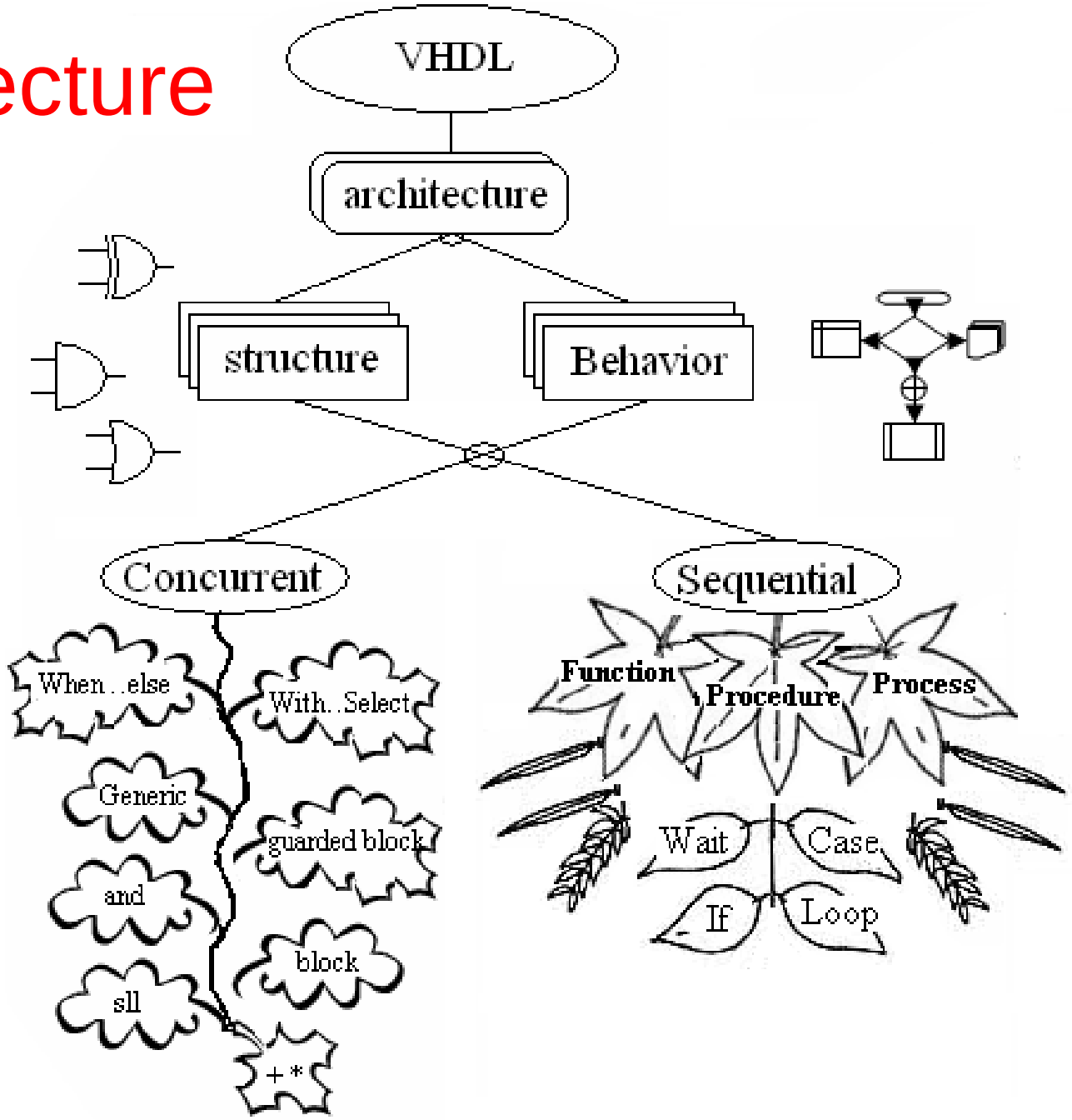
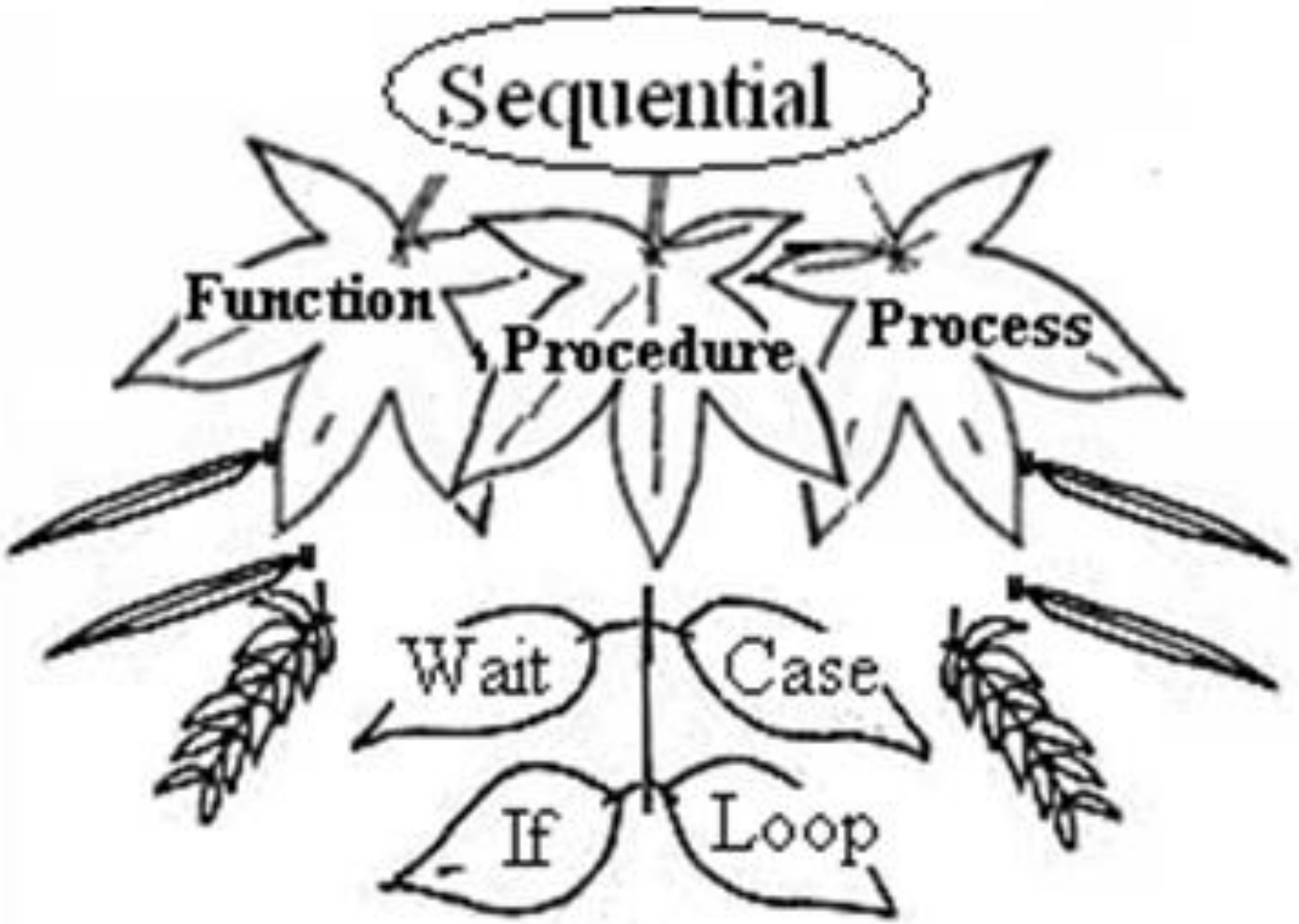
Sequential statements

