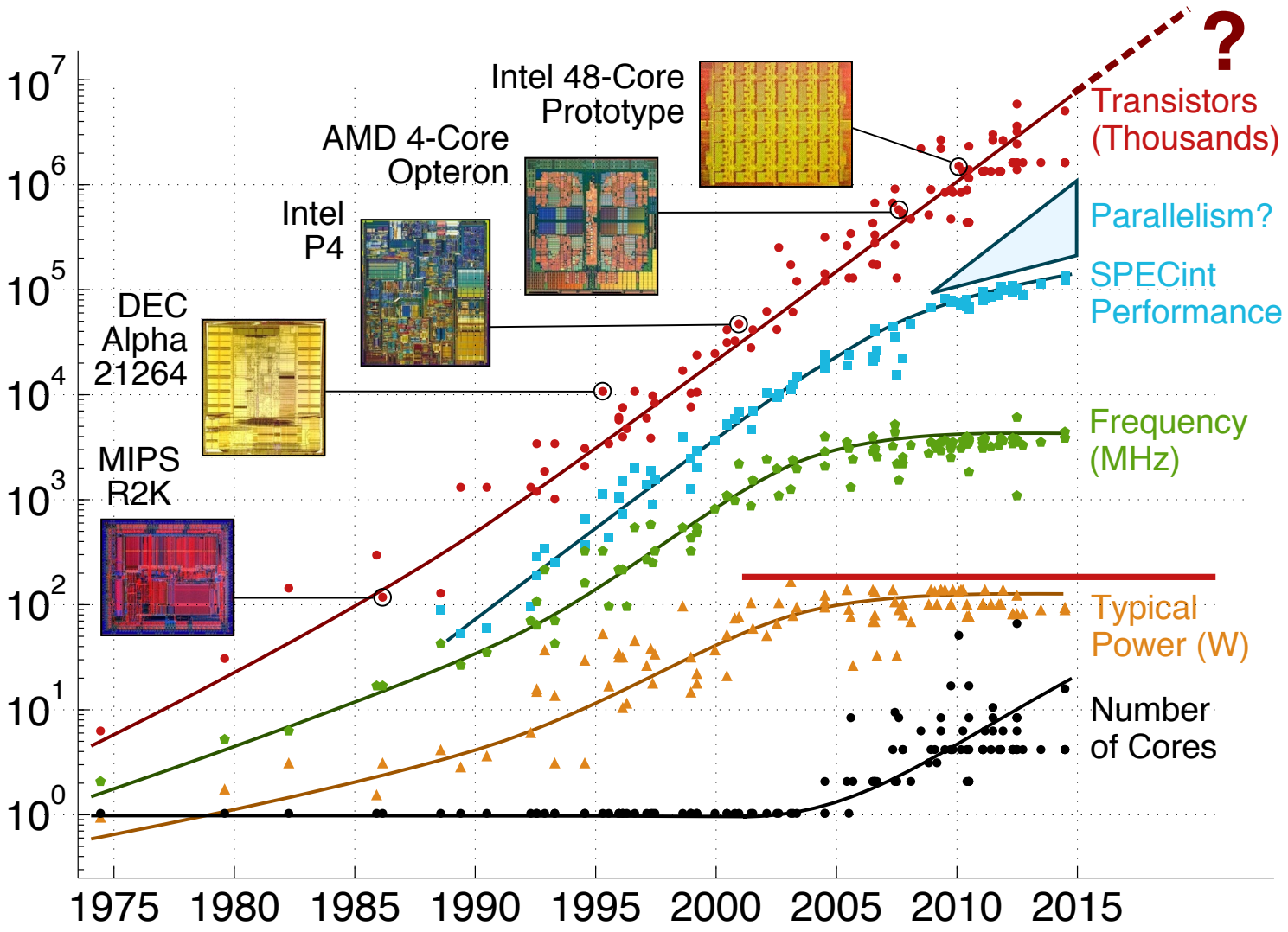
Energy-Efficient Parallel Computer Architecture

Christopher Batten

Computer Systems Laboratory
School of Electrical and Computer Engineering
Cornell University

