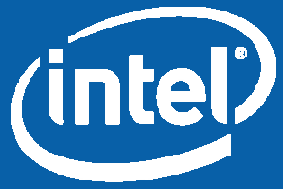
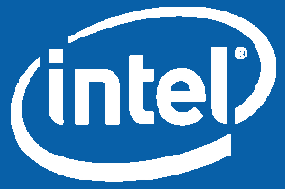


Accelerating Architecture Research

Joel Emer, Ph.D.

Visiting Faculty	Intel Fellow	Principal Investigator
CSG Group	VSSAD Group	RAMP Project
MIT	Intel	

Legal Disclaimer



INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL® PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL® PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT. INTEL PRODUCTS ARE NOT INTENDED FOR USE IN MEDICAL, LIFE SAVING, OR LIFE SUSTAINING APPLICATIONS.

Intel may make changes to specifications and product descriptions at any time, without notice.

All products, dates, and figures specified are preliminary based on current expectations, and are subject to change without notice.

Intel, processors, chipsets, and desktop boards may contain design defects or errors known as errata, which may cause the product to deviate from published specifications. Current characterized errata are available on request.

This document may contain information on products in the design phase of development. The information here is subject to change without notice. Do not finalize a design with this information.

Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them.

Intel Corporation may have patents or pending patent applications, trademarks, copyrights, or other intellectual property rights that relate to the presented subject matter. The furnishing of documents and other materials and information does not provide any license, express or implied, by estoppel or otherwise, to any such patents, trademarks, copyrights, or other intellectual property rights.

Wireless connectivity and some features may require you to purchase additional software, services or external hardware.

Nehalem, Penryn, Westmere, Sandy Bridge and other code names featured are used internally within Intel to identify products that are in development and not yet publicly announced for release. Customers, licensees and other third parties are not authorized by Intel to use code names in advertising, promotion or marketing of any product or services and any such use of Intel's internal code names is at the sole risk of the user

Performance tests and ratings are measured using specific computer systems and/or components and reflect the approximate performance of Intel products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance.

Intel, Intel Inside, Pentium, Xeon, Core and the Intel logo are trademarks of Intel Corporation in the United States and other countries.

*Other names and brands may be claimed as the property of others.

Acknowledgements

Arvind – MIT

Michael Pellauer – MIT

Murali Vijayaraghavan – MIT

Michael Adler – Intel

Angshuman Parashar - Intel

ZhiHong Yu – Intel

Tao Wang – Intel

Derek Chiou – UT – Austin

Krste Asanovic - Berkeley

Scientific Research

Take hypothesis about environment

Design experiment

Run experiment and quantify

Interpret results

If necessary, create new hypothesis

Systems Research

Take hypothesis about environment

Design experiment

Run experiment and quantify

Interpret results

If necessary, create new hypothesis

Systems Research

Take hypothesis about environment

Design Experiment - pick baseline design and workload

Run experiment and quantify

Interpret results

If necessary, create new hypothesis

Systems Research

Take hypothesis about environment

Design Experiment - pick baseline design and workload

Run experiment and quantify – run model or measure hardware

Interpret results

If necessary, create new hypothesis

Systems Research

Take hypothesis about environment

Design Experiment - pick baseline design and workload

Run experiment and quantify – run model or measure hardware

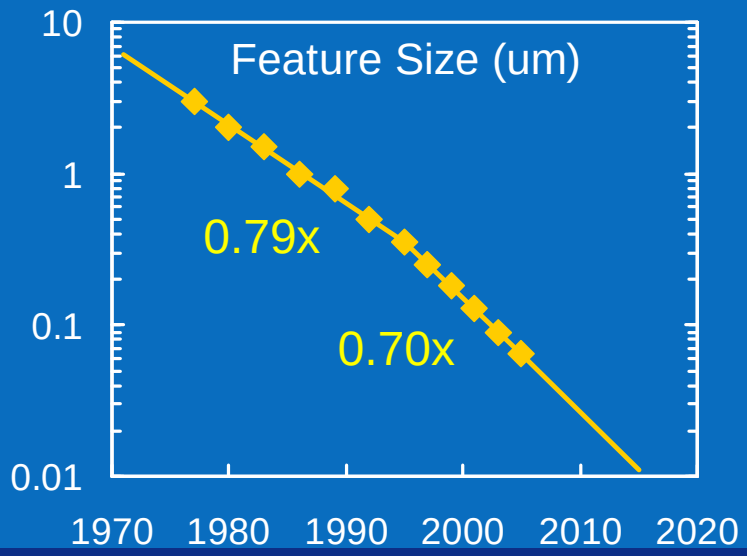
Interpret results

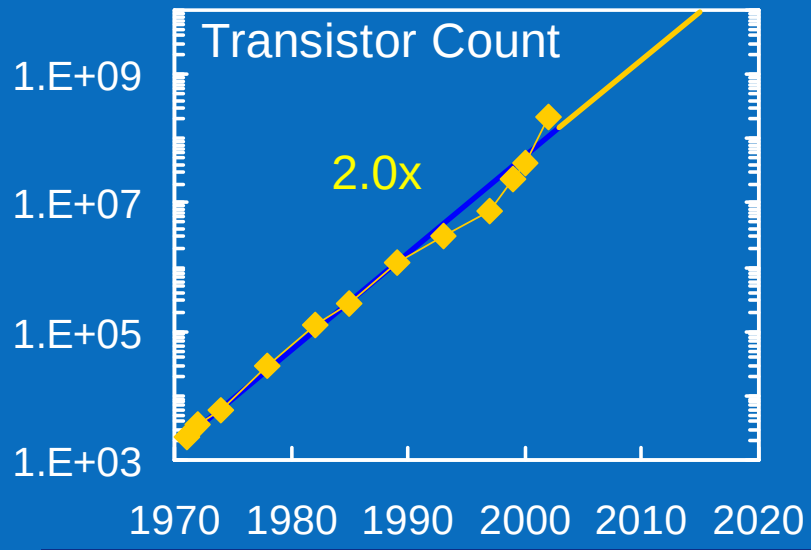
If necessary, propose new design

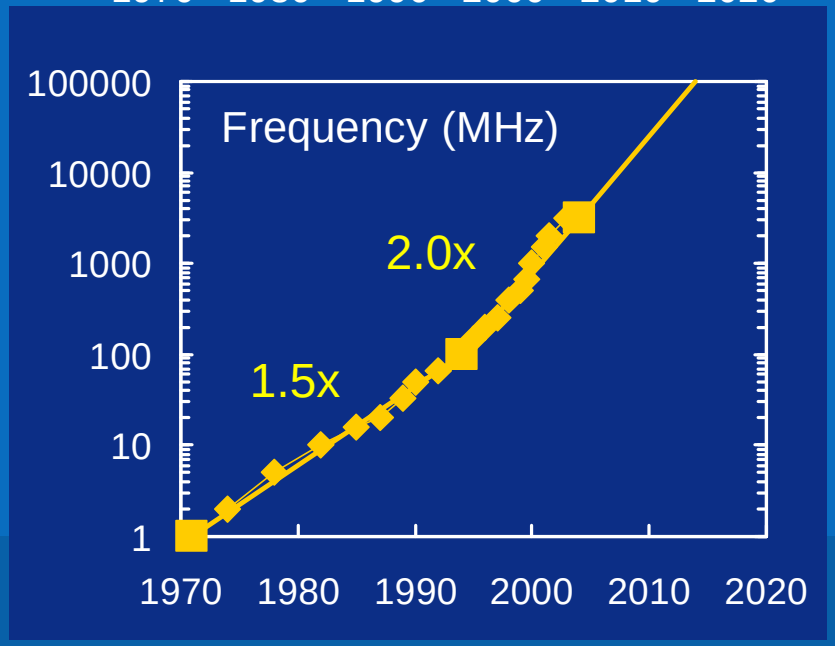
Outline

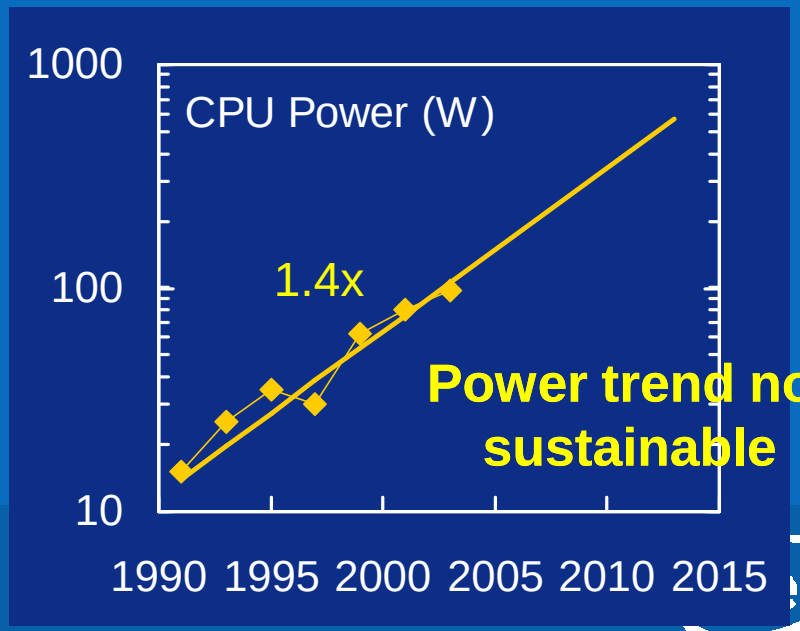
- An architecture research challenge
- A possible way to address that challenge
- Making that approach feasible

