Timing Verification

Ken Stevens

Timing Verification

- Understanding timing is required for quality design
- Good design is *all about timing*
- Optimizing timing also optimizes power
- Most failures in a circuits are timing related

Timing Verification

- Key quality metric and target in all designs
- Directly affects:
  - Cost
  - NRE
  - Power
  - Area
  - Technology node
  - Circuit styles
  - Yield and speed bin
  - Portability and scalability
  - Architecture
  - CAD requirements
- Consider adder and multiplier designs

Synchronous Timing

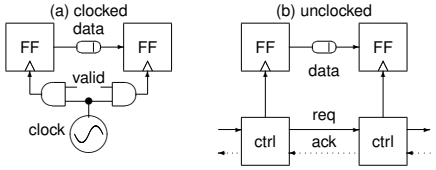
- Timing specification of synchronous system
  - Global in nature
  - Must be considered from the start of a design
  - Can enable concurrent, efficient design
  - In practice is quite wasteful
    - Performance limited by slowest path in *entire* system
    - Significant margin must be taken
      - PVT: process, temperature, voltage variations
        (VT for bundled data asynchronous design)
    - All blocks have natural frequency
    - Seldom is this the same system-wide
- Timing constraints are normally **speed paths**
  - Correctness relative to clock *frequency*

Asynchronous Timing

- Timing is local to each element
  - Interface with *handshake protocol*
  - No global timing constraints
  - Each module can be optimized for power/performance/area
  - PVT: runs at optimal speed for current values
- Protocol encodes data validity
- Asynchronous design is therefore *sequential*
- Delay-insensitive (DI) asynchronous design has *no* timing dependencies
- Timing implemented as **races**
  - Correctness relative to relative silicon delay *paths*

Speed Path vs Race

- Speed paths constrain a signal path against a frequency (clock)
  - Clock can be slowed down until all speed paths are slower than the clock frequency
- Races are order constrained signal paths
  - Failure of a speed path results in yield loss



Timing Yield

- Conservatism in timing margins:
  - $\Longrightarrow$ increased yield
  - $\Longrightarrow$ decreased performance
- Speed Bins
  - Speed path yield loss creates usable parts that run in slower speed bins and sell for less $$$
  - Race yield loss results in parts that must be discarded
    - not reliable at all frequency/temperature/process ranges
- Is extra performance gained worth low yields?

Timing Yield

- Conservatism in timing margins:
  - $\Longrightarrow$ increased yield
  - $\Longrightarrow$ decreased performance
- Speed Bins
  - Speed path yield loss creates usable parts that run in slower speed bins and sell for less $$$
  - Race yield loss results in parts that must be discarded
    - not reliable at all frequency/temperature/process ranges
- Is extra performance gained worth low yields?
  Depends on the market!

Goals of Performance Verification

- Must operate at desired frequency
  - . . . or at desired frequency ranges
- Must operate under specified range of operating conditions
- Must meet yield goals

Current State of the Art

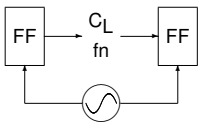
- Tools support max-delay speed paths between flops
- Tools support some latch based design
- Little or custom support for races
  - Min-delay races
  - Asynchronous design
- Little support for non-traditional timing approaches
  - Domino Logic
  - Pass gate logic
  - Sense Amps . . .

Speed Path Analysis

- Verify worst case maximum path delays are less than frequency
- $C_{ff+} \geq \max(2T_{pw}, (T_{lc+} + T_{cc+} + T_{su} + T_{skj} + T_{cq+} + T_{cad}))$
  - T_{pw}: minimum sized pulse that can be safely engineered and propagated
  - T_{lc+}: max logic delay
  - T_{cc+}: max wire & communication delay
  - T_{su}: setup time
  - T_{skj}: clock skew and jitter
  - T_{cq+}: clock to data delay
  - T_{cad}: timing CAD inaccuracies

Speed Path Analysis

- Verify worst case path delays are less than frequency
- $C_{ff+} \geq \max(2T_{pw}, (T_{lc+} + T_{cc+} + T_{su} + T_{skj} + T_{cq+} + T_{cad}))$

