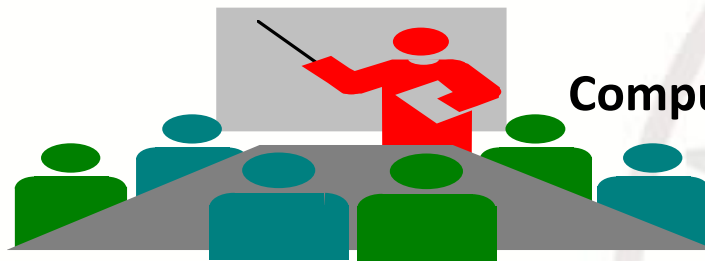




浙江大学
ZHEJIANG UNIVERSITY



Computer Organization & Design

Computer Organization & Design

实验与课程设计

实验三

RISC-V集成SOC设计

---优化CPU调试、测试和应用环境
-逻辑实验模块优化三

施青松

Asso. Prof. Shi Qingsong

College of Computer Science and Technology, Zhejiang University

zjsqs@zju.edu.cn



Course Outline



实验目的



1. 复习寄存器传输控制技术
2. 掌握CPU的核心组成：数据通路与控制
3. 设计数据通路的部件：优化逻辑ALU与Regs
4. 进一步了解计算机系统的基本结构
5. 熟练掌握IP核的使用方法

□ 实验设备

1. 计算机（Intel Core i5以上，4GB内存以上）系统
2. 计算机软硬件课程贯通教学实验系统(Sword)
3. Xilinx ISE14.4及以上开发工具

□ 材料

无

计算机软硬件课程贯通教学实验系统

贯通教学实验平台主要参数

▼ 核心芯片

Xilinx Kintex™-7系列的XC7K160/325资源:

162,240个, Slice: 25350, 片内存储: 11.7Mb

▼ 存储体系 支持32位存储层次体系结构

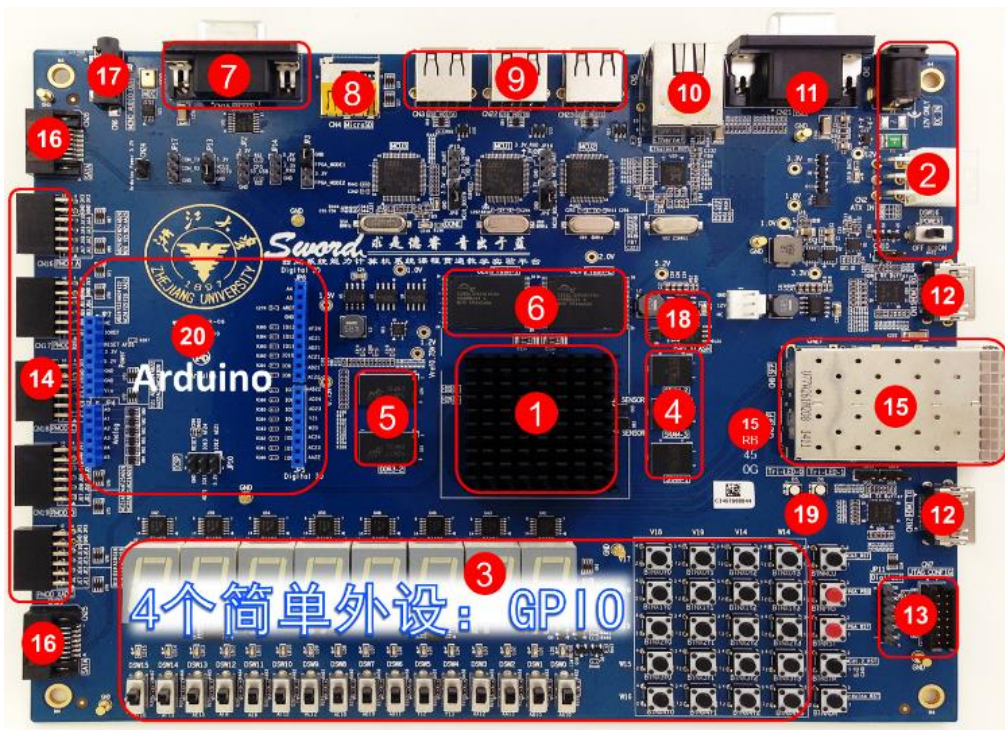
6MB SRAM静态存储器: 支持32Data, 16位TAG

512M BDDR3动态存储: 支持32Data

32MB NOR Flash存储: 支持32位Data

▼ 基本接口 支持微机原理、SOC或微处理器简单应用

4×5+1矩阵按键、16位滑动开关、16位LED、8位七段数码管



▼ 标准接口 支持基本计算机系统实现

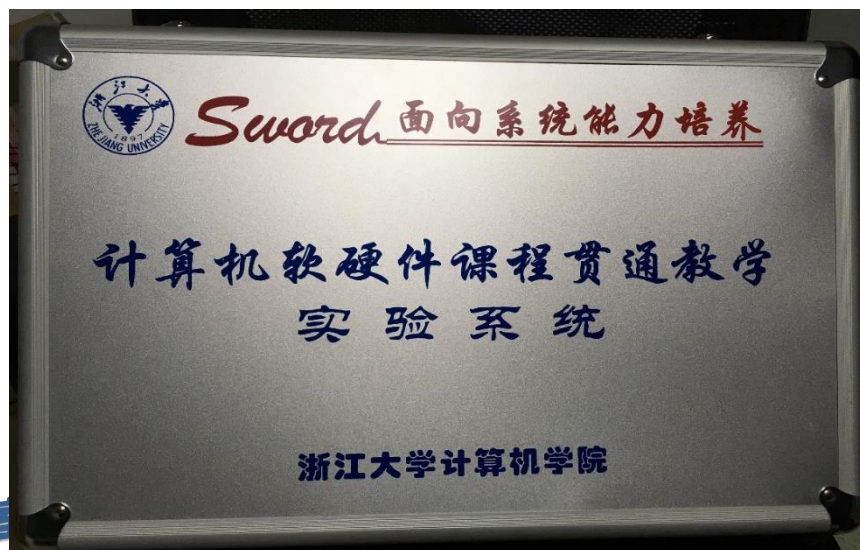
12位VGA接口 (RGB656)、USB-HID (键盘)

▼ 通讯接口 支持数据传输、调试和网络

UART接口、10M/100M/1000M以太网、SFP光纤接口

▼ 扩展接口 支持外存、多媒体和个性化设备

MicroSD(TF)、PMOD、HDMI、Arduino





Course Outline



1. 优化计算机系统测试环境CSTE

- 搭建HDL测试环境
 - 用HDL描述和RISC-V等网表核重建实验二测试环境
- 用RISC-V核替换实验二的MIPS CPU核
 - 选用教材提供的IP核集成实现SOC
 - 此实验在Exp02的基础上完成

2. 优化数据通路子部件并作时序仿真:

- ALU
- Register Files



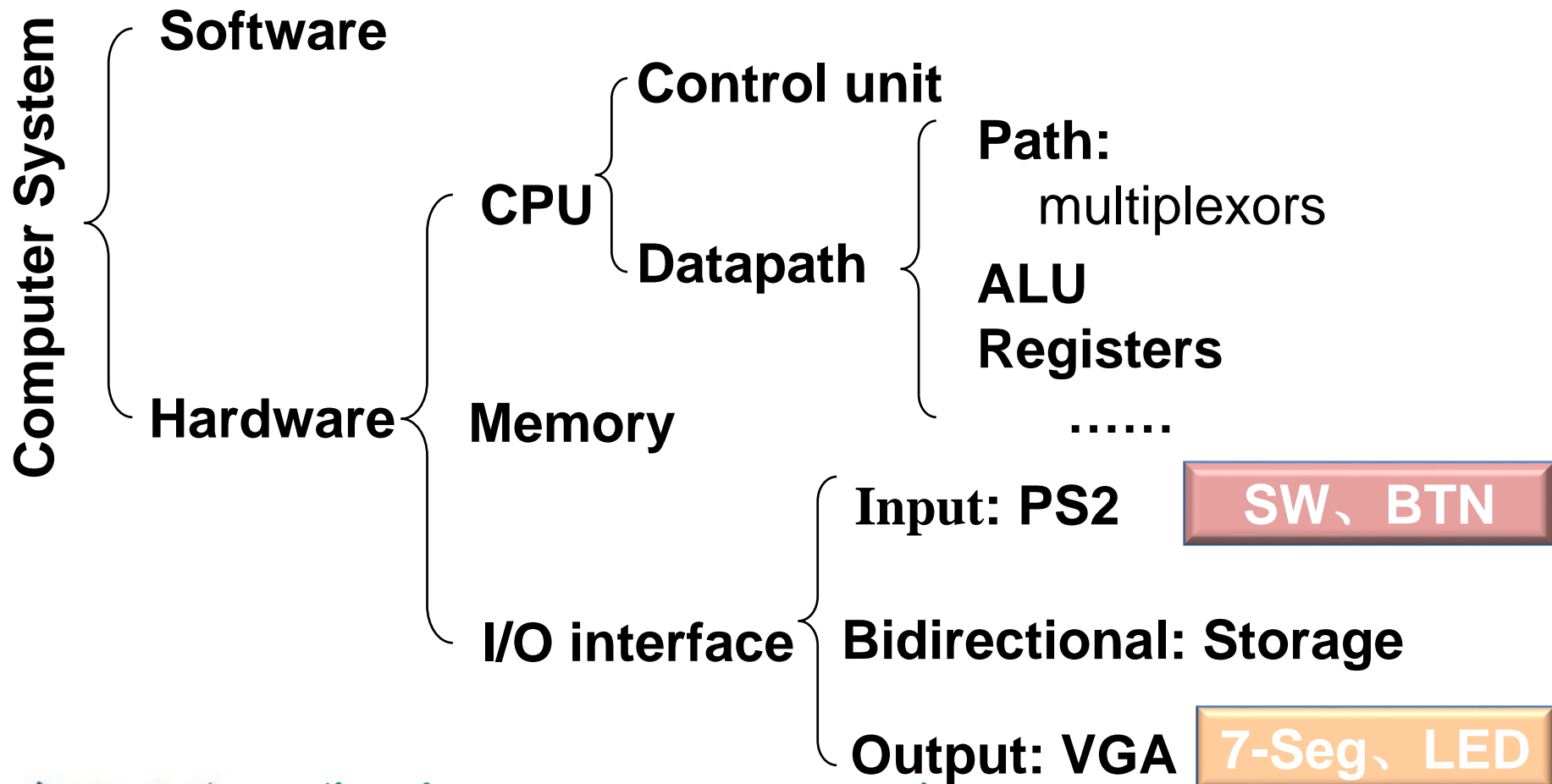
Course Outline





Computer Organization

□ Decomposability of computer systems



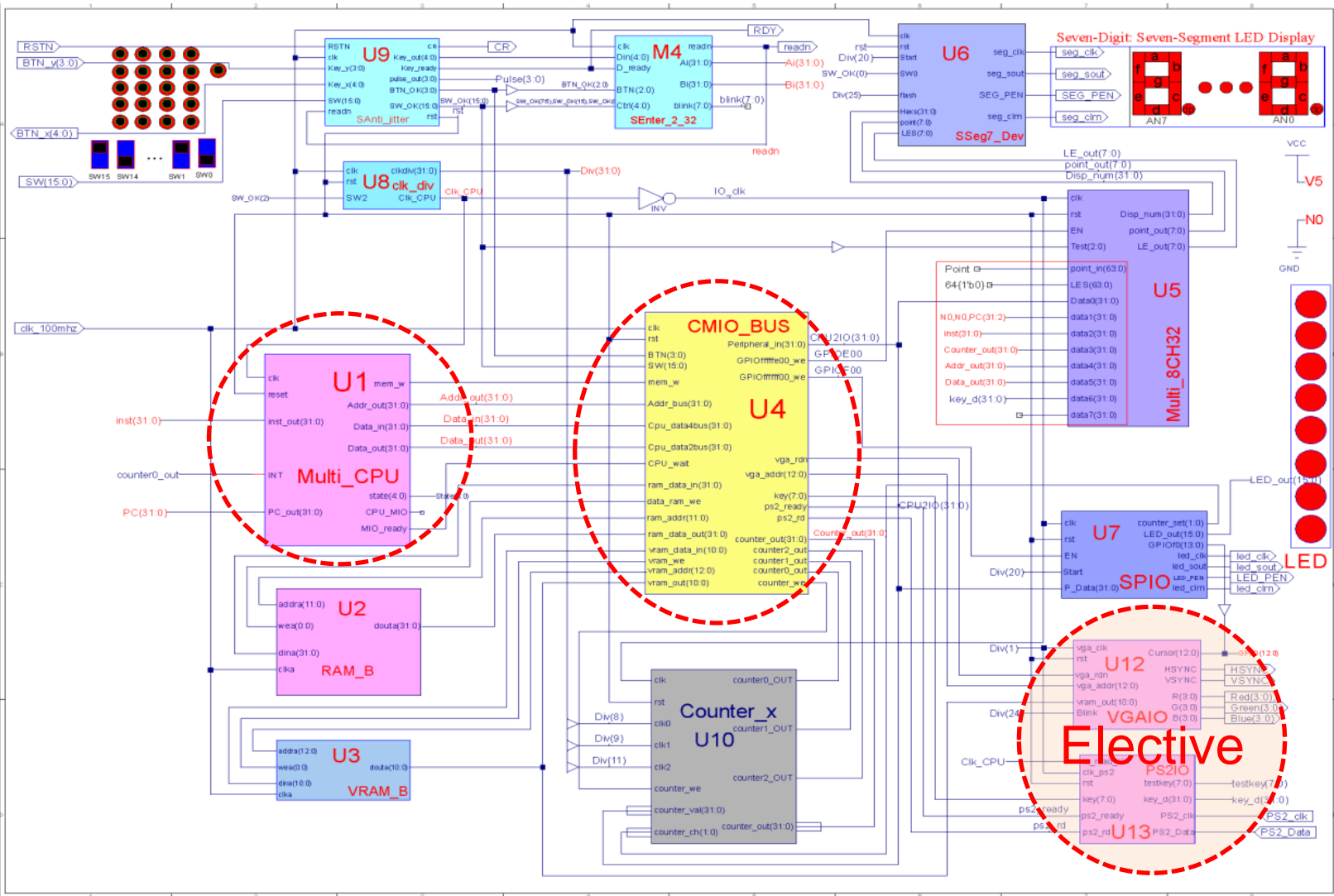
简单的SOC(ISE版)



