LOW-POWER HIGH-PERFORMANCE ASYNCHRONOUS GENERAL PURPOSE ARMv7 PROCESSOR FOR MULTI-CORE APPLICATIONS

13th International Forum on Embedded MPSoC and Multicore
July 15-19th 2013, Otsu, Japan

Octasic Inc, Montréal, Canada

Michel Laurence
michel.laurence@octasic.com

FOREWORD

- **At MPSoC 2012 I presented a multi-core asynchronous DSP architecture:**
  - High Computing Performance
  - Very Energy/Power Efficiency
- **We were wondering if the same architecture applied to a general purpose processor (like ARM) could deliver similar performance/power gains.**
- **This presentation provides a summary of the results obtained so far.**

octasic semiconductor

CONTENTS

octasic semiconductor

THE CHALLENGE OF MULTI-CORE "DARK SILICON"

Paper in COMMUNICATIONS OF THE ACM, Feb 2013 :

*Power Challenges May End the Multicore Era**

"As the number of cores increases, **power constraints** may prevent powering of all cores at their full speed, requiring a fraction of the cores to be powered off at all times. According to our models, the fraction of these chips that is "dark" may be as much as 50% within three process generations. The low utility of this "**dark silicon**" may prevent both scaling to higher core counts and ultimately the economic viability of continued silicon scaling.

. . .

Without a **breakthrough in** process technology or **microarchitecture**, other directions are needed to continue the historical rate of performance improvement."

*By Esmaeilzadeh, Blem, St-Amand, Sankaralingam, & Burger

Mike Muller, CTO of ARM had made similar warnings in 2010

octasic semiconductor

EXTENDING THE LIFE OF MULTI-CORE

- **Octasic has developed an Asynchronous core micro-architecture which increases processor (processing efficiency by a factor of 2-3x**
- **This presentation explores if the application of the micro-architecture to a general purpose processor core would entail the same or similar benefits**

octasic semiconductor

CONTENTS

octasic semiconductor

BACKGROUND ON OCTASIC

Founded 15 years ago. Currently ~100 employees

Headquartered in Montreal, Canada

- **Subsidiary in Bangalore, India**

Evolution:

- **98/00** - Design ASICs for others
- **2001** - Convert to fabless model
- **2001- 2003: VoIP** Support Products **(Synchronous)**:
  - **2001 - Voice Packetization Engine / OCT8304**
  - **2003 - Echo Cancellation Processor / OCT6100**

- **2004 – DSPs (Asynchronous)** for Voice, Video, and Wireless Baseband
  - **2008 - First Generation / OCT1010**
  - **2011 - Second Generation / OCT2224**
  - **…2014 - Third Generation / OCT3XXX**

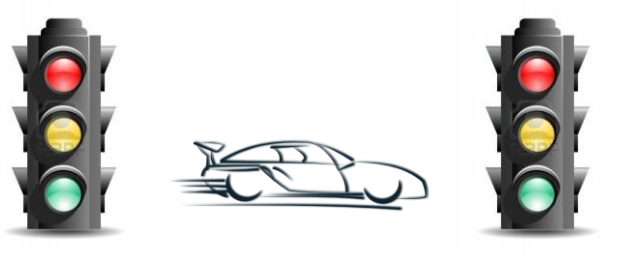
CONTENTS

octasic semiconductor

BASICS OF ASYNCHRONOUS TECHNOLOGY

With synchronous technology

- The control of the flow of information in a chip is controlled by a clock or a set of clocks
- This is analogous to the traffic flow control in a city with **traffic lights**



With asynchronous technology

- The control of the flow of information in a chip is controlled by feedback from one circuit to the other
- This is analogous to the traffic flow control in a city via **round-abouts** rather than **traffic lights**

BASICS OF ASYNCHRONOUS TECHNOLOGY

There are advantages and disadvantages with both methodologies:

?

With synchronous methodology (traffic lights):

- the flow of traffic is centrally controlled, deterministic, hence more easily modelled, tools are easier to implement
- but there are inefficiencies – cars can be waiting uselessly on a red light while there is no traffic in the perpendicular direction. … and clocks contrary to traffic lights consume a LOT OF ENERGY.

With asynchronous methodology (round-abouts)

- the flow of traffic is decentralized, thus less deterministic with tools not as easy to develop and use
- traffic can be more efficient, each car can proceed at its optimal speed not at a fixed forced speed, and overall save fuel

octasic semiconductor

CONTENTS

octasic semiconductor

ARM CORE PROJECT OBJECTIVES

Must be functionally identical with ARMv7

- Object code compatible
- Single thread performance parity
  - May improve performance with “tuned” compiler

Must be able to use off-the-shelf IDE tools

- Debug interface compatibility
  - Coresight compatibility

Must Deliver 2-3x Processing Efficiency (Energy)

- Same performance using ½ – ⅓ the power

octasic semiconductor

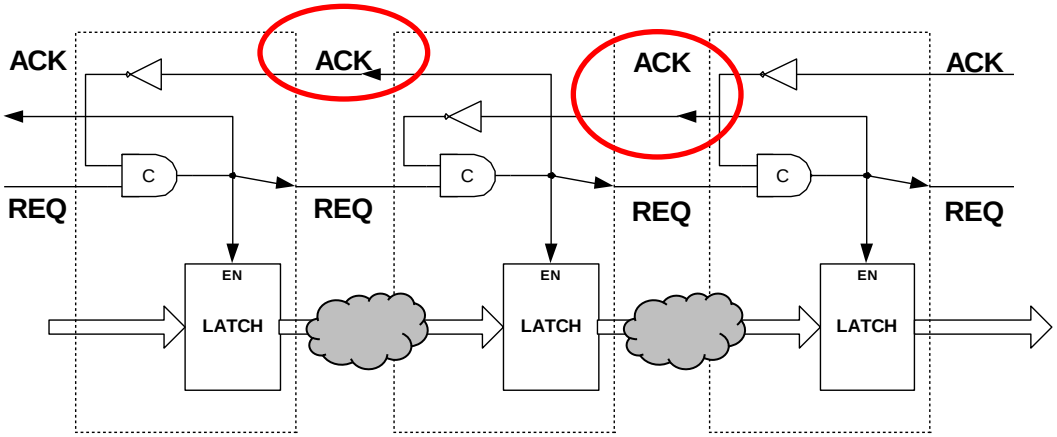
CONTENTS

octasic semiconductor

OCTASIC ASYNCHRONOUS TECHNOLOGY

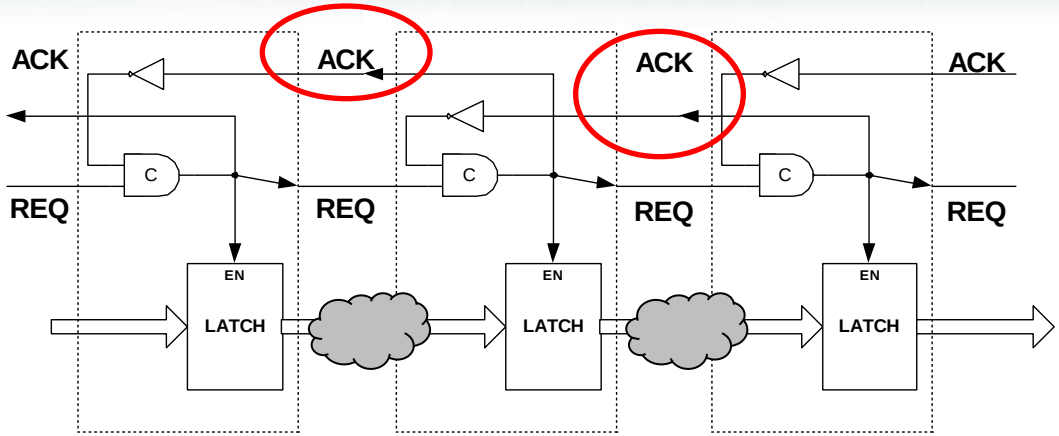
Octasic Asynchronous Architecture is loosely characterized as:
Single Rail Bundled Data (SRBD)

Traditionally with SRBD each forward path stage is timed by handshake feedback from next stage for availability (ACK)

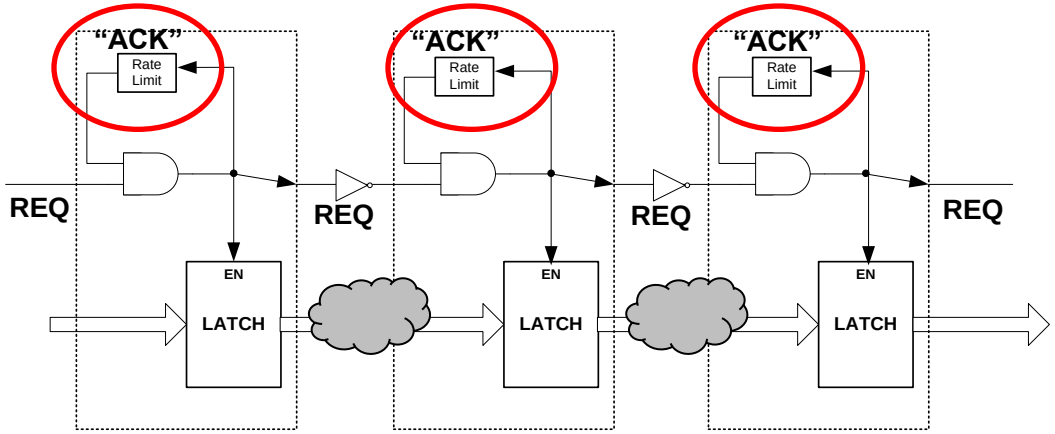


This requires **Special Silicon Cell** & **Specialized Timing Tools**

OCTASIC ASYNCHRONOUS TECHNOLOGY



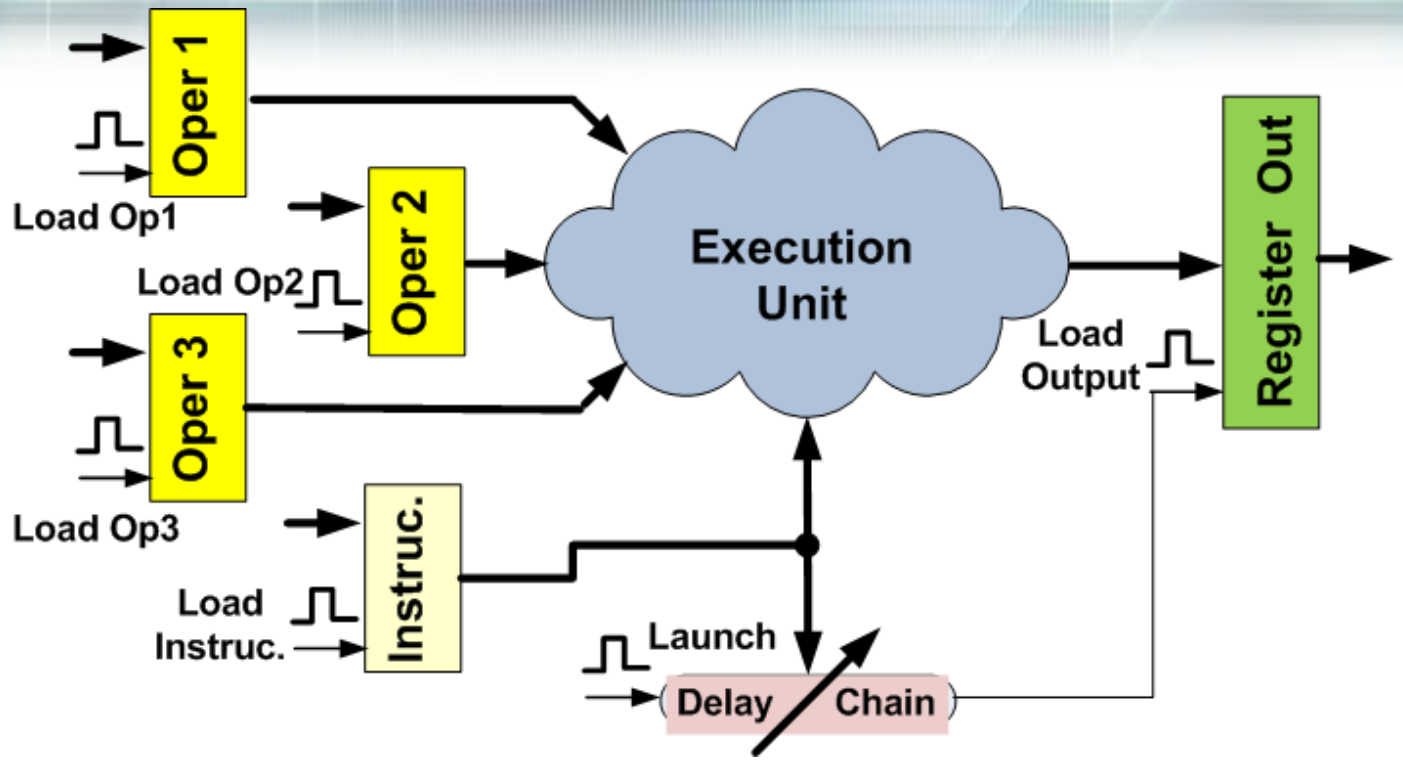
Traditional



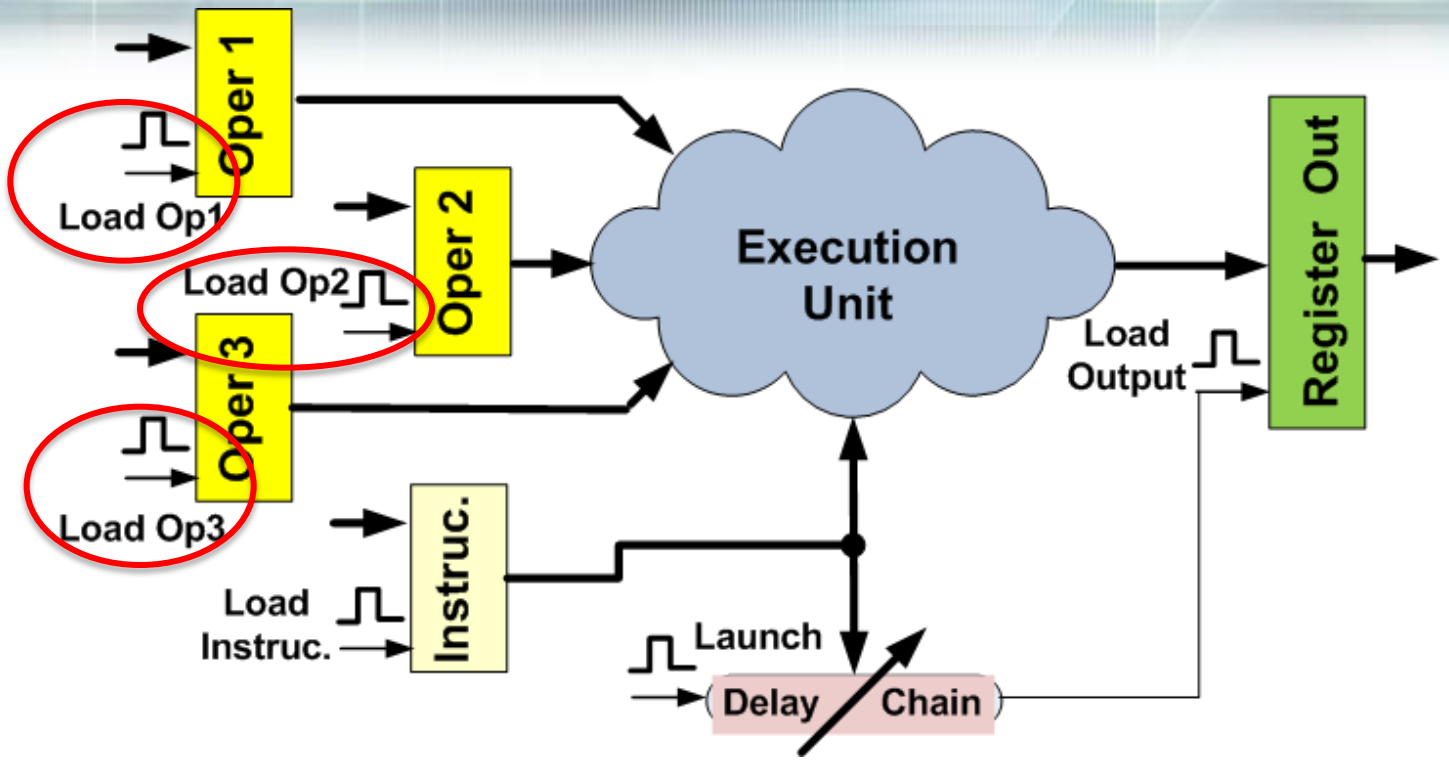
Octasic has modified the approach - no ACK but a rate limiter:

- simplified circuit
- no special silicon cell
- standard design tools

EXAMPLE: OCTASIC SIMPLIFIED EXECUTION UNIT

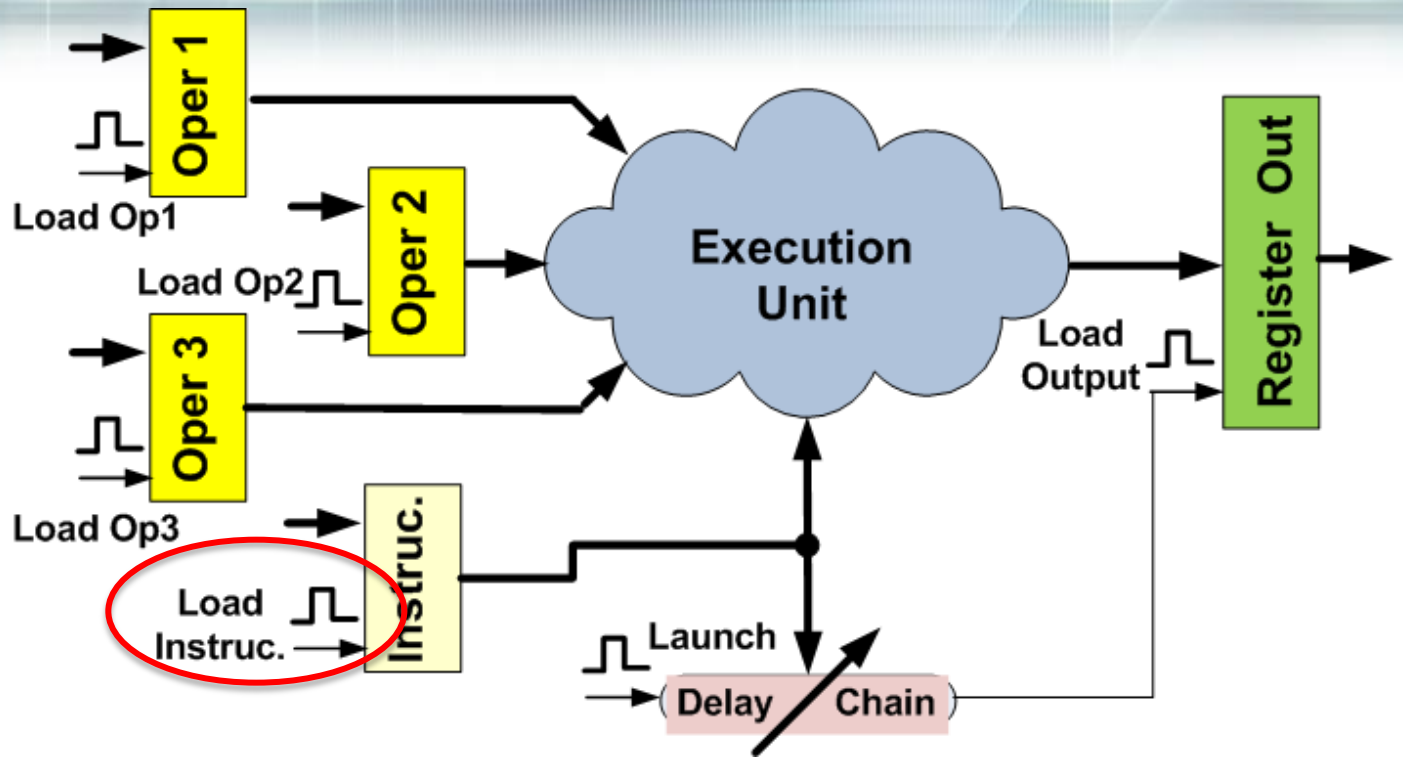


OCTASIC SIMPLIFIED EXECUTION UNIT



- The operand state registers are asynchronously loaded

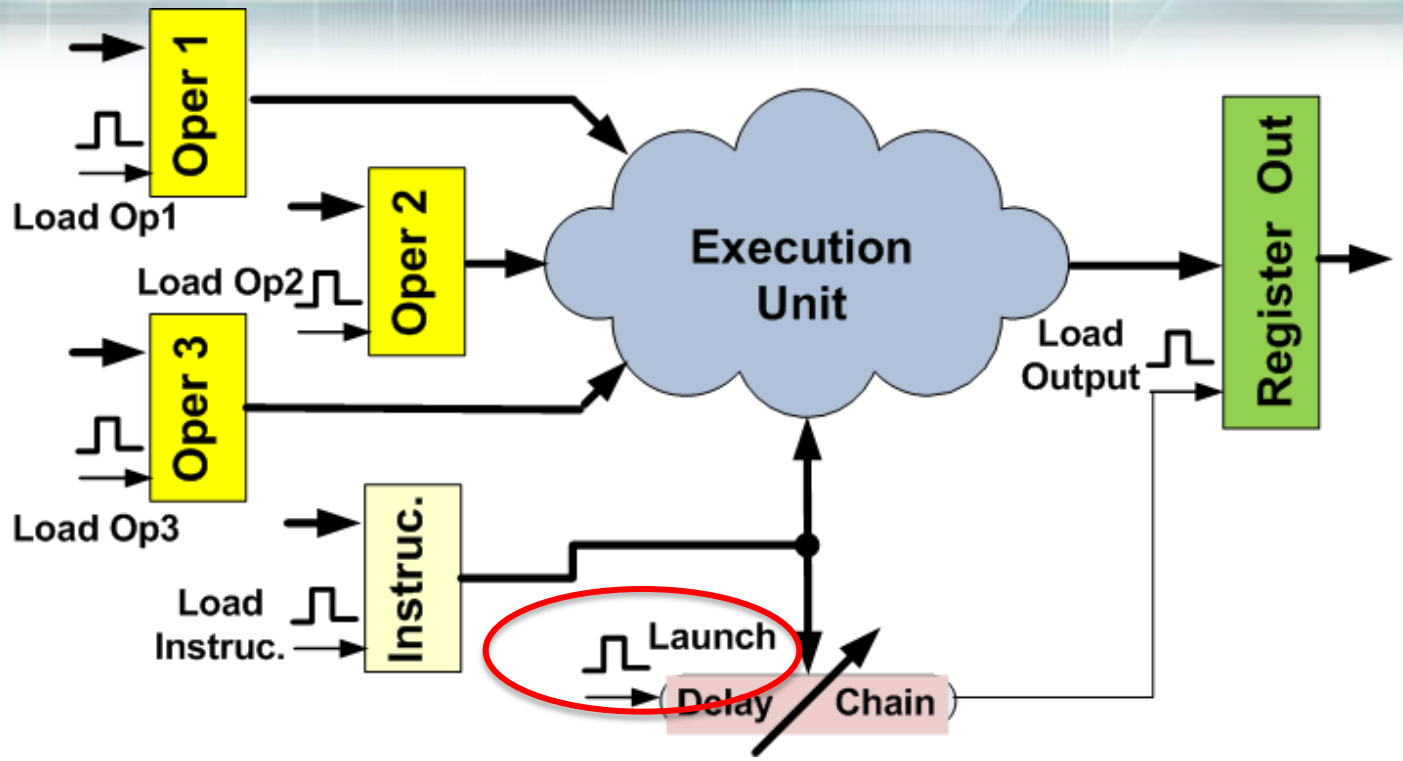
OCTASIC SIMPLIFIED EXECUTION UNIT



- The operand state registers are asynchronously loaded
- The instruction state register is asynchronously loaded

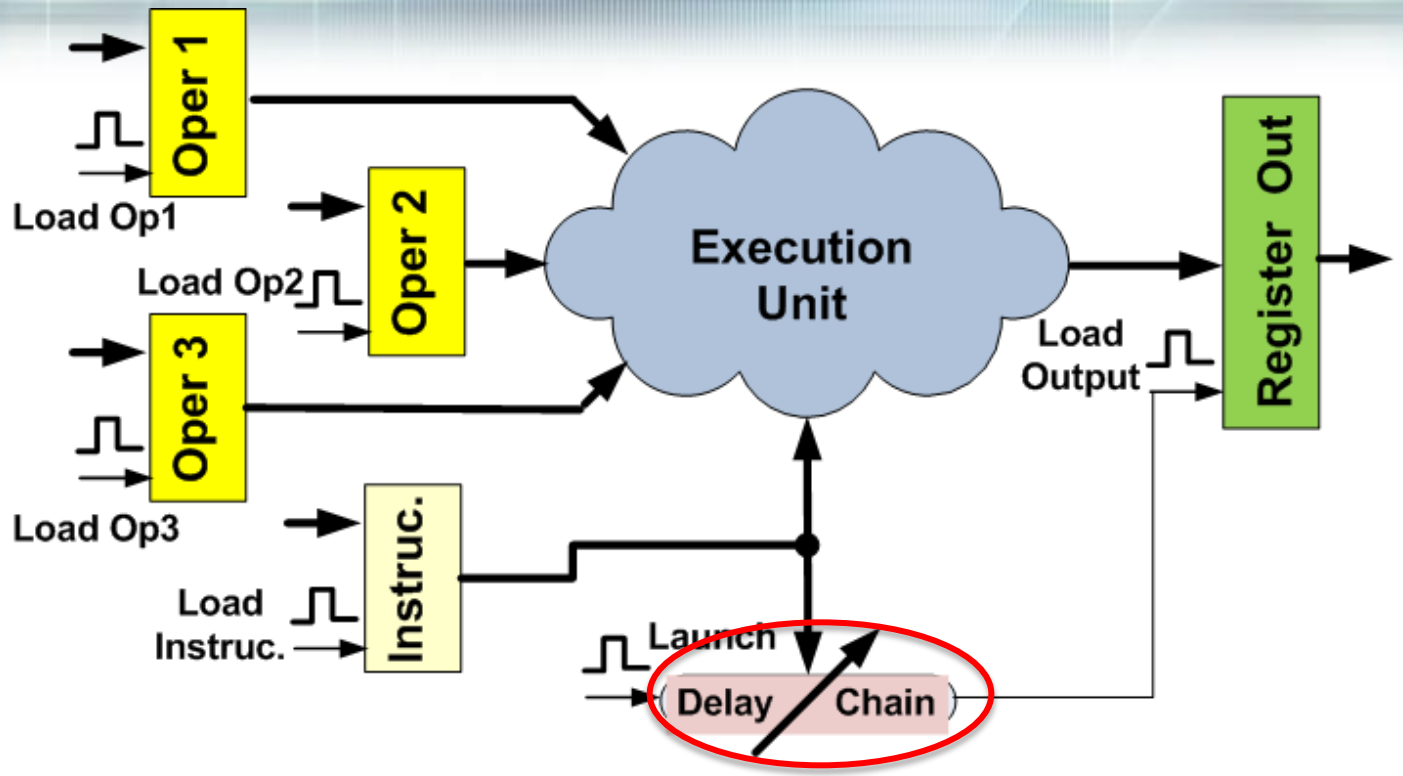
octasic
semiconductor

OCTASIC SIMPLIFIED EXECUTION UNIT



- The operand state registers are asynchronously loaded
- The instruction state register is asynchronously loaded
- When ready (input registers loaded & output register released) a launch pulse is generated

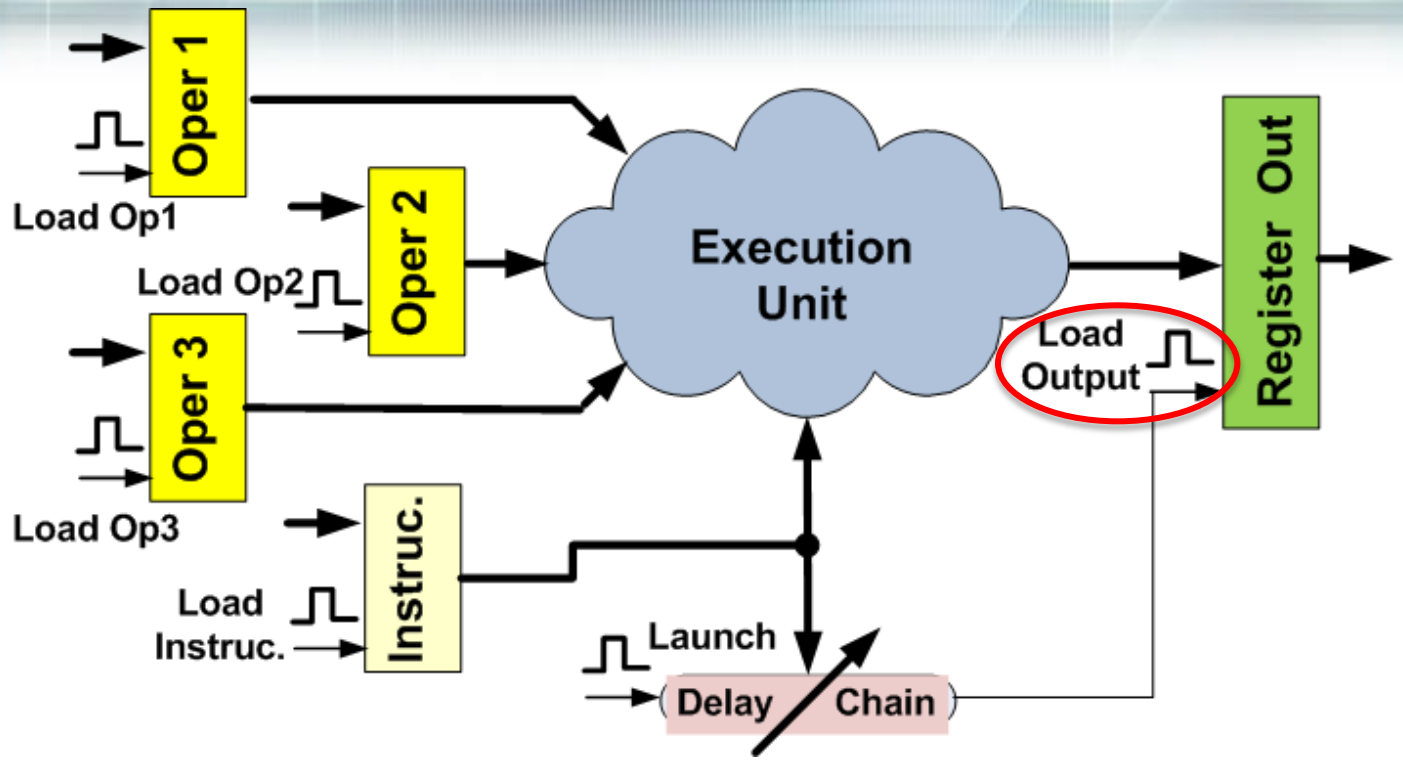
OCTASIC SIMPLIFIED EXECUTION UNIT



- The operand state registers are asynchronously loaded
- The instruction state register is asynchronously loaded
- When ready (input registers loaded & output register released) a launch pulse is generated
- Delay chain timing is modulated according to instruction

octasic semiconductor

OCTASIC SIMPLIFIED EXECUTION UNIT



- The operand state registers are asynchronously loaded
- The instruction state register is asynchronously loaded
- When ready (input registers loaded & output register released) a launch pulse is generated
- Delay chain timing is modulated according to instruction
- Output state register is asynchronously loaded with result of instruction

octasic semiconductor

BENEFITS OF OCTASIC'S APPROACH

Uses only standard ASIC library elements

- No custom cell
- Ease of porting - from one silicon node to the next / from one vendor to another

Can use standard CAD tools and concepts

- To facilitate sign-off
- To facilitate staff conversion training

Uses standard ATPG tools and principles

- Ensures manufacturability and reliability

octasic semiconductor

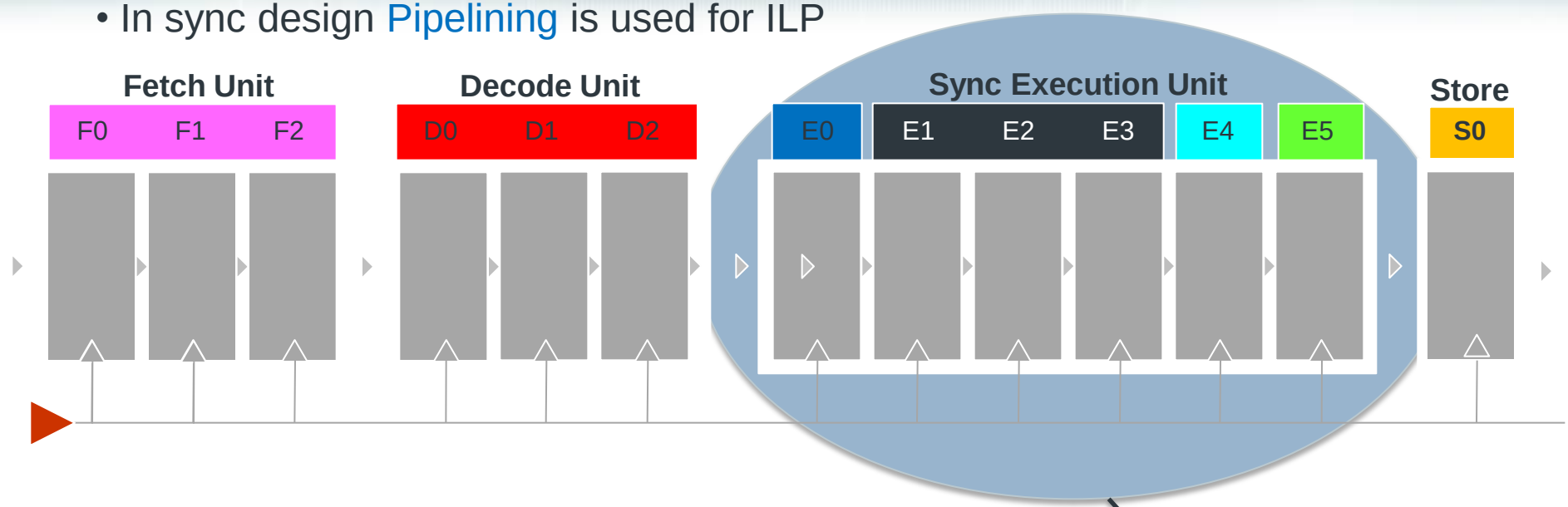
CONTENTS

octasic semiconductor

SYNC VS ASYNC PROCESSOR IMPLEMENTATION

Instruction Level Parallelism (ILP) is key to increase computing performance:

- In sync design Pipelining is used for ILP

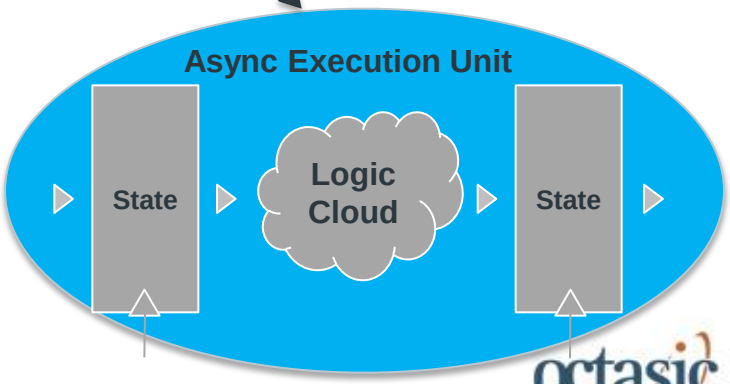


Fetch
Decode
Reg Reads
Execute
Branch
Output Write
Store

MEM load/store not show

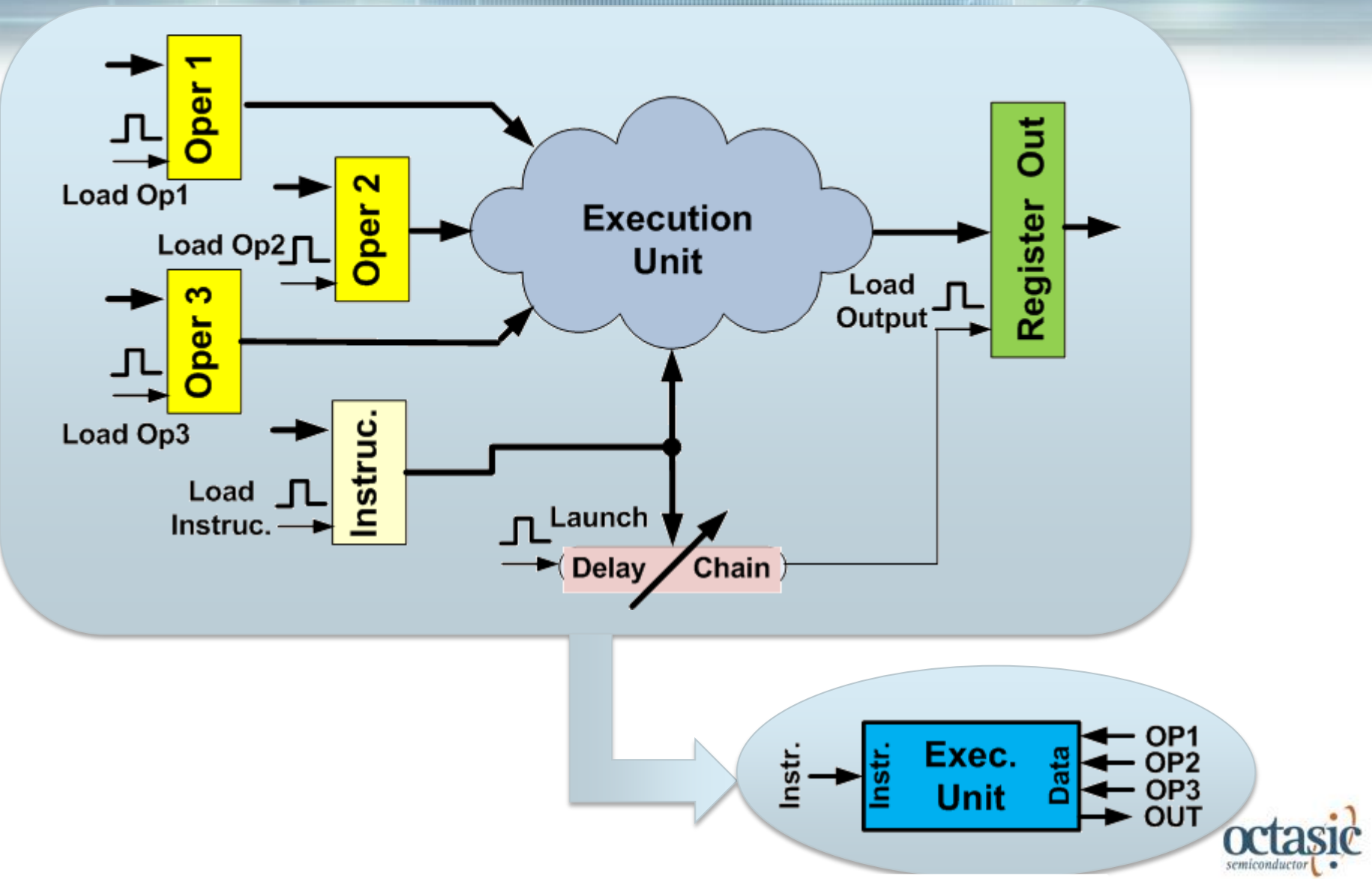
Conversion Sync => Async:

- Each **unit functionality** is maintained
- Without pipelining the async version will be slower
- **How can async architecture implements ILP to get back performance?**



octasic semiconductor

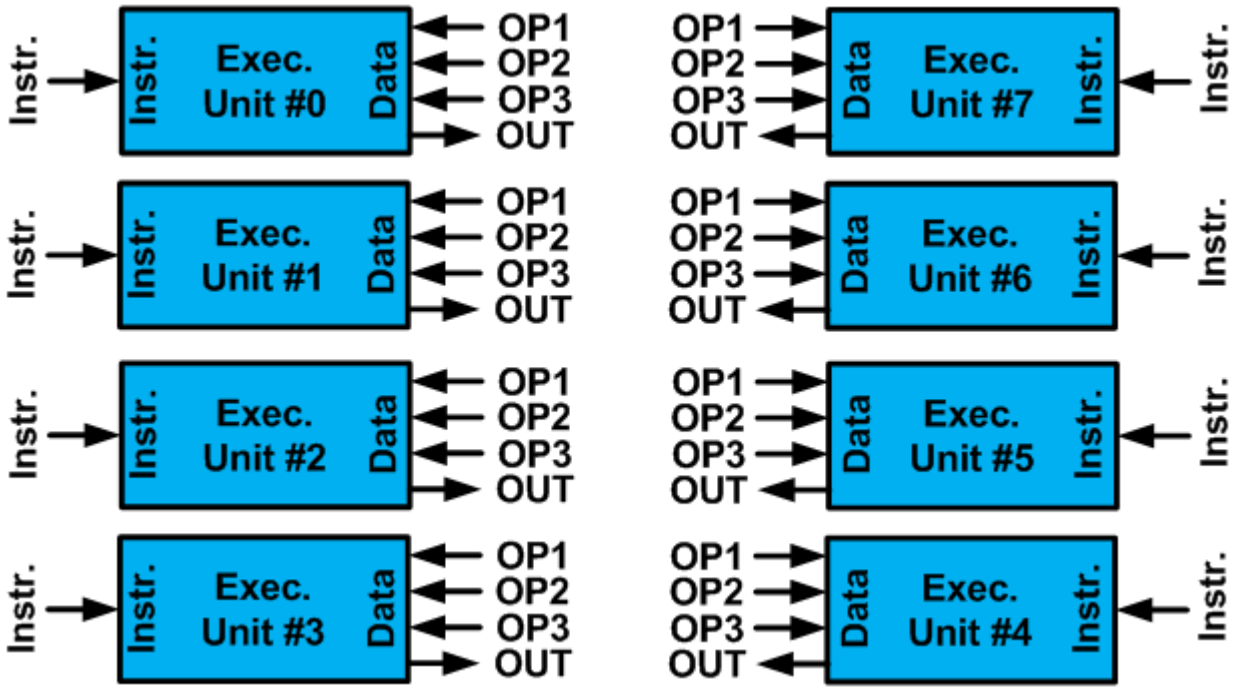
ASYNC ILP IMPLEMENTATION (1)



octasic
semiconductor

ASYNC ILP IMPLEMENTATION (2)

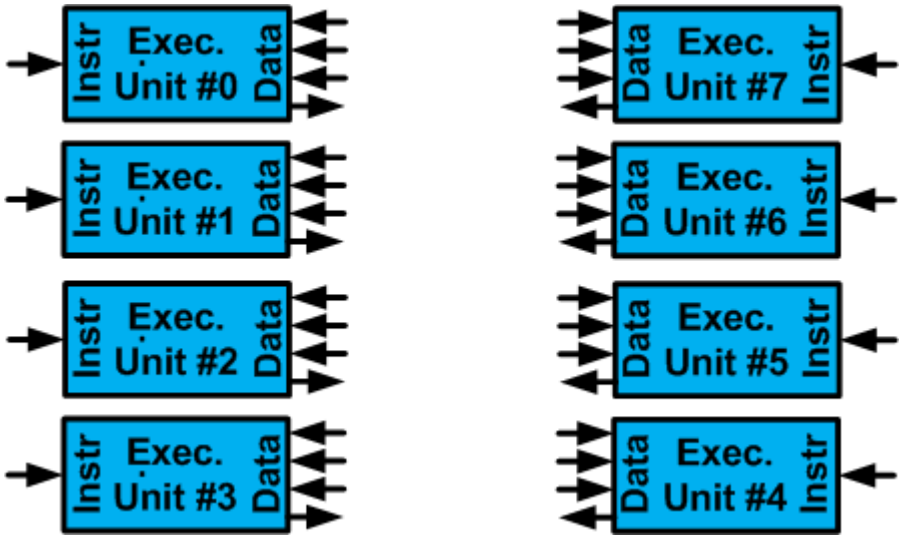
To multiply the computing power or capacity of our processor we could use multiple Exec Units (EUs) operating in parallel, **... much like is done in multi-processor and multi-core designs!**



Now how can we transparently weave together those EUs ...
....so they behave as one CPU?

ASYNC PROCESSOR ARCHITECTURE (2)

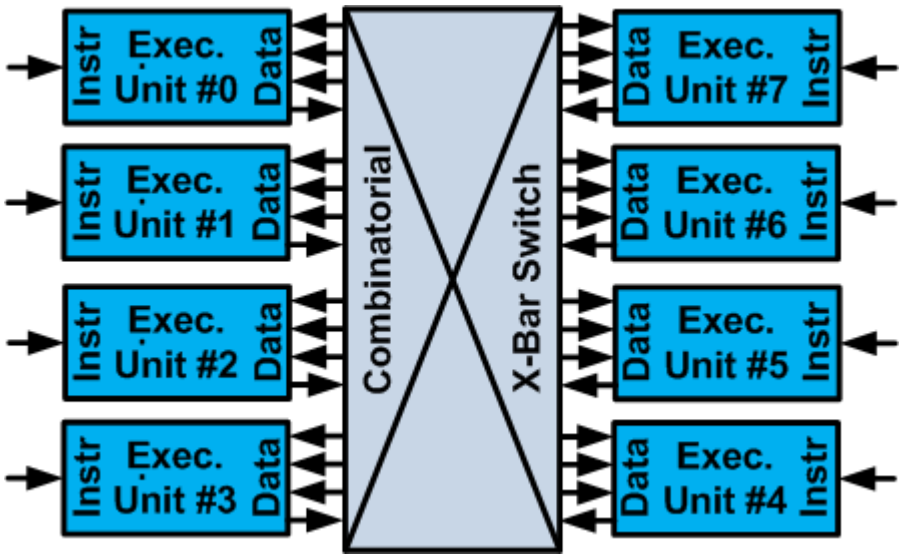
- Starting with the 8 execution units …



octasic semiconductor

ASYNC PROCESSOR ARCHITECTURE (3)

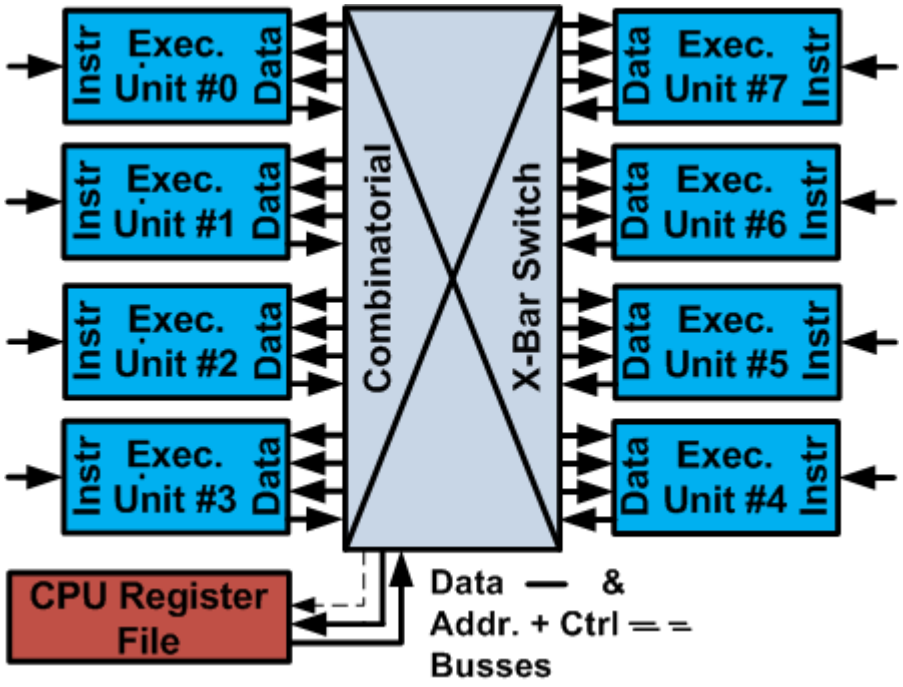
- **Adding a non-blocking combinatorial X-Bar switch to:**
  - **connect the execution units data paths among themselves, and**
  - **with external resources – register file, memory, etc.**



octasic semiconductor

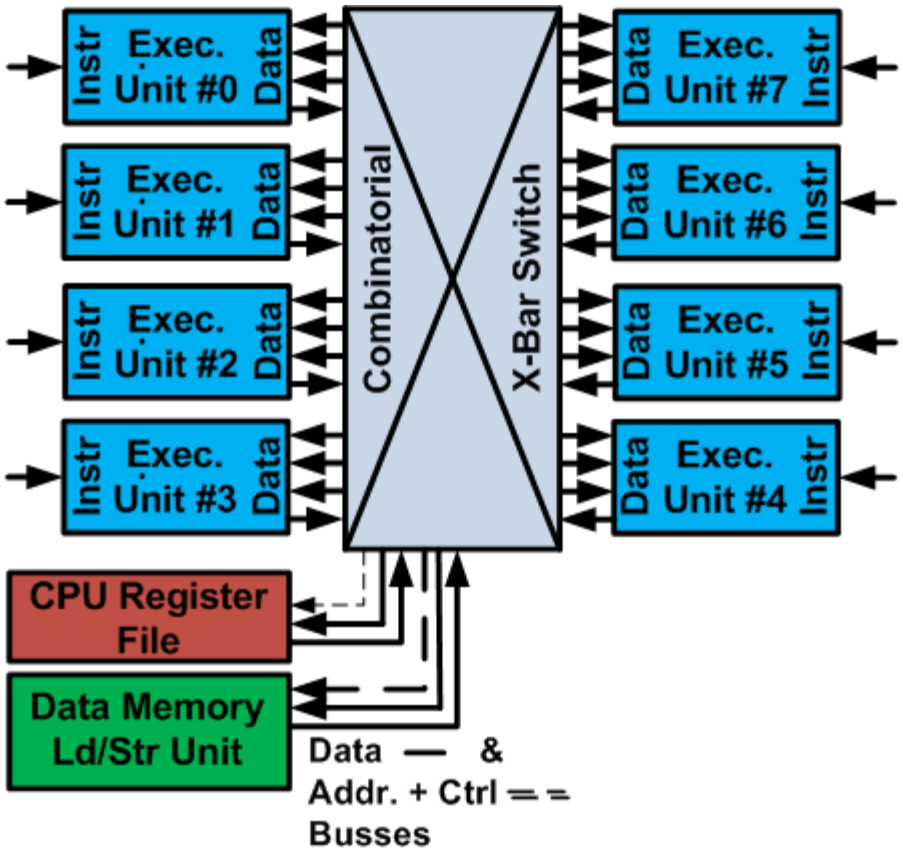
ASYNC PROCESSOR ARCHITECTURE (4)

- **Adding a CPU Register File to implement a load/store design:**



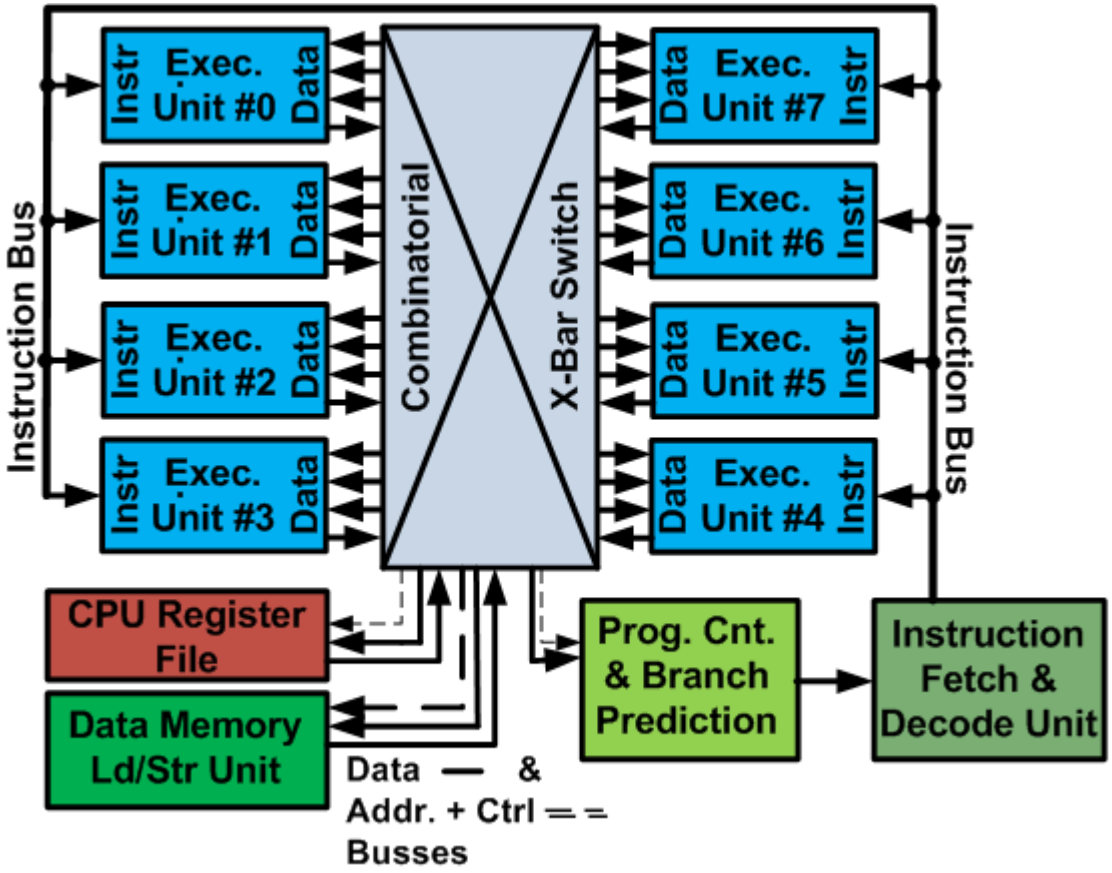
ASYNC PROCESSOR ARCHITECTURE (5)

- Adding a Data Memory Load/Store unit
  - to be able to load/store memory data into/from the CPU (registers)



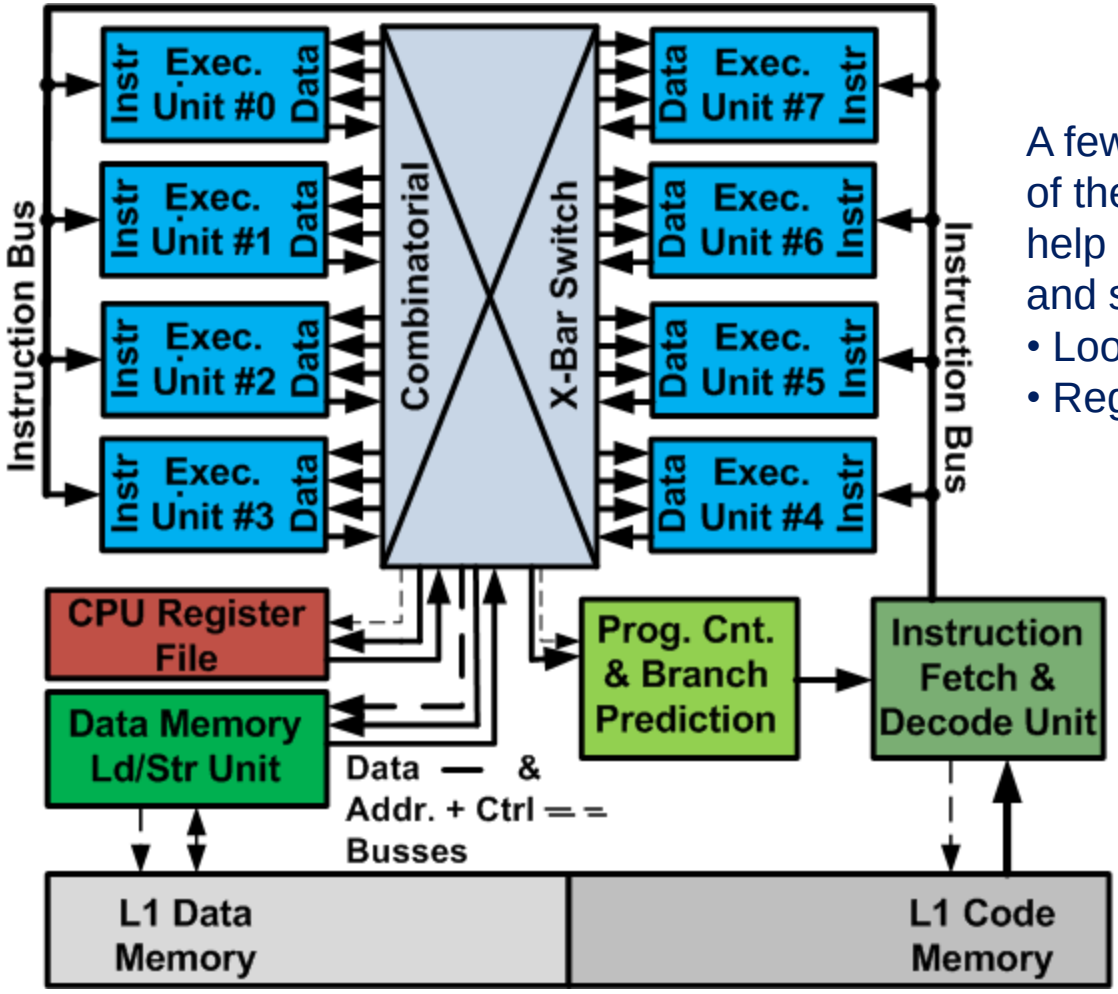
ASYNC PROCESSOR ARCHITECTURE (6)

- Adding a **Program Counter Control** unit inc a branch predictor;
- Coupled with an **Instruction Fetch & Decode** Unit
  - to be able to load instructions into the execution units



ASYNC PROCESSOR ARCHITECTURE (7)

- **Adding L1 Memory accessible for:**
  - **Data, or**
  - **Code**



A few given characteristics of the architecture to help increase performance and save power:

- Loops
- Register Shadowing

octasic semiconductor

CONTENTS

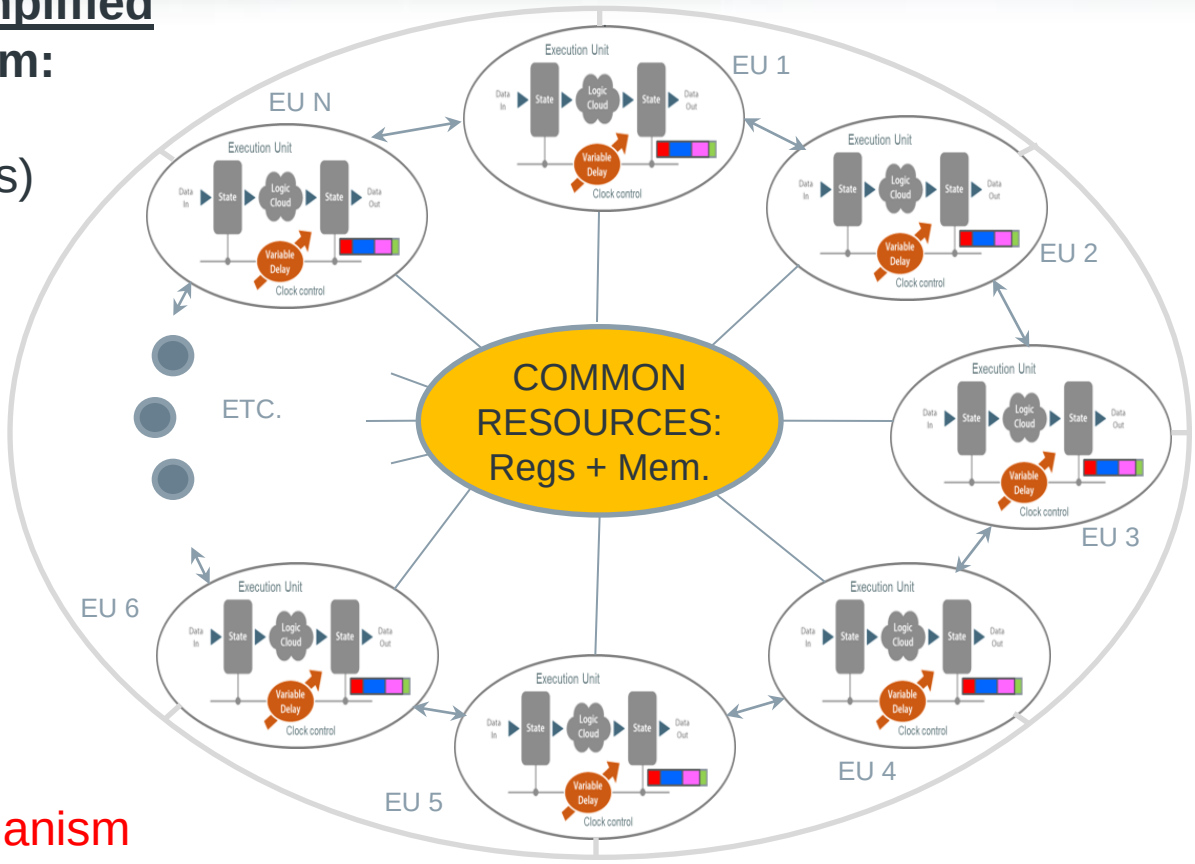
octasic semiconductor

OPERATION AND SYNCHRONIZATION (1)

This is an alternate <u>simplified</u> processor block diagram:

- the execution units (EUs) are mapped in a ring like fashion

- the EUs have access to common resources:
  - Register File
  - Data Memory
  - Code Memory
  - X-Bar
  - PC Control Logic



- a synchronization mechanism is required to arbitrate and avoid conflict in the access of the EUs to the common resources

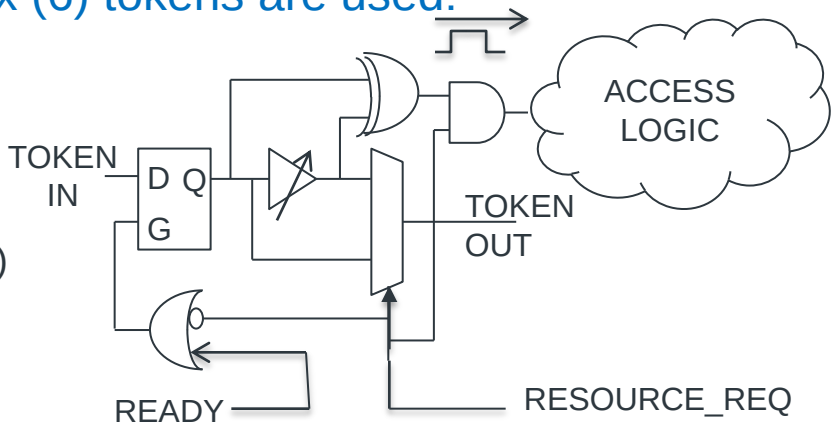
octasic semiconductor

OPERATION AND SYNCHRONIZATION (2)

The operation of a synchronous processor is generally centrally controlled.

This asynchronous processor has a fully distributed control structure:

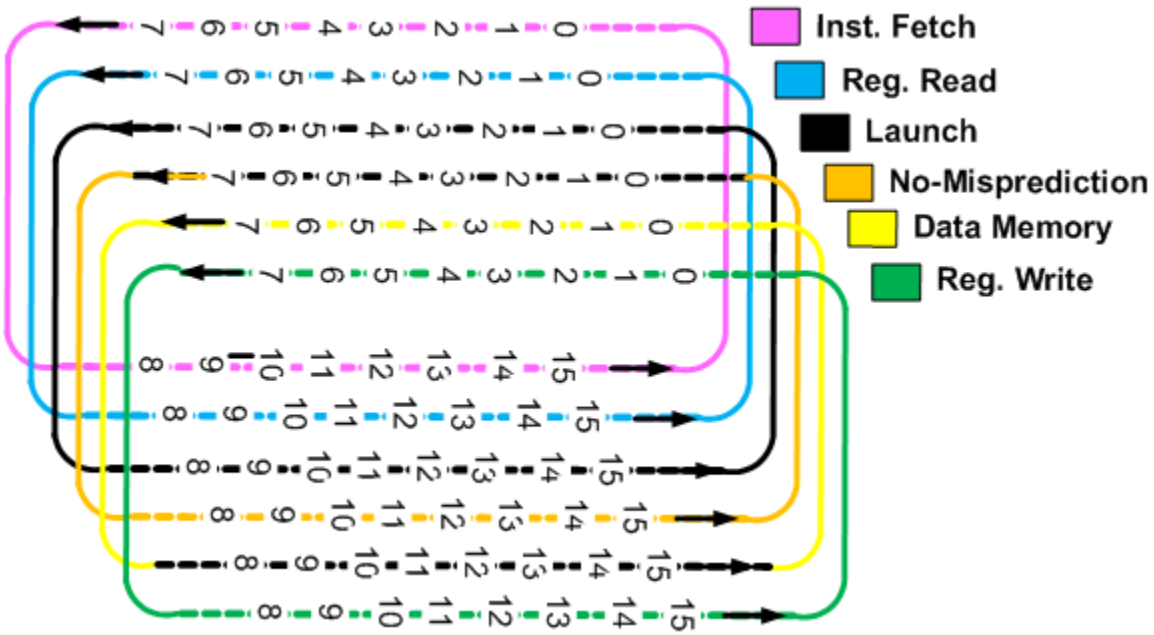
- Control is exercised individually by each Execution Unit (EU)
- Control tokens are passed asynchronously among the EUs in a ring fashion to synchronize accesses to common resources and avoid conflicts
- In the simplified model discussed herein, six (6) tokens are used:
  - Instruction Fetch Token
  - Register Read Token
  - Launch Execution Token (X-Bar, Reg Ready)
  - No Mis-Prediction Token (PC & Write Commit)
  - Data Memory Token (Rd or Wr)
  - Register Write Token



OPERATION AND SYNCHRONIZATION (3)

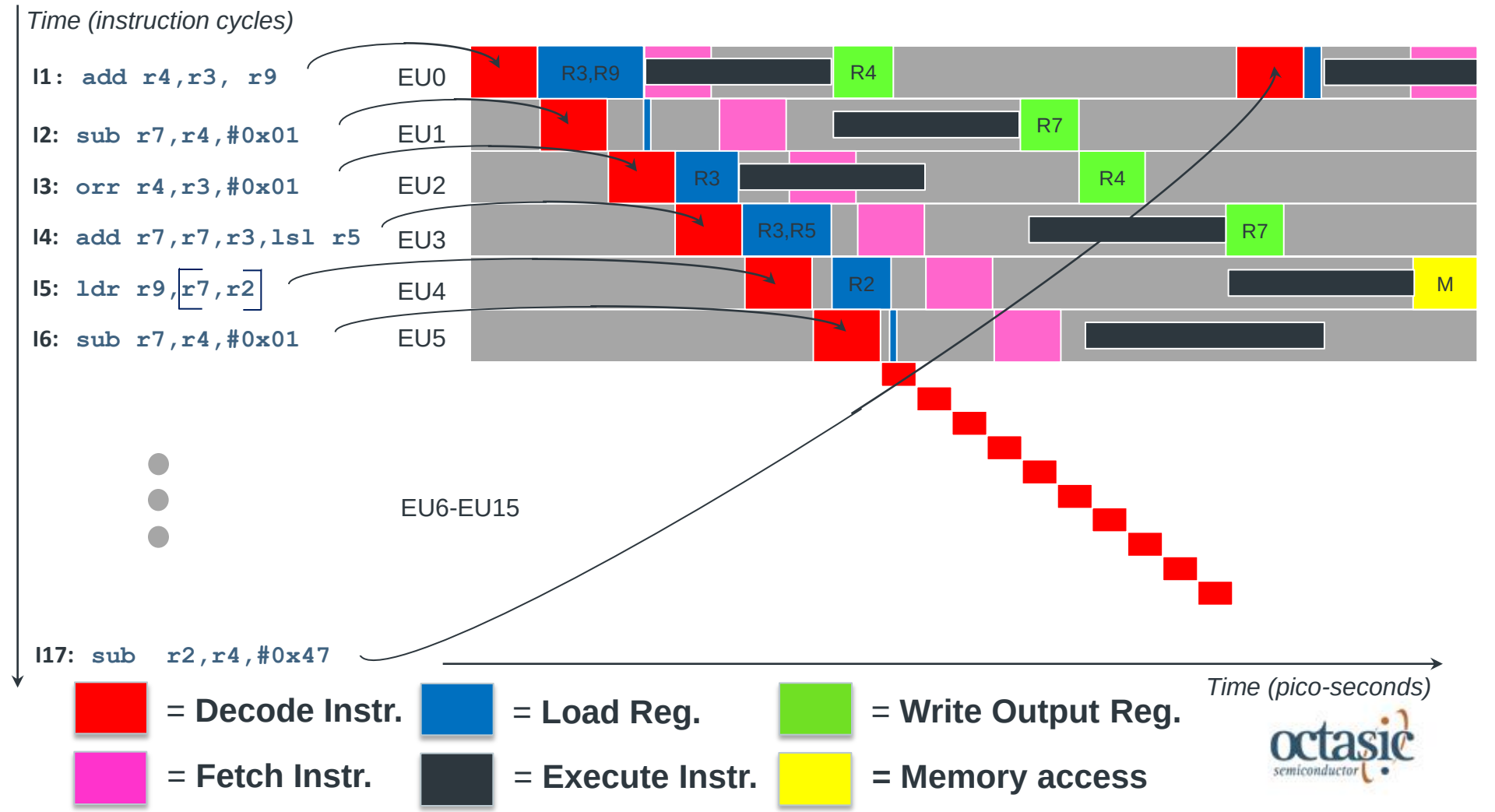
Asynchronous control tokens are used to control and synchronize the overall operation of the processor.

- Control tokens are passed from one EU to the next in a ring fashion.
- When a token is owned by an EU it can use it to request services (via Req pulses)
- When a service request is sent and a certain time has elapsed and certain conditions are met, or when the EU does not need the token (resource) the token is passed to the next EU.
- On start up or after a flush (wrongly predicted branch), all tokens are assigned to the same EU.

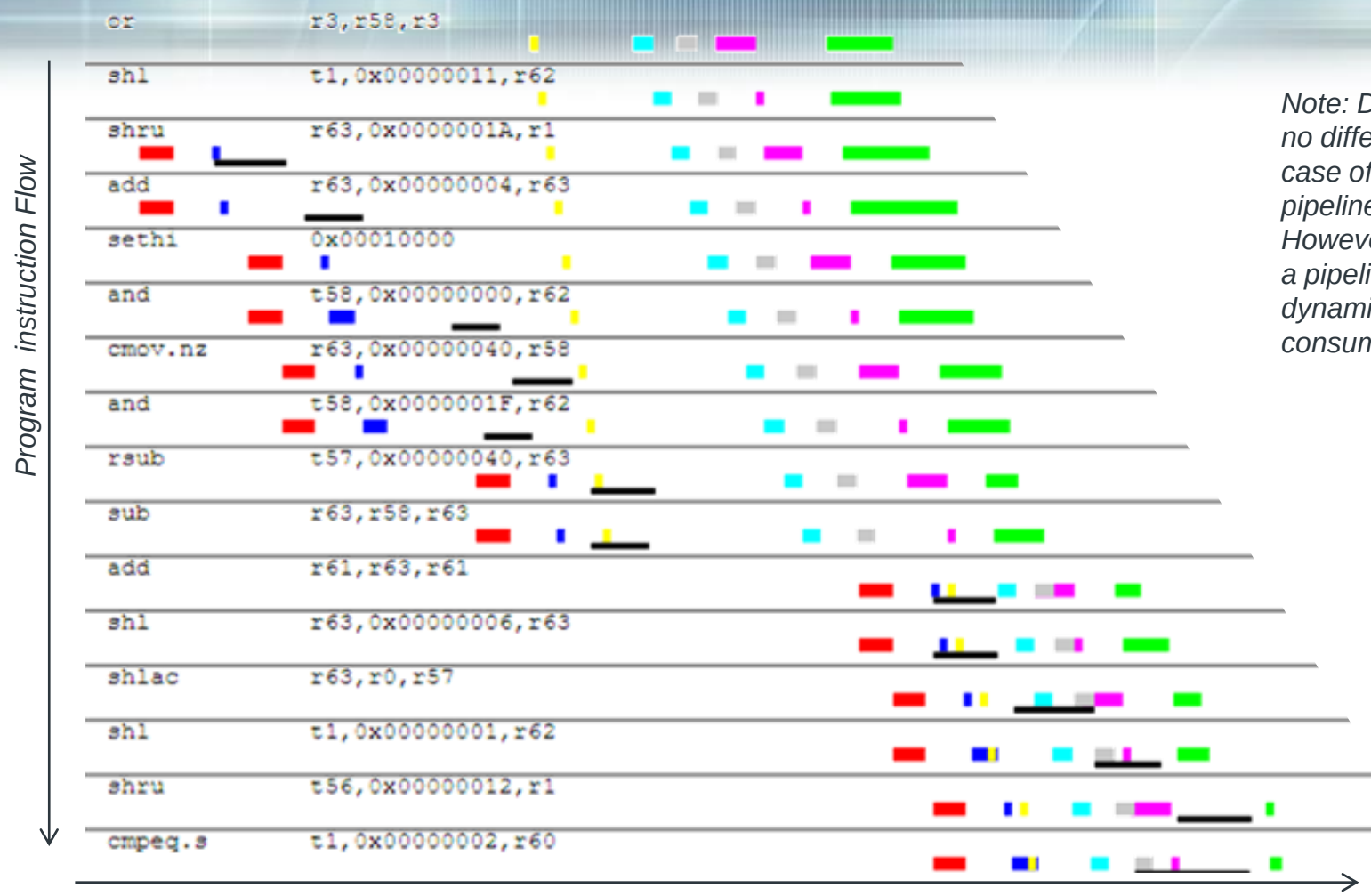


PROCESSOR OPERATION – SIMPLIFIED ILP (1)

Assuming the operation of the Execution Units and resources (registers, memory, …) are somehow synchronized, here is the flow of instructions overlap that would result in the processor; hence realizing the Instruction Level Parallelism (ILP) mechanism to boost performance



PROCESSOR OPERATION ILP: REAL-WORLD EXAMPLE (2)



Note: Dependencies are no different than in the case of synchronous pipelined processors. However in the event of a pipeline stall, no dynamic power is consumed.

= Decode Instr. **= Load Reg.** **= Write Output Reg.**

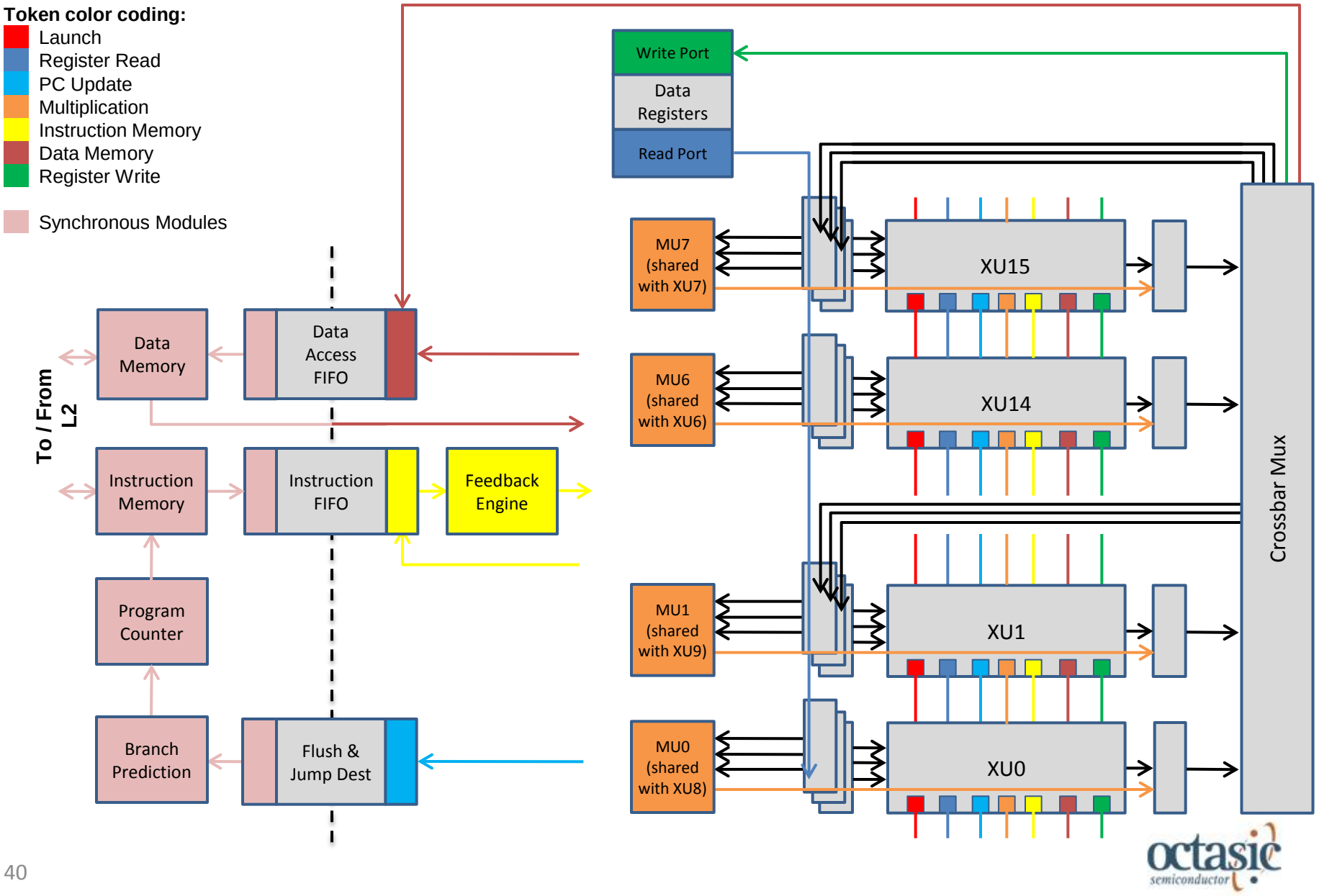
= Fetch Instr. **= Execute Instr.** **= Memory access**

octasic semiconductor

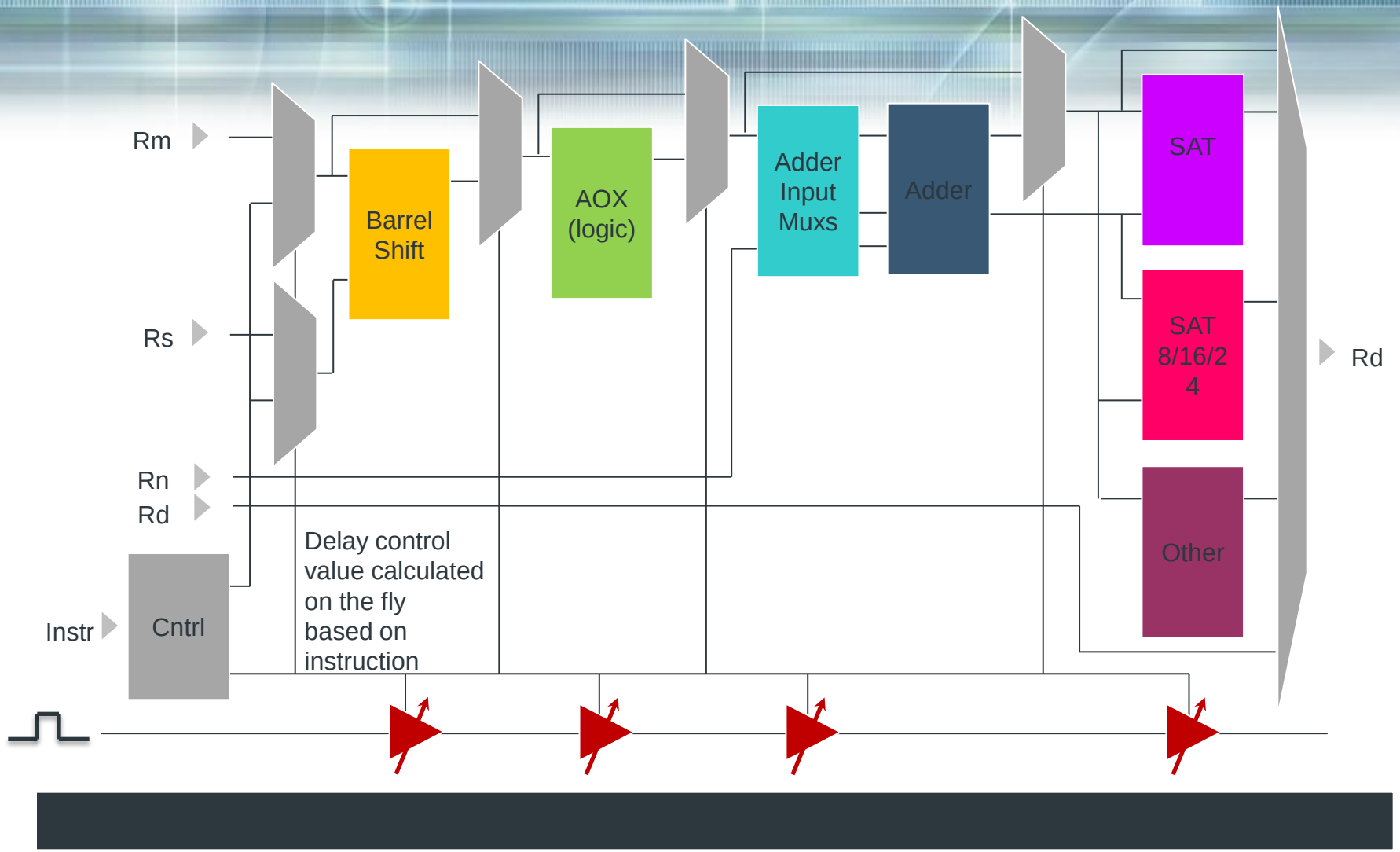
CONTENTS

octasic semiconductor

ARM BLOCK DIAGRAM



Typical ARM Execution Unit (EU) Implementation



ARM SILICON LAYOUT (TOP)

iCache
L1 Instruc. Memory
(32KB)

dCache
L1 Data Memory
(32KB)

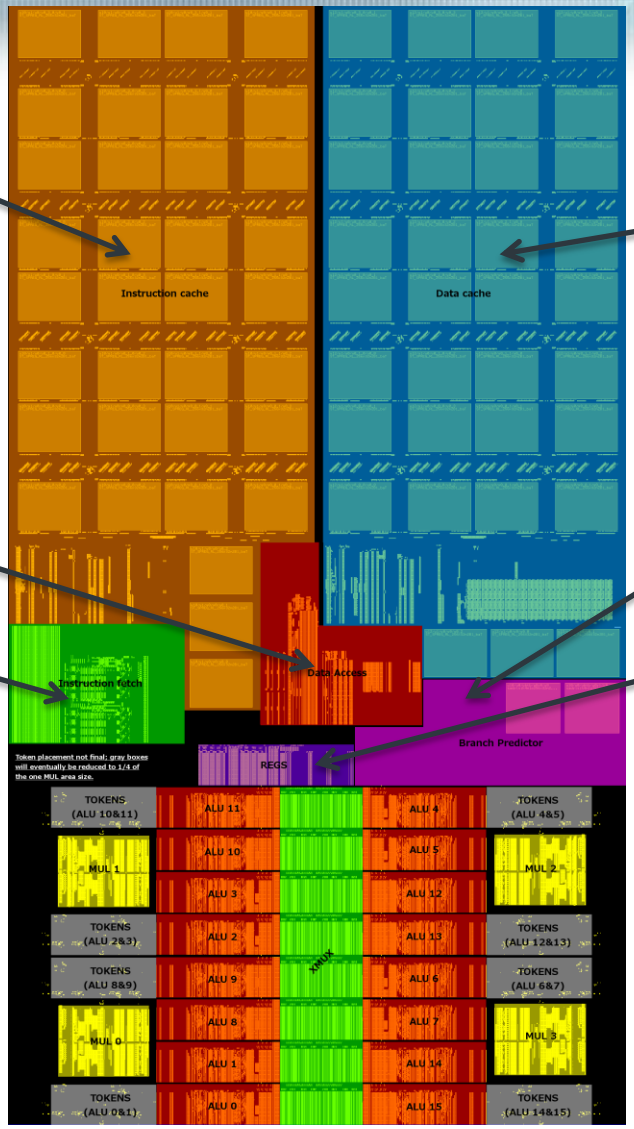
Data Access

Instruction Fetch

Branch Predictor

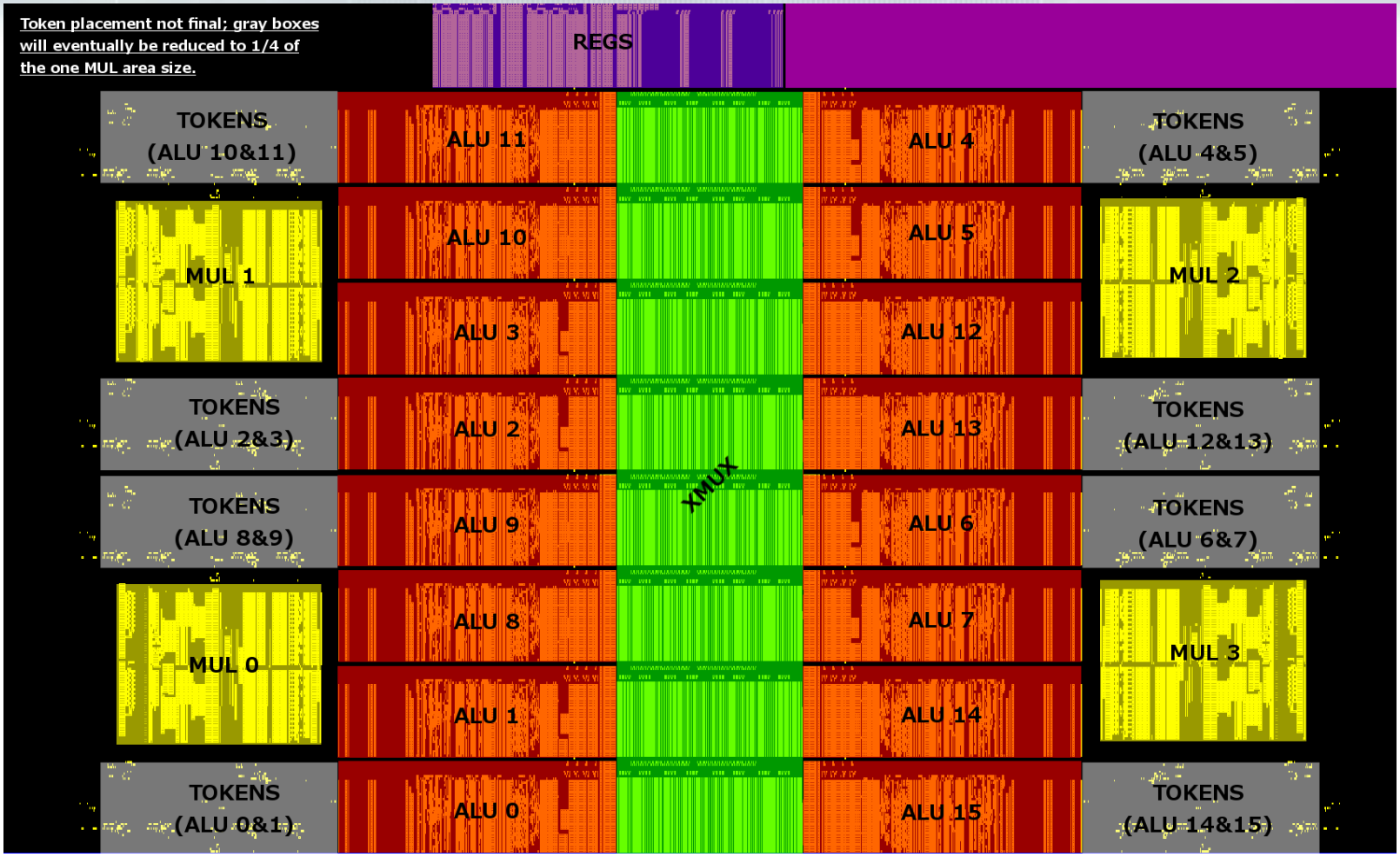
Registers

ARM Execution Core



octasic semiconductor

ARM SILICON LAYOUT (EXECUTION CORE ZOOM)



4 Mul + Tokens **8 UEs** **X-Bar** **8 UEs** **4 Mul + Tokens**

octasic semiconductor

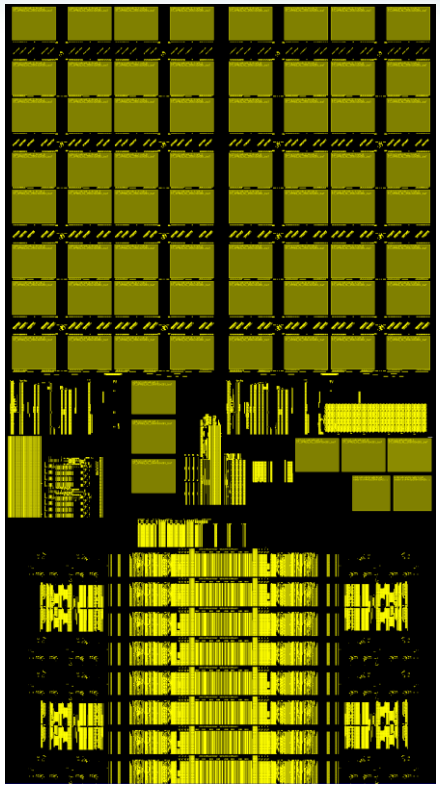
CONTENTS

- Perspective
- Background
- Processor Architecture and Operation
- **Performance Analysis**
- Conclusion

octasic
semiconductor

ARM LAYOUT SIZE (28NM)

Block	Area (μm^{2})	Qty	Area (μm^{2})
EU & Tokens	8700	16	139200
MUL	7600	8	30400
XBAR	38500		38500
Branch Predictor	12000		12000
iCache (32K) + Instruction Fetch	190000		190000
dCache (32K)	172000		172000
Mem Man + Reg Files	12600		12600
		Total	**594700**



*Note: These areas are extracted from the library which is based on drawn 32nm. Actual 28nm silicon size is smaller.

octasic semiconductor

ARM POWER BREAKDOWN (28NM)

Block	Power (mW)	% Dynamic
Central Mux + wires	2.402	3.85
Register module	1.654	2.34
Instruction fetch [synchronous]	4.867	25.0
ALU internals (including calculations)	4.343	6.96
Token modules	2.44	3.91
Everything else	3.332	5.34
Instruction cache [synchronous]	16.07	25.75
Data cache (25% loads) [synchronous]	14.54	23.30
Branch Prediction [synchronous]	12.75	20.43
Total Dynamic Power	**62.4**	
Leakage	**12.0**	
Total power	**74.4**	

Typical, @25C

Executing Dhrystone @ 2,000 DMIPS

octasic semiconductor

ARM PRELIMINARY RESULTS SUMMARY

Simple ARM Core Compatible Implementation (~A8 equivalent)

- **Technology:** 28nm LP STM
- **Performance:** 2,000 DMIPS
- **Area** (inc. 32KB L1 code and 32KB L1 data): ~0.6 mm2
- **Power:** ~75mW @ 2,000DMIPS

This is believed to be good from an area perspective and very good from a power consumption perspective:

~½ the power consumption of equivalent synchronous implementation

octasic semiconductor

CONTENTS

- Perspective
- Background
- Processor Architecture and Operation
- Performance Analysis
- **Conclusion**

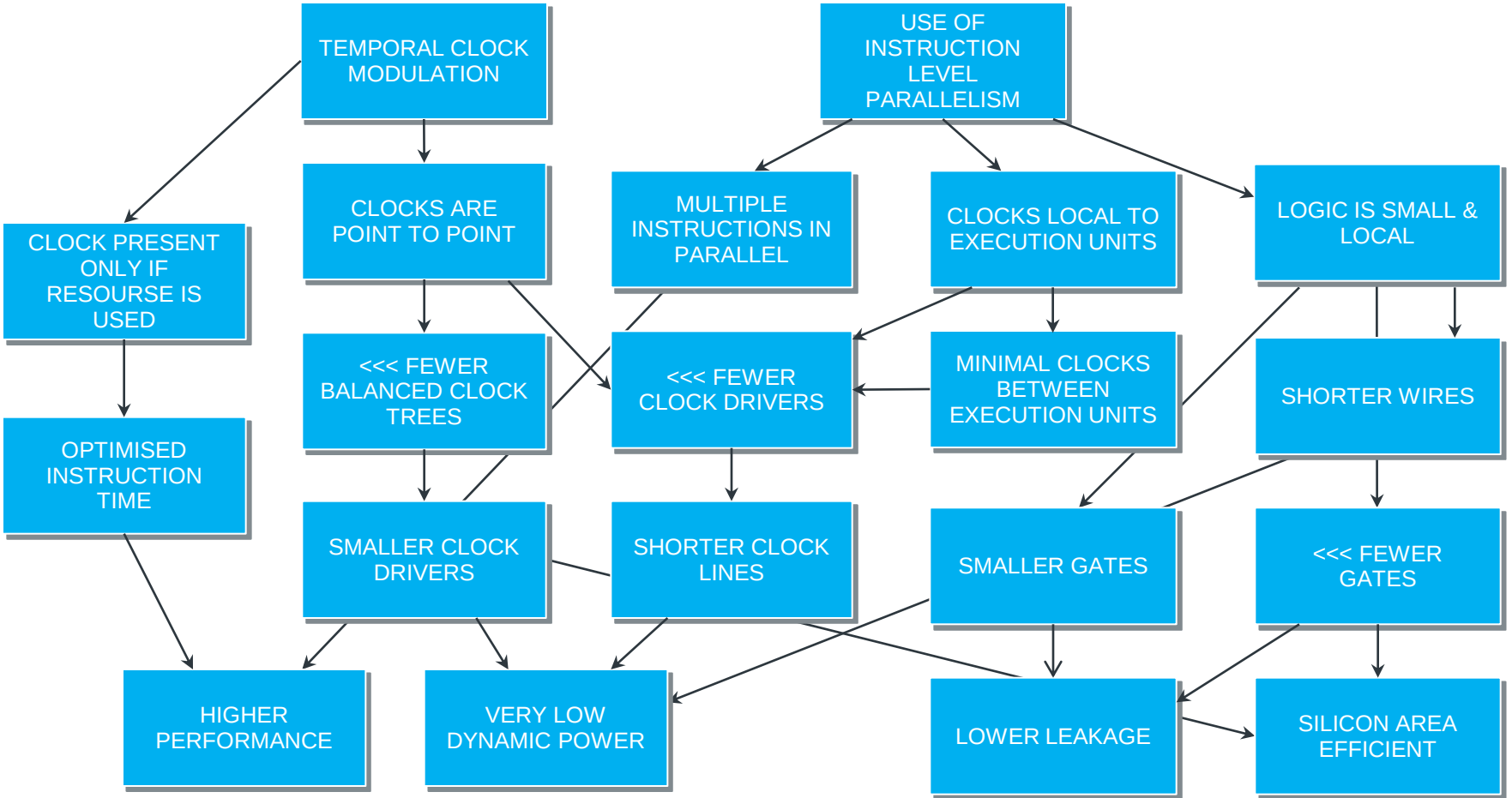
octasic semiconductor

POWER REDUCTION SOURCES

- No balanced clock trees. Clocks are point to point and not skew sensitive. Therefore smaller gates, more HVT gates and shorter wires used
- No critical paths due to frequency constraints, therefore no need to optimize with large gates and can use HVT gates
- Proximity of pipeline stages (each stage only connected to previous or next). Therefore smaller gates, use of HVT gates and shorter wires
- Clock edges are only generated when a resource is used. No wasted edges (ex: no power use during pipeline stall)
- All of the above applies to: clocks, logic and data paths
- Overall results in >> 80% HVT usage

octasic semiconductor

ADDITIONAL POWER EFFICIENCIES



octasic semiconductor

CONCLUSION

- Processing power efficiency is becoming more and more important nowadays .
- It will be imperative to push back “dark silicon” phenomenon as silicon power improvements lag performance gains.
- The application of a practical Asynchronous processor micro-architecture can improve power efficiencies of commercial general purpose processors by ~2x or more and can help with this problem.

octasic semiconductor

CONCLUSION

Thank you!

Michel Laurence
michel.laurence@octasic.com

octasic semiconductor