Chapter 1: Introduction to VLSI Physical Design

Sadiq M. Sait & Habib Youssef

King Fahd University of Petroleum & Minerals
College of Computer Sciences & Engineering
Department of Computer Engineering

September 2003

Introduction

Present day VLSI technology permits us to build systems with hundreds of thousands of transistors on a single chip.

For example:

- The Intel 80286 microprocessor has over 10^5 transistors.
- The 80386 has 275,000 transistors.
- The 80486 has approximately 10^6, transistors.
- The RISC processor from National Semiconductor NS32SF641 has over 10^6 transistors.
- The Pentium processor has over 3×10^6 transistors.

Introduction-contd

ICs of this complexity would not have been possible without computer programs which automate most design tasks.

- Designing a VLSI chip with the help of computer programs is known as CAD.
- Design Automation (DA), on the other hand, refers to entirely computerized design process with no or very little human intervention.
- CAD and DA research has a long history of over three decades.

Introduction-contd

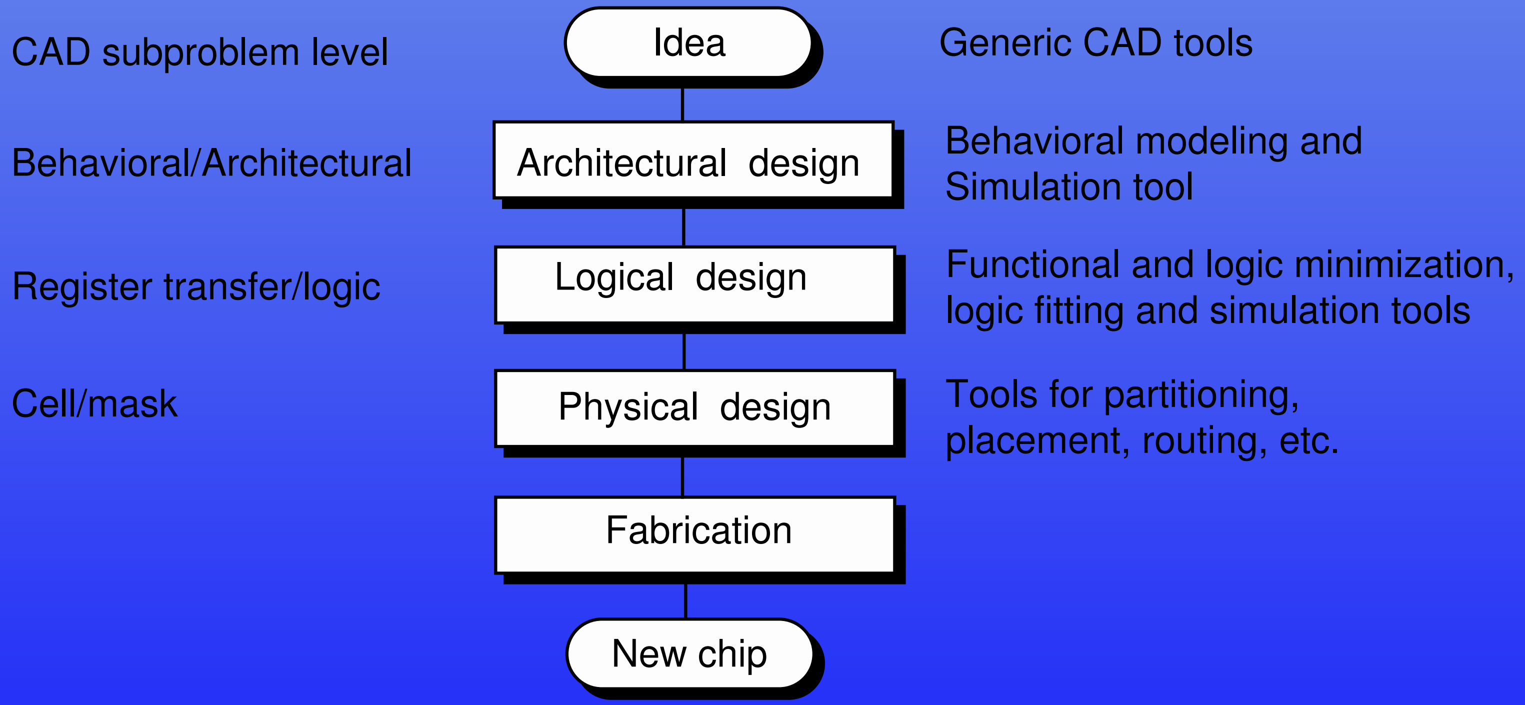
As technology has changed from SSI to VLSI:

- The demand for DA has escalated.
- The *types* of DA tools have multiplied due to changing needs.
- There has been a radical change in design issues.
- *Due to sustained research by a number of groups, sophisticated tools are available for designing ICs, and we are briskly moving towards complete DA.*

Physical Design

- Physical design is the process of generating the final layout for the circuit.
- This is a very complex task.
- In order to reduce the complexity several intermediate levels of abstractions are introduced.
- More and more details are introduced as the design progresses from highest to lowest levels of abstractions.
- Typical levels of abstractions together with their corresponding design steps are illustrated in Figure.

Levels of abstraction



CAD subproblem level	Idea	Generic CAD tools
Behavioral/Architectural	Architectural design	Behavioral modeling and Simulation tool
Register transfer/logic	Logical design	Functional and logic minimization, logic fitting and simulation tools
Cell/mask	Physical design	Tools for partitioning, placement, routing, etc.
	Fabrication	
	New chip	

Figure 1: Levels of abstraction & corresponding design step.

Logical & Architectural Design

- As indicated the design is taken from specs to fabrication step by step with the help of CAD tools.
- Architectural design of a chip is carried out by expert human engineers.
- Decisions made at this stage affect the cost and performance of the design significantly.
- Once the system architecture is defined, it is necessary to carry out two things:

  (a) Detailed logic design of individual circuit modules.

  (b) Derive the control signals necessary to activate and deactivate the circuit modules.

