Section II: Basic Issues

Overview

- The Big Five Dimensions of Physical Design
- Hierarchy Pros and Cons
- RTL practices
- Tools, Machines, Flows
- Data Prep: Netlists and IP
- Packaging

The Big Five

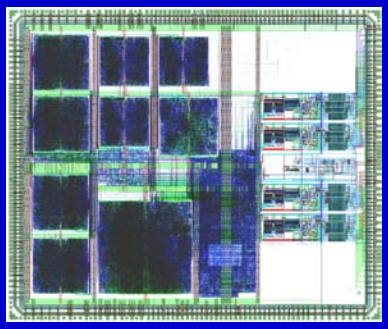
- Area: 5x5mm TSMC shuttle runs up to 18x18mm+ “boomer-class” chips
- Timing: DC to 350Mhz-ish in .18u
- Power: Battery to 60Watts+
- Schedule: 0 to Infinite
- Correctness

The Big Five, part 2

- Area, Timing, Power, Schedule all trade off with each other
- Correctness mostly trades off with Risk (Working chip vs.. Rock)
- Choose only two or three to optimize.

Classic Hierarchy

- Various blocks, channel routed together
- Padring constructed at top level

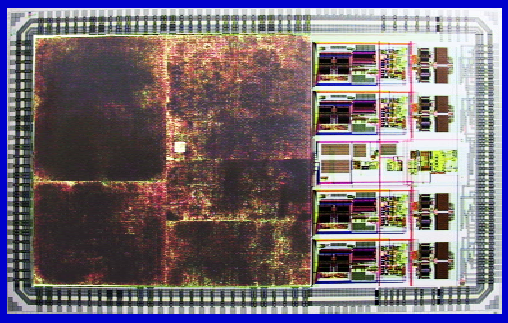


Classic Flat

- Everything at the top level, but….
- May be macro blocks that have been previous PnR'ed
- Simplest and most common technique in use today

Abutted Hierarchy

Nothing at the top level except macro blocks!



Hierarchy Pro

- Enables parallelism when building the chip
  - Engineers already decouple work on RTL and verification, physical design can also benefit
- Results come quicker if you can keep computers and licenses busy
  - Example: 1.5 million placeable objects, place and route in 24 hrs

Hierarchy Pro (#2)

- Memory capacity issues eased, don't need 64 bit machines, can use Linux (2X boost)
- Vulnerability to tool failure lower if runtime is lower. Easier to use patch releases to fix specific core dumps ant specific stages
- Trade runtime for even better quality of results
- Wireload models and other QOR influencing inputs can be tuned on a per-block basis more easily

Hierarchy Pro(#3)

- More deterministic results:
  - Global nets are the same for each build
  - Block boundary conditions (pins) are very similar from build to build
- Speed of builds can enable experimentation in chip Architecture/RTL/Floorplan/Synthesis. Take chip design to the next level

Hierarchy Pro(#4)

- Incremental, deterministic block closure becomes possible
- Easier to mix and mach different vendor tools for different blocks.
- Logic == Physical hierarchy at the block level enables rebuild of completely resynthesized netlist late in design (assuming it fits)

Hierarchy Pro (#5)

- Multiply instantiated PnR Blocks can be one block build, saving time and resources
- Clock distribution can take advantage of hierarchy to offer lower intra-block skew at the cost of higher inter-block skew

Hierarchy Con:

- Pin assignment difficult
- the “Horizon” effect
  - Global timing requires block constraints
  - ERC/DRC difficult
  - More clock tuning required

Hierarchy Con (#2)

- Channels are bad
  - Waste area
  - Coupling issues, or waste even more area
  - Repeater placement and power in channels can be difficult
  - Long timing paths around blocks
  - Feedthroughs difficult for tools, flow

Hierarchy Con (#3)

- Data management problems much worse
  - More files of ALL types
  - Experimentation creates even more versions of blocks..which is the one to tapeout?
  - Multiple tool usage only makes IP setup problems worse

Hierarchy Con (#4)

- Top level partitioning and floorplanning required, can be problematic
- Using hierarchy is fighting against tools not designed with it in mind
- Abutted block hierarchy can eliminate the incremental block closure capability
- Abutted block floorplanning can have issues with various snapping grids in the design (stdcell, power, bga bumps)

