# Simulink Modeling Techniques

Lecture 68, v01

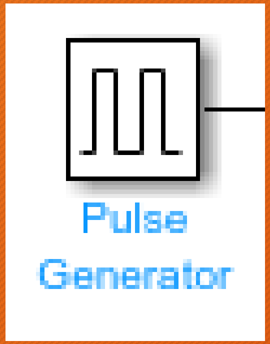
A description of basic techniques useful in developing Simulink models

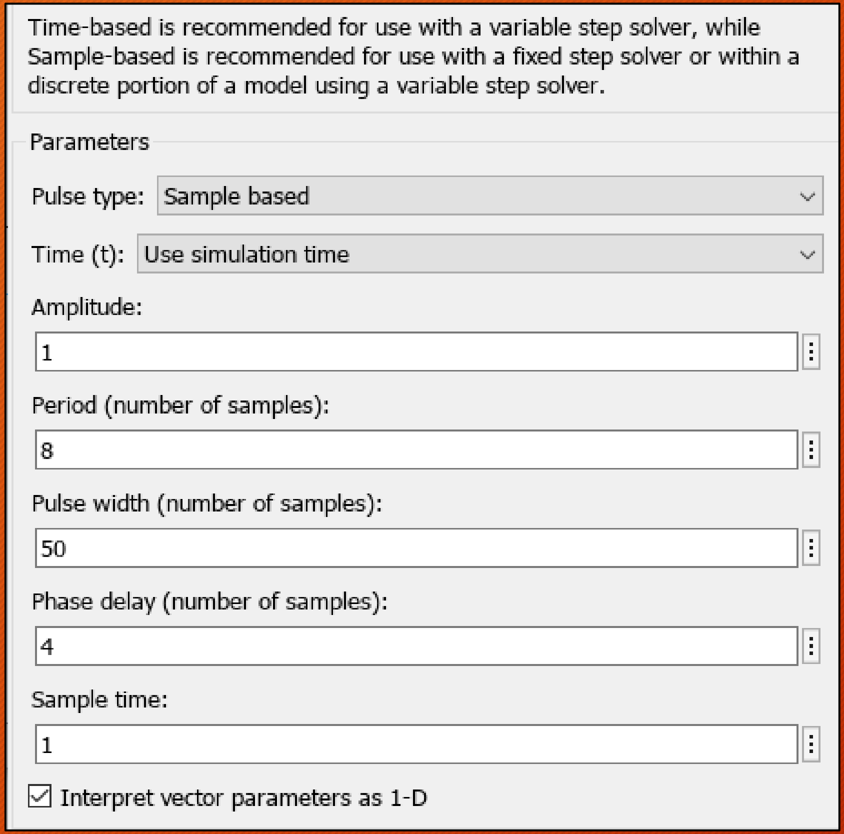
# Index

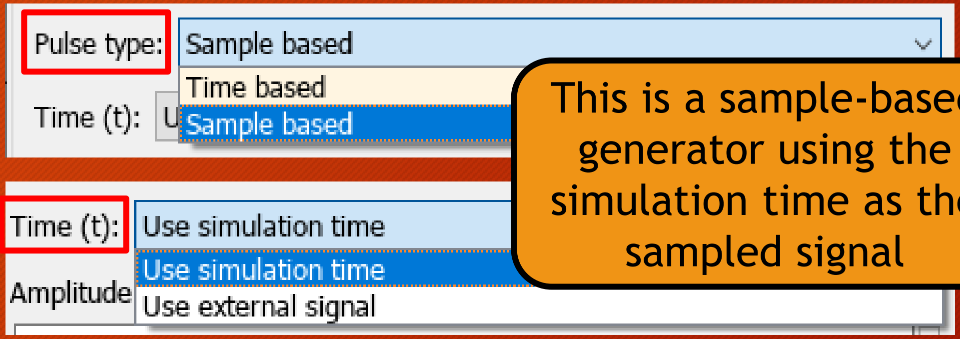
# Using the Simulink Pulse Generator to Generate a Pulse Stream

# Pulse Stream Signal

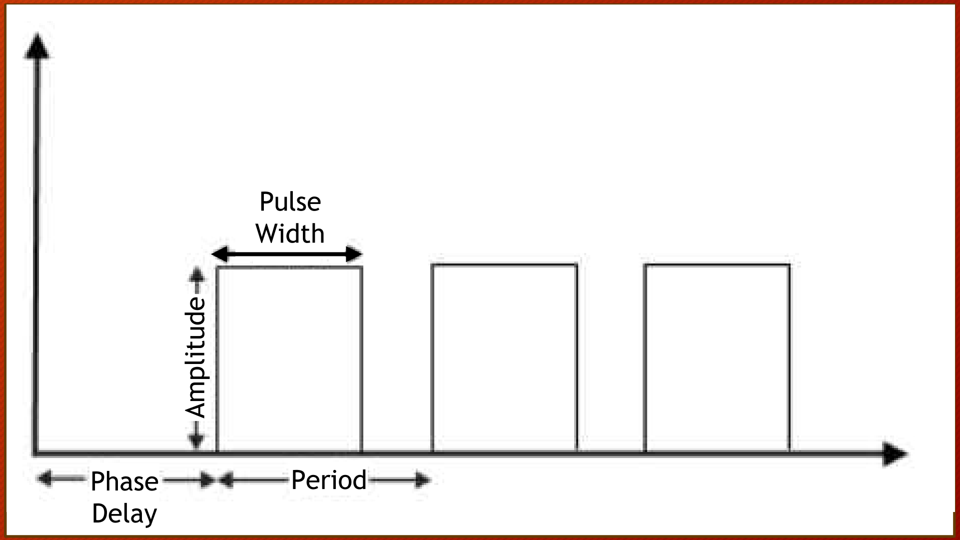
- There are several ways to generate a pulse stream, depending on the characteristics of the pulse desired
- The most basic way to generate a pulse stream is to use Simulink's own Pulse Generator block



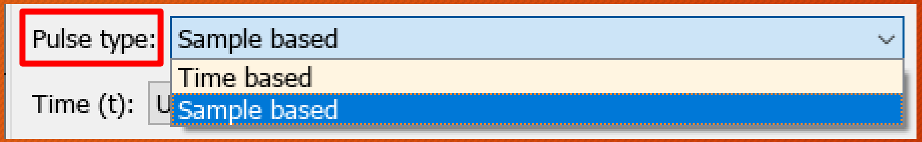




This is a sample-based generator using the simulation time as the sampled signal



# Pulse Type in Pulse Generator Block

- The Pulse Generator block output can be generated in time-based or sample-based modes, determined by the Pulse type parameter

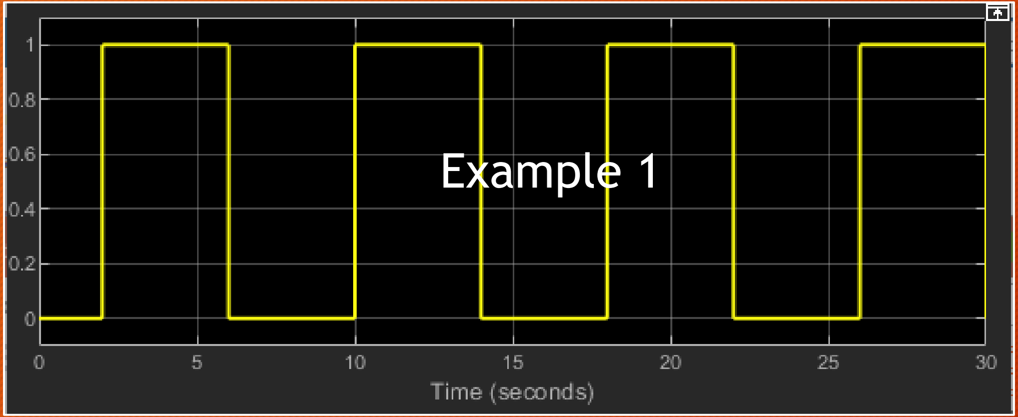


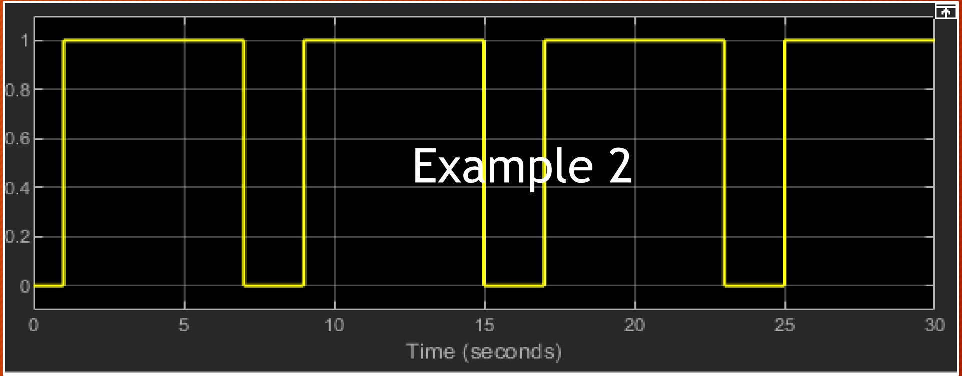

  - Time-based is recommended for use with a variable step solver
  - Sample-based is recommended for use with a fixed step solver
- In time-based mode, the block output is based on simulation time
  - Simulink computes the block output only at times when the output actually changes
  - This approach results in fewer computations for the block output over the simulation time period
- In sample-based mode, the block computes its outputs at fixed intervals that you specify
  - The block output depends only on the simulation start, regardless of elapsed simulation time
- This block supports reset semantics in sample-based mode
  - For example, if a Pulse Generator block is in a resettable subsystem that hits a reset trigger, the block output resets to its initial condition

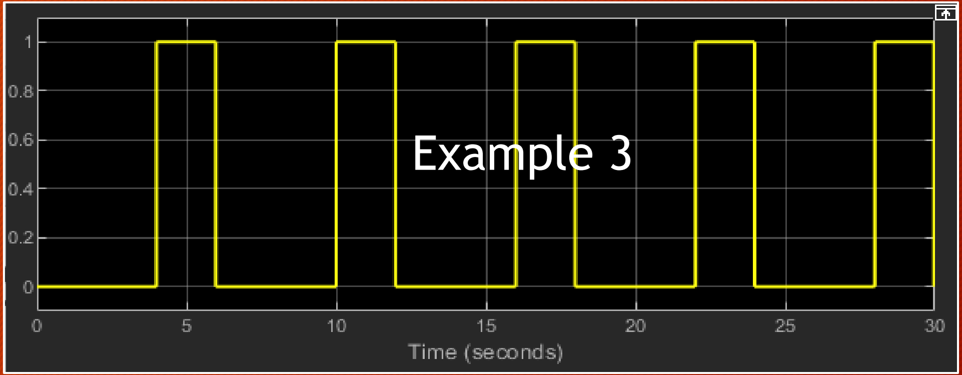
# Examples using the Pulse Generator Block

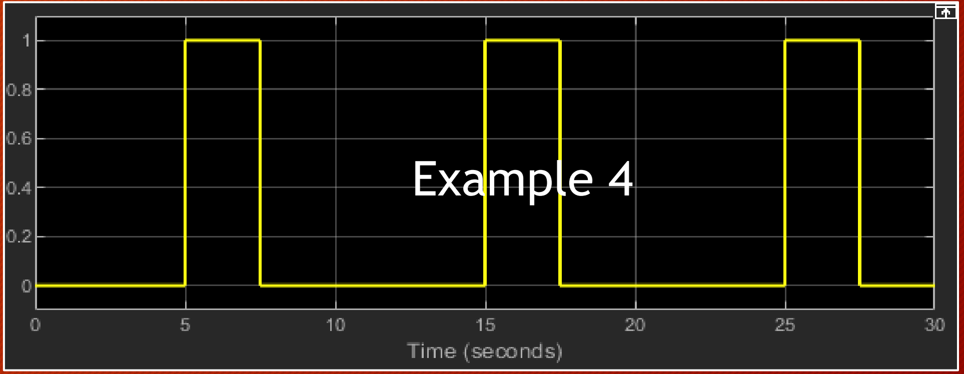
## Time-based using a variable step solver

| | Ex. 1 | Ex. 2 | Ex. 3 | Ex. 4 |
|---|---|---|---|---|
| Amplitude | 1 | 1 | 1 | 1 |
| Period | 8 | 8 | 6 | 10 |
| Pulse Width | 50 | 75 | 33 | 25 |
| Phase Delay | 2 | 1 | 4 | 5 |
| Sample Time | NA | N/A | N/A | N/A |



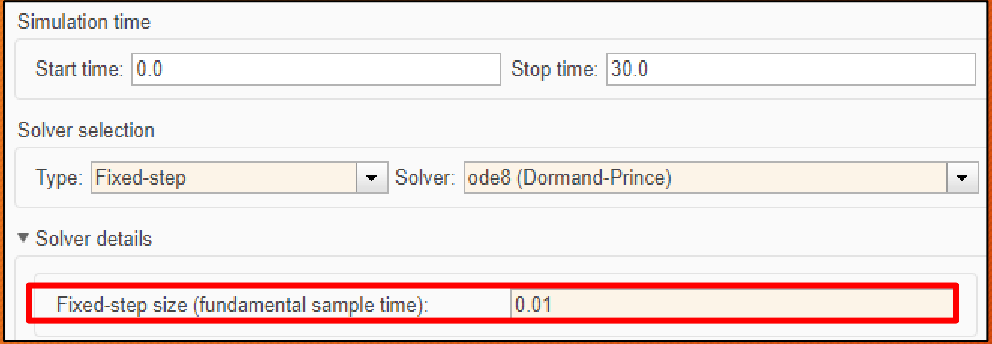



# Examples using Pulse Generator Block

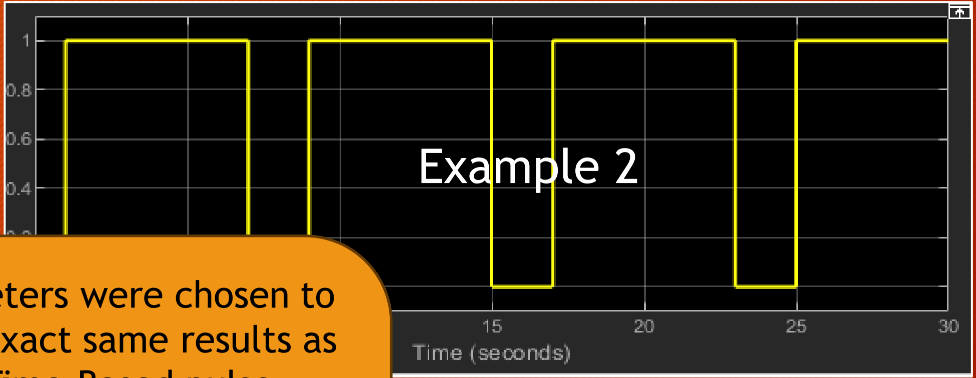
## Sample-based using a fixed step solver (0.01 sec step size)

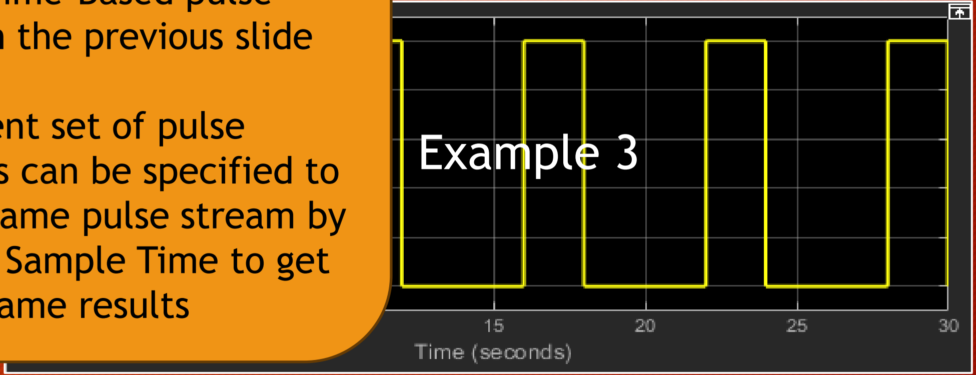


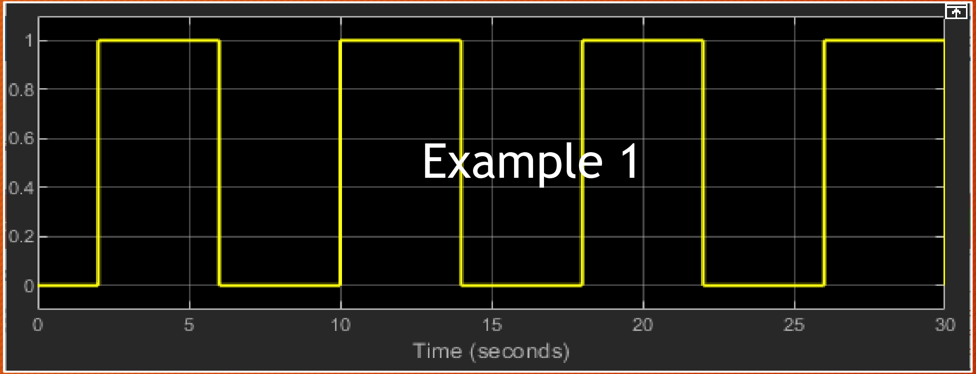

| | Ex. 1 | Ex. 2 | Ex. 3 | Ex. 4 |
|---|---|---|---|---|
| Amplitude | 1 | 1 | 1 | 1 |
| Period | 80 | 800 | 6 | 100 |
| Pulse Width | 40 | 600 | 2 | 25 |
| Phase Delay | 20 | 100 | 4 | 50 |
| Sample Time | 0.1 | 0.01 | 1 | 0.1 |

