Most Commonly Missed Topics on the CLAD

NATIONAL INSTRUMENTS™

CLAD Prep Topics

Most Commonly Missed

- Coercion Dots
- Loop Components
- Shift Registers and Feedback Nodes
- VI Timing
- Property Nodes
- Charts and Graphs
- Arrays
- Array Functions
- Mechanical Action of Booleans
- Data Representation
- Sequence Structures
- Case Structures
- Data Flow
- Breaking Data Flow
- String Functions
- Clusters
- Type Definitions
- References

Additional Concepts

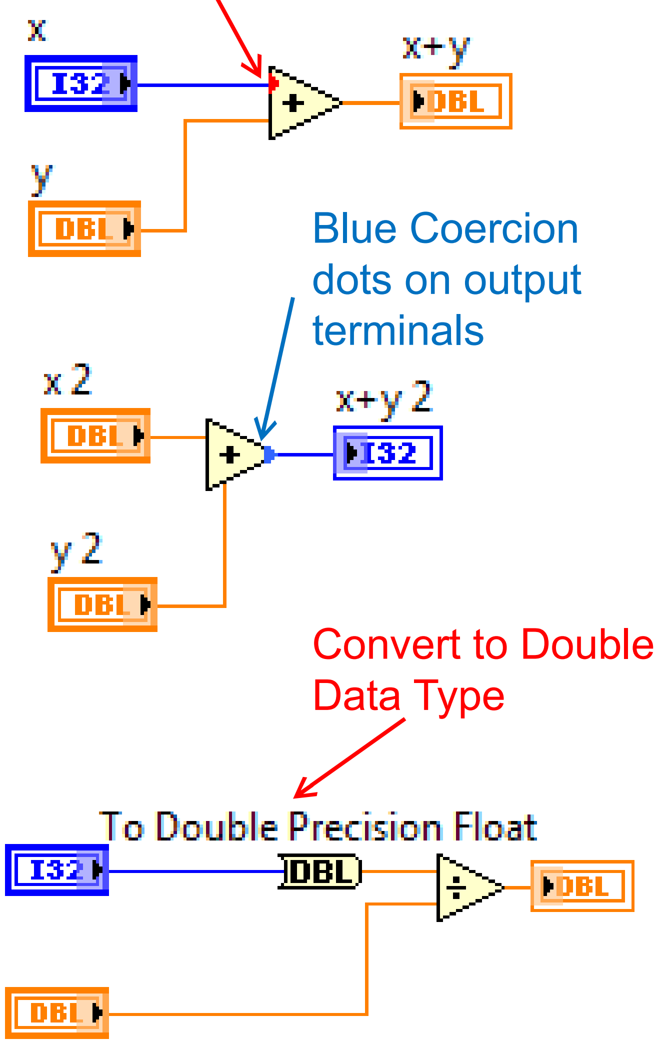
- Enumeration
- Conditional Disable Structure
- Notify and Filter Events
- Multiple Loop Design Pattern
- Notifiers
- Semaphores
- File Formats
- Modularity and SubVIs
- Icon and Connector Pane
- Errors and Warnings
- Variables
- Race Conditions from Shared Resources
- VI Server Organization
- Create references programmatically

Coercion Dots

- Coercion Dots indicate that LabVIEW has converted a value to a different numeric representation

- LabVIEW coerces values to the data type with the largest number of bits, except for For Loops (always 32-bit signed integer)

- Avoid coercion dots for best programming efficiency



Red Coercion dots on input terminals

Blue Coercion dots on output terminals

Convert to Double Data Type

Coercion Dots

When does the coercion dot appear?

A. The data types are consistent

B. A polymorphic operation is performed on the data

C. The data conversion is forced due to the mismatched numeric representation between terminals

D. Data values are being coerced because they are out of range

Coercion Dots

When does the coercion dot appear?

A. The data types are consistent

B. A polymorphic operation is performed on the data

C. The data conversion is forced due to the mismatched numeric representation between terminals

D. Data values are being coerced because they are out of range

Coercion Dots

Which of the following statements is true regarding the use of Coercion Dots?

a. Coercion Dots improve program performance.

b. Coercion Dots represent a conversion from one data type to another.

c. Coercion Dots increases memory usage

d. Both A. and B.

e. Both B. and C.

Coercion Dots

Which of the following statements is true regarding the use of Coercion Dots?

a. Coercion Dots improve program performance.

b. Coercion Dots represent a conversion from one data type to another.

c. Coercion Dots increases memory usage

d. Both A. and B.

e. Both B. and C.

Coercion dots indicate that a certain data type is being wired to terminal that accepts a different but compatible data type. When this happens, LabVIEW converts the **data to the larger of the two data types.** This requires **the creation of a memory buffer to store** the coerced data.

Coercion Dots

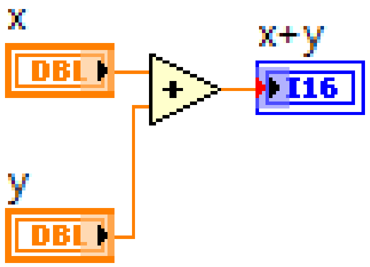
What will “Result” be after running the VI?

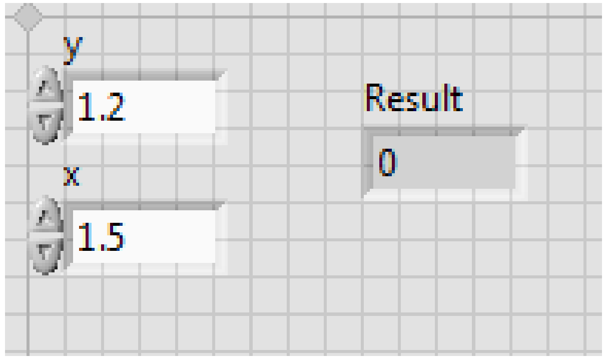
A. 2

B. 2.7

C. 3

D. NaN





y

1.2

x

1.5

Result

0

Coercion Dots

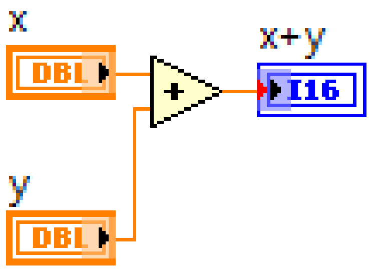
What will 'Result' be after running the VI?

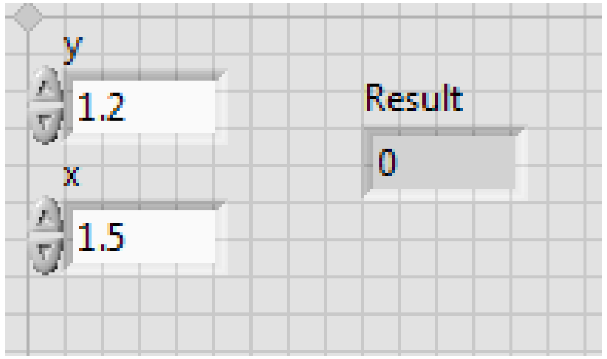
A. 2

B. 2.7

C. 3

D. NaN





y

1.2

x

1.5

Result

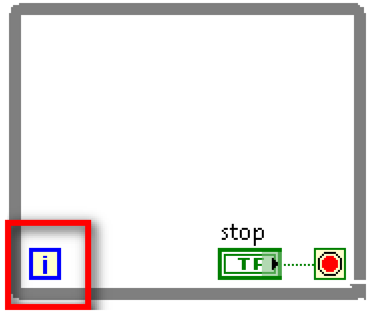
0

The block diagram places the coercion dot on the border of the terminal where the conversion takes place. The data is not coerced until after the addition, at which point it rounds to the nearest integer.

1.2 + 1.5 = 3 (2.7 rounded to 3)

Loops

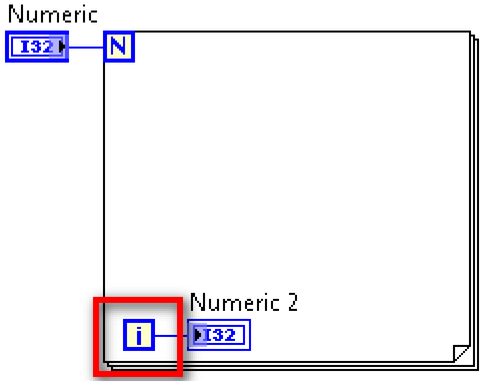
For Loops and While Loops have an Iteration Terminal to display how many times the loop has executed



•While Loops must execute at least once

•For Loops can execute zero times

•The Iteration Terminal is zero indexed; this means the output of the terminal is zero for the first iteration of the loop.



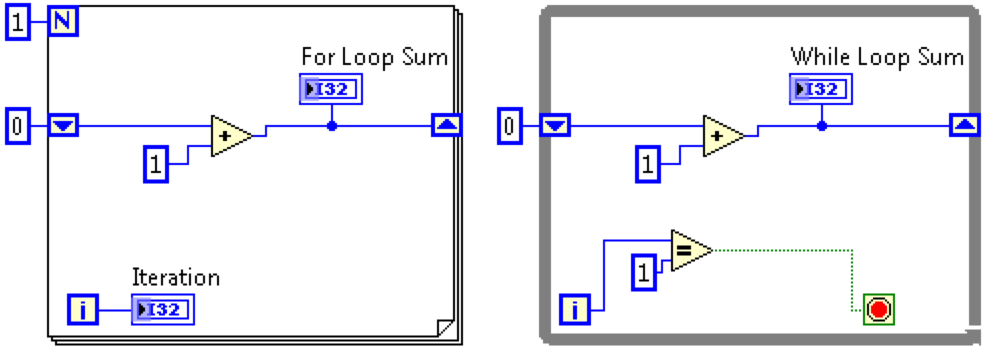
Loops

What will be shown on the “For Loop Sum”, “While Loop Sum”, and “Iteration” indicators when the program below is run?

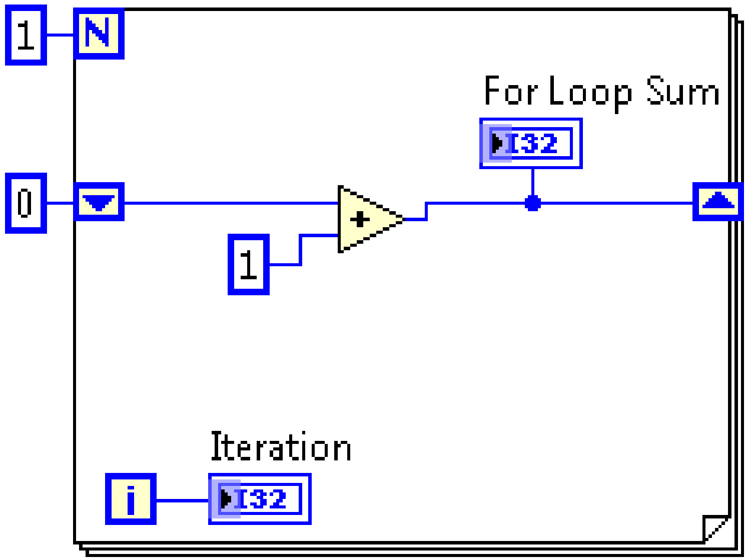
A. For Loop Sum= 1, Iteration=0, While Loop Sum= 1

B. For Loop Sum=2, Iteration=1, While Loop Sum=2

C. For Loop Sum=1, Iteration=0, While Loop Sum=2

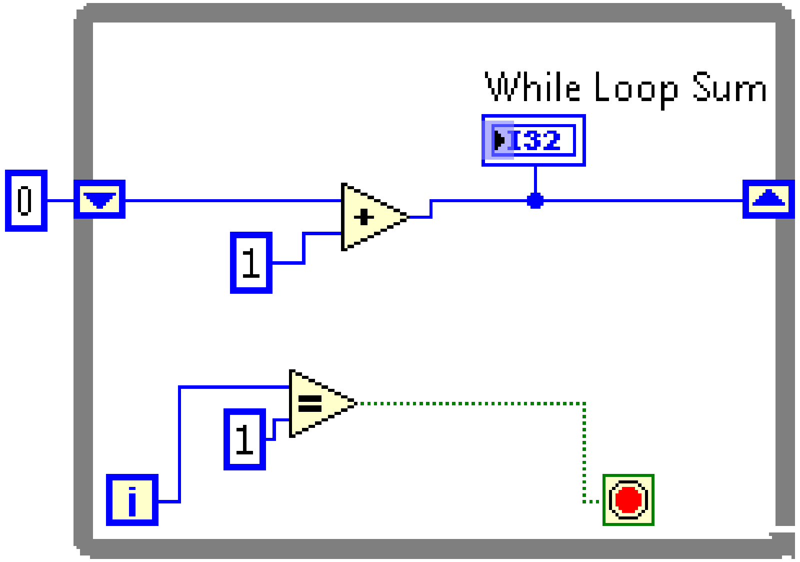


Loops



N	Stop Condition Met (N>input)?	Addition operation	"For Loop Sum" value	"Iteration" value
1	no	0+1=1	1	0
2	yes	Does not execute	1 (no change)	0 (no change)

Loops



Iteration	Addition operation	"While Loop Sum" value	i value	Stop condition met (i=1)?
1	0+1=1	1	0	no
2	1+1=2	2	1	yes

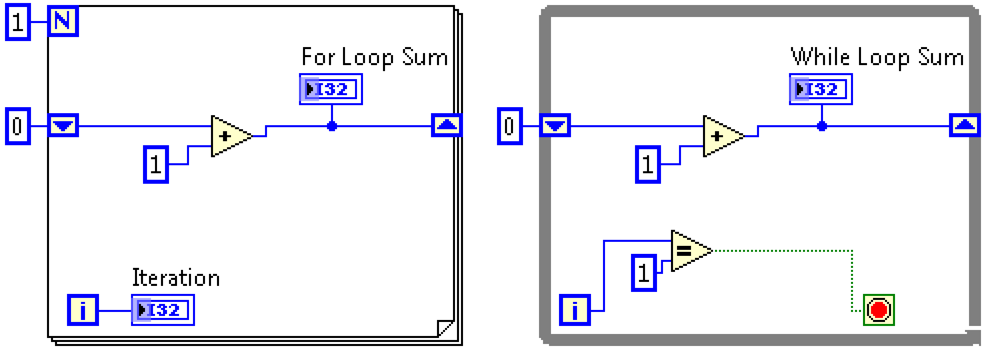
Loops

What will be shown on the “For Loop Sum”, “While Loop Sum”, and “Iteration” indicators when the program below is run?

A. For Loop Sum= 1, Iteration=0, While Loop Sum= 1

B. For Loop Sum=2, Iteration=1, While Loop Sum=2

C. For Loop Sum=1, Iteration=0, While Loop Sum=2



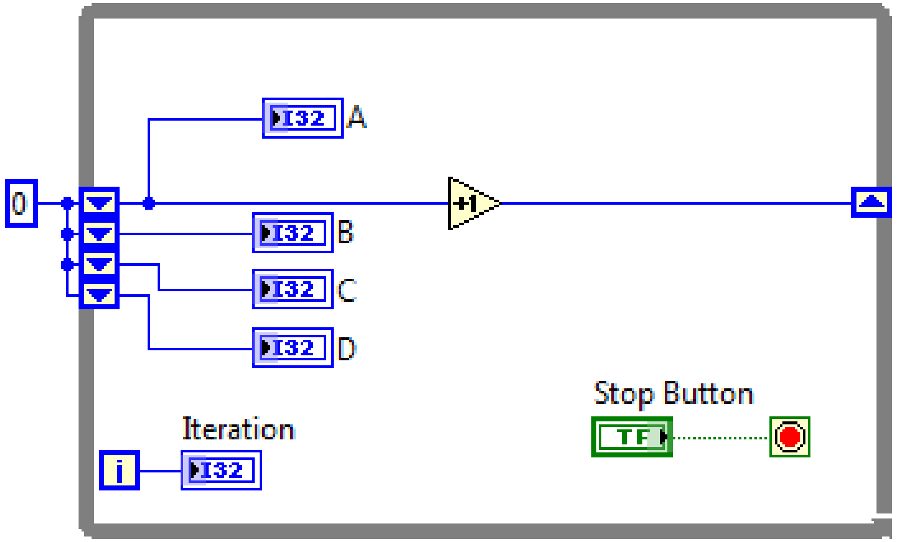
Shift Registers – Multiple Previous Values

- Shift registers can be resized to save multiple previous values.
  - Retains a certain number of previous values, oldest ones get removed as needed

Run once

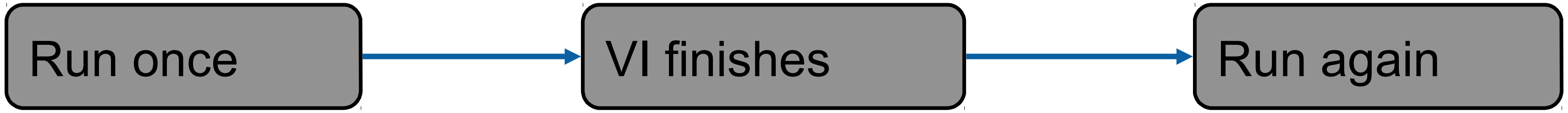
VI finishes

Run again

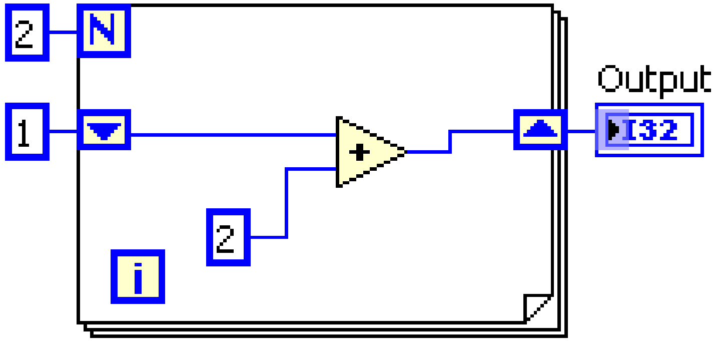
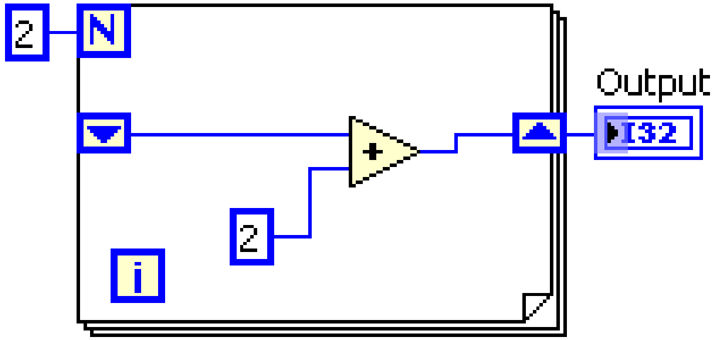


Run #:	0	1	2	3	4
Numeric:	2	5	3	9	1
A:	0	1	2	3	4
B:	0	0	1	2	3
C:	0	0	0	1	2
D:	0	0	0	0	1

Shift Registers



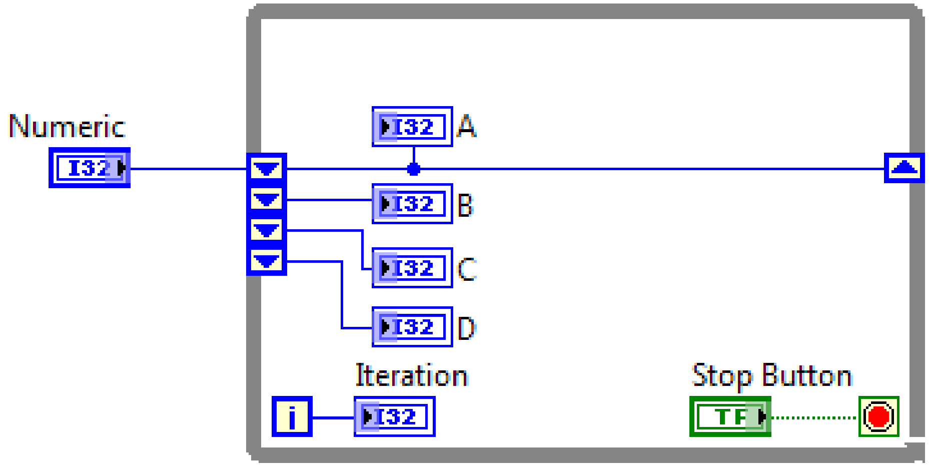
Run once → VI finishes → Run again

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

Shift Registers

After 3 iterations, what will be the value at indicator D?

a. 0
b. 10
c. 3
d. Unknown

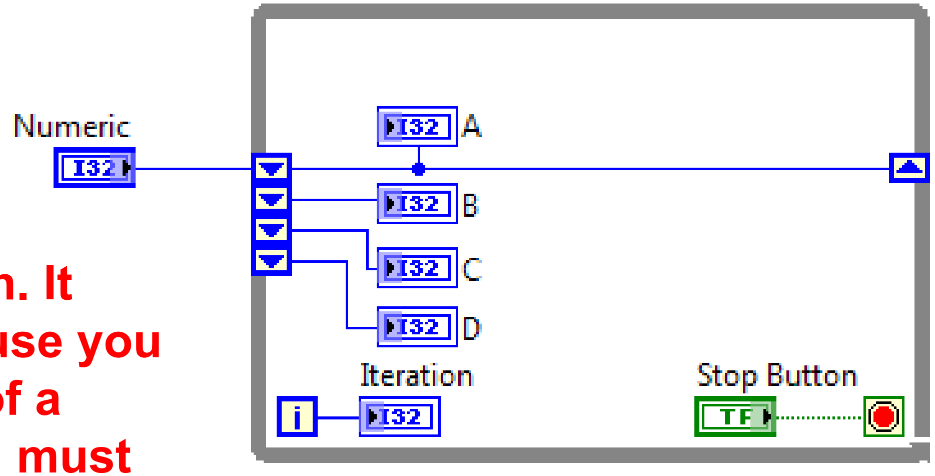


Shift Registers

After 3 iterations, what will be the value at indicator D?

a. 0
b. 10
c. 3
d. Unknown

Numeric = 10

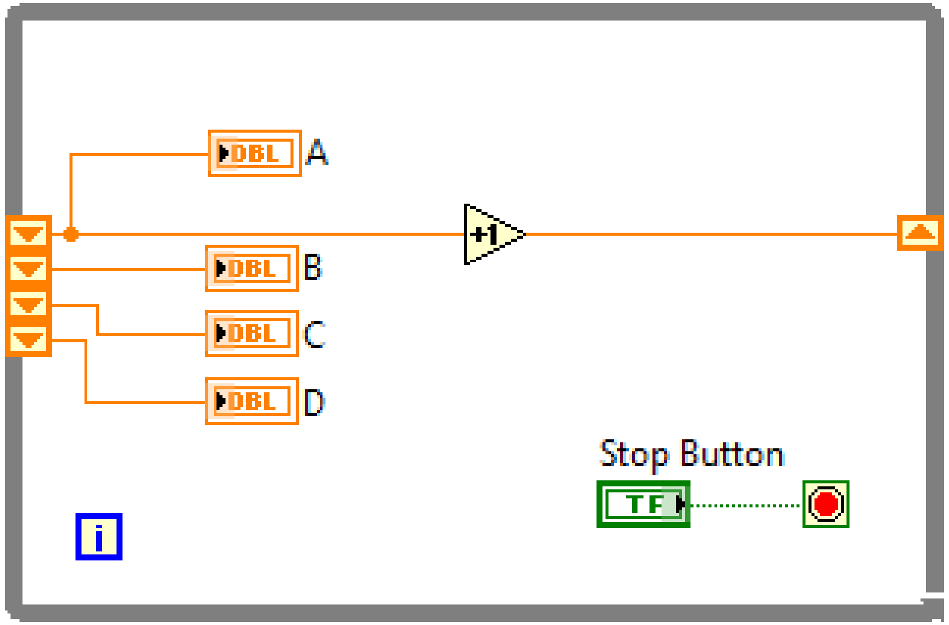


Justification: This code will not run. It will have a broken run arrow because you cannot initialize just one element of a multiple element shift register. You must either initialize all or none.

Shift Registers

After 3 iterations, what will be the value at indicator D?

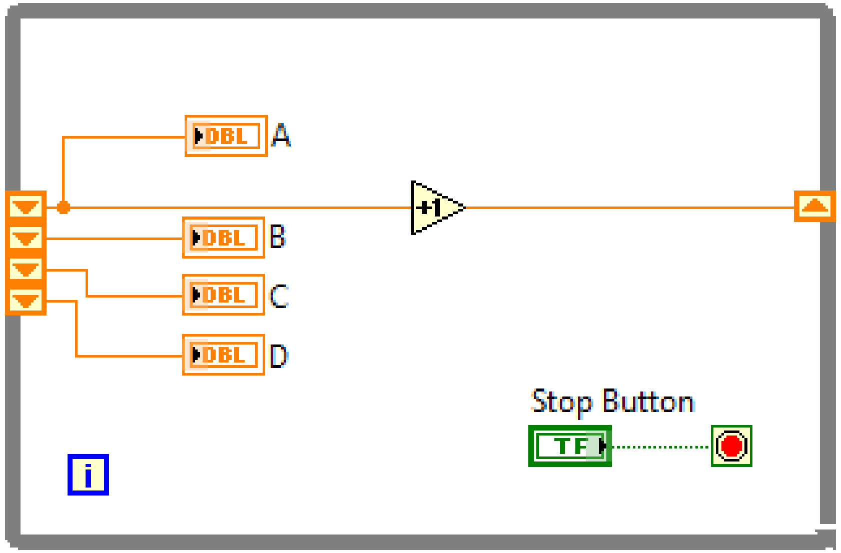
a. 0
b. 10
c. 3
d. Unknown



Shift Registers

After 3 iterations, what will be the value at indicator D?

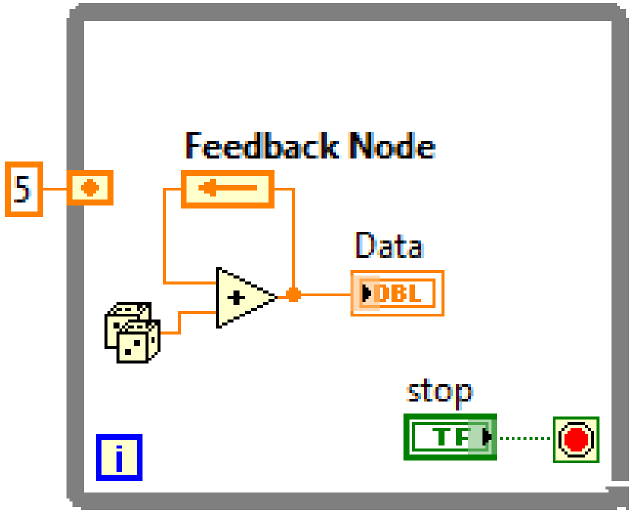
a. 0
b. 10
c. 3
d. Unknown



Justification: If this is the first time we are running this VI, D will display 0 after 3 iterations. Because the shift registers are uninitialized, we cannot say what D will be.

Feedback Node

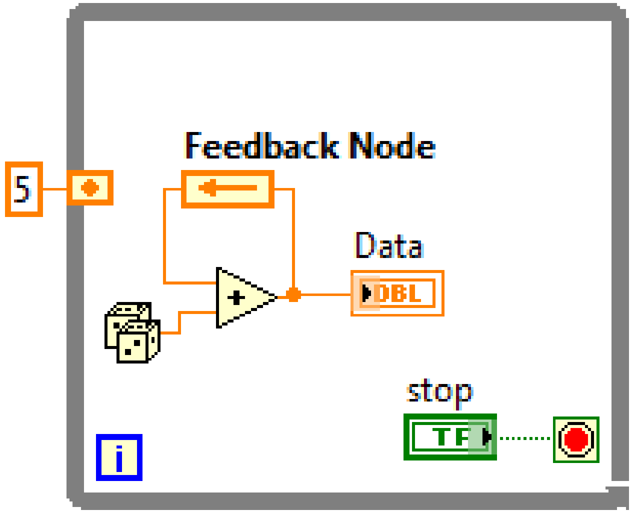
- A feedback node functions like a shift register.
  - Can set an initial value by wiring data into the terminal on the left side of the loop.



Feedback Node

What number will be added to the random number on the first iteration of this loop?

a. 0
b. Unknown
c. 5
d. 2



Feedback Node

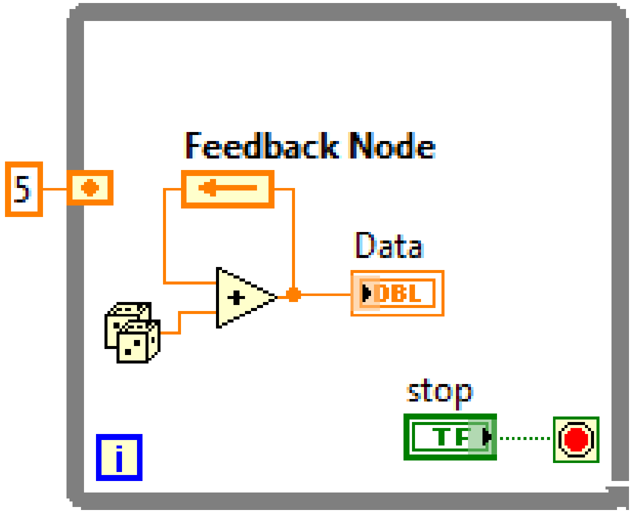
What number will be added to the random number on the first iteration of this loop?

a. 0

b. Unknown

c. 5

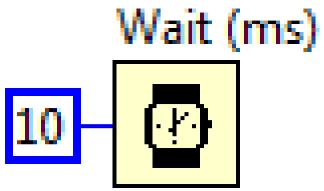
d. 2

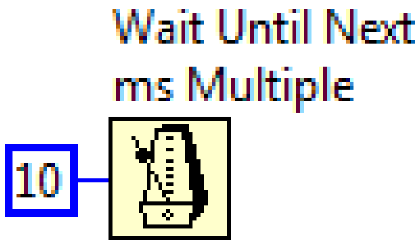


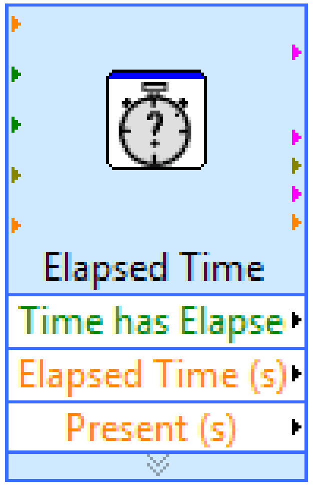
Justification: A feedback node acts exactly like a shift register. The terminal on the left side sets an initial value for the feedback node, which is used during the first iteration of the loop.

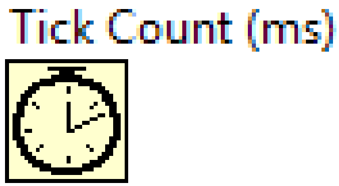
Timing Functions

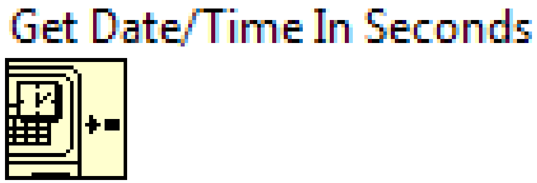
- Control or measure the frequency at which a loop executes
- Provide the processor with time to complete other tasks, such as processing the user interface
- Uses the operating system millisecond clock







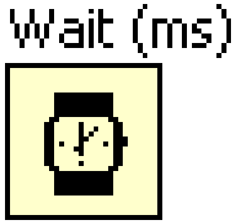




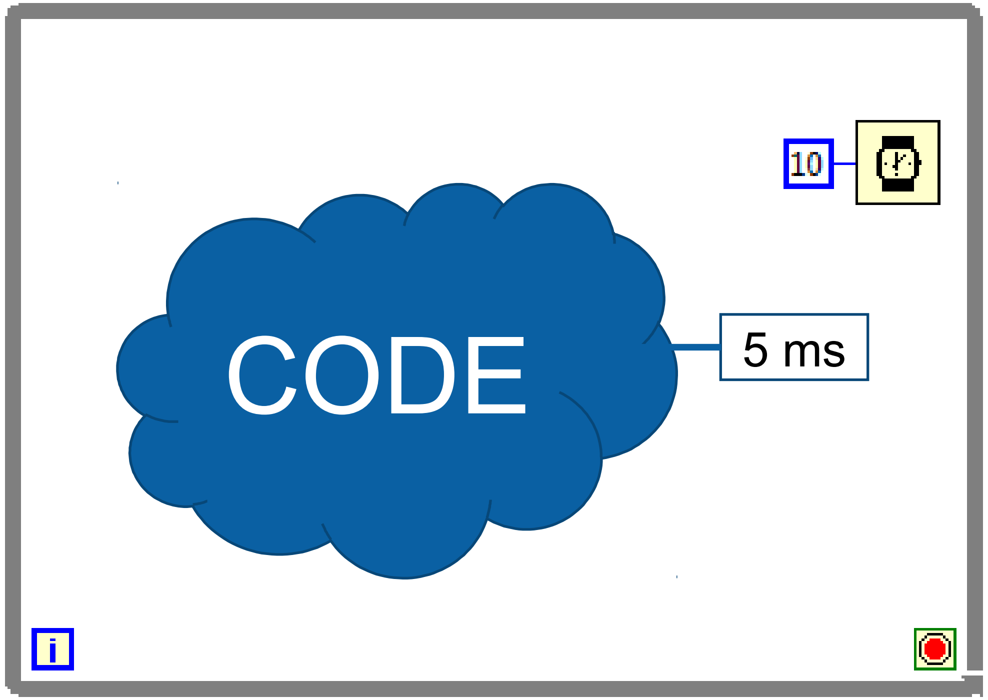
Wait (ms)

Waits for specified input number of milliseconds

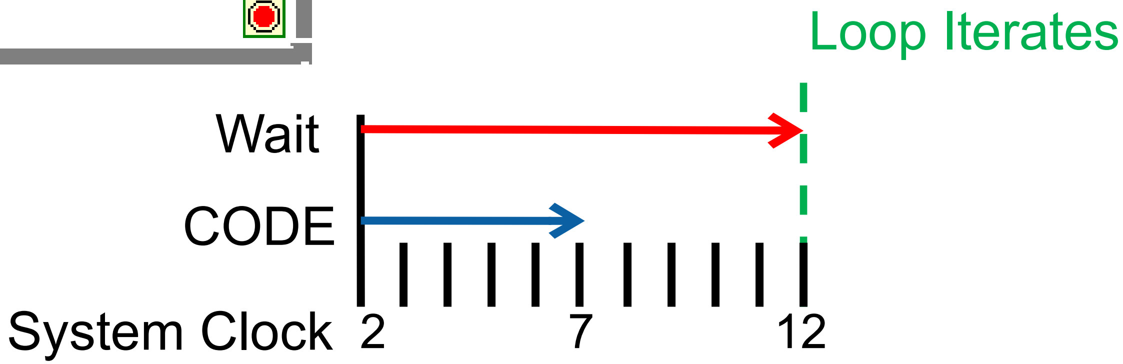
- Performs function asynchronously
- Returns only after the specified wait time
- Does not interrupt parallel code
- Loops will not iterate until function returns



Wait (ms)



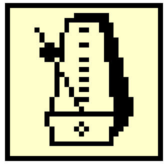
The Wait (ms) function will begin its count when the loop begins executing. The CODE will execute in 5 ms, and the loop will not continue until an additional 5 ms elapses.



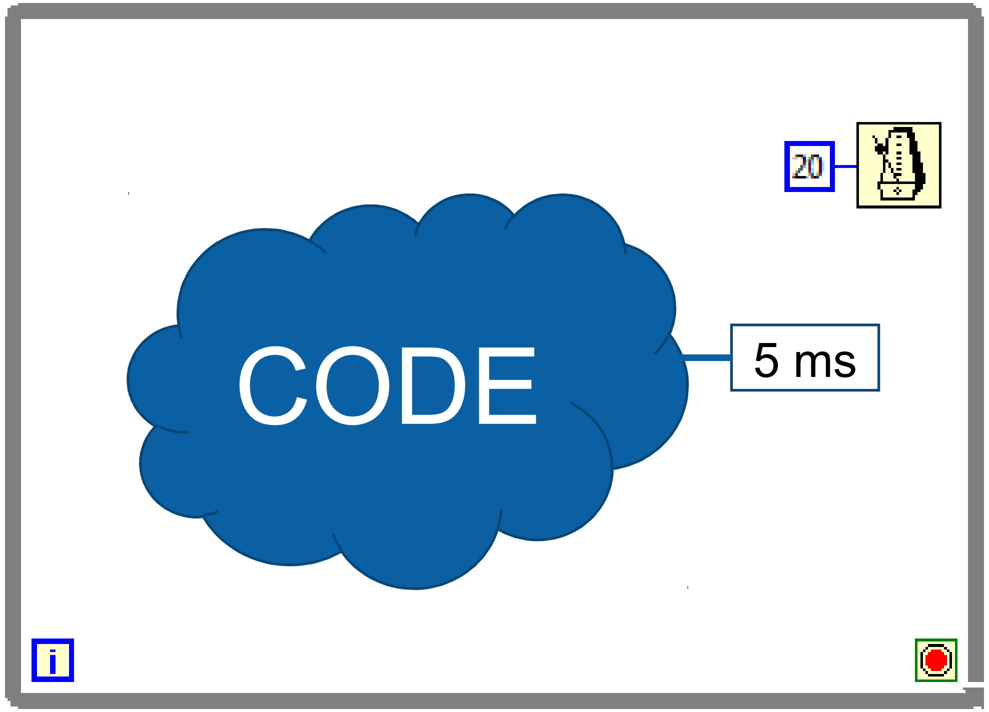
Wait Until Next ms Multiple

- Waits until the value of the millisecond timer becomes a multiple of the specified millisecond multiple
- Performs function asynchronously
- Guarantees that the loop execution rate is at least the amount of the input you specify

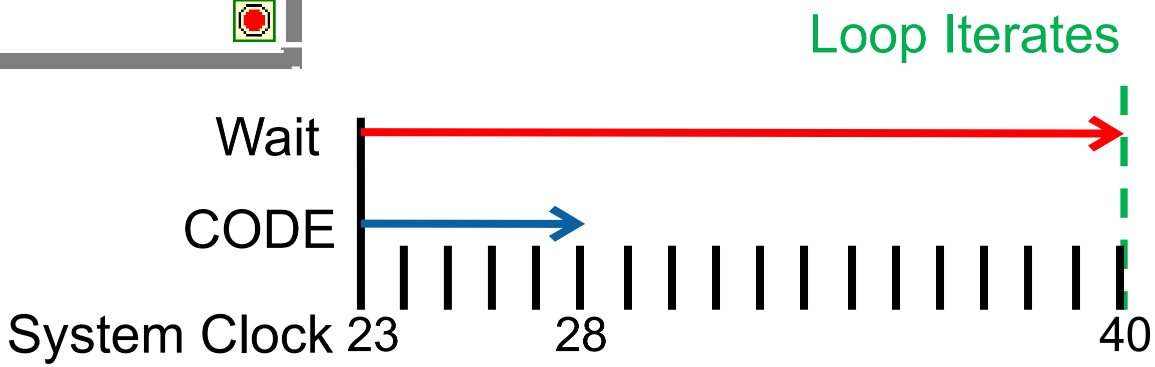
Wait Until
Next ms Multiple



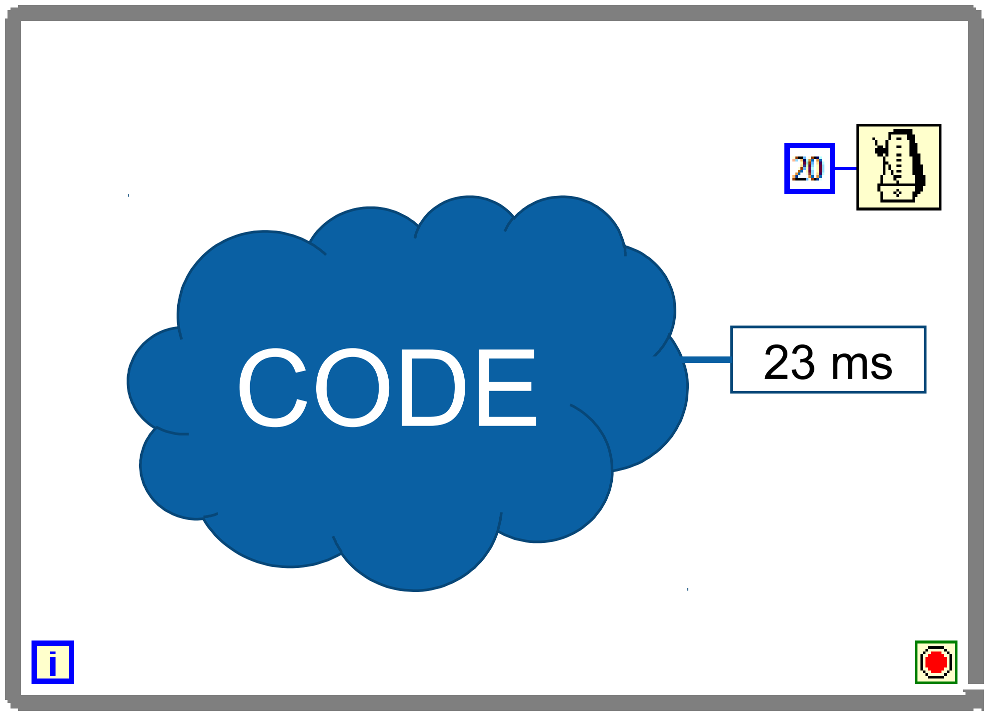
Wait Until Next ms Multiple



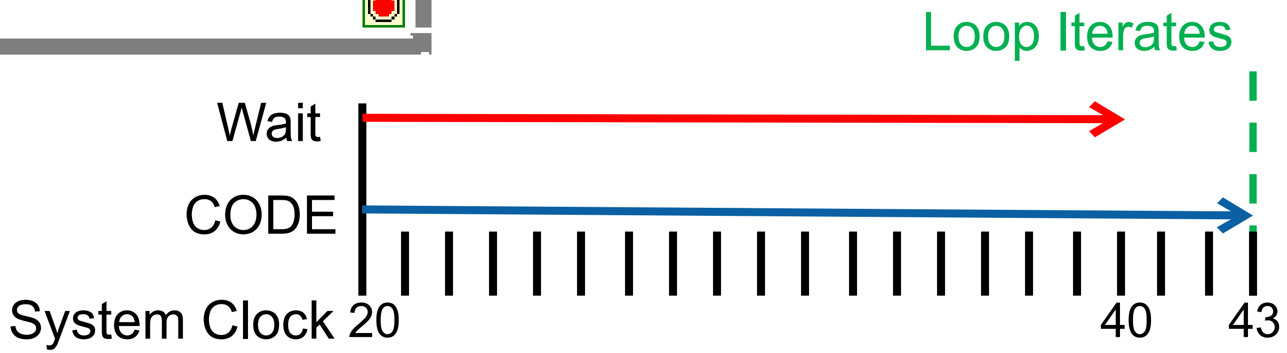
The CODE will execute in 5 ms, and the loop will not iterate until a multiple of 20 ms is reached on the system clock. The loop period may be less than 20ms in the first iteration.



Wait Until Next ms Multiple



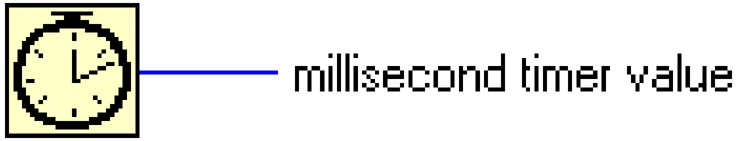
The loop will always iterate after 23 ms, since the CODE will execute in 23 ms. A multiple of 20 ms will always be reached before 23 ms, so the Wait Until Next ms Multiple has no effect in this situation.



Tick Count (ms)

Tick Count (ms) returns the value of the Windows millisecond timer.

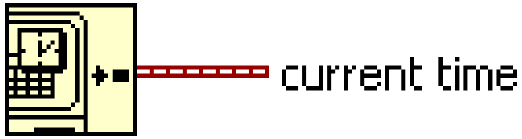
- Timer rolls over at 2^{32}-1
- Cannot be used to extract real-world date or time
- Useful for finding the duration of executing code



Get Date/Time In Seconds

Get Date/Time In Seconds returns a timestamp of the current time.

- Timestamp value is calculated using number of seconds elapsed since 12:00 AM, Jan 1, 1904, Universal Time.
- Do not use to calculate code execution duration



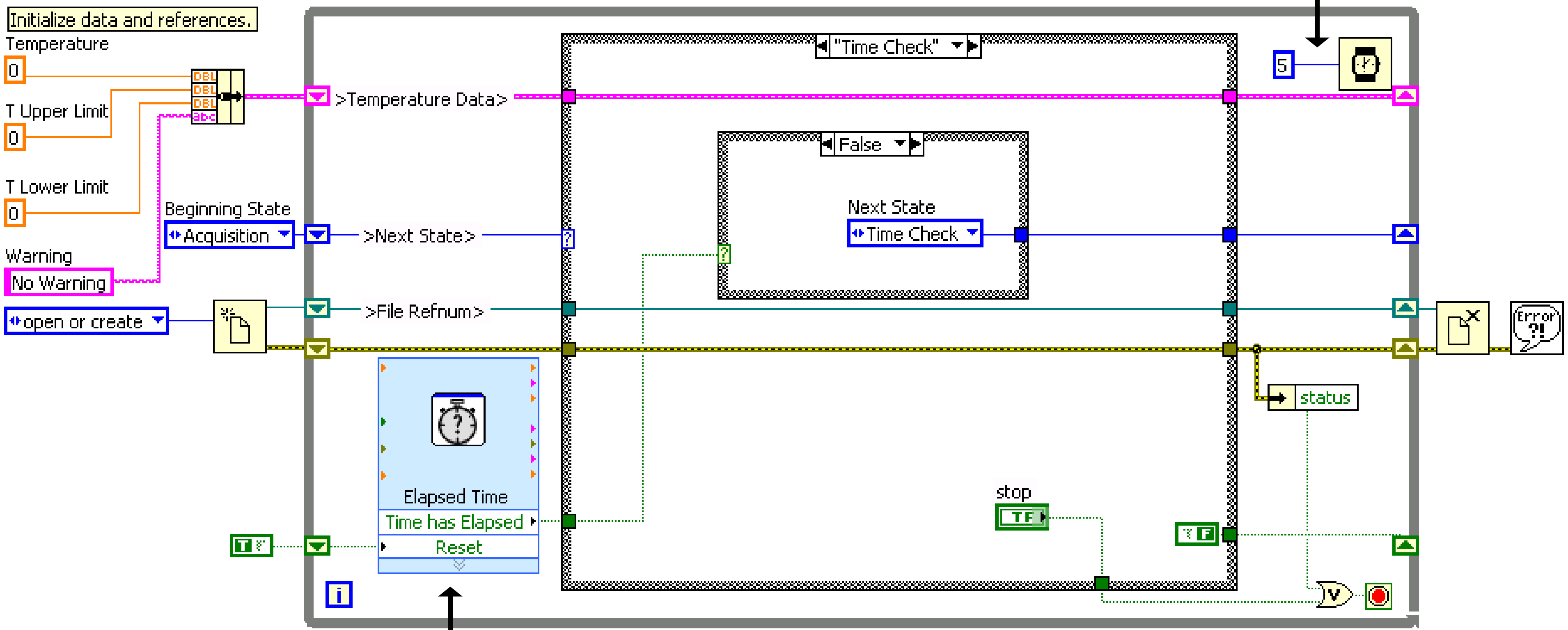
VI Timing Types

- Execution Timing
  - Provides the design pattern with a function that specifically allows the processor time to complete other tasks
  - In some cases, a Wait function is not necessary
    - Use of Timeout inputs can provide execution timing
- Software Control Timing
  - Timing (pauses, waits, time checks) you put in place to make the code execute after a certain amount of time.

  - *Example*: If you must acquire data for 5 minutes, you could remain in the acquisition state until the 5 minutes elapses. However, during that time you cannot process any user interface actions such as stopping the VI. To process user interface actions, you must implement timing (checking if 5 min has passed since the start of acquisition) so that the VI continually executes for the specified time.

VI Timing

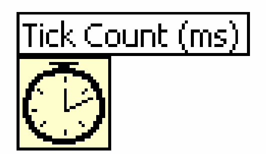
Execution Timing



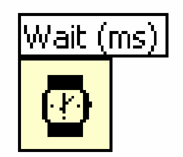
Software Control Timing

Timing Functions

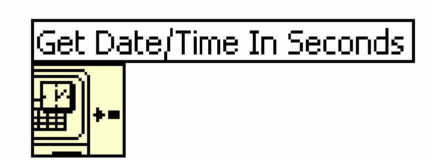
Which timing function is the best choice for timing control logic in applications that run for extended periods of time?

A. 

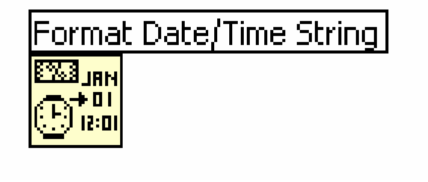
Tick Count (ms)

B. 

Wait (ms)

C. 

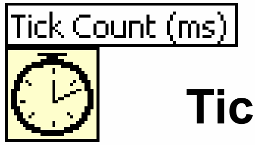
Get Date/Time In Seconds

D. 

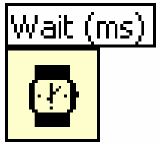
Format Date/Time String

Timing Functions

Which timing function is the best choice for timing control logic in applications that run for extended periods of time?

A. 

Tick Count rolls over, so it is best for very short periods of time

B. 

Wait should be used for execution timing, not control timing.

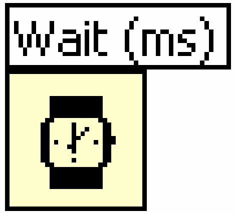
C.

Getting and storing the system time is the best way to measure elapsed time in applications that run for extended periods of time.

D.

This function formats a time stamp; it is not used for VI timing.

Timing Functions



The Wait function can be added to While Loops:

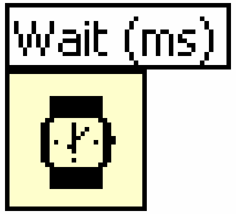
a. To free up available memory.

b. To allocate memory used by the CPU.

c. To allow the processor time to complete other tasks.

d. To reserve which processor the code is running on.

Timing Functions



The Wait function can be added to While Loops:

a. To free up available memory.

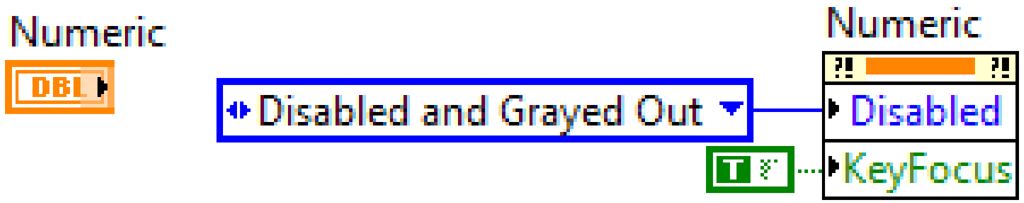
b. To allocate memory used by the CPU.

c. To allow the processor time to complete other tasks.

d. To reserve which processor the code is running on.

By default, LabVIEW executes loops and block diagrams as quickly as possible, and it immediately begins running the next. You can use a Wait function in the loop to specify the amount of time in milliseconds before the loop re-executes. Timing a loop also allows the processor time to complete other tasks such as updating and responding to the user interface.

Property Nodes



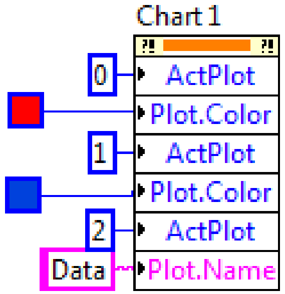
- Access the properties of an object
- Enable you to modify the appearance of front panel objects programmatically in response to certain inputs.

  For example:
  - If a user enters an invalid password, you might want a red LED to start blinking
  - If a data point is above a certain value, you might want to show a red trace instead of a green one
- Execute from top to bottom
  - By default, if an error occurs on 3rd of 5 terminals, last two do not execute and error is output

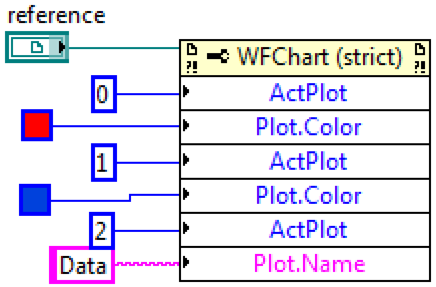
Property Nodes

- A property node can be implicit or explicit.
- A property node executes top-down

Implicit Property Node



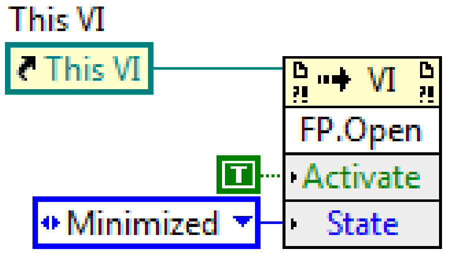
Explicit Property Node (use when subVIs are involved)



1. Plot 0 is set active
2. Active plot (0) changed to red
3. Plot 1 is set active
4. Active plot (1) is changed to blue
5. Plot 2 is set active
6. Active plot (2) changes name to “Data”

Invoke Nodes

- Use the Invoke Node to perform actions, or methods, on referenced items (VI, Control)
- Most methods have parameters
- Examples of VI Methods:
  - Front Panel: Center
  - Default Values: Reinitialize all to Default
  - Print: VI to HTML

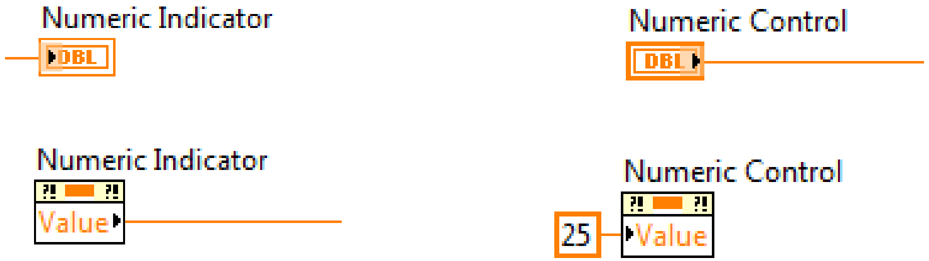


Method Name

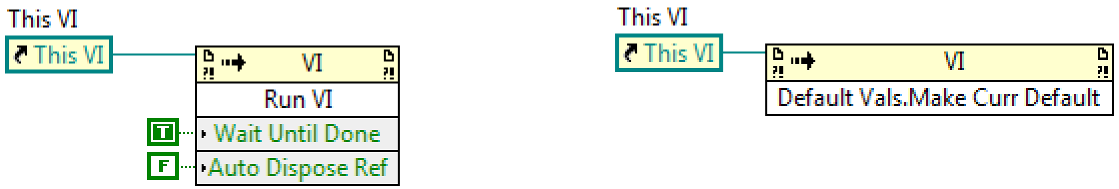
Parameters

Property and Invoke Nodes: Use Cases

- **Property Nodes** can be used to programmatically read from an indicator or write to a control



- **Invoke Nodes** can be used to programmatically control the state of a VI

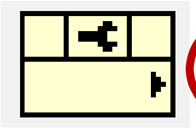


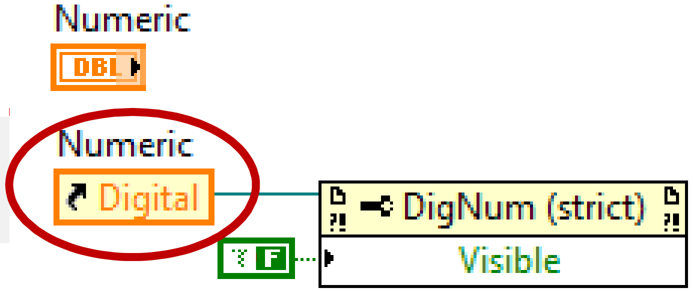
Implicitly and Explicitly Linking Invoke/Property Nodes

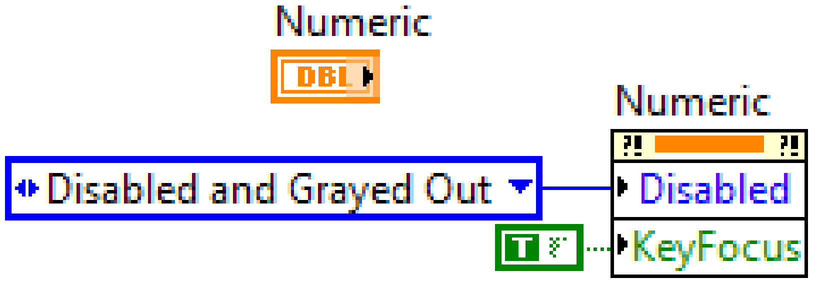
Explicitly Linked
(requires control reference)

Implicitly Linked
(color/ label for data type/label)

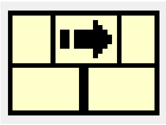
Property
Node

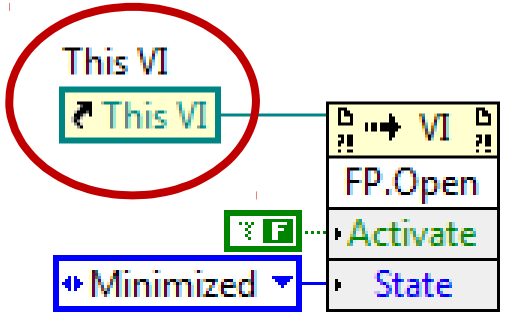


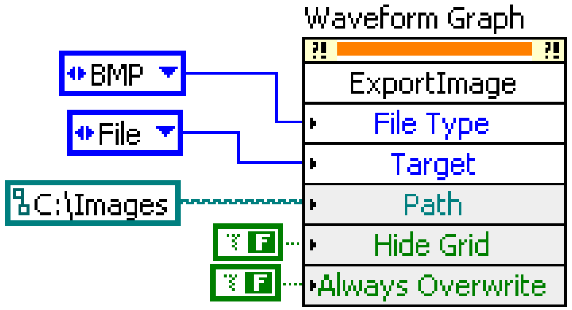




Invoke
Node







Property Nodes

Which combination of words correctly completes the following statement? Unlike ______ Property Nodes, ______ Property Nodes require ______ as inputs in order to function correctly.

a. Explicit; Implicit; Data Value References

b. Implicit; Explicit; Data Value References

c. Explicit; Implicit; Control References

d. Implicit; Explicit; Control References

Property Nodes

Which combination of words correctly completes the following statement? Unlike _____ Property Nodes, _____ Property Nodes require _____ as inputs in order to function correctly.

a. Explicit; Implicit; Data Value References

b. Implicit; Explicit; Data Value References

c. Explicit; Implicit; Control References

d. Implicit; Explicit; Control References

Implicit Property Nodes are explicitly linked to their owning control or indicator. No reference wires are needed. Explicit Property Nodes require a reference wire to determine which control the Property Node is manipulating. Data Value References have nothing to do with Property

Property Nodes

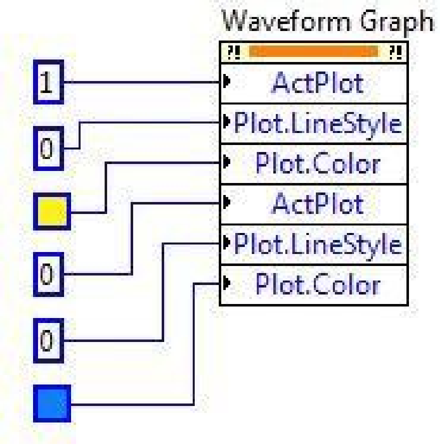
Which plot will change color first?

a. Plot 1 because properties are executed top-down

b. Plot 0 because properties are implemented in numeric order starting at zero

c. Both plots will be updated simultaneously due to the multithreading of properties

d. It cannot be determined because LabVIEW performs operations in dataflow format



Property Nodes

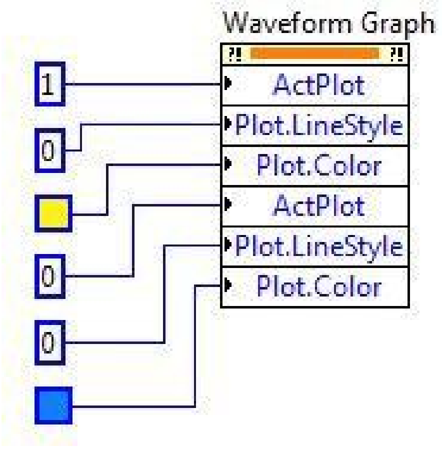
Which plot will change color first?

a. Plot 1 because properties are executed top-down

b. Plot 0 because properties are implemented in numeric order starting at zero

c. Both plots will be updated simultaneously due to the multithreading of properties

d. It cannot be determined because LabVIEW performs operations in dataflow format



Property Nodes

Which of the following apply to Property Nodes? **(More than one answer may apply.)**

a. Property Nodes allow attributes of front panel objects to be programmatically manipulated.

b. Property Nodes can be used to update the values contained in a front panel object.

c. More than one Property Node can be used for a single front panel object.

d. Property Nodes can be used to programmatically generate a Value Change event.

Property Nodes

Which of the following apply to Property Nodes? **(More than one answer may apply.)**

a. Property Nodes allow attributes of front panel objects to be programmatically manipulated.

b. Property Nodes can be used to update the values contained in a front panel object.

c. More than one Property Node can be used for a single front panel object.

d. Property Nodes can be used to programmatically generate a Value Change event.

Property Nodes

What type of property node can you use to update a property from inside a subVI?

a. Implicit Property Node

b. Local Property Node

c. Explicit Property Node

d. Value Property Node

Property Nodes

What type of property node can you use to update a property from inside a subVI?

a. Implicit Property Node

b. Local Property Node

c. Explicit Property Node

d. Value Property Node

Charts and Graphs

Graphs

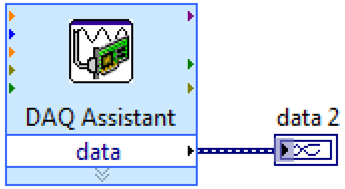
- Do not accept single point values
- All points are plotted at once

Charts

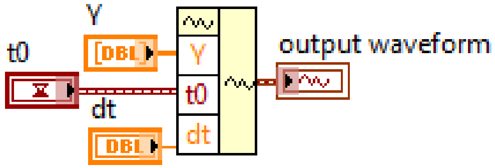
- Accept single point values
- Values are stored in a buffer, then overwritten with new values
- Points are plotted as data becomes available

Both

- Accept various data types:
  - Waveform
  - **Dynamic**
  - Arrays

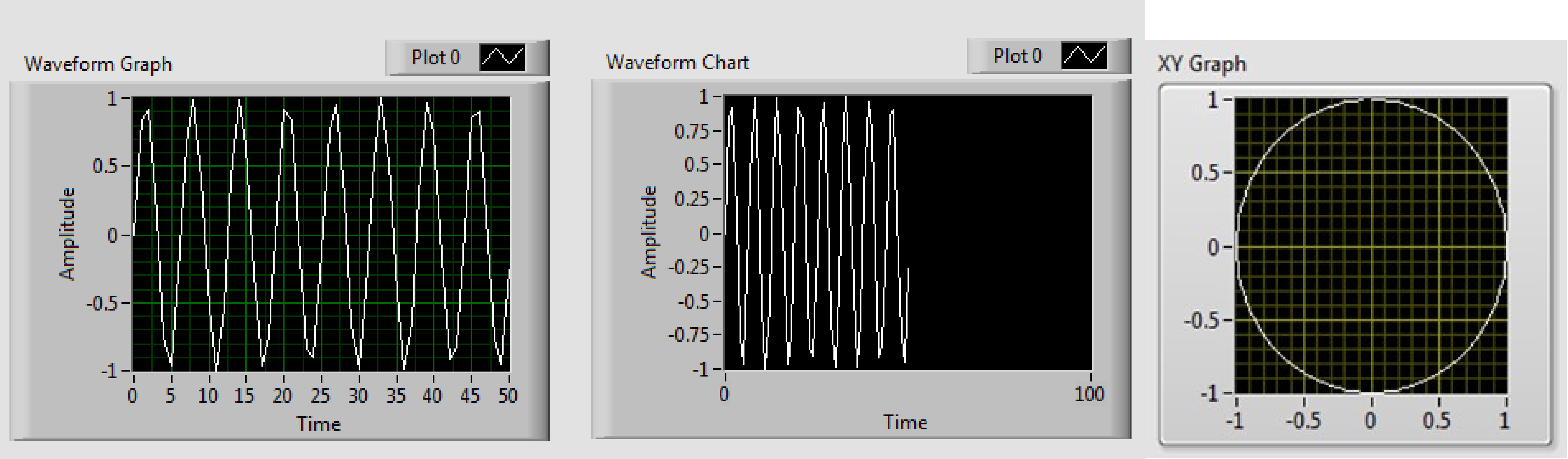


- Waveform Data types contain:
  - An array of points
  - t0
  - dt



Charts and Graphs

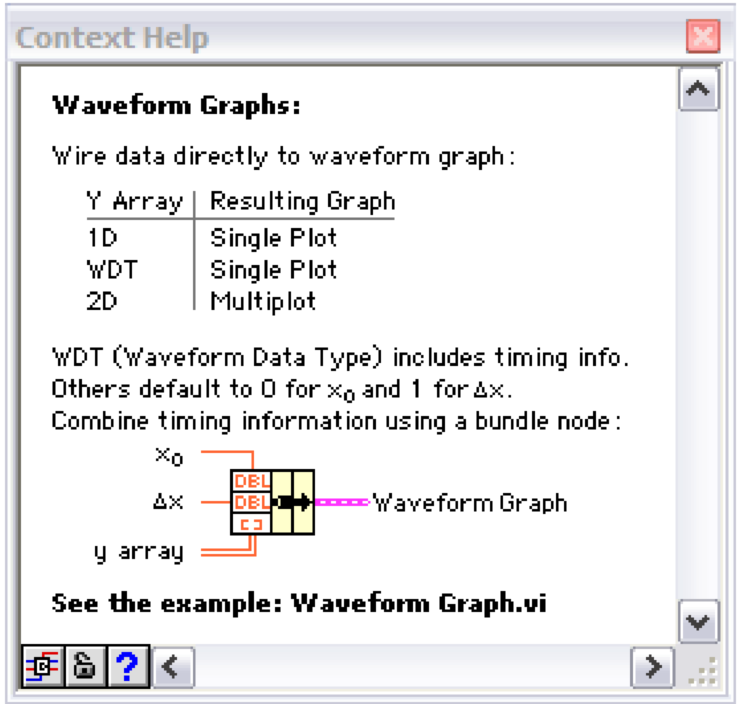
- A graph displays an entire waveform at once.
- A chart displays single points as they are received.
- An XY Graph displays individual (X,Y) pairs.
- **The graphs and chart below all display the same sine wave data.**



Charts and Graphs

- Chart:
  - Remembers history – new point added to end of plot
  - Good for inside a loop

- Graph:
  - New plot of all new data
  - Good for outside the loop
  - Use the **Context Help** window to determine how to wire multi-plot data to Waveform Graphs and XY Graphs



Waveform Graphs:

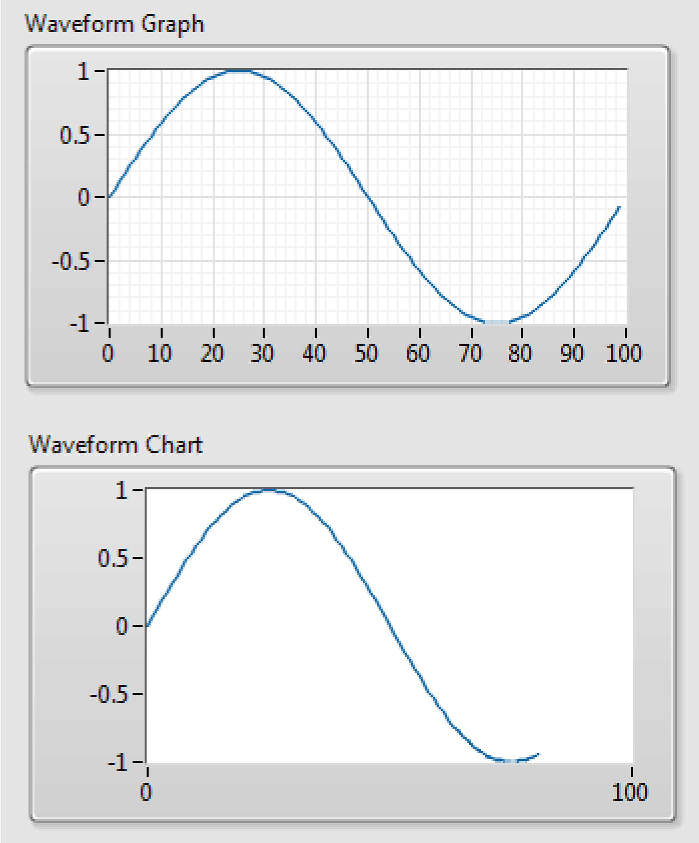
Wire data directly to waveform graph:

Y Array	Resulting Graph
1D	Single Plot
WDT	Single Plot
2D	Multiplot

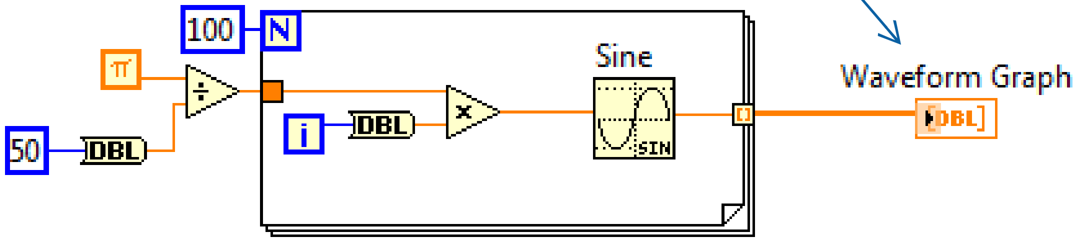
WDT (Waveform Data Type) includes timing info. Others default to 0 for x_0 and 1 for Δx. Combine timing information using a bundle node:

See the example: Waveform Graph.vi

Charts vs. Graphs – Single-Plot



Graph is **outside** the loop and only updates once: **after** the loop finishes



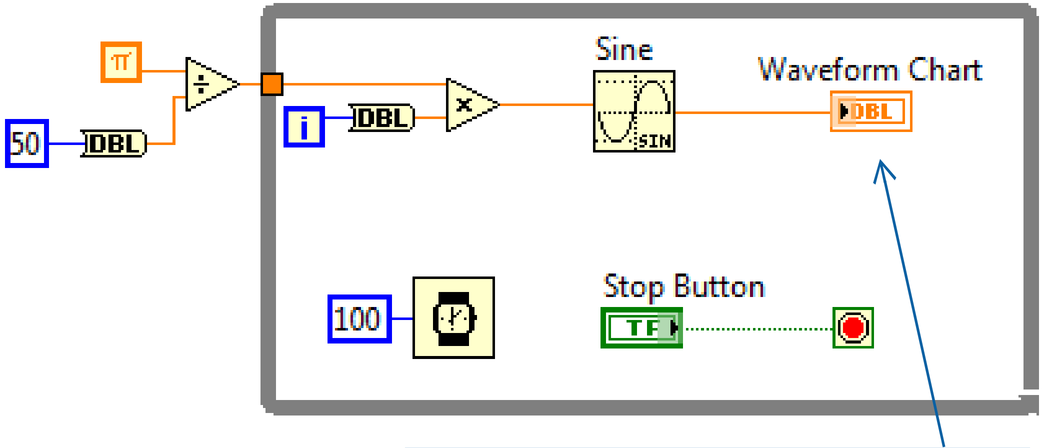
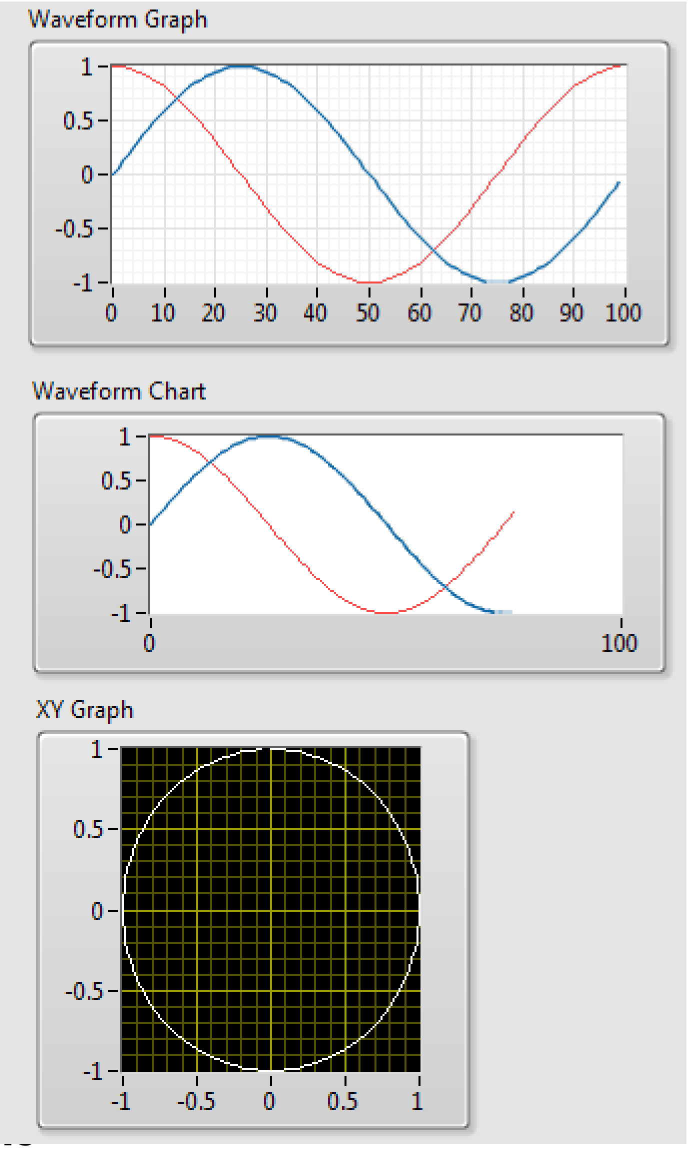


Chart is **inside** the loop and updates for **each** iteration.

Charts vs. Graphs – Multi-plot and XY Graph



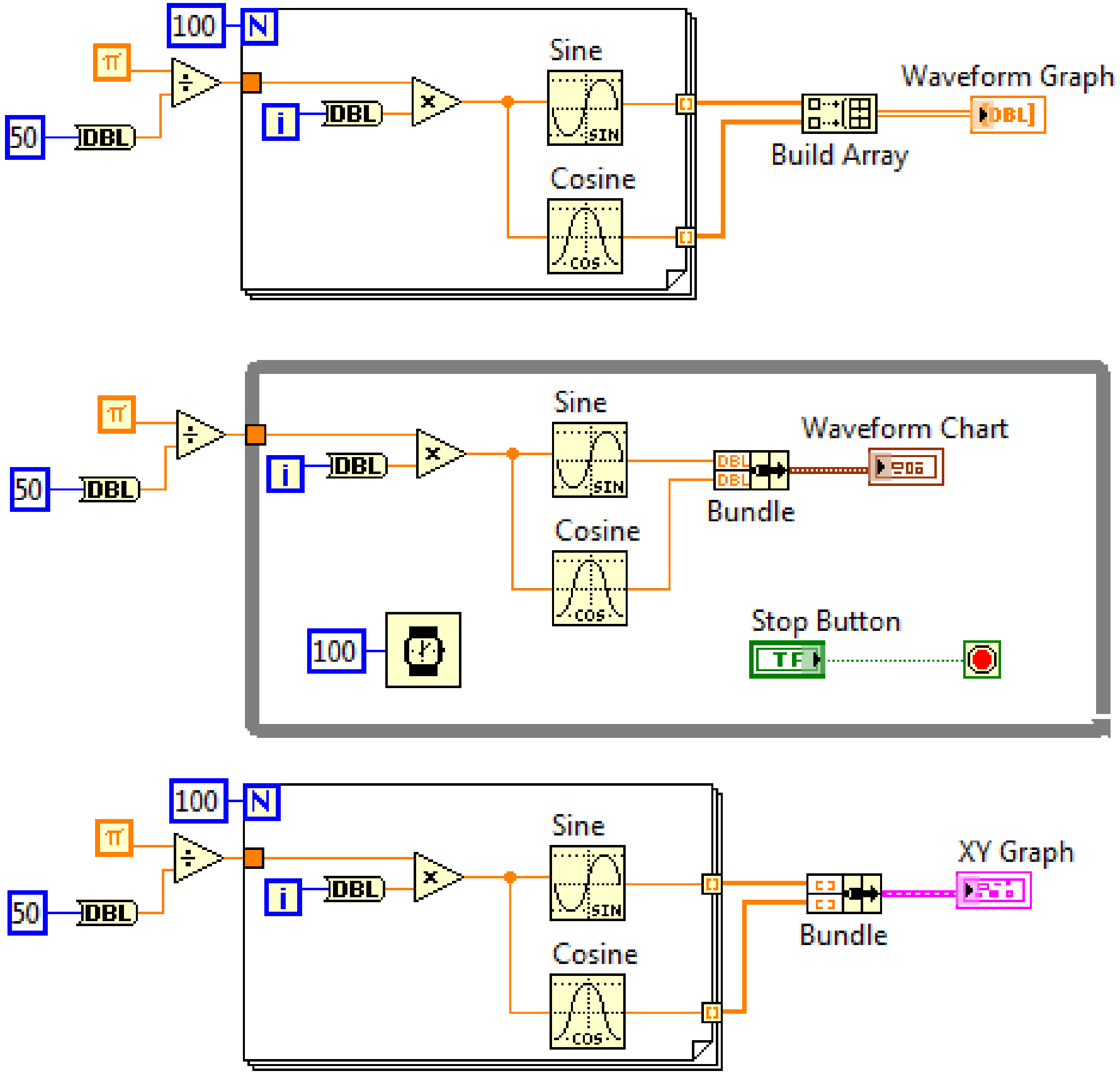
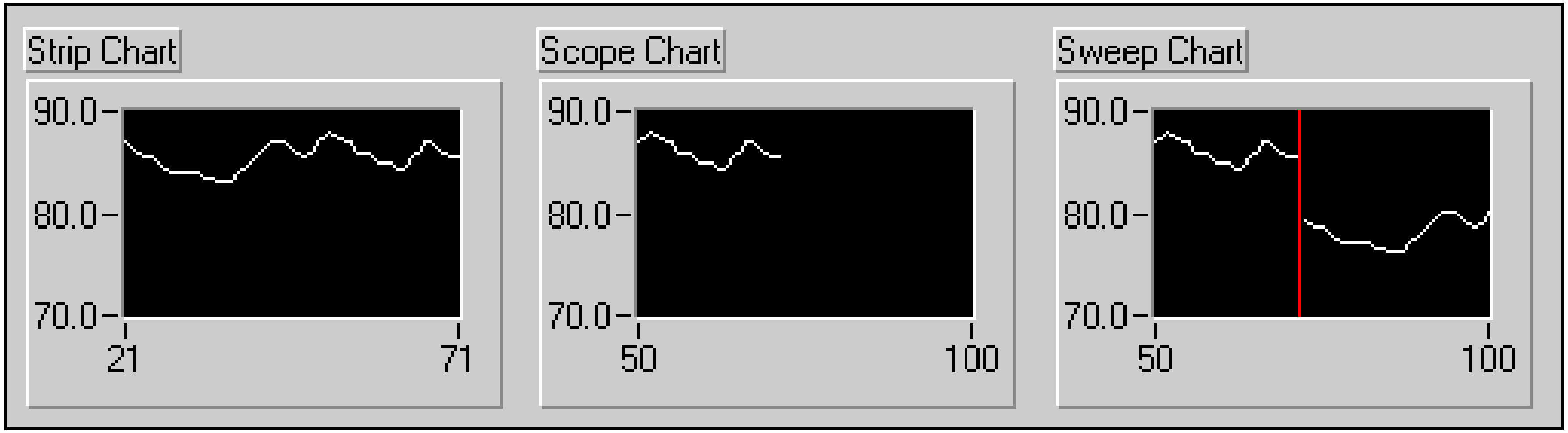


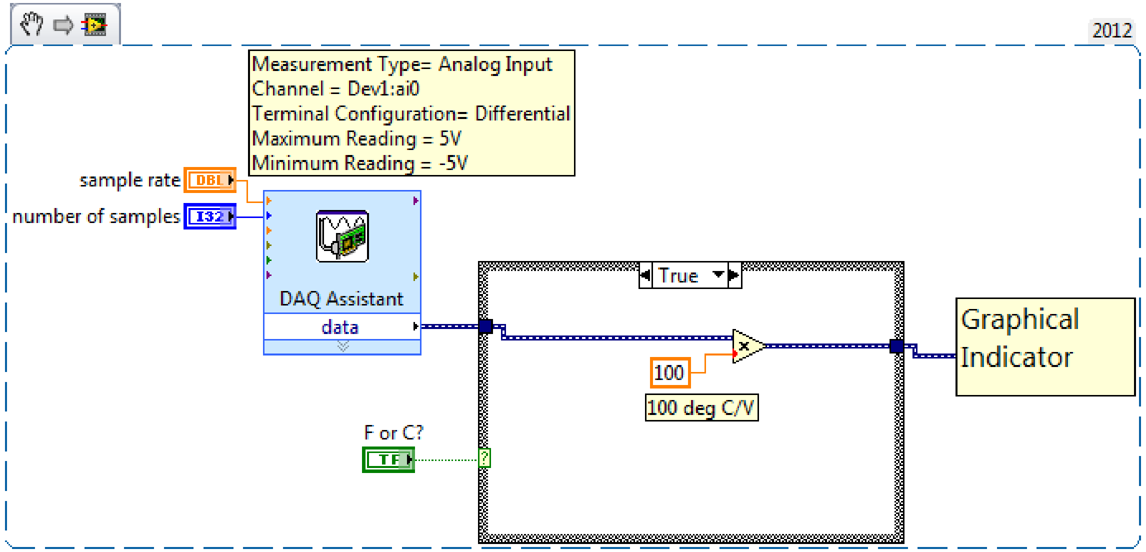
Chart Update Modes

- Right-click the chart and select Advanced»Update Mode from the shortcut menu
- Strip chart is the default update mode
- Scope chart and Sweep chart modes display plots significantly faster than the strip chart mode



Charts and Graphs

You need to add a graphical indicator to the location shown below. Which of the following is the best graphical indicator to use in this program?

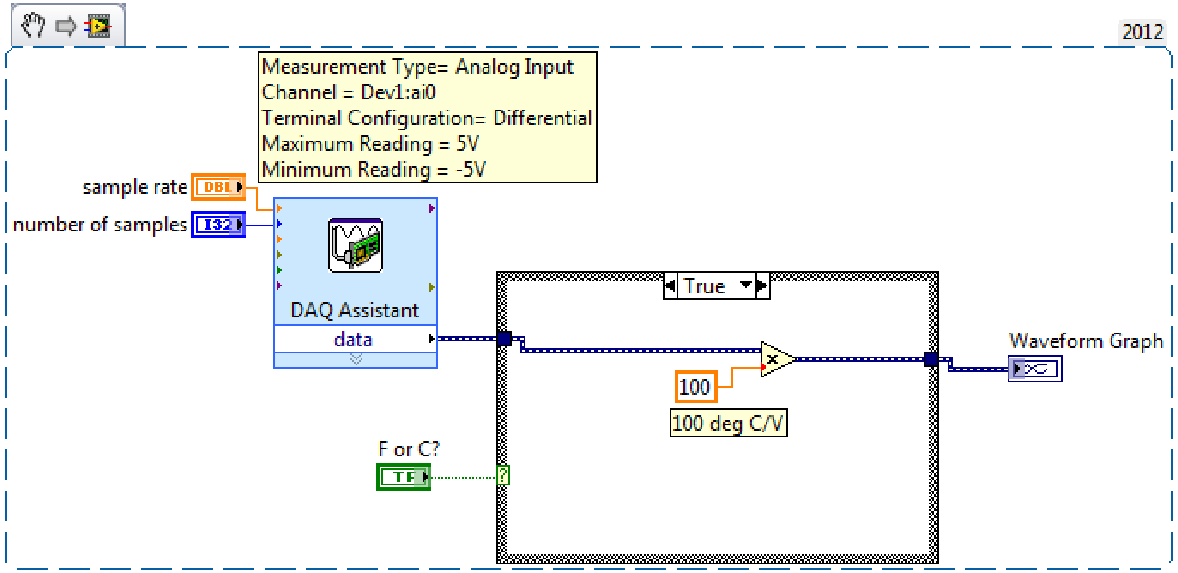


Measurement Type= Analog Input
Channel = Dev1:ai0
Terminal Configuration= Differential
Maximum Reading = 5V
Minimum Reading = -5V

A. Waveform Graph
B. Waveform Chart
C. Intensity Chart
D. XY Graph

Charts and Graphs

You need to add a graphical indicator to the location shown below. Which of the following is the best graphical indicator to use in this program?



Measurement Type= Analog Input
Channel = Dev1:ai0
Terminal Configuration= Differential
Maximum Reading = 5V
Minimum Reading = -5V

A. **Waveform Graph**
B. Waveform Chart
C. Intensity Chart
D. XY Graph

Waveform Graph is the only choice that displays multiple points at once, and can display waveforms.

Charts and Graphs

Which of the following allows you to plot any set of points, evenly distributed or not?

a. Waveform Graph
b. Waveform Chart
c. XY Graph
d. Both A. and C.

Charts and Graphs

Which of the following allows you to plot any set of points, evenly distributed or not?

~~a. Waveform Graph~~

~~b. Waveform Chart~~

c. XY Graph

d. Both A. and C.

Waveform Graph plots only single-valued functions with points **evenly distributed** along the x-axis

Waveform Chart displays one or more plots of data typically **acquired at a constant rate.**

Charts and Graphs

Which of the following allows you to plot any set of points, evenly distributed or not?

~~a. Waveform Graph~~

~~b. Waveform Chart~~

c. XY Graph

d. Both A. and C.

Waveform Graph plots only single-valued functions with points **evenly distributed** along the x-axis

Waveform Chart displays one or more plots of data typically **acquired at a constant rate.**

The XY graph is a general-purpose, Cartesian graphing object that plots multivalued functions, such as circular shapes or **waveforms with a varying time base**. The XY graph displays any set of points, evenly sampled or not.

Charts and Graphs

Which of the following will allow you to have multiple plots on a Waveform Graph?

a. Bundle two 1D arrays of X and Y data together for each plot. Then build an array of these clusters and wire it to the Waveform Graph indicator.

b. Build an n-dimensional array of data with each plot in a separate row (or column) in the array, then wire the array to the Waveform Graph indicator.

c. Bundle the elements of each waveform into a cluster and build an array of these clusters, then wire the array to the Waveform Graph indicator.

d. Both B. and C.

Charts and Graphs

Which of the following will allow you to have multiple plots on a Waveform Graph?

a. Bundle two 1D arrays of X and Y data together for each plot. Then build an array of these clusters and wire it to the Waveform Graph indicator.

b. **Build an n-dimensional array of data with each plot in a separate row (or column) in the array, then wire the array to the Waveform Graph indicator.**

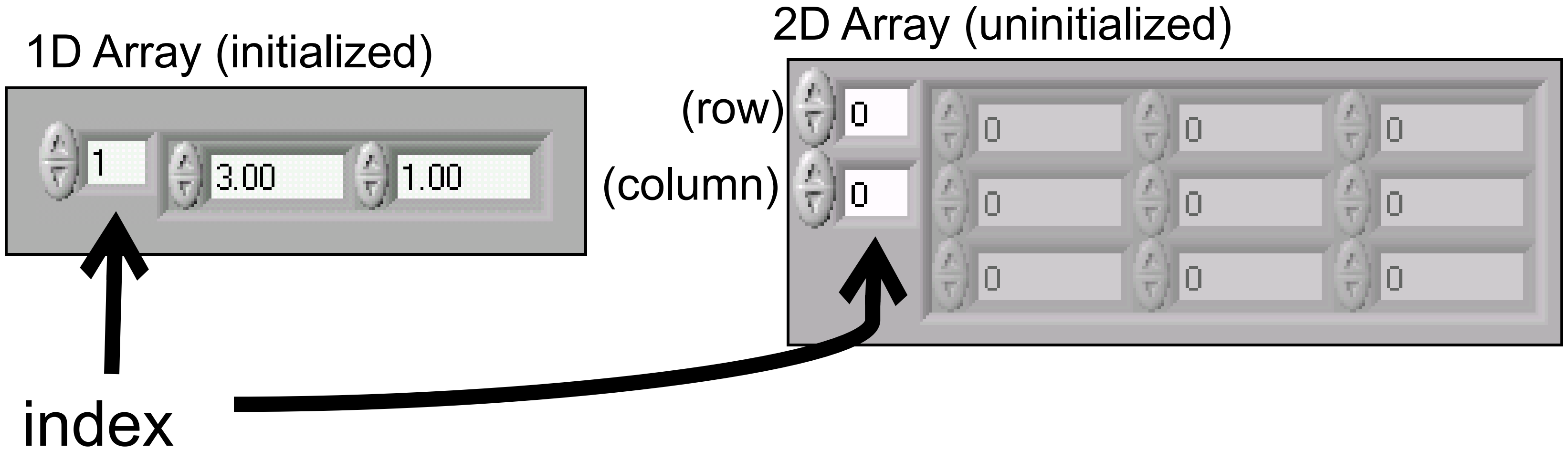
c. **Bundle the elements of each waveform into a cluster and build an array of these clusters, then wire the array to the Waveform Graph indicator.**

d. **Both B. and C.**

Arrays: the index

The index (zero-based) tells you :

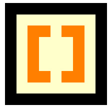
- the dimension of the array (1D, 2D, 3D, etc.)
- the index of the element displayed in the upper left corner
  - - the 1D array below is displaying index 1 to contain a value of 3.00; we do not know the value of index 0 from this image because the value at index 0 is hidden from view



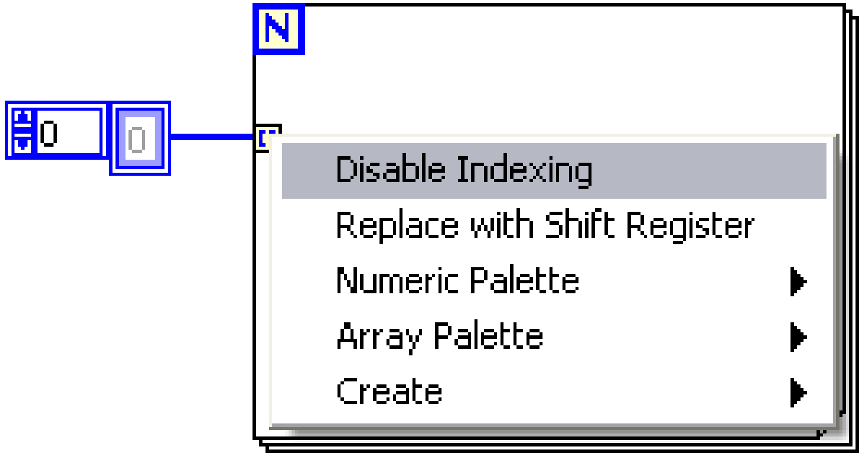
TIP: drag the edge of the index to add another dimension to an array

Arrays : Auto-indexing

- If you wire an array to or from a For Loop or While Loop, you can link each iteration of the loop to an element in that array by enabling auto-indexing on tunnel
- The tunnel changes from a solid square to the image shown below to indicate auto-indexing
- You can disable auto-indexing by right clicking on the tunnel and choosing the **Disable Indexing** option.

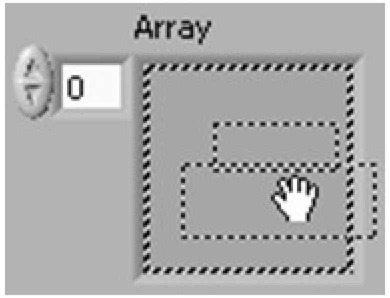


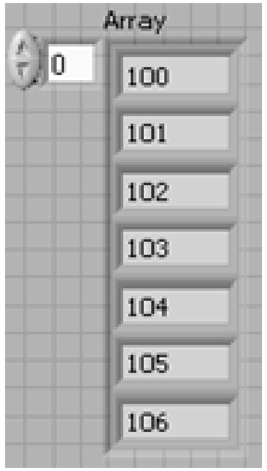
Auto-indexing tunnel

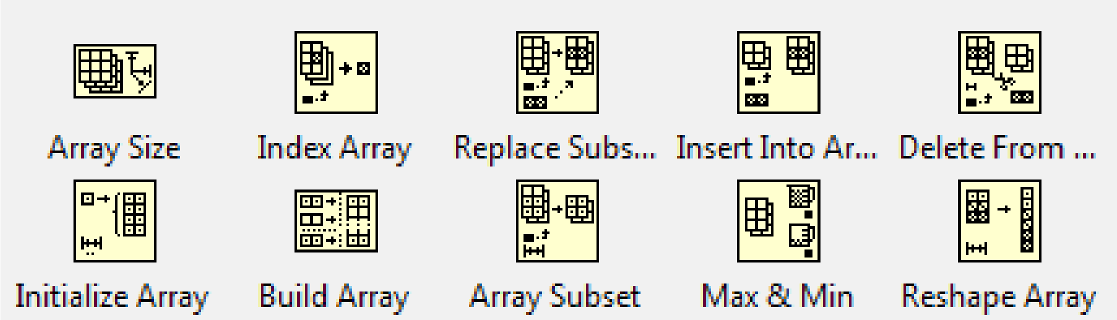


Array Functions

- 1 type of data per array- **cannot have an array of arrays.**
- Up to (2^{31}-1) elements per dimension
- Auto-indexing For Loops link each iteration with an element of the array
- For Data Acquisition:
  - Rows: Channels
  - Columns: Data

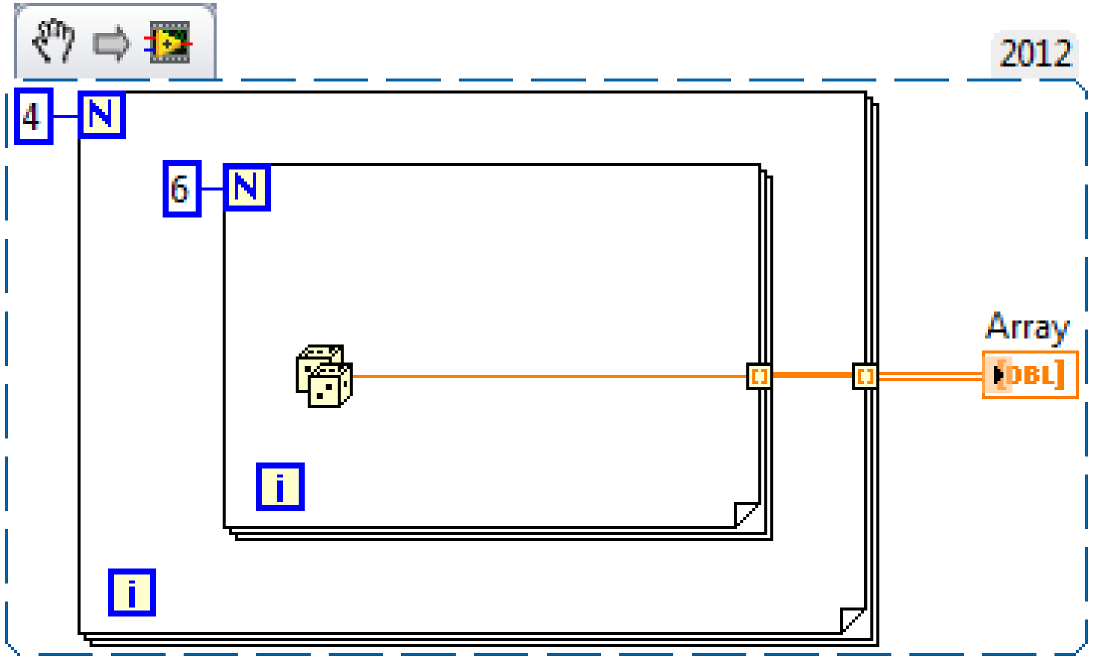






Array Functions

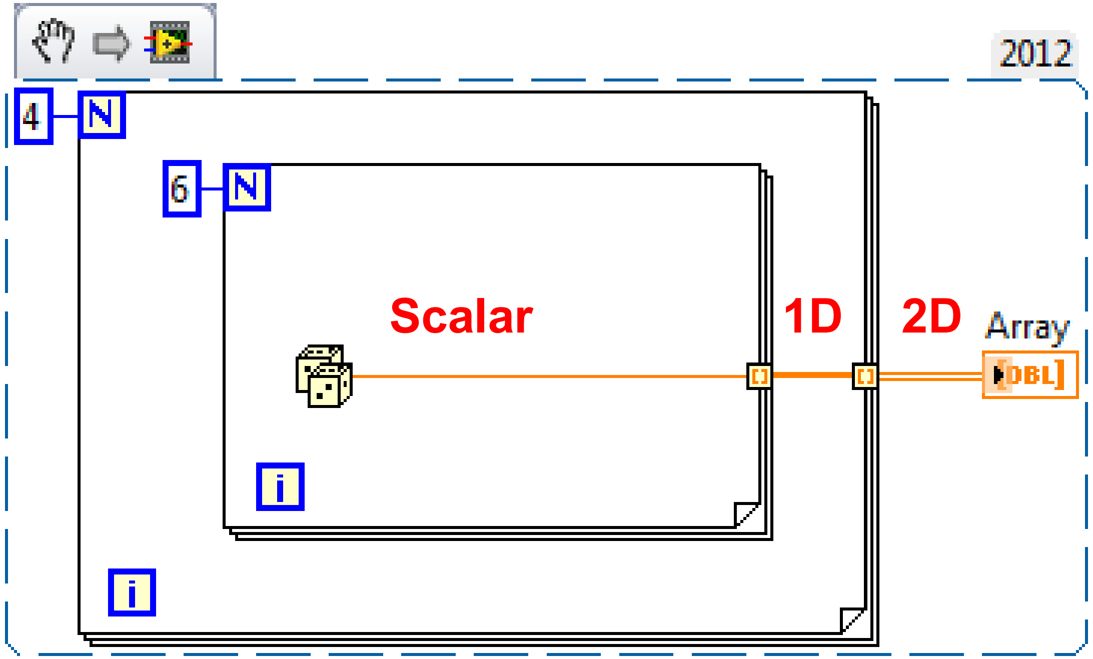
Describe the array that results from running this code.



A. A 1D Array with 10 rows
B. A 2D Array with 4 rows and 6 columns
C. A 2D Array with 6 rows and 4 columns
D. A 1D Array with 10 columns

Array Functions

Describe the array that results from running this code.



A. ~~A 1D Array with 10 rows~~

B. A 2D Array with 4 rows and 6 columns

C. A 2D Array with 6 rows and 4 columns

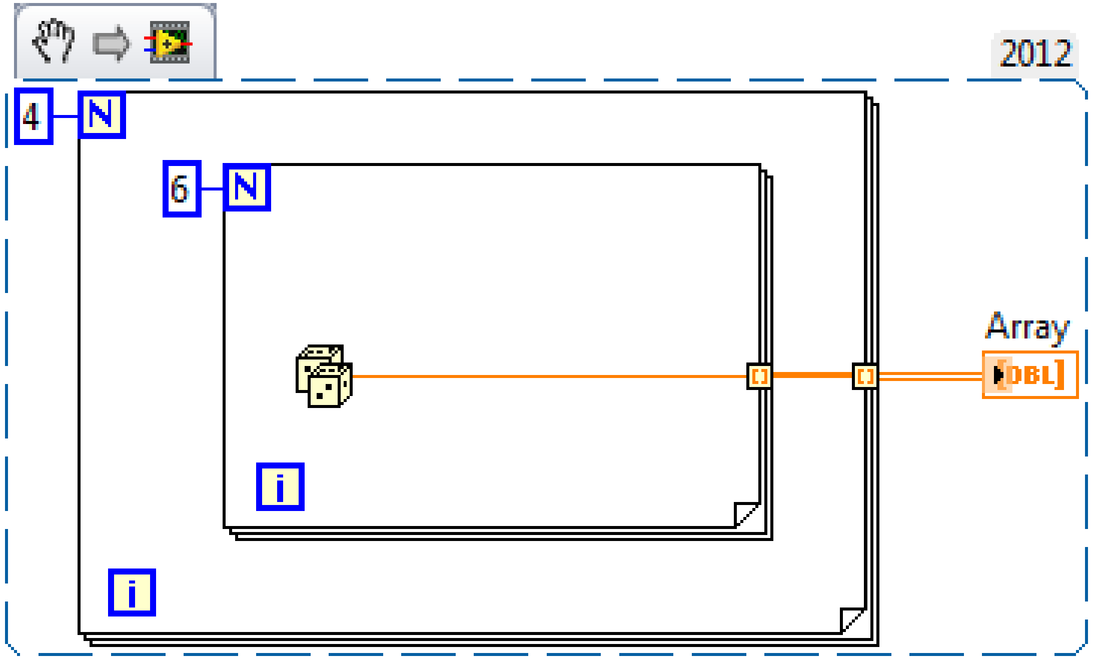
D. ~~A 1D Array with 10 columns~~

Eliminate these choices because:

Two loops means a 2D array.

Array Functions

Describe the array that results from running this code.



A. ~~A 1D Array with 10 rows~~

B. A 2D Array with 4 rows and 6 columns

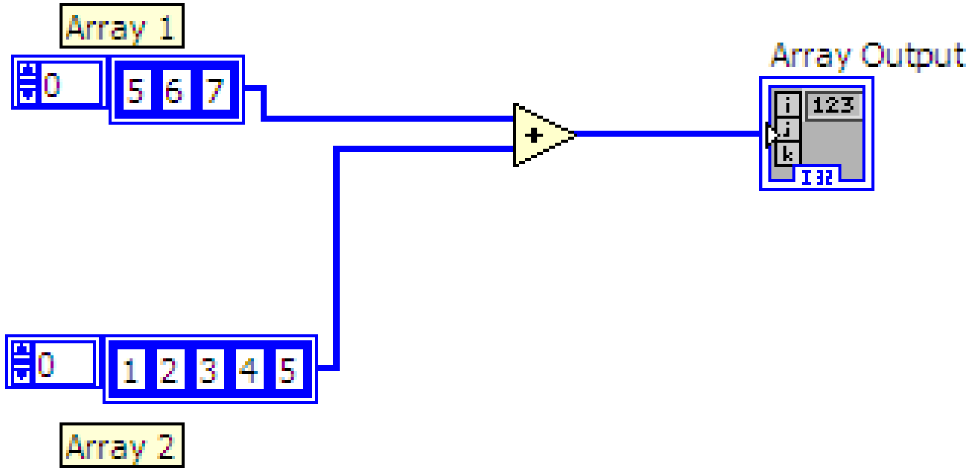
C. A 2D Array with 6 rows and 4 columns

D. ~~A 1D Array with 10 columns~~

When building arrays in nested For loops, the outside loop will always correspond to the rows of the array.

Array Functions

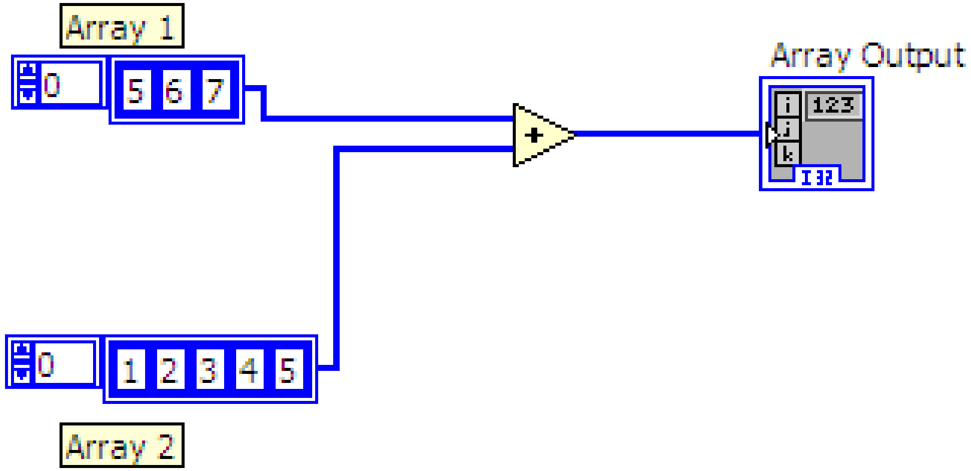
What is the result of the following Array addition?



A. A 1- D array of {6, 8, 10}
B. A 1-D array of {6, 8, 10, 4, 5}
C. A 2-D array of {{5, 6, 7}, {1, 2, 3, 4, 5}}
D. A 2-D array of {{6, 8, 10}, {4, 5}}

Array Functions

What is the result of the following Array addition?



A. A 1- D array of {6, 8, 10}

B. A 1-D array of {6, 8, 10, 4, 5}

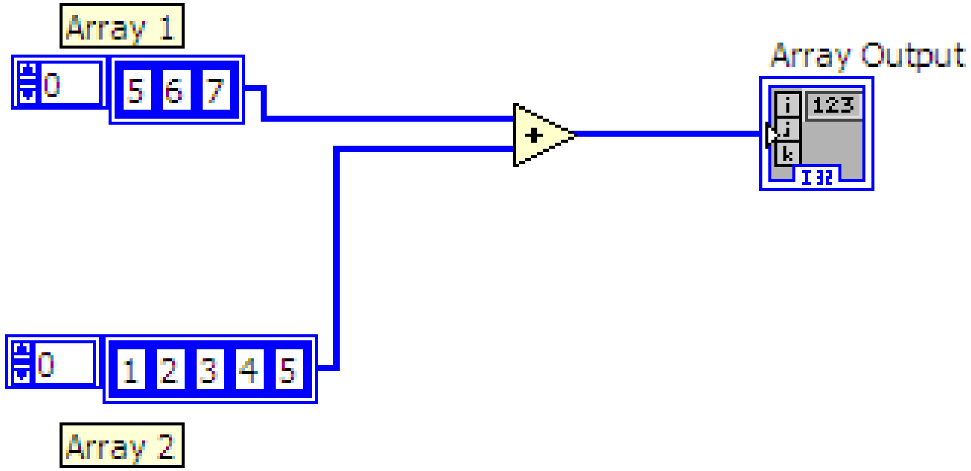
~~C. A 2-D array of {{5, 6, 7}, {1, 2, 3, 4, 5}}~~

~~D. A 2-D array of {{6, 8, 10}, {4, 5}}~~

These are not valid arrays- the row sizes are not the same

Array Functions

What is the result of the following Array addition?



When performing array arithmetic, the output array is the same size as the smaller input array.

A. A 1- D array of {6, 8, 10}

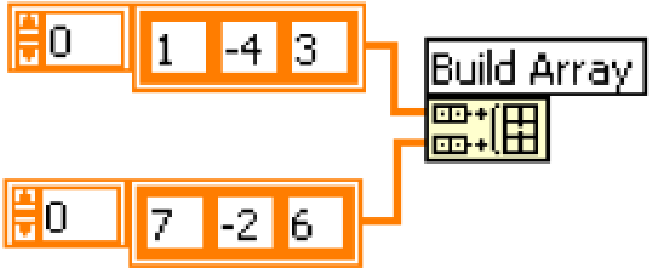
B. A 1-D array of {6, 8, 10, 4, 5}

~~C. A 2-D array of {{5, 6, 7}, {1, 2, 3, 4, 5}}~~

~~D. A 2-D array of {{6, 8, 10}, {4, 5}}~~

Array Functions

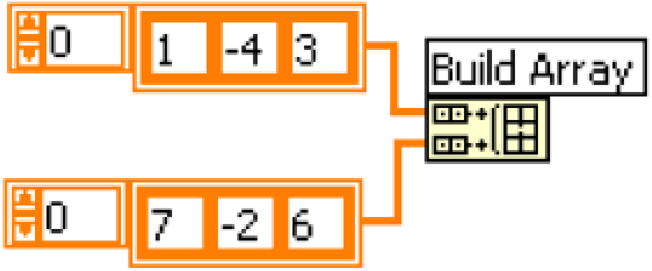
What is the output of the Build Array function in the following block diagram when the *Concatenate Inputs* is selected?



A. 1- D array of {1, -4, 3, 7, -2, 6}
B. 1-D array of {1, 7, -4, -2, 3, 6}
C. 2-D array of {{1, -4, 3, 0}, {7, -2, 6}}
D. 2-D array of {{1, -4, 3}, {7, -2, 6}}

Array Functions

What is the output of the Build Array function in the following block diagram when the *Concatenate Inputs* is selected?

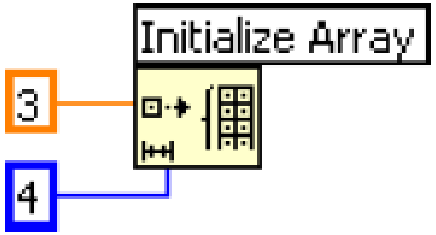


A. **1- D array of {1, -4, 3, 7, -2, 6}**
B. 1-D array of {1, 7, -4, -2, 3, 6}
C. 2-D array of {{1, -4, 3, 0}, {7, -2, 6}}
D. 2-D array of {{1, -4, 3}, {7, -2, 6}}

Build Array function concatenates multiple arrays or appends elements to an n-dimensional array. With **Concatenate Inputs** selected, it appends all inputs in order, forming an output array of the same dimensionality as the highest-dimension array input wired.

Array Functions

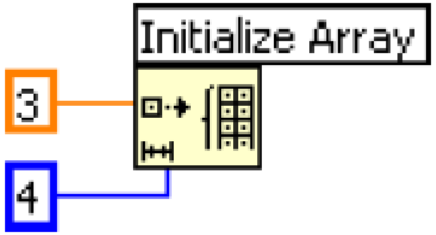
What is the output of the *Initialize Array* function after the following code has executed?



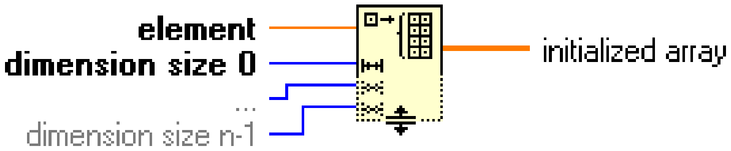
A. 1- D array of {3, 3, 3, 3}
B. 1-D array of {4, 4, 4}
C. 2-D array of {3, 4}
D. 2-D array of {4, 3}

Array Functions

What is the output of the *Initialize Array* function after the following code has executed?



A. 1- D array of {3, 3, 3, 3}
B. 1-D array of {4, 4, 4}
C. 2-D array of {3, 4}
D. 2-D array of {4, 3}

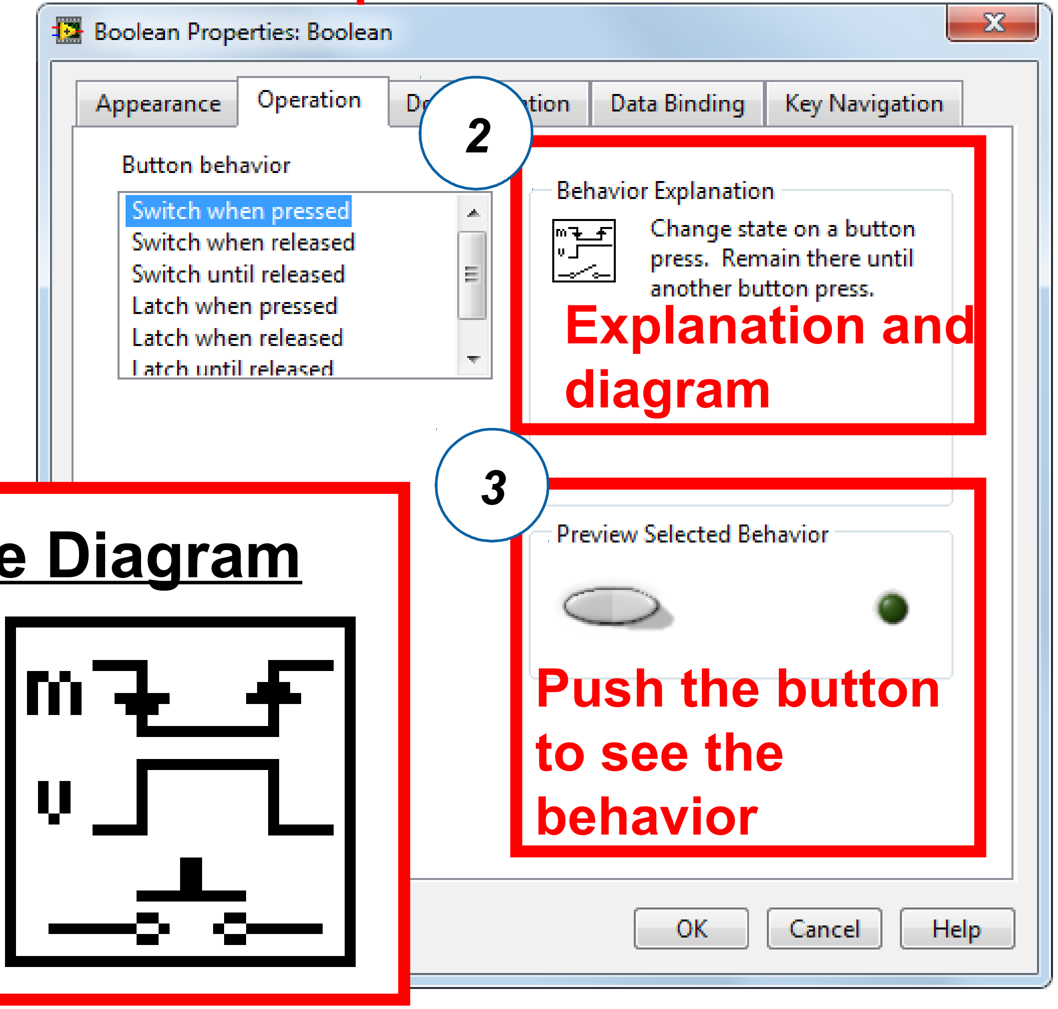


Inivitalze Array function creates an n-dimensional array (4) in which every element is initialized to the value of **element (3)**.

Mechanical Action of Booleans

- Behavior of Boolean controls is specified by the mechanical action.
- Use the Properties dialog to investigate the different mechanical actions.

1 **Place Boolean on Front Panel, Open the Boolean Properties**



Button behavior

- Switch when pressed
- Switch when released
- Switch until released
- Latch when pressed
- Latch when released
- Latch until released

Behavior Explanation: Change state on a button press. Remain there until another button press.

2 **Explanation and diagram**

3 **Push the button to see the behavior**

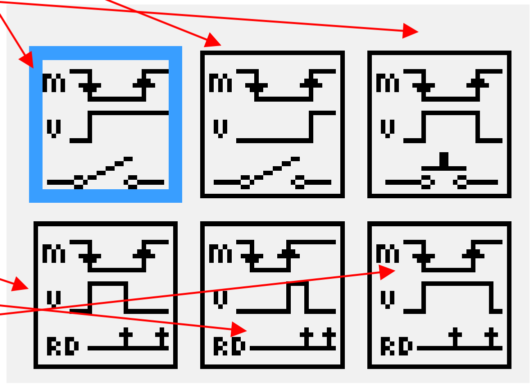
How to Read the Diagram

- •Button Position →
- •“Voltage” at LED →
- •Circuit Diagram Symbol →

Mechanical Actions of Boolean Objects

- Switch when pressed (light switch)
- Switch when released (mouse)
- Switch until released (door buzzer)

- Latch when pressed (circuit breaker)
- Latch when released (dialog box)
- Latch until released



Mechanical Action of Booleans

Open NI Example Finder, select the Mechanical Action of Booleans.vi. Browse by Task, Buiding User Interfaces» General» Mechanical Action of Booleans.vi.

DEMONSTRATION

Mechanical Action

Which category of mechanical action is guaranteed to be read at least once by LabVIEW?

A. Switch

B. Latch

C. Boolean

D. All mechanical actions are read at least once

Mechanical Action

Which category of mechanical action is guaranteed to be read at least once by LabVIEW?

A. Switch

B. Latch

C. Boolean

D. All mechanical actions are read at least once

A switch can change value without being read, and "Boolean" is not a mechanical action.

Mechanical Action

A button is pressed once and immediately turns on the light of a dark room. The light stays on after the button is released. What is the mechanical action of the button?

A. Switch when pressed

B. Switch when released

C. Switched until released

D. Latch when pressed

E. Latch when released

F. Latch until released

Mechanical Action

A button is pressed once and immediately turns on the light of a dark room. The light stays on after the button is released. What is the mechanical action of the button?

A. Switch when pressed

B. Switch when released

~~C. Switched until released~~

~~D. Latch when pressed~~

~~E. Latch when released~~

~~F. Latch until released~~

The light stays on after the button is released, so this is not a latching action, or switched until released, which would only turn the light on while the button is being pressed.

Mechanical Action

A button is pressed once and immediately turns on the light of a dark room. The light stays on after the button is released. What is the mechanical action of the button?

A. Switch when pressed

B. Switch when released

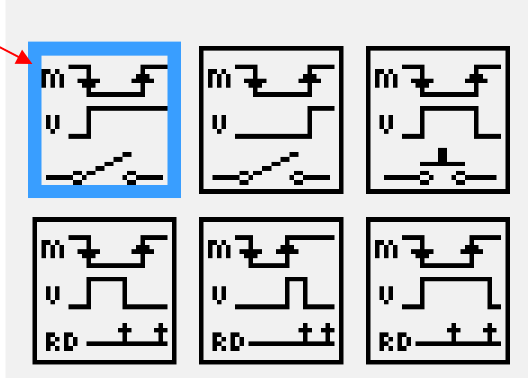
~~C. Switched until released~~

~~D. Latch when pressed~~

~~E. Latch when released~~

~~F. Latch until released~~

The key word is "immediately." The light comes on before the button is released.



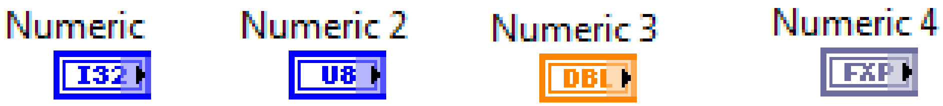
Data Representation: Numeric Range

- Data representation affects how large or small of a number can be stored.
  - A few common data types are listed here—for more types, see the LabVIEW Help

Terminal	Type	Disk Storage Size	Number of Decimal Digits	Approximate Range
	Double	64 bits	15	-1.8 x 10^{308} to 1.8 x 10^{308}
DBL	Signed 8-bit Integer	8 bits	2	-128 to 127
I8	Unsigned 8-bit Integer	8 bits	2	0 to 255
U8	Unsigned 32-bit Integer	32 bits	9	0 to 4,294,967,295
U32				

Data Representation: Storage Space

- Data representation is important when disk space or memory usage is a concern:
  - Larger types require more space (U32 requires 4x the memory of U8)
  - Signed types require more space than unsigned (I32 uses more space than U32)
  - Boolean types require one U8 of space each. Consider using a single U8 to represent eight Boolean values if several Booleans must be stored.



Data Representation: Storage Space

You need to store an integer which will only ever be between 1 and 10. Which of these would be the best choice?

A. U32

B. I8

C. DBL

D. U8

Data Representation: Storage Space

You need to store an integer which will only ever be between 1 and 10. Which of these would be the best choice?

A. U32

B. I8

C. DBL

D. U8

The U8 representation can store numbers ranging from 0 to 2^8 -1, or 0-255

Data Representation: Storage Space

Which of these allows storing decimal numbers such as 1.3 or 2.55?

A. U32

B. I8

C. DBL

D. U8

Data Representation: Storage Space

Which of these allows storing decimal numbers such as 1.3 or 2.55?

A. U32

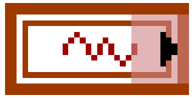
B. I8

C. DBL

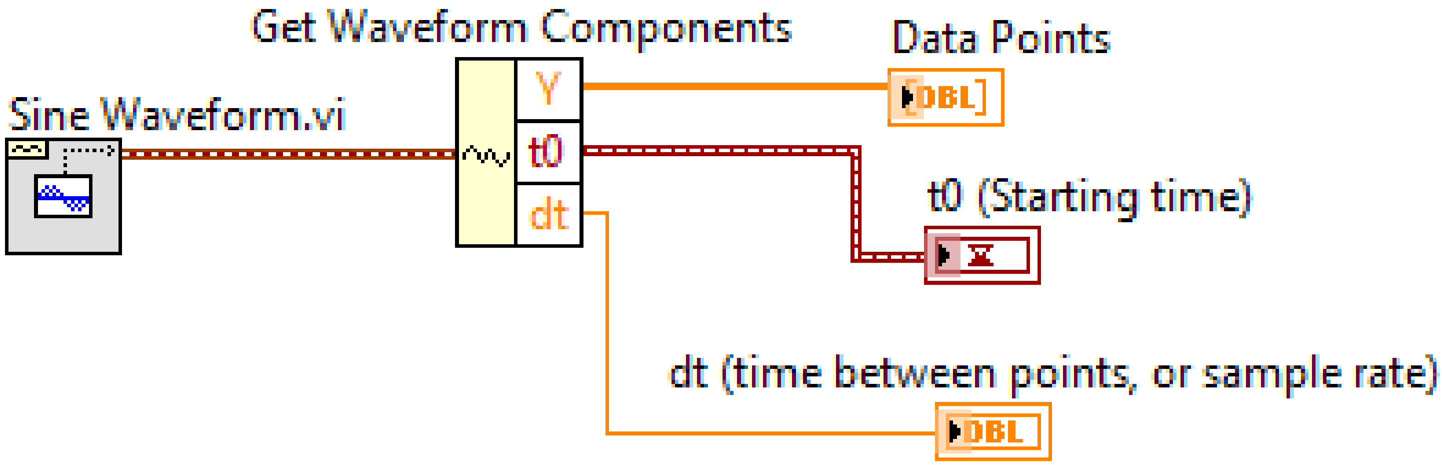
D. U8

DBL is the only data type listed that can store non-integer numbers.