Top-Down

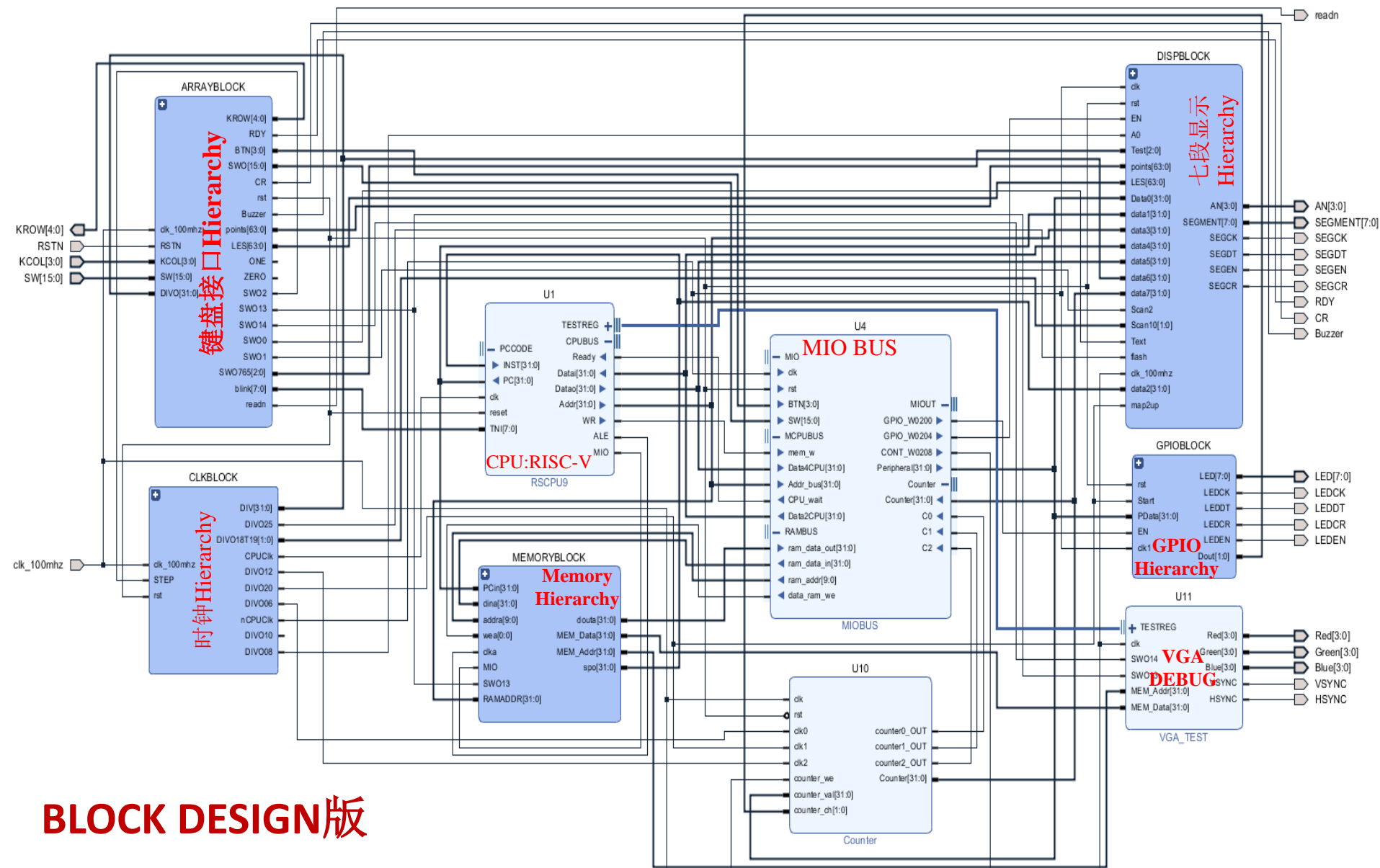
核建立实验环境

Bottom-Up: 逐个设计集成替换



简单计算机测试环境CSTE

--是简单SOC/功能计算机系统





流水处理器测试SOC或计算机系统

```
module CSTE_RISCVEDF(input clk_100mhz,
    input RSTN,
    input [15:0]SW,
    input [3:0]KCOL,
    output[4:0]KROW,
    output CR,
    output RDY,
    output readn,
    output LEDCK,
    output LEDCR,
    output LEDEN,
    output LEDDT,
    output SEGCK,
    output SEGCR,
    output SEGEN,
    output SEGDT,
    output [3:0]AN,
    output [7:0]SEGMENT,
    output [7:0]LED,
    output Buzzer,
    output [3:0]Red,
    output [3:0]Green,
    output [3:0]Blue,
    output HSYNC,
    output VSYNC
);
```

搭建简单的SOC测试处理器

HDL DESIGN版



系统IP核：RISC-V处理器

◎ 实验二是MIPS处理器

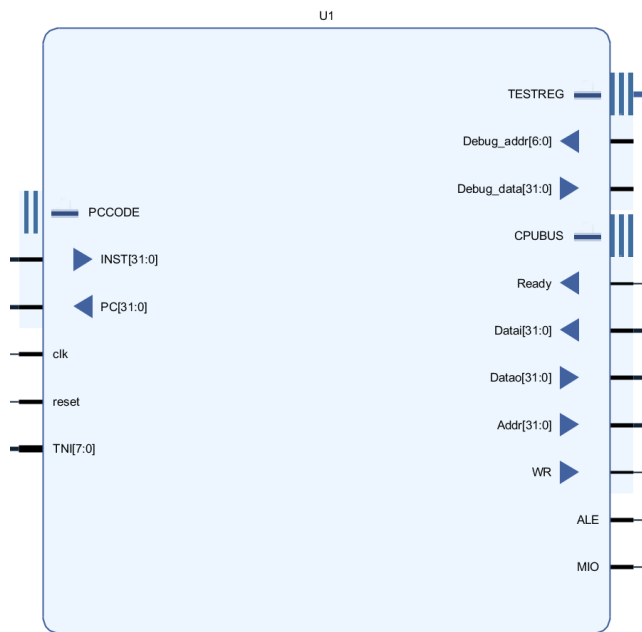
系统IP核有优化调整，详细参见实验二PPT

◎ 本课程采用RISC-V

☞ 单周期和流水线

RSCPU9

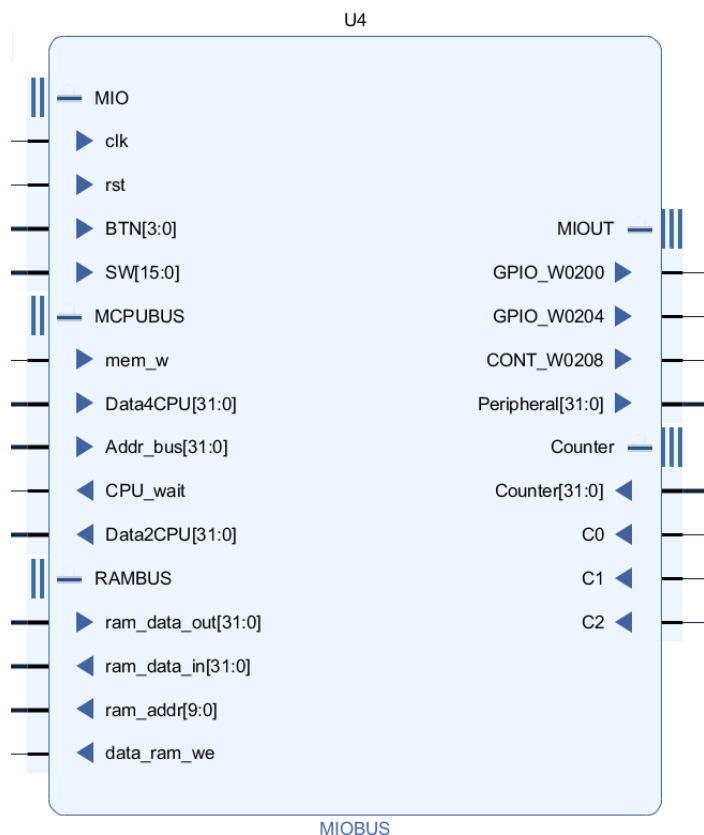
```
U1(.clk(CPUClk),
    .reset(rst),
    .TNI(8'h0),
    .Ready(Ready),
    .Addr(Addr),
    .Datai(Datai),
    .Datao(Datao),
    .INST(inst),
    .MIO(MIO),
    .PC(PC[31:0]),
    .WR(WR),
    .ALE(clka),
    .Debug_addr(Debug_addr),
    .Debug_data(Debug_data)
);
```





