Cómo trabajar con el DDS Compiler de Vivado

Creador: David Rubio G.

Entrada: https://soceame.wordpress.com/2024/11/19/como-trabajar-con-el-dds-compiler-de-vivado/

Blog: https://soceame.wordpress.com/

GitHub: https://github.com/DRubioG

Fecha última modificación: 23/02/2025

Una tecnología muy utilizada en FPGAs es la del uso de sistemas DDS (*Direct Digital Synthesis*) que se basa en crear formas de señal, digitalizarlas y añadirlas a la memoria de la FPGA, ya sea en una ROM o en lógica digital.

***NOTA**: si sabes explotar la tecnología DDS, puedes, además de generar formas de onda, realizar cálculos matemáticos complejos en pocos ciclos de reloj.*

En el caso que nos trae aquí vamos a explicar cómo utilizar el **DDS Compiler** que incorpora Vivado, que nos permite generar formas de señal basados en senos y cosenos.

Recomendación: El DDS compiler muchas veces genera errores por cómo está creado internamente, por eso recomiendo la creación de sistemas DDS propios, además, de que permite controlar el sistema de tiempos/frecuencia mejor junto con una forma de señal más predecible.

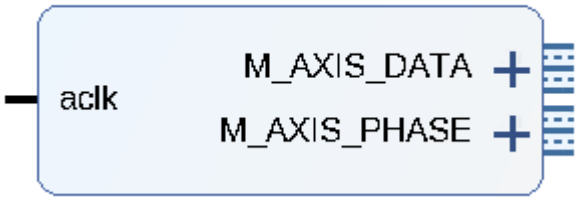
Implementación

Para acceder al DDS Compiler hay que llamarlo desde el IP Catalog.

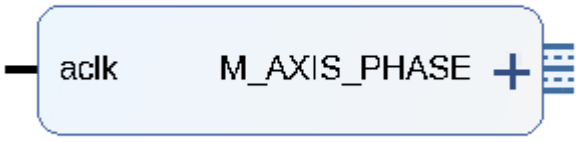
Al llamarlo, se nos abre la siguiente pestaña. En ella se puede elegir el tipo de configuración que se quiere del DDS Compiler.

Se pueden crear hasta 3 tipos de sistemas DDS.

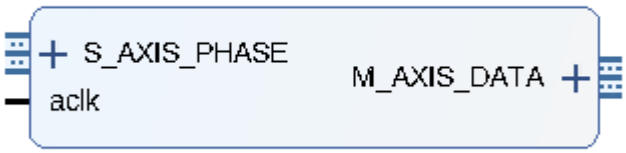
- **Phase Generator and SIN COS LUT:** esto permite crear un DDS controlado por tiempo, entonces desde el primer ciclo de reloj, generará una señal y también devolverá la fase de la señal que genera por el otro puerto.



- **Phase Generator Only:** en la que solo devuelve la fase de la señal seno, empezando desde cero.



- **SIN COS LUT:** en la que utilizando la fase devuelve el valor del seno que corresponde.



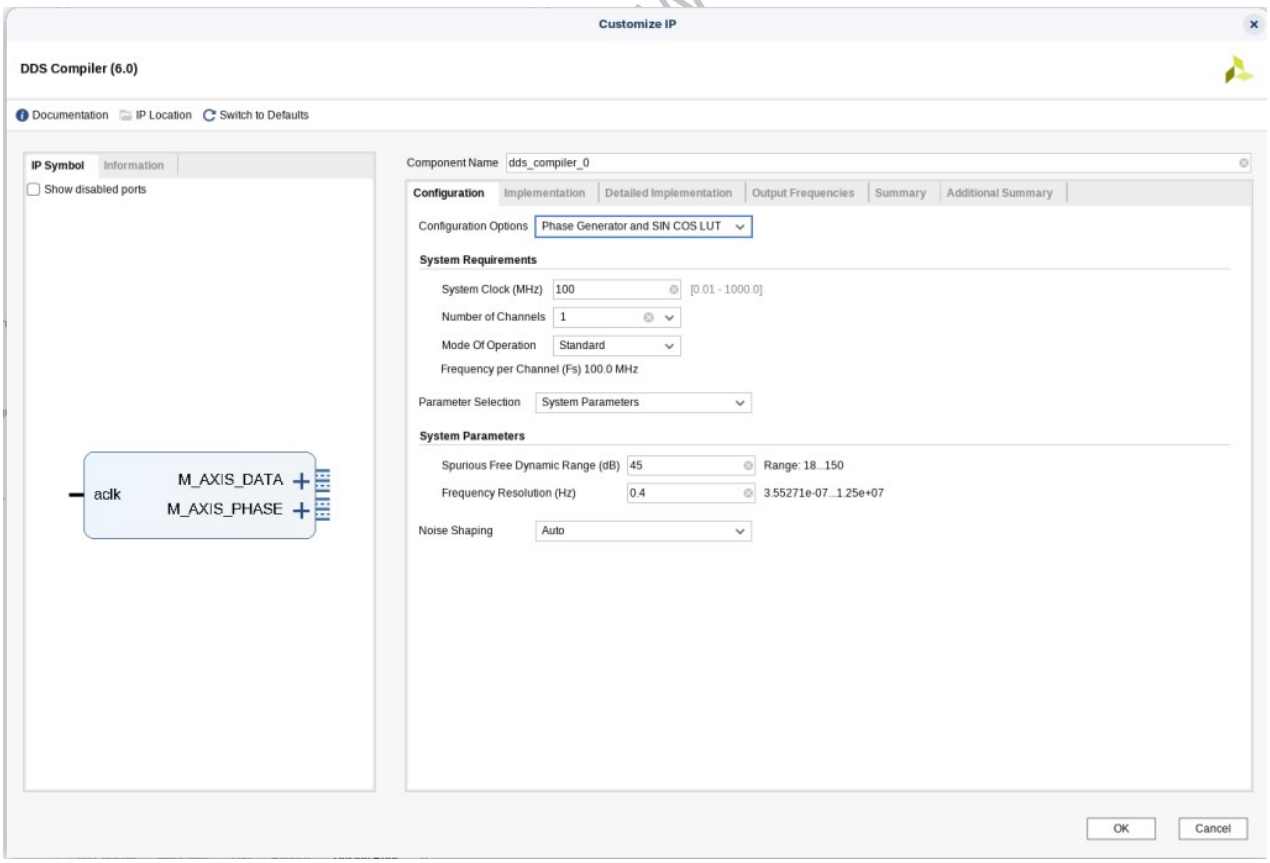
También en la imagen anterior se puede ver que se puede configurar la frecuencia de reloj base, que es la que se utiliza como referencia para los cálculos.