Waveform Data Type

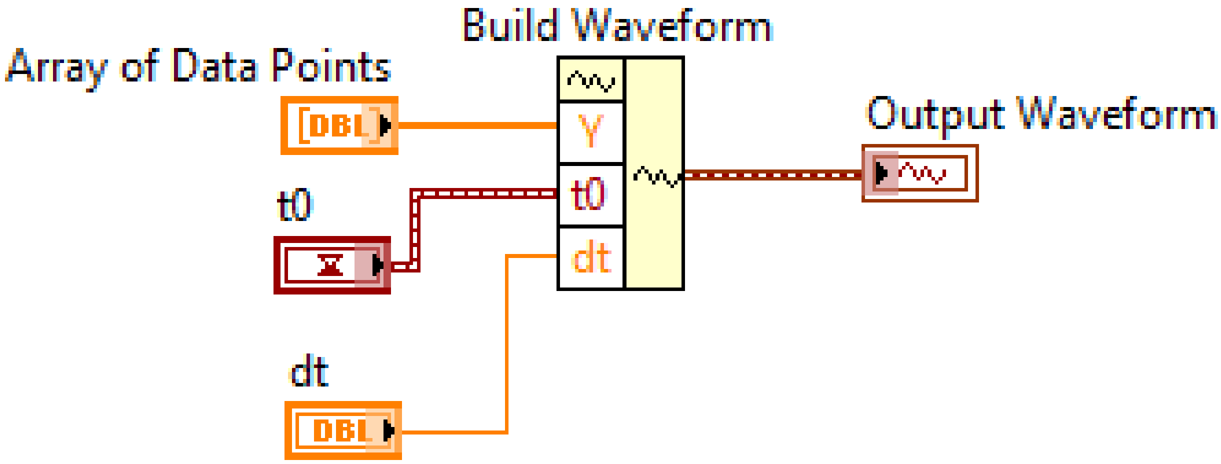


- The Waveform data type is a cluster.
  - Carries the data, start time, and dt of a waveform.
  - Get Waveform Components function is very similar to Unbundle By Name



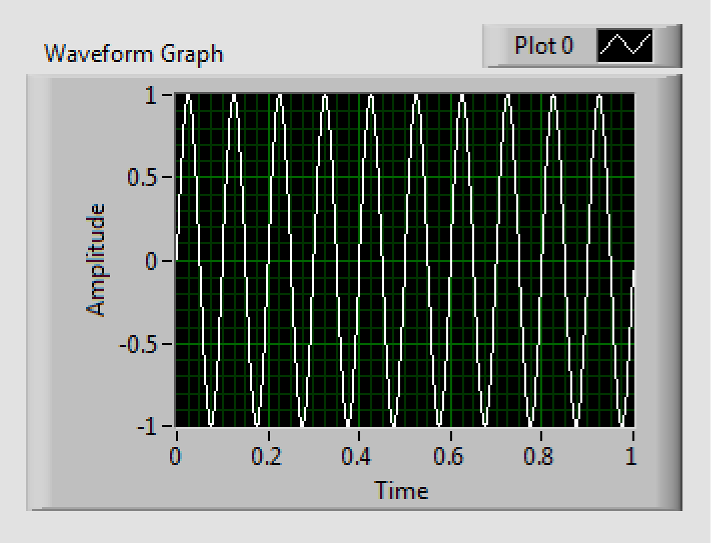
Waveform Data Type

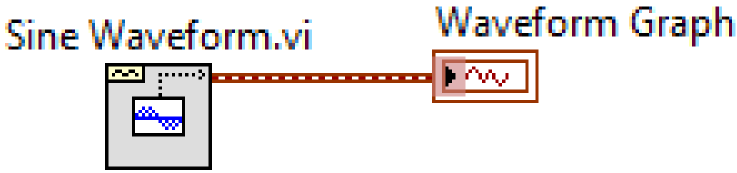
- Build Waveform function creates a waveform from its components
  - Similar to Bundle By Name



Waveform Data Type

- Some indicators can display waveform data directly.
  - Waveform Graph





Waveform Data Type

1. Which of these is the Waveform data type most similar to?
   1. String
   2. Numeric Array
   3. Cluster
   4. Timestamp

Waveform Data Type

1. Which of these is the Waveform data type most similar to?

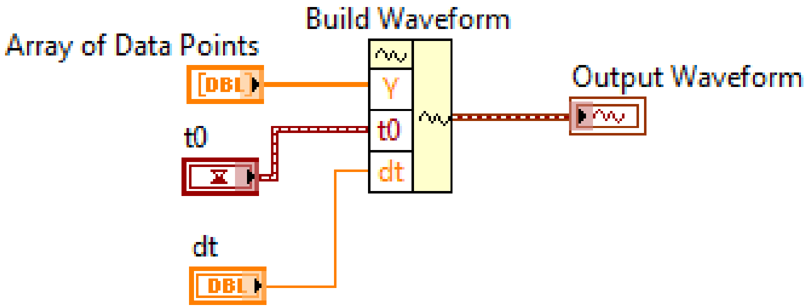
   A. String

   B. Numeric Array

   C. Cluster

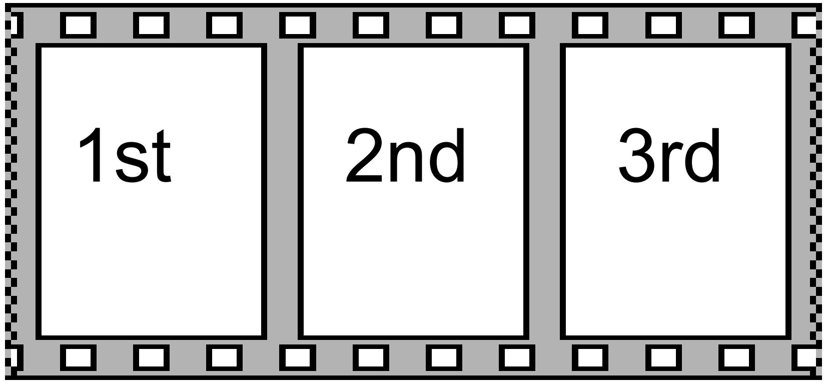
   D. Timestamp

Although a waveform contains some of th
whole is very similar to a cluster.



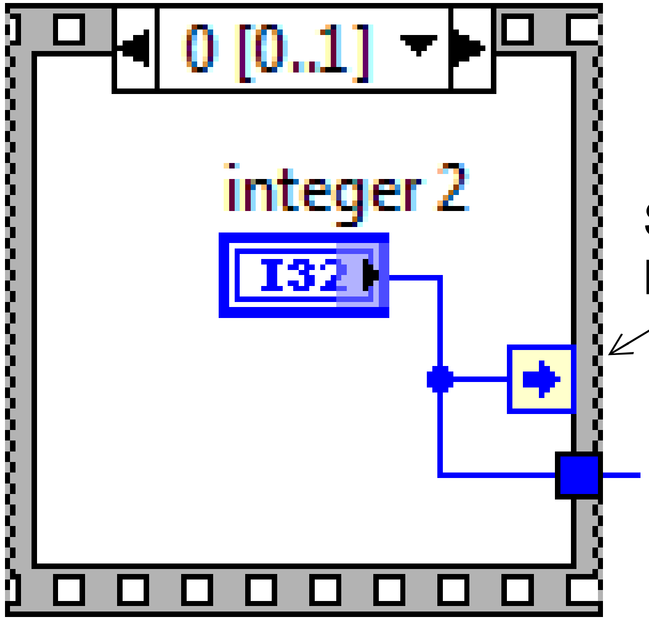
Flat Sequence Structure

- Executes each frame beginning with the left-most frame and ending with the right-most frame
- The previous frame must complete before the next frame executes
- Data can be passed out of or between frames using tunnels
- Once the sequence begins, it cannot be stopped



Stacked Sequence Structure

- Removed in LabVIEW 2014
- Executes frame 0, then frame 1, etc. until the last frame executes
    - Returns data only after the last frame executes
    - To transfer data from frame to frame, a Sequence Local must be created (right-click » Add Sequence Local)
- Once the sequence begins, it cannot be stopped



Sequence Local

Stacked Sequence Structure

What purpose do sequence locals have in a stacked sequence structure?

A. Provide reentrancy for the sequence structure

B. Transfer data between frames

C. Set values of local variables in the calling VI

D. Transfer data out of the sequence between frames

Stacked Sequence Structure

What purpose do sequence locals have in a stacked sequence structure?

A. Provide reentrancy for the sequence structure

B. Transfer data between frames

C. Set values of local variables in the calling VI

D. Transfer data out of the sequence between frames

Stacked sequence structures do not return their outputs until the final frame has executed. Use sequence locals to transfer data between subsequent frames.

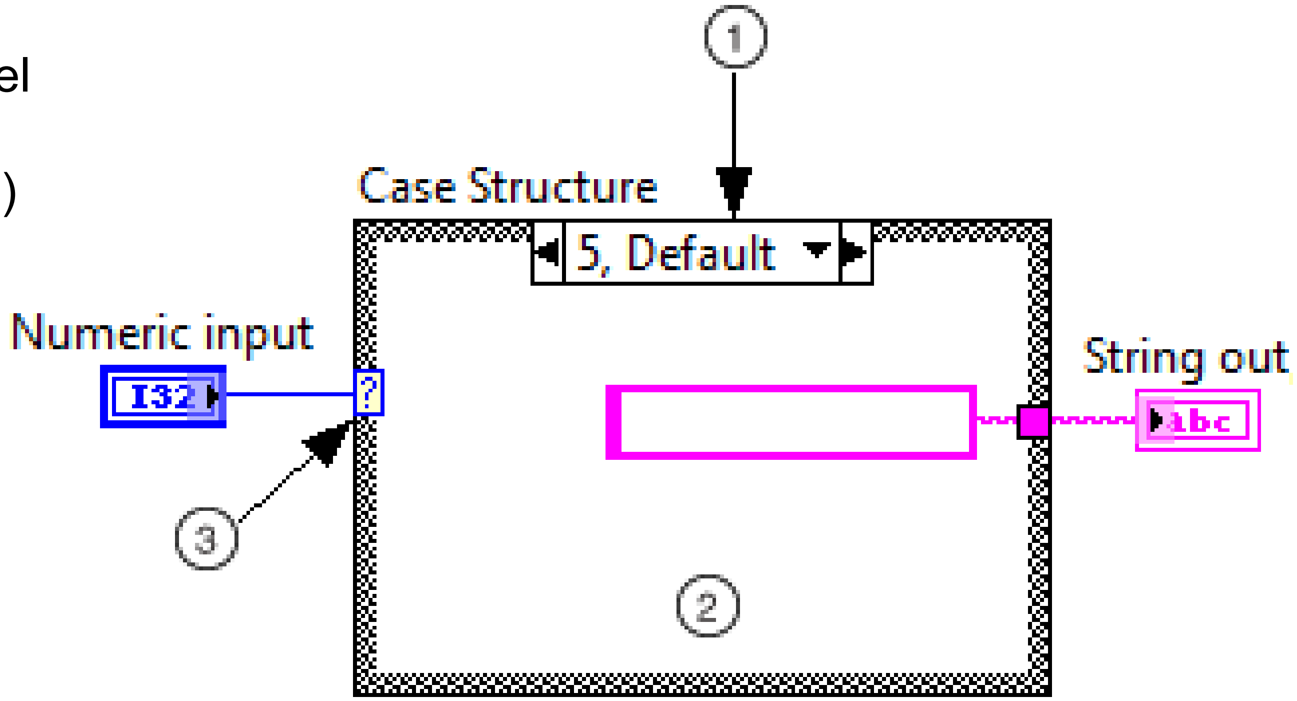
Case Structures

Identify the components of a case structure

A. Case Selector Terminal

B. Case Selector Label

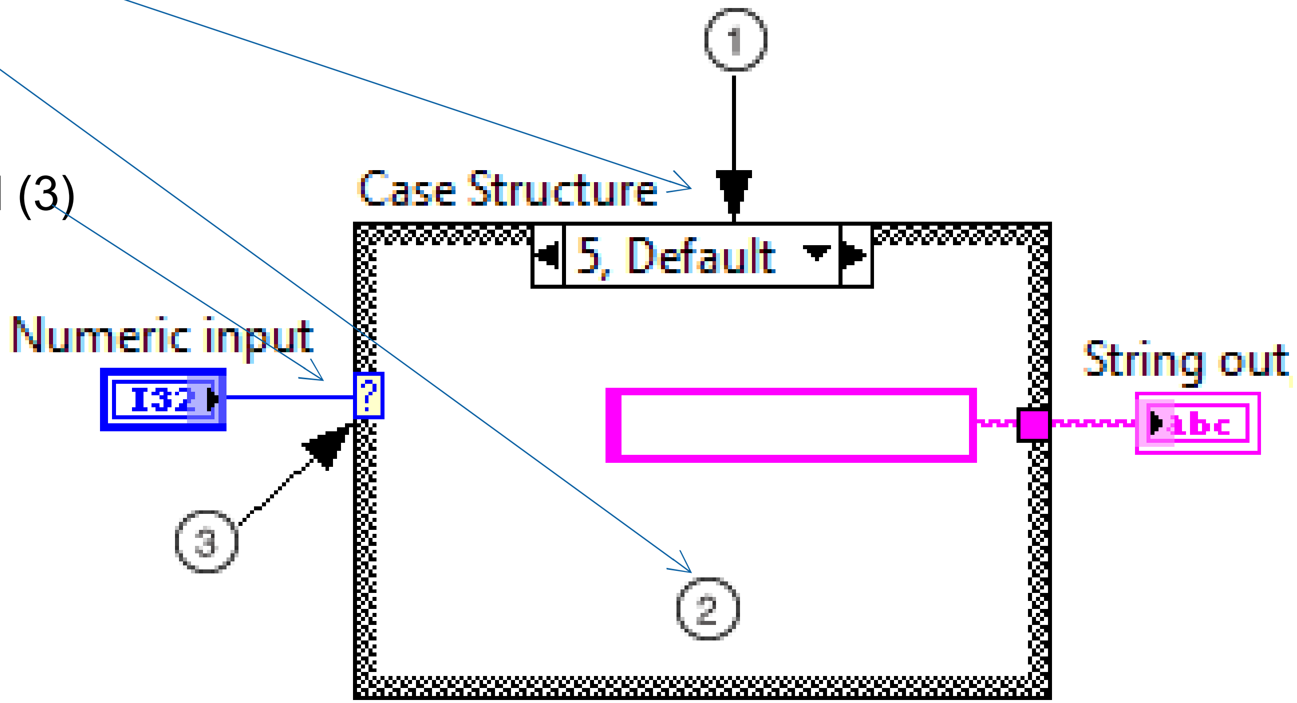
C. Subdiagram (Case)



Case Structures

Identify the components of a case structure

- Case Selector Label (1)
- Subdiagram (Case) (2)
- Case Selector Terminal (3)



Case Structures

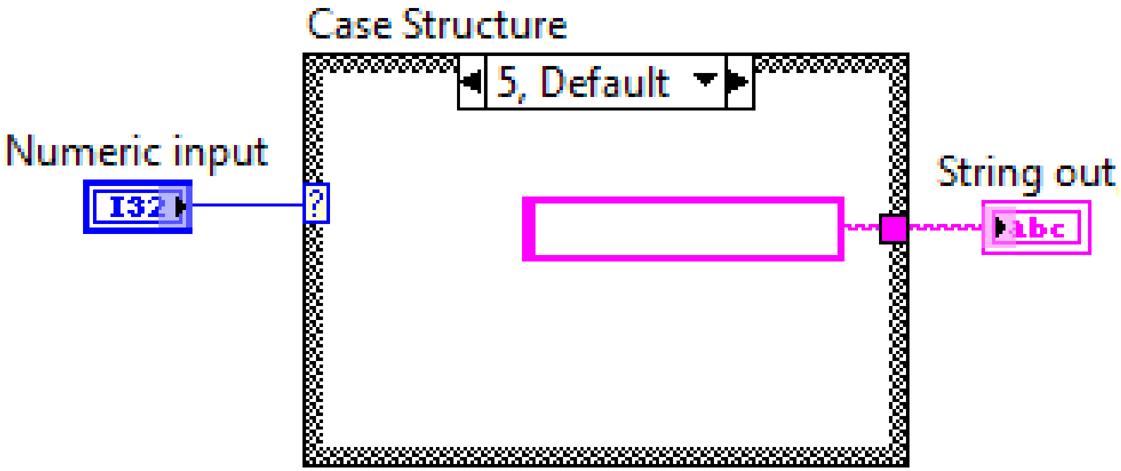
Assuming the pictured Case Structure, which of the following could result in a broken run arrow?

A. Numeric input has a value of 6 and there is no case for 6.

B. The empty string inside the case is deleted and the resulting broken wire inside the case is removed.

C. The wire between numeric input and the case selector is removed

D. A new input is wired into the case selector, but is left unused in one of the cases



Case Structures

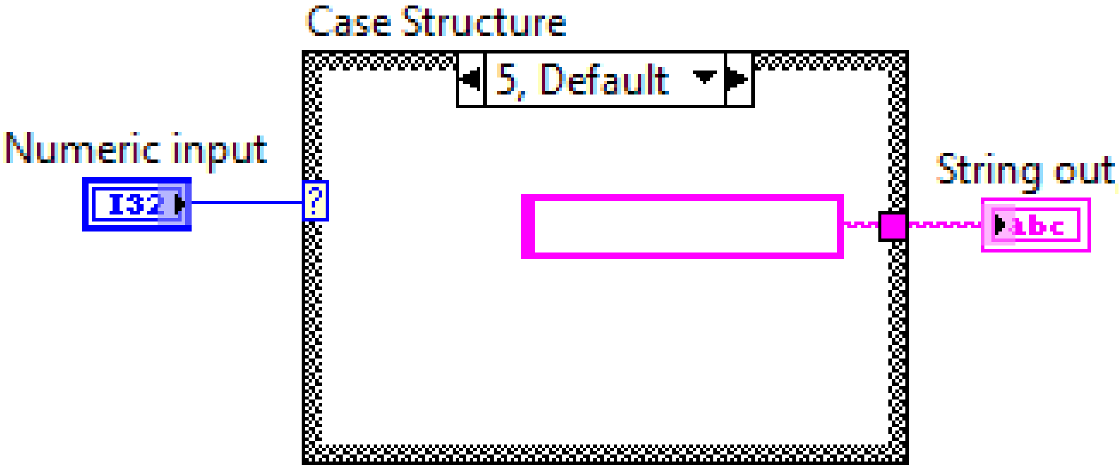
Assuming the pictured Case Structure, which of the following could result in a broken run arrow?

A. Numeric input has a value of 6 and there is no case for 6.

B. The empty string inside the case is deleted and the resulting broken wire inside the case is removed.

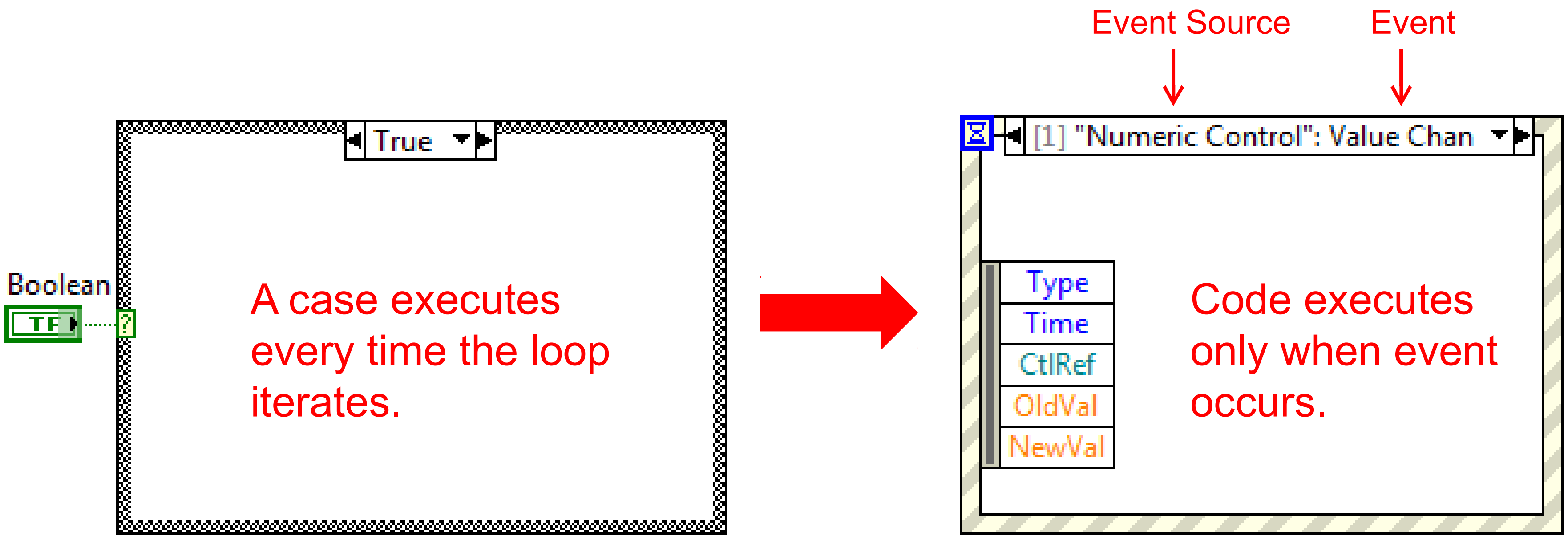
C. The wire between numeric input and the case selector is removed

D. A new input is wired into the case selector, but is left unused in one of the cases



Event Structures

An Event structure works like a Case structure with a built-in Wait on Notification function.



Event Source

Event

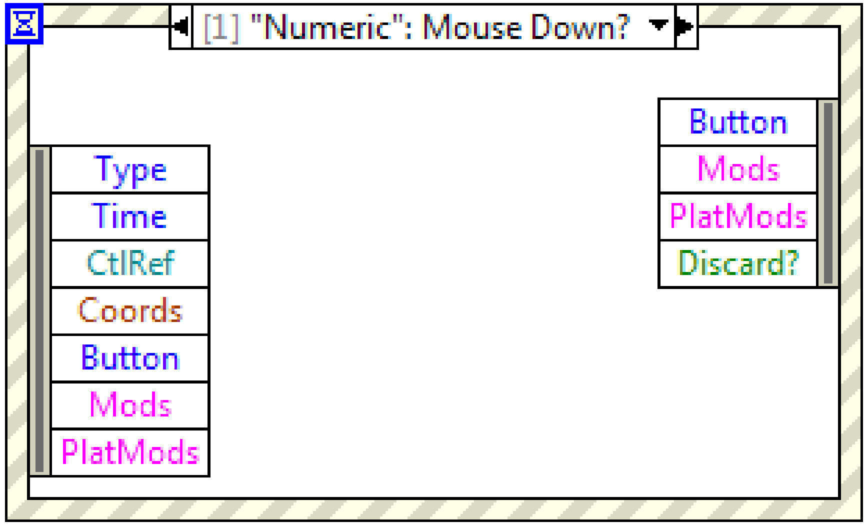
A case executes every time the loop iterates.

Code executes only when event occurs.

Event Structure

- Event Structure — LabVIEW's programmatic tool for handling events.
- Specialized type of Case Structure
- Execution of code can be dependent on whether or not an event has occurred

- Waits for an event to occur indefinitely, unless configured to timeout.



Event Structures

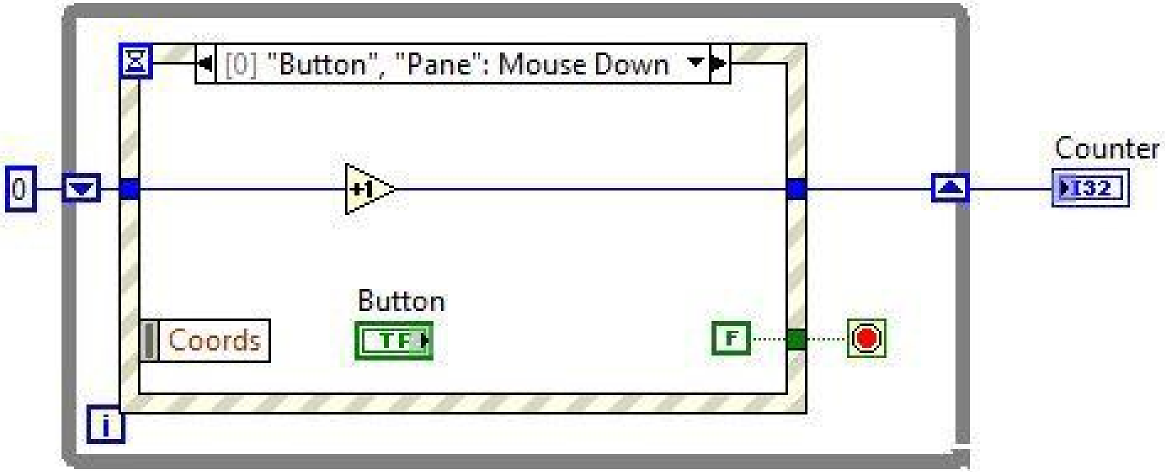
When the user clicks the **Button** control, how many times is the Increment function called?

a. 0

b. 1

c. 2

d. 3



Event Structures

When the user clicks the **Button** control, how many times is the Increment function called?

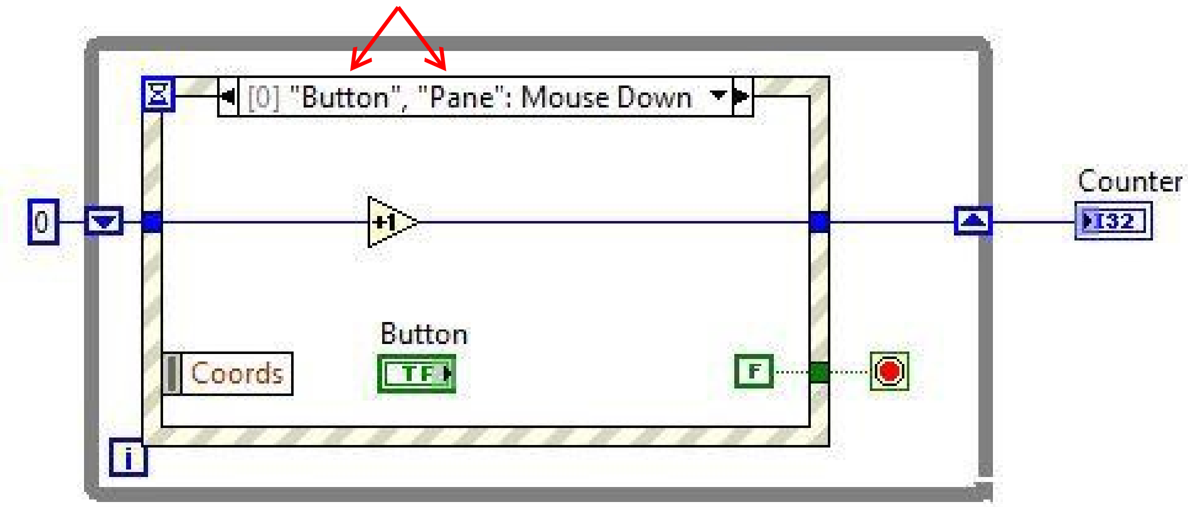
a. 0

b. 1

c. 2

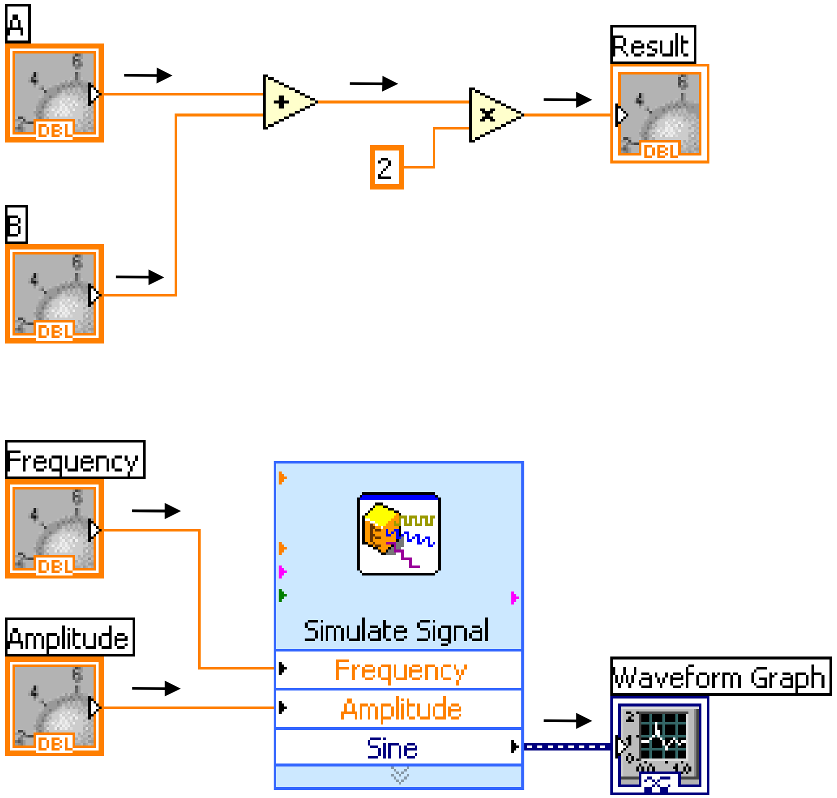
d. 3

Because a Mouse Down event occurs on both the Button and the Pane, 2 events are registered. The code executes twice.



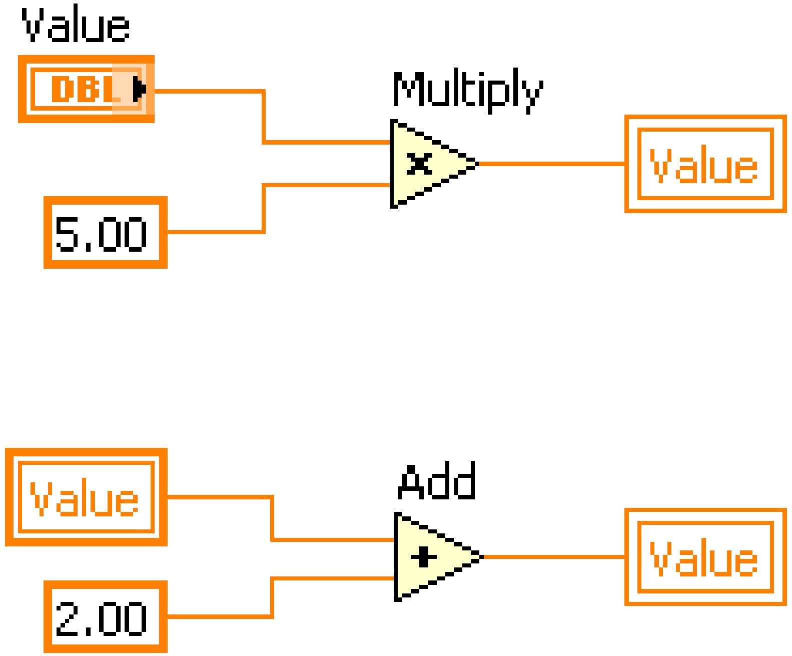
Data Flow

- Block diagram execution is dependent on the flow of data
- Block diagram does NOT execute left to right
- Node executes when data is available to ALL input terminals
- Nodes supply data to all output terminals when done
- If the computer running this code had multiple processors, these two pieces of code could run independently without additional coding



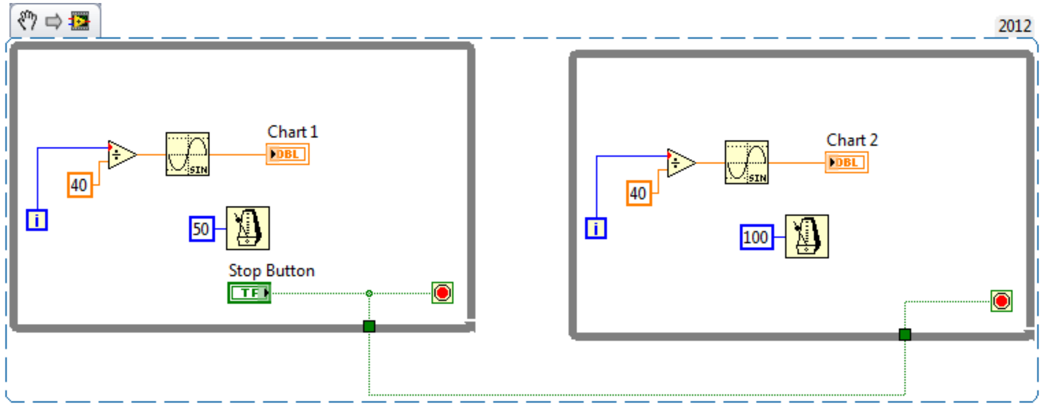
Race Conditions: Sequencing

- What is the final value? Four possible outcomes:
  - Value = (Value * 5) +2
  - Value = (Value + 2) * 5
  - Value = Value * 5
  - Value = Value +2

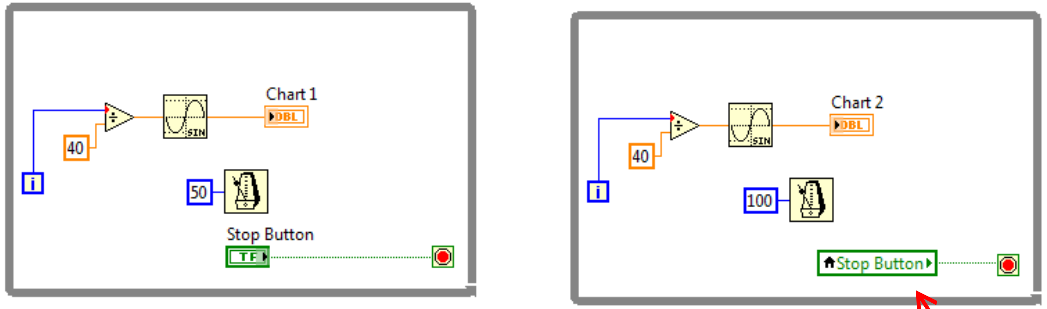


Breaking Data Flow

Situation: Run 2 Loops simultaneously with 1 Stop Button



Wiring the Stop Button from one Loop to the other will **NOT** work.

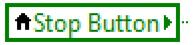
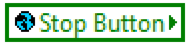
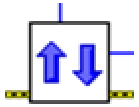


Solution: Use a Local Variable

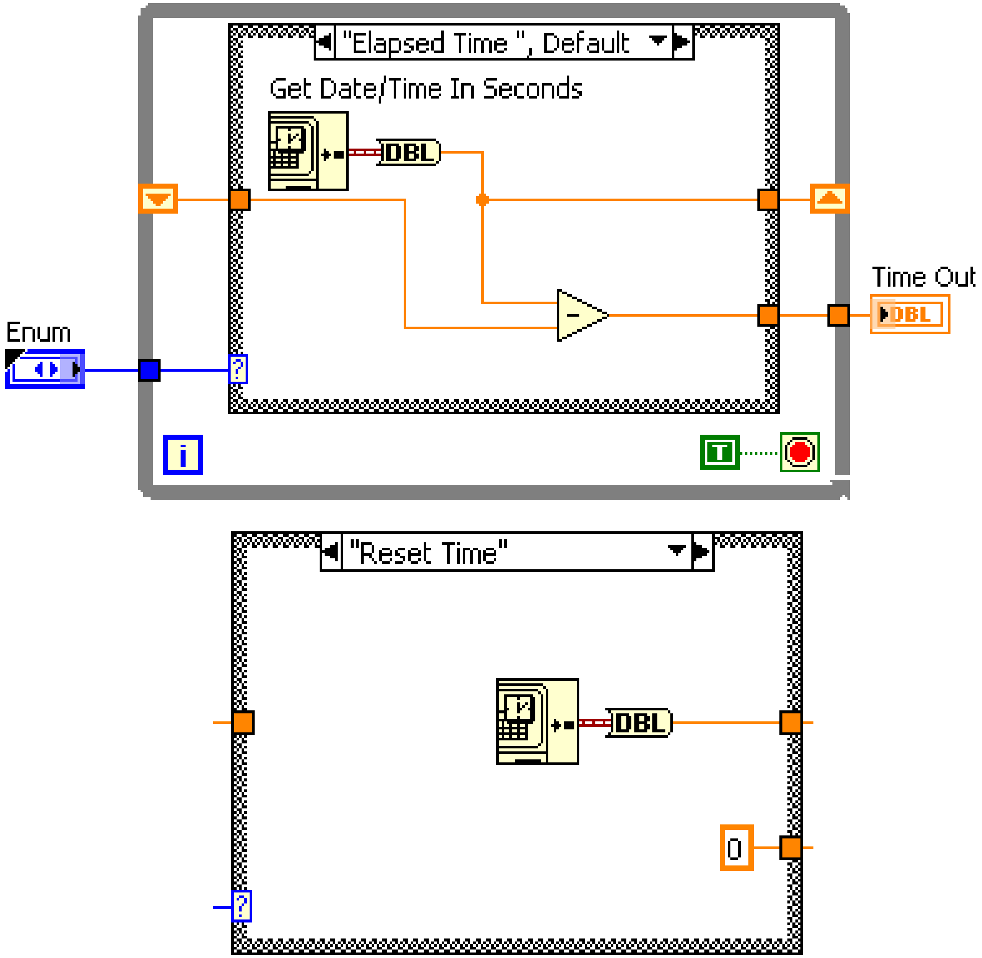
Drawbacks: Introduces a Possible Race Condition

Local Variable referencing the Stop Button

Breaking Data Flow

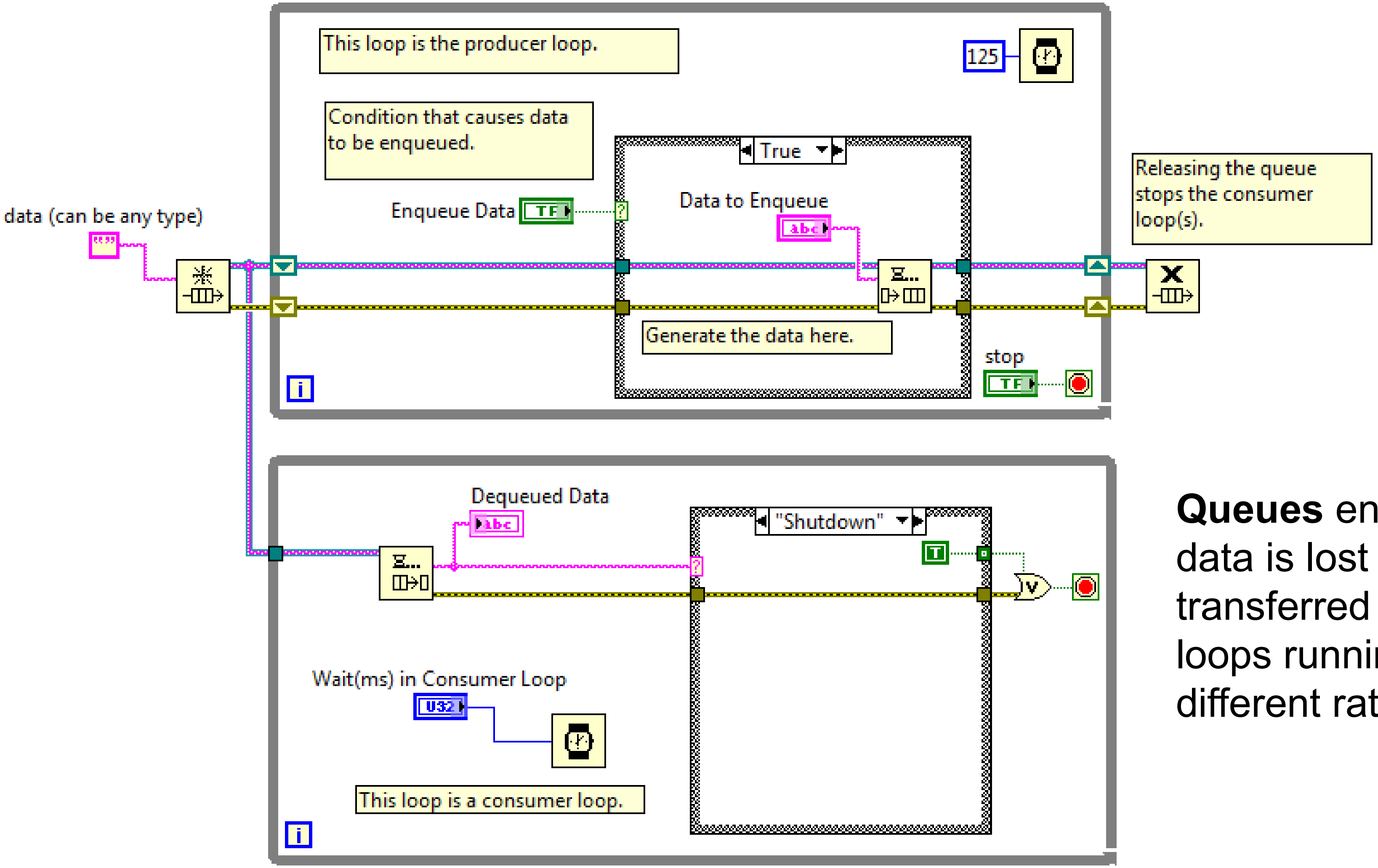
Name	Appearance	Function	Drawbacks
Wire		Connects data at different terminals	Must follow Data Flow
Local Variable		Allows a value to be accessed anywhere in a single VI	Introduces the possibility of a race condition
Global Variable		Allows a value to be accessed in any VI	Introduces the possibility of a race condition
Functional Global Variable		•Non-reentrant VI •Allows a value to be accessed in any VI •Removes possibility of a race condition •Allows actions to be performed on data	

Breaking Data Flow – Functional Global Variable



This **Functional Global Variable** allows you to get the elapsed time since the last time you called the subVI.

Breaking Data Flow - Queues


This loop is the producer loop.
125
Condition that causes data to be enqueued.
True
data (can be any type)
Enqueue Data
Data to Enqueue
Releasing the queue stops the consumer loop(s).
Generate the data here.
stop
Dequeued Data
"Shutdown"
Wait(ms) in Consumer Loop
U32
This loop is a consumer loop.

Queues ensure no data is lost when it is transferred between loops running at different rates.

Data Flow

Which of the following does not conform to the Dataflow programming paradigm?

a. Shift Registers

b. Tunnels

c. SubVIs

d. Local variables

Data Flow

Which of the following does not conform to the Dataflow programming paradigm?

a. Shift Registers

b. Tunnels

c. SubVIs

d. Local variables

Local variables do not conform to the Dataflow paradigm because they communicate by reference, not by value. The basic premise of local variables is to allow transfer of data where it is impossible to use wires. This circumvents the Dataflow paradigm.

Data Flow

Which variable is commonly used to eliminate race conditions by preventing simultaneous access to code or data?

a. Functional global variable

b. Local variable

c. Global variable

d. Shared variable

Data Flow

Which variable is commonly used to eliminate race conditions by preventing simultaneous access to code or data?

a. Functional global variable

b. Local variable

c. Global variable

d. Shared variable

You can place critical data or sections of code in functional global variables. Since functional global variables are non-reentrant VIs, the possibility of race conditions is eliminated.

Data Flow

Which data synchronization mechanism ensures that no data is lost when an application temporarily provides data faster than it is able to process it?

a. Notifier

b. Queue

c. Semaphore

d. Local Variable

Data Flow

Which data synchronization mechanism ensures that no data is lost when an application temporarily provides data faster than it is able to process it?

~~a. Notifier~~

Notifiers pass data, but they can only pass one element at a time. Data is overwritten and lost if the program writes to the notifier twice before data is read.

b. Queue

~~c. Semaphore~~

~~d. Local Variable~~

Semaphores cannot pass data.

Local variables have no mechanism for determining when data is updated, so there is no way to tell if data is newly-acquired or not.

Data Flow

Which data synchronization mechanism ensures that no data is lost when an application temporarily provides data faster than it is able to process it?

~~a. Notifier~~

Notifiers pass data, but they can only pass one element at a time. Data is overwritten and lost if the program writes to the notifier twice before data is read.

b. Queue

Queues support multiple elements and operate using a FIFO principle, guaranteeing that no data is lost or overwritten.

~~c. Semaphore~~

Semaphores cannot pass data.

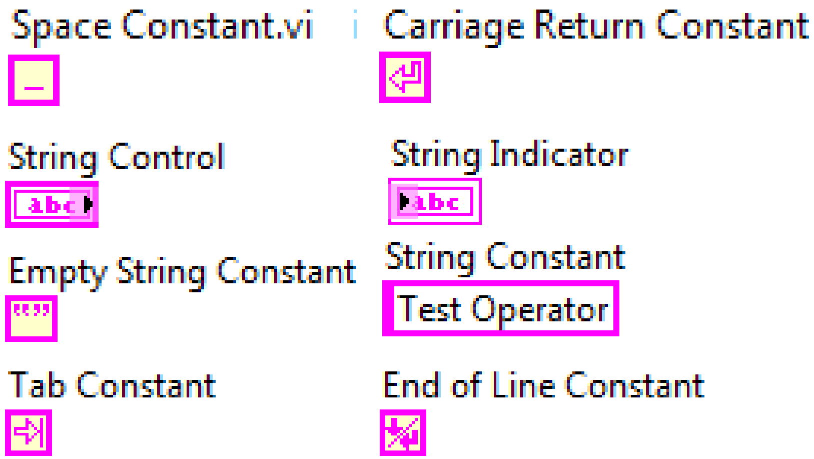
~~d. Local Variable~~

Local variables have no mechanism for determining when data is updated, so there is no way to tell if data is newly-acquired or not.

String Functions

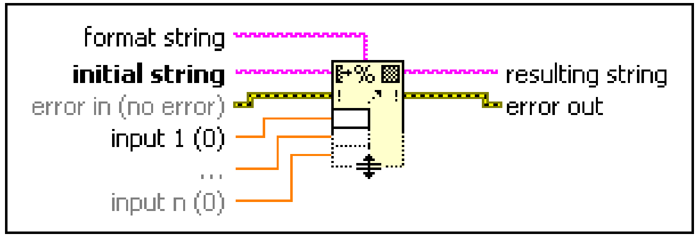
String Controls and Constants

- When possible, use the constants included in the string palette instead of typing in a string constant.



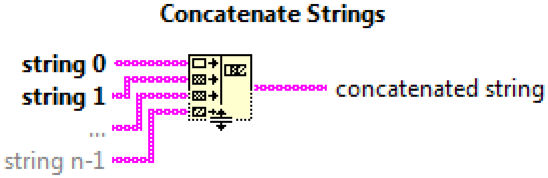
Format Into String

- Formats an input into a string
- Uses generic format string to specify output
- See detailed help for tips on the Format String syntax.



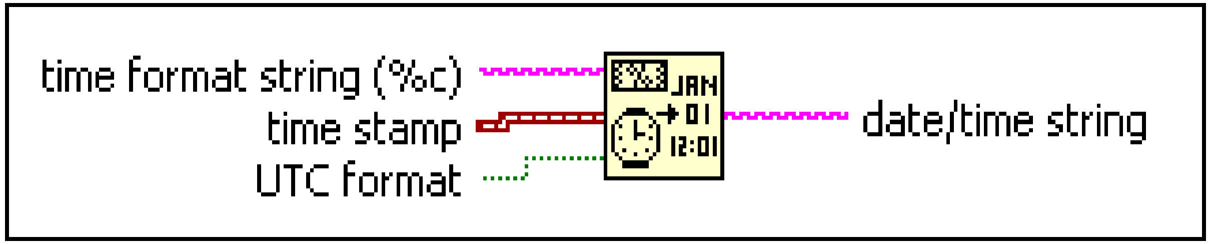
Concatenate Strings

- Combines two or more strings



Format Date/Time String

- Formats a time stamp or numeric value into a string
- Uses the time format string to specify the date/time string output
- Example:
  For a January 15, 2020 time stamp input and `%m-%d-%Y` time format string input, the function returns a date/time string: `01-15-2020`



Flatten To / Unflatten From String

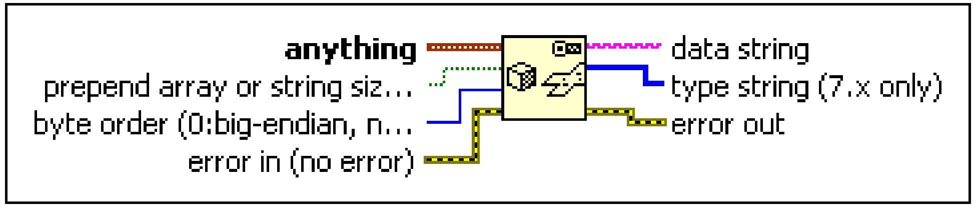
These functions *flatten* or *unflatten* LabVIEW data to or from binary strings which contain the flattened data.

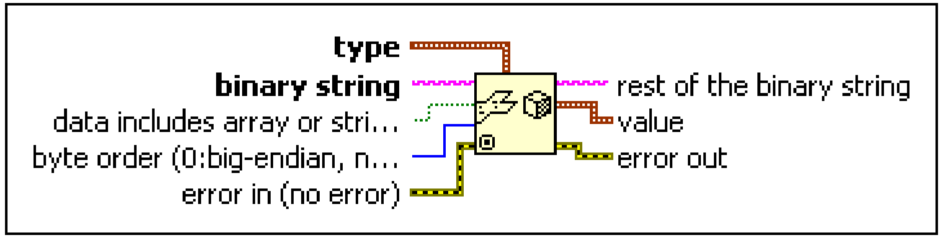
Flatten to string:

- Takes any input data and flattens it to a single string.
- Useful for transferring data to external programs

Unflatten from string:

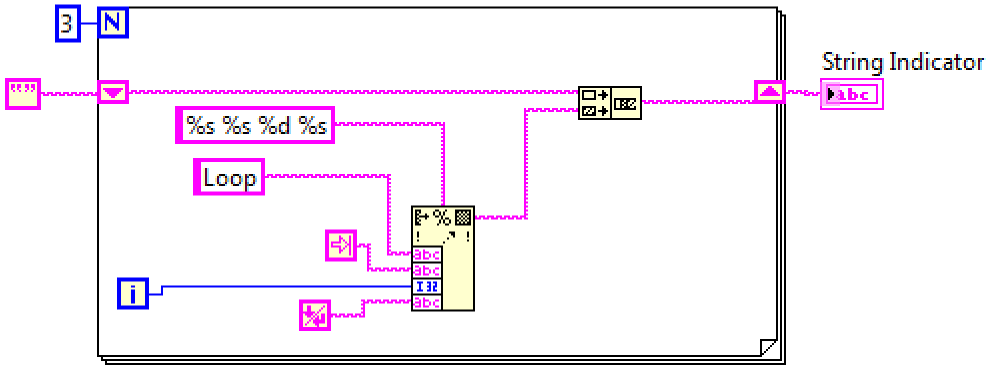
- Interprets a binary string as the specified data type
- Requires a data type template such as a numeric constant




type
binary string
data includes array or stri...
byte order (0:big-endian, n...
error in (no error)
rest of the binary string
value
error out

String Functions

What is displayed in “String Indicator” after running this code?



A. Loop0 Loop1 Loop2

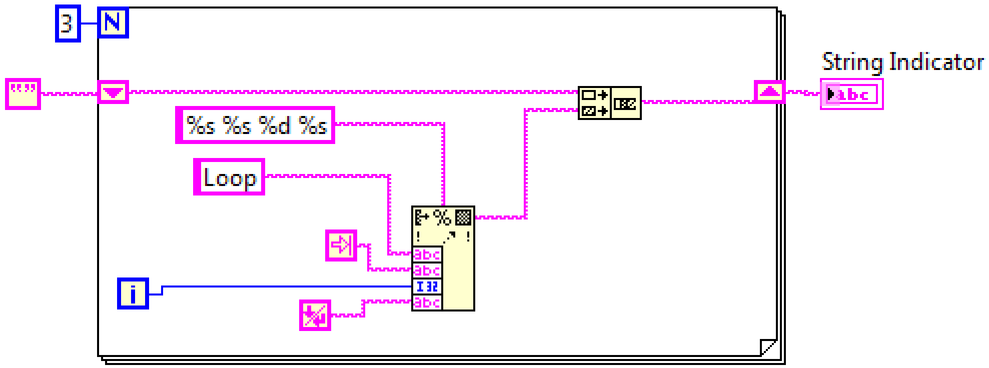
B. Loop 1
Loop 2
Loop 3

C. Loop 0 Loop 1 Loop 2

D. Loop 0
Loop 1
Loop 2

String Functions

What is displayed in “String Indicator” after running this code?



A. Loop0 Loop1 Loop2

B. Loop 1
Loop 2
Loop 3

C. Loop 0 Loop 1 Loop 2

D. Loop 0
Loop 1
Loop 2

The iteration terminal is zero-indexed, and each iteration generates a string with an EOL constant, so the correct answer must have three lines.

String Functions

You want to export data from LabVIEW into another environment, but the other environment does not recognize LabVIEW data types. You should:

A. Export the data using a property node.

B. Unflatten the data from a binary string and send it to the other environment.

C. Create a LabVIEW user event to transport the data.

D. Flatten the data to a binary string and send it to the other environment.

String Functions

You want to export data from LabVIEW into another environment, but the other environment does not recognize LabVIEW data types. You should:

A. Export the data using a property node.

B. Unflatten the data from a binary string and send it to the other environment.

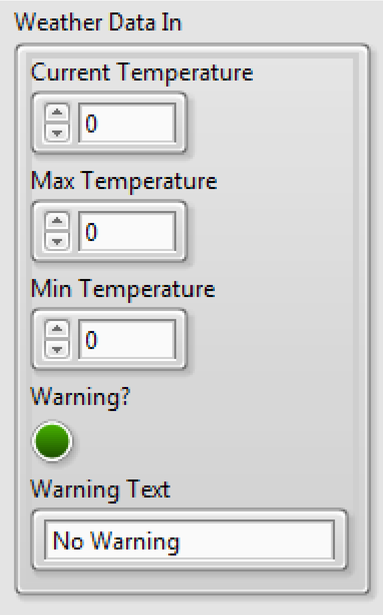
C. Create a LabVIEW user event to transport the data.

D. Flatten the data to a binary string and send it to the other environment.

Strings are widely usable. Oftentimes you will be able to bring your data into another programming environment as a string, parse the string, and consume the data.

Clusters

- Clusters group data elements of mixed types.
- Clusters are similar to a record or a `struct` in text-based programming languages.



Weather Data In

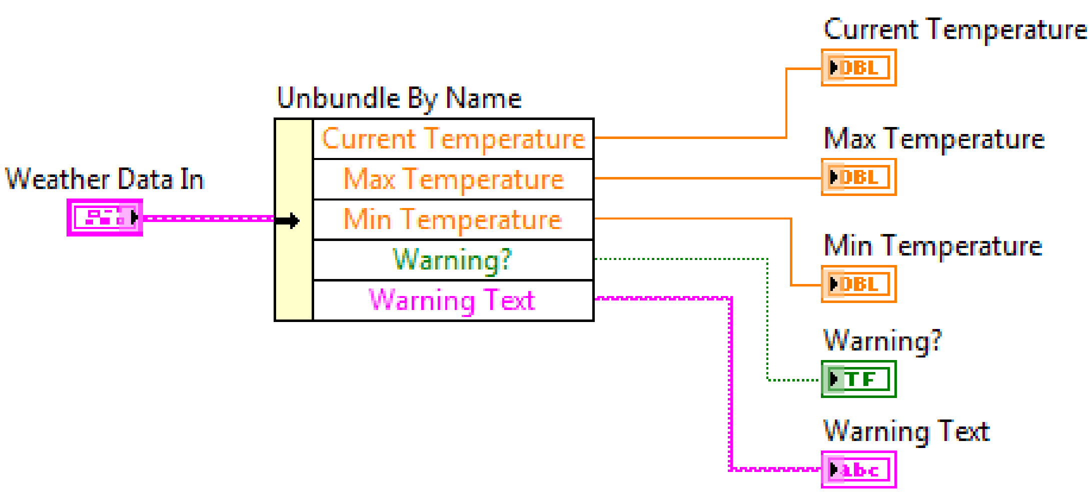
Current Temperature 0

Max Temperature 0

Min Temperature 0

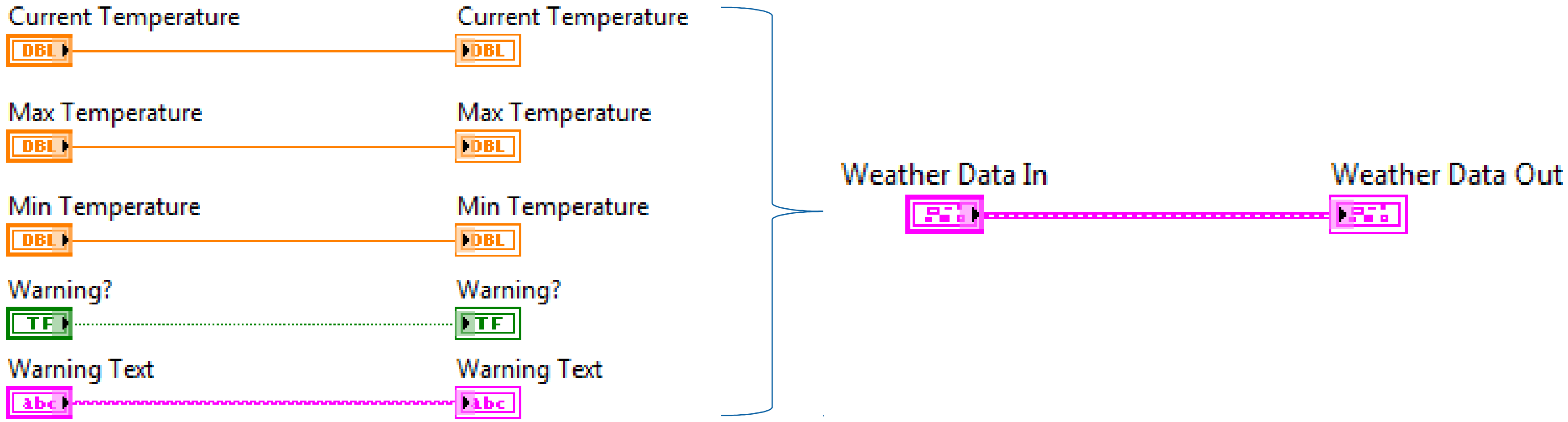
Warning? ●

Warning Text: No Warning



Why Use Clusters?

- Keep data organized.
  - Logically group related data values together.
  - Improve diagram readability by eliminating wire clutter.
- Reduce the number of connector pane terminals.



Cluster

Clusters provide a user with which of the following benefits?

a. Clusters allow a logical grouping of related data elements.

b. Clusters increase the number of Connector Pane terminals of SubVI’s.

c. Clusters help to reduce wire clutter on the Block Diagram.

d. Both A. and C.

Cluster

Clusters provide a user with which of the following benefits?

a. Clusters allow a logical grouping of related data elements.

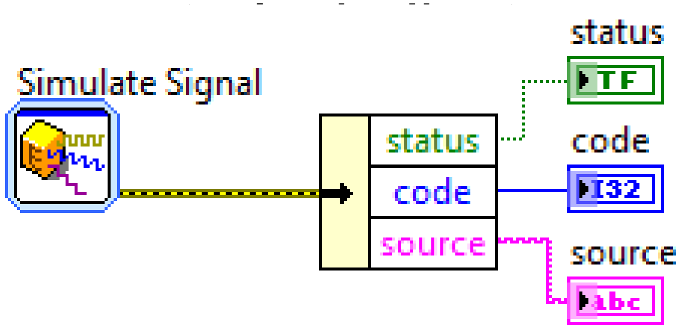
b. Clusters increase the number of Connector Pane terminals of SubVI’s.

c. Clusters help to reduce wire clutter on the Block Diagram.

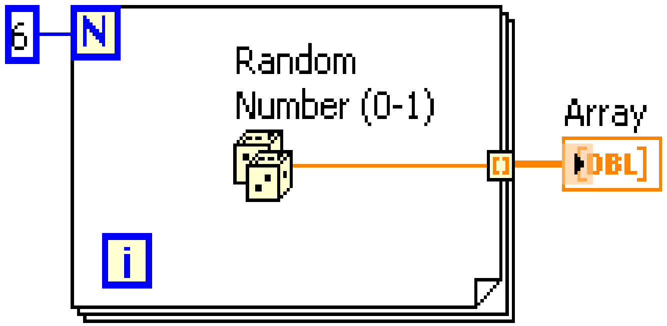
d. Both A. and C.

Clusters vs. Arrays

- Clusters are a fixed size.
- Clusters can contain mixed data types.
- Clusters can be a control, an indicator, or a constant.
  - All elements have to be



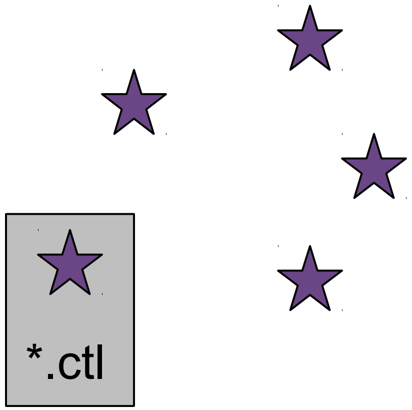
- Arrays vary in size.
- Arrays contain only one data type.
- Arrays can be a control, an indicator, or a constant.



Custom Controls & Type Definitions

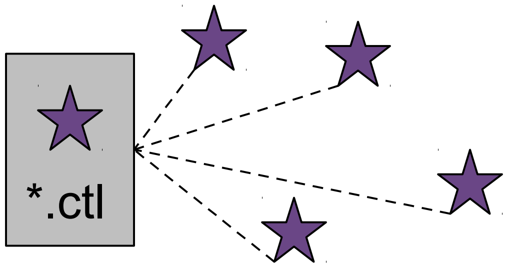
Control

- No connection between the one you saved and the instance in the VI
- Update the file, but the instances are not updated



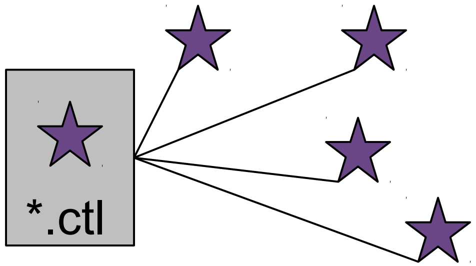
Type Def

- Connection between the saved file and all instances
- Forces the data type of each instance to be identical (clusters, enum)
- Changes made to file will populate throughout each instance



Strict Type Def

- Connection between saved file and all instances
- Forces *everything* about an instance to be identical to the strict type definition, except:
  - label
  - description
  - default value



Controls and Type Definitions

You customize a control, select Control from theType Def. Status pull-down menu, and save the control as a .ctl file.

