KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

COLLEGE OF COMPUTER SCIENCES & ENGINEERING

Department of Systems Engineering

SE 438

INSTRUMENTATION AND PROCESS CONTROL

LAB MANUAL

Systems Engineering Department

SE 438 - INSTRUMENTATION AND PROCESS CONTROL

Laboratory Objective

Emphasize the practical aspects of the course and enable each student to:

- Simulate systems described using
  - LabVIEW

To understand, simulate and observe behavior of first and second order systems.

TABLE OF CONTENTS

EXPERIMENT # 1: PROCESS AND INSTRUMENTATION DIAGRAM

OBJECTIVE:

The objective of this experiment is to study Process and Instrumentation Diagram

PROCESS AND INSTRUMENTATION DIAGRAM:

An important means for engineering communication in the process industry is the so called Process & Instrumentation (P&I) diagram. Figure 1 shows the P&I diagram of a typical industrial heat exchanger. Heat exchanger is a process unit in which steam is used to heat up a liquid material. The material (called feedstock) is pumped at a specific flow rate into the pipes passing through the heat exchanger chamber where heat is transferred from steam to the material in the pipe. It is usually desired to regulate the temperature of the outlet flow irrespective of the change in the demand (flow rate) of the feedstock or change in the inlet temperature of the feedstock. The regulation of the outlet temperature is achieved by automatic control of the steam flow rate to the heat exchanger. The P&I diagram utilizes certain standard symbols to represent the process units, the instrumentation, and the process flow.

A Process & Instrumentation diagram consists of:

1- A pictorial representation of the major pieces of equipment required with major lines of flow to and from each piece.

2- All other equipment items with design temperatures, pressures, flow, etc..

3- All interconnecting piping with size, material and fabrication specifications indicated.

4. All major instrument devices.

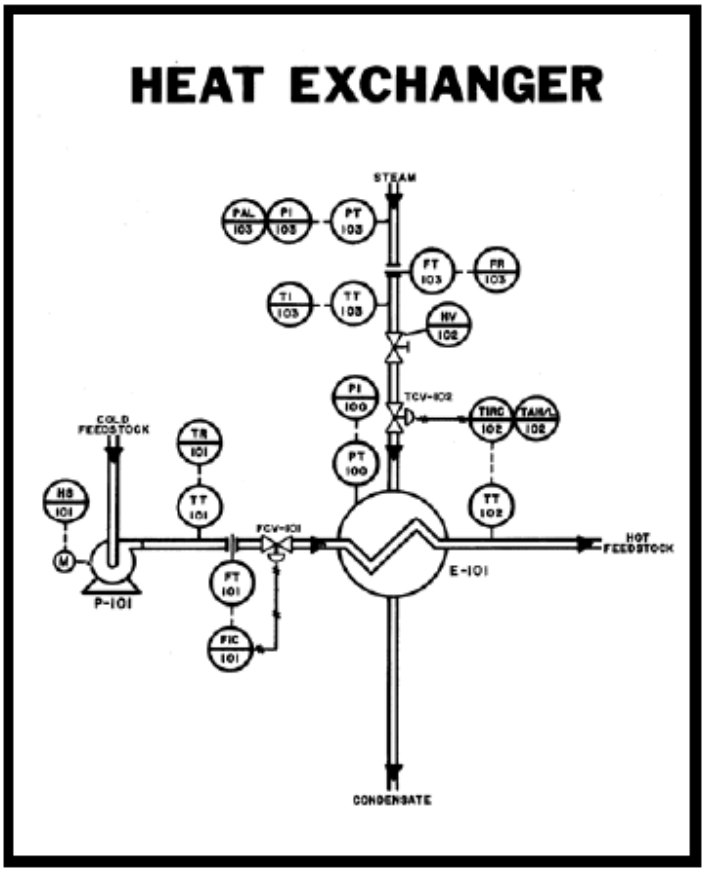


Figure 1: P&I Diagram of Heat Exchanger

A partial list of the symbols and abbreviations are given in the tables in Appendix 1.A at the end of this chapter. A comprehensive coverage may be found in the ISA standard in reference [1]. Instruments are shown on the P&I diagram by circles, usually called "balloons". The balloons contain alphanumeric which reflect the function of the instrument and its tag number. For example, TT102 means Temperature Transmitter number 2 in the process unit (or area) number 1. The number 102 is called tag number. Each Temperature Transmitter (TT) must have a unique tag number in the plant. Tag numbering may be different from one user to the other. P&I diagrams provide a valuable reference for proper project installation. The instrument engineer uses it as a source for many documents which must be prepared.

Another type of diagrams is known as ***Process flow Sheet.*** Process flow sheets consist also of a pictorial representation of the major pieces of equipment required with major lines of flow to and from each piece. However, additional information often given includes operating conditions at various stages of the process (flows, pressures, temperatures, viscosity, etc.), material balance, equipment size and configuration and, in some cases, utility requirements. On the other hand, instrumentation on process flow sheets may or may not be essentially complete.

A third type of diagrams is called ***Loop Wiring Diagrams.*** Electrical loop wiring diagrams are electrical schematic drawings which are prepared for individual (or typical) electrical loops. The simplest loop is one that contains only a transmitter and a receiver. Other loops may contain many items such as; transmitters, recorders, controllers, alarm units, control valves, transducers, integrators, and perhaps other items. Loop wiring Diagrams are intended to show the location of the instruments, their identification numbers and termination of interconnecting wiring. Cable routing, wire size intermediate terminal points and other pertinent information are necessarily shown in other drawings.

However, knowledge of these diagrams is not required to understand the material of this book. Understanding the basic P& I examples shown in this book can easily be achieved following the heat exchanger example. The reader should consult Appendix 1.A to verify and understand the following instruments list of the heat exchanger P&I diagram.

Instrument	Description
FIC-101	Flow Indicator and Controller.0 to 50 m3/Hr, (normal reading 30 T/Hr). This instrument controls the flow of cold feedstock entering the tube side of the heat exchanger by positioning a valve on the cold feedstock flow path.
FR-103	Flow Recorder, 0 to 10 Ton/Hr, (2.14 T/Hr). This instrument records the steam flow rate.
HS-101	Hand Switch, ON/OFF (ON). This switch turns on/off cold feedstock pump P-101. When the switch is in the ON condition, the pump is running. When the switch is in the OFF condition, the pump is not running.
HV-102	Hand Valve, OPEN/CLOSED, (OPEN). This switch opens/closes the steam block valve through which steam is routed from the header to the shell side of the heat exchanger. When the switch is in the OPEN condition the block valve is open. When the switch is in the CLOSED condition, the block valve is closed.
PAL-103	Pressure Alarm Low, (Normal). This alarm fires should the steam header pressure be less than 6 kg/cm.sqr.
PI-100	Pressure Indicator, 0 to 15 kg/cm.sqr , (3.18 Kg/cm2). This instrument displays the steam pressure at the shell side of the heat exchanger.
PI-103	Pressure Indicator, 0 to 15 kg/cm.sqr, (10.55 Kg/cm2). This instrument displays the steam header pressure.

TAH/L-102 Temperature Alarm High/Low, (Normal).
This alarm fires should the temperature of the feedstock at the exchanger outlet exceed 85 °C or be less than 71°C.

TI-103 Temperature Indicator, 0 to 200°C, (186 °C).
This instrument displays the temperature of the steam entering the shell side of the heat exchanger.

TIRC-102 Temperature Indicator, Recorder, and Controller, 0 to 200°C, (80°C).
This instrument controls the temperature of the feedstock at the exchanger outlet by positioning the valve that regulates the steam flow to the exchanger.

TR-101 Temperature Recorder, 0 to 200°C, (38 °C).
This instrument displays the temperature of the feedstock entering the exchanger.

APPENDIX 1.A

Process & Instrumentation Diagram

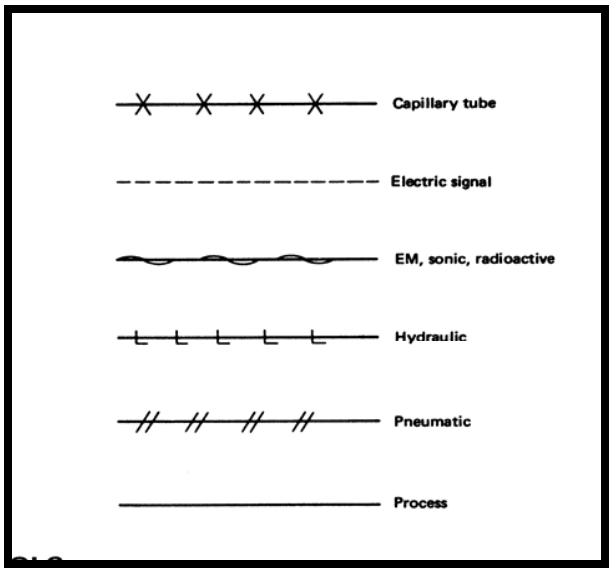


FIGURE 1.A-1
Line Symbols of the P&I Diagrams.

Table 1.A-1. Meanings of Identification Letters

First Letter	Succeeding letters	
A	Analysis	Alarm
B	Burner flame	
C	Conductivity	Control
D	Density or specific gravity	
E	Voltage	Primary element
F	Flow rate	
H	Hand (manually initiated)	High
	Current	Indicate
	Power	
K	Time or time schedule	Control station

L	Level	Light or low
M	Moisture or humidity	Middle or intermediate
O		Orifice
P	Pressure or vacuum	Point
Q	Quantity or event	
R	Radioactivity or ratio	Record or print
S	Speed or frequency	Switch
T	Temperature	Transmit
V	Viscosity	Valve, damper, or louver
W	Weight or force	Well
Y		Relay or compute
Z	Position	Drive

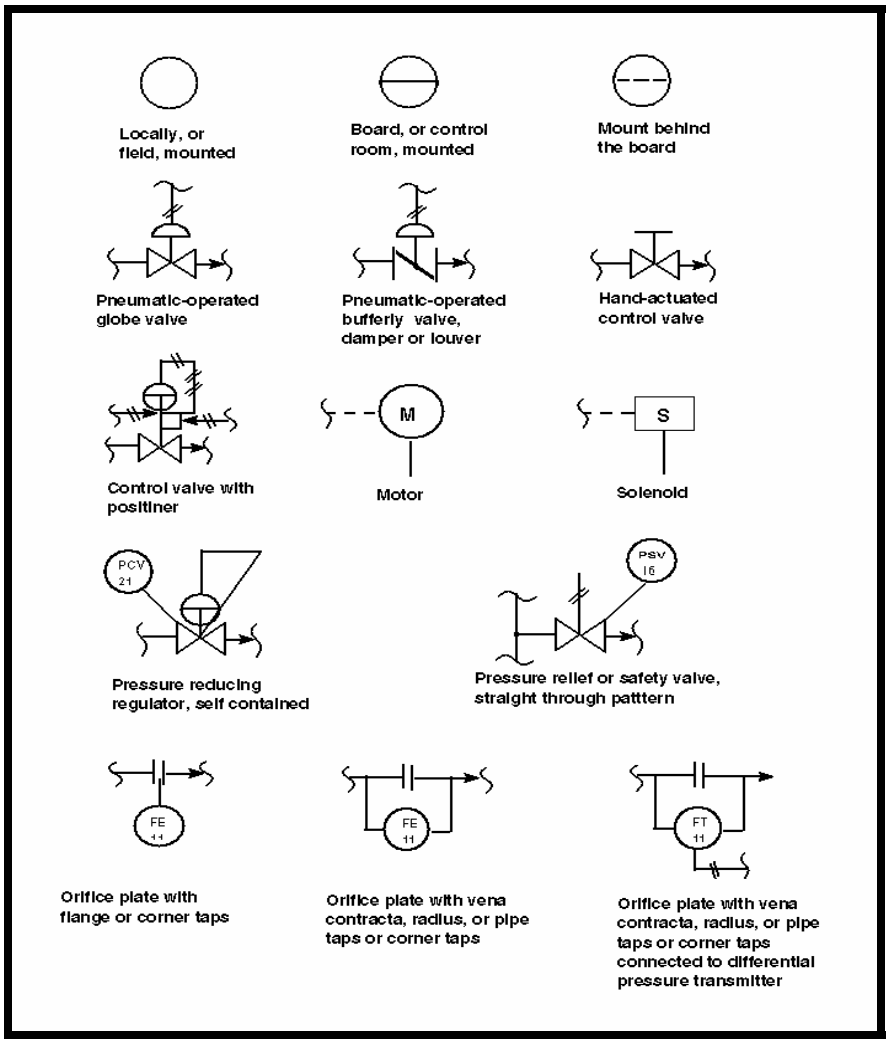
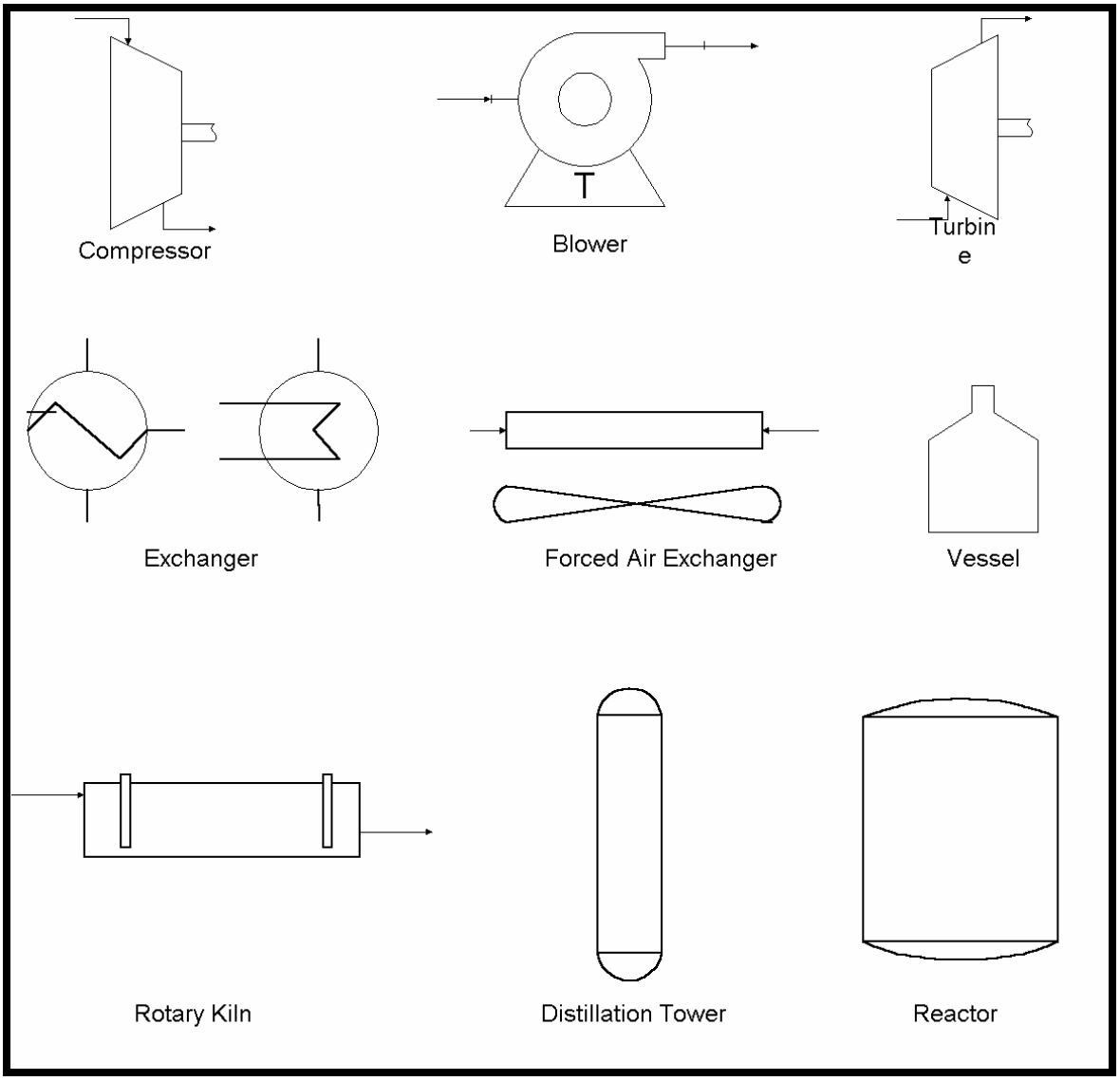


FIGURE 1.A-2
Summary of common symbols used in P&I diagrams.


