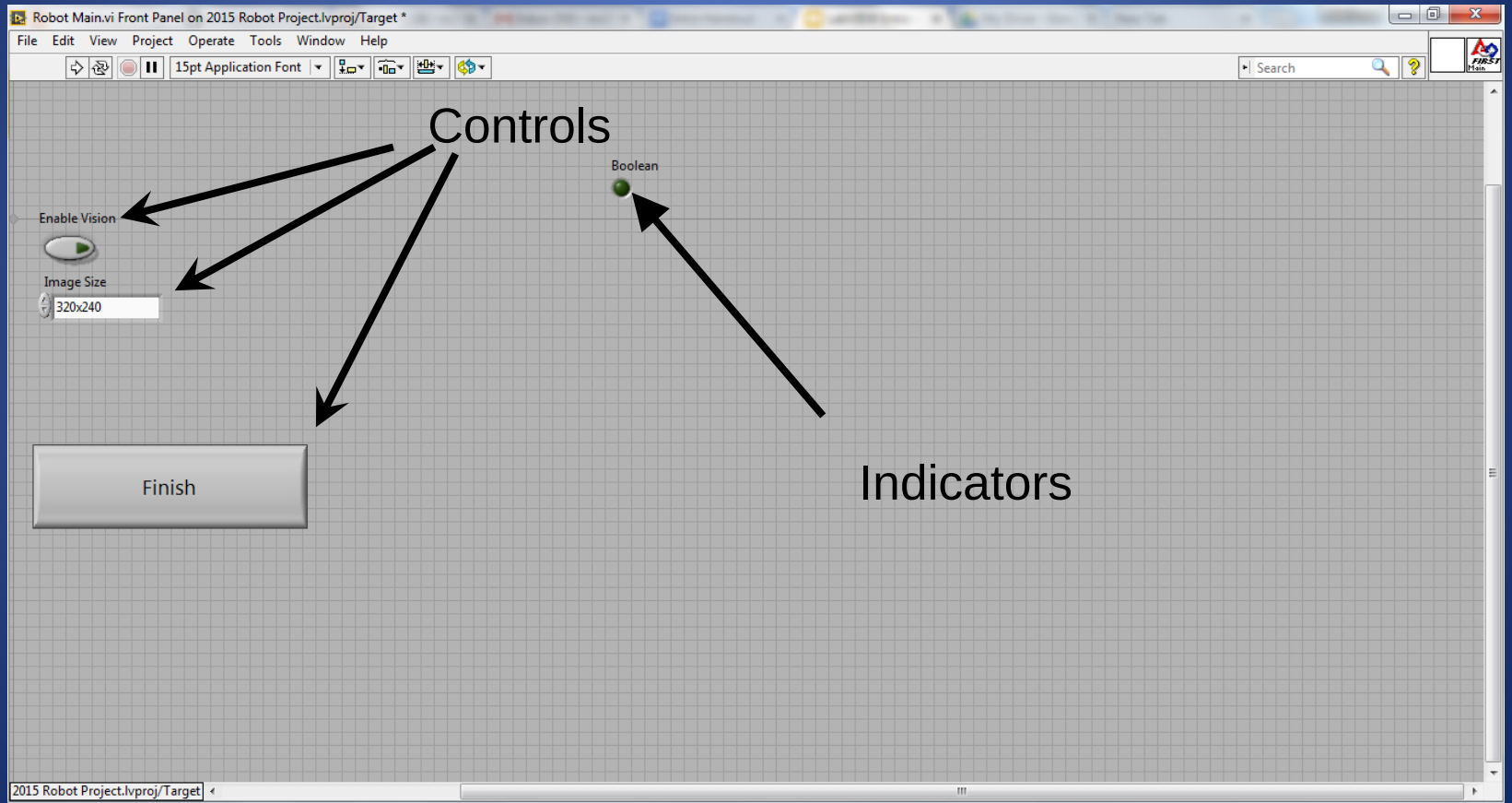
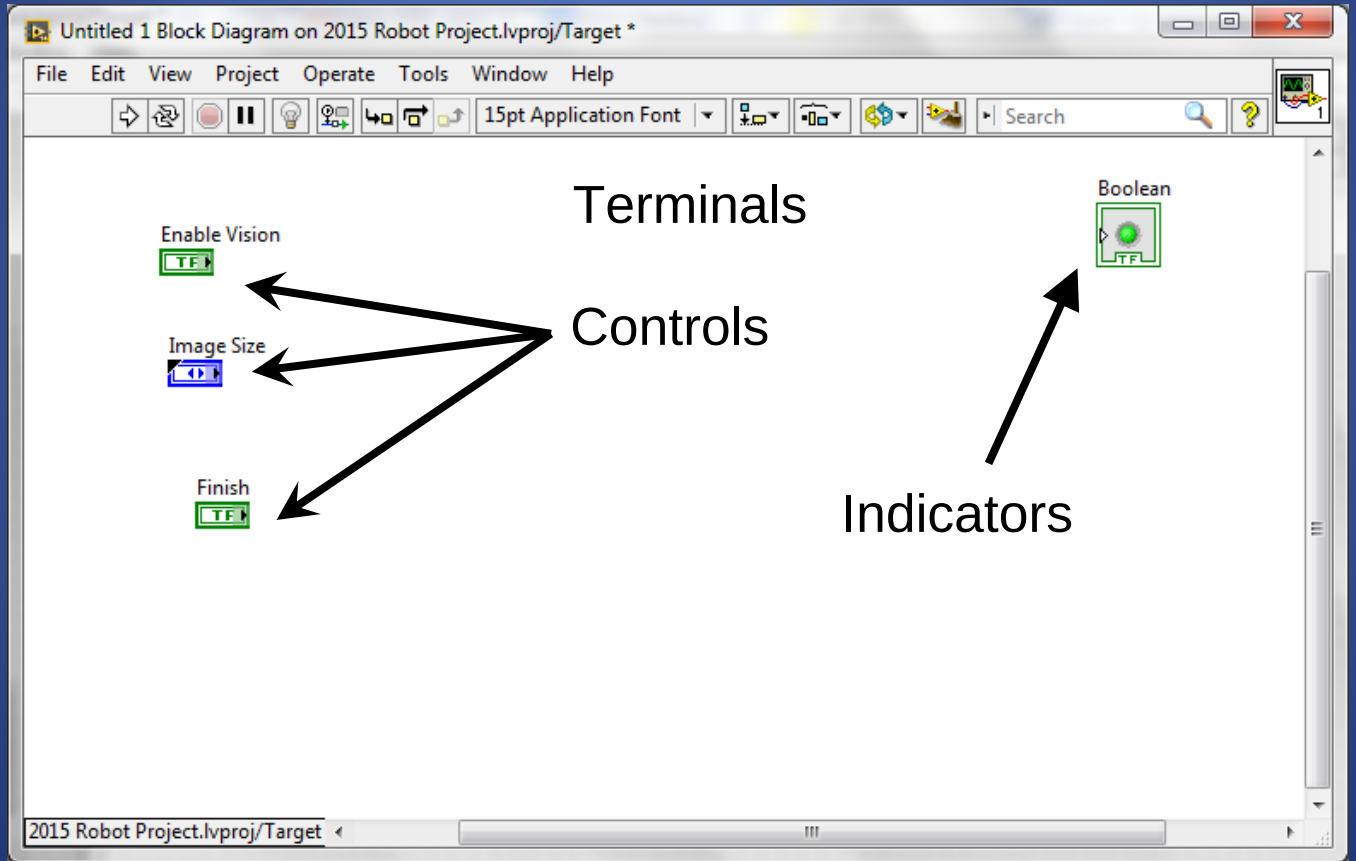
Intro to LabVIEW

