold the mouse over the vertical point slider icon. You will see a wiring point appear in the right side of the icon. When the cursor is held over the wiring port, it will automatically switch to the wiring tool. Click in the wiring port and connect it to the wiring point on the tank.

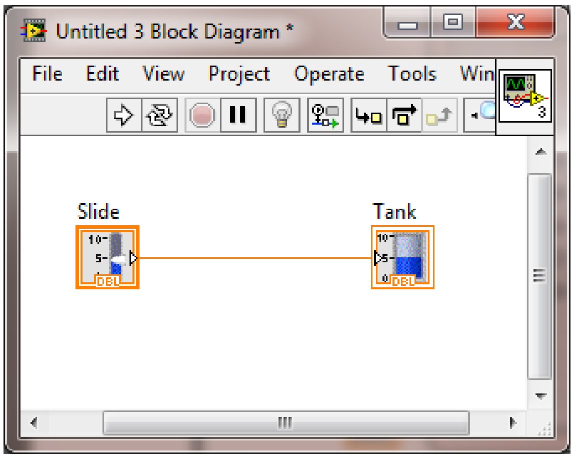
    If we move the pointer on the slider, (move the mouse over the slider pointer until the finger cursor appears, then drag the pointer to a new location), then we will see that the tank level does not change.

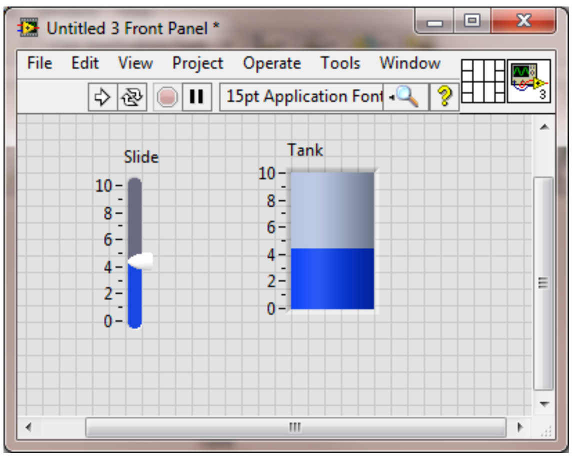
11. Run the program by pressing the run program button, (the triangle icon in the tool bar), or press <Ctrl-R> or **Operate»Run**.

    When we run the program we can see that the front panel window grid disappears momentarily and returns again. (When the grid disappears this indicates the program in running.)

    In this form the code, (slider and tank), will only run through once, because there is nothing holding the flow of control. Once executed, the IV will automatically terminate.

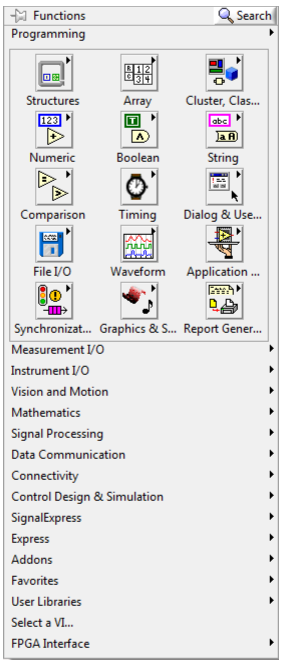
    To hold the flow of control we need to add a while loop, to make the slider-tank code execute over and over again.


Untitled 3 Block Diagram *
File Edit View Project Operate Tools Win
Slide
Tank
DBL
DBL



Part 2
Description: This part of the exercise will guide you through the process of adding a while loop with a termination button

Procedure:
1. Switch to the Block Diagram.
2. Open the functions palette by right-clicking on the block diagram. This can be pinned to the desktop, when the block diagram is in focus, by clicking on the pin symbol in the top left of the functions palette.



3. Select a while loop by navigating to **Programming»Structures»While Loop** and placing the while loop using the cursor to drag a selection rectangle around the section of the block diagram you want to repeat. When you release the mouse button, a loop boundary encloses the section you selected.
4. The conditional terminal in the bottom right corner on the while loop needs to be wired to a Boolean control to terminate the while loop. For this exercise we will wire it to a stop button on the front panel.
    a. Right clock on the conditional terminal in the bottom right of the while loop.
    b. Select **Create»Control**. You will see a STOP icon on the block diagram, and a STOP button will have appeared on the front panel. The icon and the button are automatically linked. Pressing the button when the while loop is executing will cause the while loop to terminate.

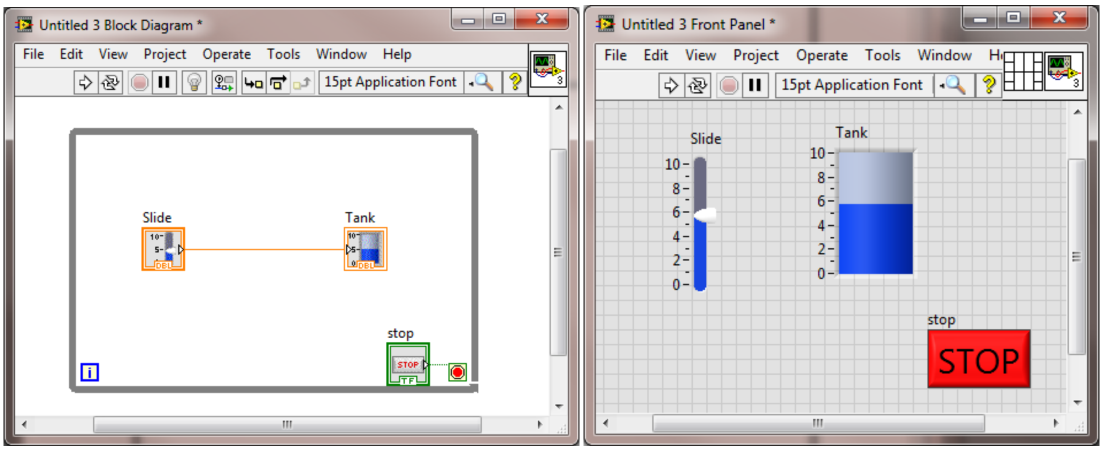
5. Switch to the Front Panel
6. Select the Stop Button by dragging the cursor over the button.
7. Resize the button to about 6-grids wide and 4-grids high.
8. Double click on the 'STOP' Text to select it.

   <Ctrl+> can be used to enlarge the text, and <Ctrl-> can be used to decrease the size.

9. Right click on the STOP button and select **Properties**.
10. In the **Appearance** Tab, click on the coloured box below the 'Text color' label, and select black
11. Click on the colour box beside the 'Off' label and select red
12. Click on the colour box beside the 'On' label and select an orange hue from the rainbow coloured bar
13. Run the program

When you run the program now, you will see that it does not automatically terminate. If the slider pointer is moved, it can be seen that the value immediately appears on the tank level.

14. Stop the program by pressing the stop button you have added. The front panel grid should now appear indicating that the VI has stopped executing.



Part 3
Description: This part of the exercise we will add a numeric display linked the slider and the tank

Procedure:
1. On the front panel, (ensure the program has stopped executing), right click on the slider and select **Properties**.

   In the properties box, look through the different tabs to see what properties can be adjusted.

2. Select the **Appearance** tab
3. Check the box labelled ‘Show digital display(s)’
4. Select the **Display Format** tab in the properties box, .
5. From the drop down box in the top left of the tab, select ‘Digital Display 0’
6. Change the **Digits** value to 2
7. From the **Precision Type** drop down, select ‘Digits of precision’
8. Click OK to commit these changes.

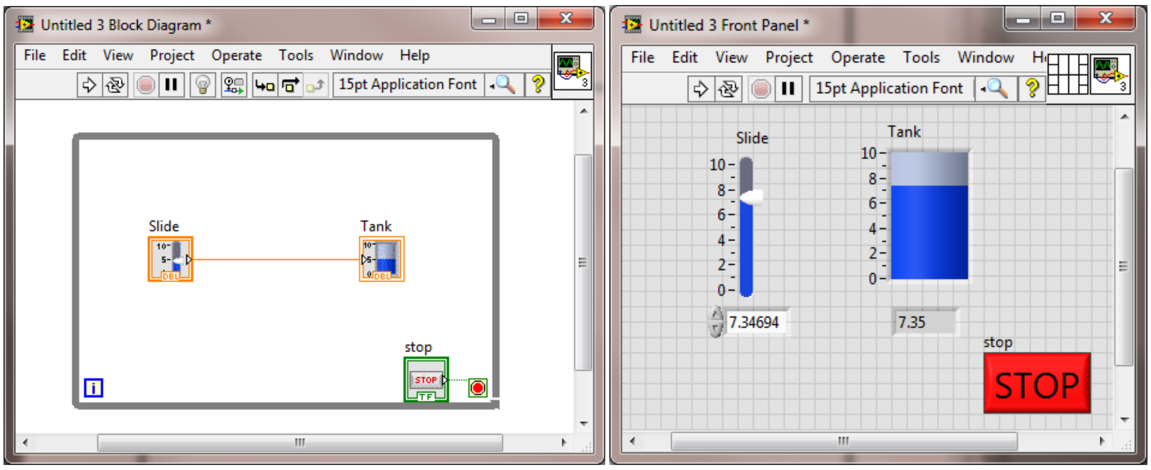
   A numerical control box should have appeared next to the vertical slider. This is intrinsically linked to the slider; any change in slider value will be reflected in the control box, and vice versa.

9. Run the program, and see how the slider and numerical control box are linked.
10. Stop the program by clicking on the STOP button on the from panel, then select and drag the numerical control box to underneath the slider.

11. Repeat the process to create a numerical indicator box under the tank.

The indicator box under the tank is only an indicator, and can not be used to alter the level value of the tank.

