

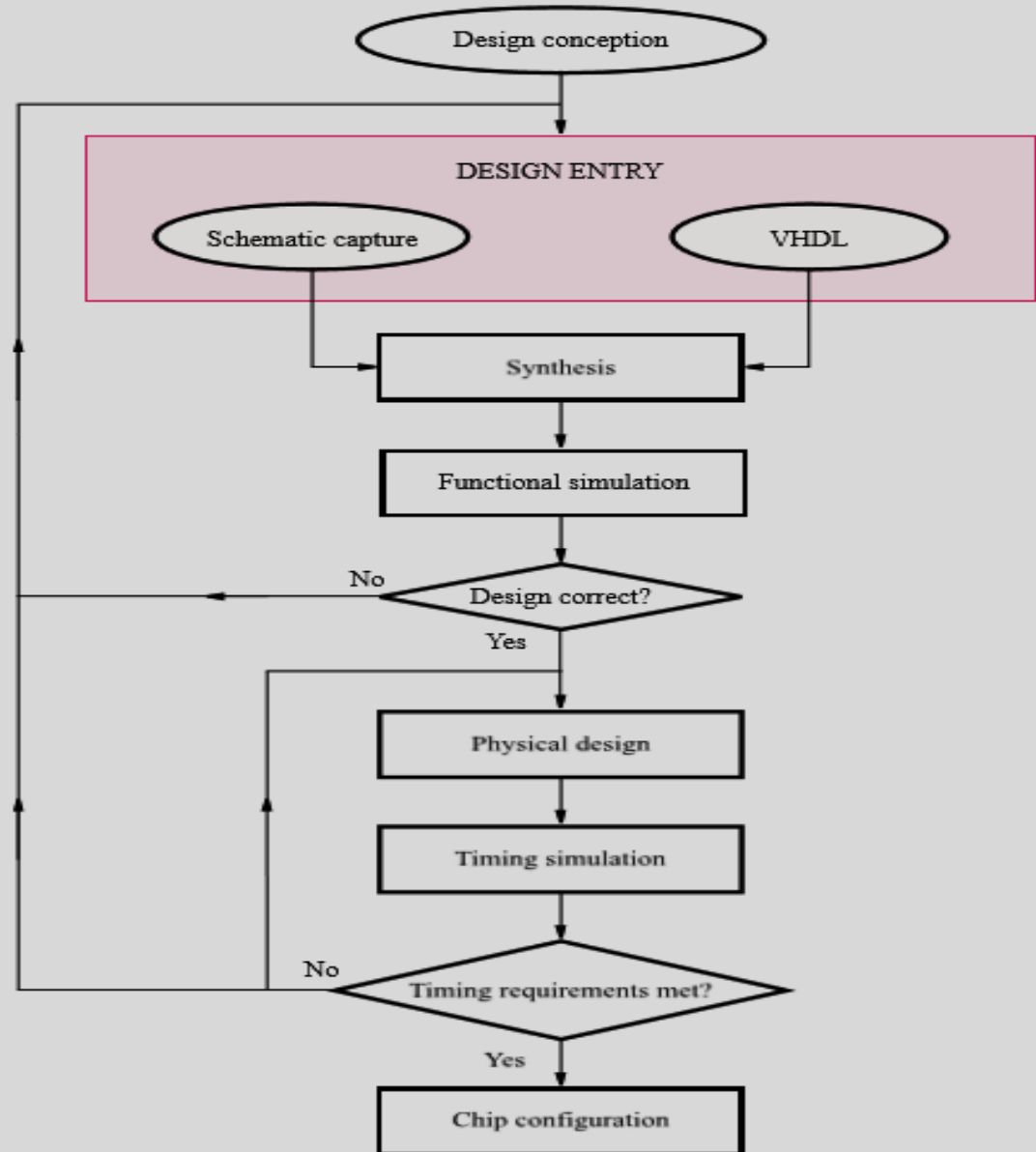
Lecture 6

CAD Tools and VHDL

Introduction to CAD tools

- ❑ A CAD system usually includes the following tools
 - Design entry
 - Synthesis and optimization
 - Simulation
 - Physical design

A typical CAD system

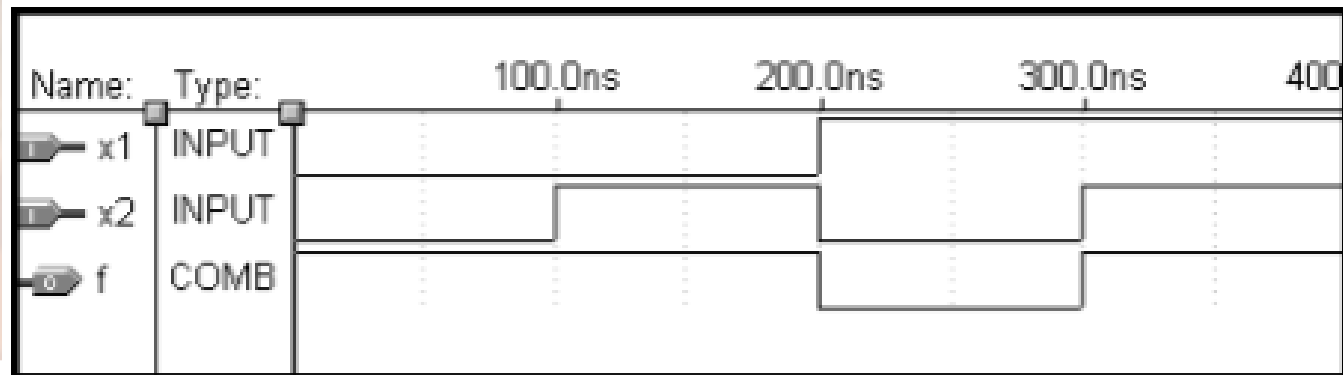


Design entry

- ❑ The process of entering into the CAD system a description of a circuit being designed is called design entry
- ❑ Three common design entry methods
 - Using truth tables
 - User enters a truth table in plain text format or draws a waveform that represents the desired functional behavior
 - Schematic capture
 - User graphically enters a desired logic circuit
 - Hardware description languages
 - User enters a programming language-like description of a desired logic circuit

