Logic and Computer Design Fundamentals

VHDL

Part 2 – Chapter 5 – Behavioral and Hierarchical Description

Charles Kime & Thomas Kaminski

Overview

- Part 1 - VHDL Basics and Types of Descriptions
- Part 2 - Behavioral and Hierarchical Description
  - VHDL Hierarchy
    - Entities
    - Entity Instantiation
  - Behavioral Descriptions
    - Using:
      - Assignment statements
      - Higher-level operators
- Part 3 - Finite State Machines
- Part 4 - Registers and Counters
- Part 5 - Algorithmic State Machine Example: Binary Multiplier

Hierarchical & Behavioral VHDL Example - 1

```
-- TOP LEVEL OF HIERARCHY
-- 4-bit adder-subtractor
library ieee;
use ieee.std_logic_1164.all;
entity addsub is
 port(A,B: in std_logic_vector(3 downto 0);
      sub: in std_logic;
      R: out std_logic_vector(3 downto 0));
end addsub;
architecture hierarchy of addsub is
 component add
  port(X,Y: in std_logic_vector(3 downto 0);
       C_in: in std_logic;
       S: out std_logic_vector(3 downto 0));
 end component;
```

Hierarchical & Behavioral VHDL Example - 2

```
component M1comp
  port(data_in: in std_logic_vector(3 downto 0);
       comp: in std_logic;
       data_out: out std_logic_vector(3 downto 0));
 end component;

signal data_out: std_logic_vector(3 downto 0);
begin
 a1: add
  port map(X => A, Y => data_out, C_in => sub, S =>
  R);
 m1: M1comp
  port map(data_in => B, data_out => data_out, comp
  => sub);
end hierarchy;
```

Hierarchical & Behavioral VHDL Example - 3

```
--add ENTITY AS INSTANTIATED ABOVE
library ieee;
use ieee.std_logic_1164.all, ieee.std_logic_unsigned.all;
-- Package "unsigned" required for type conversion to
-- perform + on std_logic signals instead of on integers
entity add is
 port(X,Y: in std_logic_vector(3 downto 0);
       C_in: in std_logic;
       S: out std_logic_vector(3 downto 0));
end add;
architecture behavioral of add is
begin
 S <= X + Y + ("000" & C_in);
end behavioral;
```

Hierarchical & Behavioral VHDL Example - 4

```
--M1comp ENTITY AS INSTANTIATED ABOVE
library ieee;
use ieee.std_logic_1164.all;
entity M1comp is
 port(data_in: in std_logic_vector(3 downto 0);
       comp: in std_logic;
       data_out: out std_logic_vector(3 downto 0));
end M1comp;
architecture behavioral of M1comp is
begin
 data_out <= (comp & comp & comp & comp) xor
  data_in;
end behavioral;
```

Terms of Use

- Return to Title Page