DIGITAL LOGIC WITH VHDL
(Fall 2013)
Unit 1

- ✓ *DESIGN FLOW*
- ✓ *DATA TYPES*
- ✓ *LOGIC GATES WITH VHDL*
- ✓ *TESTBENCH GENERATION*

Instructor: Daniel Llamocca

✓ *DESIGN FLOW*

- **Design Entry**: We specify the logic circuit using a Hardware Description Language (e.g., VHDL, Verilog).
- **Functional Simulation**: Also called behavioral simulation. Here, we will only verify the logical operation of the circuit. Stimuli is provided to the logic circuit, so we can verify the outputs behave as we expect.
- **Physical Mapping**: The inputs/outputs of our logic circuit are mapped to specific pins of the FPGA.
- **Timing Simulation**: It simulates the circuit considering its timing behavior (delays between inputs and outputs)
- **Implementation**: A configuration file ('bitstream' file) is generated and then downloaded onto the FPGA

✓ *DESIGN FLOW (ISE Software)*

- **Synthesis**: It makes sure that the VHDL file is syntax free. If ok, the logical circuit is ready for behavioral simulation.
- **Simulate Behavioral Model:** It requires the creation of a VHDL file in which we specify the stimuli to the logic circuit. This file is called ‘testbench’.
- **Implement Design** (Translate + Map + Place & Route):
- **Generate Programming File**: Here, a configuration file (bitstream) is generated. This file will configure the FPGA so that the logic circuit is implemented on it.
- **Configure Target Device** (iMPACT software): Here, the configuration file (.bit file) previously created is downloaded onto the FPGA. At this stage, we can verify whether the actual hardware is actually working.

✓ *LOGIC DATA TYPES*

- **Type:** There are many ways to define data types in VHDL. A very common IEEE standard is *std_logic_1164*. The following types are available:
  - std_logic, std_logic_vector, std_logic_2d
    - The 'std_logic' type define nine (9) possible states:
      - `'U' : Uninitialized`
      - `'X' : Forced Unknown`
      - `'0' : Zero`
      - `'1' : One`
      - `'Z' : High impedance`
      - `'W' : Weak unknown`
      - `'L' : Weak Zero`
      - `'H' : Weak One`
      - `'-' : Don't care`
- Other data types:
  - integer
  - array
  - User-defined

✓ *DATA TYPES:*

- **Mode:**
- Physical characteristics of inputs/outputs of a logic circuit. The following modes are available in VHDL:

  - IN : Input port of a circuit

  - OUT : Output port of a circuit

  - INOUT : Bidireccional port: It can be an input and an output at different times Very useful when working with bidireccional buses.

  - BUFFER : Output port of a circuit. It has the property that this output can be fed back to the circuit.

✓ *LOGIC GATES IN VHDL*

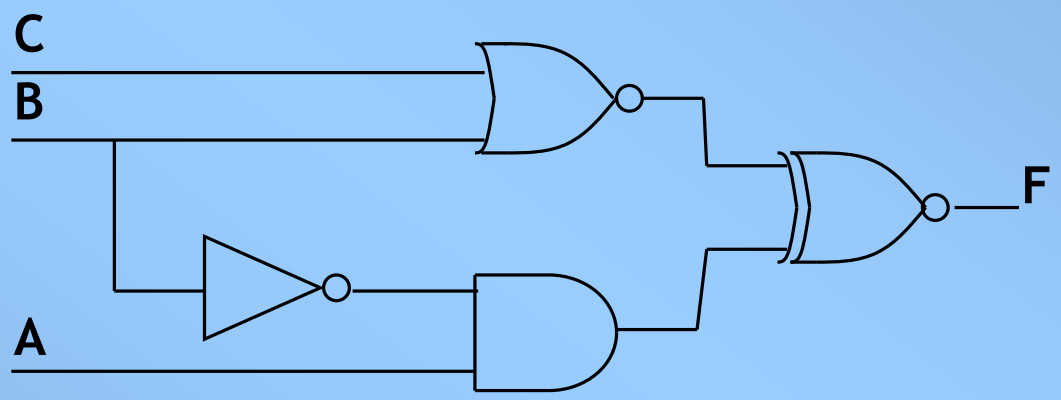
- **AND, OR, NOT, XOR:** Let's work on the following examples:

a) Write the VHDL code to generate the output 'f' given the truth table.

b) Write the VHDL code to generate the output 'F' given the circuit

X	Y	Z	f
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

(a)



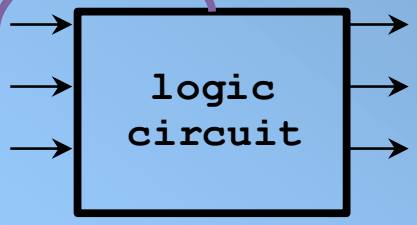
(b)

