Reporte DDS basado en memoria RAM con maquina de control

Alumno: Jhontan Alexander Gomez Gamboa ;
Profesor: Luis René Vela García

1 Introducción experimental.

La creación e implementación del siguiente diseño digital tiene la función de ser un DDS basado en una memoria RAM de doble puerto simple con una señal de reloj simple, el control de nuestro diseño digital se implementó utilizando una maquina de estados mealy de 4 estados. Todos los módulos a excepción de la maquina de estados finito son parametrizables.

El archivo .do necesario para poder ver todas las señales de la simulación es (DDS_run_msim_rtl_verilog.do) dentro de la carpeta simulation/modelsim.

2 Ambiente experimetal.

Quartus (Quartus Prime 17.7 Lite Edition).

ModelSim.

3 Programa.

El modulo_top o principal se encarga de instanciar todos los módulos, interconectarlos y poder definir las entradas y salidas de interés (figura 1).

El diseño consta de un módulo tunning_word (figura 2) el cual se encarga una vez dentro del estado de RAM_INIT mientras se están guardando los datos dentro de la memoria podemos mandar una señal de instruct para poder guardar uno de estos valores de locación de memoria como el valor de nuestro tuning Word para generar la señal q, posteriormente tomando estos saltos en los valores. Para consultar las entradasy salidas consultar la tabla 1.

Table 1: módulo tuning_word entradas y salidas

nombre	**tamaño**	**descripción**
clk	1 bit	señal de reloj
ena_tw	1 bit	señal de habilitación para registrar valor del tuning
tw_in	ADDR_WIDTH	señal tomada de la dirección de escritura para tuning
tw_out	ADDR_WIDTH	señal de salida para el valor del tuning

El módulo simple_dual_port_ram_single_clock (figura 3) el cual describe el funcionamiento de una memoria ram de doble puerto simple y una sola señal de reloj, tendrá dos señales de direcciones una para la escritura y otra para la lectura, la señal de escritura será generada desde nuestro testbench y la señal de lectura será generado por el modulo phase_generator una vez la memoria este llena, estas operaciones las

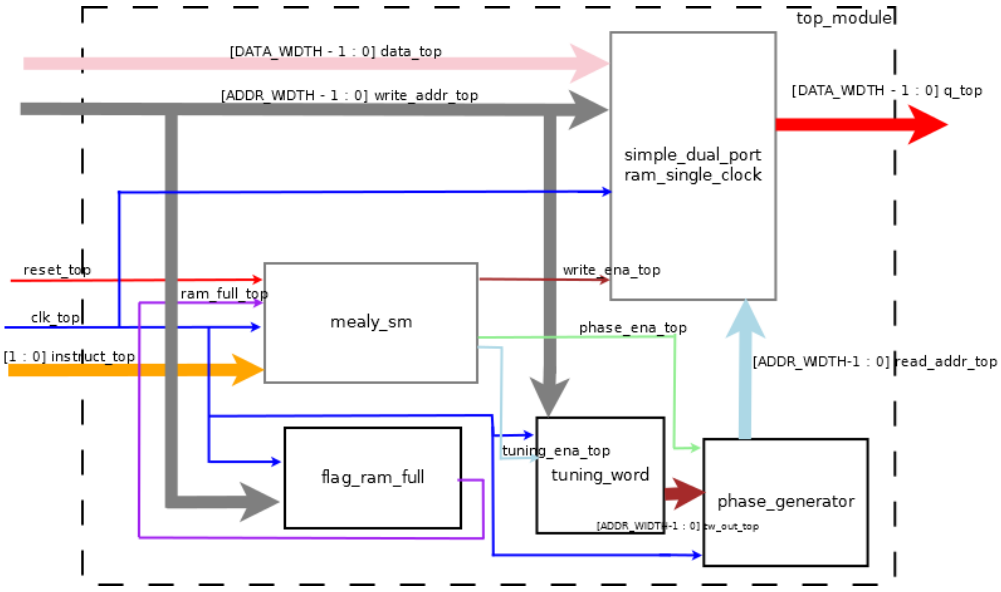

Figure 1: *diagrama de los módulos instanciados y conexiones de entradas y salidas dentro del* ***top_module***

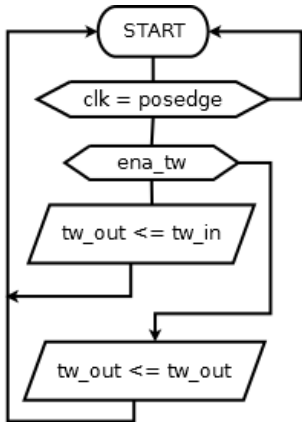

Figure 2: *Diagrama de flujo para al creación del módulo* ***tuning_word*** *parametrizado*

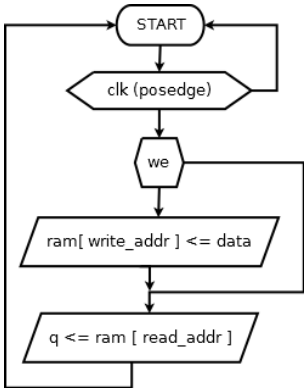

Figure 3: *Diagrama de flujo para la creación del módulo* ***simple_dual_port_ram_single_clock*** *parametrizado*

hará en cada ciclo de reloj y solo podrá inicializar esta operación si recibe una señal de habilitación de la máquina de estados. Para consultar las entradas y salidas consultar la tabla 2.

Table 2: módulo simple_dual_port_ram_simple_clock

nombre	**tamaño**	**descripción**
clk	1 bit	señal de reloj
we	1 bit	señal de habilitación para escritura en la memoria
read_addr	ADDR_WIDTH	datos para lectura de la memoria
write_addr	ADDR_WIDTH	datos para escritura de la memoria
data	DATA_WIDTH	datos de entrada que se escribiran en la memoria
q	DATA_WIDTH	datos de salida de los datos guardados en la memoria

El módulo phase_generator (figura 4) es un contador,tendrá como entrada una señal de reloj y una señal de reinicio para inicializar el valor del conteo, este conteo hace uso del valor recibido del módulo tuning_word para poder hacer los incrementos en las direcciones de memoria a leer, la señal de salida que representa las direcciones de lectura será conectada a la memoria ram. Para consultar las entradas y salidas consultar la tabla 3.