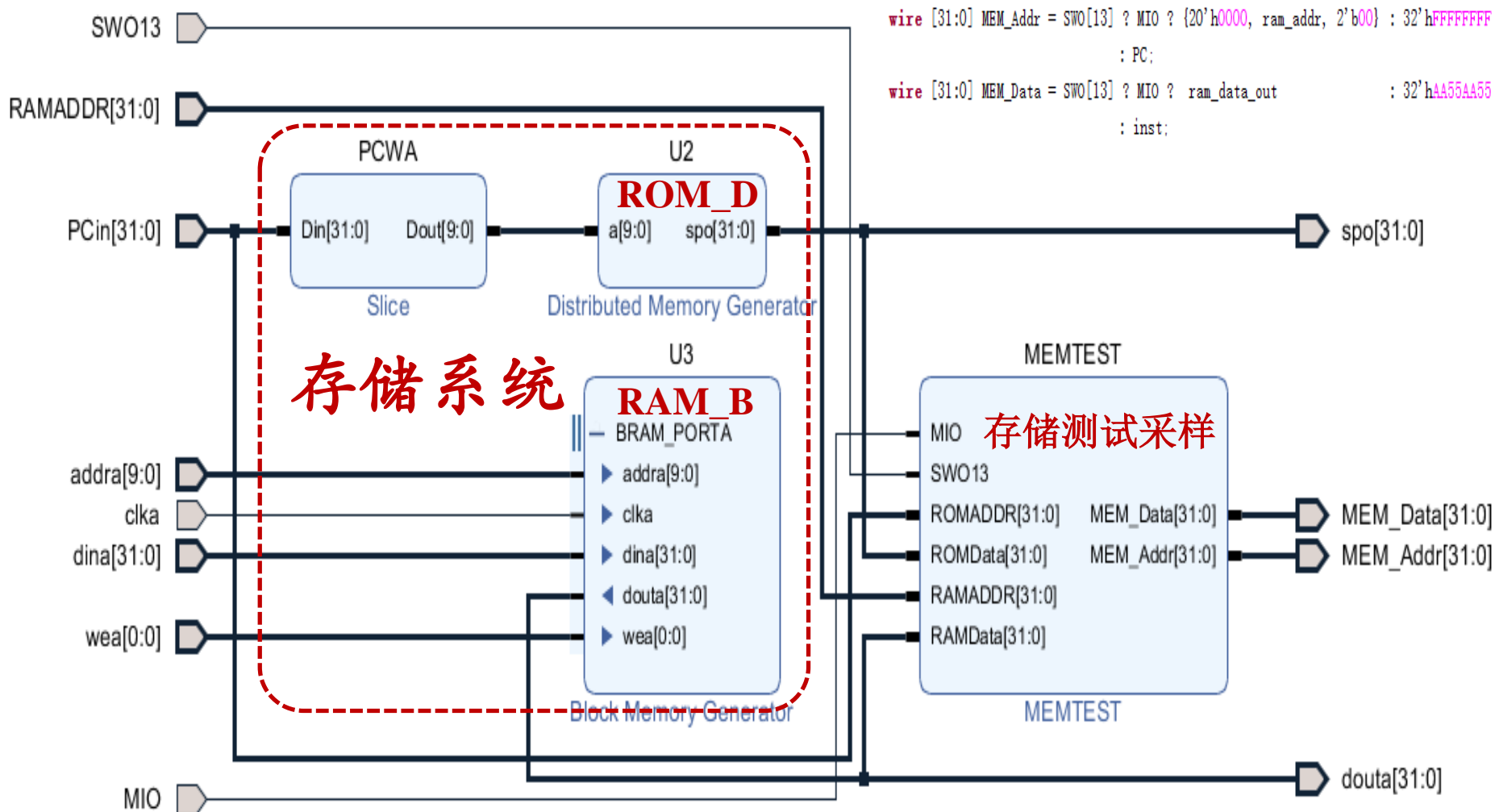
系统IP核：总线译码模块MIOBUS

简单地址译码电路

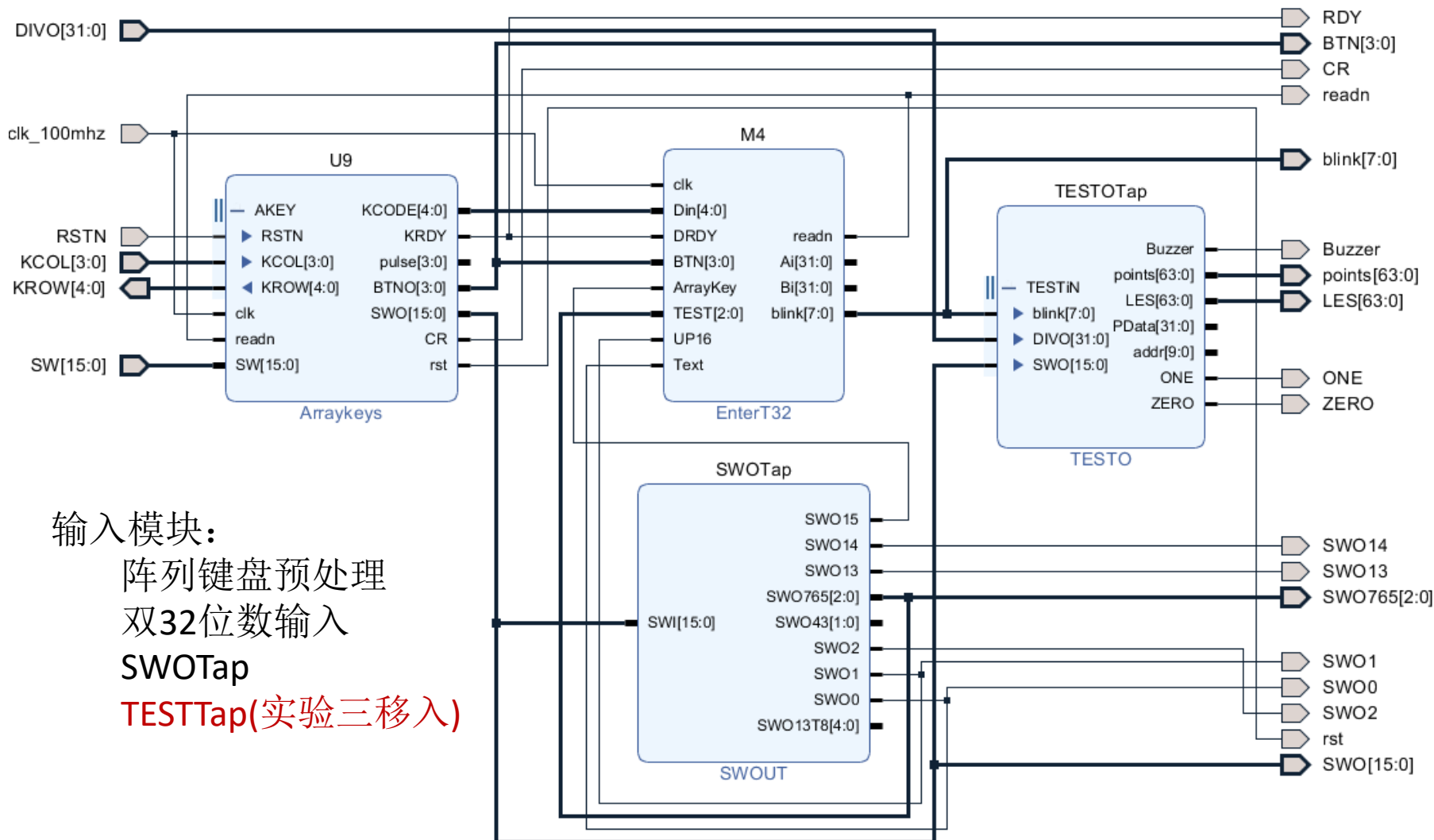


```
MIOBUS    U4(. clk(clk_100mhz),  
              .rst(rst),  
              .BTN(BTN0),  
              .SW(SW0),  
              .CPU_wait(Ready),  
              .mem_w(WR),  
              .Addr_bus(Addr),  
              .Data4CPU(Datao),  
              .Data2CPU(Datai[31:0]),  
  
              .GPIO_W0200(GPIO_W0200),  
              .GPIO_W0204(GPIO_W0204),  
              .CONT_W0208(CONT_W0208),  
              .Peripheral(Peripheral),  
              .C0(C0),  
              .C1(C1),  
              .C2(C2),  
              .Counter(Counter),  
              .ram_addr(ram_addr),  
              .data_ram_we(data_ram_we),  
              .ram_data_out(ram_data_out),  
              .ram_data_in(ram_data_in)  
              );
```


系统IP核：存储器(Memory Hierarchy)



系统IP核: ARRAYBLOCK Hierarchy



输入模块:

阵列键盘预处理

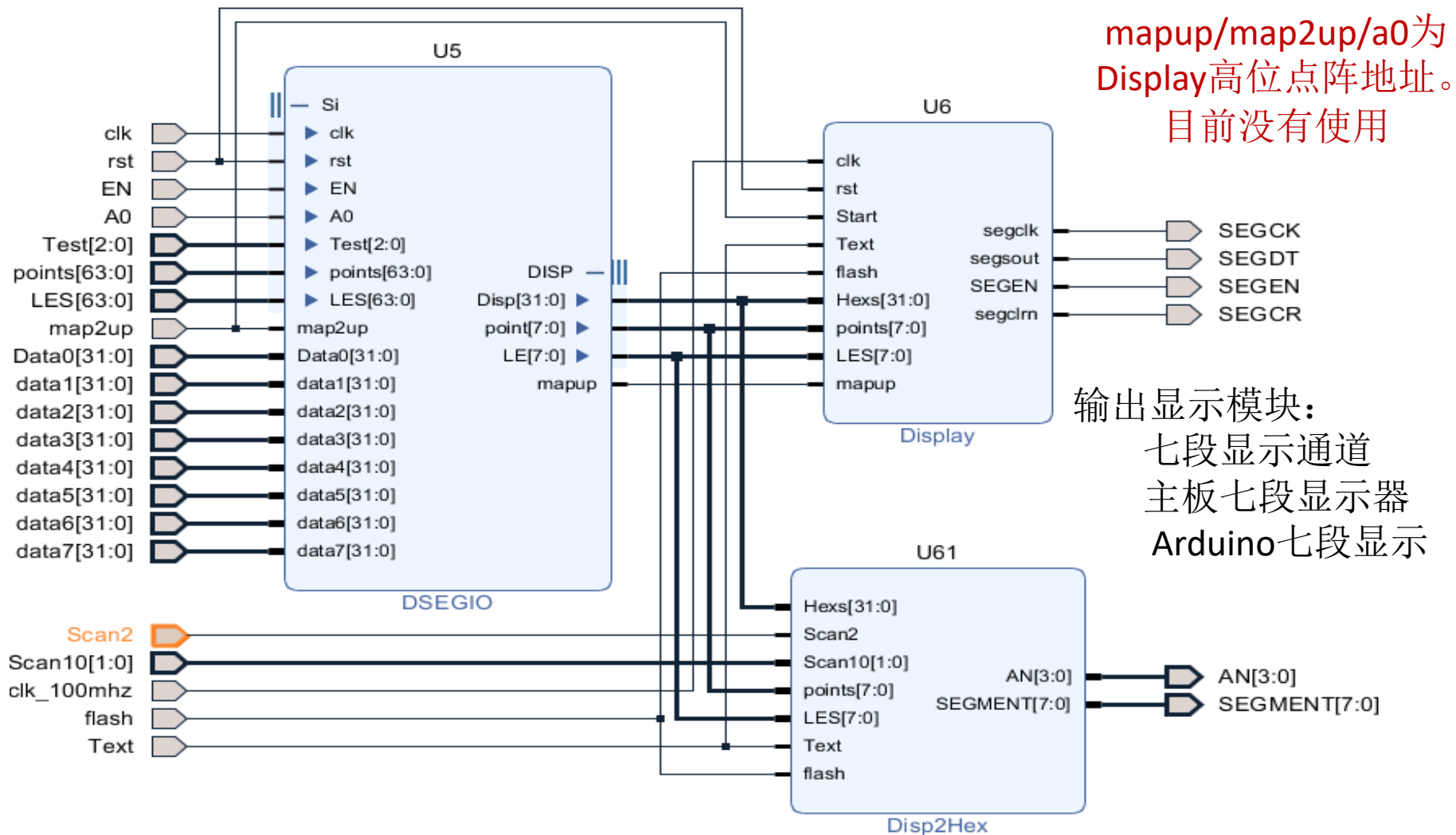
双32位数输入

SWOTap

TESTTap(实验三移入)



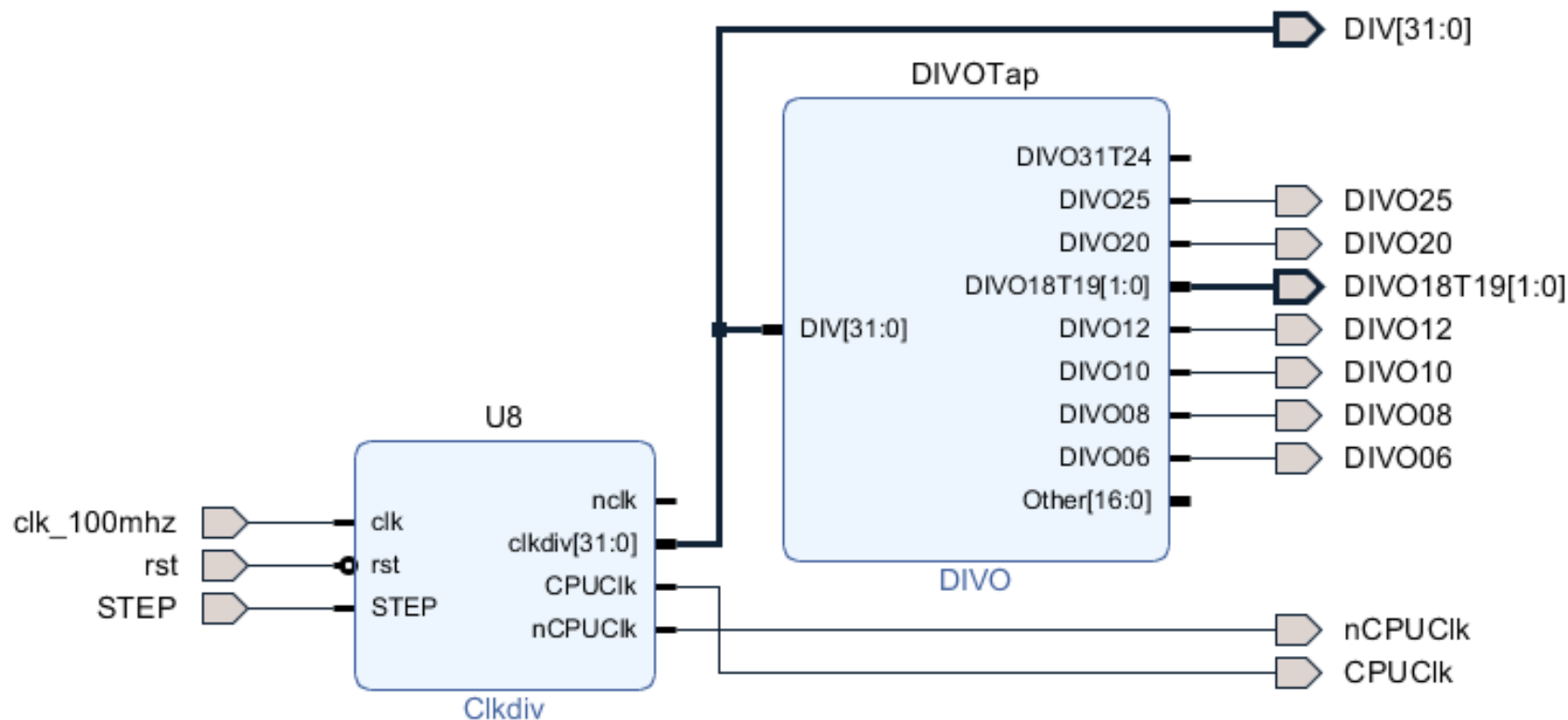
系统IP核: DISPBLOCK Hierarchy





系统IP核: CLKBLOCK Hierarchy

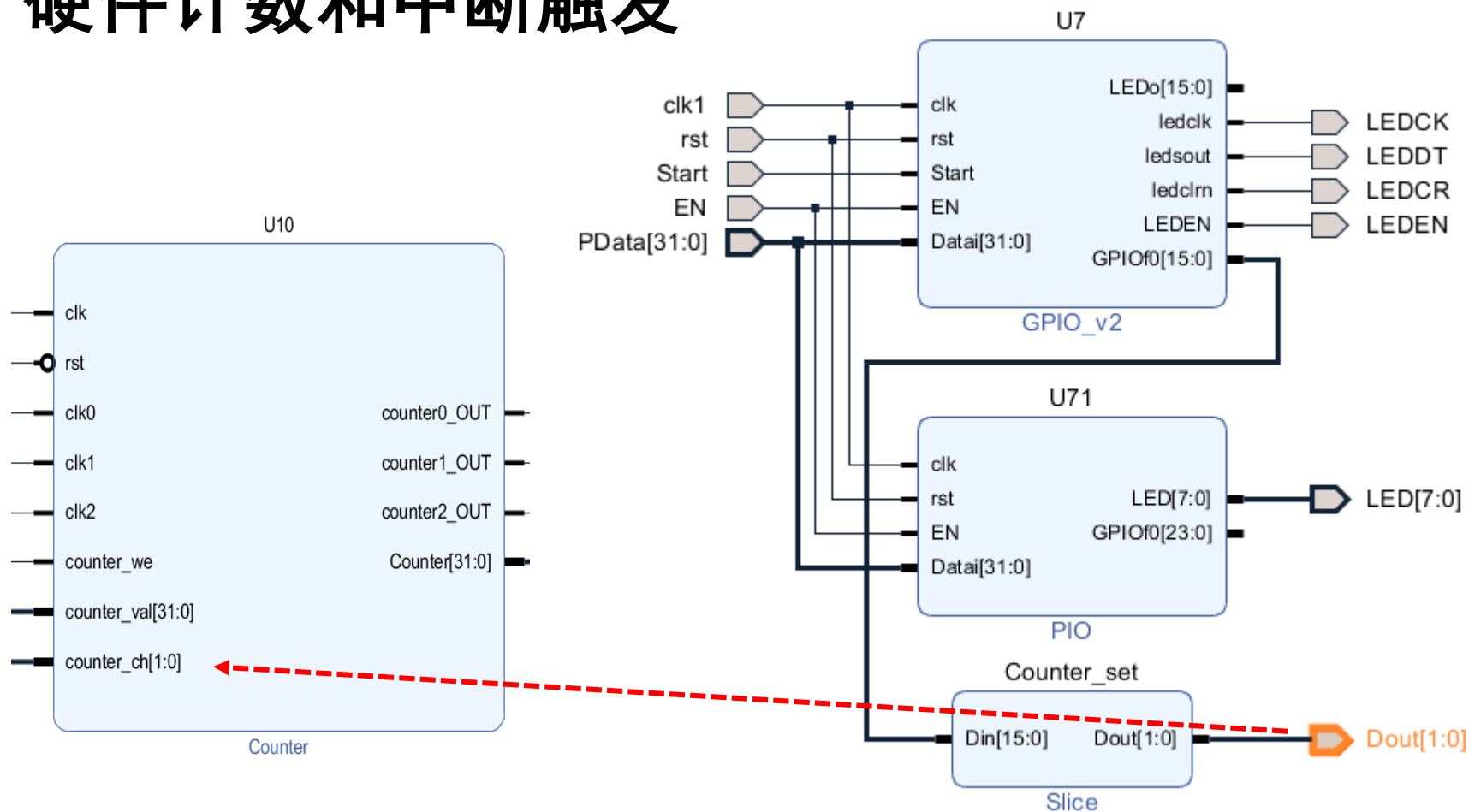
系统时钟模块



系统IP核： 硬件计数与GPIO模块



硬件计数和中断触发

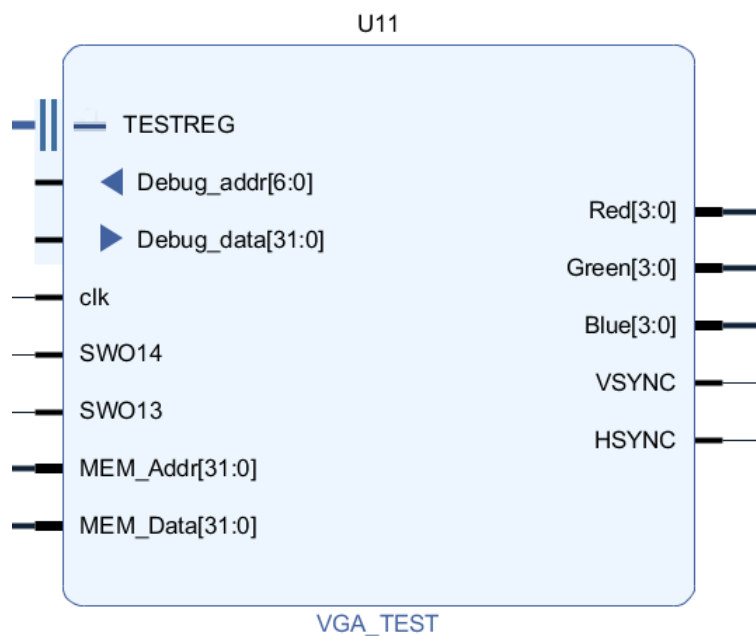




系统IP核： 系统测试模块

寄存器、存储器和CPU接口信号显示

在实验二基础上优化：增加了RAM测试端口，同时作了模块合并(与实验二不同)



```
VGA_TEST    U11(.clk(clk_100mhz),  
                //      .PCol(PCol),  
                //      .PRow(PRow),  
                .Debug_addr(Debug_addr),  
                .Debug_data(Debug_data),  
                .MEM_Addr(MEM_Addr),  
                .MEM_Data(MEM_Data),  
                .SW014(SW0[14]),  
                .SW013(SW0[13]),  
                //      .dout(dout),  
                .Red(Red),  
                .Green(Green),  
                .Blue(Blue),  
                .VSYNC(VSYNC),  
                .HSYNC(HSYNC)  
                );
```




HDL测试环境框架结构

```
1 module CSTE_RISCVEDF(input clk_100mhz,
2                       input RSTN,
3                       input [15:0]SW,
4                       input [3:0]KCOL,
5                       output[4:0]KROW,
6                       output CR,
7                       output RDY,
8                       output readn,
9
10                      output LEDCK,           //Sword LED
11                      output LEDCR,
12                      output LEDEN,
13                      output LEDDT,
14
15                      output SEGCK,           //Sword Seven-segment LED
16                      output SEGCR,
17                      output SEGEN,
18                      output SEGDT,
19
20                      output [3:0]AN,         //arduino Seven-segment LED
21                      output [7:0]SEGMENT,
22                      output [7:0]LED,       //arduino LED
23                      output Buzzer,        //arduino Buzzer
24
25                      output [3:0]Red,       //VGA
26                      output [3:0]Green,
27                      output [3:0]Blue,
28                      output HSYNC,
29                      output VSYNC
30 );
31
32 .....
33 .....
34 endmodule
```

调用系统分解网表核

11个主模块: U1~U11





处理器、存储器调用与信号定义

```
69     assign Buzzer = DIVO[25] & SWO[8];
70     assign IO_clk = nCPUClk;
71
72     RSCPU9          U1 (.clk(CPUClk), //RISV-V Single cycle CPU
73         .reset(rst),
74         .TNI(8'h0),
75         .Ready(Ready),
76         .Addr(Addr),
77         .Datai(Datai),
78         .Datao(Datao),
79         .INST(inst),
80         .MIO(MIO),
81         .PC(PC[31:0]),
82         .WR(WR),
83         .ALE(clka),
84         .Debug_addr(Debug_addr),
85         .Debug_data(Debug_data)
86     );
87
88     ROM_D          U2 (.a(PC[11:2]),
89         .spo(inst));
90
91     RAM_B          U3 (.addra(ram_addr[11:0]), //Data Memory
92         .clka(clka),
93         .dina(ram_data_in[31:0]),
94         .wea(data_ram_we[0]),
95         .douta(ram_data_out)
96     );
97
98     wire [31:0] MEM_Addr = SWO[13] ? MIO ? {20'h0000, ram_addr, 2'b00} : 32'hFFFFFFFF
99                          : PC;
100    wire [31:0] MEM_Data = SWO[13] ? MIO ? ram_data_out : 32'hAA55AA55
101                          : inst;
```

```
32    wire mapup;
33    wire [31:0] Ai, Bi;
34    wire CPUClk, nCPUClk, nclk, clka;
35    wire MIO;
36    wire C0, C1, C2;
37    wire [31:0] Counter;
38    wire [31:0] Addr, Datai, Datao, inst;
39    wire CONT_W0208, GPIO_W0200, GPIO_W0204;
40    wire [31:0] Peripheral;
41    wire [3:0] BTNO;
42    wire [4:0] KCODE;
43    wire [6:0] Debug_addr;
44    wire [7:0] blink;
45    wire [9:0] PCol, PRow;
46    wire [11:0] dout;
47    wire [15:0] GPIOF0;
48
49    wire [31:0] Debug_data;
50    wire [31:0] DIVO;
51    wire [31:0] Disp_num;
52    wire [0:0] data_ram_we;
53    wire GPIOE00;
54    wire GPIOF0;
55    wire IO_clk;
56    wire [7:0] LE;
57    wire [31:0] PC;
58    wire [7:0] point;
59    wire [3:0] Pulse;
60    wire [11:0] ram_addr;
61    wire [31:0] ram_data_in;
62    wire [31:0] ram_data_out;
63    wire Ready;
64    wire rst;
65    wire [4:0] State;
66    wire [15:0] SWO;
67    wire WR;
```



总线模块调用

```
102  MIOBUS      U4(.clk(clk_100mhz),           //IOBUS
103              .rst(rst),
104              .BTN(BTNO),
105              .SW(SWO),
106              .CPU_wait(Ready),
107              .mem_w(WR),
108              .Addr_bus(Addr),
109              .Data4CPU(Datao),
110              .Data2CPU(Datai[31:0]),
111
112              .GPIO_W0200(GPIO_W0200),
113              .GPIO_W0204(GPIO_W0204),
114              .CONT_W0208(CONT_W0208),
115              .Peripheral(Peripheral),
116              .C0(C0),
117              .C1(C1),
118              .C2(C2),
119              .Counter(Counter),
120              .ram_addr(ram_addr),
121              .data_ram_we(data_ram_we),
122              .ram_data_out(ram_data_out),
123              .ram_data_in(ram_data_in)
124              );
```



七段显示通道调用

```
126 DSEGIO U5(.clk(IO_clk), //Seven segment display channel
127 .rst(rst),
128 .EN(GPIO_W0204), //GPIOE00
129 .A0(DIVO[6]),
130 .map2up(DIVO[20]),
131 .Test(SWO[7:5]),
132 .LES(64'h0000_0000_0000_0000),
133 .points({DIVO[31:0], DIVO[31:13], 5'b00000, 8'h00}),
134 .Data0(Peripheral),
135 .data1({2'b00, PC[31:2]}),
136 .data2(inst),
137 .data3(Counter),
138 .data4(Addr),
139 .data5(Data0),
140 .data6(ram_data_out[31:0]),
141 .data7(Ai),
142
143 .Disp(Disp_num[31:0]),
144 .LE(LE),
145 .point(point),
146 .mapup(mapup)
147 );
```



输出系统：七段显示器调用

```
149 //-----Peripheral
150 Display U6(.clk(clk_100mhz),           //Device 1: Seven-segment display
151           .rst(rst),
152           .Start(DIVO[20]),
153           .Text(SWO[0]),
154           .flash(DIVO[25]),
155           .Hexs(Disp_num),
156           .LES(LE),
157           .points(point),
158           .mapup(DIVO[20]),
159
160           .segclk(SEGCK),
161           .segclrn(SEGCR),
162           .SEGEN(SEGEN),
163           .segsout(SEGDT)
164           );
165
166 Disp2Hex U61(.Scan2(SWO[1]),           //Arduino seven-segment display
167             .Scan10(DIVO[19:18]),
168             .flash(DIVO[25]),
169             .Text(SWO[0]),
170             .LES(LE),
171             .points(point),
172             .Hexs(Disp_num),
173
174             .AN(AN),
175             .SEGMENT(SEGMENT)
176             );
```

缺省地址：0xE000_0000



输出系统：GPIO和硬件计数调用

```
179 GPIO U7 (.clk(IO_clk), //Device 2: GPIO
180 .rst(rst),
181 .EN(GPIO_W0200), //GPIOF00,General Purpose Input Output For LED
182 .Datai(Peripheral),
183 .Start(DIVO[20]),
184 .GPIOf0(GPIOf0),
185 .LEDo(),
186 .ledclk(LEDCK),
187 .ledclrn(LEDCLRN),
188 .LEDEN(LEDEN),
189 .ledsout(LEDSTOUT)
190 );
```

缺省地址：0xF000_0000

```
192 PIO U71(.clk(IO_clk), //Arduino GPIO
193 .rst(rst),
194 .EN(GPIO_W0200),
195 .Datai(Peripheral),
196 .LED(LED[7:0]),
197 .GPIOf0()
198 );
```

```
201 Counter U10(.clk(clk_100mhz), //Device 3: counter module
202 .clk0(DIVO[8]),
203 .clk1(DIVO[9]),
204 .clk2(DIVO[11]),
205 .counter_ch(GPIOf0[1:0]),
206 .counter_val(Peripheral),
207 .counter_we(CONT_W0208),
208 .rst(rst),
209 .Counter(Counter),
210 .counter0_OUT(C0),
211 .counter1_OUT(C1),
212 .counter2_OUT(C2)
213 );
```

缺省地址：0xF000_0004



辅助模块调用

```
215 //-----Auxiliary module
216 Clkdiv U8 (.clk(clk_100mhz), //General clock module
217 .rst(rst),
218 .STEP(SWO[2]),
219 .clkdiv(DIVO),
220 .nclk(nclk),
221 .CPUClk(CPUClk),
222 .nCPUClk(nCPUClk)
223 );
224
225 Arraykeys U9 (.clk(clk_100mhz), //Array keyboard
226 .rst(rst),
227 .RSTN(RSTN),
228 .KCOL(KCOL),
229 .KROW(KROW),
230 .SW(SW),
231 .BTNO(BTNO),
232 .pulse(Pulse),
233 .CR(CR),
234 .readn(readn),
235 .KCODE(KCODE),
236 .KRDY(RDY),
237 .SWO(SWO)
238 );
239
240 EnterT32 M4 (.clk(clk_100mhz), //Dual Data input module by Button
241 .BTN(BTNO[3:0]),
242 .ArrayKey(SWO[15]),
243 .TEST(SWO[7:5]),
244 .Text(SWO[0]),
245 .UP16(SWO[1]),
246 .Din(KCODE),
247 .DRDY(RDY),
248 .Ai(Ai),
249 .Bi(Bi),
250 .blink(blink),
251 .readn(readn)
252 );
```

HDL软核,便于测试调试修改



VGA测试信号显示模块调用

```
254 VGA_TEST U11(.clk(clk_100mhz),
255 //      .PCol(PCol),
256 //      .PRow(PRow),
257      .Debug_addr(Debug_addr),      //Register Debug addra
258      .Debug_data(Debug_data),      //Register Debug Data
259      .MEM_Addr(MEM_Addr),           //Memory Debug addra
260      .MEM_Data(MEM_Data),           //Memory Debug Data
261      .SW014(SWO[14]),               //SW014测试内存翻页
262      .SW013(SWO[13]),               //SW13=0测试ROM, SW13=1测试RAM
263 //      .dout(dout),
264      .Red(Red),
265      .Green(Green),
266      .Blue(Blue),
267      .VSYNC(VSYNC),
268      .HSYNC(HSYNC)
269 );
270
271 endmodule
```

注释掉信号为实验二优化后删除信号

◎ HDL测试环境描述对应BLOCK DESIGN

☞ Vivado BD无法表达信号纳入Tap模块



Course Outline





设计要点：建立设计工程

◎建立工程一

⌘ OExp03-CSTE H

- ⊙ 对应搭建HDL描述测试环境

◎建立工程二

⌘ OExp03-CSTEB

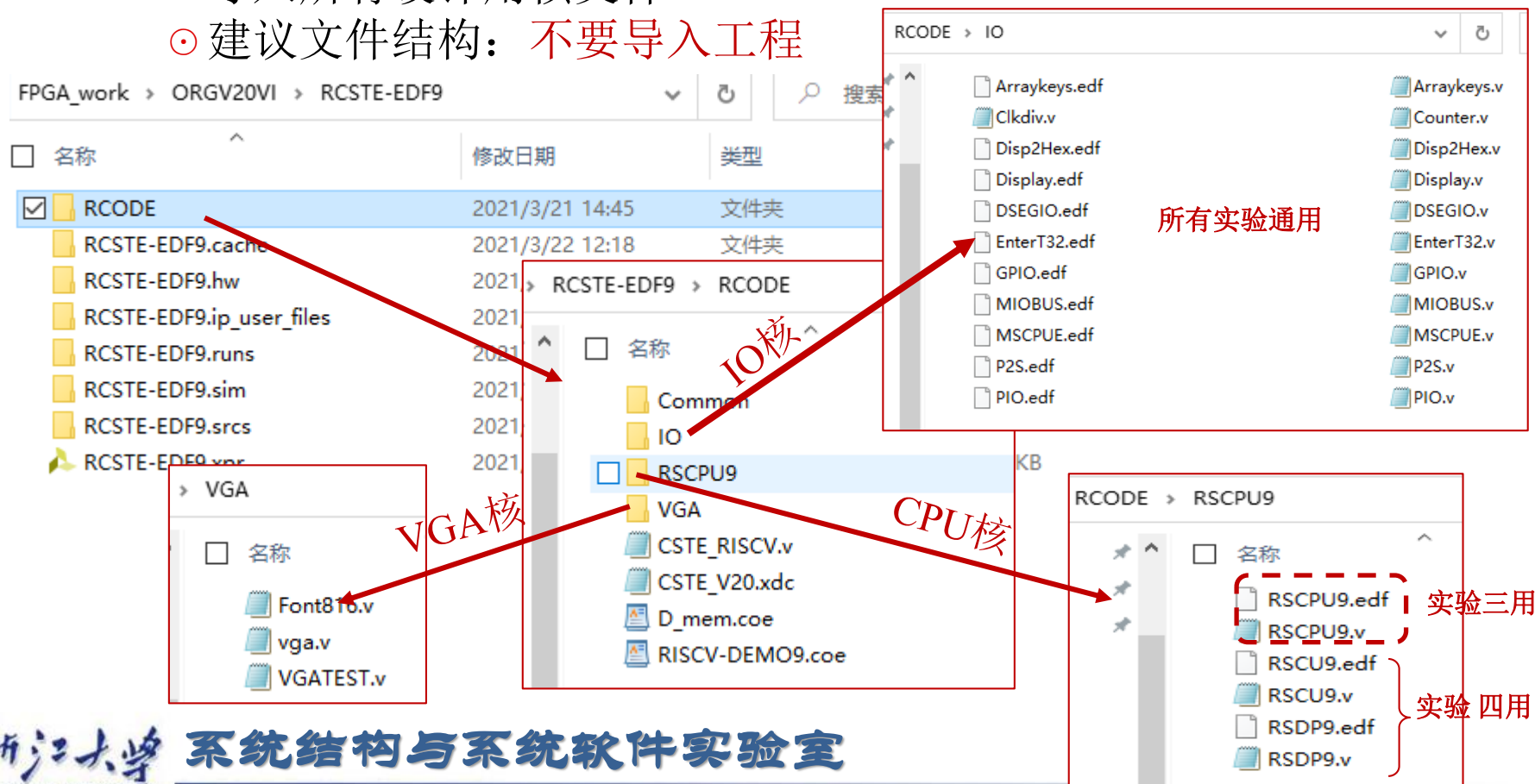
- ⊙ 对应搭建BLOCK DESIGN描述测试环境
- ⊙ 实验二基础优化

设计要点：HDL描述

◎ 用第三方EDF网表核和EXp01设计核

④ 在工程根目录建立核和代码文件夹

- 导入所有设计用核文件
- 建议文件结构：不要导入工程



The screenshot shows the directory structure of an FPGA project. The main directory is `FPGA_work > ORGV20VI > RCSTE-EDF9`. It contains several subdirectories and files. Red arrows and boxes highlight specific components:

- RCODE** (Folder): The main code directory, containing various .edf and .v files. A red box labeled "所有实验通用" (Used by all experiments) points to this directory.
- RCSTE-EDF9.cache** (File): A cache file.
- RCSTE-EDF9.hw** (File): A hardware description file.
- RCSTE-EDF9.ip_user_files** (Folder): A folder for IP user files.
- RCSTE-EDF9.runs** (Folder): A folder for run files.
- RCSTE-EDF9.sim** (File): A simulation file.
- RCSTE-EDF9.srscs** (File): A source code file.
- RCSTE-EDF9.vpr** (File): A project file.
- VGA** (Folder): A folder for VGA-related files. A red box labeled "VGA核" (VGA core) points to this folder, which contains `Font816.v`, `vga.v`, and `VGATEST.v`.
- IO** (Folder): A folder for IO-related files. A red box labeled "IO核" (IO core) points to this folder, which contains `Arraykeys.edf`, `Clkdiv.v`, `Disp2Hex.edf`, `Display.edf`, `DSEGIO.edf`, `EnterT32.edf`, `GPIO.edf`, `MIOBUS.edf`, `MSPUE.edf`, `P2S.edf`, and `PIO.edf`.
- RSCPU9** (Folder): A folder for RSCPU9-related files. A red box labeled "CPU核" (CPU core) points to this folder, which contains `CSTE_RISCV.v`, `CSTE_V20.xdc`, `D_mem.coe`, and `RISCV-DEMO9.coe`.
- RSCPU9** (Folder): A folder for RSCPU9-related files. A red box labeled "实验三用" (Used by Experiment 3) points to this folder, which contains `RSCPU9.edf`, `RSCPU9.v`, `RSCU9.edf`, `RSCU9.v`, `RSDP9.edf`, and `RSDP9.v`. A red box labeled "实验四用" (Used by Experiment 4) points to the `RSDP9.edf` and `RSDP9.v` files.

设计要点：BD描述



每次建立新工程后需将核加载到系统！

- 从学在浙大下载IPCORE核压缩包
- 解压到自己计算机的FPGA工作目录下：
 - 非工程目录，可与工程目录并级
 - 根据实验需要逐步提供教学核(有更新)
 - 若用自己设计的核，则核命名：同名核+加学号后四位

工作目录

MYIP ← 自己设计的核目录
ORGIP
ZPORT

自定义接口模板

课程提供的核目录

VGA_work > ORGV20VI > IPCORE > ORGIP

ArrayKeys
Counter
Disp2Hex
DIVOUT
EnterT32
MIOBUS
P2S
RSCPUE
TESTOUT

CLKDIV
DEBUG
Display
DSEGIO
GPIO
MSCPUE
PIO
SWOUT
VGA

课程核(有更新):
每个核1个目录

- 复制相关文档到工程根目录

- 存储器初始文件
- 引脚约束文件复制



核导入：参考实验二PPT

PROJECT MANAGER

- Settings
- Add Sources
- Language Templates
- IP Catalog

IP INTEGRATOR

- Create Block Design

Settings

Project Settings

- General
- Simulation
- Elaboration
- Synthesis
- Implementation
- Bitstream
- IP
 - Repository
 - Packager

Tool Settings

- Shortcuts
- > Strategies
- > Window/Behavior
- Help

IP > Repository

Add directories to the list of repositories. You may then add additional IP to a selected repository. If an IP is disabled then a tool-tip will alert you to the reason.

IP Repositories

+ - ↑ ↓

d:/FPGA_work/ORG20V/ORGIP (Project)

Refresh All

OK Cancel Apply Restore...

导入整个核文件夹！

自己设计的核也要导入
每次核目录更改后要刷新



设计要点：存储模块：ROM

◎ 设计32位指令存储器：

☞ SWORD实验平台 ROM用Distributed Memory

▣ ROM初始化文件(RISCV-DEMO9.coe)

▣ 这是一段功能测试程序：CPU仿真另行设计

```
memory_initialization_radix=16;  
memory_initialization_vector=
```

注意：这是RISC-V代码与OExp02不同

```
0200006F,00000033,00000033,00000033,00000033,00000033,00000033,00000033,00C02283,00502333,  
006303B3,00638E33,00738733,01CE02B3,005282B3,01C28EB3,01DE8F33,01EF0F33,01CF0433,01EF0F33,  
01EF0F33,01DF0FB3,01EF0F33,01EF0F33,01EF0F33,01EF0F33,01EF0F33,01EF0F33,01EF0F33,01EF0F33,  
01EF0F33,01EF0F33,01EF0F33,01EF0F33,01EF0F33,01EF0F33,01EF0F33,01EF0F33,01EF0F33,01EF0F33,  
01EF0F33,01EF0F33,01EF0F33,01EF0F33,01EF0F33,01EF04B3,01E4E633,00948933,012902B3,005282B3,  
406006B3,00C4A223,0004A583,00B585B3,00B585B3,00B4A023,006A8AB3,01592023,01402B03,0004A583,  
00B585B3,00B585B3,00B4A023,0004A583,0055FC33,006B0B33,040B0E63,0004A583,00E70BB3,017B8CB3,  
019B8BB3,0175FC33,000C0C63,037C0463,00E70BB3,037C0663,01592023,FB9FF06F,00D78463,0080006F,  
00D687B3,00F92023,FA5FF06F,0609AA83,01592023,F99FF06F,0209AA83,01592023,F8DFF06F,01402B03,  
00F787B3,0067E7B3,00E989B3,0089F9B3,006A8AB3,00DA8463,00C0006F,00E00AB3,006A8AB3,0004A583,  
00B58C33,018C0C33,0184A023,00C4A223,F6DFF06F;
```





设计要点存储模块：RAM

◎ 设计32位数据存储器：

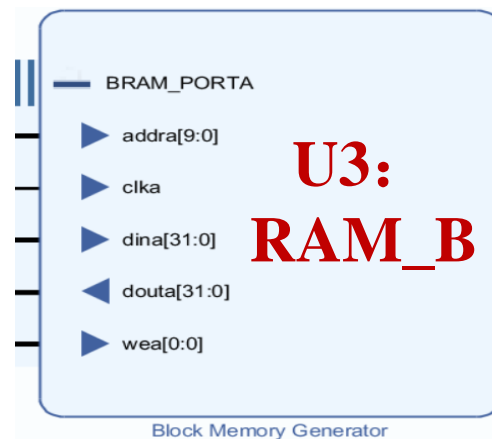
☞ SWORD实验平台 RAM用Block Memory

☞ RAM初始化数据：D_mem.coe

注意：清除BLOCK MEMORY所有输入输出寄存器

```
memory_initialization_radix=16;  
memory_initialization_vector=  
00000000, 11111111, 22222222, 33333333, 44444444, 55555555,  
66666666, 77777777, 88888888, 99999999, aaaaaaaa, bbbbbbbb,  
cccccccc, dddddddd, eeeeeeee, ffffffff, 557EF7E0, D7BDFBD9,  
D7BDFBD9, DFCFFCFB, DFCFBFFF, F7F3DFFF, FFFFDF3D, FFFF9DB9,  
FFFFBCFB, DFCFFCFB, DFCFBFFF, D7DB9FFF, D7BDFBD9, D7BDFBD9,  
FFFF07E0, 007E0FFF, 03bdf020, 03def820, 08002300;
```

RAM初始化数据。红色数据为七段LED图形



☞ 设计流程参考马德老师Lab00PPT



设计要点：硬件描述输入

◎ 阅读核接口要求

☞ 参考实验二PPT(本实验核接口有调整)

◎ 根据接口属性和测试环境要求输入描述

☞ 参考实验二连接示意和PDF文档输入描述

☞ 理解每一个语句或连线的意义和目的

◎ 实验三的BLOCK DESIGN描述有调整

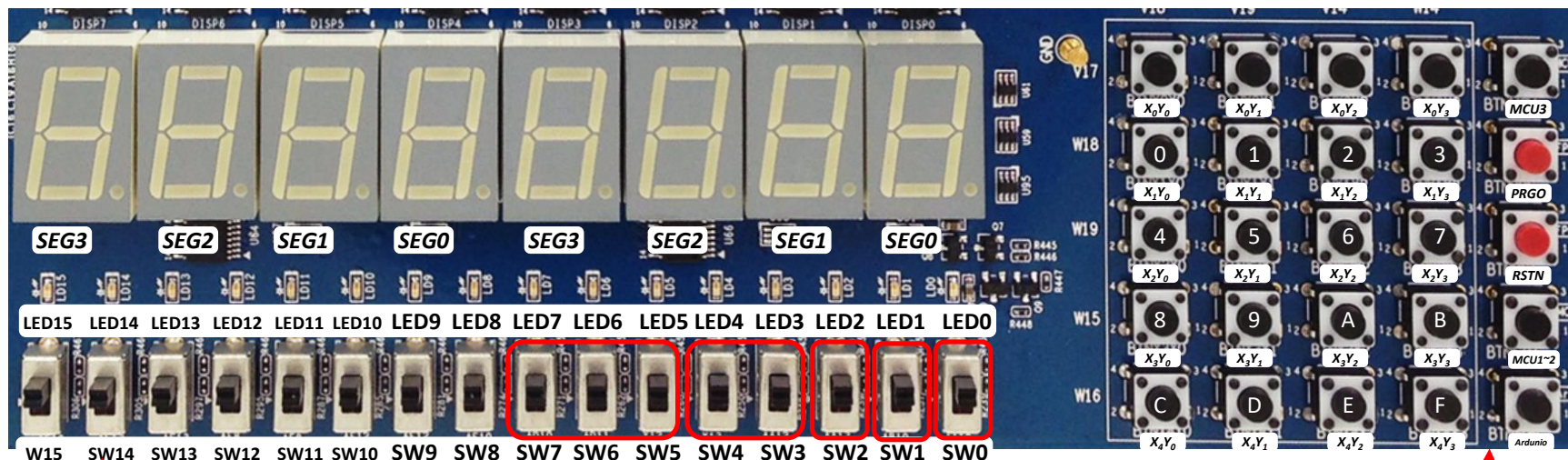
☞ 目前BD描述结构经测试正确

○ Vivadoa工具BD描述或存在BUG或属性还不了解，存在描述综合不稳定性。根据下载验证排查问题。

☞ HDL描述非常稳定。

设计要点：物理验证参见实验二

物理验证同实验二



SW[0]=文本图形选择

SW[1]=高低16位选择

SW[2]=CPU单步时钟选择

没有使用

SW[7:5]=显示通道选择

SW[7:5]=000: CPU程序运行输出

SW[7:5]=001: 测试PC字地址

SW[7:5]=010: 测试指令字

SW[7:5]=011: 测试计数器

SW[7:5]=100: 测试RAM地址

SW[7:5]=101: 测试CPU数据输出

SW[7:5]=110: 测试CPU数据输入

SW[4:3]=00, 点阵显示程序: 跑马灯

SW[4:3]=00, 点阵显示程序: 矩形变幻

SW[4:3]=01, 内存数据显示程序: 0~F

SW[4:3]=10, 当前寄存器+1显示

SW[13]=0选择测试ROM
SW[13]=1选择测试RAM
SW[14]=0/1测试数据翻页



● END