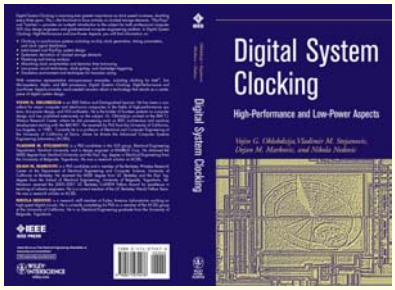
# Digital System Clocking:

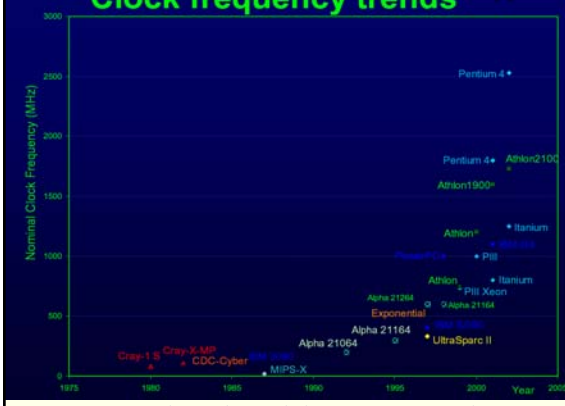
## *High-Performance and Low-Power Aspects*

Vojin G. Oklobdzija, Vladimir M. Stojanovic, Dejan M. Markovic, Nikola M. Nedovic




Wiley-Interscience and IEEE Press, January 2003

## Clock frequency trends



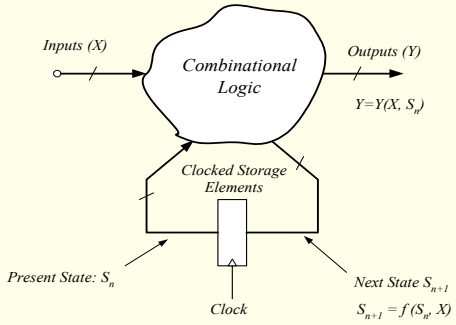



## *Clock Frequency of Selected Historic Computers and Supercomputers*

| System | Intro Date | Technology | Class | Nominal Clock Period (nS) | Nominal Clock Frequency (MHz) |
|---|---|---|---|---|---|
| Cray-X-MP | 1982 | MSI ECL | Vector Processor | 9.5 | 105.3 |
| Cray-1S,-1M | 1980 | MSI ECL | Vector Processor | 12.5 | 80.0 |
| CDC Cyber 180/990 | 1985 | ECL | Mainframe | 16.0 | 62.5 |
| IBM 3090 | 1986 | ECL | Mainframe | 18.5 | 54.1 |
| Amdahl 58 | 1982 | LSI ECL | Mainframe | 23.0 | 43.5 |
| IBM 308X | 1981 | LSI TTL | Mainframe | 24.5, 26.0 | 40.8,38.5 |
| Univac 1100/90 | 1984 | LSI ECL | Mainframe | 30.0 | 33.3 |
| MIPS-X | 1987 | VLSI CMOS | Microprocessor | 50.0 | 20.0 |
| HP-900 | 1982 | VLSI CMOS | Micro-mainframe | 55.6 | 18.0 |
| Motorola 68020 | 1985 | VLSI CMOS | Microprocessor | 60.0 | 16.7 |
| Bellmac-32A | 1982 | VLSI CMOS | Microprocessor | 125.0 | 8.0 |



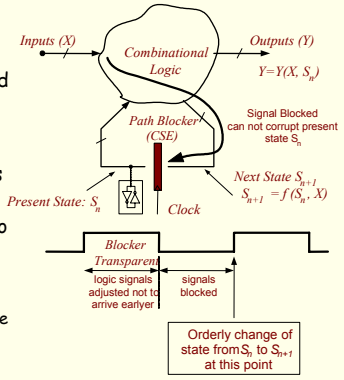
## *The concept of Finite-State Machine: FSM (Hoffman Model)*






## *Another Representation of FSM*

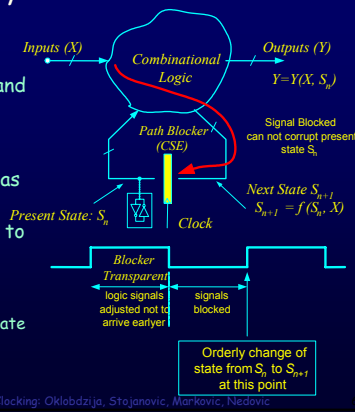
- Clocked Storage Elements: Flip-Flops and Latches should be viewed as synchronization elements, not merely as storage elements !
- Their main purpose is to synchronize fast and slow paths:
  - prevent the fast path from corrupting the state



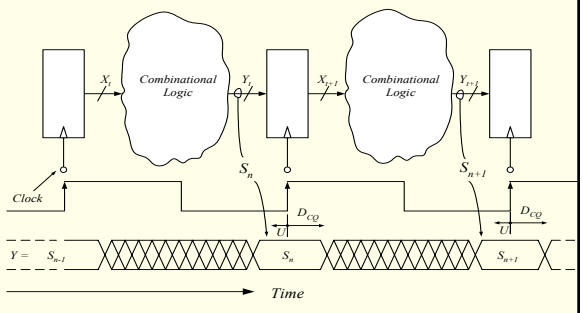



## *Another Representation of FSM*

- Clocked Storage Elements: Flip-Flops and Latches should be viewed as synchronization elements, not merely as storage elements !
- Their main purpose is to synchronize fast and slow paths:
  - prevent the fast path from corrupting the state

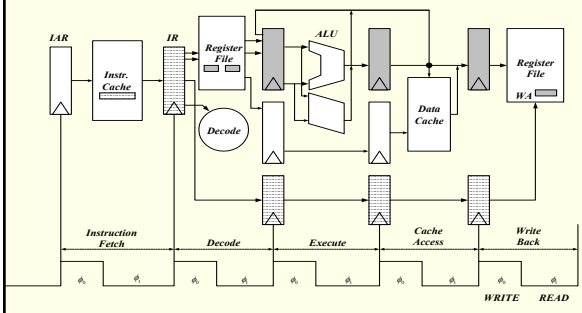


## State Changes in the Finite-State Machine



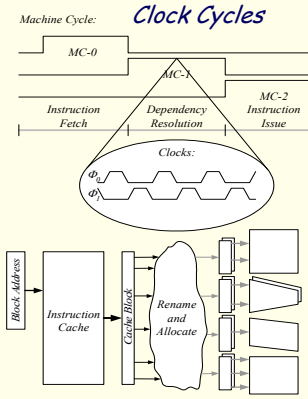



## Diagram of a Pipelined System



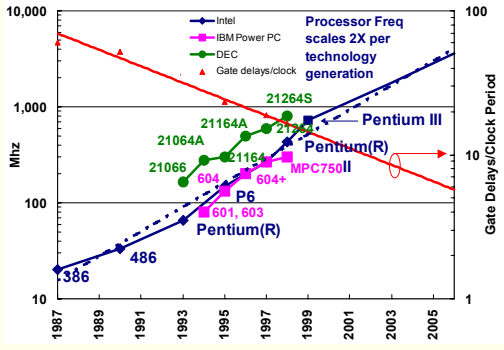



## Machine Execution Phases with Respect to the Clock Cycles



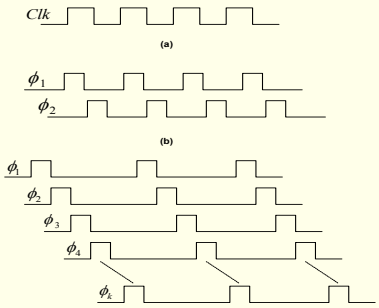



## Increase in the Clock Frequency and Decrease in the Number of Logic Levels

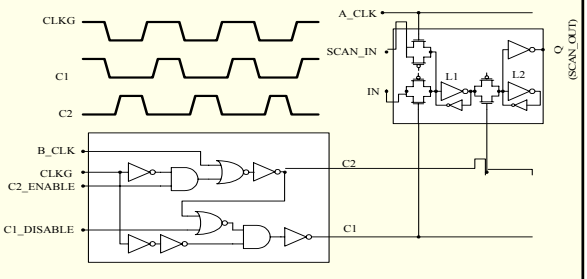


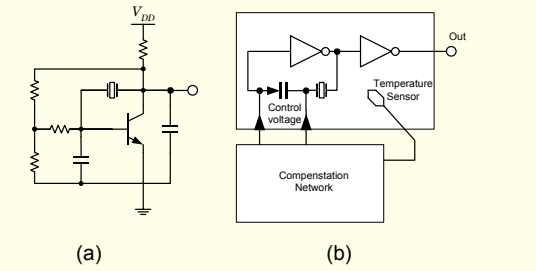



## System clocking schemes: *(a) single-phase clock; (b) two-phase clock; (c) multiple-phase clock*




## Local generation of Two-Phase Clocks as used in IBM S/390 G4

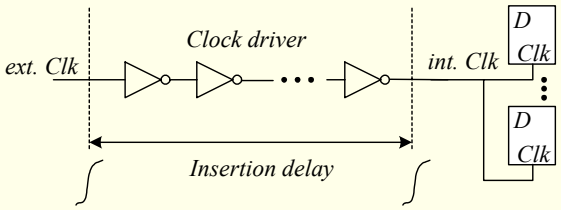





(a) Crystal oscillator

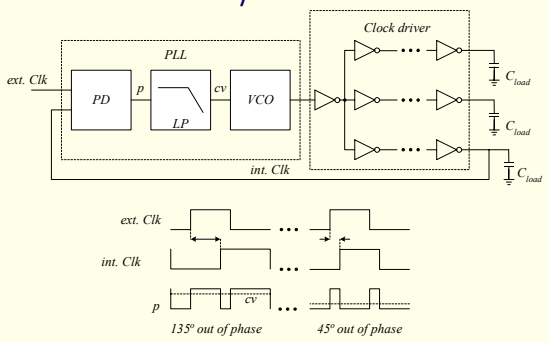
(b) Temperature-compensated crystal oscillator

## On-Chip Clock Insertion Delay



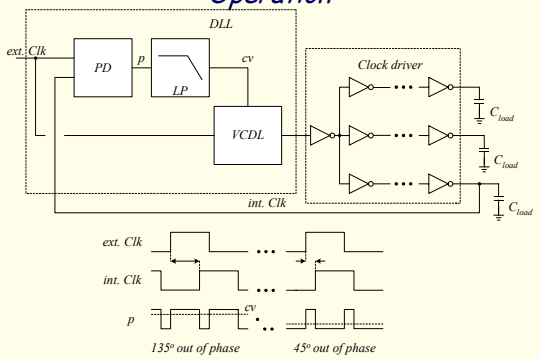



## Phase-Locked Loop Block Diagram and Operation



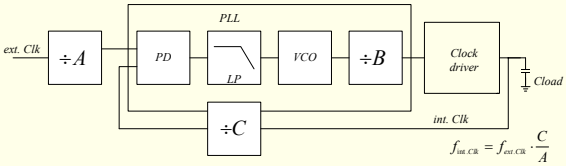



## Delay-Locked Loop Block Diagram and Operation



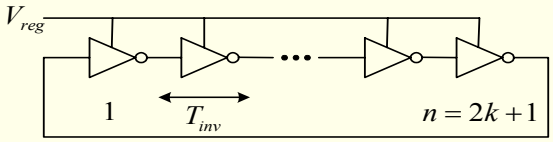



## PLL Frequency Multiplication



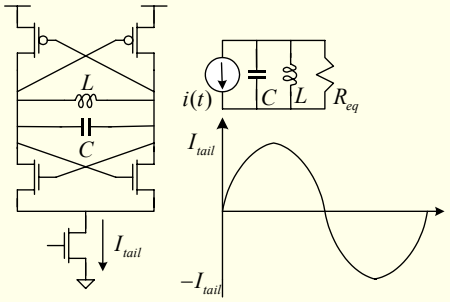



## Ring-Oscillator-Based VCO with CMOS Inverters as Delay Elements



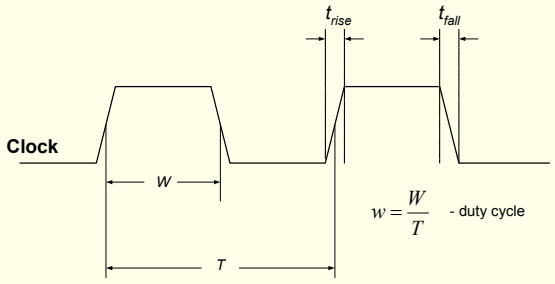

$$f_{osc} = \frac{1}{2nT_{inv}}; \quad k \geq 1$$

## LC Tank-Based VCO, Equivalent AC Circuit Model and Current Waveform



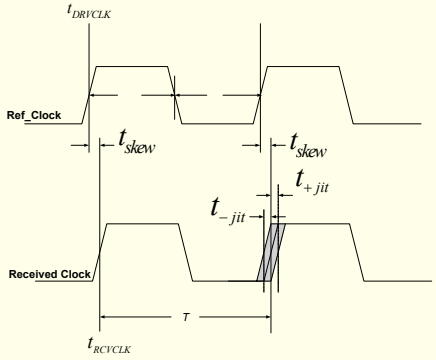



## Clock Parameters: Period (T), Width, Rise and Fall Times



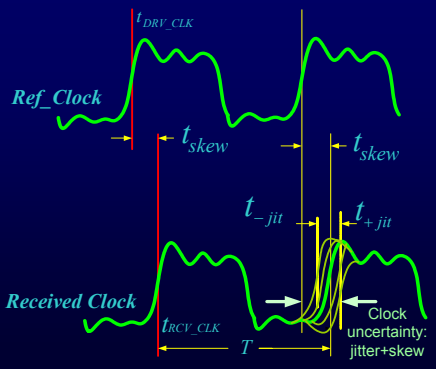



## Clock Parameters: Period, Width, Clock Skew and Clock Jitter



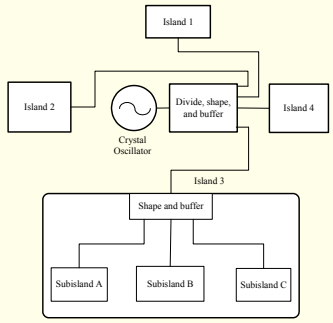



## Clock Uncertainties





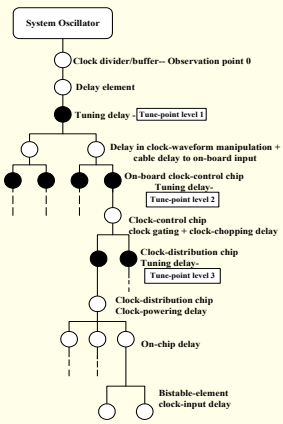

## The Concept of Logic Islands (Wagner 1988), Copyright © 1988 IEEE





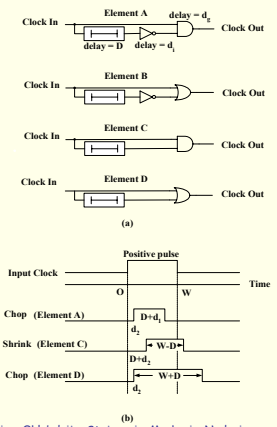

## Clock Tuning Points

(Wagner 1988), Copyright © 1988 IEEE



## *Various Clock Shaping Elements and Obtained Clock Signals.*

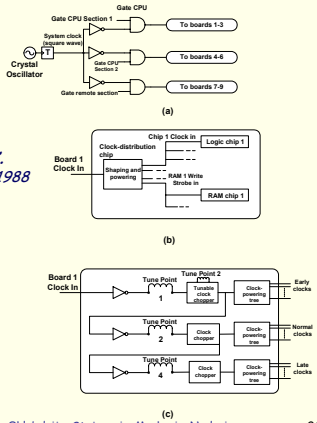
*(Wagner 1988), Copyright © 1988 IEEE*






## *Clock distribution network within a system, (b) on the board, and (c) tuning of the clock.*

*(Wagner 1988), Copyright © 1988 IEEE*

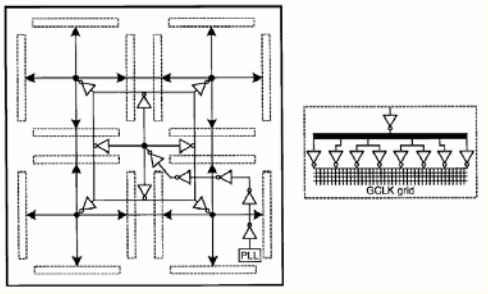





## *Clock distribution methods:*

*(a) an RC matched tree, and (b) a grid*

*(Bailey and Benschneider 1998), Copyright © 1988, IEEE*

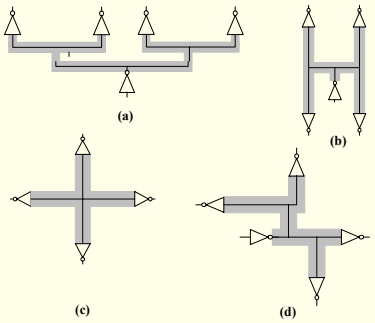





## *RC delay matched clock distribution topologies:*

*(a) a binary tree, (b) an H tree, (c) an X tree, (d) an arbitrary matched RC matched tree*

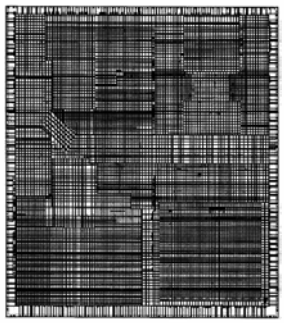
*(From Bailey in Chandrakasan et al. 2001), Copyright © 1988, IEEE*






## *Clock distribution grid used in a DEC Alpha 600-MHz processor*

*(Bailey and Benschneider 1998), Copyright © 1988, IEEE*

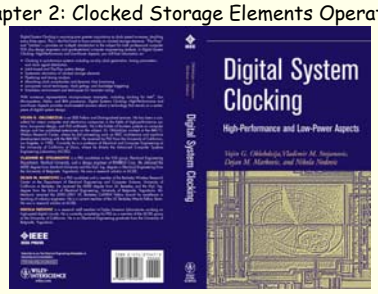


# Digital System Clocking:

## *High-Performance and Low-Power Aspects*

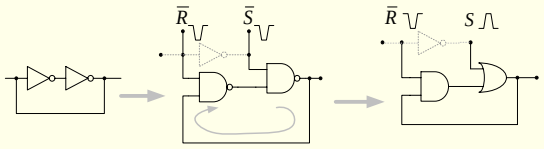
Vojin G. Oklobdzija, Vladimir M. Stojanovic, Dejan M. Markovic, Nikola M. Nedovic

Chapter 2: Clocked Storage Elements Operation




Wiley-Interscience and IEEE Press, January 2003

## *Evolution of a Latch :*



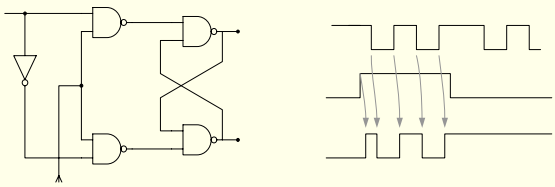

*(a) keeper; information is latched*

*(b) S-R latch; information can be modified*

*(c) S-o-P latch; S-R latch that can implement a function – IMPORTANT !*



## *Adding Clock Control to a Latch*

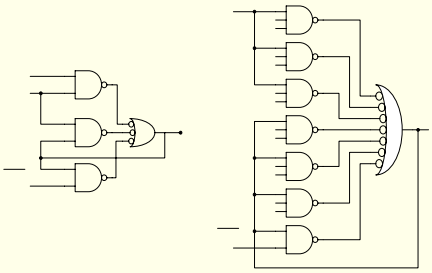


*Latching can be done only when clock is active, otherwise no change is allowed:*

*(a) Clocked D-latch; (b) timing diagram of clocked D-latch*



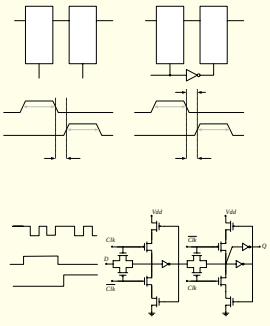
## *Importance of Latch S-o-P Configuration*



*(a) Basic Earl's Latch; (b) Implementing the Carry function*



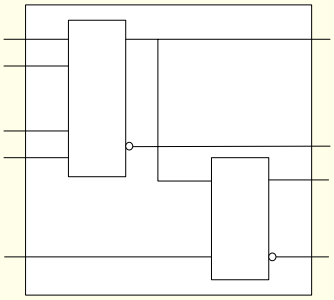
## *Master-Slave Latch*



*(a) non-overlapping clocks; (b) single external clock; (c) timing diagram; (d) PowerPC 603 MS Latch (Gerosa, JSSC 12/94), Copyright © 1994 IEEE*

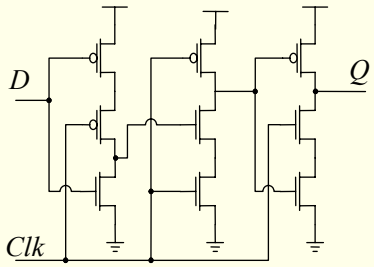


## *IBM LSSD Compatible Clocked Storage Element*



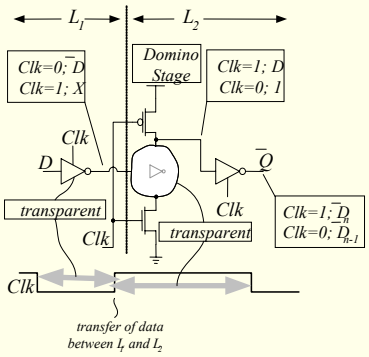
## True-single-phase-clock (TSPC) M-S latch

*Yuan and Svenson (1989), Copyright © 1989 IEEE*





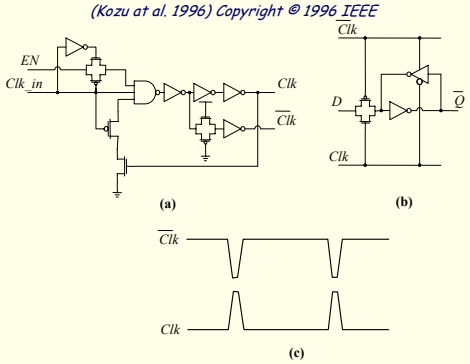

## TSPC M-S Latch Operation






## Pulse latch

*(Kozu at al. 1996) Copyright © 1996 IEEE*

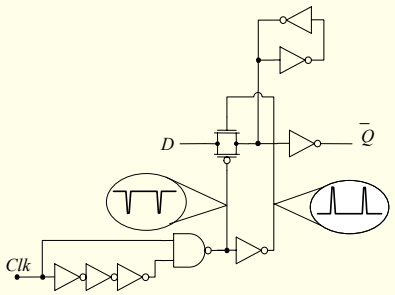



*(a) local clock generator; (b) single latch; (c) clock signals*



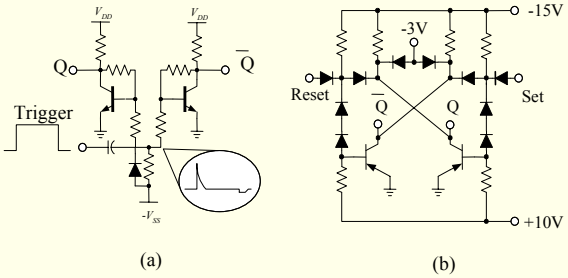
## Pulse latch: Intel's explicit pulsed latch

*(Tschanz at al. 2001), Copyright © 2001 IEEE*





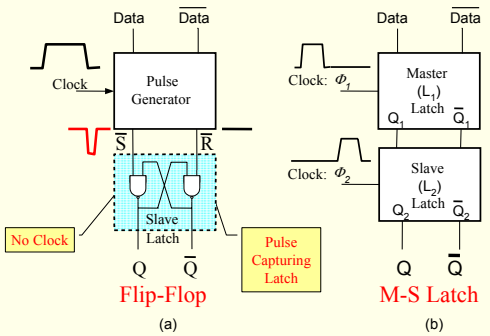

## Flip-flop




*(a) Early version of a flip-flop; (b) PDP-8 direct set-reset sequential element*

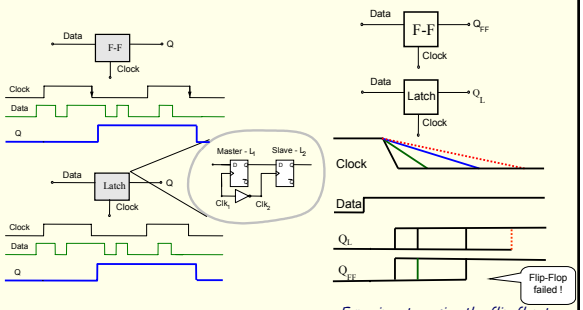


## Difference between Flip-flop and M-S Latch




*(a) General flip-flop structure; (b) general M-S latch structure*

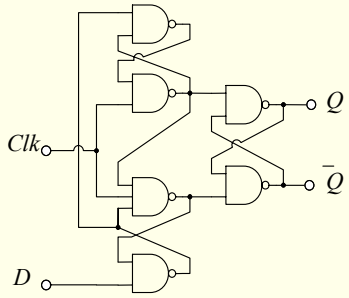
## Black-box view of the Flip-flop and M-S latch




*Experiment causing the flip-flop to fail while the M-S latch is still operational*



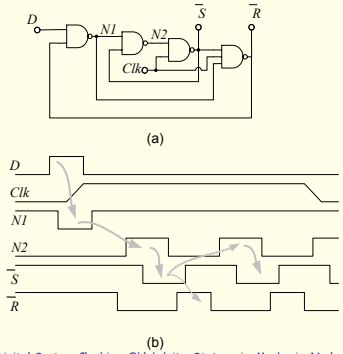
## Texas Instruments SN7474 Flip-flop




Very much used and analyzed

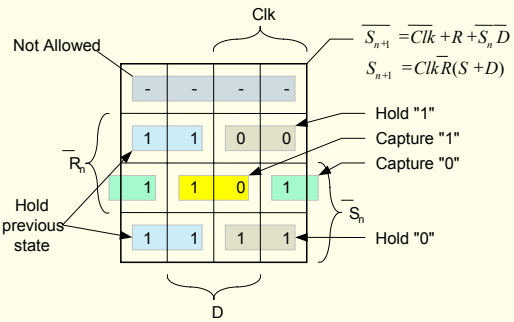


## Pulse generator block of SN7474: malfunctioning due to a gate-delay



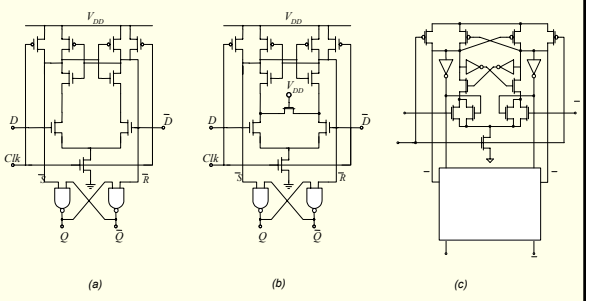



## Systematic Derivation of a FF using Karnaugh map




*Derivation of the pulse-generating stage of a flip-flop (only the Sn signal is shown)*

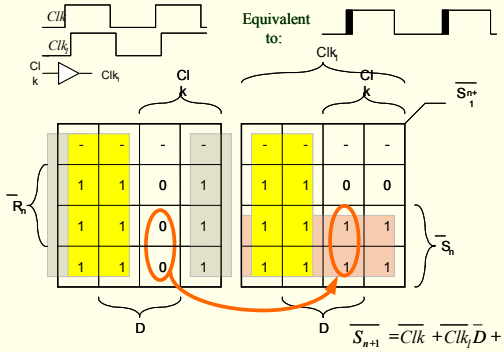




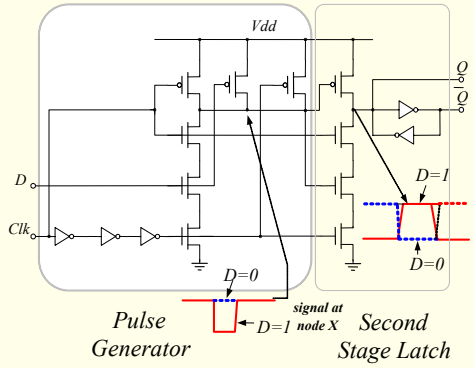

*(a) Pulse generator stage of the sense-amplifier flip-flop. (Madden and Bowhill, 1990);*
*(b) Improvement for floating nodes. (Dobberpuhl, 1997);*
*(c) Pulse generator stage improvement by proper design. (Nikolic and Oklobdzija, 1999).*



## Creating the time reference points for opening and shutting the flip-flop

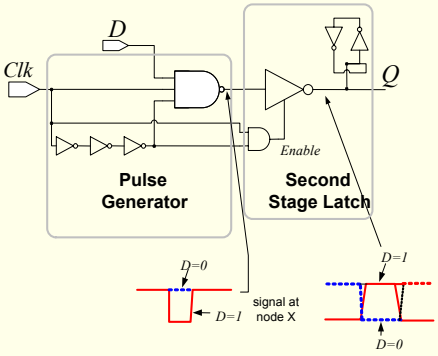


## Hybrid-Latch Flip-Flop (HLFF) introduced by Partovi at al.





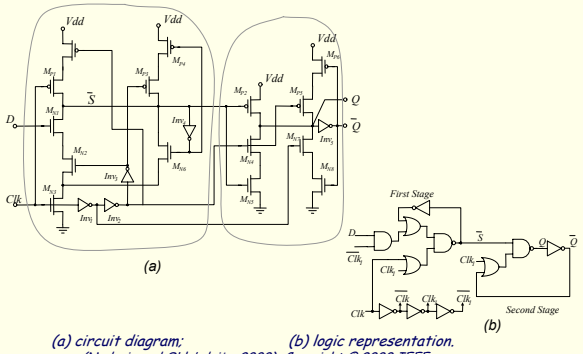

## Logic representation of Partovi's flip-flop (HLFF)






## Systematically derived single-ended flip-flop

*(according to the slide no. 18)*




*(a) circuit diagram; (b) logic representation.*

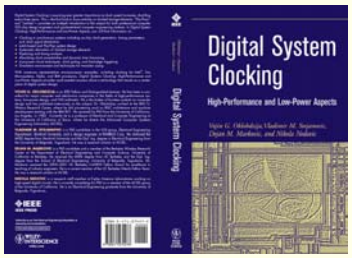
*(Nedovic and Oklobdzija, 2000). Copyright © 2000 IEEE*

# Digital System Clocking:

## *High-Performance and Low-Power Aspects*

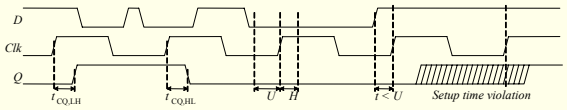
Vojin G. Oklobdzija, Vladimir M. Stojanovic, Dejan M. Markovic, Nikola M. Nedovic

Chapter 3: Timing and Energy Parameters




Wiley-Interscience and IEEE Press, January 2003

## *Basic timing diagram in flip-flops*

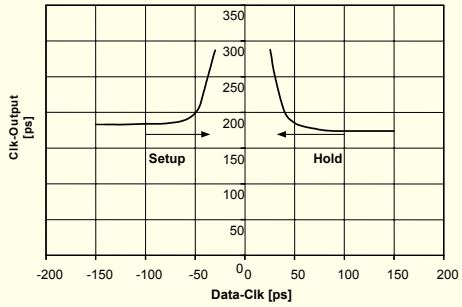



Definitions:

- Clock-to-Q Delay: $t_{CQ}$ low-to high= $t_{CQ,LH}$ high-to-low= $t_{CQ,HL}$
- Setup time U
- Hold time H



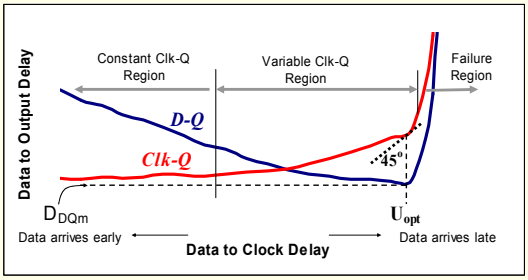
## *Setup and hold time behavior as a function of clock-to-output delay*




Neither Setup nor Hold time are fixed constant parameters.
They are function of Data-to-Clock time "distance".



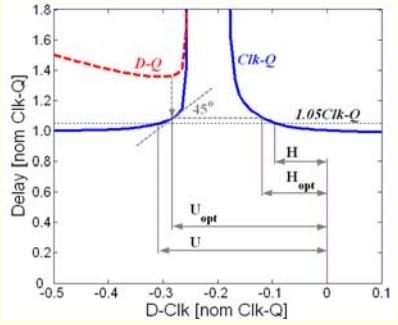
## *Setup time behavior as a function of data-to-output delay*




When D-to-Q delay is observed, we can see that data can come closer to the triggering event than we thought.
Data-to-Q is the RELEVANT parameter – NOT Clock-to-Q as many think !



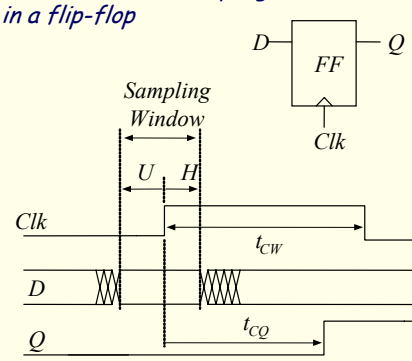
## *D-Q and Clk-Q delay as a function of D-Clk offset*



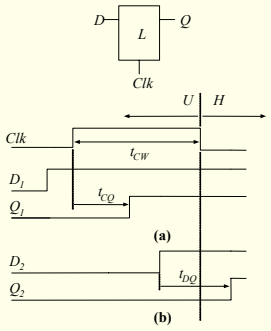

Determining the optimal setup $U_{opt}$ and hold time $H_{opt}$.



## *Setup time, hold time, sampling window and clock width in a flip-flop*



## Latch: setup and hold time

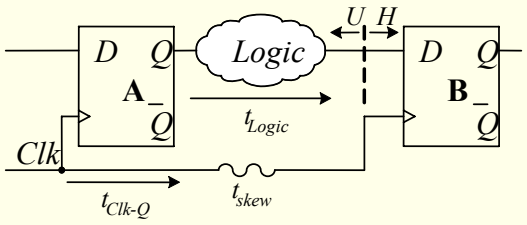



Unlike Flip-Flop, Latch has two situations:
(a) Data is ready – waiting for the clock
(b) Latch is open – data arrives during the active clock signal

*(a) early data D1 arrival; (b) late data D2 arrival*

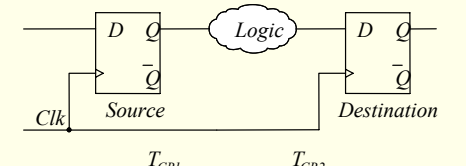


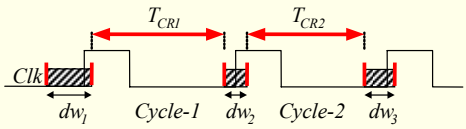
## Illustration of a data path






## Late Data Arrival and Time Borrowing in a pipelined design




$dw_1 > dw_3 > dw_2$ $\quad T_{CR1} > T_{CR2}$

Data-to-Output (Q) time window moves around the time axis; (becoming larger or smaller)



## Early Data Arrival and Internal Race Imunity

- The maximum clock skew that system can tolerate is determined by the clock storage elements:
  - If Clk-Q delay of CSE is shorter than H, race can occur if there is no logic inbetween.
  - If Clk-Q is greater than H + possible skew – no problem:

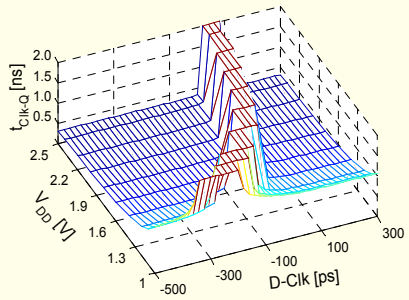
$$t_{Clk\text{-}Q} > H + t_{skew}$$

- Internal race immunity:

$$R = t_{Clk\text{-}Q} - H$$



## Impact of supply voltage on the sampling window




Sampling window determines the minimum required duration of data signal



## Energy Parameters

## Components of Energy Consumption

- Switching Energy
- Short-Circuit Energy
- Leakage Energy
- Static Energy



## Components of Energy Consumption

$$E = \int_{t}^{t+T} V_{DD} \cdot i_{V_{DD}}(\tau) \cdot d\tau$$

Energy consumed by CSE during one clock period T, where $t$ is chosen to include all relevant transitions: arrival of new data, clock pulse, and output transition.

This energy has four components:

$$E = E_{switching} + E_{short-circuit} + E_{leakage} + E_{static}$$

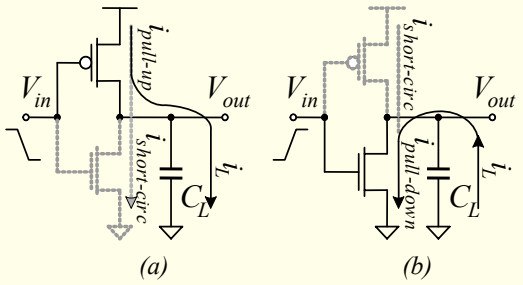
***Switching Energy:***

$$E_{switching} = \sum_{i=1}^{N} \alpha_{0-1}(i) \cdot C_i \cdot V_{swing}(i) \cdot V_{DD}$$

- $N$ is the number of nodes
- $C_i$ is the capacitance of the node $I$
- $\alpha_{0-1}(i)$ is the probability that a transition occurs at the node $i$
- $V_{swing}(i)$ is voltage swing of the node $I$



## *Short-circuit current in an inverter*

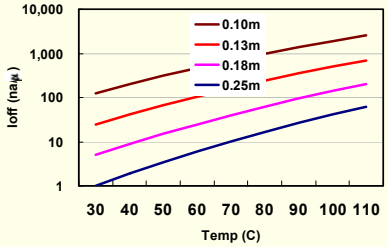



*(a) pull-up; (b) pull-down operation*



## *Projected leakage currents*

Leakage power will soon become a significant portion of the total power consumption in modern microprocessors




Assuming doubling of transistors / generation the leakage current will increase about 7.5 times corresponding to a 5 times increase in total leakage power



## Where does the Energy go in CSE ?

1. Internal clocked nodes in storage elements
2. Internal non-clocked nodes in storage elements
3. Data and clock input load
4. Output load.



## Energy Breakdown

1. Internal clocking energy
2. Data and Clock Input Energy
3. Energy in Internal Non-clocked Nodes
4. Energy in Output Load
5. Energy per Transition
6. Glitching Energy

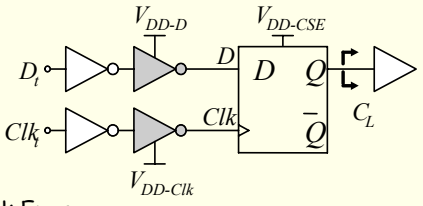
## Energy Breakdown in Clocked-Storage Elements during one of the possible input data transitions

| | $E_{0-0}$ | $E_{0-1}$ | $E_{1-0}$ | $E_{1-1}$ |
|---|---|---|---|---|
| $E_{Clk}$ | Y/N | Y | Y | Y/N |
| $E_{int}$ | Y/N | Y | Y | Y/N |
| $E_{ext}$ | N | Y/N | Y/N | N |

**Two cases:**
- **Storage elements without pre-charge nodes**
- **Storage elements with pre-charge nodes**



## CSE characterization and Test setup




- Clock Energy
- Internal Energy: $E_{\text{int}} = \frac{E_{0-1} + E_{1-0}}{2} - E_{Clk} - E_{Load}$
- Energy in Output Load

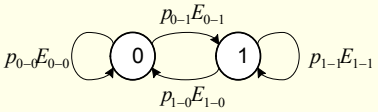


## Energy per transition

- The energy-per-transition is the total energy consumed in a CSE during one clock cycle for a specified input data transition: *0-0*, *0-1*, *1-0*, or *1-1*
- This metric is crucial in that it yields significant insight about circuit energy
- By inspection of the node activity in a CSE for different input data transitions, the energy-per-transition can be utilized to obtain the energy breakdown between clocked nodes, internal nodes, and the external output load.
- This forms a good basis for the study of alternative circuit techniques that deal with internal clock gating.
- The energy breakdown information also offers valuable information about the tradeoffs associated with reduced clocking energy and the energy penalty incurred by the clock-gating logic, thus providing a better understanding of the optimization goals for the overall design



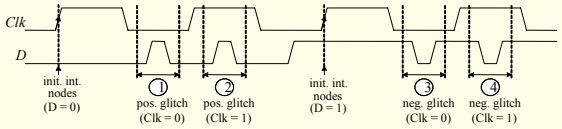
## Energy per transition




$$E_{average} = p_{0-0} \cdot E_{0-0} + p_{0-1} \cdot E_{0-1} + p_{1-0} \cdot E_{1-0} + p_{1-1} \cdot E_{1-1}$$



## Glitching Energy in CSEs




- Glitches are generated by the unintended transitions propagating from the fan-in gates, termed *propagating glitches*.
- Glitches produced by non-glitch transitions at the inputs, called *generated glitches*

$$E_{avg-glitch} = \sum_{i=1}^{4} \beta_i \cdot E_{g_i}$$



## Interface with Clock Network and Combinational Logic

- We assumed that the data and clock inputs were supplied by drivers with sufficient drive strength.
- The input clock and data capacitances are important interface parameters for the clock network and logic design.
- The clock network designer and logic designer need to be aware of these capacitances in order to design circuits that drive storage elements.

## Interface with Clock Network and Combinational Logic

***Interface with Combinational Logic:***

The relevant parameters to the combinational logic designer are:

- *CSE input data slope*
- *Input data capacitance*

The data slope affects performance and energy consumption of both driving logic and storage elements.

Clock and data slopes are generally not equal.



## Interface with Clock Network and Combinational Logic

***Interface with Clock Network:***

- CSEs are affected by *clock skew* and *clock slope*.
- The total load of the clock distribution network is defined by the input capacitance of the clock node and number of CSEs on a chip.
- Increase in *clock slope* results in degradation of the CSE performance - the clock network designer has to know what slopes CSE can tolerate.
- This is especially important if Flip-Flops are used.
- The clock slope also affects energy consumption of the clock distribution network.
  - If larger clock drivers with smaller fanout are used, the clock edges are sharper and the storage element performance better, at the expense of an increase in energy consumption of the clock network.
  - Optimal tradeoff is achieved with minimal energy consumption that delivers the desired storage element performance.



## Interface with Clock Network and Combinational Logic

- To evaluate the *total clocking energy* per clock cycle in the entire clock subsystem, one needs to add the energy consumed in the clock distribution network.
- The energy consumed in the clock distribution network depends on the total switched capacitance which is determined by the total number of clocked storage elements on a chip and the input capacitance of their clock inputs, the total wiring capacitance, and the total switched capacitance of clock drivers as given by:

$$C_{distrib-net} = N_{FF} \cdot C_{in-Clk,FF} + C_{wire} + C_{sw-buff}$$

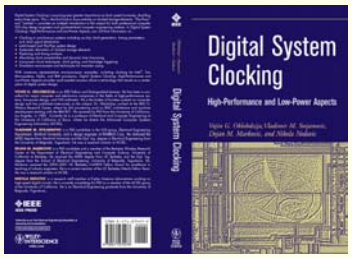
The last two terms depend on buffer insertion/placement strategy and should be minimized.

# Digital System Clocking:

*High-Performance and Low-Power Aspects*

Vojin G. Oklobdzija, Vladimir M. Stojanovic, Dejan M. Markovic, Nikola M. Nedovic

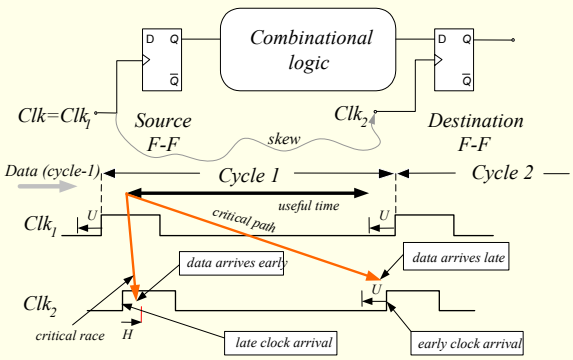
Chapter 4: Pipelining and Timing Analysis




Wiley-Interscience and IEEE Press, January 2003

# *Timing in a digital system using a single clock and flip-flops*

## *Timing in a digital system using a single clock and flip-flops*






## Late Data Arrival Analysis
(single clock, FF)

If we set the time reference to *t=0* for the leading edge of the clock

*We set max. clock uncertainties to* $T_L$ from the nominal time of arrival, $T_T$ (for the trailing edge).

$D_{CQm}$ represents the minimal *Clock-to-Q* (output) delay of the Flip-Flop

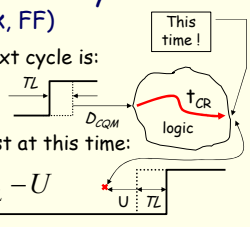
$D_{Lm}$ represents minimal delay through the logic (as opposed to index M where $D_{CQM}$ and $D_{LM}$ represent maximal delays).



## Late Data Arrival Analysis
(single clock, FF)

The latest data arrival in the next cycle is:

$$t_{DLN} = T_L + D_{CQM} + t_{CR}$$




But it should be there at least at this time:

$$t_{DLN} = P - T_L - U$$

So that:

$$P - T_L - U \geq T_L + D_{CQM} + t_{CR}$$

Giving us:

$$P \geq 2T_L + U + D_{CQM} + t_{CR}$$

$$t_{CR} \leq P - 2T_L + U + D_{CQM}$$



## Early Data Arrival Analysis
(single clock, FF)

It is commonly misunderstood that the Flip-Flop provides edge-to-edge timing and is thus easier to use, as compared to the Latch based system, because it does not need to be checked for fast paths in the logic (Hold-time violation).

*This is not true, and a simple analysis that follows demonstrates that even with the Flip-Flop design the fast paths can represent a hazard and invalidate the system operation.*

## Early Data Arrival Analysis
(single clock, FF)

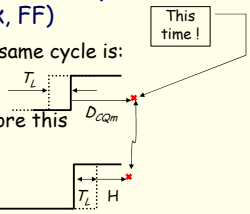
If the clock controlling the Flip-Flop releasing the data is skewed so that it arrives early,
and the clock controlling the Flip-Flop that receives this data arrives late,
a hazard situation exists.
This same hazard situation is present if the data travels through a *fast path* in the logic.
A *fast path* is the path that contains very few logic blocks, or none at all.
This hazard is also referred to as *critical race* (or *race-through*)



## Early Data Arrival Analysis
(single clock, FF)

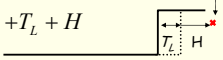
The earlyest data arrival in the same cycle is:

$$t_{DEArr} = -T_L + D_{CQm} + D_{Lm}$$




But it should be there not before this time:

$$t_{DEArrN} = +T_L + H$$




So that:

$$-T_L + D_{CQm} + D_{Lm} > +T_L + H$$

Giving us limits on the fast paths:

$$D_{Lm} > D_{LB} = 2T_L + H - D_{CQm}$$

$$W \geq T_L + T_T + t_{CWm}$$



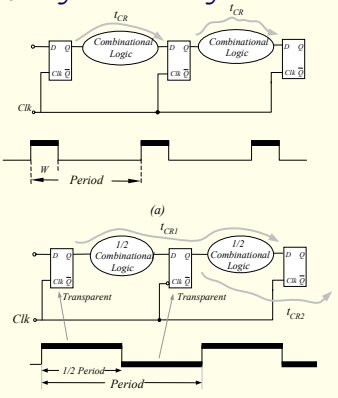
# System using a Single Latch

## Analysis of a System using a Single Latch

System using a single Latch is more complex to analyze than Flip-Flop based one.
Single Latch is transparent while the clock in active and the possibility for the *race-through* exists.
This analysis is still much simpler than a general analysis of a system using two Latches (Master-Slave Latch based system).
Use of a single Latch represents a hazard due to the transparency of the Latch, which introduces a possibility of races in the system.
Therefore, the conditions for the single-latch based system must account for critical race conditions.
Presence of the CSE delay decreases the "useful time" in the pipeline cycle. Therefore, in spite of the hazards introduced by such design, the additional performance gain may well be worth the risk.



*Two ways of using a latch in a single-latch-based system:*






## Late Data Arrival Analysis

In the case of a Latch, input signal need to arrive at least a Setup Time *U* before the trailing edge.
This edge could arrive earlier. Thus, the latest arrival of data into the latch that assures reliable capture after the period *P* has to be:

$$t_{DLArr} \leq W - T_T - U + P$$

Data captured at the end of the clock period could be a result of two events (whichever later):

a) The data was ready, and clock arrived at the latest possible moment $T_L$, and the worse case delay of the Latch i.e. $D_{CQM}$ was incurred.
b) The clock was active and data arrived at the last possible moment, which is a setup time *U* and clock skew time $T_T$ before the trailing edge of the clock.

- In both cases the path through the logic was the longest path $D_{LM}$.

Under the worse scenario data must arrive in time:

$$\max\{T_L + D_{CQM}, W - T_T - U + D_{DQM}\} + D_{LM} \geq W - T_T - U + P$$

## Late Data Arrival Analysis

- This gives a constraint for the clock speed in terms of $P$ such as:

$$P \geq \max\{T_L + T_T + U + D_{CQM} - W, D_{DQM}\} + D_{LM}$$

This inequality breaks down into two inequalities:

$$P \geq D_{LM} + D_{CQM} + T_L + T_T + U - W$$
$$P_m = P \geq D_{LM} + D_{DQM}$$

This shows the minimal bound for *Pm*, which is the time to traverse the loop:

*"Starting from the leading edge of a clock pulse, there must be time, under worst case, before the trailing edge of the clock in the next cycle, for a signal to pass through the Latch and the logic block in time to meet the setup time constraint".*

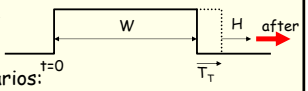
The value of *P = Pm* determines the highest frequency of the clock.



## Early Signal Arrival Analysis
(Single Latch Based System)

The fastest signal, should arrive at the minimum a hold time after the latest possible arrival of the same clock:

$$t_{DEArrN} \geq W + T_T + H$$

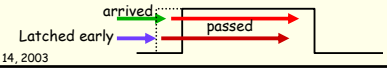



There are two possible scenarios:

(a) signal was latched early and it passed through a fast path in the logic

(b) it arrived early while the Latch was transparent and passed through the fast Latch and fast path in the logic.

$$t_{DEArrN} = \min\{t_{CEL} + D_{CQm}, t_{DEArr} + D_{DQm}\} + D_{Lm}$$






## Early Signal Arrival Analysis
(Single Latch Based System)

The earliest arrival of the clock $t_{CEL}$ happens when the leading edge of the clock is skewed to arrive at $-T_L$. Thus, the condition for preventing race in the system is expressed as:

$$\min\{-T_L + D_{CQm}, t_{DEArr} + D_{DQm}\} + D_{Lm} \geq W + T_T + H$$

The earliest possible arrival of the clock, plus clock-to-output delay of the Latch has to occur earlier in time than early arrival of the data, thus:

$$-T_L + D_{CQm} + D_{Lm} \geq W + T_T + H$$

which gives us a lower bound on the signal delay in the logic:

$$D_{Lm} > D_{LmB} \geq W + T_T + T_L + H - D_{CQm}$$



## Early Signal Arrival Analysis
(Single Latch Based System)

The conditions for reliable operation of a system using a single Latch are:

$$P_m = P \geq D_{LM} + D_{CQM} + T_L + T_T + U - W$$
$$P \geq D_{LM} + D_{DQM}$$
$$D_{Lm} > D_{LmB} \geq W + T_T + T_L + H - D_{CQm}$$

the increase of the clock width *W* may be beneficial for speed, but it increases the minimal bound for the fast paths



## Early Signal Arrival Analysis
(Single Latch Based System)

Maximum useful value for *W* is obtained when the period *P* is minimal:

$$W^{opt} = T_L + T_T + U + D_{CQM} - D_{DQM}$$

Substitute the optimal clock width $W^{opt}$ we obtain the values for the maximal speed and minimal signal delay in the logic which has to be maintained in order to satisfy the conditions for optimal single-latch system clocking:

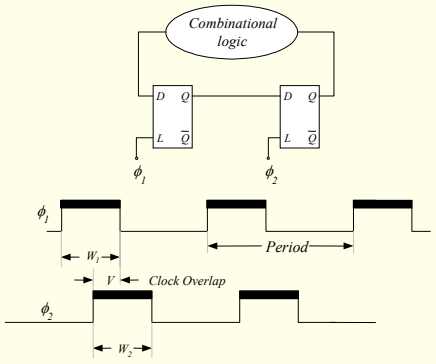
$$P \geq D_{LM} + D_{DQM}$$
$$D_{LmB} = 2(T_T + T_L) + H + U + D_{CQM} - D_{CQm} - D_{DQM}$$

In a single Latch system, it is possible to make the clock period *P* as small as the sum of the delays in the signal path: Latch and critical path delay in the logic. This can be achieved by adjusting the clock width *W*, while taking care of $D_{LmB}$



## Analysis of a System with two-phase Clock and two Latches in an M-S arrangement

## System using two-phase clock and two latches in M-S arrangement






## System using two-phase clock and two latches in M-S arrangement

From the latest signal arrival analysis, several conditions can be derived. First, we need to assure an orderly transfer into $L_2$ Latch (*Slave*) from the $L_1$ Latch (*Master*), even if the signal arrived late (in the last possible moment) into the (*Master*) $L_1$ Latch. This analysis yields the following conditions:

$$W_2 \geq V + U_2 - U_1 + D_{1DQM} + T_{2T} + T_{1L}$$

$$W_1 + W_2 \geq V + U_2 + D_{1CQM} + T_{1T} + T_{2T}$$

These conditions assure timely arrival of the signal into the *L2* Latch, thus an orderly *L1-L2* transfer (from *Master* to *Slave*)



## System using two-phase clock and two latches in M-S arrangement

The analysis of the latest arrival of the signal into *L1* Latch in the next cycle (critical path analysis) yields to the equations:

$$P \geq D_{1DQM} + D_{2DQM} + D_{LM}$$

$$W_1 \geq -P + D_{1CQM} + D_{2DQM} + U_1 + D_{LM} + T_{1L} + T_{1T}$$

$$P \geq -V + D_{2CQM} + U_1 + D_{LM} + T_{1T} + T_{2L}$$

This conditions assure timely arrival of the signal that starts on the leading edge of $\phi 1$, traverses the path through *L2*, the longest path in the logic and arrives before the trailing edge of $\phi 1$, in time to be captured.

The last equation shows that the amount of overlap *V* between the clocks $\phi 1$ and $\phi 2$ allows the system to run at greater speed.



## System using two-phase clock and two latches in M-S arrangement

If we increase *V* we can tolerate longer "*critical path*" $D_{LM}$. However, the increase of the clock introduce a possibility of race conditions, thus requiring a *fast path* analysis. High-performance systems are designed with the objective of maximizing performance. Therefore, overlapping of the clocks is commonly employed, leading to the constraint of the minimal signal delay in the logic $D_{LmB}$:

$$D_{Lm} > D_{LmB} = V + H_1 + T_{1T} + T_{2L} - D_{2CQm}$$

The maximal amount of overlap *V* that can be used is:

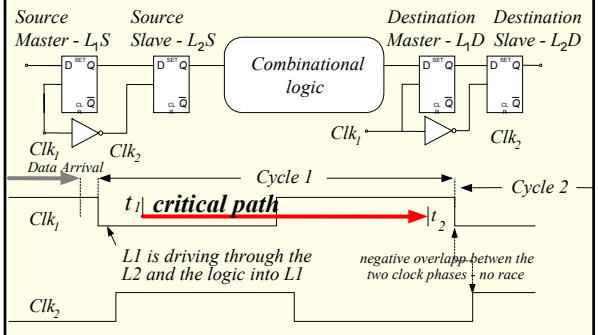
$$V_{\max} = T_{1T} + T_{2L} + D_{2CQM} + U_1 - D_{1DQM} - D_{2DQM}$$

For maximal performance, it is possible to adjust the clock overlap *V* so that the system runs at the maximal frequency.



## M-S (L1-L2 latch) with non-overlapping clocks $\Phi_1$ and $\Phi_2$ obtained by locally generating clock $\Phi_2$

*(This arrangement is also commonly referred to as flip-flop)*

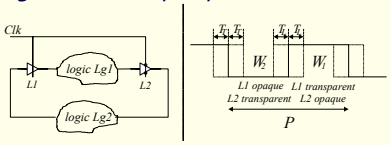




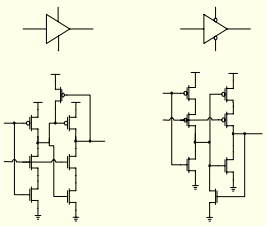

# Example: Clocking in the first generation of Alpha Processor (WD21064)

## Timing arrangement and Latches used in the first-generation Alpha processor

(a)




(b)





## Example: Clocking the Alpha Processor

Clock skew: $T_L = T_T = 20$ps, for both edges of the clock.

Latch L1 parameters are: clock to Q delay $D_{CQM} = 50$ps, $D_{CQm} = 30$ps, D to Q delay $D_{DQM} = 60$ps, setup time U = 20ps, hold time H = 30ps.

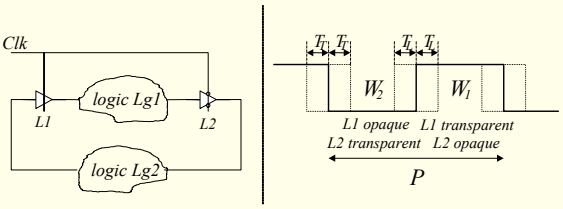
Latch L2 parameters are: $D_{CQM} = 60$ps, $D_{CQm} = 40$ps, $D_{DQM} = 70$ps, U = 30ps, H = 40ps.

The critical paths in the logic sections 1 and 2 are: $D_{L1M} = 200$pS and $D_{L2M} = 170$ps



## Example: Clocking the Alpha Processor

For the given clock setup: *U=0* and clearly *P=W1+W2*






Clock

O

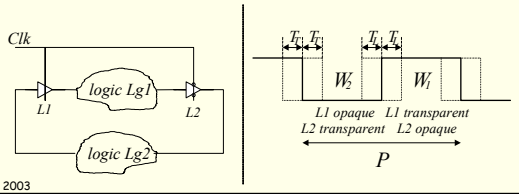
## Example: Clocking the Alpha Processor

Nominal time, t=0 is set at the leading edge of the clock. The latest allowed data arrival times into latches L1 and L2, respectively:

$$t_{D1LArr} \le W_1 - T_T - U_1 \qquad t_{D2LArr} \le P - T_L - U_2$$

The latest arrival time of data into latch L2 is limited by the time at which latch L1 releases the data into the logic stage Logic1:

$$t_{D2LArr} = \max\left\{t_{D1LArr} + D_{1DQM}, T_L + D_{1CQM}\right\} + D_{L1M}$$






## Example: Clocking the Alpha Processor

With the nominal time, t=0 set at the leading edge of the clock we obtain the latest allowed data arrival times into latches L1 and L2, respectively:

$$t_{D1LArr} \le W_1 - T_T - U_1 \qquad (4.34)$$
$$t_{D2LArr} \le P - T_L - U_2 \qquad (4.35)$$

The latest arrival time of data into latch L2 is limited by the time at which latch L1 releases the data into the logic stage Logic1:

$$t_{D2LArr} = \max\left\{t_{D1LArr} + D_{1DQM}, T_L + D_{1CQM}\right\} + D_{L1M}. \quad (4.36)$$

Combining Eq. (4.35) and Eq. (4.36) we obtain:

$$W_1 - T_T - U_1 + D_{1DQM} + D_{L1M} \le P - T_L - U_2 \qquad (4.37a)$$
$$T_L + D_{1CQM} + D_{L1M} \le P - T_L - U_2. \qquad (4.37b)$$



## Example: Clocking the Alpha Processor

Rearanging Eq. (4.37a,b) we obtain a set of bounds for $W_2$ and $P$:

$$W_2 \ge U_2 - U_1 + D_{1DQM} + T_L - T_T + D_{L1M} \qquad (4.38a)$$
$$P \ge U_2 + D_{1CQM} + 2T_L + D_{L1M}. \qquad (4.38b)$$

Due to the symmetry of the clocking scheme, moving the reference point from the leading edge of the clock to the trailing edge of the clock will give us the same equations with indexes interchanged. To check this, start from the equation dual to Eq. (4.36):

$$t_{D1LArr} = \max\left\{t_{D2LArr} + D_{2DQM}, -W_2 + T_T + D_{2CQM}\right\} + D_{L2M}. \quad (4.39)$$

## Example: Clocking the Alpha Processor

Combining Eq. (4.34) and Eq. (4.39) and rearranging, we obtain a set of bounds for $W_1$ and $P$:

$$W_1 \geq U_1 - U_2 + D_{2DQM} + T_T - T_L + D_{L2M} \quad (4.40a)$$
$$P \geq U_1 + D_{2CQM} + 2T_T + D_{L2M} . \quad (4.40b)$$

Combining Eq. (4.38a) and Eq. (4.40a) we obtain a third and often the most critical bound for the clock period:

$$P = W_1 + W_2 \geq D_{1DQM} + D_{L1M} + D_{2DQM} + D_{L2M} . \quad (4.41)$$



## Example: Clocking the Alpha Processor

Substituting the values into Eqs. (4.38,4.40 and 4.41) we obtain:

$$W_2 \geq 270ps$$
$$P \geq 320ps$$
$$W_1 \geq 230ps$$
$$P \geq 270ps ,$$

and the most critical bound for $P$,

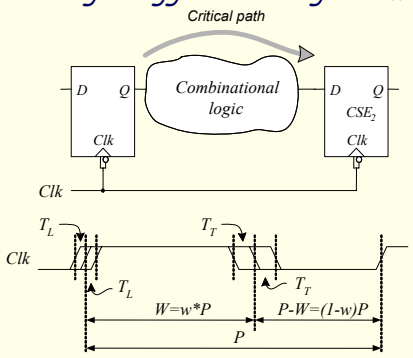
$$P = W_1 + W_2 \geq 500ps .$$

Thus the minimal clock period is $P_{min}$=500ps, and maximal frequency at which this system can run is $f_{max}$=2GHz.



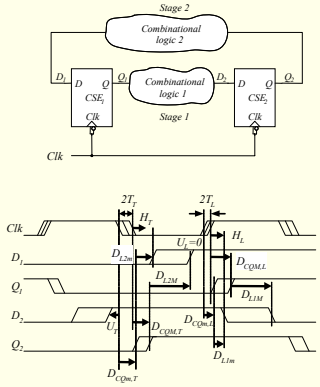
## Digital system using a single-phase clock and dual-edge triggered storage elements

## Digital system using a single-phase clock and dual-edge triggered storage elements



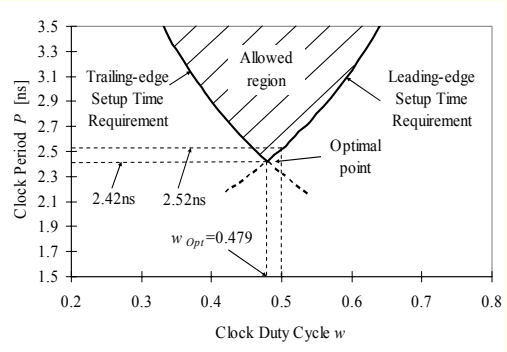



## Two-stage dual-edge-triggered system






## Allowed clock period as a function of the clock duty cycle in the dual-edge-triggered system

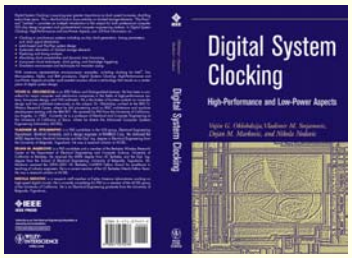


# Digital System Clocking:

*High-Performance and Low-Power Aspects*

Vojin G. Oklobdzija, Vladimir M. Stojanovic, Dejan M. Markovic, Nikola M. Nedovic

Chapter 5: High-Performance System Issues




Wiley-Interscience and IEEE Press, January 2003

# Absorbing Clock Uncertainties

- Clocking in high-performance digital systems is most seriously affected by *clock skew* and *clock jitter*.
- We will treat both of them as *clock uncertainties*.
- Trends:
  - Relative portion of the clock cycle budgeted for clock uncertainty increases.
  - Clock distribution becomes progressively difficult due to:
    - load mismatch
    - Process, voltage, and temperature variations.
  - The clock uncertainties occupy increasing portion of the cycle time; typically 2 FO4.
- The ability to reduce impact of these uncertainties is one of the most important properties of the high-performance system.

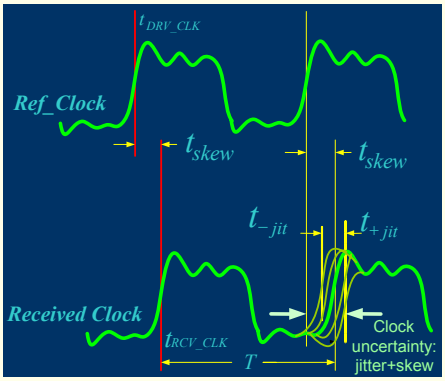


# Clock Generation and Distribution Non-idealities

- Jitter
  - *Jitter is a temporal variation of the clock signal manifested as uncertainty of **consecutive** edges of a periodic clock signal.*
  - It is caused by *temporal* noise events
  - Manifested as:
    - cycle-to-cycle or short-term jitter, $t_{JS}$
    - long-term jitter, $t_{JL}$
  - Mainly characteristic of clock generation system
- Skew
  - Is a time difference between temporally-equivalent or *concurrent edges* of two periodic signals
  - Caused by *spatial* variations in signal propagation
  - Manifests as SE-to-SE fluctuation of clock arrival at the same time instance
  - Characteristic of clock distribution system



# Clock Uncertainties

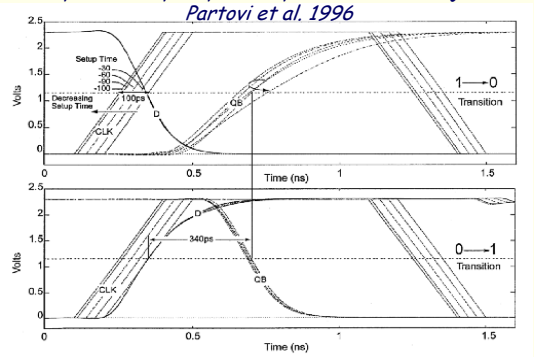





# Clock Uncertainty Absorption Using Soft Clock Edge



*Output of a flip-flop in the presence of clock jitter: Partovi et al. 1996*




A recent design of a Flip-Flop, controlled by a narrow, locally generated clock pulse, with negative Setup Time exhibits some degree of clock uncertainty absorption.

## Properties of Time-Window based Flip-Flops

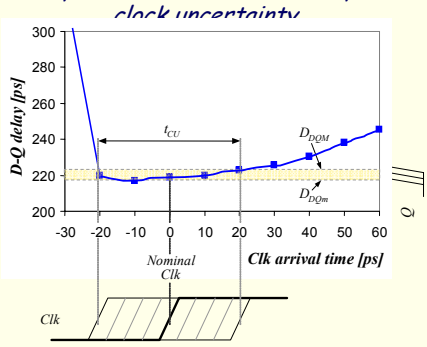
The relationship between the clock *Clk* and the output *Q*, in the presence of clock jitter, shows how the variation in the arrival time of the clock is somewhat absorbed by the Flip-Flop resulting in *smaller* variation of the time at which the output changes.

This behavior can be explained as follows:

- If the capturing pulse is sufficiently wide, the Flip-Flop exhibits short transparency to the data signal, resembling Latch behavior, and its timing is less sensitive to the clock arrival.
- This short transparency period is also known as the *soft clock edge*.
- With increase in importance of clock uncertainties, the practical use of the clocked storage element in high-performance systems will to a large extent depend on its ability to absorb them.



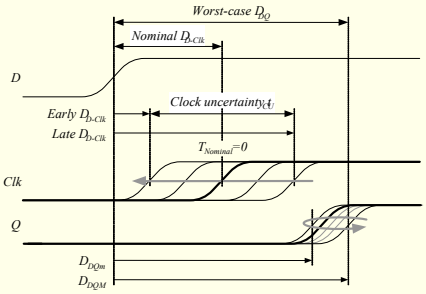
## Data-to-output characteristics in the presence of clock uncertainty




*Data-to-Output Delay* versus *Clock Arrival Time* when the data arrival time is constant. When no clock uncertainties are present, the clock is scheduled to arrive so that *D-Q* delay ($t_{DQm}$) is smallest, in order to minimize the CSE overhead.



## Dependence of data-to-output delay on clock arrival




The key role of a CSE is to minimize the propagation of clock uncertainty to the CSE output:

$$D_{DQM} = \max_{t} \; [D_{DQ}(U_{Opt}+t)], \; t \in [-t_{CU}/2, \; t_{CU}/2]$$

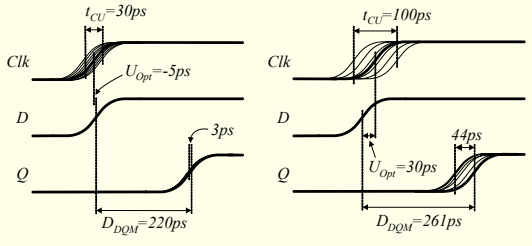


## Timing Analysis with Clock Uncertainty Absorption



## Total delay versus clock uncertainty

$$D_{DQm} + (1-\alpha_{CU})t_{CU} + D_{LM} \le P$$



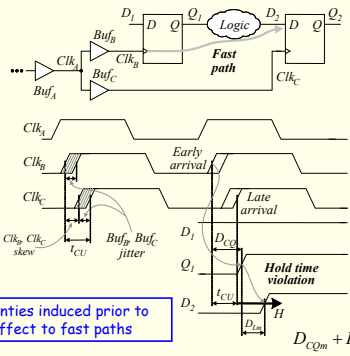

**(a)** $t_{CU}$=30ps ($a_{CU}$=90%) **(b)** $t_{CU}$=100ps ($a_{CU}$=56%)

We formulate *clock uncertainty absorption* $\alpha_{CU}$ of a storage element as the portion of the total clock uncertainty not reflected at the output:

$$\alpha_{CU} = \frac{t_{CU} - (D_{DQM} - D_{DQm})}{t_{CU}} = 1 - \frac{\Delta D_{DQ}}{t_{CU}}$$



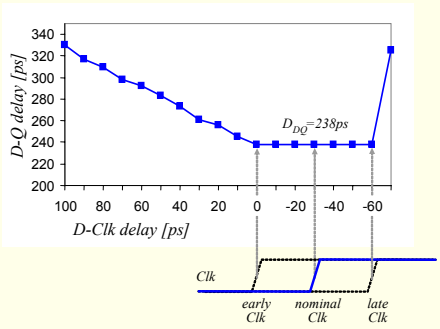
## Critical race in the presence of clock uncertainty




Clock uncertainties induced prior to $Clk_A$ have no effect to fast paths

$$D_{CQm} + D_{Lm} \ge H + t_{CU}$$

## *Idealized D-Q delay characteristic as a function of clock arrival*






# Time Borrowing



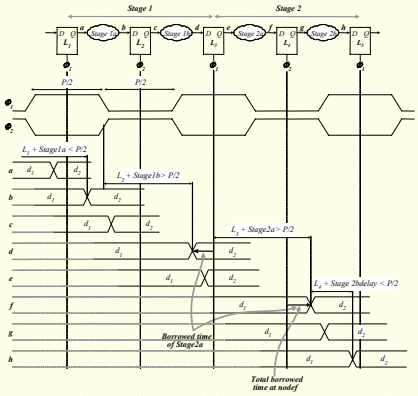
## Time Borrowing

We define time borrowing as:

- Dynamic time borrowing
  - the essential condition is that there are no "hard" boundaries between stages, i.e. that the storage elements are transparent at the time when data arrives.
  - Occurs in two clocking styles, level sensitive and soft-edge clocking.
- Static time borrowing
  - a technique of intentionally delaying the clock by inserting delay between clock inputs of the clocked storage elements. The clocks are scheduled to arrive so that the slower paths obtain more time to evaluate, taking away the time from faster paths.
  - It can operate with conventional hard-edge Flip-Flops.
  - Also called *opportunistic skew scheduling*



## *Timing of two-phase level-sensitive pipeline with time borrowing*






## *Timing Analysis with Time Borrowing:* ***Late Data Arrival***

*The arrival time of the input to the subsequent Latch $t_{D,i+1}$ is equal to the sum of the arrival times of the input to the preceding Latch $t_{D,i}$, Data-to-Output delay of the Latch $D_{DQ,i}$, and logic delay $D_{LM,i}$ of the current stage:*

$$t_{D,i+1} = t_{D,i} + D_{DQ,i} + D_{LM,i}$$

*The arrival of input to (2N+1)th Latch (input to (N+1)th stage) is:*

*(5.9)*

$$t_{D,2N+1} = t_{D,1} + \sum_{i=1}^{2N} (D_{DQ,i} + D_{LM,i})$$

*(Assumption: All logic blocks are used in time borrowing)*



## *Timing Analysis with Time Borrowing:* ***Late Data Arrival***

*We assume that the after N stages, the pipeline produces data at the same point in the transparency period of clock phase $\Phi_1$ at which the input data was acquired in the first clock cycle. Therefore, $t_{D,2N+1}$-$t_{D,1}$ is equal to N clock periods P.*

*(5.10)*

$$t_{D,2N+1} - t_{D,1} = NP$$

*Combining Eq. (5.9) and Eq. (5.10), we obtain the requirement for minimum clock period under late data arrival:*

*(5.11)*

$$P = \frac{1}{N} \sum_{i=1}^{2N} (D_{DQ,i} + D_{Logic,i})$$

*It shows that the minimum clock cycle time of the pipeline is not determined by the delay of the slowest stage in the pipeline. It is rather the average delay of the logic and Latches through all stages.*
*Note that Eq. (5.11) is valid only if the data arrive to the Latch during the time it is transparent.*

*(5.12)*

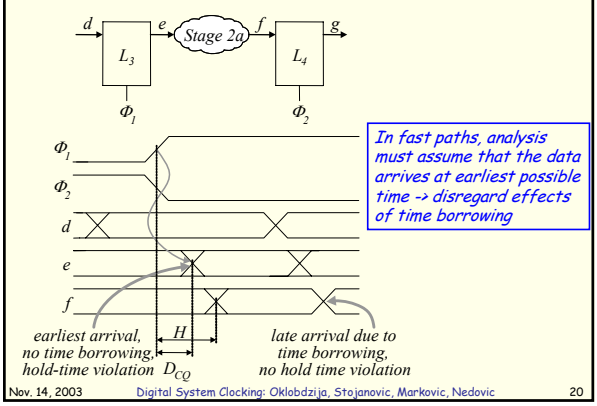
$$\frac{i-1}{2} P - D_{DQ,i} + D_{CQ,i} < t_{D,i} < \frac{i}{2} P - U_i, \quad \forall i \in [1..2N]$$

*where it is assumed that the first leading edge of $\Phi_1$ occurs at time zero.*

# Early Data Arrival



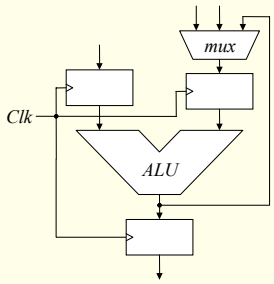
## Fast-path hazard




*In fast paths, analysis must assume that the data arrives at earliest possible time -> disregard effects of time borrowing*



## Time borrowing and signal loops



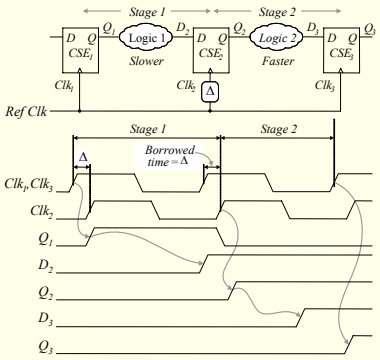

The timing of signals in the loops, should be treated separately.
If the overall propagation delay through the loop occur later with each cycle, it will result in a Setup Time violation.
Any signal loop that borrows time from itself will eventually cause a timing violation.



# Static Time Borrowing



## Opportunistic skew scheduling scheme






## Opportunistic skew scheduling scheme

Advantages:

- It can operate with conventional Flip-Flops.
- It places fewer constraints onto the circuit design, allowing longer critical paths where necessary.

Disadvantages:

- It increases the complexity of the clock distribution system.
- It is hard to control the inserted delays over process, supply and temperature variations.
- The analysis of clock skew is also complicated in this asymmetric clock distribution network.

While all these difficulties make this technique impractical on a large-scale level, it is very useful in localized critical paths where every improvement directly increases the system clock rate.

## Time Borrowing and Clock Uncertainty

Both clock uncertainty absorption and dynamic time borrowing use the property of the CSE to reduce effect of indeterminate Data-to-Clock delay to Data-to-Output delay:

- for clock uncertainty absorption, indeterminate Data-to-Clock delay occurs due to the uncertainty of clock arrival,
- for time borrowing it takes place due to uncertain data arrival.

In both cases, the transparency of the storage elements is used to suppress the input uncertainty (either that of clock or data).

The clock uncertainty absorption and time borrowing are essentially equivalent properties

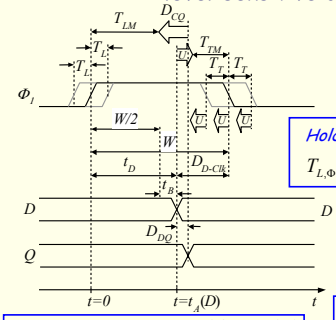
- If a clocking strategy allows dynamic time borrowing, it will also be capable of absorbing the clock uncertainty.
- Soft-clock edge property of the Flip-Flops, which is accountable for the clock uncertainty absorption, can also be used for time borrowing between the stages.



## Level-Sensitive Clocking



## *Clock uncertainty immunity in the pipeline with level-sensitive clocking*




*Hold time requirement (fast paths)*

$$T_{L,\Phi_1} + T_{T,\Phi_2} \le D_{CQm} + D_{Lm} - V - H$$

*Setup time requirement (slow paths)*

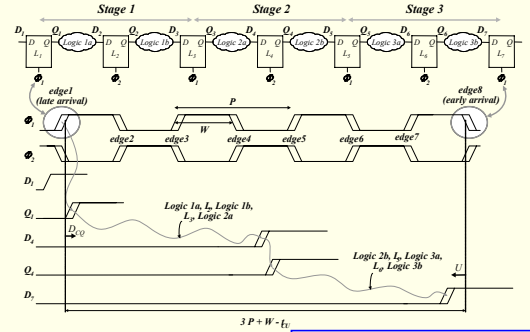
$$T_T \le W/2 - t_B - U$$

*To keep latch transparent when data arrives:*

$$T_L \le t_B + D_{DQ} - D_{CQ} + W/2$$



## *Impact of clock uncertainties on the critical path in the pipeline with time borrowing*




$$t_{CU} + D_{CQ1} + D(D_1 \to D_7) + U \le 3P + W \longrightarrow P \ge \frac{D_{CQ1} + D(D_1 \to D_7) + U - W}{3} + \frac{t_{CU}}{3}$$



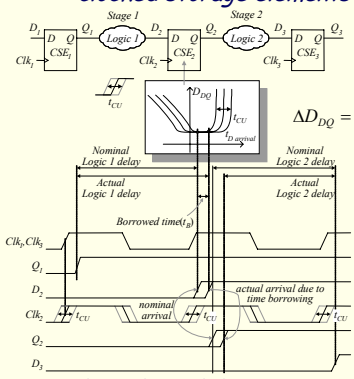
## *Effects of clock uncertainties to a time-borrowing system*

- Decreasing of the margins for time borrowing.
- The pipeline absorbs the uncertainties for the data that arrives during the transparency period of the Latch.
- The effect of the uncertainties is reduced to an average uncertainty over all stages in the path.



## Soft-Edge Sensitive Clocking

# Time borrowing with uncertainty-absorbing clocked storage elements



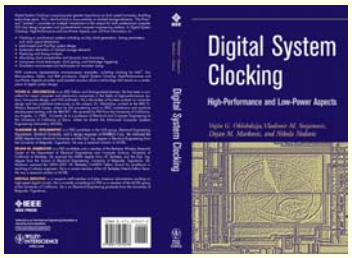

$$\Delta D_{DQ} = (t_B + t_{CU})[1 - \alpha_{CU}]$$

# Digital System Clocking:

## *High-Performance and Low-Power Aspects*

Vojin G. Oklobdzija, Vladimir M. Stojanovic, Dejan M. Markovic, Nikola M. Nedovic

Chapter 6: Low-Energy System Issues




Wiley-Interscience and IEEE Press, January 2003

## Switching Energy

*Clock is the highest activity signal ⇒ Switching energy is dominant*

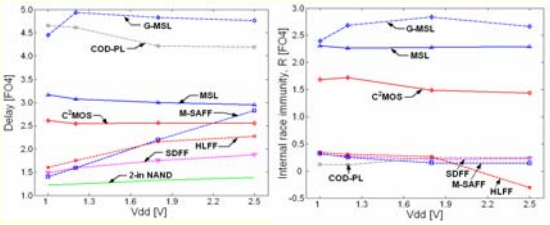
$$E_{switching} = \sum_{i=1}^{N} \alpha_{0-1}(i) \cdot C_i \cdot V_{swing}(i) \cdot V_{DD}$$

*$\alpha$- probability of a 0-1 transition*
*$C_i$ – total switched capacitance at node i*
*$V_{swing}(i)$ – voltage swing at node i*
*$V_{DD}$ – supply voltage*
*N – number of nodes*

*Energy is best reduced by scaling down $V_{DD}$, but this also means performance degradation*



## Pulsed Designs Scale the Best with $V_{DD}$




*(a)* *(b)*

(Delay is normalized to FO4 at its respective $V_{DD}$. FO4 as defined in this example increases with $V_{DD}$ ⇒ performance degradation)

*Impact of Vdd on (a) delay, and (b) internal race immunity (0.25 μm, light load). (Markovic et al. 2001), Copyright © 2001 IEEE*



## Other Energy Reduction Techniques

- Low-Swing Circuit Techniques
  - Reduced-swing Clk drivers
  - CSE redesign
  - N-only CSEs with Low-Vcc Clk
- Clock Gating
  - Global
  - Local
- Dual-Edge Triggering
  - Latch-mux
  - Pulsed-latch
  - Flip-flop

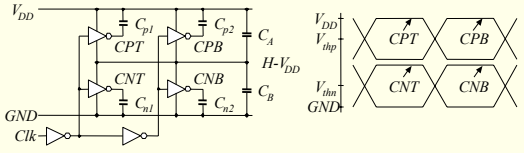


## Other Energy Reduction Techniques

- Low-Swing Circuit Techniques
  - Reduced-swing Clk drivers
  - CSE redesign
  - N-only CSEs with Low-Vcc Clk
- Clock Gating
  - Global
  - Local
- Dual-Edge Triggering
  - Latch-mux
  - Pulsed-latch
  - Flip-flop



## Low-Swing Clocking, Option #1: Clock Driver Re-design

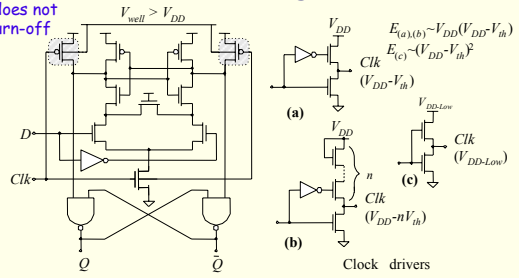



50% power reduction with half-swing clock
(minus some penalty in clock drivers)

*Clock driver for half-swing clocking (Kojima at al. 1995), Copyright © 1995 IEEE*

# Low-Swing Clocking, Option #2: CSE Re-design

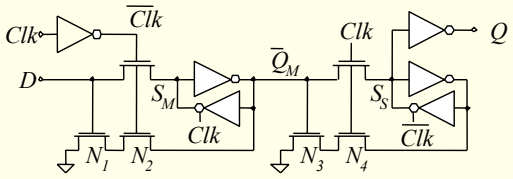
PMOS does not fully turn-off




*Reduced clock-swing flip-flop*
*(Kawaguchi and Sakurai, 1998), Copyright © 1998 IEEE*



# Low-Swing Clocking, Option #3: N-only CSEs




N-only clocked transistors, M-S Latch Example
($N_1$ and $N_2$ improve pull-up on $S_M$)

*N-Only clocked M-S latch*

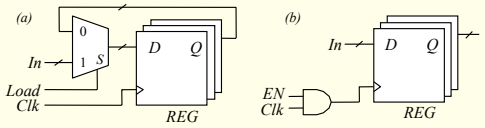


# Other Energy Reduction Techniques

- Low-Swing Circuit Techniques
  - Reduced-swing Clk drivers
  - CSE redesign
  - N-only CSEs with Low-Vcc Clk
- Clock Gating
  - Global
  - Local
- Dual-Edge Triggering
  - Latch-mux
  - Pulsed-latch
  - Flip-flop



# Clock Gating, Option #1: Global Clock Gating




Time-mux
(no gating!)

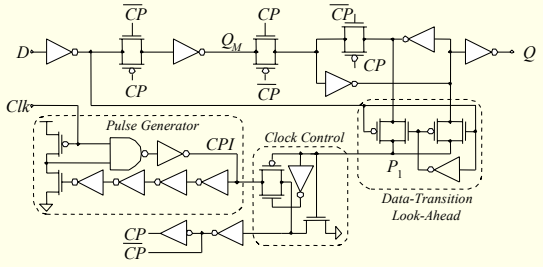
Global Clk Gating

Used to save clocking energy when data activity is low

*(a) Nongated clock circuit, (b) gated clock circuit.*
*(Kitahara et al. 1998), Copyright © 1998 IEEE*



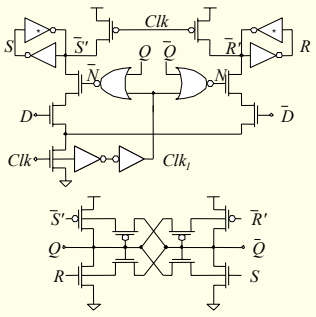
# Clock Gating, Option #2: Local Clock Gating




*Data-transition look-ahead latch*
*(Nogawa and Ohtomo, 1998), Copyright © 1998 IEEE*



# Local Clock Gating, Another Example



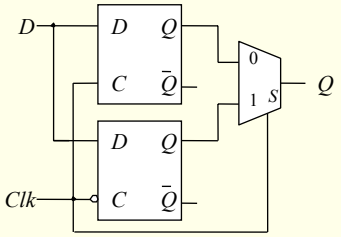

*Conditional capture flip-flop*
*(Kong et al. 2000), Copyright © 2000 IEEE*

## Other Energy Reduction Techniques

- Low-Swing Circuit Techniques
  - Reduced-swing Clk drivers
  - CSE redesign
  - N-only CSEs with Low-Vcc Clk
- Clock Gating
  - Global
  - Local
- Dual-Edge Triggering
  - Latch-mux
  - Pulsed-latch
  - Flip-flop



## Dual-Edge Triggering, Option #1: Latch-Mux

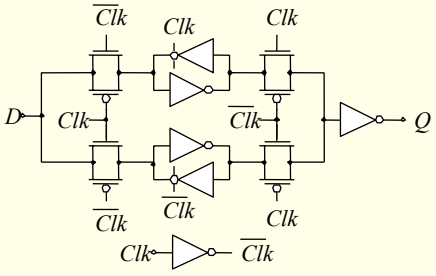



Used to save clocking energy regardless of data activity!

*Dual-edge-triggered latch-mux design*



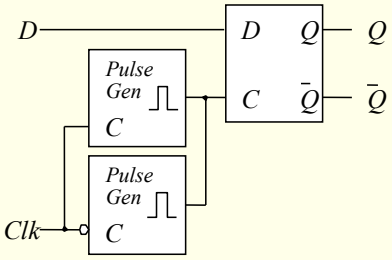
## DET Latch-Mux: Circuit Example




*Dual-edge-triggered latch-mux circuit (Llopis and Sachdev, 1996), Copyright © 1996 IEEE*



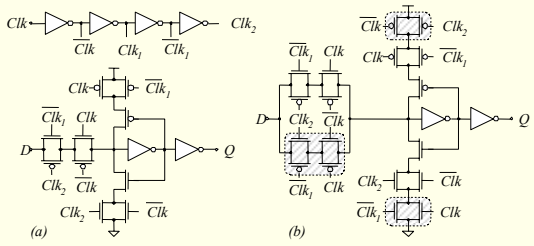
## Dual-Edge Triggering, Option #2: Pulsed-Latch




*Dual-edge-triggered pulsed-latch design*



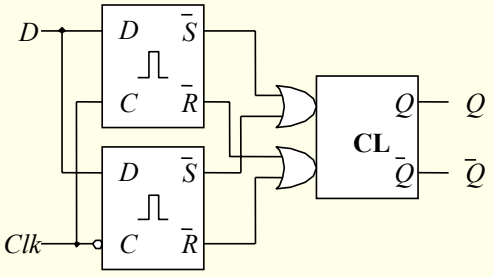
## DET Pulsed-Latch: Circuit Examples




*Pulsed-latch: (a) single-edge-triggered; (b) dual-edge-triggered*

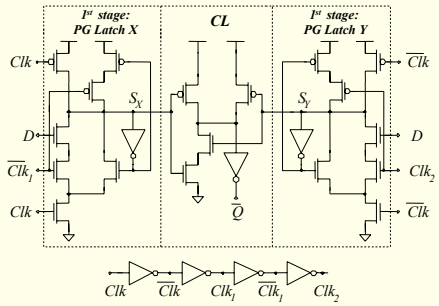


## Dual-Edge Triggered Flip-Flop




*Dual-edge-triggered flip-flop design*

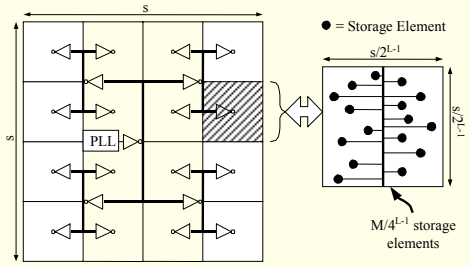
# DET Flip-Flop: Circuit Example




*DET symmetric pulse-generator flip-flop*



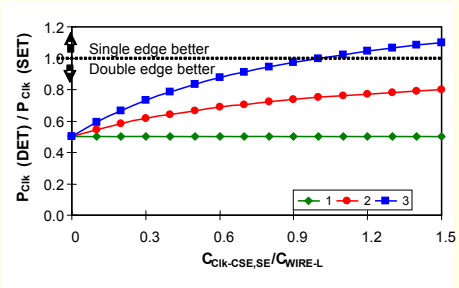
# Clock Distribution




*H-tree clock distribution network*



# Clocking Power: SET vs. DET



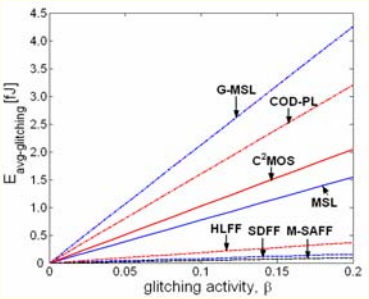

*Clocking power in single- and dual-edge-triggered systems*



# Glitch Robust Design



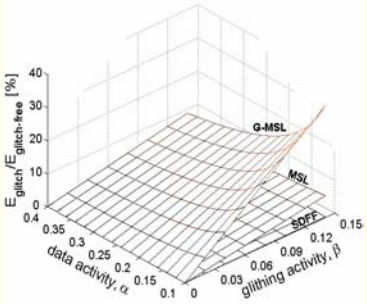
# Average Glitching Energy in CSEs




*Comparison of average glitching energy in CSEs (Markovic at al. 2001), Copyright © 2001 IEEE*



# Comparison of $E_{glitching}$ and $E_{switching}$




*Glitching energy as a percentage of switching energy in representative CSEs showing the greatest glitch sensitivity of the gated designs*

## Summary

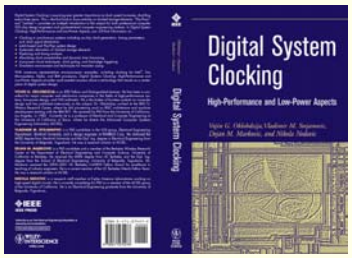
- Energy best reduced by $V_{DD}$ scaling
  - Penalty in performance
- Reducing Clk swing only reduces $E_{Clk}$
  - Still penalty in performance
- Clock gating
  - Reduces $E_{Clk}$ at low-activity
  - No penalty in perf. if gating is outside crit-path
- Dual-Edge Triggering
  - Reduces $E_{Clk}$ ideally by 2x
  - Small or no performance degradation

# Digital System Clocking:

*High-Performance and Low-Power Aspects*

Vojin G. Oklobdzija, Vladimir M. Stojanovic, Dejan M. Markovic, Nikola M. Nedovic

Chapter 7: Simulation Techniques




Wiley-Interscience and IEEE Press, January 2003

## Simulation Techniques

- Results and conclusions about the performance of different CSEs depend significantly on the simulation setup and evaluation environment.
- CSE is just one of the elements in the pipeline, and has to be sized in such a way as to achieve optimum performance for given output load.
- The CSE output loads vary a lot across the processor core, depending on the level of parallelism in each unit and also whether the CSE is on the critical path or not.
- In modern data-paths CSEs experience heavy load due to the parallel execution units and increase in interconnect capacitance.
- It is the performance of these CSEs, on the critical path, that has the highest impact on the choice of the processor cycle time.



## Simulation Techniques

- In high-speed designs, the design and evaluation of CSEs is focused on the elements on the critical path and often implicitly assumes such conditions during performance comparisons.
- There are a lot of CSEs that are placed on non-critical paths with relatively light load. While these CSEs do not directly impact the performance of the processor, careful design of these elements can significantly reduce the energy consumption and alleviate the clock distribution problems.
- The purpose of this presentation is to recommend simulation techniques that designers can use to evaluate the performance of CSEs, depending on the desired application.
- We try to build the understanding of the issues involved in creating a simulation environment of the CSE, so that such information can be used to build own setups tailored to the specific application.
- There is no universal setup that is good for all CSE applications.



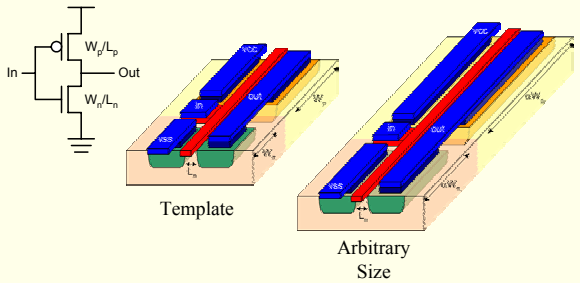
## The Method of Logical Effort



# Logical Effort Basics

Contribution of Bart Zeydel, PhD candidate
ACSEL group
University of California
Davis

## Logical Effort Components




- Input Capacitance increases by $\alpha \cdot C_{template}$
- Resistance decreases by $R_{template} / \alpha$

# Logical Effort Input Capacitance

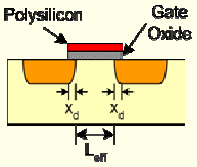
$C_{ox} = \varepsilon_{ox} / t_{ox}$ $L_{eff} = L - 2x_d$

$C_{gate} = C_{ox} W L_{eff} = C_{ox} W L (1 - 2x_d / L)$

$\kappa_1 = C_{ox}(1 - 2x_d / L)$

$C_{template(inv)} = \kappa_1 W_n L_n + \kappa_1 W_p L_p$

$C_{in(inv)} = \alpha \cdot C_{template(inv)}$




- Input Capacitance is the sum of the gate capacitances
- Using LE, $\alpha$ denotes size in multiples of the template



# Logical Effort Resistance

$R = (\rho / t)\ (L / W)$ ohms

$R = R_{sheet}\ (L / W)$ ohms

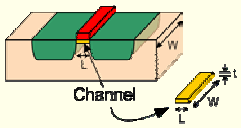
$R_{channel} = R_{sheet}\ (L / W)$ ohms

$R_{sheet} = 1 / (\mu C_{ox} (V_{gs} - V_t))$ ohms

$\kappa_2 = C_{ox} (V_{gs} - V_t)$ $\mu$ = surface mobility

$R_{channel} = L / (\kappa_2 \mu W)$ ohms

$R_{up(inv)} = R_{up\text{-}template(inv)} / \alpha$ $R_{down(inv)} = R_{down\text{-}template(inv)} / \alpha$

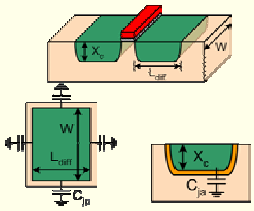



- Resistance is dependent on transistor size and process
- Larger values of $\alpha$ result in lower resistance



# Logical Effort Parasitics

| | |
|---|---|
| $C_{ja}$ | junction capacitance per $\mu^2$ |
| $C_{jp}$ | periphery capacitance per $\mu$ |
| W | width of diffusion region $\mu$ |
| $L_{diff}$ | length of diffusion region $\mu$ |

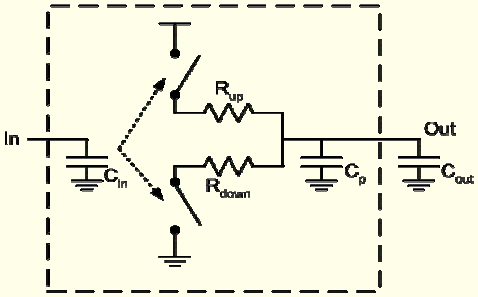



$C_j = C_{ja} \cdot W \cdot L_{diff} + C_{jp} \cdot (2 \cdot W + 2 \cdot L_{diff})$

- Larger values of $\alpha$ result in increased parasitic capacitance, however R decreases at same rate, thus constant RC delay



# RC Model for CMOS logic gate






# LE Delay derivation for step input

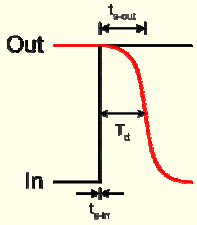
$$I_{sat} = \mu \cdot C_{ox} \cdot \frac{W}{L} \frac{(V_{GS} - V_t)^2}{2}$$

$$I_{linear} = \mu \cdot C_{ox} \cdot \frac{W}{L} \left( (V_{GS} - V_t) \cdot V_{out} - \frac{V_{out}^2}{2} \right)$$

$$t_{s-in} = 0$$

$$t_D = t_{s-out(sat)} + t_{s-out(linear)}$$

$$t_{f-out} = \int_{Vdd-Vt}^{Vdd} \frac{C_{out}}{I_{sat}} \cdot dV + \int_{Vdd/2}^{Vdd-Vt} \frac{C_{out}}{I_{linear}} \cdot dV$$






# LE Model derivation (cont.)

$$t_{f-out(sat)} = \frac{C_{out}}{\mu \cdot C_{ox} \cdot \frac{W}{L} \frac{(V_{DD} - V_t)^2}{2}} \int_{Vdd-Vt}^{Vdd} dV_{out} = \frac{2 C_{out} V_t}{\mu \cdot C_{ox} \cdot \frac{W}{L} (V_{DD} - V_t)^2}$$

$$t_{f-out(linear)} = \frac{C_{out}}{\mu \cdot C_{ox} \cdot \frac{W}{L}} \int_{Vdd-Vt}^{Vdd} \frac{1}{\left( (V_{DD} - V_t) \cdot V_{OUT} - \frac{V_{OUT}^2}{2} \right)} dV_{out}$$

$$t_{f-out} = \frac{C_{out}}{\mu \cdot C_{ox} \cdot \frac{W}{L} (V_{DD} - V_t)} \left( \frac{2 \cdot V_t}{V_{DD} - V_t} + \ln \cdot \frac{4 \cdot (V_{DD} - V_t) - V_{DD}}{V_{DD}} \right)$$

Substituting in RC to obtain

$$t_D = R \cdot C_{out} \cdot \underbrace{\left( \frac{2 \cdot V_t}{V_{DD} - V_t} + \ln\left( 3 - 4 \frac{V_t}{V_{DD}} \right) \right)}_{\kappa}$$

## Logical Effort Gate Delay Model

$$d = \kappa \cdot R_{in}(C_{out} + C_{parasitic})$$

$$d = \kappa \cdot \left(\frac{R_{template}}{\alpha}\right) \cdot C_{in} \cdot \left(\frac{C_{out}}{C_{in}}\right) + \kappa \cdot \left(\frac{R_{template}}{\alpha}\right) \cdot (\alpha \cdot C_{parasitic})$$

$$d = (\kappa \cdot R_{template} \cdot C_{template})\left(\frac{C_{out}}{C_{in}}\right) + \kappa \cdot R_{template} \cdot C_{parasitic}$$

$$\tau = \kappa \cdot R_{inv} \cdot C_{inv}$$

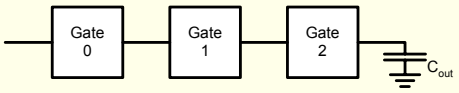
$$g = \frac{R_{template} C_{template}}{R_{inv} C_{inv}} \qquad h = \frac{C_{out}}{C_{in}} \qquad p = \frac{R_{template} C_{parasitic}}{R_{inv} C_{inv}}$$

$$\boxed{d = \tau \cdot (g \cdot h + p)}$$

Harris, Sutherland, Sproull



## LE Path Delay




| | | | |
|---|---|---|---|
| Input Capacitance: | $C_{in}$ | $C_1$ | $C_2$ |
| Logical Effort: | $g_0$ | $g_1$ | $g_2$ |
| Parasitic Delay: | $p_0$ | $p_1$ | $p_2$ |
| Stage Effort: | $f_0$ | $f_1$ | $f_2$ |

$$D = [(g_0 h_0 + p_0) + (g_1 h_1 + p_1) + (g_2 h_2 + p_2)] \cdot \tau$$

- Any n-stage path can be described using Logical Effort



## LE Path Delay Optimization

$$D = [(g_0 h_0 + p_0) + (g_1 h_1 + p_1) + (g_2 h_2 + p_2)] \cdot \tau$$

$$h_0 = \frac{C_1}{C_{in}} \qquad h_1 = \frac{C_2}{C_1} \qquad h_2 = \frac{C_{out}}{C_2}$$

$$h_0 h_1 h_2 = \frac{C_1}{C_{in}} \cdot \frac{C_2}{C_1} \cdot \frac{C_{out}}{C_2} = \frac{C_{out}}{C_{in}} = H$$

$$D = [(g_0 \frac{C_1}{C_{in}} + p_0) + (g_1 \frac{C_2}{C_1} + p_1) + (g_2 \frac{C_{out}}{C_2} + p_2)] \cdot \tau$$



## LE Path Delay Optimization (cont.)

By Definition, $C_{in}$ and $C_{out}$ are fixed.

$$D = [(g_0 \frac{C_1}{C_{in}} + p_0) + (g_1 \frac{C_2}{C_1} + p_1) + (g_2 \frac{C_{out}}{C_2} + p_2)] \cdot \tau$$

Solve for $C_1$ and $C_2$:

$$\frac{\partial D}{\partial C_1} = \frac{g_0}{C_{in}} - g_1 \frac{C_2}{C_1^2} = 0 \qquad \frac{\partial D}{\partial C_2} = \frac{g_1}{C_1} - g_2 \frac{C_{out}}{C_2^2} = 0$$

$$g_0 \frac{C_1}{C_{in}} = g_1 \frac{C_2}{C_1} \qquad g_1 \frac{C_2}{C_1} = g_2 \frac{C_{out}}{C_2}$$

$$f_0 = f_1 \qquad f_1 = f_2$$

Minimum occurs when stage efforts are equal

$$\boxed{f_0 = f_1 = f_2}$$



## Simplified Path Optimization

We want the effort of each stages to be equal.

$$F = f_0 \cdot f_1 \cdot f_2 \ldots f_{n-1}$$

$$\hat{f} = F^{1/n}$$

To quickly solve for F:

$$F = g_0 h_0 \cdot g_1 h_1 \cdot g_2 h_2 \ldots g_{n-1} h_{n-1}$$

$$H = h_0 \cdot h_1 \cdot h_2 \ldots h_{n-1} = \frac{C_{out}}{C_{in}}$$

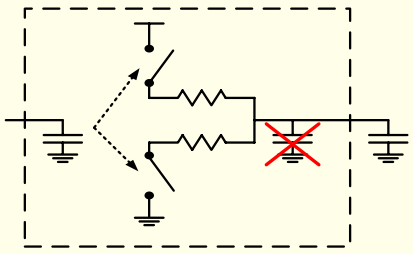
$$G = g_0 \cdot g_1 \cdot g_2 \ldots g_{n-1}$$

$$\boxed{F = G \cdot H}$$

Harris, Sutherland, Sproull

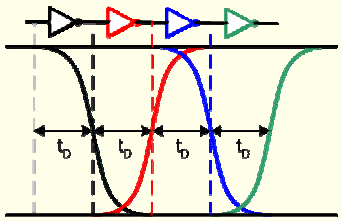


## LE Delay Model for Path Optimization



- Input slope effects on g and p are ignored
- Parasitics assumed *independent* of input slope
  - Therefore have no effect on optimization result

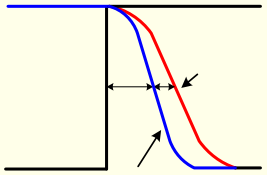
## LE Slopes Assumed for Path Optimization




- Slopes are drawn for each gate assuming step input
- Slopes are equal for each gate at completion of optimization
  - $f_i$ is constant, thus constant RC (ignoring parasitics).



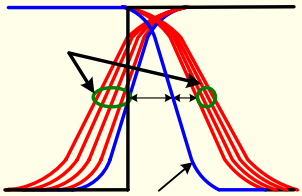
## LE Optimization Assumptions



- gh is constant when using LE optimization
- Slope variations are due to differences in p and are **not** accounted for in optimization



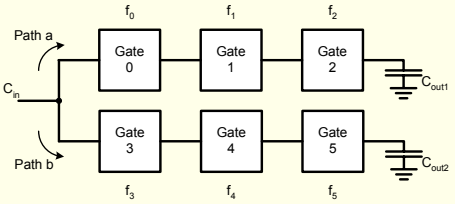
## LE Slope Variation Effects



- Input slope degradation causes output slope degradation
- Different gate types also cause output slope variation



## Logical Effort for Multi-path




From LE,

$$f_0 = f_1 = f_2 \quad \text{and} \quad f_3 = f_4 = f_5$$

Minimum delay occurs when $D_a = D_b$ or $F_a = F_b$ (ignoring parasitics)

$$f_0 = f_1 = f_2 = f_3 = f_4 = f_5$$



## Logical Effort for Multi-path (cont.)

$$D_a = [(g_0h_0 + p_0) + (g_1h_1 + p_1) + (g_2h_2 + p_2)] \cdot \tau$$

$$D_b = [(g_3h_3 + p_3) + (g_4h_4 + p_4) + (g_5h_5 + p_5)] \cdot \tau$$

$$f_0f_1f_2 = g_0g_1g_2 \cdot \frac{C_{out1}}{C_0} \qquad f_3f_4f_5 = g_3g_4g_5 \cdot \frac{C_{out2}}{C_3}$$

$$C_0 = \frac{g_0g_1g_2 \cdot C_{out1}}{f_0f_1f_2} \qquad C_3 = \frac{g_3g_4g_5 \cdot C_{out2}}{f_3f_4f_5}$$

**Branching:** Ratio of total capacitance to on-path capacitance

$$b_a = \frac{C_o + C_3}{C_0} \qquad b_b = \frac{C_3 + C_0}{C_3}$$



## Logical Effort for Multi-path (cont)

Substituting $C_0$ and $C_3$

$$b_a = \frac{\frac{g_0g_1g_2 \cdot C_{out1}}{f_0f_1f_2} + \frac{g_3g_4g_5 \cdot C_{out2}}{f_3f_4f_5}}{\frac{g_0g_1g_2 \cdot C_{out1}}{f_0f_1f_2}}$$

Assume Minimum Delay occurs when $f_0f_1f_2 = f_3f_4f_5$

$$b_a = \frac{g_0g_1g_2 \cdot C_{out1} + g_3g_4g_5 \cdot C_{out2}}{g_0g_1g_2 \cdot C_{out1}}$$

Similarly

$$b_b = \frac{g_3g_4g_5 \cdot C_{out2} + g_0g_1g_2 \cdot C_{out1}}{g_3g_4g_5 \cdot C_{out2}}$$

- Branching = 2 when $G_a = G_b$ and $C_{out1} = C_{out2}$



Out

Unaccounted

# Complex Multi-path Optimization

If each path has internal branching, $b_a$ and $b_b$ are as follows

$$b_a = \frac{g_0 b_0 g_1 b_1 g_2 b_2 \cdot C_{out1} + g_3 b_3 g_4 b_4 g_5 b_5 \cdot C_{out2}}{g_0 b_0 g_1 b_1 g_2 b_2 \cdot C_{out1}}$$

$$b_b = \frac{g_3 b_3 g_4 b_4 g_5 b_5 \cdot C_{out2} + g_0 b_0 g_1 b_1 g_2 b_2 \cdot C_{out1}}{g_3 b_3 g_4 b_4 g_5 b_5 \cdot C_{out2}}$$

- Note: This solution and previous solution differ from that are described in LE book (all simplify by ignoring parasitics)

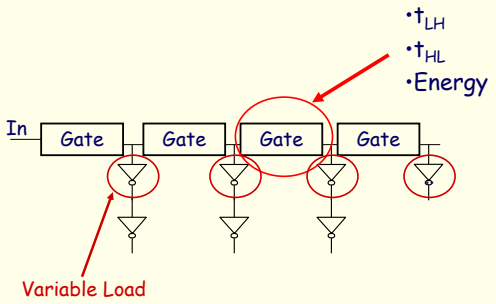


# Slopes in LE

- LE optimization assumes step input for gate
  - Hence, slope variations are not accounted for.
- Spice characterization is invalid for step input derivation of Logical Effort, as g and p relate to non-practical values.
- Fortunately Logical Effort can be derived assuming equal slopes if we assume no g and p dependence on input slope variation. These "improved" g and p values can be obtained from spice characterization

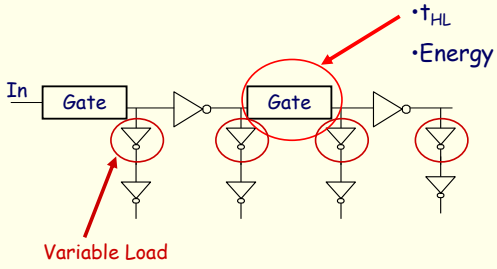


# LE Characterization Setup for Static Gates





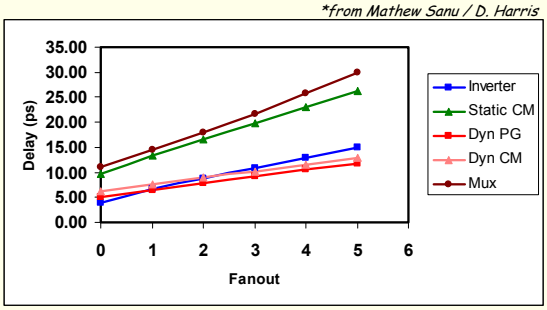

# LE Characterization Setup for Dynamic Gates






# L.E for Adder Gates

**from Mathew Sanu / D. Harris*




- Logical effort parameters obtained from simulation for std cells
- Define logical effort 'g' of inverter = 1
- Delay of complex gates can be defined with respect to **d=1**



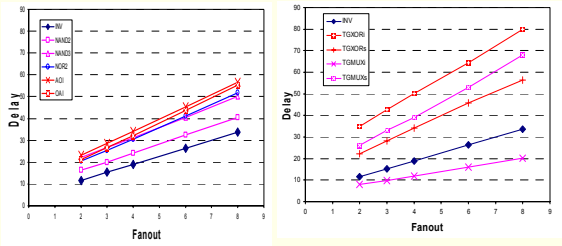
# Normalized L.E

**from Mathew Sanu*

| Gate type | Logical Eff. (g) | Parasitics (Pinv) |
|---|---|---|
| Inverter | 1 | 1 |
| Dyn. Nand | 0.6 | 1.34 |
| Dyn. CM | 0.6 | 1.62 |
| Dyn. CM-4N | 1 | 3.71 |
| Static CM | 1.48 | 2.53 |
| Mux | 1.68 | 2.93 |
| XOR | 1.69 | 2.97 |

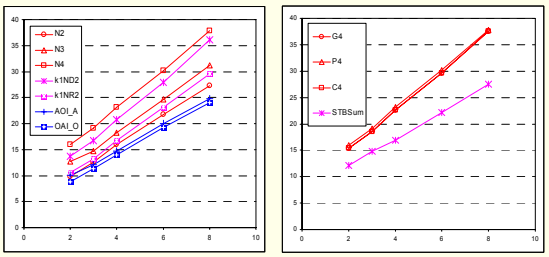
- Logical effort & parasitic delay normalized to that of inverter

# Static CMOS Gates: Delay Graphs





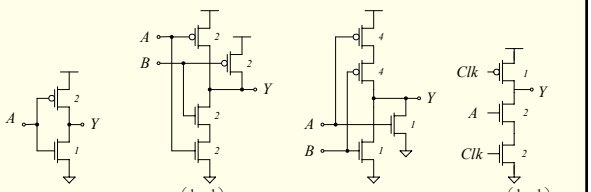

# Dynamic CMOS: Delay Graphs






# *Simple logic gates:*

*(a) reference static inverter, (b) two-input static NAND gate, (c) two-input static NOR gate, (d) domino-style inverter*




$RC_{A,inv} = 1\cdot(2+1) = 3$ $RC_{A,nand2} = \left(\frac{1}{2}+\frac{1}{2}\right)\cdot(2+2) = 4$ $RC_{A,nor2} = 1\cdot(4+1) = 5$ $RC_{A,Dom-inv} = \left(\frac{1}{2}+\frac{1}{2}\right)\cdot 2 = 2$

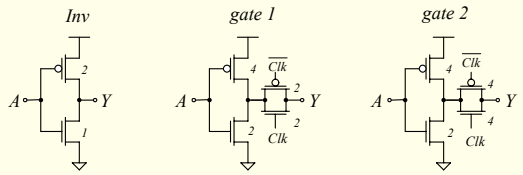
$g_{A,inv} = \frac{RC_{A,inv}}{RC_{A,inv}} = 1$ $g_{A,nand2} = \frac{RC_{A,nand2}}{RC_{A,inv}} = \frac{4}{3}$ $g_{A,nor2} = \frac{RC_{A,nor2}}{RC_{A,inv}} = \frac{5}{3}$ $g_{A,Dom-inv} = \frac{RC_{A,Dom-inv}}{RC_{A,inv}} = \frac{2}{3}$

*(a)* *(b)* *(c)* *(d)*



# *Logical effort of gate driving a transmission gate:*

*(a) reference inverter, (b) slow-data input, (c) fast-data input sizing*




$RC_{A,inv} = 1\cdot(2+1) = 3$ $RC_{A,gate1} = \left(\frac{1}{2}+\frac{1}{2}\right)\cdot(4+2) = 6$ $RC_{A,gate2} = \left(\frac{1}{2}+\frac{1}{4}\right)\cdot(4+2) = \frac{9}{2}$

$g_{A,inv} = \frac{RC_{A,inv}}{RC_{A,inv}} = 1$ $g_{A,gate1} = \frac{RC_{A,gate1}}{RC_{A,inv}} = 2$ $g_{A,gate2} = \frac{RC_{A,gate2}}{RC_{A,inv}} = \frac{3}{2}$

*(a)* *(b)* *(c)*



# Environment Setup



# Environment Setup

Several studies used different simulation setups. We present some of the important concepts and global simulation setup framework that can be further tuned for the particular intended application:

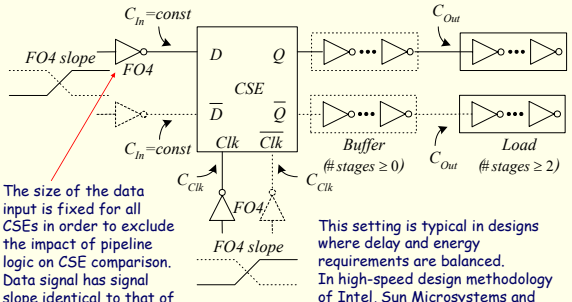
- The setup for the comparison of the CSEs in (Stojanovic and Oklobdzija 1999) used a single size load.
  - Load was chosen that resembles the typical situation in a moderately-to-heavy loaded critical path in a processor with a lot of parallelism. All the CSEs were sized such as to achieve optimum data-to-output delay for given output load, while driven from the fixed size inverters.
- In most practical situations the CSEs are designed in a discrete set of sizes, each optimized for a particular load. It is useful to examine the performance of the CSE for a range of the loads around the load for which the CSE was optimized. This technique is illustrated in (Nikolic and Oklobdzija 1999)
  - Different CSEs are initially sized to drive a fixed load, and the load is then varied.
  - The delay of a CSE exhibit linear dependence on the load, with the slope of the delay curve illustrating the logical effort of the driving stage of the CSE and the zero-load crossing illustrating the parasitic delay of the driving stage together with the delay of the inner stages of the CSE.
  - This is not the optimal behavior of the CSE delay curve, but is the best that can be achieved when there are only a few CSE sizes available in the library.

# Environment Setup

- In case where CSE can be re-optimized for each particular load, further speedup can be achieved since the effort can be shared between stages rather than resting solely on the output stage. This approach was illustrated in (Heo and Asanovic 2001).
  - Contrary to their conclusions, in case when a general performance of a CSE needs to be assessed, the proper approach is to optimize the CSE for the most important application that determines the performance of the whole system, not the most frequent application.
- In high-speed systems the most important are the elements on the critical path which is typically moderately-to-heavy loaded due to branching to parallel execution units and wire capacitance. The small number of critical paths in a processor does not decrease their importance since it is their delay that determines the clock rate of the whole system.
- The performance of the large number of the lightly loaded CSEs that are placed off of the critical path is of concern only if it can be traded for energy savings.
- The simulation approach should attempt to resemble the actual data-path environment. The number of logic stages in a CSE and their complexity highly depend on particular circuit implementation, leading to differences in logical effort, parasitic delay and energy consumption.
- Every CSE structure needs to be optimized to drive the load with best possible effort delay.



# General simulation setup




The size of the data input is fixed for all CSEs in order to exclude the impact of pipeline logic on CSE comparison. Data signal has signal slope identical to that of a FO4 inverter, which is the case in a well-designed pipeline.

This setting is typical in designs where delay and energy requirements are balanced. In high-speed design methodology of Intel, Sun Microsystems and former Digital Corp, FO3 inverter metric is more common than FO4 due to more aggressive design style.



# General simulation setup

- The size of the clocked transistors is set to the size needed in order not to compromise the speed of the whole structure.
- A direct tradeoff exists between the CSE delay and clock energy (size of clocked transistors), as some of the clocked transistors are always on the critical path of the CSE.
- Increase in sizes of clocked transistors on a critical path results in diminishing returns since data input is fixed. Depending on the CSE topology, some structures can trade delay for clocked transistor size more efficiently than others, so we allow this to happen up to a certain extent.
- Our goal is to examine CSEs that are used on a critical path, hence the assumption that designer might be willing to spend a bit more clock power to achieve better performance.
- Differences in clock loads ($C_{Clk}$) among devices illustrate potential drawbacks in terms of clock power requirements, and serve as one of the performance metrics. Clock inputs have identical signal slope to that of a FO4 inverter. This can be changed depending on the clock distribution design methodology.



# General simulation setup

The question on how to compare differential and single-ended structures has always been one of the key issues among the people characterizing and designing the CSEs.

- Differential and single-ended structures should not be compared with each other, due to the overhead that single-ended structures incur to generate the complementary output.
- We do not require that single-ended structure generate both true and complementary value at the output.

- The worst-case analysis requires that the CSE generates the output that has worse data-to-output delay. However, it is also beneficial to measure both *D-Q* and *D-*$\overline{Q}$ delay.

- Any imbalance between the two can lead to big delay savings in cases where proper logic polarity manipulation in the stages preceding or following the CSE can change the polarity requirement of the CSE, and hence its data-to-output delay.
- Load model always consists of several inverters in a chain to avoid the error in delay caused by Miller capacitance effects from the fast switching load back to the driver.

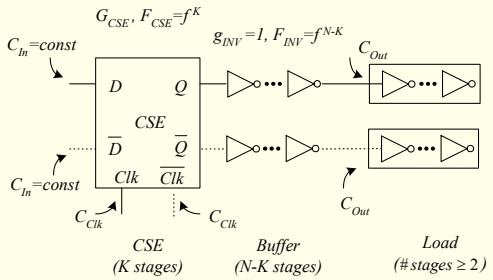


# General simulation setup

- The logical effort framework offers analogy between the CSE and a simple logic gate.
  - At light load, logic gate is dominated by its parasitic delay, i.e. self-loading.
  - At high load, effort delay becomes the dominant factor.
- Similarly, at light load, delay of a CSE with large number of stages is entirely dominated by parasitic delay.
- However, at high load, more stages are beneficial in reducing the effort delay, which then dominates over parasitic delay.
- Therefore, the performance of the CSE is best assessed if it is evaluated in a range of output loads of interest for the particular application.
- CSE evaluation can either be performed using some representative critical path load or a set of loads can be used in which case CSE has to be re-optimized for each load setting.



# Additional buffering in simulation test bed




Depending on the choice of the output load size, some CSE structures with inherently small number of stages and high logical effort may require additional buffering in order to achieve the best effort delay.

## Additional buffering in simulation test bed

For each CSE we need to find optimal effort per stage and number of stages to drive the required load. Starting from total electrical fan-out *H*, optimal number of stages *N* is obtained through rounding of the logarithm of the total effort of the path (assuming $g_{INV}$ is unity):

$$H = \frac{C_{OUT}}{C_{IN}} \qquad N = round\left(\log_4\left(G_{CSE}H\right)\right)$$

The logarithm is of base 4 since stage effort of 4 is a target for optimal speed. Once the integer number of stages is obtained, the updated value of stage effort is found from:

$$f = \sqrt[N]{G_{CSE}H}$$

After the stage effort is obtained, CSE internal stages have to be resized for new stage effort and also the external inverters, if there are any.
This sizing approach is optimal even in case no additional inverters are required, since it will serve to distribute the effort between the internal stages of the CSE.



## HLFF Sizing Example



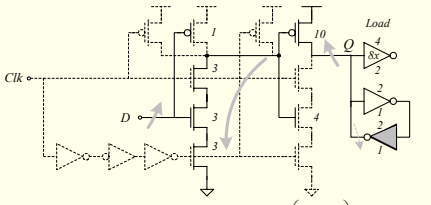
## HLFF Sizing Example

In this problem we observe the change in minimum data-to-output (*D-Q* or *D-* $\overline{Q}$) delay as the output load of the CSE increases:

- Before we start investigating the effect of different loads on the sizing of the HLFF, we show how the logical effort can be calculated for the given sizing.
- It is relatively easy to recognize that the HLFF is built up of three-input static NAND gate as the first stage and domino-like three-input NAND in the second stage.
- Minor variations from standard static NAND sizing for equal logical effort on all inputs are needed to speed up the data input and enable the first stage to evaluate before the transparency window closes. Similar situation occurs in the second stage.
- This HLFF sizing example also illustrates the application of logical effort to skewed gates (gates in which one output transition is faster than the other), and gates with keepers.



## HLFF sizing example




$$g_{1,0\to1} = \frac{\left(3\cdot\frac{1}{3}\right)\cdot(1+3)}{1\cdot(1+2)} = \frac{4}{3} \qquad g_{2,1\to0} = \frac{\left(\frac{1}{10/2-1}\right)\cdot(4+10)}{1\cdot(1+2)} = \frac{7}{6}$$

$$f_{1,0\to1} = \frac{4}{3}\cdot\frac{4+10}{3+1} = 4.7 \qquad f_{2,1\to0} = \frac{7}{6}\cdot\frac{8\cdot(2+4)+(1+2)}{4+10} = 4.25$$

The critical path of the HLFF is exercised with a "0-to-1" transition at data input. The first stage of the HLFF is a skewed NAND gate.

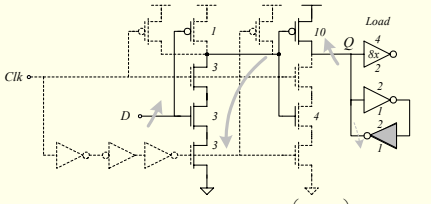


## HLFF sizing example

- The logical effort calculation of the second stage is slightly more complicated because of the keeper inverter pair. A keeper sinks a portion of the current that is sourced by the PMOS transistor to node *Q* which can be taken into account as negative conductance.
- This negative conductance accounted for by subtracting the conductance of the NMOS transistor (1) of the shaded keeper inverter, from the conductance of the driving PMOS transistor (10/2).
- For the particular load, efforts per stage were calculated to be 4.7 and 4.25, which is near the optimum value of 4, indicating that this example sizing is nearly optimal.
- The sizing in example above is somewhat simplified because the short channel stack effect has not been taken into account, the logical effort values for the NMOS transistor stack are somewhat pessimistic.
- Once the logical effort of each stage is known it can be used to adjust the sizing of each stage as the load is increased or decreased.
- The alternative method is to use one of the automated circuit optimizers, however, it is not recommend it as initial method, simply because it is essential that designer gets to know the circuit through manual sizing and logical effort estimation.
- This builds intuition about the circuit and ability to verify optimizer results.



## HLFF sizing example




$$g_{1,0\to1} = \frac{\left(3\cdot\frac{1}{3}\right)\cdot(1+3)}{1\cdot(1+2)} = \frac{4}{3} \qquad g_{2,1\to0} = \frac{\left(\frac{1}{10/2-1}\right)\cdot(4+10)}{1\cdot(1+2)} = \frac{7}{6}$$

$$f_{1,0\to1} = \frac{4}{3}\cdot\frac{4+10}{3+1} = 4.7 \qquad f_{2,1\to0} = \frac{7}{6}\cdot\frac{8\cdot(2+4)+(1+2)}{4+10} = 4.25$$

The critical path of the HLFF is exercised with a "0-to-1" transition at data input. The first stage of the HLFF is a skewed NAND gate.

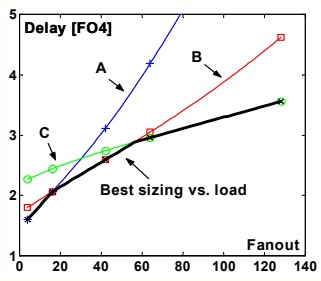
## HLFF Delay (normalized to FO4 inverter delay) vs. Fanout for Different HLFF Cell Sizes

| **Fanout** | 4 | 16 | 42 | 64 | 128 |
|---|---|---|---|---|---|
| **Load-Size (#stages)** | | | | | |
| Small-A (2) | 1.60 | 2.06 | 3.11 | 4.19 | 7.80 |
| Medium-B (2) | 1.80 | 2.06 | 2.59 | 3.05 | 4.62 |
| Large-C (2+1) | 2.27 | 2.44 | 2.74 | 2.96 | 3.56 |

There is only one optimal solution for each load size.



## Sizing versus load, HLFF example: linear scale

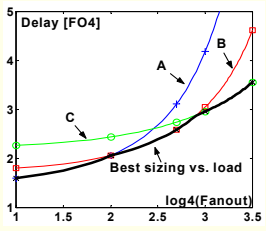



According to the logical effort theory, the optimal delay versus fan-out curve should have logarithmic shape, which indeed holds for the "best sizing vs. load curve".



## Sizing versus load, HLFF example: log4 scale

Similarly, the optimal delay is a linear function of the logarithm of the electrical effort (fanout), as shown. The logarithmic fanout scale makes it easy to see if the stage effort is properly determined.
Delay is linear function of stage effort and number of stages.
The logarithm of the electrical effort approximately illustrates the needed number of stages, and if the delay checks out to be multiple of number of stages and FO4 delay, then the optimal effort per stage is chosen.






## Case of Modified SAFF



## M-SAFF Delay vs. Fanout for Different M-SAFF Cell Sizes

| **Fanout** | 4 | 16 | 42 | 64 | 128 |
|---|---|---|---|---|---|
| **Optimal Load-Size (#stages)** | | | | | |
| Small-A (2) | 2.33 | 2.60 | 3.11 | 3.53 | 4.70 |
| Medium-B (2) | 2.35 | 2.59 | 3.01 | 3.34 | 4.24 |
| Large-C (2+1) | 3.06 | 3.15 | 3.31 | 3.44 | 3.83 |

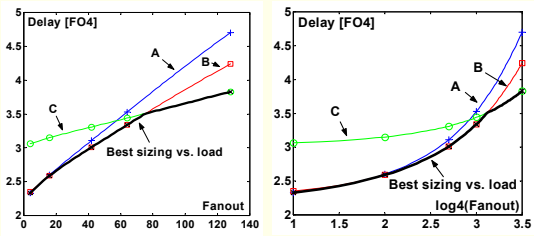
In this case we also observe the minimum data-to-output (*D-Q*) delay as the output load of the CSE increases. The performance of three different sizing solutions is illustrated versus the electrical fanout, normalized to the delay of the FO4 inverter.



## Modified SAFF

- The sizing is done in a way similar to that described in the HLFF example.
- We recognize that the logical effort of the input stage is very small, better than that of an inverter, because of the small input capacitance.
- This implies that the sizing changes will mostly be located in the output stage since the input stage can accommodate larger load variations without the need for resizing.
- While it was relatively easy to find different sizes that perform better at certain loads, in the case of HLFF, it was not so in the case of M-SAFF.
- The small logical effort of the whole structure enables it to cover a huge range of loads with a single size achieving relatively good performance.
- This is the case with structure of size B in Table. Size A is only slightly better than size B, and only for very light load of FO4, and then size B device takes the lead all the way up to the FO64 after which additional inverter is needed to prevent excessive delay.

## *Sizing versus load, M-SAFF example: (a) linear, (b) log4 scale*




-Size A is only slightly better than size B, and only for very light load of FO4

- Size B device takes the lead all the way up to the FO64 after which additional inverter is needed to prevent excessive delay.



## Energy Measurements

- While we were concerned about the performance aspects of the simulation setup, it is very important to prepare the simulation environment correctly such that the energy parameters of the CSE are measured accurately.
- We only need set the measurements to capture the energy for each of four possible binary transitions.
- With these values accurate average energy estimates can be made based on the statistics of the incoming data.
- More formal methods, using state transition diagrams (Zyban and Kogge 1999), can be used to exactly evaluate the effect of regular transitions and glitches on total switching energy of the CSE.
- It is essential to provide separate supply voltages for different stages of the CSE in order to measure different energies.

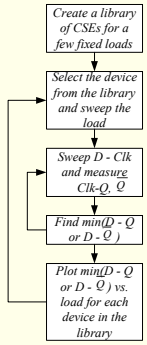


## Automating the Simulations

- The delay vs. load CSE evaluation described in the examples can be implemented automatically.
- *Perl* is suggested as one of the most convenient scripting languages today.
- For each CSE, we need to determine:
  - the logical effort of every stage based on its topology (e.g. 2 NAND-like stages, 1 inverter stage, would be 4/3, 4/3, 1), or better yet,
  - exact logical-effort values obtained from the simulation.
  - FO4 delay and other data for a given technology process.
  - The product of logical efforts of all stages should equal the total logical effort of the CSE. After total logical effort is found, optimal number of stages and updated stage effort can be calculated.
- With stage effort and logical efforts obtained from the topology of the CSE, taking the data input of fixed size, and assuming that the clock is "on" (i.e. treating the structure as cascade of logic gates), transistor sizes for every stage can be calculated, progressing from the data input to the final load in the simulation setup.
- When a library of CSEs is created, a pre-simulation should be run for each environment parameter setup. This includes various process corners, supply voltages, etc., to determine the FO4 inverter slope and set that value as the rise/fall time of signals that drive data and clock into the CSE.



## *CSE flow simulation*




1. For each device in the library, *D*-*Q*($\overline{Q}$) delay and *Clk*-*Q*($\overline{Q}$) delay are stored in each run, decreasing the delay between the edge of the input data and clock edge (setup time).
2. Script should check for the setup/hold time failure (i.e. when the CSE fails to pass the input value to the output).
   - This is typically detected by the large *Clk*-*Q* delay (i.e. measurement target occurred in the next cycle) or failure to measure delay if only one cycle is simulated.
3. The script automatically finds the minimum delay point at all the specified outputs.
4. The whole procedure is repeated for a range of loads and the best sizing curve.

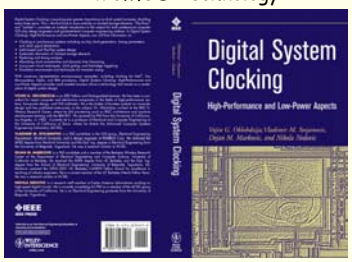
The Appendix A of this book contains an example script written in *Perl* that can serve as a basis of a more sophisticated tool for CSE characterization. In addition it also contains example spice decks for HLFF and M-SAFF used in this example. These files are a good start for a designer who wants to evaluate various CSE topologies.

# Digital System Clocking:

*High-Performance and Low-Power Aspects*

Vojin G. Oklobdzija, Vladimir M. Stojanovic, Dejan M. Markovic, Nikola M. Nedovic

Chapter 8: State-of-the-Art Clocked Storage Elements in CMOS Technology



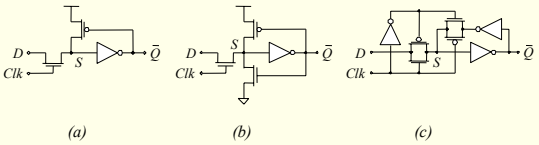

Wiley-Interscience and IEEE Press, January 2003

- Master-Slave Latches
- Flip-Flops
- CSE's with local clock gating
- Low clock swing
- Dual-edge triggering



## Transmission gate latches

*Simplest implementation* *Basic static latch* *Complete implementation*



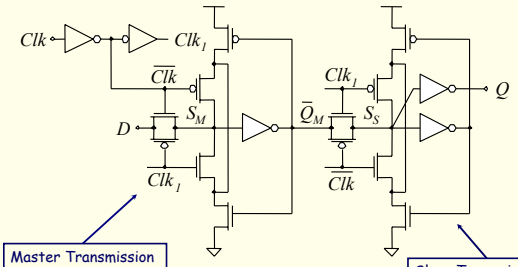

- only 4 transistors
- Dynamic when S=1
- Susceptible to noise

- pull-up/pull-down keeper
- Conflict at node S whenever new data is written

- Feedback turned off when writing to the latch
- No conflict
- Larger clock load



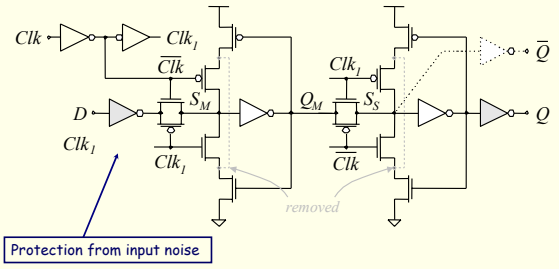
## Transmission Gate Master-Slave Latch (MSL)




*MSL with unprotected input*
*(Gerosa et al. 1994), Copyright © 1994 IEEE*



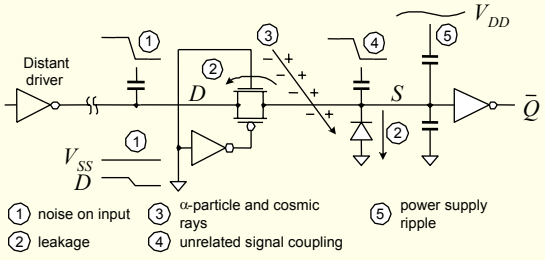
## Transmission Gate MS Latch (continued)




*MSL with input gate isolation*
*(Markovic et al. 2001), Copyright © 2001 IEEE*

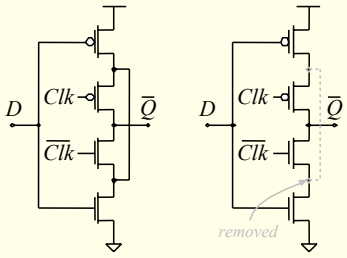


## Noise Robustness of MS Latch




*Sources of noise affecting the latch state node*
*(Partovi in Chandrakasan et al. 2001), Copyright © 2001 IEEE*

# Clocked CMOS ($C^2$MOS) Latch

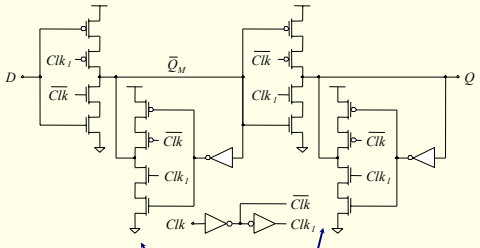



*Transmission gate latch with gate isolation (dynamic)*

*$C^2$MOS latch (dynamic)*



# Clocked CMOS ($C^2$MOS) MS Latch

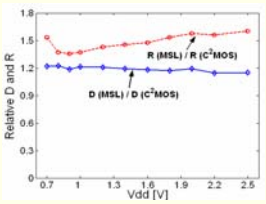



State-keeping feedbacks outside the D-to-Q path

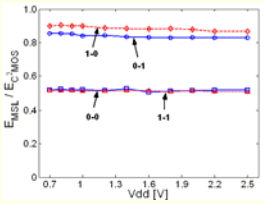
*(Suzuki et al. 1973), Copyright © 1973 IEEE*



# MS Latches: Comparison




*Delay (D) and Race immunity (R)*




*Energy per cycle*

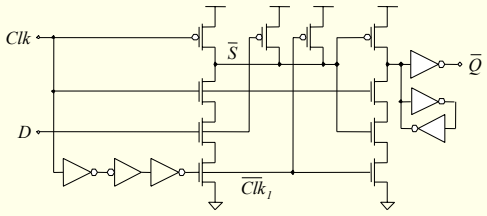
$C^2$MOS: larger clock transistors:
-Smaller delay and race immunity (80% of MSL)
-Higher energy consumption (1.4x more than MSL)



- Master-Slave Latches
- Flip-Flops
- CSE's with local clock gating
- Low clock swing
- Dual-edge triggering



# Hybrid Latch Flip-Flop (HLFF)

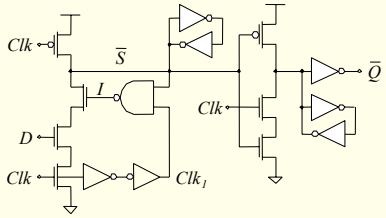



- Transparent to D only when Clk and $\overline{Clk}_1$ are both high
- Limited clock uncertainty absorption
- Small D→Q delay
- Small clock load

*(Partovi et al. 1996), Copyright © 1996 IEEE*



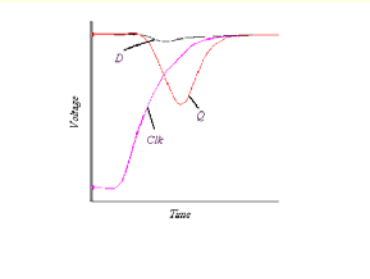
# Semidynamic Flip-Flop (SDFF)




- Dynamic-style first stage
  - Fast, small clock load, logic embedding
  - Consumes energy for evaluation whenever D=1
- Dynamic-to-static latch in second stage
  - "Static 1" hazard

*(Klass 1998), Copyright © 1998 IEEE*

# "Static 1" hazard in SDFF

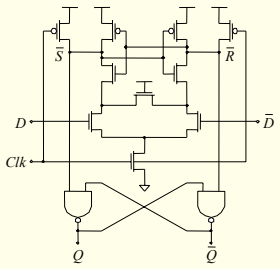



- If D=Q=1 in previous cycle, race between Clk and $\overline{S}$ causes Q to falsely switch to 0 → generated glitch
- Also seen in HLFF



# Sense-Amplifier Flip-Flop (SAFF)

- When Clk=0, $\overline{S}$ and $\overline{R}$ are high, Q and $\overline{Q}$ unchanged
- At rising edge of Clk
  - sense amplifier in 1st stage generates a "low" pulse on either $\overline{S}$ or $\overline{R}$, based on which of D and $\overline{D}$ is higher
  - Other node $\overline{R}$ or $\overline{S}$ is driven high, preventing further changes
  - Latch captures low level of $\overline{S}$ or $\overline{R}$ and updates output

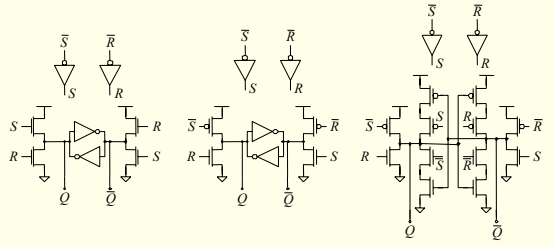



Both NAND gates must sequentially switch to change Q and $\overline{Q}$

*Original design (Montanaro et al. 1996), Copyright © 1996 IEEE*



# SAFF: Evolution of 2nd Stage Latch



*all-n-MOS push-pull (Gieseke et al. 1991);*

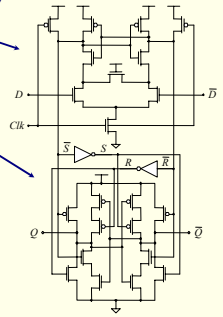
*complementary push-pull (Oklobdzija and Stojanovic 2001)*

*complementary push-pull with gated keeper (Nikolic et al. 1999).*



# Modified Sense Amplifier Flip-Flop (MSAFF)

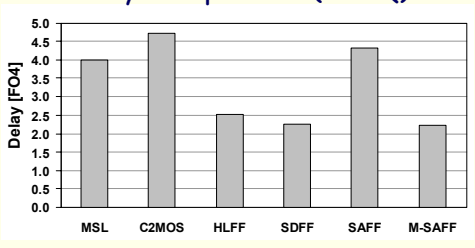
- Sense amplifier in 1st stage generates a "low" pulse on either $\overline{S}$ or $\overline{R}$, based on which of D and $\overline{D}$ is higher
- Symmetric latch in 2nd stage
  - outputs are simultaneously pulled to Vdd and Gnd → fast
  - Large drive capability → can be small
- Keeper in latch active only when there is no change
  - No conflict




*(Nikolic et al. 1999), Copyright © 1999 IEEE*



# Flip-Flops and MS Latches: Delay Comparison (D→Q)

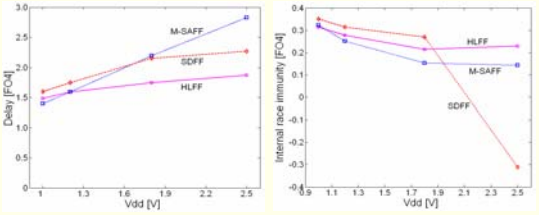



- MS Latches are slow – positive setup time, two latches in critical path
- SAFF is slow: it waits for one output to switch the other
- Fastest structures are simple flip-flops with negative setup time

*CSE delay comparison (0.18 μm, high load)*



# Flip-Flops: Timing Comparisons with Voltage Scaling

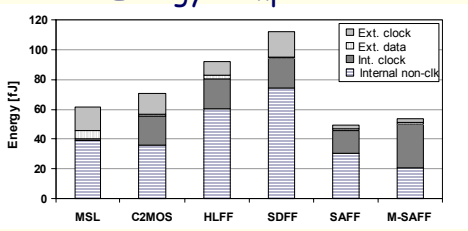



*Delay comparison:*
*- Relative delay reduces with supply voltage due to reduction of body effect*

*Internal race immunity comparison:*
*- Small race immunity, usually not a concern in critical paths*

*(0.25 μm, light load)*

# Flip-Flops and MS Latches: Energy Comparison

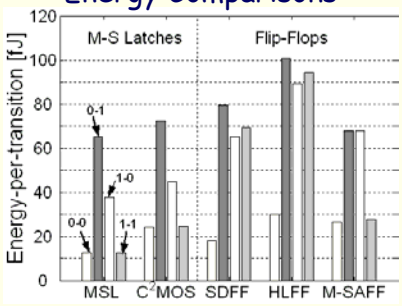



- In MS Latches, internal nodes change only when input D changes
- SAFF, M-SAFF: very small clock load, small 2nd stage latch
- Most energy consumed in HLFF, SDFF with pulse generator and high internal switching activity

*CSE energy breakdown (0.18 μm, 50% activity, high load)*



# Flip-Flops and MS Latches: Energy Comparisons




*(0.25 μm, light load)*
*(Markovic et al. 2001), Copyright © 2001 IEEE*

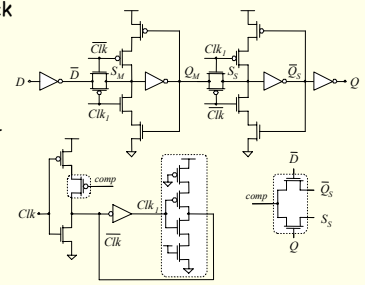


- Master-Slave Latches
- Flip-Flops
- CSE's with local clock gating
- Low clock swing
- Dual-edge triggering



# Gated Transmission Gate MS Latch

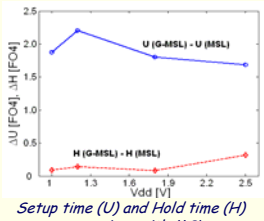
- Concept: inhibit clock switching when new D = Q
  - *comp*=D XNOR Q
  - If *comp*=0 (D≠Q), circuit works as MSL
  - If *comp*=1 (D=Q), Clk=0, Clk1=1 ⇒ latches closed, no output change, no internal power



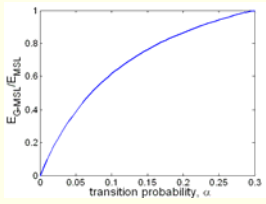

*Gated MSL*
*(Markovic et al. 2001), Copyright © 2001 IEEE*



# Gated TG MS Latch: Timing and Energy




*Setup time (U) and Hold time (H) comparison with MSL*

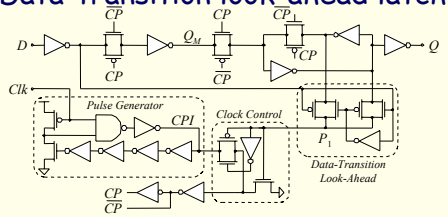



*Energy comparison with MSL*

- Increased Setup time in gated MSL due to inclusion of the comparator into the critical path ⇒ slower than conventional MSL
- Smaller energy per transition if switching activity of D is <0.3
  - For higher switching activity, comparator and clock generator dominate the energy consumption



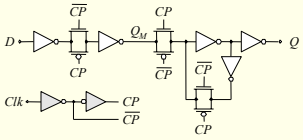
# Data-transition look-ahead latch



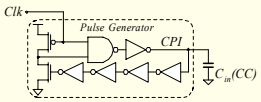

- Pulsed latch in which the generation of clock pulses are gated with XOR DTLA circuit
  - If D≠Q⇒ $P_1$=0, circuit operates as a conventional pulsed latch
  - If D=Q ⇒ $P_1$=1 ⇒ CP=0, no output change or energy consumption in the latch
- XOR circuit and Clock Control in the critical path → large setup time and D-Q delay

*(Nogawa and Ohtomo 1998), Copyright © 1998 IEEE*

# DTLA-L: Analysis of Energy Consumption




DTLA-L without clock gating




DTLA-L Pulse Generator

$$E_{CMSL} = \frac{\alpha}{2}\cdot(E_{0-1}+E_{1-0}) + \frac{1-\alpha}{2}\cdot E_{Clk} + E_{Cin}$$

$$E_{DL-DFF} = \frac{\alpha}{2}\cdot(E_{0-1}+E_{1-0}) + \frac{1-\alpha}{2}\cdot E_{D-idle}$$

$$E_{0-1} = E_{D-idle} + E_{CLK} + E_{DL+CC} + E_{int} + E_{ext} \qquad E_{1-0} = E_{D-idle} + E_{CLK} + E_G + E_{int}$$

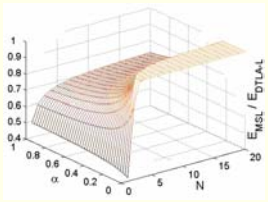
$$E_{D-idle} = E_{0-0} = E_{1-1} = \frac{E_{PG}}{N} + E_{Cin}$$

$\alpha$ – input switching activity

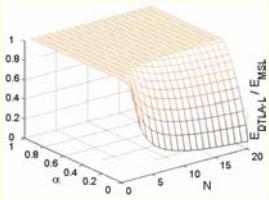
Pulse generator shared among N DTLA-L's



# Energy comparison of DTLA-L and CMSL




E(DTLA-L) < E(CMSL)



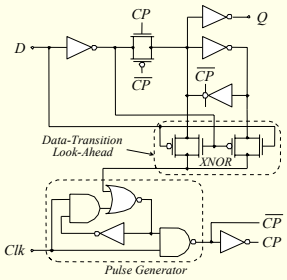

E(DTLA-L) > E(CMSL)

DTLA-L is more energy-efficient than CMSL when N>2 and $\alpha$< 0.25



# Clock-on-demand PL

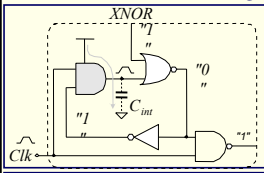
- Pulsed latch in which the generation of clock pulses are gated with XNOR DTLA circuit
  - If D≠Q⇒ XNOR=0, CP→1 when Clk↑, and CP→0 after Q has changed to D
  - If D=Q ⇒ XNOR=1 ⇒ CP=0, no output change or energy consumption in the latch
- Pulse Generator includes clock control
  - can not be shared among multiple PL's



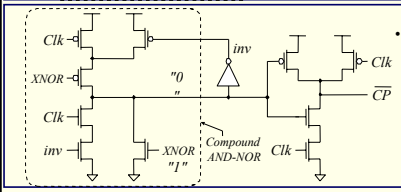

(Hamada et al. 1999), Copyright © 1999 IEEE



# Energy-Efficient Pulse Generator in COD-PL



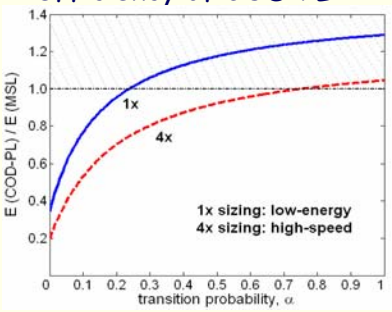

- Straightforward implementation with CMOS gates
  - $C_{int}$ switches in each cycle
  - Energy-inefficient




- Compound AND-NOR gate
  - Energy-efficient



# Impact of circuit sizing on the energy efficiency of COD-PL



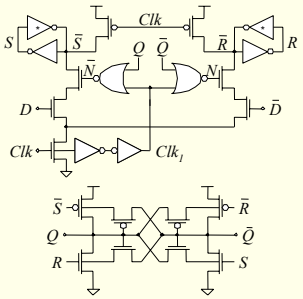

COD-PL more effective in high-speed sizing due to large clock transistors

(Markovic et al. 2001), Copyright © 2001 IEEE



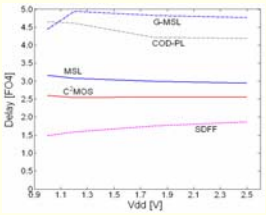
# Conditional capture flip-flop

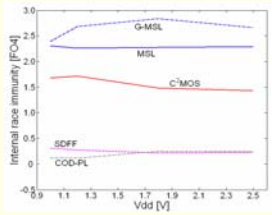
- First stage: pulse generator with internal clock gating
  - When Clk=1, $\overline{S}$=$\overline{R}$=1
  - When Clk=1, $Clk_1$=0, $\overline{S}$ can switch low if D=1, Q=0, $\overline{R}$ can switch low if D=0, Q=1
  - Otherwise, $\overline{S}$=$\overline{R}$=1 → no energy consumption
- Second stage: pass-gate implementation of M-SAFF latch (Oklobdzija, Stojanovic)
- *No setup time degradation due to clock gating*




(Kong et al. 2000), Copyright © 2000 IEEE

## Comparison of latches and flip-flops with local clock gating: Timing



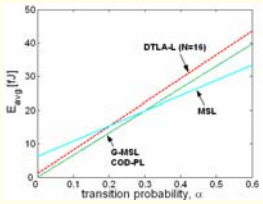

*Delay comparison:*
- *Delay relatively constant with supply voltage*
- *Latches with clock gating have very large delay due to large setup time*

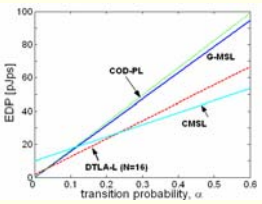
*Internal race immunity comparison:*
- *Generally R(FF)< R(MSL)< R(gated MSL)*
- *COD-PL has low race immunity due to wide clock pulse*

*(Markovic et al. 2001), Copyright © 2001 IEEE*



## Comparison of latches and flip-flops with local clock gating: Energy, EDP




*Energy comparison:*
- *Latches with gated clock consume less energy than MSL if α < 0.2 – 0.3*

*Energy-Delay Product comparison:*
- *α < 0.03 ⇒ G-MSL best*
- *0.03 < α < 0.23 ⇒ DTLA-L best*
- *0.23 < α ⇒ Conventional MSL best*

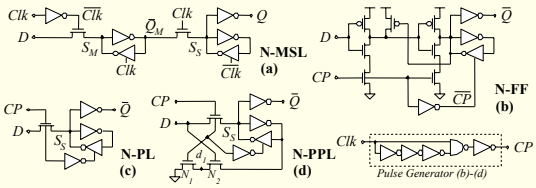
*(Markovic et al. 2001), Copyright © 2001 IEEE*



- Master-Slave Latches
- Flip-Flops
- CSE's with local clock gating
- Low clock swing
- Dual-edge triggering



## N-only clocked latches



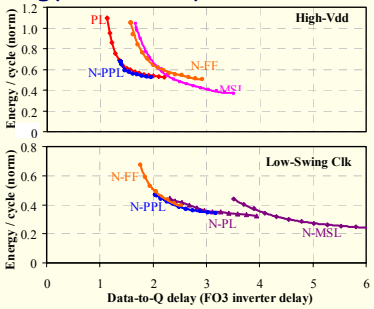

- Concept: Bring clock only to n-MOS transistors to allow reduced clock swing without conflict with partially turned-off p-MOS transistors
- Reduced clock swing reduces clocking energy with some penalty in performance
  - Clock is always in critical path as its edge signalizes when to change the output

*(a) conventional TG MSL, (b) pulsed-latch, (c) conventional PL, (d) push-pull PL*



## Low clock swing CSE's comparison: energy and delay

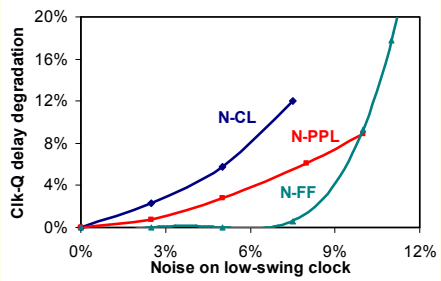
- Full-swing:
  - PL preferred for high-speed
  - MSL preferred for low energy
- Low-swing clock:
  - N-FF preferred for high-speed
  - N-PPL is preferred for low energy




*130nm technology, 50fF load, max. input cap=12.5fF, data activity=0.1: (a) high-$V_{dd}$ and (b) low-swing Clk*



## Effect of clock noise on low-swing clock latch delay



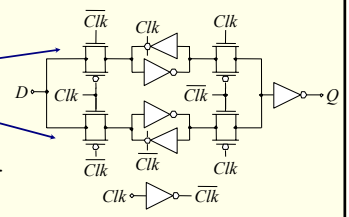

- All latches fail for clock noise > 12% of clock voltage
- N-FF gives best clock noise rejection

- Master-Slave Latches
- Flip-Flops
- CSE's with local clock gating
- Low clock swing
- Dual-edge triggering



# DET Latch-mux circuit (DET-LM)

- Pass-gate latches:
  - One transparent when Clk=0
  - One transparent when Clk=1
- Pass-gate multiplexer that selects the output of the opaque latch

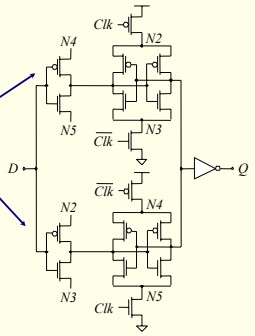



*(Llopis and Sachdev 1996), Copyright © 1996 IEEE*



# $C^2$MOS Latch-mux ($C^2$MOS-LM)

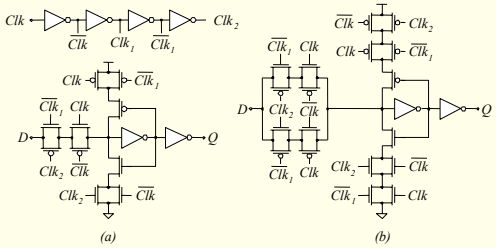
- $C^2$MOS latches:
  - One transparent when Clk=1
  - One transparent when Clk=0
- Multiplexer: two $C^2$MOS inverters that propagate the output of the opaque latch
- Large clock transistors shared between the latches and the multiplexer




*(Gago et al. 1993), Copyright © 1993 IEEE*



# Pulsed-latch (DET-PL)



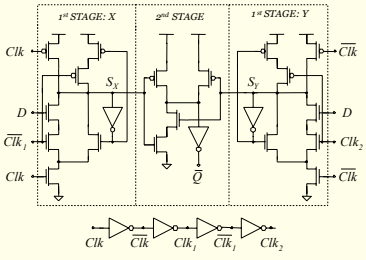

- Pulse generator transparent to D only when $Clk=\overline{Clk_1}=1$, or when $\overline{Clk}=Clk_2=1 \Rightarrow$ shortly after both edges of the clock
- DET PL consumes lot of energy for four clocked pass gates
- To improve speed, modified from original design (Strollo et al, 1999) which implemented n-MOS-only pass gate and p-MOS-only keeper

*(a) single - edge, (b) dual - edge triggered*



# DET Symmetric pulse generator flip-flop (SPGFF)

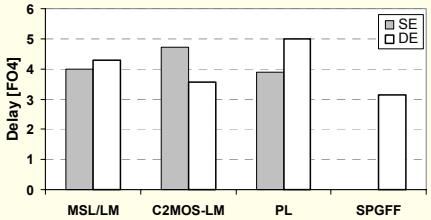
- Two pulse generators: X active at rising edge of the clock, Y active at falling edge of the clock
- $S_X$ and $S_Y$ alternately precharge and evaluate
  - At any moment, one of $S_X$ and $S_Y$ keeps the value of data sampled at the most recent clock edge
  - The other $S_X$ or $S_Y$ is precharged high




- *Pulses at $S_X$ and $S_Y$ have same width as clock*
- *Second stage is a simple NAND gate $\Rightarrow$ no need for a latch*



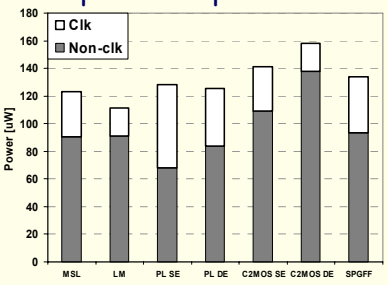
# SET vs. DET: Delay comparison




- Latch-MUX's have two equally critical paths, somewhat shorter than that of MSL
- PL is more complex, adding more capacitance to the critical path compared to SET PL
- SPGFF has short domino-like critical path $\Rightarrow$ fastest

# SET vs. DET: Power consumption comparison

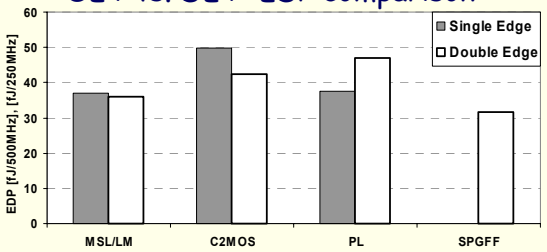
- LM's benefit from clever implementation of latch-mux structure with clock transistors sharing
- PL adds extra high-activity capacitance compared to SET PL
- SPGFF power consumption is in the middle, mainly due to alternate switching of nodes $S_X$ and $S_Y$




*(0.18 μm, 500MHz for SET, 250MHZ for DET, high load)*



# SET vs. DET: EDP comparison




- Latch-MUX's have similar or better EDP than their SET counterparts
- PL exhibits worse delay and energy compared to SET PL, due to more complex design
- SPGFF is fastest with moderate energy consumption: lowest EDP
- EDP (SPGFF) < EDP (LM) < EDP (PL)

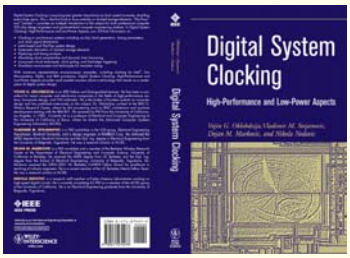
*(0.18 μm, 500MHz for SET, 250MHZ for DET high load)*

# Digital System Clocking:

*High-Performance and Low-Power Aspects*

Vojin G. Oklobdzija, Vladimir M. Stojanovic, Dejan M. Markovic, Nikola M. Nedovic

Chapter 9: Microprocessor Examples




Wiley-Interscience and IEEE Press, January 2003

## Microprocessor Examples

- Clocking for Intel® Microprocessors
  - IA-32 Pentium® Pro
  - First IA-64 Microprocessor
  - Pentium 4
- Sun Microsystems UltraSPARC-III® Clocking
  - Clocking and CSEs
- Alpha® Clocking: A Historical Overview
  - Clocking and CSEs
- IBM® Microprocessors
  - Level-Sensitive Scan Design
  - Examples of CSEs



## Microprocessor Examples

- Clocking for Intel® Microprocessors
  - IA-32 Pentium® Pro
  - First IA-64 Microprocessor
  - Pentium 4
- Sun Microsystems UltraSPARC-III® Clocking
  - Clocking and CSEs
- Alpha® Clocking: A Historical Overview
  - Clocking and CSEs
- IBM® Microprocessors
  - Level-Sensitive Scan Design
  - Examples of CSEs



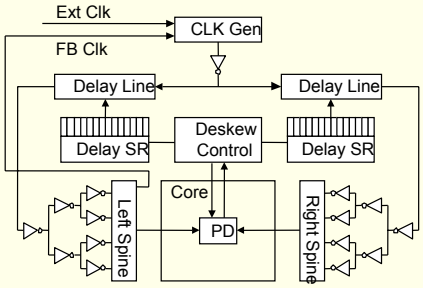
## Intel® Microprocessor Features

| | Pentium® II | Pentium® III | Pentium® 4 |
|---|---|---|---|
| **MPR Issue** | June 1997 | April 2000 | Dec 2001 |
| **Clock Speed** | 266 MHz | 1GHz | 2GHz |
| **Pipeline Stages** | 12/14 | 12/14 | 22/24 |
| **Transistors** | 7.5M | 24M | 42M |
| **Cache (I/D/L2)** | 16k/16K/- | 16K/16K/256K | 12K/8K/256K |
| **Die Size** | 203mm$^2$ | 106mm$^2$ | 217mm$^2$ |
| **IC Process** | 0.28µm, 4M | 0.18µm, 6M | 0.18µm, 6M |
| **Max Power** | 27W | 23W | 67W |

*Source: Microprocessor Report Journal*



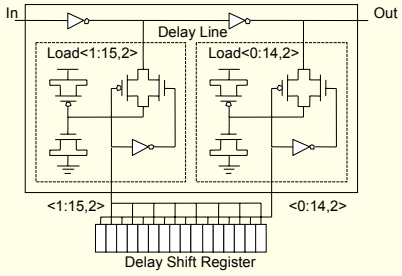
## IA-32 Pentium® Pro




*Clock distribution network with deskewing circuit (Geannopoulos and Dai 1998), Copyright © 1998 IEEE*

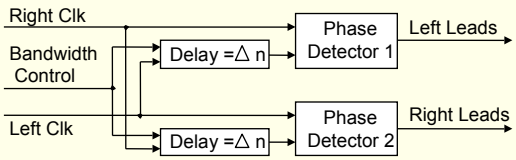


## IA-32 Pentium® Pro




*Delay shift register (Geannopoulos and Dai 1998), Copyright © 1998 IEEE*

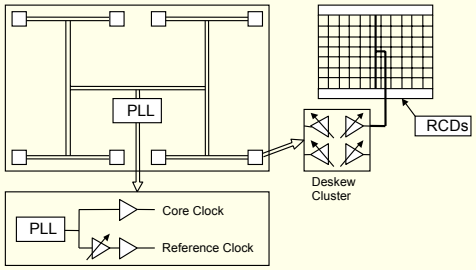
## IA-32 Pentium® Pro




*Phase detector*
*(Geannopoulos and Dai 1998), Copyright © 1998 IEEE*



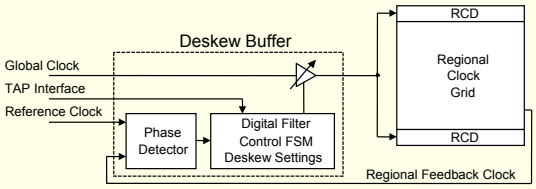
## First IA-64 Microprocessor




*Clock distribution topology*
*(Rusu and Tam 2000), Copyright © 2000 IEEE*



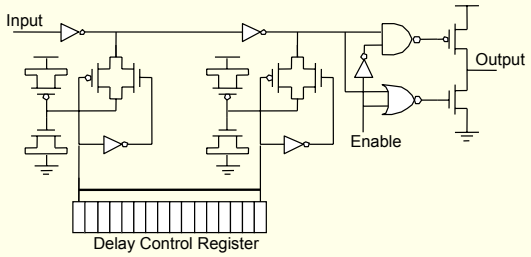
## First IA-64 Microprocessor




*Deskew buffer architecture*
*(Rusu and Tam 2000), Copyright © 2000 IEEE*



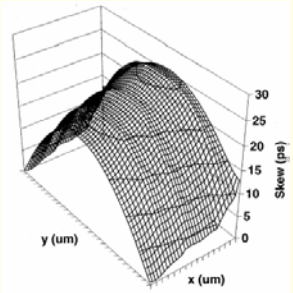
## First IA-64 Microprocessor




*Digitally controlled delay line*
*(Rusu and Tam 2000), Copyright © 2000 IEEE*



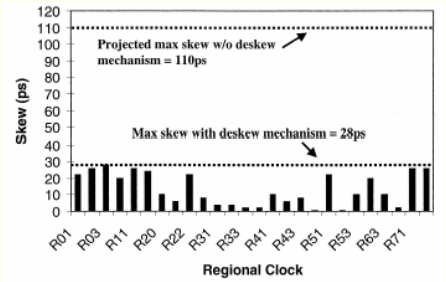
## First IA-64 Microprocessor




*Simulated regional clock-grid skew*
*(Rusu and Tam 2000), Copyright © 2000 IEEE*

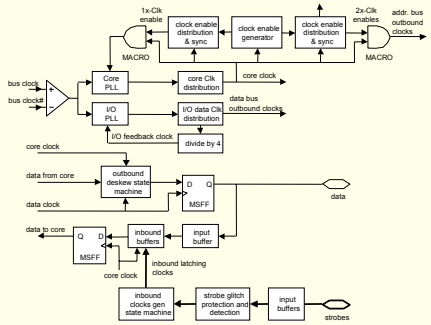


## First IA-64 Microprocessor




*Measured regional clock skew*
*(Rusu and Tam 2000), Copyright © 2000 IEEE*

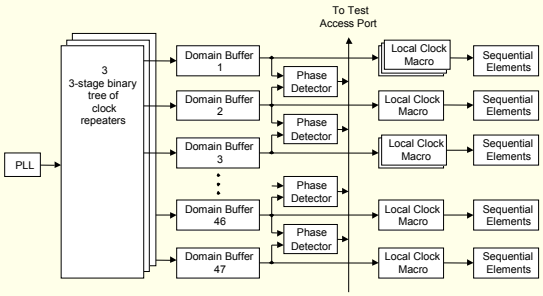
# Pentium® 4




*Core and I/O clock generation*
*(Kurd et al. 2001), Copyright © 2001 IEEE*



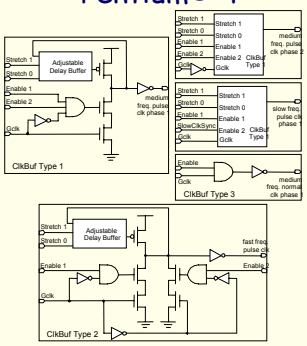
# Pentium® 4




*Logical diagram of core clock distribution*
*(Kurd et al. 2001), Copyright © 2001 IEEE*



# Pentium® 4




*Example of local clock buffers generating various frequency, phase and types of clocks*
*(Kurd et al. 2001), Copyright © 2001 IEEE*



# Microprocessor Examples

- Clocking for Intel® Microprocessors
  - IA-32 Pentium® Pro
  - First IA-64 Microprocessor
  - Pentium 4
- Sun Microsystems UltraSPARC-III® Clocking
  - Clocking and CSEs
- Alpha® Clocking: A Historical Overview
  - Clocking and CSEs
- IBM® Microprocessors
  - Level-Sensitive Scan Design
  - Examples of CSEs

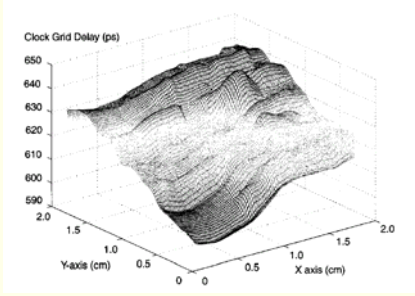


# UltraSPARC® Family Characteristics

| | UltraSPARC-I® | UltraSPARC-II® | UltraSPARC-III® |
|---|---|---|---|
| **Year** | 1995 | 1997 | 2000 |
| **Architecture** | SPARC V9, 4-issue | SPARC V9, 4-issue | SPARC V9, 4-issue |
| **Die size** | 17.7x17.8mm$^2$ | 12.5x12.5mm$^2$ | 15x15.5mm$^2$ |
| **# of transistors** | 5.2M | 5.4M | 23M |
| **Clock Frequency** | 167MHz | 330MHz | 1GHz |
| **Supply voltage** | 3.3V | 2.5V | 1.6V |
| **Process** | 0.5µm CMOS | 0.35µm CMOS | 0.15µm CMOS |
| **Metal layers** | 4 (Al) | 5 (Al) | 7 (Al) |
| **Power consumption** | <30W | <30W | <80W |

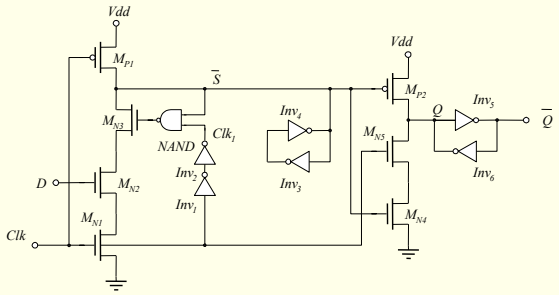


# UltraSPARC-III®: Clocking




*Clock distribution delay in UltraSPARC-III*
*(Heald et al. 2000), Copyright © 2000 IEEE*

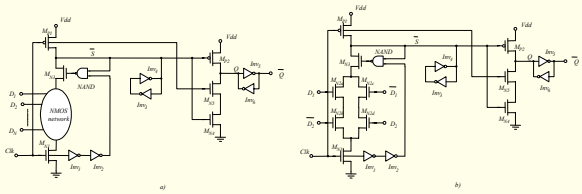
# UltraSPARC-III: Clock Storage Elements




*Semidynamic flip-flop*
*(Klass 1998), Copyright © 1998 IEEE*



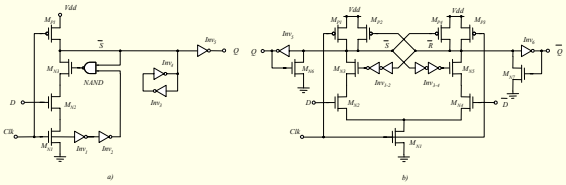
# UltraSPARC-III: Clock Storage Elements



*(a) Logic embedding in a semidynamic flip-flop;*
*(b) Two-input XOR function.*
*(Klass, 1998), Copyright © 1998 IEEE*



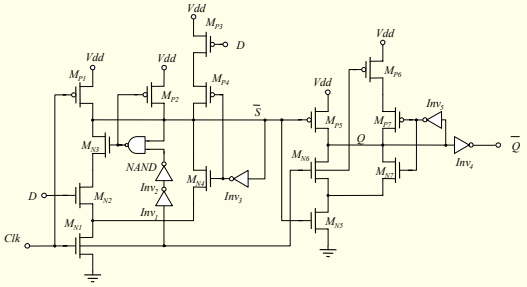
# UltraSPARC-III: Clock Storage Elements



*Dynamic versions of semidynamic flip-flop:*
*(a) single-ended; (b) Differential.*
*(Klass 1998), Copyright © 1998 IEEE*



# UltraSPARC-III: Clock Storage Elements




*UltraSPARC-III flip-flop*
*(Heald et al. 2000), Copyright © 2000 IEEE*



# Microprocessor Examples

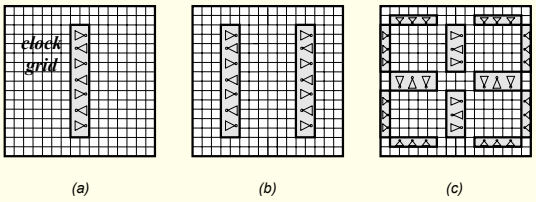
- Clocking for Intel® Microprocessors
  - IA-32 Pentium® Pro
  - First IA-64 Microprocessor
  - Pentium 4
- Sun Microsystems UltraSPARC-III® Clocking
  - Clocking and CSEs
- Alpha® Clocking: A Historical Overview
  - Clocking and CSEs
- IBM® Microprocessors
  - Level-Sensitive Scan Design
  - Examples of CSEs



# Alpha® Microprocessor Features

| | 21064 | 21164 | 21264 | 21364 |
|---|---|---|---|---|
| **# transistors [M]** | 1.68 | 9.3 | 15.2 | 152 |
| **Die Size [mm²]** | 16.8x13.9 | 18.1x16.5 | 16.7x18.8 | 21.1x18.8 |
| **Process** | 0.75µm | 0.5µm | 0.35µm | 0.18µm |
| **Supply [V]** | 3.3 | 3.3 | 2.2 | 1.5 |
| **Power [W]** | 30 | 50 | 72 | 125 |
| **Clk Freq. [MHz]** | 200 | 300 | 600 | 1200 |
| **Gates/Cycle** | 16 | 14 | 12 | 12 |

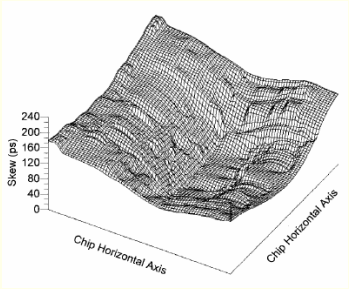
# Alpha® Microprocessors: Clocking




*Alpha microprocessor final clock driver location:*
*(a) 21064, (b) 21164, (c) 21264*



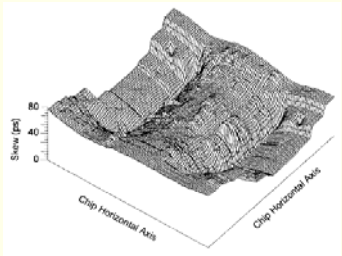
# Alpha® Microprocessors: Clocking




*21064 clock skew*
*(Gronowski et al. 1998), Copyright © 1998 IEEE*



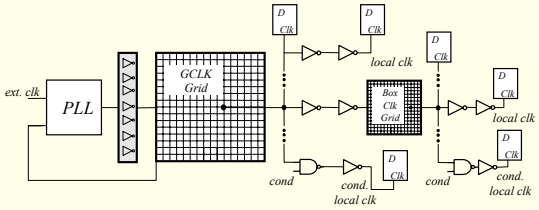
# Alpha® Microprocessors: Clocking




*21164 clock skew*
*(Gronowski et al. 1998), Copyright © 1998 IEEE*



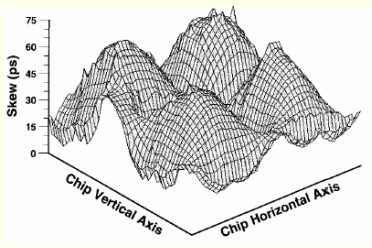
# Alpha® Microprocessors: Clocking




*21264 clock hierarchy*
*(Gronowski et al. 1998), Copyright © 1998 IEEE*



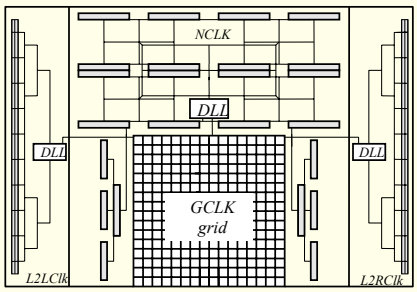
# Alpha® Microprocessors: Clocking




*21264 clock skew*
*(Gronowski et al. 1998), Copyright © 1998 IEEE*

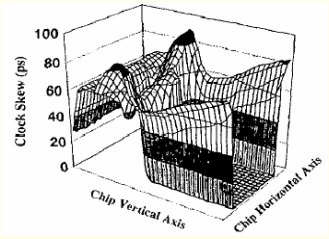


# Alpha® Microprocessors: Clocking




*21364 major clock domains*
*(Xanthopoulos et al. 2001), Copyright © 2001*

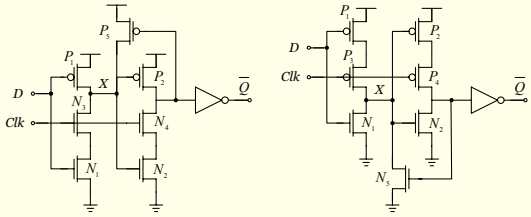
# Alpha® Microprocessors: Clocking




*21364, NCLK clock skew*
*(Xanthopoulos et al. 2001), Copyright © 2001 IEEE*



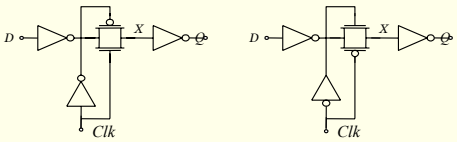
# Alpha® µP: Clock Storage Elements




*21064 modified TSPC latches*
*(Gronowski et al. 1998), Copyright © 1998*



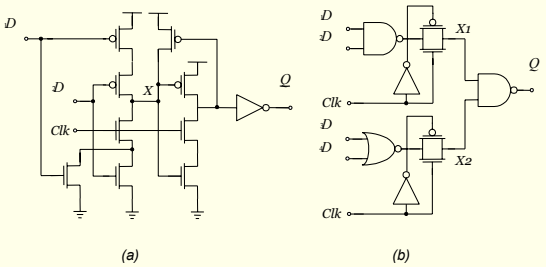
# Alpha® µP: Clock Storage Elements




*21164: (a) phase-A latch, (b) phase-B latch*
*(Gronowski et al. 1998), Copyright © 1998 IEEE*



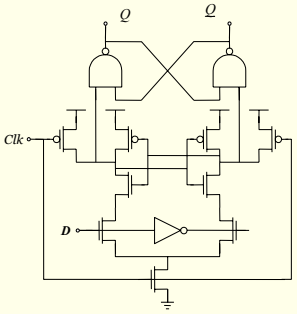
# Alpha® µP: Clock Storage Elements




*Embedding of logic into a latch:*
*(a) 21064 TSPC latch, one level of logic;*
*(b) 21164 latch, two levels of logic.*
*(Gronowski et al. 1998), Copyright © 1998 IEEE*



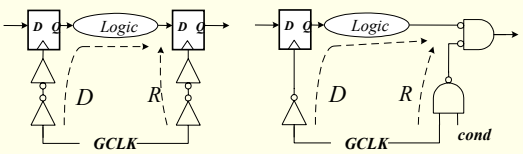
# Alpha® µP: Clock Storage Elements




*21264 flip-flop*
*(Gronowski et al. 1998), Copyright © 1998 IEEE*



# Alpha® Microprocessors: Timing




| Critical Path Definition and Criteria | Race Definition and Criteria |
|---|---|
| - Identify common clock, D and R | - Identify common clock, D and R |
| - Maximize D | - Minimize D |
| - Minimize R | - Maximize R |
| DLSRT | DRH |

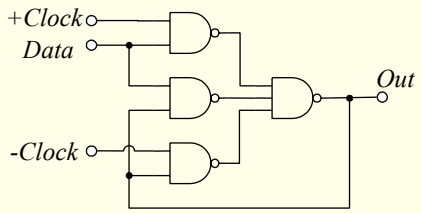
*Critical-path and race analysis for clock buffering and conditioning*
*(Gronowski et al. 1998), Copyright © 1998 IEEE*

## Microprocessor Examples

- Clocking for Intel® Microprocessors
  - IA-32 Pentium® Pro
  - First IA-64 Microprocessor
  - Pentium 4
- Sun Microsystems UltraSPARC-III® Clocking
  - Clocking and CSEs
- Alpha® Clocking: A Historical Overview
  - Clocking and CSEs
- IBM® Microprocessors
  - Level-Sensitive Scan Design
  - Examples of CSEs



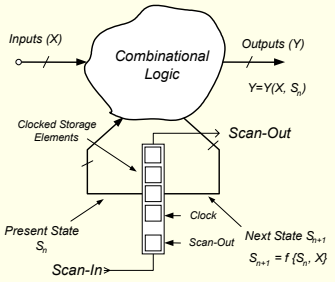
## Hazard-Free Level-Sensitive Polarity-Hold Latch




*Eichelberger 1983*

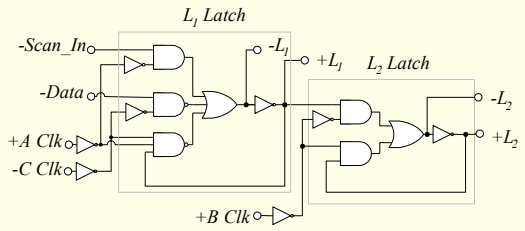


## General LSSD Configuration



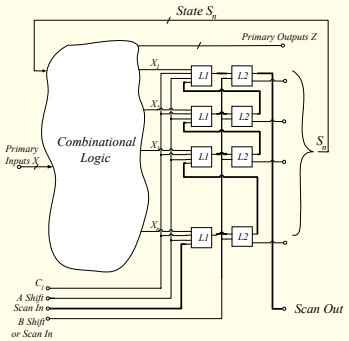



## LSSD Shift Register Latch



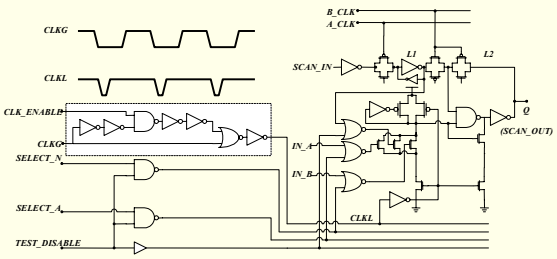



## LSSD Double Latch Design



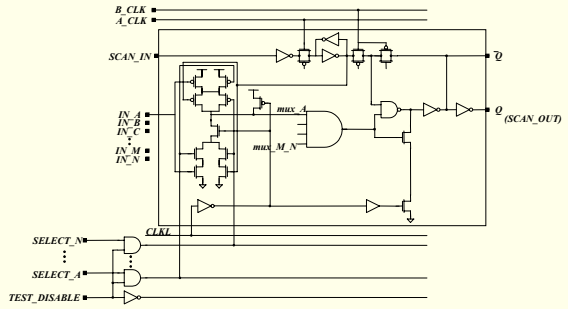



## IBM® S/390 Parallel Server Processor




*LSSD SRL with multiplexer used in the IBM S/390 G4 processor (Sigal et al. 1997), reproduced by permission*

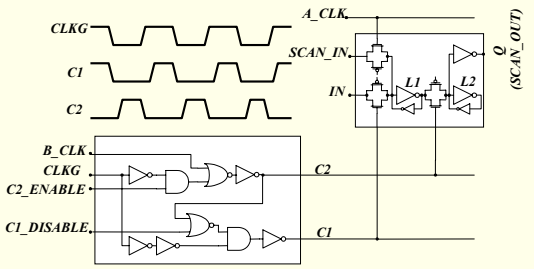
# IBM® S/390 Parallel Server Processor




*Static multiplexer version of the SRL used in the IBM S/390 G4 (Sigal et al. 1997), reproduced by permission*



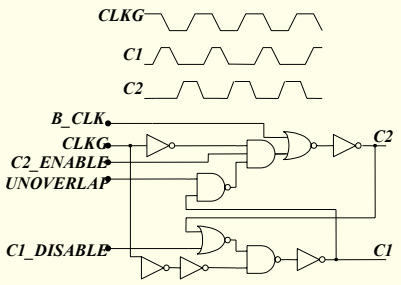
# IBM® S/390 Parallel Server Processor




*A clocked storage element is used in the non-timing-critical timing macros of the IBM S/390 G4 processor (Sigal et al. 1997), reproduced by permission*



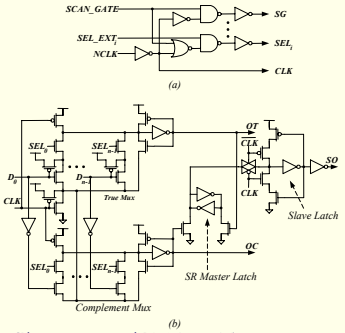
# IBM® S/390 Parallel Server Processor




*The clock-generation element used to detect problems created with fast paths: IBM S/390 G4 processor (Sigal et al. 1997), reproduced by permission*



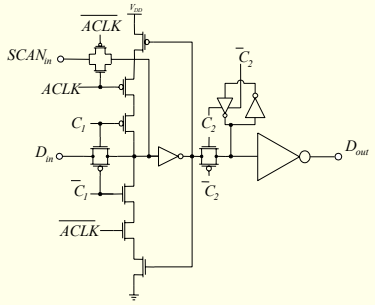
# IBM® PowerPC Processor




*The experimental IBM PowerPC processor (Silberman et al. 1998), reproduced by permission*



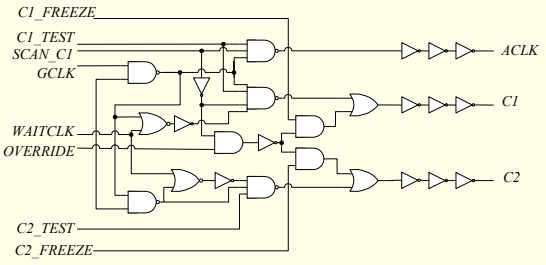
# IBM® PowerPC 603: Master-Slave Latch




*The PowerPC 603 MSL (Gerosa et al. 1994), Copyright © 1994 IEEE*



# IBM® PowerPC 603: Local Clk Generator




*The PowerPC 603 local clock regenerator (Gerosa et al. 1994), Copyright © 1994 IEEE*

## Summary

- Intel® Microprocessors
  - Active clock deskewing in Pentium® processors
- Sun Microsystems® Processors
  - Semidynamic flip-flop (one of the fastest single-ended flip-flops today, "soft-edge")
- Alpha® Processors
  - Performance leader in the '90s
  - Incorporating logic into CSEs
- IBM® Processors
  - Design for testability techniques
  - Low-power champion PowerPC 603