**Simulator**

Initialization

In the simulator we have read the memin and diskin files and put the data into arrays, output and output\_d, for the memory and disk respectably. We have also created an arrays for the regular registers and the hardware registers. Thus, we were able to perform the different operations using those arrays, without the need to read or write from files.

Unlike the memin and diskin files which have a defined maximum size, the input file irq2in doesn’t have defined size. That’s why in this case we have decided to read it one line at a time and wait to the next irq2 time in the file before reading the next line.

The main loop

After the initialization of the output arrays and the reading of the first line of irq2in, the main scripts enters the main loop. This loop runs until it reaches the end of mem\_in or it receives the halt operation (opcode 19).

In the main loop, first we check if the we have reached the time of the irq2 exception, if it is indeed the case the relevant register is updated and the time of the next irq2 exception is saved.

After that we handle any irqs, and run the timer script.

Then we decode the operation, print to the trace and invoke the main execution function Opcode\_Operation. In this function we execute the operation based on the op code – most operations are simple and require no further explanation.

Notable operations – the sw operation saves the word in the output array – which in the end of the script will be put into memout.txt

The printing to the hwregtrace occurs while executing operation in or out.

The out operation, which writes to the HW registers also writes to leds.txt, display.txt and handles the reading and writing to the hard disk.

Printing to output files

After the main loops finishes (the program has finished running), we are printing to diskout, memout, cycles and regout using the Print\_To\_Files function.

Printing to cycles and regout is simple and straight forward, printing to memout and diskout is done using the output and output\_d arrays.