



FRC Team 704, SPG Warriors

Exercise 2: FRC encoder sensor and autonomous

Introduction:

This tutorial demonstrates how to get the cRIO-FRC up and running a simple autonomous mode with the wheel encoder sensors. It walks through the hardware setup and the programming necessary to set up a sensor and setup a simple autonomous mode program.

Goal:

In the following exercise, you will build a VI that will add or subtract two numbers, depending on the user specification, and display the result. This exercise requires the following hardware:

FRC cRIO and the five included modules

Driver Station

Motor

Jaguar motor controller

Digital Side Car

12 V battery

SH37 68 pin cable

Power Distribution Board

Two Ethernet Cables

14 gauge wire

USB Joystick

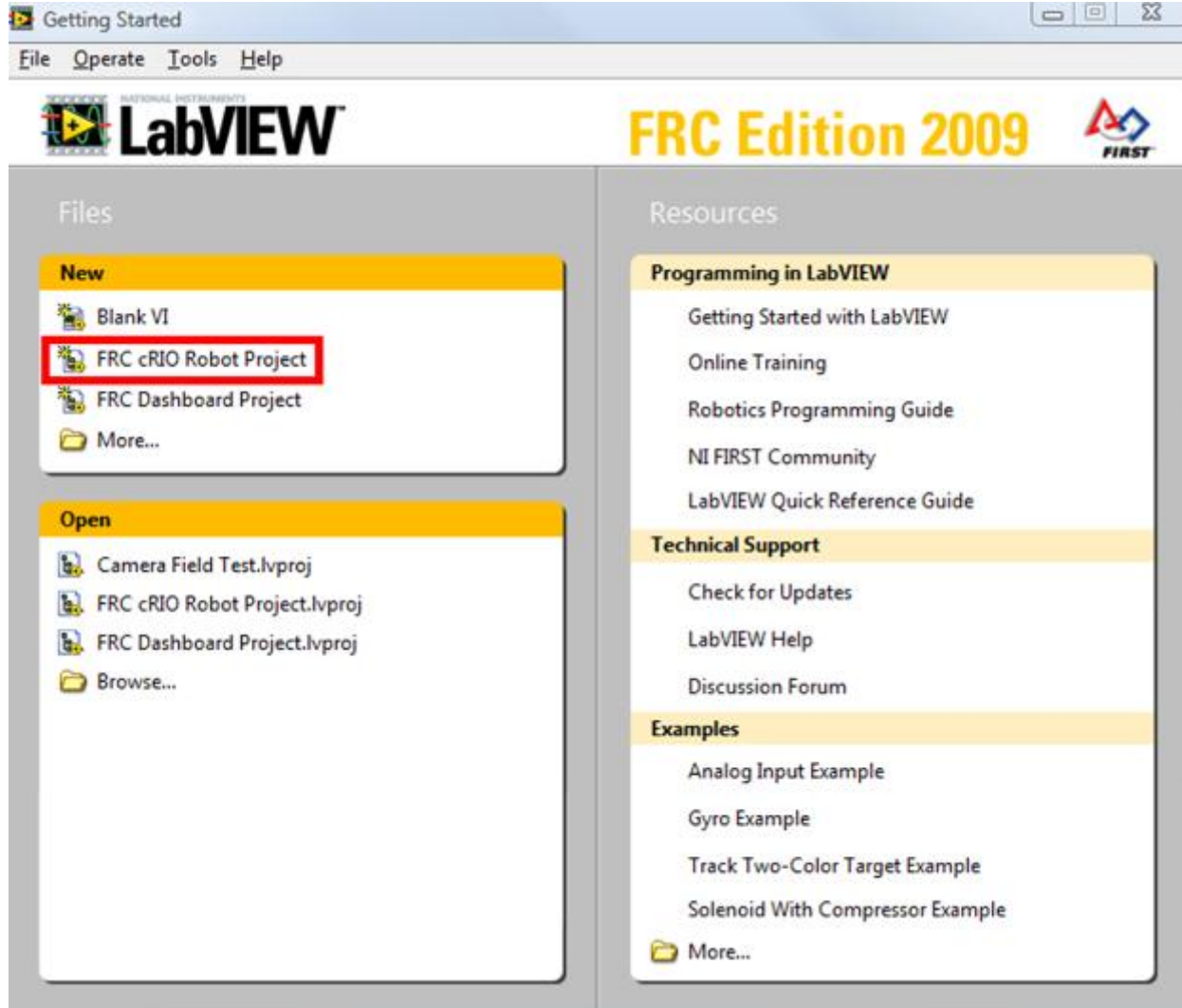
2 Wheel Encoder

NOTE: The robot should be in the basic configuration

Implementation:

- 1) Launch LabVIEW
 - Go to **Start>>Programs>>National Instruments LabVIEW 8.6.**
- 2) Create a New FRC cRIO Robot Project

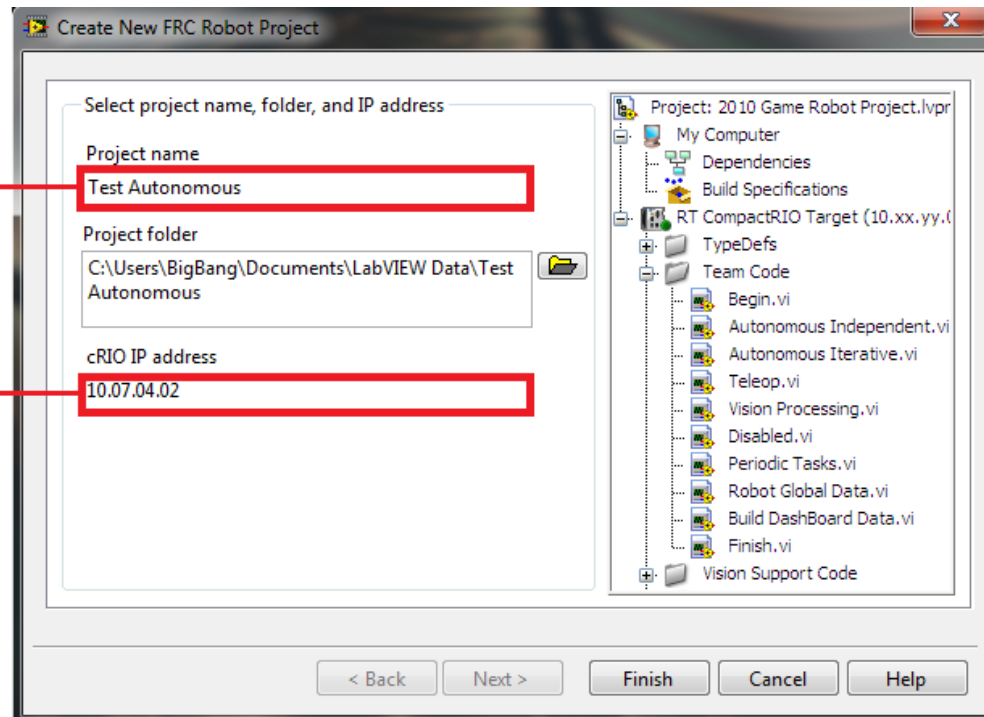
In the **Getting Started** Window, under **New**, select **FRC cRIO Robot Project**.



