计算机设计与实践

CPU设计报告

内容包括：

1. 详细设计整体框图；

clk

**时钟模块**

rst

运算模块

T0

回写模块

T1 T2 T3

访存控制

存储管理

取指模块

ir ir ir

pc

aluout temp

pcaddr

pc rdata wr

Ir pcnew data

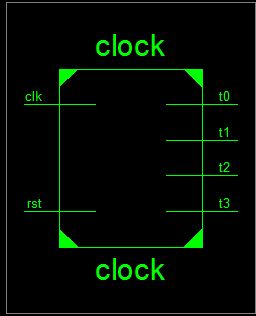
rd

存储器

Nwr nrdble bhe abus nmreq dbus

1. 各模块详细说明、数据流关系，各模块接口说明；

1.时钟模块

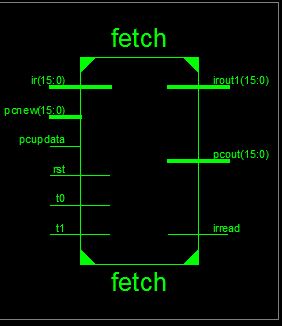


接口信号

|  |  |
| --- | --- |
| clk | 时钟信号 |
| rst | 复位信号 |
| T0 | 取指节拍信号 |
| T1 | 运算节拍信号 |
| T2 | 存储管理节拍信号 |
| T3 | 回写节拍信号 |

时钟模块由clk产生四个连续节拍t0,t1,t2,t3，分别控制之后四个模块(取指，运算，存储管理，回写)运行。Rst是复位信号，高有效时，各个模块进行清0操作.

2.取指模块



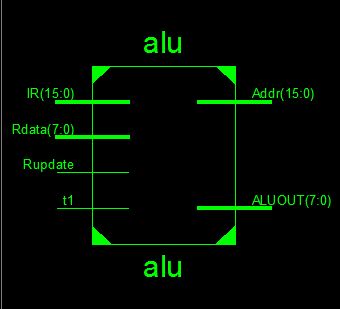
接口信号

|  |  |
| --- | --- |
| ir | 接收访存控制模块得到的新指令 |
| pcnew | 接受由回写模块回写的pc内容 |
| pcupdate | 回写pc所需的控制信号 |
| Rst | 清0 |
| T0 | 时钟模块产生的节拍，控制取指进行 |
| T1 | 控制pc+1 |
| irout | 对外输出新的指令 |
| pcout | 对外输出pc |
| irread | 读指令的控制信号 |

在t0节拍到来时，取指模块向访存控制模块发出取指令请求信号irread和指令地址pc，并将得到的ir对外输出irout；t1节拍到来时，pc+1。Pcnew是回写模块返回的pc值，pcupdate为其控制信号，并且高有效。

3.运算模块

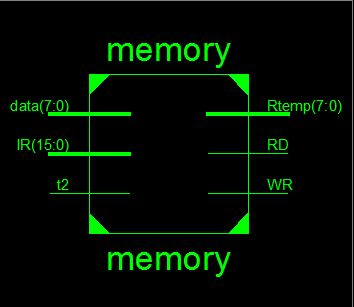
在t1节拍，由取指得到的ir进行分析，可得具体操作。若为add,sub,mov.mvi指令，结果由aluout输出到回写模块；若为lda,sta指令，访存时的地址由addr输出，数据由aluout输出；若为jmp,jz指令，新生成的pc由addr输出到回写模块；若为in/out指令，数据由aluout输出。Rdata为回写模块送回到寄存器的数据，rupdate为其控制信号。



接口信号

|  |  |
| --- | --- |
| ir | 接收由取指模块得到的指令 |
| rdata | 接受回写模块回写的数据 |
| rupdate | 回写数据的控制信号 |
| T1 | 运算模块控制节拍 |
| addr | 输出访存地址或新的pc地址 |
| aluout | 输出运算数据 |

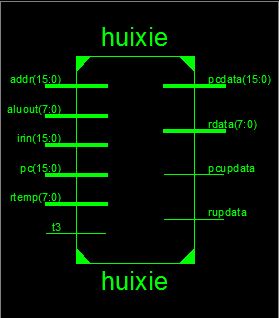
4.存储管理



|  |  |
| --- | --- |
| data | 接受alu的数据或存储器中读出的数据 |
| ir | 接受指令 |
| T2 | 控制节拍 |
| rd | 传到访存控制的读控制信号 |
| wr | 传到访存控制的写控制信号 |
| rtemp | 数据输出到回写模块或访存控制模块 |