12. Run the program, and see how these elements interact.



13. Stop the program and save VI as 'Experiment 1.VI' to a convenient file store location, and proceed to the next excercise

Exercise 2 – Adding a timed loop and loop timer to the while loop

Goal: This exercise will demonstrate how a timed programming loop can be built, and show how the simple maths functions can be linked and used. In this example we will calculate the loop iteration time and total run time

Part 1

Description: This part of the exercise will guide you through the process of adding a **Wait Until Next ms Multiple** block to force the loop to execute at regular timed intervals. (For this example 15ms)

Procedure:

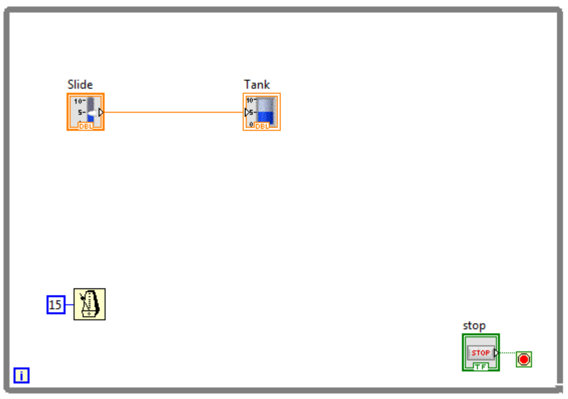
1. Starting from the program we developed in Exercise 1, switch to the block diagram, use **File»Save As…** to save the VI as '**Exercise 2.VI**'.

   You will be prompted with a number of save as options. In this case select **Substitute copy for original**.

2. Select a wait until next block by navigating to **Programming»Timing»Wait Until Next ms Multiple** and place it on the block diagram inside the while loop.

   The while loop box may need to be enlarged to accommodate the extra blocks required for this exercise. This can be easily achieved by holding the cursor over the edge of the while loop box and dragging one of the box resizing points to a suitable location. You may also want to move the stop button and termination condition to a suitable location.

3. Hold the cursor over the **Wait Until Next ms Multiple** block, and right click on the right hand blue port that appears, (with a flash up label of millisecond multiple), and select **Create»Constant**
4. Double left-click on the numerical constant box and change its value to 15. This will force the while loop to execute every 15ms if the system is capable of running at this speed, (longer if not, but it will be sufficient for our needs). To change the loop execution time frequency, change the millisecond multiple value input to this block accordingly.



Part 2

Description: During this part of the exercise we will add a timer to time the loop iteration time, in milliseconds, and also time the total run time the while loop is running for, in seconds.

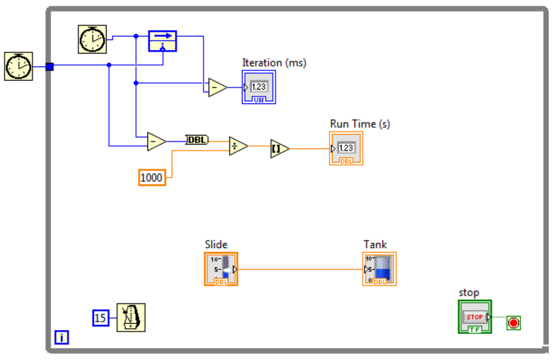
Two tick counters are used to provide the millisecond timer value at the start of execution and at each iteration of the while loop. A feedback node is used to store the timer value from the last loop iteration and this value is used to calculate the loop time.

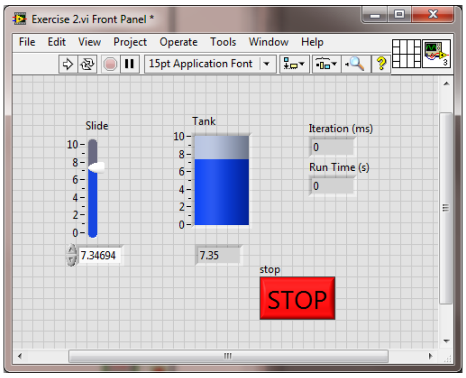
The tick counter from the outside of the while loop is passed into the loop through a tunnel port, seen on the grey box of the while loop. This port will store the value of the incoming signal, and make it available for all iterations of the loop.

The total time the while loop is running for is calculated from the current tick counter ms value minus the initial tick counter value. This value is divided by 1000 to convert it into seconds and then rounded to the nearest integer.

Procedure:
1. Select a tick counter block by navigating to **Programming»Timing»Tick Counter (ms)** and place it on the block diagram just to the left of the while loop.
2. Place another just inside while loop next to the first.
3. Select a feedback node by navigating to **Programming»Structures»Feedback Node** and placing it to the right of the tick counter inside the while loop. Change the direction of the feedback node by right clicking on the feed back node and select **Change Direction.** This should change the on the feedback node arrow to pointing to the right.
4. Connect the output of the tick counter inside the while loop to the input port, (left hand port), of the feedback node.

5. Connect the output port of the tick counter outside the while loop to the Initializer Terminal port, (bottom port), of the feedback node.
6. Select a subtract block by navigating to **Programming»Numeric»Subtract** and place it to the right and below the feedback node.
7. Connect the x port, (top left), of the subtract block to the input of the feedback node, and the y port, (bottom left), to the output of the feedback node. This provides a calculation of current tick counter value minus pervious.
8. Add an indicator node to the output port, (right side), of the subtract block by right clicking on the output port and selecting **Create»Indicator**. An indicator icon should appear, labelled 'x-y', attached to the output of the subtract block. This indicator will display the loop iteration time when the VI is run.
9. Double click on the 'x-y' label and change it to 'Iteration (ms)', notice that the front panel label has also changed.
10. Add a second subtract block under the feedback node, below the level of the 'Iteration (ms)' indicator.
11. Connect the x input to the output of the tick counter inside the while loop, and the y input to the output of the tunnel port from the other tick counter.
12. The output of the second subtract node is the form of a 32-but unsigned number – the native form of the tick counter. This should be cast into a double precision floating point number before it is passed into the divide block to convert the elapse time into seconds. Select a to double precision float block by navigating to **Programming»Numeric»Conversion»To Double Precision Float** and placing it to the right of the 2nd subtract block.
13. Connect the input to the output of the 2nd subtract block to the input of the to double precision float block.
14. Select a divide block by navigating to **Programming»Numeric»Divide** and place it to the right of the to double precision float block.
15. Connect the output of the to double precision float block to the x input of the divide block.
16. Right click on the y input of the divide and select **Create»Constant,** and change the constant value to 1000. This will divide the run time calculated at the 2nd subtract block from milliseconds to seconds.
17. We will now round the run time to the nearest second. Select a round to nearest block by navigating to **Programming»Numeric»Round to Nearest** and place it to the right of the divide block.
18. Connect the output of the divide block to the input of the round to nearest block
19. Add an indicator node to the output port, (right side), of the round to nearest block by right clicking on the output port and selecting **Create»Indicator**. An indicator icon should appear, labelled 'x-y 2', attached to the output of the subtract block. This indicator will display the run time for the while loop. Rename the indicator 'Run Time (s)'




Exercise 2.vi Front Panel *
File Edit View Project Operate Tools Window Help
15pt Application Font
Slide
Tank
Iteration (ms)
Run Time (s)
7.34694
7.35
stop
STOP

Note: The colouring of the connecting wires

- Orange: floating point numbers
- Blue: fixed point integer numbers
- Green: Boolean values

20. Run the VI and observe the output on the front panel.

    The 'Iteration (ms)' numerical indicator should show the millisecond execution time of the while loop, this should be 15, but may flicker to another value momentarily, due operating system interrupting the execution of the VI.

    The 'Run Time (s)' numerical indicator should count up in seconds, the run time of the VI.

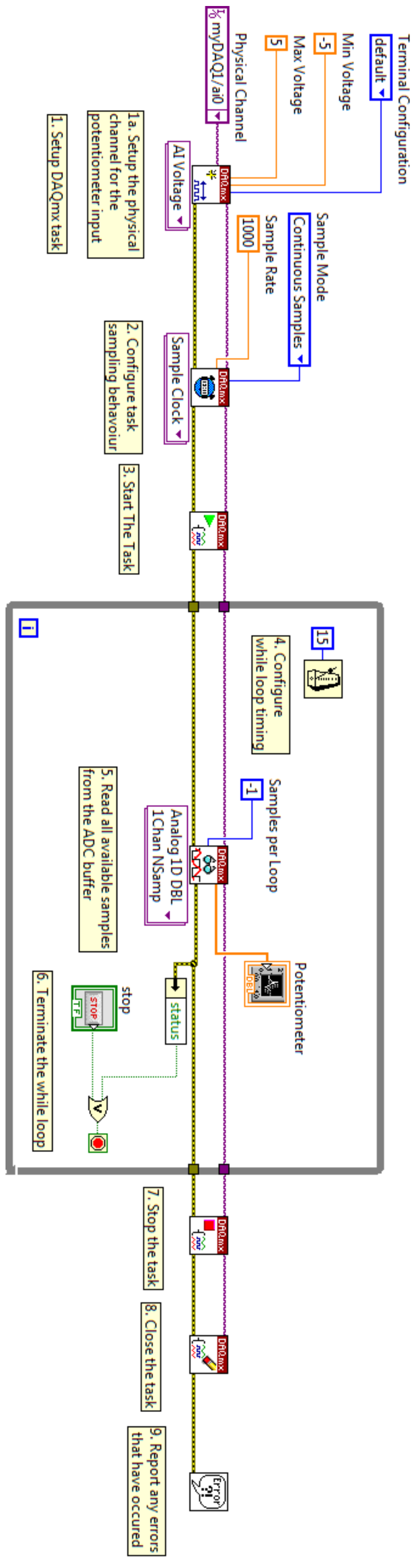
    To stop the VI please use the STOP button you have created no the front panel, **NOT** the Abort Execution button on the toolbar.

21. Save VI and proceed to the next Exercise.

Exercise 3 – DAQ – Single Channel Analogue Voltage Input

Goal: This exercise will lead you through the steps of implementing a simple single channel, multiple sample, analogue input data acquisition task, using the DAQmx libraries for National Instruments data acquisition devices.

