Jesse Chen, Ojan Thonrycroft

Assignment 4

Aside from the opportunities for extra credit, this project did not have very many different design options to choose from. In the interest of time, we chose not to implement any of the bonus features. As instructed, we simply built the required new I/O features on top of our existing pipelined processor from Assignment 3.

For the LEDR and HEX displays, the only added functionality is the ability to read the output from their respective memory addresses, in addition to simply being able to write and display their output. This was an easy task as it only required a couple extra lines to read from the LEDR and HEX addresses in the memory stage. However, the keys and switches were more work to implement. We were previously already able to read from these input devices, but the added control registers required some additional logic to ensure the ready and overrun bits were correctly handled. Based on the specifications from the lecture slides, the implementation was reasonably straightforward. Debouncing the switch data was handled by counting 10 milliseconds in clock cycles before deciding if a change in the switches' state was ready to be used. Finally, the timer was created similarly to the switch debouncing by counting milliseconds in clock cycles. We chose to run our clock at 100MHz in order to complete fmedian2 in under 45 seconds. As a result, our timer would be incremented every 100000 clock cycles. Reading and writing to both TCNT and TLIM were also implemented almost identically to how the LEDR and HEX displays were handled, and the timer's control register was also the same as that of the keys and switches.

Replicating Assignment 1's xmax light application was somewhat of a challenge as it also uncovered a few unexpected bugs both in our processor and assembler. After setting up all the registers we needed with initial values, the application runs in an infinite loop, making alternating calls to save a new value to LEDR and to a timing subroutine that uses the timer to loop until a certain amount of time has passed before returning to the main loop. During the timing subroutine, the application also checks for key presses to increase or decrease the blinking speed by ¼ seconds up to a set maximum or minimum speed. The first problem we encountered was a simple fix in that we just forgot to allow writes to reset the value of TCNT before the timing loop would begin. Second, while trying to decrement the lights' blinking speed, we discovered that our assembler was not able to translate the subi instruction when used with a symbol. Finally, in order to limit the blinking speed to 2 seconds and display 8 on HEX3, we used a beq instruction to check if the HEX display read 0x8000. Unfortunately, the immediate value in our ISA is only 16 bits, and 0x8000 meant that the sign bit was inadvertently set, altering the intended result of the comparison. To fix this, all we had to do was change the beq instruction to bgt and compare it to 0x7000 instead of 0x8000.

My contribution to this project was primarily in implementing and debugging the xmax application in assembly. I also discovered and fixed a few other bugs outside of the xmax application as well. Unfortunately, I was unable to help my partner with much else due to projects in other classes. I believe my contribution to this project was about 30%.