**实验报告**

2021年4月25日 成绩：---------

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 姓名 | 刘爱兵 | 学号 | 19114801 | 班级 | 2班 |
| 专业 | 计算机科学与技术 | 课程名称 | 计算机组成原理 | 任课教师 | 吴云志 |
| 实验序号 | 01 | 实验名称 | 平台使用和HDU-XL-01实验板卡测试实验 | | |
| 实验时间 | 2021.04.25 | 机位号 | 18 | 实验设备号 | 21 |

1. **实验目的与要求**
2. 实验目的

熟悉ISE/Vivado工具软件的编程环境和参数设置；

掌握ISE/ Vivado工具软件上打开一个工程、编译、仿真、下载bit文件以及在HDU-XL-01实验板卡调试的方法；

理解实验源文件代码，理解工程文件中管脚约束语句，掌握管脚约束的方法；

熟悉HDU-XL-01实验板卡结构。

1. 实验要求

1）阅读源代码

打开本实验工程文件，阅读源代码，理解程序功能。

2）语法检查

3）阅读测试代码

打开工程文件中仿真测试代码文件，并理解测试代码。

4）仿真、理解波形图

仿真，结合Verilog HDL语言源代码功能和测试代码理解波形图。

5）配置管脚

打开管脚约束文件，阅读拨码开关、按键开关、LED灯、数码管、系统时钟的管脚约束语句，掌握管脚的方法。理解压缩bit流的原因，掌握压缩bit流的方法。

6）逻辑综合、查看电路

逻辑综合，查看RTL电路图，结合Verilog HDL语言源代码功能理解之。

7）下载代码

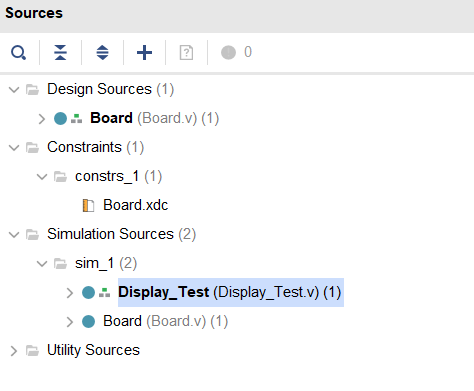
使用HDU-XL-01实验开发板专用的调试工具，将生成的bit文件下载到HDU-XL-01实验开发板上。

7）板卡实验

结合Verilog HDL语言源代码功能和管脚约束信号，在HDU-XL-01实验开发板上测试拨码开关、按键开关、LED灯、数码管、系统时钟的功能，检查外设完好率。

1. **实验设计与程序代码**
2. 模块设计说明

（描述这个实验的设计方案，分几个模块，各模块的功能，各模块之间的连接关系，可附图）



1. 实验程序源代码及注释等

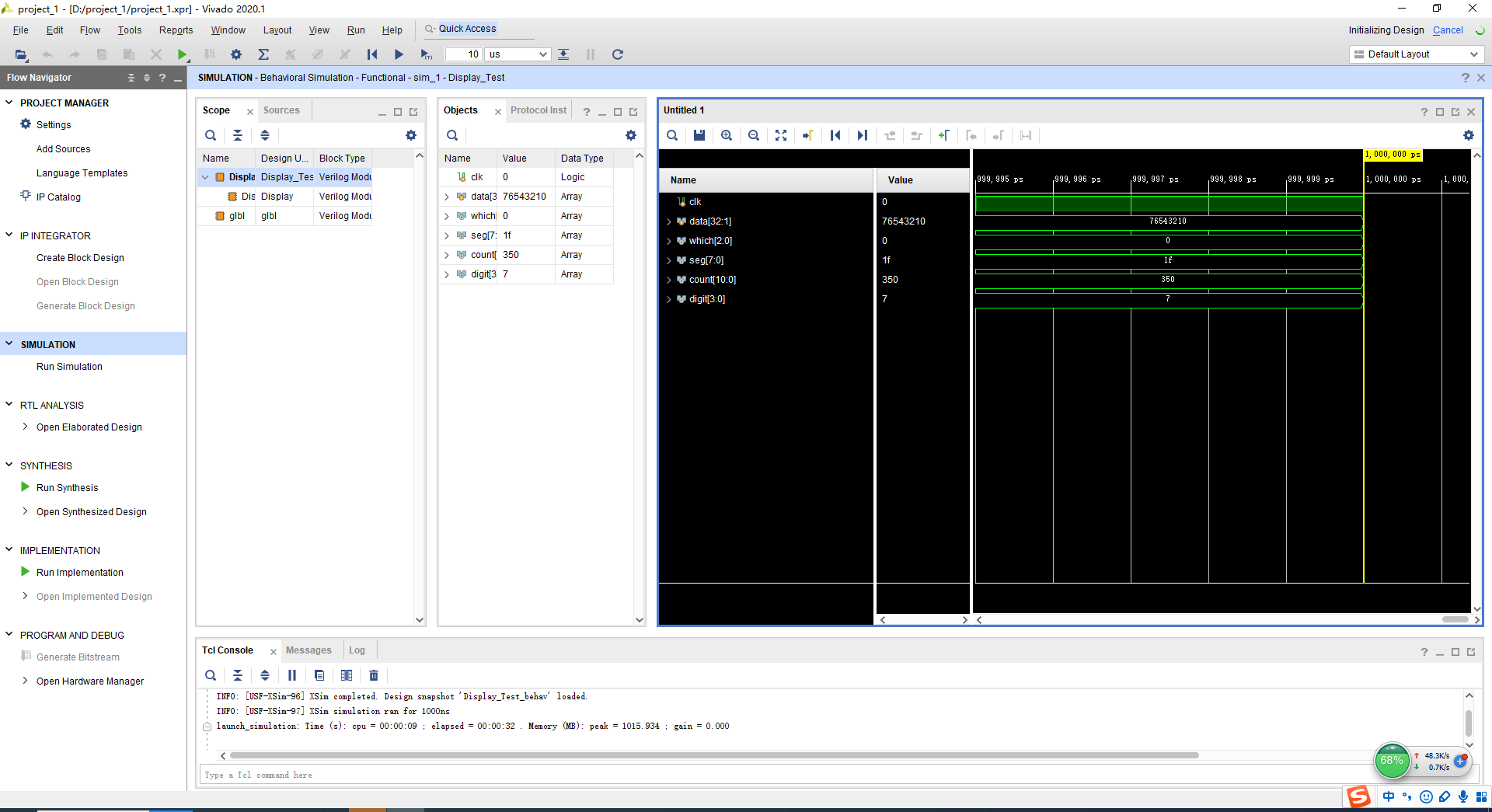
（实验各个模块的代码，包含功能注释）

|  |
| --- |
| 1. **Board.v**   `timescale 1ns / 1ps  // 通过数据输入输出测试开关、LED、数码管；通过数码管使能测试按钮  module Board(sw, swb, led, clk, which, seg, enable);  input [1:32] sw;  output [1:32] led;  assign led = sw; // 开关输入数据，直接输出到 LED  input clk; // 数码管相关  output [2:0] which;  output [7:0] seg;  output reg enable = 1; // 默认开启数码管使能  Display Display\_Instance(.clk(clk), .data(sw),  .which(which), .seg(seg));  input [1:6] swb;  assign toggle = |swb; // 按下任意按钮切换数码管使能  always @(posedge toggle) enable <= ~enable;  endmodule // Board   1. **Display.v**   `timescale 1ns / 1ps  // 8 位七段数码管扫描显示模块  module Display(clk, data, which, seg,  count, digit); // 调试接口  input clk; // 接入系统时钟  input [32:1] data; // 32 位显示数据  output reg [2:0] which = 0; // 片选编码（驱动哪一位数码管），低电平有效  output reg [7:0] seg; // 段选信号（点亮哪些笔划），低电平有效  output reg [10:0] count = 0; // 分频扫描，从左至右循环驱动每一位数码管  always @(posedge clk) count <= count + 1'b1;  always @(negedge clk) if (&count) which <= which + 1'b1;  output reg [3:0] digit; // 显示数据 片选得到 十六进制数码  always @\* case (which)  0: digit <= data[32:29]; // 最高位  1: digit <= data[28:25];  2: digit <= data[24:21];  3: digit <= data[20:17];  4: digit <= data[16:13];  5: digit <= data[12:09];  6: digit <= data[08:05];  7: digit <= data[04:01]; // 最低位  endcase  always @\* case (digit) // 十六进制数码 转换为 段选信号（a,b,c,...g,dp）  4'h0: seg <= 8'b0000\_0011; // 除 g、dp 外全亮，显示数码 0  4'h1: seg <= 8'b1001\_1111; // 仅 b、c 亮，显示数码 1  4'h2: seg <= 8'b0010\_0101;  4'h3: seg <= 8'b0000\_1101;  4'h4: seg <= 8'b1001\_1001;  4'h5: seg <= 8'b0100\_1001;  4'h6: seg <= 8'b0100\_0001;  4'h7: seg <= 8'b0001\_1111;  4'h8: seg <= 8'b0000\_0001;  4'h9: seg <= 8'b0000\_1001;  4'hA: seg <= 8'b0001\_0001;  4'hB: seg <= 8'b1100\_0001;  4'hC: seg <= 8'b0110\_0011;  4'hD: seg <= 8'b1000\_0101;  4'hE: seg <= 8'b0110\_0001;  4'hF: seg <= 8'b0111\_0001;  endcase  endmodule // Display |

1. **实验仿真**
2. 仿真代码（仿真源代码）

|  |
| --- |
| `timescale 1ns / 1ps  // 数码管扫描显示模块仿真测试  module Display\_Test();  reg clk = 0;  reg [32:1] data; // input  wire [2:0] which;  wire [7:0] seg; // output  wire [10:0] count;  wire [3:0] digit; // output for debug  // Instantiate a display module  Display Display\_Instance(.clk(clk), .data(data),  .which(which), .seg(seg),  .count(count), .digit(digit));  always #0.01 clk = ~clk; // 0.01ns == 10ps  initial begin  data = 32'hfedc\_ba98; #500;  data = 32'h7654\_3210; #500;  end  endmodule // Display\_Test |

1. 仿真波形（运行仿真时波形截图）

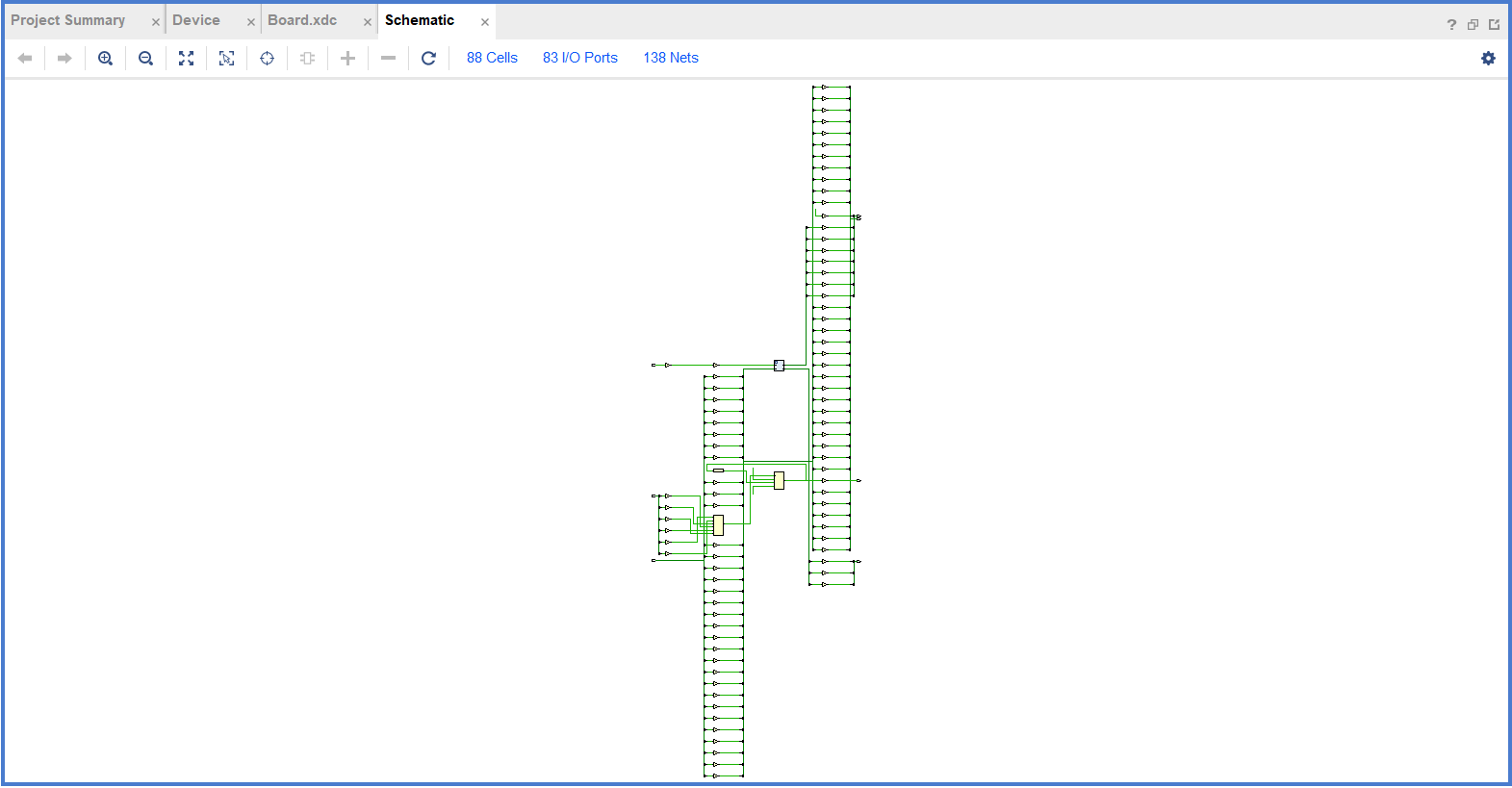


1. 仿真结果分析（对仿真波形进行分析）

这是一个单周期MIPS CPU，所有指令在一个周期内全部完成。

1. **电路图**

（开发工具中显示的电路模块图）



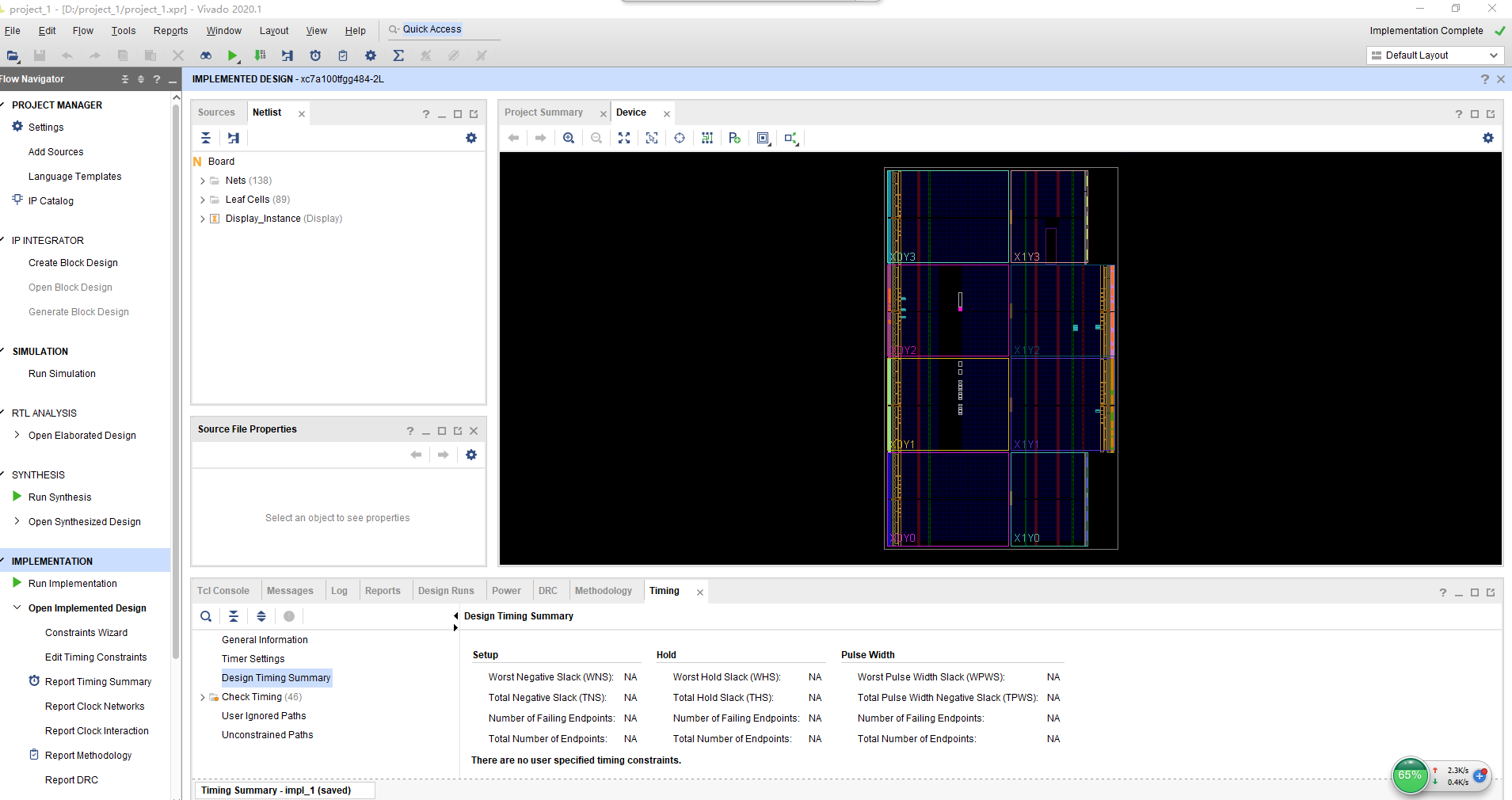
1. **引脚配置**

（引脚约束文件的内容，描述主要配置情况）

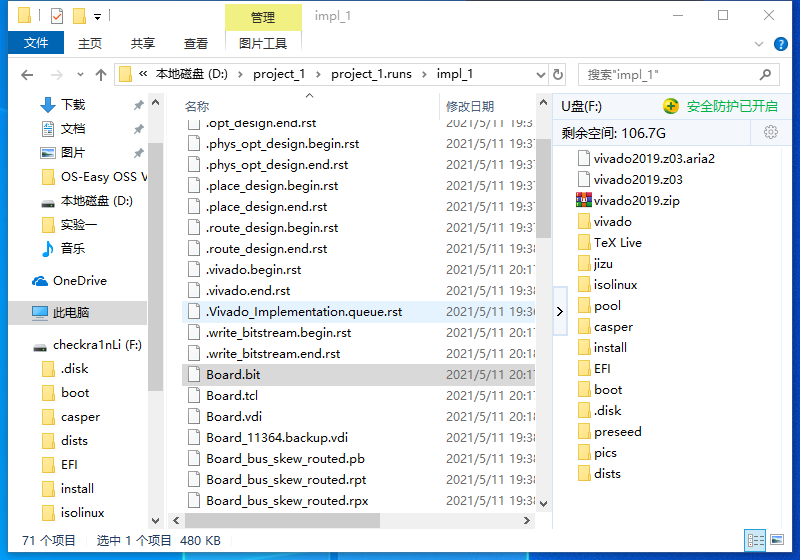
|  |
| --- |
| # 开启比特流压缩，优化 .bit 文件大小  set\_property BITSTREAM.GENERAL.COMPRESS TRUE [current\_design]  # Switch，开关  set\_property PULLDOWN true [get\_ports sw]  set\_property IOSTANDARD LVCMOS18 [get\_ports sw]  set\_property PACKAGE\_PIN T3 [get\_ports {sw[1]}]  set\_property PACKAGE\_PIN U3 [get\_ports {sw[2]}]  set\_property PACKAGE\_PIN T4 [get\_ports {sw[3]}]  set\_property PACKAGE\_PIN V3 [get\_ports {sw[4]}]  set\_property PACKAGE\_PIN V4 [get\_ports {sw[5]}]  set\_property PACKAGE\_PIN W4 [get\_ports {sw[6]}]  set\_property PACKAGE\_PIN Y4 [get\_ports {sw[7]}]  set\_property PACKAGE\_PIN Y6 [get\_ports {sw[8]}]  set\_property PACKAGE\_PIN W7 [get\_ports {sw[9]}]  set\_property PACKAGE\_PIN Y8 [get\_ports {sw[10]}]  set\_property PACKAGE\_PIN Y7 [get\_ports {sw[11]}]  set\_property PACKAGE\_PIN T1 [get\_ports {sw[12]}]  set\_property PACKAGE\_PIN U1 [get\_ports {sw[13]}]  set\_property PACKAGE\_PIN U2 [get\_ports {sw[14]}]  set\_property PACKAGE\_PIN W1 [get\_ports {sw[15]}]  set\_property PACKAGE\_PIN W2 [get\_ports {sw[16]}]  set\_property PACKAGE\_PIN Y1 [get\_ports {sw[17]}]  set\_property PACKAGE\_PIN AA1 [get\_ports {sw[18]}]  set\_property PACKAGE\_PIN V2 [get\_ports {sw[19]}]  set\_property PACKAGE\_PIN Y2 [get\_ports {sw[20]}]  set\_property PACKAGE\_PIN AB1 [get\_ports {sw[21]}]  set\_property PACKAGE\_PIN AB2 [get\_ports {sw[22]}]  set\_property PACKAGE\_PIN AB3 [get\_ports {sw[23]}]  set\_property PACKAGE\_PIN AB5 [get\_ports {sw[24]}]  set\_property PACKAGE\_PIN AA6 [get\_ports {sw[25]}]  set\_property PACKAGE\_PIN R2 [get\_ports {sw[26]}]  set\_property PACKAGE\_PIN R3 [get\_ports {sw[27]}]  set\_property PACKAGE\_PIN T6 [get\_ports {sw[28]}]  set\_property PACKAGE\_PIN R6 [get\_ports {sw[29]}]  set\_property PACKAGE\_PIN U7 [get\_ports {sw[30]}]  set\_property PACKAGE\_PIN AB7 [get\_ports {sw[31]}]  set\_property PACKAGE\_PIN AB8 [get\_ports {sw[32]}]  # Switch Button，按钮  set\_property IOSTANDARD LVCMOS18 [get\_ports swb]  set\_property PACKAGE\_PIN R4 [get\_ports {swb[1]}]  set\_property PACKAGE\_PIN AA4 [get\_ports {swb[2]}]  set\_property PACKAGE\_PIN AB6 [get\_ports {swb[3]}]  set\_property PACKAGE\_PIN T5 [get\_ports {swb[4]}]  set\_property PACKAGE\_PIN V8 [get\_ports {swb[5]}]  set\_property PACKAGE\_PIN AA8 [get\_ports {swb[6]}]  # LED  set\_property IOSTANDARD LVCMOS18 [get\_ports led]  set\_property PACKAGE\_PIN R1 [get\_ports {led[1]}]  set\_property PACKAGE\_PIN P2 [get\_ports {led[2]}]  set\_property PACKAGE\_PIN P1 [get\_ports {led[3]}]  set\_property PACKAGE\_PIN N2 [get\_ports {led[4]}]  set\_property PACKAGE\_PIN M1 [get\_ports {led[5]}]  set\_property PACKAGE\_PIN M2 [get\_ports {led[6]}]  set\_property PACKAGE\_PIN L1 [get\_ports {led[7]}]  set\_property PACKAGE\_PIN J2 [get\_ports {led[8]}]  set\_property PACKAGE\_PIN G1 [get\_ports {led[9]}]  set\_property PACKAGE\_PIN E1 [get\_ports {led[10]}]  set\_property PACKAGE\_PIN D2 [get\_ports {led[11]}]  set\_property PACKAGE\_PIN A1 [get\_ports {led[12]}]  set\_property PACKAGE\_PIN L3 [get\_ports {led[13]}]  set\_property PACKAGE\_PIN G3 [get\_ports {led[14]}]  set\_property PACKAGE\_PIN K4 [get\_ports {led[15]}]  set\_property PACKAGE\_PIN G4 [get\_ports {led[16]}]  set\_property PACKAGE\_PIN K1 [get\_ports {led[17]}]  set\_property PACKAGE\_PIN J1 [get\_ports {led[18]}]  set\_property PACKAGE\_PIN H2 [get\_ports {led[19]}]  set\_property PACKAGE\_PIN G2 [get\_ports {led[20]}]  set\_property PACKAGE\_PIN F1 [get\_ports {led[21]}]  set\_property PACKAGE\_PIN E2 [get\_ports {led[22]}]  set\_property PACKAGE\_PIN D1 [get\_ports {led[23]}]  set\_property PACKAGE\_PIN B1 [get\_ports {led[24]}]  set\_property PACKAGE\_PIN B2 [get\_ports {led[25]}]  set\_property PACKAGE\_PIN N3 [get\_ports {led[26]}]  set\_property PACKAGE\_PIN M3 [get\_ports {led[27]}]  set\_property PACKAGE\_PIN K3 [get\_ports {led[28]}]  set\_property PACKAGE\_PIN H3 [get\_ports {led[29]}]  set\_property PACKAGE\_PIN N4 [get\_ports {led[30]}]  set\_property PACKAGE\_PIN L4 [get\_ports {led[31]}]  set\_property PACKAGE\_PIN J4 [get\_ports {led[32]}]  # 数码管相关  set\_property IOSTANDARD LVCMOS18 [get\_ports seg]  set\_property PACKAGE\_PIN H19 [get\_ports {seg[7]}]  set\_property PACKAGE\_PIN G20 [get\_ports {seg[6]}]  set\_property PACKAGE\_PIN J22 [get\_ports {seg[5]}]  set\_property PACKAGE\_PIN K22 [get\_ports {seg[4]}]  set\_property PACKAGE\_PIN K21 [get\_ports {seg[3]}]  set\_property PACKAGE\_PIN H20 [get\_ports {seg[2]}]  set\_property PACKAGE\_PIN H22 [get\_ports {seg[1]}]  set\_property PACKAGE\_PIN J21 [get\_ports {seg[0]}]  set\_property IOSTANDARD LVCMOS18 [get\_ports which]  set\_property PACKAGE\_PIN N22 [get\_ports {which[0]}]  set\_property PACKAGE\_PIN M21 [get\_ports {which[1]}]  set\_property PACKAGE\_PIN M22 [get\_ports {which[2]}]  set\_property -dict {IOSTANDARD LVCMOS18 PACKAGE\_PIN L21} [get\_ports enable]  set\_property -dict {IOSTANDARD LVCMOS18 PACKAGE\_PIN H4} [get\_ports clk]  # [Place 30-574] Poor placement for routing between an IO pin and BUFG.If this  # sub optimal condition is acceptable for this design, you may use the  # CLOCK\_DEDICATED\_ROUTE constraint in the .xdc file to demote this message to a  # WARNING. However, the use of this override is highly discouraged.  set\_property CLOCK\_DEDICATED\_ROUTE FALSE [get\_nets clk\_IBUF]  set\_property CLOCK\_DEDICATED\_ROUTE FALSE [get\_nets swb\_IBUF[1]]  set\_property CLOCK\_DEDICATED\_ROUTE FALSE [get\_nets swb\_IBUF[2]]  set\_property CLOCK\_DEDICATED\_ROUTE FALSE [get\_nets swb\_IBUF[3]]  set\_property CLOCK\_DEDICATED\_ROUTE FALSE [get\_nets swb\_IBUF[4]]  set\_property CLOCK\_DEDICATED\_ROUTE FALSE [get\_nets swb\_IBUF[5]]  set\_property CLOCK\_DEDICATED\_ROUTE FALSE [get\_nets swb\_IBUF[6]] |

1. **思考与探索**
2. 实验结果记录：

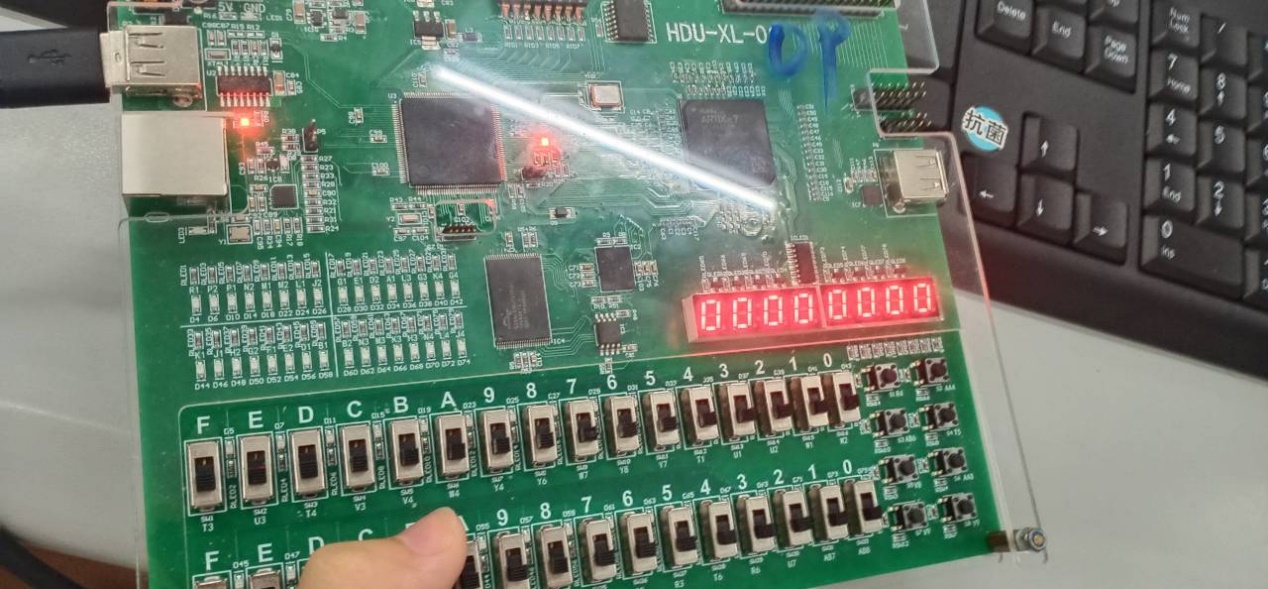
右侧显示为7654321



找到对应生成的文件Board.bit



将文件导入对应进行板卡烧录



板卡显示结果如上图

（实验操作的过程及结果记录）

1. 实验结论

（分析实验结果，给出实验结论）

第一次测试成功将生成的bit文件下载到HDU-XL-01实验开发板上。开发板上显示0000 0000.

1. 问题与解决方案：

（整个实验过程中发生了什么问题？你是如何解决的。）

答：实验过程中没有遇到问题。

4、思考与探索

（1）假如你正在实验室里上课，操作板级实验时出现开关“失灵”，或按下按键、拨动开关，却没有在LED灯和数码管上观察到预期的结果，你将如何处理？

答：可以改写约束文件，把开关/按键和对应LED灯/数码管分别约束到其他功能正常的开关/按键 /LED灯/数码管，进一步确定故障发生部件。也可以采用万用表测量疑似故障部件，协助确定是否故障；

若确认板卡故障，及时告知老师，指出故障所在，更换板卡继续实验。

（2）板级实验在插拔USB连接线时应注意什么？拨开关，按按键时应注意什么？实验操作时应该注意哪些方面以便保护板卡完好？

答: USB连接线在插入是需要注意方向，在遇到无法插入的情况，千万不要用力，换个方向就可以解决问题。并且在拔下后也不要马上接着就插入，等待5秒钟左右再插入。

按压按键是，不要用力过猛防止破坏原件，适当用力按压即可。

裸手操作板级实验时，只应接触开关、按键等外设部件，特别不应触摸板卡正反面的芯片管脚、接线柱等部分。手持板卡时，应尽量拿板卡边缘无元器件处;