An Introduction to Writing S-Functions for the MPC 5553

EECS 461 Fall 2009

Overview

MATLAB S-functions are an effective way to embed object code into a Simulink model. These S functions can be written in a few languages, C being the one most relevant for our purposes. This tutorial is meant to serve as a guide – almost a walk-through – that explains what all you need to know regarding S functions and how to use them in your final project Simulink models.

In lab 1, we wrote a simple adder program. We followed this up in lab 8 where we used Simulink blocks to represent the GPIO blocks and to shift and add integers. We will now combine concepts from labs 1 and 8 and utilize the best of both worlds. Understanding how and when to use S-functions in your final project is almost guaranteed to make your project significantly more understandable to both you and the GSI who helps you debug your project.

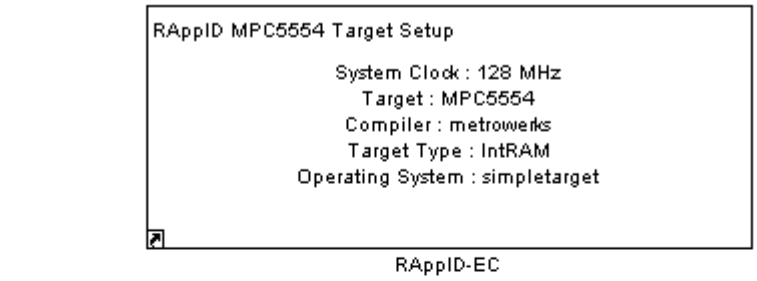
Tutorial

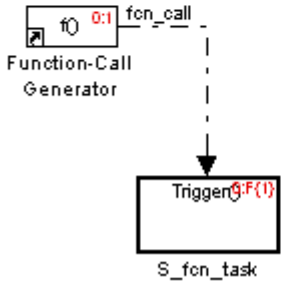
1. The first step is to select a C compiler in MATLAB. Though the MATLAB C compiler (lex) cannot generate object code for the MPC 5553, it is an effective tool at figuring out syntax errors in your C code before you begin the final build process.

   At the MATLAB prompt, enter "mex -setup"

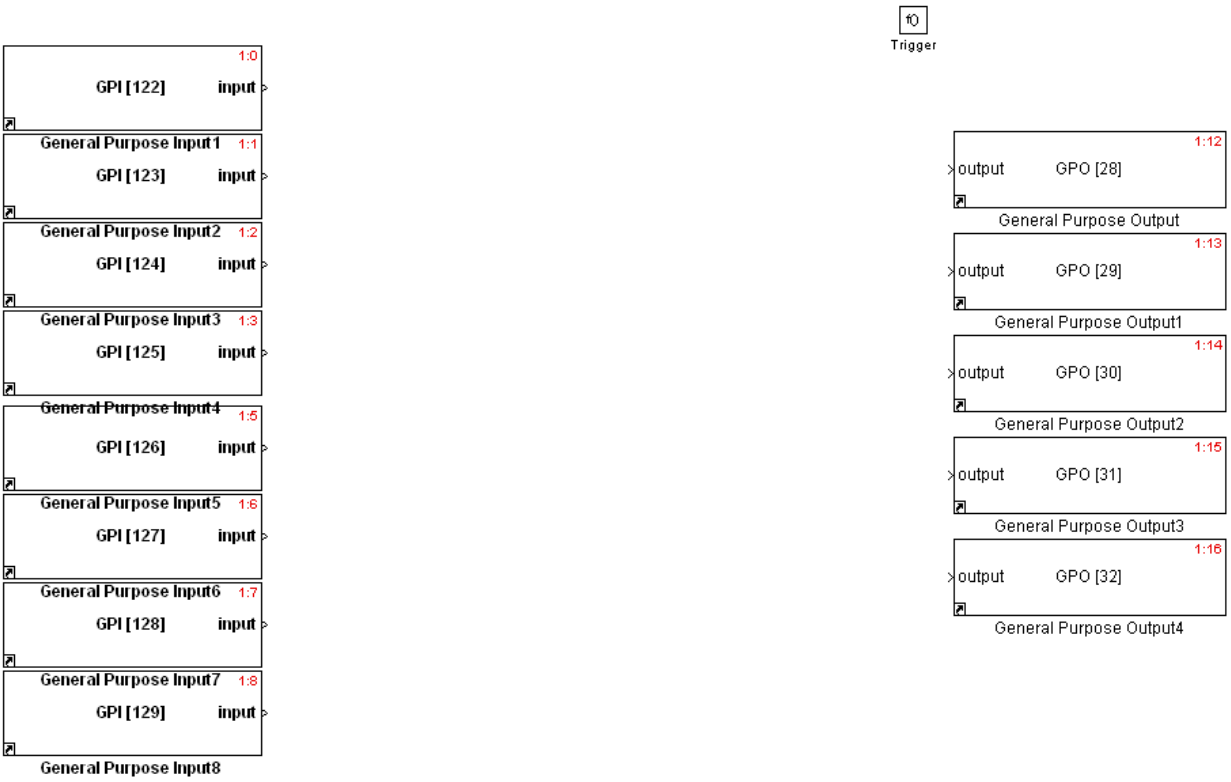
   When asked, select the Lex C compiler (should be option 1). This should then cause an error because a valid input file was not specified. Ignore this error, as the purpose of this step was to merely select a compiler, which is now done.

2. Create a new Simulink model. It is best if you make a copy of the adder model you used in lab 8 so that all the necessary settings are properly copied. You should now have something resembling the figure below.