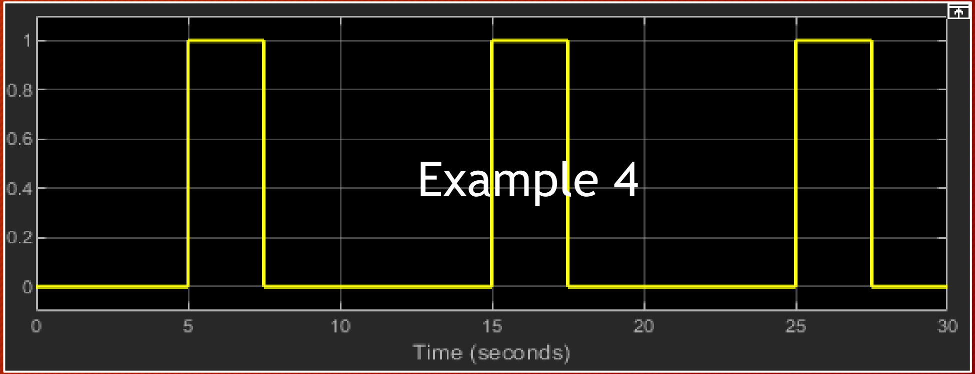
These parameters were chosen to produce the exact same results as with the Time-Based pulse generator on the previous slide

A different set of pulse characteristics can be specified to generate the same pulse stream by adjusting the Sample Time to get the same results



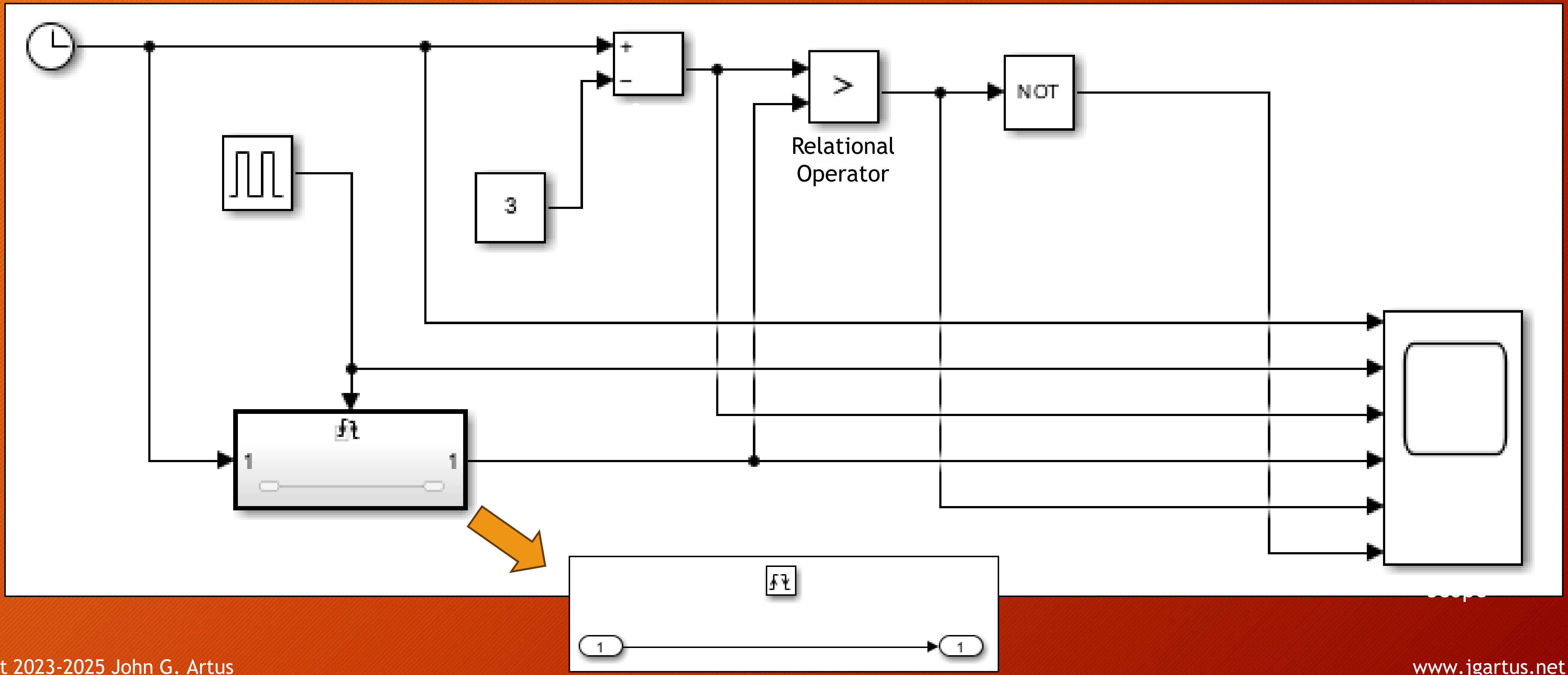



# Constructing a Clock-Based Pulse Generator

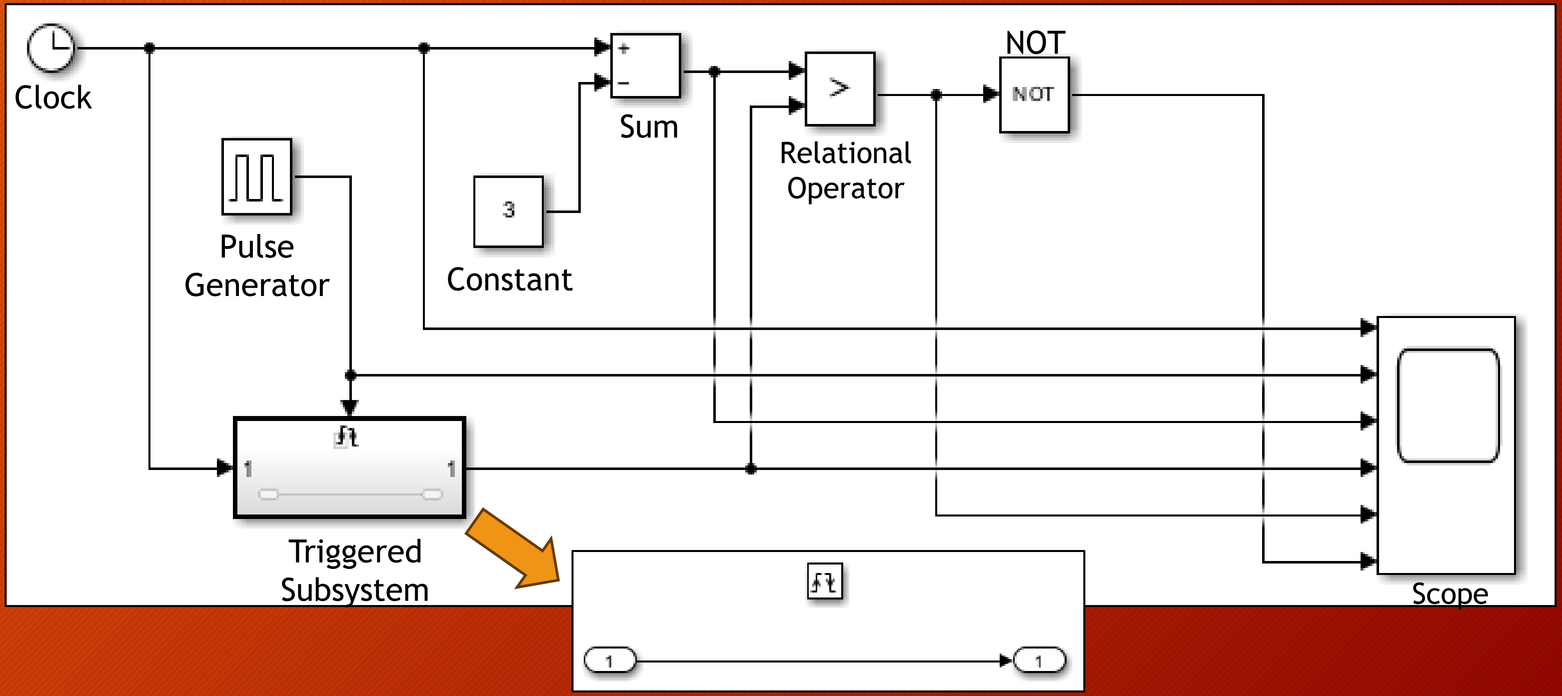
# Constructing a Clock-Based Pulse Generator

- A pulse can also be generated by using other Simulink elements with, perhaps, different kinds of control over the output signal generated
- Here is shown such an example based on a linear ramp signal from a clock



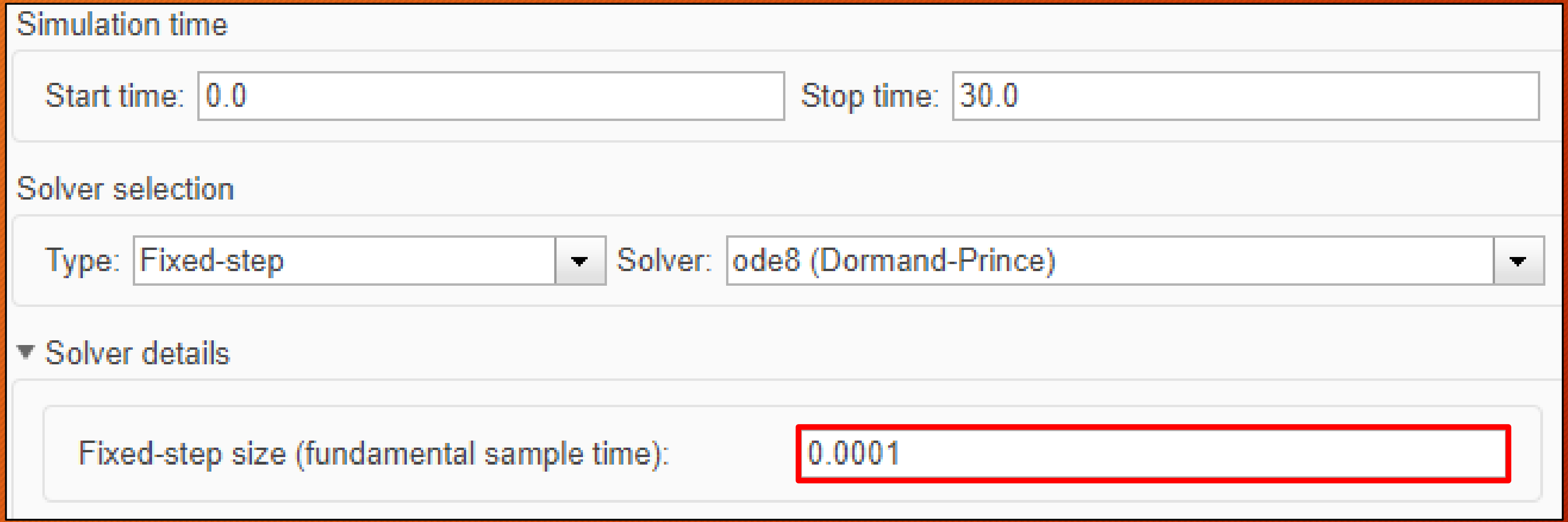
# Clock-Based Pulse Generator Overview

- In this system, a clock generates a linear ramp signal starting at 0
- A pulse generator feeds a sampling trigger signal, that triggers the sampling of the clock signal on both the rising side as well as the falling side of the pulse
- A constant value is subtracted from the clock signal, essential creating a signal that is 3 seconds behind the clock signal
- The two signals are fed into a “greater than” relational operator



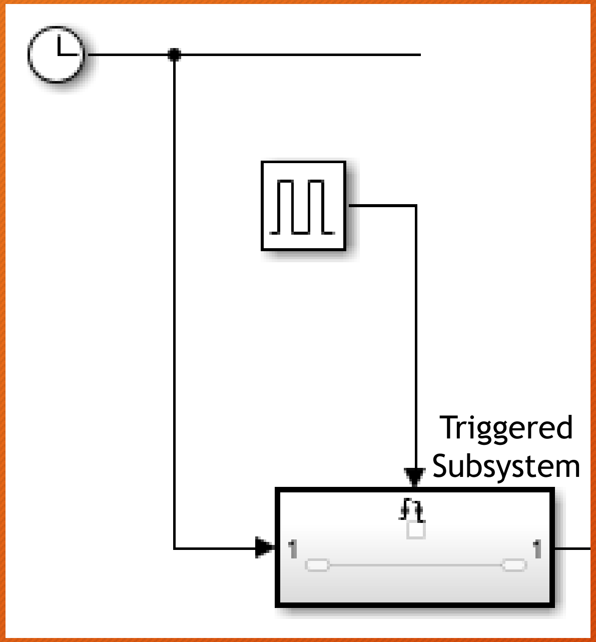
# Constructing a Clock-Based Pulse Generator (continued)

- This model uses a fixed-step solver with a step size of 0.0001 seconds

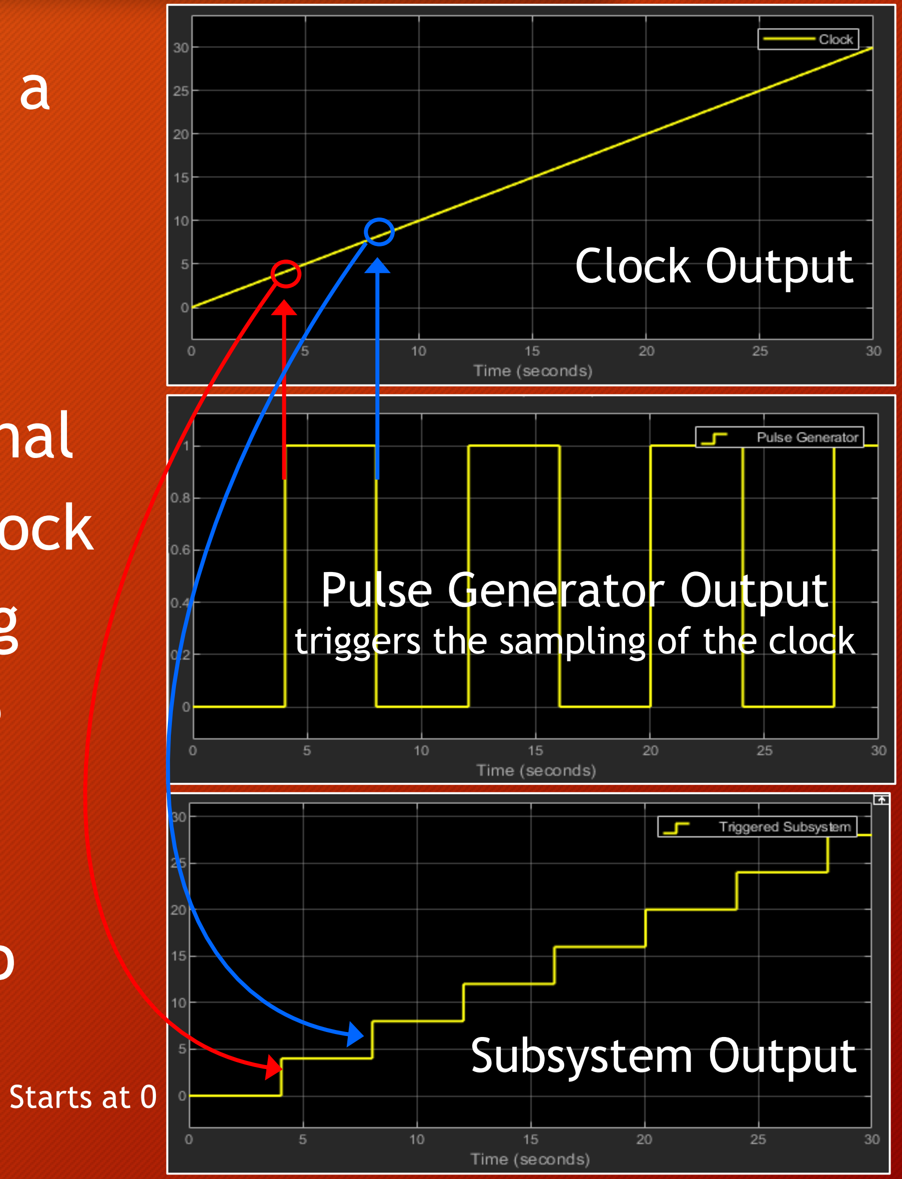



- A pulse generator drives the sampling of the clock signal
  - The pulse generator is time-based that operates with a period of 8 seconds
- A triggered subsystem simply allows the clock signal to be sampled on both the rising and falling sides of the pule from the pulse generator

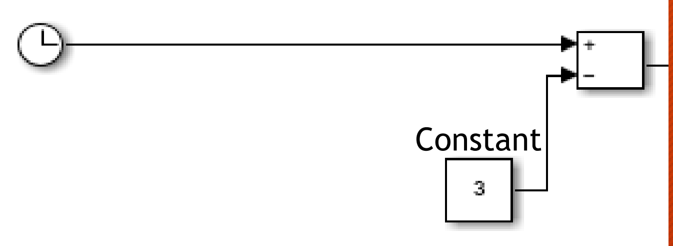
# Constructing a Clock-Based Pulse Generator (continued)

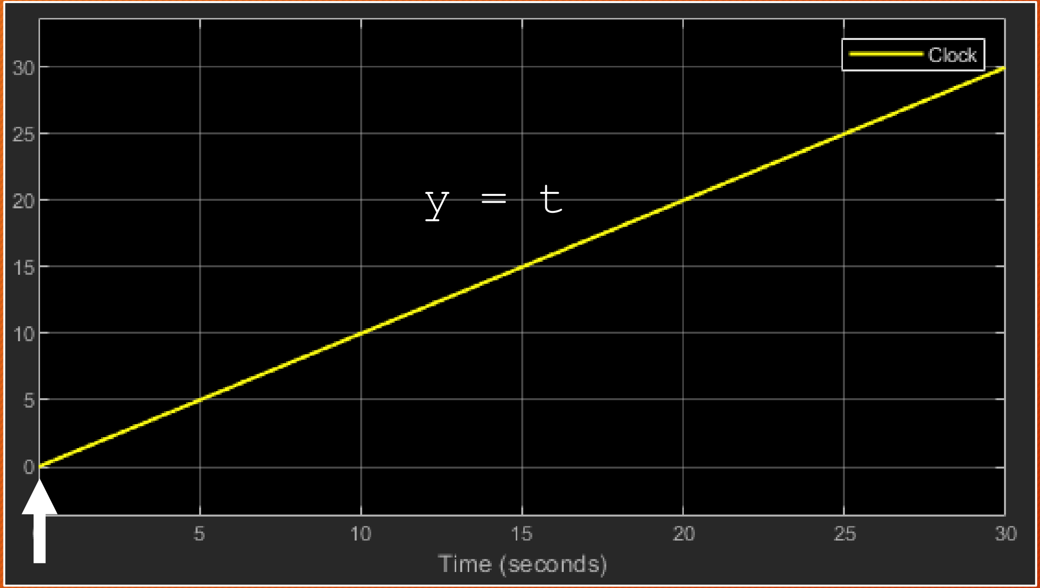



- The clock block outputs a linear ramp starting at zero
- The pulse generator signal is used to sample the clock signal for both the rising side and the falling side
- The result is a stair-step signal starting at zero



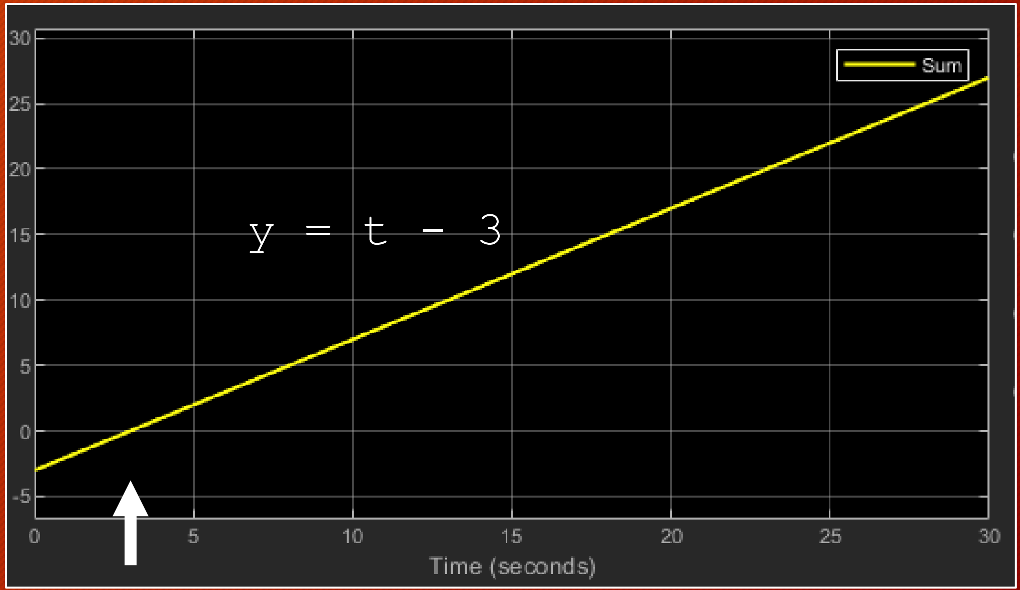
- A sum block is used to subtract 3 from the clock signal
- This has the result of generating a signal that is three seconds behind the clock signal




Zero-Crossing

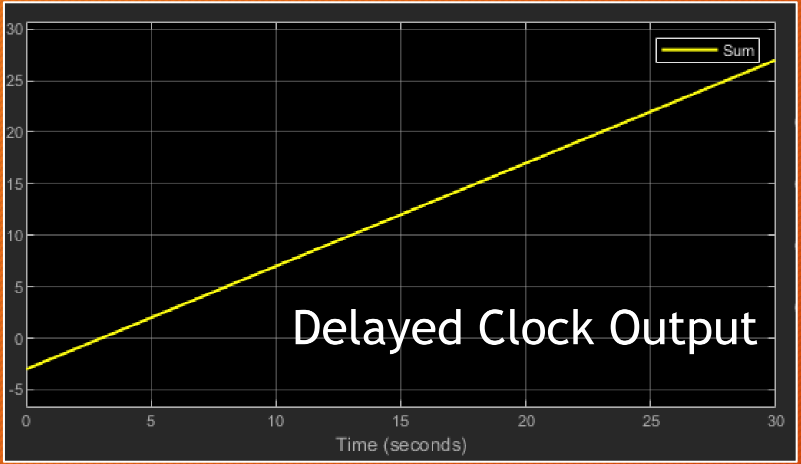
– 3 =

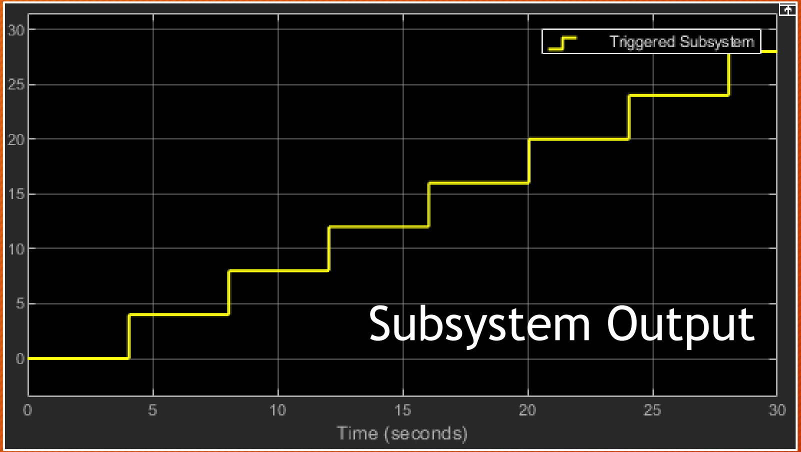



Zero-Crossing

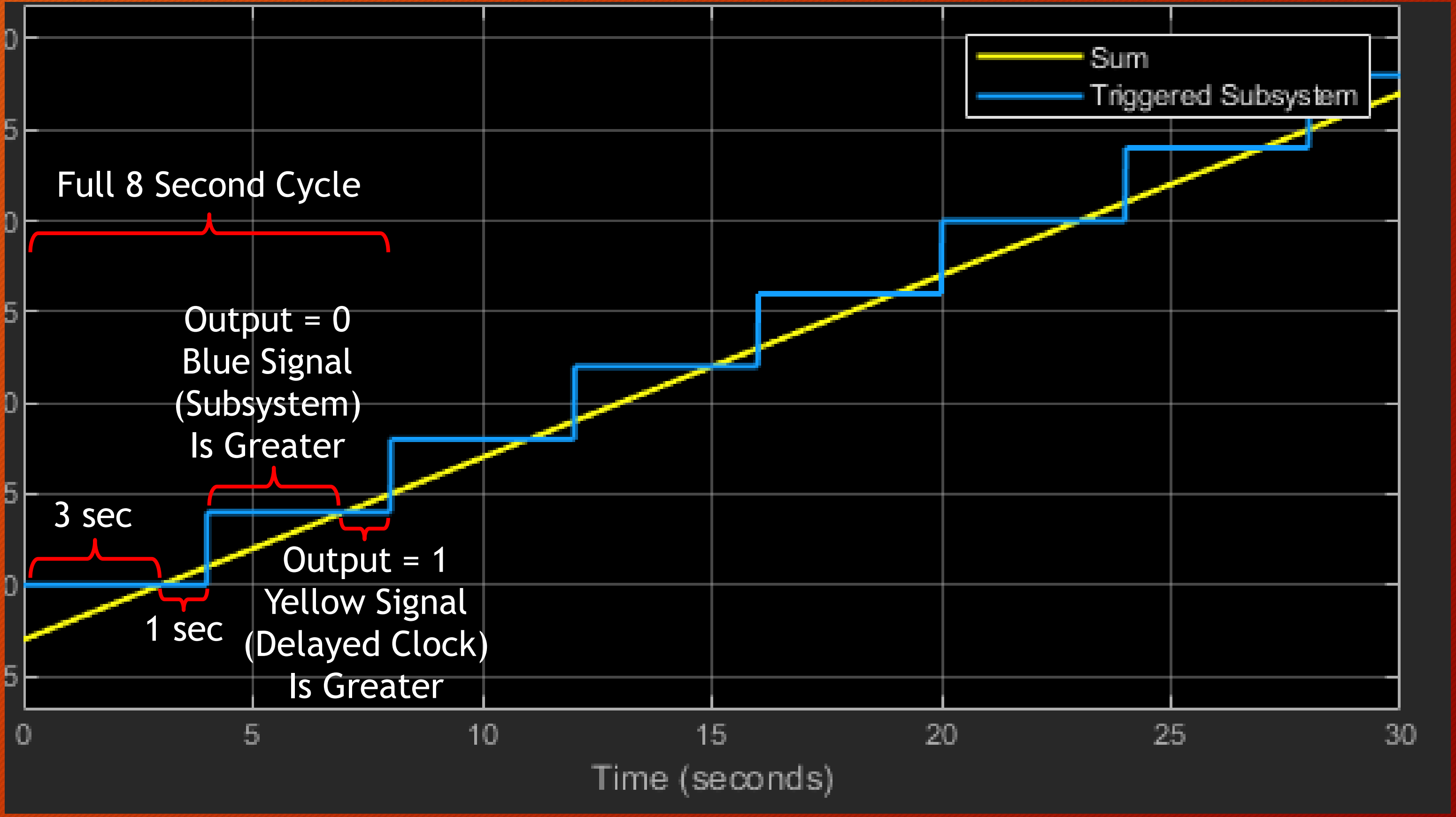
# Constructing a Clock-Based Pulse Generator (continued)

- The relational operator outputs a 1 if the first (top) input is greater than the second (lower) input
- Otherwise, it outputs a 0



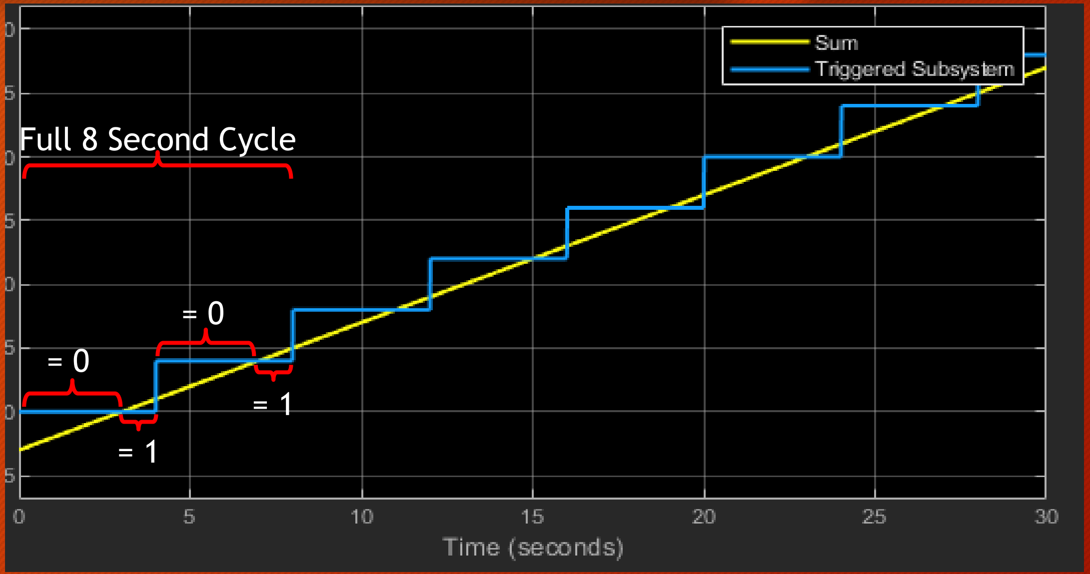



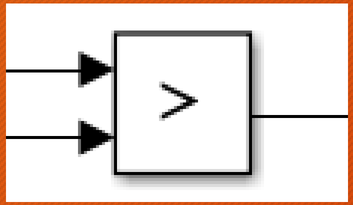

Signals Overlapped for Comparison



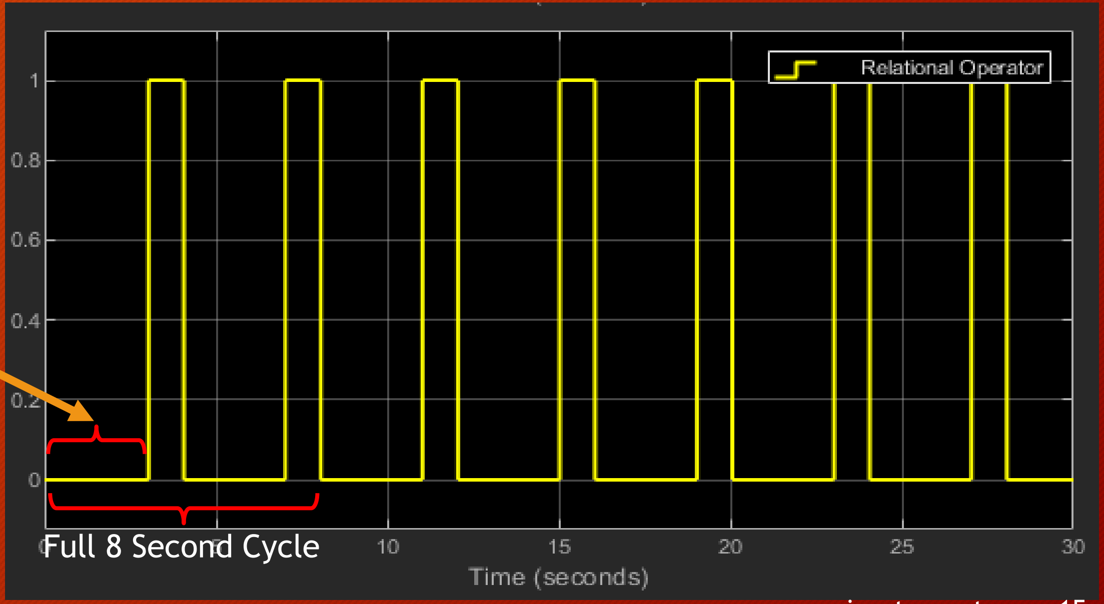
# Constructing a Clock-Based Pulse Generator (continued)

- The relational operator outputs a 1 if the first (top) input is greater than the second (lower) input
- Otherwise, it outputs a 0



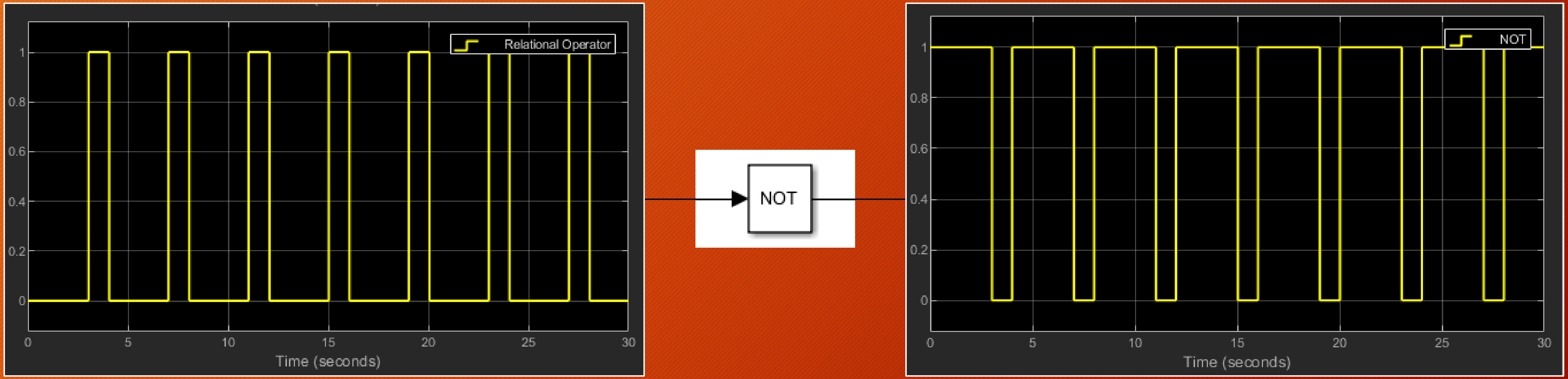




Controlled by constant block (3 sec)



