
Logic and Computer Design Fundamentals

VHDL

Part 4 – Chapter 7 – Registers and Counters

Charles Kime & Thomas Kaminski

© 2004 Pearson Education, Inc.

[Terms of Use](#)

(Hyperlinks are active in View Show mode)

Overview

- **Part 1 - VHDL Basics and Types of Descriptions**
- **Part 2 - Behavioral and Hierarchical Description**
- **Part 3 - Finite State Machines**
- **Part 4 - Registers and Counters**
 - **Registers**
 - **Shift Registers**
 - **Counters**
 - **Examples**
 - **4-bit Left Shift Register with Reset**
 - **4-bit Binary Counter with Reset**
- **Part 5 - Algorithmic State Machine Example: Binary Multiplier**

VHDL for Registers and Counters

- Register - similar description to a flip-flop except contains multiple bits:

```
signal Q, D : std_logic_vector(15 downto 0);
process (CLK, RESET)
begin
    if (RESET = '1') then
        Q <= B"0000000000000000";
    elsif (CLK'event and CLK = '1')
        Q <= D;
    end
```

- Shift Register – use:

- Shift operators

Q <= Q sll 1; -- By default fills rightmost bit with
-- 0; defaults differ depending on the shift type;

- Concatenation

Q <= Q(14:0) & SI; -- SI is one bit shift input

- Counter – use increment, decrement, add, subtract:

count <= count + "0001"; -- count is four bits

VHDL Description of Left Shift Register

```
library ieee;
use ieee.std_logic_1164.all;
entity srg_4_r is
    port (CLK, RESET, SI: in std_logic;
          Q : out std_logic_vector(3 downto 0);
          SO : out std_logic);
end srg_4_r;
architecture sequential of srg_4_r is
    signal shift: std_logic_vector(3 downto 0);
begin
    process (RESET, CLK)
    begin
        if (RESET = '1') then
            shift <= "0000";
        elsif (CLK'event and (CLK = '1')) then
            shift <= shift(2 downto 0) & SI;
        end if;
    end process;
    Q <= shift;
    SO <= shift(3);
end sequential;
```

VHDL Description of Binary Counter

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
entity count_4_r is
    port(CLK, RESET, EN: in std_logic;
         Q : out std_logic_vector(3 downto 0);
         CO : out std_logic);
end count_4_r;
architecture sequential of count_4_r is
    signal count: std_logic_vector(3 downto 0);
begin
    process (RESET, CLK)
    begin
        if (RESET = '1') then
            count <= "0000";
        elsif (CLK'event and
              (CLK = '1') and (EN = '1'))
            then
                count <= count + "0001";
            end if;
        end process;
        Q <= count;
        CO <= '1' when (count = "1111"
                       and EN = '1') else '0';
    end sequential;
```

Terms of Use

- © 2004 by Pearson Education, Inc. All rights reserved.
- The following terms of use apply in addition to the standard Pearson Education [Legal Notice](#).
- Permission is given to incorporate these materials into classroom presentations and handouts only to instructors adopting Logic and Computer Design Fundamentals as the course text.
- Permission is granted to the instructors adopting the book to post these materials on a protected website or protected ftp site in original or modified form. All other website or ftp postings, including those offering the materials for a fee, are prohibited.
- You may not remove or in any way alter this Terms of Use notice or any trademark, copyright, or other proprietary notice, including the copyright watermark on each slide.
- [Return to Title Page](#)