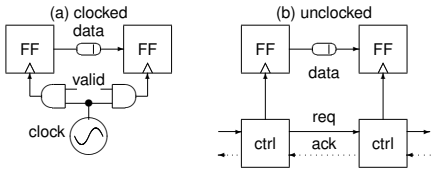


Skew and Jitter

- **Skew**
  - Differences in delay of the clock signal due to clock network delays
- **Jitter**
  - Variation in duty cycle and delay of the clock waveform primarily due to noise and voltage variations

Max Delay Race Analysis

- Max delay races compare:
  1. The maximum delay of the early path
  2. The minimum delay of the late path



Min-Delay Race

- *Hold time* is a min-delay race requirement
- The race is between the clock and data
  1. Early path: the same clock edge at different latches
     - Take maximum clock skew and jitter
  2. Late path: data racing between latches (min-delay)
     - Minimum clock-to-data-to-flop delay
     - Could just be a wire, so only T_{cq-}
- Fixed by padding the late paths

Static vs Dynamic Timing Analysis

- A classic run-time vs accuracy tradeoff
- Static timing analysis is vectorless
  - Finds worst-case conditions at each gate
  - False path problem (reconvergence and correlations)
  - Fast run-times
  - Problem of summing of worst-case conditions...

Static vs Dynamic Timing Analysis

- A classic run-time vs accuracy tradeoff
- Static timing analysis is vectorless
  - Finds worst-case conditions at each gate
  - False path problem (reconvergence and correlations)
  - Fast run-times
  - Problem of summing of worst-case conditions...
- Dynamic timing analysis consists of vector-based simulation
  - Most accurate
  - Most time consuming
  - Does this cover all cases?

Static vs Dynamic Timing Analysis

- A classic run-time vs accuracy tradeoff
- Static timing analysis is vectorless
  - Finds worst-case conditions at each gate
  - False path problem (reconvergence and correlations)
  - Fast run-times
  - Problem of summing of worst-case conditions...
- Dynamic timing analysis consists of vector-based simulation
  - Most accurate
  - Most time consuming
  - Does this cover all cases?
    NO!!!!

Static Timing Analysis

1. Input signal arrival time and slope
2. Output load
3. Gate delay approximated based on pin to pin slope and load
4. Special case clock external timing and variation
5. Propagate internal clock logic with gating
6. Propagate data logic, using false path algorithms
7. Verify early and late path conditions into synchronization points (these are flops for standard static timing analysis)

Static Max-Delay Path Modeling

- These are factors evaluated for max-path calculation
  - Worst-case gate delay calculated
    - Uses slowest DCN (diffusion connected network) to rail
    - How similar are DCN's in a circuit? e.g.: a+bc
  - Worst case skewed simulation corner used:
    - Three "corners": t: typical, f: fast, s: slow
    - Corners consist of: p-type, n-type, voltage, temperature
    - Specified as tttt, ffff, ssss, ffss, etc.
  - **MCF** (maximum coupling factors) and wire R & C models are maximized
    - Is cross talk always a bad effect (slowing circuit performance?)
    - Victim and aggressor relationship

Static Max-Delay Path Modeling

- These are factors evaluated for max-path calculation
  - Branching loads reduce load shielding of resistors
    - What type of model best?
    - RC, π, ... (we'll cover these)
  - Maximum parasitic voltages
  - Maximum miller coupling
    - C_{gs} and C_{gd}
    - Significant noise source on static and dynamic nodes
    - Can couple signals above V_{dd} and below Gnd?

Static Max-Delay Path Modeling

- These are factors evaluated for max-path calculation
  - Branching loads reduce load shielding of resistors
    - What type of model best?
    - RC, π, ... (we'll cover these)
  - Maximum parasitic voltages
  - Maximum miller coupling
    - C_{gs} and C_{gd}
    - Significant noise source on static and dynamic nodes
    - Can couple signals above V_{dd} and below Gnd? yes!
- Is this too pessimistic?

