**Mechatronics Module 4: Sensors and Actuators**

# Demonstration of a sensor’s data acquisition

This module demonstrates the use of National Instruments LabVIEW for developing data acquisition for sensors. Data from an acceleration sensor will be read and plotted through a National Instruments CompactRIO and desktop PC running LabVIEW

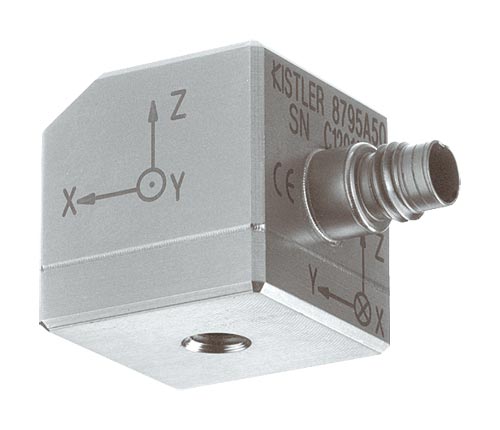
# Introduction to LabVIEW

LabVIEW is a graphical programming environment used to develop sophisticated measurement, test, and control systems using graphical icons and wires that resemble a flowchart. It offers integration with of hardware devices and provides built-in libraries for analysis, data visualization, and for creating virtual instrumentation.

LabVIEW ties the creation of user interfaces (called front panels) into the development cycle. LabVIEW programs/subroutines are called virtual instruments (VIs). Each VI has three components: a block diagram, a front panel and a connector panel. The last is used to represent the VI in the block diagrams of other, calling VIs. Controls and indicators on the front panel allow an operator to input data into or extract data from a running virtual instrument.

# Introduction to the hardware

## Accelerometer

The sensor which will be used for this demonstration is an accelerometer. It is a triaxial acceleration sensor whose three outputs measure acceleration along three axes: x, y, and z directions.

The sensor contains quartz crystal sensing elements that produce an electric charge in response to force due to the acceleration. Each of the accelerometer’s three sensing elements is internally connected to a Piezotron microelectronic circuit that converts the charge signal from the quartz piezoelectric elements into a useable high level voltage signal.

Figure : Kistler K-Shear ® Accelerometer Type 8695A50M8

## CompactRIO

Figure : CompactRIO Chassis

CompactRIO is a reconfigurable embedded control and data acquisition system. The CompactRIO system includes I/O modules, a reconfigurable field-programmable gate array (FPGA) chassis, and an embedded controller. Additionally, CompactRIO is programmed with NI LabVIEW graphical programming tools and can be used in a variety of embedded control and monitoring applications. The CompactRIO chassis includes a number of slots for modules with a number of input and output channels that can be connected to the sensors. Different types of modules allow for measurements such as voltage, current, and specialized sensors such as accelerometers.

Figure : NI Module 9233

The accelerometer’s three outputs will be connected to three channels on a NI 9233 module which provides the signal conditioning needed for the integrated electronic piezoelectric sensor.

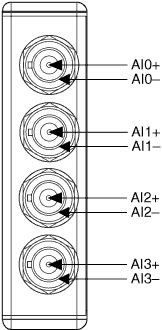
An Ethernet port connects the CompactRIO to the desktop PC.

Figure : NI 9233 Connector Assignments

The NI 9233 module has three ports labeled AIO through AI3 (fig. 4). Since the accelerometer has three output signals, three of the four channels will be used.

## FPGA

Field-programmable gate arrays (FPGAs) are reprogrammable silicon chips. In contrast to PC processors, programming an FPGA rewires the chip itself to implement its functionality rather than run a software application. A LabVIEW program running on the FPGA manages the input and output of data through the modules in the compactRIO chassis.

# LabVIEW Data Acquisition Model

The LabVIEW data acquisition program includes the code needed to read and display data from the sensor. Only the FPGA can read the sensor through the CompactRIO’s input/output modules and includes fast I/O response times, however the FPGA code is limited in some functionality including visual graphing of data. Therefore, the data acquisition will consist of two programs running simultaneously: one on the FPGA to read data from the sensor, and another on the CompactRIO chassis processor which acquires the data from the FPGA and converts it for display.

