中国科学技术大学计算机学院

《数字电路实验》报告

实验题目：实验 07 FPGA 实验平台及 IP 核使用

学生姓名：Ouedraogo Ezekiel B.

学生学号：PL19215001

完成日期：12/06/2020

计算机实验教学中心制

2020 年 10 月

【实验题目】

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 平台：vlab.ustc.edu.cn
FPGAOL 平台：fpgaol.ustc.edu.cn
Vivado
Logisim

【实验过程】

1. 使用时钟管理单元 IP 核

1.1. 通过计数器产生一个低频的脉冲信号

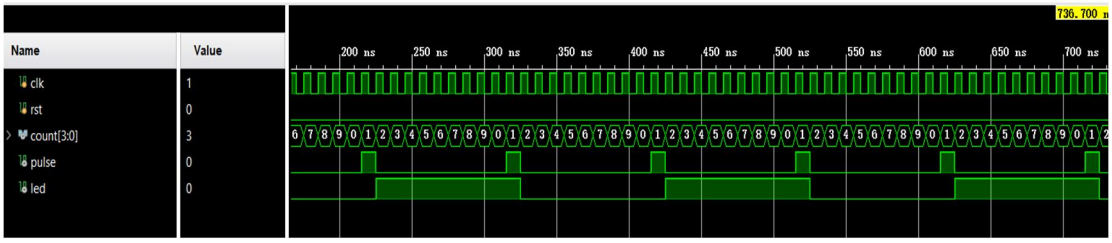
Verilog 代码

```
module clk_10(
    input clk,
    input rst,
    output reg led,
    output reg [3:0]count,
    output pulse
    );
    always@(posedge clk)
    begin
        if(rst || count >= 4'h9)
            count <= 4'h0;
        else
            count <= count + 4'h1;
    end
    assign pulse = (count == 4'h1);
    always@(posedge clk)
    begin
        if(rst)
            led <= 1'b0;
        else if(pulse)
            led <= ~led;
    end
endmodule
```

Verilog 仿真代码

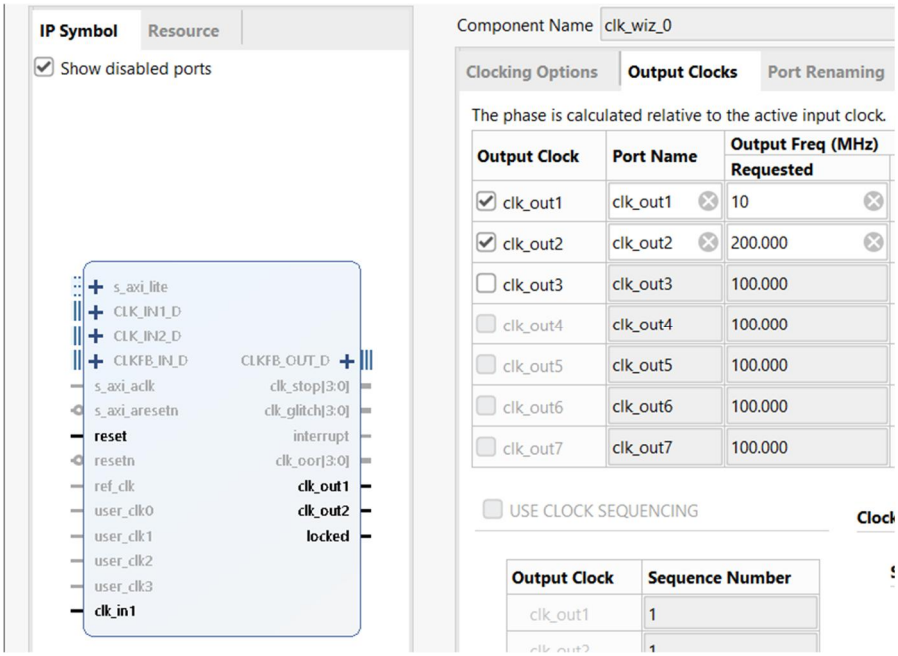
```
module test_bunch(

    );
    reg clk,rst;
    wire [3:0]count;
    wire pulse, led;
    clk_10
clk_10(.clk(clk),.rst(rst),.led(led),.count(count),.pulse(pulse));
    initial clk = 1'b0;
    always #5 clk = ~clk;
    initial
    begin
        rst = 1'b1;
        #6 rst = 1'b0;
        #1000 $finish;
    end
endmodule
```

