Grado en Ingeniería de Computadores

# Tecnología de Computadores

# Práctica 2

Universidad Rey Juan Carlos

## 1. Objetivo

En esta segunda sesión de prácticas se pretende que el alumno se familiarice con el álgebra de Boole y la representación de funciones lógicas combinacionales en sus formas canónicas.

Una vez terminada la práctica el alumno será capaz de:

- Diseñar un circuito combinacional sencillo en VHDL a partir de su función lógica
- Diseñar un circuito combinacional sencillo en VHDL a partir de su tabla de verdad
- Diseñar un circuito combinacional sencillo en VHDL a partir de su esquemático
- Aplicar los teoremas y propiedades del álgebra de Boole para simplificar funciones lógicas
- Implementar diseños sencillos en una FPGA y comprobar su funcionamiento

## 2. Desarrollo de la práctica

**Crear un circuito a partir de su función lógica.**

Primero crearemos un circuito en VHDL a partir de una función lógica dada. Supongamos que tenemos la siguiente ecuación:

$$F = \bar{X} \cdot Y \cdot Z + \bar{X} \cdot \bar{Y} \cdot Z$$

Esta ecuación define la salida (F) de un circuito lógico que tiene 3 entradas (X, Y y Z). Por lo tanto, podemos construir una entidad en VHDL para definir la caja negra de nuestro circuito. Para ello crearemos un fichero fuente en un nuevo proyecto de Vivado y editaremos la entidad quedando como sigue:

```
entity ej1 is
    Port (X, Y, Z : in  STD_LOGIC;
          F       : out STD_LOGIC
    );
end ej1;
```

Finalmente, en la arquitectura, definimos directamente la ecuación anterior, quedando:

```
architecture Dataflow of ej1 is
begin
    F <= (not(X) and Y and Z) or (not(X) and not(Y) and Z);
end Dataflow;
```

Cabe destacar que a la arquitectura de la función se la ha llamado *Dataflow*. Se podría haber elegido otro nombre, sin embargo, es conveniente que el nombre de la arquitectura indique el tipo de la misma. Por ejemplo, cuando se trata de un circuito formado únicamente por ecuaciones lógicas se suele usar el nombre *Dataflow*, mientras que cuando se describe el comportamiento de un circuito se suele denominar *Behavioral*. En las clases de teoría se incidirá más en los tipos de arquitecturas.

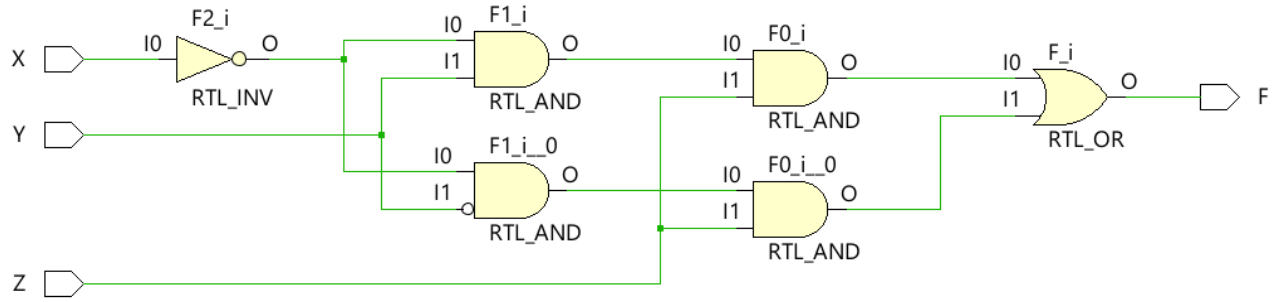
Una vez definido el circuito, se va a proceder a su simulación. Para ello utilice el código del testbench "*tb_ej1.vhd*" proporcionado por el profesor. Ejecute la simulación y construya la tabla de verdad del circuito.

Incluya ahora el fichero "*constraints.xdc*" proporcionado por el profesor y genere el bitstream del circuito. Implemente dicho circuito en la FPGA y verifique la tabla de verdad anterior manipulando los interruptores de la placa y observando si se enciende o apaga el LED.

Observando la ecuación inicial de nuestro diseño, podemos ver que se puede simplificar aplicando la propiedad distributiva del álgebra de Boole junto con el teorema del complemento:

$$F = \bar{X} \cdot Y \cdot Z + \bar{X} \cdot \bar{Y} \cdot Z \underbrace{=}_{Prop.distributiva} \bar{X} \cdot Z \cdot (Y + \bar{Y}) \underbrace{=}_{T^a\ complemento} \bar{X} \cdot Z$$

Por lo tanto, podemos implementar el mismo circuito pero utilizando otra ecuación distinta. Vamos a comprobar si existe alguna diferencia. Primero, vamos a generar el diseño elaborado del circuito inicial pulsando sobre *Open Elaborated Design* obteniendo:




Por otra parte, abra también el diseño implementado pulsando sobre *Open Implemented Design* y genere el informe de utilización de recursos pulsando en *Report Utilization*. Anote el número de LUTs e IOBs que se necesitan para implementar este circuito.

Ahora vamos a modificar la entidad y la arquitectura del código VHDL cambiando la ecuación inicial por la que simplificamos aplicando los teoremas del álgebra de Boole. Nos quedaría:

```
entity ej1 is
    Port (X, Z : in  STD_LOGIC;
          F    : out STD_LOGIC
    );
end ej1;

architecture Dataflow of ej1 is
begin
    F <= not(X) and Z;
end Dataflow;
```

Dado que no tenemos entrada Y, también tenemos que eliminarla del fichero de constraints. Hecho esto, abra de nuevo el diseño elaborado pulsando en *Open Elaborated Design* y observe el nuevo circuito. ¿Hay alguna diferencia? ¿Por qué?

Para terminar la comparativa, ejecute de nuevo la implementación del circuito y obtenga el informe de utilización de recursos. ¿Hay alguna diferencia en el número de LUTs e IOBs utilizadas por el circuito simplificado? ¿A qué cree que es debido?

**Crear un circuito a partir de su tabla de verdad.**

Ahora vamos a partir de la tabla de verdad de un circuito y, mediante las dos formas canónicas, crearemos el código VHDL que lo defina. Partimos de la siguiente tabla de verdad:

| A | B | C | Z |
|---|---|---|---|
| 0 | 0 | 0 | **1** |
| 0 | 0 | 1 | **0** |
| 0 | 1 | 0 | **0** |
| 0 | 1 | 1 | **0** |
| 1 | 0 | 0 | **0** |
| 1 | 0 | 1 | **0** |
| 1 | 1 | 0 | **0** |
| 1 | 1 | 1 | **1** |

Se observa que el circuito tiene tres entradas (A, B y C) y una salida (Z). Por lo que la entidad se puede construir fácilmente como en el ejercicio anterior. Para obtener la arquitectura del código VHDL debemos obtener previamente la ecuación que defina el circuito. Para ello haremos uso de las dos formas canónicas (suma de productos y producto de sumas).

Como suma de productos la ecuación lógica de la salida Z queda:

$$Z = A \cdot B \cdot C + \bar{A} \cdot \bar{B} \cdot \bar{C}$$

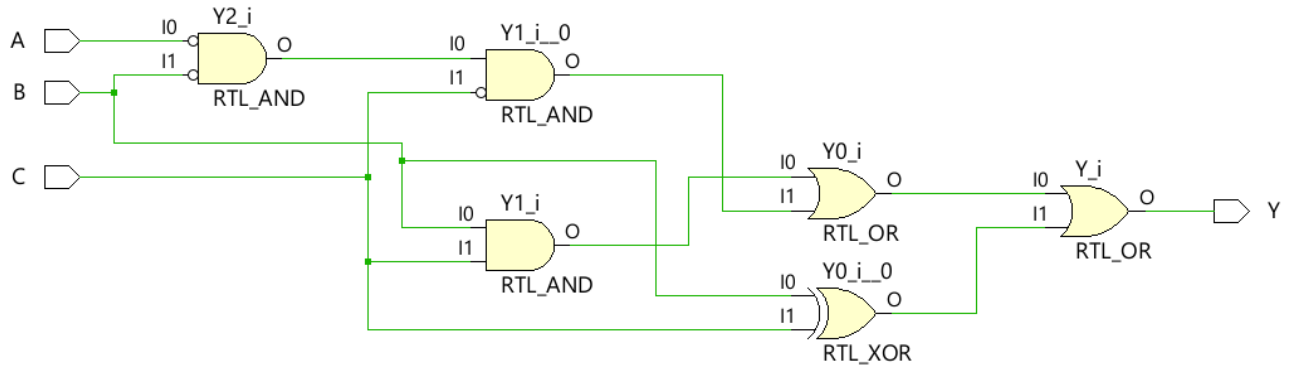
Esta ecuación está formada por los mintérminos 0 y 7 que, como se observa en la tabla de verdad, son los que hacen que la función valga 1.

Ahora, a partir de esta ecuación ya se podría definir la arquitectura del circuito, completando así el código VHDL del mismo. Sin embargo, como se mencionó anteriormente, también se puede obtener una ecuación para Z utilizando la segunda forma canónica (producto de sumas), por lo que tenemos dos formas de representar el mismo circuito.

Obtenga la segunda forma canónica de Z y determine el número de puertas lógicas (AND, OR y NOT) utilizadas por ambas implementaciones del circuito (como suma de productos y como producto de sumas). Genere el informe de utilización de recursos para ambos circuitos y determine el número de LUTs e IOBs utilizados. ¿Cuál de los dos circuitos utiliza un menor número de estos recursos de la FPGA? ¿Por qué?

**Crear un circuito a partir de su esquemático.**

Finalmente, en este tercer ejercicio vamos a obtener el código VHDL de un nuevo circuito a partir de su esquemático. Partiremos del siguiente esquemático:



Primero tendremos que obtener la ecuación de la salida calculando cada una de las funciones lógicas intermedias que se van generando a la salida de cada puerta lógica. Una vez obtenida la función lógica, construir la entidad y la arquitectura en VHDL es trivial.

Obtenga la función lógica de la salida Y del circuito anterior. Utilice el fichero de testbench "*tb_ej3.vhd*" para simular el circuito y complete su tabla de verdad. Llame a la entidad del circuito creado "ej3".

## 3. Ejercicios propuestos

Como se ha visto a lo largo de la práctica, un circuito se puede expresar como una función lógica, una tabla de verdad, un esquemático o un código VHDL. Estas cuatro representaciones son idénticas y se puede pasar de una a otra de forma sencilla. A modo de repaso de estos conceptos se plantea estos últimos ejercicios:

1. Obtener la tabla de verdad de la siguiente función lógica:

$$Y = A \cdot B \cdot C + \bar{A} + A \cdot \bar{B} \cdot C$$

2. Obtener la tabla de verdad del siguiente esquemático: