



PRÁCTICA 2

Arquitectura de Computadoras

Grupo: 5

Integrantes:

Jaime González Oscar

Martínez Jarquín Ricardo Eduardo

Salgado Salazar Carlos Eduardo

7 de Marzo de 2017



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
FACULTAD DE INGENIERÍA
DIVISIÓN DE INGENIERÍA ELÉCTRICA



- **Introducción:**

Cartas ASM

Las cartas ASM (Algorithmic State Machine) son formas de descripción de tipo gráfico especialmente enfocadas, como indica su nombre, a representar algoritmos secuenciales fue creado por Chris Clare.

Este algoritmo especifica mediante un diagrama de flujo los pasos del procedimiento y los caminos de decisión. Al ser un diagrama de flujo para un algoritmo hardware debe tener unas características especiales que ligen de cerca el desarrollo hardware del algoritmo.

Una carta ASM contiene necesariamente una entrada de comienzo, solo uno o varios bloque ASM junto con un reloj que controle su paso por él, cada uno de los bloques con una caja de estado, siempre que forme un grafo cerrado, otros pueden contener cajas de decisión con cajas de acción condicional y otros pueden contener cajas de decisión sin cajas de acción condicional.

Características

- El desarrollo de macro operaciones en micro operaciones es un proceso algorítmico secuencial, por lo que la descripción de Sistemas Digitales a nivel RT cae plenamente dentro de la materia representada con cartas ASM.
- Es una herramienta que da información sobre la estructura y sobre el comportamiento dinámico del sistema que se describe con ella, aspectos ambos de sumo interés.
- La carta ASM proporciona información tanto del algoritmo con los datos como de la secuencia de control, por lo que la propia herramienta está muy próxima a las implementaciones hardware de las Unidades de Datos y de Control.
- Se trata de una herramienta muy intuitiva, fácil de aprender y muy adecuada para trabajar a mano.
- La herramienta tiende un doble puente: 1) hacia niveles de abstracción más bajos, en concreto con los modelos de máquinas de estado que son tan útiles a nivel de conmutación; y 2) hacia niveles más abstractos, como con la representación mediante grafos de flujo de programas a nivel ISP.



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
FACULTAD DE INGENIERÍA
DIVISIÓN DE INGENIERÍA ELÉCTRICA



- **Objetivo:**

Familiarizar al alumno en el conocimiento de los algoritmos de las máquinas de estados utilizando el lenguaje VHDL.

- **Desarrollo:**

Para la realización de la práctica se nos pidió realizar las siguientes tareas.

Paso 1

“Haga un proyecto nuevo en el sistema de desarrollo Quartus en el cual se compile el código anterior.”

Para esto abrimos el programa Quartus Prime Lite en el cual seleccionamos la pestaña FILE -> New Project Wizard lo cual nos desplego una nueva ventana en la cual seleccionamos la ubicación para guardar nuestro proyecto, así como el nombre que en este caso será practica2.

ing Tools Window Help

New Project Wizard

Directory, Name, Top-Level Entity

What is the working directory for this project?
C:/Users/RicardoEduardo/Desktop/10mo semestre/Arqui de compus/Practicas/Practica 2

What is the name of this project?
practica2

What is the name of the top-level design entity for this project? This name is case sensitive and must exactly match the entity name in the design file.
practica2

Use Existing Project Settings...

Seleccionamos la opción de proyecto vacío y escogimos la Familia del procesador de nuestra tarjeta **Cyclon IV E**, así como el modelo que en este caso es **EP4CE22F17C6N**.

New Project Wizard

Family & Device Settings

Select the family and device you want to target for compilation.
You can install additional device support with the Install Devices command on the Tools menu.
To determine the version of the Quartus Prime software in which your target device is supported, refer to the [Device Support List](#) webpage.

Device family
Family: Cyclon IV E
Devices: All

Target device
☐ Auto device selected by the Filter
☒ Specific device selected in 'Available devices' list
☐ Other: n/a

Show in 'Available devices' list
Package: Any
Pin count: Any
Core Speed grade: Any
Name filter:
☒ Show advanced devices

Available devices:

Name	Core Voltage	LEs	Total I/Os	GPIOs	Memory Bits	Embedded multiplier 9-bit elements
EP4CE22E22C9L	1.0V	22320	80	80	608256	132
EP4CE22E22F7	1.2V	22320	80	80	608256	132
EP4CE22E22B8L	1.0V	22320	80	80	608256	132
EP4CE22F17A7	1.2V	22320	154	154	608256	132
EP4CE22F17C6	1.2V	22320	154	154	608256	132
EP4CE22F17C7	1.2V	22320	154	154	608256	132
EP4CE22F17C8	1.2V	22320	154	154	608256	132
EP4CE22F17C8L	1.0V	22320	154	154	608256	132

< Back Next > Finish Cancel Help



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
FACULTAD DE INGENIERÍA
DIVISIÓN DE INGENIERÍA ELÉCTRICA



- Carta ASM

Para implementar el esquemático de la carta ASM que viene en la práctica tuvimos que crear un archivo para el código VHDL y seguimos los pasos: File -> New -> VHDL una vez creado pegamos el código del algoritmo de la carta ASM que venía en la práctica y guardamos el archivo como “practica2”.

Revisamos la sintaxis mediante Processing ->Analyze Current File. Y verificamos que no había errores en el código.

```
1 library IEEE;
2 use IEEE.STD_LOGIC_1164.ALL;
3 use IEEE.STD_LOGIC_ARITH.ALL;
4 use IEEE.STD_LOGIC_UNSIGNED.ALL;
5
6 entity carta_asm_2 is
7
8   Port ( RELOJ : in STD_LOGIC;
9
10     RESET : in STD_LOGIC;
11     S : in std_logic_vector (1 downto 0); --DONDE EL BIT MAS SIGNIFICATIVO ES SI Y EL MENOS EL SD
12     atras : out STD_LOGIC;
13     adelante : out STD_LOGIC;
14     giro_der : out STD_LOGIC;
15     giro_izq : out STD_LOGIC;
16     out_epresente :out std_logic_vector (3 downto 0));
17
18 end carta_asm_2;
19
20 architecture Behavioral of carta_asm_2 is
21
22   signal esiguiente : std_logic_vector (3 downto 0) := B"0000";
23
24   constant s0 : std_logic_vector(3 downto 0) := X"0";
25   constant s1 : std_logic_vector(3 downto 0) := X"1";
26   constant s2 : std_logic_vector(3 downto 0) := X"2";
27   constant s3 : std_logic_vector(3 downto 0) := X"3";
```

Running Quartus Prime Create Symbol File
Command: quartus_map --read_settings_files=on --write_settings_files=off practica2 -c practica2 --generate_symbol="c:/Users/RicardoEduardo/Desktop/10mo s
Quartus Prime Create Symbol File was successful. 0 errors, 0 warnings

Paso 2.

“En Quartus haga un símbolo de esta máquina de estados”

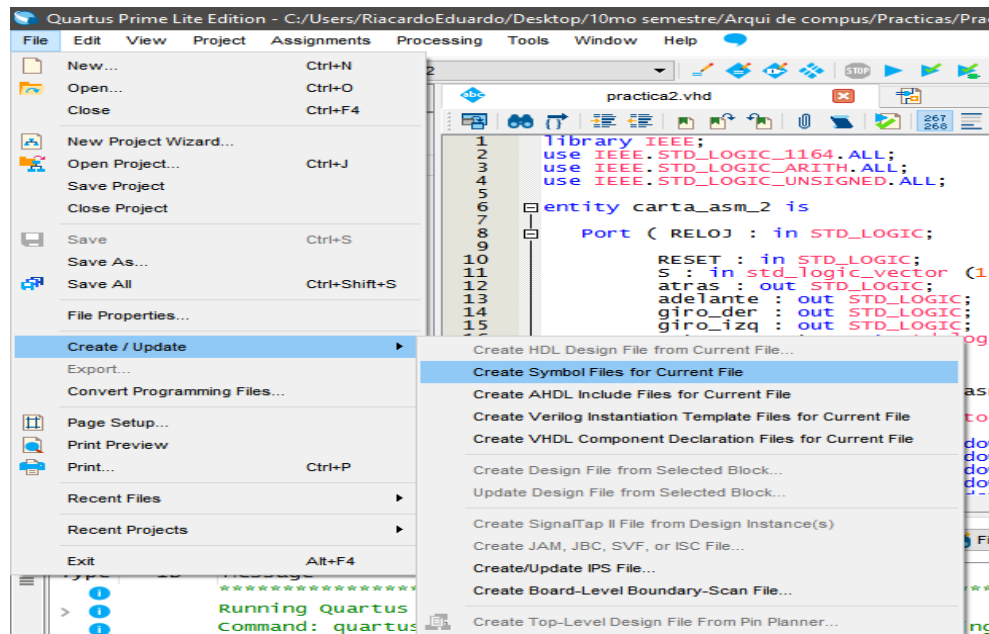
Una vez creado el proyecto en el navegador de proyecto seleccionamos nuestra practica2 y creamos un diagrama esquemático seleccionando File -> new -> Block Diagram/ Schematic, el cual guardamos con el nombre “Carta.bdf”



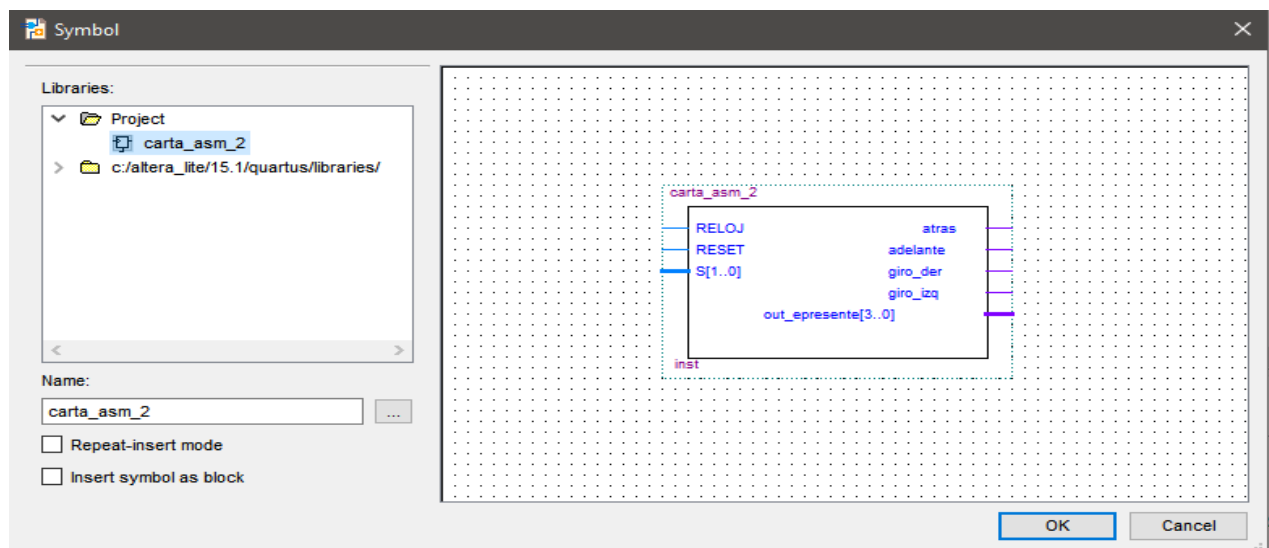
UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
FACULTAD DE INGENIERÍA
DIVISIÓN DE INGENIERÍA ELÉCTRICA



Para crear el símbolo esquemático de nuestra carta ASM seleccionamos la ventana de Carta.vhd y después seleccionamos la pestaña File -> Create/ Update -> Create Symbol files for current file.

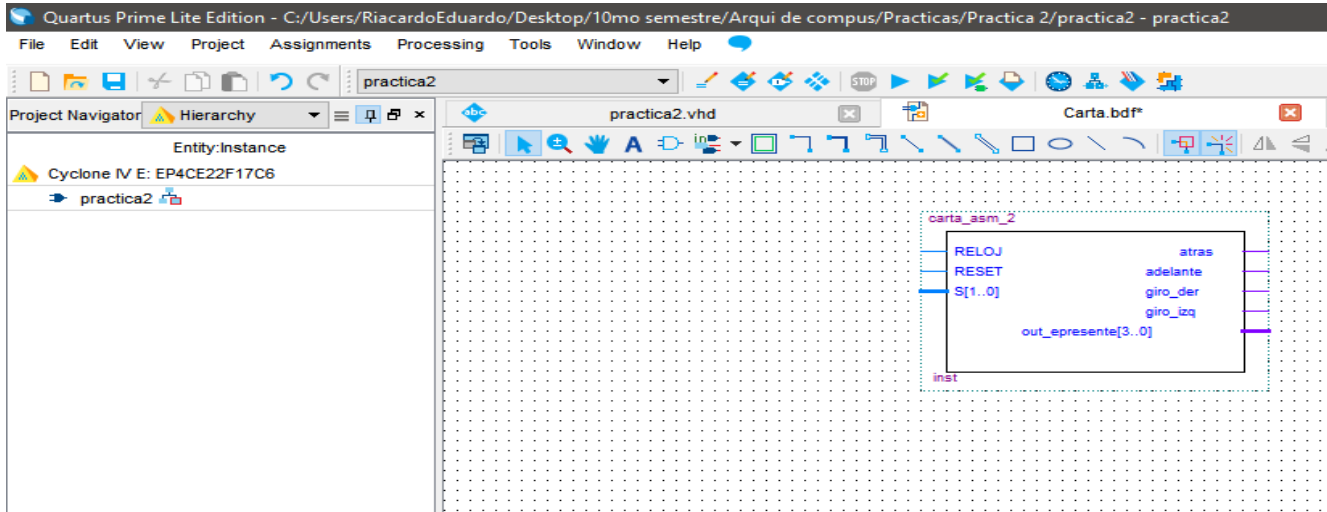


Para inserta el símbolo en nuestro esquemático dimos click derecho insert -> symbol, expandimos la carpeta “Project” y seleccionamos la opción “carta_asm_2”.





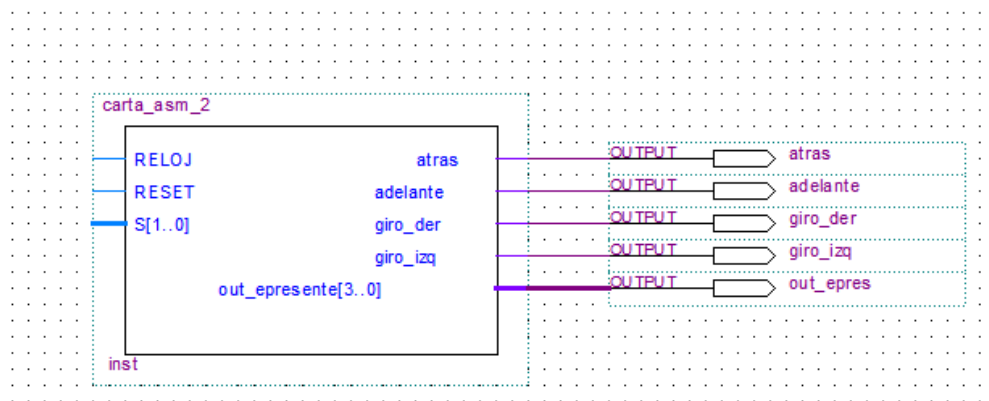
UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
FACULTAD DE INGENIERÍA
DIVISIÓN DE INGENIERÍA ELÉCTRICA



Paso 3

Programe la tarjeta para que muestre, usando cuatro leds, el estado presente, así mismo muestre también las salidas: adelante, atrás, giro_izq y giro_der.

Para utilizar los leds primero necesitamos colocarlos en el esquemático con click en el botón derecho del ratón, seleccionamos insert -> symbol -> primitives -> pin -> output y los renombramos conforme a sus respectivas variables.



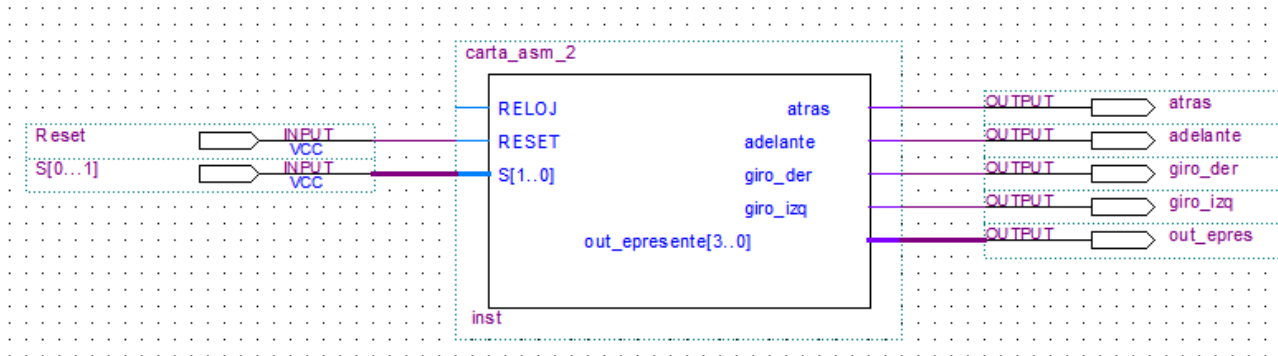
Paso 4

“Las entradas SI, SD y reset introdúzcalas usando los botones de la tarjeta”

Para la entrada se sigue el mismo procedimiento, seleccionamos insert -> symbol -> primitives -> pin -> input y los renombramos conforme a sus respectivas variables.



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
FACULTAD DE INGENIERÍA
DIVISIÓN DE INGENIERÍA ELÉCTRICA



Pasó 5

“Conecte al reloj maestro al divisor que diseño en la práctica anterior para poder visualizar mejor la operación de la máquina de estados”

Para realizar esta acción recurrimos a nuestro proyecto de la práctica 1 y copiamos el algoritmo del reloj, pasando a nuestro proyecto actual creamos un nuevo archivo de VHDL siguiendo los pasos: File -> New -> VHDL una vez creado pegamos el código del algoritmo del reloj y guardamos el archivo como “divider”.

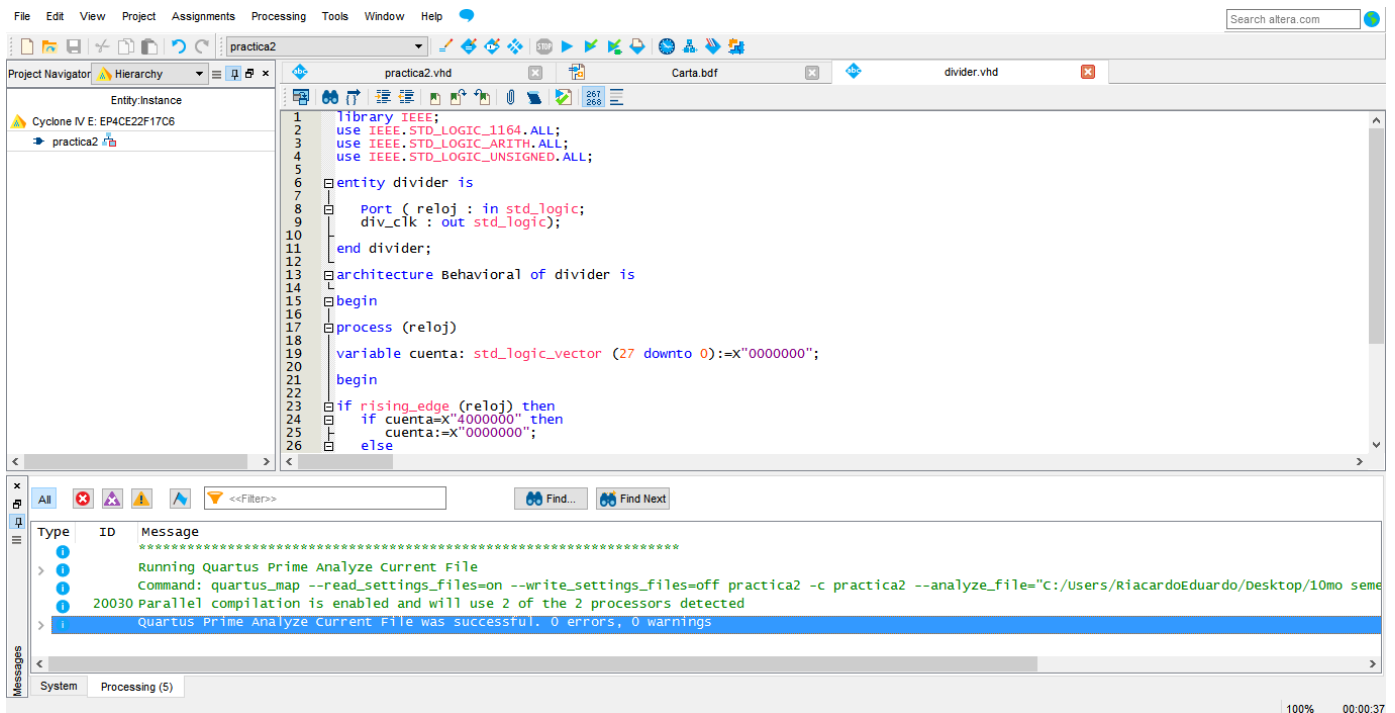
```
Quartus Prime Lite Edition - C:/Users/RicardoEduardo/Desktop/10mo semestre/Arqui de compus/Practicas/Practica 2/practica2 - practica2
File Edit View Project Assignments Processing Tools Window Help
practica2
practica2.vhd Carta.bdf divider.vhd
Entity:Instance
Cyclone IV E: EP4CE22F17C6
practica2
1 library IEEE;
2 use IEEE.STD_LOGIC_1164.ALL;
3 use IEEE.STD_LOGIC_ARITH.ALL;
4 use IEEE.STD_LOGIC_UNSIGNED.ALL;
5
6 entity divider is
7
8   port ( reloj : in std_logic;
9         div_clk : out std_logic);
10
11 end divider;
12
13 architecture Behavioral of divider is
14
15 begin
16
17 process (reloj)
18
19   variable cuenta: std_logic_vector (27 downto 0):=x"0000000";
20
21   begin
22
23     if rising_edge (reloj) then
24       if cuenta=x"4000000" then
25         cuenta:=x"0000000";
26       else
27         cuenta:= cuenta+1;
28       end if;
29     end if;
30   end if;
31 end process;
32
```



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
FACULTAD DE INGENIERÍA
DIVISIÓN DE INGENIERÍA ELÉCTRICA

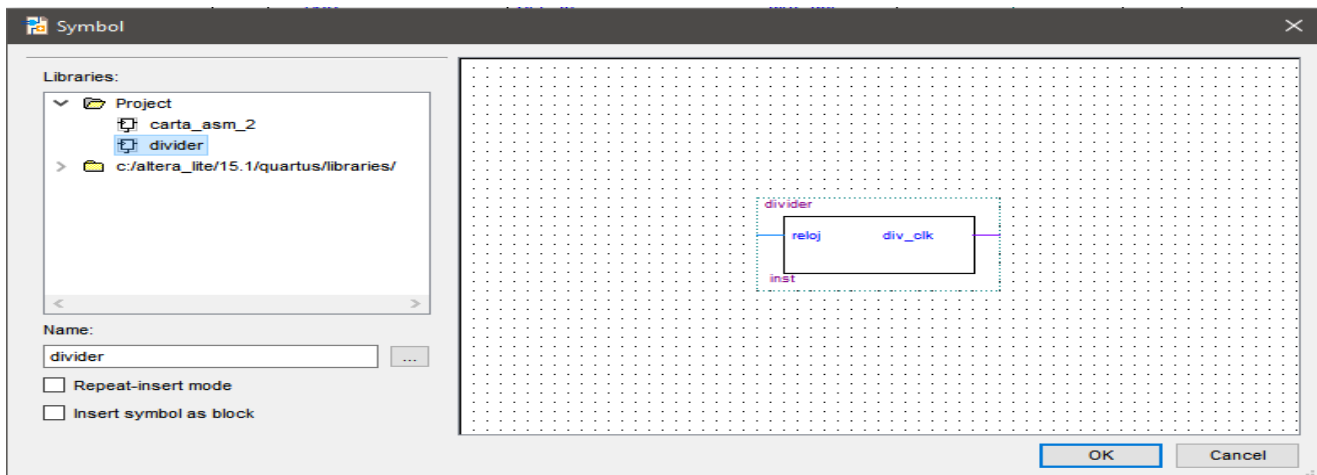


Revisamos la sintaxis mediante Processing ->Analyze Current File. Y verificamos que no había errores en el código.



Como en el caso de la carta ASM, para crear el esquemático del divisor y agregarlo a nuestro proyecto seleccionamos la pestaña File -> Create/ Update -> Create Symbol files for current file.

Después de crear el símbolo damos click derecho insert -> symbol, expandimos la carpeta “Project” y seleccionamos la opción “divider”.



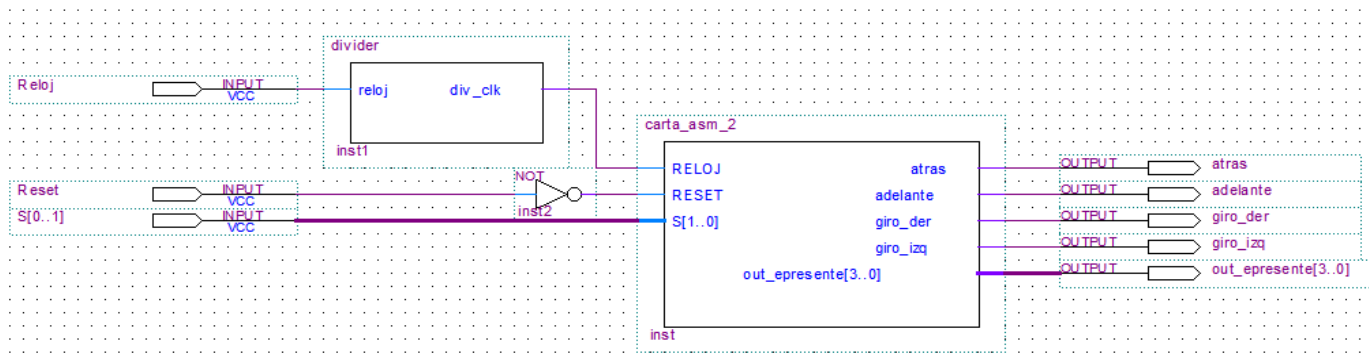


UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
FACULTAD DE INGENIERÍA
DIVISIÓN DE INGENIERÍA ELÉCTRICA



Lo agregamos y realizamos las conexiones según lo especificado en la imagen de muestra de nuestra práctica en donde nos indican que la salida del divisor de reloj debe conectarse a la entrada del reloj de nuestra máquina de estados.

Como aspecto a considerar, en la práctica se indica que se deben negar las salidas de los leds así como las de S pero como nuestra tarjeta trabaja con lógica positiva no es necesario realizar este paso aun que como vimos en la realización de la práctica pasada si es necesario negar las entrada del Reset para que no interrumpa el trabajo de la maquina hasta que sea accionado.



Guardamos nuestro trabajo y procedimos a compilar nuestro proyecto mediante Processing -> Start Compilation, en la cual no hubo ningún error y nos mostró una ventana con el resumen de las características de nuestro proyecto y una ventana de mensaje con las acciones realizadas.

Flow Summary

Flow Status	Successful - Sun Feb 26 18:12:42 2017
Quartus Prime Version	15.1.0 Build 185 10/21/2015 SJ Lite Edition
Revision Name	practica2
Top-level Entity Name	practica2
Family	Cyclone IV E
Device	EP4CE22F17C6
Timing Models	Final
Total logic elements	59 / 22,320 (< 1 %)
Total combinational functions	55 / 22,320 (< 1 %)
Dedicated logic registers	40 / 22,320 (< 1 %)
Total registers	40
Total pins	12 / 154 (8 %)
Total virtual pins	0
Total memory bits	0 / 608,256 (0 %)
Embedded Multiplier 9-bit elements	0 / 132 (0 %)
Total PLLs	0 / 4 (0 %)

Messages

Type	ID	Message
Information	204019	generated file practica2.vho in folder "C:/Users/RicardoEduardo/Desktop/10mo semestre/Arqui de compus/Practicas/Practica 2/simulation/modelsim/" for E
Information	204019	generated file practica2_6_1200mv_85c_vhd_slow.sdo in folder "C:/Users/RicardoEduardo/Desktop/10mo semestre/Arqui de compus/Practicas/Practica 2/simulat
Information	204019	generated file practica2_6_1200mv_0c_vhd_slow.sdo in folder "C:/Users/RicardoEduardo/Desktop/10mo semestre/Arqui de compus/Practicas/Practica 2/simulat
Information	204019	generated file practica2_min_1200mv_0c_vhd_fast.sdo in folder "C:/Users/RicardoEduardo/Desktop/10mo semestre/Arqui de compus/Practicas/Practica 2/simulat
Information	204019	generated file practica2_vhd.sdo in folder "C:/Users/RicardoEduardo/Desktop/10mo semestre/Arqui de compus/Practicas/Practica 2/simulation/modelsim/" f
Information	293000	Quartus Prime EDA Netlist writer was successful. 0 errors, 0 warnings
Information	293000	Quartus Prime Full Compilation was successful. 0 errors, 8 warnings



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
FACULTAD DE INGENIERÍA
DIVISIÓN DE INGENIERÍA ELÉCTRICA



Pasó 6

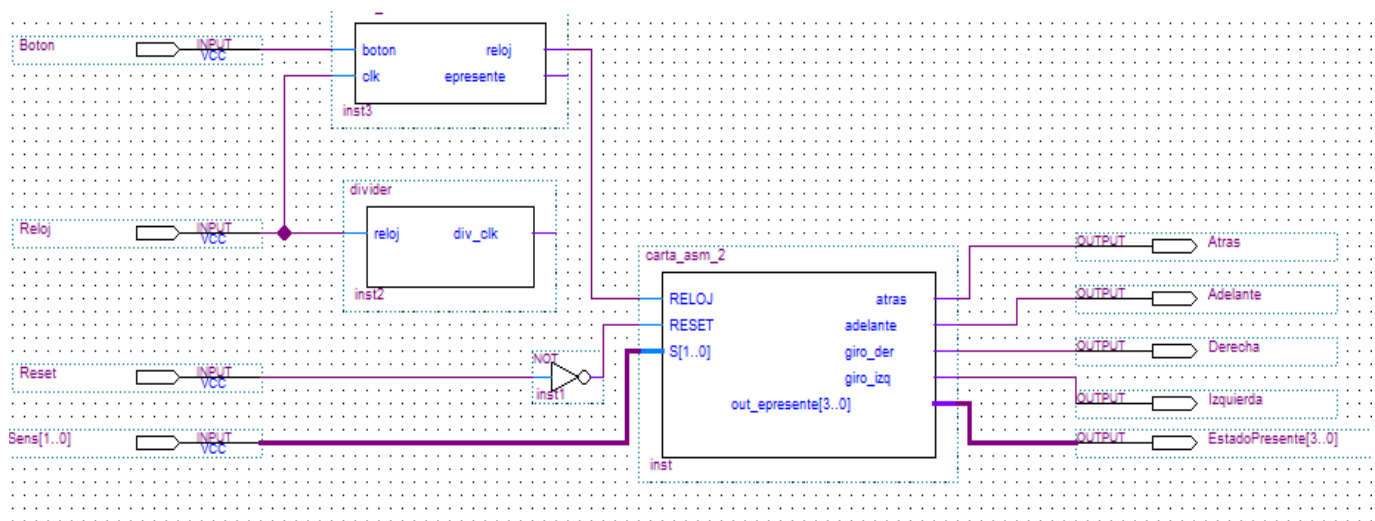
Siguiente paso nos pide la practica incluir en nuestro proyecto otra máquina de estados en la cual a partir de una señal de reloj externa de alta frecuencia y la de un botón de entrada sirve para censar cuando este es oprimido y liberado, enviando una señal que permite ver el desempeño de otra máquina de estados conectado a el paso a paso.

Por lo que se procedió a agregar el nuevo archivo en vhdl y copiar el código de la práctica y realizar la sintaxis mediante Processing ->Analyze Current File. Y verificamos que no había errores en el código.

Una vez creado el modulo lo esquinzamos mediante la pestaña File -> Create/ Update -> Create Symbol files for current file.

Después de crear el símbolo damos click derecho insert -> symbol, expandimos la carpeta “Project” y seleccionamos la opción “sensa_boton”.

Y realizamos las conexiones conforme a la imagen que presentada en la práctica.



Guardamos nuestro trabajo y procedimos a compilar nuestro proyecto mediante Processing -> Start Compilation, en la cual no hubo ningún error y nos mostró una ventana con el resumen de las características de nuestro proyecto y una ventana de mensaje con las acciones realizadas.