Otro parámetro que se puede seleccionar es el número de canales. Que lo que hace es distribuir muestras, de tal forma que la muestra t0, es del canal 0, la t1 del canal1, etc. Dividiendo las muestras de tiempo en canales, lo cuál reduce la frecuencia de las señales seno utilizando la fórmula: Fclk/Nºchannels.

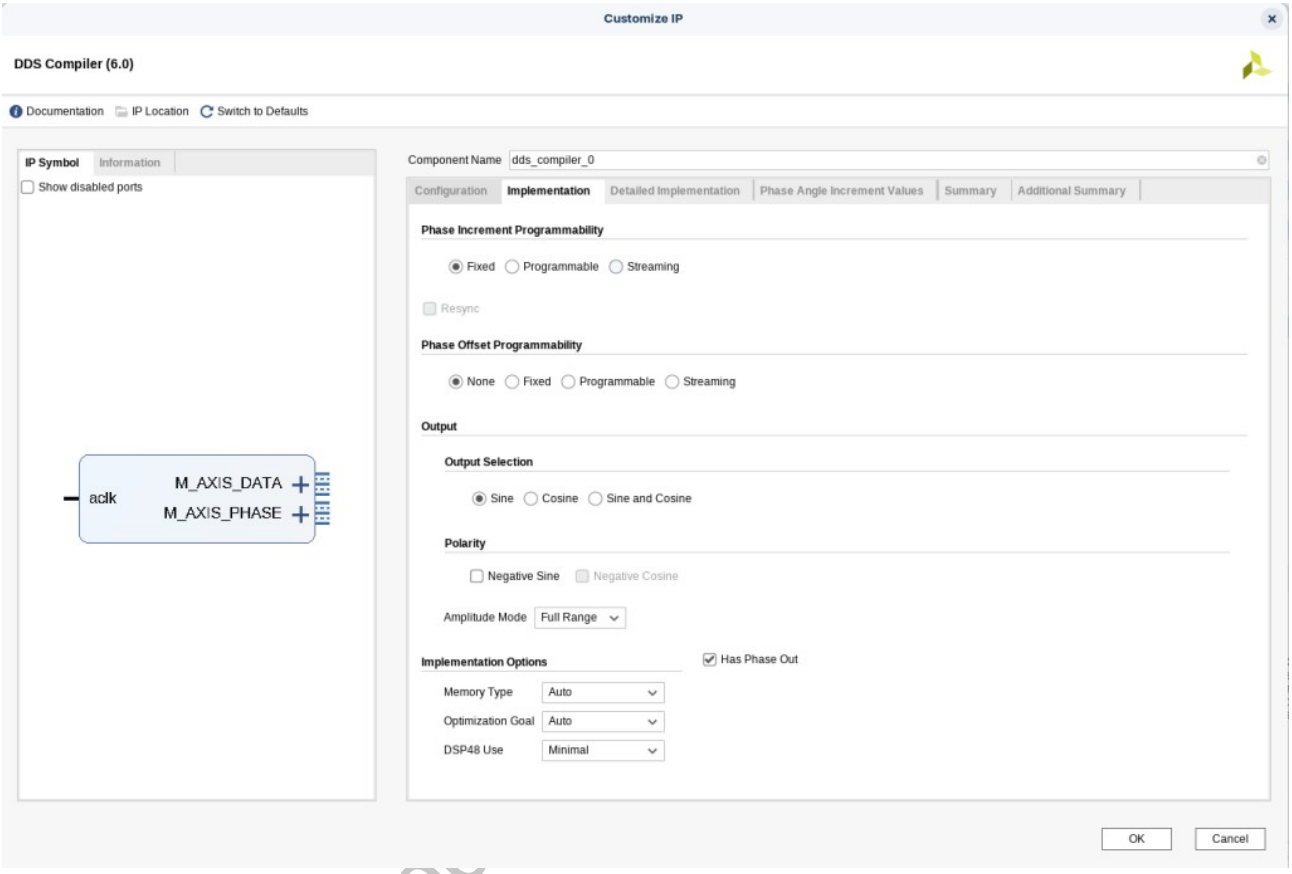
Luego tienes parámetros de ruido

Y por último tienes el tamaño de los puertos de salida y de la fase del seno. Este primer parámetro es importante porque es el que define el escalón de cuantificación del seno, o lo que es lo mismo, la precisión del seno.