You then use an instance of the custom control on your front panel window. If you open the.ctl file and modify the control, does the control on the front panel window change?

a. Yes

b. No

Controls and Type Definitions

You customize a control, select Control from theType Def. Status pull-down menu, and save the control as a .ctl file.

You then use an instance of the custom control on your front panel window. If you open the.ctl file and modify the control, does the control on the front panel window change?

a. Yes

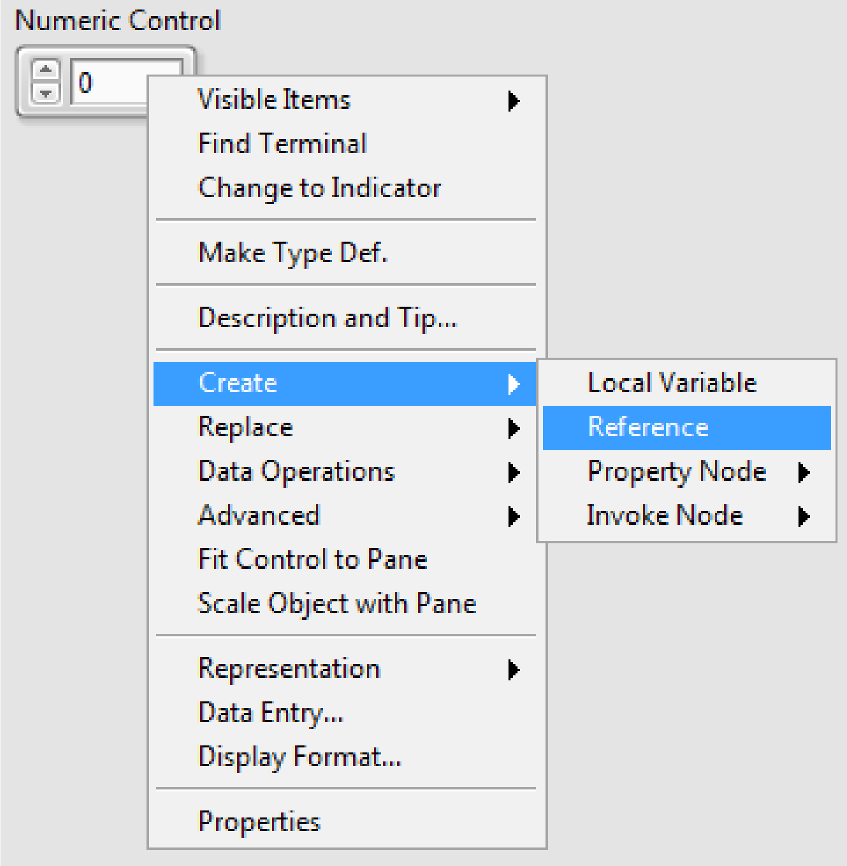
b. No

A **custom control** is used to create controls that behave like existing controls but look different. Instances are not linked to a.ctl file.

A **type definition control (type def)** is used for changing all instances of a linked control in a single edit.

Control References

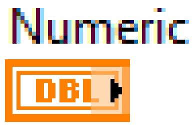
- A control reference is a reference to a front panel object.
- Wire control references to
  generic Property Nodes.
- Pass control references to subVIs.

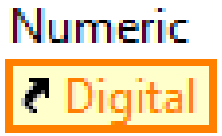


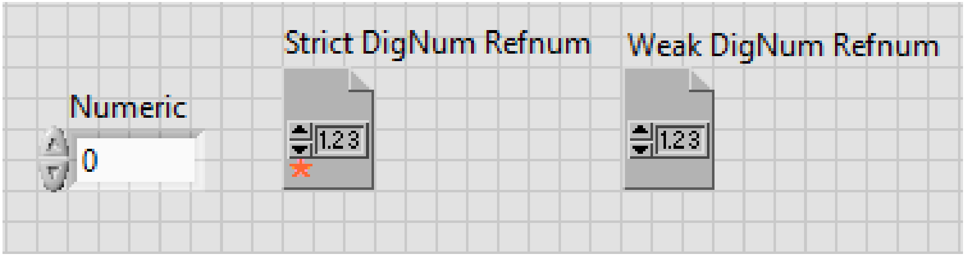
Control References

Control references allow you to manipulate controls located on the front panel of VIs loaded in memory.

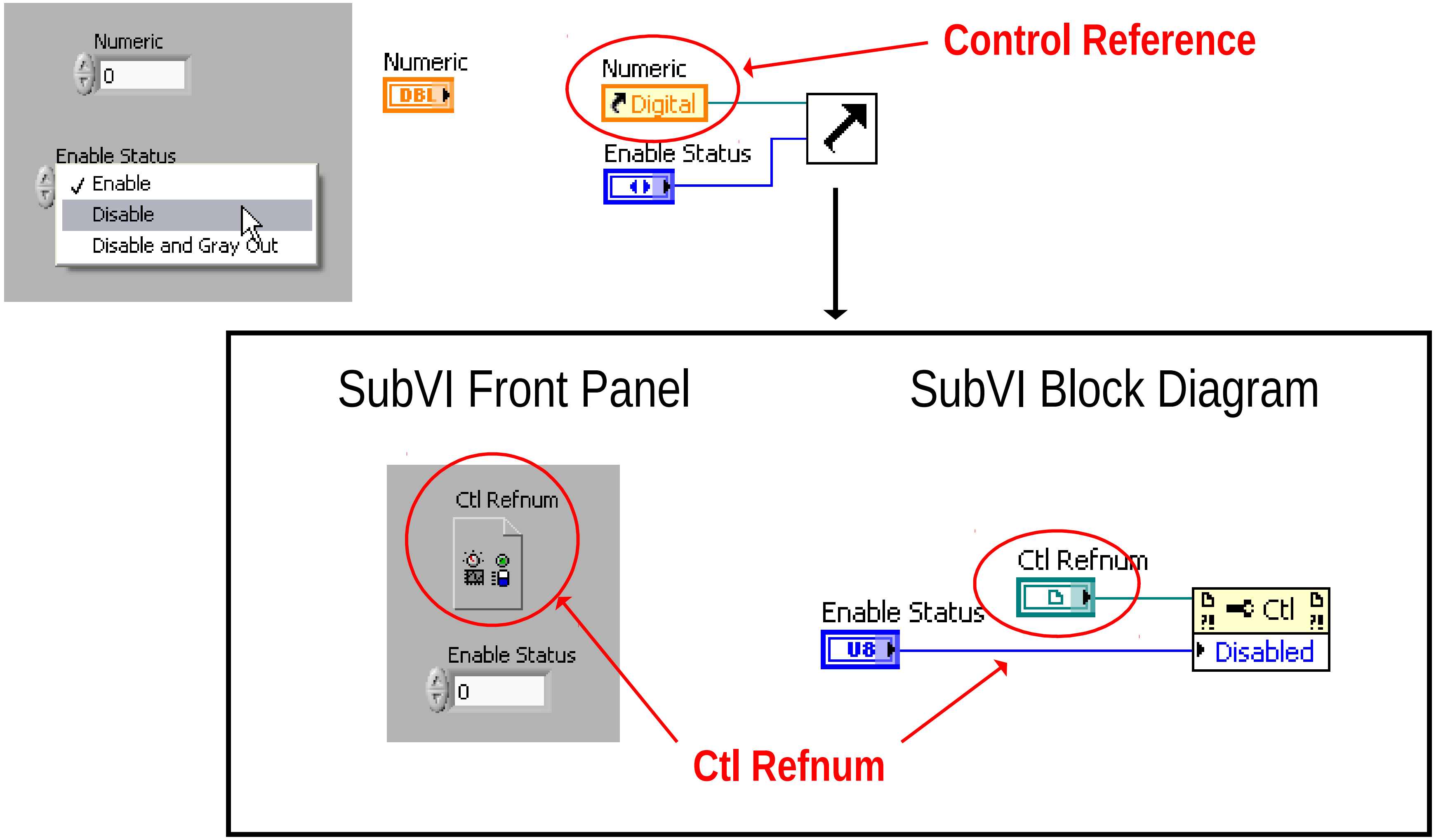
- Strictly typed control refnums – accept only control refnums of exactly the same data type
- Weakly typed control refnums - accept references to **any** DigNum control







Control References



Control Reference

SubVI Front Panel

SubVI Block Diagram

Ctl Refnum

Control References

Why would you want to use a weakly typed control refnum as a subVI input when wiring your connector pane?

A. To ensure only controls of one data type can be used as the input.

B. You want to make your subVI capable of manipulating controls with varying data types.

C. You want to be able to manipulate many controls at once.

D. It is best practice to avoid strictly typed control refnums in subVIs.

Control References

Why would you want to use a weakly typed control refnum as a subVI input when wiring your connector pane?

A. To ensure only controls of one data type can be used as the input.

B. You want to make your subVI capable of manipulating controls with varying data types.

C. You want to be able to manipulate many controls at once.

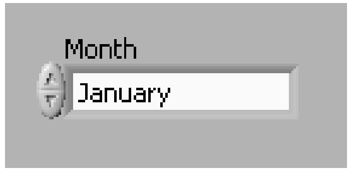
D. It is best practice to avoid strictly typed control refnums in subVIs.

Weakly typed refnums do not retain information about the data type of the control they reference. This allows you to specify a particular class of control as an input while allowing that control to have multiple representations. Data values will become variant in weakly typed refnums to account for this flexibility.

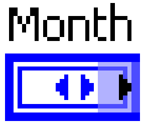
Supplemental Concepts

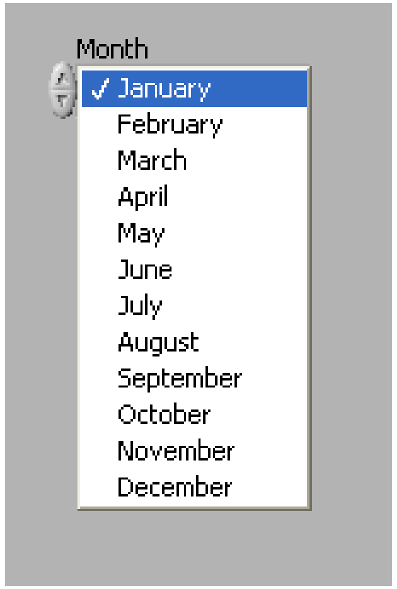
Enumeration

- An enum represents a pair of values, a string and a numeric, where the enum’s value is one of a defined list of value pairs



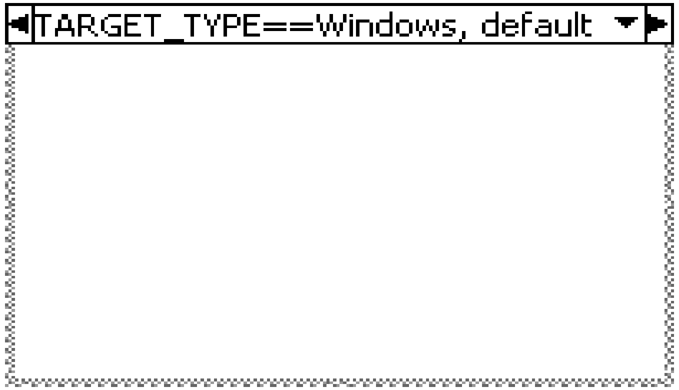
- Appears as a string to you, and a number to computer



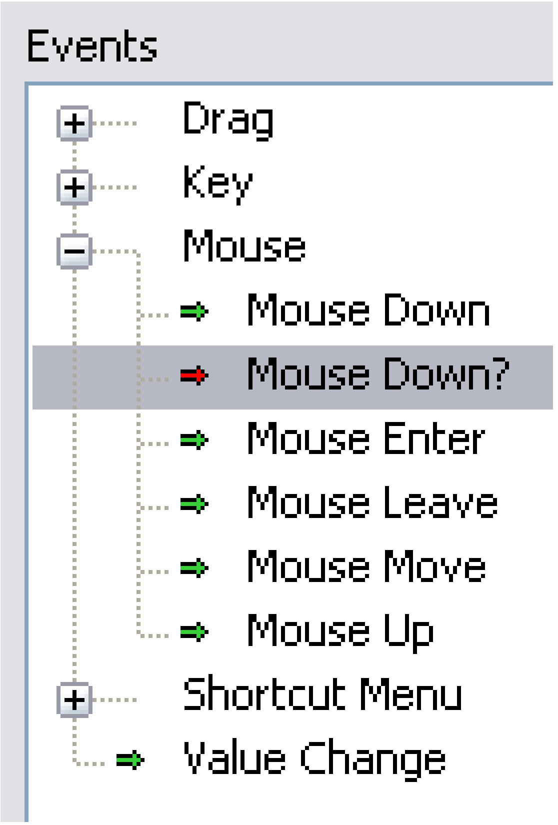


Conditional Disable Structure

- Use to define conditions that indicate which code on the block diagram executes
  - Examples:
  - If running as an executable, then programmatically close LabVIEW when VI finishes
  - If running on Windows, look for file here; if running on Mac OSX then look here.



Notify and Filter Events in Config. Window



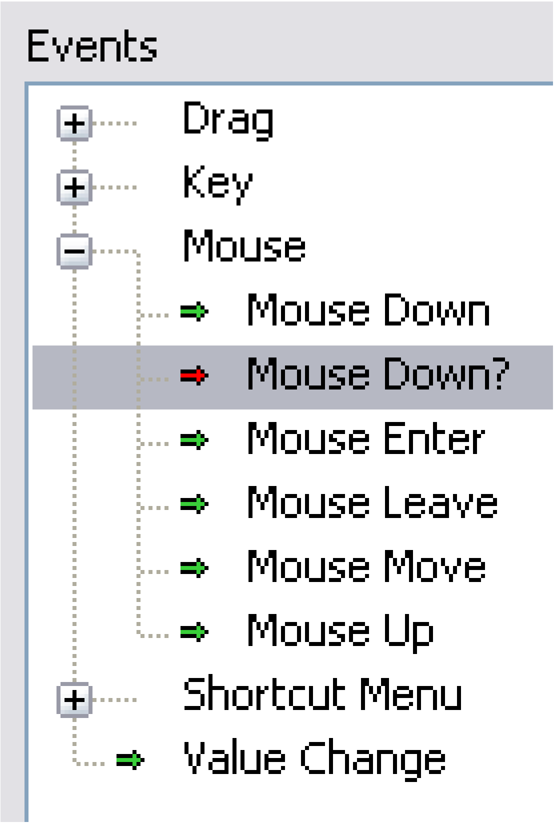
Notify Events (green arrow)

User action has already occurred

Filter Events (red arrow)

User performed action but event is not processed. Allows you to customize event handling.

Notify and Filter Events in Config. Window



Events

- Drag
- Key
- Mouse
  - Mouse Down
  - Mouse Down?
  - Mouse Enter
  - Mouse Leave
  - Mouse Move
  - Mouse Up
- Shortcut Menu
- Value Change

Notify Events (green arrow)

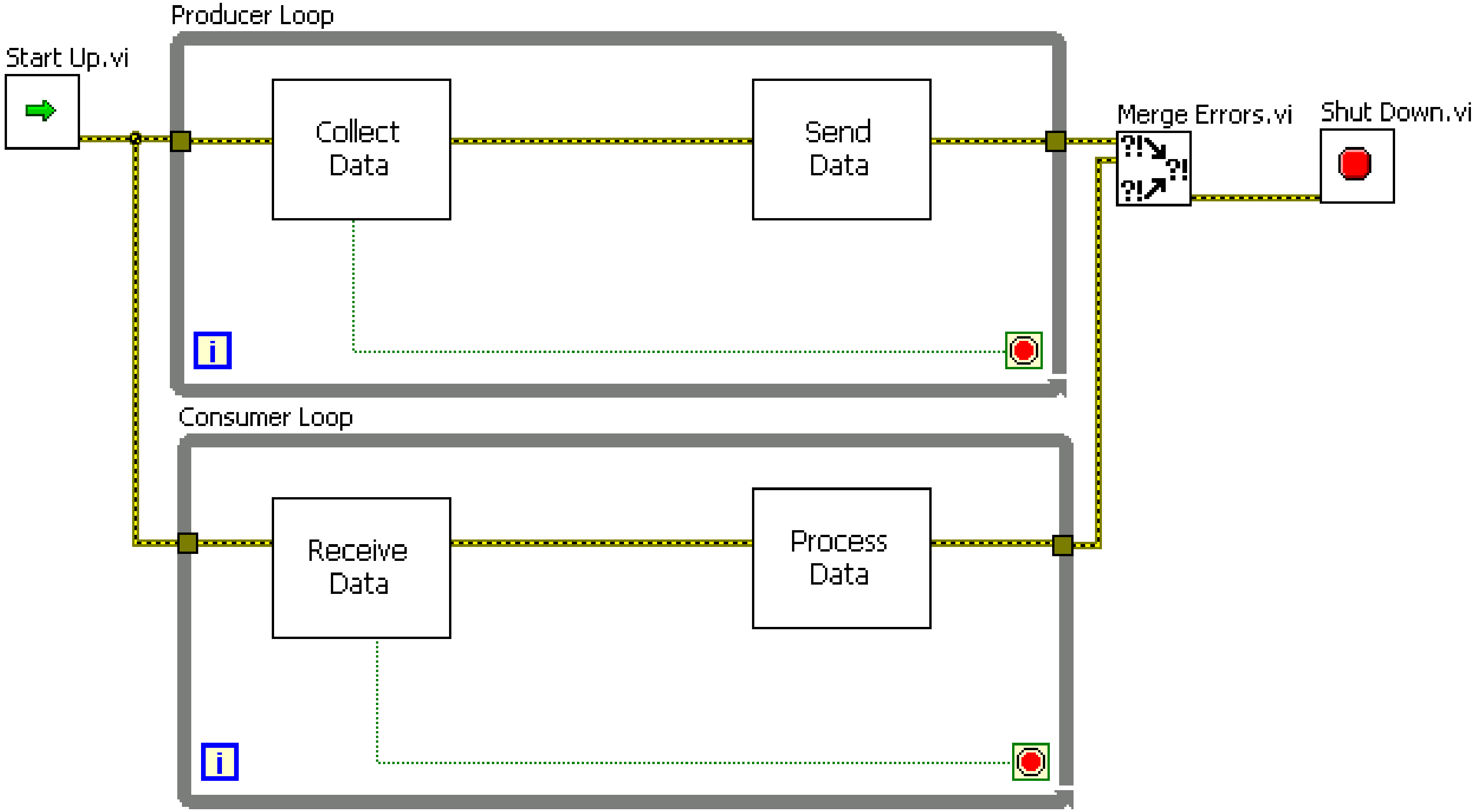
User action has already occurred

Filter Events (red arrow)

User performed action but event is not processed. Allows you to customize event handling.

Multiple Loop Design Pattern

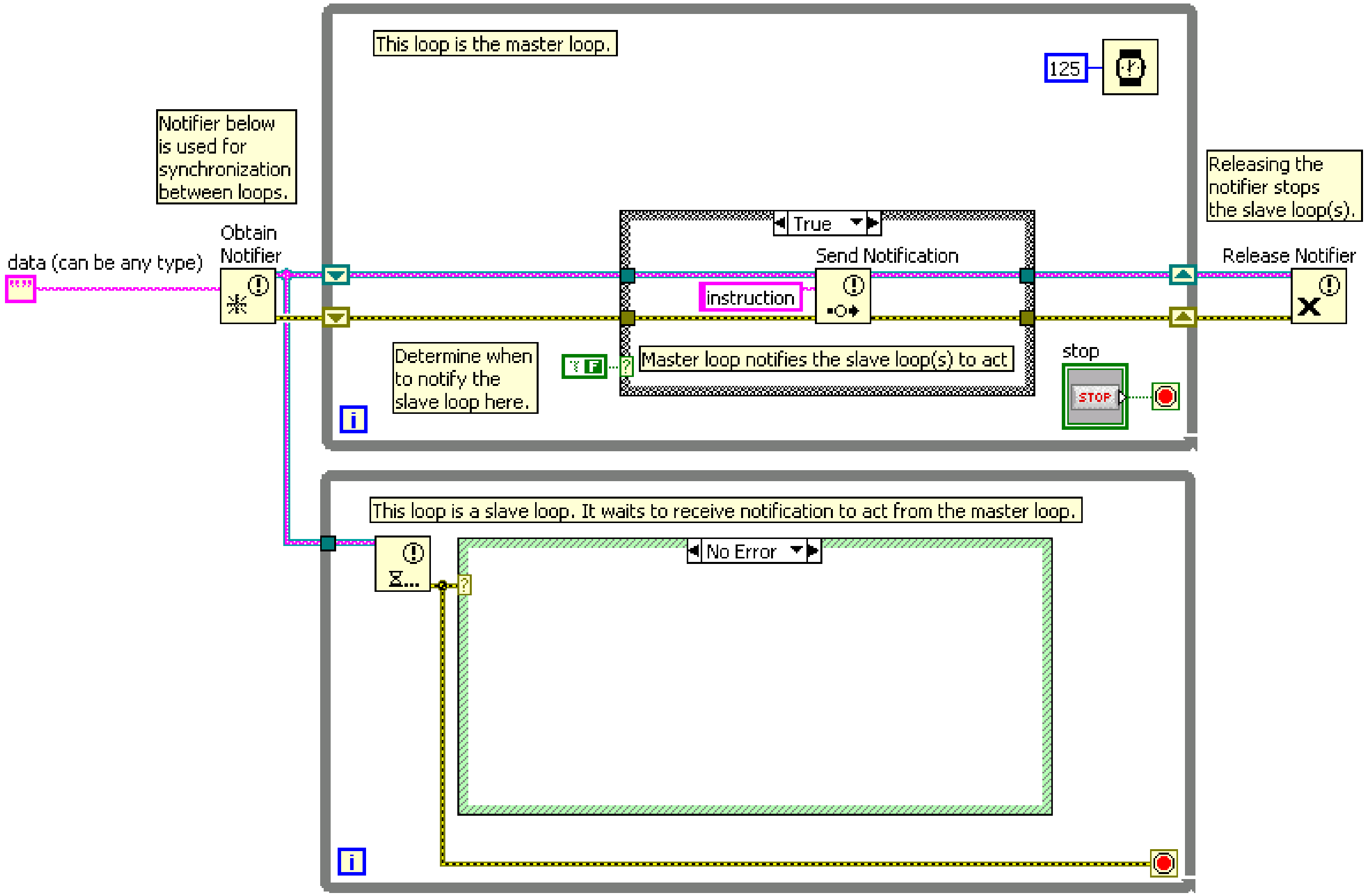
- Producer/Consumer



Multiple Loop Design Pattern

- Master/Slave: a type of Producer/Consumer

• uses notifiers to allow loops to run at different rates



Notifier below is used for synchronization between loops.

This loop is the master loop.

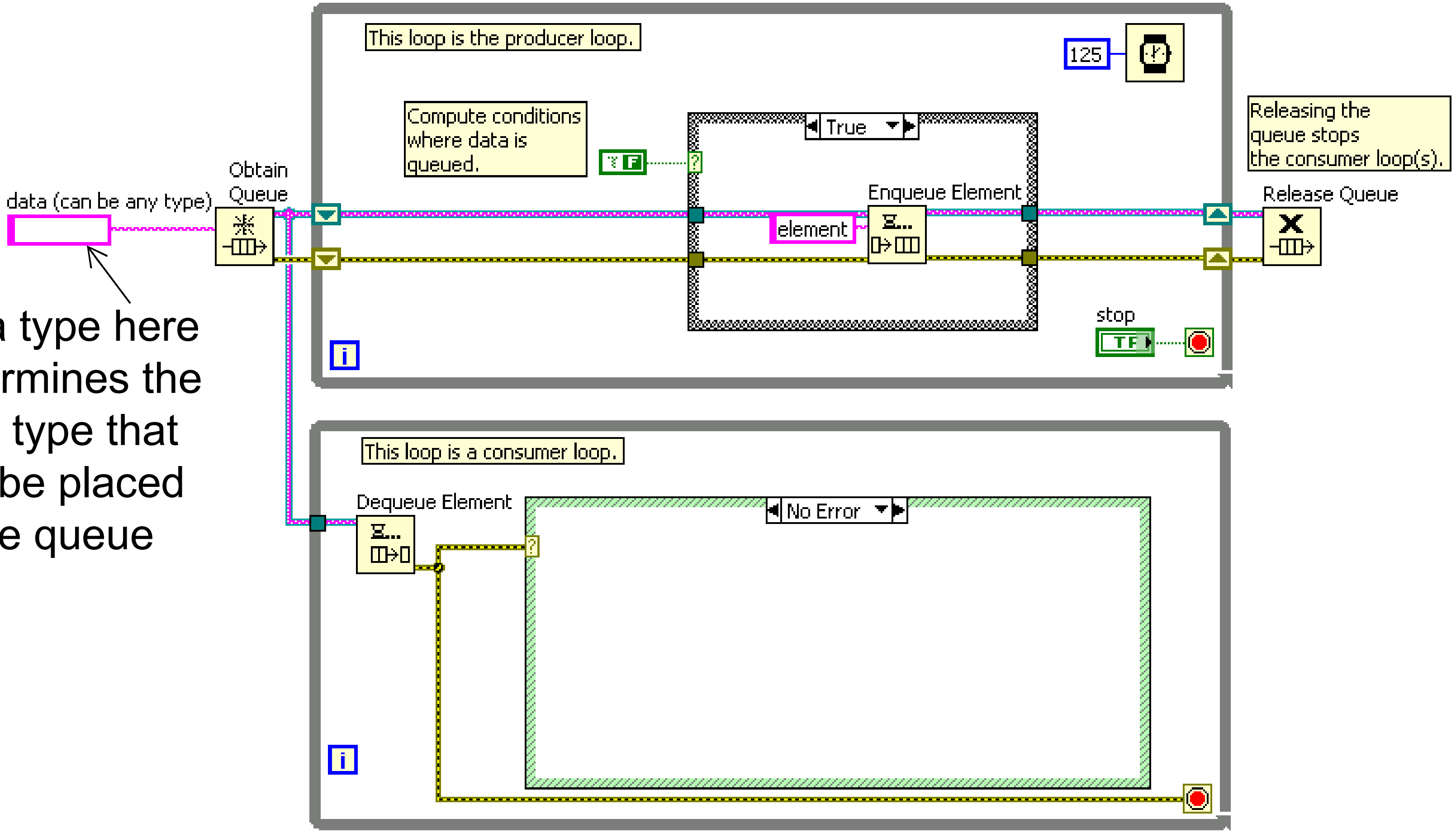
Determine when to notify the slave loop here.

Master loop notifies the slave loop(s) to act

Releasing the notifier stops the slave loop(s).

This loop is a slave loop. It waits to receive notification to act from the master loop.

