Integrated Circuits Design by FPGA

م.م. أحمد مؤيد عبدالحسين
جامعة الفرات الأوسط التقنية / الكلية التقنية الهندسية / نجف

Lecture 13

Procedure

Objectives of this Lecture

- To define **Procedure**
- To implement **Procedure** in examples design.

Contents of this Lecture

- Introduction
- Procedure
- Procedure Location
- FUNCTION versus PROCEDURE Summary

Introduction

- **Functions** and **Procedures** are collectively called subprograms.
- From a construction point of view, they are very similar to a PROCESS. For they are the only pieces of sequential VHDL code, and thus employ the same sequential statements seen there (IF, CASE, and LOOP; WAIT is not allowed).
- However, from the applications point of view, there is a fundamental difference between a **PROCESS** and a **FUNCTION** or **PROCEDURE**. While the first is intended for immediate use in the main code, the others are intended mainly for LIBRARY allocation, that is, their purpose is to store commonly used pieces of code, so they can be reused or shared by other projects.

Procedure

- A **PROCEDURE** is very similar to a FUNCTION and has the same basic purposes. However, a procedure can return more than one value.
- Like a FUNCTION, two parts are necessary to construct and use a PROCEDURE: the procedure itself **(procedure body)** and **a procedure call.**

Procedure Body

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

In the syntax above, `<parameter list>` specifies the procedure's input and output parameters; that is:

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

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

⟨parameter list⟩ = VARIABLE variable_name: mode type;

Procedure

- A **PROCEDURE** can have any number of IN, OUT, or INOUT parameters, which can be SIGNALS, VARIABLES, or CONSTANTS.
- For input signals (mode IN), the default is CONSTANT, whereas for output signals (mode OUT or INOUT) the default is VARIABLE.
- As seen before, WAIT, SIGNAL declarations, and COMPONENTS are not synthesizable when used in a FUNCTION. The same is true for a **PROCEDURE**, the exception that a SIGNAL can be declared, but then the PROCEDURE must be declared in a PROCESS.

```
PROCEDURE my_procedure ( a: IN BIT; SIGNAL b, c: IN BIT;
                         SIGNAL x: OUT BIT_VECTOR(7 DOWNTO 0);
                         SIGNAL y: INOUT INTEGER RANGE 0 TO 99) IS
BEGIN
   ...
END my_procedure;
```

Procedure

Procedure Call

Contrary to a FUNCTION, which is called as part of an expression, a PROCEDURE call is a statement on its own. It can appear by itself or associated to a statement (either concurrent or sequential).

Examples of procedure calls:

```
compute_min_max(in1, in2, 1n3, out1, out2);
                -- statement by itself

divide(dividend, divisor, quotient, remainder);
                -- statement by itself

IF (a>b) THEN compute_min_max(in1, in2, 1n3, out1, out2);
                -- procedure call associated to another statement
```