Compressor
T
Blower
Turbin
e
Exchanger
Forced Air Exchanger
Vessel
Rotary Kiln
Distillation Tower
Reactor

EXPERIMENT # 2: INTRODUCTION TO LABVIEW PROGRAMMING

OBJECTIVE:

The objective of this experiment is to learn LabVIEW programming.

LabVIEW PROGRAMMING:

Complete the following steps to create a VI that takes a number representing degrees *Celsius* and converts it to a number representing degrees *Fahrenheit.*

1. A) Front Panel

1. Open a blank VI and begin building the following front panel.

Figure 1

2. (Optional) Select ***Window>>Tile Left and Right*** to display the front panel and block diagram side by side or ***Window>>Tile Up and Down*** to display the front panel and block diagram stacked.

3. Create a numeric control. You will use this control to enter the value for degrees Celsius.

1. Select ***Controls>>Numeric Controls*** to display the ***Numeric Controls*** palette. If the Controls palette is not visible, right-click an open space on the front panel workspace to display it.

2. Select the ***Numeric Control***. Move the control to the front panel and click to place the control.

3. Type deg C in the label of the control and press the ***<Enter>*** key or click the Enter button, on the toolbar. If you do not type the name immediately, LabVIEW uses a default label.
Note: You can edit a label at any time by double-clicking the label, using the Labeling tool, or right-clicking and selecting Properties from the shortcut menu to display the property dialog box.

4. Create a numeric indicator. You will use this indicator to display the value for degrees Fahrenheit.

1. Select the ***Numeric Indicator*** located on the ***Controls>>Numeric Indicators*** palette.

2. Move the indicator to the front panel and click to place the indicator.

3. Type deg F in the label and press the ***<Enter>*** key or click the Enter button.

1.b) Block Diagram

1. Display the block diagram by clicking it or by selecting ***Window>> Show Block Diagram***. LabVIEW creates corresponding control and indicator terminal icons on the block diagram when you place controls and indicators on the front panel. The terminals represent the data type of the control or indicator. You should see two double-precision, floating-point terminals on the block diagram, one indicator, and one control. **Note**: Control terminals have a thicker border than indicator terminals.

2.  Place the Multiply function, located on the ***Functions>>Arithmetic & Comparison>>Express Numeric*** palette, on the block diagram to the right of the deg C indicator. If the Functions palette is not visible, right-click an open space on the block diagram workspace to display it.

3.  Place the Add function, located on the ***Functions>>Arithmetic & Comparison>>Express Numeric*** palette, on the block diagram to the right of the Multiply function.

4.  Place a Numeric Constant, located on the ***Functions>>Arithmetic & Comparison>>Express Numeric*** palette, to the lower left of the Multiply function. Type 1.80 in the constant. When you first place a numeric constant, it is highlighted so you can type a value. If the constant is no longer highlighted, double-click the constant to activate the Labeling tool.

5.  Place a Numeric Constant, located on the ***Functions>>Arithmetic & Comparison>>Express Numeric*** palette, to the left of the Add function. Type 32.0 in the constant.

6. Use the ***Wiring*** tool, to wire the icons as shown in Figure 2.

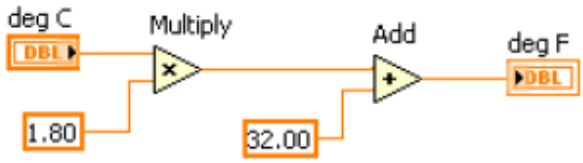


Figure 2

o To wire from one terminal to another, use the Wiring tool to click the first terminal, move the tool to the second terminal, and click the second terminal. You can start wiring at either terminal.

o You can bend a wire by clicking to tack down the wire and moving the cursor in a perpendicular direction. Press the spacebar to toggle the wire direction.

o To identify terminals on the nodes, right-click the Multiply and Add functions and select ***Visible Items>>Terminals*** from the shortcut menu to display the connector pane on the block diagram. Return to the icons after wiring by right-clicking the functions and selecting ***Visible Items>>Terminals*** from the shortcut menu to remove the checkmark.

o When you move the *Wiring* tool over a terminal, the terminal area blinks, indicating that clicking will connect the wire to that terminal and a tip strip appears, displaying the name of the terminal. If the *Context Help* window is open, the terminal area also blinks in the *Context Help* window.

o To cancel a wire you started, press the ***<Esc>*** key, right-click, or click the terminal where you started the wire.

7. Display the front panel by clicking it or by selecting ***Window>>Show Front Panel***.

8. Save the VI as ***Convert C to F.vi***

1. c) Run the VI

1. Enter a number in the numeric control and run the VI.

1. Use the *Operating* tool, or the *Labeling* tool to double-click the numeric control and type a new number.

2.  Click the *Run* button, shown at left, to run the VI.

3. Try several different numbers and run the VI again.

1.d) Icon and Connector Pane

1. Right-click the icon in the upper right corner of the front panel window and select *Edit Icon* from the shortcut menu. The *Icon Editor* dialog box appears.

2.  Double-click the *Select* tool, on the left side of the *Icon Editor* dialog box to select the default icon.

3. Press the <*Delete*> key to remove the default icon.

4.  Double-click the *Rectangle* tool, to redraw the border.

5. Create the icon in Figure 3.

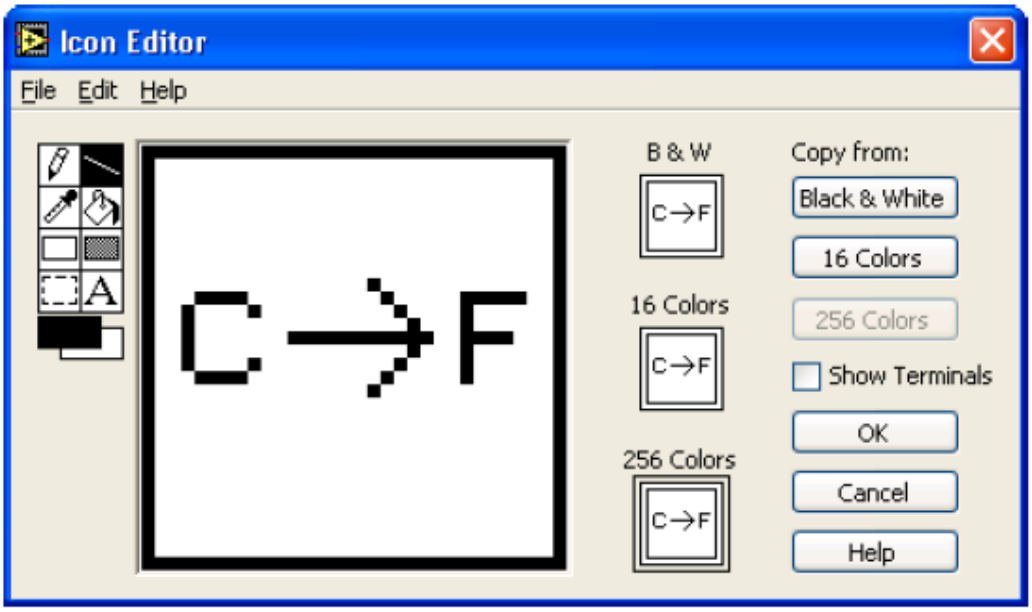


Figure 3

6. Save and close the VI. You will use this VI later in the course.

EXPERIMENT # 3: LabVIEW LOOPS PROGRAMMING

OBJECTIVE:

The objective of this experiment is to study LabVIEW Loops Programming

LABVIEW LOOPS PROGRAMMING:

While Loop

The While Loop is located on the Functions>>Execution Control palette. Select the While Loop from the palette then use the cursor to drag a selection rectangle around the section of the block diagram you want to repeat.
Add block diagram objects to the While Loop by dragging and dropping them inside the While Loop.
 The While Loop executes the subdiagram until the conditional terminal, an input terminal, receives a specific Boolean value. The default behavior and appearance of the conditional terminal is Stop If True. When a conditional terminal is Stop If True, the While Loop executes its subdiagram until the conditional terminal receives a True value.

 The iteration terminal, an output terminal, contains the number of completed iterations. The iteration count always starts at zero. During the first iteration, the iteration terminal returns 0.

For Loops

 The For Loop is located on the Functions>>All Functions>>Structures palette. You also can place a While Loop on the block diagram, right-click the border of the While Loop, and select Replace with For Loop from the shortcut menu to change a While Loop to a For Loop. The value in the count terminal (an input terminal), indicates how many times to repeat the subdiagram.

 The iteration terminal (an output terminal), contains the number of completed iterations. The iteration count always starts at zero. During the first iteration, the iteration terminal returns 0.
The For Loop differs from the While Loop in that the For Loop executes a set number of times.

Wait Functions

 The Wait Until Next ms Multiple function, monitors a millisecond counter and waits until the millisecond counter reaches a multiple of the amount you specify. Use this function to synchronize activities. Place this function within a loop to control the loop execution rate.

 The Wait (ms) function, adds the wait time to the code execution time. This can cause a problem if code execution time is variable.

Shift Registers

Use shift registers on For Loops and While Loops to transfer values from one loop iteration to the next.

 A shift register appears as a pair of terminals, directly opposite each other on the vertical sides of the loop border. The right terminal contains an up arrow and stores data on the completion of an iteration.

LabVIEW transfers the data connected to the right side of the register to the next iteration.

Create a shift register by right-clicking the left or right border of a loop and selecting Add Shift Register from the shortcut menu.

Feedback Nodes

 The Feedback Node appears automatically in a For Loop or While Loop if you wire the output of a subVI, function, or group of subVIs and functions to the input of that same VI, function, or group. Like a shift register, the Feedback Node stores data when the loop completes iteration, sends that value to the next iteration of the loop, and transfers any data type.

Exercise 1

Complete the following steps to build a VI that generates random numbers until the number generated matches a number you specify. The iteration terminal records the number of random numbers generated until a match occurs.

Front Panel

1. Open a blank VI and build the front panel shown in Figure 1. Modify the controls and indicators as shown in the front panel and as described in the following steps.

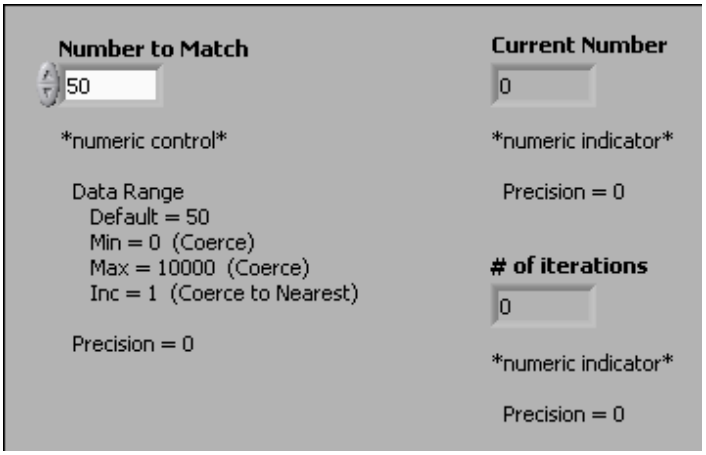


Figure 1

Setting the Data Range

Complete the following steps to set the range between 0 and 10000 with an increment of 1 and a default value of 50.

1. Right-click the Number to Match control and select Data Range from the shortcut menu. The Data Range page of the Numeric Properties dialog box appears.

    1. Remove the checkmark from the Use Default Range checkbox.

    2. Set the Default Value to 50.

    3. Set the Minimum value to 0 and select Coerce from the Out of Range Action pull-down menu.

    4. Set the Maximum value to 10000 and select Coerce from the Out of Range Action pull-down menu.

    5. Set the Increment value to 1 and select Coerce to Nearest from the Out of Range Action pull-down menu. Do not close the dialog box.

Block Diagram

1. Build the block diagram in Figure 2.

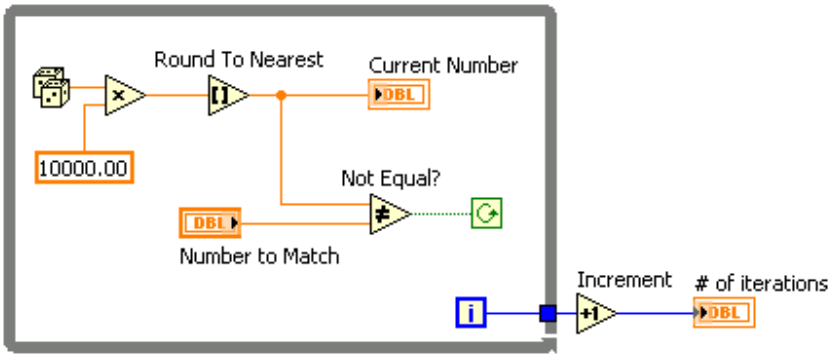


Figure 2

1. Place the Random Number (0-1) function, located on the Functions>>Arithmetic & Comparison>>Express Numeric palette, on the block diagram.

2. Place the Multiply function, located on the Functions>>Arithmetic & Comparison>>Express Numeric palette, on the block diagram.

3. 10000.00 Right-click the y terminal of the Multiply function, select Create>>Constant from the shortcut menu, type 10000, and press the <Enter> key to create a numeric constant.

4. Place the Round To Nearest function, located on the Functions>>Arithmetic & Comparison>>Express Numeric palette, on the block diagram.

5. Place the Not Equal? function, located on the Functions>>Arithmetic & Comparison>>Express Comparison palette, on the block diagram.

6. Place the While Loop, located on the Functions>>All Functions>>Structures palette, on the block diagram. Right-click the conditional terminal and select Continue if True from the shortcut menu.

7. Wire the iteration terminal to the border of the While Loop. A blue tunnel appears on the While Loop border. You will wire the tunnel to the Increment function. Increment this value by one outside the loop because the count starts at 0.

8. Place the Increment function, located on the Functions>>Arithmetic & Comparison>>Express Numeric palette, on the block diagram.

2. Save the VI as Auto Match.vi.

EXPERIMENT # 4: LABVIEW'S ARRAY'S AND CLUSTERS

OBJECTIVE:

The objective of this experiment is to study Arrays and Clusters in LabVIEW.

LABVIEW'S ARRAYS AND CLUSTERS:

Arrays

Arrays group data elements of the same type. An array consists of elements and dimensions. Elements are the data that make up the array. A dimension is the length, height, or depth of an array. An array can have one or more dimensions and as many as $2^{31}-1$ elements per dimension, memory permitting.
You can build arrays of numeric, Boolean, path, string, waveform, and cluster data types. Consider using arrays when you work with a collection of similar data and when you perform repetitive computations.
A 2D array stores elements in a grid. It requires a column index and a row index, both of which are zero-based, to locate an element.

Clusters

Clusters group data elements of mixed types, such as a bundle of wires, as in a telephone cable, where each wire in the cable represents a different element of the cluster. A cluster is similar to a record or a struct in text-based programming languages.
Bundling several data elements into clusters eliminates wire clutter on the block diagram and reduces the number of connector pane terminals that subVIs need. The connector pane has, at most, 28 terminals. Like an array, a cluster is either a control or an indicator. A cluster cannot contain a mixture of controls and indicators.

Exercise 1

Complete the following steps to build a VI that creates an array of random numbers, scales the resulting array, and takes a subset of that final array.
Front Panel

1. Open a blank VI and build the front panel shown in Figure 1.

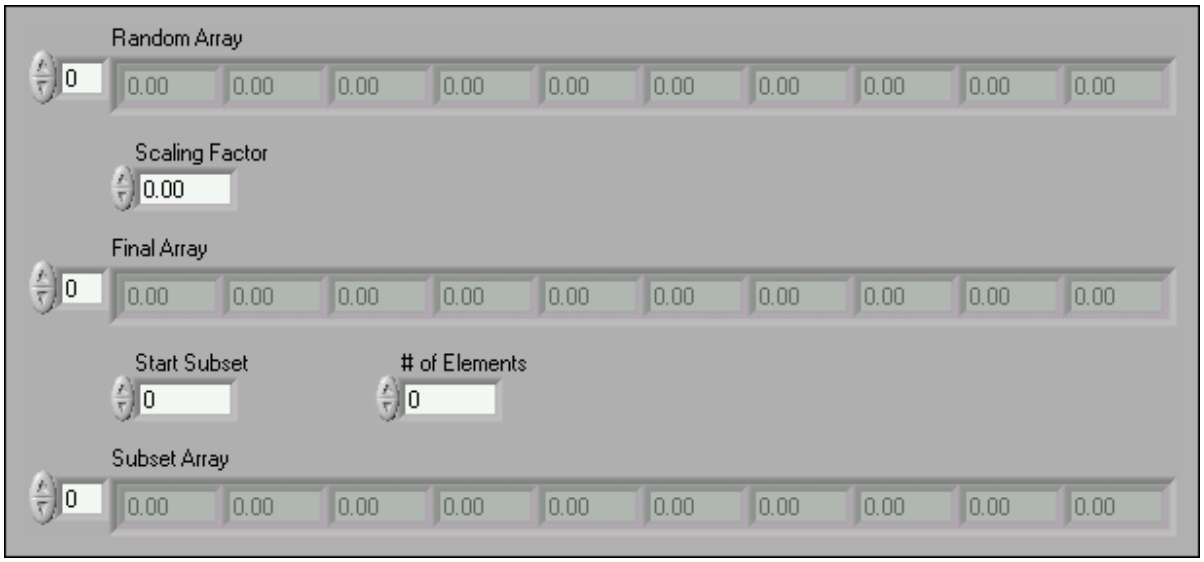


Figure 1

1. Place an `array`, located on the `Controls>>All Controls>>Array & Cluster` palette, on the front panel.
2. Label the array `Random Array`.
3. Place a `numeric indicator`, located on the `Controls>>Numeric Indicators` palette, in the `array` shell.
4. Use the `Positioning` tool to resize the array control to contain 10 `numeric indicators`.
5. Press the `<Ctrl>` key while you click and drag the `Random Array` control to create two copies of the control.
6. Label the copies `Final Array` and `Subset Array`.
7. Place three `numeric controls`, located on the `Controls>>Numeric Controls` palette, and label them `Scaling Factor`, `Start Subset`, and `# of Elements`.
8. Right-click the `Start Subset` and `# of Elements` controls and select `Representation>>I32` from the shortcut menu.
9. Do not change the values of the front panel controls.

Block Diagram

1. Build the block diagram shown in Figure 2.

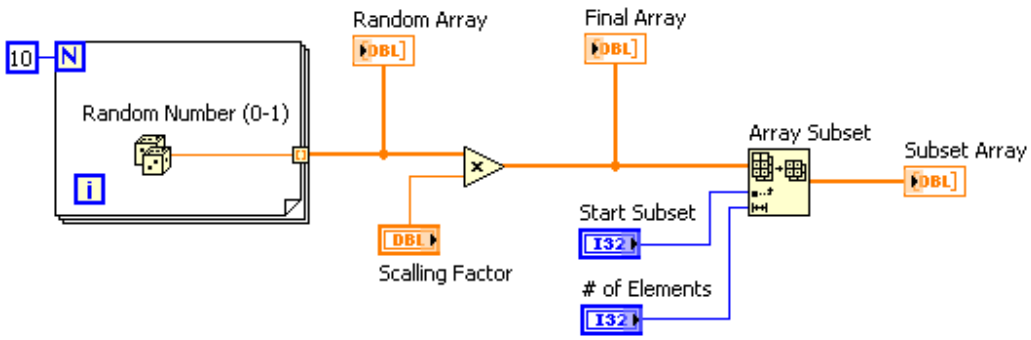


Figure 2

1. Place the `Random Number (0-1)` function, located on the `Functions>>Arithmetic & Comparison>>Express Numeric` palette, on the block diagram.

2.  Place the `For Loop`, located on the `Functions>>All Functions>>Structures` palette, on the block diagram. The loop accumulates an array of 10 random numbers at the output tunnel. Create a constant of 10 for the count terminal.
3.  Place the `Multiply` function, located on the `Functions>>Arithmetic & Comparison>>Express Numeric` palette, on the block diagram. In this exercise this function multiplies `Random Array` by `Scaling Factor` and returns `Final Array`.
4.  Place the Array Subset function, located on the `Functions>>All Functions>>Array` palette, on the block diagram. This function returns a portion of an array starting at `Start Subset` and containing `# of Elements` elements.

2. Save the VI as `Array Exercise.vi`.

- **Exercise 2**

Front Panel

1. Open a blank VI and build the front panel in Figure 3.

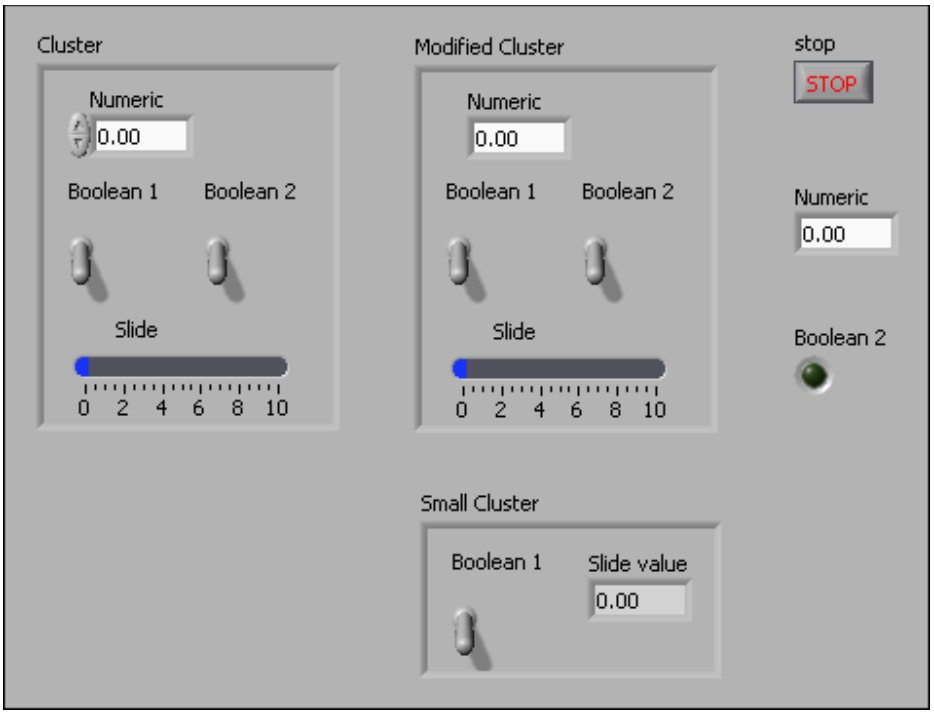


Figure 3

1. Place a `stop` button, located on the `Controls>>Buttons & Switches` palette, a numeric indicator, located on the `Controls>>Numeric Indicators` palette, and a round LED, located on the `Controls>>LEDs` palette, on the front panel.
2. Place a `cluster`, located on the `Controls>>All Controls>>Array & Cluster` palette, on the front panel.
3. Place a `numeric control`, located on the `Controls>>Numeric Controls` palette, two vertical toggle switches, located on the `Controls>>Buttons & Switches` palette, and a horizontal fill slide, located on the `Controls>>Numeric Controls` palette, in the cluster.
4. Create the `Modified Cluster` by duplicating the first cluster and relabeling it. Right-click the shell of `Modified Cluster`, and select `Change to Indicator` from the shortcut menu.

5. Copy `Modified Cluster` and relabel it to create `Small Cluster`. Remove the second toggle switch and horizontal fill slide indicators. Relabel the `numeric indicator` to `Slide value`. Resize the `cluster` as shown in Figure 3.

2. Verify the cluster order of `Cluster` and `Small Cluster`. `Modified Cluster` should have the same order as `Cluster`.
    1. Right-click the boundary of each `cluster` and select `Reorder Controls in Cluster` from the shortcut menu.
    2. Confirm the cluster orders.

Block Diagram

1. Build the block diagram in Figure 4.

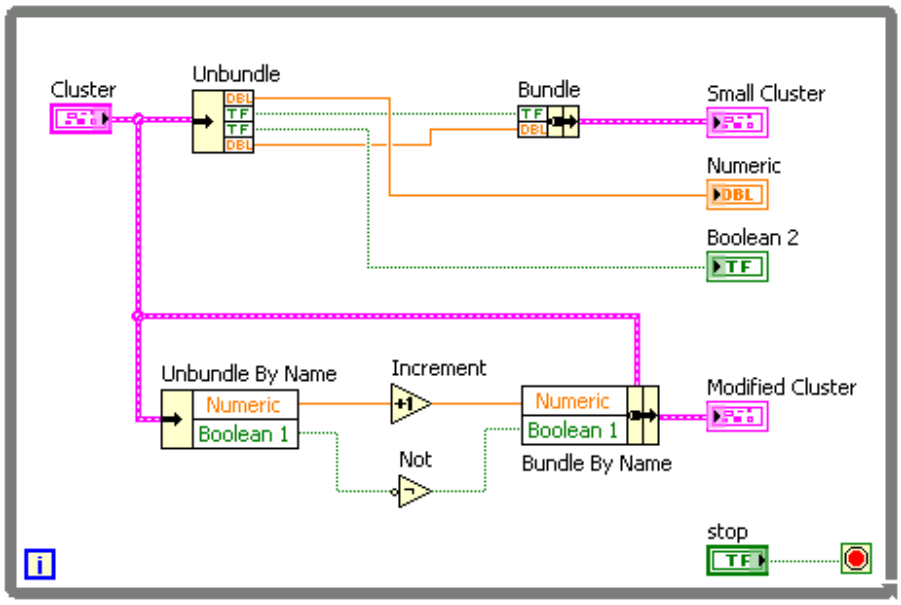


Figure 4

1. Place the `While Loop`, located on the `Functions>>All Functions>>Structures` palette, on the block diagram.
2. Place the `Unbundle` function, located on the `Functions>>All Functions>>Cluster` palette, on the block diagram. This function disassembles `Cluster`. Wire the input cluster to resize the function automatically.
3. Place the `Bundle` function, located on the `Functions>>All Functions>>Cluster` palette, on the block diagram. This function assembles `Small Cluster`.
4. Place the `Unbundle by Name` function, located on the `Functions>>All Functions>>Cluster` palette, on the block diagram. This function returns two elements from `Cluster`. Resize this function to have two output terminals. If a label name is not correct, right-click the name and select the correct name from the `Select Item` shortcut menu.
5. Place the `Increment` function, located on the `Functions>>All Functions>>Numeric` palette, on the block diagram. This function adds one to the value of `Numeric`.
6. Place the `Not` function, located on the `Functions>>Arithmetic & Comparison>>Express Boolean` palette, on the block diagram. This function returns the logical opposite of the value of the `Boolean` terminal of the `Unbundle by Name` function.

7. 

Place the `Bundle by Name` function, located on the `Functions>>All Functions>>Cluster` palette, on the block diagram. This function replaces the values of `Numeric` and `Boolean` 1 in `Cluster` and creates `Modified Cluster`. Resize this function to have two input terminals. If a label name is not correct, right-click the name and select the correct name from the `Select Item` shortcut menu.
8. Complete the block diagram and wire the objects as shown in Figure 4.

2. Save the VI as `Cluster Exercise.vi`

Run the VI

1. Display the front panel and run the VI.
2. Enter different values in `Cluster` and run the VI again. Notice how values entered in `Cluster` affect the `Modified Cluster` and `Small Cluster` indicators. Is this the behavior you expected?
3. Try changing the cluster order of `Modified Cluster`. Run the VI. How did the changed order affect the behavior?
4. Close the VI. Do not save changes.

EXPERIMENT # 5 (Part-1):
Using Compact Field Point Controllers, I/O Modules

OBJECTIVE:

To study the Compact FieldPoint Controllers, I/O modules, accessories and related Software.

Part 1. Discuss the cFP-1808 Controller and the installed I/O modules

a. How I/O Modules, Controllers and accessories are installed on the backplane.

b. How MAX (Measurement and Automation Explorer) is used to configure the Hardware.

c. Understand the configuration file IAK. (Understand the terms “Comm Resource Name”, “Device Name”, Item Name“

NOTE:

See **Appendix A and B** for equipment list for **cFP-1808** and **cFP-2120** Controllers and related IO Modules respectively.

See **Appendix C** for understanding **IAK** file configuration.

See **Appendix D** for **Background information** on using Compact Field Point Controllers, software and related configuration.

EXPERIMENT # 5 (Part-2):
Using Compact Field Point Controllers, I/O Modules

Part 2. Explore the FieldPoint Analog Output Module FP-AO-200 with help of example fp_ao_2xx.vi

The vi of the example is available on line. Students can download the file from the NI web site.

Details of the example are given on next page (page-26).

FieldPoint Analog Output Example - FP-AO-2xx Modules (LabVIEW 6.1 +)

Requirements

Filename: fp_ao_2xx.vi

Software Requirements: LabVIEW Full Development System 6.1 or later

Hardware Requirements: FieldPoint

You can use this VI to continuously output eight analog voltage or current signals on the FP-AO-200/210. It is a very basic VI that calls four main FieldPoint subVIs.

FP Open.vi
FP Create Tag.vi
FP Write (Polymorphic).vi
FP Close.vi

These FieldPoint VIs are installed with FieldPoint.

INSTRUCTIONS:
1. Select the FP module to use.
2. Specify the "IAK File Path", or leave it blank and the VI will use the most recently used IAK file.
3. Edit the "Comm Resource Name", "Device Name", and "Item Name" inputs to match the IAK file configuration that you created in FieldPoint Explorer.
4. Close FieldPoint Explorer.
5. Run this VI.

NOTE: This VI is a combination of two example VIs shipped with FieldPoint. These two VIs, FP-AO-200.vi and FP-AO-210.vi, can be found on your local hard drive at: C:\Program Files\National Instruments\LabVIEW\examples\FieldPoint\Analog Out

EXPERIMENT # 5 (Part-3):
Using Compact Field Point Controllers, I/O Modules

Part 3. Setup the Real Time Controller Workbench

a. Do the hardware setup using cFP-2120, I/O modules, Power Supply, Backplane, Ethernet Cross Cable etc.

b. Do the Software Configure of this Hardware Setup

   i. Install LabVIEW 8.5 (if not installed)

   ii. Install FieldPoint 6.0.1 Software (if not installed)

   iii. Use MAX to make the configuration

Note: See **Appendix E** for specifications of cFP-2120 Controller.

EXPERIMENT # 5 (Part-4):
Using Compact Field Point Controllers, I/O Modules

Part 4. Self Project

Build a complete example of networked instrumentation by sending and collecting data from one of the Field-Point IO Modules. Students can use examples from National Instrument Development Center to demonstrate the working of their Hardware Setup which is configured in previous steps.

APPENDIX A-1

List of Compact FieldPoint (cFP-1808) Equipment

OTUPUT MODULES

- cFP-AO-200
  - Analog Output module (mA)
  - External power supply required
  - Channels 8
  - Qty.

- cFP-RLY-423
  - Relay Output Module (AC/DC Switching)
  - 1.5 Amps/channel
  - Uses 1 Amp or more from cFP backplane
  - Channels 4
  - Qty.

INPUT MODULES

- cFP-RTD-122
  - 16 Bit RTD Input Module (RTD, Ohms)
  - Channels 8
  - Qty. ?

- cFP-AI-110
  - 16 Bit Analog Input Module (mA, mV, V)
  - Channels 8
  - Qty.

CONTROLLER

- cFP-1808
  - LabVIEW Ethernet Network Controller
  - Serial Port
  - Qty. 2

BACKPLANE

- cFP-BP-8
  - 8-Slot Backplane
  - Qty. 2

CONNECTOR BLOCK

- cFP-CB-1
  - Connector Block
  - Qty.

Note: **POWER SUPPLY** 24 Volts-DC from Rack

APPENDIX A-2

List of Compact FieldPoint (cFP-21xx) Equipment

OTUPUT MODULES

- cFP-AO-200
  - Analog Output module (mA)
  - External power supply required
  - Channels 8
  - Qty. 2

- cFP-RLY-423
  - Relay Output Module (AC/DC Switching)
  - 1.5 Amps/channel
  - Uses 1 Amp or more from cFP backplane
  - Channels 4
  - Qty. 2

- cFP-RLY-425
  - Relay Output Module (AC/DC switching)
  - Channels 8
  - Qty. 2

- cFP-PWM-520
  - Pulse Width Modulation Output Module
  - External power supply required
  - Channels 8
  - Qty. 1

INPUT MODULES

- cFP-RTD-122
  - 16 Bit RTD Input Module (RTD, Ohms)
  - Channels 8
  - Qty. 2

- cFP-AI-110
  - 16 Bit Analog Input Module (mA, mV, V)
  - Channels 8
  - Qty. 2

- cFP-TC-120
  - 16 Bit Thermocouple Input Module (TC, mV)
  - Channels 8
  - Qty. 2

- cFP-QUAD-510
  - Quadrature Encoder Input Module
  - 4 Axis
  - External power supply optional
  - Uses 1 Amp or more from cFP Backplane
  - Qty. 1

- cFP-DI-330
  - Digital Input Module (AC/DC V sink/source)
  - Channels 8
  - Qty. 2

CONTROLLER

- cFP-2120
  - LabVIEW Real-Time/Ethernet Network Controller
  - 4 Serial Ports
  - Extended Memory (128MB)
  - Qty. 2

  (Note: Removable Compact Flash is an optional, so we don't have it)

BACKPLANE

- cFP-BP-8
  - 8-Slot Backplane
  - Qty. 2

CONNECTOR BLOCK

- cFP-CB-1
  - Connector Block
  - Qty. 12

- cFP-CB-3
  - Isothermal Connector Block
  - Qty. 2

POWER SUPPLY

- PS-5
  - Power Supply 24 Volts-DC, 5 Amp
  - Universal Power Input Easy way to distribute power from one power supply to up to 8 devices with replaceable blade fuses
  - Qty. 2

APPENDIX C

What Does Current IAK File Mean When Using FieldPoint?

Primary Software: Driver Software>>NI-FieldPoint
Primary Software Version: 3.0.2
Primary Software Fixed Version: N/A
Secondary Software: N/A

Problem:
In several places I have seen references to Current, Active or Last IAK file when using FieldPoint. How exactly is this determined?

Solution:
The **Current IAK** file is used by FP Open when a specific IAK file path has not been specified. This applies to FP Open in both LabVIEW and CVI. Also FieldPoint OPC uses the Current IAK file to determine the items that are available through OPC.

For LabVIEW 7.0 and later:

An IAK file becomes current by:

1. Saving the IAK file in Measurement and Automation Explorer (MAX) OR
2. Opening an IAK file in MAX.

In order for these changes to take place in LabVIEW, you must close and reopen LabVIEW.

For LabVIEW 6.1 and earlier:

An IAK file becomes current by:

1. Saving the IAK file in FieldPoint Explorer AND
2. Closing FieldPoint Explorer.

An IAK file does not become current by:

1. Opening an IAK file and keeping it open while closing FieldPoint Explorer.
2. Saving an IAK file in FieldPoint Explorer without closing FieldPoint Explorer. This allows you to open and view an IAK file with out affecting which IAK is the current. If you want to edit an IAK file with out making it the current IAK file:
    1. Save it without closing FieldPoint Explorer.
    2. Before you close FieldPoint Explorer reopen the correct current IAK file and save it.

Note: When using FP Open it is easiest to specify the IAK file path. This will guarantee that the correct IAK file is opened. If you leave the path unspecified, your program may stop working if the current IAK is changed through the procedure above.

APPENDIX D
Background Notes on FieldPoint

James Trevelyan, July 2004.

These notes have been written for students wishing to use the FieldPoint facilities in the mechatronics laboratory.

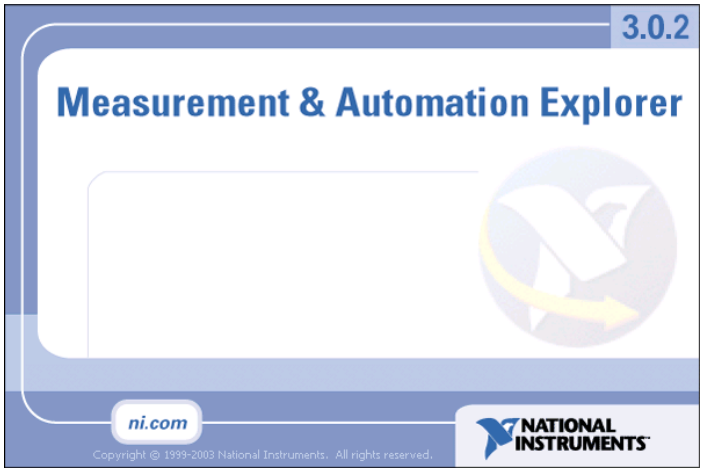
FieldPoint is a proprietary method for interfacing devices to computers developed by National Instruments but it is very similar in principle to the concept of a fieldbus interfacing method used by many process control equipment suppliers.

The idea of fieldbus grew out of the problem of interfacing hundreds or thousands of sensors and actuators to PLCs and process control computers in large industrial plants. Rather than connect each sensor or actuator to a central plant computer, requiring hundreds or thousands of kilometres of wiring, the idea of fieldbus was to connect related groups of sensors and actuators to a local microcomputer that communicates with the central plant computer via an ethernet local area network (LAN). Earlier fieldbus units used serial interface lines for communication of the principal remained the same. The result was an enormous reduction in wiring and a corresponding increase in reliability.

For a more detailed background on industrial process communications read the Mechatronics Design 310 notes "Industrial Process Communications".

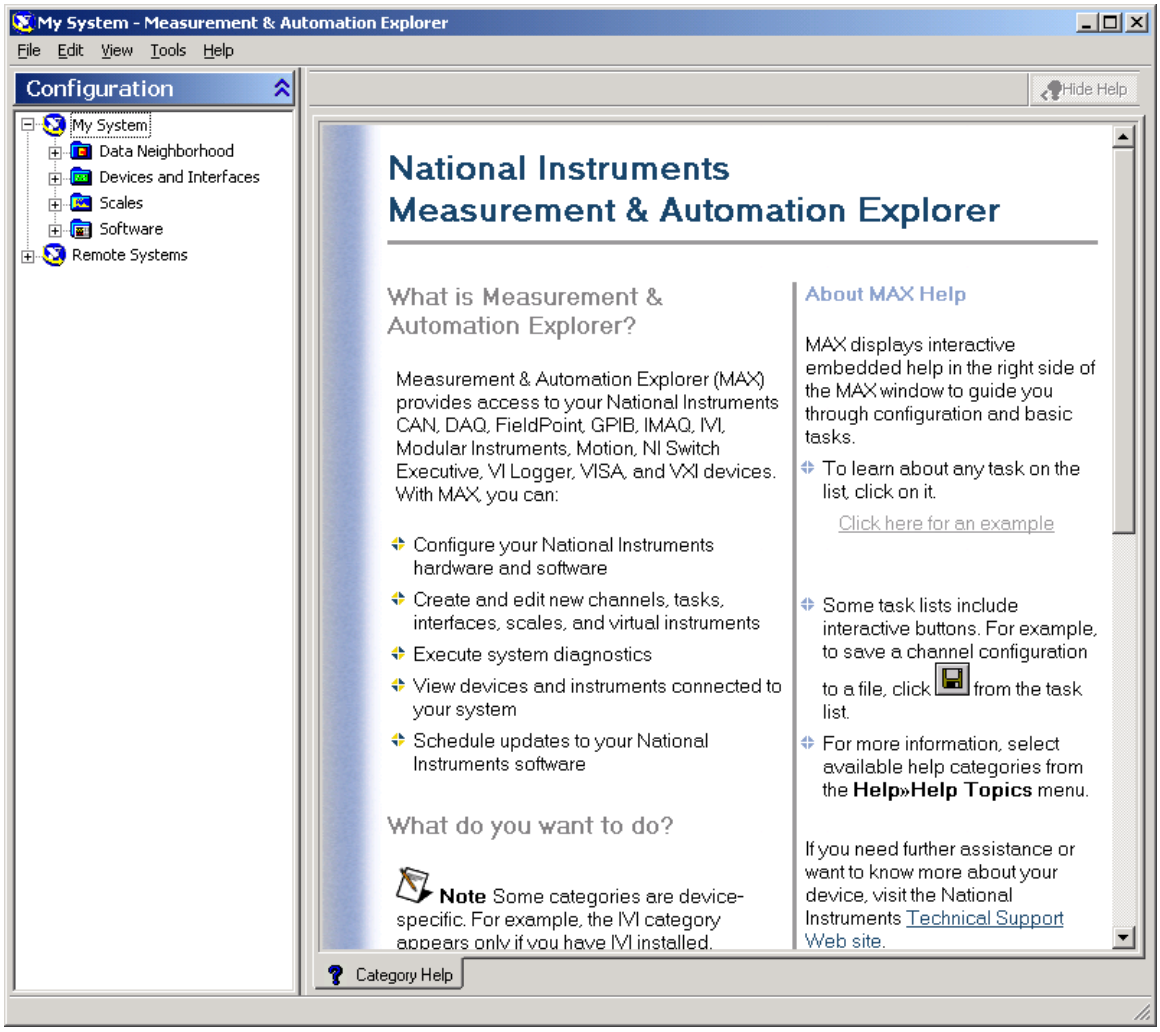
These background notes are based on my own understanding at the time of writing and may not be 100% correct. Please tell me about any errors or mistakes and feel free to contribute further material for these notes.
What are the hardware and software components that you need to know about?

MAX: (Measurement and Automation Explorer)



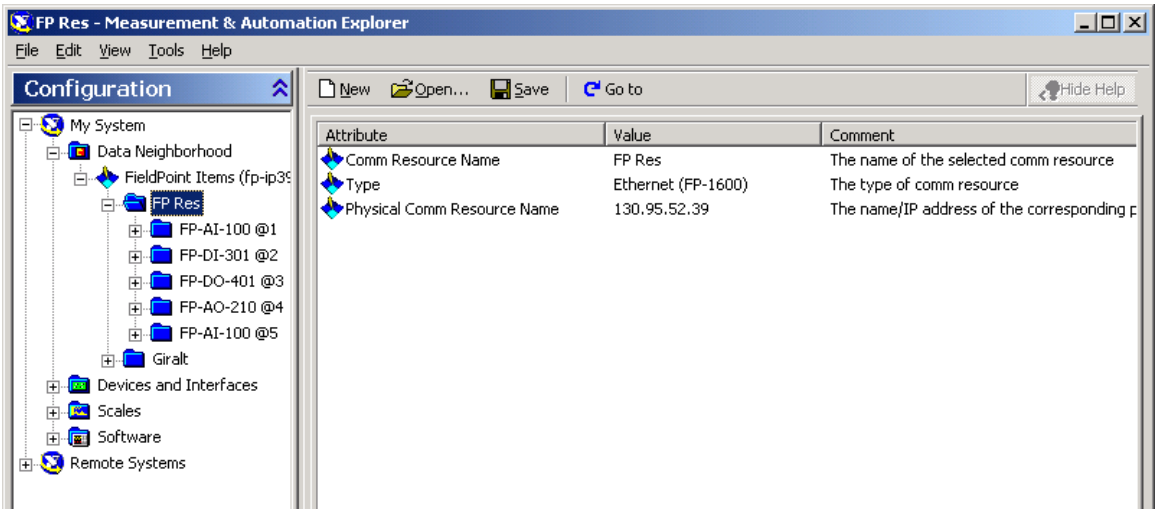
MAX should first be used on a computer in the mechatronics laboratory so that you can access the FieldPoint hardware. Once you are familiar with this application you

will be able to locate FieldPoint units, read data from input devices and send data output devices. You will also learn how to configure individual FieldPoint units but before you do that you need to understand a little bit more about the other components in the system.



This is the opening screen for MAX. The right hand side panel provides help. The help information provided is very extensive.

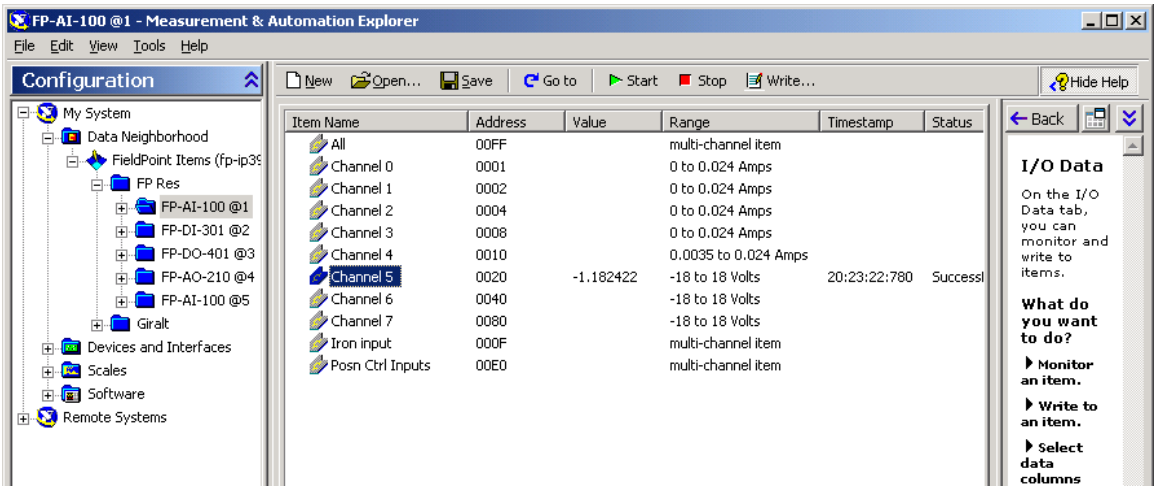
The function key F5 tells MAX to search for interface devices in its neighbourhood including devices directly connected to the computer on which is running. They show up in the configuration tree in the left hand window. Use F5 again after powering up remote devices that did not show on the first attempt.



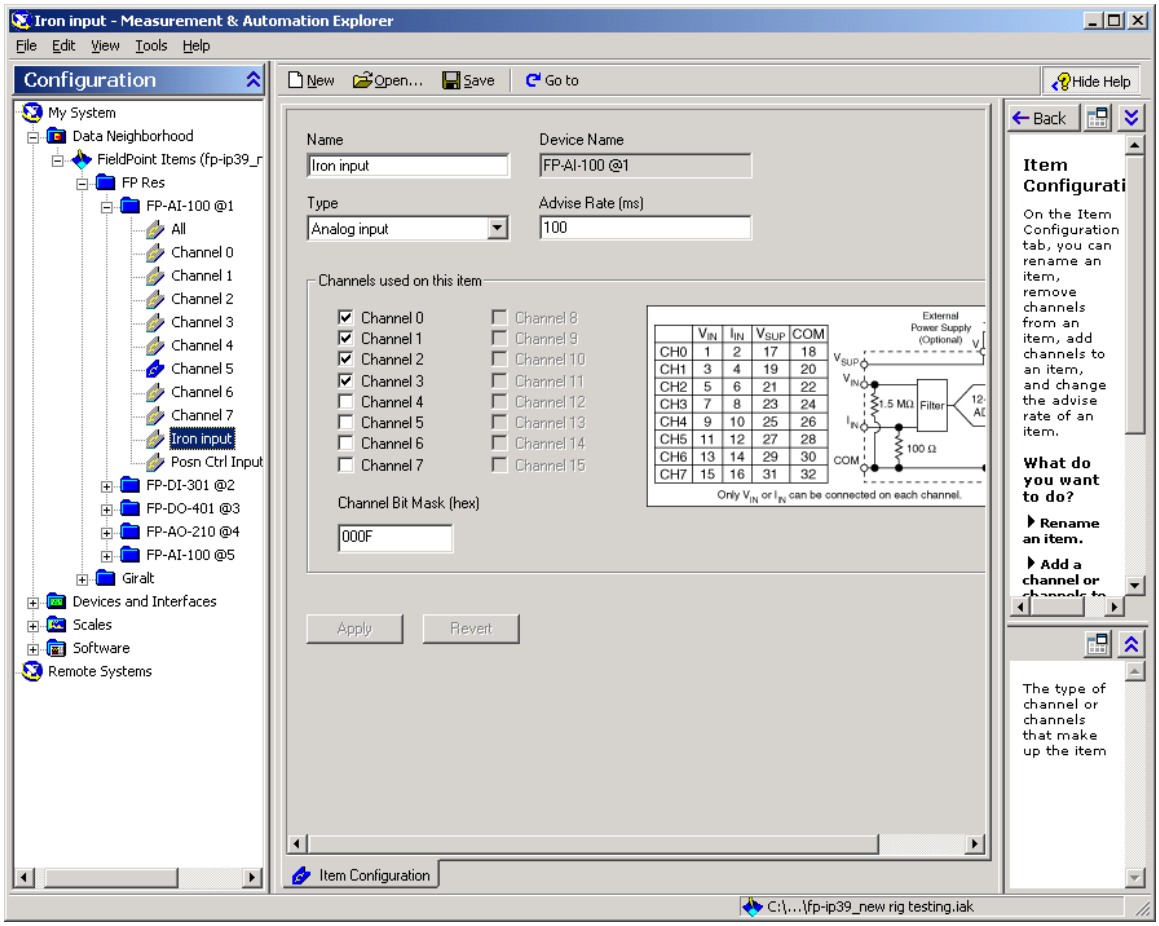
This shows the top half of the screen after expanding the 'Data Neighbourhood' item to reveal field point items. We can see 'FP Res' – this is the field point unit (default name) currently installed in G.19 at IP address 130.95.52.39. This unit is connected to the lab computers through a local 100 Mbps switch unit: this way the data exchanged between the lab computers and the field point unit does not flood the main building network with IP data packets.

'Giralt' is another field point unit. Note how we have expanded the 'FP Res' unit to show each of the interface modules:

FP-AI-100 @1 (Analog input – 8 channels)
FP-DI-301 @2 (Discrete input – 16 bits)
FP-DO-401 @3 (Discrete output – 16 bits)
FP-AO-210 @4 (Analog output – 8 channels)
FP-AI-100 @5 (Analog input – 8 channels)



Here we have selected one of the units to display the 'data items' in the main window. We clicked on 'Channel 5' and the green 'Start' control above to make MAX start reading the data from 'Channel 5'. Relevant help is displayed on the right hand side.



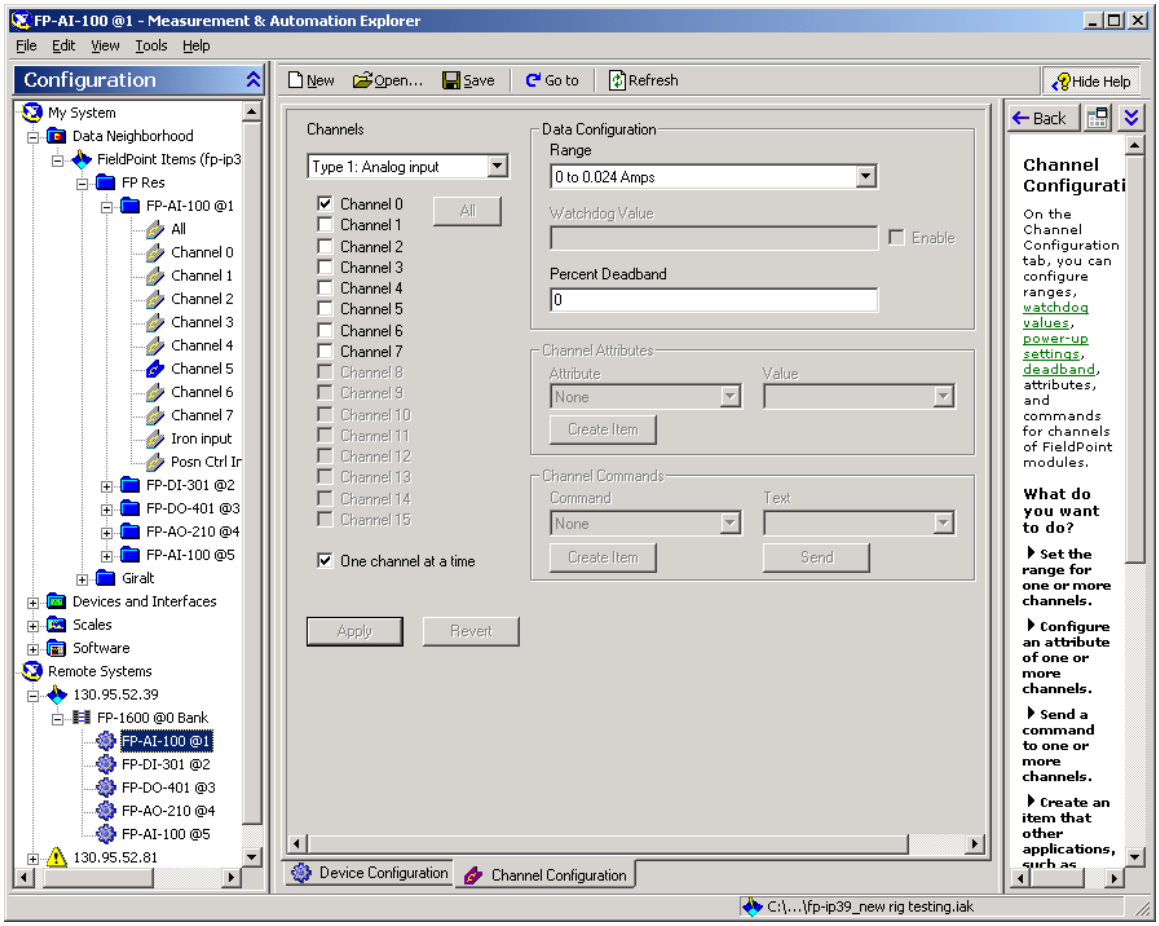
Here we have expanded the 'FP-AI-100 @1' device and selected the 'Iron input' item. This item has been defined (by us) to represent as a single item all 4 analog input channels used for the electric iron temperature sensors. We could change the channel assignment without having to reconfigure the software, just by clicking different channels. Notice also how basic documentation is displayed. The signals can be obtained by a single 'FP Read' function call in an array of SGL voltage values (see later sections).

This configuration information is stored in a standard configuration file.

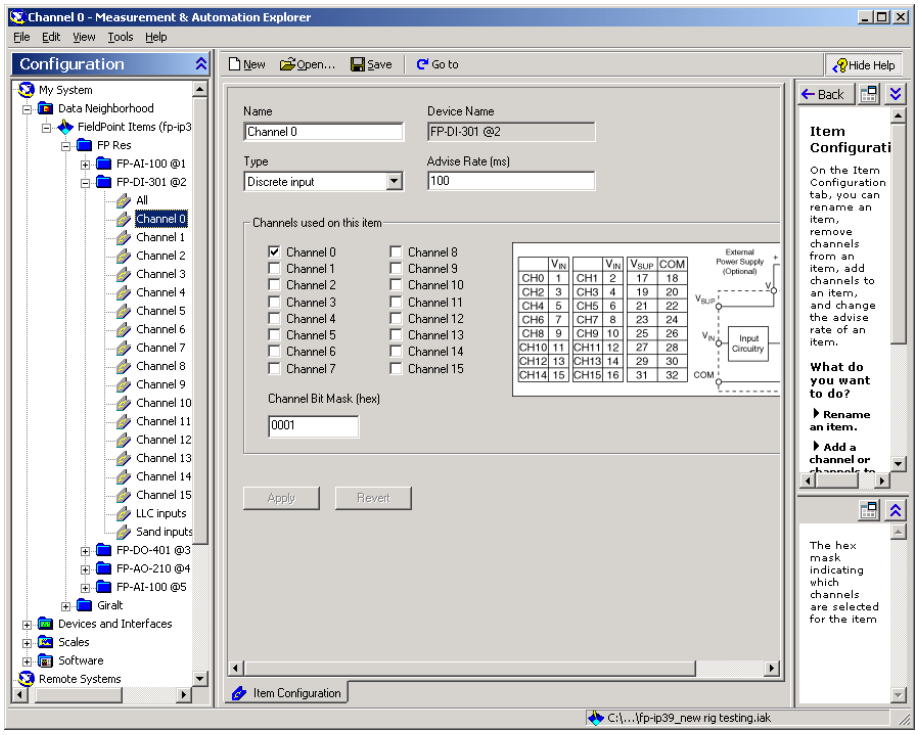
> In our case the configuration file is **C:\Icon\LOL Hardware\ip-39.iak –** the file is stored in the same location on every lab computer for consistency, and the file is the same (or supposed to be) on every lab computer. Since FieldPoint is extensively used for Telelabs, we chose to place it in the same folder as hardware clients to simplify programming. The hardware clients normally reside on the computer Joe_Engelberger which is the Telelabs LOL Server computer.

MAX automatically opens the configuration file that was used last time.

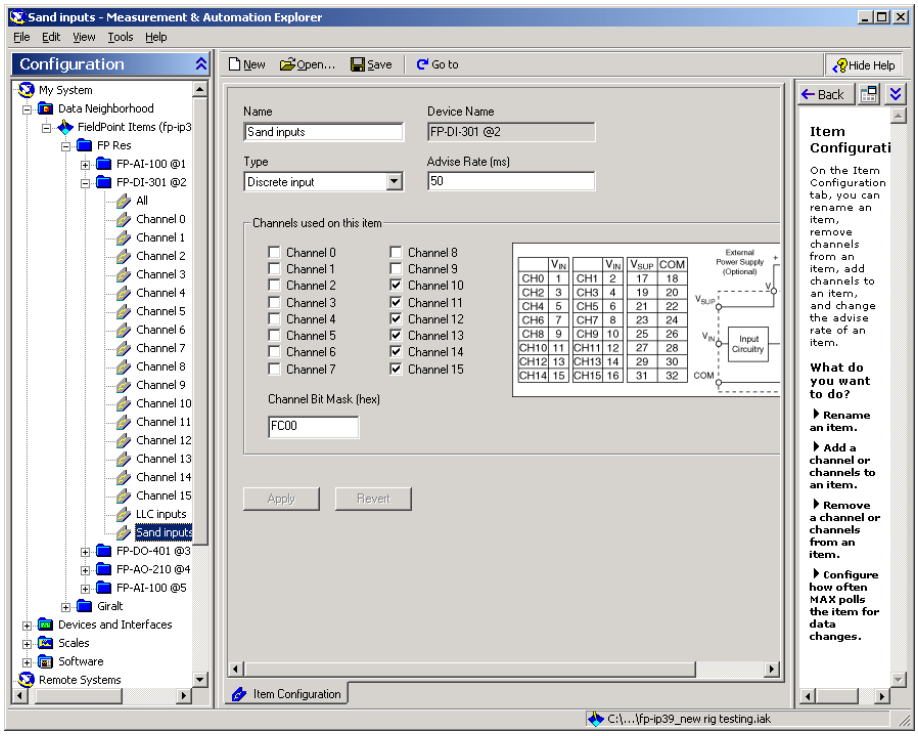
To see how each hardware channel is configured, right click on the channel item and choose ‘Go To Device Configuration’. Now the window will look like this:



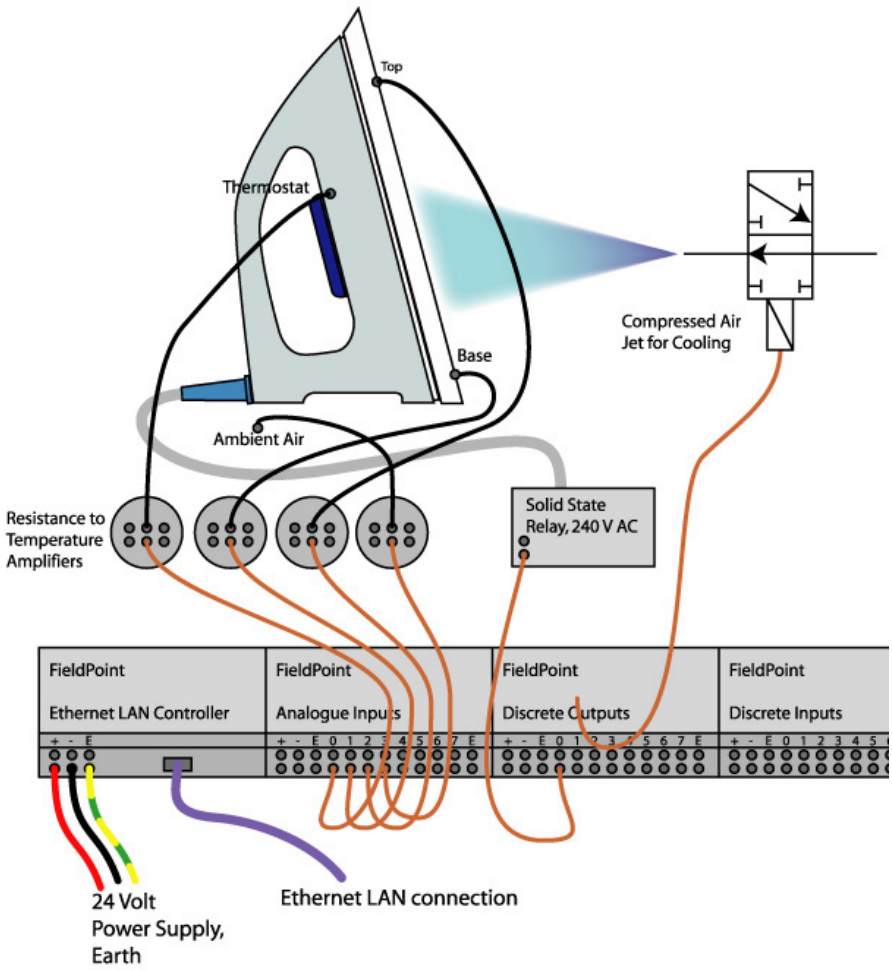
Notice how the 'Remote Systems' part of the tree has opened up in the left hand panel. That is because we are now configuring hardware rather than just looking at the data received from the hardware (or to be sent to it). On this screen we can set the type of analogue input (Voltage or current, voltage range etc.)



Here we are back in the 'Data Neighbourhood' looking at 'Channel 0' discrete input device. Notice how there is an item configured for the sand weighing machine discrete inputs so they are handled together – see below.



FieldPoint Interface Hardware



This diagram shows the arrangement of the electric iron and FieldPoint interface unit installed in the mechatronics laboratory.

On the left-hand side of the FieldPoint unit areas the ethernet LAN controller. This is a microcomputer that communicates with your computer and other computers in the laboratory. It also controls the way that data is received from and send to input and output devices connected to the FieldPoint bus that runs along the backplane to the right of the controller module.

There are several input and output devices plugged into the FieldPoint bus. They are numbered from left to the right: 1, 2, 3 etc. In the diagram above 1 is an eight channel analog input device, 2 is an eight channel discrete output device and 3 is an eight channel discrete input device. This is not necessarily the way the different devices are arranged in the lab – it is only for illustrative purposes.

One of the nice characteristics of FieldPoint is that the interface devices can be unplugged and replaced with other devices at any time without even turning off the power. (*Note, however, it is a routine and basic precaution that power is normally turned off when making or breaking any electrical connections!*) The reason for this

"hot connect and disconnect" capability is that individual FieldPoint units can be replaced while a control system using other adjacent units is running.

Note that in the diagram only single wires shown connecting the FieldPoint units to the instrument devices themselves: in practice there are always two wires -- a signal wire and a common earth connection. However, it is possible for a single common earth connection to be used with several signal wires, particularly when the wires are short and electromagnetic interference is not a major issue.

FieldPoint OPC Server

This is an invisible element of software that is invoked whenever you set up the FieldPoint connection with your computer. It is important to understand that communication with the FieldPoint unit does not necessarily occur at the precise instant when your application instructs it to happen. Instead, communication occurs both as a direct result of requests from your program and also as a result of the way that you configure the FieldPoint controller. For example, you can configure the FieldPoint controller to notify the computer only when the value of an input signal has changed by more than a certain amount. Or alternatively you can configure the FieldPoint controller to notify the computer only at certain time intervals. These capabilities are useful when you want to limit the amount of communication that takes place between the FieldPoint controller and the computer.

The notion of the server on your own computer is also handy when you have several different applications running on your computer, all of which need to access the FieldPoint unit for some purpose or other.

This concept also enables several different computers to access the same signals on the FieldPoint unit. However, there are several possible conflicts that can occur when you do this. At the time of writing I am not sure to what extent the system prevents these conflicts from occurring. It is possible for different computers to specify different measurement ranges for input signals. For example, one computer could specify that an input signal has a range from -10 volts to +10 volts and another computer could specify the same signal having a range from zero to +2 volts. One computer could specify that signal values should only be notified after a change of 0.5 volts and another computer could specify that signal values should be notified continuously.

Configuration information for every input and output signal is stored in a configuration data file (~.iak) – see previous discussion of MAX. When an application using FieldPoint first starts running it reads this configuration data file. Configuration data tells the application what kind of devices are available and where they are located on the network. In an industrial situation this is extremely valuable because it is possible to make hardware changes without having to change software that uses that hardware.

To try and avoid conflicts of the kind that I have just described in the mechatronics laboratory we have used a single configuration data file for the FieldPoint units. Until recently we had only one FieldPoint unit and we used the configuration file:

C:\Icon\LOL Hardware\fp-ip39.iak

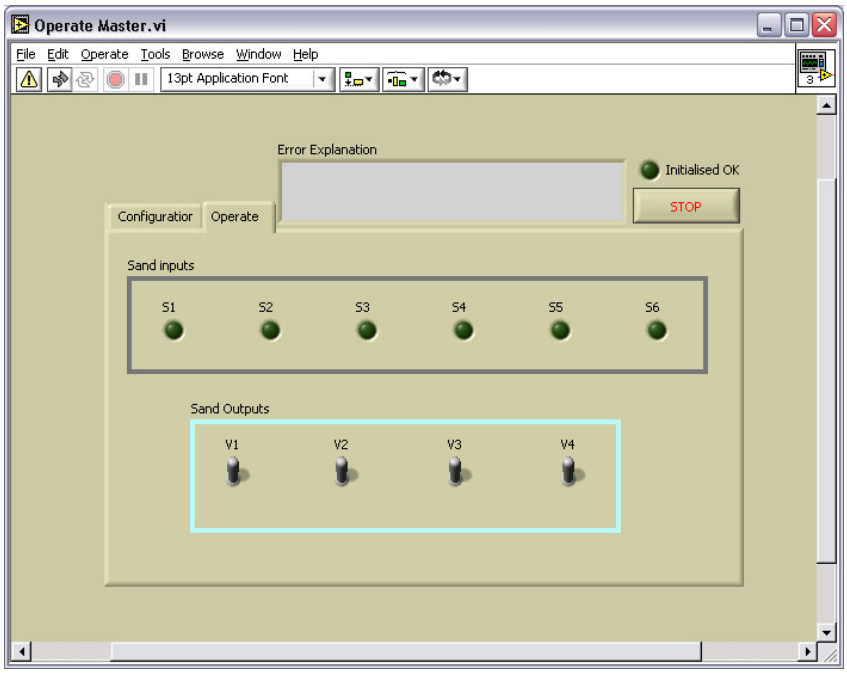
This configuration data file was installed on every computer in the laboratory to reduce the chance of confusion between different computers and different applications.

While you are experimenting with different input and output devices on one particular computer it is quite OK for you change the configuration data file. You can access the configuration data file from MAX.

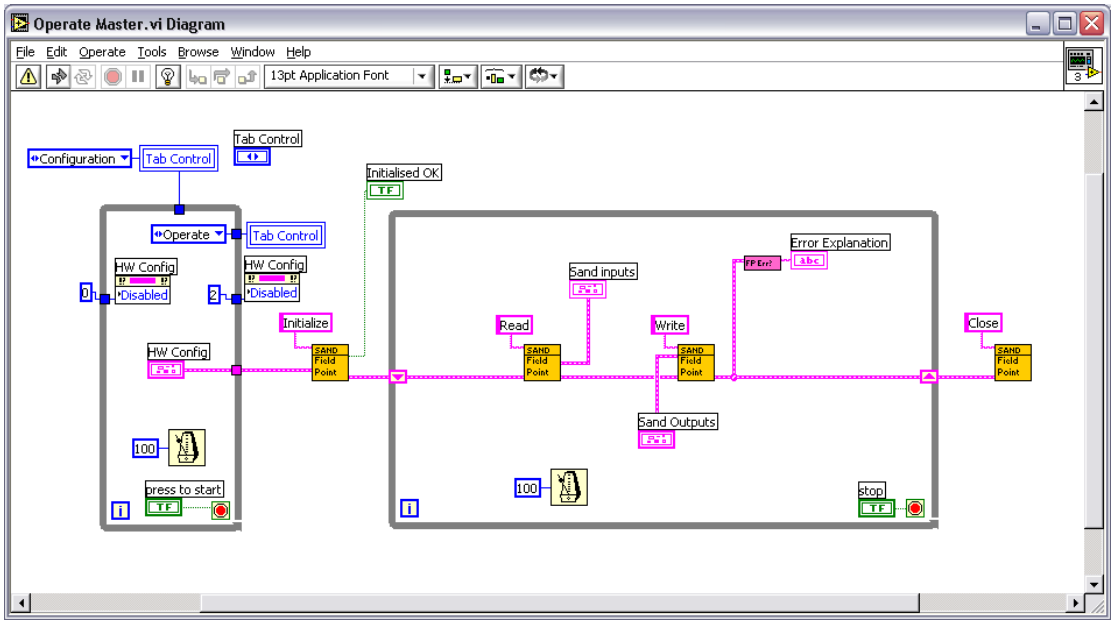
LabVIEW FieldPoint Handler

Any LabVIEW application that uses FieldPoint can be conveniently structured by incorporating all FieldPoint operations into a single module -- a FieldPoint Handler. This module performs four different operations: initialisation, close, read, and write.

I supply a prototype application "Operate Master" to enable students to start working with the sand weighing machine that illustrates how a FieldPoint Handler can dramatically simplify the code.

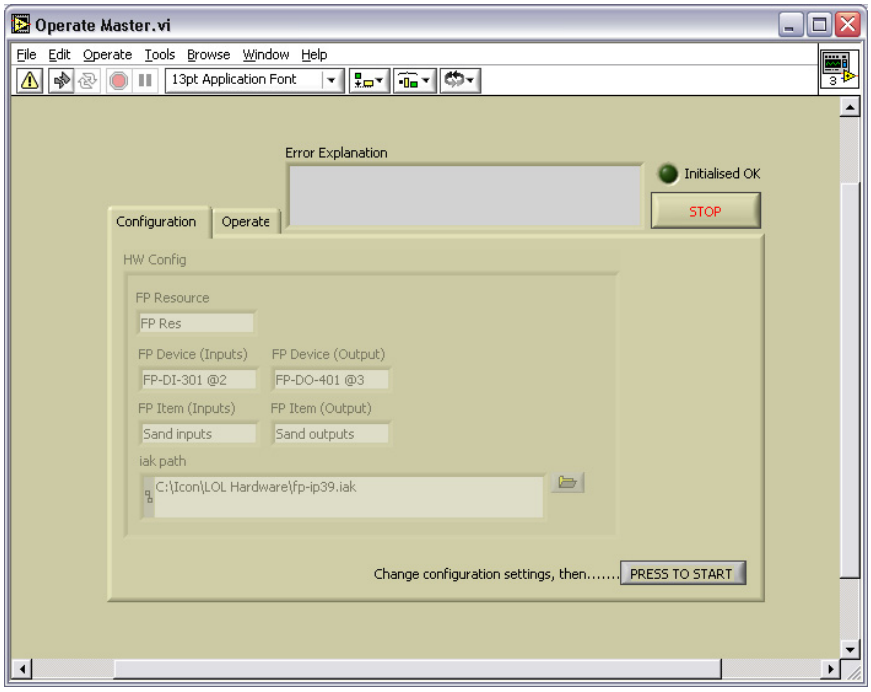


The front panel, shown above, incorporates two clusters: "sand outputs" consists of four boolean values which are transmitted to the sand weighing machine and "sand inputs" consists of six boolean values from six sensors mounted on the machine.



The main part of the diagram is an endless loop that consists of a "read" operation that obtains values from the sand inputs and displays them on front panel followed by a "write" operation that takes output values from switch settings on the front panel and transmits them to the sand weighing machine.

The loop at the left-hand side of the diagram is used only at startup. It allows hardware configuration settings to be changed before pressing the "start" button. It is also arranged so that the configuration panel (shown below) is displayed by default when the program starts.



The cluster "HW Config" contains the FieldPoint configuration information obtained from MAX. For example, the particular FieldPoint unit to be used (FP Resource) is named "FP Res": this is the name of the FieldPoint unit located near the electric iron.

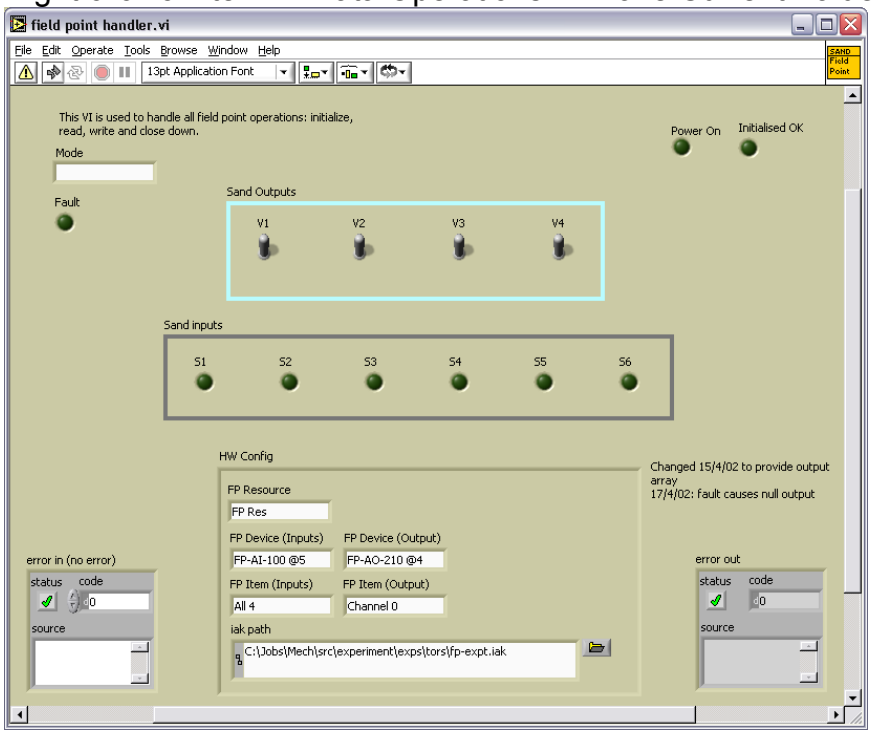
The discrete input device FP-DI-301@2 means an eight channel digital input device located at position 2 on the field point bus. The group of signals to be read (known as an 'item') is called "Sand inputs". Note how more than one signal can be grouped into an item.

If you use MAX to examine the FieldPoint configuration file you will be able to see which particular channels correspond to the items "Sand inputs" and "Sand outputs".

Note that it is very important that the resource, device, and item names are exactly copied from MAX.

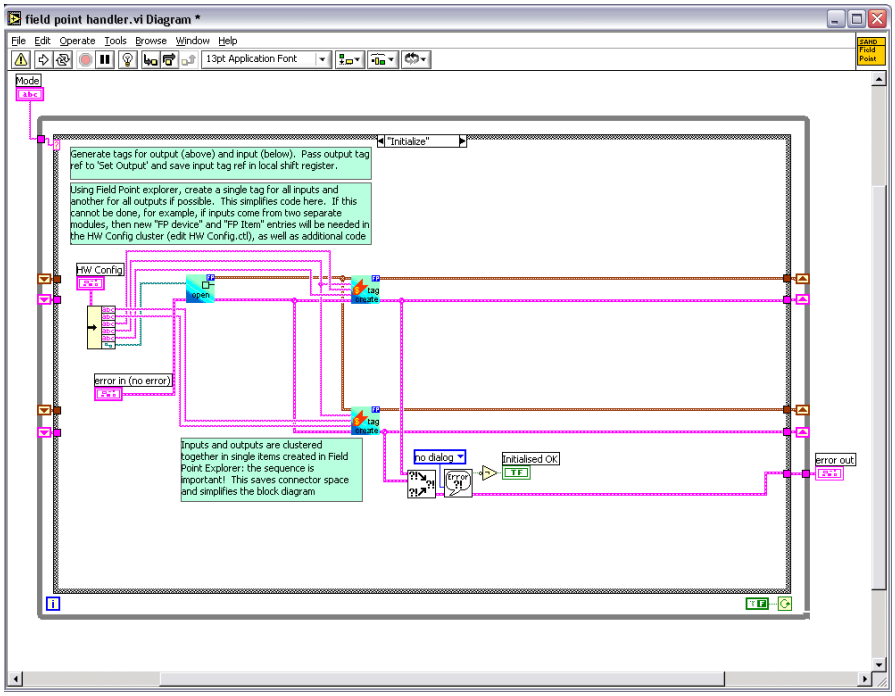
After changing the "HW config" entries remember to change the default values for the panel:

Right click on item -> Data Operations -> Make Current Value Default



Now we will look closer at the FieldPoint Handler itself: the front panel is shown in the picture above.

We can see the "sand outputs", "sand inputs" and "HW Config" clusters on the front panel.



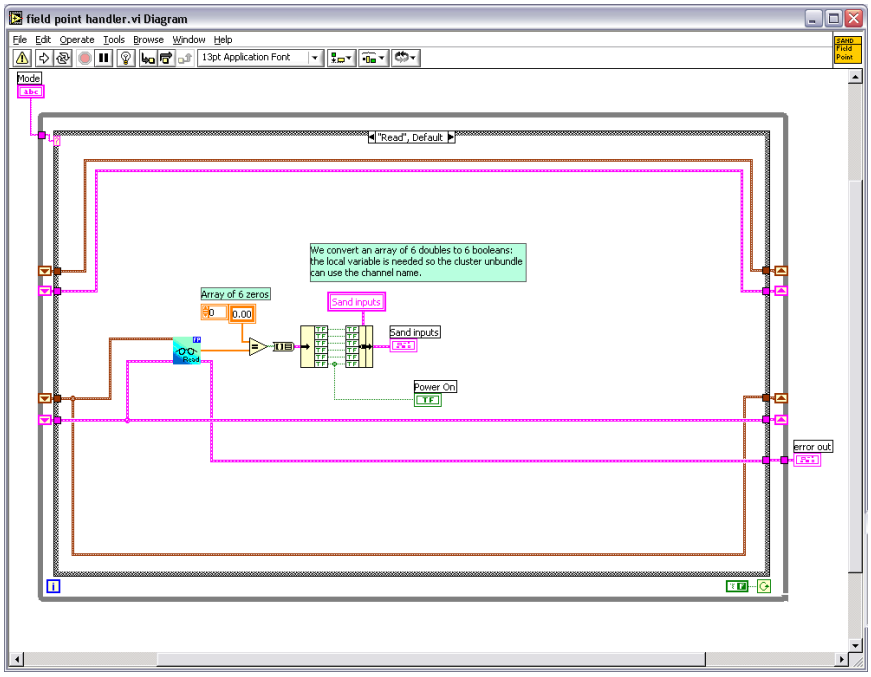
The internal structure of the FieldPoint Handler module is identical to the structure of an uninitialised shift register module: a once only execution while loop enclosing a case structure. Shift registers on the while loop boundary provide internal storage that persist between separate executions of the FieldPoint Handler module.

Here we see the initialise case. Note that the icons representing the FieldPoint functions called in this case are different from the real FieldPoint module icons that you will find on a computer with FieldPoint installed. I prepared these notes on my laptop computer that does not have FieldPoint installed. Instead I have a series of dummy FieldPoint modules with these characteristic blue icons so that I can wire up FieldPoint LabVIEW code on a computer without FieldPoint installed. (Make sure that these dummy FieldPoint modules are never installed on a computer that has a FieldPoint software installed on it.)

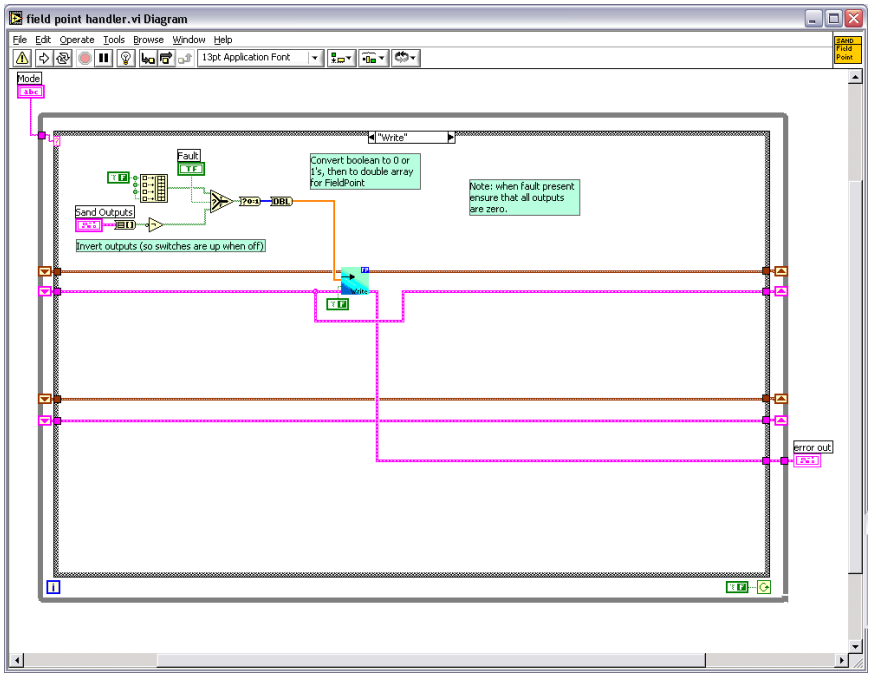
It is best to examine this code on a computer with FieldPoint installed and read FieldPoint application notes from the National Instruments web site for further technical background.

The first module called is "FP open". This starts the FieldPoint OPC server on the local computer and generates a cluster of FieldPoint tags that must be passed to every other FieldPoint module called in the application.

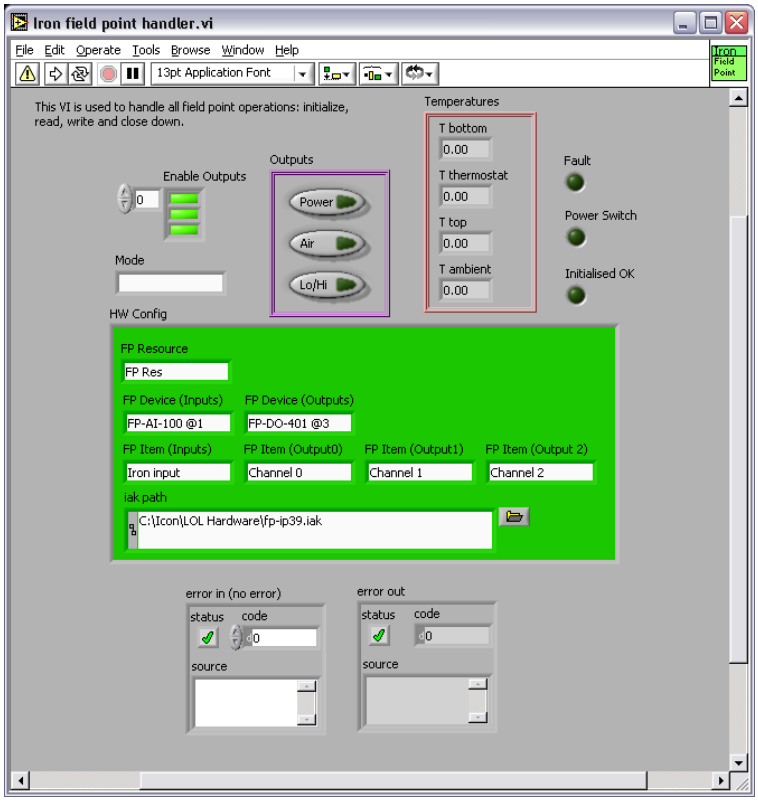
The next calls are to "FP tag create". This call sets up tag values for each of the items specified in the hardware configuration cluster.



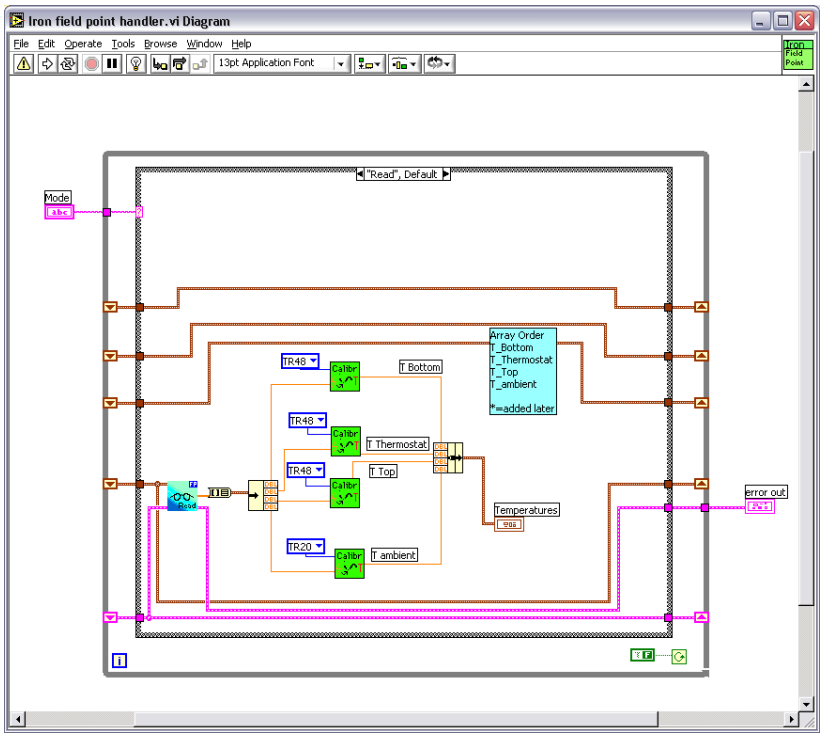
Here we see the "read" case. Note that the item values emerge from the "FP read" module in an array of double precision real values -- these need to be converted before being formed into the output cluster.



Here we see the "write" case. Note that if the boolean value "fault" is true we output zero or "false" values to the interface. This capability is not used in the sample application provided to students but is a wise precaution.

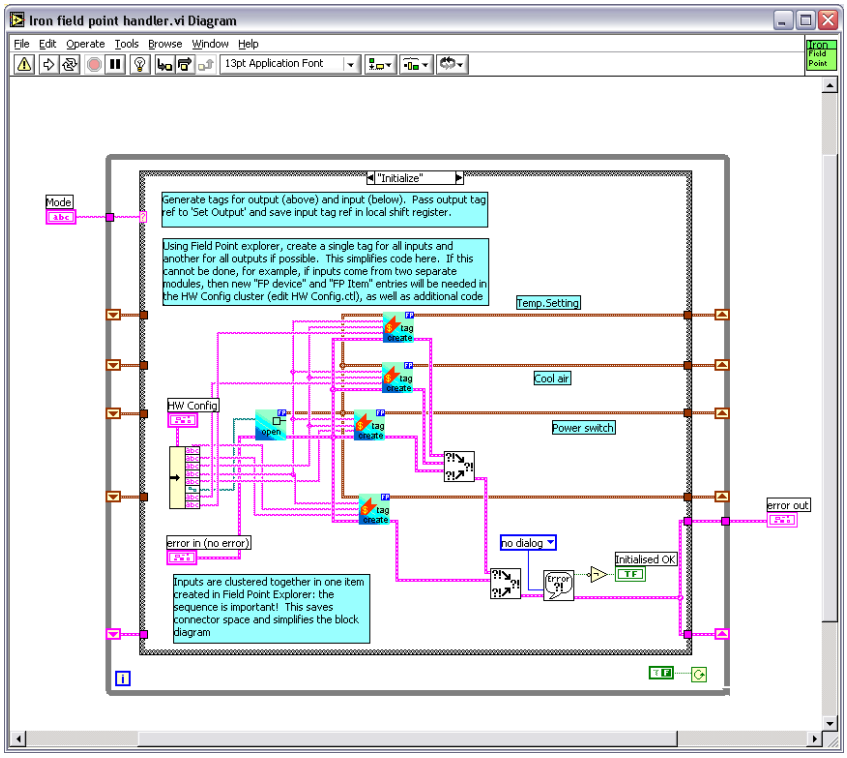


Here we see the front panel for the electric iron FieldPoint Handler. This is structured in almost exactly the same way as the FieldPoint Handler for the sand weighing machine.



Here is the "read" case and the only difference is that an extra calibration operation occurs after reading the input values which are, of course, analog voltages from the resistance to temperature (RTD) sensors. We use a different calibration characteristic

for the ambient air indication because we used a different kind of RTD sensor for the lower ambient temperature range.



Here is the initialise case where we can see that we generate tags for the two boolean output signals separately rather than using a single data item. The only consequence of this is extra calls to "FP tag create" and extra configuration parameters in the "HW config" cluster.

Suggested Development Sequence

From previous experience the best way to start is to adapt one of the existing pieces of code to suit your application. Start with the FieldPoint Handler and create a unique one for your particular application. Create an application similar to "Operate Master" to enable you to operate the equipment manually. Include all the required controls and sensor signal displays in the relevant input and output clusters.

Introduce changes step by step so that you are comfortable with each of the changes and can make sure that each one works.

Note that the clusters for inputs and outputs have been defined using strict typedef control clusters. Refer to the original LabVIEW notes provided for the second year class to understand how these work and how to modify them.

Once you are confident that you have developed a comprehensive FieldPoint Handler for your piece of equipment then you can build it into more complex applications such as a Telelabs hardware client.

Step 1: Define your hardware!

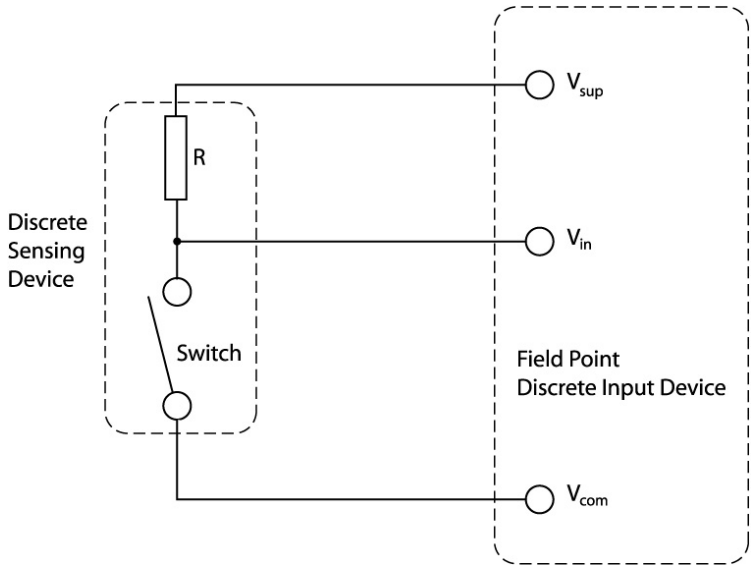
The first step is to design your electronic interface to match fieldpoint capabilities.

Discrete interfacing is almost trivial.

On the output side the discrete channels are either 0 Volts or 24 Volts (nominal) depending on whether the discrete signal is 0 or 1. (Note that the discrete values transmitted to the **FP Write** function are of type SGL array (32 bit floating point array, one value for each signal in an item), regardless of the data type of the hardware device. Thus, to output 24 Volts, send a value of 1.0 to the **FP Write** function.

The current capacity of each output is significant and you may not require an amplifier or 'relay driver' device to power small relays or solenoids. Check the specification sheet first, however. Note also that the output current capacity is limited by the available current from the power supply. It is possible to connect an independent power supply, in fact this would be sensible practice. By doing this, you will ensure that an overload on the output side will not kill the power supply to the FieldPoint controller logic!

On the input side, you should use a conventional industrial circuit arrangement:



Vin is 24 Volts when the switch is open, and 0v when the switch is closed. The value of R should be selected to keep current drain moderate (typically 5 mA) while not so small that the circuit can be affected by electromagnet noise. Vsup is 24 Volts for FieldPoint devices, hence R should be about 4.7 k.

The vast majority of industrial sensors use this simple arrangement.

(to be continued)

Compact FieldPoint LabVIEW Real-Time Controllers with Ethernet

NI cFP-21xx *NEW!*

- Rugged LabVIEW Real-Time controller
  - 188 MHz processor and up to 128 MB of SDRAM
  - Up to 128 MB of nonvolatile storage and 512 MB removable CompactFlash
- Ethernet communication for distributed real-time systems
- Dual redundant 11 to 30 VDC power supply inputs, low power
- Up to 4 serial ports (3 RS232 and 1 RS485) for communication
- Industrial certifications, Class I, Div 2, and -40 to 70 °C

Operating Systems
- Windows XP/2000/NT
- LabVIEW Real-Time

Recommended Software
- LabVIEW
- LabVIEW Real-Time Module
- LabVIEW Datalogging and Supervisory Control Module

Driver Software (included)
- Measurement & Automation Explorer
- OPC server (2.0 compliant)

Controller	SDRAM Memory (MB)	Internal Nonvolatile Storage (MB)	Removable CompactFlash	Ethernet Ports	RS232 Ports	RS485 Ports
cFP-2120	128	128	✓	1	3	1
cFP-2110	128	64	–	1	2	0
cFP-2100	64	64	–	1	1	0

Table 1. cFP-21xx Selection Guide

Overview

National Instruments Compact FieldPoint is a programmable automation controller (PAC) designed for industrial control applications performing advanced embedded control, data logging, and network connectivity. It combines the packaging, specifications, and reliability of a PLC with the software, flexibility, connectivity, and functionality of a PC. Compact FieldPoint is a reliable platform designed for rugged industrial environments with shock, vibration, and temperature extremes.

National Instruments cFP-21xx controllers run LabVIEW Real-Time, providing the functionality, connectivity, and flexibility of NI LabVIEW software on a small industrial platform. The modular I/O architecture with built-in signal conditioning and isolation provides direct connectivity to industrial sensors such as analog voltage, 4 to 20 mA current, thermocouples, RTDs, pressure, strain, flow, pulse-width modulation (PWM), and 24 V digital I/O. You can use NI cFP-21xx controllers in intelligent distributed applications requiring industrial-grade reliability – such as process and discrete control systems – to open and close valves, run control loops, log data on a centralized or local level, perform real-time simulation and analysis, and communicate over serial and Ethernet networks.

System Basics

A single cFP-21xx controller manages a bank of up to eight Compact FieldPoint I/O modules. The controller mounts securely on a metal backplane that provides the communication bus as well as a solid surface for the Compact FieldPoint I/O modules and controller. The system is modular; select the I/O modules and connector blocks or cabling options best suited for your application. Compact FieldPoint I/O banks have a number of features for industrial operation, including 2,300 V transient overvoltage protection, a wide temperature range for operation in extreme environments, backup power supply connections to protect against primary power failure, and hot-swappable modules to simplify maintenance and minimize downtime. cFP-21xx controllers feature an industrial 188 MHz x86 processor that reliably and deterministically executes your LabVIEW Real-Time applications.

Choose from thousands of built-in LabVIEW functions to build your multithreaded embedded system for real-time control, analysis, data logging, and communication. cFP-21xx controllers also offer up to 128 MB of 100 MHz SDRAM and 128 MB of internal nonvolatile storage and a removable CompactFlash slot. All cFP-21xx controllers feature a 10/100 Mb/s Ethernet port for communication over the network (including e-mail) and built-in Web (HTTP) and file servers (FTP). Using the LabVIEW Remote Panel feature, you can automatically publish the front panel graphical user interface (GUI) for your embedded application so that multiple clients can monitor or control it remotely using a Web browser.

Software

NI LabVIEW is a graphical development environment that delivers unparalleled flexibility and ease of use in demanding industrial measurement, automation, and control applications. With LabVIEW, you quickly create user interfaces for interactive software system control. LabVIEW makes it easy to construct simple or complex applications using an extensive palette of functions and tools – from simple analog PID process control loops to high-channel-count hybrid control systems. Each LabVIEW Real-Time hardware target, including Compact FieldPoint embedded controllers, has a dedicated processor running a real-time OS for reliability, stability, and determinism. Use the LabVIEW platform for your industrial measurement, automation, and control applications by following these three steps:

- Choose your I/O – LabVIEW delivers access to the widest selection of I/O, from data acquisition, motion control, and vision integration to machine vision and custom-designed I/O from a single environment.
- Choose your analysis or control methods – With more than 450 LabVIEW analysis and control functions, you define your system to meet your specific requirements.
- Choose your real-time deployment platform – Once you create your LabVIEW application, deploy it to run deterministically on the hardware platform you choose.

LabVIEW 8 Project and FieldPoint Programming

Drag-and-Drop Programming

With the new LabVIEW 8 Project, programming is simplified with drag-and-drop functionality. You can add local or distributed I/O from any Compact FieldPoint bank simply by dragging I/O from a LabVIEW Project to the VI where you want to read/write to that FieldPoint tag. Figure 1 shows I/O from a cFP-AI-102 module on a cFP-2120 bank being added to a new VI. LabVIEW automatically creates the FieldPoint tag and appropriate FieldPoint Read/Write VI when you drag an item from the LabVIEW Project Explorer.

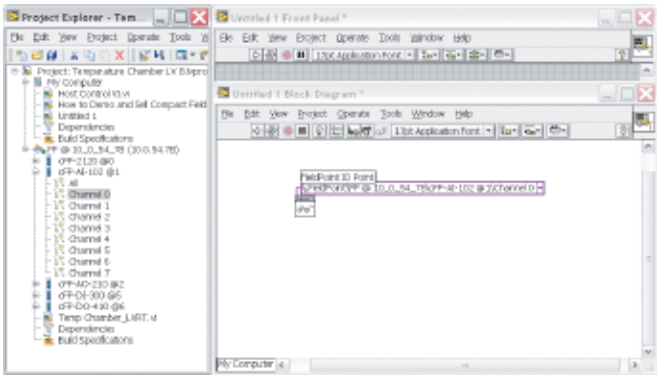


Figure 1. A LabVIEW Project provides easy access to Compact FieldPoint I/O.

LabVIEW Shared Variable

cFP-21xx controllers are compatible with the new LabVIEW shared variable that simplifies communication between distributed PAC systems. To create a shared variable within a LabVIEW Project, right-click on the location that will host the shared variable, either the Windows system or a cFP-21xx real-time controller. Then create a shared variable, or choose to bind the variable to an I/O source such as a FieldPoint I/O tag on a FieldPoint controller on the network. Figure 2 shows a LabVIEW Project with a cFP-2120 target that is hosting shared variables for the PID Setpoint, dt, and Process Variable. When this code is deployed to the cFP-2120 real-time controller, the shared variable configuration is deployed. The cFP-2120 real-time controller updates values for these shared variables independently from the Windows system.

Note: cFP-21xx controllers are recommended if you want to host shared variables on the real-time controller. Hosting shared variables requires the shared variable engine to be installed to the embedded target.

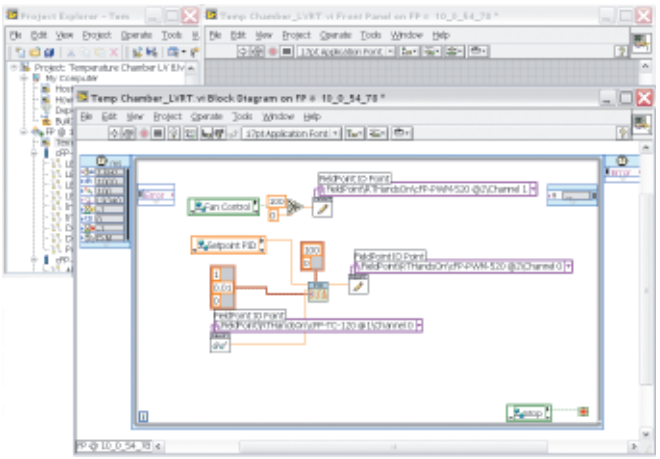


Figure 2. The cFP-2120 real-time controller hosts LabVIEW shared variables.

Expansion I/O with NI cFP-1808

For systems that require more than eight Compact FieldPoint modules, the new National Instruments cFP-1808 network interface provides an easy way to add expansion I/O over Ethernet or serial. A single cFP-21xx controller can connect to as many additional NI cFP-1808 network interface systems as the Ethernet network allows. With LabVIEW 8, the I/O from an expansion system appears in the LabVIEW Project and is easy to program using the FieldPoint API by simply dragging a tag from the cFP-1808 bank to a VI.

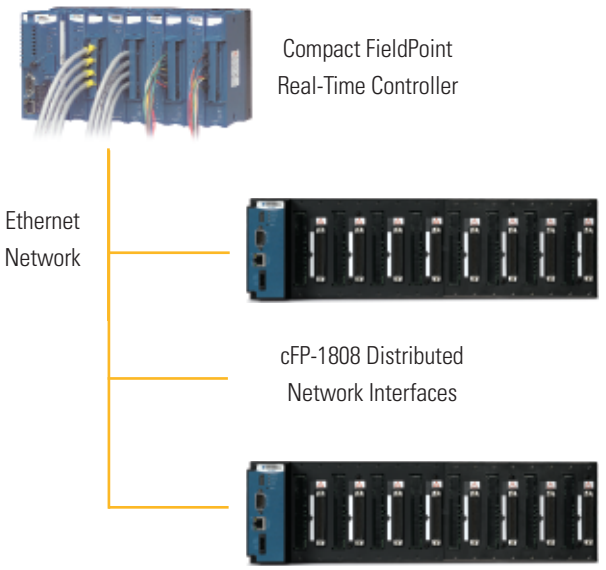


Figure 3. Expansion I/O for cFP-21xx Controllers with NI cFP-1808 Network Interface

Embedded Data Logging

The cFP-2120 features 128 MB of nonvolatile removable CompactFlash storage for data logging or additional storage capacity. You can store the data in standard format, including CSV and XML. Once you store the data, you can easily transfer it to a PC using the embedded FTP server on the cFP-21xx. LabVIEW Real-Time expands the functionality beyond the typical data logger because you can make additional calculations and decisions to eliminate logging unneeded data and to perform onboard real-time calculations. Compact FieldPoint combines data logging, data reduction, control algorithms, a Web-based human machine interface (HMI), and the ability to communicate with other nodes on the network.

Communication

A cFP-21xx controller connects directly to your network through the built-in Ethernet port. The Ethernet port serves as a high-speed link for downloading application code, performing run-time debugging and probing, and transmitting control and indicator values with a GUI running on a networked PC. You also can use the Ethernet port for programmatic network communication using standard protocols such as TCP, UDP, FTP, HTTP, and DataSocket. Once deployed, the controller can communicate with any Ethernet-enabled device on the network. In addition, a cFP-21xx can communicate with a Windows computer running LabVIEW or any third-party HMI/SCADA software compatible with OPC. By using LabVIEW libraries and industrial gateways, you can add a Compact FieldPoint bank to any existing setup and communicate with industrial devices through standard communication protocols such as Modbus TCP and PROFIBUS.

Serial Connectivity

cFP-21xx controllers have up to three RS232 serial ports and one RS485 port (cFP-2120) to communicate programmatically with other serial devices such as remote FieldPoint banks, LCD display/keypad units, bar code readers, or phone and radio modems.

Power Supply Backup and Regulation

cFP-21xx controllers require an 11 to 30 VDC power supply. An extra set of screw terminals is available on the network controllers for a backup UPS or battery. The controller filters and regulates the power input, redistributing power to all the I/O modules in the node through the backplane bus. Refer to Ordering Information for suitable power supplies.

Ordering Information

NI cFP-2120 777317-2120
NI cFP-2110 777317-2110
NI cFP-2100 777317-2100

Recommended Compact FieldPoint System Products

NI cFP-BP-4 (4-slot backplane).....778617-04
NI cFP-BP-8 (8-slot backplane).....778617-08
NI cFP-CB-1 (internal connecter block) 778618-01
NI cFP-CB-3 (isothermal connector block) 778618-03
NI PS-5 Power Supply 778805-90

BUY NOW!

For complete product specifications, pricing, and accessory information, call 800 813 3693 (U.S.) or go to **ni.com/fieldpoint**.

Specifications

Network

Network interface	10BaseT and 100BaseTX Ethernet
Compatibility	IEEE802.3
Communication rates	10 Mb/s, 100 Mb/s, autonegotiated
Maximum cabling distance	100 m/segment
Maximum number of banks	Determined by network topology

Memory

cFP-2100	64 MB nonvolatile; 64 MB DRAM
cFP-2110	64 MB nonvolatile; 128 MB DRAM
cFP-2120	128 MB nonvolatile; 128 MB DRAM
Memory lifetime (nonvolatile)	300,000 writes per sector

For information about the memory used by the LabVIEW Real-Time Module and the OS, go to **ni.com/info** *and enter* **rdfpec**.

Serial Ports

cFP-2100	1 RS232
cFP-2110	2 RS232
cFP-2120	3 RS232; 1 RS485

RS232 (DTE) Ports

Baud rate	110 to 115,200 b/s
Data bits	5, 6, 7, 8
Stop bits	1, 1.5, 2
Parity	Odd, even, mark, space
Flow control	RTS/CTS, XON/XOFF, DTR/DSR

RS485 (DTE) Port

Baud rate	110 to 115,200 b/s
Data bits	5, 6, 7, 8
Stop bits	1, 1.5, 2
Parity	Odd, even, mark, space
Flow control	XON/XOFF
Mode	4-wire
Maximum continuous	
Isolation voltage	100 V_{rms}
Dielectric withstand	740 V_{rms}, 1 minute

Power Requirement

Power supply range	11 to 30 VDC
Recommended power supply	
cFP-BP-4 system	15 W
cFP-BP-8 system	20 W
Power consumption	6.1 W + 1.1(I/O module power requirements)
Maximum power to connected I/O modules	9 W

Physical Characteristics

Screw-terminal wiring	16 to 26 AWG copper conductor wire with 7 mm (0.28 in.) of insulation stripped from the end
Torque for screw terminals	0.5 to 0.6 N•m (4.4 to 5.3 lb•in.)
Weight	278 g (9.8 oz)

Environmental

FieldPoint modules are intended for indoor use only. For outdoor use, they must be installed in a suitable sealed enclosure.

Operating temperature	-40 to 70 °C
Storage temperature	-55 to 85 °C
Relative humidity	10 to 90%, noncondensing
Maximum altitude	2,000 m; at higher altitudes the isolation voltage ratings must be lowered
Pollution degree	2

Shock and Vibration

Operating vibration	
Random (IEC 60068-2-64)	10 to 500 Hz, 5 g_{rms}
Sinusoidal (IEC 60068-2-6)	10 to 500 Hz, 5 g
Operating shock	
(IEC 60068-2-27)	50 g, 3 ms half sine, 18 shocks at 6 orientations; 30 g, 11 ms half sine, 18 shocks at 6 orientations

Safety

This product is designed to meet the requirements of the following standards of safety for electrical equipment for measurement, control, and laboratory use:

- IEC 61010-1, EN 61010-1
- UL 61010-1, CSA 61010-1

Note: *For UL and other safety certifications, refer to the product label or visit* **ni.com/certification**, *search by model number or product line, and click the appropriate link in the Certification column.*

Electromagnetic Compatibility

This product is designed to meet the requirements of the following standards of EMC for electrical equipment for measurement, control, and laboratory use:

- EN 61326 EMC requirements; Industrial Immunity
- EN 55011 Emissions; Group 1, Class A
- CE, C-Tick, ICES, and FCC Part 15 Emissions; Class A

Note: *For EMC compliance, operate this device according to product documentation.*

CE Compliance

This product meets the essential requirements of applicable European Directives, as amended for CE marking, as follows:Low-Voltage Directive (safety)

- 73/23/EEC; Low-Voltage Directive (safety)
- 89/336/EEC; Electromagnetic Compatibility Directive (EMC)

Note: Refer to the Declaration of Conformity (DoC) for this product for any additional regulatory compliance information. To obtain the DoC for this product, visit **ni.com/certification**, *search by model number or product line, and click the appropriate link in the Certification column.*

Waste Electrical and Electronic Equipment (WEEE)

EU Customers: At the end of their life cycle, all products must be sent to a WEEE recycling center. For more information about WEEE recycling centers and National Instruments WEEE initiatives, visit **ni.com/environment/weee.htm**.

NI Services and Support

NI has the services and support to meet your needs around the globe and through the application life cycle – from planning and development through deployment and ongoing maintenance. We offer services and service levels to meet customer requirements in research, design, validation, and manufacturing. Visit **ni.com/services**.

Training and Certification

NI training is the fastest, most certain route to productivity with our products. NI training can shorten your learning curve, save development time, and reduce maintenance costs over the application life cycle. We schedule instructor-led courses in cities worldwide, or we can hold a course at your facility. We also offer a professional certification program that identifies individuals who have high levels of skill and knowledge on using NI products. Visit **ni.com/training**.

Professional Services

Our NI Professional Services team is composed of NI applications and systems engineers and a worldwide National Instruments Alliance Partner program of more than 600 independent consultants and integrators. Services range from start-up assistance to turnkey system integration. Visit **ni.com/alliance**.

NATIONAL INSTRUMENTS™
Certified Alliance Partner

OEM Support

We offer design-in consulting and product integration assistance if you want to use our products for OEM applications. For information about special pricing and services for OEM customers, visit **ni.com/oem**.

Local Sales and Technical Support

In offices worldwide, our staff is local to the country, giving you access to engineers who speak your language. NI delivers industry-leading technical support through online knowledge bases, our applications engineers, and access to 14,000 measurement and automation professionals within NI Developer Exchange forums. Find immediate answers to your questions at **ni.com/support**.

We also offer service programs that provide automatic upgrades to your application development environment and higher levels of technical support. Visit **ni.com/ssp**.

Hardware Services

NI Factory Installation Services

NI Factory Installation Services (FIS) is the fastest and easiest way to use your PXI or PXI/SCXI combination systems right out of the box. Trained NI technicians install the software and hardware and configure the system to your specifications. NI extends the standard warranty by one year on hardware components (controllers, chassis, modules) purchased with FIS. To use FIS, simply configure your system online with **ni.com/pxiadvisor**.

Calibration Services

NI recognizes the need to maintain properly calibrated devices for high-accuracy measurements. We provide manual calibration procedures, services to recalibrate your products, and automated calibration software specifically designed for use by metrology laboratories. Visit **ni.com/calibration**.

Repair and Extended Warranty

NI provides complete repair services for our products. Express repair and advance replacement services are also available. We offer extended warranties to help you meet project life-cycle requirements. Visit **ni.com/services**.

NATIONAL INSTRUMENTS™

ni.com • 800 813 3693
National Instruments • info@ni.com

351498A-01 2007-8951-161-101-D