✓ *EXAMPLE (b):*
VHDL Coding

```
library ieee;
use ieee.std_logic_1164.all;
```

```
entity ex_a is
  port ( A, B, C: in std_logic;
         F: out std_logic);
end ex_a;
```

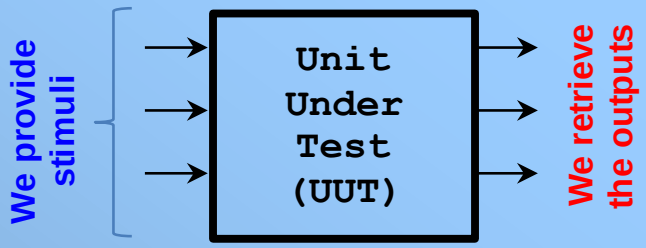
I/Os are specified here




```
architecture struct of ex_a is
    signal x,y: std_logic;
begin
    x <= C nor B;
    y <= A and not(B);
    F <= not(x xor y);
end struct;
```

Internal Description of the logic circuit is specified here

✓ Testbench Generation:




```
library ieee;
use ieee.std_logic_1164.all;

entity tb_ex_a is
end tb_ex_a;

architecture behav of tb_ex_a is
    component ex_a
    port ( A,B,C: in std_logic;
           F: out std_logic);
    end component;
    -- Inputs
    signal A: std_logic := '0';
    signal B: std_logic := '0';
    signal C: std_logic := '0';
    -- Outputs
    signal F: std_logic;
begin
    uut: ex_a port map (A=>A, B =>B, C=>C, F=>F);
    stim_proc: process -- Stimulus process
    begin
        wait for 100 ns -- reset state
        -- Stimuli:
        A <='0'; B <='0'; C <='0'; wait for 20 ns;
        A <='1'; B <='0'; C <='1'; wait for 20 ns;
        wait;
    end process;
end;
```

✓ ***EXAMPLE*** *(b):*

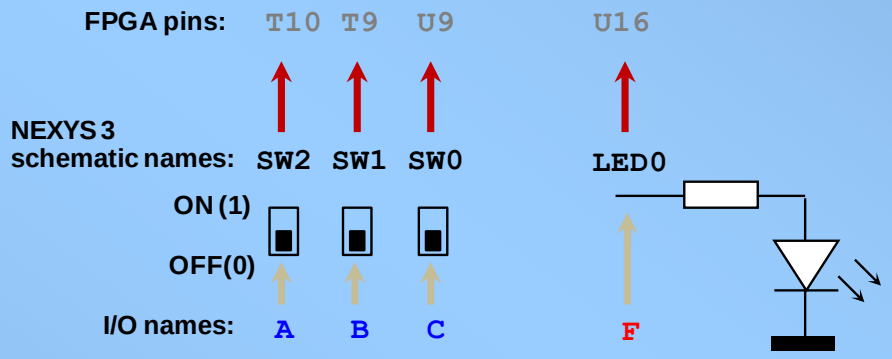
UCF file (for Xilinx Design Flow)

We need to map the I/Os of our logic circuit to physical FPGA pins. In a board (e.g., Nexys 3) these FPGA pins are connected to specific resources: LEDs, switches, buttons, etc.

UCF FILE:

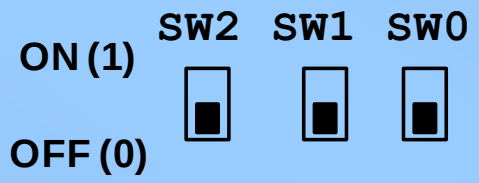
```
# Inputs
  NET "A" LOC = "T10"; # SW0
  NET "B" LOC = "T9";  # SW1
  NET "C" LOC = "V9";  # SW2

# Outputs
  NET "F" LOC = "U16"; # LED0
```

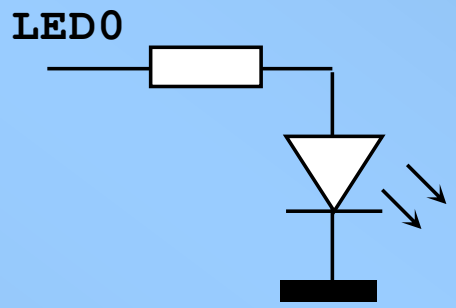


✓ *EXAMPLE: Light Control*

- There are three available switches. We want to light an LED when only one of the switches is in the ON position (logic ‘1’).



SW2	SW1	SW0	LED0
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0



✓ *EXAMPLE: Security Combination*

- A lock is opened only when a certain combination of switches exist: Switches: `01101011`
- For the NEXYS3 board, the lock will be represented by an LED.