- The first step is known as *data path design*.
- The second step is called *control path design*.

Example

It is required to design an 8-bit adder. The two operands are stored in two 8-bit shift registers A and B. At the end of the addition operation, the sum must be stored in A. The contents of B must not be destroyed. The design must be as economical as possible in terms of hardware.

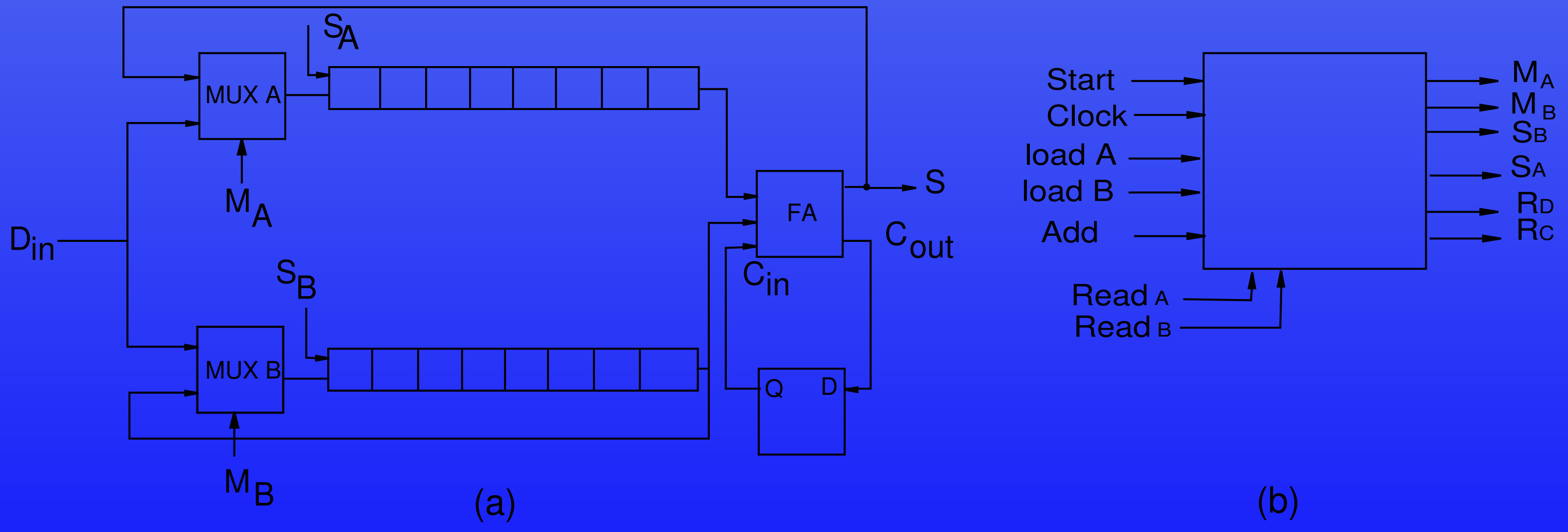


Figure 2: Organization of a serial adder: (a) Data Path (b) Control path block diagram.

Example-contd

- There are numerous ways to design the above circuit.
- Since it is specified that the hardware cost must be minimum, it is perhaps best to design a serial adder.
- The organization of such an adder is shown in figure.
- The relevant control signals are tabulated below.

S_A	Shift the register A right by one bit
S_B	Shift the register B right by one bit
M_A	Control multiplexer A
M_B	Control multiplexer B
R_D	Reset the D flip-flop
R_C	Reset the counter
$START$	A control input, which & commences the addition

Example-contd

- The control algorithm for adding A and B is given below.

```
forever do
        while (START = 0) skip;
        Reset the D flip-flop and the counter;
        Set $M_A$ and $M_B$ to 0;
        repeat
                Shift registers $A$ and $B$ right by one;
                counter = counter + 1;
        until counter = 8;
```

High-level Synthesis

- Several observations can be made by studying the example of the serial-adder.
- First, note that designing a circuit involves a trade-off between cost, performance, and testability.
- The serial adder is cheap in terms of hardware, but slow in performance.
- It is also more difficult to test the serial adder, since it is a sequential circuit.
- The parallel 8-bit CLA is likely to be fastest in terms of performance, but costliest in hardware.

High-level Synthesis-contd

- All the different ways that we can think of to build an 8-bit adder constitute what is known as the *design space* (at that particular level of abstraction).
- Each method of implementation is called a point in the design space.
- There are advantages and disadvantages associated with each design point.
- When we try optimizing the hardware cost, we usually lose out on performance, and vice versa.
- There are many more design aspects, such as *power dissipation*, *fault tolerance*, *ease of design*, and *ease of making changes* to the design.

High-level Synthesis-contd

- A circuit specification may pose *constraints* on one or more aspects of the final design.
- For example, when the specification says that the circuit operate at a minimum of 15 MHz, we have a constraint on the timing performance.
- *Given a specification, the objective is to arrive at a design which meets all the constraints posed by the specification, and optimizes on one or more of the design aspects.*
- This problem is also known as *hardware synthesis*.
- Computer programs have been developed for data path synthesis as well as control path synthesis.
- The automatic generation of data path and control path is known as *high-level synthesis*.

Logic Design

- The data path and control path will have components such as arithmetic/logic units, shift registers, multiplexers, buffers, etc.
- Further design steps depend on the following factors.
  1. How is the circuit to be implemented, on a PCB or as a VLSI chip?
  2. Are all the components available as off-the-shelf ICs circuits or as predesigned modules?
- If the circuit must be implemented on a PCB using off-the-shelf components, then the next stage is to select the components.

Logic Design-contd

- Following this, the ICs are placed on boards and the necessary interconnections are established.
- A similar procedure may be used in case the circuit is implemented on a VLSI.
- These modules are also known as *macro-cells*.
- The cells must be placed on the layout surface and wired together using metal and polysilicon (poly) interconnections.

