HDL (Hardware Description Language) Introduction

Programmable Device based System

- The two main leading domain in the electronic world is the Embedded Systems and Programmable Device (chip) based System.

- While a processor is a hard-hardware component and its partial flexibility can be achieved using different program for each application. Programmable Device is a flexible-hardware component which designed using HDL (***h****ardware* ***d****escription* ***l****anguage*) coding.

What is VHDL?

- VHDL is a ***h**ardware **d**escription **l**anguage*. It describes the behavior of an electronic circuit or system, from which the physical circuit or system can then be implemented.
- ***V**ery high speed integrated circuits **H**ardware **D**escription **L**anguage*
- VHDL is intended for circuit synthesis as well as circuit simulation.
- A fundamental motivation to use **VHDL** (or **Verilog**) is that VHDL is a standard, technology or vendor independent language, and is therefore portable and reusable.
- The two main applications of VHDL are in the field of Programmable Logic Devices (CPLDs, **FPGAs**) and in the field of Application Specific Integrated Circuits (**ASICs**).

What is VHDL?

- Once the VHDL code has been written, it can be used either to implement the circuit in a programmable device or can be submitted to a foundry for fabrication of an ASIC chip.

- In contrary to regular computer programs which are sequential, HDL statements are inherently concurrent (parallel). For that reason, VHDL is usually referred to as a code rather than a program. For that reason when we write code in HDL we should think HW.

- C is a programming language, VHDL is a hardware description language.

VHDL **vs** Verilog (main differences)

- Verilog is a weakly typed language as compared to VHDL which is a strongly typed language. This means that Verilog won't easily compile a script which is not very strongly typed. Intermixing of classes and variables would be difficult in VHDL.

- Verilog is case sensitive while VHDL is case insensitive.

- Verilog (more like C) is easier to learn as compared to VHDL (VHDL is much more complicated than Verilog).

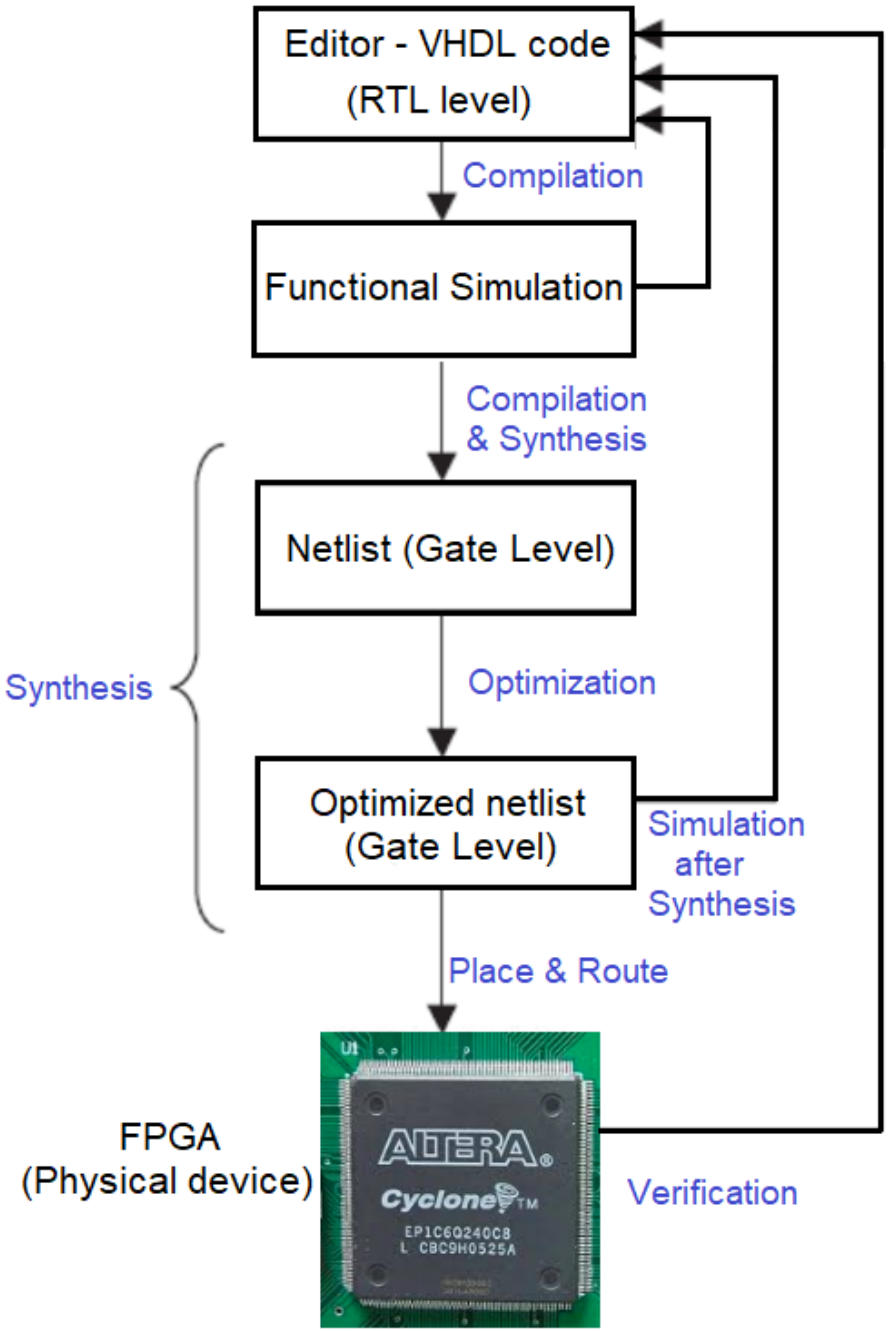
Who uses it?

