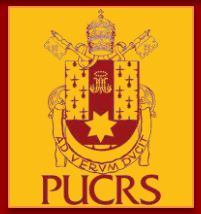
Blade – A Timing Violation Resilient Asynchronous Template

Dylan Hand*, Matheus Trevisan Moreira*†, Hsin-Ho Huang*, Danlei Chen*, Frederico Butzke‡, Zhichao Li*, Matheus Gibiluka†, Melvin Breuer*, Ney Laert Vilar Calazans†, and Peter A. Beerel*

May 4th, 2015

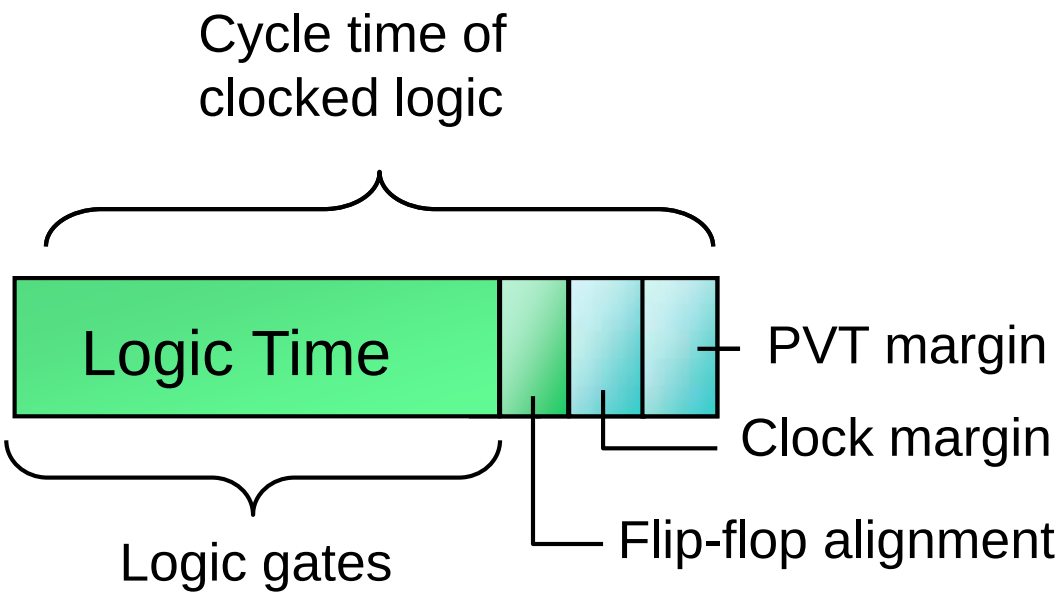
* University of Southern California, Los Angeles, CA
† Pontifícia Universidade Católica do Rio Grande do Sul, Porto Alegre, Brazil
‡ Universidade de Santa Cruz do Sul, Santa Cruz do Sul, Brazil

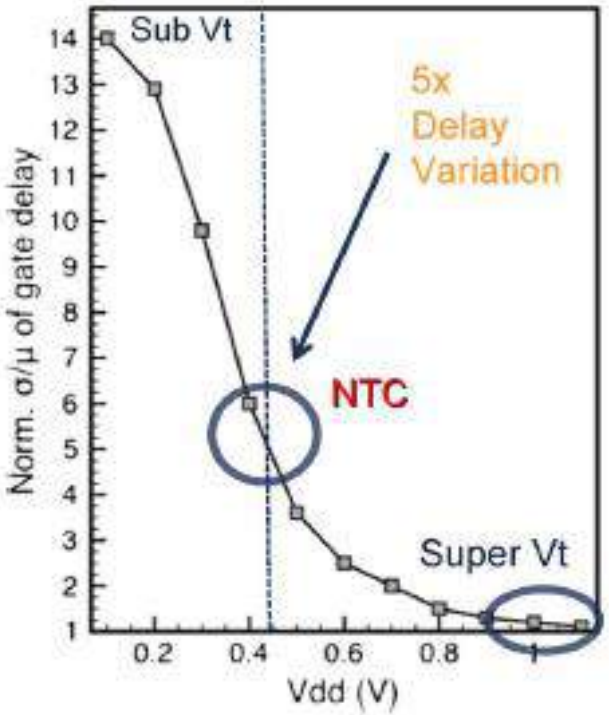
USC Viterbi School of Engineering



University of Southern California

Delay Overheads



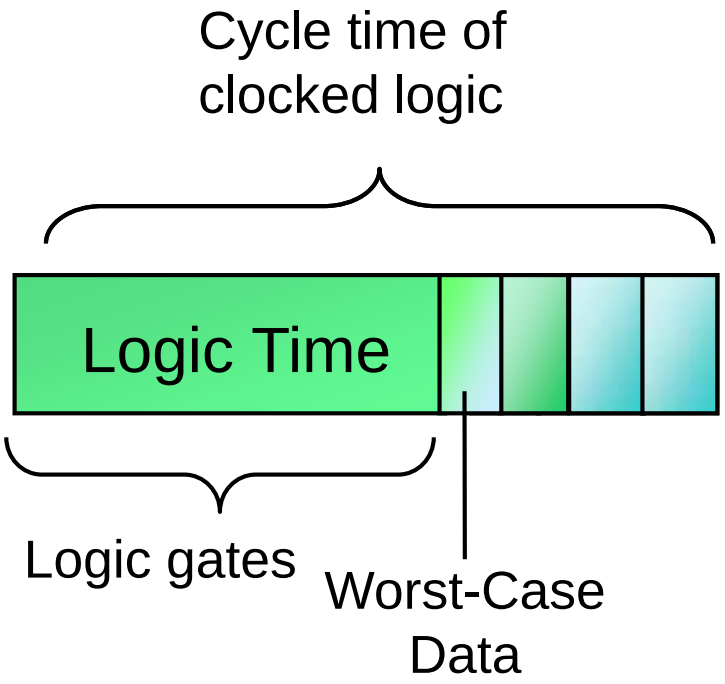


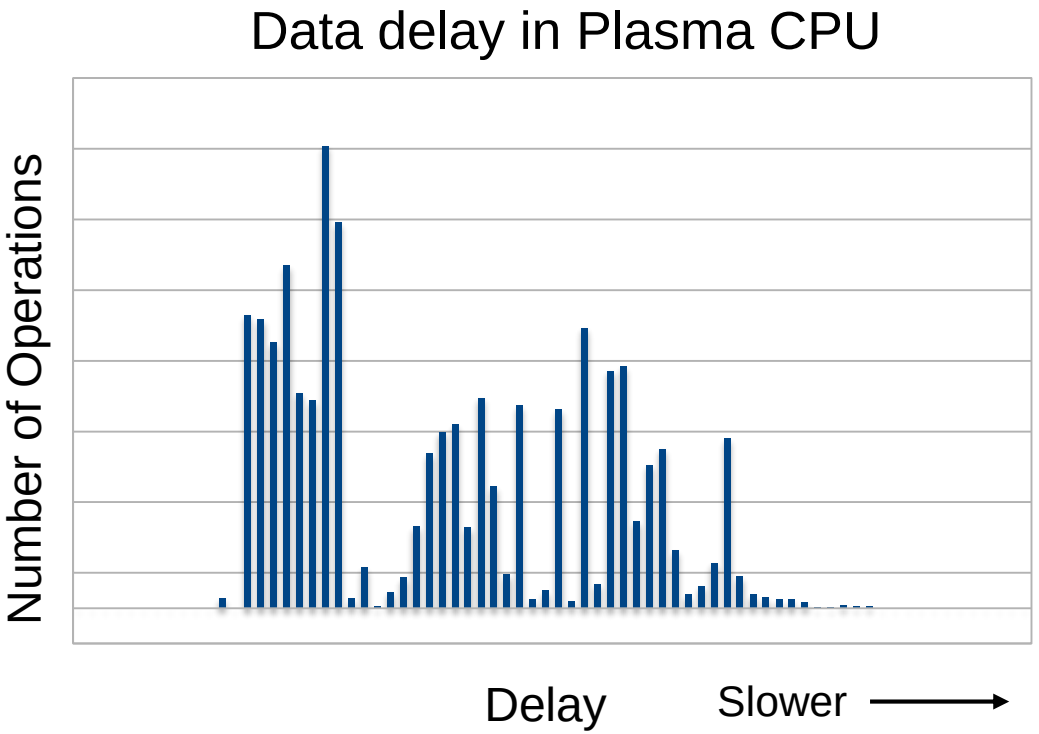
[Dreslinksi et al., IEEE Proc. 2010]

Traditional synchronous design suffers from increased margins

- Worse at low and near-threshold regions

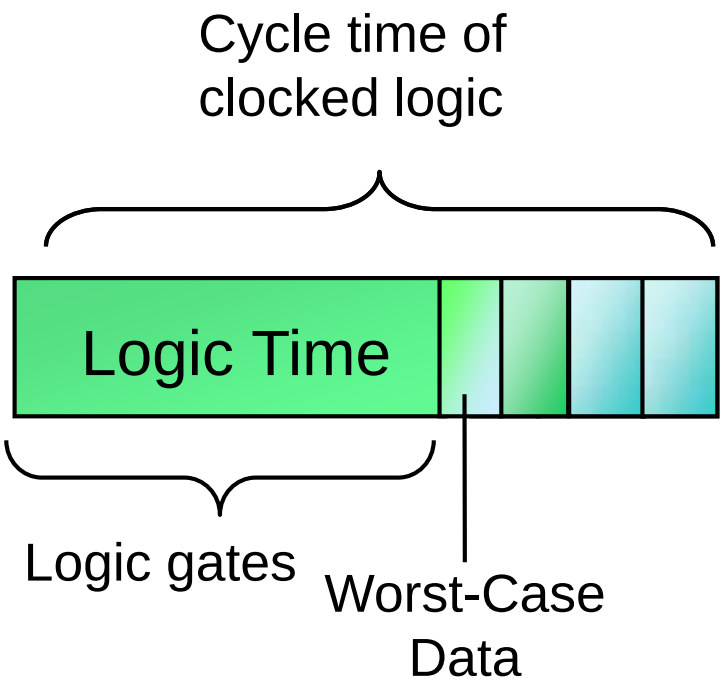
Data Dependent Delays

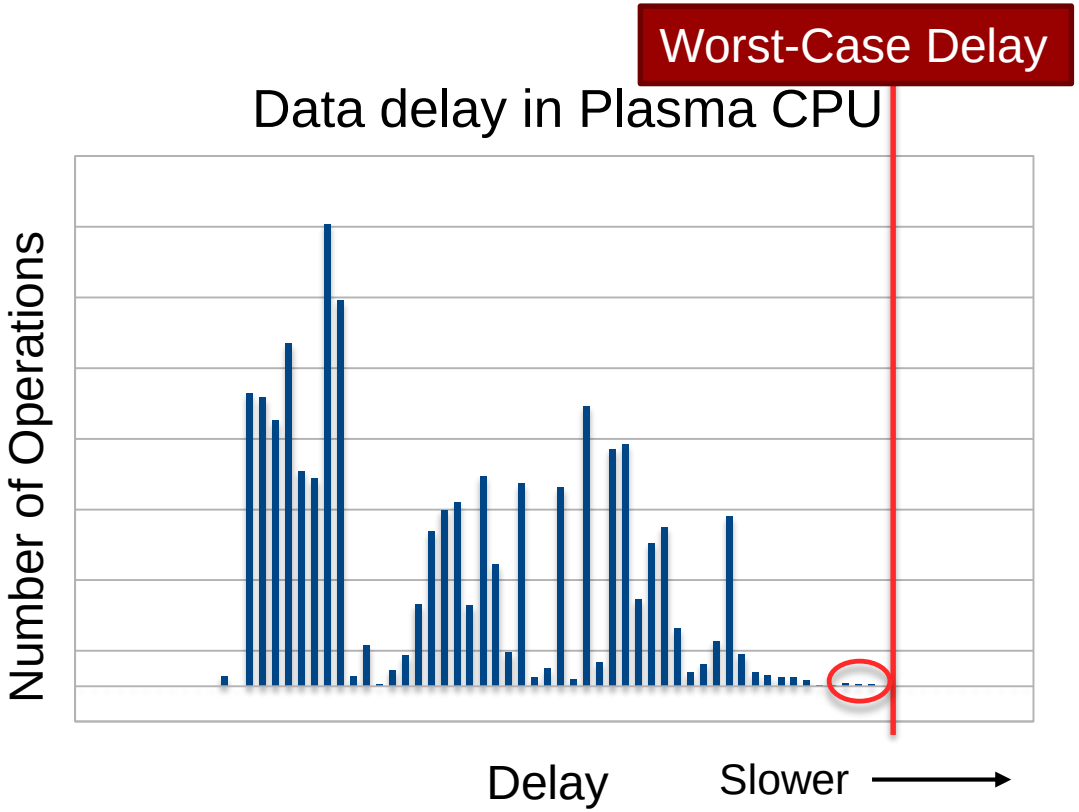




Delay variation due to data is rarely exploited in traditional designs

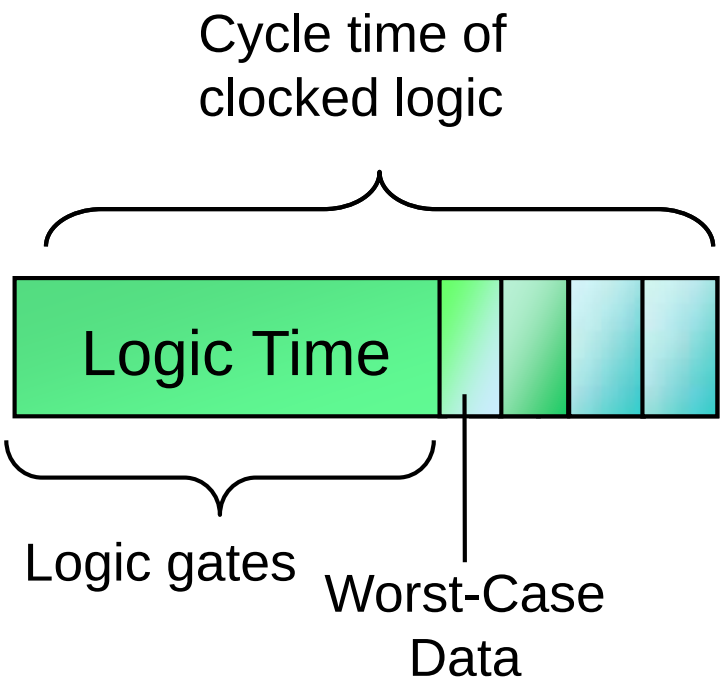
Data Dependent Delays

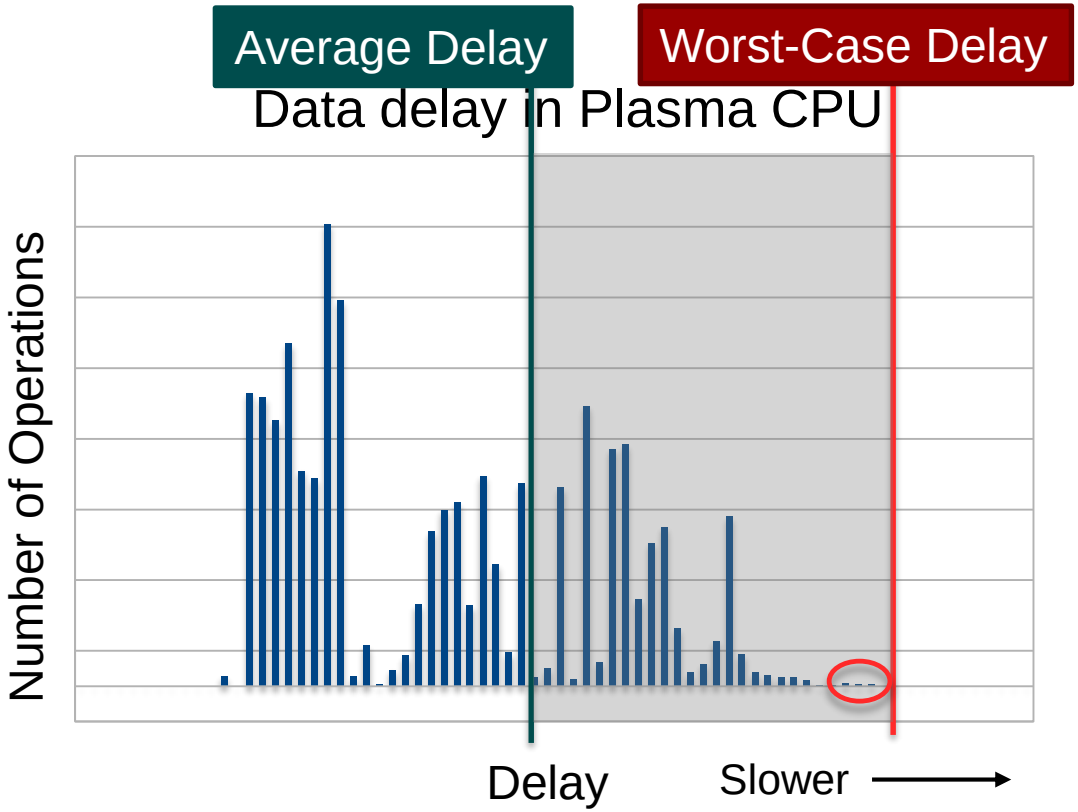




Delay variation due to data is rarely exploited in traditional designs

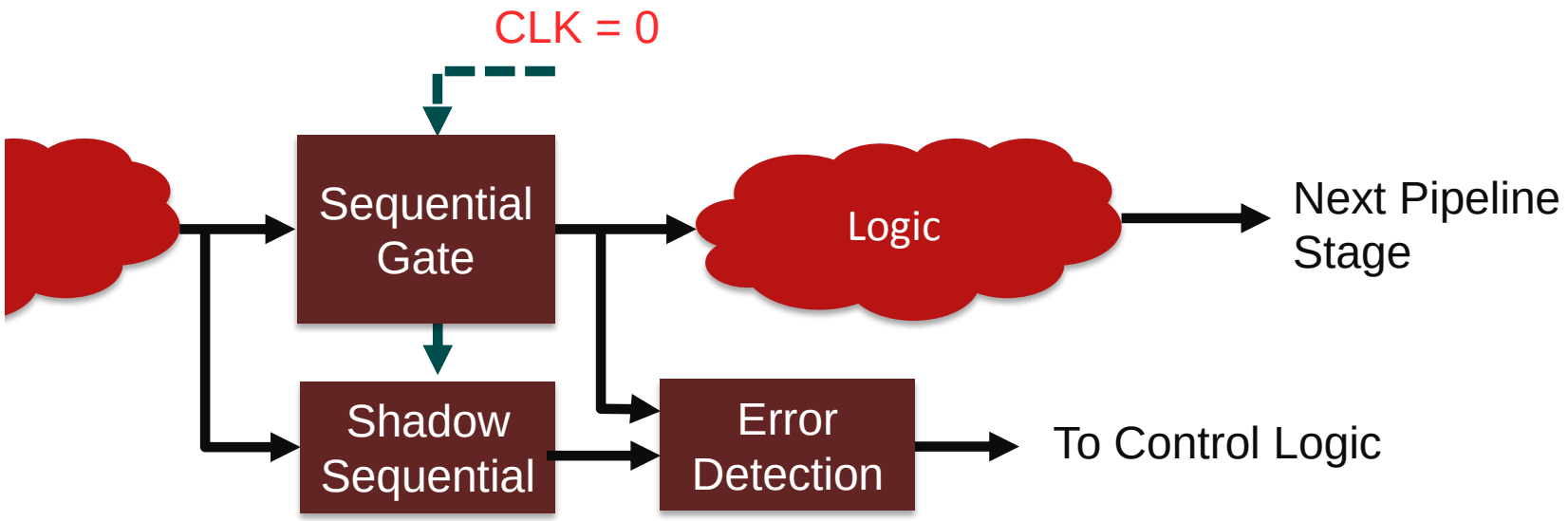
Data Dependent Delays





Delay variation due to data is rarely exploited in traditional designs

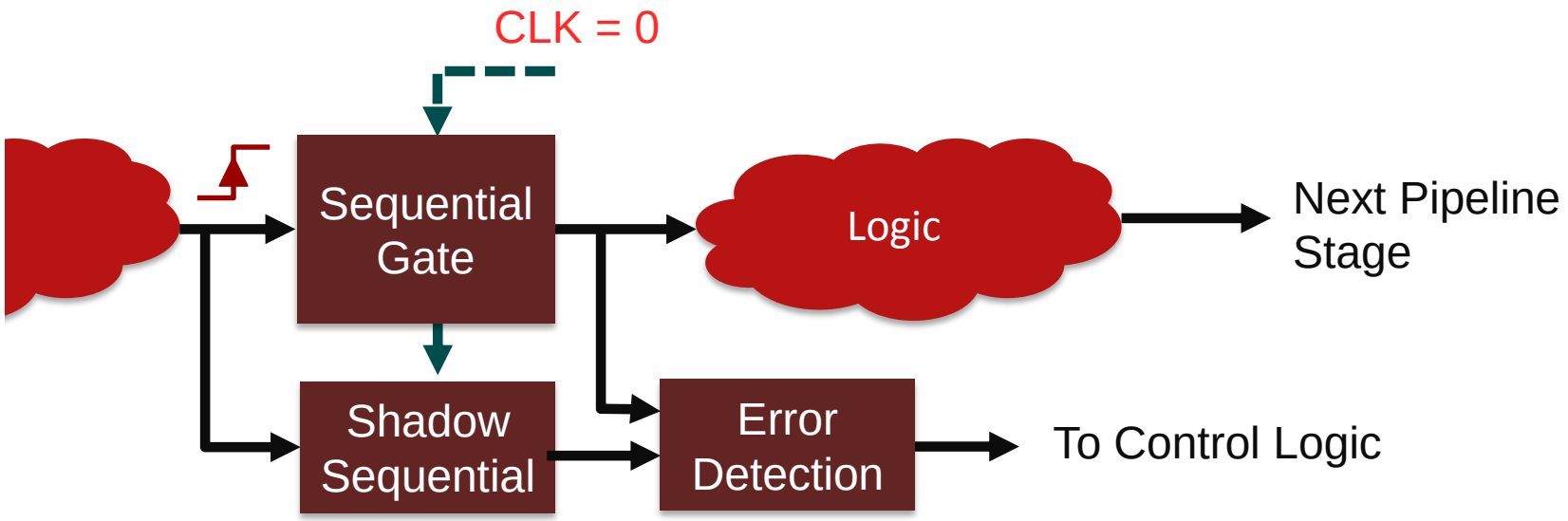
Resilient Design



Allow and detect timing errors in datapath

- Correct via architectural replay or gating/pausing clock

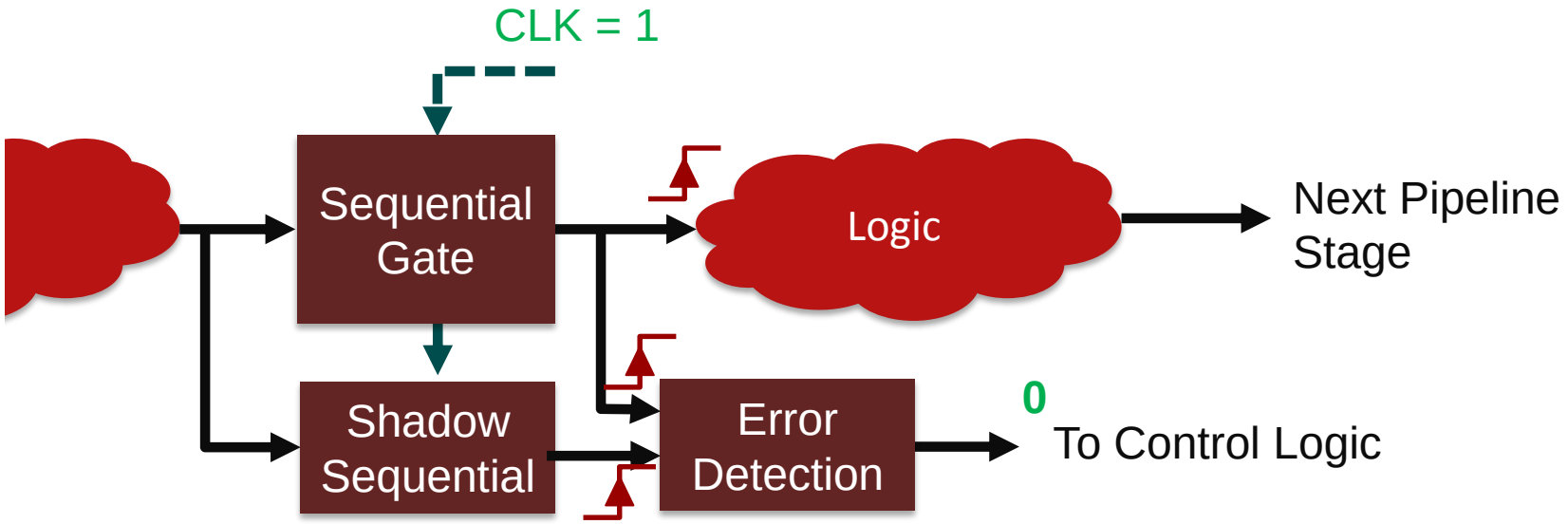
Resilient Design



Allow and detect timing errors in datapath

- Correct via architectural replay or gating/pausing clock

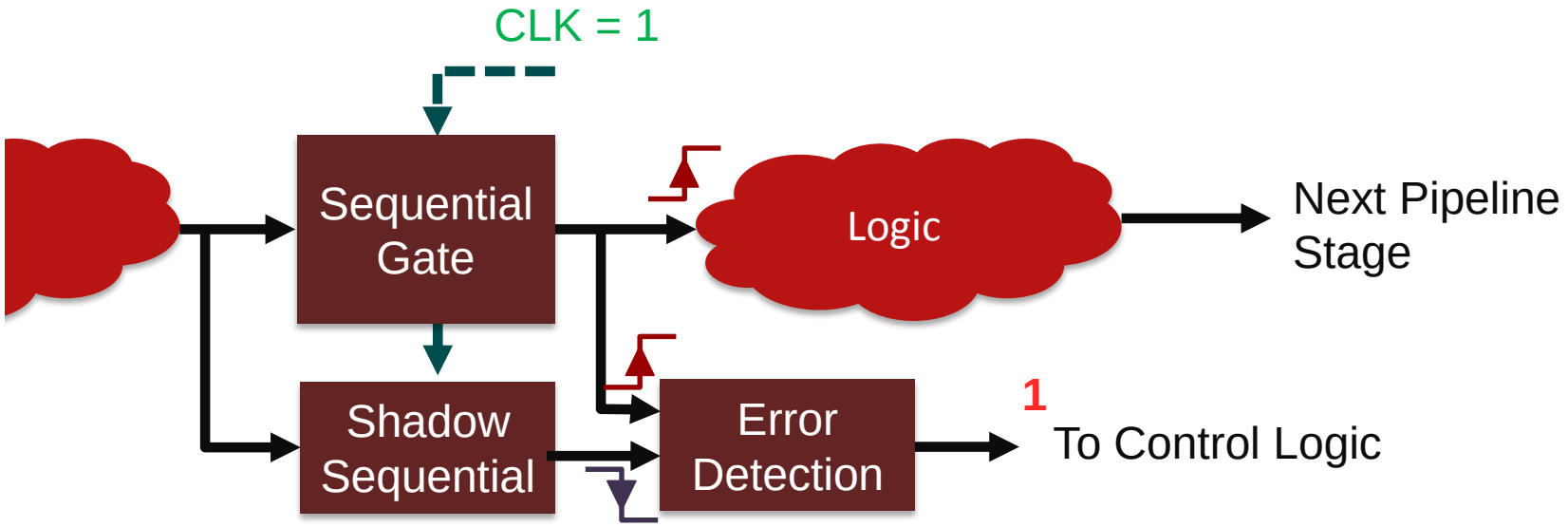
Resilient Design



Allow and detect timing errors in datapath

- Correct via architectural replay or gating/pausing clock

Resilient Design

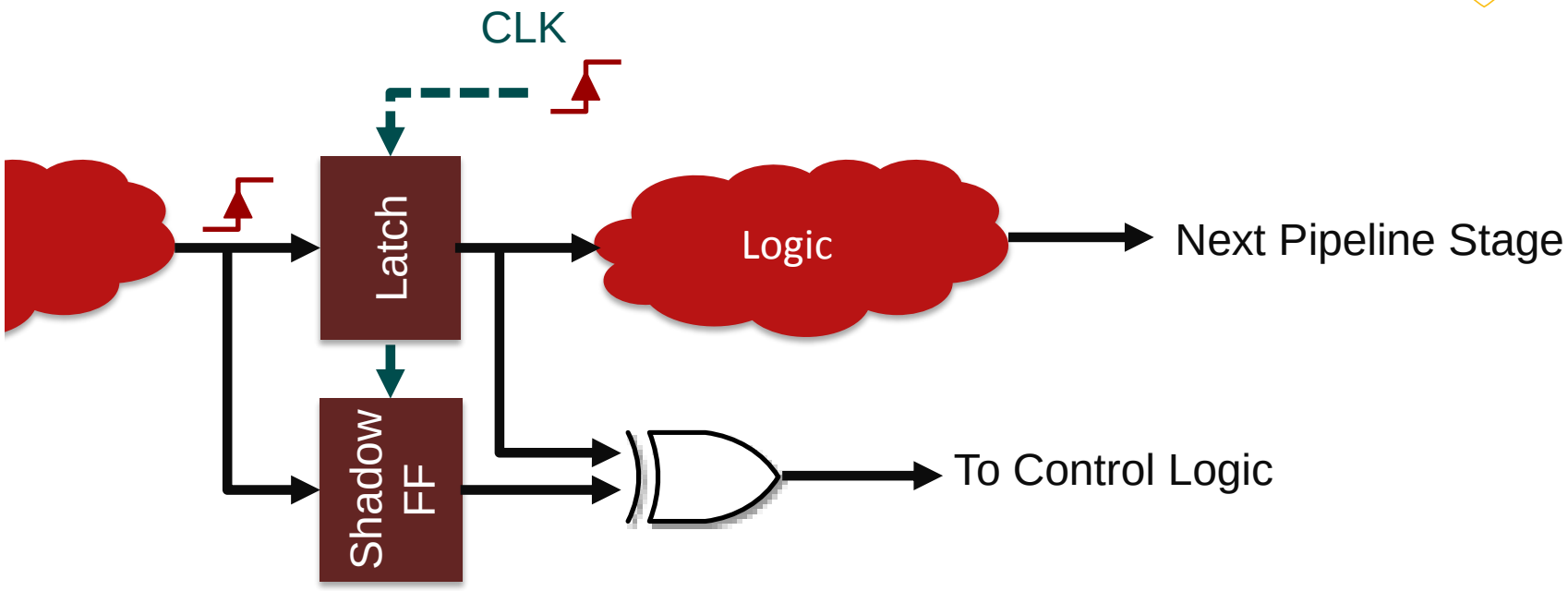


Allow and detect timing errors in datapath

- Correct via architectural replay or gating/pausing clock

Effective approaches have been elusive!