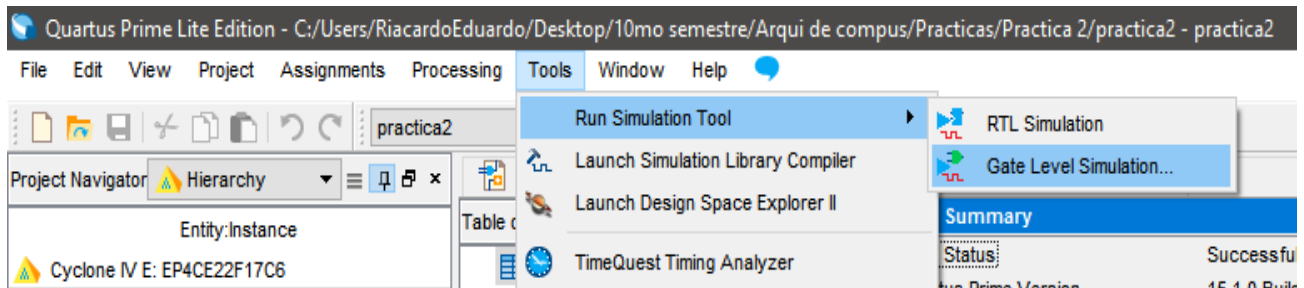


UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
FACULTAD DE INGENIERÍA
DIVISIÓN DE INGENIERÍA ELÉCTRICA

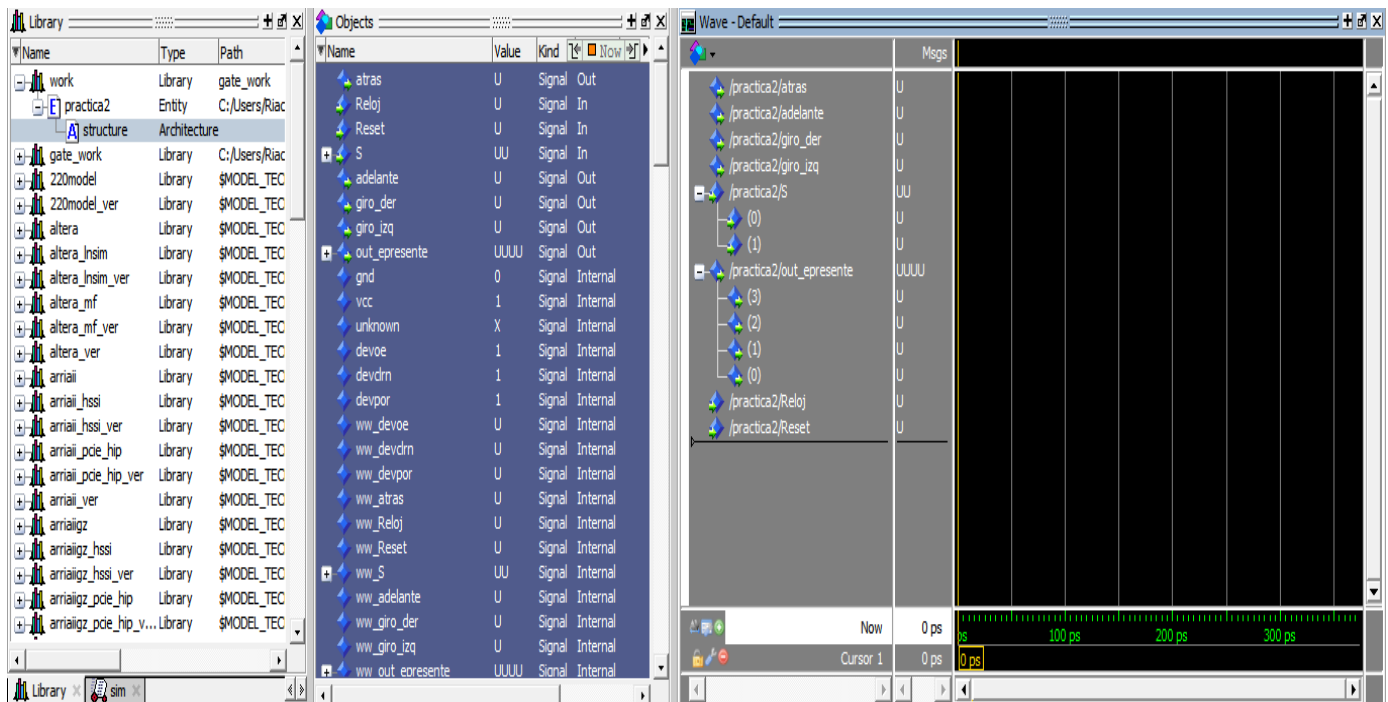


- Simulación:

Siempre que nuestro proyecto compila exitosamente es buena idea simularlo para verificar el comportamiento de nuestro diseño y verificar si es lo que se espera o si algo está mal diseñado, para esto damos click en Tools > Run EDA Simulation Tool > EDA Gate Level Simulation



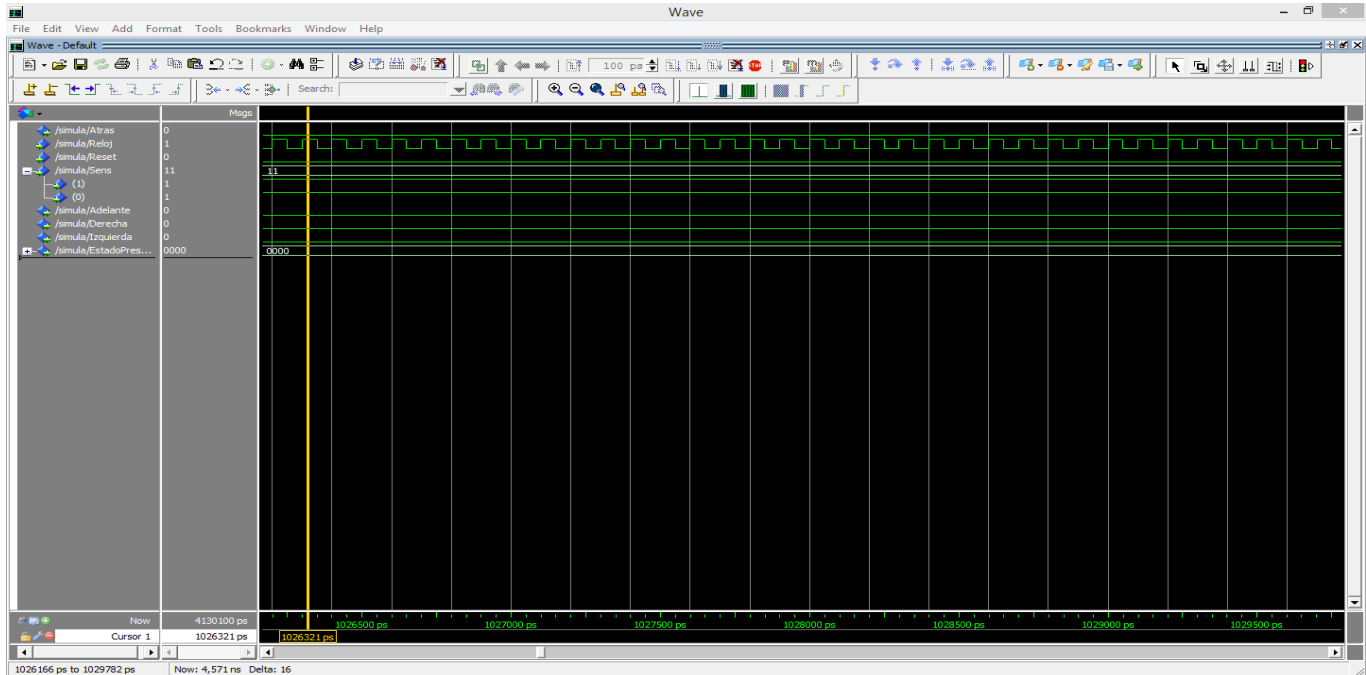
Una vez desplegada la ventana de trabajo, seleccionamos la opción “work” para poder visualizar las variables en la ventana de objetos, arrastramos nuestras variables a la ventana de simulación.



