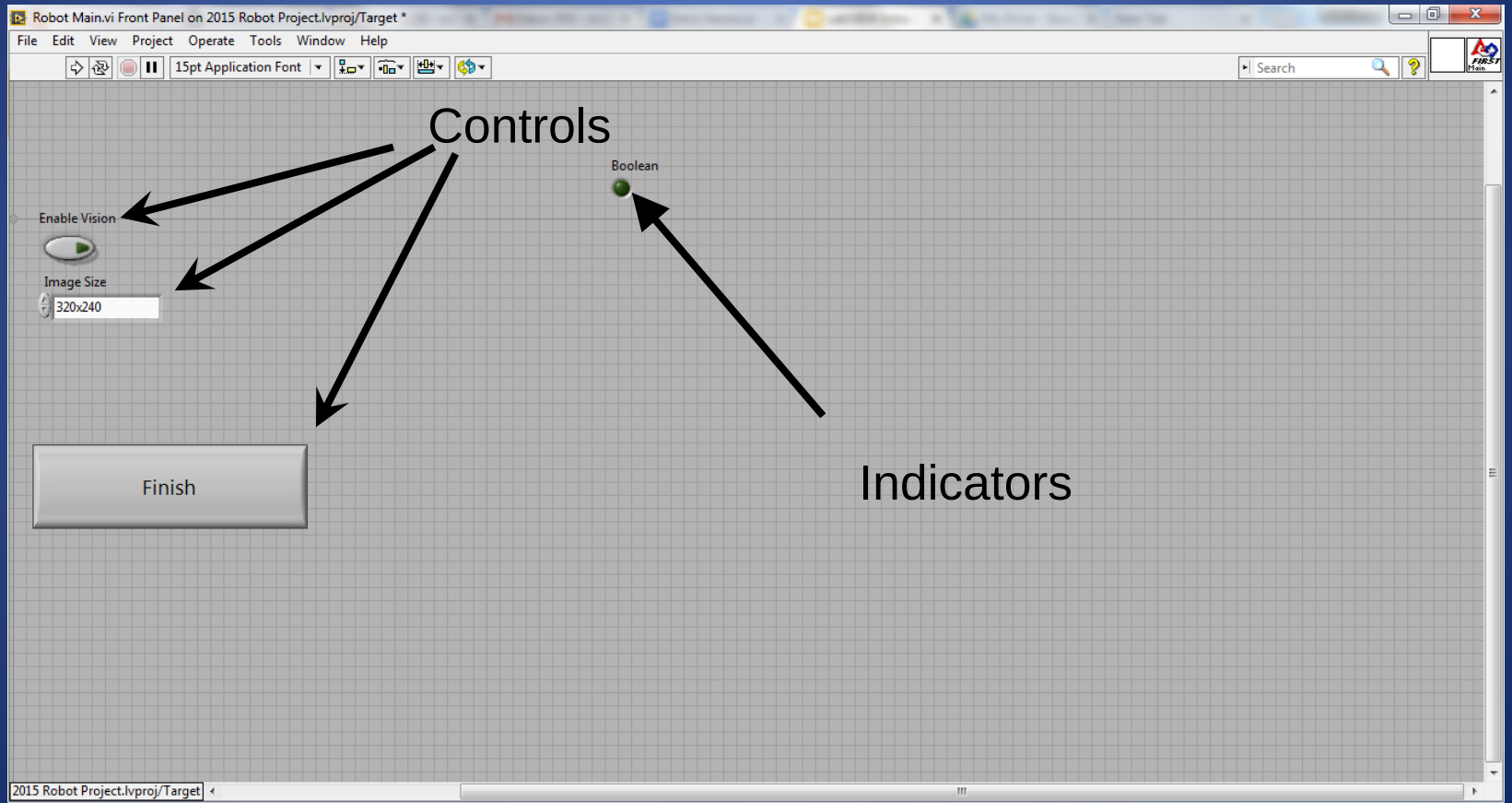
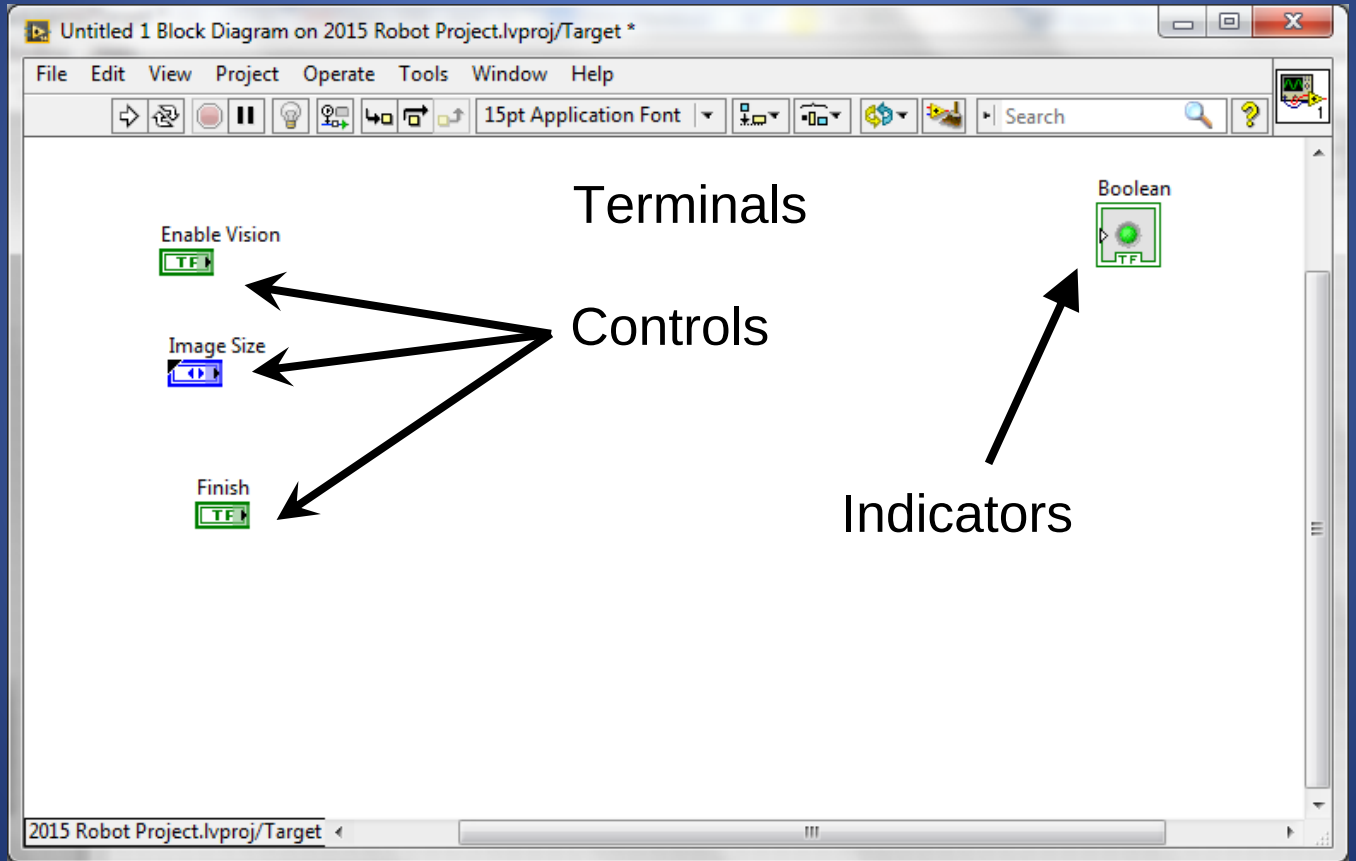
Intro to LabVIEW

frclabviewtutorials.com/workshop

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

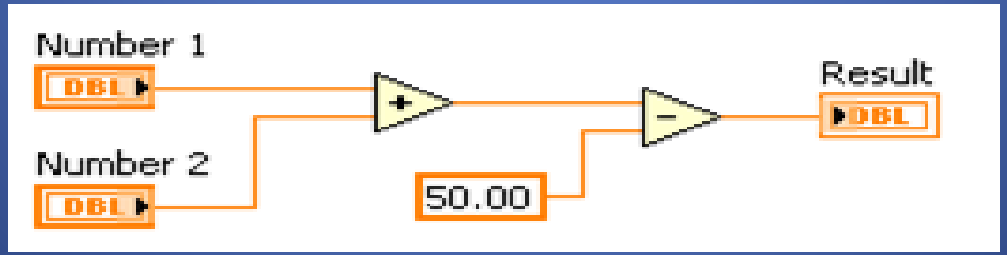
Demo

Adding controls and indicators

Data Flow

LabVIEW follows a dataflow model for running Vis

- A node executes only when data are 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

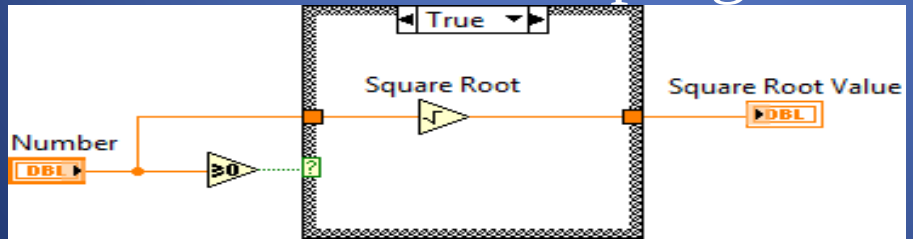
- Read Joystick
- Set Drive motors

Exercise

Drive a motor.pdf

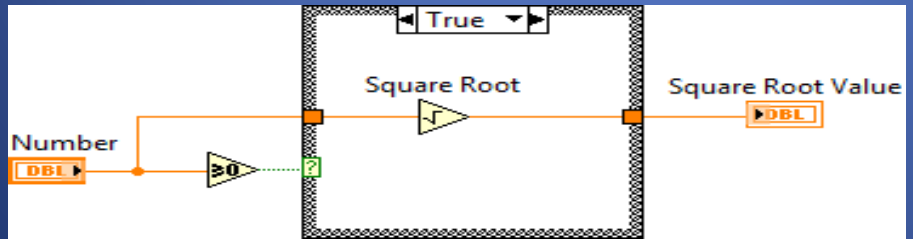
Case Structures

- 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.



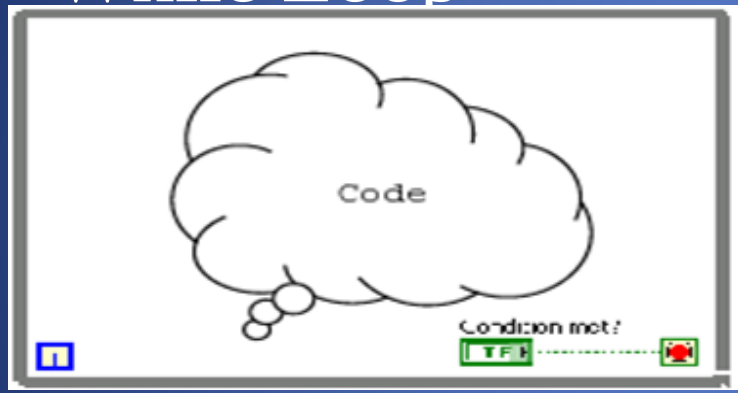
Case Structures

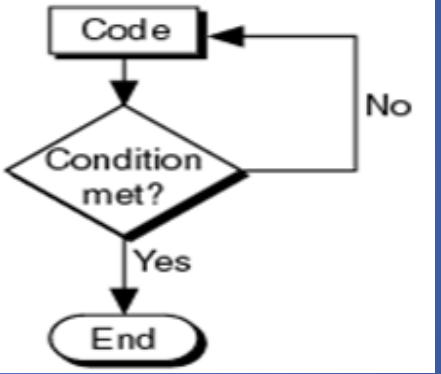
- Input and Output Tunnels
  - You can create multiple input and output tunnels.
  - Input tunnels are available to all cases if needed.
  - You must define each output tunnel for each case.



