



Energy-Efficient Memory Systems

Krste Asanovic

MIT Laboratory for Computer Science



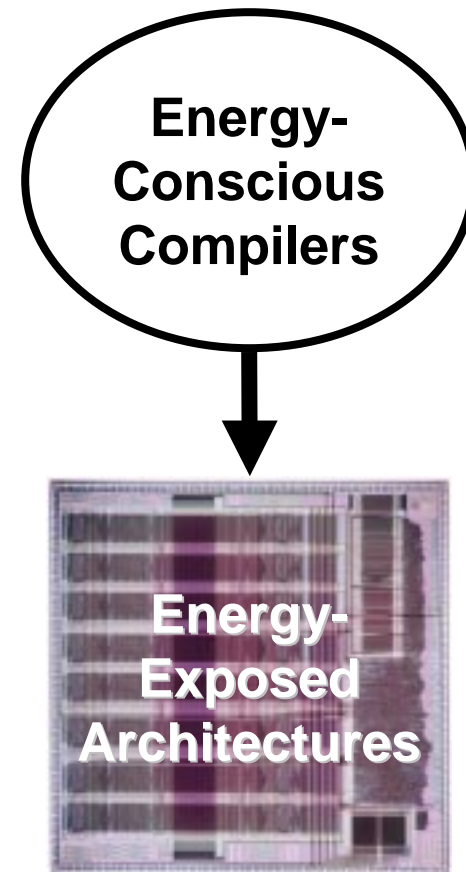
How to Build Low-Energy Malleable Caches?

- Column caching simple, flexible, based on standard associative cache
 - But associative cache has considerable energy overhead
- => *Develop new hardware/software interface to expose cache structure including energy consumption***



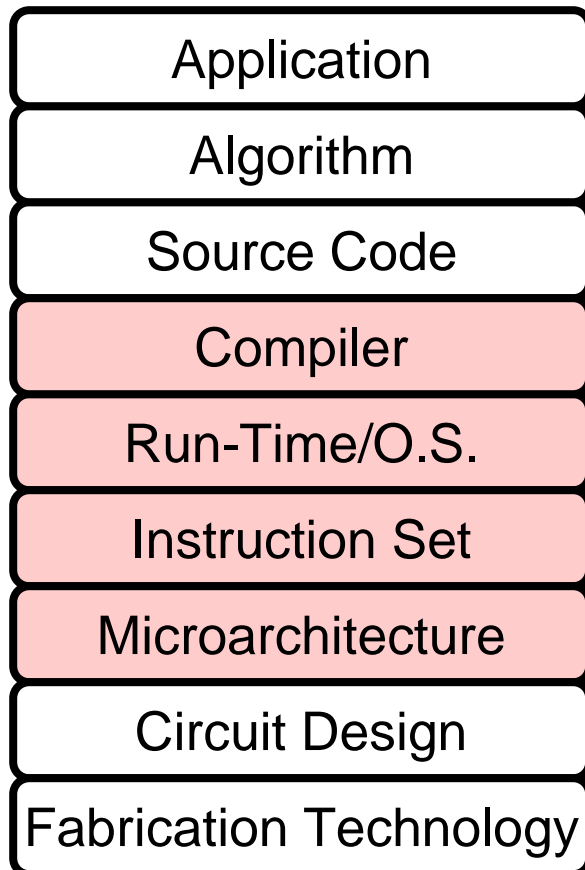
SCALE Project Background

*Improve Energy-Efficiency of
Programmable Processors
by Re-Examining
Hardware-Software Interface*





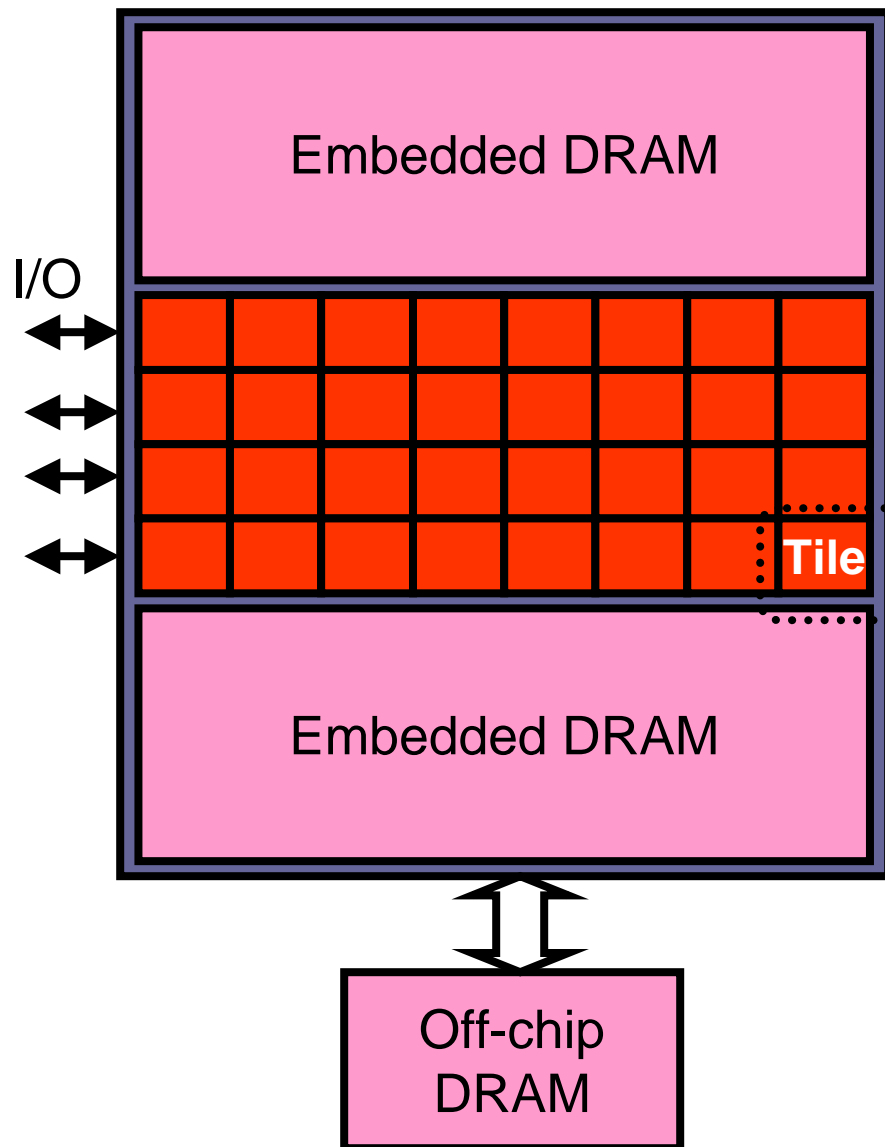
Can Optimize Energy Efficiency at Many Levels in System



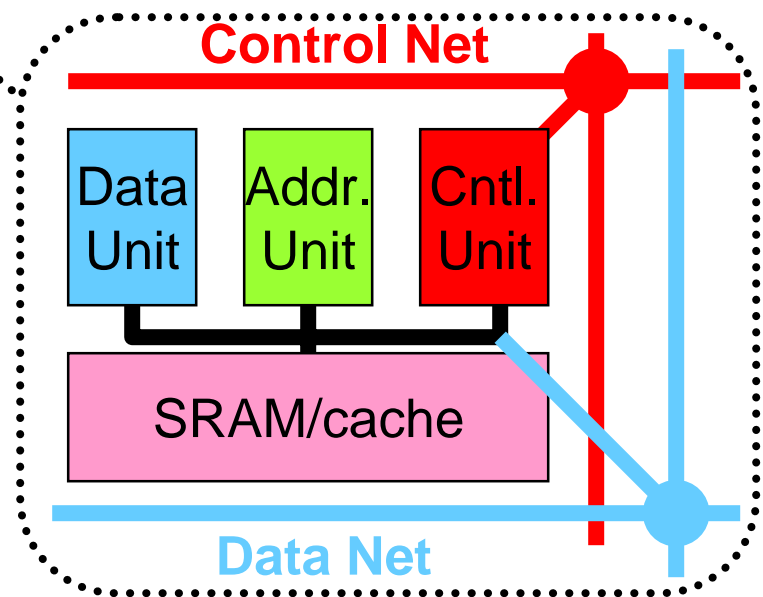
SCALE Focus Areas



SCALE Processor Overview

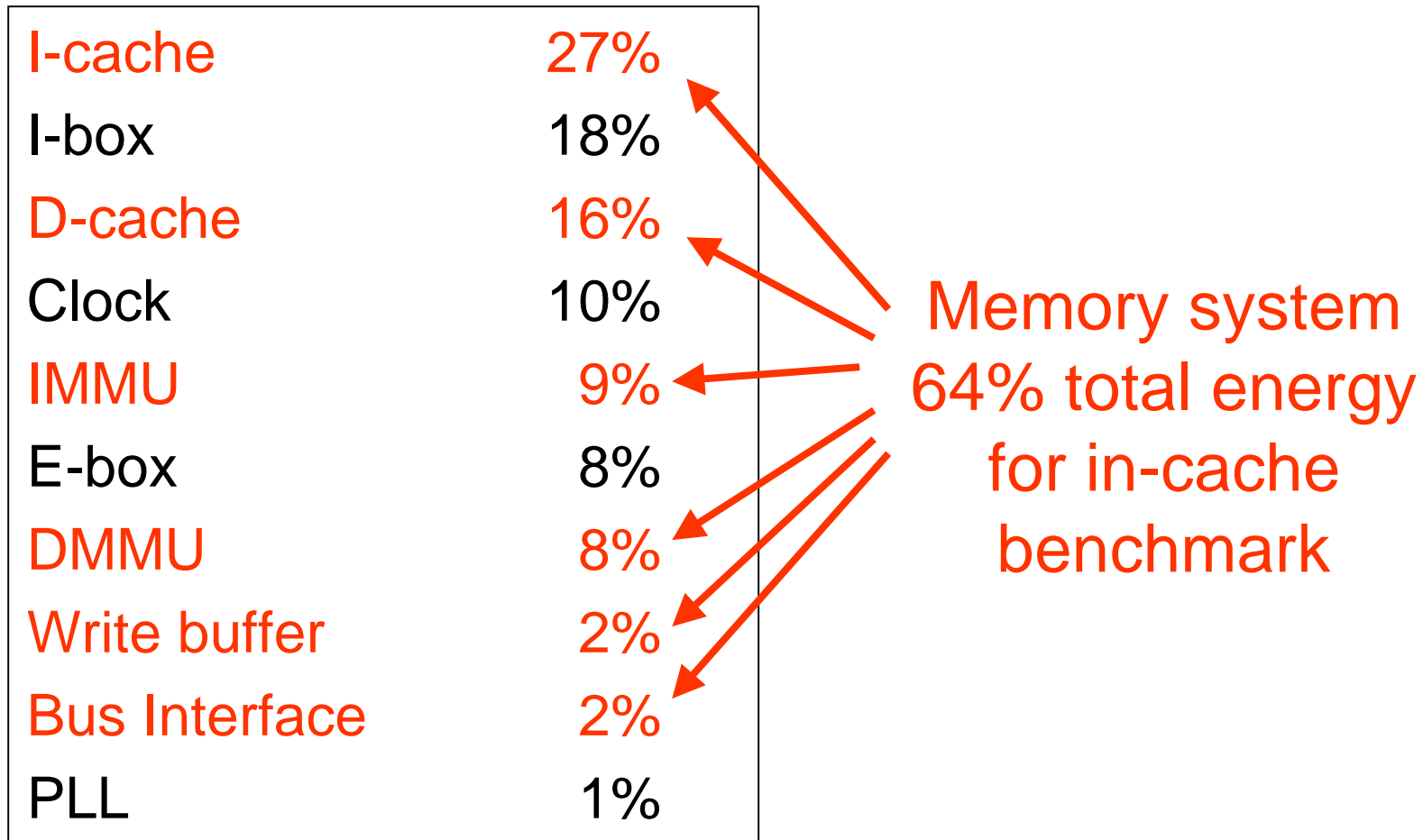


- 32 processing tiles
- Separate control/data networks
- 128x32b FLOP/cycle total
- 4096x8b OP/cycle total
- 128MB on-chip DRAM
- External DRAM interface
- Chip-chip interconnect channels
- 20x20mm² in 0.1μm CMOS





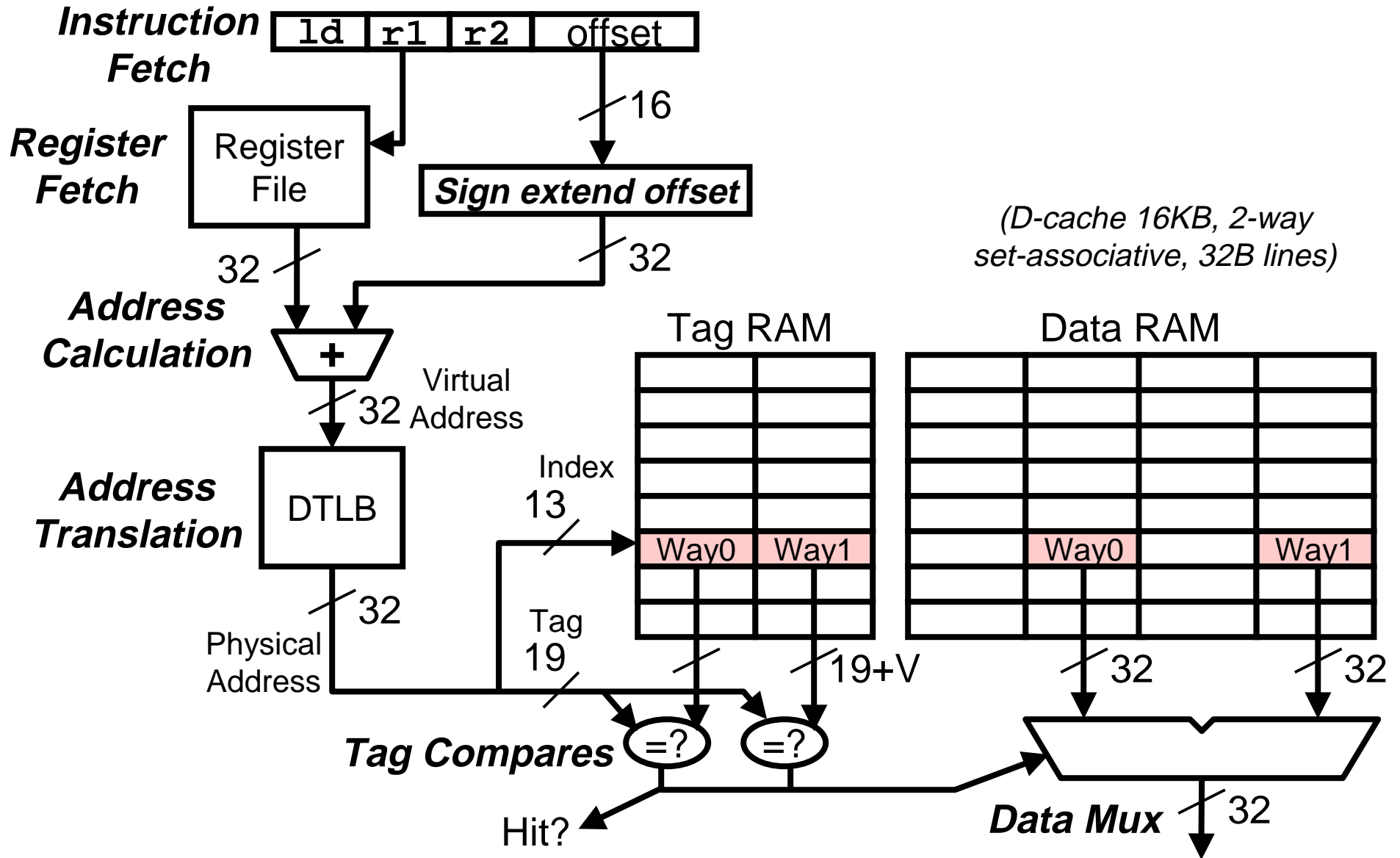
StrongARM Energy Breakdown



[Montanaro et al, ISSCC 1996]

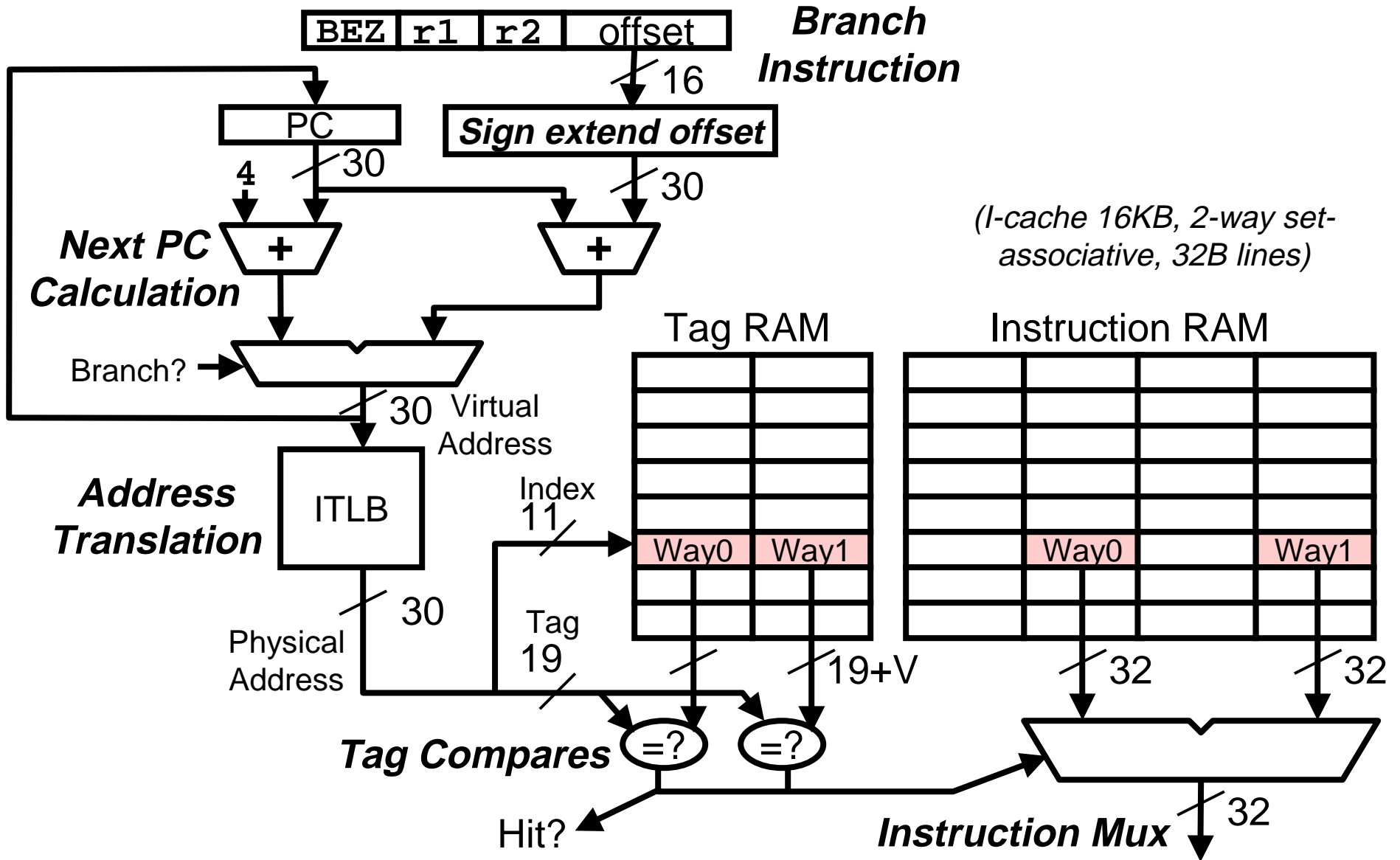


Data Memory Access Components





Instruction Fetch Components



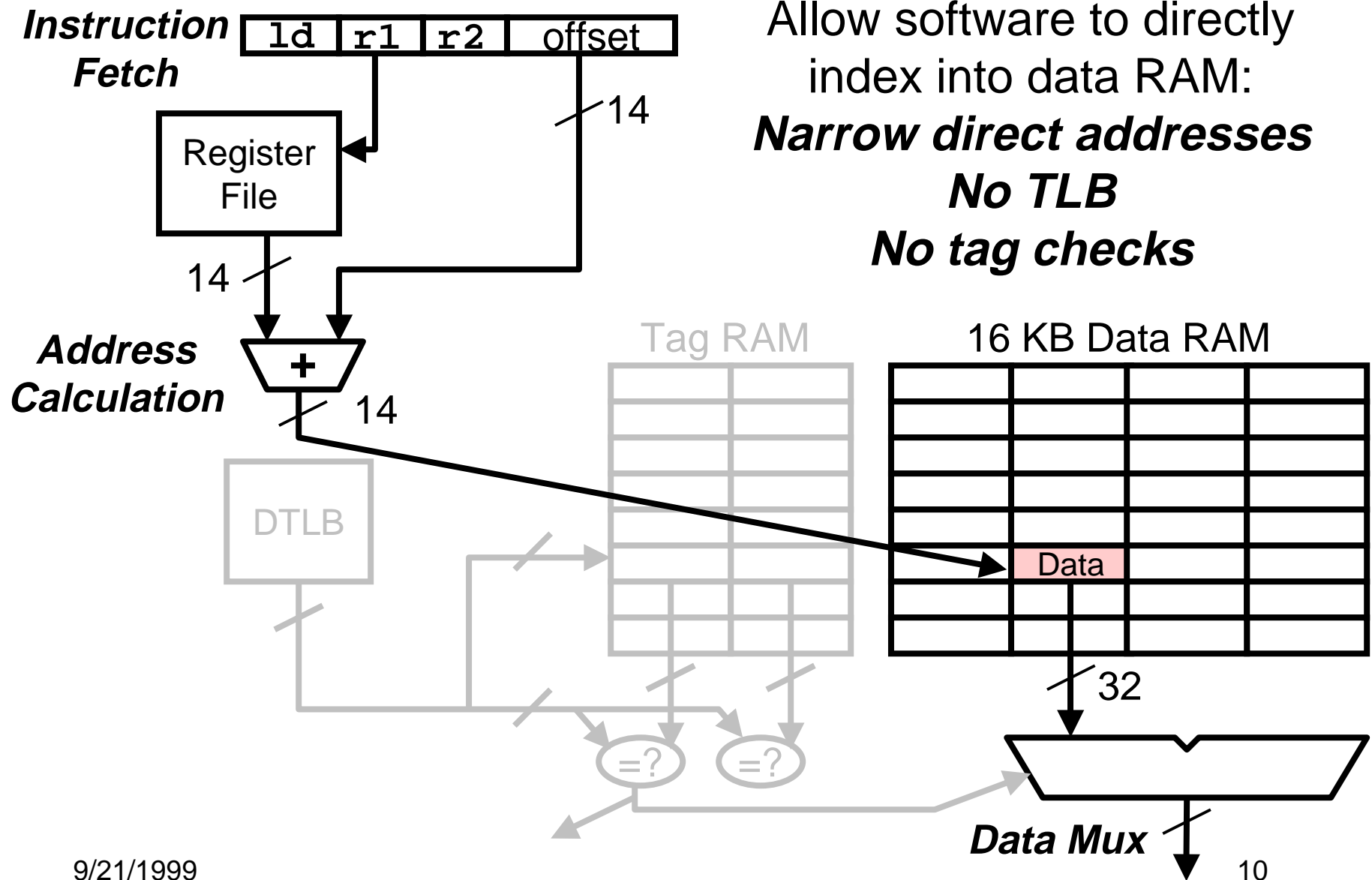


Reducing Memory Energy Overhead

- Most overhead is due to addressing:
 - wide virtual address generation
 - wide-virtual to wide-physical translation
 - cache tag compares on wide physical address



Direct-Addressed Caches





Example Software Interface: Tag-Unchecked Load Instructions

Allow software to use direct addressing when successive memory accesses are to same cache line

```
ld r1, (r2)
```

```
ld.nochk r3,4(r2)  $\Leftarrow$  Must be to same cache line
```

Energy reductions:

- no tag RAM read
- no tag compare
- only low order address bits need to be computed
- no TLB lookup

\Rightarrow ***Reduces cache access energy to just RAM read***



Direct-Addressed Caches

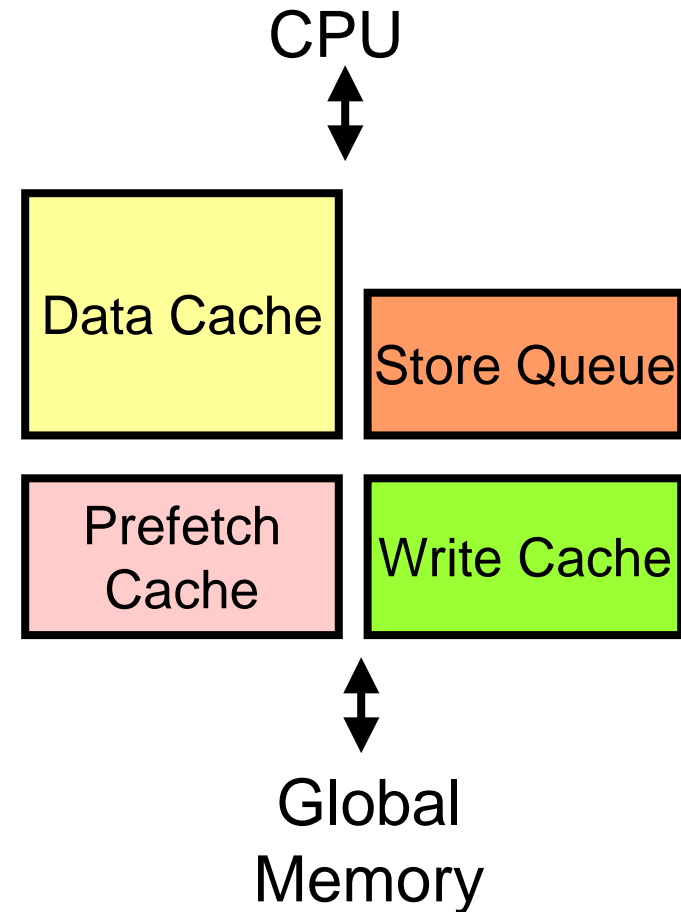
Advantages:

- ***lower energy***
 - can bypass TLB+tags when safe to do so
 - use narrow addresses to access storage
- ***backwards compatible***
 - fall back on TLB+tags if software unsure of cache state
- ***more flexible use of cache storage***
 - can provide arbitrary cache partitioning



UltraSPARC-III Memory System

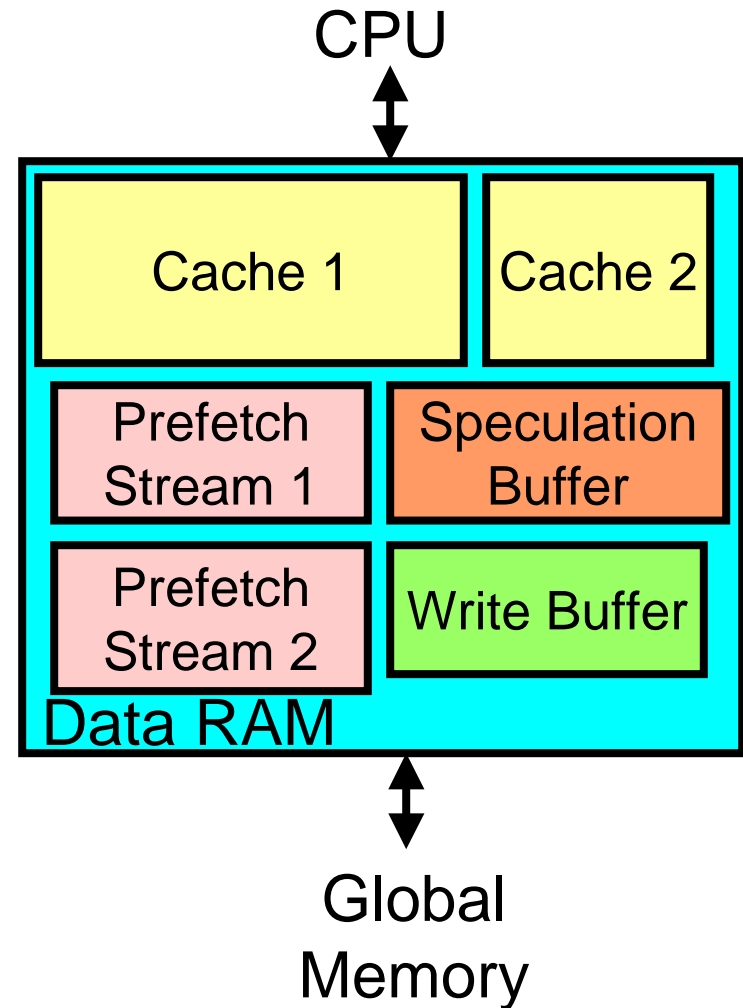
- Four special purpose processor-memory buffers
- Extensive associative circuitry to maintain coherent memory model





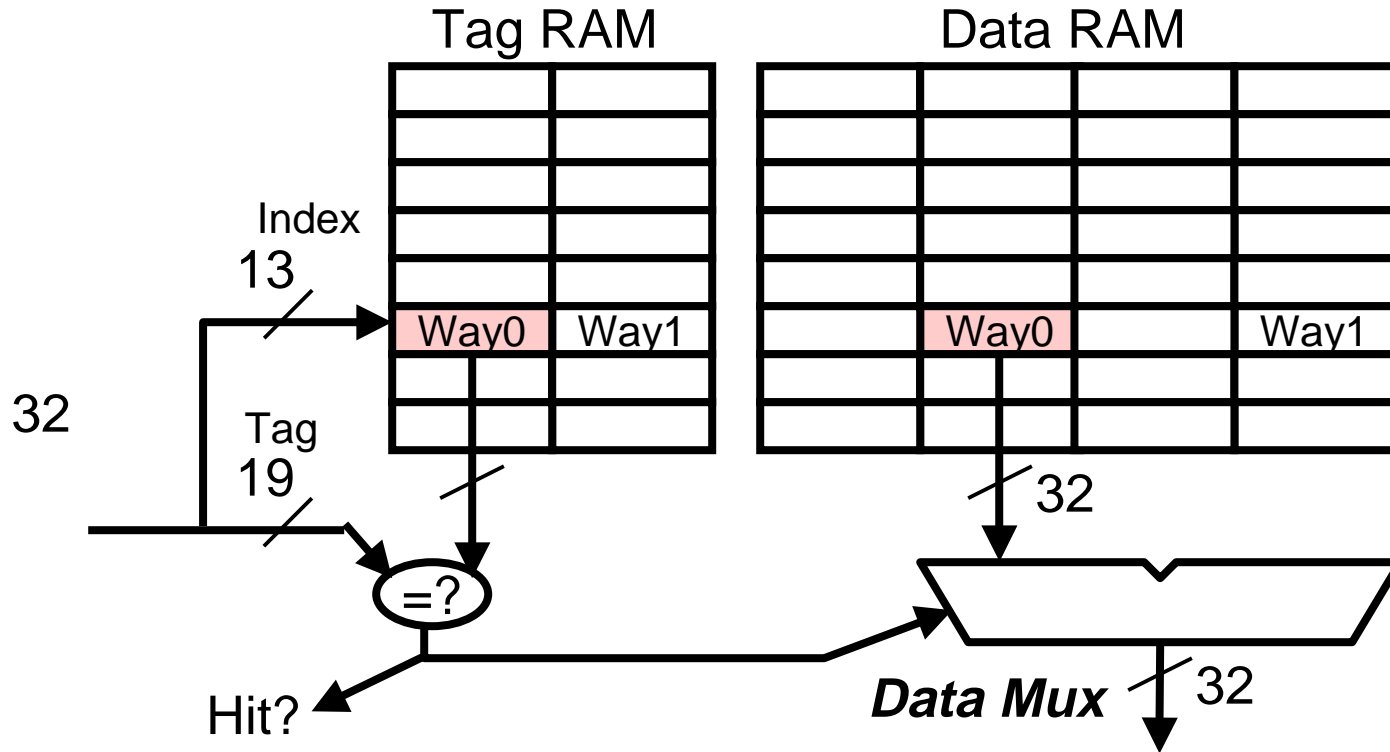
Direct-Address Cache Partitioning

- Flexible allocation of RAM to various forms of processor-memory buffer
- Hardware support for intra-cache coherence (tags or directory)
- Software can bypass coherence checks to reduce energy or to change memory model semantics





Possible Low-Power Column-Cache



- Check only the likely column first
- If miss, check other columns before going to memory



Direct-Address Cache Research Tasks

Research goal: Complete system-level design (silicon to API) of next-generation processor-memory interface

- Evaluate potential energy benefits of direct addressing
- Evaluate potential performance benefits of flexible cache partitioning
 - low-power version of malleable caches
- Evaluate alternative hardware/software mechanisms for managing intra-cache coherence
- Develop precise memory model as software interface to multiple active processor-memory buffers
 - application for CRF memory model formalism