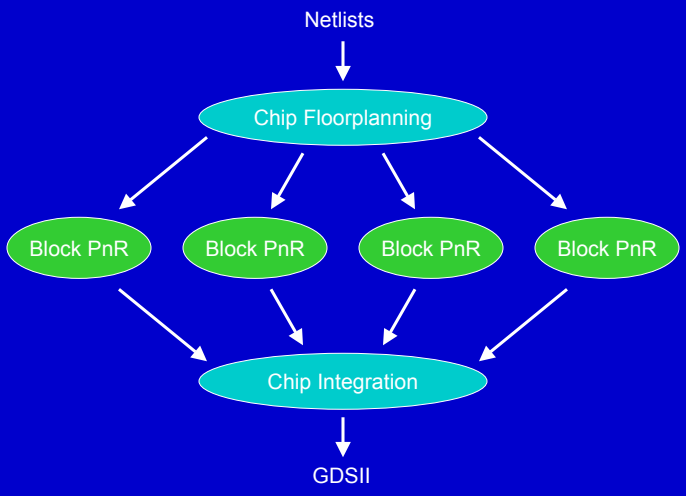
Hierarchy Con (#5)

- Multiply instantiated blocks have to diverge due to their different environments
- Classic Hierarchy requires a top level router tool as well as the block level router

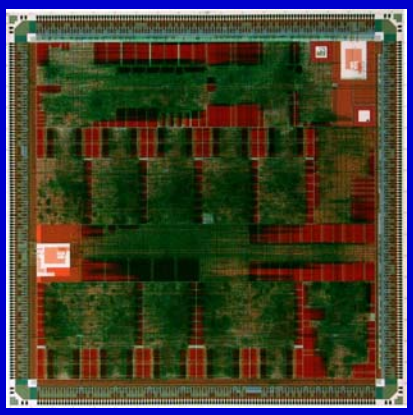
Hierarchy …..or not?

- Weigh design size, team experience, tool selections.
- Probably best to use modified flat design for now

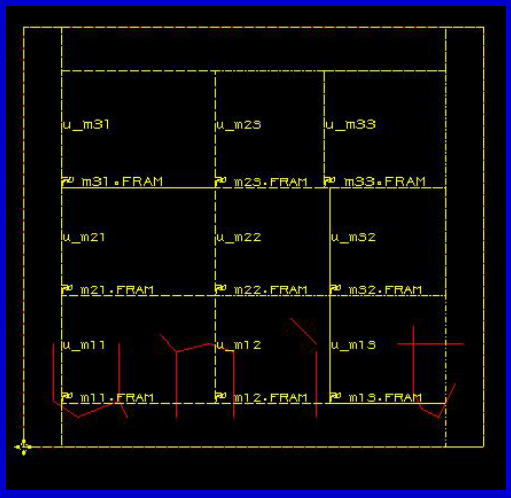
ReShape's Flow



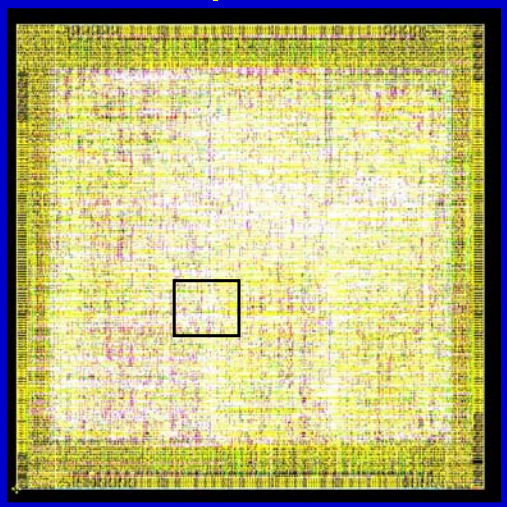
Abutted Chip Example 1



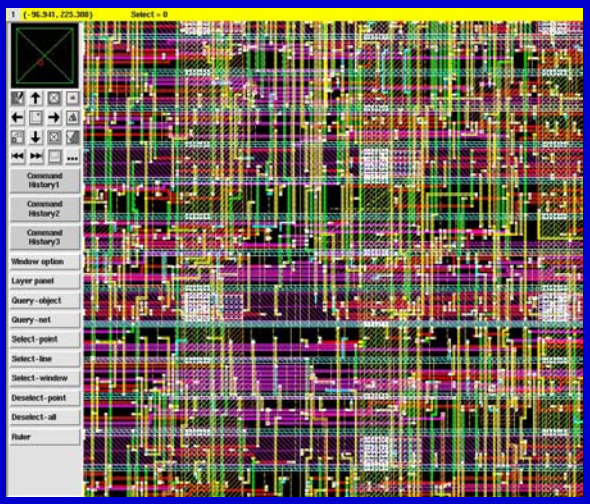
Chip Example 2



Routed Chip



Abutted Pins

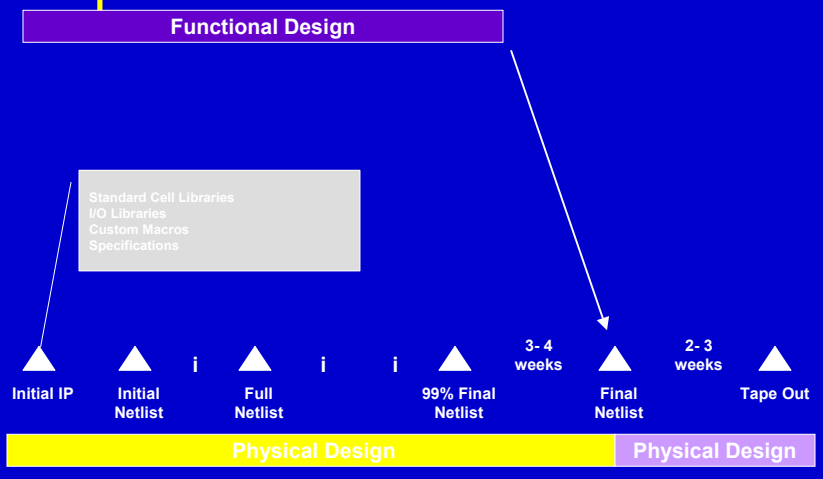


RTL Practices

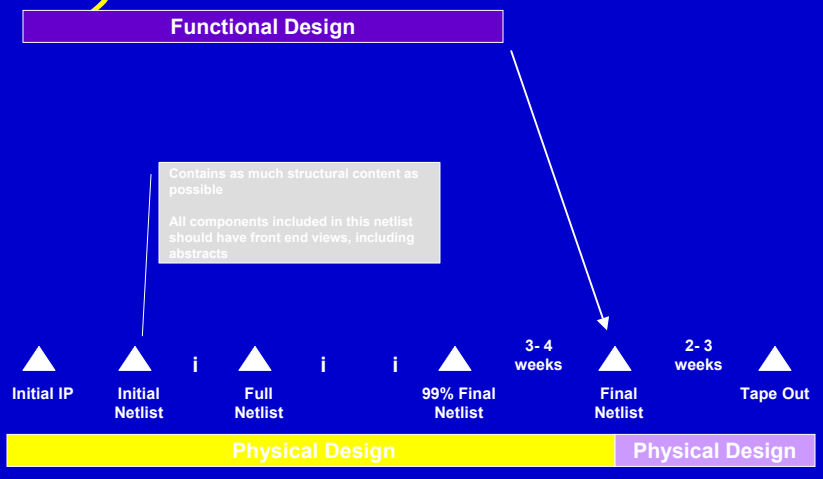
(or……. “The Rules of the Tavern”)

