Real-Time Clock Lab

ECEN 427

RealTimeClock.c

18 September 2015

Hampton and Ty Madsen

**Time spent:** 9 hours

In this lab we used a Fixed Interval Timer (FIT) and Push Buttons (PB) to create a real-time clock. It was somewhat frustrating and confusing primarily due to issues we had with the hardware and software not cooperating. Nonetheless, we persevered and completed the lab.

In order to complete this lab, we had to take into consideration the specification that we could not poll the buttons. We had to save the value of the buttons each time any PB interrupt occurred. We were able to debounce the buttons, by having a using the FIT to increment a timer each millisecond, and waited 7ms after the last PB interrupt to declare the button debounced. At which point we could change the clock if buttons are pushed. When no buttons were pushed, we used the FIT to increment another clock that reset every second, allowing us to make the clock count up every second. We used the FIT to increment 2 other counters, as mentioned below in the Interrupt Handler function section, that were useful in completing this lab.

We used several global variables and kept all of our functions in one file, so we didn’t have to pass variables between functions, and in an effort to keep the program as efficient as possible.

**Enabling Interrupts:**

In order to enable interrupts we had to do the following:

* On the GPIO we had to enable the push button IP interrupts both globally and on the push button peripheral. This allows the GPIO to fire interrupts at the Interrupt Controller (Intc).
* We had to enable interrupts on the IntC to register both the fit timer and push buttons interrupts (XIntc\_EnableIntr()), which uses a combination of the masks to let the IntC know which bits to set for enabling interrupts.
* We enabled the master interrupt from the IntC (XIntc\_MasterEnable()), so the IntC will actually fire interrupts to the microblaze.
* Finally, we enabled interrupts on the microblaze (microblaze\_enable\_interrupts()).

**Interrupt handler function:**

The interrupt handler determines which IP fired the interrupt, then calls the appropriate handler, either for the FIT or PB.

In the FIT interrupt handler, we increment 4 clocks which are used to count seconds, delay for updating the screen, counting half seconds for auto incrementing, and for debouncing buttons.

The PB interrupt handler first disables PB interrupts, so we don’t get interrupted with other button interrupts. It then gets the button state and saves it. It then resets the counter used for debouncing. Finally it acknowledges the interrupts and re-enables the interrupts from the PB.