Physical Design

- Physical design of a circuit is the phase that precedes the fabrication of a circuit.
- In most general terms it refers to all synthesis steps succeeding logic design and preceding fabrication.
- These include all or some of the following steps:
  1. Circuit Partitioning.
  2. Floorplanning and Channel Definition.
  3. Circuit Placement.
  4. Global Routing.
  5. Channel Ordering.
  6. Detailed routing of power and ground nets.
  7. Channel and Switchbox Routing.

Physical Design-contd

- The performance of the circuit, its area, its yield, and its reliability depend on the layout.
- Long wires and vias affect the performance and area of the circuit.
- The area of a circuit also has a direct influence on the $yield$ of the manufacturing process.

Layout Styles

- These approaches differ mainly in the structural constraints they impose on the layout elements and the layout surface.
- They belong to two general classes:
  (a) The full-custom layout approach.
  (b) The semi-custom approaches.
- Current layout styles are:
  1. Full-custom.
  2. Gate-array design style.
  3. Standard-cell design style.
  4. Macro-cell (Building block layout).
  5. PLA (Programmable Logic Array), and
  6. FPGA (Field Programmable Gate-Array) layout.

Full-Custom Layout

- *Full-custom layout* refers to manual layout design.
- Full-custom design is a time-consuming and difficult task.
- Gives full control to the artwork designer in placing/interconnecting.
- A high degree of optimization in both the area and performance is possible.
- Takes several man-months to layout a chip.
- Therefore is used only for circuits that are mass produced (microprocessors).
- For circuits which will be reproduced in millions, it is important to optimize on the area as well as performance.
- Designers productivity is increased with the help of a good layout editor.

Gate-Array Layout

- A gate-array (MPGAs) consists of transistors prefabricated on a wafer in the form of a regular 2-D array.
- Initially the transistors in an array are not connected to one another.
- In order to realize a circuit on a gate-array, metal connections must be placed using the usual process of masking (*personalizing*).
  - Short fabrication time (only four processing steps).
  - Low cost of production due to yield.
  - Limited wiring space, therefore present difficulties to automatic layout generator.

Gate-Array Layout-contd

- A special case of the gate-array architecture is when routing channels are very narrow, or virtually absent is called Sea of Gates (Channel-less Gate-Arrays).
- A typical gate-array cell personalized as a two-input NAND gate is shown in Figure.

Gate-Array Layout-contd

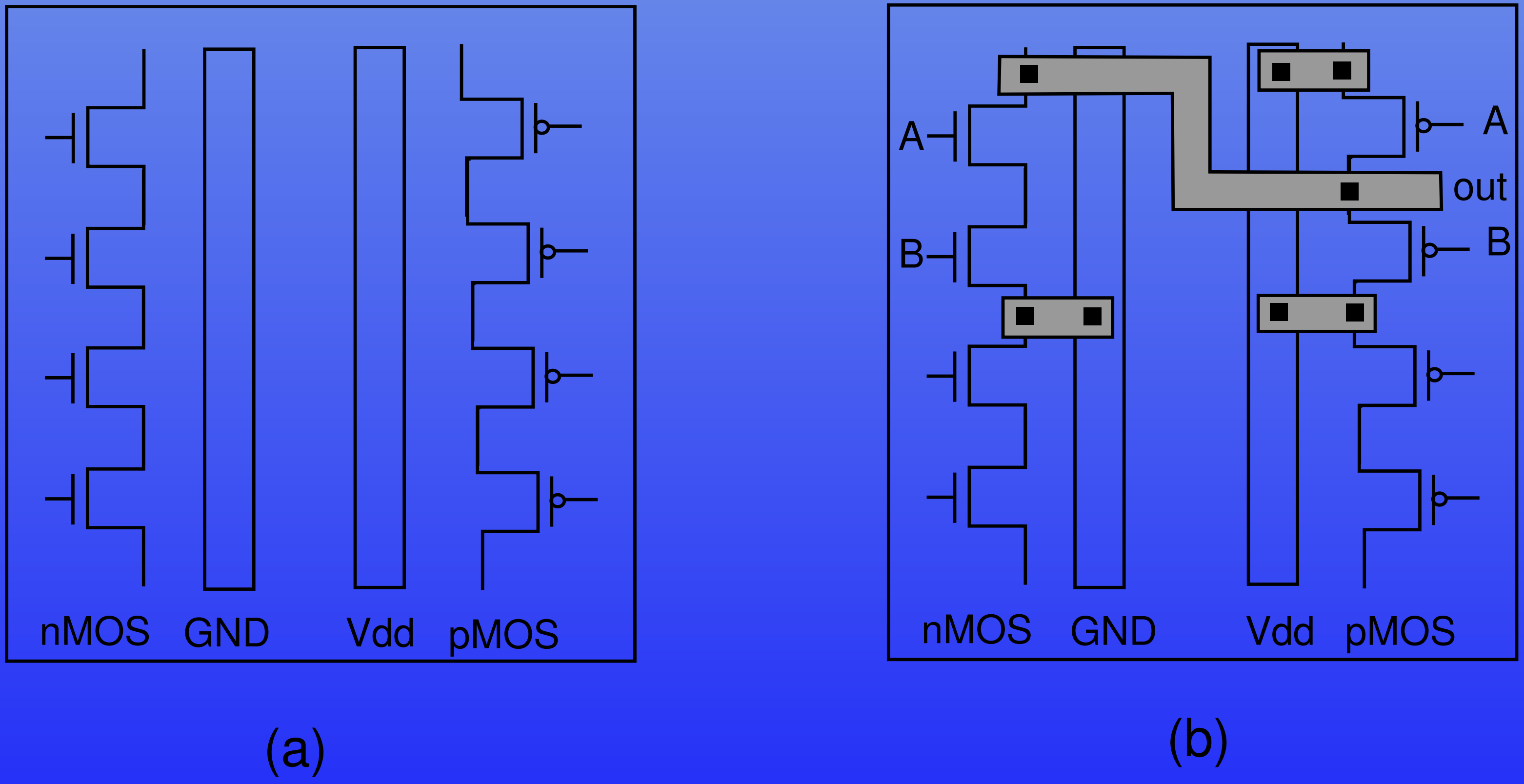


Figure 3: (a) Example of a basic cell in a gate array (b) Cell personalized as a two-input NAND gate.

Gate-Array Layout-contd

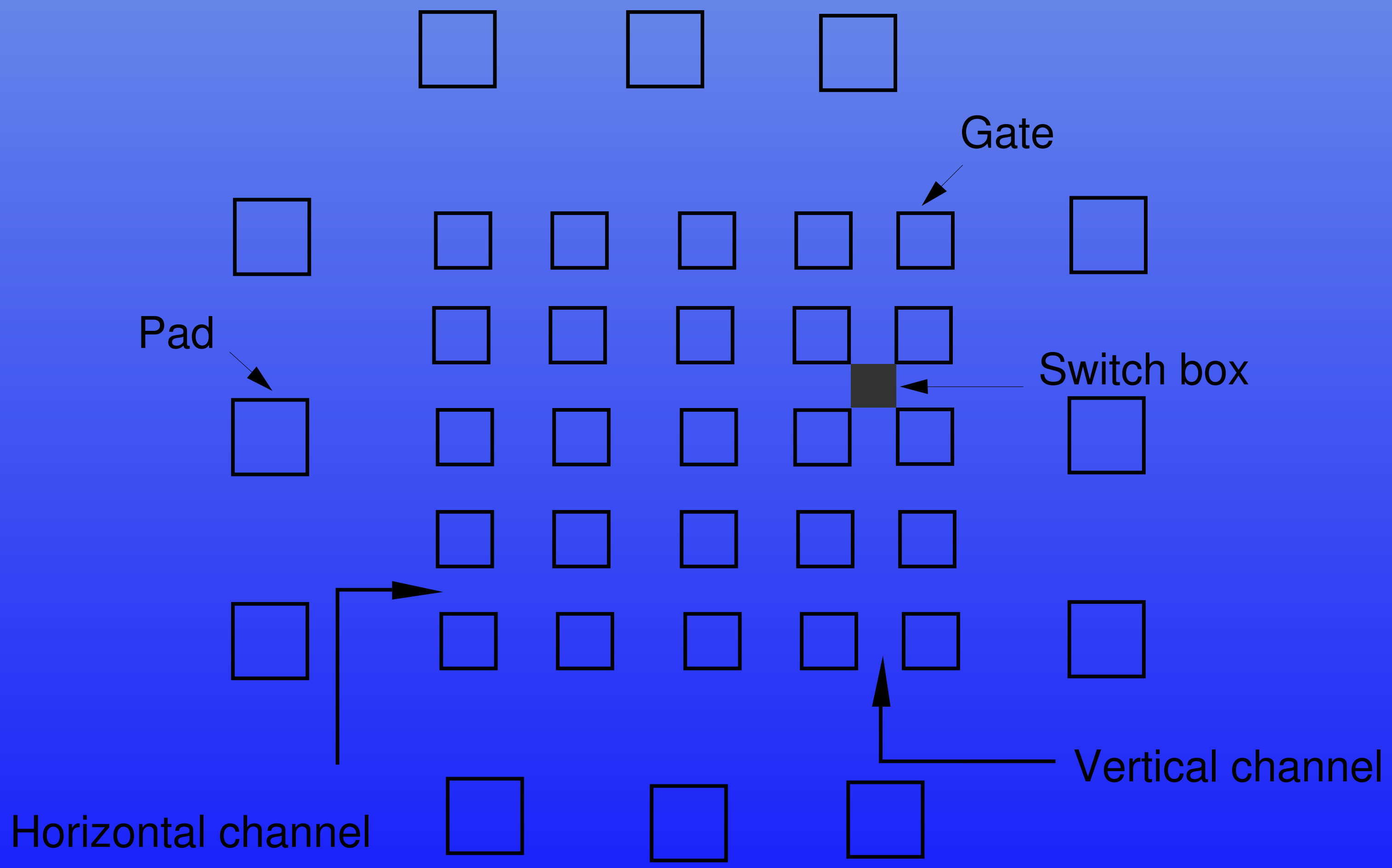


Figure 4: A gate array floor plan.

Standard Cell Layout

- A standard cell, known also as a polycell, is a logic that performs a standard function.
- Examples of standard-cells are two-input NAND gate, two-input XOR gate, D flip-flop, two-input multiplexer.
- A cell library is a collection of information pertaining to standard-cells.
- The relevant information about a cell consists of its name, functionality, pin structure, and a layout for the cell in a particular technology such as $2\mu m$ CMOS.
- Cells in the same library have standardized layouts, i.e., all cells are constrained to have the same height.
- Description of an inverter logic module named `i1s`. Focus only on the `profile`, `termlist`, and `siglist` statements.

Standard Cell Layout

```
cell begin i1s generic=i1 primitive=INV area=928.0 transistors=2
 function="q = INV((a))"
 logfunction=invert
 profile top (-1,57) (15,57); profile bot (-1,-1) (15,-1);
 termlist
 a { (1-4,-1) (1-4,57) }
 pintype=input
 rise\_delay=0.35 rise\_fan=5.18  fall\_delay=0.28 fall\_fan=3.85
 loads=0.051 unateness=INV; q { (9-12,-1) (9-12,57) }
 pintype=output;
 siglist GND Vdd a q ;
 translist
 m0 a GND q length=2000 width=7000 type=n
 m1 a Vdd q length=2000 width=13000 type=p ;
 caplist
 c0 Vdd GND 2.000f
 c1  q  GND 5.000f
 c2 Vdd   a 2.000f
 c3  a GND 11.000f ;
 cell end i1s
```

