CMPE 260.01 Laboratory Project 1

Pipelined MIPS Processor

Rhythm Patel

Performed: 04/18/2021

Submitted: 05/04/2021

Lab Section: 1

Instructor: Stephen Moskal

TA: Jacob Myerson, Dennis Lam

Lecture Section: 1

Professor: Richard Cliver

By submitting this report, you attest that you neither have given nor have received any unwarranted assistance (including writing, collecting data, plotting figures, tables, or graphs, or using previous student reports as a reference), and you further acknowledge that giving or receiving such assistance will result in a failing grade for this course.

Your Signature: Rhythm Patel

Abstract

The objective for this lab exercise was to put together all the stages of the MIPS Pipeline DataPath. The stages that were created in the previous lab exercises which were the Instruction Fetch Stage, Decode Stage, Execute Stage, Memory Stage, and the write back stage were put together in a single project and several functions of the MIPS datapath were tested. Aside from just testing the functionality of the MIPS Datapath, a fibonacci program was also created and the first 10 numbers of the fibonacci sequence were calculated from the instructions provided in the instruction memory and then stored in the data memory part of the MIPS memory. The project was successful because all the functions of the MIPS processor were correctly performed and the fibonacci sequence was also successfully calculated.

Design Methodology

The MIPS Reduced Instruction Set Computer Architecture is made up of a multi-stage process which is used to retrieve, break down, and process instructions provided by the user to the memory. The stages included a Fetch Stage, Decode Stage, Execute Stage, Memory Stage, and a WriteBack stage. The purpose of the fetch stage was to parse through the instruction memory and handle each instruction one by one and set it to the decode stage. The decode stage breaks down the instructions into it's appropriate bits and sends to the register file, control unit and other components of the MIPS decode stage. Then the instruction goes to the execute stage where they are processed in an ALU and then the data is stored in the data memory module during the Memory Stage. The WriteBack stage sends multiple signals back to the appropriate components of the MIPS such as the registerfile for the data to be written back to a register. Figure 1 shows the simplified data path of the MIPS processor.

