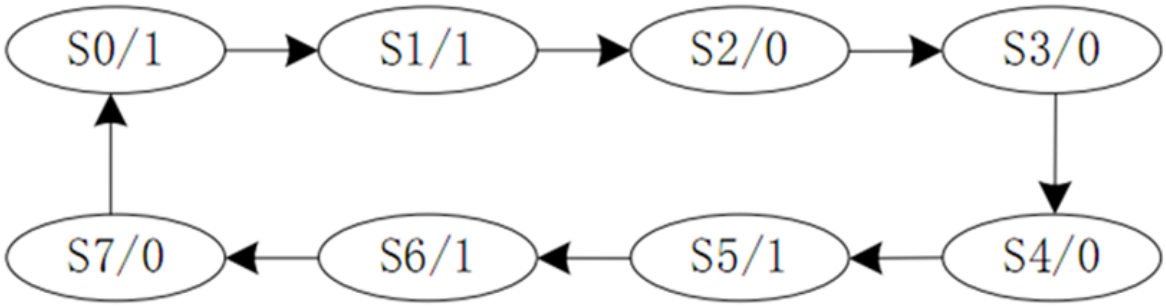
电子系统设计-第二次作业

PB20051061 牟真伟

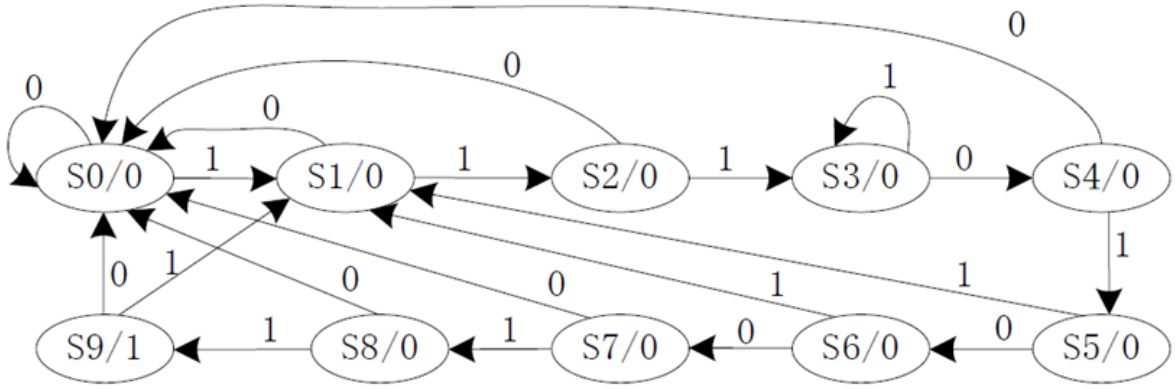
一、序列产生器可以采用有限状态机来描述。如图所示为状态机正常状态下的时序转换图，输入为clk和reset，reset有效时状态机处于S7状态，reset无效时在clk上升沿进行状态转换。参考例5-48编写该序列产生器的VHDL程序。




```
-- 作业1 序列产生器 @PB20051061牟真伟

library ieee;
use ieee.std_logic_1164.all;

entity SEQ_GEN is
    port(
        clk :IN std_logic;
        reset :IN std_logic;
        output :OUT std_logic
    );
end SEQ_GEN;

architecture arch_SEQ_GEN of SEQ_GEN is
    type states is (S0,S1,S2,S3,S4,S5,S6,S7);
    signal st :states;
begin
    -- 状态寄存器和次态逻辑
    process(clk,reset)
    begin
        if(reset = '1') then
            st <= S7;
        elsif(clk'event and clk='1') then
            case st is
                when s0 => st <= S1;
                when s1 => st <= S2;
                when s2 => st <= S3;
                when s3 => st <= S4;
                when s4 => st <= S5;
                when s5 => st <= S6;
```

```
                when s6 => st <= S7;
                when s7 => st <= S0;
            end case;
        end if;
    end process;

    -- 输出逻辑
    process(st)
    begin
        if(st=S0 or st=S1 or st=S5 or st=S6) then
            output <= '1';
        else
            output <= '0';
        end if;
    end process;
end arch_SEQ_GEN;
```

二、序列检测器可以采用有限状态机来描述。如图所示为序列检测器的状态转换图，输入为clk、reset和外部数据，reset有效时状态机处于S0状态，reset无效时在clk上升沿进行状态转换。参考例5-48编写该序列检测器的VHDL程序。



若检测序列为111010011时，上图S5状态有误，S5状态输入为1时应跳转到S2状态，而不是S1状态。

```
-- 作业2 序列检测器 @PB20051061牟真伟

library ieee;
use ieee.std_logic_1164.all;

entity SEQ_DECT is
    port(
        clk :IN std_logic;
        reset :IN std_logic;
        data_input :IN std_logic;
        detector_out :OUT std_logic
    );
end SEQ_DECT;

architecture arch_SEQ_DECT of SEQ_DECT is
    type states is (S0,S1,S2,S3,S4,S5,S6,S7,S8,S9);
    signal st :states;
begin
```

```
    -- 状态寄存器和次态逻辑
    process(clk,reset)
    begin
        if(reset = '1') then
            st <= S0;
        elsif(clk'event and clk='1') then
            case st is
                when s0 => if(data_input = '1') then
                                st <= S1;
                           else
                                st <= S0;
                           end if;
                when s1 => if(data_input = '1') then
                                st <= S2;
                           else
                                st <= S0;
                           end if;
                when s2 => if(data_input = '1') then
                                st <= S3;
                           else
                                st <= S0;
                           end if;
                when s3 => if(data_input = '1') then
                                st <= S3;
                           else
                                st <= S4;
                           end if;
                when s4 => if(data_input = '1') then
                                st <= S5;
                           else
                                st <= S0;
                           end if;
                when s5 => if(data_input = '1') then
                                st <= S2; --按检测序列时
                                -- st <= S1; -- 按状态转移图时
                           else
                                st <= S6;
                           end if;
                when s6 => if(data_input = '1') then
                                st <= S1;
                           else
                                st <= S7;
                           end if;
                when s7 => if(data_input = '1') then
                                st <= S8;
                           else
                                st <= S0;
                           end if;
                when s8 => if(data_input = '1') then
                                st <= S9;
                           else
                                st <= S0;
                           end if;
                when s9 => if(data_input = '1') then
                                st <= S1;
```

```
                                else
                                    st <= S0;
                                end if;
            end case;
        end if;
    end process;

    -- 输出逻辑
    detector_out <= '1' when st=S9 else '0';
end arch_SEQ_DECT;
```