数字系统设计实验论文

论文题目：看门狗定时器实验论文

作者姓名：苏瑞芝　王航

小组序号：21

联系方式：704311473@qq.com

单位　　：中国海洋大学　2012 级通信工程

导师姓名：郑海永

摘要：看门狗，即所谓的 watchdog ，它的作用是看程序而不是看门。当软件跑飞时，可为系统进行复位操作；

关键词：复位、控制、检测计时。

选题背景：利用 FPGA 或 CPLD 的剩余资源设计看门狗模块，相当于硬件看门狗的一种可节省专用的看门狗芯片，且在不同时序要求上灵活修改；可根据系统要求增加与主控芯片的握手信号。现在许多我们做实验很多程序因为错误会无限循环，浪费大量的时间和缓存，同时我们现在实验课时间有限，因此，看门狗实验很有必要。

意义：通过这次实验，我们不仅认识到了自己的不足，更认识到了队合作的重要性。对 VHDL 的适用性感到惊奇，坚定了对本专业和对未来前途的信心。同时，我们在制作过程中也遇到了许多困难，这要求我们不断努力学习和向身边的学霸们学习。

相关工作：

1、由功能分析看门狗定时器的各种状态，分析状态转移，画出状态转移图。

2、 认真学习有关状态机设计的相关知识，初步设计 FSM 状态机。

3、根据状态转移图，编写 VHDL 代码。

4、进行波形仿真，电路下载.

分工：

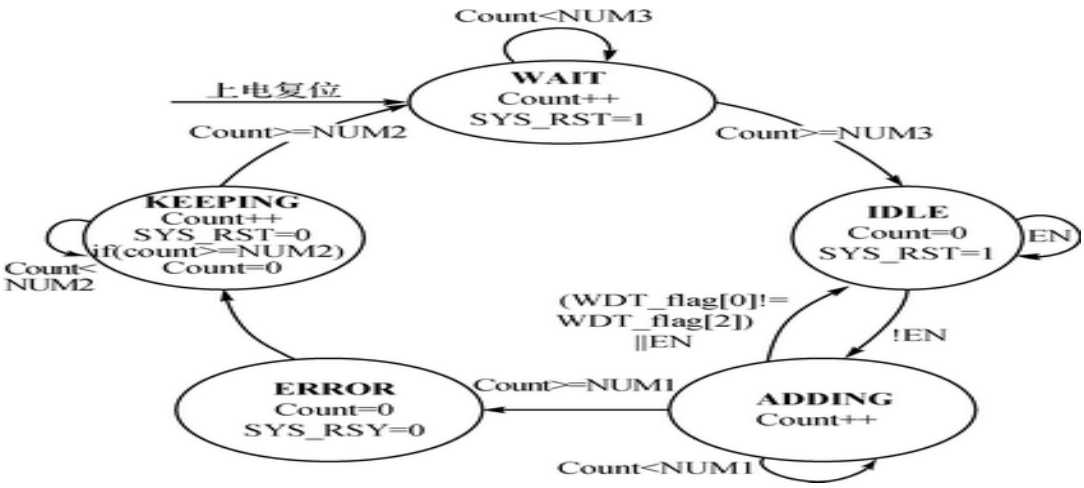
共同学习有关状态机设计的相关知识，初步设计 FSM 状态机。

王航：初步编写 VHDL 代码。

苏瑞芝：修改代码，进行波形仿真，电路下载。

实验与分析：

实验原理：



总结：在数字伺服控制系统中，利用其中已包含的 CPLD 或 FPGA 电路设计硬件看门狗模块，既可以满足系统对硬件看门狗功能的需求，又可以节省专用的看门狗芯片，节省电路板的空间，提高了系统的可靠性，提高了可编程逻辑器件的资源利用率，并且可以针对不同的系统上电、复位等时序要求灵活配置时间参数。经验证，设计达到了看门狗电路的功能要求，能够有效保证软件的可靠性，亦可应用于其他数字控

制系统平台。

致谢：高年级学长帮我们修改程序，我们为了实验程序和同年级同学交换课程或者耽误他们的实验课。

附录一实验代码：

```
-- Company:
-- Engineer:
--
-- Create Date:    09:05:08 06/14/2014
-- Design Name:
-- Module Name:    bbc - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------
```

```
----------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity bbc is
    Port ( clk : in  STD_LOGIC;
           fed : in  STD_LOGIC;
           ed : in  STD_LOGIC;
           reset : out  STD_LOGIC);
end bbc;
```

```
architecture Behavioral of bbc is
 type state_type is (A ,B ,C ,D,E);
signal y :state_type;
begin
process(fed ,clk ,ed)
variable count :integer;
variable start :STD_LOGIC;
begin
start:='0';
count:=0;
if(count=0 and start='0') then y<=D;
elsif(clk'event and clk='1')  then
case y is
when D => y<=E;
when E => if count<20  then count:=count+1 ;
           else y <= A;
               count:=0;
          end if;
when A =>  if count<30 then count:=count+1;
            else y<=B; count:=0;
             end if;
when B => if ed='1' then y<=B; end if  ;
```

```
            if ed='0'  then y<=C; end if;
when C => if((fed'event and fed='1') or ed='1') then
y<=B;
            elsif count<50  then count:=count+1  ;
               else y <= D;
         end if;
end case;
if(y=A or y=B or y=C) then start:='1';
end if;
end if;
end process;
reset <='1' when (y=A or y=B or y=C) else '0';
end Behavioral;
```

附录二波形图：