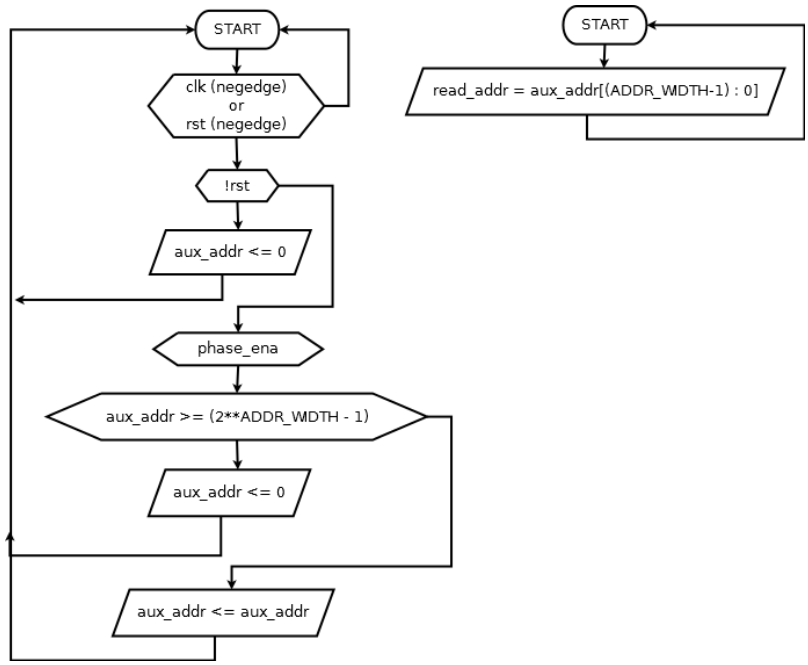

Figure 4: *Diagrama de flujo para la creación del módulo* ***phase_generator*** *parametrizado*

Table 3: módulo phase_generator entradas y salidas

nombre	**tamaño**	**descripción**
clk	1 bit	señal de reloj
rst	1 bit	señal de reset para inicializar el generador de fase
phase_ena	1 bit	señal de habilitación para el generador de fase
tw_in	DATA_WIDTH	datos de entrada del modulo tuning para el generador

nombre	**tamaño**	**descripción**
read_addr	ADDR_WIDTH	datos de locación de memoria enviados a la ram

El módulo Mealy_sm (figura 5) el cual describirá el funcionamiento de la maquina de estados la cual tendrá la labor de generar las señales de habilitación para los demás módulos tomando en consideración las banderas de memoria llena y el valor de la señal instruct, en el estado IDLE todas las señales de habilitación están en cero, si introducimos el valor de 1 en la variable instruct nos moveremos al estado RAM_INIT donde se inicializaran los valores de la memoria, si la bandera de memoria ram llena esta en alto nos devolvemos al estado IDLE, de lo contrario se continuara con la inicialización, si estando en este estado colocamos la variable instruct con un valor de 2, se habilitara la señal del tuning Word para poder capturar el valor actual de dirección y regresamos al estado de RAM_INIT, una vez la bandera de memoria llena este en alto nos devolvemos al estado IDLE, donde si colocamos el valore de instruct en 3, nos moveremos en el siguiente ciclo de reloj al estado DDS_RUNNING donde comenzaremos a generar los valores que están en la memoria, si la bandera de memoria llena esta en bajo nos regresaremos al estado IDLE, de lo contrario nos quedaremos en el estado DDS_RUNING hasta que termina la simulación. Para consultar las entradas y salidas consultar la tabla 4.

Table 4: módulo mealy_sm entradas y salida

nombre	**tamaño**	**descripción**
clk	1 bit	señal de reloj
reset	1 bit	señal de reset para inicializar la máquina de estados
ram_full	1 bit	bandera para indicar que la memoria ram esta llena
instruct	2 bits	instrucciones para cambiar de estados del controlador
write_ena	1 bit	señal de habilitación para escribir en la memoria
phase_ena	1 bit	señal de habilitación para activar el generador de fase
tuning_ena	1 bit	señal de habiliación para activar el registro del tuning

El módulo flag_ram_full (figura 6) nos permitirá generar la bandera de memoria llena cuando se evalué que se utilizó la última locación de memoria, si no, mantendrá la bandera en bajo, esta evaluación se hará en cada ciclo de reloj positivo. Para consultar las entradas y salidas consultar la tabla 5.

Table 5: módulo flag_ram_full entradas y salida

nombre	**tamaño**	**descripción**
clk	1 bit	señal de reloj
addr	ADDR_WIDTH	datos de locación de memoria para escritura
ram_full	1 bit	bandera de ram llena

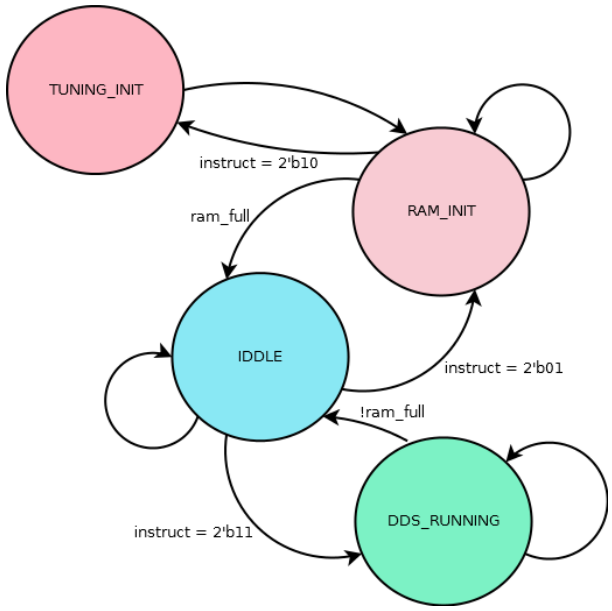

Figure 5: *Diagrama de estados que representa la finite state machine Mealy de 4 estados del módulo* ***Mealy_sm***

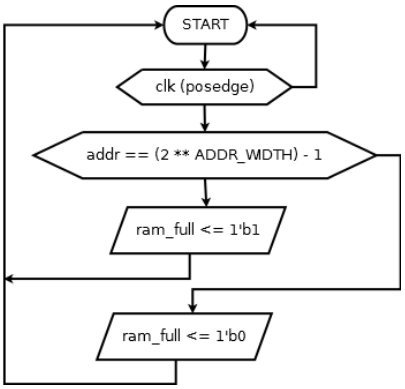

Figure 6: *Diagrama de flujo para la creación del módulo* ***flag_ram_full*** *parametrizado*

4 Resultados simulados.

A continuación, se presentará las imágenes de los resultados obtenidos en la simulación. Primero mostraremos la inicialización de cada una de las 4 fases, para posteriormente mostrar las salidas con diferentes valores de tuning.

4.1 Estado de inicialización de la memoría RAM

En la figura 7 podemos apreciar como mediante la señal de reset inicializamos nuestra maquina de estados finito mantiéndose en el estado IDLE mientras no le asignemos un valor a la variable instruct, posteriormente en el primer ciclo de reloj de asignamos a la variable instruct el valor de 1 para entrar en el estado RAM_INIT, en el tercer ciclo de reloj podemos apreciar que se comienzan a tener valores para los datos de entrada y sus respectivas locaciones de memoria, como podemos apreciar en la figura 8, los valores concuerdan con los primeros 11 valores del archivo de texto que contiene las magnitudes que se guardaran en la memoria, para después una vez inicializada la memoria.

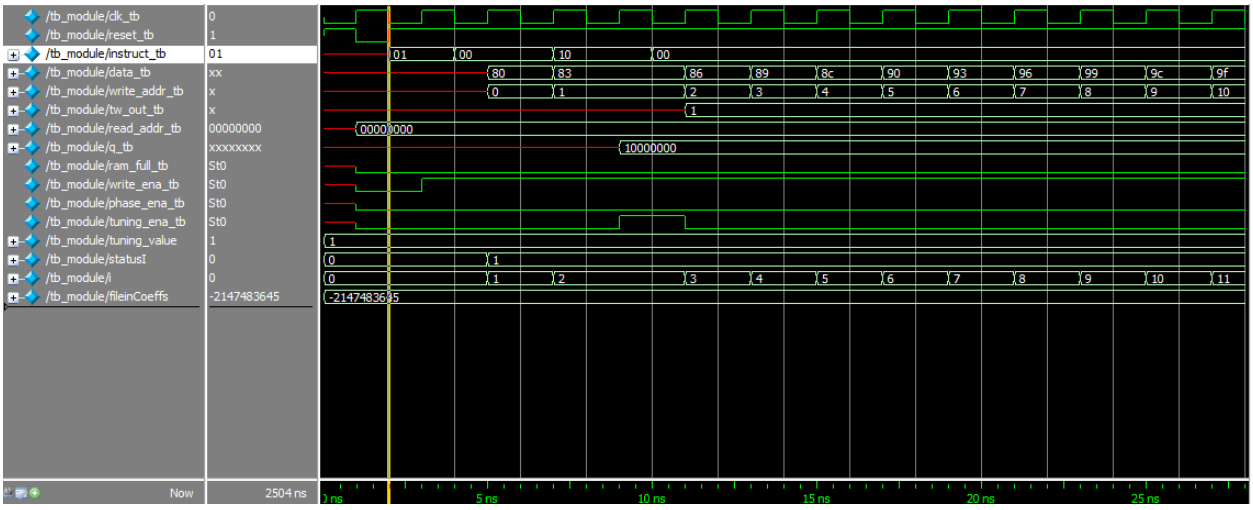

Figure 7: *Fase de inicialización de la memoria ram junto con las señales que permiten la inicialización de este proceso*

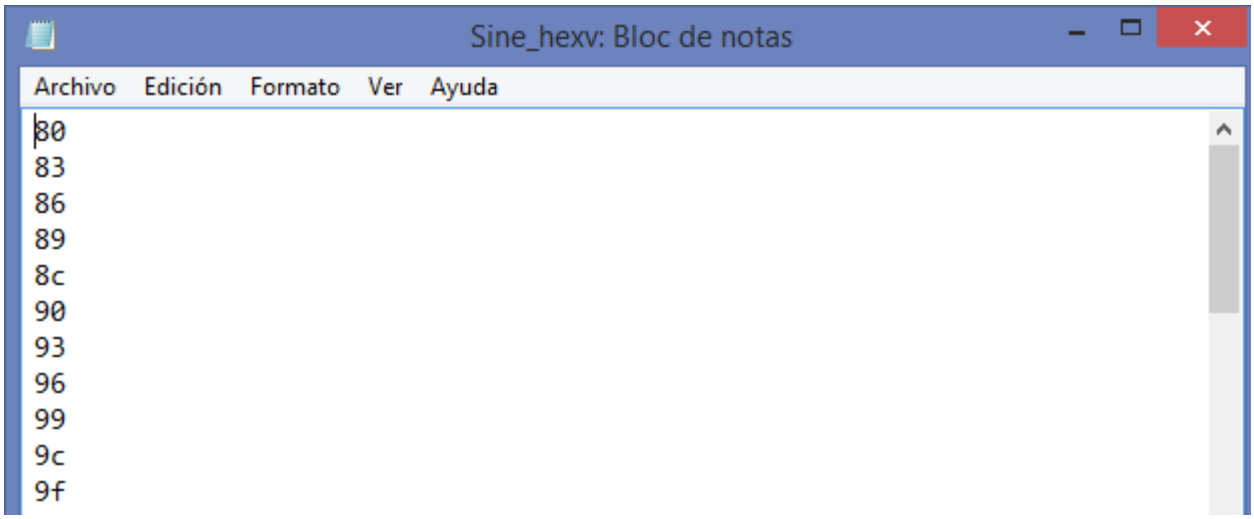

Figure 8: *Los primeros 11 valores que seran guardados en la memoria RAM*

4.2 Estado de tuning word

En la figura 9 localizándonos en el estado RAM_INIT, en el ciclo 3 de reloj asignamos el valor 2 a la variable instruct lo cual nos permite entrar en el estado TUNING_INIT, y asociar el valor de la dirección de memoria del siguiente ciclo de reloj al registro del tuning, esta asignación sucede después del ciclo 4 de la señal de reloj.

