Cómo generar puertos diferenciales en Vivado

Creador: David Rubio G.

Entrada: https://soceame.wordpress.com/2024/11/30/como-generar-puertos-diferenciales-en-vivado/

Blog: https://soceame.wordpress.com/

GitHub: https://github.com/DRubioG

Fecha última modificación: 23/02/2025

En una anterior entrada ya comenté cómo se puede definir un puerto de reloj diferencial.

https://soceame.wordpress.com/2024/10/19/declarar-pines-de-reloj-diferenciales-en-vivado/

En esta entrada vamos a centrarnos en definir señales diferenciales.

Para la definición de puertos diferenciales se tiene que hacer mediante buffers internos, y para ello hay tres grupos: *puertos diferenciales de entrada*, *puertos diferenciales de salida* y *puertos diferenciales de entrada/salida*.

Todas las diferentes opciones de buffers se declaran desde la librería *UNISIM* de Vivado. Algunos de estos buffers no están disponibles en todas las diferentes familias, pero los básicos sí que están para todas las familias.

```
library UNISIM;
use UNISIM.vcomponents.all;
```

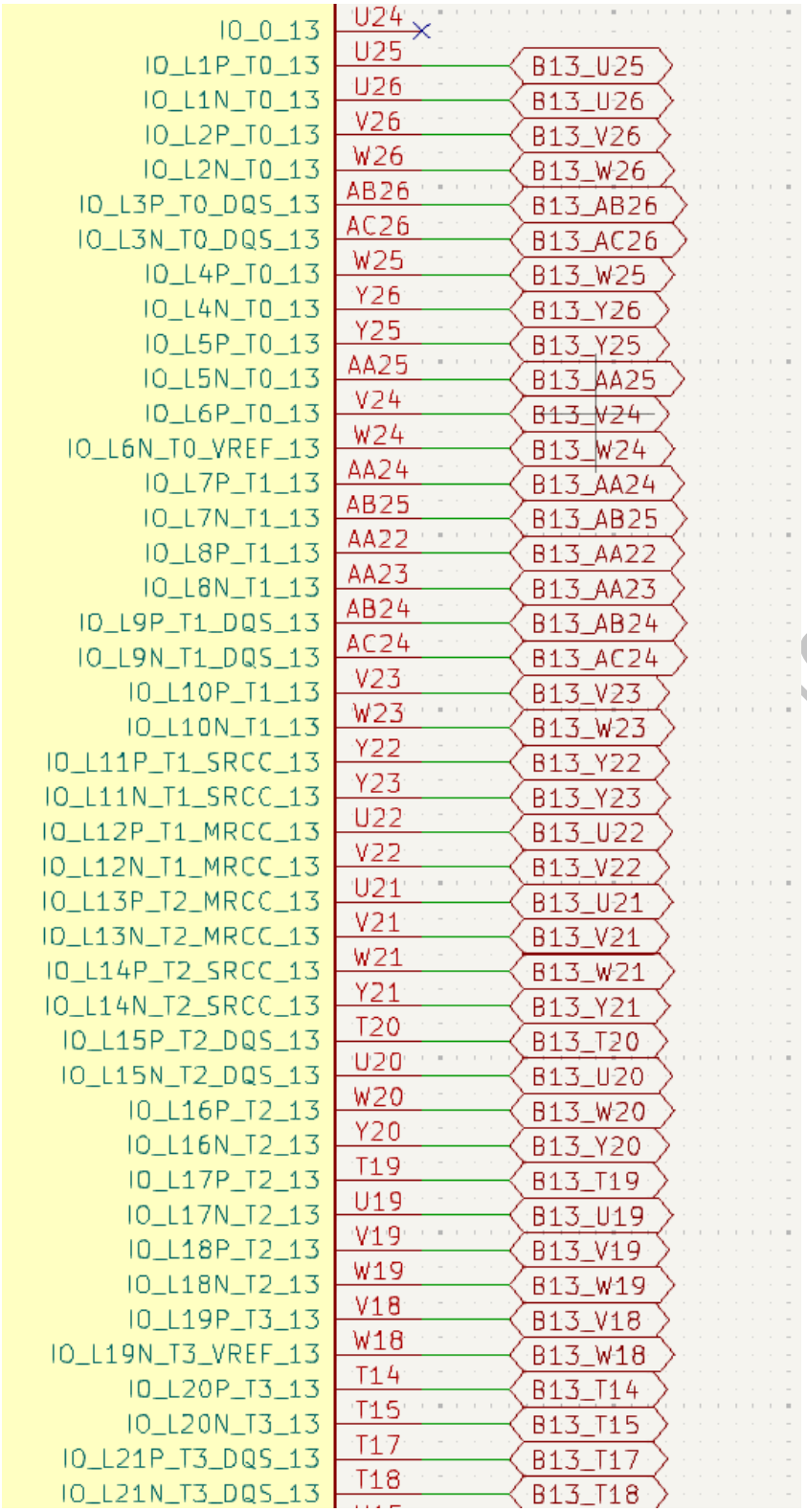
Nota: *todas las entradas a los buffers son entradas simples, o sea, de un bit.*

NOTA: *Todos los buffers tienen al menos una conexión con el exterior o con el interior. Bien, pues las conexiones con el interior no se pueden sacar directamente al exterior, requieren de un buffer, tanto de entrada como de salida.*

Y solo es posible utilizar pines diferenciales definidos en los chips de Xilinx.

