

# Arquitectura de Computadora

## **TP1: ALU**

### Autores

- Orecchini Alem, Stefano Mauricio
- Molina, Franco Elias

Año 2023

## Introducción

Se va a llevar a cabo el primer trabajo práctico de la materia de Arquitectura de Computadoras, el consiste en desarrollar e instanciar una Unidad Aritmético Lógica o ALU en una board con una FPGA.

Algunos requerimientos son:

- Debe soportar 8 operaciones (Véase anexo)
- Debe ser parameterizable, para soportar datos de entrada de distintos tamaños.
- Debe ser desarrollada en verilog.

Algunas herramientas:

- Xilinx Vivado 2022.2
- Digilent Arty A7 (Véase anexo)

## Módulos

### Módulo ALU (module ALU)

- Se encuentra **parametrizada**, por lo que se definen los parámetros *p\_dataLength* y *p\_operatorsInputSize*, los cuales establecen el tamaño que tendrán los buses de entrada y salida.
- La placa Arty A7 cuenta con 4 switches y 4 pulsadores o botones, por lo que se decidió definir ambos buses de entrada y el bus de salida de 4 bits.
- El módulo ALU tiene 3 buses de **entrada**.
  - Un bus para el primer operando de nuestra operación: *i\_A*.
  - Un bus para el segundo operando de nuestra operación: *i\_B*.
  - Un bus para definir la operación que realiza la alu: *i\_ALUBitsControl*.
  - Tanto el *i\_A* e *i\_B* son signados.
- El módulo ALU tiene dos **salidas**
  - Un bus de 4 bits de salida, que tendrá el resultado de la operación: *o\_ALUResult*.

- Un wire que se pondrá en “alto” cuando la operación dè cero: *o\_Zero*.
- Es requerimiento que la ALU sea **puramente combinacional**, por lo que definió un bloque **always** el cual tiene como lista de sensibilización todas las entradas de la ALU.
- Cada una de las operaciones se realizan dentro del bloque **always** usando un **case**.
- Para cada caso, haciendo uso de un operador y ambas entradas se opera bit a bit, y el valor se **asigna de manera bloqueante** a un registro interno *o\_reg\_ALUResult*.
- El registro interno mediante **asignación continua** se conecta a la salida: *o\_ALUResult*.

## Módulo Clock Wizard

- Se instancia dentro del módulo top, como *clk\_wiz\_0*.
- Es un módulo IP ( de propiedad intelectual) propio de Xilinx, permite instanciar clocks de distintas frecuencias.
- El clock wizard cuenta con varios puertos de **entrada y salida**, sin embargo solo se usaron los siguientes:
  - Un wire de entrada para el clock propio de la placa: *clockCustom*,
  - Un wire de entrada para reset: *resetGral*.
  - Un wire de salida con el clock definitivo: *o\_clockWizzard*.
  - Un wire de salida que se pone en alto indicando que el clock de salida es estable y utilizable (Véase Referencias): *o\_locked*.

## Módulo Top (module topLevelAlu)

- Es el módulo encargado de **instanciar** la ALU y el Clock Wizard.
- El módulo top cuenta con las siguientes **entradas**:
  - Un bus de entrada de 4 bits conectado a los slide switches de la board. *Switch*.
  - Tres wires de entrada para cada uno de los botones. *button1*, *button2*, *button3*.
  - Un wire para el clock que ingresa de la placa: *clockCustom*
  - Un wire para reset: *resetGral*.
- Tiene un único bus de **salida** de 4 bits que se conecta a los leds de la placa.
- Además el Top cuenta con **registros internos** necesarios para almacenar el estado del bus switch en un determinado momento:
  - Un registro de 4 bits con signo para el primero datos de la suma: *reg\_dataA*.
  - Un registro de 4 bits con signo para el segundo dato de la suma: *reg\_dataB*.
  - Un registro de 4 bits para la operación: *reg\_OPCODE*.

