CPU设计报告

201508010117 张　奎

部件说明：

CPU.vhd：CPU的主框架，里面有CPU的输入输出，定义了控制信号，还有实例化了各个组件

Registers.vhd：通用寄存器组，由三个寄存器组成，分别用00,01,10指定。

有三个数据输入口：来自RAM、来自ALU、来自CPU输入。有两个输出，分别是输出CPU和输出到暂存器或MAR。

SA.vhd：SA暂存器，暂时存储数据，给ALU提供操作数输入的寄存器

SB.vhd：SB暂存器，暂时存储数据，给ALU提供操作数输入的寄存器

ALU.vhd：执行逻辑计算的模块(M=1)，也允许数据直接通过(M=0

RAM.vhd：分为存指令域和存数据域两部分，各是16x8的寄存器组

IR.vhd：指令寄存器，RAM的指令一条条输出到这里

ID.vhd：指令译码器，接受从IR传过来的指令并进行译码，并产生各种控制信号控制CPU其他组件的运行

部件说明：

MAR.vhd：地址寄存器，里面存了目前访问内存的位置

PC.vhd：指令计数器，每执行一条指令都加一

ChooseMAR.vhd：选择器，即是选择输入MAR的数据，有PC和通用寄存器组两个来源

ChooseSB.vhd：也是选择器，选择输入SB的数据，有通用寄存器组和RAM两个来源

指令流程

一、 设计CPU文件

1. CPU的端口

CLK ：时钟

Start：启动标志

input_CPU：输入(如输入开关组)

output_CPU：输出(如数字灯)

2. 每个组件的声明

component XXX is

...

end component;

...

指令流程

3.定义控制信号

signal POWEROFF ..

signal put_outsize

4. 定义数据通路

signal SA_out ..

signal IR_out ..

5. 实例化组件

my_XXX : XXX port map(...)

指令流程

二、设计各个部件

1. RAM

nCS为1时RAM不可访问，此时输出指令

nCS为0时RAM可访问，此时当DL为1时可以从RAM数据域读数据，XL为1时可以向RAM数据域写数据

RAM接受MAR的输出作为输入，里面是访问的地址(因为RAM可看成是两个16x8的数组)

2. IR指令寄存器

当LodIR为1时从RAM读取指令，因此为了实现关机(=不再读指令)，只需将LodIR设为0即可

3. ALU运算器

S代表运算的指令，如加法对应1001,减法对应0110（见文档中的表格）,M为1代表进行算术或逻辑运算，M为0代表不进行运算，直接输出

指令流程

二、设计各个部件

4. 通用寄存器组Registers

通用寄存器组里面有三个寄存器R_A,R_B,R_C

因为RE，WE定义为低电平有效，所以重取名为nRE，nWE

nRE,RA控制读，nRE为0时可读,当RA="00",代表读取R_A的值，RA="01"代表读取R_B的值，RA="10"代表读取R_C的值

nWE,WA控制写，nWE为0时可写，WA对应的寄存器与上面一样

三条输入分别来自RAM，ALU和outside，对应其信号名

5. 指令计数器PC

PC代表运行到第几条指令，它存的初始值为"00000000"代表第一条指令，每条指令运行完后，都要对PC存的数据+1（除了JMP指令实现跳转地址的情况)LodPC为1时，PC加一

指令流程

二、设计各个部件

6. 暂存器SA，SB

本质是寄存器，LodSA或LodSB为1时output<=input

7. MAR地址寄存器

本质也是寄存器，当LodMAR为1时output<=input,由于input连着PC，output连着RAM，即是将PC存的地址传递给RAM

指令流程

三、控制系统的设计

这个CPU的控制系统全由指令译码器ID负责

ID接受从IR传过来的指令后，将八位的指令分成三部分,如

CMD R1 R2

1000 00 00

ID有非常多的输出信号，都是控制信号

然后就是对不同的指令用if分别处理

一条指令通常需要多个步骤，

体现在CPU上就是多个时钟周期，为了统一时间，将每条指令设计成4个时钟周期，

代码中的step即控制周期的信号，其中step="00"表示第一步，

以此类推

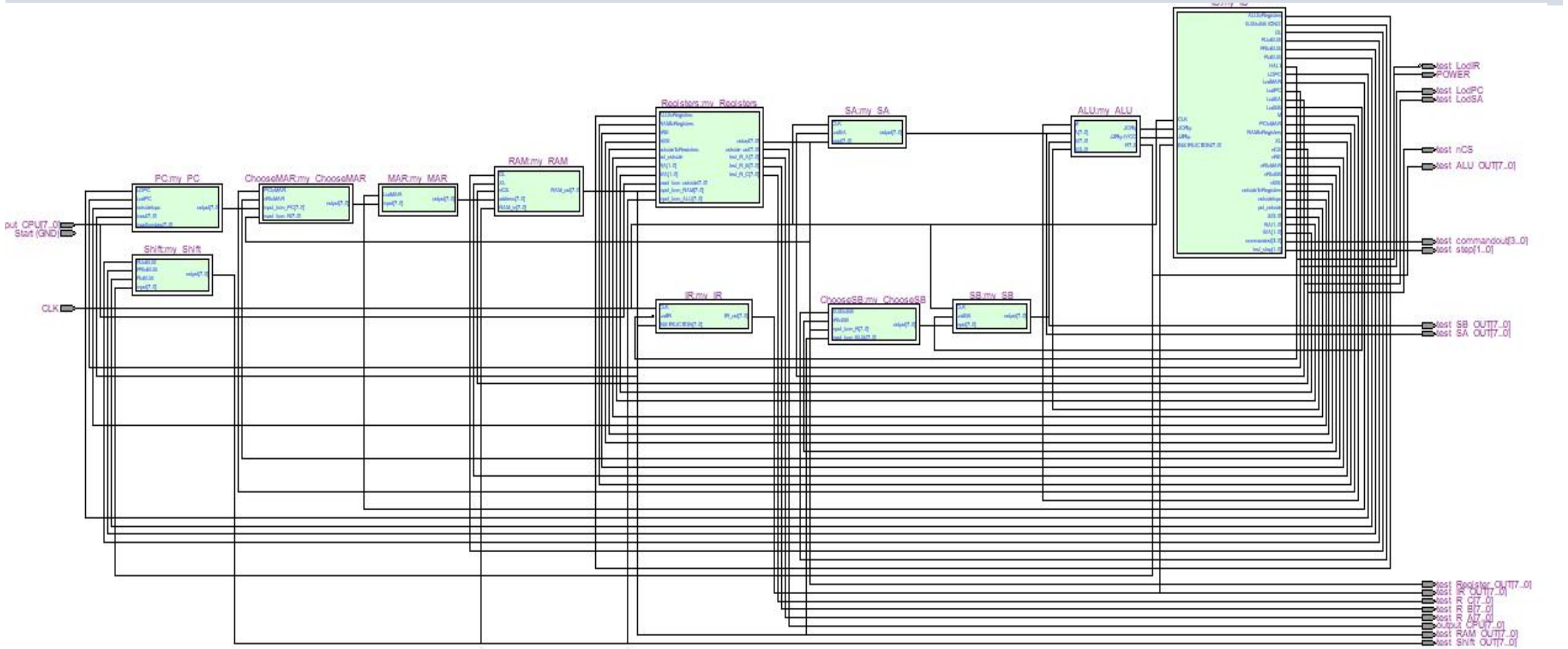
循环相加指令设计

1. 0010 00 00 : IN R_A : 0-> R_A
2. 00100100 : IN R_B : 1 -> R_B
3. 00101000 : IN R_C : 1-> R_C
4. 10010001 : A+B->A
5. 10010110 : B+C->B
6. 00010000 00000011 : jmp 3 : 跳转到第3条指令

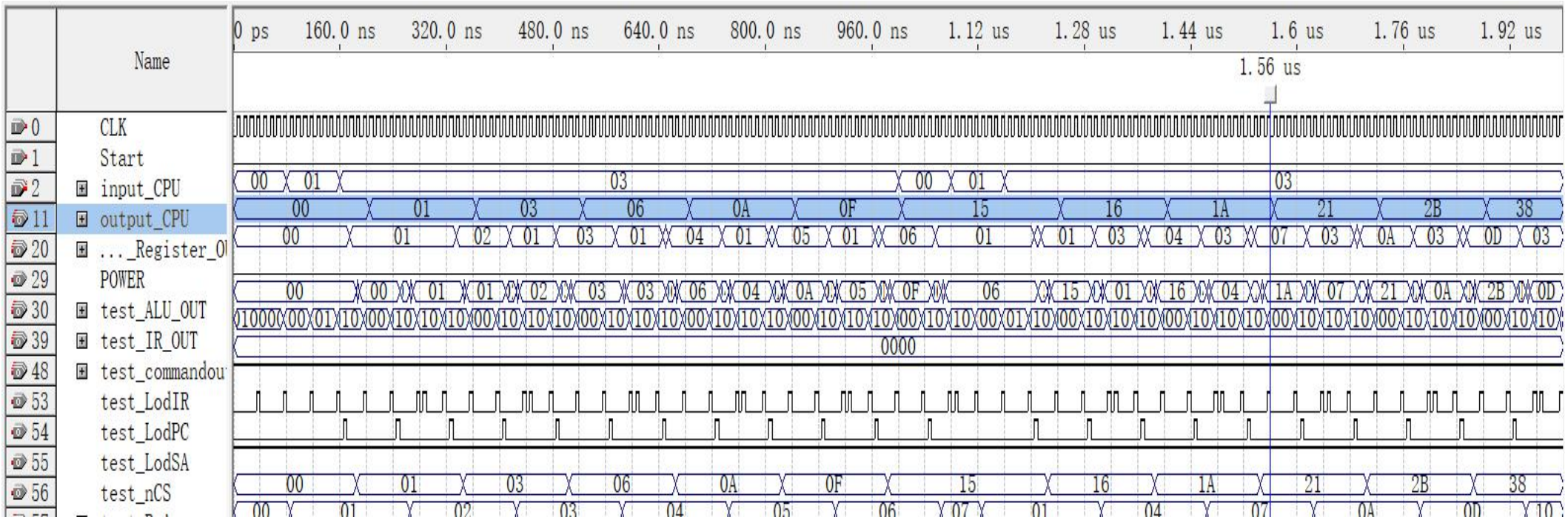
7-15 01110000 :NOP

16.10000000 :停机

NTL仿真图


PC:my_PC
ChooseMAR:my_ChooseMAR
MAR:my_MAR
RAM:my_RAM
Registers:my_Registers
SA:my_SA
ALU:my_ALU
Shift:my_Shift
IR:my_IR
ChooseSB:my_ChooseSB
SB:my_SB
CLK
Start (GND)
test LoadIR
POWER
test LoadPC
test LoadSA
test nCS
test ALU_OUT[7..0]
test commandout[3..0]
test step[1..0]
test SB_OUT[7..0]
test SA_OUT[7..0]
test Register_OUT[7..0]
test IR_OUT[7..0]
test R_C[7..0]
test R_B[7..0]
test R_A[7..0]
output_CPU[7..0]
test RAM_OUT[7..0]
test Shift_OUT[7..0]

波形仿真图



其中，output就是1+2+3+4...+n的输出结果，很明显，输出结果是正确的。