# به نام خدا

تمرین چهارم

# محمدمهدی آقاجانی

9331056

استاد : دکتر صاحب الزمانی

سوال پنجم

کد این سوال به صورت زیر است :

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity chronometer is Port (

time\_in : in integer ;

reset : in std\_logic;

pause : in std\_logic;

resume : in std\_logic;

clk : in std\_logic;

timer : inout integer

);

end chronometer;

architecture Behavioral of chronometer is

type State\_type IS (start , counting);

signal state : state\_type ;

begin

process(clk)

begin

if( reset = '1' )then

state <= start ;

end if;

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

case state is

when start =>

timer <= time\_in;

if( resume = '1' )then

state <= counting ;

end if;

when counting =>

if( reset = '1' )then

state <= start;

end if;

if( resume = '1') then

if( pause = '1' )then

timer <= timer;

else

timer <= timer-1;

end if;

else

if( pause = '1' )then

timer <= timer;

else

timer <= timer-1;

end if;

end if;

when others =>

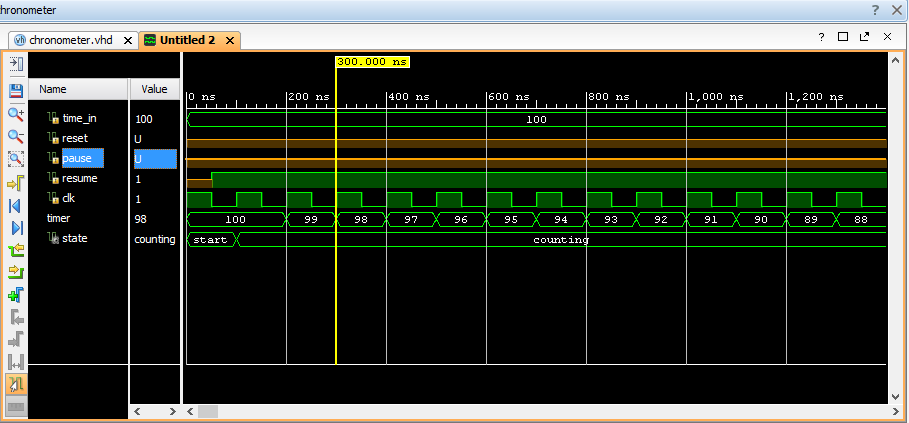
state <= start;

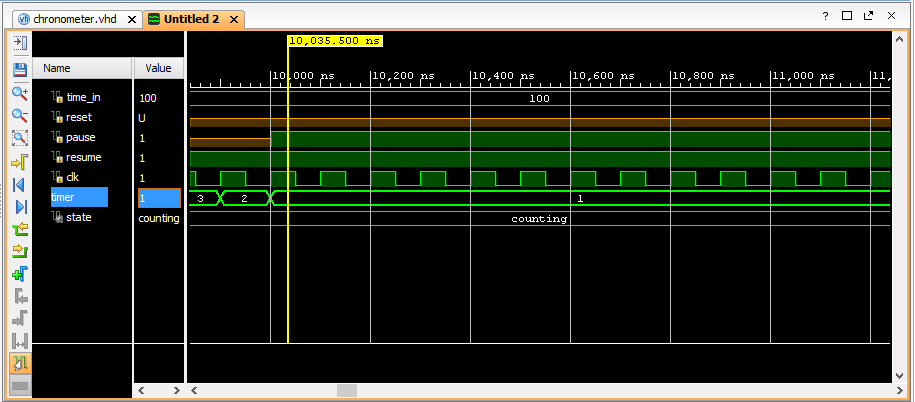
end case;

end if;

end process;

end Behavioral;

شکل موج تولید شده به صورت زیر است :



سوال ششم

کد این ماژول به صورت زیر است :

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity lock is Port (

one , zero , enter , rst , clk :in std\_logic;

unlock : out std\_logic

);

end lock;

architecture Behavioral of lock is

type state\_type is ( start , d0 , d01 , d010 , d0101, d01011 , suc , fail );

signal state : state\_type;

attribute fsm\_encoding : string;

attribute fsm\_encoding of STATE : signal is "sequential";

begin

process(clk)

begin

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

if( rst = '1')then

state <= start;

end if;

case state is

when start =>

if( rst = '1' or enter = '1' )then

state <= start;

elsif one = '1' then

state <= fail ;

elsif zero = '1' then

state <= d0;

else

state <= start;

end if;

when d0 =>

if( rst = '1' or enter = '1' ) then

state <= start;

elsif one = '1' then

state <= d01;

elsif zero = '1' then

state <= fail;

else

state <= d0;

end if;

when d01 =>

if( rst = '1' or enter = '1' ) then

state <= start;

elsif one = '1' then

state <= fail;

elsif zero = '1' then

state <= d010;

else

state <= d01;

end if;

when d010 =>

if( rst = '1' or enter = '1' ) then

state <= start;

elsif one = '1' then

state <= d0101;

elsif zero = '1' then

state <= fail;

else

state <= d010;

end if;

when d0101 =>

if( rst = '1' or enter = '1' ) then

state <= start;

elsif one = '1' then

state <= d01011;

elsif zero = '1' then

state <= fail;

else

state <= d0101;

end if;

when d01011 =>

if( one = '1' or zero = '1') then

state <= fail;

elsif rst = '1' then

state <= start;

elsif enter = '1' then

state <= suc;

else

state <= d01011;

end if;

when suc =>

if( rst = '1' or enter = '1') then

state <= start;

elsif zero = '1' or one = '1' then

state <= fail;

else

state <= suc;

end if;

when fail =>

if( rst = '1' or enter = '1' ) then

state <= start;

elsif one = '1' or zero = '1' then

state <= fail;

else

state <= fail;

end if;

end case;

end if;

end process;

process( state )

begin

if( state = suc ) then

unlock <= '1';

else

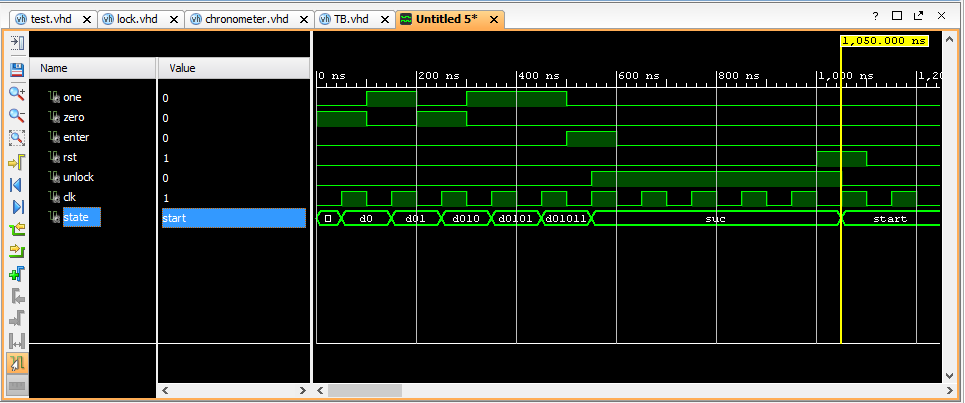
unlock <= '0';

end if;

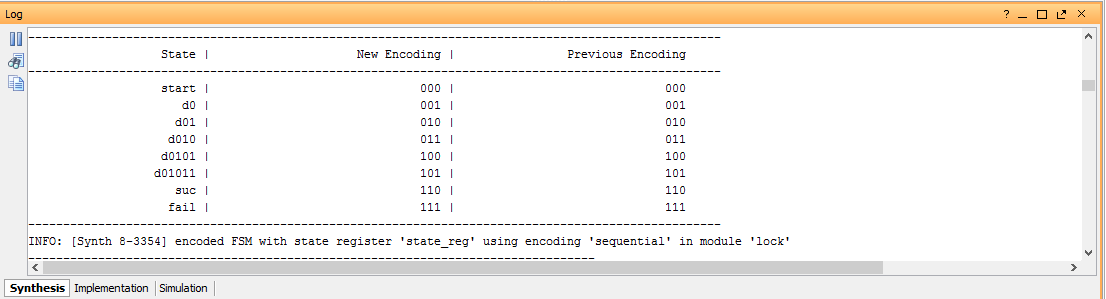
end process;

end Behavioral;

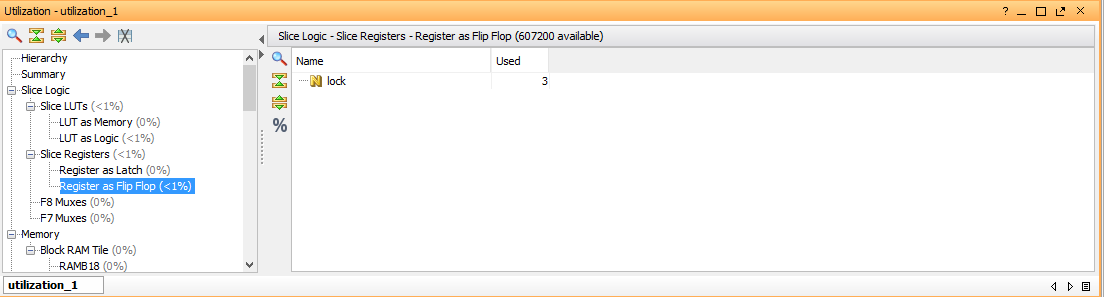
شکل موج آن به صورت زیر است :



در این طراحی از روش sequential استفاده شد زیرا تعداد حالت ها دقیقا برابر 8 بود همچنین سیستم نیز از همین حالت استفاده کرده که گزارش آن در زیر موچود می باشد :



همچنین تعداد لچ ها برابر با صفر و تعداد فلیپ فلاپ ها 3 عدد می باشد که آن هم در گزارش زیر موجود است :



و اما در مورد قسمت آخر هم کد آن به صورت زیر می باشد :

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity lock\_users is Port (

one , zero , enter , rst , clk :in std\_logic;

unlock : out std\_logic

);

end lock\_users;

architecture Behavioral of lock\_users is

type state\_type is ( start , get\_user\_id , get\_pass , get\_new\_pass , insert\_id, insert\_pass , suc , fail );

signal state : state\_type;

attribute fsm\_encoding : string;

attribute fsm\_encoding of STATE : signal is "sequential";

begin

process(clk)

variable user\_id1 : std\_logic\_vector( 1 downto 0 ) := "01";

variable user\_id2 : std\_logic\_vector( 1 downto 0 ) := "10";

variable user\_id3 : std\_logic\_vector( 1 downto 0 ) := "11";

variable temp\_id : std\_logic\_vector( 0 to 1 );

variable index\_id : integer range 0 to 1 := 0;

variable pass1,pass2,pass3 : std\_logic\_vector( 7 downto 0 );

variable temp\_pass : std\_logic\_vector( 0 to 7 );

variable temp\_new\_pass : std\_logic\_vector( 0 to 7 );

variable index\_pass : integer range 0 to 7 := 0;

variable index\_new\_pass : integer range 0 to 7 := 0;

begin

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

if( rst = '1')then

state <= start;

end if;

case state is

when start =>

if( one = '1' or zero = '1' )then

state <= insert\_id;

elsif rst = '1' then

state <= start ;

elsif enter = '1' then

state <= get\_user\_id;

else

state <= start;

end if;

when get\_user\_id =>

if( zero = '1' or one = '1' ) then

if( zero = '1')then

temp\_id(index\_id) := '0';

else

temp\_id(index\_id) := '1';

end if;

index\_id := (index\_id + 1) mod 2;

state <= get\_user\_id;

elsif enter = '1' then

state <= get\_pass;

elsif rst = '1' then

state <= start;

else

state <= get\_user\_id;

end if;

when get\_pass =>

if( one = '1' or zero = '1' ) then

if( zero = '1')then

temp\_pass(index\_pass) := '0';

else

temp\_pass(index\_pass) := '1';

end if;

index\_pass := (index\_pass + 1) mod 8;

state <= get\_pass;

elsif enter = '1' then

state <= get\_new\_pass;

elsif rst = '1' then

state <= start;

else

state <= get\_pass;

end if;

when get\_new\_pass =>

if( one = '1' or zero = '1' ) then

if( zero = '1' )then

temp\_new\_pass(index\_new\_pass) := '0';

else

temp\_new\_pass(index\_new\_pass) := '1';

end if;

index\_new\_pass := (index\_new\_pass + 1) mod 8;

state <= get\_new\_pass;

elsif enter = '1' then

if( pass1 = temp\_pass)then

case temp\_id is

when "01" =>

pass1 := temp\_new\_pass;

when "10" =>

pass2 := temp\_new\_pass;

when "11" =>

pass3 := temp\_new\_pass;

when others =>

end case;

end if;

state <= start;

elsif rst = '1' then

state <= start;

else

state <= get\_new\_pass;

end if;

when insert\_id =>

if( one = '1' or zero = '1' ) then

if( zero = '1')then

temp\_id(index\_id) := '0';

else

temp\_id(index\_id) := '1';

end if;

