

实验07 FPGA 实验平台及 IP 核使用

实验目的

- 熟悉 FPGAOL 在线实验平台结构及使用
- 掌握 FPGA 开发各关键环节
- 学会使用 IP 核（知识产权核）

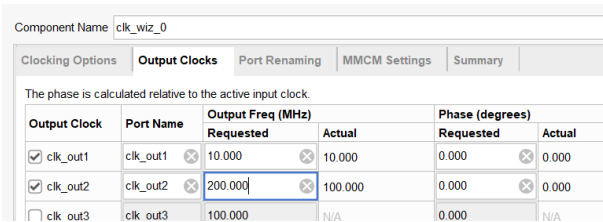
实验环境

- VLAB: vlab.ustc.edu.cn
- FPGAOL: fpgaol.ustc.edu.cn
- Logisim
- Vivado

实验练习

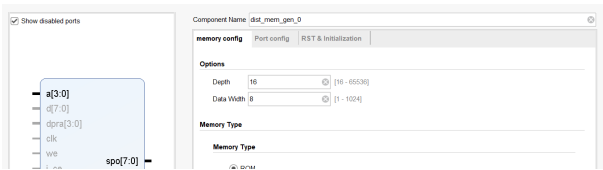
题目 1. 例化一个 16*8bit 的 ROM，并对其进行初始化，输入端口由 4 个开关控制，输出端口连接到七段数码管上（七段数码管与 LED 复用相同的一组管脚），控制数码管显示与开关相对应的十六进制数字，例如四个开关输入全为零时，数码管显示“0”，输入全为 1 时，数管显示“F”。

1. First, customize clock wizard to input 100MHz, output 10MHz, 200MHz. 设置完成后点击确认，生成 IP 核。

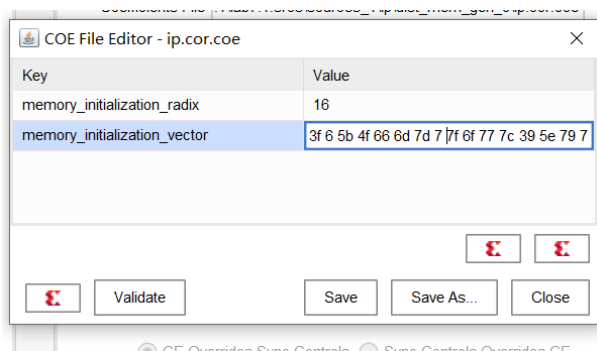


The phase is calculated relative to the active input clock					
Output Clock	Port Name	Output Freq (MHz)		Phase (degrees)	
		Requested	Actual	Requested	Actual
<input checked="" type="checkbox"/> clk_out1	clk_out1	10.000	10.000	0.000	0.000
<input checked="" type="checkbox"/> clk_out2	clk_out2	200.000	100.000	0.000	0.000
<input type="checkbox"/> clk_out3	clk_out3	100.000	N/A	0.000	N/A

2. 按照要求例化 ROM。



3. 初始化 ROM 使之与七段显示数码管的16进制相对应。



4. verilog 代码:

```
module t1 (
    input [3:0] a,
    output [7:0] spo
);
    dist_mem_gen_0 dist_mem_gen_0(
        .a(a),
        .spo(spo)
    );
endmodule
```

5. 仿真文件:

```
module test_bench (
);
    integer i=0;
    reg [3:0] a;
    wire [7:0] spo;
    display display(.a(a),.spo(spo));
    initial begin
        a=0;
        for(i=0;i<16;i=i+1)begin
            #10 a=i;
        end
        #10$stop;
    end
endmodule
```

