

CECS 360 Fall 2016 Project 3

Designing A Testbench #2: Verification

By

Shinesunny Som

11/13/2016

Instructor: John Tramel

**Lab Description:** A continuation of the previous project which put emphasis on designing testbenches, this project will require you to demonstrate how to create a test bench that will utilize vectors for the stimulus and vectors for analyzing the results, thus “self checking” verification. We will use the technique discussed in class using high level language to create the input stimulus and generate the expected results.

**Verification Report:** First and for most, understanding what the module is suppose to do. Given schematic and lab description it is a module that perform a multiply-accumulate function. (Multiply Accumulate circuit.) Of the many choices of the HLL to module the testbench from, I chose C++/C. Modeling the verilog code with C++ with correct syntax was not too difficult to do. The next part however, depending on competency of HLL courses, may take a while to create the array of memory and taking the input stimulus into the verilog inputs.

**Results:**

Aligning my results from the array to match the input stimulus took time. Verilog, when it reads the array has difficulty reading at blank spaces. After fixing that I begin clearing my accumulator from my C code and being able to match it with my Verilog test bench code. The accumulator in C not being cleared at the same time that the Verilog was being cleared brought timing issues. Nevertheless, the testbench results matches with the output.txt I created from the HLL.

**Problems/Issues:** Taking only one semester of C++ and not execising as much caused me to do extensive reinforcement. Of course, no excuses. The pseudocode I wrote for my vectors in Visual Studio was giving me plenty of errors, so I migrated using the C compiler on Linux and checked to see consistency of my values which I can verify. This project has cause me to do plenty of things beyond my comfort zone: aligning the array results to match and with verilog, and relearning C, and having to print out a text took me a while to learn.

**C++/C source:**

*Visual Studio 2015, Linux C complier.*

// ConsoleApplication1.cpp : Defines the entry point for the console application.

//

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <iostream>

#include <fstream>

int main()

{

FILE \*ptrfile;

int x = 0;

unsigned short a = 0;

unsigned short b = 0;

unsigned short oldResult = 0;

unsigned short multA = 0;

unsigned short multB = 0; // check

unsigned short adderOut = 0;

unsigned short clearAccum = 0;

//unsigned short reset;

time\_t t;

/\* Intializes random number generator \*/

srand((unsigned)time(&t));

ptrfile = fopen("output.txt", "w");

if (!ptrfile)

return 1;

//check

for (x = 1; x <= 10; x++)

{

if (clearAccum == 0)

{

oldResult = adderOut; //check

}

else

oldResult = 0;

//rand functions modulo 256

a = rand() % 256; //255

b = rand() % 256; //255

multA = a \* b;

adderOut = oldResult + multA;

//blank space

//edit format later

fprintf(ptrfile, "%x\r\n\r\n%x\r\n\r\n%x\r\n\r\n%x\r\n\r\n", clearAccum, a, b, adderOut);

clearAccum = ~clearAccum;

}

fclose(ptrfile);

return 0;

}

**Output.txt:**

*Excerpt:*

*0*

*4e*

*b7*

*37c2*

*ffff*

*fe*

*d2*

*d05c*

*0*

*7b*

*2b*

*e505*

*ffff*

*b*

*6d*

*4af*

*0*

*6a*

*30*

*188f*

*ffff*

*fd*

*ae*

*abf6*

*0*

*66*

*ec*

*9fe*

*ffff*

*61*

*67*

*2707*

*0*

*7*

*5f*

*29a0*

*ffff*

*37*

*fd*

*365b*

**Console output:**

ISim P.20131013 (signature 0x7708f090)

WARNING: A WEBPACK license was found.

WARNING: Please use Xilinx License Configuration Manager to check out a full ISim license.

WARNING: ISim will run in Lite mode. Please refer to the ISim documentation for more information on the differences between the Lite and the Full version.

This is a Lite version of ISim.

Time resolution is 1 ps

Simulator is doing circuit initialization process.

ERROR: Too many words specified in datafile output.txt

Finished circuit initialization process.

multa = 0

multa = 14274

match

dataA from my\_memory:4e

dataB from my\_memory:b7

expected\_value from my\_memory:0000xxxx, adder\_out from DUT:0000

multa = 53340

match

dataA from my\_memory:fe

dataB from my\_memory:d2

expected\_value from my\_memory:000037c2, adder\_out from DUT:37c2

multa = 5289

match

dataA from my\_memory:7b

dataB from my\_memory:2b

expected\_value from my\_memory:0000d05c, adder\_out from DUT:d05c

multa = 1199

adder\_out HIGH

match

dataA from my\_memory:0b

dataB from my\_memory:6d

expected\_value from my\_memory:0000e505, adder\_out from DUT:e505

multa = 5088

match

dataA from my\_memory:6a

dataB from my\_memory:30

expected\_value from my\_memory:000004af, adder\_out from DUT:04af

**ISim>**