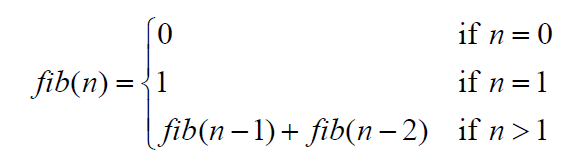
Lab2-Excerises5斐波那契数列

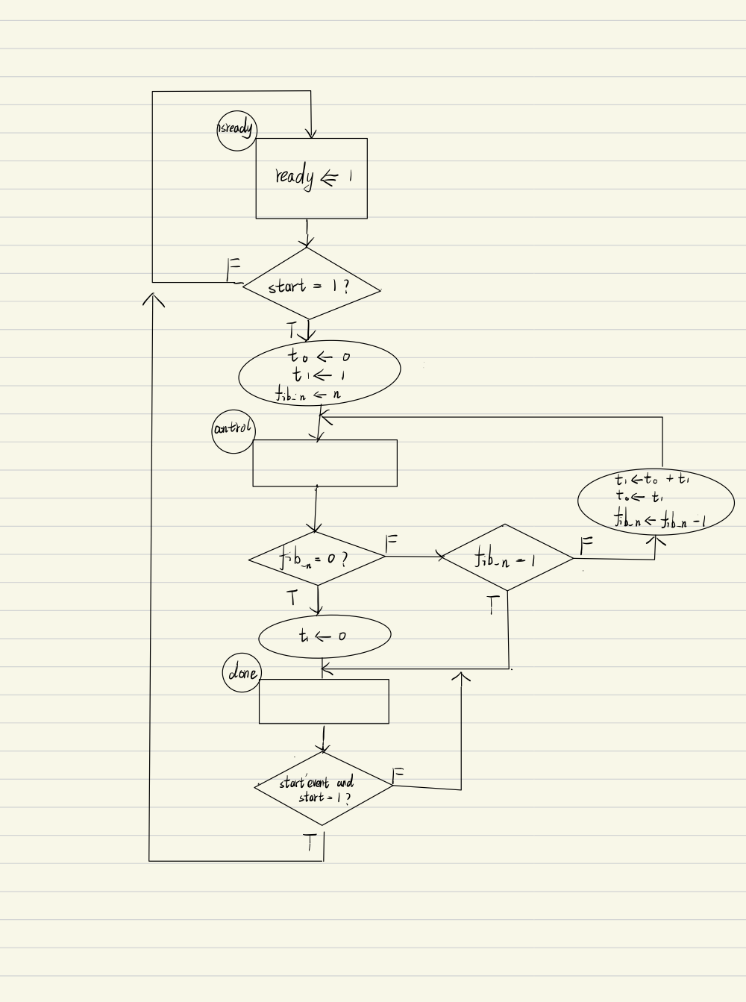
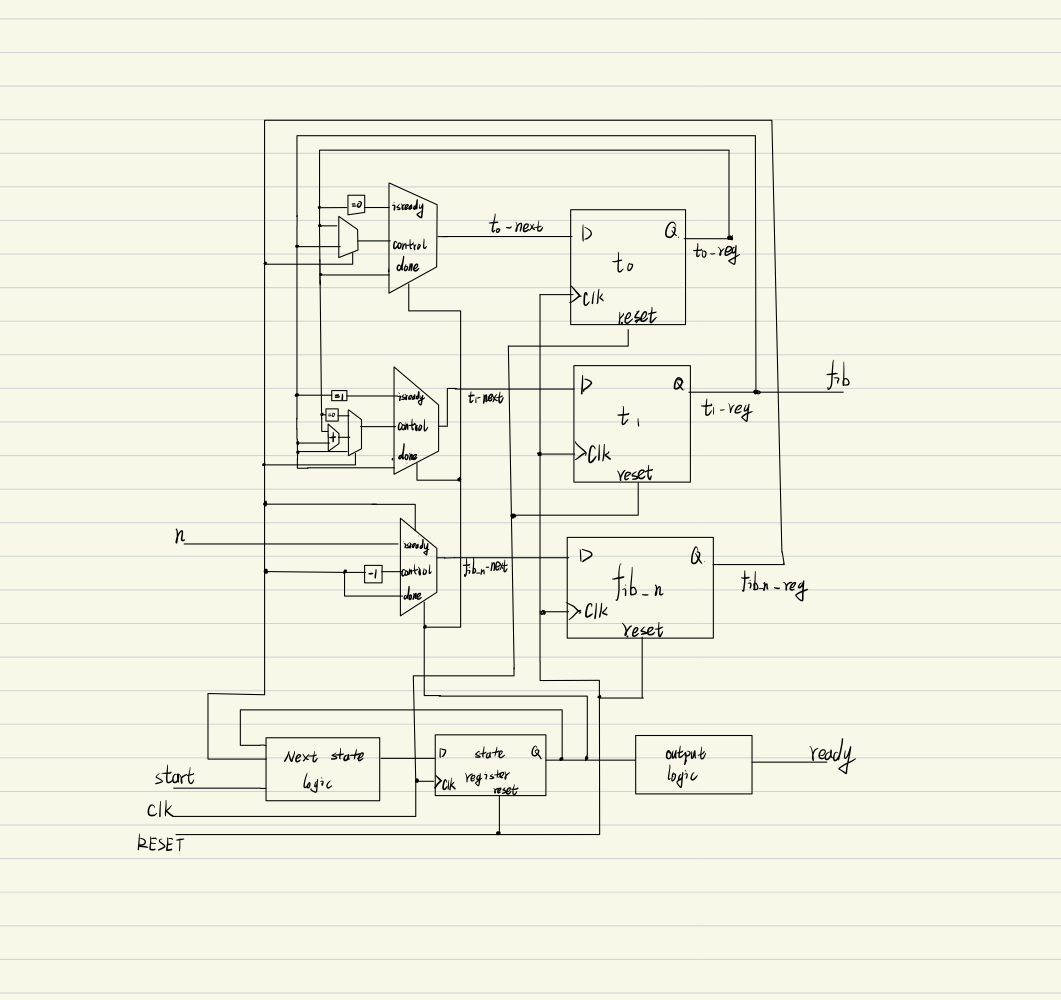
高政 11912015

