Implementation Technologies

• We can implement a design with many different implementation technologies - different implementation technologies offer different tradeoffs
  – **HDL Synthesis** offers an easy way to target a model toward different implementations
  – There are also re-targeting tools which will convert a netlist from one technology to another (from a standard cell implementation to a Field Programmable Gate Array implementation).

Complete Cycle for Design

IDEA

- model
- synthesize
- verify
- optimize
- layout

DESIGN

FABRICATE

- mask production
- wafer production

DEVICE

- slice/dice
- package

PACKAGE

TEST

- generate test vectors
- wafer tester

- wafer production
Design Process

- **Model**
  - Must represent the desired functionality at some level
- **Synthesize**
  - Must create a new abstraction (or model) of the functionality typically at a more detailed level.
- **Optimization**
  - During synthesis, the details added to the newly created abstraction are a result of the specified optimizations
- **Verification**
  - Must ensure that two abstractions created in the design process have the same functionality.

*Synthesis* is the process of creating a new abstraction of a specification.

*Verification* is the process of comparing two abstractions to determine equivalence of some property.

What is the difference between verification, testing, simulation and emulation???

Design Target Alternatives

- **Full-Custom Design**
  - design to the transistor level and perform all place/route; most control over shape of the mask pattern

- **Semi-Custom (Standard Cell)**
  - design to the cell level, cells must be placed and routed; less control over shape of mask pattern

- **Gate Array**
  - design to cell level, cells are already placed and routed in given mask pattern; must “fit” design into pre-specified mask pattern

- **Standard Components**
  - Pre-manufactured chips are chosen and placed on a circuit board to achieve the desired functionality. This option is used less frequently in modern design.
Different Implementation Technologies

- **Full-Custom Design**
  - ALU Core in Modern Microprocessors

- **Semi-Custom (Standard Cell)**
  - Many Modern Microprocessors
  - Dedicated ASICs (*Application Specific Integrated Circuits*)

- **Gate Array and Programmable Logic Devices (PLDs)**
  - Mask Programmable
  - OTP Field Programmable
  - Field Programmable
  - Complex PLDs
  
  *NOTE: CPLDs and FPGAs undergo a place/route where specific signal paths are “chosen” from the fixed paths present in the device*

- **Standard Components**
  - Fixed Application
  - Software Configurable

*from Brown/Vranesic Reference*
## Full Custom

- Geometries are Hand-drawn that specify transistors and other devices for an integrated circuit.
- Requires expertise in VLSI (Very Large Scale Integration) design
- Very high transistor density (transistors per square micron)
- Design time can be relatively long (multiple months).
- Involves the creation of a completely new chip, which consists of about a dozen masks (for the photo-lithographic manufacturing process)
- Mask creation is expensive

## Full Custom (cont)

- Very high performance can be achieved based on
  - available process technology
  - designer skill
  - CAD tool assistance.
- Fabrication costs are high
  - *non-recurring engineering* costs (NRE) are high (in the thousands of dollars)
  - need large volume to spread NRE costs among chips for custom-design to be economical
- A single custom chip can cost hundreds of thousands of dollars
**Full Custom (cont)**

- Fabrication time from geometry submission (i.e. “tape out”) to returned chips is weeks/months.
- Full custom is generally how mixed Analog/Digital cores are designed.
- New methods emerging for “Systems on Chip” (SoC)
- An example VLSI layout is shown below.

**NMOS Layout Example (33 transistors)**

Red - Polycrystalline silicon (Poly or Poly Si)
Green - N-doped silicon
Blue - metal 1 (typically aluminum)
White - conducting contact
Tan - metal 2

*Poly is conducting material used for gate construction of transistor*
Standard Cell Design

- IC is composed of Interconnection of subcircuits from Library of Standard Cells
- Automatic Place and Route tool used to produce the Layout
- Designer does not have to be a VLSI expert.
- Transistor density and performance degradation depends on type of design being done
  - Usually Performs well Not for random logic
  - Performance degradation can be significant for datapath type designs.
  - Quality of available library and tools make a significant difference.
- Design time can be much faster than full custom because layout is automatically generated.

Standard Cell Design Flow
Standard Cell Example Layout

- Still involves creation of custom chip so all masks must still be made; manufacturing costs same as full custom.
- Fabrication time same as full custom.

Mask Programmable Gate Array

- Library of standard cells used based on assets available on programmable device
- Design mapped onto an array of transistors already present on wafer
- Wafers with transistor arrays created ahead of time - this avoids automatic placement
- Automated routing tool creates the masks for the routing layers and "customizes" the pre-created gate array
Mask Programmable Gate Array

- Transistor density can be almost as good as standard cell design
- Design time advantages are approximately same as for standard cell
- Fabrication Costs Reduced Compared to Standard Cell Design
- Performance can be very good - depend on quality of library and routing

Mask Programmable Gate Arrays

- Fabrication costs are cheaper than standard cell or full custom because the gate array wafers are mass produced
  - the non recurring engineering costs are lower because only a few (1-3) unique routing masks have to be created for each design.
- Fabrication time can be much shorter (1-2 weeks) because the wafers are already created and are only missing the routing layers.
- The more routing layers, the higher the cost, the longer the fabrication time, but the better usage of the available transistors on the gate array.
- Almost all high volume production of complex digital designs are done in either Standard Cell or Gate Array
  - Gate arrays used to be more popular, but now Standard cells more common
(Field) Programmable Logic

- Logic devices which can be programmed/configured on the desktop.
- Three families (in increasing density)
  - PALs (Programmable Array Logic), PLAs, GPLAs Programmable Logic Devices
  - Complex PLDs (CPLDs)
  - Field Programmable Gate Arrays (FPGAs)
- It should be noted that memories are the earliest type of programmable logic (PROMs, EPROMs, EEPROMs, Flash)

First Generation Programmable Logic Devices (PLDs)

- **Fixed AND array (decoder)**
- **Programmable OR array**

(a) Programmable read-only memory (PROM)

- **Programmable AND array**
- **Fixed OR array**

(b) Programmable array logic (PAL)

- **Programmable AND array**
- **Programmable OR array**

(c) Programmable logic array (PLA)

Fig. 7-13 Basic Configuration of Three PLDs
Programmable Gate Notation

- Represents a Variable Multi-input Gate
- Unused Gate Inputs are Not Present Logically

ROM Implementation of Logic Function

© 2002 Prentice Hall, Inc.
M. Morris Mano
DIGITAL DESIGN, 3e
PALs and PLAs

• An early type of programmable logic - still in common use today.
• Logic is represented in SOP form (Sum of Products)
• The number of PRODUCTS in an SOP form will be limited to a fixed number (usually 4-10 Product terms).
• The number of VARIABLES in each product term limited by number of input pins on PLD (usually a LOT, minimum of 10 inputs)
• The number of independent functions limited by number of OUTPUT pins.

PAL Structure

AND gates inputs

© 2002 Prentice Hall, Inc.
M. Morris Mano
DIGITAL DESIGN, 3e

Fig. 7-16 PAL with Four Inputs, Four Outputs, and Three-Wide AND-OR Structure
PAL Implementation

PLA Structure with ‘Programmable Inverters’

Give the Boolean expression for $F_1$ and $F_2$
Example of a PLA

$$F(A, B, C) = A \oplus B \oplus C$$
$$= \overline{AB} \overline{C} + \overline{A}BC + A\overline{B}C + ABC$$

$$G(A, B, C) = AB + AC + BC$$

Sequential Programmable Logic Device (SPLD)

Fig. 7-18  Sequential Programmable Logic Device
Typical SPLD Macrocell

Fig. 7-19 Basic Macrocell Logic

© 2002 Prentice Hall, Inc.
M. Morris Mano
DIGITAL DESIGN, 3e.
Complex PLDs

- What is the next step in the evolution of programmable logic?
  - More gates!
- How do we get more gates? We could put several PALs on one chip and put an interconnection matrix between them!!
  - This is called a **Complex PLD (CPLD)**.

---

**Logic Block Diagram**

**Cypress CPLD**

Programmable interconnect matrix.

Each logic block is similar to a 22V10.

*Figure 1. Ultra37128 Block Diagram*
Other Approaches

Another approach to building a “better” PLD is place a lot of primitive gates on a die, and then place programmable interconnect between them:

Field Programmable Gate Arrays

The FPGA approach is to arrange primitive logic elements (logic cells) arrange in rows/columns with programmable routing between them.

What constitutes a primitive logic element? Lots of different choices can be made! Primitive element must be classified as a “complete logic family”.

