**Universidad Autónoma de Guadalajara**

Ingeniería Electrónica Biomédica

System designing with Microprocessors

*“*Practice 3: Ringtone Player”

Jesús Arnoldo Zerecero Núñez

2885993

Andrea Alejandra Mondragón Olivos

2915351

Practice 3. Ringtone Player

**Introduction**

A timer is a specialized type of clock for measuring time intervals. A hardware timer utilizes a quartz clock for high precision timing measurements, while a software timer relies on a digital clock generator. Timers are a very valuable resource for many applications, as using them can lead to easier information gathering.

In this practice, a Ringtone Player application was built using multiple software timers, based on the Nokia Ringtone Format. This format consists on a character string containing information for every tone:

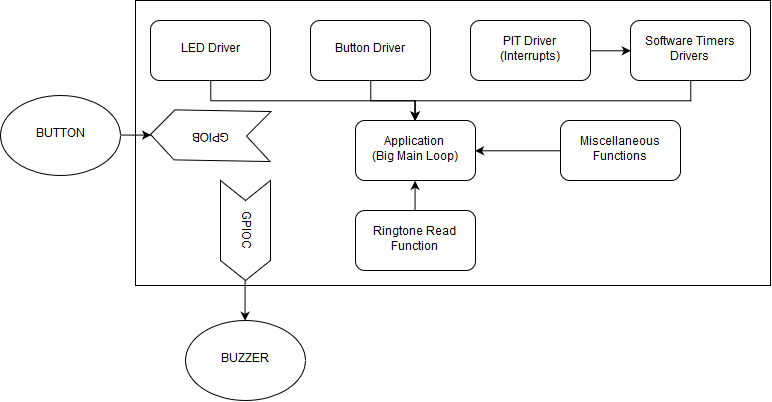
“4c3”

* **4** represents the duration of the tone (1/**4** of a second).
* **C** represents the note to be played.
* **3** represents the octave of the selected note.

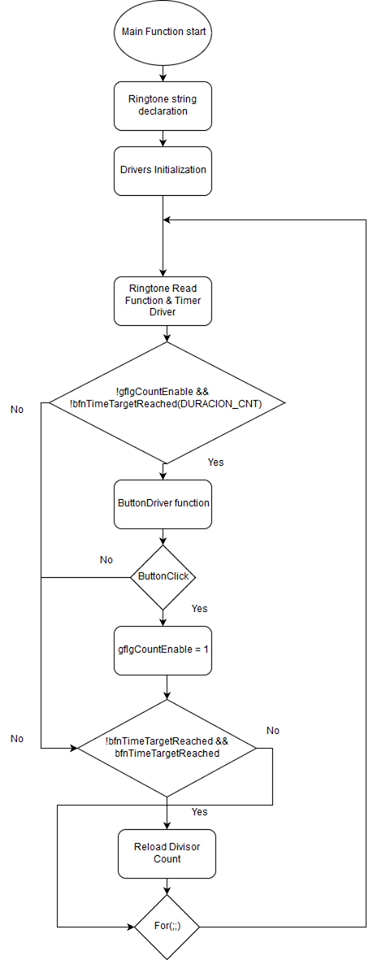
The ringtone was played sending pulses to an external buzzer. A button was implemented to play the ringtone whenever it finishes. A led blinks at the rhythm of the ringtone as well.

**Development.**

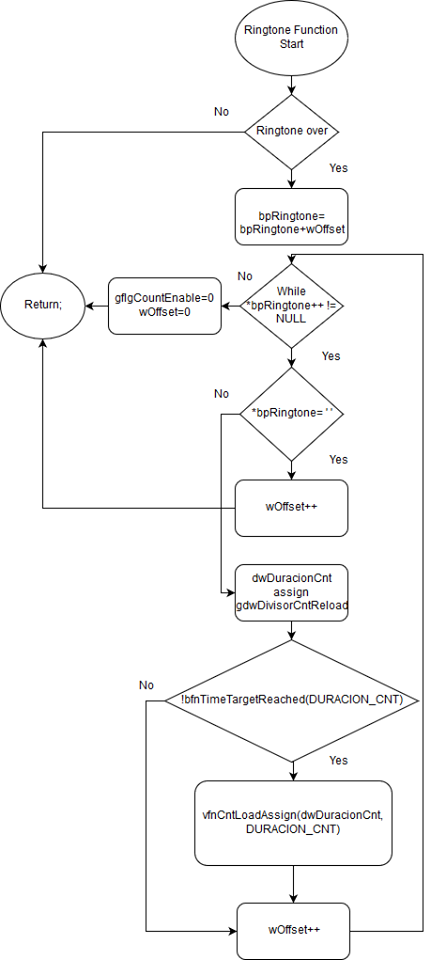
**Block diagram/modules implemented:**

****

**Main Function Flow chart:**



**Ringtone Function Flowchart:**

****

**Conclusion:**

**Arnoldo:**

With this practice I learned the importance of using software timers for any application. Personally, I considered this practice an easy task, yet, we had several problems and learned some valuable lessons for future codes: The base time (the PIT module time target) needed to be low enough to have high quality tones (helping differentiate the highest frequency tones from each other more easily) but high enough for the microprocessor to have enough time to cycle through the big main loop’s conditions and drivers before the next timer interruption arrived. If the big main loop’s execution wasn’t quick enough, the timer will be delayed and the ringtone will be severely altered. We struggled to find that “golden number” to use as our base time, which ended up being 30us, giving us good quality tones and exact timer counts without execution delays. Other smaller problems included planning the code’s structure and drivers’ format in order to make the code very time efficient.

Con esta práctica aprendí la importancia de utilizar timers de software para cualquier aplicación. Personalmente, consideré esta práctica como una relativamente sencilla; sin embargo, tuvimos diversos problemas los cuales nos servirán como lecciones en códigos futuros. El principal problema fue elegir una base de tiempo ideal para la cuenta del PIT. Esta base debe ser lo suficientemente baja como para diferenciar los tonos de mayor frecuencia y de este modo, tener una mejor calidad de sonido. A su vez, la base no debe ser muy baja o de otra manera el tiempo puede ser desfasado si los drivers y las condiciones tardan mucho en ejecutarse. Finalmente encontramos la base ideal de 30us para el PIT, el cual nos otorgaba buena calidad de sonido sin retrasos en el tiempo. Por esta razón, debimos cuidar mucho el formato de nuestro código y drivers, asegurándose de que fuese de ejecución rápida.

**Alejandra:**

Como conclusión para esta práctica se pueden resumir los resultados como la utilización básica de puertos de entrada y salida del micro, con un manejo más completo de los timers para el desarrollo de la aplicación requerida, en este caso el reproductor de ringtones enviados como cadenas y leídos por apuntadores entrando a diferentes casos según su lectura para utilizar los dos software timers que usamos con diferentes frecuencias según los casos, lo que nos llevó a uno de los problemas principales que fue elegir la base de tiempo para los timers y que se pudiera ajustar a nuestras frecuencias.

In conclusion for this practice, the results can be resumed as the basic usage of the inputs and outputs of the microcontroller, focusing on a better timer management for the development of the required application, in this case, the ringtone player. The ringtone was read as a string and read using pointers to the string, entering different conditions according to the character read in order to use the two software timers with different frequencies. This lead us to one of the biggest problems which was choosing the time base for the timers so it can adjust to every possible frequency.

**Demonstration video link:** [**https://photos.app.goo.gl/AScxTTao4WZS9muP2**](https://photos.app.goo.gl/AScxTTao4WZS9muP2)