Design entry with truth tables

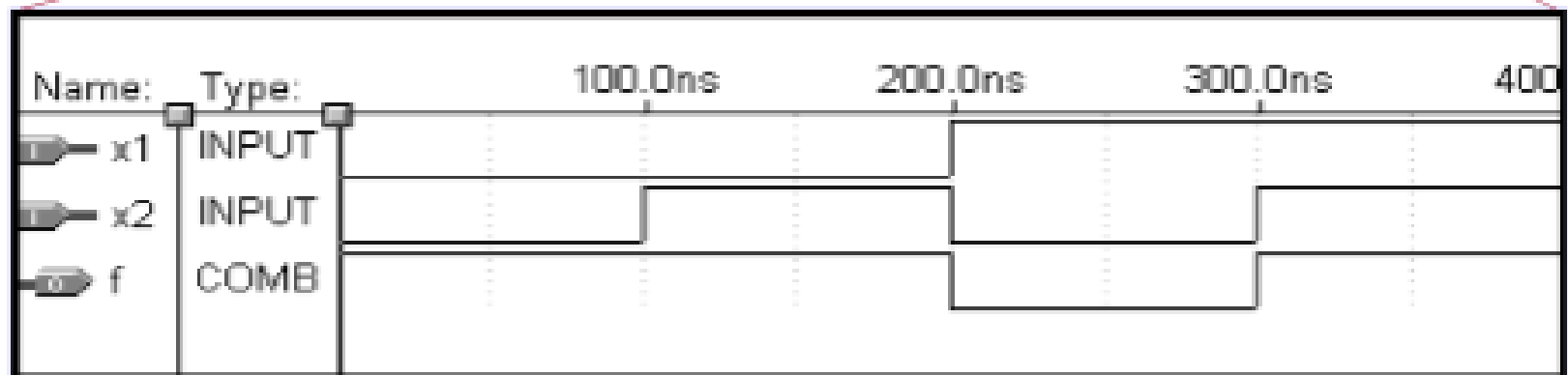
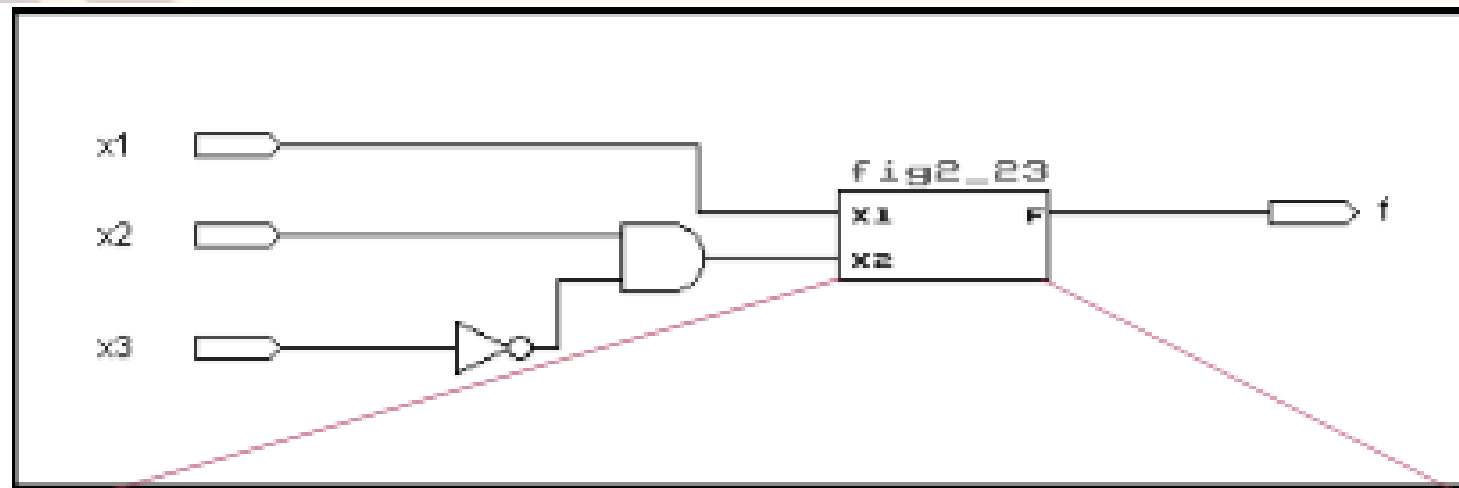
- ❑ Commonly use a waveform editor to enter a timing diagram that describes a desired functionality for a logic circuit
 - CAD system transforms this into equivalent logic gates
 - Not appropriate for large circuits, but can be used for a small logic function that is to be part of a larger circuit



Schematic capture

- ❑ A common type of CAD tool
- ❑ **Schematic:** refers to a diagram of a circuit in which circuit elements (logic gates) are shown as graphical symbols and connections between them are drawn as lines
- ❑ Tool provides a collection of symbols that represent gates of various types with different inputs and outputs. A **library**.
- ❑ Previously designed circuits can be represented with a graphical symbol and used in larger circuits. Known as **hierarchical design** and provides a way of dealing with complexities of large circuits

Schematic capture



Hardware description languages

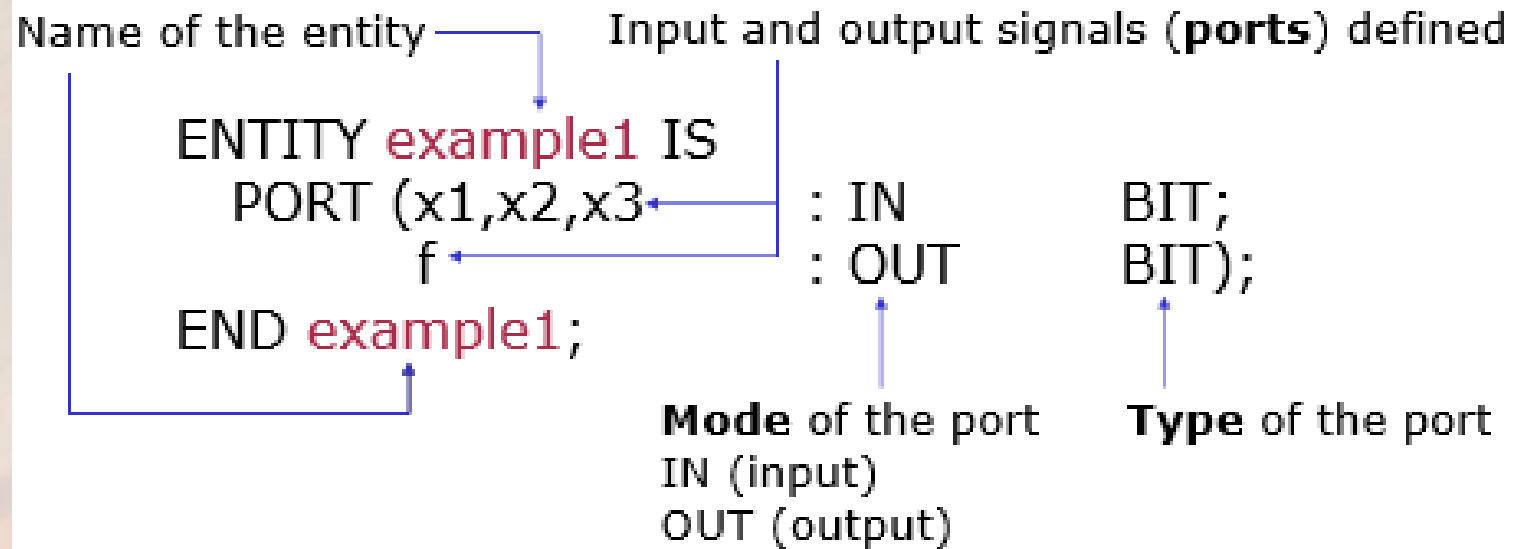
- ❑ A hardware description language (HDL) is similar to a computer program except that it is used to describe hardware
- ❑ Common HDLs
 - VHDL (VHSIC Hardware Description Language)
 - Verilog – Many others (vendor specific)
- ❑ VHDL and Verilog are standards
 - Offer portability across different CAD tools and different types of programmable chips

Introduction to VHDL

- ❑ Designer writes a logic circuit description in VHDL source code
- ❑ VHDL compiler translates this code into a logic circuit
- ❑ Representation of digital signals in VHDL
 - Logic signals in VHDL are represented as a data object
 - VHDL includes a data type called **BIT**
 - BIT objects can assume only two values: 0 and 1

Writing simple VHDL code

- ❑ First step in writing VHDL code is to declare the input and output signals
- ❑ Done using a construct called an entity



The diagram shows a VHDL entity declaration with blue arrows pointing from descriptive text to specific parts of the code. The code is as follows:

```
ENTITY example1 IS
  PORT (x1,x2,x3 : IN
        f       : OUT
              BIT;
              BIT);
END example1;
```

Annotations and their targets:

- Name of the entity** points to `example1` in `ENTITY example1 IS`.
- Input and output signals (**ports**) defined** points to the `PORT` clause.
- Mode of the port** (with sub-points `IN (input)` and `OUT (output)`) points to the `IN` and `OUT` modes.
- Type of the port** points to the `BIT` type.

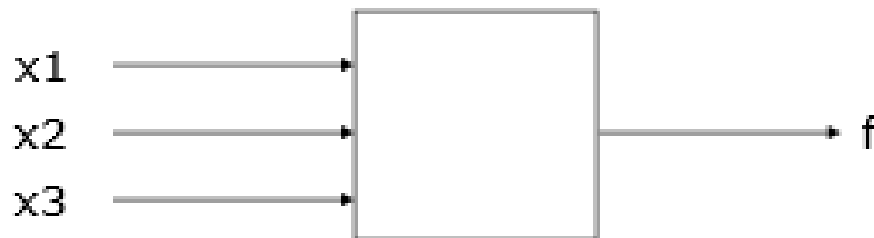
Writing simple VHDL code

Name of the entity Input and output signals (ports) defined

```
ENTITY example1 IS
  PORT (x1,x2,x3 : IN
        f       : OUT
              BIT;
              BIT);
END example1;
```

Mode of the port Type of the port

IN (input)
OUT (output)



Writing simple VHDL code

- ❑ The entity specifies the inputs and outputs for a circuit, but does not describe the circuit function
- ❑ Circuit functionality is specified using a VHDL construct called an **architecture**

Architecture name

Entity used by LogicFunc

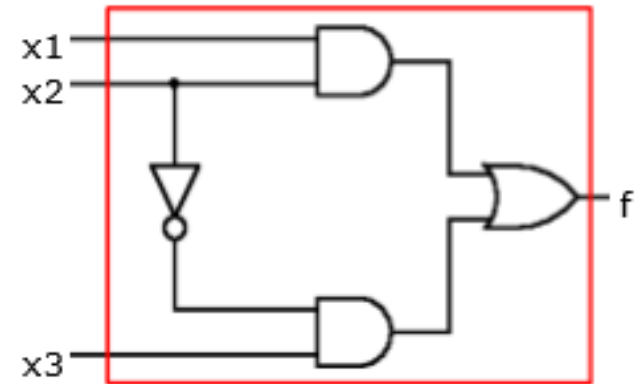
```
ARCHITECTURE LogicFunc OF example1 IS  
BEGIN  
    f <= (x1 AND x2) OR (NOT x2 AND x3);  
END LogicFunc;
```

VHDL statement that describes
the circuit functionality

Complete VHDL code example

```
ENTITY example1 IS
    PORT (x1,x2,x3 : IN  BIT;
          f          : OUT BIT);
END example1;
```

```
ARCHITECTURE LogicFunc OF example1 IS
BEGIN
    f <= (x1 AND x2) OR (NOT x2 AND x3);
END LogicFunc;
```



Boolean operators in VHDL

- ❑ VHDL has built-in support for the following operators
 - AND logical AND
 - OR logical OR
 - NOT logical NOT
 - NAND, NOR, XOR, XNOR
- ❑ Assignment operator `<=`
 - A variable (usually an output) should be assigned the result of the logic expression on the right hand side of the operator
- ❑ VHDL does not assume any precedence of logic operators. Use parentheses in expressions to determine precedence
- ❑ In VHDL, a logic expression is called a **simple assignment statement**. There are other types that will be introduced that are useful for more complex circuits.

Assignment and Relational Operators

❑ Assignment Operators

- For signal `<=`
- For variables `:=`

❑ Relational Operators

- `(x=y)` `(x/=y)` `(x<=y)` `(x>=y)`
- Example of use
- `(x='1')` `AND` `(y='0')`



VHDL code for the full-adder.

```
LIBRARY ieee ;  
USE ieee.std logic 1164.all ;  
ENTITY fulladd IS  
    PORT ( Cin, x, y : IN STD LOGIC ;  
          s, Cout : OUT STD LOGIC ) ;  
END fulladd ;  
ARCHITECTURE LogicFunc OF fulladd IS  
BEGIN  
    s <= x XOR y XOR Cin ;  
    Cout <=(x AND y) OR (Cin AND x) OR (Cin AND y) ;  
END LogicFunc ;
```

VHDL code for a four-bit adder

```
LIBRARY ieee ;
USE ieee.std logic 1164.all ;
USE work.fulladd package.all ;
ENTITY adder4 IS
PORT ( Cin : IN STD LOGIC ;
      X, Y : IN STD LOGIC VECTOR(3 DOWNT0 0) ;
      S : OUT STD LOGIC VECTOR(3 DOWNT0 0) ;
      Cout : OUT STD LOGIC ) ;
END adder4 ;
ARCHITECTURE Structure OF adder4 IS
SIGNAL C : STD LOGIC VECTOR(1 TO 3) ;
BEGIN
    stage0: fulladd PORT MAP ( Cin, X(0), Y(0), S(0), C(1) ) ;
    stage1: fulladd PORT MAP ( C(1), X(1), Y(1), S(1), C(2) ) ;
    stage2: fulladd PORT MAP ( C(2), X(2), Y(2), S(2), C(3) ) ;
    stage3: fulladd PORT MAP ( C(3), X(3), Y(3), S(3), Cout ) ;
END Structure ;
```

VHDL code for a 16-bit adder.

```
LIBRARY ieee ;
USE ieee.std logic 1164.all ;
USE ieee.std logic signed.all ;
ENTITY adder16 IS
    PORT ( X, Y : IN STD LOGIC VECTOR(15 DOWNT0 0) ;
          S : OUT STD LOGIC VECTOR(15 DOWNT0 0) ) ;
END adder16 ;
ARCHITECTURE Behavior OF adder16 IS
BEGIN
    S <=X+Y;
END Behavior ;
```

Reading

- Stephen Brown and Zvonko Vranesic
“Fundamentals of Digital Logic with VHDL
Design” THIRD EDITION
 - Sections 2.9 to 2.10