主程式

library ieee;

use ieee.std\_logic\_1164.all;

use ieee.std\_logic\_unsigned.all;

entity ushift is

generic ( n: integer: =8);

port ( clk: in std\_logic;

clear: in std\_logic;

load: in std\_logic;

lr\_sel: in std\_logic;

di: in std\_logic\_vector(n-1 downto 0);

sdi: in std\_logic;

qo: buffer std\_logic\_vector(n-1 downto 0));

end ushift;

architecture behavior of ushift is

begin

with clk , clear, load select

qo <= di when '001' ,

sdi when '000' ;

end behavior ;

8bit暫存器

library ieee ;

use ieee.std\_logic\_1164.all;

entity shiftn is

generic(n: integer:= 8);

port( r :in syd\_logic\_vector(n-1 downto 0);

clock: in std\_logic ;

L,w: in std\_logic ;

Q: buffer std\_logic\_vector(n-1 downto 0));

end shiftn ;

architecture behavior of shiftn is

begin

process

begin

wait until clock's event and clock ='1';

if L = '1' then

Q <= r ;

else

Genbits: for i in 0 to n-2 loop

Q(i) <= Q(i+1);

end loop ;

Q(n-1) <= w ;

end if ;

end process ;

end behavior ;

library ieee;

use ieee.std\_logic\_1164.all;

--use work.seven\_segment\_display\_package.all;

entity SHIFT is

generic ( n: integer:=8);

port(

CLK,C0 :in std\_logic;

MD :in std\_logic\_vector(2 downto 0);

D :in std\_logic\_vector(n-1 downto 0);

QB :buffer std\_logic\_vector(n-1 downto 0);

CN :out std\_logic);

end entity SHIFT;

architecture BHV of SHIFT is

signal REG : std\_logic\_vector(n-1 downto 0);

signal CY : std\_logic;

begin

process(CLK,MD,C0)

begin

--if c0='1' then QB(7 downto 0)<= (others => '0');

--end if ;

if CLK'EVENT AND CLK='1' then

case MD is

when "000" =>REG(7 downto 0)=>(others => '0') ;

when "001" =>REG(0)<=C0;

for REG(i) in 7 downto 1 loop

REG(i) <= REG(i-1) ;

CY<=REG(7);

end loop;

--REG(7 downto 1)<=REG(6 downto 0); CY<=REG(7);

when "010" =>REG(7)<=C0;

REG(6 downto 0)<=REG(7 downto 1); CY<=REG(7);

when "011" =>REG(0)<=REG(7);

REG(7 downto 1)<=REG(6 downto 0); CY<=REG(7);

when "100" =>REG(7)<=REG(0);

REG(6 downto 0)<=REG(7 downto 1); CY<=REG(7);

when "101" =>REG(7 downto 0)<= D(7 downto 0);

when others =>REG<=REG; CY<=CY;

end case;

end if;

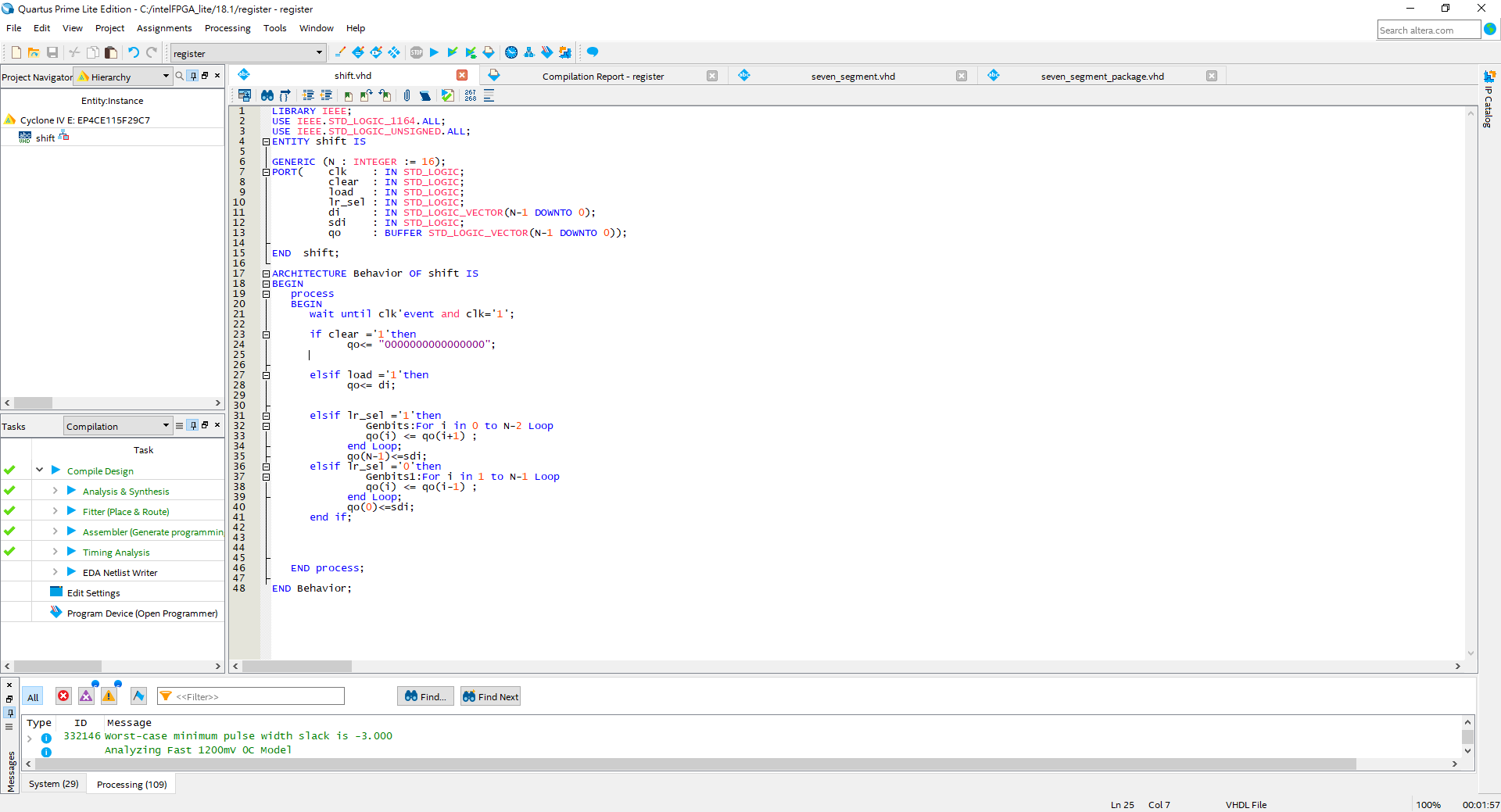
end process;

QB(7 downto 0)<=REG(7 downto 0); CN<=CY;

--output: seven\_segment\_display port map('0', result(6), result(5), CN,QB(7),QB(7 downto 0), QB(7 downto 0),QB(7 downto 0), QB(7 downto 0),QB(7 downto 0), QB(7 downto 0)) ;

end BHV;

報告:



LIBRARY IEEE;

USE IEEE.STD\_LOGIC\_1164.ALL;

USE IEEE.STD\_LOGIC\_UNSIGNED.ALL;

ENTITY shift IS

GENERIC (N : INTEGER := 16);

PORT( clk : IN STD\_LOGIC;

clear : IN STD\_LOGIC;

load : IN STD\_LOGIC;

lr\_sel : IN STD\_LOGIC;

di : IN STD\_LOGIC\_VECTOR(N-1 DOWNTO 0);

sdi : IN STD\_LOGIC;

qo : BUFFER STD\_LOGIC\_VECTOR(N-1 DOWNTO 0));

END shift;

ARCHITECTURE Behavior OF shift IS

BEGIN

process

BEGIN

wait until clk'event and clk='1';

if clear ='1'then

qo<= "0000000000000000";

elsif load ='1'then

qo<= di;

elsif lr\_sel ='1'then

Genbits:For i in 0 to N-2 Loop

qo(i) <= qo(i+1) ;

end Loop;

qo(N-1)<=sdi;

elsif lr\_sel ='0'then

Genbits1:For i in 1 to N-1 Loop

qo(i) <= qo(i-1) ;

end Loop;

qo(0)<=sdi;

end if;

END process;

END Behavior;

