Faculty of Computer Science Institute for Computer Engineering, Chair for VLSI-Design, Test and Architecture

Dresden, July 2, 2014

Summerschool

Modeling and Designing Dependable Embedded Systems

— Lab 3 —

Problem 1 Consider the two following implementations of the **entity** sigvsvar:

```
library IEEE;
use IEEE.std_logic_1164.all;

entity sigvsvar is
end entity;
```

Variant 1:

```
architecture sig of sigvsvar is
  signal a : std_logic := '0';
  signal b : std_logic := '1';
  signal c : std_logic := '0';
begin
  process
  begin
    wait for 5 ns;
    c <= a xor b;
    b <= a xor c;
    wait for 5 ns;
    c <= a xor b;
    b <= a xor c;
    wait;
  end process;
end architecture;
```

Variant 2:

```
architecture var of sigvsvar is
  signal a : std_logic := '0';
  signal b : std_logic := '1';
begin
  process
    variable c : std_logic := '0';
  begin
    wait for 5 ns;
    c := a xor b;
    b <= a xor c;
    wait for 5 ns;
    c := a xor b;
    b <= a xor c;
    wait;
  end process;
end architecture;
```

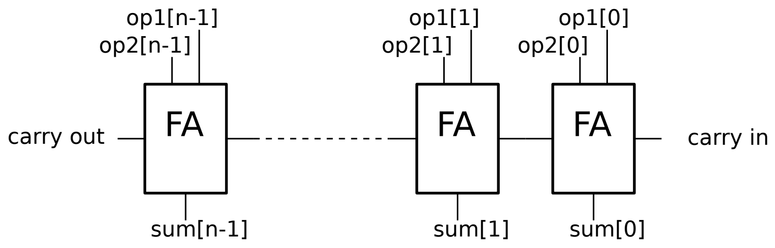
(a) Draw the waveform that you expect for the signals a and b for either implementation.

(b) Compare your assumption with the waveforms produced by simulation.

Problem 2

(a) Design a full adder implementing an **entity** FA with the appropriate input and output ports. Let the computation of all combinational outputs take 5 ns.

(b) Create a ripple-carry adder implementation, which instantiates full adders as **component**s to form a binary word adder:



Provide a generic parameter to specify the operand width of an individual instance.

(c) Instantiate a ripple-carry adder adding two 4-bit words in a testbench. Provide appropriate input stimuli to validate its correct operation.

(d) Within your testbench, find those stimuli, which maximize the time between changing input operands and the completion of the computation. What maximum delay were you able to observe?

Problem 3 Assume the following type declaration:

```
type tFaRes is record
                   c : std_logic; -- Carry Bit
                   s : std_logic; -- Sum Bit
               end record;
```

(a) Implement a **function** fa(a, b, c : std_logic) **return** tFaRes so that it models the operation of a full adder.

(b) Use this function in an alternative ripple-carry adder design, which uses a loop within a process to implement binary word addition.
Note: Do not try to enforce any explicit timing.

(c) Validate the correctness of this ripple-carry adder implementation using your testbench from the previous problem.