Layout-Inverter Standard Cell

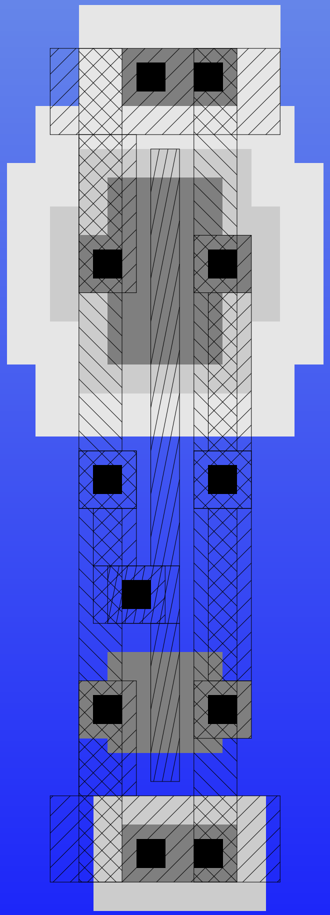


Figure 5: Layout of a standard-cell i1s.

Cell-Based Design

- Similar to designing a circuit using SSI and MSI components.
- Selection is now from a cell-library, and components are placed in silicon instead of PCB.
- Advantage: Designs can be completed rapidly.
- The Layout program will only be concerned with:

  (1) the location of each cell; and

  (2) interconnection of the cells.
- Placement and routing is again simplified using a standard floorplan (see Figure).
- The disadvantage in relation to gate-array is that all the fabrication steps are necessary to manufacture.

Cell-Based Design-contd

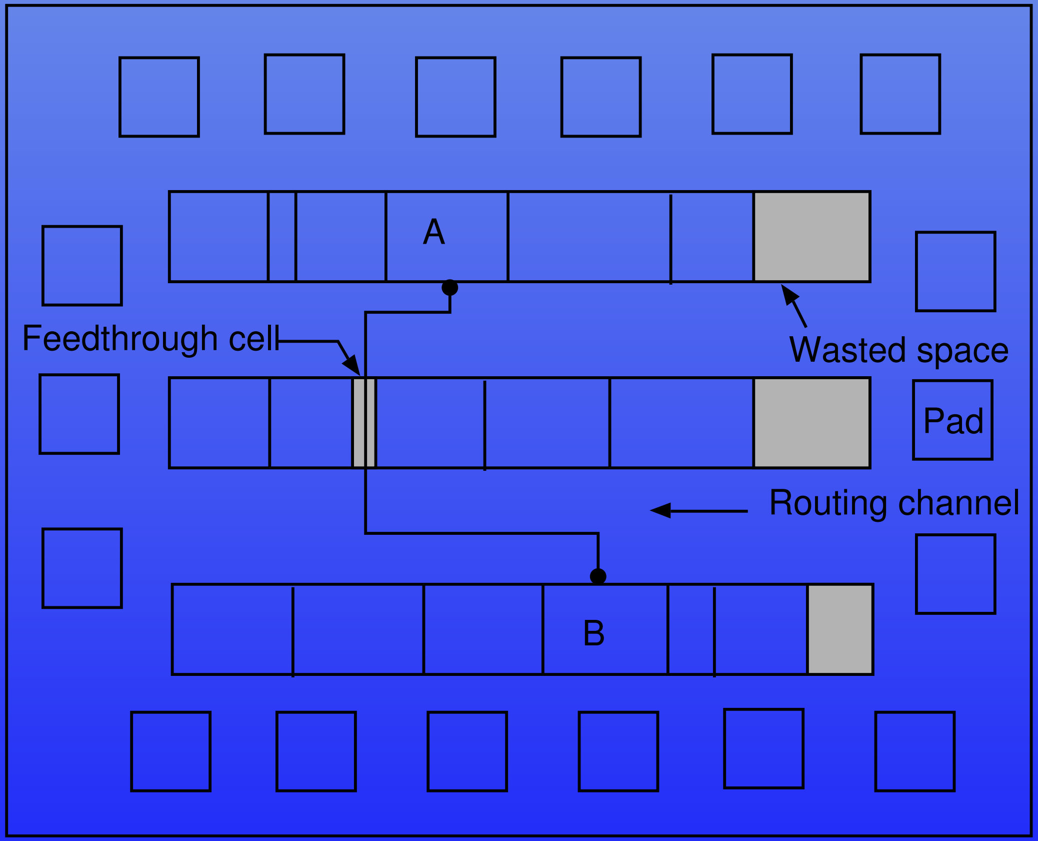


Figure 6: Floorplan of a standard-cell layout.

Macro-cell Layout

- No restrictions as in gate-array and standard-cell design.
- The cells can no longer be placed in a row-based floorplan.
- This design style is called *macro-cell* design style, or *building-block* design style.

