Tyler Kim

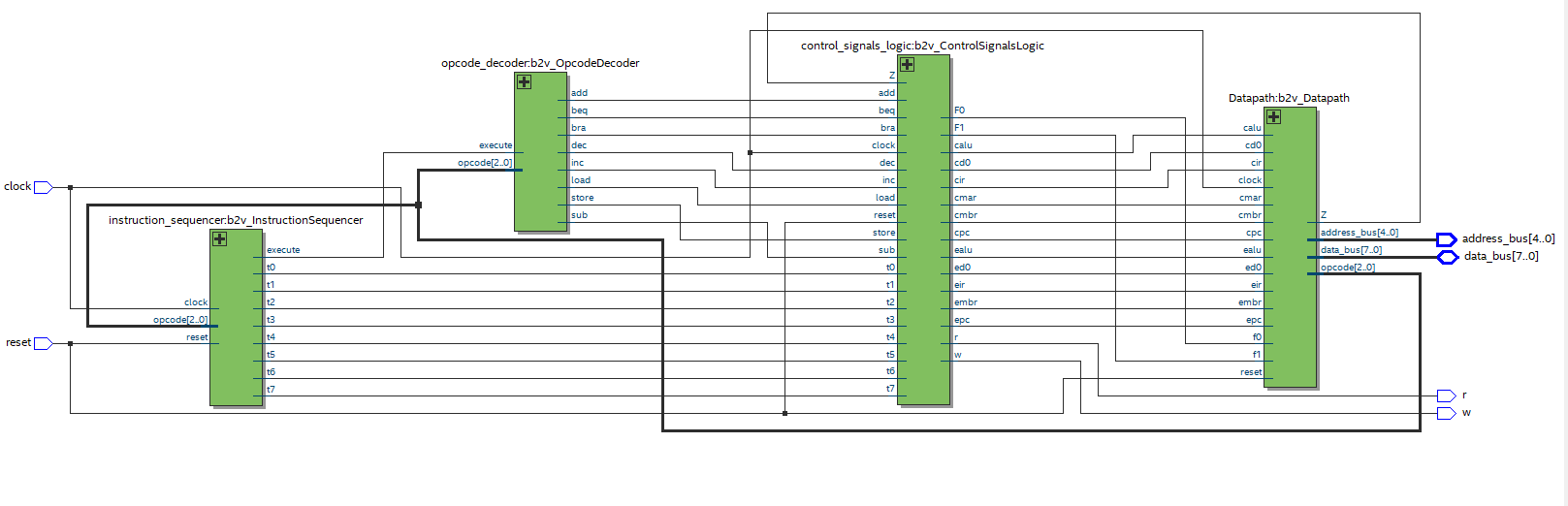
Digital Logic Design

***Simple CPU***

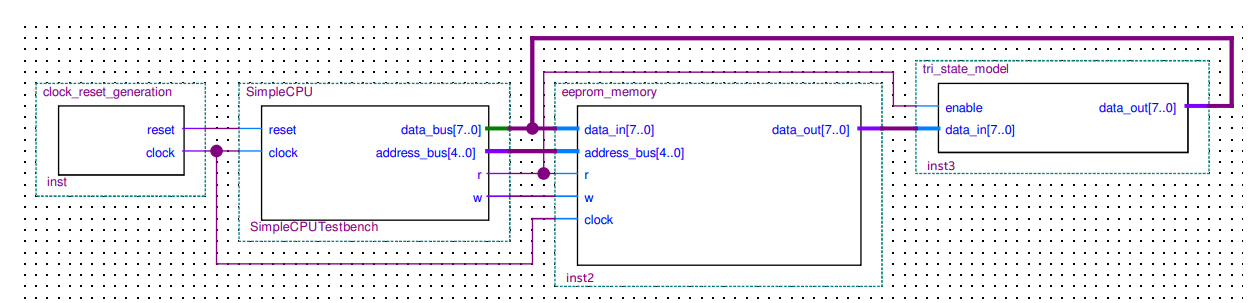
**Description**

The purpose of this learning activity is to test the final CPU after combining all the other components made before. More specifically, I was tasked to create a program to test all the functions of the CPU as a whole.

**Schematic**



**Figure 1: Simple CPU Schematic**

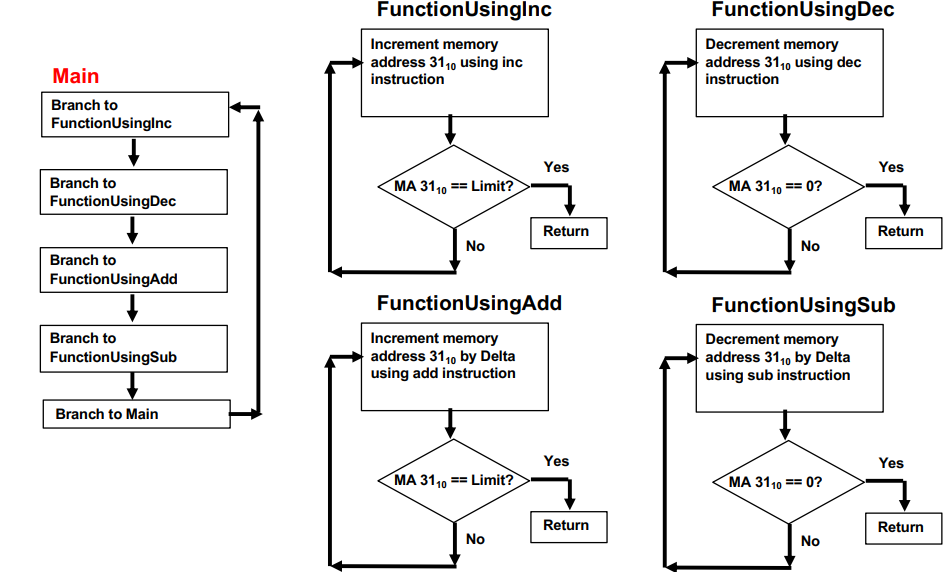
****

**Figure 2: Simple Computer System**

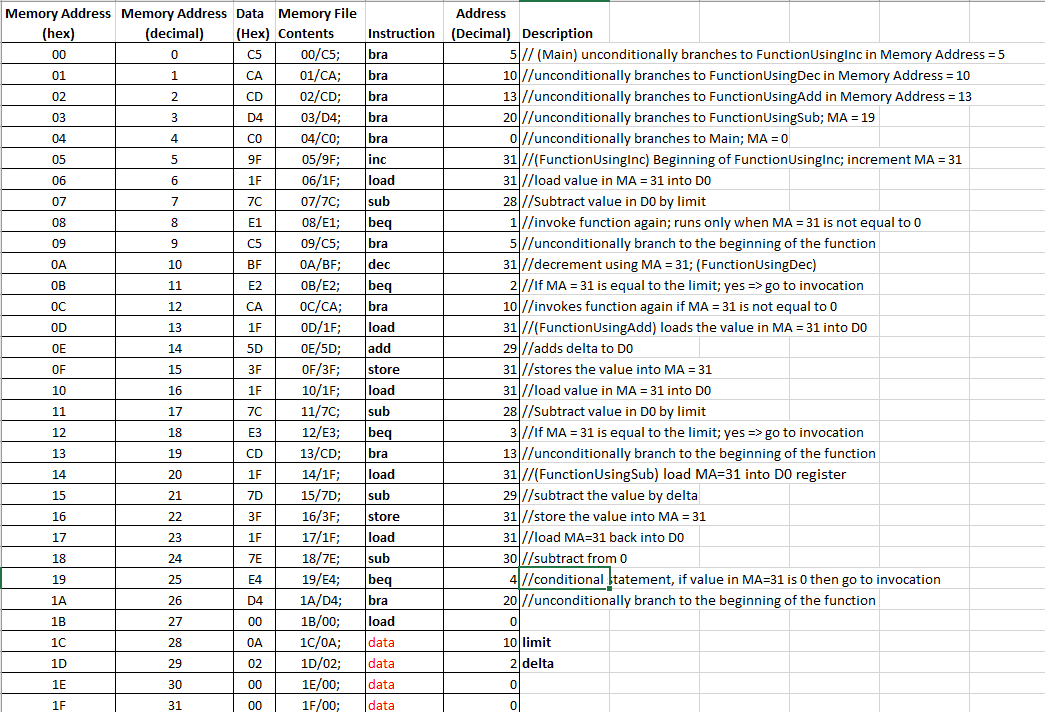
The schematic in Figure 1 illustrates the schematic for the simple CPU only using the components created from previous learning activities. The schematic in Figure 2 shows the entire computer system with the clock reset generation, CPU, memory, and tri-state model.

**Program Explanation**

The program used to verify the functionality of the simple CPU follows a simple logic.



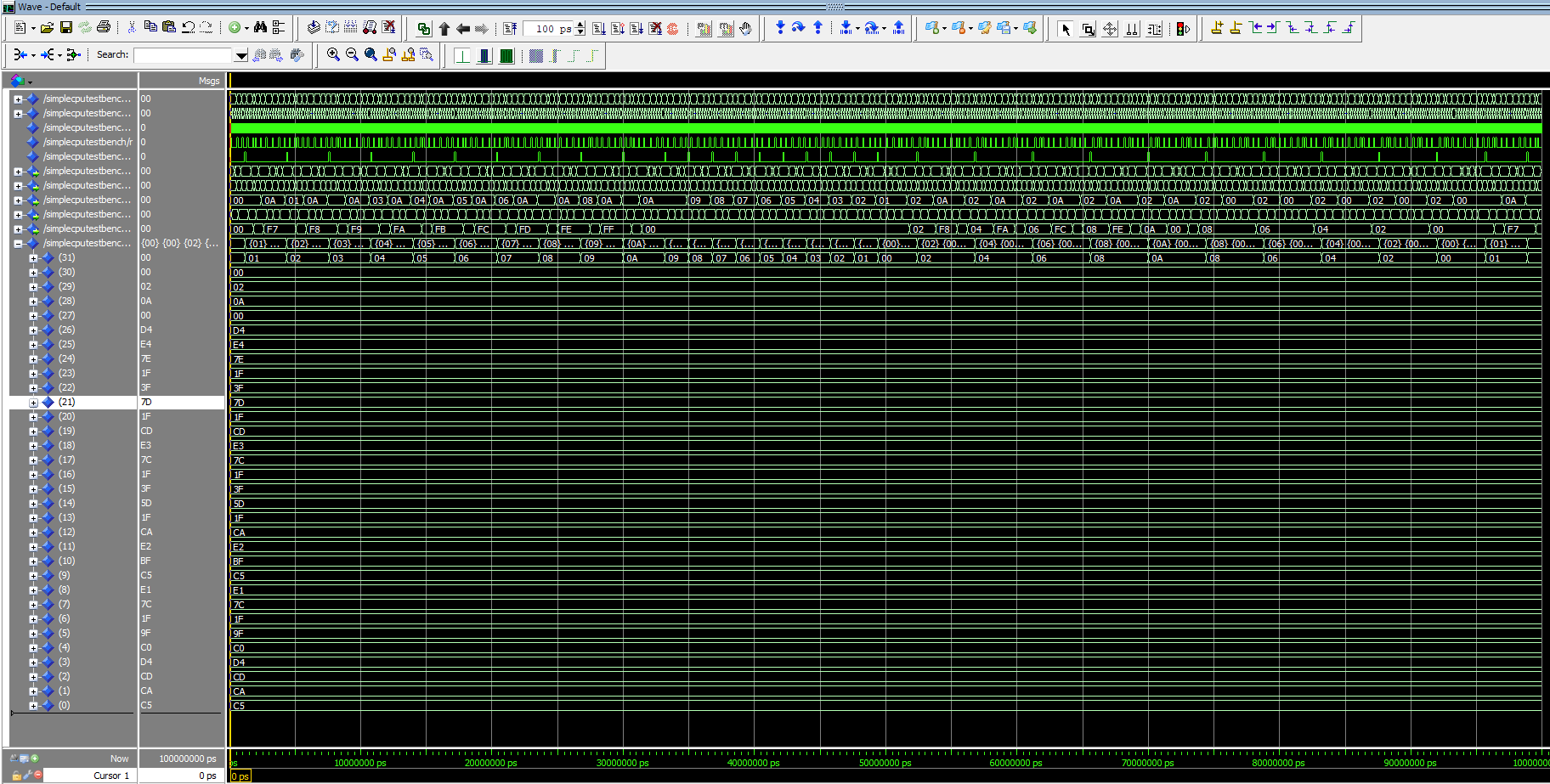
**Figure 3: Flowchart of Program**

****

**Figure 4: Implementation in Excel**

Figure 3 shows a simple flowchart of the logic of the program used to verify the functionality of the CPU and Figure 4 illustrates the actual implementation. In short, the program has four functions and two constants: delta and limit. In the case of my particular implementation, delta and limit were set to 2 and 10 respectively. The program first tests the *FunctionUsingInc* function where it increments the value in memory address 31 until it reaches the limit. When it reaches the limit, it decrements the value until it reaches 0 through *FunctionUsingDec*. Then the program increments the value in memory address 31 by delta using *FunctionUsingAdd* until reaches the limit and then decrements by delta using *FunctionUsingSub* until it reaches the value of 0.

**Verification**

****

**Figure 5: Overall Verification Test**

Figure 5 shows the overall verification tests of the functionality of the program and the functionality of the CPU where memory address 31 is the variable that was transformed by the operations the most. To further illustrate the success of the functionality of the program and CPU, the verification section in this report is divided into four additional parts for each function.

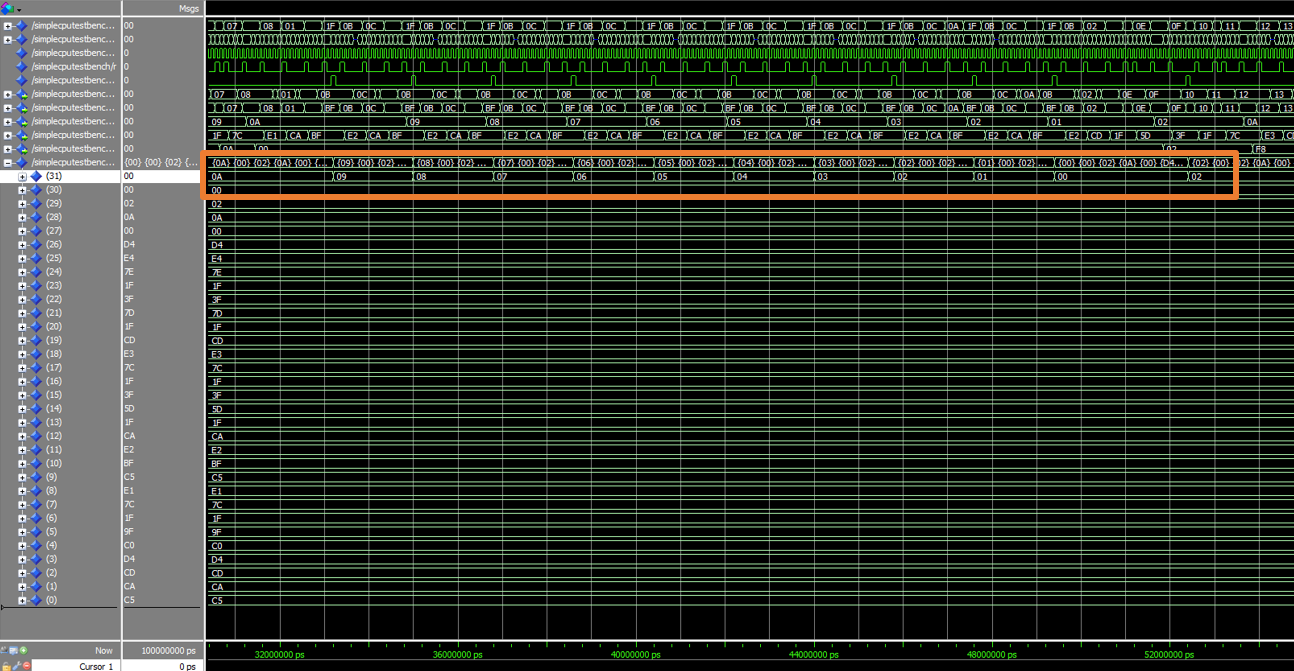
FunctionUsingInc

****

**Figure 6: FunctionUsingInc Verification**

Figure 6 shows the *FunctionUsingInc* works correctly because the value in memory address 31 increases by one every time. When the value in memory address 31 equals 10, it proceeds to begin decreasing showing the start of the *FunctionUsingDec* function.