Renombramos la variable reloj y seleccionamos clock, seleccionamos OK. Después activamos la simulación con Simulate -> Run

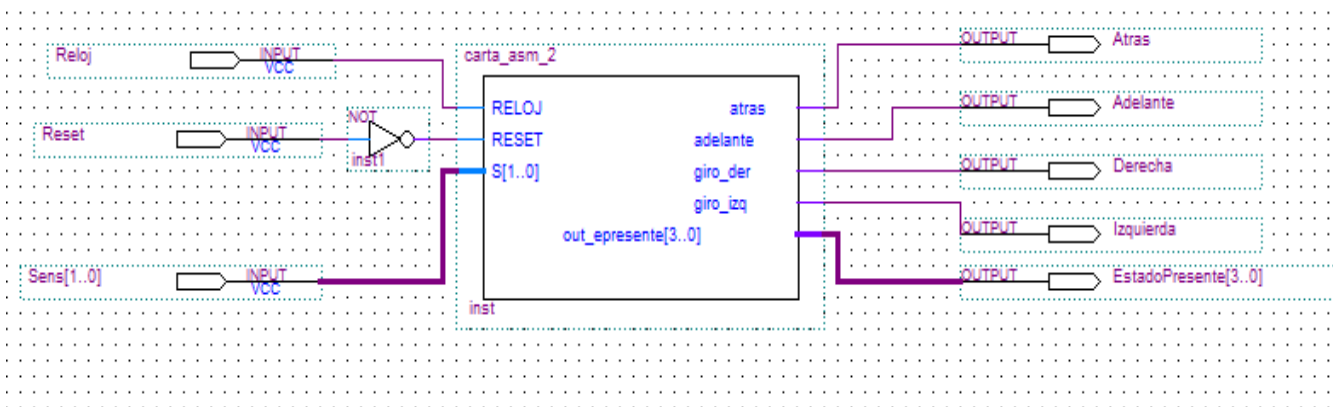


UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
FACULTAD DE INGENIERÍA
DIVISIÓN DE INGENIERÍA ELÉCTRICA



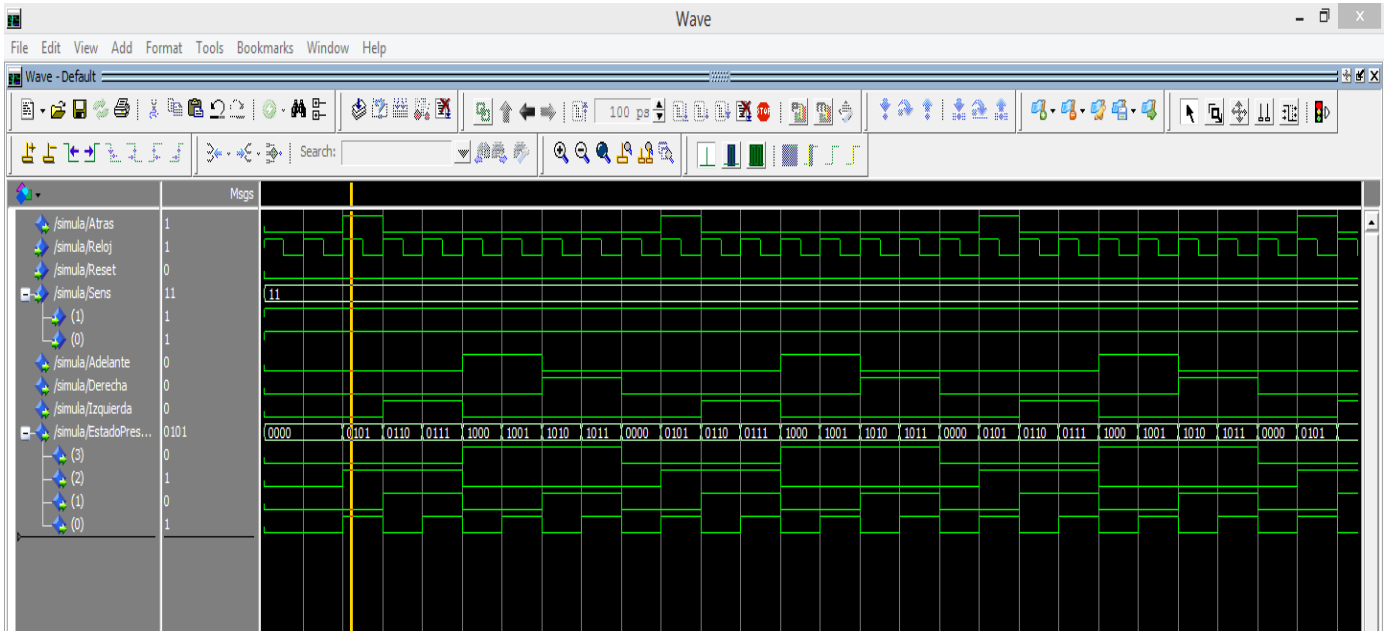
En esta simulación no pudimos observar los cambios que efectúa nuestra máquina de estados ya que al estar presente el divisor del reloj en nuestro esquemático la escala en el tiempo no permitía ver el comportamiento completo del sistema esto es debido a que se llena la memoria asignada al proyecto.

Para poder visualizar la simulación de forma correcta realizamos varias pruebas en las cuales se quitó el modulo del divisor del reloj de la entrada del reloj y se procedió a simular nuestro proyecto.



