

## Exercise 4-1: Managing Configuration Settings Using an INI File

### Goal

- Modify a DAQmx-based application to load its hardware configuration settings from an INI file.

### Instructions

#### Hardware Setup

**(Hardware)** In the exercises where we work with Analog Input/Output channels, we use PCI-6221/USB-6212 multifunction I/O device paired with the BNC-2120 shielded connector block. Analog Input 2 should be connected to the Sine/Triangle BNC connector. Analog Input 3 should be connected to the TTL Square Wave BNC connector. The Sine/Triangle waveform switch should be set to Sine.

#### Scenario

You developed an application that acquires and generates data, analyzes that data, and writes to a log file when the analysis determines that the values are out of range.

After using this application for some time, you decide that you would like to use it with a variety of DAQ hardware, without maintaining separate copies of the code to handle different settings. Instead, you decide to use INI files to store the settings needed for each type of hardware that you work with. That way, you can change the hardware settings by choosing a different INI file.

#### Design

You will develop a utility to create new INI files. Default.ini should contain these settings for your analog input device:

- AI Channel = "PCI-6221/ai0"
- Sample Rate = 1000
- Samples per Channel = 100

Default.ini should contain these settings for your analog output device:

- AO Channel = "PCI-6221/ao1"

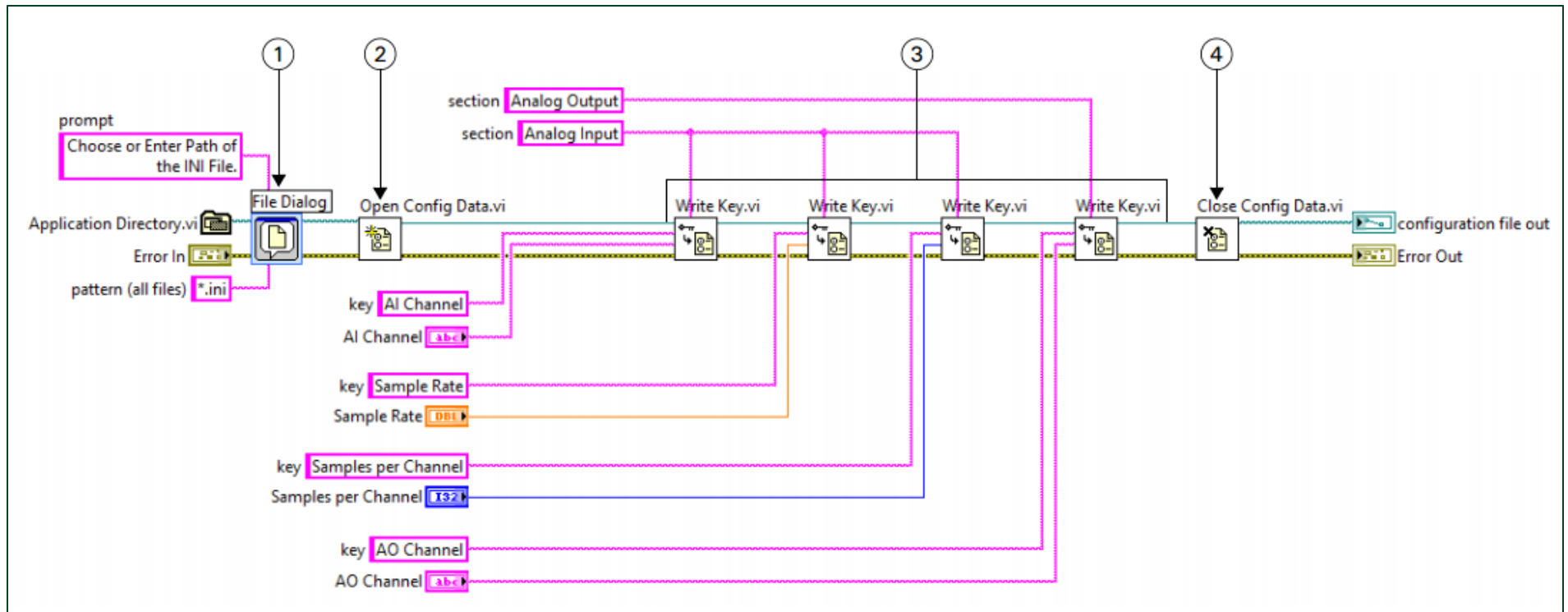
You will modify the Initialize case sub diagram of your application to load the values from Default.ini.

You will add a button to the user interface to allow the user to load settings from a different INI file. When the user clicks that button, the application should prompt them to choose an INI file to load.

### Guided Instructions

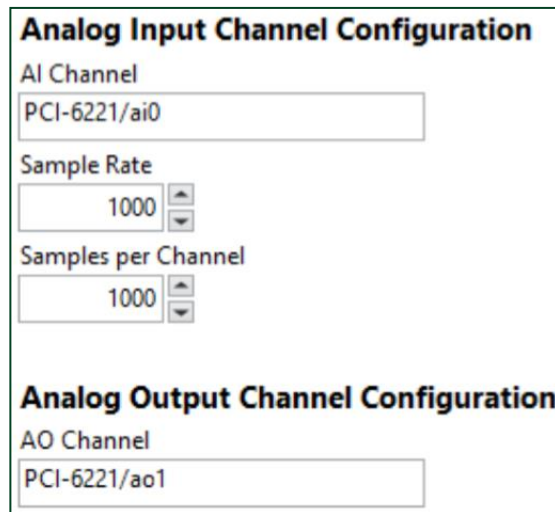
1. Open `C:\Exercises\LabVIEW Core 2>Loading and Storing Configuration\Load Config Settings.lvproj`.
2. Create a VI that generates an INI file.
  - From the **Project Explorer** window, open the Write Config Data VI from the support folder.

- Modify the block diagram as shown in the following figure.



1. **File Dialog VI** – Specify the path and name of the INI file that you want to create or modify.
2. **Open Config Data VI** – Creates or opens the INI file.
3. **Write Key VI** – Specify the section, key, and value that you want to write to the INI file. This VI automatically adapts to whatever data type you wire to the **value** input.
4. **Close Config Data VI** – Closes the INI file.

3. Save the VI.
4. Specify values to controls on the front panel.



**Analog Input Channel Configuration**

AI Channel  
PCI-6221/ai0

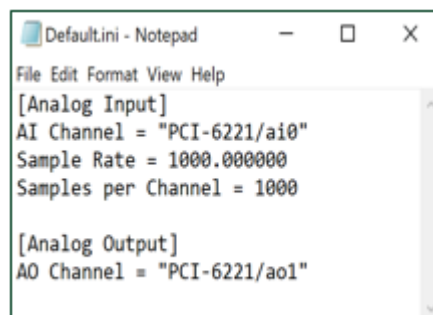
Sample Rate  
1000

Samples per Channel  
1000

**Analog Output Channel Configuration**

AO Channel  
PCI-6221/ao1

5. Run the VI.
  - Name the file `Default.ini` in Choose or Enter Path of the INI File window.
  - Save it in the same directory that the `.lvproj` file resides in.
  - When the VI stops, open `Default.ini` and review the format of the INI file.



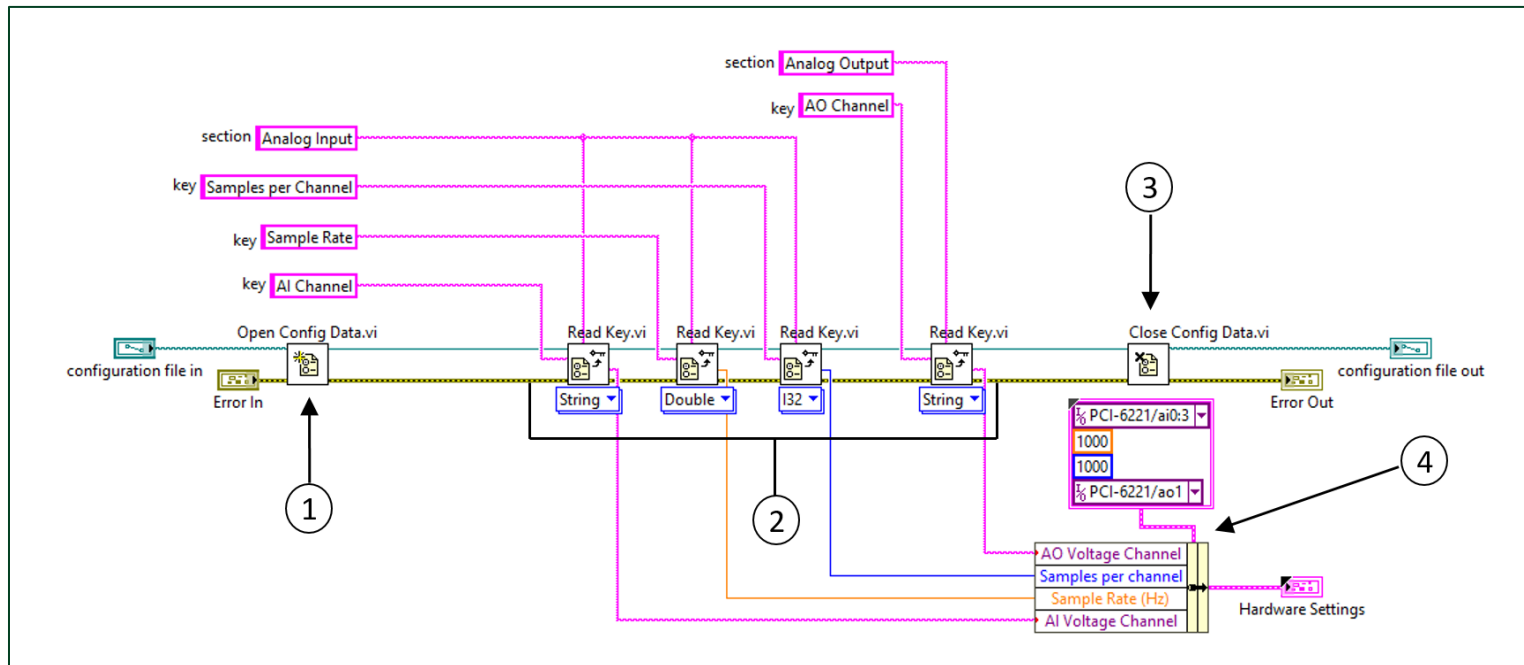
```
File Edit Format View Help
[Analog Input]
AI Channel = "PCI-6221/ai0"
Sample Rate = 1000.000000
Samples per Channel = 1000

[Analog Output]
AO Channel = "PCI-6221/ao1"
```

- Values are written to two sections: Analog Input and Analog Output.
- Each section contains one or more keys with their associated values.

6. Create a VI that reads configuration data from an INI file.
  - From the **Project Explorer** window, open Read Config Data VI from the support folder.

- Modify the block diagram as shown in the following figure.



1. **Open Config Data VI** – Opens the specified INI file.
2. **Read Key VI** – Specify the section and key that you want to read from the INI file. Use the Polymorphic VI Selector to specify the data type that the function will read from the INI file. The values for the **section** and **key** constants must match the values in the INI file exactly.
3. **Close Config Data VI** – Closes the INI file.
4. **Hardware Settings Type Definition and Bundle By Name function** – Place the Hardware Settings type definition on the block diagram from the Project Explorer window and wire the Read Key VIs' outputs as shown in the picture above using the Bundle By Name function.

7. Save the VI.
8. Verify that the front panel looks similar to the following figure.

configuration file in

configuration file out

Hardware Settings

AI Voltage Channel

$I_o$

Sample Rate (Hz)