Static Max-Delay Path Modeling

- These are factors evaluated for max-path calculation
  - Branching loads reduce load shielding of resistors
    - What type of model best?
    - RC, π, ... (we'll cover these)
  - Maximum parasitic voltages
  - Maximum miller coupling
    - C_{gs} and C_{gd}
    - Significant noise source on static and dynamic nodes
    - Can couple signals above V_{dd} and below Gnd? yes!
- Is this too pessimistic?
  YES for deep submicron, but also depends on models

Static Min-Delay Path Modeling

- The following factors and conditions are used:
  - Fastest switching path in the DCN (diffusion connected network) is used
  - Best case process corners are used
  - MCF is minimized
  - Lumped RC model used
  - Parasitics are minimized

Process Variation and Static Timing Analysis

- Wost case models do poor job
  - Long paths result in *statistical averaging*
  - Short paths are likely optimistic
    - Side-by-side inverters can differ by factor of 2
- **OCV** (on chip variation) models used to *derate* designs to remove pessimism
- **AOCV** (Advanced OCV) models add methodlogy and process: (28nm)
  - Traditional Table based appraoch
  - Path logic depth
  - Physical distance
- **POCV** (Parametric OCV) models adds: (14nm node)
  - Delay function parameterized to variable specific to instance
  - Some statistical derating for random variations

Clocking Strategies

- Two-phase non-overlapping clocks
  - Can be used for latch based design
  - Mitigates hold-time
  - Reduces edge rate requirements
  - Big penalty at high frequency
- Single clock
  - Nominal 50/50 duty cycle
  - Rapid edge requirements
  - Gated or shaped locally

Clock Uncertainty

- Skew and jitter
- Reduces usable cycle time - pure overhead
  - $C_{ff+} \geq \max(2T_{pw}, (T_{lc+} + T_{cc+} + T_{su} + T_{skj} + T_{cq+} + T_{cad}))$
- Part of all timing analysis
  - E.g.: from timing.dc in lab2: `clk_skew = 0.1`

Clock Uncertainty

- Sources of clock skew and jitter
  - Slope combined with V_{th} variations
  - Cross coupling
  - Interconnect RC variation
  - Data-dependent loads
  - Power and ground noise
  - Temperature (thermal) variations
  - Within-die process variation
- Skew is typically spatially dependent
  - Larger for cross-chip signal communication

Clock Uncertainty

- Clock propagation direction
  - Same direction as data
    - Hold times harder to meet (min-delay race)
  - Opposite direction as data
    - Setup times harder to meet (speed-path)

Clock Deskew Strategies

- Topological
  - H-Tree
  - Spine
- Parametric
  - Widen wires
  - Sharpen slopes
    - Reduces threshold skew
    - . . . but: increases power supply noise

Clock Deskew Strategies

- Passive
  - Hierarchical short circuiting
- Active
  - Deskew logic (Intel)
    - Sample offset of neighborhood clocks in grid
    - Skew local clock to reduce skew
    - Changes skew results of neighbors
    - Potential instability
- All Deskew strategies cost significant area and power
- 40-60% of active power in high-speed circuits is clock related
- Is this a significant potential benefit for asynchronous design?

Multiple Frequency Designs

- Synchronous:
  - "Double pumped" ALUs
  - Base frequency with multiplication
  - Synchronizers
- Asynchronous
  - Desirable, most efficient implementation
  - Normal dynamic interactions and interfaces
  - Optimize power/performance and pipelines
- Hybrid
  - GALS (Globally Asynchronous Locally Synchronous)
  - Synchronizers

Multiple Frequency Domains

- Clocks work best in homogeneous, single frequency domain
- Modern designs have multiple frequencies
  - Multi-frequency becoming more prevalent
  - CAD Tools don't directly support this yet...
  - SoC
  - Hard/soft IP
  - Power constraints

Clock Gating

- Mimic asynchronous design with data validity information
- Reduce clock and data energy for invalid tokens
- However:
  - Increases clock skew and latency
    - Exacerbates hold time (min-delay races)
    - Different logic (buffers vs NAND/NOR gates):
      - Data-dependent capacitative loads
      - Miller and back-gate coupling
      - Different reactions to process skews
    - Need to pulse shape signals

Clock Gating

- Mimic asynchronous design with data validity information
- Reduce clock and data energy for invalid tokens
- However:
  - **First Droop**
    - Significantly changes dI/dt
    - Result in sustained drops of up to 200mV or more
    - Significantly higher demand for decoupling caps

Clocked Design Styles

- Flop design (edge triggered)
  - Simple to analyze
  - Cannot hide clock uncertainty
- Latch design
  - Can eliminate clock uncertainty
  - Average logic between flops must be one phase
  - **Time borrowing**
    - Each phase can "borrow" up to one more clock phase of time from adjacent latches
    - Gives added flexibility in synthesis and design partitioning
  - Very difficult to validate multi-cycle latch constraints
- Async designs are latch based

