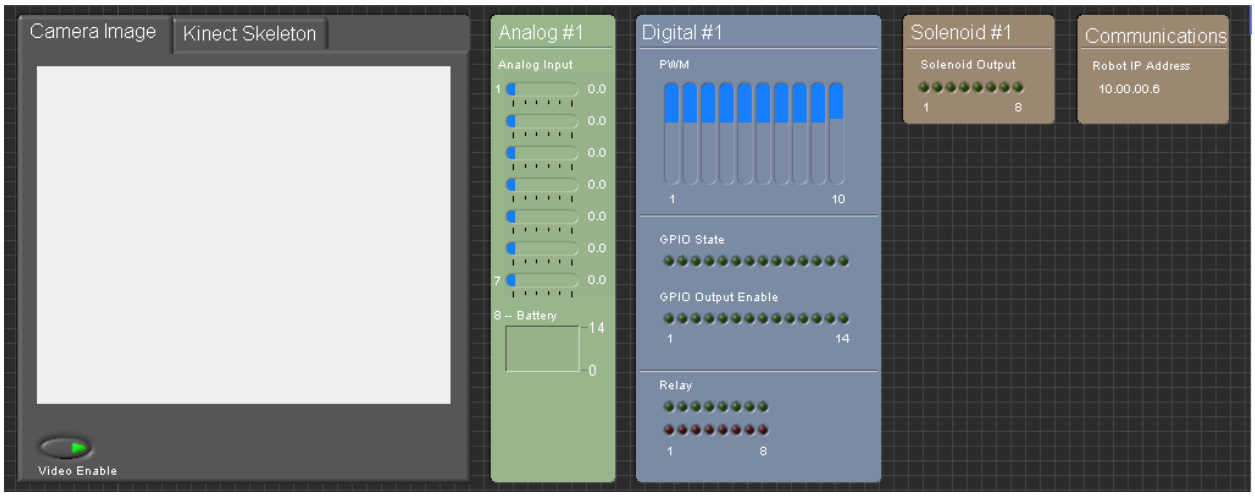
FRC Dashboard Tutorial

Contents

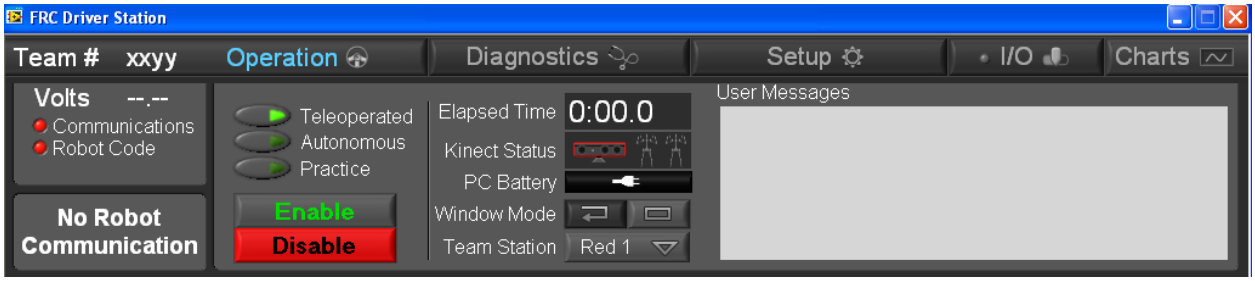
Introduction

In this tutorial, you will explore the FRC Dashboard, and learn how to effectively use the dashboard to troubleshoot your FRC robot project.

The dashboard displays information from the camera and cRIO-FRC modules, as well as communication information.



Below the dashboard you see the FRC Driver Station. The driver station displays information about the control system and can display messages sent from the robot in the User Messages Section on the Operation Page.



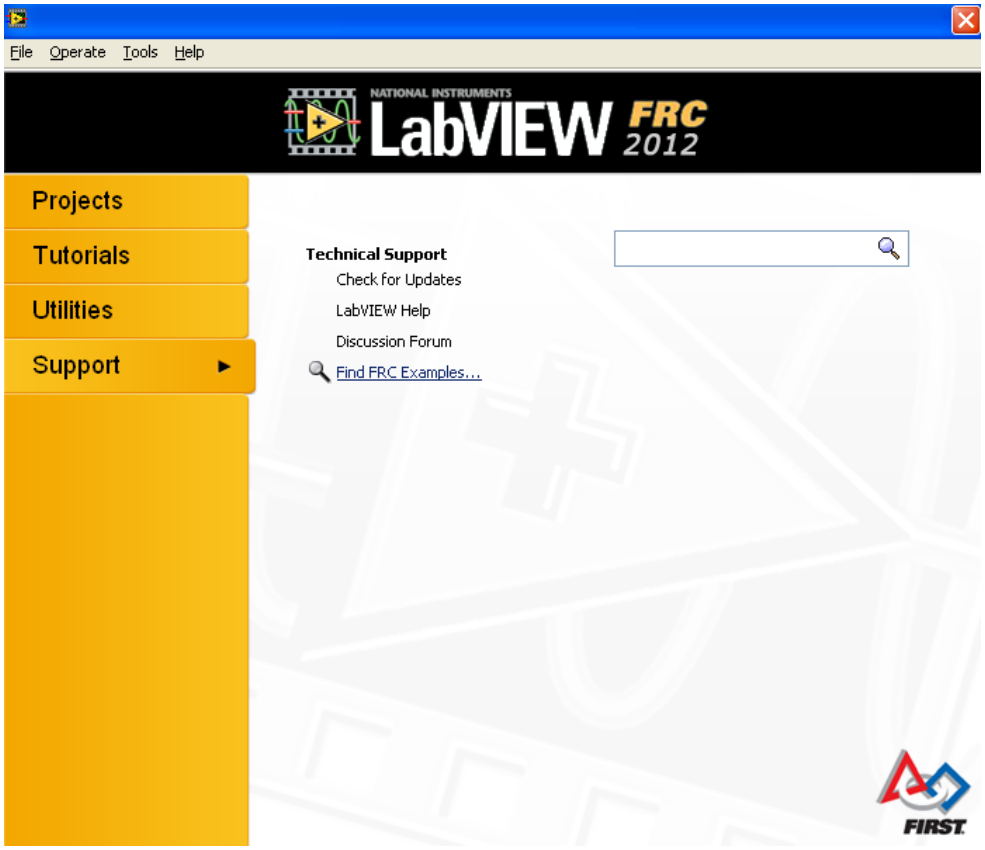
The dashboard is a great tool for testing – it shows you at a glance the state of any I/O channel and displays your own custom-defined data. We will discuss how to display data from both the driver station and the cRIO on your dashboard.

Throughout this tutorial, we will assume that you have all the necessary software for FRC installed on your computer, your cRIO is imaged, and the firmware on your driver station is up to date. If you don't have these steps completed, you can refer to the *Getting Started* tutorials for information and instructions.

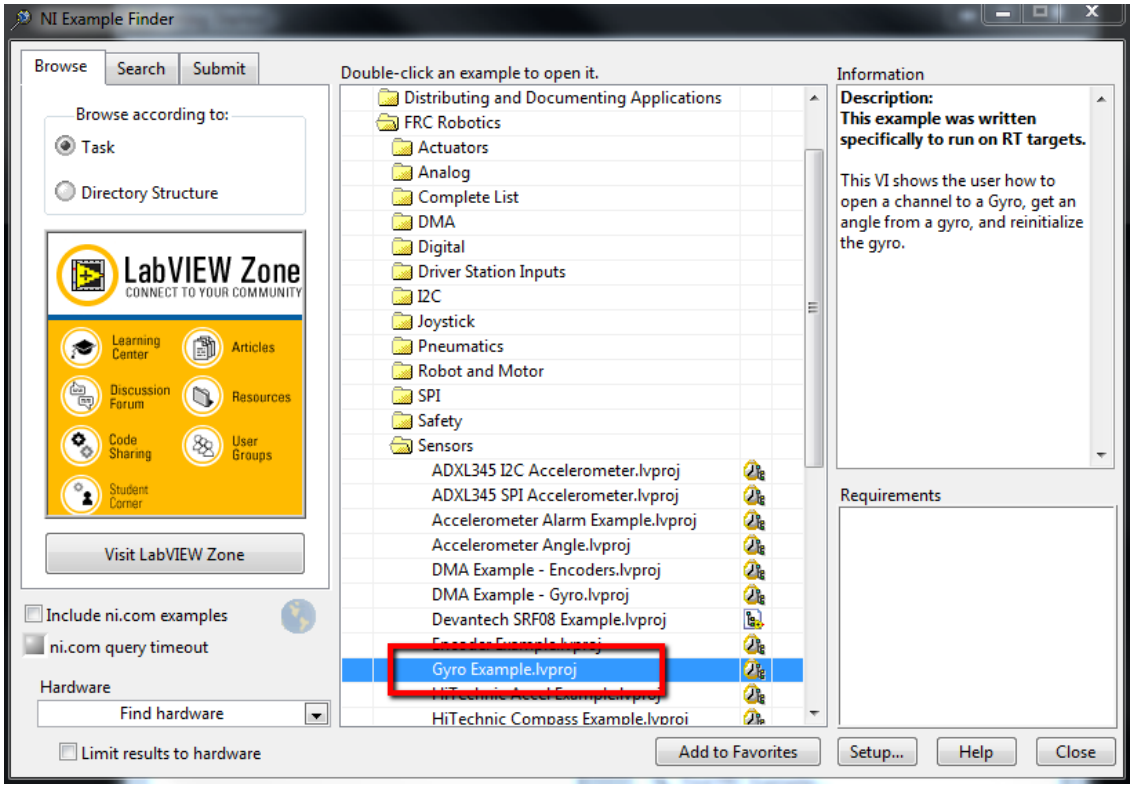
Testing the Gyro Example

Let's open up an example program that will generate data from the cRIO. We will be looking at the gyro example to demonstrate how to send directional data from the gyro sensor on the cRIO to the driver station and the dashboard running on your computer. This is a great way to monitor the gyro's functionality in different test conditions.

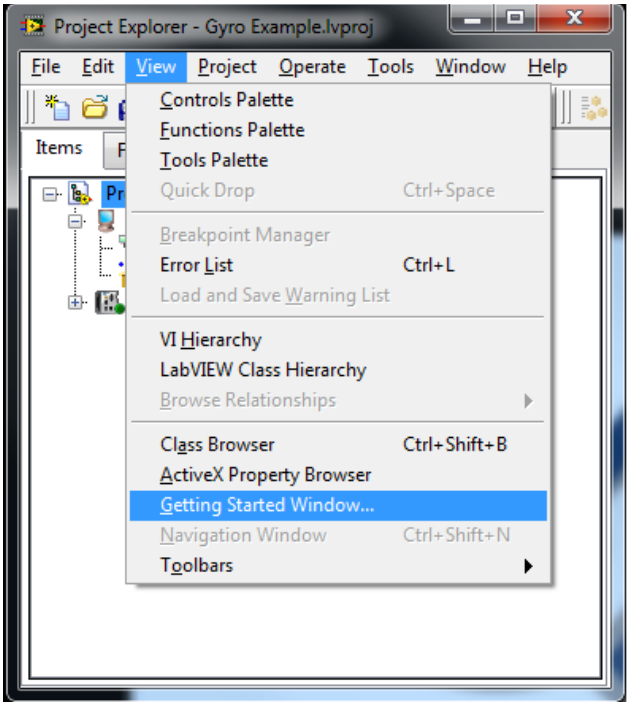
From the LabVIEW Getting Started window, select **Find FRC Examples** from the **Support** section.



From here, you can locate the **Gyro Example** by navigating to FRC **Robotics»Sensors»Gyro Example.lvproj**.

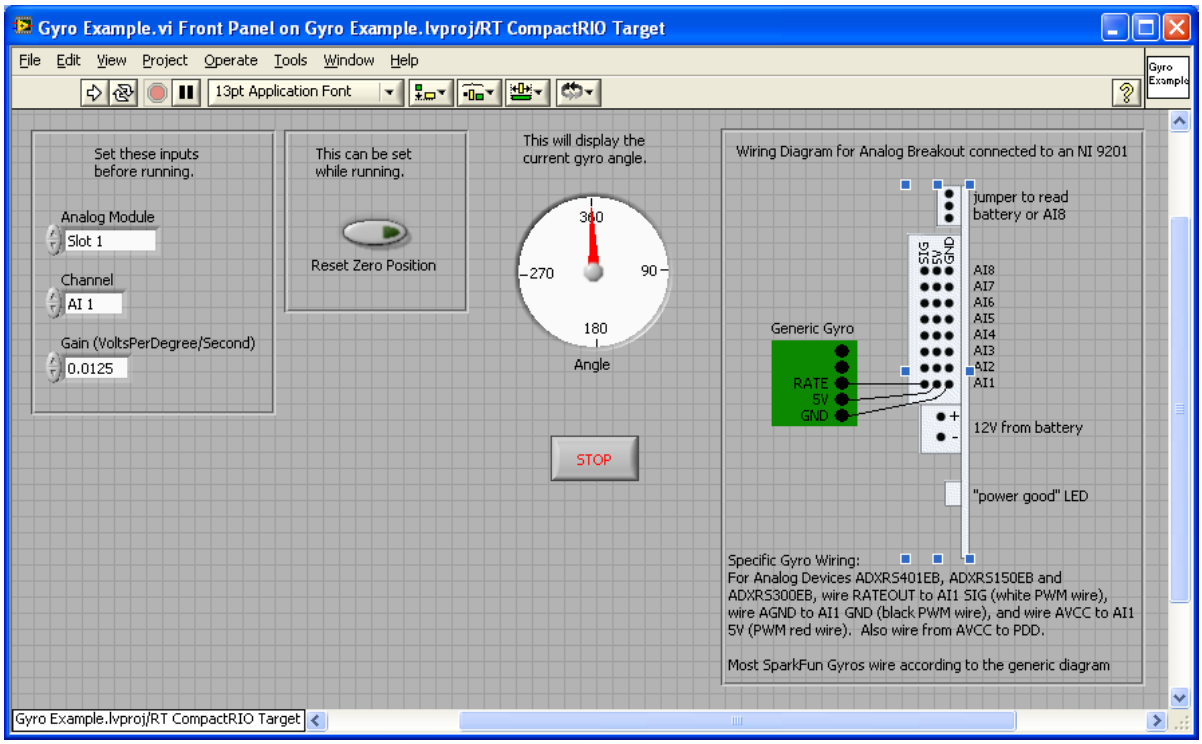


If you are already in another project or VI, you can access the Getting Started window at any time by selecting **View»Getting Started Window…**

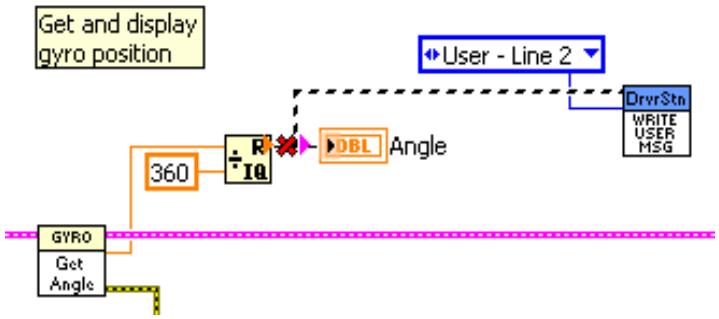


Once the example project loads, Select **File»Save As** and save a copy of this project to a new folder. Name the folder *Dashboard Example*.

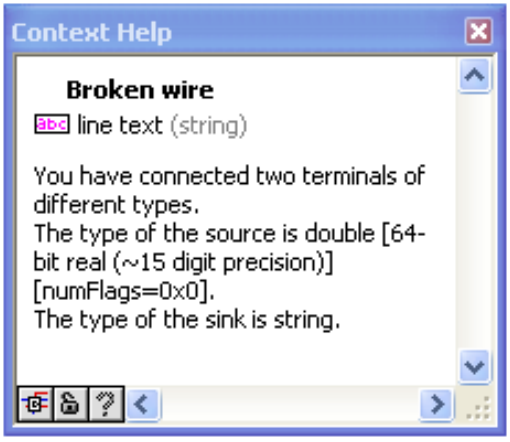
Double-click on **Gyro Example.vi** in the project to open it. This VI runs on the FRC cRIO and allows you to view the angle of the gyro. We are going to modify it to send information to the driver station running on the computer.



Go to the block diagram by pressing <Ctrl>+E or selecting **Window»Show Block Diagram**. Add a **Write User Message** VI from the **WPI Robotics Library»Driver Station** palette to the right of the *Angle* indicator. This VI allows us to add user data to the communication packet being constructed and sent to the computer. Right-Click the **Line** input of the Write User Message VI and select **Create»Constant**. This allows you to choose which line of the User Messages section of the driver station we will write the data to. Set the constant to **User – Line 2.** Wire the **Angle** output from the Get Angle VI to the **User Data** input of the Write User Message VI to have the Driver Station send this information to the dashboard. You'll notice that there is a type mismatch between the Angle output (a double) and User Message inputs (a string).



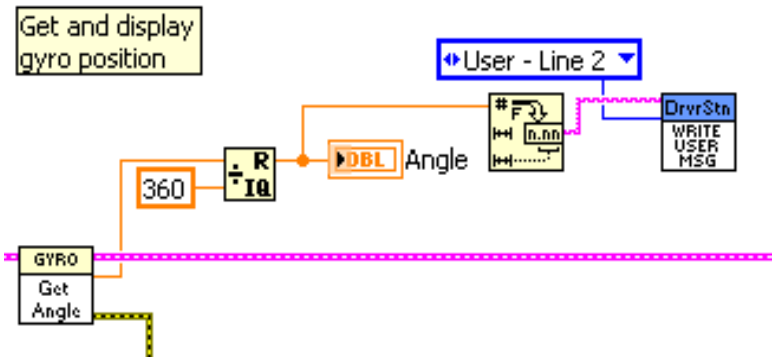
You can see the details of this mismatch by hovering the mouse over the red X and viewing the Context Help window (press <Ctrl>+H to open the Context Help window).



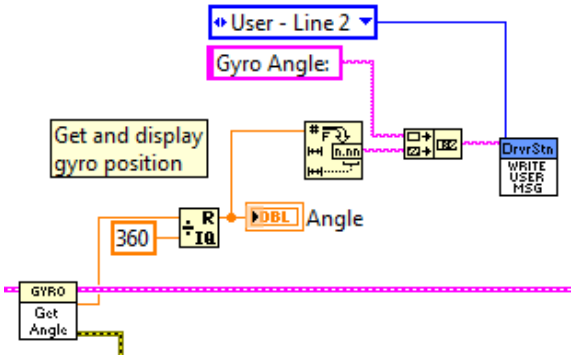
To remedy this, we will need to put our data in a form that can be interpreted by the either the driver station or the dashboard.

Sending information to the Driver Station

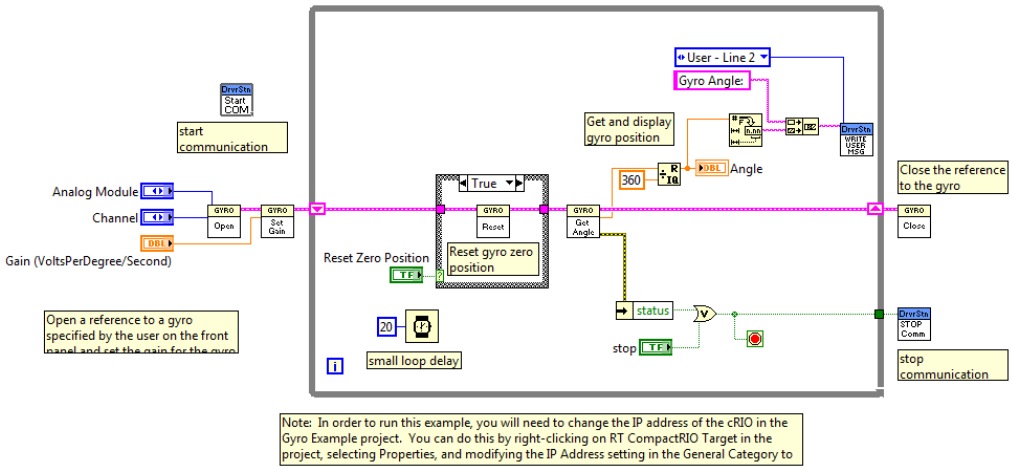
To send data to the User Messages section of the driver station, we'll need to convert it to a string data type as this is the only data type that the driver station accepts for messages. Add a **Number to Fractional String** function from the **Programming»String»String/Number Conversion** palette. Make more room on the block diagram if you need it.



To clarify the information we are sending to the driver station we can label the data we are sending. Add a **Concatenate Strings** function from the **Programming»String** palette. Wire the converted gyro data to the second input of the concatenate stings Right-click the first input of the concatenate strings function and select **Create»Constant.** You can now use that constant to label the data that it is paired with.



After wiring these VIs together as shown, we now have a VI that will send information to the User Messages section of the dashboard.

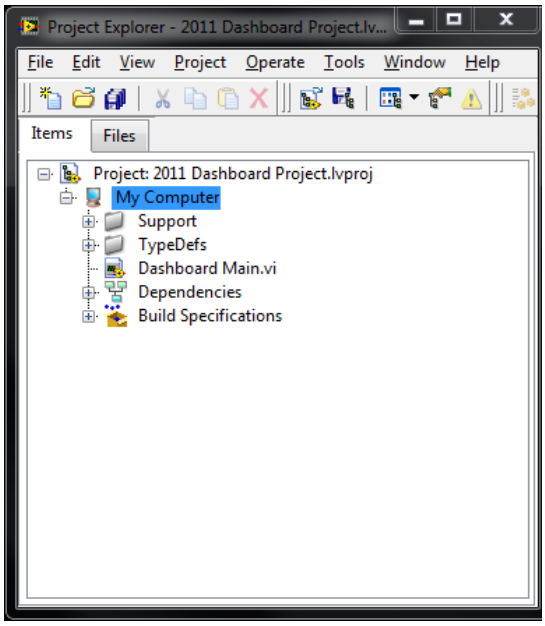


We had to convert the numeric to a string to change the data returned from the gyro into a data type that the driver station communication VI accepts. Save and run the VI.

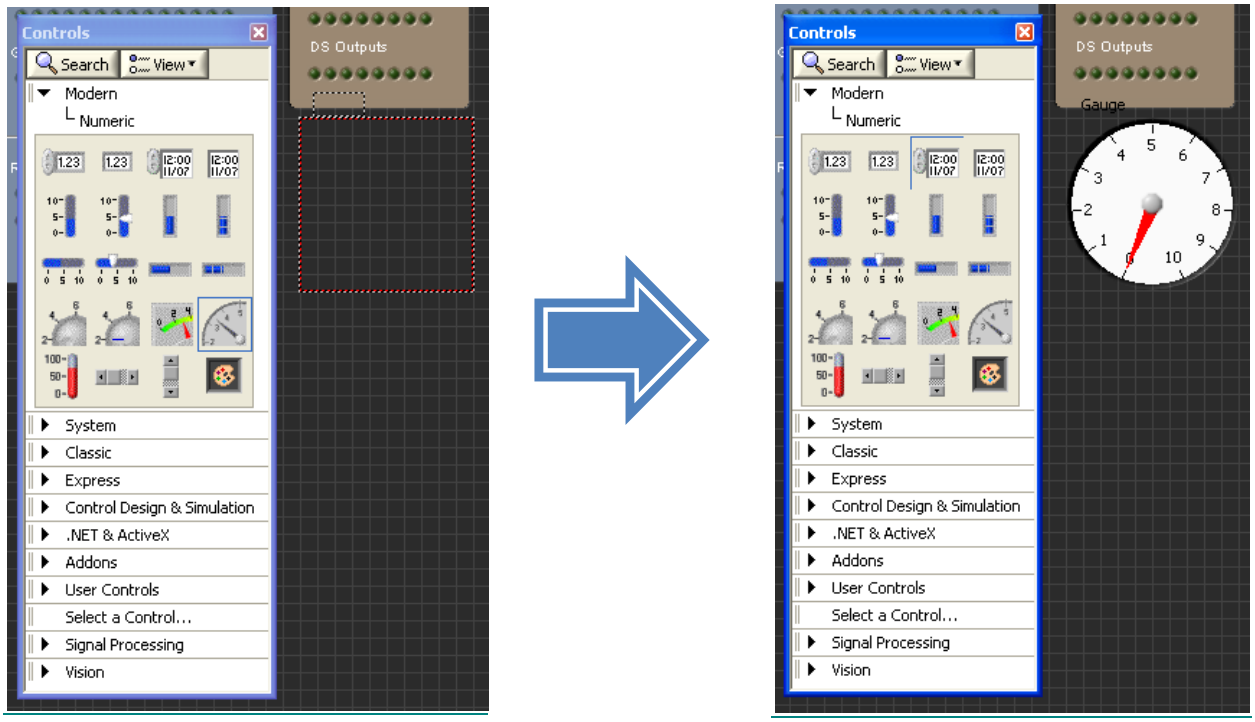
Sending information to the Dashboard

Since we have now modified the gyro example to send angle data to the driver station, let's look into how we can set up the dashboard to receive and display the data.

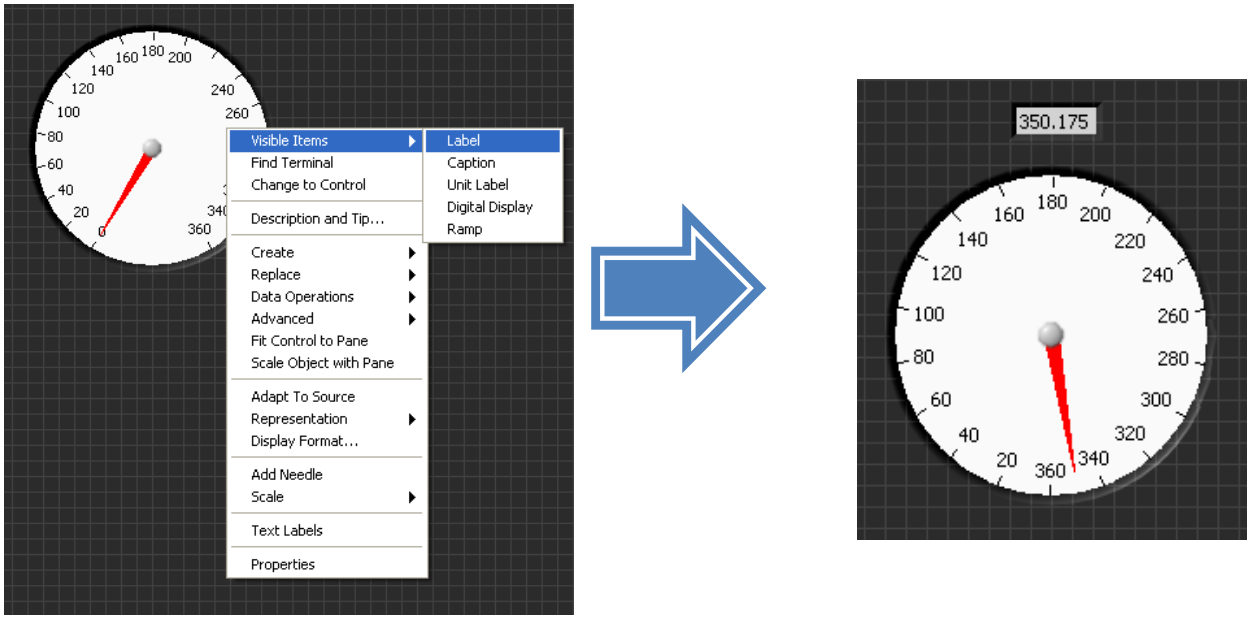
Open a new FRC Dashboard Project and a new FRC cRIO Robot Project. To stay organized, it may be a good idea to save these projects in the same directory as your modified gyro example project. The FRC Dashboard Project will already have a Dashboard Main VI. Open that VI.