其仿真波形如下图所示



1.2. 通过时钟管理单元 IP 核生成

用频率为 100Mhz 的时钟生成频率 10Mhz 和 200Mhz 时钟



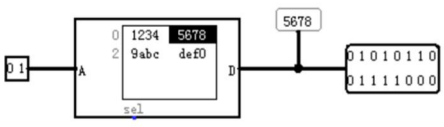
仿真文件代码

```
module test(
input clk,
input rst,
output [7:0] led);
wire clk_10m,clk_200m,locked;
reg [31:0] cnt_1,cnt_2;
always@(posedge clk_200m)
begin
 if(~locked)
 cnt_1 <= 32'hAAAA_AAAA;
 else
 cnt_1 <= cnt_1+1'b1;;
end
always@(posedge clk_10m)
begin
 if(~locked)
 cnt_2 <= 32'hAAAA_AAAA;
 else
 cnt_2 <= cnt_2+1'b1;;
end
assign led = {cnt_1[27:24],cnt_2[27:24]};
clk_wiz_0 clk_wiz_0_inst(
.clk_in1 (clk),
.clk_out1 (clk_10m),
.clk_out2 (clk_200m),
.reset (rst),
.locked (locked));
endmodule
```

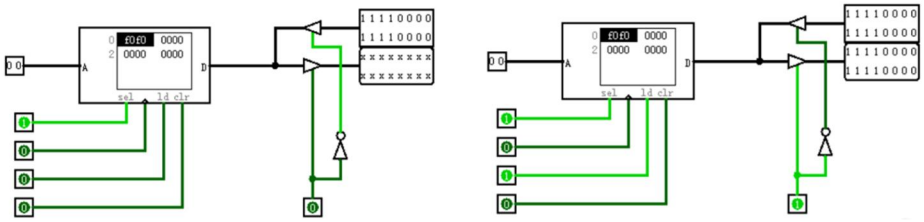
2. 使用片内存储单元

Logism 中的 ROM 和 RAM

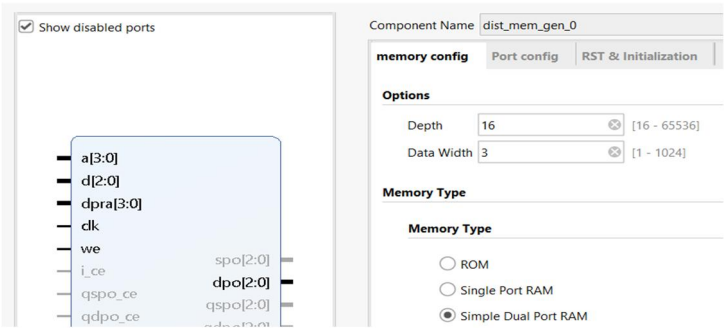
ROM



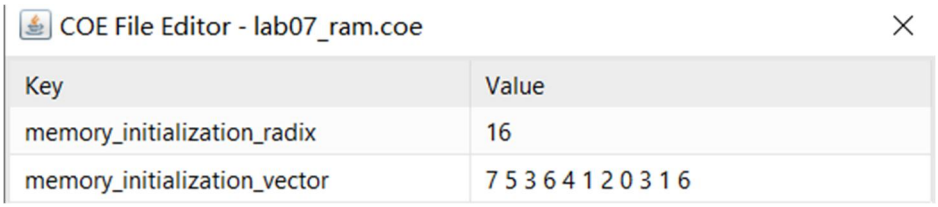
RAM



利用 Vivado 中的存储器
例化一个 16*3bit 的 RAM



初始化

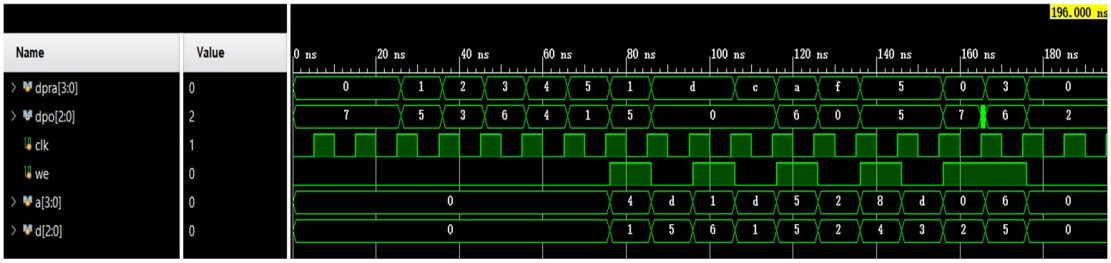


仿真文件代码

```
module tb( );
reg [3:0] dpra;
wire [2:0] dpo;
reg clk;
reg we;
reg [3:0] a;
reg [2:0] d;
initial
begin
   clk = 0;
   forever
   #5 clk = ~clk;
end
initial
begin
   a = 0;dpra=0;d=0;we=0;
   #20
   repeat(5)
   begin
      @(posedge clk); #1;
      dpra = dpra +1;
   end
   repeat(10)
   begin
      @(posedge clk); #1;
      a = $random%16;
      dpra = $random%16;
```

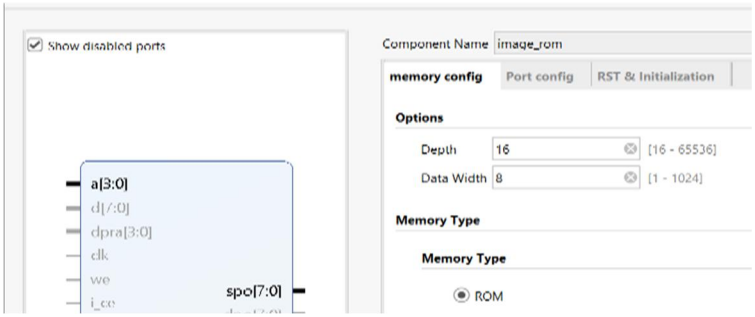
```
        d = $random%8;
        we = $random%2;
    end
    @(posedge clk); #1;
    a = 0;
    dpra = 0;
    d = 0;
    we = 0;
    #20 $stop;
end
dist_mem_gen_0 dist_mem_gen_0(
.a (a),
.d (d),
.dpra (dpra),
.clk (clk),
.we (we),
.dpo (dpo));
endmodule
```

其仿真波形如下图



【实验练习】

1. 例化一个 16*8bit 的 ROM



初始化

memory_initialization_radix	16
memory_initialization_vector	0 1 2 3 4 5 6 7 8 9 a b c d e f

代码

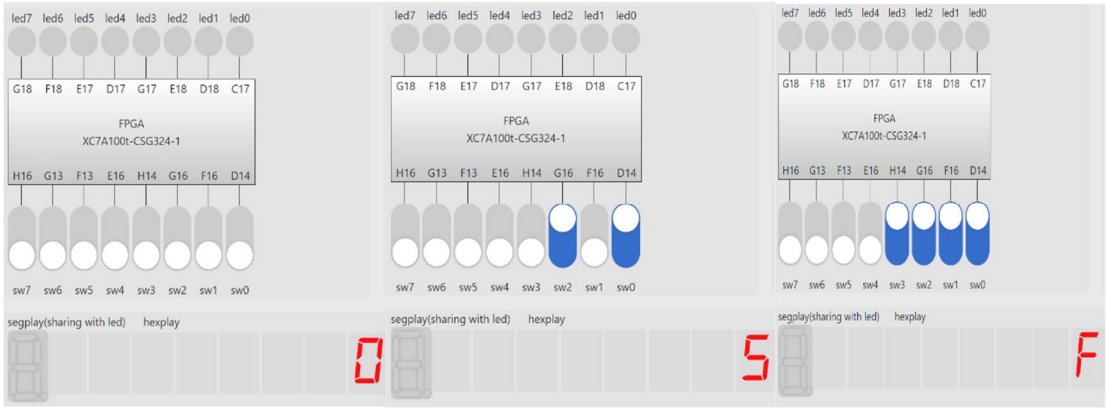
```
module rom(
```

```
input [3:0]a,
output [3:0]out
);
wire [7:0]spo;
assign out = spo[3:0];
image_rom rom(.a(a),.spo(spo));
endmodule
```

XDC 文件

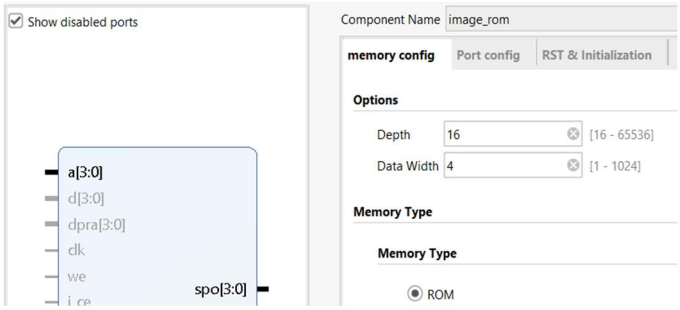
```
## Hexa Digit Display
set_property -dict { PACKAGE_PIN A14  IOSTANDARD LVCMOS33 } [get_ports { out[0] }];
set_property -dict { PACKAGE_PIN A13  IOSTANDARD LVCMOS33 } [get_ports { out[1] }];
set_property -dict { PACKAGE_PIN A16  IOSTANDARD LVCMOS33 } [get_ports { out[2] }];
set_property -dict { PACKAGE_PIN A15  IOSTANDARD LVCMOS33 } [get_ports { out[3] }];

## FPGAOL 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] }];
```

结果



2．式将开关的十六进制数值在两个数码管上显示

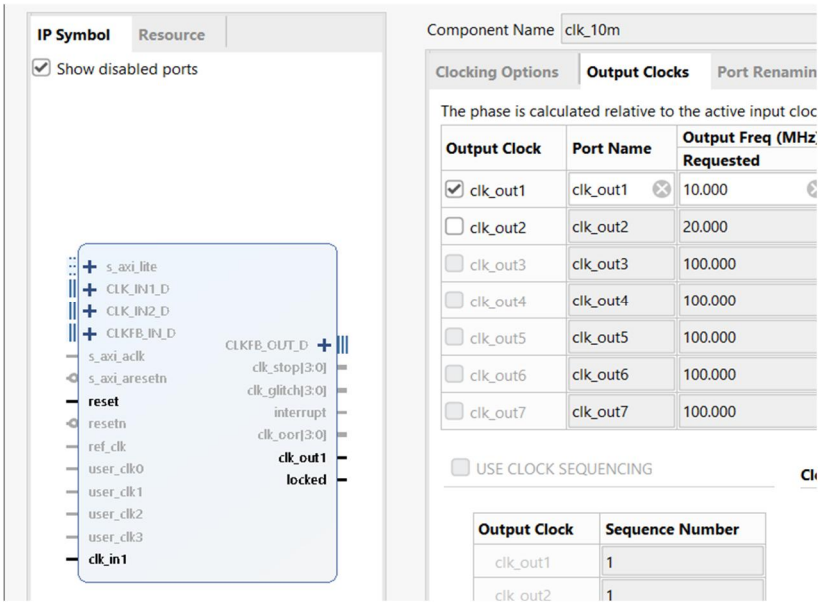
例化一个 16*4bit 的 ROM



初始化

memory_initialization_radix	16
memory_initialization_vector	0 1 2 3 4 5 6 7 8 9 a b c d e f

例化 10Mhz 的时钟



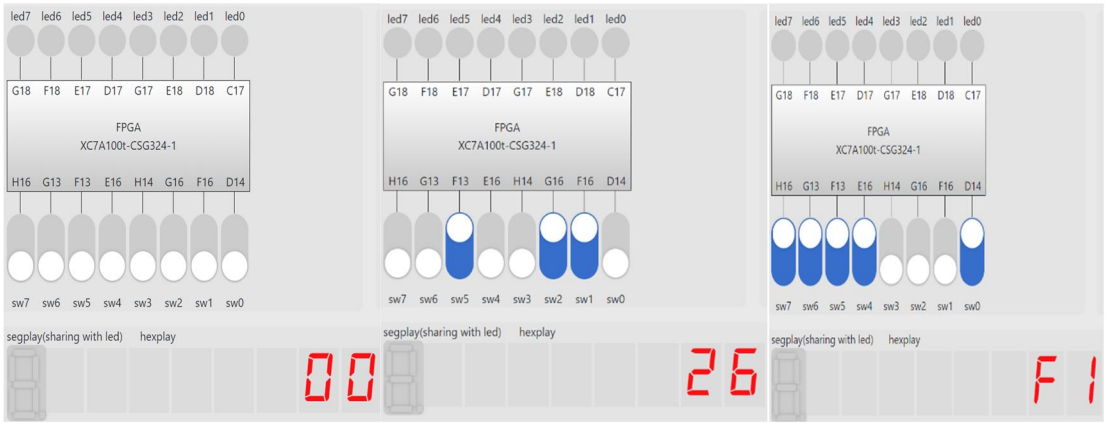
Verilog 代码

```
module rom(
input clk,
input [7:0]sw,
output reg [3:0]out,
output reg [2:0]an
    );
    wire clk_out1;
    wire [3:0]spo0,spo1;
    image_rom rom0(.a(sw[3:0]),.spo(spo0));
    image_rom rom1(.a(sw[7:4]),.spo(spo1));
    clk_10m clk_10m(.clk_in1(clk),.reset(1'b0),.clk_out1(clk_out1));
    always@(*)
    begin
        case(clk_out1)
        0:  begin
                an = 3'b0;
                out = spo0;
            end
        1:  begin
                an = 3'b1;
                out = spo1;
            end
        endcase
    end
endmodule
```

XDC 文件

```
## Clock signal
set_property -dict { PACKAGE_PIN E3   IOSTANDARD LVCMOS33 } [get_ports { clk }]; #IO_L12P_T1_MRCC_35 Sch=clk100mhz
## Hexa Digit Display
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 { 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] }];

## FPGAOL 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] }];


set_property -dict { PACKAGE_PIN D17  IOSTANDARD LVCMOS33 } [get_ports { out[4] }];
set_property -dict { PACKAGE_PIN E17  IOSTANDARD LVCMOS33 } [get_ports { out[5] }];
set_property -dict { PACKAGE_PIN F18  IOSTANDARD LVCMOS33 } [get_ports { out[6] }];
set_property -dict { PACKAGE_PIN G18  IOSTANDARD LVCMOS33 } [get_ports { out[7] }];
```

结果



3. 设计一个精 度为 0.1 秒的计时器

Verilog 代码

```
module display(
    input clk,
    input rst,
    output reg [3:0] out,
    output reg [2:0] an
    );
    //create 10mhz clk
    wire clk_out1;
    clk_10m clk_10m(
        .clk_in1(clk),
        .reset(rst),
        .clk_out1(clk_out1)
```

```
        );
    //get time(min,sec,...)
    wire [3:0]tenth,sec,ten_sec,min;
    count_time count_time(
        .clk_10m(clk_out1),
        .rst(rst),.tenth(tenth),
        .sec(sec),.ten_sec(ten_sec),
        .min(min)
        );
     //display
    reg [1:0]count;
    always@(posedge clk_out1)
    begin
        if(rst) count <= 2'b0;
        else count <= count + 2'b1;
    end
    always@(*)
    begin
        case(count)
        2'h0:begin
                an <= 3'h0;
                out <= tenth;
             end
        2'h1:begin
                an <= 3'h1;
                out <= sec;
            end
        2'h2:begin
                an <= 3'h2;
                out <= ten_sec;
            end
        2'h3:begin
                an <= 3'h3;
                out <= min;
            end
        endcase
    end
endmodule

module count_time(
    input clk_10m,
    input rst,
    output reg [3:0] tenth, sec, ten_sec, min
    );
```