# Constructing a Clock-Based Pulse Generator continued) (

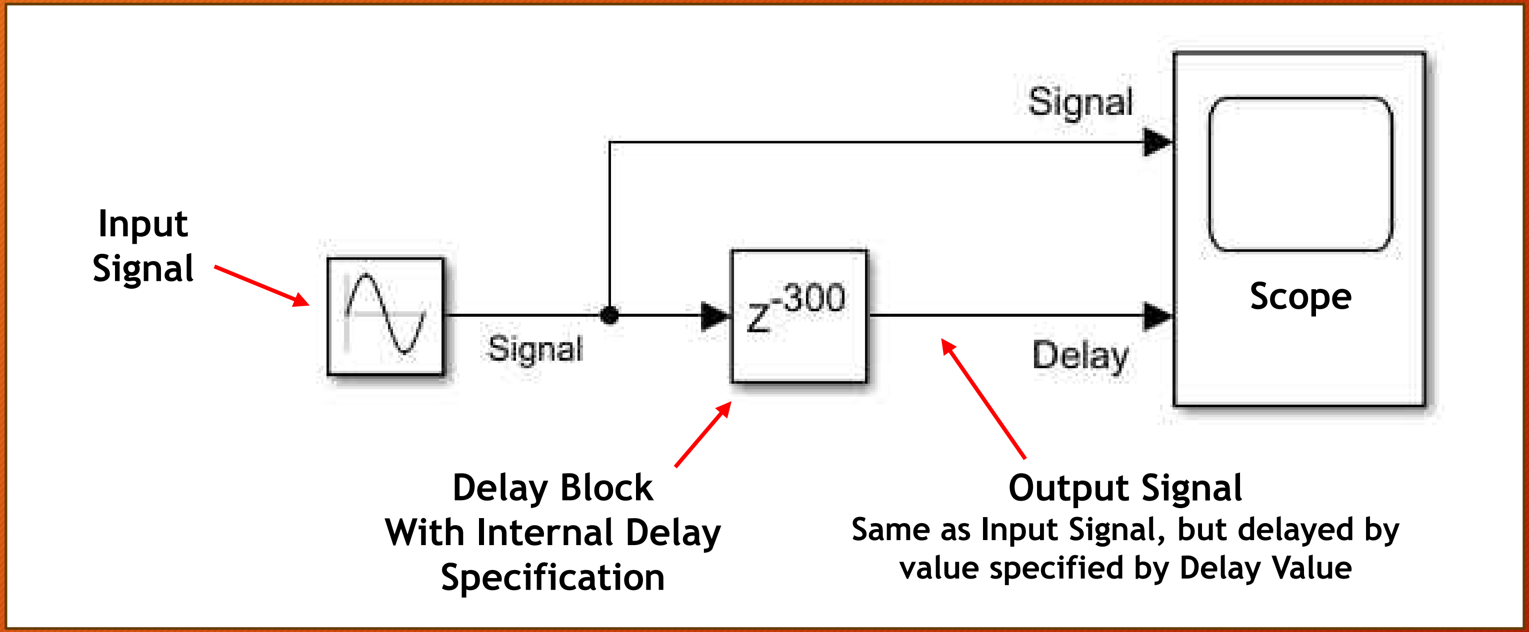
- If needed, the signal can be easily inverted

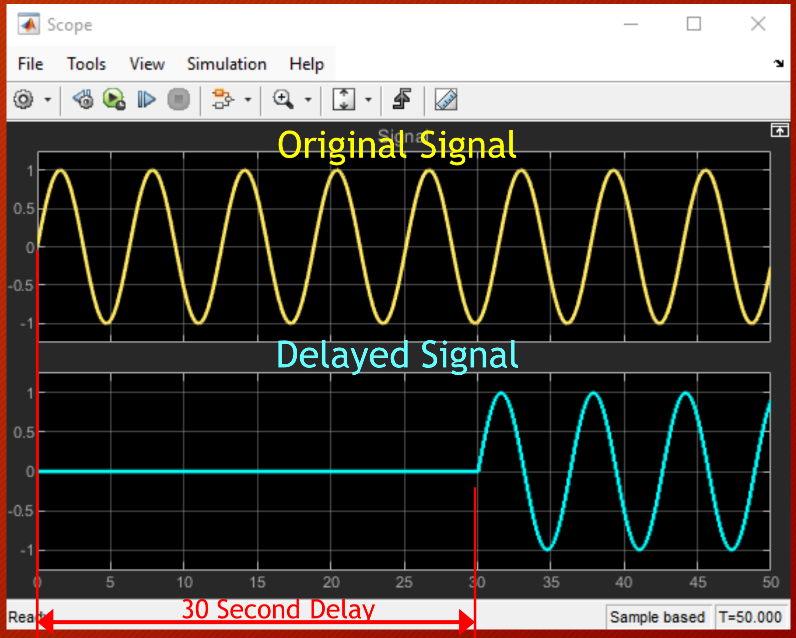


# How to Use the Simulink Delay Block

# Delay Block Basics

- The Variable Integer Delay block accepts an input signal and presents an output signal that is a reflection of the input signal, delayed by a specified amount
- The amount of delay can be fixed or variable, depending on how the Delay block is set up
- In the most basic usage, the delay can be set up to be a constant value specified internally to the Delay block
- In other usages, a constant or variable amount of delay can be specified by an external signal
- The most fundamental version of the use of the Variable Integer Delay block is shown here

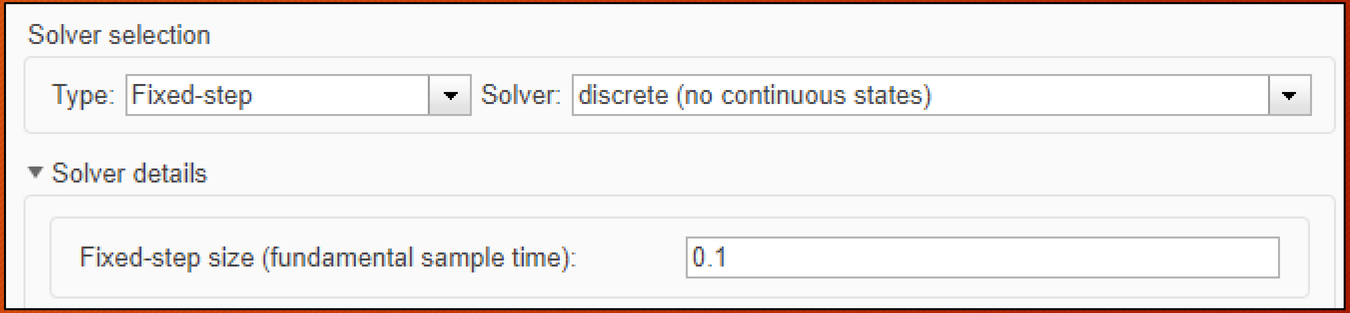


# How to Setup the Delay Interval

- For simple circuits involving delays, it is best to set up the simulation solver to be of type “Fixed-step” (rather than “Variable-step”), for example:

Found in:
Modeling -> Model Settings -> Model Settings




- Then, in the delay block, set the Delay Length Value and Sample time to be inherited from the solver:

Found in:
Delay Block -> Block Parameters



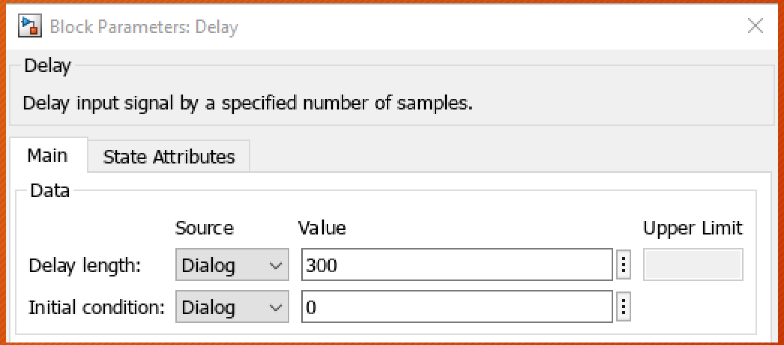
- To calculate the values needed to give you the delay you want, use the following:

Desired Delay Interval = Delay Length Value * Fixed-step size

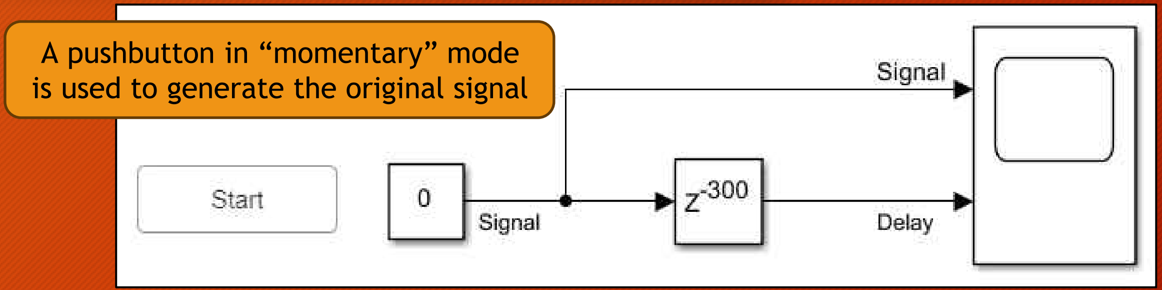
- Thus, adjust the Delay Length Value for the Fixed-step size you have set up in your solver to produced the Desired Delay Interval

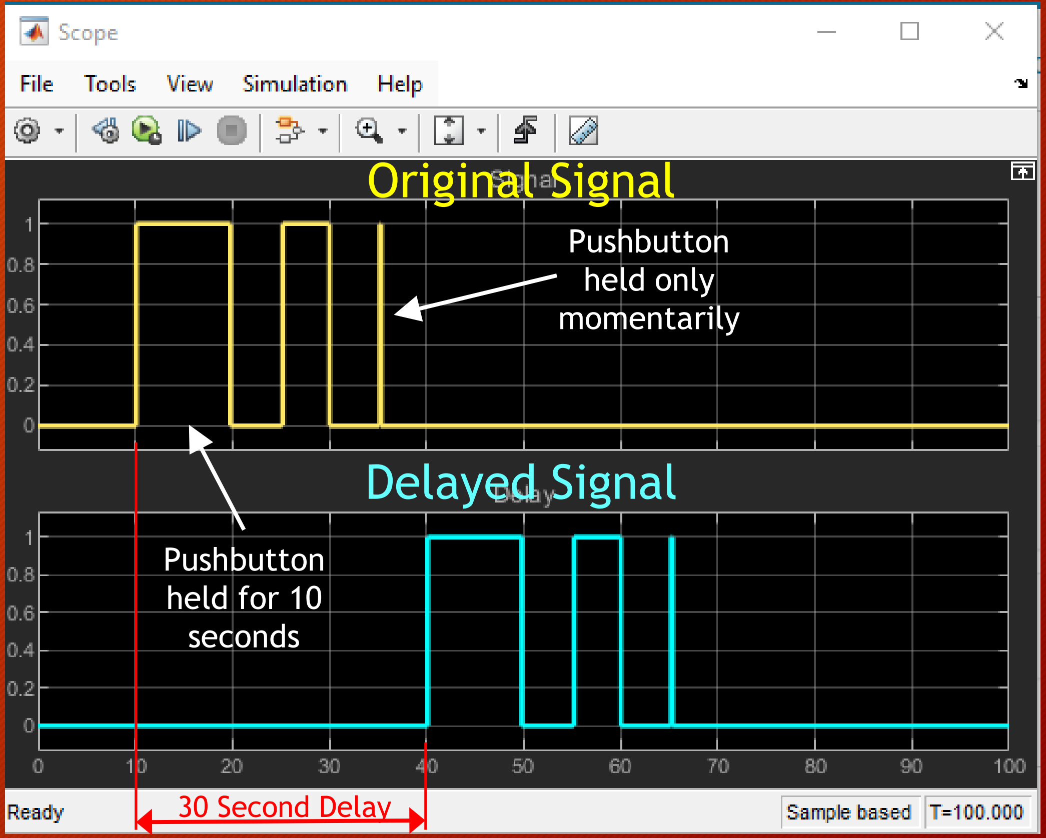
# Delay Block using Internal Delay Specification

- In this usage, the Delay block can be set up to have a constant delay value specified internally to the Delay block
  - For "Delay length" specify
    - Source as "Dialog" and
    - Set the Value as desired
  - Note: The actual delay amount is calculated as:
    - Value * Fixed-step size (in Model Settings)
    - To get a 30 second delay with fixed-step size of 0.1 seconds, use Value = 300



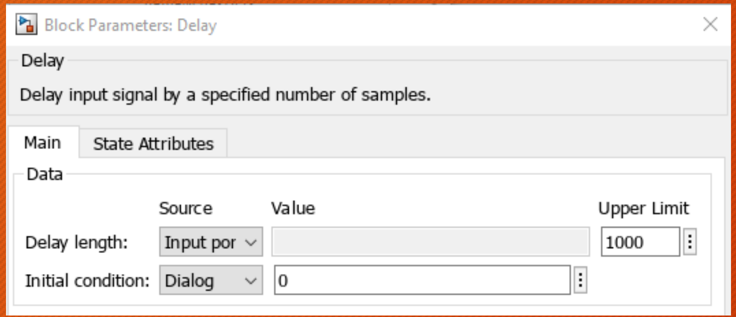

- The example on the right shows an internal delay set as 30 seconds
- You can see how the input signal was delayed by exactly 30 seconds to produce the delayed signal



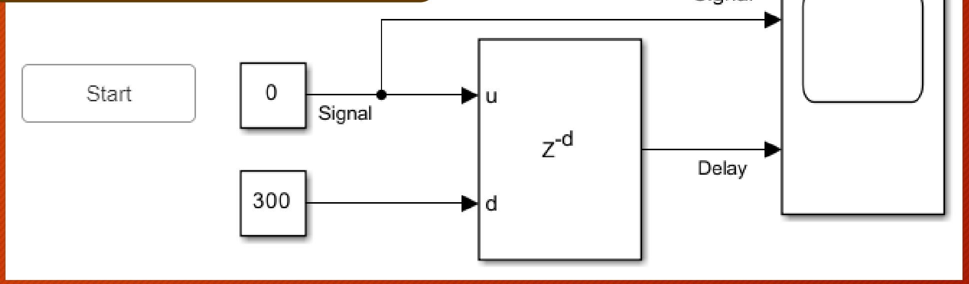
# Delay Block using a Constant External Delay Specification

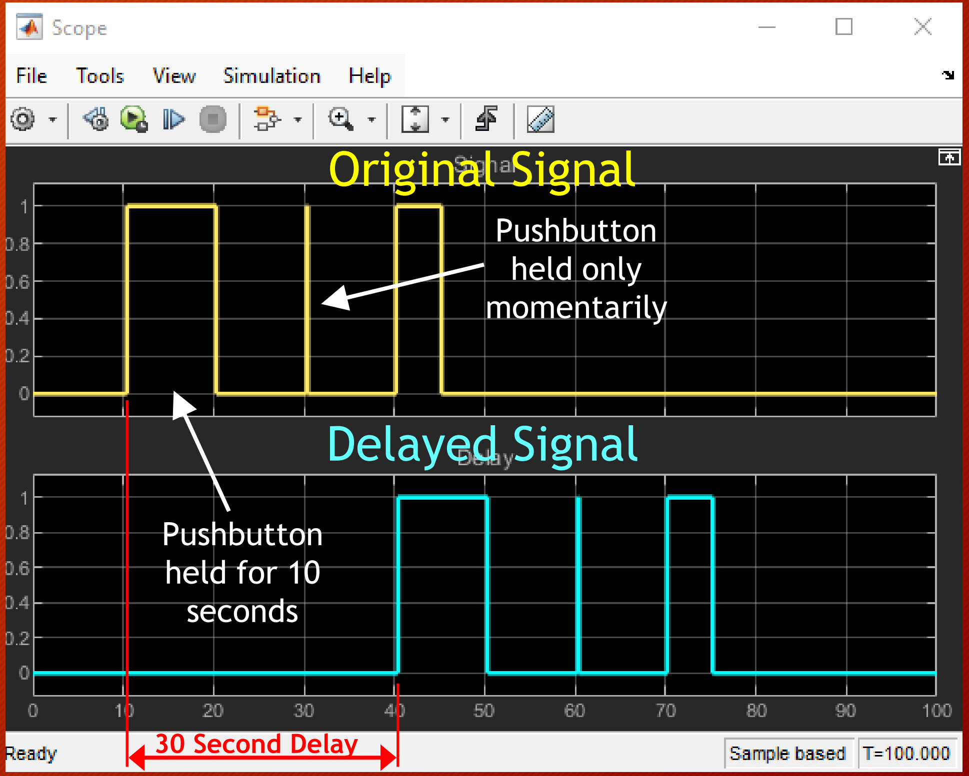
- In this usage, the Delay block can be set up to have a constant delay value specified externally to the Delay block
  - For "Delay length" specify
    - Source as "Input port"
    - This will open up a new input port called "d" on the Delay block
  - Note: The actual delay amount is calculated as:
    - <u>Value on "d" port * Fixed-step size</u> (in Model Settings)
    - To get a 30 second delay with fixed-step size of 0.1 seconds, use a Constant block value = 300




- The example on the right shows an external constant block set at 300 to produce a delay of 30 seconds
- You can see how the input signal was delayed by exactly 30 seconds to produce the delayed signal

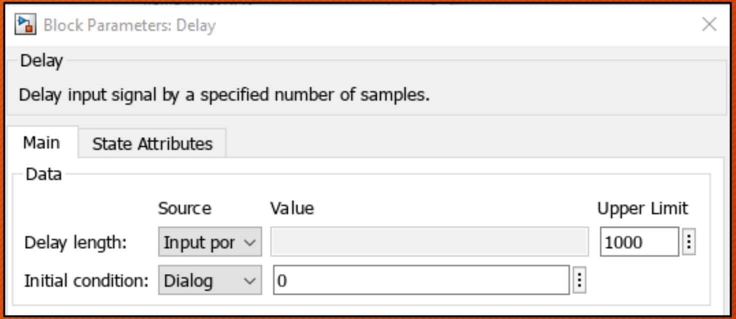
A pushbutton in "momentary" mode is used to generate the original signal

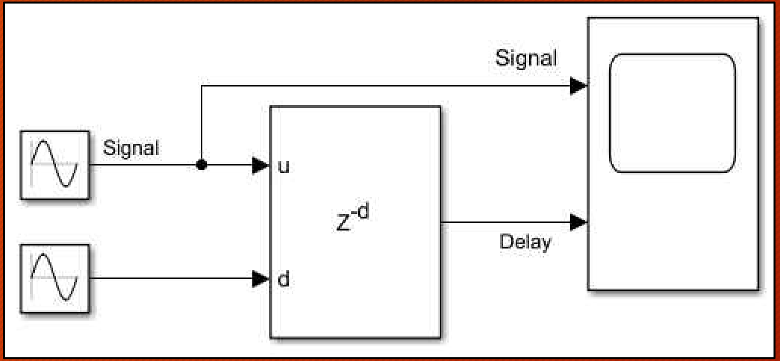


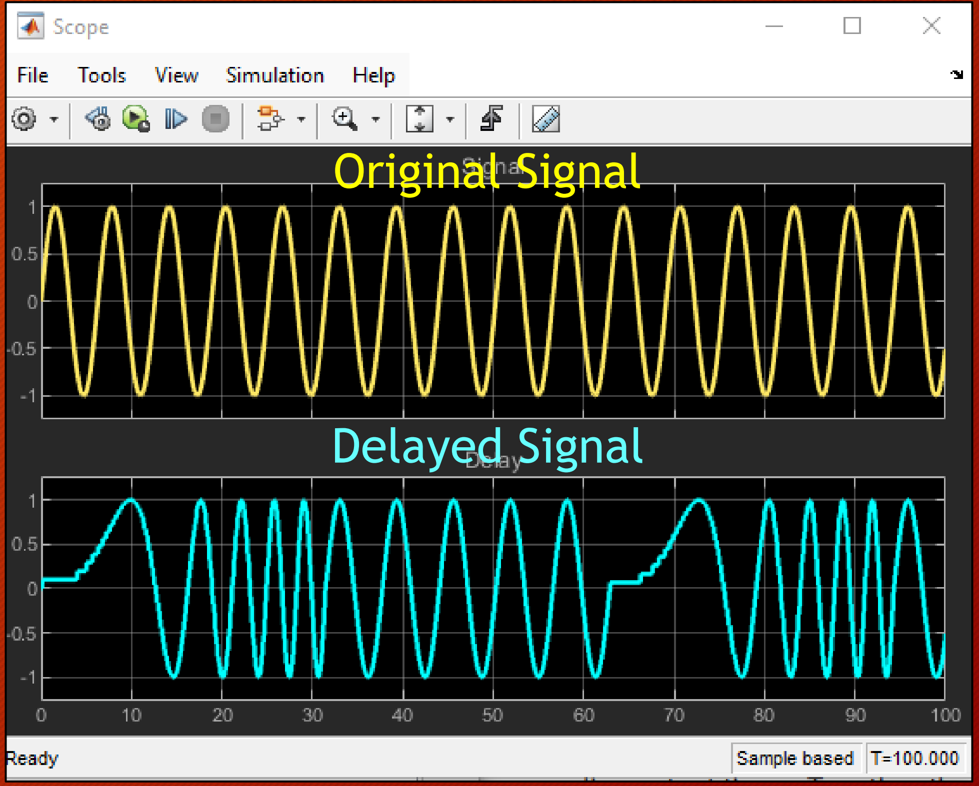



# Delay Block using a Variable External Delay Specification

- In this usage, the Delay block can be set up to have a variable delay value specified externally to the Delay block
  - For "Delay length" specify
    - Source as "Input port"
    - This will open up a new input port called "d" on the Delay block
  - Note: The actual delay amount is calculated as:
    - Value on "d" port * Fixed-step size (in Model Settings)
    - Since the value "d" is variable, the actual delay can be difficult to know at any one time



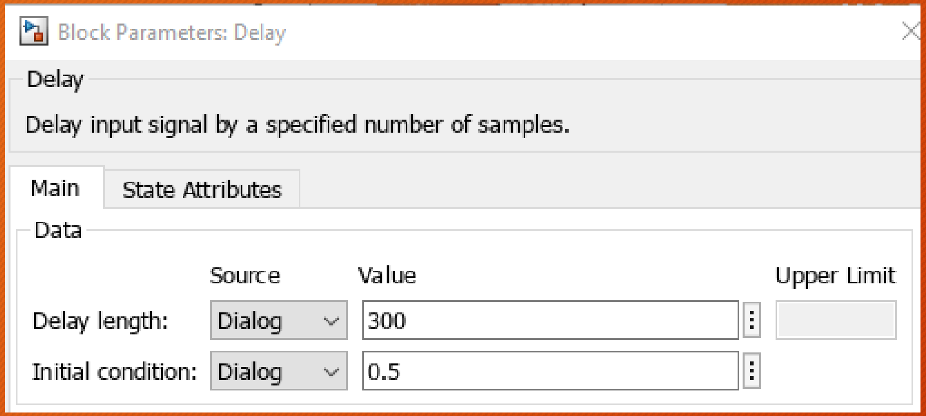



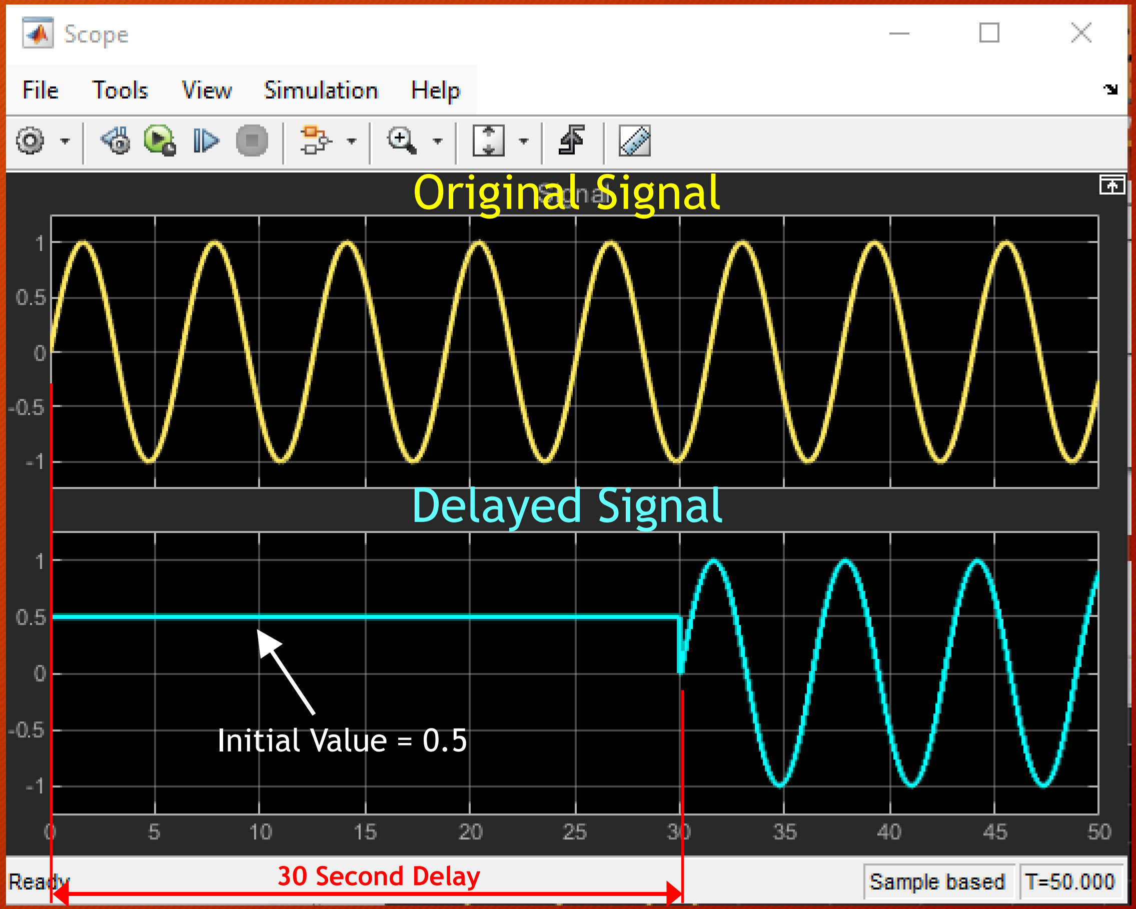




- In the example, a sine wave is used as the variable externally supplied value, which can make for some very strange output effects

# Initial Value using an Internal Specification

- In this usage, the Delay block can be set up to have an initial value specified internally to the Delay block
- For “Initial condition” specify
  - Source as “Dialog”

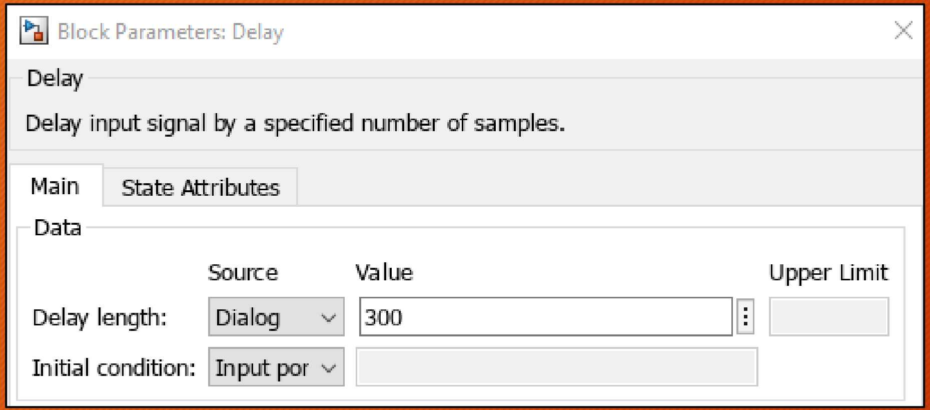


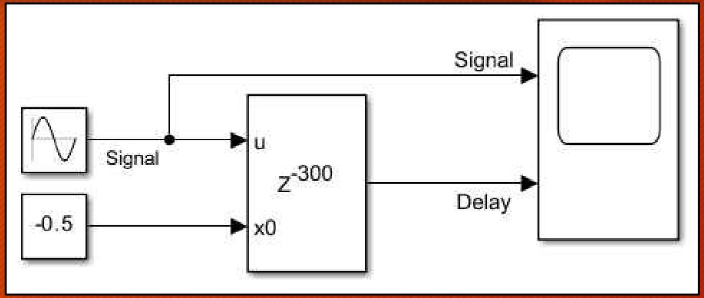



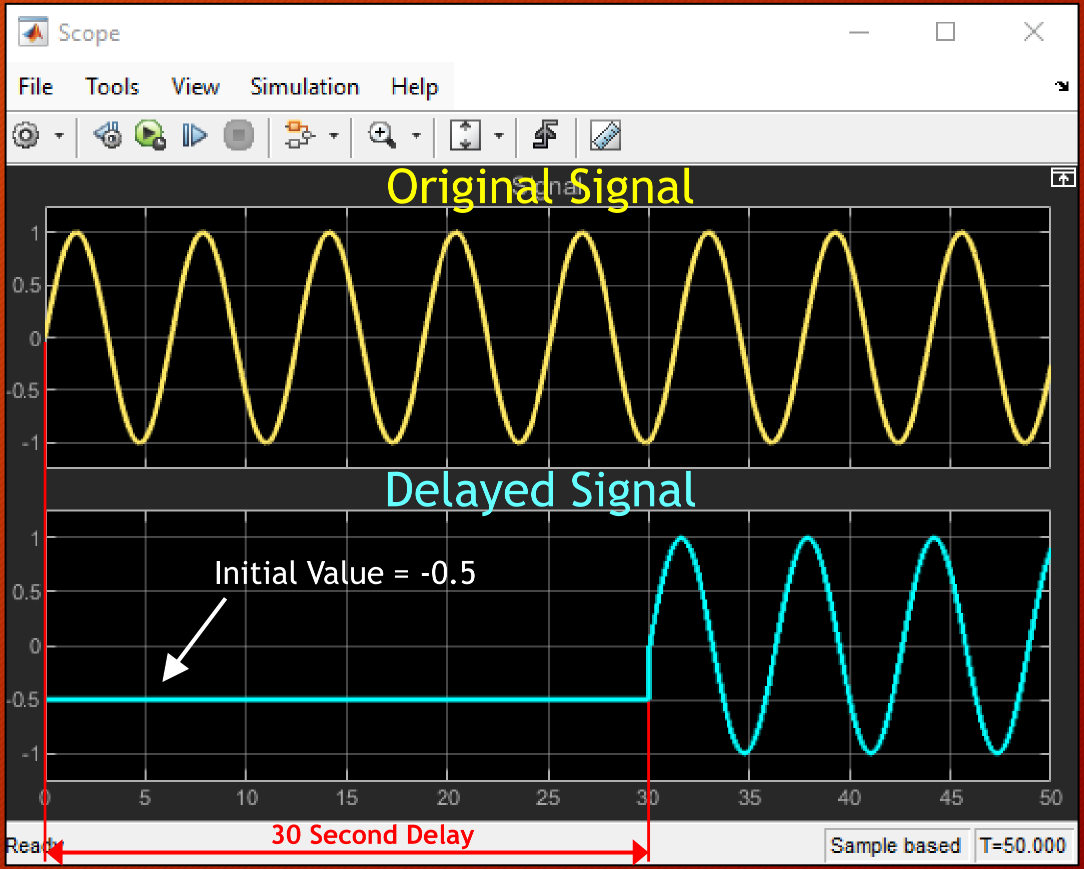

- The Initial Value will be output until it is time to begin outputting the delayed signal

# Initial Value using an External Specification

- In this usage, the Delay block can be set up to have an initial value specified externally to the Delay block
  - For “Initial condition” specify
    - Source as “Input port”
    - This will open up a new input port called “x0” on the Delay block



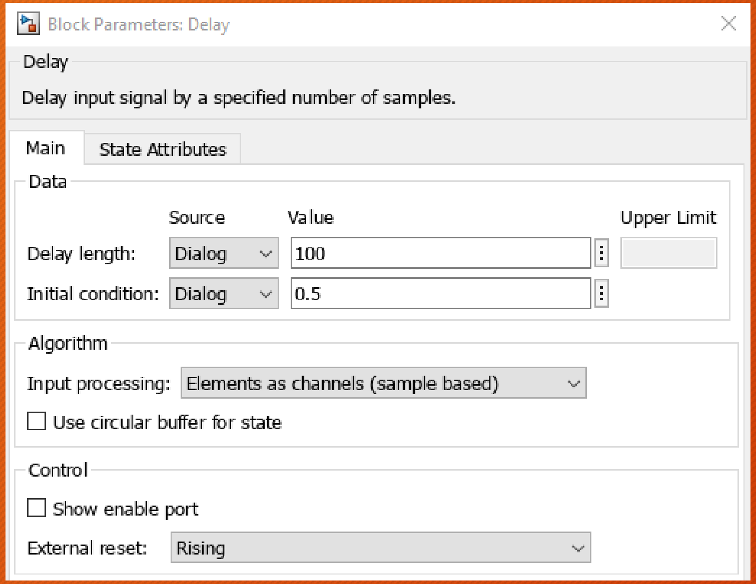




- The Initial Value will be output until it is time to begin outputting the delayed signal

# Reset to the Initial Value Based On a Signal Trigger

- In this usage, an external signal trigger is used to return the Delay block to an initial condition that lasts for the duration of the delay period
  - Under “Control” specify
    - “External reset” as one of the following (except “None”)



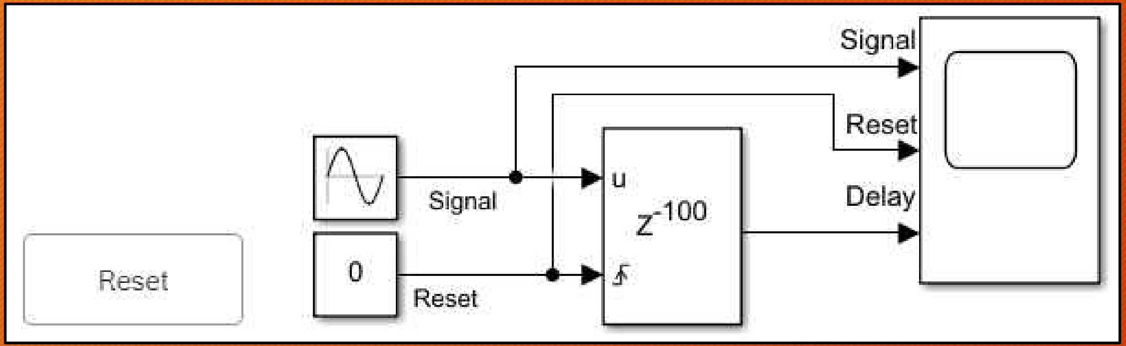

| Trigger Type | Port Symbol | Description |
|---|---|---|
| None | | No reset |
| Rising | | Reset on a rising edge |
| Falling | | Reset on a falling edge |
| Either | | Reset on either a rising or falling edge |
| Level | | Reset in either of these cases:<br>• When the reset signal is nonzero at the current time step<br>• When the reset signal value changes from nonzero at the previous time step to zero at the current time step |
| Level Hold | | Reset when the reset signal is nonzero at the current time step |

- This will open up a new input port with the symbol shown in the above table as the indicator of the type of port

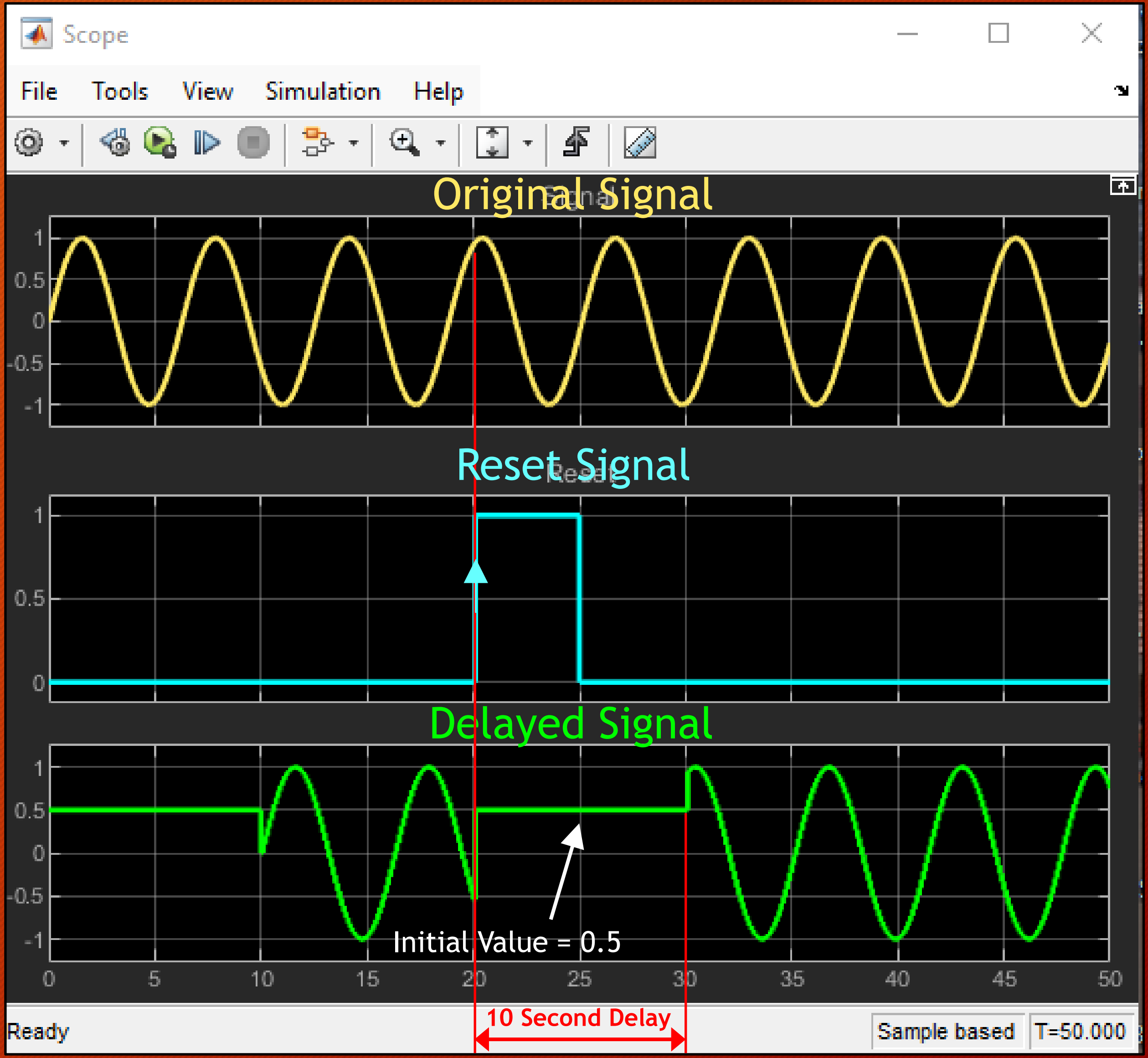
- Once triggered, the Initial Value will be output until the delay has expired at which time the delayed signal will be output

# Reset to the Initial Value on a Rising Trigger

- In this usage, the Delay block is set up to return to an initial condition when triggered by a rising signal

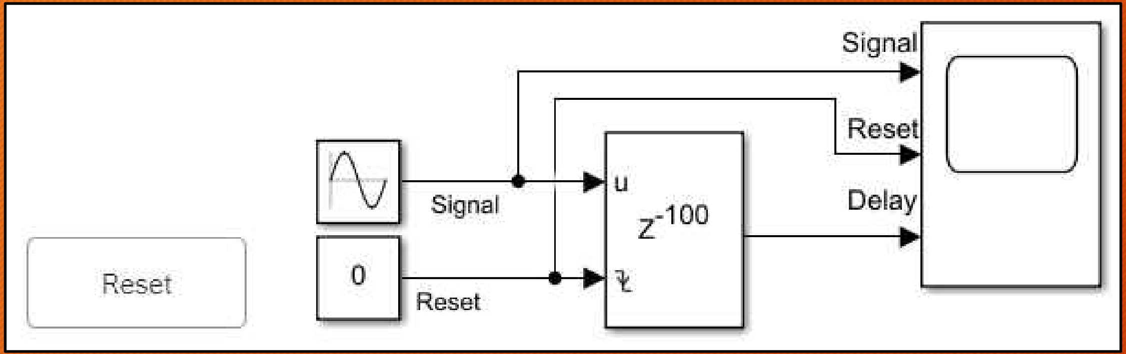



- The delay is set to 10 seconds
- When the rising edge of the reset signal is detected, the output goes to the initial condition for the duration of the 10 second delay
- Then the original signal is output, delayed by 10 seconds

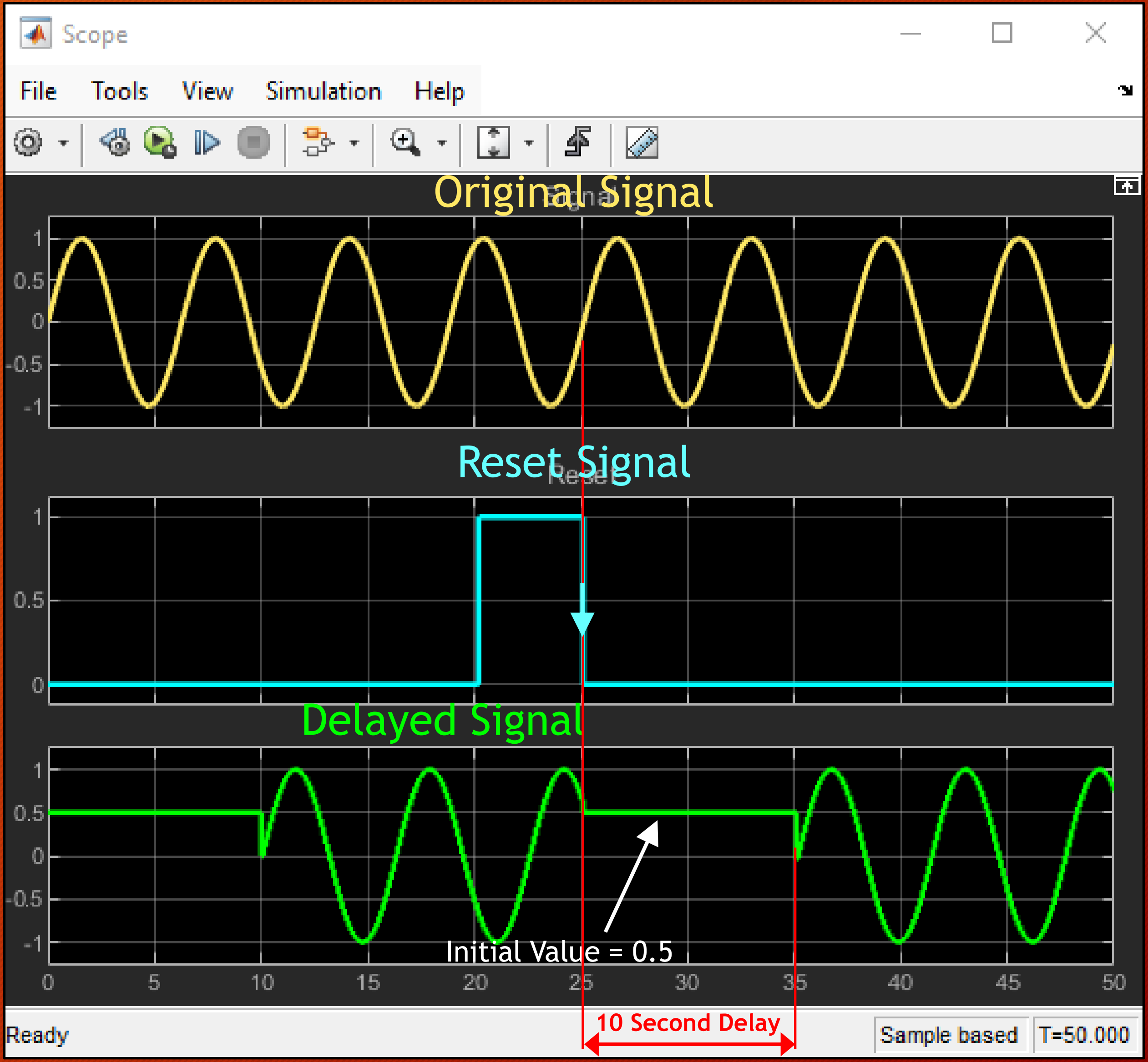


# Reset to the Initial Value on a Falling Trigger

- In this usage, the Delay block is set up to return to an initial condition when triggered by a falling signal

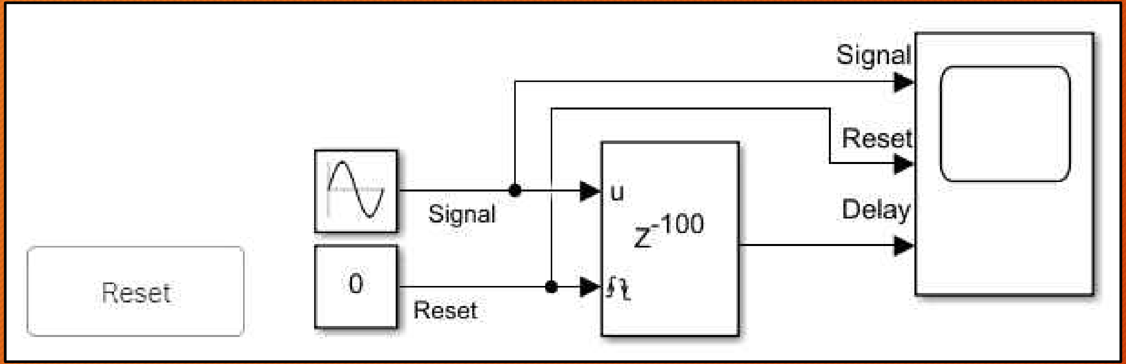



- The delay is set to 10 seconds
- When the falling edge of the reset signal is detected, the output goes to the initial condition for the duration of the 10 second delay
- Then the original signal is output, delayed by 10 seconds

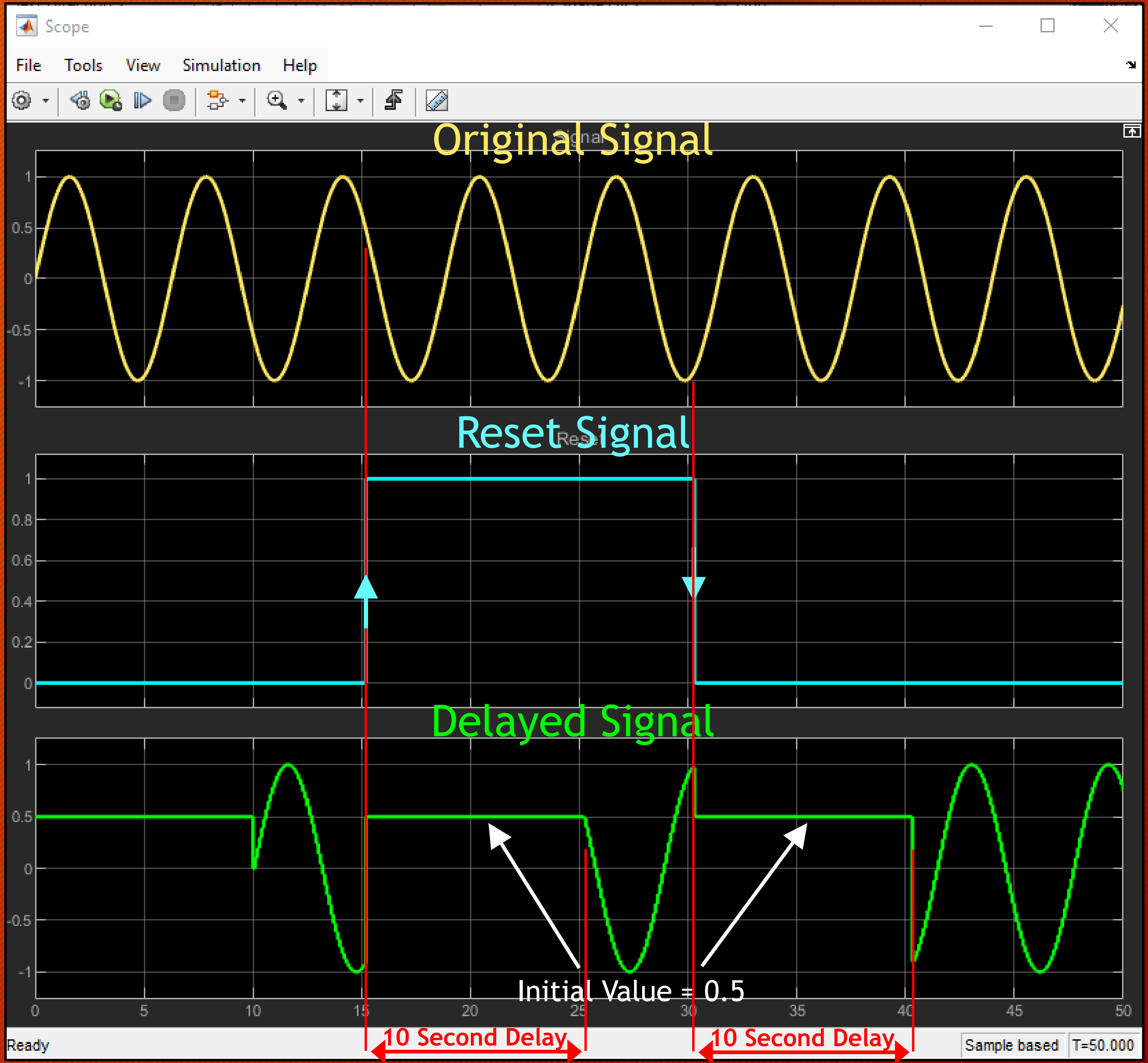


# Reset to Initial Value on both Rising and Falling Trigger

- In this usage, the Delay block is set up to return to an initial condition when triggered by a rising or a falling signal

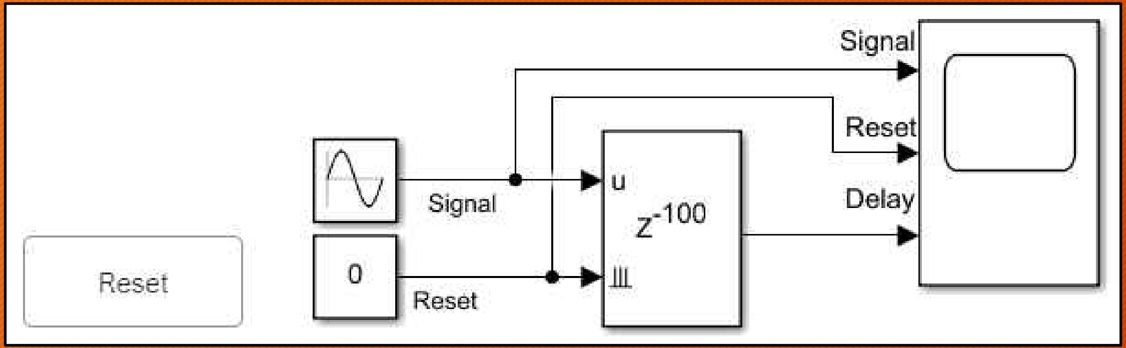



- The delay is set to 10 seconds
- When either a rising or falling edge of the reset signal is detected, the output goes to the initial condition for the duration of the 10 second delay
- Then the original signal is output, delayed by 10 seconds

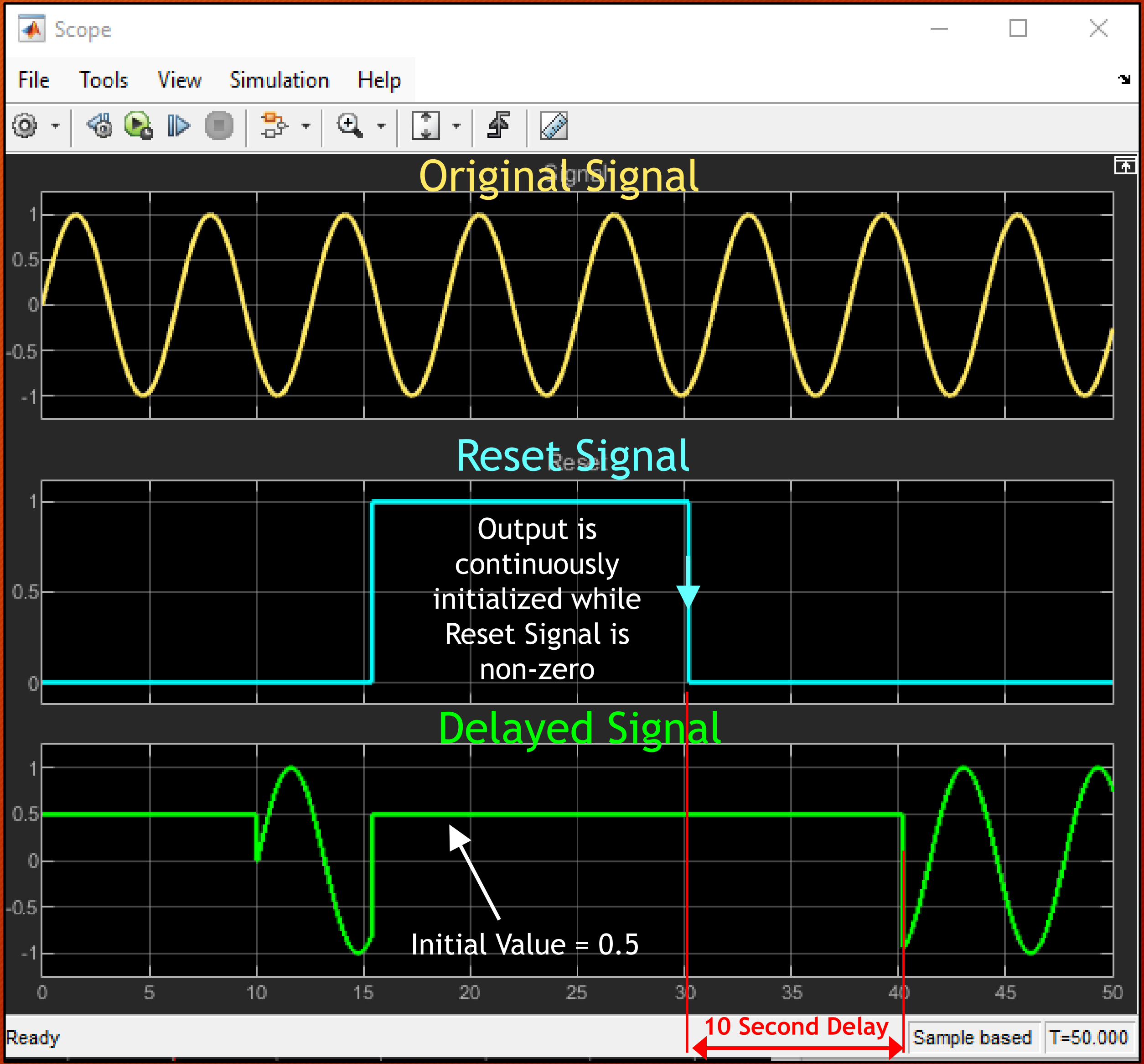


# Reset to the Initial Value on a Level Trigger

- In this usage, the Delay block is set up to return to an initial condition when triggered by a level signal

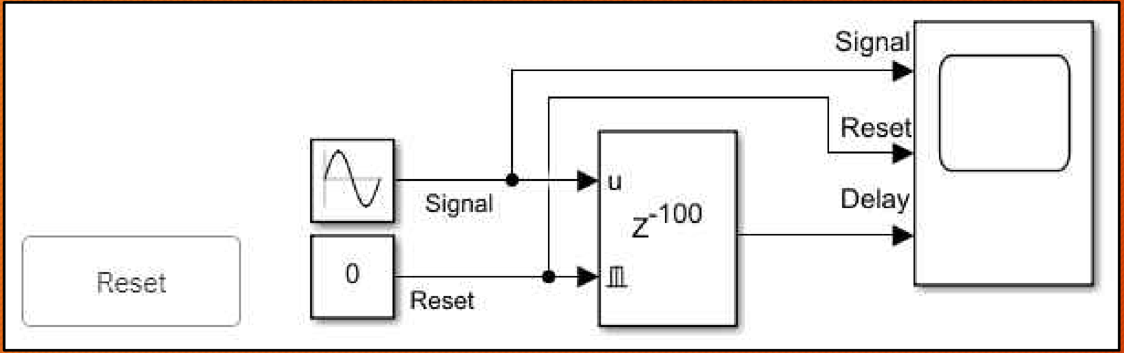



- The delay is set to 10 seconds
- As long as the reset signal is non-zero, the output will be continuously initialized
- When the reset signal transitions to zero, the output goes to the initial condition for the duration of the 10 second delay
- Then the original signal is output, delayed by 10 seconds

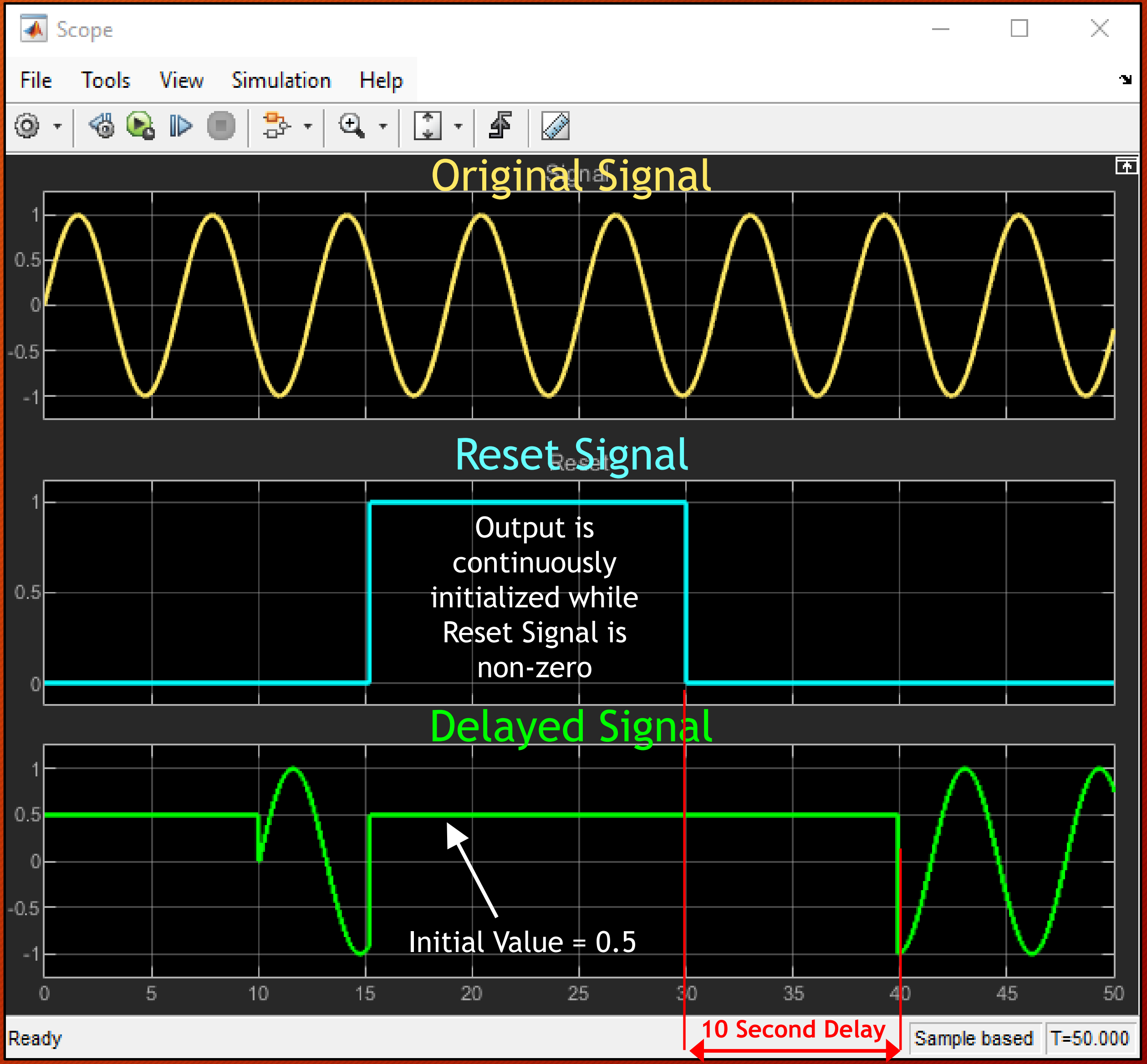


# Reset to the Initial Value on a Held Level Trigger

- In this usage, the Delay block is set up to return to an initial condition when triggered by a held level signal



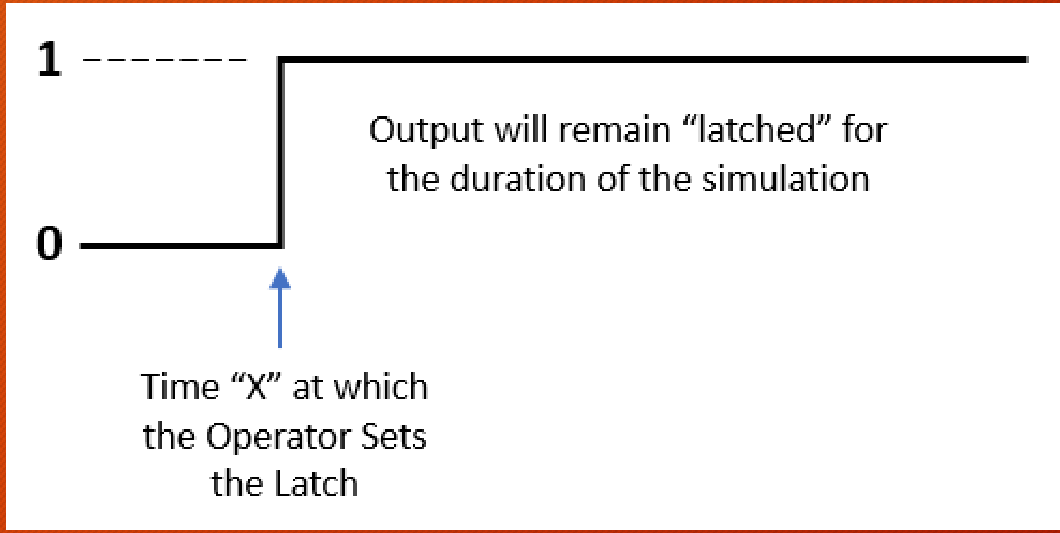

- The delay is set to 10 seconds
- When the reset signal transitions to zero, the output goes to the initial condition for the duration of the 10 second delay
- Then the original signal is output, delayed by 10 seconds



# Using the S-R Flip Flop Block to Latch a Signal

# Purpose

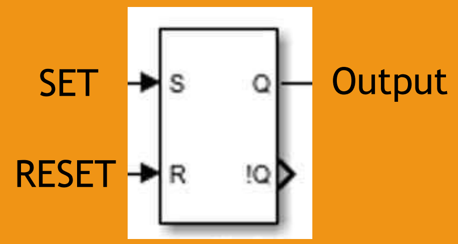
- To “latch” a signal means to capture it and hold it for some period of time, possible indefinitely
- Assume that it is desired to latch an output signal that starts at a specified simulation time, and lasts for the duration of the rest of the simulation
- Assume that a Simulation Operator uses simulation controls to control the starting of the latched output signal
- The resulting output signal would appear as follows:



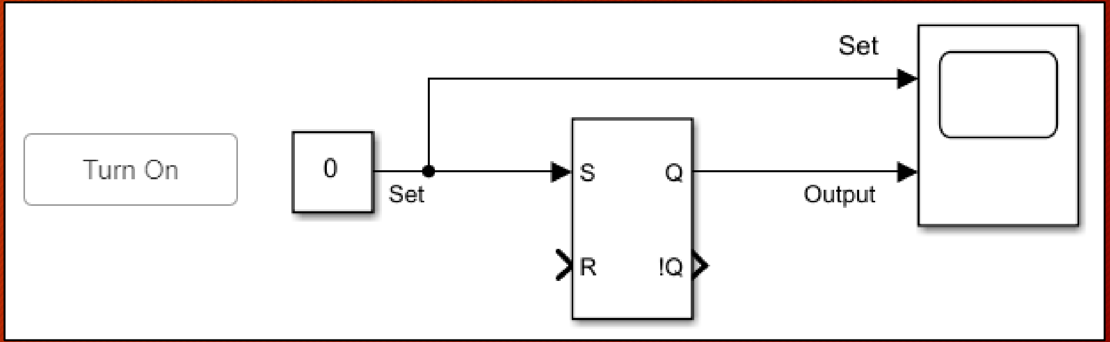
# One Possible Solution: S-R Latch

- You can use the basic S-R Latch to turn on (Set) and latch (hold) a Boolean output signal
  - When S goes 0 -> 1 (while R = 0), the output signal (Q) is set to 1 (turned on)
  - In this setup, Reset is not used; it is always = 0
  - When S = R = 0 the output is unchanged
- This device can be used to advantage in conjunction with a momentary pushbutton connected to input S
  - In such a setup, when the momentary pushbutton is pressed, it sends a signal of amplitude = 1 to S, which Sets the output to on (=1)
  - When the momentary pushbutton is release, its value goes back to 0, such that S = R = 0, causing the output signal to remain on (=1) indefinitely

Boolean refers to a signal that is 0 (False) or 1 (True)



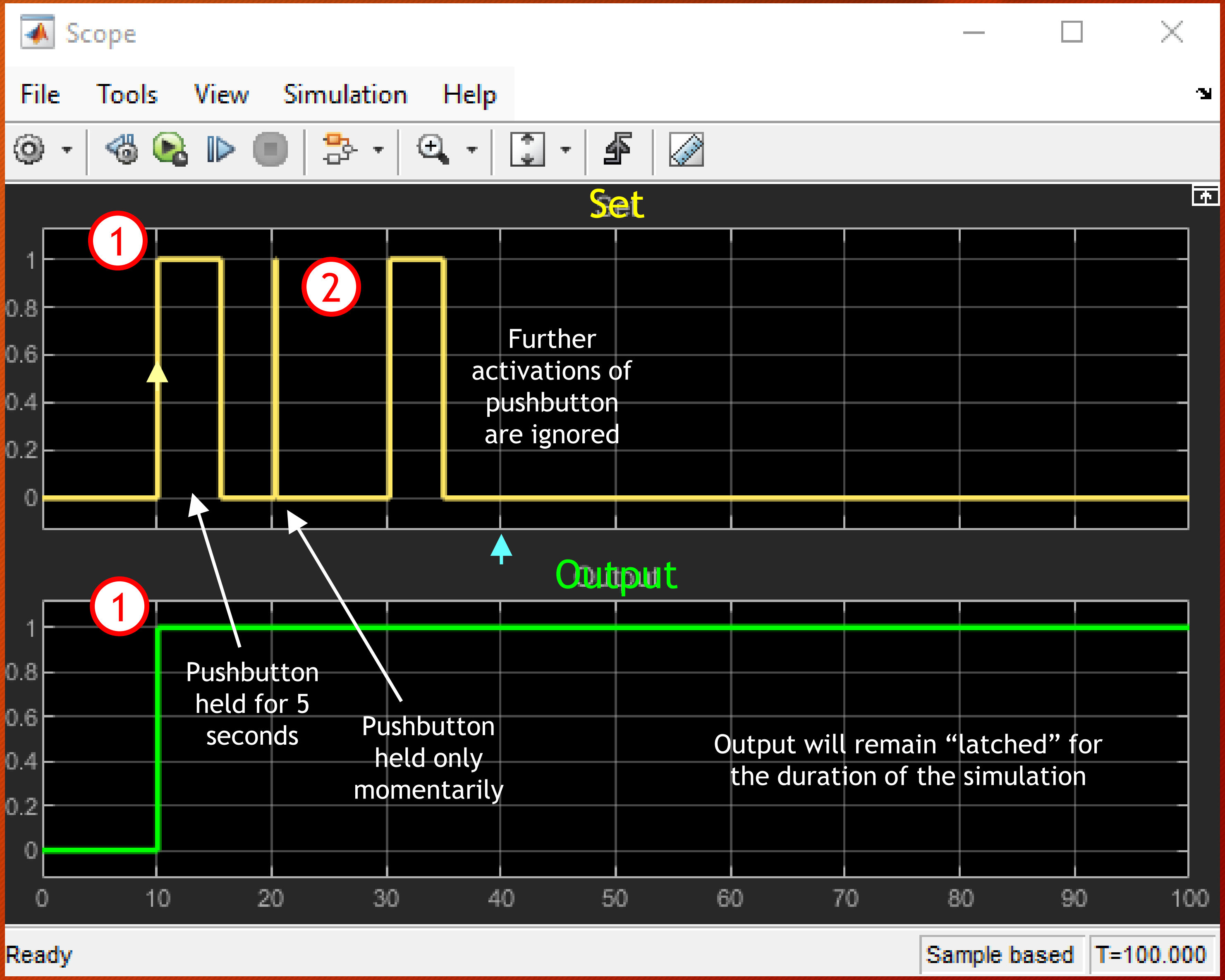

While Simulink calls this block a Flip Flop, it actually functions as a simple latch