6. 修改脚管约束文件:

```
## Clock signal
set_property -dict { PACKAGE_PIN E3      IOSTANDARD LVCMOS33 } [get_ports {
```

```

clk }]]; #IO_L12P_T1_MRCC_35 Sch=clk100mhz
#create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5}
[get_ports {CLK100MHZ}]];
set_property -dict { PACKAGE_PIN B18      IOSTANDARD LVCMOS33 } [get_ports {
rst }]];
## FPGA0L LED (signle-digit-SEGPLAY)

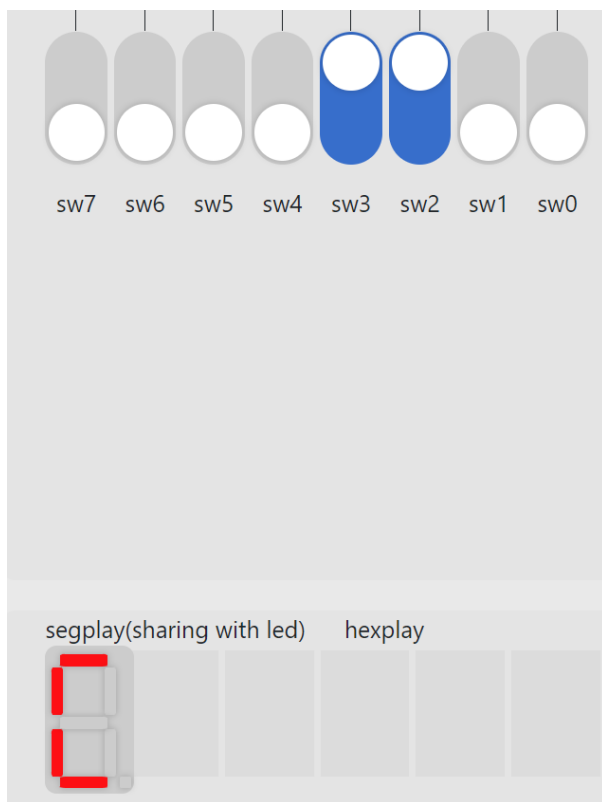
set_property -dict { PACKAGE_PIN C17      IOSTANDARD LVCMOS33 } [get_ports {
spo[0] }]];
set_property -dict { PACKAGE_PIN D18      IOSTANDARD LVCMOS33 } [get_ports {
spo[1] }]];
set_property -dict { PACKAGE_PIN E18      IOSTANDARD LVCMOS33 } [get_ports {
spo[2] }]];
set_property -dict { PACKAGE_PIN G17      IOSTANDARD LVCMOS33 } [get_ports {
spo[3] }]];
set_property -dict { PACKAGE_PIN D17      IOSTANDARD LVCMOS33 } [get_ports {
spo[4] }]];
set_property -dict { PACKAGE_PIN E17      IOSTANDARD LVCMOS33 } [get_ports {
spo[5] }]];
set_property -dict { PACKAGE_PIN F18      IOSTANDARD LVCMOS33 } [get_ports {
spo[6] }]];
set_property -dict { PACKAGE_PIN G18      IOSTANDARD LVCMOS33 } [get_ports {
spo[7] }]];

## FPGA0L SWITCH

set_property -dict { PACKAGE_PIN D14      IOSTANDARD LVCMOS33 } [get_ports {
a[0] }]];
set_property -dict { PACKAGE_PIN F16      IOSTANDARD LVCMOS33 } [get_ports {
a[1] }]];
set_property -dict { PACKAGE_PIN G16      IOSTANDARD LVCMOS33 } [get_ports {
a[2] }]];
set_property -dict { PACKAGE_PIN H14      IOSTANDARD LVCMOS33 } [get_ports {
a[3] }]];

```

7. 利用 vivado 综合生成 bitstream 文件并烧写进入 FPGA 在线平台。结果如下：



题目 2. 采用 8 个开关作为输入，两个十六进制数码管作为输出，采用时分复用的方式将开关的十六进制数值在两个数码管上显示出来，例如高四位全为 1，低四位全为 0 时，数码管显示“F0”。

1. 编写 verilog 代码如下，将数码管多个位同时显示的方式为对 hexplay_an 以较快的频率进行扫描，每次显示数码管对应位，这样在人眼看来数码管所有位是同时显示的。

```
module t2(  
    input [7:0]sw,  
    input clk,  
    output reg [2:0]an,  
    output reg [3:0]hexplay_data);  
    reg [32:0] hexplay_cnt;  
    always@(posedge clk) begin  
        if (hexplay_cnt >= (20000000 / 8))  
            hexplay_cnt <= 0;  
        else  
            hexplay_cnt <= hexplay_cnt + 1;  
    end  
    always@(posedge clk)  
    begin  
        if(hexplay_cnt==0)  
            begin  
                if(an==3'b000)  
                    an<=3'b001;  
                else
```

```

        an<=3'b000;
    end
end
always@(posedge clk)
begin
    if(an==3'b000)
        hexplay_data<=sw[3:0];
    else
        hexplay_data<=sw[7:4];
    end
endmodule

```

2. 编写仿真文件：

```

module t2_testbanch();
reg clk;
reg [7:0]sw;
wire [2:0]an;
wire [3:0]hexplay_data;
integer i;
initial
begin
    sw=0;
    for(i=8'b0;i<=8'b11111111;i=i+1)
    begin
        sw=sw+1;
        #10;
    end
end
initial
begin
    clk=0;
    forever
    #1 clk=~clk;
end
p2 p2(.sw(sw),.clk(clk),.an(an),.hexplay_data(hexplay_data));
endmodule

```

3. 修改脚管文件，这次需要用到 hexplay.

```

## Clock signal
set_property -dict { PACKAGE_PIN E3      IOSTANDARD LVCMOS33 } [get_ports {

```

```

clk }]]; #IO_L12P_T1_MRCC_35 Sch=clk100mhz
#create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5}
[get_ports {CLK100MHZ}]];

## FPGA0L SWITCH

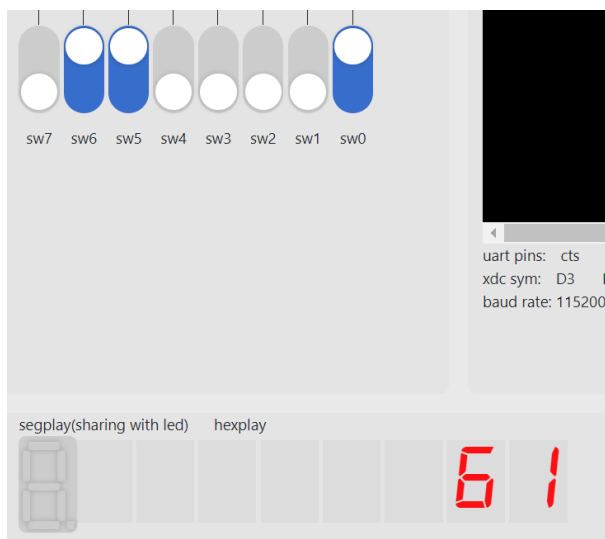
set_property -dict { PACKAGE_PIN D14      IOSTANDARD LVCMOS33 } [get_ports {
sw[0] }]];
set_property -dict { PACKAGE_PIN F16      IOSTANDARD LVCMOS33 } [get_ports {
sw[1] }]];
set_property -dict { PACKAGE_PIN G16      IOSTANDARD LVCMOS33 } [get_ports {
sw[2] }]];
set_property -dict { PACKAGE_PIN H14      IOSTANDARD LVCMOS33 } [get_ports {
sw[3] }]];
set_property -dict { PACKAGE_PIN E16      IOSTANDARD LVCMOS33 } [get_ports {
sw[4] }]];
set_property -dict { PACKAGE_PIN F13      IOSTANDARD LVCMOS33 } [get_ports {
sw[5] }]];
set_property -dict { PACKAGE_PIN G13      IOSTANDARD LVCMOS33 } [get_ports {
sw[6] }]];
set_property -dict { PACKAGE_PIN H16      IOSTANDARD LVCMOS33 } [get_ports {
sw[7] }]];

## FPGA0L HEXPLAY

set_property -dict { PACKAGE_PIN A14      IOSTANDARD LVCMOS33 } [get_ports {
hexplay_data[0] }]];
set_property -dict { PACKAGE_PIN A13      IOSTANDARD LVCMOS33 } [get_ports {
hexplay_data[1] }]];
set_property -dict { PACKAGE_PIN A16      IOSTANDARD LVCMOS33 } [get_ports {
hexplay_data[2] }]];
set_property -dict { PACKAGE_PIN A15      IOSTANDARD LVCMOS33 } [get_ports {
hexplay_data[3] }]];
set_property -dict { PACKAGE_PIN B17      IOSTANDARD LVCMOS33 } [get_ports {
an[0] }]];
set_property -dict { PACKAGE_PIN B16      IOSTANDARD LVCMOS33 } [get_ports {
an[1] }]];
set_property -dict { PACKAGE_PIN A18      IOSTANDARD LVCMOS33 } [get_ports {
an[2] }]];

```

4. 最后进行 generate bitstream, 烧写进实验平台得到如下结果:



题目 3. 利用本实验中的时钟管理单元或周期脉冲技术，设计一个精度为 0.1 秒的计时器，用 4 位数码管显示出来，数码管从高到低，分别表示分钟、秒钟十位、秒钟个位、十分之一秒，该计时器具有复位功能（可采用按键或开关作为复位信号），复位时计数值为 1234，即 1 分 23.4 秒。

