

# Memoria Práctica E2

## Ruta de datos del modulador configurable de AM/FM

David Martínez Esteso

Néstor García García

# Tabla de contenido

**Sección 1: Descripción del módulo. .... 3**

**Sección 2: Interfaz. .... 4**

**Sección 3: Recursos Hardware. .... 5**

**Sección 4: frecuencia de operación. .... 7**

**Sección 5: Verificación..... 8**

**Sección 6: Resolución de problemas encontrados. .... 10**

## Sección 1: Descripción del módulo.

El esquema del módulo a desarrollar e implementar se muestra a continuación:

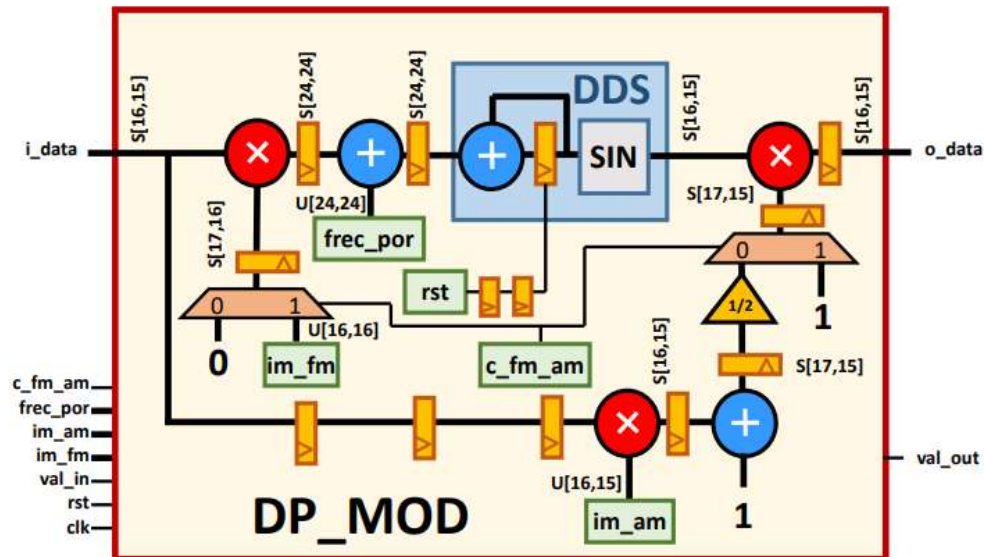


Ilustración 1: Esquema del módulo DP\_MOD

Este sistema implementa la ruta de datos de un modulador de señal configurable FM/AM, es decir, se ocupa de realizar modulación en fase o amplitud de la señal de entrada utilizando el módulo desarrollado en la práctica anterior (DDS). Para ello se deben realizar las operaciones que se muestran en el esquema anterior respetando los formatos especificados en las señales.

La entrada de datos (**i\_data**) y su salida (**o\_data**) son señales con signo de 16 bits, de los cuales son 15 fraccionales. La señal **c\_fm\_am** se trata de una entrada de un bit que configura el modulador en modo FM o AM. La señal **frec\_por** es una entrada con el valor de configuración de la frecuencia portadora, la cual tiene formato sin signo de 24 bits, de los cuales son 24 fraccionales. Las señales **im\_fm** e **im\_am** se tratan de señales de entrada de configuración con los valores de índice de modulación FM y AM, respectivamente. Además de las señales de reloj (**clk**) y reset (**rst**) para el módulo, se utiliza las señales **val\_out** y **val\_in** para indicar la presencia de una muestra válida en **o\_data** e **i\_data**, respectivamente.

El funcionamiento del sistema es el siguiente: primero se introducen los datos de configuración desde un fichero exterior, a continuación, se introducen desde otro fichero, el resto de las entradas del sistema, el cual funciona en modo FM o AM según el valor de las variables de configuración. Finalmente, tras los ciclos de reloj correspondientes según los registros indicados en el esquema, se tiene a la salida la onda resultante de la modulación.

