Project 2

EEE 212 - Microprocessors

05.05.2023

Description of the Assignment

In this project, you will be building a signal generator using FRDM-KL25Z, the internal ADC and DAC of KL25Z, two push-buttons, and a power supply. Your implementation should be capable of the following:

- Generating sawtooth, square wave, and rectified sinusoidal (i.e., the negative parts have to be clipped to be zero) waveforms simultaneously upon the press of a specified button. The order of the generated waves should be: *(i)* square wave *(ii)* rectified sinusoidal, *(iii)* and sawtooth. For example, if the first generated wave is a square wave, pressing the specified button should switch the signal to a rectified sinusoidal wave, and so on.

- Reading the amplitude of the signal from the reference voltage supplied to the ADC every 2 seconds. Each amplitude value should be visible for at least 2 seconds before being updated. If the current amplitude changes, the new amplitude value should be reflected on the generated waveform.

- The frequency of the generated signal should be `last two digits of your ID number` $\times$ `10`. For example, if my ID number is `21601705` and 2.1 V has been read from ADC, the following should be printed on the LCD:

```
Current frequency:  50 Hz
Current amplitude:  2.1 V
```

  If the read amplitude changes, the value printed on the LCD must be updated, that is, the LCD should be refreshed every 2 seconds.

1. The ADC Part

- The push-buttons cannot function without interrupts. Therefore, attempting to fulfill this requirement without interrupts will not be successful. It is important to understand why interrupts are necessary, as this may be asked by your TAs and can affect your grade.

- The KL25Z has an onboard ADC that can be configured. If you are interested in exploring the full capabilities of the KL25Z and have a complete understanding of its functions, it is recommended that you refer to the datasheet of KL25Z.

- It is highly recommended that you configure the onboard ADC before implementing the signal generator to ensure that everything functions correctly.
- It is important to note that you can use an adjustable power supply to generate a reference voltage to be sampled. If you have access to a potentiometer, you can use a fixed power supply in combination with it to generate an adjustable reference voltage to be sampled. After flashing the example program to the memory, you should locate the appropriate pin to connect the reference voltage. If the reference voltage varies and the color of the LED changes accordingly, then you are ready to proceed.
- Since you will be reading the voltage value every 2 seconds, there is no need to read it every time. Instead, you can set a timer to update the amplitude by checking the ADC every 2 seconds. It is acceptable if there are occasional delays due to reading and processing.
- **VERY IMPORTANT:** Do not feed more than 3.3 Volts to the pins that will be used as an input to the ADC.

2. The DAC Part

- The onboard DAC FRDM-KL25Z should be used to generate the required waveforms. To do this, you should evaluate one period of all three functions and store them in separate arrays or a multidimensional array.
- Once again, it is crucial to verify that the onboard DAC is working correctly before proceeding with further development.
- The next step is to integrate the ADC and DAC components. You will read the reference voltage fed by an adjustable power source through ADC. The amplitude of the signal you will generate using DAC **MUST** be the same as what you have read. However, the amplitude of the generated signal may differ slightly due to the noise induced by the physical components used.
- **VERY IMPORTANT:** One important parameter to consider in the above procedure is the number of samples to be taken in one period. It is crucial to keep in mind that even though you can compile a program without errors, you are still subject to hardware limitations. If you try to store too many samples, you may encounter runtime errors such as *HardFault*. Therefore, it is best to keep everything as simple as possible.

3. The Push-Button

Defining an ISR (Interrupt Service Routine) to act when the button is pressed is a crucial step in the implementation process. This will allow the type of signal generated by the DAC to be switched between sinusoidal, triangular, and square waveforms as the button is pressed. To achieve this, it may be necessary to define a global variable to keep track of the current waveform type, but why? Your TAs may ask this as a part of your grade.