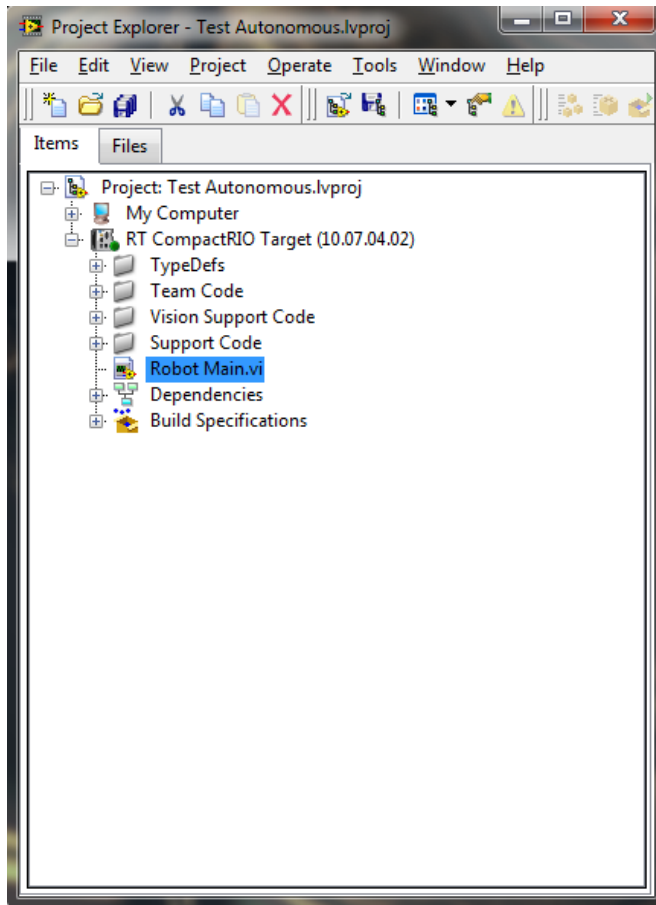
Name the project, set the save path, enter the cRIO-FRC's IP address (10.xx.yy.2), and select **FRC cRIO Robot Project**. The IP address is 10.xx.yy.2 where xx is the first 2 digits of the team number and the yy is the second 2 digits of the team number. i.e. for team 704 xx would be 07 and yy would be 04. Once you're done, click **Finish**.