## Sección 2: Interfaz.

En la siguiente sección se define el interfaz, sus formatos y sus parámetros:

Módulo DDS: PARÁMETROS	
Nombre	Descripción
M (24)	Tamaño del acumulador
L (15)	Número de bits usados para truncar la fase del acumulador
W (16)	Tamaño de los datos de salida

Módulo DP_MOD: INTERFAZ			
Nombre	Tipo	Formato	Descripción
clk	in	bit	Entrada de reloj
rst_ac	in	bit	Reset síncrono, activo a nivel alto
val_in	in	bit	Entrada de validación de la muestra de entrada
i_data	in	S [16,15]	Entrada de datos a modular
c_fm_am	in	bit	Entrada configuración modulador FM/AM
frec_por	in	U[24,24]	Entrada del valor de configuración de la frecuencia portadora.
im_am	in	S[W,W-1]	Entrada de configuración del índice de modulación de AM.
im_fm	in	S[W,W-1]	Entrada de configuración del índice de modulación de FM.
o_data	out	S[16,15]	Salida de datos modulados.
val_out	out	bit	Señal de validación de la muestra de salida

La nomenclatura de los distintos elementos del esquema de implementación se muestra en la siguiente imagen. Esta nomenclatura es la que se ha utilizado para distinguir los elementos en el código Verilog desarrollado.

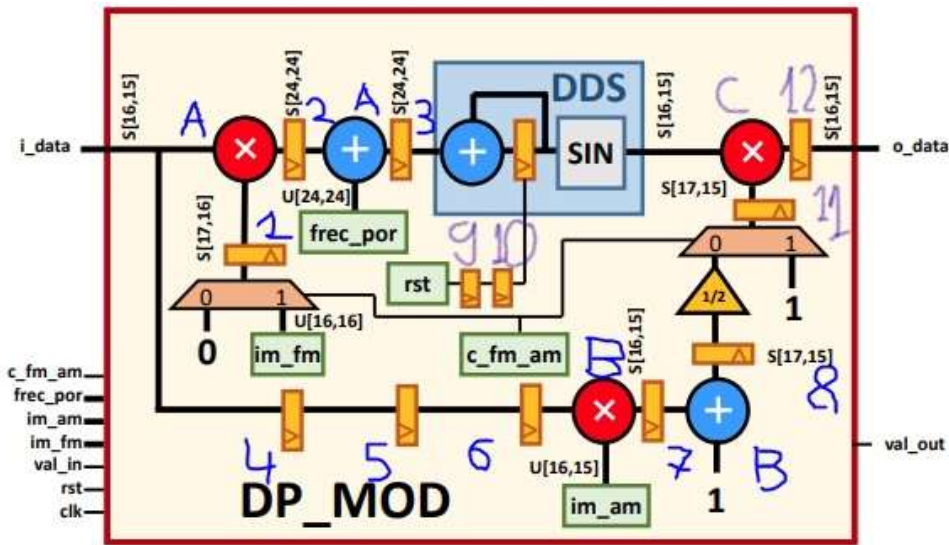


Ilustración 2: Nomenclatura elementos esquema.

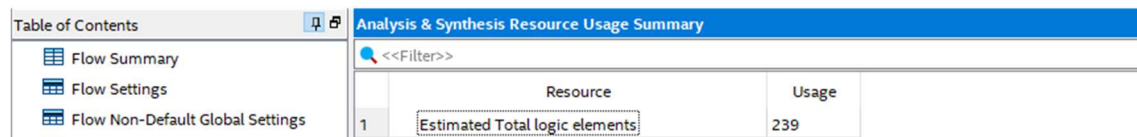
## Sección 3: Recursos Hardware.

### Logical Elements (LEs):

Se estiman los elementos lógicos utilizados teniendo en cuenta los bits utilizados en cada bloque del sistema y sus registros:

LEs (estimación) =  $24(\text{salida reg2}) + 24(\text{salida reg3}) + 16(\text{salida DDS}) + 17(\text{salida reg1}) + 16(\text{salida reg12}) + 16(\text{salida reg4}) + 16(\text{salida reg5}) + 16(\text{salida reg6}) + 17(\text{salida mux2}) + 16(\text{salida reg7}) + 17(\text{salida reg8}) + 17(\text{salida reg11}) + 1(\text{salida reg9}) + 1(\text{salida reg10}) = 214 \text{ LEs.}$

Se comprueba que el número estimado se asemeja a los LEs utilizados en el proyecto Quartus:

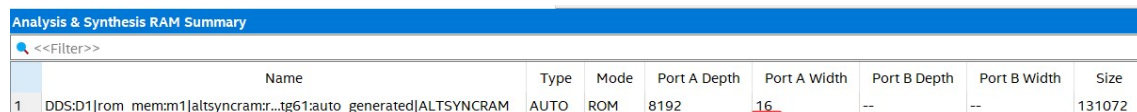


Analysis & Synthesis Resource Usage Summary	
<<Filter>>	
Resource	Usage
1 Estimated Total logic elements	239