VHDL Design Flow

- We start the design by writing the VHDL code, which is saved in a file with the extension **.vhd** and the same name as its ENTITY's name (each entity is located in separate file).

- The first step in the synthesis process is compilation, is the conversion of the high-level VHDL language, which describes the circuit at the Register Transfer Level (RTL), into a netlist at the gate level.

- The second step is optimization, which is performed on the gate-level netlist for speed or for area, at this stage, the design can be simulated.



Design Flow

- The third step is a place and route (fitter) software will generate the physical layout for a FPGA chip or will generate the masks for an ASIC chip.

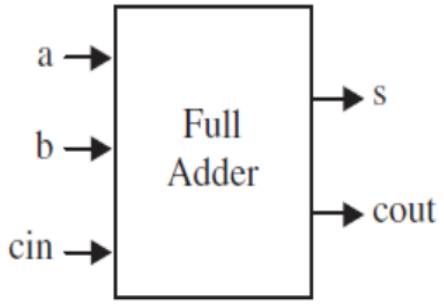
Translation of VHDL Code into a Circuit

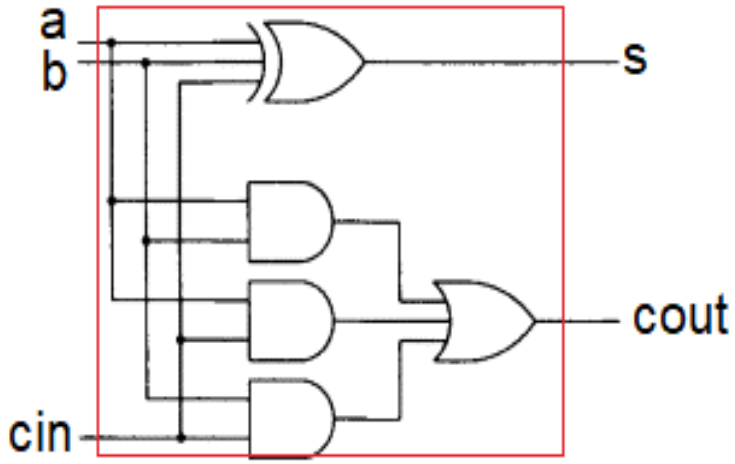
- From the next VHDL code, a physical circuit is inferred.

- There are several ways of implementing the equations described in the **ARCHITECTURE**, so the actual circuit will depend on the compiler and optimizer being used and more importantly, on the target technology.

Translation of VHDL Code into a Circuit

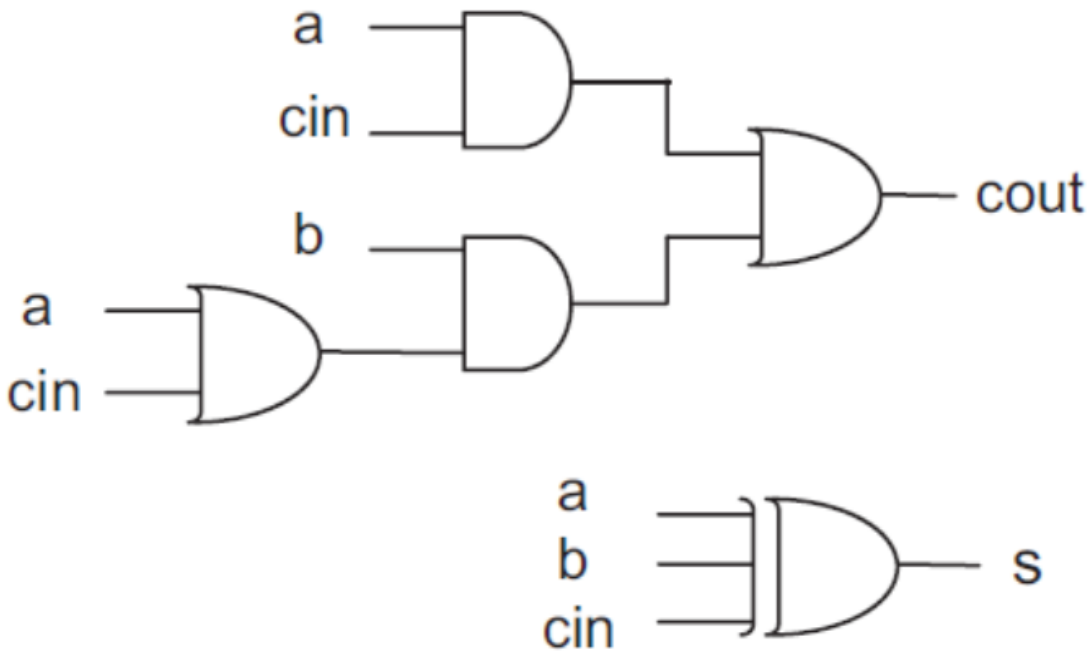
```
ENTITY full_adder IS
    PORT (a, b, cin: IN BIT;
          s, cout: OUT BIT);
END full_adder;
-----------------------------------------
ARCHITECTURE dataflow OF full_adder IS
BEGIN
    s <= a XOR b XOR cin;
    cout <= (a AND b) OR (a AND cin) OR(b AND cin);
END dataflow;
```





