Advances in Asynchronous and GALS Design:

an SoC for HPC Perspective

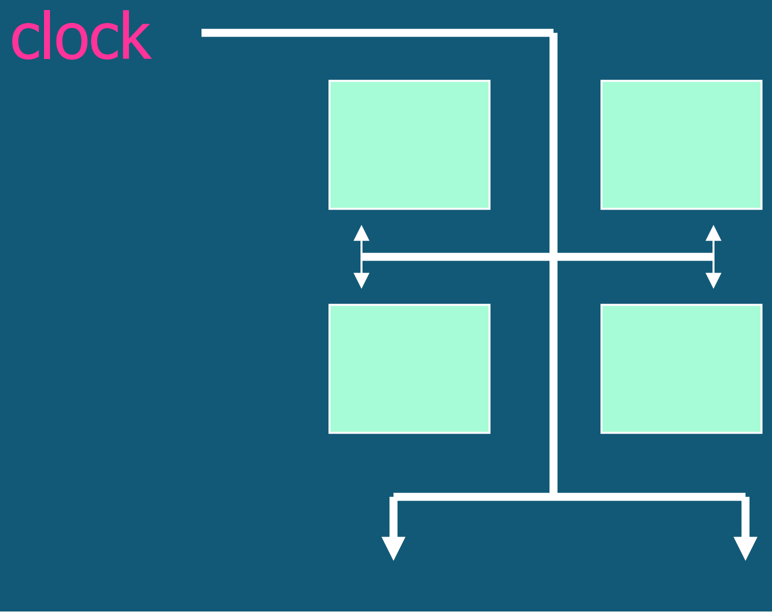
Prof. Steven M. Nowick

nowick@cs.columbia.edu

Department of Computer Science (and Elect. Eng.)
Columbia University
New York, NY, USA

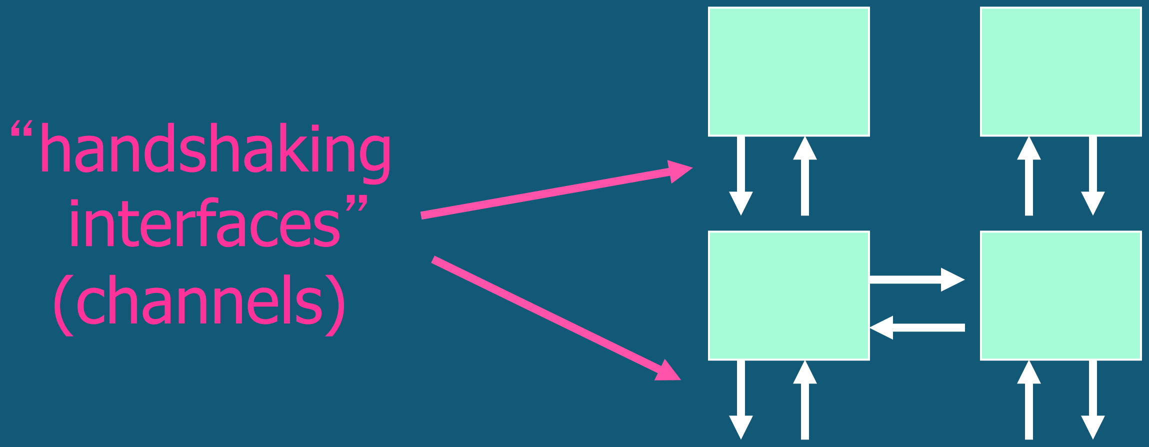
Introduction

- Synchronous vs. Asynchronous Systems?
  - Synchronous Systems: use a *global clock*
    - entire system operates *at fixed-rate*
    - uses *"centralized control"*



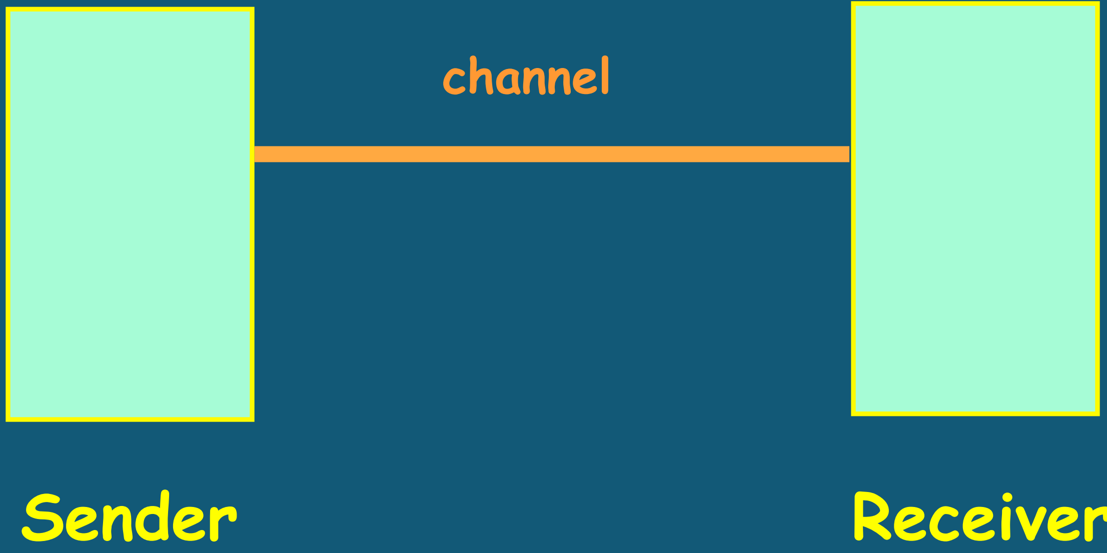
Introduction (cont.)

- Synchronous vs. Asynchronous Systems? (cont.)
  - Asynchronous Systems: *no global clock*
    - components can operate at *varying rates*
    - *communicate locally* via *"handshaking"*
    - uses *"distributed control"*



Overview: Asynchronous Communication

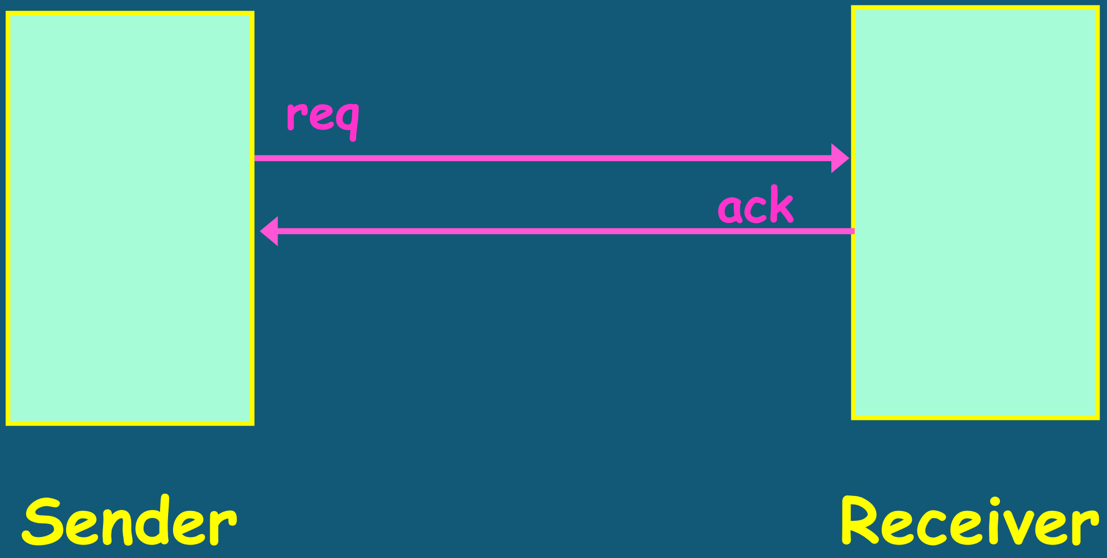
Components usually communicate & synchronize on channels



synchronization: without data

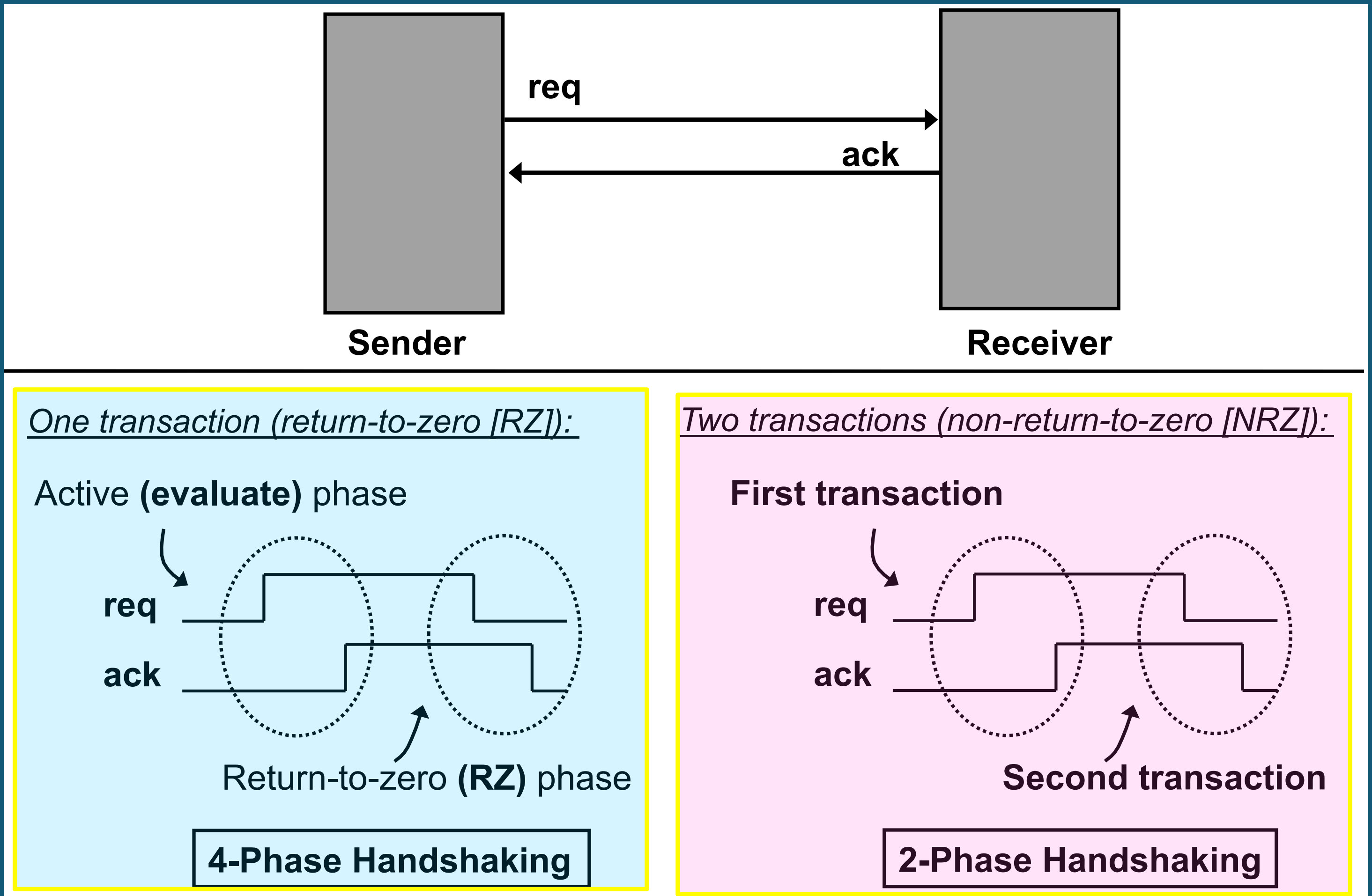
Overview: Signalling Protocols

Communication channel: usually **instantiated as 2 wires**



synchronization: without data

Overview: Signalling Protocols



One transaction (return-to-zero [RZ]):

Active **(evaluate)** phase

Return-to-zero **(RZ)** phase

4-Phase Handshaking

Two transactions (non-return-to-zero [NRZ]):

First transaction

Second transaction

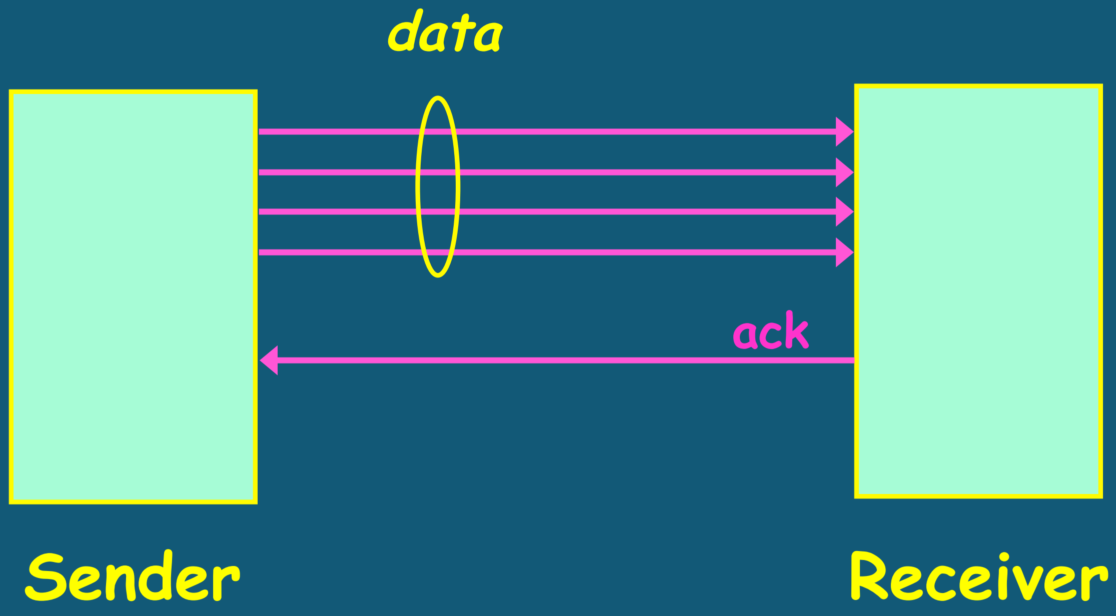
2-Phase Handshaking

synchronization: without data

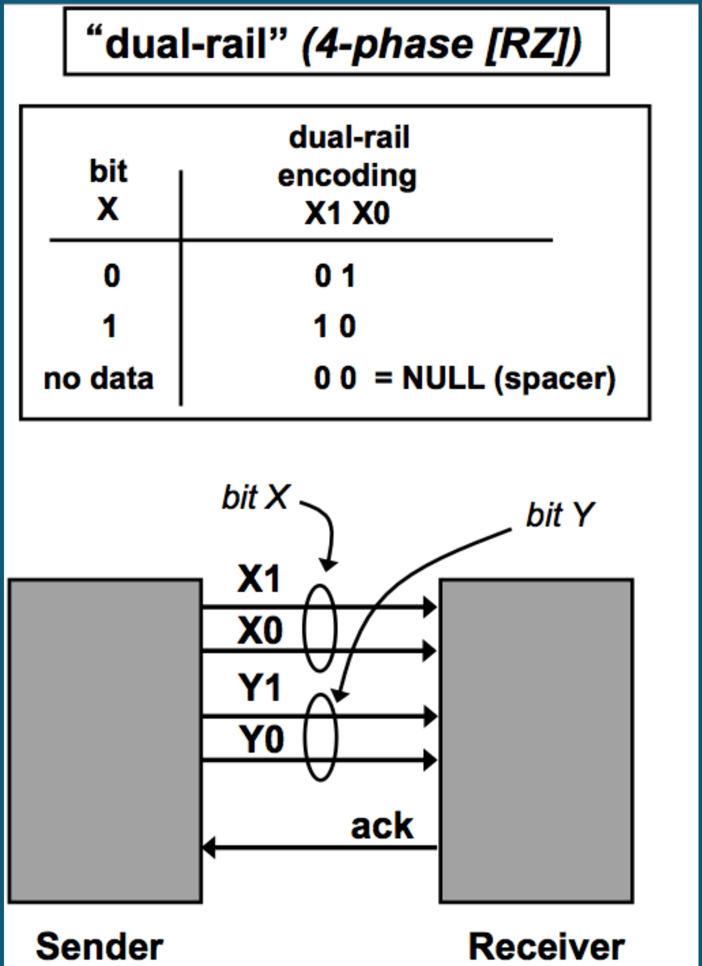
Overview: How to Communicate Data?

Data channel: replace "req" by (encoded) data bits

- … still use 2-phase or 4-phase protocol

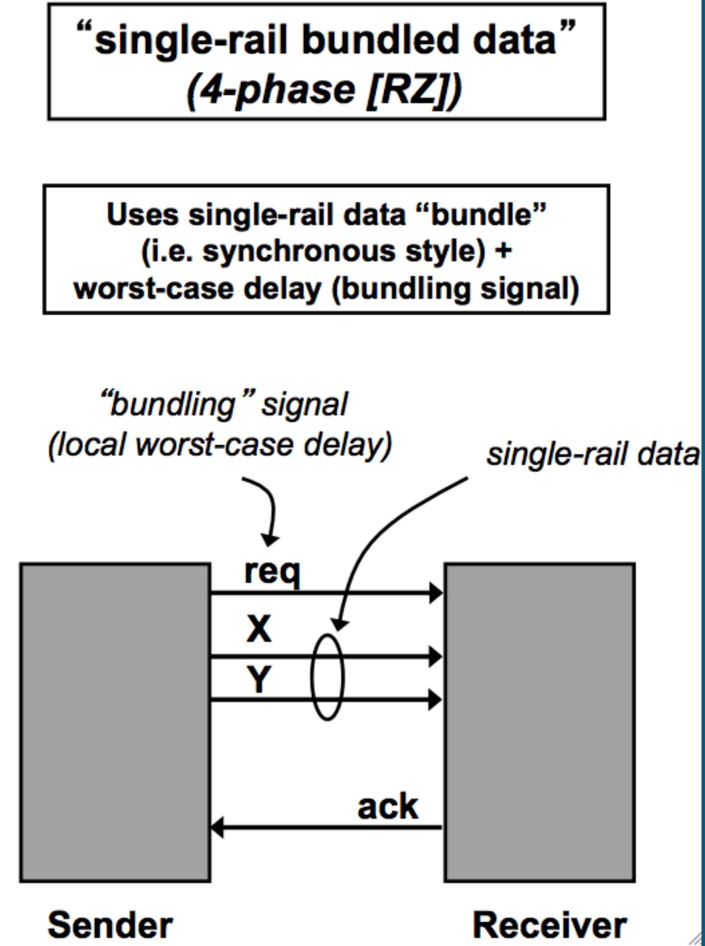


Overview: How to Encode Data?



"dual-rail" *(4-phase [RZ])*

bit X	dual-rail encoding X1 X0
0	0 1
1	1 0
no data	0 0 = NULL (spacer)



"single-rail bundled data" *(4-phase [RZ])*

Uses single-rail data "bundle" (i.e. synchronous style) + worst-case delay (bundling signal)

Dual-rail = delay-insensitive (DI) codes
(alternatives: 1-of-4, m-of-n, …)

Single-rail "bundled data"

Asynchronous Design: Potential Benefits

Lower Power

- no clock
  - ➔ components inherently use dynamic power only "on demand"
  - ➔ *no global clock distribution*
  - ➔ effectively provides automatic clock gating at arbitrary granularity

Robustness, Scalability, Modularity: "Lego-like" construction

- no global timing: plug-and-play design
  - ➔ "mix-and-match" variable-speed components, different block sizes
  - ➔ supports dynamic voltage scaling
- modular design style ➔ "object-oriented"

Higher Performance *(… sometimes)*

- not limited to "worst-case" clock rate

"Demand- (Data-) Driven" Operation

- instantaneous wake-up from standby mode

Potential Targets

Large variety of asynchronous design styles

- Address different points in "design-space" spectrum…
  - extreme timing-robustness:
    - supports unknown transmission times, arbitrary inter-bit skews
    - PVT variation tolerant: providing near "delay-insensitive (DI)" operation
  - ultra-low power, energy:
    - "on-demand" operation, instant wakeup
    - sub-/near-threshold benefits: J. Rabaey, K. Roy, S. Nowick/M.Seok
  - ease-of-design/moderate performance/low EMI (electro-magnetic interference)
    - e.g. goal at Philips Semiconductors
  - very high-speed: asynchronous pipelined systems
    - … comparable throughput to high-end synchronous design
    - with added benefits: lower system latency, support variable I/O rates
  - modular heterogeneous systems: integrate clock domains via async
    - "GALS-style" (globally-async/locally-sync)
  - use in emerging technologies: QCA, CNT, wireless, photonic/digital, etc.

A Brief History...:

Phase #1: Early Years (1950's-1960's)

- Leading processors: Illiac, Illiac II (U. of Illinois), Atlas, MU-5 (U. of Manchester)
- Macromodules Project: plug-and-play design (Washington U., Wes Clark/C. Molnar)
- Commercial graphics/flight simulation systems: LDS-1 (Evans & Sutherland, C. Seitz)
- Basic theory, controllers: Unger, McCluskey, Muller

Phase #2: The Quiet Years [VLSI epoch] (1970's-mid 1980's)

- VLSI success: leads to synchronous domination and major advances

Phase #3: Coming of Age (late 1980's to 2000)

- Re-inventing the field:
  - correct new methodologies, controllers, high-speed pipelines, basic CAD tools
  - initial industrial uptake: Philips Semiconductors products, Intel/IBM projects
  - First microprocessors: Caltech, Manchester Amulet [ARM]

Phase #4: The Modern Era (early 2000's-present)

- Leading applications, commercialization, tool development, demonstrators

Asynchronous Design: Recent Industrial Developments

1. Philips Semiconductors: low-/moderate-speed embedded systems

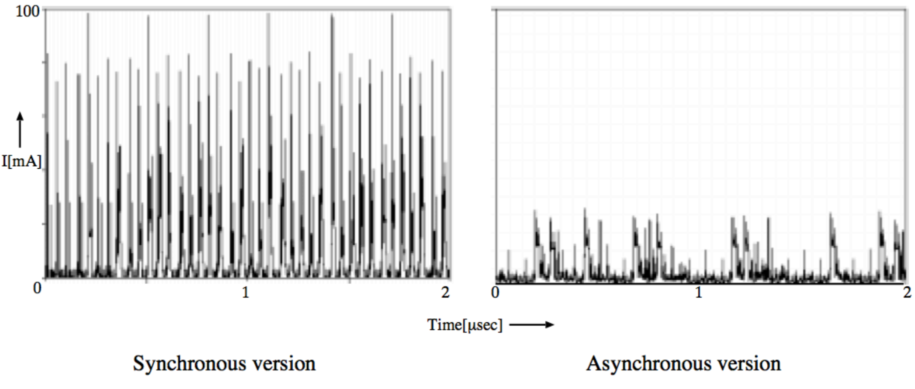
- Wide commercial use: 700 million async chips *(mostly 80c51 microcontrollers)*
  - consumer electronics: *pagers, cell phones, smart cards, digital passports, automotive*
  - commercial releases: 1990's-2000's
- Benefits (vs. sync):
  - *3-4x* lower power (and lower energy consumption/op)
  - 5x lower peak currents
  - much lower "electromagnetic interference" (EMI) – no shielding of analog components
  - Correct operation over wide supply voltage range
  - instant startup from stand-by mode (no PLL's)
- Complete commercial CAD tool flow: synthesis, testing, design-space exploration
  - "Tangram": Philips (late 1980's to early 2000's)
  - "Haste": Handshake Solutions (incubated spinoff, early to late 2000's)

Asynchronous Design: Recent Industrial Developments

1. Philips Semiconductors (cont.)

- Synthesis strategy: *syntax-directed compilation*
  - *starting point: concurrent HDL* (Tangram, Haste)
  - *2-step synthesis:*
    - front-end: HDL spec => intermediate netlist of concurrent components
    - back-end: each component => standard cell (… then physical design)
      - Integrated flow with Synopsys/Cadence/Magma tools
  - *+: fast, 'transparent', easy-to-use*
  - *-: few optimizations, low/moderate-performance only*

Asynchronous 80c51:
5x lower current peaks [Philips, 2000]*



Synchronous version

Asynchronous version

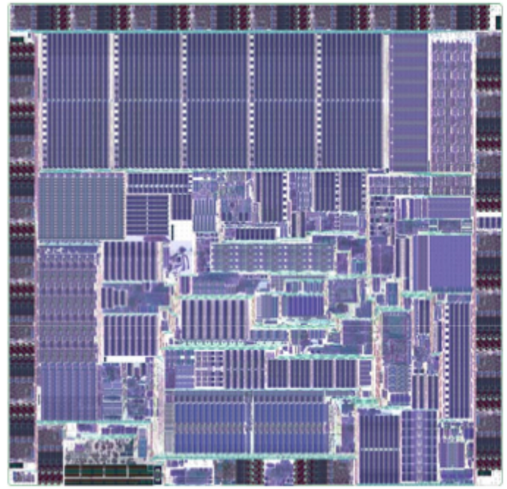
Figure 3. Current peaks of 80C51 micro-controller

**J. Kessels, T. Kramer, G. den Besten, A Peeters, and V. Timm,*
"Applying Asynchronous Circuits in Contactless Smart Cards," IEEE Async-Symposium (2000)

Asynchronous Design: Recent Industrial Developments

2. Fulcrum Microsystems/Intel: high-speed Ethernet switch chips

- Async start-up out of Caltech ➔ now Intel's Switch & Router Division (SRD) (2011)
- Target: low system latency, extreme functional flexibility
- Alta Chip: Intel's FM5000-6000 Series (~2013 release)
  - 72-port 10G Ethernet switch/router
  - Very low cut-through latency: 400-600ns
  - 90% asynchronous → external synchronous interfaces
  - 1.2 billion transistors: largest async chip ever manufactured (at release time)
  - > 1 GHz asynchronous performance (65 nm TSMC process)
  - CAD flow: semi-automated, including spec language (CAST)



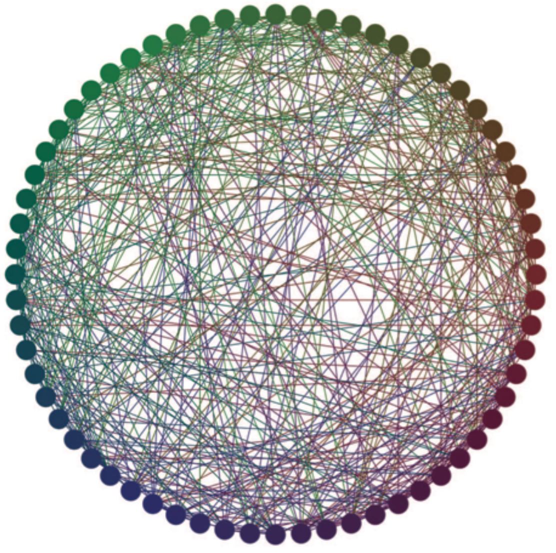
**M. Davies, A. Lines, J. Dama, A. Gravel, R. Southworth, G. Dimou and P. Beerel, "A 72-Port 10G Ethernet Switch/Router Using Quasi-Delay-Insensitive Asynchronous Design," IEEE Async-Symposium (2000)*

Asynchronous Design: Recent Industrial Developments

3. Neuromorphic Chips: IBM's "TrueNorth" (Aug. 2014)

- Developed out of DARPA's SyNAPSE Program
- Massively-parallel, fine-grained neuromorphic chip
  - Fully-asynchronous chip! → neuronal computation (bundled data) + interconnect (DI)
  - IBM's largest chip ever: 5.4 billion transistors
  - Models 1 million neurons/256 million synapses ➔ contains 4096 neurosynaptic cores
    - *... MANY-CORE SYSTEM!*
  - Extreme low energy: 70 mW for real-time operation → 46 billion synaptic ops/sec/W
  - Asynchronous motivation: extreme scale, high connectivity, power requirements, tolerance to variability

Example network topology:
showing only 64 cores (out of 4096)
[IBM, 2014]*



**P.A. Merolla, J.V. Arthur, et al., "A Million Spiking-Neuron Integrated Circuit with a Scalable Communication Network and Interface," Science, vol. 345, pp. 668-673 (Aug. 2014) [COVER STORY]*

Asynchronous Design: Recent Industrial Developments

3. Neuromorphic Chips: Other Recent Asynchronous Designs

a. U. of Manchester (UK): SpiNNaker Project, ~2005-present (S. Furber et al.)

b. Stanford: "Neurogrid" (Brains in Silicon) (K. Boahen et al.)

- Scientific American (May 2005) – cover story
- Proceedings of the IEEE (May 2014)

➔ Each uses robust async NoC's to integrate massively-parallel many-core system

Designing a Low-Power and Low-Latency NoC Switch Architecture for Cost-Effective GALS Multicore Systems

[in 2013 ACM/IEEE Design, Automation and Test in Europe Conference (DATE-13)]

Alberto Ghiribaldi
ENDIF
University of Ferrara
Ferrara, Italy

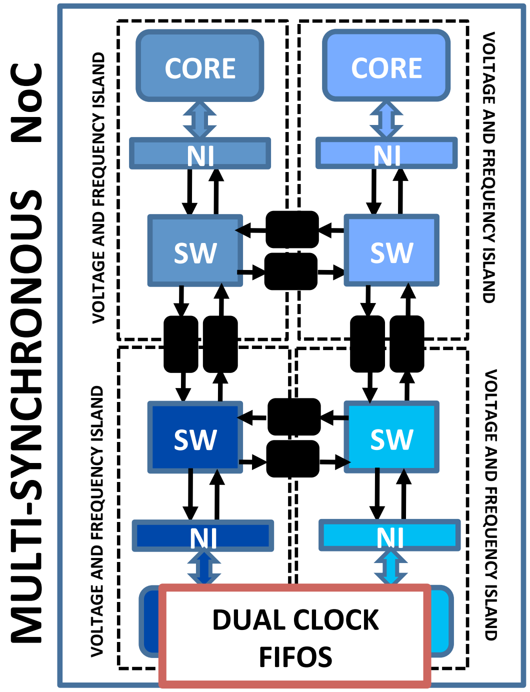
Davide Bertozzi
ENDIF
University of Ferrara
Ferrara, Italy

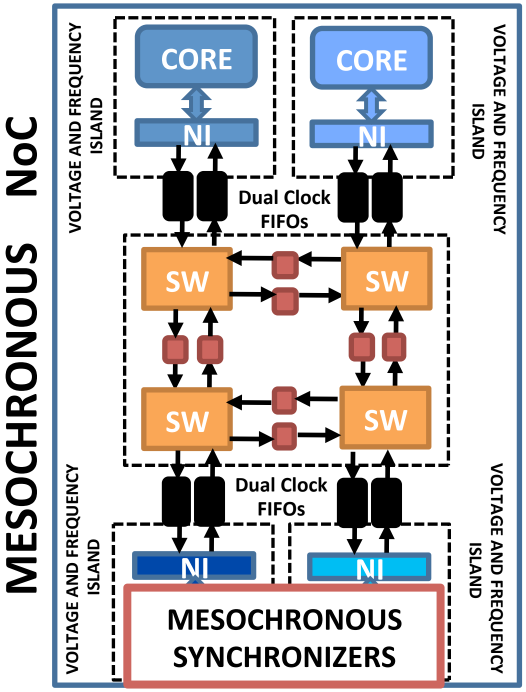
Steven M. Nowick
Dept. of Computer Science
Columbia University
New York, NY, USA

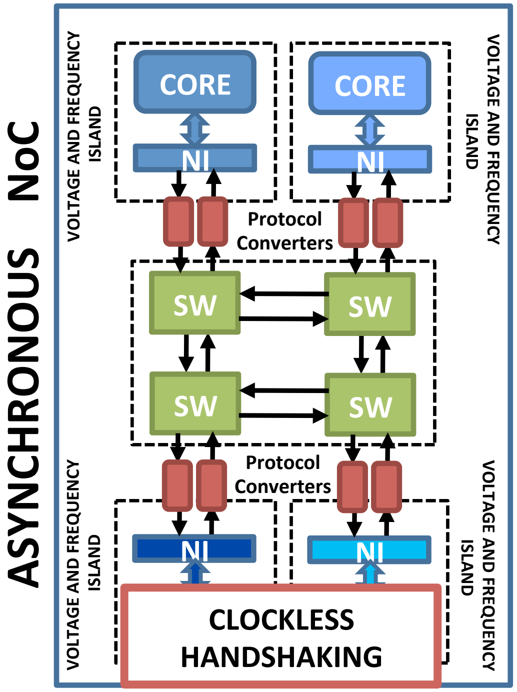
State of the Art

MPSoCs increasingly structured as multiple voltage/frequency islands, making their system interconnect challenging

Examples of heterogeneous MPSoCs with multiple clock domains:


MULTI-SYNCHRONOUS NoC
VOLTAGE AND FREQUENCY ISLAND
CORE
NI
SW
CORE
NI
SW
VOLTAGE AND FREQUENCY ISLAND
VOLTAGE AND FREQUENCY ISLAND
SW
NI
SW
NI
VOLTAGE AND FREQUENCY ISLAND
DUAL CLOCK FIFOS


MESOCHRONOUS NoC
VOLTAGE AND FREQUENCY ISLAND
CORE
NI
CORE
NI
VOLTAGE AND FREQUENCY ISLAND
Dual Clock FIFOs
SW
SW
SW
SW
Dual Clock FIFOs
VOLTAGE AND FREQUENCY ISLAND
NI
NI
VOLTAGE AND FREQUENCY ISLAND
MESOCHRONOUS SYNCHRONIZERS


ASYNCHRONOUS NoC
VOLTAGE AND FREQUENCY ISLAND
CORE
NI
CORE
NI
VOLTAGE AND FREQUENCY ISLAND
Protocol Converters
SW
SW
SW
SW
Protocol Converters
VOLTAGE AND FREQUENCY ISLAND
NI
NI
VOLTAGE AND FREQUENCY ISLAND
CLOCKLESS HANDSHAKING

Multi-synchronous NoC's

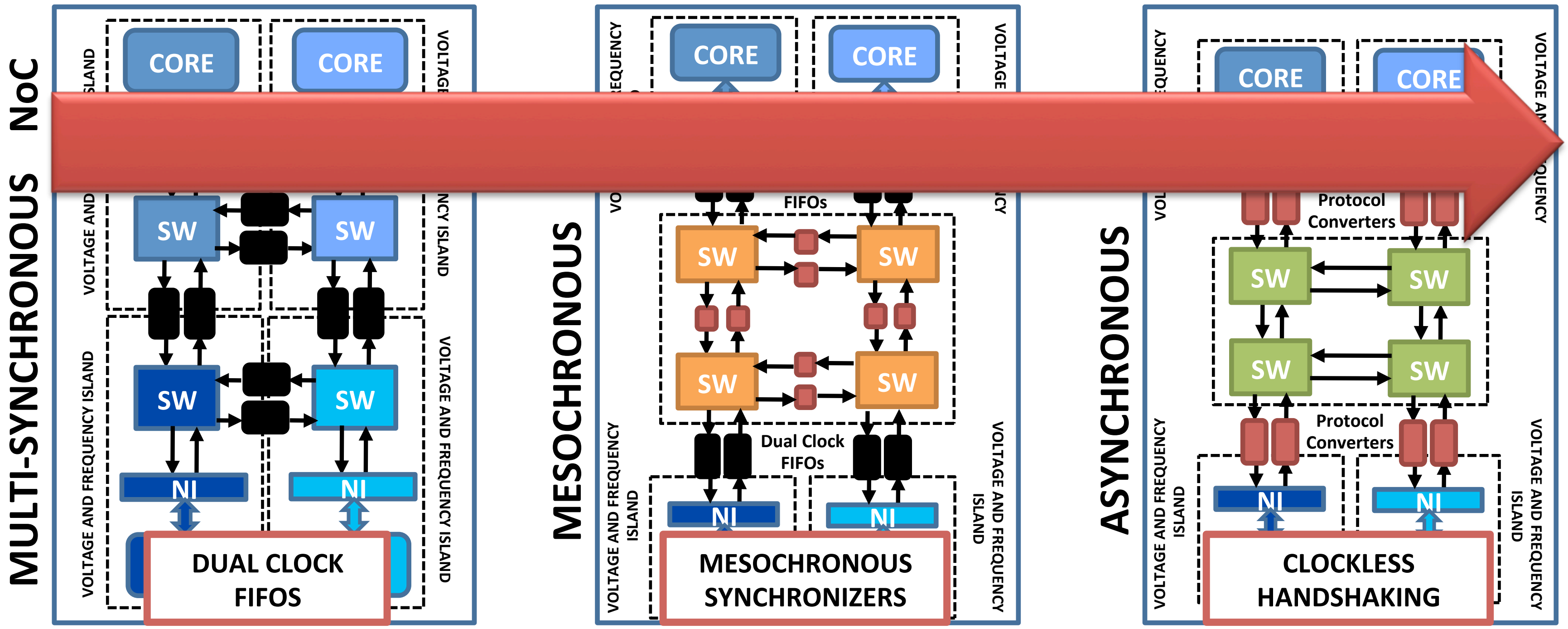
GALS NoC

NoC's: alternative synchronization strategies

State of the Art

MPSoCs increasingly structured as multiple voltage/frequency islands, making their system interconnect challenging

Examples of heterogeneous MPSoCs with multiple clock domains:

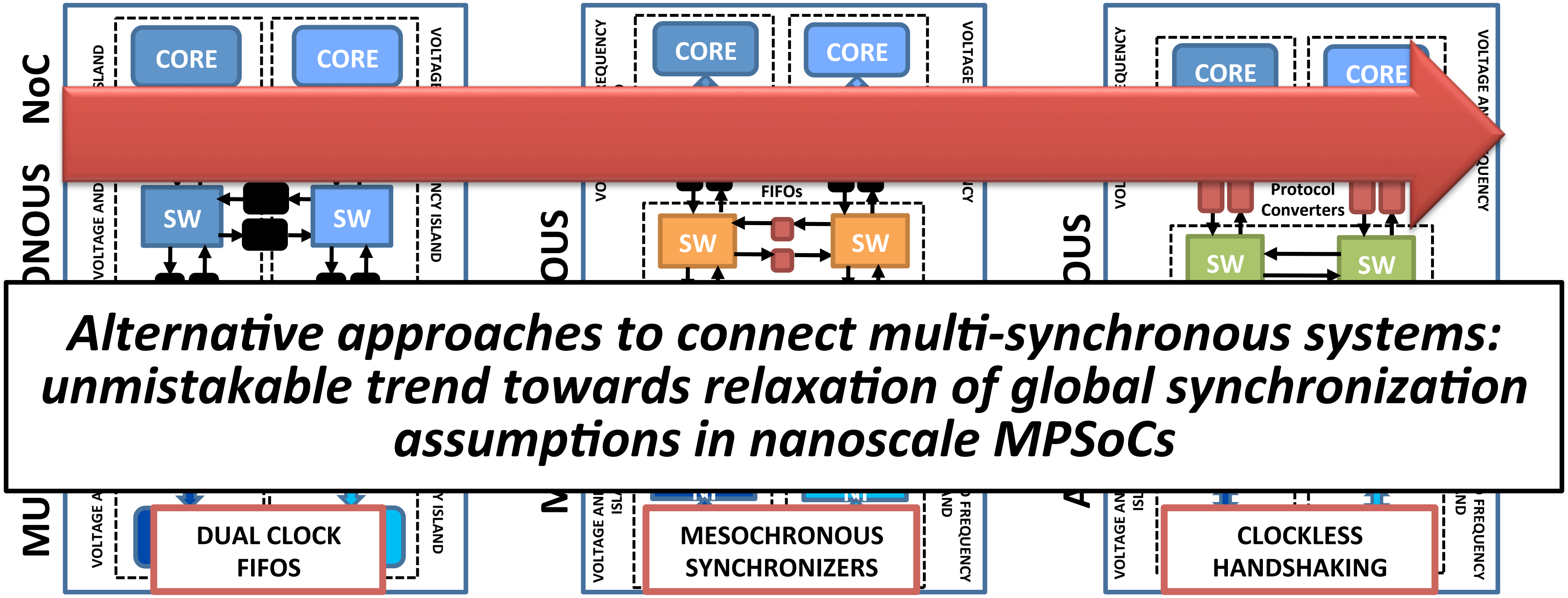


Alternative approaches to connect multi-synchronous systems: unmistakable trend towards relaxation of global synchronization assumptions in nanoscale MPSoCs

State of the Art

MPSoCs increasingly structured as multiple voltage/frequency islands, making their system interconnect challenging

Examples of heterogeneous MPSoCs with multiple clock domains:



Alternative approaches to connect multi-synchronous systems: unmistakable trend towards relaxation of global synchronization assumptions in nanoscale MPSoCs

Clockless handshaking for inter-domain communication holds promise of:

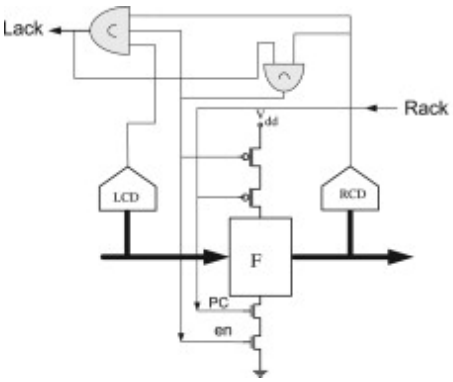
- *average-case performance (instead of worst-case)*
- *no switching power of a clock tree/no clock gating management*
- *robustness to PVT variations*
- *efficient delivery of differentiated per-link performance*

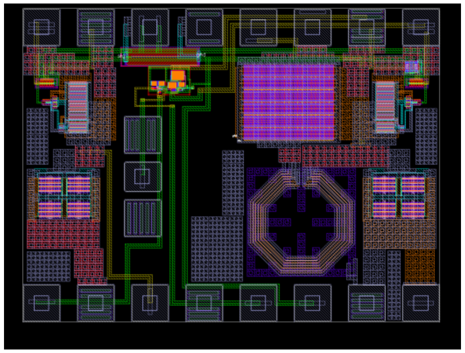
Challenges

However, the potential benefits of asynchronous design paradigm **are not reflected** into the actual industrial uptake.

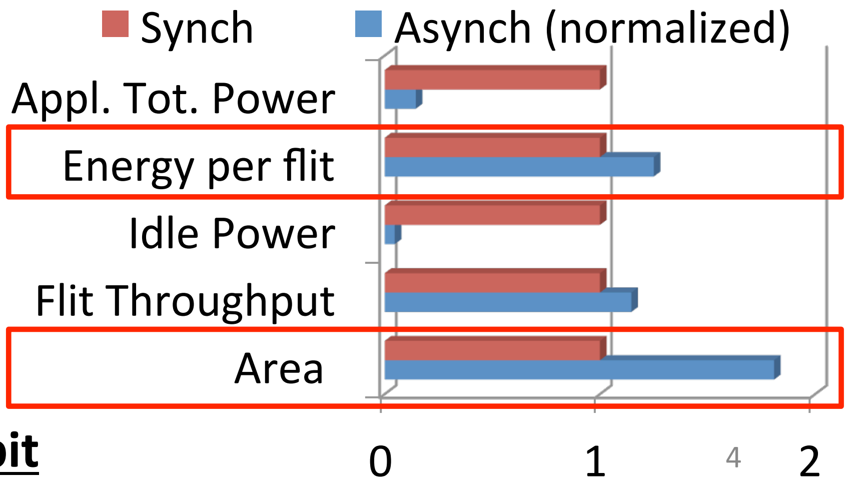
There are **two fundamental barriers**:

- ***Poor CAD tool support***
  - Full-custom approach
  - Rigid hard macrocells





- ***Overly large area and energy-per-bit overhead***
  - Four-phase return-to-zero (RZ) protocols
  - Delay-insensitive (DI) data encodings
  - **Power savings come mainly from reduction of idle power, not energy per bit**

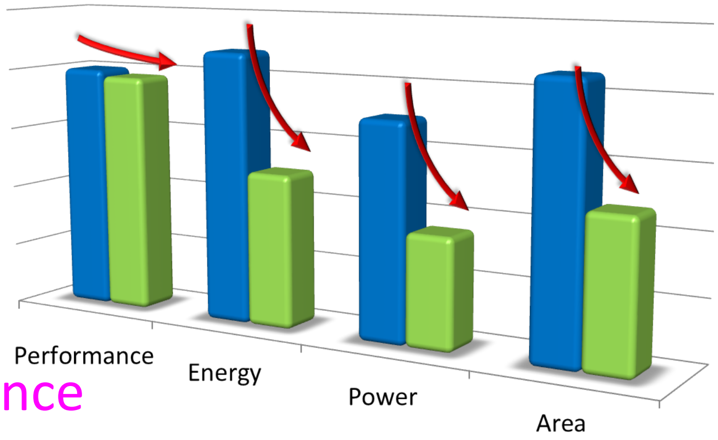


Objectives

Full 5-ported asynchronous switch designed with transition-signaling bundled data protocol

Our goal: a switch architecture that outperforms its synchronous counterpart in terms of:

- energy-per-bit
- power consumption
- area footprint



… while obtaining comparable or better performance

- ➢ we compare to an ultra-low complexity synch NoC as baseline
  - makes this objective even more challenging!

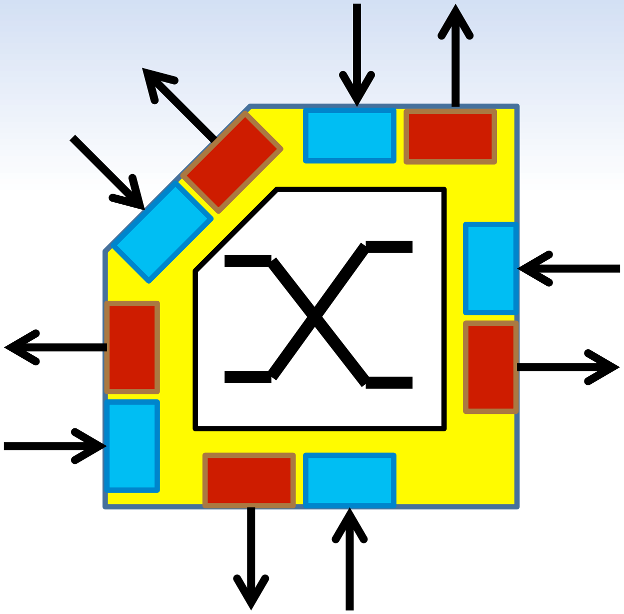
Our goal: be fully compatible with a standard cell design methodology and a mainstream CAD tool flow for synchronous design

- Partially relaxing the hard macro requirement

Detailed Contributions

- Extend state-of-the-art routing and arbitration primitives to a **full 5-ported switch**
- Two-phase protocol and bundled data encoding in **both link and switch**
- **High performance** **(>900 Mflit/s)** in **low-power** standard-Vth 40nm technology
- **Semi-automated design flow**
  - exploits mainstream tools for synchronous design
  - generates partially-reconfigurable standard cell macros
- **Comparison of post-layout designs**
  - new asynchronous switch
  - ultra-lightweight synchronous switch architecture (***xpipesLite)***
- **Link parasitic effects** considered during analysis

Target Switch Architecture



🟦 *Input Port Module*

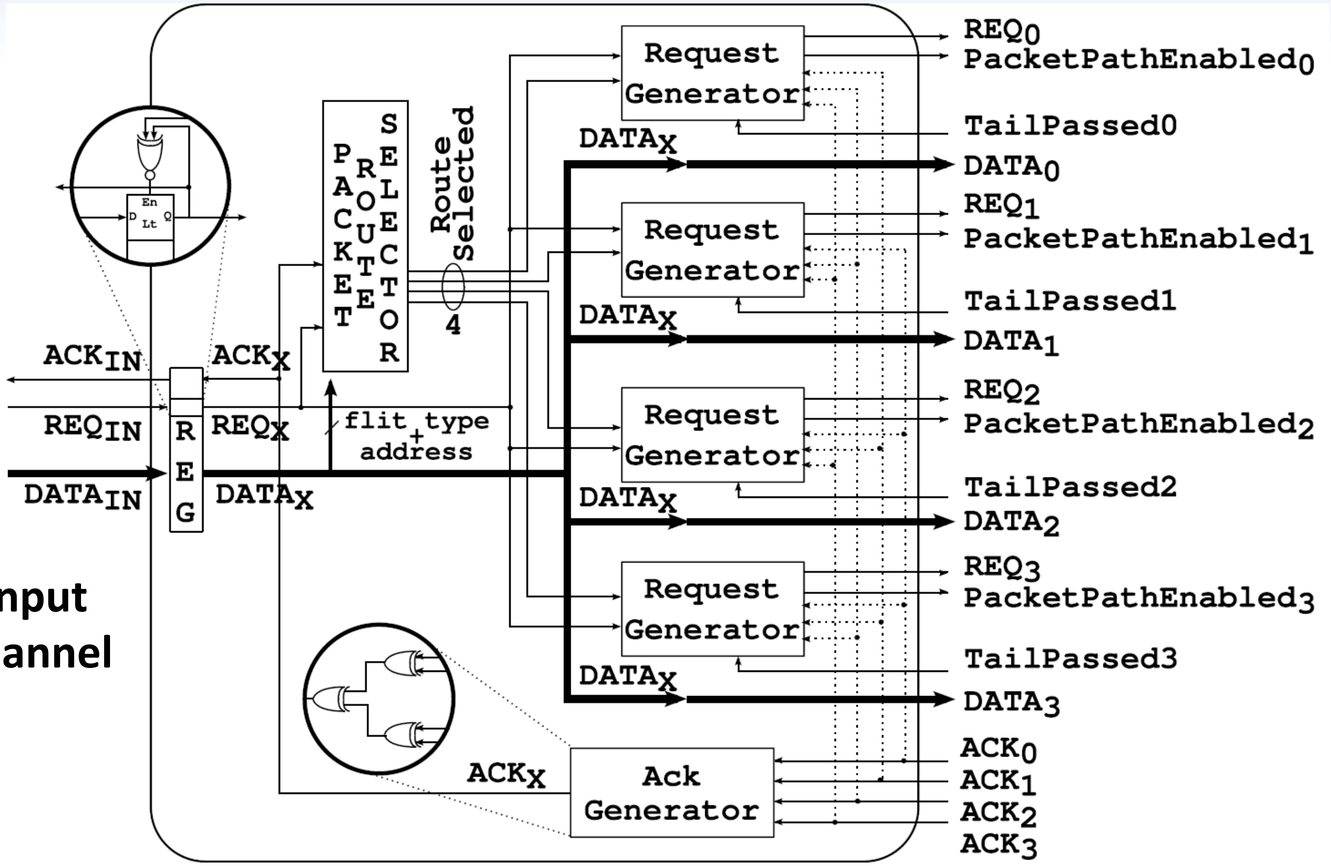
🟥 *Output Port Module*

- ***Features:***
  - ○ **5 input** and **5 output ports**
    - suitable for 2D mesh topology
  - ○ **Parameterizable flit width**
    - E.g., 32 bits
  - ○ **Wormhole switching**
  - ○ **Logic-based distributed routing**
    - algorithmic routing
- ***Based on:***
  - ▪ ***Arbitration and Routing primitives***[1] – simple 1:2 routing and 2:1 arbitration primitives for Mesh-of-Trees network
- ***Inspired by** **(*and benchmarked against*):***
  - ▪ ***xpipesLite***[2] – ultra-low complexity synchronous NoC switch

[1] **M.N. Horak, S.M. Nowick** et. al.,*"A low-overhead asynchronous interconnection network for GALS chip multiprocessors"*, IEEE Trans. on CAD, vol. 30:4, April 2011

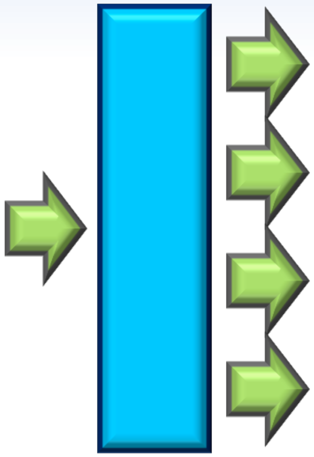
[2] **S.Stergiou** et al., *"xpipesLite: a synthesis-oriented design flow for networks on chip"*, DATE 2005

Switch Architecture
Input Port Module (IPM)



Input channel

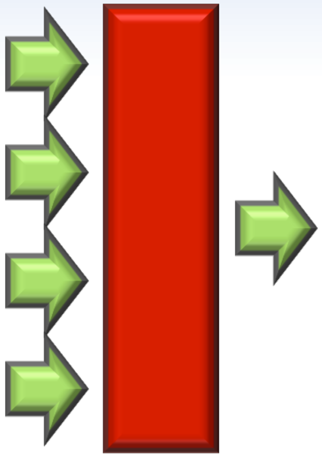
To the associated Output Port Module



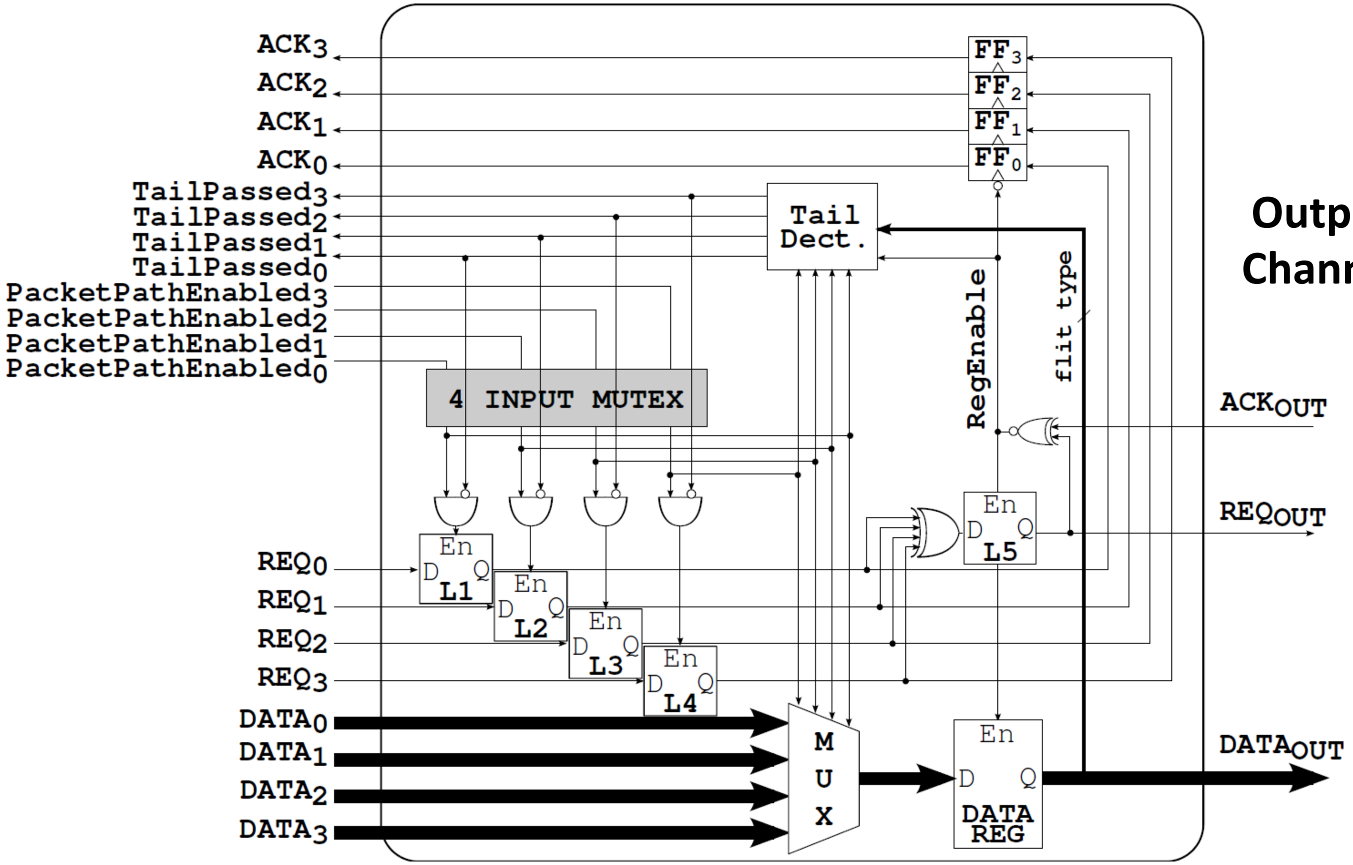
U-turns not allowed: routing only to 4 out of 5 output channels

***IPM**: routes incoming packets to correct Output Port Module, by comparing current switch address with destination address contained in header flit.*

Switch Architecture
Output Port Module (OPM)



From the associated Input Port Module

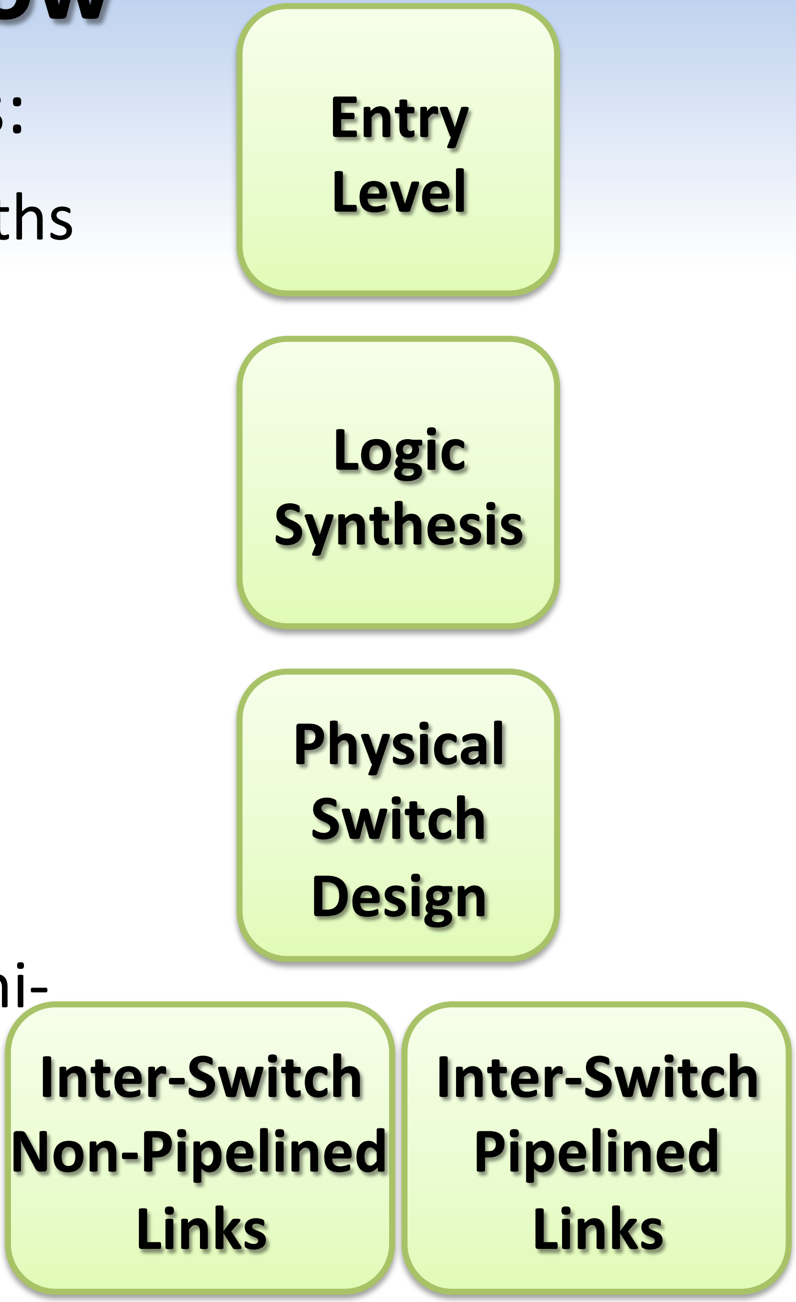


Output Channel

***OPM**: arbitrates between multiple incoming requests trying to access a single associated output channel.*

Design Flow

- Bundled-data protocol requires:
  - **Relative constraints** between paths
    - ➢ correct operation
  - **Absolute constraints**
    - ➢ increase performance
- These constraints have been enforced across all steps
  - – from logical synthesis to layout
- Design methodology
  - – use mainstream CAD tools in semi-automated design methodology



Entry Level

Logic Synthesis

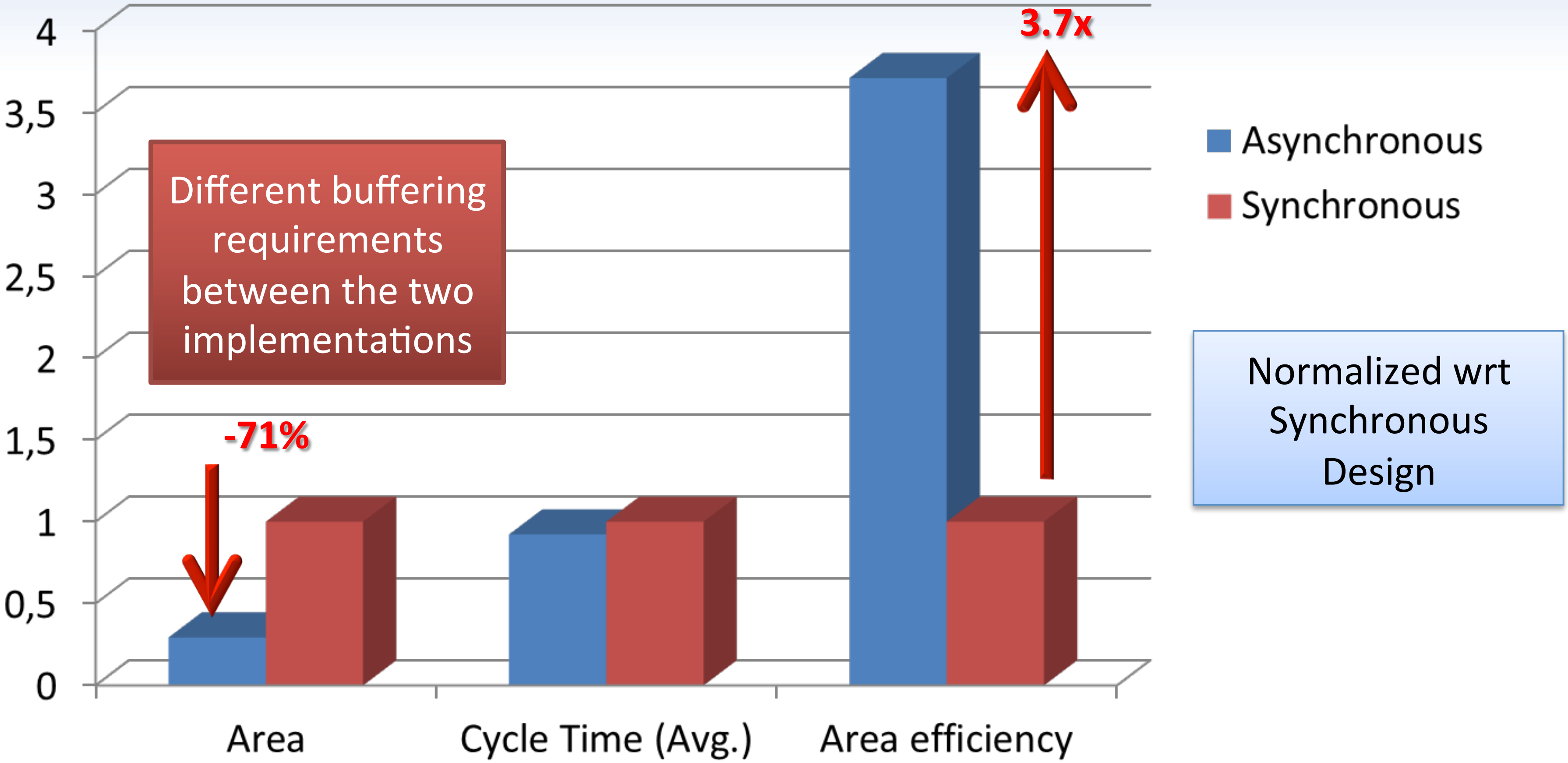
Physical Switch Design

Inter-Switch Non-Pipelined Links

Inter-Switch Pipelined Links

Experimental Setup and Results

Synch vs. Asynch: Comparative Analysis

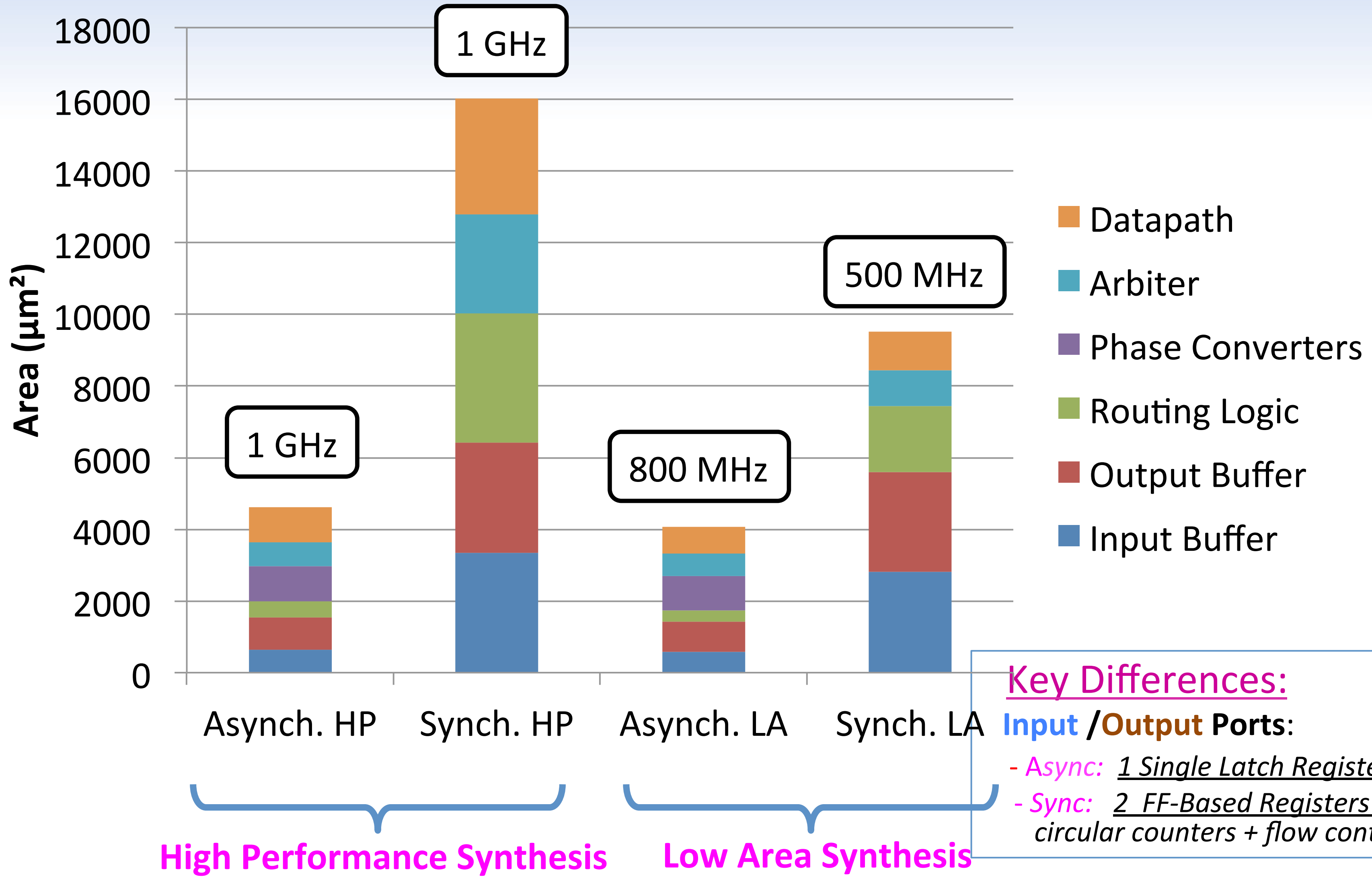


Different buffering requirements between the two implementations

Normalized wrt Synchronous Design

Area efficiency: $\frac{Delivered\ Throughput}{Area\ Occupancy}\left[\frac{Mfps}{mm^2}\right]$

Synch vs. Asynch: Detailed Area Comparison



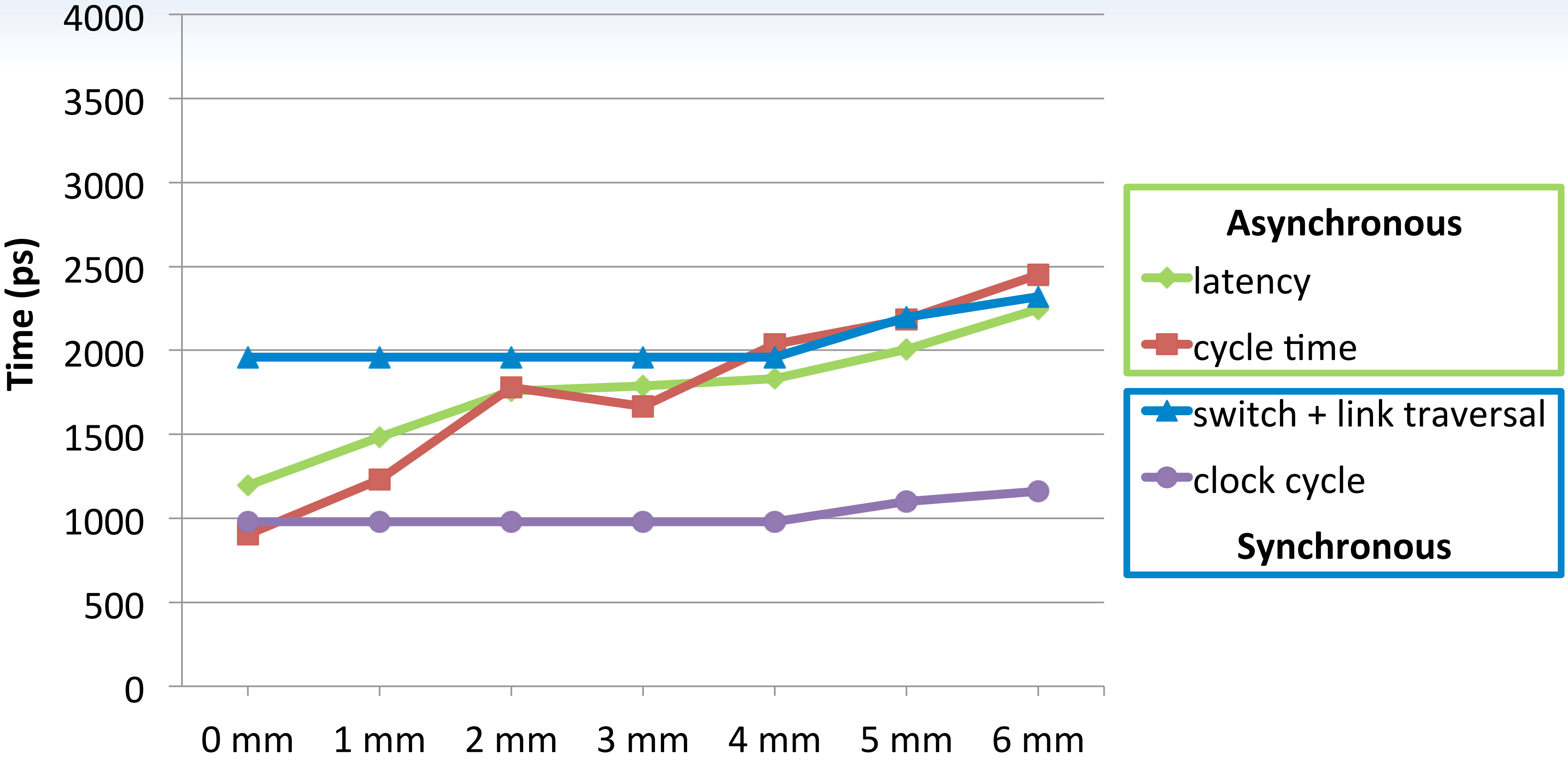
Key Differences:

Input /Output Ports:

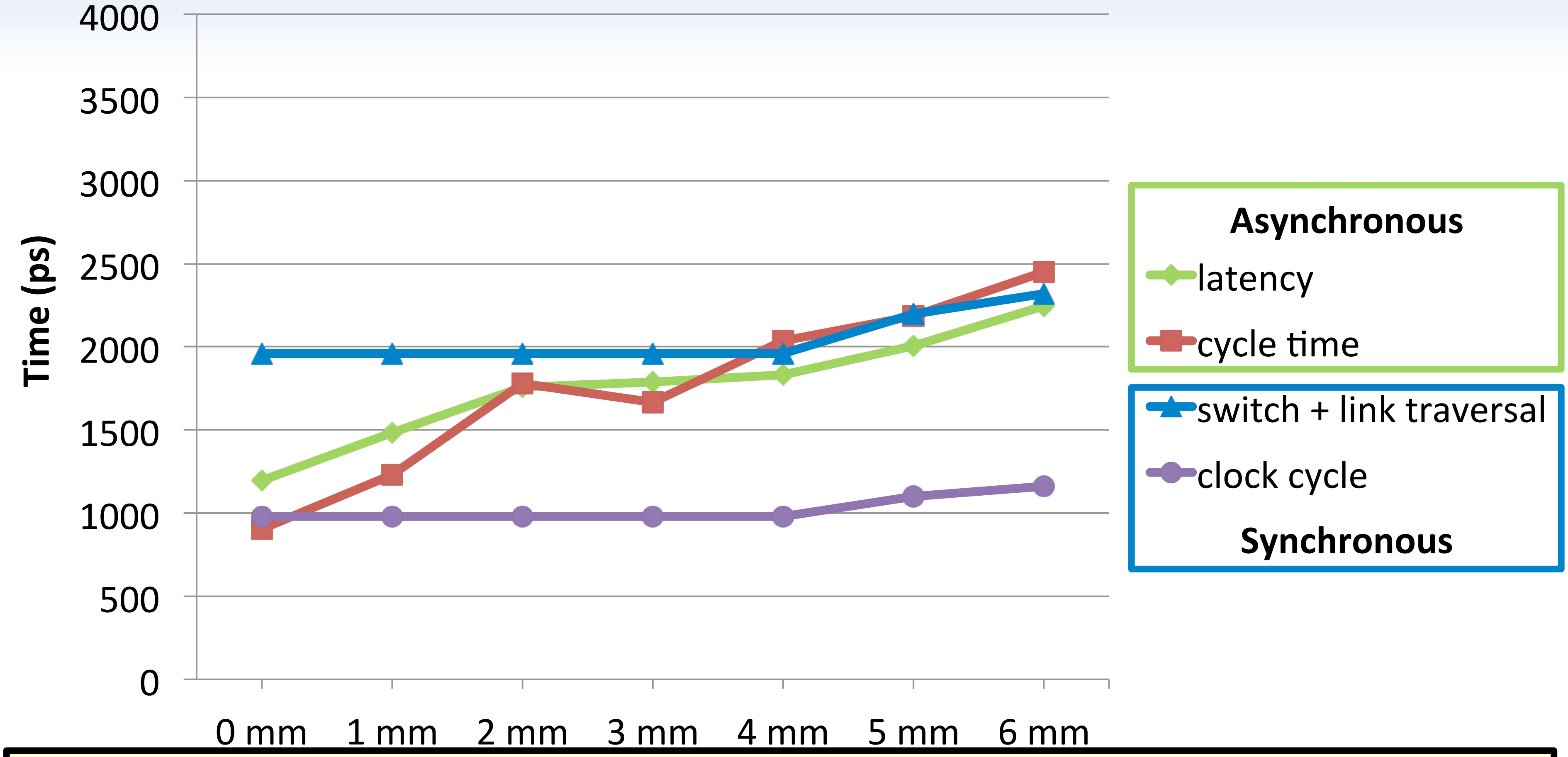
- *Async: 1 Single Latch Register*
- *Sync: 2 FF-Based Registers + circular counters + flow control logic*

→ Async Flow Control: implicitly supported by handshaking protocol

Synch vs. Asynch: Non-Ideal NoC Link Effect



Synch vs. Asynch: Non-Ideal NoC Link Effect



Synchronous switch: stable performance up to 4mm link length, then increase cycle time.

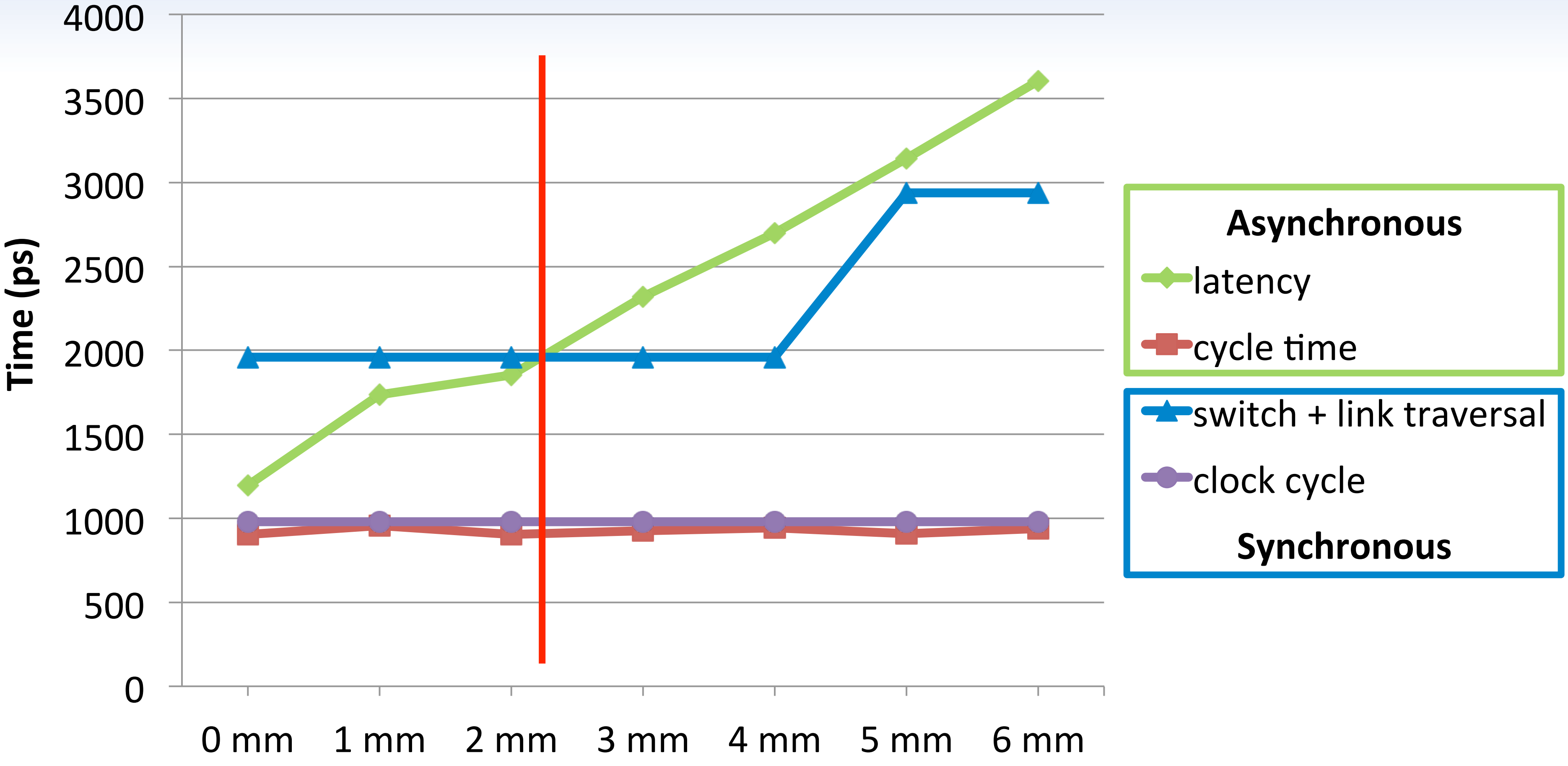
- *1 entire clock cycle reserved for link traversal.*

Asynchronous switch: performance gracefully degrades with increasing link length.

- *Latency always lower, cycle time has steeper increase due to handshaking protocol.*

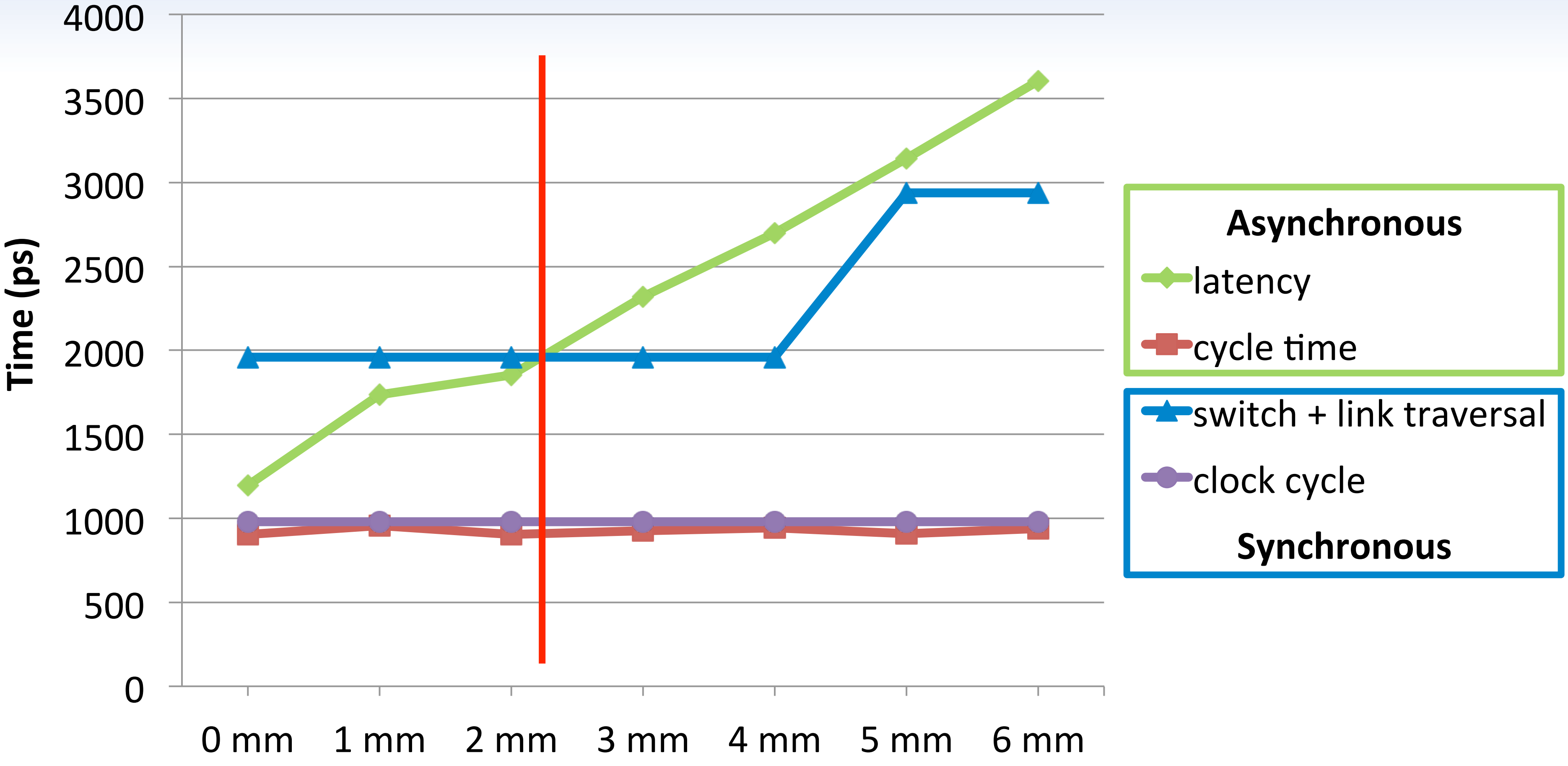
Synch vs. Asynch: Pipelined NoC Link Effect

(maintaining a given throughput)



Synch vs. Asynch: Pipelined NoC Link Effect

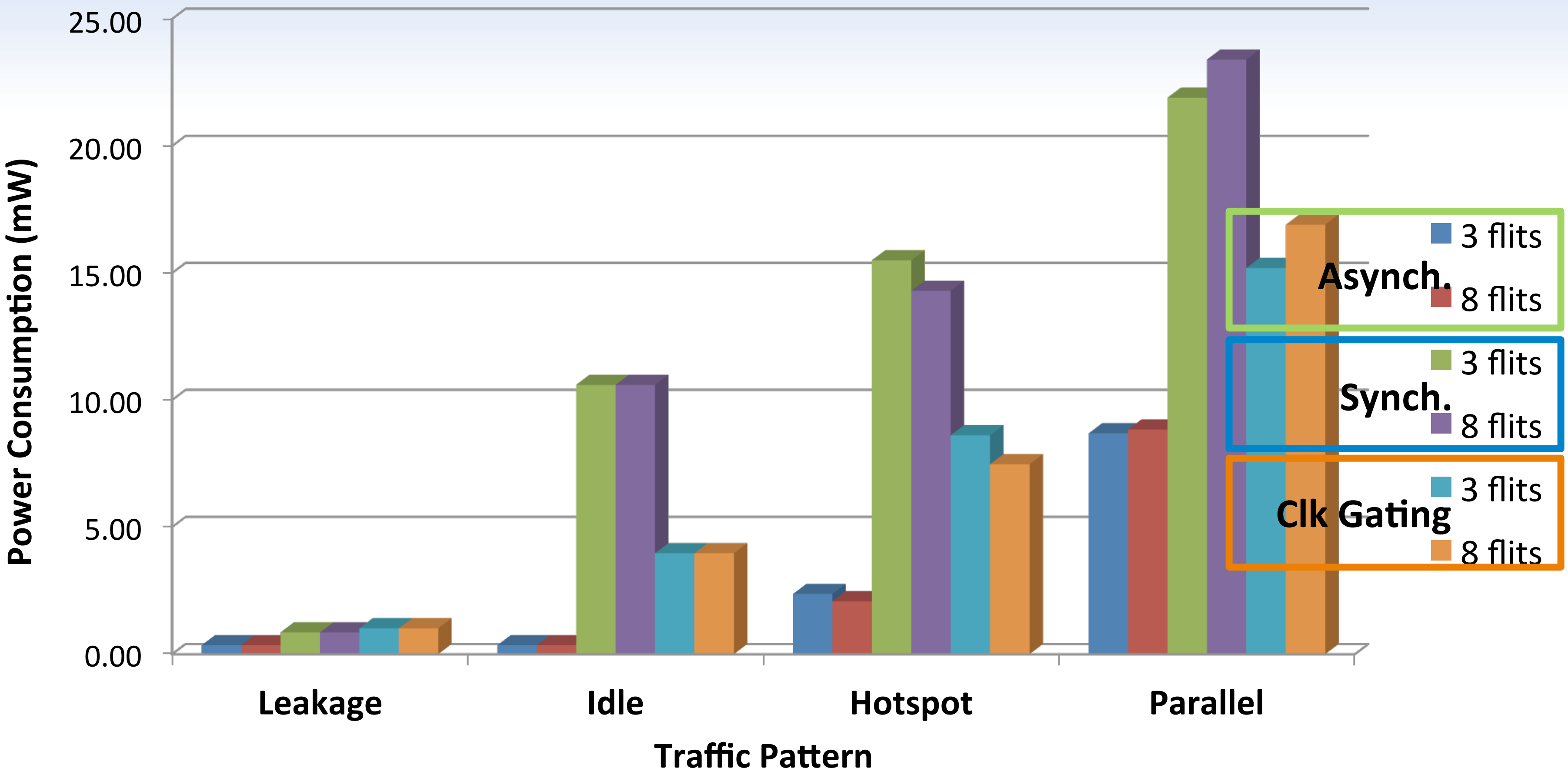
(maintaining a given throughput)



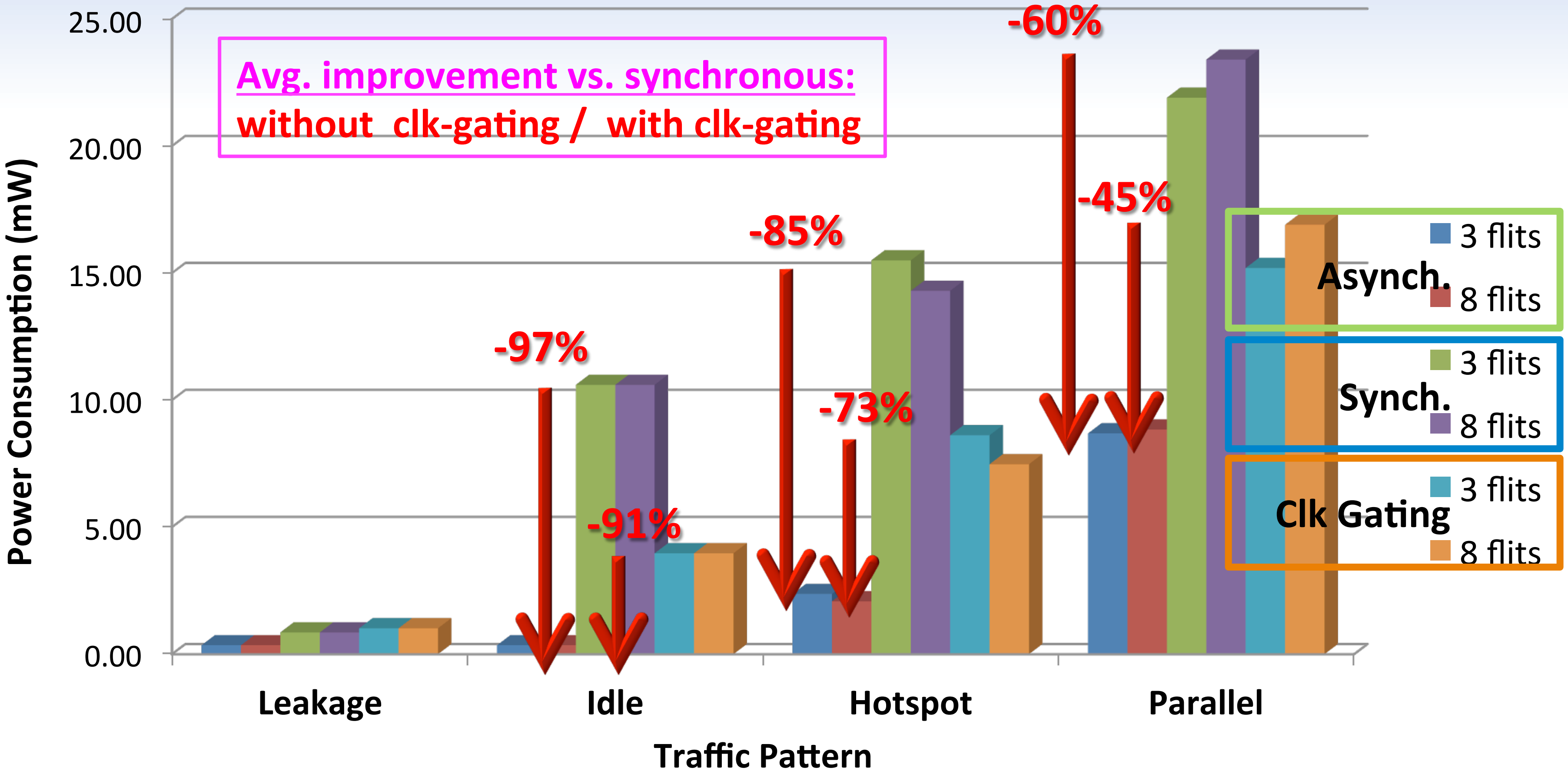
Implications of link pipelining completely different for 2 design styles:

- **Synchronous design*:*** ***a pipeline always requires <u>one additional clock cycle of latency.</u>***
- **Asynchronous design:** ***a pipeline stage adds only a few gate delays*.**

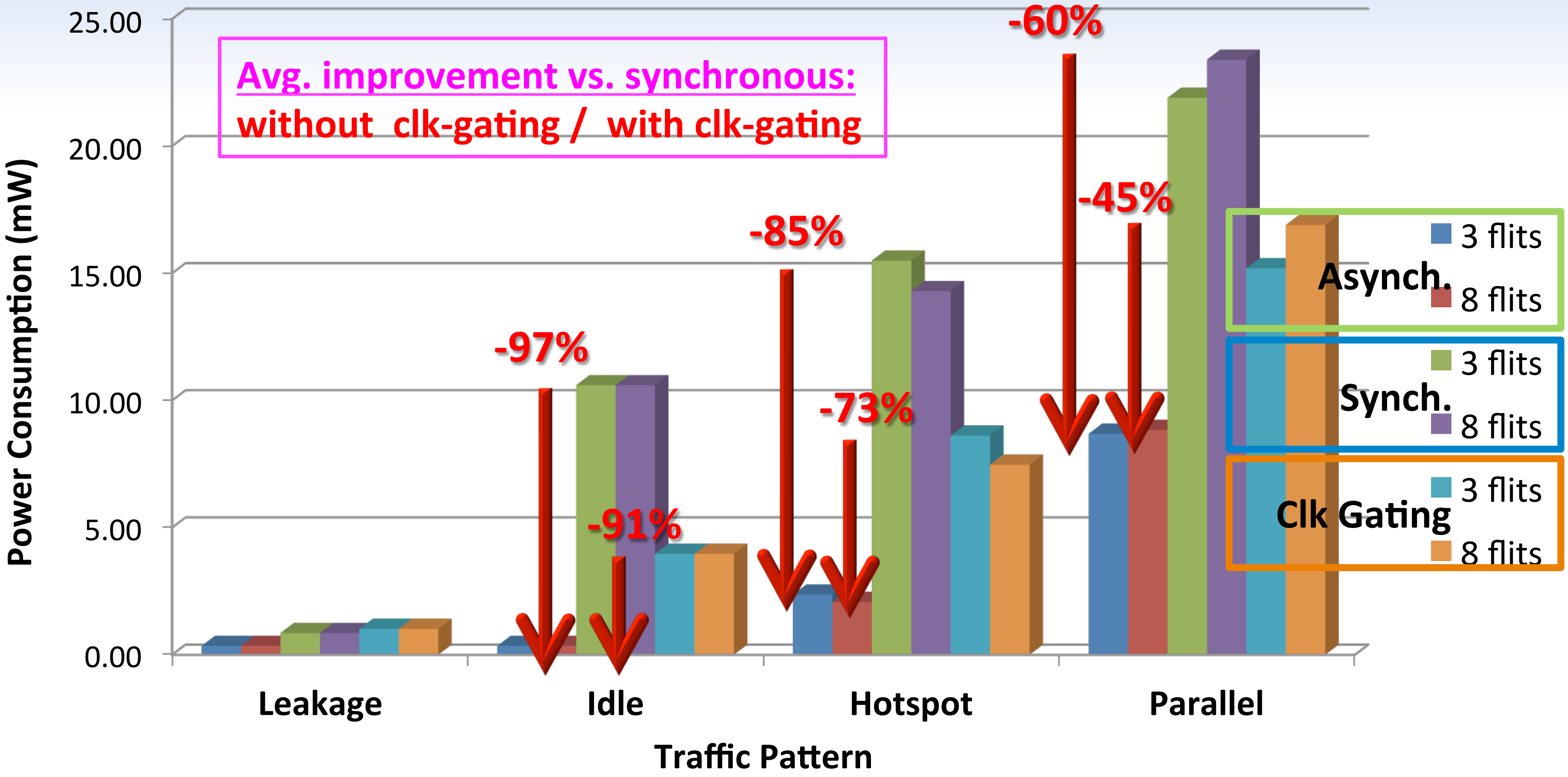
Synch vs. Asynch: Total Power Consumption



Synch vs. Asynch: Total Power Consumption



Synch vs. Asynch: Total Power Consumption



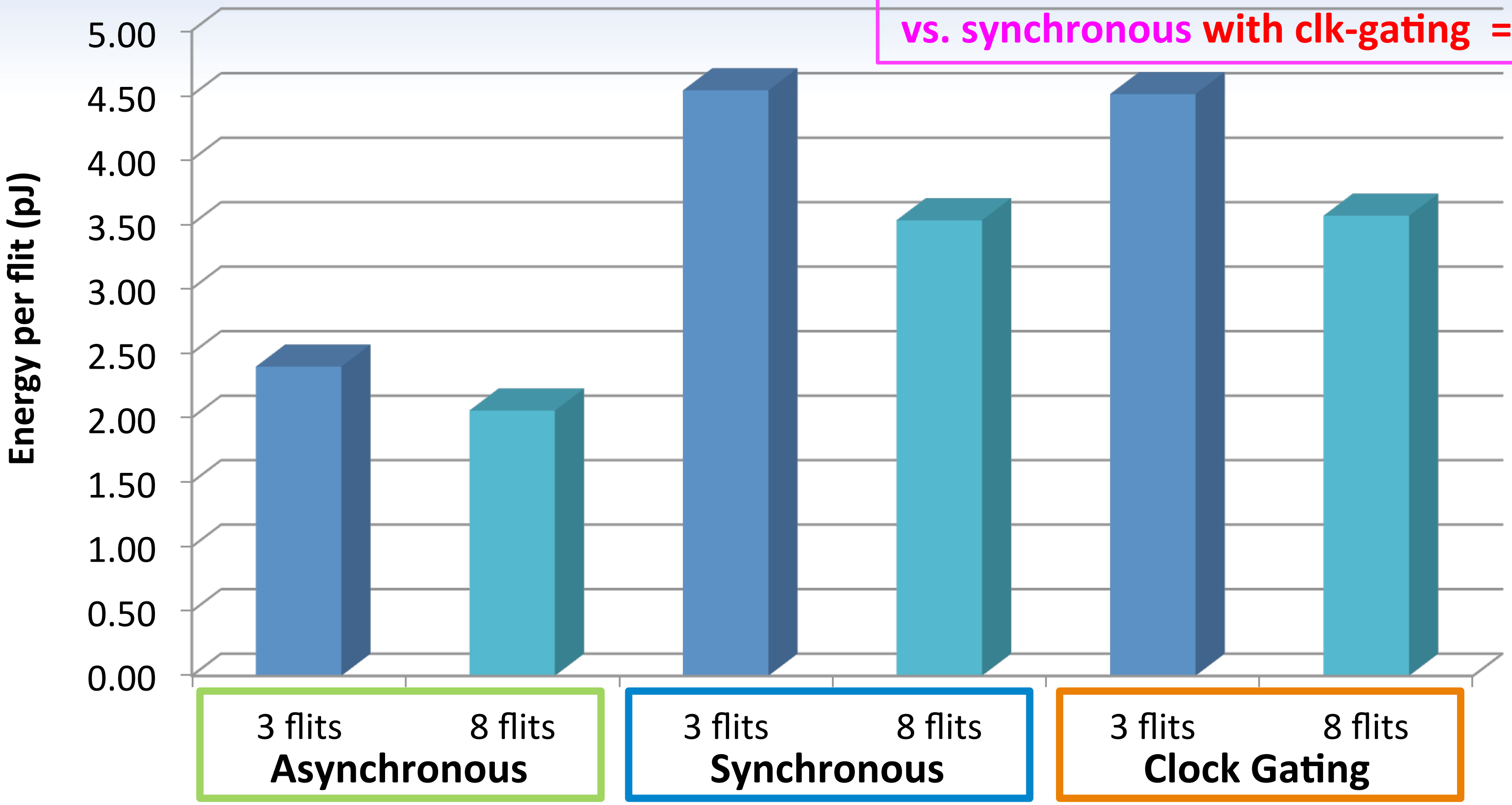
Avg. improvement vs. synchronous:
without clk-gating / with clk-gating

Asynchronous switch reduces idle power consumption even when compared with clock gating techniques → entirely removes clock.

Asynchronous design has significant dynamic power reduction for every traffic pattern considered.

Synch vs. Asynch: Energy per flit

avg. async improvement:
vs. synchronous with clk-gating = 44%



Power savings not only come from idle power (demand-driven operation), but also from reduced energy-per-flit (due to its lower complexity and footprint).

Conclusions

Target a largely unexplored design point in async NoC switch architectures

- Uses transition-signaling (2-phase) protocol + single-rail bundled data
  - low overhead design
  - meets performance of synchronous counterparts
- Post-layout comparison with synchronous counterpart:
  - area: **71% reduction**
  - idle power: **90% reduction**
  - energy-per-flit: **45% reduction**
  - throughput: **comparable**
  - latency: **lower up to link lengths of 2.5mm**
  - overall area efficiency: **3.7x improvement**
- Timing closure achieved through a **semi-automatic design flow** relying on mainstream synchronous CAD tools → still more work to be done.
- Finally, the switch is delivered as a **partially-reconfigurable standard cell macro** for hierarchical design flows.