

# INSTRUCTION MANUAL

---

## ECD140/ECD141 Eddy-Current Displacement Sensor for National Instruments™ CompactRIO

Measurement Systems  
from  
LION PRECISION

# **Approvals and Safety Considerations**

---

The ECD140/ECD141 is compliant with the following CE directives:

Safety: 61010-1:2001

EMC: 61326-1, 61326-2-3

To maintain compliance with these standards, the following operating conditions must be maintained:

- All I/O connecting cables must be less than three meters in length
- Sensors must not be attached to parts operating at hazardous voltages in excess of 30VRMS or 60VDC
- All external connections must be SELV (Safety Extra Low Voltage).

Use of the equipment in any other manner may impair the safety and EMI protections of the equipment.

The ECD140/ECD141 is made in the United States of America.

## **ECD140 and ECD141**

---

The ECD141 is identical to the ECD140 except that resolution is limited to 0.3  $\mu\text{m}$ . For this reason, the ECD141 does not require an export license.

This manual will only refer to the ECD140, but all instructions apply to the ECD141.

### **Description**

---

The Lion Precision ECD140 Eddy-Current Displacement Sensor provides high-resolution, noncontact measurement of position changes of a conductive target. The system consists of driver electronics and a probe calibrated for a specific material and measurement range. The calibration information is detailed on a calibration certificate which is shipped with the system and can be downloaded and printed from the device with a VI.

Using the downloadable VIs, the ECD140 provides data indicating changes in relative position of the target and probe surface. The data is presented in  $\mu\text{m}$  (micrometers)

### **Intended Users**

---

The ECD140 and ECD141 are intended for LabVIEW users experienced in FPGA and Real Time processor programming in the RIO platform. The devices and the software provide a tool kit for developers creating a complete system which requires precision noncontact displacement sensors. The system is not intended to be out-of-the-box operational, but requires implementation and programming by experienced LabVIEW FPGA users.

Those unfamiliar with LabVIEW FPGA programming may want to visit National Instrument's FPGA training page:

<http://zone.ni.com/devzone/cda/tut/p/id/3555>

### **Using the Correct Revision Levels**

---

For the system to function correctly, the Software version (LabVIEW drivers, demos etc.) and the Firmware version (internal program in the ECD140 module) must be correlated. The download area of [www.ecd140.com](http://www.ecd140.com) indicates which software version correlates to specific firmware versions.

The ECD140 Firmware version is indicated by “Firmware Rev” label on the module. The Firmware version can also be read from the module by reading the upper byte in the Status data (see Data Type and Error Code section on page 18).

The software version is indicated in the name of the downloaded zip file.

The Software and Firmware versions are read and displayed by both demo programs.

## **Important Notes**

---

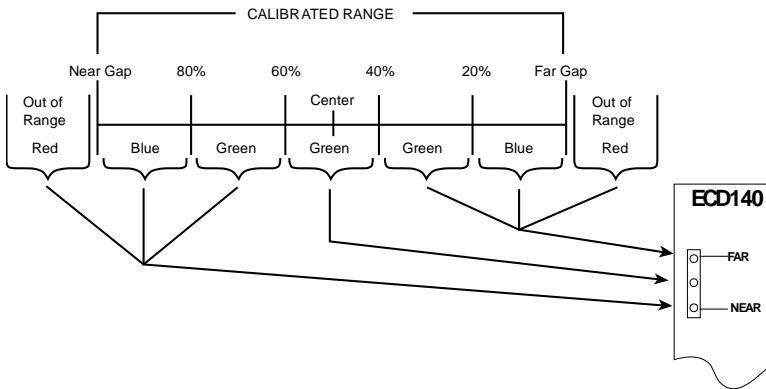
The ECD140 depends on a cRIO System Chassis FPGA derived clock for operation. On power up, the ECD140 Near/Far range indicator will be in a random state until the FPGA derived clock is started. The clock must be set to run at 50MHz for proper operation. Specific details are given later in this manual.

The ECD140 does NOT have a sleep mode.

# Front Panel Controls and Indicators

## LED Range Indicator

The Range Indicator monitors and displays the probe position within its calibrated range. The graphic below shows the indicator condition at various points within the calibrated range.



The ECD140 depends on a cRIO System Chassis FPGA derived clock for operation. On power up, the ECD140 Near/Far range indicator will be in a random state until the FPGA derived clock is started.