By: M. SHIPLE

Architecture



sequnetial statements



"Process"

Syntax:

```
[label:] PROCESS (sensitivity list)
[VARIABLE_name: Variable_type [:= initial_value]];
BEGIN
(sequential code)
END PROCESS [label];
```

Example

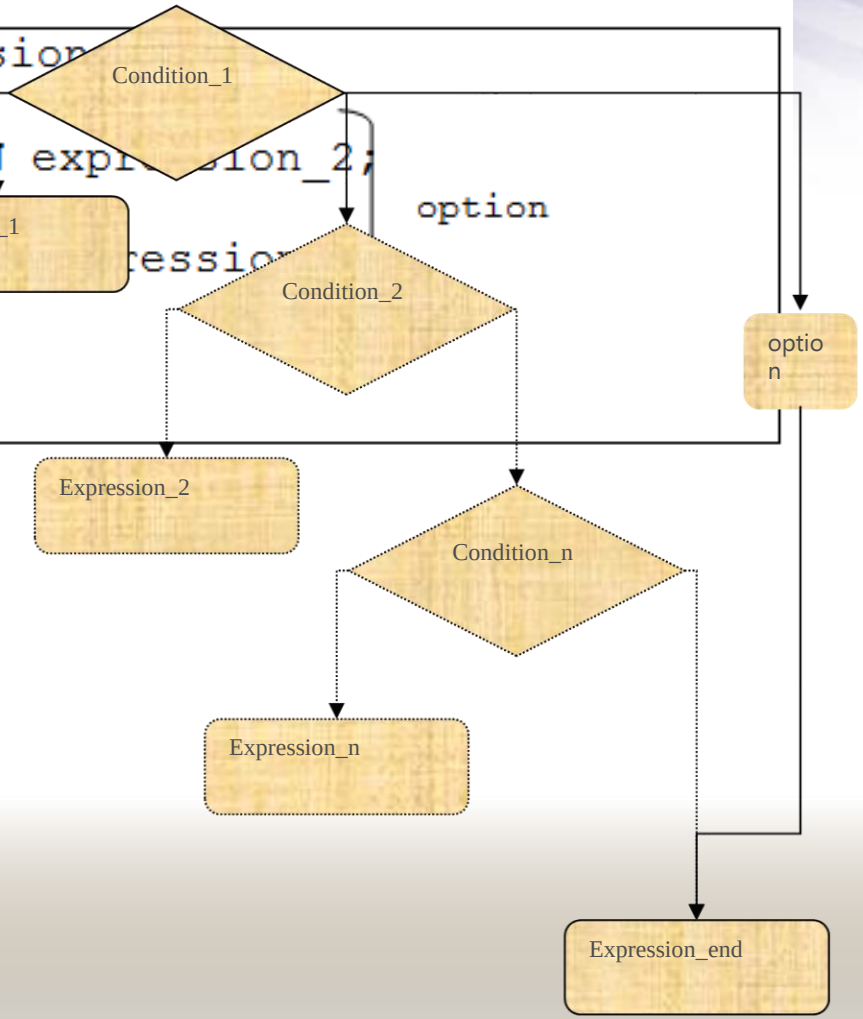
Syntax:

```
IF condition_1 THEN expression_1;

    [ ELSIF condition_2 THEN expression_2;
      |                 |                  ]  option
    [ ELSIF condition_n THEN expression_n;   ]

      ELSE expression_end;
END IF;
```

"With...select"

Syntax:

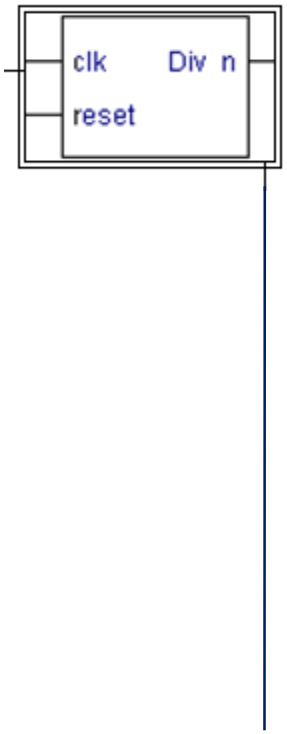
```
IF condition_1 THEN expression
    ELSIF condition_2 THEN expression_2;
    ELSIF condition_n THEN expression_n;
    ELSE expression_end;
END IF;
```



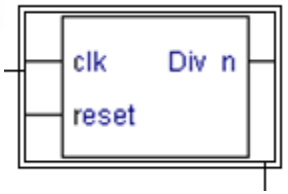
Blinking led

Example

```
begin
        process (clk,reset)
                        variable count:integer range 0 to 7 ;
        Begin
        if reset ='1' then
        Div_n <= '0';
                elsif rising_edge(clk) then

                        if count=0 then
                                Div_n <= not (Div_n);
                        end if;
                        count := count+1;
                        if count=n1 then
                                Div_n <= not (Div_n);
                                count := 0;
                        end if;
                end if;
        end process;
end Behavioral;library IEEE;
```



Exercise



Write a VHDL code for FORWARD-BACKWARD counter.

Write a VHDL code for FORWARD-BACKWARD counter.