Repetition

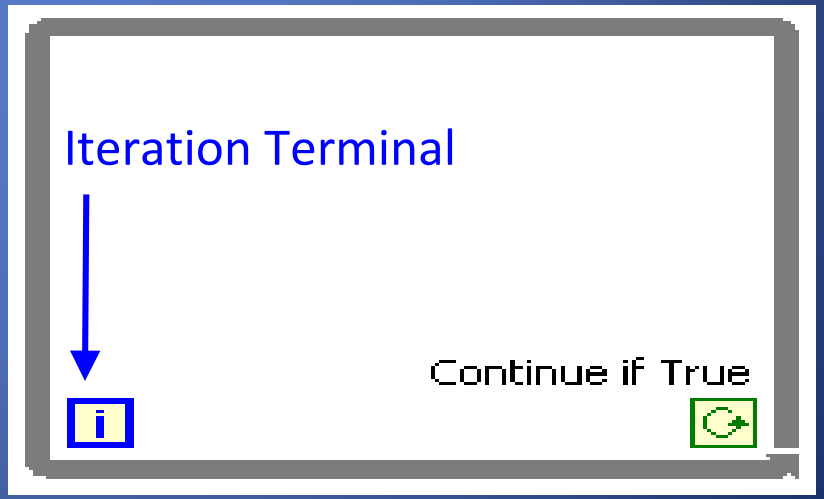
- While Loop





Repetition

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

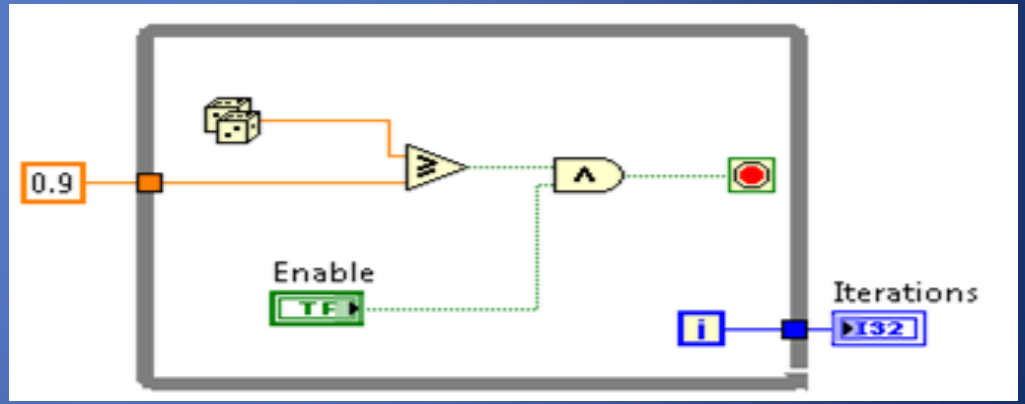


Repetition

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

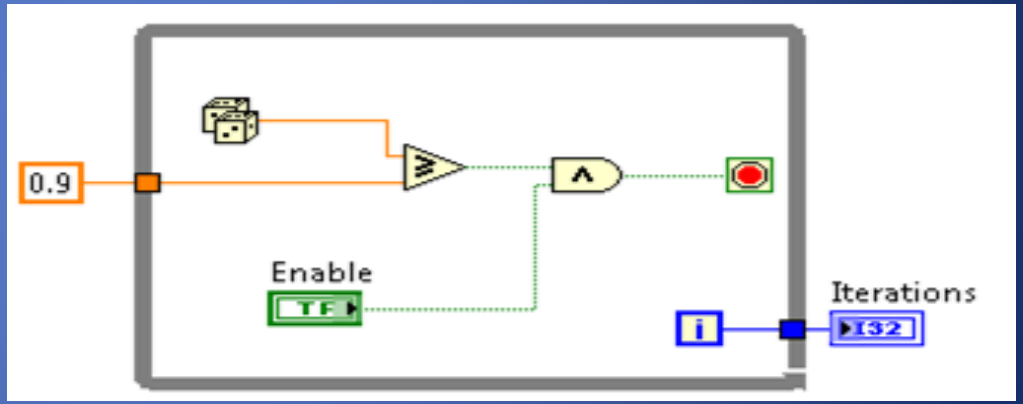
Repetition

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



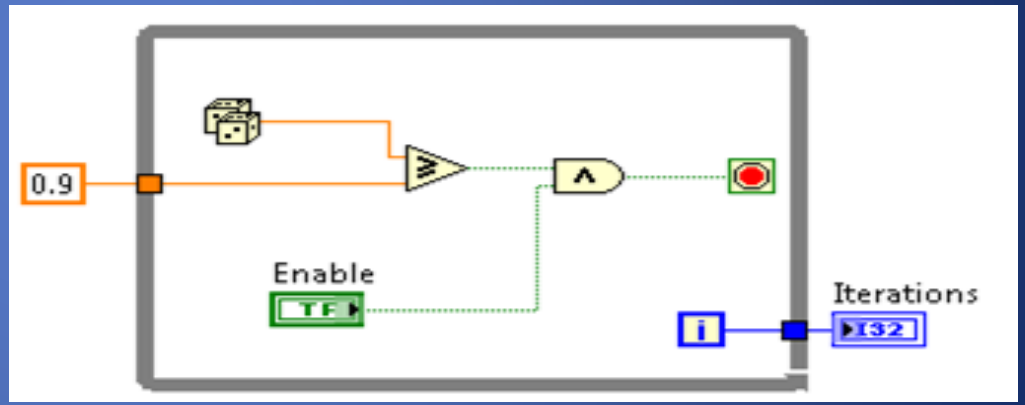
Repetition

- 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.



Exercise – While Loops

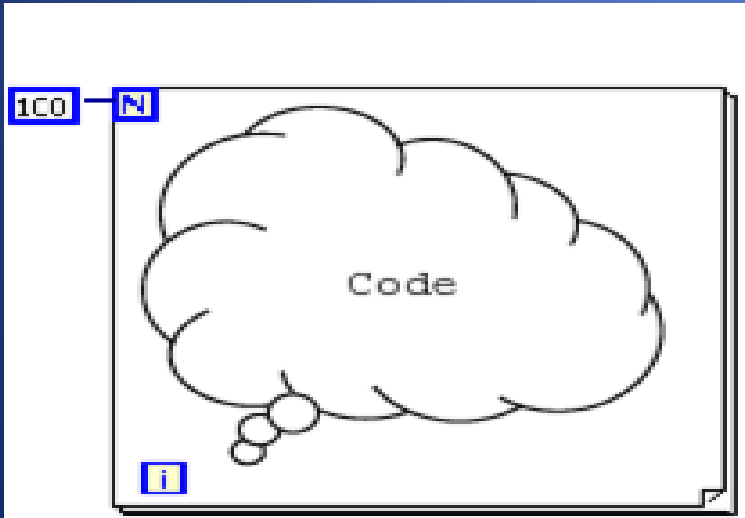
While Loops.pdf

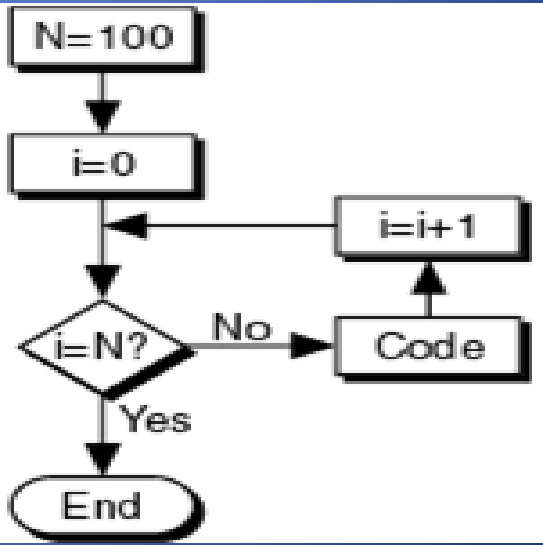
Exercise – While Loops

- How many times is the Number of Iterations indicator updated? Why?

Repetition

- While Loop
- For Loop





Repetition

- While Loop
- For Loop
  - 
Count Terminal

Repetition

- Comparison
  - Description
    - For the following scenarios, decide whether to use a While Loop or a For Loop.

Repetition

- Comparison
  - Description
    - For the following scenarios, decide whether to use a While Loop or a For Loop.
  - **Scenario 1**
    - Acquire pressure data in a loop that executes once per second for one minute.
    - 1. If you use a While Loop, what is the condition that you need to stop the loop?
    - 2. If you use a For Loop, how many iterations does the loop need to run?
    - 3. Is it easier to implement a For Loop or a While Loop?

Repetition

- Comparison
  - Description
    - For the following scenarios, decide whether to use a While Loop or a For Loop.
  - **Scenario 1**
    - Acquire pressure data in a loop that executes once per second for one minute.
    - 1. If you use a While Loop, what is the condition that you need to stop the loop?
    - 2. If you use a For Loop, how many iterations does the loop need to run?
    - 3. Is it easier to implement a For Loop or a While Loop?
  - **Scenario 2**
    - Acquire pressure data until the pressure is greater than or equal to 1400 psi.
    - 1. If you use a While Loop, what is the condition that you need to stop the loop?
    - 2. If you use a For Loop, how many iterations does the loop need to run?
    - 3. Is it easier to implement a For Loop or a While Loop?