Recordatorio

A modo de recordatorio, se recuerda que las FPGAs de Xilinx tienen pines diferenciales definidos. Por lo que si se quieren declarar pines diferenciales solo se pueden utilizar estos pines.

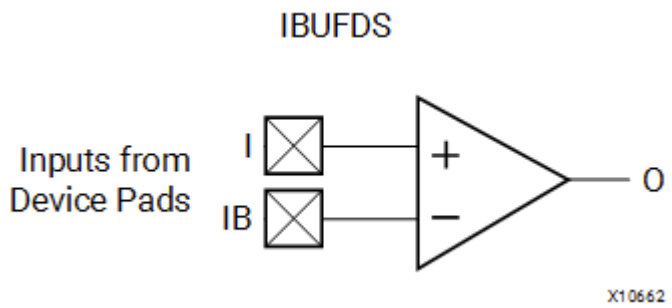


Si se utilizan los pines diferenciales complementarios de Xilinx, en la definición de pines solo es necesario declarar uno de los dos pines en el XDC, porque deduce el otro como el complementario (el nombre lo saca de la definición del buffer).

Puertos diferenciales de entrada

Para los puertos diferenciales de entrada tenemos diferentes opciones de buffers.

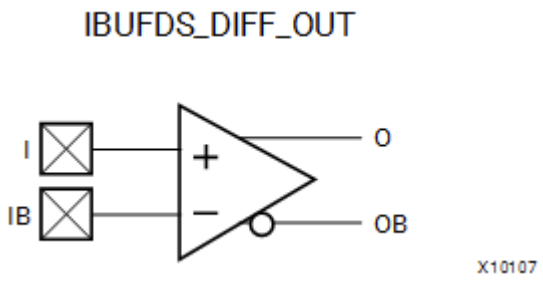
- **IOBUFDS**: Este buffer recibe una entrada diferencial a la entrada y la convierte en un señal simple de salida. Tiene dos entrada, la I para la señal positiva y la IB para la señal negativa.



Para instanciarlo:

```
IBUFDS_inst : IBUFDS
port map (
   O => O,
   I => I,
   IB => IB
);
```

- **IBUFDS_DIFF_OUT**: Este es un buffer de entrada diferencial y de salida diferencial.

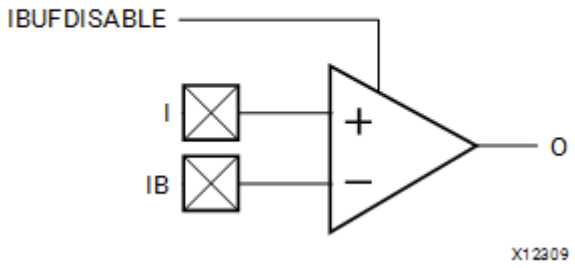


Para instanciarlo:

```
IBUFDS_DIFF_OUT_inst : IBUFDS_DIFF_OUT
port map (
   O => O,
   OB => OB
   I => I,
   IB => IB
);
```

- **IBUFDS_IBUFDISABLE**: Este buffer es un buffer con entrada diferencial y salida simple con señal de habilitación. La señal de deshabilitación (*IBUFDISABLE*) es activa a nivel alto.

IBUFDS_IBUFDISABLE



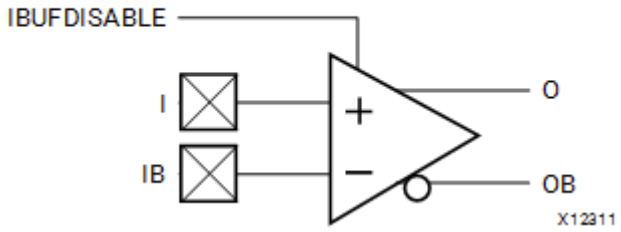
Para instanciarlo, este buffer tiene un genérico por si se quiere activar la señal de deshabilitación del buffer.

```
IBUFDS_IBUFDISABLE_inst : IBUFDS_IBUFDISABLE
generic map (
   SIM_DEVICE => "VERSAL_AI_CORE",
   USE_IBUFDISABLE => "TRUE"
)
port map (
   O => O,
   I => I,
   IB => IB,
   IBUFDISABLE => IBUFDISABLE
);
```

- **IBUFDS_DIFF_OUT_IBUFDISABLE**: Este es un buffer con señal de entrada diferencial y con salida diferencial, pero con señal de deshabilitación (*IBUFDISABLE*). Esta señal es activa a nivel bajo.

*Este buffer no está soportado en la familia UltraScale.

