EE141- Spring 2003
Lecture 23
Timing and Clocks

Announcements

- Hardware lab this week
- Project-2 will be launched next week
Today’s Lecture

- Sequential Circuits (Cont.)
- Timing

Sequential Logic
Sequential Circuits (Cont.)

- Schmitt Trigger
- Monostable Multivibrators
- Astable Multivibrators

Schmitt Trigger

- VTC with hysteresis
- Restores signal slopes
Noise Suppression using a Schmitt Trigger

\[ V_{in} \]

\[ V_{M+} \]

\[ t_0 \]

\[ V_{M-} \]

\[ V_{out} \]

\[ t_0 + t_p \]

CMOS Schmitt Trigger

Moves switching threshold of the first inverter
Schmitt Trigger
Simulated VTC

Voltage-transfer characteristics with hysteresis.

The effect of varying the width of the PMOS device $M_4$. \(W(M_4) = k \times 0.5 \mu m\)

CMOS Schmitt Trigger (2)
Multivibrator Circuits

Bistable Multivibrator
flip-flop, Schmitt Trigger

Monostable Multivibrator
one-shot

Astable Multivibrator
oscillator

Transition-Triggered Monostable

Delay element controls the duration of the pulse.
Monostable Trigger (RC-based)

Trigger circuit

Waveforms

Astable Multivibrators (Oscillators)

simulated response of a 5-stage oscillator
Voltage Controlled Oscillator (VCO)

Current starved inverter

Schmitt Trigger restores signal slopes

propagation delay as a function of control voltage

Timing
Outline

- **Timing parameters**
- Clock nonidealities (skew and jitter)
- Impact of Clk skew on timing
- Impact of Clk jitter on timing
- Flip-flop- vs. Latch-based timing
- Clock distribution

---

Datapath and Timing Parameters

R1 and R2 can be latches or flip-flops
Latch Parameters

Delays can be different for rising and falling data transitions

Flip-Flop Parameters

Delays can be different for rising and falling data transitions
Timing Constraints
(Cycle Time and Race Margin)

Cycle time: $T_{\text{Clk}} > t_{c-q} + t_{\text{logic}} + t_{su}$
Race margin: $t_{\text{hold}} < t_{c-q,cd} + t_{\text{logic,cd}}$

Outline

- Timing parameters
- Clock nonidealities (skew and jitter)
- Impact of Clk skew on timing
- Impact of Clk jitter on timing
- Flip-flop- vs. Latch-based timing
- Clock distribution
Clock Nonidealities

- **Clock skew**
  - Spatial variation in temporally equivalent clock edges; deterministic + random, $t_{SK}$

- **Clock jitter**
  - Temporal variations in consecutive edges of the clock signal; modulation + random noise
  - Cycle-to-cycle (short-term) $t_{JS}$
  - Long term $t_{JL}$

- **Variation of the pulse width**
  - for level sensitive clocking

Clock Skew and Jitter

- Both skew and jitter affect the effective cycle time
- Only skew affects the race margin
Clock Skew

# of registers

![Graph showing clock skew and insertion delay](image)

Sources of Skew and Jitter

1. Clock Generation
2. Devices
3. Interconnect
4. Power Supply
5. Temperature
6. Capacitive Load
7. Coupling to Adjacent Lines
Positive Skew

Launching edge arrives before the receiving edge

Negative Skew

Receiving edge arrives before the launching edge
Positive and Negative Skew

(a) Positive skew

(b) Negative skew

Outline

- Timing parameters
- Clock nonidealities (skew and jitter)
- **Impact of Clk skew on timing**
- Impact of Clk jitter on timing
- Flip-flop- vs. Latch-based timing
- Clock distribution
Timing Constraints

Cycle time: \( T_{\text{Clk}} > t_{c-q} + t_{\text{logic}} + t_{su} \)
Race margin: \( t_{\text{hold}} < t_{c-q, cd} + t_{\text{logic}, cd} \)

Impact of Clock Skew on Timing: Cycle Time (Long Path)

\( t_{c-q} + t_{\text{logic}} + t_{su} < T_{\text{Clk}} + \delta \)
\( T_{\text{Clk}} > t_{c-q} + t_{\text{logic}} + t_{su} - \delta \)
Impact of Clock Skew on Timing: Race Margin (Short Path)

Data must not arrive before this time

\[ t_{c-q,cd} + t_{logic,cd} > t_{hold} + \delta \]

\[ t_{hold} + \delta < t_{c-q,cd} + t_{logic,cd} \]

Impact of Clock Skew on Timing

Positive skew improves performance

\[ T_{Clk} > t_{c-q} + t_{logic} + t_{su} - \delta \]

Negative skew improves race margin

\[ t_{hold} + \delta < t_{c-q,cd} + t_{logic,cd} \]

Worst-case |\( \delta \) | really matters
How to counter Clock Skew?

Outline

- Timing parameters
- Clock nonidealities (skew and jitter)
- Impact of Clk skew on timing
  - Impact of Clk jitter on timing
- Flip-flop- vs. Latch-based timing
- Clock distribution
Impact of Clock Jitter

Cycle Time (Late-Early Problem)

\[ t_{c-q} + t_{\text{logic}} + t_{su} < T_{\text{Clk}} - t_{jitter} - t_{jitter} \]

\[ T_{\text{Clk}} > t_{c-q} + t_{\text{logic}} + t_{su} + 2t_{jitter} \]
Impact of Clock Jitter on Timing

Negative impact on cycle time

\[ T_{\text{Clk}} > t_{\text{c-q}} + t_{\text{logic}} + t_{\text{su}} + 2 \, t_{\text{jitter}} \]

No direct effect on race immunity
(same Clk edge)

Jitter reduces performance

Combined Impact of Clock Jitter and Skew
Impact of Clock Skew and Jitter: Cycle Time (Late-Early Problem)

\[ t_{c-q} + t_{\text{logic}} + t_{su} < T_{\text{Clk}} - t_{\text{jitter}} - t_{\text{jitter}} + \delta \]

\[ T_{\text{Clk}} > t_{c-q} + t_{\text{logic}} + t_{su} - \delta + 2 t_{\text{jitter}} \]

Impact of Clock Skew and Jitter: Race Margin (Early-Late Problem)

\[ t_{c-q,cd} + t_{\text{logic},cd} - t_{\text{jitter}} > t_{\text{hold}} + t_{\text{jitter}} + \delta \]

\[ t_{\text{hold}} + 2 t_{\text{jitter}} + \delta < t_{c-q,cd} + t_{\text{logic},cd} \]
Combined Impact of Clock Skew and Jitter on Timing

- **Cycle time** \( T_{\text{Clk}} > t_{c-q} + t_{\text{logic}} + t_{su} - \delta + 2 \ t_{jitter} \)
  - Positive skew improves performance
  - Negative skew reduces performance
  - Jitter reduces performance

- **Race** \( t_{\text{hold}} + 2 \ t_{jitter} + \delta < t_{c-q,cd} + t_{\text{logic,cd}} \)
  - Skew reduces race margin
  - Jitter reduces acceptable skew

Outline

- Timing parameters
- Clock nonidealities (skew and jitter)
- Impact of Clk skew on timing
- Impact of Clk jitter on timing
- **Flip-flop- vs. Latch-based timing**
- Clock distribution
**Flip-Flop – Based Timing**

Flip-flops are used only with static logic.

[Horowitz96]

**Flip-Flops and Dynamic Logic**

Flip-flops are used only with static logic.

EE141
Latch Timing

When data arrives to transparent latch
Latch is a ‘soft’ barrier

When data arrives to closed latch
Data has to be ‘re-launched’

Latch Timing (Cont.)

EE141
**Single-Phase Clock with Latches**

(Unger and Tan, Trans. on Comp. 10/86)

**Latch-Based Design**

L1 latch is transparent when $\phi = 1$

L2 latch is transparent when $\phi = 0$
Latch-Based Timing

Can tolerate skew!

Outline

- Timing parameters
- Clock nonidealities (skew and jitter)
- Impact of Clk skew on timing
- Impact of Clk jitter on timing
- Flip-flop- vs. Latch-based timing
- Clock distribution
Clock Distribution

Observe: Only Relative Skew is Important

More realistic H-tree

[Restle98]
Clock Network with Distributed Buffering

- Reduces absolute delay, and makes Power-Down easier
- Sensitive to variations in Buffer Delay

The Grid System

- No rc-matching
- Large power
Example: DEC Alpha 21164

Clock Frequency: 300 MHz - 9.3 Million Transistors
Total Clock Load: 3.75 nF
Power in Clock Distribution network: 20 W (out of 50)
Uses Two Level Clock Distribution:
- Single 6-stage driver at center of chip
- Secondary buffers drive left and right side clock grid in Metal3 and Metal4
  Total driver size: 58 cm!

21164 Clocking

- 2 phase single wire clock, distributed globally
- 2 distributed driver channels
  - Reduced RC delay/skew
  - Improved thermal distribution
  - 3.75nF clock load
  - 58 cm final driver width
- Local inverters for latching
- Conditional clocks in caches to reduce power
- More complex race checking
- Device variation
Clock Skew in Alpha Processor
EV6 (Alpha 21264) Clocking
600 MHz – 0.35 micron CMOS

Global clock waveform

- 2 Phase, with multiple conditional buffered clocks
  - 2.8 nF clock load
  - 40 cm final driver width
- Local clocks can be gated “off” to save power
- Reduced load/skew
- Reduced thermal issues
- Multiple clocks complicate race checking

21264 Clocking
EV6 Clock Results

GCLK Skew (at Vdd/2 Crossings)

GCLK Rise Times (20% to 80% Extrapolated to 0% to 100%)

EV7 Clock Hierarchy

Active Skew Management and Multiple Clock Domains

+ widely dispersed drivers
+ DLLs compensate static and low-frequency variation
+ divides design and verification effort
- DLL design and verification is added work
+ tailored clocks