0

Samples per channel

0

AO Voltage Channel

$I_o$

Error In

status code

d 0

source

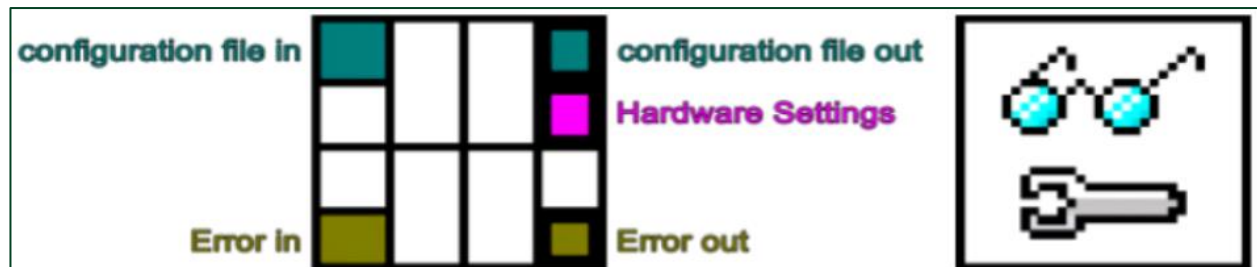
Error Out

status code

d 0

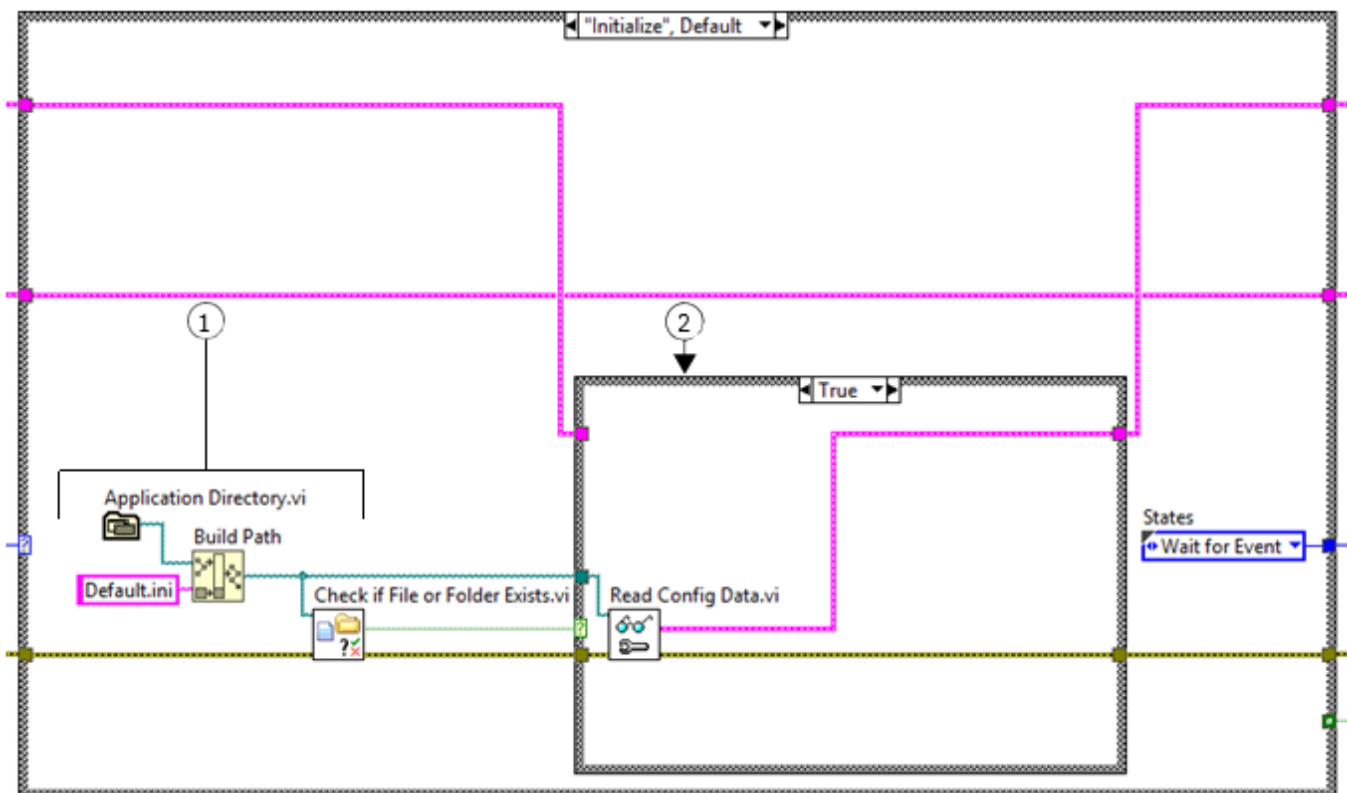
source

9. Verify the icon and terminals, as shown in the following figure.

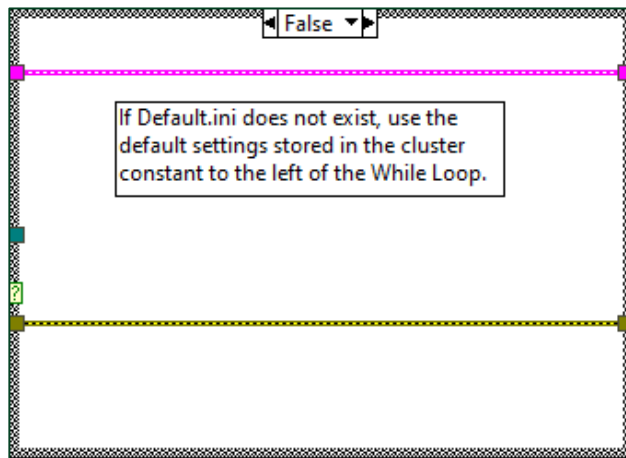


10. Test the read config file VI.

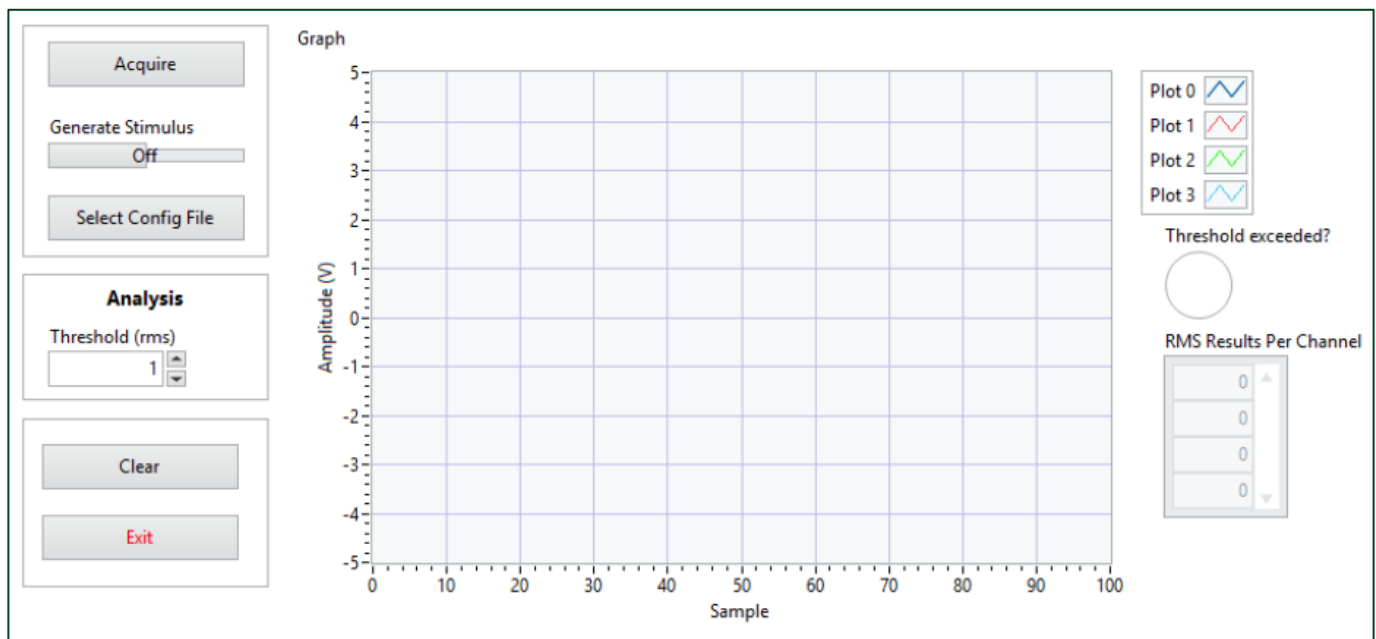
- On the front panel, browse the **configuration file** in control and navigate to C:\Exercises\LabVIEW Core 2\ Loading and Storing Configuration\Default.ini.
  - Run the VI.
  - Verify that each indicator is populated with the value that you specified in the INI file.
11. From the **Project Explorer** window, open the Event-Driven State Machine – Load Configuration VI.
  12. Modify the Event-Driven State Machine – Load Configuration VI as shown in the following figure, to load values from Default.ini each time the VI starts running.



1. **Application Directory, Build Path functions and Check if File or Folder Exists VI** – Use these tools to determine if a Default.ini file exists in the same directory as the LabVIEW project file (.lvproj) file for this application.
2. **Case Structure** – If the Default.ini file exists, then the application executes the **True** case, which uses the Read Config Data VI to read the configuration data from the INI file and store the configuration data into the Hardware Settings shift register. If the Default.ini file does not exist, then the application executes the **False** case and use the default values that initialized the Hardware Settings shift register. The False case passes the cluster and error wires through, as shown in the following figure.



13. Modify the front panel to include a **Select Config File** button that the user can click to select and load an INI file containing different DAQ configuration settings.

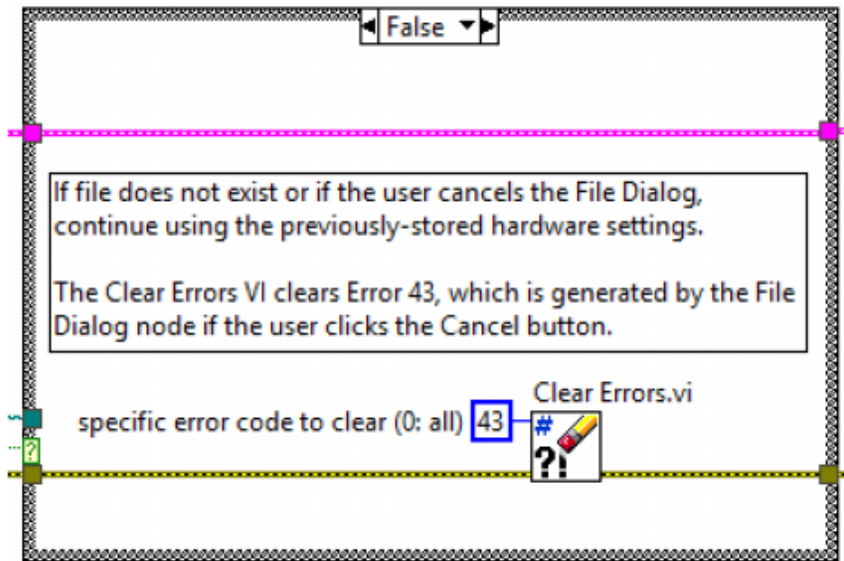


14. Modify the Wait for Event case to handle the event when the user clicks on the **Select Config File** button.

1. **“Select Config File”: Value Change event** – Select the **event structure border** and add an event case.
2. **File Dialog VI** – Use this VI to launch a file dialog for the user to select an **existing configuration file (.ini)**. In the **Configure File** dialog window, set the options to File and Existing. Use the Application Directory VI to set the start path of the file dialog to the directory containing the INI file.
3. **Case Structure** – Wire the exists output of the File Dialog VI to the **Selector** input of the Case structure.  
If the user selects a file that exists, the Case structure executes the True case, which uses the Read Config Data subVI to read the configuration data from the INI file and stores the configuration data into the Hardware Settings shift register.  
If the user selects a file that does not exist or clicks the **Cancel** button, the Case structure executes the False case, which continues to use the previously-stored hardware settings. The False case also uses the Clear Errors VI to clear Error 43, which the File Dialog VI passes out if the user clicks the **Cancel** button. This ensures that a user clicking **Cancel** in a file dialog will not cause the application to exit. You will learn more about the Clear Errors VI and error handling strategies in a later lesson.



- Wire the False case as shown in the following figure.



- The False case, which executes when the user cancels, passes the existing cluster values through and wires an empty error to the output terminal. Canceling Choose or Enter Path of File dialog box should not cause the application to halt or exit.

15. Run the Event-Driven State Machine – Load Configuration VI.

- Use execution highlight and probes to verify that the VI is loading configuration data from the INI file.
- Examine the values passed in the following sections of code.

Case Subdiagram	Expected Behavior
Initialize Case	Verify that the Read Config Data VI reads correct values if a Default.ini file exists in the LabVIEW project directory.
Acquire Case	Click the <b>Acquire</b> button. Verify that the Acquire case outputs the AI Voltage Channel, Sample Rate (Hz), and Samples per Channel values specified in the INI file.
Update Stimulus case	Click the <b>Generate Stimulus</b> button. Verify that the Update Stimulus case outputs the correct AO Voltage Channel value specified in the INI file.
Wait for Event»"Select Config File": Value Change event case	Click the <b>Select Config File</b> button. Choose the Default.ini file. Verify that the Read Config Data VI reads the correct values from the INI file. Click the <b>Select Config File</b> button again. In the file dialog box, click the <b>Cancel</b> button. Verify that the Case structure executes the False case, passes the previous shift register cluster data, and clears Error 43.

- Click **Exit** to stop the VI.

### Challenge

- Create additional INI files with different settings and observe the effects of those settings on the acquired data. For example, use a different AI Channel, such as PCI-6221/ai2:3, and a different number of Samples per Channel, such as 500.
- Modify the code to handle the situation where the INI file is missing one or more keys.
- Modify the Initialize case to programmatically create a new `Default.ini` file with default settings if `Default.ini` does not exist.

**On the Job**

Would any of your applications benefit from using a configuration file?

If so, write a draft below of the contents for your configuration file. For example, if you will use an INI file, write down the Section names, Key names, and Key values.

---

---

---

---

---

---

---

---

---

**End of Exercise 4-1**