what your project will be called

10.<2 digit team>.<2 digit team>.2 e.g. team 704 is 10.07.04.2

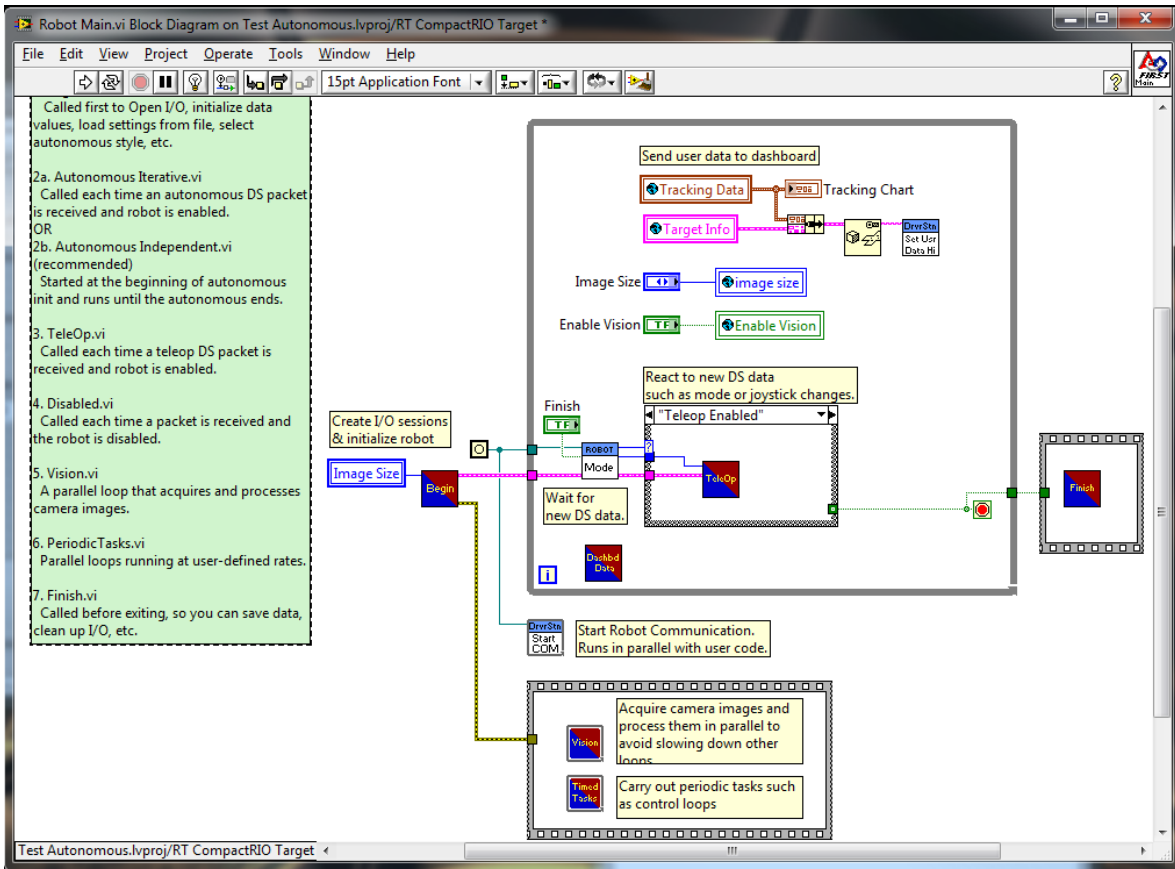


- 3) In the project window, double click the **Robot Main.vi**. followed by hitting <ctrl>e to open the block diagram.

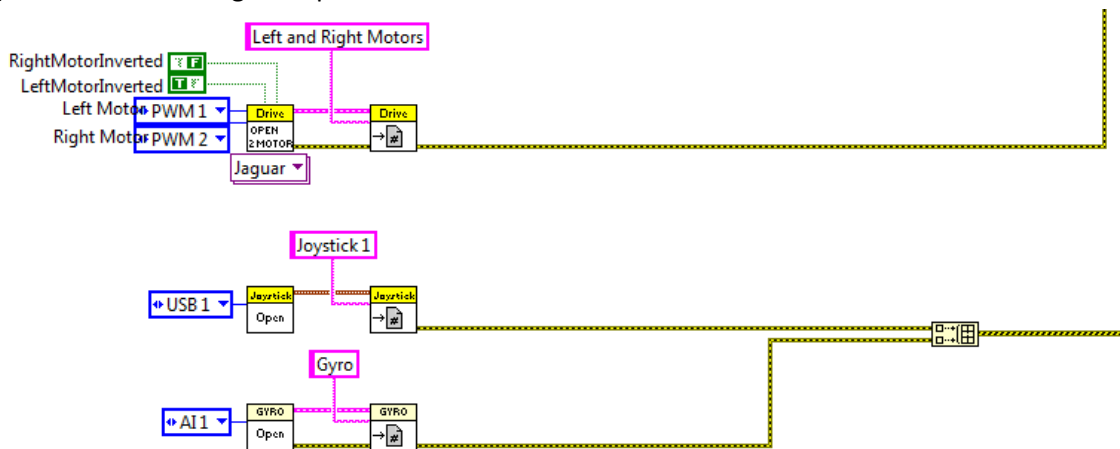


- All opening of sensors should be done in the **Begin.vi**, start by double clicking on the Begin vi.

