FUNCTIONALITY

**Stackp( stack pointer)**

The stack pointer points to the stack’s 8 levels. When it encounters an int i.e. an interrupt and when the PCctrl is PC push than in that case it increments the stack counter by 1. In case when the PCctrl is PCpop in that case it decreases the stack counter by 1.The generated address is than assigned to the file address register in both of these cases.

**File address generation block**

Depending upon the last 7 bits of the instruction it determines the type of addressing either direct or indirect addressing. Also when the PCctrl is pcpush or pcpop than in that case it would load the stack pointer into the fadd(lower 3 bits) .Also wFSR flag can be used to load the FSR register with the contents of the result. And in indirect addressing mode the fadd is loaded from the FSR register.

**ADD Decoder**

Depending on the address of the fadd Address decoder would compare it with the address of FSR, PCLATH, STAT, and PCL SFR’s. Now depending on the fread and fwrite signals it would either read from or write to these particular registers. And the output of this entity goes to the above mentioned registers in the form of wPCL, wPCLATH, wSTAT and wFSR or rPCL,rPCLATH,rSTAT and rFSR depending on fread or fwrite signal.

**CPU PC register**

This mode of the CPU generates the program counter. Depending on the value of PCctrl it either increments it, loads the program counter, writes to the program counter when wPCL is set, pushes or pops the program counter etc. When an interrupt request signal is set than the program counter is loaded with count “0x04”. If skip is encountered than it would skip the next instruction. Also the upper bits of the program counter are loaded from the PCLATH in case with the conditional branch instruction.

We can write to the PCLATH register when wPCLATH is set and the content of the result is loaded from the w register of the ALU.

**STATUS register**

The STATUS register contains the arithmetic status of the ALU, the RESET status and the bank select bits for data memory. The STATUS register can be the destination for any instruction, as with any other register. If the STATUS register is the destination for an instruction that affects the Z, DC or C bits, then the write to these three bits is disabled. These bits are set or cleared according to the device logic. Furthermore, the TO and PD bits are not writable.

It consists of the bank select bits, register bank select bit, Z, DC or C bits, power down or time out bit and register bank select bits.

**REGDATA block**

The output of this register consists of the contents of FSR, status register, PCL, PCLATH etc. when the corresponding flags rFSR, rSTAT, rPCL, rPCLATH are set.

**VHDL CODE**

-- VHDL Entity CPU.CPU\_ADDR\_STRUT.symbol

-- by - drpatel20.UNKNOWN (NH-A4072-01)

-- at - 14:54:20 12/ 9/2013

-- Generated by Mentor Graphics' HDL Designer(TM) 2008.1 (Build 17)

LIBRARY ieee;

USE ieee.std\_logic\_1164.all;

USE ieee.std\_logic\_arith.all;

LIBRARY CPU;

USE CPU.CPUfunc.all;

USE ieee.std\_logic\_signed.all;

ENTITY CPU\_ADDR\_STRUT IS

PORT(

PCctrl : IN PC\_FN;

clrwdt : IN std\_logic;

fread : IN std\_logic;

fwrite : IN std\_logic;

iclk : IN std\_logic;

instr : IN std\_logic\_vector (13 DOWNTO 0);

int : IN std\_logic;

irq : IN std\_logic;

phi : IN std\_logic\_vector (3 DOWNTO 0);

por : IN std\_logic;

reset : IN std\_logic;

result : IN std\_logic\_vector (7 DOWNTO 0);

skip : IN std\_logic;

sleep : IN std\_logic;

status : IN std\_logic\_vector (2 DOWNTO 0);

upSTAT : IN std\_logic;

wdgTO : IN std\_logic;

rdata : OUT std\_logic\_vector (7 DOWNTO 0) := B"0000\_0000";

fadd : INOUT std\_logic\_vector (7 DOWNTO 0) := B"0000\_0000";

pc : INOUT std\_logic\_vector (12 DOWNTO 0);

sdata : INOUT std\_logic\_vector (12 DOWNTO 0);

PCLadd : BUFFER std\_logic;

wSTAT : BUFFER std\_logic

);

-- Declarations

END CPU\_ADDR\_STRUT ;

-- VHDL Architecture CPU.CPU\_ADDR\_STRUT.struct

-- Created:

-- by - drpatel20.UNKNOWN (NH-A4072-01)

-- Generated by Mentor Graphics' HDL Designer(TM) 2008.1 (Build 17)

LIBRARY ieee;

USE ieee.std\_logic\_1164.all;

USE ieee.std\_logic\_arith.all;

USE ieee.numeric\_bit.all;

USE ieee.std\_logic\_signed.all;

LIBRARY CPU;

USE CPU.CPUfunc.all;

LIBRARY PIC16C63A\_lib;

USE PIC16C63A\_lib.address.all;

LIBRARY CPU;

ARCHITECTURE struct OF CPU\_ADDR\_STRUT IS

-- Architecture declarations

-- Internal signal declarations

SIGNAL FSR : std\_logic\_vector(7 DOWNTO 0) := B"1111\_1010";

SIGNAL PCL : std\_logic\_vector(7 DOWNTO 0);

SIGNAL PCLATH : std\_logic\_vector(7 DOWNTO 0) := B"1111\_0000";

SIGNAL rFSR : std\_logic;

SIGNAL rPCL : std\_logic;

SIGNAL rPCLATH : std\_logic;

SIGNAL rSTAT : std\_logic;

SIGNAL stackp : std\_logic\_vector(2 DOWNTO 0);

SIGNAL stat : std\_logic\_vector(7 DOWNTO 0);

SIGNAL wFSR : std\_logic;

SIGNAL wPCL : std\_logic;

SIGNAL wPCLATH : std\_logic;

-- Component Declarations

COMPONENT PCReg

PORT (

PCctrl : IN PC\_FN ;

iclk : IN std\_logic ;

instr : IN std\_logic\_vector (13 DOWNTO 0);

int : IN std\_logic ;

irq : IN std\_logic ;

phi : IN std\_logic\_vector (3 DOWNTO 0);

reset : IN std\_logic ;

result : IN std\_logic\_vector (7 DOWNTO 0);

skip : IN std\_logic ;

wPCL : IN std\_logic ;

wPCLATH : IN std\_logic ;

PCLATH : OUT std\_logic\_vector (7 DOWNTO 0) := B"1111\_0000";

PCL : INOUT std\_logic\_vector (7 DOWNTO 0) := B"0000\_0000";

pc : INOUT std\_logic\_vector (12 DOWNTO 0) := B"0000\_0000\_0000\_0";

sdata : INOUT std\_logic\_vector (12 DOWNTO 0) := B"0000\_0000\_0000\_0"

);

END COMPONENT;

-- Optional embedded configurations

-- pragma synthesis\_off

FOR ALL : PCReg USE ENTITY CPU.PCReg;

-- pragma synthesis\_on

BEGIN

-- Architecture concurrent statements

-- HDL Embedded Text Block 1 stack

-- stack 1

P0:PROCESS (phi(3),reset,int,PCctrl,phi(0),iclk)

VARIABLE C: STD\_LOGIC\_VECTOR(2 DOWNTO 0):="000";

begin

IF (reset = '0') THEN

stackp <= "000";

ASSERT FALSE

REPORT "stack first line"

SEVERITY Error;

END IF;

IF RISING\_EDGE(iclk) THEN

IF int='1' THEN

IF PCctrl=PCpush AND phi(3)='1' THEN

C:=C+1;

ASSERT FALSE

REPORT "stack second line"

SEVERITY Error;

IF C/="111" THEN

stackp(2 DOWNTO 0)<= C(2 DOWNTO 0);

ASSERT FALSE

REPORT "stack third line"

SEVERITY Error;

END IF;

END IF;

END IF;

END IF;

IF RISING\_EDGE(iclk) THEN

IF C ="111" THEN

stackp(2 DOWNTO 0)<= C(2 DOWNTO 0);

ASSERT FALSE

REPORT "stack fourth line"

SEVERITY Error;

END IF;

end if;

IF RISING\_EDGE(iclk) THEN

IF PCctrl=PCpop THEN

IF phi(0)='1' THEN

C:=C-1;

stackp(2 DOWNTO 0)<= C(2 DOWNTO 0);

ASSERT FALSE

REPORT "stack fifth line"

SEVERITY Error;

END IF;

end if;

END IF;

END PROCESS;

-- HDL Embedded Text Block 2 FileAddr

-- FileAddr 2

PROCESS(phi(3),PCctrl,wFSR,reset) IS

variable F\_ADD :std\_logic\_vector(7 downto 0);

BEGIN

IF instr(6 downto 0) /= "0000000" THEN

IF rising\_edge(phi(3)) THEN

F\_ADD(6 downto 0):= instr(6 downto 0);

F\_ADD(7) := stat(5);

ASSERT FALSE

REPORT "fadd first line"

SEVERITY Error;

END IF;

END IF;

IF instr(6 downto 0)= "0000000" THEN

IF rising\_edge(phi(3)) THEN

F\_ADD(6 downto 0) := FSR(6 downto 0);

F\_ADD(7) := stat(7);

ASSERT FALSE

REPORT "fadd second line"

SEVERITY Error;

END IF;

END IF;

IF wFSR='1' THEN

IF rising\_edge(phi(3)) THEN

FSR(7 downto 0) <= result(7 downto 0);

ASSERT FALSE

REPORT "fadd third line"

SEVERITY Error;

END IF;

END IF;

IF reset='0' THEN

F\_ADD(7 DOWNTO 0):=B"0000\_0000";

ASSERT FALSE

REPORT "fadd reset line"

SEVERITY Error;