## Maximizing Performance

### Extension Cables

Sensors which are calibrated with a probe extension cable must be operated with the extension cable to meet specifications. Operating without the extension cable will result in inaccurate measurements.

### Probe Mounting

The sensing field around an eddy-current probe is three to five times larger than the probe diameter. If multiple probes are mounted together, they must be separated by at least three to five probe diameters to prevent interference between the channels.

Metal objects such as mounting brackets and panels can also affect the sensor. Probes should be mounted with the flat, end surface of the

probe at least 1.5 diameters away from the panel through which it is mounted and at least 3-5 diameters away from any metallic objects to the side of the probe.

### **Ungrounded Targets**

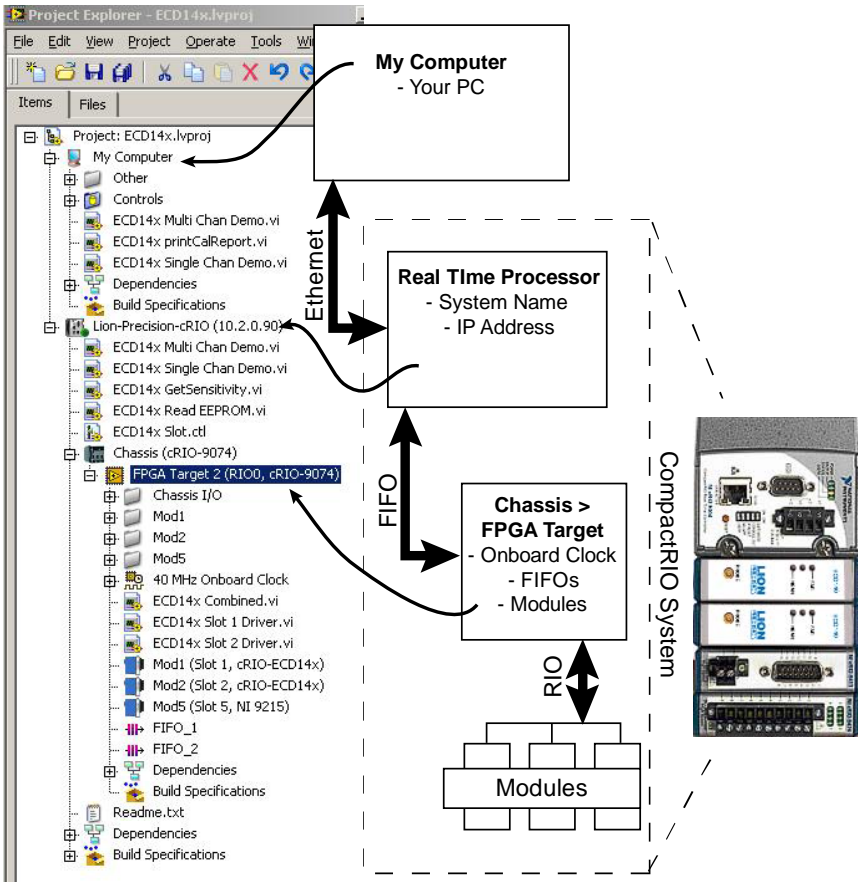
Ungrounded targets have the potential to inject noise into the sensor. If the output is unusually noisy, be sure the target is grounded. On moving/rotating targets this can be accomplished with a small metal brush or thin piece of metal which is connected to ground.

### **Multiple Sensors**

It is critical that each ECD140 be connected to the probe indicated on the "PROBE SERIAL" label on the side of each ECD140.

# Understanding CompactRIO Architecture

*This block diagram and example Project tree lists a CompactRIO system (Lion-Precision-cRIO) at the Real-Time level that was in use when the screen image was captured. A similar level exists in the demo program. You will have to add YOUR CompactRIO system to the project which will appear as another level in the Project tree. Specific instructions are included in the next section.*



Installing and running VIs, drivers, and other files/programs requires an understanding of the system architecture. Files and devices can be installed and run at different levels depending on how they will be used. The system consists of three nested levels:

- My Computer (your PC)
- Real-Time Processor (top-level CompactRIO processor; exists within the CompactRIO hardware). *This level is listed in the Project Tree by the System name and IP address.*
- FPGA Target (lowest level CompactRIO processor; exists within the CompactRIO hardware and communicates directly with the modules)

These levels are displayed in the Project Tree of an open project. Some files will run in different levels; others will only run in specific levels.

*The Real-Time level is listed in the Project Tree by the System name and IP address (the words “Real-Time” do not appear in the tree).*

Your PC (the My Computer level) communicates with the CompactRIO system through an Ethernet connection. To operate reliably, the CompactRIO system must be assigned an IP address in MAX (Measurement and Automation Explorer); details are listed later. The CompactRIO system on the other side of the Ethernet connection is the Real-Time processor. Files can run in the Real-Time processor and avoid being interrupted by other processes on your PC. The Real-Time processor level is shown in the project tree as the Name and IP Address that are assigned to the CompactRIO system in MAX.

The FPGA Target is a processor that runs in the CompactRIO system chassis and connects directly to the interface to the modules. The modules are operated by and data exchanged with files that run on the FPGA Target level. This is the level at which high-speed operations can occur because of its direct communication with the modules. Module drivers must be run in the FPGA processor.

The Real-Time processor acquires data from the FPGA Target processor through a FIFO (First In First Out) that exists on the FPGA Target level. The drivers running at the FPGA level load the FIFO, and the programs running at the Real-Time processor level extract the data from the FIFO. The instructions that follow require a basic understanding of these different levels and how they are listed in the Project tree.

## Sample Rate

---

The sample rate of the ECD140 is between 62 kHz and 67 kHz and cannot be changed. If less data is desired, it is recommended that an averaging function be created to produce less frequent data for your program.

## Working at the FPGA Level

---

*If you are new to FPGA level programming with LabVIEW, carefully follow these instructions step-by-step, and begin with the Basic version.*

## Using the Correct Revision Levels

---

For the system to function correctly, the Software version (LabVIEW drivers, demos etc.) and the Firmware version (internal program in the ECD140 module) must be correlated. The download area of [www.ecd140.com](http://www.ecd140.com) indicates which software version correlates to specific firmware versions.

The ECD140 Firmware version is indicated by “Firmware Rev” label on the module. The Firmware version can also be read from the module by reading the upper byte in the Status data (see Data Type and Error Code section on page 18).

The software version is indicated in the name of the downloaded zip file.

The Software and Firmware versions are read and displayed by both demo programs.

## Installing the Software on the PC

---

Two “zip” files containing all the necessary VIs and other support files can be download from [www.ecd140.com](http://www.ecd140.com). There is a separate zip file for the Basic version and the Multiple Channel version.

The ECD140 comes with LabVIEW demo projects, VI drivers, and other required support files. To install these components:

1. Decompress the “zip” file
2. The file, “Lion Precision.ini” (contained in each project) must be placed in this location:

```
C:\Program Files\National Instruments\LabVIEW  
2009\Targets\NI\FPGA\cRIO\Other\Lion Precision\Lion  
Precision.ini.
```

The path may be slightly different depending on the drive in which you installed LabVIEW. The “Other” folder may already exist or you may have to create it.

This file location is critical and must be exact for LabVIEW’s AutoDetect to work.

3. Place the other project files in a convenient project location of your choosing.

## Connecting the ECD140 and Setting the IP

---

1. Install the ECD140 in the CompactRIO system in your chosen slot (use slot 1 for Basic Project, any or all slots 1-4 for Multiple Channel Project) and apply power.
2. Connect the system to the PC via the Ethernet connection.
3. Set IP addresses for your system (if not already done)
  - a. Open MAX (Measurement and Automation Explorer)
  - b. Select/Click your CompactRIO system under the Remote Systems heading
  - c. The window will display information about the system including the IP address. The IP should be set to an available address that does not conflict with any other devices on your Ethernet network. The address should not be 0.0.0.0 which is the default when first

connected. On local area networks, a typical IP address is 192.168.1.xxx where “xxx” represents the address of the device. You may need to contact a network administrator for assistance in selecting an appropriate IP address.

- d. Close MAX.

## Using the Demo Projects

---

When opening the included demo projects, the Project tree will show the CompactRIO system which was used for development of the demos.

*You must add YOUR CompactRIO hardware to the project. Each system will have a Real-Time level and Chassis > FPGA Target level. VIs and other files must be added to YOUR system.*

Unless specified otherwise, when the instructions for using the demo projects speak of Real-Time and FPGA levels, they are speaking of levels within your CompactRIO system.

## Using the Basic Project

---

The Basic Project uses one ECD140 in slot 1 of the CompactRIO system. It provides for initializing and collecting displacement data from the ECD140. The VIs will need to be modified to use in any slot other than Slot 1.