Metastability Issue

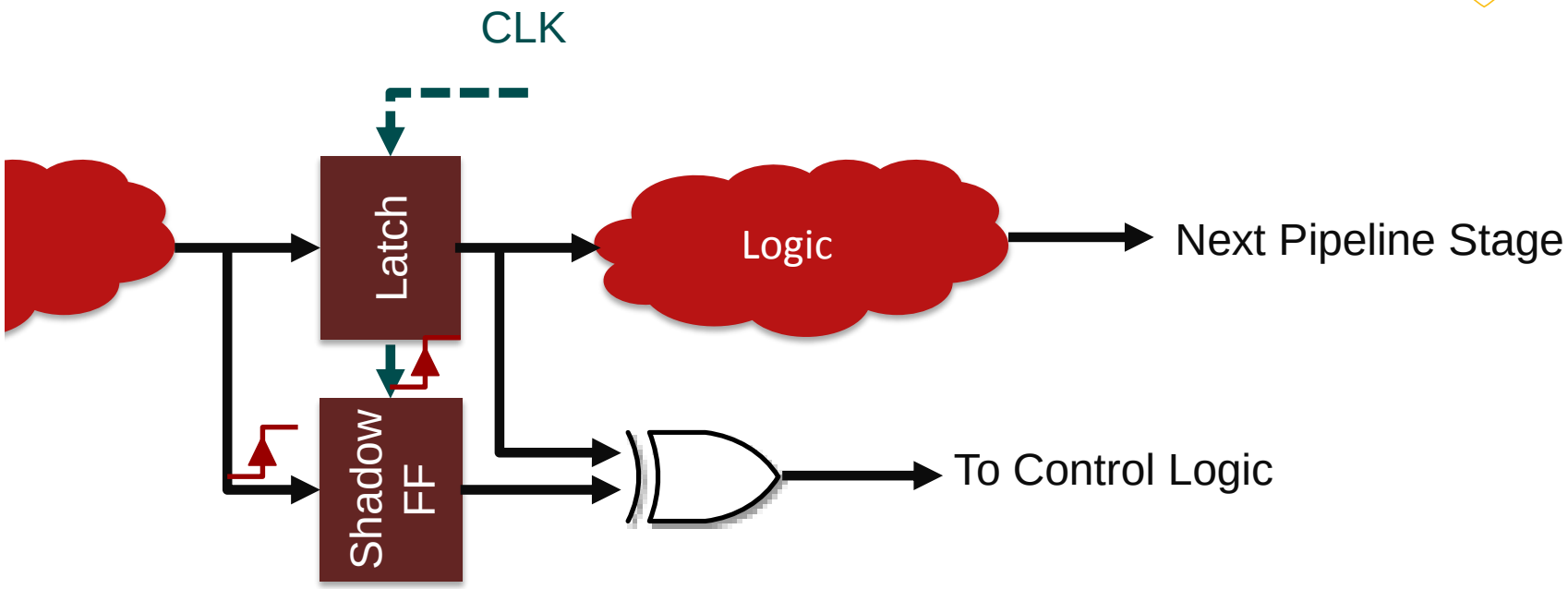


[Bubble Razor, Fojtik, 2013]

The Problem [Beer et al, 2014]

- Flop metastability can propagate to ERR signal
- Metastability in control path can cause system failure

Metastability Issue

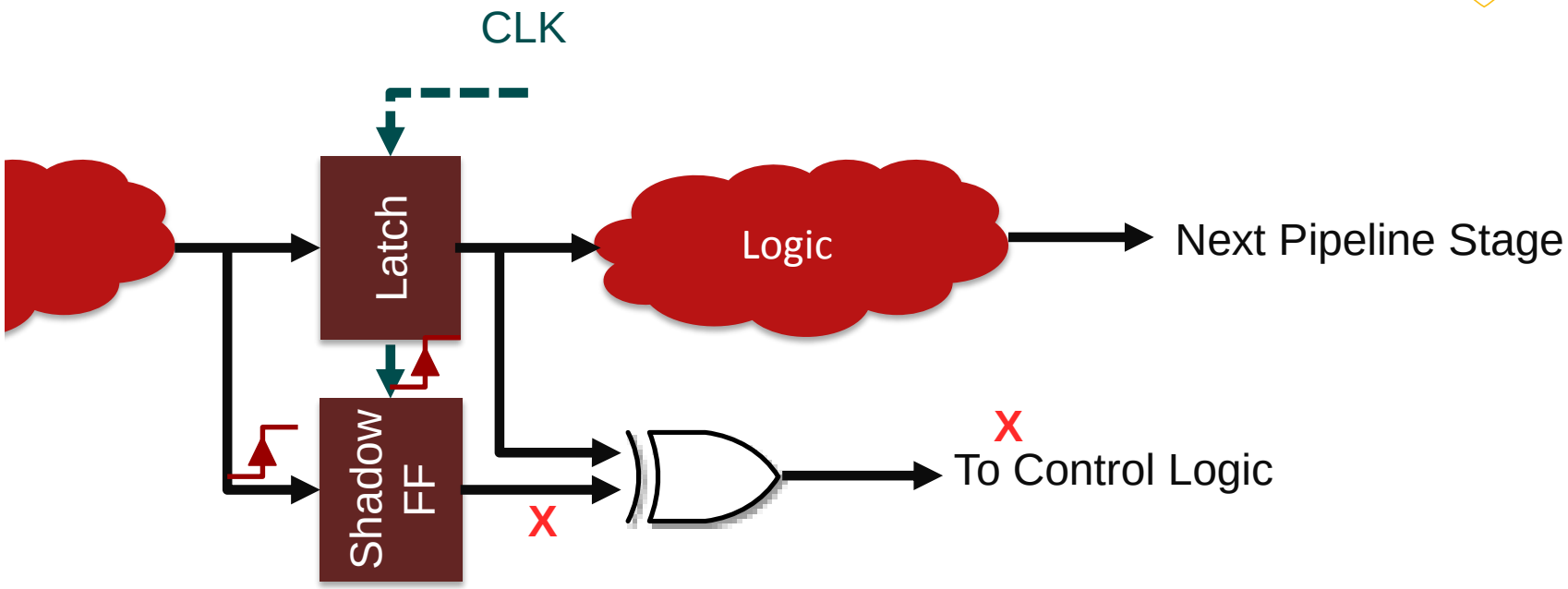


[Bubble Razor, Fojtik, 2013]

The Problem [Beer et al, 2014]

- Flop metastability can propagate to ERR signal
- Metastability in control path can cause system failure

Metastability Issue

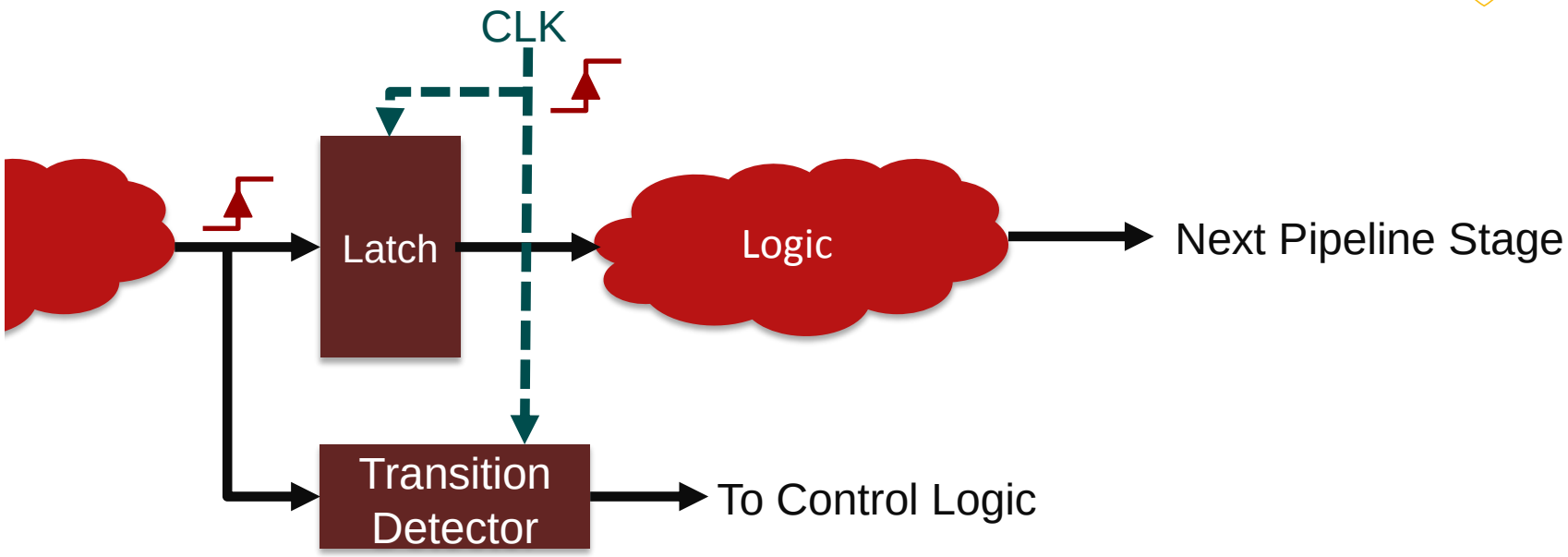


[Bubble Razor, Fojtik, 2013]

The Problem [Beer et al, 2014]

- Flop metastability can propagate to ERR signal
- Metastability in control path can cause system failure

Handling Metastability

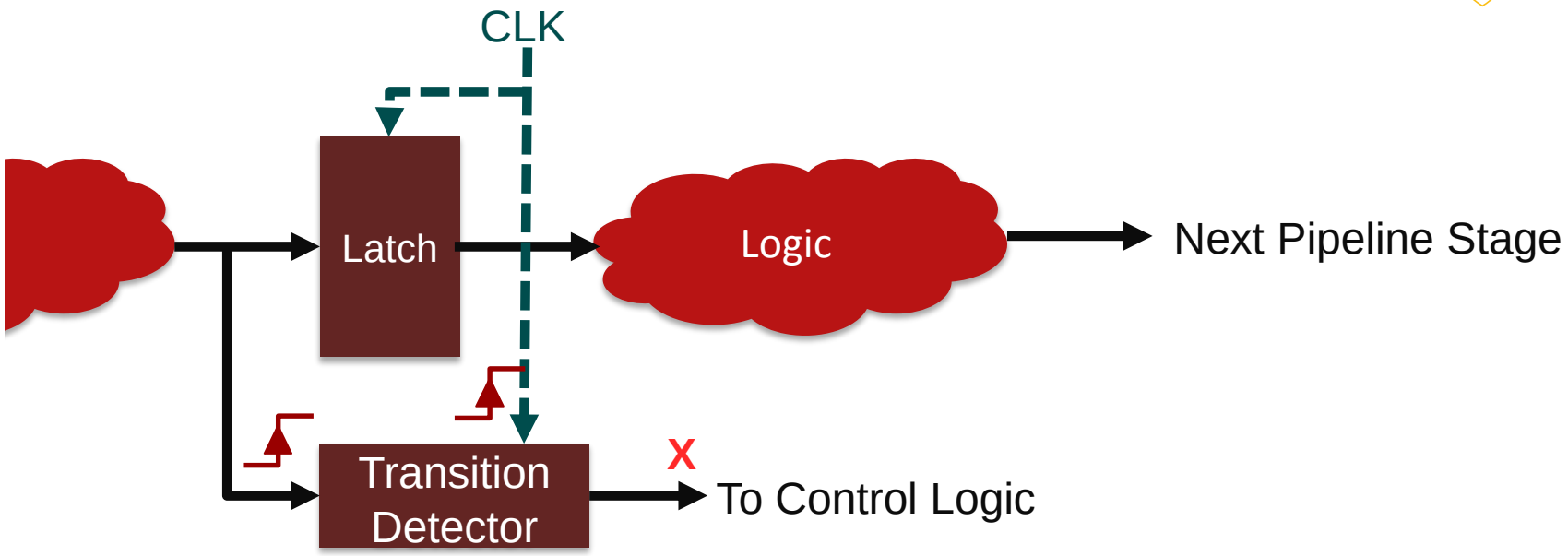


[RazorII, Das, 2009]

Transition detector may exhibit metastability

- Error signal must go through synchronizer, increasing delay
- Uses architectural replay to recover from errors

Handling Metastability

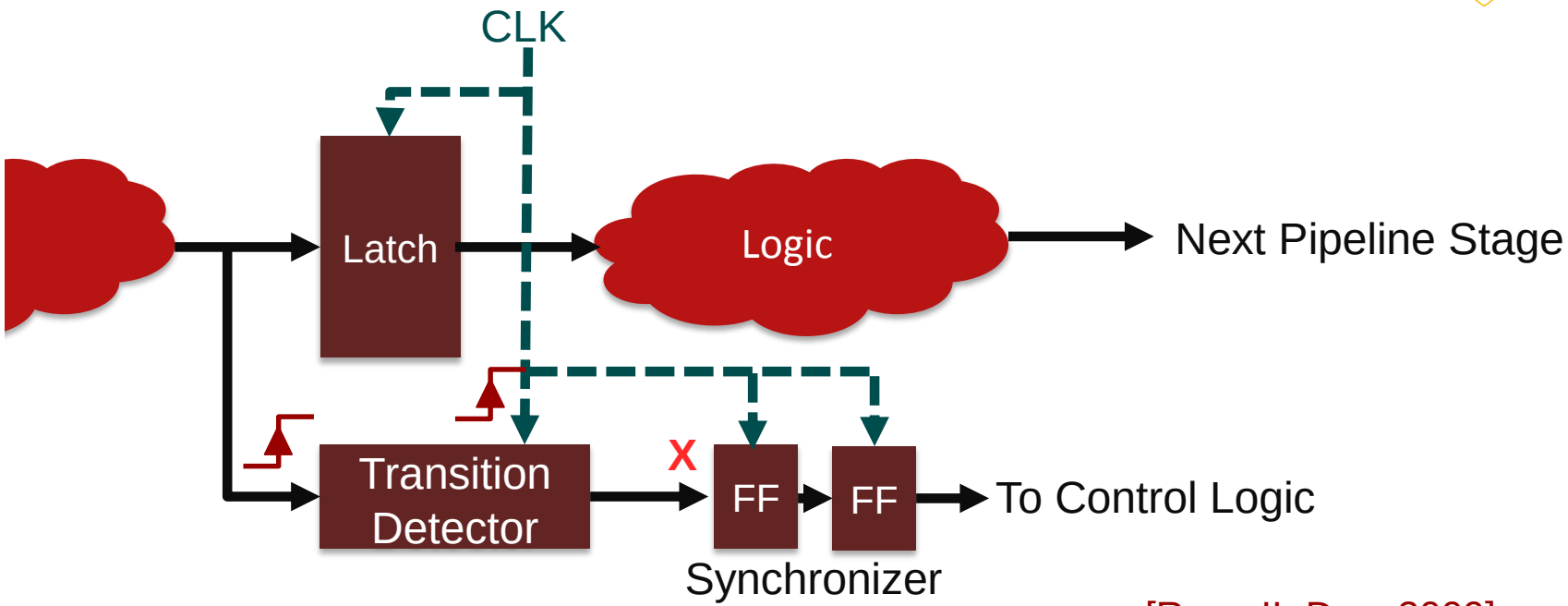


[RazorII, Das, 2009]

Transition detector may exhibit metastability

- Error signal must go through synchronizer, increasing delay
- Uses architectural replay to recover from errors

Handling Metastability

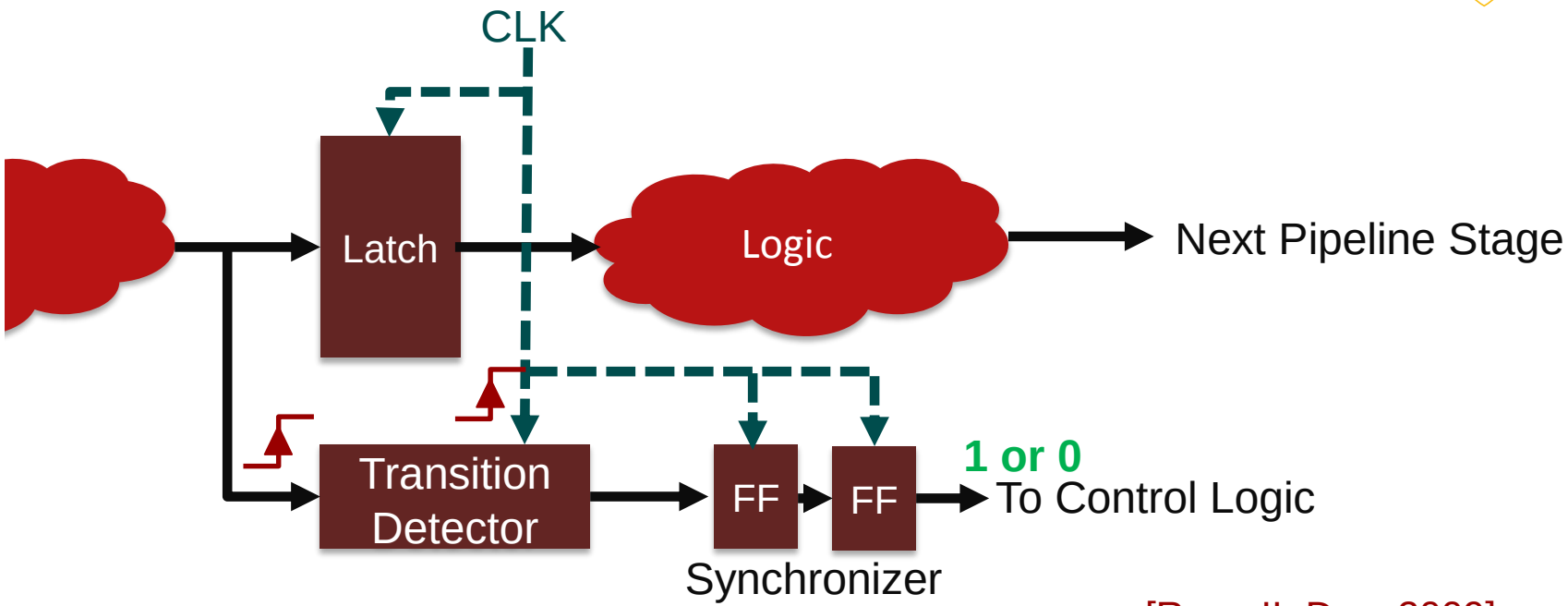


[RazorII, Das, 2009]

Transition detector may exhibit metastability

- Error signal must go through synchronizer, increasing delay
- Uses architectural replay to recover from errors

Handling Metastability

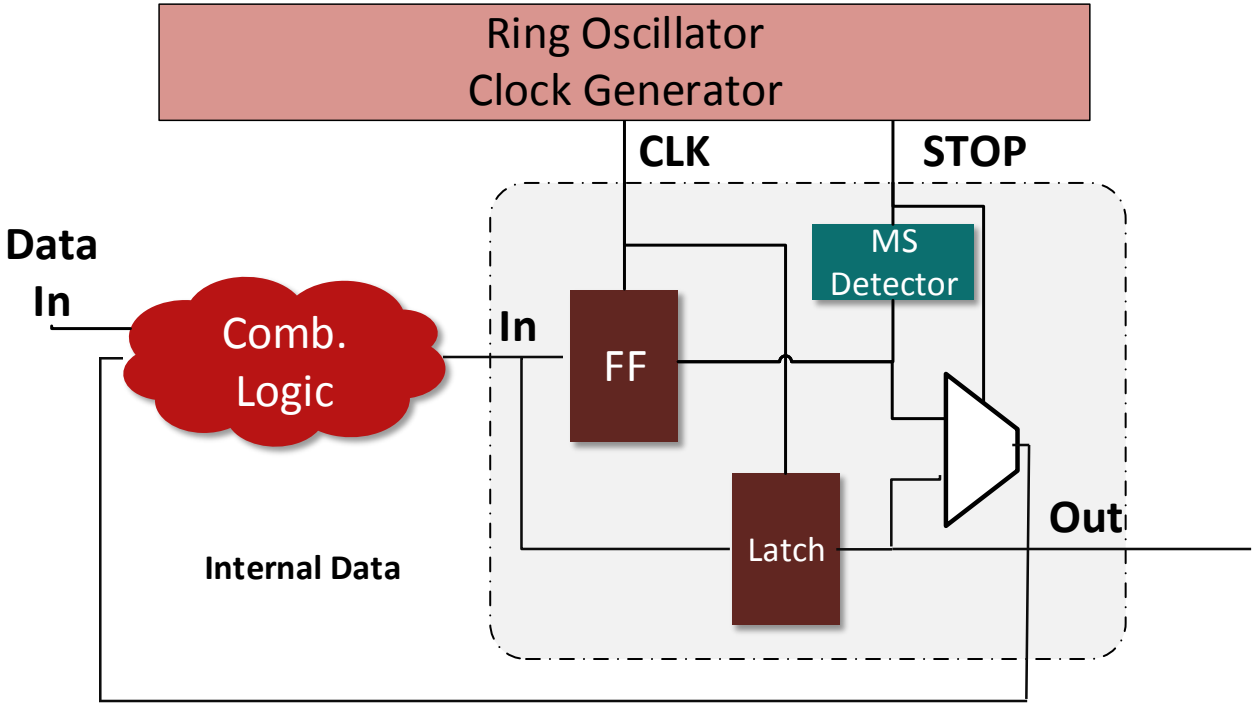


[RazorII, Das, 2009]

Transition detector may exhibit metastability

- Error signal must go through synchronizer, increasing delay
- Uses architectural replay to recover from errors

Hold Time Concerns

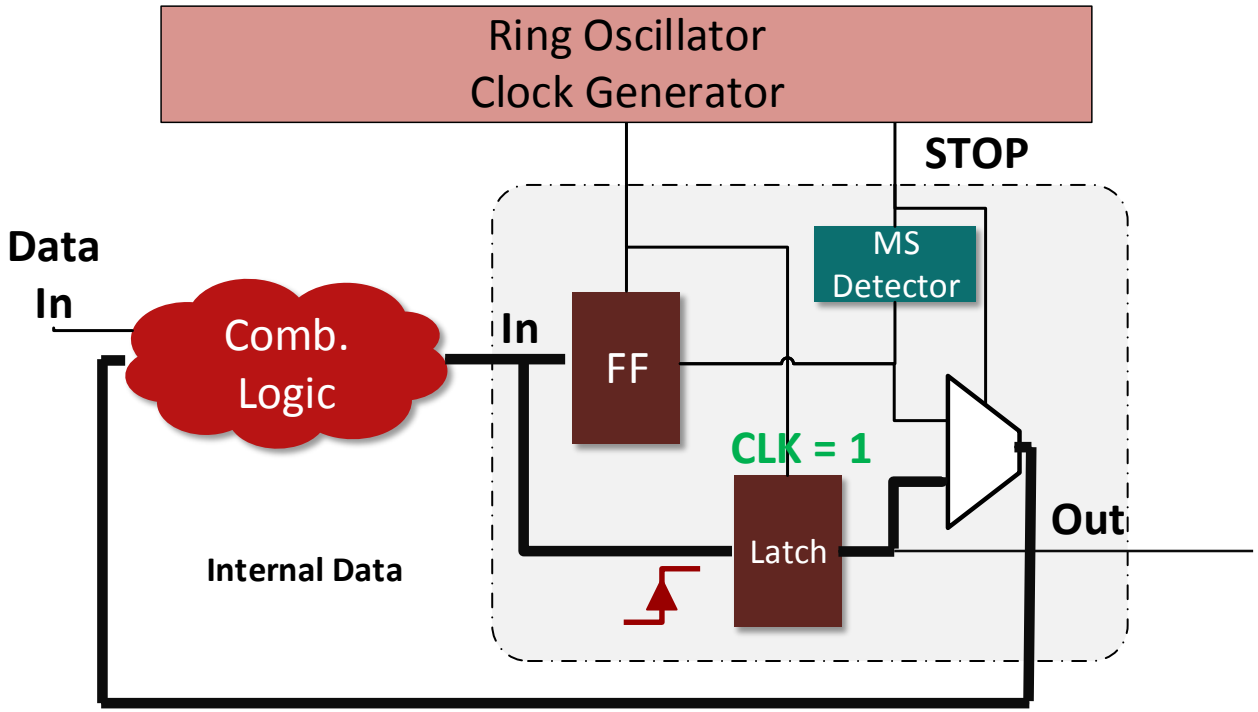


[SafeRazor, Cannizzaro, 2014]

