SP-25.7: Skew-Tolerant Domino Circuits

David Harris[1] and Mark Horowitz

Stanford University Stanford, CA

[1]Also with Intel Corporation Santa Clara, CA

Overview

General Domino Design

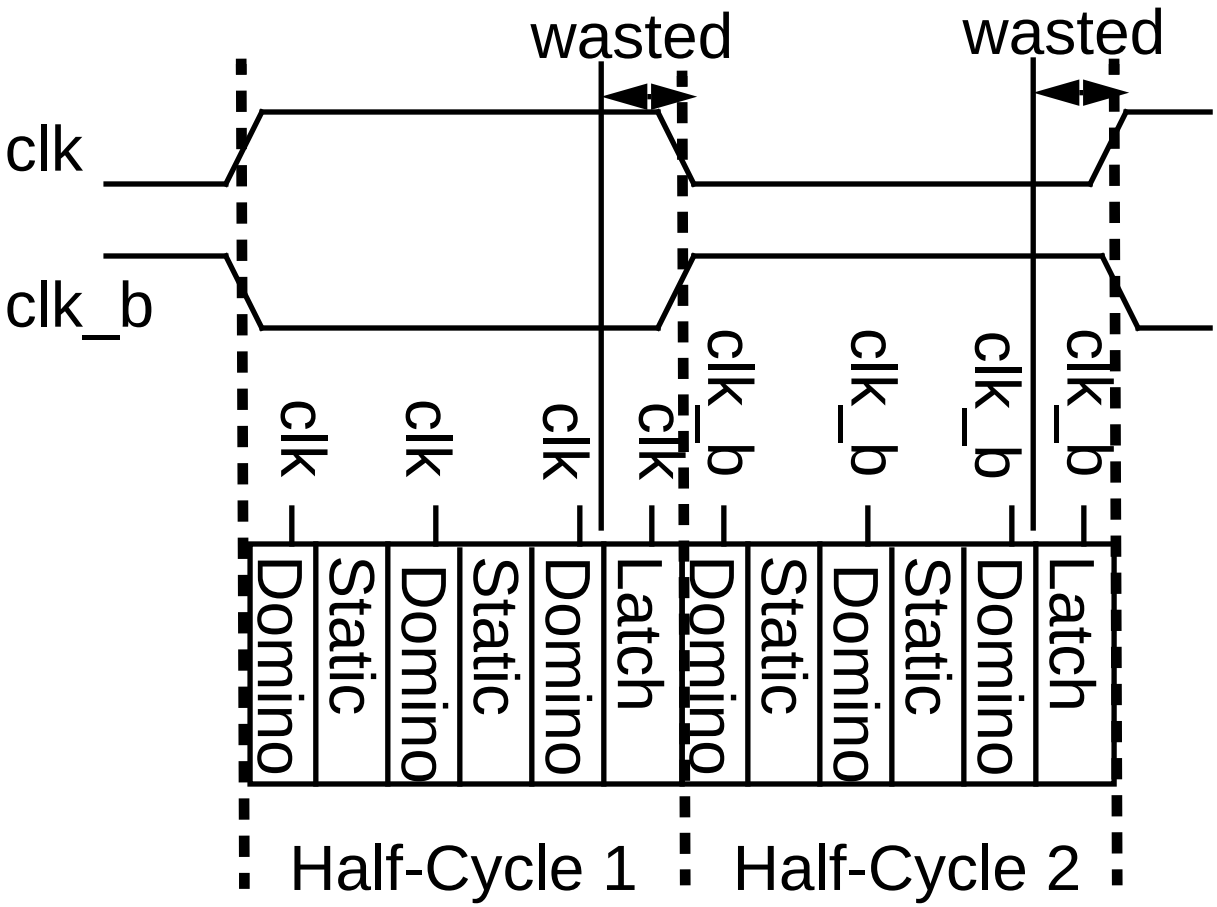
- **Domino performance**
- **Domino overhead**
- **Eliminating overhead**

Skew-Tolerant Domino

- **Implementation issues**
- **Performance results**

Conclusions

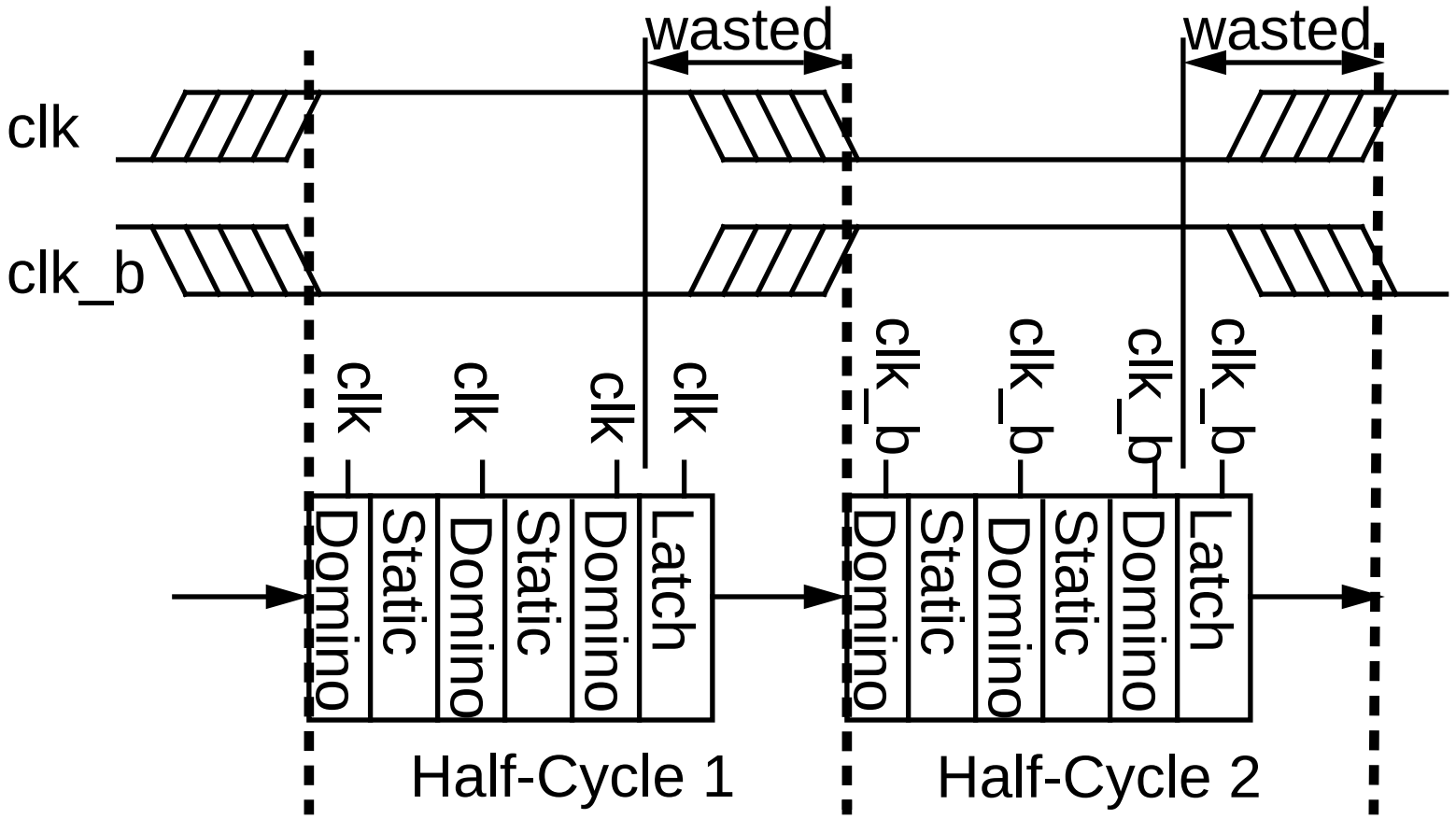
Ideal Domino Clocking



Ping-pong between half-cycles

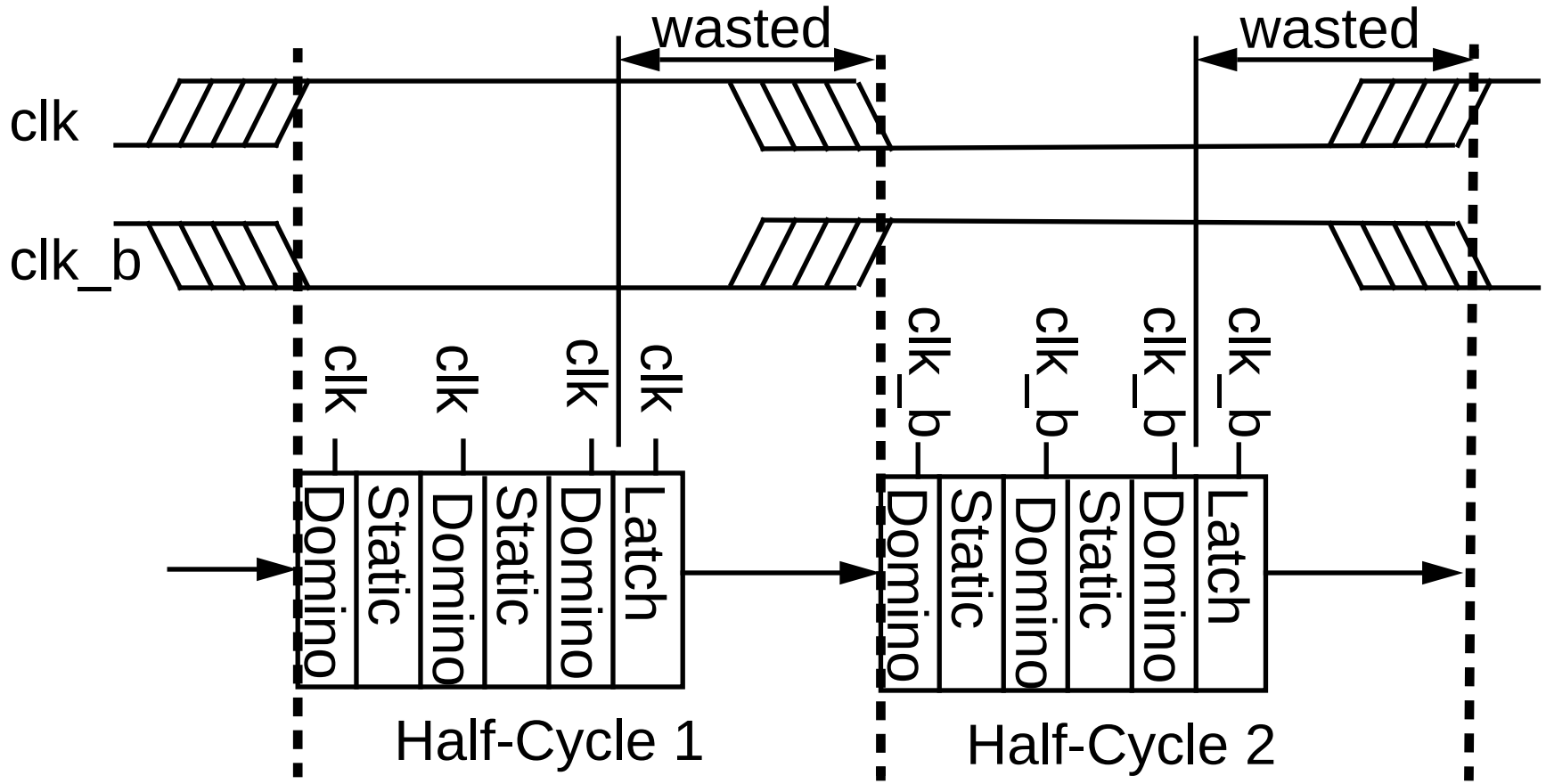
- **One evaluates while other precharges**

Clock Skew



- **Evaluation begins at latest rising edge**
- **Latch input setup before earliest falling edge**
- **Clock skew twice each cycle**

Balancing Logic



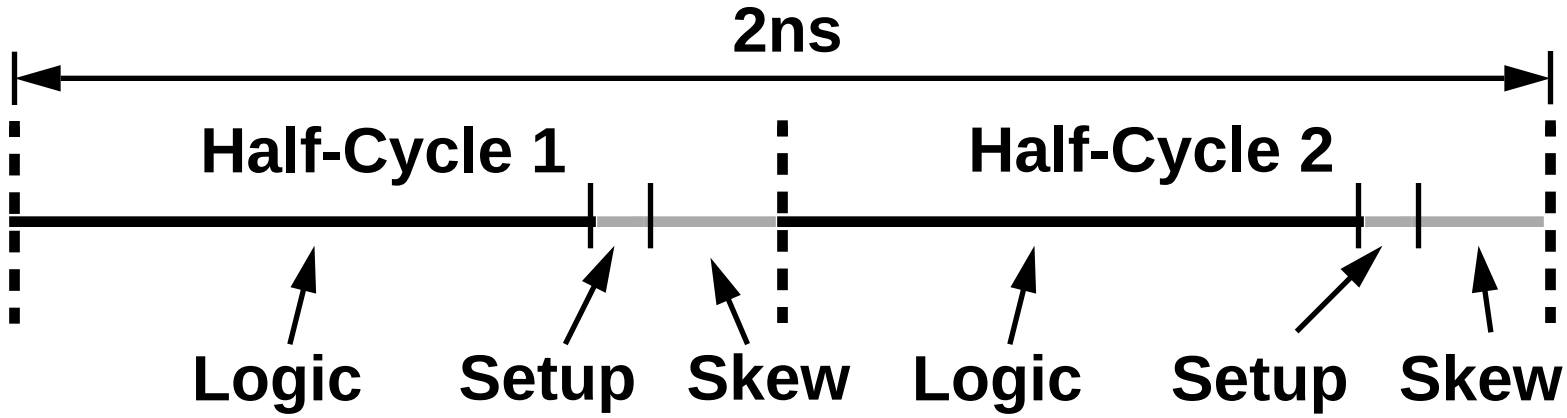
Logic may not exactly fit half-cycle

- **No flexibility to borrow time**

How bad is it?

Consider 500 MHz DEC Alpha:

- **T = 2ns**
- **t_{skew} = 200 ps (reported budget)**
- **t_{setup} = 50 ps (optimistic estimate)**

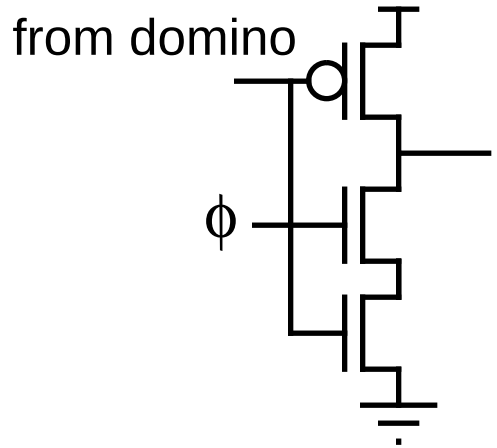


- **25% of cycle consumed by skew & setup**

Prior Art

Assuming monotonic inputs to domino logic:

- **Self-timed circuits (HaL Divider)**
- **Use better latch:**



TSPC Latch

- **Remove latch (Alpha 21164)**

General Solution

How could Alpha remove latches?

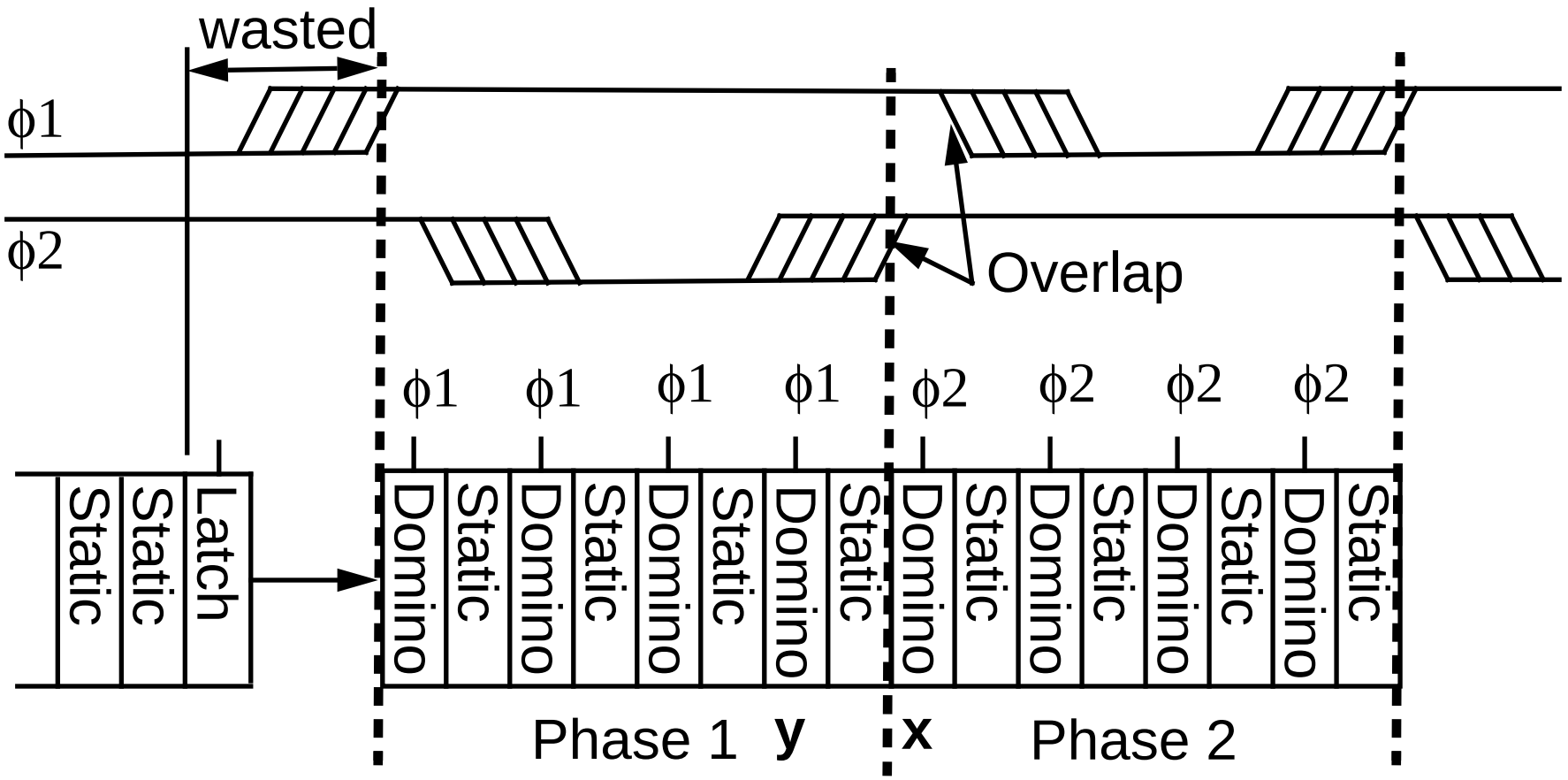
Function of latches:

- **(1) Prevent glitches on domino input**
- **(2) Hold result during precharge**

Latches can be safely eliminated if:

- **(1) Input is from domino logic (no glitches)**
- **(2) Result is consumed before precharge**

Skew-Tolerant Domino Circuits

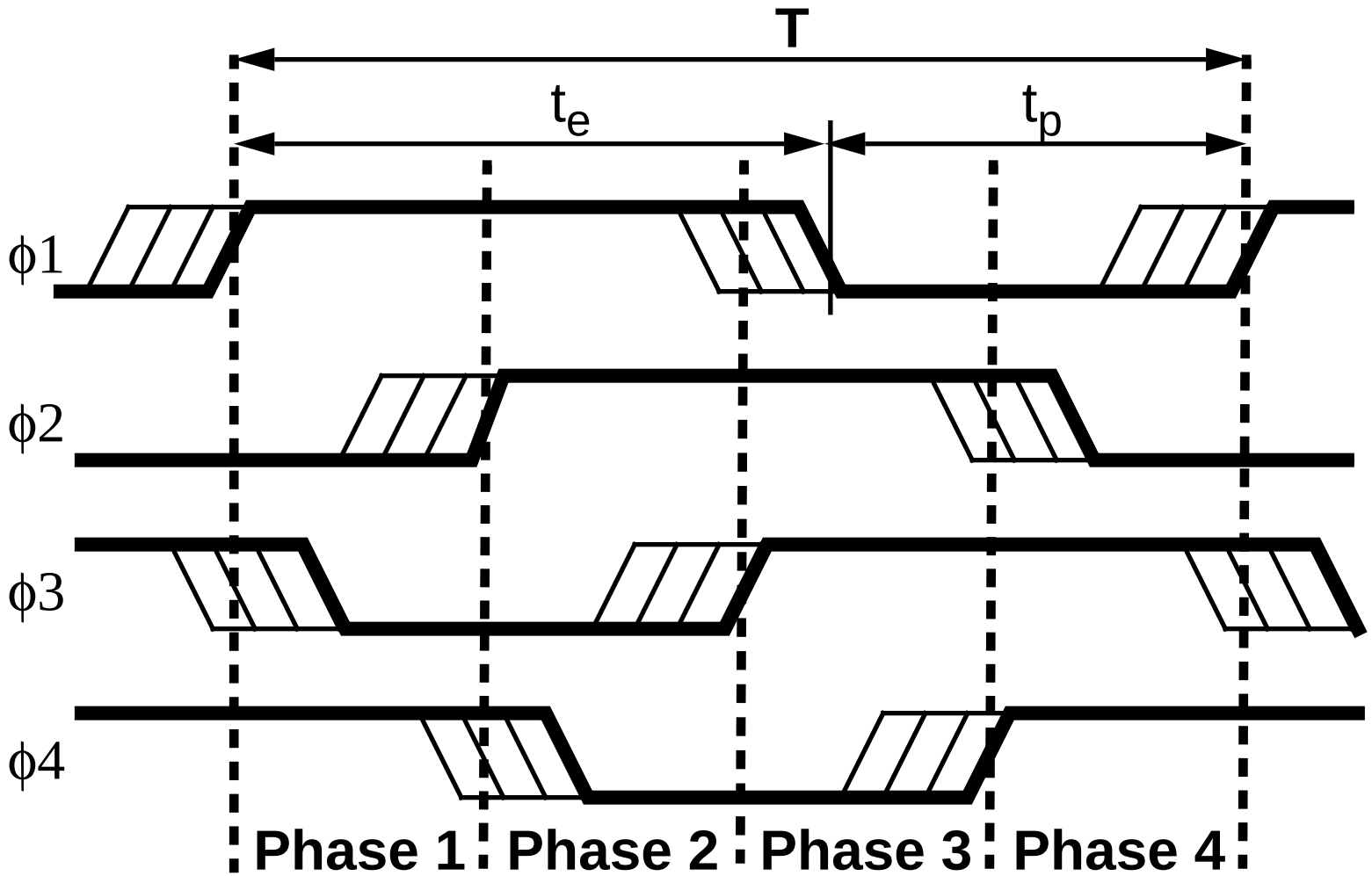


Overlap clocks so x evaluates before y precharges

- **We can remove latches within domino pipe**
- **Still budget skew from static → domino**

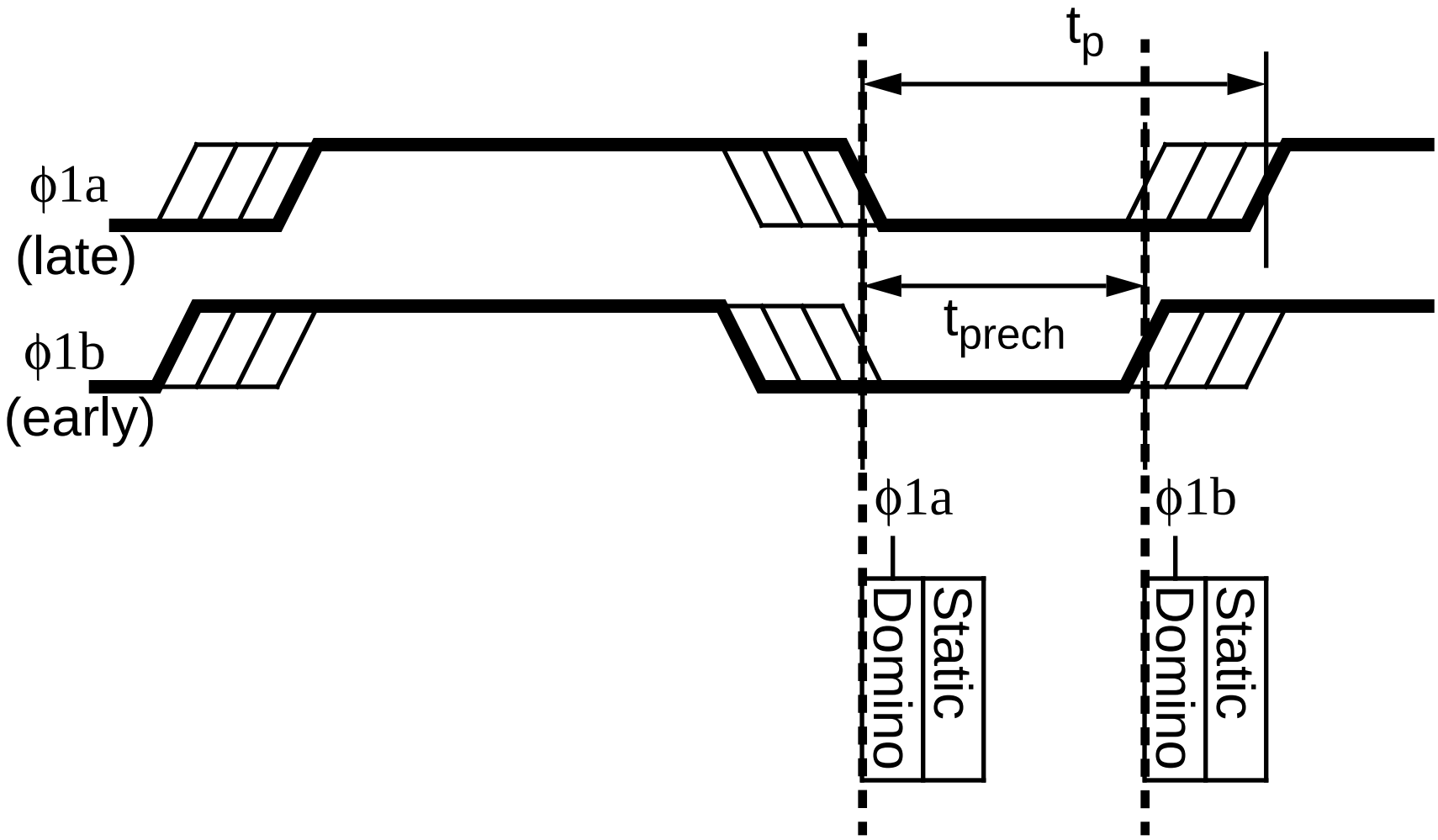
Skew-Tolerant Waveforms

In general: N overlapping clock phases



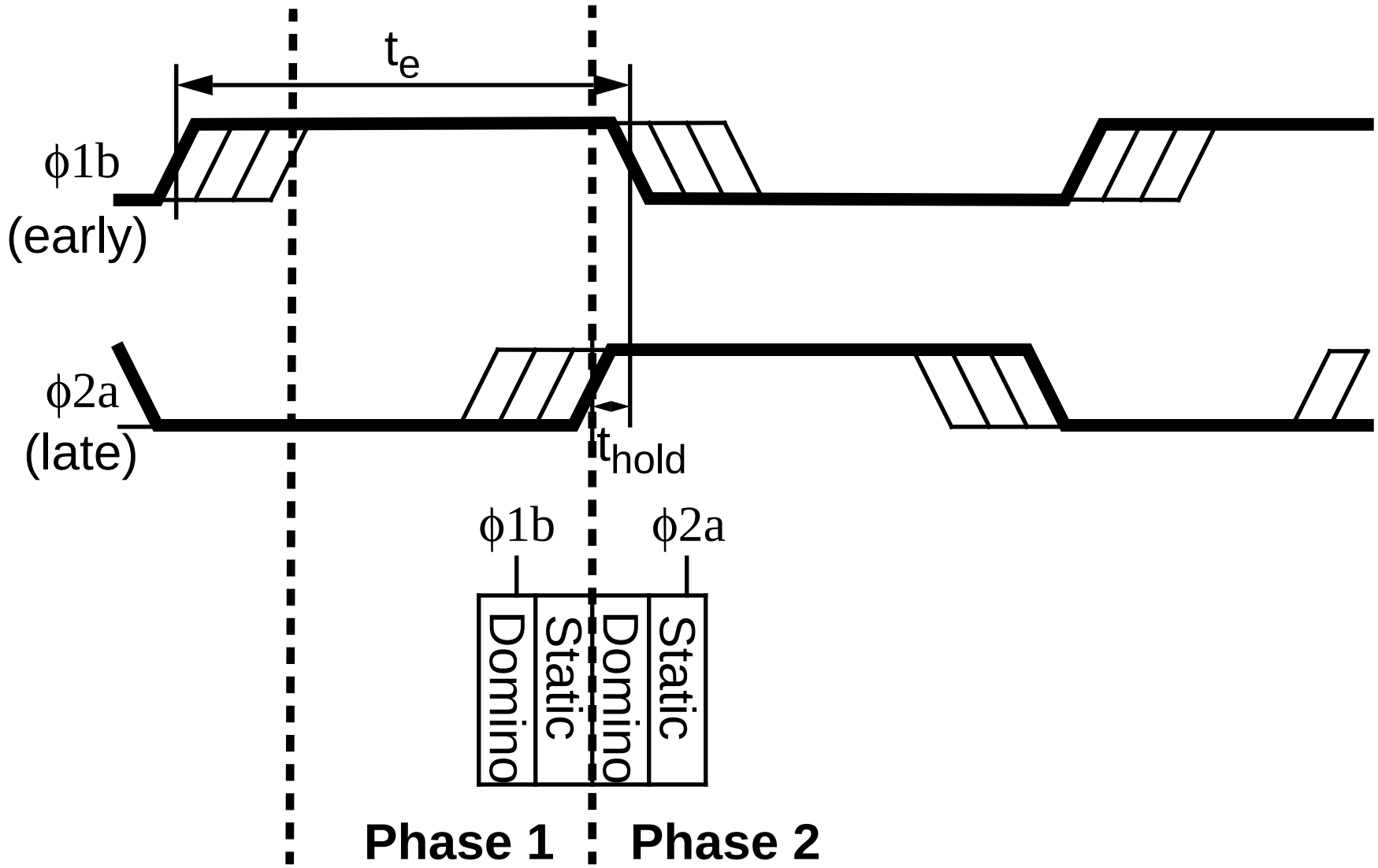
- **Identical, each delayed T/N from the previous**

Precharge Time (t_p)



Precharge: $t_p = t_{prech} + t_{skew}$

Evaluation Time (t_e)



Overlap: $t_e = T/N + t_{skew} + t_{hold}$

Skew Tolerance

We've found the optimal duty cycle

- **Precharge: $t_p = t_{prech} + t_{skew}$**
- **Overlap: $t_e = T/N + t_{skew} + t_{hold}$**

Solve for the maximum tolerable skew

- $$t_{skew\text{-}max} = \frac{\frac{N-1}{N} T - t_{prech} - t_{hold}}{2}$$
- **Skew tolerance increases with N**

Local Skew & Time Borrowing

Local clock domains

- **Reduce skew within a phase of logic**
- **Thereby to tolerate more skew globally**

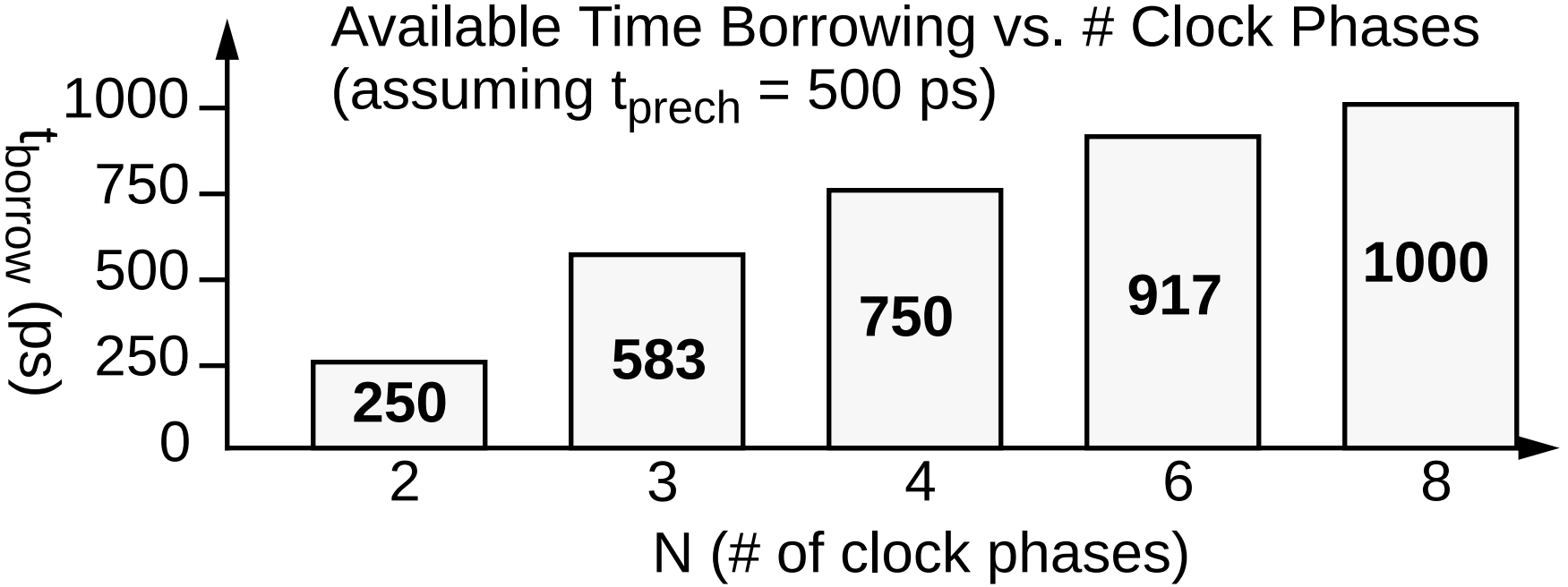
Time borrowing

- **Clocks may overlap by more than global skew**
- **Overlap allows time borrowing into next phase**

Example

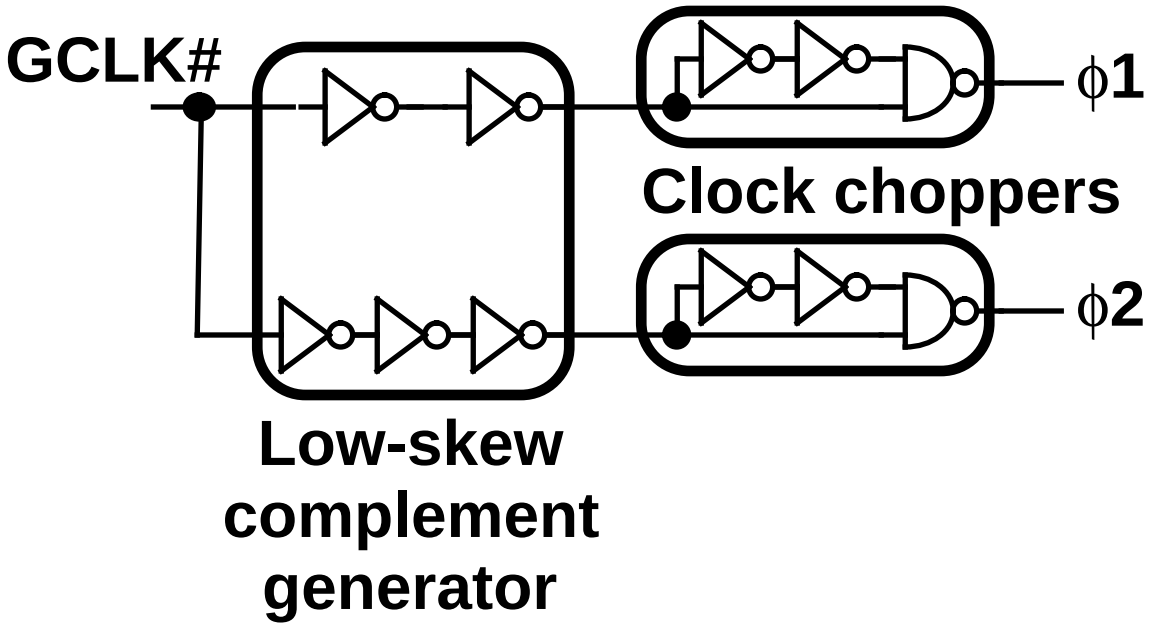
Again consider 500 MHz DEC Alpha:

- **T = 2ns**
- **t_{skewG} = 200 ps / t_{skewL} = 50 ps**
- **t_{hold} = 0 (conservative)**



2-Phase Clock Generation

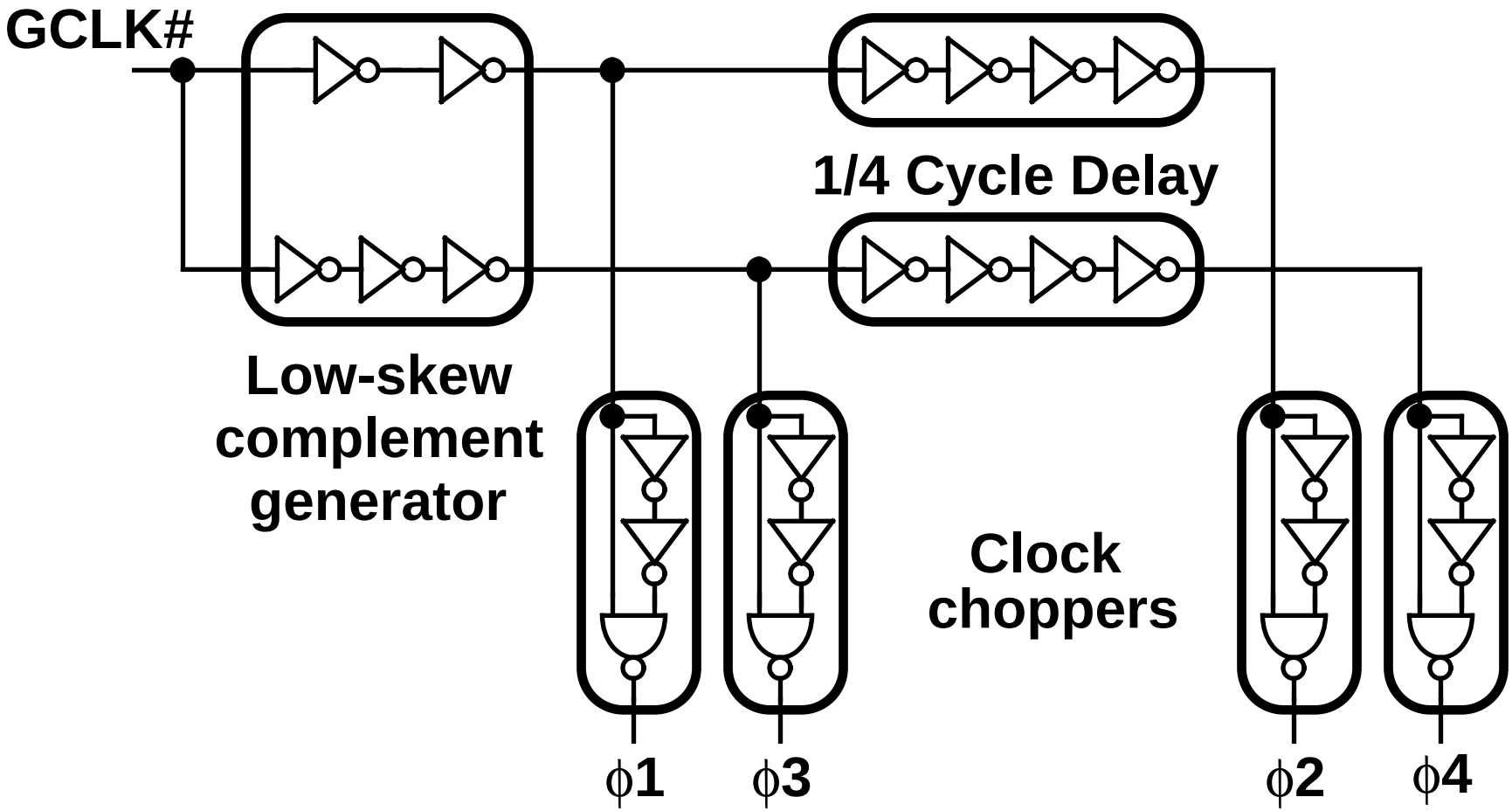
Delay falling edge with clock chopper



4-Phase Clock Generation

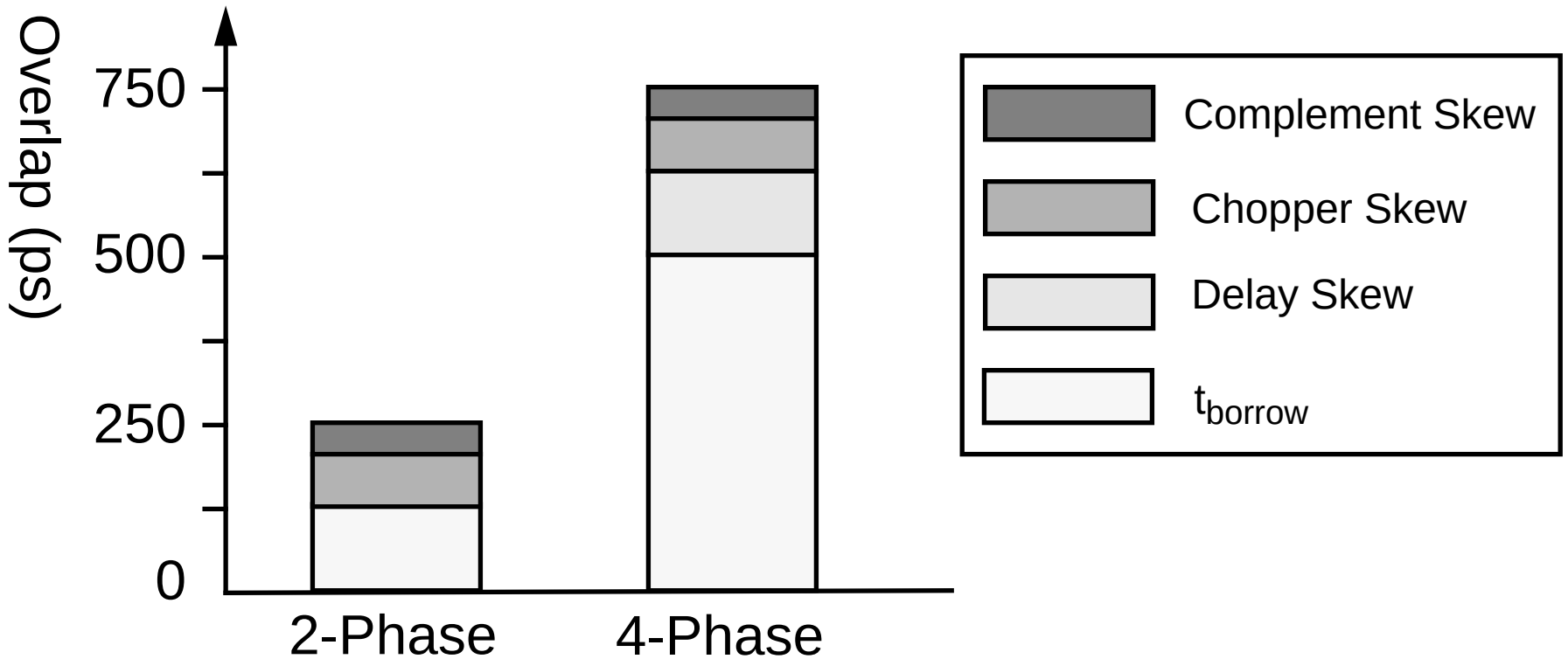
Inverters delay ϕ2 and ϕ4 by 1/4 cycle

- **4-phase has 1/4 cycle extra nominal overlap**



Clock Generator Skew Analysis

How much skew does the clock generator cause?

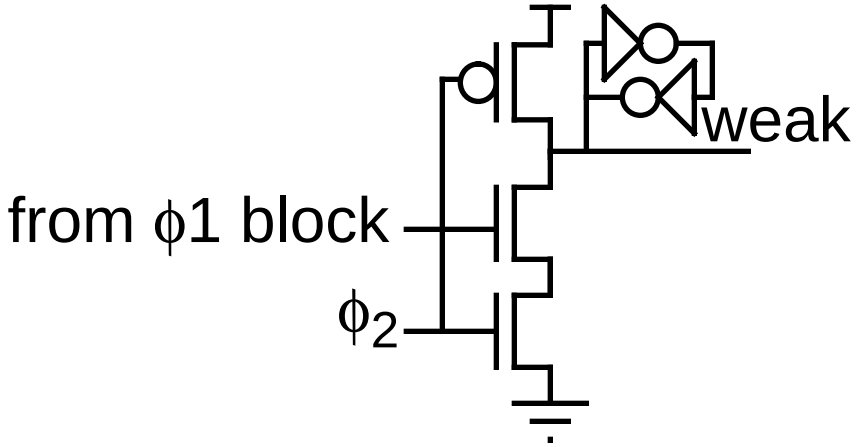


- **Variation in delay 20-30% relative to other clks**
- **4-Phase is still a large net benefit**

Other Design Issues

State stored in first gate of domino phase

- **Use a "full keeper" to allow stop-clock**



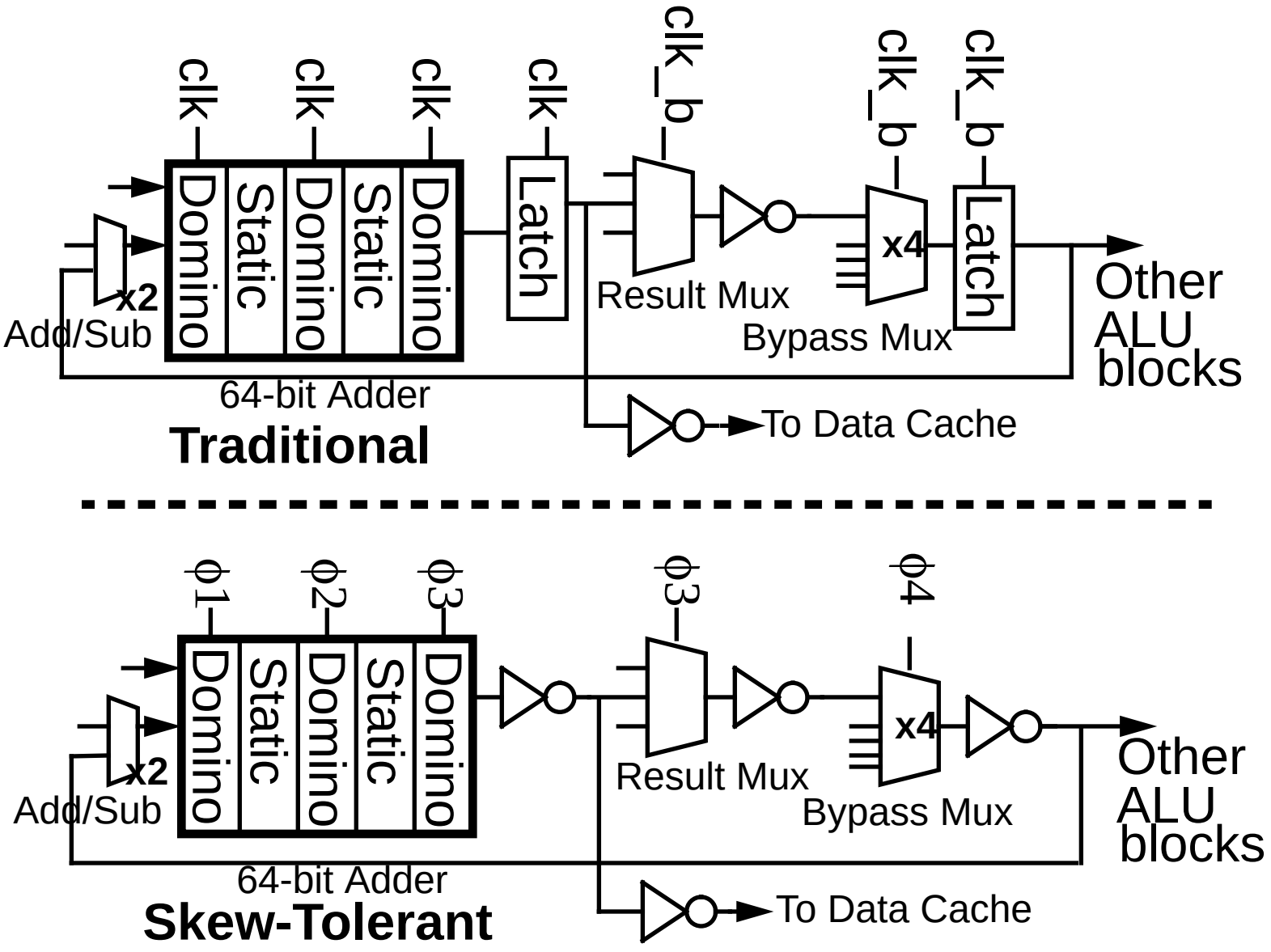
- **Scan can be added to domino gate**

Min-Delay

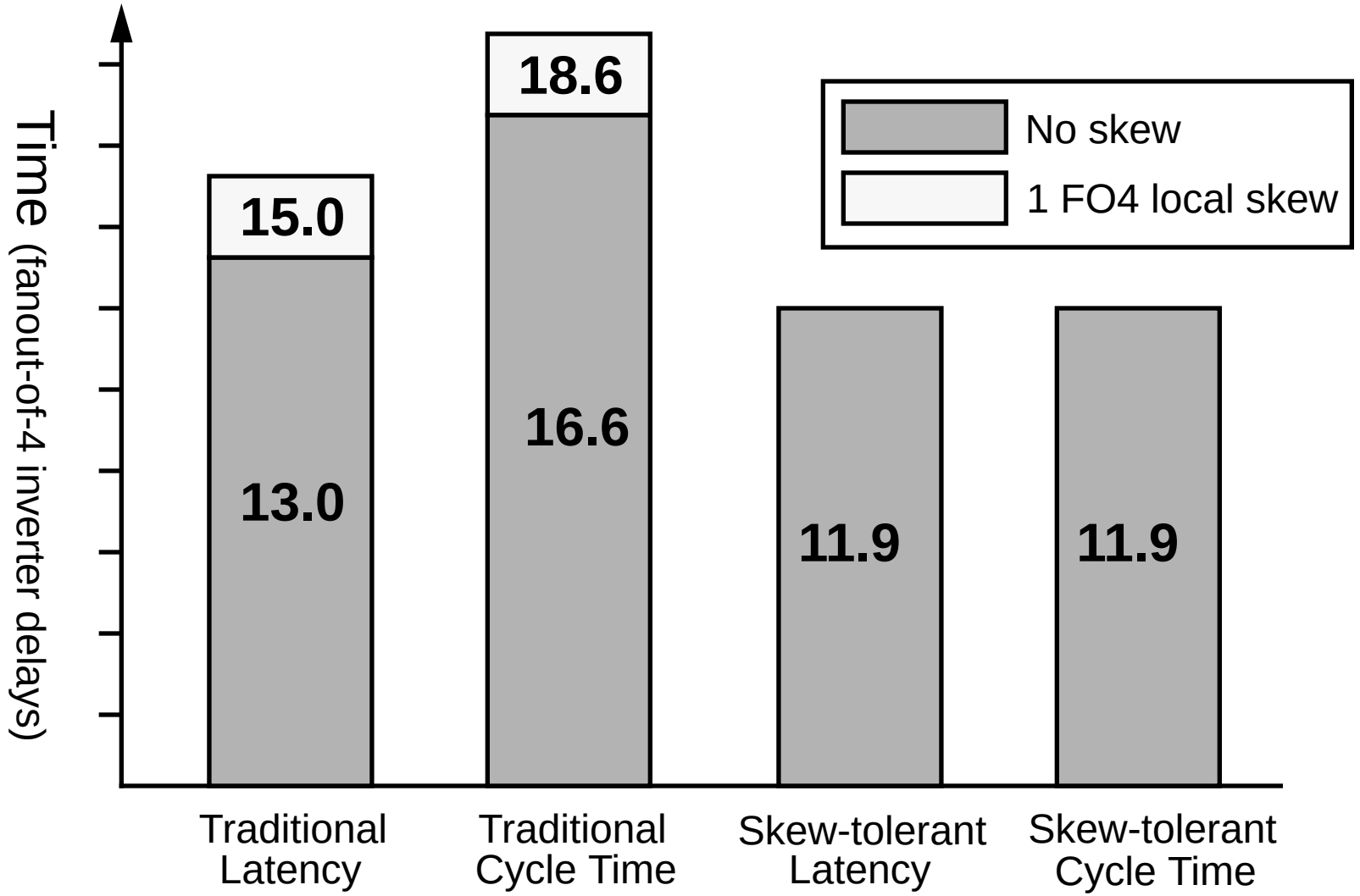
- **Check when t_{skew} is large**

Performance Evaluation

- **64 bit superscalar ALU self-bypass path**



Simulation Results



- **Skew-tolerant domino 25%$^{+}$ faster**

Conclusion

- Domino very attractive for high speed ICs
- Traditional domino limited by overhead
- Skew-tolerant domino eliminates overhead
- Clock generation
- Expect widespread use