***NOTE:** Read all the comments in the vi's coming from National Instruments, they contain useful information.

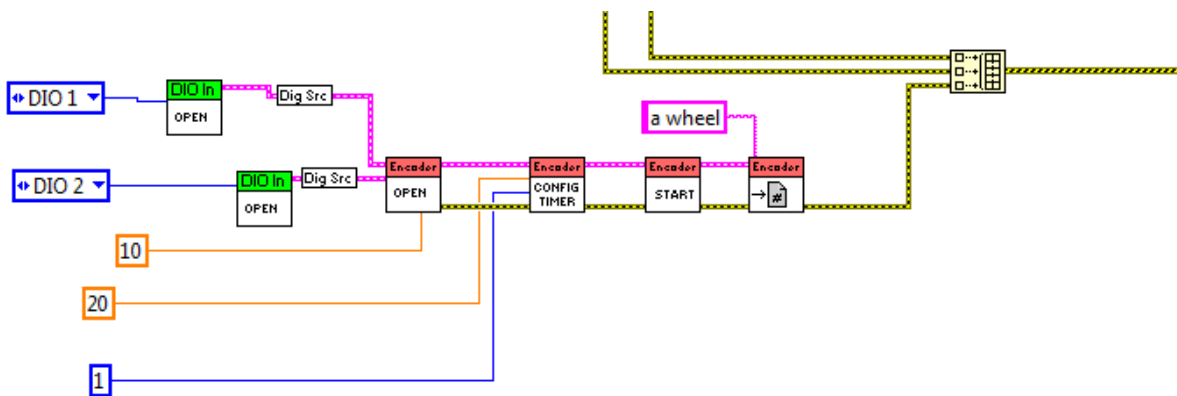


4) Add code to Begin vi open and start the encoder.



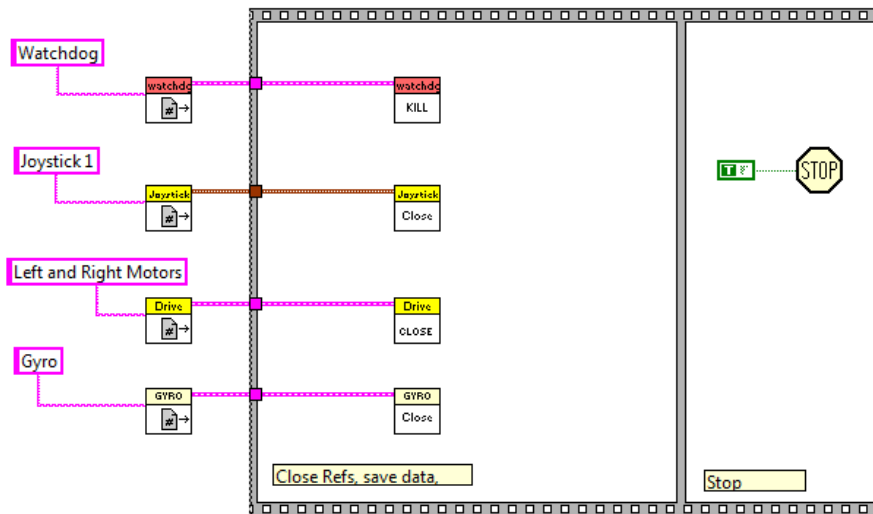
- At the bottom of the Begin vi, in the white space below the Gyro open, open a Digital IO WPI **Robotics Library>>IO>>Digital IO>>DigitalIO>>DIO Open.vi** and place the block on the diagram.
- Right click on Digital IO terminal and **Create>>Constant** and select **DIO 1**