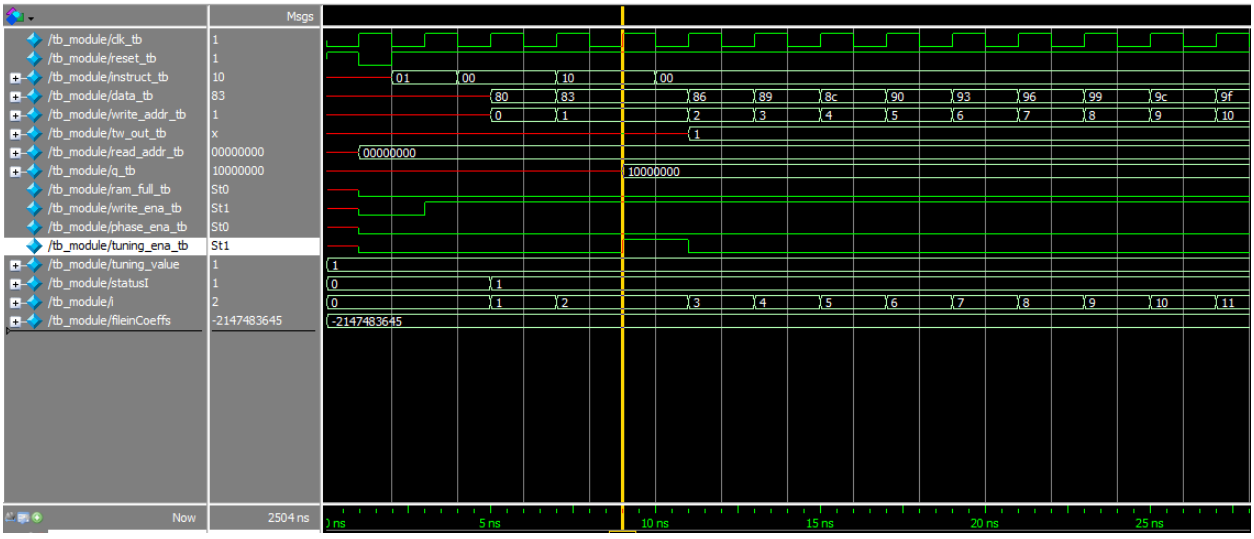

Figure 9: *Fase de inicialización del tuning word, asi como la señal que permite el inicio del proceso*

4.3 Estado incialización del DDS

En la figura 10 podemos observar como entramos en el estado DDS_RUNNING mediante el valor de 3 asignado a la variable instruct una vez que la bandera de ram llena se encuentra en alto. Podemos apreciar como la bandera de ram_full sube mientras la señal de escritura se coloca en cero, esto como resultado de entrar en el estado DDS_RUNING donde la señal para el modulo generador de fase se coloca en alto y entonces la memoria ram comienza a entregar sus valores almacenados en sus locaciones. En la figura 11

podemos ver como al colocar la señal de salida en formato analógico podemos observar la tendencia que tiene y como se comporta como una señal cíclica de tipo senoidal. En la figura 12 podemos ver como el generador el entregar la última locación de memoria esta este conteo se reinicia y la función resultante de la ram vuelve a mostrarse de forma cíclica.

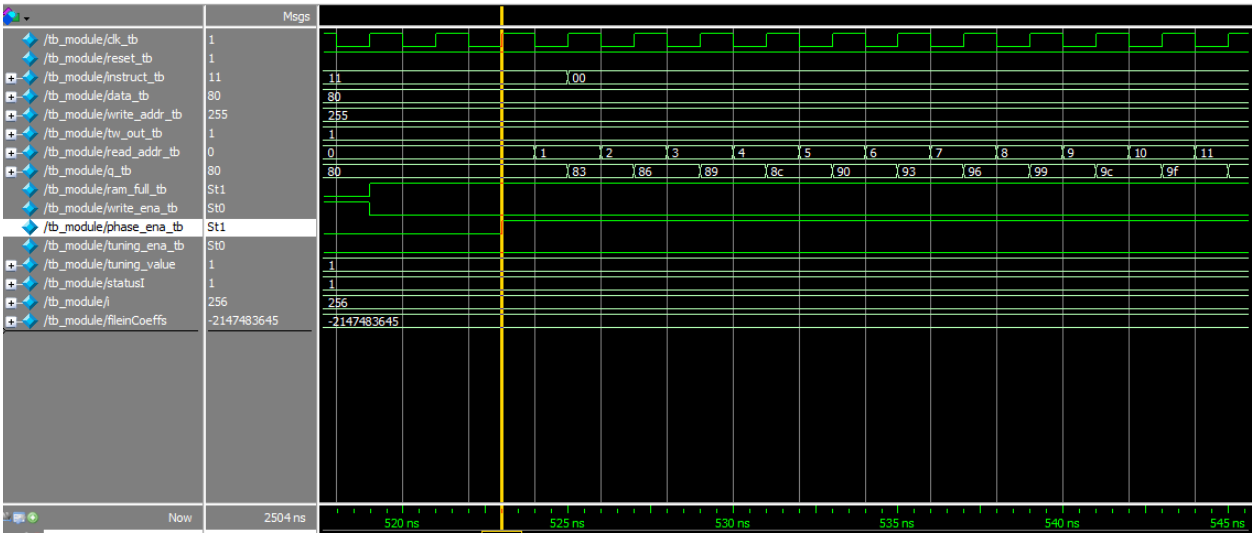

Figure 10: *Fase del generador DDS, donde la memoria ram comenzara a entragar los valores en las locaciones de memeoria proporcioadas por el generador de fase, al igual que las señales para inciar el proceso*

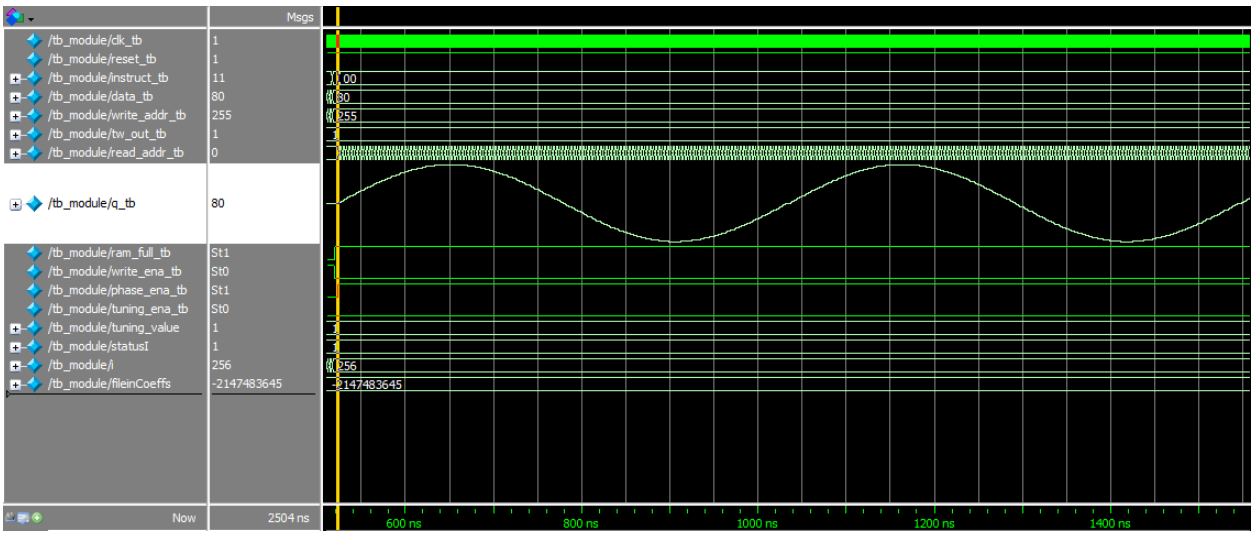

Figure 11: *Figura que muestra los valores en formato análogico de los valores de la salida de la memoria RAM en la fase de DDS_RUNNING*

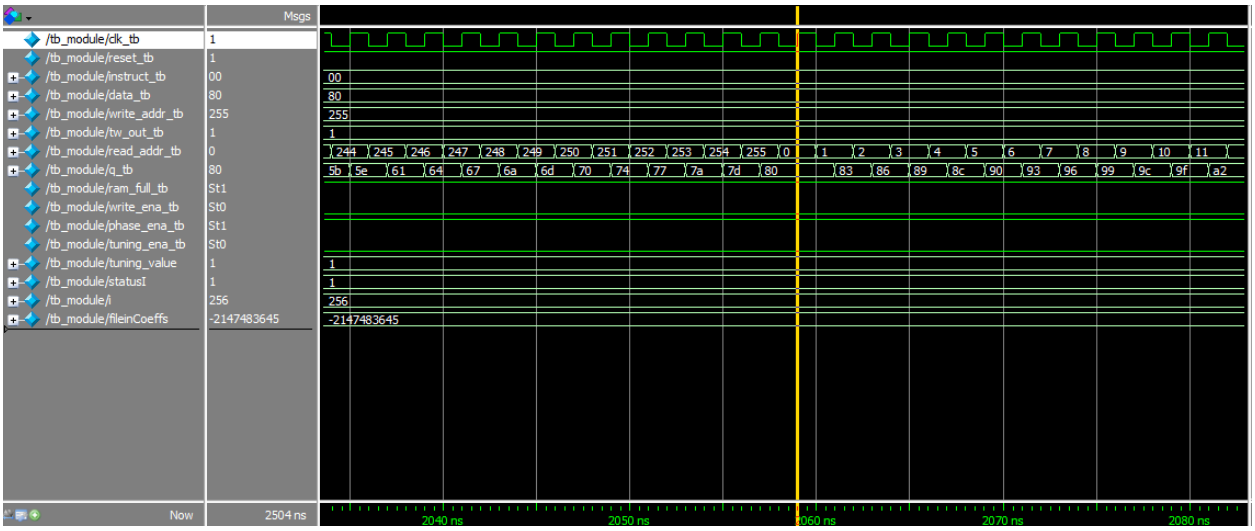

Figure 12: *Figura que muestro como al llegar a la ultima locación de memoria de la ram esta se reinicia y el ciclo vuelve a suceder*

4.4 Salidas del DDS con diferentes valores del tuning word

En las siguientes imágenes mostramos como se muestra la señal de salida del DDS con diferentes valores del tuning, esto permitiendo que la frecuencia se incremente conforme incrementamos el valor del tuning.

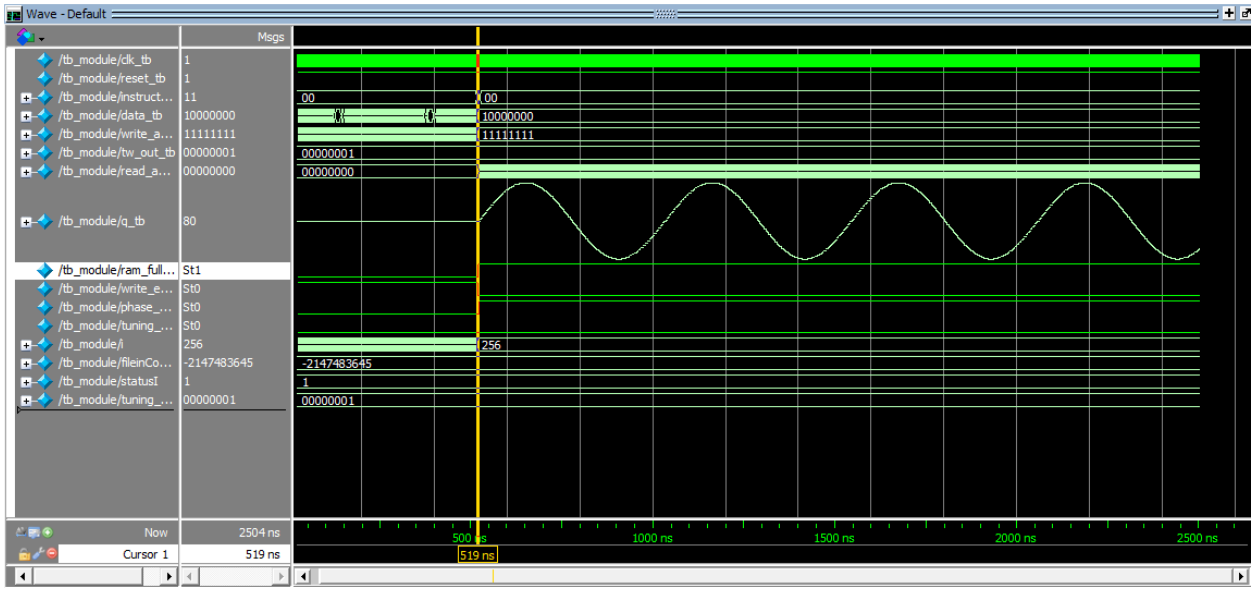

Figure 13: *Figura que muestra la salida de la ram en la fase de DDS_RUNNING con un valor de tuning word de 1*

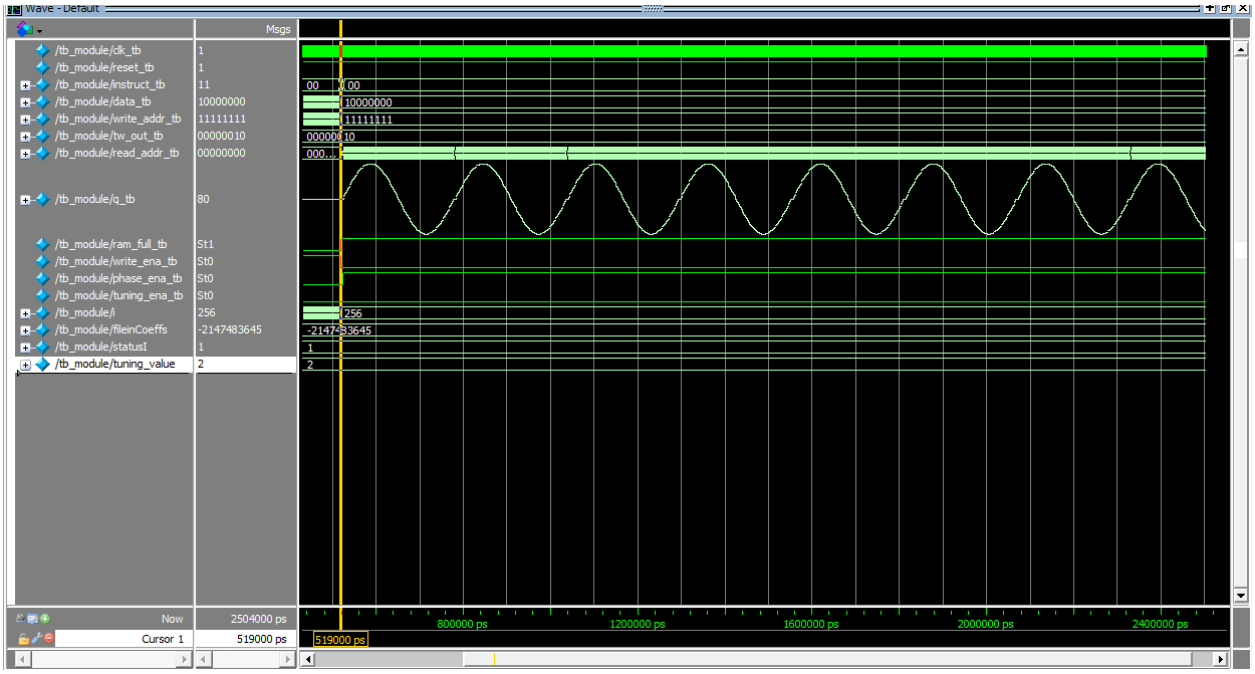

Figure 14: *Figura que muestra la salida de la ram en la fase de DDS_RUNNING con un valor de tuning word de 2*

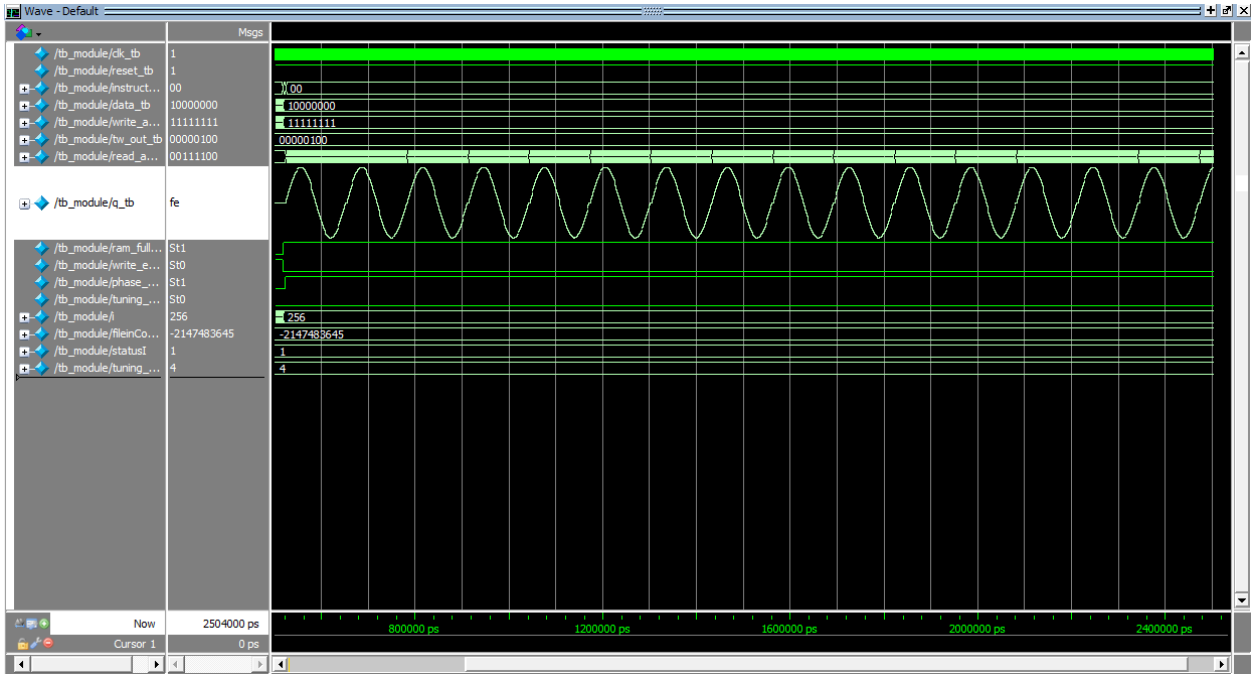

Figure 15: *Figura que muestra la salida de la ram en la fase de DDS_RUNNING con un valor de tuning word de 4*

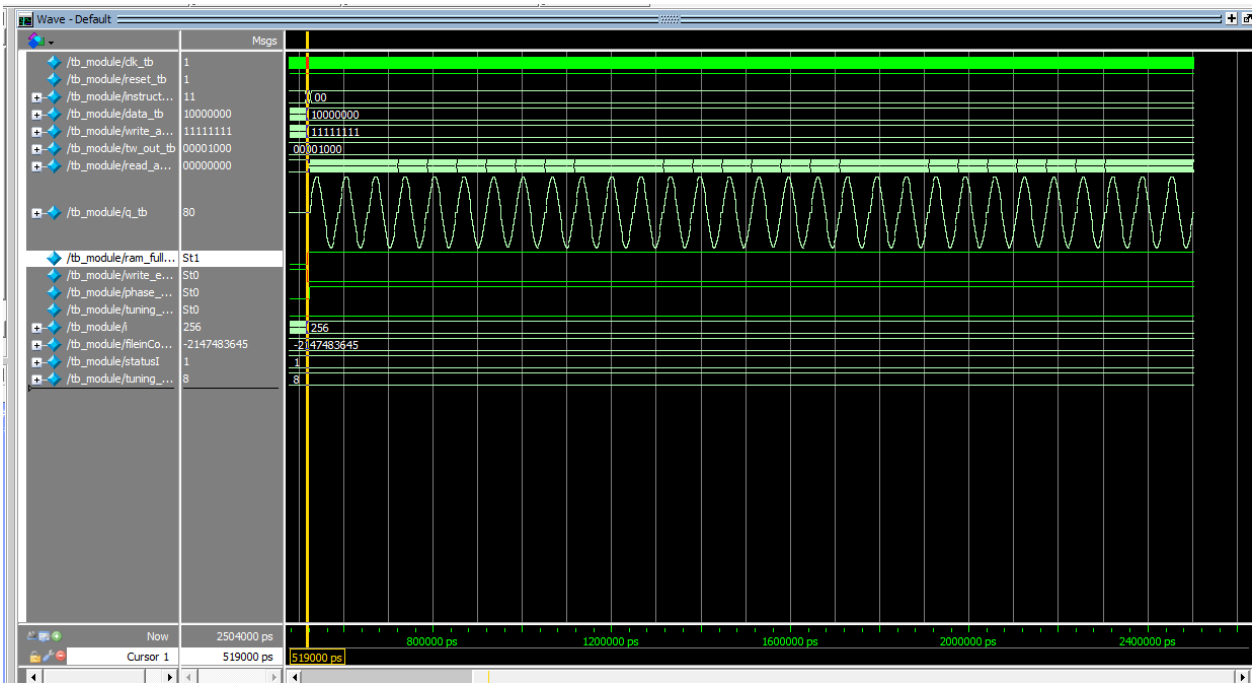

Figure 16: *Figura que muestra la salida de la ram en la fase de DDS_RUNNING con un valor de tuning word de 8*