END IF;

IF PCctrl=PCpop THEN

IF rising\_edge(phi(3)) THEN

F\_ADD(2 downto 0) := stackp(2 downto 0);

ASSERT FALSE

REPORT "fadd fourth line"

SEVERITY Error;

END IF;

END IF;

IF PCctrl=PCpush THEN

IF rising\_edge(phi(3)) THEN

F\_ADD(2 downto 0) := stackp(2 downto 0);

ASSERT FALSE

REPORT "fadd fifth line"

SEVERITY Error;

END IF;

END IF;

fadd<=F\_ADD;

END PROCESS;

-- HDL Embedded Text Block 3 AddDecode1

-- AddDecode1 3

PROCESS(fread,fwrite,PCctrl,phi(3),reset) is

BEGIN

IF reset='0' THEN

rFSR<='0';

rSTAT<='0';

rPCL<='0';

rPCLATH<='0';

wFSR<='0';

wSTAT<='0';

wPCL<='0';

wPCLATH<='0';

ASSERT FALSE

REPORT "Add decode first line"

SEVERITY Error;

END IF;

IF fread='1' THEN

IF rising\_edge(phi(3)) THEN

CASE fadd(6 downto 0) IS

WHEN aFSR => rFSR<='1';

WHEN aSTATUS => rSTAT<='1';

WHEN aPCL => rPCL<='1';

WHEN aPCLATH => rPCLATH<='1';

when others => rFSR<='0';rSTAT<='0';rPCL<='0';rPCLATH<='0';

ASSERT FALSE

REPORT "add decode second line"

SEVERITY Error;

END CASE;

END IF;

END IF;

IF fwrite='1' THEN

IF rising\_edge(phi(3)) THEN

CASE fadd(6 downto 0) IS

WHEN aFSR => wFSR<='1';

WHEN aSTATUS => wSTAT<='1';

WHEN aPCL => wPCL<='1';

WHEN aPCLATH => wPCLATH<='1';

when others => wFSR<='0';wSTAT<='0';wPCL<='0';wPCLATH<='0';

ASSERT FALSE

REPORT "ADD decode third line"

SEVERITY Error;

END CASE;

END IF;

END IF;

IF PCctrl=PCpush THEN

rFSR<=rFSR;

rSTAT<=rSTAT;

rPCL<=rPCL;

rPCLATH<=rPCLATH;

wFSR<=wFSR;

wSTAT<=wSTAT;

wPCL<=wPCL;

wPCLATH<=wPCLATH;

END IF;

IF PCctrl=PCpop THEN

rFSR<=rFSR;

rSTAT<=rSTAT;

rPCL<=rPCL;

rPCLATH<=rPCLATH;

wFSR<=wFSR;

wSTAT<=wSTAT;

wPCL<=wPCL;

wPCLATH<=wPCLATH;

END IF;

IF rising\_edge(phi(3)) THEN

IF PCctrl=PCload THEN

PCLadd<='1';

ELSE IF PCctrl=PCwrite THEN

PCLadd<='1';

END IF;

END IF;

END IF;

END PROCESS;

-- HDL Embedded Text Block 4 StatusReg

-- StatusReg 4

PROCESS (por,clrwdt,wdgTO,sleep,status,wSTAT,phi(3),reset,upSTAT)

BEGIN

if reset='0' then

stat(7 downto 0)<=B"0000\_0000";

end if;

IF upSTAT='1' THEN

IF RISING\_EDGE(phi(3)) THEN

stat(2 DOWNTO 0) <= status(2 DOWNTO 0);

END IF;

END IF;

IF wSTAT='1' THEN

IF RISING\_EDGE(phi(3)) THEN

stat(7 DOWNTO 5) <= result(7 DOWNTO 5) ;

END IF;

END IF;

IF clrwdt='1' OR por ='1' THEN

IF rising\_edge(phi(3)) THEN

stat(3)<='1';

IF sleep='1' THEN

IF rising\_edge(phi(3)) THEN

stat(3)<='0';

END IF;

END IF;

END IF;

END IF;

IF por='1' THEN

IF clrwdt='1' OR sleep ='1' THEN

IF rising\_edge(phi(3)) THEN

stat(4)<='1';

IF wdgTO='1' THEN

IF rising\_edge(phi(3)) THEN

stat(4)<='0';

END IF;

END IF;

END IF;

END IF;

END IF;

END PROCESS;

-- HDL Embedded Text Block 5 RegData

-- RegData 5

PROCESS (rPCL,rSTAT,rFSR,rPCLATH,phi(3),reset)

BEGIN

IF rPCL='1' THEN

IF rising\_edge(phi(3)) THEN

rdata(7 DOWNTO 0) <= pc(7 DOWNTO 0);

ASSERT FALSE

REPORT "rdata rpcl line"

SEVERITY Error;

END IF;

END IF;