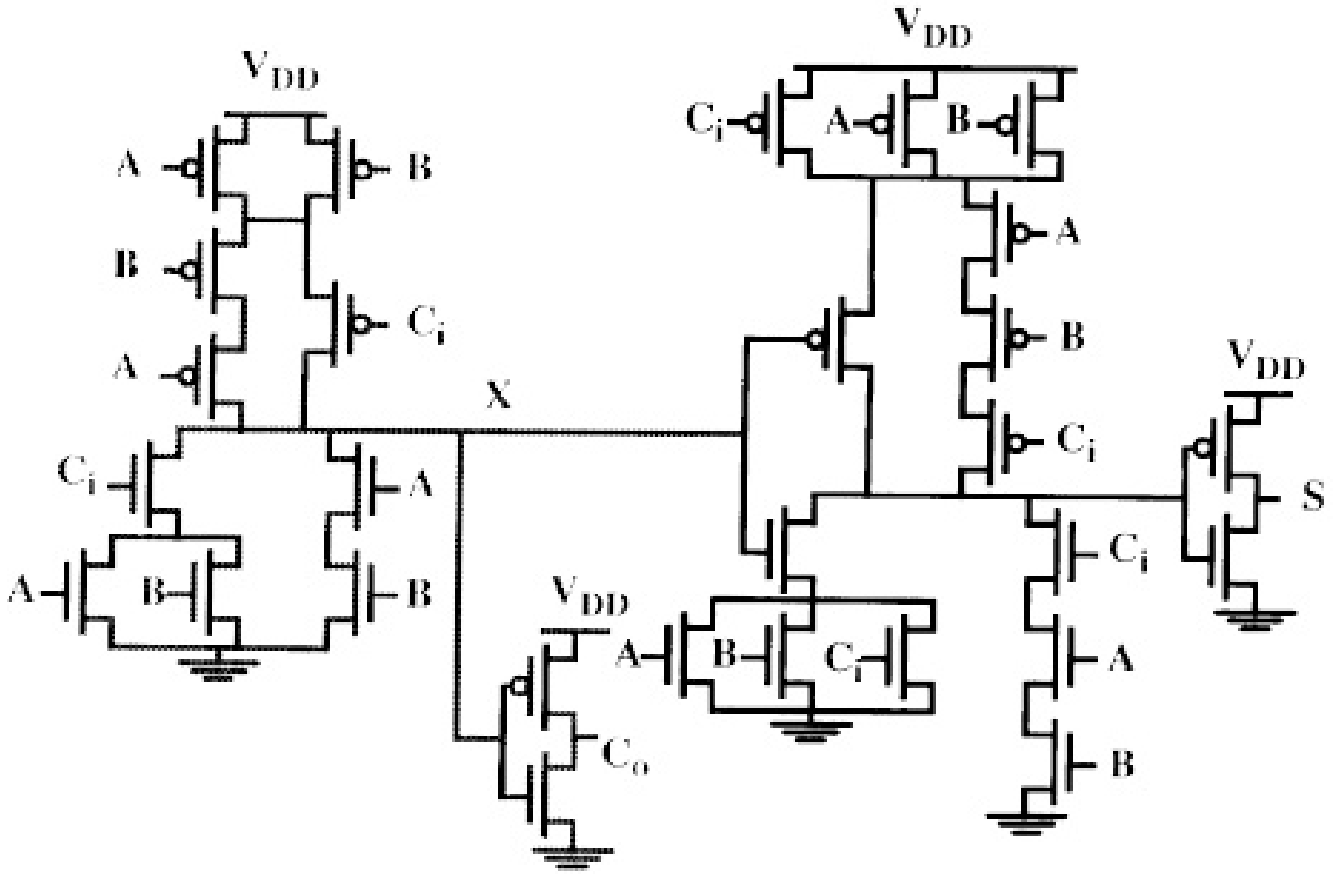
Translation of VHDL Code into a Circuit

- For Full Adder instance, if our target is a **FPGA** device then one possible result is:



Translation of VHDL Code into a Circuit

- For Full Adder instance, if our target technology is an **ASIC**, then a possible CMOS implementation, is:



Translation of VHDL Code into a Circuit

- For Full Adder instance, if our target technology is an **ASIC**, then a possible Layout implementation, is: