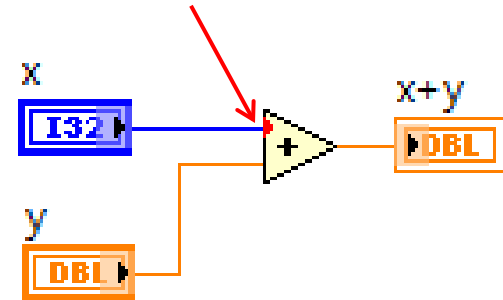


CLAD Most Missed Concepts

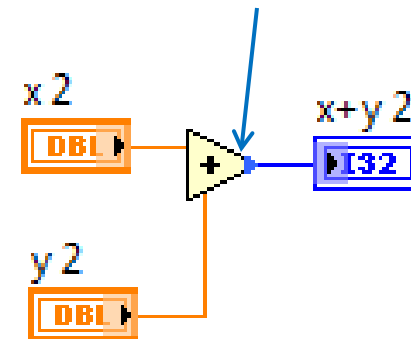
Coercion Dots

- 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



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

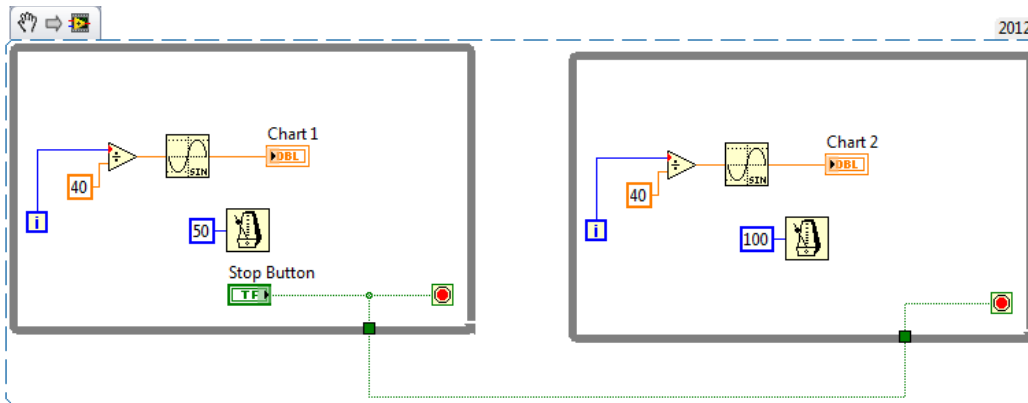
Coercion Dots

When does the coercion dot appear?

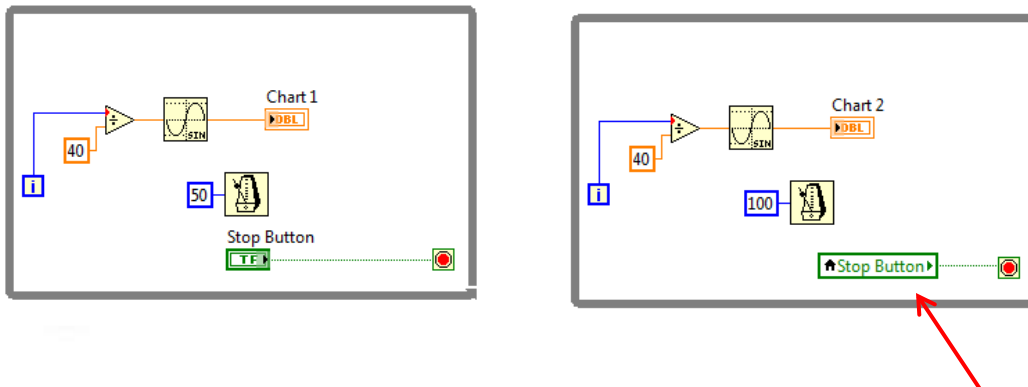
- A. The data types are consistent
- B. A polymorphic operation is performed on the data
- C. The data conversion is forced due to the mismatched numeric representation between terminals**
- D. Data values are being coerced because they are out of range

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

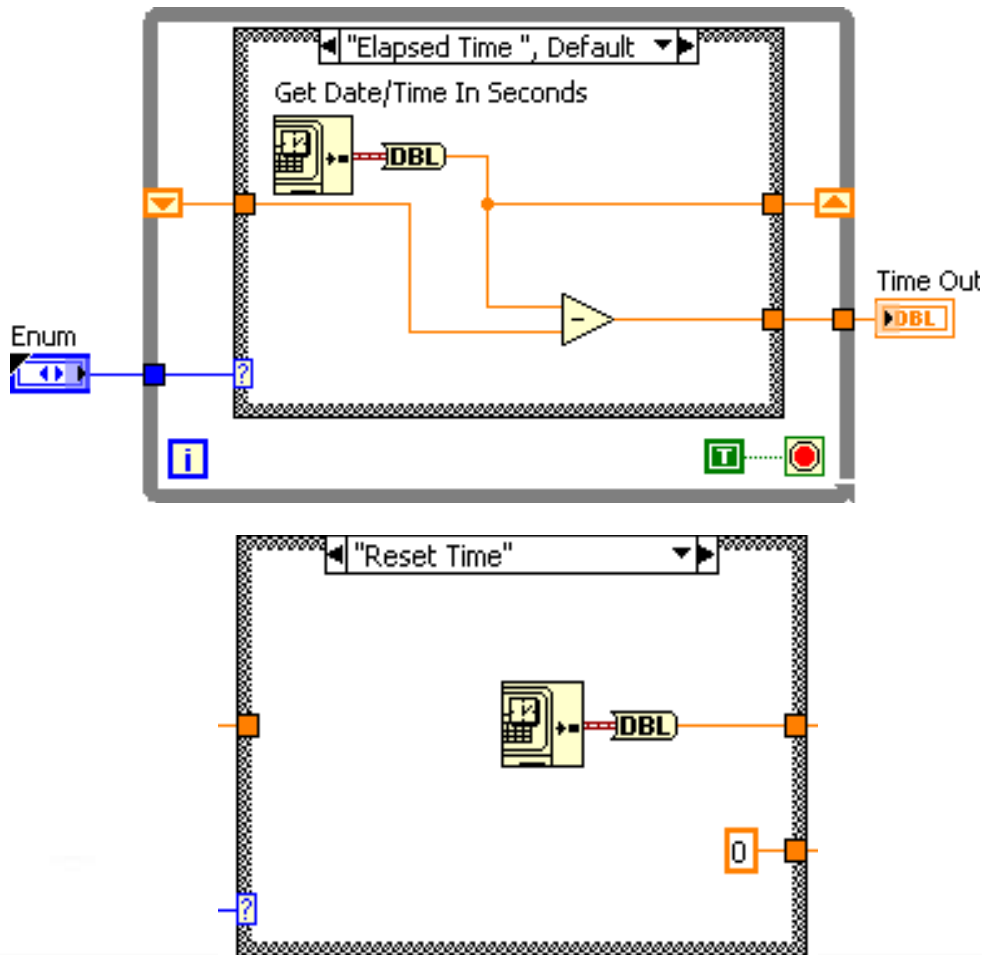
Drawbacks: Introduces a Possible Race Condition

Local Variable referencing the Stop Button

Breaking Data Flow

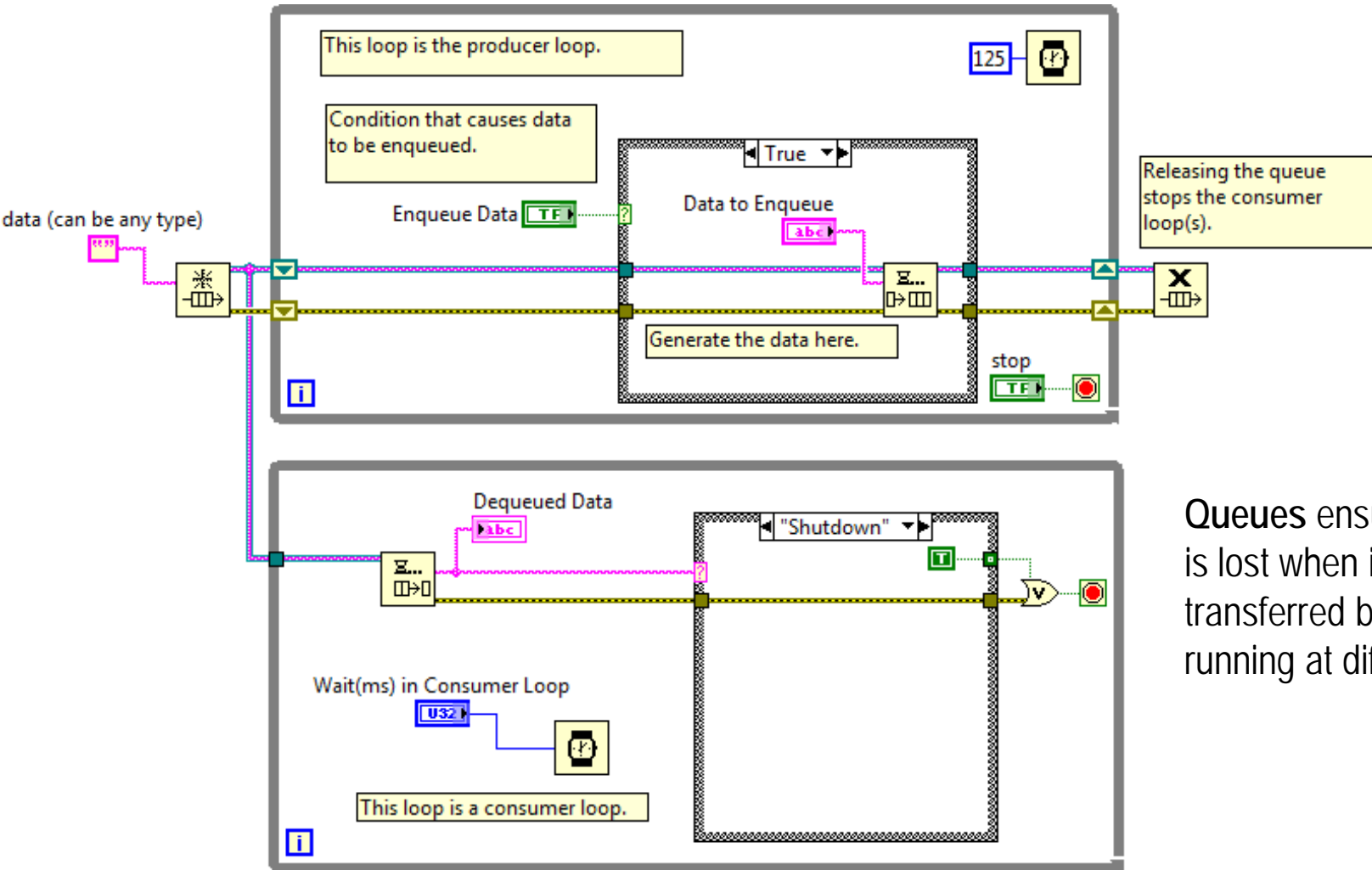
Name	Appearance	Function	Drawbacks
Wire		Connects data at different terminals	Must follow Data Flow
Local Variable		Allows a value to be accessed anywhere in a single VI	Introduces the possibility of a race condition
Global Variable		Allows a value to be accessed in any VI	Introduces the possibility of a race condition
Functional Global Variable		<ul style="list-style-type: none"> •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



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

Breaking Data Flow

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

- a. Shift Registers
- b. Tunnels
- c. SubVIs
- d. Local variables

Breaking Data Flow

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

a. Shift Registers

b. Tunnels

c. SubVIs

d. Local variables

Justification: 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.

Breaking 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

Breaking 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

Justification: 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.

Breaking 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

Breaking 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

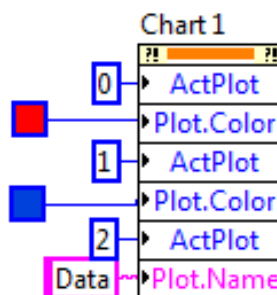
Justification: Answer C is incorrect because semaphores cannot pass data. Answer A is incorrect because notifiers pass data, but they can only pass one element at a time. Data is overwritten and lost if the program writes to the notifier twice before data is read. Answer D is incorrect because local variables have no mechanism for determining when data is updated, so there is no way to tell if data is newly-acquired or not. Queues support multiple elements and operate using a FIFO principle, guaranteeing that no data is lost or overwritten.

Property Nodes

A property node can be implicit or explicit.

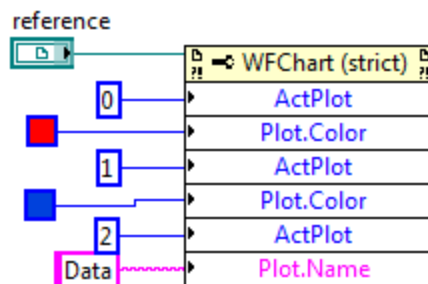
A property node executes top-down

Implicit Property Node



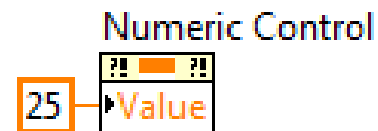
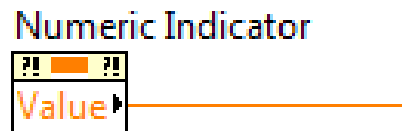
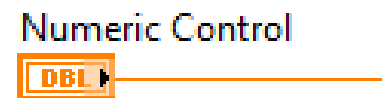
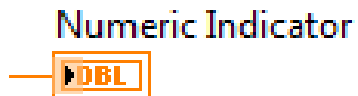
1. Plot 0 is set active
2. Active plot (0) changed to red
3. Plot 1 is set active
4. Active plot (1) is changed to blue
5. Plot 2 is set active
6. Active plot (2) changes name to "Data"

Explicit Property Node
(use when subVIs are involved)



Property Nodes

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



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.

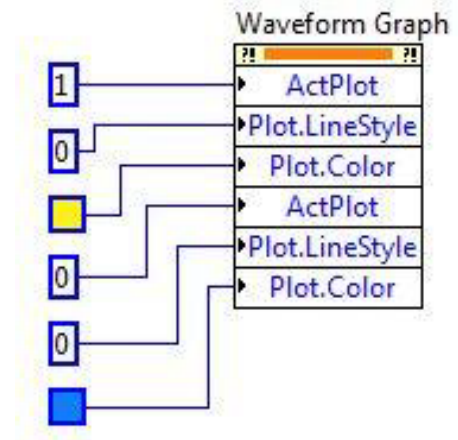
- a. Explicit; Implicit; Data Value References
- b. Implicit; Explicit; Data Value References
- c. Explicit; Implicit; Control References
- d. Implicit; Explicit; Control References

Justification: 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?