- También cuenta con una serie de **asignaciones continuas** necesarias para la conexión de los registros internos a las entradas de ALU.
- Por último el módulo Top cuenta con un **bloque always** sensibilizado por flanco positivo de la señal de salida del clock wizard. El bloque es el encargado de detectar la pulsación de los botones y almacenar su valor mediante asignaciones no bloqueantes a los registros internos.
  - Las sentencias if dentro del always tienen en cuenta el estado e *o\_locked* al momento de realizar la asignación. Esto se hace para que las asignaciones se realicen una vez el clock se haya estabilizado.

## Constraints

Mediante el archivo de constraints se **mapean** los puertos de nuestros módulos, junto a los pines de la Artix-7. Estos últimos pines tienen distintos tipos de componentes conectados por defecto.

Entonces se mapean:

- El puerto *switch de 4 bits* del módulo *top*, con los pines A8 ,C11, C10 y A10 respectivamente. Estos pines están a su vez conectados a los **slide switches** de la board.
- El puerto *LED de 4 bits* del módulo *top*, con los pines H5, J5, T9 y T10. Estos pines están conectados a los **LEDs** de la placa.
- El puerto *resetGral* del módulo top con el pin D9, el cual se conecta al botón **BTN0**.
- El puerto *button1* del módulo top con el pin C9, el cual se conecta al botón **BTN1**.
- El puerto *button2* del módulo top con el pin B9, el cual se conecta al botón **BTN2**.
- El puerto *button3* del módulo top con el pin B8, el cual se conecta al botón **BTN3**.
- El puerto *clockCustom* del módulo top con el pin E3, el cual se conecta al **oscilador** de 100Mhz propio de la placa.

Véase Figura 1 y Figura 2 para mayor comprensión.

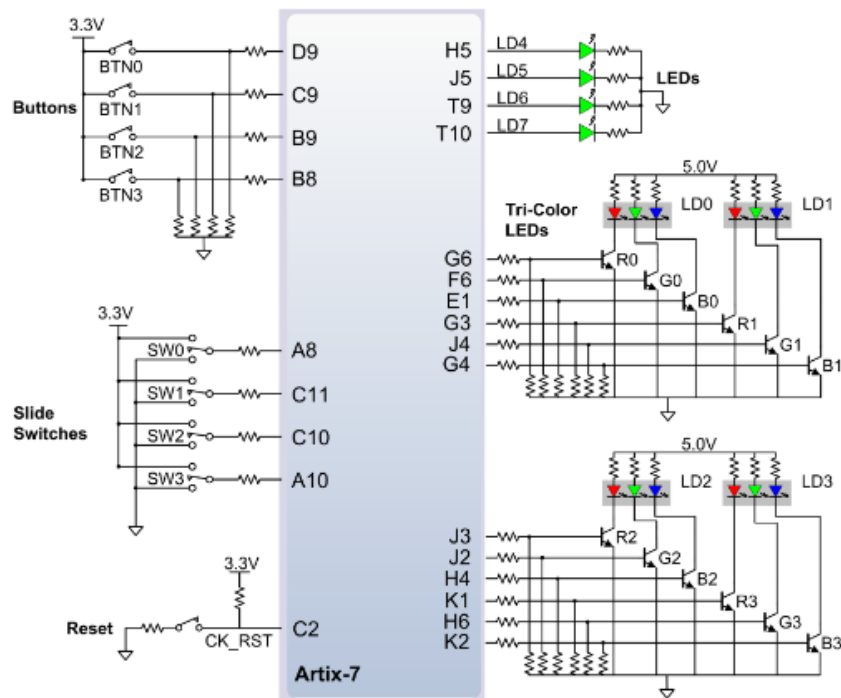


Figura 1. Pines de algunos de los componentes de Arty A7

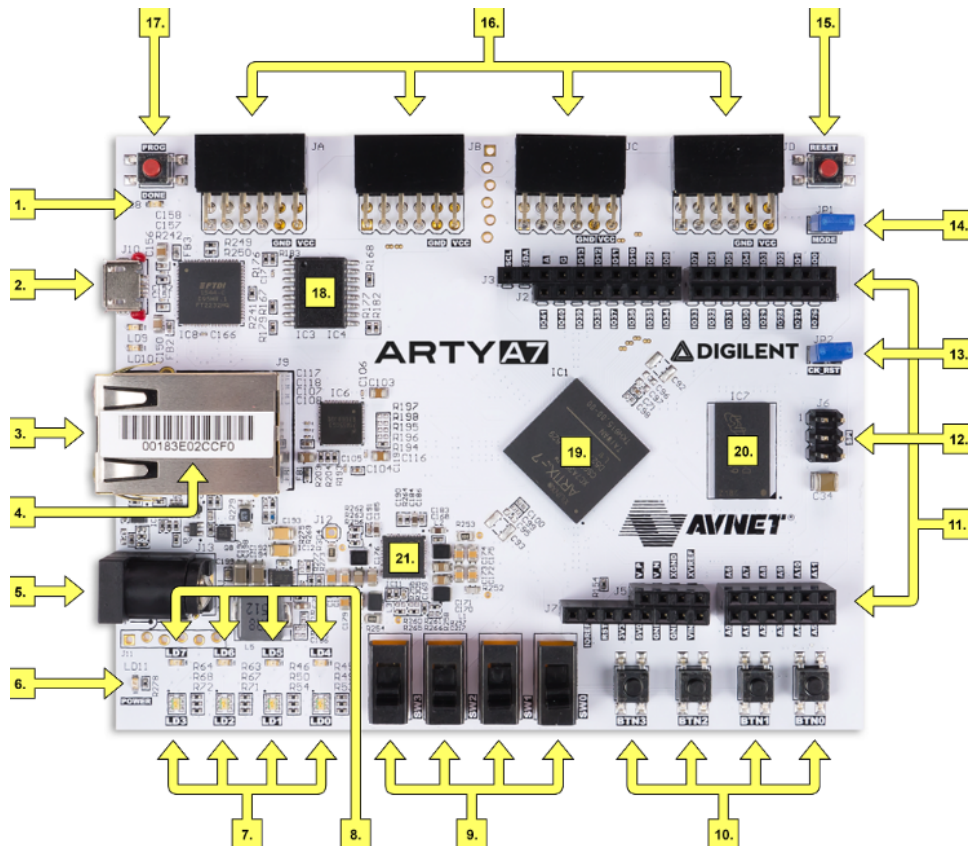


Figura 2. Componentes de Arty A7

## Simulación Lógica

La simulación lógica (Behavioral Simulation) de la ALU:

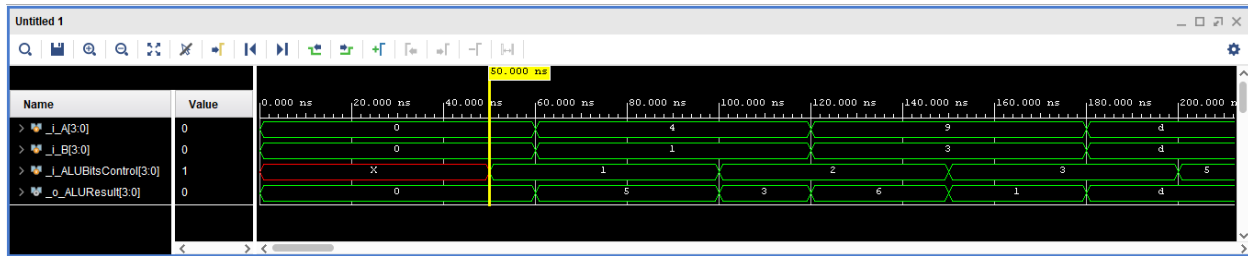


Figura 3. Simulación lógica usando el testbench del módulo ALU.

