**Universidad Autónoma de Guadalajara**

Ingeniería Electrónica Biomédica

System designing with Microprocessors

*“*Practice 7: Digital to Analog Converter”

Jesús Arnoldo Zerecero Núñez

2885993

Andrea Alejandra Mondragón Olivos

2915351

Practice 7. Digital to Analog Converter

**Introduction**

The KL25z includes a DAC module in which the user may set a value which corresponds to a certain voltage range. Setting multiple values to the DAC module and cycling these transactions enables the capability of creating a frequency generator.

The KL25z’s DAC module permits the user to use one of two voltage references: 3.3v or 5v, being 0v the lower limit for both cases.

In this practice, a frequency generator was implemented using this DAC module. The following signals were calculated and programmed to be painted:

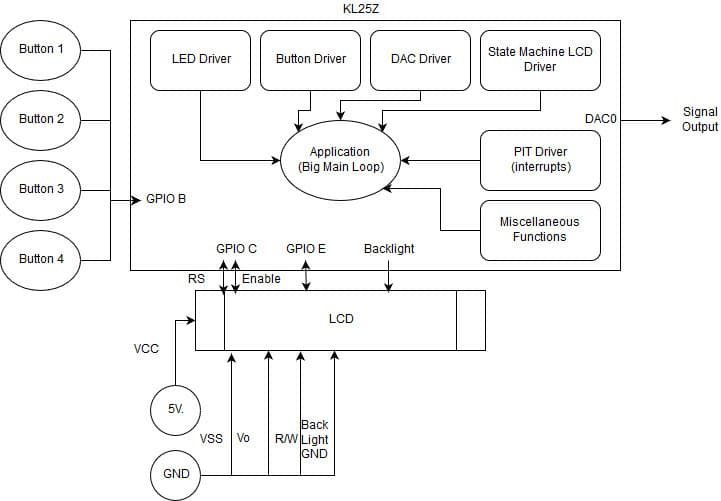
* Sine wave.
* Square wave.
* Triangle wave.
* Sawtooth wave.
* Capacitor’s charge and discharge.
* Rectified sine wave.
* QRS Complex.
* Exponential wave.

4 buttons were implemented for these practice. One for cycling through the different signals, one to turning the signals on or off and the last two to increment or decrement the signals’ frequency by 100Hz. The application may then generate signals with frequencies ranging from 100Hz to 2.5KHz.

In order to generate the signals at a constant frequency without depending on the efficiency of the rest of the code, PIT interrupts were used. Each time the PIT counter reaches overflow, a signal’s coordinate is painted. The frequencies were then changed by modifying the PIT’s maximum value (LDVAL).

Also, a state-machine-controlled LCD was also used, in which the signal’s name and frequency is shown.

**Development.**

**Block diagram/modules implemented:**

**Conclusion:**

Many uses can be given to this simple frequency generator, such as an ECG signal simulator, linear PWM modulation and fault finding activities in other devices. 2 signals may be generated at once using this same application as the DAC module includes two DACs (DAC0, DAC1) which could be each called and given a value to paint each PIT interruption.

The most difficult part of this practice was finding a PIT value which is low enough to reach the 2500Hz mark (400u, divided by the size of the signals’ point buffer) whilst giving space for good resolution signals (bigger buffer sizes) AND not taking up the whole bandwidth in the process, as the LCD, buttons and LED needed to be part of the big main loop. In the end, an interruption every 8u was enough to permit the detection of button inputs and nice signal resolutions (buffer size of 50).

Each signal’s buffer was calculated using different formulas, ranging from calculator usage and excel tables to MatLab graphing. These depended greatly on the buffer size.

**Link to video demonstration:** [**https://photos.app.goo.gl/GruwuIGN9bWVHR6Q2**](https://photos.app.goo.gl/GruwuIGN9bWVHR6Q2)