Repetition

- Comparison
  - Scenario 3
    - Acquire pressure and temperature data until both values are stable for two minutes.
    - 1. If you use a While Loop, what is the condition that you need to stop the loop?
    - 2. If you use a For Loop, how many iterations does the loop need to run?
    - 3. Is it easier to implement a For Loop or a While Loop?

Repetition

- Comparison
  - Scenario 3
    - Acquire pressure and temperature data until both values are stable for two minutes.
    - 1. If you use a While Loop, what is the condition that you need to stop the loop?
    - 2. If you use a For Loop, how many iterations does the loop need to run?
    - 3. Is it easier to implement a For Loop or a While Loop?
  - Scenario 4
    - Output a voltage ramp starting at zero, increasing incrementally by 0.5 V every second, until the output voltage is equal to 5 V.
    - 1. If you use a While Loop, what is the condition that you need to stop the loop?
    - 2. If you use a For Loop, how many iterations does the loop need to run?
    - 3. Is it easier to implement a For Loop or a While Loop?

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

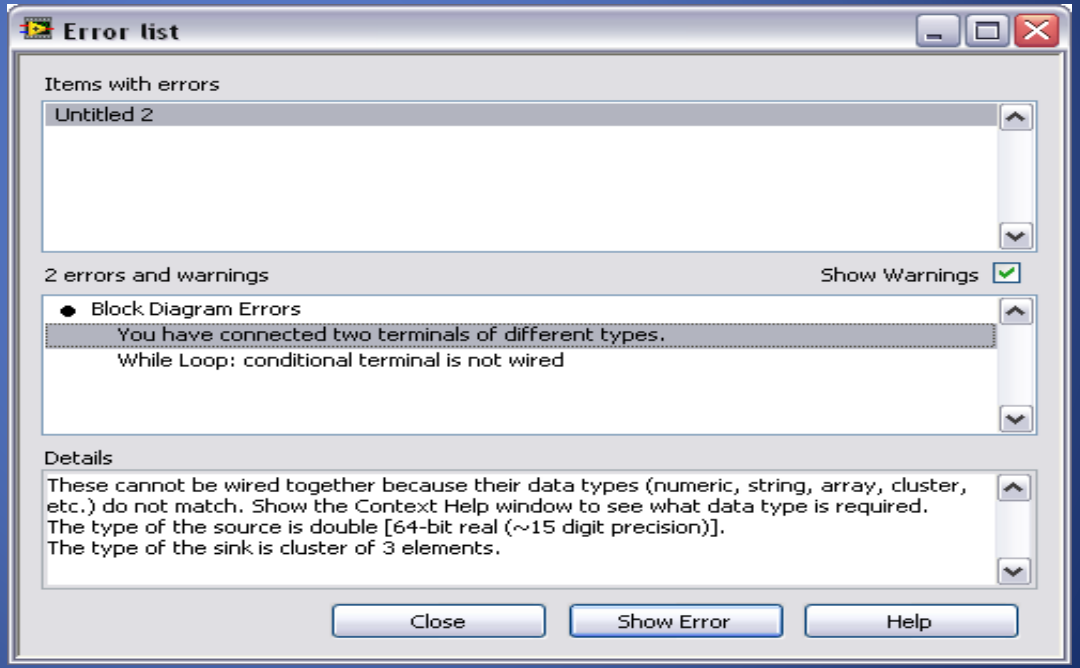
- Begin
- Teleop
- Autonomous
- Timed Tasks

FRC Arhitecture

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

Debugging Techniques

- Correcting Broken VI's



Debugging Techniques

- Correcting Broken VI's
  - Broken Wires Exist
    - 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
    - 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
    - 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

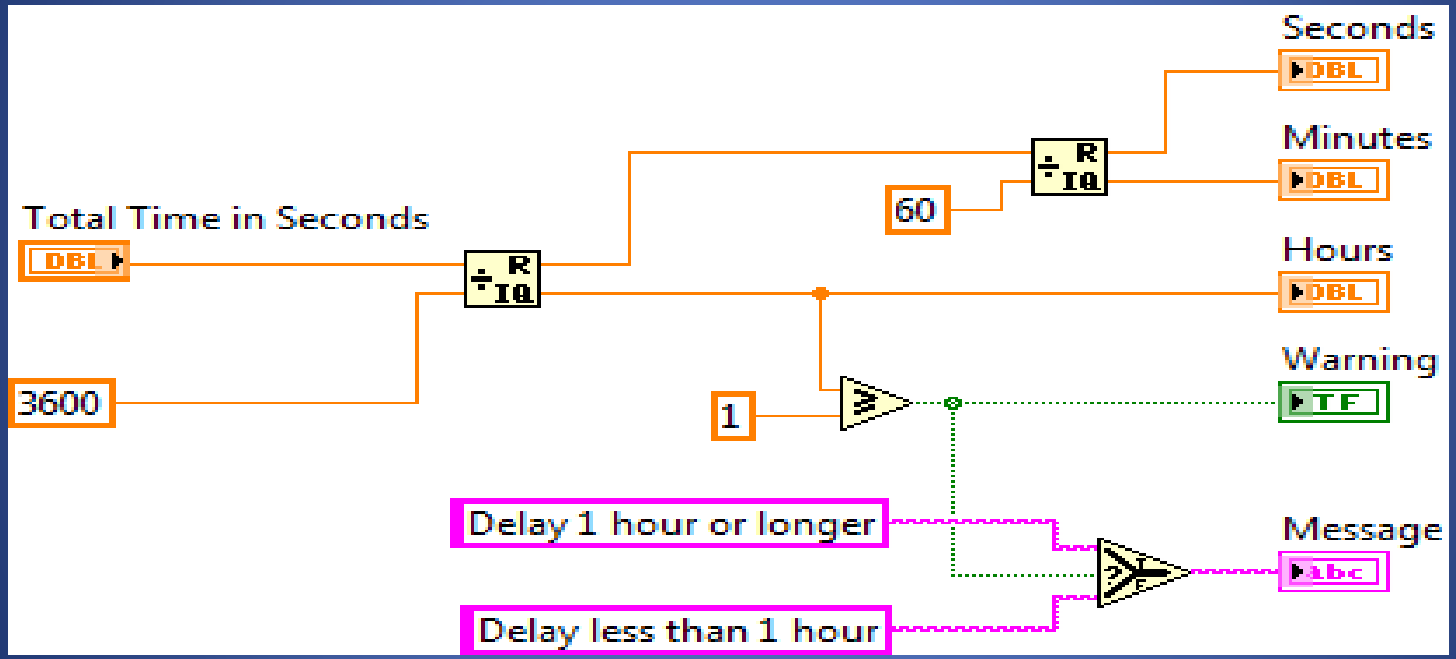
Debugging Techniques

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

Debugging Techniques

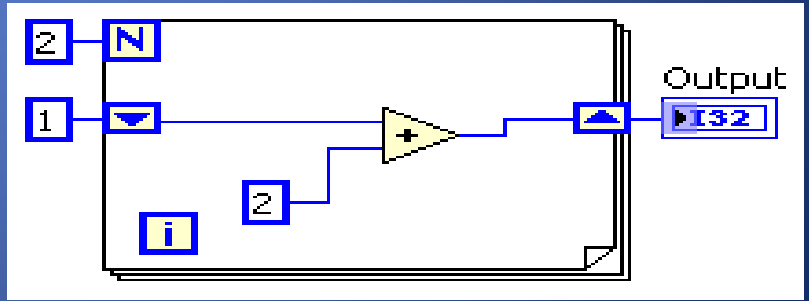
- Correcting Broken VI's
- Correcting Dataflow
  - Are there any unwired or hidden subVIs?
  - Is the default data correct?
  - Does the VI pass undefined data?
  - Are numeric representations correct?
  - Are nodes executed in the correct order?