1. 简述
   1. 实验目的

通过本实验了解带数据路径的有限状态机（FSMD）的基本设计流程，学习VHDL语言中的case语句的使用。

* 1. 斐波那契数列

斐波那契数列数列的求解公式如上所示，本次实验中输入n是一个六比特的二进制信号，输出则是43比特的信号。

1. 实验准备

斐波那契数列的ASM框图如左上图，其相应电路如右上图。

伪代码如下：

Fib(n):{

if (n = 0) then

fib = 0;

else if (n = 1) then

fib = 1;

else {

fib = Fib(n - 1) + Fib(n - 2);

}

}

1. VHDL代码与test bench代码

斐波那契数列VHDL代码实现：

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.std\_logic\_unsigned.all;

use IEEE.std\_logic\_arith.all;

entity fsmd is

Port ( CLK : in STD\_LOGIC;

RESET : in STD\_LOGIC;

start : in std\_logic;

n : in STD\_LOGIC\_VECTOR (5 downto 0);

ready : out std\_logic;

fib : out STD\_LOGIC\_VECTOR (42 downto 0));

end fsmd;

architecture Behavioral of fsmd is

type state\_type is(isready, contro, done);

signal state, state\_next : state\_type;

signal t0\_next, t1\_next, t0\_reg, t1\_reg : std\_logic\_vector(42 downto 0);

signal fib\_n\_next, fib\_n\_reg : std\_logic\_vector(5 downto 0);

signal do : std\_logic;

begin

clk\_pro: process(CLK, RESET)is

begin

if RESET = '1' then

state <= isready;

t0\_reg <= (others => '0');

t1\_reg <= (others => '0');

fib\_n\_reg <= n;

elsif (CLK'event and clk = '1')then

state <= state\_next;

t0\_reg <= t0\_next;

t1\_reg <= t1\_next;

fib\_n\_reg <= fib\_n\_next;

end if;

end process clk\_pro;

-- combinational circuit

process(state, state\_next, t0\_reg, t0\_next, t1\_reg, t1\_next, fib\_n\_reg, fib\_n\_next) is

begin

state\_next <= state;

t0\_next <= t0\_reg;

t1\_next <=t1\_reg;

fib\_n\_next <= fib\_n\_reg;

ready <= '0';

case state is

when isready =>

if start = '1' then

t0\_next <= (others => '0');

t1\_next <= (0 => '1', others => '0');

state\_next <= contro;

else

state\_next <= isready;

end if;

ready <= '1';

when contro =>

if fib\_n\_reg = 0 then

t1\_next <= (others => '0');

state\_next <= done;

elsif fib\_n\_reg = 1 then

state\_next <= done;

else

t0\_next <= t1\_reg;

t1\_next <= t0\_reg + t1\_reg;

fib\_n\_next <= fib\_n\_reg - 1;

state\_next <= contro;

end if;

when done =>

state\_next <= isready；

end case;

end process;

fib <= t1\_reg;

end Behavioral;

斐波那契数列testbench代码如下：

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity tb\_fsmd is

end tb\_fsmd;

architecture Behavioral of tb\_fsmd is

component fsmd is

Port ( CLK : in STD\_LOGIC;

RESET : in STD\_LOGIC;

start : in std\_logic;

n : in STD\_LOGIC\_VECTOR (5 downto 0);

ready : out std\_logic;

fib : out STD\_LOGIC\_VECTOR (42 downto 0));

end component;

--input

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

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

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

signal n : std\_logic\_vector(5 downto 0) := "000000";

--output

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

signal fib : std\_logic\_vector(42 downto 0) := (others => '0');

constant per : time := 1 us;

begin

uut: fsmd

port map(

CLK => CLK,

RESET => RESET,

start => start,

n => n,

ready => ready,

fib => fib

);

clk\_pro: process

begin

CLK <= '0';

wait for per/2;

CLK <= '1';

wait for per/2;

end process;

reset\_pro: process

begin

RESET <= '1';

for i in 1 to 2 loop

wait until CLK = '1';

end loop;

RESET <= '0';

wait;

end process;

in\_pro: process

begin

start <= '1';

n <= "111111";

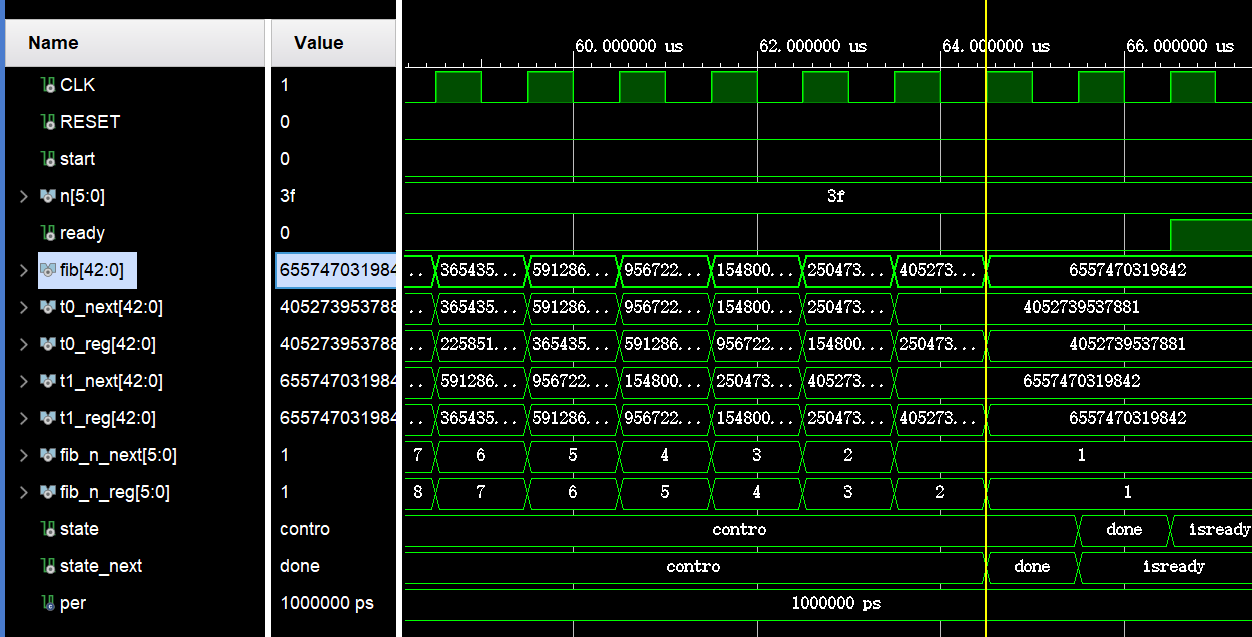
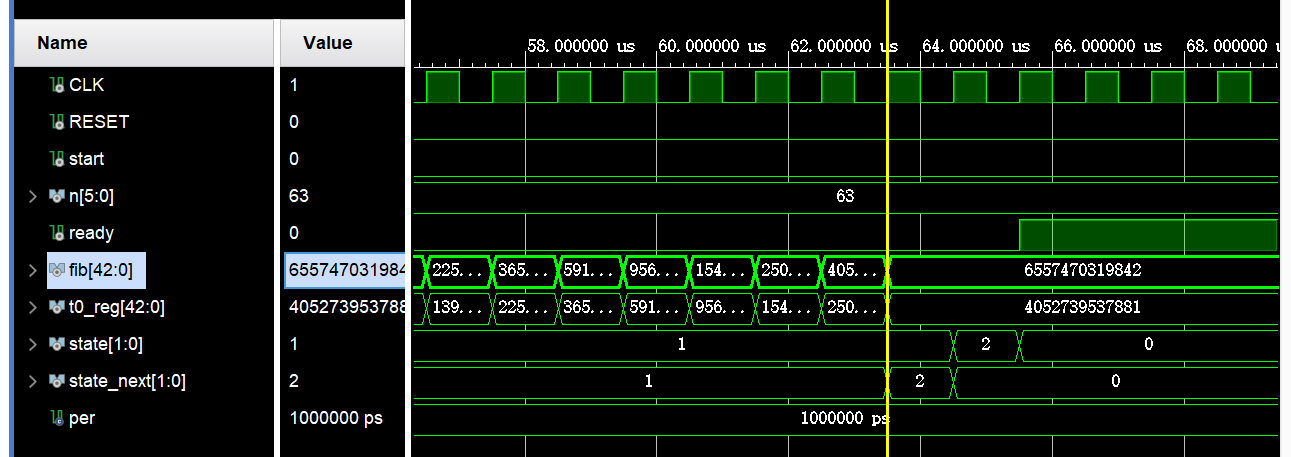
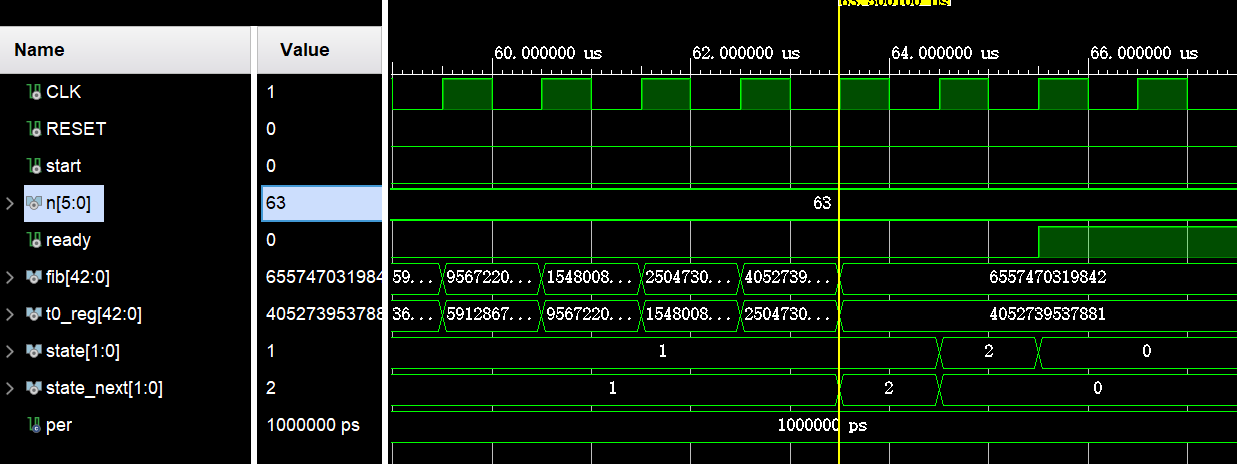
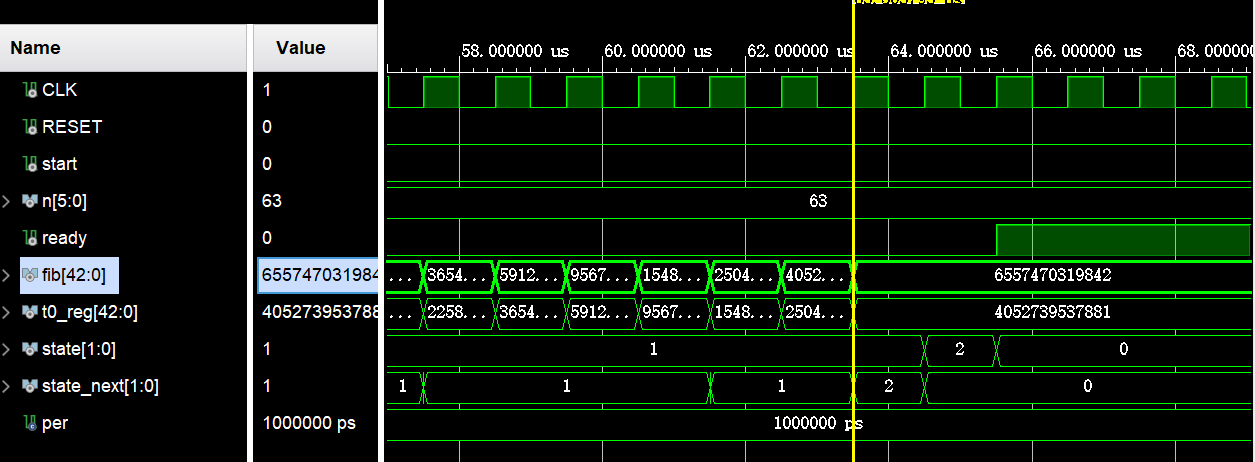
wait for 10\*per;

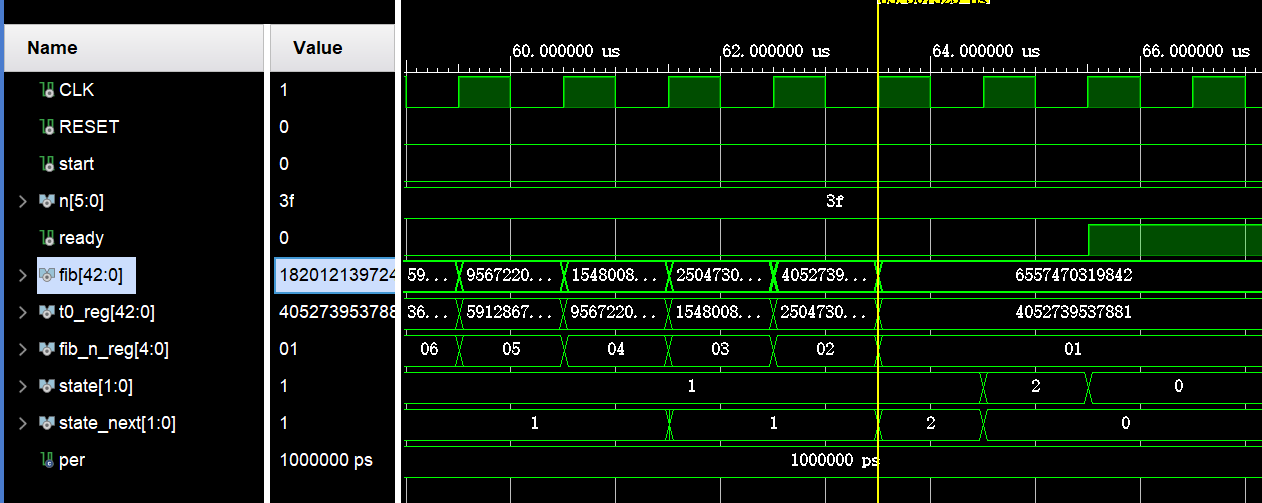
start <= '0';

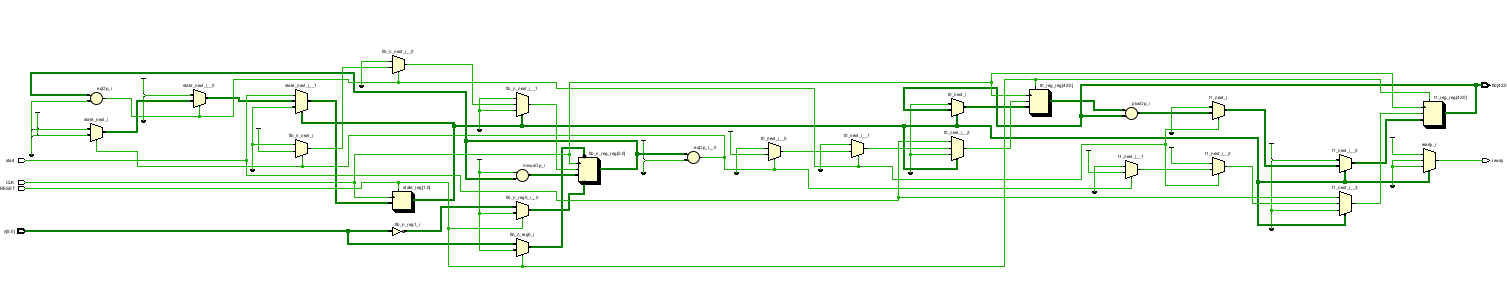
wait;

end process;

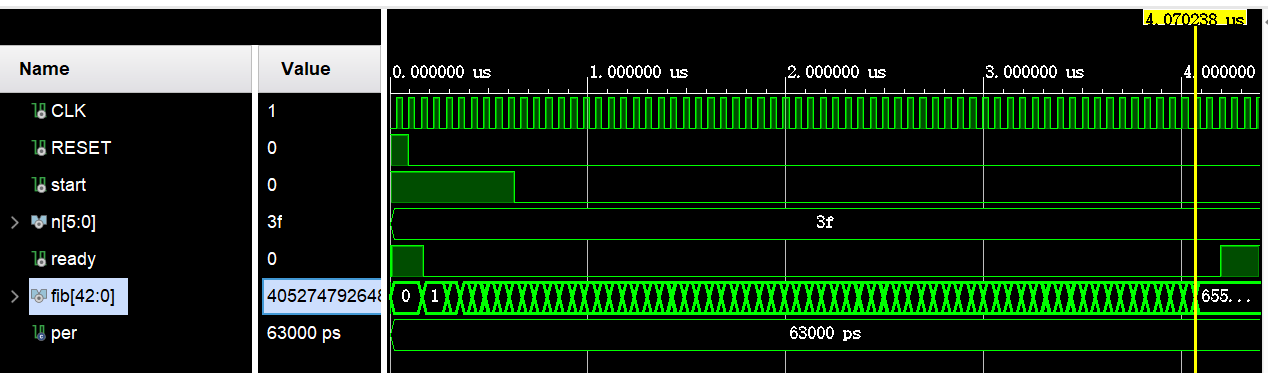
end Behavioral;

1. 实验结果与分析
   1. 行为仿真
   2. 综合后功能仿真
   3. 综合后时序仿真
   4. 布线后功能仿真
   5. 布线后时序仿真



* 1. RTL原理图

由以上结果可以看出，代码基本实现预期目标，能够完成计算斐波那契数列的效果。

* 1. 性能估计

经测试，本程序能正常工作的最短时钟周期为63ns。

1. 实验结论

通过本次实验我了解了FSMD的基本设计流程和VHDL语句中的case语句的使用。