IF rSTAT='1' THEN

IF rising\_edge(phi(3)) THEN

rdata(7 DOWNTO 0) <= stat(7 DOWNTO 0);

ASSERT FALSE

REPORT "rdata rstat line"

SEVERITY Error;

END IF;

END IF;

IF rFSR='1' THEN

IF rising\_edge(phi(3)) THEN

rdata(7 DOWNTO 0) <= FSR(7 DOWNTO 0);

ASSERT FALSE

REPORT "rdata fsr line"

SEVERITY Error;

END IF;

END IF;

IF rPCLATH='1' THEN

IF rising\_edge(phi(3)) THEN

rdata(7 DOWNTO 0) <= PCLATH(7 DOWNTO 0);

ASSERT FALSE

REPORT "rdata PCLATH line"

SEVERITY Error;

END IF;

END IF;

IF reset='0' THEN

rdata(7 DOWNTO 0)<="00000000";

ASSERT FALSE

REPORT "rdata reset line"

SEVERITY Error;

END IF;

END PROCESS;

-- Instance port mappings.

U\_0 : PCReg

PORT MAP (

PCctrl => PCctrl,

iclk => iclk,

instr => instr,

int => int,

irq => irq,

phi => phi,

reset => reset,

result => result,

skip => skip,

wPCL => wPCL,

wPCLATH => wPCLATH,

PCLATH => PCLATH,

PCL => PCL,

pc => pc,

sdata => sdata

);

END struct;

-- VHDL Entity CPU.PCReg.interface

-- Created:

-- by - drpatel20.UNKNOWN (NH-A4072-01)

-- at - 14:54:20 12/ 9/2013

--

-- Generated by Mentor Graphics' HDL Designer(TM) 2008.1 (Build 17)

--

LIBRARY ieee;

USE ieee.std\_logic\_1164.all;

USE ieee.std\_logic\_arith.all;

USE ieee.numeric\_bit.all;

USE ieee.std\_logic\_signed.all;

LIBRARY CPU;

USE CPU.CPUfunc.all;

LIBRARY PIC16C63A\_lib;

USE PIC16C63A\_lib.address.all;

ENTITY PCReg IS

PORT(

PCctrl : IN PC\_FN;

iclk : IN std\_logic;

instr : IN std\_logic\_vector (13 DOWNTO 0);

int : IN std\_logic;

irq : IN std\_logic;

phi : IN std\_logic\_vector (3 DOWNTO 0);

reset : IN std\_logic;

result : IN std\_logic\_vector (7 DOWNTO 0);

skip : IN std\_logic;

wPCL : IN std\_logic;

wPCLATH : IN std\_logic;

PCLATH : OUT std\_logic\_vector (7 DOWNTO 0) := B"1111\_0000";

PCL : INOUT std\_logic\_vector (7 DOWNTO 0) := B"0000\_0000";

pc : INOUT std\_logic\_vector (12 DOWNTO 0) := B"0000\_0000\_0000\_0";

sdata : INOUT std\_logic\_vector (12 DOWNTO 0) := B"0000\_0000\_0000\_0"

);

-- Declarations

END PCReg ;

-- VHDL Architecture CPU.PCReg.struct

--

-- Created:

-- by - drpatel20.UNKNOWN (NH-A4072-01)

LIBRARY ieee;

USE ieee.std\_logic\_1164.all;

USE ieee.std\_logic\_arith.all;

LIBRARY CPU;

USE CPU.CPUfunc.all;

USE ieee.std\_logic\_signed.all;

LIBRARY PIC16C63A\_lib;

USE PIC16C63A\_lib.address.all;

ARCHITECTURE struct OF PCReg IS

-- Architecture declarations

-- Internal signal declarations

-- Implicit buffer signal declarations

SIGNAL PCLATH\_internal : std\_logic\_vector (7 DOWNTO 0);

BEGIN

-- Architecture concurrent statements

-- HDL Embedded Text Block 1 PCLATH

-- PCLATH\_internal 1

PROCESS (wPCLATH,phi(3),reset) IS

BEGIN

IF reset='0' then

PCLATH\_internal (7 downto 0)<= B"0000\_0000";

ASSERT FALSE

REPORT "PCLATH\_internal 1st line"

SEVERITY Error;

END IF;

IF wPCLATH ='1' THEN

IF rising\_edge(phi(3)) THEN

PCLATH\_internal(7 downto 0) <= result(7 downto 0);

ASSERT FALSE

REPORT "PCLATH\_internal 2nd line"

SEVERITY Error;

END IF;

END IF;

END PROCESS;

-- HDL Embedded Text Block 2 PC

-- PC 2

PROCESS(PCctrl,int,irq,wPCL,reset,skip,phi(3),iclk,phi(1)) IS

VARIABLE A: STD\_LOGIC\_VECTOR(7 DOWNTO 0) :=B"0000\_0000";

