library IEEE;

use IEEE.std\_logic\_1164.all;

use IEEE.std\_logic\_unsigned.all;

entity reg2 is

port( sel\_rlsmux\_in: in std\_logic;

sel\_rlsmux\_out: out std\_logic;

opcode\_in:in std\_logic\_vector(3 downto 0);

opcode\_out:out std\_logic\_vector(3 downto 0);

sel\_in: in std\_logic\_vector(3 downto 0);

sel\_out: out std\_logic\_vector(3 downto 0);

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

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

rst\_in: in std\_logic;

rst\_out: out std\_logic;

rdx\_in: in std\_logic\_vector(3 downto 0);

rdx\_out: out std\_logic\_vector(3 downto 0);

rdx\_en\_in: in std\_logic;

rdx\_en\_out: out std\_logic;

rdy\_in: in std\_logic\_vector(3 downto 0);

rdy\_out: out std\_logic\_vector( 3 downto 0);

rdy\_en\_in: in std\_logic;

rdy\_en\_out: out std\_logic;

wr\_en\_in: in std\_logic;

wr\_en\_out: out std\_logic;

wrd\_in: in std\_logic\_vector(3 downto 0);

wrd\_out: out std\_logic\_vector(3 downto 0);

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

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

sel\_rymux\_in: in std\_logic;

sel\_rymux\_out: out std\_logic;

sel\_wbmux\_in: in std\_logic;

sel\_wbmux\_out: out std\_logic;

clk: in std\_logic

);

end entity;

architecture main of reg2 is

begin

reg\_alu: process(clk,sel\_rlsmux\_in,opcode\_in,sel\_in,rl\_addr\_in,rst\_in,rdx\_in, rdx\_en\_in,rdy\_in,rdy\_en\_in,wr\_en\_in,wrd\_in,imm\_val\_in,sel\_rymux\_in,sel\_wbmux\_in)

begin

--, sel\_rlsmux\_in,opcode\_in,sel\_in,rl\_addr\_in,rst\_in,rdx\_in, rdx\_en\_in,rdy\_in,rdy\_en\_in,wr\_en\_in,wrd\_in,imm\_val\_in,sel\_rymux\_in,sel\_wbmux\_in

if (clk'event and clk = '1') then

opcode\_out <= opcode\_in;

end if;

if (clk'event and clk = '1') then

sel\_out <= sel\_in;

end if;

if (clk'event and clk = '1') then

rl\_addr\_out <= rl\_addr\_in;

end if;

if (clk'event and clk = '1') then

rst\_out <= rst\_in;

end if;

if (clk'event and clk = '1') then

rdx\_out <= rdx\_in;

end if;

if (clk'event and clk = '1') then

rdx\_en\_out <= rdx\_en\_in;

end if;

if (clk'event and clk = '1') then

rdy\_out <= rdy\_in;

end if;

if (clk'event and clk = '1') then

rdy\_en\_out <= rdy\_en\_in;

end if;

if (clk'event and clk = '1') then

wr\_en\_out <= wr\_en\_in;

end if;

if (clk'event and clk = '1') then

wrd\_out <= wrd\_in;

end if;

if (clk'event and clk = '1') then

imm\_val\_out <= imm\_val\_in;

end if;

if (clk'event and clk = '1') then

sel\_rlsmux\_out <= sel\_rlsmux\_in;

end if;

if (clk'event and clk = '1') then

sel\_rymux\_out <= sel\_rymux\_in;

end if;

if (clk'event and clk = '1') then

sel\_wbmux\_out <= sel\_wbmux\_in;

end if;

end process;

--rdx\_en\_out <= rdx\_en\_in after 49 ps;

--

--rdy\_en\_out <= rdy\_en\_in after 49 ps;

--

--wr\_en\_out <= wr\_en\_in after 49 ps;

--

end main;