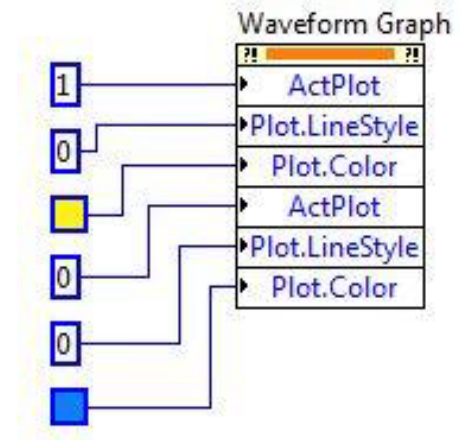
- a. Plot 1 because properties are executed top-down
- b. Plot 0 because properties are implemented in numeric order starting at zero
- c. Both plots will be updated simultaneously due to the multithreading of properties
- d. It cannot be determined because LabVIEW performs operations in dataflow format



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.

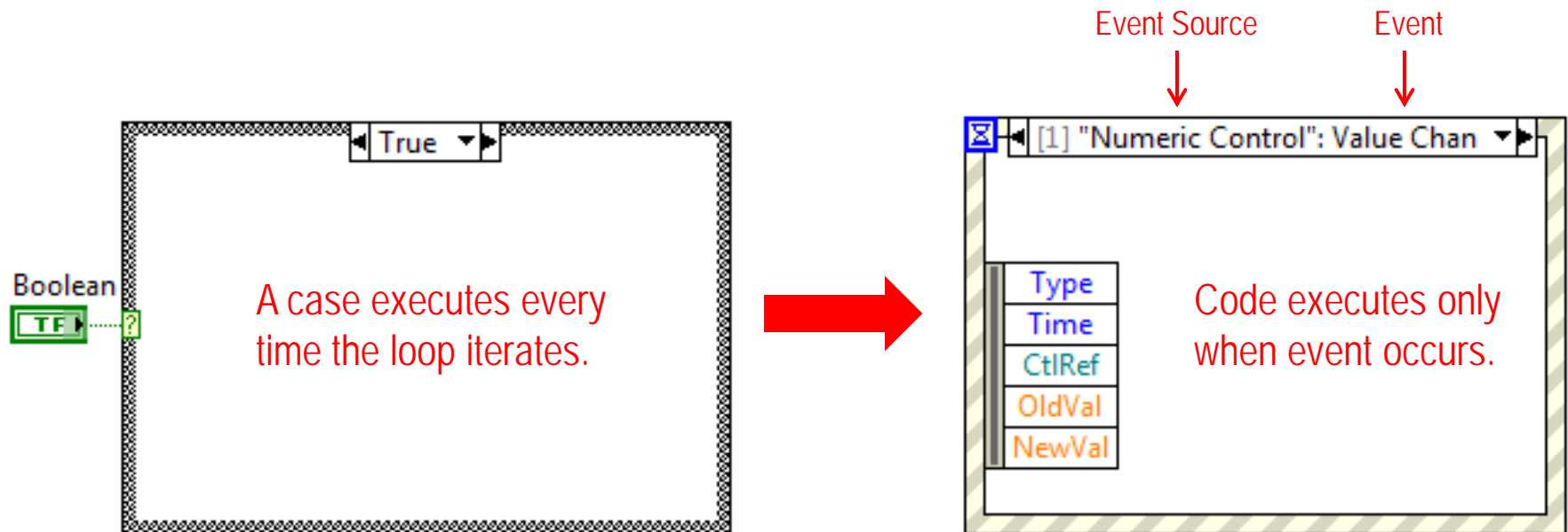
Property Nodes

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.

Event Structures

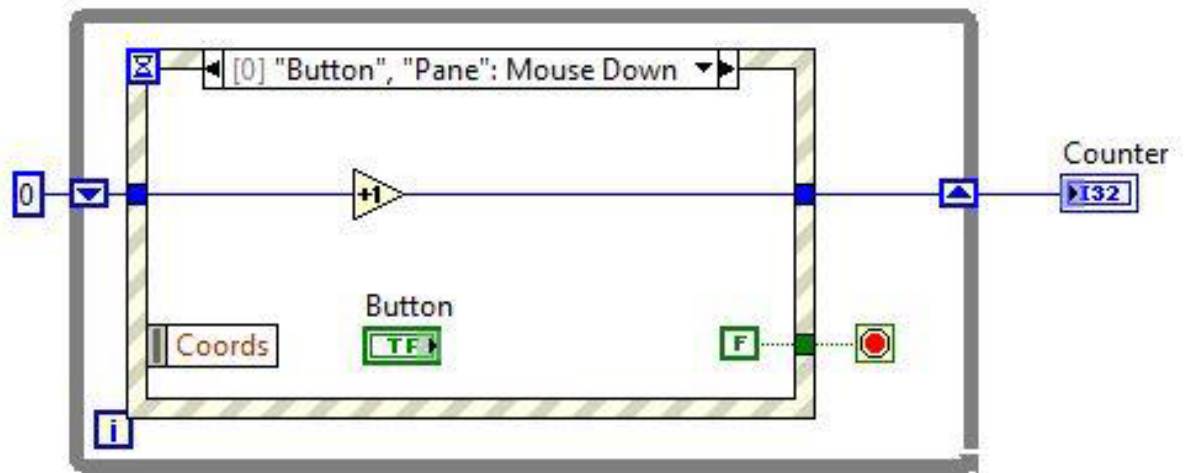
An Event structure works like a Case structure with a built-in Wait on Notification function.



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

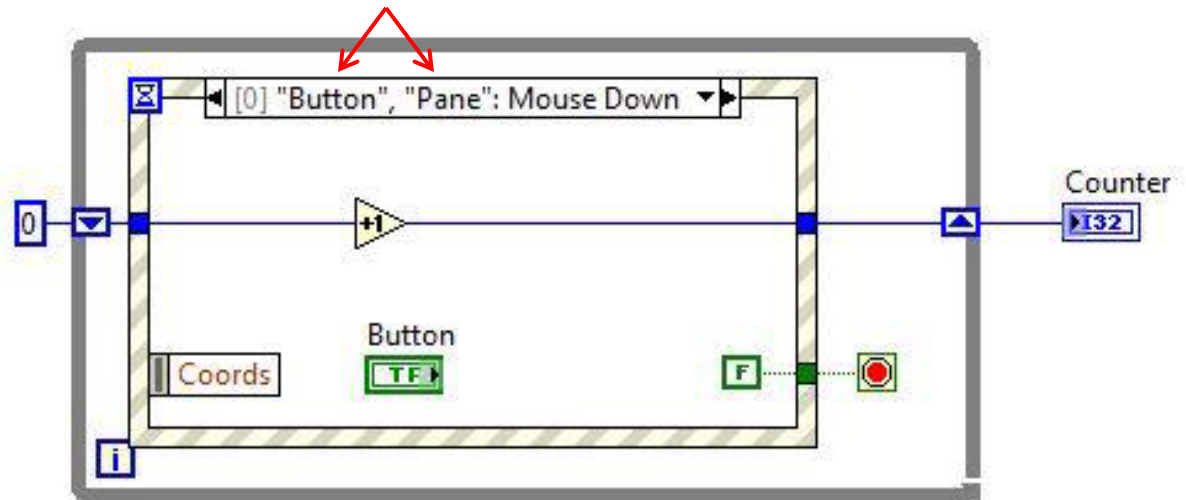


Event Structures

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

- a. 0
- b. 1
- c. 2
- d. 3

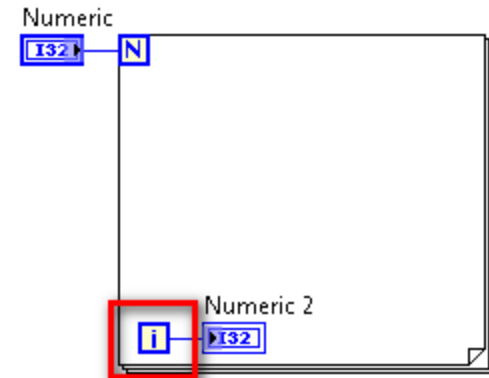
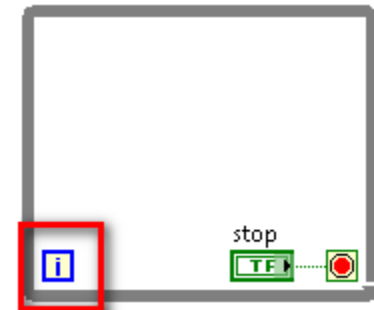
Because a Mouse Down event occurs on both the Button and the Pane, 2 events are registered. The code executes twice.



Loops

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

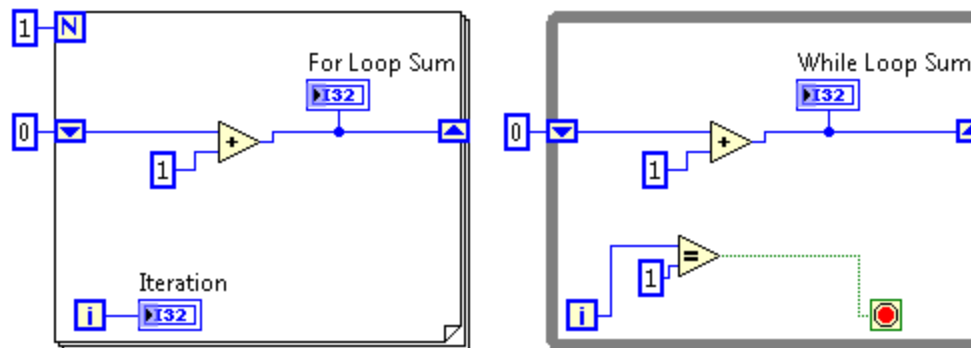
- While Loops must execute at least once
- For Loops can execute zero times
- The Iteration Terminal is zero indexed; this means the output of the terminal is zero for the first iteration of the loop.



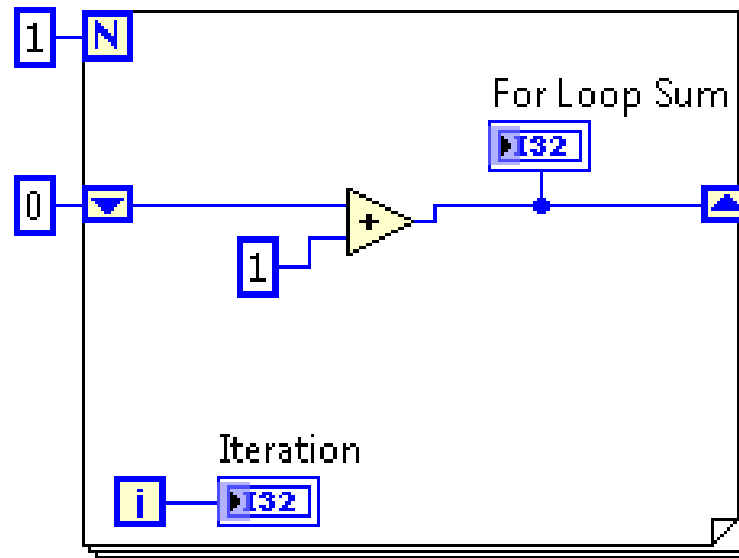
Loops

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

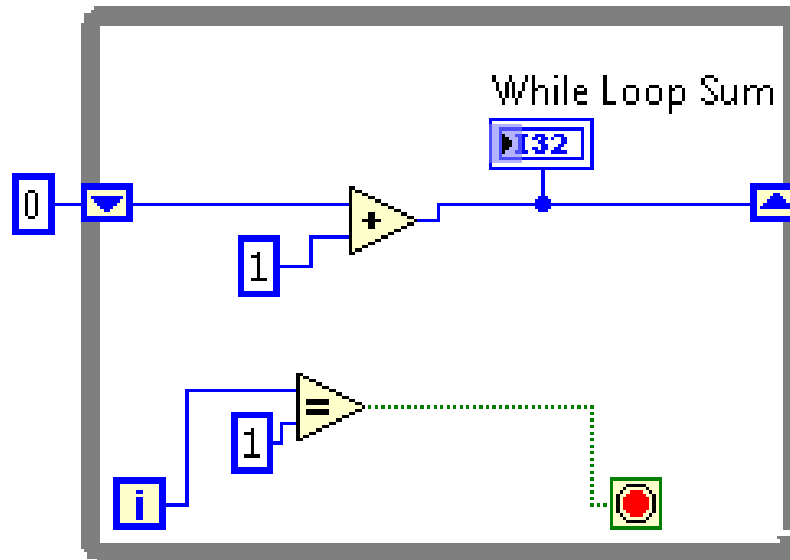


Loops



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>i</i> value	Stop condition met (i=1)?
1	0+1=1	1	0	no
2	1+1=2	2	1	yes

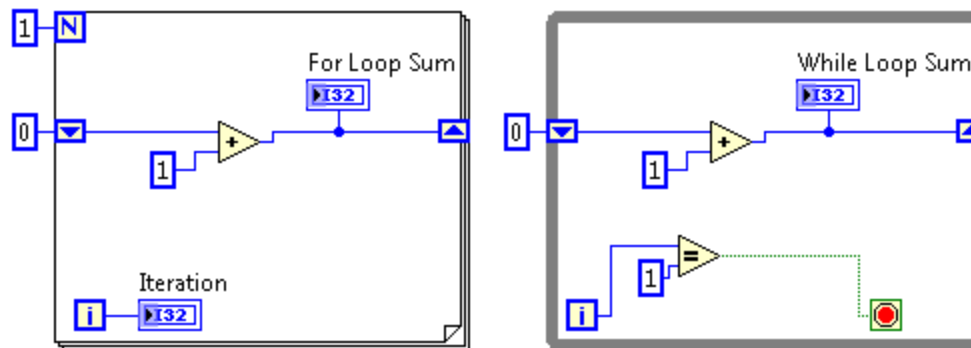
Loops

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



Charts and Graphs

Graphs

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

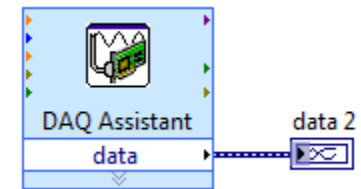
Charts

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

Both

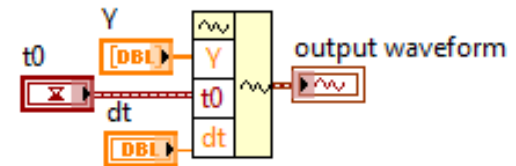
- Accept various data types:

- Waveform
- Dynamic
- Arrays

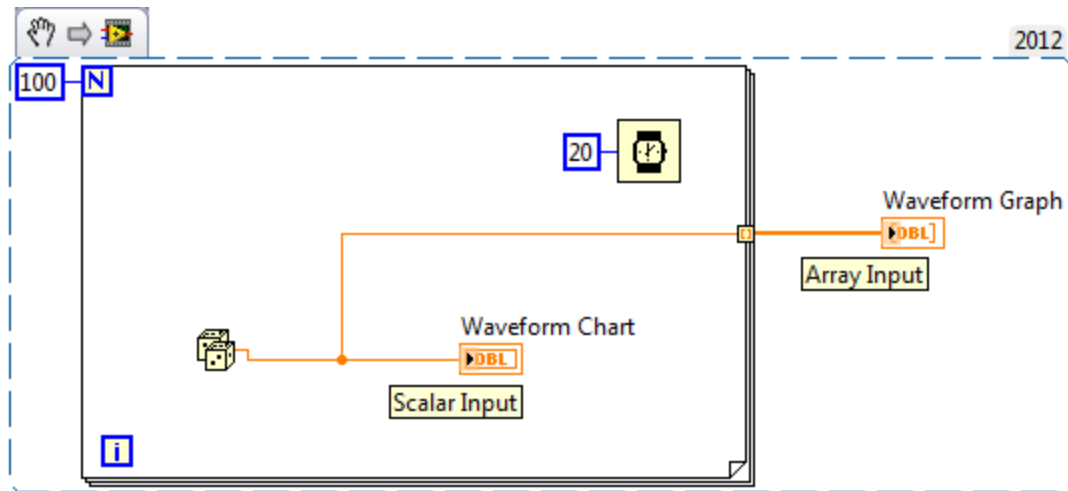


- Waveform Data types contain:

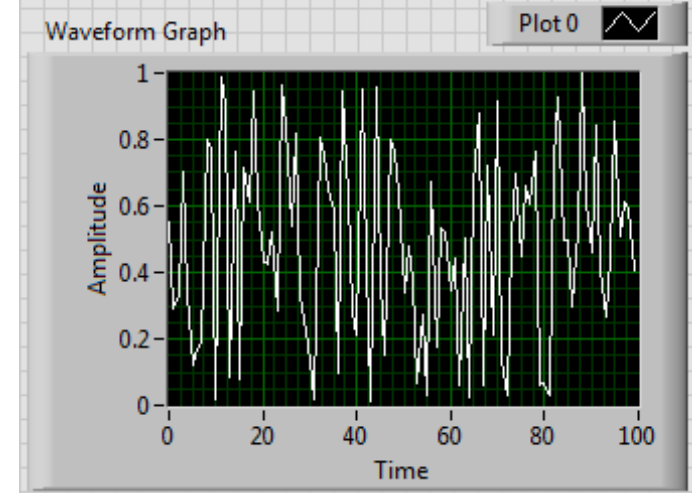
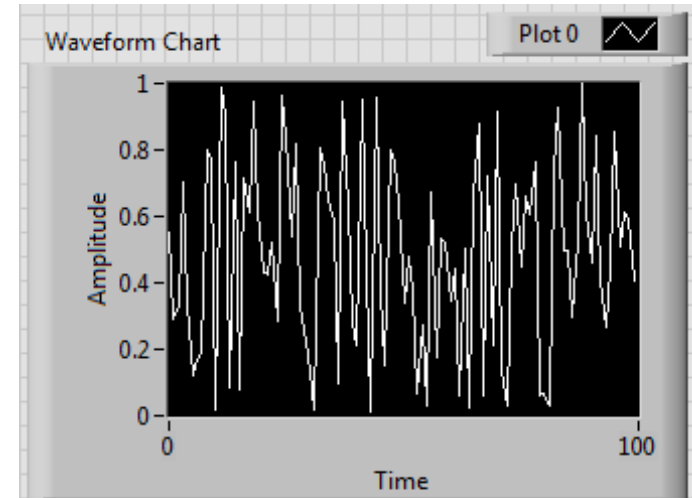
- An array of points
- t_0
- dt



Charts and Graphs

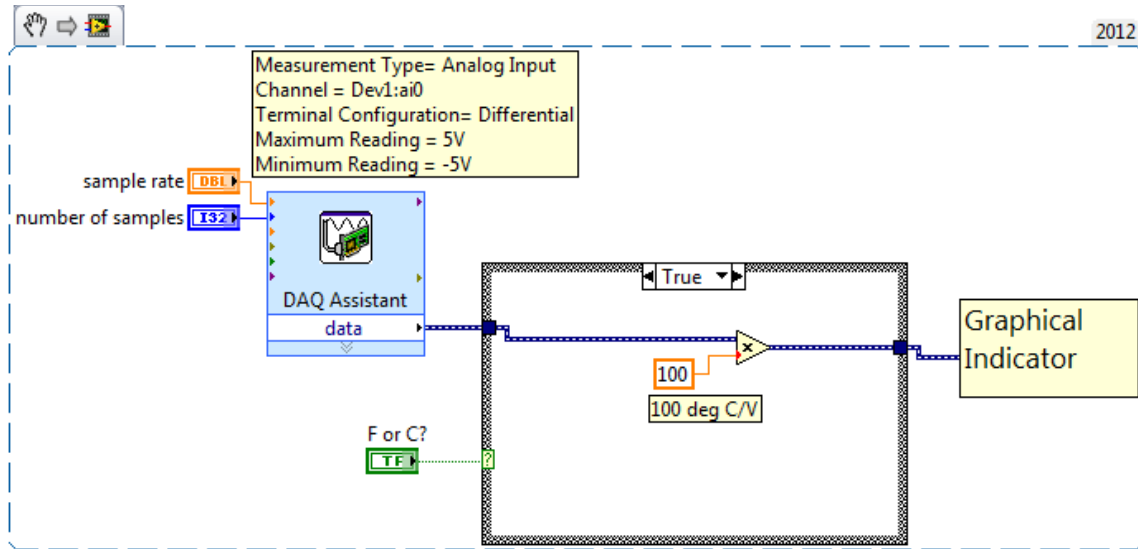


- The chart is inside the loop and updates for each iteration.
- The graph is outside the loop and only updates once: after the loop finishes.



Charts and Graphs

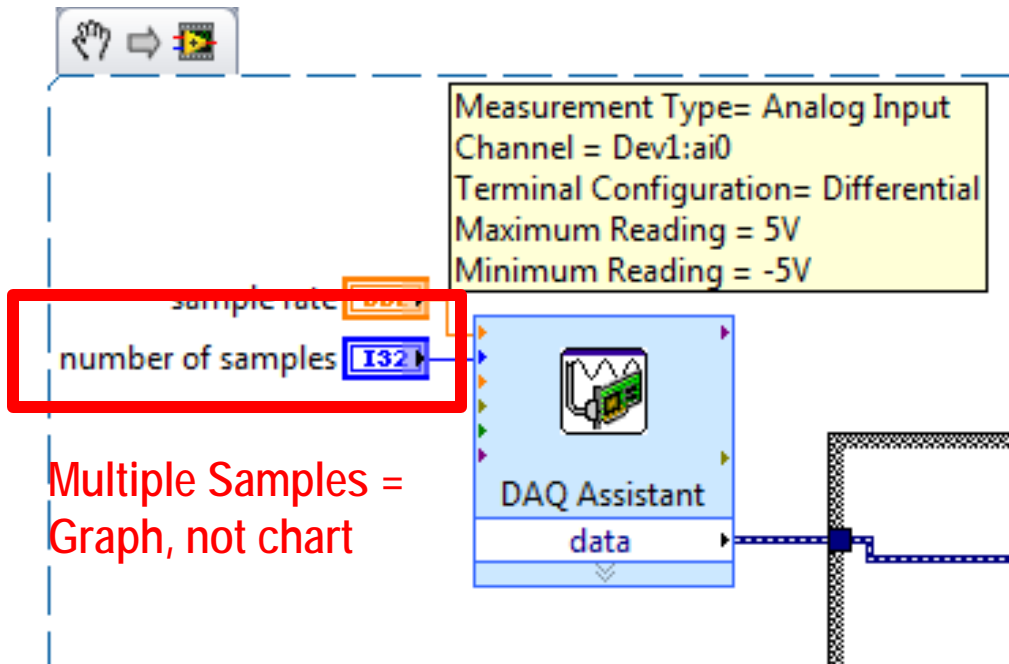
You need to add a graphical indicator to the location shown below. Which of the following is the best graphical indicator to use in this program?



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

Charts and Graphs

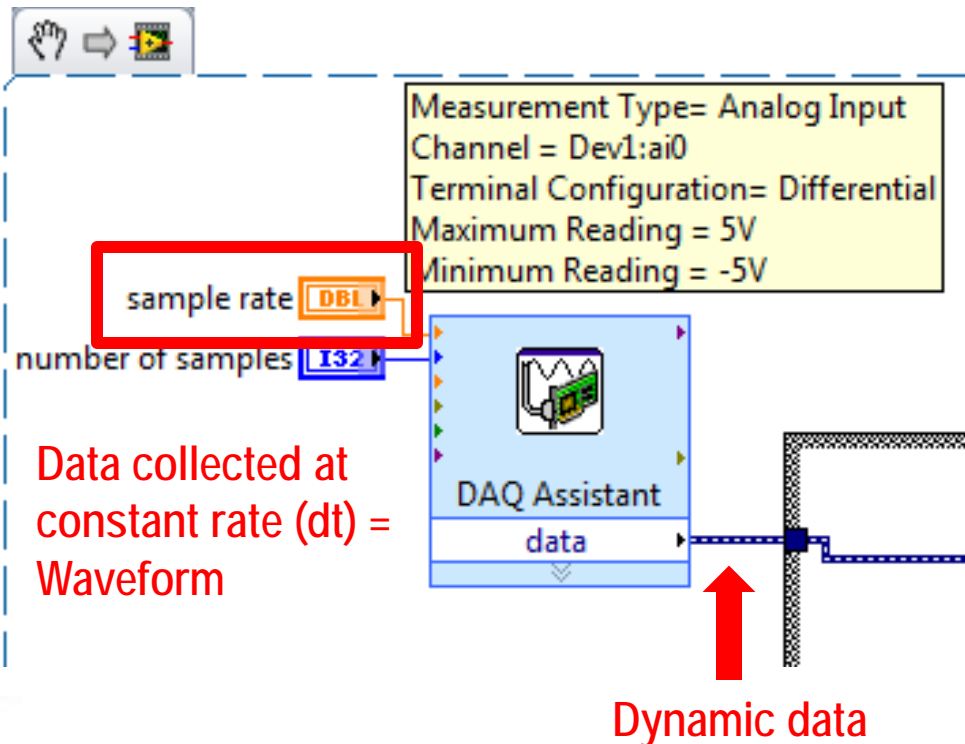
You need to add a graphical indicator to the location shown below. Which of the following is the best graphical indicator to use in this program?



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

Charts and Graphs

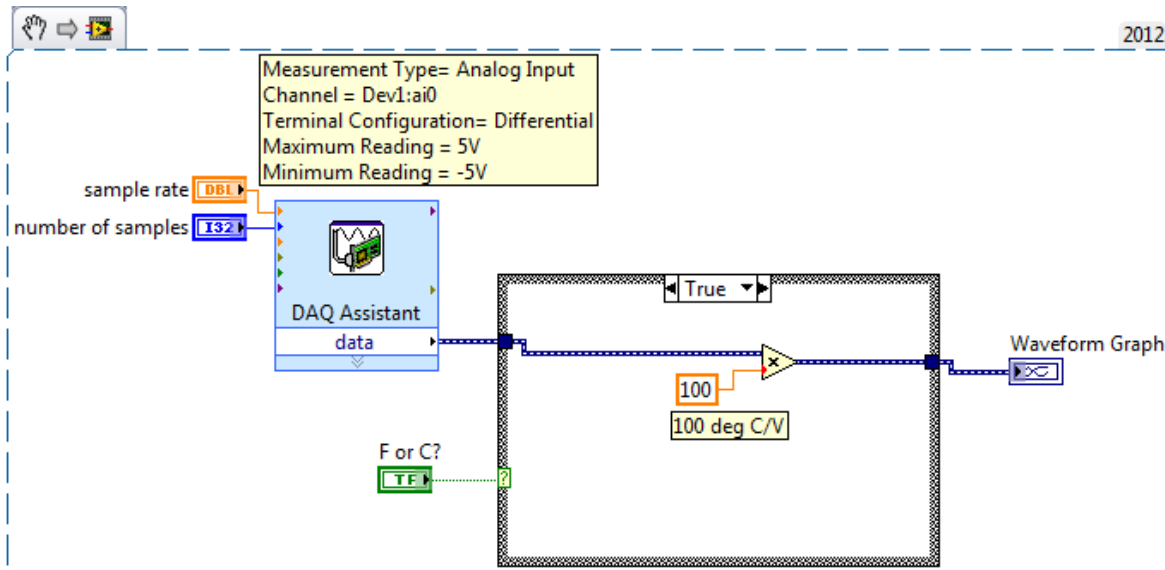
You need to add a graphical indicator to the location shown below. Which of the following is the best graphical indicator to use in this program?



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

Charts and Graphs

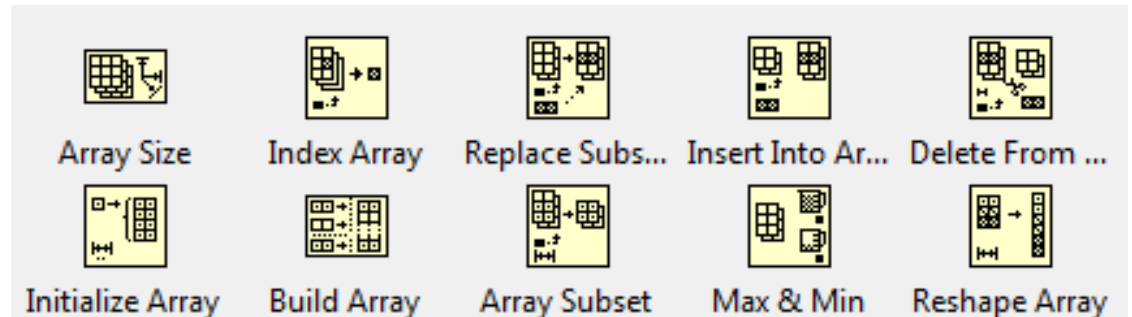
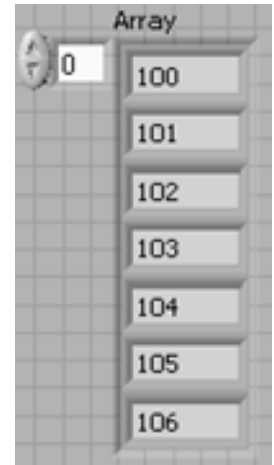
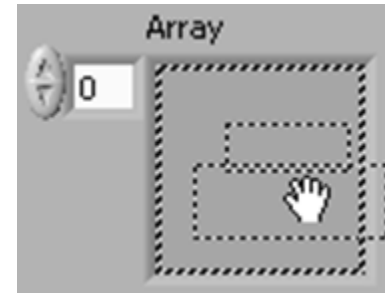
You need to add a graphical indicator to the location shown below. Which of the following is the best graphical indicator to use in this program?



