Dresden Database Systems Group

Steps towards HW/SW-DB-CoDesign

Wolfgang Lehner

Modern Hardware – All over the place…

Dresden Database Systems Group

Transaction Logging Unleashed with NVRAM*

Tianzheng Wang Ryan Johnson
University of Toronto
{tzwang, ryan.johnson}@cs.toronto.edu

Utilization Wall: Dark Silicon's Effect on Multicore Scaling

Spectrum of tradeoffs between # of cores and frequency

Example:
65 nm → 32 nm (S = 2)

2x4 cores @ 1.8 GHz
(8 cores dark, 8 dim)

4 cores @ 1.8 GHz

65 nm

SGI Scales Up HANA On UV NUMA Syste

June 3, 2014 by Timothy Prickett Morgan

Data-Oriented Transaction Exe

Ippokratis Pandis[1,2] ipandis@ece.cmu.edu
Ryan Johnson[1,2] ryanjohn@ece.cmu.edu
Nikos Hardavell… nikos@northwester…

[1]Carnegie Mellon University Pittsburgh, PA, USA
[2]École Polytechnique Fédérale de Lausanne Lausanne, VD, Switzerland

ABSTRACT

While hardware technology has undergone major advancements over the past decade, transaction processing systems have remained largely unchanged. The number of cores on a chip grows exponentially, following Moore's Law, allowing for an ever-increasing number of transactions to execute in parallel. As the number of concurrently-executing transactions increases, contended critical sections become scalability burdens. In typical transaction processing systems the centri… often the first contended component and sc…

In this paper, we identify the c… transaction assignment policy as the prima… then, we design DORA, a system t… transaction to smaller actions and assigns … in which data each action is about to a… allows each thread to mostly access threa… minimizing interaction with the contention… manager. Built on top of a conventional … maintains all the ACID properties. Eva… implementation of DORA on a multicore s… DORA attains up to 4.8x higher throughp… storage engine when running a variety of … OLTP workloads.

Fast Updates on Read-Optimized Databases Using Multi-Core CPUs

Jens Krueger†, Changkyu Kim*, Martin Grund†, Nadathur Satish*, David Schwalb†, Jatin Chhugani*, Hasso Plattner†, Pradeep Dubey*, Alexander Zeier†

†Hasso-Plattner-Institute, Potsdam, Germany
Contact: jens.krueger@hpi.uni-potsdam.de

*Parallel Computing Lab, Intel Corporation
Contact: changkyu.kim@intel.com

ABSTRACT

Read-optimized columnar databases use differential updates to handle writes by maintaining a separate write-optimized delta partition which is periodically merged with the read-optimized and compressed main partition. This merge process introduces significant overheads and unacceptable downtimes in update intensive systems, aspiring to combine transactional and analytical workloads into one system.

In the first part of the paper, we report data analyses of 12 SAP Business Suite customer systems. In the second half, we present an optimized merge process reducing the merge overhead of current systems by a factor of 30. Our linear-time merge algorithm exploits the underlying high compute and bandwidth resources of modern multi-core CPUs with architecture-aware optimizations and efficient parallelization. This enables compressed in-memory column stores to handle the transactional update rate required by enterprise applications, while keeping properties of read-optimized databases for analytic-style queries.

partition. Inserting into the write-optimized structure can be … formed fast if the size of the structure is kept small enough. As … additional benefit, this also ensures that the read performance d… not degrade significantly. However, keeping this size small imp… merging frequently, which increases the overhead of updates.

To the best of our knowledge we are not aware of any sophi… cated implementation and therefore compare against a naïve imp… mentation. Based on the result of analyzing 12 SAP Business S… customer systems, we found that current systems would merge … prox. 20 hours every month, while supporting a maximum of ~1… updates per second (see Section 2 for more detail). In read-mos… scenarios this limitation is not a major problem since the wo… load can be stopped during reload, modifications are invisible u… applied in batch or performance degradation is acceptable. H… ever, when engineering a system for both transactional and ana… ical workloads as described in [22, 17, 13], it becomes essen… to reduce the merge overhead and to support the required sin… update rates for handling transactional workloads. Systems un… …o cope with even longer times for merging or f…

Staring into the Abyss: An Evaluation of Concurrency Control with One Thousand Cores

Xiangyao Yu
MIT CSAIL
yxy@csail.mit.edu

George Bezerra
MIT CSAIL
gbezerra@csail.mit.edu

Andrew Pavlo
Carnegie Mellon University
pavlo@cs.cmu.edu

Srinivas Devadas
MIT CSAIL
devadas@csail.mit.edu

Michael Stonebraker
MIT CSAIL
stonebraker@csail.mit.edu

ABSTRACT

Computer architectures are moving towards an era dominated by many-core machines with dozens or even hundreds of cores on a single chip. This unprecedented level of on-chip parallelism introduces a new dimension to scalability that current database management systems (DBMSs) were not designed for. In particular, as the number of cores increases, the problem of concurrency control becomes extremely challenging. With hundreds of threads running in parallel, the complexity of coordinating competing accesses to data will likely diminish the gains from increased core counts.

To better understand just how unprepared current DBMSs are for future CPU architectures, we performed an evaluation of concurrency control for on-line transaction processing (OLTP) workloads on many-core chips. We implemented seven concurrency control algorithms on a main-memory DBMS and using computer simula-

that instruction-level parallelism and single-threaded performance will give way to massive thread-level parallelism.

As Moore's law continues, the number of cores on a single chip is expected to keep growing exponentially. Soon we will have hundreds or perhaps a thousand cores on a single chip. The scalability of single-node, shared-memory DBMSs is even more important in the many-core era. But if the current DBMS technology does not adapt to this reality, all this computational power will be wasted on bottlenecks, and the extra cores will be rendered useless.

In this paper, we take a peek at this dire future and examine what happens with transaction processing at one thousand cores. Rather than looking at all possible scalability challenges, we limit our scope to concurrency control. With hundreds of threads running in parallel, the complexity of coordinating competing accesses to data will become a major bottleneck to scalability, and will likely dwin-

LLAMA: A Cache/Storage Subsystem for Modern Hardware

Justin Levandoski
Microsoft Research
One Microsoft Way
Redmond, WA 98052
justinle@microsoft.com

David Lomet
Microsoft Research
One Microsoft Way
Redmond, WA 98052
lomet@microsoft.com

Sudipta Sengupta
Microsoft Research
One Microsoft Way
Redmond, WA 98052
sudipta@microsoft.com

ABSTRACT

LLAMA is a subsystem designed for new hardware environments that supports an API for page-oriented access methods, providing both cache and storage management. Caching (CL) and storage (SL) layers use a common mapping table that separates a page's logical and physical location. CL supports data updates and management updates (e.g., for index re-organization) via latch-free compare-and-swap atomic state changes on its mapping table. SL uses the same mapping table to cope with page location changes produced by log structuring on every page flush. To demonstrate LLAMA's suitability, we tailored our latch-free Bw-tree implementation to use LLAMA. The Bw-tree is a B-tree style index. Layered on LLAMA, it has higher performance and scalability using real workloads compared with BerkeleyDB's B-tree, which is known for good performance.

