Generador De Ondas Sobre Raspberry Pi Pico

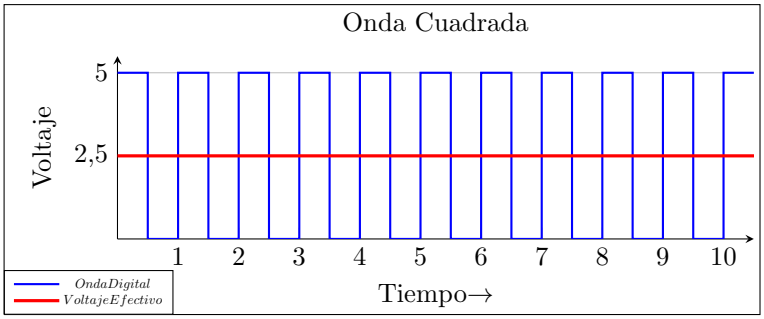
Santiago Calligari, Gerónimo Nestares

18 de junio de 2022

Preámbulo

En este artículo vamos a contar el proceso sobre la generación de ondas sinusoidales en una placa de desarrollo, vamos a exponer todos los obstáculos que nos topamos, las soluciones que les dimos y las ideas tanto matemáticas como electrónicas que utilizamos para poder hacer esta idea una realidad que pueda ser replicada.
Nuestra placa de desarrollo cuenta con algunos pines **G.P.I.O.** (General Purpouse Input/Output) que son utilizados para propósitos generales de entrada/salida. Nosotros vamos a utilizar uno de estos pines para llegar a nuestra onda sinusoidal, pero nos encontramos con un problema ¡No existen las ondas analógicas en el mundo digital! Nosotros contamos solamente con una función "prendido"que nos entrega $5V$ en nuestro pin y una función .ªpagado"que nos entrega $0V$. Aquí entra nuestro primer concepto a explicar, la modulación por ancho de pulsos.

Modulación Por Ancho de Pulsos



Esto es una onda cuadrada, el único tipo de ondas que puede generar un microcontrolador. Desde aquí tenemos que llegar a una onda que sea un poco más parecida al seno que buscamos. Bien, esta onda particular está en 5V la mitad del tiempo y el resto se encuentra apagada ¿por qué no le ponemos un nombre al tiempo que está en $5V$? ¡Ya sé! Lo vamos a llamar Duty Cycle (Ciclo de trabajo). Lo vamos a definir como D_c y que sea igual al porcentaje de tiempo

que está en $5V$, en el caso de nuestra gráfica sería un $D_c = 50\,\%$.

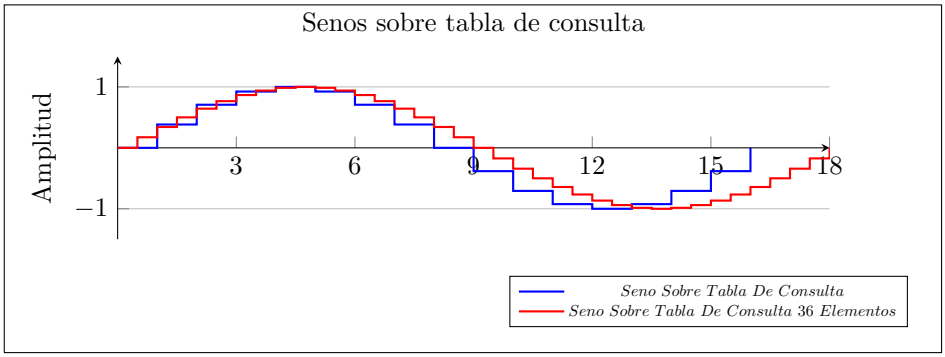
Vamos a hacer un pequeño paréntesis para hablar de procesadores, a muy groso modo estos son un tipo de calculadoras extremadamente básicas y exorbitantemente rápidas. Dentro de sí hay un pequeño cristal que en nuestro caso oscila a $133MHz$, que equivale a 133 Millones de oscilaciones por segundo. Cada ciclo de reloj nuestro procesador puede tanto sumar, comparar o cargar. En el caso que nosotros encendamos nuestro pin cuando el numero de oscilaciones es par y lo apaguemos cuando sea impar, estaríamos generando una onda cuadrada de $D_c = 50\,\%$ que al oscilar tan extremadamente rápido entre esas dos condiciones nos generaría, a modo práctico una onda plana de un $50\,\%$ del voltaje total, o sea $2{,}5V$ (Onda roja) llamada voltaje efectivo.
Con el D_c podemos generar un V_e desde $0V$ hasta los $5V$ ganamos la capacidad de generar ondas escalonadas, cuando modifiquemos el D_c modificamos el V_e. Ahora tenemos que aprender como modificar el D_c para poder generar algo parecido a un seno.

