Subprograms

- *A subprogram defines a sequential algorithm that performs a certain computation and executes in zero simulation time. There are two kinds of subprograms:*
- 1. *Functions: These are usually used for computing a single value.*
- 2. *Procedures: These are used to partition large behavioral descriptions. Procedures can return zero or more values.*

Subprograms cont..

- A subprogram is defined using a *subprogram body. The typical format for a subprogram body is*

subprogram-specification ***is***

   subprogram-item-declarations

begin

subprogram-statements

end [*subprogram-name];*

Subprograms cont..

- The *subprogram-specification specifies the name of a subprogram and defines its interface, that is, it defines the formal parameter names, their class (i.e., signal, variable, or constant), their type, and their mode (whether they are in, out, or inout).*

Functions

- Functions are used to describe frequently used sequential algorithms that return a single value.
- This value is returned to the calling program using a return statement. Some of their common uses are as resolution functions.

Functions cont..

- The general syntax of a subprogram specification for a function body is function

function-name (parameter-list) ***return return-type is***

Begin

sequential statements

return return value

End *function-name;*

Functions cont..

A function call has the form-

function-name (list-of-actual-values)

Types of function-

1.Pure function:-*It return the same value whenever it is called.*

2. Impure function:-*It can return different values each time when it is called.*

Example of Function

- This function adds two 4 bit vectors and return a 4 bit sum.

```
Function ADD(A,B:bit_vector(0 to 3))
Return bit_vector is
Variable cout:bit;
Variable cin:bit:='0';
Begin
  for i in 0 to 3 loop
```

Example of Function cont..

```
Cout:= (A(i) and B(i)) or (A(i) and cin) or
   (B(i) and cin);
Sum(i):=A(i) xor B(i) xor cin;
Cin:=Cout;
End loop;
Return sum;
End Add;
```

Procedures

- Procedures allow decomposition of large behaviors into modular sections. In contrast to a function, a procedure can return zero or more values using parameters of mode out and inout. The syntax for the subprogram specification for a procedure body is
- **procedure** ***procedure-name (parameter-list)***
- Parameters may be constants, variables, or signals and their modes may be in, out, or inout.

Example of **Procedures**

```
type OP_CODE is (ADD, SUB, MUL, DIV,
   LT, LE, EQ);
. . .
procedure ARITH_UNIT (A, B: in
   INTEGER; OP: in OP_CODE;
Z: out INTEGER; ZCOMP: out
   BOOLEAN) is
begin
case OP is
when ADD=>Z:=A+B;
when SUB=>Z:=A-B;
when MUL=>Z:=A*B;
when DIV => Z := A/B;
when LT => ZCOMP := A < B;
when LE => ZCOMP := A <= B;
when EQ => ZCOMP := A = B;
end case;
end ARITH_UNIT;
```