- Create another Digital IO open.vi and create a constant for the Digital IO and select **DIO2**
- Hit <ctrl><space> and enter **Digital Source**. Then drag the **ToDigitalSource.vi** to the block diagram. Do this again for a second vi.
- Create a **WPI Robotics Library>>Sensor>>Encoder>> Encoder open.vi**
- Wire the **DIO Open vi DigitalInputDevRef** to the **Digital Source vi**
- Wire the **Digital Source vi** to the **Encoder Open.vi ASource** terminal
- Do the same for the **BSource** of the **Encoder Open.vi**
- Create a constant for the **DistancePerCount** terminal and set it to the number of inches per encoder tick.
- Create a **WPI Robotics Library>>Sensor>>Encoder>>ConfigureTimer.vi** and place it to the right of the **Encoder Open.vi**
- Right click on the **Minimum Rate** terminal and select **Create>>Constant** and set the rate at 20. This will have to be adjusted as testing goes.
- Right click on the **Number of samples to Average** terminal and create a constant of 1. This can be changed for consistent tick counts.
- Connect the **EncoderDevRef** and **Error** wires from the **Encoder Open.vi** to the **Config Timer.vi**
- Create a **WPI Robotics Library>>Sensor>>Encoder>>Start.vi** and wire the **EncoderDevRef** and **Error** wires.
- Create a **WPI Robotics Library>>Sensor>>Encoder>>RefNumRegistry.vi** and wire the **EncoderDevRef** and **Error** wires.
- Right click on the **refnum name** terminal and **Create>>Constant** and name the encoder reference to “a wheel”
- select the **Build Array** block and pull down on the bottom until one more terminal opens up.
- Wire the **Error** terminal to the **Build Array** block.
- Below is what the encoder portion of the Begin.vi should look like when complete

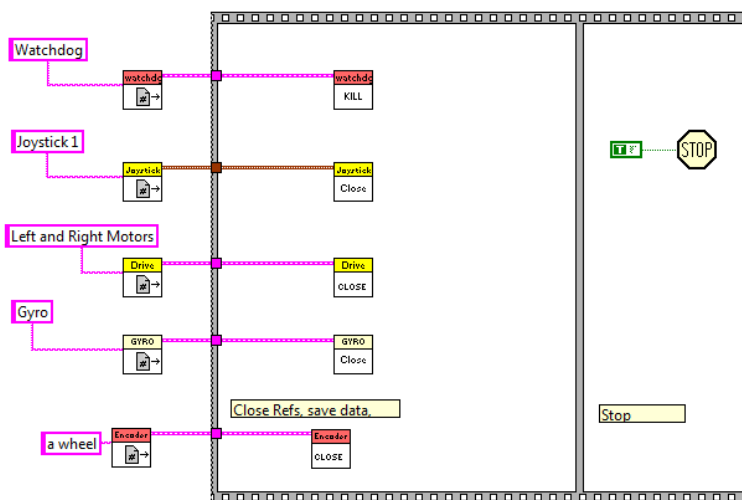


- 5) Once the code has been entered in the Begin.vi it's best to enter the code in the Finish.vi to close the encoder, before you forget.
- Open the Robot Main.vi block diagram.
 - Double click the Finish.vi on the right hand side of the Begin.vi.

- Enter <ctrl> e to open the Finish.vi block diagram.