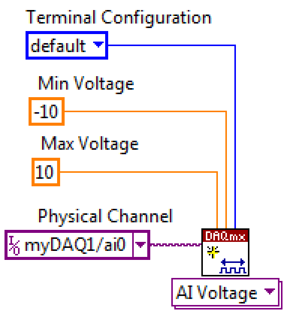
During this exercise you will build the following analogue input VI:


Terminal Configuration
default
Min Voltage
-5
Max Voltage
5
Physical Channel
myDAQ1/ai0
AI Voltage
1a. Setup the physical channel for the potentiometer input
1. Setup DAQmx task
Sample Mode
Continuous Samples
Sample Rate
1000
Sample Clock
2. Configure task sampling behaviour
3. Start The Task
15
4. Configure while loop timing
Samples per Loop
-1
Analog 1D DBL 1Chan NSamp
5. Read all available samples from the ADC buffer
Potentiometer
status
stop
6. Terminate the while loop
7. Stop the task
8. Close the task
9. Report any errors that have occured

The analogue input VI, shown above, comprises of a number of elements:

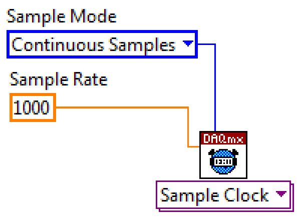
- Creation of the DAQmx task
    - The DAQmx task is a data acquisition function, e.g. analogue input (AI), analogue output (AO), digital input (DI), digital output (DO), or digital counter input (CNTR).

    - Only one function can be attributed to a DAQmx task, e.g. AI, but multiple tasks can be assigned to a device, e.g. one task for AI, one task AO, one task for DI, one task for DO and a CNTR task.
    - A task can comprise of a single physical channel or multiple physical channels.
    - The physical channel relates to the terminal port of the DAQ device. For the example above the physical channel is Dev1/ai0, or DAQ device 0 (assigned automatically by LabVIEW) and AI channel 0.
    - When creating the task you should configure the required port attributes, such as for this example: terminal configuration, maximum input voltage and minimum input voltage. (The terminal configuration for these examples we will use default)



    - The Create Task VI is located on the functions palette: **Measurement I/O»DAQmx – Data Acquisition»Create Channel**
    - Configure the block by clicking on the white drop down box blow the icon and selecting: **Analog Input»Voltage**
    - The input parameters, shown above, are all constants created by right clicking on the input ports, labelled, and adding a constant, **Create»Constant** from the pop-up menu.

- DAQmx Timing configuration
    - The DAQmx Timing VI configures the number of samples to acquire or generate, and the associated timing source.
    - For these exercises will only be considering continuous sampling at a fixed frequency of 1000 samples per second, using the internal sample clock; for the analogue input channels.



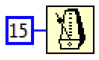
    - The DAQmx Timing VI is located on the functions palette: **Measurement I/O»DAQmx – Data Acquisition»Timing**
    - Configure the block by clicking on the white drop down box blow the icon and selecting: **Sample Clock(Analog, Counter, Digital)**

    - The input parameters, shown above, are all constants created by right clicking on the input ports, labelled, and adding a constant, **Create»Constant** from the pop-up menu.

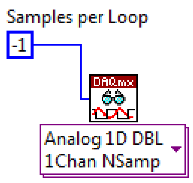
- Start the DAQmx task
    - After the DAQmx task is configured, then the task should be started before attempting to read from the DAQ device.

    - The DAQmx Start Task VI is located on the functions palette: **Measurement I/O»DAQmx – Data Acquisition»Start**

- Configure the while loop timing
    - The speed in which the main while loop iterates is limited by the speed of the system processor.
    - To slow the iteration speed of the while loop, loop timing control is required.
    - The method used in this exercise is discussed in Exercise 2.



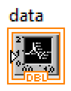
- Read acquired samples from the DAQ device buffer.
    - Inside the main while loop we wish to retrieve a number of samples of data acquired for this iteration of the while loop
    - For this exercise we will configure the task to read all samples in the ADC data buffer, (to achieve this set the ‘Samples per Loop’ to a value of -1).
    - Here, the DAQmx read VI is configured to read a task with a single physical channel, with multiple samples. The output of the block is configured to output a 1-D array of double precision floating point data. This data will be the voltage read at the terminal of the DAQ device.



    - The DAQmx Read VI is located on the functions palette: **Measurement I/O»DAQmx – Data Acquisition»Read**
    - Configure the block by clicking on the white drop down box blow the icon and selecting: **Analog»Single Channel»Multiple Samples»1D DBL**
    - The input parameters, shown above, are all constants created by right clicking on the input ports, labelled, and adding a constant, **Create»Constant** from the pop-up menu.

- Display the acquired data to a waveform chart

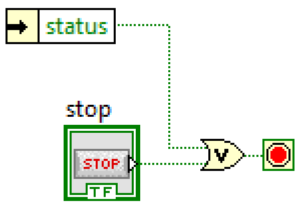
- The waveform chart is a front panel indicator that will display the time history of a variable or array



- To create the waveform chart, for this example:

a. Right click on the data port of the DAQmx read VI, and **Create»Indicator.** This will create a basic array indicator on the front panel
b. Switch to the front panel
c. Right click on the white index box of the array indicator and select **Replace»Graph»Waveform Chart**, (move and resize to suit)
d. Right click on the waveform chart and select **Chart History Length…**; set this value to 1024. This sets the size of the chart data buffer during run time, i.e. the chart will record and store the last 1024 data point from the DAQ module.

- Termination of the while loop
  - In this exercise the while loop is configured to terminate either by pressing the stop button, as with the previous exercises, or the loop is terminated in the event of an error occurring in data acquisition read process
  - The block linked to the brown striped error cluster line is an **Unbundle By Name** block, (this block is used to extract single or multiple named variables from a variable cluster). In this example this block is used to extract the error status flag from the error bundle. If an error occurs in the DAQmx read process, then this flag is set.
  - The error status output is OR'ed with the value of the STOP button on the front panel and fed into the termination condition, to allow either source to terminate the while loop



  - The Unbundle By Name VI is located on the functions palette: **Programming»Cluster, Class and Varient»Unbundle By Name**
  - The specific names variable is selected by clicking on the output namespace and selecting the required variable from the list
  - The OR VI is located on the functions palette: **Programming»Boolean»OR**

- Stop the DAQmx Task
  - The stop DAQmx task VI stops the DAQmx process, allowing the task to be reconfigured or cleared.

    - The DAQmx Start Task VI is located on the functions palette: **Measurement I/O»DAQmx – Data Acquisition»Stop**

- Clear the DAQmx Task
    - The Clear DAQmx task deleted the DAQmx task from the VI, and releases the resources, so they can be used by another task or process.
    - This should be enforced before terminating the VI to ensure the resources are correctly released.

    - The DAQmx Clear Task VI is located on the functions palette: **Measurement I/O»DAQmx – Data Acquisition»Clear**

- Report any error that have occurred
    - The simple error handle VI indicates whether an error occurred. If an error occurred, this VI returns a description of the error and can be configured to display a dialog box.

    - The DAQmx Clear Task VI is located on the functions palette: **Programming»Dialog and User Interface»Simple Error Handle**

Description: During this exercise you will build the VI described above, and use the test board, plugged into the top of the helicopter control module, to produce an analogue input for the DAQ system.

Procedure:
1. Build the single channel analogue input VI described above, using the setting described and illustrated above.
2. For the physical channel input constant, into the create channel VI, select the channel labelled ai0 channel from the pop-up menu
3. Set the maximum voltage, into the create channel VI, to a value of 10
4. Set the minimum voltage, into the create channel VI, to a value of -10
5. Ensure the Sample Mode on the DAQmx Timing Configuration block is set to **Continuous Sampling**
6. On the front panel, right click on an x-axis marker value and ensure the **AutoScale X** menu item is ticked, (if it *is not ticked* click on the menu item to select it)
7. Right click on a y-axis marker value and unsure the **AutoScale Y** menu item is un-ticked, (if it *is ticked* click on the menu item to deselect it)
8. Double click on the maximum y-axis value, and change its value to 12
9. Double click on the minimum y-axis value, and change its value to -12
10. Run the VI
11. Adjust the potentiometer on the test board and observe then change in the value on the waveform chart.

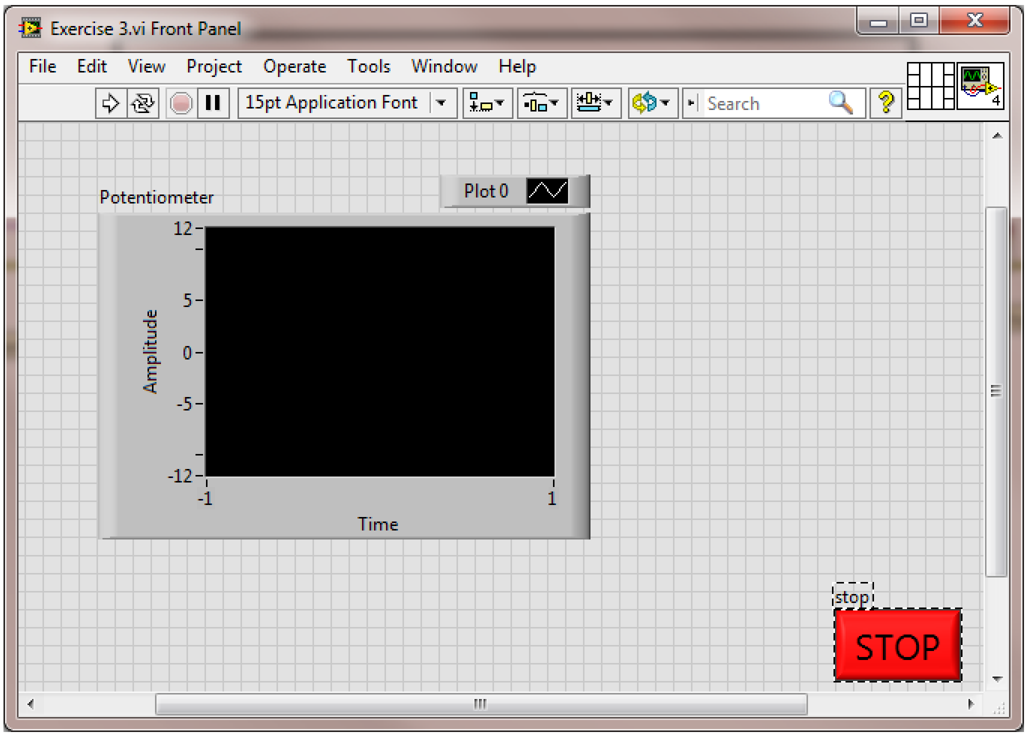
Notice that only about 1 seconds worth of data is stored on the chart. This is due to the data being samples at 1000 samples/second, and the chart history length being only 1024 samples.

12. Stop the VI
13. Right click on the waveform chart and set the **Chart History Length** value to 10,000.
14. Run the VI
15. Adjust the potentiometer on the test board and observe then change in the value on the waveform chart.

    Notice that this time about 10 seconds worth of data is stored on the chart. This is due to the data being samples at 1000 samples/second, and the chart history length now being set to 10,000 samples.

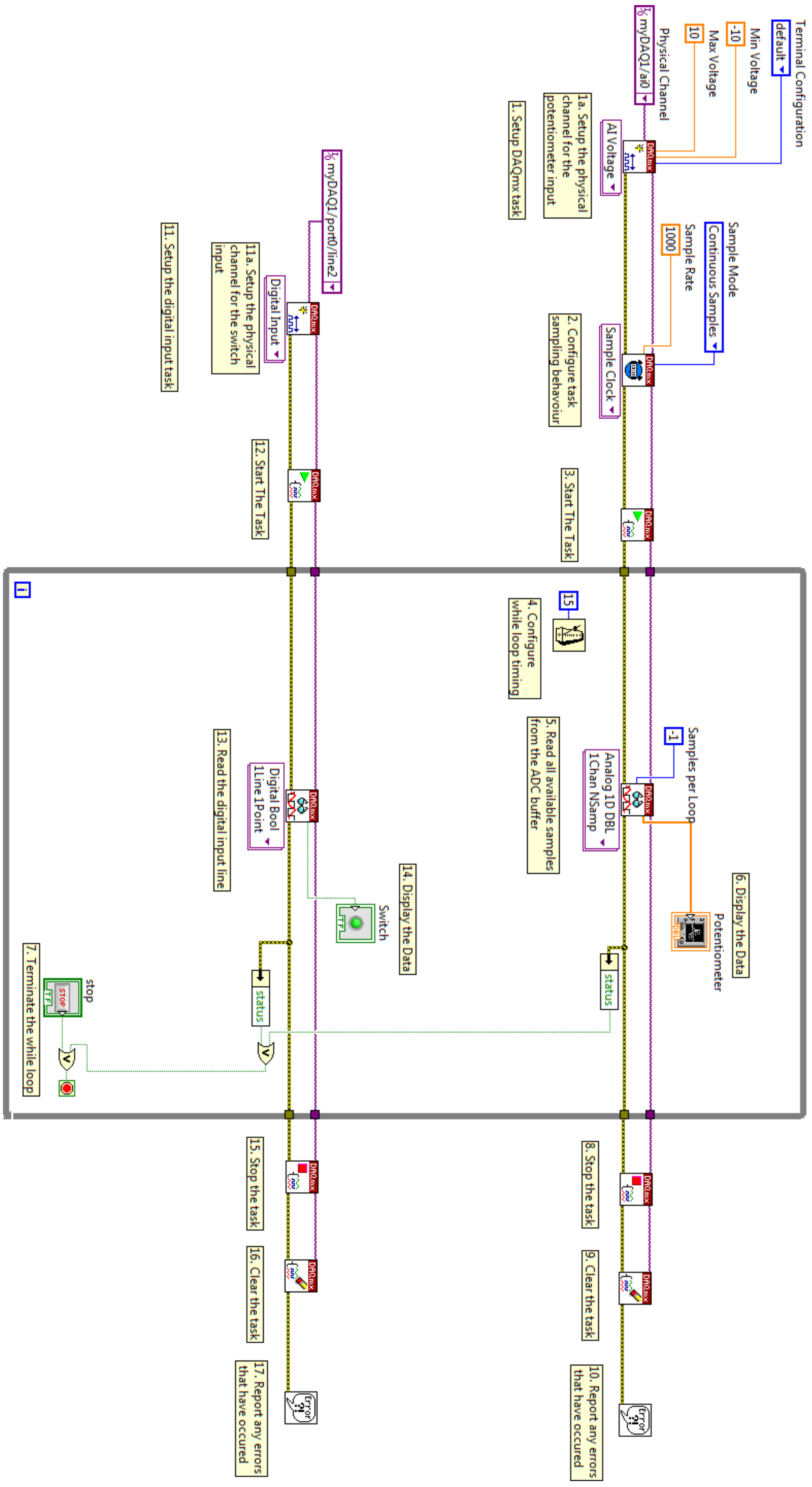
16. Save your VI as '**Exercise 3.VI**' and proceed to the next exercise

Your front panel can be rearranged to look something like this:



Exercise 4 – DAQ – Adding Single Channel Digital Input

Goal: This exercise will build on the previous exercise, and add in a digital input task to your VI, as shown below:

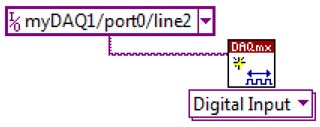


Terminal Configuration
default
Min Voltage
-10
Max Voltage
10
Physical Channel
myDAQ1/ai0
AI Voltage
1a. Setup the physical channel for the potentiometer input
1. Setup DAQmx task
Sample Mode
Continuous Samples
Sample Rate
1000
Sample Clock
2. Configure task sampling behaviour
3. Start The Task
15
4. Configure while loop timing
Samples per Loop
-1
Analog 1D DBL 1Chan NSamp
5. Read all available samples from the ADC buffer
6. Display the Data
Potentiometer
status
8. Stop the task
9. Clear the task
10. Report any errors that have occured
myDAQ1/port0/line2
Digital Input
11a. Setup the physical channel for the switch input
11. Setup the digital input task
12. Start The Task
Digital Bool 1Line 1Point
13. Read the digital input line
14. Display the Data
Switch
status
stop
7. Terminate the while loop
15. Stop the task
16. Clear the task
17. Report any errors that have occured

Description:

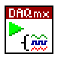
The VI shown above comprises of a number of elements, of which, 1-10 are discussed in the previous exercise. Elements 11-17 will be discussed below. Many if the VI block are the same as the previous exercise, and configured in a similar manner. You will need to use the procedures from the previous exercise to find and add the required blocks for the digital input task. Any major differences in configuration are discussed below.

11. Configuration of the digital input task
    - Similar to the previous task, this stage creates the digital input task, assigning the physical input channel to the task.

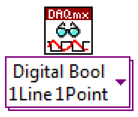


    - Configure the block by clicking on the white drop down box blow the icon and selecting: **Digital Input**
    - You will need to create the physical channel constant, in the manner described in the previous exercise, after the block task type has been configured, (i.e. place the block, configure the tack type then create the physical channel constant)

12. Start the DAQmx task
    - After the DAQmx task is configured, then the task should be started before attempting to read from the DAQ device.

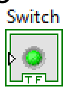


13. Read the digital input port
    - Use a DAQmx read block to read the digital port line



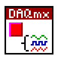
    - Configure the block by clicking on the white drop down box blow the icon and selecting: **Digital»Single Channel»Single Sample»Boolean (1 Line)**

14. Display the digital output data
    - The digital input value is a single Boolean value



    - To create the indicator right click on the DACmx data port and select **Create»Indicator**

15. Stop the DAQmx Task
    - The stop DAQmx task VI stops the DAQmx process, allowing the task to be reconfigured or cleared.



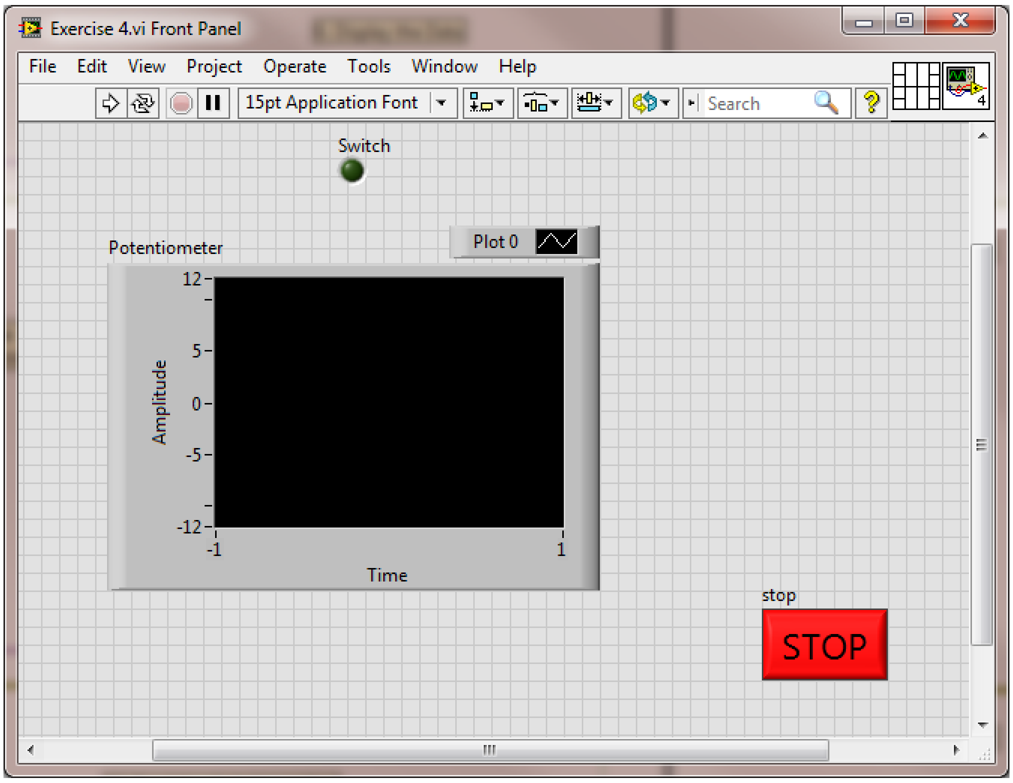
16. Clear the DAQmx Task
    - The Clear DAQmx task deleted the DAQmx task from the VI, and releases the resources, so they can be used by another task or process.

