LABORATORIO 7

DEPARTAMENTO DE FÍSICA

FACULTAD DE CIENCIAS EXACTAS Y NATURALES

UNIVERSIDAD DE BUENOS AIRES

1er cuatrimestre 2021

Aplicación del efecto Josephson a la generación de señales arbitrarias con aplicaciones a la metrología

Alumno: Pinto Zárate, José Daniel
dann.2207@gmail.com

14 de enero de 2022

Resumen

El JAWS (Josephson Arbitrary Waveform Synthesizer) es un sintetizador de señales basado en el efecto Josephson. Cuando se conectan miles de junturas Josephson en serie, se puede aprovechar este efecto para generar señales arbitrarias. Para esto, se envían pulsos de corriente de alta frecuencia hacia esos arrays de junturas, sumergidas en Helio a 4.2K, y las mismas generan una tensión de salida que depende de constantes fundamentales, y por lo mismo tiene muy baja incertidumbre, del orden de los $1\mathrm{V}/10^{10}\mathrm{V}$. *Existe un patrón específico de pulsos para cada señal que se quiera generar, y el mismo se obtiene mediante un algoritmo llamado modulación Sigma-Delta. En este trabajo se realizaron y comprobaron numéricamente distintos patrones de pulsos, generados mediante un algoritmo escrito siguiendo bibliografía especializada [**?**] [**?**]*

1. Introducción

1.1. Efecto Josephson

Los sistemas conocidos como Josephson Arbitrary Waveform Synthetizer (JAWS) [**?**] [**?**] [**?**] son sistemas basados en el efecto Josephson utilizados para la generación de formas de onda arbitrarias de tensión, en el cual la tensión generada depende de constantes fundamentales.

El efecto Josephson es un fenómeno que ocurre en junturas superconductoras separadas por una barrera muy delgada de algún material no superconductor al ser irradiadas por una frecuencia de microondas. En el caso del JAWS las junturas usadas son del tipo SNS (superconductor-normal-superconductor).

Estos sistemas utilizan arreglos, o *arrays*, de miles de junturas Josephson, que son irradiadas por un tren de pulsos de corriente de tiempo corto, del orden de los GHz, con un patrón de repetición determinado por el tipo de señal que se quiere obtener. La tensión en una juntura está dada por:

$$V_J = n\frac{h}{2e}f = n\Phi_0 f \tag{1}$$

donde h es la constante de Plank, e la carga eléctrica elemental, Φ_0 es el cuanto de flujo magnético, n un número entero y f la frecuencia de trabajo, en nuestro caso referida a un reloj atómico. Los pulsos que usaremos son binarios, lo que significa que los valores de n posibles son $n = \pm 1$.

Cuando se conecta un número grande de junturas Josephson en serie, el efecto descrito en la ecuación 1 se suma por cada una de las junturas presentes, dando lugar a la posibilidad de generar voltajes ordenes de magnitud mayores a los de la juntura simple.

El chip mostrado en la figura 1, el que se va a utilizar en el futuro, contiene 2 arrays, uno de 5000 y otro de 4996 junturas conectadas en serie. Las junturas en nuestro caso son de tipo SNS (superconductor-normal-superconductor). En la figura 1 se muestra el chip JAWS, construido en el PTB (Physikalisch-Technische Bundesanstalt), que será usado para armar el sintetizador o sistema JAWS. [**?**] [**?**]

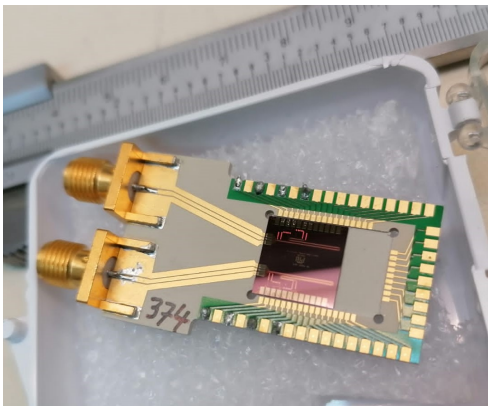

Figura 1: Chip JAWS construido en el PTB con 2 arrays de junturas Josephson disponible en el INTI

Un esquema general de la futura construcción del sistema JAWS se muestra en la figura 2. Lo que se hizo en este trabajo fue la generación numérica de los patrones de pulsos necesarios para cada señal que queramos generar.