q (escalón de cuantificación) = Amplitud del seno / 2^número de bits -1

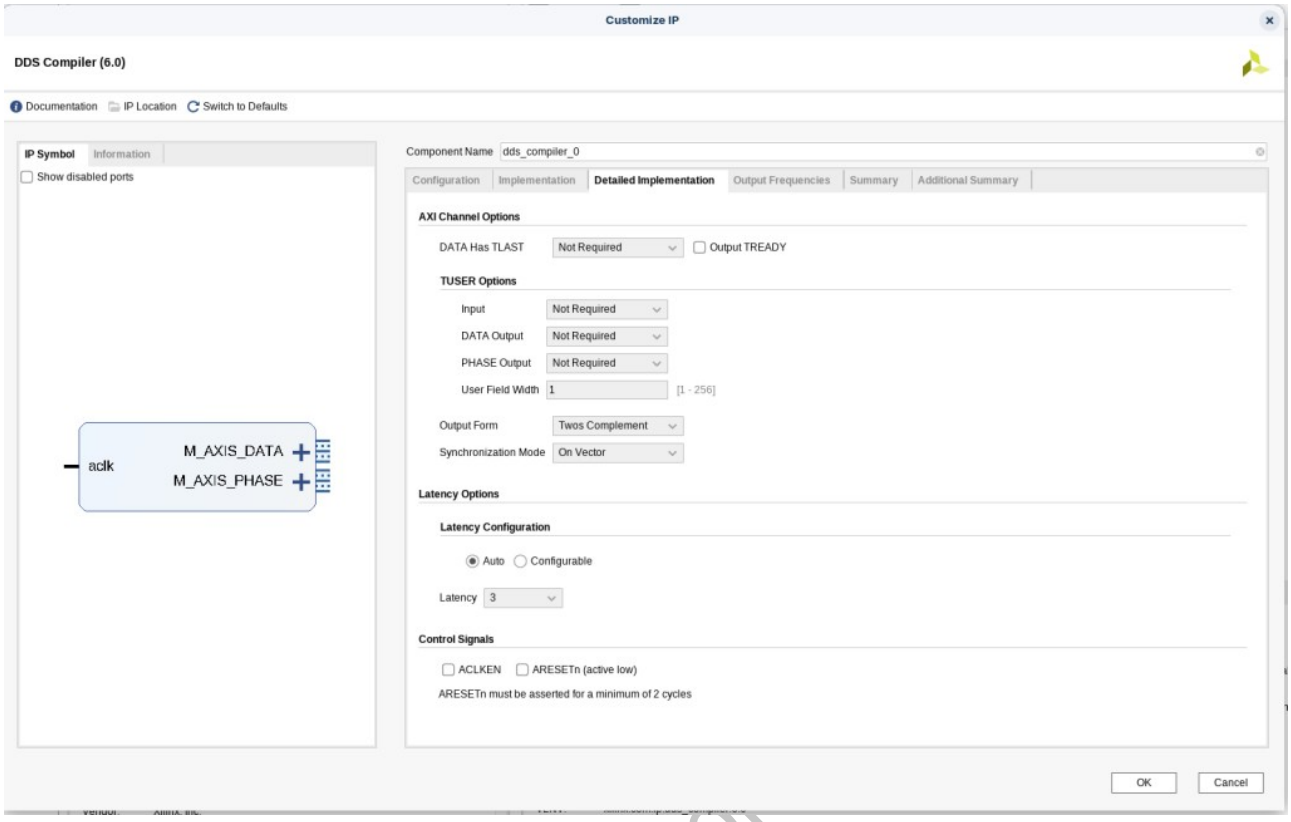


La pestaña de implementación permite elegir el tipo de muestras elegidas, o la selección de como se introduce la fase. También permite elegir si se quiere generar una señal seno, coseno o la suma de ambas señales, o la polaridad del seno/coseno. Y por último dónde se va a almacenar esa señal, si en una ROM distribuida o en una BROM *(BRAM preinicializada)*, y todas las optimizaciones deseadas para implementarlo.

