**Timing Design in Digital Systems**

Dr. Paul D. Franzon

Outline
1. Timing design in Synchronous (clocked) Logic
   - Min/Max timing with flip-flops
   - Latch-based design
2. Timing Issues in CMOS circuits
3. Timing verification and SDF

References
2. Smith, Sections 2.4, 13.6, 13.7, 16.4.1
Attachments: Need sample library sheets.
General Approach to Timing Design

- In general, all signals start and end in registers every clock period

- In general, there is only one clock (in any one sub-block of the chip) and each flip-flop is clocked every cycle
  - If a chip needs multiple clocks, they should share a common root clock (e.g. divide a master clock) and be fed to different sub-designs within the chip
Clock Level Timing

- Example (Revision):

![Clock Level Timing Diagram]

- Example (Revision):

![Clock Level Timing Diagram]
**Critical Path**

- Thus, the clock speed is determined by the slowest feasible path between registers in the design
  - Often referred to as “the critical path”

![Critical Path Diagram]

Critical path is longer with increased logic depth (\# gates in series)
**Clock Distribution**

- The goal of clock tree is for the clock to arrive at every leaf node at the same time:

- Usually designed after synthesis: Matched buffers; matched capacitance loads
- Some clock skew (difference in arrival times) is unavoidable
  - Clock skew specification is an important parameter
  - Clock tree carefully designed to meet a skew requirement
- Clock jitter (cycle to cycle skew variations) is also important but is treated as being lumped in with the fixed clock skew to make a total skew (+jitter) budget
  - Comes from phase noise in Phase Lock Loop, etc.
Flip-Flop based design

Edge triggered D-flip-flop
Q becomes D after clock edge

Set-up time:
Data can not change no later than this point before the clock edge.

Hold time:
Data can not change during this time after the clock edge.

$t_{clock-Q}$
Delay on output (Q) changing from positive clock edge
Preventing Set-Up Violations

Set-up violation:
Logic is too slow for the correct logic value to arrive at the inputs to the register on the right before one set-up time before the clock edge

Constraint to prevent this:

\[ t_{\text{clock}} > t_{\text{clock-Q}} + t_{\text{logic-max}} + t_{\text{set-up}} + t_{\text{skew}} \]

- The amount of time required to turn ‘<’ into ‘=’ is referred to as **timing slack**

©2004 Dr. Paul D. Franzon, www.ece.ncsu.edu/erl/faculty/paulf.html
Preventing hold violations

Hold violations occur when race-through is possible

Constraint to prevent hold violations:

\[ t_{hold} + t_{skew} < t_{clock-Q} - \text{min} + t_{logic-min} \]

- \( clock \)
- \( clock' \)
- \( t_{clock-Q} \)
- \( Q1 \)
- \( D2 \)
- \( Q2 \)

*sometimes have to insert additional logic to prevent hold violations*
**Latch Based Design**

**D-latch**
- Q follows D while clock is high
- Value on D when clock goes low is stored on Q

**Set-up and hold times:**
D can not change close to the falling (‘latching’) clock edge.

\[ t_{\text{hold}} - t_{\text{setup}} \]

Delay from clock going high to Q changing
**Latch timing constraints**

To prevent set-up violations:

\[ t_{clock} + t_{clock\text{-}high\text{-}max} > t_{clock\text{-}Q\text{-}max} + t_{logic\text{-}max} + t_{set\text{-}up} + t_{skew} \]

**Notes:**
- The percentage of time the clock is high is referred to as the **duty-cycle**
- If part of the following clock-high time is used to allow this logic to be slower, then the logic-block connected to Q2 must be proportionally faster
- Using the clock-high time like this is called **cycle-stealing**
Latch Set Up Violations

Notes:
• The percentage of time the clock is high is referred to as the duty-cycle

• If part of the following clock-high time is used to allow this logic to be slower, then the logic-block connected to Q2 must be proportionally faster

• Using the clock-high time like this is called cycle-stealing

• Normally cycle stealing is not enabled
Latches ... Cycle Stealing

![Diagram of latches with cycle stealing](image)

**Figure 11-9:** Timing considerations in D-latch design.
...Latch timing constraints

To prevent hold violations:

\[
t_{\text{clock-high-max}} + t_{\text{hold}} + t_{\text{skew}} < t_{\text{clock-Q-min}} + t_{\text{logic-min}}
\]

Note:
- Hold violations are harder to prevent in latch-based designs
Example:

<table>
<thead>
<tr>
<th></th>
<th>min : typ : max</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T_{clock-Q}$</td>
<td>3 : 4 : 5</td>
</tr>
<tr>
<td>$TNOR$</td>
<td>1 : 2 : 3</td>
</tr>
<tr>
<td>$T_{su_{max}}$</td>
<td>1</td>
</tr>
<tr>
<td>$T_{hold_{max}}$</td>
<td>2</td>
</tr>
<tr>
<td>$T_{skew}$</td>
<td>1 ns</td>
</tr>
</tbody>
</table>

If this is the critical path, what is the fastest clock frequency?
Is there potential for a hold violation?
**CMOS Drive Strength**

Revision: CMOS transistors operating in the linear region:

\[ I_{ds} = \beta ((V_{gs} - V_t)V_{ds} - V_{ds}^2 / 2 \]

where \( \beta = (\mu e t_{ox})(W / L) \)

where \( W \) is the transistor width, and \( L \) is the channel length

i.e. To a first approximation,

\[ I_{ds} \approx V_{ds} / R_{on} \]

\[ R_{on} \approx 1 / \beta (V_{GS} - V_t) \]

Thus, delay in CMOS circuits depends largely on \( W/L \) of the drive transistor and the capacitance of the load it is driving.

That capacitance consists of:

- Input gates of cells being driven, and
- Capacitance of wiring

\[ \tau = R_{on} C_{load} \]
**Data Sheet Example**

- Using timing approximations in the datasheet, what is the maximum clock frequency for this circuit (ignore wire load, Tskew):

  ![D-Flip-Flop Circuit Diagram](image)

  **T\_clock-Q:**
  
  \[ T_{cp-Q\_max} = \max (1.12 + 1.59 \times CL, 1.09 + 1.32 \times CL) \]
  
  \[ = \max (1.12 + 1.59 \times 0.0488, 1.09 + 1.32 \times 0.0488) \]
  
  \[ = \max (1.2, 1.15) = 1.2 \text{ ns} \]

  **Q -> A2:**

  \[ T_{cp-Q\_max} = 1.2 \text{ ns} \]

  **Q -> A1:**

  \[ CL_{max} = 0.0371 + 0.0117 \text{ pF} = 0.0488 \text{ pF} \]

  \[ T_{cp-Q\_max} = \max (1.12 + 1.59 \times CL, 1.09 + 1.32 \times CL) \]
  
  \[ = \max (1.12 + 1.59 \times 0.0488, 1.09 + 1.32 \times 0.0488) \]
  
  \[ = \max (1.2, 1.15) = 1.2 \text{ ns} \]
### Example

**Tlogic:**

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>Tlogic</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0.57 ns</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0.57 ns</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0.61 ns</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0.61 ns</td>
</tr>
</tbody>
</table>

**CL:**

<table>
<thead>
<tr>
<th>Pin Capacitance (pF)</th>
<th>NOR2</th>
</tr>
</thead>
<tbody>
<tr>
<td>a0</td>
<td>10.5</td>
</tr>
<tr>
<td>a1</td>
<td>13.9</td>
</tr>
<tr>
<td>y</td>
<td>10.1</td>
</tr>
</tbody>
</table>

**DFF**

<table>
<thead>
<tr>
<th>Pin Capacitance (pF)</th>
<th>NOR2</th>
</tr>
</thead>
<tbody>
<tr>
<td>a0</td>
<td>10.5</td>
</tr>
<tr>
<td>a1</td>
<td>13.9</td>
</tr>
<tr>
<td>y</td>
<td>10.1</td>
</tr>
</tbody>
</table>

**CL = 0.0136 + 0.0184 = 0.032 pF**

From A0: \( T_{\text{logic}} = \max \left( 0.503 + 3.27 \times 0.032, 0.371 + 3.21 \times 0.032 \right) \)

= 0.61 ns

From A1: \( T_{\text{logic}} = \max \left( 0.420 + 2.17 \times 0.032, 0.505 + 2.12 \times 0.032 \right) \)

= 0.57 ns
Example

\[ T_{\text{clock}} > T_{\text{clock-Q_max}} + T_{\text{logic_max}} + T_{\text{su_max}} \]
\[ = 1.2 + 0.61 + 1.4 \]
\[ = 3.21 \text{ ns} \]

\[ F_{\text{clock}} < 311 \text{ MHz} \]
Timing in Design

- The ASIC vendor will develop delay models for each cell
- Best/worst models developed as transistor $\beta$ changes with process, temperature, etc.
- Models also developed for delay as a function of capacitive load (fan-out)
- Timing models are stored as part of the cell library
- **Timing Verification is done with a combination of the following:**
  - Logic simulation using actual delays from the timing library
    - Dynamic timing analysis
  - **Static timing analysis:**
    - A static timing analyzer traces through the actual logic and calculates slowest and fastest delay for each true path through the logic
      - Sometimes, however, it can not tell the difference between true and false (not possible) paths
        - Often the designer has to identify false paths
    - It outputs a timing report for critical (Slowest) and other paths
Timing in ASIC Design Flow

Standard Delay Format (SDF) files:
- Tools communicate timing information using SDF files
  - Specifies timing information as calculated from:
    - ASIC vendor timing models (Synopsys outputs as SDF file based on this information)
    - Wiring delay calculations after place and route
- The SDF file is then used by the logic simulator or static timing verifier to verify that the design will not have any setup or hold violations

Timing Closure
- Ensuring no timing problems after “Place and Route”
  - More than simple back-annotation needed
- Becoming more difficult with increasing clock frequency and finer lithography
  - Interconnect RC delay becoming large compared with logic delay
  - Current synthesis tools are “interconnect blind”
  - New classes of tools emerging (Physical compilers)
  - New planning tools needed in the future
Good and Bad Timing Design Practices

- Good General Practices:
  - In each module, use only one clock and one edge.
  - Pay attention to the amount of logic specified in long logic paths during design.
  - Only use the globally distributed clock(s).

- Bad Design Practices:
  - Feeding back combinational logic on itself
    - Specifying a latch!
  - Using multiple clocks, multiple edges
  - Gating the clock or generating local clocks
**Exercise**

- With a flip-flop based design, what is the minimum clock period?
- Is there potential for a hold violation?
- If $t_{\text{clock\_high}}$ is 50% of the clock period ("50% duty cycle"), and cycle stealing is enabled, what is potentially the fastest possible latch-based design?

D1 | Q1 | D2 | Q2
---|---|---|---
Comb Logic | 6-7 ns | 6-7 ns | Skew = 1 ns

$T_{\text{clock\_Q}} = 1-2$ ns
$T_{\text{s}} = 1$ ns
$T_{\text{hold}} = 2$ ns

Flip Flop:

$T_{\text{clock}} > T_{\text{clock\_Q\_max}} + T_{\text{logic\_max}} + T_{\text{s}} + T_{\text{skew}}$

$= 2 + 7 + 1 + 1 = 11$ ns

90 MHz

Is $T_{\text{hold}} + T_{\text{skew}} < T_{\text{clock\_Q\_min}} + T_{\text{logic\_min}}$?

$2 + 1 < 1 + 6$ No Hold violation

Latch:

$T_{\text{clock}} > T_{\text{clock\_Q\_max}} + T_{\text{logic\_max}} + T_{\text{s}} + T_{\text{skew}}$

$> 11$

$T_{\text{clock}} > 7.3$ ns

136 MHz
Summary

- What determines the maximum clock frequency?

- What is a hold violation?

- Why do we prefer flip-flop designs over using latches?

- What tool is used to check timing at design closure?
**Remember**

- Methodology for purposes of ECE 520
  - If at all possible one-edge of one clock
  - If you need multiple clocks, they must have a common root and be related by factors of 2
    - E.g. Root clock : Tclock = 5 ns
    - This is OK:
      - Tclock, Tclock10, Tclock20
      - Tclock10 = 10 ns (Tclock*2)
      - Tclock20 = 20 ns (Tclock*4)
    - This is NOT OK
      - Tclock, Tclock15, Tclock17
      - Tclock15 = 15 ns (Tclock*3)
      - Tclock17 = 17 ns (?)