1 INTRODUCTION

1.1 Modern Architectures

Modern computer platforms have changed sufficiently that it is

We believe there are fundamental problems posed by current hardware that impact all access methods: B-trees, hashing, multi-attribute, temporal, etc. Further, these problems can be solved with general mechanisms applicable to most access methods.

1. Good processor utilization and scaling with multi-core processors via latch-free techniques.
2. Good performance with multi-level cache based memory systems via delta updating that reduces cache invalidations.
3. Write limited storage in two senses: (1) limited performance of random writes; (2) flash write limits; via log structuring.

The Bw-tree [16], an index resembling B-trees [4, 7], is an example of a DC or key-value store that exploits these techniques. Indeed, it is an instance of a paradigm for how to achieve latch-freedom and log structuring more generally. In this paper, we describe a new architecture where the latch-free and log-structure techniques of the Bw-tree are implemented in a cache/storage subsystem capable of supporting multiple access methods, in the same way that a traditional cache/storage subsystem deals with latched access to fixed size pages that are written back to disks as in-place updates.

A Look at Hardware Trends – 201x

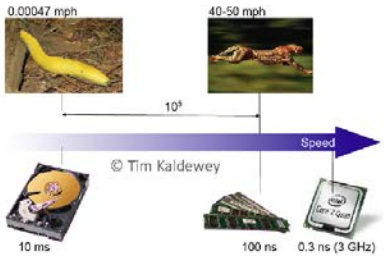
Dresden Database Systems Group

> A Look at Hardware Trends

Database Technology Group

Increasing Main Memory Capacity*

„Main Memory“ is the new disk!(?)



Increasing Number of Cores

- CPU/GPU, hybrids
- FPGA (**Field Programmable Gate Array**)

„Parallelism“ is the name of the game!

*stable RAM will be an additional game changer

TECHNISCHE UNIVERSITÄT DRESDEN

11

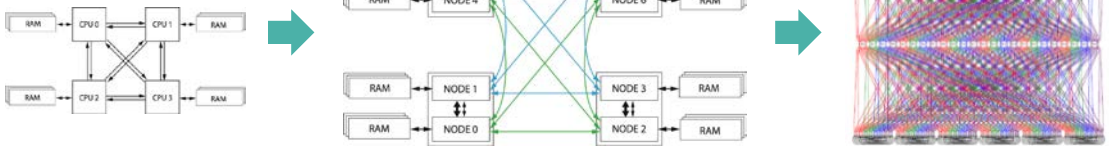
A Look at Hardware Trends - 2015

Dresden Database Systems Group

System Level

Component Level

Extreme NUMA Effects



2

„Main Memory“ is the new disk!(?)

„Parallelism“ is the name of the game!

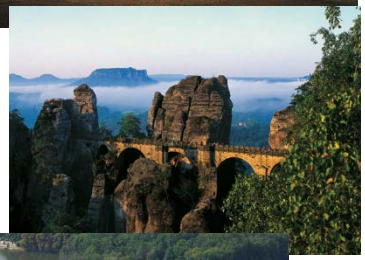
Storage-Class Memory

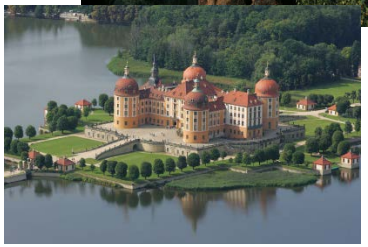
Application-Specific Instruction Sets

1

The Dresden Agenda …

Dresden Database Systems Group





Culture

- (inoffical) cultural capital of Germany (theaters, museums, etc.)

Semi Conductor Industry

- No.1 semiconductor site in Europe
- No.5 semiconductor site on the planet
  - Siltronic wafer production, Toppan / Dupont Photomasks
  - Infineon Global Foundries
  - ZMD, ATMEL, Applied Materials
  - Intel, Amazon etc.

Large top-notch R&D organizations

- TU Dresden with >38.000 students
- Fraunhofer with 11 institutes (1.200 employees)
- Max Planck with 3 institutes (900 employees)
- Leibnitz Gesellschaft with 4 institutes and 1.500 employees
- Helmholtz Institute, Rossendorf (800 employees)

amazon

SAP

IBM

intel

infineon

T

GLOBALFOUNDRIES

Systems Multimedia Solutions

The “Dresden Data(base) System Group”

Dresden Database Systems Group

5G Lab Germany

cfaed CENTER FOR ADVANCING ELECTRONICS DRESDEN

ScaDS DRESDEN LEIPZIG

ROSI

HAEC CRC 912 Highly Adaptive Energy-Efficient Computing

IT4BI

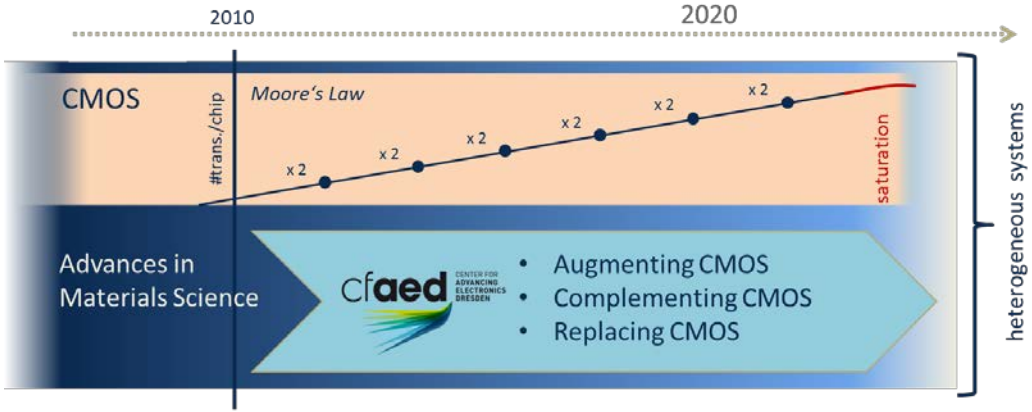
Federal Ministry of Education and Research