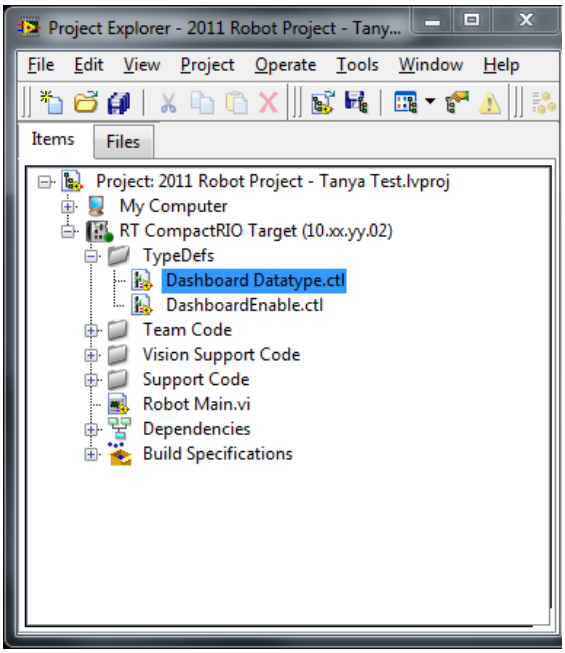
We will add a gauge indicator to the front panel to make reading the angle information easier. Now, let's add a Gauge indicator below the communications section of the dashboard. The gauge can be accessed from the **Modern»Numeric** palette



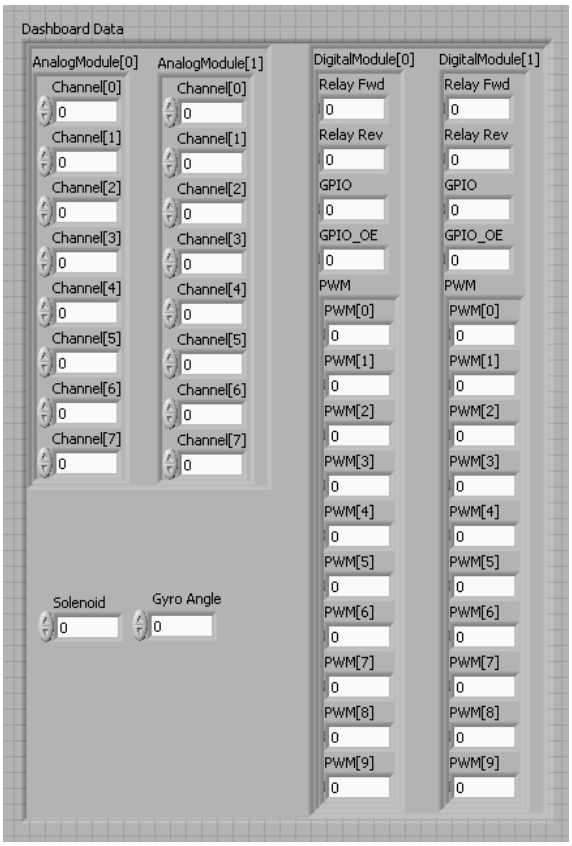
Adjust the maximum of the scale to **360** degrees. Next, align 360 with the 0 point. Adjust the size of the gauge and the appearance as you see fit. You can add a Label, digital display, and other features by right-clicking on the gauge and selecting **Visible Items** from the shortcut menu. Adding a digital display lets you see the exact value of the gauge.



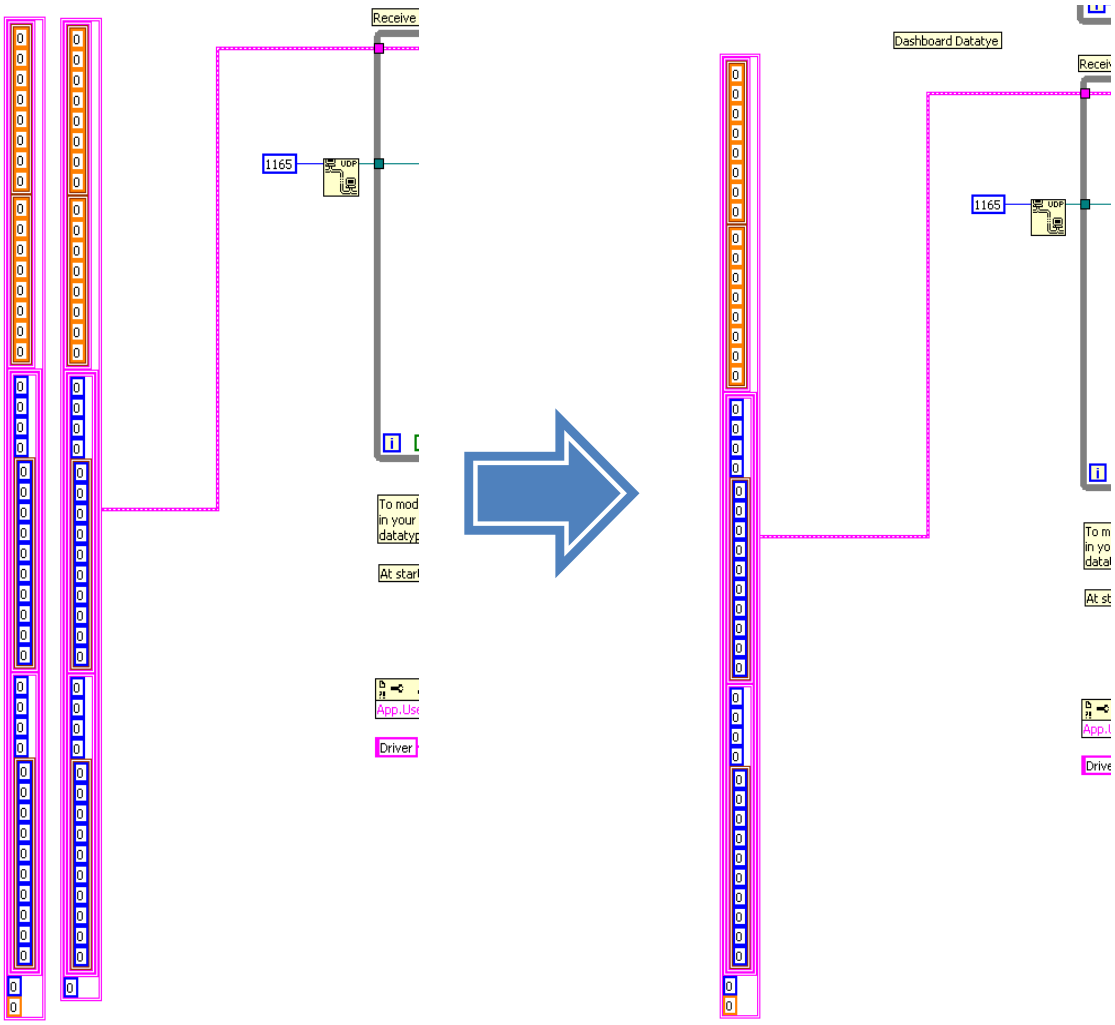
Now we need to configure the Dashboard DataType type definition to have a new data item for the gauge. The **Dashboard Datatype.ctl** can be found in the FRC cRIO Robot Project as seen below. Open this file from the project we created earlier.



