桂林电子科技大学2017-2018学年 第2学期

**计算机组成原理 实验报告**

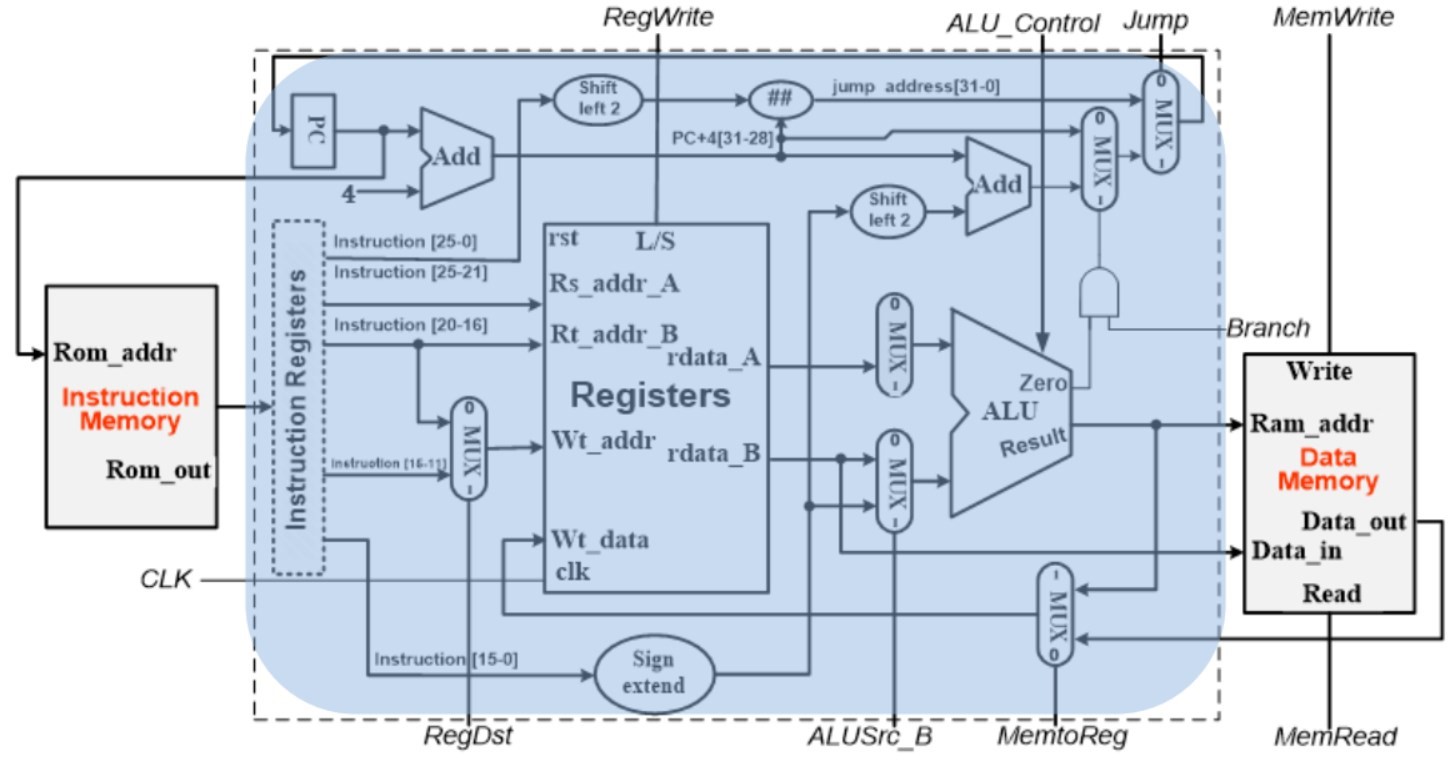
|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 实验名称 | **实验三、CPU 设计之数据通路** | | | | | | | |  | 辅导教师意见：  成绩 教师签名： |
| 院 系 | **计算机与信息安全学院** | | | 专业 | | **计算机科学与技术** | | |
| 学 号 |  | | | 姓名 | |  | | |
| 同 作 者 |  | | | | | | | |
| 实验日期 | **2020** | 年 | **7** | | 月 | | **4** | 日 |
|  |  | | | | | | | |

# 1.实验准备

### 1.1实验目的

1. 掌握 CPU 的核心：数据通路组成与原理
2. 设计数据通路
3. 学习测试方案的设计
4. 学习测试程序的设计

### 1.2实验原理

 图 1-1 数据通路

CPU 模块的核心组成是数据通路模块与控制器模块。实验一中，我们使用的 CPU 模块是 SCPU 模块（工程文件是 SCPU.ngc 和 SCPU.v）。实验二中，我们使用单独的数

据通路模块 Datapath（工程文件是 SDpath.ngc 和 Datapath.v）与控制器模块

Controller（SCtrl.ngc 和 Controller.v）替换了实验一中的 SCPU.ngc 和 SCPU.v。本质上

SCPU 模块就是由 Data\_path 模块和 Controller 模块组成的。

Data\_path 模块也可以细分为 Path 模块，ALU 模块和 Register 模块等子部件。本实验中，将设计细分的子部件来替换 Datapath 模块。

### 1.3实验任务

1. 设计 9+条指令的数据通路
   1. 用逻辑原理图设计实现数据通路
   2. 替换实验二的数据通路核
   3. 此实验在实验二的基础上完成
2. 设计数据通路测试方案：
   1. 通路测试：R-格式指令测试、I-格式指令测试
3. 设计数据通路测试程序：
   1. R-格式指令测试 b) I-格式指令测试

# 2.实验操作与实现

### 2.1建立工作目录

在 D:/COD/labs/目录或其他目录下新建一个名为 lab3 的文件夹（注意路径中不能有中文和空格），将…/lab\_solution/lab2 文件夹下的所有文件拷贝到 lab3 文件夹下。

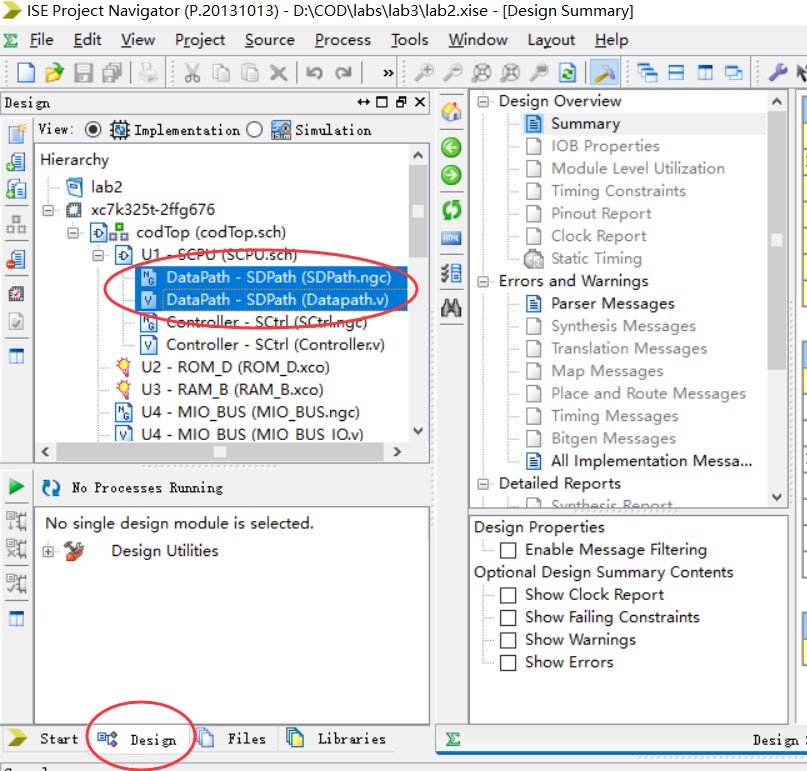
运行 ISE 14.7 软件。Win10 系统可以在左下角搜索框输入 ISE，然后在搜索结果中点击 ISE Design Suite 14.7，如下图红色椭圆框所标示。Win7 系统可以在开始菜单中去找到 ISE Design Suite 14.7 软件，然后点击运行。

### 2.2 建立工程

在 ISE 软件界面左上角，点击 File -> Open Project，在弹出的对话框中点击选中 lab3 文件夹下的 lab2 文件，在对话框右下角点击”打开”按钮。

### 2.3 打开 lab1 工程

打开 ISE 工程后界面如下图所示。在左侧的 Design 窗口，红色椭圆框标记的 Datapth 模块（SDpath.ngc 和 Datapah.v 两个文件）是接下来需要替换的数据通路 IP 核。



### 2.4 设计

点击选中 SDpath.ngc 文件，右键单击选中 Remove 来移除该文件。对 Datapath.v 文件进行同样的移除操作。

选中顶层文件，右键单击 Add Copy of Source…，然后选

择…/lab\_source/lab3/CPU 目录下的 alu.sch，Regs.v，REG32.v 以及 SDpath.sch 这 4 个文件，点击 OK。

继续右键单击 Add Copy of Source…，然后选择…/lab\_source/lab3/common 目录下的

文件添加完成后，实验一中的 Datapath 核已经被替换成了细分子部分，包括 ALU 子部件，Register 子部件以及多路选择器 MUX 子部件等。

将…/lab\_source/lab3/sym 目录下的所有文件拷贝到 lab3 根目录下。双击

SDpath.sch 文件，可以在界面右侧打开该文件的原理图设计窗口。可以看到，Datapath 模块包含 ALU，Regs，PC，MUX 等多个子模块

点击原理图设计窗口右下方的“X”关闭 SDpath.sch 窗口，双击 alu.sch 文

件，可以在界面右侧打开该文件的原理图设计窗口。可以看到，ALU 模块包含 and32，or32，xor32，srl32 等多个子模块

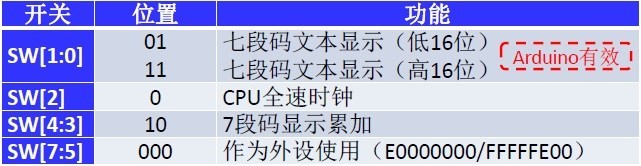
### 2.5 生成 bitstream 文件

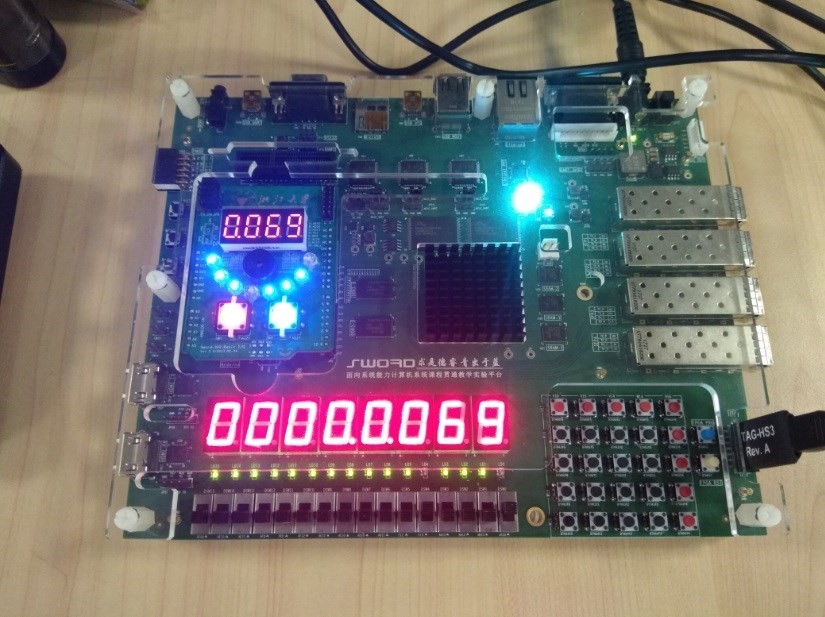
点击原理图设计窗口右下方的“X”关闭 alu.sch 窗口，然后单击选中顶层文件 codTop.sch，在左侧界面点击 Design 窗口中的 Generate Programming File 按钮，编译该工程。

编译成功结束如下图所示。此时，工程目录下生成了名为 codtop.bit（顶层文件名）的二进制文件。

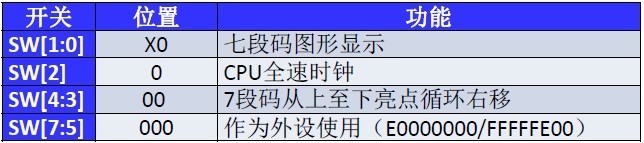
# 实验验证与结果

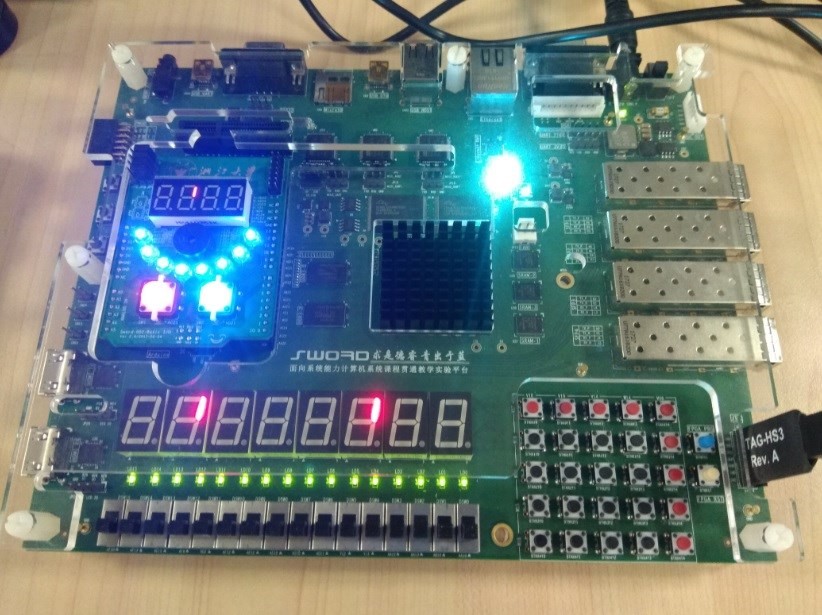
2.3.1 测试文本功能。按照下图配置 SWORD4.0 的开关，DSW0 和 DSW4 向上，其他开关向下。七段数码管上会显示不断增加的计数器数值，同时 sword-002 上会显示高 16 位或低 16 位（可以使用 DSW1 切换）。



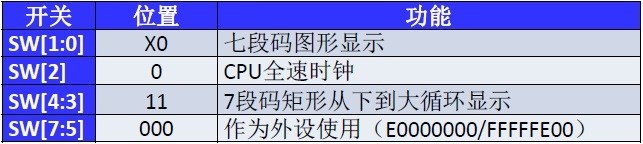


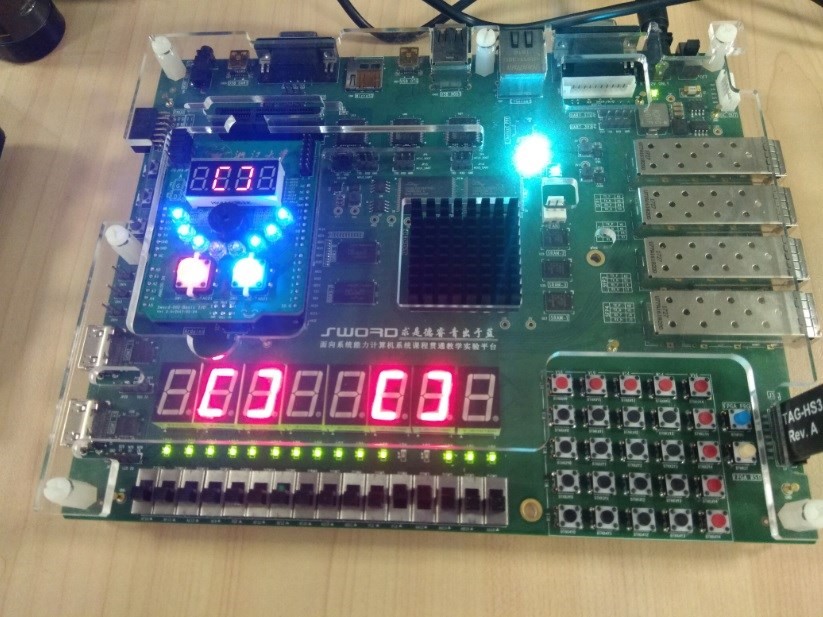
2.3.2 测试图形功能。按照下图配置 SWORD4.0，全部拨码开关向下，七段数码管会显示短横线或竖线。





2.3.3 测试图形功能。按照下图配置 SWORD4.0，DSW3、DSW4 向上，其他拨码开关向下，七段数码管会显示不断变化的矩形图形。

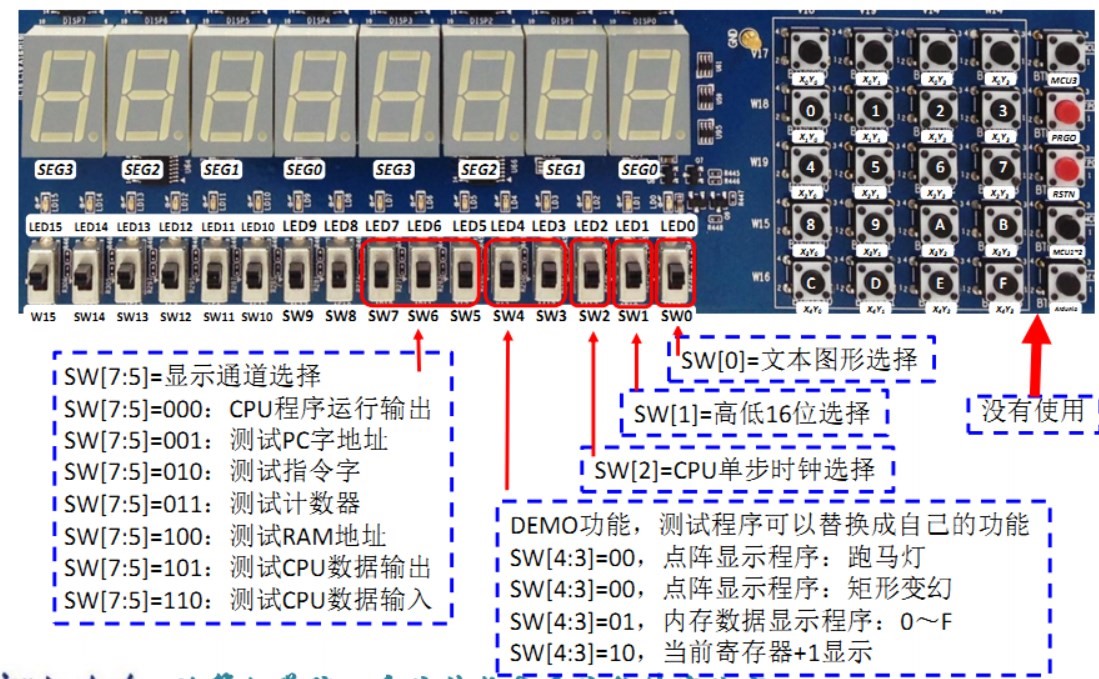




# 指令测试

MIPS 指令集包含 R-格式，I-格式和 J-格式这 3 种类型的指令。本实验中，J-格式指令作为指令循环运行的基础，不做单独的测试，主要是针对 R-格式，I-格式的指令进行相关测试

## 指令测试操作



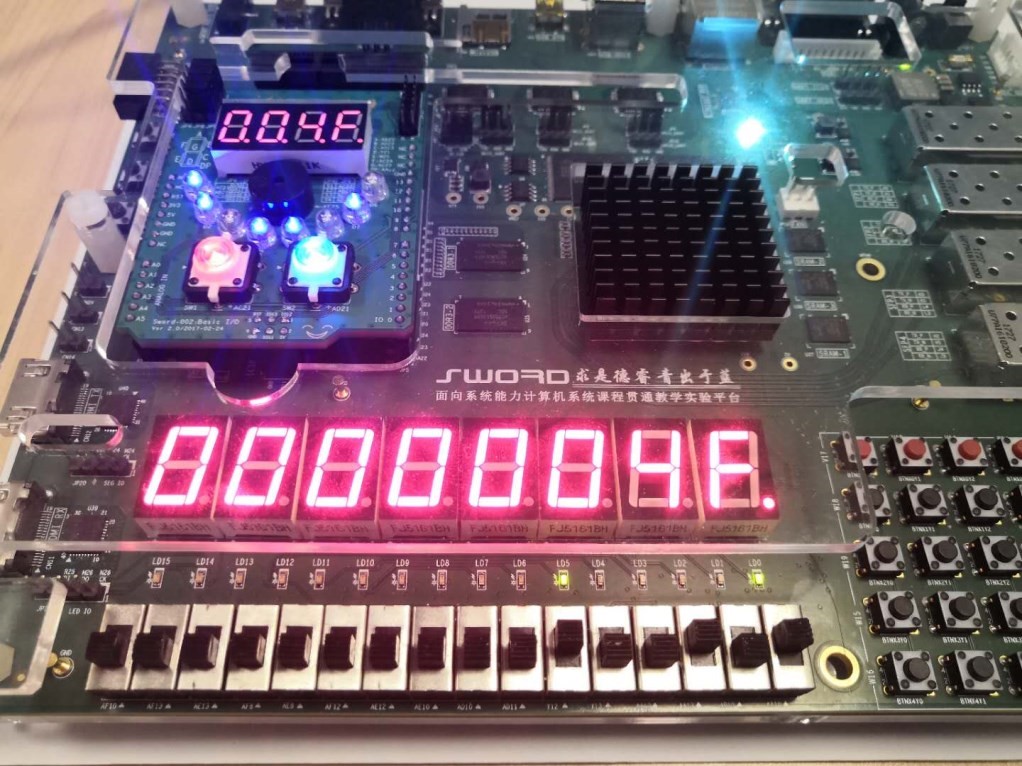
在 2.3 中，我们验证了基础的 DEMO 功能，仅使用了 SW[4:3]和 SW[0]这 3 个拨码开关。指令测试的操作需要用 CPU 单步时钟，然后根据 SW[7:5]选择不同的显示通道。

DEMO 程序是指令 ROM 中默认使用的 coe 文件，工程文件夹 Code 目录下名为 SCPU\_DEMO9\_sign.coe。CPU 执行的程序就是指令 ROM 中存放的指令字，也就是 coe 文件中的指令字数据。

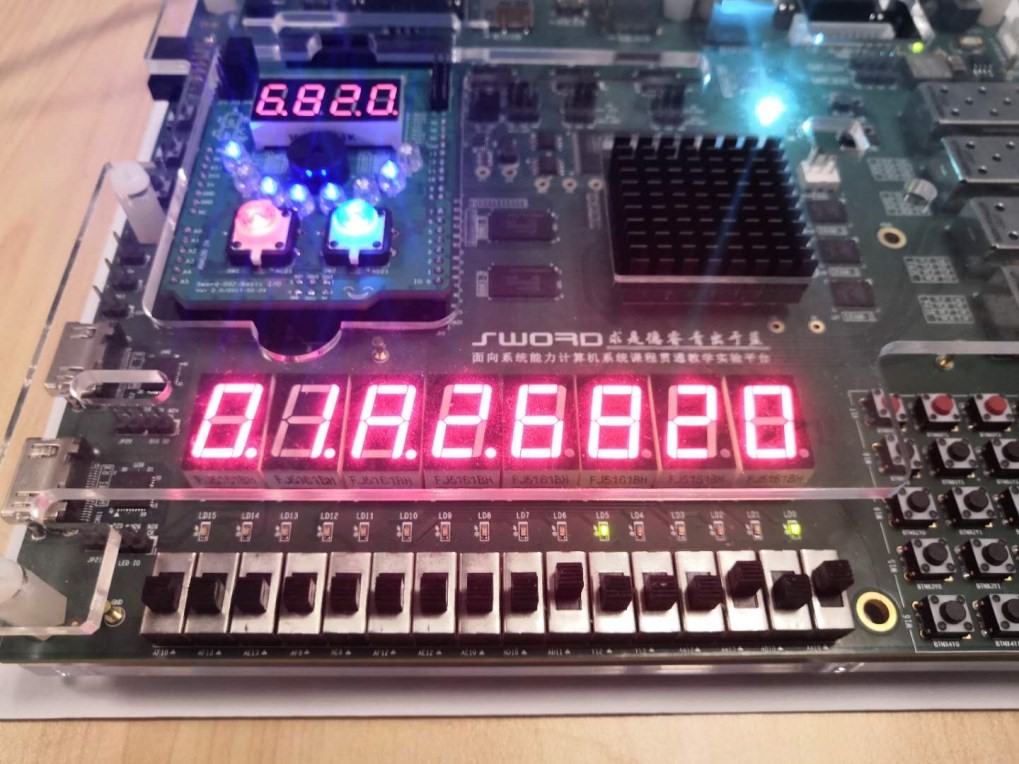
3.1.1 首先，将拨码开关 SW[2]、SW[0] 和 SW[5]状态设置为 1，其余拨码开关状态设置为 0，根据上图的说明，目前 CPU 时钟切换成了单步时钟（约 3 秒执行

1 条指令），然后 SW[7:5]状态为 001，于是 7 段数码管上显示的是 PC 字地址（对应指令 ROM 的地址），指令顺序执行时，PC 字地址每次加 1，遇到分支或跳转指令，则地址会突然变化。注意到每次 PC 字地址到 56 后，会跳转到

37。



3.1.2 将 SW[7:5]设置为 010，7 段数码管上显示的是 CPU 正在执行的指令字（对应指令 ROM 中存储的指令数据）。注意，08000037 指令是 J-格式跳转指令，跳转到 37 的 PC 字地址。



## R-格式指令测试

设计 ALU 和 Regs 测试程序替换 DEMO 程序 a) ALU、Regs 测试参考设计，测试结果通过 CPU 输出信号单步观察

b)

SW[7:5]=100,

ALU

输出

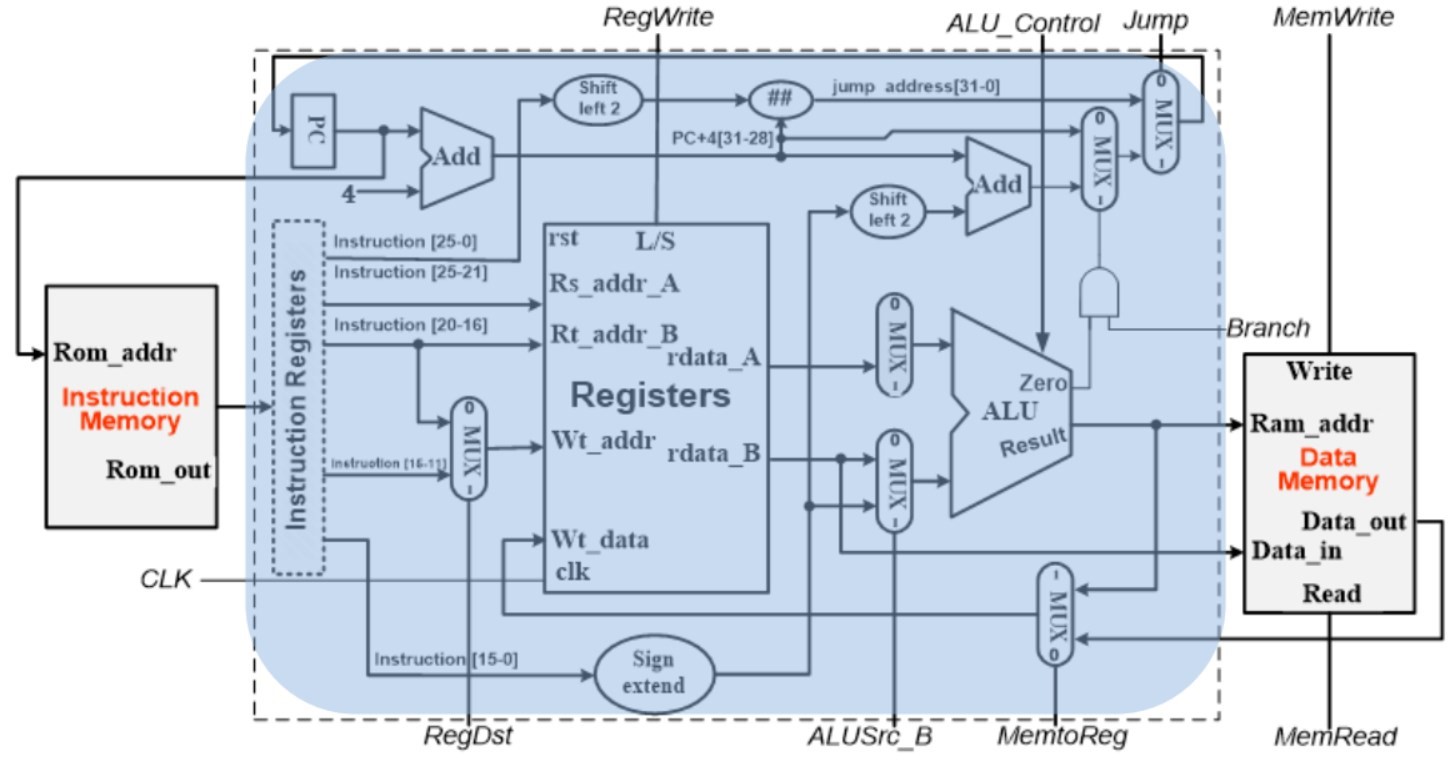
c)

SW[7:5]=101

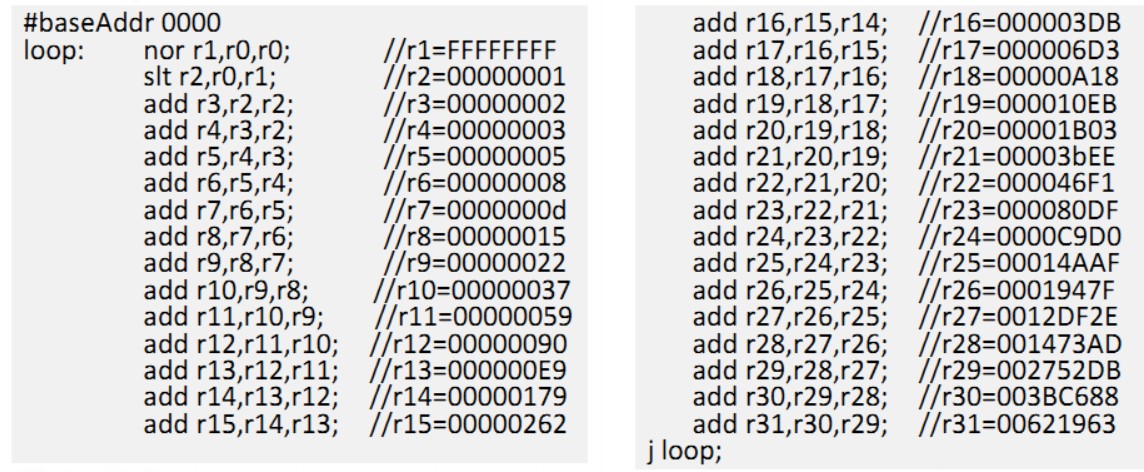
，寄存器

B

输出



基本思路是使用一些 R-格式指令，然后通过观察 ALU 输出（设置 SW[7:5]为 100），可以参考下面的汇编代码来进行测试。



下面左侧是指令汇编代码，右侧是对应的指令机器码。对应了上图左侧的 15 条指令，最后的 Jump 指令则是让 CPU 循环执行这 15 条 R-格式指令。

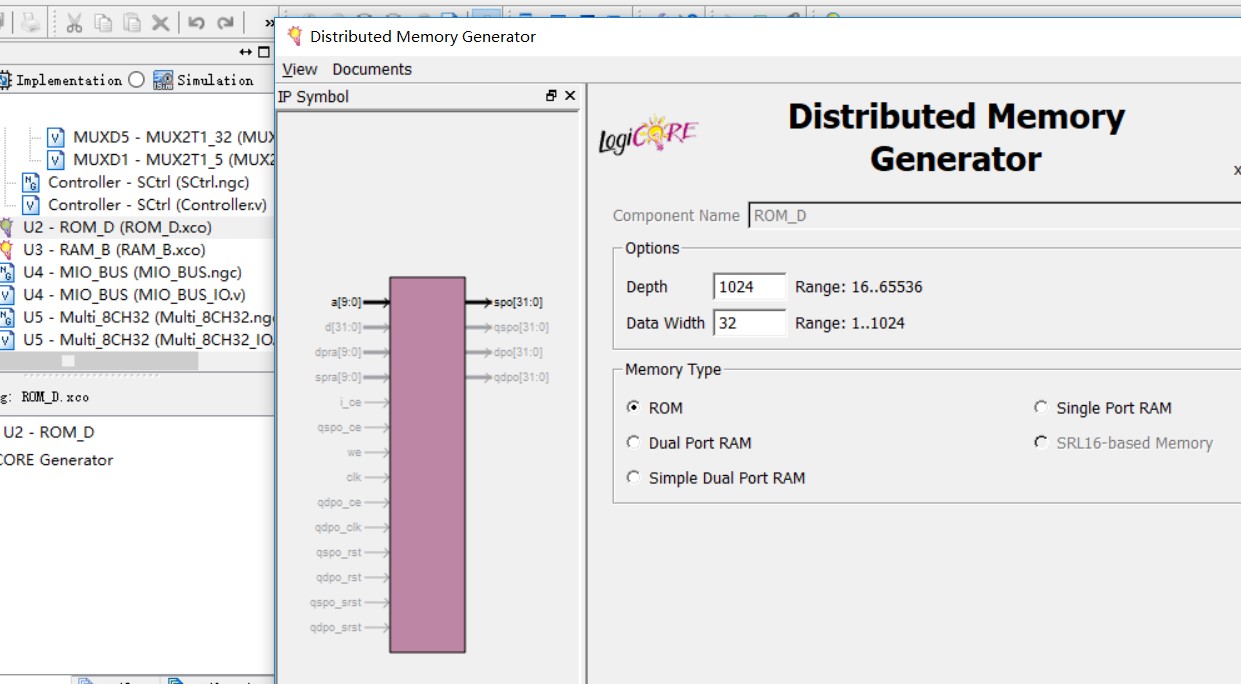
nor r1,r0,r0; 00000827 slt r2,r0,r1; 0020102a add r3,r2,r2; 00421820 add r4,r3,r2; 00622020 add r5,r4,r3; 00832820 add r6,r5,r4; 00a43020 add r7,r6,r5; 00c53820 add r8,r7,r6; 00e64020 add r9,r8,r7; 01074820 add r10,r9,r8; 01285020 add r11,r10,r9; 01495820 add r12,r11,r10; 016a6020 add r13,r12,r11; 018b6820 add r14,r13,r12; 01ac7020 add r15,r14,r13; 01cd7820

jump 0000; 08000000

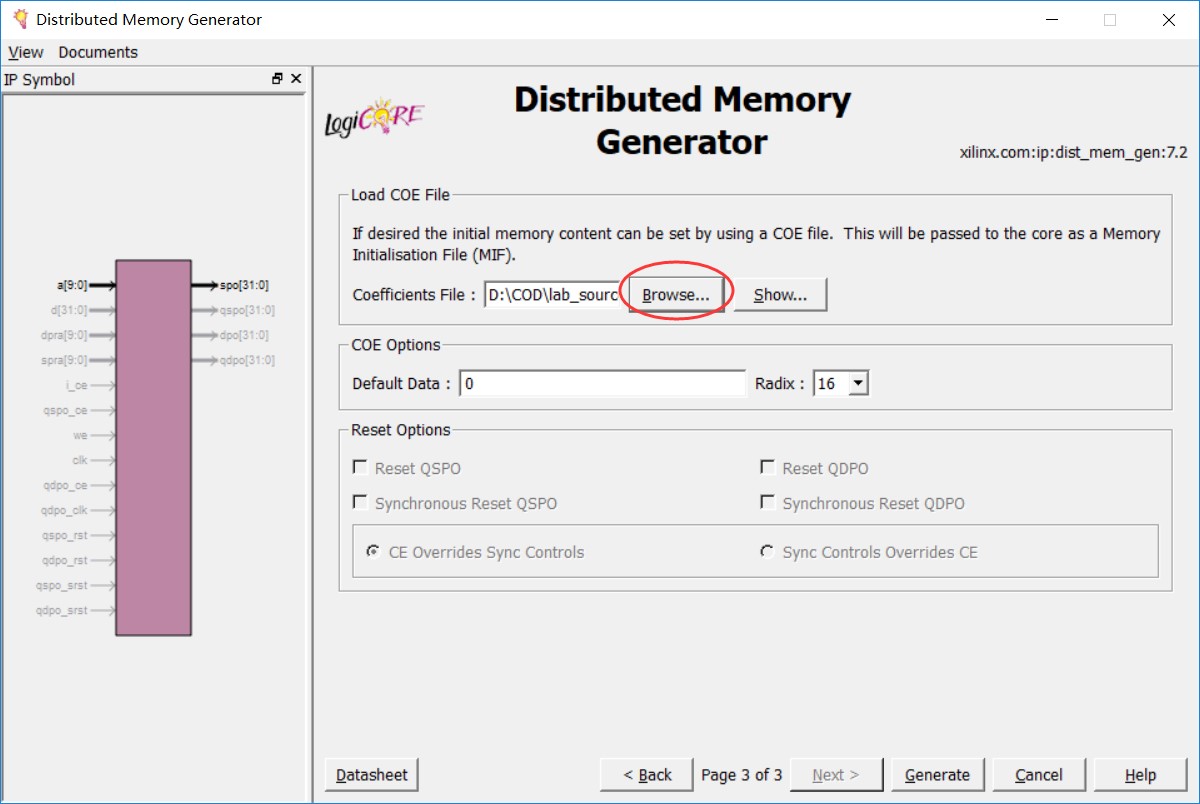
上面的指令已对应的 coe 文件，存放在…/lab\_source/lab3/coe 目录下，名为 rtype.coe。

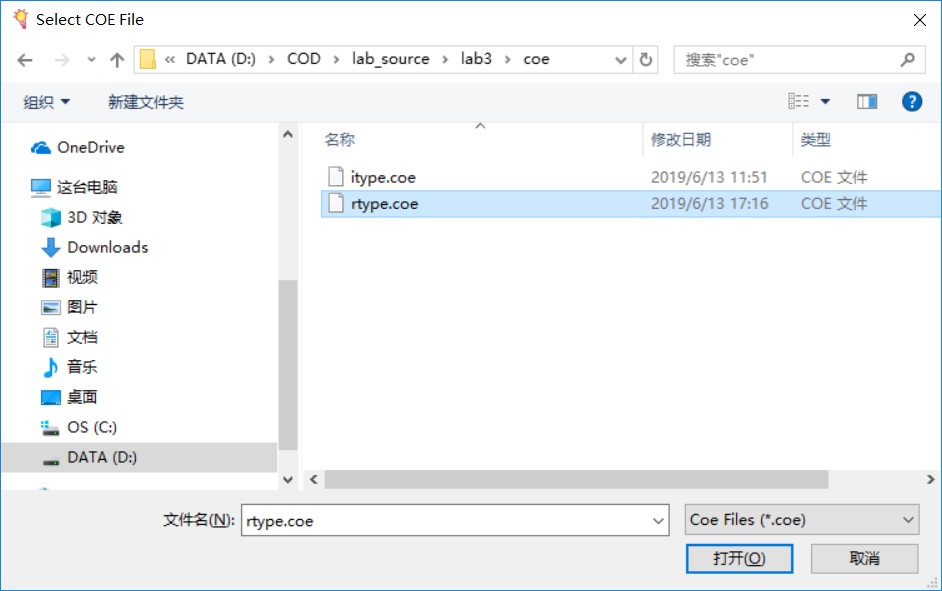
3.2.1 在 lab3 工程界面，双击 ROM\_D 模块，会弹出 IP 配置界面，点击两次

Next。

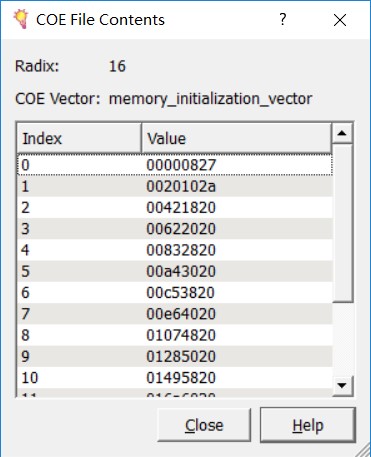


3.2.2 点击 **Browse** 按钮，然后选择…/lab\_source/lab3/coe 目录下的 rtype.coe 文件，点击**打开**。

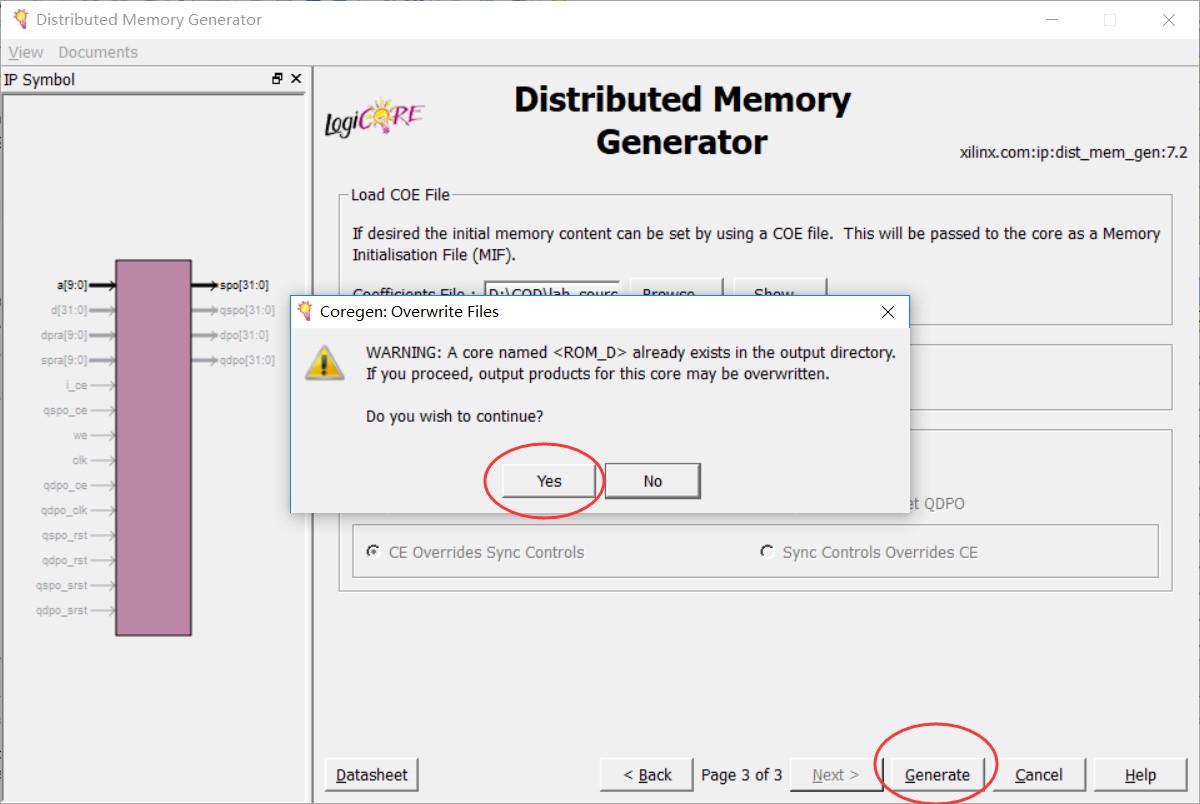




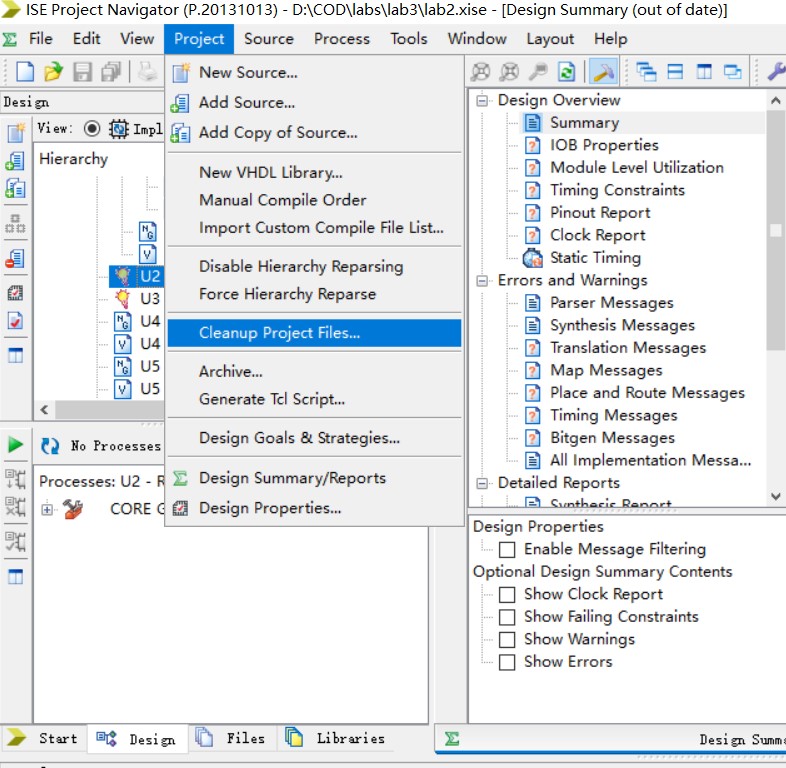
3.2.3 点击 **Show** 按钮，可以查看 coe 文件内容。Index 可以当成是 ROM 的数据字地址（对应 PC 字地址），Value 则是初始存储的数据字（对应指令字）。



3.2.4 点击 **Close，**然后点击 **Generate** 按钮，然后会由对话框弹出，点击 **Yes**。

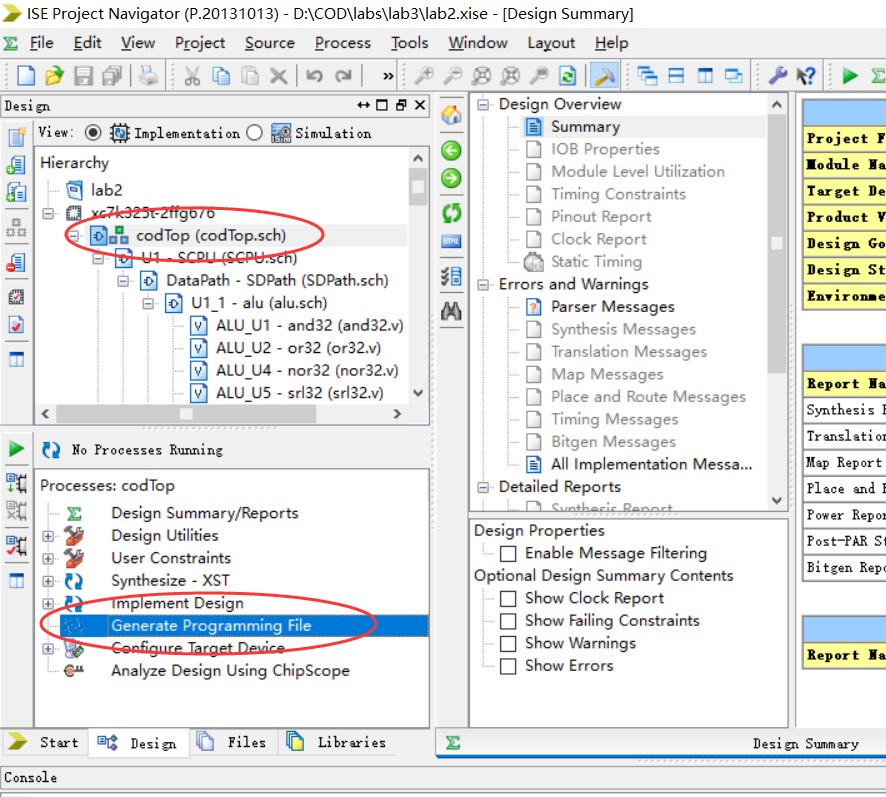


3.2.5 等 ROM\_D IP 重新 Generate 结束。点击 **Project->Cleanup Project Files…，**然后在随后弹出的对话框中点击 **OK**。



3.2.6 在 Design 窗口点击选中顶层文件 codTop.sch，然后点击 **Generate**

**Programming File**，编译工程，重新生成 bitstream 文件后，按照 **2.2** 的步骤下载 bitstream。



3.2.7 Bitstream 下载成功后。将拨码开关 SW[2]、SW[0] 状态设置为 1，SW[7:5]状态为 100，其余拨码开关状态设置为 0。此状态下，CPU 时钟切换成了单步时钟（约 3 秒执行 1 条指令），7 段数码管上显示的是 ALU 的输出结果。观察 7 段数码管的显示结果，与图 3-5 的结果进行印证。



## I-格式指令测试

基本思路是使用 lw 指令，然后通过观察寄存器 B 输出（设置 SW[7:5]为 101）来进行测试，可以参考下面的汇编代码来进行测试。注意，指令中访问存储器 RAM\_B 的地址是按字节（8bit）计算，需要在 RAM\_B 的数据字（32bit）地址基础上乘以 4 进行计算。

lw r5, 0($zero); 8C050000 lw r5, 4($zero); 8C050004 lw r5, 8($zero); 8C050008 lw r5, 12($zero); 8C05000C lw r5, 16($zero); 8C050010 lw r5, 20($zero); 8C050014 lw r5, 24($zero); 8C050018 lw r5, 28($zero); 8C05001C lw r5, 32($zero); 8C050020 lw r5, 36($zero); 8C050024 lw r5, 40($zero); 8C050028 lw r5, 44($zero); 8C05002C lw r5, 48($zero); 8C050030 lw r5, 52($zero); 8C050034 jump 0000; 08000000

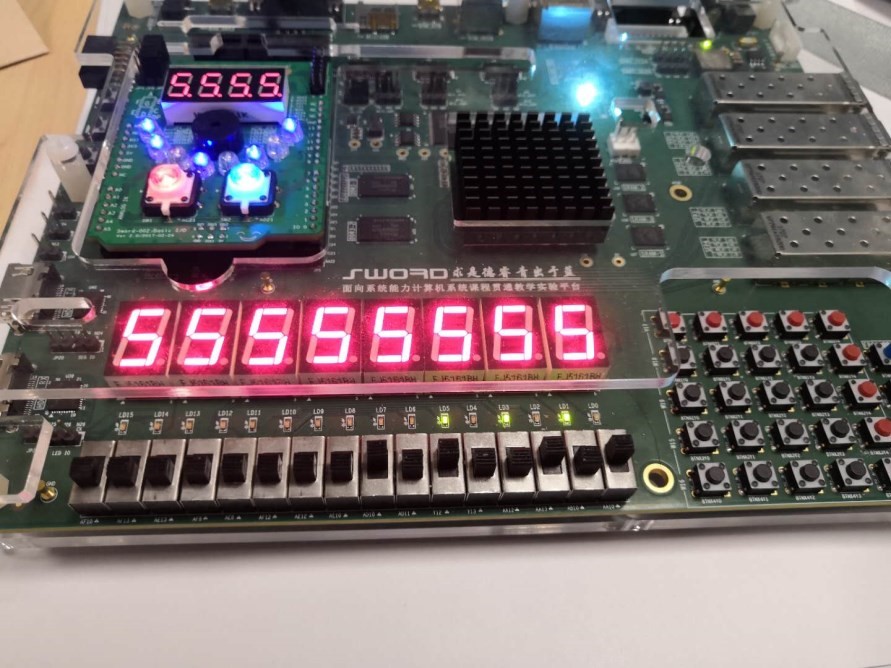
上面的指令已对应的coe文件，存放在…/lab\_source/lab3/coe目录下，名为itype.coe。

3.3.1 在 lab3 工程界面，双击 ROM\_D 模块，使用…/lab\_source/lab3/coe 目录下的 itype.coe 文件。重复步骤 3.2.1 到 3.2.6。

3.3.2 重新生成 bitstream 文件后，按照 2.2 的步骤下载 bitstream。

3.3.3 bitstream 下载成功后。将拨码开关 SW[2]、SW[0] 状态设置为 1，SW[7:5]状态为 101，其余拨码开关状态设置为 0。此状态下，CPU 时钟切换成了单步时钟（约 3 秒执行 1 条指令），7 段数码管上显示的是寄存器 B 的输出。观

察 7 段数码管的显示结果，与存储器模块 RAM\_B 初始化数据（…/lab3/Code/ 目录下的 D\_mem.coe 文件）进行印证。



# 4.实验小结

本次实验学习了与数据通路有关的cpu设计，实验中收获了对于数据通路组成与原理的理解，在老师的引导下对数据通路进行设计，并设计测试方案对实验结果进行验证。

实验中从建立工程文件到增加ip到生成Bitstream文件。其中最让我们小组感到头疼的是对于增加ip的实验过程。实验在一步步的进行中不断的询问老师和指导学长，在老师和学长的指导下对cpu通路有进一步了解。Cpu数据通路就像期间的道路，是cpu中不可或缺的一环。实验可以分为三大部分。在第一部分我们依照指导书进行操作很快得出结果。第二部分，通过两人的讨论，不断对实验细节和实验原理，加以刨析，对实验目的加一分析，得出cpu数据通路的具体设计体现，依据具体设计展开实验操作。在第三部分进行设计验证，对设计方案设计验证方案。

在这次实验中我们知道了数据通路的设计思想，和具体的设计实现路径，通过实验了解到如何设计建立数据通路，同时对cpu有了新的认识，更加深入了解cpu的内部构造和具体设计，及工作原理。同时对于指令的测试，让我们加深指令的具体运行方式和具体与运行路线。本次实验我深入了解到数据如何在cpu各个系统间进行流转。通过具体实验操作加深对理论的理解。

本子实验在老师和学长的指导帮助下，掌握了数据通路的设计方法和对工程操作的具体操作。同时利用设计的验证方案进行结果测试。根据结果进行反馈差错修改等操作。通过实验过程加深对理论的理解。