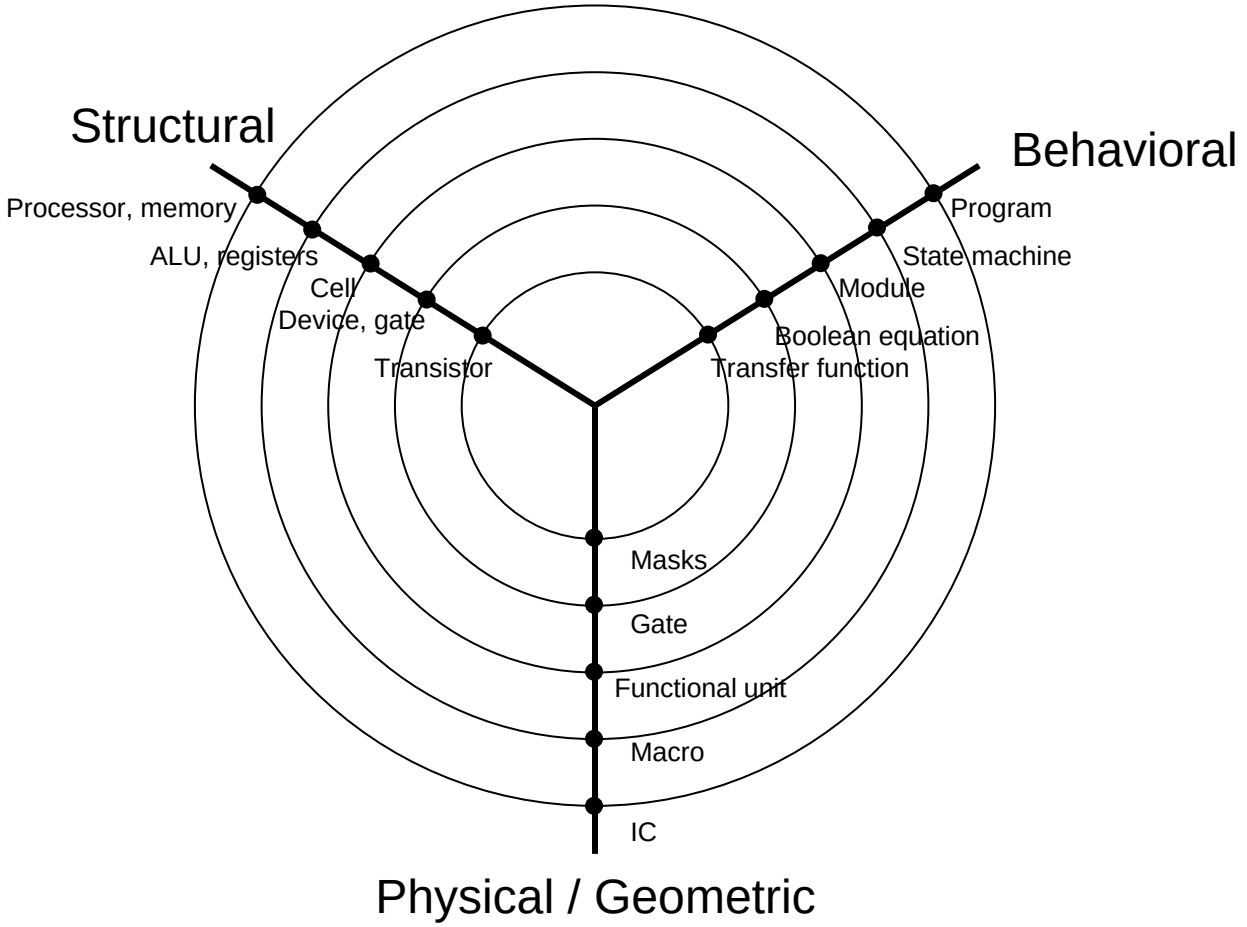
Disseny físic

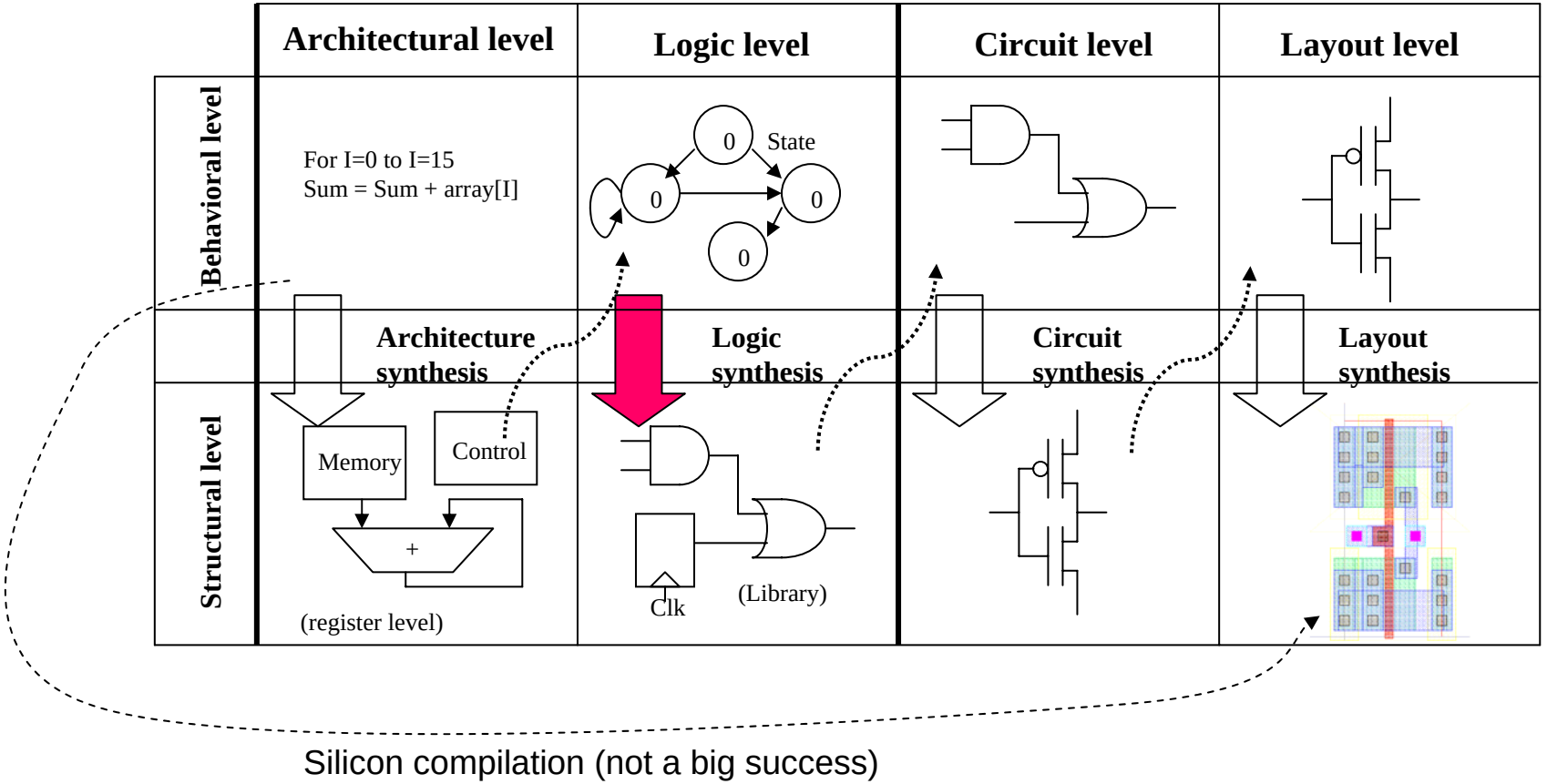
Disseny en *Standard Cells*

Enric Pastor
Rosa M. Badia
Ramon Canal
DM
Tardor 2005

Design domains (Gajski)

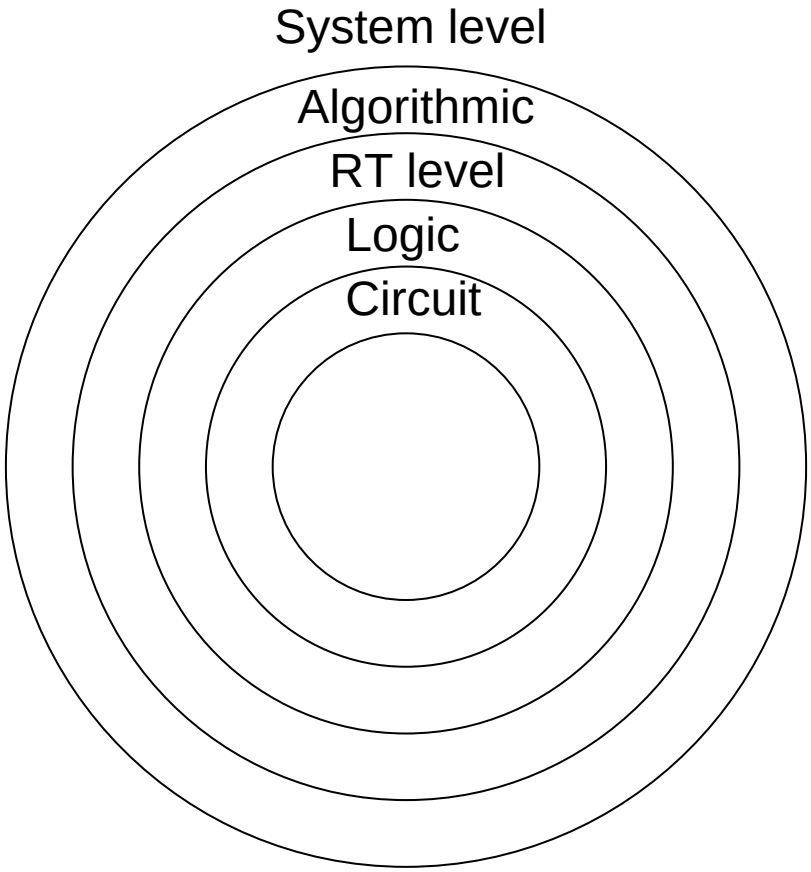


Abstraction levels and synthesis

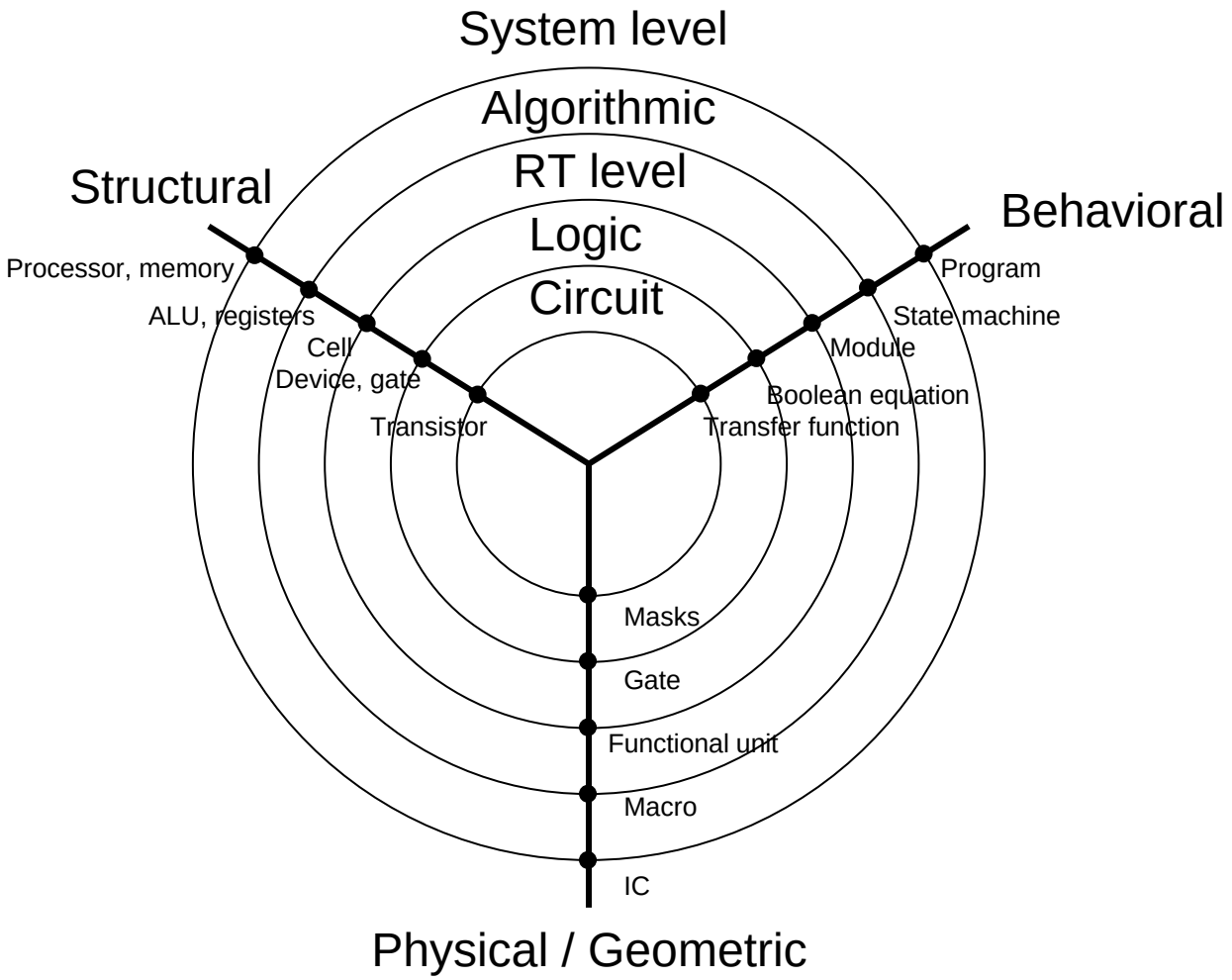


Silicon compilation (not a big success)

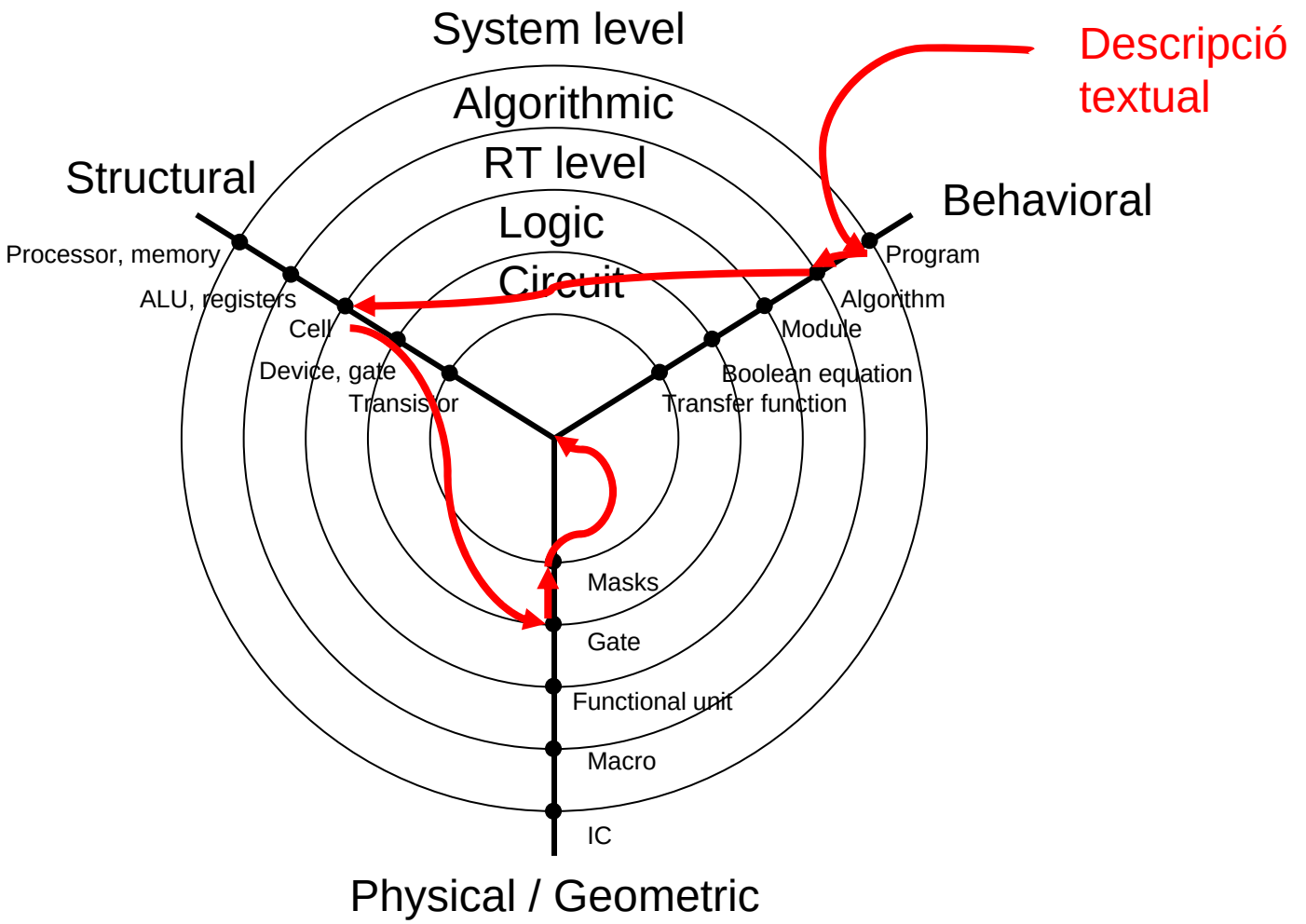
Design domains (Gajski)



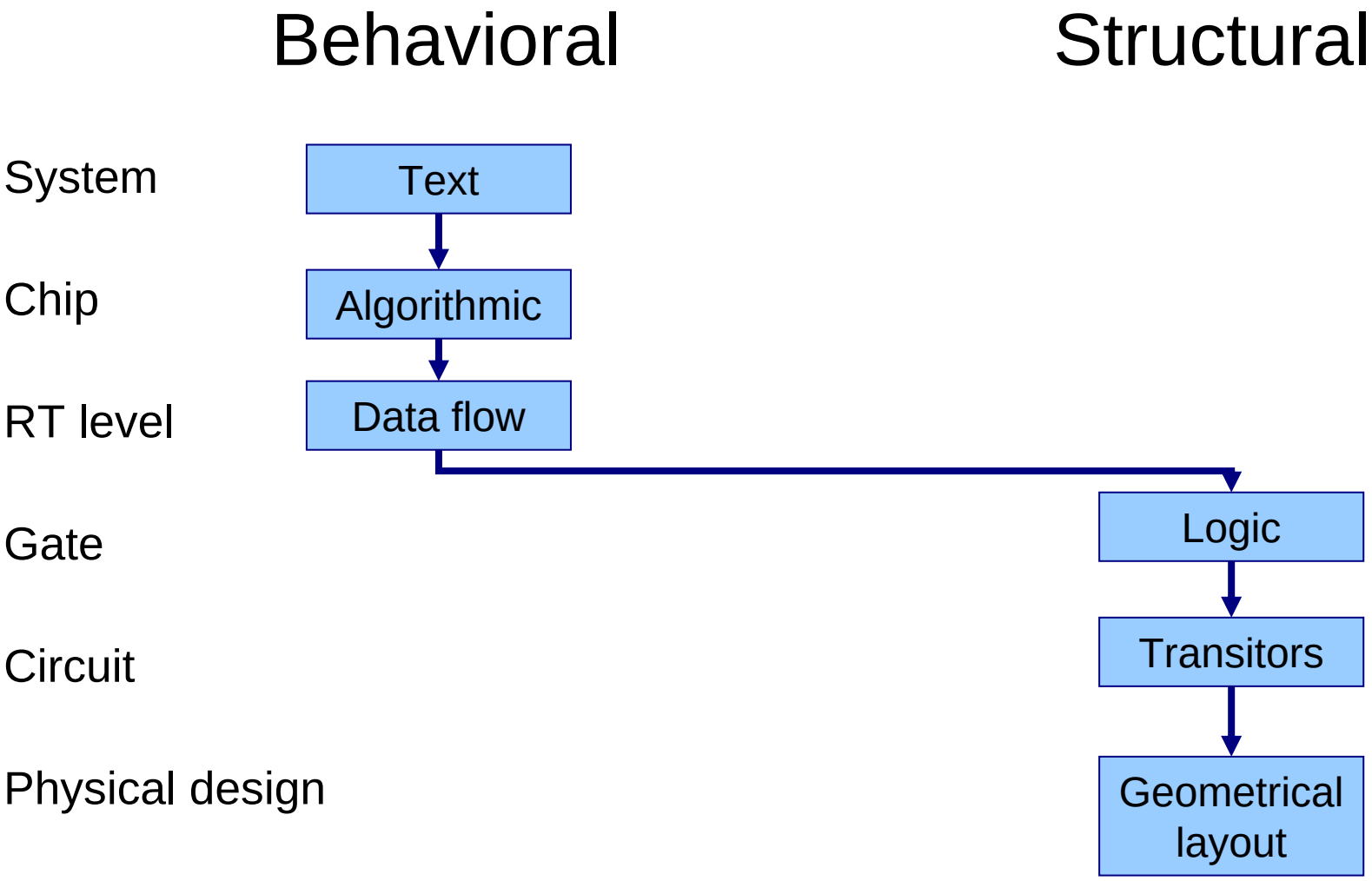
Design domains (Gajski i Kuhn)



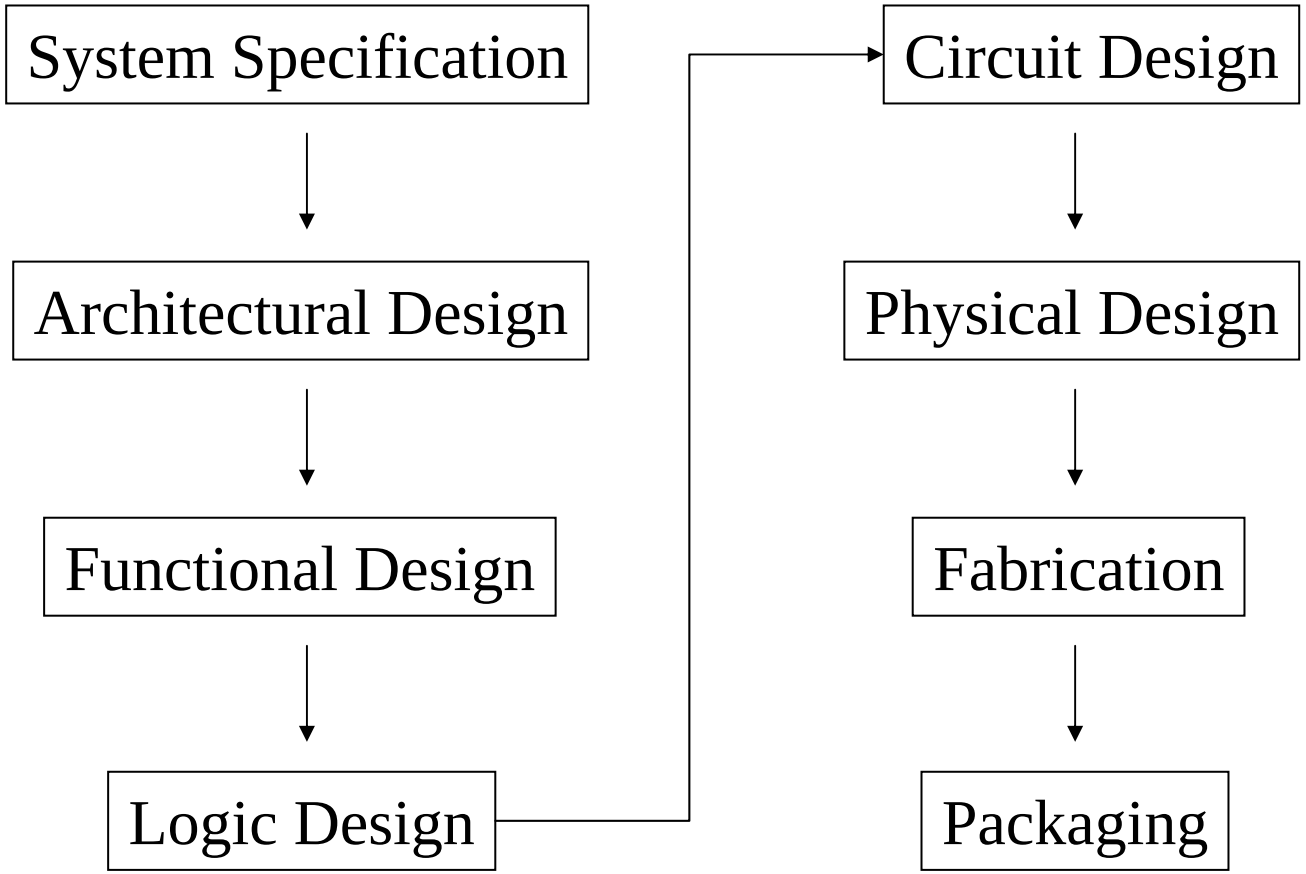
Design domains (Gajski i Kuhn)



