Design_Organization.parametrization.passing_values

```
ARCHITECTURE iterative OF nibble_comparator IS
  COMPONENT comp1
    GENERIC (tplh1 : TIME := 2 NS; tplh2 : TIME := 3 NS;
           tplh3 : TIME := 4 NS;  tphl1 : TIME := 4 NS;
           tphl2 : TIME := 5 NS; tphl3 : TIME := 6 NS);
    PORT (a, b, gt, eq, lt : IN BIT; a_gt_b, a_eq_b, a_lt_b : OUT BIT);
  END COMPONENT;
  FOR ALL : comp1 USE ENTITY WORK.bit_comparator_t (passed_delay);
  SIGNAL im : BIT_VECTOR ( 0 TO 8);
BEGIN
  c0: comp1 PORT MAP (a(0), b(0), gt, eq, lt, im(0), im(1), im(2));
  c1to2: FOR i IN 1 TO 2 GENERATE
    c: comp1 PORT MAP (a(i), b(i), im(i*3-3), im(i*3-2), im(i*3-1),
                 im(i*3+0), im(i*3+1), im(i*3+2) );
  END GENERATE;
  c3: comp1 PORT MAP (a(3), b(3), im(6), im(7), im(8),
                a_gt_b, a_eq_b, a_lt_b);
END iterative;
```

- Using *bit_comparator_t* in a larger design
- Specification of generic parameters is required
- Exclusion of Generic Map leaves all parameters OPEN
- Association with OPEN causes default values to be used

Design_Organization.Passing_values.OPEN

```
ARCHITECTURE iterative OF nibble_comparator IS
   . . .
BEGIN
 c0: comp1
         GENERIC MAP (OPEN, OPEN, 8 NS, OPEN, OPEN, 10 NS)
         PORT MAP (a(0), b(0), gt, eq, lt, im(0), im(1), im(2));
   . . .
 END iterative;
```

```
ARCHITECTURE iterative OF nibble_comparator IS
   . . .
BEGIN
 c0: comp1
         GENERIC MAP (tplh3 =>  8 NS, tphl3 => 10 NS)
         PORT MAP (a(0), b(0), gt, eq, lt, im(0), im(1), im(2));
   . . .
 END iterative;
```

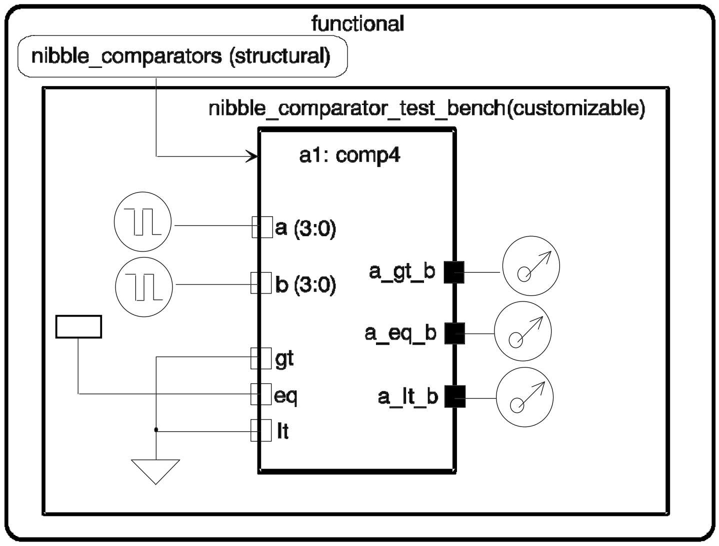
- A Generic Map can specify only some of the parameters
- Using OPEN causes use of default values
- Alternatively, association by name can be used
- Same applies to Port Map

Design_Organization.design_configuration

```
USE WORK.basic_utilities.ALL;
ARCHITECTURE customizable OF nibble_comparator_test_bench IS
  COMPONENT comp4 PORT (
    a, b : IN bit_vector (3 DOWNTO 0); gt, eq, lt : IN BIT;
    a_gt_b, a_eq_b, a_lt_b : OUT BIT);
  END COMPONENT;
  SIGNAL a, b : BIT_VECTOR (3 DOWNTO 0);
  SIGNAL eql, lss, gtr : BIT;
  SIGNAL vdd : BIT := '1';
  SIGNAL gnd : BIT := '0';
BEGIN
  a1: comp4 PORT MAP (a, b, gnd, vdd, gnd, gtr, eql, lss);
  apply_data (a, 00&15&15&14&14&14&14&10&00&15&00&00&15, 500 NS);
  apply_data (b, 00&14&14&15&15&12&12&12&15&15&15&00&00, 500 NS);
END customizable;
```

- Will present examples for configuration of unbound designs
- A general purpose testbench
- The *comp4* component is unbound
- Configure it to test any of the previous nibble_comparators