IBUFDS_DIFF_OUT_IBUFDISABLE

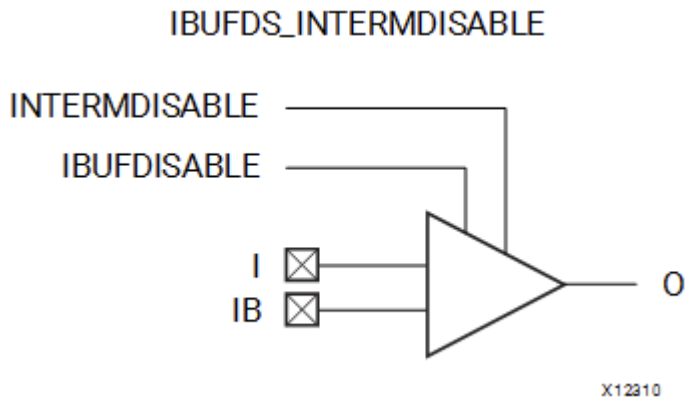


Este buffer tiene diferentes simulaciones, dependiendo de la familia por defecto la serie 7 (*7SERIES*). Para instanciarlo:

```
IBUFDS_DIFF_OUT_IBUFDISABLE_inst : IBUFDS_DIFF_OUT_IBUFDISABLE
generic map (
   SIM_DEVICE => "7SERIES"
)
```

```
port map (
   O => O,
   OB => OB,
   I => I,
   IB => IB,
   IBUFDISABLE => IBUFDISABLE
);
```

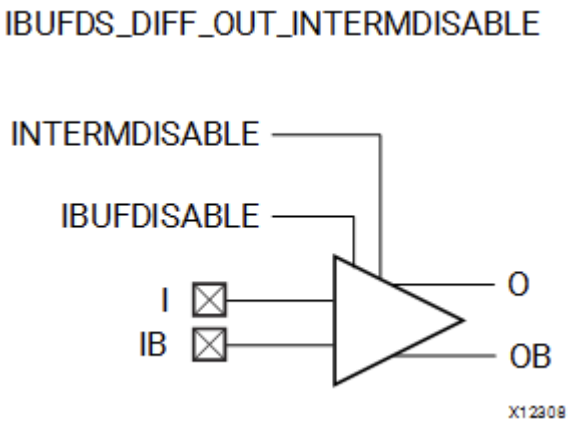
- **IBUFDS_INTERMDISABLE**: Este buffer tiene una entrada diferencial y una salida simple con dos señales de deshabilitación. *IBUFDISABLE*, que es activa a nivel alto, y *INTERMDISABLE*, que es activa a nivel bajo.



Para instanciarlo:

```
IBUFDS_INTERMDISABLE_inst : IBUFDS_INTERMDISABLE
generic map (
   SIM_DEVICE => "VERSAL_AI_CORE",
   USE_IBUFDISABLE => "TRUE"
)
port map (
   O => O,
   I => I,
   IB => IB,
   IBUFDISABLE => IBUFDISABLE,
   INTERMDISABLE => INTERMDISABLE
);
```

- **IBUFDS_DIFF_OUT_INTERMDISABLE**: Este buffer es un buffer de entrada diferencial y salida diferencial, con doble puerto de habilitación. Esta el *IBUFDISABLE* que es activo a nivel bajo e *INTERMDISABLE*, que es activo a nivel bajo.

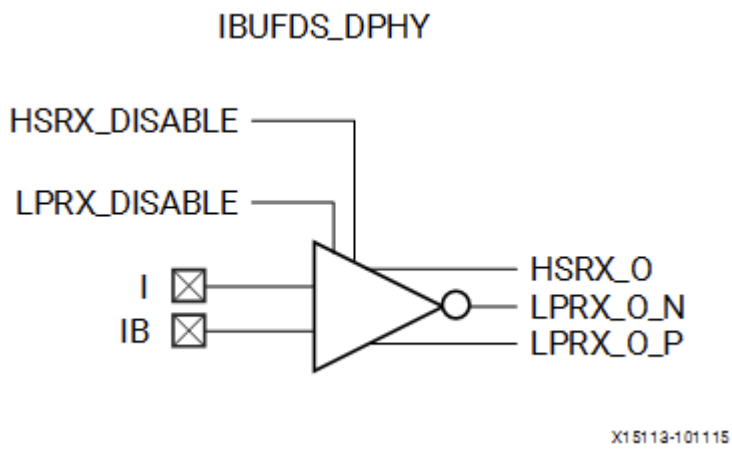


Para instanciarlo:

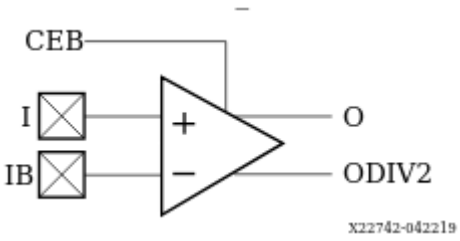
```
IBUFDS_DIFF_OUT_INTERMDISABLE_inst :
IBUFDS_DIFF_OUT_INTERMDISABLE
generic map (
   SIM_DEVICE => "VERSAL_AI_CORE"
)
port map (
   O => O,
   OB => OB,
   I => I,
   IB => IB,
   IBUFDISABLE => IBUFDISABLE,
   INTERMDISABLE => INTERMDISABLE
);
```

Luego también tenemos los buffers de aplicación específica.

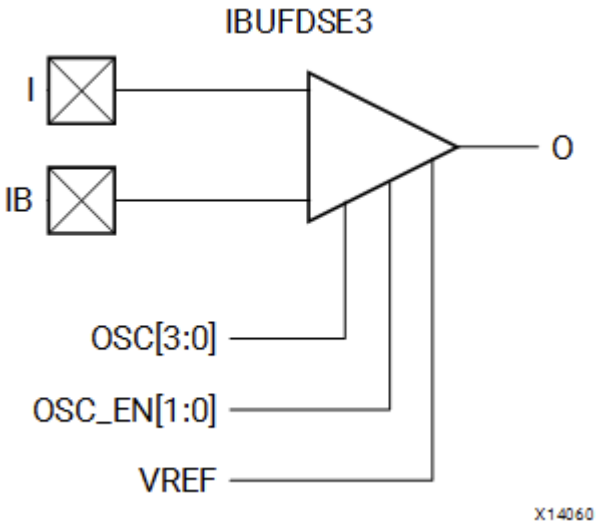
- **IBUFDS_DPHY**: Este es un buffer con soporte para MIPI (Mobile Industry Processor Interface)



- **IBUFDS_GTE2/IBUFDS_GTE3/IBUFDS_GTE4/IBUFDS_GTE5**: Este buffer es el buffer que se utiliza en los transceivers. Depende del dispositivo se utiliza uno u otro.



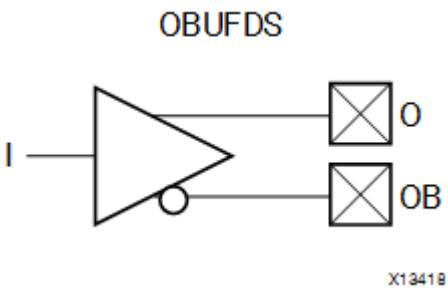
- **IBUFDSE3**: Este es un buffer con calibración de offset.



Puertos diferenciales de salida

Para los puertos diferenciales de salida tenemos diferentes opciones de buffers.

- **OBUFDS**: Este buffer es un buffer de entrada simple y de salida diferencial.

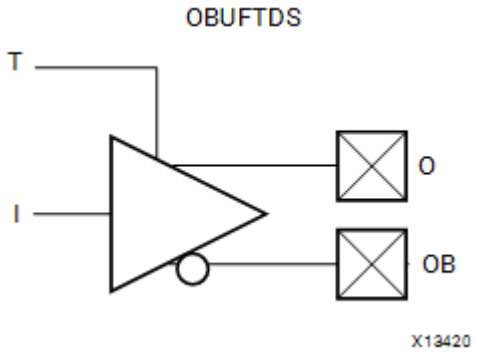


Para instanciarlo:

```
OBUFDS_inst :
OBUFDS
port map (
   O => O,
```

```
   OB => OB,
   I => I
);
```

- **OBUFTDS**: Este es un buffer de entrada simple y de salida diferencial con señal de habilitación. La señal de habilitación (T) es activa a nivel bajo.

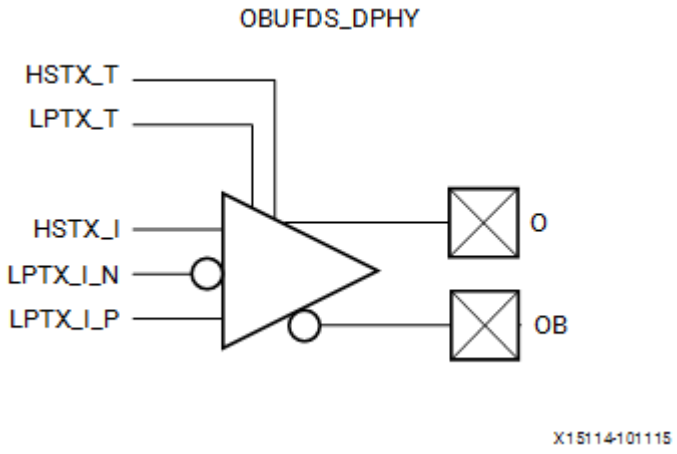


Para instanciarlo:

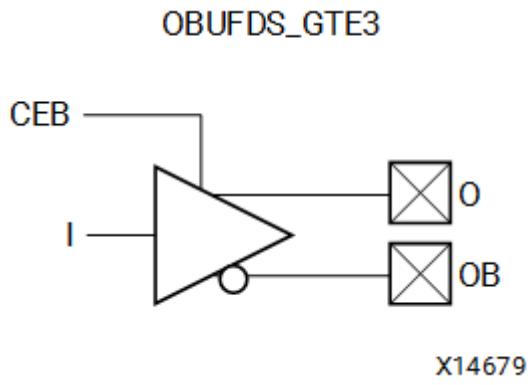
```
OBUFTDS_inst :
OBUFTDS
port map (
   O => O,
   OB => OB,
   I => I,
   T => T
);
```

Luego también tenemos los buffers de aplicación específica.

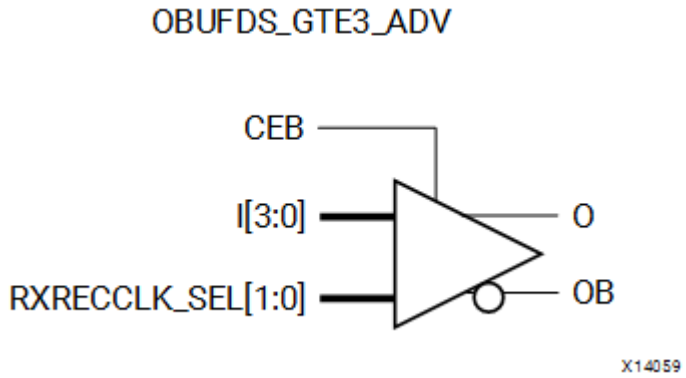
- **OBUFDS_DPHY**: Este es un buffer con soporte para MIPI (Mobile Industry Processor Interface)



- **OBUFDS_GTE2/OBUFDS_GTE3/OBUFDS_GTE4/OBUFDS_GTE5**: Este buffer es el buffer que se utiliza en los transceivers. Depende del dispositivo se utiliza uno u otro.



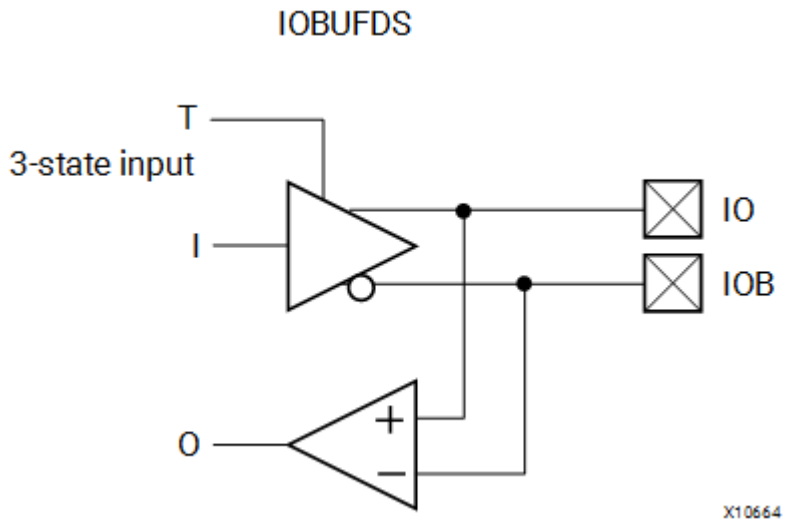
- **OBUFDS_GTE3_ADV/OBUFDS_GTE4_ADV**: Este buffer se utiliza también en transceivers.



Puertos diferenciales de entrada/salida

Para los puertos diferenciales de entrada/salida tenemos diferentes opciones de buffers.

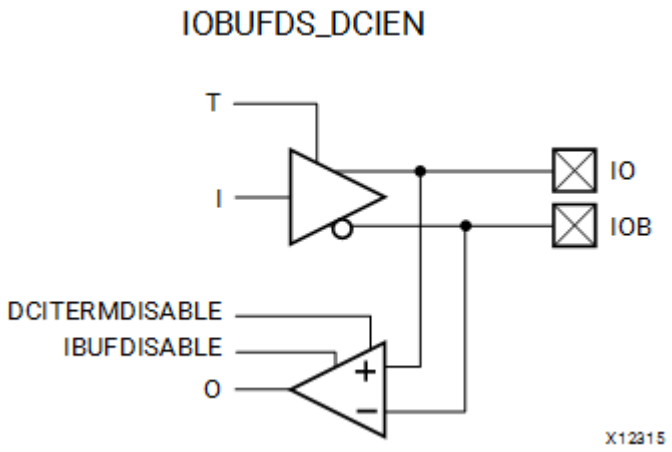
- **IOBUFDS**: Este es un buffer de entrada y salida simple, desde la FPGA, y de entrada y salida diferencial hacia el exterior. Para ello cuenta con una señal de selección de entrada y salida, a nivel alto actúa como entrada y a nivel bajo actúa como salida.



Para instanciarlo:

```
IOBUFDS_inst : IOBUFDS
port map (
   O => O,
   I => I,
   IO => IO,
   IOB => IOB,
   T => T
);
```

- **IOBUFDS_DCIEN**: Este es un buffer de entrada y salida simple desde la FPGA, y de entrada y salida diferencial, desde el exterior, y además con habilitación de entrada.