http://workshop.frclabviewtutorials.com

Front Panel


Robot Main.vi Front Panel on 2015 Robot Project.lvproj/Target *
File Edit View Project Operate Tools Window Help
15pt Application Font
Search
Controls
Boolean
Enable Vision
Image Size
320x240
Finish
Indicators
2015 Robot Project.lvproj/Target

Block Diagram



Demo

Adding controls and indicators

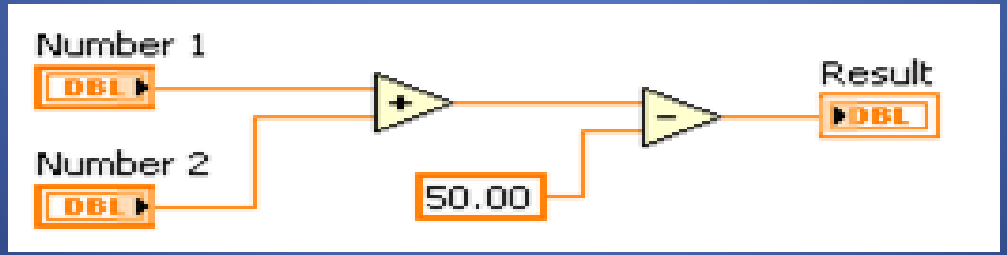
Demo

Adding controls and indicators

Data Flow

LabVIEW follows a dataflow model for running Vis

- A node executes only when data is available at all of its required input terminals.
- A node supplies data to the output terminals only when the node finishes execution.