Moore's Law at Intel

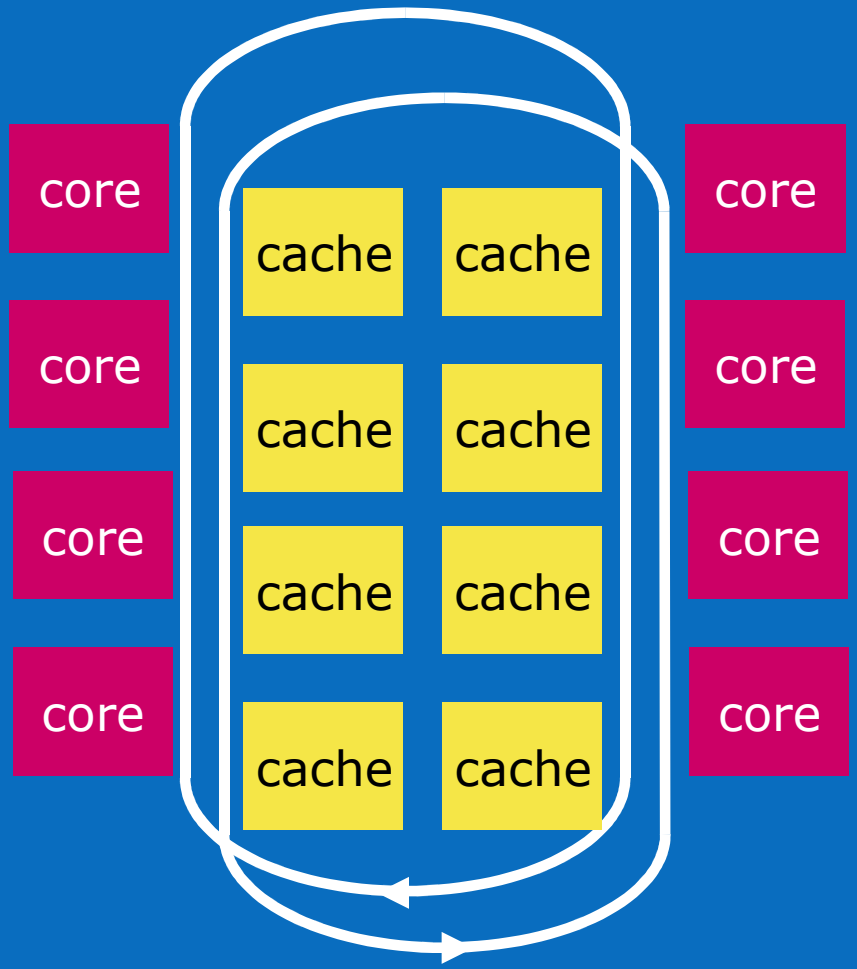








Many-core processors

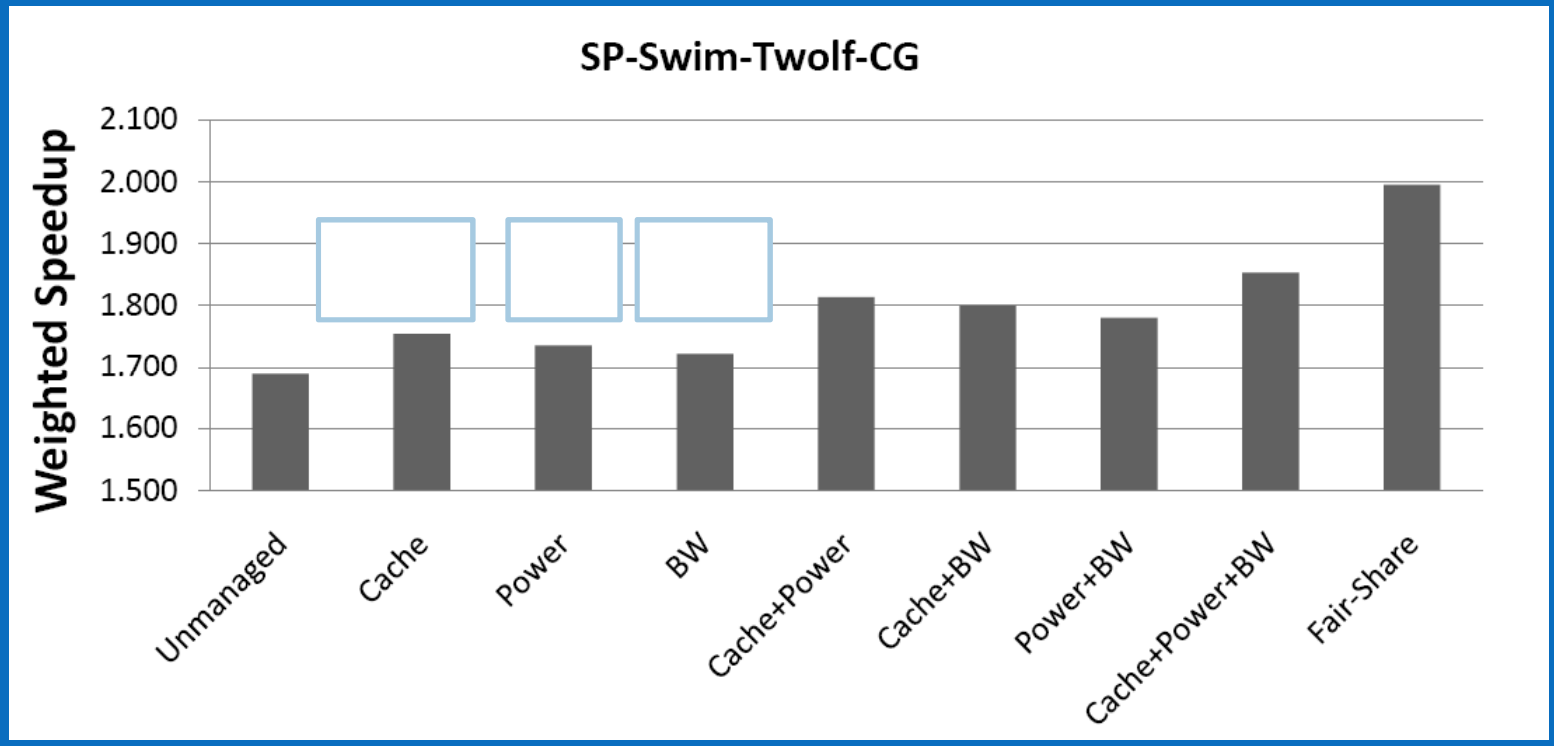


64 MB Cache
- 64 byte blocks
- → 1M blocks

Replace each block 10X
- 4 miss / 1000 instructions
- → 2.5B instructions

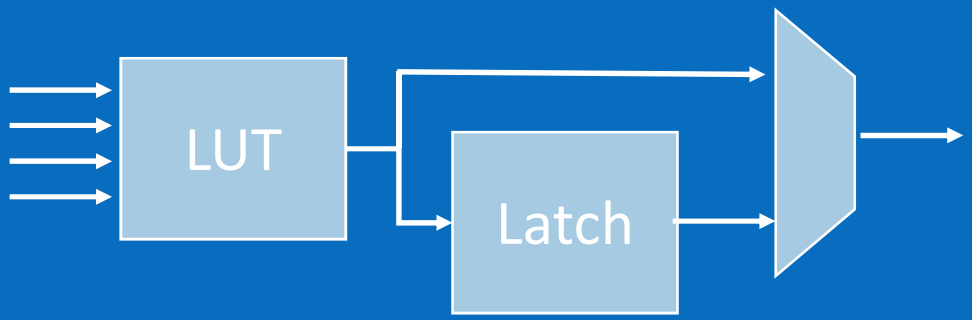
Simulation
- 10K instructions/sec
- → over 2.5 days

Motivation

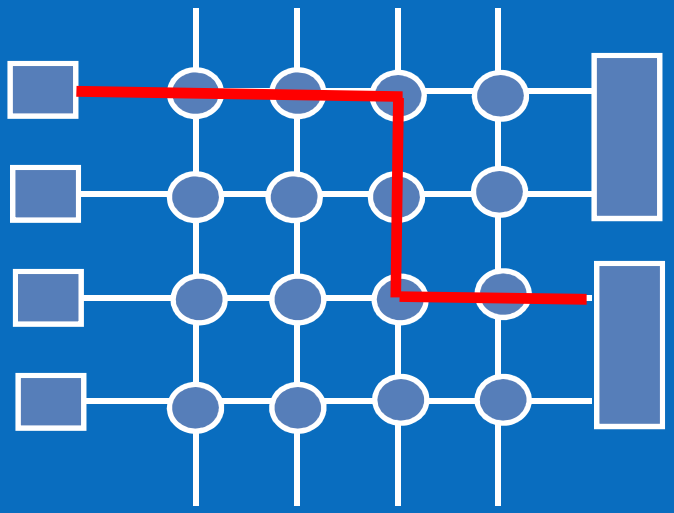


Source Ipek Engin

Field Programmable Gate Arrays



.....



Research Accelerator for MultiProcessors

Goal: FPGA emulation of large-scale MPs

- easy to put together experimental MP designs
- fast enough to support synergistic research in SW

Sharing and spreading the technology

- scalable FPGA emulation fabric
- library of composible architecture modules
- "glue" for instrumentation and interfacing
- reference starter systems

Multi-university collaboration:

- David Patterson (UCB), Arvind (MIT), Krste Asanovíc (MIT), Derek Chiou (UT), Joel Emer (Intel/MIT), James Hoe (CMU), Christos Kozyrakis(Stanford), Shih-Lien Lu (Intel), Mark Oskin (UW), Jan Rabaey (UCB), Jose Renau (UCSC) and John Wawrzynek (UCB)

RAMP Emulation Fabric

Generation 1: Berkeley Emulation Engine 2 (BEE2)

- 5 Virtex II-70, 18 banks DDR2-400 memory, 20 10GigE conn.

Generation 2: RAMP2/BEE3

- 4 Virtex-5-110, 64-GByte DDR, Infiniband, PCI-express
- specifically designed with RAMP applications in mind
- Chuck Thacker/Microsoft is leading the charge and paying the NRE
- Xilinx is supporting a large, shared remote-access cluster to be built

3 Reference Designs

Transactional Memory RAMP (Red)

- 8 CPUs with 32KB L1 data-cache with Transactional Memory
- Led by Kozyrakis at Stanford (based on Stanford TCC)

Message Passing RAMP (Blue)

- 768 Microblaze softcores over 16 BEE2 boards
- NAS benchmarks (MPI) and Internet Services (LAMP)
- Led by Patterson and Wawrzynek at Berkeley

Cache Coherent RAMP (White/Purple)

- Shared memory/Cache coherent (ring-based)
- Led by Chiou at UT and Oskin at UW

and more . . . .

please see [Wawrzynek, et al. IEEE Micro, Mar 2007]

Definitions

- Functional Emulator – A circuit functionally equivalent to a design, but which does not provide any insights on any specific design metrics.

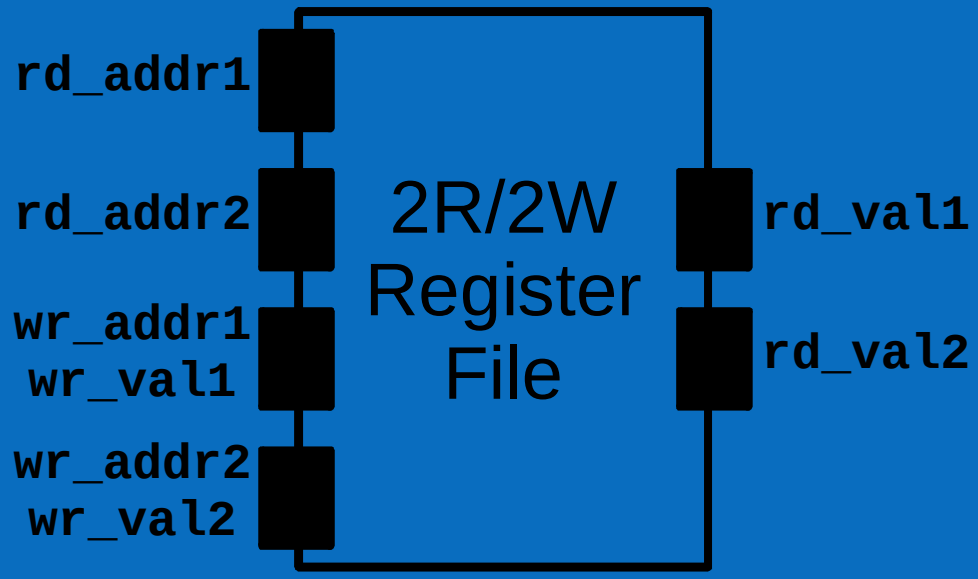
- Prototype (or Structural Emulator) – A logically isomorphic and functionally equivalent representation of a design – often implemented in a different technology, e.g., FPGAs.

