High-throughput pipelined interconnect in FPGAS

MASc Examination
Paul Teehan

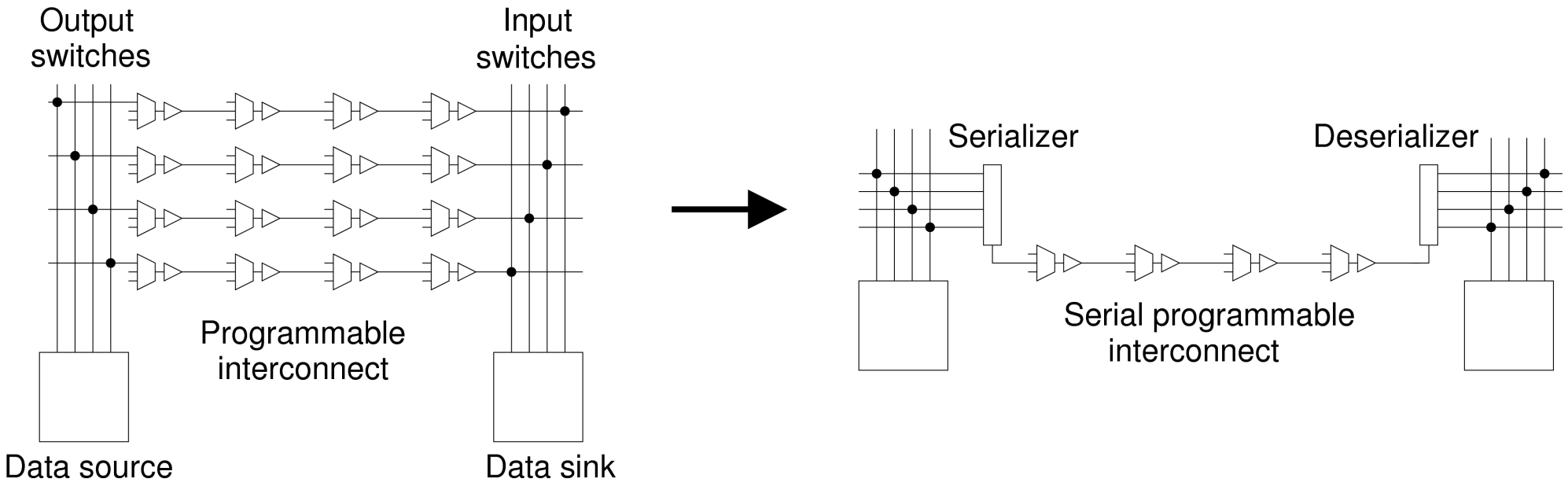
Supervisors:
Dr. Guy Lemieux
Dr. Mark Greenstreet

Oct. 28th, 2008

Motivation: FPGA interconnect

- Programmable wires are costly
  - Muxes and buffers need lots of area
- Wires are in demand
  - Often run out before logic
  - Especially in datapath (word-oriented) circuits
- But wires are underutilized
  - Actual throughput: 0.1 to 0.2 Gbps (user clock constrained)
  - Wire bandwidth: 6Gbps (if wave pipelined)

Pipelined serial interconnect



- Increase throughput, reduce area
- Pipelining techniques: wave pipelining, surfing

Research questions

- In serial interconnect in FPGAs worth pursuing?
  - How would it work?
  - What are the area savings?
  - How fast can it go?
  - How much overhead?
- How reliable is interconnect pipelining?
  - Is surfing better than wave pipelining?
  - How well do they work in high-noise environments?

Related work

- Datapath FPGAs [1]
- Wave pipelining in FPGAs [2]
  - Predicted throughput: 1.4Gbps (simulated)
  - This thesis: 3Gbps (simulated); focus on reliability
- Surfing pipelines [3]
  - Not previously applied to FPGAs
  - Should be more reliable than wave pipelining

[1]: A. Ye, “FPGA architectures and algorithms optimized for implementing datapath circuits,” Ph.D dissertation, University of Toronto, 2004.
[2]: T. Mak et al, “Wave-pipelined signalling for on-FPGA communciation,” FPT 2008
[3]: S. Yang et al, “A jitter attenuating timing chain,” ASYNC 2007

Contributions

- Design of serial interconnect for FPGAs using wave pipelining and surfing
  - High-level area estimation shows 10-60% area savings
- Reliability estimates of serial signaling
  - Using a new statistical model of timing uncertainty
  - Surfing is shown to be more robust than wave pipelining
- HSPICE simulations in CMOS 65nm process
  - Achieved throughput of 3Gbps for 50-stage/25 mm link
  - Latency penalty of 50 to 100%; power penalty of 6X to 14X

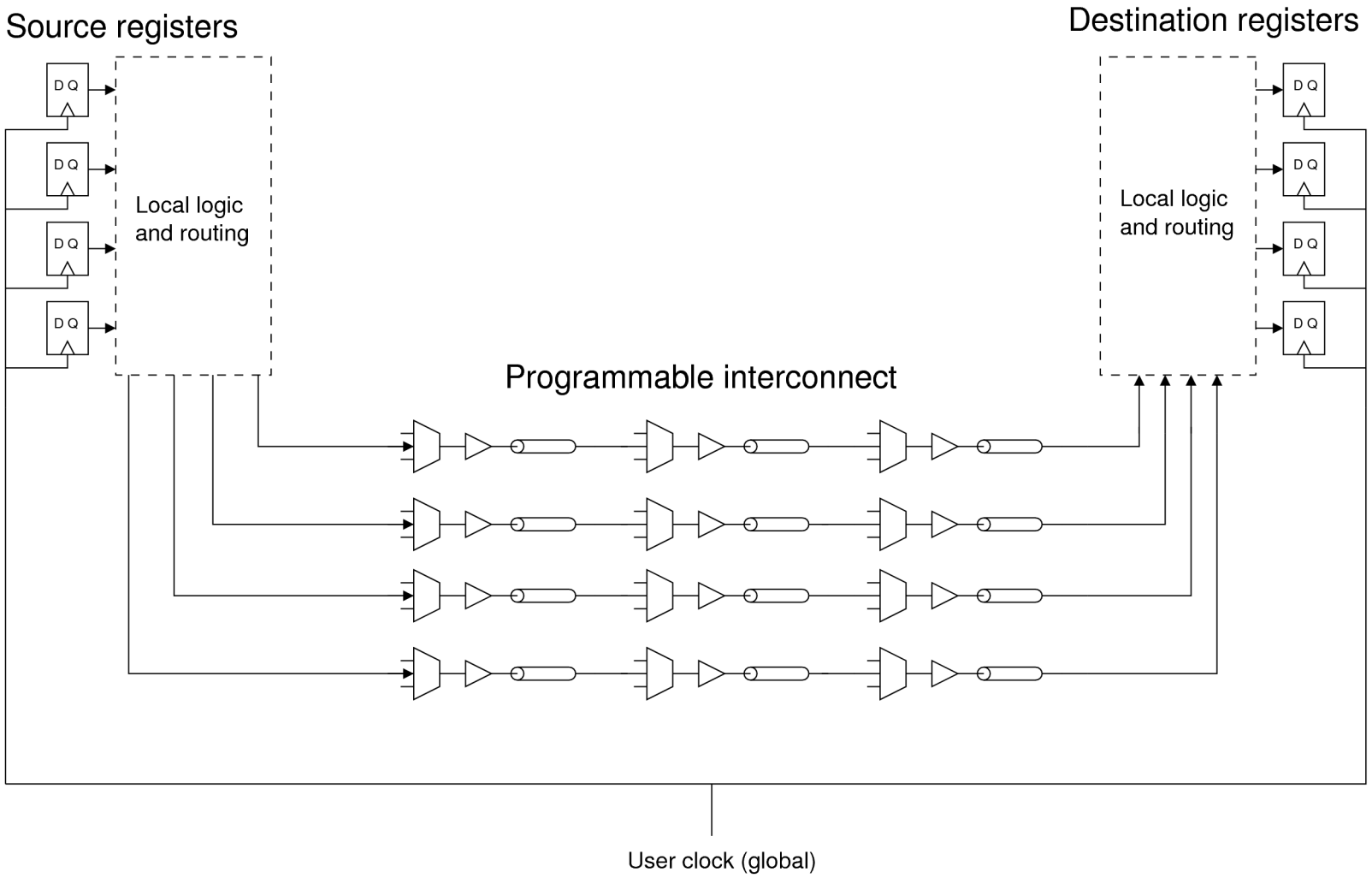
Contributions

- Recently submitted to FPGA 2009
  - P. Teehan, G. Lemieux, and M. Greenstreet, “Towards reliable 5Gbps wave-pipelined and 3Gbps surfing interconnect in 65nm FPGAs”
  - Currently under review

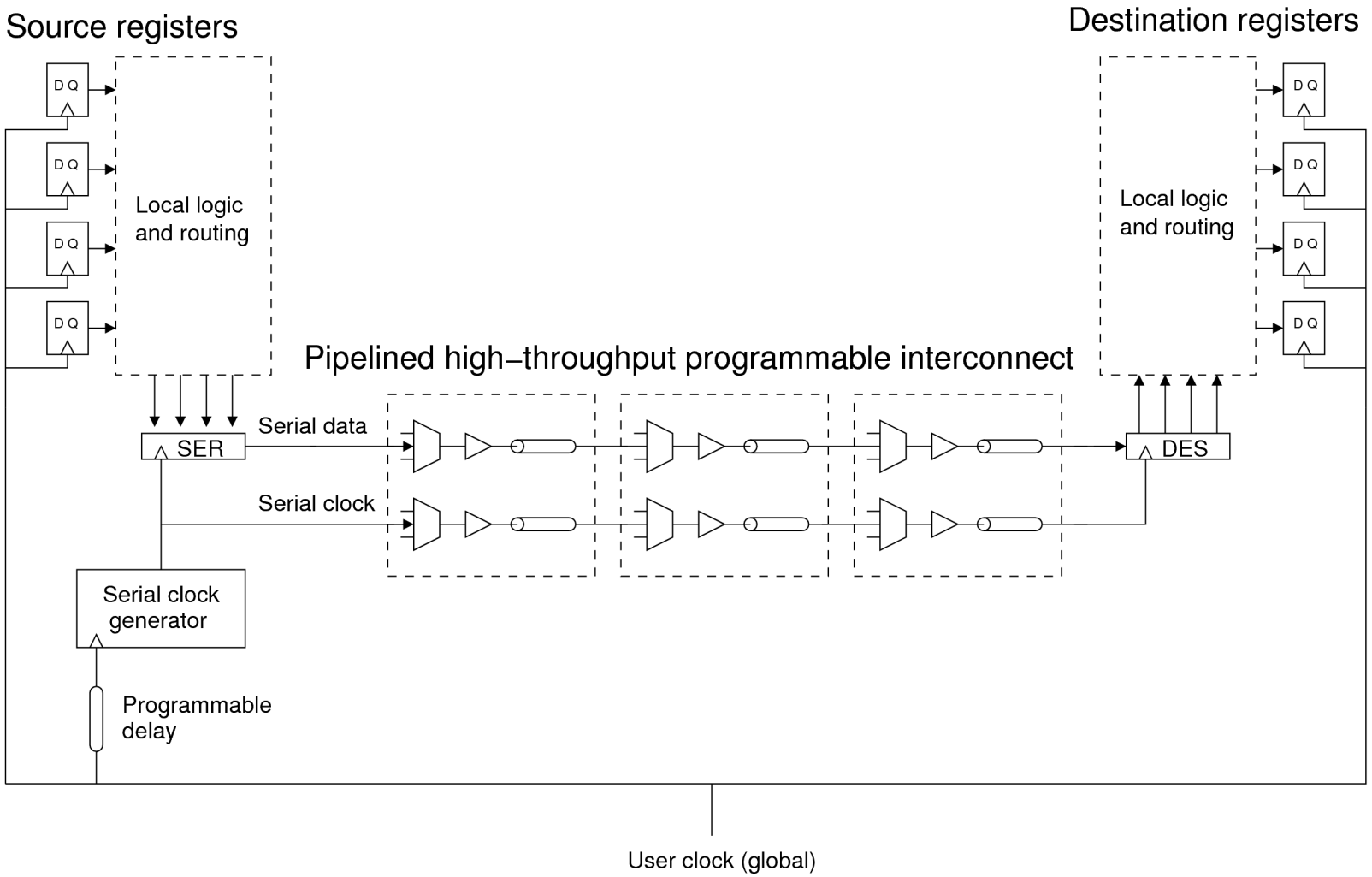
Research questions

- Are serially-interconnected FPGAs worth pursuing?
  - How would it work?
  - What are the area savings?
  - How fast can it go?
  - How much overhead?
- How reliable is interconnect pipelining?
  - Is surfing better than wave pipelining?
  - How well do they work in high-noise environments?

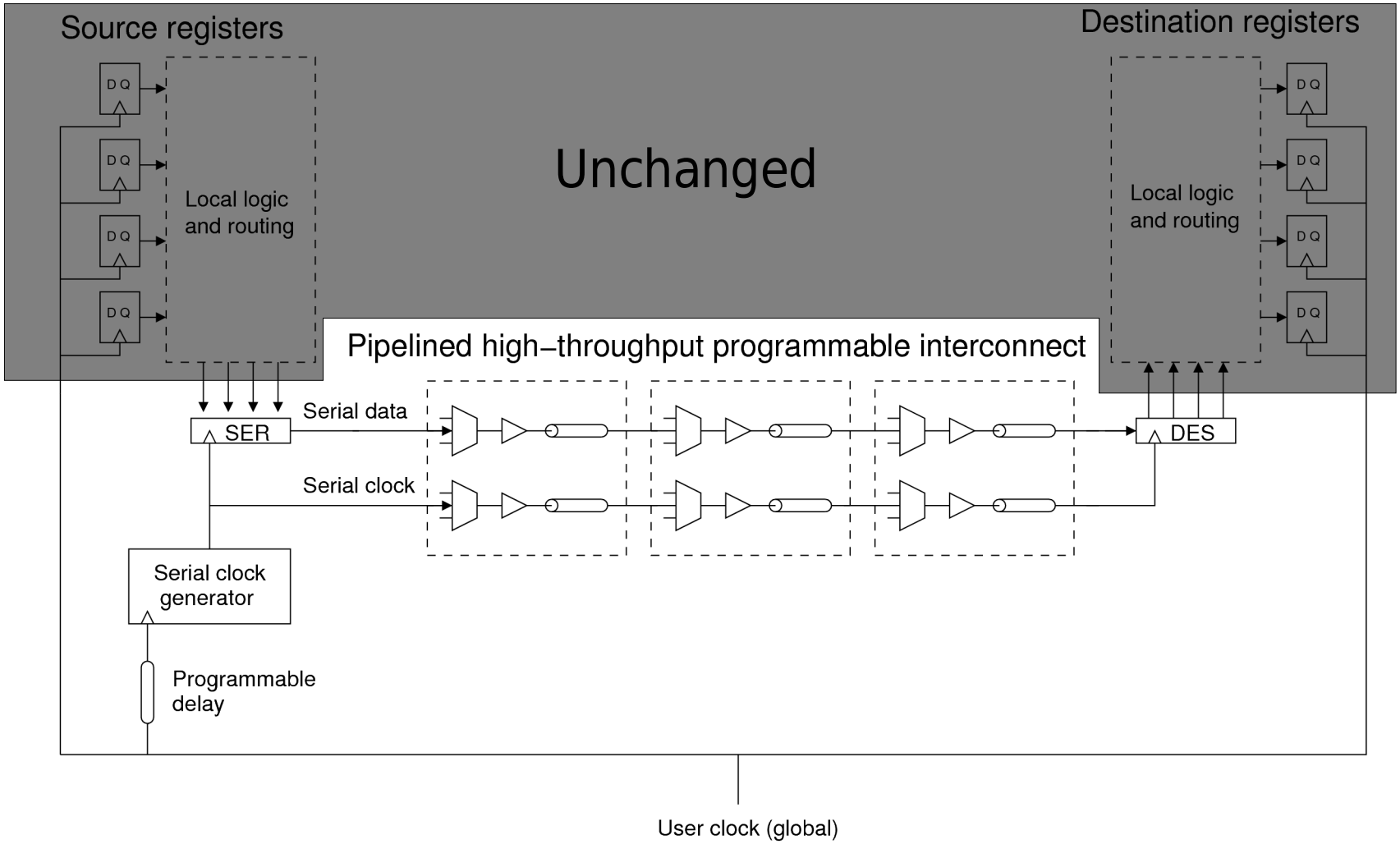
FPGA interconnect model



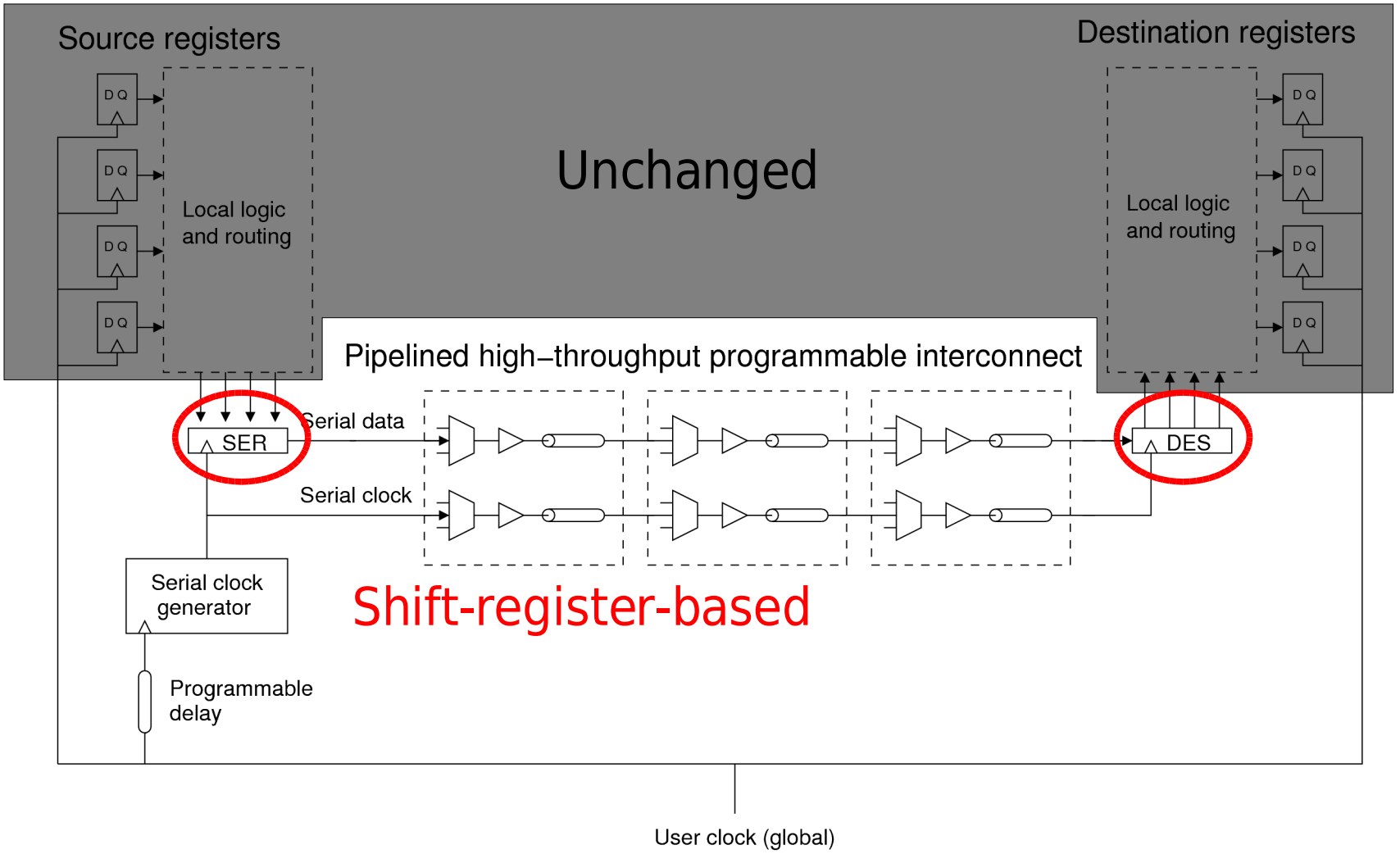
Convert to serial



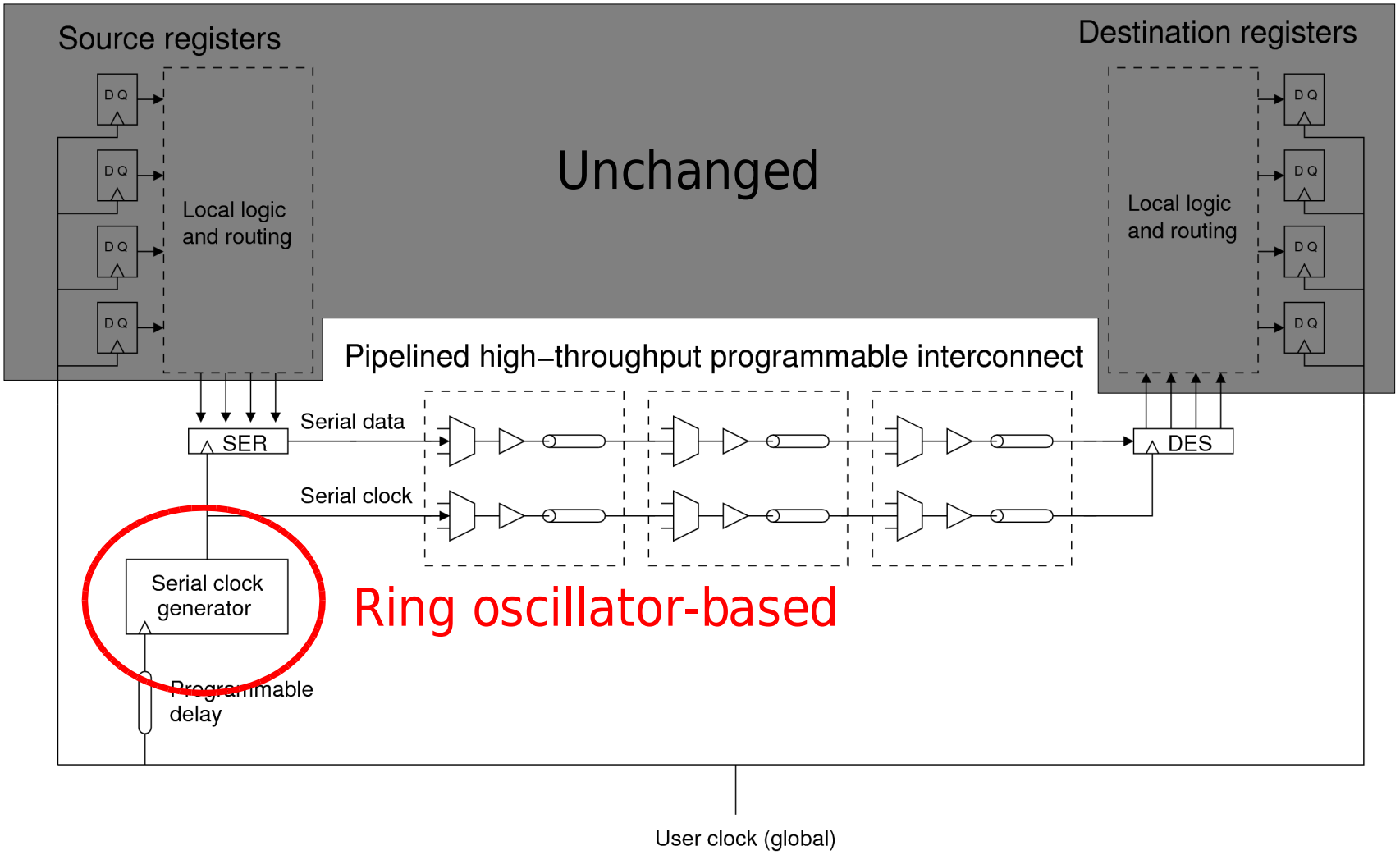
Changes in serial circuit



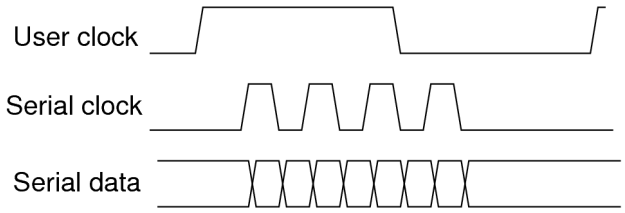
Serializers/Deserializers



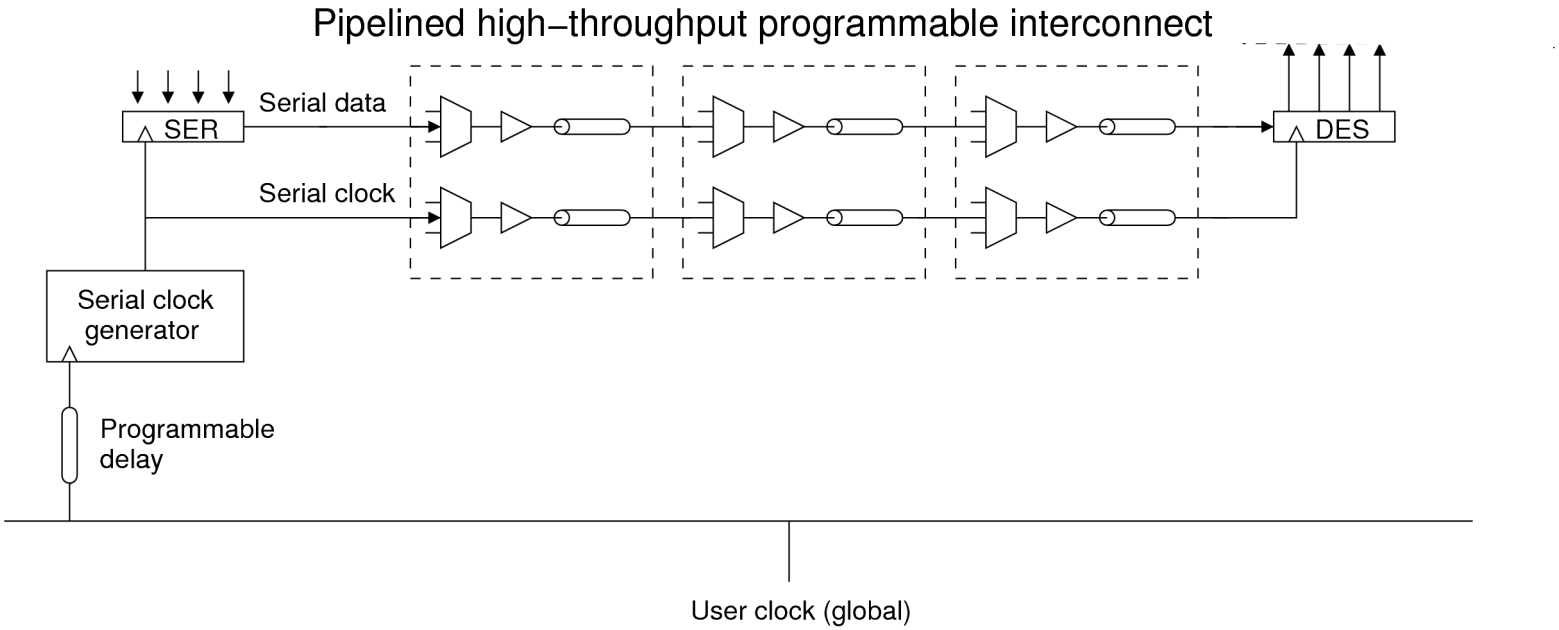
Serial clock



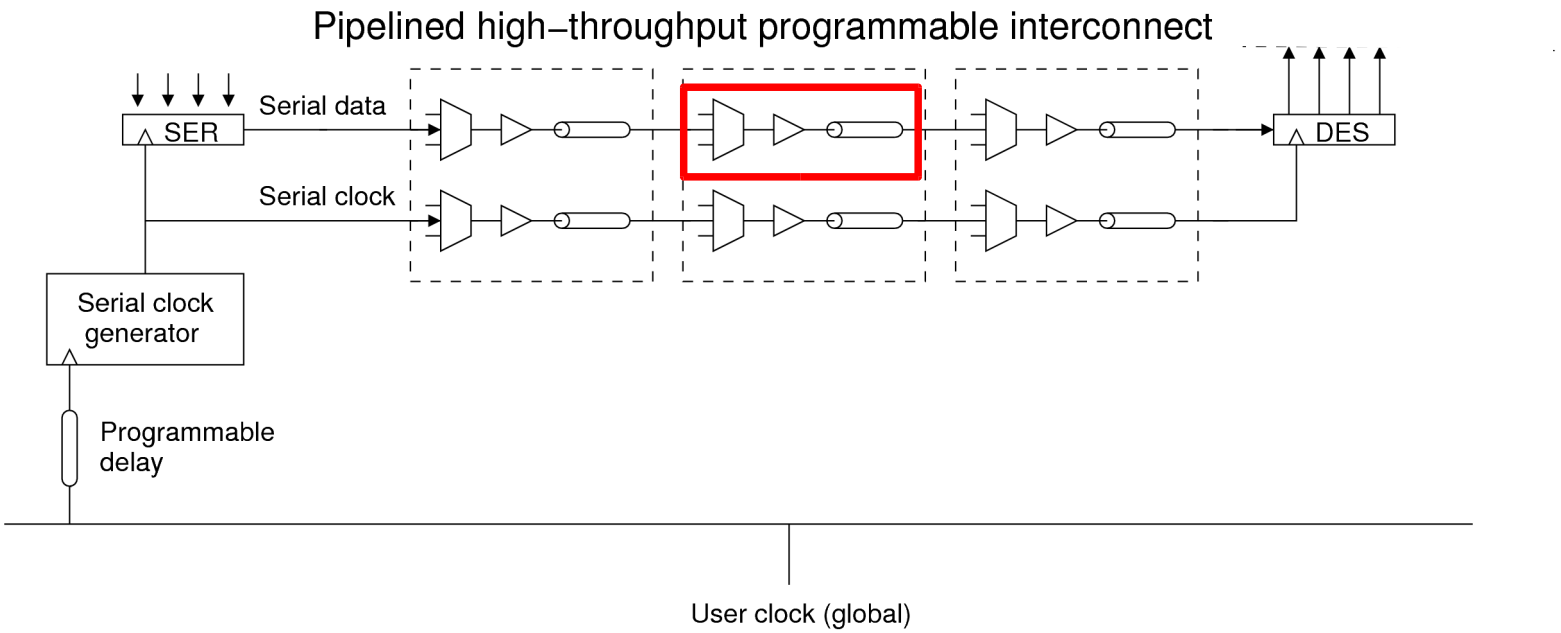
Signaling



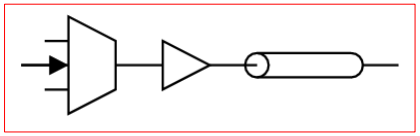
- Source-synchronous timing
- Serial clock sent alongside serial data
- Bursts (e.g. 8 bits)
- Double Data Rate clocking

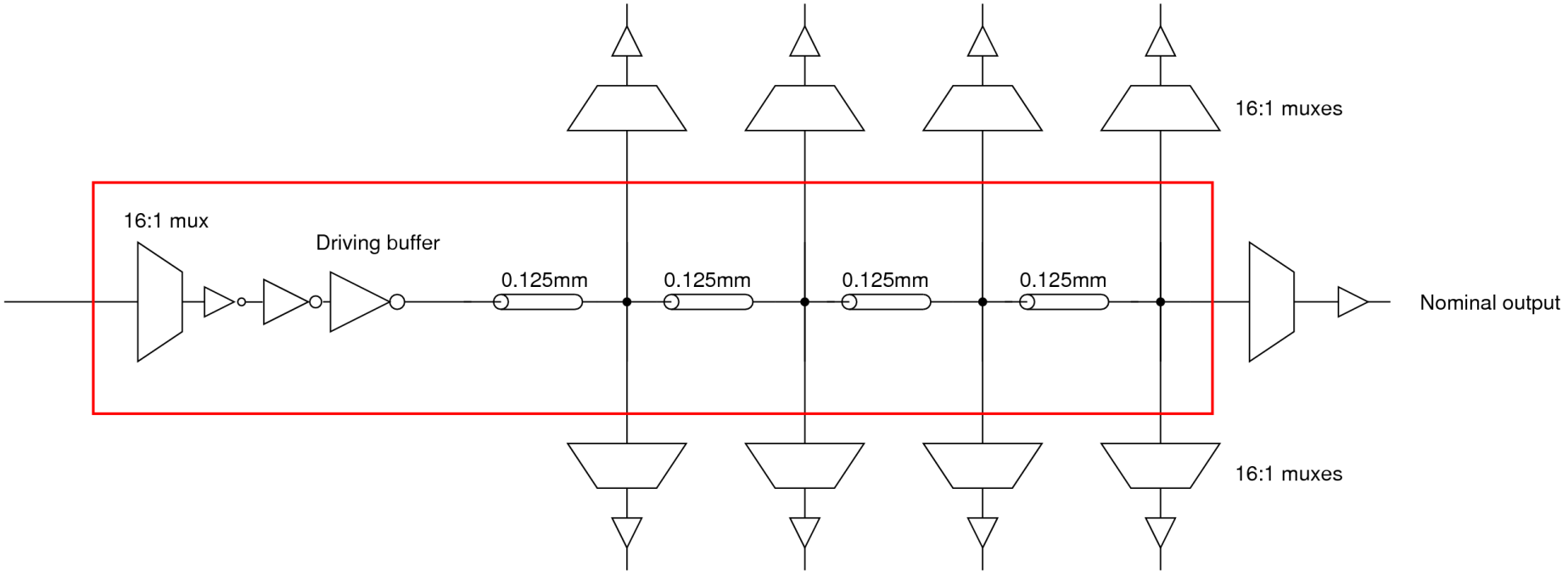


Interconnect stage detail



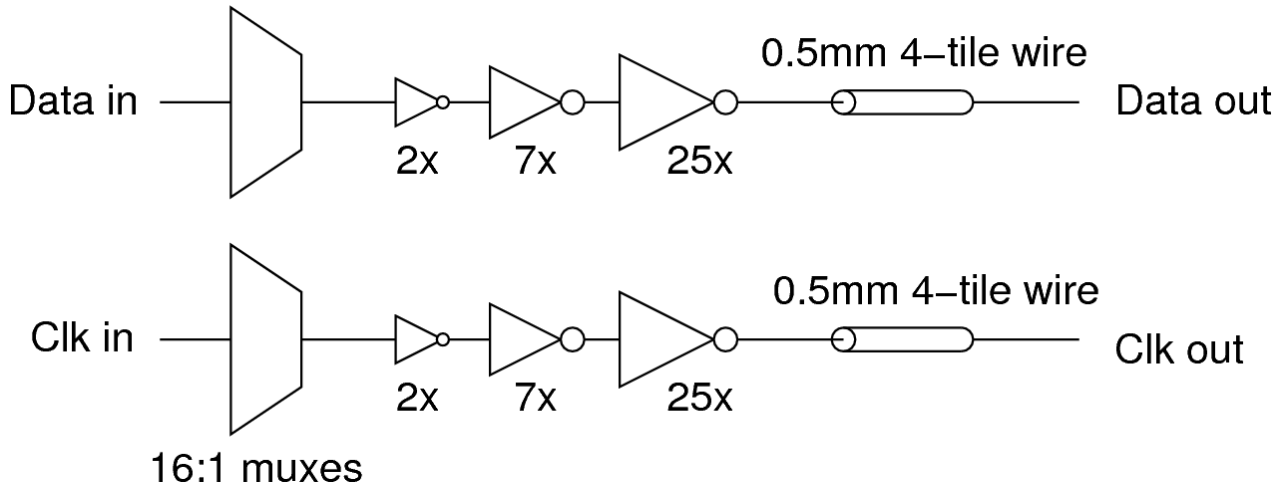
Interconnect stage detail





Wave pipelining circuit design

- Driver sizes chosen for high throughput
- Muxes are full CMOS pass-gates
  - Limiting factor for throughput!



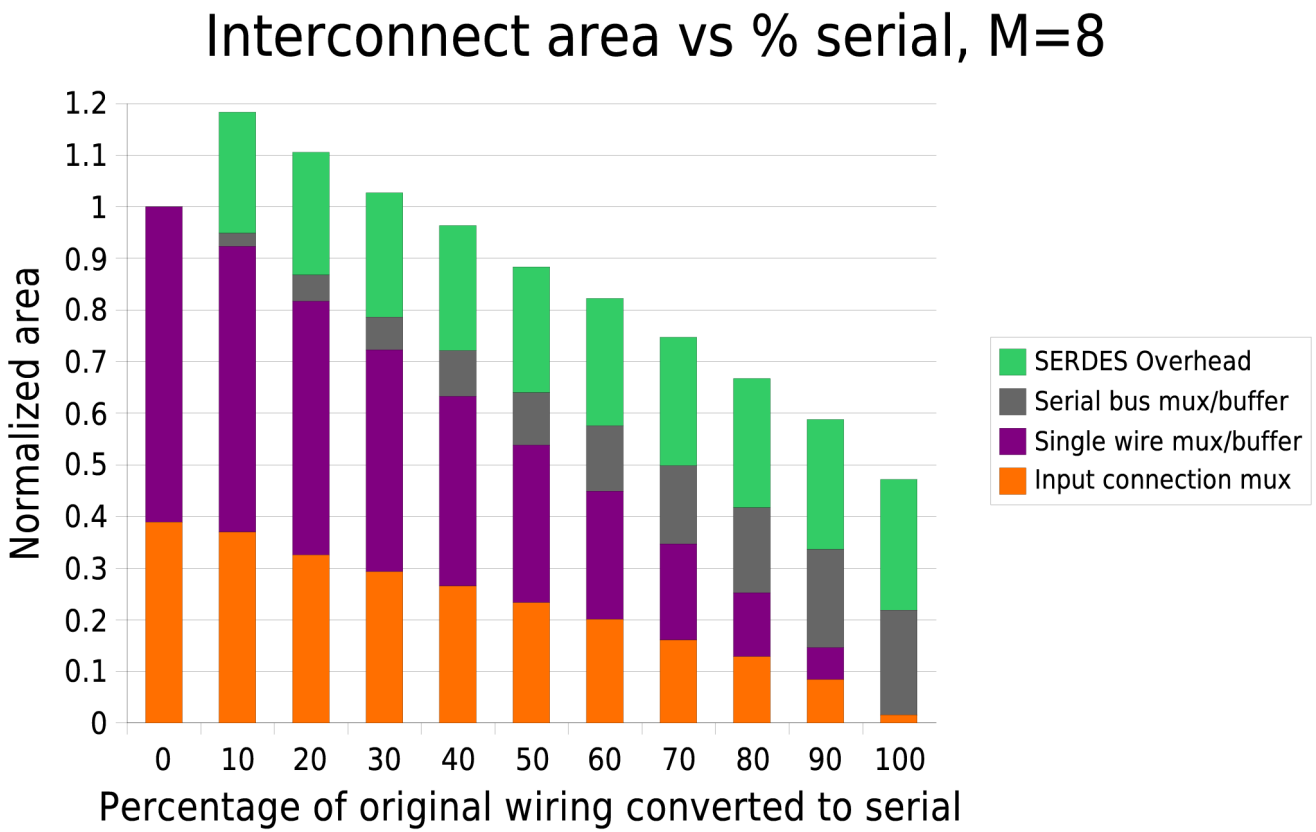
Wave pipelining vs surfing

- Simple; minimal modifications
  - But jitter and skew will accumulate over long links
- Surfing: an alternative pipelining technique
  - Includes control circuitry (delay locked loop, pulsed latch) to remove jitter and skew.
  - More reliable, but small area, power, and latency overhead
  - See thesis for circuit details

Research questions

- Are serially-interconnected FPGAs worth pursuing?
  - How would it work? √ **Source-synch pipelining**
  - What are the area savings?
  - How fast can it go?
  - How much overhead?
- How reliable is interconnect pipelining?
  - Is surfing better than wave pipelining?
  - How well do they work in high-noise environments?

System-level area estimate



- 10% area savings if half of wires are serial
- 50-60% if all wires are serial

Research questions

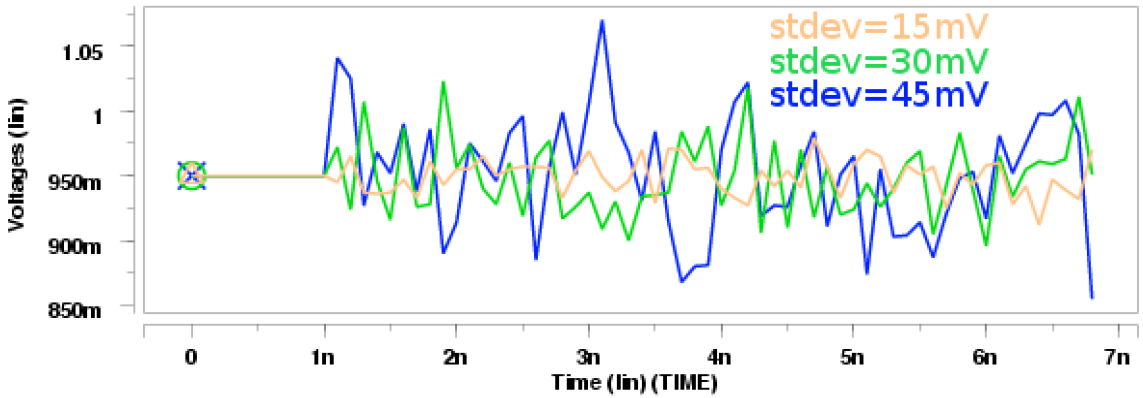
- Are serially-interconnected FPGAs worth pursuing?
  - How would it work? √ **Source-synch pipelining**
  - What are the area savings? √ **10-60%**
  - How fast can it go?
  - How much overhead?
- How reliable is interconnect pipelining?
  - Is surfing better than wave pipelining?
  - How well do they work in high-noise environments?

Failure modes

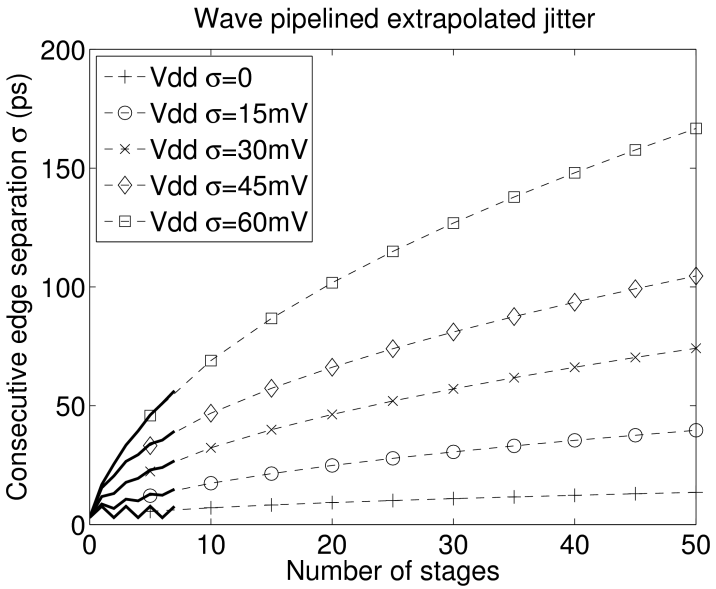
- Skew
  - Data and clock out of synch, incorrect sampling
- Jitter
  - Variation in period: consecutive edges may interfere
- The problem: cycle to cycle timing uncertainty
  - Crosstalk (shielding is necessary, and effective)
  - Fast supply variation (a problem!)

Timing uncertainty: supply noise

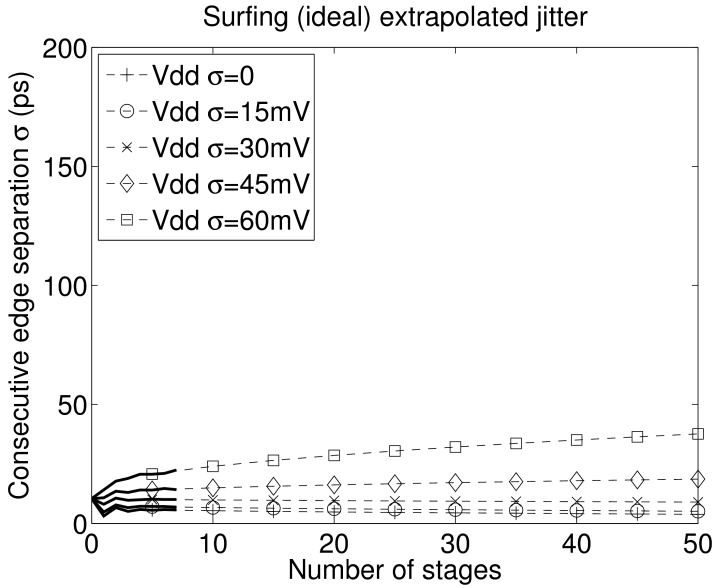
- What does supply noise look like?
  - We don't know – in FPGAs, depends on user design
  - Slow variations have little effect on cycle-to-cycle timing
  - Assume fixed DC component and quickly-varying component
  - Model fast noise as normal random variable which takes a new value every 100ps; standard deviation left as a parameter.



Jitter due to fast supply noise



Wave pipelining



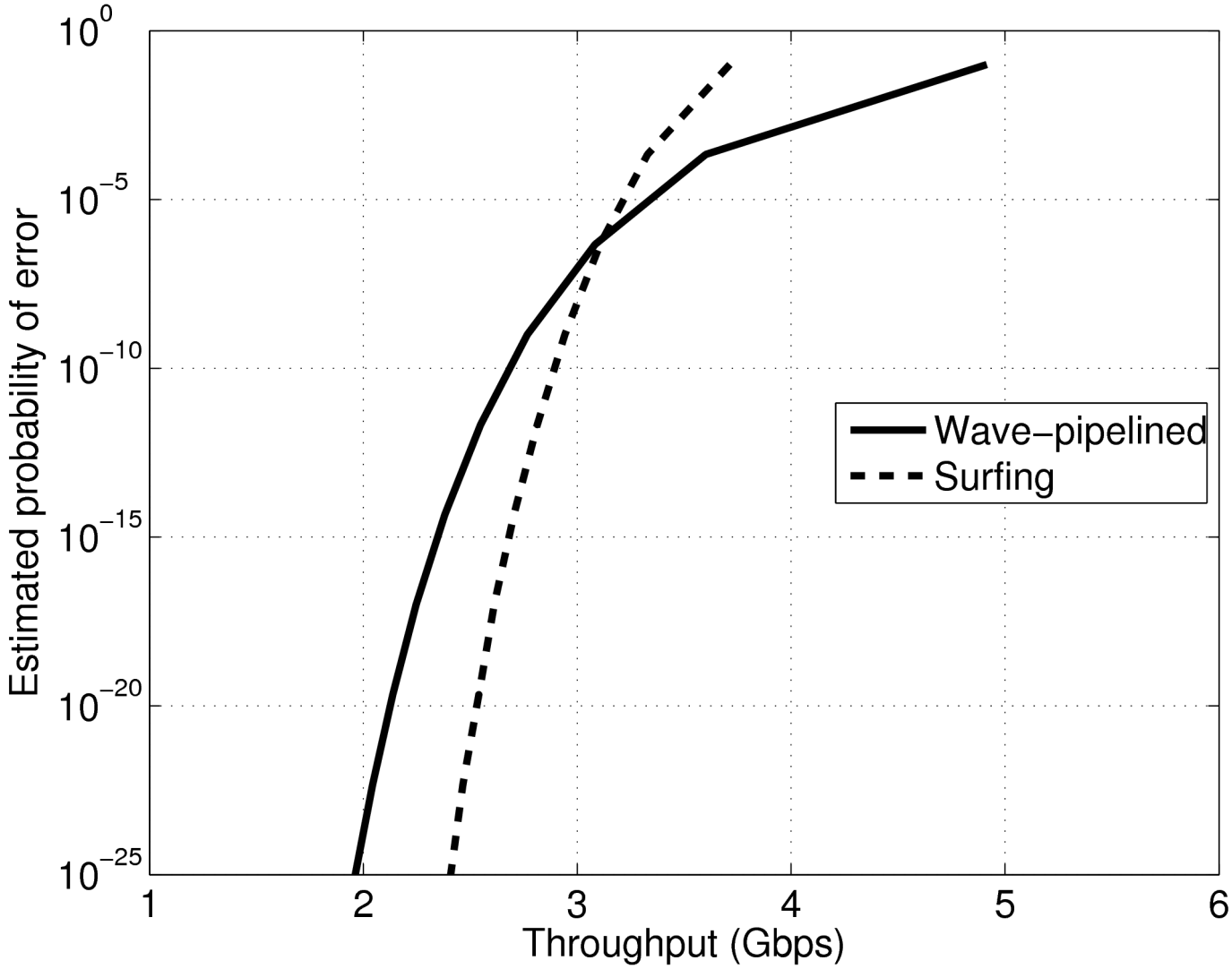
Surfing

- Curves show jitter for varying amounts of fast supply noise
- Bold lines are simulation; dashed lines are extrapolated ($\sqrt{n}$)
- Increases with link length and magnitude of noise
- Skew curves are similar, about half the magnitude

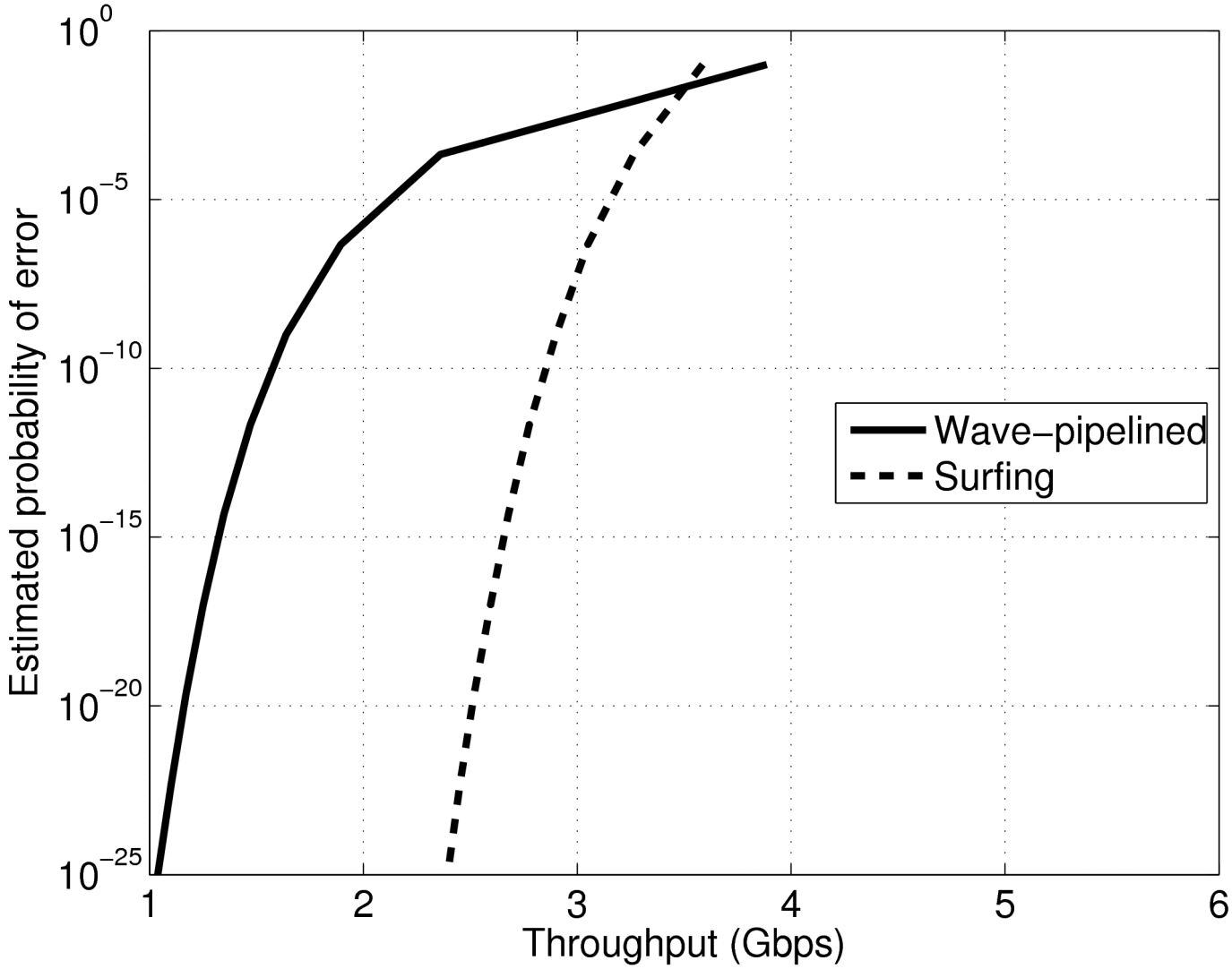
How does this impact reliability?

- Higher jitter or skew: more margin is required
- Jitter and skew are normally distributed, measured from previous simulations
- We can estimate P(E) for a given bit period
  - Need to define maximum tolerable jitter and skew as a function of bit period (not hard)
  - Period is reciprocal of throughput

Reliability estimate – 10 stage link



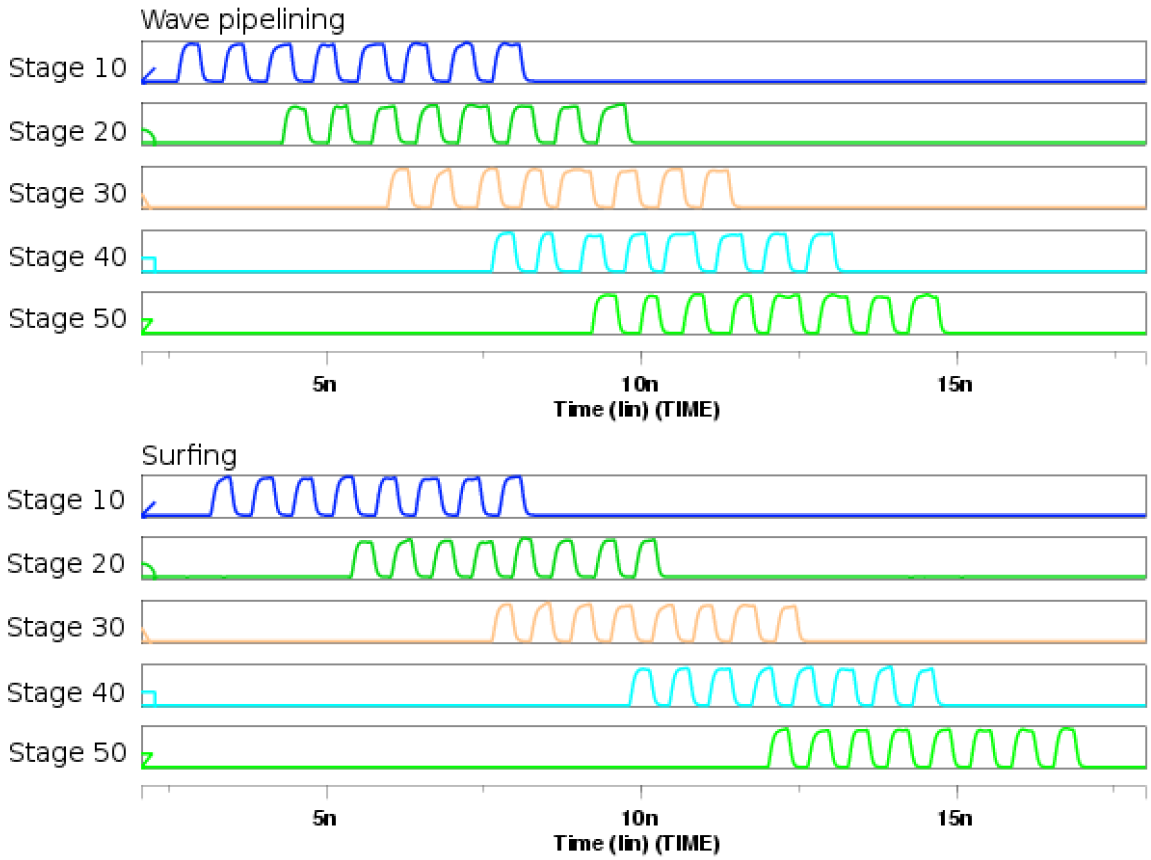
Reliability estimate – 50 stage link



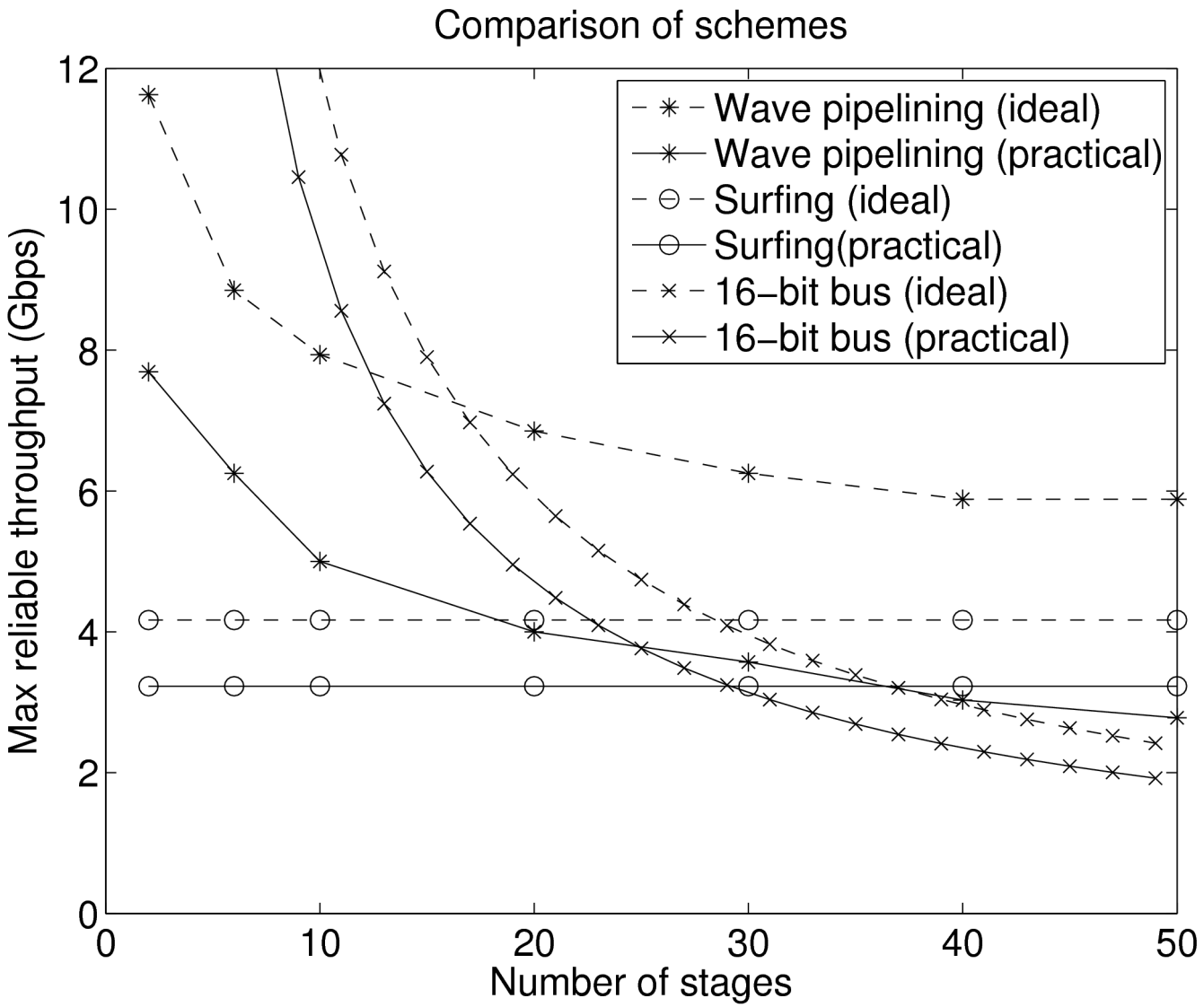
Research questions

- Are serially-interconnected FPGAs worth pursuing?
  - How would it work? √ **Source-synch pipelining**
  - What are the area savings? √ **10-60%**
  - How fast can it go?
  - How much overhead?
- How reliable is interconnect pipelining?
  - Is surfing better than wave pipelining? √ **yes**
  - How well do they work in high-noise environments? √ **OK**

Clock waveforms (50 stage link)



Throughput



Ideal: no noise

Practical: 0.95 V supply with σ=30mV high frequency noise

Latency (serialization penalty)

- Arrival time relative to regular wire

	Stage number:	10	20	30	40	50
First bit	Wave	1.0	1.0	1.0	1.0	1.0
	Surf	1.3	1.3	1.3	1.3	1.3
8th bit	Wave	2.0	1.7	1.6	1.5	1.4
	Surf	2.3	2.0	1.7	1.6	1.6
16th bit	Wave	3.1	2.3	2.0	2.0	1.7
	Surf	4.0	2.6	2.1	2.0	1.8
32nd bit	Wave	5.2	3.5	3.1	2.8	2.3
	Surf	6.8	4.1	3.1	2.8	2.3

Impractical

Impractical

Power

	8-bit transfer		16-bit transfer	
	Energy (fJ)	Normalized	Energy (fJ)	Normalized
Parallel bus	62	**1**	124	**1**
Wave	744	**12**	992	**8**
Wave/LEDR encoded	496	**8**	744	**6**
Surfing	912	**14.7**	1168	**9.4**

- Power comparison assumes 12.5% activity bus
- 100% activity clock causes severe penalty
- Using transition encoding reduces it somewhat

Research questions

- Are serially-interconnected FPGAs worth pursuing?
  - How would it work? √ **Source-synch pipelining**
  - What are the area savings? √ **10-60%**
  - How fast can it go? 3-5Gbps simulated
  - How much overhead? 50 to 100% latency, 6X to 14X power
- How reliable is interconnect pipelining?
  - Is surfing better than wave pipelining? √ **yes**
  - How well do they work in high-noise environments? √ **OK**

Summary of findings

- Are serially-interconnected FPGAs worth pursuing?
  - How would it work? √ **Source-synch pipelining**
  - What are the area savings? √ **10-60%**
  - How fast can it go? **3-5Gbps simulated**
  - How much overhead? **50 to 100% latency, 6X to 14X power**
- How reliable is interconnect pipelining?
  - Is surfing better than wave pipelining? √ **yes**
  - How well do they work in high-noise environments? √ **OK**

Conclusions

- Are serially-interconnected FPGAs worth pursuing?
  - Doubtful, due to high latency and power penalties
  - But very high throughput per area achieved in this thesis may find specialized application
- How reliable is interconnect pipelining?
  - Difficult to say without better supply noise models
  - Surfing is better than wave pipelining
  - Both of them show vulnerability which needs to be considered

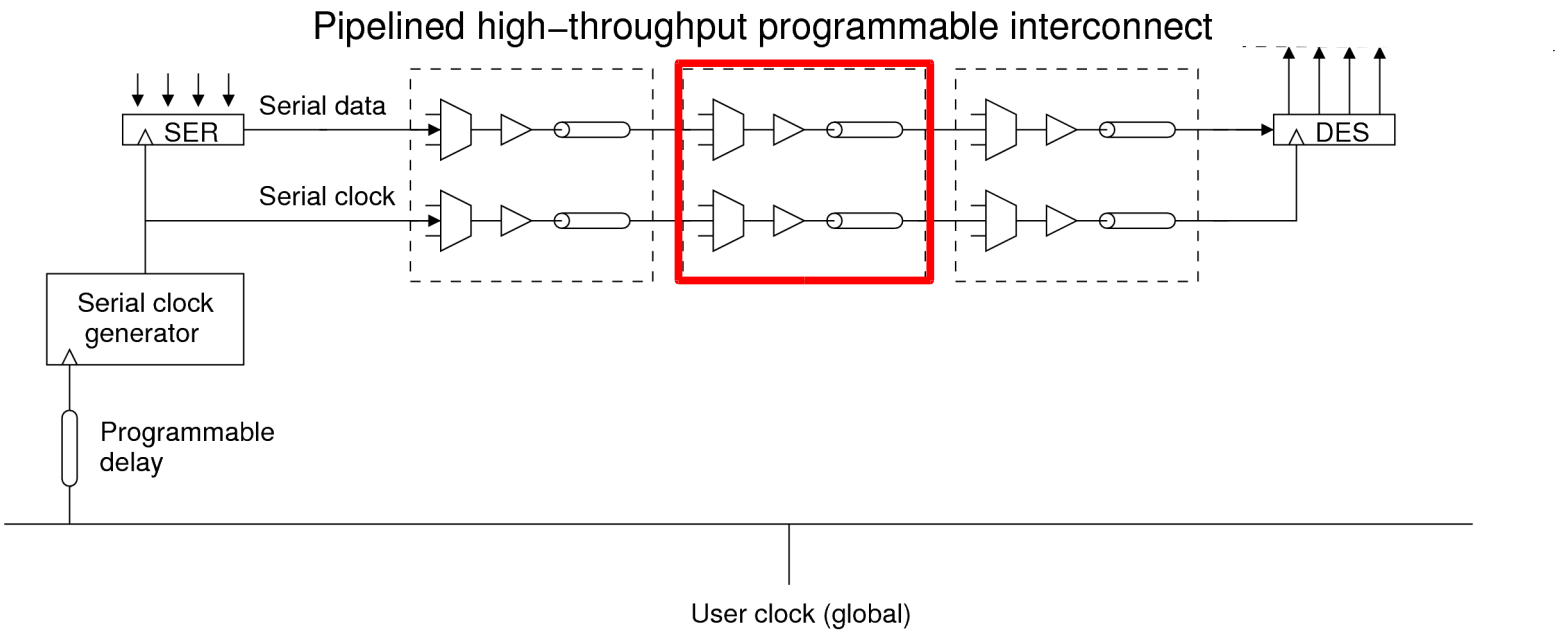
Future work

- Noise and reliability
  - Better supply noise models; silicon implementation
- Architectural exploration
  - Verify area estimates; find applications (serial logic?)
- Low-power circuit design
- Jitter/skew attenuation in wave pipelining
  - Insert FIFOs every 10 stages to resynchronize; would improve throughput considerably

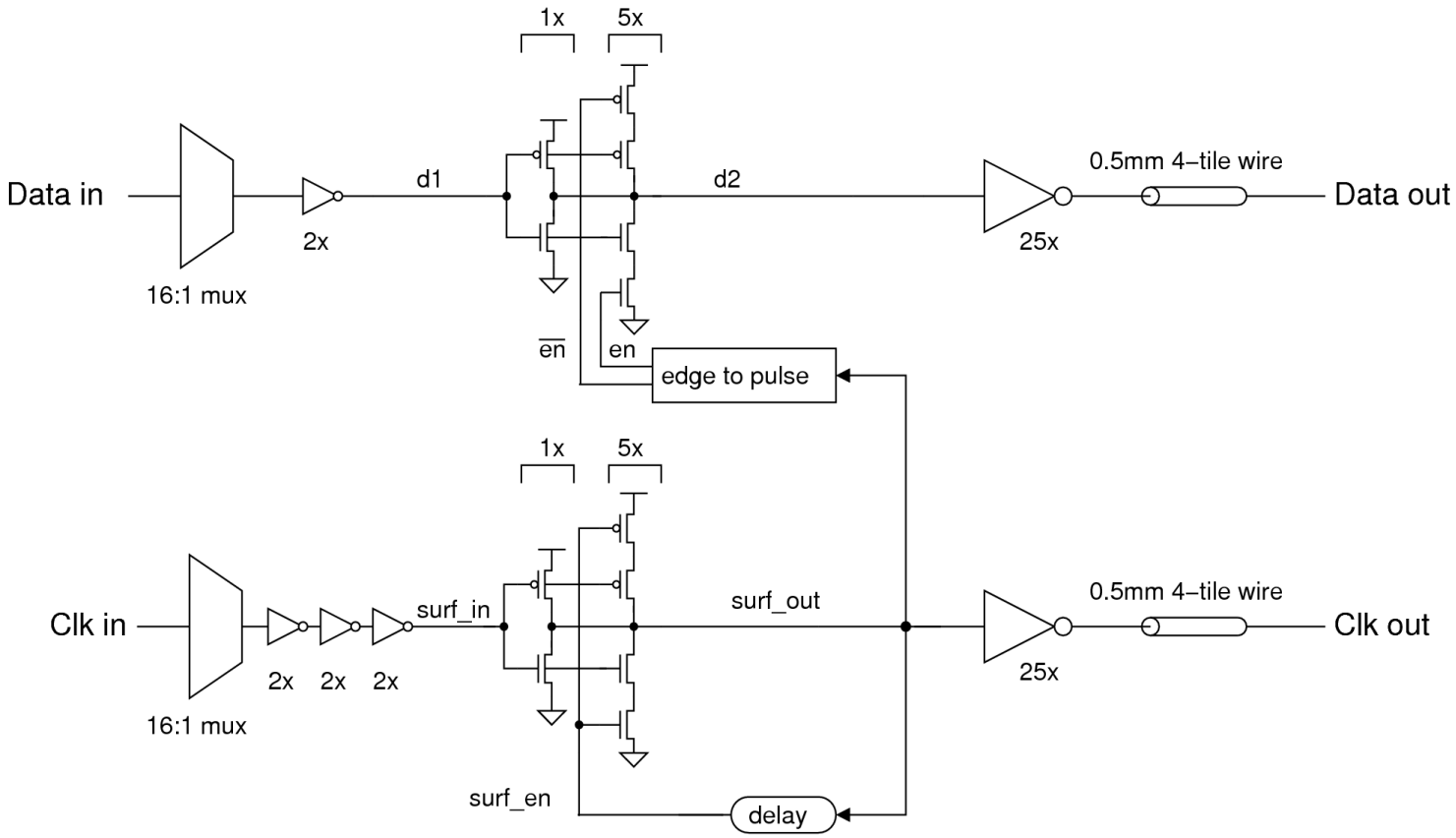
Thank you

Pipelining schemes

- Custom modifications required to support wave pipelining and surfing

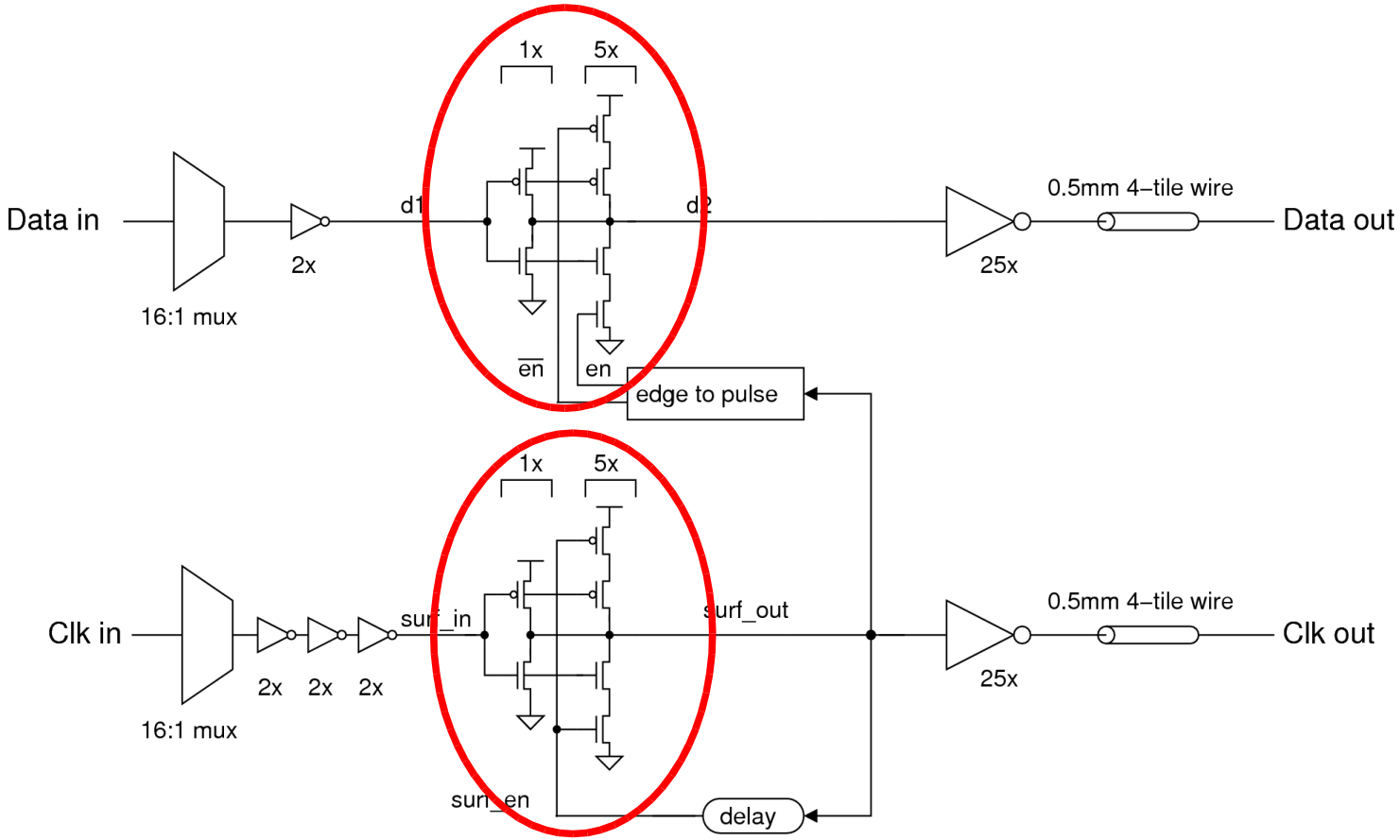


Surfing circuit design

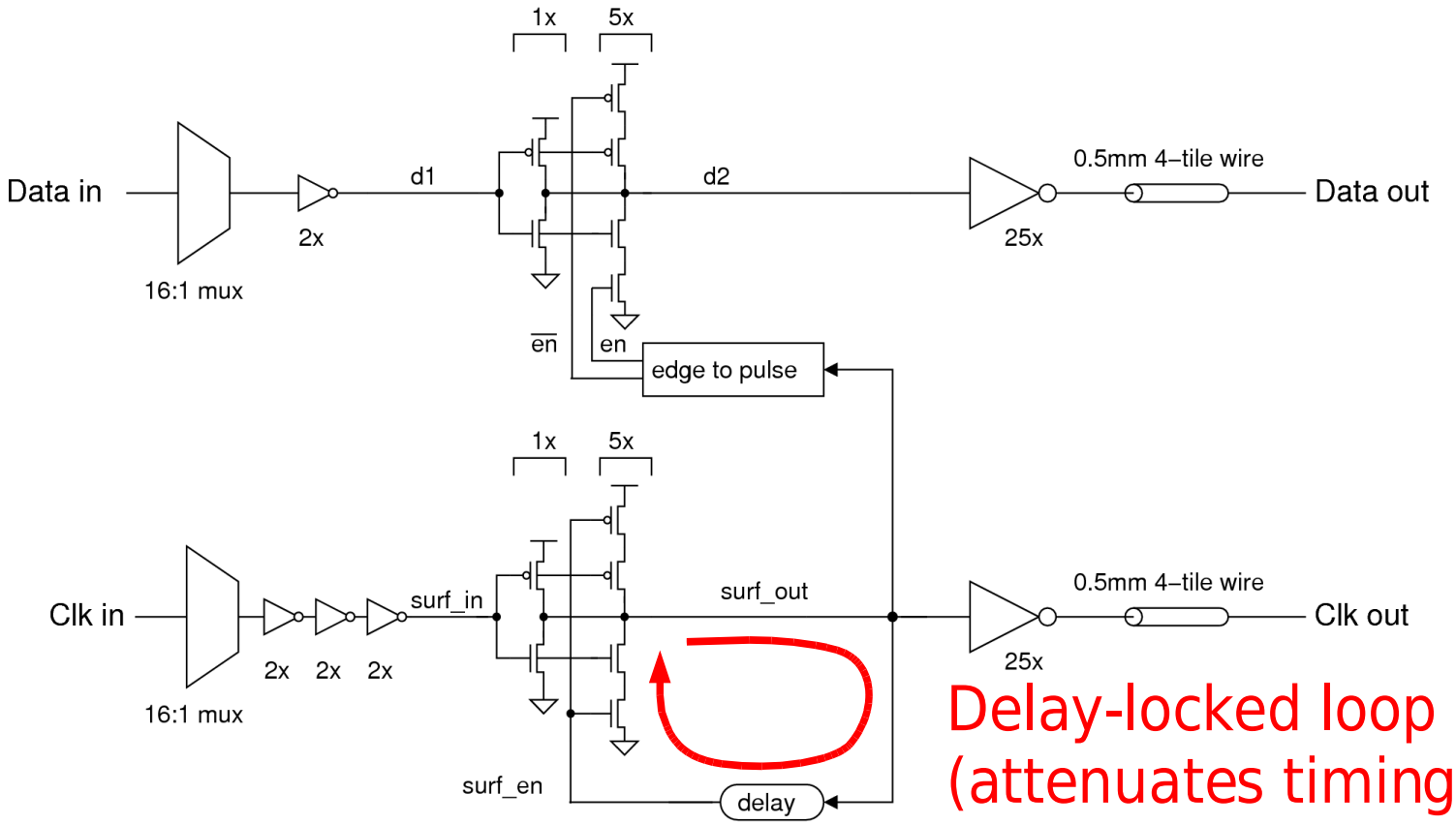


Surfing circuit design

Variable-strength buffers

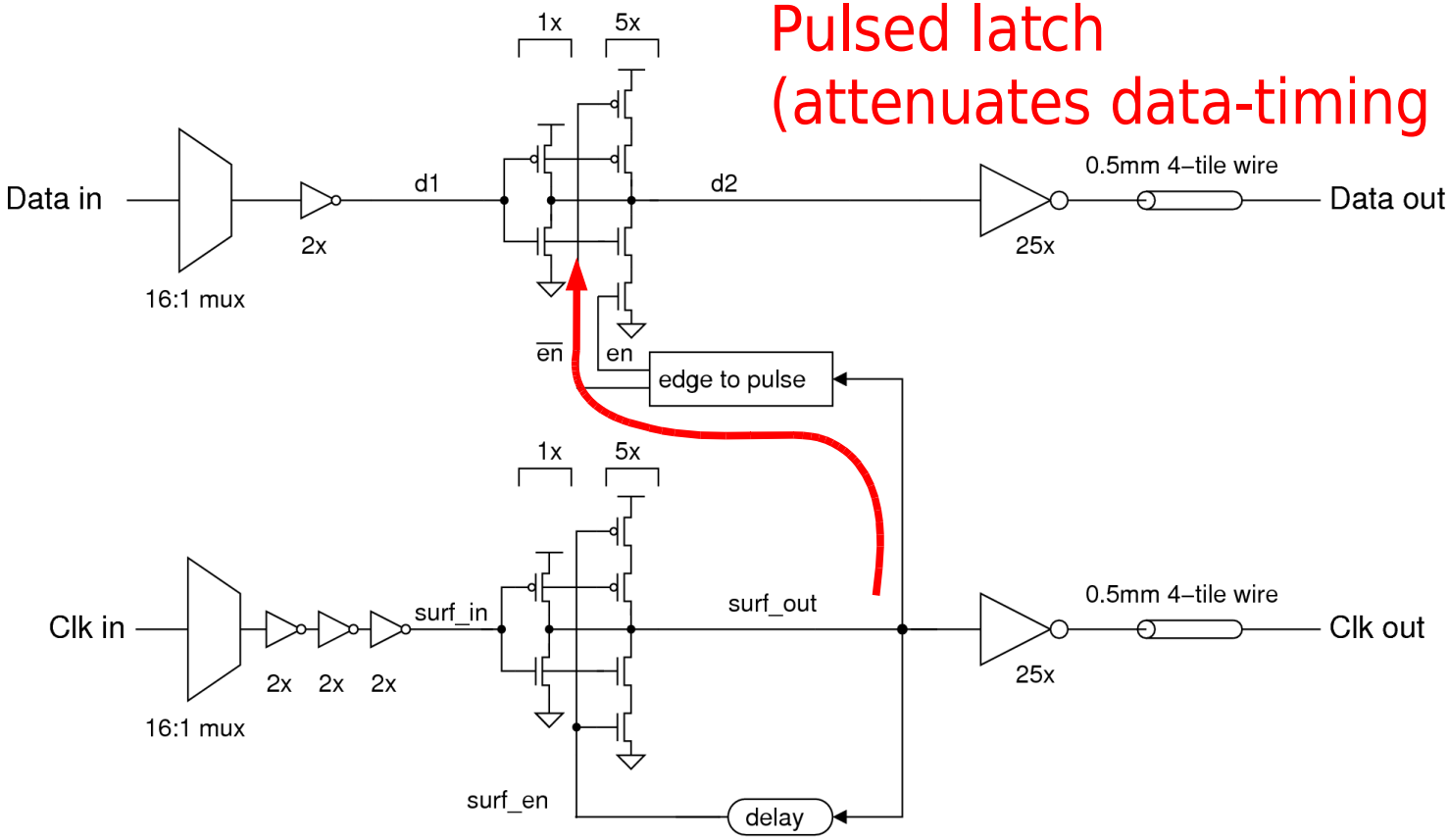


Surfing circuit design



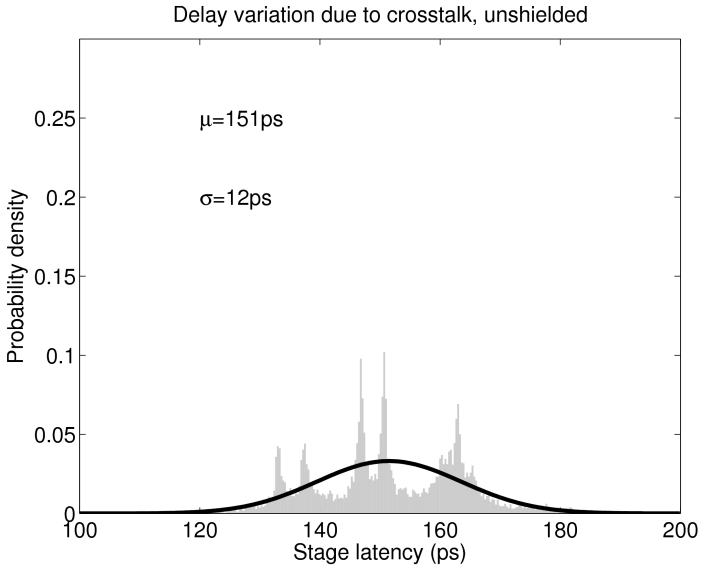
Delay-locked loop
(attenuates timing jitter)

Surfing circuit design

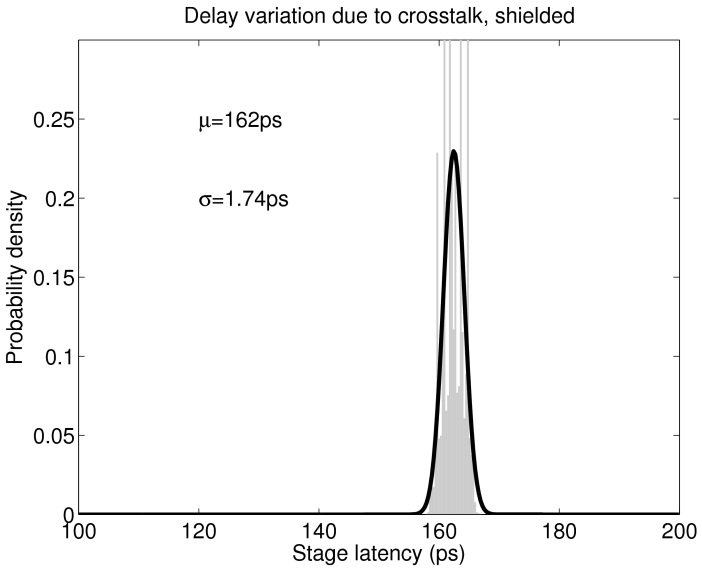


Timing uncertainty: crosstalk

- Delay variation due to random aggressor data



Unshielded



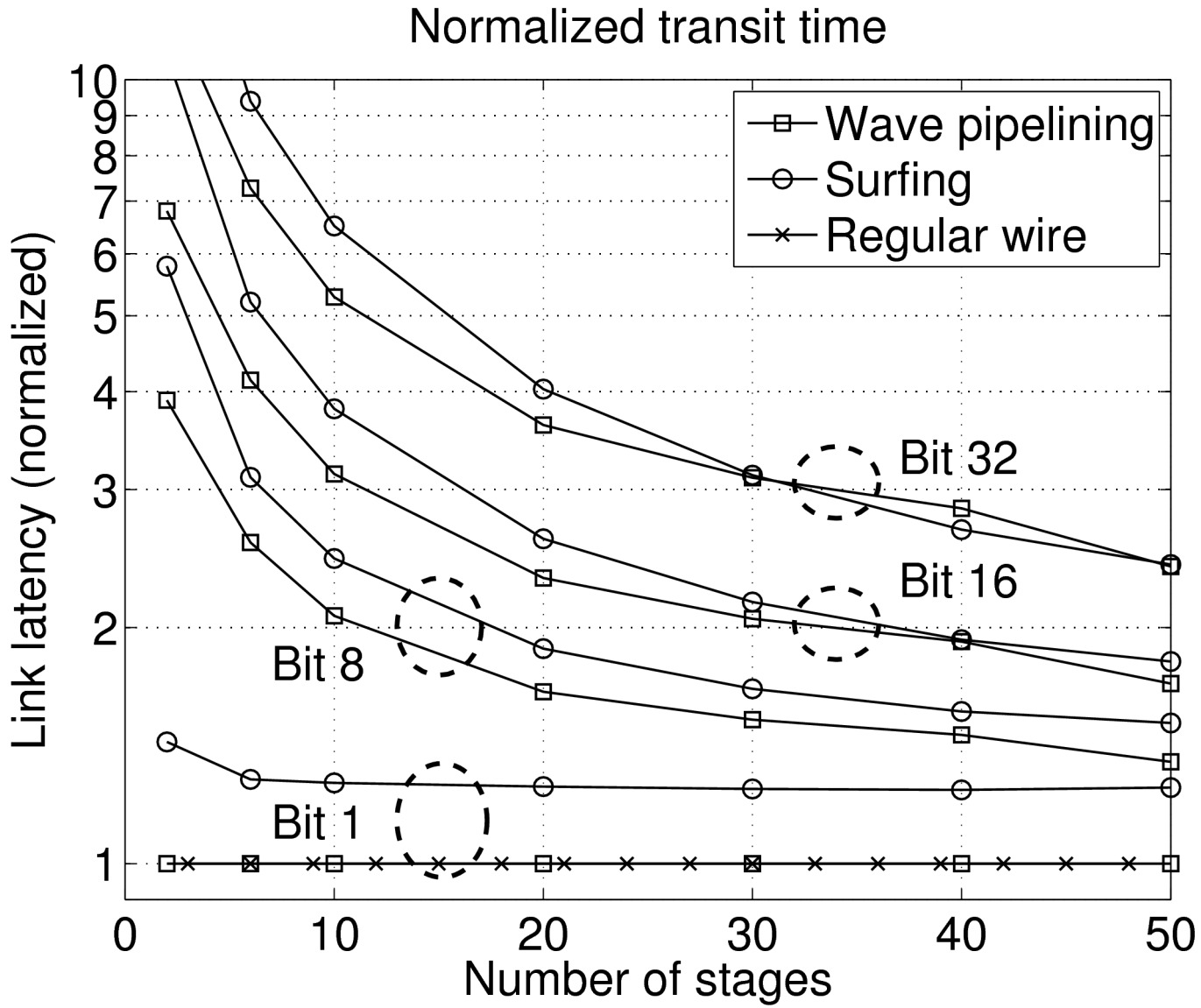
Shielded

- Shielding is necessary, and effective

Why source-synchronous?

- Robust with respect to variation
- Power scales with amount of serial signaling
- Alternative: global high speed clock?
  - Large design complexity, power overhead for 2-3Ghz clock
  - Difficult to implement clock gating
- Alternative: asynchronous?
  - Delay, power overheads due to handshaking
  - More drastic modification

Latency (serialization penalty)



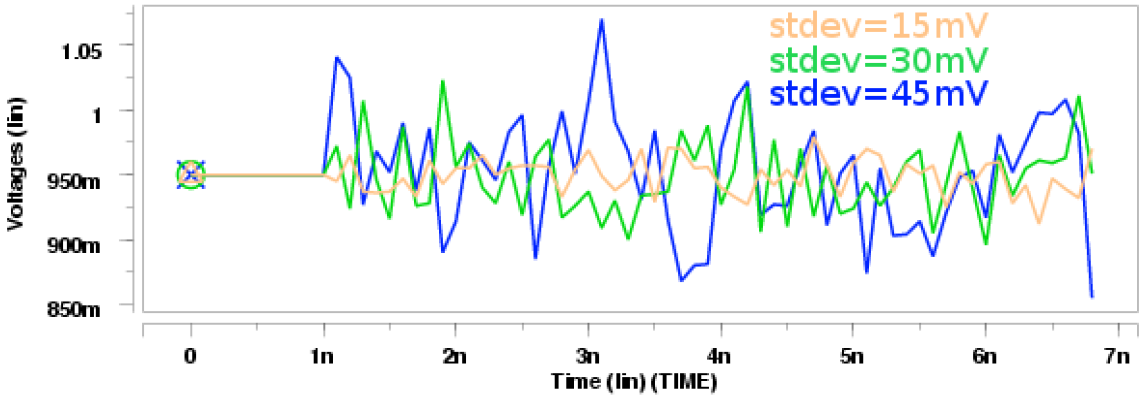
Regular wire:
123 ps per stage

Wave pipelining:
123 ps per stage

Surfing:
156 ps per stage

Supply noise waveforms

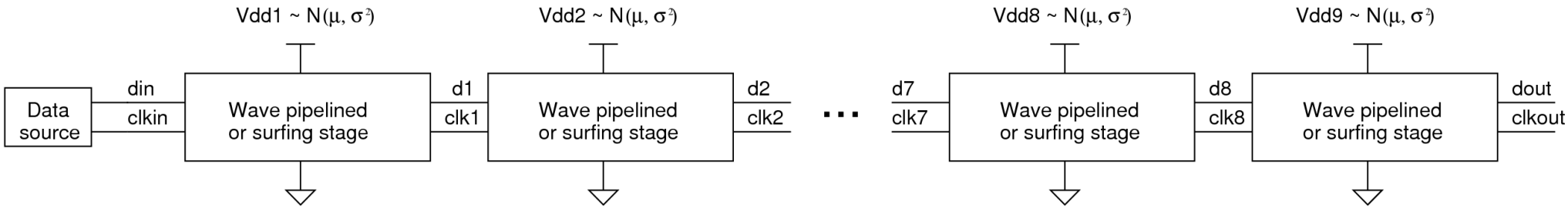
- Characterize by standard deviation (σ)



  - Memoryless process, new value every 100ps for high impact on cycle-to-cycle delay
  - Vary mean (DC value, μ) independently; nominal 1.0V
  - σ of 45mv means 3σ is about $\pm$ 10% (pessimistic)

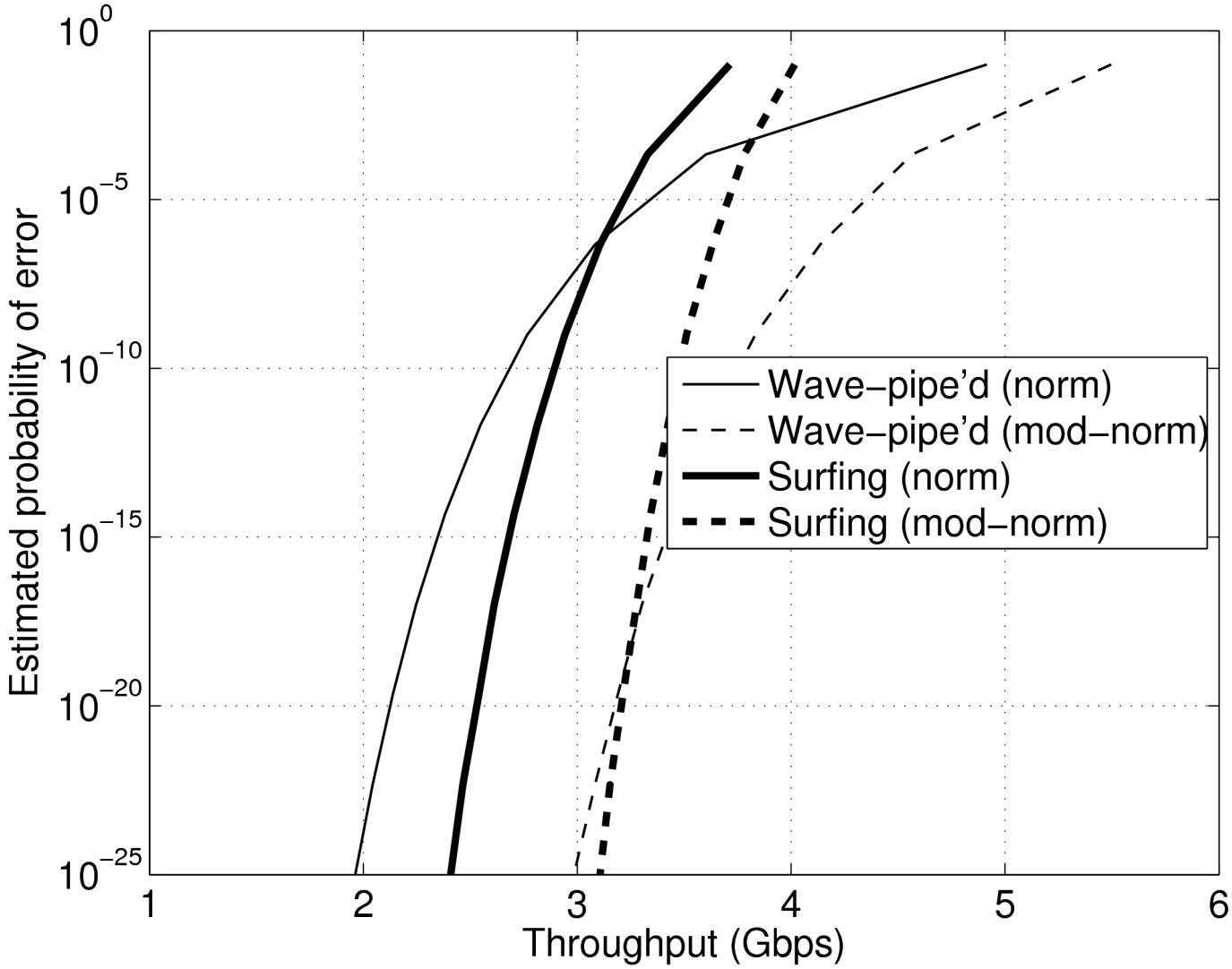
Noise response: simulation setup

- Apply random supply noise to a multi-stage link; measure jitter and skew at each state



- Expect lower jitter and skew with surfing

Reliability estimate – 10 stage link



Reliability estimate – 50 stage link