

PDS 下 PLL 实验

1 文档简介

很多初学者看到板上只有一个 50MHz 时钟输入的时候都产生疑惑，时钟怎么才 50MHz？如果要工作在 100MHz、150MHz 怎么办？在很多 FPGA 芯片内部都集成了 PLL，其他厂商可能不叫 PLL，但是也有类似的功能模块，通过 PLL 可以倍频分频，产生其他很多时钟。本实验通过调用 PLL ip core 来学习 PLL 的使用方法。

2 实验环境

- Windows 10 64 位
- PDS
- AXP100 开发板
- 示波器

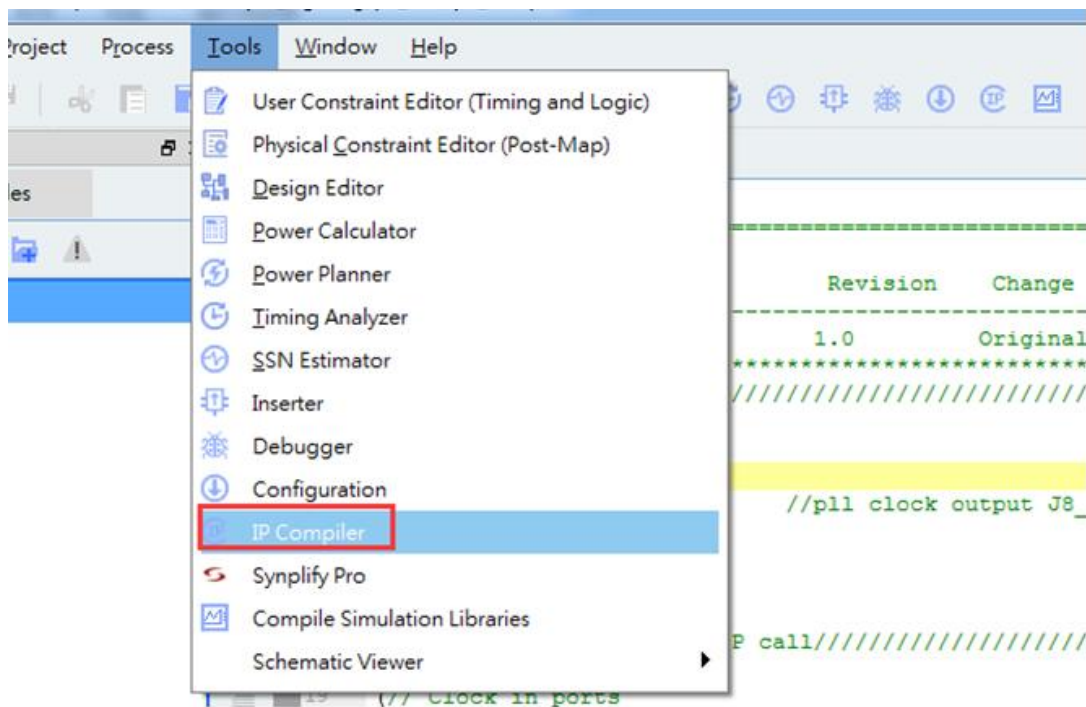
3 实验原理

PLL(phase-locked loop)，即锁相环。是 FPGA 中的重要资源。由于一个复杂的 FPGA 系统往往需要多个不同频率，相位的时钟信号。所以，一个 FPGA 芯片中 PLL 的数量是衡量 FPGA 芯片能力的重要指标。FPGA 的设计中，时钟系统的 FPGA 高速的设计极其重要，一个低抖动，低延迟的系统时钟会增加 FPGA 设计的成功率。