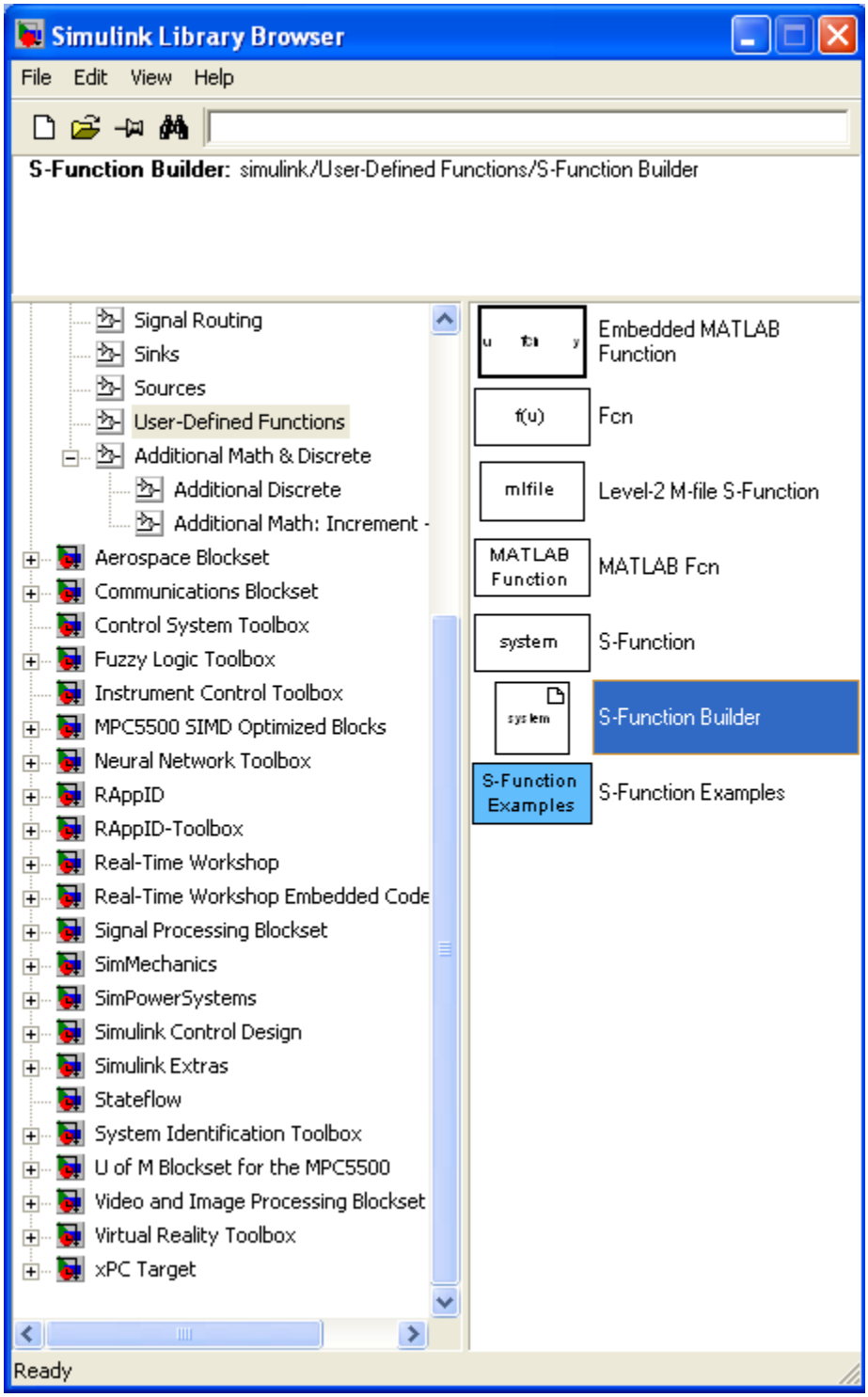




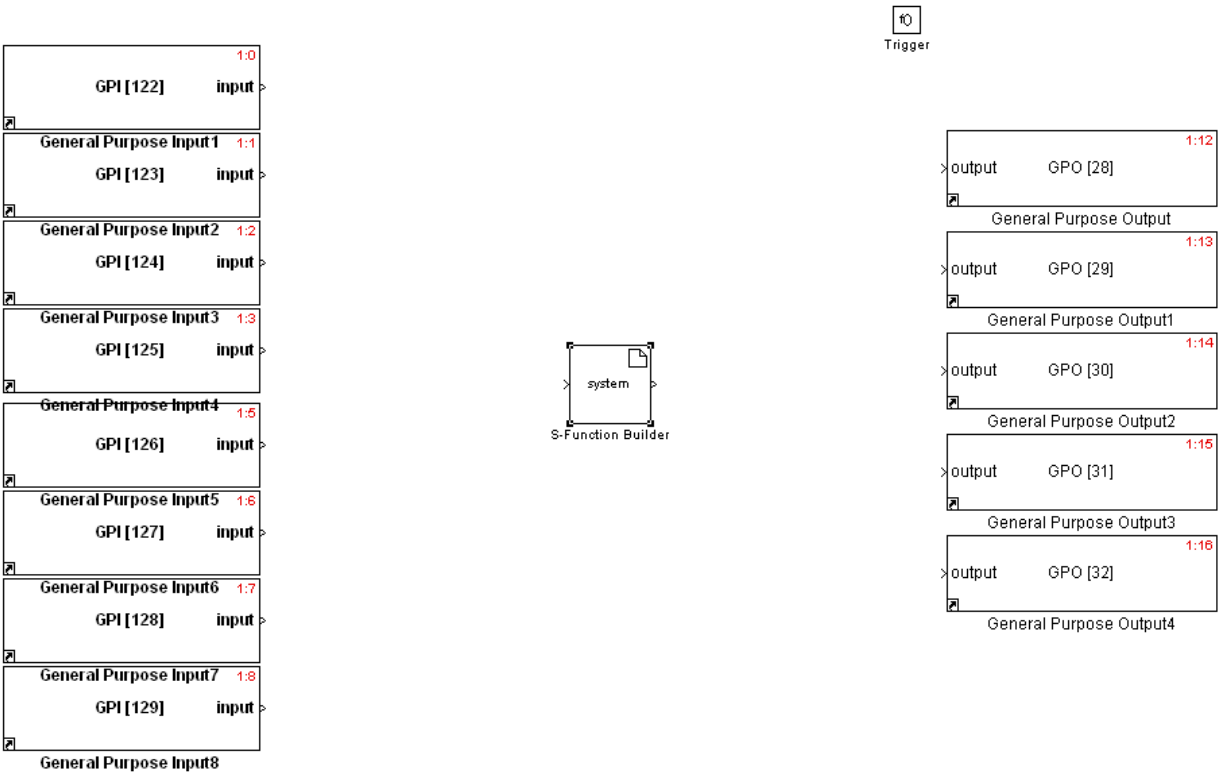
3. In the periodic task, instantiate 12 GPIO bits. Set up bits 122-129 as input bits and bits 28-32 as output. You should now have something resembling the figure below.



