Introduction to VHDL

Module #5

Digilent Inc. Course

Background

- Availability of CAD tools in the early 70's
  - Picture-based schematic tools
  - Text-based netlist tools
- Schematic tools dominated CAD through mid-1990's
  - Using a graphics editor to build a structural picture of a circuit was easy compared to typing a detailed, error-free netlist
  - expensive graphics-capable workstations
  - Designs not compatible between computers or CAD tools

Background

- Early text-based tools gained momentum
  - Tools weren’t tied to high-end computers
- Progress in IC fabrication made it possible to place more transistors on a chip
  - Schematic methods were not scaling very well
- A designer could specify the behavior of a circuit that requires several thousand logic gates
  - Several layout engineers need weeks or months to transfer that behavior to patterns of transistors.
- Increase in complexity require more engineers on larger teams
  - Much larger technical data shared between workers.

Background

- 1981, U.S. DOD brought together a consortium of leading technical companies, and asked them to create a new “language” that could be used to precisely specify complex, high-speed integrated circuits.
  - detailed behavior of any digital circuit could be specified
- This work resulted in the advent of VHDL, an acronym for “Very-high-speed integrated-circuit Hardware Description Language”.

Background

- VHDL is used to provide a detailed design specification of a digital circuit
  - little thought given to how a circuit might be implemented
- A ”synthesizer” produces a low-level, structural description of a circuit based on its VHDL description
  - Automated behavioral-to-structural translation
  - Reduced amount of human effort

Background

- Use of HDL and synthesizers revolutionized the way in which digital engineers work
  - Early 1990s: very few new designs were started using HDLs (the vast majority were schematic based).
  - Mid 1990’s: roughly half of all new designs were using HDLs
  - Today: all but the most trivial designs use HDL methods.

CAD Tools

- Front-end tools
  - Allow a design to be captured and simulated
  - Virtual circuits
- Back-end tools
  - Synthesize a design, map it to a particular technology, and analyze its performance
  - Physical circuits
- Several companies offer CAD HDL tools
  - VHDL
  - Verilog

VHDL vs. Verilog

- Both are similar in appearance and application
- Both have their relative advantages.
- We will use VHDL because a greater number of educational resources have been developed for VHDL than for Verilog
- It should be noted that after learning one of the two languages, the other could be adopted quickly

Digital Design Today

- HDLs have allowed design engineers to increase their productivity many fold in just a few years.
  - A well-equipped engineer today is as productive as a team of engineers a few years ago.
- To support this increased level of productivity, engineers must master a new set of design skills
  - Craft behavioral circuit definitions that meet design requirements
  - Understand synthesis so results can be interpreted
  - Model external interfaces to the design so that it can be verified
- The extra degree of abstraction that HDL allows brings many new sources of potential errors
  - Designers must be able to recognize and address such errors when they occur

Structural vs. Behavioral Design

- A behavioral circuit design is a description of how a circuit's outputs are to behave when its inputs are driven by logic values over time.
  - no information to indicate how a circuit might be constructed
- A structural circuit definition is essentially a plan, recipe, or blueprint of how a circuit is to be constructed
  - no information to indicate how a circuit might behave
- HDL Files: commonly a mixture of the two

Structural vs. Behavioral Design

- When a behavioral circuit is synthesized, the synthesizer must search through a large collection of template circuits, and apply a large collection of rules to try to create a structural circuit that matches the behavioral description.
  - The synthesis process can result in one of several alternative circuits being created due to the variability inherent in generating rule based solutions.
- When a structural description is synthesized, the synthesizer's job is a relatively straightforward, involving far fewer rules and inferences.
  - A post-synthesis structural circuit will closely resemble the original structural definition (preferred by designers)

Structural vs. Behavioral Design

- In general, it is far easier and less time consuming to define a given circuit using behavioral methods
  - Allow engineers to focus on high-level design considerations
  - Not allow engineers to control structure of final circuit.
  - Synthesizers must use rules that are applicable to wide range of circuits, and cannot be optimized for a particular circuit.
  - In some situations, engineers must have greater control over final structure of their circuits.
- Often, engineers start design with behavioral description so they can readily study the circuit and possible alternatives.
  - Once a particular design is chosen, it is recoded in structural form so synthesis becomes more predictable.

Structural vs. Behavioral Design

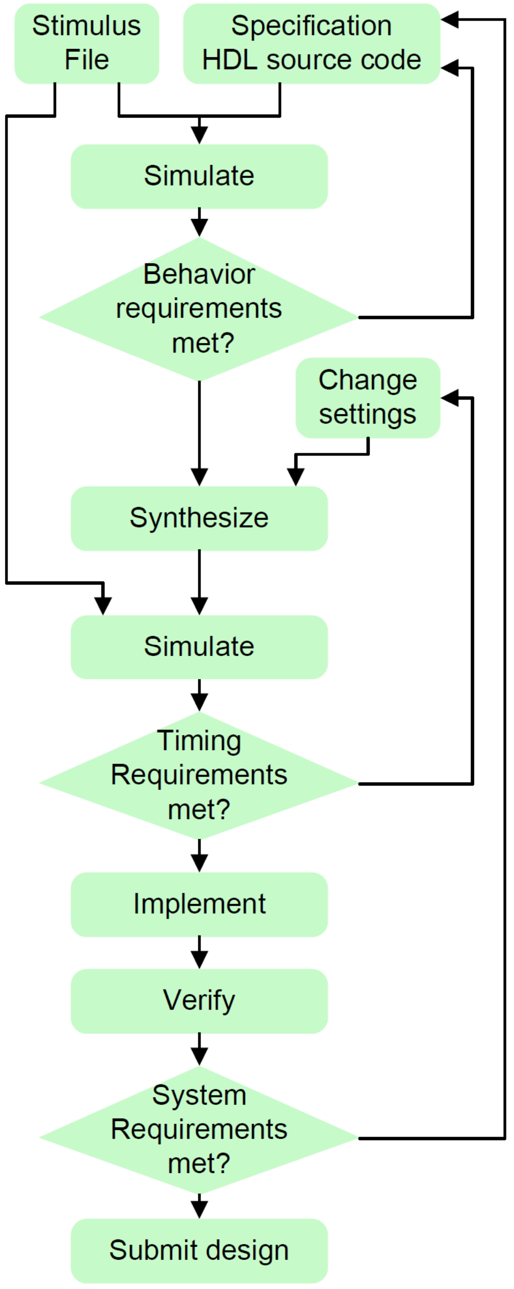
- Instead of using GUI to add gates and wires to a schematic, HDLs editors use a text editor to add structural or behavioral descriptions to a text file.
- Behavioral descriptions describe the conditions required for a given signal to take on a new value.
- Structural descriptions use components interconnected by signal names to create a netlist

Simulation and Synthesis

- A VHDL design can be simulated to check its behavior, and/or synthesized so that it can be implemented.
- These two functions, simulation and synthesis, are really separate functions that do not need to be related.
- In a typical flow, a new design would be simulated, then synthesized, and then simulated again after synthesis to ensure the synthesizer did not introduce any errors.

HDL Design Flow

- During synthesis, designer can impose design constraints
  - Power consumption
  - Implementation area
  - Operating speed
- Designers must understand synthesis process very well
  - Must be able to thoroughly analyze the postsynthesis circuit to make sure that all required specifications are met



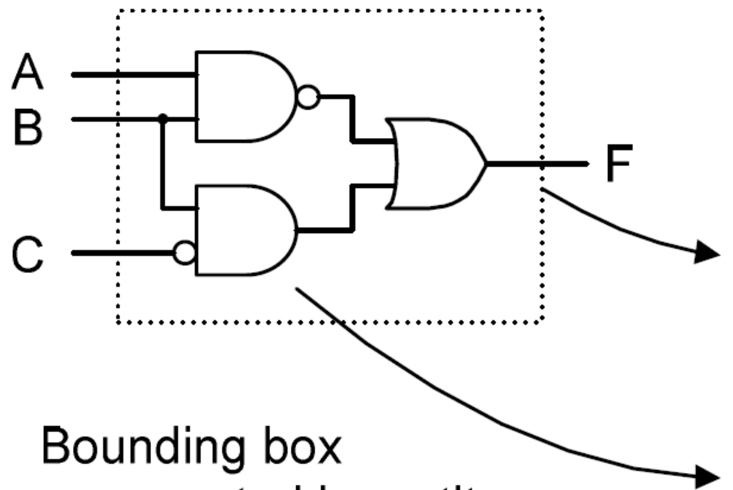
Structure of VHDL Source Code

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

entity circuit_name is
  port (list of inputs, outputs and type);
end circuit_name;

architecture arch_name of circuit_name is
begin
    (statements defining circuit go here);
end arch_name;
```

VHDL Example



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

entity Example is
  port (A,B,C    : in STD_LOGIC;
        Y        : out STD_LOGIC);
end Example

architecture behavioral of Example is
begin
  Y <= (not (A and B) or (B and not C));
end behavioral;
```

Bounding box represented by entity statement; behavior by architecture statement

VHDL Syntax

- Port : input and output signals
- “std_logic” type: physical signals
- Other signal data types: abstract only
- Signal assignment operator “<=”: indicate how an output signal is to be driven
- “A <= B”: signal A gets assigned signal B
  - VHDL simulator requires some time passes before signal is allowed to take new value
  - voltage on wire cannot change instantaneously
  - Different from C language

VHDL Syntax

- VHLD code is inherently *concurrent*
  - At any given time, several signal assignments may be pending.
  - Cause-and-effect relationships are not a function of where a statement occurs in the VHDL code, but rather how time is modeled
- Signal assignment operators assign output signal new value based on a *function* that operates on input signals
  - *and*, *or*, *nand*, *nor*, *xor*, *xnor*, and *not*
  - Must be terminated with a semicolon.