Creating Logic Circuits

* Standard Chips: breadboards were used for prototyping and *printed circuit boards* (PCBs) for ﬁnal products
* Programmable Logic Devices (PLDs)
* Application-speciﬁc Integrated Circuits (ASICs)
  + Manufacture complex logic circuits from scratch directly on a single chip
  + When you can’t sacriﬁce on performance and/or scalability, you have no choice but to use an ASIC

Programmable Logic Devices (PLDs)

* Instead of simple fixed-function units, PLDs have programmable logic circuit networks
  + Often, this programmability is exposed through a sequence of switches
  + Different switch settings enable/disable different functionality
  + About 103 – 1014 transistors
  + PLDs can be re-programmed many times
  + One of the most popular and flexible types PLDs are field-programmable gate arrays are *field-programmable gate arrays* (FPGAs)
  + These FPGAs allow for almost arbitrary logic configurations to be programmed
    - Here, PLD “switches” are connections that can be programmed in order to configure different design functionality

VHDL

* Industry-standard hardware description language
* Designed with two purposes in mind:
  + Documentation language: to describe the structure of complex digital circuits (designer-agnostic **documentation**)
  + Language features to model the behavior of digital circuits
    - Computer-aided design (CAD) tools could use these models to simulate circuit behavior
* Modern CAD tools can take VHDL and realize a physical circuit
* We will rely on VHDL as our synthesis tool: for both simulation and realization

What are the advantages of using an FPGA instead of a processor?

* Programmable functionality
* Implement very complex circuits on a single unit
* / – overkill for simple logic circuits/tasks
* / – decent use of PCBsarea (transistor density)
  + Cost implications at scale
* Task-specialized circuits will be more efficient
  + Performance implications at scale

What are the main components of a VHDL circuit design?

Every VHDL design description consists of at least one *entity / architecture* pair, or one entity with multiple architectures. This idea of separating a description from the implementation/functionality was well ahead of its time

Before we can code up the functionality of a HW unit, we need to provide the VHDL compiler with a description of the HW’s inputs and outputs

* There is a VHDL mechanism, called an **entity**, for specifying such a descriptor

An entity doesn’t specify anything about what the circuit actually does or what the inputs/outputs represent. A circuit’s functionality is speciﬁed in a VHDL construct called an **architecture.**

Explain the functionality of **entity** and **architecture** in a VHDL design.

The entity section of the VHDL design is used to declare the *I/O ports* of the circuit, while the description code resides within architecture portion. Standardized design libraries are typically used and are included prior to the entity declaration.

* Think of ports as all the wires that can go in and out of your circuit
* Inputs and outputs are all speciﬁed here
  + and each one has a type
* The architecture uses the signal assignment operator to specify that the output f should be assigned the value of the logic expression to the right (<=)