计算机系统结构课程实验

总结报告

题目：静态流水线设计与性能分析

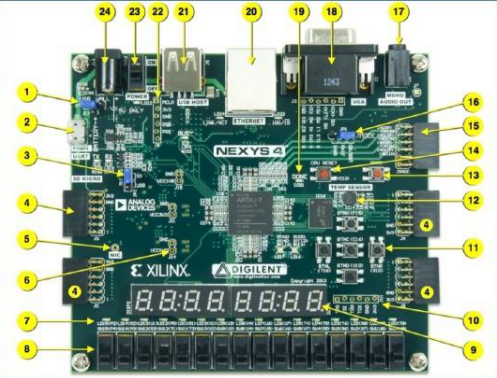
1652270 冯舜

老师：陆有军

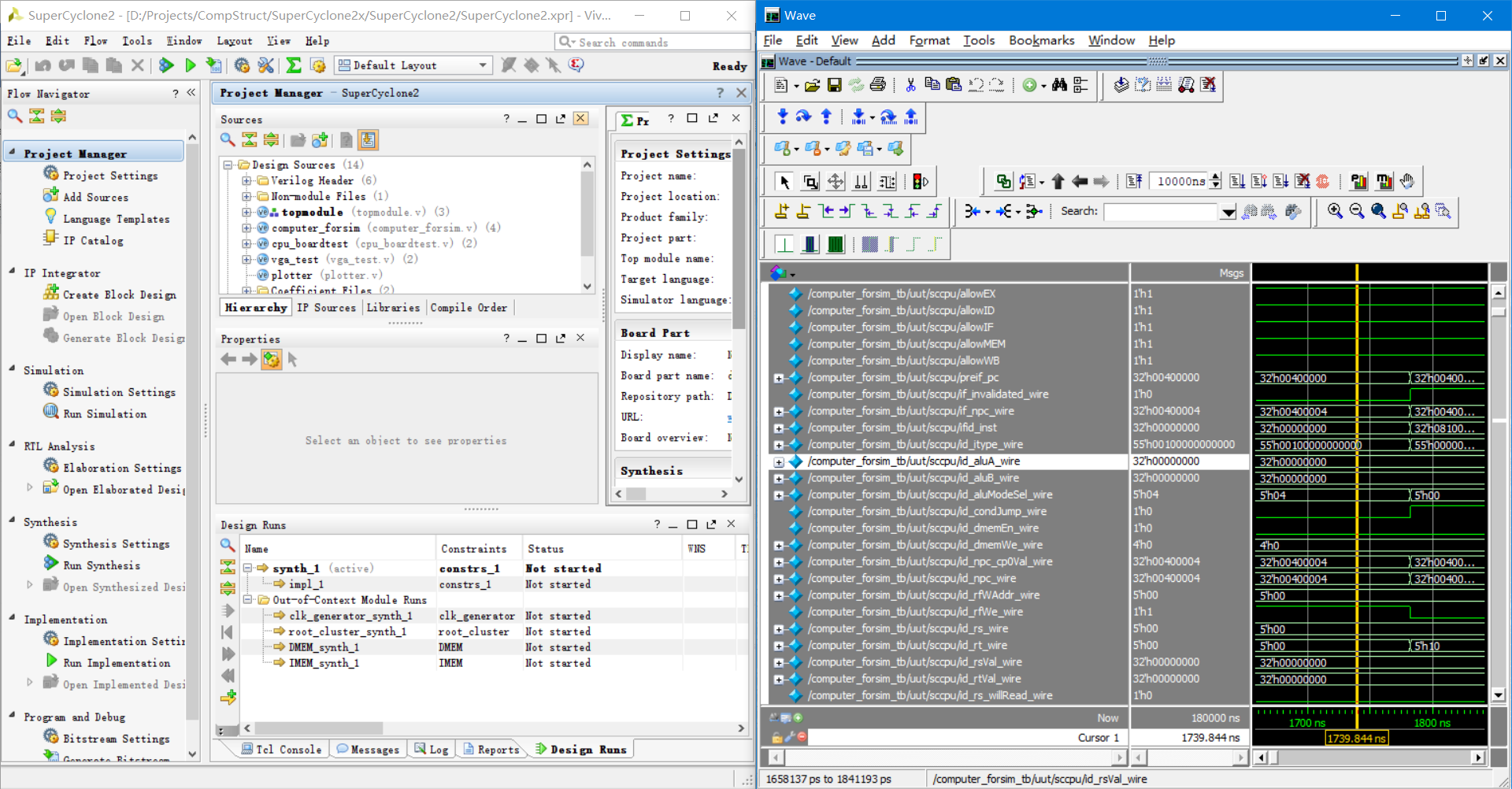
2018/11/18

## 实验环境部署与硬件配置

实验所用硬件为Windows PC以及Xilinx Nexys 4 DDR开发板。



软件包括开发板配套的Vivado 2016（ISE集成开发环境，综合、实现、比特流生成软件）以及业界领先仿真软件ModelSim。



实验环境部署的过程为：打开Vivado，新建一个空项目，硬件选择为xc7a100tcsg324-1，不添加任何源代码，确认。

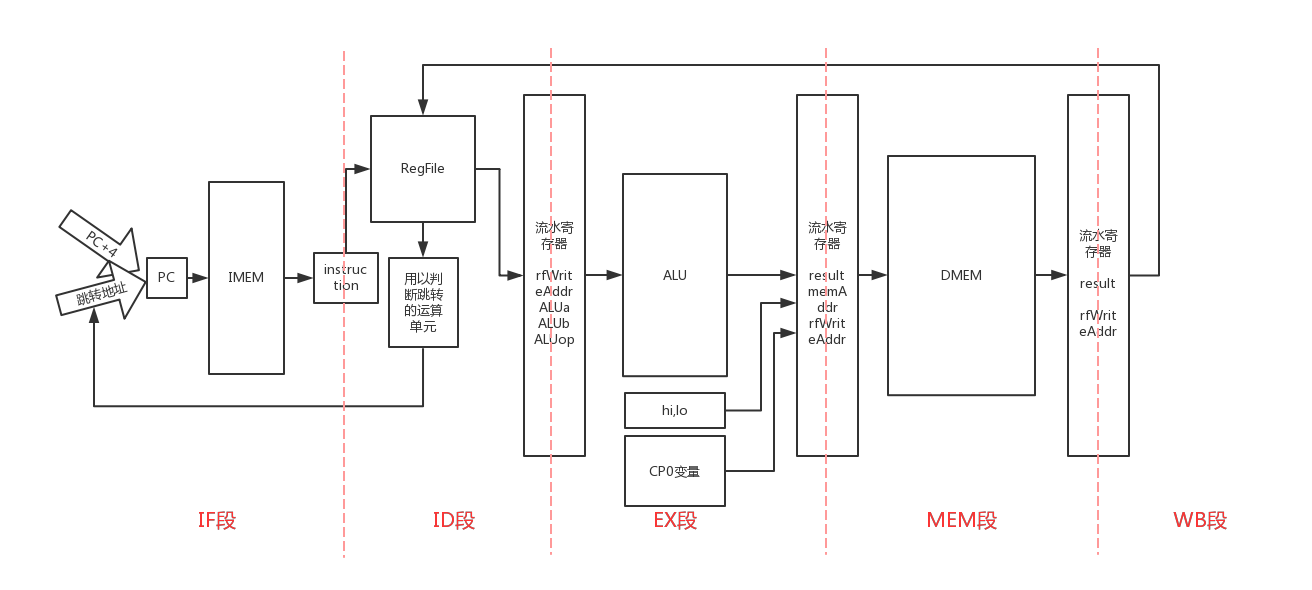
## 实验目的与目标

本次实验的目的是实现静态流水线模型，了解静态流水线的实现原理。

本实验的目标是通过编写Verilog HDL程序实现静态流水线CPU，运行一个编写好的汇编程序，验证给出的数学模型。

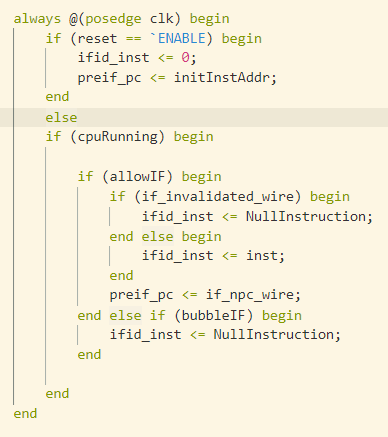
## 实验的总体结构和部件

### 静态流水线的总体结构



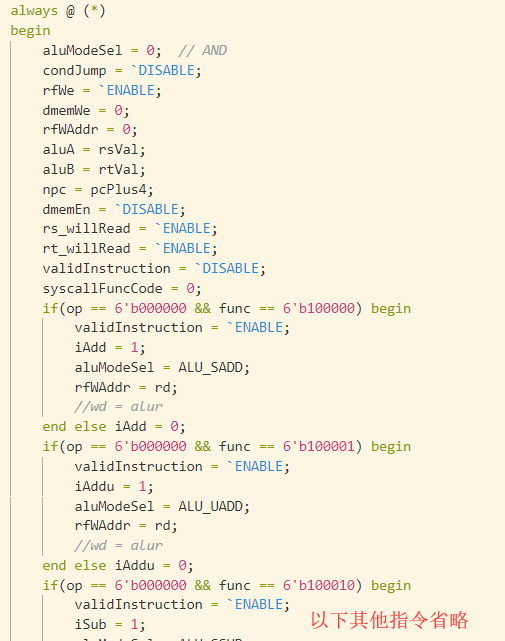
### 部件及其解释说明

#### IF（取指令）段逻辑

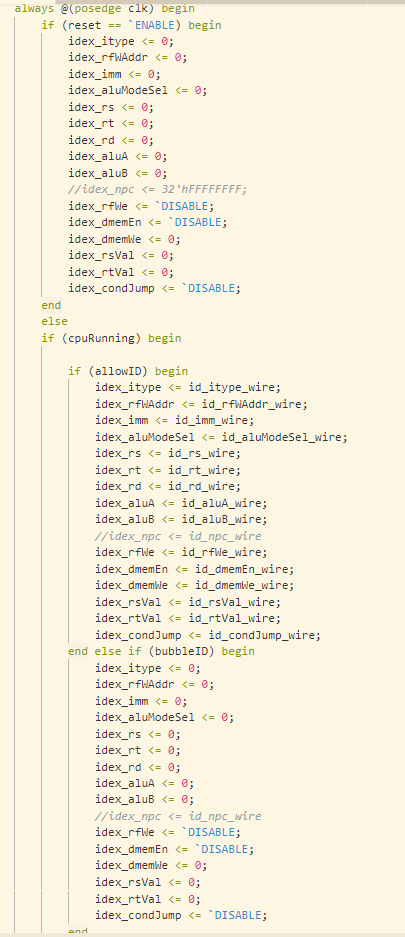


IF段的逻辑实现是一段时序逻辑，其主要功能包括更新PC，以及根据PC的值将指令内容取出，存入流水寄存器。

#### ID（指令译码）段逻辑



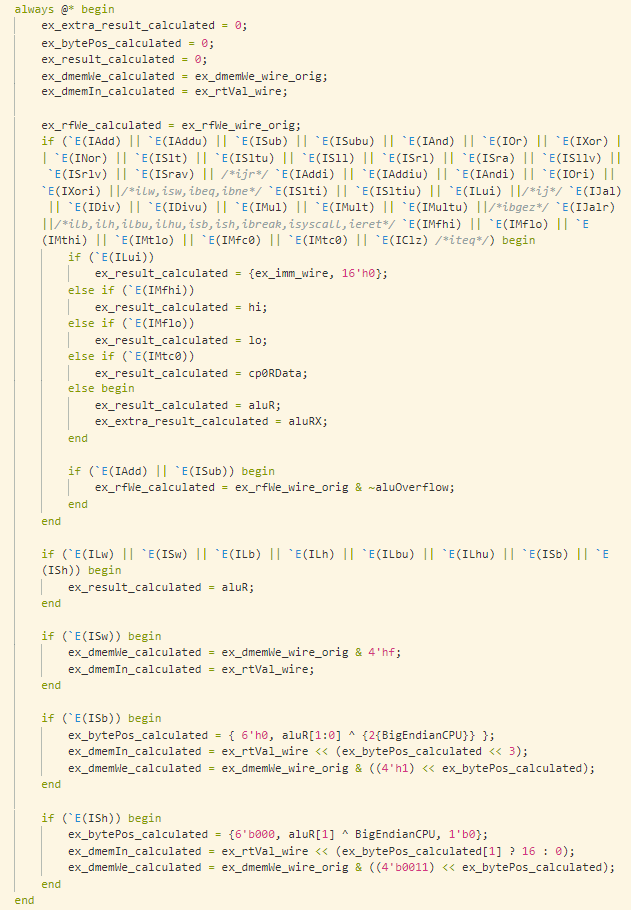
组合逻辑部分



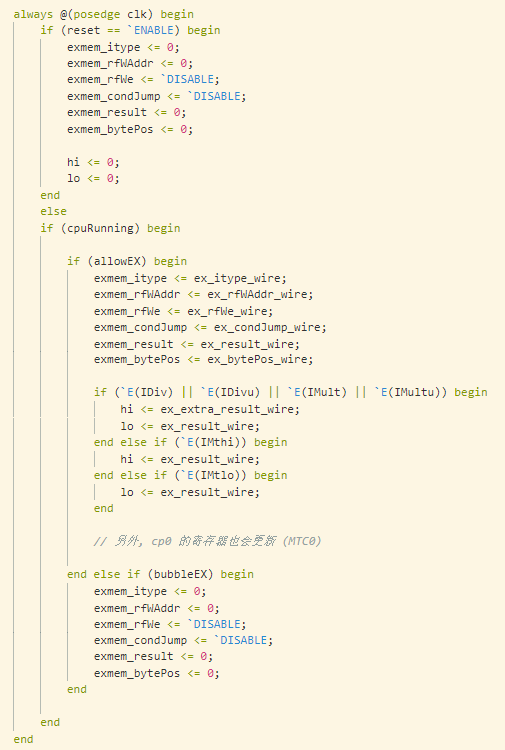
时序逻辑部分

ID段的实现方式是一段组合逻辑加一段时序逻辑。它的功能较丰富，包括对指令进行译码、取出指令涉及到需要读的寄存器值放置到流水寄存器、预先判断是否满足跳转条件以便控制IF段的PC等。

#### EX（执行）段逻辑



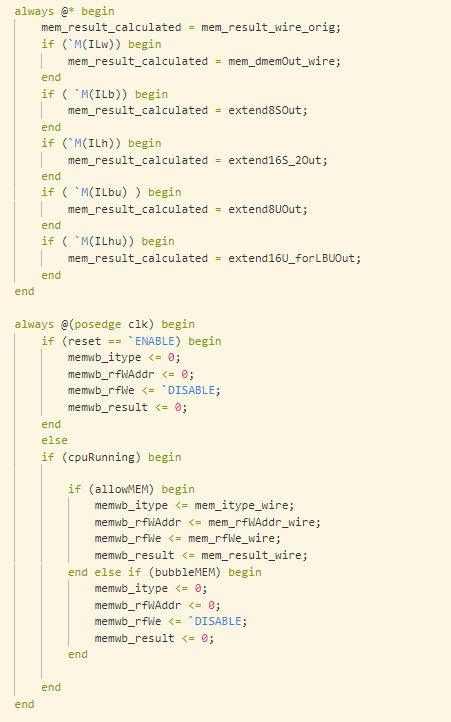
EX段组合逻辑



EX段时序逻辑

EX段的实现也是一段组合逻辑加一段时序逻辑。对于计算类指令，EX调用ALU获得结果；对于访存指令，EX调用ALU计算出内存地址。结果统一存放于流水寄存器中。此外，EX段还对HI、LO寄存器、CP0协处理器的内部寄存器进行取值和更新。

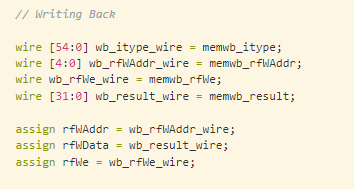
#### MEM（仿存）段逻辑



MEM段组合逻辑与时序逻辑

MEM段的实现也是组合逻辑加时序逻辑。主要功能是将DMEM数据存储器送出的数据存入流水寄存器中。

#### WB（写回）段逻辑



WB段逻辑

WB段逻辑非常简单，是将寄存器堆的写有关值赋予前一段流出的值，是组合逻辑。当然，时钟上升沿到来时，在寄存器堆内部会做写入更新。

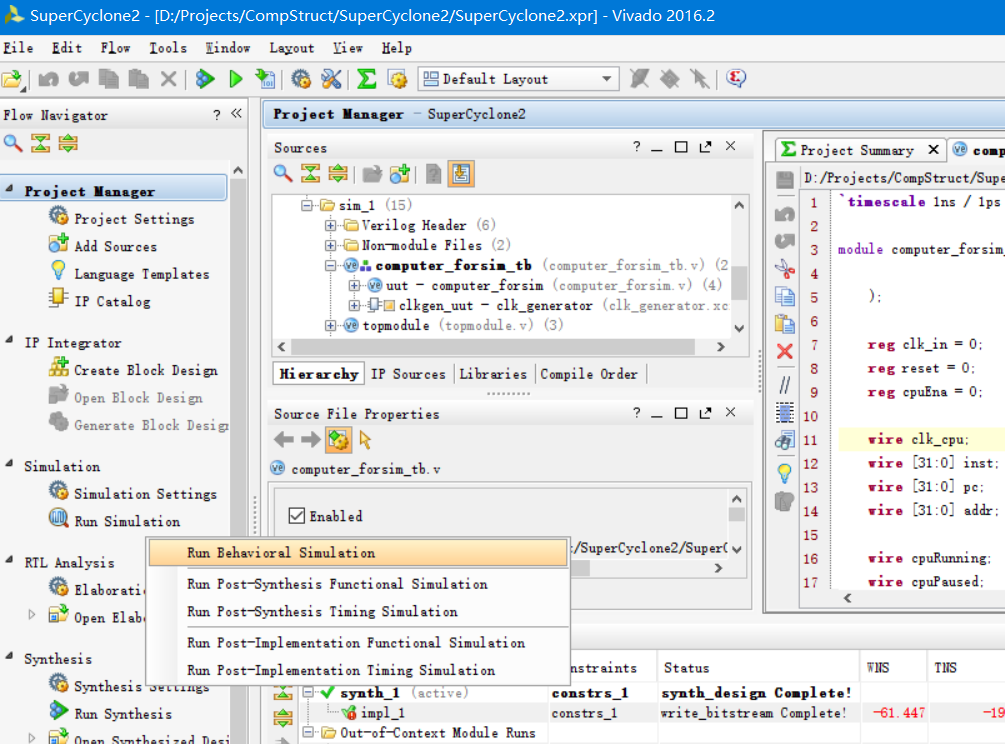
## 实验仿真过程

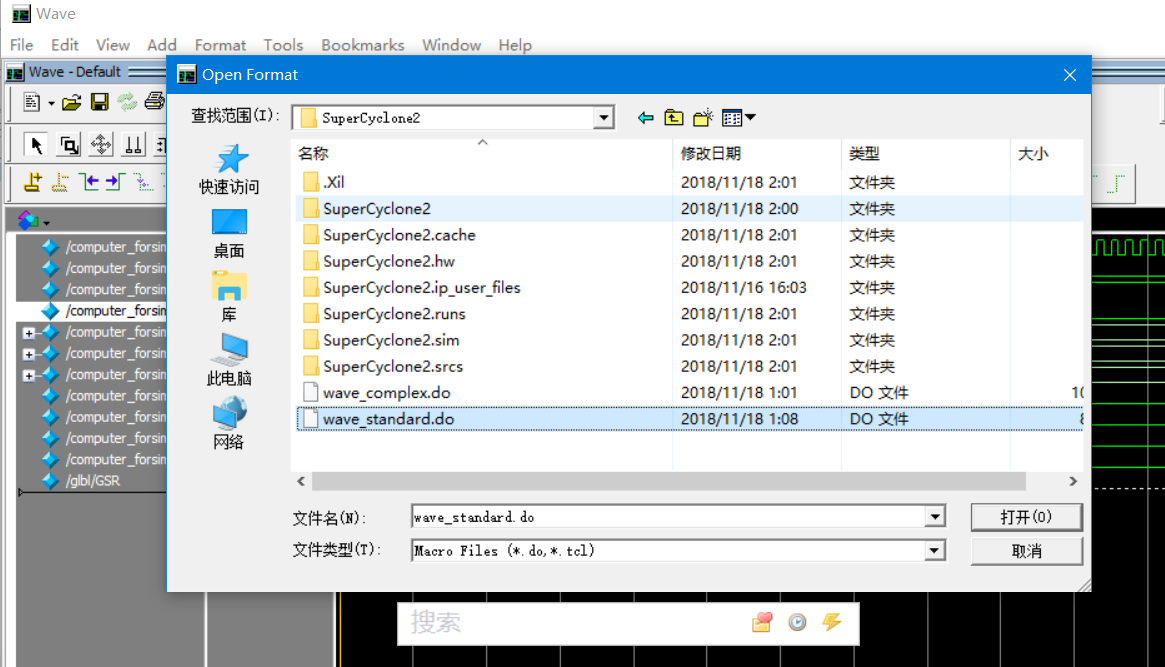
### 静态流水线仿真过程

写仿真顶层文件：

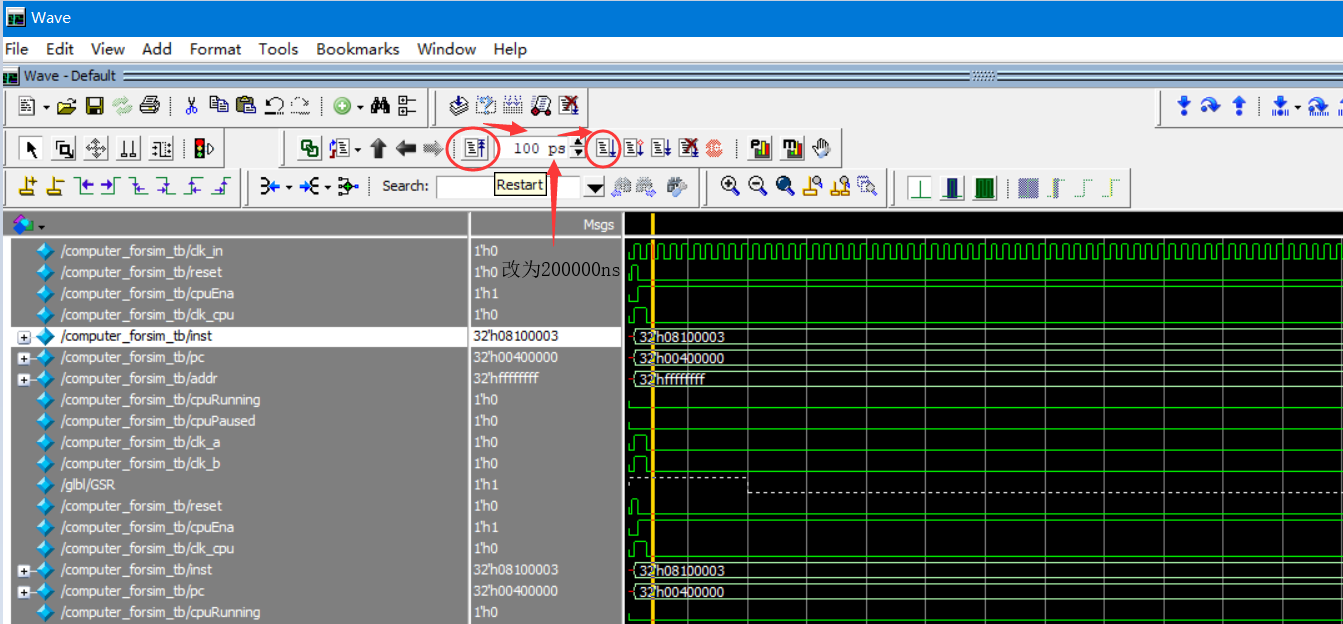
|  |
| --- |
| `timescale 1ns / 1ps  module computer\_forsim\_tb(  );    reg clk\_in = 0;  reg reset = 0;  reg cpuEna = 0;    wire clk\_cpu;  wire [31:0] inst;  wire [31:0] pc;  wire [31:0] addr;  wire cpuRunning;  wire cpuPaused;  always begin  #5 clk\_in = ~clk\_in;  end  initial begin  #3 reset = 1;  #5 reset = 0;  cpuEna = 1;  end  computer\_forsim uut(  clk\_in,  reset,  cpuEna,  clk\_cpu,  inst,  pc,  addr,  cpuRunning,  cpuPaused  );  wire clk\_a;  wire clk\_b;  clk\_generator clkgen\_uut(  .clk\_100MHz(clk\_in),  .clk\_vga(clk\_a),  .clk\_cpu(clk\_b)  );  endmodule |