Relies on latch for error correction

Hold times are problematic

Hold Time Concerns



[SafeRazor, Cannizzaro, 2014]

Relies on latch for error correction

Hold times are problematic

Resiliency Landscape

Design Template	Sync / Async	MTBF Safe	Avoids Replay Logic	Hold Time Robust	Low Error Penalty
Bubble Razor	Sync	No	Yes	Yes	Yes
Razor II	Sync	Yes	No	No	No
SafeRazor	Async	Yes*	Yes	No	Yes

Resiliency Landscape

Design Template	Sync / Async	MTBF Safe	Avoids Replay Logic	Hold Time Robust	Low Error Penalty
Bubble Razor	Sync	No	Yes	Yes	Yes
Razor II	Sync	Yes	No	No	No
SafeRazor	Async	Yes*	Yes	No	Yes
Blade	Async	Yes	Yes	Yes	Yes

Blade combines the best features of past resiliency schemes

Outline

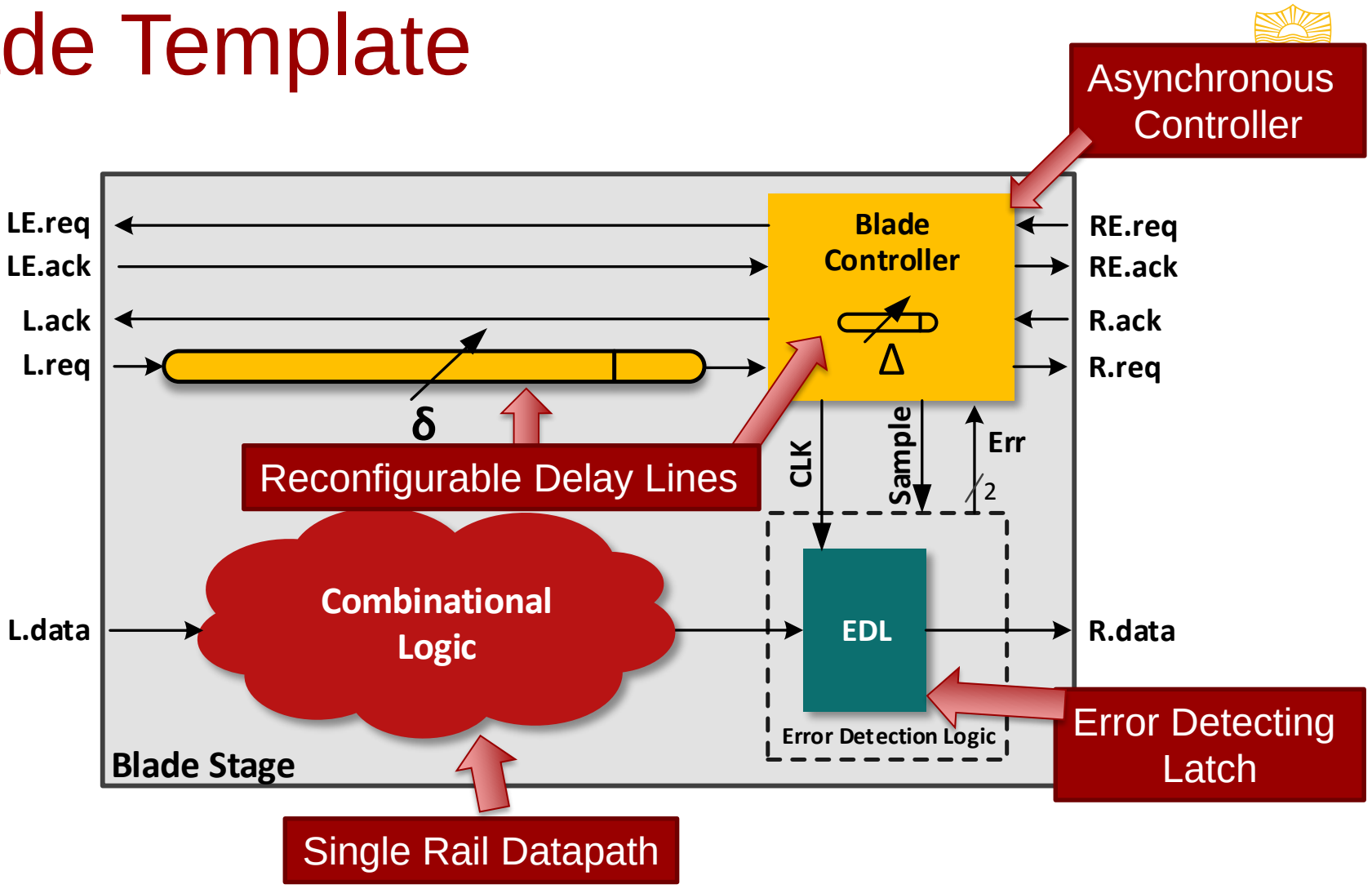
Our proposed resilient solution – Blade

Case study – 3-stage Plasma CPU

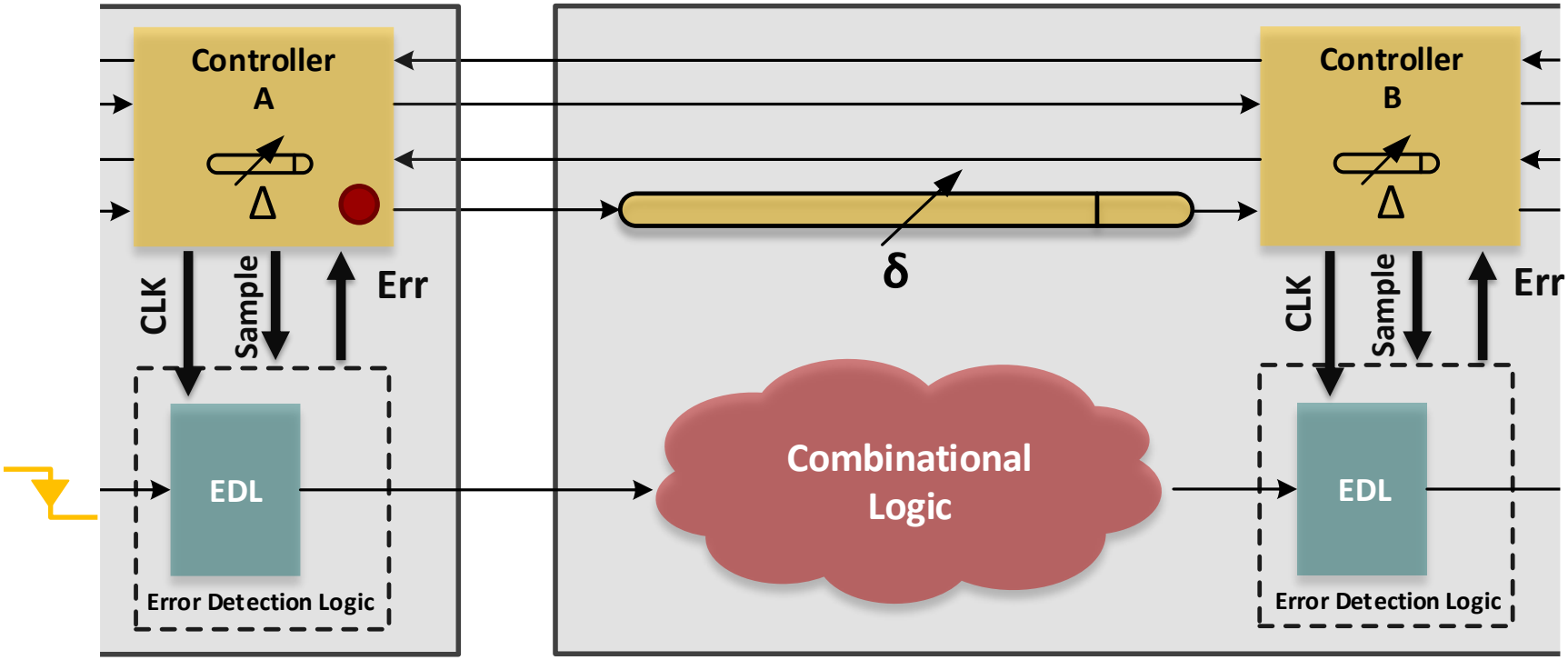
- Automated flow
- Area efficiency features
- Results and comparisons

