```
-- full_adder.vhd
-- VHDL file for a full adder: a + b + c_in = c_out sum
--    e.g. for a=0, b=1, c_in=1: 0+1+1 = 10
entity full_adder is
      port(
            a, b, c_in     : IN std_logic;
            c_out, sum : out std_logic);
end full_adder;

architecture [name]…
begin
      -- full adder descriptive code
end [name];
```

```
-- parallel_adder_4bit.vhd
library…
use…

entity parallel_adder_4bit is
      generic (width: integer := 8);
      port(
            a, b       : IN std_logic_vector(width downto 1);
            c_in       : IN std_logic;
            c_out      : out std_logic;
            sum        : out std_logic_vector(width downto 1));
end parallel_adder_4bit;

arch… arch_name2…
      -- declare the component full_adder here (once)
      component full_adder
            port(
                  a, b, c_in : IN std_logic;
                  c_out, sum : out std_logic);
      end component;

      signal carry : std_logic_vector(1 to width-1);
begin
```

```
    -- instantiate the component here (once for every instance; four times)
```

Component instantiation

Method 1: explicit port mapping

```
    adder1: full_adder
        PORT MAP(   a       =>  a(1),
                    b       =>  b(1),
                    c_in    =>  c_in,
                    c_out   =>  carry(1),
                    sum     =>  sum(1));

    adder2: full_adder
        PORT MAP(   a       =>  a(2),
                    b       =>  b(2),
                    c_in    =>  carry(1),
                    c_out   =>  carry(2),
                    sum     =>  sum(2));
```

Method 2: implicit port mapping

```
adder1: full_adder PORT MAP(a(1), b(1), c_in,     carry(1), sum(1));
adder2: full_adder PORT MAP(a(2), b(2), carry(1), carry(2), sum(2));
adder3: full_adder PORT MAP(a(3), b(3), carry(2), carry(3), sum(3));
adder4: full_adder PORT MAP(a(4), b(4), carry(3), c_out,    sum(4));
```

Method 3: use GENERATE statement to create multiple component

```
architecture…
    signal carry : std_logic_vector(0 to width); -- include carry in and out in signal
    component full_adder…                    -- declare full adder component
begin
    carry(0)    <= c_in;     -- Assign carry input and output ports to carry signal
    c_out       <= carry(width);

    adders: for i in 1 to width generate
        adder: full_adder port map (a(i), b(i), carry(i-1), carry(i), sum(i));
    end generate;
end [architecture name];
```