DFG Deutsche Forschungsgemeinschaft

SAP HANA Student Campus

The cfAED mission

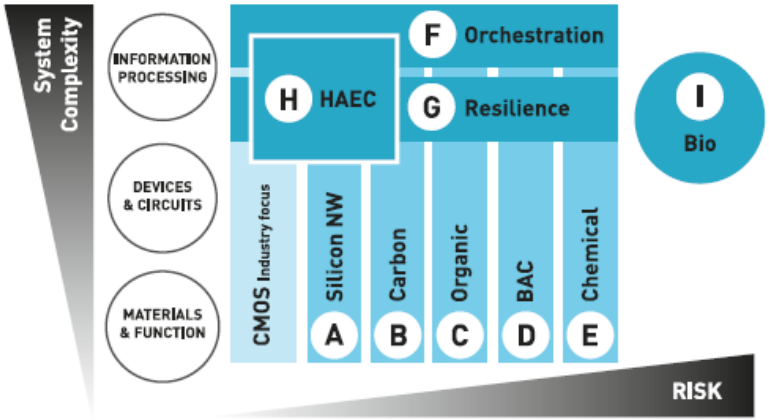
Dresden Database Systems Group



cfaed CENTER FOR ADVANCING ELECTRONICS DRESDEN

Unique Window of Opportunity
for shaping the next big technology waves

…more shots on a goal



SOME NUMBERS

- FUNDING VOLUME // € 34 million
- FUNDING PERIOD // 1 Nov 2012 – 31 Oct 2017
- PARTICIPATING INSTITUTIONS // 11
- INVESTIGATORS // ~ 60

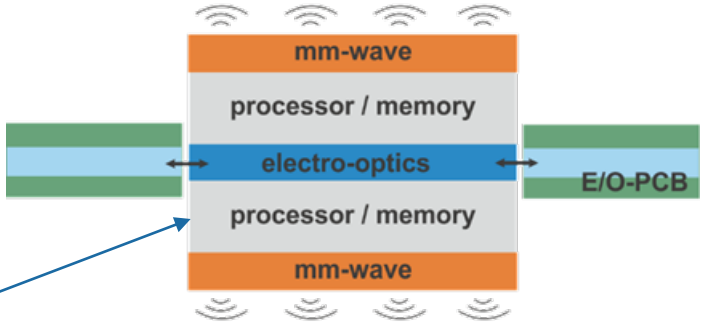
The „HAEC-Box“

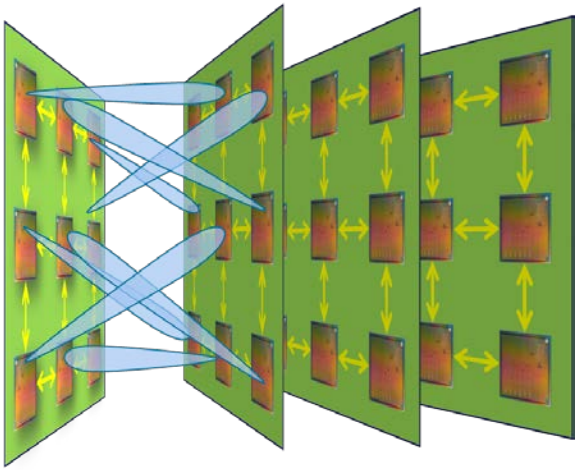
HAEC
CRC 912
Highly Adaptive Energy-Efficient Computing

Dresden Database Systems Group

Key Characteristics

- Optical communication on board
- Adaptive wireless backplane communication
- 3D stacking
- Self-* capabilities on SW-level

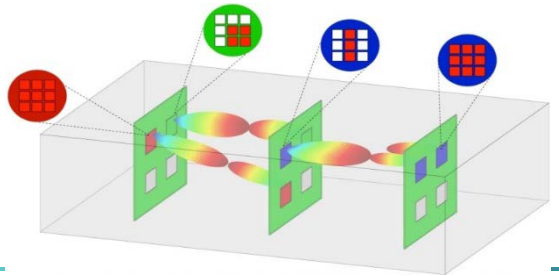




Multiple layers of memory and processing units

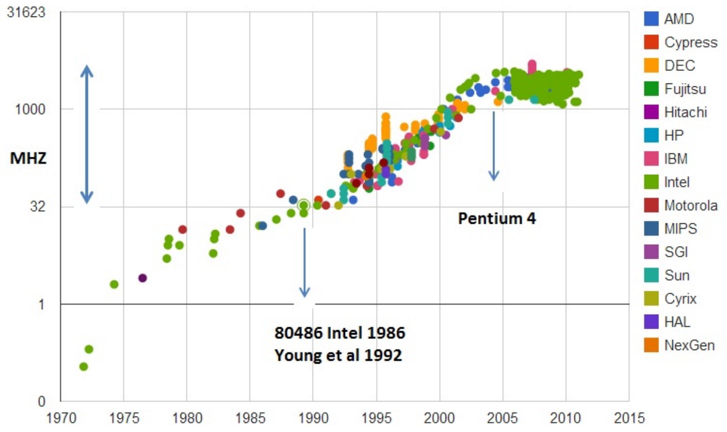
- Impact on processor design
- React on computer architecture

1. Instruction set extensions
2. DB architecture for Scale-up

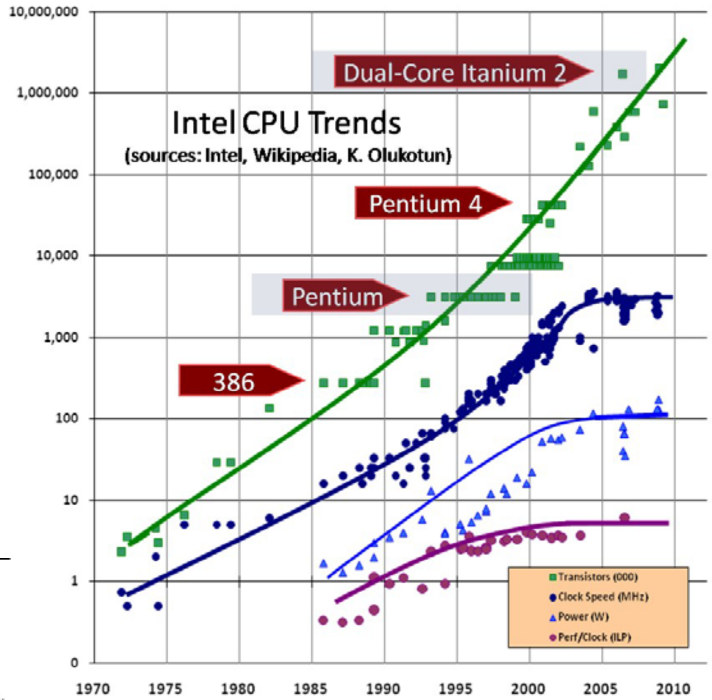


Database-specific Instruction Sets

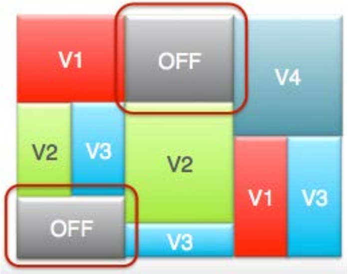
xPU Developments and Consequences



http://upload.wikimedia.org/wikipedia/en/c/ce/Clock_CPU_Scaling.jpg



http://www.extremetech.com/wp-content/uploads/2012/02/CPU-Scaling.jpg



Appears in the Proceedings of the 38th International Symposium on Computer Architecture (ISCA '11)