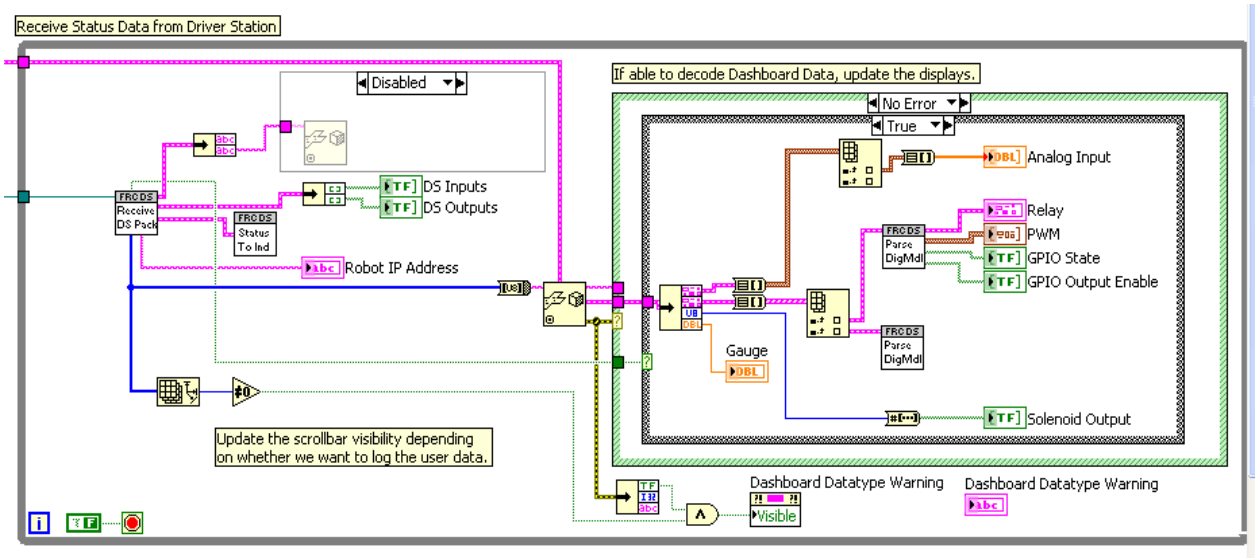
Add a numeric control from the next to the solenoid control. The numeric control can be found on the controls palette under **Modern»Numeric**. Rename the control to **Gyro Angle.** We have now added a new data item to the Dashboard Data type.



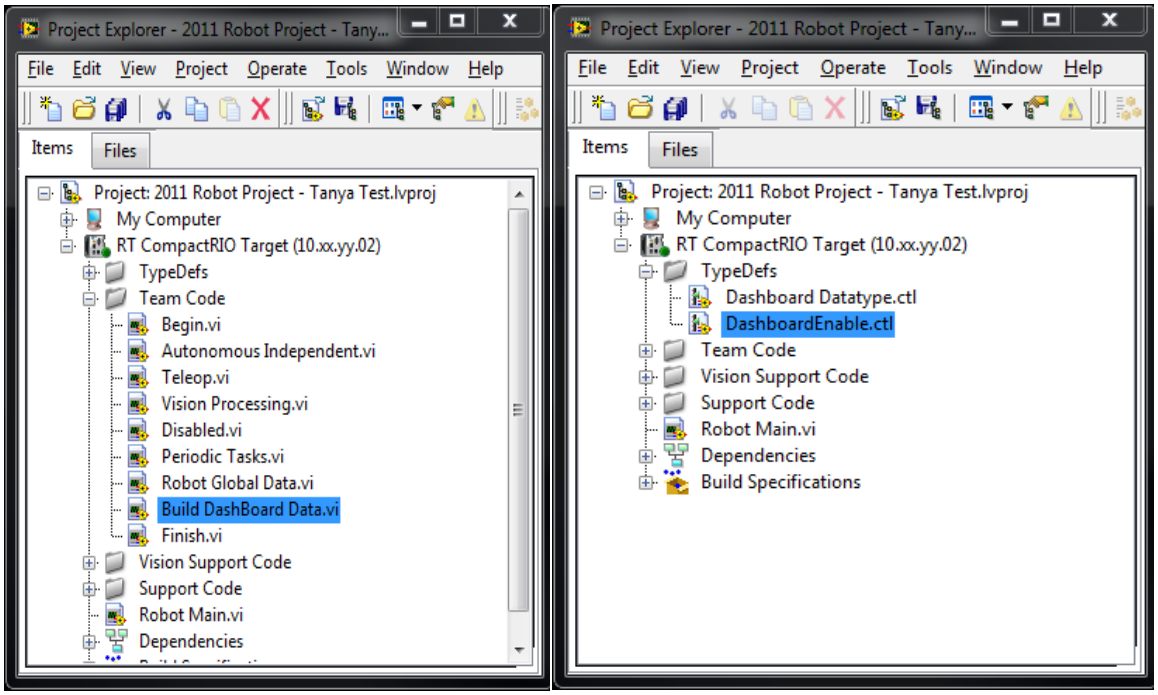
Open the Dashboard Main VI again and go to the block diagram. On the left you'll notice a large cluster. This is the Dashboard Data type. Drag the new updated Dash Board Data type from the FRC cRIO Robot Project window to the block diagram. You can see a new double data type constant at the bottom on the cluster. This is our new Data Item for the Gyro Gauge. Delete the old Data Type and wire the new data type.



Go to the block diagram and move the Gauge indicator inside the middle while loop. Wire the new double output and the gauge indicator together as shown below.