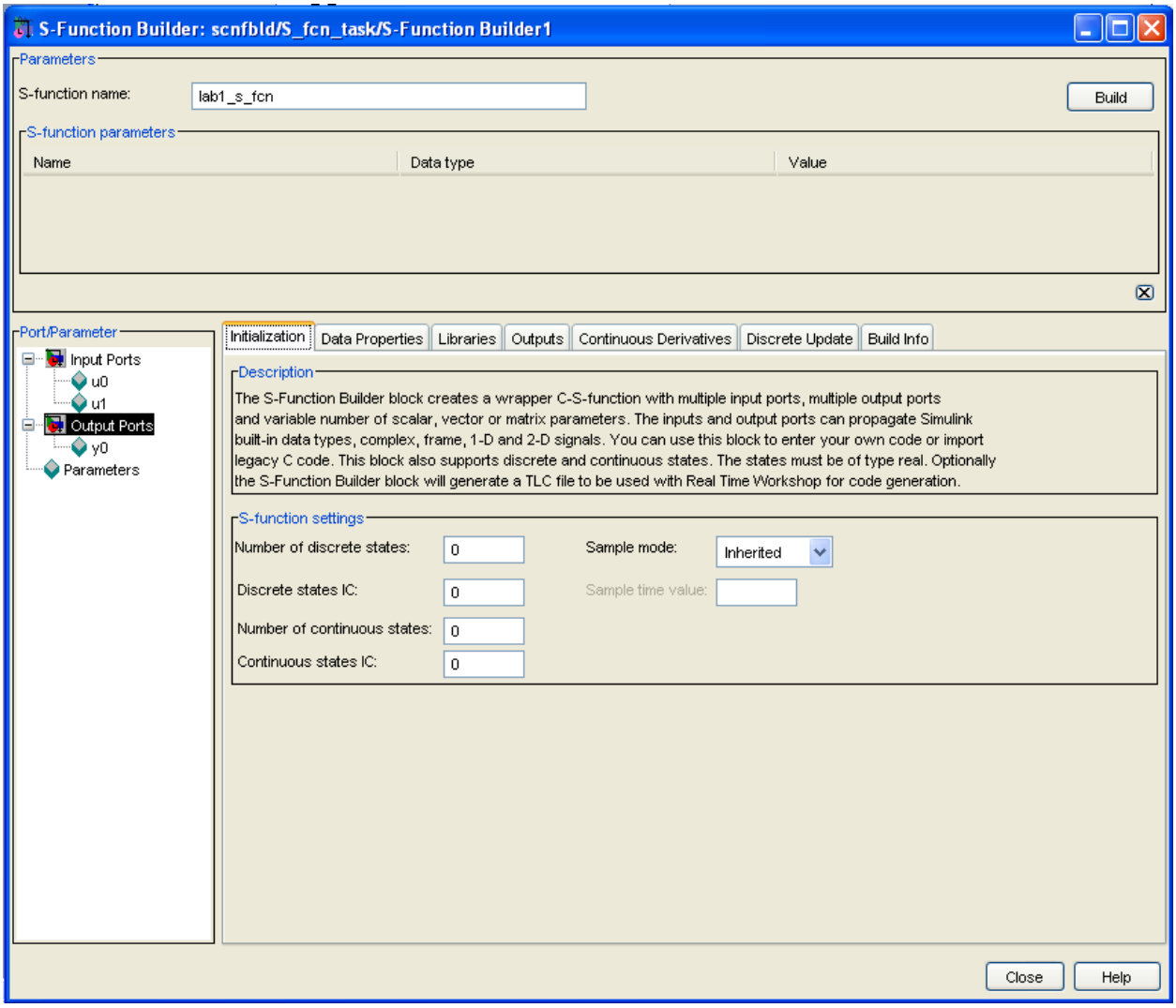
4. Go to the Simulink library browser. Under “User-Defined Functions” you should see a block called “S-Function Builder” (see the figure below).