Dark Silicon and the End of Multicore Scaling

Hadi Esmaeilzadeh† Emily Blem‡ Renée St. Amant§ Karthikeyan Sankaralingam‡ Doug Burger◇
†University of Washington ‡University of Wisconsin-Madison
§The University of Texas at Austin ◇Microsoft Research
hadianeh@cs.washington.edu blem@cs.wisc.edu stamant@cs.utexas.edu karu@cs.wisc.edu dburger@microsoft.com

ABSTRACT

Since 2005, processor designers have increased core counts to exploit Moore's Law scaling, rather than focusing on single-core per-

ture, and compiler advances, Moore's Law, coupled with Dennard scaling [11], has resulted in commensurate exponential performance increases. The recent shift to multicore designs has aimed to in-

Motivation of „DB Processor“

Dresden Database Systems Group

Today's Database Servers

- Fat cores (area & power)
- Few HW adaptions
- CMOS scaling

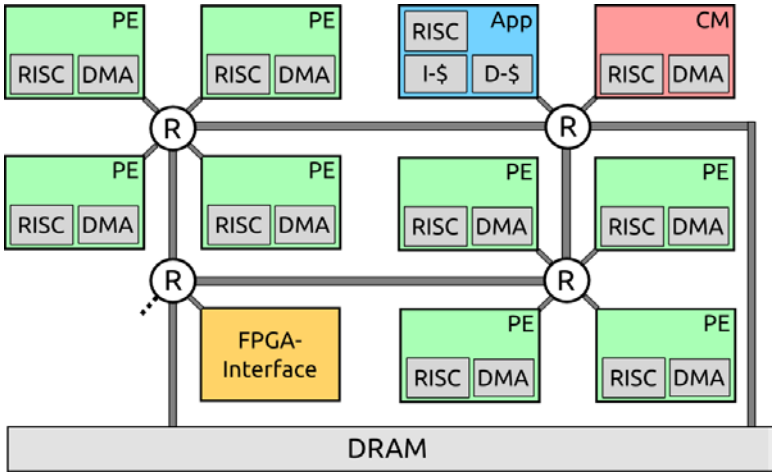
Database Machines (Ancient)

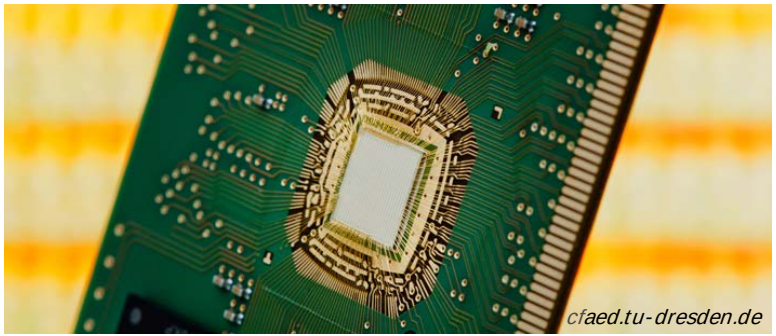
- Processors build from scratch
- Long development cycles
- High development costs

Our Approach

- HW/SW Co-design
- Customizable processor
- Application-specific ISA extensions
- Tool flow & short HW development cycles

Currently available HW: The Tomahawk2





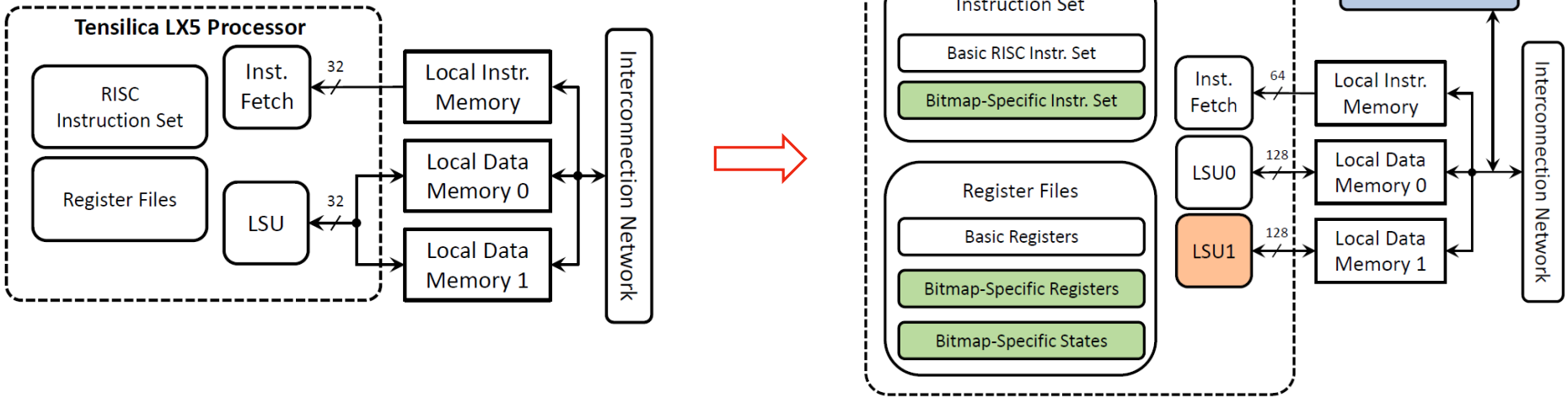
Core manager (CM):
- Extended Xtensa-LX4
- Scheduling specific instruction set
- 32KB for code
- 64KB for data

Processing Elements (PEs)
- Xtensa-LX4 from Tensilica (now Cadence)
- 32KB for code
- 32KB for data

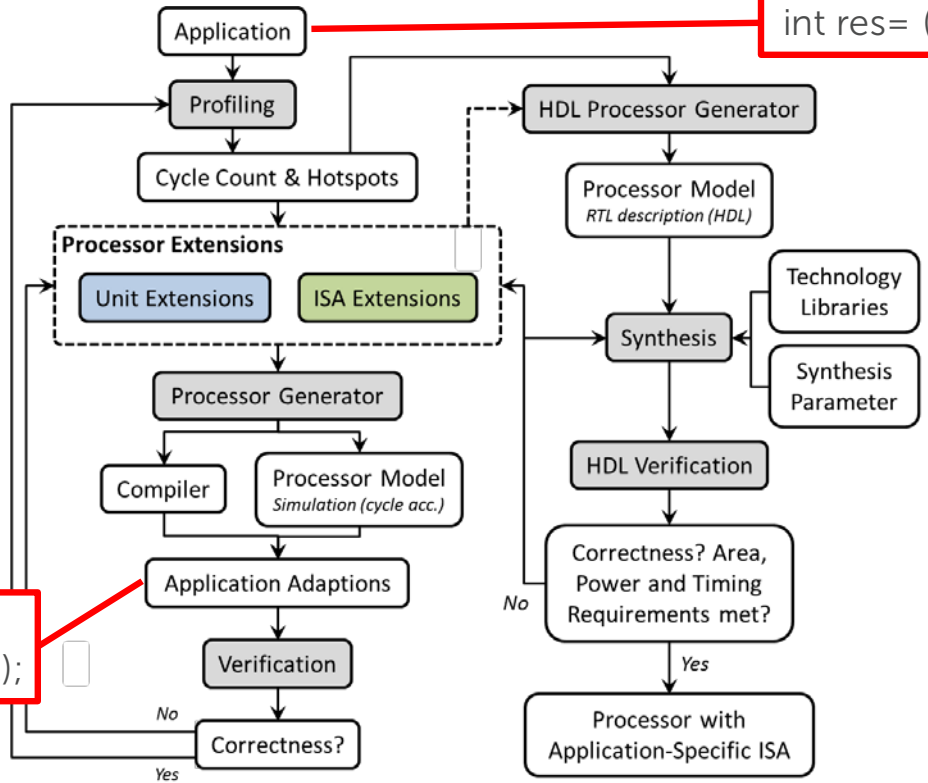
Application Core (App)
- 570T core from Tensilica (now Cadence)
- 16KB cache for code
- 16KB cache for data

2 x 128MB DRAM

Customizable Processor Model



Overview: Tool Flow




```
int res= (v0 + v1 + v2) >> shift8;
```

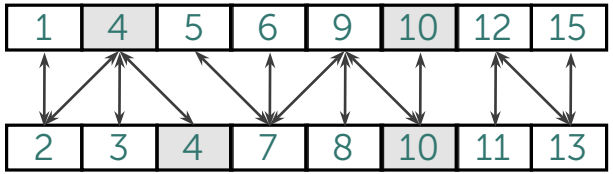
```
// shift8 -> internal state
int res=add3_shift(v0, v1, v2);
```

A selection of database primitives…

Dresden Database Systems Group

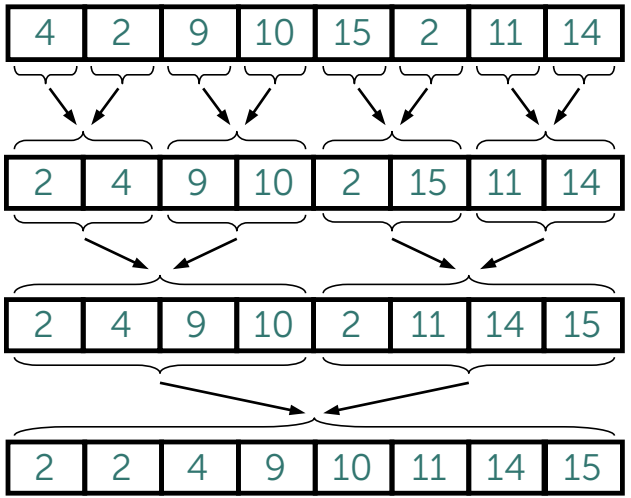
Set Operations (for RID lists)

- Intersection
- Difference
- Union



Sorting

- Merge Sort



Hash Operations

- Integer Hashing
- String Hashing
- Hash Table Management

Compression (for Bitmap indexes)

- Word-Aligned Hybrid (WAH)
- Position List Word Aligned Hybrid (PLWAH)
- COMPressed Adaptive indeX (COMPAX)

Sorted-Set Intersection

Dresden Database Systems Group

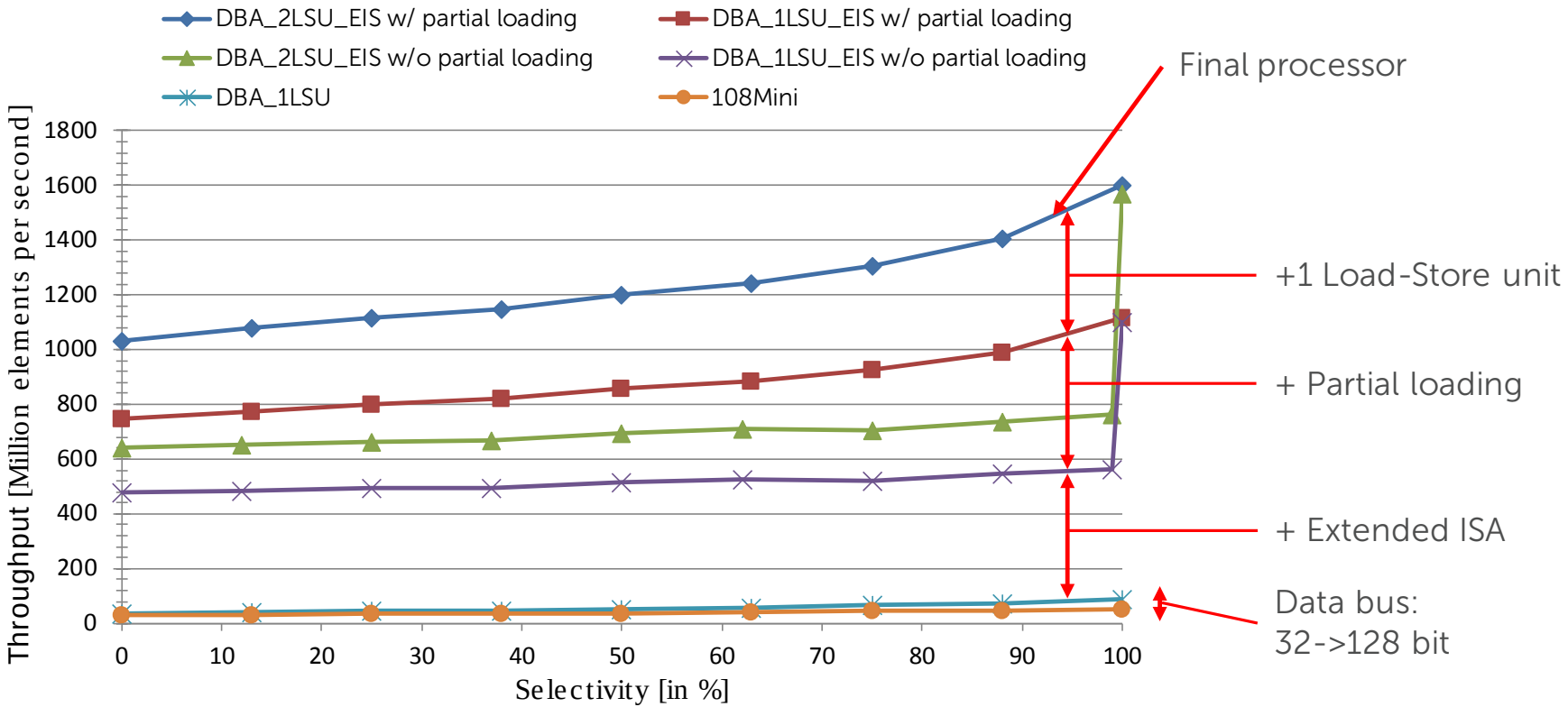
```
int intersect(int* A, int* B, int l_a, int l_b, int* C) {
        int pos_a = 0, pos_b = 0, pos_c = 0;
        while( pos_a < l_a && pos_b < l_b ) {
                if( A[ pos_a ] == B[ pos_b ] ) {
                        C[ pos_c++ ] = A[ pos_a ];
                        pos_a++;
                        pos_b++;
                }
                else if( A[ pos_a ] < B[ pos_b ] )
                        pos_a++;
                else
                        pos_b++;
        }
        return pos_c;
}
```

internal states

merged in one inst.