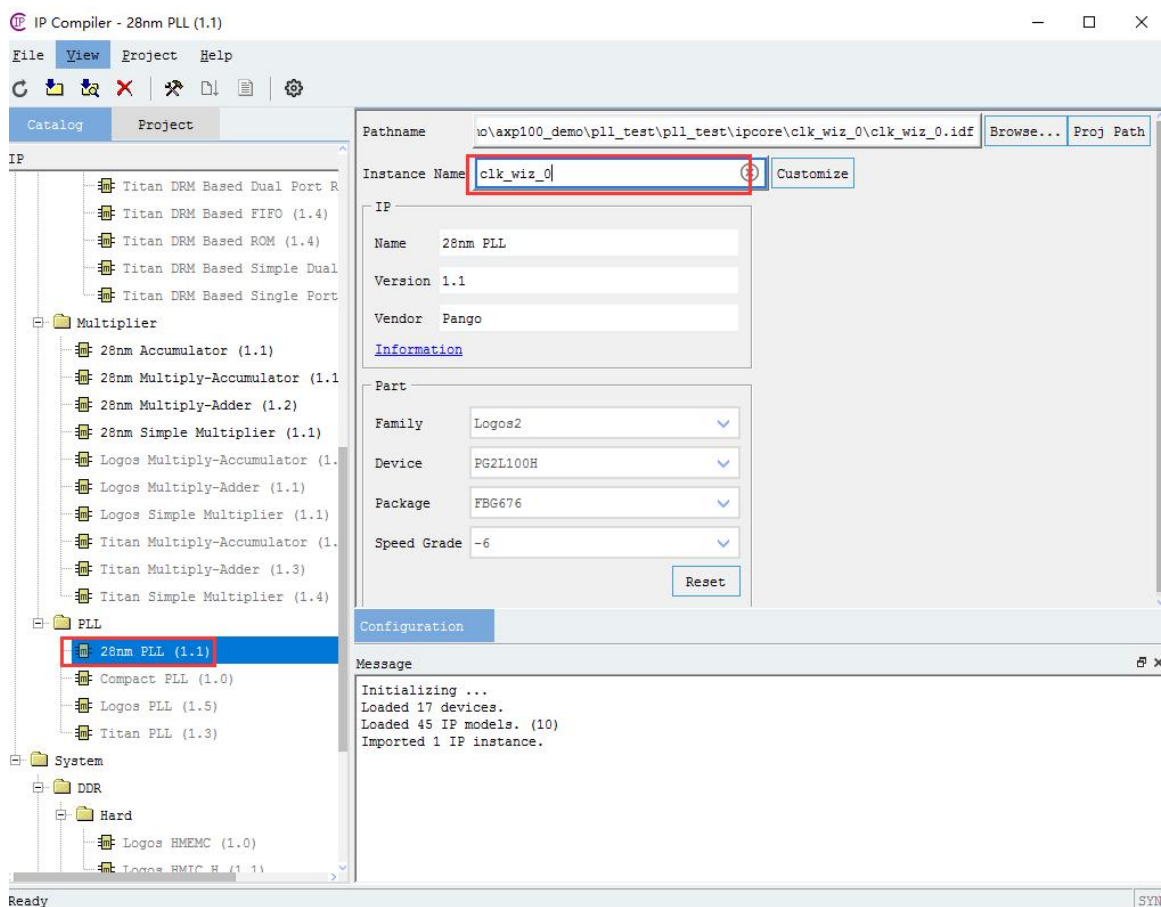
本实验将通过使用 PLL，输出一个方波到开发板上的扩展口 J8 的 PIN3 脚，来给大家演示使用 PLL 的方法。

➤ 数字锁相环(PLL)

PLL 内部的功能框图如下图所示:Logos2 PLL 主要由鉴频鉴相器 (PFD,Phase Frequency Detector)、环路滤波器 (LF,Loop Filter) 和压控振荡器 (VCO,Voltage Controlled Oscillator) 等组



2. 再在弹出的界面里选择 PLL 下的 28nm PLL，再对 PLL 取名 clk_wiz_0，然后单击 Customize。



3. 在 Basic Configurations 界面里，勾选复位管脚使能"Enable Port rst"，输入的时钟频率为 200MHz。输出选择 clk_out0~clk_out3 四个时钟的输出，频率分别为 200MHz, 100MHz, 50MHz, 25MHz。这里还可以设置时钟输出的相位，我们不做设置，保留默认相位，点击 OK 完成，