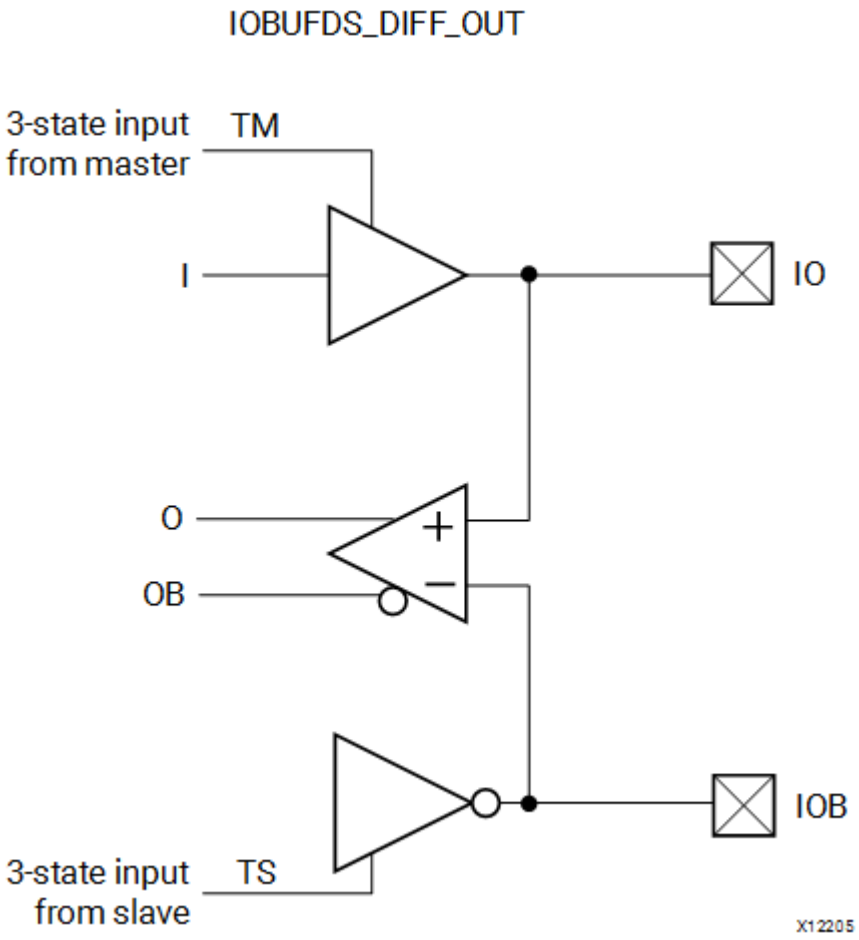


Para instanciarlo:

```
IOBUFDS_DCIEN_inst : IOBUFDS_DCIEN
generic map (
   SIM_DEVICE => "ULTRASCALE",
   USE_IBUFDISABLE => "TRUE"
```

```
)
port map (
   O => O,
   DCITERMDISABLE => DCITERMDISABLE,
   I => I,
   IBUFDISABLE => IBUFDISABLE,
   IO => IO,
   IOB => IOB,
   T => T
);
```

- **IOBUFDS_DIFF_OUT**: Este es un buffer diferencial de entrada y de salida diferencial.



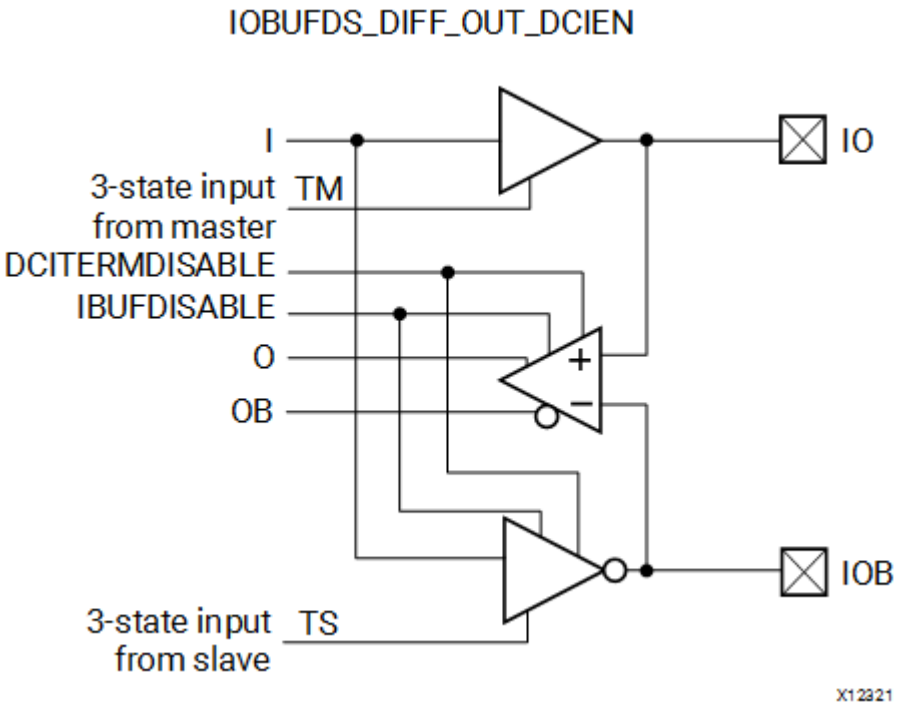
Para instanciarlo:

```
IOBUFDS_DIFF_OUT_inst : IOBUFDS_DIFF_OUT
port map (
   O => O,
   OB => OB,
   I => I,
   IO => IO,
   IOB => IOB,
```

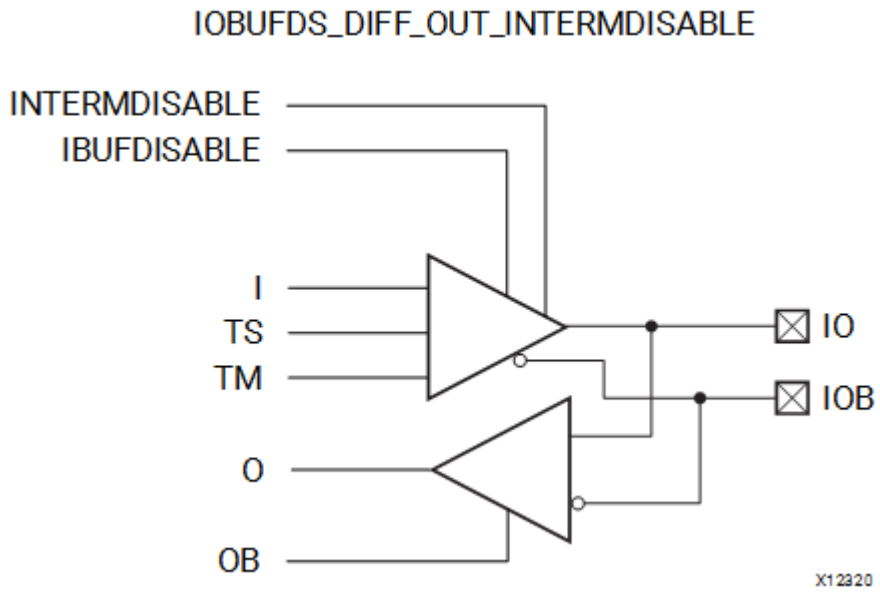
```
    TM => TM,
    TS => TS
);
```

Luego también tenemos los buffers de aplicación específica y de complejidad importante.

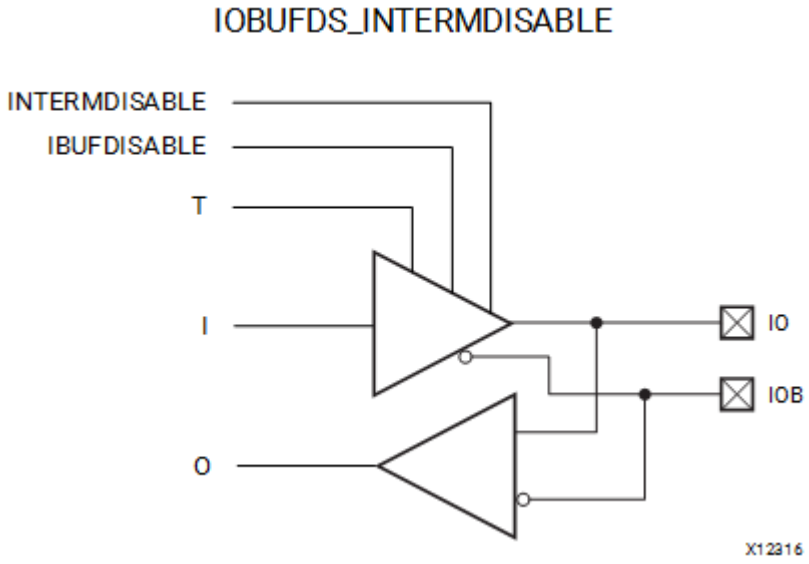
- **IOBUFDS_DIFF_OUT_DCIEN**: Este es un buffer bidireccional con entrada y salida diferencial. Con habilitación de entrada y de salida negativa.



- **IOBUFDS_DIFF_OUT_INTERMDISABLE**: Este es un buffer bidireccional simple con salida diferencial y habilitación de salida.



- **IOBUFDS_INTERMDISABLE**: Este en un buffer bidireccional simple y de entrada y salida diferencial.

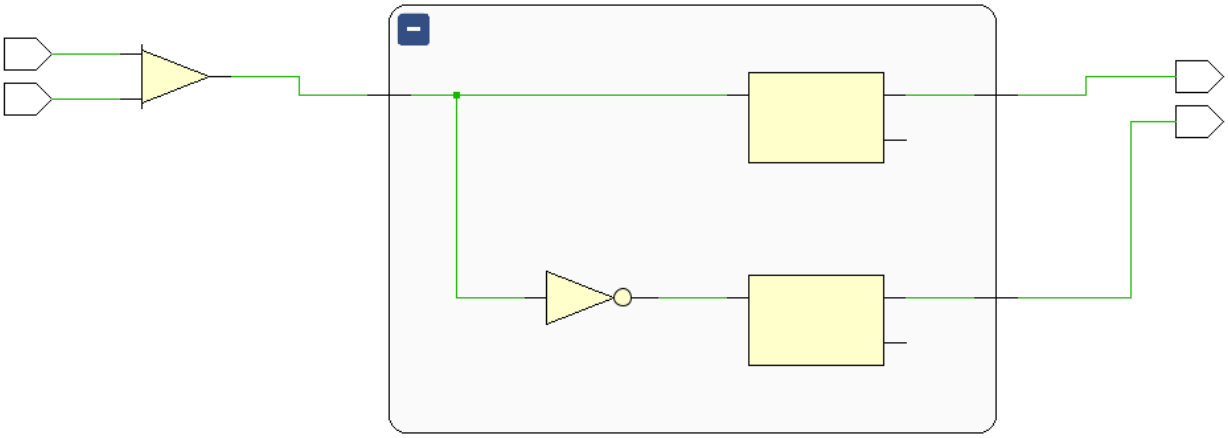


Ejemplo

Pongo un pequeño ejemplo, si utilizamos dos buffers diferenciales *IBUFDS* y el *OBUFDS*, para plasmar las dos opciones de buffers diferenciales. Para evitar que el sintetizador se coma todo lo generado y conecte la entrada directamente a la salida, se tiene que poner algo de lógica en el medio.

