NetTM: Faster and Easier Synchronization for Soft Multicores via Transactional Memory

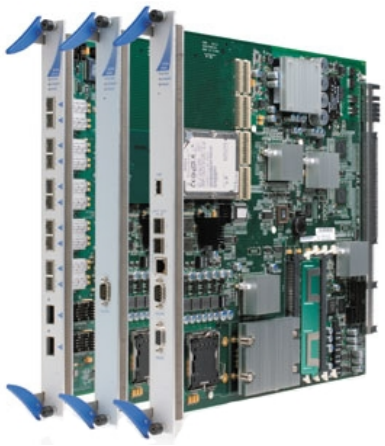
Martin Labrecque

Prof. Greg Steffan

University of Toronto

FPGA, February 27th 2011

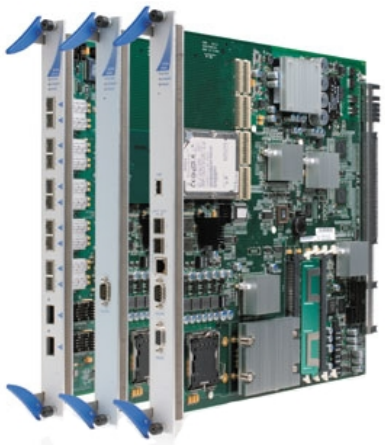
Processors in FPGAs



FPGAs in Telecommunications:

- Present in most high-end routers
- More than 40% of FPGA market

Processors in FPGAs

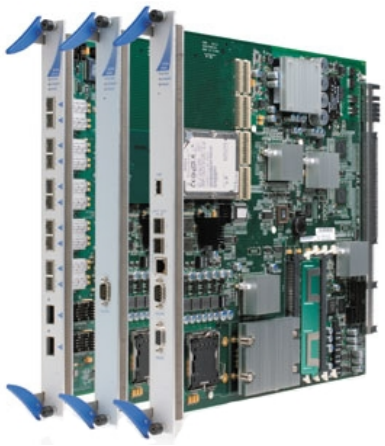


FPGAs in Telecommunications:

- Present in most high-end routers
- More than 40% of FPGA market

Deep packet inspection requires: software + CPUs

Processors in FPGAs



FPGAs in Telecommunications:

- Present in most high-end routers
- More than 40% of FPGA market

Deep packet inspection requires: software + CPUs

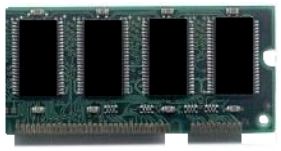
Our goal: implement those cores directly in the FPGA

Processors in FPGAs

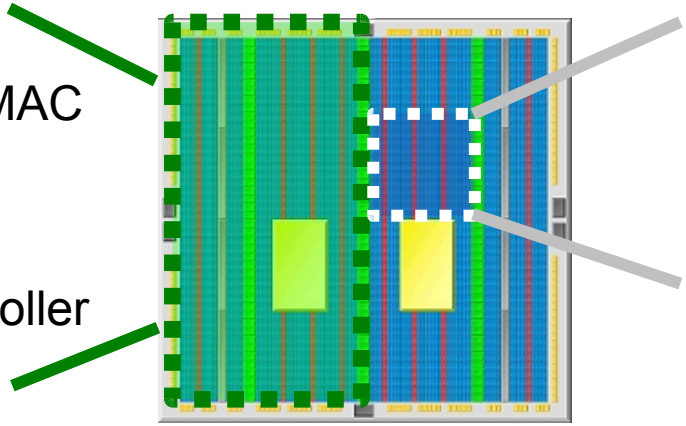
FPGAs in Telecommunications:

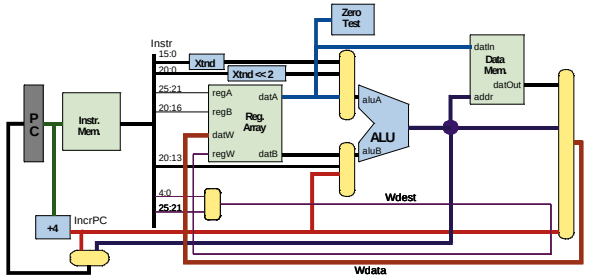
- Present in most high-end routers
- More than 40% of FPGA market

Ethernet MAC



DDR controller



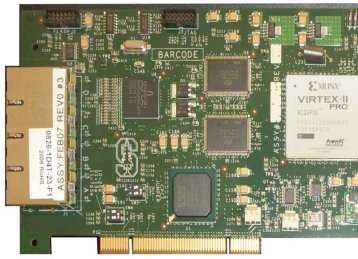


Processor(s)

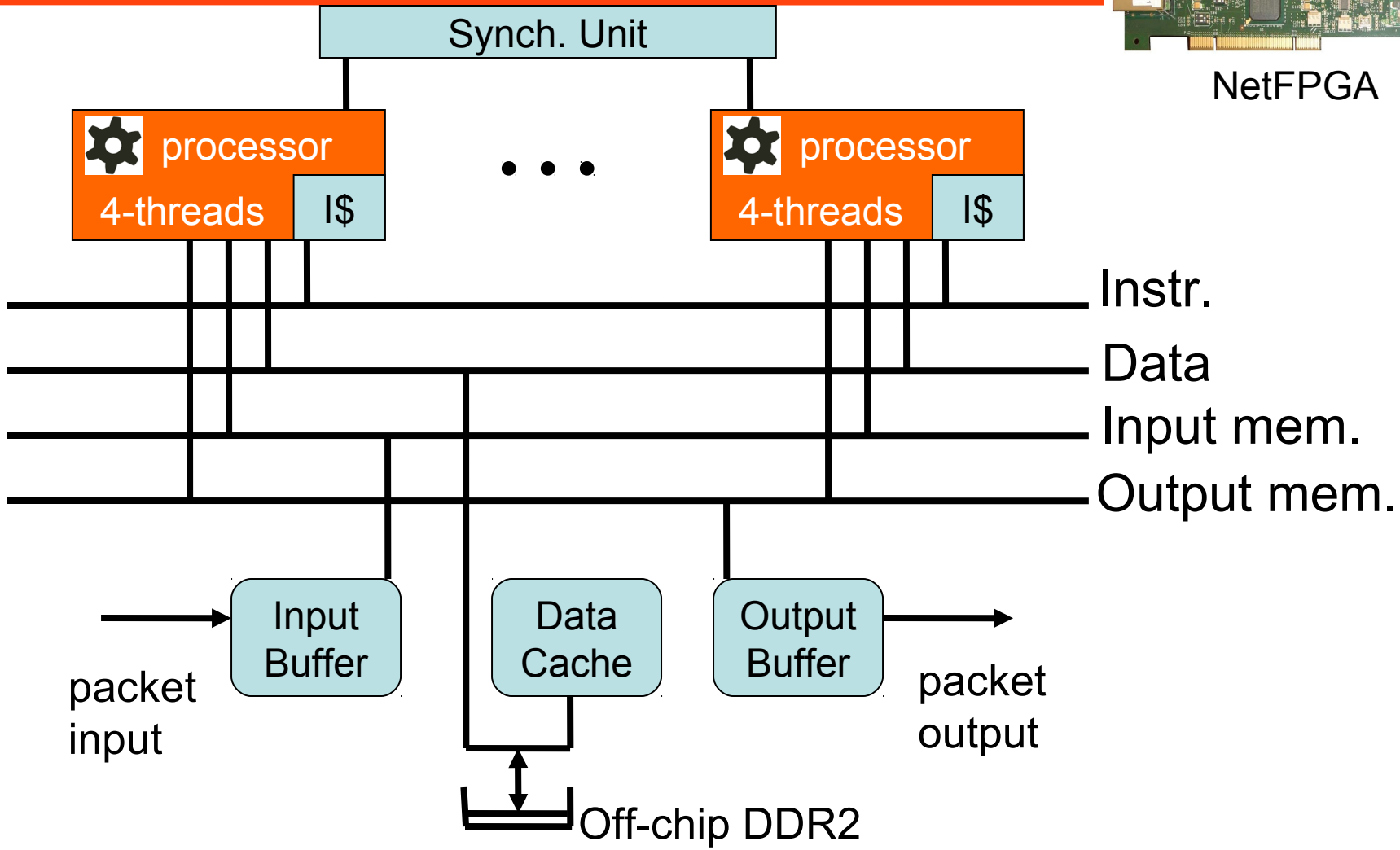
Deep packet inspection requires: software + CPUs