index\_id := (index\_id + 1) mod 2;

state <= insert\_id;

elsif enter = '1' then

state <= insert\_pass;

elsif rst = '1' then

state <= start;

else

state <= insert\_id;

end if;

when insert\_pass =>

if( one = '1' or zero = '1') then

if( zero = '1')then

temp\_pass(index\_pass) := '0';

else

temp\_pass(index\_pass) := '1';

end if;

index\_pass := (index\_pass + 1) mod 8;

state <= insert\_pass;

elsif rst = '1' then

state <= start;

elsif enter = '1' then

case temp\_id is

when "01" =>

if( pass1 = temp\_pass)then

state <= suc;

else

state <= fail;

end if;

when "10" =>

if( pass2 = temp\_pass)then

state <= suc;

else

state <= fail;

end if;

when "11" =>

if( pass3 = temp\_pass)then

state <= suc;

else

state <= fail;

end if;

when others =>

end case;

else

state <= insert\_pass;

end if;

when suc =>

if( rst = '1' or enter = '1') then

state <= start;

elsif zero = '1' or one = '1' then

state <= fail;

else

state <= suc;

end if;

when fail =>

if( rst = '1' or enter = '1' ) then

state <= start;

elsif one = '1' or zero = '1' then

state <= fail;

else

state <= fail;

end if;

end case;

end if;

end process;

process( state )

begin

if( state = suc ) then

unlock <= '1';

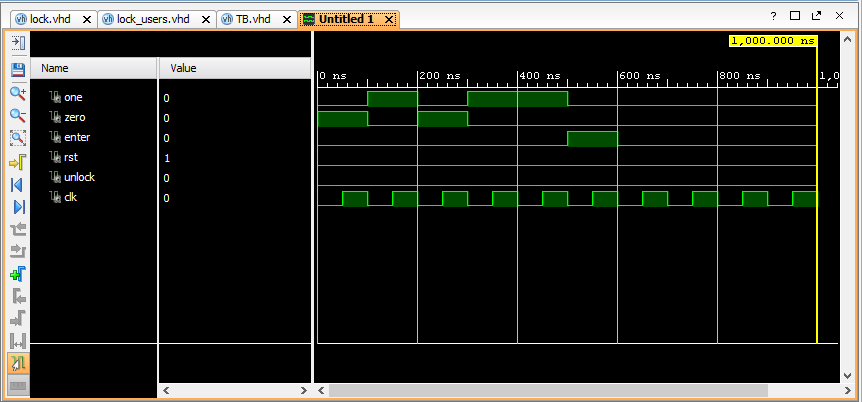
else

unlock <= '0';

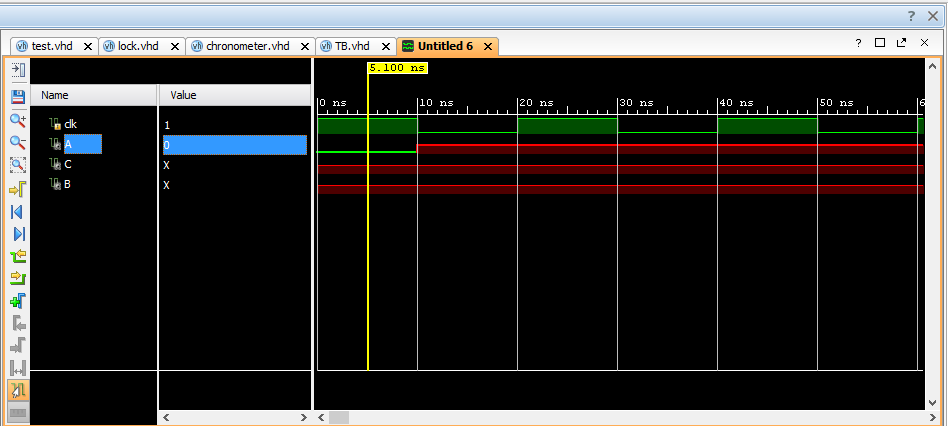
end if;

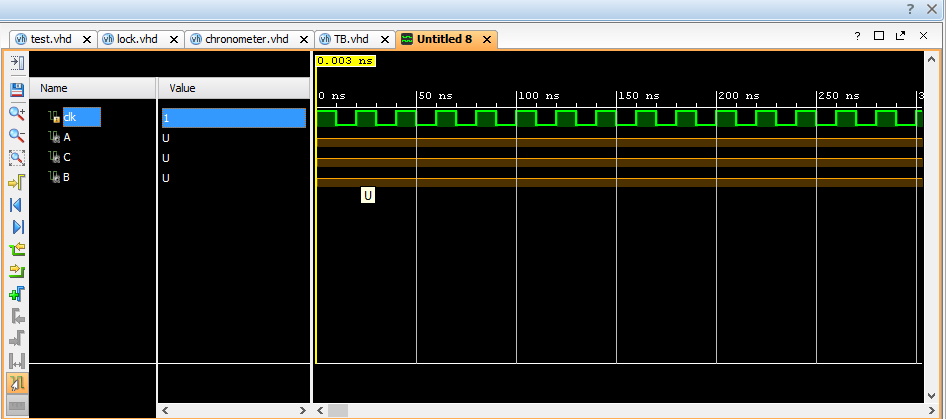
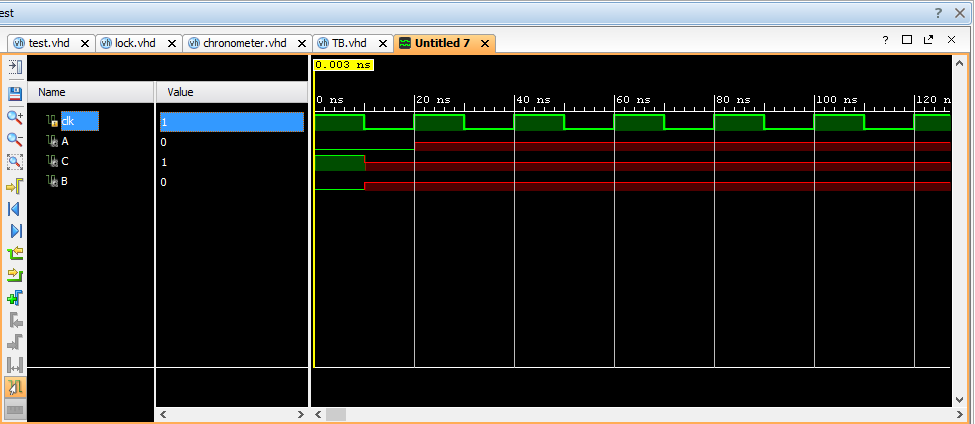
end process;

end Behavioral;



سوال هفتم

در این سوال برای سیگنال های B , C دو درایور در نظر گرفته شده است که همین امر باعث می شود مقدار آن ها در شرایطی تبدیل به X بشود . ( البته کد داده شده اندکی مشکل دارد و باید سیگنال ها حتما مقدار اولیه بگیرند و الا ابزار vivado مقدار آن ها را برابر با U در نظر میگیرد. در تصاویر زیر انواع مقدار دهی اولیه انجام شده است و در آخری مقدار دهی اولیه نشده)



سوال هشتم

کد رمز کننده به صورت زیر است :

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity encryptor is Port (

input : in std\_logic\_vector( 1 to 15 );

output : out std\_logic\_vector( 1 to 20 )

);

end encryptor;

architecture Behavioral of encryptor is

signal p1 , p2 , p4 , p8 , p16 : std\_logic;

begin

p1 <= input(1) xor input(2) xor input(4) xor input(5) xor input(7) xor

input(9) xor input(11) xor input(12) xor input(14);

p2 <= input(1) xor input(3) xor input(4) xor input(6) xor input(7) xor

input(10) xor input(11) xor input(13) xor input(14);

p4 <= input(2) xor input(3) xor input(4) xor input(8) xor input(9) xor

input(10) xor input(11) xor input(15);

p8 <= input(5) xor input(6) xor input(7) xor input(8) xor input(9) xor

input(10) xor input(11);

p16 <= input(12) xor input(13) xor input(14) xor input(15);

output <= p1 & p2 & input(1) & p4 & input( 2 to 4) & p8 & input( 5 to 11 ) & p16 & input( 12 to 15 );

end Behavioral;

کد رمز گشا به صورت زیر است :

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

USE ieee.numeric\_std.ALL;

entity decryptor is Port (

input : in std\_logic\_vector( 1 to 20 );

output : out std\_logic\_vector( 1 to 15)

);

end decryptor;

architecture Behavioral of decryptor is

signal numsig : integer ;

begin

process( input )

variable p1,p2,p4,p8,p16,tempsig : std\_logic ;

variable num : integer range 1 to 20 ;

variable temp : std\_logic\_vector( 1 to 5 );

variable temp\_input : std\_logic\_vector( 1 to 20 );

begin

p1 := input(1) xor input(3) xor input(5) xor input(7) xor input(9) xor

input(11) xor input(13) xor input(15) xor input(17) xor input(19);

p2 := input(2) xor input(3) xor input(7) xor input(6) xor input(10) xor

input(11) xor input(14) xor input(15) xor input(18) xor input(19);

p4 := input(4) xor input(5) xor input(6) xor input(7) xor input(12) xor

input(13) xor input(14) xor input(15) xor input(20);

p8 := input(8) xor input(9) xor input(10) xor input(11) xor input(12) xor

input(13) xor input(14) xor input(15);

p16 := input(16) xor input(17) xor input(18) xor input(19) xor input(20);

if( p1 = '0' and p2 = '0' and p4 = '0' and p8 = '0' and p16 = '0' )then

output <= input(3)& input( 5 to 7 ) & input(9 to 15) & input(17 to 20);

else

temp := p16&p8&p4&p2&p1;

num := to\_integer(unsigned(temp));

numsig <= num;

if( num > 0 and num < 16 ) then

temp\_input := input;

tempsig := not temp\_input(num);

temp\_input(num) := tempsig;

output <= temp\_input(3) & temp\_input(5 to 7) & temp\_input(9 to 15) & temp\_input(17 to 20);

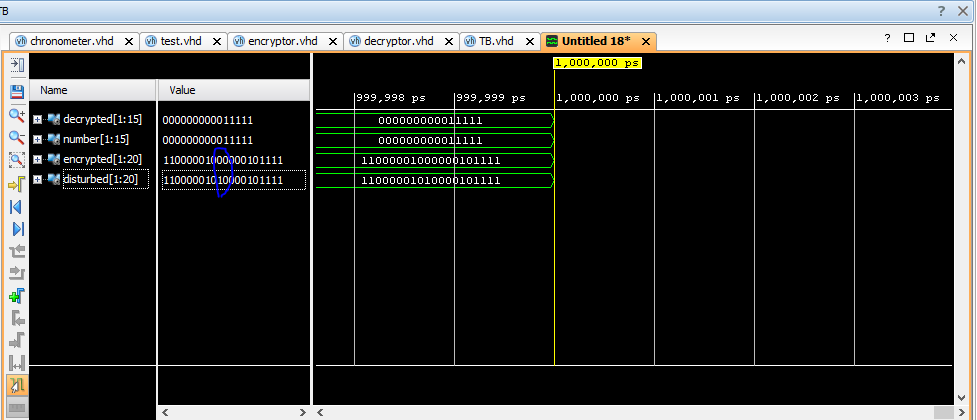
end if;

end if;

end process;

end Behavioral;

شکل موج ها به صورت زیر است :



کد تست بنج آن به صورت زیر نوشته شده است :

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity TB is

-- Port ( );

end TB;

architecture encryptor of TB is

component encryptor is Port (

input : in std\_logic\_vector( 1 to 15 );

output : out std\_logic\_vector( 1 to 20 )

);

end component;

component decryptor is Port (

input : in std\_logic\_vector( 1 to 20 );

output : out std\_logic\_vector( 1 to 15)

);

end component;

signal decrypted,number : std\_logic\_vector( 1 to 15);

signal encrypted , disturbed: std\_logic\_vector( 1 to 20 );

begin

number <= "000000000011111";

disturbed <= encrypted( 1 to 9 ) & not encrypted(10) & encrypted( 11 to 20);

encrypt: encryptor port map ( number , encrypted );

decrypt: decryptor port map ( disturbed , decrypted );

end encryptor;

architecture lock of TB is

component lock is Port (

one , zero , enter , rst , clk :in std\_logic;

unlock : out std\_logic

);

end component;

signal one , zero , enter , rst , unlock: std\_logic;

signal clk : std\_logic := '0' ;

begin

MODULE: lock port map ( one , zero , enter , rst , clk , unlock );

one <= '0' ,

'1' after 100ns,

'0' after 200ns,

'1' after 300ns,

'0' after 500ns;

zero <= '1',

'0' after 100ns,

'1' after 200ns,

'0' after 300ns ;

enter <= '0',

'1' after 500ns,

'0' after 600ns;

rst <= '0',

'1' after 1000ns,

'0' after 1100ns;

clk <= not clk after 50ns;

end lock;