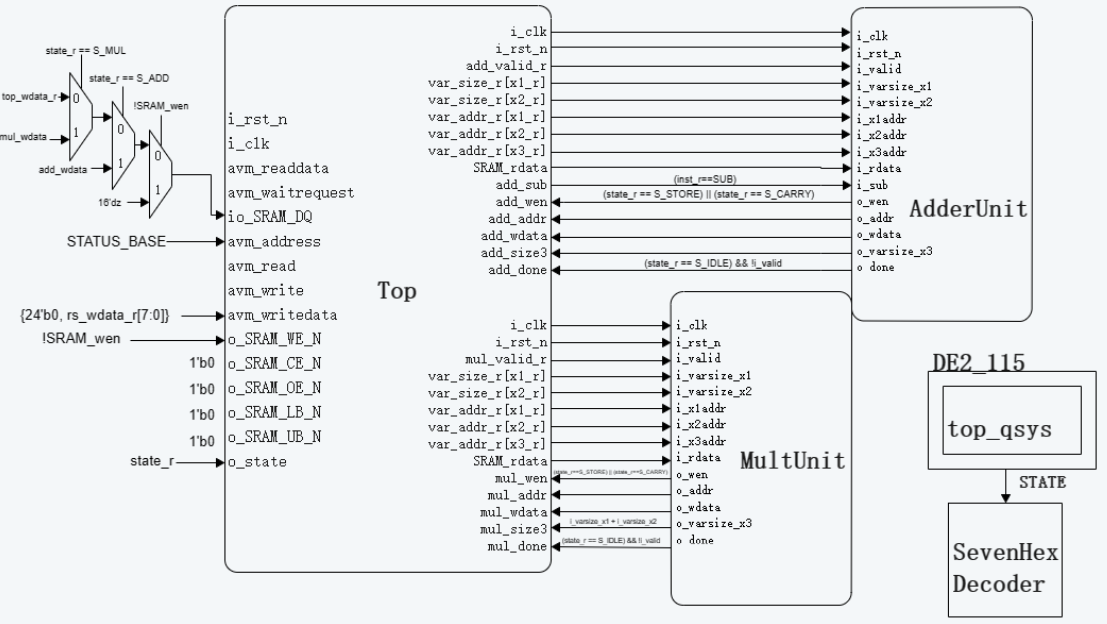
Team09\_Final\_Project\_Report

File Structure:

* team09\_final\_project/team09\_final\_project\_report : This file contains information about the source code in the directory and the instructions of final project.
* team09\_final\_project/src/Top.sv : This file is the top module that implemented arbitrary-precision integer calculator.
* team09\_final\_project/src/DE2\_115 : For the files in this folder, they are the DE2\_115 related files.
* team09\_final\_project/src/pc\_python/rs232.py: This file contains the implementations of import input file, export result , translate the instruction and check the golden case.
* team09\_final\_project/src/pc\_python/tb\_1.txt: This is the input file.

System Architecture:

Hardware Scheduling:

1. Top.sv:

1. Request instruction
2. Determine the instruction likes add, store, load, etc
   * If arithmetic,
     1. Request variables
     2. Transform the variables to submodule
   * If store,
     1. Request variable size
     2. Request variable value
   * If load,
     1. Send variable size
     2. Send variable value
3. Repeat again

2. Workflow:

1. Run Verilog code into FPGA
2. Press KEY0 to reset(reset each time)
3. Open the cmd on pc
4. “cd YOUR\_ADDRESS/team09\_final\_project/src/pc\_python”
5. “python rs232.py COMx” for x is the port of the rs232
6. It will generate the “dat.bin” that contains the variable size and the result value

3. More options:

1) You can modify the tb\_1.txt

2) Instruction rule:

add “variable number2” “variable number0” “variable number1”

sub “variable number2” “variable number0” “variable number1”

mul “variable number2” “variable number0” “variable number1”

load “variable number0”

store “variable number0” “variable size”

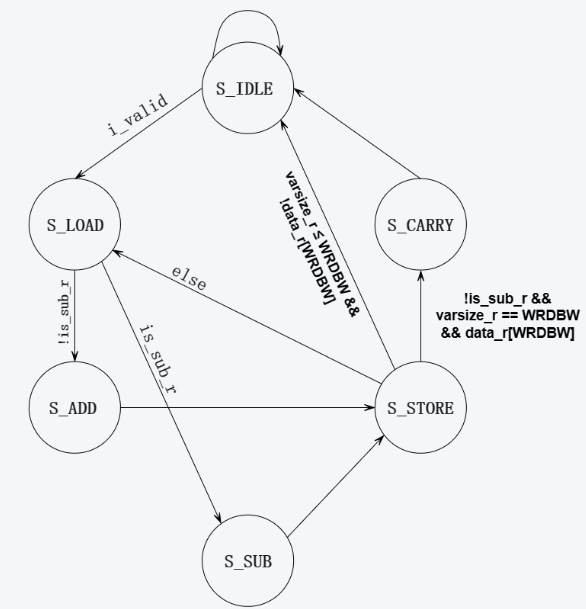
e.g.

store 0 16 import x0

store 1 8 import x1

sub 2 0 1 x2=x0-x1

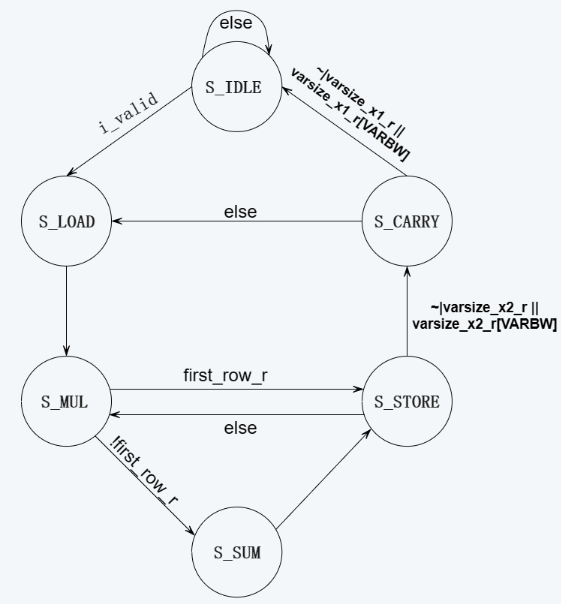
load 2 export x2



Algorithms:

Addition : Carry ripple adder

Subtraction : Similar as addition, but one of the integers transforms to 2’s complement integer(we expected x1-x2, where x1>x2)



Algorithms

Multiplication : Standard Algorithm(long multiplication)

Fitter Summary:

Timing Analyzer:

Problem & Solution:

* The gate level design occurs some latches
  + Check if there are any unassign signals in the combinational circuit.
* Signals struck in GND or fanout
  + Check if no declare logic or not enough bits.
* Generating the input data
  + Using python to generate bytes file from hex.
* Wrong variable size
  + Cut variable size address properly

Conclusion & Suggestion:

The journey through this semester’s digital circuit lab(dc-lab) has been improved our Verilog skill and a deeper understanding of using the fundamental components of FPGA, and practical applications lab.

Bridging theory with practice was a cornerstone of this lab. We also learn how to build a system architecture and design a FSM for each sub-module.

The challenges encountered during the dc-lab propelled the development of robust problem-solving abilities. Debugging and optimizing designs demanded analytical thinking and perseverance, skills essential in any technological endeavor.

The collaborative nature of this project enhanced our ability to work effectively within a team. Sharing ideas, and navigating through challenges collectively fortified the learning experience.

Finally, we suggested that provides an additional lecture for explaining some useful FPGA components such as SDRAM, NiosII, and etc. Moreover, we encourage to enhance the integrity of DE2\_115 documents and resources to students having a better problem-solving tools. Thank you so much and Happy New Year.