1. Open ECD14x Basic Demo Rev 2.lvproj.
2. Add your CompactRIO system to the project (remember to save the project as you add to it):
  - a. Right click the top level “ECD14x Basic Demo Rev 2.lvproj” in the project tree and select New Targets and Devices.
  - b. In the dialog box, select the type of RIO system you’re using.
  - c. Your system should be detected and displayed.
  - d. Click to highlight your system and click OK
3. Add the VIs and other required files  
(When the project file is opened, the required files will be

present in the project tree, but they will have to be added to your specific system's Real-Time and FPGA levels):

VI's and other files are added (as necessary) to the project at the Real-Time level (your system name), and the FPGA level.

Either of two methods can be used: Files can be added by copying them from elsewhere in the project tree and pasting into the correct location, or they can be added by adding them from the software installation folder on your computer disk drive.

### **Copy and Paste Method**

All the required VI's exist in the project tree, but most are located under a CompactRIO system used during development. They must be moved to the Real-Time and FPGA levels under your system. No files are installed in the My Computer level for the Basic Project. The basic Copy/Paste operation is this:

- a. Find the VI or support file elsewhere in the tree (Lion Precision system)
- b. Copy the file (ctrl-C)
- c. Click the level under your system where you want to paste the file (i.e. Real-Time, FPGA Target)
- d. Paste the file (ctrl-V).

### **Add From Hard Drive Method**

- a. Under your CompactRIO system, right click the Real-Time or FPGA Target level in the tree and select Add > File.
- b. In the popup dialog, navigate to the location of the desired file (in the "Other" folder created during software installation) and select the desired file. Multiple files can be selected by holding the Ctrl key while clicking other files.
- c. Click Add File Button

Using either method, add the following files to the indicated levels in the Project tree:

- a. **Real-Time level** (listed by your system name and IP)
    - ECD14x Basic Demo.vi
    - ECD14x GetSensitivity.vi
    - ECD14x Read Slot 1 EEPROM.vi
    - ECD14x Counts and Gaps.ctl
  - b. **FPGA Level**
    - ECD14x Clock and FIFO.vi
    - ECD14x Slot 1 Driver.vi
4. Create the FIFO at the FPGA level
  - a. Under your system, right-click FPGA Target and select New > FIFO
  - b. Create FIFO\_1 with these parameters (exact name is important; used for data transfer from ECD14x module):
    - Name: FIFO\_1
    - Type: Target to Host DMA
    - Data Type: U16
    - Number of Elements: 1023 (default)
5. Create 50MHz derived clock at FPGA level
  - a. Right click on 40MHz Onboard Clock and select New FPGA Derived Clock.
  - b. Set the frequency to 50MHz and the name to "50MHz" (default, name must be exact)
6. Add the ECD140 module  
(If the ECD140 was installed in the CompactRIO system when the system was added to the project, then the module should already be present under the FPGA level and this step is not necessary)
  - a. Click the "+" left of your system (Real-Time level)
  - b. Right-click FPGA Target
  - c. Select New > C Series Module
  - d. Select "Existing target or device" for AutoDetect
  - e. Click the "+" next to the C Series Module folder
  - f. LabVIEW will search for the module and display it in the tree under the folder.

- g. Click/Select the cRIO-ECD14x module
  - h. Click OK (be sure to click/select the module first)
- If AutoDetect does not locate the module
- a. Click the “+” left of your system (Real-Time level)
  - b. Right-click FPGA Target
  - c. Select New > C Series Module
  - d. Select “New target or device”
  - e. Click the “+” next to the C Series Module folder
  - f. Select the “cRIO-ECD14x” at or near the bottom of the list of devices
  - g. Click OK
7. Compile VIs in the following sequence:  
(Each file may take several minutes to compile)
- a. Right-click each “Driver” VI in the FPGA level and select Compile.
  - b. Compile Clock and FIFO.vi.

## **Running the Basic Demo**

---

1. Open the "ECD14x Basic Demo.vi" file at the Real Time level.
2. Start/Run the program.
3. There is a short delay while the program reads the EEPROMs.
4. After reading the EEPROMs, the displayed Range and Sensitivity values should be real numbers.
5. Verify operation by moving the probe toward and away from a conductive target within the calibrated range as stated in the Calibration Sheet, or simply make contact with the target and slowly move the probe away. Any metallic object will usually work for testing basic functionality, but accurate results will only be achieved with the same material to which the system was calibrated.
6. When in contact with the target, the gap value should decrease to near zero micrometers.

7. When moving the probe away from the target, the reported gap value should increase to greater than the Far Gap value.

## Using the Multi-Channel Project

---

The Multi-Channel Project uses one or more ECD140s in slots 1-4 of the CompactRIO system along with an NI 9201 DAQ module. The project is an example of integrating the ECD140 in a complete CompactRIO system with other modules. The NI 9201 DAQ module need not be physically installed to use the project.

The Multi-Channel Project is built around the ECD14x Project and the ECD14x Multi Chan Demo VI.

Because of this project's complexity, if you are new to FPGA level programming with cRIO devices and LabVIEW, do not attempt this until you have successfully installed and run the Basic Project listed above.

1. Open ECD14x.lvproj.
2. Add your CompactRIO system to the project (remember to save the project as you add to it):
  - a. Right click the top level "ECD14x.lvproj" in the project tree and select "New Targets and Devices."
  - b. In the dialog box, select the type of RIO system you're using.
  - c. Your system should be detected and displayed.
  - d. Click to highlight your system and click OK
3. Add the VIs and other required files  
(Needed files will be present when the project file is open, but they will have to be added to your specific system's Real-Time and FPGA levels):

VIs and other files are added (as necessary) to the project at the My Computer level, the Real-Time level (your system name), and the FPGA level.

Either of two methods can be used: Files can be added by copying them from elsewhere in the project tree and pasting into the correct location, or they can be added by adding them from the software installation folder on your computer disk drive.

## Copy and Paste Method

All the required VIs exist in the project tree, but most are located under a system used during development. They must be moved to the Real-Time and FPGA levels under your system. The files installed in the My Computer level will not need to be moved. The basic Copy/Paste operation is this:

- a. Find the VI or support file elsewhere in the tree
- b. Copy the file (ctrl-C)
- c. Click the level under your system where you want to paste the file (i.e. Real-Time, FPGA Target)
- d. Paste the file (ctrl-V).

## Add From Hard Drive Method

- a. Right click the Real-Time or FPGA Target level in the tree and select Add > File.
- b. In the popup dialog, navigate to the location of the desired file (in the “Other” folder created during software installation) and select the desired file. Multiple files can be selected by holding the Ctrl key while clicking other files.
- c. Click Add File Button

Using either method, add the following files to the indicated levels in the Project tree:

- a. **My Computer** level  
ECD14x printCalReport.vi  
ECD14x Read EEPROM.vi
- b. **Real-Time** level (listed by your system name and IP)  
ECD14x GetSensitivity.vi  
ECD14x Multi Chan Demo.vi  
ECD14x Read EEPROM.vi  
ECD14x Slot.ctl
- c. **FPGA** Level  
ECD14x Combined.vi  
ECD14x Slot 1 Driver.vi  
ECD14x Slot 2 Driver.vi  
ECD14x Slot 3 Driver.vi  
ECD14x Slot 4 Driver.vi

4. Create the FIFO at the FPGA level
  - a. Under your system right-click FPGA Target and select New > FIFO
  - b. Create FIFO\_1 with these parameters (exact name is important; used for data transfer from ECD14x module):
    - Name: FIFO\_1
    - Type: Target to Host DMA
    - Data Type: U64
    - Number of Elements: 1023 (default)
  - c. Create FIFO\_2 with these parameters (exact name is important; used for data transfer from NI 9201 DAQ module):
    - Name: FIFO\_2
    - Type: Target to Host DMA
    - Data Type: FXP
    - Number of Elements: 1023 (default)
    - Encoding: Signed (default)
    - Word Length: 16 bits (default)
    - Integer word length: 5 bits
  
5. Create 50MHz derived clock at FPGA level
  - a. Right click on 40MHz Onboard Clock and select New FPGA Derived Clock.
  - b. Set the frequency to 50MHz and the name to “50MHz” (default, name must be exact)
  
6. Add the ECD140 module(s)

(If the ECD140(s) was installed in the CompactRIO system when the system was added to the project, then the module should already be present under the FPGA level and this step is not necessary).  
ECD140 modules will have to be added slots 1-4 using these instructions, but they do not have to be physically present in the slots.

