Lab Assignment 5
Introduction to Simulink

Lab 5.0 Introduction

In this lab we will design a circuit capable of adding two binary numbers, and we will use the Simulink tool to simulate and validate its behavior. This tool allows you to construct complex circuits with a hierarchical schematic design, which you can then test with different artificial inputs. In the following lab sessions, we will use Simulink to also upload our designs into the FPGA of the ZedBaord.

Accessing Simulink in MATLAB

On the Windows *Start* menu, find MATLAB by browsing *All Programs*, or by typing it in the search box. If available, use version MATLAB 2014a. On the top of the main MATLAB window, you should see three toolbar tabs, titled *Home*, *Plots*, and *Apps*. In the *Home* tab, click on the icon ▦, titled *Simulink Library*. This will launch Simulink.

Pre-Lab Assignment
The following reading list will help you to complete the pre-lab assignment. The readings will also help you in subsequent lab assignments.

Require Reading: Simulink Tutorial in blackboard - Chapter 1 only
https://blackboard.neu.edu/

Please submit the solution to the following pre-lab assignments in PDF format via blackboard.

a) Go through the first chapter of Simulink tutorial [1] to design your half adder. Name the design HAdder. Include a screenshot of your resulting Simulink model as part of the pdf submitted in your pre-lab writeup.

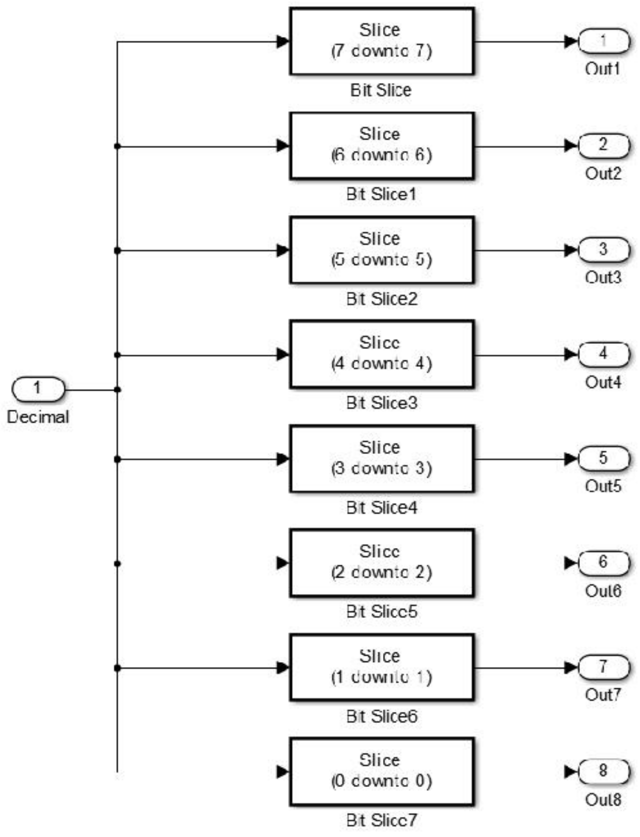
b) Design a 1-bit full adder with three 1-bit inputs (2 inputs and a carry in) and two 1-bit outputs (a sum and a carry out) with logic gates in Simulink. Include a screenshot of your design in your pre-lab report

Lab Assignment

Lab 5.1 Extracting bits from a bus

Our first logic block design will consist in a simple bit-slicer circuit. This is a circuit that takes a multi-bit value (also called a *bus*) as an input, and slices it into its individual bit signals.

1. On the main Simulink window, titled *Simulink Library Browser*, create a new Simulink model by clicking on the *New Model* icon. From this window, you can always go back to the previous window by clicking on the *Library Browser* icon.
2. On the *Simulink Library Browser*, double click on the *Sources* library, and find the *In1* component. Drag and drop it into the Simulink model window, then change its name to *X* by clicking and editing its label. We will repeat the process of adding new components to your Simulink model in the following steps.
3. Edit the properties of the new component by double clicking on it. A new dialog box will open containing tabs *Main* and *Signal Attributes*. In the *Signal Attributes* tab, change the data type to *int8*. Also change the *Sample time* to 1.
4. We are interested in extracting each individual bit from the 8-bit input port. A component named *Bit slice* allows us to perform this action. Click on the search bar of the *Simulink Library Browser* window and type *Bit Slice*. The component should be available in the *HDL Coder* library. Again, drag and drop the component into your model.
5. We can configure the behavior of the bit slicer by double clicking on it, and providing the range of bits that we're interesting in selecting from its input. The first bit slicer should be in charge of selecting only bit 7. We can accomplish this by setting both the MSB (most significant bit) and LSB (least significant bit) properties to 7.
6. Copy and paste the bit slicer to create a total of 8 copies of it. Edit each copy of the slicer to extract a different bit in each case. The first bit slicer extracts bit 7, the second extracts bit 6, …, and the last extracts bit 0.
7. Next, add 8 *Out1* components from the library, available in section *Simulink* → *Sinks*. Make sure the data types are specified as “*Inherit: auto*”. Rename the labels of the *Out1* components to *Y0*, *Y*1, …, *Y*7.
8. Connect all components as shown in the figure below, by clicking on the input and output ports and dragging the moving connection.



Lab 5.2 Testing the Design

Simulink allows you to test your designs by providing specific values for the input signals, and inspecting the values available in the output signals.

9. The first step here is creating a logic block for our bit slicer. Select all components (Ctrl+A), right click on them, and click on “*Create Subsystem from Selection*”. Simulink groups all components into a box, and creates a new logic block. You can rename it by clicking on its label and typing *Slicer*.
10. You can edit the internal design of your logic block again by double-clicking on it. Once your enter the box content, you can use the blue arrows on the toolbar to navigate though the design hierarchy. You can also click on the names of the components on the top bar under the icons to return to the top-level design: lab7 ▸ 8-bit adder ▸ Full adder2 ▸
11. Delete all inputs and outputs connected to the box. Add a *Constant* component from the library, available in section *Simulink* → *Source*, and attach it to the input of the slicer. Then add 8 *Display* components, available in section *Simulink* → *Sink*, and connect them to the outputs of the slicer. Change the data type of the *Constant* component to *int8*, and assign any decimal value you want. Verify that the *Display* components show a number in binary that is equivalent to your decimal input.
12. You can simulate your design by clicking on the *Run* icon.

Assignment 1

Simulate your design and verify its correctness.

a) Take a screenshot of the simulated design after running it, and include it in your report.

b) Describe the outputs displayed by Simulink and discuss whether they show a correct behavior.

Lab 5.3 Merging multiple bits into a bus

We will now design a new logic block with the opposite behavior, that is, one that combines individual signals into a bus interpreted as an *int8* value. Simulink already has a logic component that does this for us, named *Bit Concat*. Find the logic block in the Simulink library, edit its properties to the appropriate values, attach *Constant* blocks with the appropriate type as inputs, and attach a *Display* block to its output.

Assignment 2

Test the bit-concatenate block by assigning different values to the constant blocks and simulate it multiple times.

a) Take a screenshot of one of your simulations and include it in your report.
b) Describe the observed behavior and justify its correctness.

Lab 5.4 The half adder

A half adder is a circuit capable of adding two bits. The circuit has two inputs, *x* and *y*, each interpreted as a separate 1-bit binary number. The largest result of the addition is $2_{10} = 10_2$, in the case that both inputs are 1. This means that we need two outputs, one for the least significant digit (called the *sum*, or simply *S*), and one for the most significant digit (called the *carry*, or simply *C*).

Design a half adder in Simulink (or use the one from your pre-lab) and pack it into a logic block called *Half Adder*. You can find all logic gates under one common component in *Simulink → Logic and Bit Operations → Logical Operator*. Once you drag and drop a logical operator, you can edit the exact operation by double-clicking on it and modifying the corresponding property.

Assignment 3

Take a screenshot of your half adder design and add it to your report. Explain how the logic gates solve the problem of adding two 1-bit numbers by showing the truth tables for the *S* and *C* outputs of the half adder.

Lab 5.5 The full adder

A full adder is a circuit capable of adding three 1-bit numbers. It takes three inputs, named *x*, *y*, and *z*, and it outputs a 2-bit binary number, capable of representing the largest possible result of the sum, which is $3_{10} = 11_2$. Like in the half adder, the least significant bit of the result is labelled *S*, and the most significant bit is called *C*.

Assignment 4

Implement the full adder in Simulink, using two half adders (designed above), and pack it into a logic block named *Full Adder*. Take a screenshot of your design and include it in the report. Describe the implementation of the full adder.

Lab 5.6 The 8-bit binary adder

We are now ready to combine all previously designed components in order to build a complete 8-bit binary adder, a circuit that is capable of adding two 8-bit binary numbers.

1. Using your previous designs, develop an 8-bit adder. The model will receive two inputs (int8) and calculate the sum.
2. Add the necessary logic to your adder to detect overflow.
3. Validate your design with examples, some should produce overflow, and some should not. Report experiments and your results.

Assignment 5

Design an 8-bit binary adder in Simulink. Attach two bit slicers to inputs A0...A7 and B0...B7, and connect a bit-concatenate component to outputs S0...S7. Connect two *int8* constant blocks to the inputs of the bit slicers, and a display to the output of the bit-concatenate component. Connect a 1-bit constant block to the *Cin* input, and another display to the *Cout* output. Simulate your design with meaningful values for the inputs, and verify the correctness of the outputs. Some inputs should produce overflow, and some should not.

Take a screenshot and add it to the report. Add a description of your design and justify the correctness of the simulation results. What is the purpose of output signal *Cout*?

Lab 5.7 Extra Credit

Design a multiplier using the logic blocks you designed in the previous parts. The design will receive two inputs (int8) and calculate the product.

Consult the "Combinational Multiplier" document on Blackboard for extra help.