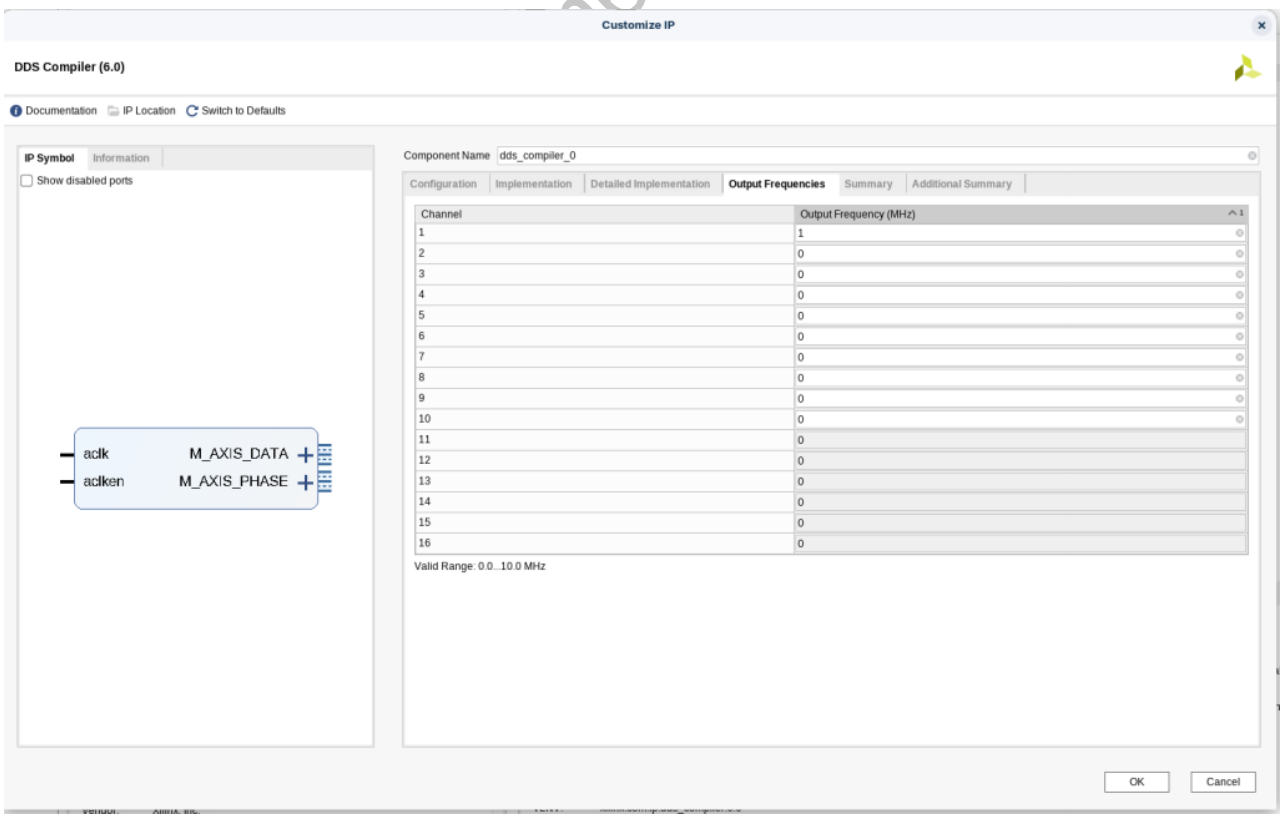


Luego tenemos la pestaña de *Detailed Implementation*, que está enfocada al AXI, pero tiene dos parámetros en la parte inferior que permiten añadir los puertos de *enable* y *reset negado*.

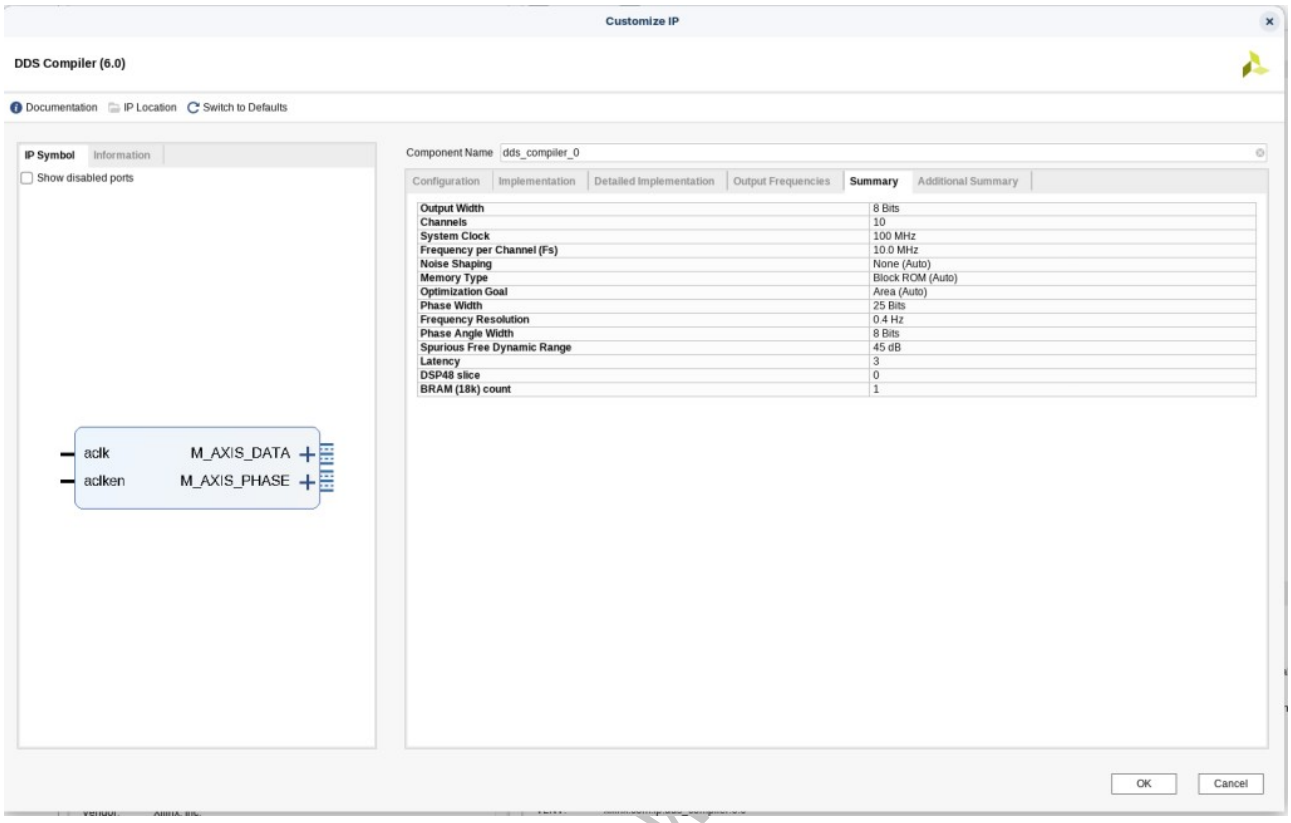
https://soceame.wordpress.com/2024/11/19/como-trabajar-con-el-dds-compiler-de-vivado/



