



How to Initialize an Array Tutorial

Functions -> Programming -> Array  -> Initialize Array 

Start by opening the Functions palette and select the Programming palette. Under the Programming palette select the Array palette  where you should find the Initialize Array  function.

After you have placed the Initialize Array function into your Block Diagram you will notice that by default there are two inputs on the right and one output on the left. You will see in Figure 1 that the first input at the top (the “element” input in Figure 1) is the default value you want to initialize all the array entries to. The subsequent inputs (from “dimension size 0” through “dimension size n-1” are Numeric I32s that determine the dimensions of the array that is to be initialized.

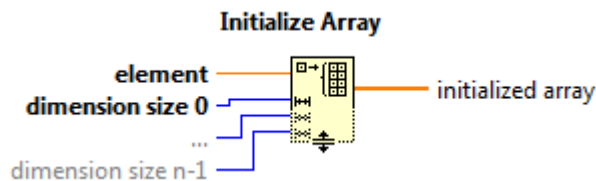


Figure 1

By default the array is one-dimensional, however, if you wanted to produce a two dimensional array, simply resize the function by dragging the bottom down until you get the desired number of inputs as shown in Figure 1.

The output of the function is the array where all the default values in the array are initialized to the value of the first input “element” as shown in Figure 1.

When we specify the dimension inputs for two dimensional arrays, the “dimension size 0” input corresponds to the number of rows, while the second dimension input (which would be “dimension size 1”) corresponds to the number of columns.

For example, if you wanted to initialize a 4 x 5 array of 1s, we would start by expanding the array function so that there are two “dimension size” inputs as shown in Figure 1. We would then connect the constant 1 to the “element” input so that the Initialize Array function knows which value to put as the default.

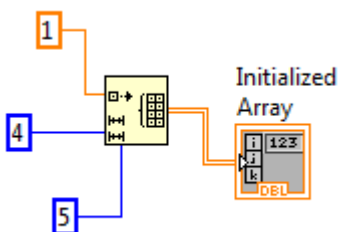


Figure 2



Figure 3

We would also connect 4 to the first dimension input and 5 to the second dimension input since we want the array to be 4 x 5, or have 4 rows and 5 columns. The initialized array as displayed on the front panel is shown in Figure 3.

Notice that the order in which we wire the inputs to the dimension matters. If we wired the inputs in the opposite configuration we would get a 5 x 4 array, not 4 x 5.