UTS VLSI forum

VHDL for sequential circuit design

By

K. Vasu
Senior Application Engineer
VLSI & EH group
Unistring Tech Solutions Pvt. Ltd
# D10, 5th Floor, Eureka court, Beside Image hospitals, Ameerpet,
Hyderabad, ph:040-23732798, 09440318188
www.unistring.com

String Technologies
#309, Vederi complex, opp Dilshuk Nagar bus Depot, Dilsukh nagar, Hyderabad,
ph: 040 – 24151900
www.stringtechnologies.net

1. VHDL code is inherently concurrent
2. Process, function and Procedure are only three methods to write sequential statements.
3. Sequential code can be used for realizing sequential logic and also the combinational logic.
4. Sequential code is also called behavioral code
5. IF, WAIT, CASE and LOOP are sequential statements and can be used only inside a function, procedure or Process.

6. Variable is allowed only inside the sequential code only, so its scope is always local (signals scope is global)

7. The template of Process is given below

```
[label:] PROCESS (sensitivity list)
  [VARIABLE name type [range] [:= initial_value;]]
BEGIN
  (sequential code)
END PROCESS [label];
```

A process must have a sensitivity list or a wait statement

IF statement

```
IF conditions THEN assignments;
ELSIF conditions THEN assignments;
...
ELSE assignments;
END IF;
```

Example:

```
IF (x<y) THEN temp:="11111111";
ELSIF (x=y AND w='0') THEN
    temp:="11110000";
ELSE temp:=(OTHERS =>'0');
```

Example : DFF with asynchronous reset (behavioral modeling)

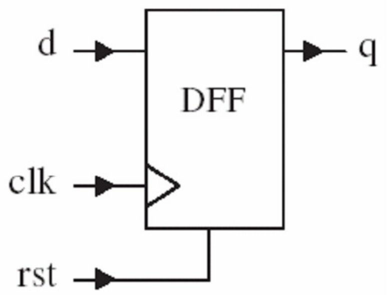
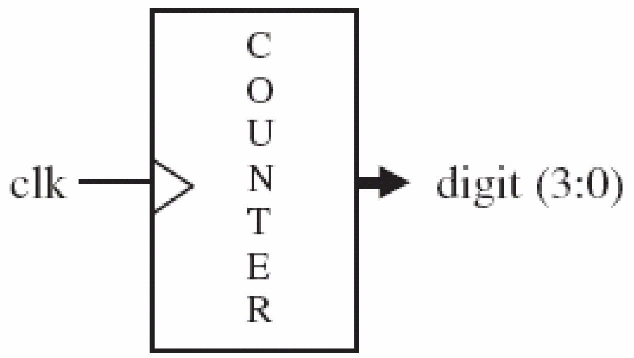


Figure
DFF with asynchronous reset.

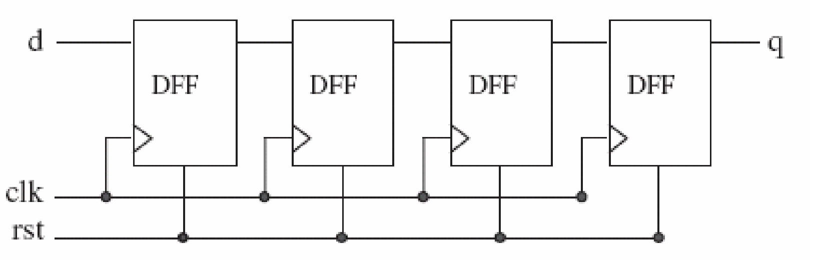
```
---------------------------------------
LIBRARY ieee;
USE ieee.std_logic_1164.all;
---------------------------------------
ENTITY dff IS
   PORT ( d, clk, rst: IN STD_LOGIC;
          q: OUT STD_LOGIC);
END dff;
---------------------------------------
ARCHITECTURE behavior OF dff IS
BEGIN
   PROCESS (rst, clk)
   BEGIN
      IF (rst='1') THEN
         q <= '0';
      ELSIF (clk'EVENT AND clk='1') THEN
         q <= d;
      END IF;
   END PROCESS;
END behavior;
---------------------------------------
```

Example : One digit counter



```
ENTITY counter IS
   PORT (clk : IN STD_LOGIC;
         digit : OUT INTEGER RANGE 0 TO 9);
END counter;
----------------------------------------------
ARCHITECTURE counter OF counter IS
BEGIN
   count: PROCESS(clk)
      VARIABLE temp : INTEGER RANGE 0 TO 10;
   BEGIN
      IF (clk'EVENT AND clk='1') THEN
         temp := temp + 1;
         IF (temp=10) THEN temp := 0;
         END IF;
      END IF;
      digit <= temp;
   END PROCESS count;
END counter;
```

Example : Serial In Serial Out Shift Register



```
ENTITY shiftreg IS
   GENERIC (n: INTEGER := 4);    -- # of stages
   PORT (d, clk, rst: IN STD_LOGIC;
         q: OUT STD_LOGIC);
END shiftreg;
--------------------------------------------------
ARCHITECTURE behavior OF shiftreg IS
   SIGNAL internal: STD_LOGIC_VECTOR (n-1 DOWNTO 0);
BEGIN
   PROCESS (clk, rst)
   BEGIN
      IF (rst='1') THEN
         internal <= (OTHERS => '0');
      ELSIF (clk'EVENT AND clk='1') THEN
         internal <= d & internal(internal'LEFT DOWNTO 1);
      END IF;
   END PROCESS;
   q <= internal(0);
END behavior;
```

Three forms of WAIT statement

```
WAIT UNTIL signal_condition;
```

```
WAIT ON signal1 [, signal2, ... ];
```

```
WAIT FOR time;
```

WAIT UNTIL can accept only one statement

Example: 8-bit register with synchronous reset.

```
PROCESS          -- no sensitivity list
BEGIN
   WAIT UNTIL (clk'EVENT AND clk='1');
   IF (rst='1') THEN
      output <= "00000000";
   ELSIF (clk'EVENT AND clk='1') THEN
      output <= input;
   END IF;
END PROCESS;
```

WAIT ON can consists of multiple signals.

```
ENTITY dff IS
   PORT (d, clk, rst: IN STD_LOGIC;
         q: OUT STD_LOGIC);
END dff;
---------------------------------------
ARCHITECTURE dff OF dff IS
BEGIN
   PROCESS
   BEGIN
      WAIT ON rst, clk;
      IF (rst='1') THEN
         q <= '0';
      ELSIF (clk'EVENT AND clk='1') THEN
         q <= d;
      END IF;
   END PROCESS;
END dff;
```

WAIT FOR is intended for simulation only. This statement is useful for generating the test benches.

CASE statement

```
CASE identifier IS
    WHEN value => assignments;
    WHEN value => assignments;
    ...
END CASE;
```

Example:

```
CASE control IS
   WHEN "00" => x<=a; y<=b;
   WHEN "01" => x<=b; y<=c;
   WHEN OTHERS => x<="0000"; y<="ZZZZ";
END CASE;
```

LOOP statement

FOR / LOOP: The loop is repeated a fixed number of times.

```
[label:] FOR identifier IN range LOOP
   (sequential statements)
END LOOP [label];
```

WHILE / LOOP: The loop is repeated until a condition no longer holds.

```
[label:] WHILE condition LOOP
   (sequential statements)
END LOOP [label];
```

EXIT: Used for ending the loop.

```
[label:] EXIT [label] [WHEN condition];
```

NEXT: Used for skipping loop steps.

```
[label:] NEXT [loop_label] [WHEN condition];
```

Thank you