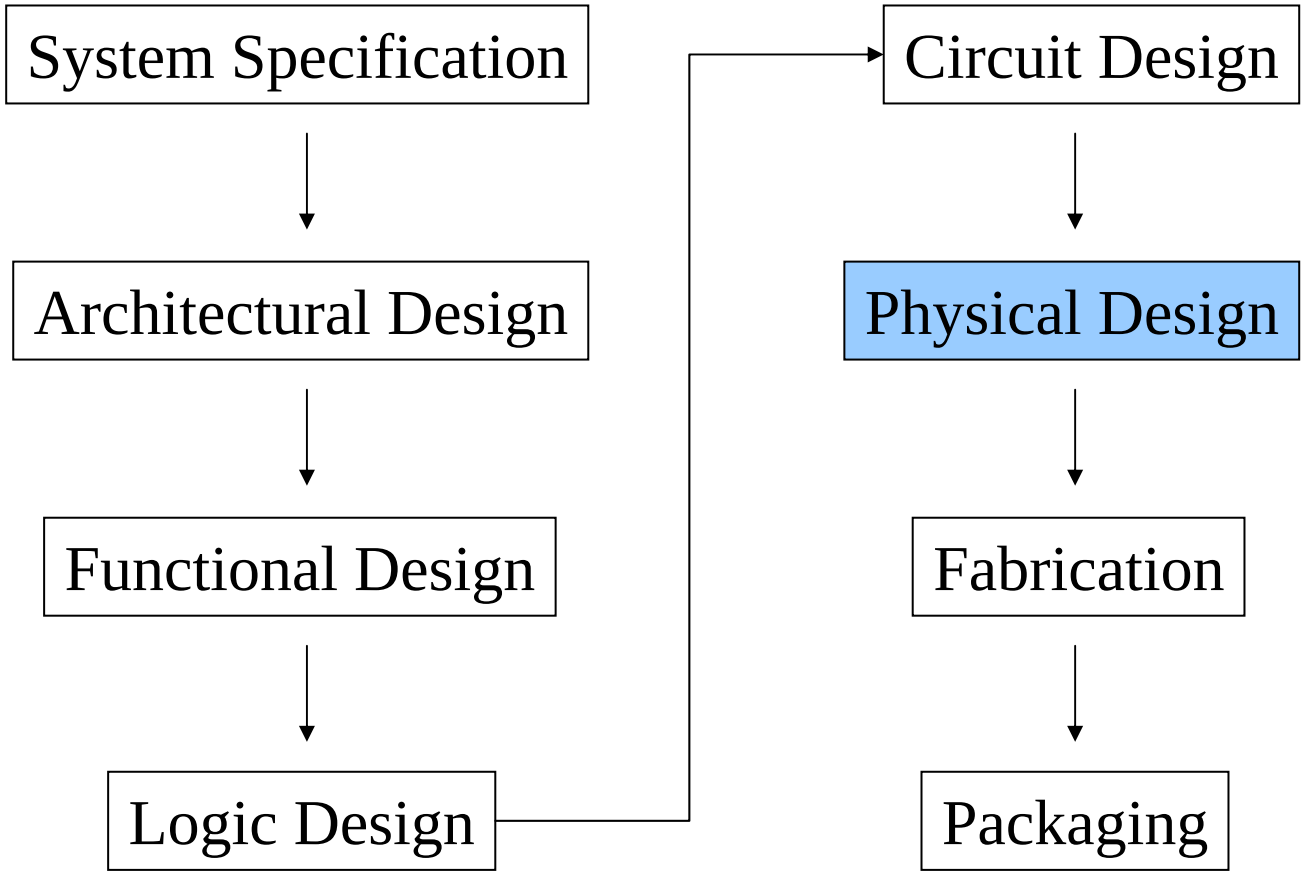
Typical Design Path



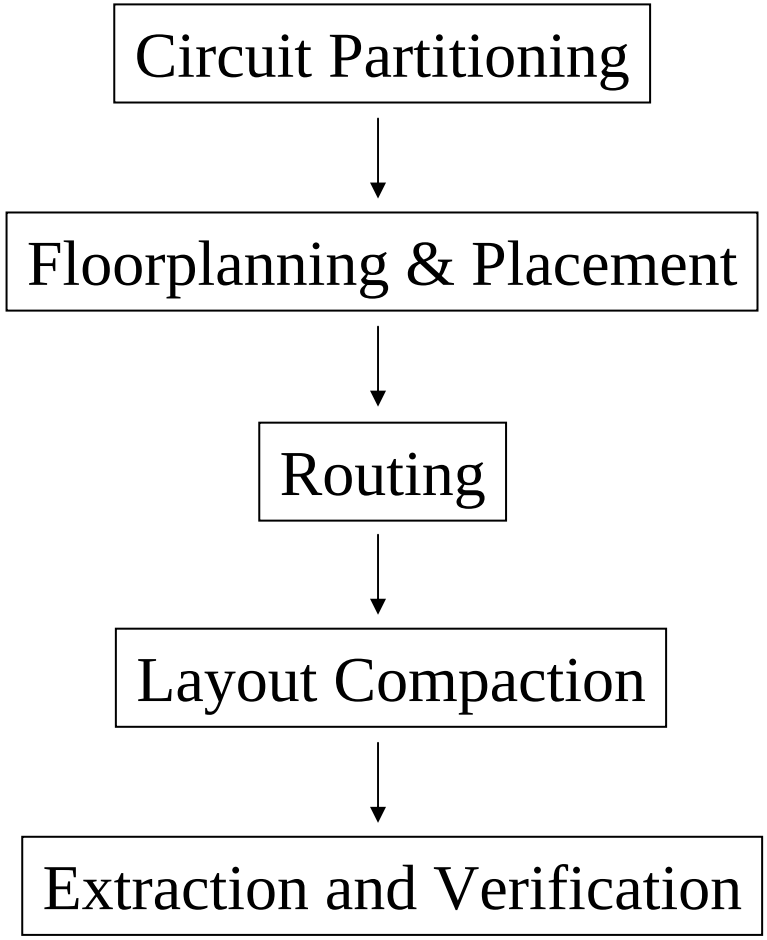
VLSI Design Cycle



VLSI Design Cycle

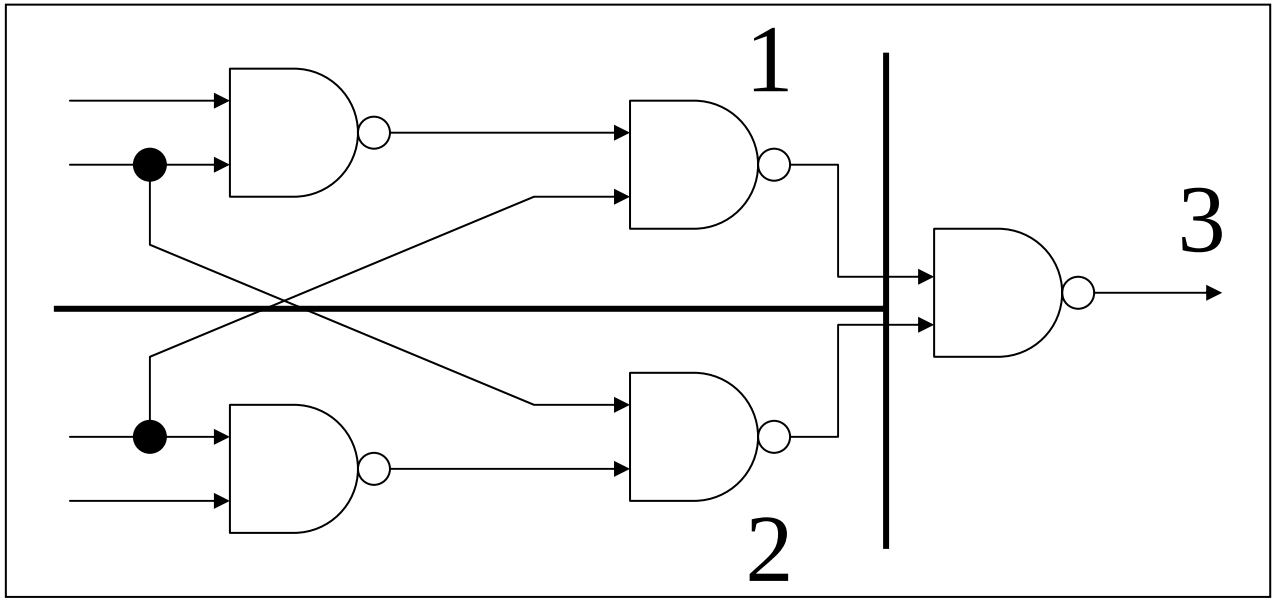


Physical Design Cycle



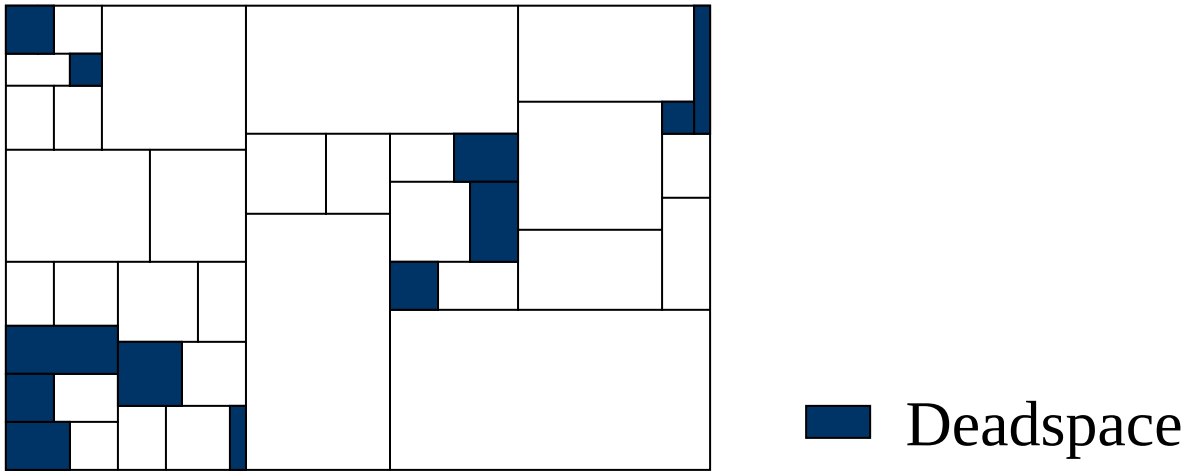
Physical Design Cycle

Circuit Partitioning – Partition a large circuit into sub-circuits (called blocks). Factors like #blocks, interconnections between blocks, … are considered.



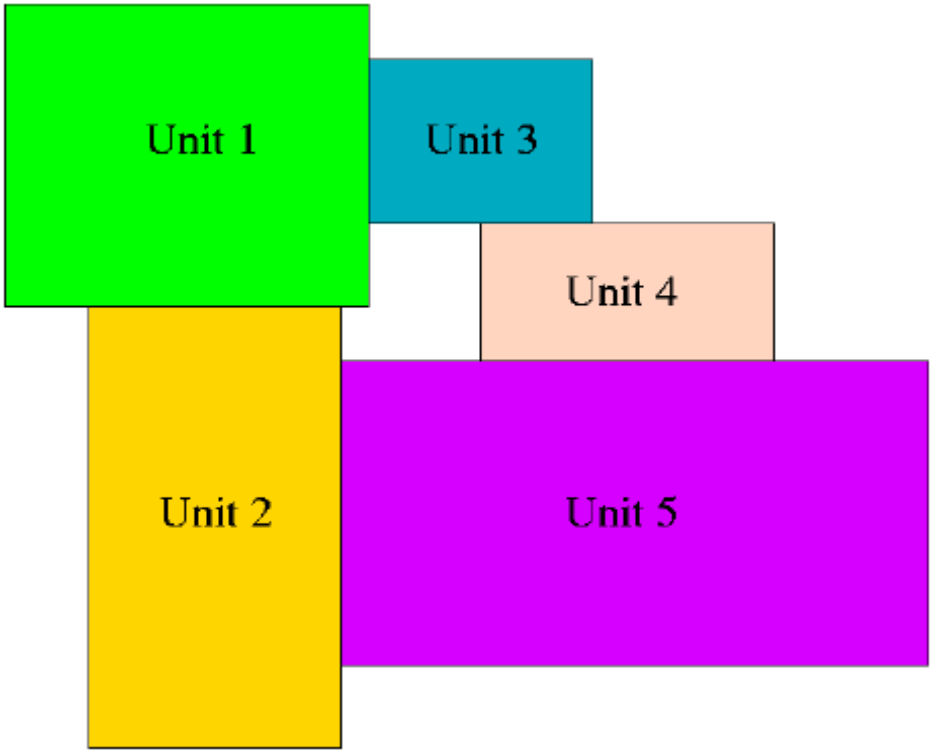
Physical Design Cycle

Floorplanning – Set up a plan for a good layout. Place the blocks at an early stage when details like shape, area, positions of I/O pin, … are not yet fixed.



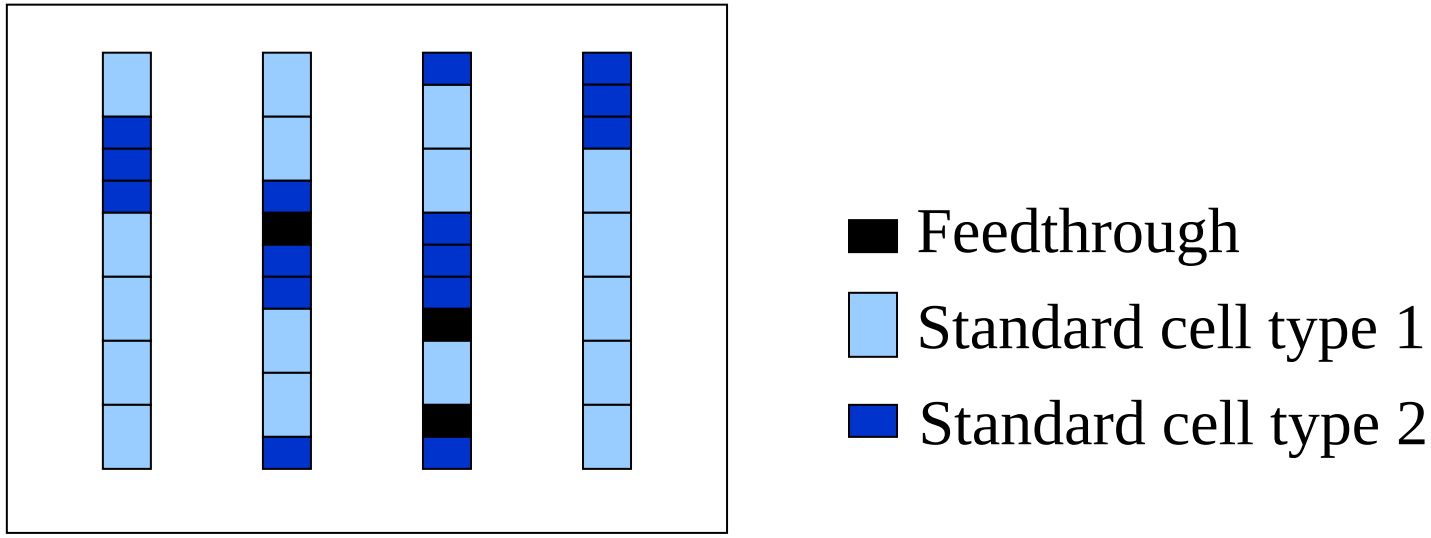
Floorplanning

- Blocks are placed in order to minimize area and the connections between them.



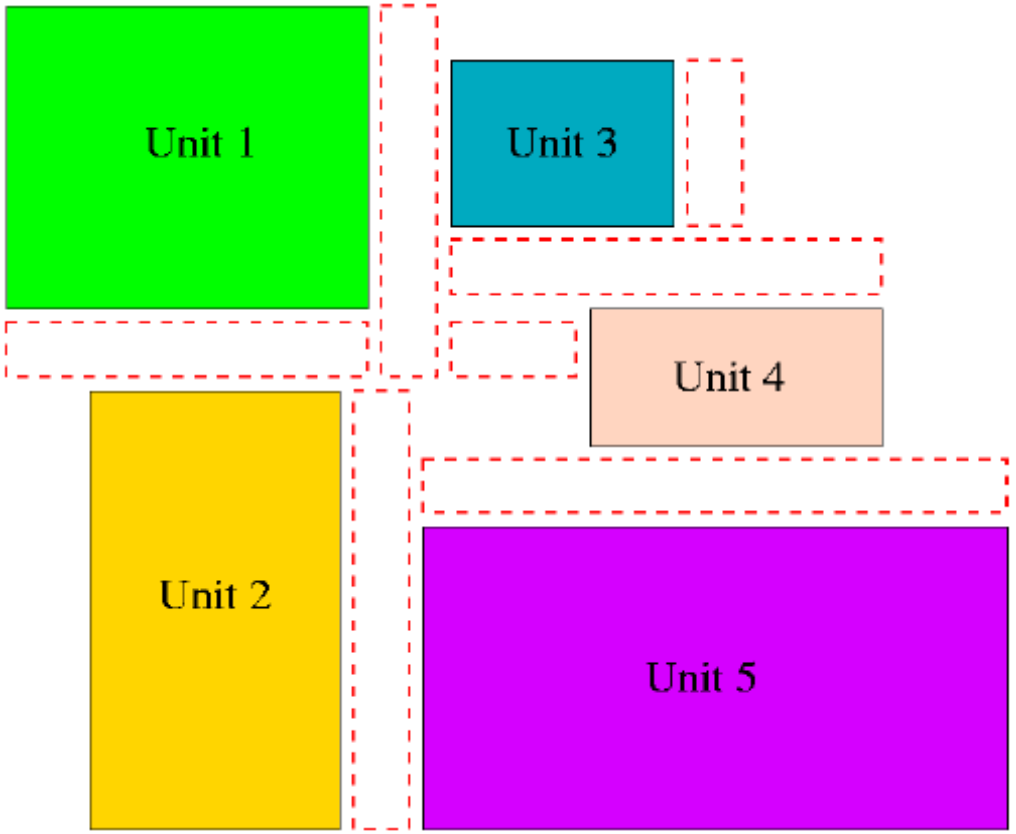
Physical Design Cycle

Placement – Exact placement of the modules (modules can be gates, standard cells, …). Details of the design are known and the goal is to minimize the total area and interconnect cost.



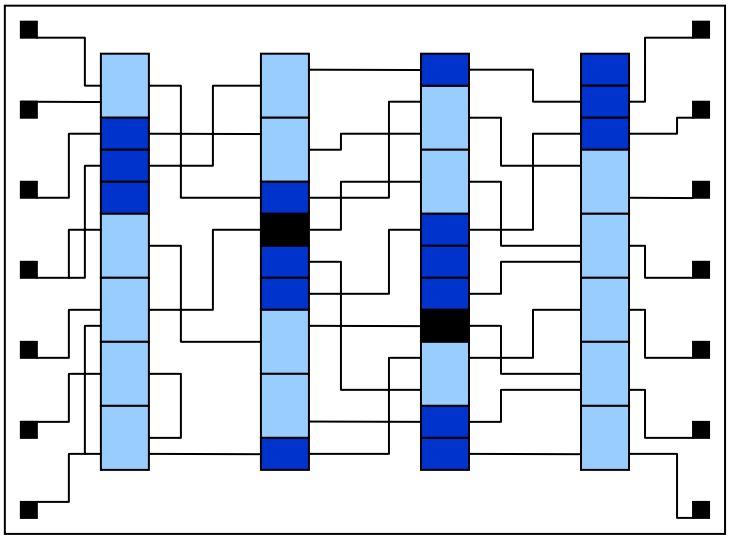
Channel definitions

- Blocks are placed so empty rectangular spaces are left between them. These spaces will be later used to make the interconnection.



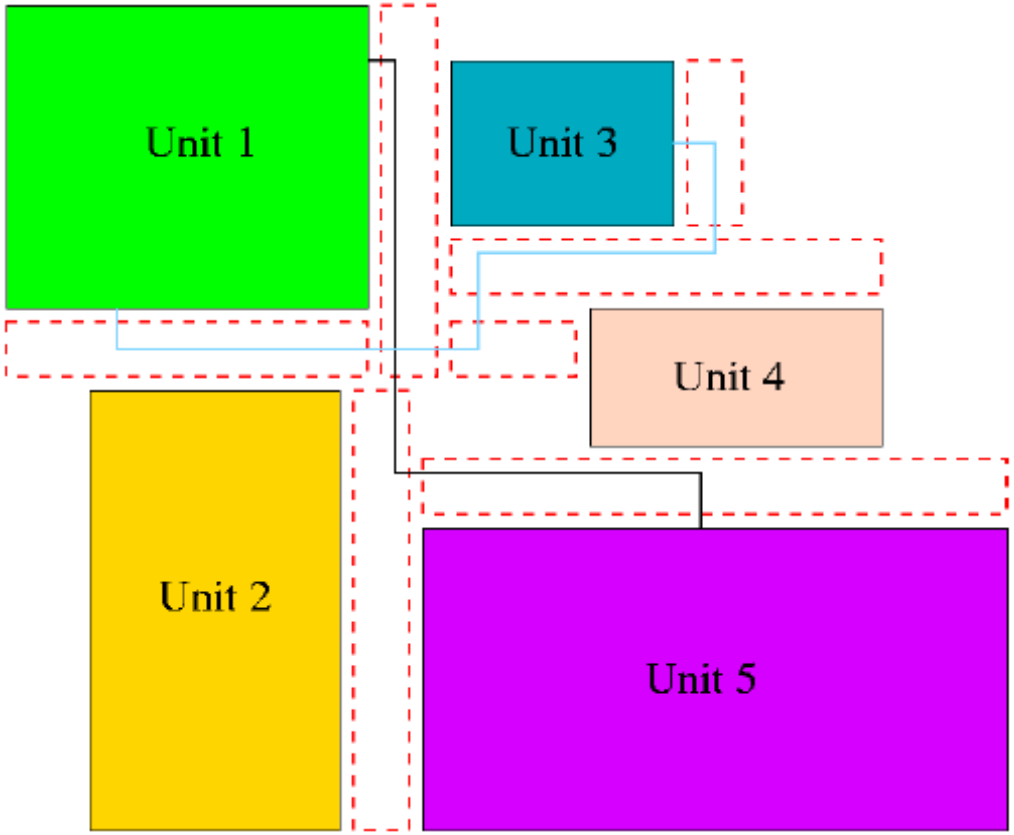
Physical Design Cycle

Routing – Complete the interconnections between the modules. Factors like critical path, wire spacing, … are considered. Include *global routing* and *detailed routing*.



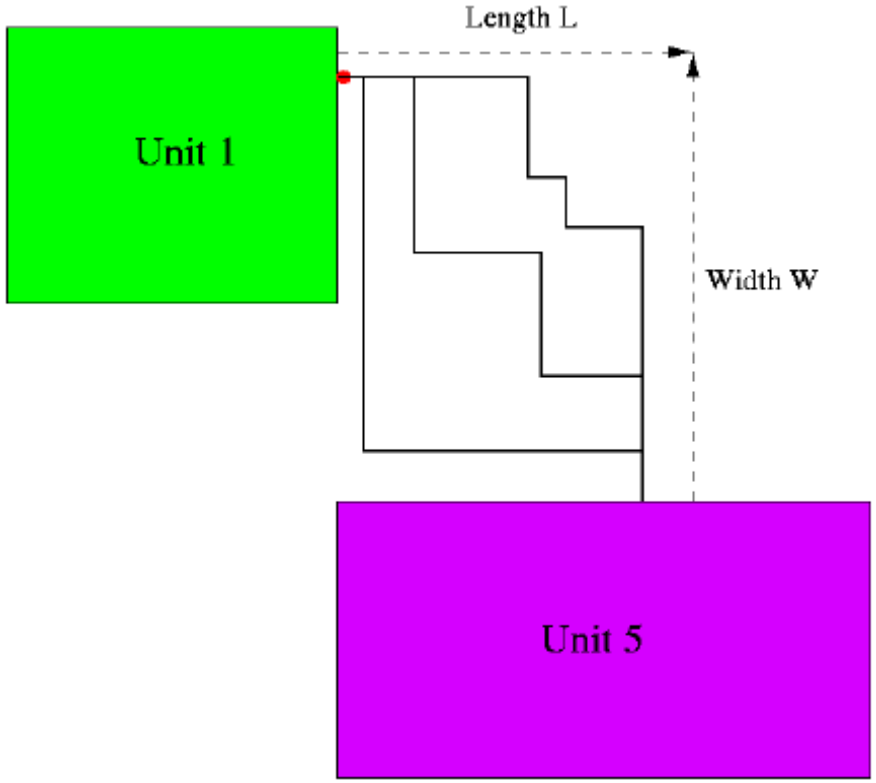
Global Routing

- Each connection will go from each origin block, through the channels until the end block.



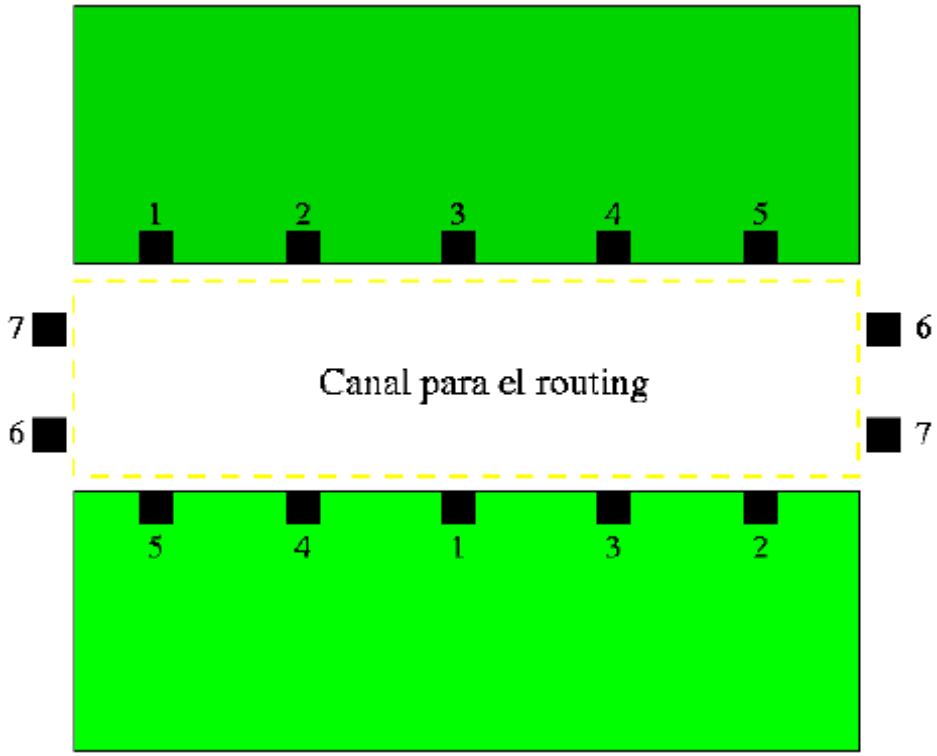
Global Routing

- The length of the connections will depend on the situation of the blocks rather than the way the routing is done.



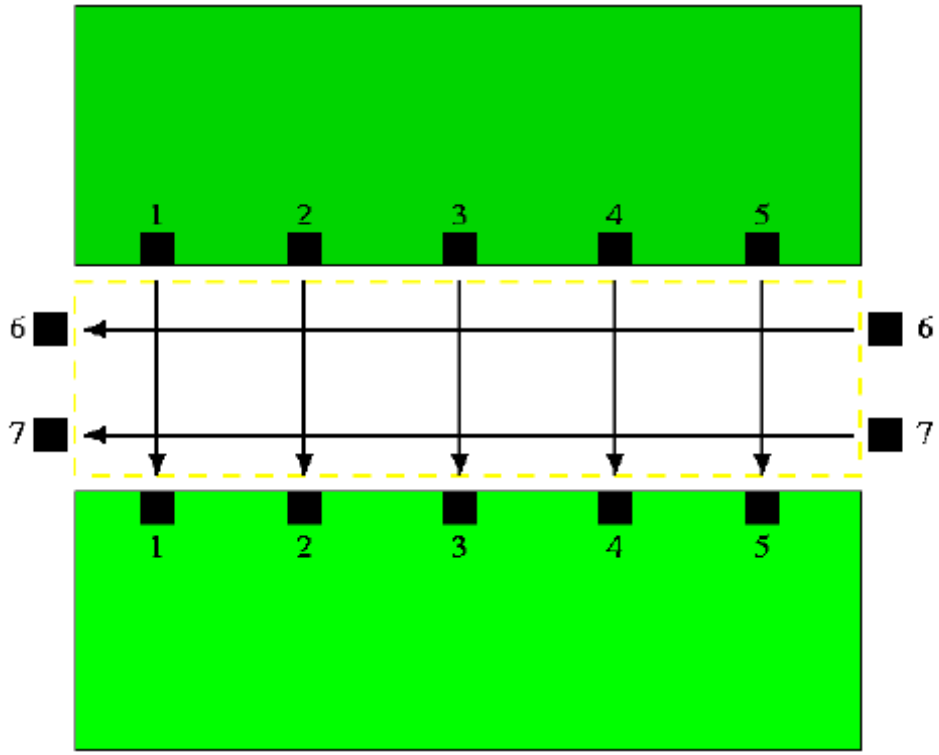
Detailed Routing

- The space required for each channel will depend on the complexity and density of the connections.



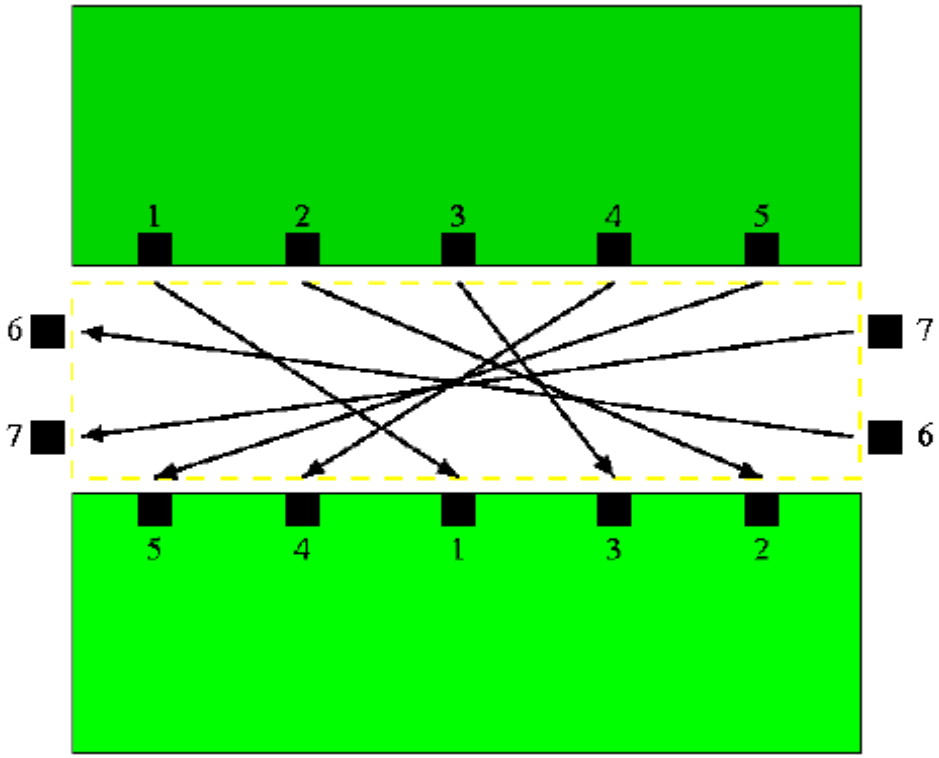
Detailed Routing

- Aligned connections require a smaller channel (similar to that of a bus).



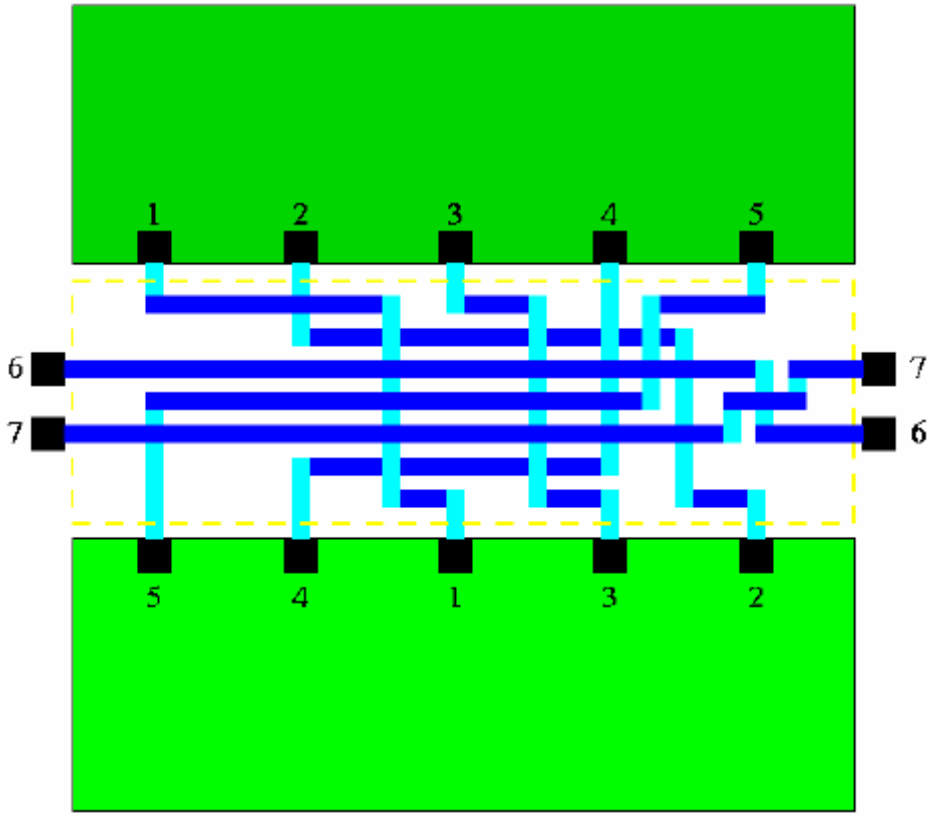
Detailed Routing

- Non-aligned connections increment the complexity of the routing and it increases the amount of space required.



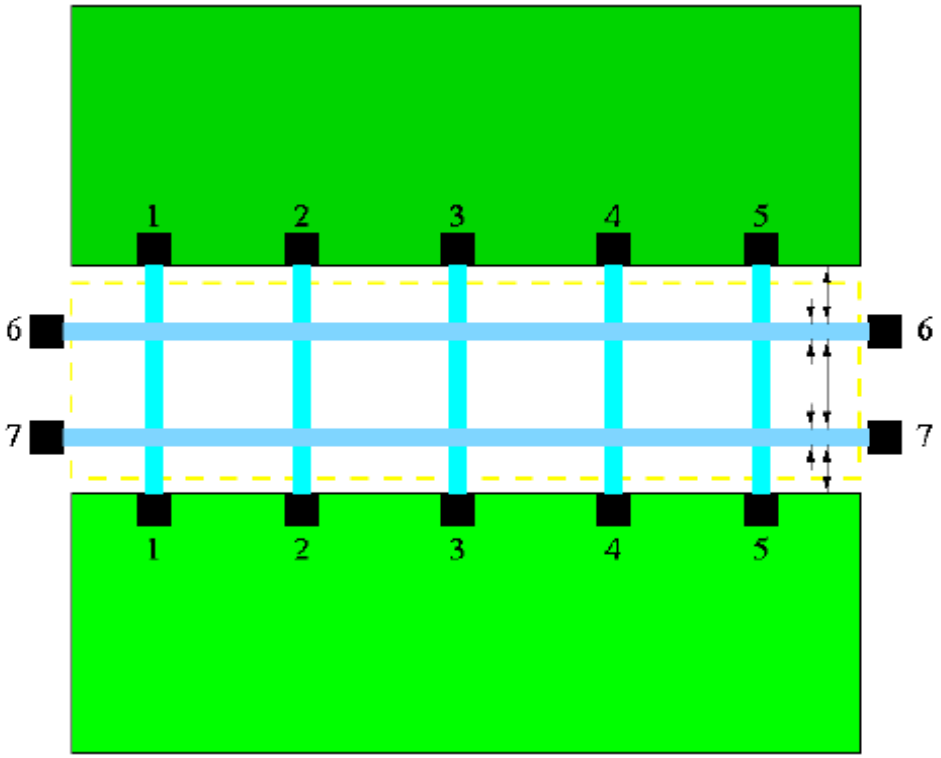
Detailed Routing

- The number of crossings determines the space required. The size of the channel may have to be increased.



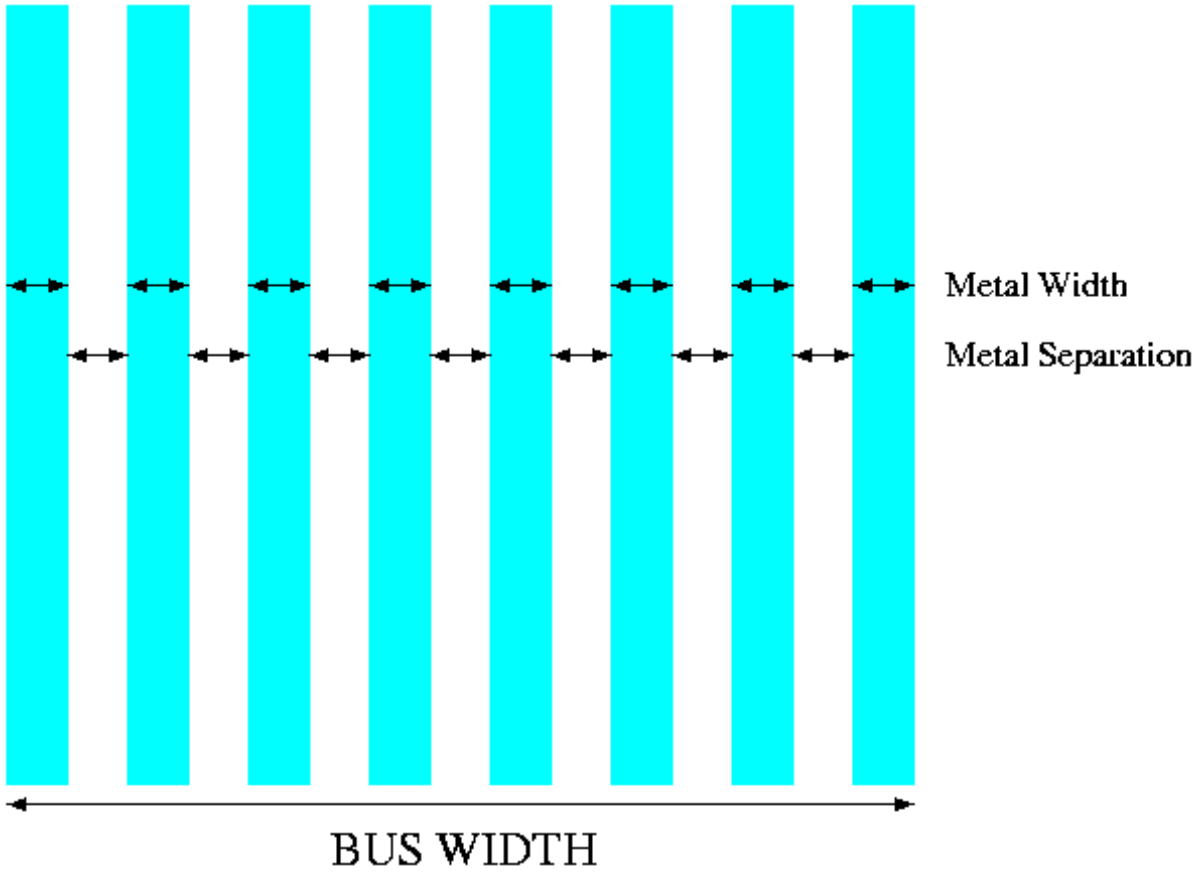
Detailed Routing

- Aligned connections reduce dramaticaly the area required of the channel for completing the routing.



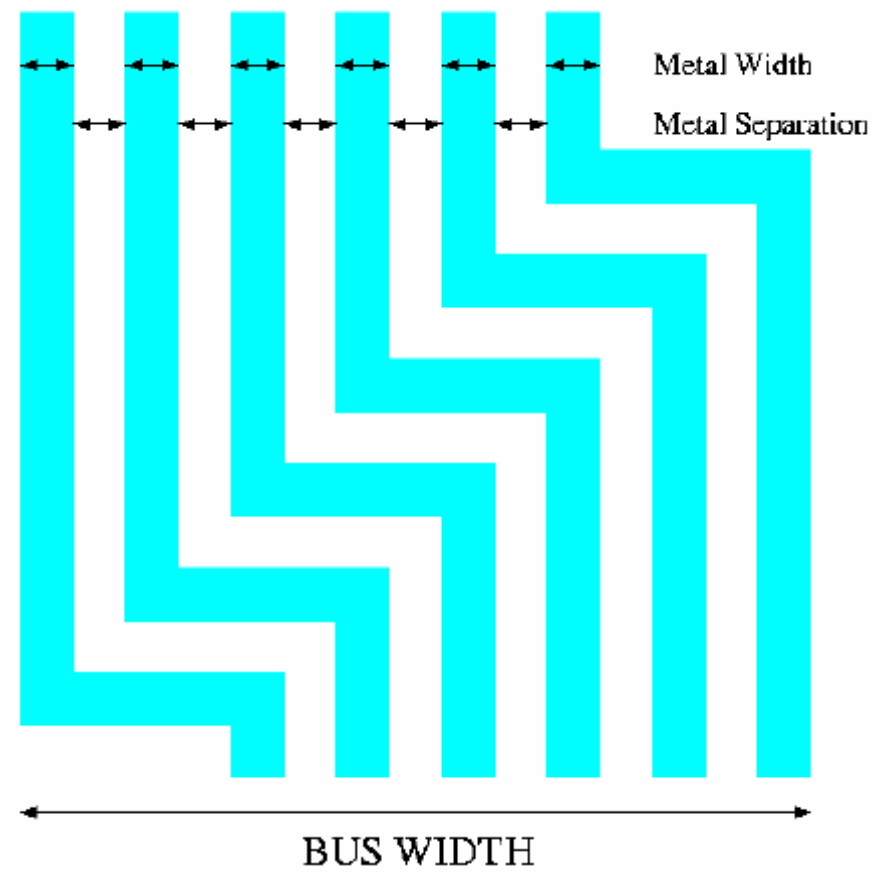
Bus area

- The area of each bus is proportional to the number of bits of the bus.



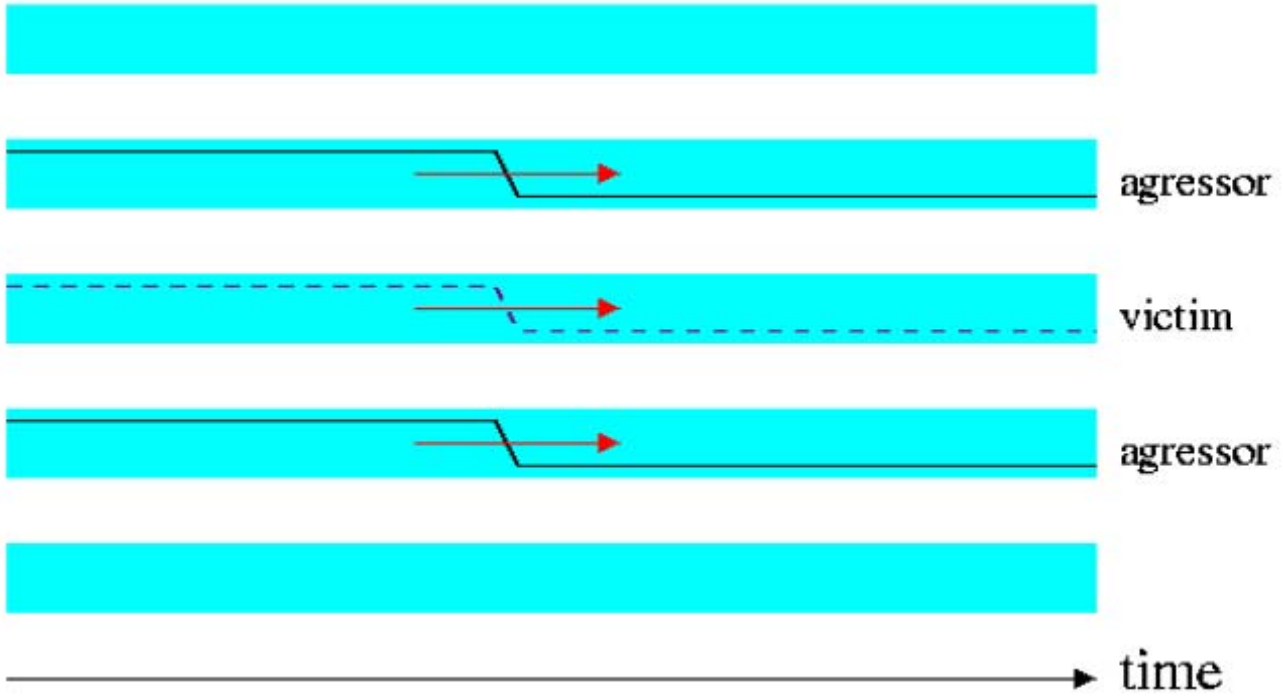
Bus area

- Los giros deben realizarse al estilo “Manhattan”, lo cual aumenta el área requerida.



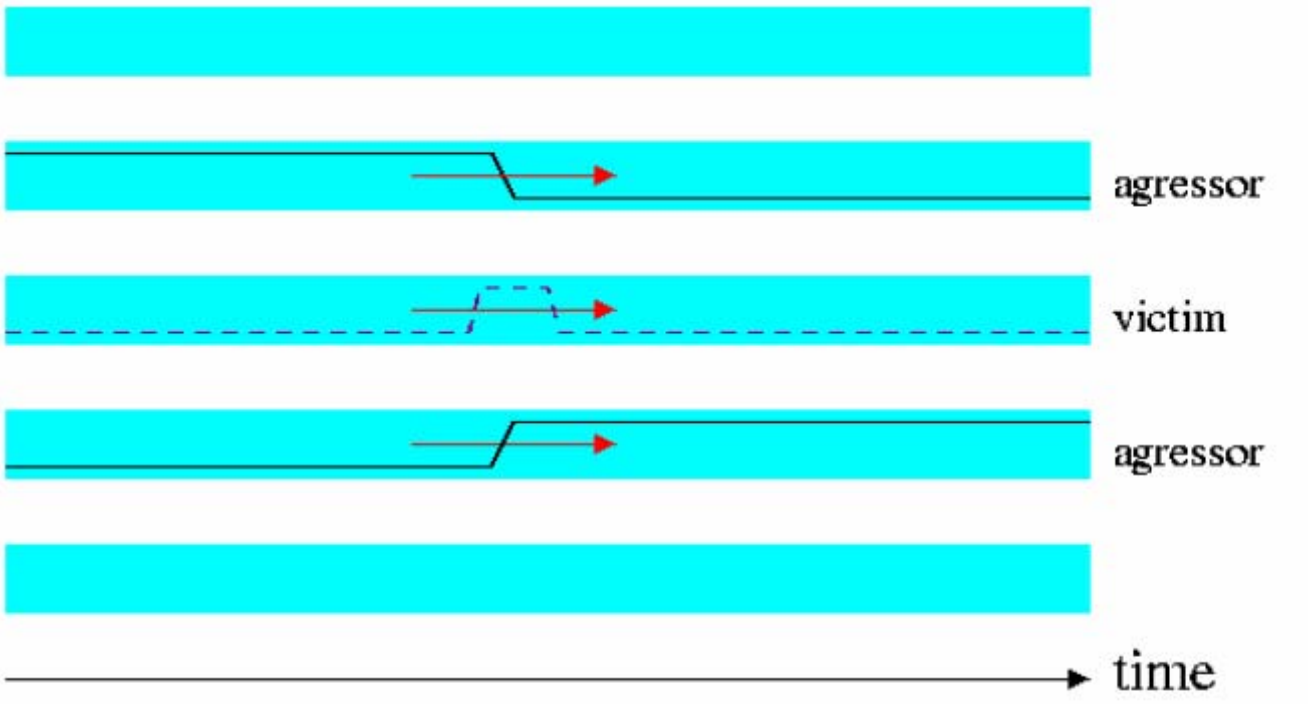
Bus cross-coupling

- The bits in a bus can interfere negatively between them: cross-coupling.



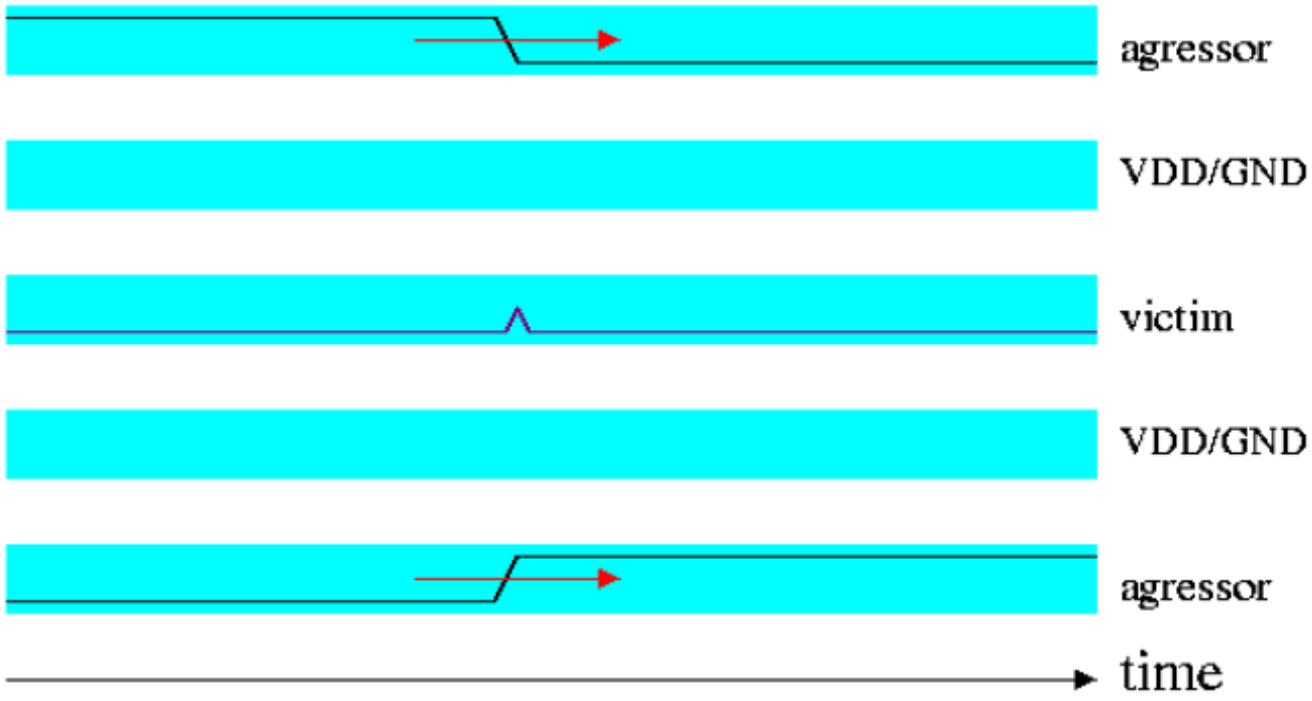
Bus cross-coupling

- Los distintos bits en un bus pueden interaccionar entre ellos deforma negativa: cross-coupling.



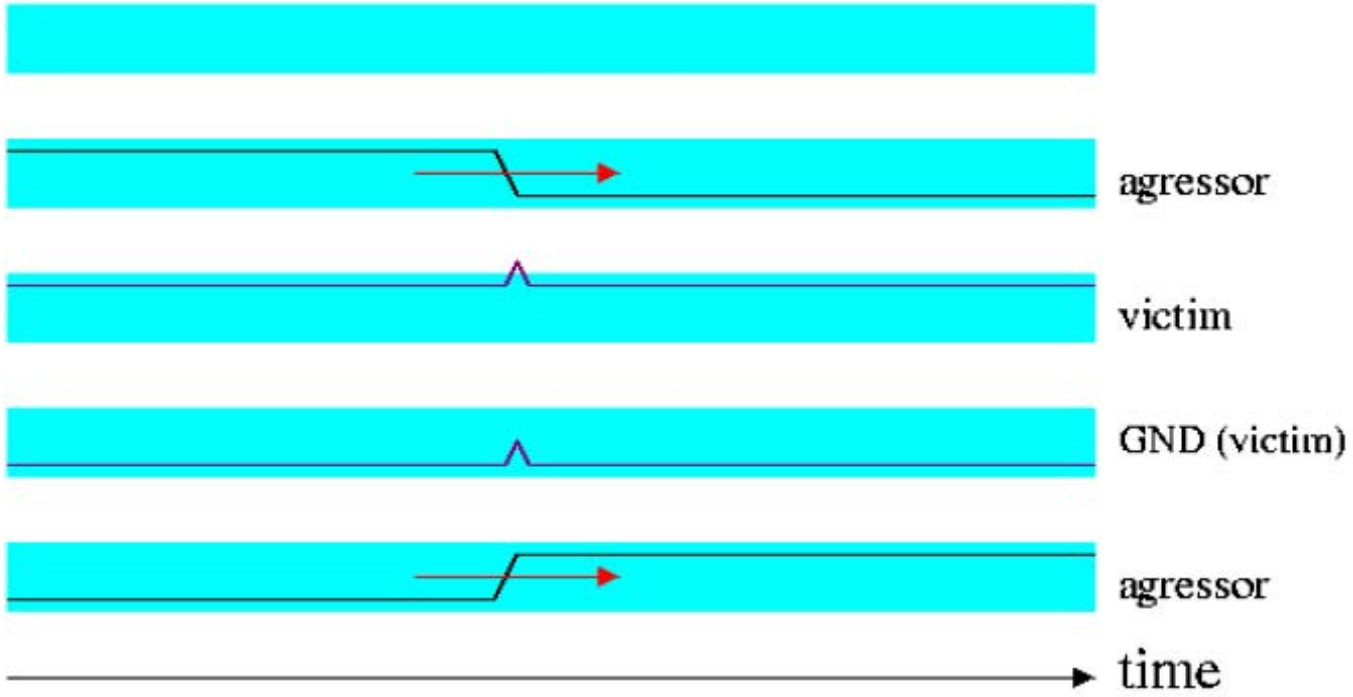
Bus cross-coupling

- If this is the case, the victim must be “protected” using any isolating technique (e.g. increasing the separation).



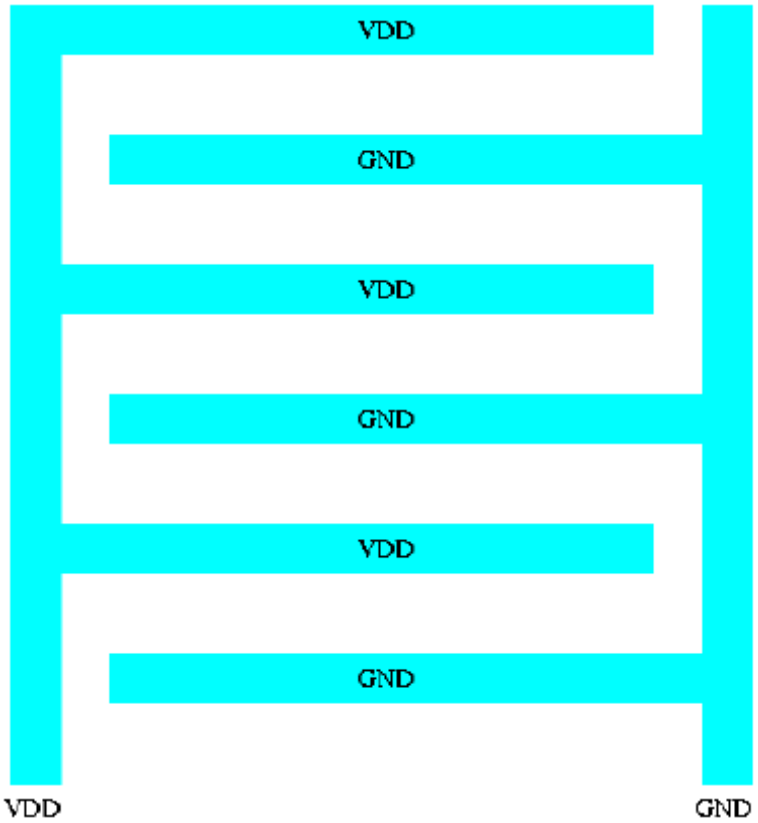
Bus cross-coupling

- Another solution is to use differential information between one signal and the neighbouring GND signal: the glitch appears in both.
- We can measure the difference among both victims.



Power distribution

- Interlazed distribution minimizes the number of metal levels required. The thickness of each channel will vary according to the power consumption.



Physical Design Cycle

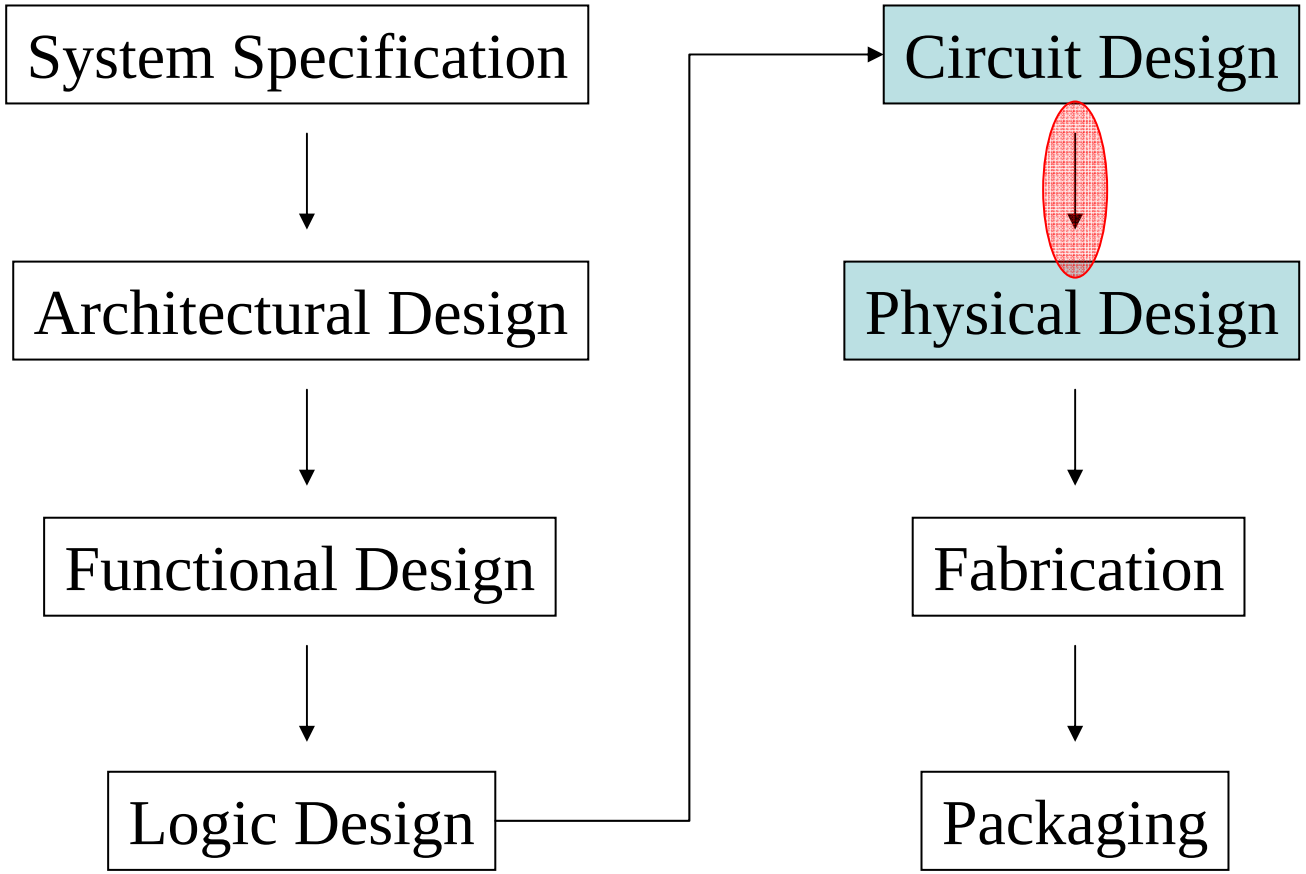
Compaction – Compress the layout from all directions to minimize the chip area.

Verification – Check correctness of the layout. Include DRC (*Design Rule Checking*), *circuit extraction* (generate a circuit from the layout to compare with the original netlist), *performance verification* (extract geometric information to compute resistance, capacitance, delay, ...)

Conclusions

- The area of the blocks is predictable:
  - Depends of the number of transistors and the spacing.
  - It does not depend on the connections.
- The area of the connections is not much predictable.
- If the blocks are larger than the routing: the connections are determined by the separation among the blocks.
- If the routing dominates the blocks (in the same way as buses): The buses behave now like blocks.
- In any other case it is needed to simulate the connections to estimate their area (very sensible to the floorplanning).

VLSI Design Cycle

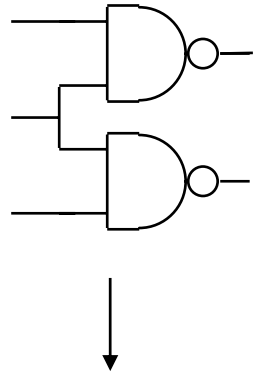


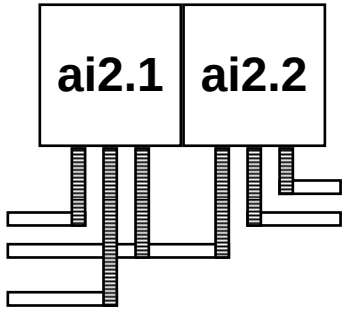
Circuit Design

- Outline
  - What is Circuit Design
  - Design Methods
  - Design Styles
  - Analysis and Verification
- Goal
  - Understand circuit design topics

What is circuit Design?

- Mapping logic to physical implementation
  - implementation
    - components
    - component locations
    - component wiring
    - geometrical shapes
  - examples
    - TTL chips on a PC board
    - single FPGA
    - custom CMOS chip
- Issues
  - level of circuit abstraction
  - target implementation technology





Implementation Methods

- Implementation methods
  - integrated circuits
    - programmable arrays - e.g. ROM, FPGA
    - full custom fabrication
  - hybrid integrated circuits
    - thin film - built-in resistors
    - thick film - ceramics
    - silicon-on-silicon - multi-chip modules
  - circuit boards
    - discrete wiring - wire-wrap
    - printed circuits
- Design rules
  - topology and geometry constraints
  - imposed by physics and manufacturing
  - example: wires must be > 2 microns wide

Design Methods

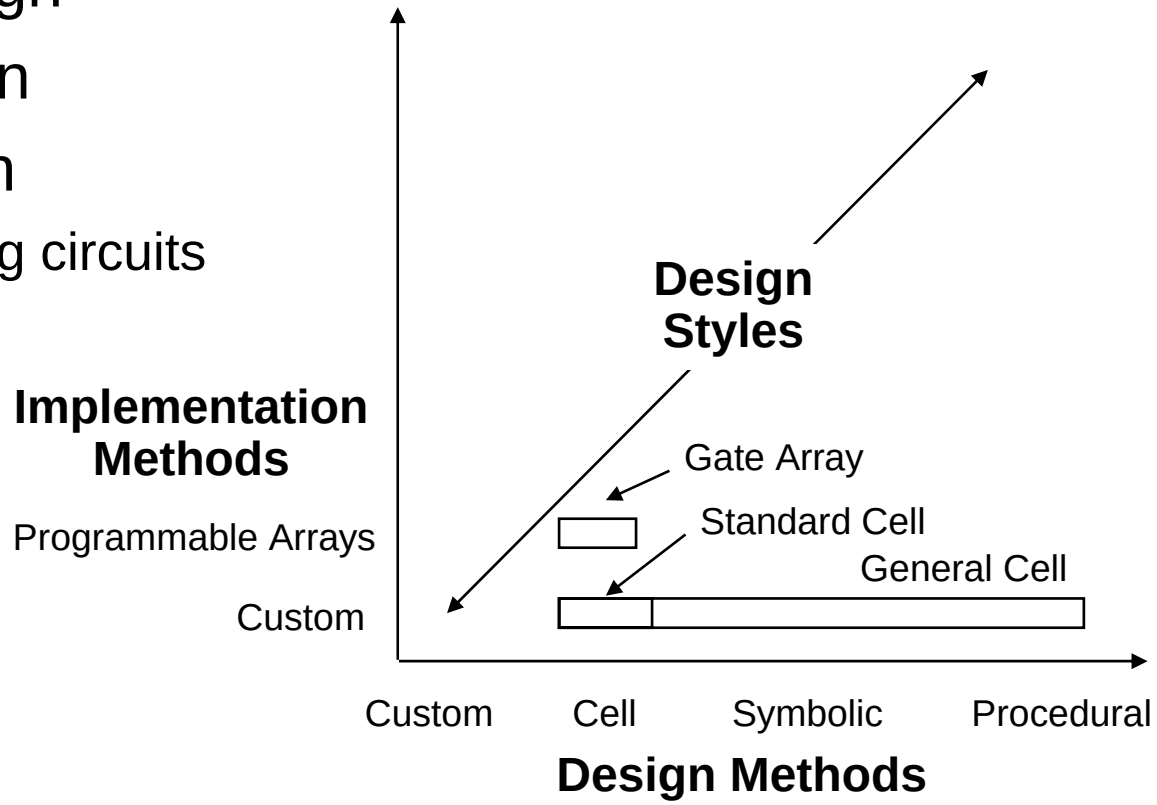
- Full custom design
  - no constraints - output is geometry
    - highest-volume, highest performance designs
  - requires some handcrafted design
    - 5-10 transistors/day for custom layout
  - use to design cells for other methods
  - primary CAD tools
    - layout editor, plotter
- Cell-based design
  - compose design using a library of cells
  - at board-level, cells are chips
  - cell = single gate up to microprocessor
  - primary CAD tools
    - partitioning
    - placement and routing

Design Methods

- Symbolic design
  - reduce problem to topology
  - let tools determine geometry (following design rules)
  - can reuse same topology when design rules change
    - e.g. shrink wires from 2 microns to 1 micron
  - used mostly to design cells
- Procedural design
  - “cells” are programs
  - module generation - ROMs, RAMs, PLAs
  - silicon compilation - module assembly from HLL
- Analysis and verification
  - design rule checking - geometry widths, spacings ok?
  - circuit extraction - geometry => circuit
  - interconnect verification - circuit A == circuit B?

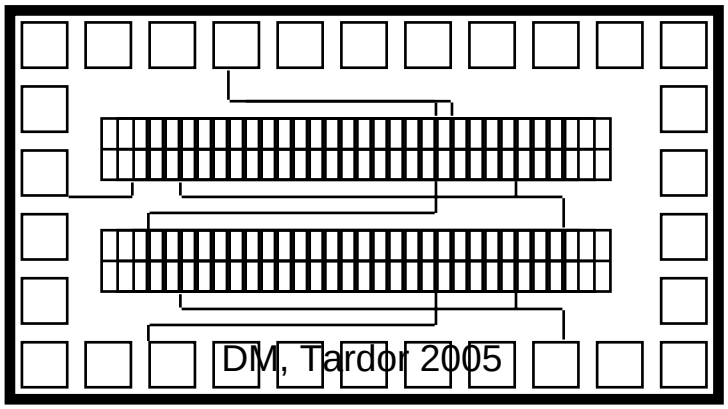
Design Styles

- Gate array design
  - FPGAs are a form of gate array
- Standard cell design
- General cell design
- Full custom design
  - still used for analog circuits



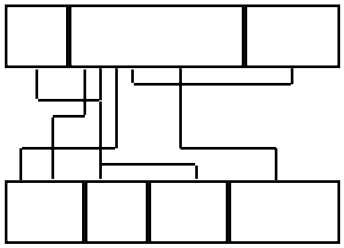
Gate Array Design

- Array of prefabricated gates/transistors
- Map cell-based design onto gates
- Wire up gates
  - in routing channels between gates
  - over top of gates (sea of gates)
  - predefined wiring patterns to convert transistors to gates
- CAD problems
  - placement of gates/transistors onto fixed sites
  - global and local wire routing in fixed space



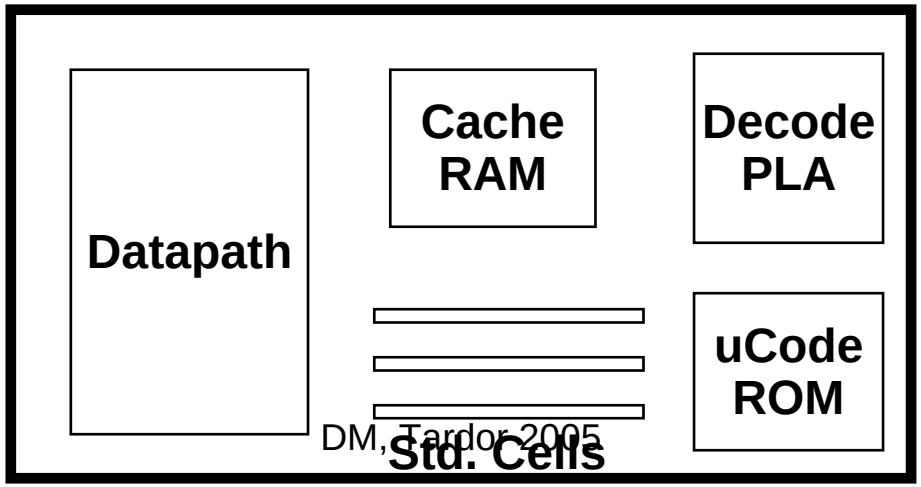
Standard Cell Design

- Design circuit using standard cells
  - cells are small numbers of gates, latches, etc.
- Technology mapping selects cells
- Place and wire them
  - cells placed in rows
    - all cells same height, different widths
  - wiring between rows - channels
- CAD problems
  - cell placement - row and location within row
  - wiring in channels
  - minimize area, delay



General Cell Design

- Generalization of standard cells
- Cells can be large, irregularly shaped
  - standard cells, RAMs, ROMs, datapaths, etc.
- Used in large designs
  - e.g. Pentium has datapaths, RAM, ROM, standard cells, etc.
- CAD problems
  - placement and routing of arbitrary shapes is difficult



Case study

Standard Cell implementation

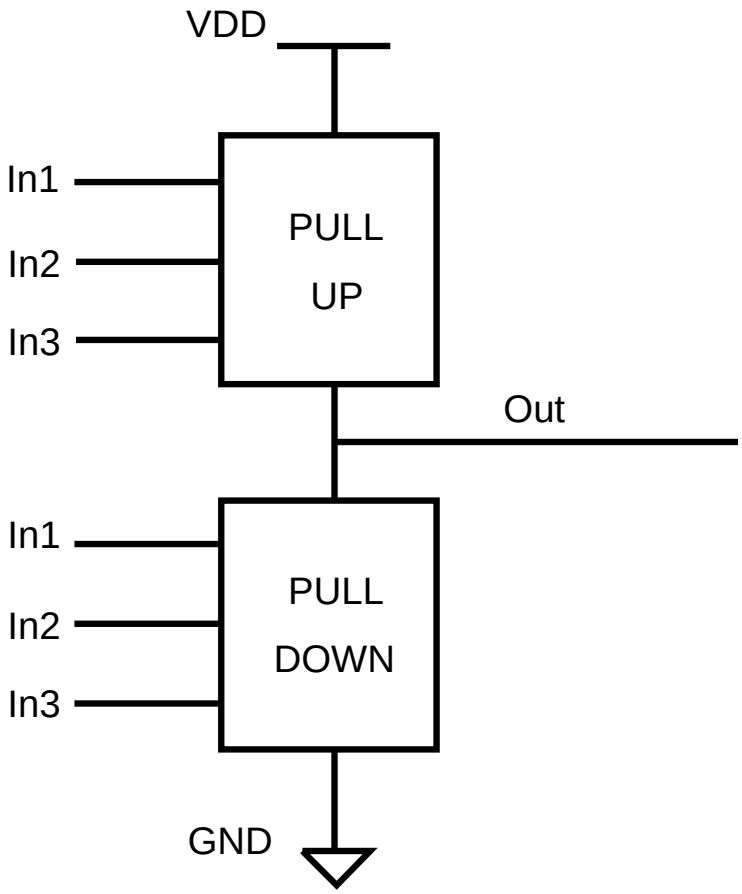
Standard Cell Implementation

- Cells designed in a certain pattern
- Due to the *standard* pattern they can be used in several circuits

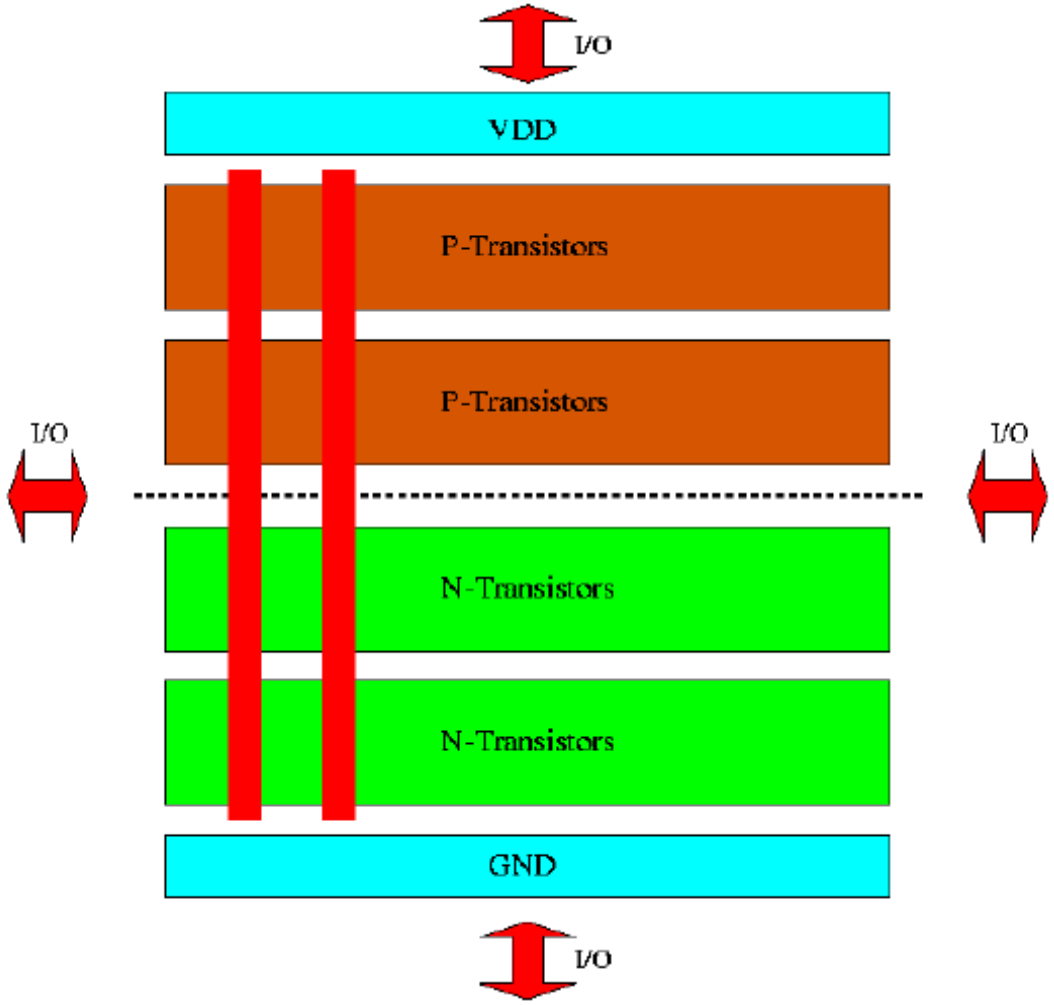
\+ Design reuse

\- Non-adaptative design

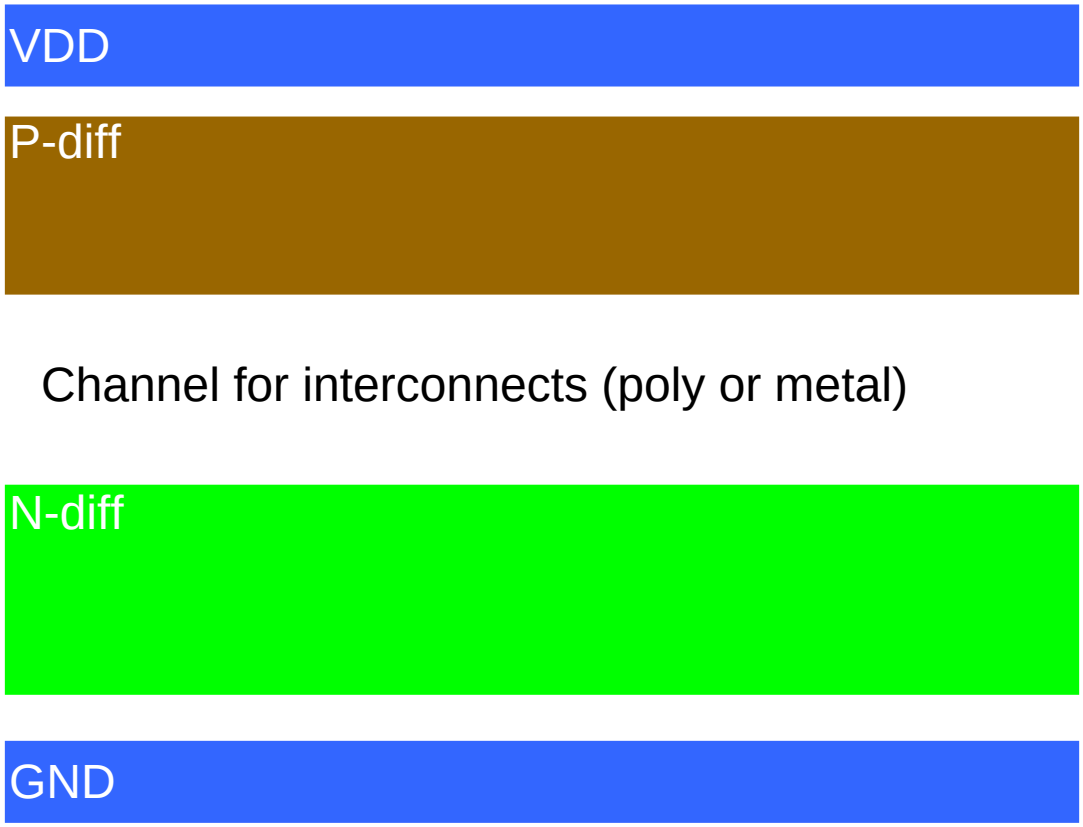
A typical MOS gate



Standard Cell Structure



Standard Cell Structure



Poly

Metal 1

Metal 2

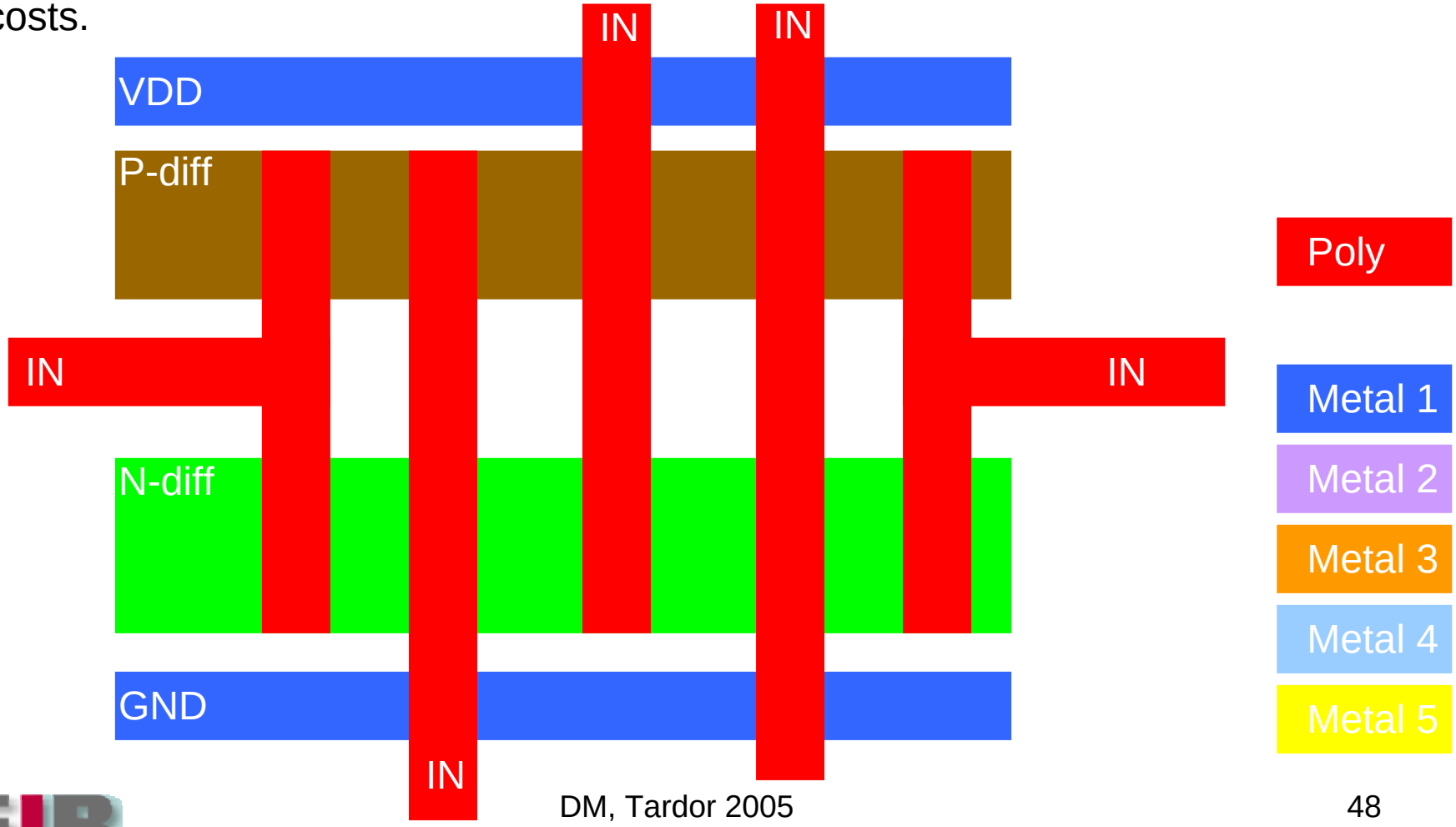
Metal 3

Metal 4

Metal 5

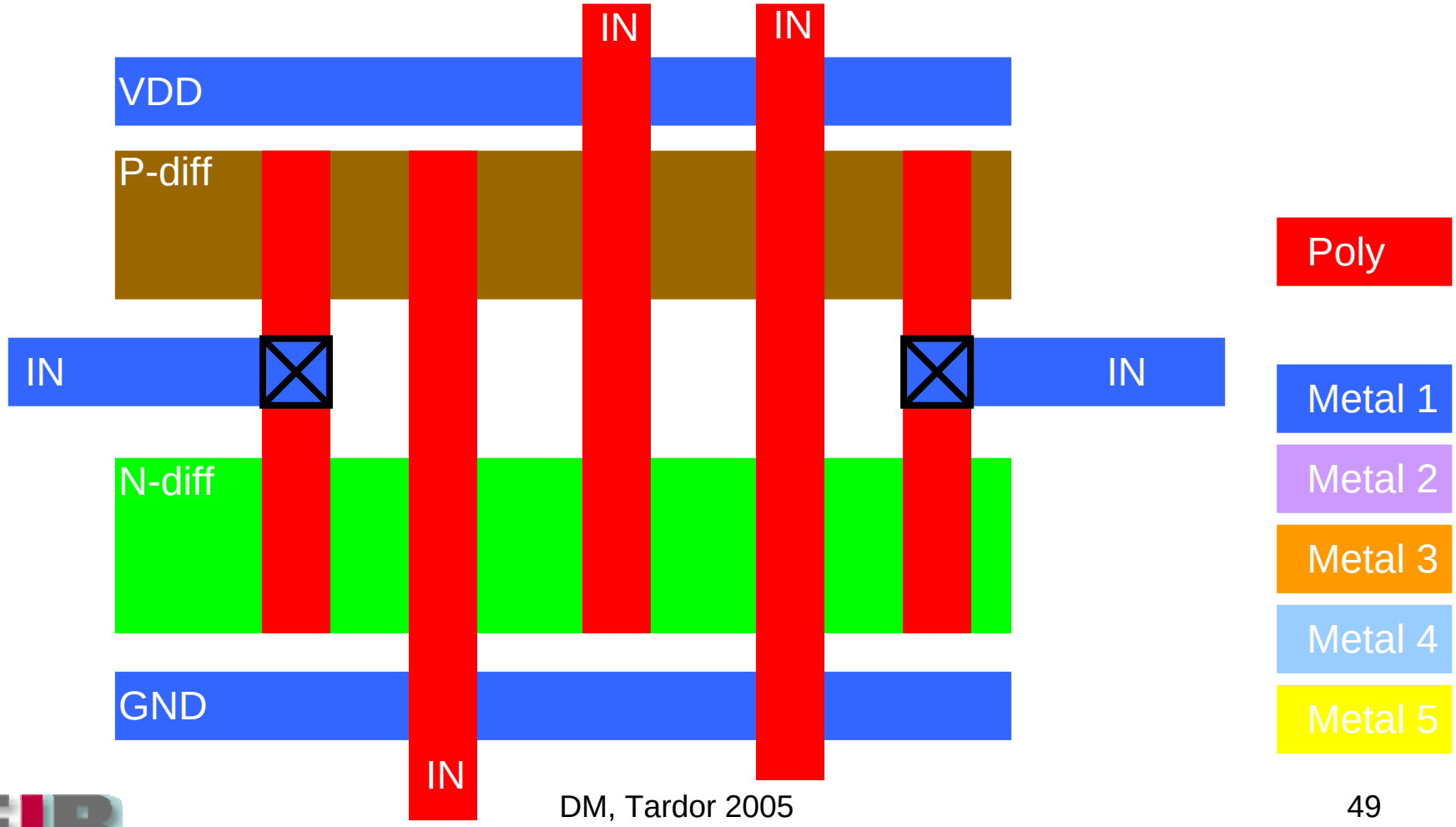
Standard Cell Structure

The inputs of the transistors are connected to the poly layer. Horizontal connections are not recommendable since they increase the manufacturing costs.



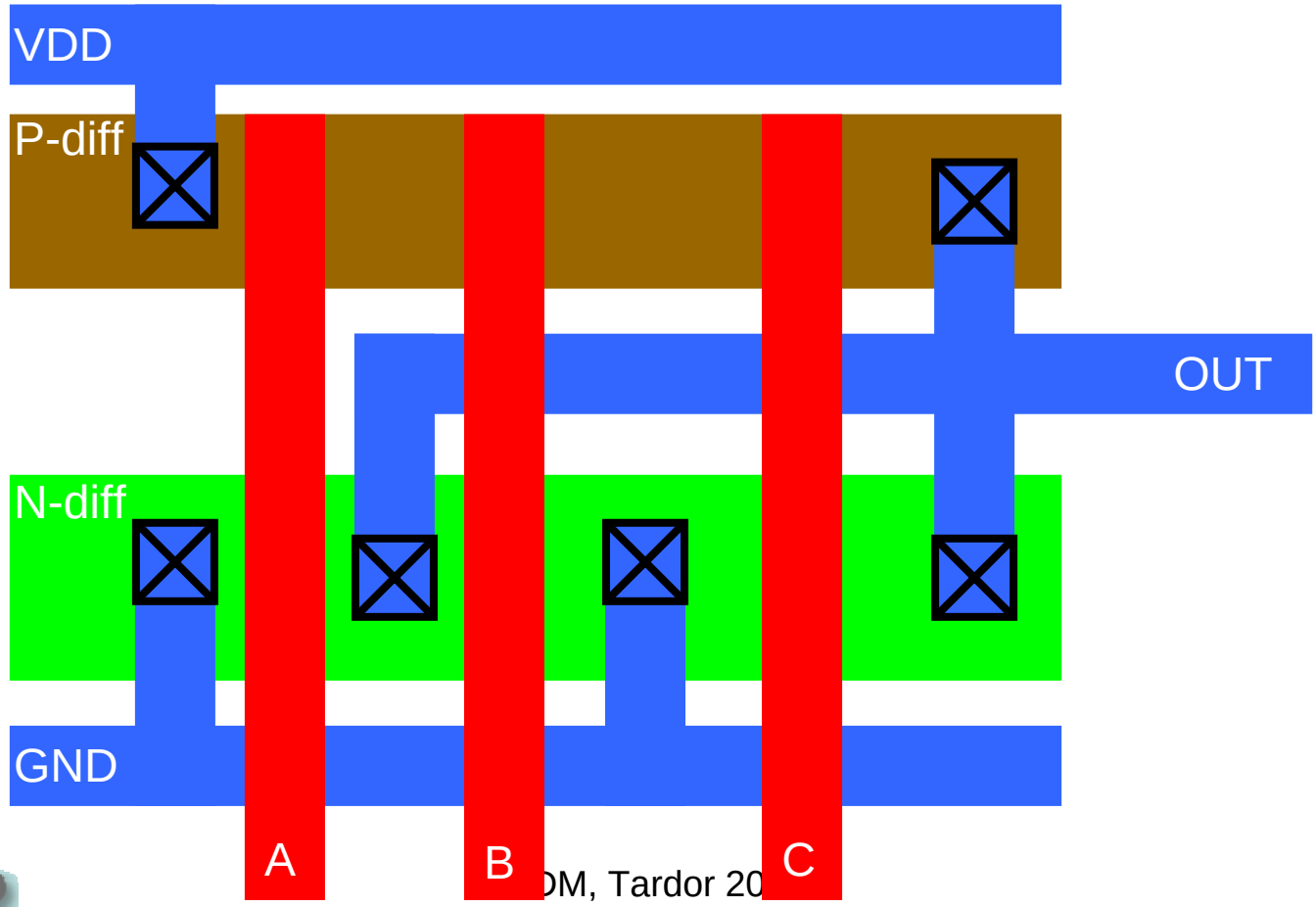
Standard Cell Structure

The inputs of the transistors are connected to poly or to metal –in case they are lateral inputs.



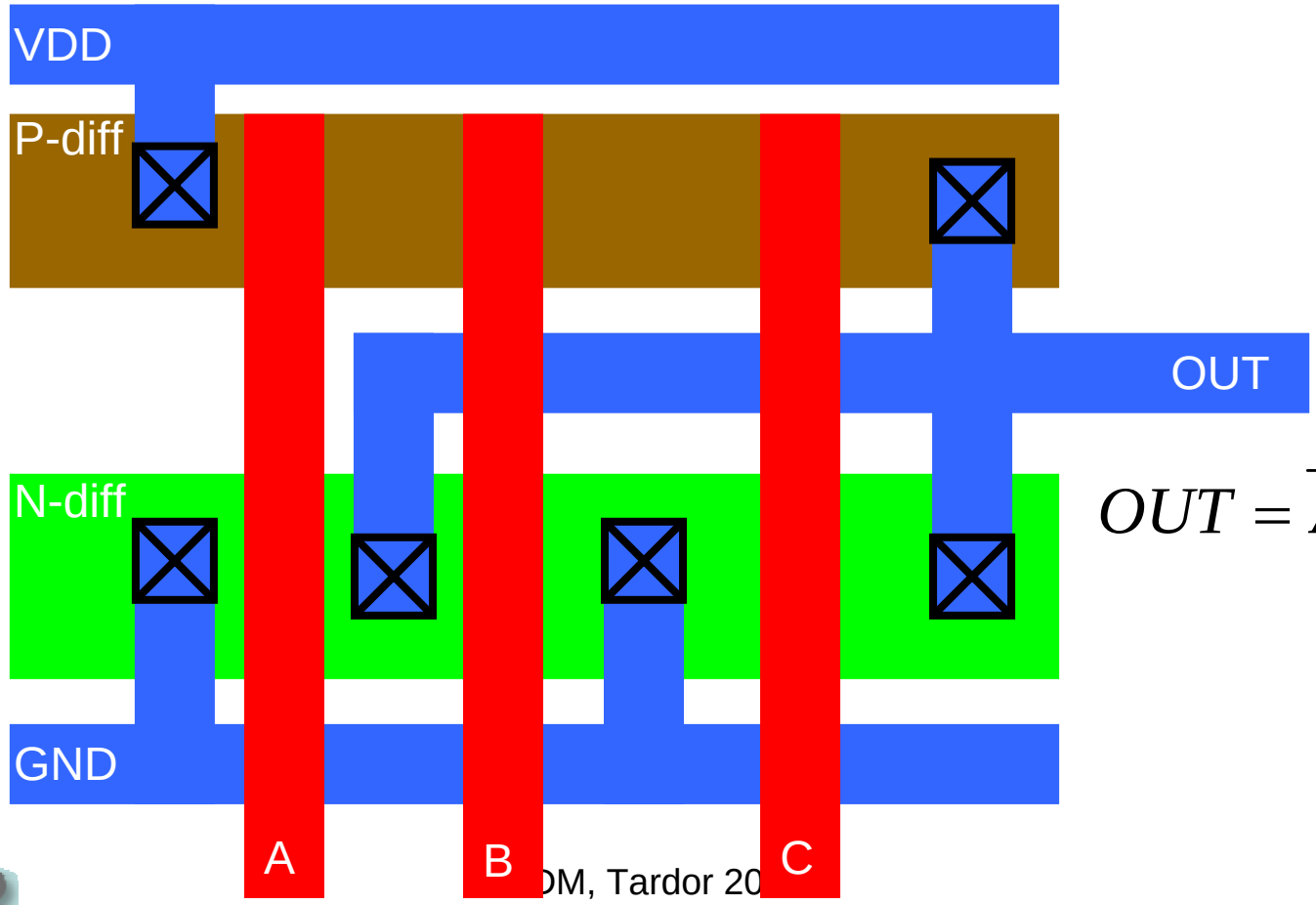
Standard Cell Structure

Transistors are placed in a serial way. If we want them in paral·lel we will have to add metal connections.



Standard Cell Structure

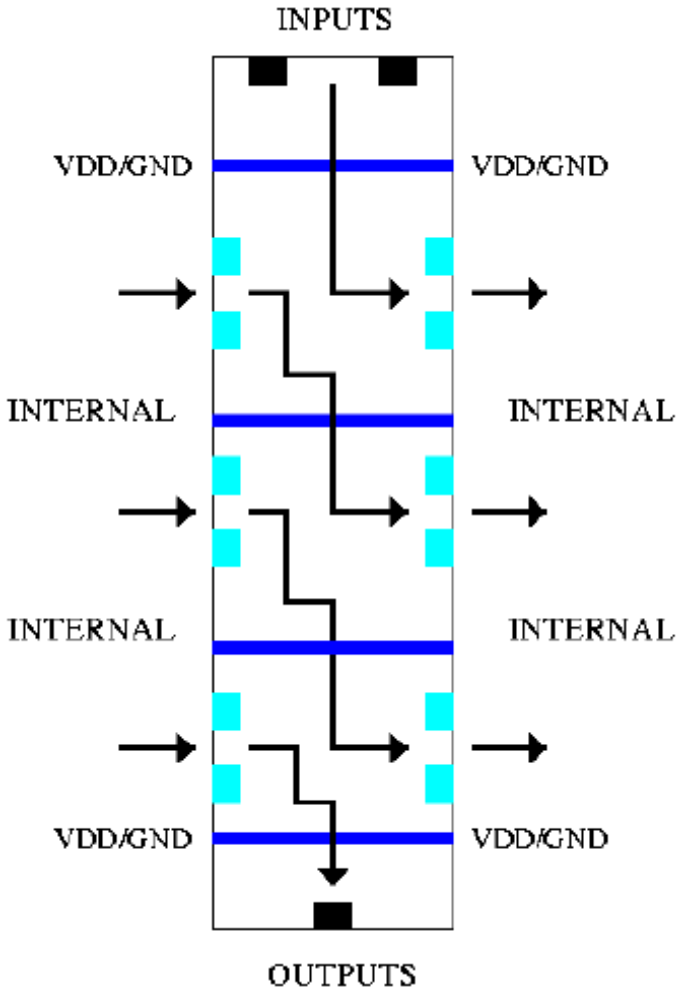
Transistors are placed in a serial way. If we want them in paral·lel we will have to add metal connections.



$$OUT = \overline{A + B + C}$$

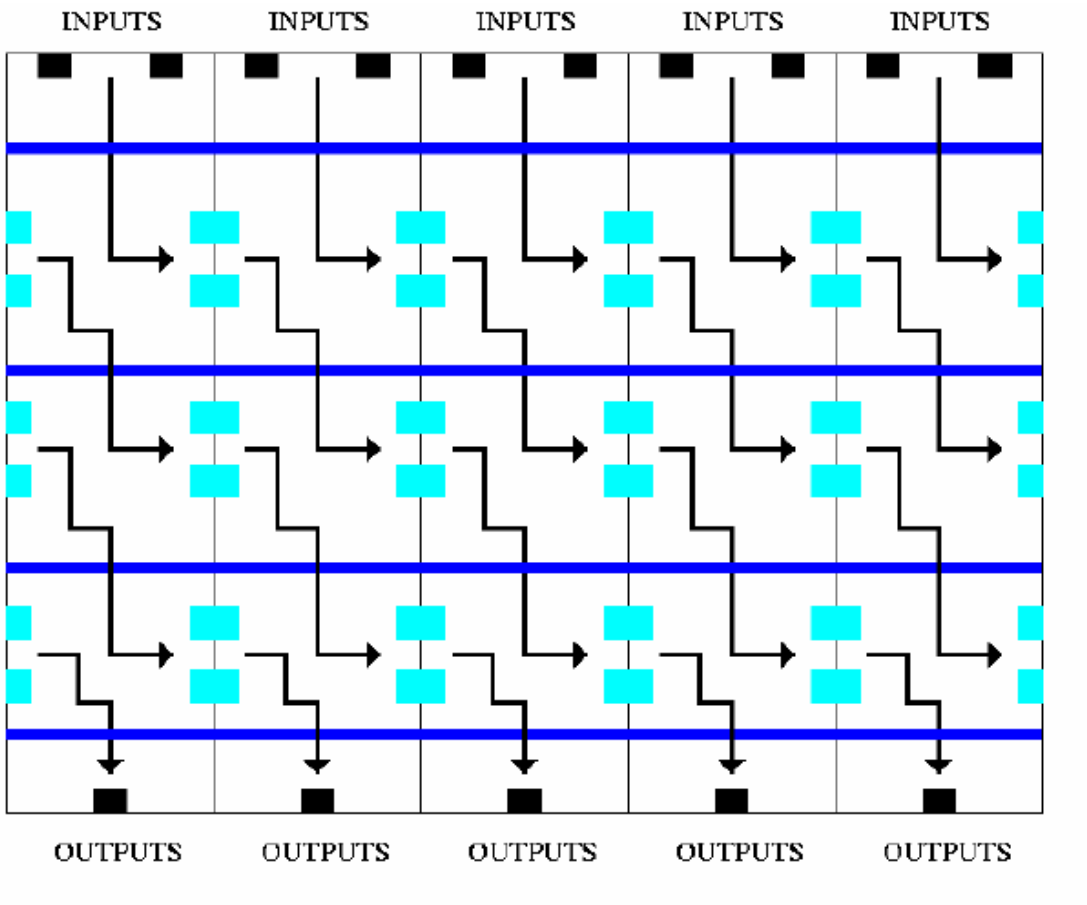
Bit slice design

- We can design a block that implements all the operations for one bit (e.g. in a multi-bit design, as in an adder)
- We have to take into account all input, ouput and internal connections.
- Putting each cell together generates a regular and dense structure.
- Usual way of designing a processor's datapath (registers + FU + shifters).



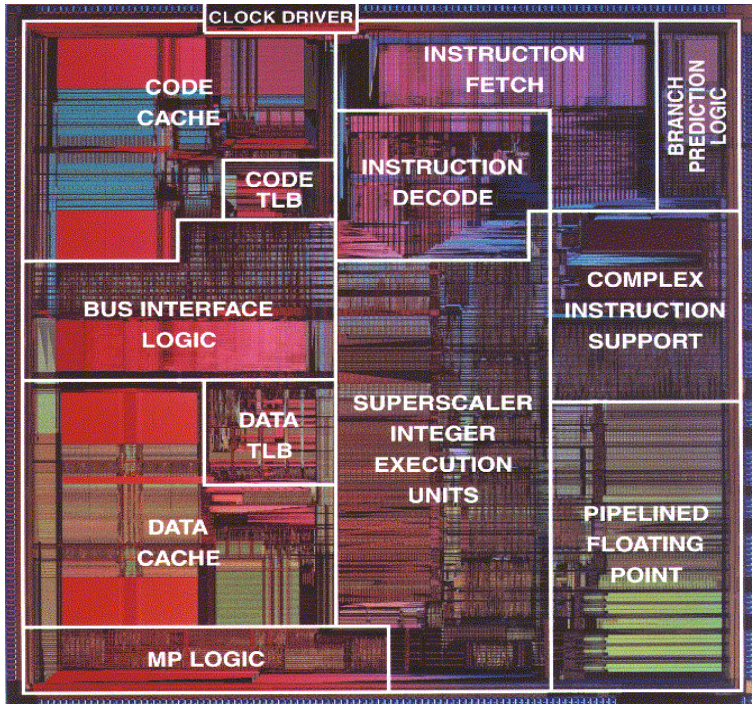
Bit slice design

- Block design from standard 1-bit cells

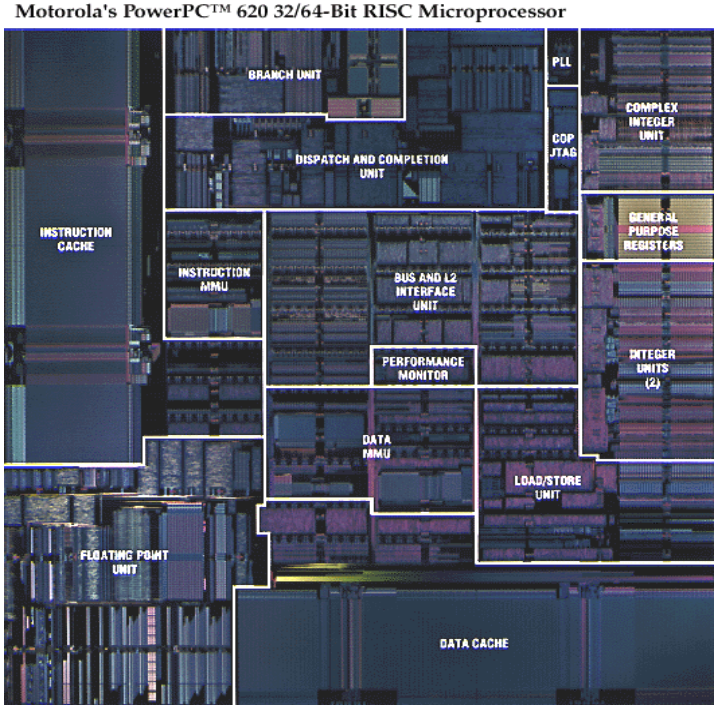


High performance devices

- Mixture of full custom, standard cells and macro's
- Full custom for special blocks: Adder (data path), etc.
- Macro's for standard blocks: RAM, ROM, etc.
- Standard cells for non critical digital blocks



Pentium



Power PC