打开Vivado工程，使用Vivado的“行为级仿真”功能，调用ModelSim进行仿真。

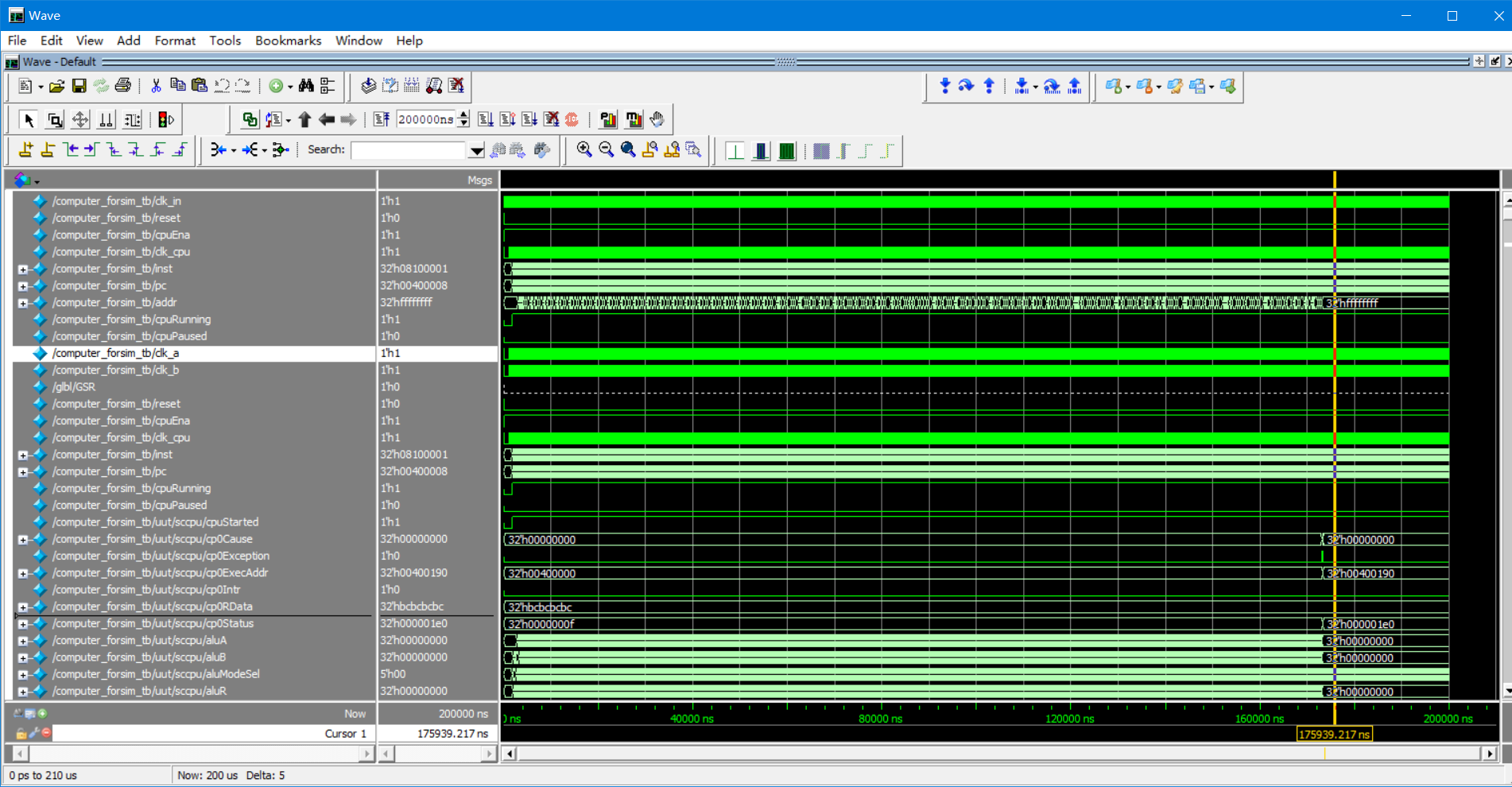




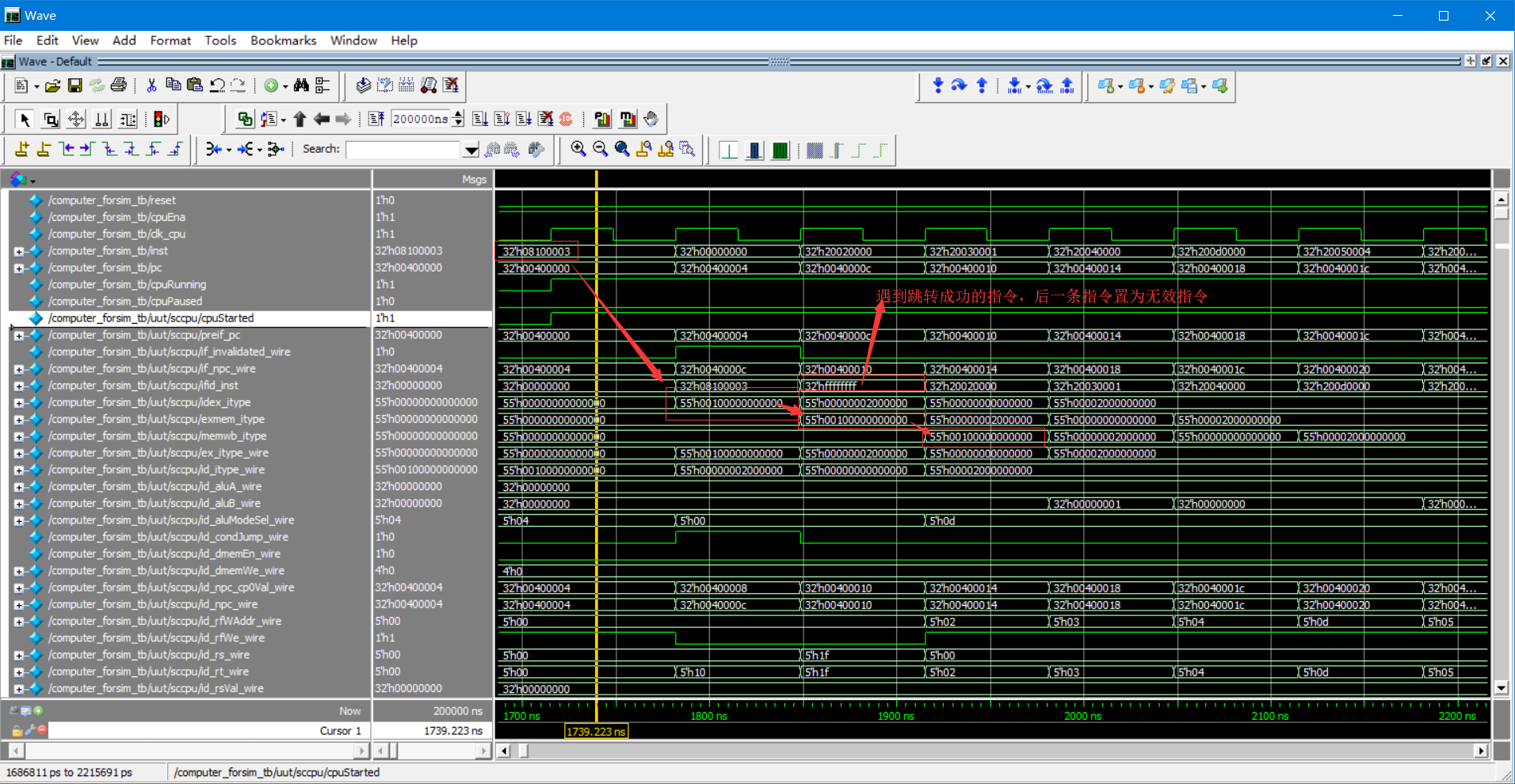
在波形表中，加载工程文件夹根目录下的wave\_standard.do。

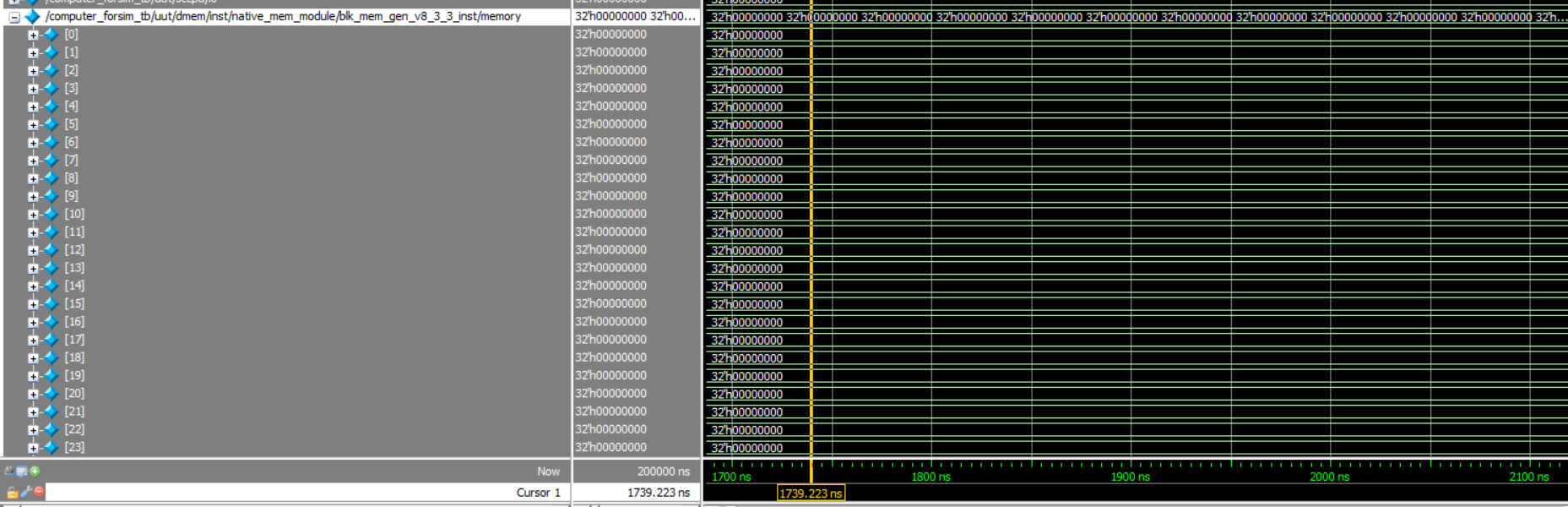


点击“Restart”，修改仿真时间为200000ns，点击“Run”，可以看到从程序开始运行到停止运行整个过程的仿真波形。



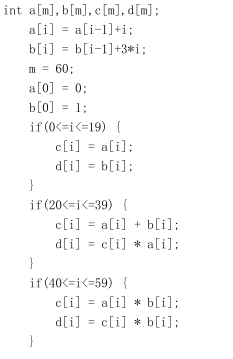
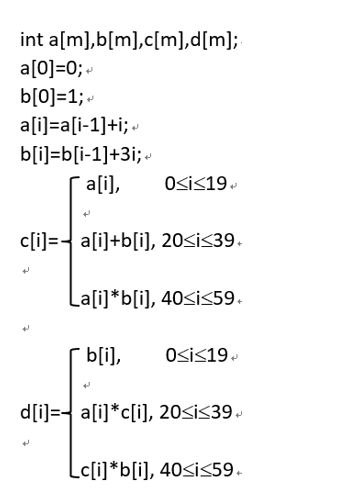
### 波形图和寄存器波形的意义



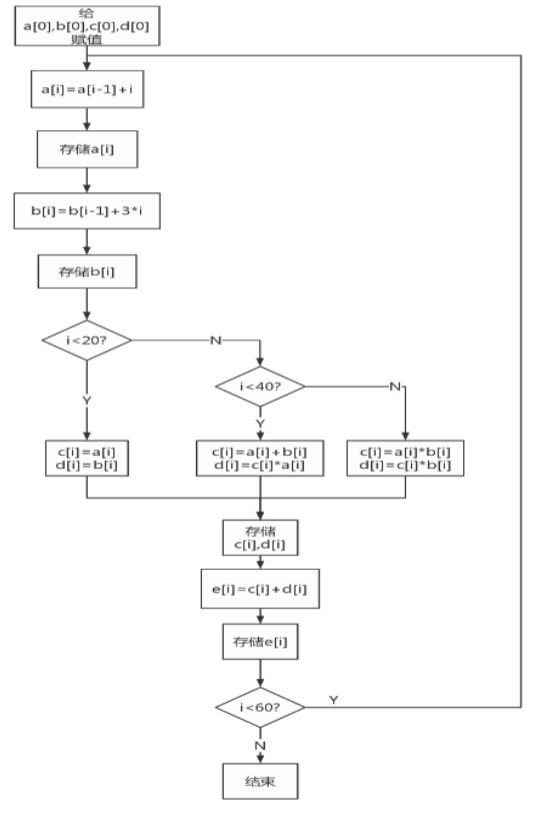


* clk\_cpu：CPU的时钟
* cpuRunning：CPU是否在运行
* pc：当前流入CPU的指令地址（PC寄存器）
* inst：当前流入CPU的指令字
* ifid\_inst：当前待译码的指令字
* \*\_itype：对应两个段之间的解析后的指令字（55比特长的独热码，每个比特对应一种指令）。ID之后每个段都有一个波形，总共四个波形，与ifid\_inst结合可以看到指令流过CPU的过程。
* id\_\*：ID段有关的信号内容
* idex\_\*：ID段和EX段之间的流水寄存器
* ex\_\*：EX段有关的信号内容
* exmem\_\*：EX段与MEM段之间的流水寄存器
* mem\_\*：MEM段有关的信号内容
* memwb\_\*：MEM段与WB段之间的流水寄存器
* wb\_\*：WB段有关的信号内容
* blk\_mem\_gen\_v8\_3\_3\_inst/memory：DMEM的内容

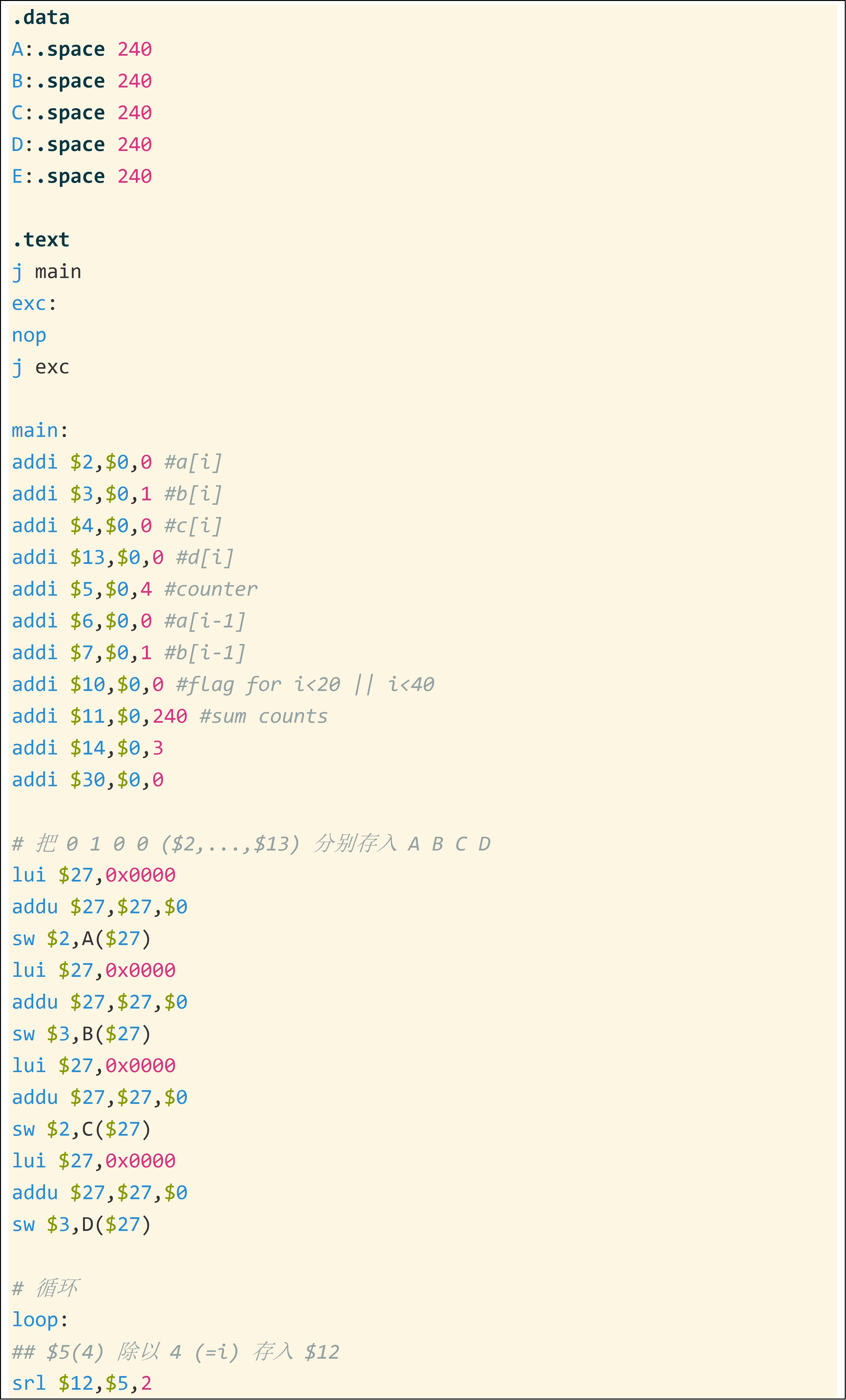
## 实验验算数学模型及算法程序

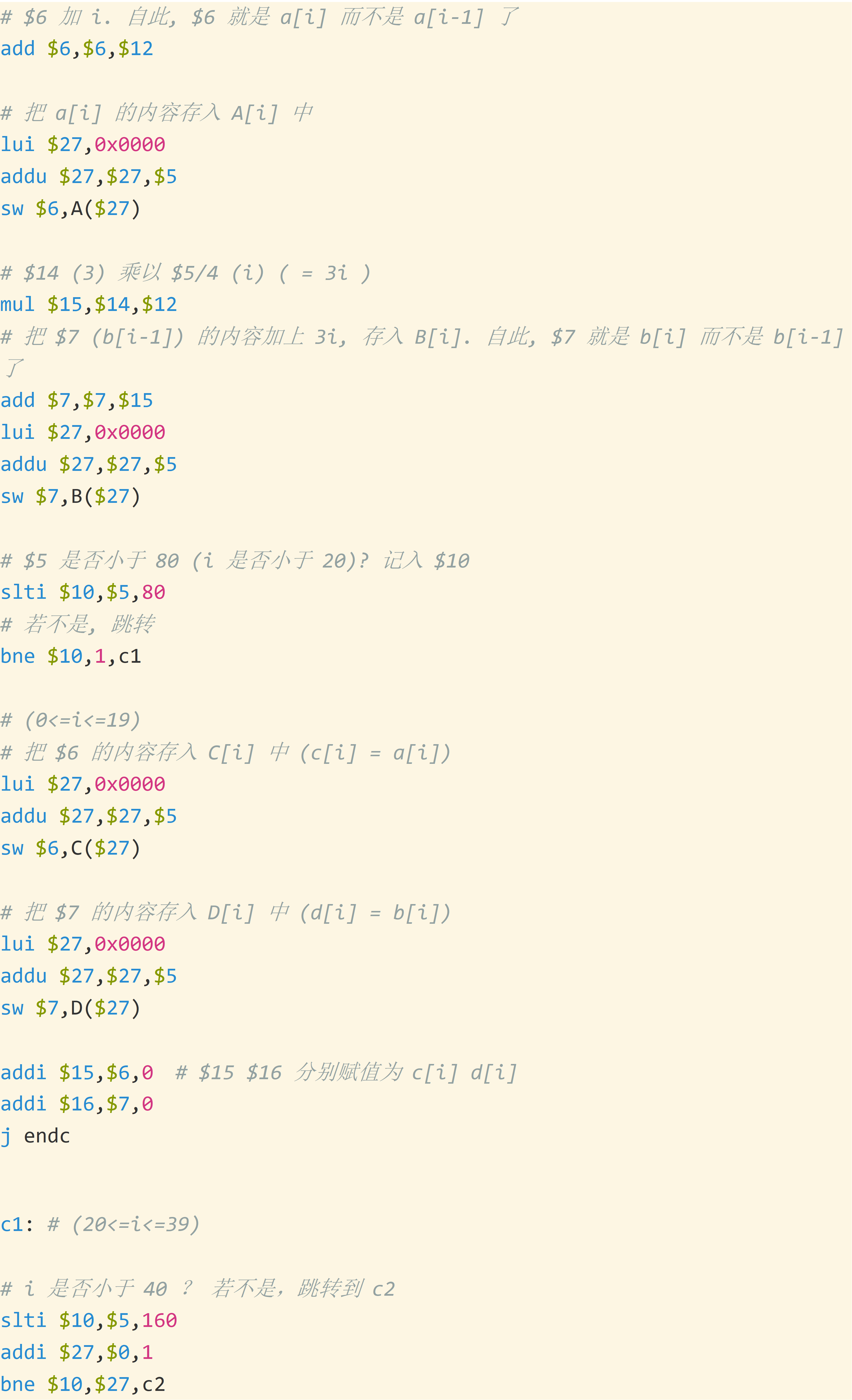


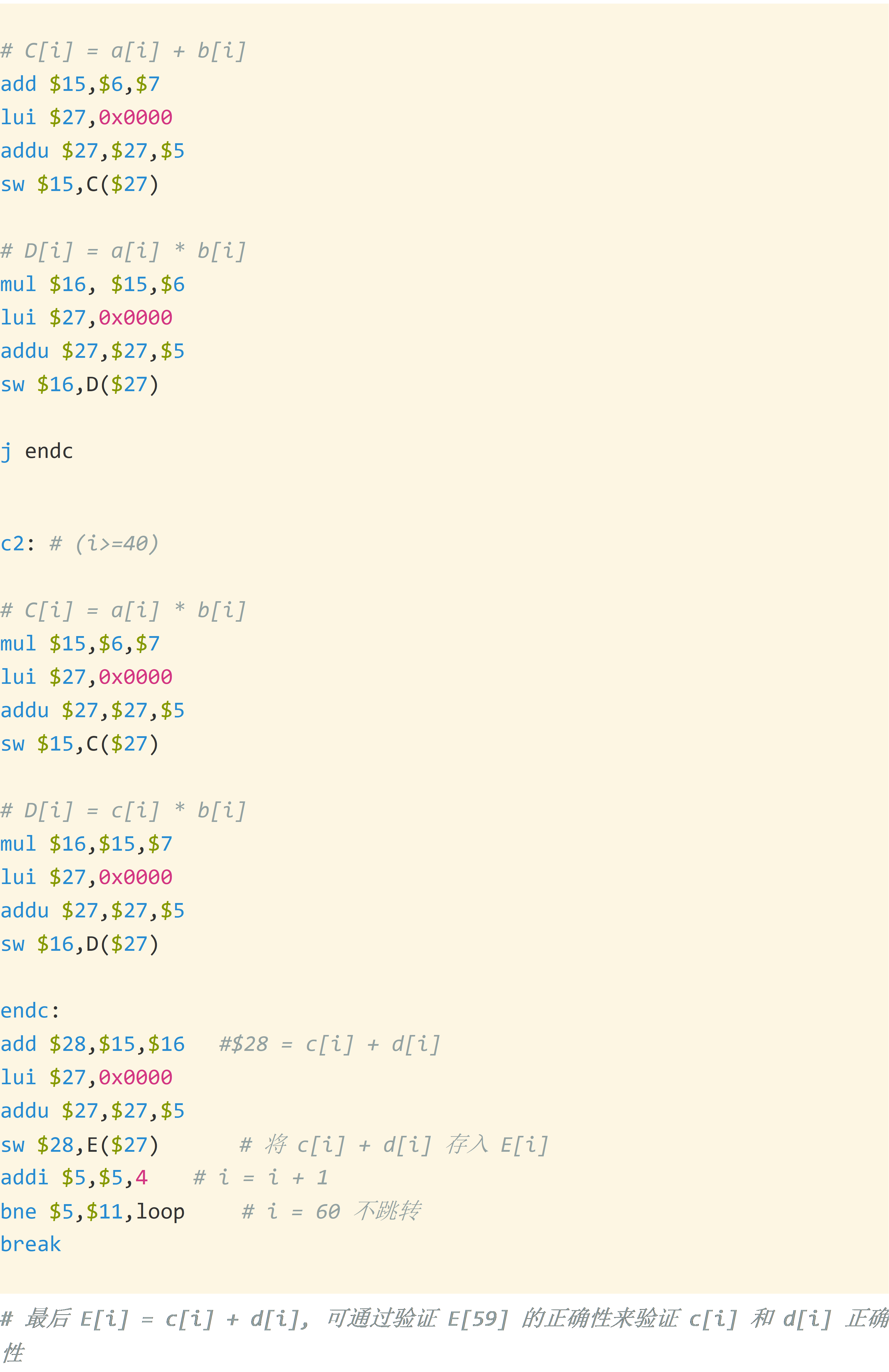
给出的数学模型



数学模型对应的算法流程图

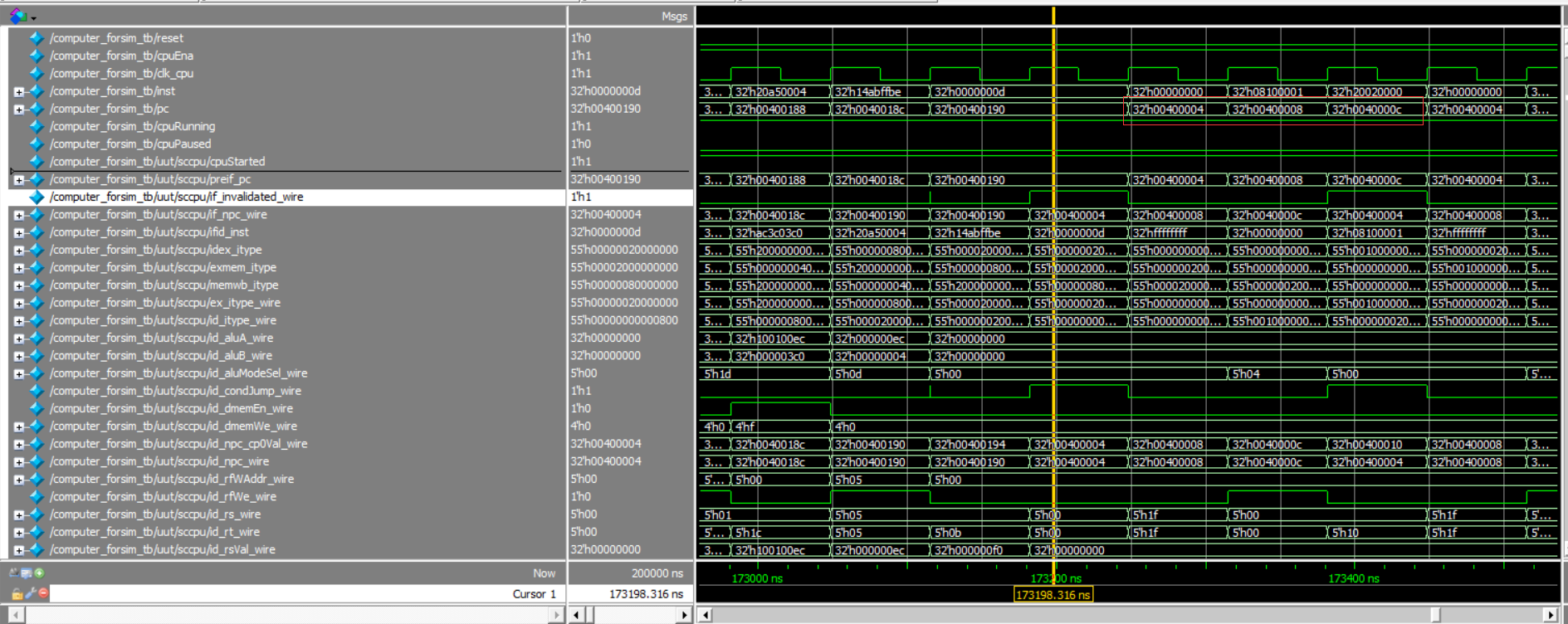




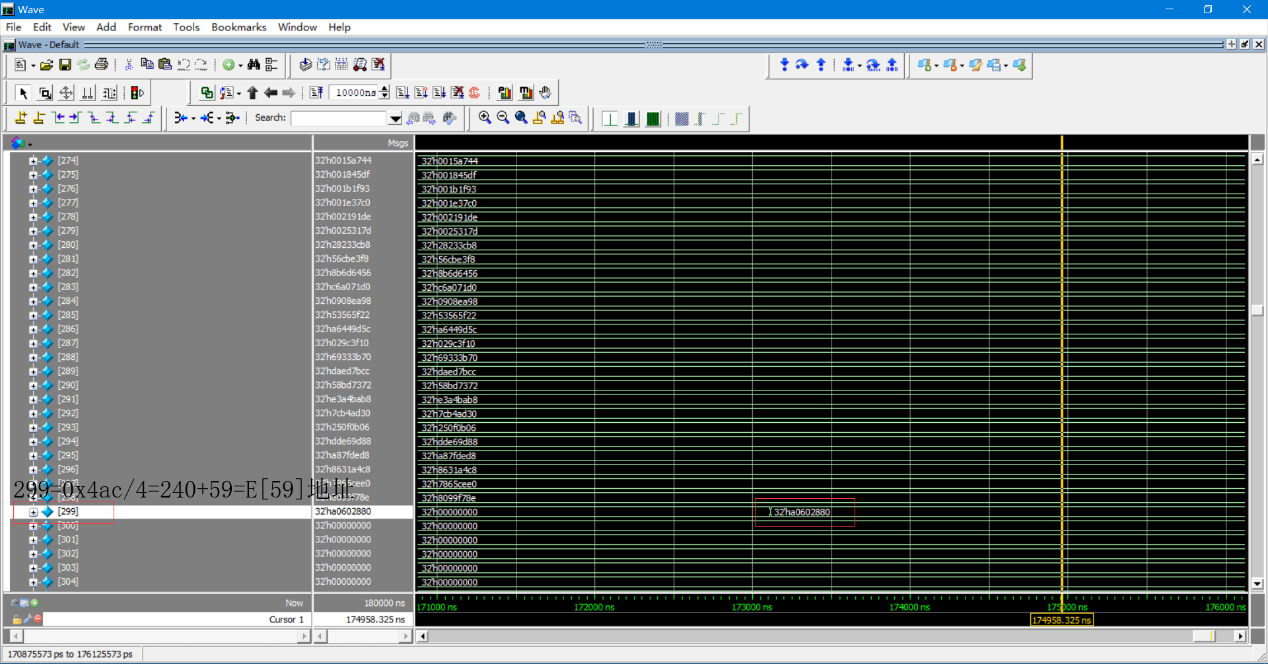


数学模型对应的代码

## 实验验算程序仿真过程



如图，在约173200ns，PC陷入了0x0040004-0x00400008-0x0040000c的循环（0x0040000c对应无效化的延迟槽），证明程序终止。在第299个DMEM字（299=240+59）查看E[59]的内容：

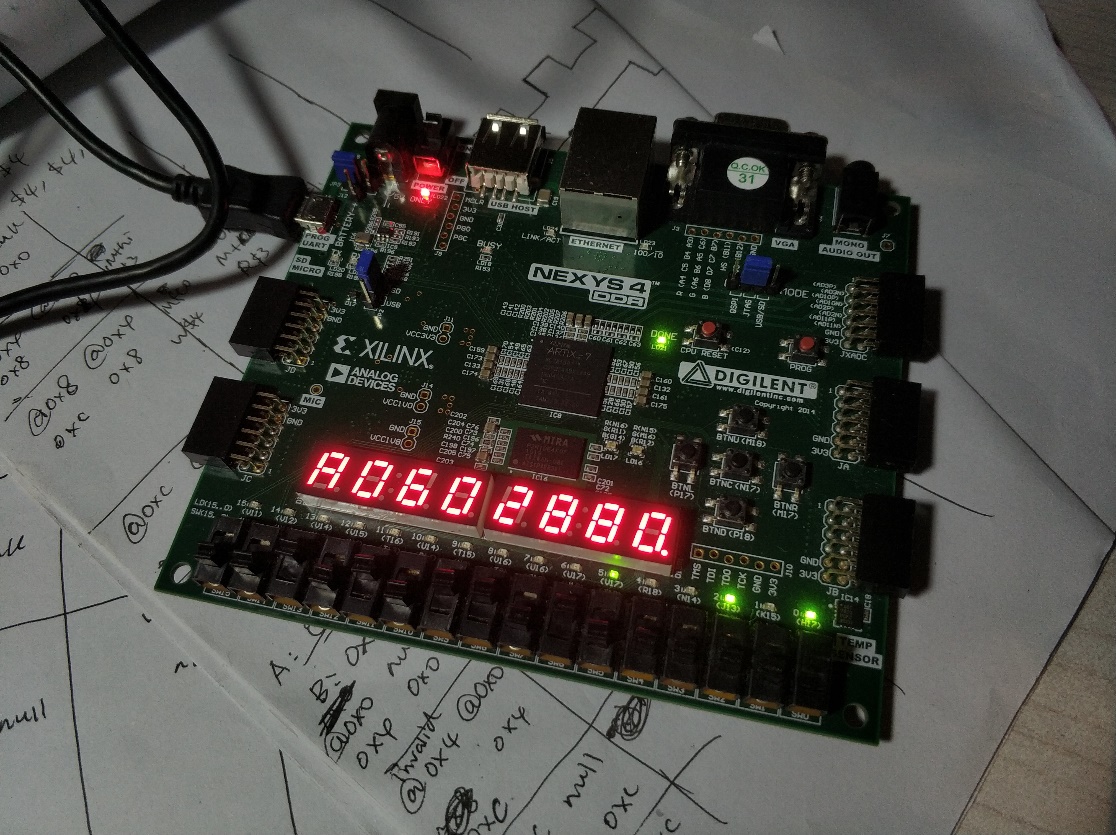


是A0602880，运行结果正确，数学模型得到验证。

## 实验验算程序下板测试过程与实现

编写可以输出DMEM内容的顶层模块topmodule.v（见工程中文件），将工程综合、实现、生成比特流后下载到开发板进行运行。可以直接下载预编译好的SuperCyclone.bit文件。

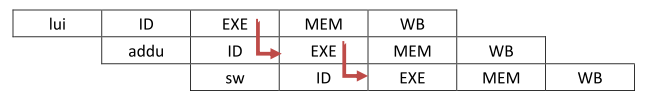
将十五个开关从左到右依此拨为“关关开关开关开开，关开关关关关关开”（左8位00101011为E[59]所在字的序号0x4ac/4的低8位，右侧的开关表示在调试输出模式下输出内存序号为0x1XXXXXXXX（X表示左侧开关对应的低八位）的字的值），按下红色的CPU\_RESET按钮重置CPU，再按下“中”键。程序应该立刻跑完终止，七段数码管显示E[59]的值A0602880。验证了数学模型，CPU制作正确。



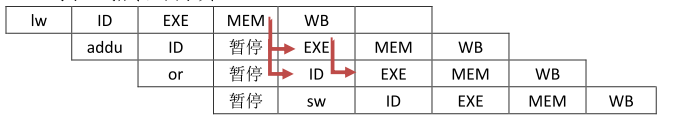
## 流水线的性能指标定性分析

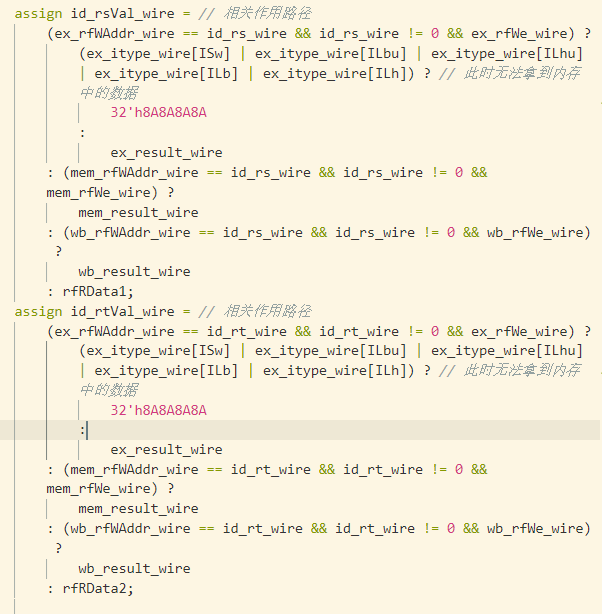
### 数据相关的情况

当距离较近的指令（相邻、隔一条指令、隔两条指令）存在对同一寄存器先写后读的情况时，称为数据相关。数据相关采用相关作用路径的方式解决，即能够在EX段或MEM段确定的该寄存器写入的结果，通过一条旁路导入需要读该寄存器的地方，用多路选择器最终选定该寄存器的读出值。



对于大多数情况，可将EX的结果导入ID，不会产生气泡。但是，对于前一条指令为从存储器加载到寄存器的指令、后一条指令为需要读到这个寄存器的指令的情况，由于前一条指令必须到MEM才能产生结果，所以必须插入一条指令的“气泡”（从ID处产生一条空指令，ID及之前的流水段不前进，后面的前进），再用相关作用路径。

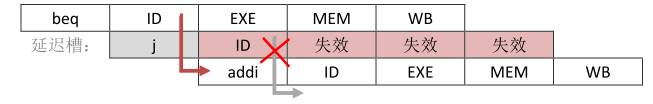




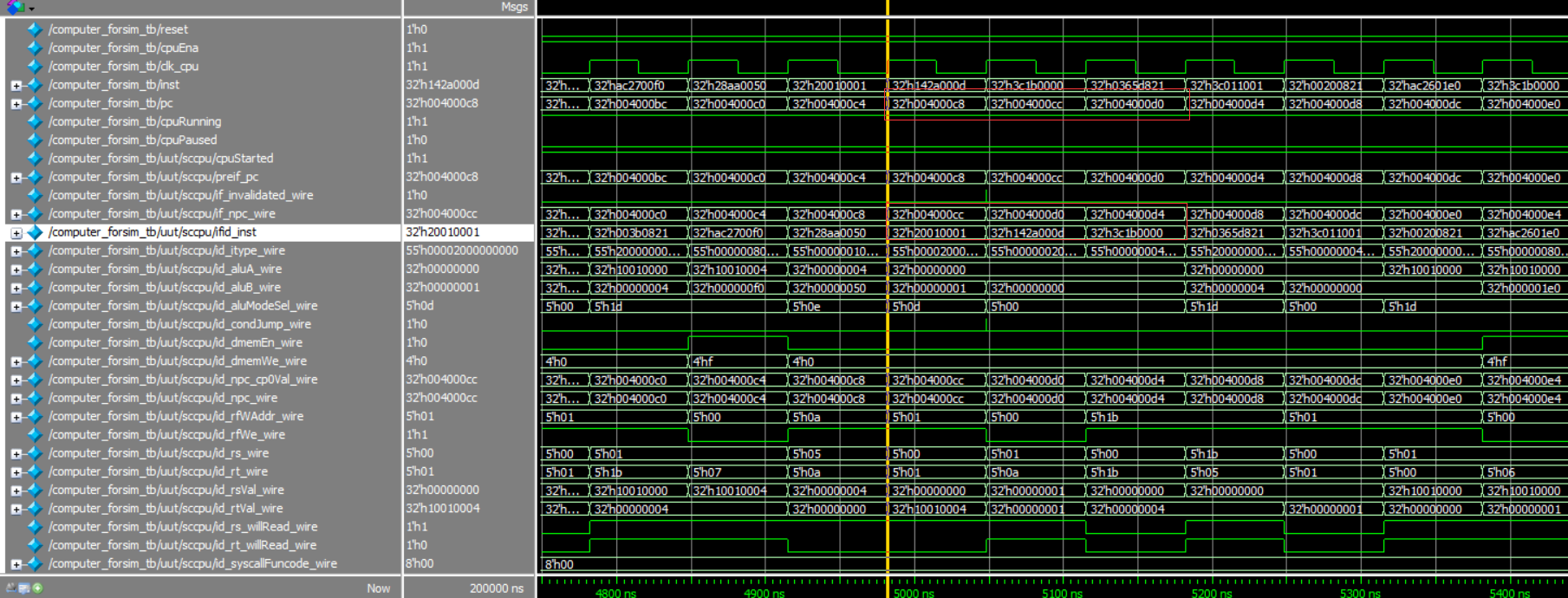
相关作用路径有关的Verilog HDL代码

### 控制相关的情况

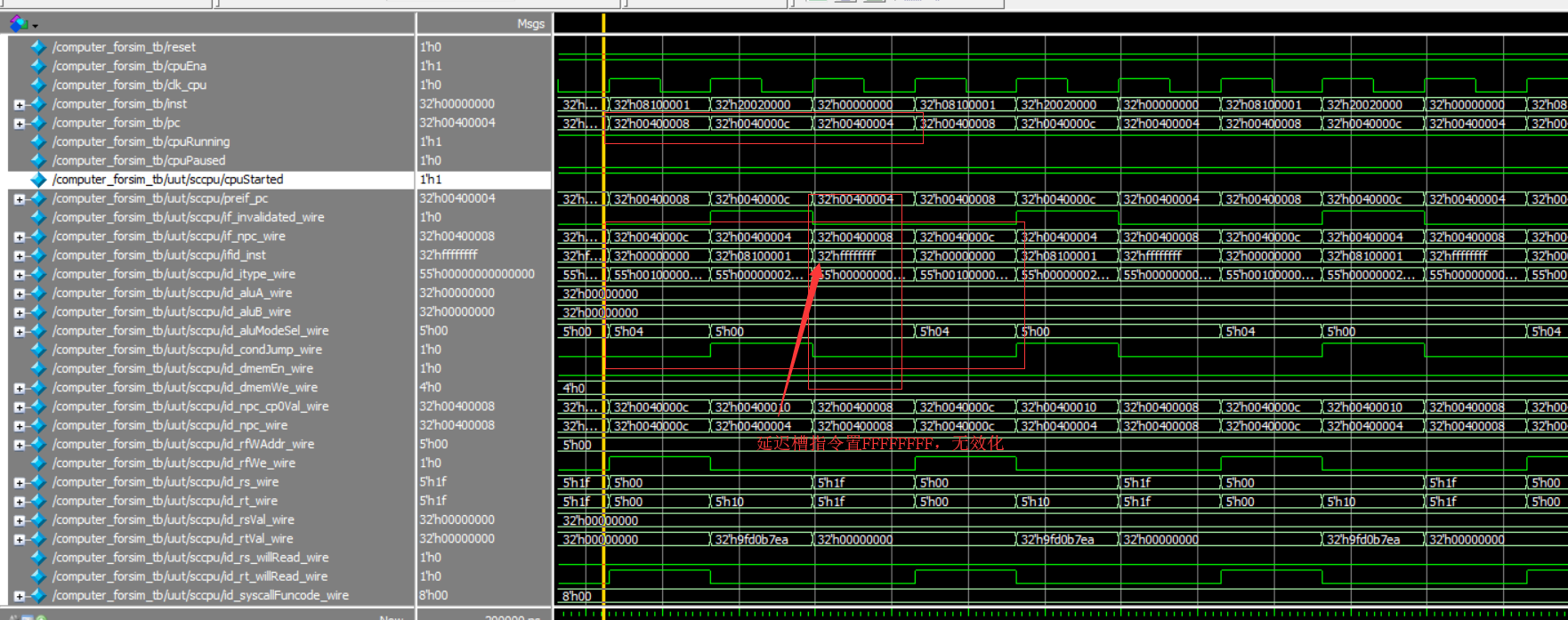
对于分支或跳转指令，先默认转移失败。当ID段检测到转移成功时，应立即将前一条已经取指的指令无效化（此后，它作为延迟槽经过流水线）。本CPU的实现方式是，将已经取指的指令字改为无效指令0xFFFFFFFF，并在之后的功能段加以识别，不执行这条指令。



转移失败的情况，流水照常运行：



转移成功的情况，延迟槽无效化，之后跳转：



### 吞吐率

完成任务共执行了2532条指令，其中包括延迟槽158条，没有气泡（暂停）。

算上启动时间，吞吐率为：

### 加速比

与所有指令用四倍时间单周期运行相比，加速比为

### 效率

认为指令平均使用四个功能段，效率为