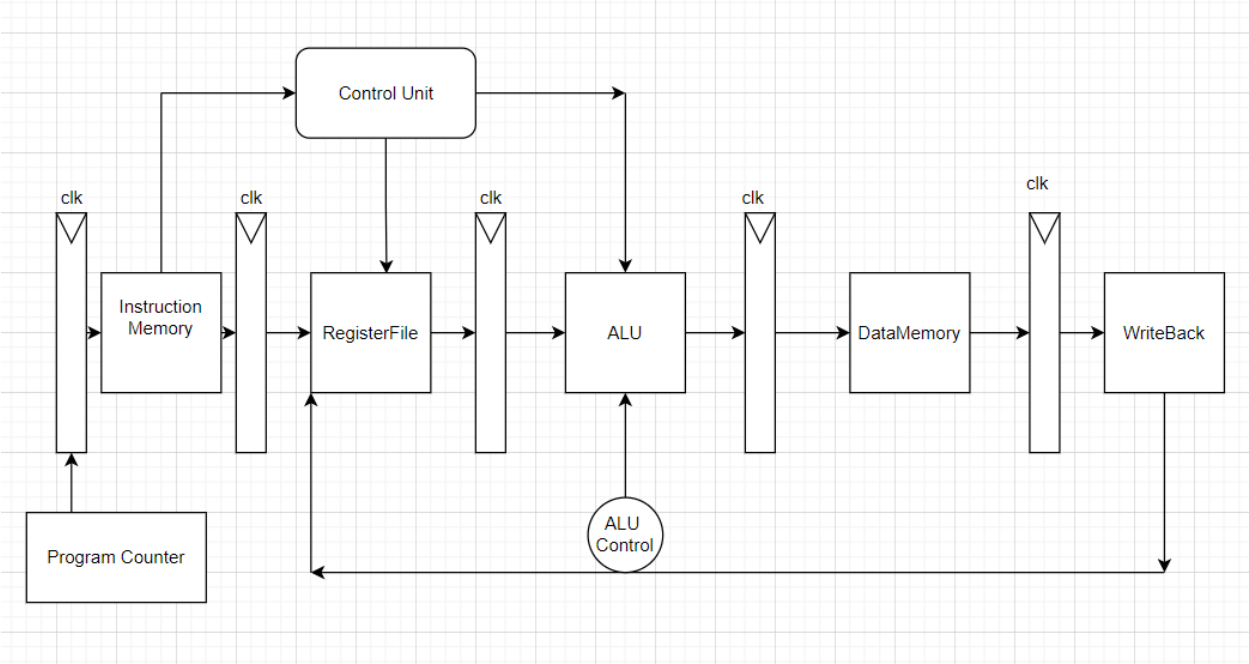

Figure 1: Simplified MIPS Data Path

The fibonacci sequence was written using a series of MIPS instruction code in hex. The first step was to populate two registers with the first two fibonacci numbers which are 0 and 1 and then multiple steps were carried out where the numbers were added using temporary registers and storing each fibonacci number into memory before moving on to the next one. Figure 2 shows the instruction code used to write the fibonacci program.

FIGURE 2

Results and Analysis

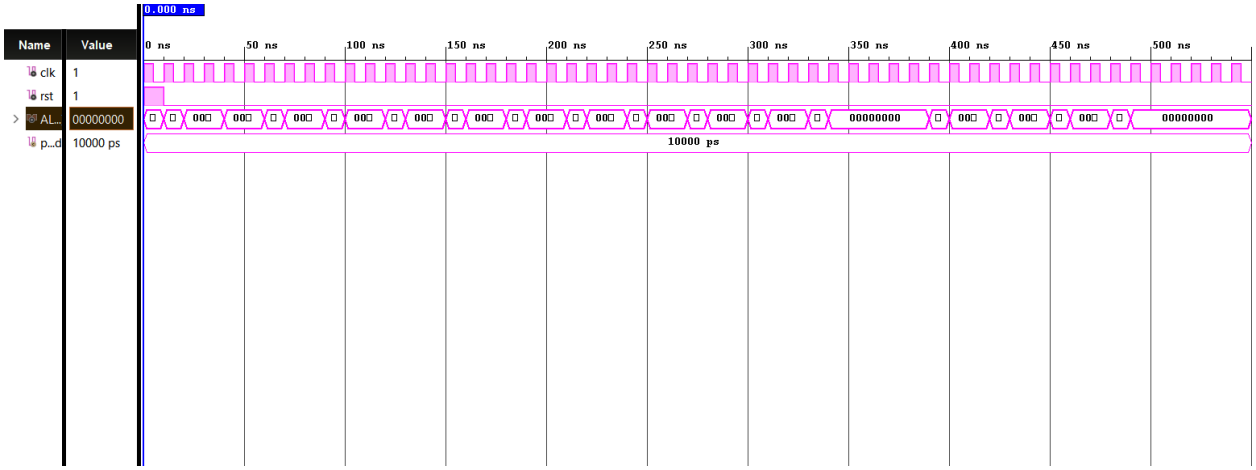

Figure 3: Part A Behavioural Simulation

For part A, multiple commands were tested for the MIPS instruction set.

- The first instruction was Addi Instruction to put in the value 0x10 into $t1 (x0010)
- The second instruction was Addi Instruction to put in the value 0x20 into $t0 (x0020)
- The third instruction was $t1 and $t0 being added and stored to $t2 (x0030)
- The fourth instruction was $t0 and $t1 being subtracted and stored to $t4 (x0010)
- The fifth instruction was $t1 and $t0 being multiplied and stored to $t2 (0x200)
- The sixth instruction was Addi Instruction to put in the value 0x04 into $t0 (x0004)
- The seventh instruction was a left logical shift where $t1 was shifted by the amount in $t0 and then the result was stored in $t2 (0x0100)
- The eighth instruction was a right arithmetic shift where $t1 was shifted by the amount in $t0 and then the result was stored in $t2 (0x0001)

Several Logical operations were also performed and then the values of those results were checked in the simulation to see if they were accurate. During these operations, the value of $t2 was stored at memory space 0xC. The memory was also checked to see if the store word functionality worked as expected. All the instructions performed obtained the correct results.

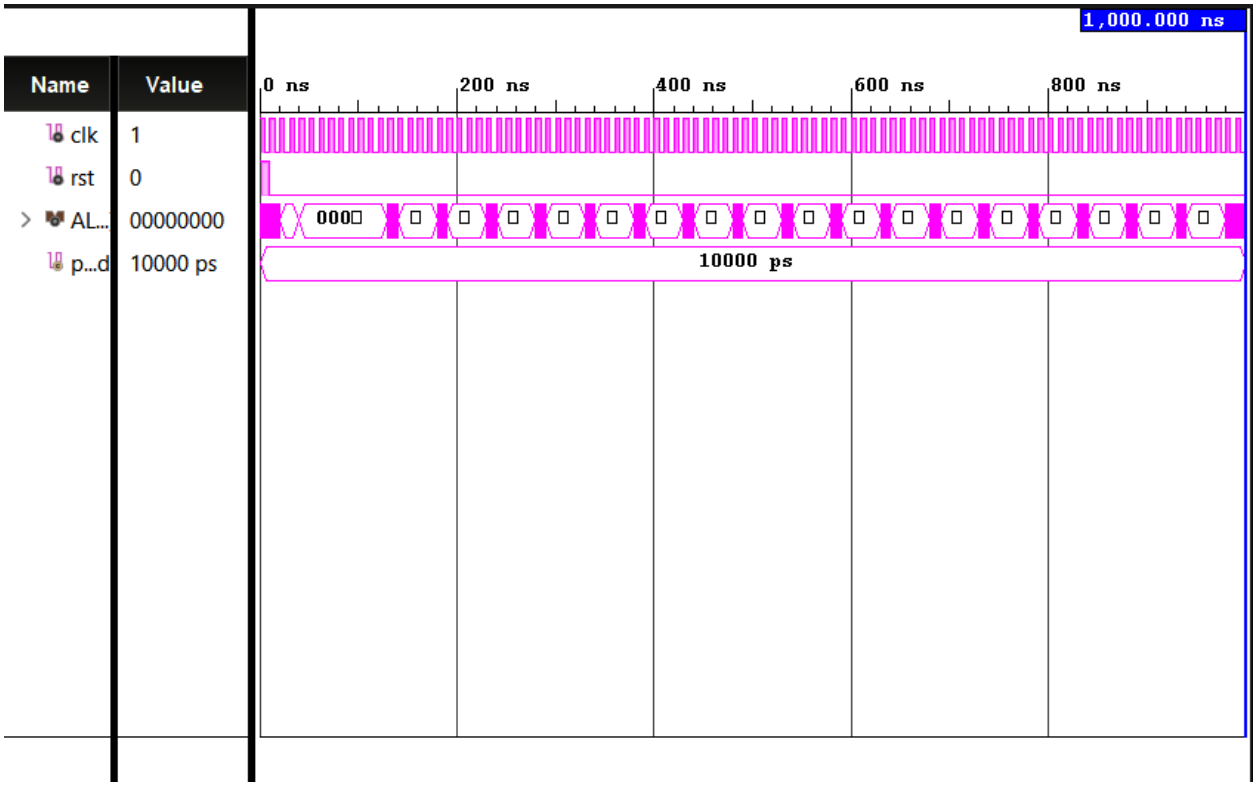

Figure 3: Part B Behavioural Simulation

The final number that was obtained in the simulation was 0x37 which is 55 in decimal and is the 10th unique number in the fibonacci sequence by order. All the fibonacci numbers were also stored in the memory after they were calculated using the store word functionality.

Conclusion

The project was successful because all the functionalities of the ALU that was in the MIPS worked as expected. The MIPS architecture was able to read instructions, decode them, execute and store them according to the specifications of the instructions. The fibonacci sequence program was also successful because correct values were obtained and stored from the fibonacci sequence. A lot of VHDL and digital design concepts were understood during the lab exercises and projects. Moreover, concepts on how to design a project with multiple complex components was also learnt from this project and the previous exercises.