Demo - Setting a motor

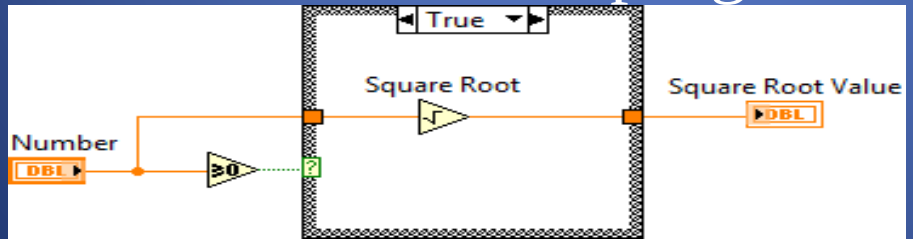
- Read Joystick
- Set Drive motors

Demo - Setting a motor

- Read Joystick
- Set Drive motors

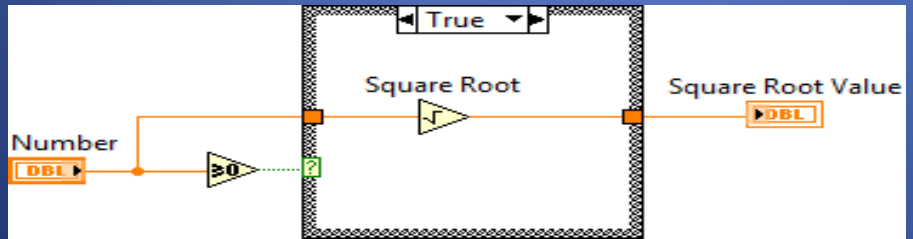
Case Structures

- Have two or more sub diagrams or cases.
- Use an input value to determine which case to execute.
- Execute and display only one case at a time.
- Are similar to **case** statements or **if...then...else** statements in text-based programming languages.



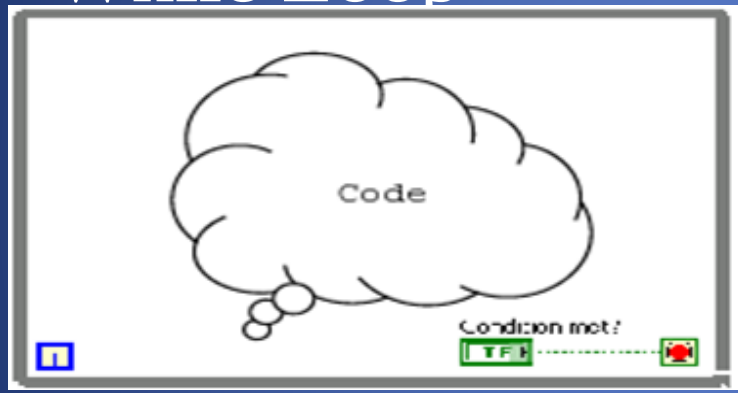
Case Structures

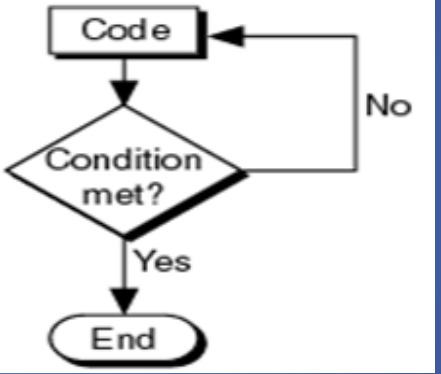
- Input and Output Tunnels
  - You can create multiple input and output tunnels.
  - Input tunnels are available to all cases if needed.
  - You must define each output tunnel for each case.*



Repetition

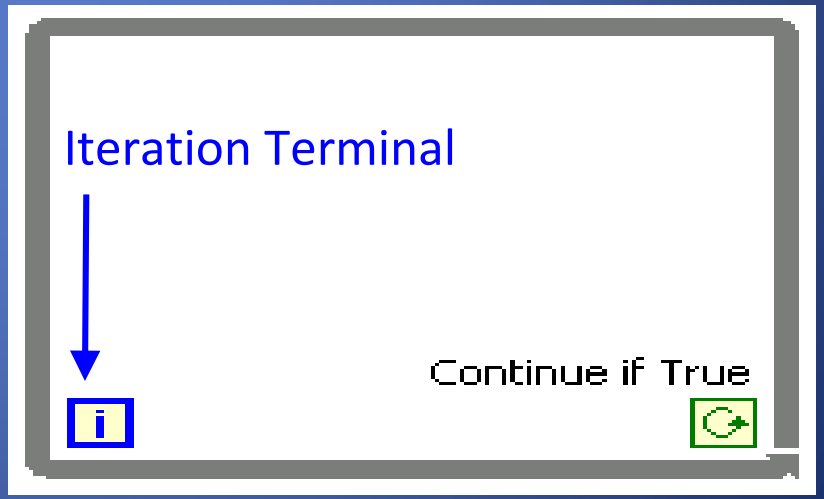
- While Loop





Repetition

- While Loop
  - Iteration terminal
    - Returns number of times loop has executed.
    - Is zero-indexed.

