DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN
ESCUELA DE INGENIERÍA
PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE

IIC2343 - Arquitectura de Computadores (II/2019)

Entrega 1-C

Entrega: 9 de Septiembre de 2019 | 10:59:59 a.m.

Requisitos

- Esta entrega es estrictamente individual. Cualquier tipo de falta a la honestidad académica será sancionada con la **reprobación** del curso con la nota mínima.
- Los nombre de archivos y el cómo deben ser ejecutados son parte del formato, no respetarlo será penalizado.
- El programa de la **placa** deberá ser realizado en `VHDL`.
- La **documentación** deberá ser realizada en un archivo `Markdown` y subirlo junto a su tarea, de nombre README.md, en el mismo repositorio.
- Esta entrega deberá ser subida a su repositorio personal de *GitHub* correspondiente en la fecha y hora dada.
- **La entrega de la placa debe ser realizada previa o al inicio de la hora de ayudantía del curso. El no cumplimiento, no solo perjudicará su nota, sino también a sus compañeros**

Introducción

El *full-adder* es un circuito digital que permite sumar dos números de un bit, más el carry de un número anterior.

La representación decimal codificado en binario, o también conocida como **BCD**, es un estándar de representación de la base decimal mediante números binarios.

Misión

Deberá crear un módulo VHDL que represente la suma de dos números de 4 bits + carry in, en tal caso, debe **encender los leds representando la suma**, y en caso de que la suma requiera más de 4 bits, **encender el led** indicando un carry out. Finalmente, **encender los led del display 7-segmentos con el resultado en base decimal**

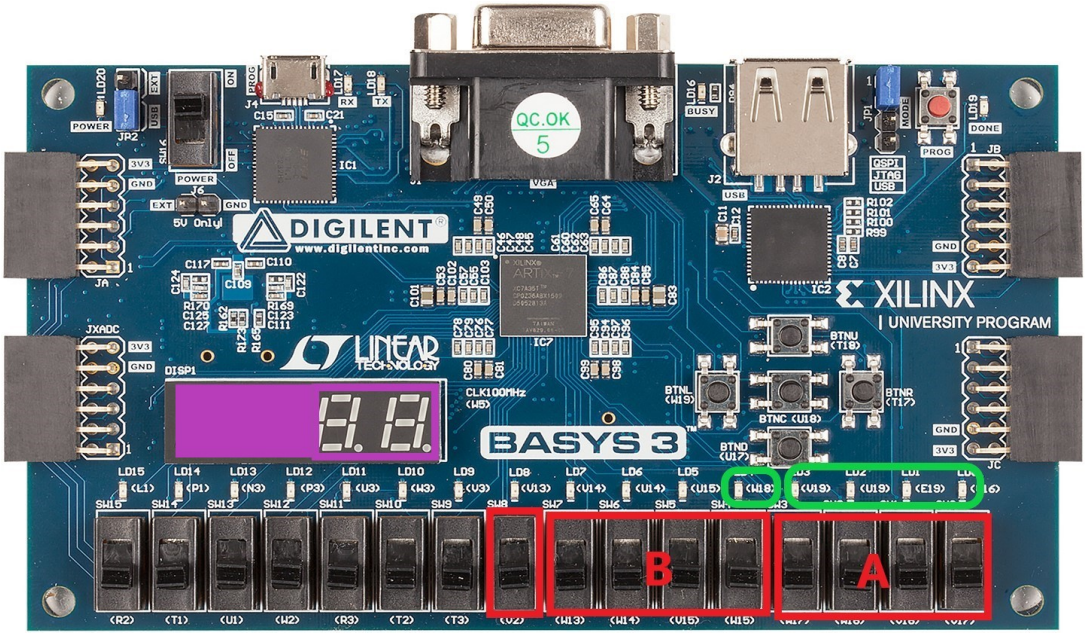
Se te entregará los módulos `Display_Controller.vhd` y `binary_bcd_decoder.vhd` que puedes utilizar de manera opcional.

Debe **documentar con lo solicitado en el README.md** .

Placa

Utilizarán la placa Digilent Basys3 presente en la imagen, de la cual, usando los **4 primeros switches** como input del número A, los **siguientes 4 switches** como input del número B, y el siguiente **siguiente switch** como el carry in.

El output será representado como los **primeros 4 leds** dando el resultado de la suma, y el **siguiente led** entregando el carry out. Además el resultado en base decimal debe quedar representado en los **dos display de 7-segmentos**.



Módulos entregados

Se le entrega dos módulos VHDL para facilitar su entrega, estos pueden ser utilizados de **manera opcional**

- `binary_bcd_decoder.vhd`

  Este módulo recibe como entrada un vector `n[4:0]` de 5 bits. y entrega como salida dos vectores `dec[3:0]` y `uni[3:0]`, ambos de 4 bits.

  Su función es decodificar la señal `n[4:0]`, en su equivalente BCD, separando la decena de la unidad en las señales `dec[3:0]` y `uni[3:0]`

  Puedes ver la sección Anexo con la tabla de verdad del módulo.

- `Display_Controller.vhd`

  Este módulo recibe 4 vectores y un valor lógico: `dis_a[3:0]`, `dis_b[3:0]`, `dis_c[3:0]`, `dis_d[3:0]` y `clk`. Entrega como salida los vectores `seg[7:0]` y `an[3:0]`.

  Su función es decodificar los valores que quieres entregar en los display 7 segmentos correspondientes:

  Para que este módulo funcione, debe conectar la señal de entrada de la Basys3 `clk`, a la señal de entrada `clk` de este módulo.

  La salida de este módulo genera las señales de salida de la Basys3: `seg[7:0]` y `an[3:0]` para encender el display.

Requerimientos

Para implementar declaraciones condicionales **solamente** se permite hacer uso de bloques `with/select`. El uso de los *statements process*, `case` e `if/else` quedan absolutamente prohibidos. Esto porque se privilegia el uso de selectores y operaciones lógcias básicas para el desarrollo de esta tarea. **Para esta entrega, queda estrictamente prohibido utilizar cualquier tipo de librería aritmética que simplifique la tarea de la suma**

- Crear el proyecto
  - Seleccionar las opciones correctas para crear el proyecto en Vivado, que funcione con la placa correspondiente.
  - Importar correctamente el archivo `Basys3.xdc` .
  - Configurar correctamente las *constrains* del archivo `Basys3.xdc`. Descomentando las líneas correctas del archivo.
- (6 pts) Crear el módulo `fulladder4.vhd`
  - (4 pts) Crear una *source* llamada `fulladder4`, que contiene la arquitectura requerida para resolver el problema y entrega el resultado de la suma en los leds.
  - (2 pts) La arquitectura entrega el resultado de la suma en los display 7 - segmentos.
  - Puede crear más *sources* para facilitar el problema
- **Incluir el README.md con lo solicitado.**
- **Entregar inmediata de la placa la siguiente semana**

Entrega

La entrega se realizará a través de GitHub. El repositorio debe contener una carpeta con su proyecto de Vivado y el archivo `.bit`. En el caso de la carpeta del proyecto, deben subir solo la carpeta `.srcs`, el archivo `.xpr` y el archivo `Basys3.xdc`

Anexo

Tabla de verdad `binary_bcd_decoder.vhd`

n_{10}	n_2[4:0]					dec[3:0]				uni[3:0]			
0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	1	0	0	0	0	0	0	0	1
2	0	0	0	1	0	0	0	0	0	0	0	1	0
3	0	0	0	1	1	0	0	0	0	0	0	1	1
4	0	0	1	0	0	0	0	0	0	0	1	0	0
5	0	0	1	0	1	0	0	0	0	0	1	0	1
6	0	0	1	1	0	0	0	0	0	0	1	1	0
7	0	0	1	1	1	0	0	0	0	0	1	1	1
8	0	1	0	0	0	0	0	0	0	1	0	0	0
9	0	1	0	0	1	0	0	0	0	1	0	0	1
10	0	1	0	1	0	0	0	0	1	0	0	0	0
11	0	1	0	1	1	0	0	0	1	0	0	0	1
12	0	1	1	0	0	0	0	0	1	0	0	1	0
13	0	1	1	0	1	0	0	0	1	0	0	1	1
14	0	1	1	1	0	0	0	0	1	0	1	0	0
15	0	1	1	1	1	0	0	0	1	0	1	0	1
16	1	0	0	0	0	0	0	0	1	0	1	1	0
17	1	0	0	0	1	0	0	0	1	0	1	1	1
18	1	0	0	1	0	0	0	0	1	1	0	0	0
19	1	0	0	1	1	0	0	0	1	1	0	0	1
20	1	0	1	0	0	0	0	1	0	0	0	0	0
21	1	0	1	0	1	0	0	1	0	0	0	0	1
22	1	0	1	1	0	0	0	1	0	0	0	1	0
23	1	0	1	1	1	0	0	1	0	0	0	1	1
24	1	1	0	0	0	0	0	1	0	0	1	0	0
25	1	1	0	0	1	0	0	1	0	0	1	0	1
26	1	1	0	1	0	0	0	1	0	0	1	1	0
27	1	1	0	1	1	0	0	1	0	0	1	1	1
28	1	1	1	0	0	0	0	1	0	1	0	0	0
29	1	1	1	0	1	0	0	1	0	1	0	0	1
30	1	1	1	1	0	0	0	1	1	0	0	0	0
31	1	1	1	1	1	0	0	1	1	0	0	0	1