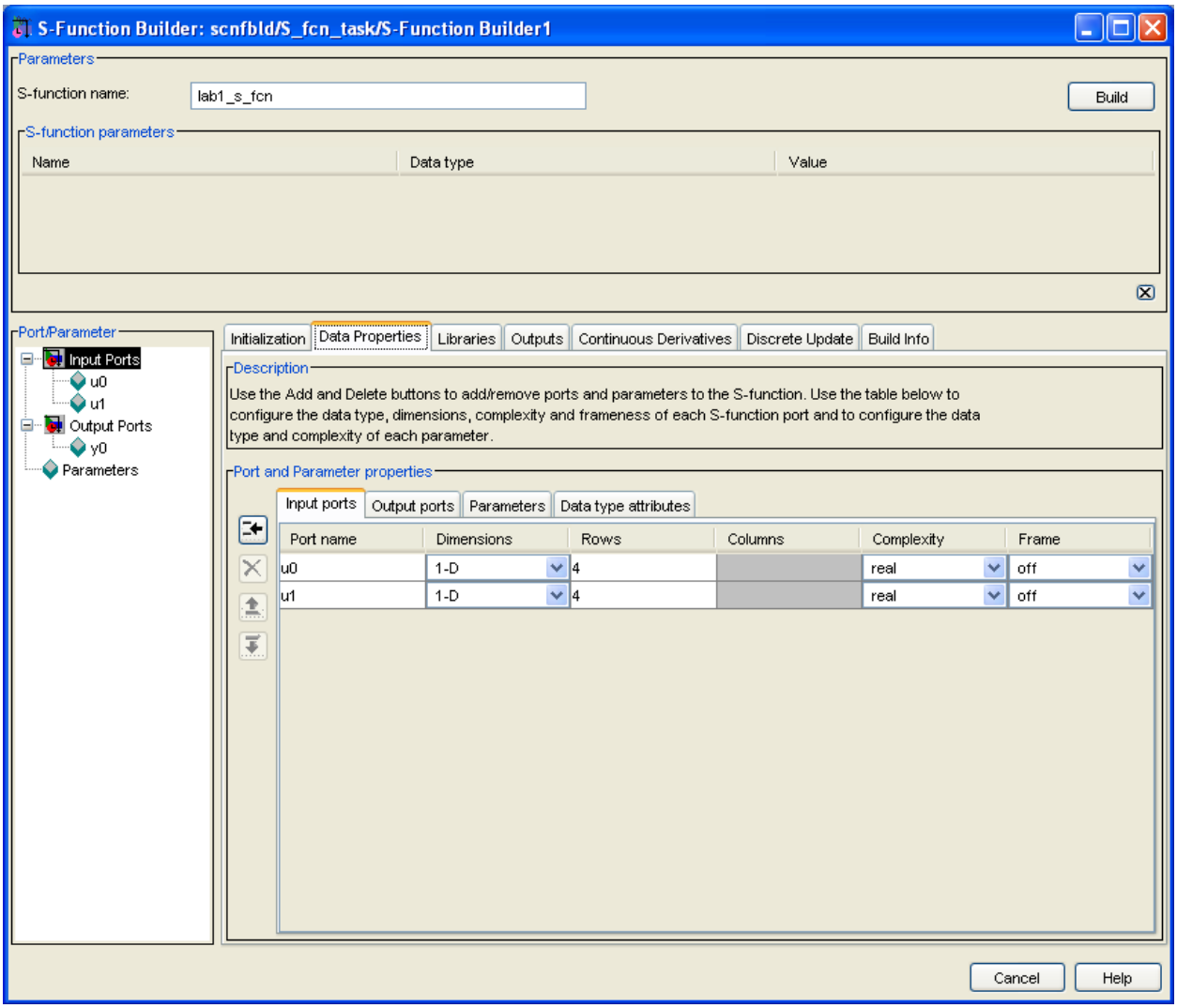
5. Instantiate an S-Function Builder block in your model, as seen in the figure below.