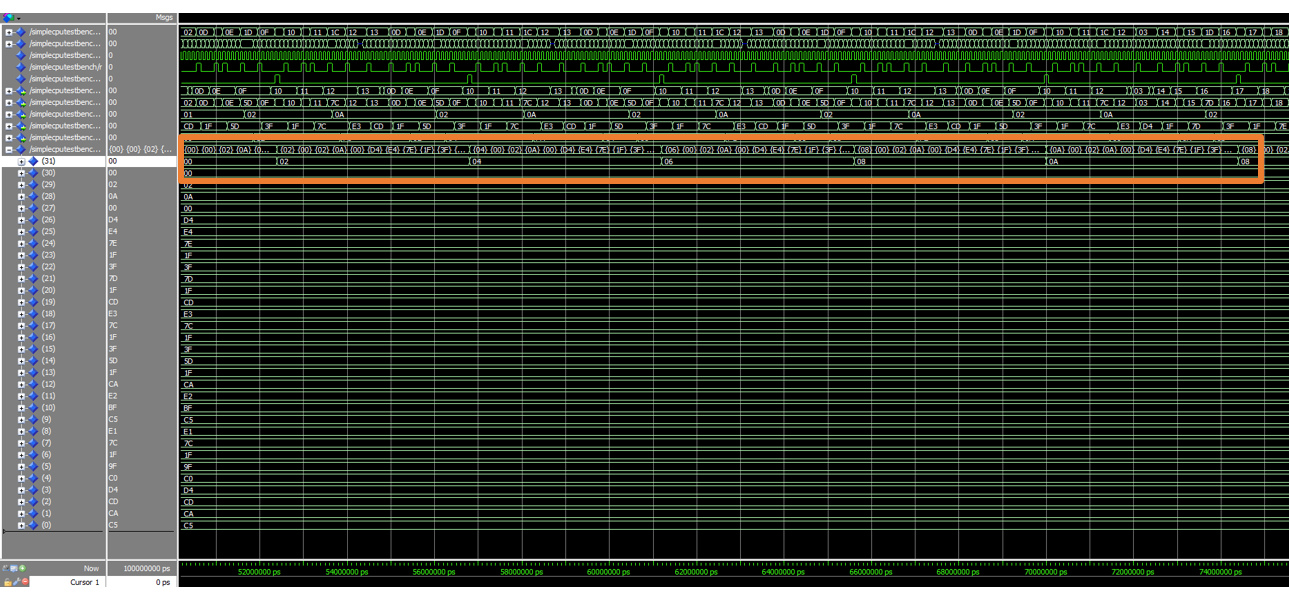
FunctionUsingDec

****

**Figure 7: FunctionUsingDec Verification**

Figure 7 shows that the *FunctionUsingDec* works correctly because when the value in memory address 31 is 10, it begins decreasing until it reaches 0. Then, the value starts transforming by the *FunctionUsingAdd* function.

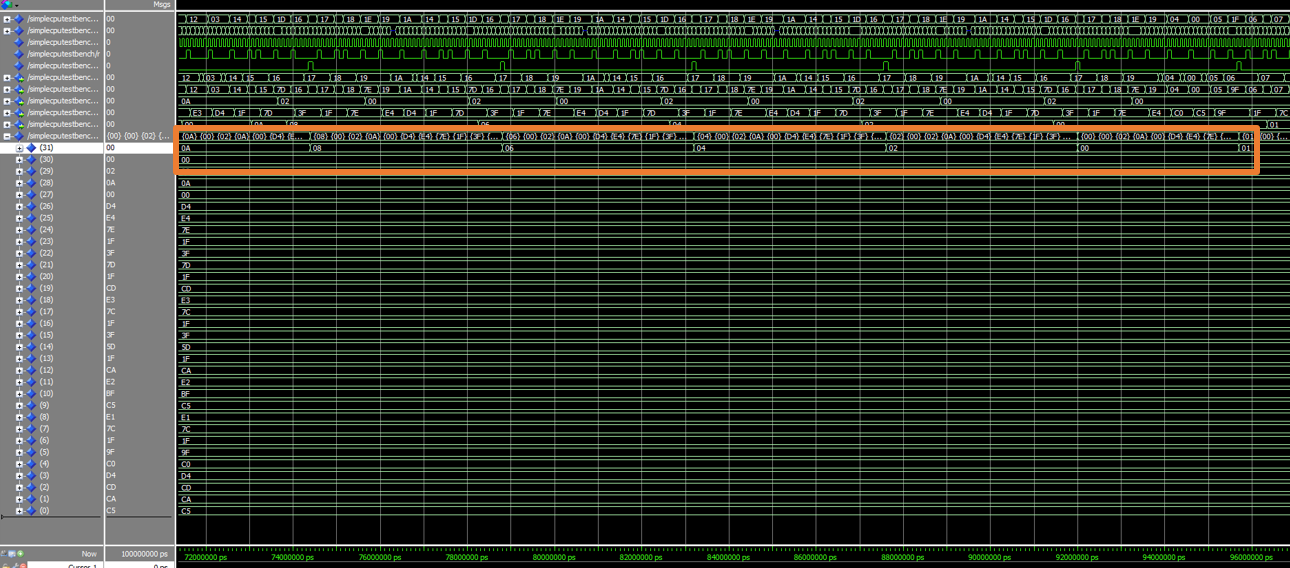
FunctionUsingAdd



**Figure 8: FunctionUsingAdd Verification**

Figure 8 shows that the *FunctionUsingAdd* works correctly because when the value in memory address 31 is 0, it begins incrementing by delta, 2. Then, the value starts transforming by the *FunctionUsingSub* function when the value reaches 10.

FunctionUsingSub

****

**Figure 9: FunctionUsingSub Verification**

Figure 8 shows that the *FunctionUsingSub* works correctly because when the value in memory address 31 is 10, it begins decrementing by delta, 2. Then, the value starts transforming by the *FunctionUsingInc* function when the value reaches 0 since the program will run as a continuous loop.