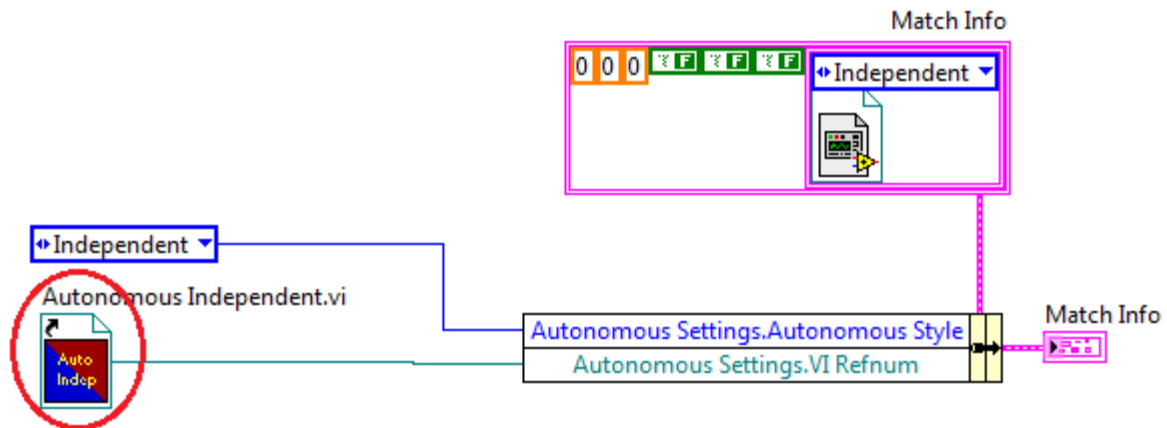


- Right click in a blank space near the bottom of the block diagram. Select **WPI Robotics Library>>Sensors>>Encoder>>Refnum Registry Get.vi** and drop it.
- On the **RefnumRegistry.vi**, right click on the **refnum name** and select **Create>>Constant**. Enter a **wheel** in the constant. This matches the reference name in the Begin.vi.
- In the flat sequence (the box in the diagram) right click and select **WPI Robotics Library>>Sensors>>Encoder>>Close.vi** and drop it in the flat sequence.
- Wire the **EncoderDevRef** from the **Refnum Registry Get.vi** to the encoder **Close.vi**.
- The Finish.vi block diagram should look like the diagram below

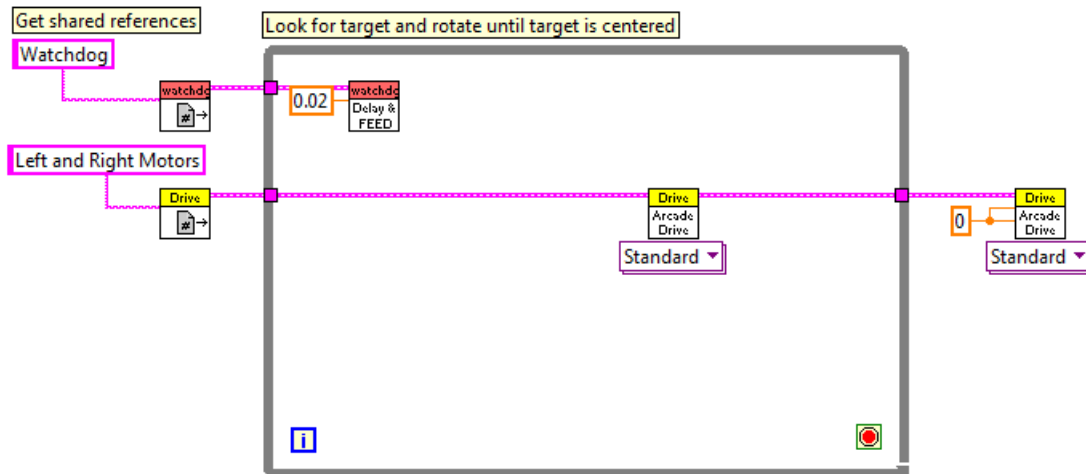


5) Next add code the Autonomous mode.

- Get back to the Begin.vi. At the top of the block diagram double click on the Autonomous Independent.vi reference. It's circled in red in the picture below. This vi is run when the autonomous mode is run.