Macro-cell Layout-contd

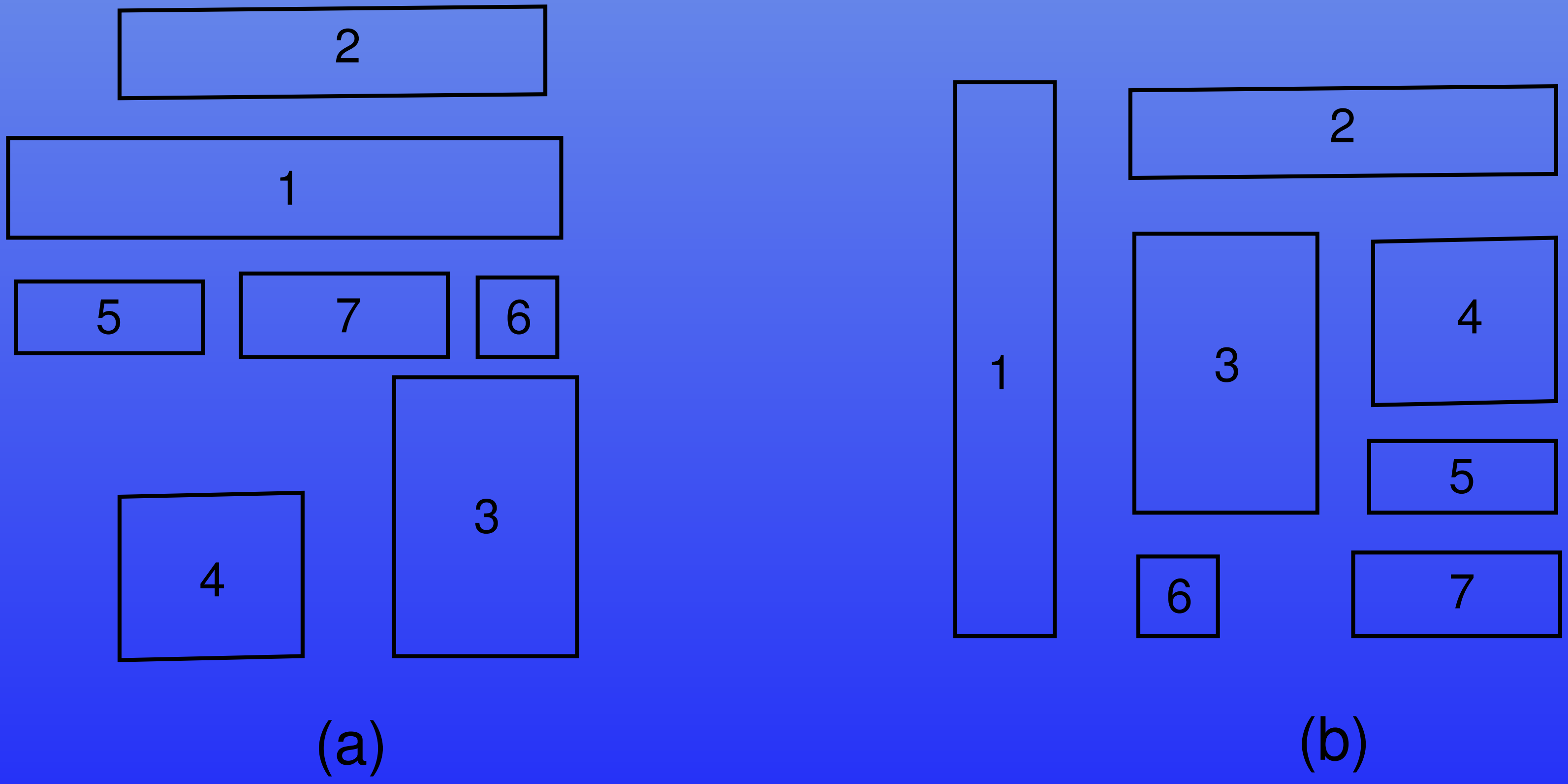


Figure 7: (a) Cells of varying heights and widths placed in a row-based floorplan (b) A more compact floorplan for the same circuit.

Macro-cell Layout-contd

Advantages:

- Cells of significant complexity are permitted in the library (registers, ALUs, etc).
- Building-block layout (BBL) comes closest to full-custom layout.
- Like standard-cell layout style, all the processing steps are required to manufacture a BBL chip.

Disadvantages:

- It is much more difficult to design layout programs for the BBL design style.
- This is because there is no standard floorplan to adhere to. As a result, the routing channels are not predefined either.
- Floorplanning and channel definition are additional steps required in a BBL layout system.

FPGA layout

- Similar to an MPGA, an FPGA also consists of a 2-D array of logic blocks.
- Each logic block (CLB) can be programmed to implement any logic function of its inputs.
- In addition to this, the channels or switchboxes contain interconnection resources.
- They contain programmable switches that serve to connect the logic blocks to the wire segments, or one wire segment to another.
- Furthermore, I/O pads are also programmable to be either input or output pads.

FPGA layout-contd

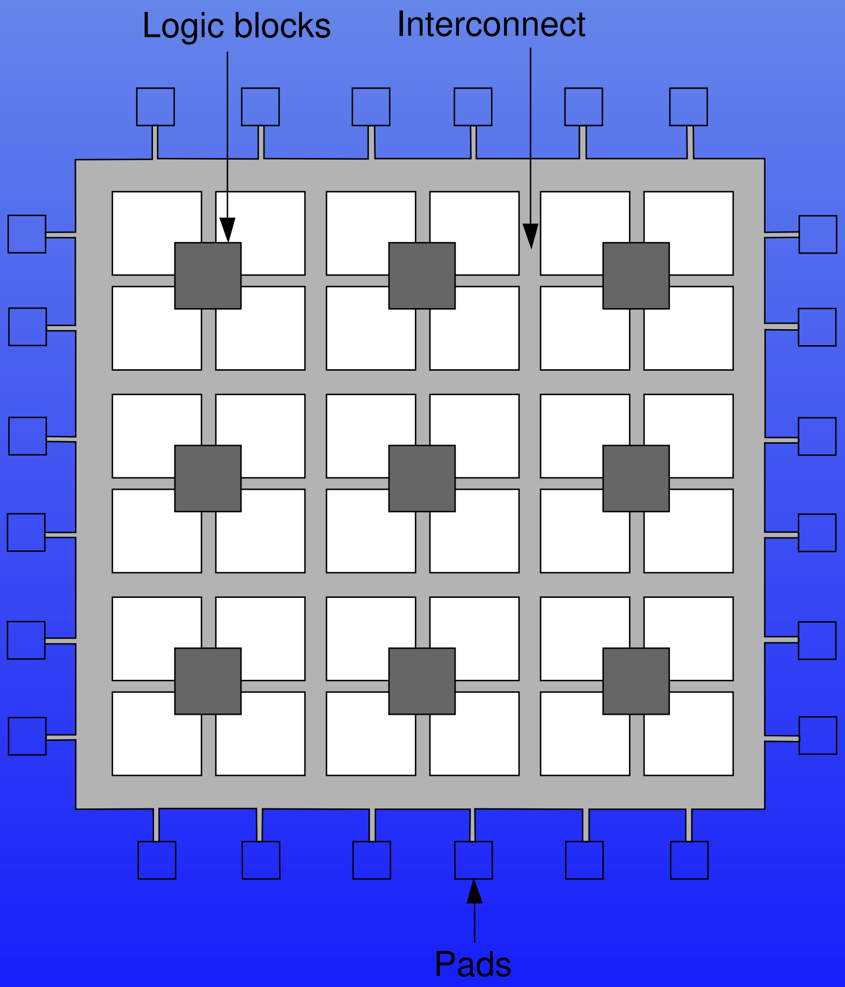


Figure 8: Diagram of a typical FPGA.

FPGA layout-contd

- The main design steps when using FPGAs to implement digital circuits are:
  1. Technology Mapping,
  2. Placement,
  3. Routing, and finally,
  4. Generating the bit patterns.
- The main disadvantages are their lower speed of operations and lower gate density.
- FPGAs are most ideally suited for prototyping applications, and implementation of random logic using PALs.

Difficulties in Physical Design

- Physical design is a complex optimization problem, involving several objective functions.
- Some of these objectives are conflicting.
- It is therefore customary to adopt a stepwise approach and subdivide the problem (e.g., Circuit Partitioning, Floorplanning, etc.).
- All of the aforementioned subproblems are constrained optimization problems.
- Unfortunately these layout subproblems (even simplified versions) are NP-Hard.
- Therefore, instead of optimal enumerative techniques, *heuristics* are used.
- Solution quality of heuristics is determined using artificial inputs whose optimal solutions are known or using test inputs comprising of real circuits, called *benchmarks*, (MCNC, ISCAS).

Terminology and Definitions

- A *cell*.
- A *macro-cell*.
- The *aspect ratio* of a cell.
- *Logic module*, or *module*, refers to macro or standard cells.
- A module interfaces to other modules through *pins*.
- A *signal net*, or simply *net*, is a collection of pins.
- A *netlist* description is, as the name suggests, a list of all the nets in the circuit.
- A graph is an abstract representation.
- The connectivity information can be represented in the form of an $n \times n$ matrix C.

Full-Adder

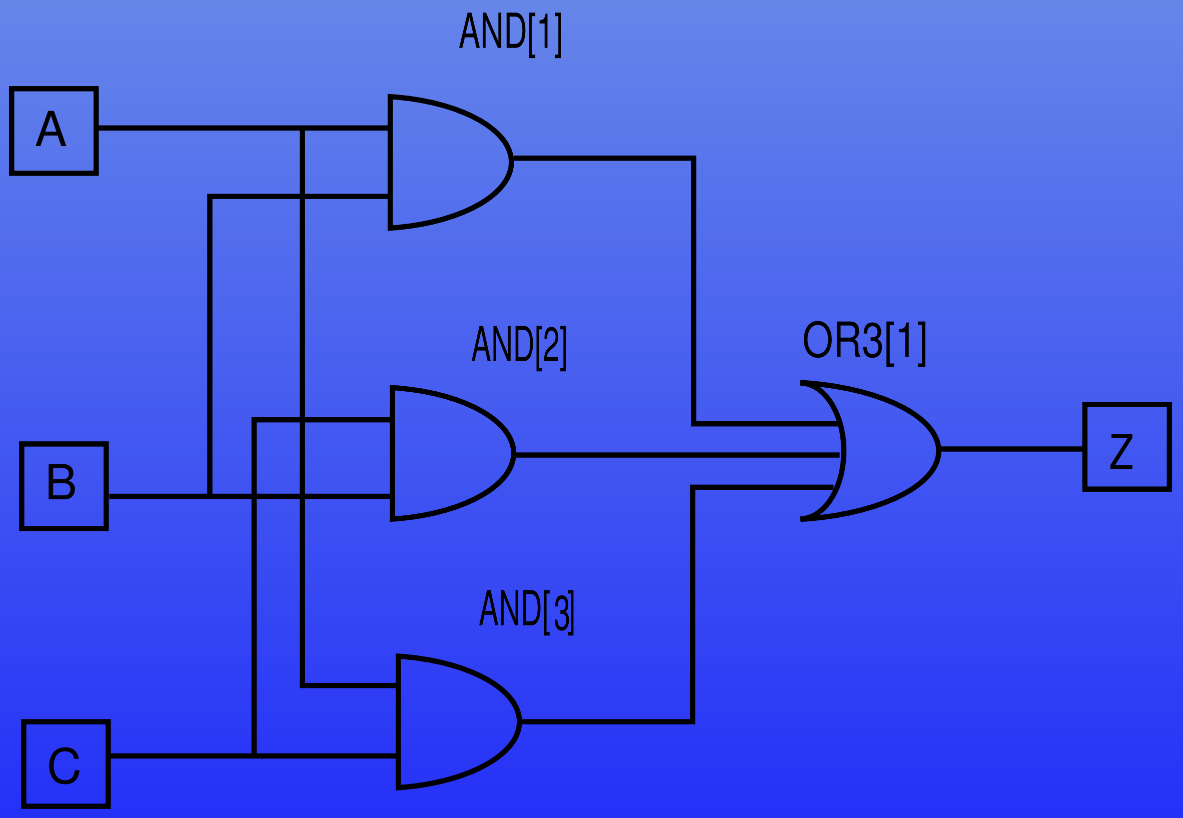


Figure 9: Logic diagram of ’full-adder carry’ circuit.

Full-Adder-contd

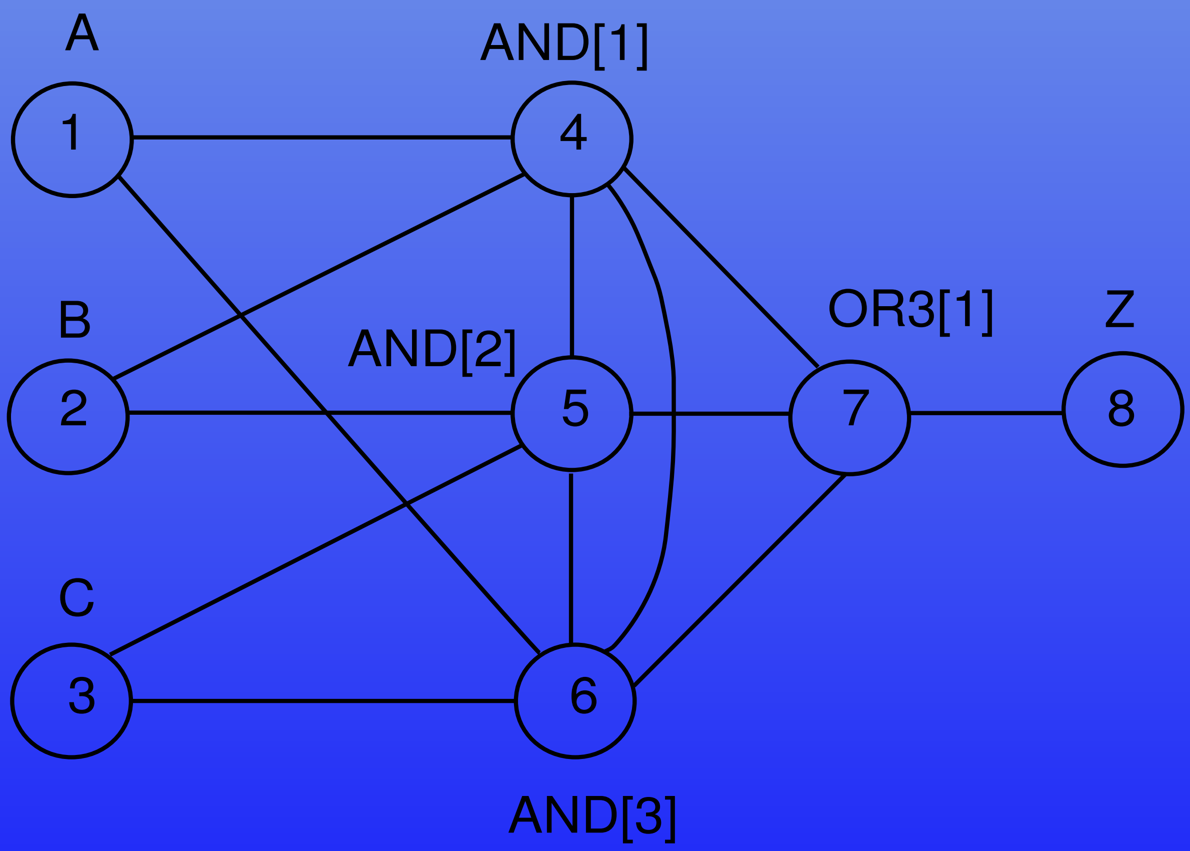


Figure 10: The connectivity graph for the full-adder carry circuit.

Full-Adder-contd

$$\begin{array}{c|cccccccc}
 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\
\hline
1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\
2 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\
3 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
4 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\
5 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 \\
6 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 \\
7 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\
8 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0
\end{array}$$

Figure 11: The connectivity matrix for the full-adder carry circuit.

Full-Adder-contd

- *Manhattan distance* between the two pins located at coordinates (x_1, y_1) and (x_2, y_2), is given by:

$$d_{12} = |x_1 - x_2| + |y_1 - y_2| \tag{1}$$

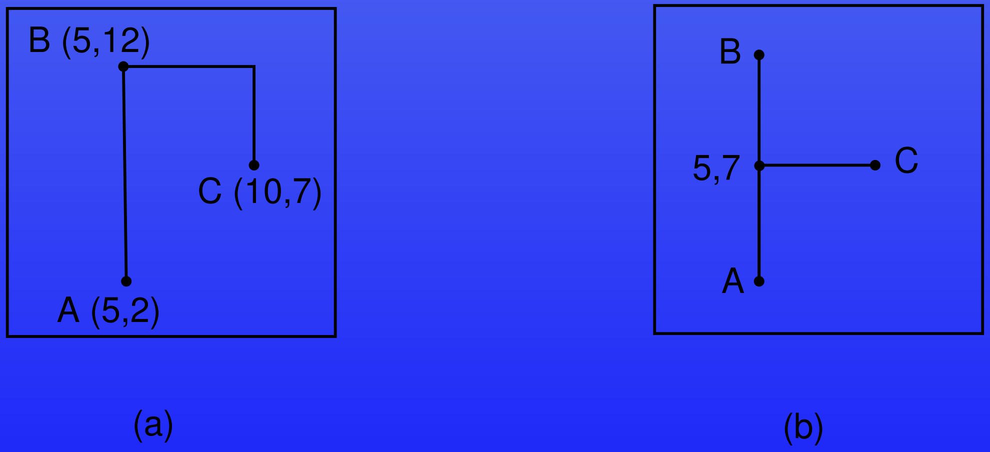


Figure 12: (a) A minimum spanning tree (b) Steiner tree.

Conclusion

- In this session we introduced the basic concepts of VLSI physical design and Design Automation.
- In order to automate the layout procedure, it is common to impose restrictions on the layout architecture.
- Several layout architectures are popular.
- In order to reduce the complexity the layout process is broken into a sequence of physical design phases.
- The important design phases were presented.
- Each of these phases amounts to solving a combinatorial optimization problem.

Conclusion

- Table below assesses and compares several aspects of the various layout styles.
  Comparison of Layout styles. The number in parentheses indicates a rank to grade the layout style in comparison to others in the same category.

Design Style	Appl	Design time	Fab Effort	Cost	Performance
F-Custom	High volume	High(4)	High(2)	High(4)	High(4)
Gate-array	ASICs	Low(2)	Low(1)	Low(2)	Low(4)
Stan-cell	ASICs	Low(3)	High(2)	Low(3)	High(1)
Macro-cell	General	High(1)	High(2)	Low(3)	High(2)
FPGA	ASICs	Low(1)	Nil	Low(1)	Low(1)