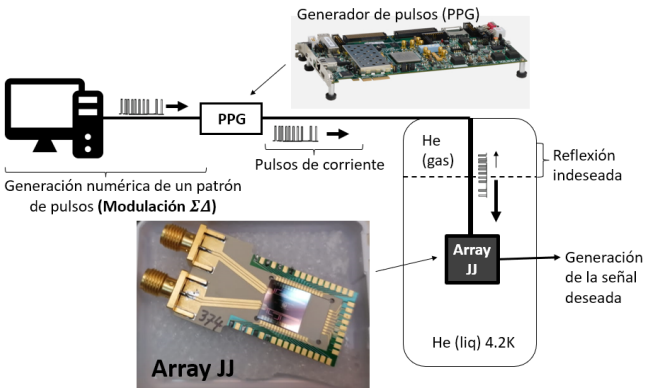

Figura 2: Esquema general de la construcción del sistema JAWS

Los patrones de pulsos generados siguen una técnica denominada modulación Sigma-Delta. La ventaja de este método particular es que tiene mucha fidelidad a la señal original, más precisamente: aumenta la relación señal a ruido. Aunque también hay que señalar que tiene algunas desventajas, como requerir mucho poder de cómputo frente a otras técnicas de conversión digital. Una comparación de algunas de estas técnicas se muestra en la figura 3

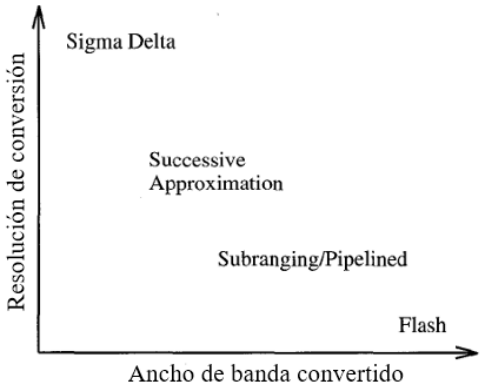

Figura 3: Comparación de distintos métodos conversión digital. La modulación Sigma-Delta es una de ellas.

La fidelidad a la señal consiste en limpiar el ruido presente en la zona de frecuencias de interés, y empujarlo hacia otras zonas del espectro, por lo general las altas frecuencias, para luego ser filtrado por un filtro pasabajos. Hay muchos tipos diferentes de modulación Sigma-Delta, como los que se centran en limpiar el ruido en una franja de frecuencia de interés (para luego pasar el filtro pasabanda correspondiente), pero no nos centramos en estos otros. También existen optimizaciones, que lo que hacen es generar pulsos que tengan aún menos ruido, habiéndolo empujado más, pero esto es a costa del hardware del que se disponga. Los pulsos generados por el algoritmo van a ser enviados a un generador de pulsos de corriente, FPGA (Field Programmable Gate Array) [**?**], en el cual se cargarán los pulsos de salida del script y generará la salida electrónica necesaria para el array de junturas Josephson. Las características de hardware del FPGA nos dirán qué modulaciones Sigma-Delta podremos usar, por ejemplo, no podremos usar ordenes L elevados.

1.2. Modulación $\Sigma\Delta$

La modulación Sigma-Delta [**?**] [**?**] es una técnica que permite representar una señal analógica a través de una señal digital de pocos valores. Intuitivamente, compensamos los pocos valores que podemos expresar digitalmente con una alta tasa de muestreo. Es una técnica muy conveniente para la construcción del JAWS, ya que tiene la particularidad de redistribuir el ruido no deseado hacia las altas frecuencias, lo que es ideal para el JAWS ya que el chip tiene impresos filtros pasabajos (figura 4), de modo que el ruido final se reduce considerablemente.

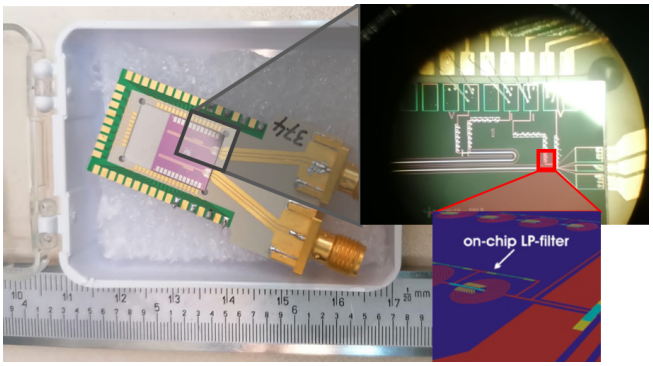

Figura 4: Filtros pasabajos impresos en el chip JAWS del PTB

La figura 4 muestra dos fotografías del chip del PTB actualmente presente en el INTI, junto a un gráfico sacado de la figura 24 del paper de Behr [**?**] mostrando el filtro pasabajos impreso en el chip.