  - a. Click the “+” left of your system (Real-Time level)
  - b. Right-click FPGA Target
  - c. Select New > C Series Module

- d. Select “Existing target or device” for AutoDetect
- e. Click the “+” next to the C Series Module folder
- f. LabVIEW will search for the module and display it in the tree under the folder.
- g. Click/Select the cRIO-ECD14x module
- h. Click OK (be sure to click/select the module first)

If AutoDetect does not locate the module (these steps are also used to add to the project ECD140 modules that are not physically present)

- a. Click the “+” left of your system (Real-Time level)
  - b. Right-click FPGA Target
  - c. Select New > C Series Module
  - d. Select “New target or device”
  - e. Click the “+” next to the C Series Module folder
  - f. Select the “cRIO-ECD14x” at or near the bottom of the list of devices
  - g. Click OK
7. Compile VIs in the following sequence:  
(Each file may take several minutes to compile)
- a. Right-click each “Driver” VI in the FPGA level and select Compile.

## **Running the Multi-Channel Demo Program**

1. Open the "ECD14x Multi Chan Demo.vi" file at the Real Time level.
2. Before starting the program, use the switches to select the active slot(s).
3. Start/Run the program.
4. There is a short delay while the program reads the EEPROMs.

5. After reading the EEPROMs, the displayed Ranges and Sensitivity values should be real numbers.
6. Verify operation by moving the probe toward and away from the target within the calibrated range as stated in the Calibration Sheet, or simply make contact with the target and slowly move the probe away.
7. When in contact with the target, the gap value should decrease to near zero micrometers.
8. When moving the probe away from the target, the reported gap value should increase to greater than the Far Gap value.

## **Data Type and Error Checking**

---

The ECD140 is able to self-test some critical parameters. Version 2 of the firmware can detect a shorted probe. The detection is accomplished in the Clocks and FIFOs VI (Basic) or Combined VI (Multiple Channel). The VI panel includes an error indicating output and associated “LED” and a “Data Type” selector. The Data Type selector has three options:

- EEPROM (0) – Direct EEPROM readings for calibration information and other module data
- Probe (1) – Measurement data stream
- Status (2) – Status information including firmware rev level (upper byte) and error codes (lower byte)

Normally, Data Type is set to Probe (1). If the error indicator is showing the presence of an error, the data type can be changed to Status (2) and the resulting data will indicate the type of error.

The upper byte of the Status data is the revision level of the Firmware; the lower byte is the error code. A “1” in the LSB (bit 0) indicates a shorted probe.

# Required National Instruments Software Revision Levels

---

If problems arise, confirm the following software versions in the Measurement and Automation Explorer (MAX):

On cRIO chassis:

NI-RIO 3.2.0 - June 2009

The correct selection will have about 15 items listed under Software>>NI-RIO3.2.0 - June 2009

Do NOT choose the “NI-RIO 3.2.0 (minimal)” as it does not have all the necessary components.

On the PC:

LabVIEW 2009 9.0.0, including:

FPGA 9.0.0

Real-Time 9.0.0

NI-RIO 3.2

If changes are necessary,

1. Go to Remote Systems (in MAX left window)
2. Select your cRIO chassis
3. Select “Software”
4. Click “Add/Remove Software” button at the top of the right window

# VI Listing

---

## Levels at which files are installed and run

Many VIs and other files can be installed and run at different levels in the project tree. Running some VIs at the My Computer level may limit performance due to limited Ethernet bandwidth and other demands on the PC's processor. These VIs should be run at the Real-Time level where they are not interrupted by other computing processes and do not depend on the Ethernet connection for communicating with the ECD140 modules.

Each of the files listed below indicate the “Best” and “Optional” levels at which they can operate.

### Project software:

#### *ECD14x Basic Demo Rev 2.lvproj*

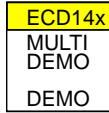
This is a very simple project that initializes and gets measurement data from a single ECD140 in slot 1. This allows verification of basic system operation in a simple environment. It's the best place to start.

#### *ECD14x.lvproj*

This is a sample project using multiple ECD140s and an NI DAQ module. It demonstrates the use of the ECD140 in a complete system with other modules. Additional ECD140s and other NI modules do not need to be physically present to use the project, but the modules will have to be “installed” in the project.

**VIs:**

*ECD14x Multi Chan Demo.vi*



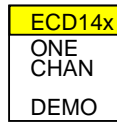
Runs up to four ECD14x devices in slots 1-4 and a NI DAQ module.

Included in Multi-Channel Project (ecd14x.lvproj)

*Installation Levels:*

- Best – Real-Time Processor
- Optional – My Computer. Can run at this level but may not perform well. A less demanding adaptation could run here.