1. Verilog 代码如下，计时器的更改以 0.1s 为单位，需要一个 10HZ 的时钟。先用 IP 核降频为 10MHZ 再用计数的方法降频为 10HZ，要显示 4 个 16 进制数码管，需要 2 位的使能信号实现分时输出不同信号。

```
module t3 (
    input clk,rst,
    output reg [3:0] out,
    output reg [2:0] selsct
);
    wire clk_n,locked;
    wire pulse_10hz;
    reg [3:0] outm ;reg [3:0] outss;reg [3:0] outsg;reg [3:0] outst;
    reg [1:0] enable;
    reg [19:0] cout;
    clk_wiz_0
clk_wiz_0_insrt(.clk_in1(clk),.clk_out1(clk_n),.reset(rst),.locked(locked
));
    always@(*)
    begin
        case(enable)
            2'd0:begin out=outst; selsct=3'd0; end
            2'd1:begin out=outsg; selsct=3'd1; end
            2'd2:begin out=outss; selsct=3'd2; end
            2'd3:begin out=outm; selsct=3'd3; end
            default:begin out=4'b0; selsct=3'd0; end
        endcase
    end
```

```

end
always@(posedge clk_n or posedge rst)
begin
    if(rst)
    begin
        cout<=20'b0;
        enable<=2'b0;
    end
    else
    begin
        if(cout>=20'd999999)
        cout<=20'b0;
        else cout<=cout+20'b1;
        enable<=enable+2'b1;
    end
end
end
assign pulse_10hz=(cout==20'd999999);
always@(posedge clk_n or posedge rst)
begin
    if(rst==1'b1)
    begin
        outm<=4'd1;
        outss<=4'd2;
        outsg<=4'd3;
        outst<=4'd4;
    end
    else
    begin
        if(pulse_10hz)
        begin
            if(outst>=4'd9)
            begin
                outst<=4'b0;
                if(outsg>=4'd9)
                begin
                    outsg<=4'b0;
                    if(outss>=4'd5)
                    begin
                        outss<=4'b0;
                        outm<=outm+4'b1;
                    end
                else outss<=outss+4'b1;
            end
        end
    end
end
end

```



```

                else outsg<=outsg+4'b1;
            end
            else outst<=outst+4'b1;
        end
    end
end
end
endmodule

```

2. 修改脚管文件:

```

## This file is a general .xdc for FPGA0L_BOARD (adopted from Nexys4 DDR
Rev. C)
## To use it in a project:
## - uncomment the lines corresponding to used pins
## - rename the used ports (in each line, after get_ports) according to
the top level signal names in the project

## Clock signal
set_property -dict { PACKAGE_PIN E3      IOSTANDARD LVCMOS33 } [get_ports {
clk }]; #IO_L12P_T1_MRCC_35 Sch=clk100mhz
#create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5}
[get_ports {CLK100MHZ}];
## FPGA0L BUTTON & SOFT_CLOCK
set_property -dict { PACKAGE_PIN B18     IOSTANDARD LVCMOS33 } [get_ports {
rst }];
## FPGA0L LED (single-digit-SEGPLAY)
# set_property -dict { PACKAGE_PIN C17    IOSTANDARD LVCMOS33 } [get_ports
{ spo[0] }];
# set_property -dict { PACKAGE_PIN D18    IOSTANDARD LVCMOS33 } [get_ports
{ spo[1] }];
# set_property -dict { PACKAGE_PIN E18    IOSTANDARD LVCMOS33 } [get_ports
{ spo[2] }];
# set_property -dict { PACKAGE_PIN G17    IOSTANDARD LVCMOS33 } [get_ports
{ spo[3] }];
# set_property -dict { PACKAGE_PIN D17    IOSTANDARD LVCMOS33 } [get_ports
{ spo[4] }];
# set_property -dict { PACKAGE_PIN E17    IOSTANDARD LVCMOS33 } [get_ports
{ spo[5] }];
# set_property -dict { PACKAGE_PIN F18    IOSTANDARD LVCMOS33 } [get_ports
{ spo[6] }];
# set_property -dict { PACKAGE_PIN G18    IOSTANDARD LVCMOS33 } [get_ports

```

```
{ spo[7] }];
```

FPGA0L SWITCH

```
# set_property -dict { PACKAGE_PIN D14      IOSTANDARD LVCMOS33 } [get_ports  
{ a[0] }];  
# set_property -dict { PACKAGE_PIN F16      IOSTANDARD LVCMOS33 } [get_ports  
{ a[1] }];  
# set_property -dict { PACKAGE_PIN G16      IOSTANDARD LVCMOS33 } [get_ports  
{ a[2] }];  
# set_property -dict { PACKAGE_PIN H14      IOSTANDARD LVCMOS33 } [get_ports  
{ a[3] }];  
# set_property -dict { PACKAGE_PIN E16      IOSTANDARD LVCMOS33 } [get_ports  
{ sw[4] }];  
# set_property -dict { PACKAGE_PIN F13      IOSTANDARD LVCMOS33 } [get_ports  
{ sw[5] }];  
# set_property -dict { PACKAGE_PIN G13      IOSTANDARD LVCMOS33 } [get_ports  
{ sw[6] }];  
# set_property -dict { PACKAGE_PIN H16      IOSTANDARD LVCMOS33 } [get_ports  
{ sw[7] }];
```

FPGA0L HEXPLAY

```
set_property -dict { PACKAGE_PIN A14      IOSTANDARD LVCMOS33 } [get_ports {  
out[0] }];  
set_property -dict { PACKAGE_PIN A13      IOSTANDARD LVCMOS33 } [get_ports {  
out[1] }];  
set_property -dict { PACKAGE_PIN A16      IOSTANDARD LVCMOS33 } [get_ports {  
out[2] }];  
set_property -dict { PACKAGE_PIN A15      IOSTANDARD LVCMOS33 } [get_ports {  
out[3] }];  
set_property -dict { PACKAGE_PIN B17      IOSTANDARD LVCMOS33 } [get_ports {  
selsct[0] }];  
set_property -dict { PACKAGE_PIN B16      IOSTANDARD LVCMOS33 } [get_ports {  
selsct[1] }];  
set_property -dict { PACKAGE_PIN A18      IOSTANDARD LVCMOS33 } [get_ports {  
selsct[2] }];
```

##USB-RS232 Interface

```
#set_property -dict { PACKAGE_PIN C4      IOSTANDARD LVCMOS33 } [get_ports
```

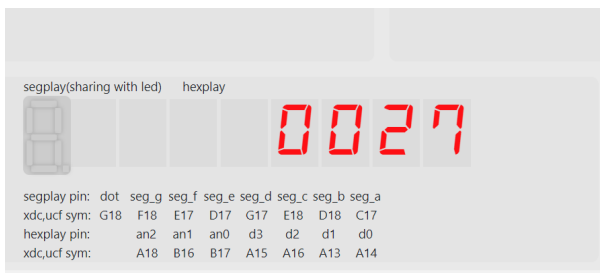
```

{ UART_TXD_IN }]; #IO_L7P_T1_AD6P_35 Sch=uart_txd_in
#set_property -dict { PACKAGE_PIN D4      IOSTANDARD LVCMOS33 } [get_ports
{ UART_RXD_OUT }]; #IO_L11N_T1_SRCC_35 Sch=uart_rxd_out
#set_property -dict { PACKAGE_PIN D3      IOSTANDARD LVCMOS33 } [get_ports
{ UART_CTS }]; #IO_L12N_T1_MRCC_35 Sch=uart_cts
#set_property -dict { PACKAGE_PIN E5      IOSTANDARD LVCMOS33 } [get_ports
{ UART_RTS }]; #IO_L5N_T0_AD13N_35 Sch=uart_rts

#set_property CLOCK_DEDICATED_ROUTE FALSE [get_nets clk];
#set_property CFGBVS VCC0 [current_design]
#where value1 is either VCC0 or GND
#set_property CONFIG_VOLTAGE 3.3 [current_design]
#where value2 is the voltage provided to configuration bank 0

```

3. 生成 bitstream, 烧写到实验平台得到以下结果:



总结与思考

1. 本次实验的收获

- 掌握了有关 IP 核的概念, 会使用它进行辅助开发
- 对时序电路的 verilog 代码编写有了进一步的练习

2. 本次实验的难易程度

- 非常难, 一方面对于 IP 核的使用案例介绍不够多导致操作不够熟练, 一方面对电路设计的要求很高, 需要花大量时间进行思考

3. 本次实验的任务量

- 很大, 一方面要学很多新知识, 一方面又要设计很难的电路

4. 对本次实验的建议

- 建议以后把助教写的指南加到讲义里面, 没有那个真的做不出来