- A primitive gate like a NAND gate
- A 2:1 mux (this happens to be a complete logic family)
- A Lookup table (I.e, 16x1 lookup table can implement any 4 input logic function).

Often combine one of the above with a DFF to form the primitive logic element.
Other FPGA features

- Besides primitive logic elements and programmable routing, some FPGA families add other features
- Embedded memory
  - Many hardware applications need memory for data storage. Many FPGAs include blocks of RAM for this purpose
- Dedicated logic for carry generation, or other arithmetic functions
- Phase locked loops for clock synchronization, division, multiplication.
- Embedded Processor Cores
  - (ARM, Altera Excaliber, PowerPC, Xilinx Virtex-II)
Other FPGA Comments

• Performance is usually several factors to an order of magnitude lower than standard cell.
• Performance depends heavily on quality of FPGA technology.
• Design time advantages are the same as for standard cell (use same type of cell/macro library).
• Densities are an order of magnitude lower than standard cell but an order of magnitude higher than 1st Generation PLDs (no Programmable Interconnect).
• Very good for prototype design because many FPGAs are re-usable.
  – Can be used to prototype and verify designs before investing in technologies with high start-up costs (e.g. full custom).
  – Also used for dedicated Emulation platforms.

Programmability Options

• PLDs, CPLDs, and FPGAs have different types of programmability.
• One time programmable (OTP): Part is programmed once and holds its programming "forever". Not reusable, but usually the cheapest.
• UV-Erasable: Erasable with UV light. Needs a ceramic package with window; package adds expense to part.
  – Programming retained after power down.
  – Programming/Erasing limited to 1000s of cycles.
• Electrically Erasable: Both reprogramming and erasing is electrical.
  – Part can programmed/erased on circuit board, no special packaging needed.
  – Erase time much faster than UV erase.
  – Programming retained after power down.
  – Programming/Erasing limited to 1000s of cycles.
Programmability Options (cont.)

• Static Random Access Memory (SRAM) Programming:
  – Configuration bits are stored in SRAM. Can be reprogrammed infinite number of times.
  – Programming contents NOT retained after power down; FPGA must be 'configured' everytime on power up.
  – External non-volatile memory device required to hold device programming; on power up contents of external device transferred to FPGA to configure the device.
  – Altera, Xilinx (and other) corporations offer these types of FPGAs.

• Highest density FPGAs typically use SRAM for basic logic cells.

Comparing Technologies - Density (gates per chip)

• Highest to lowest density: Full Custom, Standard Cell, Mask Programmable Gate Array, FPGA, CPLD, PLD

• Full Custom, Standard Cell, Mask Programmable Gate Array are called ASIC technologies (Application Specific Integrated Circuit).

• Density gap between ASIC technologies and Programmable logic technologies (FPGAs, CPLD, PLD).

• Highest end FPGA density blurred with low-end ASIC density (i.e., hundreds of thousands of gates with embedded SRAMs).
Comparing Technologies - Speed

- Highest to lowest performance: Full Custom, Standard Cell, Mask Programmable Gate Array, PLDs, CPLDs, FPGAs.
- Performance gap between ASIC technologies and programmable technologies; however this is decreasing
- Performance of programmable technologies is in reverse order of their densities.

Comparing Technologies - Cost

- Depends heavily on volume. If only need a few hundred, then FPGAs can be cheaper. If need thousands, then ASIC technologies are cheaper.
- NRE cost (non-recurring engineering costs) are higher for ASIC technologies than FPGAs
- Per-unit-cost (chip cost) higher for FPGAs
Summary

- Full custom can give best density and performance
- Faster design time and ease of design are principle advantages of gate array and standard cell over full custom.
- Fast fabrication time and lower cost are principle advantages of mask programmable gate arrays over standard cell.
- Mask Programmable Gate arrays offer higher density over FPGAs/CPLDs and are cheaper in volume production.
Summary (cont.)

- FPGAs/CPLDs principle advantage over MP gate arrays is 'instant' fabrication time (programmed on desktop, “field”).
- FPGAs/CPLDs are also cheaper than gate arrays in low volume.
  - Densities ~100's of thousands of gates/chip.
  - Can be used to prototype full custom/standard cell designs.
- PLDs (and CPLDs) still hold a speed advantage over most FPGAs, they are useful primarily for high speed decoding and speed critical glue logic.
- “FPGA” is Commonly used Acronym for both CPLD and FPGA style Architectures.