在t2节拍，data接受数据，根据操作码可知是alu得到的数据还是控制模块的数据，并且接受的数据由rtemp输出。如果是访存指令，wr,rd某个有效，控制访问存储器。

5.回写模块

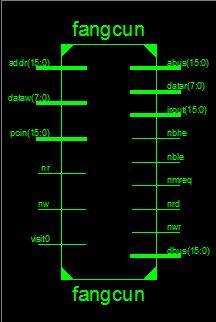


接口信号

|  |  |
| --- | --- |
| addr | 接收跳转指令生成的新pc |
| aluout | 接收运算模块非访存指令产生的数据 |
| irin | 接收指令 |
| pc | 接收pc |
| rtemp | 接收存储管理模块得到的数据 |
| T3 | 节拍信号 |
| pcdata | 输出回写的pc |
| rdata | 输出回写的数据 |
| pcupdate | Pc回写控制信号 |
| rupdate | 数据回写控制信号 |

在t3节拍，回写模块根据操作码，回写数据或pc，数据存放在rdata，pc在pcdata中，都分别有控制信号rdata，rupdate。

6.访存控制



接口信号

|  |  |
| --- | --- |
| addr | 接收运算模块传来的访存地址 |
| dataw | 接收存储管理传来的要写入存储器中的数据 |
| pcin | 接收取指模块传来的pc地址 |
| nr | 接收存储管理传来的读信号 |
| nw | 接收存储管理传来的写信号 |
| Visit0 | 取指模块传来的读取指令信号 |
| abus | 地址总线 |
| datar | 输出存储器读出的数据 |
| irout | 输出存储器读出的指令 |
| nbhe | 存储器高位有效 |
| nble | 存储器低位有效 |
| nmreq | 访存控制信号 |
| nrd | 输出访存读信号 |
| nwr | 输出访存写信号 |
| dbus | 数据总线 |

1. 各模块仿真波形、系统仿真波形；

1.时钟模块

代码：

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.STD\_LOGIC\_ARITH.ALL;

use IEEE.STD\_LOGIC\_UNSIGNED.ALL;

entity clock is

Port ( clk : in STD\_LOGIC;

rst : in STD\_LOGIC;

t3 : out STD\_LOGIC;

t2 : out STD\_LOGIC;

t1 : out STD\_LOGIC;

t0 : out STD\_LOGIC);

end clock;

architecture Behavioral of clock is

signal temp : std\_logic\_vector(3 downto 0);

begin

process ( clk, rst )

begin

if rst = '1' then

t0 <= '0';

t1 <= '0';

t2 <= '0';

t3 <= '0';

temp <= "0001";

elsif clk' event and clk ='1' then

t0 <= temp(0);

t1 <= temp(1);

t2 <= temp(2);

t3 <= temp(3);

temp(0) <=temp(3);

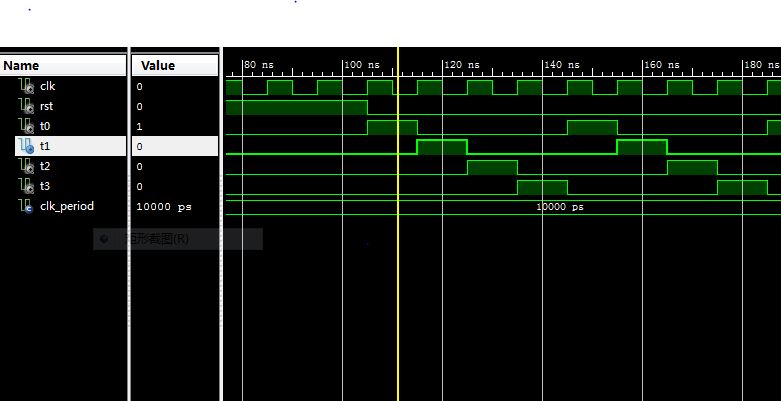
temp(3 downto 1) <=temp(2 downto 0);

end if ;

end process;

end Behavioral;

波形：



1. 取指模块

代码：

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.STD\_LOGIC\_ARITH.ALL;

use IEEE.STD\_LOGIC\_UNSIGNED.ALL;

library UNISIM;

entity fetch is

Port ( rst : in STD\_LOGIC;

t0 : in STD\_LOGIC;

t1 : in STD\_LOGIC;

pcupdata : in STD\_LOGIC;

pcnew : in STD\_LOGIC\_VECTOR (15 downto 0);

ir : in STD\_LOGIC\_VECTOR (15 downto 0);

irread : out STD\_LOGIC;

pcout : out STD\_LOGIC\_VECTOR (15 downto 0);

irout1 : out STD\_LOGIC\_VECTOR (15 downto 0));

end fetch;

architecture Behavioral of fetch is

signal pctemp,ir0 :STD\_LOGIC\_VECTOR (15 DOWNTO 0);

begin

process(t0,t1,pcupdata,pcnew,ir,rst)

begin

if rst = '1' then

pctemp <= "0000000000000000";

irread <= '0';

ir0 <= "ZZZZZZZZZZZZZZZZ";

elsif t0 = '1' then

irread <= '1';

ir0 <= ir;

elsif t1 = '1' and t1' event then

irread <= '0';

pctemp <= pctemp+1;

elsif pcupdata = '1' then

pctemp <= pcnew;

end if;

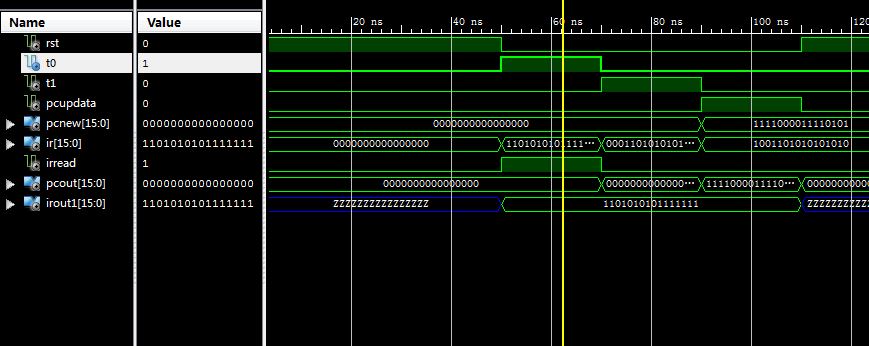
end process;

pcout <= pctemp;

irout1 <= ir0;

end Behavioral;

波形：



1. 运算模块

代码：

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.STD\_LOGIC\_UNSIGNED.ALL;

entity alu is

Port ( t1 : in STD\_LOGIC;

Rupdate : in STD\_LOGIC;

IR : in STD\_LOGIC\_VECTOR (15 downto 0);

Rdata : in STD\_LOGIC\_VECTOR (7 downto 0);

Addr : out STD\_LOGIC\_VECTOR (15 downto 0);

ALUOUT : out STD\_LOGIC\_VECTOR (7 downto 0));

end alu;

architecture Behavioral of alu is

type array\_size is array(7 downto 0) of std\_logic\_vector(7 downto 0);

signal reg : array\_size:=("00000011","00000011","00000011","00000011","00000011","00000011","00000011","00000000");

type data4 IS array(0 to 3) of std\_logic\_vector(7 downto 0);

signal io:data4 := ("11110000","11110000","11110000","11110000");

begin

process(t1)

begin

if(t1 ='1')then

case IR(15 downto 11) is

when "11111" => ALUOUT <= reg(conv\_integer(IR(10 downto 8))) + reg(conv\_integer(IR(2 downto 0)));

Addr <= "ZZZZZZZZZZZZZZZZ";

when "11110" => ALUOUT <= reg(conv\_integer(IR(10 downto 8))) - reg(conv\_integer(IR(2 downto 0)));

Addr <= "ZZZZZZZZZZZZZZZZ";

when "11101" => ALUOUT <= reg(conv\_integer(IR(2 downto 0)));

Addr <= "ZZZZZZZZZZZZZZZZ";

when "11100" => ALUOUT <= IR(7 downto 0);

Addr <= "ZZZZZZZZZZZZZZZZ";

when "11011" => ALUOUT <= reg(conv\_integer(IR(10 downto 8)));

Addr <= reg(7)&IR(7 downto 0);

when "11010" => Addr <= reg(7)&IR(7 downto 0);

ALUOUT <= "ZZZZZZZZ";

when "11000" => ALUOUT <= reg(conv\_integer(IR(10 downto 8)));

if(reg(conv\_integer(IR(10 downto 8))) = "00000000") then

Addr <= reg(7)&IR(7 downto 0);

end if;

when "11001" => Addr <= reg(7)&IR(7 downto 0);

ALUOUT <= "ZZZZZZZZ";

when "00000" => aluout <= io(conv\_integer(ir(1 downto 0)));

Addr <= "ZZZZZZZZZZZZZZZZ";

when "00001" => Addr <= "ZZZZZZZZZZZZZZZZ";

io(conv\_integer(ir(1 downto 0))) <= reg(conv\_integer(IR(10 downto 8)));

when others => Addr <= "ZZZZZZZZZZZZZZZZ";

ALUOUT <= "ZZZZZZZZ";

end case;

end if;

end process;

process(Rupdate)

variable temp : integer range 0 to 1 := 0;

begin

if(Rupdate = '1')then

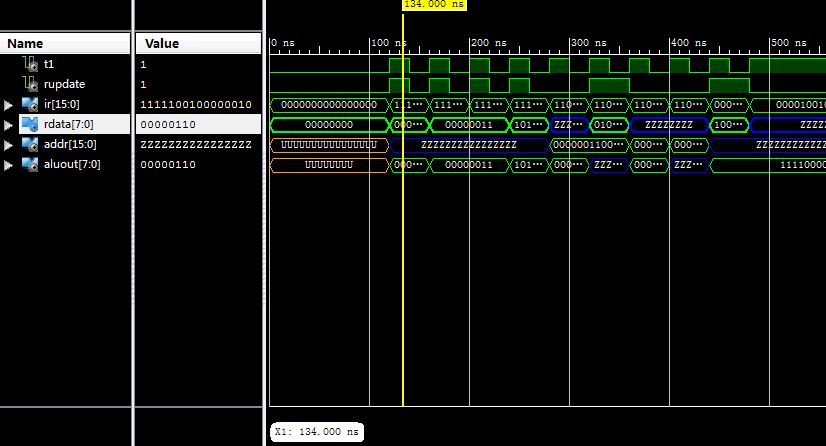
reg(conv\_integer(IR(10 downto 8))) <= Rdata;

end if;

end process;

end Behavioral;

波形：



1. 存储管理

代码：

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity memory is

Port ( t2 : in STD\_LOGIC;

IR : in STD\_LOGIC\_VECTOR (15 downto 0);

data : in STD\_LOGIC\_VECTOR (7 downto 0);

WR : out STD\_LOGIC;

RD : out STD\_LOGIC;

Rtemp : out STD\_LOGIC\_VECTOR (7 downto 0));

end memory;

architecture Behavioral of memory is

begin

process(t2)

begin

if(t2= '1')then

Rtemp <= data(7 downto 0);

case IR(15 downto 11) is

when "11011" => WR <= '0'; RD <= '1';

when "11010" => RD <= '0'; WR <= '1';

when others => WR <= '1'; RD <= '1';

end case;

else

WR <= '1';

RD <= '1';

Rtemp <= "ZZZZZZZZ";

end if;

end process;

end Behavioral;

波形：



1. 回写模块

代码：

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.STD\_LOGIC\_ARITH.ALL;

use IEEE.STD\_LOGIC\_UNSIGNED.ALL;

entity huixie is

port(

t3 : in std\_logic;

aluout : in std\_logic\_vector(7 downto 0);

rtemp : in std\_logic\_vector(7 downto 0);

pc : in std\_logic\_vector(15 downto 0);

irin : in std\_logic\_vector(15 downto 0);

rupdata : out std\_logic;

rdata : out std\_logic\_vector(7 downto 0);

pcupdata : out std\_logic;

pcdata : out std\_logic\_vector(15 downto 0);

addr : in std\_logic\_vector(15 downto 0)

);

end huixie;

architecture Behavioral of huixie is

begin

process(t3,aluout,rtemp,pc,irin)

begin

if(t3 = '1')then

case irin(15 downto 11) is

when "11111" => --add

rupdata <= '1';

rdata <= aluout;

pcupdata <= '0';

when "11110" => --sub

rupdata <= '1';

rdata <= aluout;

pcupdata <= '0';

when "11101" => --mov

rupdata <= '1';

rdata <= aluout;

pcupdata <= '0';

when "11100" => --mvi

rupdata <= '1';

rdata <= aluout;

pcupdata <= '0';

when "11011" => --sta

rupdata <= '0';

pcupdata <= '0';

when "11010" => --lda

rupdata <= '1';

rdata <= rtemp;

pcupdata <= '0';

when "11001" => --jmp

rupdata <= '0';

pcdata <= addr;

pcupdata <= '1';

when "11000" => --jz

rupdata <= '0';

if(aluout = "00000000") then

pcupdata <= '1';

pcdata <= addr;

else

pcupdata <= '1';

pcdata <= pc;

end if;

when "00000" => --in

rupdata <= '1';

rdata <= aluout;

pcupdata <= '0';

when "00001" => --out

rupdata <= '0';

pcupdata <= '0';

when others =>

rupdata <= '0';

pcupdata <= '0';

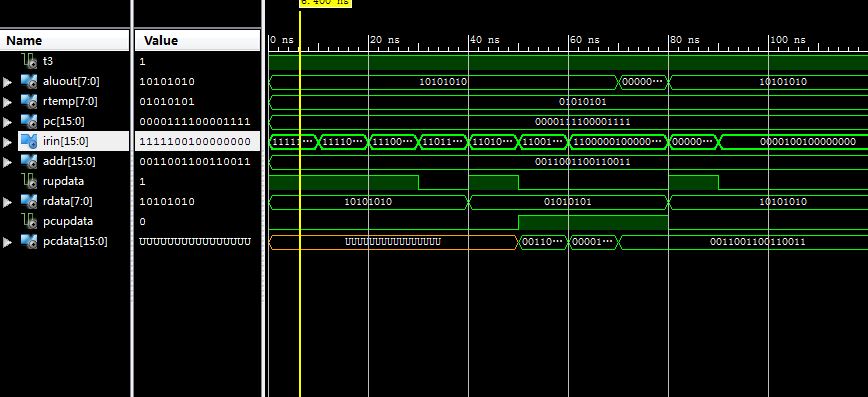
end case;

end if;

end process;

end Behavioral;

波形：



1. 访存控制

代码：

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity fangcun is

port(

visit0 : in std\_logic;

pcin :in std\_logic\_vector(15 downto 0);

irout :out std\_logic\_vector(15 downto 0);

addr:in std\_logic\_vector(15 downto 0);

nw,nr : in std\_logic;

dataw : in std\_logic\_vector(7 downto 0);

datar : out std\_logic\_vector(7 downto 0);

abus :out std\_logic\_vector(15 downto 0);

dbus :inout std\_logic\_vector(15 downto 0);

nwr,nrd,nble,nbhe,nmreq : out std\_logic

);

end fangcun;

architecture Behavioral of fangcun is

signal sign : std\_logic\_vector(4 downto 0);

begin

process(visit0,pcin,nw,nr,dbus)

begin

if(visit0 = '1')then

abus <= pcin;

irout <= dbus;

sign <= "10000";

elsif(nr = '1' and nw<= '0')then

abus <= addr;

datar <= dbus(7 downto 0);

sign <= "10000";

elsif(nw = '1' and nr <= '0') then

abus <= addr;

dbus <= dataw&dataw;

sign <= "01000";

else

dbus <= "ZZZZZZZZZZZZZZZZ";

abus <= "0000000000000000";

sign <= "11111";

end if;

end process;

nwr <= sign(4);

nrd <= sign(3);

nble <= sign(2);

nbhe <= sign(1);

nmreq <= sign(0);

end Behavioral;

波形：



1. 系统

例化代码

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity cpu is

port(

rst : in std\_logic;

clk : in std\_logic;

abus : out std\_logic\_vector(15 downto 0);

dbus : inout std\_logic\_vector(15 downto 0);

nmerq : out std\_logic;

nrd : out std\_logic;

nwr : out std\_logic;

nbhe : out std\_logic;

nble : out std\_logic;

ioad : out std\_logic\_vector(1 downto 0);

iodb : inout std\_logic\_vector(7 downto 0);

npreq : out std\_logic;

nprd : out std\_logic;

in\_data : in std\_logic\_vector(7 downto 0);

npwr : out std\_logic

);

end cpu;

architecture Behavioral of cpu is

component shizhong

Port ( clk : in STD\_LOGIC;

reset : in STD\_LOGIC;

t0 : out STD\_LOGIC;

t1 : out STD\_LOGIC;

t2 : out STD\_LOGIC;

t3 : out STD\_LOGIC);

end component;

component quzhi

Port ( t0 : in STD\_LOGIC;

t1,clk: in STD\_LOGIC;

pc\_data : in STD\_LOGIC\_VECTOR (15 downto 0);

pc\_updata : in STD\_LOGIC;

ir\_in: in STD\_LOGIC\_VECTOR (15 downto 0);

visit\_t0 : out STD\_LOGIC;

reset:in std\_logic;

ir : out STD\_LOGIC\_VECTOR (15 downto 0);

pc\_out : out STD\_LOGIC\_VECTOR (15 downto 0));

end component;

component yunsuan

port(

t1,clk : in std\_logic;

ir\_out1 : in std\_logic\_vector(15 downto 0);

r\_updata : in std\_logic;

r\_data : in std\_logic\_vector(7 downto 0);

data\_r : in std\_logic\_vector(7 downto 0);

alu\_out : out std\_logic\_vector(7 downto 0);

salu\_out : out std\_logic\_vector(7 downto 0);

addr\_out : out std\_logic\_vector(15 downto 0);

pc\_addr : out std\_logic\_vector(15 downto 0);

i\_oad : out std\_logic\_vector(1 downto 0);

i\_odb : inout std\_logic\_vector(7 downto 0);

in\_data : in std\_logic\_vector(7 downto 0);

w\_out :out std\_logic;

r\_out :out std\_logic;

p\_req : out std\_logic;

p\_wr : out std\_logic;

p\_rd : out std\_logic

);

end component;

component cunchu

port(

t2 : in std\_logic;

ir\_out2 : in std\_logic\_vector(15 downto 0);

data\_r : in std\_logic\_vector(7 downto 0);

r\_temp : out std\_logic\_vector(7 downto 0)

);

end component;

component huixue

port(

t3,reset : in std\_logic;

alu\_in : in std\_logic\_vector(7 downto 0);

alu\_in1 : in std\_logic\_vector(7 downto 0);

pc\_in1: in std\_logic\_vector(15 downto 0);

ir\_out3 : in std\_logic\_vector(15 downto 0);

pc\_addr : in std\_logic\_vector(15 downto 0);

r\_updata : out std\_logic;

r\_data : out std\_logic\_vector(7 downto 0);

pc\_updata : out std\_logic;

pc\_data : out std\_logic\_vector(15 downto 0)

);

end component;

component fangcun

port(

visit\_t0 : in std\_logic;

pc\_in :in std\_logic\_vector(15 downto 0);

addr\_in:in std\_logic\_vector(15 downto 0);

w\_in,r\_in : in std\_logic;

data\_w : in std\_logic\_vector(7 downto 0);

ir\_out :out std\_logic\_vector(15 downto 0);

data\_r : out std\_logic\_vector(7 downto 0);

a\_bus :out std\_logic\_vector(15 downto 0);

d\_bus :inout std\_logic\_vector(15 downto 0);

n\_wr,n\_rd,n\_ble,n\_bhe,n\_mreq : out std\_logic

);

end component;

signal s\_t0,s\_t1,s\_t2,s\_t3,visitpc,p\_reg,p\_wr,p\_rd: std\_logic;

signal ir0,irout,ir\_out1,ir\_out2,ir\_out3,ir\_in,pcin,pc\_out,pc\_new : std\_logic\_vector(15 downto 0);

signal addrin,pcdata,pcaddr: std\_logic\_vector(15 downto 0);

signal saluout,aluin,rtemp,rdata,datar,raluout,temp: std\_logic\_vector(7 downto 0);

signal reset,win,rin,pcupdata,rupdata: std\_logic;

begin

sz : shizhong port map(clk=>clk,reset=>rst,t0=>s\_t0,t1=>s\_t1,t2=>s\_t2,t3=>s\_t3);

qz : quzhi port map(t0=>s\_t0,t1=>s\_t1,clk=>clk,pc\_data=>pcdata, pc\_updata=>pcupdata ,reset=>rst,pc\_out=>pcin,ir\_in=>irout,visit\_t0=>visitpc,ir=>ir0);

ys : yunsuan port map(t1=>s\_t1,clk=>clk,ir\_out1=>ir0,r\_updata=>rupdata,r\_data=>rdata,data\_r=>datar,alu\_out=>aluin,w\_out=>win,r\_out=>rin,addr\_out=>addrin,p\_req=>npreq,p\_wr=>npwr,p\_rd=>nprd,salu\_out=>rtemp,pc\_addr=>pcaddr,i\_oad=>ioad,i\_odb=>iodb,in\_data=>in\_data);

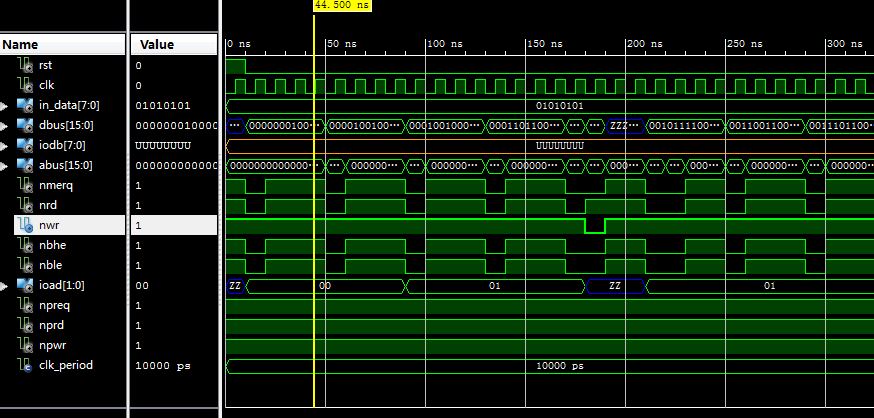
cc : cunchu port map(t2=>s\_t2,ir\_out2=>ir0,data\_r=>datar,r\_temp=>temp);

hx : huixue port map(t3=>s\_t3,reset=>rst,ir\_out3=>ir0,alu\_in=>aluin,r\_updata=>rupdata,r\_data=>rdata,pc\_updata=>pcupdata,pc\_data=>pcdata,pc\_in1=>pcin,pc\_addr=>pcaddr,alu\_in1=>temp);

fc :fangcun port map(visit\_t0=>visitpc,pc\_in=>pcin,addr\_in=>addrin,w\_in=>win,r\_in=>rin,data\_w=>rtemp,ir\_out=>irout,data\_r=>datar,a\_bus=>abus,d\_bus=>dbus,n\_bhe=>nbhe,n\_ble=>nble,n\_mreq=>nmerq,n\_rd=>nrd,n\_wr=>nwr);

end Behavioral;

波形：



1. 设计、调试、波形、下载过程中遇到的问题及解决方法。
2. 开始设计各个模块时，各个模块的连线非常多，也非常杂。在经过老师指导后，各个模块完成的工作非常明确，使得模块间的信号控制线一下子减少了很多。使我在写各个模块代码是比较轻松。
3. 在写alu模块时，每个when语句后我都有一大堆语句，写的很复杂，不好理解。在同学的帮助后，我非常明确的哪个指令该干什么，什么不需要干，这样缩减了一大段代码，有利于后期的优化。同样在存储管理模块，我一开始把十条判断语句都写上去了，之后与同学对比发现不需要这样麻烦，因为设计时存储管理只需要对访存指令负责，不需要一个一个判断其他指令，这样简化了很多的代码。
4. 在跑波形的时候，我经常性的会忽视信号传入的先后顺序，导致生成波形时经常性的与事实不符。解决办法就是一次一次的改动测试文件，直到最后结果正确。
5. 在下载过程中，试了很多次，因为VHDL代码虽然跑波形的时候没有错，但是在实际下载的时候出现了一大堆的错误。所以经过很长时间一次一次的改动最终得以实现下载功能。