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;


library UNISIM;
use UNISIM.VComponents.all;

entity test is
    Port (
        i, ib : in std_logic;
        o, ob : out std_logic
    );
end test;

architecture Behavioral of test is

signal conection, connection_not : std_logic;

begin


IBUFDS_inst : IBUFDS
generic map (
   DIFF_TERM => FALSE,
   IBUF_LOW_PWR => TRUE,
   IOSTANDARD => "DEFAULT")
port map (
   O => conection,
   I => I,
   IB => IB
);

connection_not <= not conection;

OBUFDS_inst : OBUFDS
generic map (
   IOSTANDARD => "DEFAULT",
   SLEW => "SLOW")
port map (
   O => O,
   OB => OB,
   I => conection
);

end Behavioral;
```


Al sintetizarlo se genera un esquema como el siguiente.



NOTA: *el sintetizador es lo suficientemente inteligente como para saber que si se niega la entrada diferencial, esa entrada se convierte en la señal negativa diferencial de salida.*

NOTA 2: *es muy posible que el modelo diferencial de puertas lógicas de salida que utilice Xilinx sea el que se ve en el bloque, utilizando una NOT con un buffer, y lo que haga es alternar las salidas dependiendo de la salida del modelo sintetizado.*

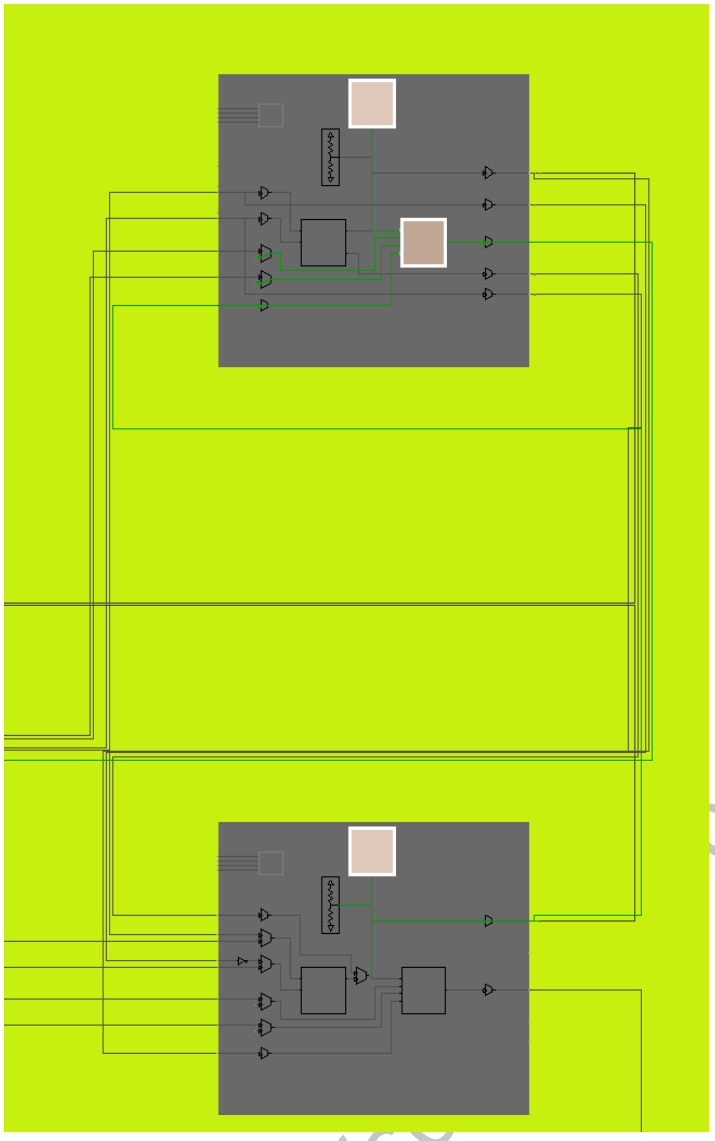
Ahora para definir los pines de entrada, solo hay **una opción** y es **utilizar un para de pines diferenciales del chip**, **NO hay opción a utilizar otros pines para hacerlos diferenciales**, es más, Vivado si lo intentas elige el pin negativo como pin de referencia y deduce el positivo. Entonces, para definir los pines solo es necesario declarar el pin positivo, en mi caso el D19, y Vivado deduce como el negativo el D20.
Con los pines de salida igual, se define el Y18, y Vivado deduce que el negativo es el Y19.

```
set_property -dict { PACKAGE_PIN D19   IOSTANDARD DIFF_SSTL18_I } [get_ports { i }];

set_property -dict { PACKAGE_PIN Y18   IOSTANDARD LVCMOS33 } [get_ports { o }];
```

NOTA: *También es conveniente recordar que no se pueden declarar los pines diferenciales como LVCMOS porque esto es solo para señales simples. Para las señales diferenciales se tiene que utilizar otro tipo de tecnología como DIFF_SSTL, TMDS o DIFF_HSTL.*
Y no todas estas tecnologías aceptan tensiones de 3.3V (es conveniente revisar bien las tensiones de alimentación del banco de pines, las tensiones de 1.8V y de 2.5V son las que más se utilizan para estos pines).
Recuerdo también esta entrada, para la comprobación de las tensiones.

https://soceame.wordpress.com/2024/10/19/se-puede-cambiar-cambiar-la-tension-de-salida-de-un-banco-de-una-fpga/

En la implementación Vivado selecciona los dos buffers de entrada *(que son los que están en la parte superior)* y el buffer diferencial *(que es el que está en la parte superior derecha).*



Con los diferenciales de salida también pasa lo mismo, utiliza los buffers diferenciales, *que son los de la izquierda*, y los pines de salida, *los de la parte superior.*