Project

Jessie Herrera

Andre Martin

May 6, 2022

CSE 140

Spring 2022

We partitioned tasks pretty evenly. Jessie Herrera worked on Fetch, Decode, and execute. Andre worked on Mem, Writeback, ControlUnit, and our main. We both agree it would be fair if we received the same grade on the project.

1st part:

Fetch(): We make a fetch function that takes the string “instruct” and initializes it towards the size and how many clock cycles there will be. From this the “decode” and the “instruct” will update the “pc” in order to show how the pc is modified and the total execution time.

Decode(): The function begins initializing a string called “str” which would initialize the field of the instruction types. By doing this we create an integer array that has 32 entries and is registered towards the “registerfile”. It would then jump the instruction address towards where it needs to be with the global variable “jump_target”. We would then be able to update the “next_pc” variable.

Execute(): By having all the register files to their certain discrepancies, the control unit will behave the computations from the ALU and in a sense decode the given information.

Mem(): The function begins with initializing an integer for a register file and an address. By using the “ControlUnit” signals it will use the given info and have them imputed towards its respected register file but by as well setting it towards the “Writeback” function for writing back towards the display.

Writeback(): The function starts off with “rg” and “result” being initialized as an integer. This is done in order for the computation results from the “ALU” from the .txt in order to update the destination register in the “registerfile” array; furthermore this will make the “total_clock_cycles" variable.

Control_Unit(): The function begins with a string called “op”. This string would make the control signals for the function. This will come to behave the register files of the program, from that the “ALU” will receive the inputs from the “ControlUnit” and execute them in the “Execute” function.

2nd Part:

JAL: The “JAL” will jump to the target instruction and update $ra register with the PC+4 value. This can be seen in the “ControlUnit”.

JR: The “JR” has read a $ra register value and jump to the address contained in the $ra. This can be seen in the “ControlUnit”.