### M9Ks:

M9Ks(estimación) =  $W * 2^{(L-2)} * 1(\text{ROM\_sin\_wave}) = 16 * 2^{(13)} * 1 = 16 \text{ M9Ks.}$

Como se comprueba en la siguiente imagen encaja con el análisis de la RAM del proyecto Quartus:



Analysis & Synthesis RAM Summary								
<<Filter>>								
	Name	Type	Mode	Port A Depth	Port A Width	Port B Depth	Port B Width	Size
1	DDS:D1 rom_memm1 altsyncramr...tg61:auto_generated ALTSYNCRAM	AUTO	ROM	8192	16	--	--	131072

### Mults:

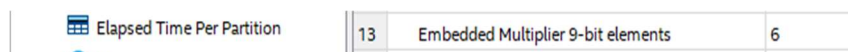
Nº multiplicadores = 3

Nº sumadores = 2

Nº divisores = 1

Total = 6

Se muestran a continuación los mults utilizados por el proyecto Quartus:



Elapsed Time Per Partition		
13	Embedded Multiplier 9-bit elements	6

Finalmente se muestra el sumario de recursos total del proyecto Quartus:

Table of Contents

- Flow Summary
- Flow Settings
- Flow Non-Default Global Settings
- Flow Elapsed Time
- Flow OS Summary
- Flow Log
- Analysis & Synthesis
  - Summary
  - Settings
  - Parallel Compilation
  - Source Files Read
  - Resource Usage Summary
  - Resource Utilization by Entity
  - RAM Summary
  - DSP Block Usage Summary
  - Optimization Results
  - Source Assignments
  - Parameter Settings by Entity Instance
  - LPM Parameter Settings
  - Connectivity Checks
  - Post-Synthesis Netlist Statistics for
  - Elapsed Time Per Partition
  - Messages
- Fitter
  - Flow Messages
  - Flow Suppressed Messages
- Assembler
- TimeQuest Timing Analyzer

Analysis & Synthesis Resource Usage Summary

<<Filter>>

	Resource	Usage
1	Estimated Total logic elements	239
2		
3	Total combinational functions	110
4	Logic element usage by number of LUT inputs	
1	-- 4 input functions	0
2	-- 3 input functions	61
3	-- <=2 input functions	49
5		
6	Logic elements by mode	
1	-- normal mode	49
2	-- arithmetic mode	61
7		
8	Total registers	238
1	-- Dedicated logic registers	238
2	-- I/O registers	0
9		
10	I/O pins	93
11	Total memory bits	131072
12		
13	Embedded Multiplier 9-bit elements	6
14		
15	Maximum fan-out node	clk~input
16	Maximum fan-out	254
17	Total fan-out	1213
18	Average fan-out	2.18

Ilustración 3: Sumario recursos proyecto Quartus

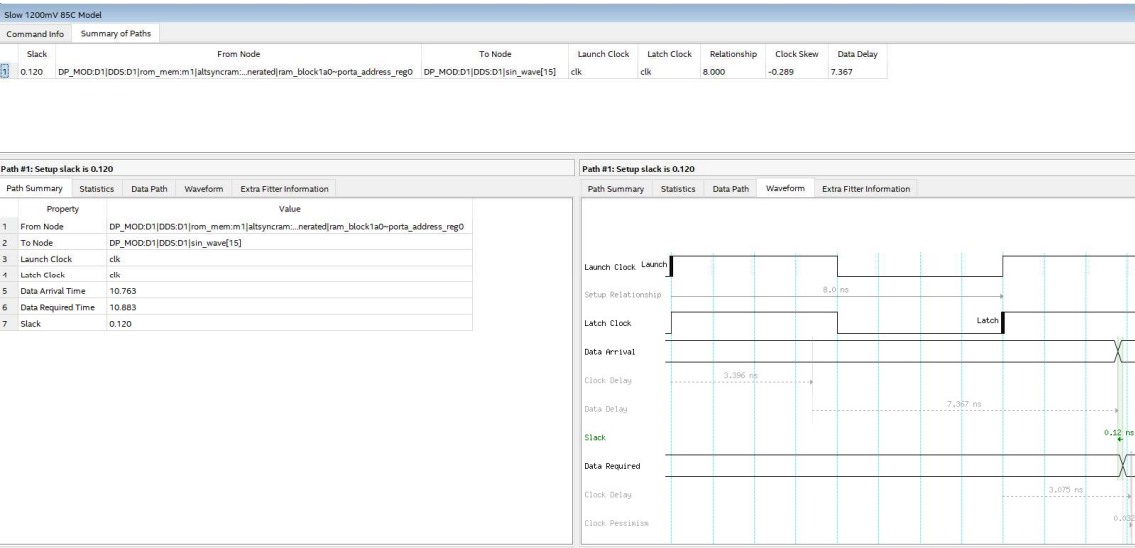
## Sección 4: frecuencia de operación.

Para la medida de la frecuencia de operación, se tiene en cuenta que la entrada que almacena el valor del índice de modulación de fase (im\_fm) entra un ciclo anterior al resto de señales, razón por la que presenta un registro (a la salida del mux) para sincronizarse con estas (esto también se ha tenido en cuenta en la realización del test\_bench). Por esta razón, en la realización del fichero de wrap (DP\_MOD\_wrap.v), no se ha registrado esta señal (im\_fm) y simplemente se ha conectado con un cable, simulando que todas las entradas se incorporan al circuito en el mismo ciclo. De no realizarse así, el cálculo de la frecuencia Fmax con la herramienta Time Quest Analyzer da un error temporal.

La frecuencia de operación del sistema obtenida mediante esta herramienta es de 126.9 MHz, cumpliendo así con la especificación de  $F_{max} \geq 125 \text{ MHz}$ .

Set Operating Conditions		Slow 1200mV 85C Model			
<input checked="" type="radio"/> Slow 1200mV 85C Model		Fmax	Restricted Fmax	Clock Name	Note
<input type="radio"/> Slow 1200mV 0C Model		1	126.9 MHz	126.9 MHz	clk
<input type="radio"/> Fast 1200mV 0C Model					

El camino crítico del sistema se encuentra, de nuevo, en la memoria ROM del bloque DDS que se utiliza para la generación de la onda senoidal.







funcionamiento del sistema, del cual cabe destacar la señal de entrada de configuración en modo fase (im\_fm), la cual requiere tener un valor asignado un ciclo antes que las variables de reset (rst), entrada de datos (i\_data) y muestra válida (val\_in). Para ello es necesario incorporar, en un primer ciclo, todas las variables de configuración (provenientes del fichero de configuración). Un ciclo después se introducirá el primer dato de entrada (i\_data) y se activará a nivel alto la señal de reset (rst) y muestra válida (val\_in). Para llevar a cabo esta sincronización, se ha utilizado la variable load\_data en el código desarrollado.

Mediante la variable val\_out se sincroniza el instante en el cual se comienzan a comparar los datos del modelo desarrollado con el modelo de ejemplo de Matlab. En la comparación se lleva la cuenta de las muestras comparadas (sample\_cnt) y de los posibles errores (error\_cnt).

De esta forma, en un primer ciclo se comienza el proceso de lectura de datos de configuración y en el siguiente se comienza con el proceso de lectura de datos de entrada. En el ciclo en que se indique muestra válida en salida, se comienza con el proceso de lectura de datos de salida y su comparación. Finalmente se muestran por consola el número de muestras procesadas y los errores encontrados.

**Importante:** Para cambiar de un modo de funcionamiento a otro (FM o AM), se deben comentar / descomentar las correspondientes líneas que abren los ficheros que se deseen en el proceso inicial (está indicado).

Formas de onda con configuración del modulador en modo FM:

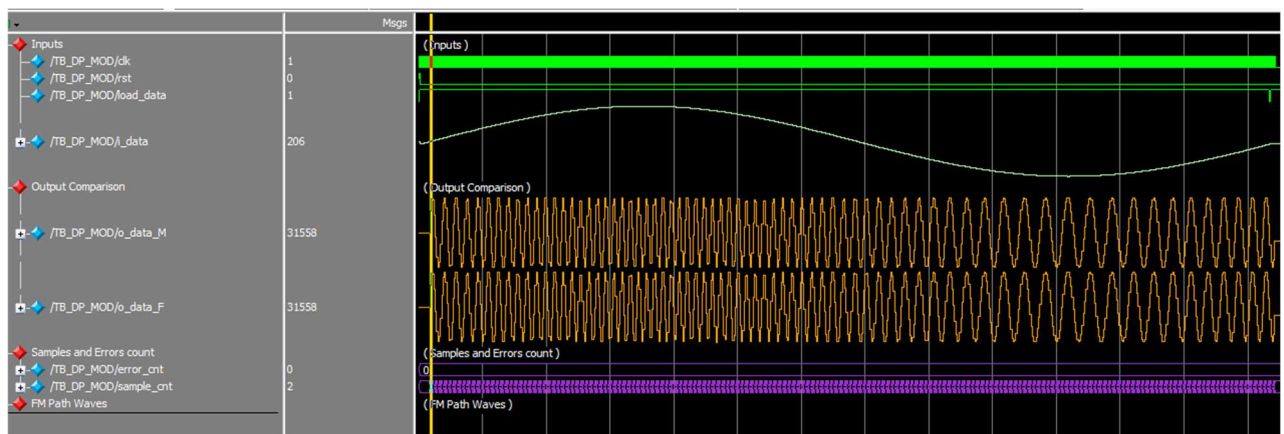


Ilustración 4 4: Ondas de salida modo FM

Formas de onda con configuración del modulador en modo AM:

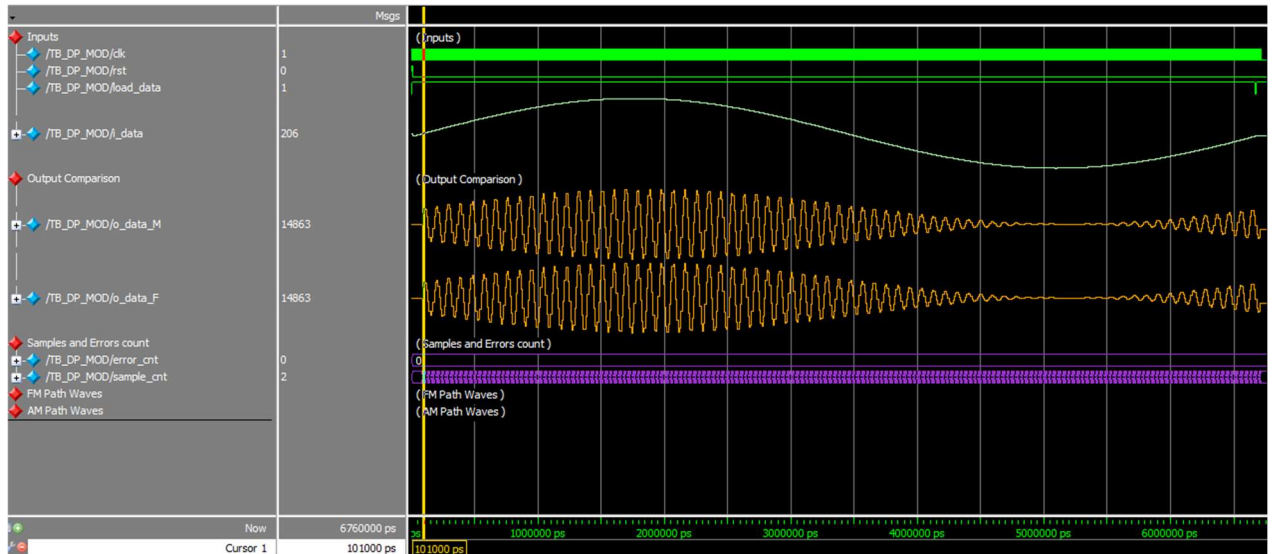


Ilustración 55: Ondas de salida modo AM

En ambos casos se observa que se han cumplido con las especificaciones del diseño ya que se tiene completa coincidencia entre las formas de onda generadas a partir del sistema desarrollado y las generadas en Matlab.

## Sección 6: Resolución de problemas encontrados.

A continuación, se presentan los problemas encontrados en la práctica y su solución:

- El principal problema que se ha tenido ha sido con la entrada de los datos de configuración un instante antes que el resto de las entradas. En un primer momento no se tuvo en cuenta y no se obtenía la salida adecuada puesto que se perdían muestras en el acumulador o no se llegaba a tiempo a este. Finalmente, se observó que la electrónica estaba preparada para recibir los datos de configuración un ciclo antes, y se modificó el testbench en consecuencia.
- La variedad de formatos en las variables y su tratamiento en operaciones también supusieron fallos a lo largo del test del sistema, pero se logró identificar y corregir dichos errores.
- En la obtención de la frecuencia máxima de operación se tuvo un problema con el hecho de que los datos de configuración entrasen un ciclo anterior al resto pero se solucionó tal y como se comenta en la sección 4.