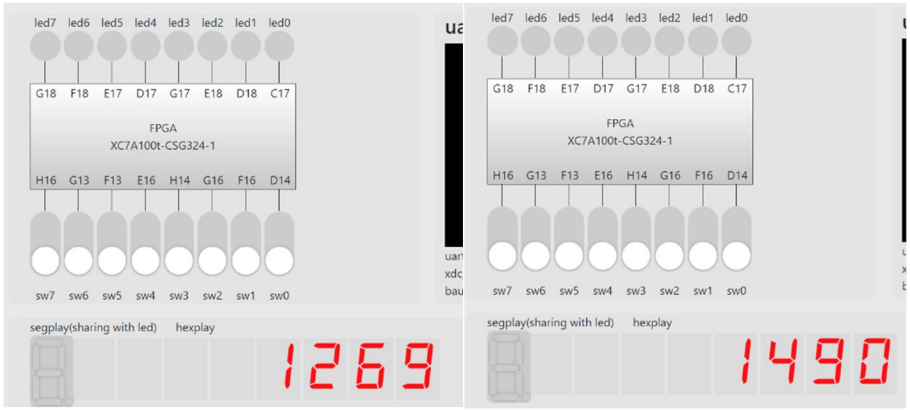
```
    //create 0.1s pulse signal
    reg [19:0]count;
    wire pulse;
    always@(posedge clk_10m or posedge rst)
    begin
        if(rst || count >= 20'hf423f) //hex f 423f = dec 999 999
            count <= 20'h0;
        else
            count <= count + 20'h1;
    end
    assign pulse = (count == 20'h1);
    //count time
    //tenth sec
    always@(posedge clk_10m or posedge rst)
    begin
        if(rst) tenth <= 4'h4;
        else if(pulse)
        begin
            if(tenth >= 4'h9) tenth <= 4'h0;
            else              tenth <= tenth + 4'h1;
        end
    end
    //sec
    always@(posedge clk_10m or posedge rst)
    begin
        if(rst) sec <= 4'h3;
        else if(pulse)
        begin
            if(sec >= 4'h9 && tenth >= 4'h9)
                sec <= 4'h0;
            else if(tenth >= 4'h9) sec <= sec + 4'h1;
        end
    end
    //ten_sec
    always@(posedge clk_10m or posedge rst)
    begin
        if(rst) ten_sec <= 4'h2;
        else if(pulse)
        begin
            if(ten_sec >= 4'h5 && sec >= 4'h9 && tenth >= 4'h9)
                ten_sec <= 4'h0;
            else if(sec >= 4'h9 && tenth >= 4'h9) ten_sec <= ten_sec +
4'h1;
        end
```

```
    end
    //min
    always@(posedge clk_10m or posedge rst)
    begin
        if(rst) min <= 4'h1;
        else if(pulse)
        begin
            if(min >= 4'h9 && ten_sec >= 4'h5 && sec >= 4'h9 && tenth >=
4'h9)
                min <= 4'h0;
            else if(ten_sec >= 4'h5 && sec >= 4'h9 && tenth >= 4'h9) min
<= min + 4'h1;
        end
    end
endmodule
```

XDC 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}];
## FPGAOL BUTTON & SOFT_CLOCK
set_property -dict { PACKAGE_PIN B18  IOSTANDARD LVCMOS33 } [get_ports { rst }];
## Hexa Digit Display
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 { 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] }];
```

结果



【总结与思考】

通过本次实验我们熟悉了 FPGAOL 在线实验平台结构及使用，掌握了 FPGA 开发各关键环节也学会了如何使用 IP 核。