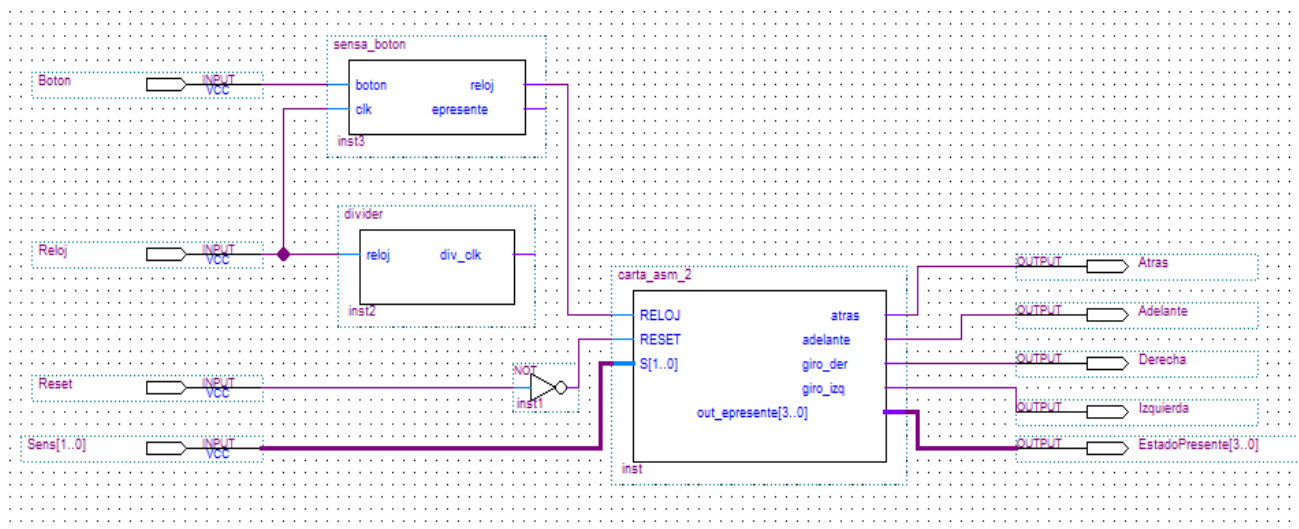


UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
FACULTAD DE INGENIERÍA
DIVISIÓN DE INGENIERÍA ELÉCTRICA



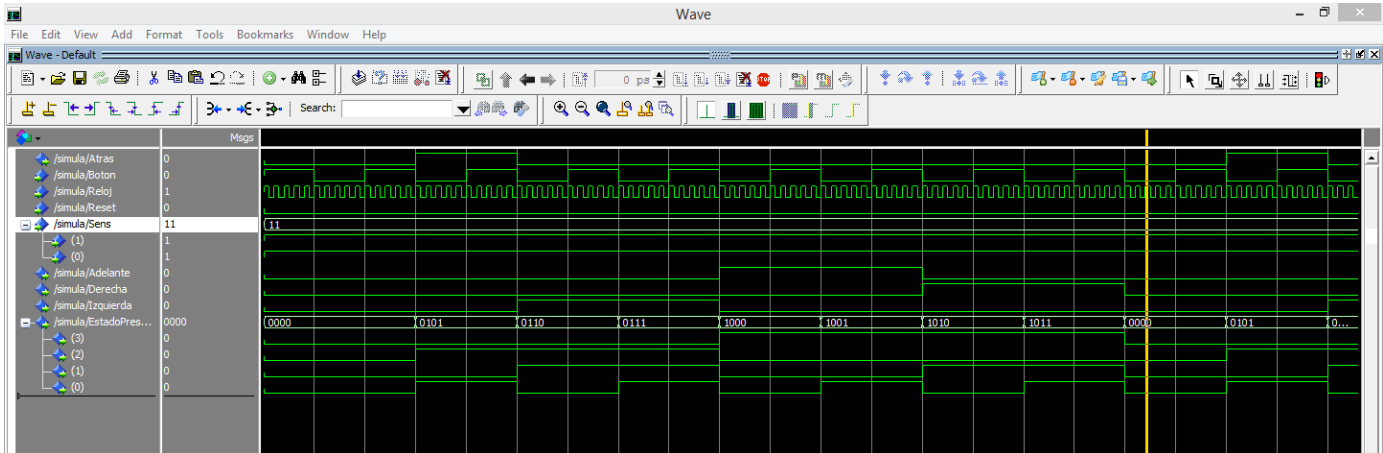
Como podemos ver la escala en tiempo se muestra en nanosegundos lo que nos permite ver el comportamiento de nuestra maquina la cual de acuerdo al impulso del reloj va cambiando los estados dependiendo de la entrada que se le suministre, así mismo se puede apreciar en la gráfica como va cambiando el estado presente dependiendo de la acción que realiza nuestra máquina, siendo este el comportamiento esperado.

También se realizó una simulación de la máquina de estado con el sensor de botón en la cual se variaron los valores del **reset = 0**, **S0= 1** y **S1= 1**.





UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
FACULTAD DE INGENIERÍA
DIVISIÓN DE INGENIERÍA ELÉCTRICA



En esta simulación pudimos observar como al introducir los valores del reset y de las entradas de sensor se reproducía el cambio en los estados siguientes y las salidas eran encendidas y pagadas conforme a los impulsos de reloj transcurrían, esta simulación también nos mostró los valores esperados de acuerdo a la carta ASM que se implementó.

Pasó 7

Programa la tarjeta para que muestre, usando cuatro leds, el estado presente, así mismo muestre también las salidas: adelante, atras, giro_izq y giro_der. Las entradas SI, SD y reset introdúzcalas usando los botones de la tarjeta.

Para la asignación de pines en nuestro proyecto seleccionamos Assignments -> Pin Planner lo cual nos despliega una nueva ventana.

- Para la salida del estado presente se le asignaran cuatro leds de la tarjeta siendo el valor menos significativo el led 7 y el más significativo el led 4.
- Para las salidas giro_izq, giro_der, adelante y atrás, se le asignaran el led 0, led 1, led 2, led 3 respectivamente.
- Para las entradas Sensores 1 y Sensores 2 se les asignara entradas del dip switch cuyos pines son el B9 y T8. respectivamente.



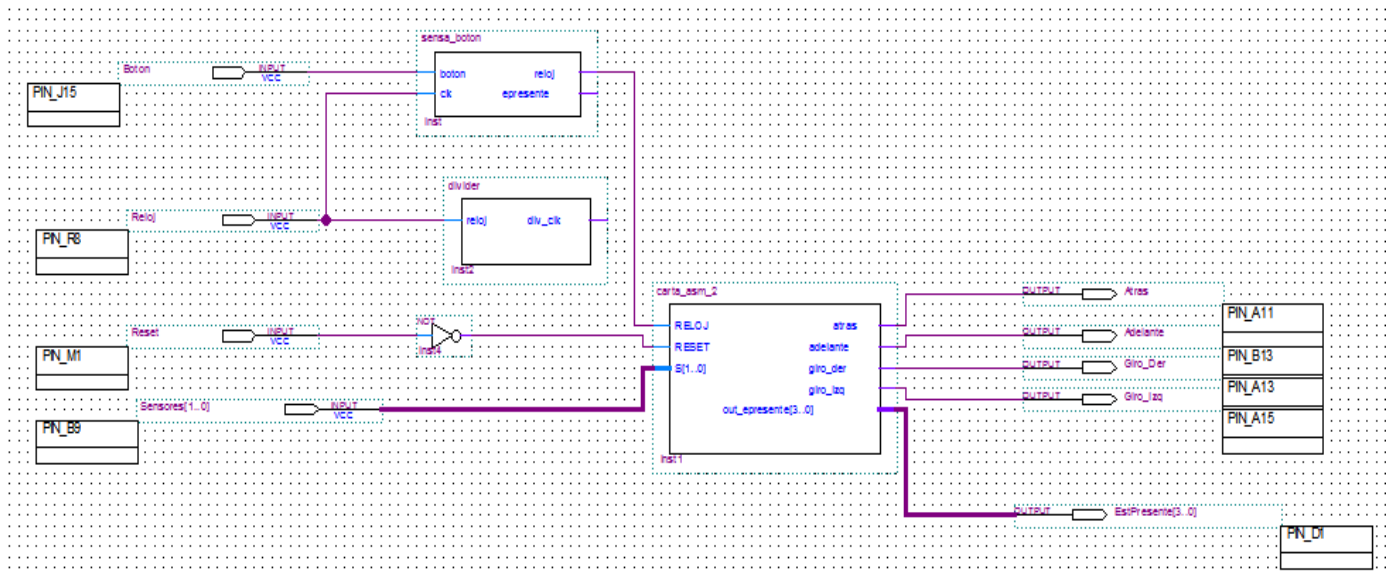
UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
FACULTAD DE INGENIERÍA
DIVISIÓN DE INGENIERÍA ELÉCTRICA



- Para el RESET se le asigna el primer switch del dip switch que es el pin M1 y para el reloj se le asigna el pin R8.

Node Name	Direction	Location	IO Bank	VREF Group	Filter Location	IO Standard
out Adelante	Output	PIN_B13	7	B7_N0	PIN_B13	2.5 V
out Atras	Output	PIN_A11	7	B7_N0	PIN_A11	2.5 V
in Boton	Input	PIN_J15	5	B5_N0	PIN_J15	2.5 V
out EstPresente[3]	Output	PIN_L3	2	B2_N0	PIN_L3	2.5 V
out EstPresente[2]	Output	PIN_B1	1	B1_N0	PIN_B1	2.5 V
out EstPresente[1]	Output	PIN_F3	1	B1_N0	PIN_F3	2.5 V
out EstPresente[0]	Output	PIN_D1	1	B1_N0	PIN_D1	2.5 V
out Giro_Der	Output	PIN_A13	7	B7_N0	PIN_A13	2.5 V
out Giro_Izq	Output	PIN_A15	7	B7_N0	PIN_A15	2.5 V
in Reloj	Input	PIN_R8	3	B3_N0	PIN_R8	2.5 V
in Reset	Input	PIN_M1	2	B2_N0	PIN_M1	2.5 V
in Sensores[1]	Input	PIN_B9	7	B7_N0	PIN_B9	2.5 V
in Sensores[0]	Input	PIN_T8	3	B3_N0	PIN_T8	2.5 V
<<new node>>						

Cerramos la ventana y podemos ver que en nuestro esquemático se realizó de forma correcta la asignación de pines.



Una vez compilado nuestro proyecto nos arroja el resumen de las acciones realizadas.



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
FACULTAD DE INGENIERÍA
DIVISIÓN DE INGENIERÍA ELÉCTRICA



La imagen muestra la interfaz de Quartus Prime Lite Edition. En la parte superior, se ve el menú de archivos y el título de la ventana. El panel izquierdo muestra el 'Project Navigator' con el proyecto 'Pract2'. El panel central superior muestra el 'Table of Contents' y el 'Flow Summary'. El panel central inferior muestra el 'Messages' con un log de compilación exitoso. El panel derecho muestra el 'Compilation Report - Pract2'.

Flow Status	Successful - Tue Mar 07 05:01:30 2017
Quartus Prime Version	15.1.0 Build 185 10/21/2015 SJ Lite Edition
Revision Name	Pract2
Top-level Entity Name	Pract2
Family	Cyclone IV E
Device	EP4CE22F17C6
Timing Models	Final
Total logic elements	22 / 22,320 (< 1 %)
Total combinational functions	17 / 22,320 (< 1 %)
Dedicated logic registers	14 / 22,320 (< 1 %)
Total registers	14
Total pins	13 / 154 (8 %)
Total virtual pins	0
Total memory bits	0 / 608,256 (0 %)
Embedded Multiplier 9-bit elements	0 / 132 (0 %)
Total PLLs	0 / 4 (0 %)

Messages:

- 204019 Generated file Pract2.vho in folder "C:/Users/RicardoEduardo/Desktop/10mo semestre/Arqui de compus/Practicas/Practica2/Practica2/simulation/modelsim/"
- 204019 Generated file Pract2_6_1200mv_85c_vhd_slow.sdo in folder "C:/Users/RicardoEduardo/Desktop/10mo semestre/Arqui de compus/Practicas/Practica2/Practica2/simulation/modelsim/"
- 204019 Generated file Pract2_6_1200mv_0c_vhd_slow.sdo in folder "C:/Users/RicardoEduardo/Desktop/10mo semestre/Arqui de compus/Practicas/Practica2/Practica2/simulation/modelsim/"
- 204019 Generated file Pract2_min_1200mv_0c_vhd_fast.sdo in folder "C:/Users/RicardoEduardo/Desktop/10mo semestre/Arqui de compus/Practicas/Practica2/Practica2/simulation/modelsim/"
- 204019 Generated file Pract2_vhd.sdo in folder "C:/Users/RicardoEduardo/Desktop/10mo semestre/Arqui de compus/Practicas/Practica2/Practica2/simulation/modelsim/"
- quartus Prime EDA Netlist writer was successful. 0 errors, 0 warnings
- 293000 Quartus Prime Full compilation was successful. 0 errors, 8 warnings

- Programando la tarjeta

Para cargar nuestro proyecto en la tarjeta Cyclon IV es necesario cargar los drivers de la tarjeta a nuestra computadora por lo cual el primer paso es conectar la tarjeta al puerto USB e ir al administrador de dispositivos buscar la controladora de bus serial y seleccionar "Altera USB-Blaster", click derecho y en propiedades seleccionar el botón de actualizar los controladores tras cargar los drivers el primer paso estará finalizado.

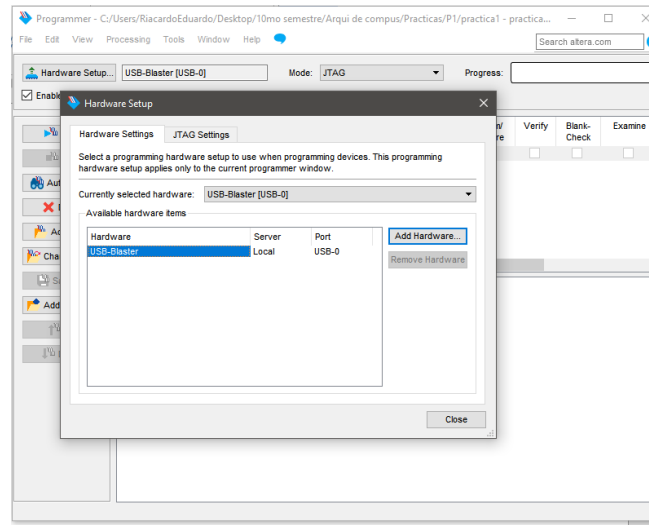
La imagen muestra el 'Administrador de dispositivos' de Windows. En el árbol de dispositivos, se expande 'Controladoras de bus serie universal' y se selecciona 'Altera USB-Blaster'. A la derecha, se muestra la ventana 'Propiedades: Altera USB-Blaster' con pestañas para General, Advanced, Power Management, Detalles y Eventos. La pestaña 'General' está activa, mostrando información sobre el proveedor (Altera), la fecha del controlador (26/08/2014), la versión (2.12.0.0) y el firmante digital (Delaware Altera Corporation). Se ven botones para 'Detalles del controlador', 'Actualizar controlador...', 'Revertir al controlador anterior...', 'Deshabilitar' y 'Desinstalar'.



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
FACULTAD DE INGENIERÍA
DIVISIÓN DE INGENIERÍA ELÉCTRICA

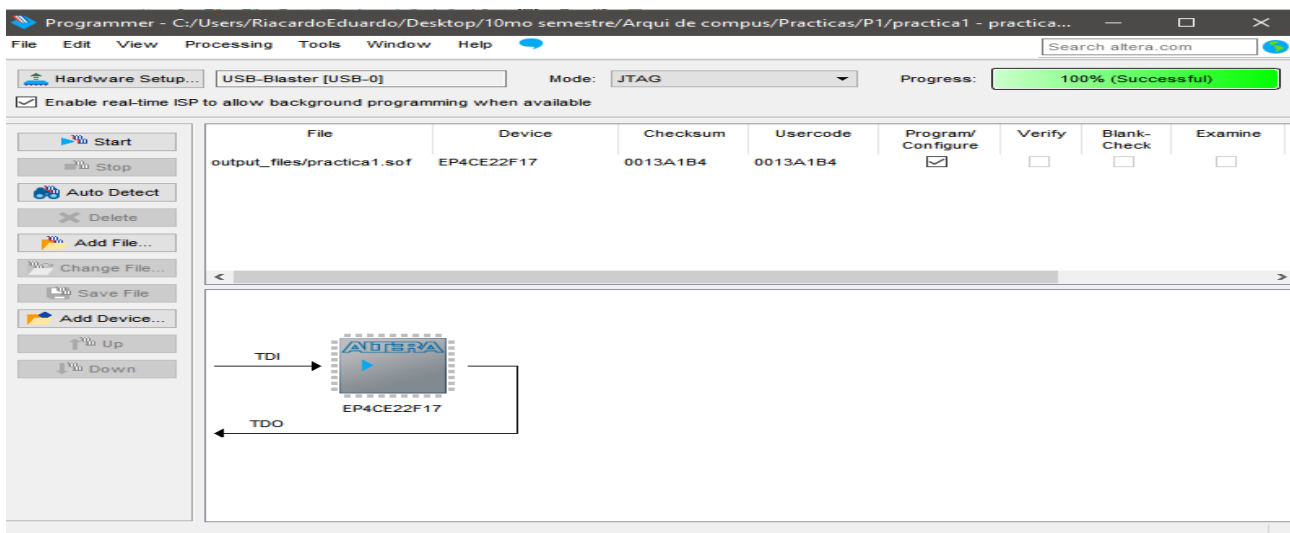


Regresamos a Quartus y seleccionamos Tool-> Programmer, después en Hardware Setup seleccionar USB-Blaster.



Cerramos la ventana, activamos el cuadro Program/Configure, seleccionaremos “ADD File” y buscaremos dentro de la carpeta de nuestro proyecto la carota llamada “outout_file” y cargaremos nuestro archivo practica2.sof finalmente y seleccionamos Start para comenzar la carga de nuestro archivo.

Si la carga fue exitosa nos mostrara la siguiente ventana y habremos terminado.





UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
FACULTAD DE INGENIERÍA
DIVISIÓN DE INGENIERÍA ELÉCTRICA



Conclusiones:

En esta práctica se implementó el algoritmo que describe el comportamiento de un sistema mediante la realización de una carta ASM que describe las acciones que deseamos tenga nuestro sistema.

Al implementar el código del algoritmo en un archivo de VHDL y verificar su sintaxis logramos cumplir con el objetivo de esta práctica.

En esta práctica pudimos observar mejor el funcionamiento de las cartas ASM, reforzando los conocimientos vistos en clase y permitiéndonos afinar el manejo de estas.

Durante la revisión, se pudo observar que el contenido de la memoria contiene: estado presente, entradas, estado siguiente y salidas, lo que visualizamos en los leds de la tarjeta son: estado siguiente y salidas. Se tendría que a ver considerado que se muestre el estado presente en lugar del siguiente, para evitar errores.

En ambas simulaciones, nos ocurrió lo mismo que en la primera partica, ya que al quitar el divisor pudimos observar en la simulación, las salidas y el estado presente.

En la tercera parte el botón realizo la función de "STOP", para el timer esto nos dio tiempo para tomar los valores de los sensores.

En las simulaciones se agregaron valores de uno, ya que con estos valores pudimos observar mayores cambios en los estados, esto no nos permitió entender mejor el funcionamiento de la practica.