BEGIN

IF PCctrl=PCpush THEN

IF int='1' THEN

IF rising\_edge(phi(3)) AND iclk='1' THEN

sdata <= pc ;

ASSERT FALSE

REPORT "PC first line"

SEVERITY Error;

END IF;

END IF;

END IF;

IF PCctrl=PCpop THEN

IF rising\_edge(phi(1)) AND iclk='1' THEN

pc(12 DOWNTO 0) <= sdata(12 DOWNTO 0);

PCL(7 DOWNTO 0) <= pc(7 DOWNTO 0);

PCLATH\_internal(4 downto 0) <= pc(12 downto 8);

ASSERT FALSE

REPORT "PC 2th line"

SEVERITY Error;

END IF;

END IF;

IF PCctrl=PCload THEN

IF rising\_edge(phi(3)) THEN

pc(10 DOWNTO 0)<= instr(10 DOWNTO 0);

PCL(7 DOWNTO 0) <= pc(7 DOWNTO 0);

pc(12 DOWNTO 11) <= PCLATH\_internal( 4 DOWNTO 3);

ASSERT FALSE

REPORT "PC 3th line"

SEVERITY Error;

END IF;

END IF;

IF PCctrl=PCwrite THEN

IF wPCL='1' THEN

IF rising\_edge(phi(3)) THEN

PCL(7 downto 0) <= result(7 downto 0) ;

pc(7 DOWNTO 0) <= PCL(7 DOWNTO 0);

pc(12 DOWNTO 8)<=PCLATH\_internal(4 DOWNTO 0);

ASSERT FALSE

REPORT "PC 4th line"

SEVERITY Error;

END IF;

END IF;

END IF;

IF irq='1' THEN

PCL <=X"04";

pc(7 DOWNTO 0) <= PCL(7 DOWNTO 0);

ASSERT FALSE

REPORT "PC 5th line"

SEVERITY Error;

END IF;

IF reset='0' THEN

PCL <= X"00";

PCLATH\_internal <= X"00";

pc<=B"0000\_0000\_00000";

sdata <= pc ;

ASSERT FALSE

REPORT "PC 6th line"

SEVERITY Error;

END IF ;

IF skip='1' THEN

IF rising\_edge(phi(3)) THEN

IF instr="00000000000000" THEN

A(7 downto 0):=PCL(7 downto 0);

A:=A+1;

PCL(7 downto 0)<=A(7 downto 0);

pc(7 DOWNTO 0) <= PCL(7 DOWNTO 0);

ASSERT FALSE

REPORT "PC 7th line"

SEVERITY Error;

END IF;

END IF;

end if;

IF rising\_edge(phi(3)) THEN

IF PCctrl=PCnothing THEN

IF instr="00000000000000" THEN

PCL(7 downto 0)<=PCL(7 downto 0);

else IF instr="00000000000000" THEN

A(7 downto 0):=PCL(7 downto 0);

A:=A+1;

PCL(7 downto 0)<=A(7 downto 0);

pc(7 DOWNTO 0) <= PCL(7 DOWNTO 0);

ASSERT FALSE

REPORT "PC 8th line"

SEVERITY Error;

end if;

end if;

end if;

end if;

IF PCctrl=PCinc THEN

IF rising\_edge(phi(3)) THEN

A(7 downto 0):=PCL(7 downto 0);

A:=A+1;

PCL(7 downto 0)<=A(7 downto 0);

pc(7 DOWNTO 0) <= PCL(7 DOWNTO 0);

ASSERT FALSE

REPORT "PC 9th line"

SEVERITY Error;

END IF;

END IF;

END PROCESS;