- Se observa que los primeros valores que se cargan son cuatro y uno como operandos y la operación definida por el número uno definida como suma. Además se observa el valor de resultado (5) cargado de manera instantánea a los 60 us.

La simulación lógica del top:

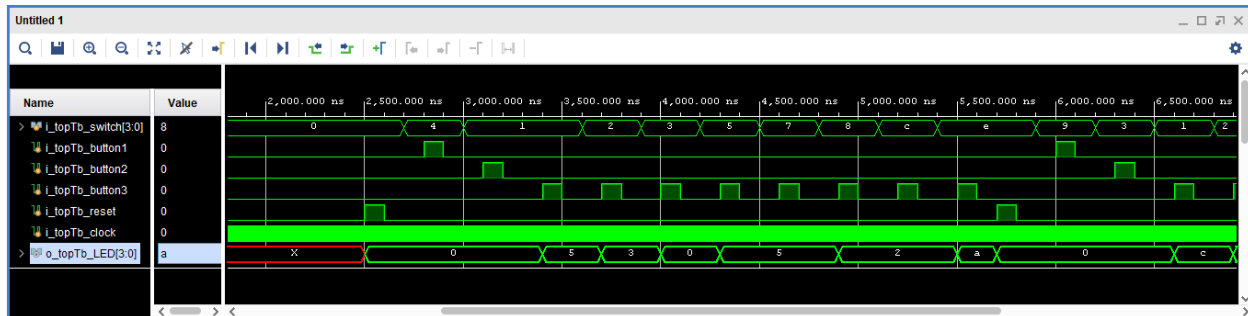


Figura 4. Simulación lógica usando el testbench del módulo Top.

Se observa la siguiente secuencia:

- Se presiona el BTN0 para resetear.
- Se carga el valor de los switches en 0100b o 4d y se pulsa el BTN1. Es decir el primero operado es igual a cuatro.
- Se carga el valor de los switches en 0001b o 1d y se pulsa el BTN2. Es decir el segundo operando es igual a uno.

- Se mantiene el valor de los switches en 0100b o 1d y se pulsa el BTN3. Es decir se selecciona la operación de suma y aparece el valor del resultado (5) en los leds.
- Luego se va iterando entre las distintas operaciones que tiene la alu.
- Por último se resetea de nuevo.

## Esquemáticos RTL

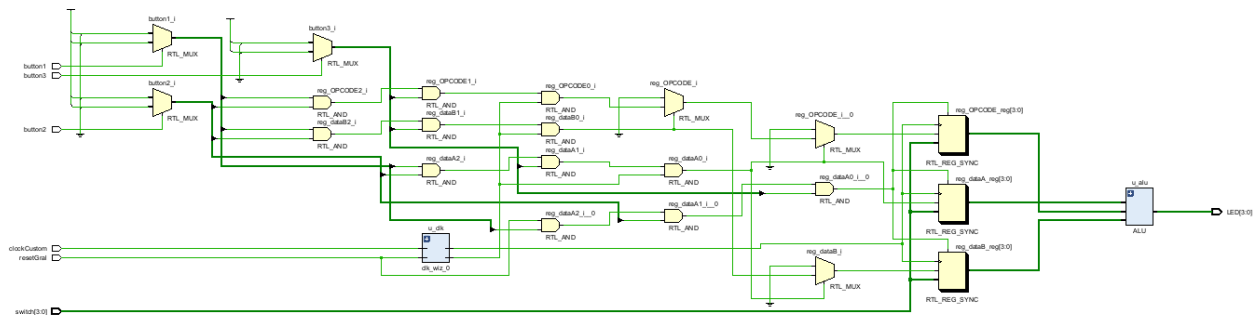


Figura 4. Esquemático RTL del módulo Top.

- Se observa la instancia del módulo ALU y del clock wizard.

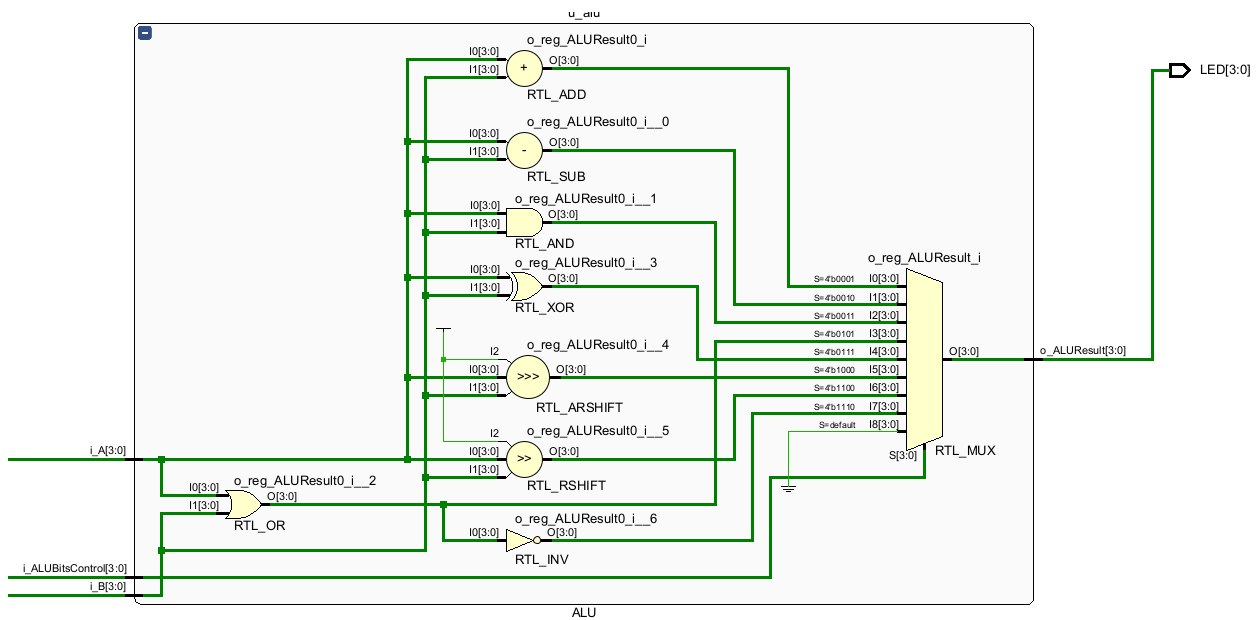


Figura 5. Esquemático RTL del módulo ALU.

- Se observa que la u\_alu es puramente combinacional, no contiene registros en ella.

## Síntesis

Una vez realizada la síntesis, se observa que se han sintetizado 27 Lookup tables, 12 Flip Flip y 12 bloques de I/O.

Resource	Estimation	Available	Utilization %
LUT	27	20800	0.13
FF	12	41600	0.03
IO	12	210	5.71

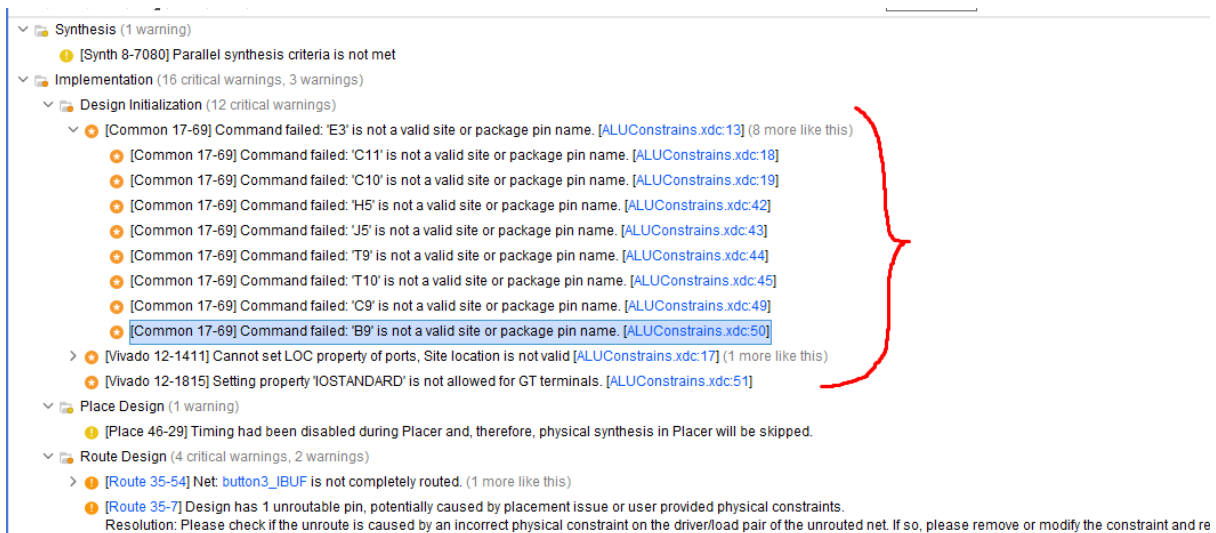
Además Vivado nos realiza una estimación del consumo de la placa y su temperatura.

Power		Summary   On-Chip
Total On-Chip Power:	0.17 W	
Junction Temperature:	25.8 °C	
Thermal Margin:	74.2 °C (15.4 W)	
Effective $\theta_{JA}$ :	4.8 °C/W	
Power supplied to off-chip devices:	0 W	
Confidence level:	Low	

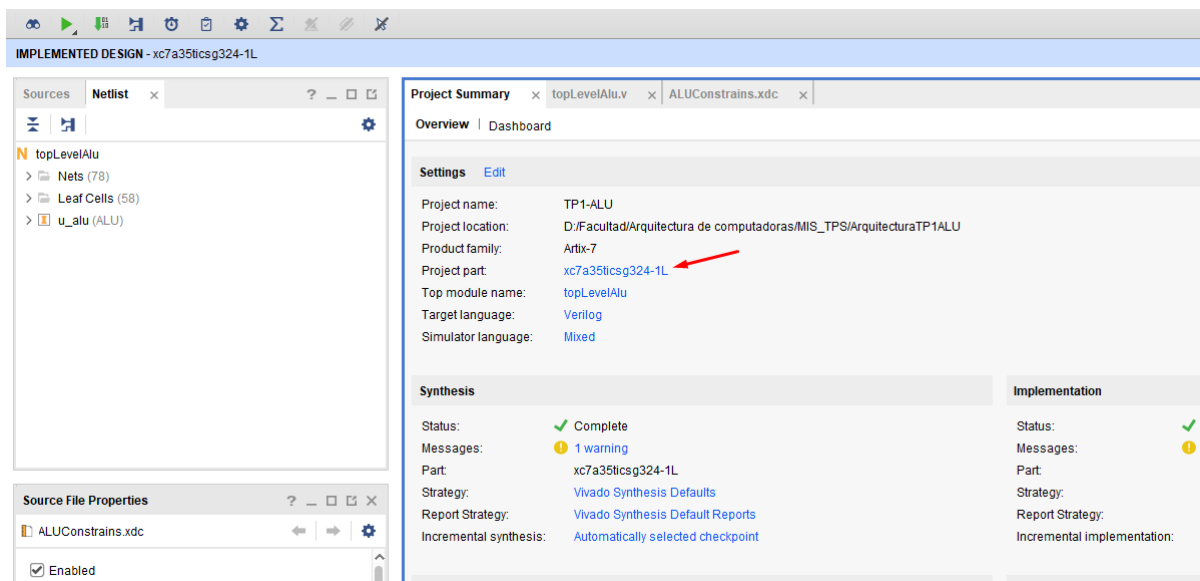
## Implementación

A la hora de sintetizar, o mejor dicho, después de la sintetización surgieron problemas o warnings como el de la siguiente imagen:



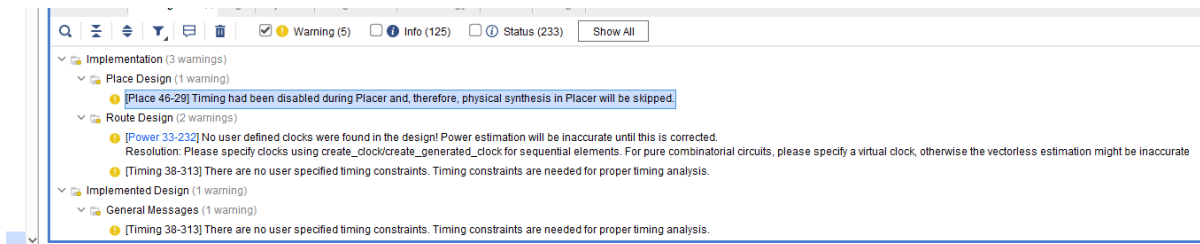


Esto se debió fue se tenía configurado un modelo de placa erroneo.



Para solucionarlo, se hizo click en donde está señalado con la flecha roja y se buscó el modelo de placa correspondiente.

Sin embargo, algunos warnings persistían.

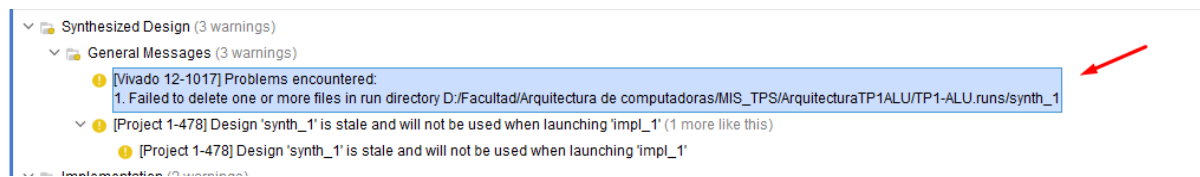


⚠ [Timing 38-313] There are no user specified timing constraints. Timing constraints are needed for proper timing analysis.

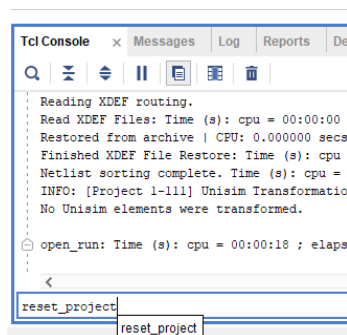
El error de la imagen anterior se soluciona modificando el archivo de constraints


```
...
set_property -dict { PACKAGE_PIN E3 IOSTANDARD LVCMOS33 } [get_ports {
clockCustom }];
create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5}
[get_ports {clockCustom}];
...
```

Apareció un nuevo error después del último cambio.



Hice varias cosas puse en la consola reset\_project





Por último se eliminó el modo incremental, Y por último reinicie VIVADO con eso se fueron esos 2 warnings.

## Referencia

- [ 1 ] Arty A7 Manual. <https://digilent.com/reference/programmable-logic/arty-a7/reference-manual>
- [ 2 ] Clock Wizard Xilinx. <https://docs.xilinx.com/r/en-US/pg065-clk-wiz/Auto-Primitive>
- [ 3 ] Repositorio del Proyecto. <https://github.com/RugenKunkka/ArquitecturaTP1ALU>.

## Referencia

**Two's complement 4 bit  
integer values**

<b>Two's complement</b>	<b>Decimal</b>
0111	7.
0110	6.
0101	5.
0100	4.
0011	3.
0010	2.
0001	1.
0000	0.
1111	-1.
1110	-2.
1101	-3.
1100	-4.
1011	-5.
1010	-6.
1001	-7.
1000	-8.