17. Report any error that have occurred
    - The simple error handle VI indicates whether an error occurred. If an error occurred, this VI returns a description of the error and can be configured to display a dialog box.

Procedure:

1. Starting from the program we developed in Exercise 3, switch to the block diagram, use **File»Save As…** to save the VI as '**Exercise 4.VI**'.
2. Add the VI components discussed above to build the VI shown in the diagram at the start of Exercise 4.
3. Ensure the while loop termination is also terminated if an error occurs in the digital input task. (see the VI diagram, at the start of the exercise, and Exercise 3 for details)

Your front panel can be rearranged to look something like this:



Ensure that the switch indicator is visible on the front panel.

4. Run the VI.

5. Adjusting the potentiometer should still produce the same result on the waveform graph.
6. Toggle the switch on the test board.

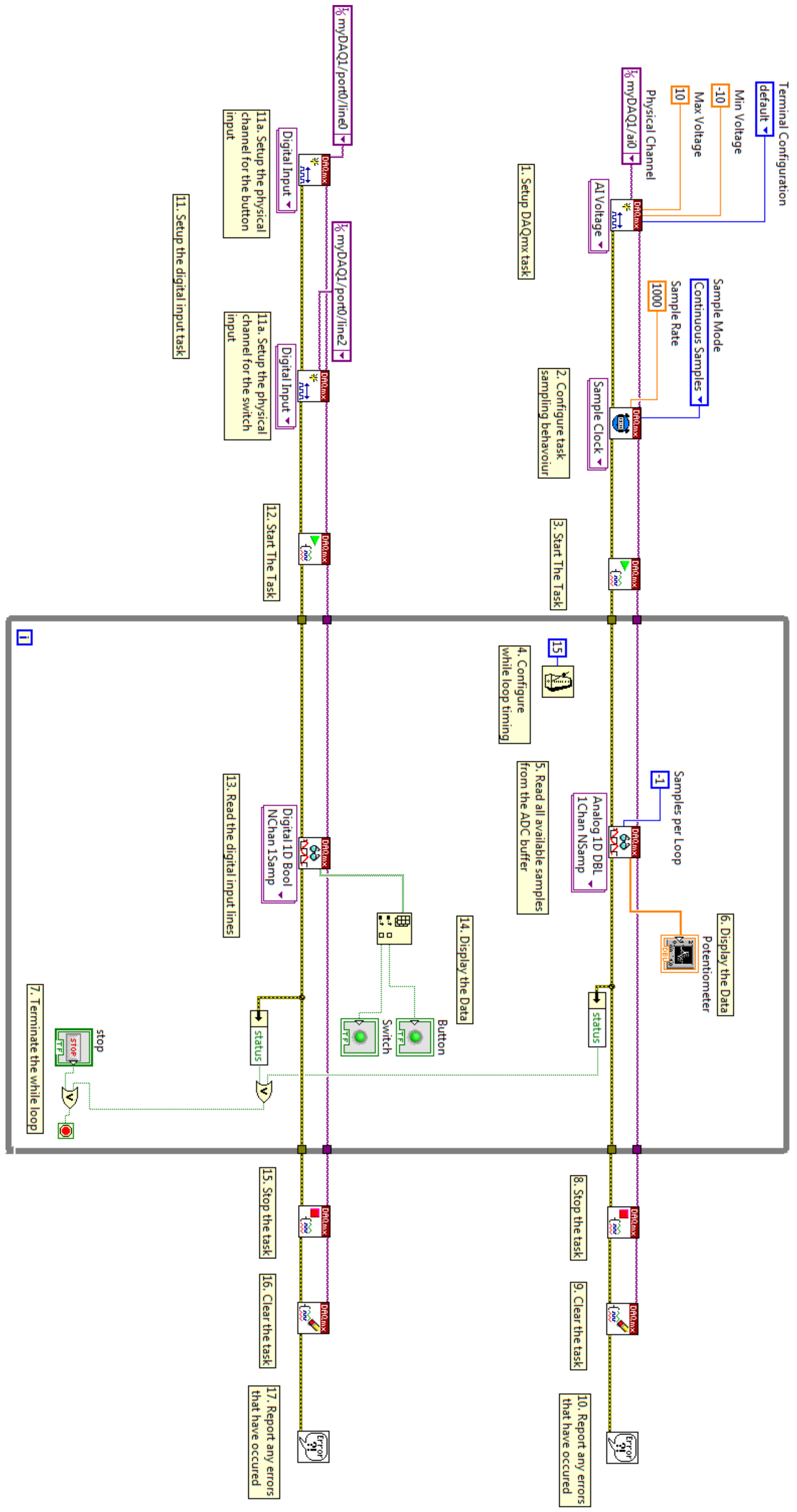
   You should notice the switch indicator light on the front panel switches on and off as you move the switch position backward and forward.

7. Stop the VI by pressing the STOP button on the front panel.

8. Save your VI, by **File»Save,** and proceed to the next Exercise.

Exercise 5 – DAQ - Adding Multichannel Digital Digital Input

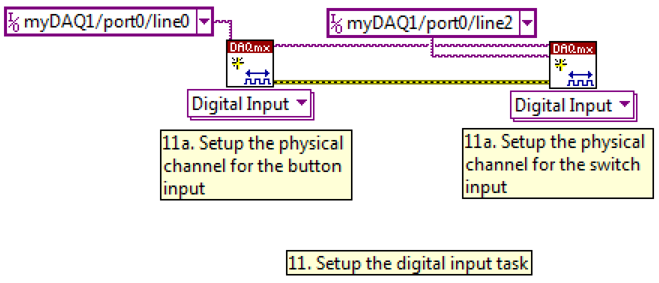
Goal: This exercise will build on the previous exercises, to demonstrate how a second physical channel can be added to a task, to incorporate multi-channel operation on the digital input task.


Terminal Configuration
default
Min Voltage
-10
Max Voltage
10
Physical Channel
myDAQ1/ai0
AI Voltage
1. Setup DAQmx task
Sample Mode
Continuous Samples
Sample Rate
1000
Sample Clock
2. Configure task sampling behaviour
3. Start The Task
15
4. Configure while loop timing
Samples per Loop
-1
Analog 1D DBL 1Chan NSamp
5. Read all available samples from the ADC buffer
6. Display the Data
Potentiometer
status
8. Stop the task
9. Clear the task
10. Report any errors that have occured
myDAQ1/port0/line0
Digital Input
11a. Setup the physical channel for the button input
myDAQ1/port0/line2
Digital Input
11a. Setup the physical channel for the switch input
11. Setup the digital input task
12. Start The Task
Digital 1D Bool NChan 1Samp
13. Read the digital input lines
14. Display the Data
Button
Switch
status
stop
7. Terminate the while loop
15. Stop the task
16. Clear the task
17. Report any errors that have occured

Description: To incorporate the multi-channel digital input, we start from the VI we created in Exercise 4. It can be seen that there is little difference between the VI of Experiment 4 and that shown above.

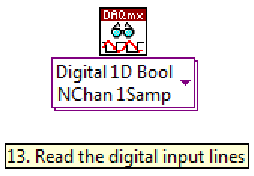
The key differences are:

A. There are two DAQmx task configuration blocks on the digital input task, one for each physical input channel.


myDAQ1/port0/line0
myDAQ1/port0/line2
DAQmx
Digital Input
11a. Setup the physical channel for the button input
11a. Setup the physical channel for the switch input
11. Setup the digital input task

The 'task out' port of the left-hand DAQmx configuration block is wired into the 'task in' port of the right-hand DAQmx configuration block. Using this method, multiple physical channels can be 'stringed' together to form a multi-channel task.

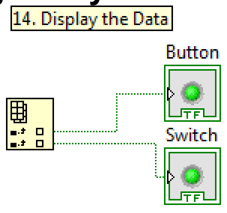
B. The DAQmx read clock is configured to **Digital»Multiple Channels»Single Sample»1D Boolean (1 line per channel)**



The output of the block is a 1D-Boolean array, where each physical channel value is attributed a corresponding element in the output Boolean array. The channel order in the array is arranged in order of adding the physical channels into the task.

For this example the output array has 2 elements. The first element in the array is the first digital port value to be configured into the task, (i.e. digital line2 corresponding to the button input). The second element is the in the array is the second digital port value to be configured into the task, (i.e. digital line0 corresponding to the switch input).

C. The output array, from the DAQmx read block, is separated into the individual indicator lines using an **Index Array** block found in the function palette by navigating to **Programming»Array»Index Array**.

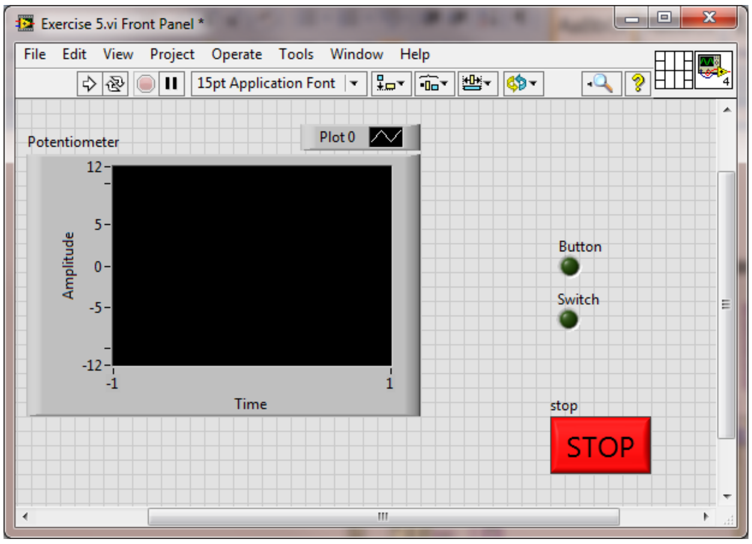


- In this example the array is implicitly indexed for elements 0 and 1, relating to the button value and the switch value, respectively. There are two explicit index ports for this VI block, (located in the bottom right corner), but we are not using this feature in these exercises. See VI block help for more details, (right click on the block and select **Help**; this will bring up the LabVIEW help for the specific block)

Procedure:

9. Starting from the program we developed in Exercise 4, switch to the block diagram, use **File»Save As…** to save the VI as '**Exercise 5.VI**'.
10. Redesign your VI for this exercise, as described above, so the block diagram is as shown in the figure at the start of the exercise.