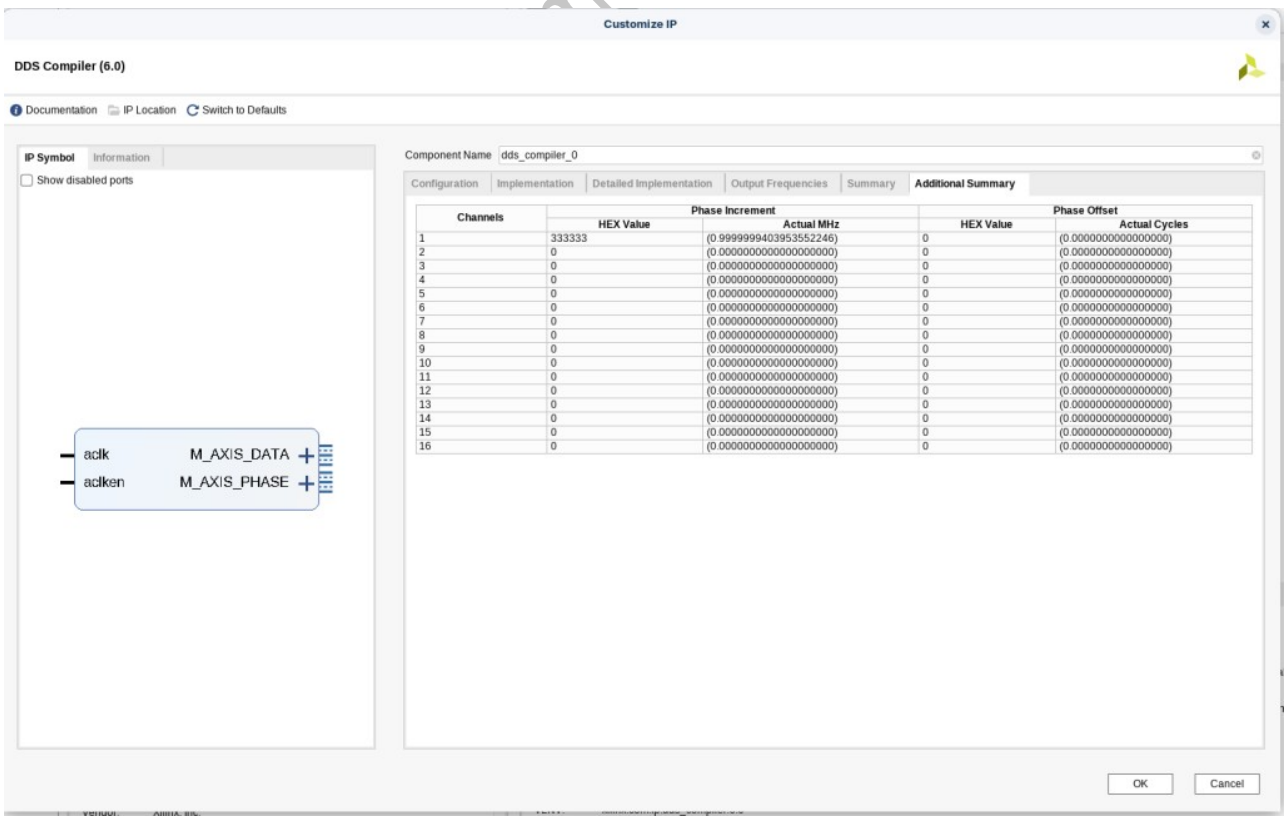
Luego tenemos la pestaña *Output Frequencies*, que permite seleccionar las frecuencias de cada canal, limitado a la fórmula anterior.



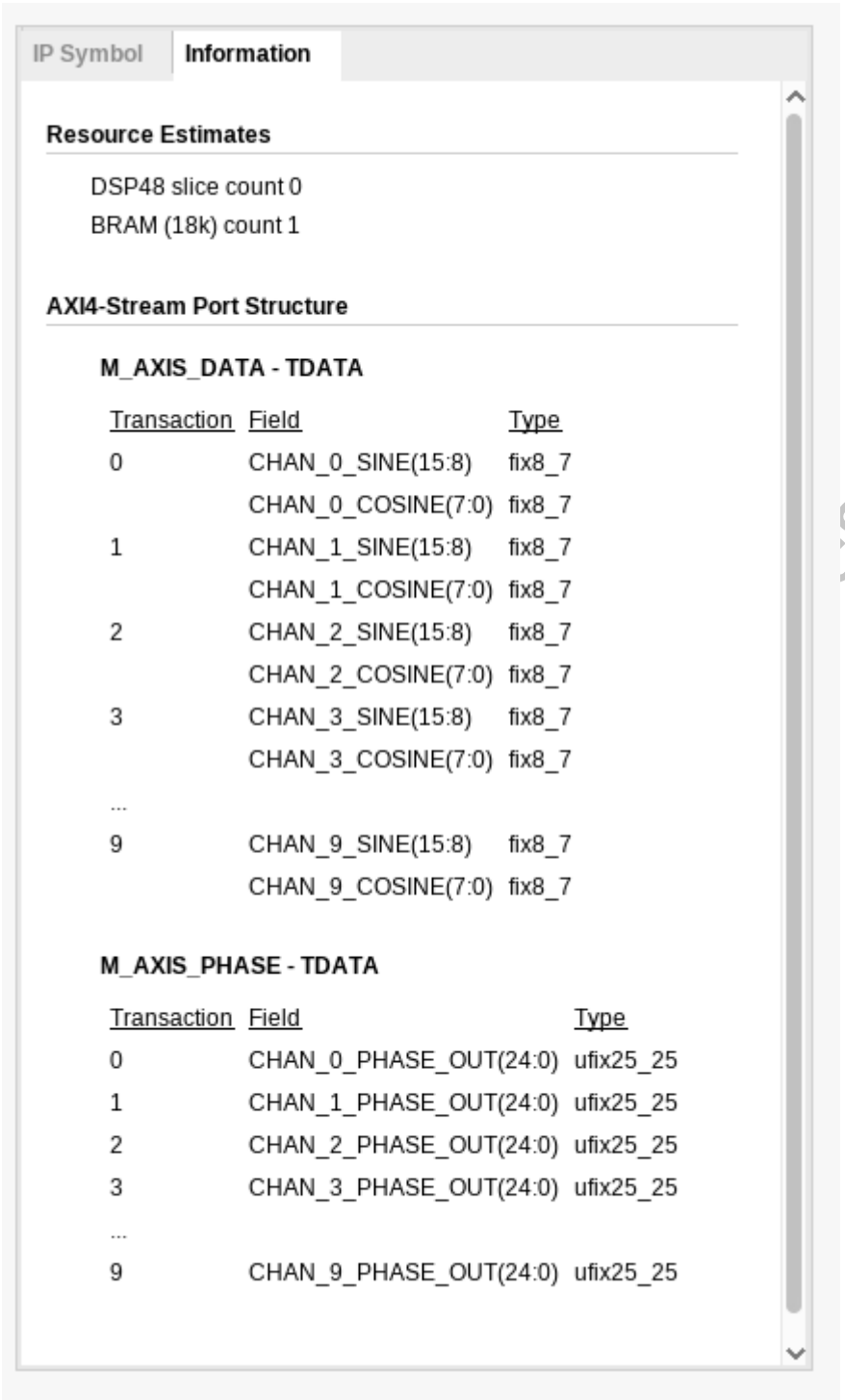
Luego un resumen de todos los recursos utilizados.



Luego una pestaña adicional.



Y por último la pestaña *Information*, que contiene información sobre los tipos de datos que se van a utilizar.



IP Symbol | **Information**

Resource Estimates

DSP48 slice count 0
BRAM (18k) count 1

AXI4-Stream Port Structure

M_AXIS_DATA - TDATA

Transaction	Field	Type
0	CHAN_0_SINE(15:8)	fix8_7
	CHAN_0_COSINE(7:0)	fix8_7
1	CHAN_1_SINE(15:8)	fix8_7
	CHAN_1_COSINE(7:0)	fix8_7
2	CHAN_2_SINE(15:8)	fix8_7
	CHAN_2_COSINE(7:0)	fix8_7
3	CHAN_3_SINE(15:8)	fix8_7
	CHAN_3_COSINE(7:0)	fix8_7
...		
9	CHAN_9_SINE(15:8)	fix8_7
	CHAN_9_COSINE(7:0)	fix8_7

M_AXIS_PHASE - TDATA

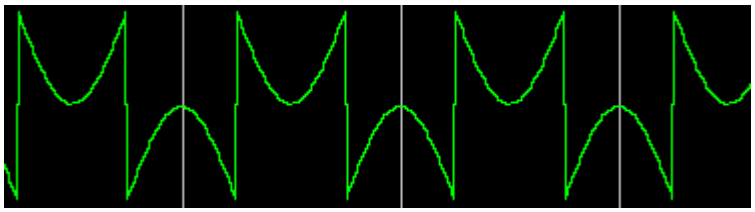
Transaction	Field	Type
0	CHAN_0_PHASE_OUT(24:0)	ufix25_25
1	CHAN_1_PHASE_OUT(24:0)	ufix25_25
2	CHAN_2_PHASE_OUT(24:0)	ufix25_25
3	CHAN_3_PHASE_OUT(24:0)	ufix25_25
...		
9	CHAN_9_PHASE_OUT(24:0)	ufix25_25

NOTA: Si ves que son muchos parámetros para configurar, lo mejor es que te crees tus propios DDS *(en una entrada futura te cuento cómo se hace en Python)*

Ejemplo

La forma más fácil de explicar es mediante un ejemplo, este ejemplo va a ser el más simple, vamos a generar tres DDS, una para el *seno*, una para el *coseno* y otra para el *Sine and Cosine*. Para ello vamos a configurar una frecuencia de reloj de 100MHz a 1 canal en cada DDS.

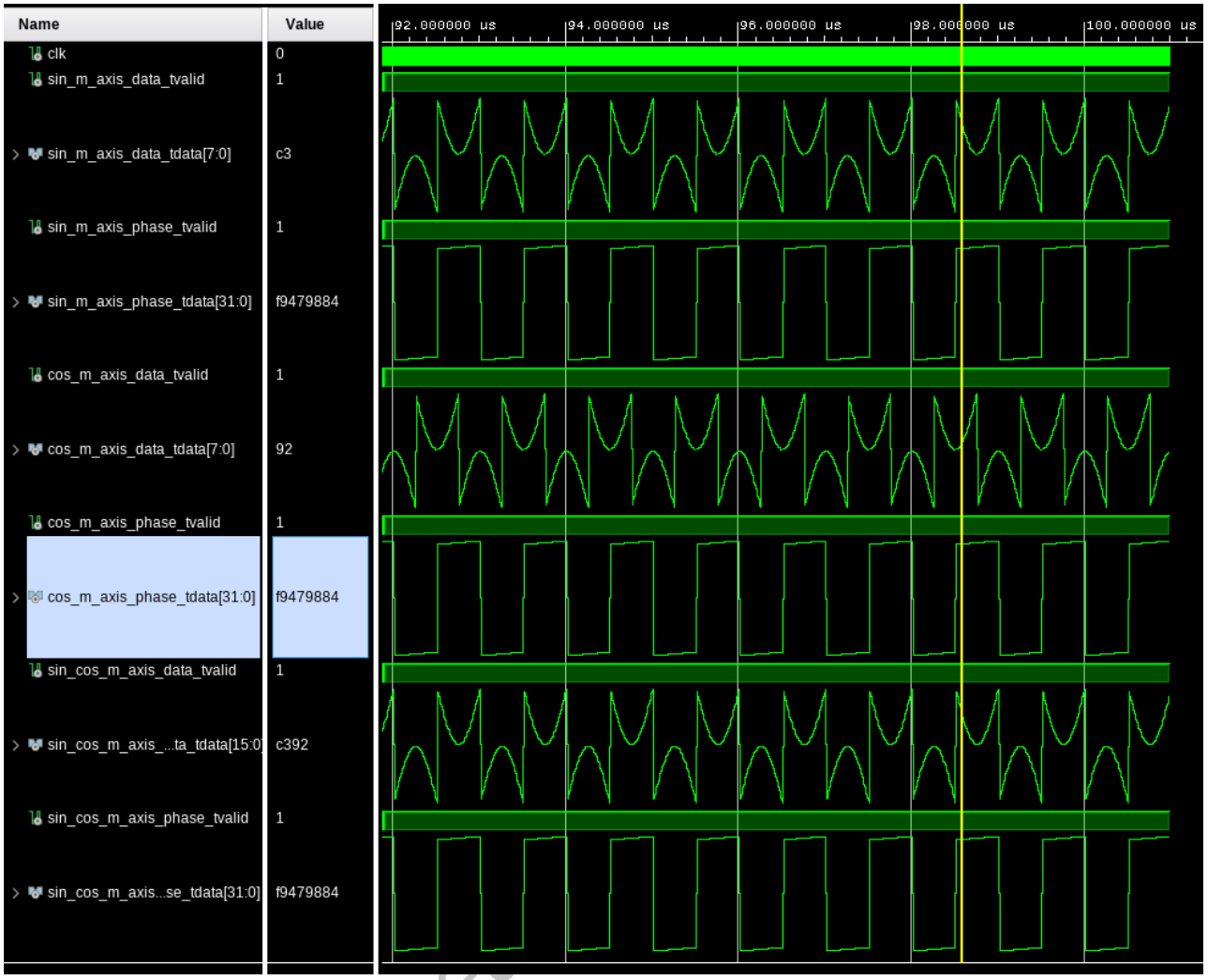
NOTA: si veis una forma de señal tal que así, es debido a que la señal no tiene un offset visible, siendo el 0 de la señal el valor 0x0, los valores positivos a hasta la mitad de la señal, y los valores negativos en complemento a dos, por eso al comparar datos en binario normal con datos en complemento a 2, el complemento a dos es más grande.



Para solucionarlo, lo que se puede hacer es añadir un offset de amplitud de la mitad de la señal, o cambiar el complemento a 2.

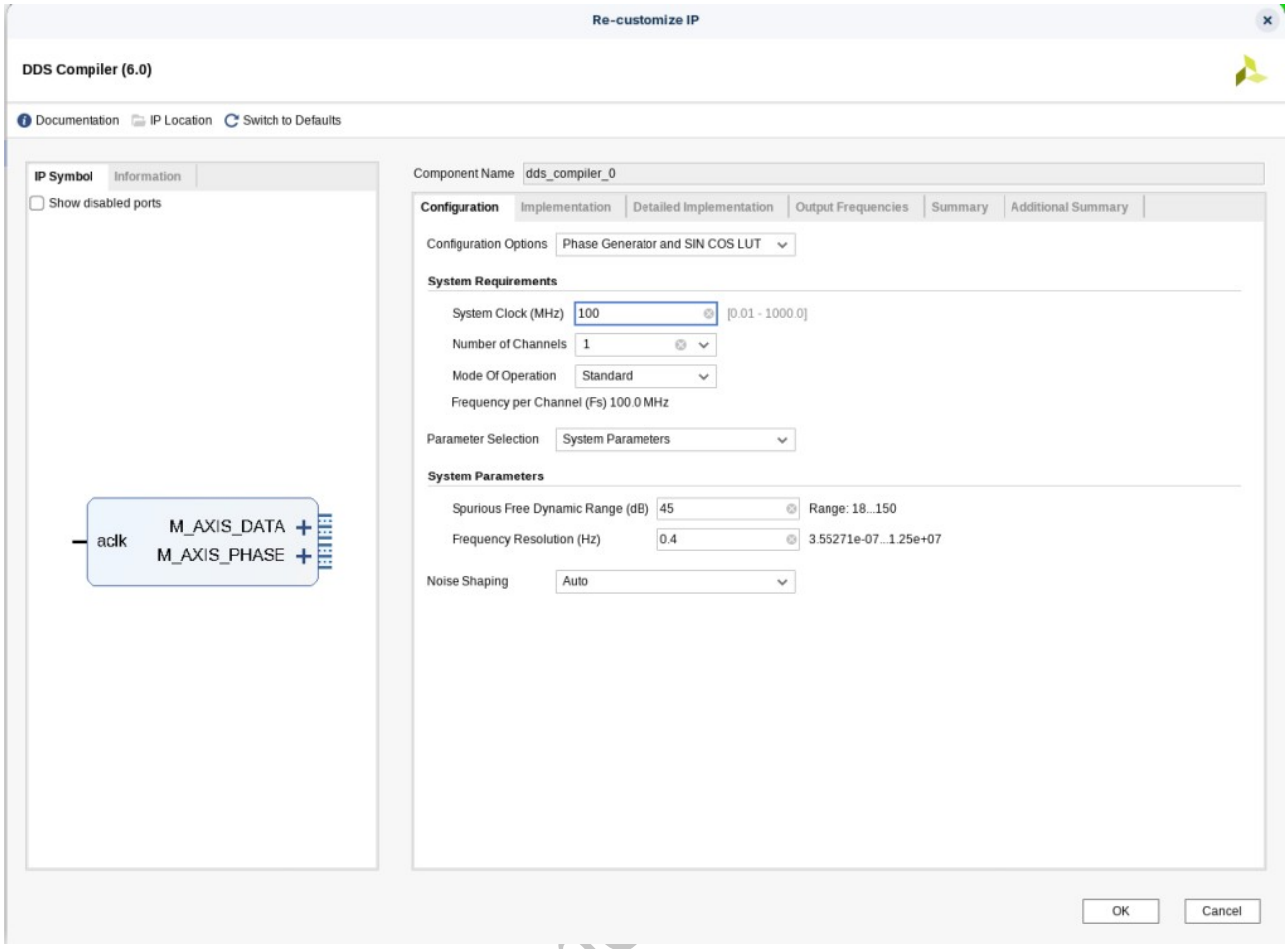
NOTA 2: sin veis cosas raras en la simulación, cortad. *Resetead* la simulación o *cerrad* y volved a *abrir* Vivado.