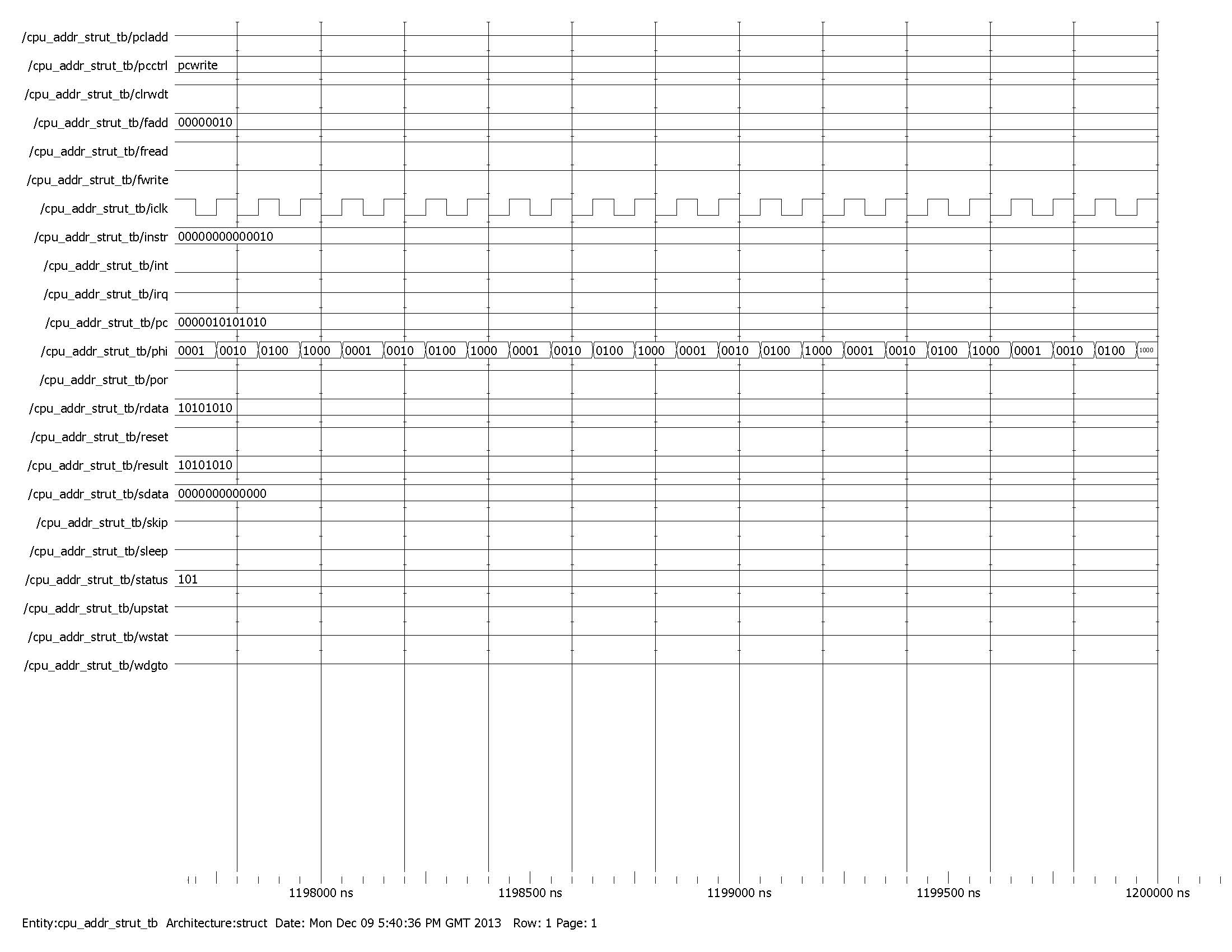
-- Instance port mappings.

-- Implicit buffered output assignments

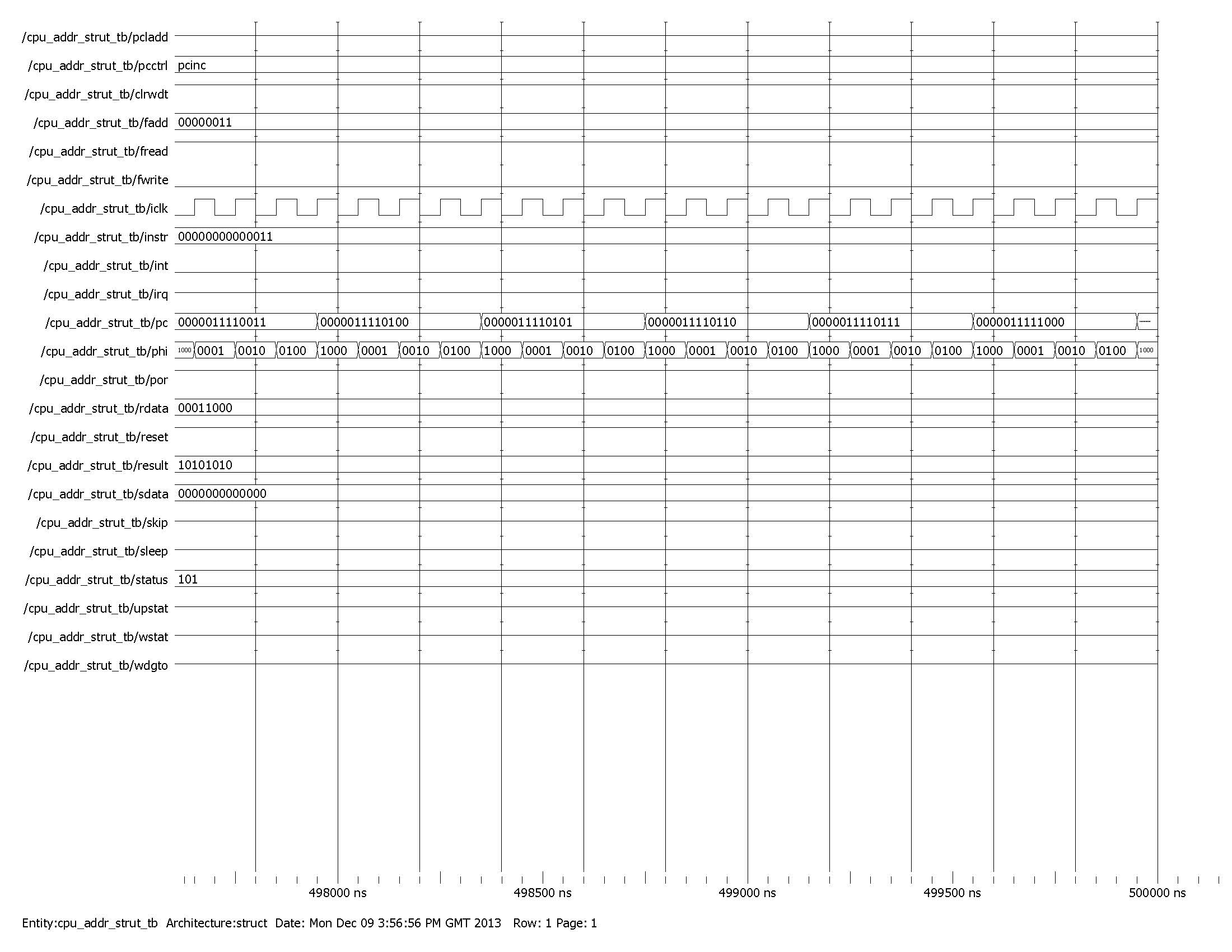
PCLATH <= PCLATH\_internal;

END struct;

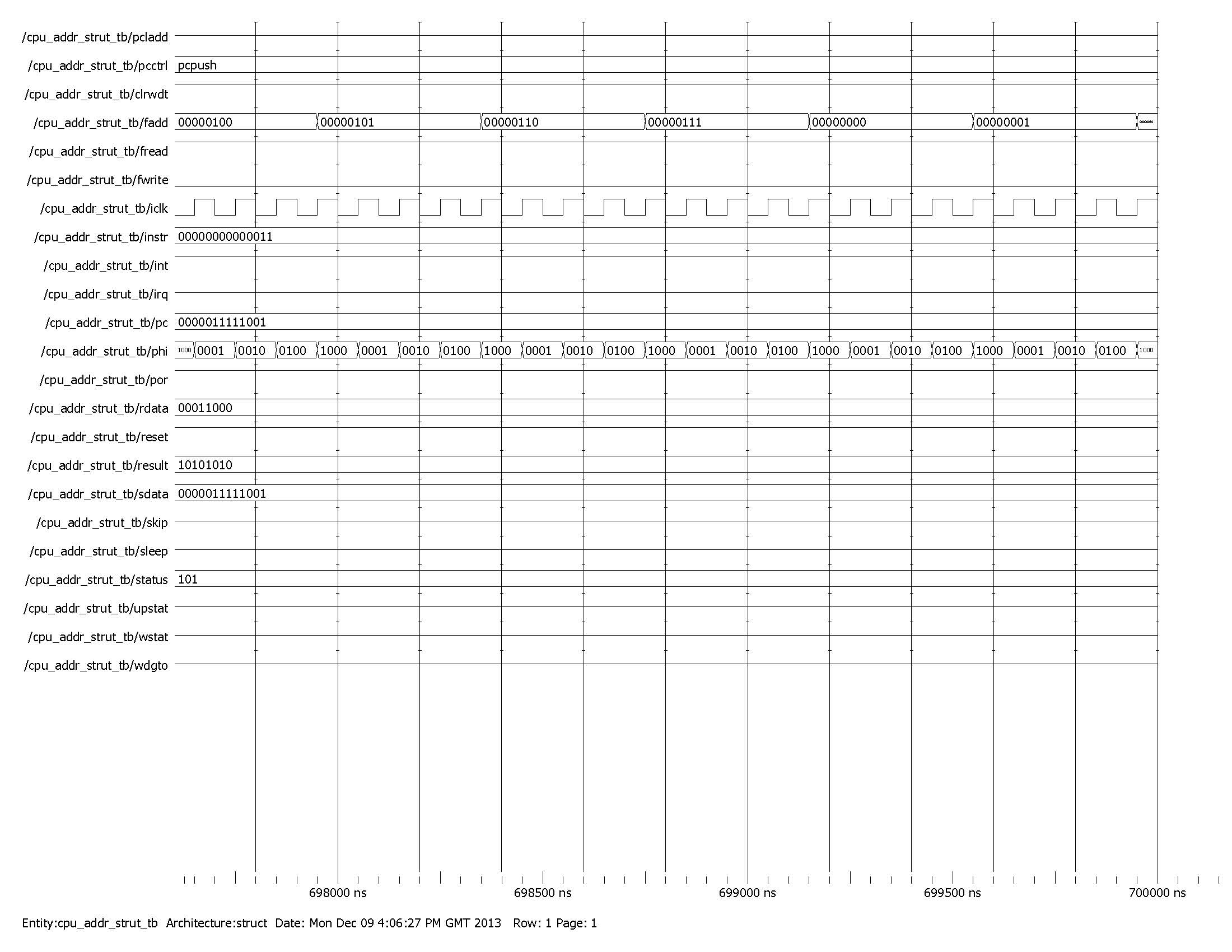




TEST WAVEFORM for PCctrl=PCwrite the result is loaded into pc from the ALU.