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

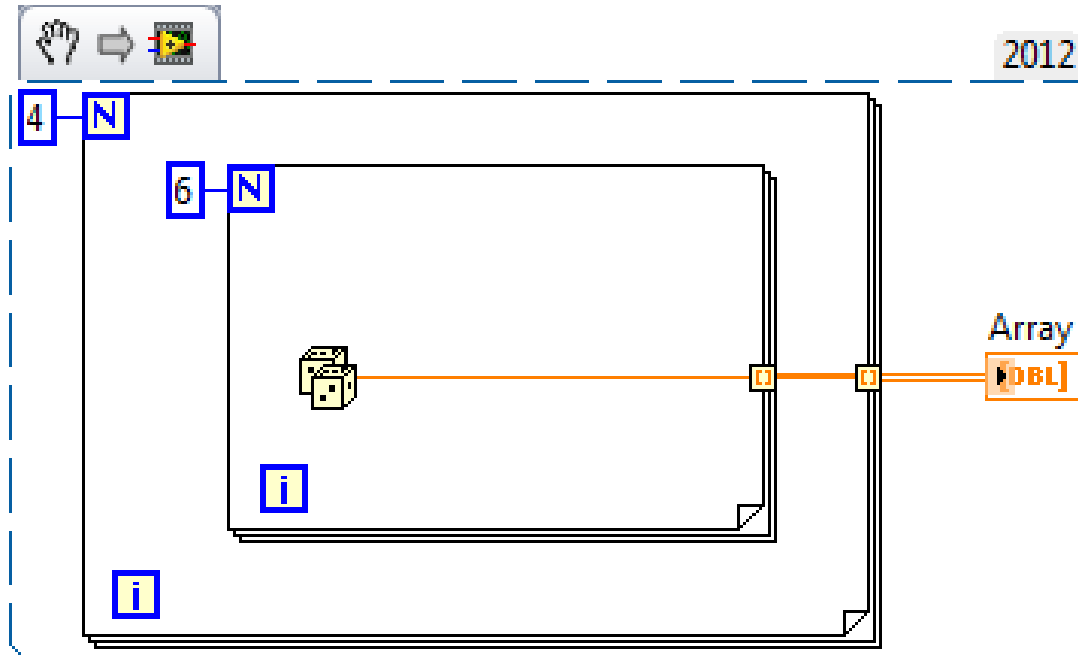
Array Functions

- 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

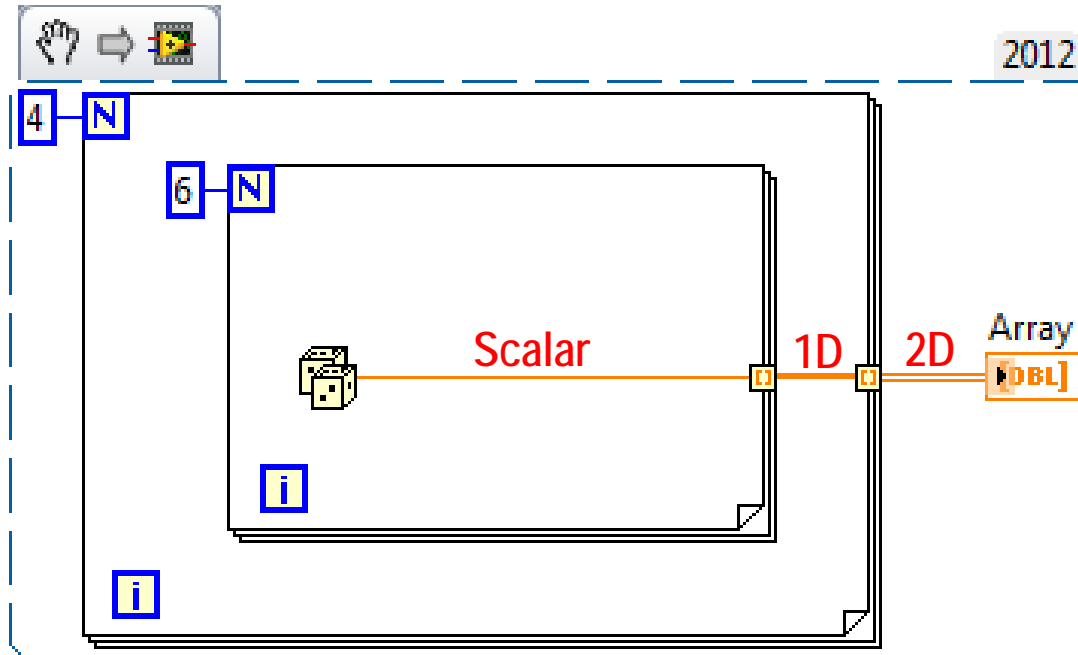
Describe the array that results from running this code.



- 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.

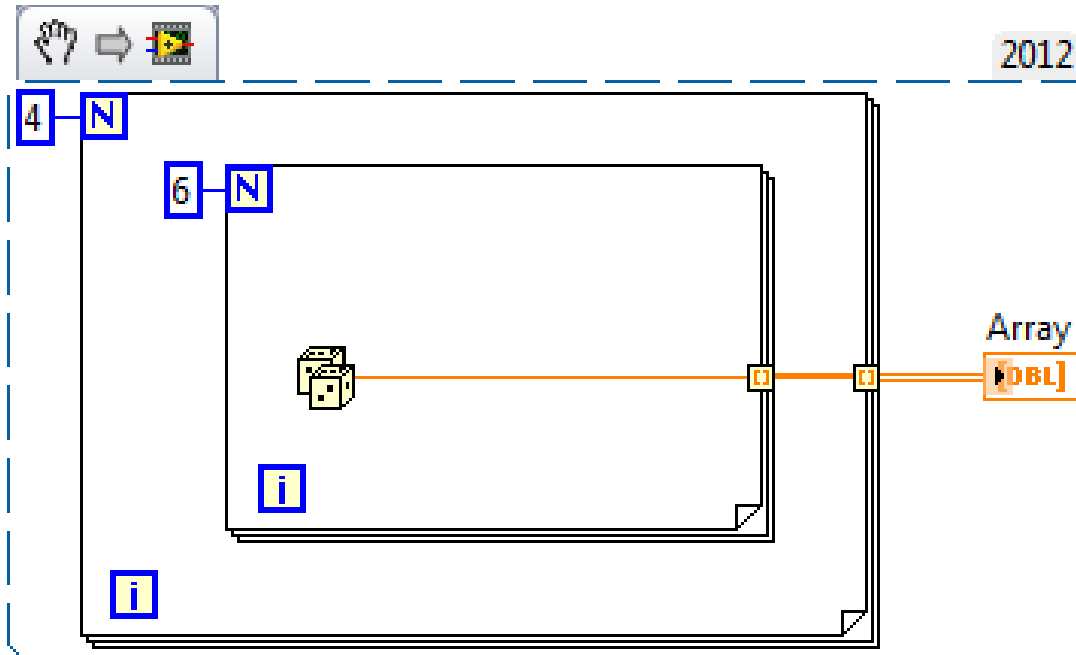


Two loops means a
2D array.

