FPGA 实验系列

Based on AMD Xilinx™ Kintex-7 series

流水灯实验手册

flowing water light experiment

文档版本 01

发布日期 2022-03-26

陈语.cn

自豪地使用 FPGA 配置无限可能

关于本文档

作者信息

作者	陈语	**时间**	2022-03-30
评审		**时间**	
签发		**时间**	

内容简介

本文档介绍了流水灯实验的全流程，完整、全面的进行了实验内容的分析，设计，详尽的展现了实验的每一步。

修订记录

修改记录累积了每次文档更新的说明。最新版本的文档包含以前所有文档版本的更新内容。

文档版本 01 (2022-03-26)

第一次正式发布。

文档版本 02 (2022-03-30)

第二次正式发布。重构了文档结构，在各个章节增加了解释性说明内容。

目录

1 相关理论

发光二极管简称为 LED。由含镓（Ga）、砷（As）、磷（P）、氮（N）等的化合物制成，是一种常用的发光器件，通过电子与空穴复合释放能量发光，它在照明领域应用广泛。当在发光二极管中通以电流时，电子与空穴会直接复合，从而释放能量发光，其具有功耗小、使用寿命长等优点。[1]

所谓流水灯，就是多个灯泡能够轮流点亮，就像是在流水一样。例如有 4 个发光二极管（就是灯泡的一种，下文简称 LED），在 0-1 秒的时候第一个 LED 灯点亮，在 1-2 秒的时候第二个 LED 灯点亮，在 2-3 秒的时候第三个 LED 灯点亮，在 3-4 秒的时候第四个 LED 灯点亮，而当第 4-5 秒的时候又回到第一个 LED 灯点亮的状态，接下来继续循环点亮 LED 灯。

那么如何通过 FPGA 去控制 LED 灯呢？

正如上面说的，只有当 LED 中通过电流时，它才会点亮，就像下面的电路一样。

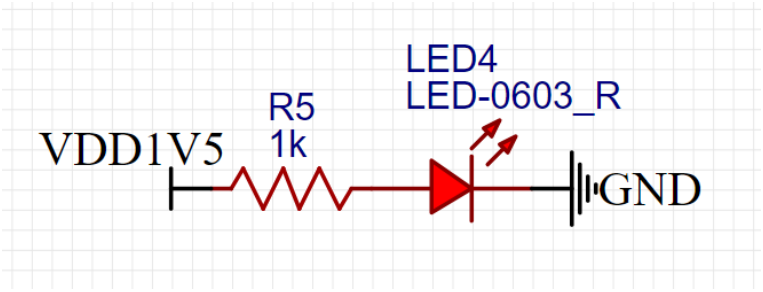


图 1.1 LED 电路原理图

那么如何使用 FPGA 去实现 LED 的控制呢？有两个思路，第一个是如上电路的 GND 连接上 FPGA 的引脚，FPGA 通过控制该引脚为低电平来点亮 LED，也可以通过令该引脚输出高电平来熄灭 LED，因为一端是 VDD1V5，也就是高电平，我们只有给另外一端接上低电平，才能产生电流，LED 才能点亮。

在另外一个思路中，将电路中的 VDD1V5 换成 FPGA 的引脚，类似于上面的第一种思路，我们可以通过使 FPGA 对应引脚输出高电平来点亮 LED，使对应引脚输出低电平来熄灭 LED 灯。

在大多数情况下，都是采用的第一种做法，也就是 LED 的一端连接电源，而 FPGA 的引脚连接 LED 的另外一端。当然，虽然第二种做法的出现概率较低，我们也需要提前检查相关电路的原理图，相信实际而不是经验是一个好习惯，下面让我们开始吧！

[1] 部分引用自百度百科
https://baike.baidu.com/item/%E5%8F%91%E5%85%89%E4%BA%8C%E6%9E%81%E7%AE%A1/1521336

2 实验目标

2.1 成果性目标

- 使用 FPGA 开发板上的 4 颗 LED 实现流水灯效果。

2.2 约束性目标

- 流水灯的切换时间应大于 200ms，并且小于 2s。
- 流水灯的切换应流畅自然，不能出现残影等情况。

3 实验原理

3.1 硬件原理

开发板板载了 8 个 LED 灯，在本实验中，采用 LED4 到 LED7 进行流水灯的展示。

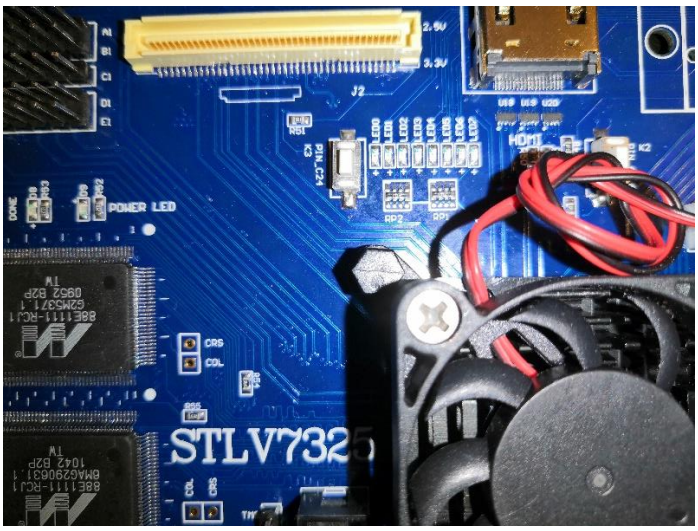


图 3.1 开发板 LED 实物图

找到开发板 LED 对应的电路，我们可以发现，LED 的一端已经被连接到 1.8V，也就是高电平，我们如果想要点亮对应的 LED，就需要在另外一端的引脚输出低电平。

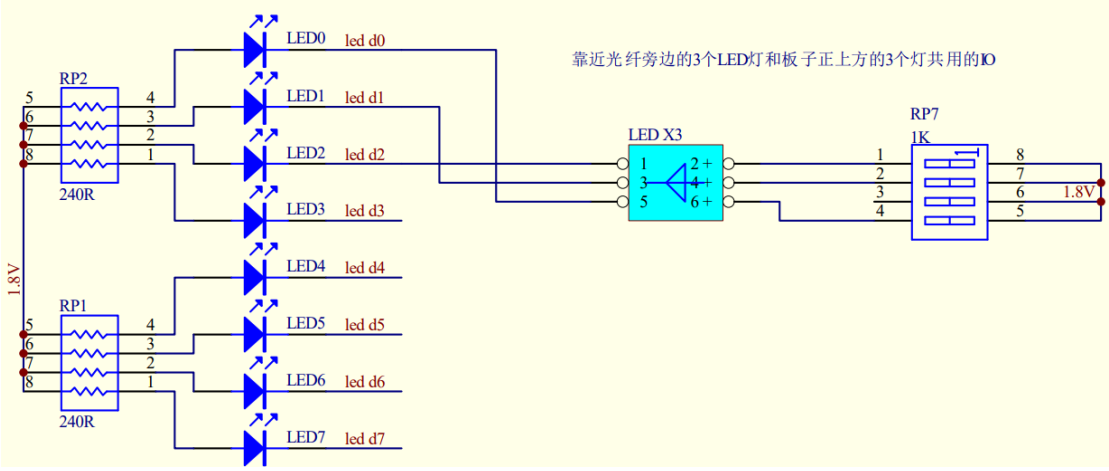


图 3.2 开发板 LED 原理图

3.2 软件原理

3.2.1 模块框图

图 3.3 LED_FLOW 模块原理图

LED_FLOW 模块的作用是产生 LED 流水灯的控制信号。

表 3.1 LED_FLOW 模块端口列表

信号名	方向	位宽	说明
Sys_clk	Input	1	系统时钟，100Mhz
Sys_rst_n	Input	1	系统复位信号，低电平有效
Io_led	Output	4	LED 输出信号，连接到 4 个 LED

3.2.2 时序图

需要实现流水灯的效果，我们必须控制不同的 LED 在不同的时间段内点亮或熄灭。果切换的速度过快，就会发生视觉残留现象。如在夏天的时候看向工作的电风扇，你会认为旋转的是一个圆面，而不是几片扇片。

而我们系统时钟源是 100Mhz 的，这很明显是一个非常快的频率，所以我们需要将它降低到足够我们能够看得清楚的频率，例如 4hz，也就是每 0.25 秒切换一个 LED 灯。为了实现这样的功能，我们引进了一个计数器，并给它命名为 Time_counter，它的作用是，每当 Sys_clk 产生一个上升沿(既从低电平变成高电平)的时候，它的计数会自增 1，倘若我们想要计时 0.25s，我们就需要这个计数器记录 25000000 次。我们刚刚看了原理图，这个开发板上的 LED 是共阳极的(其中一端被连接到电源)，所以在复位有效的情况下，Io_led 为 4 位的 1111，也就是四个 LED 都熄灭，在第一个 0.25s 中，最低位的 LED 被点亮，其他保持熄灭。

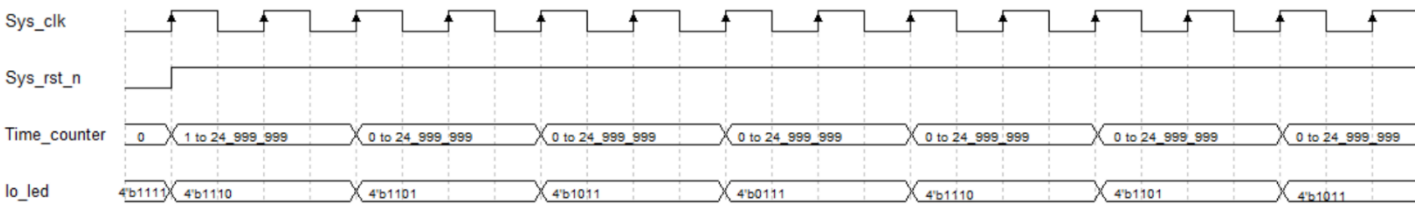


图 3.4 LED_FLOW 模块时序图

4 实验设计

4.1 代码编写

为了方便读者理解，本实验分别采用 Verilog 语言和 VHDL 实现功能。

Verilog 代码源文件(LED_FLOW.v)

```
//------------------------------------------------------
//Filename        : LED_FLOW.v
//Author          : ChanRa1n
//Description     : LED 流水灯实验
//Calledby        : /
//RevisionHistory : 2022-03-30 23:08:48
//Revision        : 1.0
//Email           : chenyu@myfpga.cn
//Website         : https://www.myfpga.cn
//Copyright(c) 2018-2022, MYFPGA.CN, All right reserved
//------------------------------------------------------

module LED_FLOW (
    input            Sys_clk,  // 系统时钟，100Mhz
    input            Sys_rst_n,// 系统复位信号，低电平有效
    output reg [3:0] Io_led    // LED 输出信号，连接到 4 个 LED
);

parameter TIME_SET = 25_000_000; // 在系统时钟为 100Mhz 的情况下，25M 为
0.25 秒
reg [24:0] Time_counter;       // 计时器，用来计算 0.25s

always @(posedge Sys_clk or negedge Sys_rst_n) begin
    if(~Sys_rst_n)begin
        Io_led <= 4'b0000;     // 当复位有效的时候，点亮所有 LED
    end
    else begin
        if(Time_counter==TIME_SET)begin // 如果计时到达 0.25s
            case(Io_led) // 根据当前状态决定下一个状态
                4'b0000:Io_led<=4'b1110;  // 如果当前全部都是点亮的，则只点
亮第一个 LED
                4'b1110:Io_led<=4'b1101;  // 如果当前第一个 LED 点亮，则只点
亮第二个 LED
```

```
                4'b1101:Io_led<=4'b1011;  // 如果当前第二个 LED 点亮，则只点
亮第三个 LED
                4'b1011:Io_led<=4'b0111;  // 如果当前第三个 LED 点亮，则只点
亮第四个 LED
                4'b0111:Io_led<=4'b1110;  // 如果当前第四个 LED 点亮，则只点
亮第一个 LED
            endcase
        end
        else begin
            Io_led<=Io_led; // 如果计时没有到达 0.25s，则保持 LED 状态
        end
    end
end

always @(posedge Sys_clk or negedge Sys_rst_n) begin
    if(~Sys_rst_n)begin
        Time_counter <= 0;     // 当复位有效的时候，计时器清零
    end
    else begin
        if(Time_counter < TIME_SET)begin
            Time_counter <= Time_counter + 1; // 如果计时器没有记到
0.25s，则继续计时
        end
        else begin
            Time_counter <= 0; // 如果计时器记满 0.25s，则清空计时器
        end
    end
end

endmodule
```

VHDL 代码源文件(LED_FLOW.vhd)

```
------------------------------------------------------------
--Filename        : LED_FLOW.vhd
--Author          : ChanRa1n
--Description     : LED 流水灯实验
--Calledby        : /
--RevisionHistory : 2022-03-30 23:19:40
--Revision        : 1.0
--Email           : chenyu@myfpga.cn
--Website         : https://www.myfpga.cn
--Copyright(c) 2018-2022, MYFPGA.CN, All right reserved
```

```
------------------------------------------------------------

Library ieee;
use ieee.std_logic_1164.all;

entity LED_FLOW is
    generic(TIME_SET:integer := 25_000_000);
    port(
        Sys_clk  :in  std_logic;
        Sys_rst_n:in  std_logic;
        Io_led   :buffer std_logic_vector(3 downto 0));
end LED_FLOW;

architecture behav of LED_FLOW is
signal Time_counter:integer range 0 to 25000000;
begin
    process(Sys_clk) begin
        if Sys_rst_n='0' then Io_led <= "0000";
        else
            if Time_counter = TIME_SET then
                case Io_led is
                    when "0000" => Io_led <= "1110";
                    when "1110" => Io_led <= "1101";
                    when "1101" => Io_led <= "1011";
                    when "1011" => Io_led <= "0111";
                    when "0111" => Io_led <= "1110";
                    when others => Io_led <= "0000";
                end case;
            else
                Io_led <= Io_led;
            end if;
        end if;
    end process;

    process(Sys_clk) begin
        if Sys_rst_n='0' then Time_counter <= 0;
        else
            if Time_counter < TIME_SET then
                Time_counter <= Time_counter + 1;
            else
                Time_counter <= 0;
            end if;
        end if;
```

```
    end process;

end behav;
```

4.2 资源消耗

Total Power	Failed Routes	LUT	FF	BRAM	URAM	DSP
		40	**30**	**0**	**0**	**0**
0.403	0	40	30	0	0	0

5 仿真测试

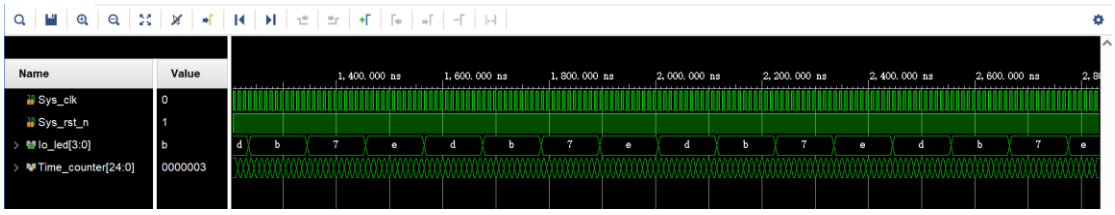
5.1 模块测试

代码源文件(LED_FLOW_TB.v)

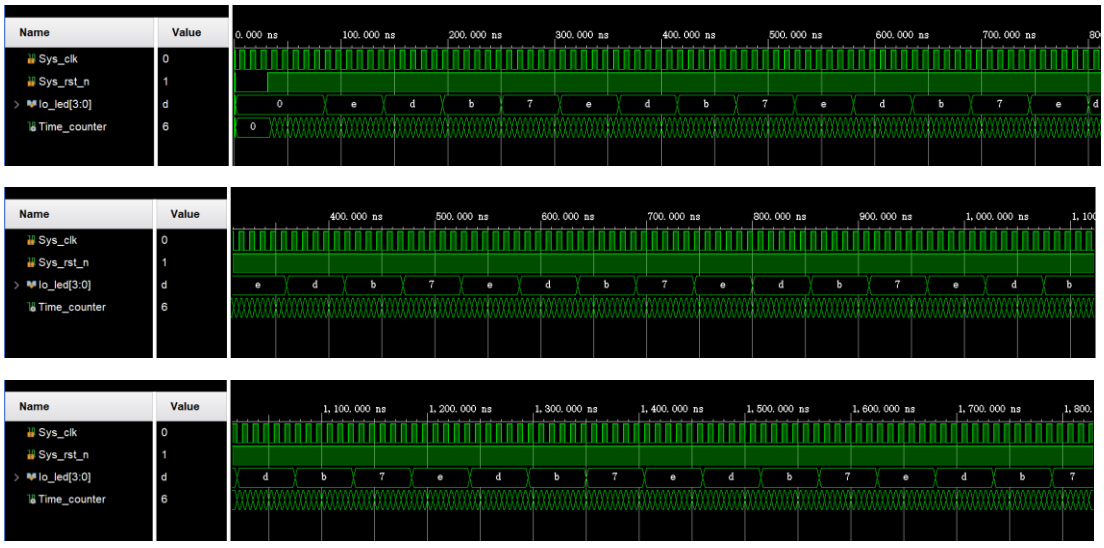
```
//------------------------------------------------------------
//Filename        : LED_FLOW_TB.v
//Author          : ChanRa1n
//Description     : LED_FLOW.v 的测试文件
//Calledby        : Topmodule
//RevisionHistory : 2022-03-30 23:19:40
//Revision        : 1.0
//Email           : chenyu@myfpga.cn
//Website         : https://www.myfpga.cn
//Copyright(c) 2018-2022, MYFPGA.CN, All right reserved
//------------------------------------------------------------

`default_nettype wire
`timescale 1ns/1ns

module LED_FLOW_TB;
reg clk; // 测试时钟信号
reg rst_n; // 测试复位信号
wire [3:0] Io_led;

LED_FLOW  #(.TIME_SET(10))
inst_LED_FLOW ( //模块例化
    .Sys_clk          (clk),
    .Sys_rst_n        (rst_n),
    .Io_led           (Io_led)
);
```

```
localparam CLK_PERIOD = 10;
always #(CLK_PERIOD/2) clk=~clk; // 测试时钟信号为 50Mhz

initial begin
    #1 rst_n<=1'b0;clk<=1'b0;
    #(CLK_PERIOD*3) rst_n<=1;
    #(CLK_PERIOD*300) rst_n<=0;
    $stop;
end

endmodule
```

下面是对 Verilog 版本的流水灯模块的仿真图。



下面是对 VHDL 版本的流水灯模块的仿真图。



5.2 集成测试

该实验只有一个模块，不需要做集成测试。

6 上板验证

6.1 引脚约束

6.1.1 引脚分配表

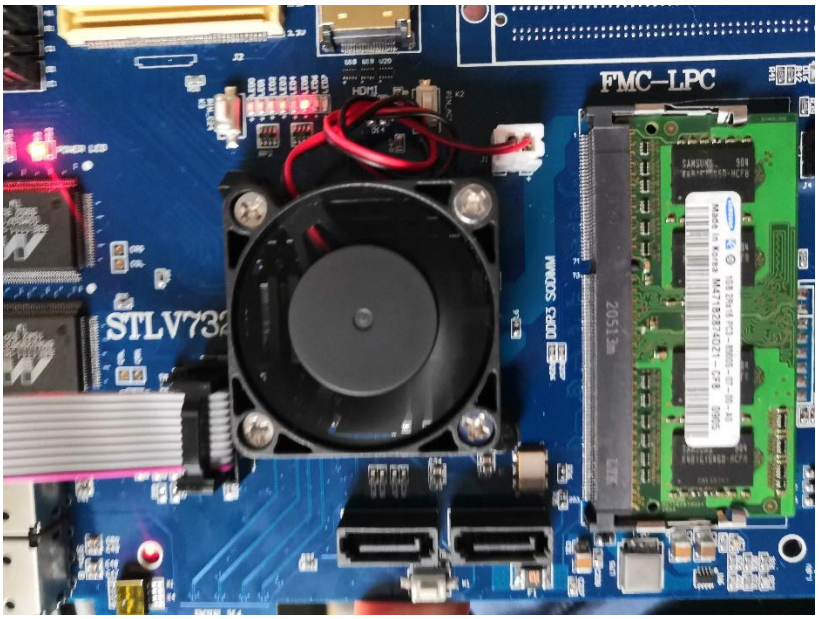
信号名	引脚	电压	说明
Sys_clk	F17	3.3V	系统时钟 100Mhz
Sys_rst_n	C24	3.3V	系统复位，KEY3
LED[0]	AE10	1.5V	LED4
LED[1]	W11	1.5V	LED5
LED[2]	V11	1.5V	LED6
LED[3]	Y12	1.5V	LED7

6.1.2 引脚 XDC 文件

```
set_property PACKAGE_PIN F17 [get_ports Sys_clk]
set_property PACKAGE_PIN C24 [get_ports Sys_rst_n]
set_property PACKAGE_PIN AE10 [get_ports {Io_led[0]}]
set_property PACKAGE_PIN W11 [get_ports {Io_led[1]}]
set_property PACKAGE_PIN V11 [get_ports {Io_led[2]}]
set_property PACKAGE_PIN Y12 [get_ports {Io_led[3]}]

set_property IOSTANDARD LVCMOS15 [get_ports {Io_led[3]}]
set_property IOSTANDARD LVCMOS15 [get_ports {Io_led[2]}]
set_property IOSTANDARD LVCMOS15 [get_ports {Io_led[1]}]
set_property IOSTANDARD LVCMOS15 [get_ports {Io_led[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports Sys_rst_n]
set_property IOSTANDARD LVCMOS33 [get_ports Sys_clk]
```

6.2 结果验证

将项目编译后下载至开发板，发现 LED 流水效果正常，且满足预期目标。



7 实验展望

为了进一步降低资源的消耗，提高最高工作频率，对于时序上要求不是特别严格的情况下，这里提出了一种折中的代码实现方式。

代码源文件(LED_FLOW.v)

```
//-------------------------------------------------------
//Filename        : LED_FLOW.v
//Author          : ChanRa1n
//Description     : Led flow function.
//Calledby        : Topmodule
//RevisionHistory : 2022-03-22 09:47:42
//Revision        : 1.0
//Email           : chenyu@myfpga.cn
//Website         : https://www.myfpga.cn
//Copyright(c) 2018-2022, MYFPGA.CN, All right reserved
//-------------------------------------------------------

`define LED_WIDTH 4
`default_nettype wire
module LED_FLOW (
    input   wire [0:0]              Sys_clk_in,
    input   wire [0:0]              Sys_rst_n,
    output  wire [`LED_WIDTH-1:0]  Io_led
);

    reg   [25:0]   time_counter;
    assign Io_led = 1<<(time_counter>>24);

always @(posedge Sys_clk_in or negedge Sys_rst_n) begin
    if(~Sys_rst_n)begin
        time_counter<=0;
    end
    else begin
        time_counter<=time_counter+1;
    end
end

endmodule
```