Process Statement

- A process statement contains sequential statements that describe the functionality of a portion *of an entity in sequential terms. The syntax of a process statement is*
- [*process-label:]* ***process [(sensitivity-list)]***

begin

  - *sequential-statements; these are ->*
    - *variable-assignment-statement*
    - *signal-assignment-statement*

Process Statement Cont..

wait-statement

if-statement

case-statement

loop-statement

null-statement

- *exit-statement*
- *next-statement*
- *assertion-statement*
- *procedure-call-statement*
- *return-statement.*
- **end process [*process-label];**

Variable Assignment Statement

- Variables can be declared and used inside a process statement. A variable is assigned a value using the variable assignment statement that typically has the form
- *variable-object := expression;*

Variable Assignment Statement cont..

- Consider the following process statement.

```
process (A)
   variable EVENTS_ON_A: INTEGER := 0;
begin
   EVENTS_ON_A := EVENTS_ON_A+1;
end process;
```

Signal Assignment Statement

- Signals are assigned values using a signal assignment statement *The simplest form of a signal assignment statement is*
- *signal-object* ***<= expression [after delay-value];***
- A signal assignment statement can appear within a process or outside of a process. If it occurs outside of a process, it is considered to be a concurrent signal assignment statement.

Signal Assignment Statement cont..

- When a signal assignment statement appears within a process, it is considered to be a sequential signal assignment statement . When a signal assignment statement is executed, the value of the expression is computed and this value is scheduled to be assigned to the signal after the specified delay.

Signal Assignment Statement cont..

- If no after clause is specified, the delay is assumed to be a default delta delay.
- Some examples of signal assignment statements are
- COUNTER <= COUNTER+ "0010"; - Assign after a delta delay.
- PAR <= PAR **xor DIN after 12 ns;**
- Z <= (AO **and A1) or (BO and B1) or (CO and C1) after 6 ns;**

Wait Statement

- The **wait statement provides an alternate way to suspend the execution of a process. There are three basic forms of the wait statement.**
- **wait on *sensitivity-list;***
- **wait until *boolean-expression ;***
- **wait for *time-expression ;***

Wait Statement cont..

- They may also be combined in a single wait statement. For example,
- **wait on *sensitivity-list until boolean-expression for time-expression-,***
- Some examples of **wait statements are**
- **wait on A, B, C; -- statement 1**
- **wait until (A = B); -- statement 2**
- **wait for 10ns; -- statement 3**
- **wait on CLOCK for 20ns; -- statement 4**
- **wait until (SUM > 100) for 50 ms; -- statement 5**

If Statement

- An if statement selects a sequence of statements for execution based on the value of a condition. The condition can be any expression that evaluates to a boolean value. The general form of an if statement is

if *boolean-expression then* *sequential-statements*

[**elsif *boolean-expression then***

sequential-statements]

[**else** *sequential-statements]*

end if;