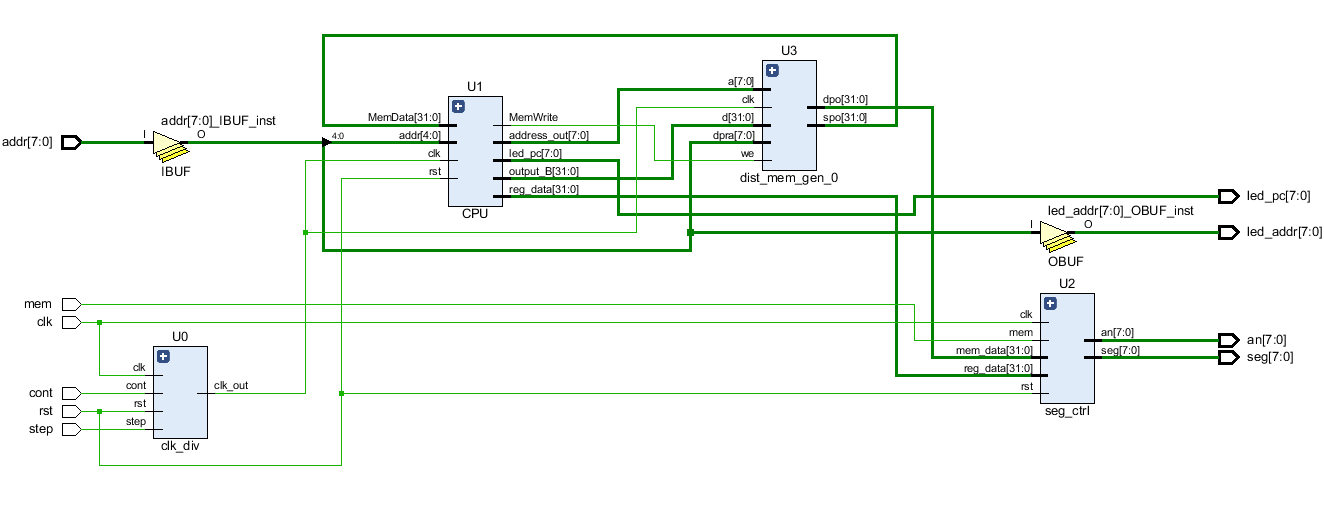
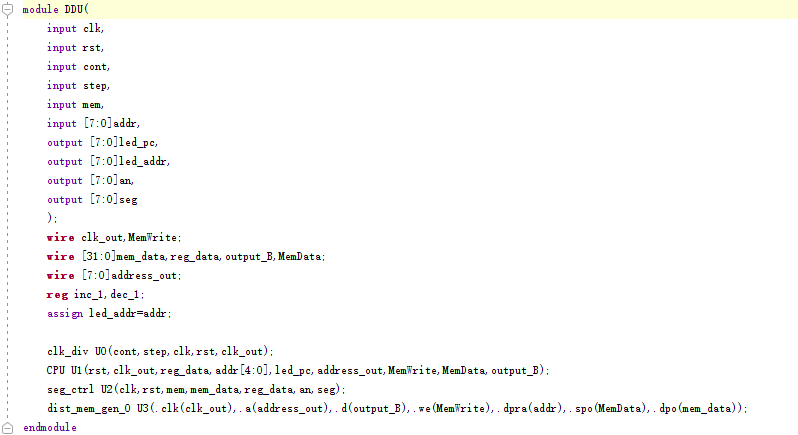
**Lab5** **多周期MIPS-CPU**

1. 设计逻辑&核心代码

使用模块化设计，大块模块一共有个：显示模块DDU，时钟分频模块clk\_div，多周期CPU，七段数码管控制模块seg\_ctrl，和dist\_mem\_gen0。

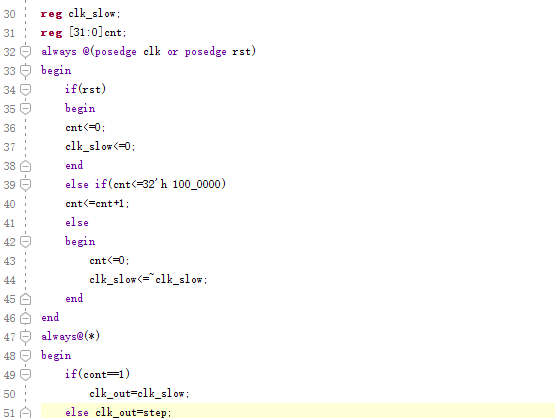


1. 显示模块DDU，用以显示CPU的运行与输出情况，因此将此模块作为顶层模块。

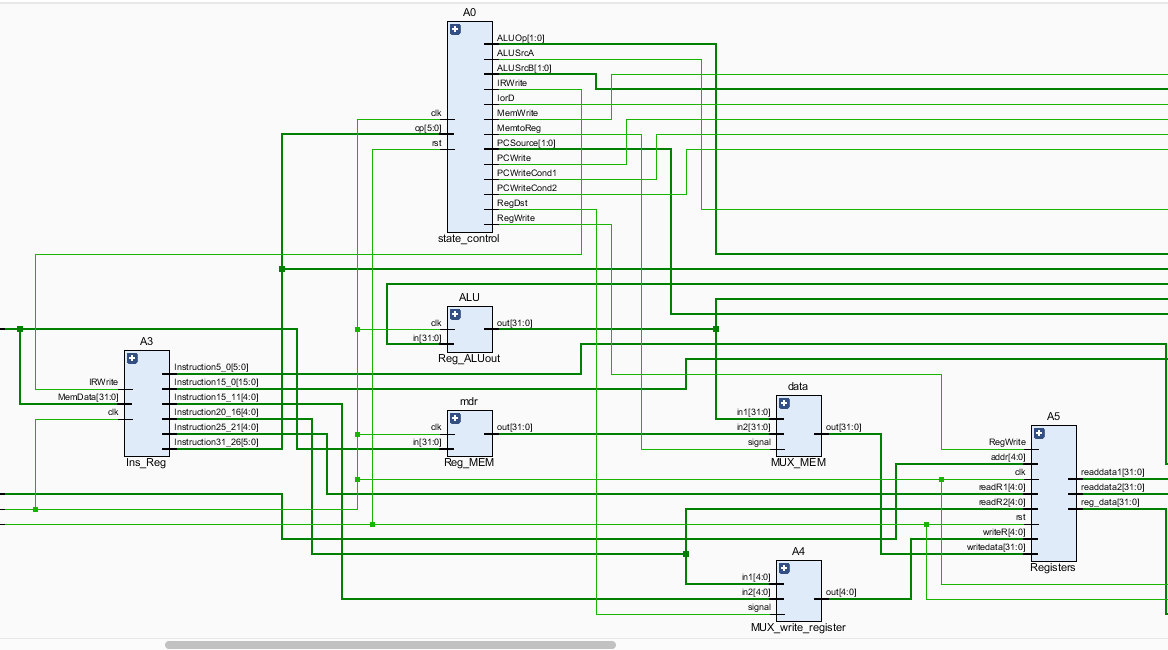


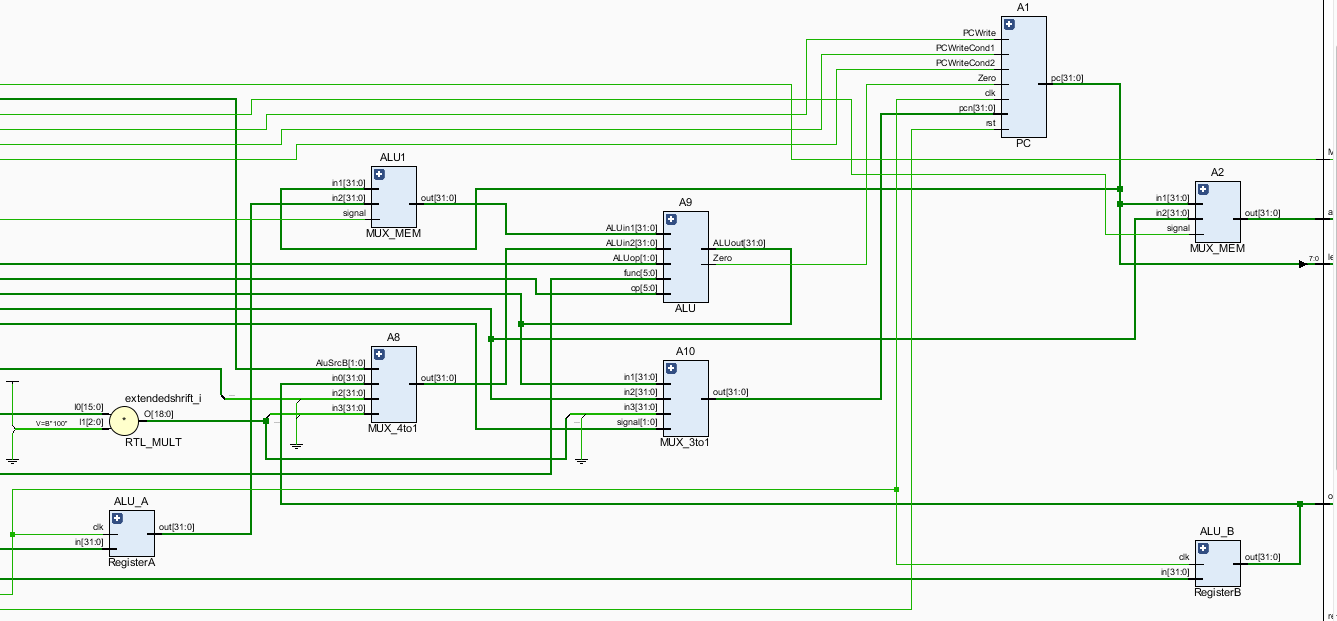
顶层模块只做输入输入接口作用，不对逻辑设计有任何影响。

1. 时钟分频模块clk\_div，把板载100MHz时钟分频成1MHz，当const=0，要求单步执行时，把时钟改为手剥时钟，即把step当作时钟，拨一下执行一个时钟周期。



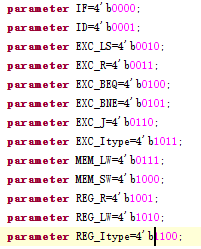
1. 多周期MIPS CPU模块，这是设计上最核心的模块。一共有state\_contrl，PC，MUX\_MEM，Ins\_Reg，MUX\_write\_register，Registers，RegisterA，RegisterB，Reg\_ALUout，Reg\_MEM，MUX\_MEM，MUX\_4to1，ALU，MUX\_3to1，MUX\_MEM这些小模块构成。





(3.1) state\_control模块

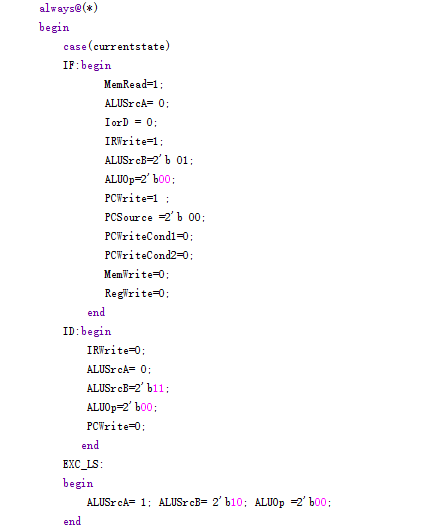
状态机状态为一个周期内执行指令的步骤。

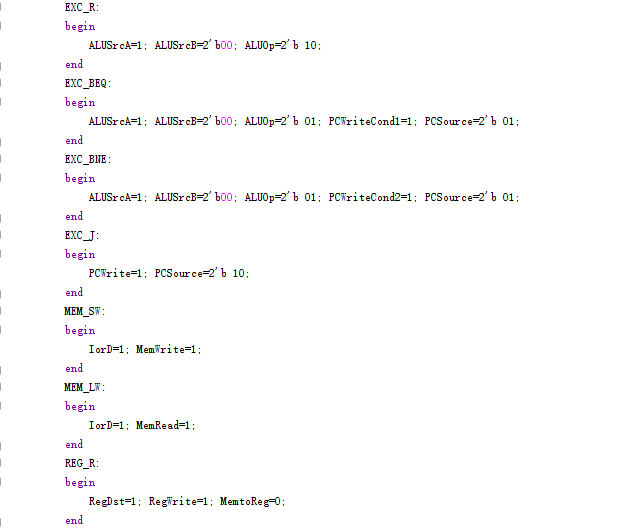


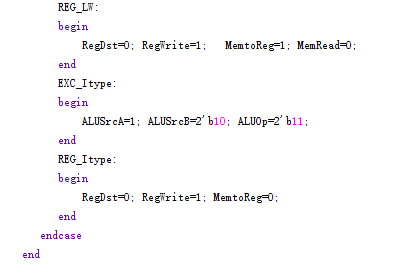
两段式状态机，第一段组合逻辑指明下一个状态



第二段通过时许逻辑电路来指明每一个状态执行怎样的操作。

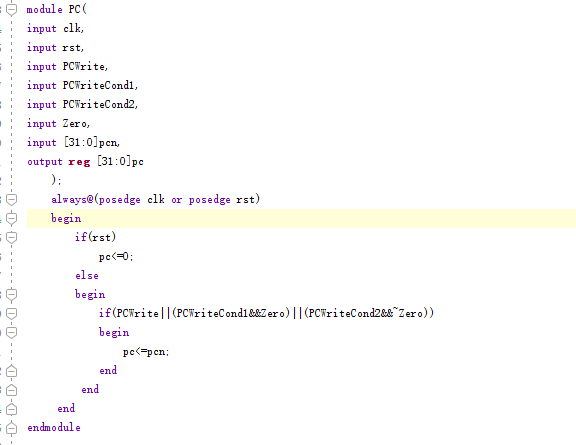




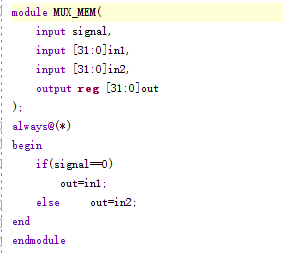


相比PPT上的状态图，补充了EXC\_Itype，REG\_Itype，EXE\_BEQ，EXE\_BNE这几个状态

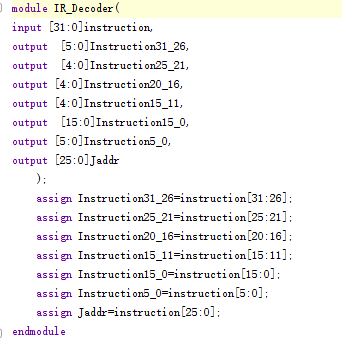
(3.2) PC模块对PC的刷新操作。执行完一条指令之后，更新PC的值。



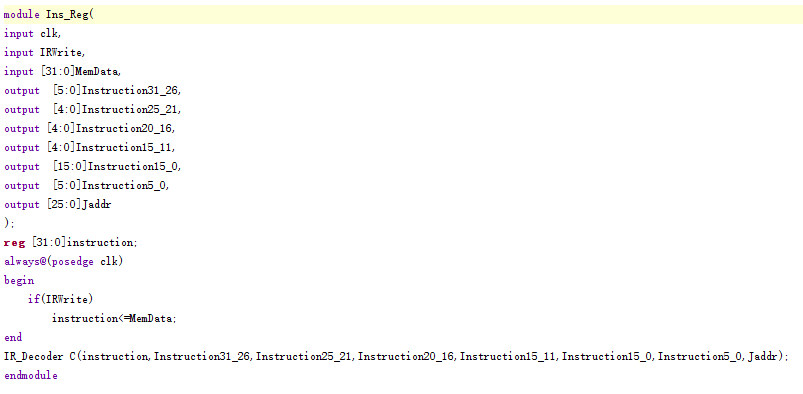
(3.3) MUX\_MEM模块，内存中指出的多选器模块，其他的多选器模块都是同一个逻辑，不再展开赘述。



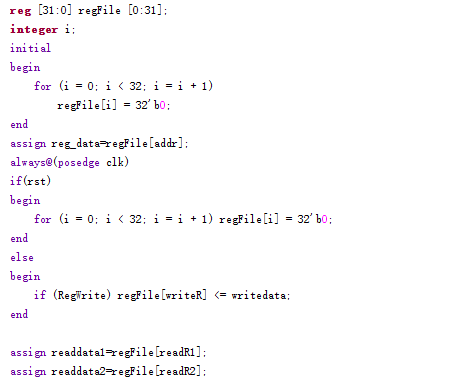
(3.4) Ins\_Reg指令寄存器模块，内部包含Decoder译码器模块。译码器模块把32位指令翻译成操作码、操作数、地址等。



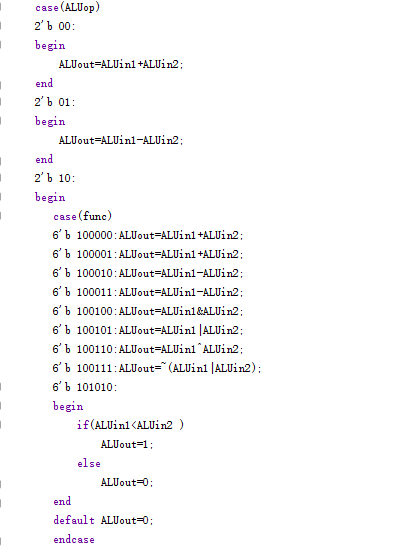
寄存器作为寄存单元用来保存数据一个时钟周期，接下来的寄存器不再赘述。

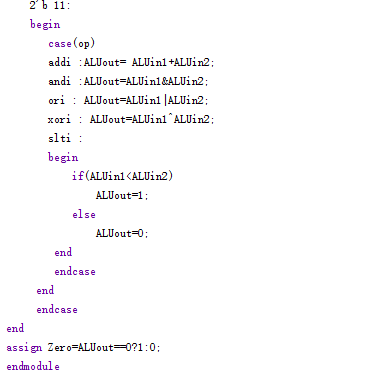


(3.5) Registers模块，寄存器堆。

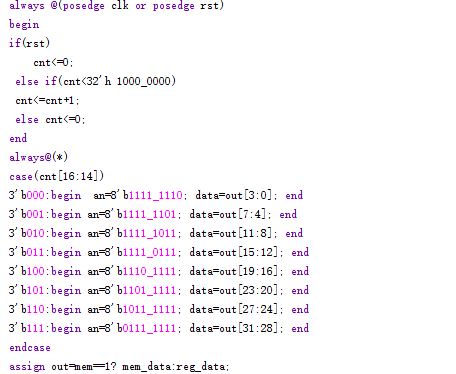


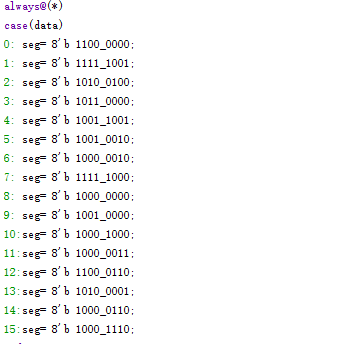
(3.6) ALU模块，CPU运算的核心模块。执行add and or xor slt运算。用状态机来写。



