CSCE 230 Project

Phase IV

Group #9

Johnathan Carlson, Tyler Zinsmaster, Jake Ediger

This project details the creation and implementation of phase IV. Phase IV involves adding I/O the current processor design.

Current Abilities of Processor

The current processor has acquired almost all of the abilities needed to be completely functional. Each phase has added abilities/functionality, making it more useful. All parts of the processor have been implemented in VHDL, which stands for "Very High Speed Integrated Circuit Hardware Description Language." The abilities detailed below also contain a description of how they were implemented in VHDL.

Instruction Types

- **R-type –** R-type instructions were the first instructions to be implemented on the processor. R-type instructions include Add, Sub, And, Or, *and* Xor.
- **D-type –** D-type instructions deal with data transfer between the memory and CPU. These instructions include load word, store word, add immediate, and sub immediate. Add immediate and sub immediate differ from the regular add and sub commands because they allow adding or subbing a numerical value, rather than the value of another register.
- **J-type-** J-type instructions involve jumping commands. Jump, jump and link, and load immediate are all J-type instructions.
- **B-type-** B-type instructions deal with branching, the main way loops and functions can be implemented in assembly. Instructions include Branch and Branch and link.

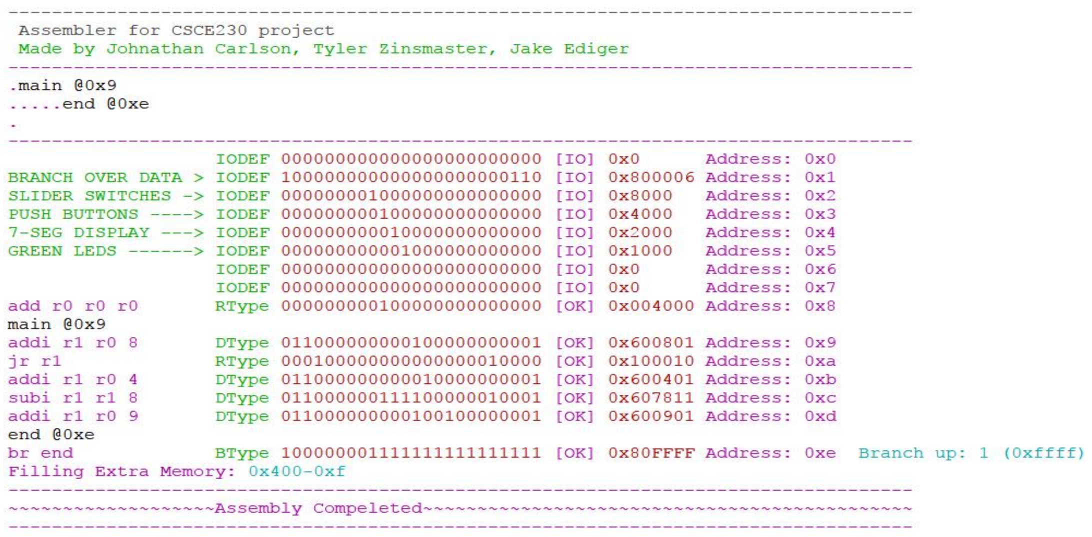
I/O

Green LED's, push buttons, and slider switches were all implemented in phase IV. The VHDL code determines which action to be performed based on the 4 bit code that corresponds to each part of I/O. A lot of difficulty occurred when moving the VHDL from phase III to the FPGA, which was necessary for phase IV I/O. Prior to implementing the program on the FPGA, a modeling software was used to test correctness. In order to transfer the functionality from the modeling software to the FPGA, the data-path needed to be drawn out so each stage could be understood. After creation of the data-path diagram, full implementation could be completed.

Assembler

The assembler had already been partially completed and had limiting functionality prior to phase IV. During phase IV, the rest of the assembler was completed. The assembler was designed in python, and proved to be a valuable tool in testing as well as implementation.

The assembler used is known as a two-pass assembler, which was necessary to be sure everything was correct. It outputs a .mif file, as well as an organized debugging interface. The assembler can take in any test file, and doesn't need to be redesigned for each new test. It stores the first 8 memory addresses in an array for use with I/O.



```
-----------------------------------------------------------------------------------
 Assembler for CSCE230 project
 Made by Johnathan Carlson, Tyler Zinsmaster, Jake Ediger
-----------------------------------------------------------------------------------
.main @0x9
.....end @0xe
.
-----------------------------------------------------------------------------------
                   IODEF 000000000000000000000000 [IO] 0x0      Address: 0x0
BRANCH OVER DATA > IODEF 100000000000000000000110 [IO] 0x800006 Address: 0x1
SLIDER SWITCHES -> IODEF 000000001000000000000000 [IO] 0x8000   Address: 0x2
PUSH BUTTONS ----> IODEF 000000000100000000000000 [IO] 0x4000   Address: 0x3
7-SEG DISPLAY ---> IODEF 000000000010000000000000 [IO] 0x2000   Address: 0x4
GREEN LEDS ------> IODEF 000000000001000000000000 [IO] 0x1000   Address: 0x5
                   IODEF 000000000000000000000000 [IO] 0x0      Address: 0x6
                   IODEF 000000000000000000000000 [IO] 0x0      Address: 0x7
add r0 r0 r0       RType 000000000100000000000000 [OK] 0x004000 Address: 0x8
main @0x9
addi r1 r0 8       DType 011000000000100000000001 [OK] 0x600801 Address: 0x9
jr r1              RType 000100000000000000010000 [OK] 0x100010 Address: 0xa
addi r1 r0 4       DType 011000000000010000000001 [OK] 0x600401 Address: 0xb
subi r1 r1 8       DType 011000000111100000010001 [OK] 0x607811 Address: 0xc
addi r1 r0 9       DType 011000000000100100000001 [OK] 0x600901 Address: 0xd
end @0xe
br end             BType 100000001111111111111111 [OK] 0x80FFFF Address: 0xe  Branch up: 1 (0xffff)
Filling Extra Memory: 0x400-0xf
-----------------------------------------------------------------------------------
~~~~~~~~~~~~~~~~~~Assembly Compeleted~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
-----------------------------------------------------------------------------------
```

Figure 1: This image is a sample output of the assembler. The data is from one of the test files used in phase IV to test I/O with the FPGA, condensed to save room.

Speed Overview

The speed of a processor is based on clock speed. 50Hz is the current clock speed that allows everything to run correctly. Fast adders were implemented to allow for higher clock speeds should the need arise. Although the clock speeds are relatively slow, higher clock speeds could be easily achieved with a few modifications. This would heighten the risk of unforeseen errors however, so the clock speed was kept slower.