Appendix L: Using Phase-Locked Loops with VHDL

This tutorial shows how to instantiate PLLs in FPGAs when using Vivado or Quartus Prime. In both cases, the PLL's default ports are *clock_in*, *clock_out* (one or more), *reset*, and *locked* (of the last two, at least *locked* can be disabled). The circuit of figure L.1 will be used as an example (see also complete application in example 17.7).

I. Instantiating PLLs with Vivado

1) Start Vivado, create a project, and enter a VHDL code for the circuit of figure L.1, without the PLL yet (i.e., without lines 14–18 and 22).

2) Under **Project Manager**, select **IP Catalog > FPGA Features and Design > Clocking > Clock Wizard**, which opens the window of figure L.2. Pay particular attention to the five arrows:

- On the left, notice the PLL's default ports (for clock in, clock out, reset, and locked).
- In the center, mark PLL.
- At the top, enter a name for the component (say, *my_pll*).
- At the bottom, enter name (*clk_50MHz*) and frequency (50 MHz) for the input clock.

2) Open the second tab (Output Clocks) of figure L.2, leading to figure L.3. Note the four arrows:

- Near the top, enter name (*clk_120MHz*) and frequency (120 MHz) for the output clock; observe that not any speed is allowed.
- At the bottom, unmark the *reset* and *locked* ports.
- On the left, notice that now the only active ports are those for clock in and clock out.
- Click OK.

3) Returning to your design, observe in the project **IP Sources** list (figure L.4) that a component with instantiation template called *my_pll.vho* was created. Click on it, and the code on the right of figure L.4 will be displayed, even having indications on where to cut to get just the right portion of code for component declaration.

```
1  library ieee;
2  use ieee.std_logic_1164.all;
3
4  entity using_PLLs is
5    port (
6      clk_50MHz: in std_logic;
7      din: in std_logic;
8      dout: out std_logic);
9  end entity;
10
11 architecture example of using_PLLs is
12   signal clk_120MHz: std_logic;
13
14   component my_pll is
15     port (
16       clk_120MHz: out std_logic;
17       clk_50MHz: in std_logic);
18     end component;
19
20 begin
21
22   comp: my_pll port map (clk_50MHz => clk_50MHz, clk_120MHz => clk_120MHz);
23
24   process (clk_120MHz)
25   begin
26     if rising_edge(clk_120MHz) then
27       dout <= din;
28     end if;
29   end process;
30
31 end architecture;
```

Figure L.1

4) Copy this component declaration to your project (lines 14–18 in the code shown above) and make the instantiation of that component (line 22).

Result: The RTL view produced by Vivado after compilation is shown in figure L.5.

II. Instantiating PLLs with Quartus Prime

1) Start Quartus Prime, create a project, and enter a VHDL code for the circuit of figure L.1 but without the PLL yet (i.e., without lines 14–18 and 22).

2) Access **Tools > IP Catalog > Library > Basic Functions > Clocks, PLLs, and Resets > PLL > Altera PLL**, which leads to figure L.6.

- Mark VHDL.
- Enter a name for your component (say, *my_pll*).
- Click OK, which opens the MegaWizard (figure L.7).