*ECD14x Basic Demo.vi*



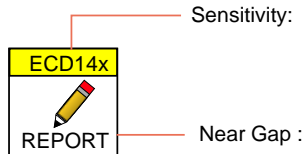
Runs a single ECD14x device in slot 1.

Included in Basic Project (ecd14x Basic Demo Rev 2.lvproj)

*Installation Levels:*

- Best – Real-Time Processor
- Optional – My Computer; Can run at this level but may not perform well. A less demanding adaptation could run here.

*ECD14x printCalReport.vi*



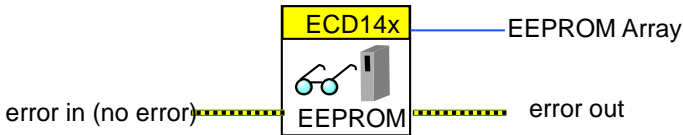
Reads the factory calibration report from the selected ECD14x and displays the report on screen or printer depending on installation.

Included in Multi-Channel Project (ecd14x.lvproj)

*Installation Levels:*

- Best – My Computer. Can retrieve calibration report, display it on screen, and send to a printer.
- Optional – Real-Time Processor; Can only display the report on screen.

*ECD14x Read Slot 1 EEPROM.vi*



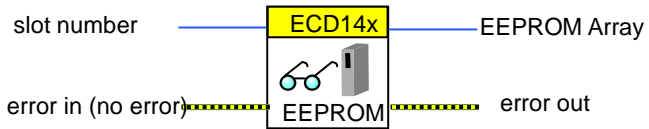
Retrieves system data (not measurement data) from the ECD140 EEPROM in slot 1. Support VI for other VIs to retrieve system information such as calibration and sensitivity.

Included in Basic Project (ecd14x Basic Demo Rev 2.lvproj)

*Installation Levels:*

- Best – Real-Time Processor
- Optional – My Computer.

*ECD14x Read EEPROM.vi*



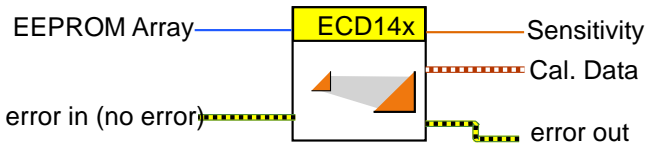
Retrieves system data (not measurement data) from the ECD140 EEPROM in the selected slot. Support VI for other VIs to retrieve system information such as calibration and sensitivity.

Included in Multi-Channel Project (ecd14x.lvproj)

*Installation Levels:*

- Best – Real-Time Processor
- Optional – My Computer.

## ECD14x GetSensitivity.vi



Returns values required to determine dimensional values from the ECD140 output. The ECD140 is factory calibrated for a particular range and sensitivity. The calibrated range begins at the Near Gap distance from the target and extends through the “Range.”

**Sensitivity** (floating point): returned as counts per micrometer ( $\mu\text{m}$ ).

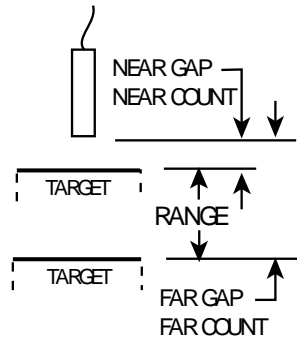
**Cal. Data** cluster returns the following values:

**Far Gap** (floating point):  
The distance, in micrometers, between the probe and the target when at Far Count. This is the furthest point in the calibrated range.

**Far Count** (unsigned integer): The count returned when the probe is at Far Gap.

**Near Gap** (floating point):  
The distance, in micrometers, between the probe and the target when at Near Count. This is the nearest point in the calibrated range.

**Near Count** (unsigned integer): The count returned when the probe is at Near Gap.



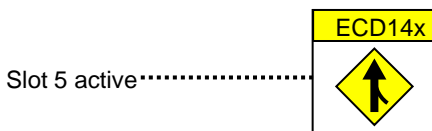
Measurement values from the ECD140 can be returned that are beyond the Far Count or Near Count. These values indicate that the probe is beyond its calibrated range and the accuracy is not guaranteed to be within specifications.

Included in Multi-Channel Project (ecd14x.lvproj) and Basic Project (ecd14x Basic Demo Rev 2.lvproj) .

*Installation Levels:*

- Real-Time Processor

*ECD14x Clock and FIFO.vi*



Takes data from slot 1 module and puts data into FIFO\_1.

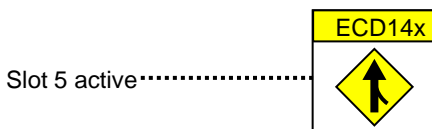
Two data types are available, measurement data and error data. The type of data in the output is determined by setting the “data type” variable (1=measurement data, 2=error data). See Error Checking section on page 18 for more information.

Included in Basic Project (ecd14x Basic Demo Rev 2.lvproj)

*Installation Levels:*

- FPGA Target

*ECD14x Combined.vi*



Combines data from 1 to 4 modules and puts data into FIFO\_1. Can be edited to improve efficiency if fewer modules are used.

Two data types are available, measurement data and error data. The type of data in the output is determined by setting the “data type” variable (1=measurement data, 2=error data). See Error Checking section on page 18 for more information.

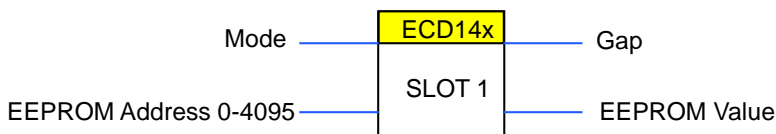
Also puts data from NI9201 into FIFO\_2. This portion can be deleted or edited for use with other NI modules.

Included in Multi-Channel Project (ecd14x.lvproj)

### *Installation Levels:*

- FPGA Target

### *ECD14x Slot (X) Driver.vi*



A separate driver is provided for each of slots 1-4. Can be modified by user for slots other than 1-4.

Slot 1 Driver included in Basic Project (ecd14x Basic Demo Rev 2.lvproj)

Slot 1-4 Drivers included in Multi-Channel Project (ecd14x.lvproj)

### *Installation Levels:*

- FPGA Target

## **Support Files**

### *ECD14x Slot.ctl (Multi Chan Demo)*

A simple file that provides for easy selection of the slot number.

### *ECD14x Counts & Gaps.ctl (Basic Demo, Multi Chan Demo)*

This control provides a space to store and display the information returned by the GetSensitivity VI.

### *ECD14x Data Select.ctl (Multi Chan Demo)*

This is a required control in the demo VIs. The control selects the type of data to be acquired from the module:

- EEPROM (0) – Direct EEPROM readings for calibration information and other module data
- Probe (1) – Measurement data stream

- Status (2) – Status information including firmware rev level (upper byte) and error codes (lower byte)

In the Basic Project, the Data Select control is embedded in the VIs and not included as a separate ctl file.

## **Required FPGA Target Level Configurations**

### *50MHz Derived Clock*

A 50MHz derived clock must be created.

### *FIFO*

A FIFO named FIFO\_1 must be created for the data transfer to the VIs. The FIFO must have these characteristics:

Type: Target to Host DMA

Data Type: U64

Number of Elements: 1023 (default)

# Specifications

---

Parameter		Specification	Notes
Resolution @ 15 kHz (Typical) <sup>1,2</sup>	nonferrous	0.006 to 0.008%F.S.	See calibration sheet for specifics
	ferrous	0.007 to 0.1%F.S.	
Linearity <sup>1</sup>		±0.2%F.S.	
Error Band <sup>1</sup>		±0.4%F.S.	
Driver Operating Environment		4°C-50°C, IP40	
Probe Operating Environment	Standard Probes	-25°C to +125°C, IP67	
	High Temp. Probes	-25°C to +200°C, IP63	

<sup>1</sup>Actual values depend on probe and range and are listed on the calibration certificate shipped with the product. Contact Lion Precision for replacement certificates.

<sup>2</sup>In High EMI environments (10 V/m), output noise levels may rise causing resolution to be reduced to 0.2%.

## Specifications Continued

Parameter		Specification		Notes
Temperature Coefficient (Driver)	nonferrous	U8 Probe	±0.04% F.S./°C	Over 15°C to 50°C temperature range
		U12 Probe		
	ferrous	U8 Probe		
		U12 Probe		
Temperature Coefficient (Probe)	nonferrous	U8 Probe	±0.01% F.S./°C	Over 15°C to 65°C temperature range
		U12 Probe	±0.02% F.S./°C	
	ferrous	U8 Probe	±0.04% F.S./°C	
		U12 Probe	±0.03% F.S./°C	