- ~~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

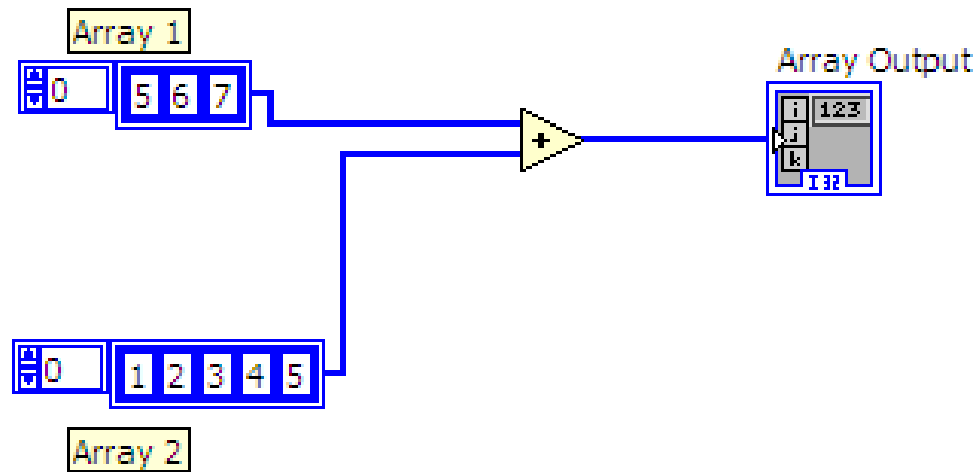
Describe the array that results from running this code.



- 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

Array Functions

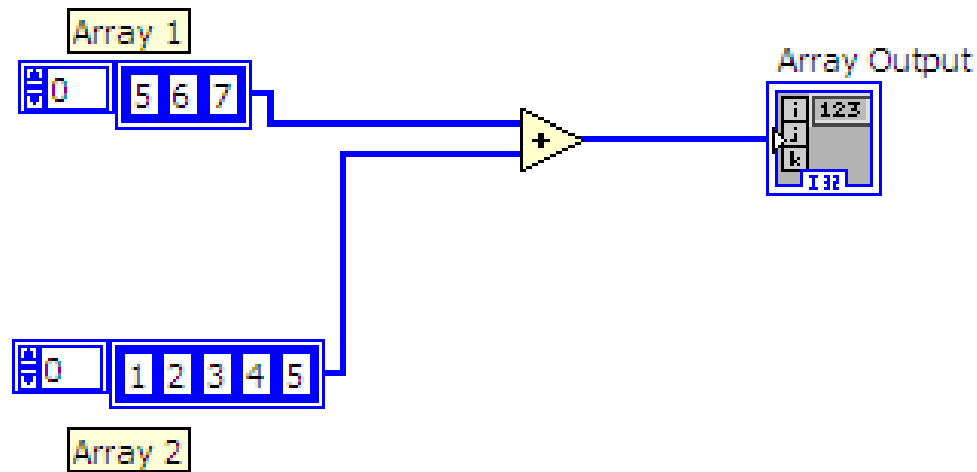
What is the result of the following Array addition?



- 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?

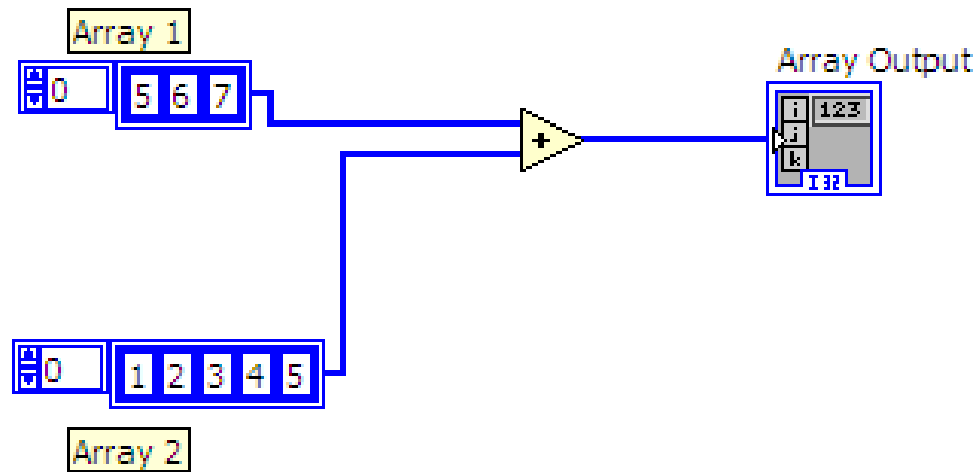


These are not valid
arrays- the row sizes
are not the same

- 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?

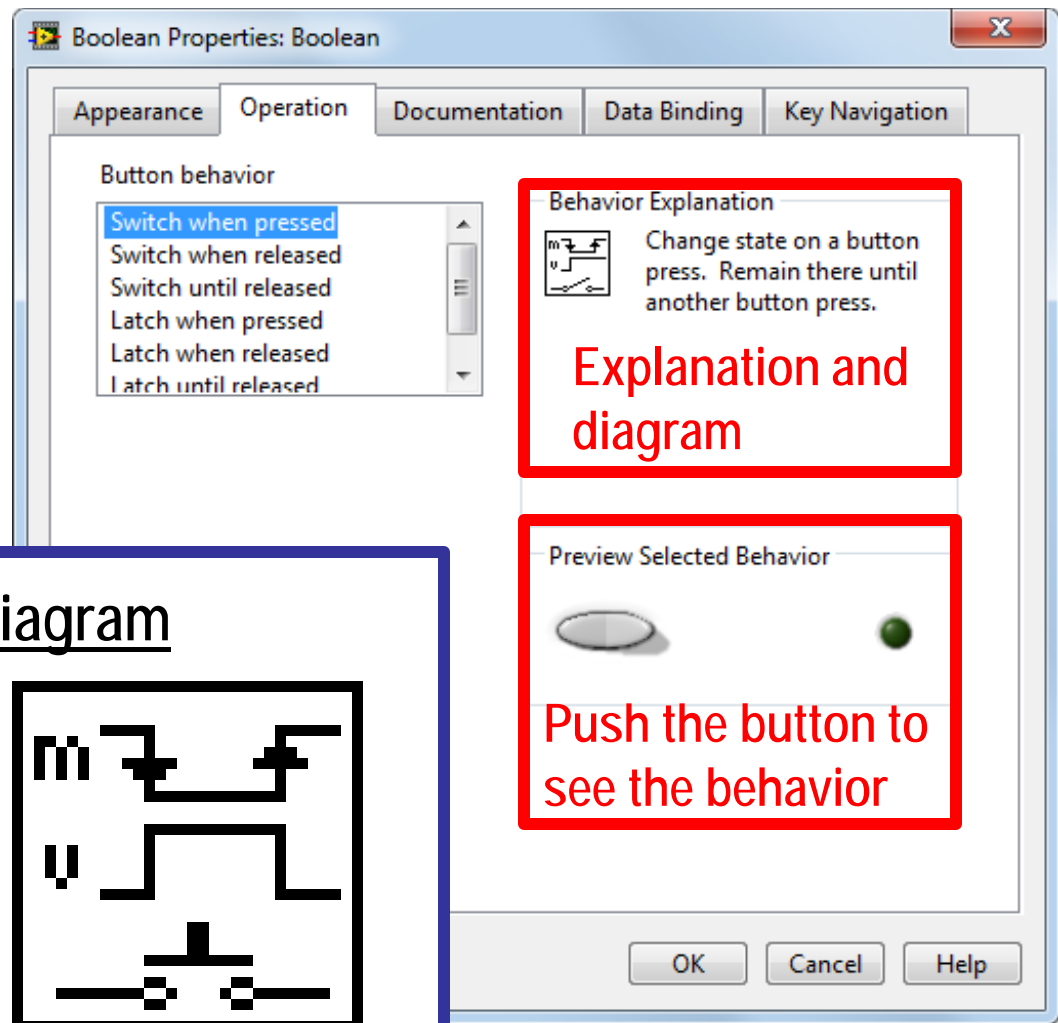


Output is the same size as the smaller input.

- 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}}

Mechanical Action of Booleans

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



How to Read the Diagram

- Button Position →
- "Voltage" at LED →
- Circuit Diagram Symbol →