Design_Organization.configuration.testbench

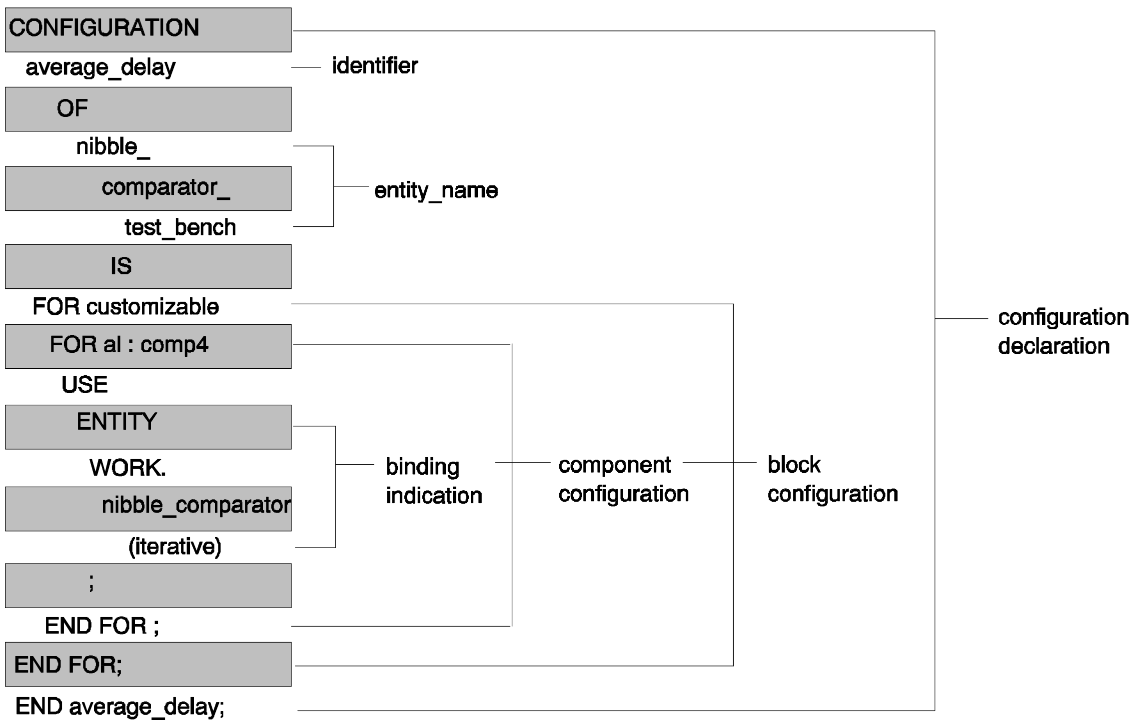



```
CONFIGURATION functional OF nibble_comparator_test_bench IS
  FOR customizable
    FOR a1 : comp4
      USE ENTITY WORK.nibble_comparator (structural);
    END FOR;
  END FOR;
END functional;

CONFIGURATION average_delay OF nibble_comparator_test_bench IS
  FOR customizable
    FOR a1 : comp4
      USE ENTITY WORK.nibble_comparator (iterative);
    END FOR;
  END FOR;
END average_delay;
```

- Configuration of testbench requires binding of *a1:comp4*
- Bind *a1:comp4* to a *nibble_comparator* architechture
- Testbench configured for
  1) Structural architecture of *nibble_comparator*
  2) Iterative architecture of *nibble_comparator*

Design_Organization.configuration.syntax



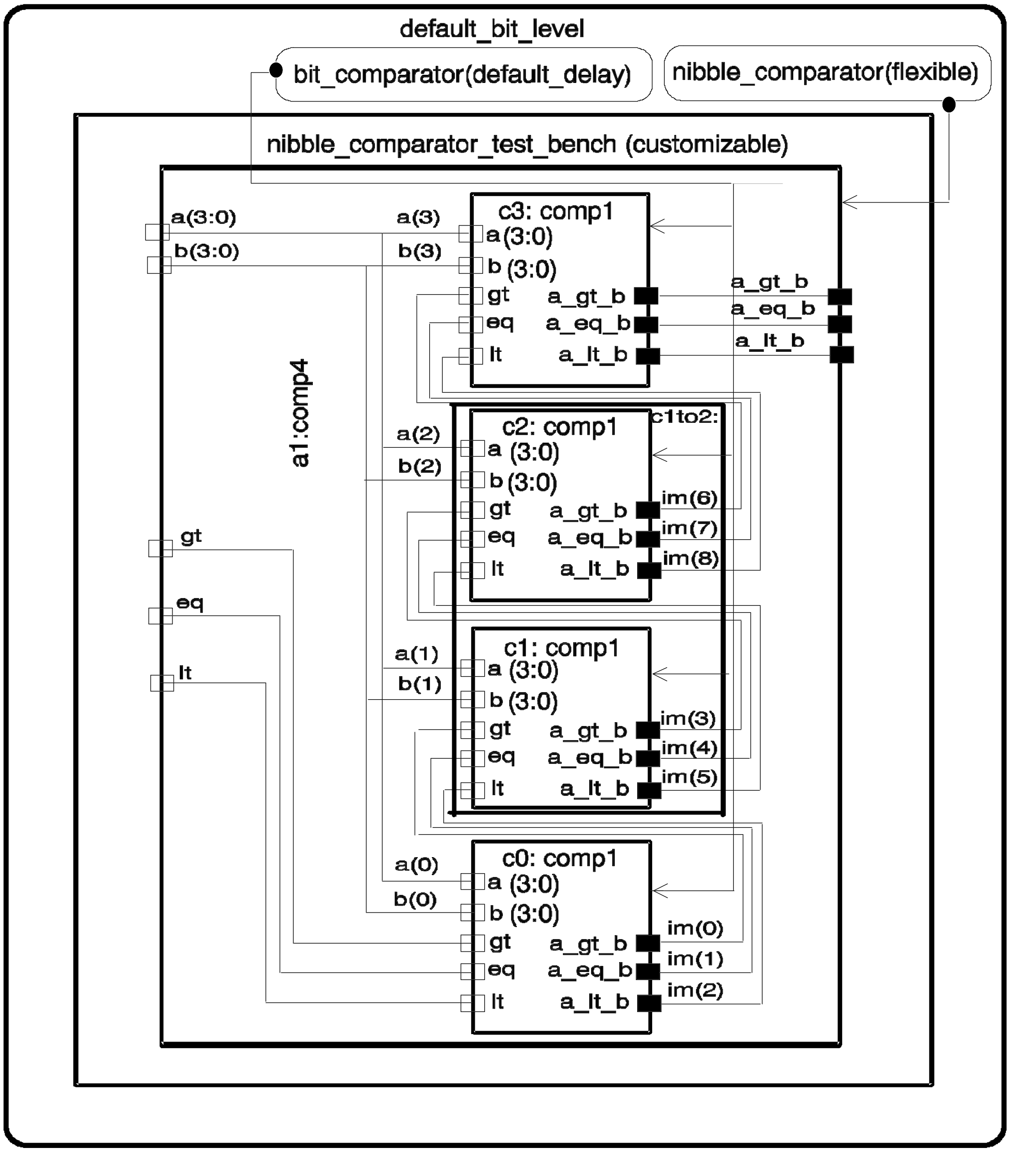
- Syntax details of configuration declaration
- Block configuration gives visibility
- Component configuration configures a component

Design_Organization.configuration.nesting

```
ARCHITECTURE flexible OF nibble_comparator IS
  COMPONENT comp1
    PORT (a, b, gt, eq, lt : IN BIT; a_gt_b, a_eq_b, a_lt_b : OUT BIT);
  END COMPONENT;
  SIGNAL im : BIT_VECTOR ( 0 TO 8 );
BEGIN
  c0: comp1 PORT MAP (a(0), b(0), gt, eq, lt, im(0), im(1), im(2));
  c1to2: FOR i IN 1 TO 2 GENERATE
    c: comp1 PORT MAP (a(i), b(i), im(i*3-3), im(i*3-2), im(i*3-1),
                       im(i*3+0), im(i*3+1), im(i*3+2) );
  END GENERATE;
  c3: comp1 PORT MAP (a(3), b(3), im(6), im(7), im(8),
                      a_gt_b, a_eq_b, a_lt_b);
END flexible;
```

- Will demonstrate nesting by two levels of flexibility
- Flexible *nibble_comparator*
- No particular *bit_comparator* is specified
- Using *customizable* testbench creates flexibility for:
  1) Specifying *nibble_comparator* for testbench
  2) Specifying *bit_comparator* for *nibble_comparator*

Design_Organization.configuration.nesting



- Nested configuration binds *nibble* and *bit* comparators

Design_Organization.configuration.nesting

```
USE WORK.ALL;
CONFIGURATION default_bit_level OF nibble_comparator_test_bench IS
  FOR customizable
    FOR a1 : comp4
      USE ENTITY WORK.nibble_comparator(flexible);
      FOR flexible
        FOR c0, c3: comp1
          USE ENTITY WORK.bit_comparator (default_delay);
        END FOR;
        FOR c1to2
          FOR c: comp1
            USE ENTITY WORK.bit_comparator (default_delay);
          END FOR;
        END FOR;
      END FOR;
    END FOR;
  END FOR;
END default_bit_level;
```

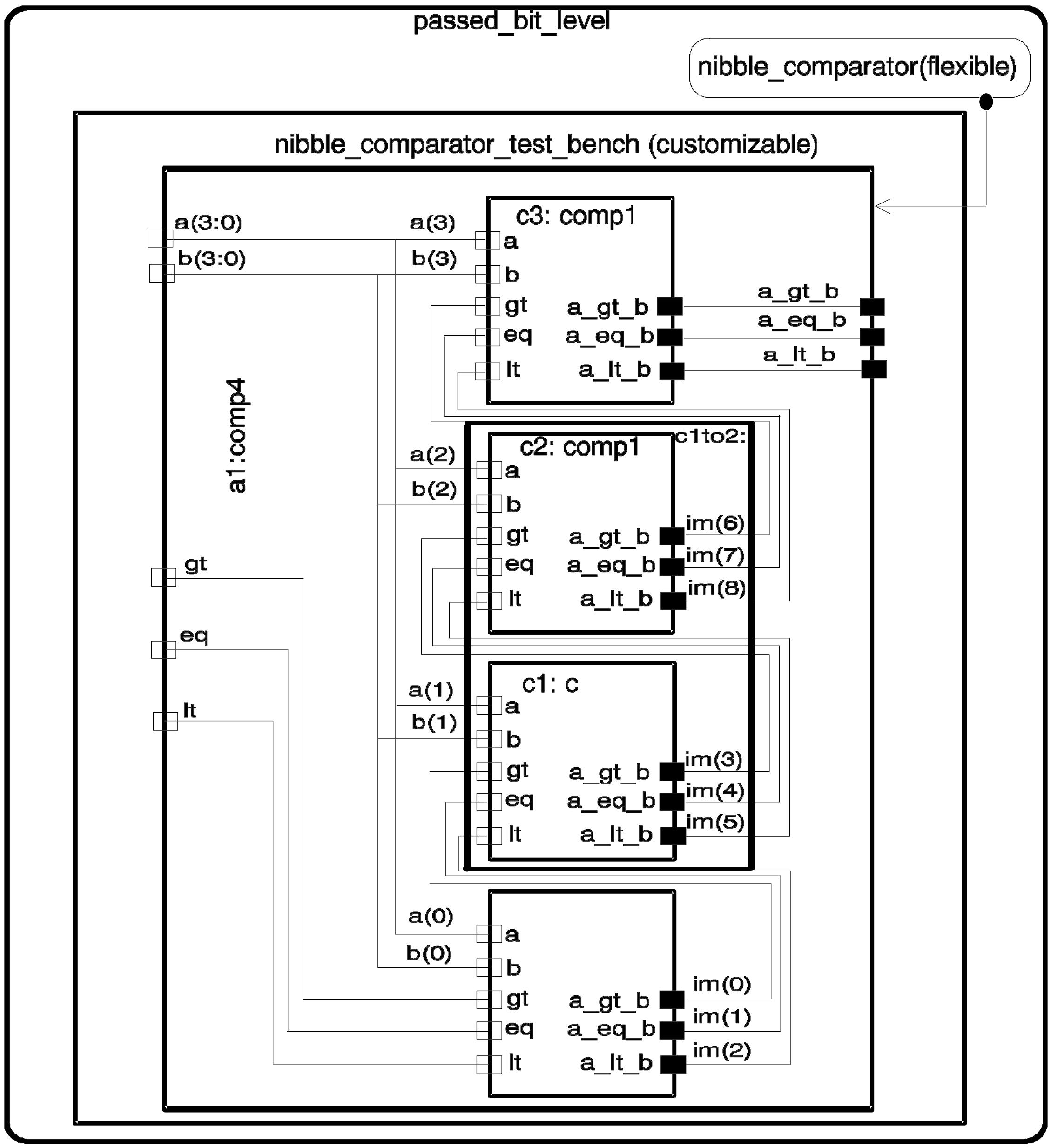
- Configuring customizable testbench for testing
  default_delay bit_comparator
  instantiated through
  flexible nibble_comparator
- Visibility is obtained before component configuration is done

Design_Organization.configuration.nesting

```
USE WORK.ALL;
CONFIGURATION fixed_bit_level OF nibble_comparator_test_bench IS
  FOR customizable
    FOR a1 : comp4
      USE ENTITY WORK.nibble_comparator(flexible);
      FOR flexible
        FOR c0, c3: comp1
          USE ENTITY WORK.bit_comparator (fixed_delay);
        END FOR;
        FOR c1to2
          FOR c: comp1
            USE ENTITY WORK.bit_comparator (fixed_delay);
          END FOR;
        END FOR;
      END FOR;
    END FOR;
  END FOR;
END fixed_bit_level;
```

- Configuring customizable testbench for testing
  fixed_delay bit_comparator
  instantiated through
  flexible nibble_comparator
- Visibility is obtained before component configuration is done

Design_Organization.configuration.generics



ERROR: undefined
OFFENDING COMMAND: 13initia

STACK: