

# **Cómo implementar un PLL en Lattice Radiant**

Creador: David Rubio G.

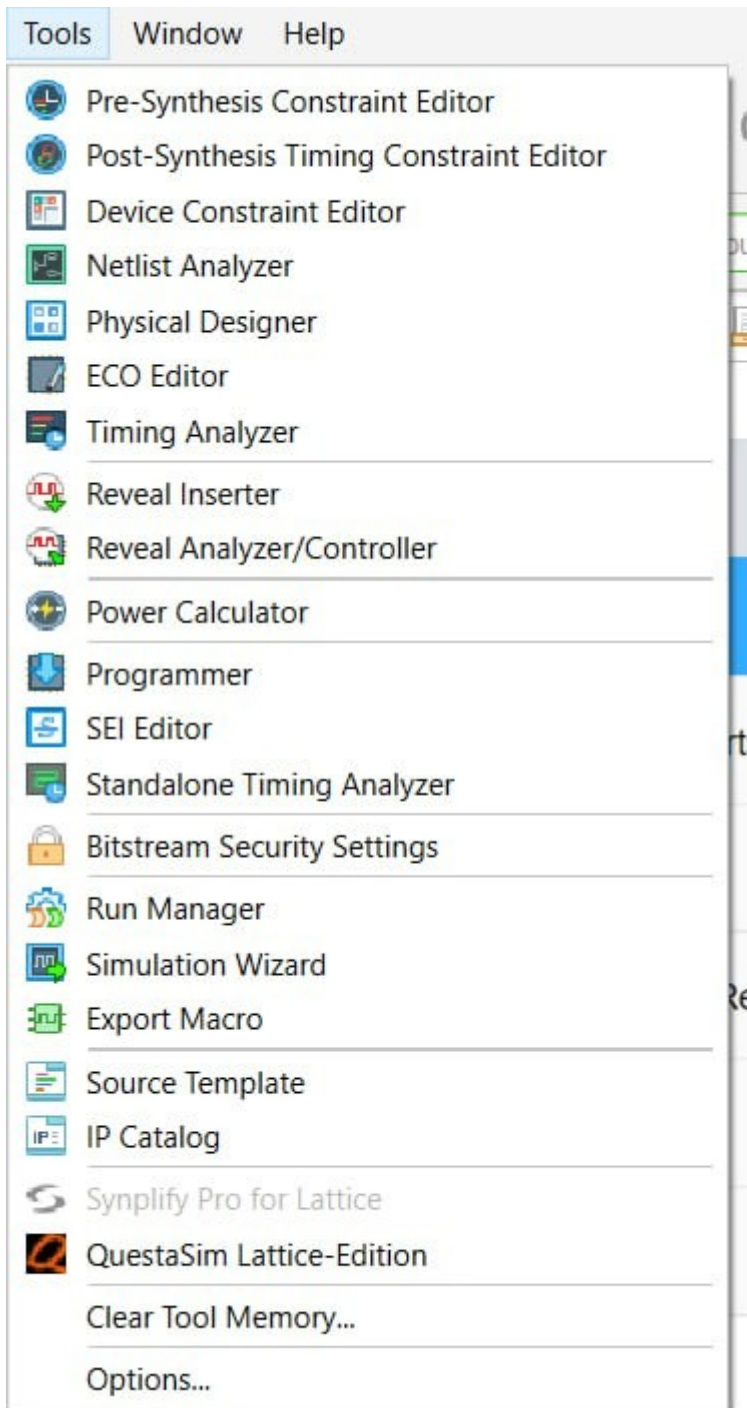
Entrada: <https://soceame.wordpress.com/2024/12/19/como-implementar-un-pll-en-lattice-radiant/>

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

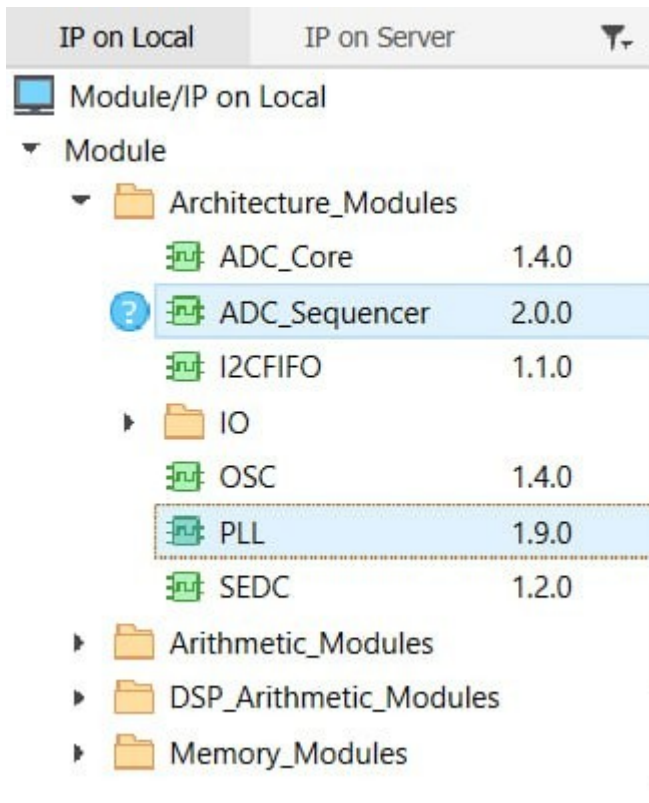
GitHub: <https://github.com/DRubioG>

Fecha última modificación: 24/02/2025

Para implementar un PLL en Lattice Radiant, se tiene que ir a la opción *Tools* y a la opción *IP Catalog*.

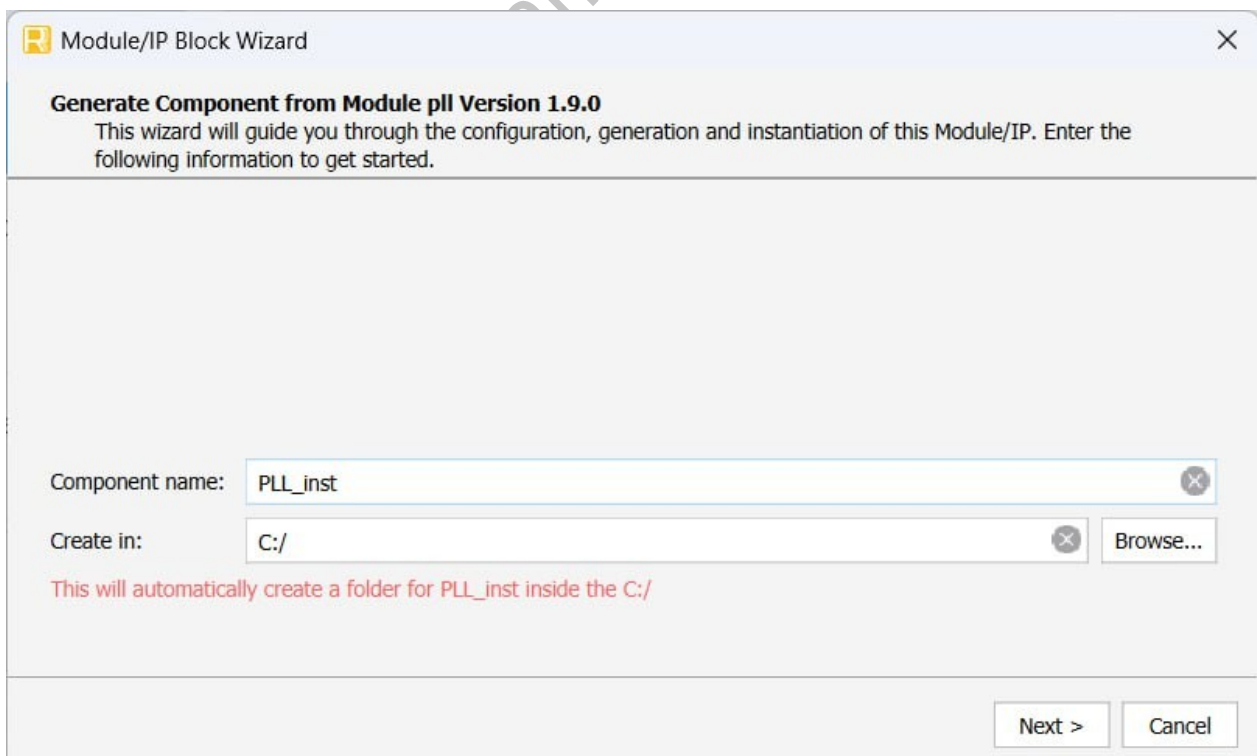


Después tienes que ir a la opción *IP on Local*, allí aparecerá la opción de añadir un PLL.

