Summary

Part1

Stage 1

The input and output CPU module and the top level module myComputer were defined for later use.

Stage 2

The Synchroniser and Debounce module was created in AUXMOD.v.

Stage 3

This stage checks the correctness of input output assignment of CPU and myComputer by setting when reset=1, Dout=0, when reset=0, all pins of GPO equals to 1.

Stage 4

Created module Disp2cNum, DispHex, DispDec. Disp2cNum is used to display 3 bit decimal number on 4 7-seg displays(the MSB could only be a negative sign). The mechanism of this module is to cascade 4 DispDec instances together to decide what to show on each of the display. Each of the DispDec module checks whther the input is still greater than 1 or not, if it is, get the number by mod 10, if less than 1, pass enable=0 to the next module to shut off that bit of 7-seg display.

DispHex is just Displaying 2 bit of hexadecimal number of 2 7-seg display.

Stage 5

Step 1 utilise a counter to slow down the 50MHz clock, by only “go” (do calculations) when the counter is finished counting.

Step 2, 3 introduced machine codes that tell the CPU what to do.

Stage 6

Assigned SW8 to implement the turbo feature, which is just bypassing the counter and use the 50MHZ clock.

Stage 7

Used case statement in CPU to recognise command group MOV and command in this group.MOV command group contains command for moving value from one reg to another, shift one bit of a certain reg to the left and assign the LSB with 0, or to the right, assign MSB by 0. Functions are introduced in this stage (get\_number, get\_location).

Stage 8

Implemented the ACC command group to do calculations and check argrimatic overflow.

Stage 9

Implemented JMP command group to jump to a specific state under certain condition.

Stage 10

If one bit of the RFLAG register is HIGH, jump to a specific address. After this, turn off that bit of the RFLAG register. Function regarding to command group MOV and JMP is implmemted in this stage.

Stage 11

Assigned Flag registers to debug output and created set\_bit and clr\_bit function to set or clear a certain bit in the register.

Stage 12

Synchronised all inputs from switches and buttons. Added falling edge detection for push buttons to prevent multiple times assignment to its corresponding register.

**Fully functional computer:**

The computer (part1) is working properly as shown in the tests, for instance: stage 9 shows the computer could execute commands from command group MOV, ACC, JMP; stage 10 shows the execution of ATC command group; finally, in stage 12, the test was performed to demonstrate that input switches could be used to take inputs, sample button pb[3] could use to move datum from switches to reg[28], one of the CPU registers, and be displayed on the 7seg display(Output).

**Fully functional RPN calculator:**

A RPN calculator program is written and it could perform calculation task according to the project specification. As it will be demonstrated on the board.

**Understanding of the microprocessor architecture:**

Microprocessor architecture is learnt over the development of the CPU module. The CPU module is designed to be able to read command which written in a specific format. Moreover, according to the machine codes in the instruction, it could perform specific tasks like calculations or reading and write registers. This is shown in the code of CPU module and how it work could be find in Stage 7 ~ Stage 11 in the report.

**Understanding of timing issues, including synchronisers:**

Synchroniser explained in mechanism part of the stage 2 in report.

The speed of the program (computer) is limited by Fmax given by Quartus software. It is important that the speed of the clock do not go over Fmax, which is considered as the maximum speed of the design. This measure depends on the longest delay along any path, called the critical path, between two registers clocked by the same clock.

There are timing issue related questions answered in the report, please refer to Stage 7, Stage 8 and Stage 12.

**Understanding of how the machine code works:**

A program is a collection of machine code to drive the CPU to perform tasks. It was implemented as different stages in this project. By changing its states, the CPU is able to perform tasks(do calculations) as a RPN calculator. (Refer to Report part2)

**Successfully running your computer at 50 MHz or higher:**

By using Clock Control Block Megafunction in Quartus, the clock speed was increased and therefore the CPU clock was able to run above 50 MHz. However, it is not safe to do so because Quartus gave the timing issue warning saying that Fmax is about 91 MHz (in the slow mode).

**Exceptionally good Verilog coding style:**

* No magic number
* Detailed comments
* No mix in use of blocking assignments and non-blocking assignments