Terminals and LabVIEW datatypes



Data Feedback in Loops

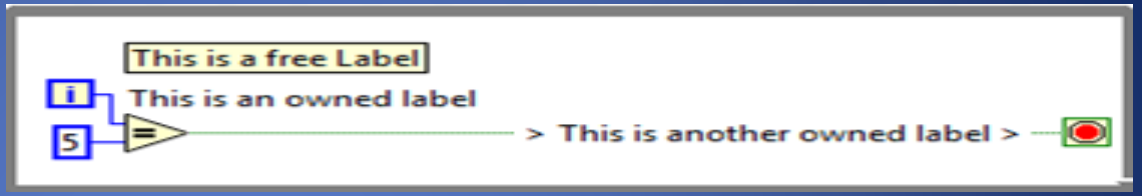
- Shift Registers
  - When programming with loops, you often need to know the values of data from previous iterations of the loop.
  - Shift registers transfer values from one loop iteration to the next.



Documentation

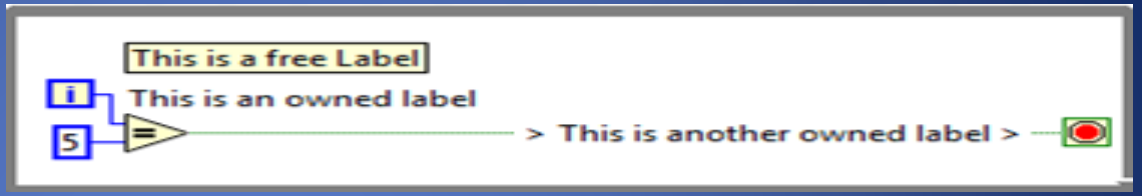
- Free Labels

Documentation



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

Documentation

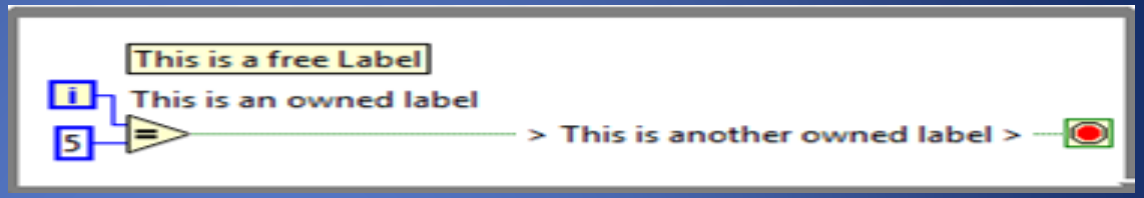


- Free Labels
  - Describe algorithms.
  - Have pale yellow backgrounds.
  - Double-click in any open space to create.

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.

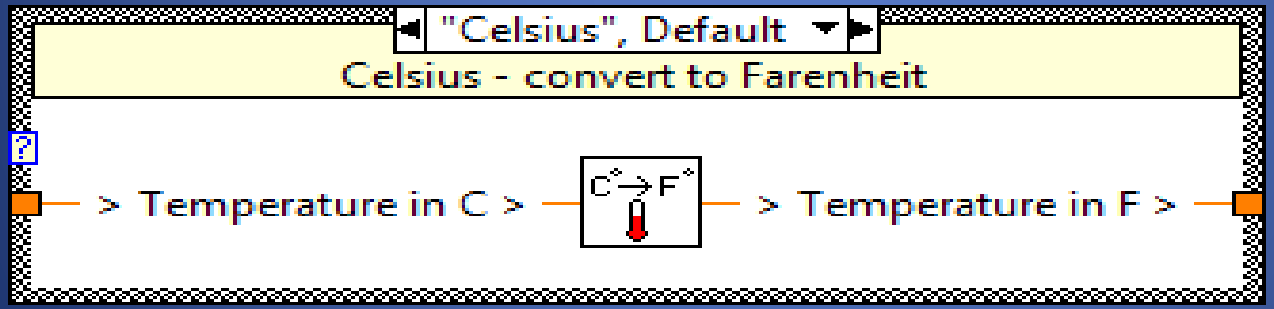
Documentation



- 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.

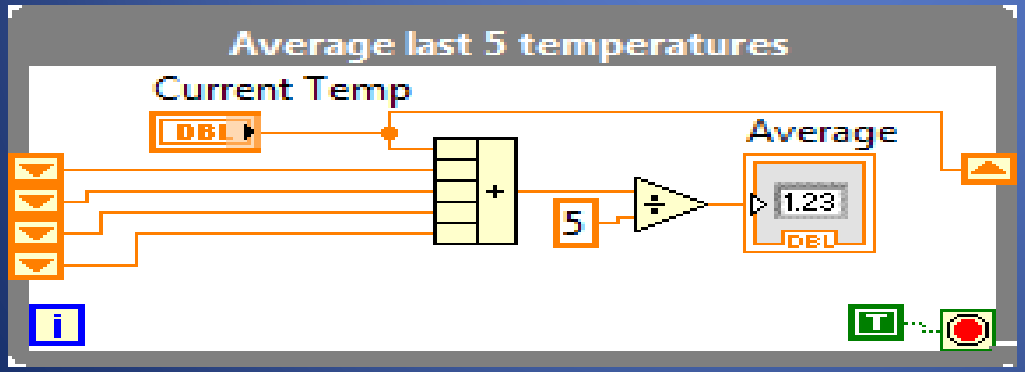
Documentation

- 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

Advanced LabVIEW

frclabviewtutorials.com/workshop

Let LabVIEW do the work!

TypeDefs

Demo

TypeDefs - Teleop Optimization

Demo

TypeDefs - Teleop Optimization

Let LabVIEW do the work!

TypeDefs
Functional Global Variable (FGV)

Variables

How would you handle the following dataflow challenges?

- Initialize front panel controls with values from a configuration file?
- Copy a “Ship To” address to a “Bill To” address?
- Initialize indicators that will be written to later in your code?
- Write to an indicator in two cases of a Case structure without writing to it in all cases?

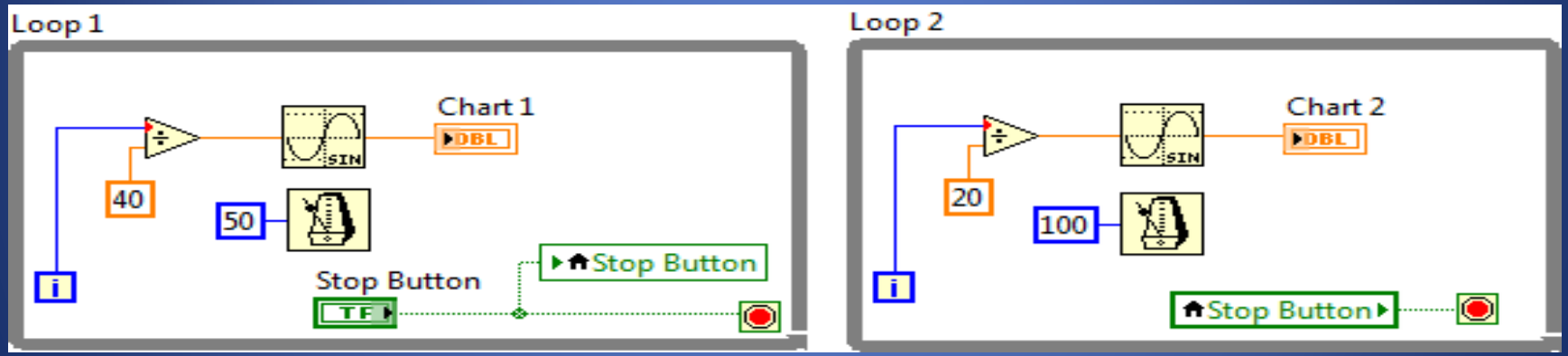
Variables

Variables can be of the following types:

- **Local**—Stores data in front panel controls and indicators.

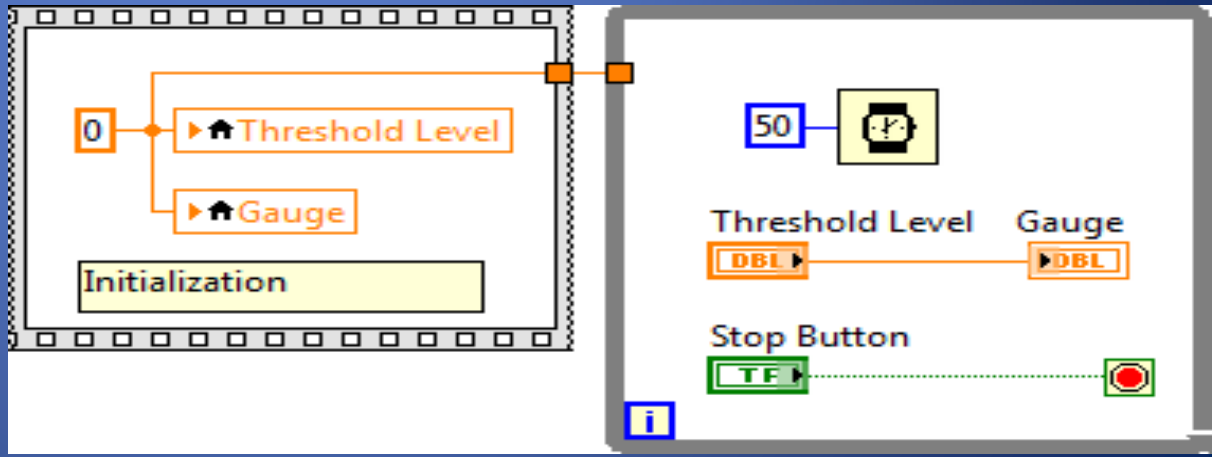
Local Variables

- Use local variables to pass data within a single VI.



Local Variables

- Use local variables to pass data within a single VI.
- Use local variables to modify front panel control values.



Local Variables - Demo

- Race Conditions

Local Variables Exercise

- Local Variable Exercise.pdf

Variables

Variables can be of the following types:

- **Local**—Stores data in front panel controls and indicators.
- **Global** —Stores data in special repositories that can be accessed from multiple VIs.

Global Variable

- Store data
- Can be accessed across the entire project

Variables

Variables can be of the following types:

- **Local**—Stores data in front panel controls and indicators.
- **Global** —Stores data in special repositories that can be accessed from multiple VIs.
- **Functional Global**—Stores data in While Loop shift registers.

Functional Global Variable

- Store Data
- Can be accessed across the entire project
- Can perform operations on the data
- Used to avoid read/write race conditions
- Used to implement custom boundaries on data

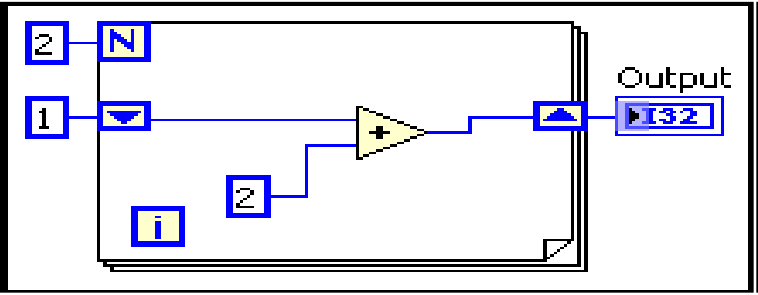
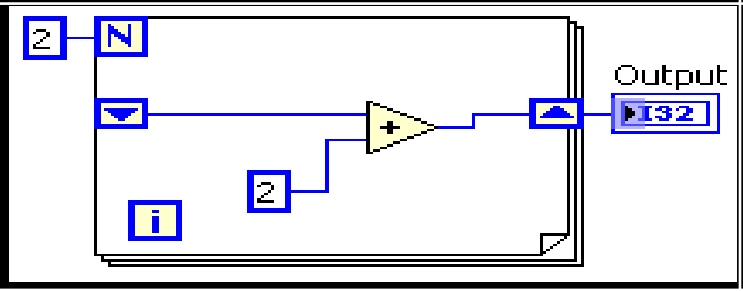
FGV

Implemented with a shift register

Shift Registers

- Right-click the border and select **Add Shift Register** from the shortcut menu.
- Right shift register stores data on completion of an iteration.
- Left shift register provides stored data at beginning of the next iteration.

Shift Registers

Block Diagram	1st run	2nd run
Initialized Shift Register 	Output = 5	Output = 5
Not Initialized Shift Register 	Output = 4	Output = 8

Shift Registers

- Default Values

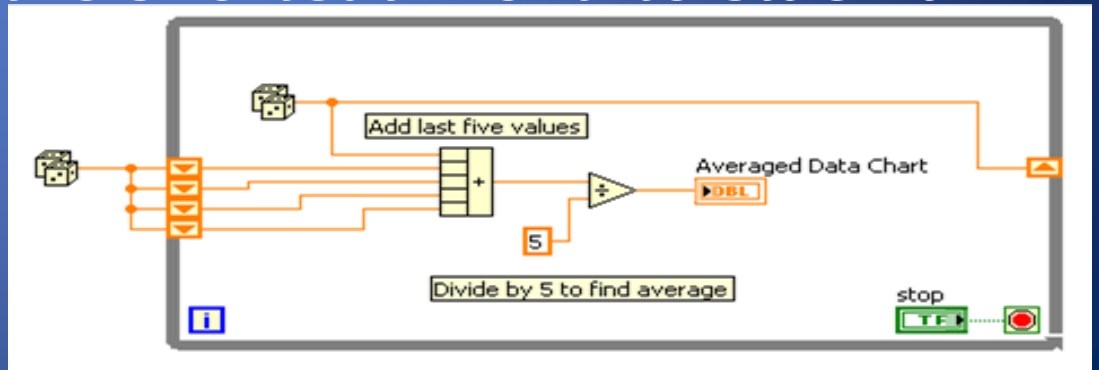
Data Type	Default Value
Numeric	0
Boolean	FALSE
String	Empty

Shift Registers

- Stacked shift registers remember values from multiple previous iterations and carry those values to the next iterations.

Shift Registers

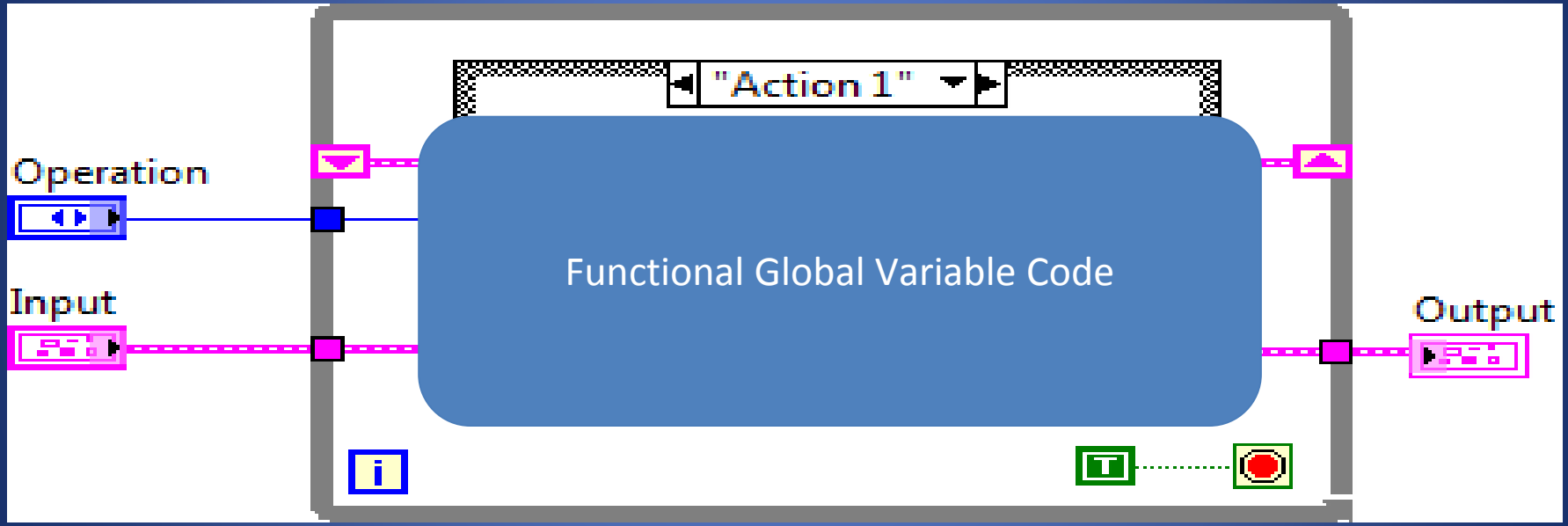
- Stacked shift registers remember values from multiple previous iterations and carry those values to the next iterations.
- Right-click the left shift register and select Add Element from the shortcut menu to stack a shift register.



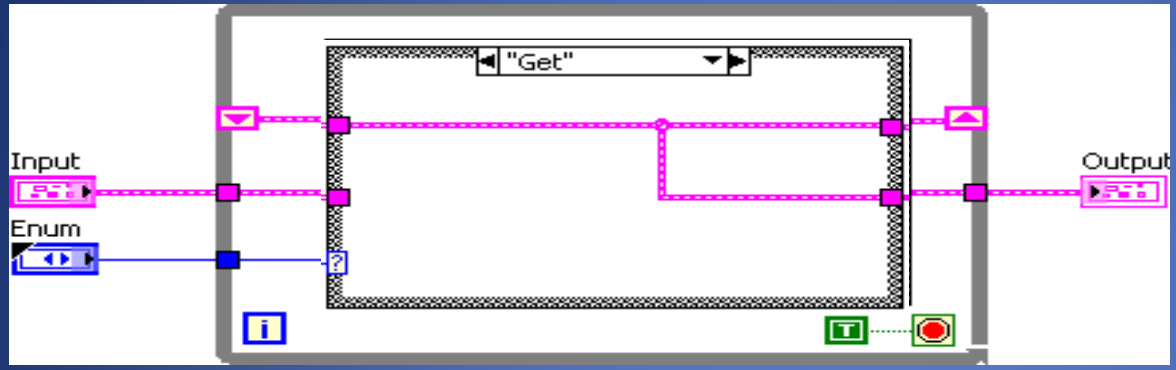
Shift Registers

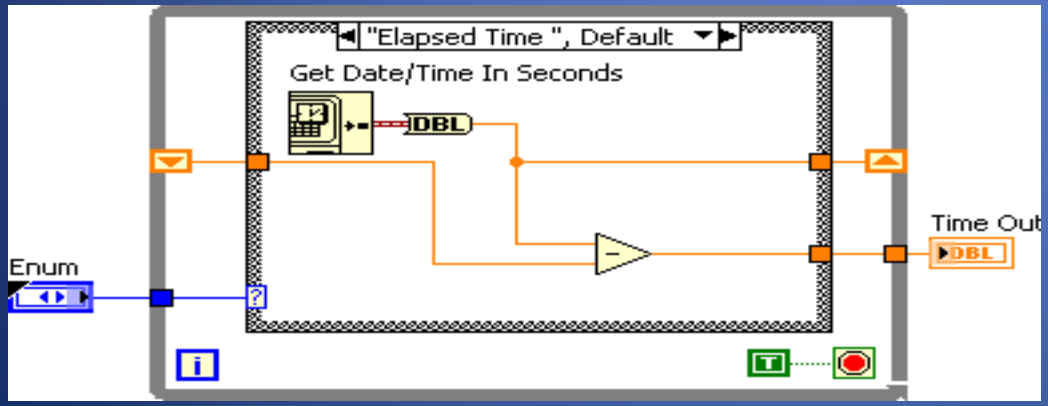
- Stacked shift registers remember values from multiple previous iterations and carry those values to the next iterations.
- Right-click the left shift register and select Add Element from the shortcut menu to stack a shift register.

FGV


"Action 1"
Operation
Input
Functional Global Variable Code
Output
i
T

Implementing An FGV





FGV - Demo

Demo - Shooter Speed

Demo

- Parallel Loop Paradigm

Variables

Variables can be of the following types:

- **Local**—Stores data in front panel controls and indicators.
- **Global** —Stores data in special repositories that can be accessed from multiple VIs.
- **Functional Global**—Stores data in While Loop shift registers.
- **Shared**—Transfers data between various distributed targets connected together over a network.

Readable Code

Enums

Enums

- Control – menu
- Constant – readable code
- Example

Enums

- Control – menu
- Constant – readable code
- Example

Timer - Demo

- FGV
- Enums

Timer - Demo

- FGV
- Enums

Questions

Auto Wire and Auto Index

- Auto Wire
  - Useful to quickly connect unchanged values in loop or case structure

Auto Wire and Auto Index

- Auto Wire
  - Useful to quickly connect unchanged values in loop or case structure
- Auto Index
  - Primarily for reading or creating arrays

Auto Wire and Auto Index

- Auto Wire
  - Useful to quickly connect unchanged values in loop or case structure
- Auto Index
  - Primarily for reading or creating arrays

VI Properties

- SR Flip Flop Demo

VI Properties

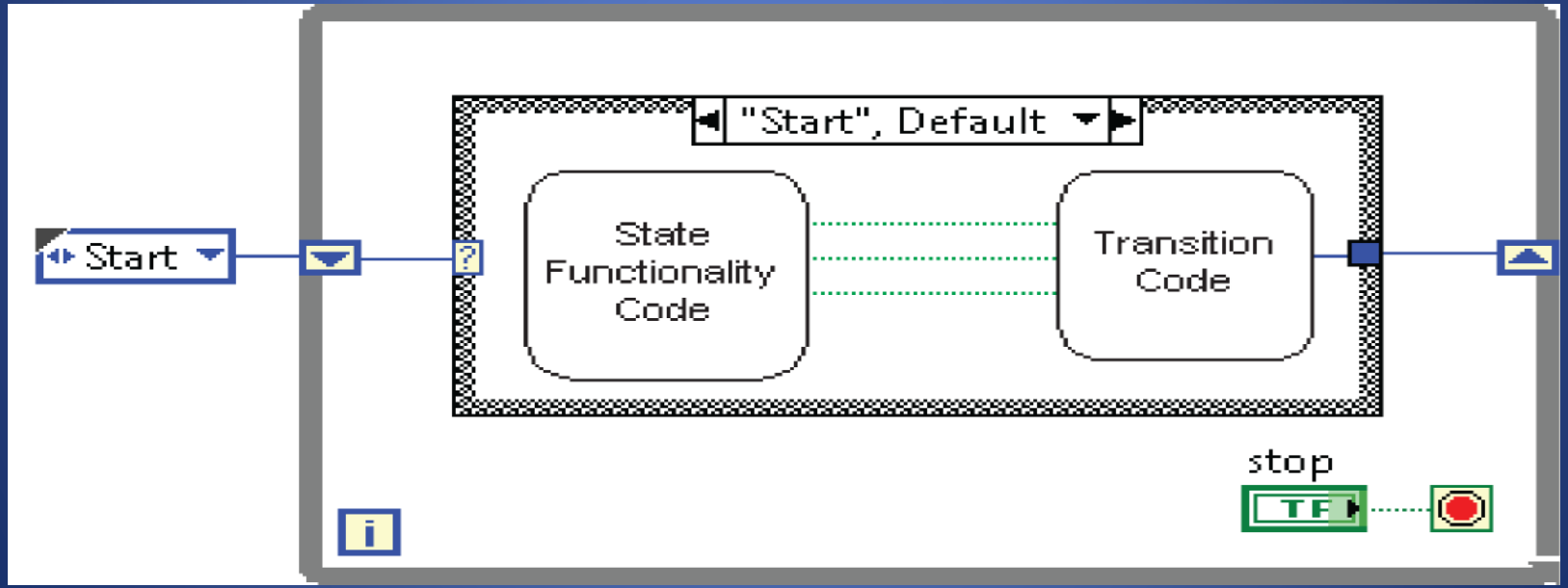
- SR Flip Flop Demo
  - Edge Detector

Architectures

- State Machine

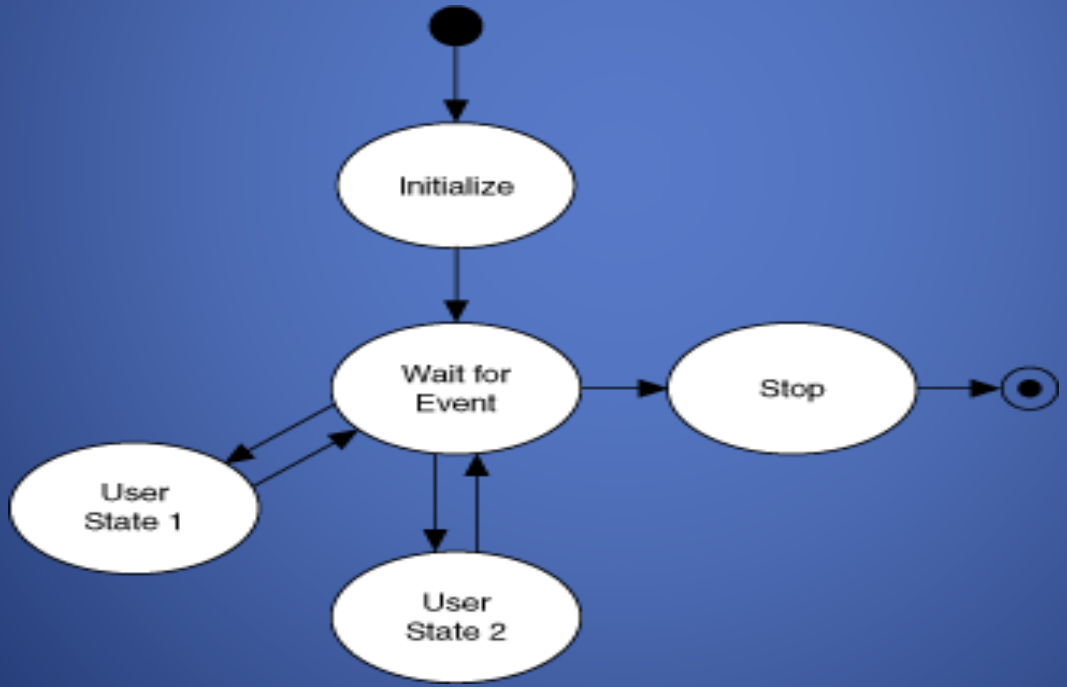
Architectures

- State Machine



Architectures

- State Machine



Architectures

- State Machine
- Producer-Consumer
  - Parallel loops
    - First creating data or instructions
    - Other handling

Architectures

- State Machine
- Producer-Consumer
  - Parallel loops
  - Use either queue or fgv

Producer Consumer Demo

Type Def.

- Useful for passing data – both controls and indicators
- Demo

Type Def.

- Useful for passing data – both controls and indicators
- Demo

Advanced Debugging Tools

- VI Profiler
  - Tools>>Profile>>Performance and Memory

OOP and LVOOP

- Object Oriented Programming
  - Used in C++, C#, Java, Python, etc.
  - A method of grouping where
    - An object represents the data
    - Has attributes and/or properties
    - Has methods that act on the object and its properties
- LVOOP is OOP in LabVIEW

LVOOP

- Demos – custom motor control

LVOOP

- Demos – custom motor control
  - Create a class

LVOOP

- Demos – custom motor control
  - Create a class
  - Create methods

LVOOP

- Demos – custom motor control
  - Create a class
  - Create methods
  - Create an object

Demo

Auto Wire & Auto Index