جامعة نينوى
كلية هندسة الإلكترونيات

# Circuit Design with VHDL 8

Textbook: Volnei A. Pedroni

Submitted By: Hussein Aideen

# VHDL> Loop statement

- LOOP is useful when a piece of code must be instantiated several times.
- inside a PROCESS, FUNCTION, or PROCEDURE.
- FOR / LOOP: repeated a fixed number of times.

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

```
FOR i IN 0 TO 5 LOOP
   x(i) <= enable AND w(i+2);
   y(0, i) <= w(i);
END LOOP;
```

# VHDL> Loop statement

- WHILE / LOOP: repeated until a condition no longer holds.

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

```
WHILE (i < 10) LOOP
  WAIT UNTIL clk'EVENT AND clk='1';
  (other statements)
END LOOP;
```

# VHDL> Loop statement

- EXIT: Used for ending the loop.

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

- NEXT: Used for skipping loop steps.

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

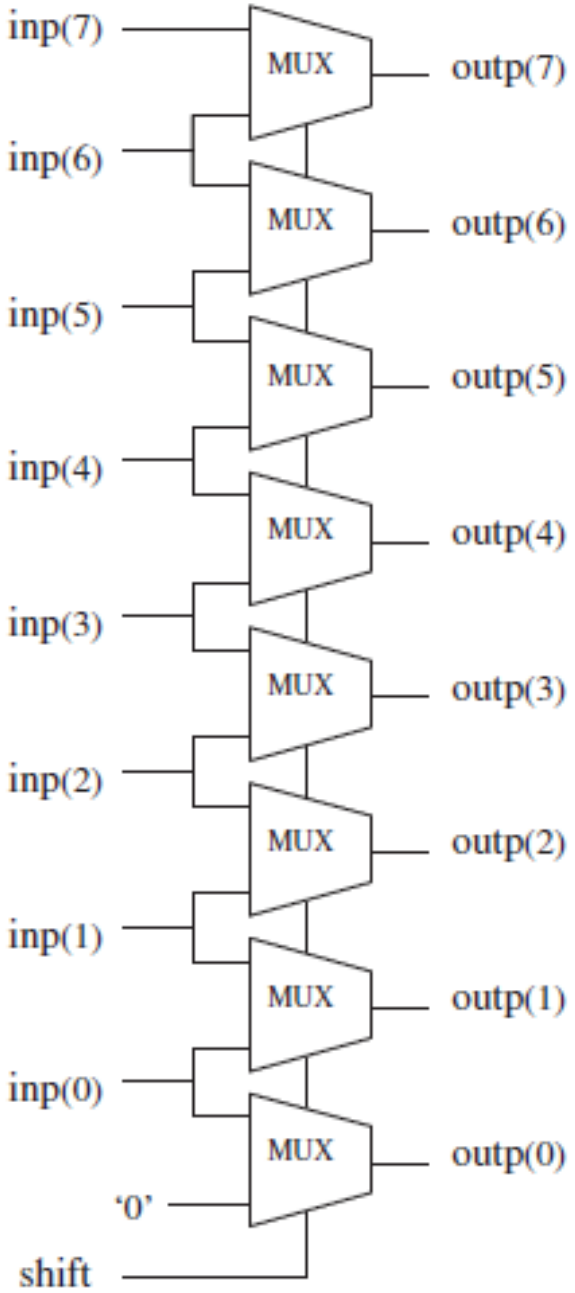
# VHDL> Loop statement

- Simple Barrel Shifter:
- The circuit must shift the input vector (of size 8) either 0 or 1 position to the left. When actually shifted (shift = 1), the LSB bit must be filled with '0' (shown in the botton left corner of the diagram).

- If shift = 0, then outp = inp;
- if shift = 1, then outp(0) = '0' and outp(i) = inp(i - 1), for 1≤ i ≤7.

# VHDL> Loop statement

- If shift = 0, then outp = inp;
- if shift = 1, then outp(0) = '0' and outp(i) = inp(i - 1), for 1≤ i ≤7.

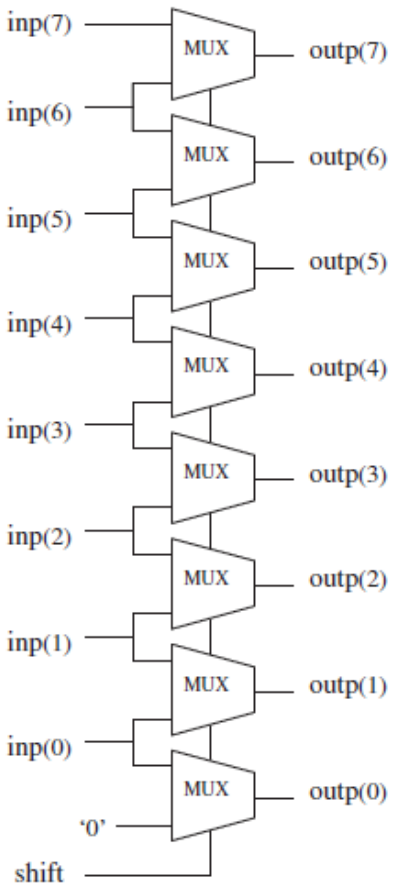


# VHDL> Simple Barrel Shifter

```
 1  ----------------------------------------------
 2  LIBRARY ieee;
 3  USE ieee.std_logic_1164.all;
 4  ----------------------------------------------
 5  ENTITY barrel IS
 6  GENERIC (n: INTEGER := 8);
 7  PORT ( inp: IN STD_LOGIC_VECTOR (n-1 DOWNTO 0);
 8   shift: IN INTEGER RANGE 0 TO 1;
 9   outp: OUT STD_LOGIC_VECTOR (n-1 DOWNTO 0));
10  END barrel;
11  ----------------------------------------------
```

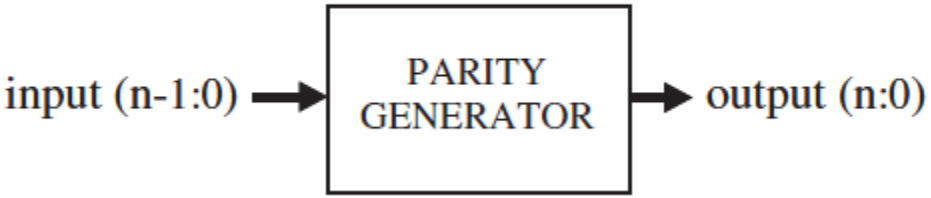
# VHDL> Simple Barrel Shifter

```
11  ----------------------------------------
12  ARCHITECTURE RTL OF barrel IS
13  BEGIN
14     PROCESS (inp, shift)
15       BEGIN
16       IF (shift=0) THEN
17         outp <= inp;
18       ELSE
19         outp(0) <= '0';
20         FOR i IN 1 TO inp'HIGH LOOP
21             outp(i) <= inp(i-1);
22         END LOOP;
23       END IF;
24     END PROCESS;
25  END RTL;
26   ----------------------------------------
```