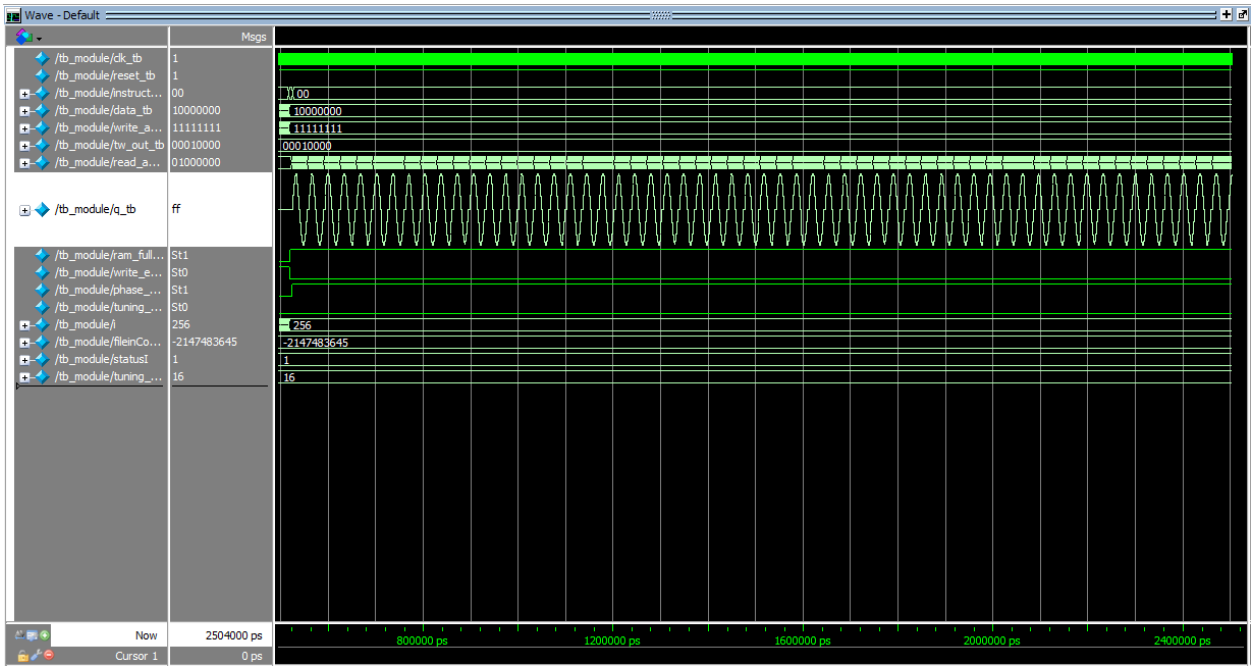

Figure 17: *Figura que muestra la salida de la ram en la fase de DDS_RUNNING con un valor de tuning word de 16*

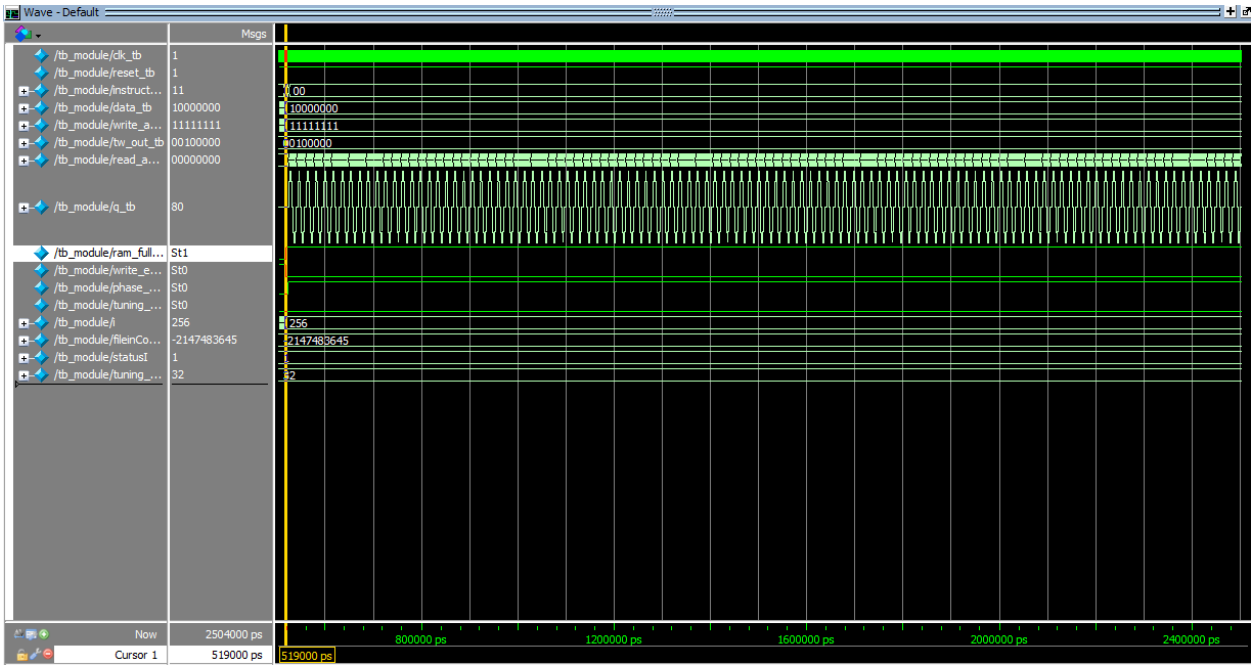

Figure 18: *Figura que muestra la salida de la ram en la fase de DDS_RUNNING con un valor de tuning word de 32*

4.5 Valores introducidos en la cama de pruebas.

Tendremos dos bloques de procesos initial que se estarán ejecutando de forma concurrente, en el primer initial utilizaremos la function task $fopen() para poder abrir el archivo .txt que contiene los valores a inicializar en la memoria ram, en el instante 0 inicializamos el reloj clk en bajo, y la señal de reset en alto, esperamos una unidad de tiempo y colocamos la señal de reset en bajo una unidad de tiempo para posteriormente colocarle en alto, esto resultara en el evento necesario para inicializar la maquina de estados en el esto IDLE, después la señal instruct le damos el valor de 1 como señal para cambiar al estado RAM_INIT, esperamos 2 unidades de tiempo y le asociamos a esta señal el valor de 0, esperamos 5000 unidades de tiempo y detenemos la simulación.

En el segundo proceso initial, esperamos hasta que el valor de instruct sea 1, y esperamos hasta un ciclo de reloj negativo, dentro de un ciclo while leeremos cada valor y lo asociaremos a sus respectivas variables que serán dirigidas a la ram, una vez habiendo terminado de leer todos los datos se cerrara el archivo. Dentro del while tenemos un condicional donde si el valor de la locación de memoria concuerda con nuestro valor de interés, introduciremos la señal instruct con el valor 2 para poder tomar ese valor de dirección como tuning Word. Fuera del ciclo while colocamos el valor de instruct en 2 y esperamos 2 unidades de tiempo antes de poner dicha variable con valor a 0.