Producer/Consumer with Queues

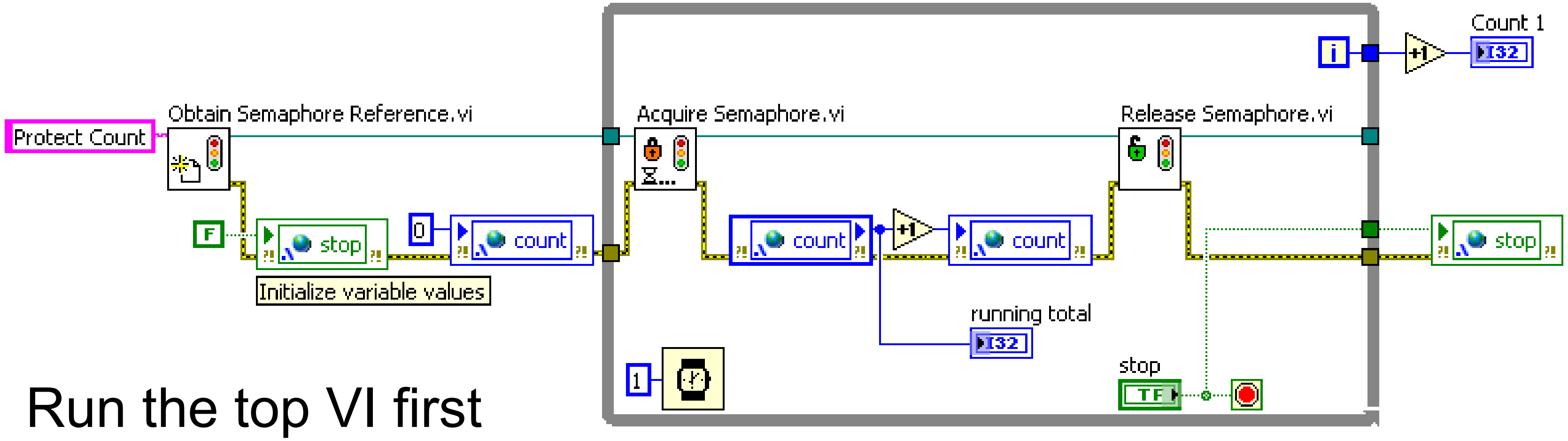

This loop is the producer loop.
125
Compute conditions where data is queued.
True
Enqueue Element
element
Releasing the queue stops the consumer loop(s).
Release Queue
Obtain Queue
data (can be any type)
stop
This loop is a consumer loop.
Dequeue Element
No Error

Data type here determines the data type that can be placed in the queue

Notifier vs. Queue

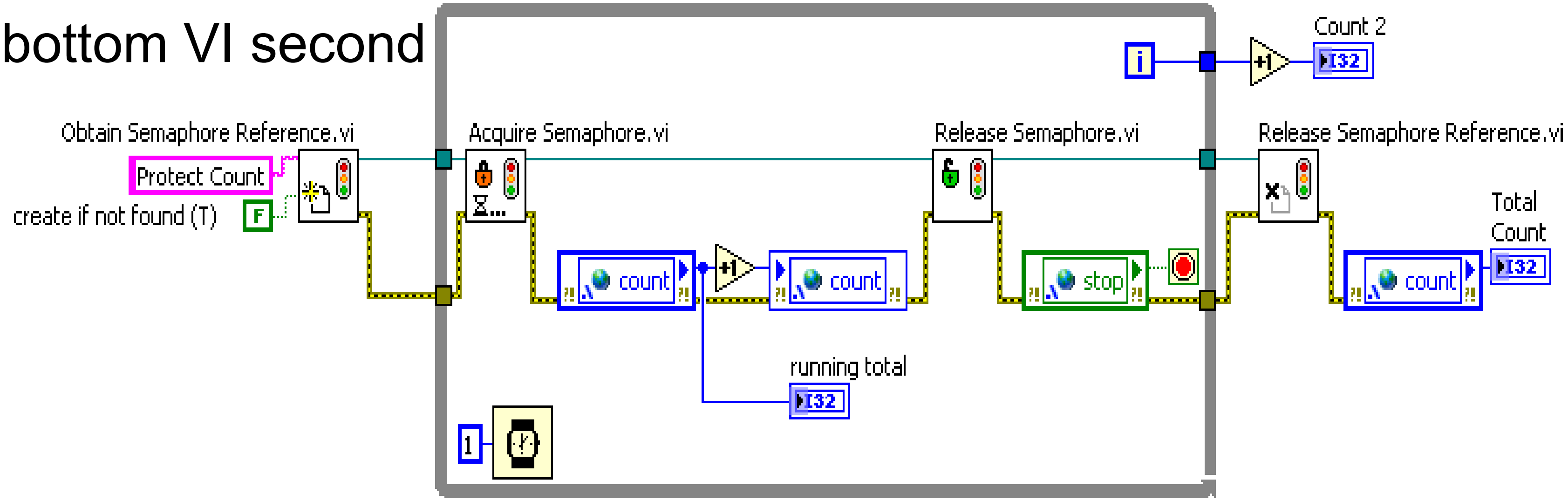
Tool	Function	When to use
Notifier	- sends alert to helps control timing of parallel loops - can transfer data between loops - data NOT buffered (lossy)	- have a parallel loops that are running at different rates and 1+ loop is dependent on another - want to transfer data from one loop to another
Queue	- helps control timing of parallel loops - transfers data between loops - buffers data to be transferred (FIFO)	- have a parallel loops that are running at different rates and 1+ loop is dependent on another - want to transfer ALL data from one loop to another

Semaphores



Run the top VI first

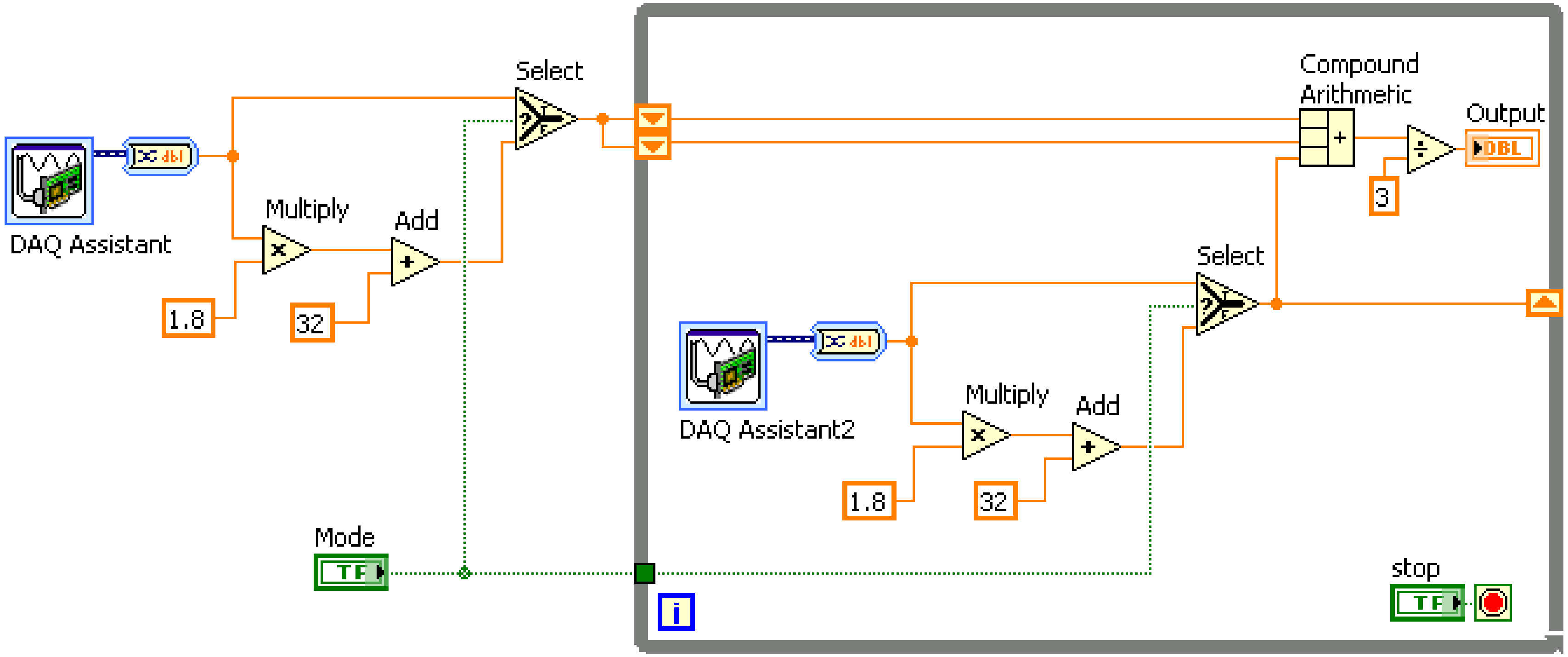
Run the bottom VI second



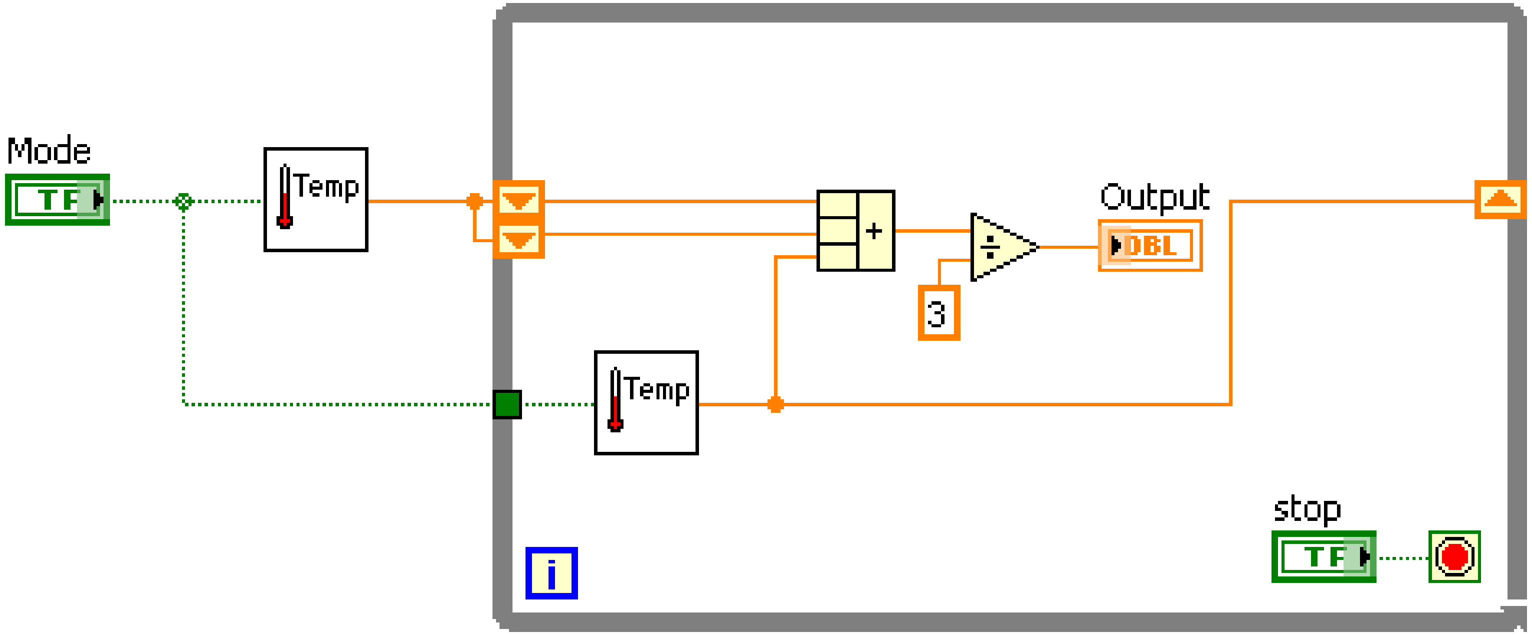
File Formats

	ASCII	TDMS	Direct Binary
Numeric Precision	Good	Best	Best
Share data	Best (Any program easily)	Better (NI Programs easily)	Good (only with detailed metadata)
Efficiency	Good	Best	Best
Ideal Use	Share data with other programs when file space and numeric precision are not important	Share data with programs when storing simple array data and metadata	Store numeric data compactly with ability to random access

Modularity and SubVIs

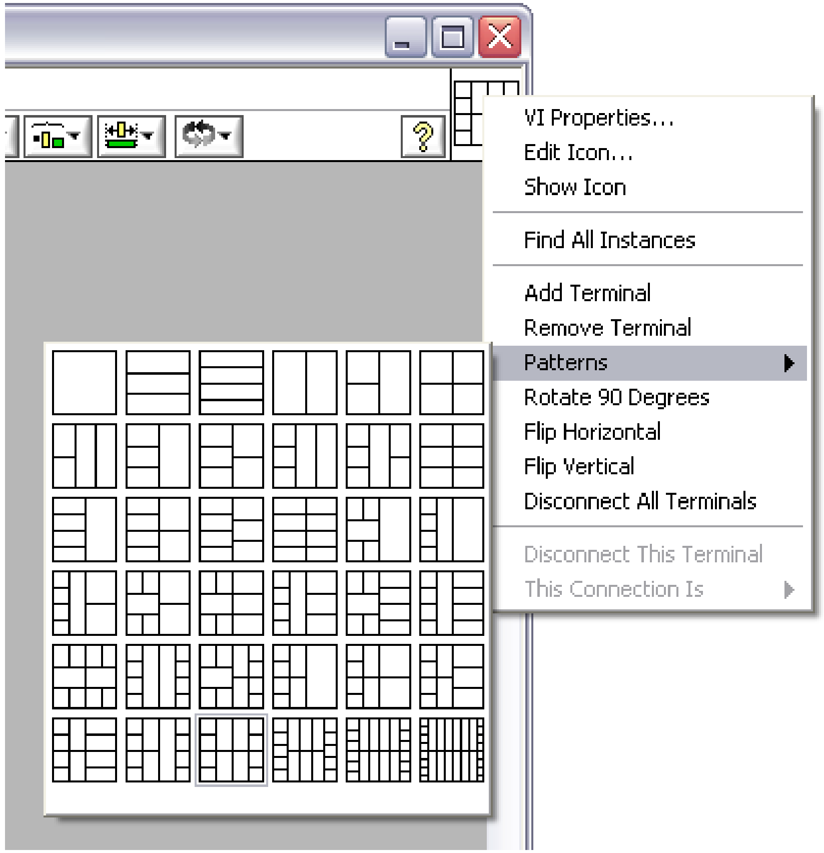


Modularity and SubVIs



Icon and Connector Pane: Setting up the Connector Pane

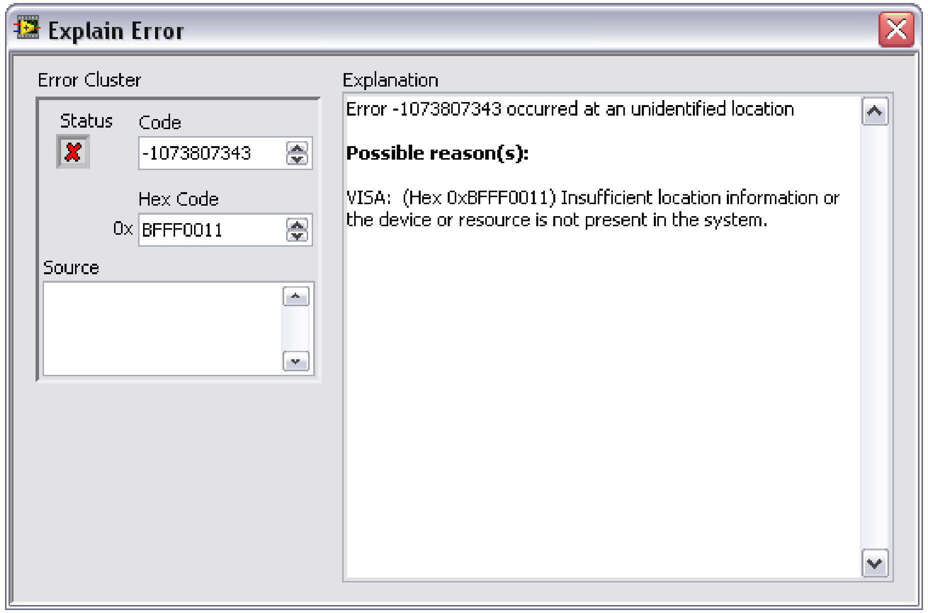
- Right-click the icon in the upper right corner of the front panel and select Show Connector (before LV 2011)
  - Each rectangle on the connector pane represents a terminal
- Select a different pattern by right-clicking the connector pane and selecting Patterns from the shortcut menu



Errors vs. Warnings

- Error
- Status = TRUE

- Warning
- Status = FALSE
- Code is non-zero



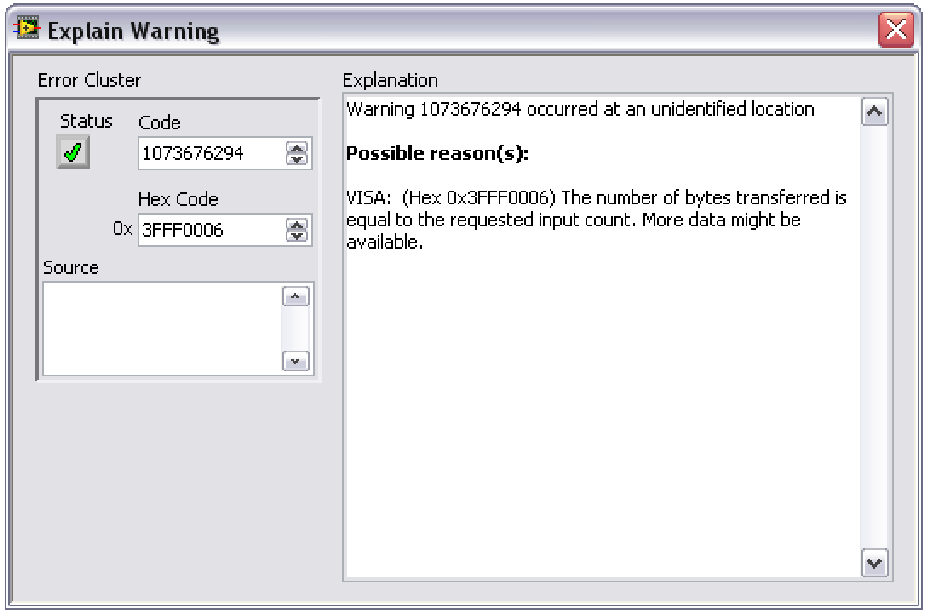
Code: -1073807343

Hex Code: 0x BFFF0011

Error -1073807343 occurred at an unidentified location

Possible reason(s):

VISA: (Hex 0xBFFF0011) Insufficient location information or the device or resource is not present in the system.



Code: 1073676294

Hex Code: 0x 3FFF0006

Warning 1073676294 occurred at an unidentified location

Possible reason(s):

VISA: (Hex 0x3FFF0006) The number of bytes transferred is equal to the requested input count. More data might be available.

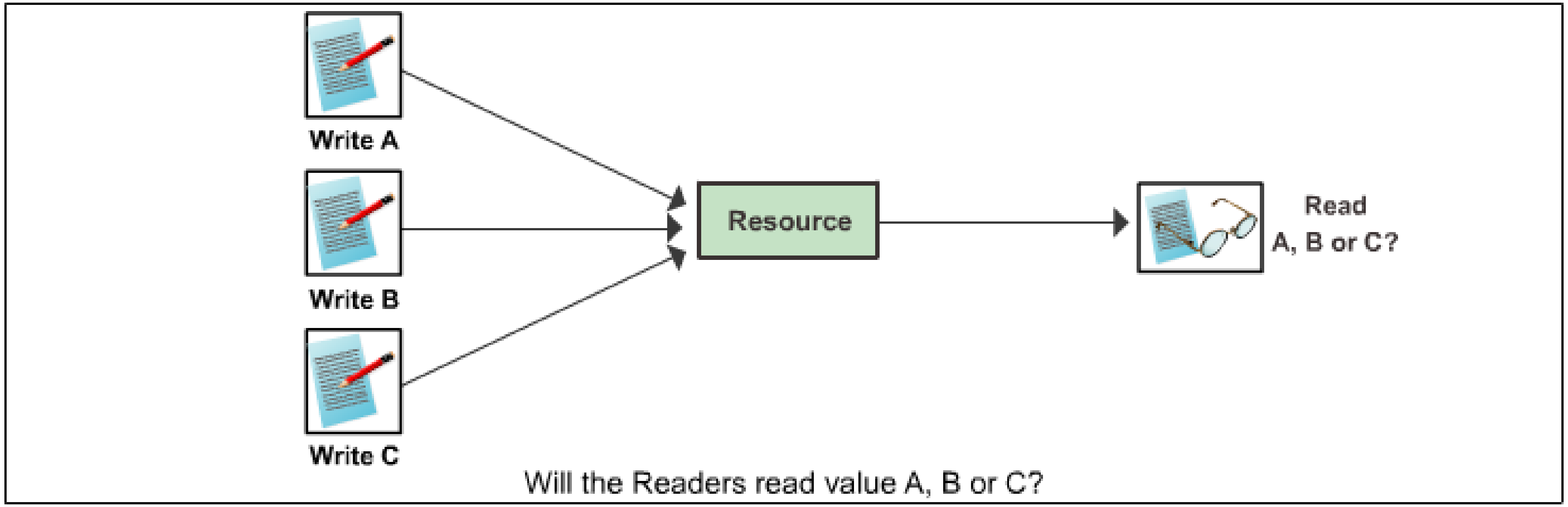
Debugging Tools

- use highlight execution
- use probes to see the current value in the wire
- set breakpoints to pause execution at a specific point in the code
- single-step through the code

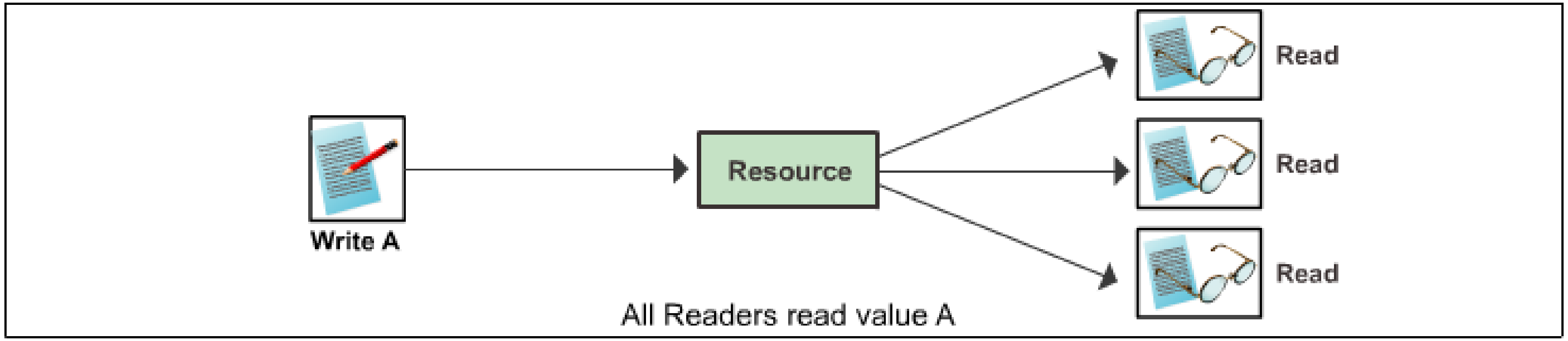
Variables: Recap

Variable Type	Scope	Notes
Local variable	A single VI	• Tied to a front panel control/indicator
Global variable	Multiple VIs on same computer	• Tied to a special global VI that has a front panel but no block diagram
Functional global	Multiple VIs on same computer	• Implemented using a While Loop with an uninitialized shift register to store global data
Single-process shared variable	Multiple VIs on same computer	• Implemented using a project library in a project • Can easily convert into a network-published shared variable
Network-published shared variable	Across an Ethernet network	• Implemented using a project library in a project • Often used to communicate with Real-Time targets

Race Conditions: Shared Resources

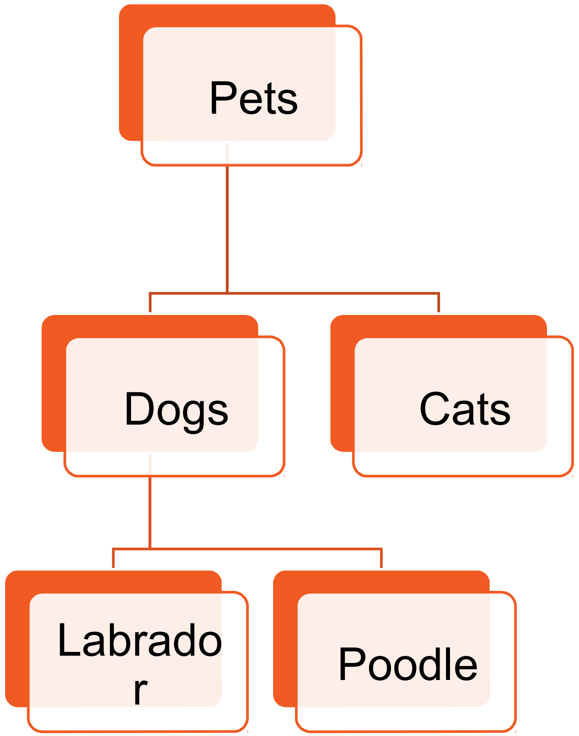


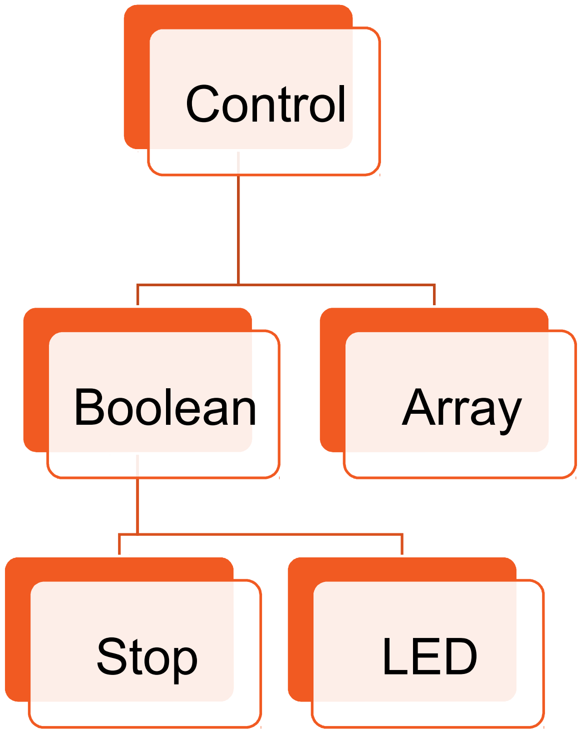
Will the Readers read value A, B or C?

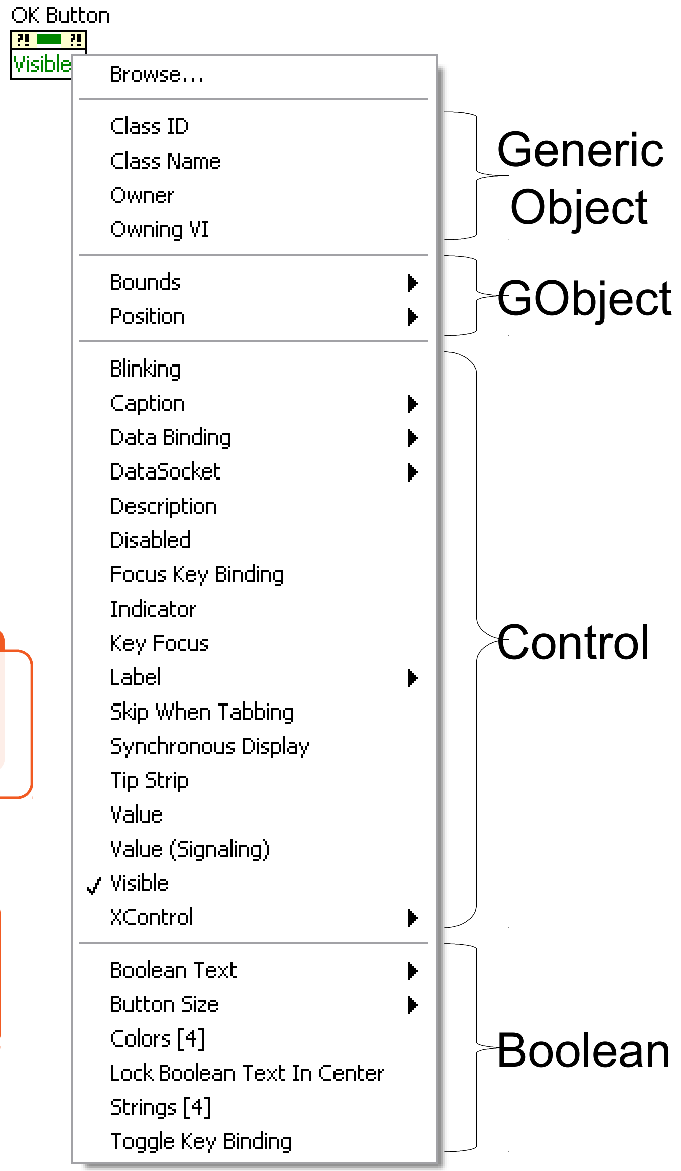


All Readers read value A

VI Server: Organization



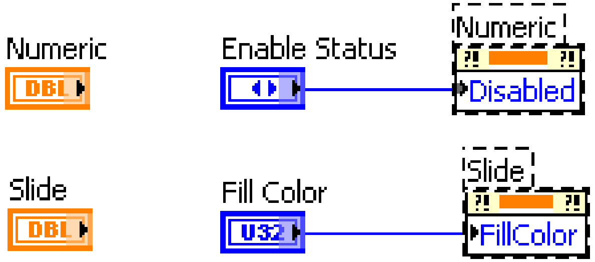


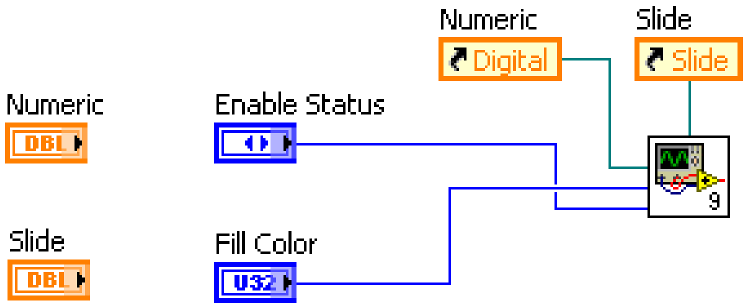


Control References – Create SubVI

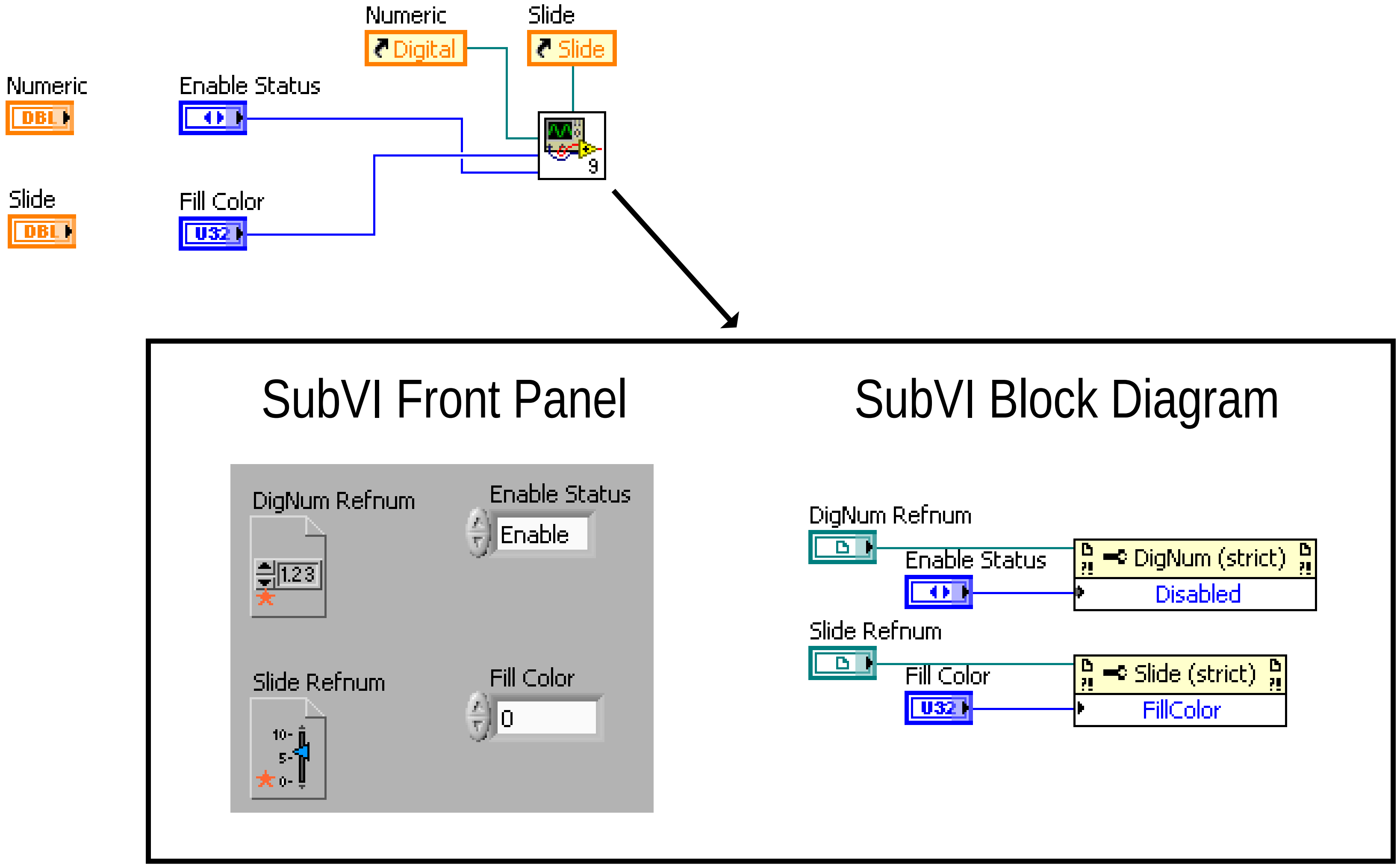
To create explicitly-linked Property Nodes in a subVI:

1. Create your VI
2. Select the portion of the block diagram that will be in the subVI
3. Select **Edit»Create SubVI**; LabVIEW automatically creates the control references needed for the subVI
4. Customize and save the subVI





Control References – Create SubVI



SubVI Front Panel

SubVI Block Diagram