Procedure Location

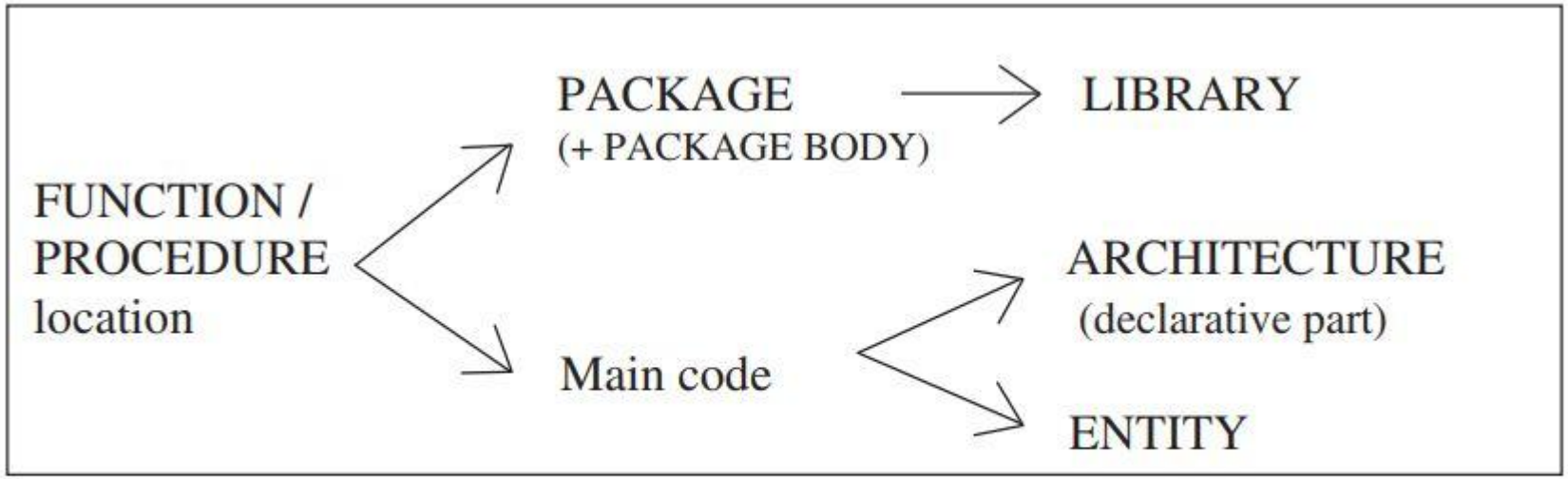

Figure 11.1
Typical locations of a FUNCTION or PROCEDURE.

Procedure Location

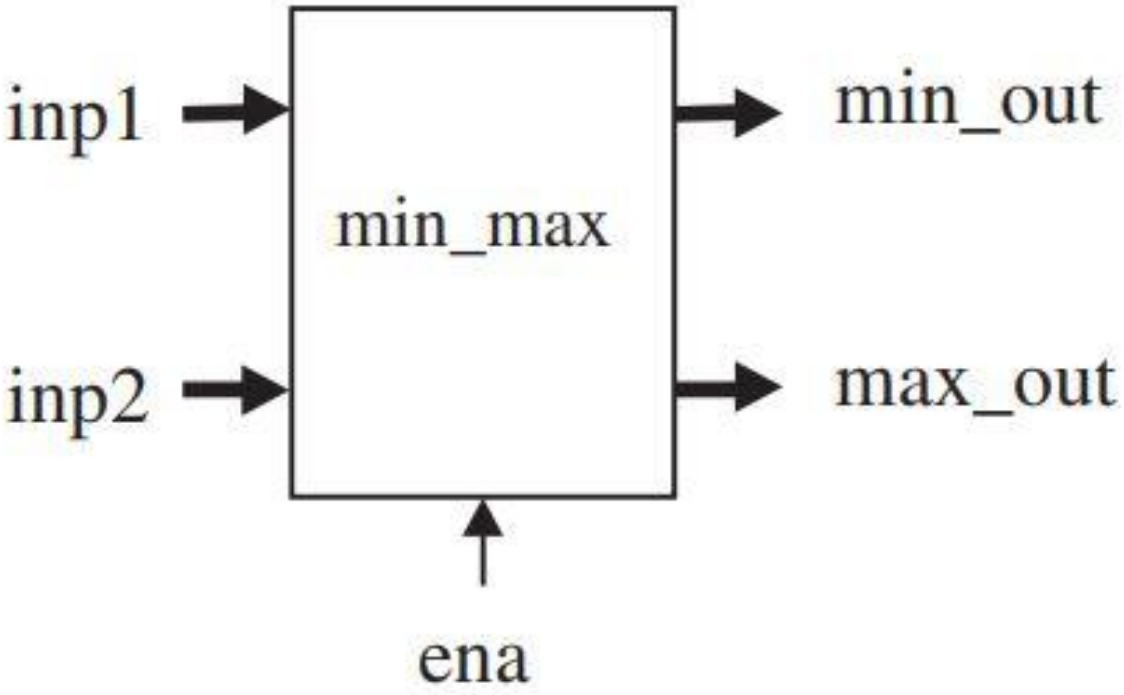

Figure 11.5
min_max circuit of example 11.9.