## Sensor acquisition FPGA block diagram

Figure : FPG Block Diagram

Above is the block diagram of the FPGA code. The gray boxes are a **flat sequence structure** which controls the order of operation of the code. The sequence proceeds from left to right: after all events in the left box are run, the program proceeds to the next panel to the right. The left panel includes the start command to the module.

The program input the data from the accelerometer as the specific time rate. The data that the **FPGA I/O Node** receives is sent to the **Build Array Function** to concatenate its multiple signals into a 3-dimensional array.

**FPGA I/O Node*:*** performs specific FPGA I/O operations on FPGA targets. This is used for reading output data from the sensor and sending start and stop commands. The loop stop control is wired from an **or** gate which indicates that the FPGA can either be stopped from the stop button on its front panel or if the *Stop FPGA* variable is true. This allows it to be also stopped from the second VI running on the CompactRIO.

Inside the second panel is a **while loop*.*** It repeats the subdiagram inside it until the conditional terminal, an input terminal, receives a particular Boolean Value (given by the stop command which ends execution of the loop. The stop signal is also wired to the I/O node to send a stop command to the data acquisition module.

**Loop timer*:*** waits the value specified in between loop iterations. This function is called in the loop to control the loop execution rate, which sets the frequency at which the program reads data from the sensor (scan rate). The highest rate possible is limited by the capabilities of the sensor, module, and system as a whole, which decreases with each additional channel that needs to be read.

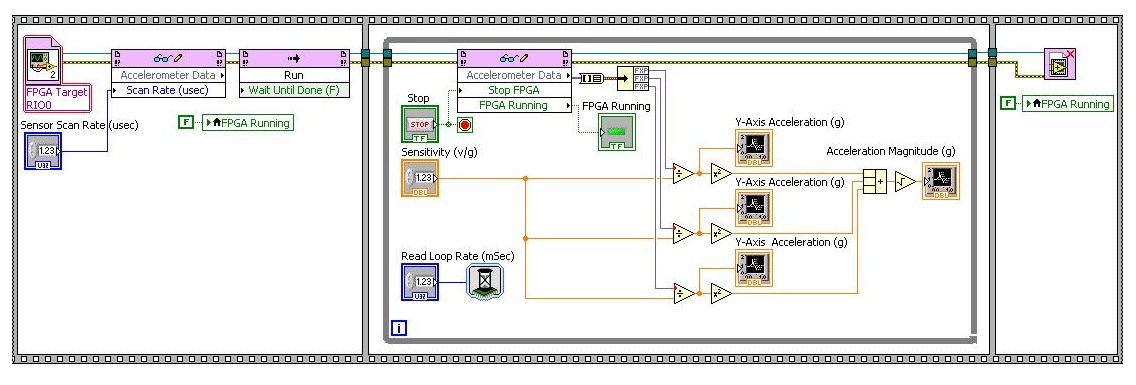
## Sensor acquisition FPGA front panel



Figure : FPGA Front Panel

The front panel allows manipulation of controllable parameters in the code, such as the scan rate. The *FPGA Stop* is the button for stopping the program. You can click on it to stop the measurement. The *Accelerometer Data* measures the exact the data from the accelerometer sensor. *FPGA Running* indicates that the FPGA program is active and *Stop FPGA* indicates that a stop signal has been sent.

## Accelerometer data read block diagram

Figure : RT Block Diagram

The second program is run on the CompactRIO’s real-time microprocessor. This program is set up to manage the running of the FPGA code: it opens, runs, reads data from , and finally closes the FPGA program.

In the first panel of the sequence structure, an ***open FPGA VI Reference*** functionopens the reference to the FPGA VI or bitfile and FPGA target specified, and is wired sequentially to later blocks witch pass the reference to the FPGA file to those blocks so that they may perform several operations on it: for example, a ***Read/Write Control Function*** passes the value of the sensor scan rate controller to the scan rate variable on the FPGA, allowing controls on the FPGA program to be controlled from the front panel on this one.

The **while loop** in the second panel repeats the reading of data from the FPGA. It acquires data from the FPGA program’s *Accelerometer Data* variable and splits it into its three separate signals (x, y, and z acceleration).

The accelerometer outputs its signals as a voltage, so each one is multiplied by a controllable setting to convert it into more readable units. In addition to the acceleration along each axis of the sensor, the total magnitude of the acceleration is calculated (). Each signal is then wired to a **waveform chart** to be displayed on the front panel.

The stop button ends execution of the loop and also sends the stop signal to the FPGA, stopping execution of both programs at once. After the loop is ended, it proceeds to the third panel where the **close FPGA VI reference** ends closes the reference to the FPGA VI and resets it.

## Accelerometer data read front panel

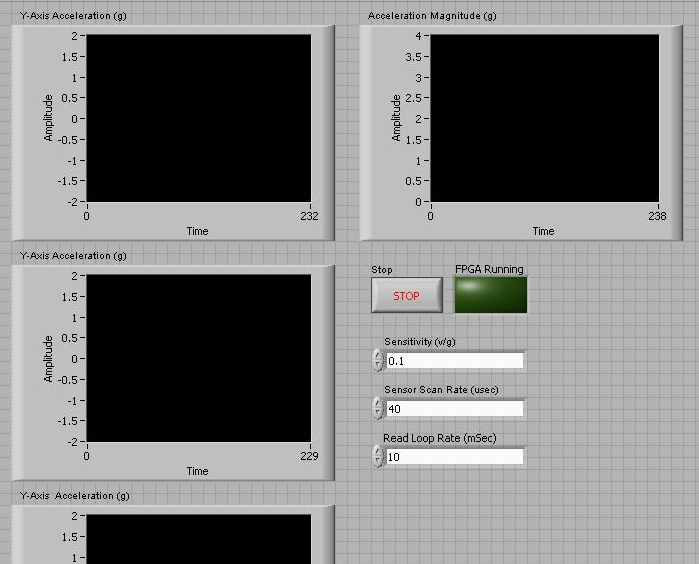


Figure : RT Front Panel

This panel visually displays data from the sensor and contains all controllable parameters. Since the controls are wired to the FPGA program, control of all aspects of the program is possible from this panel alone.

The data we get from the accelerometer is the voltage. So we need to transfer it to the acceleration. The *Sensitivity* is the relation of the voltage and the acceleration of this accelerometer. The sensitivity of the accelerometer is 100 mV/g, or 0.1 V/g. Therefore, the output voltage signal from the sensor is divided by 0.1 to give a value of acceleration in g. The sensor has a range of ±50 g and outputs voltage in a range of ±5 V.

# Instructions

Figure : Sensor connected to NI 9233 module in slot 7

1. Connect the hardware:

* The NI 9233 I/O module should be placed in slot 7 of the CompactRIO chassis (fig. 9).
* The sensor cables are connected to the terminals in the 9233 module as shown in the table below:

|  |  |
| --- | --- |
| **Module Terminal** | **Sensor Axis** |
| AI0 | X (red) |
| AI1 | Y (yellow) |
| AI2 | Z (white) |

1. Open Labview project:

Figure : Project Explorer

* Open LabVIEW.
* From the menu bar in the *Getting Started* page, navigate to **File🡪Open Project**
* Open **C:\accelerometer-sensor\Accelerometer\_Sensor.lvproj**.

1. Compile FPGA

(This will reconfigure the FPGA chip to run the sensor VI. Note that this only needs to be done the first time the program is run and will last until a different FPGA program is used.)

* From the project explore items view, navigate to **Project: Accelerometer\_Sensor.lvproj🡪MSME-cRIO-05🡪Chassis🡪FPGA Target** and open **Sensor\_Acquisition\_FPGA.vi**.
* Run the program with the *run* arrow button on the upper toolbar.

Figure : Run button

* Wait for the VI to compile (this will take several minutes).
* Stop the program with the stop button on the front panel and close the VI.

1. Run sensor data acquisition:

* From the project explore items view, navigate to **Project: Accelerometer\_Sensor.lvproj🡪MSME-cRIO-05** and open **Accelerometer\_Data\_Read.vi**.
* Run the program with the *run* arrow button on the upper toolbar.
* Move the sensor and check the change in acceleration on the charts.
* When finished, stop the program with the **stop** button on the front panel.

# Questions

1. What are the output axes of the accelerometer?

1. What is the function of a **flat sequence structure?**

1. What is the sensitivity of an accelerometer that outputs a 4V signal in response to an acceleration of 20g?