**Simulink Model**

# Scope Results for S-R Latch Example Model

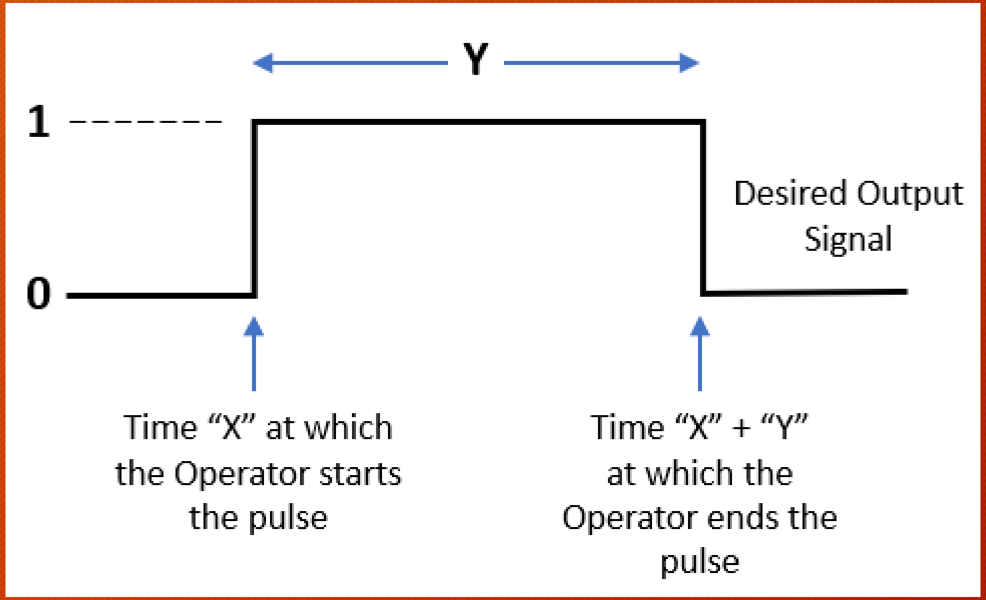
1. Output goes high (=1) when Set signal goes high
   a) It doesn't matter how long the pushbutton is held (how long the Set signal is applied)
   b) It doesn't matter how short the pushbutton is held
   c) All that matters is when the rising edge of the Set signal occurs
   d) When the rising edge is detected, the S-R latch output (Q port) goes high (=1)
2. Additional pulsings of the Set signal are ignored



# Using the S-R Flip Flop Block to Produce a Pulse

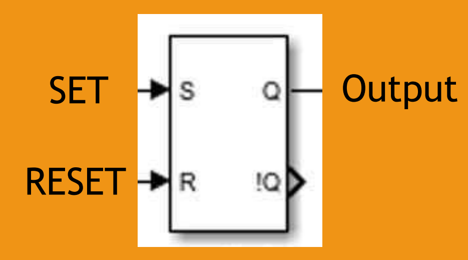
# Purpose

- Assume that it is desired to produce an output pulse that starts at a specified simulation time, and lasts for a specified duration
- Assume that a Simulation Operator uses simulation controls to control the starting and stopping of the output signal pulse
- The resulting output signal would appear as follows:

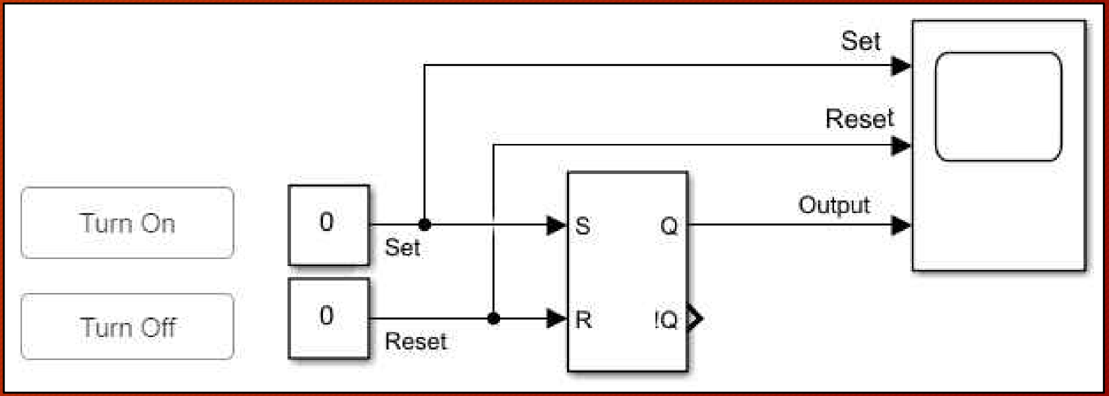


# One Possible Solution: S-R Latch

- You can use the basic S-R Latch to turn on (Set) and turn off (Reset) a Boolean (0 or 1, True or False) output signal
  - When S goes 0 -> 1 (while R = 0), the output signal (Q) is set to 1 (turned on)
  - When R goes 0 -> 1 (while S = 0), the output signal is set to 0 (turned off)
  - When S = R = 0 the output is unchanged
  - When S = R = 1 the output is undefined (avoid this condition)
- This device can be used to advantage in conjunction with a momentary pushbutton connected to input S
  - In such a setup, when the momentary pushbutton is pressed, it sends a signal of amplitude = 1 to S, which Sets the output to on (=1)
  - When the momentary pushbutton is release, its value goes back to 0, such that S = R = 0, causing the output signal to remain on (=1) until reset
  - When R is set to one (with a second momentary pushbutton), the output is Reset (=0)
  - At this point, S = R = 0, which is the initial condition, allowing the process to repeat



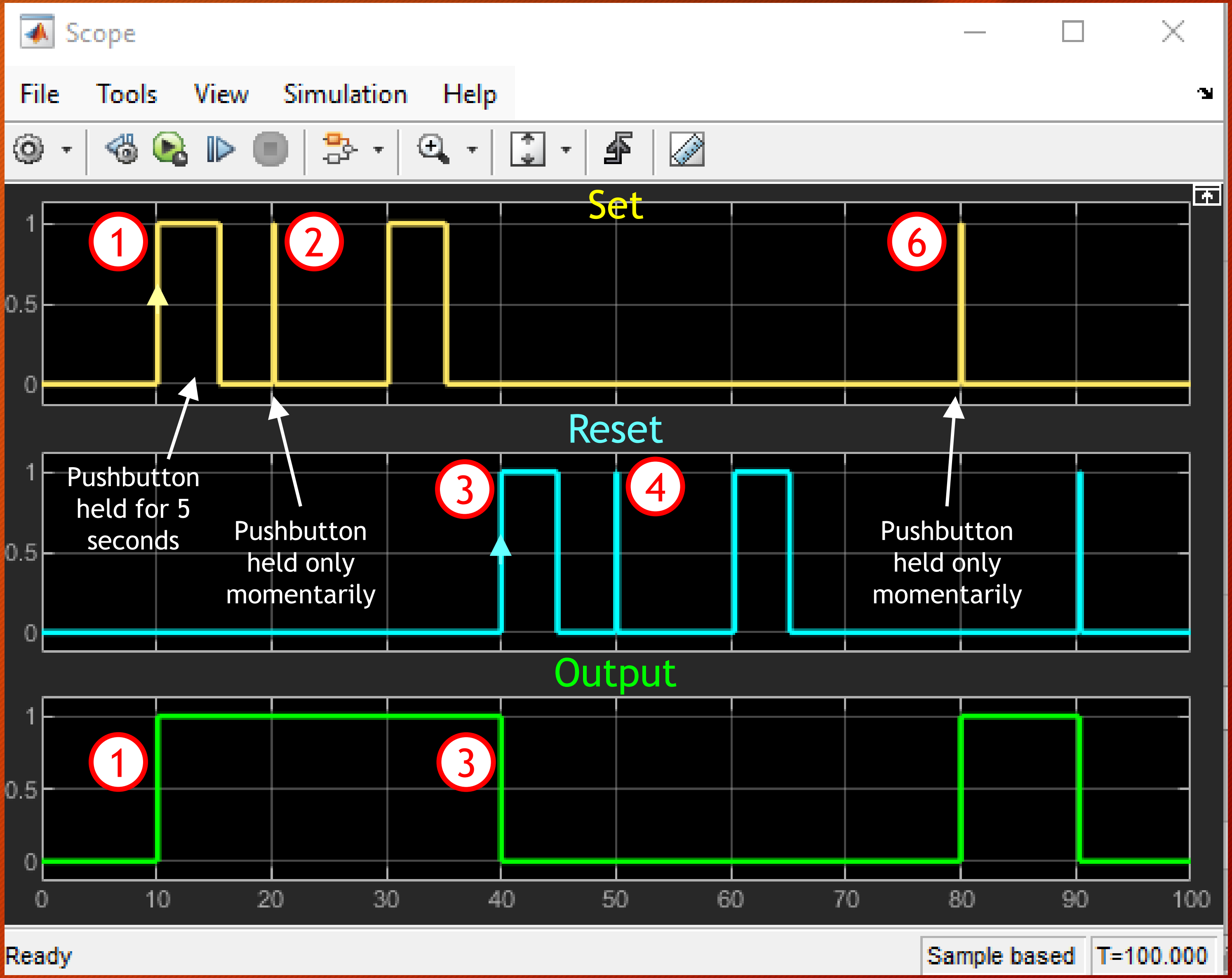

While Simulink calls this block a Flip Flop, it actually functions as a simple latch




**Simulink Model**

# Scope Results for S-R Latch Pulse Example Model

1. Output goes high (=1) when Set signal goes high
2. Additional pulsings of Set signal are ignored
3. When Reset signal goes high, the Output goes low (=0)
4. Additional pulsings of Reset signal are ignored
5. After reset, system is reinitialized to be used again
6. Second cycle of pulsing of output signal



# Using the S-R Flip-Flop in Combination with a Delay Block to Generate a Pulse with Controllable Pulse Width

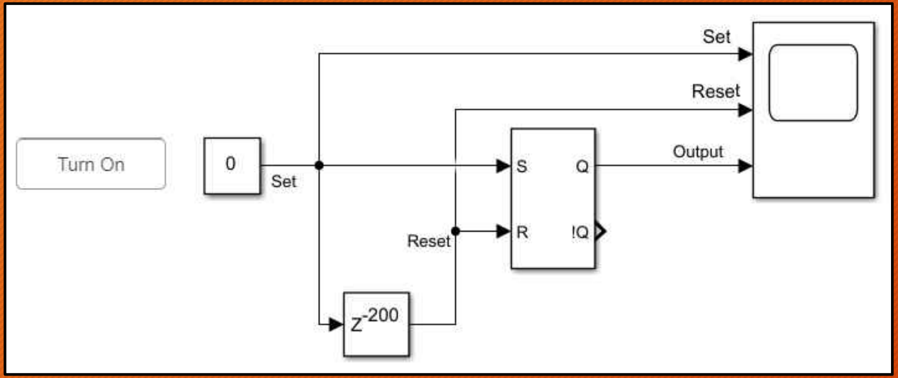
The objective is to allow an operator pushing a button to initiate a pulse of a specified duration (pulse width)

The circuit needs to be designed as a kind of “Fire-and-Forget” system

Such that once the pulse is “Fired”, it will automatically be terminated at the specified pulse width, and the operator can “Forget” about it once the pulse is initiated

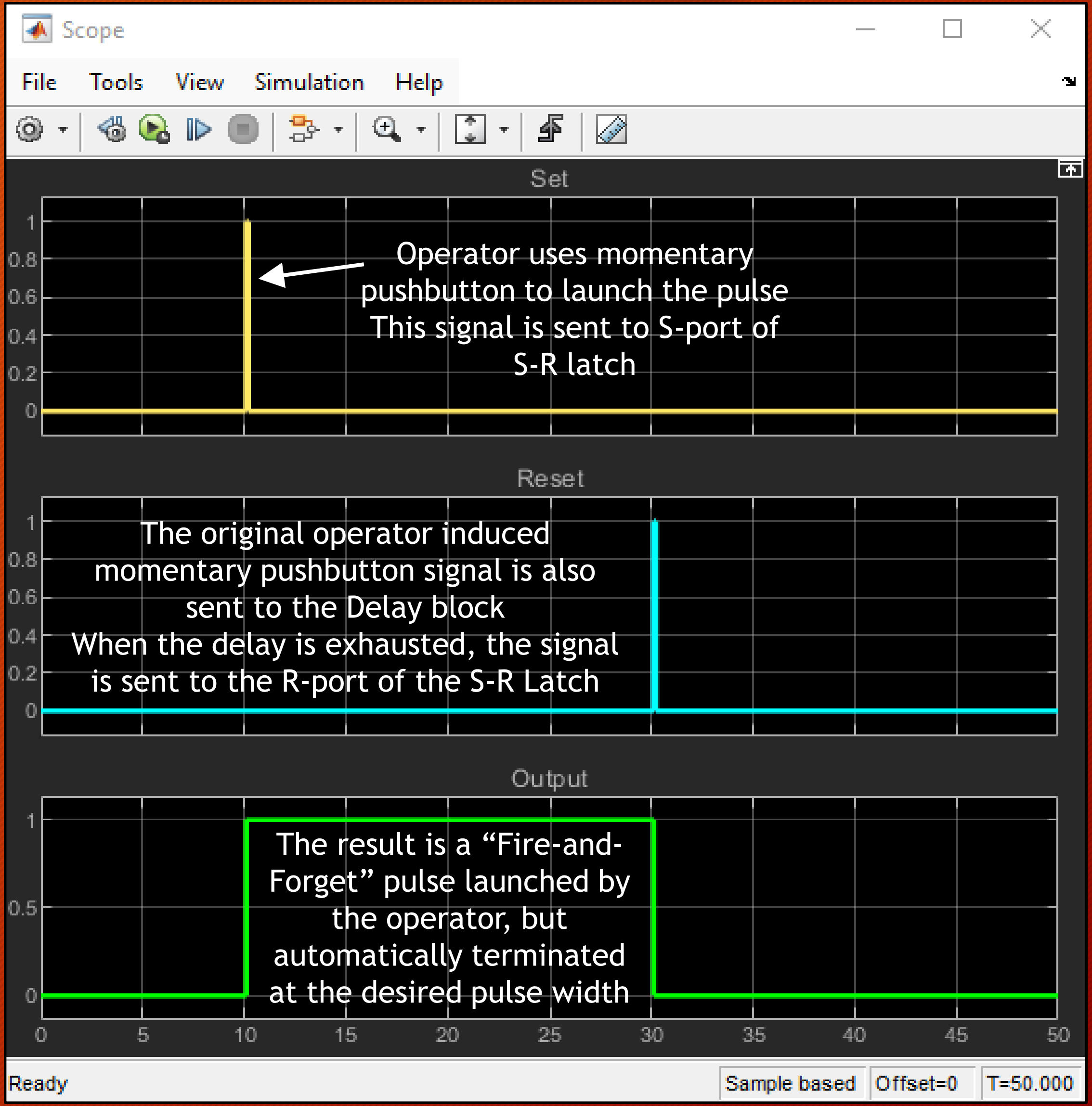
# Operator Starts a Pulse, that is then Automatically Terminated

- The circuit is controlled by a momentary pushbutton that "Fires" off the pulse




- The same momentary signal that fired off the pulse is sent to a delay block with the specified amount of delay
- When the delay is exhausted, the delayed momentary pushbutton signal is sent to the Reset port of the S-R Flip Flop, and the pulse is automatically terminated

Obviously, this simple solution is limited in that the operator cannot hold the momentary pushbutton longer than the desired pulse width, or the circuit fails to achieve its objective



# References

# References


1. The MathWorks' User Guide for Simulink
   https://www.mathworks.com/help/pdf_doc/simulink/simulink_ug.pdf