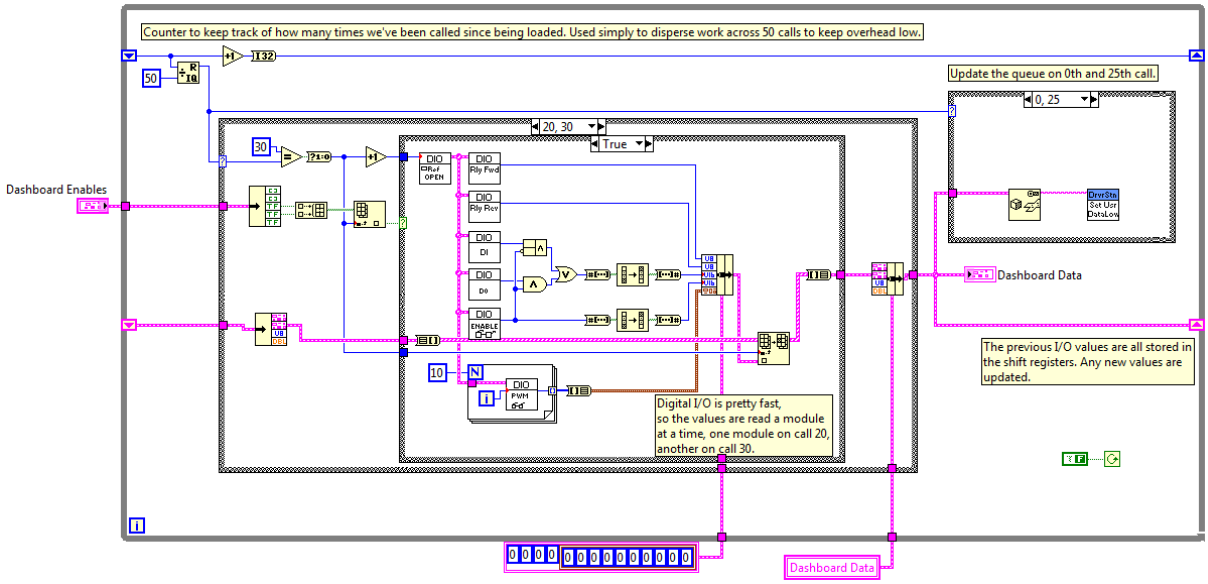


We've now configured the FRC Dashboard Project to receive and display data from our gauge. Next we'll take a look at how to supply that data from the FRC cRIO Robot Project to the Dashboard. Going back to the FRC cRIO Robot Project we need to open the Build Dashboard Data VI and the DashboardEnable.ctl type definition.

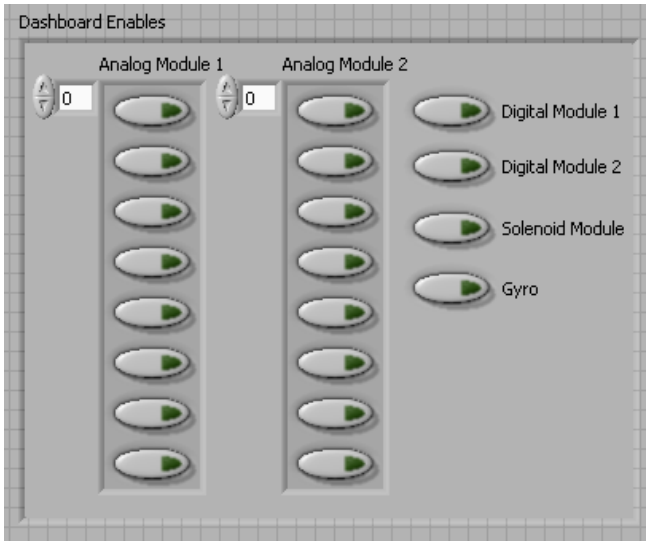


The Build DashBoard Data.vi compiles all the data from your robot and sends it to the driver station. In the center of the while loop there is a case structure that operates off of counter. As the count cycles 0 through 50 it runs different cases that collect different data. For instance cases 0-15 are analog inputs and case 40 is solinoid input. The DashboardEnable type definition allows you to select which cases in

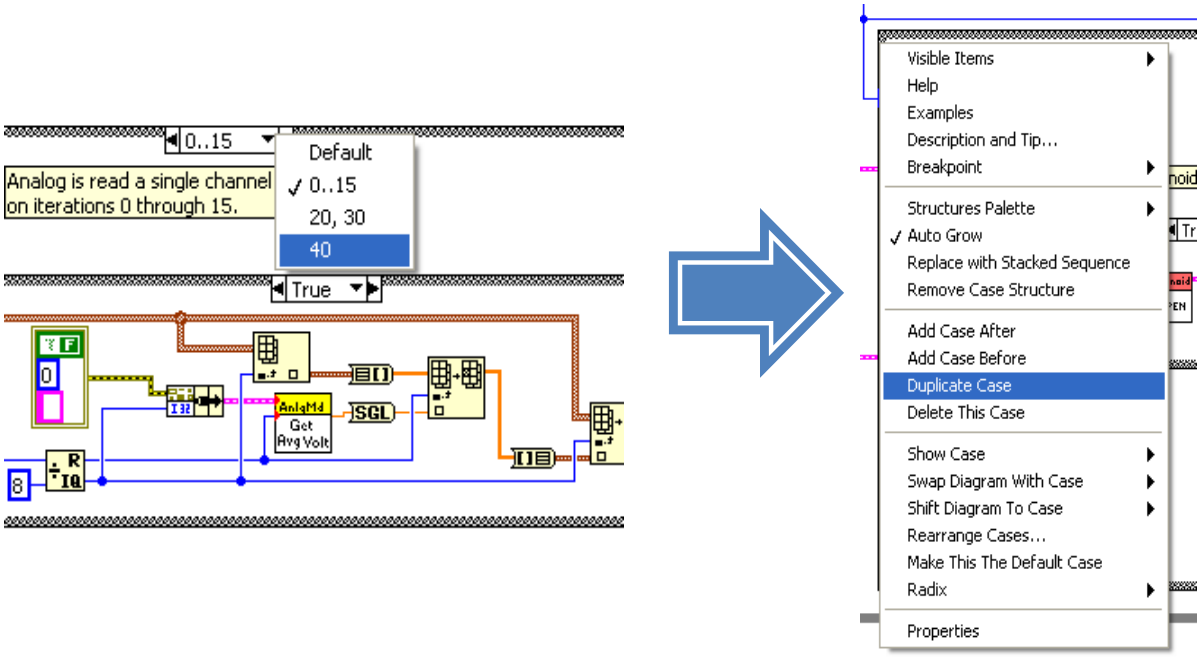
the Build Dashboard Data VI will be checked . We'll be creating a new case for the Gyro data collection and adding a new control to the the DashboardEnable type definition.



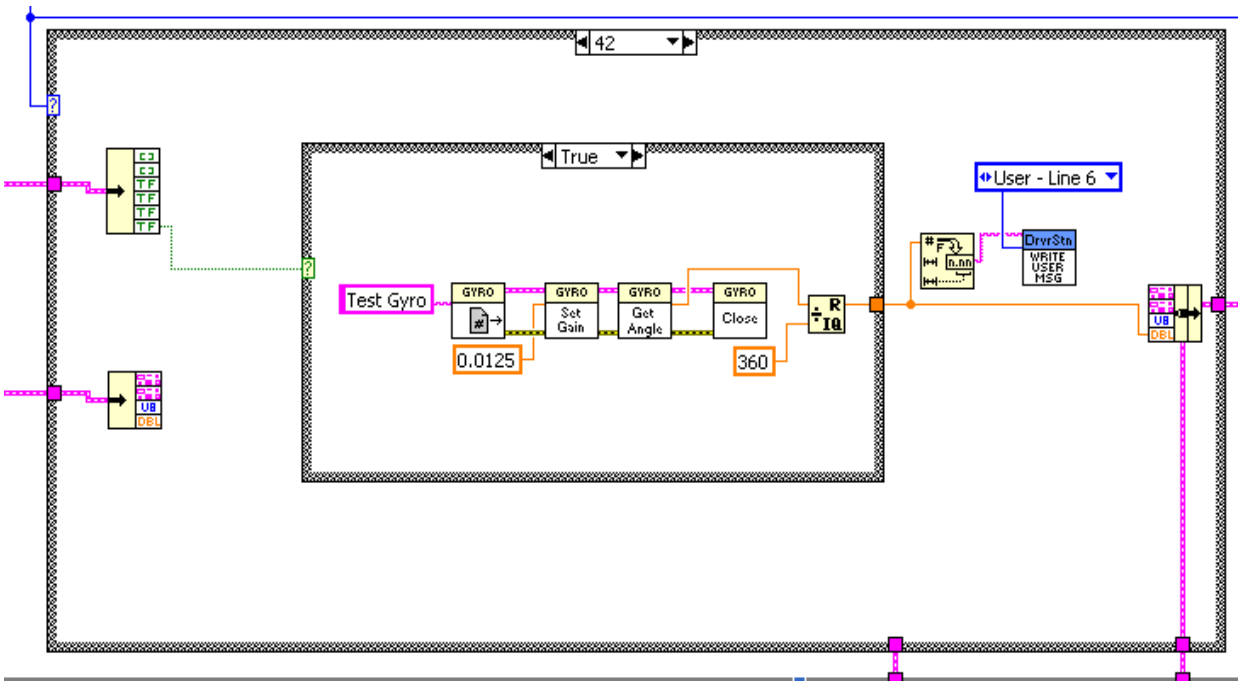
First, let's update the DashboardEnable type definition. In the type defintion add another push button to the front panel from the controls palette under **Modern»Boolean**. Rename the push button to **Gryo.** We now have the ability to turn the gyro data collection on and off in the Build Dashboard Data VI.



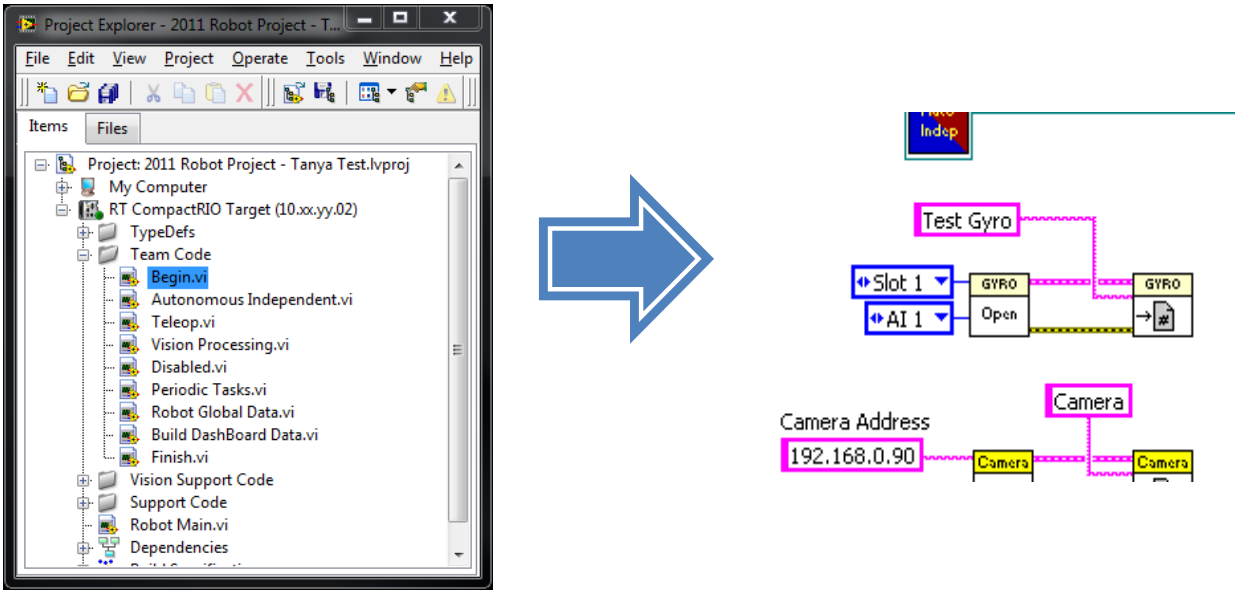
Going back to the Build Dashboard Data VI, select case 40 and then right-click the case border and select **Duplicate Case**. This creates a copy of case 40 and name it 41, rename the case to 42.



Using the image below as a template, edit case 42 to include the Gyro Data collection from the Gyro Example.



The difference between this code and the Gyro example is that we will be initializing the Gryo measurements by moving the Gyro Open VI to the Begin VI and creating a reference to it named **Test Gyro.**



We've now configured the FRC Robot Project to acquire data from our gyro and send it to the dashboard. Build the Dashboard project to compile your new dashboard executable.

Running the Dashboard

To tell the driver station which dashboard executable you want to run navigate to **C:\Documents and Settings\All Users\Documents\FRC DS Data Storage.ini.** This .ini file has a file path item called **DashboardCMD Line.** That path points to the dashboard executable that will be loaded when you run the driver station. Change that file path to point to your executable.

Build and deploy the Robot project to the cRIO and open your driver station. The Gryo data should be updating to your new dashboard and placing data in the User Messages Section of the driver station.

Conclusion

By now you should have a good idea on setting up a dashboard test station for the I/O modules on the cRIO and transferring customized data to the dashboard and driver station. The dashboard has many uses when it comes to project research and development. It's a fantastic tool for the following use cases:

1. Monitoring systems in different operation conditions
2. Logging data points for statistical analysis while in the research and development stage
3. Gaining a better understanding on how components interacts with each other

Congratulations! You've now learned how to return and add user data to the driver station and dashboard! The dashboard also can be used to get information from the joystick, sensors, actuators, etc… This information can be invaluable when testing out a new aspect of your robot.