Spring 2022 ECEN-5813:

Principles of Embedded Software

Final Project Report

Accelerometer-based musical note controller

by

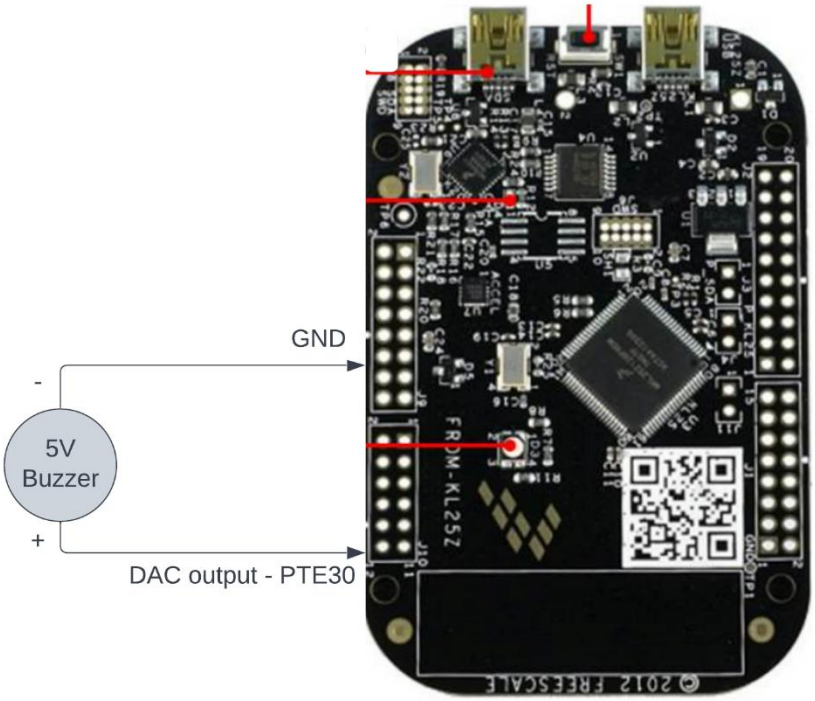
Pranav .M. Bharadwaj

Advisor: Howdy Pierce

Project Description:

This project builds a musical note generator where the frequency of the note being played is mapped to the roll-angle of the KL25Z development board in X-axis. This angle is measured using the onboard MMA8451Q 3-axis accelerometer. The peripherals in use are as follows: I2C0, DAC0, TPM1, TPM0, ADC0 and DMA0.

Block Diagram:



A Piezoelectric 5V buzzer is used to listen to the musical notes being produced. The steps to use this product is given below:

1. Connect a speaker/buzzer to the board in this manner: positive lead to the DAC output pin (PTE30), and negative pin to any of the GND pins.
2. Power on the board. Connect a terminal to it at the following settings: baud rate: 115200. Data bits: 8-bits, Stop-bits: 2-bits. (Alternatively, use the standard terminal provided by MCUXpresso)
3. The system shall complete the initialization procedures for the peripherals in use and begin testing. The sine function generator is tested first, followed by the ADC calibration routine. The list of available musical notes is played in succession next, with a delay of 500 msec between each note and each note being played twice. The accelerometer calibration routine is next, prompting the user to place the board in the following positions: flat 0-degree on a known flat surface, followed by flat against a vertical 90-degree surface. The error for margin is 4 degrees.
4. The user is next prompted to enter the passphrase to begin musical note generation, which is “Start” (case sensitive). The note generation begins after the correct phrase is entered.

5. The system measures angles from 0-180 degrees, with angle in the negative plane also presented as the absolute positive value.

Program files and associated functionality:

1. i2c.c & i2c.h - contains all code to initialize I2C0 peripheral at a baud rate of 100k. Further, it also contains code to allow I2C read and write by accepting device address, register address and input data as the parameters.

2. mma8451q.c & mma8451q.h - contains all code to initialize MMA8451Q 3-axis accelerometer in its active mode and use 14-bit mode at an output data rate (ODR) of 800 Hz. Further, it contains code that reads the acceleration data from the sensor and converts it into usable roll-angle in degrees with a range of [0-180].

3. DAC.c & DAC.h - contains all code to initialize DAC0 peripheral to generate the required musical notes. The DAC sampling frequency is 48 KHz and is driven by DMA interrupt. It also contains code to initialize the DMA peripheral to transfer data from sine lookup table present in flash to the DAC buffer.

4. ADC.c & ADC.h - contains all code to initialize the ADC0 peripheral with a sampling frequency of 96 KHz in 16-bit mode. It also contains code to initialize TPM1, which is used by the ADC code to time sampling windows and ADC calibration routine.

5. sine_generator.c & sine_generator.h - contains code to generate required sine values using integer-math. The lookup table is precomputed and stored in memory. This also handles linear interpolation for values that do not match the exact lookup table values.

6. sample_generator.c & sample_generator.h - contains code to generate musical notes at required frequency. Fills up the buffer accessed by the DMA peripheral with the correct samples.

7. sine_test.c & sine_test.h - contains code to automatically test the accuracy of the integer-math based sine function. Iterates over all values between -Pi to Pi. Returns maximum error and sum of squares error.

8. delay.c & delay.h - contains code to generate delays using hard-spin loops. Input is provided in number of milliseconds of delay required.

9. PES_Assignment_7.c - main file that calls all initialization functions and runs the main while loop. Reused PES Assignment-7 MCUXpresso project for this.

Section 2: Updated Testing Plan

The final test plan is described below. These tests are performed at the powerup, calibrating the peripherals in use.

- The DAC outputs can be tested by feeding the output to the onboard ADC, converting it back to digital, and performing autocorrelation between the ADC's output and the DAC's input. Mean sum of squares can be calculated to further investigate the differences. This can be performed by an automated test.
- The accelerometer provides us with offset registers for each axis, where offset values can be loaded to calibrate. A calibration routine described in the reference manual was employed, prompting the user to hold the board in different orientations against known surfaces to calculate the necessary offset.

- The list of available musical notes was played in succession, with their output fed into the ADC to compare the difference between reproduced sound and the frequency defined in code.

Section 3: Project outcomes and references

This section covers the expected project outcome and the possible challenges to face during its course. It also mentions the references used in creating this proposal and the project itself.

3.1 Project outcome and challenges

The project helped me in realizing a complete embedded system that fulfilled the required functionality, albeit with some challenges. During the course of this project, I vastly improved my knowledge in implementing a working I2C module from scratch, understanding the process of reading to and writing from registers onboard a I2C device. This was the biggest takeaway from the project, as serial communication is employed in most embedded projects, with I2C being dominant with commercial sensors.

Some of the challenges faced included the integration of a UART-based command processor from scratch for the following reasons:

1. Once the musical note generation has started, the system constantly measures acceleration values and generates musical notes, without the need for any user intervention via the command prompt except the roll-angle of the board.
2. Exiting to the command prompt using the board orientation would not be possible as this is already in use for frequency control. (We could use orientation in the other 2 axes, but I believe that this would make the product less user-friendly)

The buzzer employed serves a limited purpose for reproducing musical notes, as the quality is poor and the frequency range is limited, prompting me to use notes in the frequency range of 85 Hz to 3000 Hz. Beyond this range, the sound quality either drastically degraded or was not reproduced at all due to extremely low frequency.

3.2 Project outcome and challenges

- Alexander G Dean github supporting code and materials for textbook, https://github.com/alexander-g-dean/ESF
- List of musical notes in western music with their corresponding frequencies: https://pages.mtu.edu/~suits/notefreqs.html
- NXP MMA8451Q reference manual and calibration document: https://www.nxp.com/docs/en/data-sheet/MMA8451Q.pdf