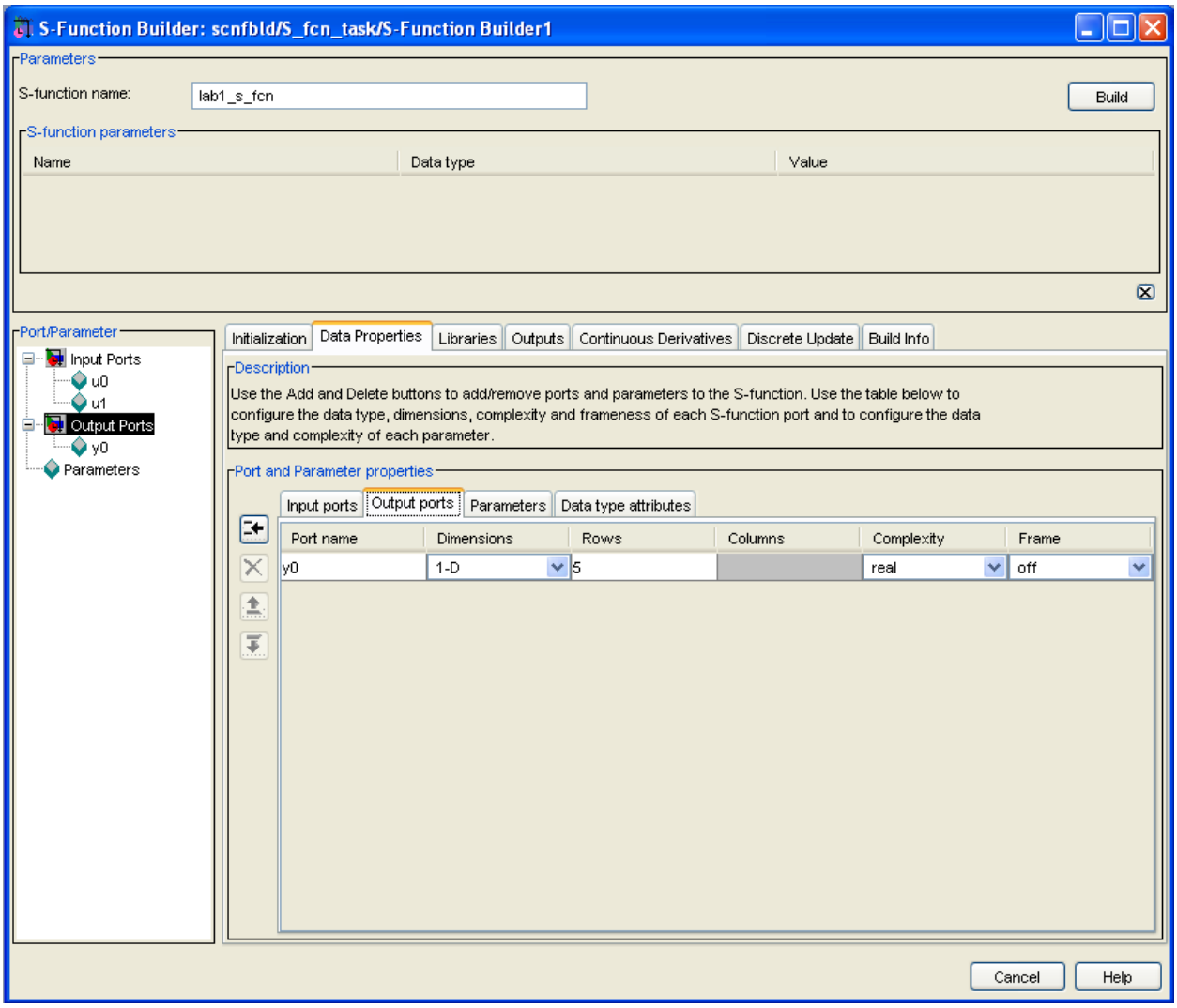
6. Double-click on the S-Function Builder block. In the text box labeled "S-function name," give your new S-Function a suitable name, as seen in the figure below (lab1_s_fcn in this example).



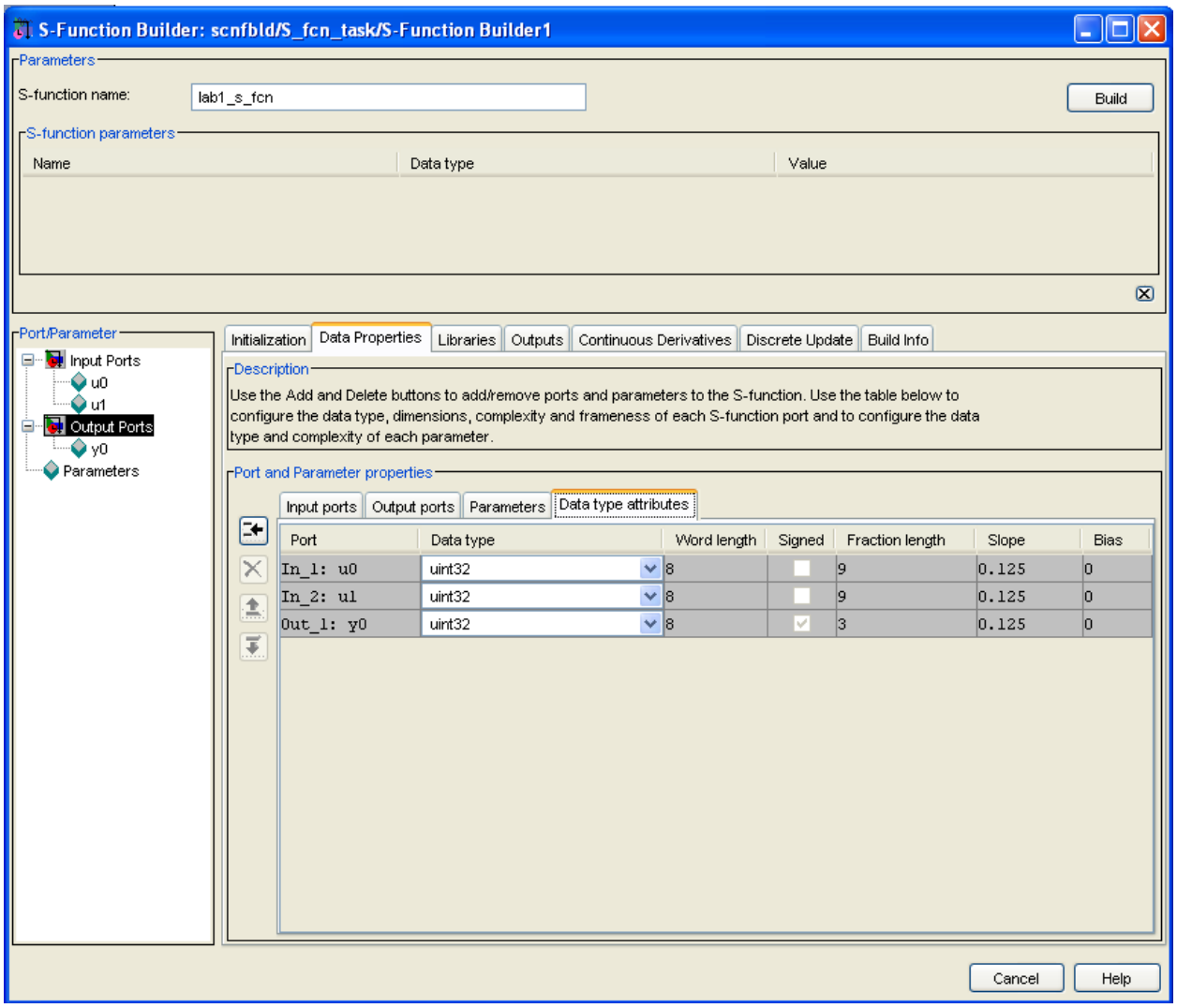
7. Click on the “Data Properties” tab and then on the “Input ports” tab. This is where you will specify the names, types and sizes of your function’s inputs. Create two input ports, u0 and u1. These should be 1-D vectors and have 4 rows, as seen in the figure below. These settings mean that your S-Function will receive 2 vector inputs, each containing 4 elements.



8. Now click on the “Output ports” tab. This is where you will specify the names, types and sizes of your function’s outputs. Create an output port, y0, a vector containing 5 elements as seen in the figure below.



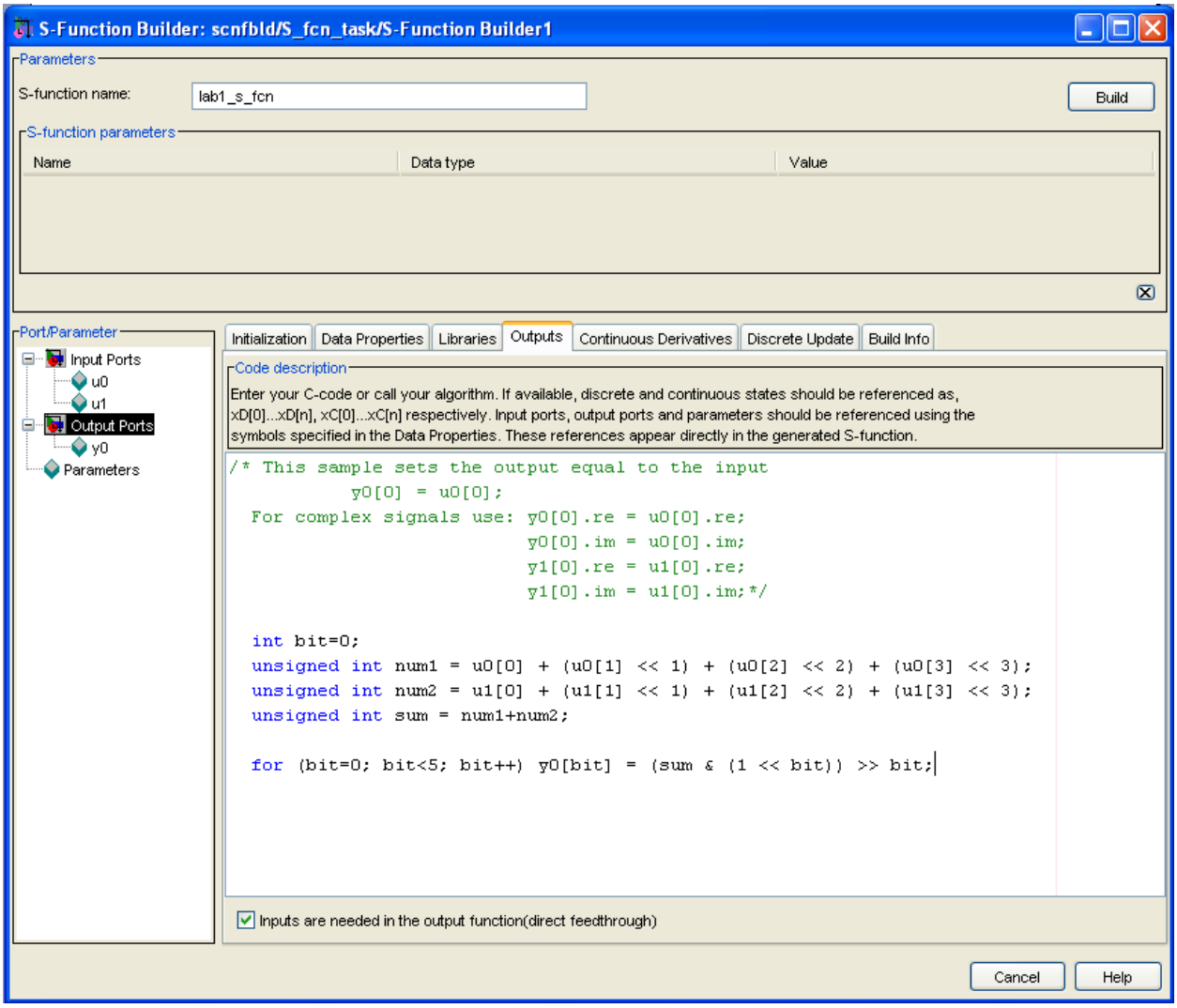
9. Click on the “Data type attributes” tab and select a data type of “uint32” for the inputs and outputs.



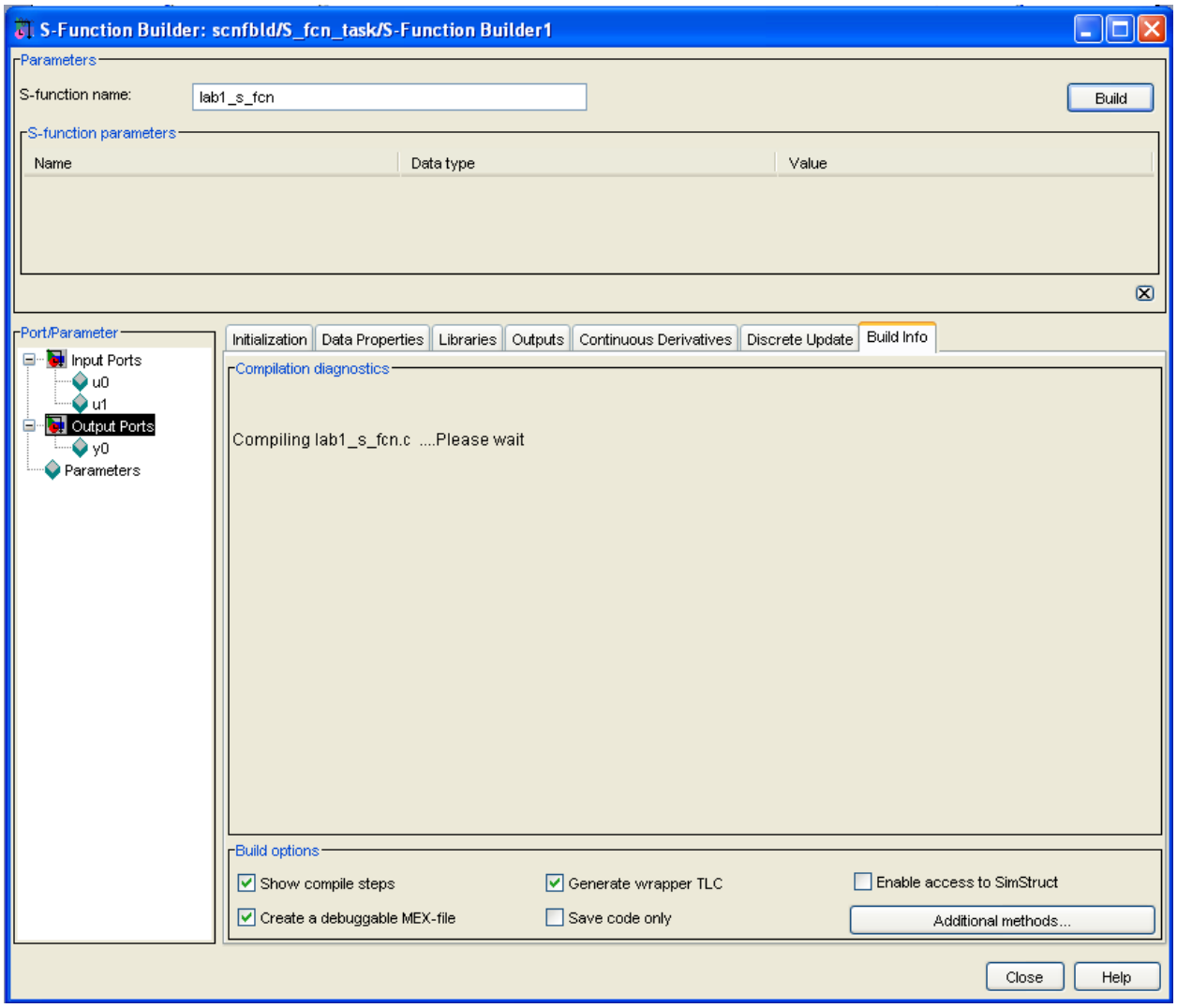
10. Click on the “Outputs” tab. This is where you will write the C code that connects your function’s inputs to its outputs. Each element of the input vector can be accessed by indexing into the vector the same way you would do in C. For example, if you wanted the first output element to be the logical OR of the first input element of each input, you would write:

```
y[0] = u0[0] | u1[0];
```

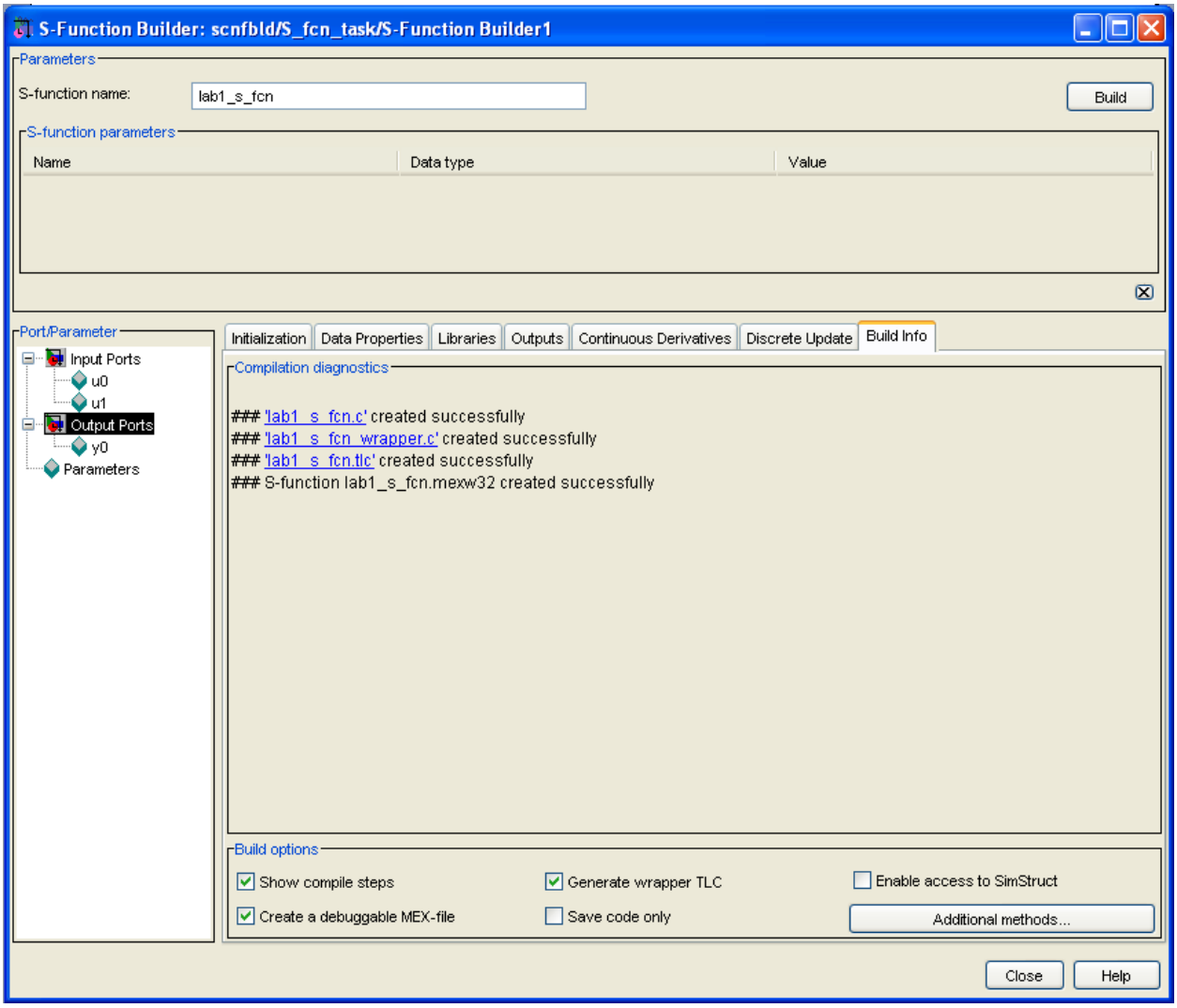
Sample code that adds the numbers represented by the 2 input vectors together and writes the sum to the output vector is shown in the figure below. You may use this code for your S-function or write your own version that has the same functionality.



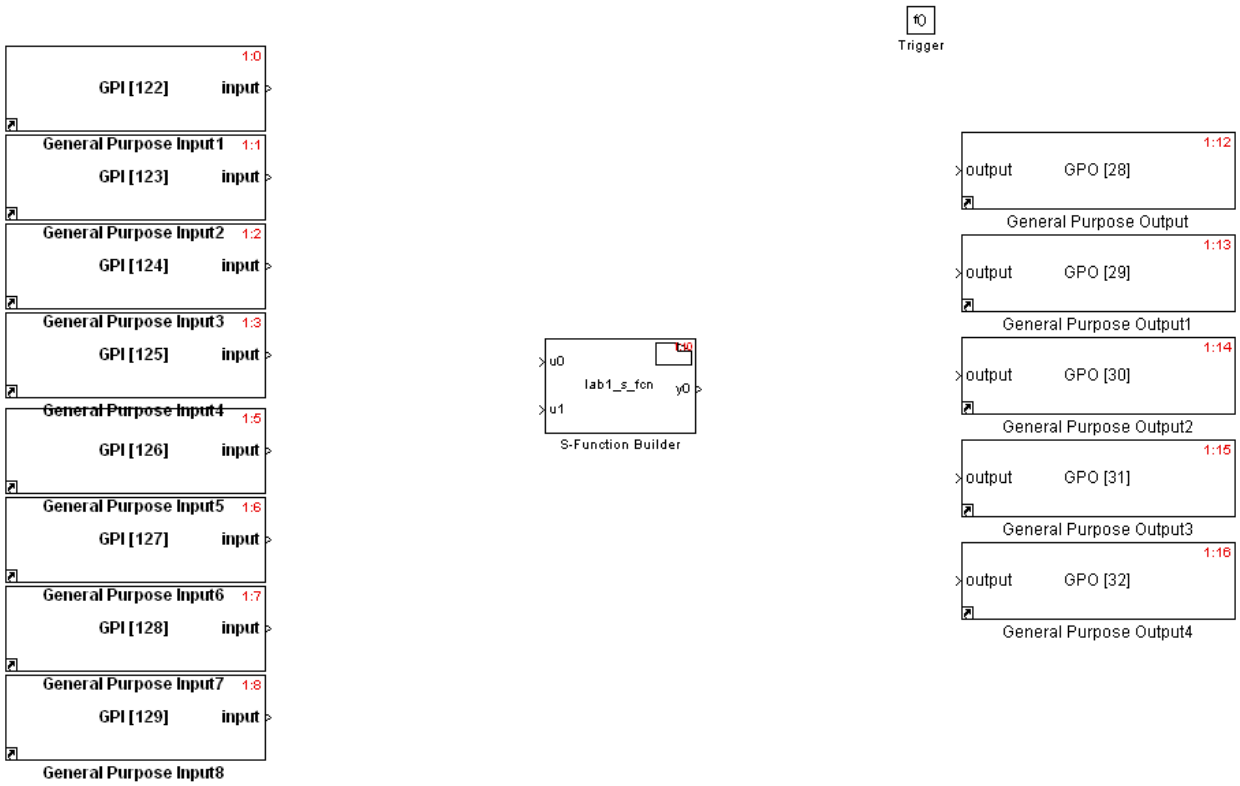
11. You are now ready to build your function. Click on the “Build Info” tab. Check the boxes named “Show compile steps,” “Create a debuggable MEX-file” and “Generate wrapper TLC” as seen in the figure below. Now click “Build.”



12. If the build process was successful, your window should look similar to the figure below.



13. Your S-Function block should now be updated with the input and output ports, as seen in the figure below.



14. Fill in the necessary routing (muxes, casting blocks, etc.) to connect the S function to your digital IO blocks, as seen in the figure below. You may now build the model for use on the MPC 5553 by pressing *ctrl-B*.