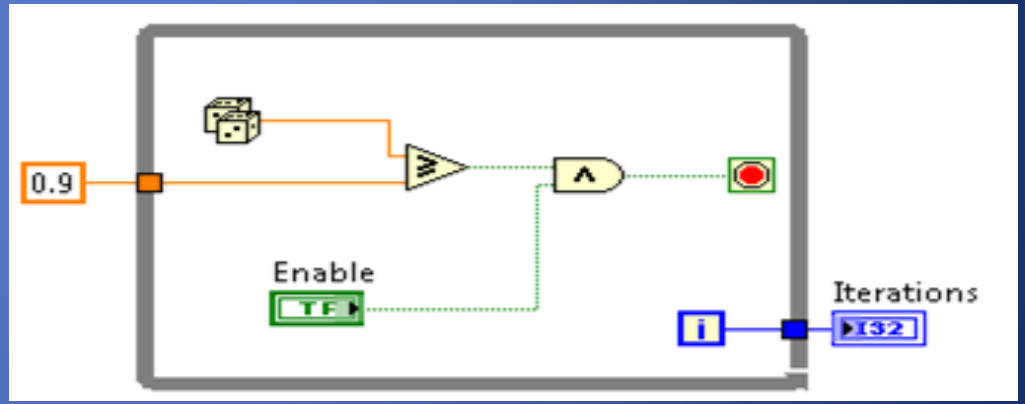


Repetition

- While Loop
  - Conditional terminal
    - Defines when the loop stops.
    - Has two options.
      - Stop if True
      - Continue if True

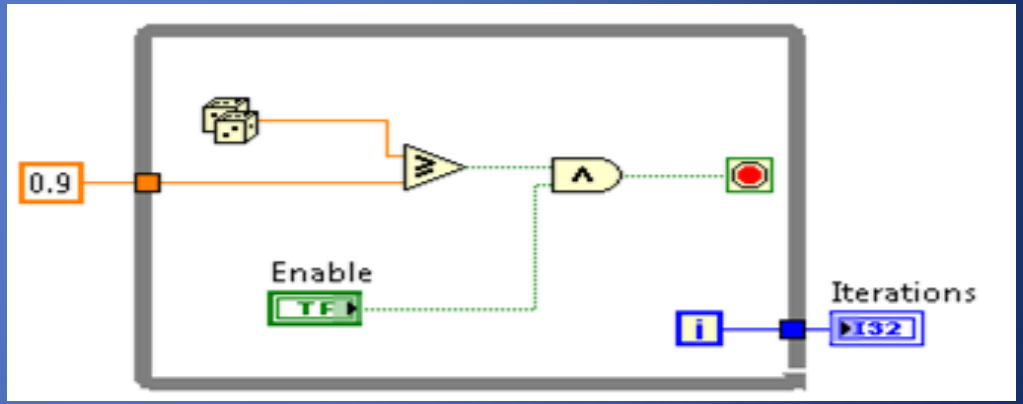
Repetition

- While Loop
  - Tunnels transfer data into and out of structures.



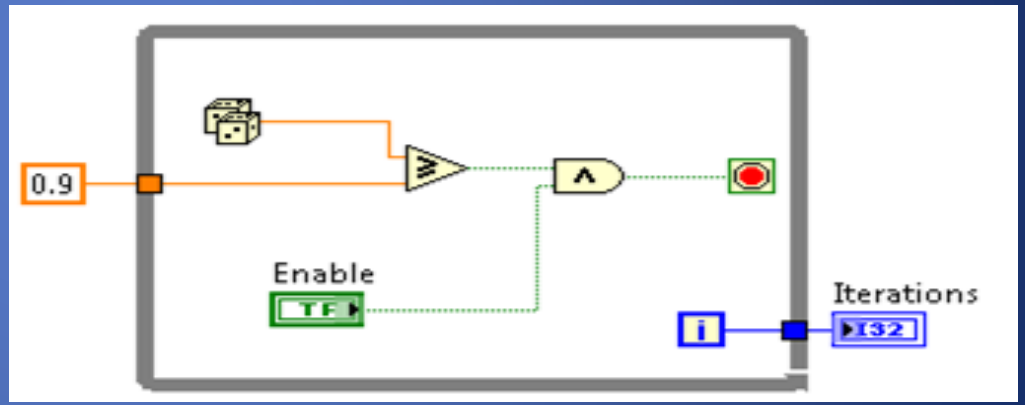
Repetition

- While Loop
  - Tunnels transfer data into and out of structures.
  - Data pass out of a loop after the loop terminates.



Repetition

- While Loop
  - Tunnels transfer data into and out of structures.
  - Data pass out of a loop after the loop terminates.
  - When a tunnel passes data into a loop, the loop executes only after data arrives at the tunnel.

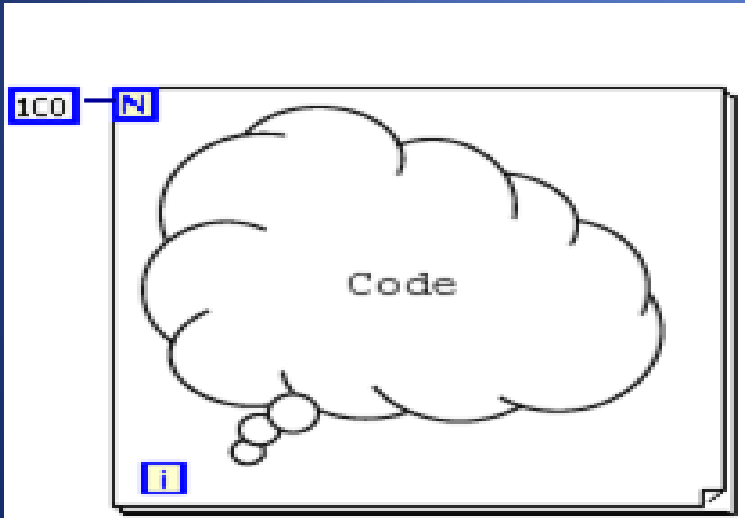


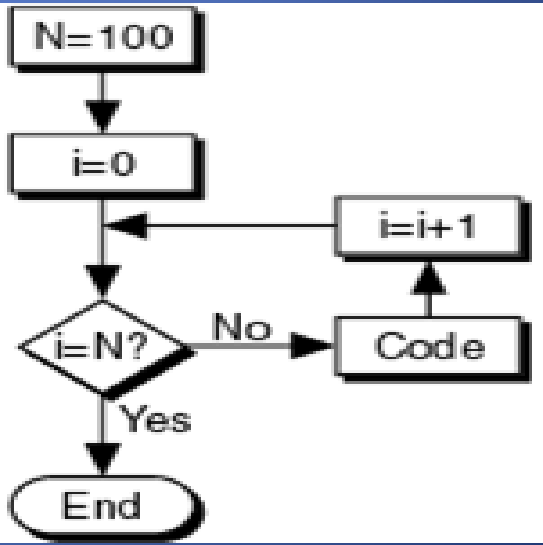
Repetition

- While Loop - Demo

Repetition

- While Loop
- For Loop





Repetition

- While Loop
- For Loop
  - 
Count Terminal

FRC Arhitecture

- Begin

FRC Arhitecture

- Begin

FRC Arhitecture

- Begin
  - Create references for all joysticks, motors, and sensors
  - Runs at power up

FRC Arhitecture

- Begin
- Teleop

FRC Arhitecture

- Begin
- Teleop

FRC Arhitecture

- Begin
- Teleop
  - Primarily used to read joysticks and set drive motors and actuators
  - Only runs while Teleop enabled

FRC Arhitecture

- Begin
- Teleop
- Autonomous

FRC Arhitecture

- Begin
- Teleop
- Autonomous

FRC Arhitecture

- Begin
- Teleop
- Autonomous
  - Runs when Autonomous is enabled

FRC Arhitecture

- Begin
- Teleop
- Autonomous
- Timed Tasks

FRC Arhitecture

- Begin
- Teleop
- Autonomous
- Timed Tasks

FRC Arhitecture

- Begin
- Teleop
- Autonomous
- Timed Tasks
  - Runs once enabled (during both auto and teleop)

FRC Deploying Code

- Run From Main

FRC Deploying Code

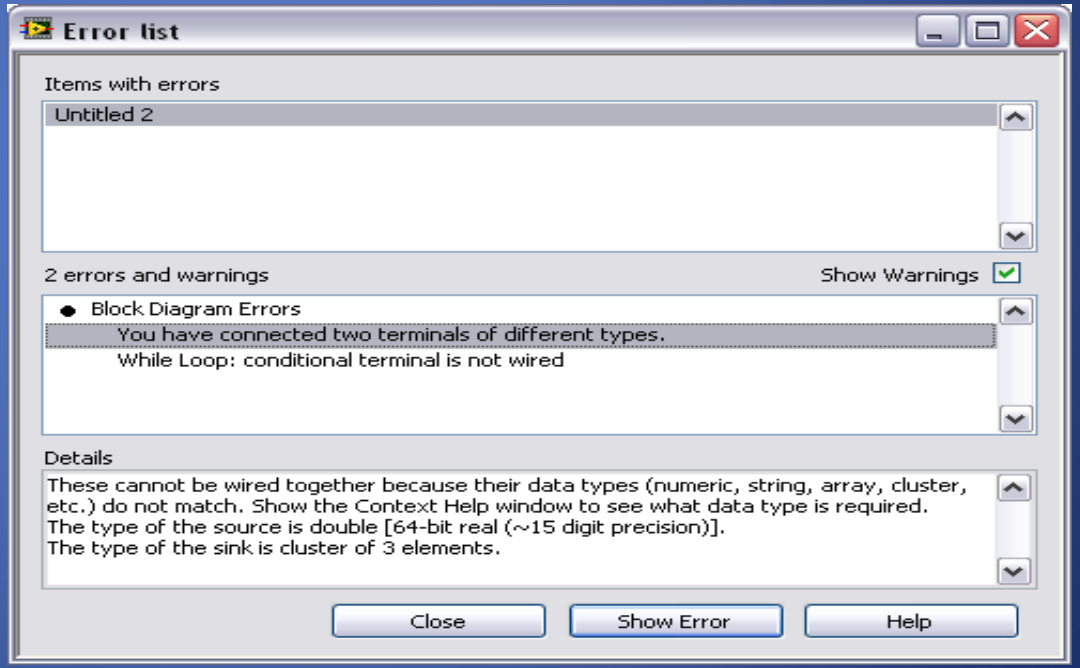
- Run From Main
- Deploy

FRC Deploying Code

- Run From Main
- Deploy
- Run as Startup

Debugging Techniques

- Correcting Broken VI's



Debugging Techniques

- Correcting Broken VI's
  - Broken Wires Exist (e.g.)
    - You wired a Boolean control to a String indicator.
    - You wired a numeric control to a numeric control.

Debugging Techniques

- Correcting Broken VI's
  - Broken Wires Exist (e.g.)
    - You wired a Boolean control to a String indicator.
    - You wired a numeric control to a numeric control.
  - A required block diagram terminal is unwired.

Debugging Techniques

- Correcting Broken VI's
  - Broken Wires Exist (e.g.)
    - You wired a Boolean control to a String indicator.
    - You wired a numeric control to a numeric control.
  - A required block diagram terminal is unwired.
  - A subVI is broken

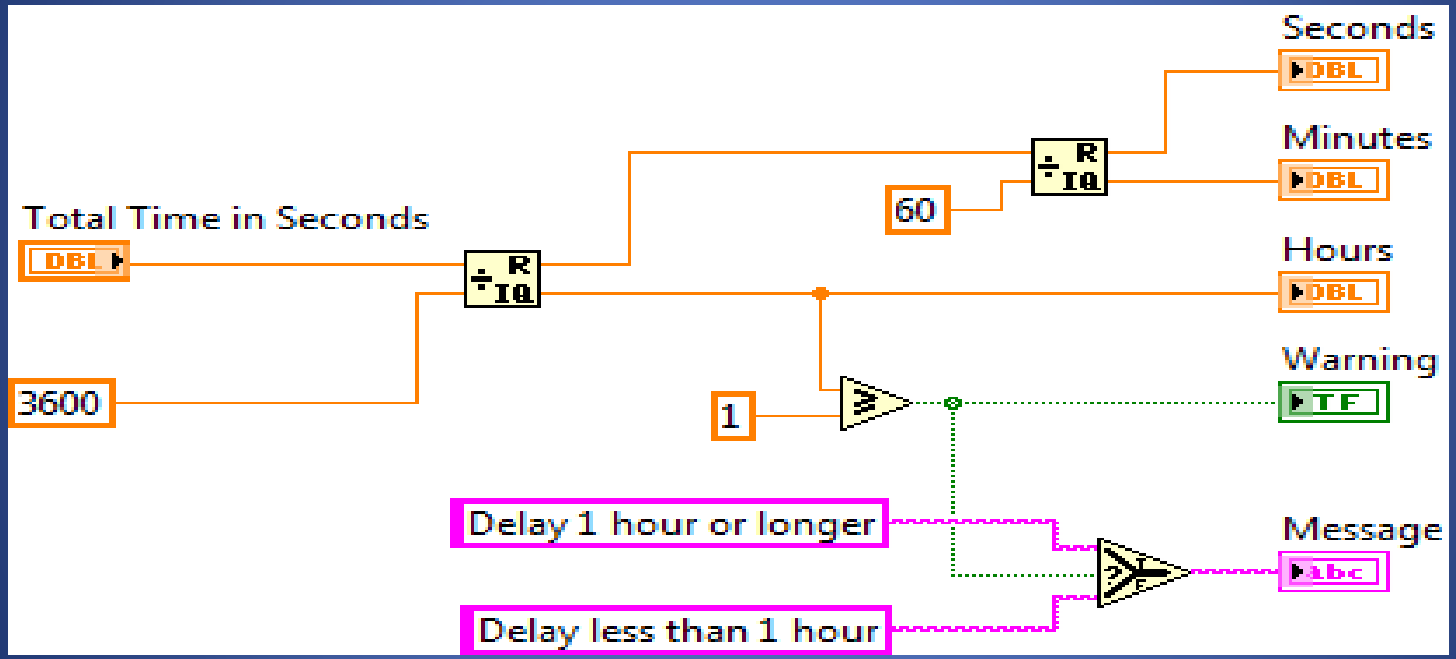
Debugging Techniques

- Correcting Broken VI’s
- Correcting Dataflow
  - Execution Highlighting
  - Single-Stepping & Breakpoints
  - Probes

Debugging Techniques

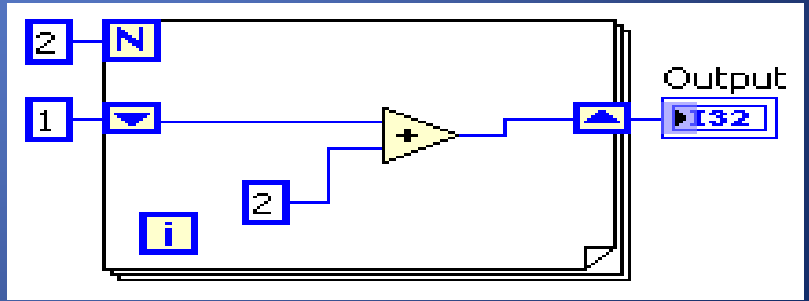
- Correcting Broken VI's
- Correcting Dataflow
  - Are there any unwired or hidden subVIs?
  - Is the default data correct?
  - Does the VI pass undefined data?
  - Are numeric representations correct?
  - Are nodes executed in the correct order?

Terminals and LabVIEW datatypes



Data Feedback in Loops

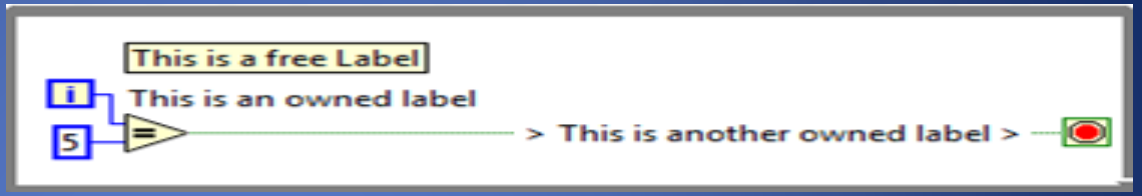
- Shift Registers
  - When programming with loops, you often need to know the values of data from previous iterations of the loop.
  - Shift registers transfer values from one loop iteration to the next.



Documentation

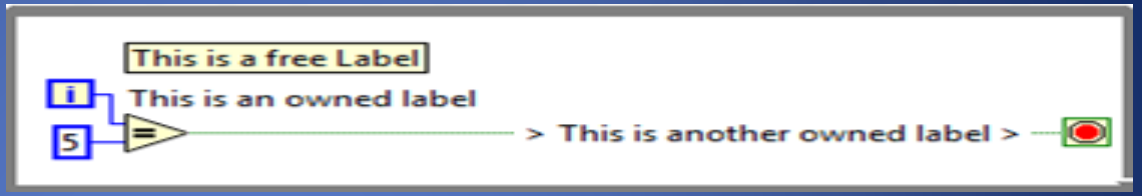
- Free Labels

Documentation



- Free Labels
  - Describe algorithms.
  - Have pale yellow backgrounds.
  - Double-click in any open space to create.

Documentation

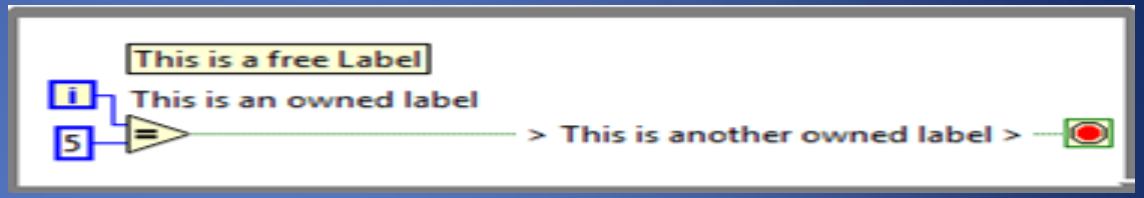


- Free Labels
  - Describe algorithms.
  - Have pale yellow backgrounds.
  - Double-click in any open space to create.

