**Lab #3 Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_**

|  |  |  |
| --- | --- | --- |
| **Item** | **Grade** | **Points** |
| **Req** | **40** |  |
| **B Func** | **10** |  |
| **A Func** | **10** |  |
| **Github** | **5** |  |
| **Code Style** | **10** |  |
| **Readme** | **25** |  |
| **Total** | **100** |  |

**Required Functionality**

In order to make required functionality you will need to properly trigger the oscilloscope on channel 1 using a positive edge trigger. Control of this process is to be performed using the microBlaze. The main tasks of the microBlaze will include:

* Moving audio samples into a pair of circular buffer. These circular buffers will be maintained in the address space of the microBlaze. That is you should have two big arrays defined in your program. Use polling of the ready bit of the flag register.
* Examining the samples looking for a trigger event.
* Fill the remaining sample slots in memory.
* Move the appropriate buffer values into the display memory of the oscilloscope (lab2) component.
* Provide a user menu (through the terminal) allowing the user to adjust the trigger voltage and trigger time

**B-level functionality**

Achieve required functionality. Use the ready bit of the flag register to trigger an interrupt. The ISR should store the samples (left and right), look for a triggering even, and signal when the stored samples should be transferred to the BRAM in the oscilloscope component.

**A-level functionality**

Achieve B-level functionality.

* Ability to enable and disable channels to display
* Ability to trigger off channel 2
* Ability to change the slope direction for the trigger.

**Software**

* All the memory mapped hardware registers will have their names setup as #define's with a name ending in "Reg".
* Any register with bit fields will have the bit index setup as #define's with a name ending in "Bit".
* The flagQ and flagClear registers need to be at the same address.