Tabla de consulta

El seno es una función periódica, con periodo T $= \frac{1}{F} = 2\pi$. Si calculamos $sen(2)$ es exactamente lo mismo que calcular $sen(2 + 2\pi)$, con esto en mente podríamos no precisar de calcular cada vez el seno de un numero para generar nuestra onda sino tener almacenado el resultado de un $f(x) = y$ como la tabla siguiente que el valor de x comienza en 0° y va escalando de a 22,5° (Este escalado es conocido como Δx) hasta llegar a los 337,5° para luego reiniciar el ciclo y volver a leer desde 0°.
Si llenamos una tabla con estos valores podemos conseguir una tabla de consulta, a la cual llegaríamos y preguntaríamos:
Señora tabla ¿Cual es el resultado de $sen(22{,}5°)$? y ella nos responderá extremadamente rápido ¡0,7071!

0.0000	0.3827	0.7071	0.9239
1.0000	0.9239	0.7071	0.3827
0.0000	-0.3827	-0.7071	-0.9239
-1.0000	-0.9239	-0.7071	-0.3827

Esta tabla de senos con $\Delta x = 22{,}5°$ genera la o



Desde la gráfica de estas dos funciones podemos deducir que mientras más elementos tenga nuestras tablas de consulta mayor va a ser la precisión del seno. Sabiendo el valor de los senos en un punto X podemos llegar a un valor de D_c para poder generar una onda cuasi analógica.

Librería PWM

Ya es momento de ensuciarse las manos, vamos a hablar de código! Gracias a esta librería podemos hacer realidad este proyecto de generar ondas sinusoidales. Con lo aprendido hasta aquí y con la lectura de "Raspberry Pico SDK"podemos empezar a codear, vamos a empezar a explicar nuestro código.

```
#include "pico/stdlib.h"
#include "hardware/pwm.h"
#include "tabla.h"       //Nuestras Dos Tablas de consulta.
#define SONIDO_OUT 0     //Pin de salida.
#define ON 1
#define OFF 0
#define RESOLUCION 512   //Cantidad De Valores de Nuestra
//Tabla de Consulta de Senos
#define A PWM_CHAN_A     //Abreviaturas
#define PWM GPIO_FUNC_PWM//Abreviaturas

float Hz(int frecuencia)
{return hertz[frecuencia];}

void setDuty(int valor, int canal)
{pwm_set_chan_level(canal, A, valor);}//Mandamos nuestra señal
//por el canal hasta el
//valor dado.
```

```
int main()
{
        int iterar = 0;//Creamos un canal PWM en el pin de Sonido
        int canal = pwm_gpio_to_slice_num(SONIDO_OUT);
        //Convertimos el pin de sonido de GPIO a PWM.
        gpio_set_function(SONIDO_OUT, PWM);
        pwm_set_enabled(canal, true);//Iniciamos PWM en el canal
        while(1) //Loop Infinito
        {
                //Ponemos la ventana igual a nuestra Resolucion.
                pwm_set_wrap(canal, RESOLUCION*2);
                setDuty(valores[iterar], canal);
                //dormimos el tiempo suficiente para nuestra F.
                sleep_us(Hz(X));
                (iterar == RESOLUCION-1) ? iterar = 0 : iterar++;
        }
}
```

Con estas sencillas lineas de código podemos generar nuestras ondas, allí podemos ver dos cosas interesantes, la función Hz y la X dentro de ella, esa pequeña X puede tomar el valor de cualquier numero entre 1 y 20000 para generar una onda sinusoidal de amplitud 1 y frecuencia X. Ahora ¿Cómo llegamos a la función Hz? Esta es una simple tabla de consulta, nuevamente.

¡Tablas de Consulta al ataque!

Como hemos observado las tablas de consulta son una forma liviana de conocer un resultado estático sin la necesidad de calcularlo. Por ese motivo llegamos a la conclusión que la mejor forma de manejar frecuencia de nuestra onda es con otra tabla de consulta.

Controlar frecuencia de la Onda

Con el fin de controlar la frecuencia de nuestra onda vamos a utilizar la librería "*hardware/pwm.h*".
Vamos a comentarle la idea matemática tras esto. Para jugar con la frecuencia de una onda contamos con 3 valores;

Wrap Es la cantidad de ciclos de reloj pasan hasta que el reloj vuelve a 0.

Step Cuantas celdas recorremos por paso.

Sleep Es la cantidad de tiempo que espera el procesador antes de ejecutar la próxima orden.

Vamos a comenzar a calcular:
Nuestra tabla de senos tiene 512 valores que van de 0 a 1024 por conveniencia y para limitar el uso de flotantes al menos posible. Tenemos un reloj PWM de $125MHz$ lo que nos muestra una cantidad de 125000000 ciclos por segundo, aproximadamente $8Nanosegundos$ por ciclo. Nuestro Wrap va a estar capado a 1024 ciclos por conveniencia, cada vez que cambiemos el D_c tenemos un tiempo de $Wrap * 8ns$ donde no es posible hacer nada. Eso nos da una $8196ns$ donde no pasa nada. La fórmula para la frecuencia de nuestra onda sinusoidal es:

$$f =$$