Documentation

- Free Labels
- Owned Labels
  - Explain data contents of wires and objects.
  - Move with object.
  - Have transparent backgrounds.
  - Select Visible Items»Label from the shortcut menu to create.

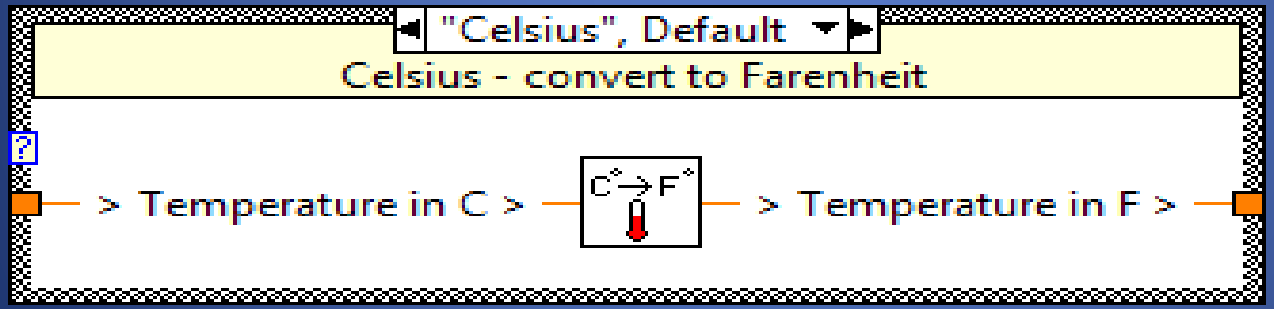
Documentation



- Free Labels
- Owned Labels
  - Explain data contents of wires and objects.
  - Move with object.
  - Have transparent backgrounds.
  - Select Visible Items»Label from the shortcut menu to create.

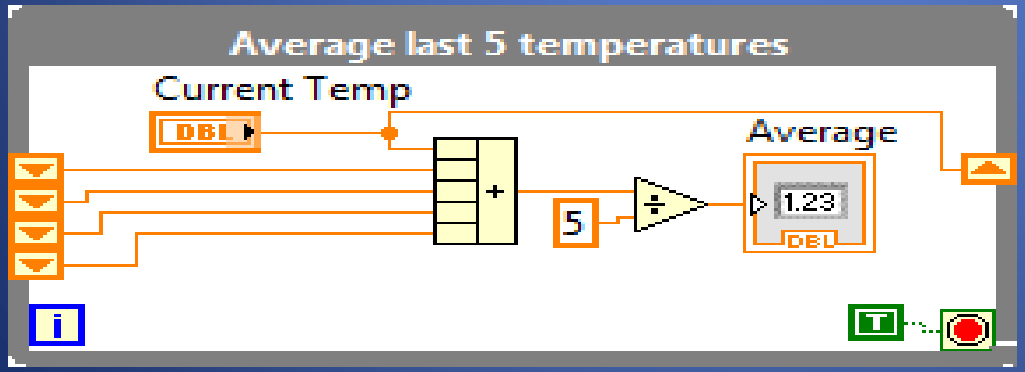
Documentation

- Free Labels
- Owned Labels
- Sub diagram Labels



Documentation

- Free Labels
- Owned Labels
- Sub diagram Labels
  - Case Structures



Documentation

- Free Labels
- Owned Labels
- Sub diagram Labels
- White Papers

Documentation

IR Based Line Following

This document describes:

1. Assumptions about robot construction
2. Information about mounting, wiring, and calibrating the IR sensors
3. How the control code operates
4. How to troubleshoot and tune the sample code to work after robots are modified and no longer meet the assumptions

1. Assumptions about Robot Construction

- Six-wheel drop-center skid-steer robot with gray wheels eight inches in diameter
- PWM channel 1 controls the left center wheel
- PWM channel 2 controls the right center wheel
- Left and right motors are both controlled by Jaguar motor controllers with the jumper set to brake mode
- IR sensors are rigidly mounted on the front-center of the robot relatively far from the center of rotation and about two inches above the carpet
- The active portion of the sensors face the carpet and are connected to digital input signals 1, 2, and 3 in slot four and are wired to appropriate power and ground signals

(Note that for general driving, you may want to switch the mode to coast. You can accomplish this using a digital output or you can retune the control code so that it works with the jumper set to coast.)

Keyboard Shortcuts

- CTRL + u = diagram cleanup
- Right Click = palette
- CTRL + Space = quick drop
- CTRL + e = switch window
- CTRL + Shift + e = activate project window
- CTRL + r = Run
- CTRL + t = split window