High Level Synthesis of GALS Systems

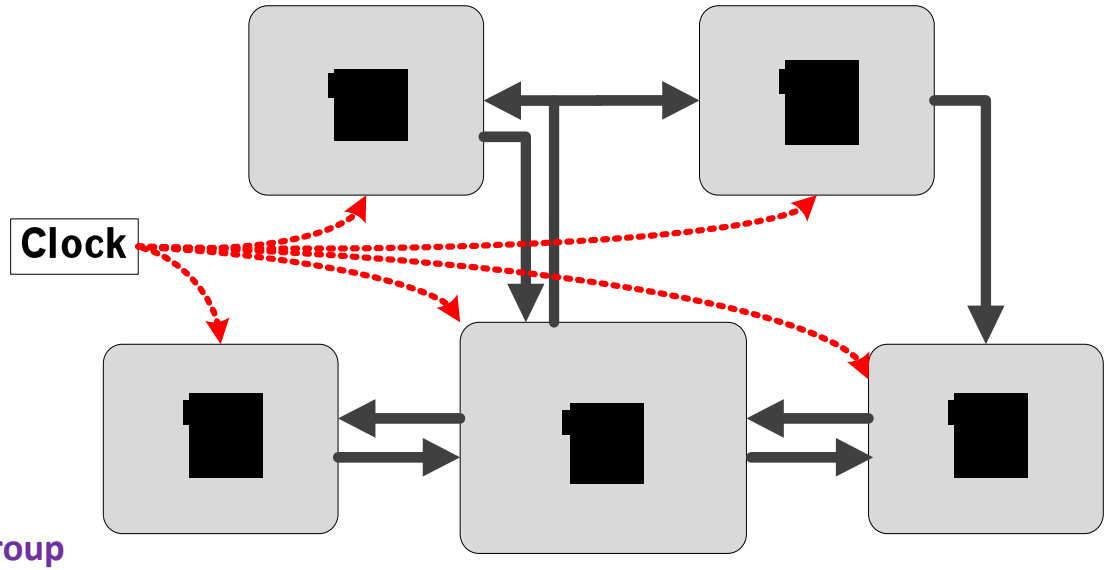
Mahdi Jelodari Mamaghani and Jim D. Garside

School of Computer Science
University of Manchester, UK
17th March 2014

Technology Scaling: Beyond Moore's Law

Aggressive technology scaling has brought up:

- Mismatch between gate and interconnect delays
- Variability in terms of power and clock speed
- Difficulties in power budget management
- Difficulties in clock distribution within a chip



Combating On-Chip Variability

- **Static techniques**
  - ❑ Statistical static timing analysis
  - ❑ Compiler-assisted techniques
- **Dynamic techniques**
  - ❑ Software-level techniques
  - ❑ Tuneable replica circuits
  - ❑ Dynamic Voltage Frequency Scaling
  - ❑ Online error detection and correction
  - ❑ Delay Insensitive/Elastic/Handshake circuits

Are these all applicable to SoC design with heterogeneous timing?

Our Contributions

- to provide designers with an interface to cover their unfamiliarity with **Elastic techniques**, protocols or data-encoding in circuit implementation

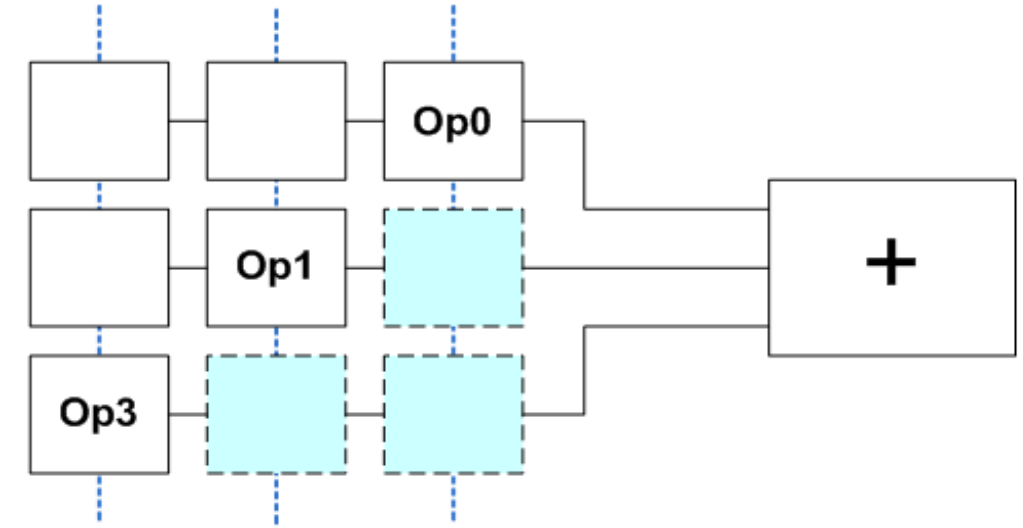
- to specify the hardware in the form **of concurrent data flows** rather than thinking **sequentially** and squeezing tasks into time boundaries

- to raise the **level of abstraction** giving the tool a better chance of **flexible exploration** of the design space based on **formal models** where it is possible to consider different analyses and measurements targeting:
  - Area overhead reduction
  - Performance improvement
  - Optimise power consumption

... All integrated into an EDA flow

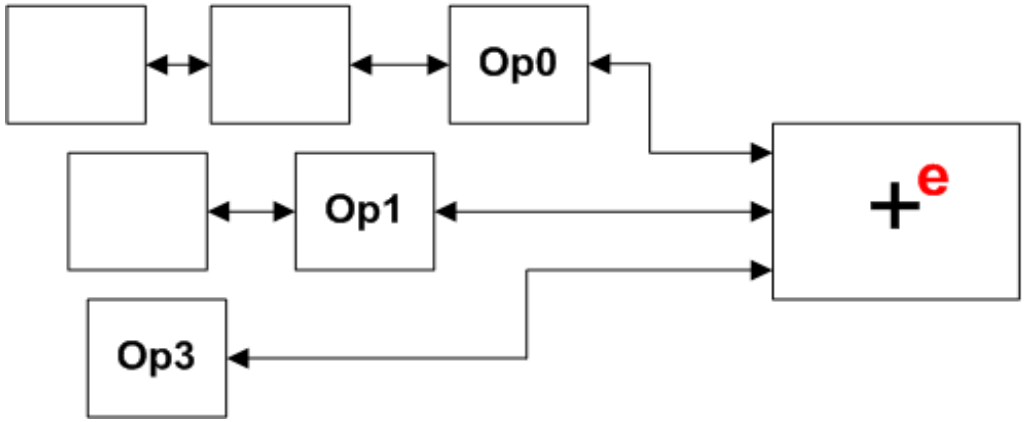
Elastic vs. Conventional Circuits

Timing alignment by Inserting buffers in post-synthesis stage



In Conventional Systems
Latency = 0

No worries!
System tolerates variations in latencies



In
Elastic Systems
Latency can vary

Asynchronous Elasticity

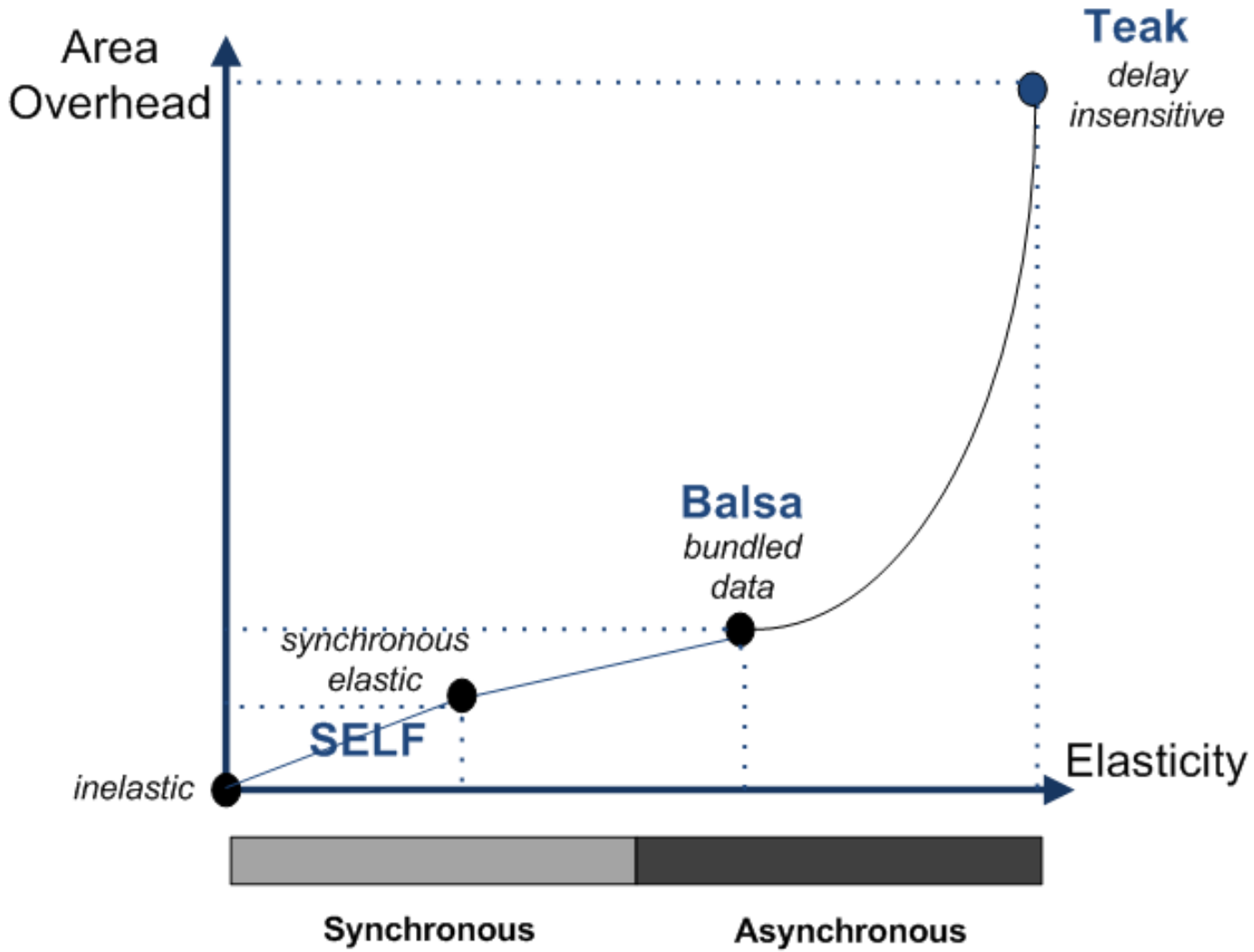
Pros

- Concurrency
- Robustness
  - handshake based communication and computation
- Average-case performance
- Low power & Low EMI
  - Distributed control and intrinsic clock gating

Cons

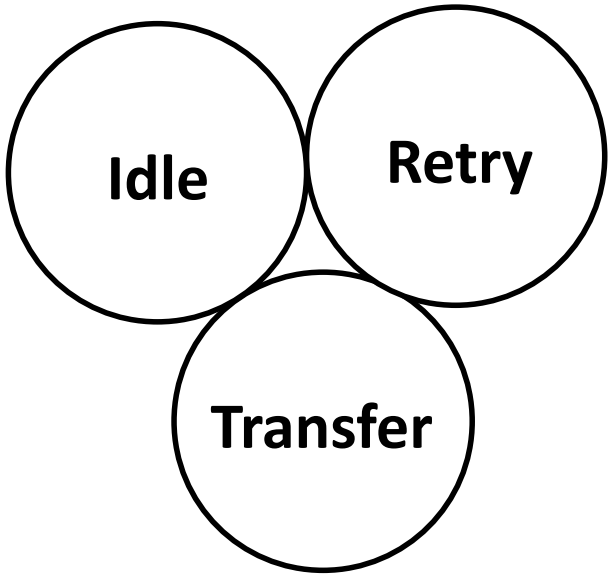
- Lack of mature EDA tools
- Complex handshake protocols
  - Area-expensive structure
  - Complex timing analysis

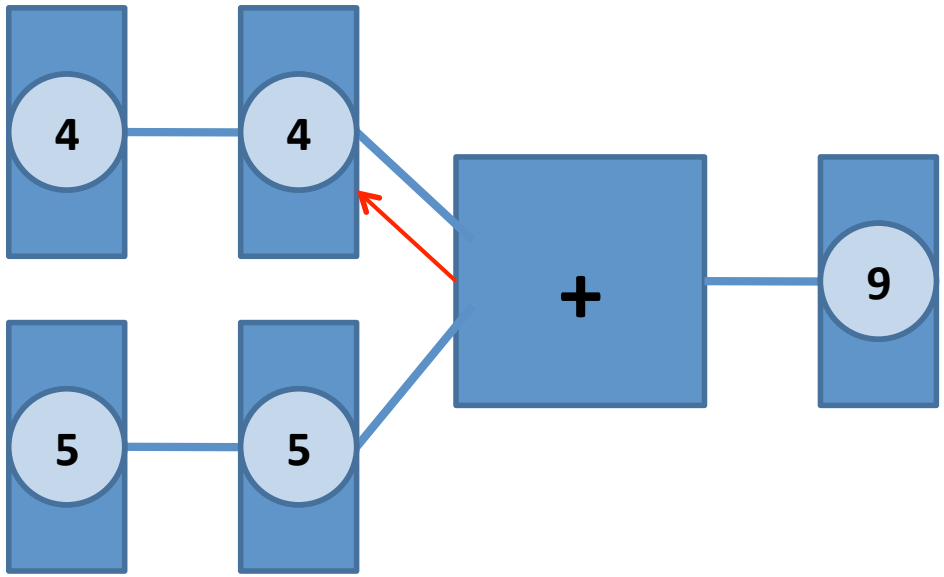
Cost of Elasticity

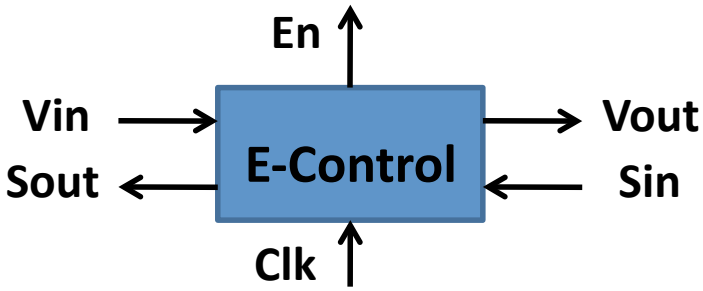


Synchronous Elastic Protocol

Cycle 0
Cycle 1
Cycle 2
Cycle 3
Cycle 4
Cycle 5





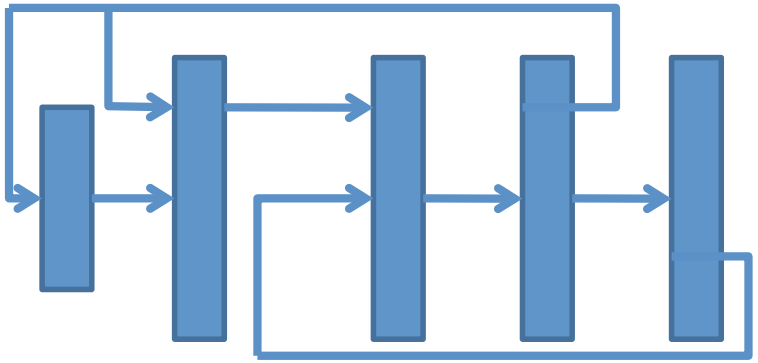


[Jordi Cortadella et al.,2006]

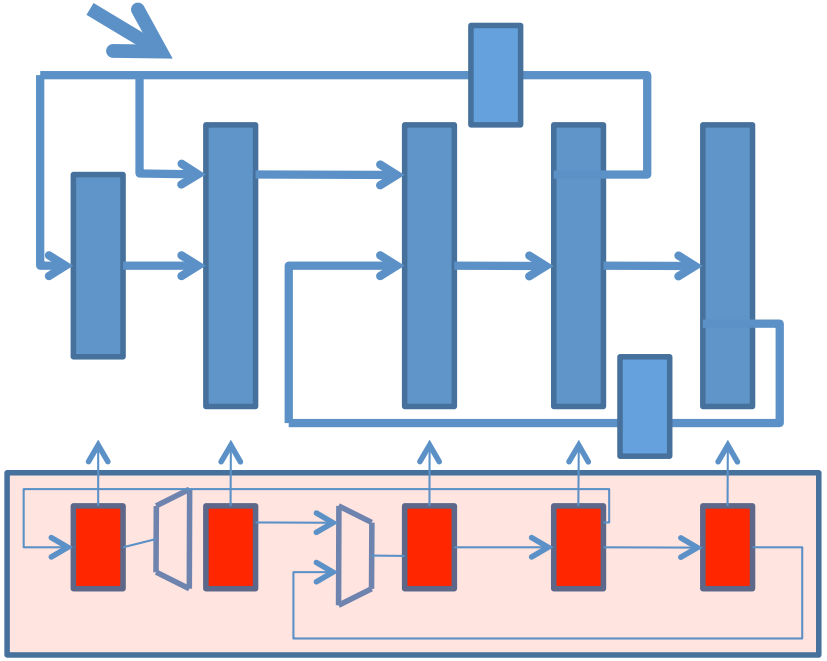
Synchronous Elastic Flow (SELF)

Takes RTL implementation and desynchronises it

Synchronous circuit



Synchronous Elastic Flow



[Josep Carmona et al.,2009]

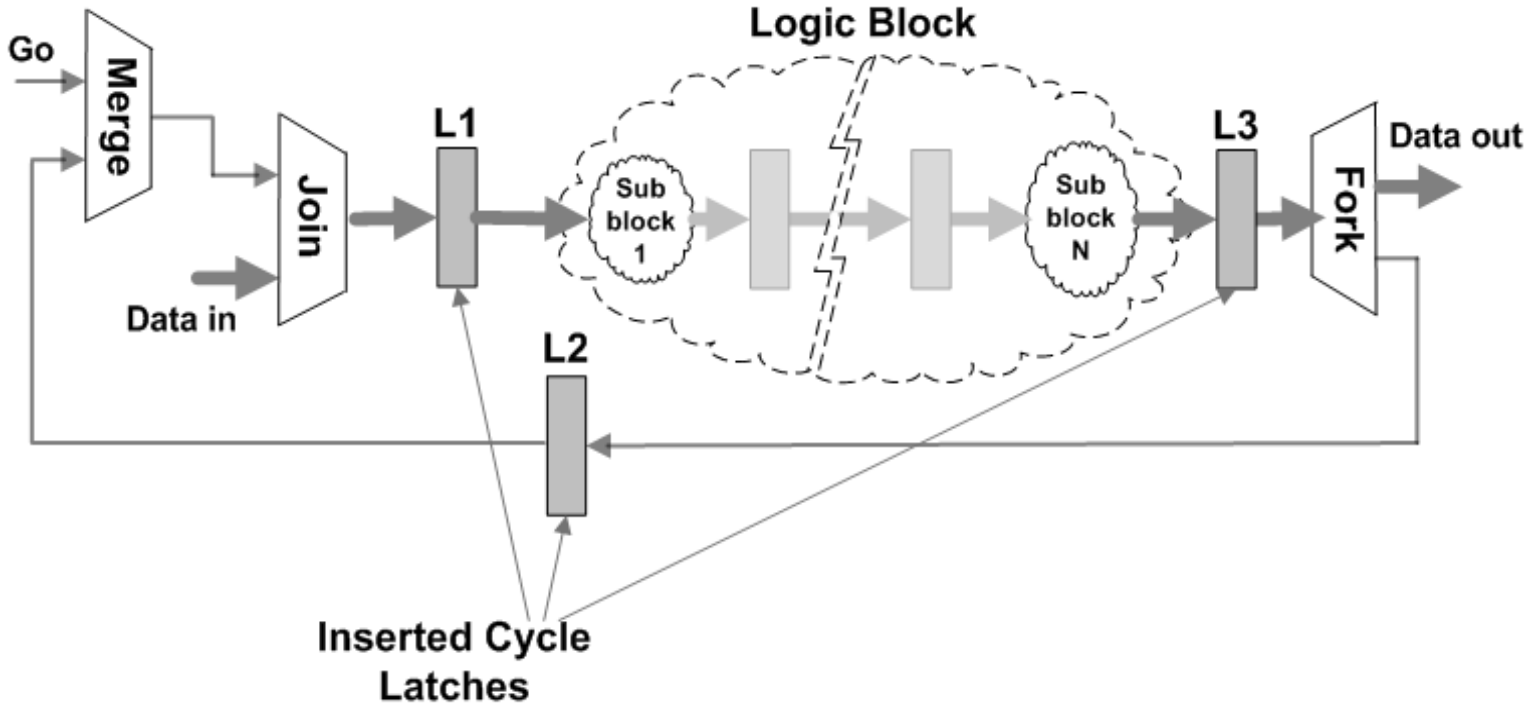
Our Approach: Teak Synthesis System

- Emerged in 2009 as a Dataflow Syntax-directed Synthesis backend for Balsa language

- From the **communication** perspective:
  - **Point-to-point communication** between computation blocks at hardware level; this contributes to concurrency and synchronous massage passing.
  - **Slack elastic:** the communication channels are capable of storing any number of tokens. This feature enables modification in the level of pipelining over the channels without affecting the behaviour of the circuit
- From the **computational** perspective:
  - **Macro-module** style with separate **Go** and **Done** activation signals. These modules are chained in sequence or parallel according to the source level directives. The macro-module architecture contributes to a distributed control mechanism
  - **Dataflow** which realises data-dependent computation. It means that independent data streaming can exist within a module which can significantly influence the performance of the circuit.

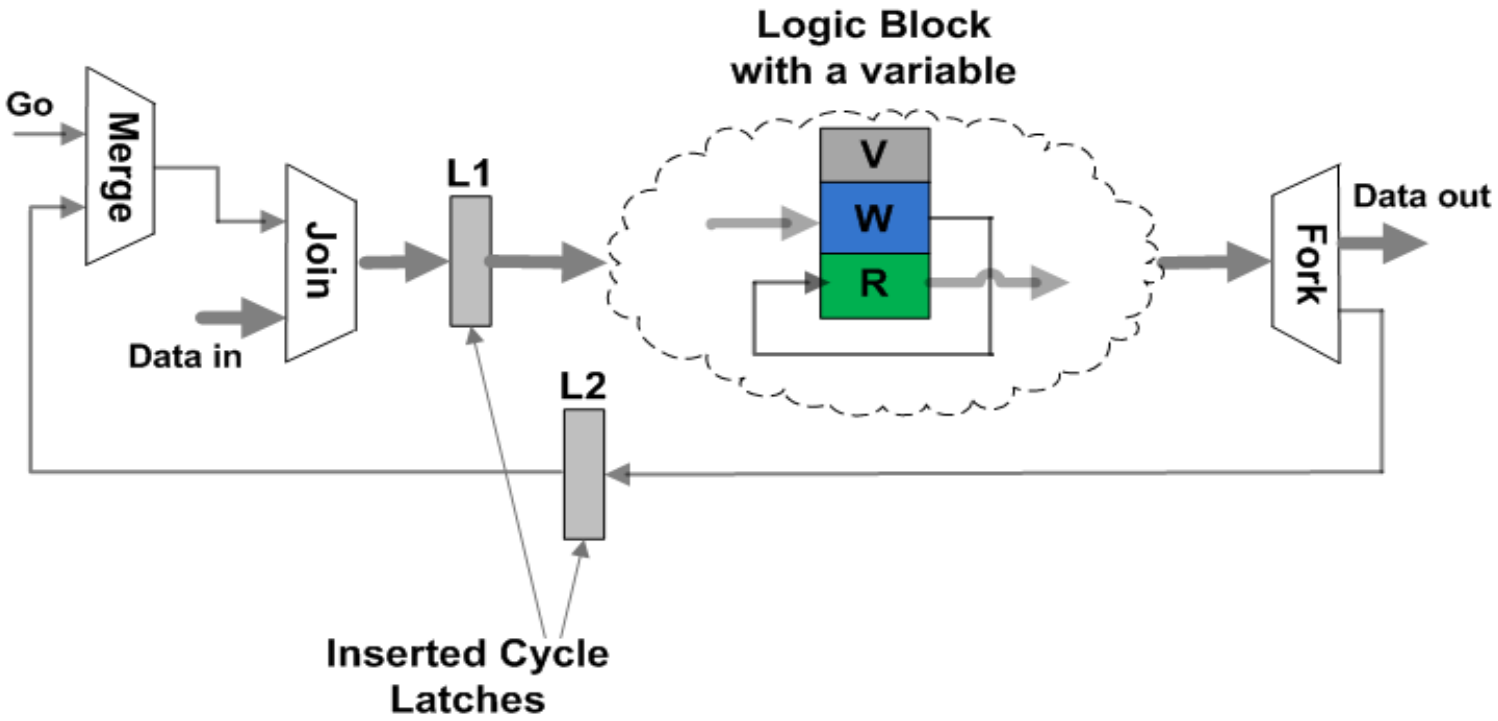
1. Correctness in Teak networks

- 3(+) latches (places) per loop to ensure deadlock freedom
- Any degree of storage on channels
- Decoupling components



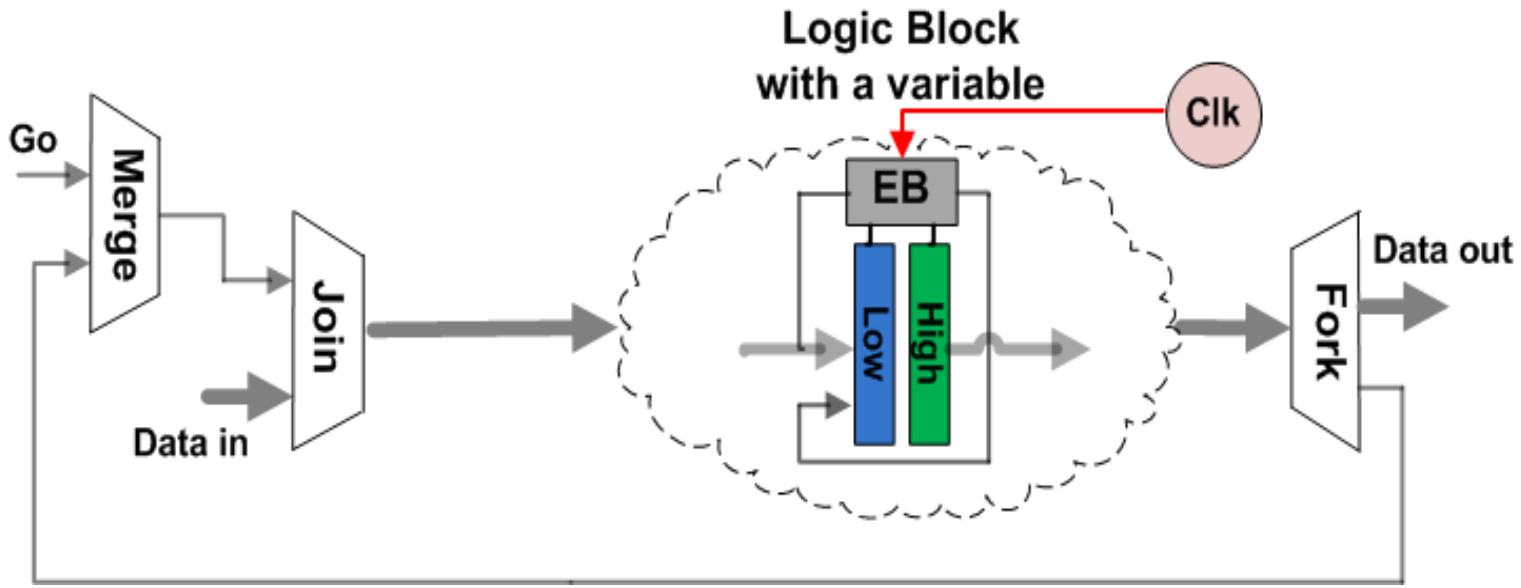
Correctness in Teak networks

- Variables in dataflow networks: Multiple write/ multiple read storage components
- Variable provides a place for data tokens, so 2 < latches to ensure deadlock freedom

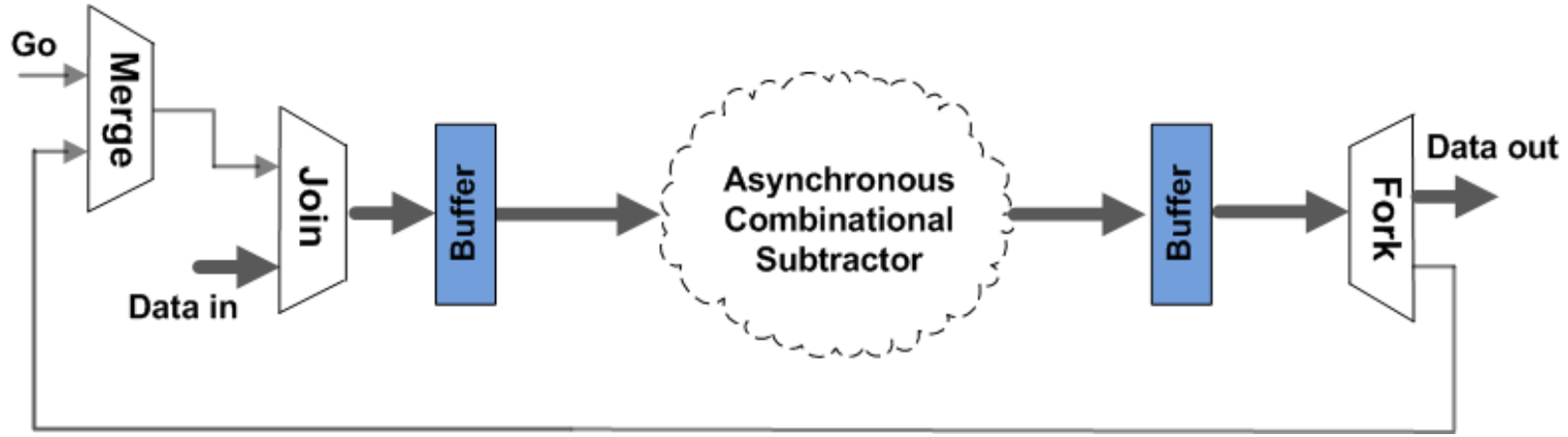


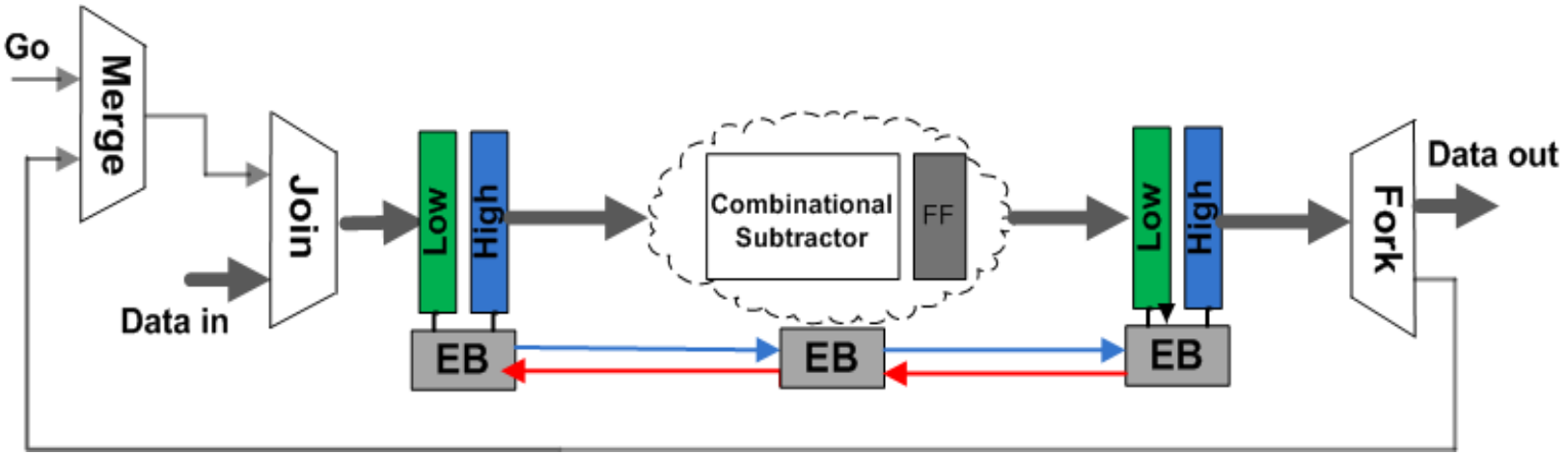
Correctness in eTeak networks

- Variables in eTeak: Elastic Controllers with a pair of latches operating at opposite clock phases
- Each variable provides *two* places for data tokens
- Loops involved in write/read operations do not need extra latches



2. Combinational Logic Synthesis using EDA





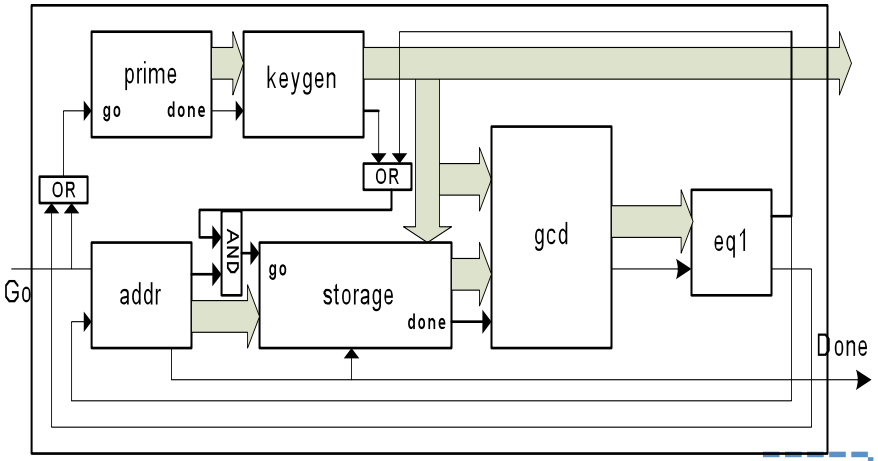
Principles for Partitioning the eTeak Networks

- **SSM**
  - is a network of combinational logic such as binary gates, and sequential logic such as registers.
  - In a SSM a cycle consisting only of combinational elements is not allowed.
  - Synchronous EDA is able to synthesise SSMs from behavioural or structural HDL specifications.

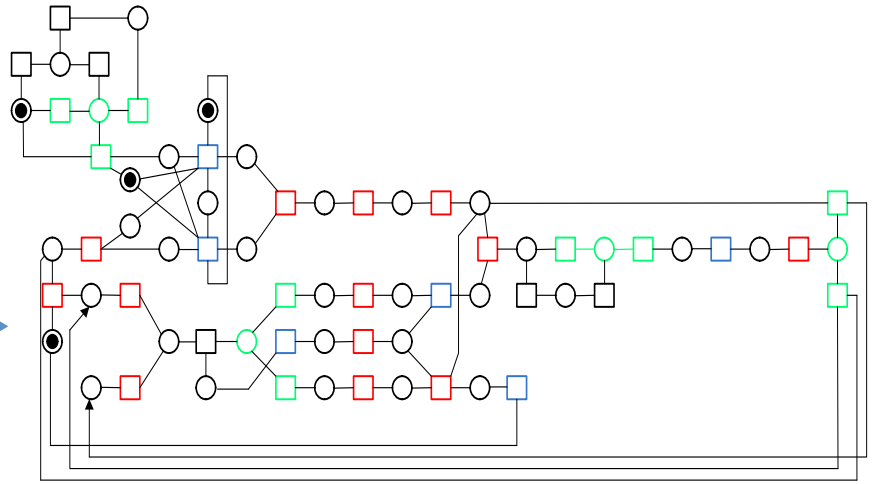
- **Patient SSM**
  - is a latency-insensitive machine whose registers are controlled by a global enable signal.
  - When this signal is low, the state of the sequential elements freezes; no state updates occur.
  - Any SSM is transformable into a patient SSM.

- **Bounded Dataflow Networks**
  - are Dataflow Networks where nodes are connected by bounded FIFOs of any size >= 1.

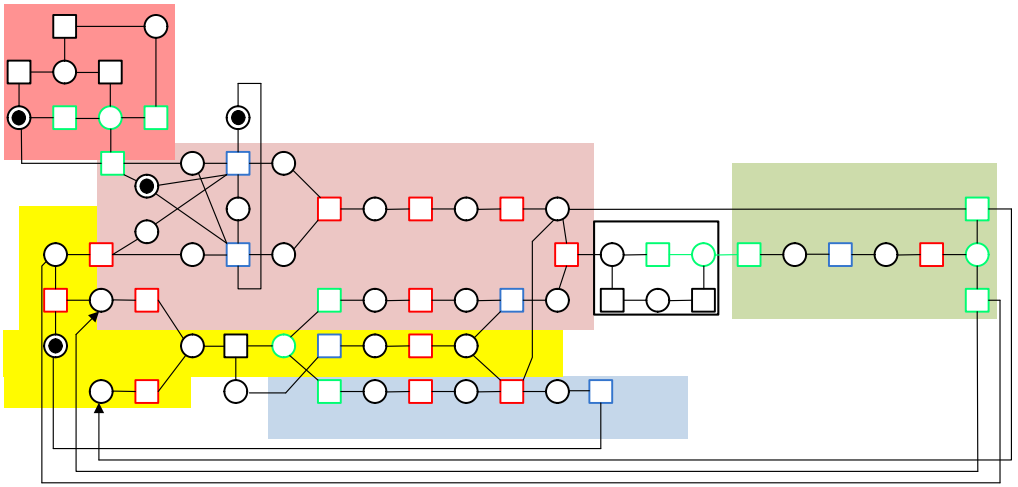
3. Exploiting Synchrony in the Async. Domain



1. Abstract behavioural specification of a circuit



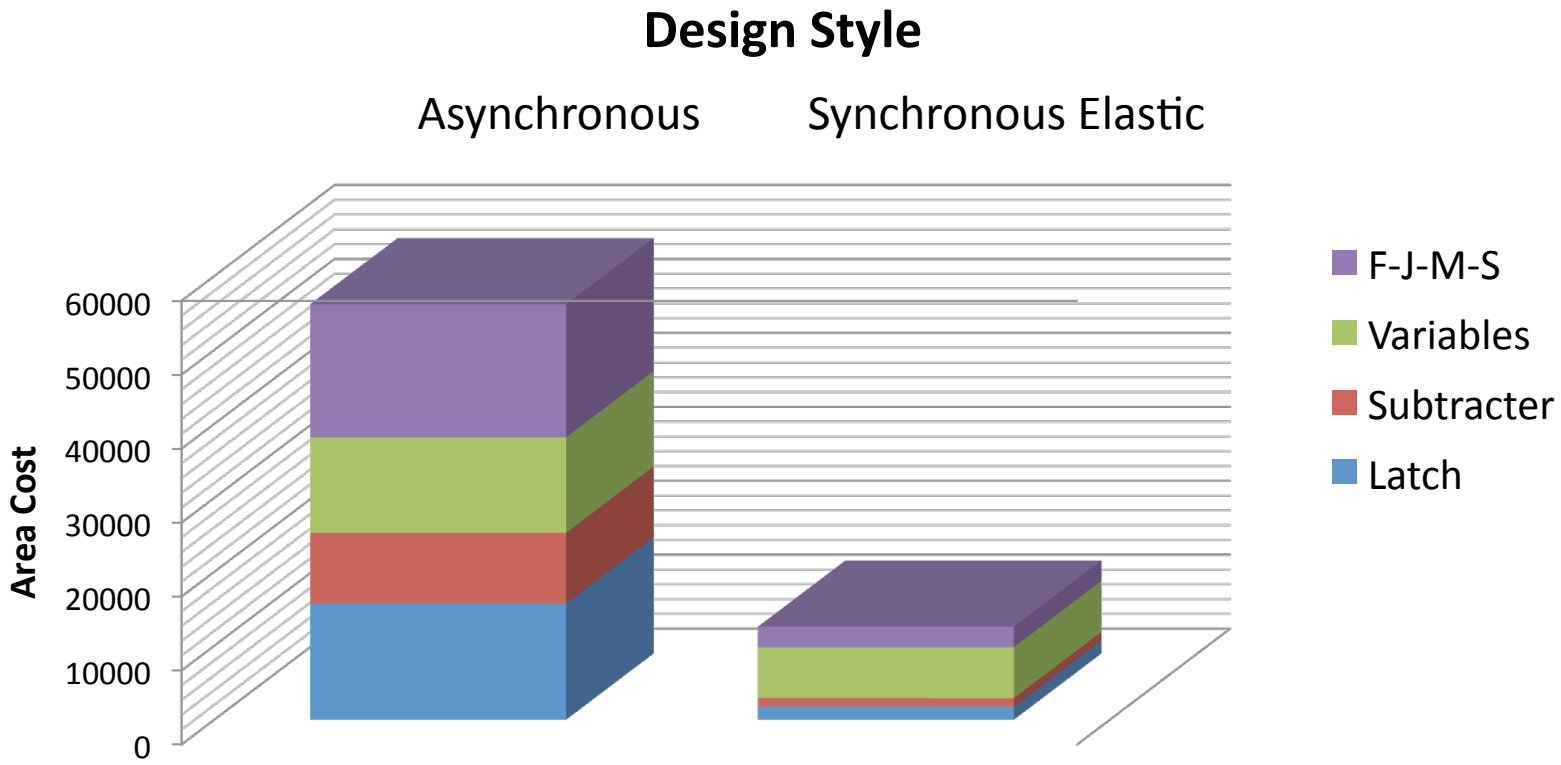
2. Mid-Level representation of the circuit with annotated physical characteristics



3. Synthesis at this level enables us to rapidly explore different trade-offs between power, performance and area

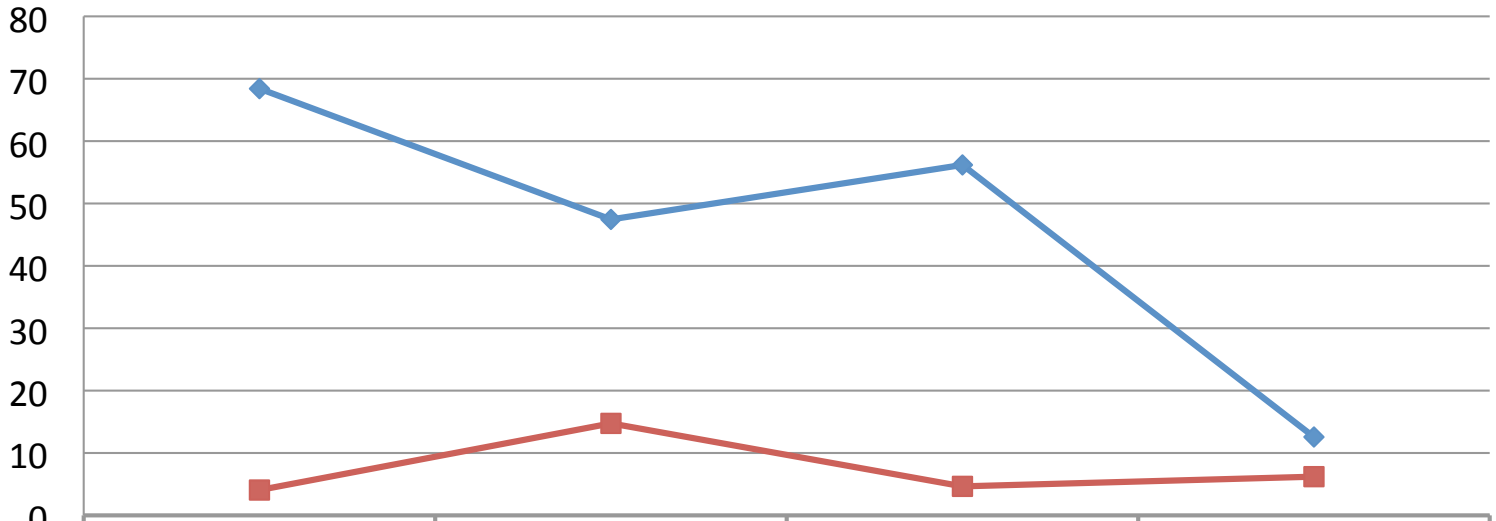
Async. vs. Sync. Elastic: Area Cost

- Case Study: SSEM, A three stage iterative Processor implemented in Balsa
- Deadlock-free design: Async. (**65 Buffers**) vs. Sync. Elastic (**6 Buffers**)
- The slack elastic property is preserved



Asynchronous vs. Synchronous Elastic SSEM

- Application: GCD (67, 2) : 250 Instructions
- Slack Matching is required to improve the performance



	*Asynchronous**	*Synchronous Elastic* (f = 1.250 GHz)*	*Asynchronous*	*Synchronous Elastic (f = 435 MHz)*
Total Cell Area (k)	*68.41*	*47.447*	*56.183*	*12.563*
Exec. Time (ms)	*4.061*	*14.747*	*4.65*	*6.204*

*Fully buffered to approve the slack elastic property

Summary & Future work

- We presented an extension to the Teak EDA flow which provides a framework for exploring *GALSification* techniques
- In this respect, we exploit the synchronous elastic protocol to move from the asynchronous domain to the synchronous domain where it is possible to leverage synchronous EDAs to improve the circuits
- Whilst potential properties of the dataflow circuits are preserved to be employed towards partitioning the network
- Currently, we are working on a high performance communication fabric to allow SSMs interact with each other efficiently
- we will consider running machines with different clock frequencies based on the functional behaviour of each SSM

Thanks for Listening