Your front panel can be rearranged to look something like this:


Exercise 5.vi Front Panel *
File Edit View Project Operate Tools Window Help
15pt Application Font
Potentiometer
Plot 0
Amplitude
Time
Button
Switch
stop
STOP

11. Run the VI.
12. Adjusting the potentiometer should still produce the same result on the waveform graph.
13. Toggling the switch on the test board, should still exhibit the same behaviour as seen in the previous exercise.
14. Press the button on the on the test board

    You should notice the switch indicator light on the front panel, corresponding to the button, changes as the button is pressed and depressed.

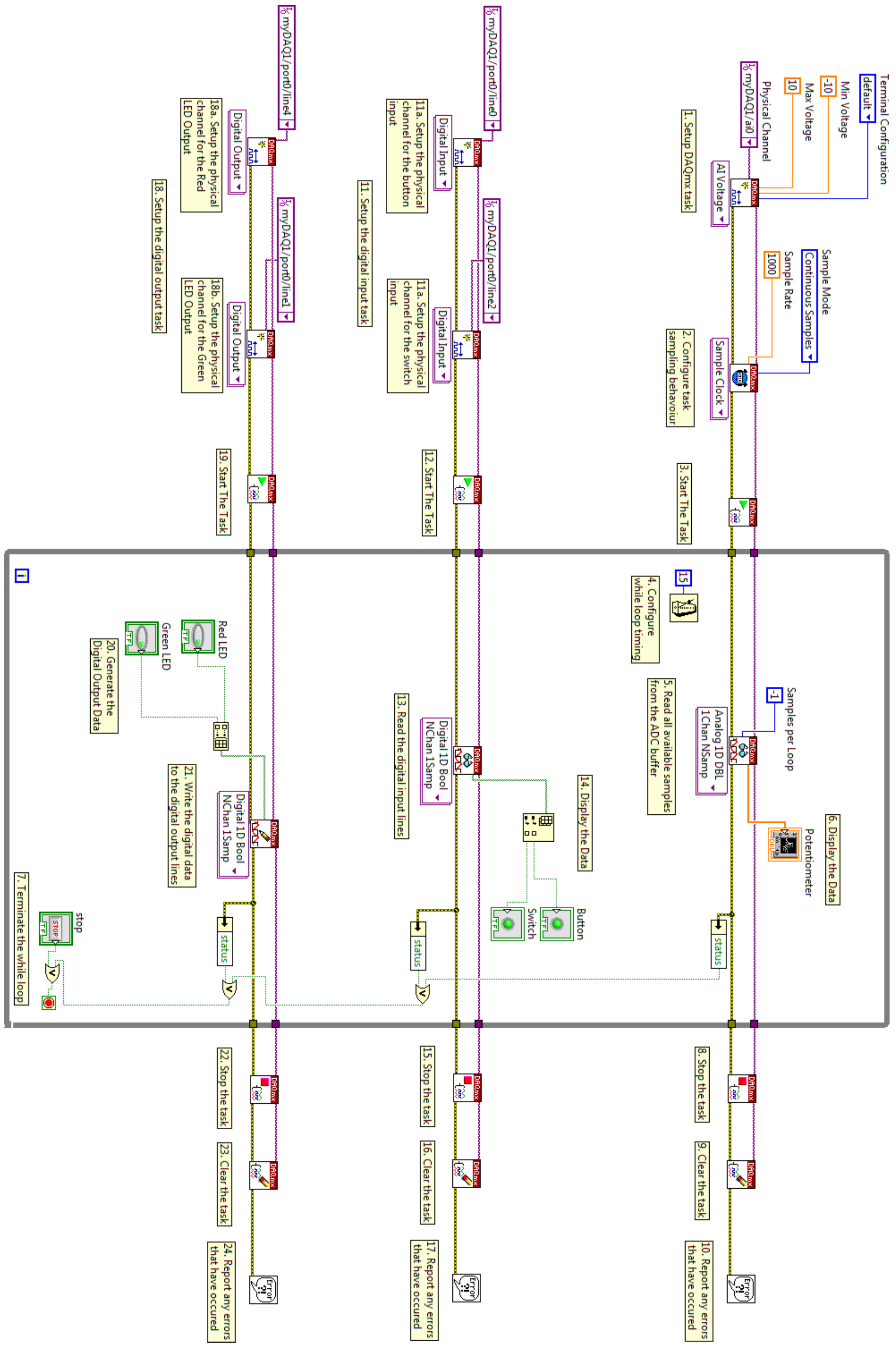
    You should now have a VI that a single analogue channel, and two independent digital input lines.

15. Stop the VI by pressing the STOP button on the front panel.

16. Save your VI, by **File»Save,** and proceed to the next Exercise.

Exercise 6 – DAQ – Adding Multichannel Digital Output

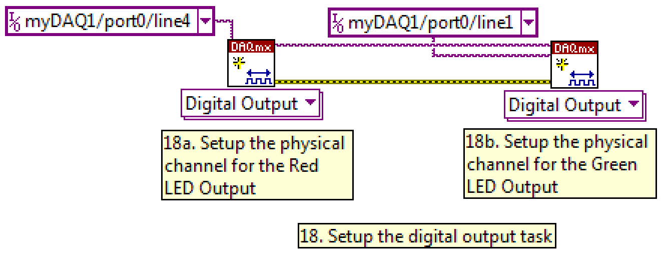
Goal: This exercise will build on the previous exercises, to demonstrate how a multichannel digital output task can be incorporated into the VI that we have evolved over the previous exercises.



Terminal Configuration
default
Min Voltage
-10
Max Voltage
10
Physical Channel
myDAQ1/ai0
1. Setup DAQmx task
AI Voltage
Sample Mode
Continuous Samples
Sample Rate
1000
Sample Clock
2. Configure task sampling behaviour
3. Start The Task
15
4. Configure while loop timing
Samples per Loop
-1
Analog 1D DBL 1Chan NSamp
5. Read all available samples from the ADC buffer
Potentiometer
6. Display the Data
status
8. Stop the task
9. Clear the task
10. Report any errors that have occured
myDAQ1/port0/line0
myDAQ1/port0/line2
Digital Input
11a. Setup the physical channel for the button input
11a. Setup the physical channel for the switch input
11. Setup the digital input task
12. Start The Task
Digital 1D Bool NChan 1Samp
13. Read the digital input lines
14. Display the Data
Button
Switch
15. Stop the task
16. Clear the task
17. Report any errors that have occured
myDAQ1/port0/line4
myDAQ1/port0/line1
Digital Output
18a. Setup the physical channel for the Red LED Output
18b. Setup the physical channel for the Green LED Output
18. Setup the digital output task
19. Start The Task
Red LED
Green LED
20. Generate the Digital Output Data
Digital 1D Bool NChan 1Samp
21. Write the digital data to the digital output lines
stop
7. Terminate the while loop
22. Stop the task
23. Clear the task
24. Report any errors that have occured

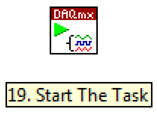
Description: The VI shown above comprises of a number of elements, of which, 1-17 are discussed in the previous exercises. Elements 18-24 will be discussed below. Many if the VI block are the same as the previous exercises, and configured in a similar manner. You will need to use the procedures from the previous exercise to find and add the required blocks for the digital input task.

18. Configure the DAQmx digital output task, and physical channels

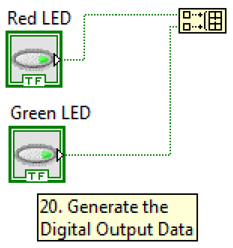

myDAQ1/port0/line4
myDAQ1/port0/line1
DAQmx
DAQmx
Digital Output
Digital Output
18a. Setup the physical channel for the Red LED Output
18b. Setup the physical channel for the Green LED Output
18. Setup the digital output task

- Remember to set the configuration block to Digital output

19. Start the task

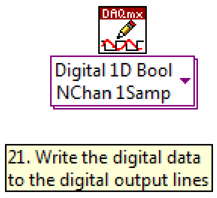


20. Generate the digital output values
    - The digital output data comprises of a 1-D Boolean array, generated from two front panel buttons.
    - A build array VI block is used to construct the array from the two independent Boolean wires.



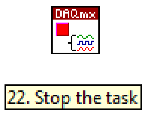
    - The buttons are selected from the front panel controls palette by navigating to **Modern»Boolean»Push Button**.
    - The two buttons that you have placed on the front panel can be labelled as shown.
    - The **Build Array** block found in the function palette by navigating to **Programming»Array»Build Array**.

21. Write the multi channel digital data to the digital output port lines
    - Use a DAQmx write block to write the array to the digital port

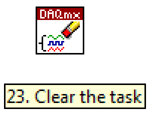

DAQmx
Digital 1D Bool NChan 1Samp
21. Write the digital data to the digital output lines

- Configure the block by clicking on the white drop down box blow the icon and selecting: **Digital»Multiple Channels»Single Sample»1D Boolean (1 line per channel)**

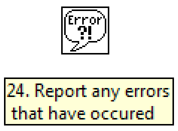
22. Stop the DAQmx Task



23. Clear the DAQmx Task



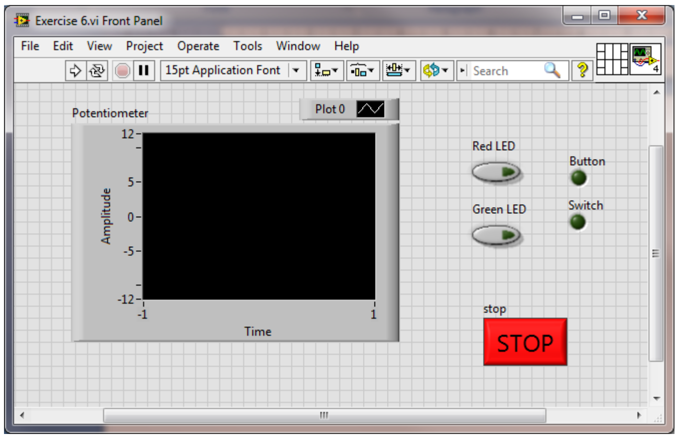
24. Report any error that have occurred



Procedure:

17. Starting from the program we developed in Exercise 5, switch to the block diagram, use **File»Save As…** to save the VI as '**Exercise 6.VI**'.
18. Add the VI components discussed above to build the VI shown in the diagram at the start of Exercise 4.
19. Ensure the while loop termination is also terminated if an error occurs in the digital output task. (see the VI diagram, at the start of the exercise, and Exercise 3 for details)

Your front panel can be rearranged to look something like this:



20. Run the VI.
21. Adjusting the potentiometer should still produce the same result on the waveform graph.
22. Toggling the switch on the test board, should still exhibit the same behaviour as seen in the previous exercise.
23. Pressing the button on the on the test board, should still exhibit the same behaviour as seen in the previous exercise.
24. Press the Green LED button on the front panel.

    The indicator on the front panel button should illuminate, and latch in that state. The green LED on the test board should extinguish.

25. Press the Green LED button again.

    The indicator on the front panel button should extinguish, and latch in that state. The green LED on the test board should illuminate.

26. Press the Red LED button on the front panel.

    The indicator on the front panel button should illuminate, and latch in that state. The Red LED on the test board should extinguish.

27. Press the Red LED button again.

    The indicator on the front panel button should extinguish, and latch in that state. The Red LED on the test board should illuminate.

    You should now have a VI that provides:
    - An input task for a single analogue input channel, with the input waveform recorded from the potentiometer onto a waveform chart
    - An input task for 2-digital input channels, with the values of the button and the switch displayed on independent screen indicators
    - An output task for 2-digital output channels, with the screen button values transferred to the status LEDs on the test board

28. Stop the VI by pressing the STOP button on the front panel.

29. Save your VI, by **File»Save,** and proceed to the next Exercise.

Exercise 7 – DAQ – Adding an analogue loopback, from an analogue output port to an analogue input port, (and configuring the digital output to drive the analogue multiplexer on the helicopter control board)

Goal: this exercise will guide you through configuring the analogue loopback facility on the helicopter control board.

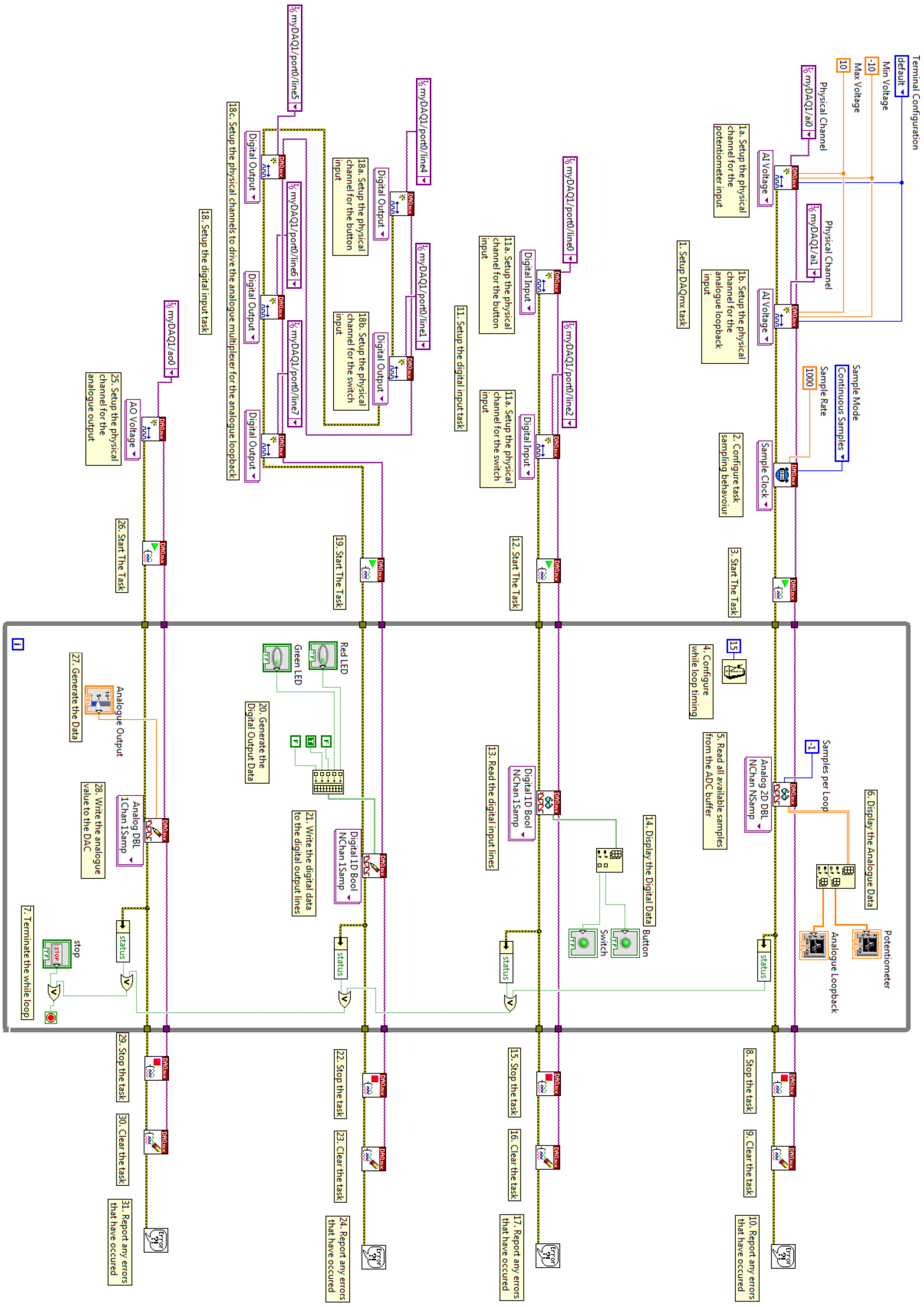
Description:
The analogue loopback facility on the helicopter control board allows an analogue voltage, from analogue output AO0 port, to be fed directly back into analogue input AI1 port.

To achieve this, the on-board analogue multiplexer on the helicopter control board must be configured to connect the AO0 port to the AI1 port. This is achieved by setting the digital output lines, DIO5-7, to the value of False-True-False, respectively.

This exercise will build on the VI developed in exercises 3-6, by adding:

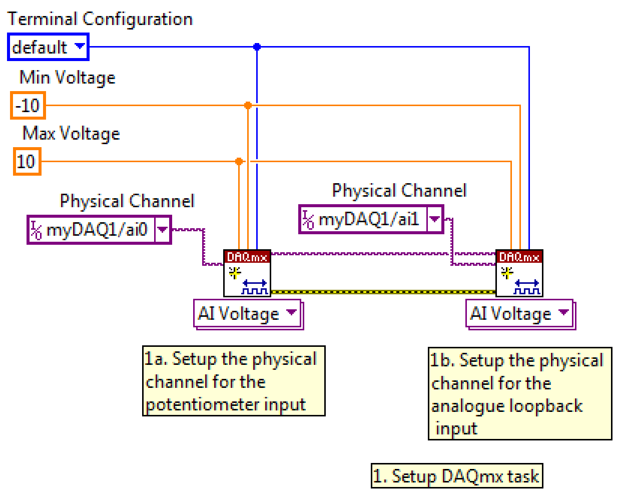
- Multi-channel analogue input capability to the single analogue channel input task
- The configuration address for the analogue multiplexer to the digital output task
- The analogue output task

The VI diagram for this exercise is shown on the next page.

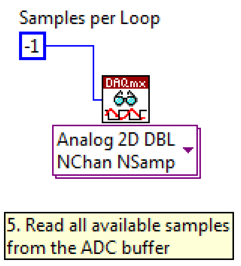

Terminal Configuration
default
Min Voltage
-10
Max Voltage
10
Physical Channel
myDAQ1/ai0
1a. Setup the physical channel for the potentiometer input
AI Voltage
Physical Channel
myDAQ1/ai1
1b. Setup the physical channel for the analogue loopback input
AI Voltage
1. Setup DAQmx task
Sample Mode
Continuous Samples
Sample Rate
1000
2. Configure task sampling behaviour
Sample Clock
3. Start The Task
15
4. Configure while loop timing
Samples per Loop
-1
Analog 2D DBL NChan NSamp
5. Read all available samples from the ADC buffer
6. Display the Analogue Data
Potentiometer
Analogue Loopback
status
8. Stop the task
9. Clear the task
10. Report any errors that have occured
myDAQ1/port0/line0
Digital Input
11a. Setup the physical channel for the button input
myDAQ1/port0/line2
Digital Input
11b. Setup the physical channel for the switch input
11. Setup the digital input task
12. Start The Task
Digital 1D Bool NChan 1Samp
13. Read the digital input lines
14. Display the Digital Data
Button
Switch
status
15. Stop the task
16. Clear the task
17. Report any errors that have occured
myDAQ1/port0/line4
Digital Output
18a. Setup the physical channel for the button input
myDAQ1/port0/line1
Digital Output
18b. Setup the physical channel for the switch input
myDAQ1/port0/line5
myDAQ1/port0/line6
myDAQ1/port0/line7
Digital Output
Digital Output
Digital Output
18c. Setup the physical channels to drive the analogue multiplexer for the analogue loopback
18. Setup the digital input task
19. Start The Task
Red LED
Green LED
20. Generate the Digital Output Data
Digital 1D Bool NChan 1Samp
21. Write the digital data to the digital output lines
status
22. Stop the task
23. Clear the task
24. Report any errors that have occured
myDAQ1/ao0
AO Voltage
25. Setup the physical channel for the analogue output
26. Start The Task
27. Generate the Data
Analogue Output
Analog DBL 1Chan 1Samp
28. Write the analogue value to the DAC
status
stop
7. Terminate the while loop
29. Stop the task
30. Clear the task
31. Report any errors that have occured

The key differences to the analogue input task are:

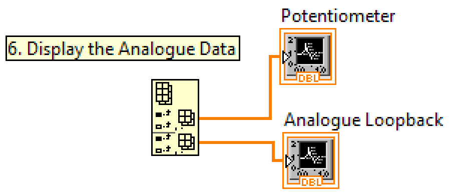
A. Rebuild the task configuration as follows:



B. Reonfiguration of the DAQmx read block by selecting **Analogue»Multiple Channels» Multiple Samples»2D DBL** from the white drop down box


Samples per Loop
-1
DAQmx
Analog 2D DBL NChan NSamp
5. Read all available samples from the ADC buffer

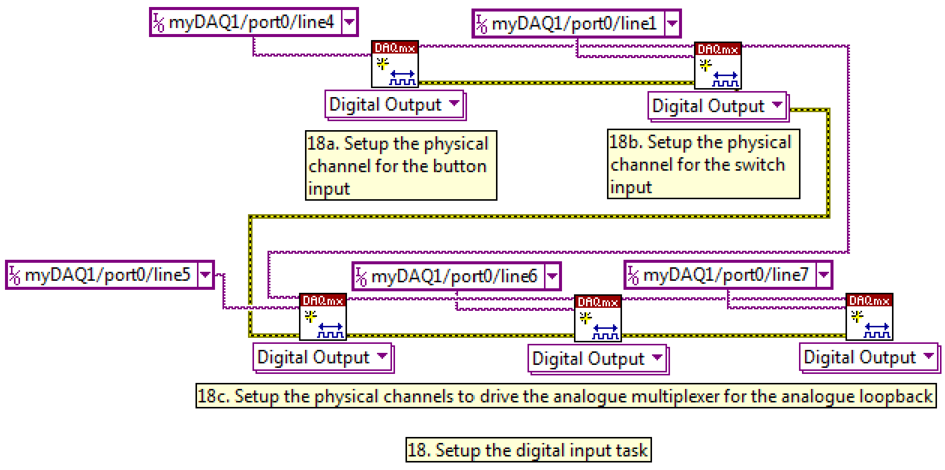
C. The displaying of the analogue data



- Delete the wire connecting the potentiometer waveform chart icon to the DAQmx write block
- Add an **Index Array** block found in the function palette by navigating to **Programming»Array»Index Array**.
- Wire it to the output of the read DAQmx block
- Copy and paste the potentiometer waveform chart icon, (<Ctrl+C> for copy, and <Ctrl+V> for paste) and re label it Analogue Loopback
- Connect the blocks as shown above
- On the front panel, change the scale on the y-axis of the Analogue Loopback waveform chart to 0 to 12

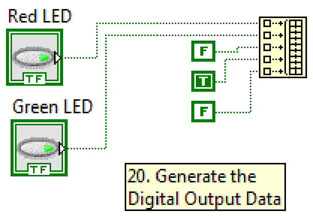
The key differences to the digital output task are:

A. The addition of the analogue multiplexer address output lines to the digital task configuration


myDAQ1/port0/line4
myDAQ1/port0/line1
DAQmx
Digital Output
Digital Output
18a. Setup the physical channel for the button input
18b. Setup the physical channel for the switch input
myDAQ1/port0/line5
myDAQ1/port0/line6
myDAQ1/port0/line7
Digital Output
Digital Output
Digital Output
18c. Setup the physical channels to drive the analogue multiplexer for the analogue loopback
18. Setup the digital input task

a. Three more DAQmx configuration clocks have been added to the digital output task, as shown above, for the analogue multiplexer address

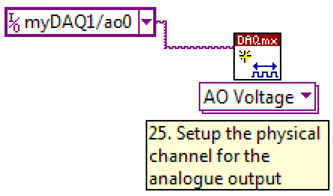
B. The addition of the analogue multiplexer address to the digital output data



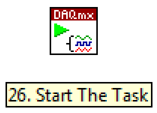
- The array size for the digital output must be the same size as the number of physical channels the task contains, (in this case 5).
- To achieve this we can increase the number of elements in the build array block to 5 by resizing it using the resizing points that appear when the curser is held over the bottom of the block.
- We need to add the Boolean constants to the build array block, in the usual way, right click on the input port and select **Create»Constant**
- To change the Boolean constant value, move the cursor over the constant icon, until the finger cursor appears, and click the constant icon. This should toggle the value from F, (False), to T, (True), or vice-versa.

The analogue output task is shown on the VI diagram, for the exercise, in elements 25-31:

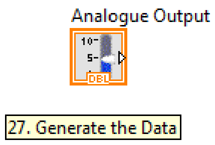
25. Configure the analogue output task and the physical channel, by selecting **Analogue Output»Voltage** from the white drop down box



26. Start the task

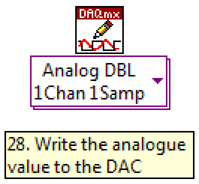


27. Generate the analogue data

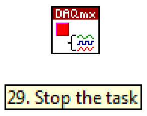


- Create the vertical point slider in the same manner as described in exercise 1.
- Set the scale of the vertical pointer slider to 0 to 10

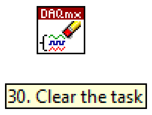
28. Configure the DAQmx write block of single channel, single sample write to the analogue output port, by selecting **Analogue»Single Channel»Single Sample»DBL** from the white drop down box



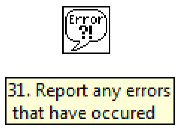
29. Stop the task



30. Clear the task



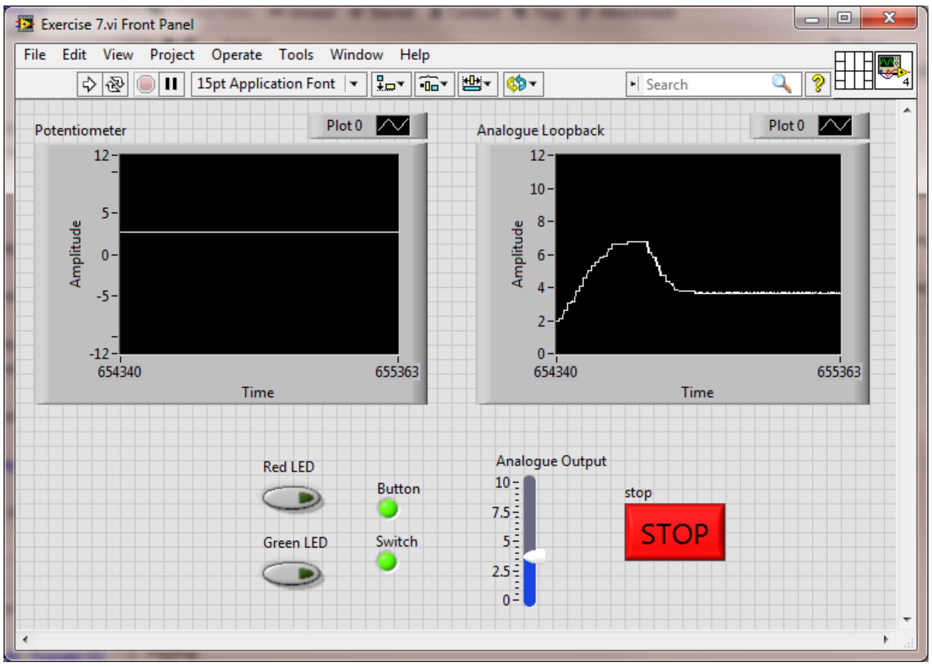
31. Report any error that have occurred



Procedure:

30. Starting from the program we developed in Exercise 6, switch to the block diagram, use **File»Save As…** to save the VI as '**Exercise 7.VI**'.
31. Add the VI components discussed above to build the VI shown in the diagram at the start this exercise.
32. Ensure the while loop termination is also terminated if an error occurs in the analogue output task. (see the VI diagram, at the start of the exercise, and Exercise 3 for details)

Your front panel can be rearranged to look something like this:



33. Run the VI.
34. Adjusting the potentiometer should still produce the same result on the waveform graph.
35. Toggling the switch on the test board, should still exhibit the same behaviour as seen in the previous exercise.
36. Pressing the button on the on the test board, should still exhibit the same behaviour as seen in the previous exercise.
37. Toggling the Green LED button on the front panel, should still exhibit the same behaviour as seen in the previous exercise.
38. Toggling the Red LED button on the front panel, should still exhibit the same behaviour as seen in the previous exercise.
39. Move the slider on the Analogue Output slider on the front panel and observe the effect on the Analogue Loopback waveform chart

    You should see the waveform adjust as the slider value adjusted.

40. Stop the VI by pressing the STOP button on the front panel.

You should now have a VI that provides:

- An input task for multiple analogue input channels, with the input waveforms recorded, from the potentiometer and the loopback channel, onto a waveform charts
- An input task for 2-digital input channels, with the values of the button and the switch displayed on independent screen indicators
- An output task for 5-digital output channels, with the screen button values transferred to the status LEDs on the test board, and digital address sent to the analogue multiplexer address lines.
- An output task for a single analogue output, with a vertical slider driving the analogue output value.

41. Save your VI, by **File»Save,** and proceed to the next Exercise.