Clocked Design Styles

- Maximum clocked flop cycle time:
  $C_{ff+} \geq \max(2T_{pw}, (T_{lc+} + T_{cc+} + T_{su} + T_{skj} + T_{cq+} + T_{cad}))$
- Maximum clocked latch cycle time:
  $C_{l+} \geq \max(2T_{pw}, (T_{lc+} + T_{cc+} + 2T_{dql} + T_{cad}))$
  - T_{dql}: data-to-output delay of a latch.

Other Logic Styles

- Efficient STA base timing flows work well for:
  - Combinational blocks between flops
- . . . sometimes work automatically for
  - Time borrowing latch design
- . . . and doesn't work for
  - Domino circuits
  - Pass gate logic
  - Asynchronous circuits
  - Memory arrays . . .
- How much async logic are there in our μ-processors today?
  Over 50%!

Other Logic Styles

- Custom design flows required:
- STA can be used for glue logic
- Spice simulation or CBD (Cell Based Design) models used for other logic
- Memory Blocks
  - RAMs are typically self-timed (asynchronous)
    - Word line, column line (request)
    - Address (data)
    - Dual-rail data indicating validity

Memory Blocks

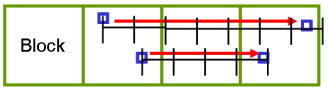
- Well characterized to operate in clocked domain
  - Worst-case cycle times
  - Full handshaking removed by timing
  - Cell to bit line cap ratios, write and read
  - Known differential generation time
  - Sense-amp pulse-clocked
  - Precharge constraints
  - Word line line latency
  - Decode latencies
  - Data latencies
  - Setup and hold times

Full-Chip Timing

- Complicated problem of prediction, implementation, and validation
- Area, Placement, and design of cells directly influences pipelining and architecture.
- Estimation of physicals (placement, speed, caps, etc.) critical

Full-Chip Timing

- Errors discovered late in the game are *very* costly
  - This is a *global* constraint for clocked designs
  - Significant potential benefit for asynchronous interconnect
    - Can repipeline without correctness problems
    - This also improves link bandwidth
- Wire delays scale differently than transistor delays



Full-Chip Timing

- Timing budget determined
- Interface and wire delays, flopped repeaters
- Blocks locally designed based on budgets on schematics
  - Parasitic estimation
- Timing validated on blocks after APR and parasitics
- Timing interfaces defined and simulated (required times, etc.)
- Freeze RTL, STA, negative slack analysis
- Global routing, repeater and DFT (Design for Test) insertion
- Full-chip timing integration

Lots of Interesting CAD

- Current tools good at:
  - Path tracing, propagating timing, analyzing slacks
- Humans currently better at:
  - False path recognition
  - Off-path load models
  - Custom circuits and timing constraints
  - Finding and modeling hazards and glitches

Opportunities

- *Relative Timing* to find and evaluate race paths
- Joint timing and placement and routing
- Multiple clock domain circuit design and validation
- Second order effects such as MIS (Multiple Input Switching)
- Support for custom and asynchronous circuits
- Improving accuracy of path delays

"improvements . . . being pursued by CAD and design engineers today . . . will, to a large degree, determine how long Moore's law will hold true in the years ahead."

Victor Peng

Timing Review

1. Relation of design quality metrics to timing
2. Synchronous timing characteristics, benefits and demerits
3. Asynchronous timing characteristics, benefits and demerits
4. clocking strategies
5. Skew and Jitter causes, effects, magnitudes
6. Clock deskew strategies (topological, parametric, passive, active)
7. Speed Path
8. Race

Timing Review

1. Timing yield and speed bins
2. General goals of PV (performance Verification)
3. Factors in flopped speed path analysis
4. Race Analysis: min-delay and max-delay
5. Static vs dynamic timing analysis
6. Process corners
7. MCF modeling

Timing Review

1. Timing in multi-frequency designs
2. Clock gating
3. First Droop
4. Latch "time borrowing"
5. Current STA (static timing analysis) application areas
6. OCV, AOCV, POCV
7. Analysis of non-STA supported designs and blocks
8. Full-chip timing