Conclusions and future work

Blade Template



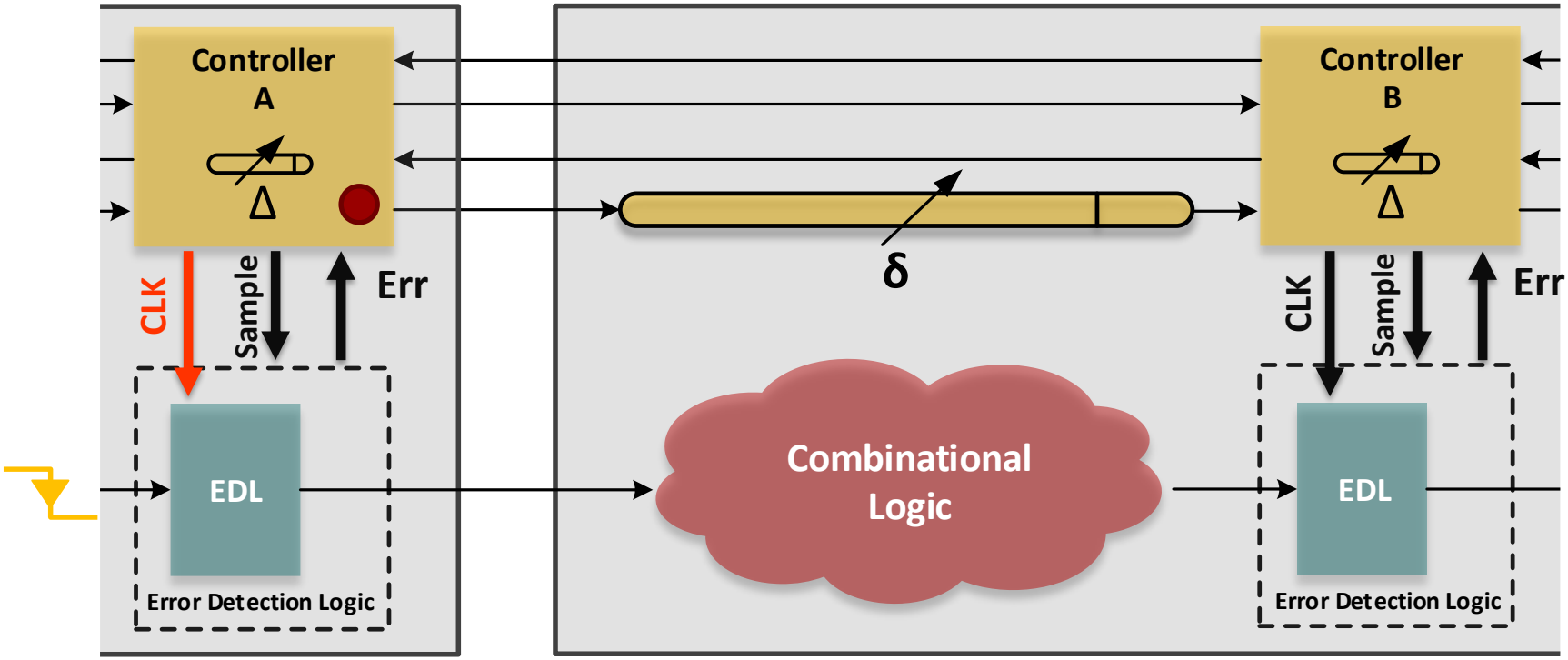
Blade Template Operation



- Send request speculatively before data is guaranteed stable

Timing errors delay handshaking signals

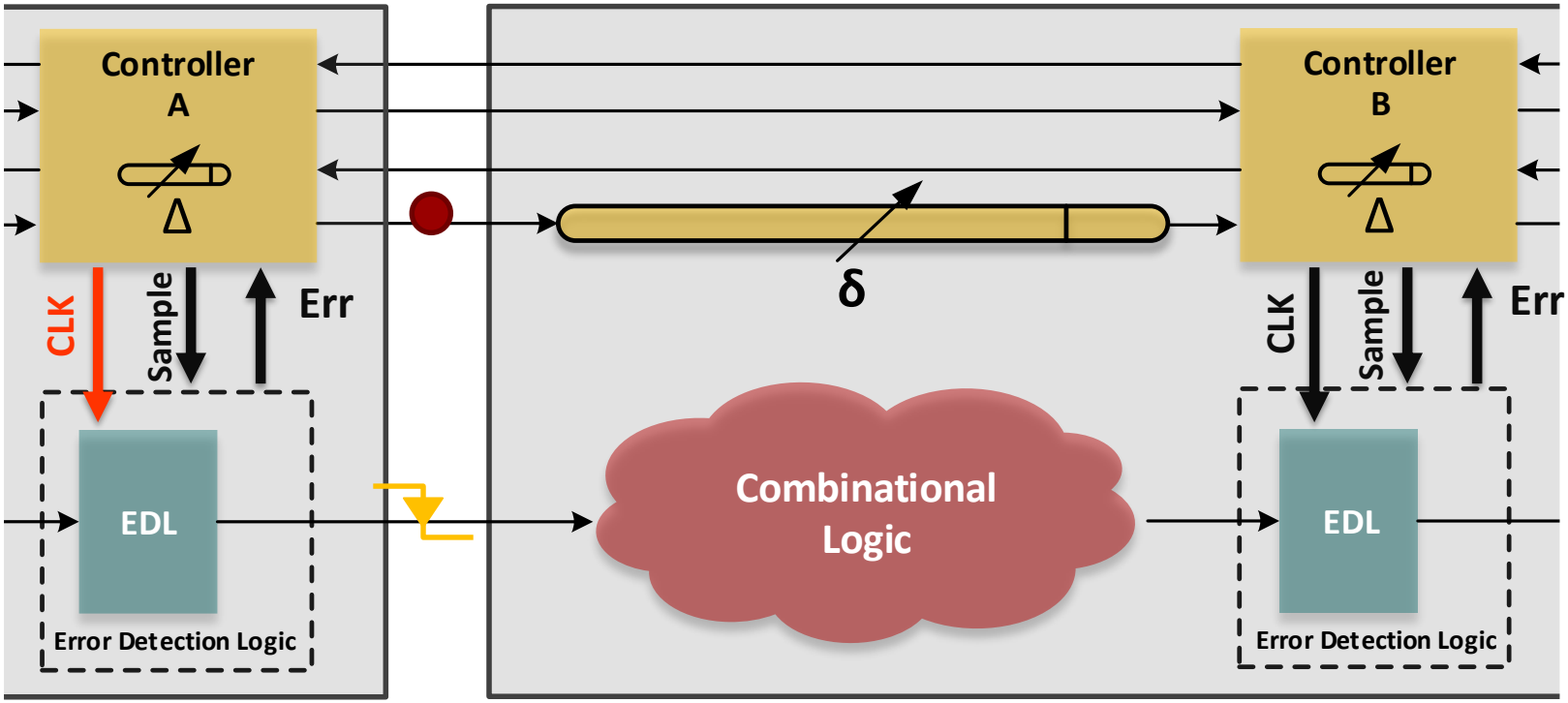
Blade Template Operation



- Send request speculatively before data is guaranteed stable

Timing errors delay handshaking signals

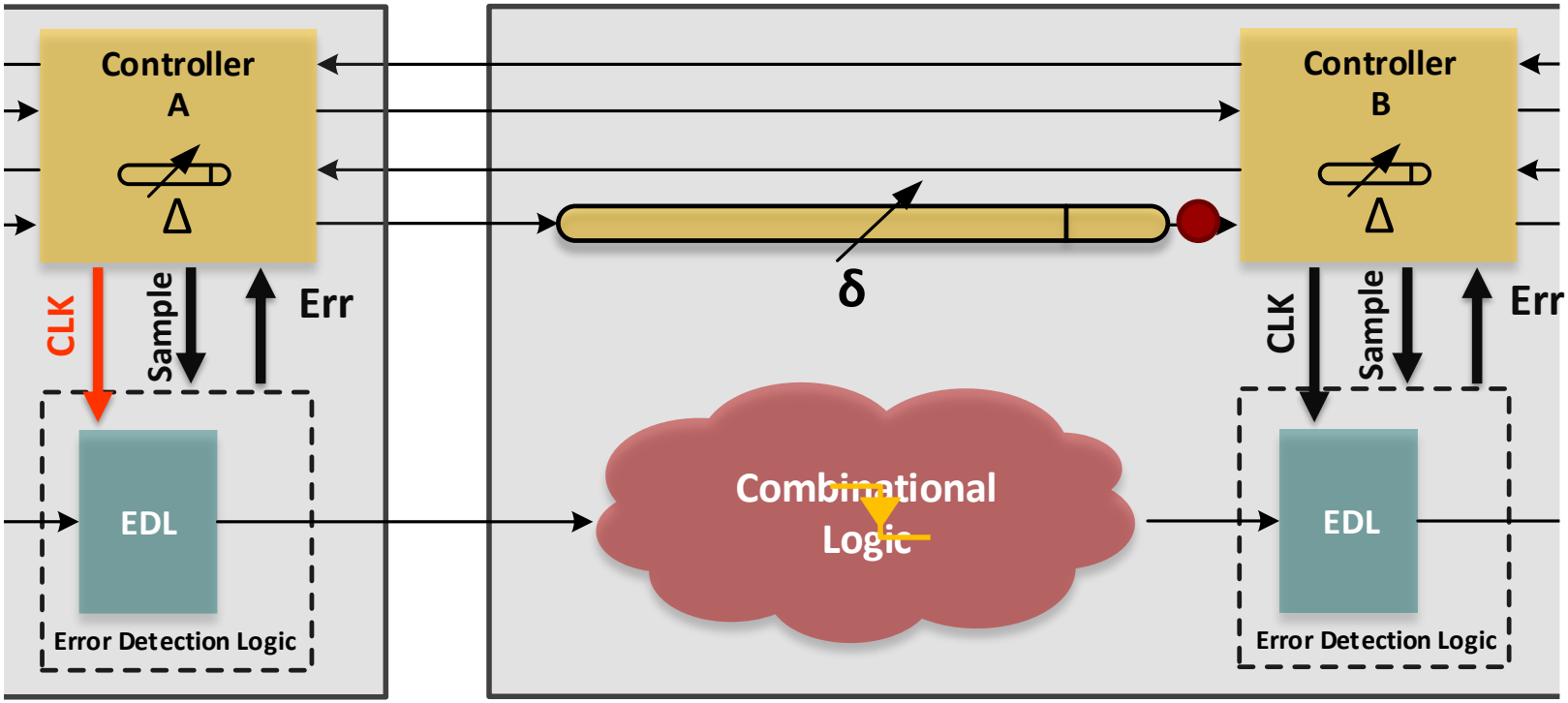
Blade Template Operation



➡ Send request speculatively before data is guaranteed stable

Timing errors delay handshaking signals

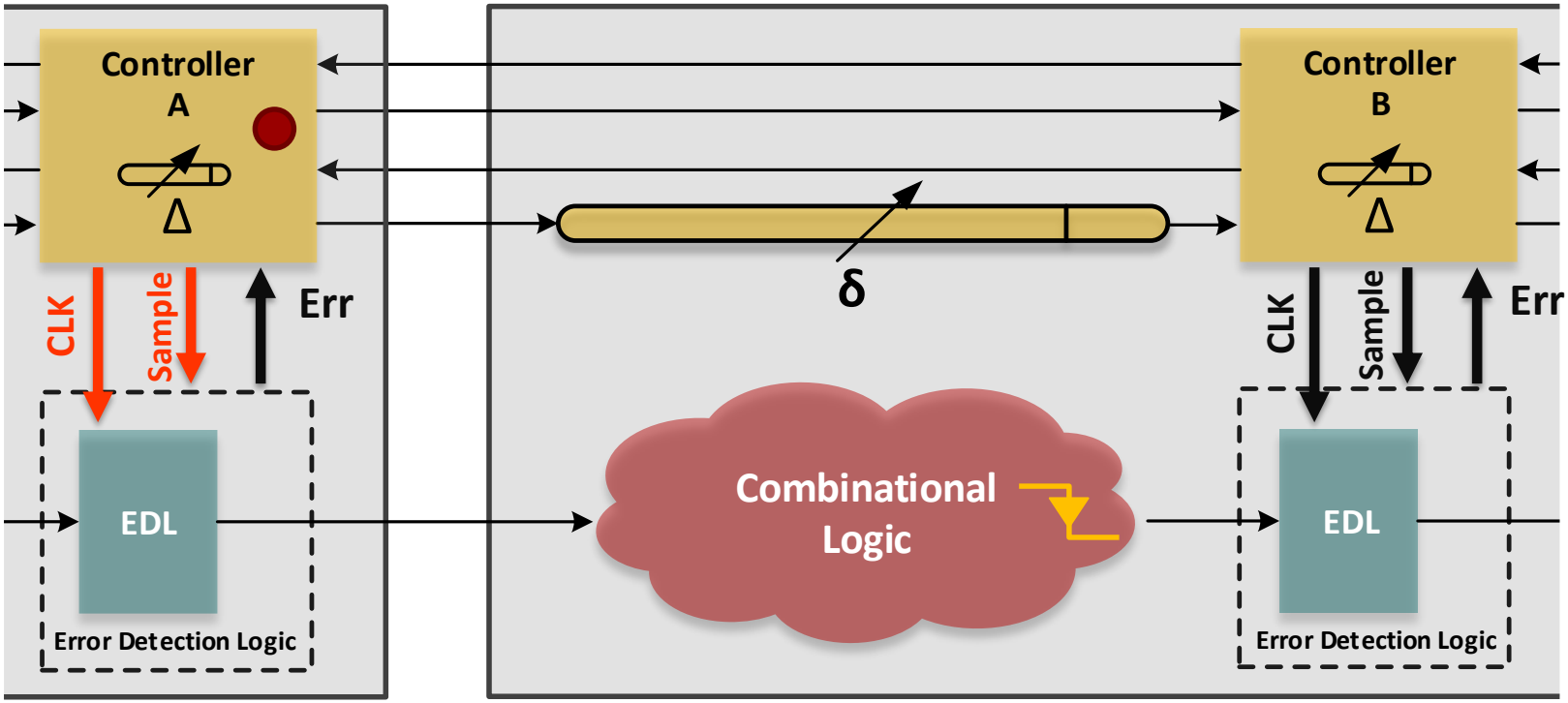
Blade Template Operation



➡ Send request speculatively before data is guaranteed stable

Timing errors delay handshaking signals

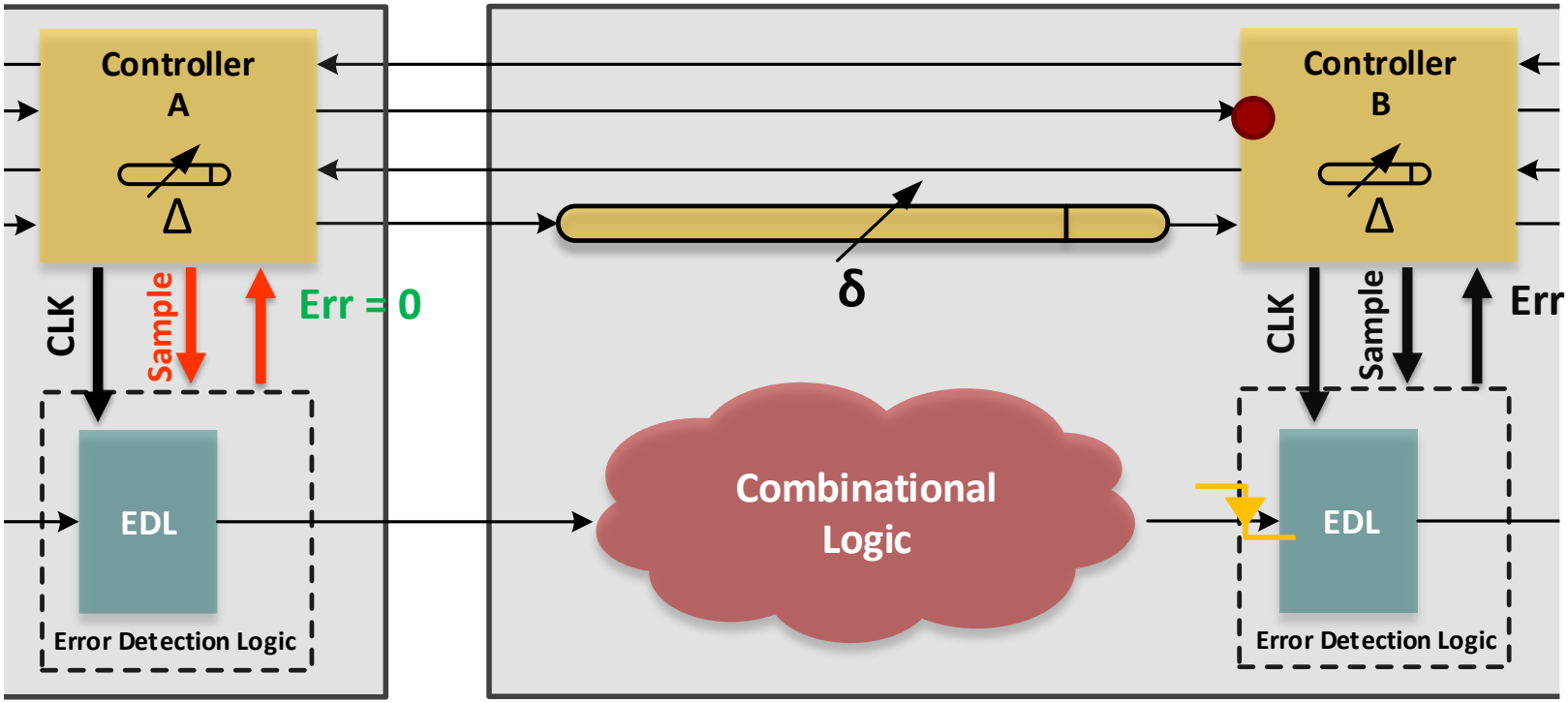
Blade Template Operation



➡ Send request speculatively before data is guaranteed stable

Timing errors delay handshaking signals

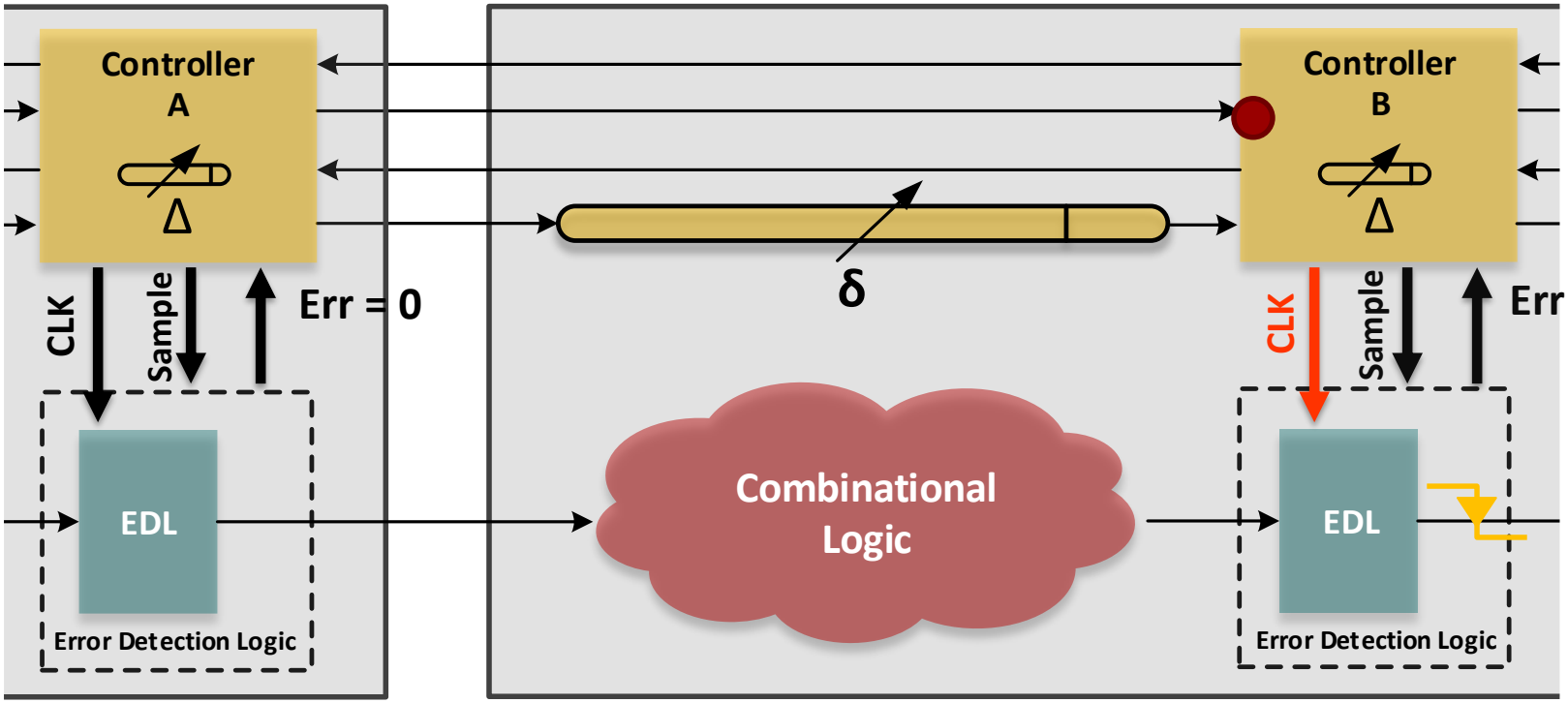
Blade Template Operation



➡ Send request speculatively before data is guaranteed stable

Timing errors delay handshaking signals

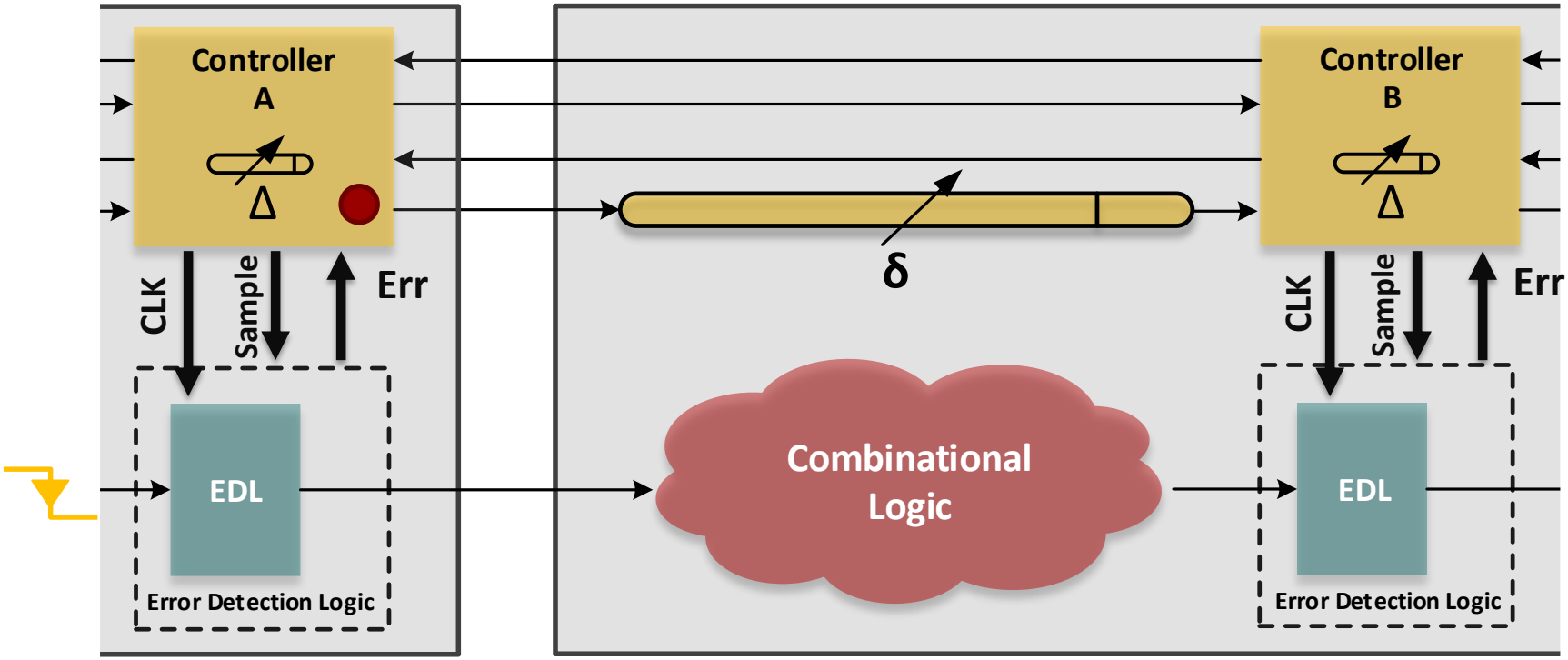
Blade Template Operation



➡ Send request speculatively before data is guaranteed stable

Timing errors delay handshaking signals

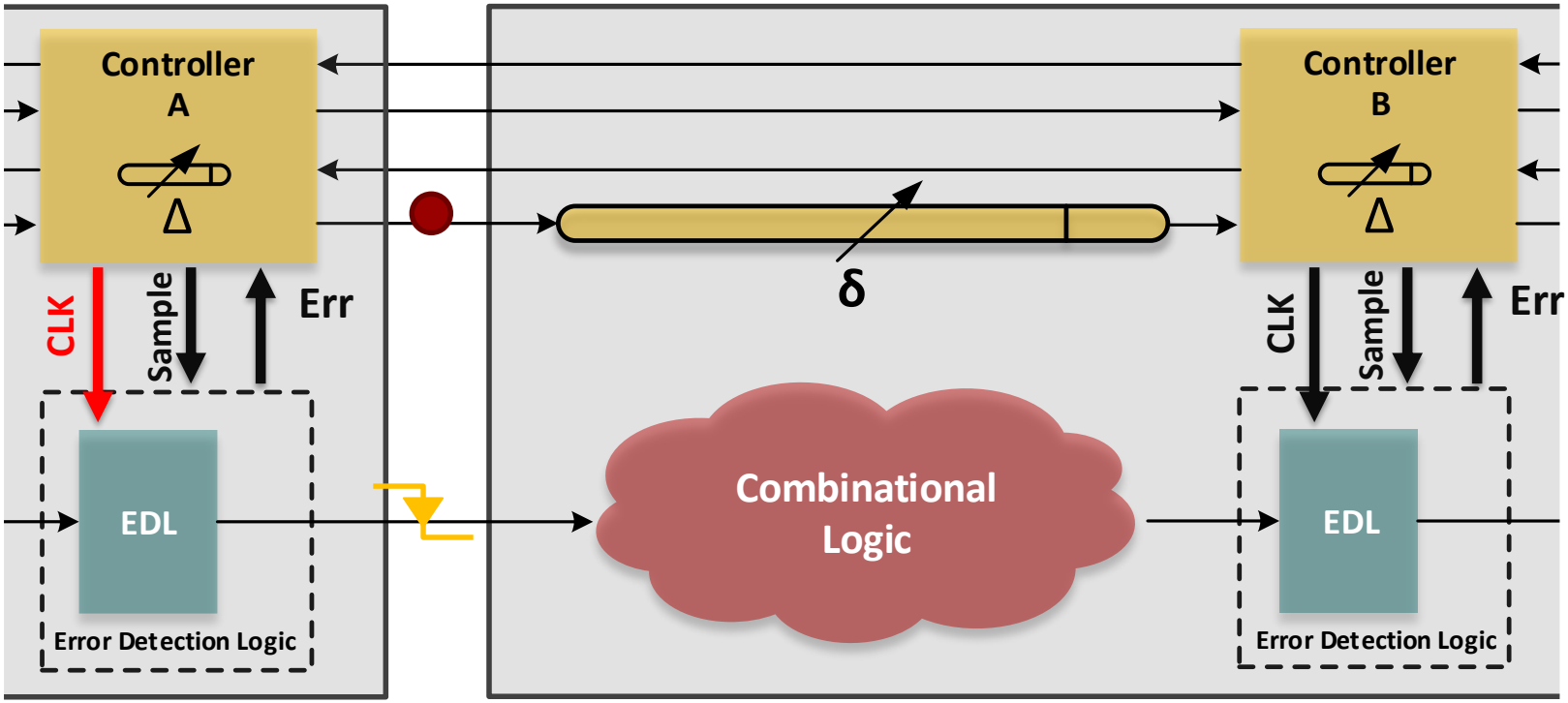
Blade Template Operation



Send request speculatively before data is guaranteed stable

➡ Timing errors delay handshaking signals

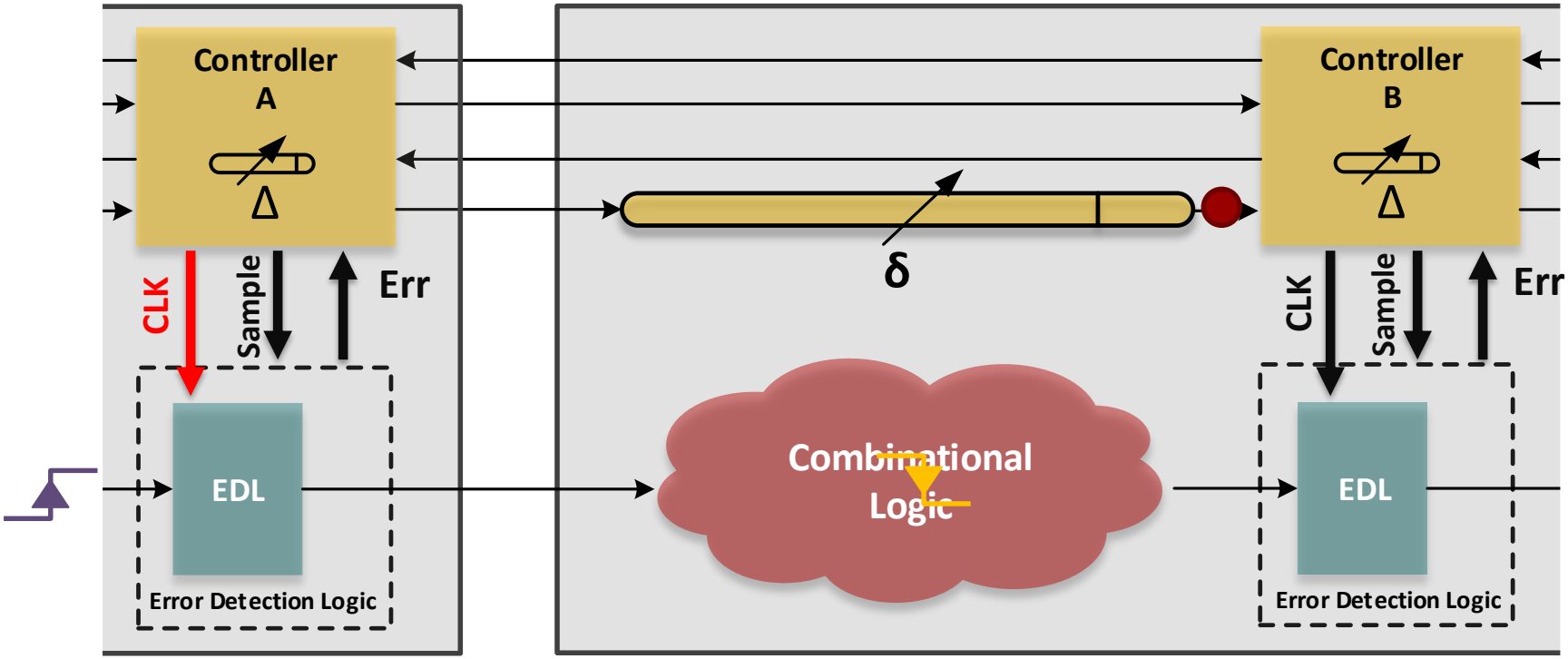
Blade Template Operation



Send request speculatively before data is guaranteed stable

➡ Timing errors delay handshaking signals

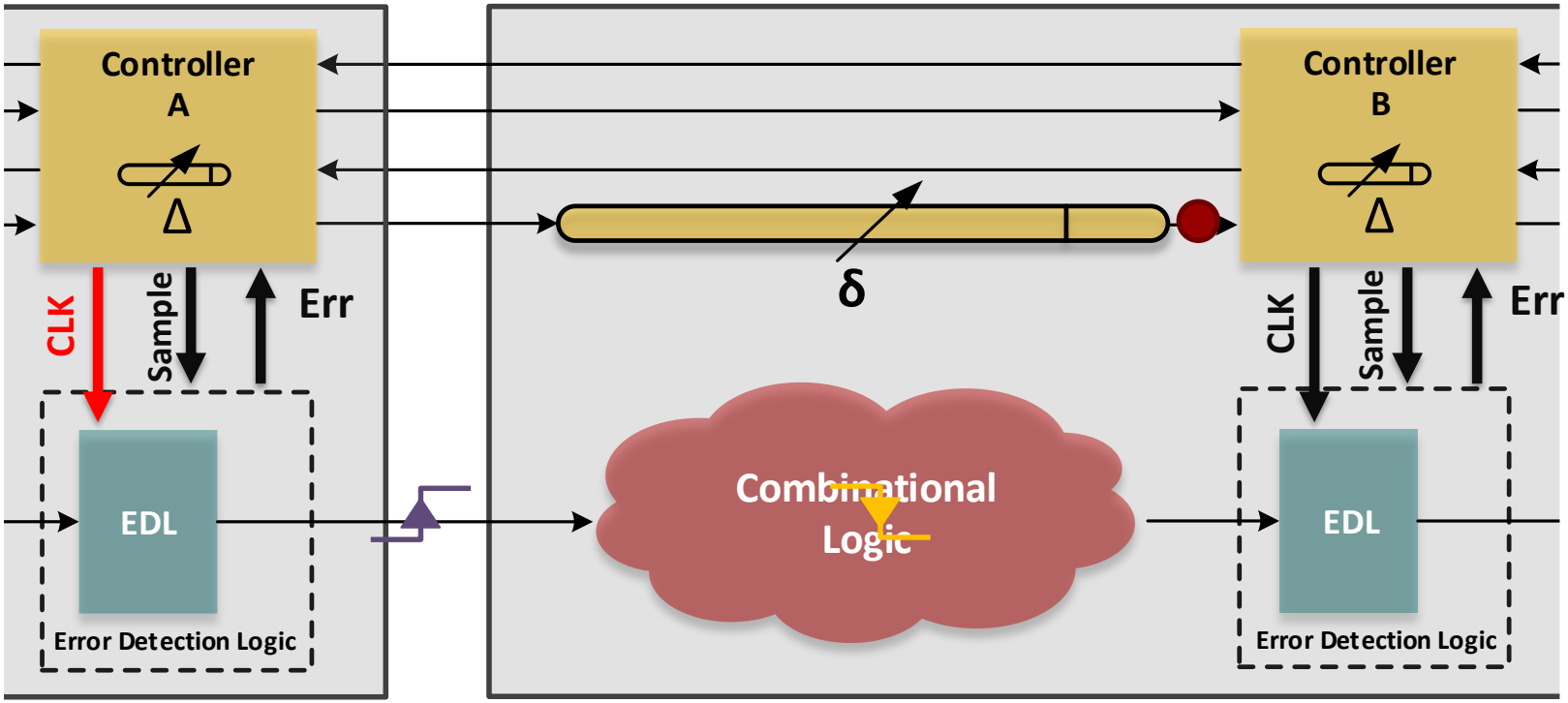
Blade Template Operation



Send request speculatively before data is guaranteed stable

➡ Timing errors delay handshaking signals

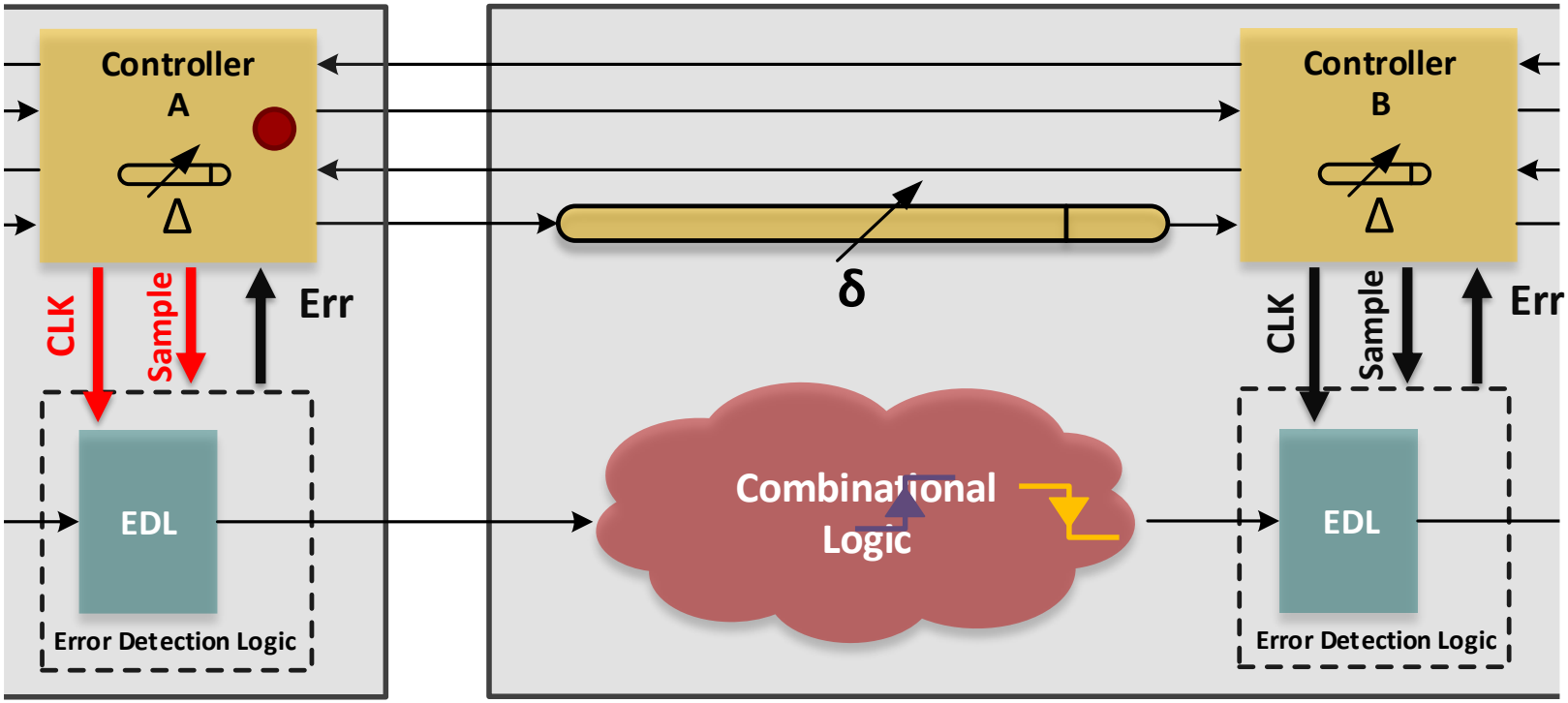
Blade Template Operation



Send request speculatively before data is guaranteed stable

➡ Timing errors delay handshaking signals

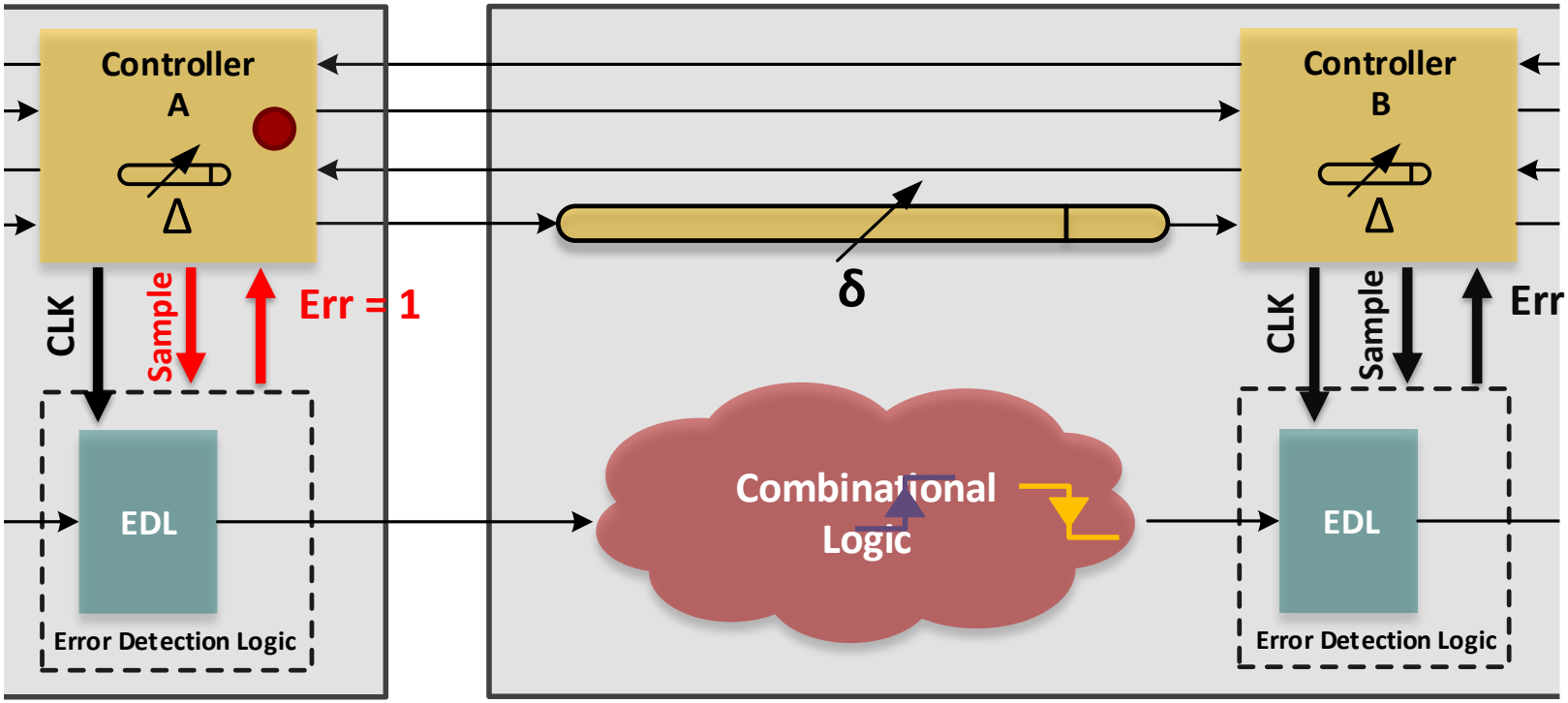
Blade Template Operation



Send request speculatively before data is guaranteed stable

➡ Timing errors delay handshaking signals

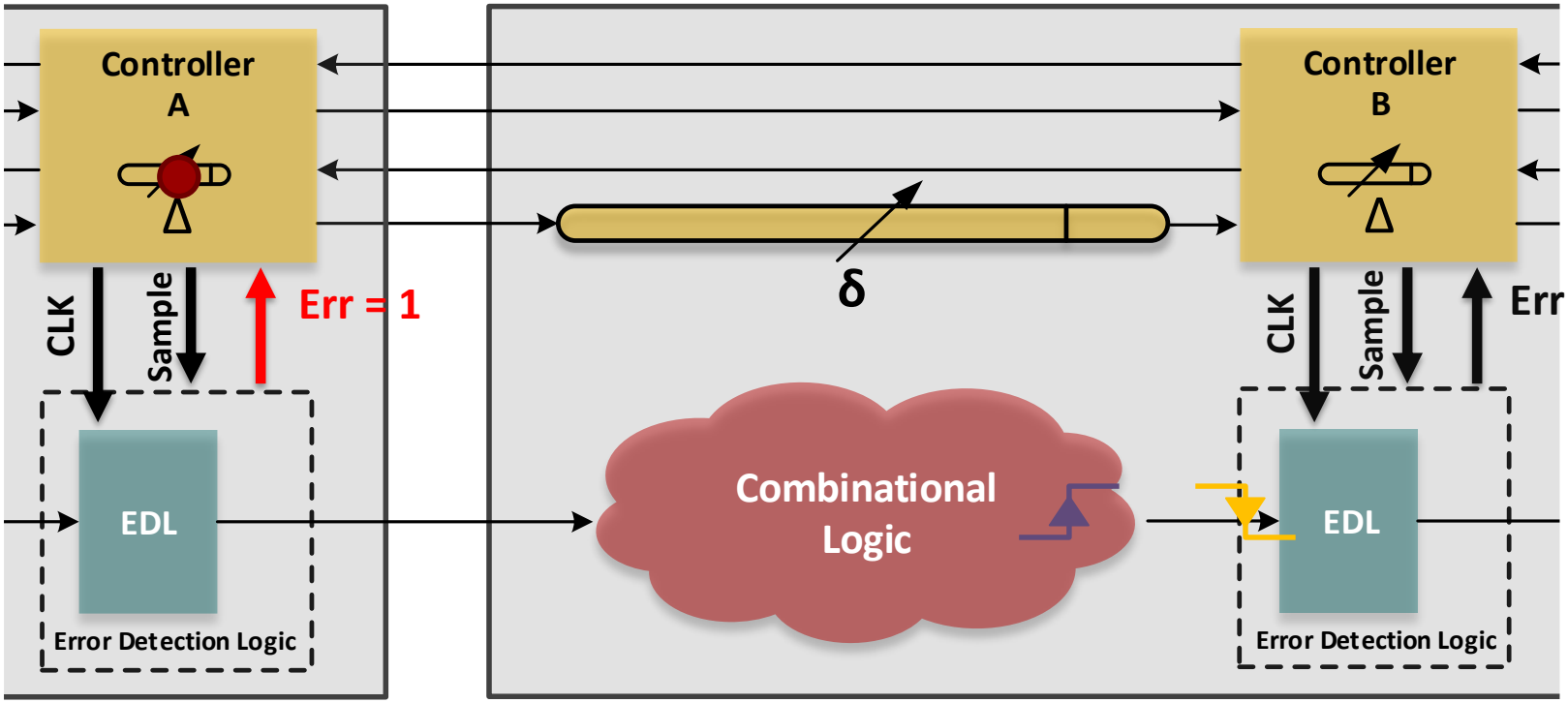
Blade Template Operation



Send request speculatively before data is guaranteed stable

➡ Timing errors delay handshaking signals

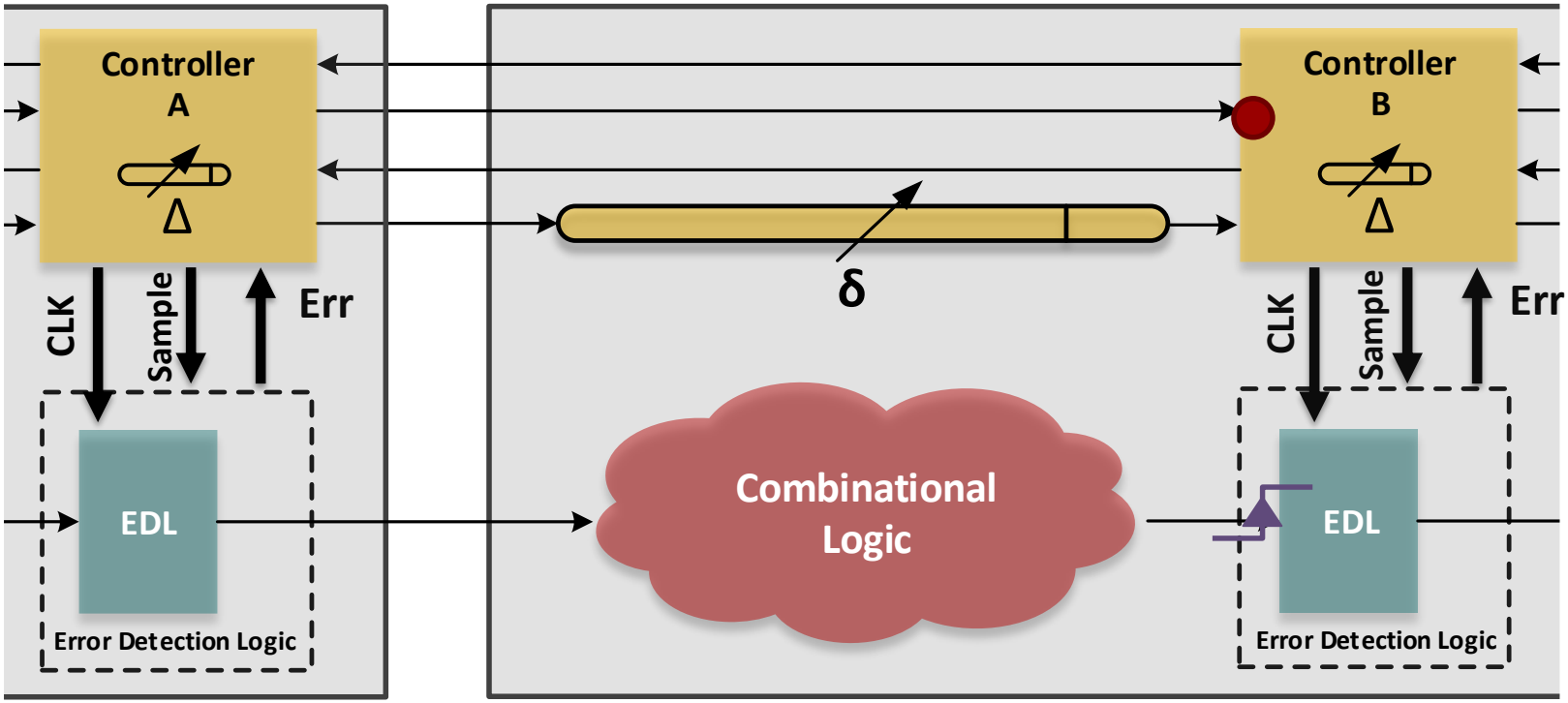
Blade Template Operation



Send request speculatively before data is guaranteed stable

➡ Timing errors delay handshaking signals

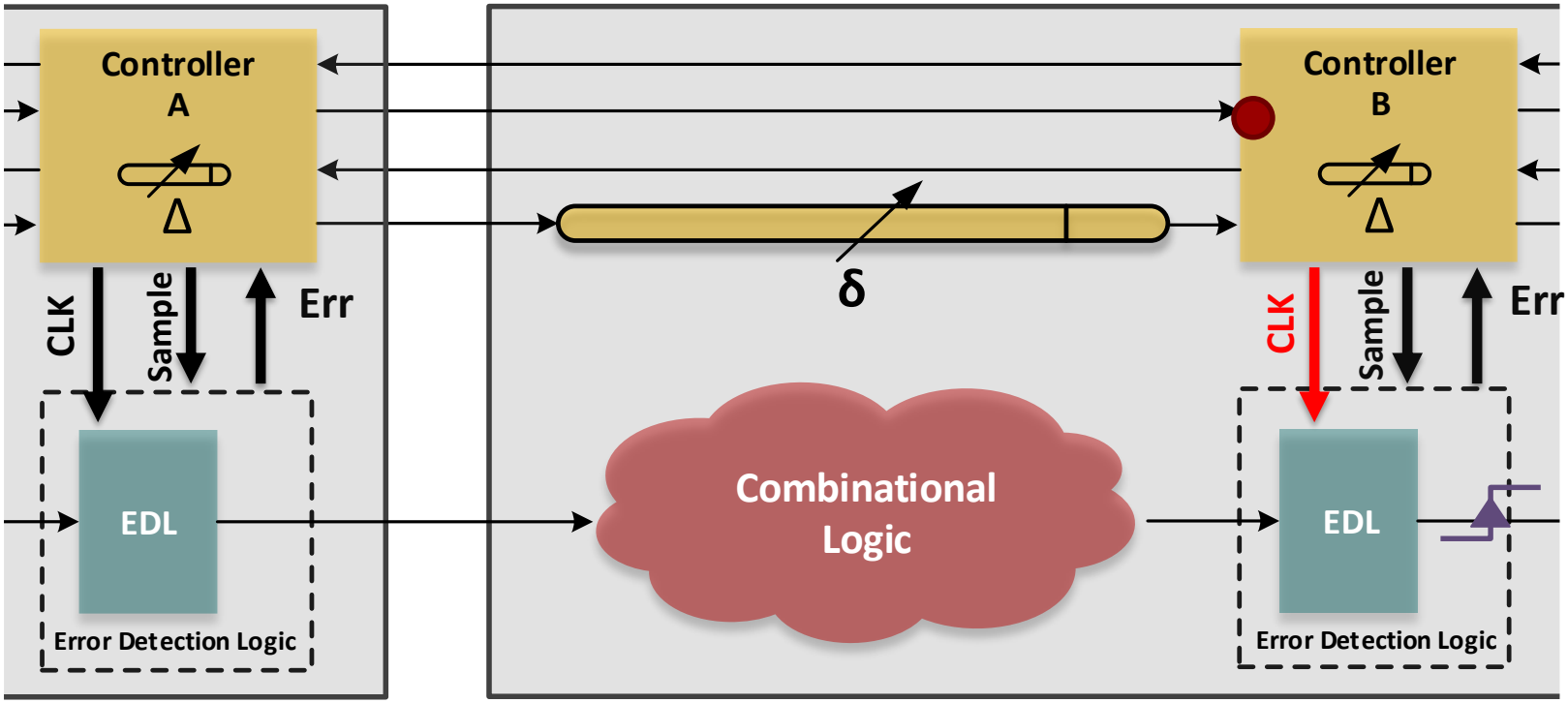
Blade Template Operation



Send request speculatively before data is guaranteed stable

➡ Timing errors delay handshaking signals

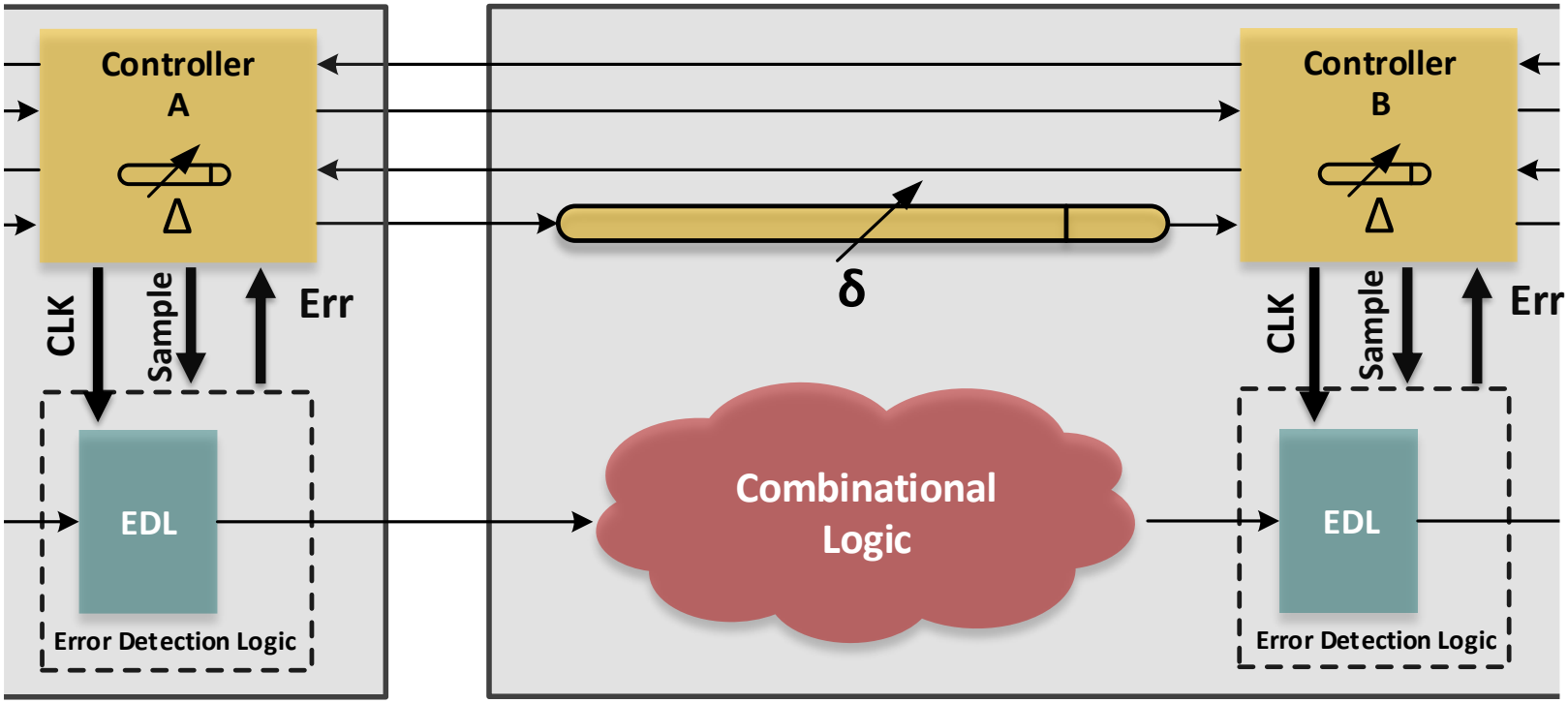
Blade Template Operation



Send request speculatively before data is guaranteed stable

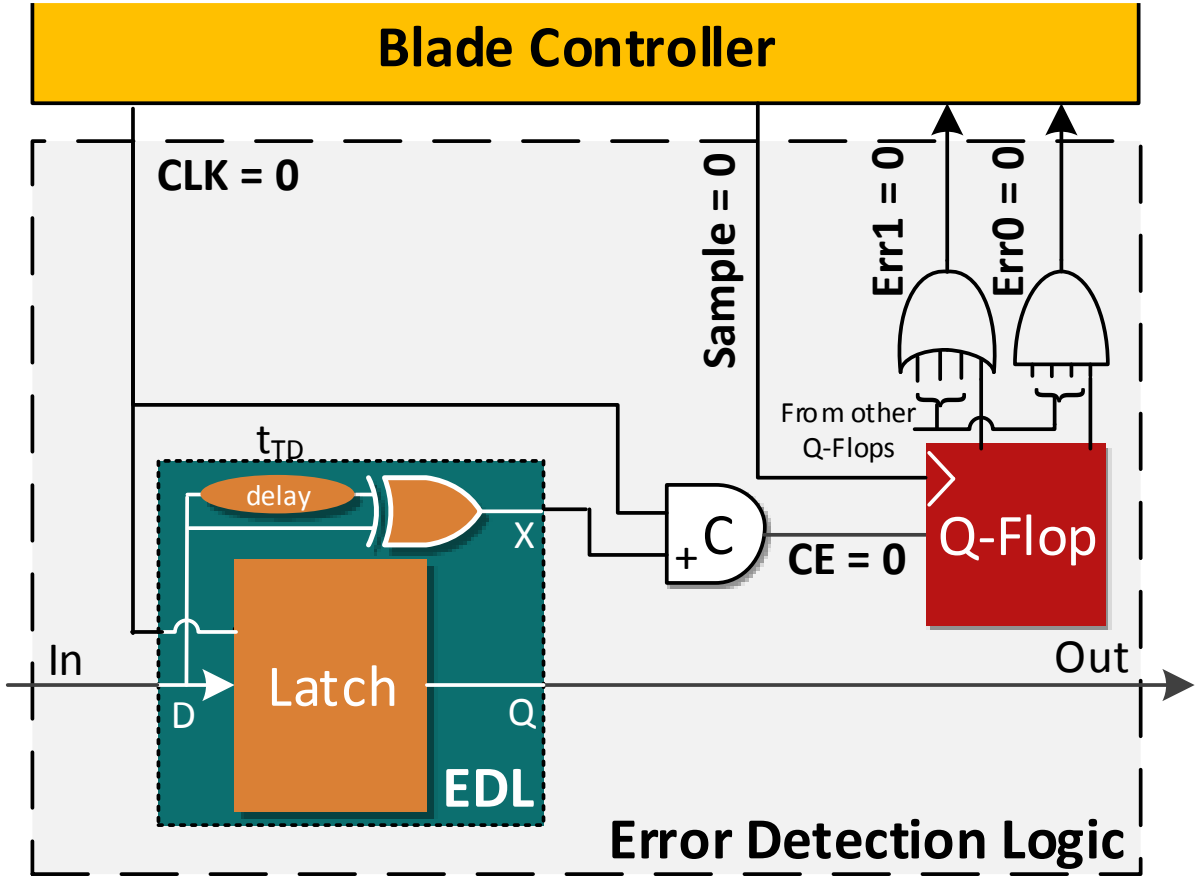
➡ Timing errors delay handshaking signals

Positive Hold Margins



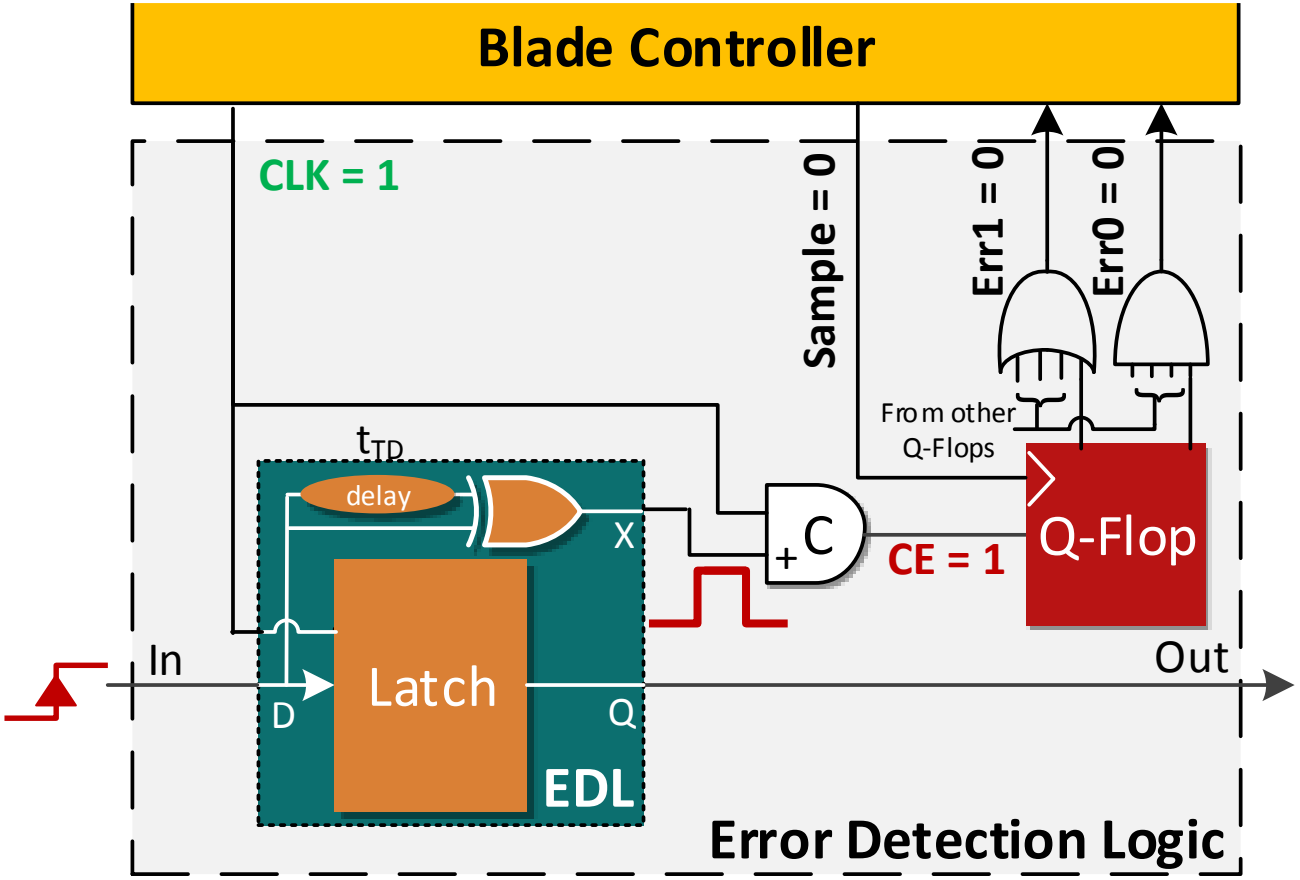
Handshaking delays create positive hold margin!

Error Detection Logic



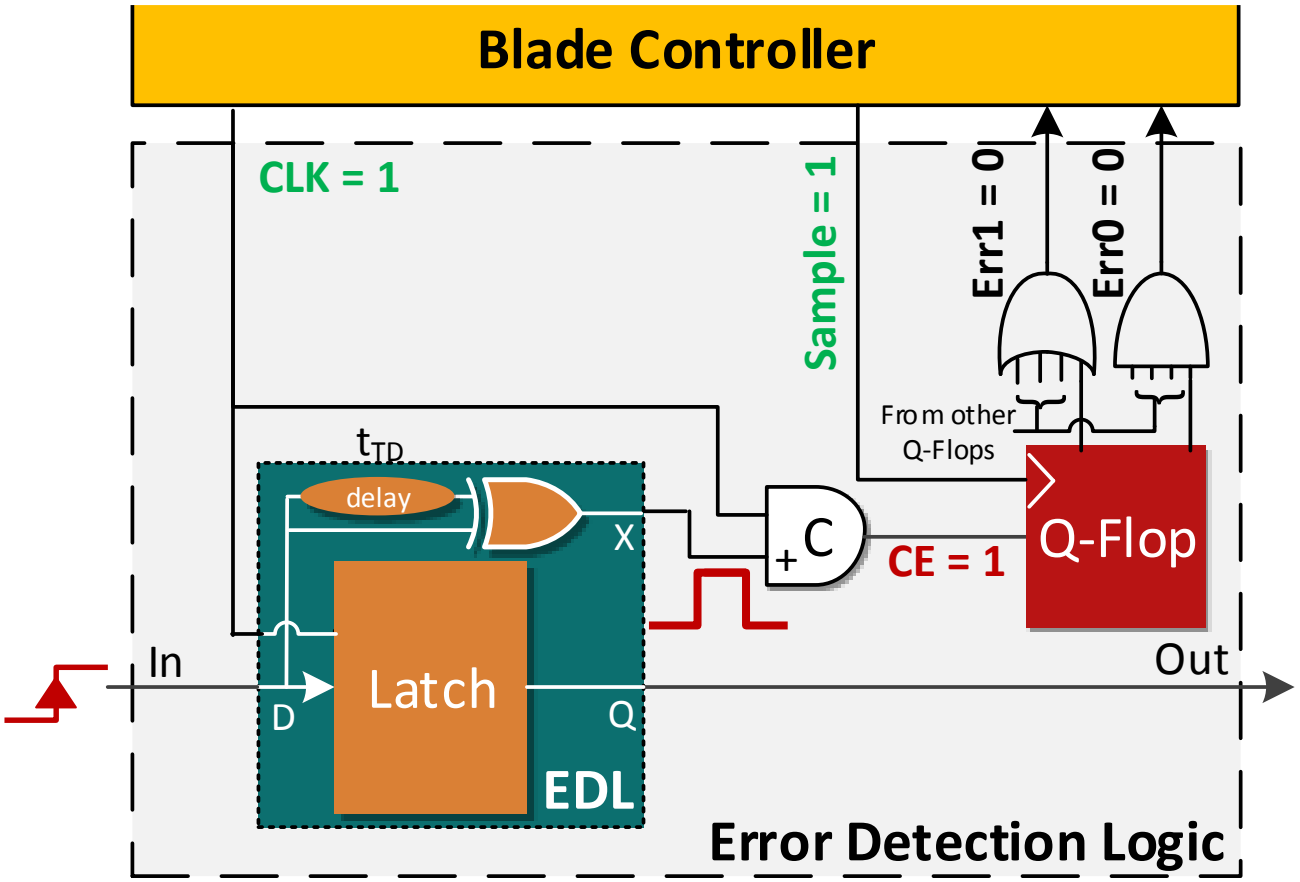
C-element stores error signal, which is sampled by Q-Flop

Error Detection Logic



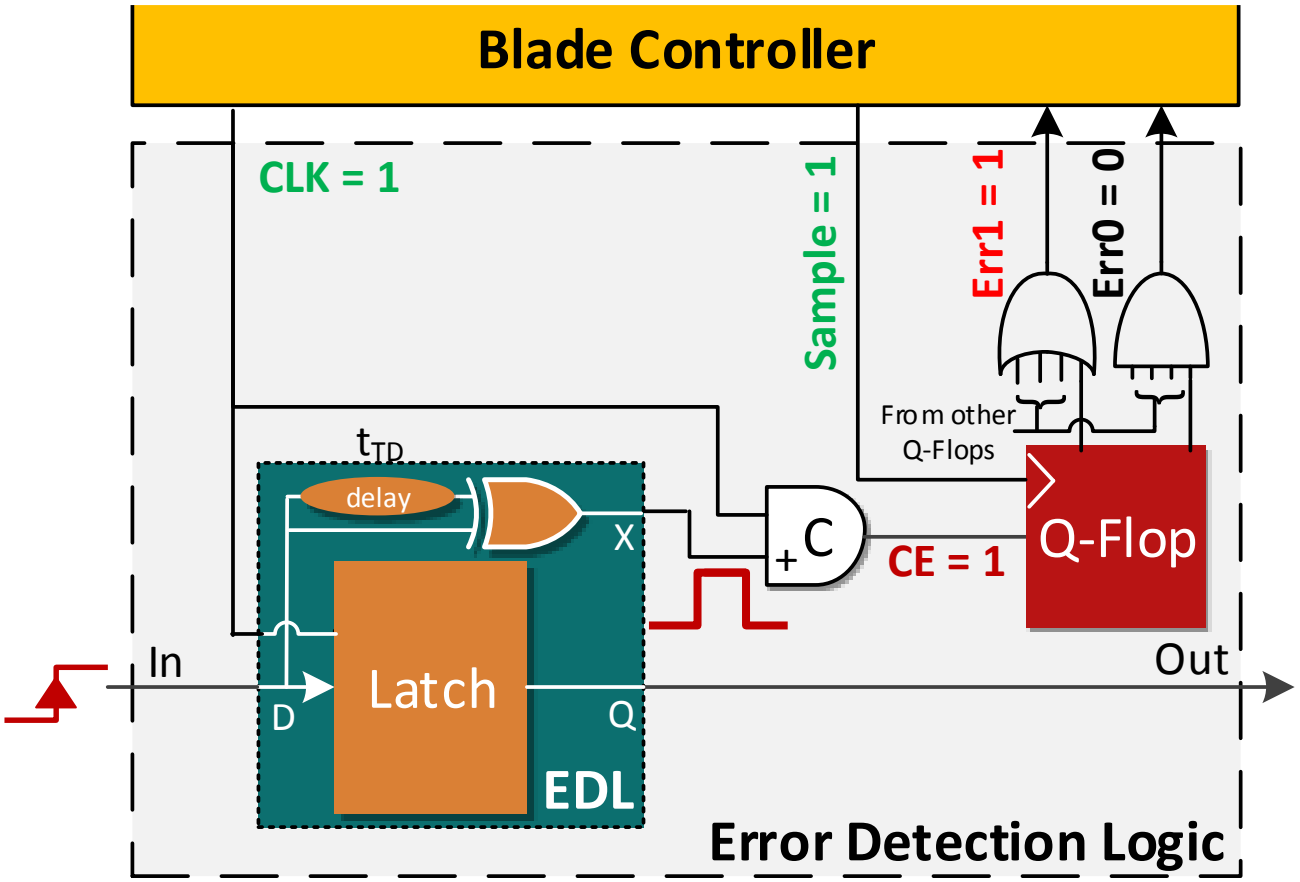
C-element stores error signal, which is sampled by Q-Flop

Error Detection Logic



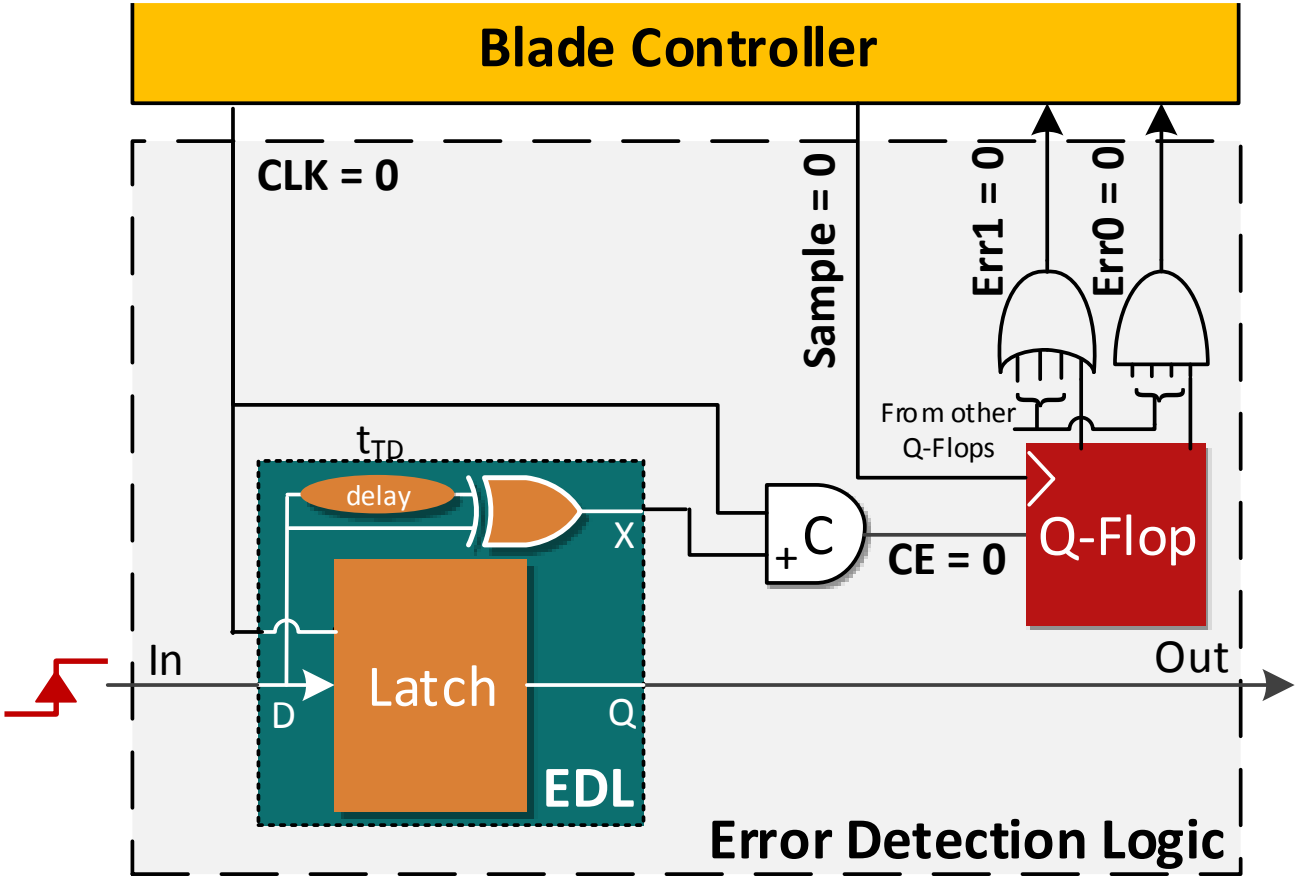
C-element stores error signal, which is sampled by Q-Flop

Error Detection Logic



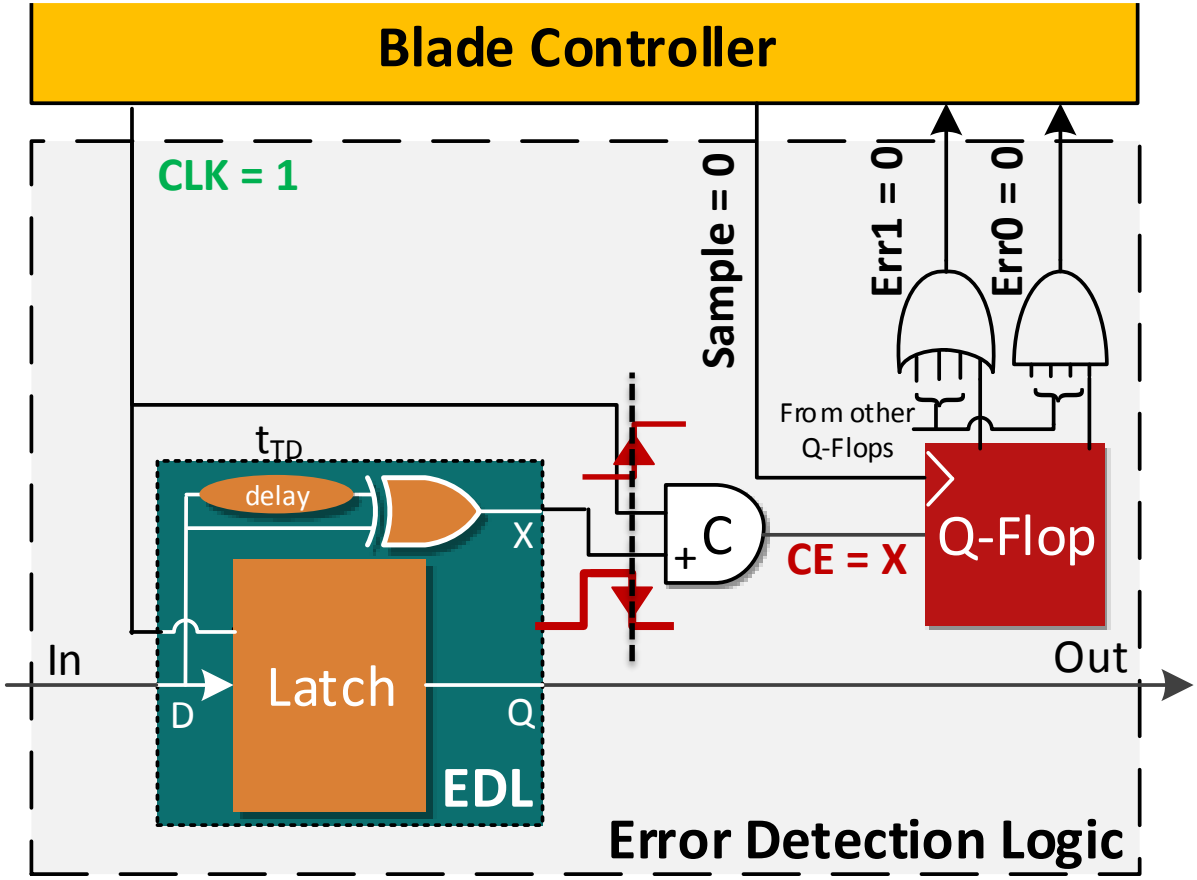
C-element stores error signal, which is sampled by Q-Flop

Metastable-Safe Operation



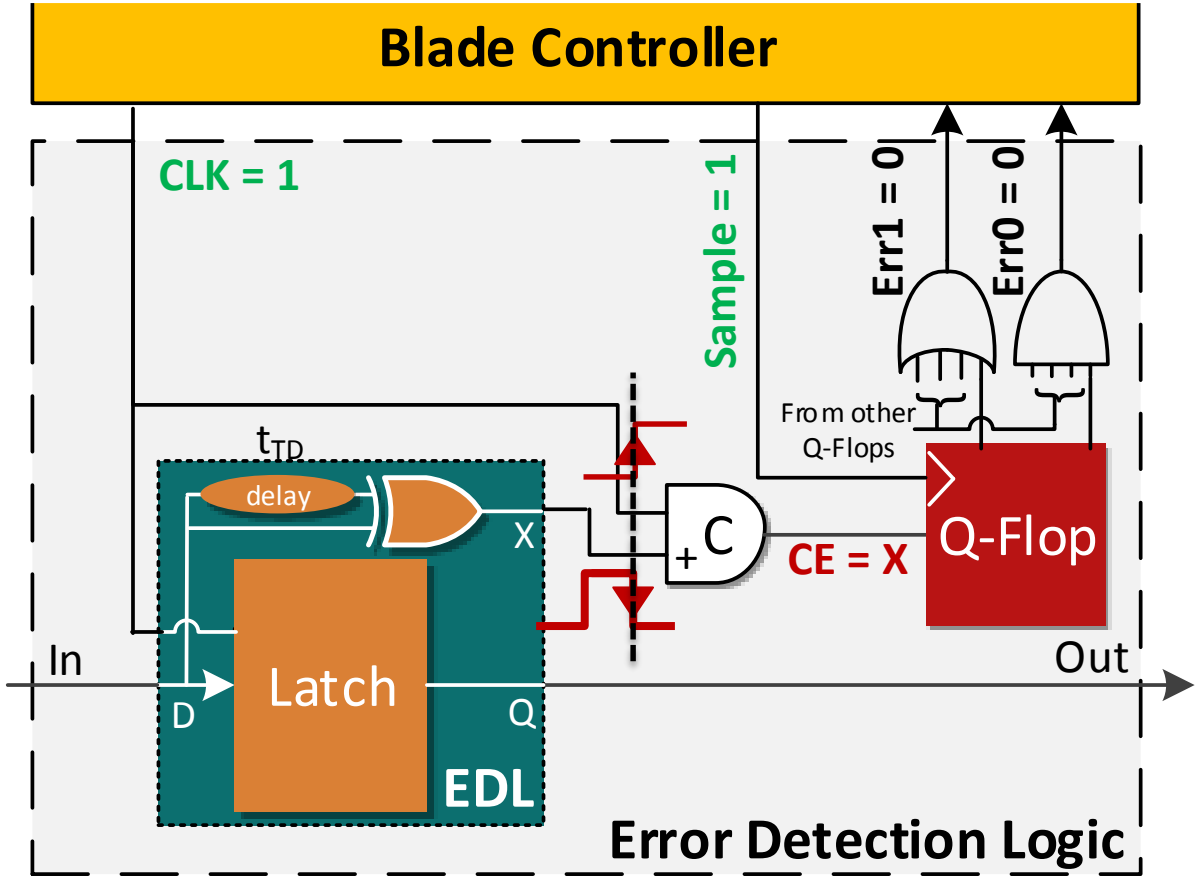
Q-Flop prevents metastability propagation to control path

Metastable-Safe Operation



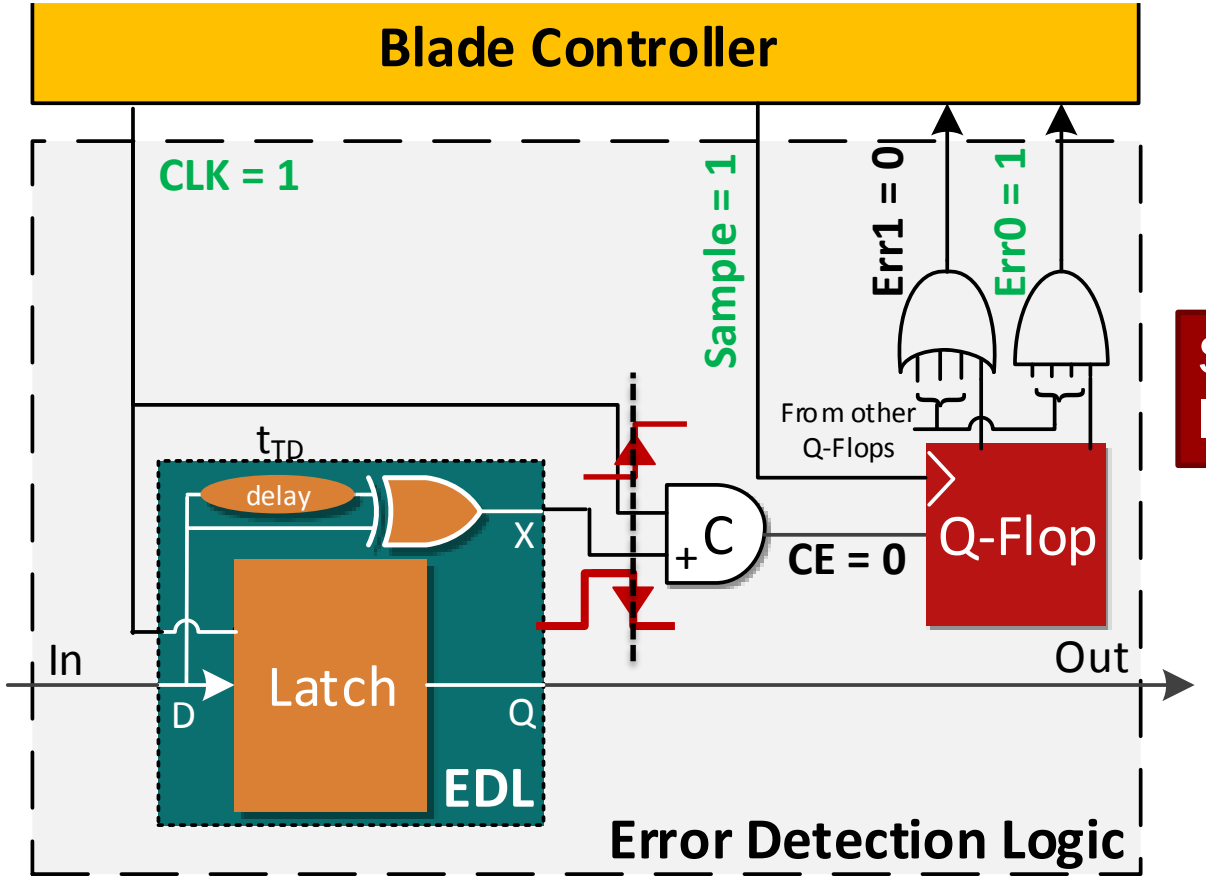
Q-Flop prevents metastability propagation to control path

Metastable-Safe Operation



Q-Flop prevents metastability propagation to control path

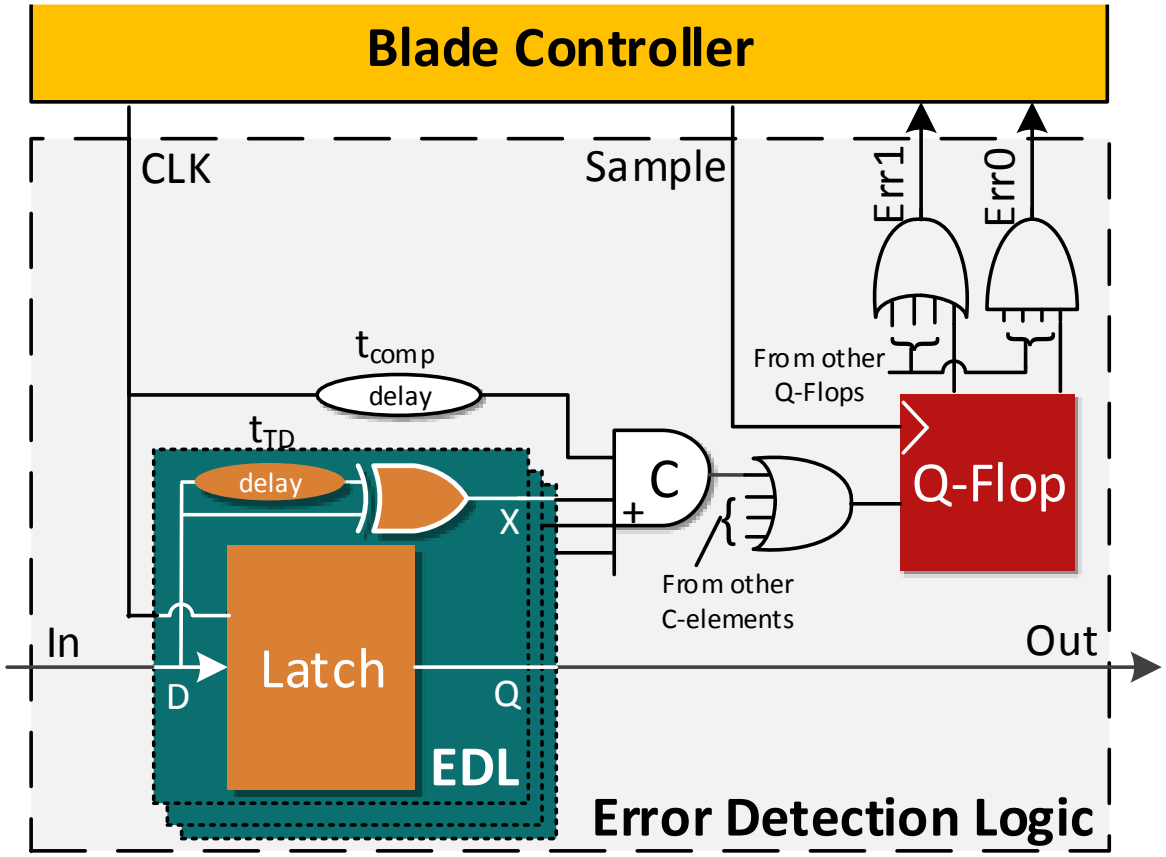
Metastable-Safe Operation



Some time later…

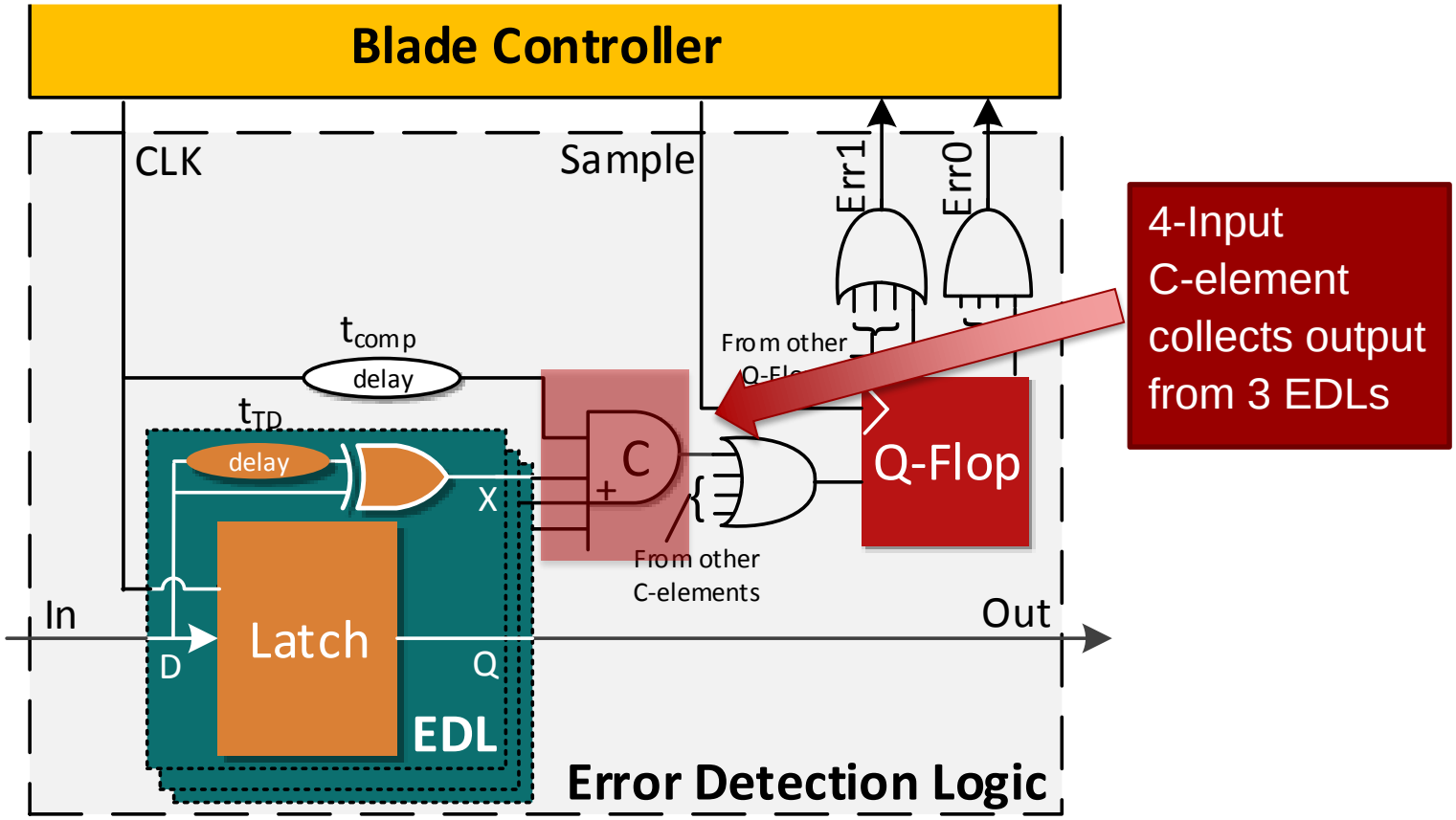
Q-Flop prevents metastability propagation to control path

Overhead Reduction



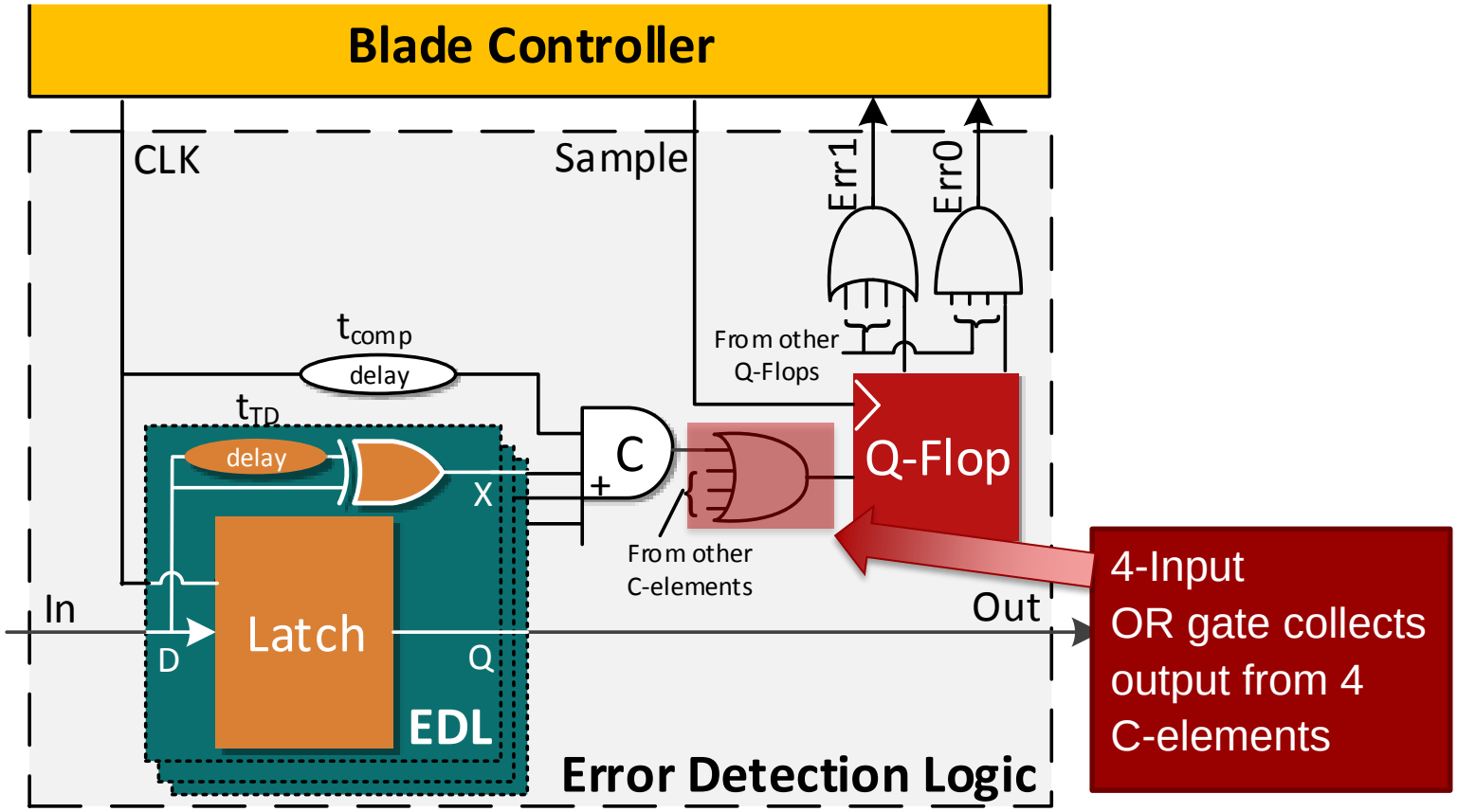
C-element and OR gates amortize overhead over many EDLs

Overhead Reduction



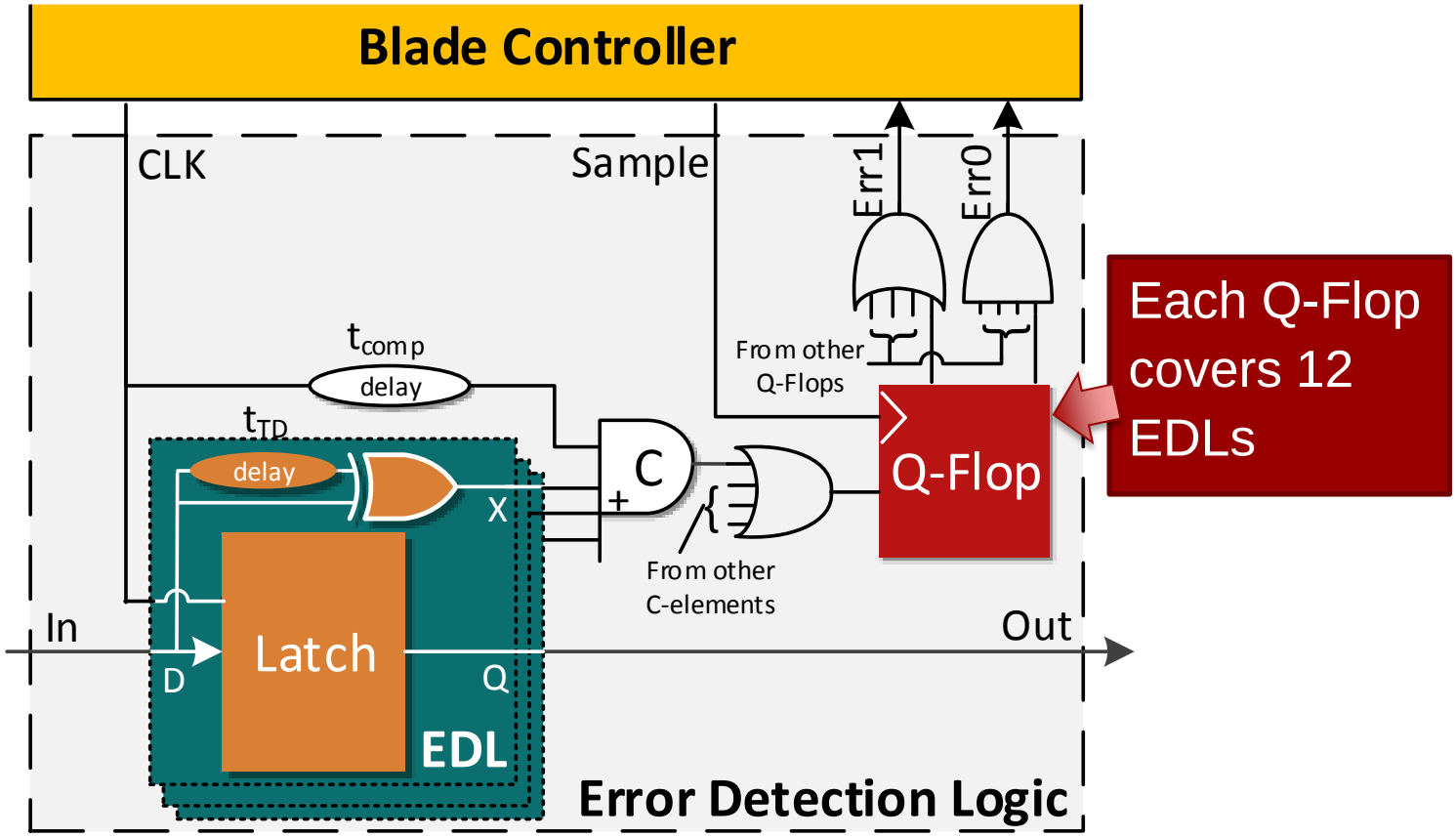
C-element and OR gates amortize overhead over many EDLs

Overhead Reduction



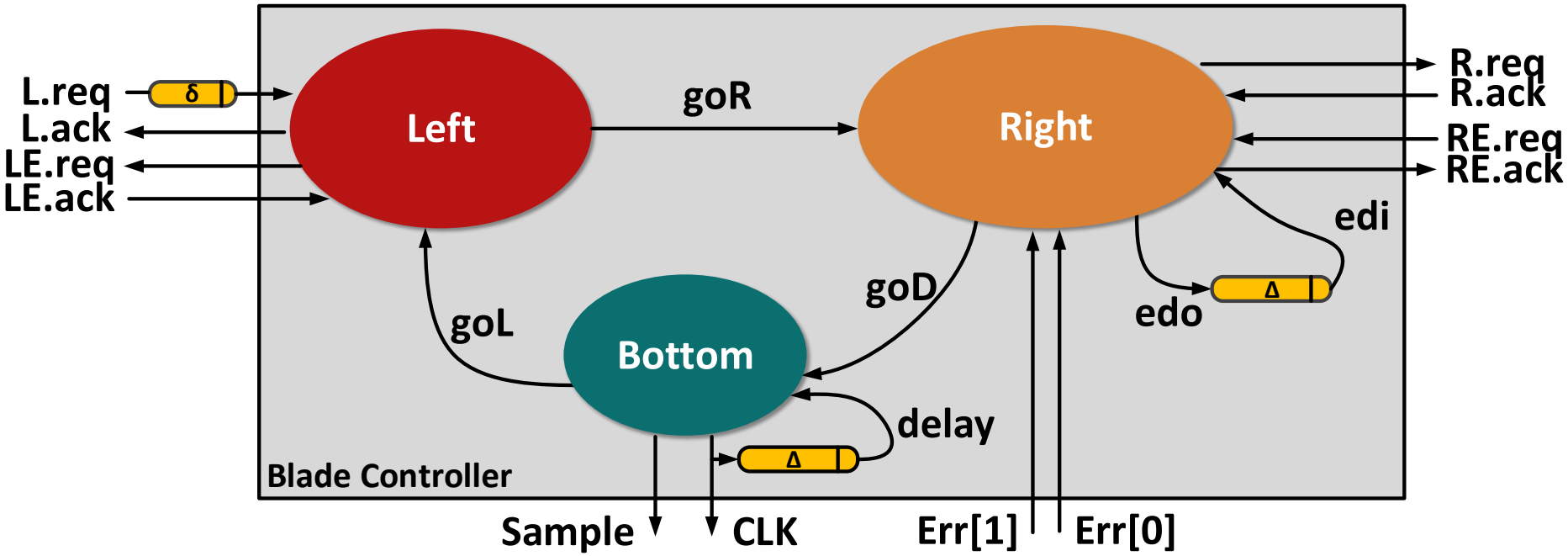
C-element and OR gates amortize overhead over many EDLs

Overhead Reduction



C-element and OR gates amortize overhead over many EDLs

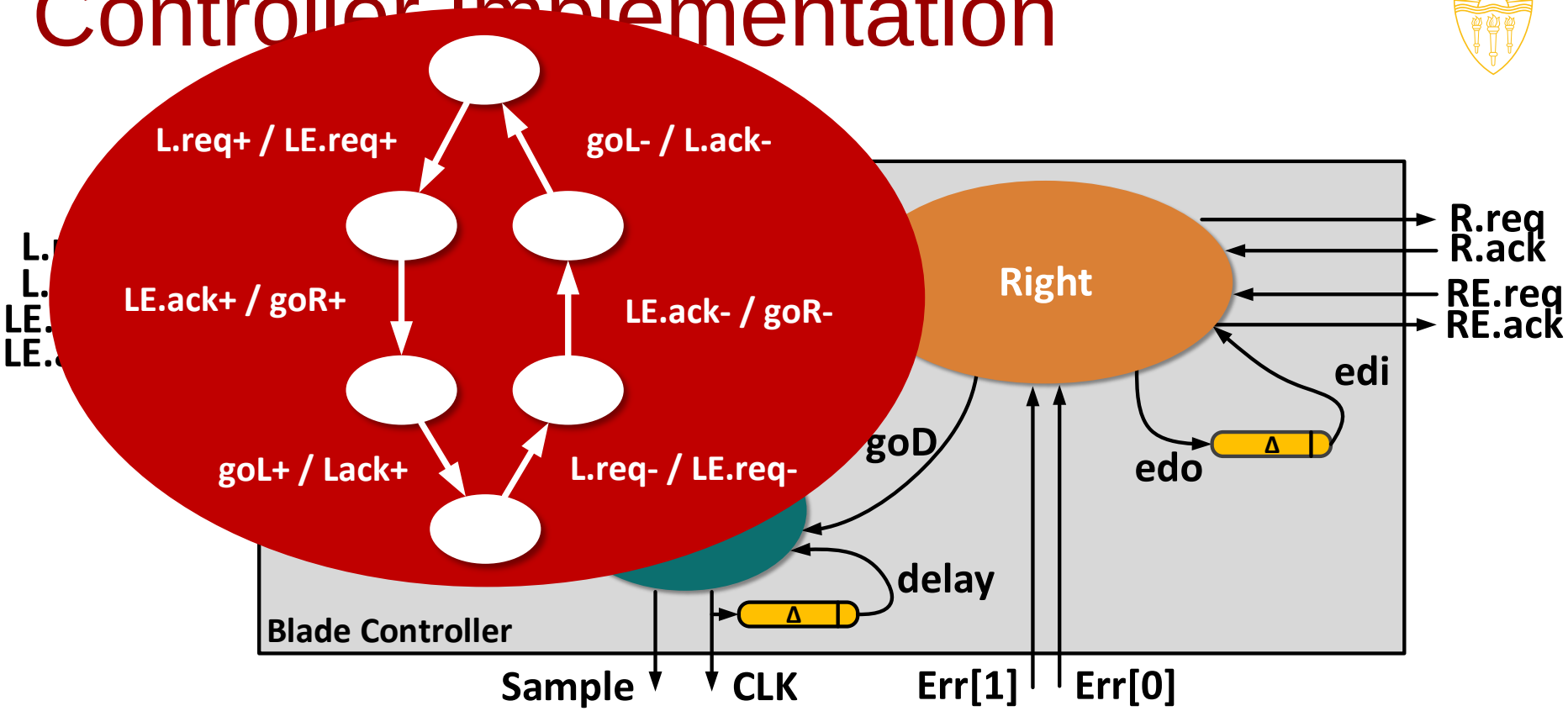
Controller Implementation



Three part burst-mode state machine

- Implemented using 3D [Yun, 1992]

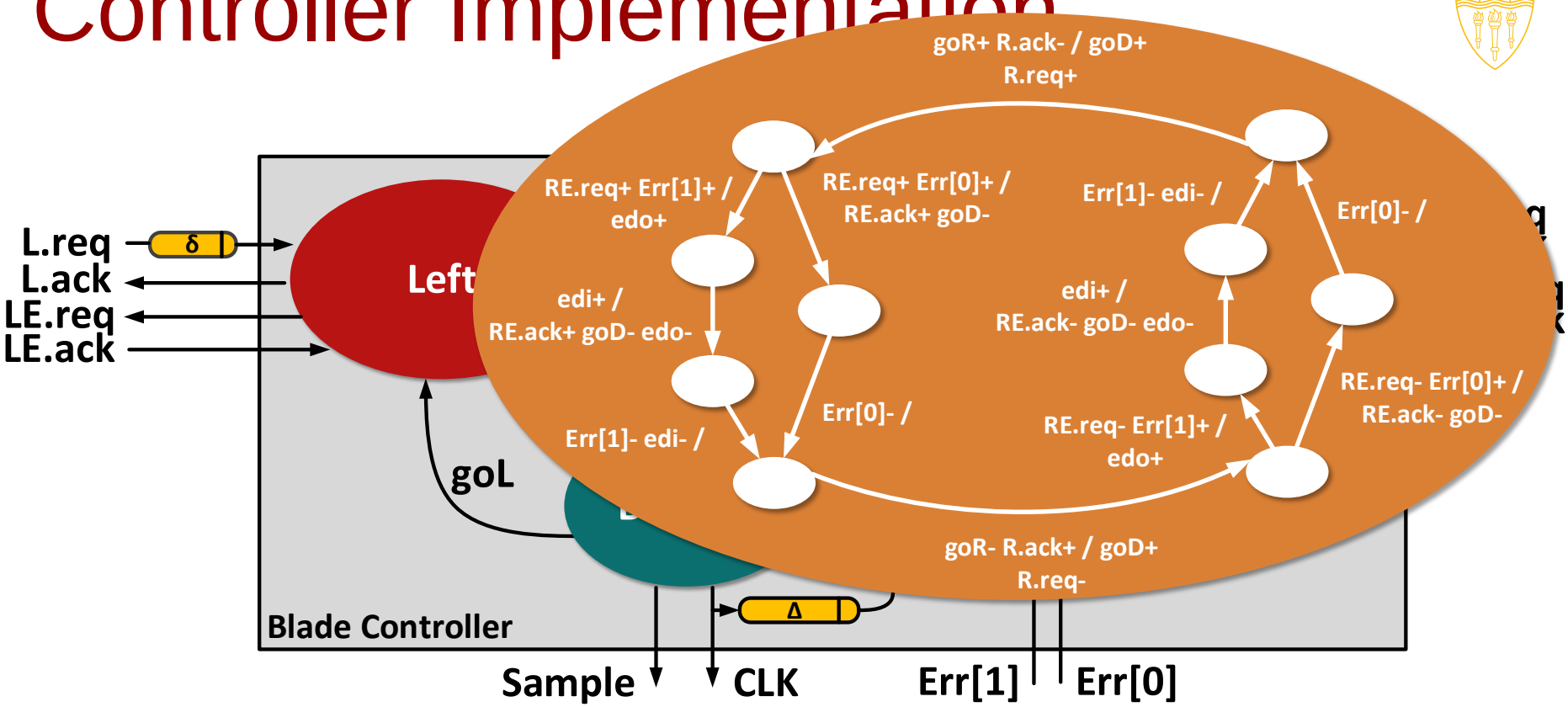
Controller Implementation



Three part burst-mode state machine

- Implemented using 3D [Yun, 1992]

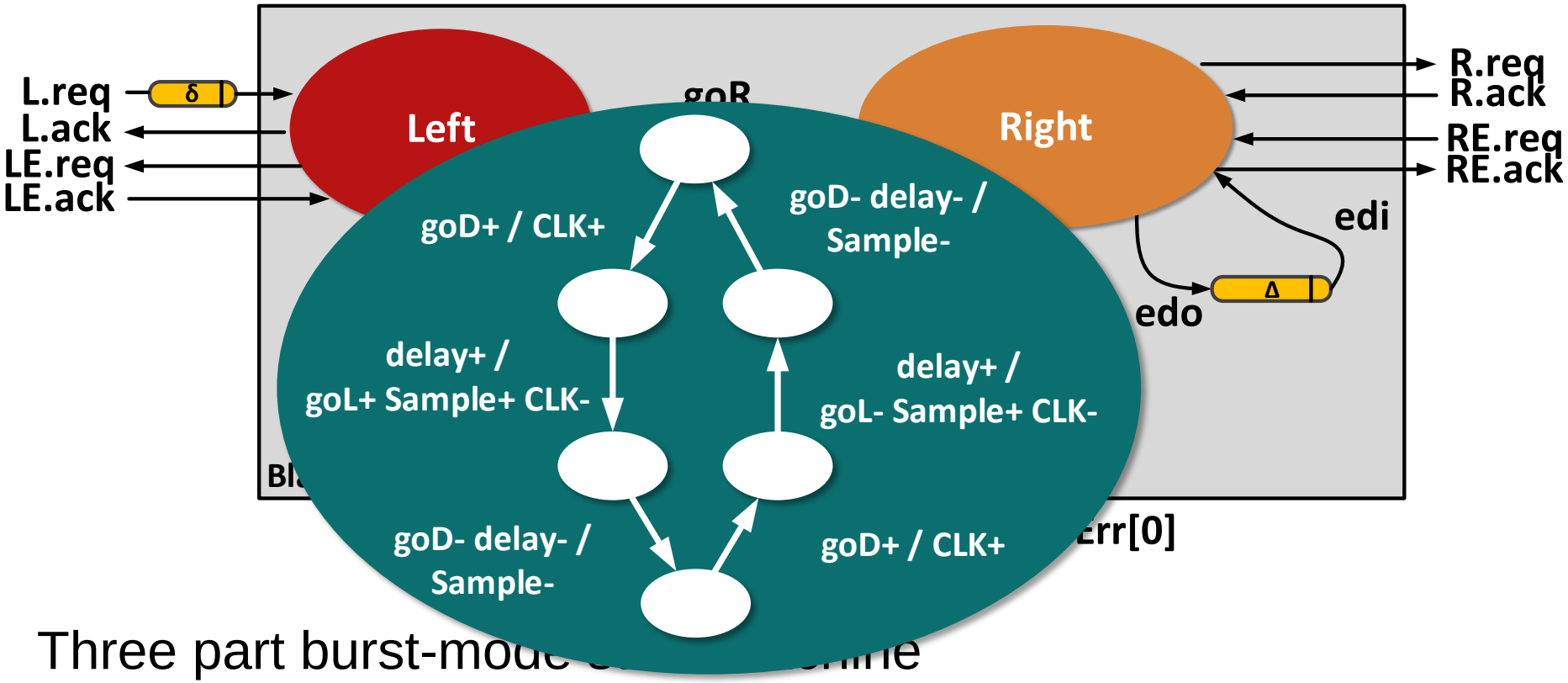
Controller Implementation



Three part burst-mode state machine

- Implemented using 3D [Yun, 1992]

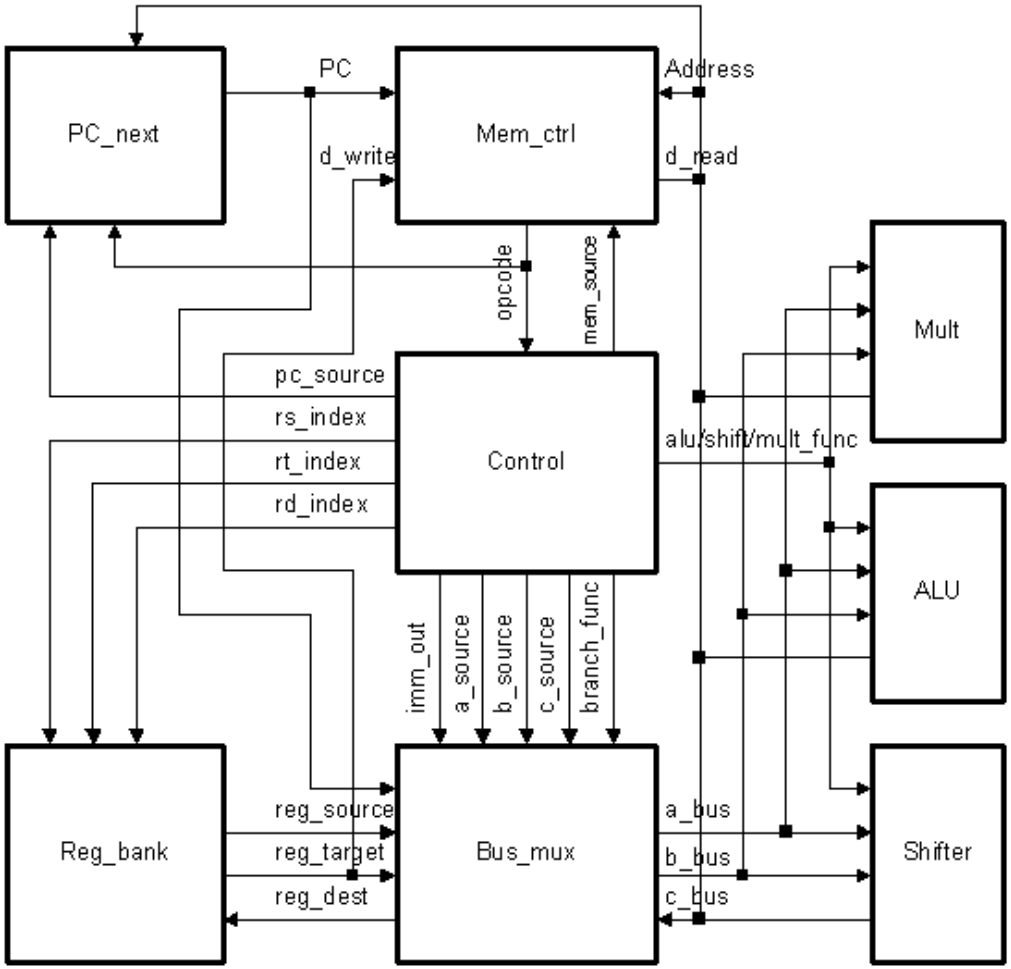
Controller Implementation



Three part burst-mode

- Implemented using 3D [Yun, 1992]

Case Study: Plasma



MIPS OpenCore

3-stage pipeline

28nm FDSOI @ 666MHz
(w/ ideal clock and Vdd)

Type	Count
Combinational	11,740
Buf/Inv	1,683
Seq. (Non-RF)	531
RF	2,048
Total	14,319

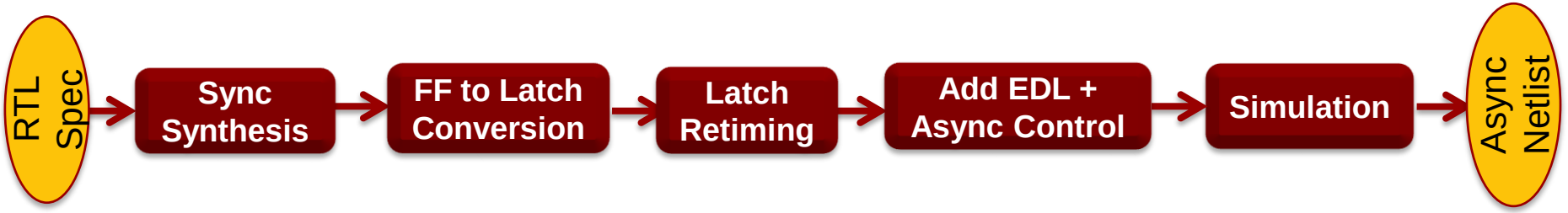
[1] http://opencores.org/project,plasma

Automatic Conversion Flow

Convert single-clock sync RTL design to Blade

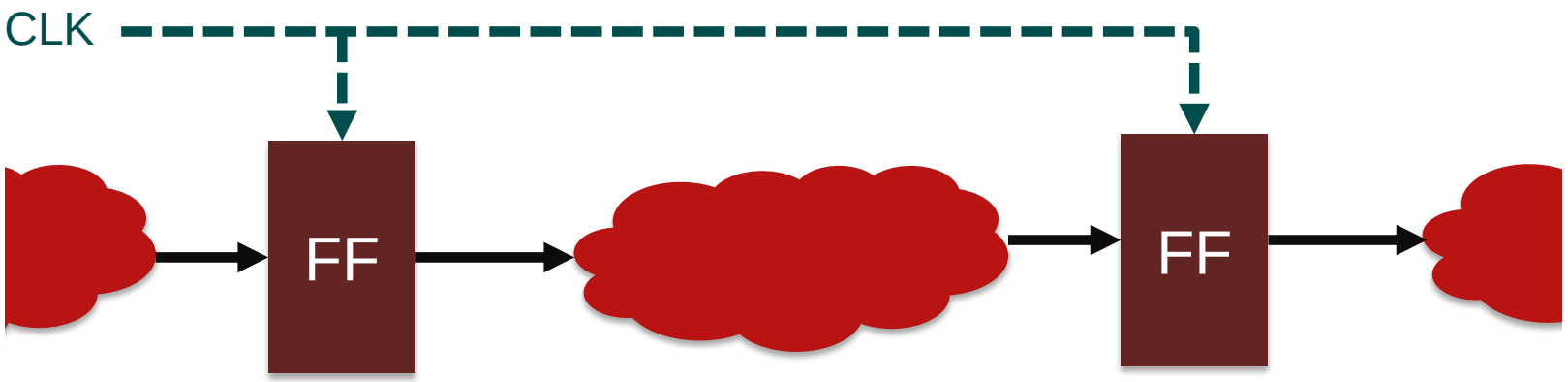
Re-uses synchronous EDA tools and libraries

Seamless integration into existing flows



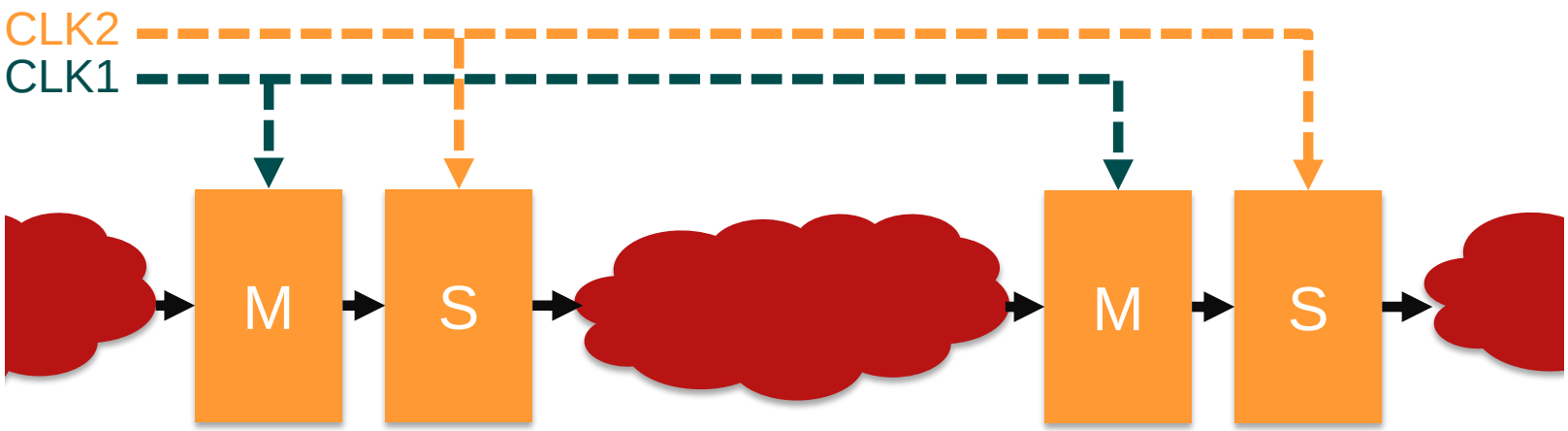
Blade Design Flow

Synthesis



Synthesize sync RTL design using standard EDA tools

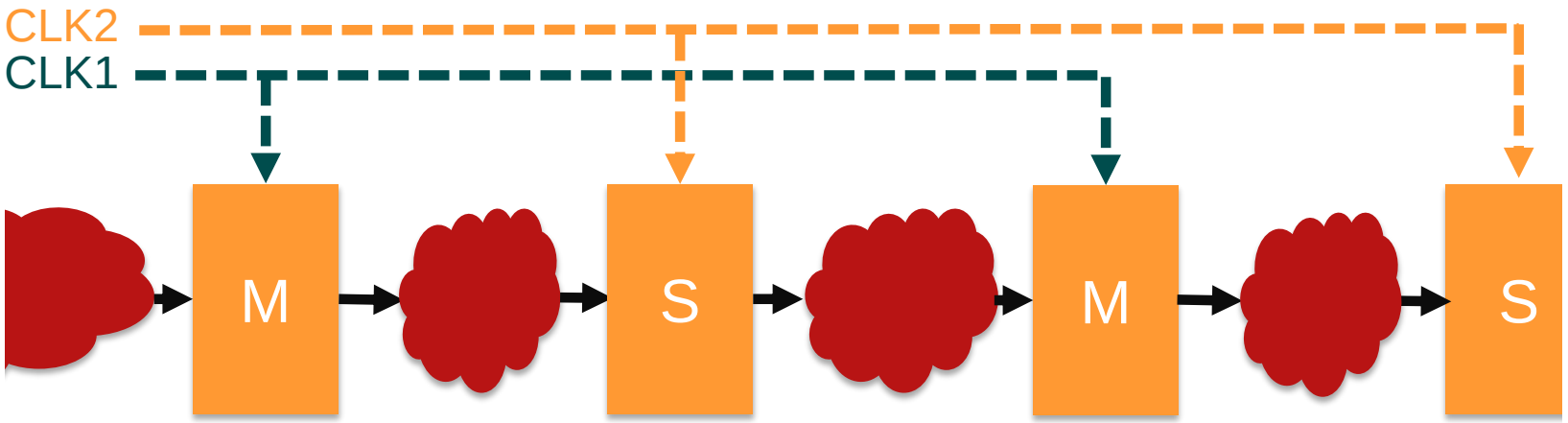
Latches



Replace flip flops with master-slave latches

Two-phase non-overlapping clocking

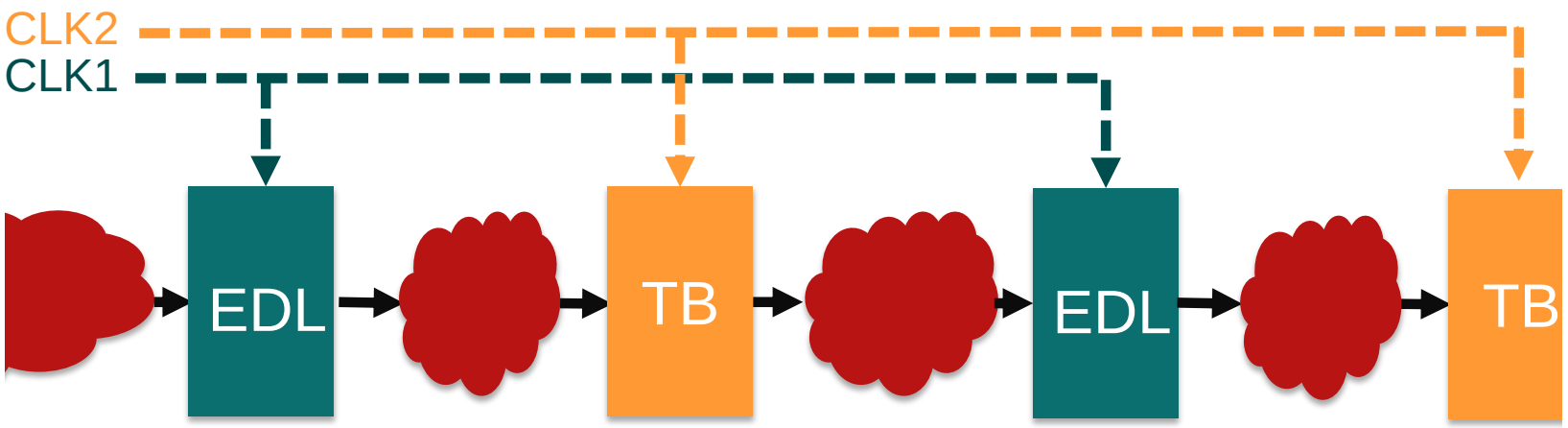
Latch Retiming



Retime latches to spread logic delay across stages

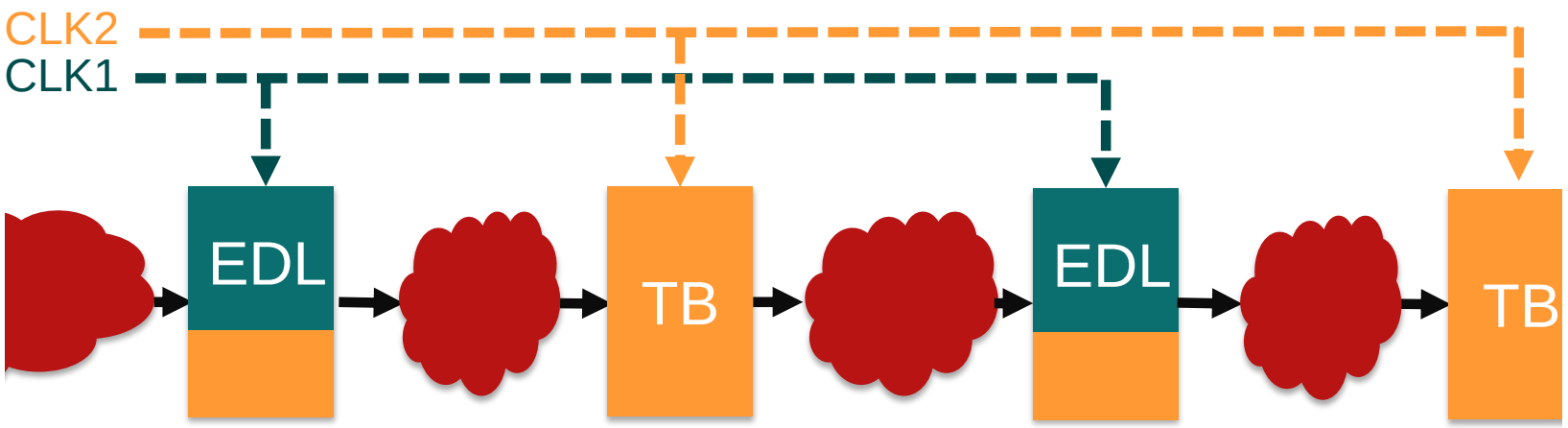
Allow time borrowing to reduce area overhead

EDL Insertion



Replace non-TB latches with error detecting latches

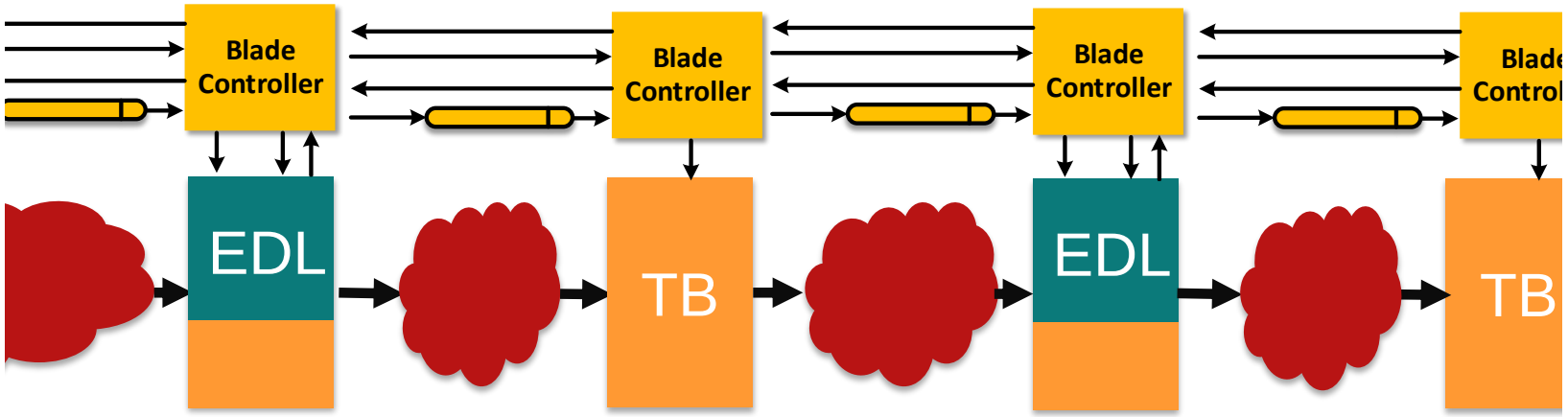
EDL Insertion



Replace non-TB latches with error detecting latches

Not all latches need be error detecting

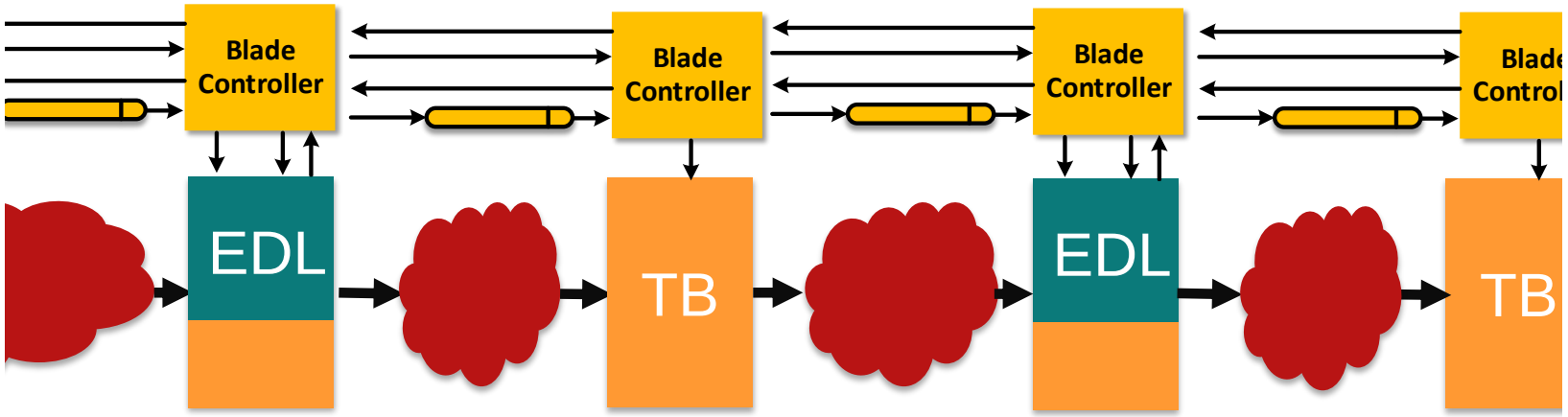
Async Control



Remove synchronous clock trees

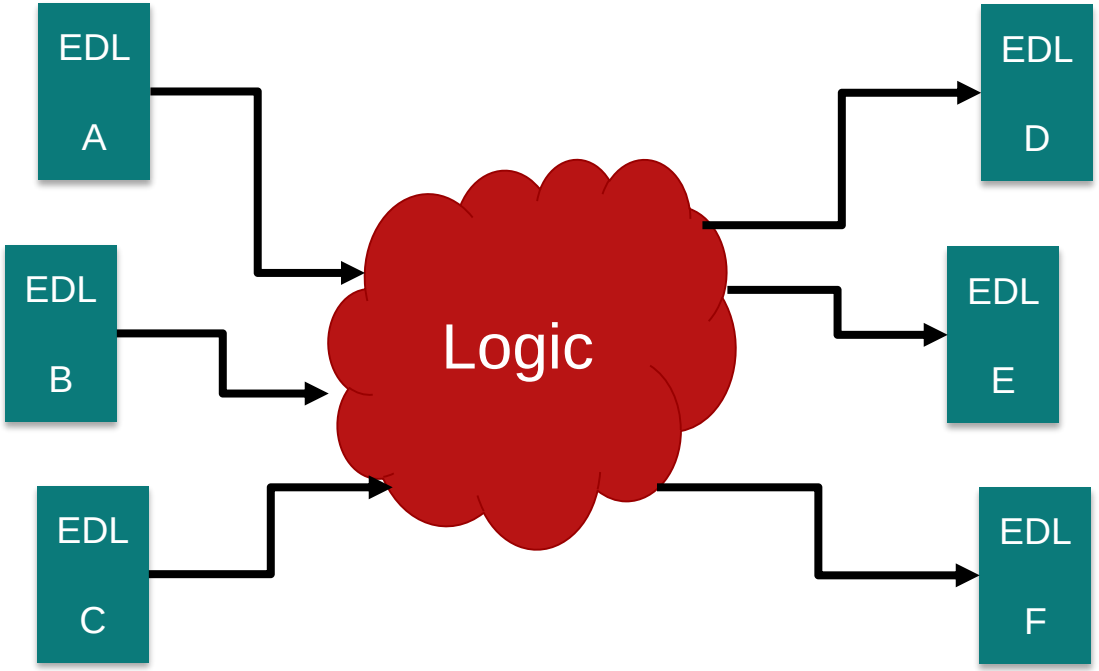
Add Blade controllers and delay lines

Simulation

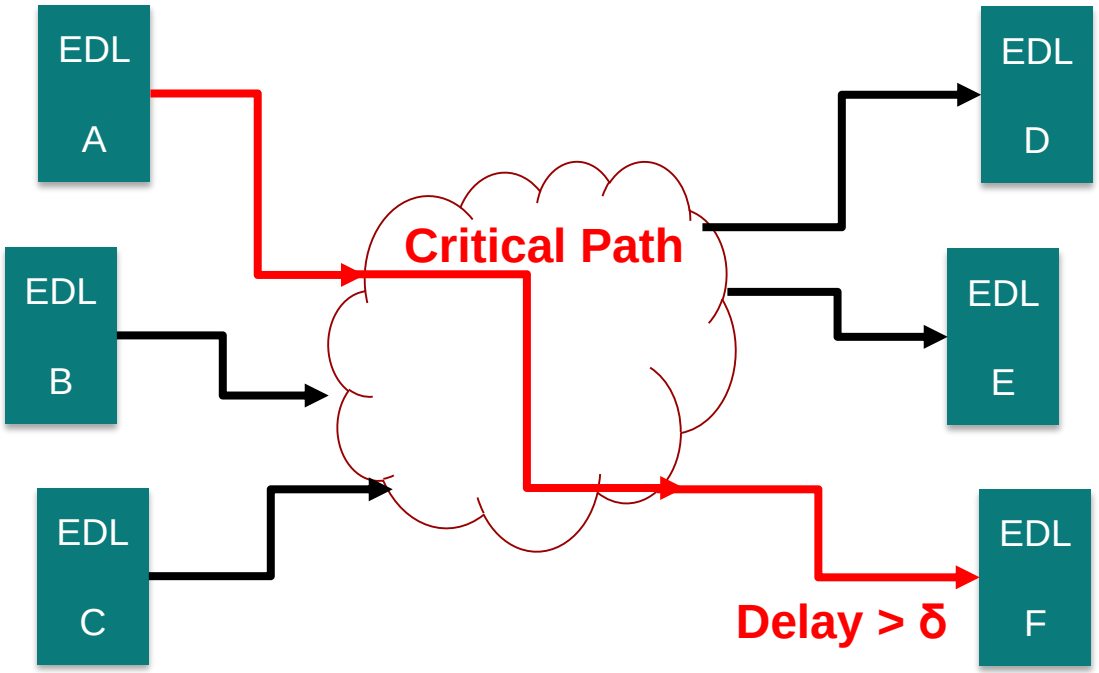


Back annotated SDF simulation using final netlist

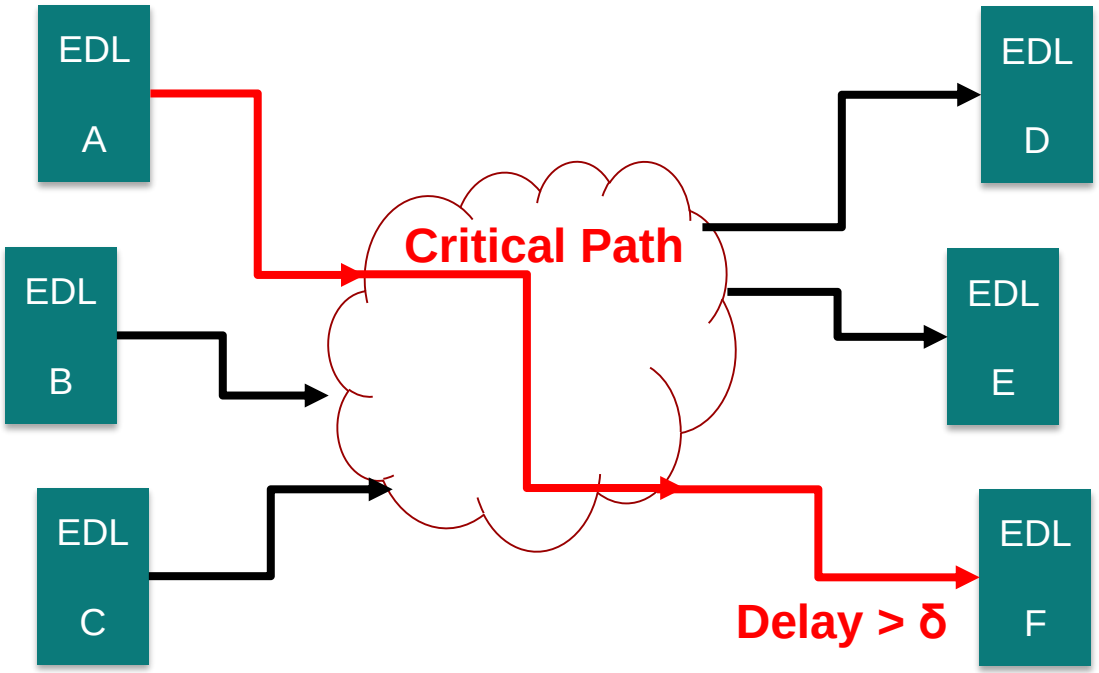
Resynthesis



Resynthesis

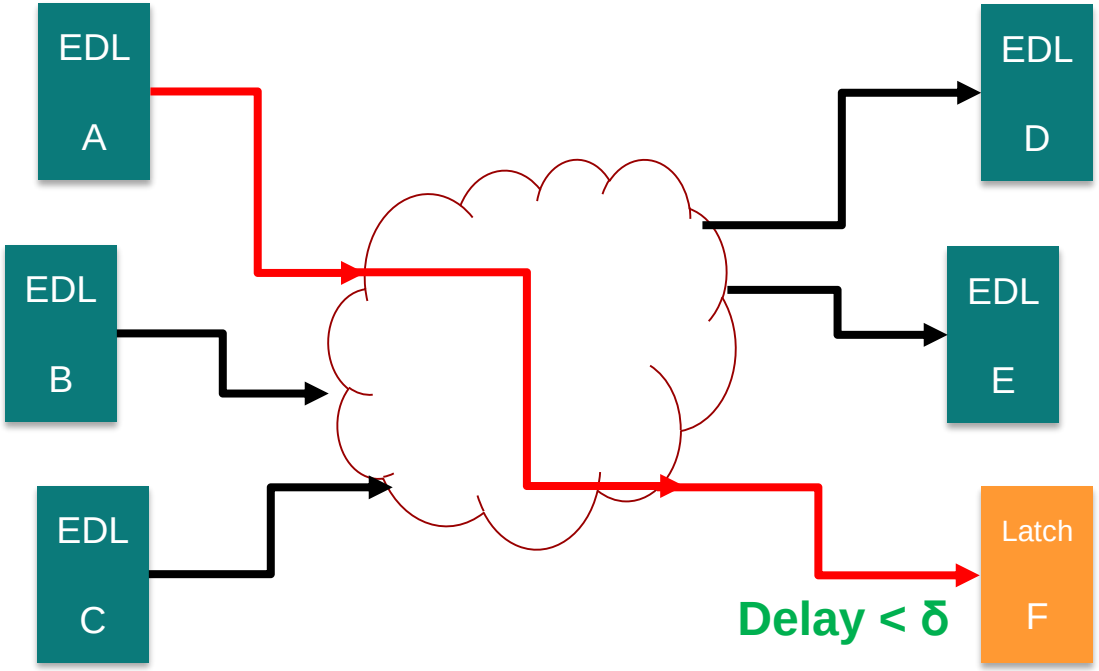


Resynthesis



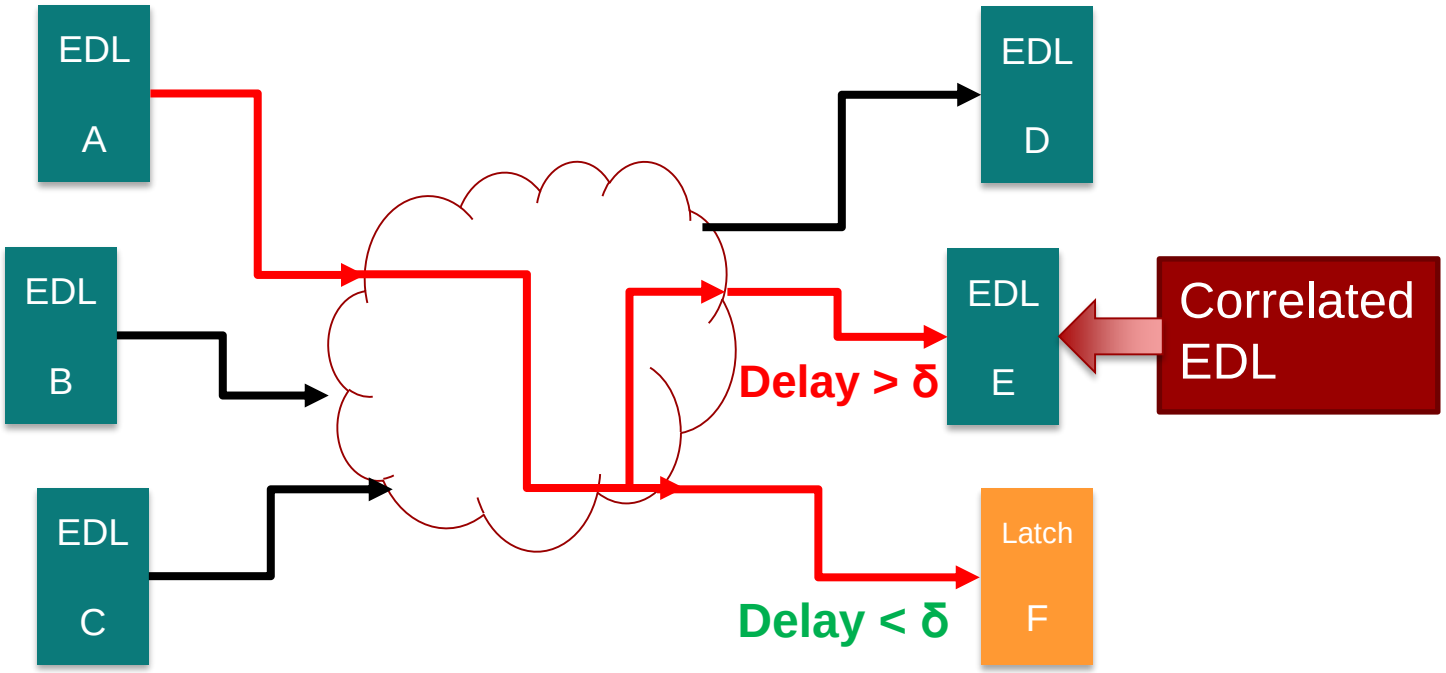
Add set_max_delay from A to F = δ

Resynthesis



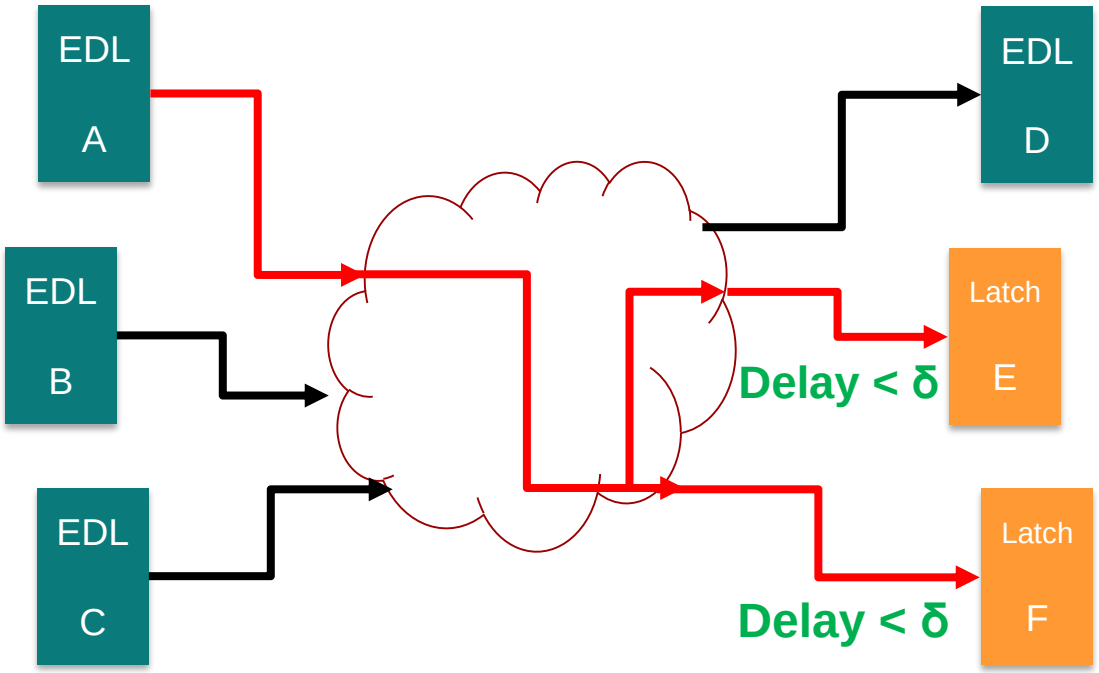
Add set_max_delay from A to F = δ

Resynthesis



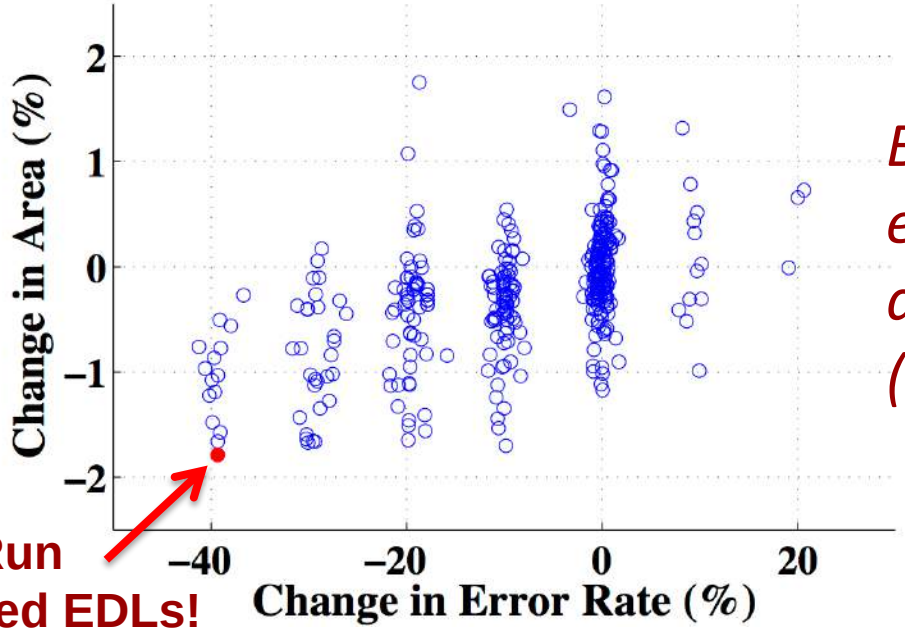
Add set_max_delay from A to F = δ

Resynthesis



Add set_max_delay from A to F = δ

Brute Force Resynthesis



Best Run
122 Correlated EDLs!

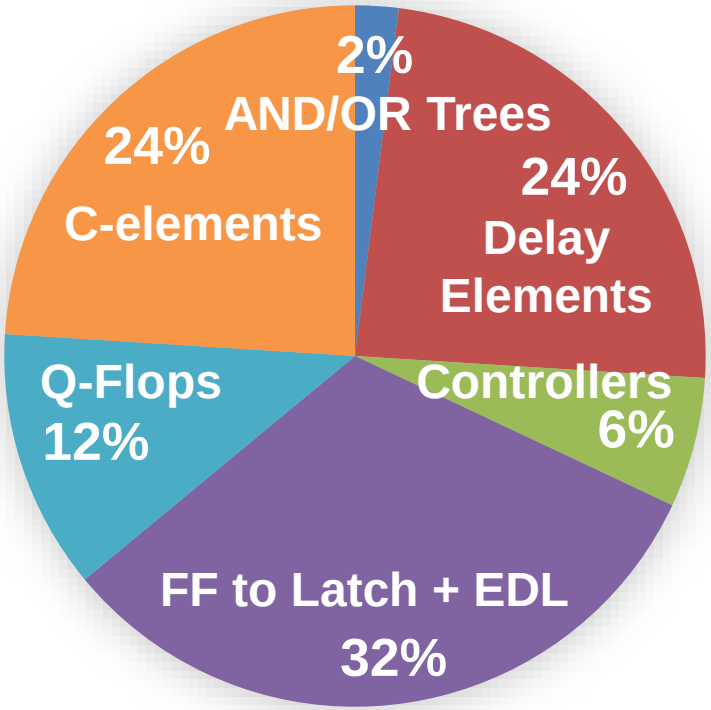
Best run reduced error rate by 39% and EDLs by 27% (-1.9% area)

Evaluated hundreds of resynthesis runs

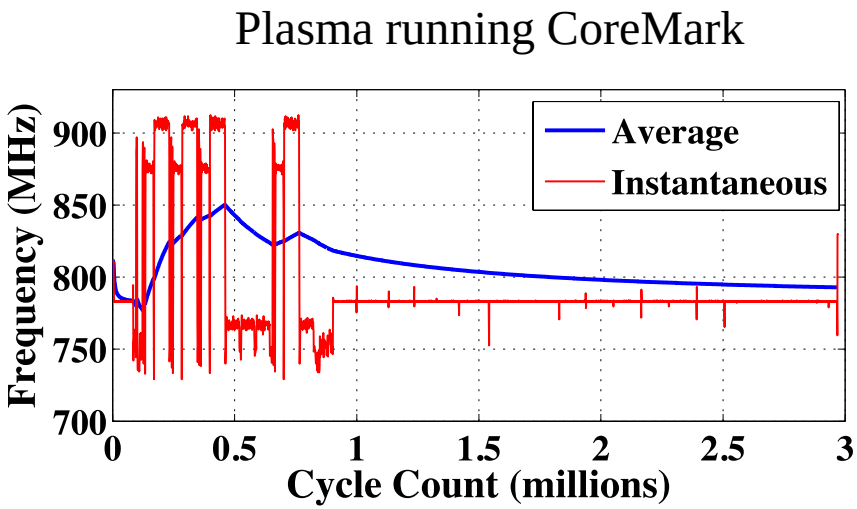
- Each run sets a max delay constraint to a single latch

Chose result that led to largest reduction in area and error rate

Area and Performance



Overall area overhead is 8.4%



Performance increases

- Average: 19%
- Peak: 42%

Performance Comparison with Margins

Must add margins for PVT variation, clock skew / jitter, and/or aging

- Synchronous frequency degraded to accommodate

Margin in Blade design is only imposed when an error occurs

- A 30% error rate reduces impact of margin by ~70%

Margin	Synchronous	Blade (30% ER)	% Advantage
0%	666MHz	800MHz	20%
15%	566MHz	764MHz	35%
30%	466Mhz	728MHz	**56%**

Other Related Work

Canary Circuits [Sato, 2007]

- Removes some PVT margins
- But cannot take advantage of data dependency

Bundled Data Designs [Sutherland'89, Nowick'97]

- Speculative completion sensing exploits *some* data dependency
- Margins impact performance on every cycle
- No observability of errors

Soft Mousetrap [Liu, 2013]

- Hold time constraints remain difficult to meet

Conclusions

Blade Template

- Achieves higher performance by exploiting data dependency
- Benefits from average vs worst-case MS resolution times
- Reduces impact of margins for PVT variations
- Enables voltage scaling for power savings

Plasma Case Study

- Highlights design and CAD techniques for area efficiency
- Achieves 19% increase in performance with 8.4% area overhead

Questions?