TEST WAVEFORM for PCctrl=PCinc



For PCctrl=PCPUSH the value of the program counter is loaded into the stack and at the same time the value of the fadd represents the stack pointer’s lower 3 bits.

TESTBENCH 

CODE of TEST BENCH

-- VHDL Entity CPU.CPU\_ADDR\_STRUT\_tester.interface

--

-- Created:

-- by - drpatel20.UNKNOWN (NH-A4072-01)

-- at - 16:24:20 12/ 9/2013

--

-- Generated by Mentor Graphics' HDL Designer(TM) 2008.1 (Build 17)

--

LIBRARY ieee;

USE ieee.std\_logic\_1164.ALL;

USE ieee.std\_logic\_arith.ALL;

LIBRARY CPU;

USE CPU.CPUfunc.ALL;

USE ieee.std\_logic\_signed.ALL;

LIBRARY PIC16C63A\_lib;

USE PIC16C63A\_lib.address.all;

ENTITY CPU\_ADDR\_STRUT\_tester IS

PORT(

fadd : IN std\_logic\_vector (7 DOWNTO 0) := B"0000\_0000";

pc : IN std\_logic\_vector (12 DOWNTO 0);

rdata : IN std\_logic\_vector (7 DOWNTO 0) := B"0000\_0000";

sdata : IN std\_logic\_vector (12 DOWNTO 0) := B"0000\_0000\_00000";

PCctrl : OUT PC\_FN;

clrwdt : OUT std\_logic;

fread : OUT std\_logic;

fwrite : OUT std\_logic;

iclk : OUT std\_logic;

instr : OUT std\_logic\_vector (13 DOWNTO 0);

int : OUT std\_logic;

irq : OUT std\_logic;

phi : OUT std\_logic\_vector (3 DOWNTO 0);

por : OUT std\_logic;

reset : OUT std\_logic;

result : OUT std\_logic\_vector (7 DOWNTO 0);

skip : OUT std\_logic;

sleep : OUT std\_logic;

status : OUT std\_logic\_vector (2 DOWNTO 0);

upSTAT : OUT std\_logic;

wSTAT : OUT std\_logic;

wdgTO : OUT std\_logic;

PCLadd : BUFFER std\_logic

);

-- Declarations

END CPU\_ADDR\_STRUT\_tester ;

-- VHDL Architecture CPU.CPU\_ADDR\_STRUT\_tester.flow

-- by - drpatel20.UNKNOWN (NH-A4072-01)

LIBRARY ieee;

USE ieee.std\_logic\_1164.ALL;

USE ieee.std\_logic\_arith.ALL;

LIBRARY CPU;

USE CPU.CPUfunc.ALL;

USE ieee.std\_logic\_signed.ALL;

LIBRARY PIC16C63A\_lib;

USE PIC16C63A\_lib.address.all;USE PIC16C63A\_lib.tester.all;

ARCHITECTURE flow OF CPU\_ADDR\_STRUT\_tester IS

-- Architecture declarations

SIGNAL i\_clk: std\_logic:='0';

SIGNAL phase: std\_logic\_vector(3 downto 0) :="0000";

SIGNAL rst:std\_logic:='1';

SIGNAL iprog : std\_logic\_vector(12 downto 0) := B"0\_0000\_0000\_0000";

BEGIN

-----------------------------------------------------------------

process0\_proc : PROCESS (fadd, i\_clk, pc, rdata)

-----------------------------------------------------------------

BEGIN

PCctrl<=PCwrite;

int<='0';

iclk<=i\_clk;

instr<=B"00\_0000\_0000\_0011";

if fadd=01101010 then

ASSERT FALSE

REPORT "file address error"

SEVERITY Error ;

else

ASSERT TRUE;

END IF;

fread<='1';

fwrite<='0';

result<="00011110";

if pc=1111111111 then

ASSERT FALSE

REPORT "PC error"

SEVERITY Error;

else

ASSERT true;

end if;

result<="10101010";

status<="101";

por<='1';

clrwdt<='1';

if rdata=10110101 then

ASSERT FALSE

REPORT "rdata error"

SEVERITY Error;

else

ASSERT true;

end if;

END PROCESS process0\_proc;

-- Architecture concurrent statements

i\_clk<= NOT i\_clk AFTER 50ns;

phi<=phase;

reset<=rst;

gen\_clock(i\_clk,rst,phase);

iprog <= pc;

END flow;