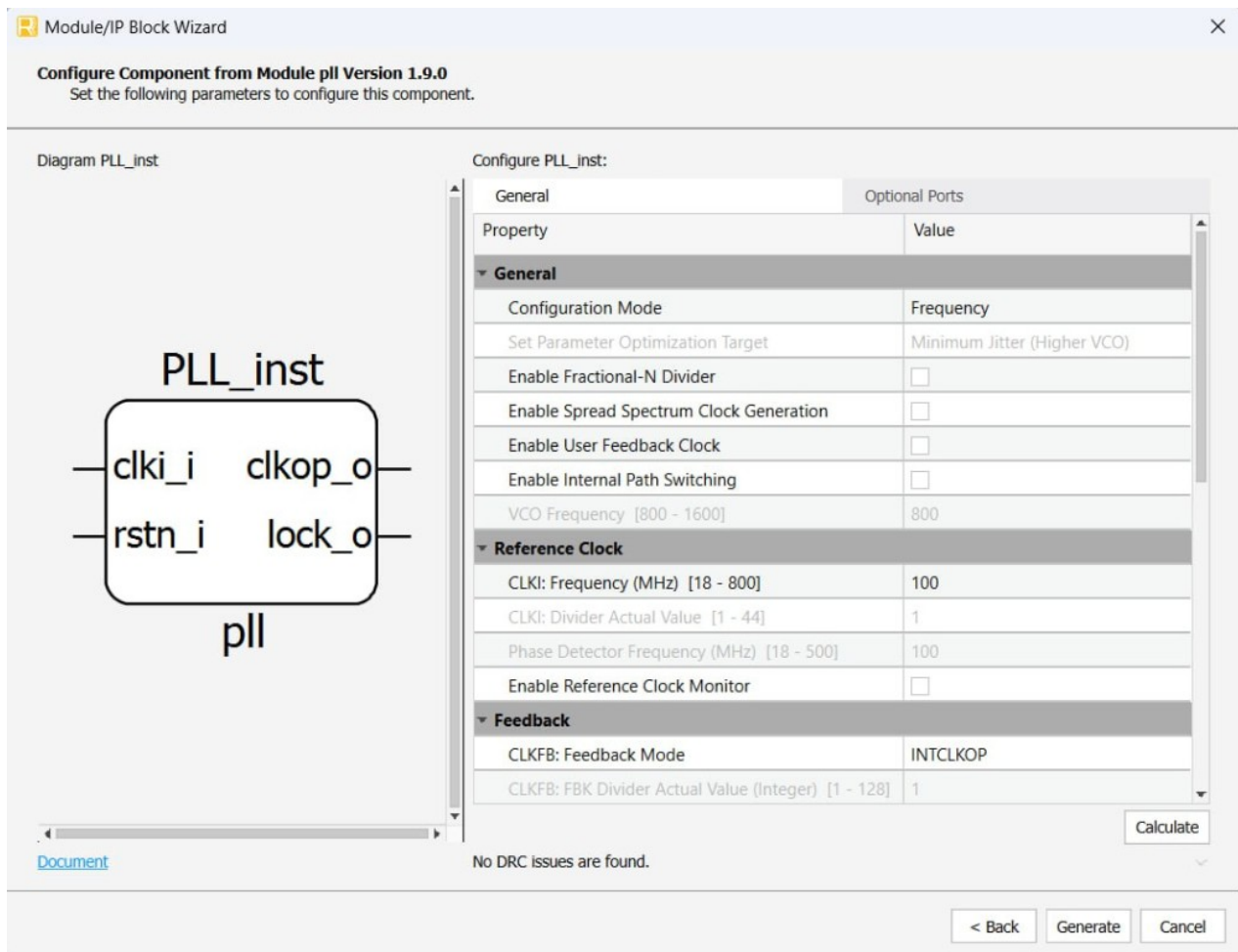


Al darle, se nos abre una pestaña como la siguiente. En ella nos pide el nombre que va a tener el PLL y la ubicación donde se nos va a guardar.

**NOTA:** lo que pone en rojo abajo no es un error es una información sobre el guardado del bloque.



La siguiente pestaña que se no abre es la que nos permite configurar el PLL. En esta pestaña nos pregunta la frecuencia del reloj de entrada (entre 18 y 800 MHz).



En esta misma pestaña se pueden configurar varios relojes de salida, para ello primero se marca la casilla *CLKOS2 Enable* y después se nos abren las opciones de ese reloj.

**NOTA:** Todos los relojes secundarios que se creen terminan por la letra 's', y luego '\_o'.

Configure PLL\_inst:

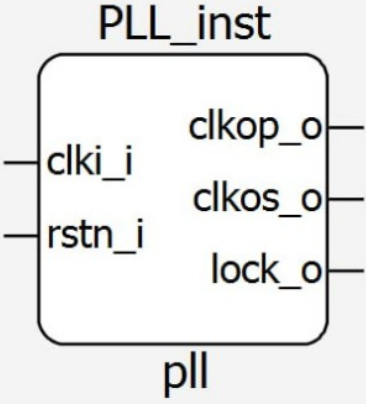
General		Optional Ports
Property	Value	
CLKOP: Enable Trim for CLKOP	<input type="checkbox"/>	
<b>▼ Secondary Clock Output</b>		
CLKOS: Enable	<input checked="" type="checkbox"/>	
CLKOS: Bypass	<input type="checkbox"/>	
CLKOS: Frequency Desired Value (MHz) [6.25 - 800]	100	
CLKOS: Divider Actual Value [1 - 128]	8	
CLKOS Tolerance (%)	0.0	
CLKOS: ERROR (PPM)	0	
CLKOS: Static Phase Shift (Degrees)	0	
CLKOS: Enable Trim for CLKOS	<input type="checkbox"/>	
<b>▼ Secondary Clock Output (2)</b>		
CLKOS2: Enable	<input type="checkbox"/>	
<b>▼ Secondary Clock Output (3)</b>		
CLKOS3: Enable	<input type="checkbox"/>	
<b>▼ Secondary Clock Output (4)</b>		
CLKOS4: Enable	<input type="checkbox"/>	
<b>▼ Secondary Clock Output (5)</b>		
CLKOS5: Enable	<input type="checkbox"/>	

Calculate

No DRC issues are found.

En la pestaña de *Optional Ports* se puede configurar los puertos opcionales, como el puerto de reset del PLL o el puerto de Lock.

Diagram PLL\_inst



Configure PLL\_inst:

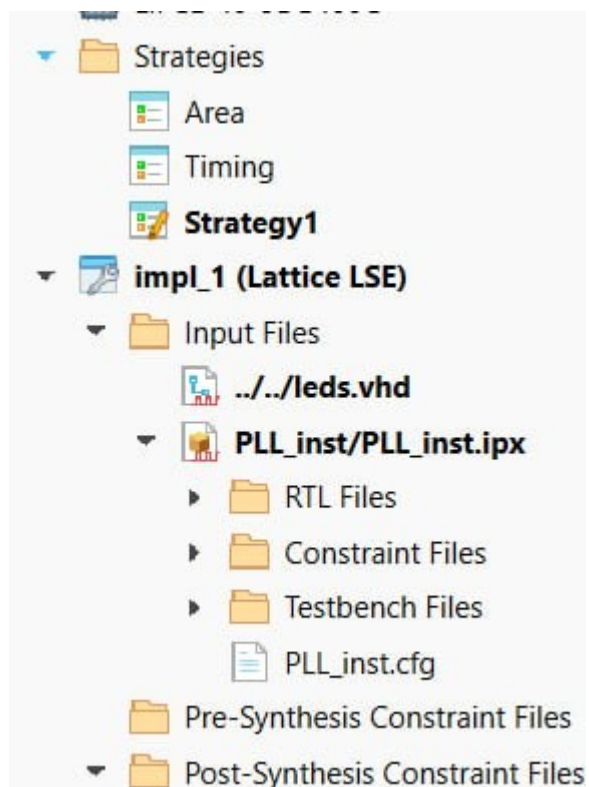
General		Optional Ports
Property	Value	
<b>Reference Clock I/O Pin</b>		
Set I/O Pin for PLL Reference Clock	<input type="checkbox"/>	
<b>Dynamic Phase Control Ports</b>		
Enable Dynamic Phase Ports	<input type="checkbox"/>	
<b>Clock Enable Ports</b>		
CLKOS Enable Port	<input type="checkbox"/>	
<b>PLL Reset</b>		
Provide PLL Reset	<input checked="" type="checkbox"/>	
<b>PLL Lock</b>		
Provide PLL Lock Signal	<input checked="" type="checkbox"/>	
PLL Lock is Sticky	<input type="checkbox"/>	
<b>Register Interface</b>		
Select Register Interface	None	
<b>Power Mode Settings</b>		
Enable Legacy Mode	<input type="checkbox"/>	
Enable Powerdown Mode	<input type="checkbox"/>	

Document

No DRC issues are found.

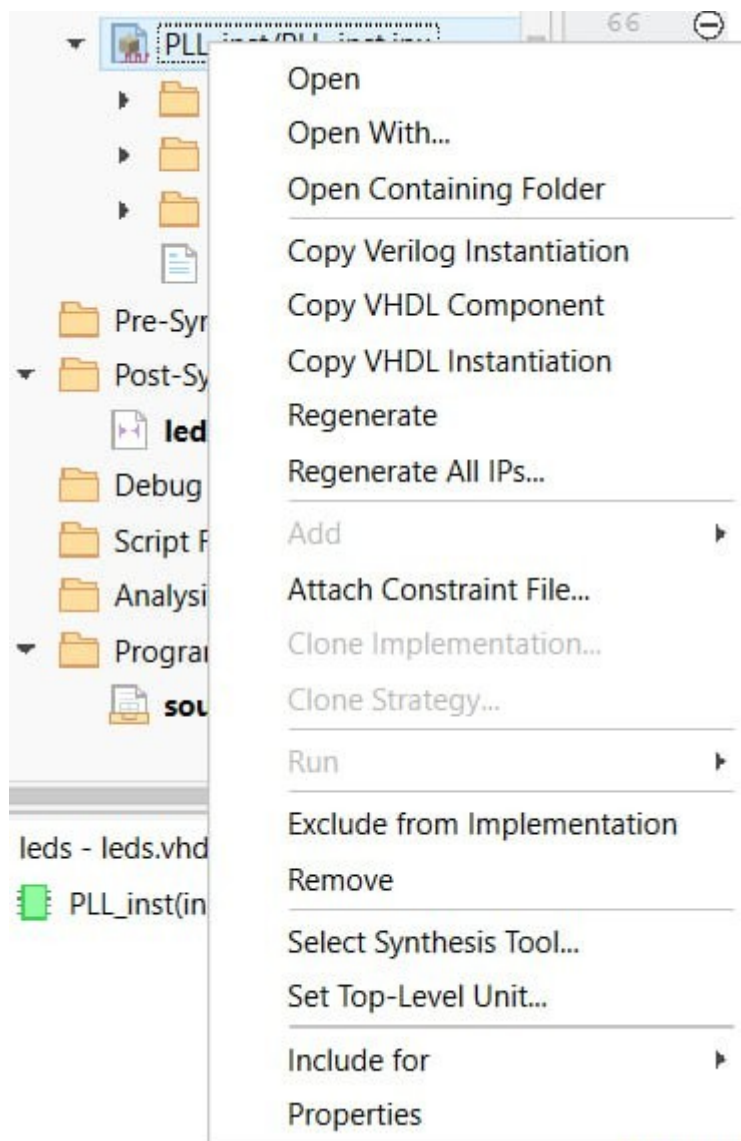
Calculate

Después en la estructura del proyecto se puede ver que se ha añadido un bloque PLL, ahora si buscas el fichero de instanciación en las carpetas que se han creado, verás que no aparece ningún .vhd o .v, eso es porque se instancia de otra manera.



Para instanciar el PLL, se da clic derecho al bloque PLL, y aquí aparecen varias opciones: *Copy Verilog Instantiation* (para Verilog), y *Copy VHDL Component* y *Copy VHDL Instantiation* (para VHDL).





Al darle clic se guarda en el portapapeles la instanciación, ahora solo tienes que pegarla en el proyecto.



```
-- Verilog
PLL_inst __(.clki_i( ),
            .clkop_o( ));


-- VHDL
-- component
component PLL_inst is
    port(
        clki_i: in std_logic;
        clkop_o: out std_logic
    );
end component;


-- Instantiation
__: PLL_inst port map(
    clki_i=>,
    clkop_o=>
);
```

**NOTA:** si quieres buscar los ficheros de instanciación tienes que ir a la siguiente ruta:  
<proyecto>\<nombre PLL>\misc

Aquí aparecerán dos ficheros, uno en VHDL y otro en Verilog.

Nombre

 PLL\_inst\_tmpl.v

 PLL\_inst\_tmpl.vhd

Si abres el de VHDL verás que la instanciación es la misma que se hace por el otro método.

```
component PLL_inst is
    port(
        clki_i: in std_logic;
        clkop_o: out std_logic
    );
end component;

__: PLL_inst port map(
    clki_i=>,
    clkop_o=>
);
```

Si pegamos la instanciación en nuestro código, ya podemos sintetizar.

```
component PLL_inst is
  port(
    clki_i: in std_logic;
    clkop_o: out std_logic
  );
end component;

signal cont : integer range 0 to 25000000;
signal cont1_seg : integer := 25000000;
signal clk_i : std_logic;
begin

inst_PLL_inst : PLL_inst port map(
  clki_i=> clk,
  clkop_o=> clk_i
);
```

Al sintetizar nuestro modelo, podemos ver que en la síntesis aparece un bloque que se llama igual que el bloque que hemos creado.