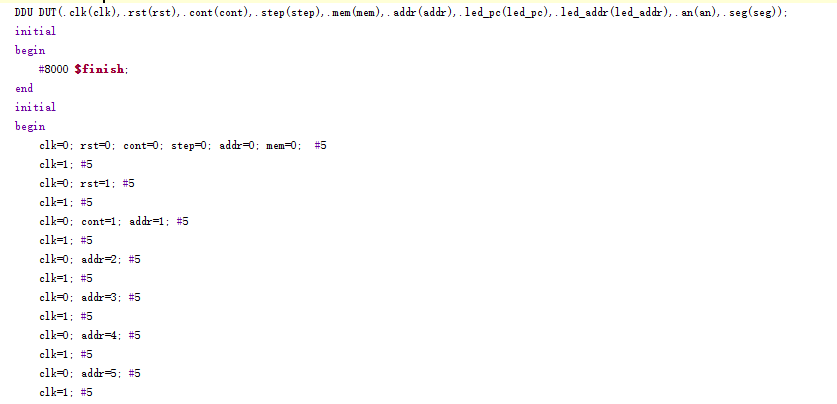


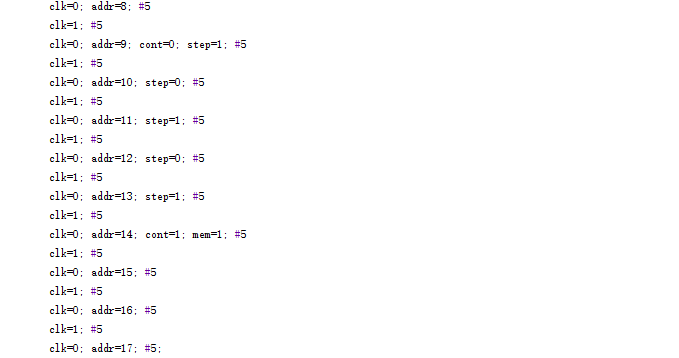
1. seg\_ctrl模块，用来控制寄存器内容或者内存地址输出到七段数码管上。



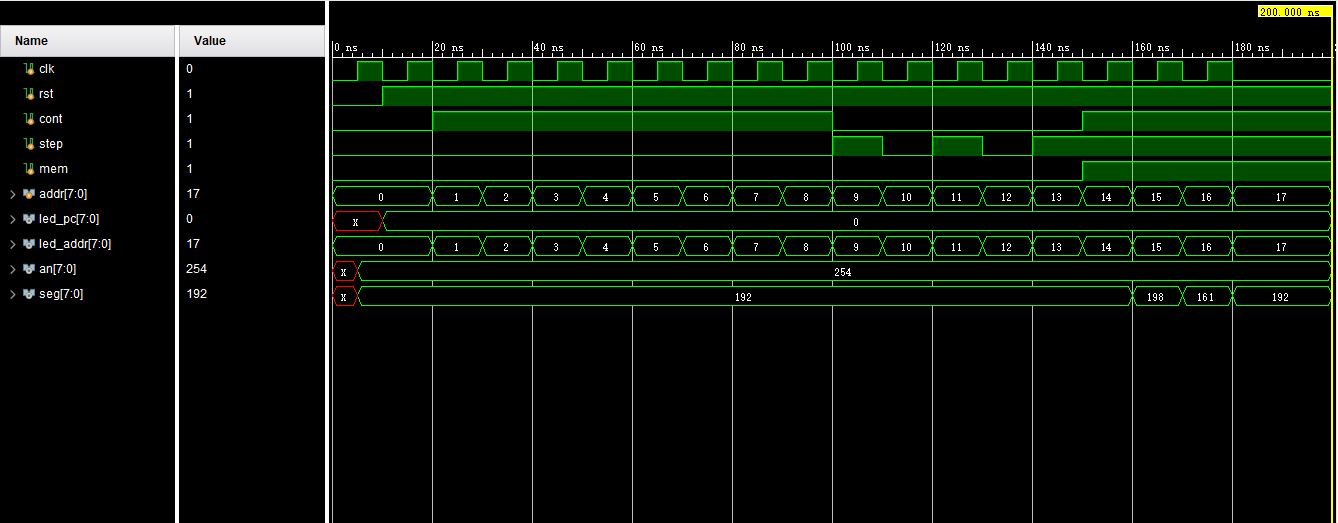


1. dist\_mem\_gen0模块，存储器，引入coe文件完成测试，相当于内存。
2. 仿真结果与下载结果：
3. 仿真



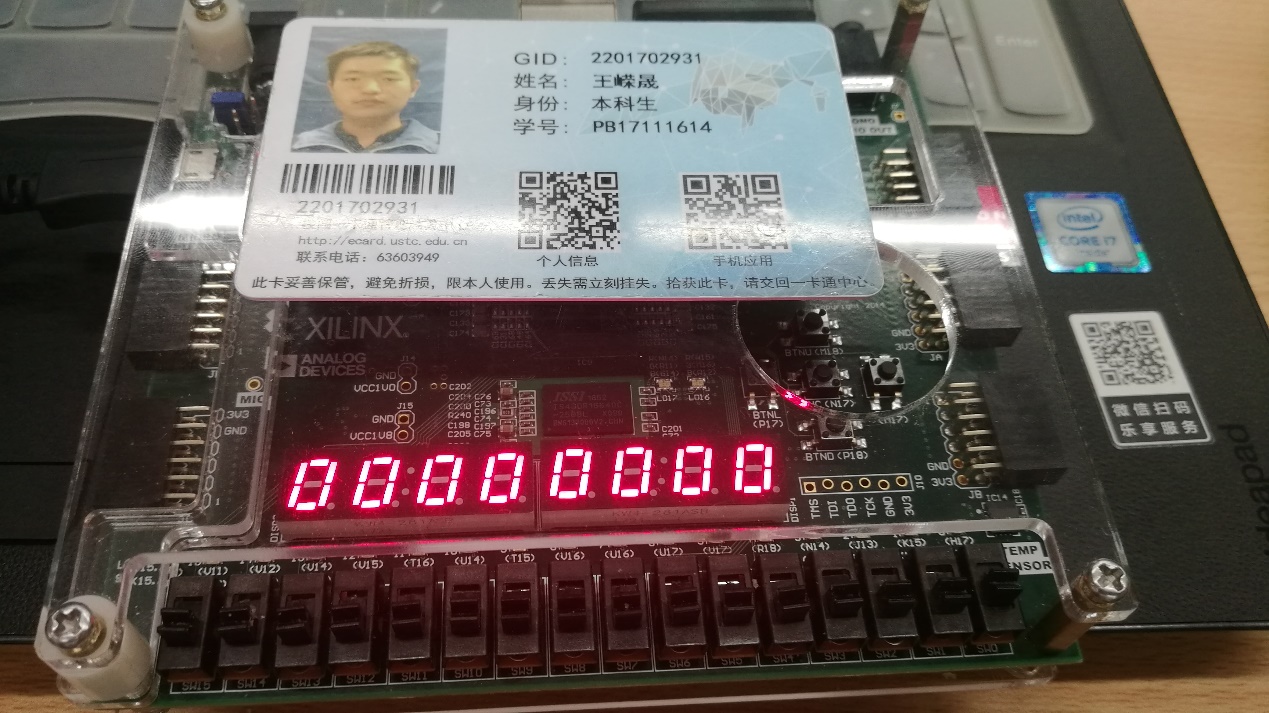


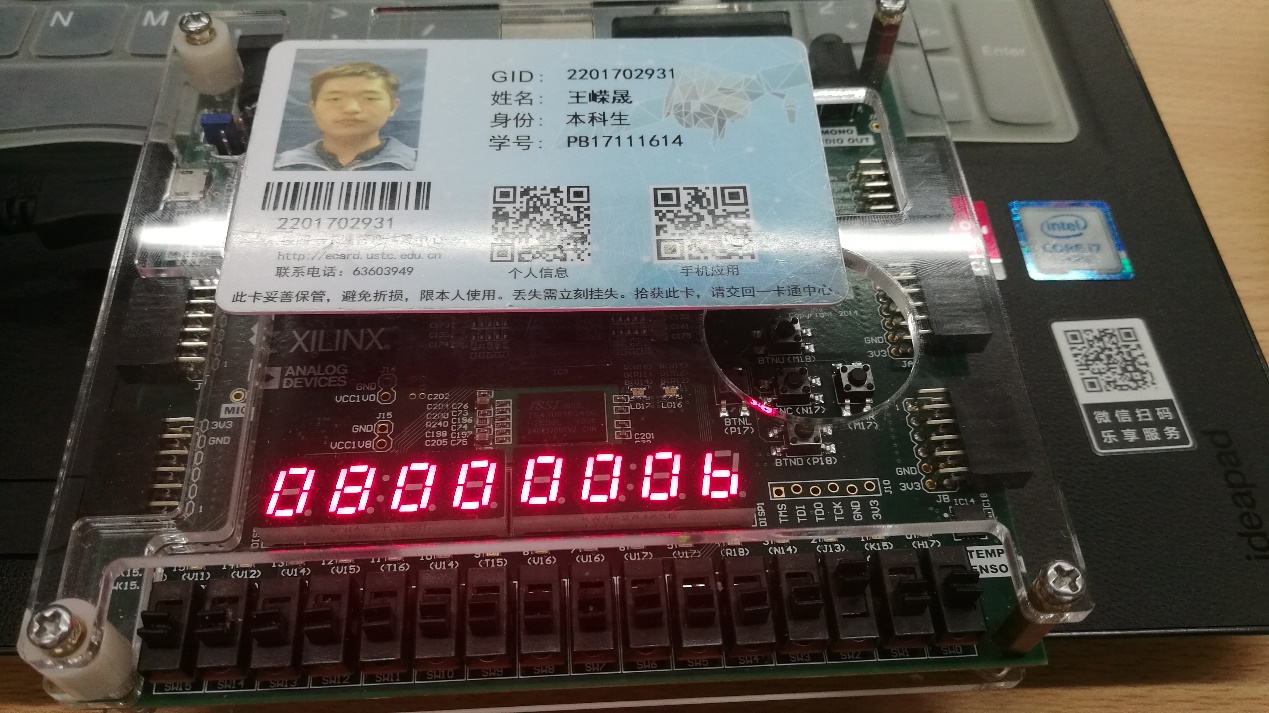
仿真代码



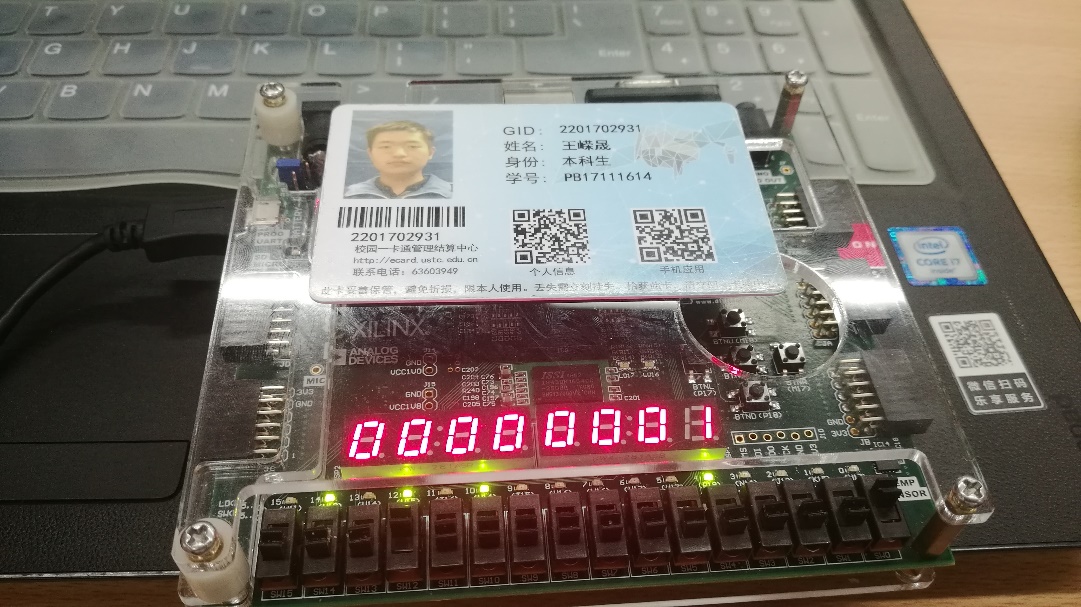
仿真结果

1. 下载













1. 结果分析

下载结果正确， 连续运行时稍微有点慢。

1. 实验总结

本次试验算是一次综合实验，把之前做的状态机、ALU、寄存器堆等试验都连接到了一起，同时结合计算机组成原理这门课上所学的多周期MIPS CPU原理，才能完成这次实验，难度的确不小。写控制信号的状态机是一个难点，另外由于模块众多，对连接总线也是充满了难度。不过学习过这个试验之后，对CPU的组成有了一定的深入理解。