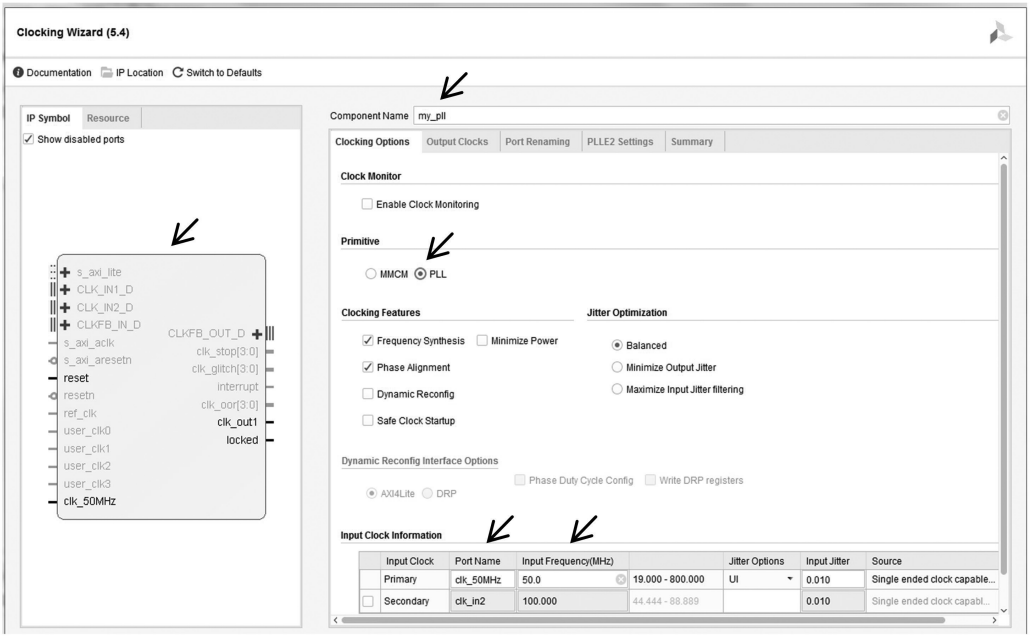


Figure L.2

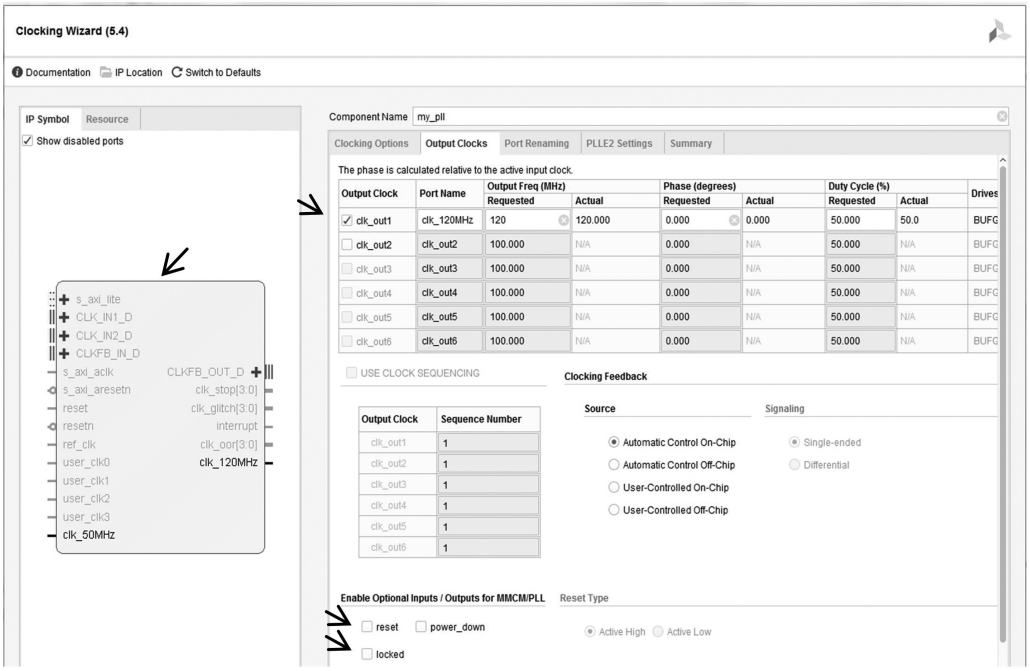
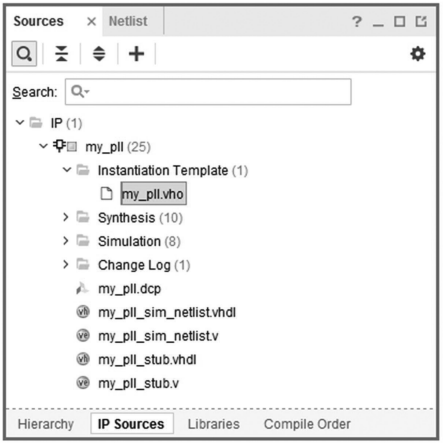


Figure L.3




```
----Begin Cut here for COMPONENT Declaration
component my_pll
port
 (-- Clock in ports
  -- Clock out ports
  clk_120MHz          : out    std_logic;
  clk_50MHz           : in     std_logic
 );
end component;
-- COMP TAG END -- End COMPONENT Declaration
```

Figure L.4

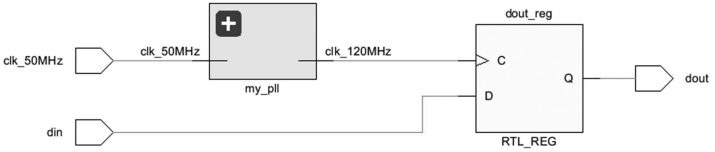


Figure L.5

Figure L.6

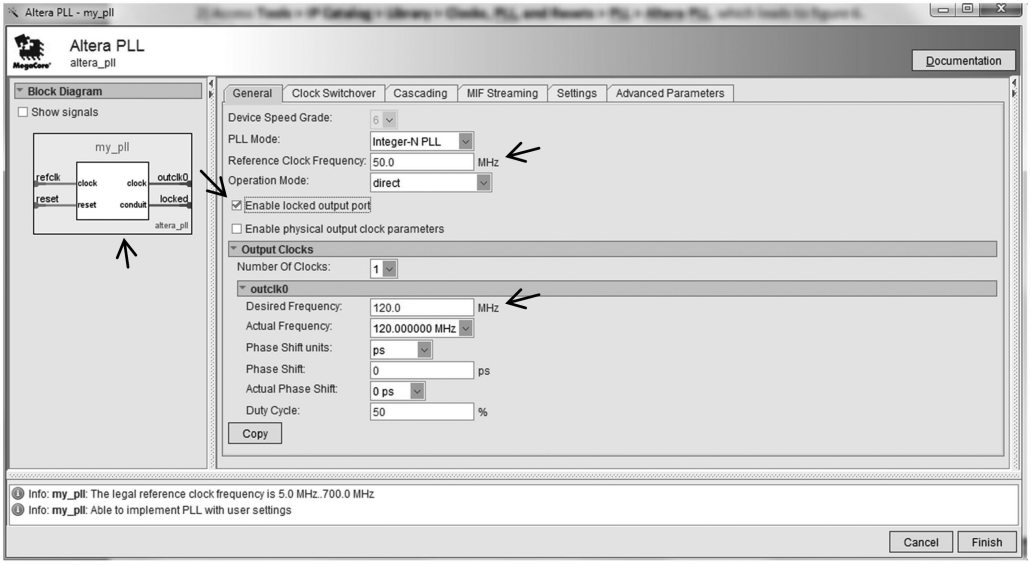


Figure L.7

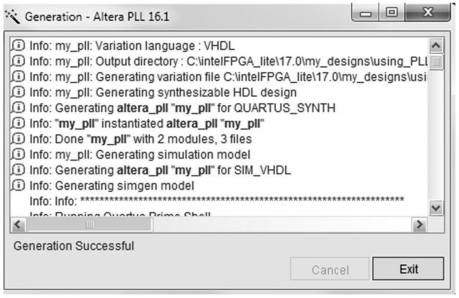


Figure L.8

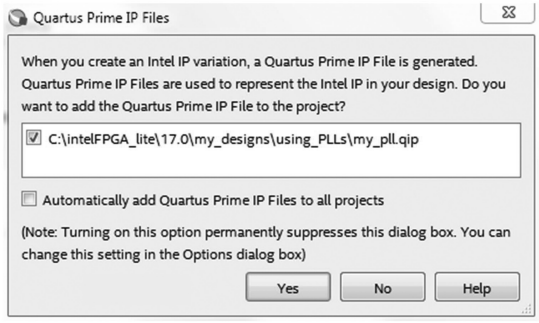


Figure L.9

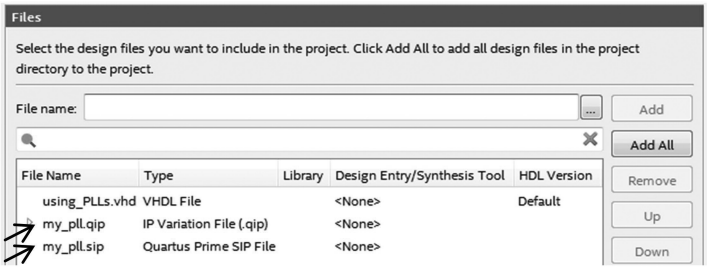


Figure L.10

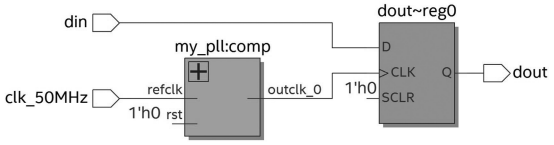


Figure L.11

3) In figure L.7, pay particular attention to the four arrows:

- On the left, notice the PLL's default ports (clock in, clock out, reset, and locked).
- Near the top, enter the frequency (50 MHz) for the reference (input) clock.
- In the center, enter the desired frequency (120 MHz). The frequency that will be actually generated is displayed right below it (not all speeds are possible). Choosing Fractional-N PLL instead of Integer-N PLL gives additional options.
- Notice that the *locked* port can be disabled (as we did for Vivado above).
- Click Finish.

4) Figure L.8 is exhibited when done. Click Exit, which leads to figure L.9. Click Yes.

5) Go to **Project > Add/Remove Files in Project** and confirm that the *my_pll.qip* and *my_pll.sip* files were added automatically to the design, as shown in figure L.10.

6) Open, *for editing*, the file *my_pll.vhd*. Copy its entity to your design to be used as a component. If the *rst* and *locked* ports were not disabled, lines 14–22 of the previous code would become lines 14–24 below. Note that, in this example, *rst* and *locked* were disabled during instantiation (line 24), with the former set to zero and the latter left unconnected.

```
14      component my_pll is
15         port (
16            refclk: in std_logic := '0';      --input clock
17            rst: in std_logic := '0';         --asynchronous reset
```

```
18          outclk_0: out std_logic;          --output clock
19          locked: out std_logic);           --PLL has locked
20      end component;
21
22  begin
23
24      comp: my_pll port map (clk_50MHz, '0', clk_120MHz, open);
```

Result: The RTL view produced by Quartus Prime is shown in figure L.11.