EECS 388 Laboratory Exercise #2 SpeakerBuzz

1 Introduction

This laboratory implements a new task to generate a tone using the speaker on the BOOSTXL-AUDIO BoosterPack and to change the pitch with by using buttons on the TIVA LaunchPad. The first week's task is to generate a tone and the second week will be to get user input to change the pitch.

To begin this lab, make a copy of the Program_Blinky project in your workspace.

1. Select the Program_Blinky project and right-click.
2. Select "Copy"
3. In the workspace pane, right-click and select "Paste"
4. When asked for a project name, use the project name "Program_SpeakerBuzz"
5. Within the new project, create a task called "Task_PWM.c"

1.2 Tone Generation Task

Create and call an instance of this task in the main function. Go back to your Task_PWM.c file and enabling the following pins and ports.

1.2.1 Set-up AMP

You must initialize GPION and write a high value to Pin2 to enable the AMP to drive the speaker.

```
//Enable GPIO Port N.
//  SysCtlPeripheralEnable(SYSCTL_PERIPH_GPION);
// Configure GPIO_N to drive the Speaker.
GPIOPinTypeGPIOOutput(GPIO_PORTN_BASE, GPIO_PIN_2);
//set PortN Pin2 as an output
GPIOPinTypeOutput();
//write a high value here
```

1.2.2 Creating an infinite while loop and defining frequency

1. Define a uint_t named halfFrequency.
2. Create a while loop with a delay using vTaskDelay(). VTaskDelay to use parameters configTICK_RATE_HZ and frequency. The VTaskDelay parameter is evaluated as an integer and computed as tics. The amount of tics per second is determined by the configTick_Rate_HZ.
3. Use VTaskDelay, set the period to 0.004 seconds.

1.2.3 Set-up Timers

```
//Enable GPIO Port N.
SysCtlPeripheralEnable(SYSCTL_PERIPH_GPION);
// Configure GPIO_N to drive the Speaker.
GPIOPinTypeGPIOOutput(GPIO_PORTN_BASE, GPIO_PIN_2);
//Turn on the amp. At this point you should hear an audible click coming from the speaker on startup.
GPIOPinWrite( GPIO_PORTN_BASE, GPIO_PIN_2,1);
//Sets up pin to use alternative function. The General Purpose timer. Timer B Base 3 has 3 different modes: Capture, Compare, and PWM
GPIOPinConfigure(GPIO_PM3_T3CCP1);
GPIOPinTypeTimer(GPIO_PORTM_BASE, GPIO_PIN_3); //Sets as a timer pin.
//Specifies mode as PWM.
TimerConfigure(TIMER3_BASE,
TIMER_CFG_SPLIT_PAIR|TIMER_CFG_B_PWM);
//Initialized Pulse Width value
TimerMatchSet(TIMER3_BASE, TIMER_B, Pulse_Width);
//Set period
TimerLoadSet(TIMER3_BASE, TIMER_B, period);
//enables timer
TimerEnable(TIMER3_BASE, TIMER_B);
```

1.2.4 Generating a tone

1. Inside your while loop alternate the pulse width from a high to low value every half period.

1.2.3 Laboratory Measurements

Your GTA will ask you to display your frequency using "Task_PWM" using UART.

1.3 Buttons

Configure buttons (labed as USR_SW1 and USR_SW2) on the Tiva board. Use these buttons to change the PWM frequency from 1.2. USR_SW1 will increase the frequency and USR_SW2 will decrease the frequency. Your GTA will assign you values to modify the frequency.

1.3.1 Set-up buttons

To set-up SW_0 and SW_1 you must enable GPIOJ Pin0 and Pin1

```
//Enable GPIO Port N.
// SysCtlPeripheralEnable(SYSCTL_PERIPH_GPION);
// Configure GPIO_N to drive the Speaker.
GPIOPinTypeGPIOOutput(GPIO_PORTN_BASE, GPIO_PIN_2);
```

1.3.2 Change the period only during button down of SW_0 and SW_1

Use flags to determine if the button has been pressed or if it is released.

1.3.3 Laboratory Measurements

Your GTA will ask you to demonstrate the functionality of the buttons of your "Task_PWM" using UART.