- Model – An abstract representation of a design sufficiently logically and functionally equivalent to that to allow estimation of design metrics of interest, e.g., performance, power or reliability.

Prototype Example

Register File with 2 Read Ports, 2 Write Ports

- Reads take zero clock cycles
- Direct configuration onto FPGA: 9242 slices, 104 MHz

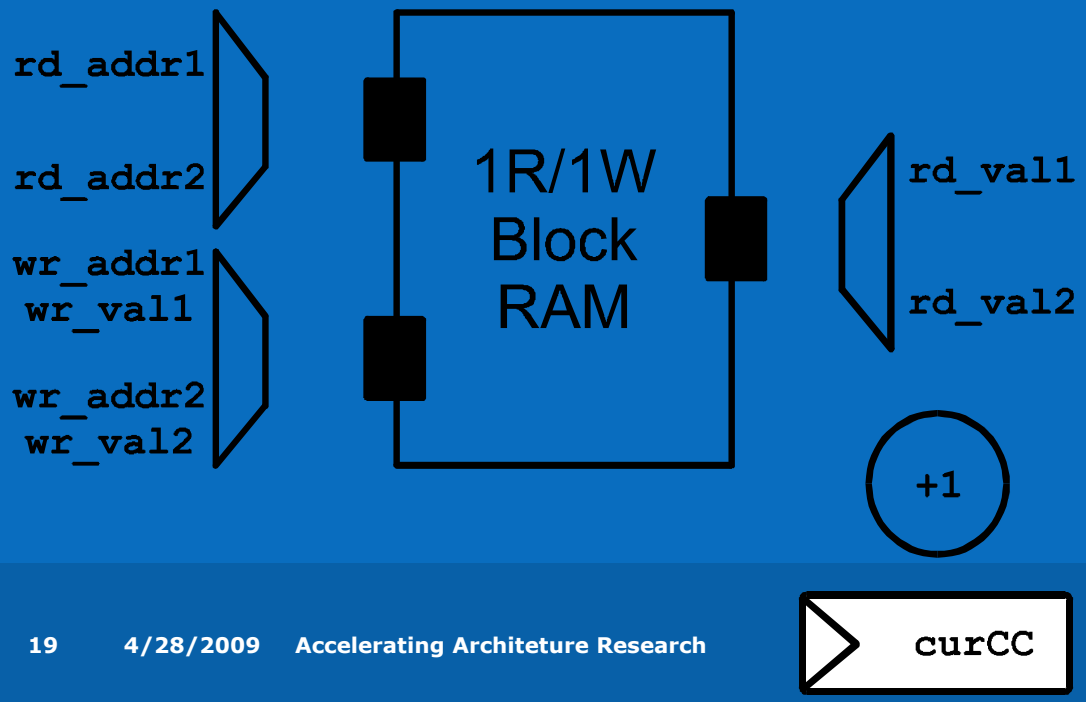


	CC 1	CC 2
rd_addr1	A	C
rd_val1	V(A)	V(C)
rd_addr2	B	D
rd_val2	V(B)	V(D)

Performance Model Example

Simulate the circuit using synchronous BlockRAM

- First do reads, then serialize writes
- Only update model time when all requests are serviced
- Results: 94 slices, 1 BlockRAM, 224 MHz
- Simulation rate is 224 / 3 = 75 MHz (FPGA-to-Model Ratio)



	Model CC 1		
FPGA CC:	1	2	3
rd_addr1	A	A	A
rd_val1		V(A)	V(A)
rd_addr2	B	B	B
rd_val2			V(B)

FPGA Performance Model Metrics

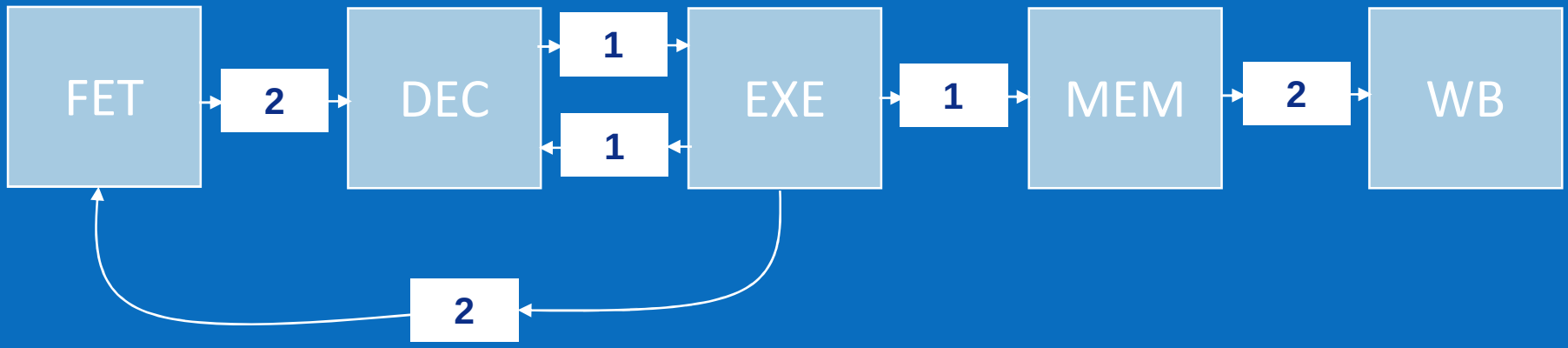
FPGA cycle to Model cycle Ratio (FMR):

$$FMR = \frac{Cycles_{FPGA}}{Cycles_{Model}}$$

Simulator Frequency:

$$Frequency_{simulator} = \frac{Frequency_{FPGA}}{FMR}$$

Asim Ports in Software



Used in software Asim performance models

- All communication goes through Ports
- Ports have a model time latency
- Beginning of a model cycle a module reads all Ports
- End of a model cycle write all Ports

Related: RAMP RDL channels, UT Fast Connectors

Software Simulation

Step	FET	DEC	EXE	MEM	WB
0	A				
1	A				
2		NOP			
3			NOP		
4				NOP	
5					NOP
6	B				
7	B				
8		A			
9		A			
10			NOP		

 = model cycle

Barrier Synchronization

FPGA CC	FET	DEC	EXE	MEM	WB
0	A	NOP	NOP	NOP	NOP
1	A				
2	A				
3	B	A	NOP	NOP	NOP
4	B	A			
5		A			
6	C	B	A	NOP	NOP
7		B			
8	D	C	B	A	NOP
9	D				
10	D				

 = model cycle

A-Ports Distributed Synchronization

FPGA CC	FET	DEC	EXE	MEM	WB
0	A	NOP	NOP	NOP	NOP
1	B	A	NOP	NOP	NOP
2	C	B	A	NOP	NOP
3	D	B		A	
4	E (full)	B		A	
5		B		A	
6		B		A	
7		C	B	A	
8	F	D	C	B	A
9	G (full)	D		C	B
10		D			C
11		D			
12		D			
13		E	D		
14		F	E	D	

run-ahead in time until buffering fills

long-running ops can overlap

Observed results of simulation do not change!

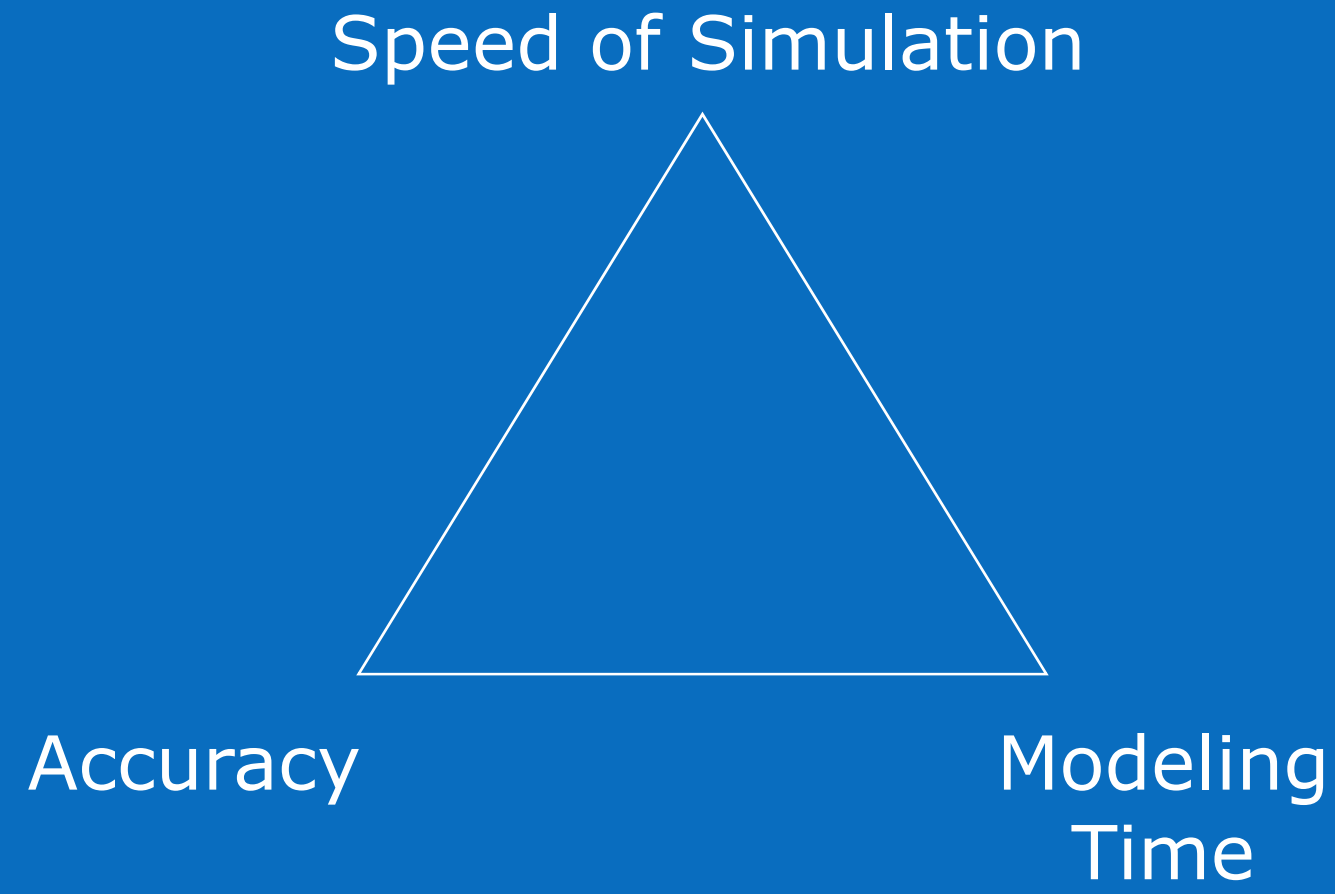
HAsim: Current status - models

- RISC ISA functional model operating
  - Full user-mode RISC ISA
  - Pipelined multi-phase instruction execution
  - Supports speculative OOO design
    - Physical Reg File and ROB
    - Speculative rewinds
    - Full virtually addressed memory
- Instruction-per-cycle model
  - Runs full SPECMark benchmarks
- In-order pipeline
  - Supports pipeline flow control / branch mis-speculation
  - Simple cached memory hierarchy
  - Runs full SPECMark benchmarks
- Simple out-of-order (R10K-like) model
  - Four wide super-scalar
  - Supports out-of-order instruction issue
  - Ran full SPECMark benchmarks

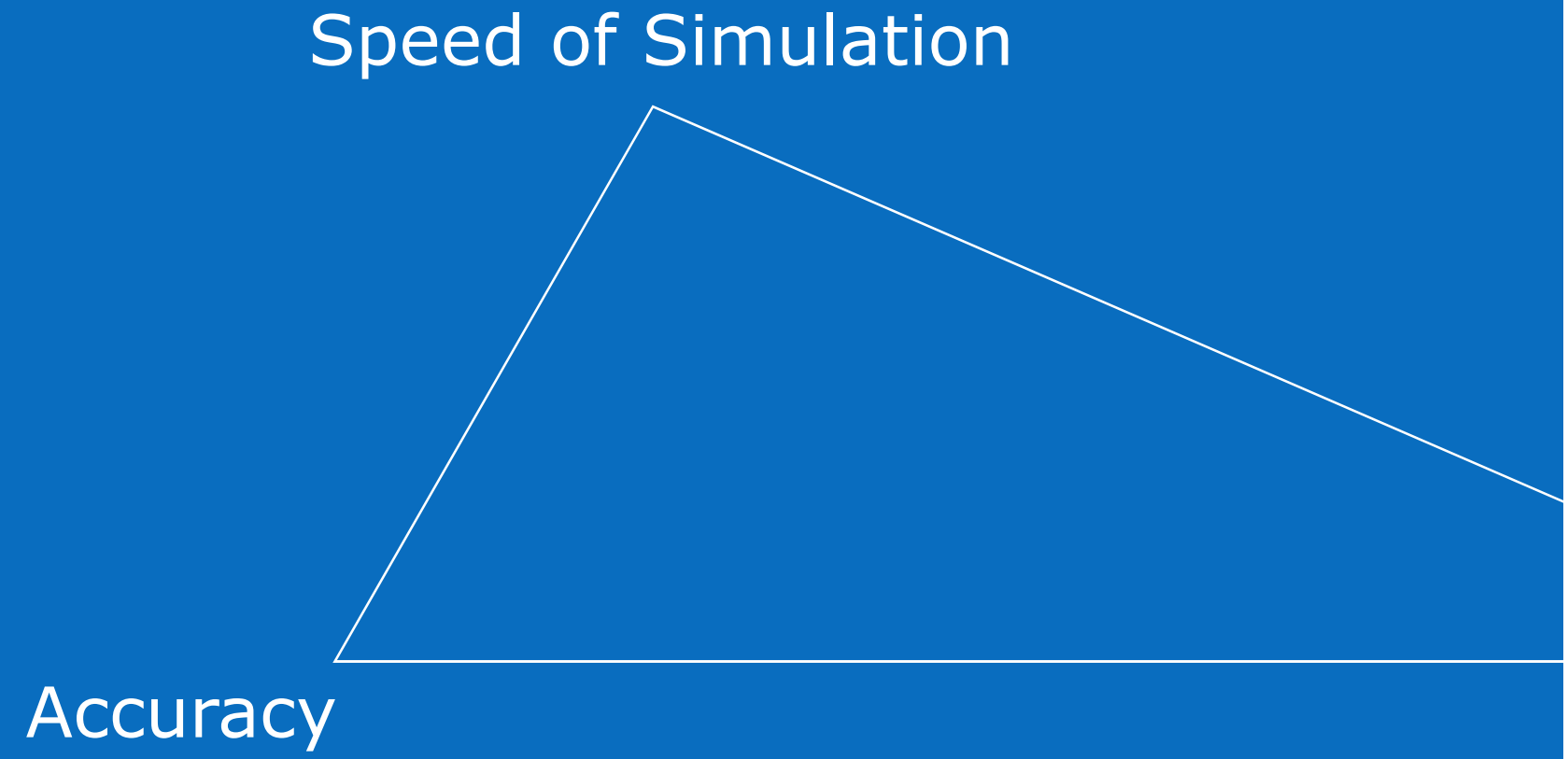
FPGA Modeling

- The FPGA Modeling Promise
  - Raw speed of today's FPGAs: 100-400 MHz
  - Estimated FMR: 10
  - → Frequency Simulator = 100 /10 -> 10 MHz
- The FPGA Modeling Trap
  - Hardware design is hard

Simulation Tradeoffs



Simulation on FPGAs



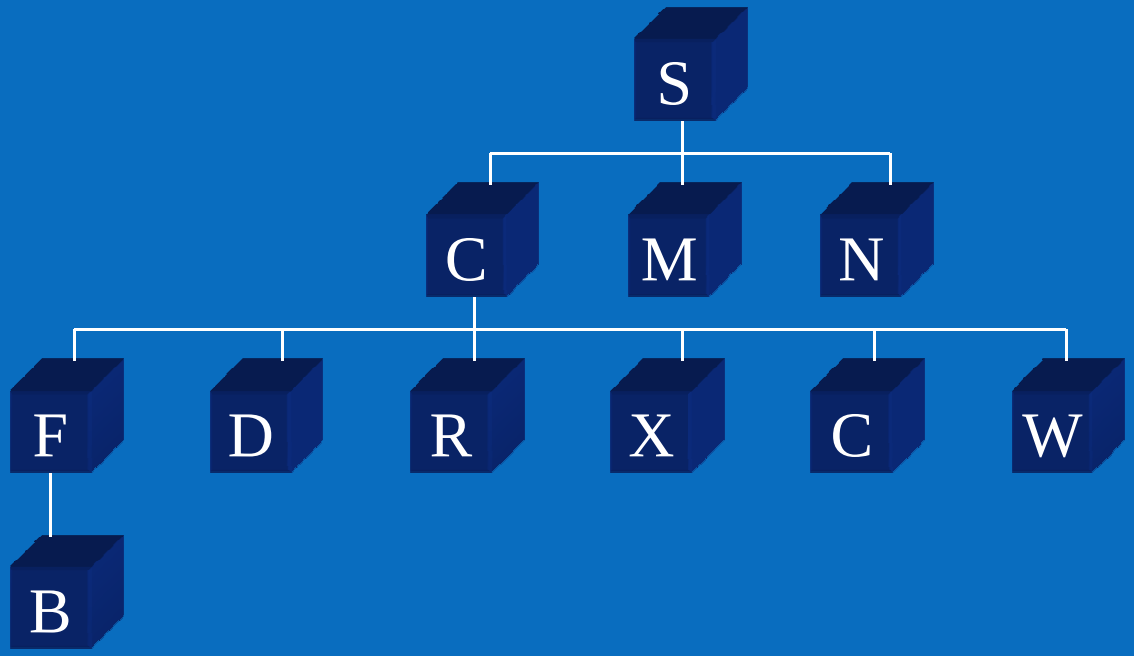
Common Infrastructure

- Modularity
- Support utilities
- Hybrid Modules

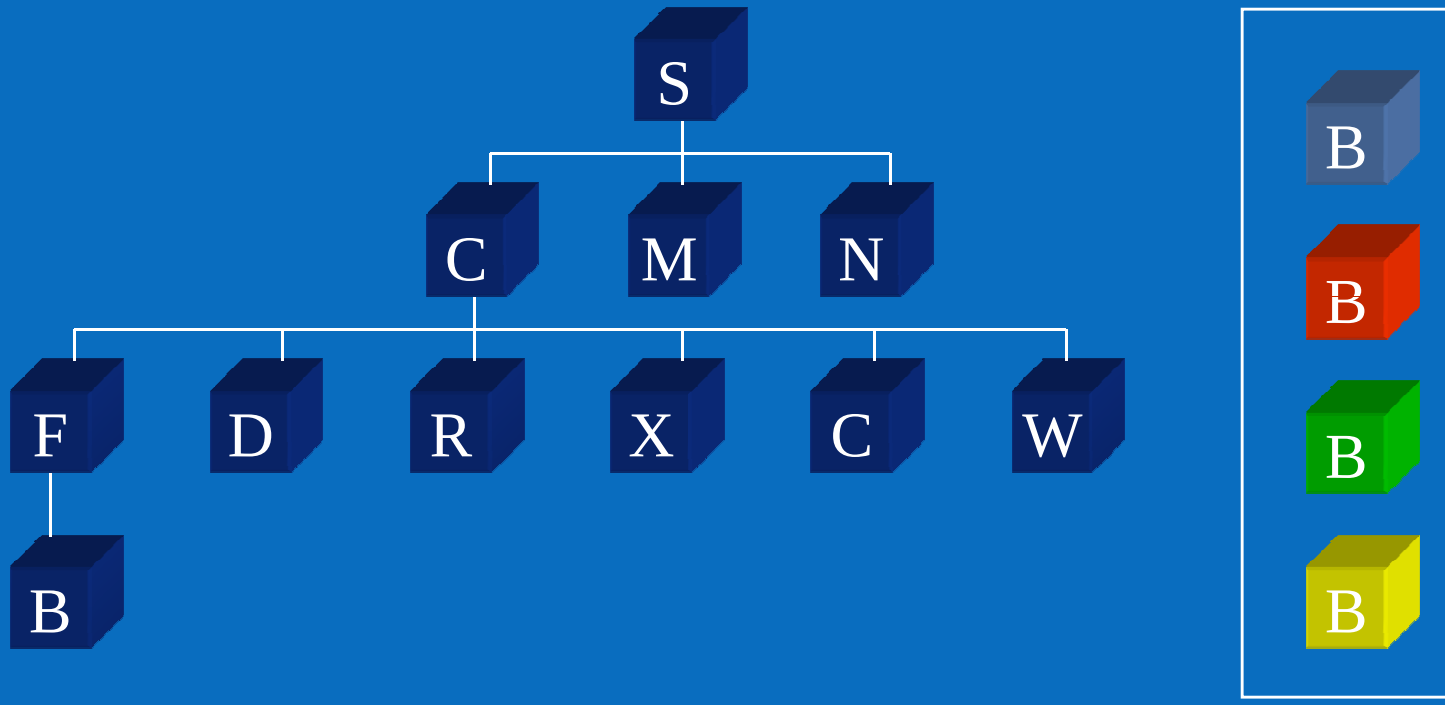
Why modularity?

