تصميم النظم المنطقية باستخدام الدارات المنطقية المبرمجة

المحاضرة السابعة

د.م. خولة حموي
khawla.hamwi@gmail.com

العام الدراسي: 2023-2024

جَامعة المَنارة
MANARA UNIVERSITY

عناوين المحاضرة

- البرامج الجزئية

1. الوظيفة Function
2. الإجرائية Procedure

الوظيفة Function

تتيح لغة VHDL نوعين من البرامج الجزئية وهي الوظيفة FUNCTION و الإجرائية PROCEDURE.

تستخدم بشكل أساسي لتعريف وظائف و إجرائيات مسبقة التعريف.

تكتب الصيغة التفصيلية للوظيفة على النحو التالي:

```
FUNCTION function_name [<parameter list>] RETURN data_type IS
    [declarations]
BEGIN
    (sequential statements)
END function_name;
```

يستخدم لائحة البارامترات لتخصيص مداخل للوظيفة ولا يسمح في حالة الوظيفة إلا باستخدام بارامتر Constant و Signal

⟨parameter list⟩ = [CONSTANT] constant_name: constant_type; or

⟨parameter list⟩ = SIGNAL signal_name: signal_type;

الوظيفة Function

مثال: وصف دارة القلاب D باستخدام وظيفة لوصف الحافة الصاعدة الموجبة لنبضة الساعة

```
------ Function body: -------------------------------
FUNCTION positive_edge(SIGNAL s: STD_LOGIC) RETURN BOOLEAN IS
BEGIN
   RETURN (s'EVENT AND s='1');
END positive_edge;
------ Function call: -------------------------------
...
IF positive_edge(clk) THEN...
...
-----------------------------------------------------
```

الإجرائية Procedure

تكتب الصيغة التفصيلية للوظيفة على النحو التالي:

```
PROCEDURE procedure_name [<parameter list>] IS
    [declarations]
BEGIN
    (sequential statements)
END procedure_name;
```

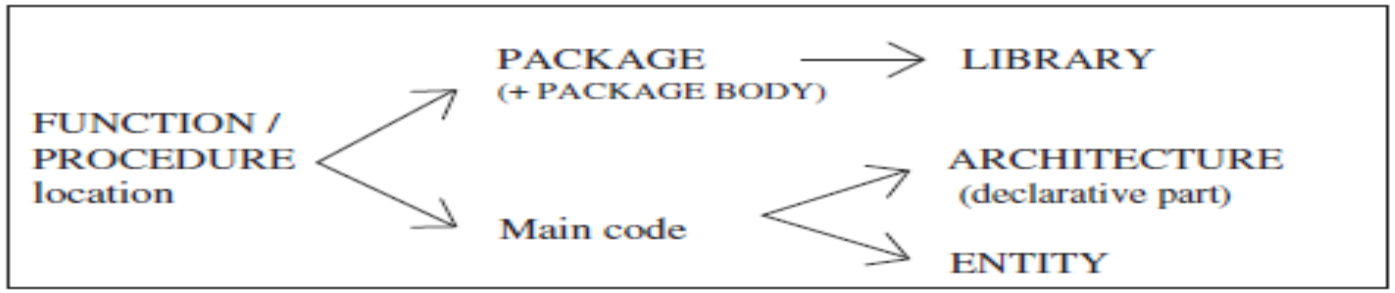
تستخدم البارمترات لتخصيص مداخل الإجرائية ومخارجها وهي

⟨parameter list⟩ = [CONSTANT] constant_name: mode type;

⟨parameter list⟩ = SIGNAL signal_name: mode type; or

⟨parameter list⟩ = VARIABLE variable_name: mode type;

موقع الوظيفة أو الإجرائية



أمثلة:

وظيفة موجودة في البرنامج الرئيسي :

```
1  ----------------------------------------
2  LIBRARY ieee;
3  USE ieee.std_logic_1164.all;
4  ----------------------------------------
5  ENTITY dff IS
6     PORT ( d, clk, rst: IN STD_LOGIC;
7            q: OUT STD_LOGIC);
8  END dff;
9  ----------------------------------------
10 ARCHITECTURE my_arch OF dff IS
11 ----------------------------------------
12    FUNCTION positive_edge(SIGNAL s: STD_LOGIC)
13       RETURN BOOLEAN IS
14    BEGIN
15       RETURN s'EVENT AND s='1';
16    END positive_edge;
17 ----------------------------------------
18 BEGIN
19    PROCESS (clk, rst)
20    BEGIN
21       IF (rst='1') THEN q <= '0';
22       ELSIF positive_edge(clk) THEN q <= d;
23       END IF;
24    END PROCESS;
25 END my_arch;
26 ----------------------------------------
```