# VHDL> Examples

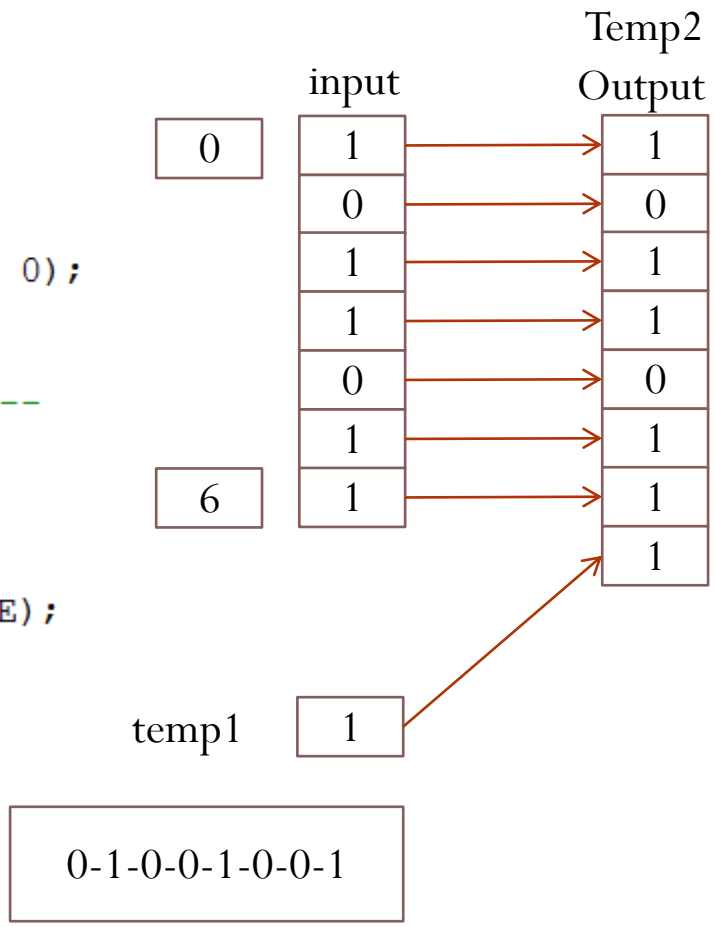
- Generic Parity **Generator**:
  - The circuit must add one bit to the input vector (on its left).
  - Such bit must be a '0' if the number of '1's in the input vector is even, or a '1' if it is odd, such that the resulting vector will always contain an even number of '1's (even parity).



# VHDL> Examples

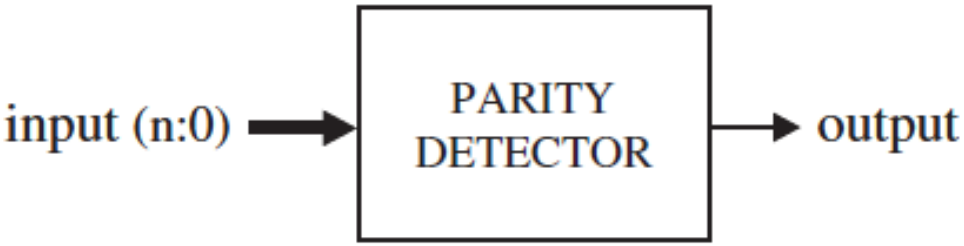
- Generic Parity **Generator**:

```
-----------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
-----------------------------------------
ENTITY parity_gen IS
GENERIC (n : INTEGER := 7);
PORT ( input: IN STD_LOGIC_VECTOR (n-1 DOWNTO 0);
output: OUT STD_LOGIC_VECTOR (n DOWNTO 0));
END parity_gen;
-----------------------------------------------
ARCHITECTURE parity OF parity_gen IS
BEGIN
PROCESS (input)
VARIABLE temp1: STD_LOGIC;
VARIABLE temp2: STD_LOGIC_VECTOR (output'RANGE);
BEGIN
temp1 := '0';
FOR i IN input'_RANGE LOOP
temp1 := temp1 XOR input(i);
temp2(i) := input(i);
END LOOP;
temp2(output'HIGH) := temp1;
output <= temp2;
END PROCESS;
END parity;
```



# VHDL> Examples

- Generic Parity Detector:
- The circuit must provide output = '0' when the number of '1's in the input vector is odd, or output = '1' otherwise.



# VHDL> Examples

- Generic Parity Detector:

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
-------------------------------------------
ENTITY parity_det IS
GENERIC (n : INTEGER := 7);
PORT ( input: IN Std_logic_VECTOR (n DOWNTO 0);
output: out Std_logic);
END parity_det;
-------------------------------------------
ARCHITECTURE parity OF parity_det IS
BEGIN
PROCESS (input)
VARIABLE temp: Std_logic;
BEGIN
temp := '0';
FOR i IN input'RANGE LOOP
temp := temp XOR input(i);
END LOOP;
output <= not temp;
END PROCESS;
END parity;
-------------------------
```