- Planning milestones
- Large scale issues
- Small scale issues

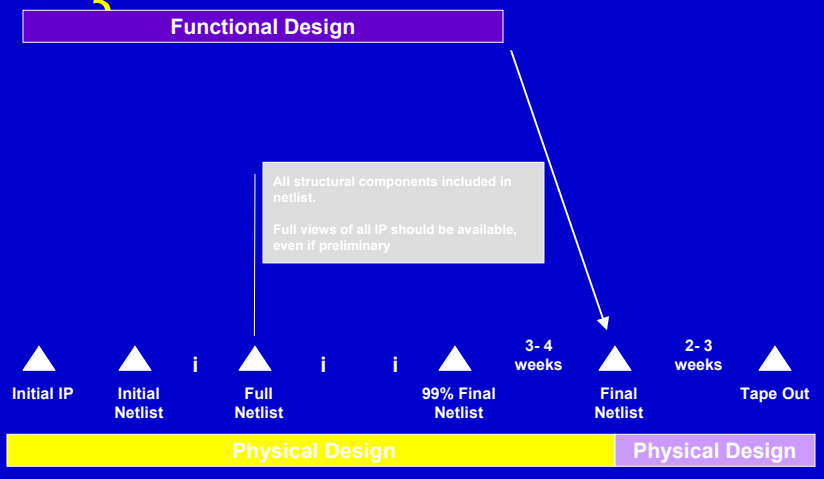
Concurrent Design Milestone 1



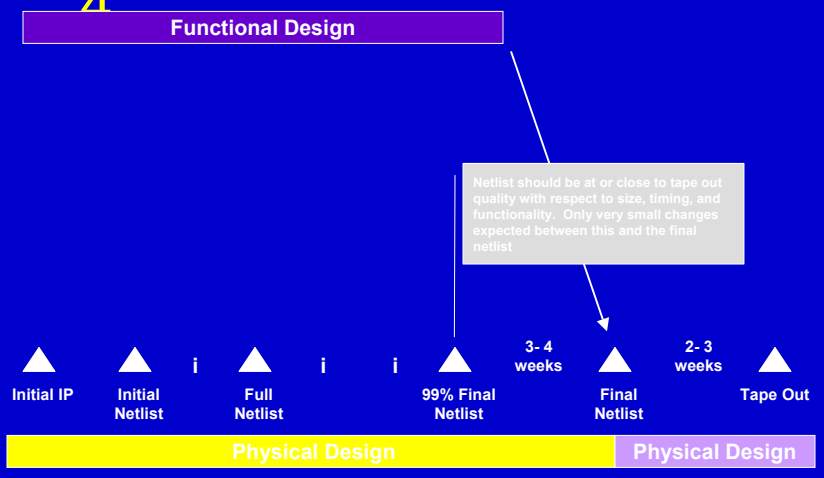
Concurrent Design Milestone 2



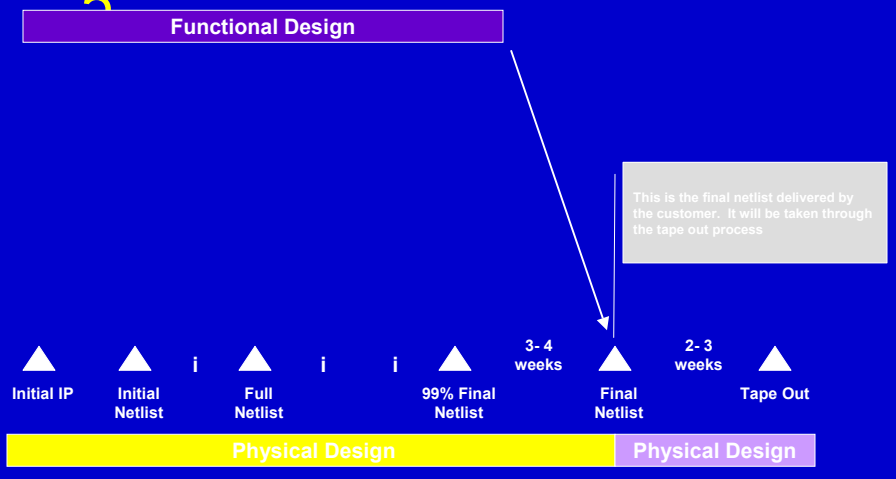
Concurrent Design Milestone 3



Concurrent Design Milestone 4



Concurrent Design Milestone 5



Large Scale Issues, "Think Physical"

- Where is this logic going to live on the chip? I.e. Have a trial floorplan in mind
- No "dirty hierarchy", I.e. keep logic out of top levels of tree. Even better generate top levels using tools
- Snoopers, etc coded in such a way that hierarchy can be changed later

“Think Physical” (#2)

- No snake paths, problematic for optimization and analysis
- Registered-in or Registered-out paradigm for the major blocks
- “Edge” or “io affinity logic” carefully partitioned to make it easer to find later

“Think Physical” (#3)

- Choose global repeater and repeater distance (e.g. BUFX20 every 2.2mm, TSMC .18u)
- Calibrate process for ‘chip crossing time’, I.e. repeater insertion delays. Know these numbers by heart
- Add default constraints for the block synthesis flow that shows a full chip crossing delay by default on the input (registered-out) and a repeater loading on every output

Small Scale Issues

- Just say NO:
  - Tri-states. Problems galore with ERC, timing checks. Rarely seen, not usually any need for them in 6+ layers of metal
  - Multi-cycle paths (complexity for timing, and a pure area saving issue)
  - Bulk use of asynchronous set/resets or latches

Small Scale Issues, #2

- Make sure Synthesis is:
  - Not using “dangerous cells” (don’t_use)
  - Not using XL (lower power) cells
  - Not using AOI gates for muxes (or with feedback for clock enabled flops or sync reset flops)
  - Not using 8:1 muxes (slow, big and congested)
  - Has slightly over-constrained clocks
  - Has a max transition run: 800ps – 1.5 ns

Small Scale Issues (#3)

- Use a “reasonable” wireload model (more on this later). Take post-synthesis timing with a grain of salt
- Bad scan methodology dangers:
  - Test_compiler paradigm assumes ALL synopsys compiles are done “test aware”
  - However, most people too cheap to buy that many licenses, and want to use it as a ‘translator’ after main synthesis is done
  - Careful or it may undo your synthesis QOR, e.g.undoes max transition fixes

Small Scale Issues (#4)

- Assembly of gate-level netlist from RTL should match simulation netlist. Bad sensitivity lists, bad include file handling, RTL not under version control, etc can cause formal verification to fail.
- Backend netlist should be checked after each build to make sure it is free of gtech cells, raw RTL, other clutter.

Small Scale Issues (#5)

- Assign statements ok inside pnr block hierarchy, but not ok at block boundary. Use synthesis to add buffers here.
- <slide that explains ports vs.. pins issues here>

Small Scale Issues (#6)

- Ram instantiation wrappers a good idea: specify logic construct (fifo, reg file), width+depth. Wrapper adds control, BIST and adds optimal physical ram object to construct the whole

Ram wrappers (#2)

- Minimize ram overhead area (decoders and sense amps)
- Make floorplanning easy by keeping each ram the same form factor if possible
- Small changes in widths or depths should keep instance names the same (give the floorplanner a break)

Ram wrappers (#3)

- Re-partitioning will change the name, so try to use leaf names and make these unique
- Round widths and depths up so that little changes in RTL don't effect the floorplanning (or require new ram builds!)

Small Scale issues

- Hand instantiation: used to for precise control of gates and/or placement. Name no longer changes. Use macros or other tricks to allow actually choice of gate type to be changed later
- Logic loops: e.g. process monitors (procmon ckts), etc must have loops opened or some timing aware tools will freak out

Choices: Tools, Machines, Flows and Languages

- Pick hardware+OS that is the “first release” platform for your newest, least stable tools. Watch for Linux.
- The tool you already know how to use can be a lot better than the shiny new tool you don’t know. Never overestimate the capabilities of a tool you haven’t used before.
- Avoid “Science Projects” and focus on what really matters. Remember another name for “Engineer” is “A person that fixes problems we don’t have”

Choices (#2)

- On the other hand, anticipate when your current tools/flow will be out of gas in a particular area.
- Beware of “estimator” tools. Push forward before too long to check them.
- Hierarchy capable solutions are many, which do you think is best?

Choices (#3)

- Understand the inherent limitations of accuracy in the tools: ex How accurate is extracted timing + primetime? How about switch windows used in coupling analysis? How about IR thermal maps in static power analysis?
- Pick your process and think carefully. .13u is no picnic. Check the availability of all IP on that process. Has it seen silicon yet or are you the guinea pig for it?

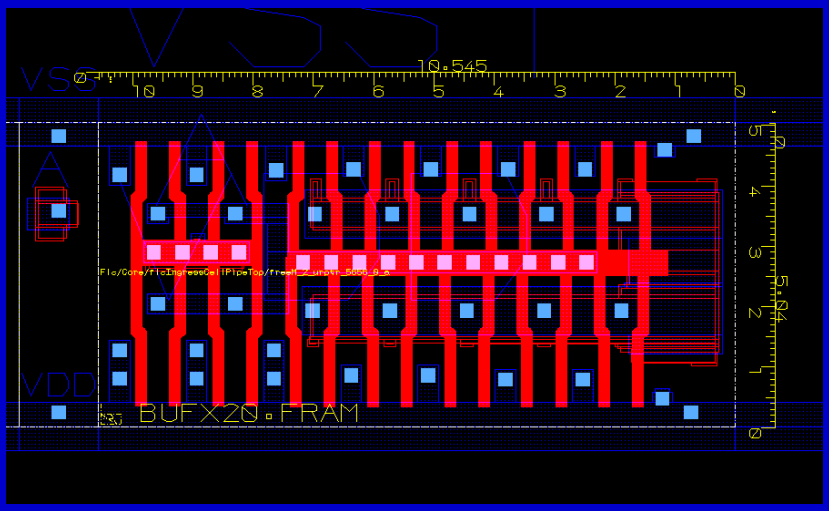
Data Prep: Netlists

- Careful with assigns, they indicate fuzzy thinking at the block level.
- 1'b0, 1'b1 and tie-high, tie-low cells.
- Bizarre name space problems:
  - Best to avoid netnames like 'input, output'. Duh.
  - Truly bizarre cases: qp, next, csh
- Mismatches with IP, case sensitivity issues
- Debus netlists, but not IP ports
- Normalize (remap) block names in backend so that downstream flow is always broken: e.g. "FarbleBlockUnit0" -> "fb0"

Data Prep: IP Types

- Stdcells
  - Power connected by abutment, placed in sea-of-rows (rarely rotated)
  - DRC clean in any combination
  - Circuit clean (I.e. no naked T-gates, no huge input capacitances)

Stdcell Example: BUFX20



Stdcells, Continued

- 8,9,10+ tracks in height
- Metal 1 only used (hopefully)
- Separate scan outputs vs.. dual rail outputs
- Setup/hold margin on scan flops determines clock skew target (ex: artisan 120ps or so)
- Strange timings (ex: setup on artisan FFs)

Stdcells, Continued

- Multi-height stdcells
- Buffers: sizes, intrinsic delay steps, optimal repeater selection, EM issues for largest?
- Special clock buffers + gates (balanced P:N)
- Special metastability hardened flops
- Cap cells (metal1 used?)
- Gap fillers (metal1 used?)
- Tie-high, tie-low

Stdcells, Continued

- Stdcells, continued
  - Spares, tied off in netlist? Or tied off internally. Added and placed after placement, or as part of incoming netlist? FIB-able? Spare nets and routes?
  - Antenna protected stdcells, .13u

Macrocells: Rams

- Artisan compiled rams example
  - Rings for power: rotated rams require new master?
  - BIST/redundancy
  - Antenna protected?
  - Reasonable drive?
  - Layer use?

Macrocells: analog

- PLLS, DLLs
- DAC, ADC
- XO, Voltage reference generators
- RAC, Serializer/Deserializers

Tool and Flow Setup

- Tech files
  - Correct process and process corners
  - Verify the extraction and library data with test structures
  - Place under revision control, you may have to fiddle with it (e.g. artisan Apollo tech file does not have via stacking turned on)

Queuers and Wrappers for tool execution

- Wrappers Allows version and patch version selection for every tool, can also help with licenses management.
- Give users correct std version for tool, but always allow special version to be used at any point in flow
- Setup queuer: lsf, openpbs, gridware. Must deal with different machine speeds/memory, OS.
- Automatic machine selection + user specified hostname

Batch mode issues with Avant!

- License use problems (variable route rules with tapering option selected suddenly need “HPO” license!? How did I know?)
- Tools should not grab all licenses by default upon startup
- Tools should spin lock by default for license if they don’t have it
- Tools should not hold license if they aren’t using it anymore

Batch mode issues with Avant! (#2)

- NullX Server issues: tools riddled with assumption that X server is present.
- Aserver process may or may not fork upon tool execution, can confuse scripts waiting for exit (e.g. make)

Flow automation

- ‘make’ a great tool, but:
  - No dependencies on program outputs directly
  - No control over depth vs. breadth first execution
- Automatic Log checking essential, but watch out for signal or instance names with ‘error’ as part of the name 8)
- Automatic command file generation

IO and Packaging

- Pin count
- Performance (electrical, thermal)
- Availability/risk
- Cost

Packaging Styles, Level 1

- Level 1 (chip to package)
  - Wirebond
    - Padring –> linear or staggered
    - Right-angle or radial bonding
  - Flipchip (BGA)
    - Padring – peripheral/perimeter vs. area array
    - Eutectic or high temp (C4) solder bumps
    - Under fill for thermal stress management

Packaging Styles, Level 2

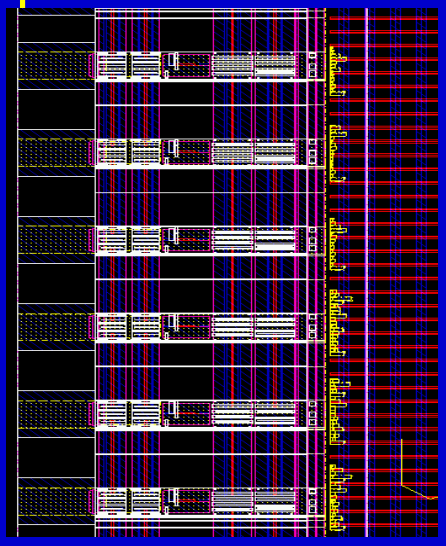
- Level 2 (package to board)
  - Leaded (e.g QFP)
    - Low pin count (typically < 304)
    - Low performance (plastic, inductive) 2.5 – 3 Watts max
    - Low Cost
    - Fast turnaround

Packaging Styles, Level 2

- Level 2, continued
  - Leadless (e.g. BGA , CSP [Chip Scale package], CGA [Column Grid Array])
    - High pinout (> 2000)
    - High performance (planes, matched impedence, low Inductance)
    - Higher cost (1.5 to 10 cents/pin, 1 cent/pin the “holy grail”
    - Longer Lead Time (10-14 weeks once in the queue!)
      - One customer pkg cost == 5 X silicon cost

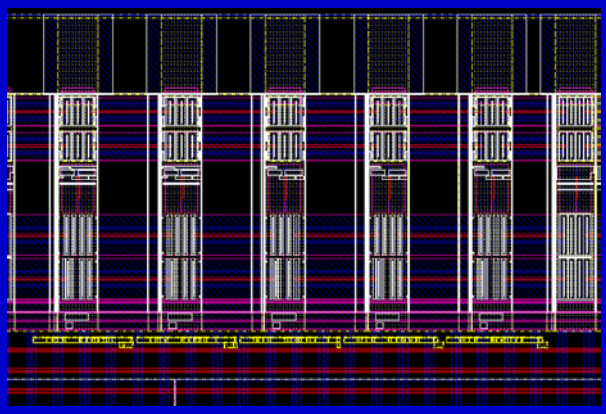
Padring Shot 1

- vertical



Padring Shot 2

horizontal



Packaging “Gotchas”

- Cavity packages have min and max die size limits
- Flip has no diearea wiggle room once package build started
- Bumps vs. Balls: Bump to Ball ration is NOT 1:1

Packaging Gotchas (#2)

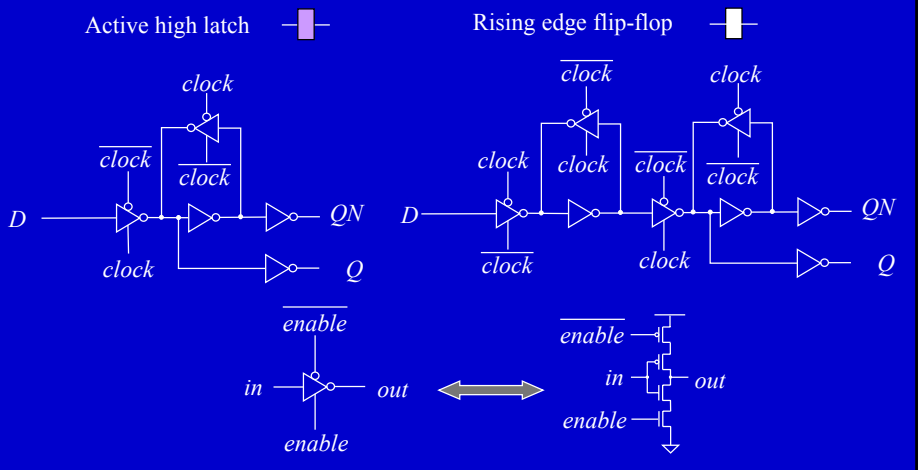
- Material limits exist: the coefficient of thermal expansion limits the body size options of some packages (e.g. ceramic)
- Routability is limited by via pitch and file line technology. I.e. going to bump pitch < 200um and ball pitch < 1mm doesn't make sense right now

IO and Package

- Power hookup
  - Rings, slotting
  - EM issues, double bonding
  - Unusual LVS issues

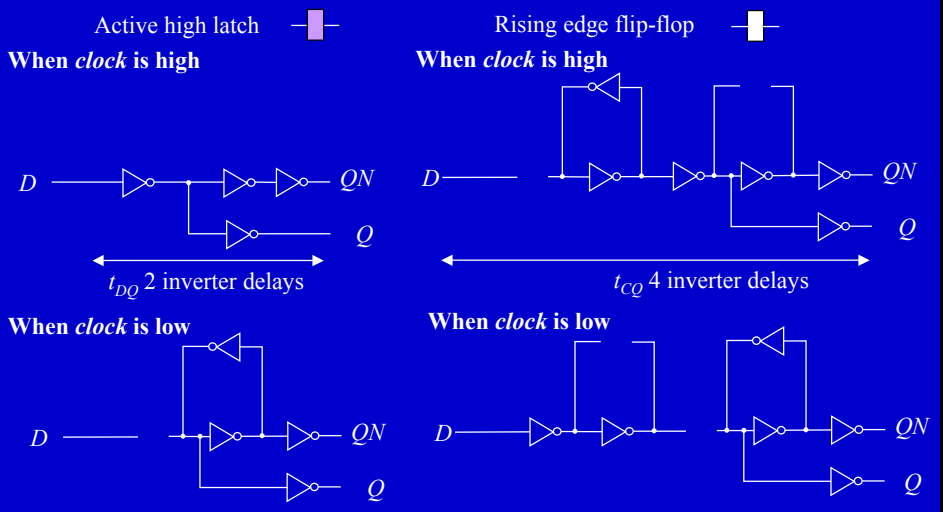
Flip-Flop vs. Latch Timing

Latch and Flip-Flop Gates

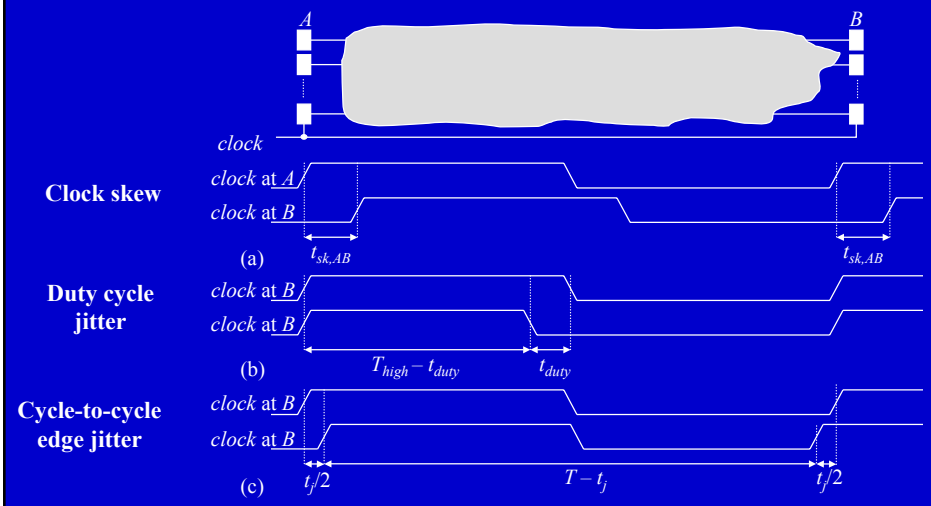


Latch and flip-flop schematics from TSMC 0.13um LV Artisan Sage-X Standard Cell Library.

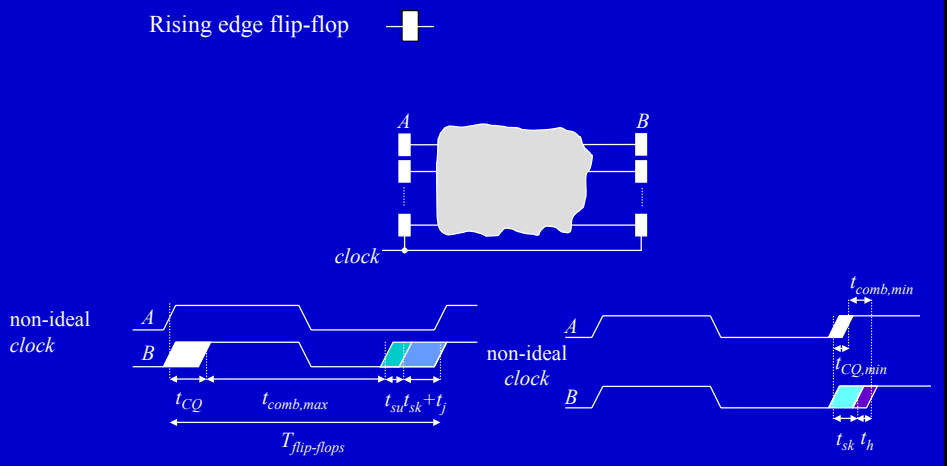
Latch and Flip-Flop Behavior



Clock Characteristics



Flip-Flop Timing Characteristics



Latch Setup Time and Transparency