أمثلة:

وظيفة موجودة في حزمة :

```
1  ------- Package: ----------------------------
2  LIBRARY ieee;
3  USE ieee.std_logic_1164.all;
4  ----------------------------------------------
5  PACKAGE my_package IS
6     FUNCTION positive_edge(SIGNAL s: STD_LOGIC) RETURN BOOLEAN;
7  END my_package;
8  ----------------------------------------------
9  PACKAGE BODY my_package IS
10    FUNCTION positive_edge(SIGNAL s: STD_LOGIC)
11       RETURN BOOLEAN IS
12     BEGIN
13        RETURN s'EVENT AND s='1';
14     END positive_edge;
15 END my_package;
16 ----------------------------------------------
```

```
1  ------ Main code: ----------------------------
2  LIBRARY ieee;
3  USE ieee.std_logic_1164.all;
4  USE work.my_package.all;
5  ----------------------------------------------
6  ENTITY dff IS
7     PORT ( d, clk, rst: IN STD_LOGIC;
8              q: OUT STD_LOGIC);
9  END dff;
10 ----------------------------------------------
11 ARCHITECTURE my_arch OF dff IS
12 BEGIN
13    PROCESS (clk, rst)
14    BEGIN
15       IF (rst='1') THEN q <= '0';
16       ELSIF positive_edge(clk) THEN  q <= d;
17       END IF;
18    END PROCESS;
19 END my_arch;
20 ----------------------------------------------
```

أمثلة:

إجرائية موجودة في البرنامج الرئيسي:

```
1  ------------------------------------------------
2  LIBRARY ieee;
3  USE ieee.std_logic_1164.all;
4  ------------------------------------------------
5  ENTITY min_max IS
6     GENERIC (limit : INTEGER := 255);
7     PORT ( ena: IN BIT;
8            inp1, inp2: IN INTEGER RANGE 0 TO limit;
9            min_out, max_out: OUT INTEGER RANGE 0 TO limit);
10 END min_max;
11 ------------------------------------------------
12 ARCHITECTURE my_architecture OF min_max IS
13    -------------------------
14    PROCEDURE sort (SIGNAL in1, in2: IN INTEGER RANGE 0 TO limit;
15       SIGNAL min, max: OUT INTEGER RANGE 0 TO limit) IS
16    BEGIN
17       IF (in1 > in2) THEN
18          max <= in1;
19          min <= in2;
20       ELSE
21          max <= in2;
22          min <= in1;
23       END IF;
24    END sort;
25    -------------------------
26 BEGIN
27    PROCESS (ena)
28    BEGIN
29       IF (ena='1') THEN sort (inp1, inp2, min_out, max_out);
30       END IF;
31    END PROCESS;
32 END my_architecture;
33 ------------------------------------------------
```

أمثلة:

إجرائية موجودة في حزمة:

```
1  ------------ Package: -------------------------
2  LIBRARY ieee;
3  USE ieee.std_logic_1164.all;
4  ----------------------------------
5  PACKAGE my_package IS
6     CONSTANT limit: INTEGER := 255;
7     PROCEDURE sort (SIGNAL in1, in2: IN INTEGER RANGE 0 TO limit;
8        SIGNAL min, max: OUT INTEGER RANGE 0 TO limit);
9  END my_package;
10 ----------------------------------
11 PACKAGE BODY my_package IS
12    PROCEDURE sort (SIGNAL in1, in2: IN INTEGER RANGE 0 TO limit;
13       SIGNAL min, max: OUT INTEGER RANGE 0 TO limit) IS
14    BEGIN
15       IF (in1 > in2) THEN
16          max <= in1;
17          min <= in2;
18       ELSE
19          max <= in2;
20          min <= in1;
21       END IF;
22    END sort;
23 END my_package;
24 ---------------------------------------------
```

أمثلة:

إجرائية موجودة في حزمة:

```
1  --------- Main code: ----------------------
2  LIBRARY ieee;
3  USE ieee.std_logic_1164.all;
4  USE work.my_package.all;
5  ------------------------------------
6  ENTITY min_max IS
7     GENERIC (limit: INTEGER := 255);
8     PORT ( ena: IN BIT;
9            inp1, inp2: IN INTEGER RANGE 0 TO limit;
10           min_out, max_out: OUT INTEGER RANGE 0 TO limit);
11 END min_max;
12 ------------------------------------

13 ARCHITECTURE my_architecture OF min_max IS
14 BEGIN
15    PROCESS (ena)
16    BEGIN
17       IF (ena='1') THEN sort (inp1, inp2, min_out, max_out);
18       END IF;
19    END PROCESS;
20 END my_architecture;
21 ----------------------------------------
```