Lab 6
Musical Keyboard

Objective
In this experiment, students will be exposed to:
- creating ISR’s to handle interrupts on the TS7250 board
- using and combining most of what they have learned in previous assignments

Prelab
For this prelab you should investigate and include information about the following RTAI functions related to interrupts:
- rt_request_irq
- rt_enable_irq
- rt_disable_irq
- rt_release_irq

You should also list the ep9301 registers that need to be modified to control interrupts on port B. One of those is: GPIOBIntEn, the General Purpose I/O Interrupt Enable register for port B.

Lab Procedure

Part 1: The Keyboard
For this part of the lab you are to create a module that contains code to set up an ISR and a real time task. The purpose of the real time task is to create a square wave that will be played on a speaker. To create the square wave, you will need to toggle pin 1 of port F.

NOTE: The other pins in port F are used by other processes on the board, so when writing values to pin 1, be sure to not change the values of the other pins.

The purpose of the ISR is to handle one of five events, which correspond to the five lower pins of port B (those connected to the push buttons on the auxiliary board). These lower pins of port B should be programmed to cause an interrupt that, when triggered, changes the frequency of the square wave being played by the real time task depending on which pin triggered the interrupt. We want to program these interrupts to be falling edge sensitive, so program the registers accordingly.

Part 2: Master – Slave, Software Interrupt
For this part of the lab you are to use your code from the previous lab to decide a master slave relationship with the other students’ boards in the lab. This time, however, the master board
sends the current note that it is playing to all of the other boards, so that all of them play the same note as the master board. This requires a few steps:

1. You must program your master/slave server program to accept messages that begin with @. These messages represent one of the five notes to be played: @A, @B, @C, @D, @E.

2. If your board is a slave and it receives one of those messages, it must change the frequency of the note being played. To do this, you will use software interrupts. Specifically, you will use software interrupt 63 (reference the ep93xx manual), so you need to write a handler for this interrupt in your module. To trigger the interrupt in your server program, you simply write a 1 to the bit in the software interrupt register that corresponds to interrupt 63. Your module should still change the notes when the buttons are pressed. In other words, your module should handle both interrupts.

3. If your board is a master and it receives a note message, it must also change the frequency of the note being played. Furthermore, it should “forward” that message to all the slave boards in the network.

**Hint:**

A test program is provided in the `/opt/ts7250/students` directory on the nfs1 server. The name of the binary is `Lab6_client` and it is a client program similar to the one used for lab 5. In addition to the functionality of `Lab5_client`, this new client can also be used to send messages to specific IP’s, not only broadcast messages. To do that, type: & # where # is the last number of the IP address that you want to send to. For example: & 12 After you enter that, you'll be asked to enter the message that you want to send. In the previous example, whatever you type after you enter & 12, will be sent to IP 10.3.52.12 (including the 'return' character).

As before, anything else you type (except if it starts with & or !) will be broadcast. And if you type ‘!’ (without quotes) you exit the client program.

You should use this new client program to test your Lab6 server. It will be used on the post-lab due date to verify your programs.

**Extra part (optional, for extra credit):**

Modify your programs so that, if your board is the master, and a button is pressed, it sends the corresponding note to all the slave boards in the network.