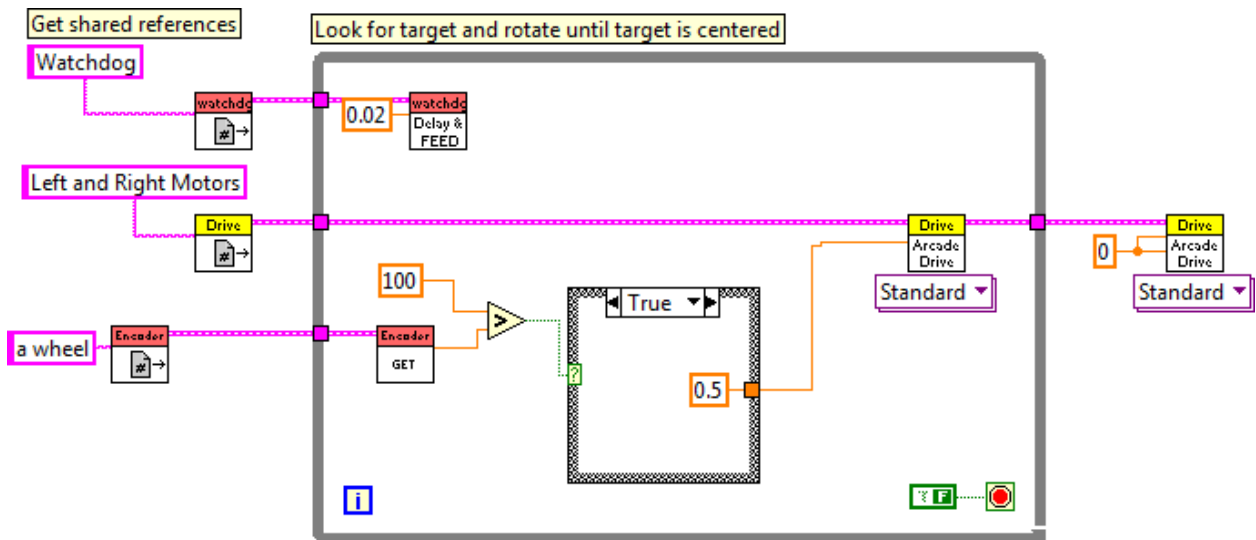


- For this example you can delete everything except for the watch dog timer Vi's and the Drive Vi's. Make the Vi look like the picture below.



- Right click on while terminator (the small stop sign in the lower right corner) and **Create>>Constant**. Since the autonomous is stopped by the Field Management Services all we need on this is a false constant.
- Get an open encoder reference **WPI Robotics Library>>Sensors>>Encoder>>Registry Refnum Get.vi** and drop it outside the while loop.
- Right click on the **refnum Name** terminal and **Create>>Constant**. Change the name to a wheel (to match its name in the **Begin.vi**).

- Get a Encoder Get.vi. **WPI Robotics Library>>Sensors>>Encoder>>Registry Refnum Get.vi** and drop it inside the while loop.
- Connect a wire the **EncoderDevRef** on the **Registry Refnum Get.vi** to the **Encoder Get.vi**.
- Drop a greater than comparison in the while loop. Right Click and select **Comparison>>Greater?** Vi inside the while loop.
- Wire from the **Encoder Get.vi** Distance terminal to the bottom greater symbol.
- Right click on the top terminal and **Create>>Constant** and change the constant to 100.
- Next create a case structure by right clicking and select **Structures>>Case Structure** and drop it inside the while loop.
- Wire from the point of the Greater symbol to the question mark on the case structure.
- Right click the down arrow at the top of the case structure and select the **True** case.
- Create a numeric constant by right clicking **Numeric>>Constant**. It will be an integer constant (blue color box)
- Change the constant to a double (orange color box) by right clicking the constant **Representation>>Double**. And change the number to 0.5 for half speed.
- Wire from the constant, through the case structure wall to the Y axis of the **Drive Arcade Drive.vi**
- Select the **False** case of the case structure.
- Right click the Orange block on the case structure wall and **Create>>Constant**. Leave the constant at 0.
- The autonomous Independent.vi should look like the picture below.



- 7) Save the Encoder Test code Code
 - Go to the Project Explorer and select **File>>Save**
 - Save the VI
- 8) Start the driver station code
 - Place the robot on blocks so it doesn't move, it's safer that way.
- 9) Run the autonomous code
 - Click the run button to download the code to the cRIO.
 - Once the code downloads,
 - On the dashboard, select the Autonomous switch and select Enable
 - Press disable on the dashboard to stop the program from running

Conclusion:

Congratulations, now you're up and running with a simple autonomous example. Explore the rest of the *Getting Started Programming with FRC Hardware* tutorials to learn about acquiring data from sensors, taking and processing images from the camera, and integrating the data you read from both the sensors and the camera into a project to move motors. Have fun programming and use the skills and techniques you learn in these tutorials to build a better robot!

Note that this has been tested but it still may not be perfect. Have fun and happy learning