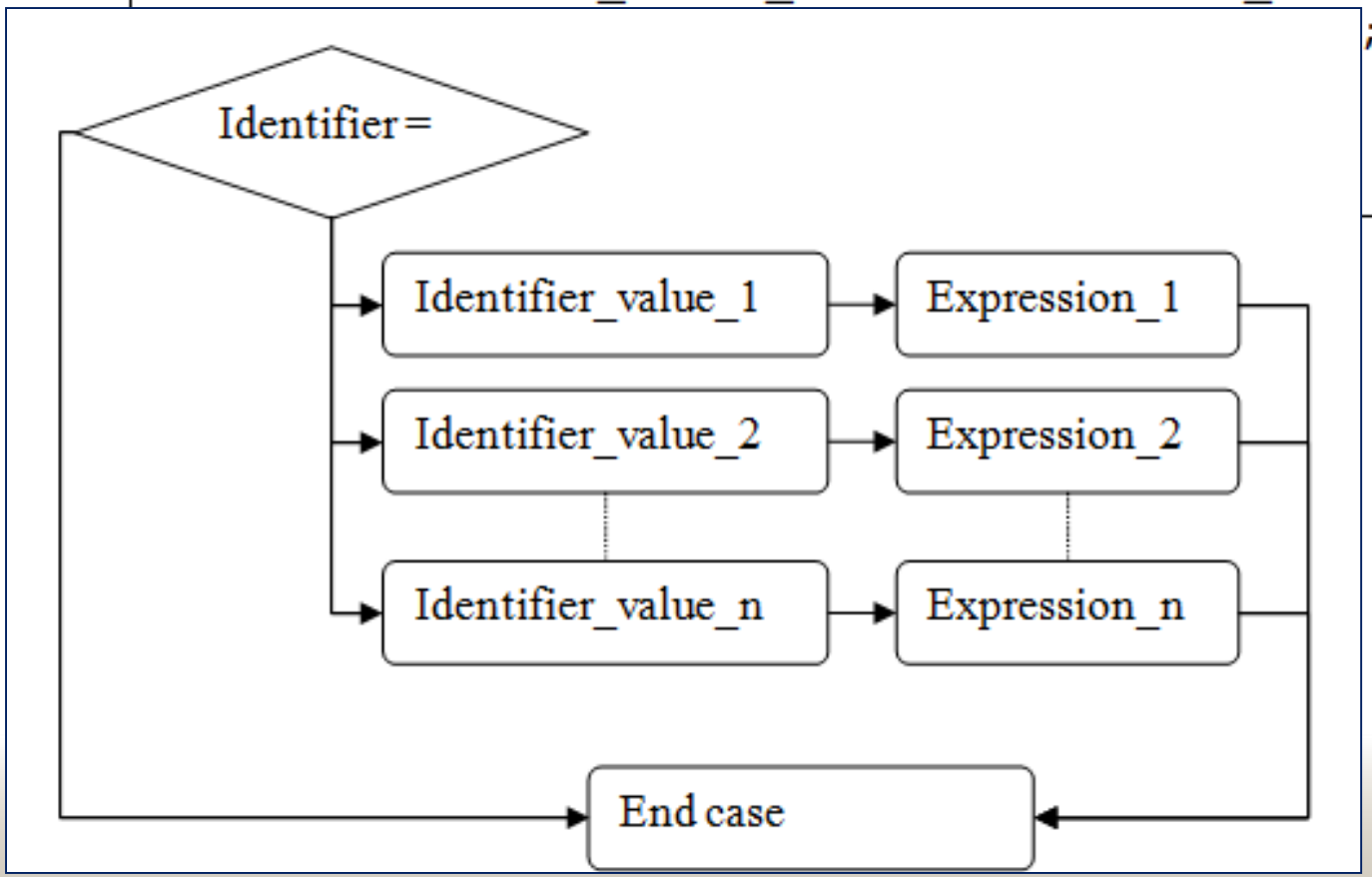
```
entity counter is
GENERIC (N : INTEGER := 5);
PORT ( reset,clock,count_direction : in std_logic;
       count : buffer INTEGER RANGE 0 to N);
end counter;

architecture Behavioral of counter is

begin
process (clock, reset)
   begin
      if reset='1' then
         count <= 0;
      elsif clock='1' and clock'event then
            if count_direction='1' then
               if count = N then
                  count <= 0;
               else
                  count <= count + 1;
               end if;
            else
               if count=0 then
                  count <= N;
               else
                  count <= count - 1;
               end if;
            end if;
      end if;
   end process;
   end Behavioral;
```

Exercise

"Case "

Syntax:

```
CASE identifier IS
WHEN  identifier_value_1 => expressions_1;
                                                    ;
```



Example

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity module6_case is
    Port ( clk : in std_logic;
           std_out : out std_logic_vector (3 downto 0));
end module6_case;

architecture Behavioral of module6_case is
        signal counter : integer range 0 to 9;
begin
  process (clk)
   begin
  if rising_edge(clk )then
  counter <= counter +1;
                        CASE counter IS
                                        WHEN 0 => std_out <= "0000";
                                        WHEN 1 => std_out <= "0001";
                                        WHEN 2 => std_out <= "0010";
                                        WHEN 3 => std_out <= "0011";
                                        WHEN 4 => std_out <= "0100";
                                        WHEN 5 => std_out <= "0101";
                                        WHEN 6 => std_out <= "0110";
                                        WHEN 7 => std_out <= "0111";
                                        WHEN others => std_out <= "1111";
                        END CASE;
        end if;
  end process;
end Behavioral;
```


END