Our goal: implement those cores directly in the FPGA

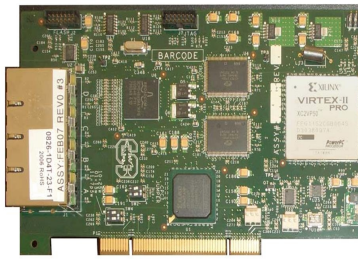
NetThreads: Our Base System

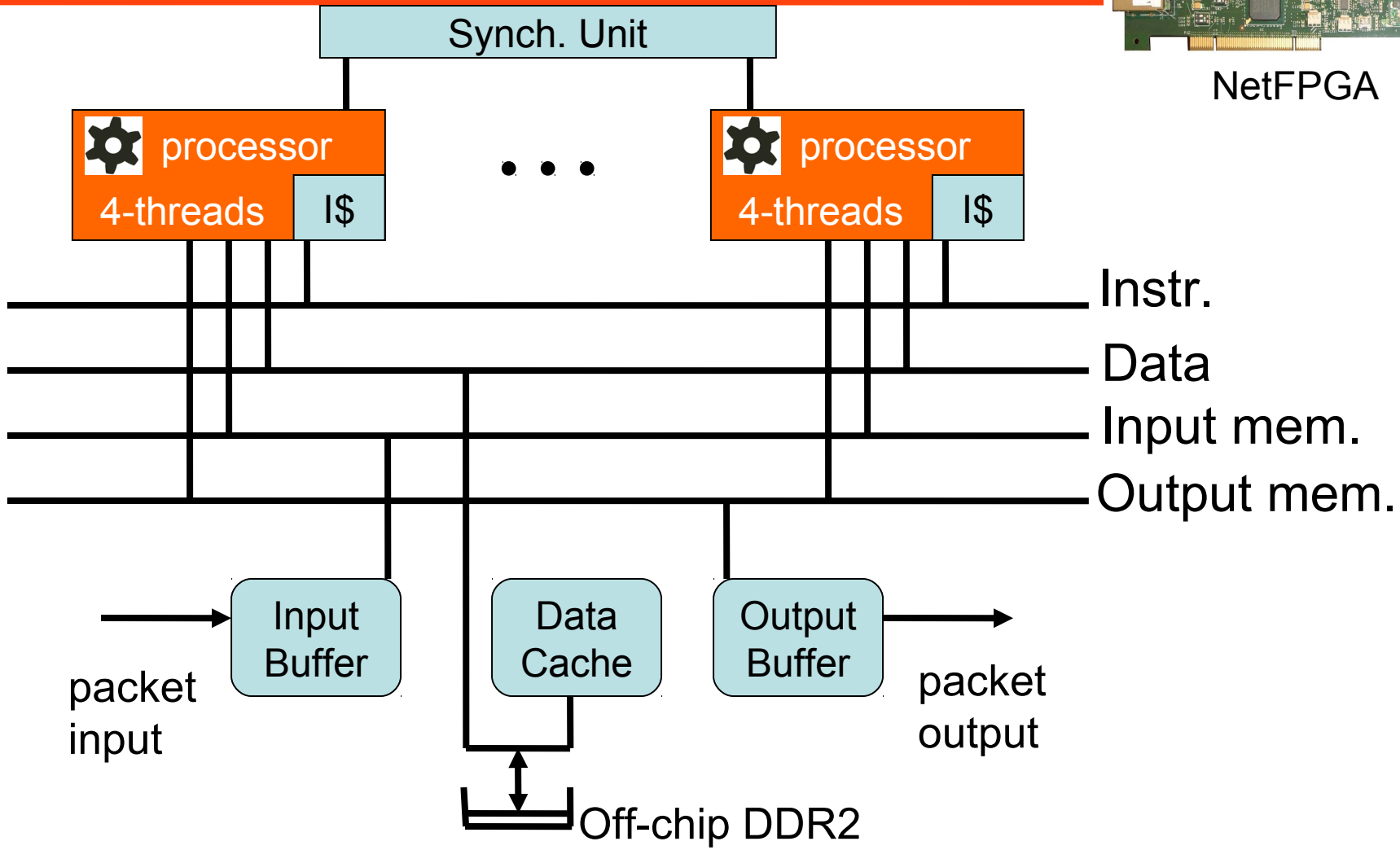


NetFPGA



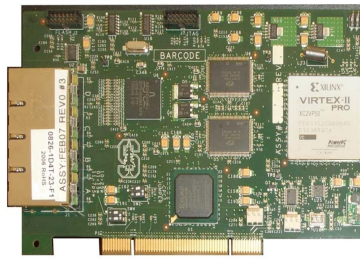
NetThreads: Our Base System

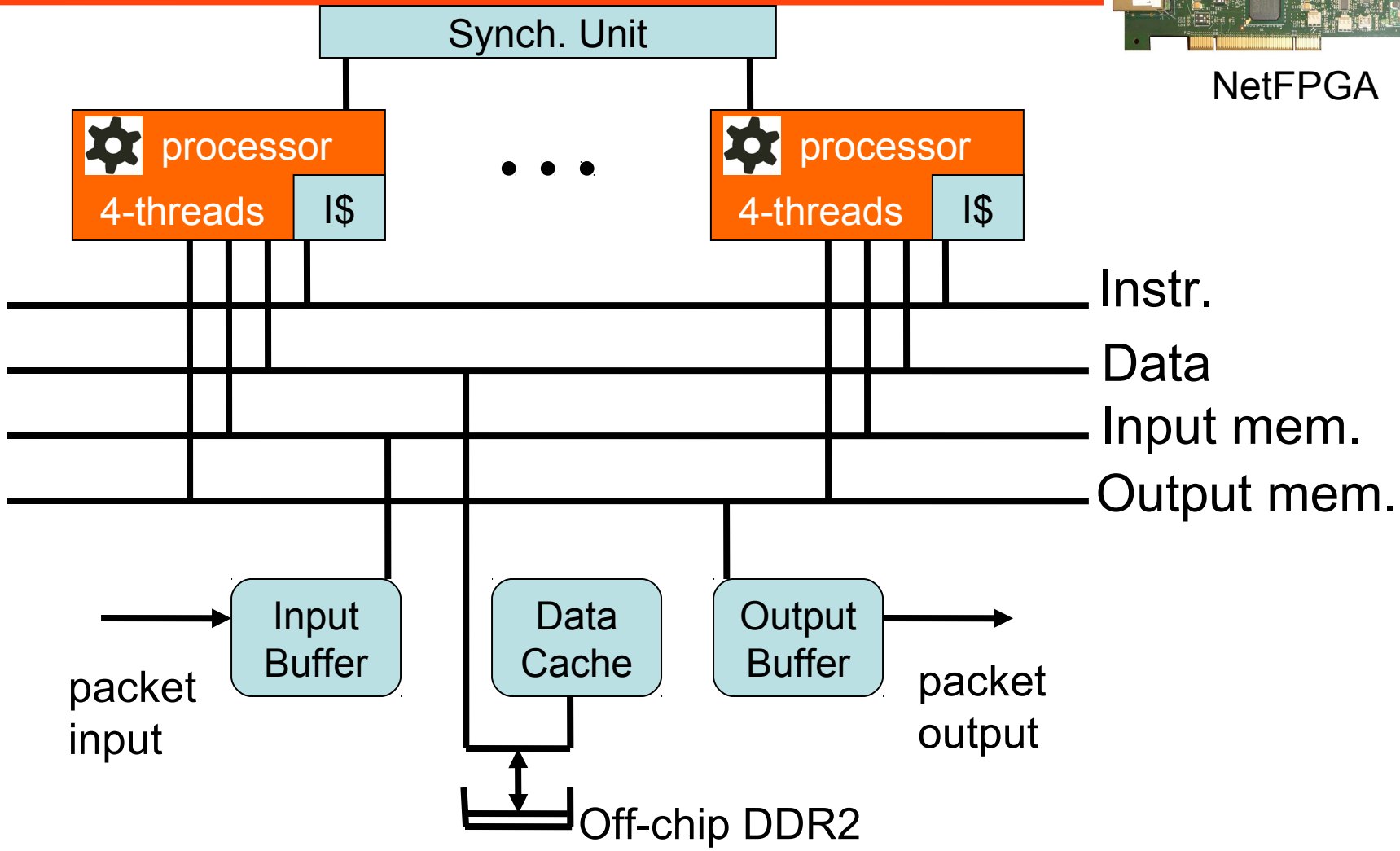




8 threads?

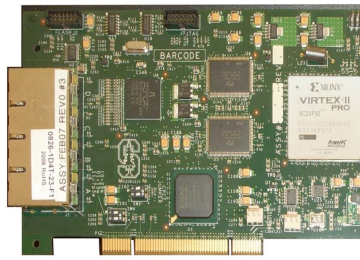
NetThreads: Our Base System



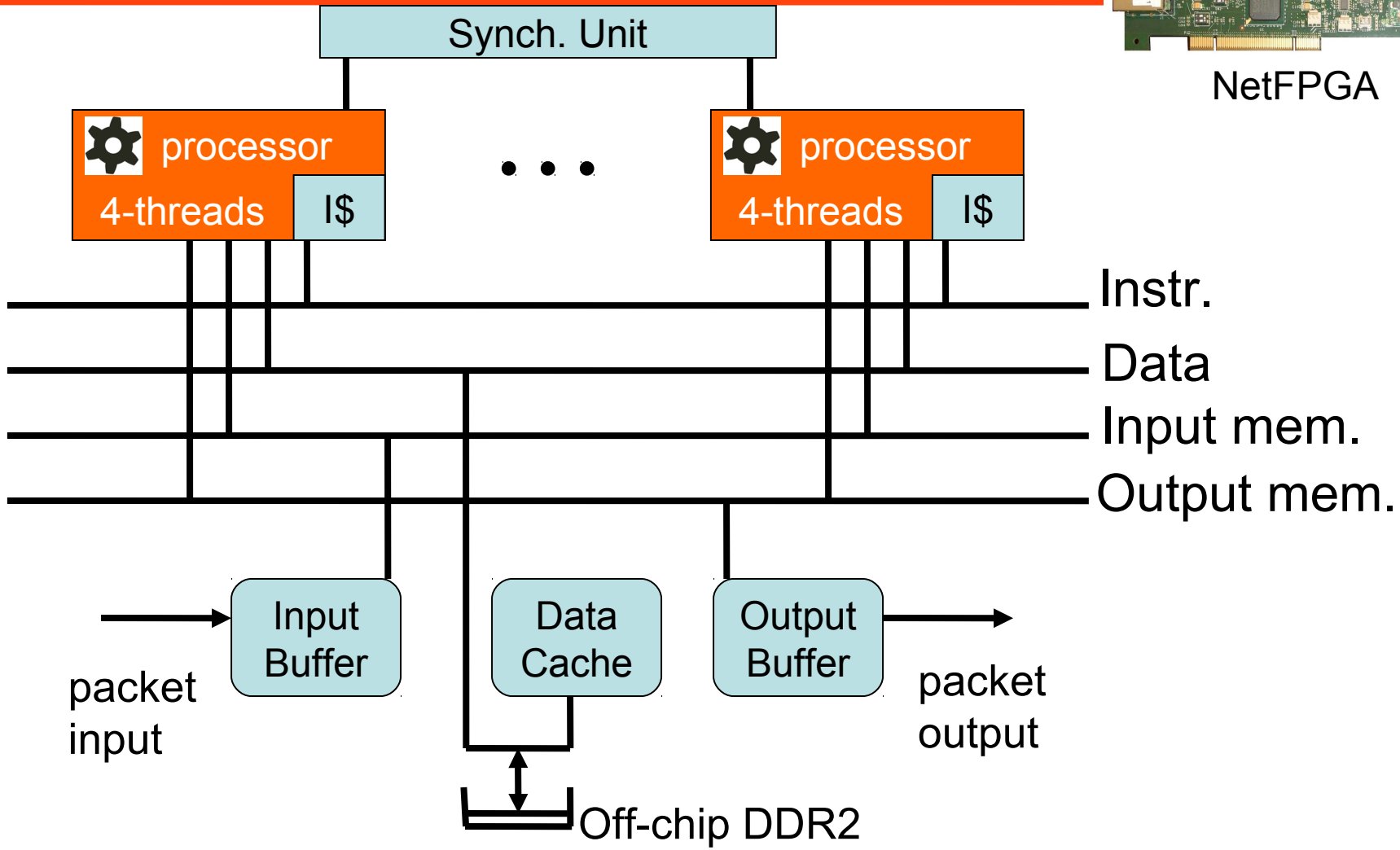


8 threads? Write 1 program, run on all threads!

NetThreads: Our Base System



NetFPGA



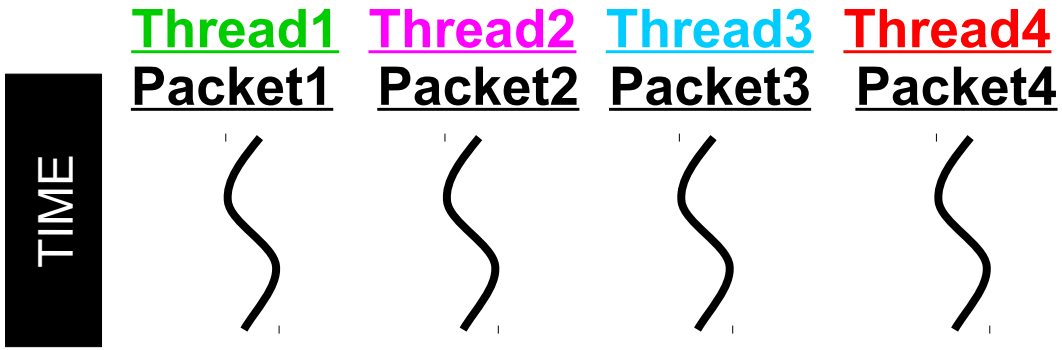
8 threads? Write 1 program, run on all threads!

Released online: Google netfpga+netthreads

Parallelizing Stateful Applications

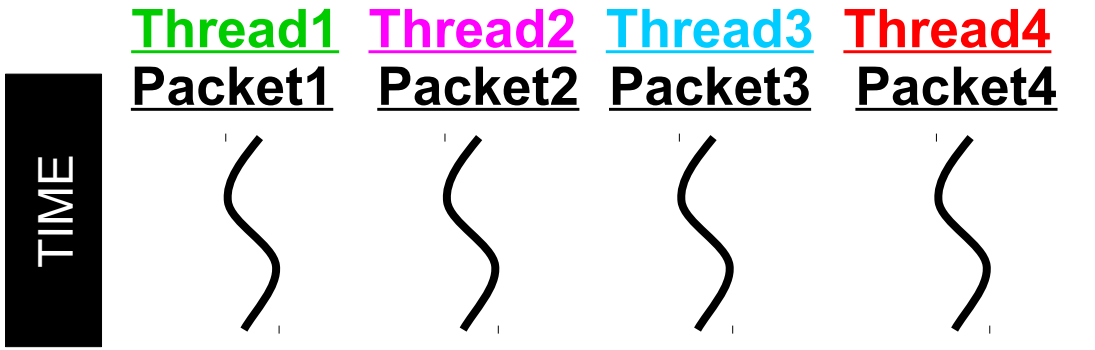
Ideal scenario:

Packets are data-independent and are processed in parallel

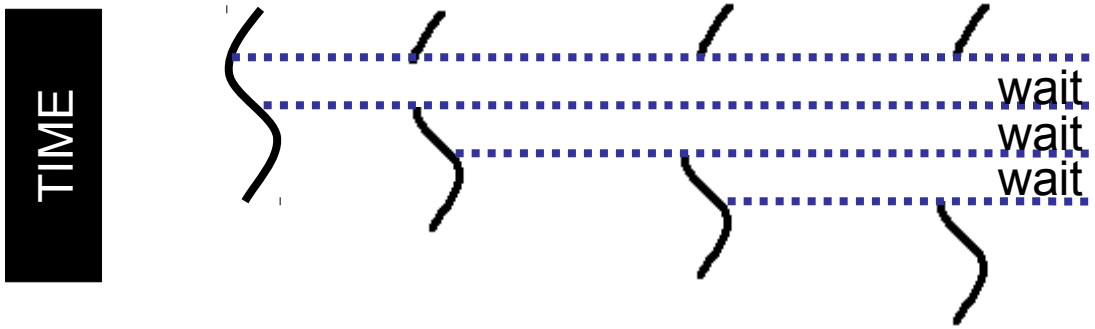


Parallelizing Stateful Applications

Ideal scenario:
Packets are data-independent and are processed in parallel

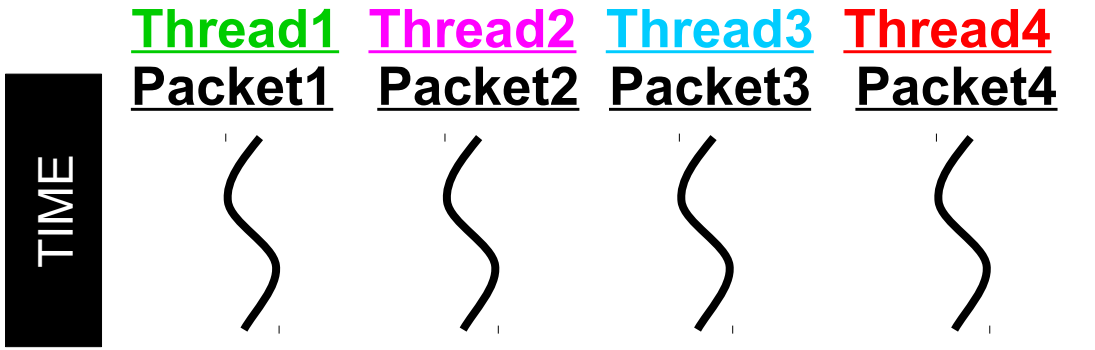


Reality:
Programmers need to insert locks **in case** there is a dependence

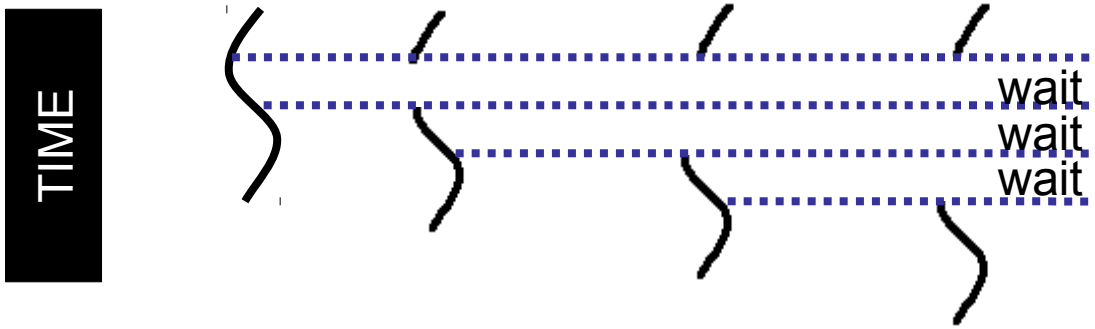


Parallelizing Stateful Applications

Ideal scenario:
Packets are data-independent and are processed in parallel



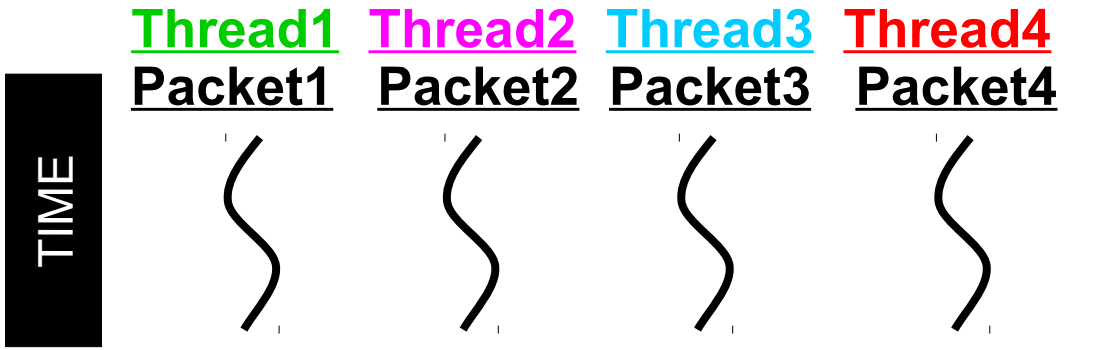
Reality:
Programmers need to insert locks **in case** there is a dependence



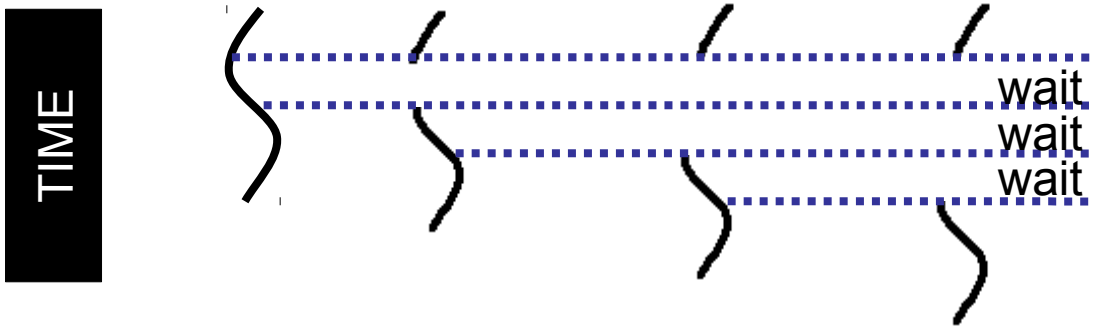
Experimental result: Synchronizing packet processing threads with fine/medium-grained global locks is overly-conservative 80-90% of the time [ANCS'10]

Parallelizing Stateful Applications

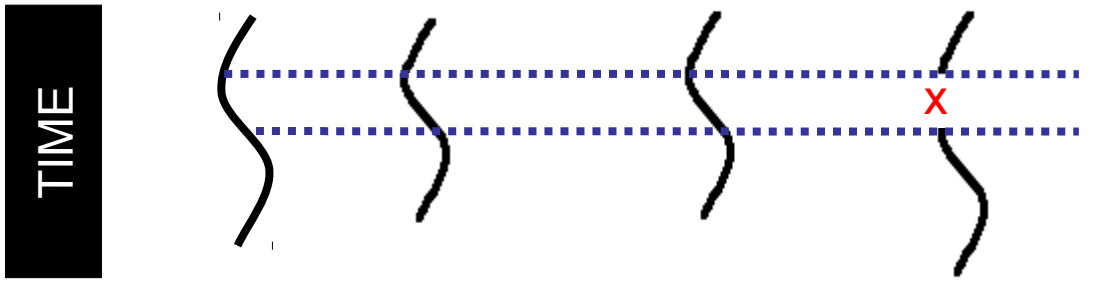
Ideal scenario:
Packets are data-independent and are processed in parallel



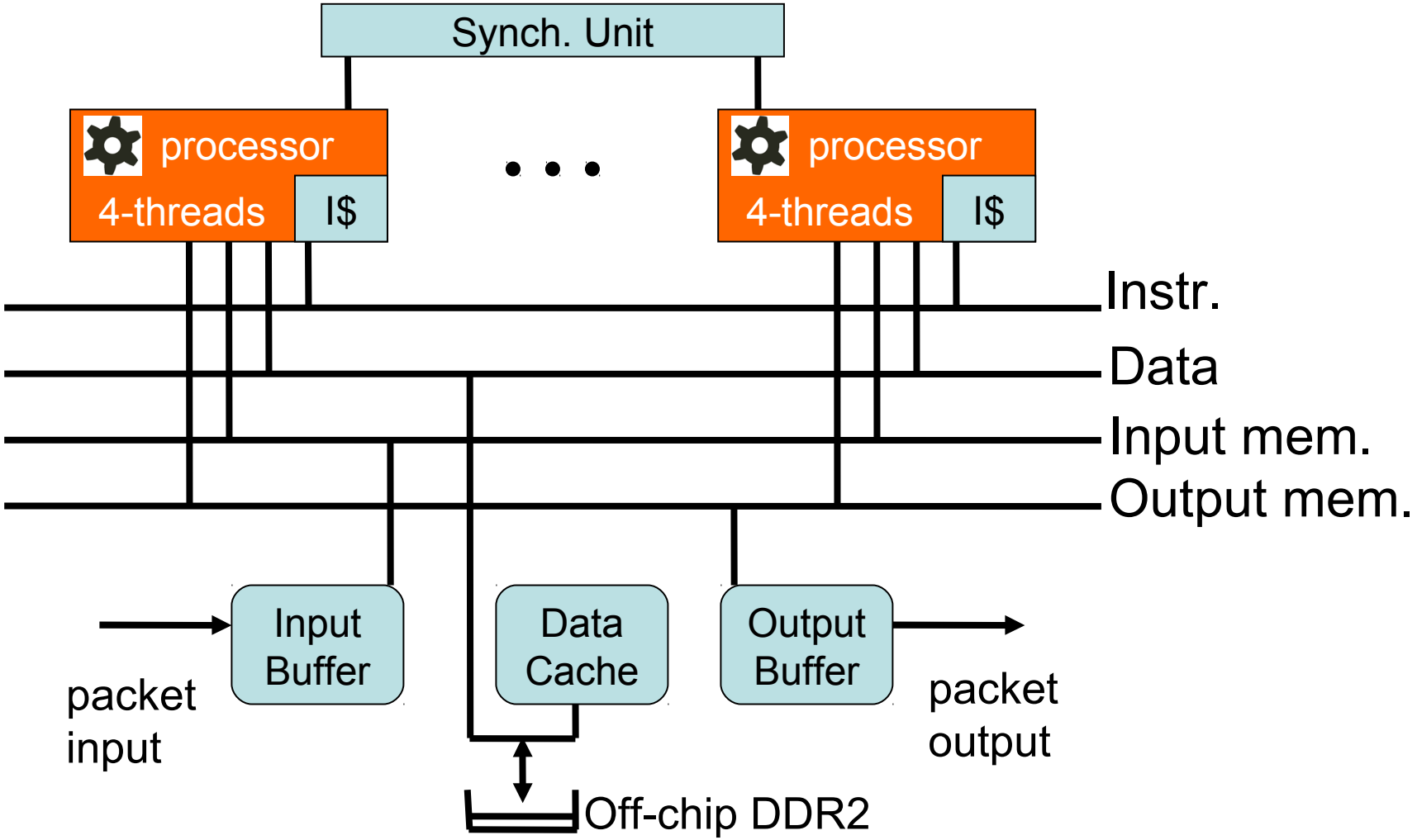
Reality:
Programmers need to insert locks **in case** there is a dependence



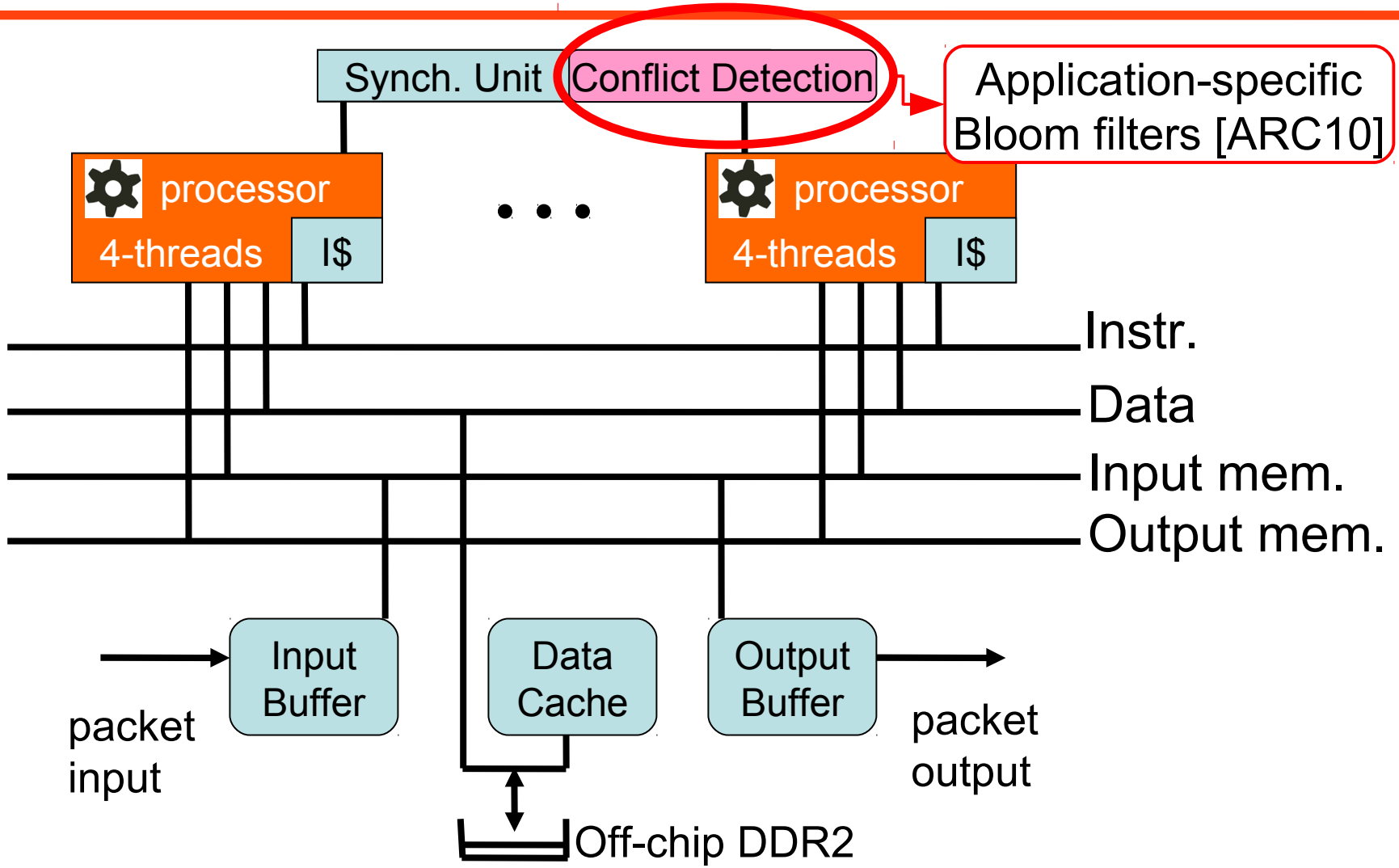
Transactional memory
Data-independent packets are processed in parallel



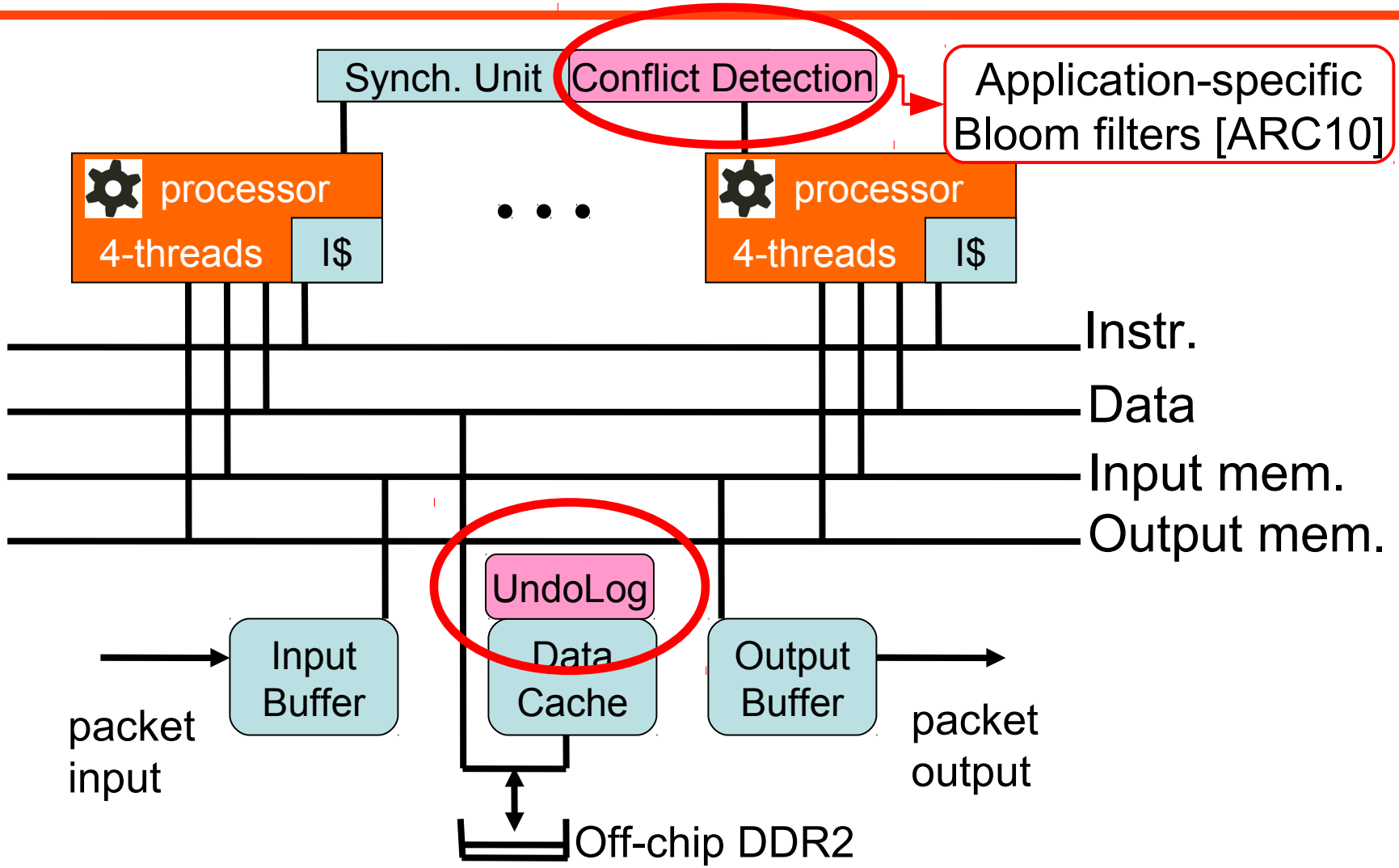
NetTM: extending NetThreads for TM



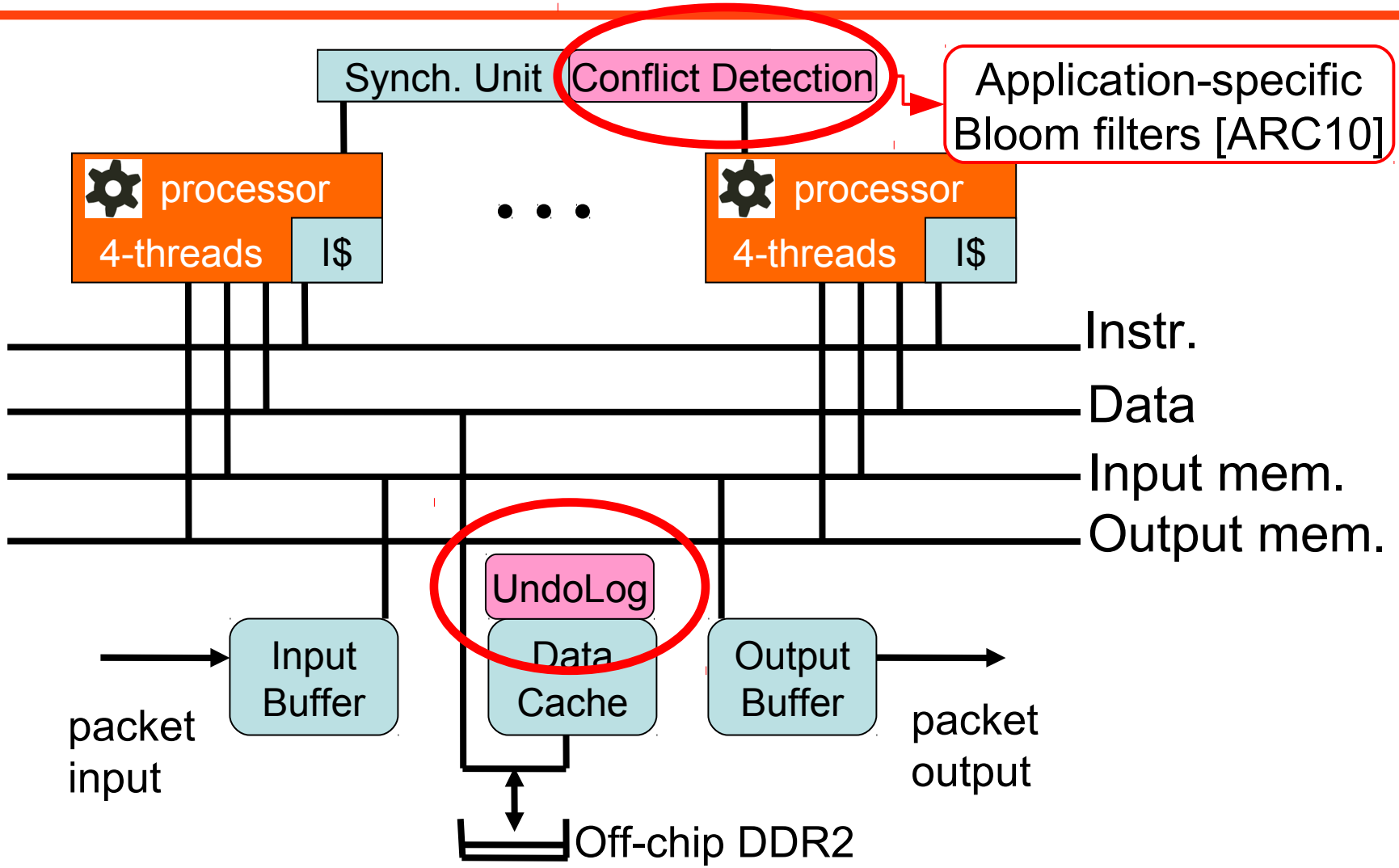
NetTM: extending NetThreads for TM



NetTM: extending NetThreads for TM

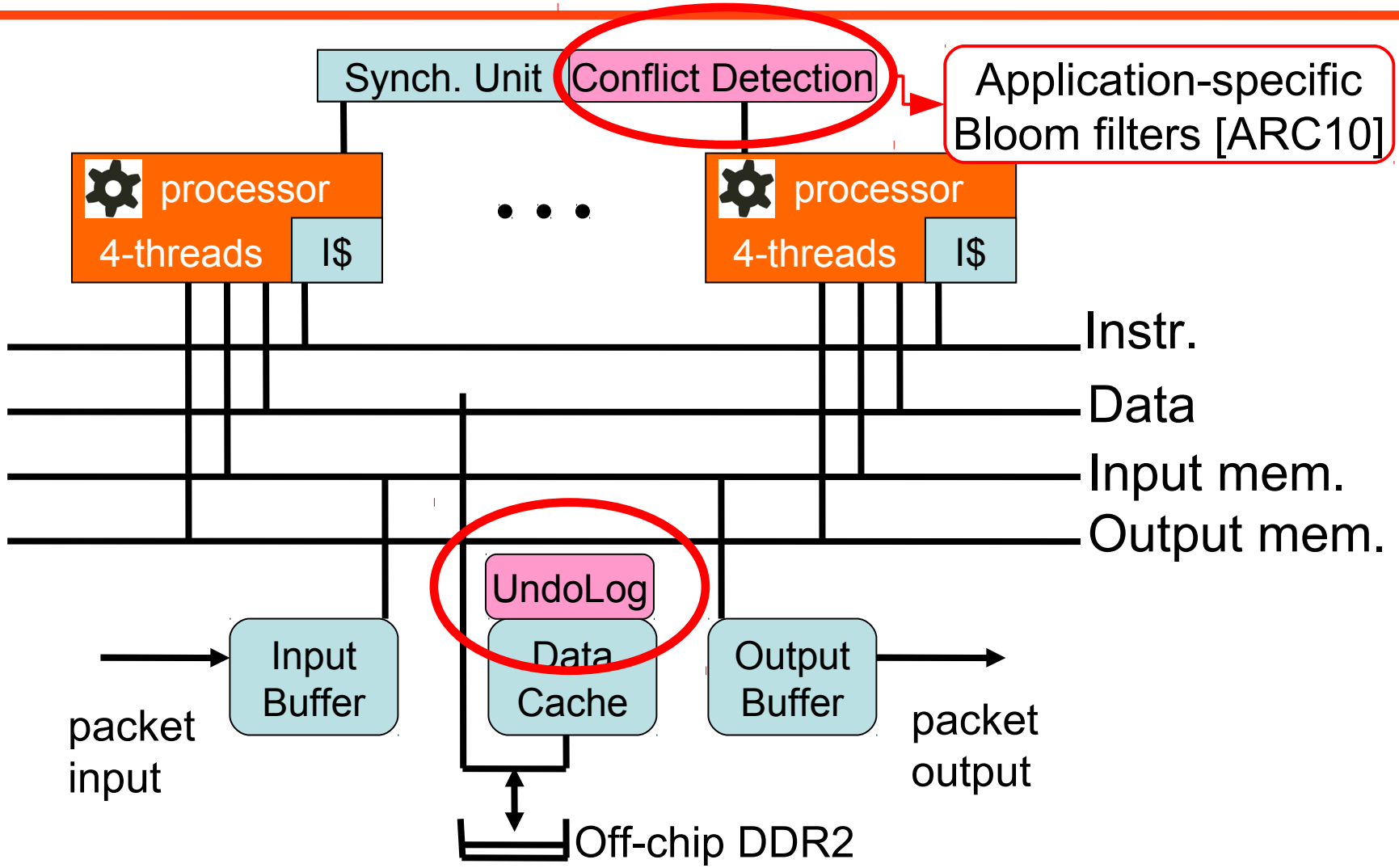


NetTM: extending NetThreads for TM



- 1K words speculative writes buffered per thread

NetTM: extending NetThreads for TM



- 1K words speculative writes buffered per thread
- 4-LUT: +21% 16K BRAMs: +25% Preserved 125 MHz

Transactional Memory

- 1st HTM implementation tightly integrated with soft processors

Transactional Memory

- 1st HTM implementation tightly integrated with soft processors
- Supports conventional locks and TM without code modification!

Transactional Memory

- 1st HTM implementation tightly integrated with soft processors
- Supports conventional locks and TM without code modification!
- Can extract optimistic parallelism across packets
  - Improves benchmark throughput: +6%, +54%, +57%

Transactional Memory

- 1st HTM implementation tightly integrated with soft processors
- Supports conventional locks and TM without code modification!
- Can extract optimistic parallelism across packets
  - Improves benchmark throughput: +6%, +54%, +57%
- Coarse critical sections and deadlock avoidance simplify program

Transactional Memory

- 1st HTM implementation tightly integrated with soft processors
- Supports conventional locks and TM without code modification!
- Can extract optimistic parallelism across packets
  - Improves benchmark throughput: +6%, +54%, +57%
- Coarse critical sections and deadlock avoidance simplify program
- Processor and conflict detection integration works well on FPGA

Transactional Memory

- 1st HTM implementation tightly integrated with soft processors
- Supports conventional locks and TM without code modification!
- Can extract optimistic parallelism across packets
  - Improves benchmark throughput: +6%, +54%, +57%
- Coarse critical sections and deadlock avoidance simplify program
- Processor and conflict detection integration works well on FPGA

Future work: scale to more cores on newer FPGA/NetFPGA!

Transactional Memory

- 1st HTM implementation tightly integrated with soft processors
- Supports conventional locks and TM without code modification!
- Can extract optimistic parallelism across packets
  - Improves benchmark throughput: +6%, +54%, +57%
- Coarse critical sections and deadlock avoidance simplify program
- Processor and conflict detection integration works well on FPGA

Future work: scale to more cores on newer FPGA/NetFPGA!

NetTM and NetThreads available online

Google : netfpga+netthreads

martinL@eecg.utoronto.ca