Después de muchas, pero que muchas simulaciones, llego a estas formas de señal generadas. Las de arriba son las señales seno, las del medio son señales coseno, y las del final las *Sine and Cosine*.



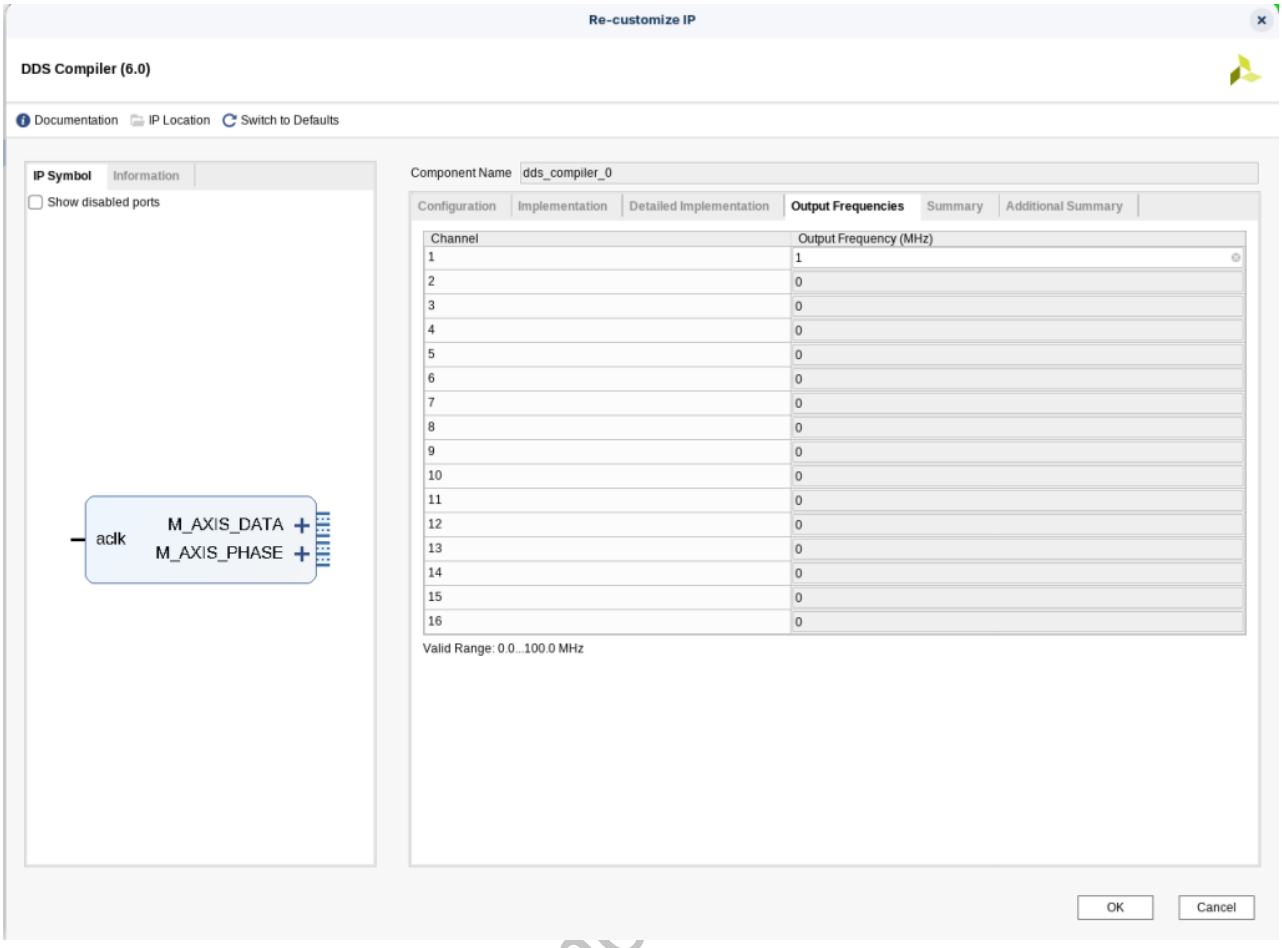
Se puede ver que la señal seno y coseno están desfasadas 90º grados, pero la fase de ambas son iguales, por lo que son dos señales totalmente distintas. Y la señal *Sine and Cosine* es muy parecida a la seno.

Para llegar a estas simulaciones tengo que declarar el **Parameter Selection** como *System Parameters*.



Y en **Output Frequencies** tengo que declarar que quiero una señal seno de 1MHz.

https://soceame.wordpress.com/2024/11/19/como-trabajar-con-el-dds-compiler-de-vivado/



NOTA FINAL

La recomendación que hago es **NO utilicéis este bloque IP**, da demasiados problemas.

Lo mejor que podéis hacer es crear vuestros propios senos/cosenos de forma manual como se puede ver en la imagen, en la que tengo un seno y un coseno, con un offset de la mitad de la amplitud para ver el seno completo, y que luego puedo modificar para poder operar con la señal.