Integrated Circuits Design by FPGA

م.م. أحمد مؤيد عبدالحسين
جامعة الفرات الأوسط التقنية / الكلية التقنية الهندسية / نجف

Lecture 14

System Design

Multiply Accumulate Circuits (MAC)

Objectives of this Lecture

- To define **MAC**
- To implement **MAC** in an example design.

Contents of this Lecture

- Introduction
- MAC Example

Introduction

Multiply Accumulate Circuits:

Multiplication followed by accumulation is a common operation in many digital systems, particularly those highly interconnected, like digital filters, neural networks, data quantizers, etc.

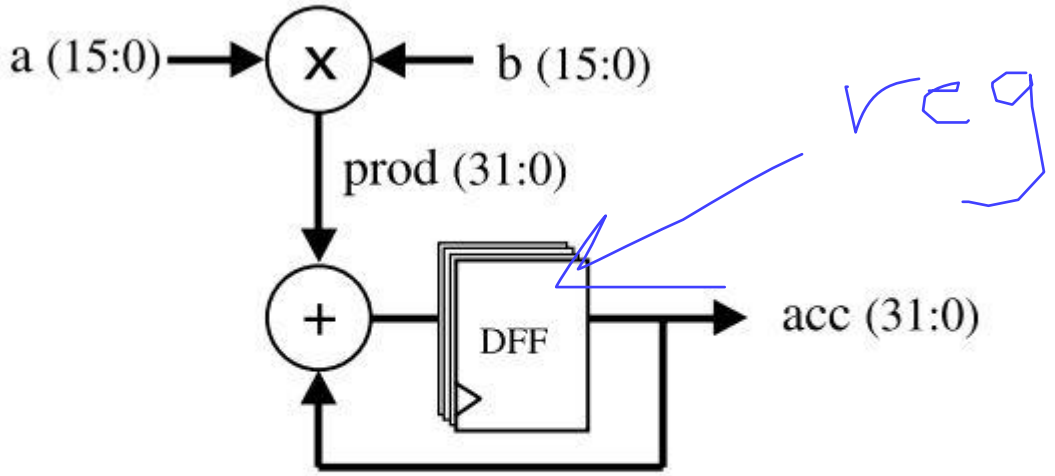


Figure 12.6
MAC circuit.

Introduction

Multiply Accumulate Circuits:

Overflow:

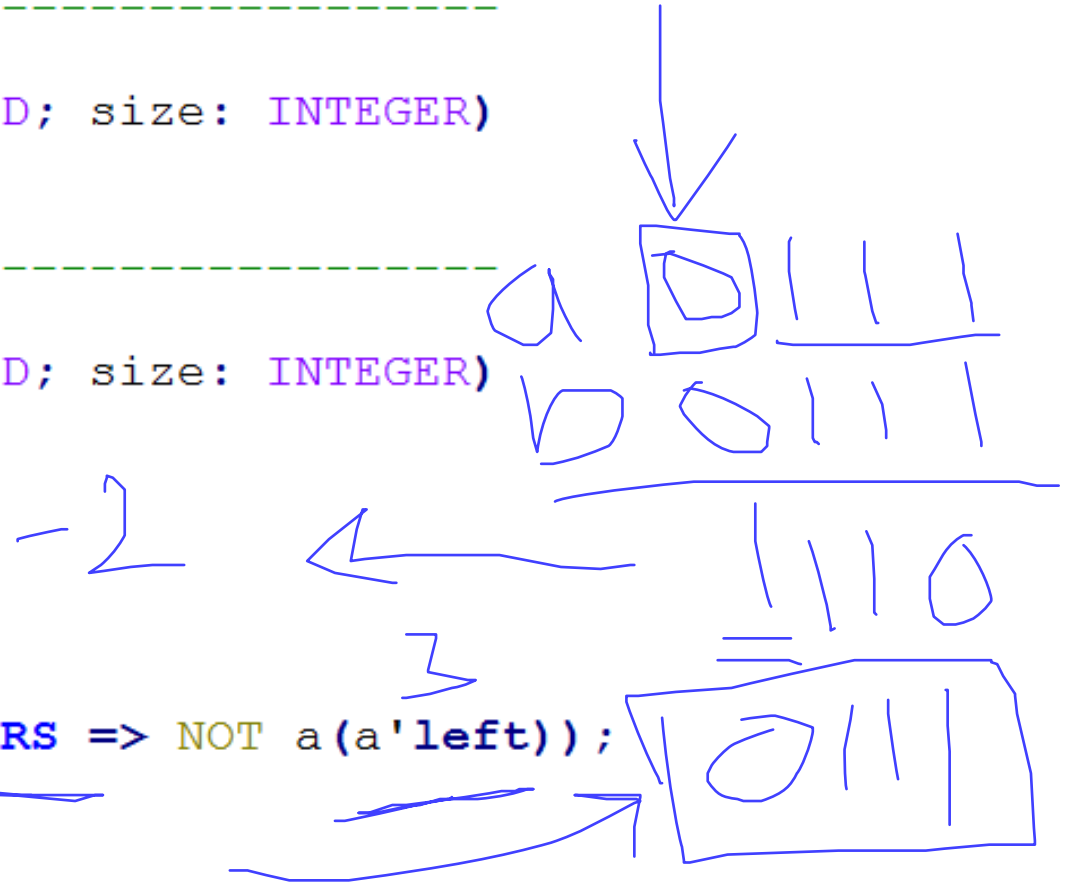
- For example, if eight bits **(signed)** are used to encode the values, the addition of two positive numbers must fall in the interval from **0** to **127**, while the addition of two negative numbers must fall between **-128** (that is, +128 in unsigned representation) and **-1** (255 in unsigned representation).
- 65 + 65 = 130 , which is indeed -126 (overflow), so the result should be truncated to the largest positive value (127).
- -70 + (-70) = -140 , which is, indeed, 116 (overflow), so the result should be truncated to the most negative value (-128).

MAC Example

```
------ Main code: ----------------------
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;
USE work.my_functions.all;
---------------------------------------
ENTITY mac IS
PORT ( a, b: IN SIGNED(3 DOWNTO 0);
clk, rst: IN STD_LOGIC;
acc: OUT SIGNED(7 DOWNTO 0));
END mac;
---------------------------------------
ARCHITECTURE rtl OF mac IS
SIGNAL prod: SIGNED(7 DOWNTO 0);
signal reg : signed(7 downto 0) := "00000000";
BEGIN
PROCESS (rst, clk)
VARIABLE sum: SIGNED(7 DOWNTO 0);
BEGIN
prod <= a * b;
IF (rst='1') THEN
reg <= (OTHERS=>'0');
ELSIF (clk'EVENT AND clk='1') THEN
sum := add_truncate (prod, reg, 8);
reg <= sum;
END IF;
acc <= reg;
END PROCESS;
END rtl;
---------------------------------------
```

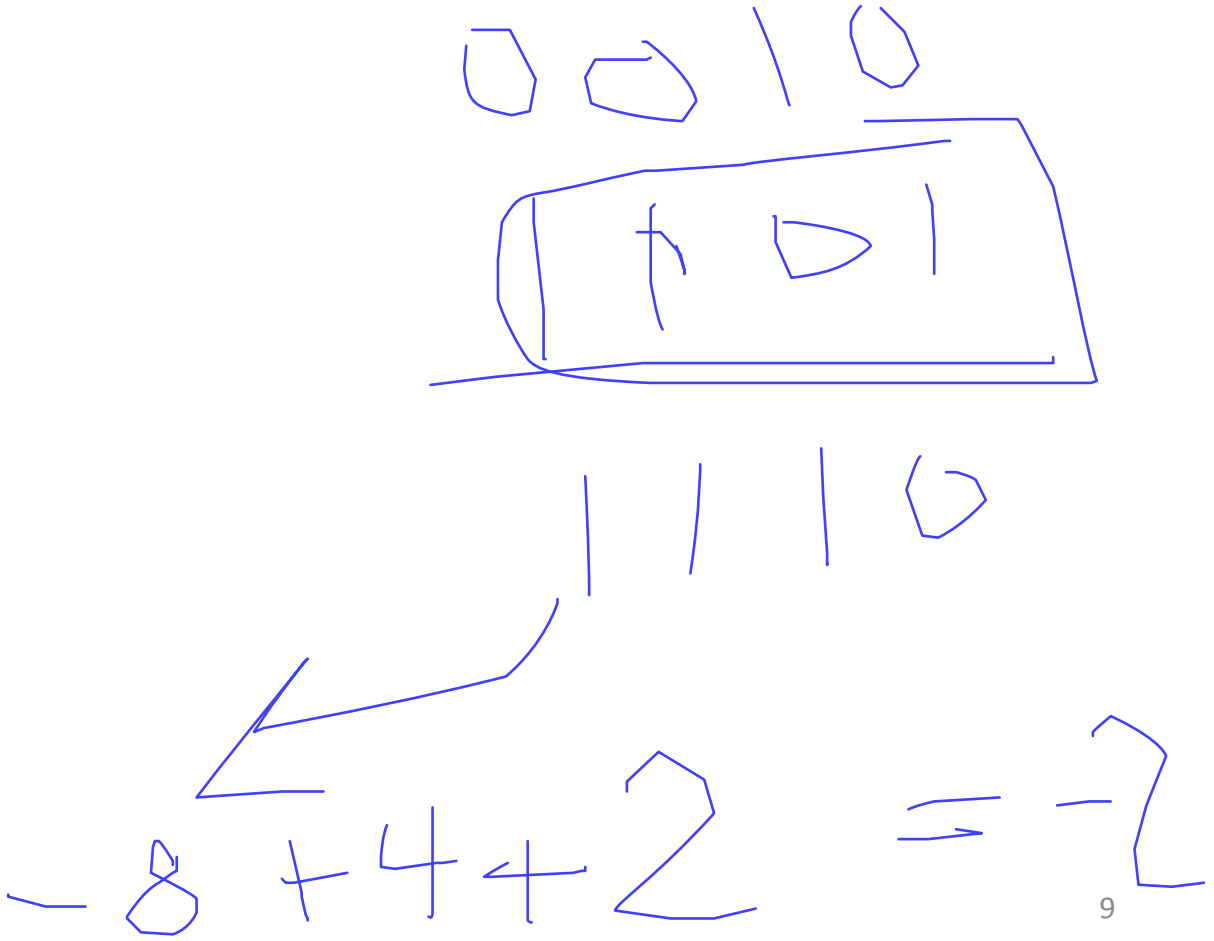
MAC Example

```
1   ------- PACKAGE my_functions: ----------------------------
2   LIBRARY ieee;
3   USE ieee.std_logic_1164.all;
4   USE ieee.std_logic_arith.all;
5   ------------------------------------------------------------
6   PACKAGE my_functions IS
7   FUNCTION add_truncate (SIGNAL a, b: SIGNED; size: INTEGER)
8   RETURN SIGNED;
9   END my_functions;
10  ------------------------------------------------------------
11  PACKAGE BODY my_functions IS
12  FUNCTION add_truncate (SIGNAL a, b: SIGNED; size: INTEGER)
13  RETURN SIGNED IS
14  VARIABLE result: SIGNED (7 DOWNTO 0);
15  BEGIN
16  result := a + b;
17  IF (a(a'left)=b(b'left)) AND
18  (result(result'LEFT)/=a(a'left)) THEN
19  result := (result'LEFT => a(a'LEFT), OTHERS => NOT a(a'left));
20  END IF;
21  RETURN result;
22  END add_truncate;
23  END my_functions;
24  ------------------------------------------------------------
```



Assignments

The assignments will be attached to your class room



End of lecture 14
Any Questions ?