PLL Mode Configurations

Mode Select: GPLL

Public Configurations

Feedback Clock Mode: Internal Feedback

Feedback from: CLKOUTIF

Input Clock clk_in Frequency [10:800]: 200.0000 MHz

SSC Mode: DISABLE

SSC Frequency [25:250]: 25 KHz

☒ Enable Port rst

☐ Enable clk_in Dynamic Select

☐ Enable Dynamic Phase Bus

☐ Enable APB Bus

☐ Enable Port pll_pwd

Clockoutphy Configurations (Only use for DDRPHY)

☐ Enable clkoutphy

Clockout0 Configurations

☒ Enable clkout0

☐ Enable Clock Gate for clkout0

☐ Enable clkout0 DPS

☐ Enable clkout0_n

Desired Frequency: 200.0000 MHz

Desired Phase Shift: 0.0000 degrees

Desired Duty Cycle: 50.0000 %

Actual Frequency: 200.00000000 MHz

Actual Phase Shift: 0.00000000 degree

Actual Duty Cycle: 50.00000000 %

Clockout1 Configurations

☒ Enable clkout1

☐ Enable Clock Gate for clkout1

☐ Enable clkout1 DPS

☐ Enable clkout1_n

Desired Frequency: 100.0000 MHz

Desired Phase Shift: 0.0000 degrees

Desired Duty Cycle: 50.0000 %

Actual Frequency: 100.00000000 MHz

Actual Phase Shift: 0.00000000 degree

Actual Duty Cycle: 50.00000000 %

Clockout2 Configurations

☒ Enable clkout2

☐ Enable Clock Gate for clkout2

☐ Enable clkout2 DPS

☐ Enable clkout2_n

Desired Frequency: 50.0000 MHz

Desired Phase Shift: 0.0000 degrees

Desired Duty Cycle: 50.0000 %

Actual Frequency: 50.00000000 MHz

Actual Phase Shift: 0.00000000 degree

Actual Duty Cycle: 50.00000000 %

Clockout3 Configurations

☒ Enable clkout3

☐ Enable Clock Gate for clkout3

☐ Enable clkout3 DPS

☐ Enable clkout3_n

Desired Frequency: 25.0000 MHz

Desired Phase Shift: 0.0000 degrees

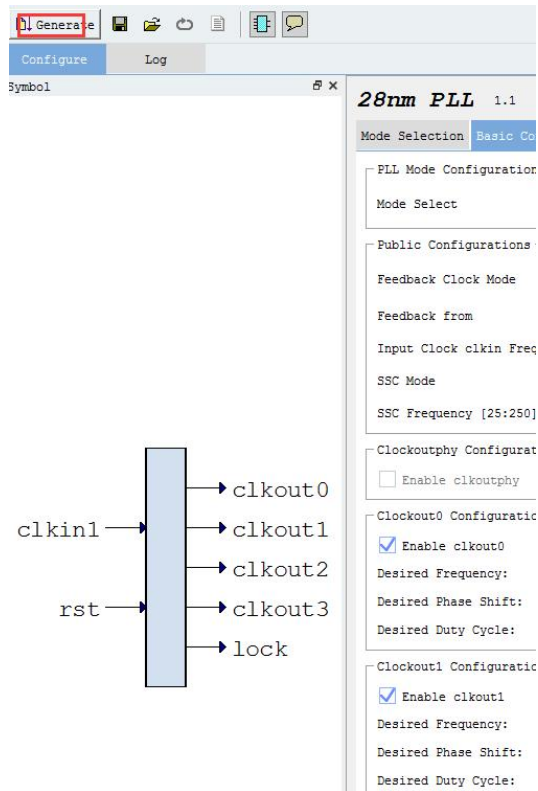
Desired Duty Cycle: 50.0000 %

Actual Frequency: 25.00000000 MHz

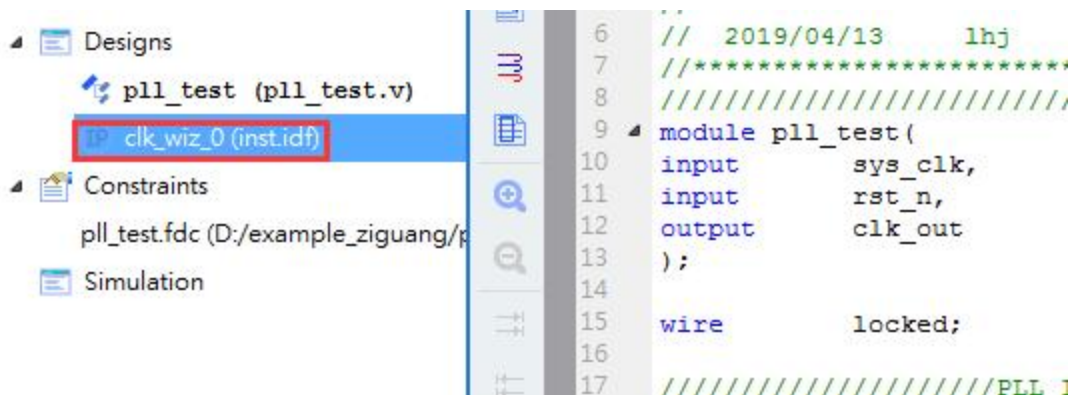
Actual Phase Shift: 0.00000000 degree

Actual Duty Cycle: 50.00000000 %

5. 然后点击 Generate 按钮生成 PLL IP 的设计文件。



6. 这时一个 clk_wiz_0 IP 会自动添加到我们的 pll_test 项目中, 用户可以双击它来修改这个 IP 的配置。



7. 我们再来编写一个顶层设计文件来实例化这个 PLL IP, 编写 pll_test.v 代码如下。

```

`timescale 1ns / 1ps
//=====
// Revision History:
// Date      By      Revision  Change Description
//-----
// 2019/04/13 lhj      1.0      Original
//*****
//=====
module pll_test(

```

```

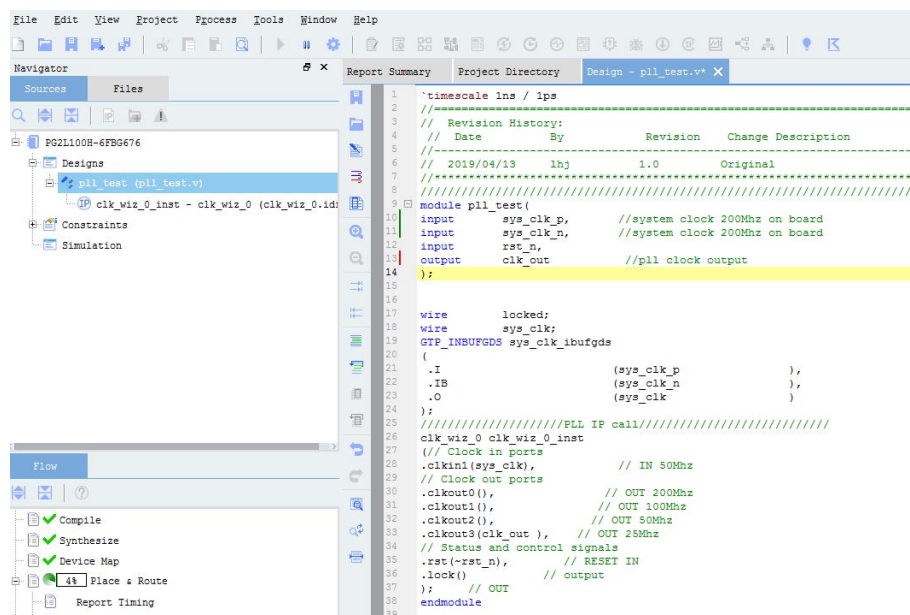
input  sys_clk_p,    //system clock 200Mhz on board
input  sys_clk_n,    //system clock 200Mhz on board
input  rst_n,
output  clk_out       //pll clock output
);

wire    locked;
wire    sys_clk;
GTP_INBUFGDS sys_clk_ibufgds
(
    .I          (sys_clk_p      ),
    .IB         (sys_clk_n      ),
    .O          (sys_clk       )
);
//////////PLL IP call//////////
clk_wiz_0 clk_wiz_0_inst
// Clock in ports
.clkin1(sys_clk),      // IN 50Mhz
// Clock out ports
.clkout0(),           // OUT 200Mhz
.clkout1(),           // OUT 100Mhz
.clkout2(),           // OUT 50Mhz
.clkout3(clk_out ),   // OUT 25Mhz
// Status and control signals
.rst(~rst_n),         // RESET IN
.lock()               // output
); // OUT
endmodule

```

程序中先用实例化 clk_wiz_0, 把输入的 200MHz 的差分时钟通过 GTP_INBUFGDS 变成单端时钟 sys_clk, 然后再到 clk_wiz_0 的 clkin1, 把 clk_out3 的输出赋给 clk_out。

8. 保存工程后, pll_test 自动成为了 top 文件, clk_wiz_0 成为 Pll_test 文件的子模块。



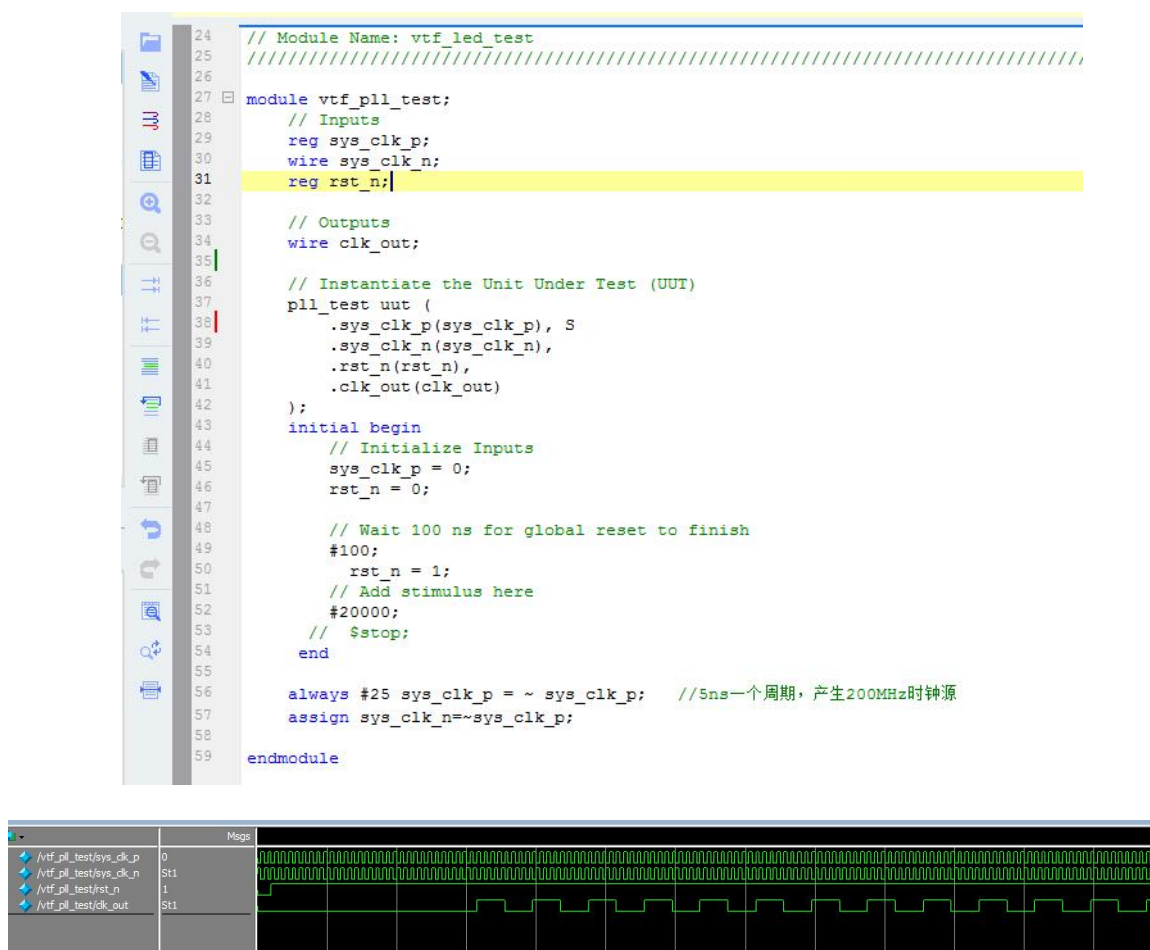
9. 再为工程进行管脚约束，这里不做介绍，见教程《LED 流水灯实验及仿真》中的内容。

注意：例化的目的是在上一级模块中调用例化的模块完成代码功能，在 Verilog 里例化信号的格式如下：模块名必须和要例化的模块名一致，包括信号名也必须一致，模块与模块之间的连接信号不能相互冲突，否则会产生编译错误。

```
模块名  扩展名
(
    .模块信号 1      (连接信号 1)
    .模块信号 2      (连接信号 2)
    .模块信号 3      (连接信号 3)
    .....
    .模块信号 N      (连接信号 N)
);
```

5 Modelsim 仿真 PLL 输出波形

创建仿真文件 vtf_pll_test.v 文件，鼠标右键点击仿真文件，点击 “run Behavior Simulation” 会打开 Modelsim 软件进行波形仿真。仿真文件和仿真波形如图所示：



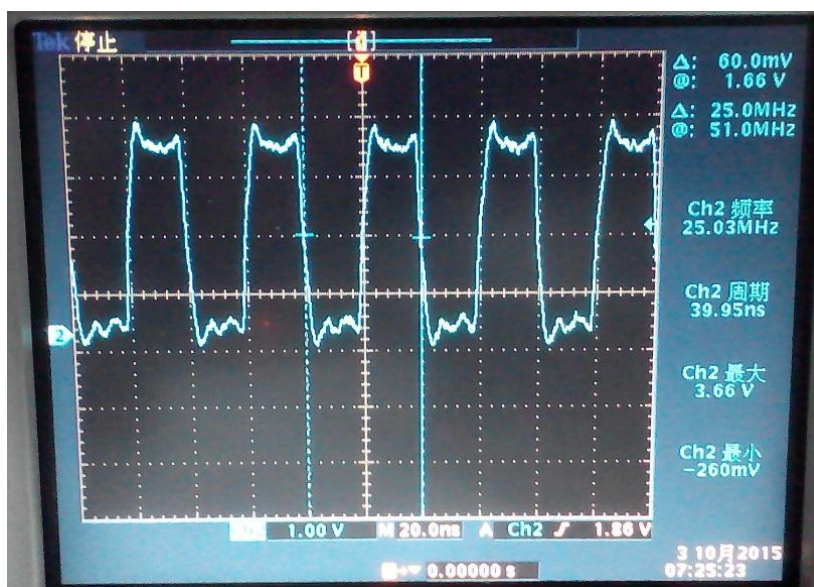
波形图中 sys_clk_p 为 200MHz 的系统时钟，rst_n 为复位信号，clk_out 为 PLL 输出的 25MHz。

6 测量 PLL 输出波形

编译工程并生成 pll_test.sbit 文件，再把 sbit 文件下载到 FPGA 中，接下去我们就可以用示波器来测量输出时钟波形了。

用示波器探头的地线连接到开发板上的地（开发板 J13 的 PIN1 脚），信号端连接开发板 J13 的 PIN3 脚（测量的时候需要注意，避免示波器表头碰到其它管脚而导致电源和地短路）。

这时我们可以在示波器里看到 25Mhz 的时钟波形，波形的幅度为 3.3V, 占空比为 1:1, 波形显示如下图所示：



如果您想输出其它频率的波形，可以修改时钟的输出为 clk_wiz_0 的 clkout0 或 clkout1 或 clkout2。也可以修改 clk_wiz_0 的 clkout3 为您想要的频率，这里也需要注意一下，因为时钟的输出是通过 PLL 对输入时钟信号的倍频和分频系数来得到的，所以并不是所有的时钟频率都可以用 PLL 能够精确产生的，不过 PLL 也会自动为您计算实际输出接近的时钟频率。