



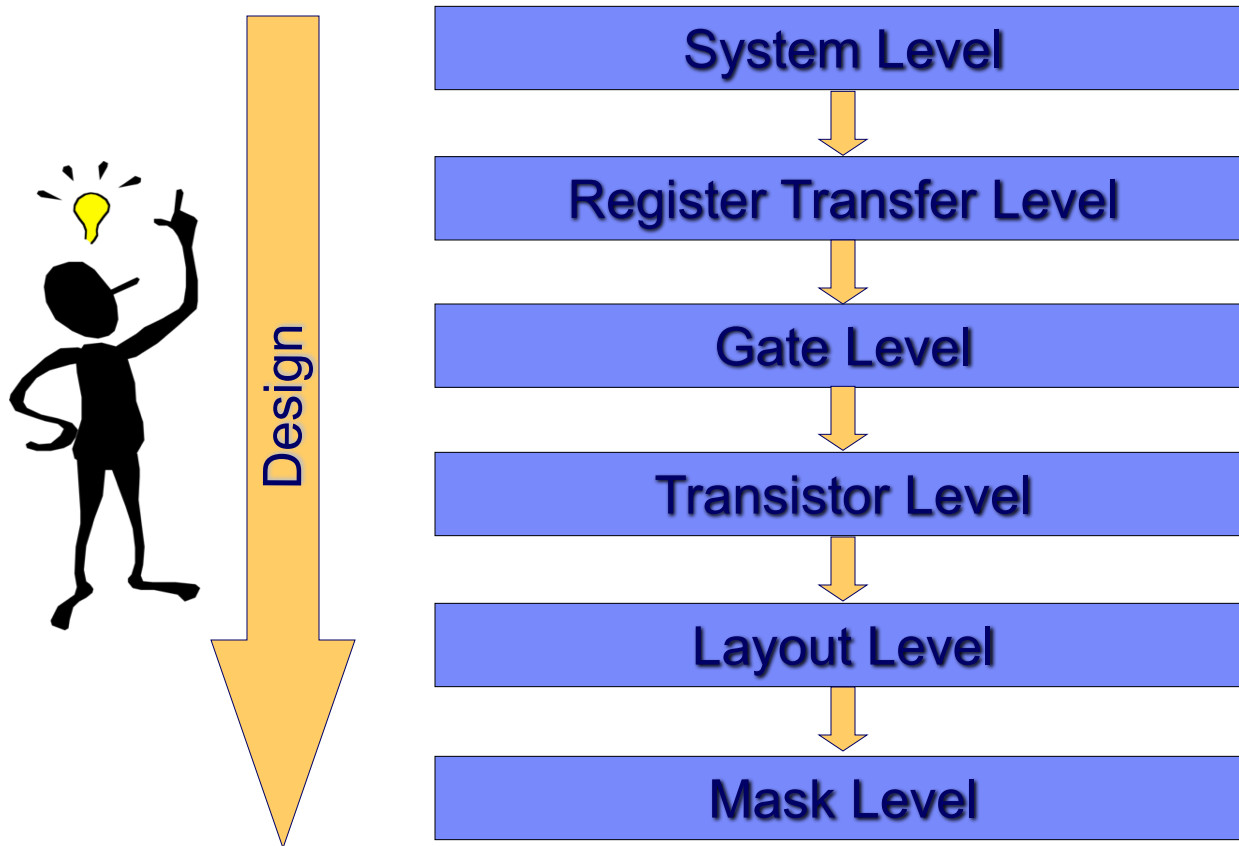
Design Flow



Agenda

- ▶ Design Phases
- ▶ General Design Approach
 - Top Level Flow
 - Block Level Flow
- ▶ Chip Planning
 - FloorPlanning
 - Pin Assignment
- ▶ Logic Synthesis
- ▶ Placement
- ▶ Routing
- ▶ Placement Driven Synthesis
- ▶ Timing Analysis
- ▶ Power Analysis
- ▶ Noise analysis

Design of Integrated Circuits



System Level

- ▶ Abstract algorithmic description of high-level behavior

- e.g. C-Programming language

```
Port*
compute_optimal_route_for_packet(Packet_t *packet,
                                Channel_t *channel)
{
    static Queue_t *packet_queue;

    packet_queue = add_packet(packet_queue, packet);
    ...
}
```

- abstract because it does not contain any implementation details for timing or data
- efficient to get a compact execution model as first design draft
- difficult to maintain throughout project because no link to implementation

RTL Level

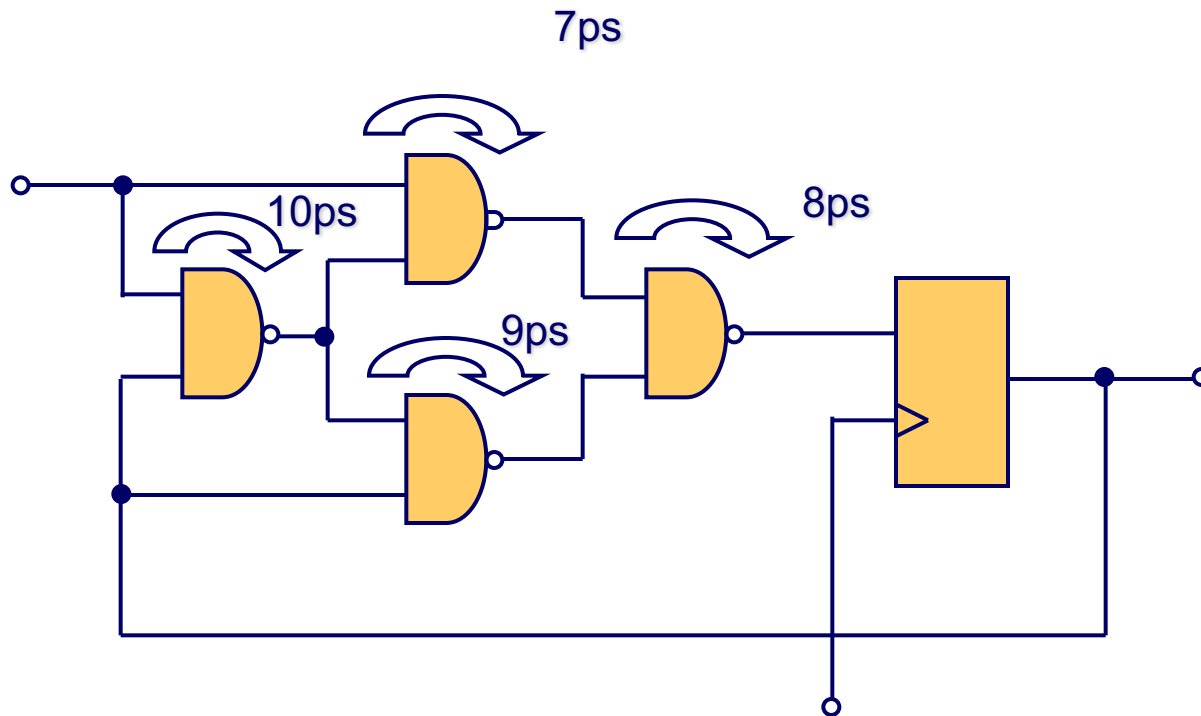
- ▶ Cycle accurate model “close” to the hardware implementation
 - bit-vector data types and operations as abstraction from bit-level implementation
 - sequential constructs (e.g. if - then - else) to support modeling of complex control flow

```
module mark1;
reg [31:0] m[0:8192];
reg [12:0] pc;
reg [31:0] acc;
reg[15:0] ir;
always
  begin
    ir = m[pc];
    if(ir[15:13] == 3b'000)
      pc = m[ir[12:0]];
    else if (ir[15:13] == 3'b010)
      acc = -m[ir[12:0]];
    ...
  end
endmodule
```

Gate Level

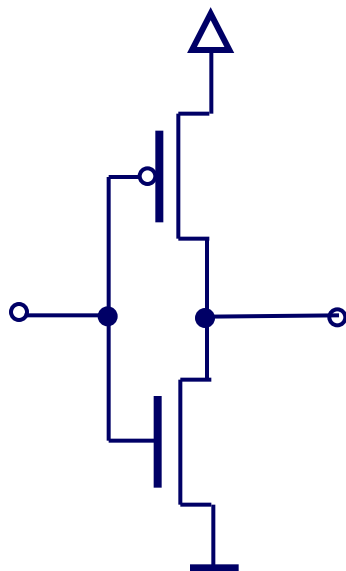
► Model on finite-state machine level

- models function in Boolean logic using registers and gates
- various delay models for gates and wires



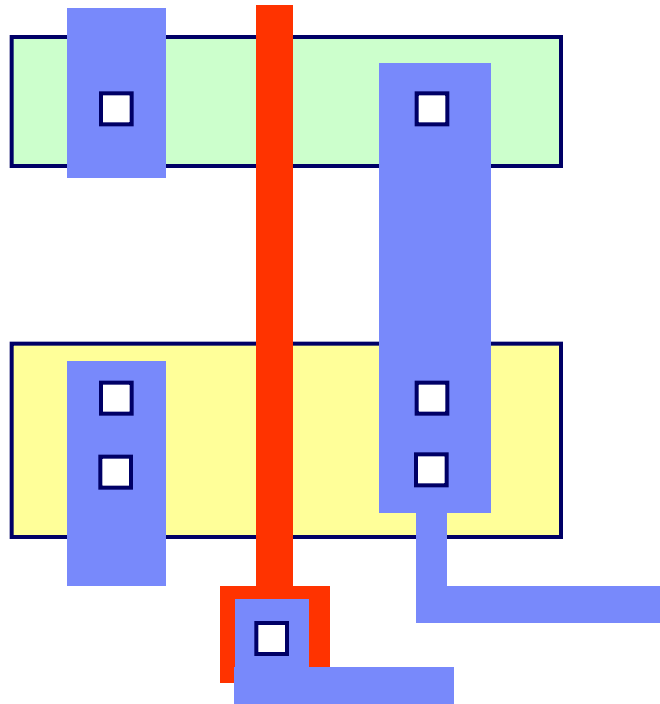
Transistor Level

- ▶ Model on CMOS transistor level
 - depending on application function modeled as resistive switches
 - used in functional equivalence checking
 - or full differential equations for circuit simulation
 - used in detailed timing analysis



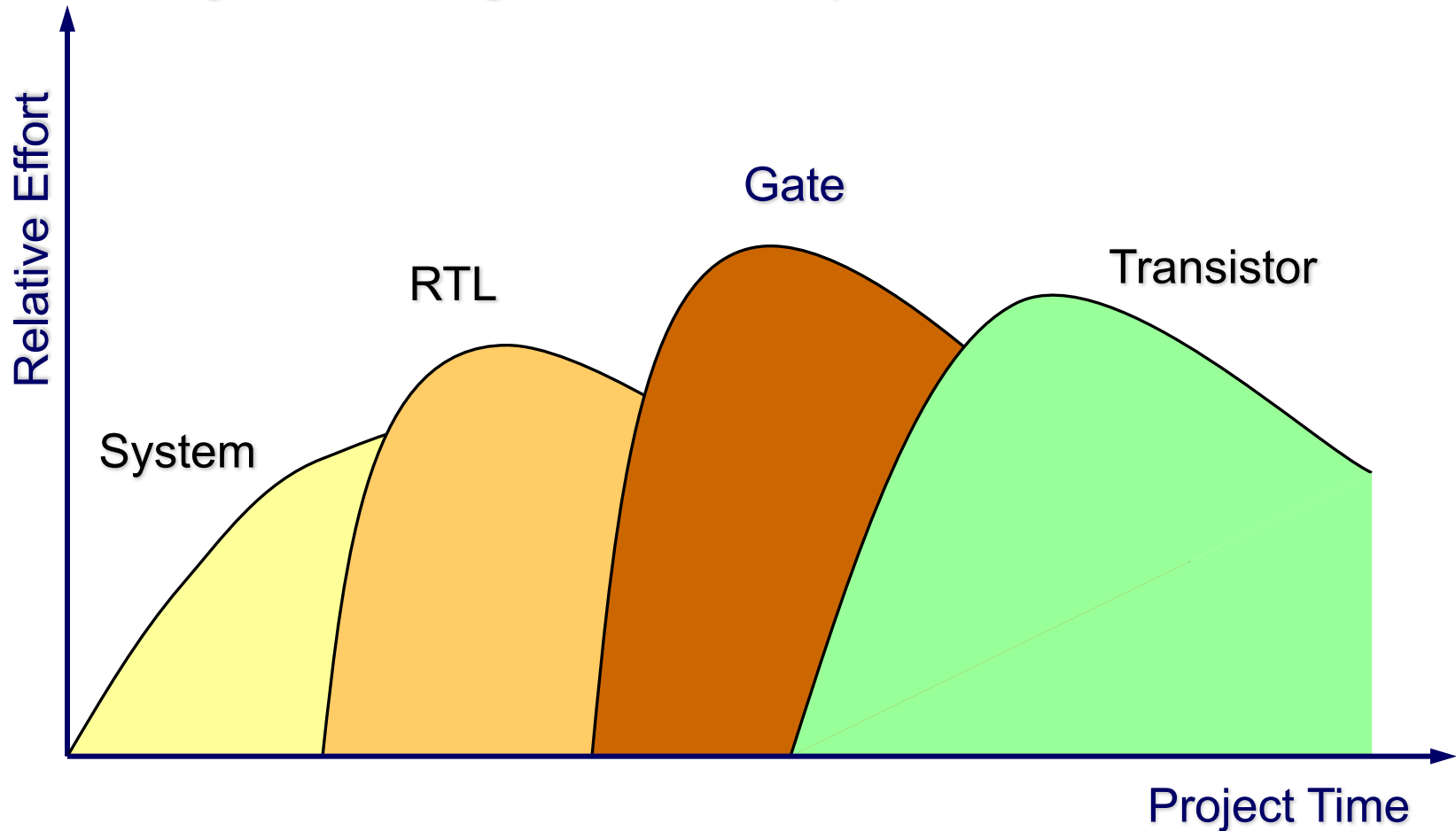
Layout Level

- ▶ Transistors and wires are laid out as polygons in different technology layers such as diffusion, polysilicon, metal, etc.



Design Progression

- Design phases overlap to large degrees
- Parallel changes on multiple levels, multiple teams
- Tight scheduling constraints for product



Design Challenges

- ▶ Systems are becoming huge, design schedules are getting tighter
 - > 2 billion transistors for high end processors
 - > Half Million lines of C-code to describe system behavior
 - > 5-10 Million lines of RTL code
 - > 50 Million gates becoming common for ASICs
- ▶ Design teams are getting very large for big projects
 - several hundred people spread across the globe
 - differences in skills
 - concurrent work on multiple levels
 - management of design complexity and communication very difficult
- ▶ Design tools are becoming more complex but still evolving
 - typical designer has to run ~50 tools on each component
 - tools have bugs, interfaces do not line up etc.

Design Challenges

- ▶ Decision about design point very difficult
 - compromise between performance / costs / time-to-market
 - decision has to be made > 3 years before design finished
 - design points are increasingly difficult to predict without actually doing the design
 - scheduling of product cycles

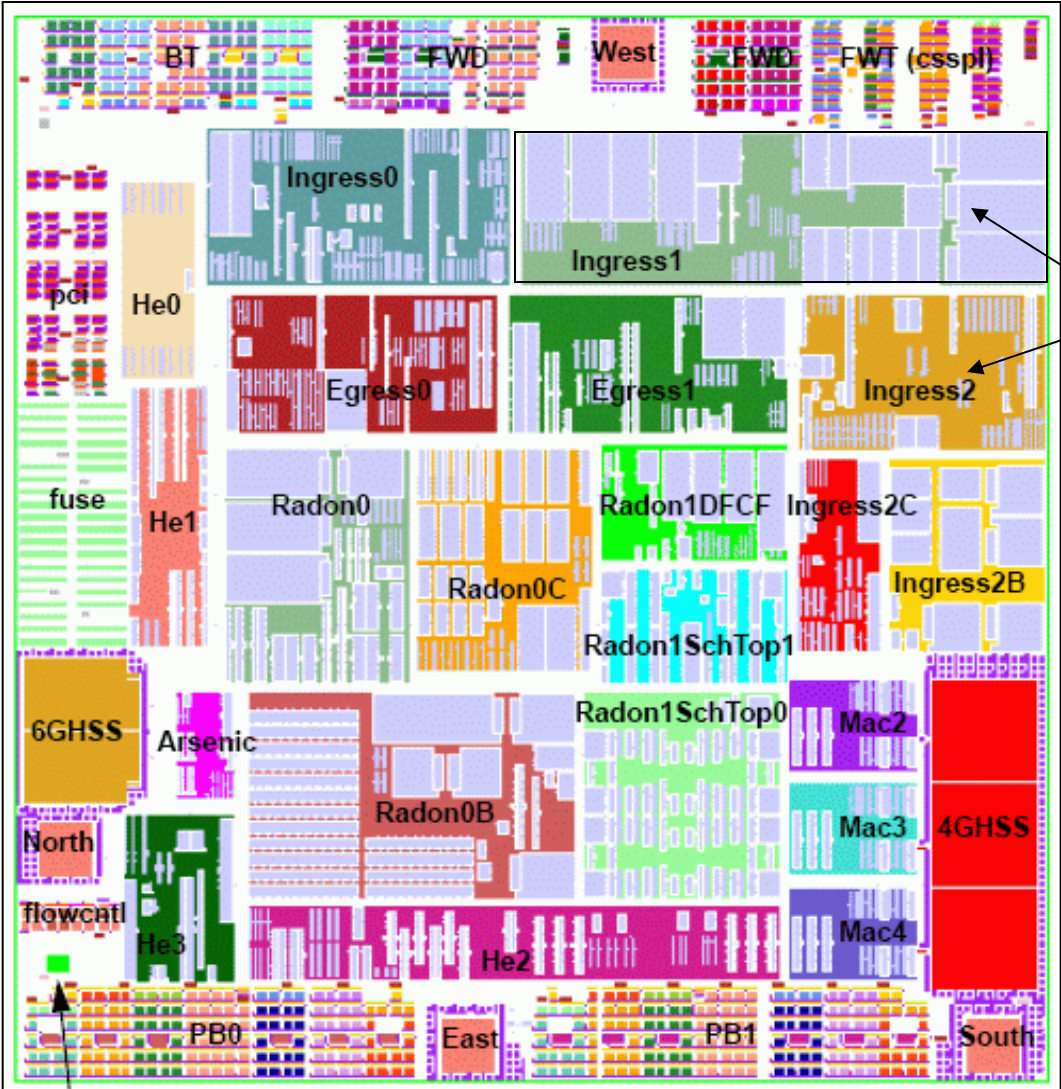
- ▶ Functional verification
 - simulation still main vehicle for functional verification but inadequate because of size of design space
 - results in bugs in released hardware that is very expensive to recover from (different in software ;-)

General Design Approach

- ▶ How do engineers build a bridge?
- ▶ Divide and conquer !!!!
 - partition design problem into many sub-problems which are manageable
 - define mathematical model for sub-problem and find an algorithmic solution
 - **beware of model limitations and check them !!!!!!!**
 - implement algorithm in individual design tools, define and implement general interfaces between the tools
 - implement checking tools for boundary conditions
 - concatenate design tools to general design flows which can be managed
 - see what doesn't work and start over

Generic Design Hierarchy

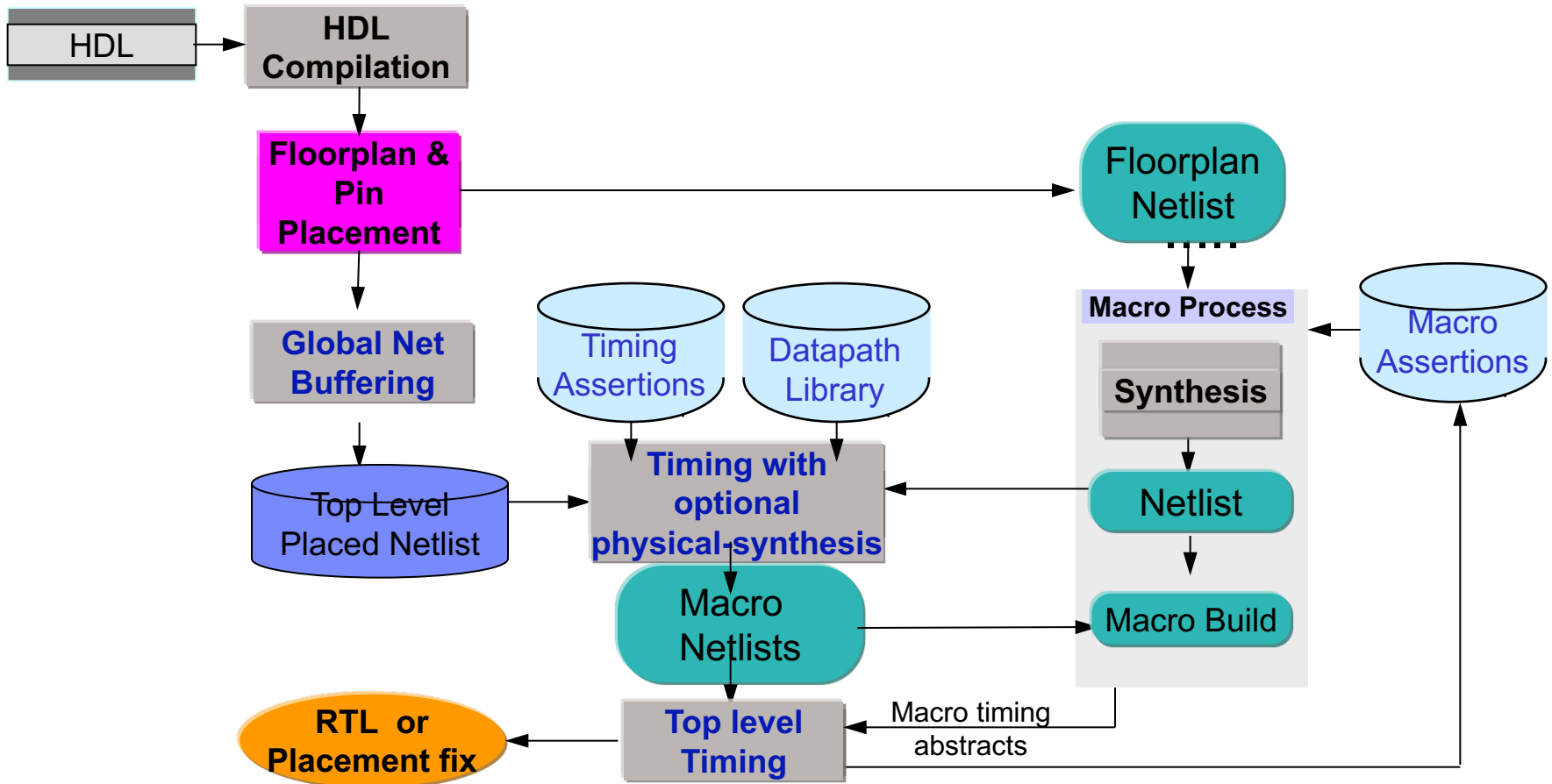
Chip Top Level



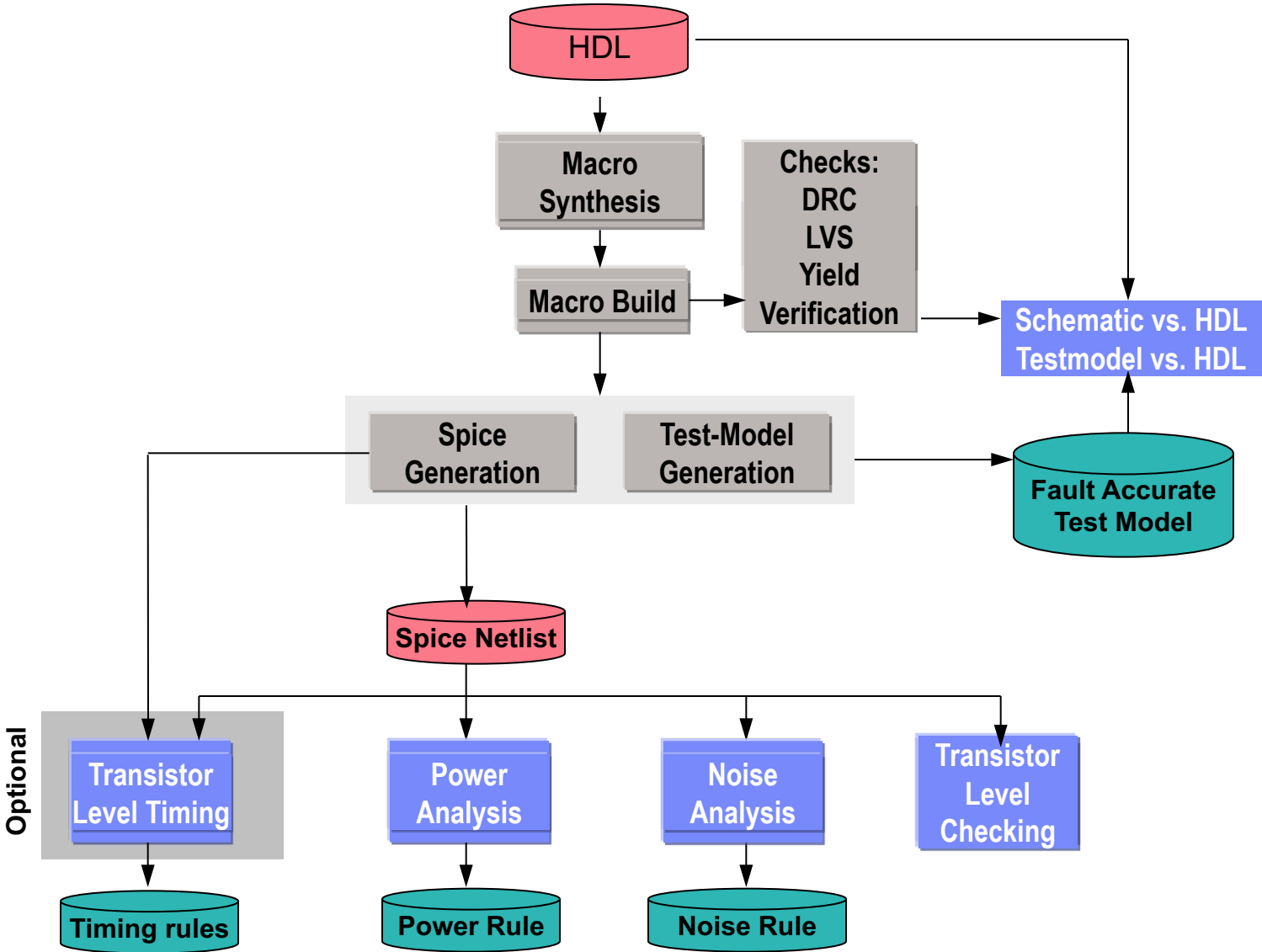
Macros

Hierarchical Design Flow: Top Level Flow

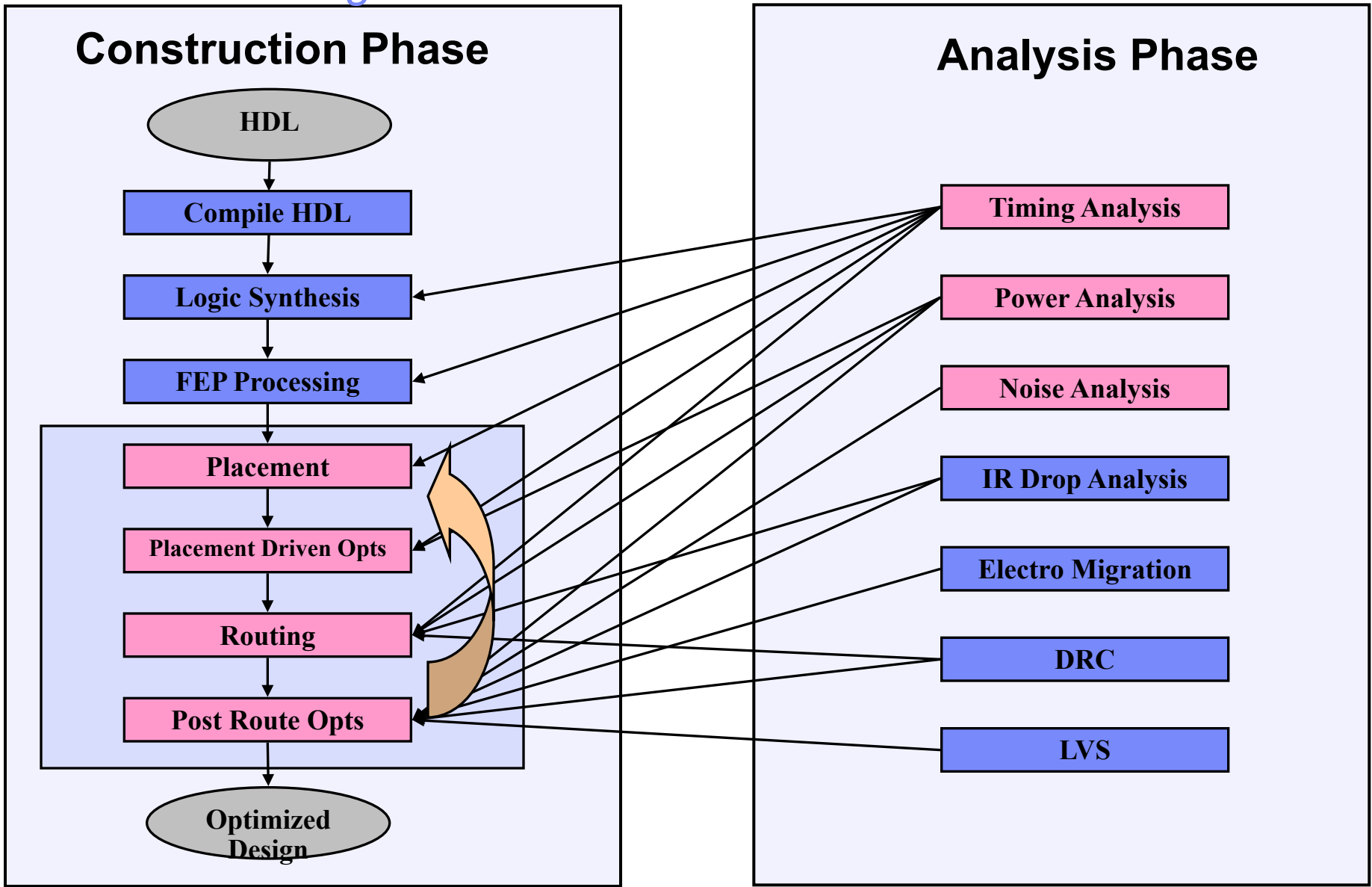
⋮



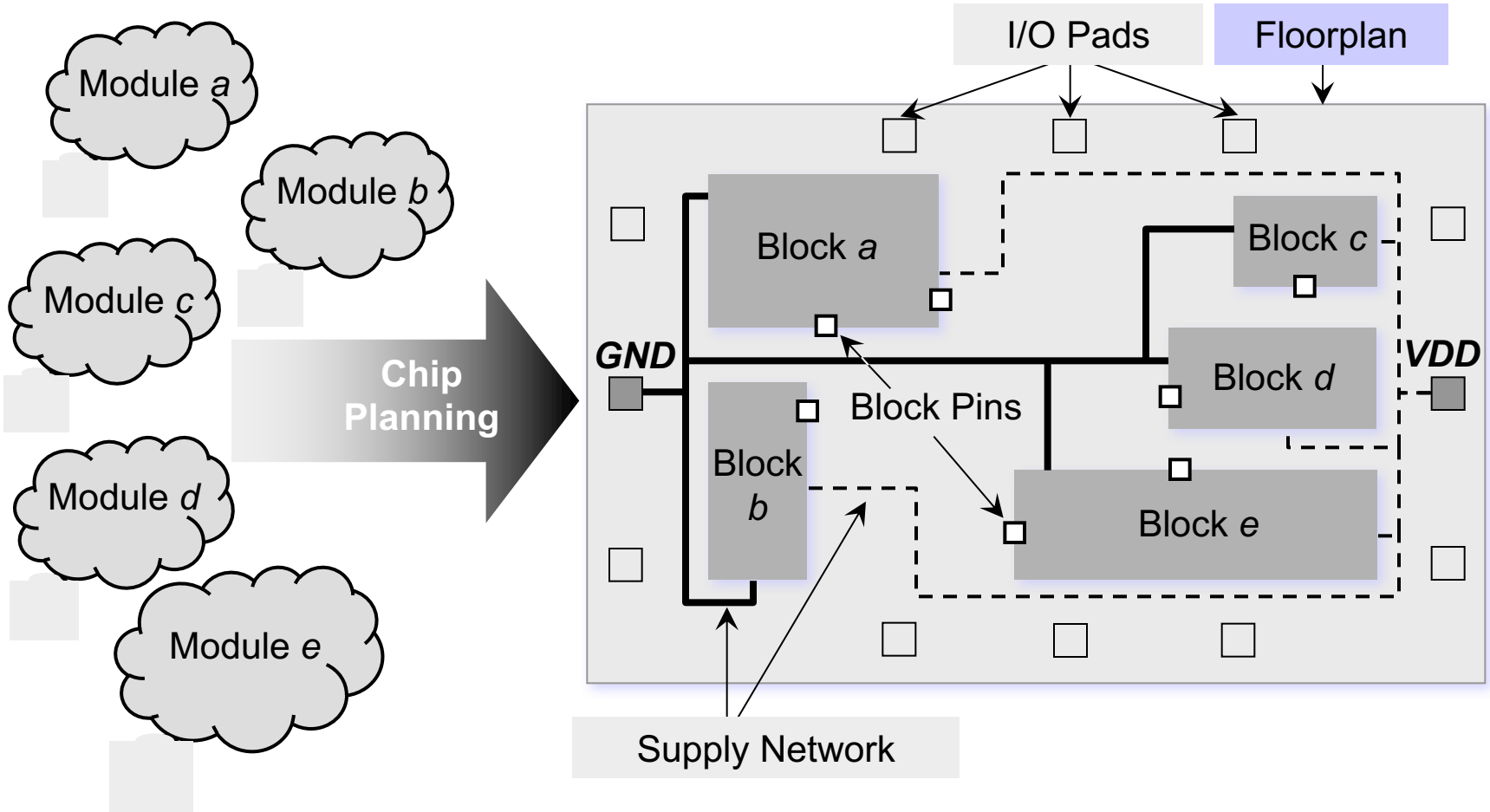
Hierarchical Design Flow: Macros



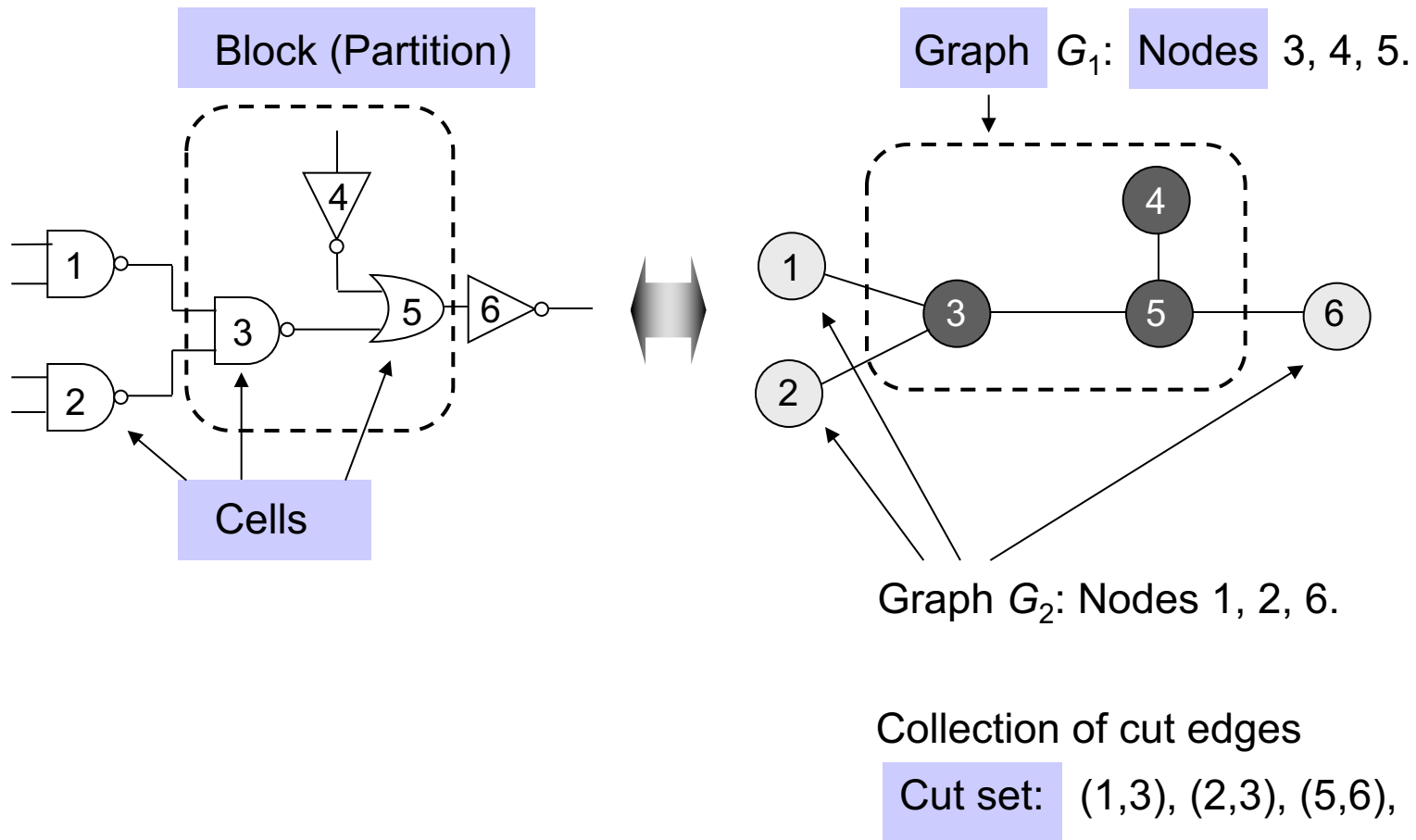
Block level Design Flow



Chip Planning



Chip partitioning



FloorPlanning

Example

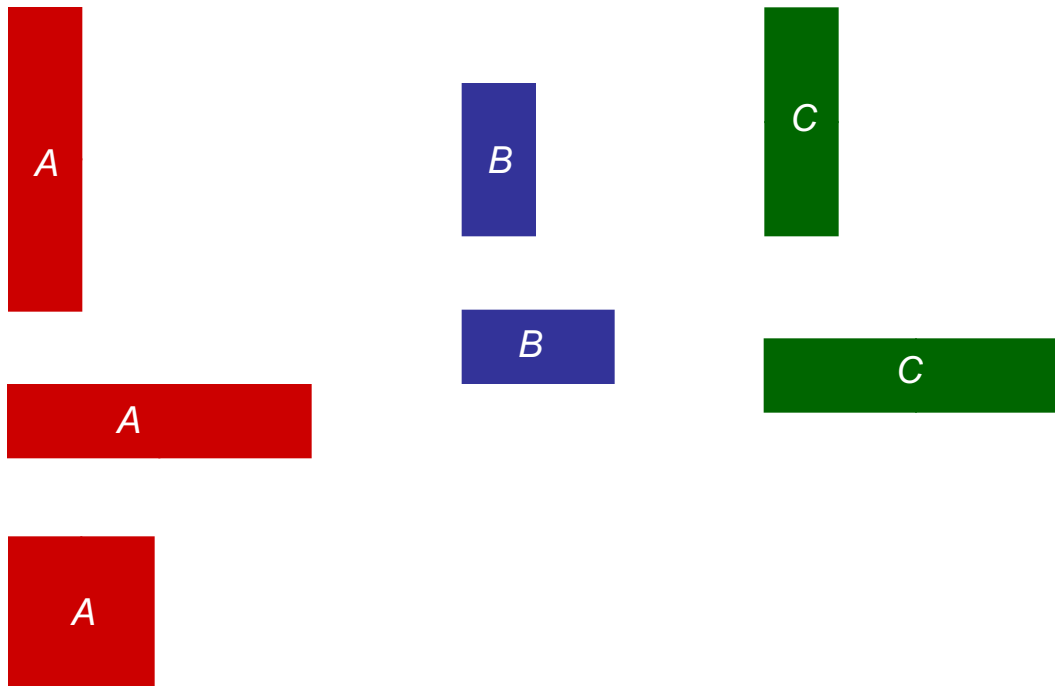
Given: Three blocks with the following potential widths and heights

Block A: $w = 1, h = 4$ or $w = 4, h = 1$ or $w = 2, h = 2$

Block B: $w = 1, h = 2$ or $w = 2, h = 1$

Block C: $w = 1, h = 3$ or $w = 3, h = 1$

Task: Floorplan with minimum total area enclosed



FloorPlanning

Example

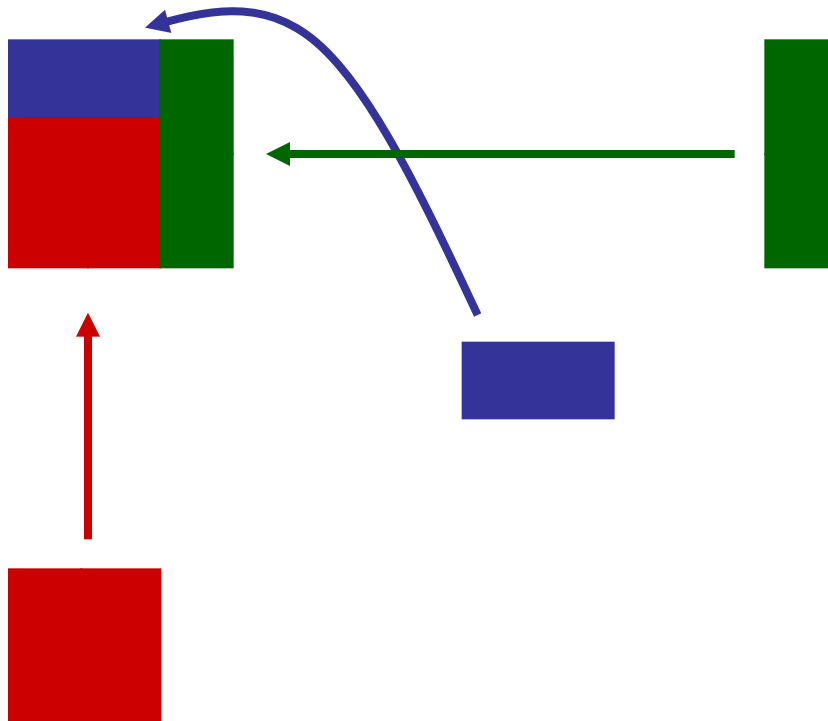
Given: Three blocks with the following potential widths and heights

Block A: $w = 1, h = 4$ or $w = 4, h = 1$ or $w = 2, h = 2$

Block B: $w = 1, h = 2$ or $w = 2, h = 1$

Block C: $w = 1, h = 3$ or $w = 3, h = 1$

Task: Floorplan with minimum total area enclosed



FloorPlanning

Example

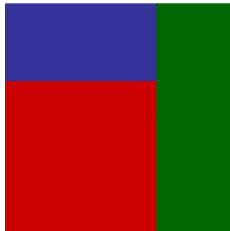
Given: Three blocks with the following potential widths and heights

Block A: $w = 1, h = 4$ or $w = 4, h = 1$ or $w = 2, h = 2$

Block B: $w = 1, h = 2$ or $w = 2, h = 1$

Block C: $w = 1, h = 3$ or $w = 3, h = 1$

Task: Floorplan with minimum total area enclosed



Solution:

Aspect ratios

Block A with $w = 2, h = 2$; **Block B** with $w = 2, h = 1$; **Block C** with $w = 1, h = 3$

This floorplan has a global bounding box with minimum possible area (9 square units).

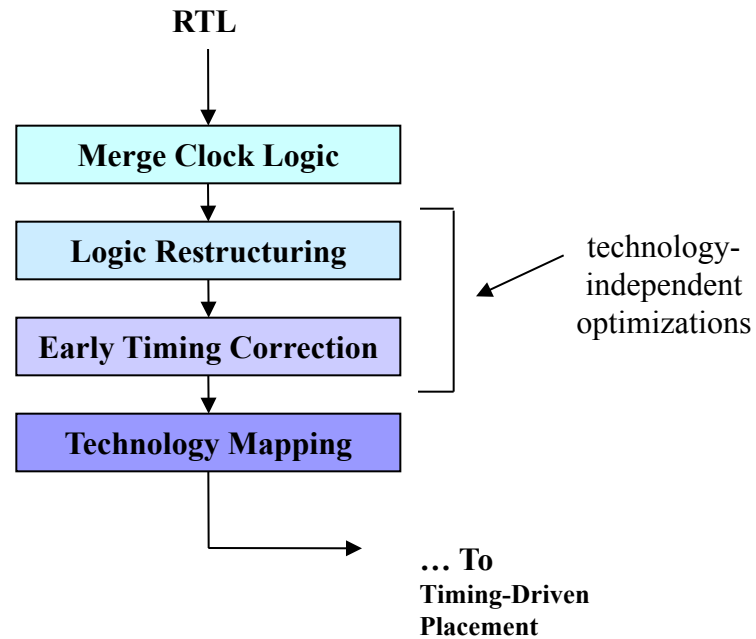
Floorplanning – Optimization Goal

- ▶ Area and shape of the global bounding box
 - Area of the global bounding box represents the area of the top-level floorplan
 - Minimizing the area involves finding (x,y) locations, as well as shapes, of the individual blocks.
- ▶ Total wirelength
 - Long connections between blocks may increase signal propagation delays in the design.
- ▶ Combination of area $area(F)$ and total wirelength $L(F)$ of floorplan F
 - Minimize $\checkmark \cdot area(F) + (1 - \checkmark) \cdot L(F)$
where the parameter $0 \leq \checkmark \leq 1$ gives the relative importance between $area(F)$ and $L(F)$
- ▶ Algorithms
 - Floorplan Sizing – find minimum floorplan area for a slicing floorplan in polynomial time. For Non slicing floorplan the problem is NP hard
 - Cluster Growth – Iteratively add blocks to the cluster until all blocks are assigned
 - Simulated Annealing – Iterative in nature


- ▶ Input: **RTL**; Output: **Gate level optimized netlist**
- ▶ Objective:
 - Minimize area
 - in terms of literal count, cell count, register count, etc.
 - Minimize power
 - in terms of switching activity in individual gates, deactivated circuit blocks, etc.
 - Maximize performance
 - in terms of maximal clock frequency of synchronous systems, throughput for asynchronous systems
 - Any combination of the above
 - combined with different weights
 - formulated as a constraint problem
 - “minimize area for a clock speed > 3 GHz”
 - More global objectives
 - feedback from layout: actual physical sizes, delays, placement and routing

Logic Synthesis

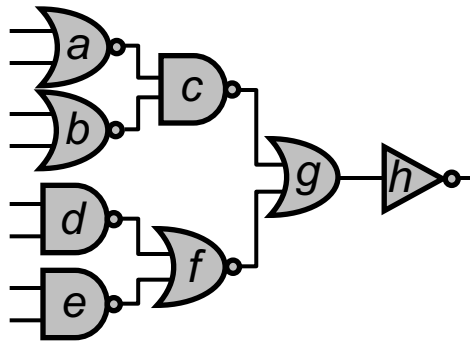
▶ Logic Synthesis Sub-steps...



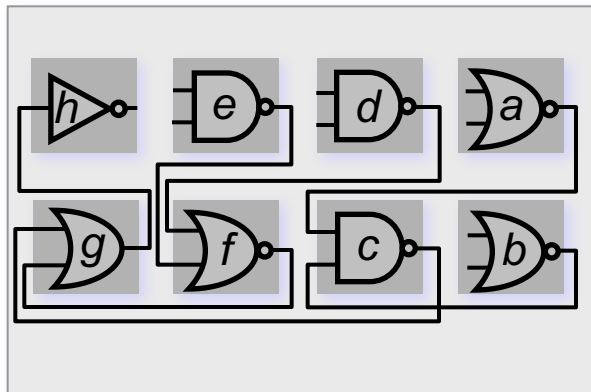
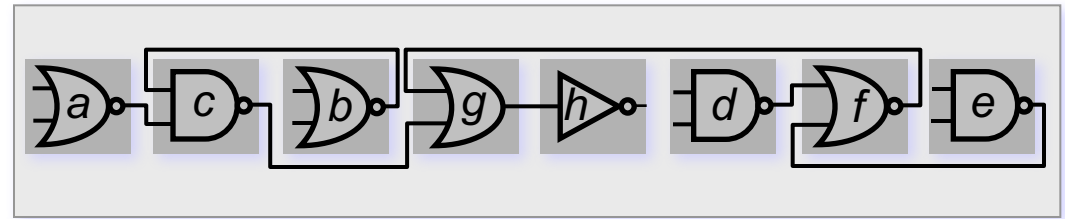
Placement

- ▶ Input
 - Netlist of connected circuit elements and nets
- ▶ Output
 - Exact locations of circuit elements without overlaps
- ▶ Goal is to guarantee routability, but extremely difficult to model this
- ▶ Usually use wirelength minimization
 -  Length(net i)
- ▶ How to estimate length of a net
 - Half perimeter wire length model (HPWL)
 - Minimum Spanning Tree (MST)
 - Steiner Tree (ST)

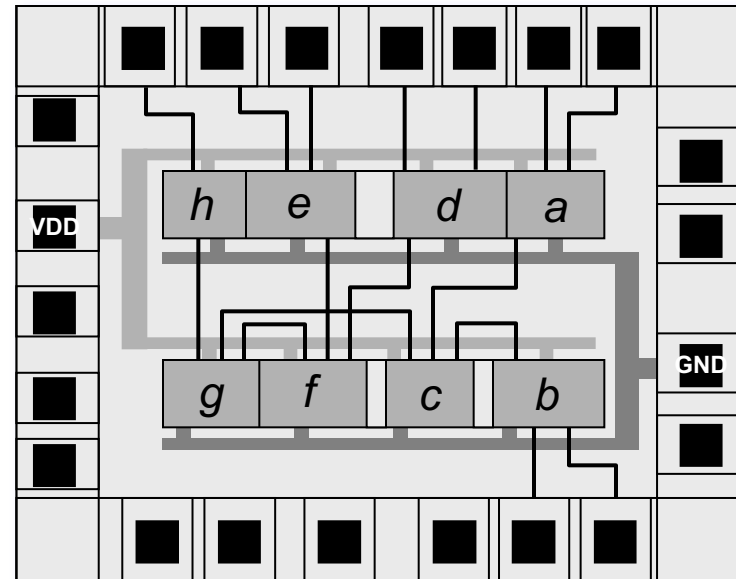
Placement



Linear Placement



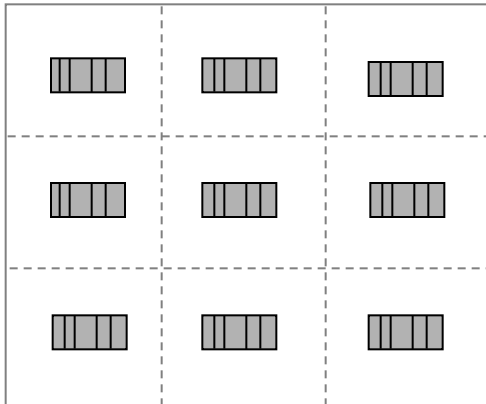
2D Placement



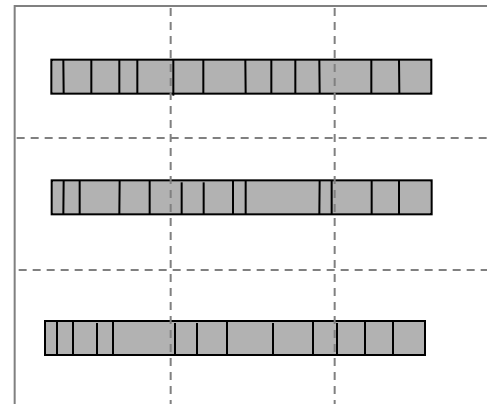
Placement and Routing with Standard Cells

Placement - Steps

Global
Placement



Detailed
Placement



Routing

▶ Input

- Placed netlist and technology information

▶ Output

- Physically connected cells

▶ Goal is to

- determine the necessary wiring, e.g., net topologies and specific routing segments, to connect these cells
- while respecting constraints, e.g., design rules and routing resource capacities, and
- optimizing routing objectives, e.g., minimizing total wirelength and maximizing timing slack

Routing Overview

Netlist:

$$N_1 = \{C_4, D_6, B_3\}$$

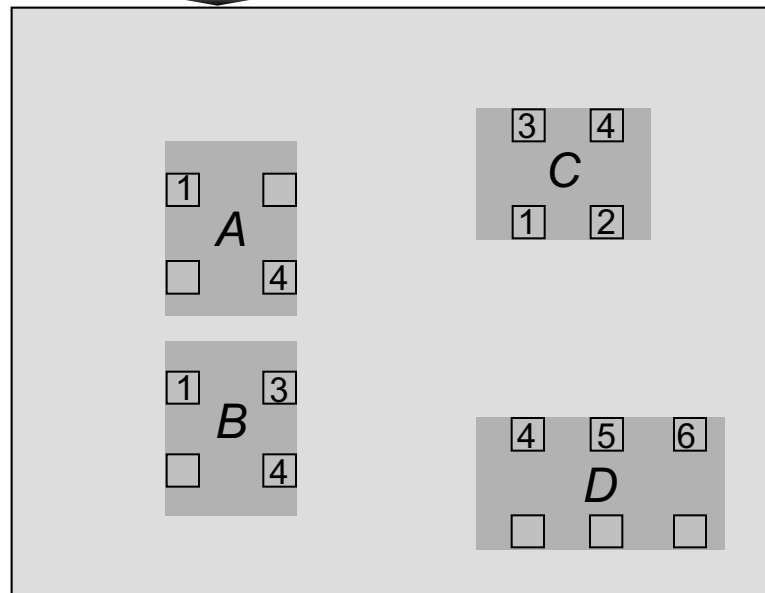
$$N_2 = \{D_4, B_4, C_1, A_4\}$$

$$N_3 = \{C_2, D_5\}$$

$$N_4 = \{B_1, A_1, C_3\}$$

Technology Information
(Design Rules)

Placement result



Routing Overview

Netlist:

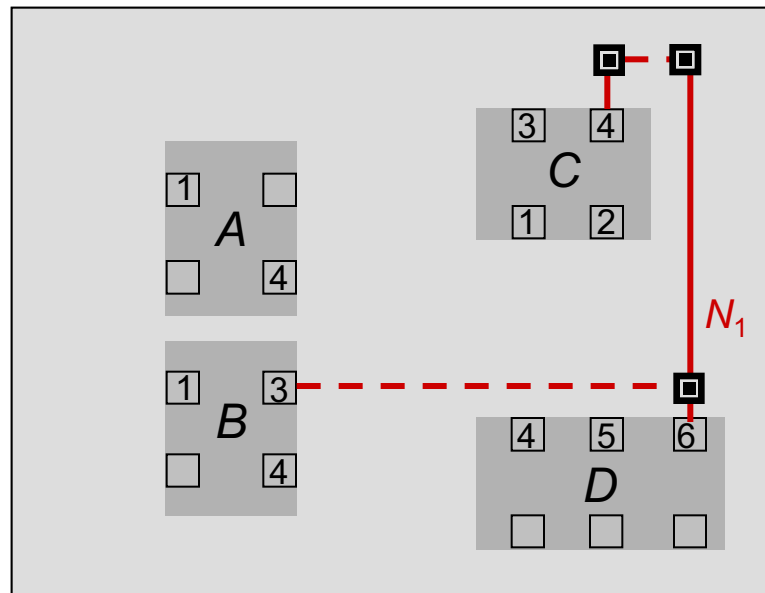
$N_1 = \{C_4, D_6, B_3\}$

$N_2 = \{D_4, B_4, C_1, A_4\}$

$N_3 = \{C_2, D_5\}$

$N_4 = \{B_1, A_1, C_3\}$

Technology Information
(Design Rules)



Routing Overview

Netlist:

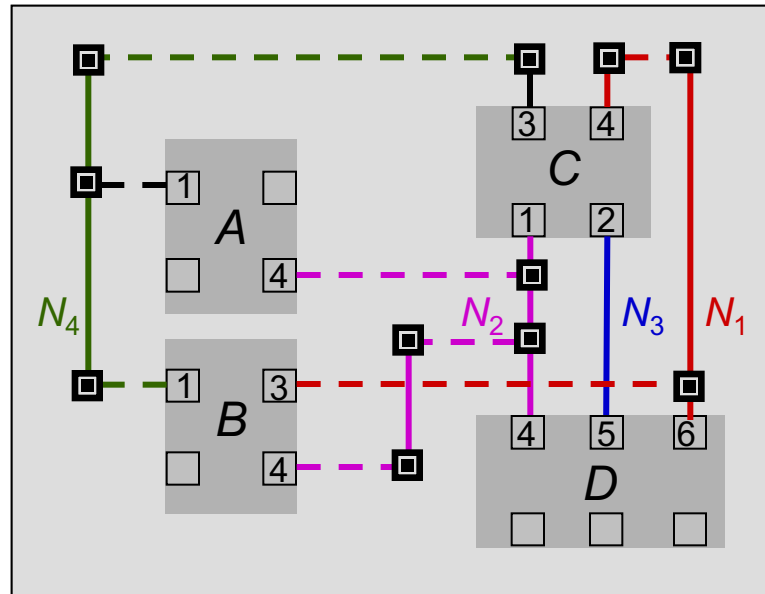
$N_1 = \{C_4, D_6, B_3\}$

$N_2 = \{D_4, B_4, C_1, A_4\}$

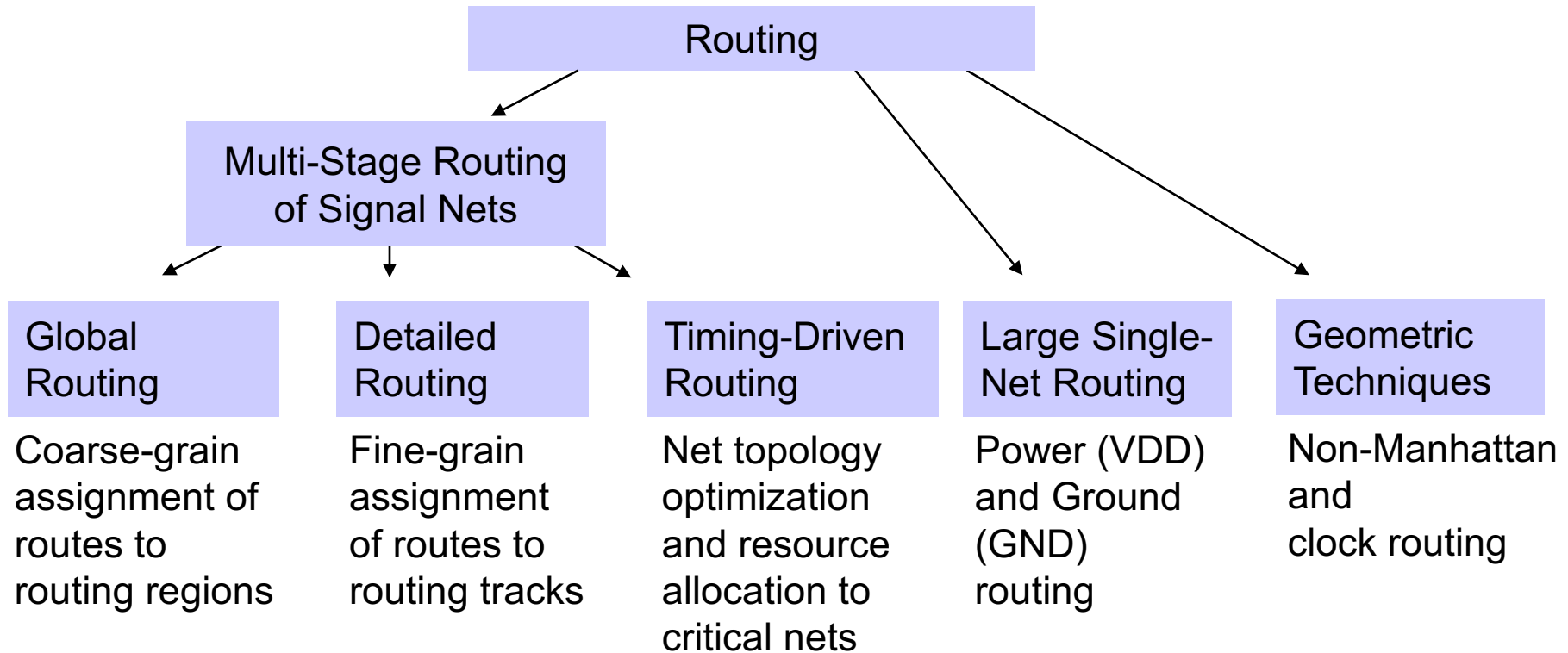
$N_3 = \{C_2, D_5\}$

$N_4 = \{B_1, A_1, C_3\}$

Technology Information
(Design Rules)

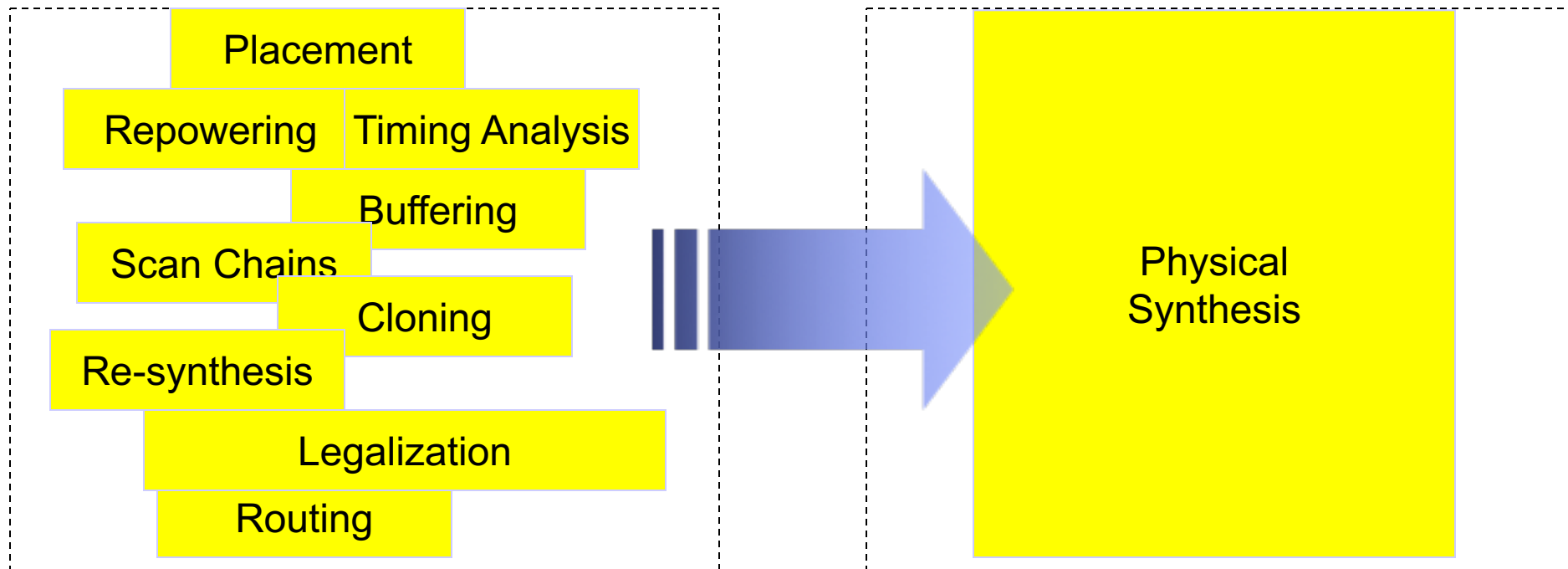


Routing Steps



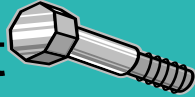
What is Physical Synthesis?

Combines multiple separate steps into one to perform timing (design) closure



Physical Synthesis Nuts and Bolts

Placement



(find non-overlapping locations to minimize wirelength)

Critical Path Opts



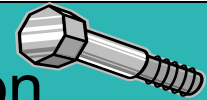
(incremental synthesis to optimize most critical regions)

Electrical Correction



(buffer and repower to fix slew and cap constraints)

Compression



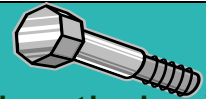
(optimize rest of critical regions)

Legalization



(place those buffers and logic in legal locations)

Recovery



(reduce resource without hurting timing)

Routability Analysis



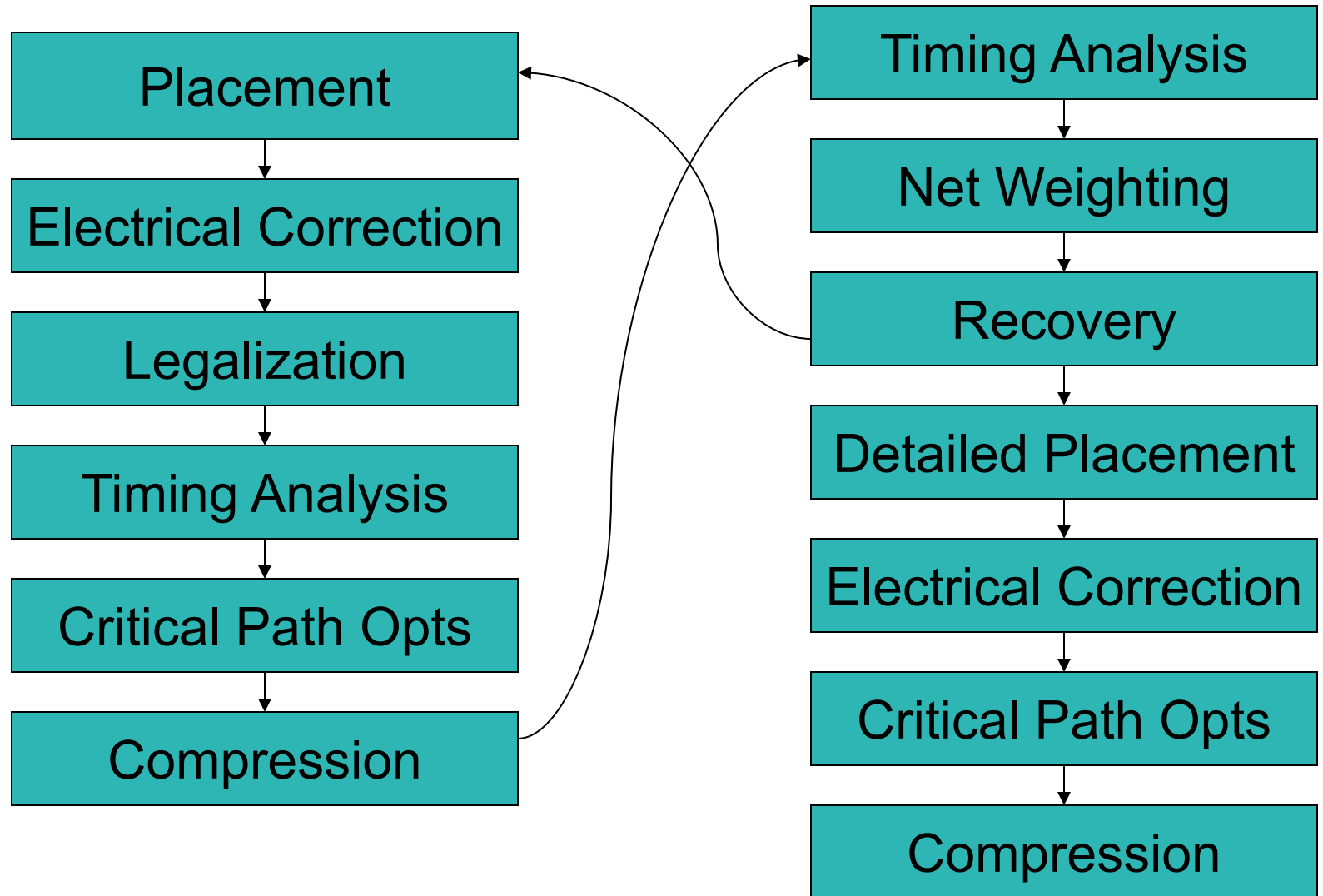
(identify routing hotspots)

Timing Analysis



(identify critical logic)

Physical Synthesis Flow



Static Timing Analysis

▶ Purpose

- Ensure design will operate properly at a target frequency

▶ Inputs

- Netlist
- Library rules
- User-specified assertions/constraints

▶ Outputs

- List of (failing) tests
- List of (failing) paths
- Abstract

What is timing analysis?

- ▶ Problem: My boss usually leaves home at 7:30 a.m. At what time should I leave home to be absolutely sure I reach the office before him?



My commute:
10 mins.
+5 mins. for “late mode”



Weather, traffic, signals



Weather, traffic, signals



Answer: 7:35 a.m.



My boss's commute:
30 mins.
-10 mins. for “early mode”

Slight twist on the problem

- ▶ Problem: My boss usually leaves home at 7:30 a.m. If I also leave home at 7:30 a.m., does that give me some slack?



My commute:
10 mins. +5 mins. for “late mode”

Answer: I arrive no later than $7:30 + 10 + 5 = 7:45 =$ my late arrival time

My boss arrives no sooner than $7:30 + 30 - 10 = 7:50 =$ his early arrival time

My required arrival time = my boss’s early arrival time = 7:50 a.m.

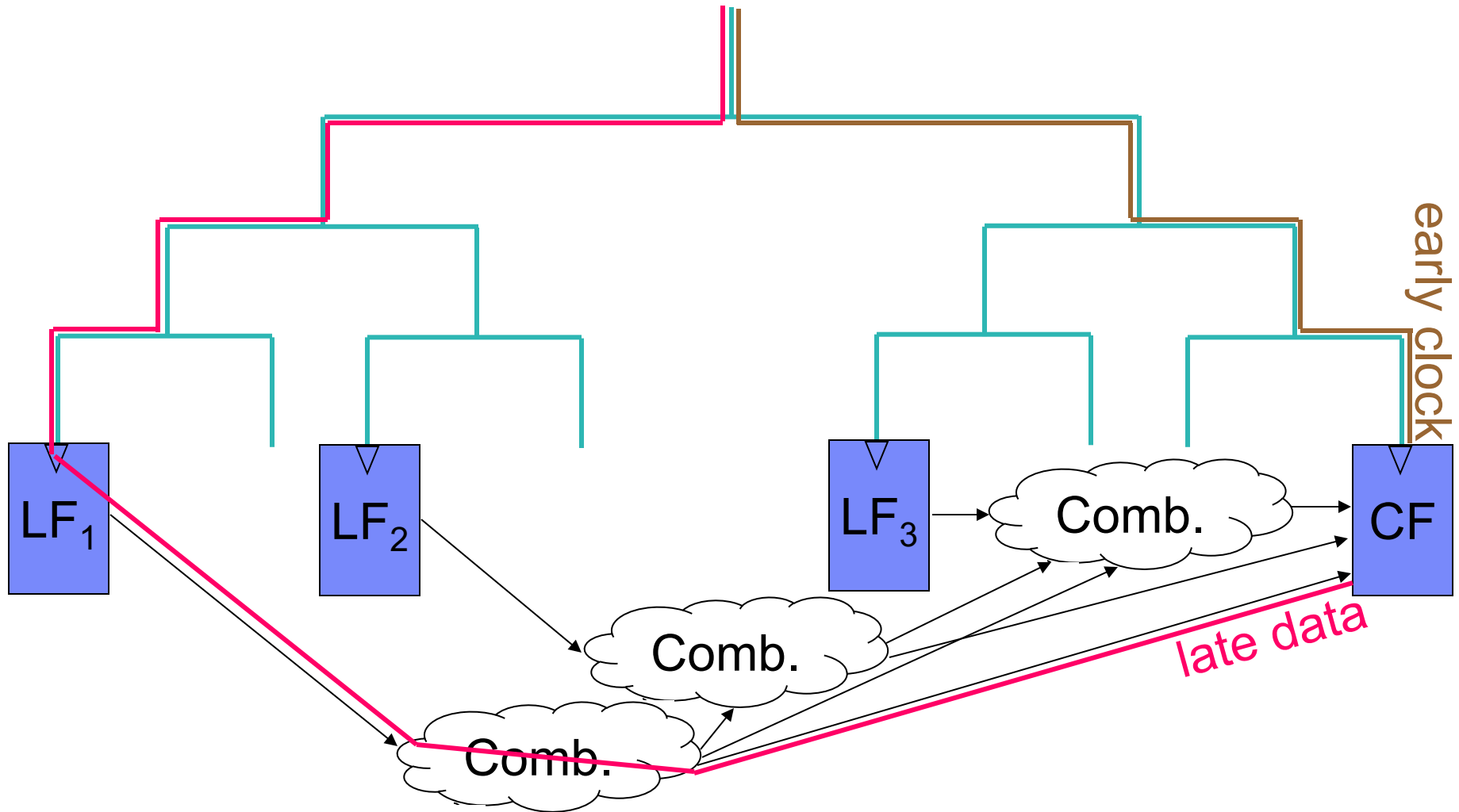
I have slack = required arrival time – arrival time = at least 5 minutes!

s

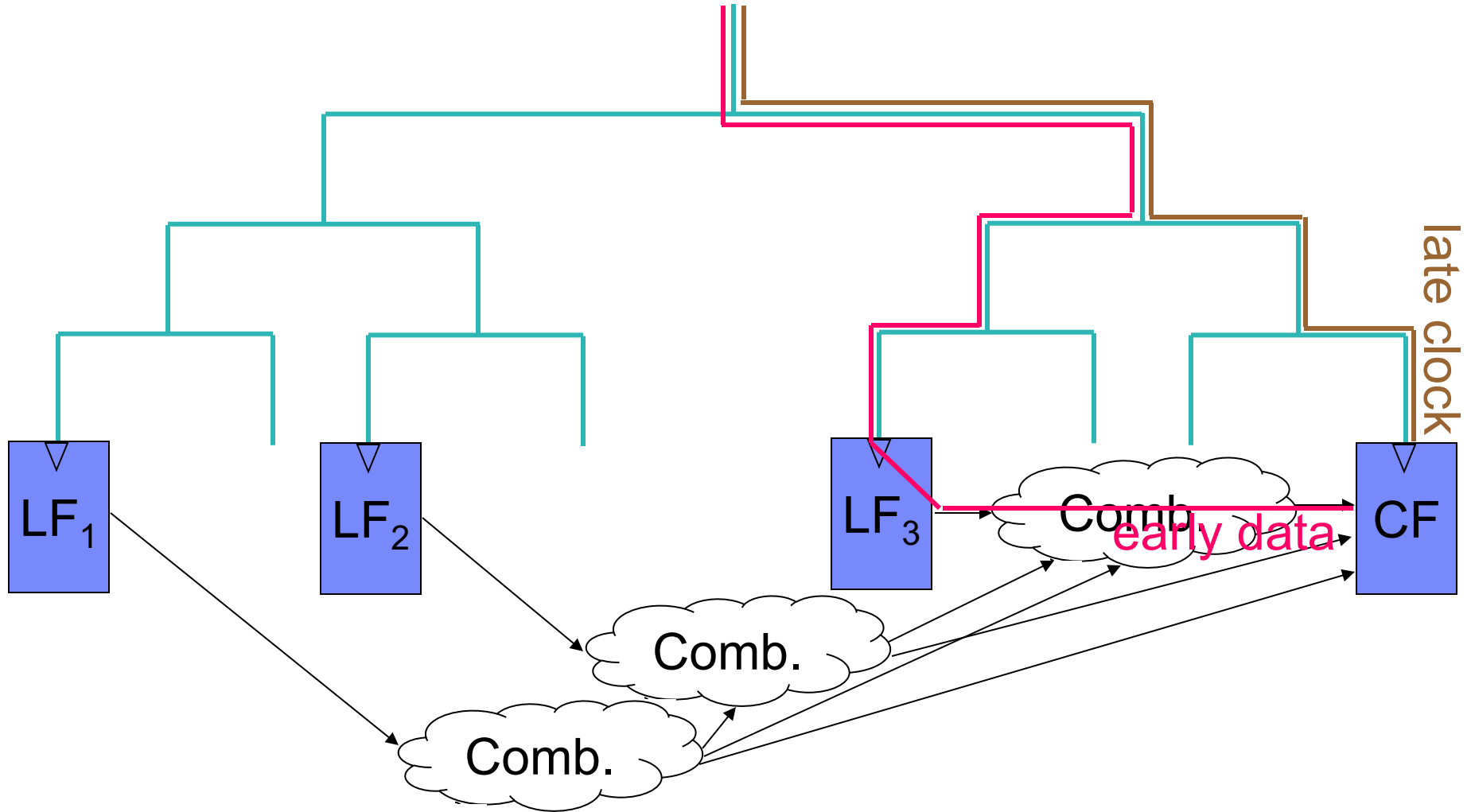


My boss’s commute:
30 mins.
-10 mins. for “early mode”

So it is with circuits: setup tests



Hold tests



Timing analysis of circuits

- ▶ With every signal, we associate four arrival times
 - Early (rise and fall)
 - Late (rise and fall)
- ▶ Late arrival time is 100 ps means, “if at all a signal switches in any given cycle, it will be stable after 100 ps”
- ▶ Early arrival time is 50 ps means, “if at all a signal switches in any given cycle, it will not change from its previous cycle stable value sooner than 50 ps”
- ▶ Setup test: compare late data arrival time to early clock
- ▶ Hold test: compare early data arrival time to late clock
- ▶ Requires just one simple pass of the circuit (or “timing graph”);

Power

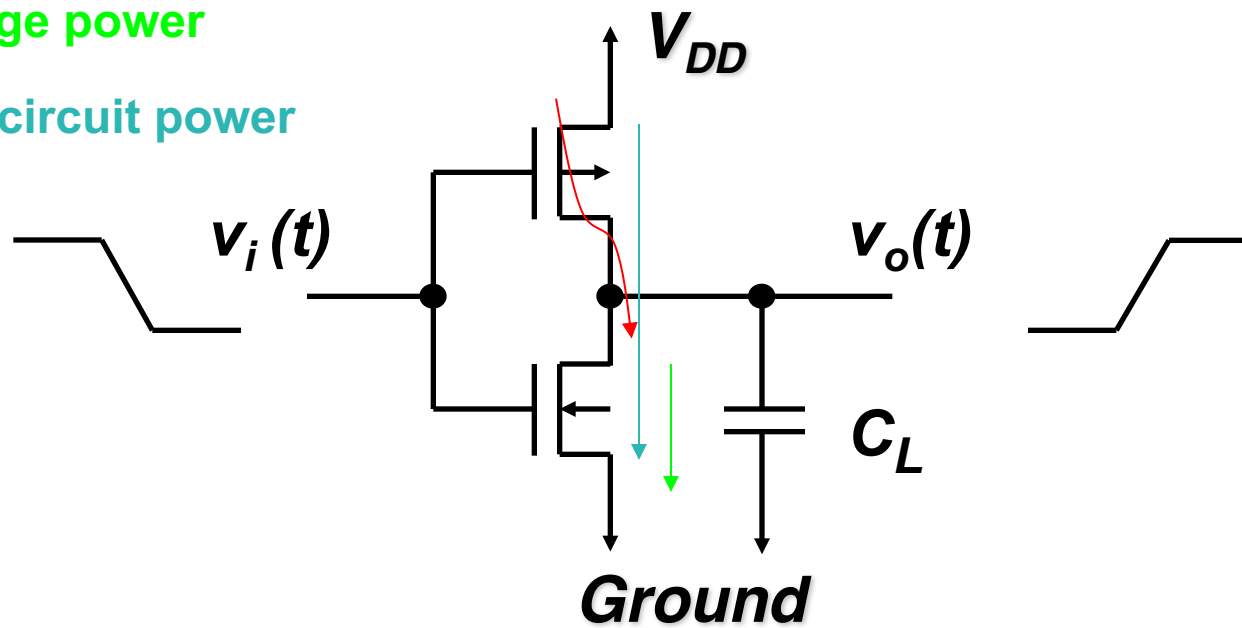
- ▶ Dynamic
 - Power due to Signal transitions.
 - Short Circuit component.
- ▶ Static
 - Leakage power (due to leakage currents).
- ▶ Clock Power

Power Components

Dynamic power

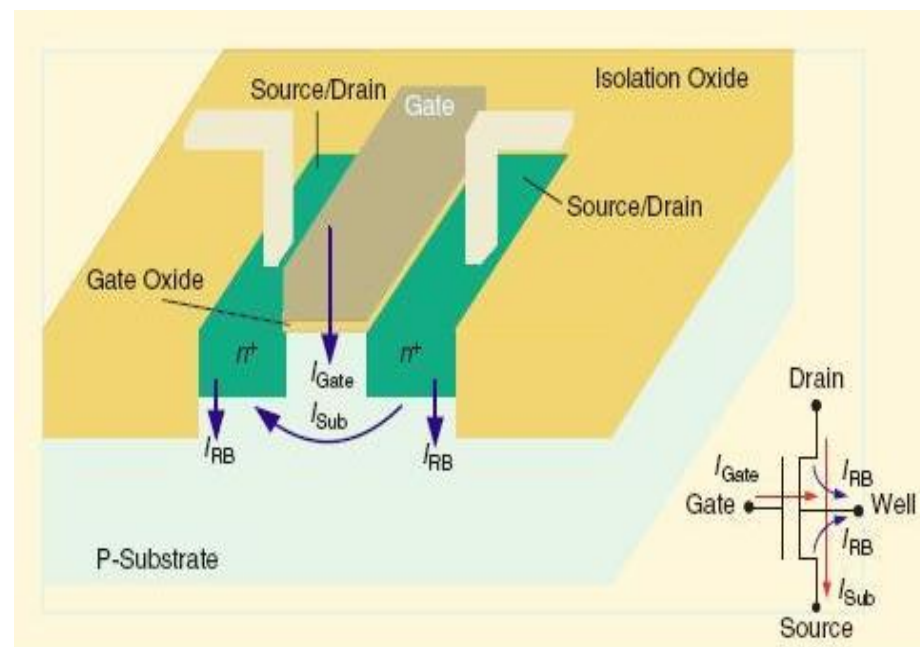
Leakage power

Short circuit power



Leakage Power

- ▶ **Three dominant leakage mechanisms:**
 - **Subthreshold Leakage** - Strong dependence on VDD, Process and Temperature
 - **Gate Oxide Leakage** - Strong dependence on VDD and T_{ox}
 - **Drain Junction Leakage** – Reverse Bias current between Source/Drain and the substrate, due to high doping concentrations.

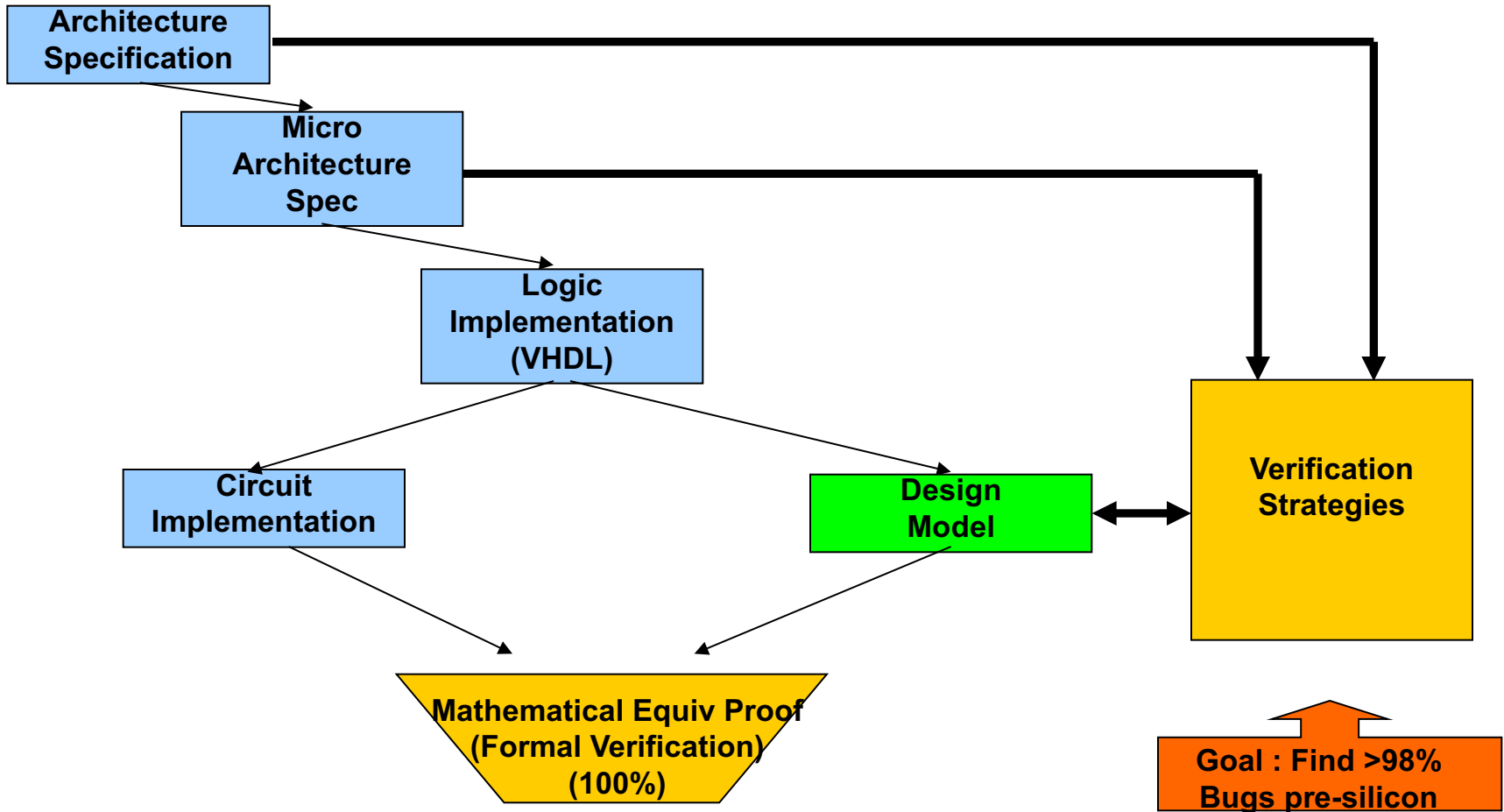


Factors affecting Dynamic Power

$$P_{avg} = 0.5 * C_{load} * VDD^2 * f * SF$$

- ▶ **Capacitance** : Function of fan-out, wire length, transistor sizes
- ▶ **Supply Voltage** : The quadratic term has been dropping with successive generations
- ▶ **Activity factor** : How often, on average, do wires switch
- ▶ **Clock frequency** : Increasing, and now decreasing with the new multi-core and many-core systems.

Verification



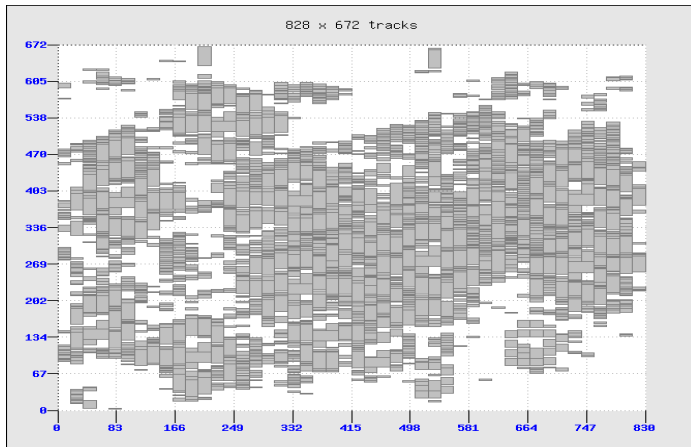
Courtesy:K. W. Roesner

Late Engineering Changes (ECOs)

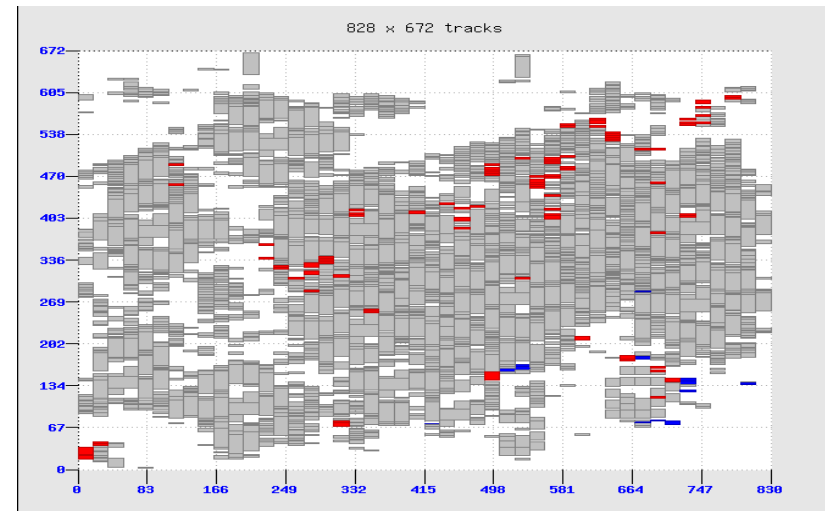
New(Modified) RTL

```
--%EPINS -- TurboVHDL: End Pins Definition. --:474
--:475
--%entity --:476
Entity l3cpr_zmac Is --:192
  Port ( --:192
    gnd : Inout power_logic;
    vdd : Inout power_logic;
    act  : In std_ulogic;
    align_0 : In std_ulogic;
    func_scan_in : In std_ulogic;
    func_scan_in_opt : In std_ulogic_vector;
```

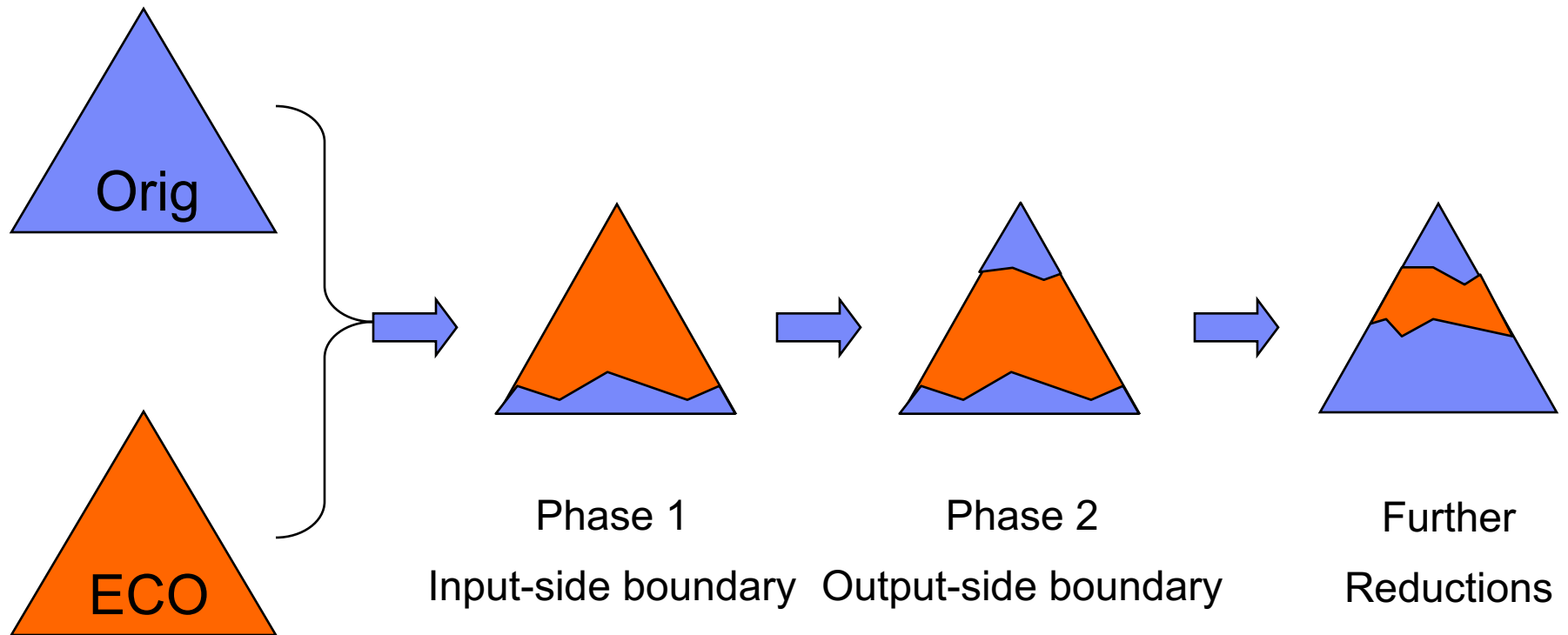
Original Synthesized/Routed Netlist



Modified Circuit



ECO Flow



Release to Manufacturing

- ▶ **Final edits to the layout are made**
- ▶ **Metal fill and metal stress relief rules are checked**
- ▶ **Manufacturing information such as scribe lanes, seal rings, mask shop data, part numbers, logos and pin 1 identification information for assembly are also added**
- ▶ **DRC and LVS are run to verify the correctness of the modified database; Electrical rule checking;**
- ▶ **'Tapeout' documentation is prepared prior to release of the GDSII to the foundry**
- ▶ **Pad location information is prepared, typically in a spreadsheet**
- ▶ **Custom-manual edits of the mask layers**
- ▶ **Manufacturing steps**
 - generation of masks; silicon processing; wafer testing; assembly and packaging; manufacturing test