**实验报告**

**物联1601 刘小芬 201608010107**

**实验名称**

相对简单CPU电路设计

**实验目标**

利用VHDL设计相对简单CPU的电路并验证

**实验要求**

* 采用VHDL描述电路及其测试平台
* 采用时序逻辑设计电路
* 采用从1累加到n的程序进行测试

**实验内容**

**相对简单CPU的规格说明**

* 地址总线16位，数据总线8位
* 有一个8位累加寄存器AC，一个8位通用寄存器R，一个1位的零标志
* 有一个16位AR寄存器，一个16位程序计数器PC，一个8位数据寄存器DR，一个8位指令寄存器IR，一个8位临时寄存器TR
* 有16条指令，每条指令1个或3个字节，其中操作码8位。3字节的指令有16位的地址

**相对简单CPU设计方案**

1. 指令执行过程分为取指、译码、执行三个阶段
2. 取指包括三个状态，FETCH1，FETCH2，FETCH3
3. 译码体现为从FETCH3状态到各指令执行状态序列的第一个状态
4. 执行根据指令的具体操作分为若干状态
5. 执行的最后一个状态转移到FETCH1状态
6. 控制器根据每个状态需要完成的操作产生相应的控制信号

**设计内容**

增加了FETCH4状态一共四个状态。取址包括四个状态，而译码体现为从FETCH4状态到各指令执行状态序列的第一个状态。执行根据指令的具体操作分为不同的状态，并且其最后一个状态转移到FETCH1状态。

C:\Users\user\AppData\Roaming\Tencent\Users\470637926\TIM\WinTemp\RichOle\OD$}Q}%Z6[QC1$4_]F(~KTJ.png

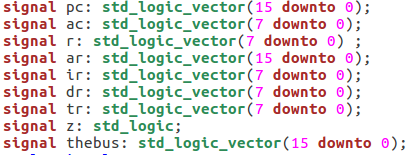
C:\Users\user\AppData\Roaming\Tencent\Users\470637926\TIM\WinTemp\RichOle\`6$583Z7`GFD[FD7W7QQB}S.png

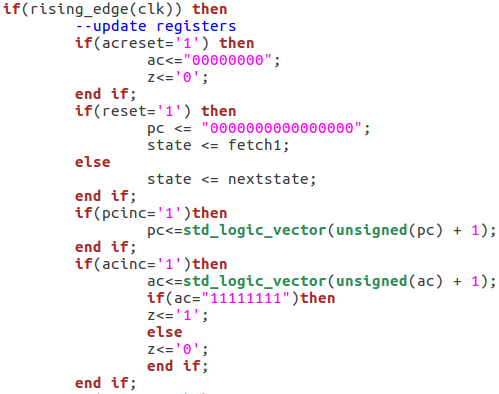
C:\Users\user\AppData\Roaming\Tencent\Users\470637926\TIM\WinTemp\RichOle\`KYV_]]AOZ68@[0YB6{[230.png

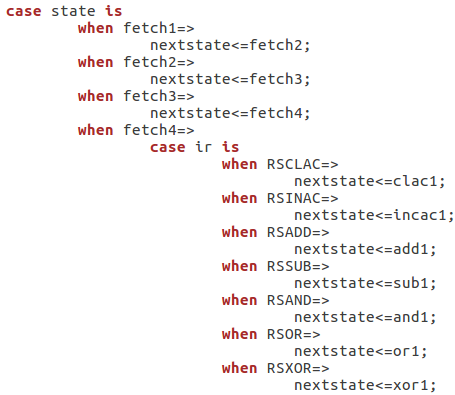
C:\Users\user\AppData\Roaming\Tencent\Users\470637926\TIM\WinTemp\RichOle\$6{D}IC3B}G~IC@08Q$1OVY.png

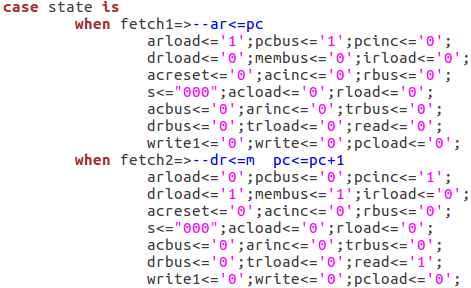
在cpu.vhd文件中，声明了cpu的内部总线、寄存器、控制信号和所有状态所对应的编码，以及对alu相关运算进行设计，并声明了所有会执行的指令。在时钟信号的上升沿，对信号0、1时刻下对应的操作做了规定。在进程for\_nextstate中判断所处状态的下一个状态是什么，在另一个进程中gen\_controls规定所处状态下所有控制信号的0、1状态。

**其中部分代码：**

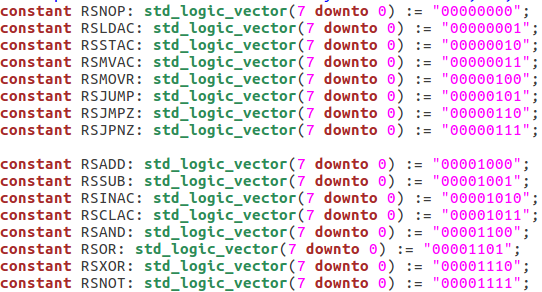






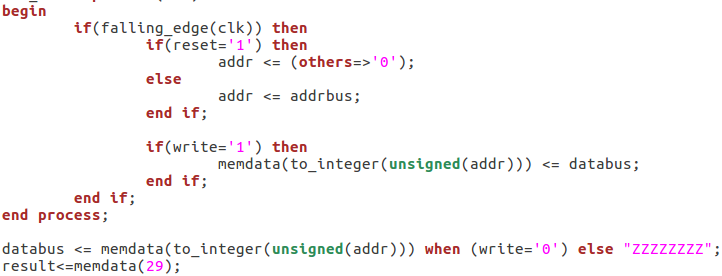


在rsisa.vhd文件中，对FETCH4所要进行的下一个状态进行了指令的一个编码，并且给它一个相关的变量名，这是为了在CPU.vhd中对变量名的判断从而执行下一个状态。



在mem.vhd文件中，对内存大小进行了规定为0到65535，之前的4096太小了会溢出。对内存所要执行的东西进行了设计，可以从1到n进行加法运算，n暂时设为8，result变量对应没进行一次运算的结果，可以很直观的看出结果是否正确。

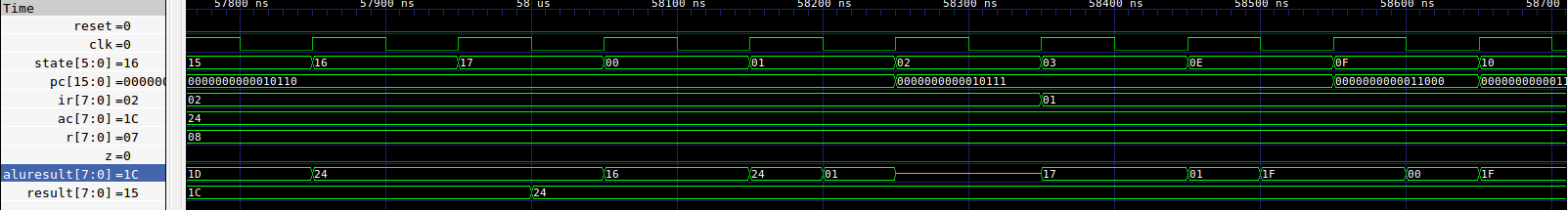
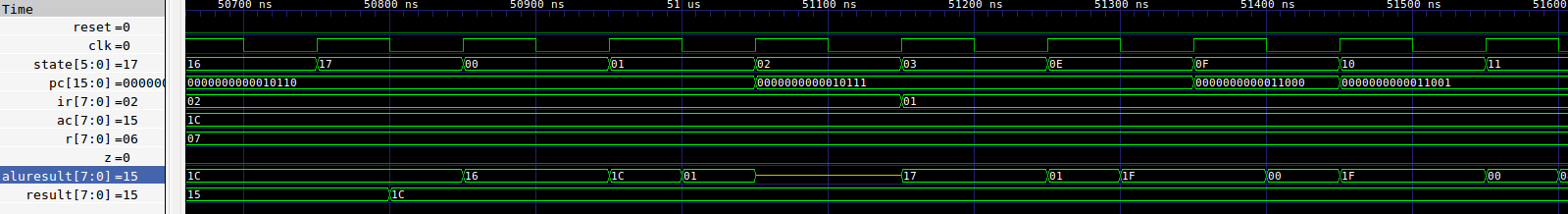
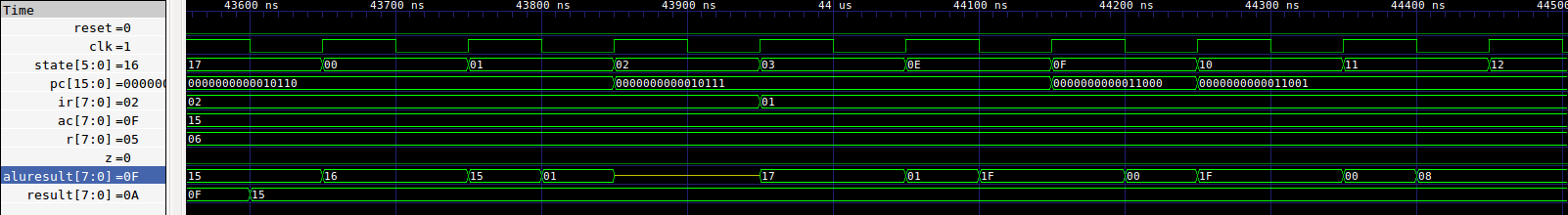
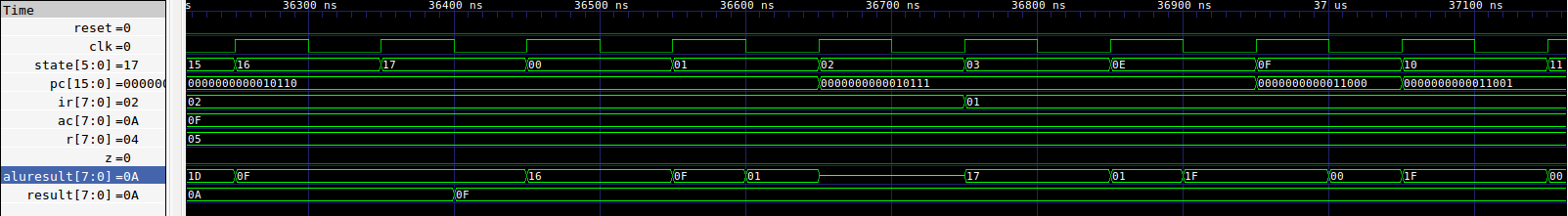
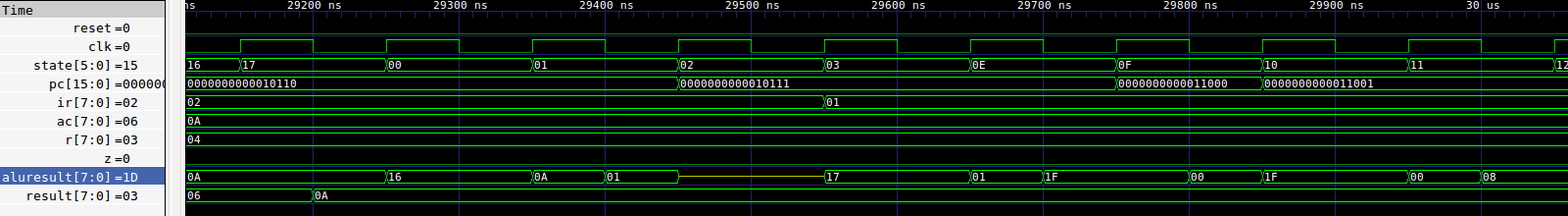
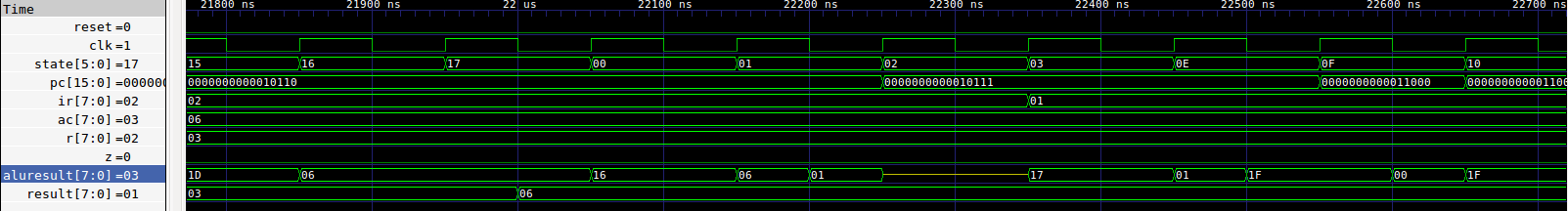
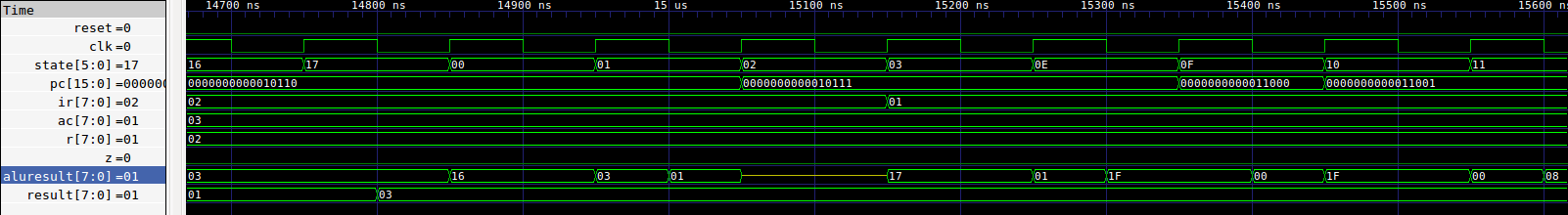
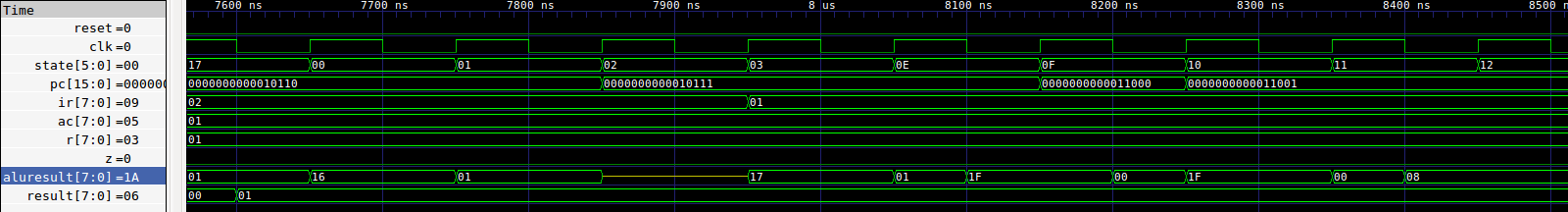
C:\Users\user\AppData\Roaming\Tencent\Users\470637926\TIM\WinTemp\RichOle\WA{F3T_H`IW64[7H19_VO_G.png



在comp.vhd文件中，对cpu和mem端口进行了连接，并规定了时钟周期和reset的时间大小。

**C:\Users\user\AppData\Roaming\Tencent\Users\470637926\TIM\WinTemp\RichOle\PNP5`FLYIZ~J8F[I(2XFJC3.png**

**仿真结果**



可以看到仿真结果是正确的，到了0x24就不再变化了也就是36。