- Speed of model development
- Shared components between products
- Reuse across generations
- Encourages isomorphism to design
- Improved fidelity
- Facilitates speed/fidelity trade-offs
- Architectural experimentation
- Factorial development and evaluations
- Sharing

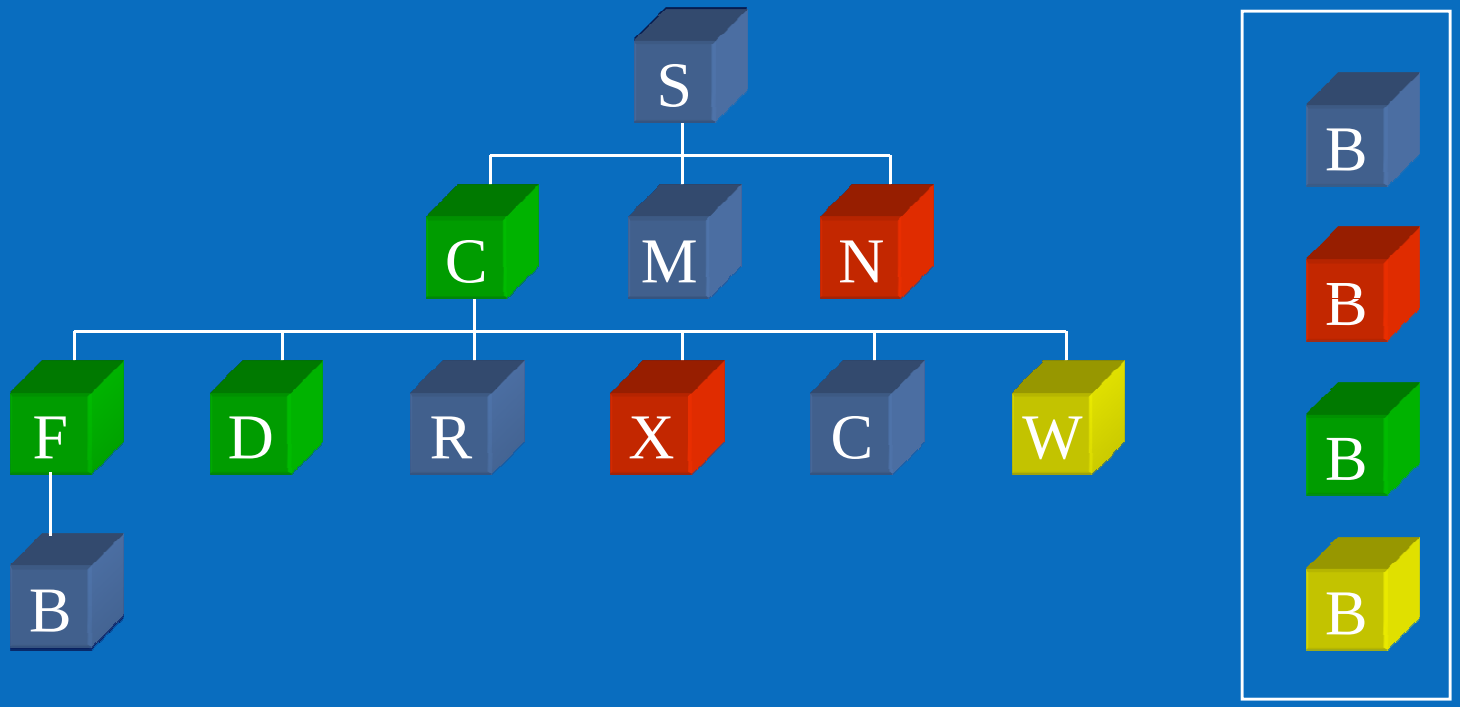
AWB Module Hierarchy



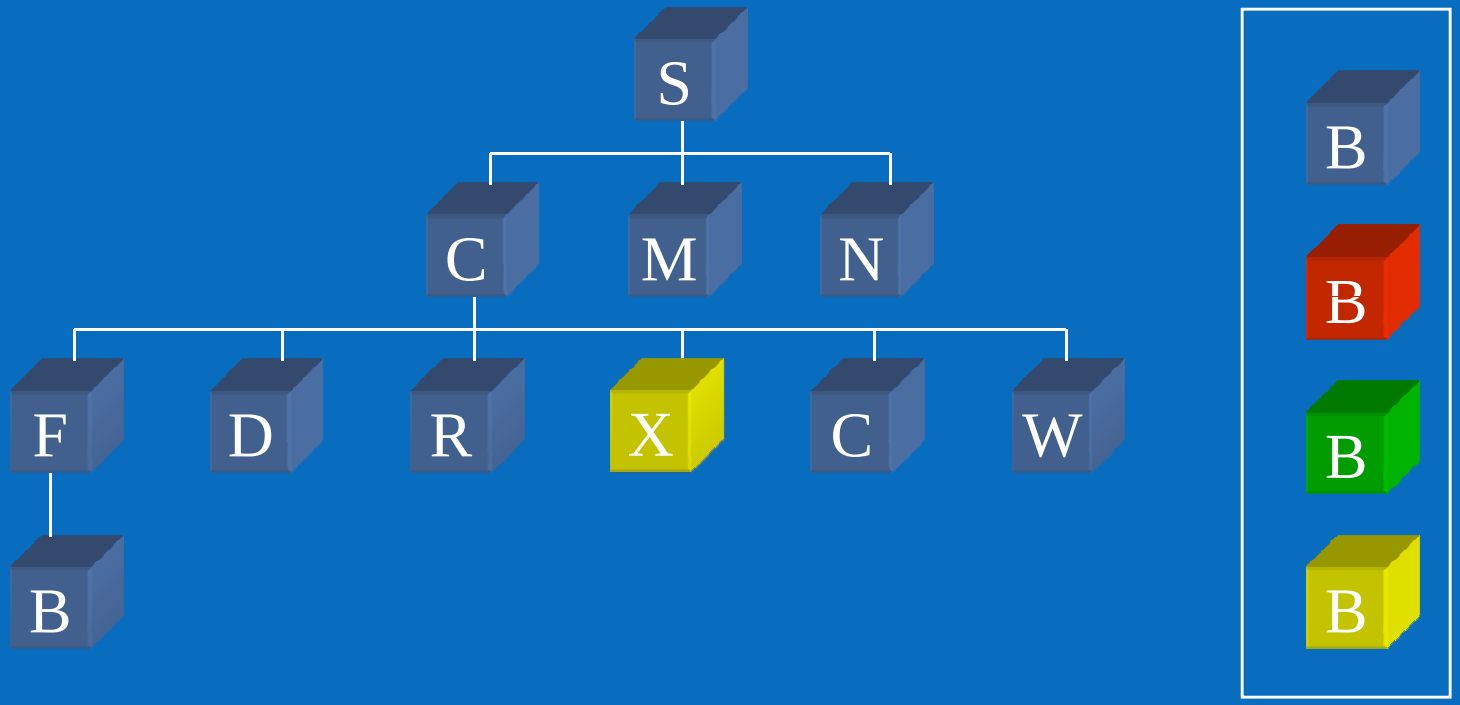
AWB Module Selection



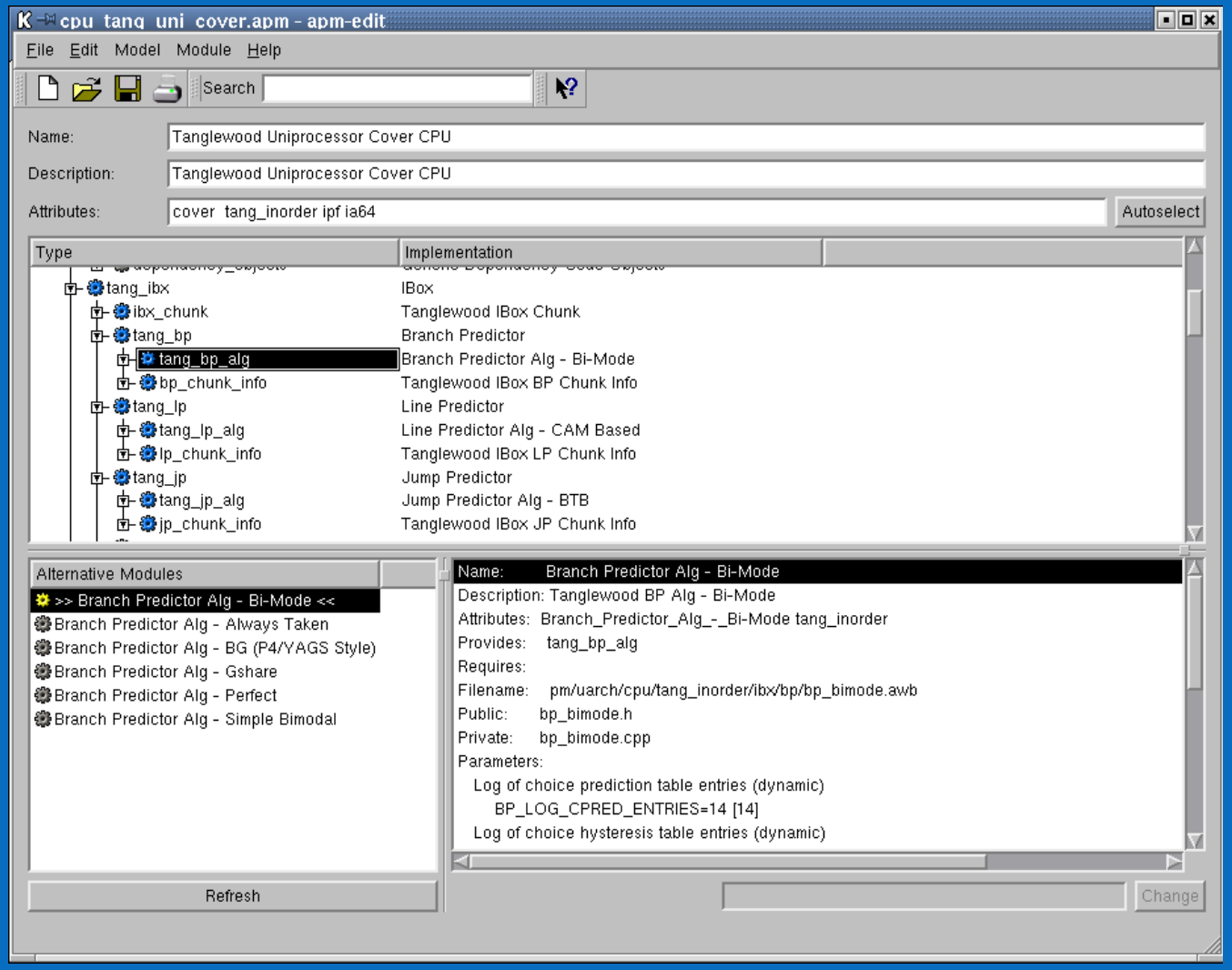
Module Selection



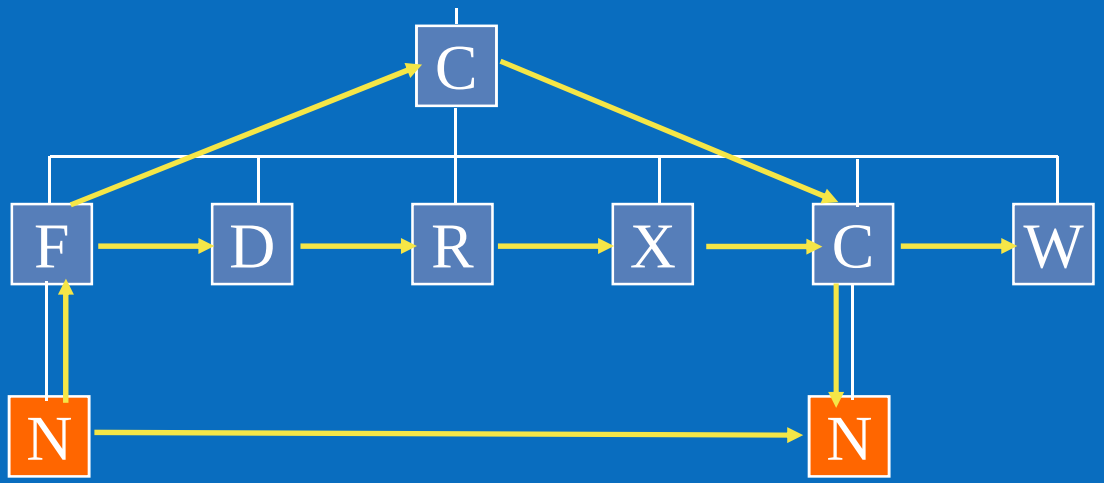
Module Replacement



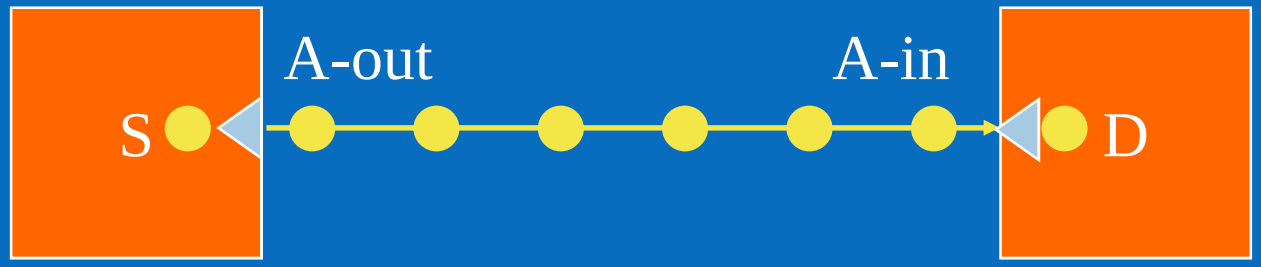
(H)ASIM Module Hierarchy



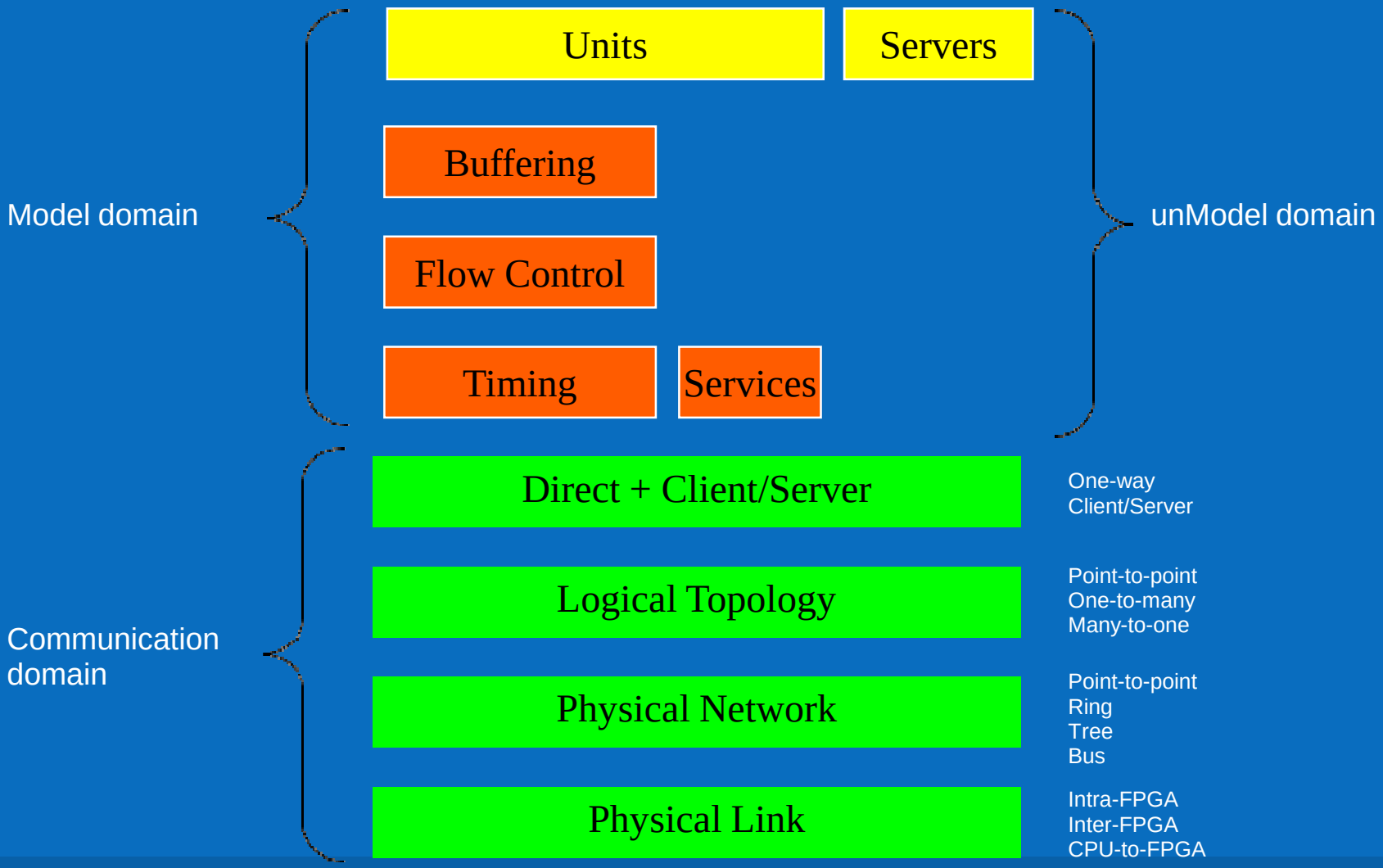
Communication:
A modularity speedbump



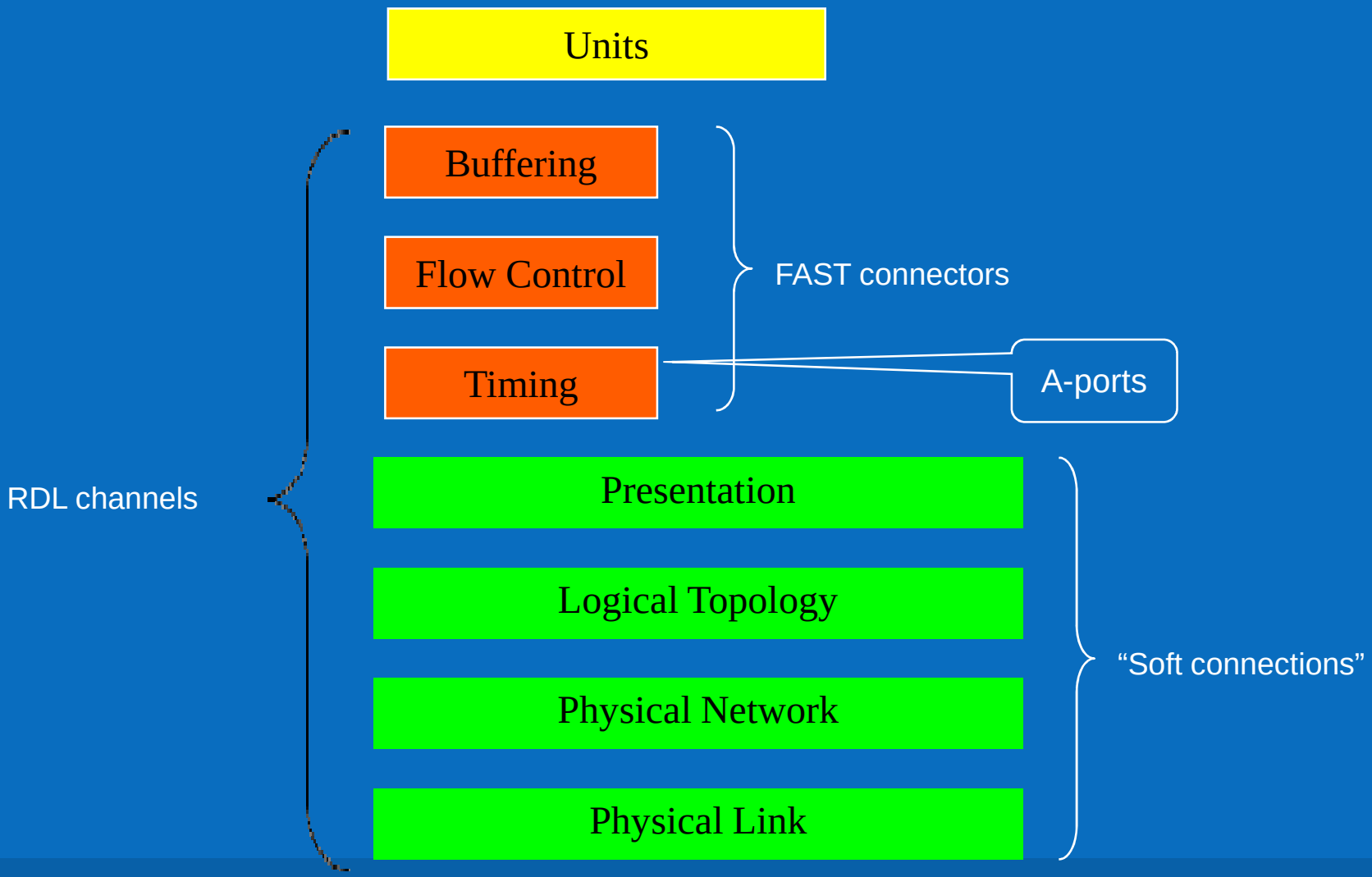
Soft connections:
Flattening the speedbumps



Interface Layers



Multi-layer implementations



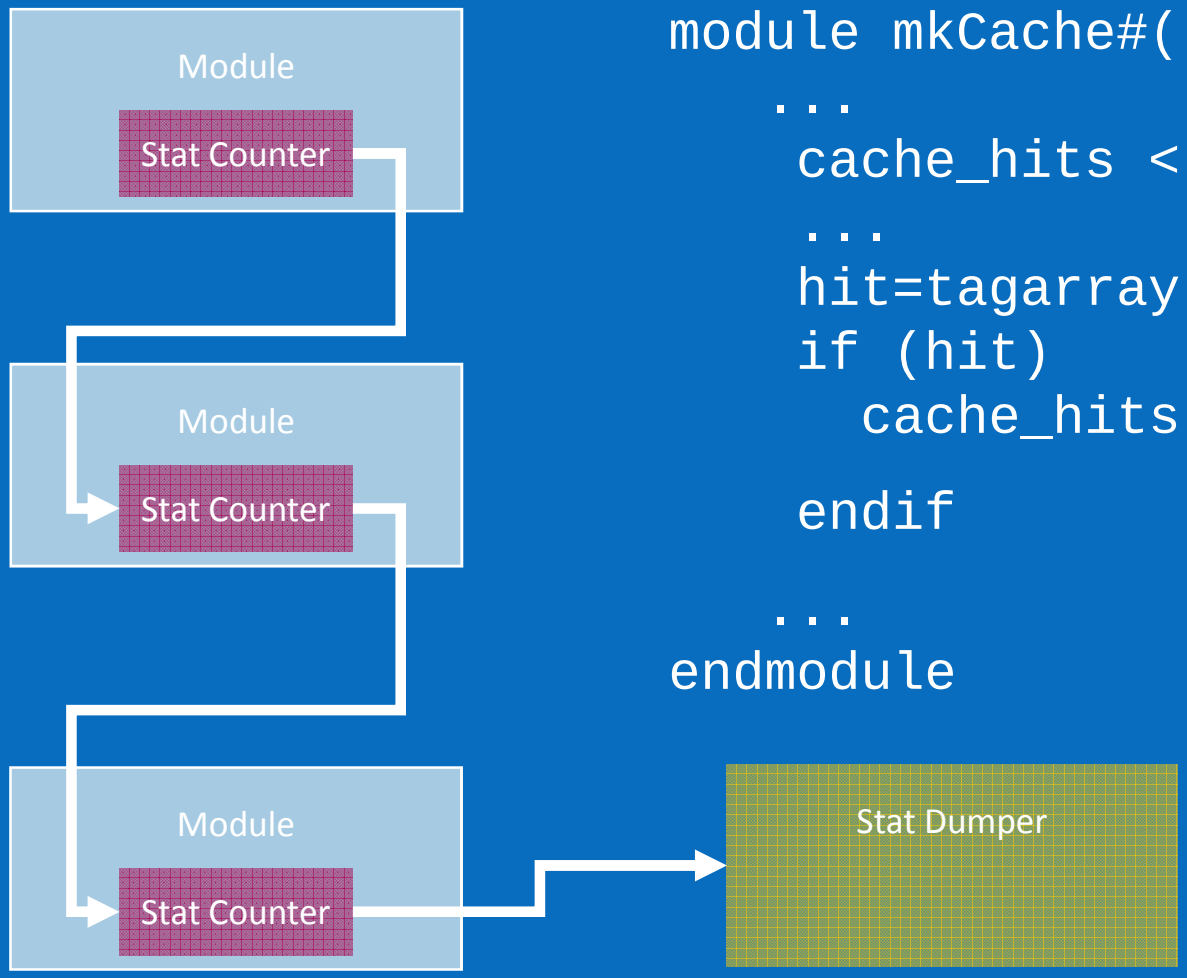
Bluespec (Asim-style) module

```
module [HAsim_module] mkCache#() (Empty);

   Port#(Addr) req_port  <- mkSendPort('a2cache');
   Port#(Bool) resp_port <- mkRecvPort('cache2a');

   TagArray tagarray <- mkTagArray();

   rule cycle(True);
     Maybe#(Addr) mx = req_port.get();

     if (isValid(mx))
       resp_port.put(tagarray.lookup(validValue(mx)));

   endrule
endmodule
```

Bluespec (Asim-style) submodule

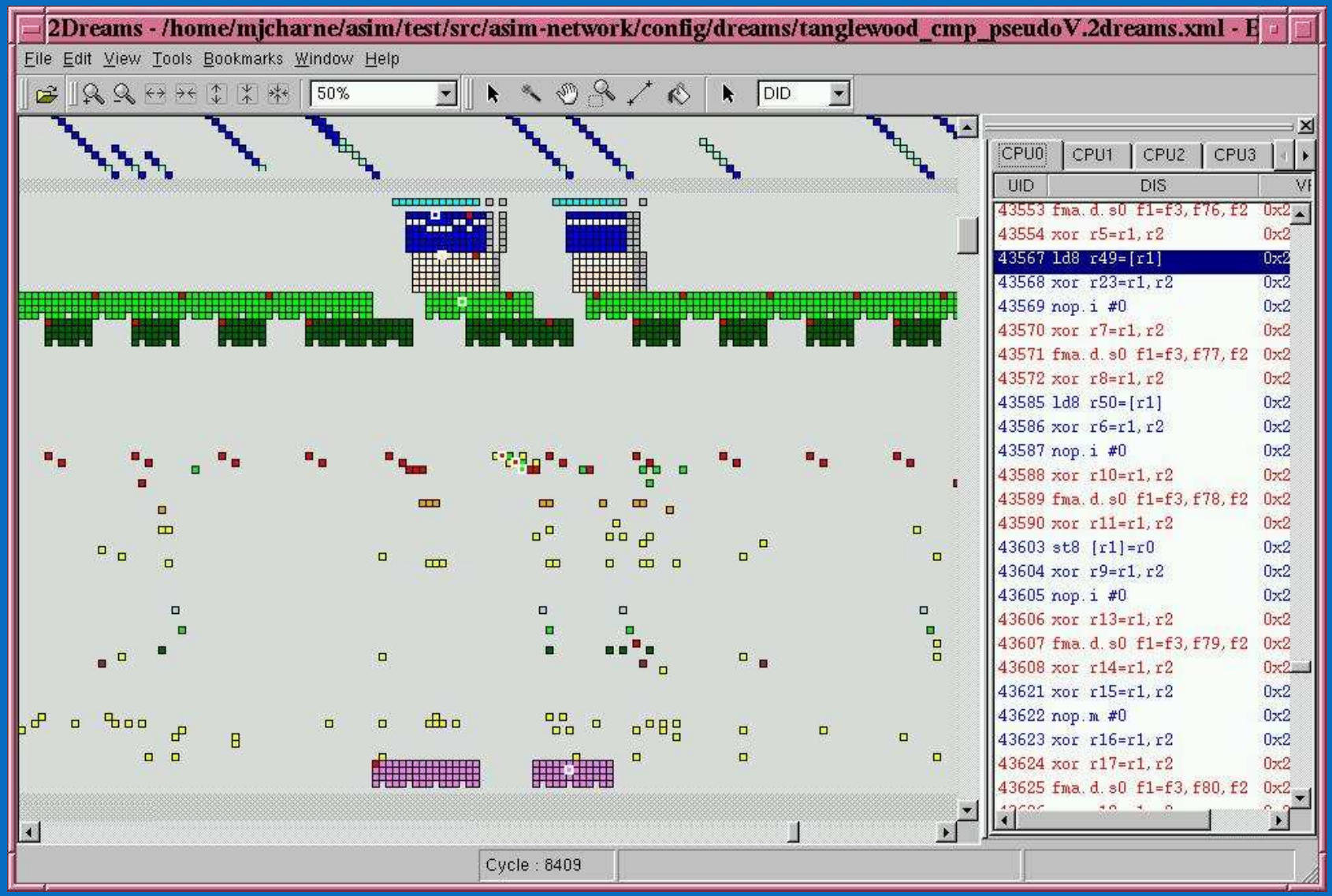
```
module mkTagArray(TagArray);

  RegFile#(Bit#(12),Bit#(4)) tagArray<- mkRegFileFull(...);

  method Bool lookup(Bit#(16) a);
    return (tagArray.sub(getIndex(a)) == getTag(a));
  endmethod


  function Bit#(4) getTag(Address x);
    return x[15:12];
  endfunction


  function Bit#(12) getIndex(Address x);
    return x[11:0];
  endfunction

endmodule
```

Support functions - stats

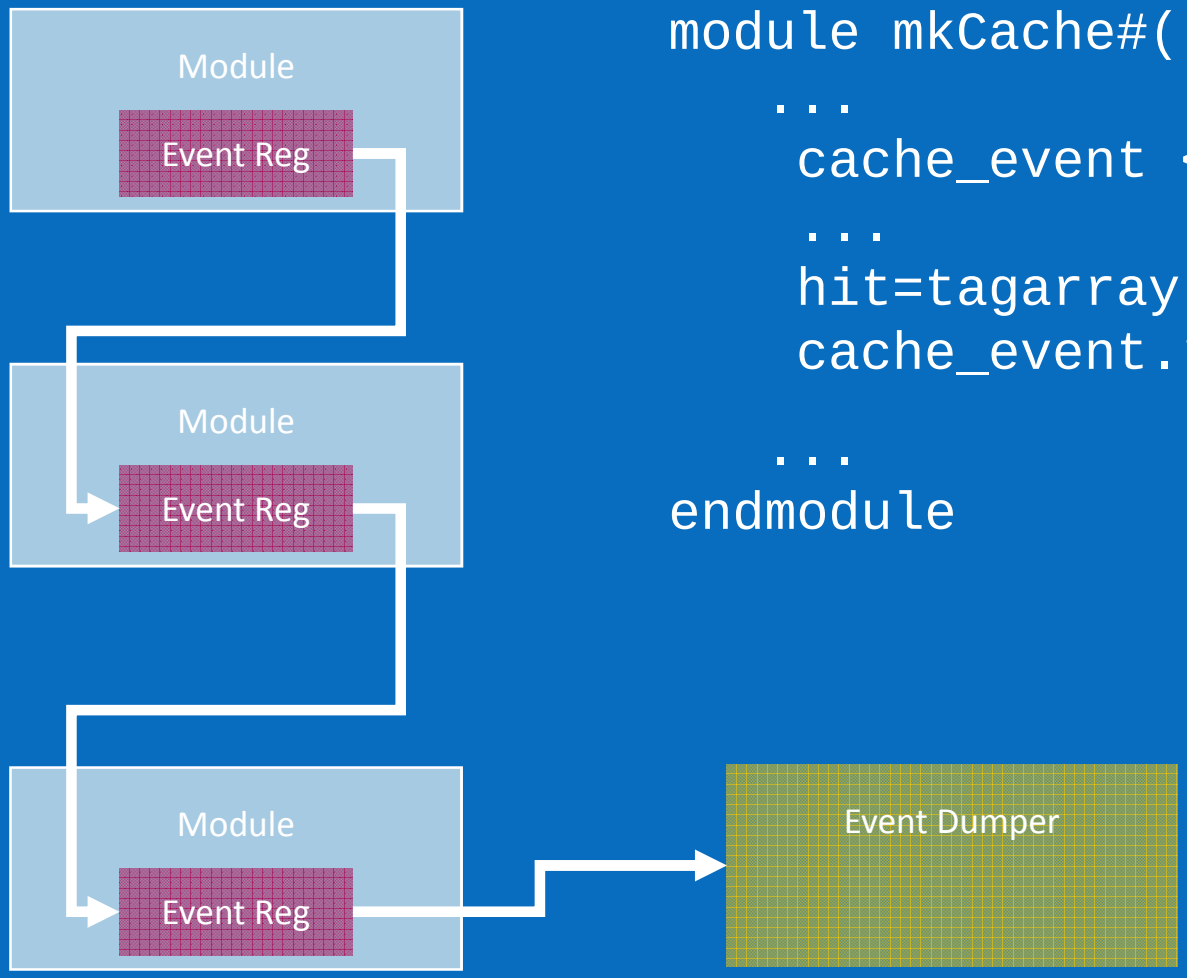



```
module mkCache#(...) (Empty);
    ...
     cache_hits <- mkStat(...);
     ...
     hit=tagarray.lookup(...);
     if (hit)
       cache_hits.increment();
     endif
    ...
endmodule
```

Cycle Display - 2Dreams

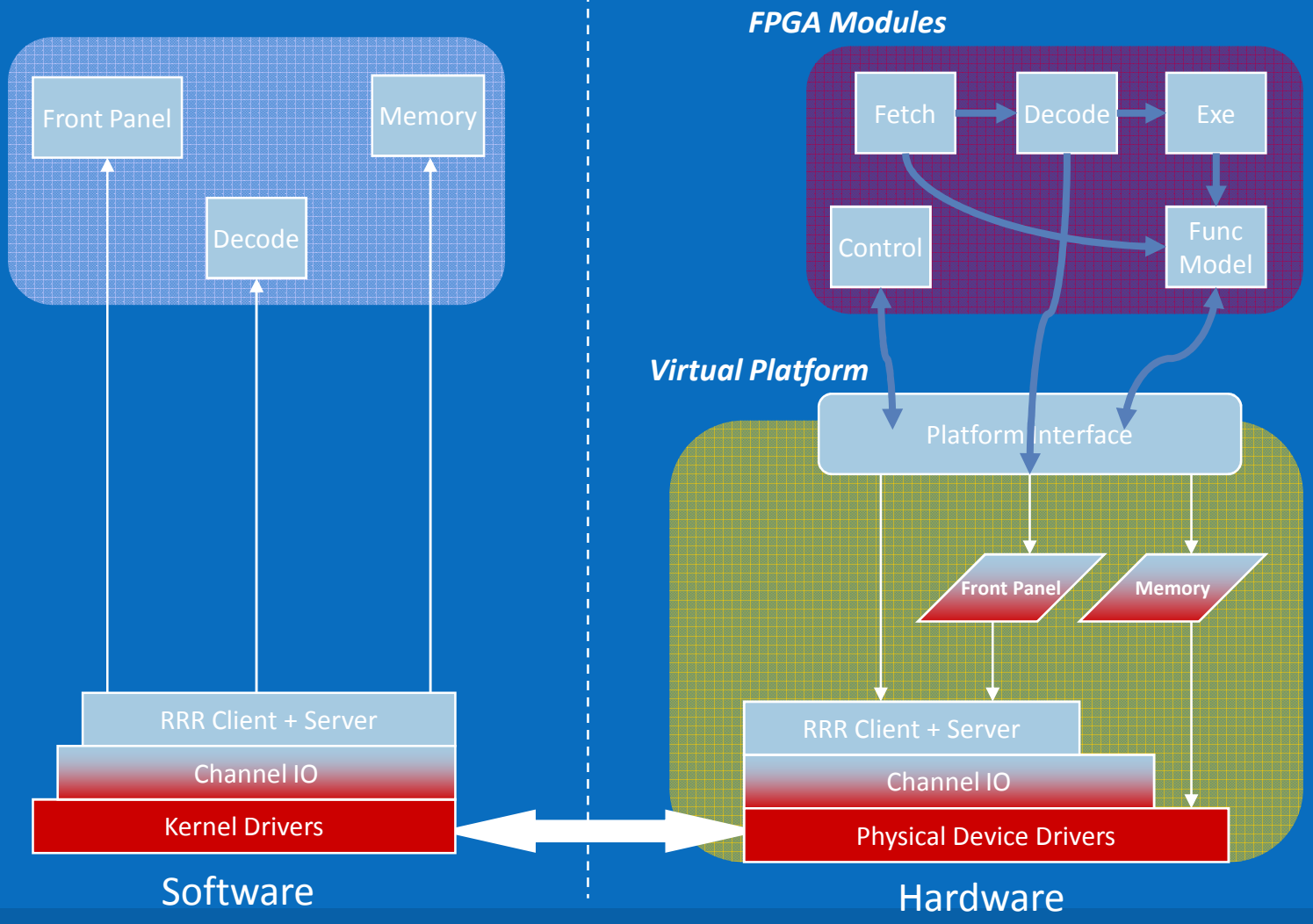


Support functions - events



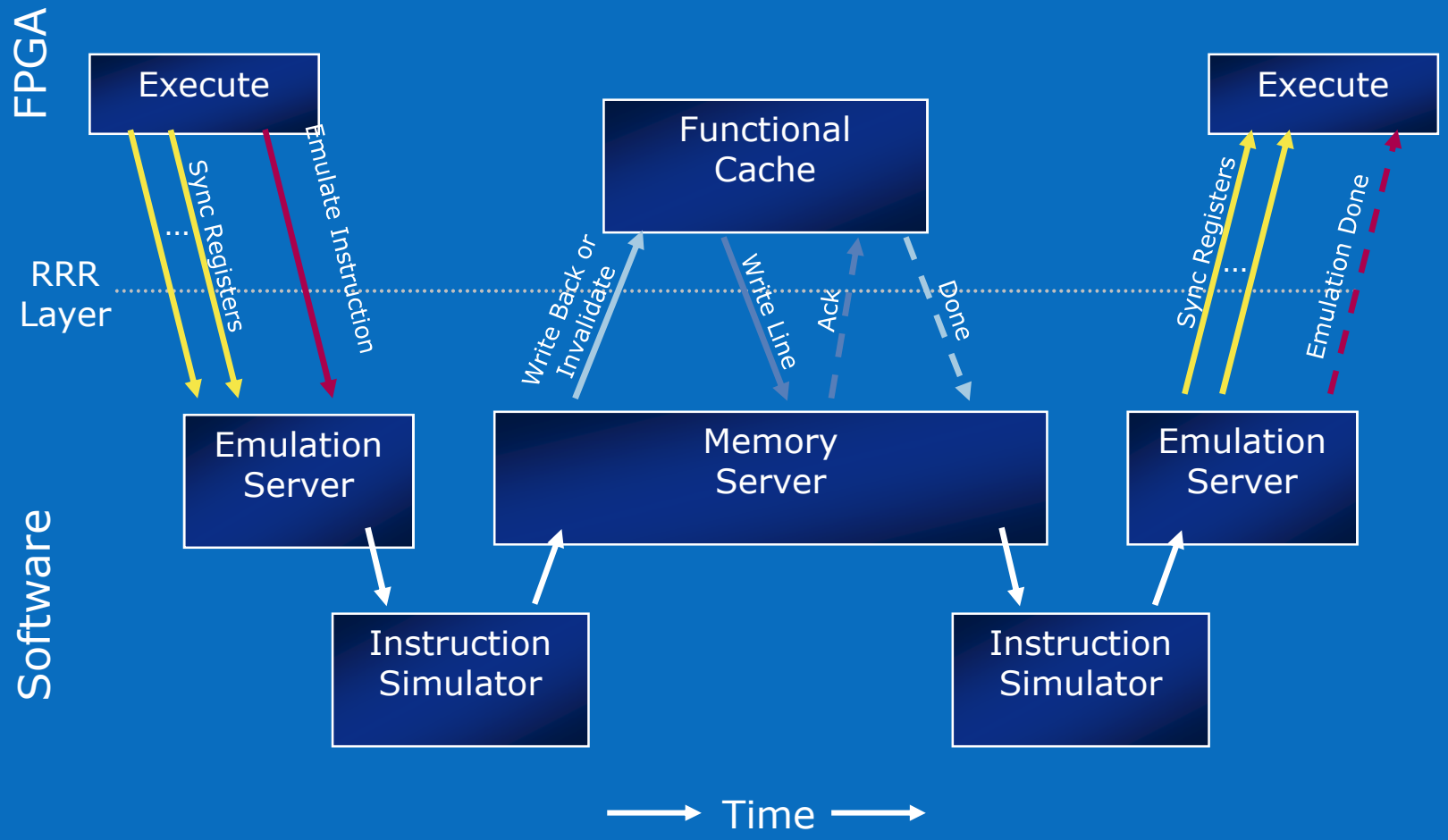

```
module mkCache#(...) (Empty);
    ...
     cache_event <- mkEvent(...);
     ...
     hit=tagarray.lookup(...);
     cache_event.report(hit);

    ...
endmodule
```

Hybrid Modules/Virtual Platform



Source: Intel, Angshuman Parashar - VSSAD

Hybrid Instruction Emulation



Source: Intel, Michael Adler - VSSAD

Implemented in a day

HAsim: Current status - infrastructure

- Build infrastructure
  - Model specification (using .awb files and apm-edit)
    - Supports public/private implementation files and parameters (static)
  - Model configure, parallel compile, and synthesis flow operational

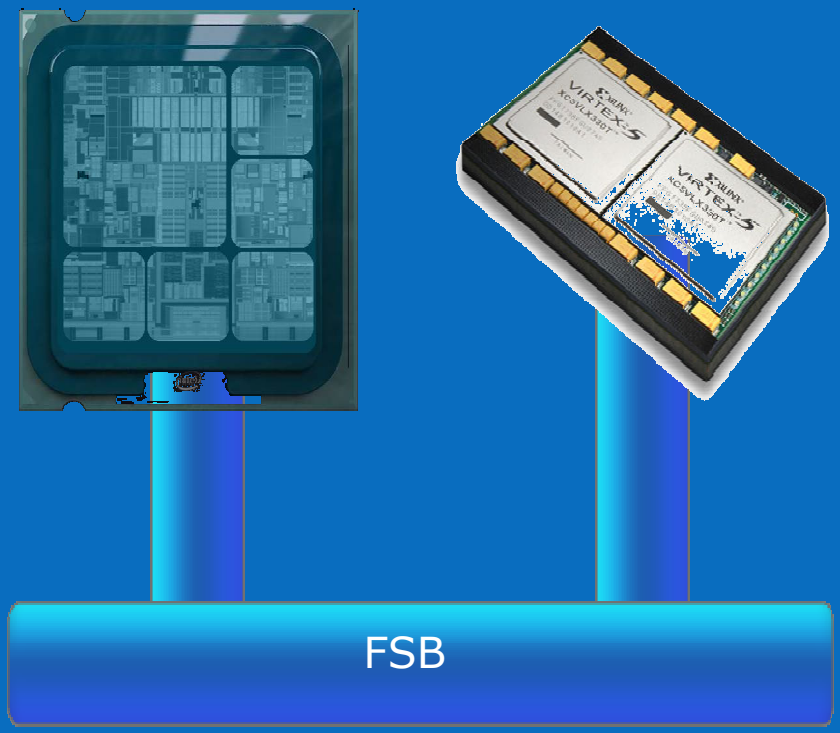
    … and available from awb

- Operational support structures
  - Stall (bounded FIFO) ports
  - Flow-control ports
  - Ports - distributed synchronization
  - Named connections - autoconnecting across synthesis boundaries
  - Point-to-point and ring-based soft connections

  - Global Controller – run control from host via RRR
  - Events – dump to file via RRR
  - Stats – dump to file via RRR
  - Debug print – to host console via RRR

- RRR hybrid modules
  - Running on Hi-Tech Global PCIe board and in simulation

Nallatech ACP Platform



Other efforts (partial list)

- Intel
  - Pentium FPGA - Shih-Lien Lu
  - Dragonhead – hardware trace driving FPGA cache models
  - Co-sim – SoftSDV driving FPGA cache model
- Academic
  - RAMP – Dave Patterson, Krste Asanovic, ….(me)
  - PROTOFLEX – James Ho - CMU
  - FAST – Derek Chiou – UT-Austin
  - UNUM – Nirav Dave, Michael Pellauer - MIT


intel®
Leap ahead™