Procedure

Example 11.9: PROCEDURE Located in the Main Code

```
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 ------------------------------------------------
```

```
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;
```

Procedure

Example 11.10: PROCEDURE Located in a Package

```
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;
```

```
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 ------------------------------------------
```

Procedure

Example 11.10: PROCEDURE Located in a Package

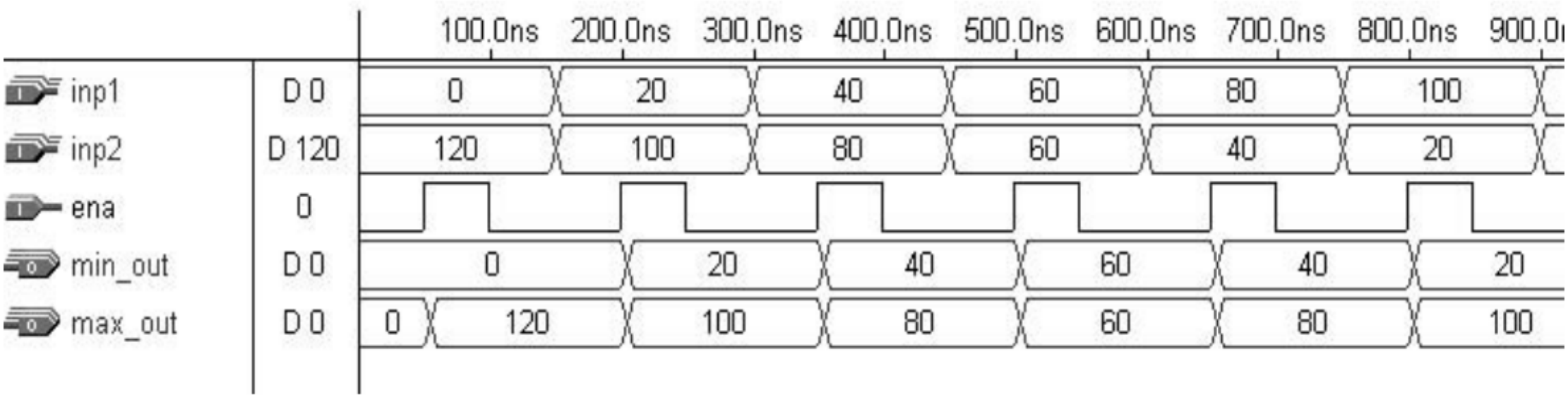

Figure 11.6
Simulation results of example 11.9.

FUNCTION versus PROCEDURE Summary

- A FUNCTION has zero or more input parameters and a single return value. The input parameters can only be CONSTANTS (default) or SIGNALS (VARIABLES are not allowed).
- A PROCEDURE can have any number of IN, OUT, and INOUT parameters, which can be SIGNALS, VARIABLES, or CONSTANTS. For input parameters (mode IN) the default is CONSTANT, whereas for output parameters (mode OUT or INOUT) the default is VARIABLE.
- A FUNCTION is called as part of an expression, while a PROCEDURE is a statement on its own.
- In both, WAIT and COMPONENTS are not synthesizable.
- The possible locations of FUNCTIONS and PROCEDURES are the same (figure 11.1). Though they are usually placed in PACKAGES (for code partitioning, code sharing, and code reuse purposes), they can also be located in the main code.

Assignments

The assignments will be attached to your class room.

- Problem 11.7: **Statistical Procedure**

End of lecture 13
Any Questions ?