Fall 2014

Motivating Trends in Computer Architecture

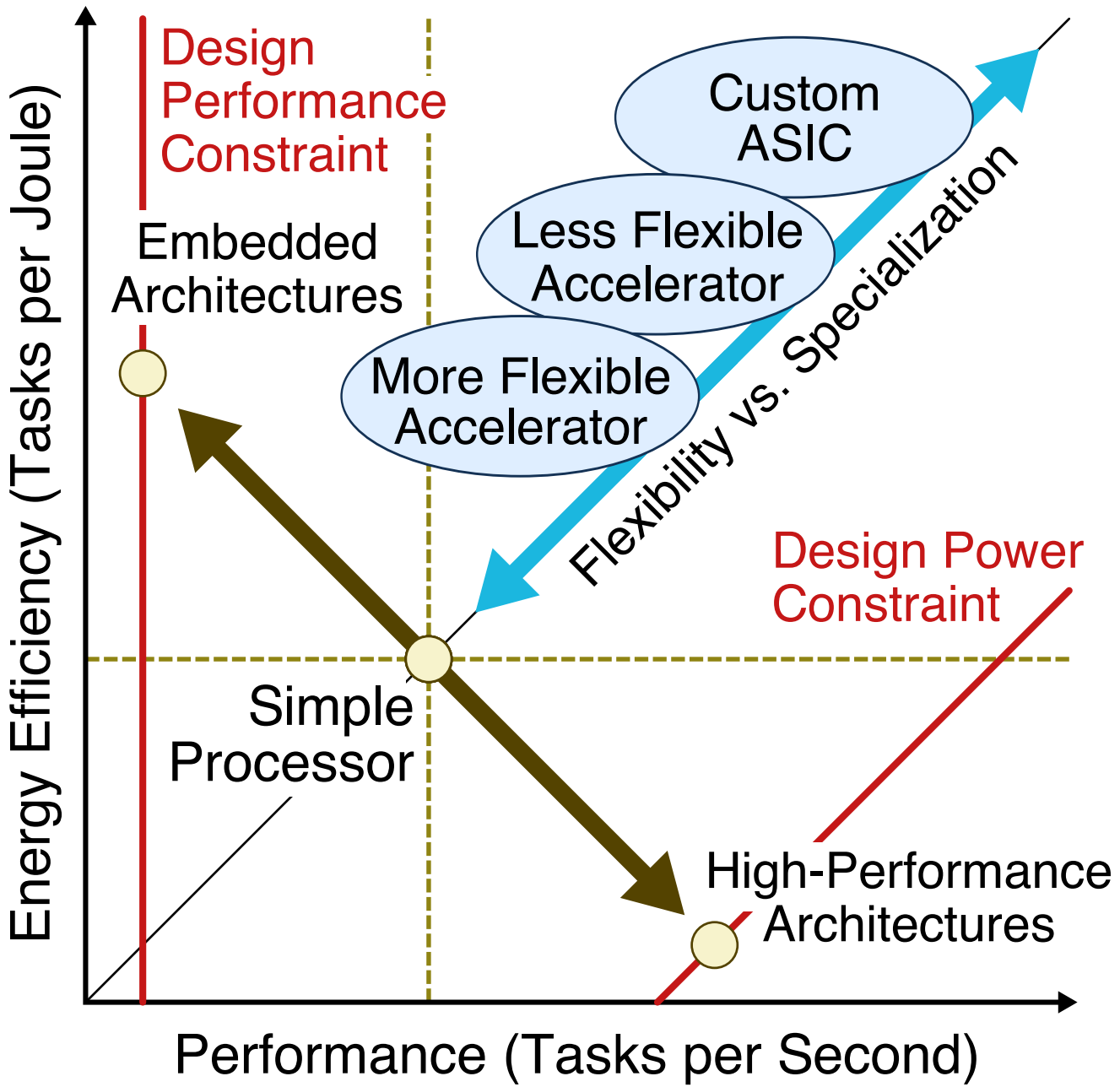


Data collected by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, C. Batten

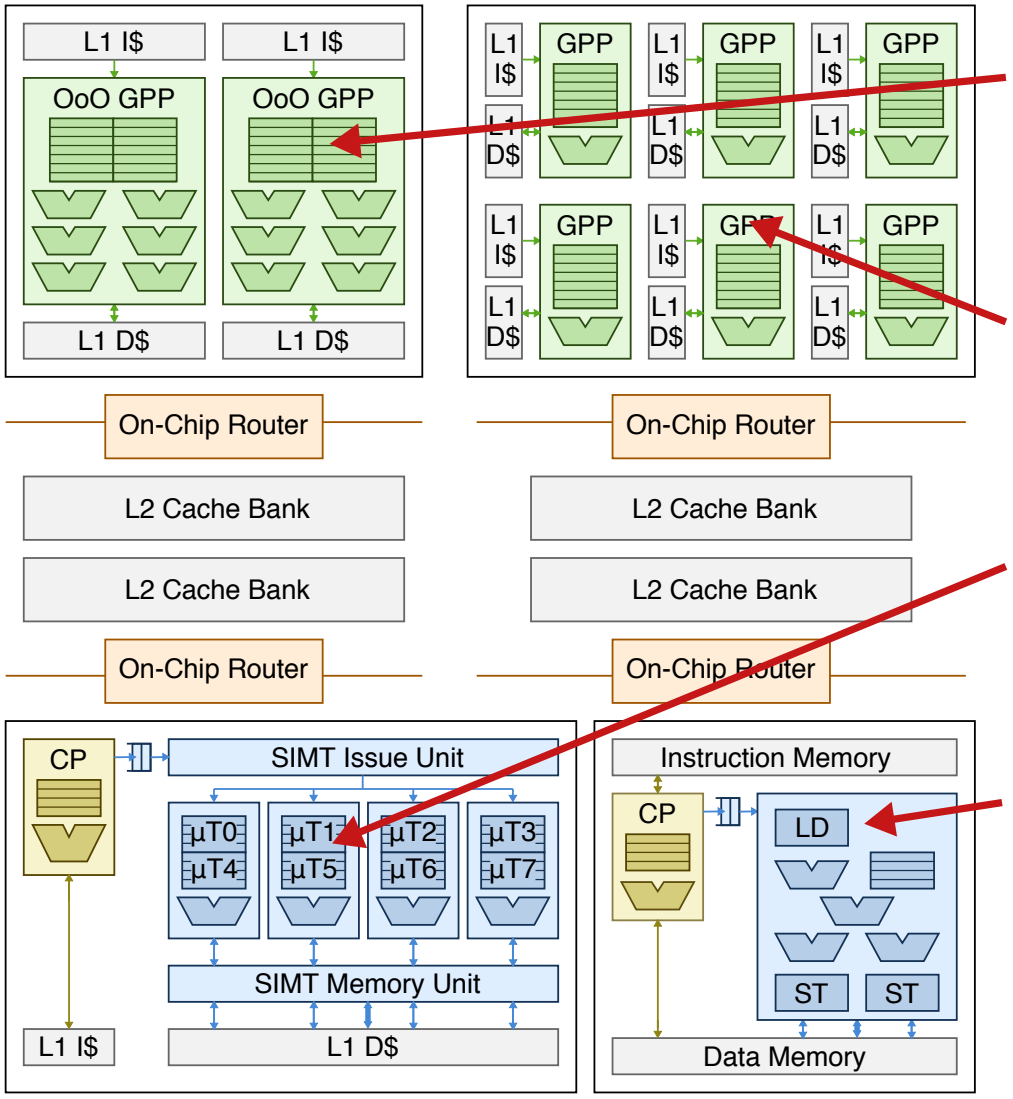
Trend 1
Power & energy constrain all systems

Trend 2
Transition to multicore processors

Trend 3
Inevitable end of Moore's law


Design
Performance
Constraint
Embedded
Architectures
Custom
ASIC
Less Flexible
Accelerator
More Flexible
Accelerator
Flexibility vs. Specialization
Design Power
Constraint
Simple
Processor
High-Performance
Architectures
Energy Efficiency (Tasks per Joule)
Performance (Tasks per Second)

Fine-Grain Heterogeneous Architectures



Latency-Optimized Tile:
few large out-of-order cores

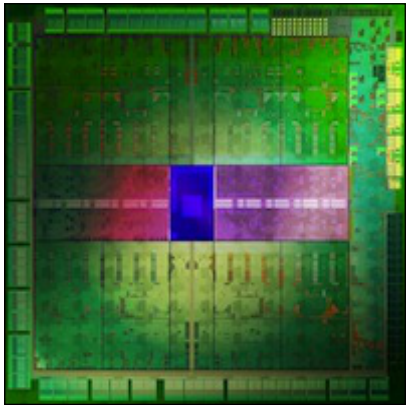
Throughput-Optimized Tile:
many small in-order cores

Data-Parallel-Optimized Tile:
flexible data-parallel accelerator

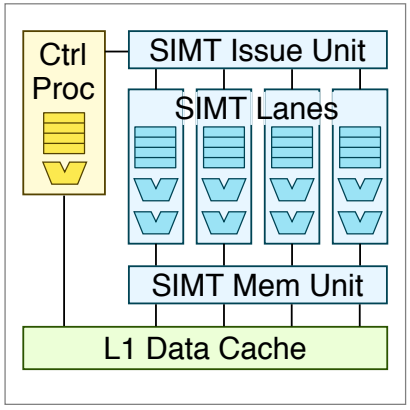
Application-Optimized Tile:
instruction set specialization;
flexible algorithm or
data-structure accelerators

Projects Within the Batten Research Group

GPGPU Microarchitecture



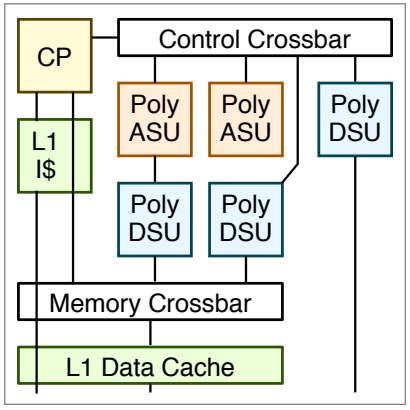
XLOOPS: Specialization for Loop Patterns



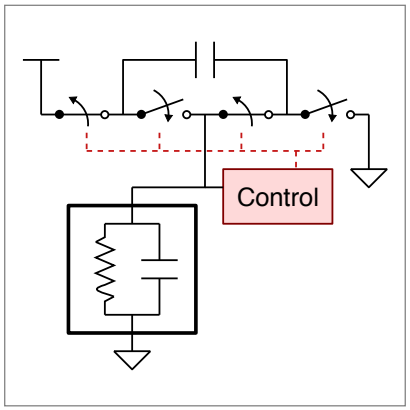
XPC: Explicit-Parallel-Call Architectures

```
instrA
instrB
pcall n, func1
...
psync   func1:
          instrC
          pcall func2
          ...
          pret
```

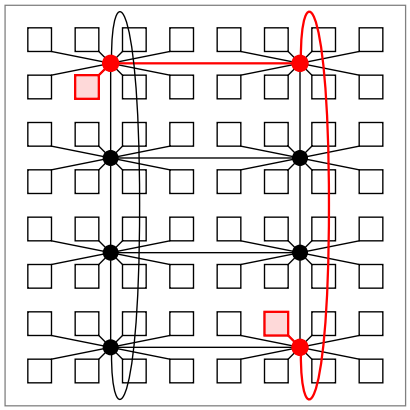
Polymorphic Hardware Specialization



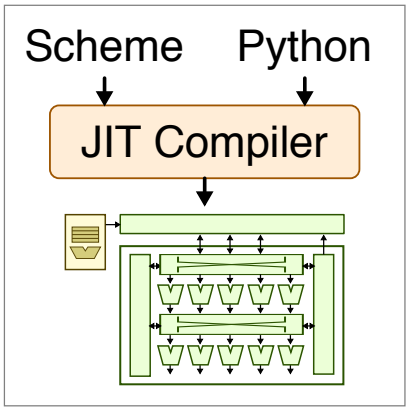
Reconfigurable Power Distribution Networks



Complexity Effective On-Chip Networks



Hardware Acceleration for Dynamic Languages

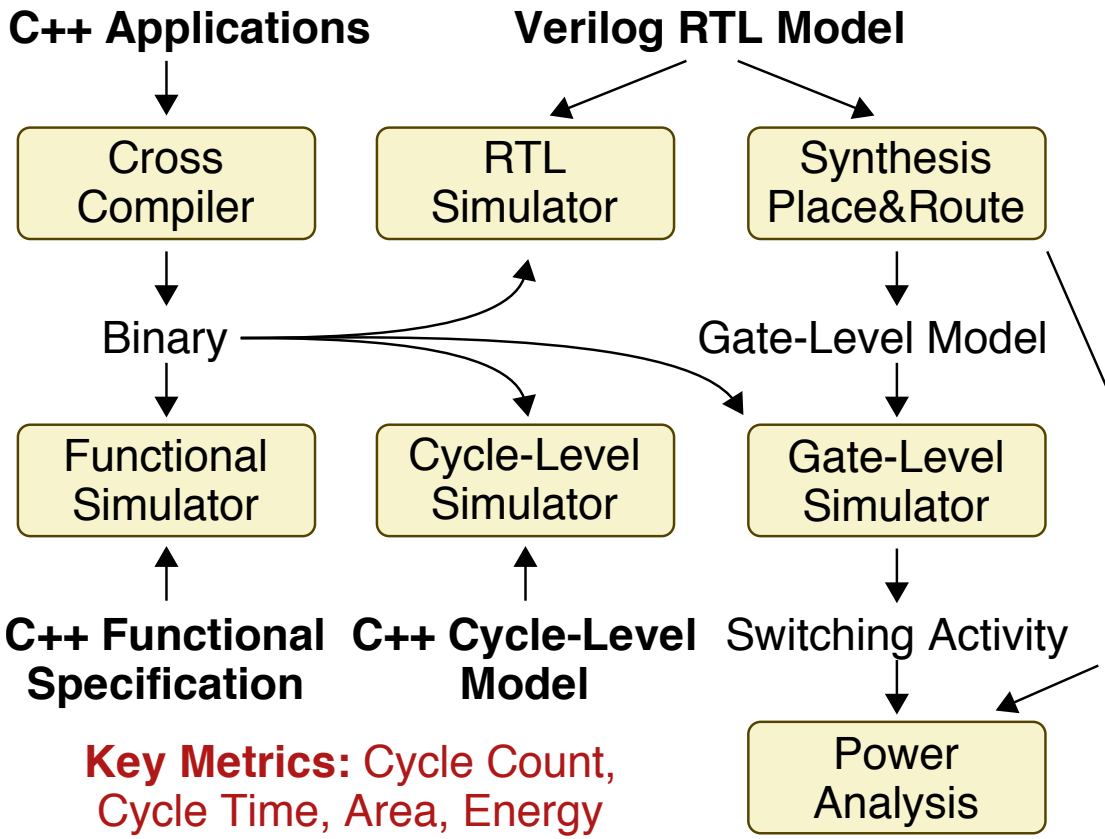


Python-Based Hardware Modeling

```
class Reg (Model):

def __init__( s ):
 s.in  = InPort(32)
 s.out = OutPort(32)

def elaborate( s ):

 @s.posedge_clk
 def seq_logic():
  s.out.next = s.in
```

Vertically Integrated Research Methodology

Our research involves reconsidering all aspects of the computing stack including applications, programming frameworks, compiler optimizations, runtime systems, instruction set design, microarchitecture design, VLSI implementation, and hardware design methodologies



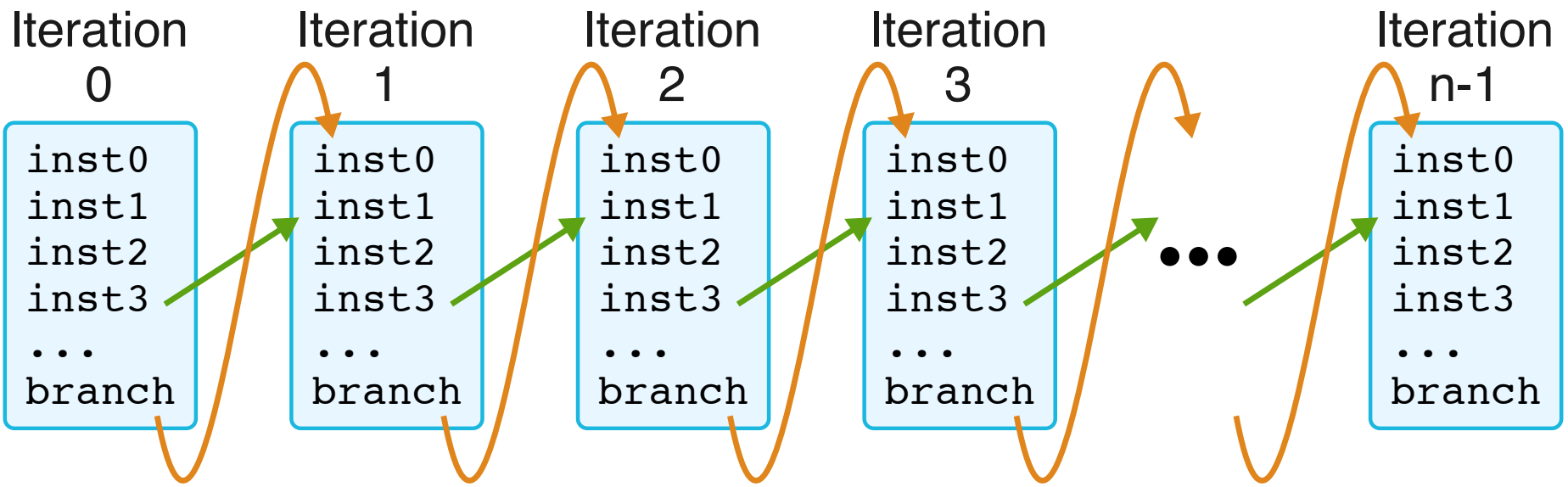
Experimenting with full-chip layout, FPGA prototypes, and test chips is a key part of our research methodology

XLOOPS: Architectural Specialization for Inter-Iteration Loop Dependence Patterns

Shreesha Srinath, Berkin Ilbeyi, Mingxing Tan, Gai Liu, Zhiru Zhang, and Christopher Batten

47th ACM/IEEE Int'l Symp. on Microarchitecture (MICRO)
Cambridge, UK, Dec. 2014

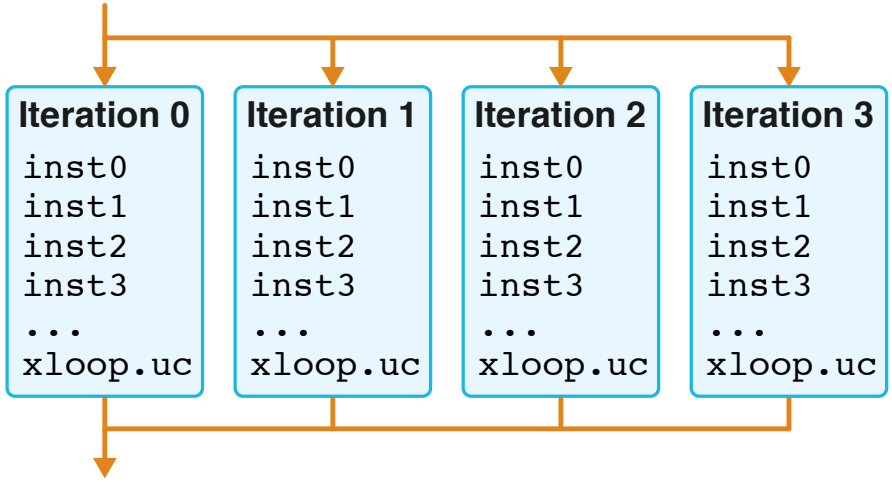
Inter-Iteration Loop Dependence Patterns



Explicit loop specialization (XLOOPS) elegantly encodes inter-iteration loop dependence patterns in the instruction set and enables traditional, specialized, and adaptive execution.

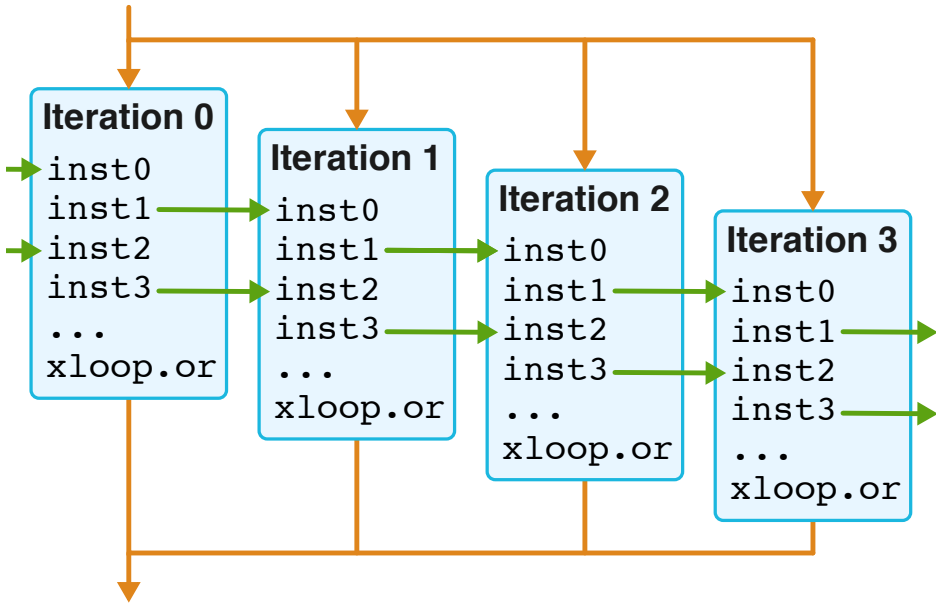
XLOOPS ISA: Unordered Concurrent

```
for ( i=0; i<N; i++ )
  C[i] = A[i] * B[i]

loop:
  lw          r2, 0(rA)
  lw          r3, 0(rB)
  mul         r4, r2, r3
  sw          r4, 0(rC)
  addiu.xi    rA, 4
  addiu.xi    rB, 4
  addiu.xi    rC, 4
  addiu       r1, r1, 1
  xloop.uc    r1, rN, loop
```



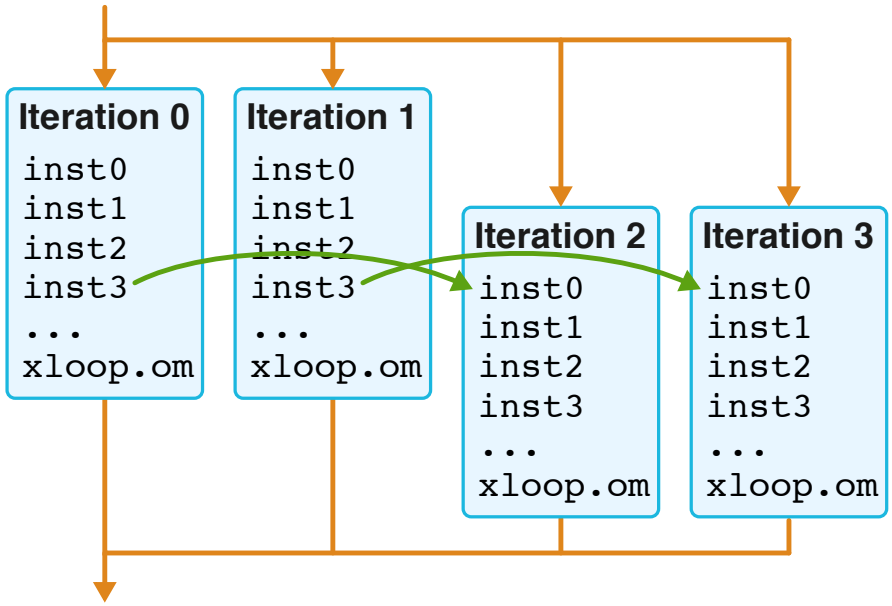
XLOOPS ISA: Ordered-Through-Registers

```
for ( X=0, i=0; i<N; i++ )
  X += A[i]
  B[i] = X

loop:
  lw          r2, 0(rA)
  addu        rX, r2, rX
  sw          rX, 0(rB)
  addiu.xi    rA, 4
  addiu.xi    rB, 4
  addiu       r1, r1, 1
  xloop.or    r1, rN, loop
```



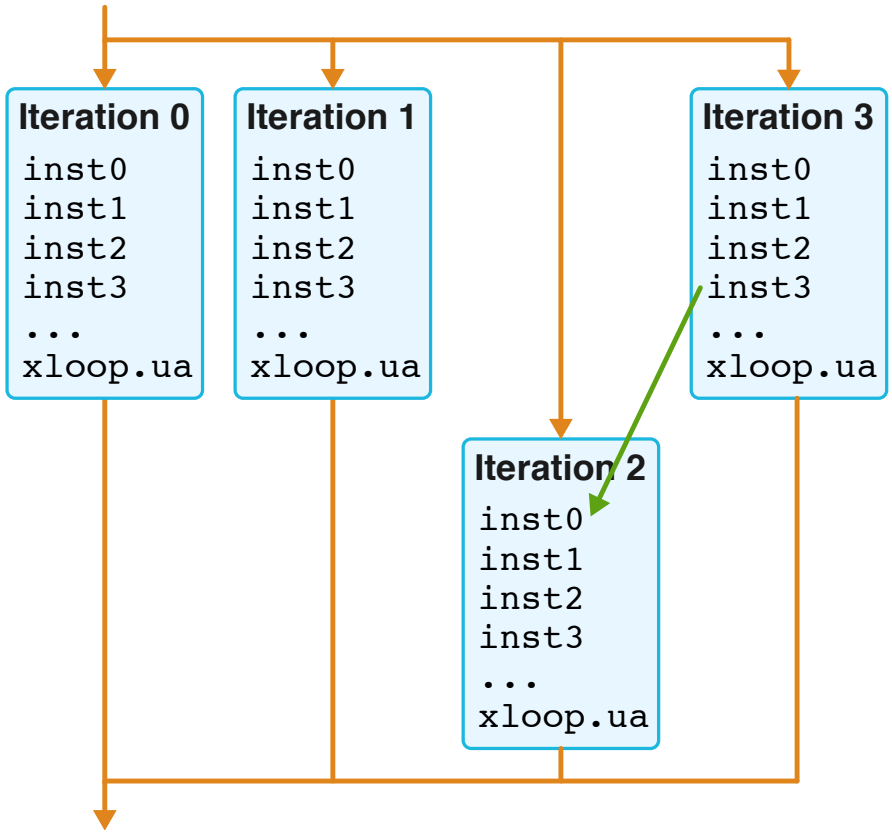
XLOOPS ISA: Ordered-Through-Memory

```
for ( i=K; i<N; i++ )
  A[i] = A[i] * A[i-K]

  # r1 = rK
  # r3 = rA + 4*rK
loop:
  lw          r4, 0(r3)
  lw          r5, 0(rA)
  mul         r6, r4, r5
  sw          r6, 0(r3)
  addiu.xi    r3, 4
  addiu.xi    rA, 4
  addiu       r1, r1, 1
  xloop.om    r1, rN, loop
```

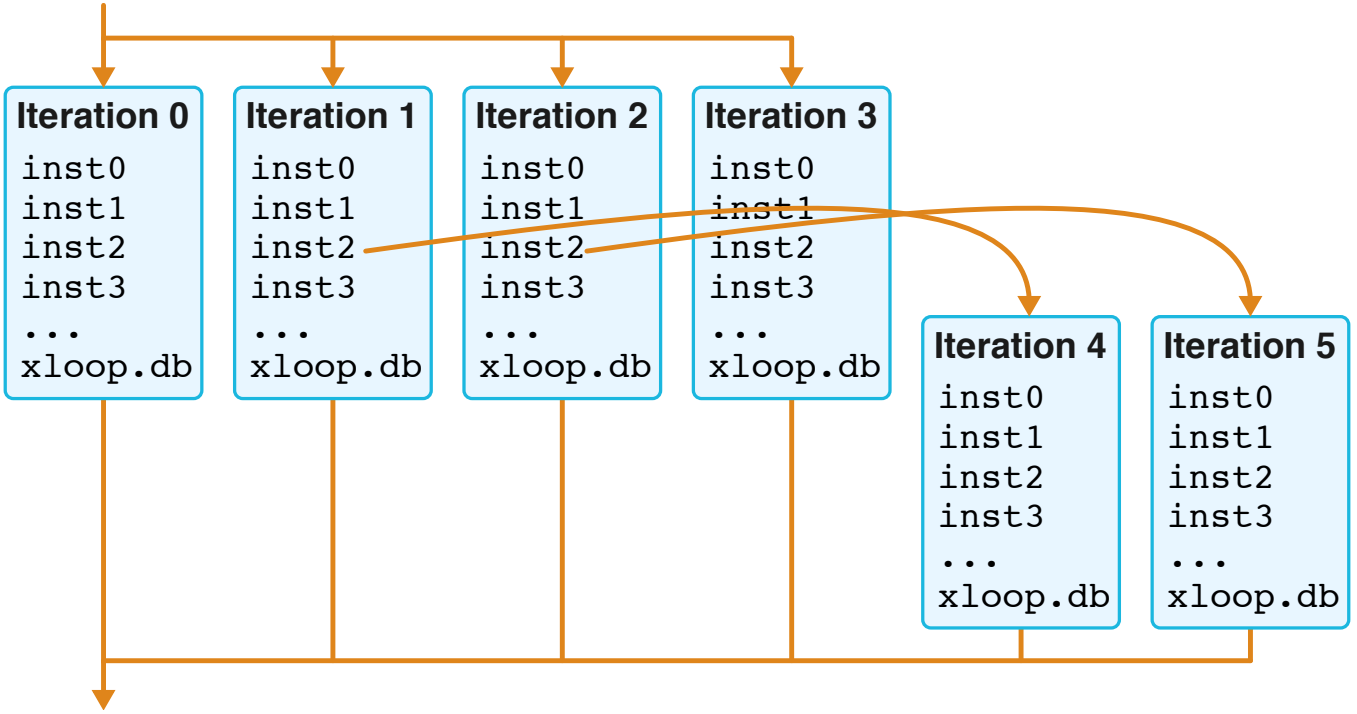


XLOOPS ISA: Unordered Atomic

```
for ( i=0; i<N; i++ )
  B[ A[i] ]++
  D[ C[i] ]++

loop:
  lw          r6, 0(rA)
  lw          r7, 0(r6)
  addiu       r7, r7, 1
  sw          r7, 0(r6)
  addiu.xi    rA, rA, 4
  ...
  addiu       r1, r1, 1
  xloop.ua    r1, rN, loop
```



XLOOPS ISA: Dynamic Bound



XLOOPS Compiler

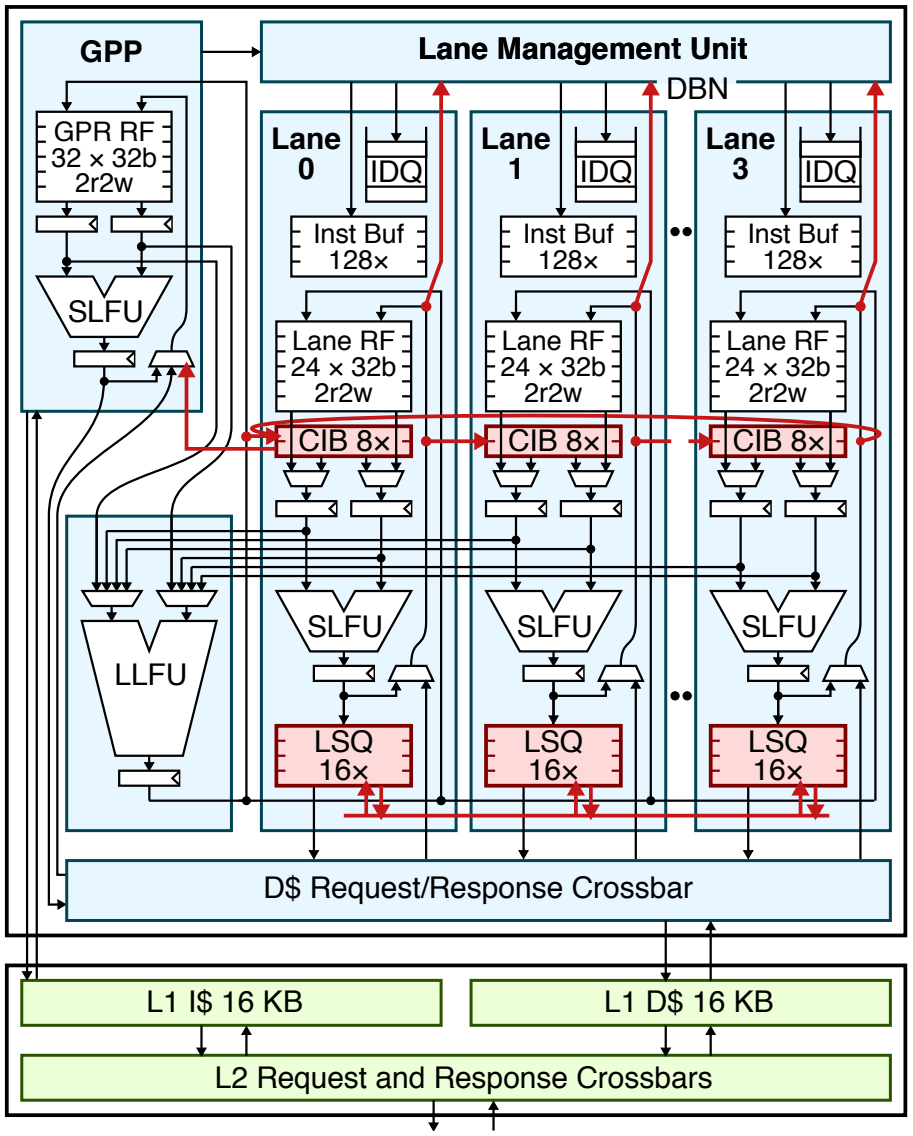
Kernel implementing Floyd-Warshall shortest path algorithm

```
for ( int k = 0; k < n; k++ )
  #pragma xloops ordered
  for ( int i = 0; i < n; i++ )
    #pragma xloops unordered
    for ( int j = 0; j < n; j++ )
      path[i][j] = min( path[i][j], path[i][k] + path[k][j] );
```

Kernel implementing greedy algorithm for maximal matching

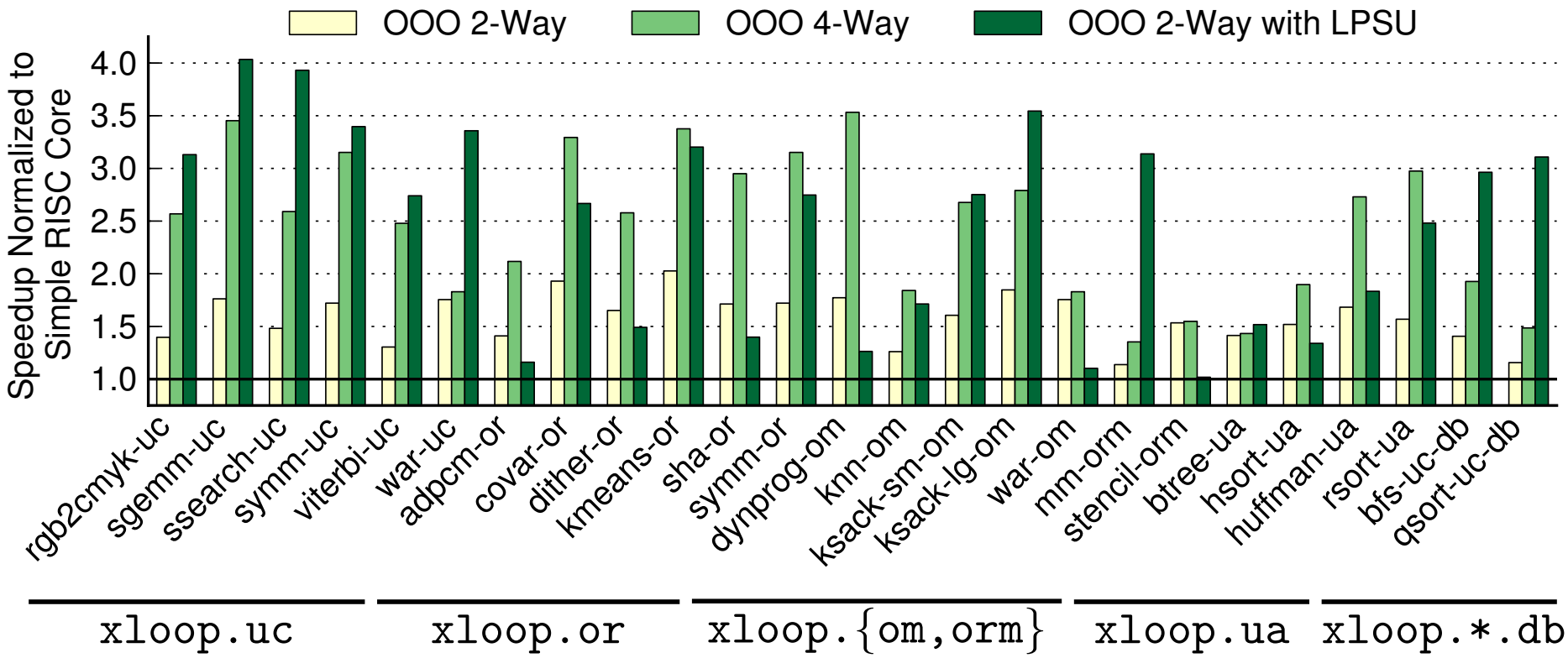
```
#pragma xloops ordered
for ( int i = 0; i < num_edges; i++ )
  int v = edges[i].v; int u = edges[i].u;
  if ( vertices[v] < 0 && vertices[u] < 0 )
    vertices[v] = u; vertices[u] = v; out[k++] = i;
```

XLOOPS Microarchitecture



- Elegant hardware/software abstraction enables efficient traditional execution on general-purpose processor
- Loop-pattern specialization unit enables specialized execution for improved performance and efficiency
- Adaptive execution uses two profiling phases to adaptively determine best location to run loop

XLOOPS Cycle-Level Performance Results



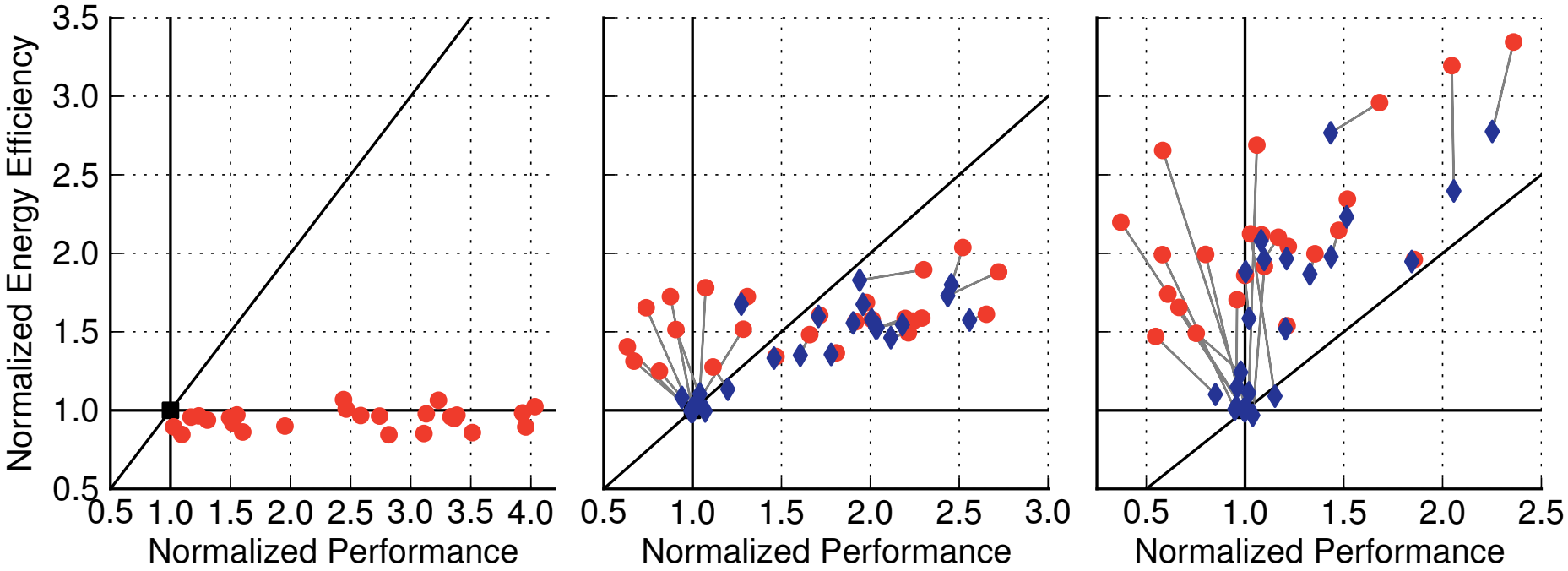
- XLOOPS vs. Simple Core : Always higher performance
- XLOOPS vs. OOO 2-way : Generally competitive or higher performance
- XLOOPS vs. OOO 4-way : Mixed results

XLOOPS Cycle-Level Energy-Efficiency Results

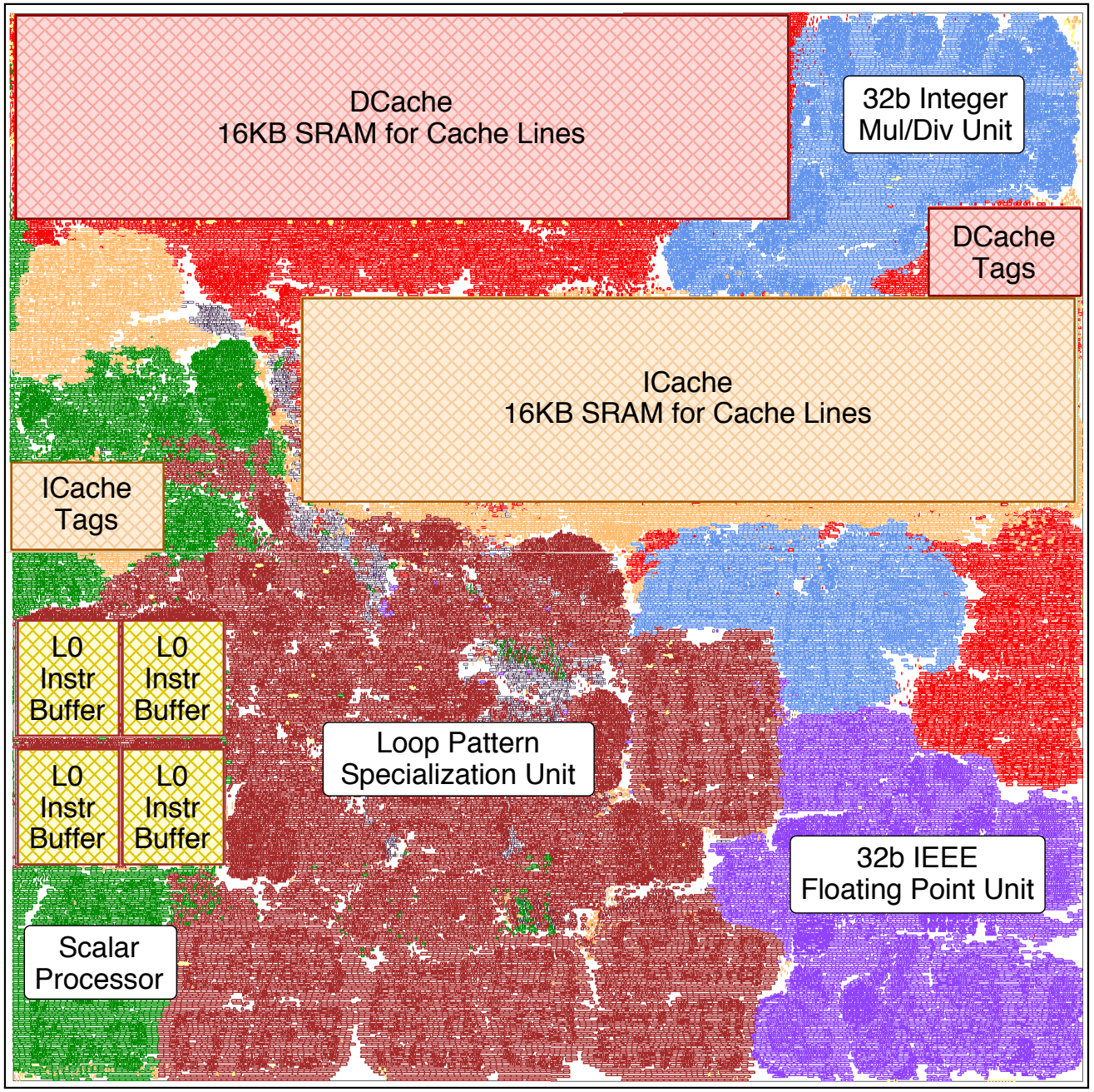
XLOOPS vs. Simple RISC Core

XLOOPS vs. OOO 2-Way

XLOOPS vs. OOO 4-Way

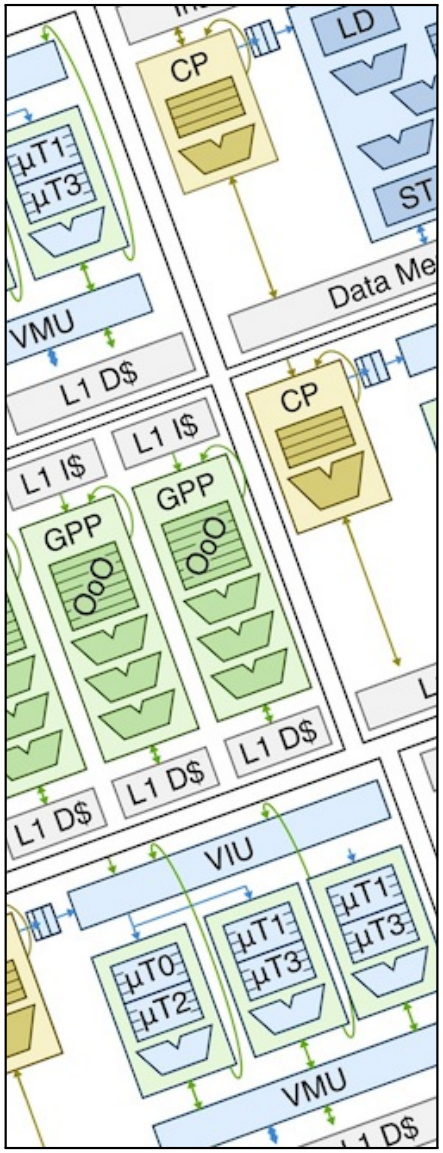


- XLOOPS vs. Simple Core : Competitive energy efficiency
- XLOOPS vs. OOO 2-way : Always more energy efficient
- XLOOPS vs. OOO 4-way : Always more energy efficient



VLSI Implementation

- TSMC 40 nm standard-cell-based implementation
- RISC scalar processor with 4-lane LPSU
- Supports `xloop.uc`
- Implemented in TSMC 40 nm using ASIC CAD toolflow
- ≈40% extra area compared to simple RISC processor



Take-Away Points

- We envision future computing systems using fine-grain heterogeneous architectures with a mix of tiles optimized for latency, throughput, classes of applications, or even a specific application
- We are using a vertically integrated research methodology to explore a variety of projects that will hopefully enable such fine-grain heterogeneous architectures