Anthony Guzman
Blake Russell
ECE 153A
Prof. Forrest Brewer
October 20, 2021

Lab 1b - Stopwatch

Purpose and expected goals of the lab

Set up GPIO and timer as interrupts with FPGA board, using the Seven Segment Display and Push Buttons. Learn how interrupts can be triggered and controlled to accomplish tasks.

Methodology

Using a grand loop design, we consistently updated the seven segment display on the FPGA board, fast enough so that the numbers did not appear flickering. Once the timer reaches the reset value of F4240 (10ms), an interrupt is triggered, causing the timer to add 10ms to the timer and update the timer display. The display for the seven segment display was saved in an array, which was modified frequently to get the desired result. When a button is pressed, a corresponding interrupt is triggered which results in; holding timer value and stopping the count procedure, setting the direction in which the timer counts up, setting the direction in which the timer counts down, starting or continuing the timer count with respect to the direction, or reset ting the value of the timer completely.

Results

1. How fast of a stop watch are you able to build using your hardware and software configuration? What was the limiting factor in the accuracy of your stopwatch?
   a. The fastest stopwatch that we could build was about 1/100,000 of a second. However, this is not practical due to the rate at which our human eyes can register. The limiting factors were the overhead (if ran fast enough) when updating the seven segment display, and the clock speed of 100MHz (Nexys4_Handout_2021.xdc file).
2. How did you control the display update timing (i.e. loop period) and did this affect accurate timing measurement? If so, how? What display update period did you choose?
   a. We chose to display numbers greater than 0.01 of a second, approximately This value was chosen based on the fact that human eyes can only see between 30-60fps. If a human eye were to be able to see beyond 100fps, we'd have to carry it out to the next decimal place.

We noticed that with greater accuracy (more decimal places), the numbers were obscured from very rapid changes, appearing to be flickering.

3. The push buttons in the stop-watch are implemented using interrupts. Would it be possible to instead check the values of your push buttons while executing the grand loop? Would this polling approach change the timing of your stopwatch?
   a. It is possible to check for push buttons while executing the grand loop, however could cause some unwanted delays in our stopwatch. It caused these delays because previously, the system only had to fire the interrupts and update the 7 segment display, but additionally now have to check for button presses, potentially causing overhead. To potentially have it not affect the timer, we can check the button presses every certain amount of cycles, however when the number of cycles are significant, an issue can arise where the button press may not be registered if the button was pressed during the time where it wasn't checking for button press.
4. What will happen if two push buttons are pressed at once?
   a. When two buttons are pressed, only one of them gets registered. This was probably due to the fact that one button was pressed down first by a very small fraction of a second faster, thus getting registered first. In a very rare case, neither of them would register.
5. Describe the UI of the stopwatch. List the input and output devices.
   a. The inputs were five buttons: Reset (Center), Start (Left), Stop (Right), Count up (Top), and Count down button (Bottom). The output was the eight, seven segment display, to display the time.
6. Describe anything unique that you did to make your stopwatch work better.
   a. Neglecting the digits in the thousandth and ten thousandth place have made the stopwatch easier on the eyes (by not seeing all the flickering), at a low cost of precision.

Bounce and Glitches

   a. The glitch occurred randomly only on depression, sometimes toggling to the opposite direction, or maintaining the same direction before the press. This is because while holding down the button a little longer, it will continuously fire the interrupt, switching back and forth, and only register the very last toggle. This is not an issue without the modification because if the button was held down, it will repeat the same instructions.