Para el algoritmo, se parte de una señal temporal idealizada que llamamos $x(t)$, que representa la señal que queremos generar. Para ilustrar, supondremos que la señal es de tensión, aunque puede ser de unidades arbitrarias (u.a.). A lo largo de este trabajo, 1 u.a. representa al valor máximo que puede generar la salida del cuantizador, es decir, el generador de pulsos FPGA. A su vez, -1 u.a. es el valor negativo correspondiente.

A $x(t)$ se le realiza un muestreo temporal, i.e. quedarse con muestras de la misma a intervalos de tiempo iguales T_s, obteniendo una señal discretizada en el tiempo $x[n]$.

En procesamiento de señales, el teorema de Nyquist [**?**] dice que para toda señal cuyo espectro de frecuencias esté contenido en una banda de frecuencias $[-f_B, +f_B] \subset \mathbb{R}$, con f_B la menor frecuencia de muestreo tal que no se pierda ninugna información de la señal original es exactamente $f_N = 2f_B$, y es la llamada frecuencia de Nyquist. A f_B se la llama la frecuencia de banda, y es la mayor frecuencia que puede estar presente en la señal para que funcione el método.

El teorema de Nyquist asume que contamos con la posibilidad de generar cualquier señal en un rango contínuo que contenga a la señal original. Pero nosotros tenemos un generador de pulsos binarios, es decir, solo dos valores de tensión posibles: -1 y 1. Entonces, para poder llevar la información de la señal, vamos a tener que usar frecuencias de muestreo mucho mayores a f_N. Este proceso se llama oversampling.

Se define el ratio de sobremuestreo OSR (oversampling ratio) como en la ecuación 2, usando la misma definición que de la Rosa [**?**]. [1]

$$\text{OSR} = \frac{f_s}{f_N} = \frac{f_s}{2f_B} \tag{2}$$

[1] Comparando con el paper de de la Rosa [**?**], lo que ahí se llama B_w nosotros lo llamamos f_B

También definimos el ratio de muestreos por período N_s, exactamente como hace Benz en su paper de 1998 [**?**], en la ecuación 3.

$$N_s = f_s / f \tag{3}$$

La ecuación 3 solo aplica cuando la señal de entrada es AC, y su frecuencia es f.

En la modulación sigma delta, es fundamental usar frecuencias de muestreo lo más altas posibles, ya que es de esta forma que se aumenta la precisión al solo contar con dos valores del generador de pulsos. Para llevar la información que lleva una señal que vive en un rango contínuo, usando solo dos valores discretos, se tiene que tener una gran cantidad de estos valores discretos por cada valor contínuo que queramos representar, y esta cantidad aumenta conforme aumentemos el valor de f_s. En nuestro caso, el generador tiene una $f_s = 5\,\text{GHz}$

Luego, se discretiza la tensión, lo que quiere decir que de todo el contínuo de tensiones, solo nos quedaremos con un conjunto finito de valores. En nuestro caso son 2, que definen los límites del rango de la señal que podremos generar.

En el apéndice y en la documentación del código se describe como se obtiene la modulación de orden L. Como ya mencionamos, limitaciones de hardware impedirán en la práctica realizar modulaciones con un orden L elevado.

Al realizar la discretización en voltaje, lo que sucede es que estamos introduciendo un error, que llamamos ruido o error de cuantización e_q. Este error, en condiciones que son usuales en la práctica, puede ser modelado como ruido blanco, en otras palabras, como una variable aleatoria con función de distibución rectangular, entre los valores extremos del cuantizador.

Asumiendo que el error es ruido blanco, el efecto producido por la modulación Sigma-Delta de orden $L \in \mathbb{N}$ es el aplicarle un filtro pasabajos con la forma funcional mostrada en la ecuación

$$NTF(z) = (1 - z^{-}1)^L \tag{4}$$

$$z = e^{i2\pi f / f_s} \tag{5}$$

Así, en la figura 5 se muestra el efecto deseado de la modulación sobre el ruido de cuantización. El objetivo de esta modulación de pulsos particular es lograr que el ruido de cuantización se corra hacia lo más alto en frecuencias que nos sea posible. Mediante la modulación de orden L se logra que el ruido de cuantización se multiplique por una función moduladora llamada NTF (noise transfer function)

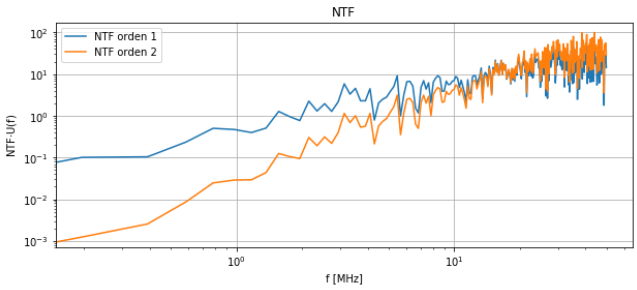

Figura 5: Comparación de los espectros de frecuencia de las NTF de orden 1 y 2

Para obtener lo observado en la figura 5, se modeló el ruido de cuantización como una variable aleatoria de dsitribución uniforme entre los valores del cuantizador, i.e. ruido blanco. Haciendo una transformada rápida de Fourier (FFT) de este ruido, se obtiene también ruido blanco, y este espectro es modulado por el valor absoluto de la NTF, que es lo que se observa en la figura 5. En particular, se observa que la modulación de orden 2 es más efectiva al trasladar más ruido desde las bajas hacia las altas frecuencias. Es importante notar que esto es solo un modelo teórico, dado que en realidad e_q no es una variable aleatoria. Pero más adelante (ver figura **??**) se va a confirmar que el ruido de la señal generada se corre conforme indica la NTF.

2. Descripción de la simulación numérica

El código utilizado genera distintas variables presentes en la modulación sigma delta, y permite hacer un análisis muy detallado a partir de las mismas. En la figura 6 se muestra el diagrama de bloques representativo del algoritmo de modulación de 1er orden [**?**].

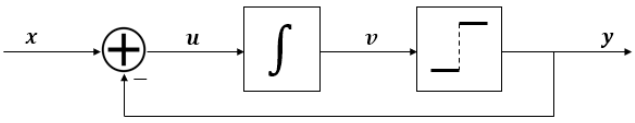

Figura 6: Diagrama de bloques correspondiente a la modulación ΣΔ de 1er orden

En el apéndice se describe muy brevemente como se leen estos diagramas, y como se pueden traducir a un método iterativo en las 4 variables presentes: x, u, v, y. La primera, x, representa la señal analógica de entrada, que es la señal que queremos generar. u y v son variables auxiliares, en donde v es la integral de u, y u es la diferencia $x - y$. y es la salida final del cuantizador, que tiene en nuestro caso solo 2 valores posibles.

En este trabajo se implementó, siguiendo diferentes publicaciones ([**?**], [**?**]), el algoritmo de modulación Sigma-Delta de 1er orden en Python. Esto permitió explorar diferentes aspectos del método que no eran muy accesibles desde otras implementaciones, de manera muy sencilla. Las implementaciones anteriores contienen optimizaciones adicionales que por el momento nuestro código no incluye. La más conocida es el paquete de Matlab de Richard Scherier [**?**], sobre la cual se basa el paquete de Python de G. Venturini [**?**].

En la figura 7 se muestra mejor el comportamiento de las variables u y v para cuando se quiere generar una señal DC de valor 0.8 u.a. Se observa que hay un período en el que la señal $u = x - y$ es ligeramente negativa, esto es porque $x = 0{,}8$ es ligeramente inferior al nivel superior $y = 1$ (todo en unidades arbitrarias). Durante este período, se van sumando estos valores de u al valor de v, que en la figura 7 es la pendiente negativa que está en el área de integración negativa, sombreada en azul.

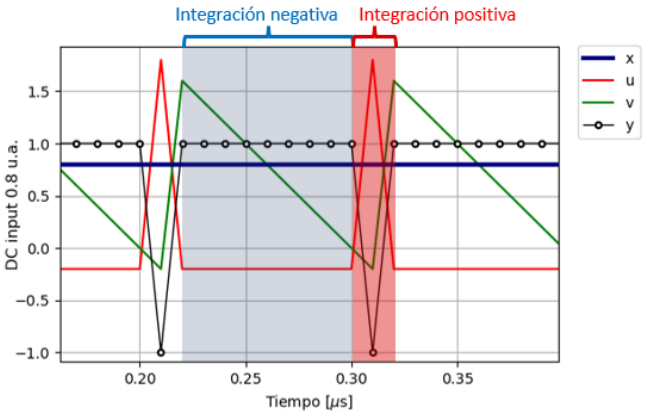

Figura 7: Ejemplo del funcionamiento del integrador en la modulación a 1er orden

Llega un punto en que v pasa a ser negativo, de modo que, al pasar al cuantizador, generará un pulso y negativo, luego de varios positivos. Esto es lo que se ve en el área de integración positiva: el valor de u, en ese lapso muy corto de tiempo, es positivo y muy grande, de tal manera que basta solo uno para volver al régimen de pulsos positivos. De esta manera el algoritmo genera la proporción de pulsos negativos y positivos deseada.

3. Descripción del experimento

4. Resultados y Análisis

Usando nuestro script [**?**], generamos los pulsos correspondientes a una señal DC para reproducir los resultados que se muestran en [**?**]. En la figura