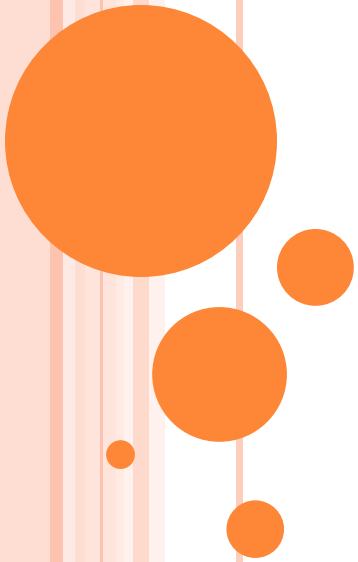

SoC Design Issues

System-Level Design Issues

Key Aspects of Design Reuse

- Fundamentals
  - **Well-designed IP** is the key to successful SOC design
- System level design guidelines
  - To produce well-designed IP
  - To integrate well-designed IP to an SOC design
  - Driven by the needs of IP integrator and chip designer
- Principles behind these guidelines
  - **Discipline**
    - Consistent good practices
  - **Simplicity**
    - The simpler the design, the easier to fix the bugs
  - **Locality**
    - Make timing and verification problem local by careful block and interface design

Full Custom Design in Reuse

- Full custom design
  - Design that are not from synthesis
- Major problems
  - Performance gain is limited
  - Non-portable, hard to modify designs
  - Redesign take time
- **Limit** full custom design for only small part of design
  - Even aggressive processor designer uses full custom only for data path

Interface and Timing Closure

- Timing problems due to deep submicron process
  - Dominated wire delay
  - Imprecise wireload model due to uncertainty of wire delays
- Solution
  - Tools
    - Timing driven P&R, Physical synthesis
  - **Tactics** for fundamental good design
    - Register all inputs/outputs of the macro
      - Unit for floorplan
    - Register all outputs of the subblock of macro
      - Unit for synthesis
    - Exception
      - Cache interface
      - Design likes PCI interface that needs glue logic at the interface

SYNCHRONOUS V.S. ASYNCHRONOUS

- Synchronous
  - Avoid asynchronous and multi-cycle paths
  - Tools work best for synchronous design
    - Accelerate synthesis and simulation
  - Ease static timing analysis
- Register based
  - Use (positive) edge triggered DFF
  - Latches shall be used only in small memory or FIFOs

CLOCKING

- Clock planning
  - Minimize the number of clock domains
  - Isolate the interface between clock domains
  - Careful synchronizer design to avoid metastability
  - Isolate clock generation and control logic
- Document the clock scheme
  - Required clock frequencies and PLL
  - Interface timing requirements to other parts of the system
- PLL
  - Disabling/bypassing scheme
  - Ease testing
- For hard blocks
  - Eliminate the clock delay using a PLL
  - Balance the clock insertion delay

RESET

- Synchronous reset
  - Easy to synthesize
  - Requires a free-running clock
- Asynchronous reset
  - Do not require a free-running clock
  - Not affect flip-flop data timing due to separated input
  - Harder to implement, like clock, CTS is required
  - Synchronous de-assertion problem
  - Make STA and cycle-based simulation more difficult

- **Asynchronous reset** is preferred

Internal Generated Reset

- Internal generated reset causes unwanted reset during scan shift
- Solution
  - Force internal generated reset signal **inactive** during test