➔ + 2x 128 bit data busses + explicit load-store + SIMD + …

Selectivity: Intersection



Throughput [Mib/s]

Dresden Database Systems Group

Selectivity: 50%

+1 Load-Store unit

Processor	Partial Load–Store	f[MHz]	Intersection	Union	Difference	Merge–Sort
108Mini[1]	-	442	31.3	26.4	35.7	1.7
DBA_1LSU	-	435	50.7	47.7	50.4	3.2
DBA_1LSU_EIS	no	424	513.4	665.0	658.8	29.3
DBA_2LSU_EIS	no	410	693.0	643.0	637.0	28.3
DBA_1LSU_EIS	yes	424	859.0	574.2	859.0	29.3
DBA_2LSU_EIS	yes	410	1203.0	780.4	1192.6	28.3

Data bus: 32->128 bit

Final processor

➔ DBA_2LSU_EIS *vs.* 108MINI: *38x* *30x* *33x* *17x*

Comparison

Dresden Database Systems Group

SORTED-SET INTERSECTION

	INTEL I7-920	DBA_2LSU_EIS	
Throughput (elements/s)	1,100 mio	1,203 mio	
Clock frequency	2.67 GHz	0.41 GHz	→ 7x improvement
Max. TDP	130 W	0.135 W	→ 963x improvement
Cores/Threads	4/8	1/1	
Feature size	45 nm	65 nm	
Area (logic & memory)	263 mm^2	1.5 mm^2	→ 175x improvement

Timing and Area

Dresden Database Systems Group

Technology	Processor	$A_{LOGIC}\,[mm^2]$	$A_{MEM}\,[mm^2]$	f_{MAX} [MHz]	P[mW] @f_{MAX}
65 nm	108Mini	0.220[1]	-	442[1]	27.4[1]
	DBA_1LSU	0.177	0.874	435	56.6
	DBA_2LSU	0.177	0.870	429	57.1
	DBA_1LSU_EIS	0.523	0.874	424	123.5
	DBA_2LSU_EIS	0.645	0.870	410	135.1
28 nm	DBA_2LSU_EIS	0.169	0.232	500	47.0

[1] http://www.tensilica.com/uploads/pdf/108Mini.pdf

Final processor

Relative Area Consumption(DBA_2LSU_EIS)

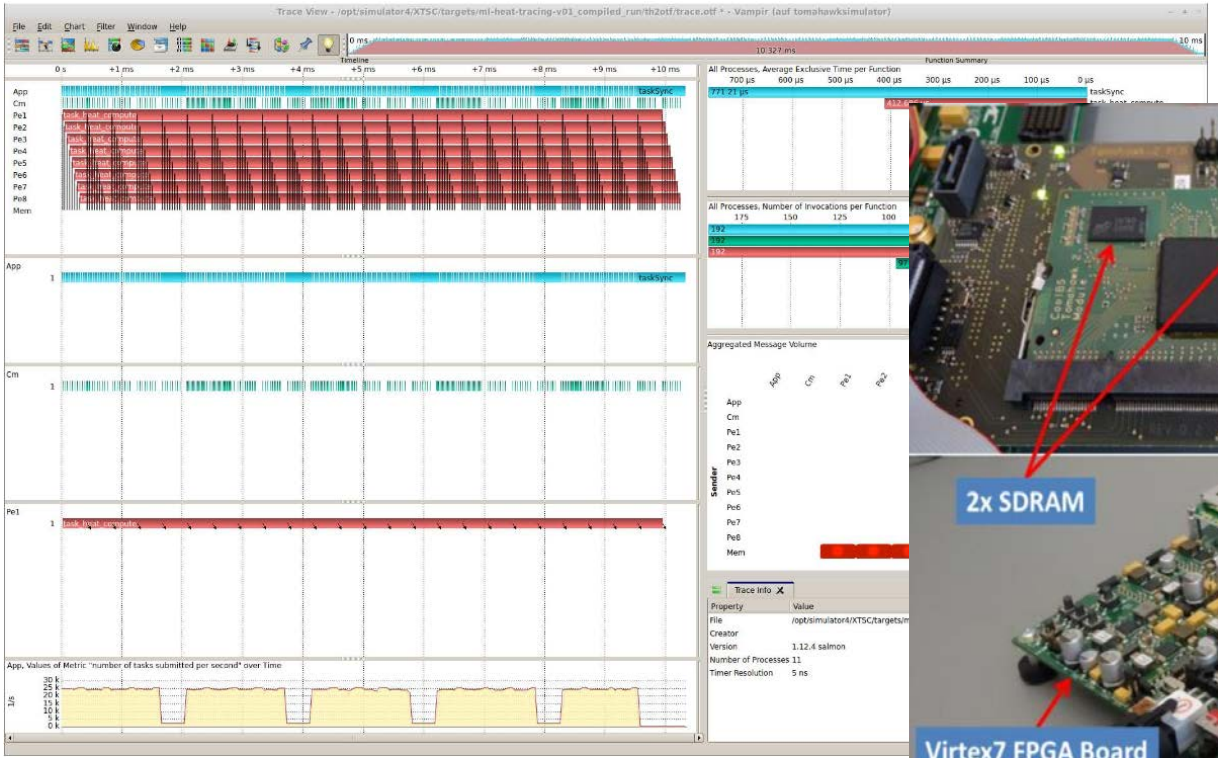
Part	Area[%]
Basic Core	20.5
Decoding/Muxing	14.4
States	14.7
Op: All	11.3
Op: Intersection	6.8
Op: Difference	9.0
Op: Union	17.6
Op: Merge-Sort	5.7
SUM	100

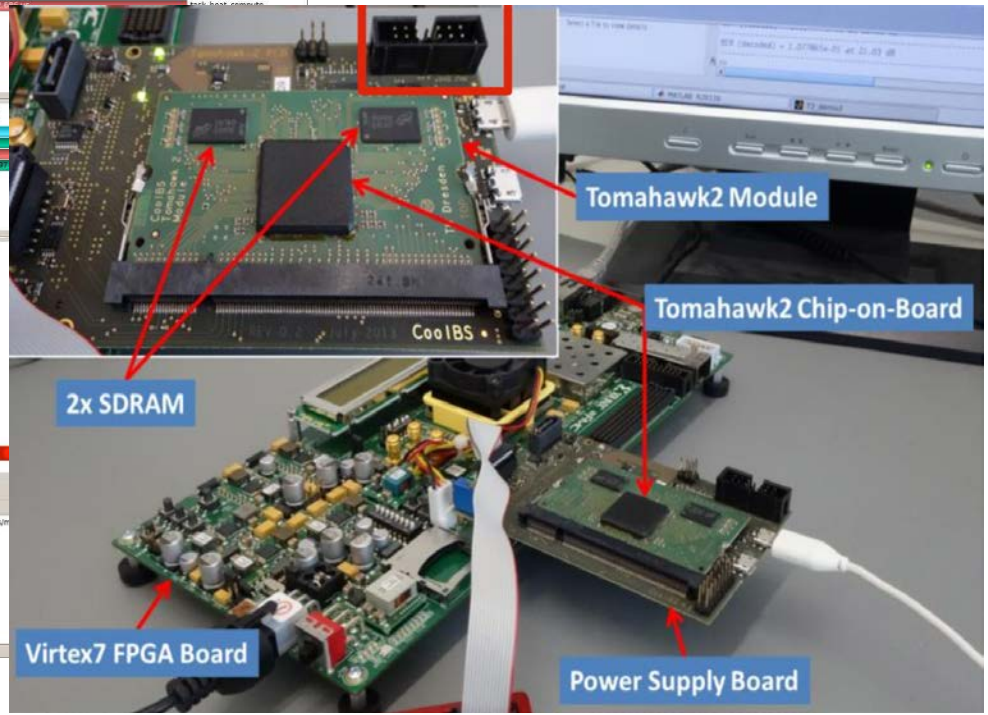
Tomahawk2 Programming

Dresden Database Systems Group

Profiling Software

Hardware Setup

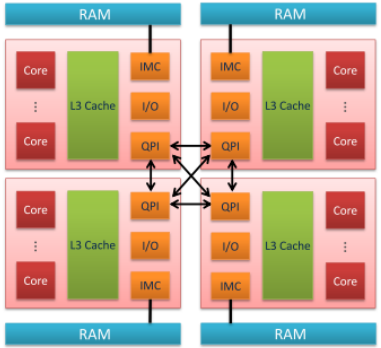




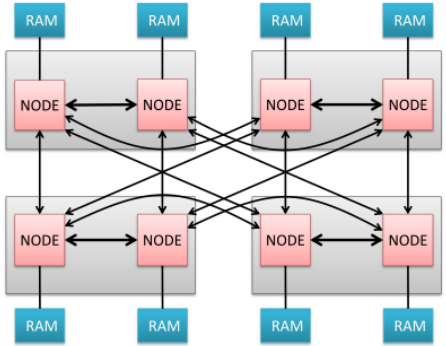
DB-Principles for „Mega-Core“ Machines

Main Driver: NUMA Awareness

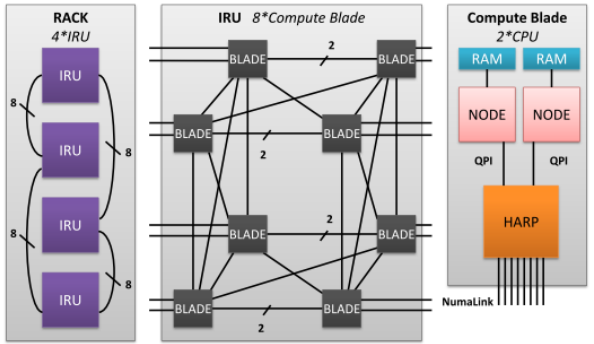
Dresden Database Systems Group



(a) Intel Machine (Detailed).



(b) AMD Machine (Topology View).

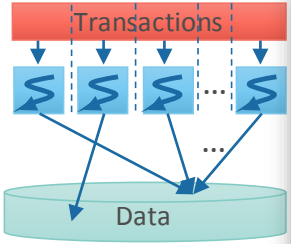
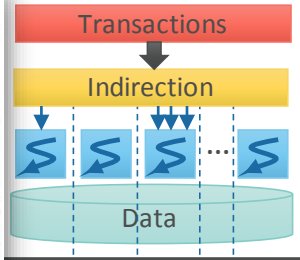


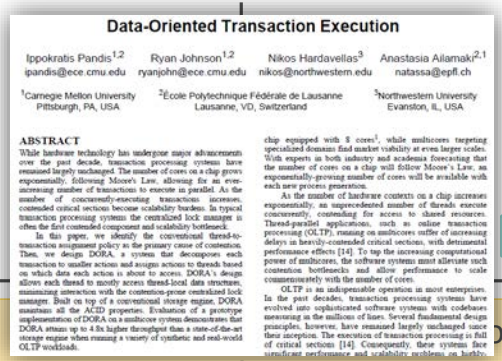
(c) SGI Machine (Topology View).

Intel machine			**AMD machine**			**SGI machine**		
distance	bandwidth (GB/s)	latency (ns)	distance (link width)	bandwidth (GB/s)	latency (ns)	distance	bandwidth (GB/s)	latency (ns)
local	26.7	129	local	16.4	85	local	36.2	81
1 hop QPI	10.7	193	1 hop HT (full link)	5.8	136	2nd processor	9.5	400
			1 hop HT (split,single)	4.2	152	1 hop NUMALink	7.5	505 - 515
			1 hop HT (split,dual)	2.9	152	2 hop NUMALink	7.5	625 - 635
			2 hop HT (split,single)	3.7	196	3 hop NUMALink	7.1	745 - 755
			2 hop HT (split,dual)	1.8	196	4 hop NUMALink	6.5	870

TA versus Data-Oriented Architecture (DORA)

Dresden Database Systems Group

	Transaction-Oriented Architecture *shared-everything*	Data-Oriented Architecture *mixed shared-everything & shared-nothing*
Which Architecture ?		
Pros & Cons	− Lack of scalability + **No** load balancing & indirection required + Energy proportional by design	− …n massive parallel systems − Load Balancing and indirection required − **Not** energy proportional by design
Challenges	Well investigated and widely deployed	*(1)* *Speed up* load balancing indirection to work efficiently for ***in-memory*** systems (2) How to make the data-oriented architecture **energy proportional**



Data-Oriented Transaction Execution

Ippokratis Pandis[1,2] ipandis@ece.cmu.edu
Ryan Johnson[1,2] ryanjohn@ece.cmu.edu
Nikos Hardavellas[3] nikos@northwestern.edu
Anastasia Ailamaki[2,1] natassa@epfl.ch

[1]Carnegie Mellon University Pittsburgh, PA, USA
[2]École Polytechnique Fédérale de Lausanne Lausanne, VD, Switzerland
[3]Northwestern University Evanston, IL, USA

ABSTRACT

While hardware technology has undergone major advancements over the past decade, transaction processing systems have remained largely unchanged. The number of cores on a chip grows exponentially, following Moore's Law, allowing for an ever-increasing number of transactions to execute in parallel. As the number of concurrently-executing transactions increases, contended critical sections become scalability burdens. In typical transaction processing systems the centralized lock manager is often the first contended component and scalability bottleneck.

In this paper, we identify the conventional thread-to-transaction assignment policy as the primary cause of contention. Then, we design DORA, a system that decomposes each transaction to smaller actions and assigns actions to threads based on which data each action is about to access. DORA's design allows each thread to mostly access thread-local data structures, minimizing interaction with the contention-prone centralized lock manager. Built on top of a conventional storage engine, DORA maintains all the ACID properties. Evaluation of a prototype implementation of DORA on a multicore system demonstrates that DORA attains up to 4.8x higher throughput than a state-of-the-art storage engine when running a variety of synthetic and real-world OLTP workloads.

chip equipped with 8 cores[1], while multicores targeting specialized domains find market viability at even larger scales. With experts in both industry and academia forecasting that the number of cores on a chip will follow Moore's Law, an exponentially-growing number of cores will be available with each new process generation.

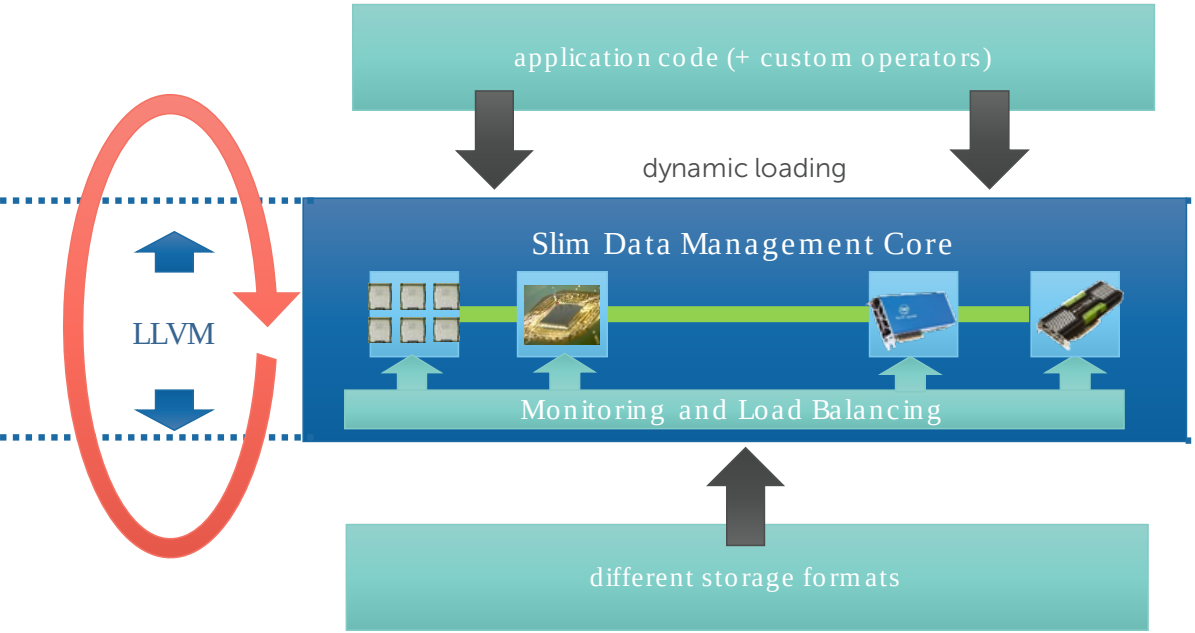
As the number of hardware contexts on a chip increases exponentially, an unprecedented number of threads execute concurrently, contending for access to shared resources. Thread-parallel applications, such as online transaction processing (OLTP), running on multicores suffer of increasing delays in heavily-contended critical sections, with detrimental performance effects [14]. To tap the increasing computational power of multicores, the software systems must alleviate such contention bottlenecks and allow performance to scale commensurately with the number of cores.

OLTP is an indispensable operation in most enterprises. In the past decades, transaction processing systems have evolved into sophisticated software systems with codebases measuring in the millions of lines. Several fundamental design principles, however, have remained largely unchanged since their inception. The execution of transaction processing is full of critical sections [14]. Consequently, these systems face

ERIS Data Management Core

Dresden Database Systems Group

... an academic playground for modern DB techniques



CHARACTERISTICS

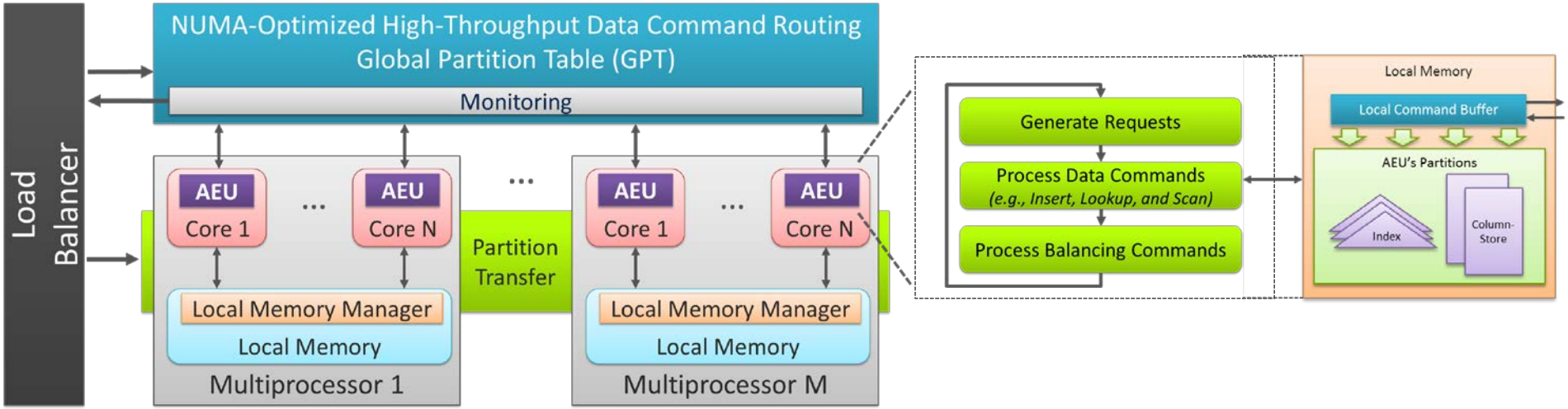
- data-oriented architecture (via message passing)
- NUMA awareness
- heterogeneous hardware
- aggressive elasticity strategies
- dynamic data placement policies

ERIS Overall Architecture

Dresden Database Systems Group

Data-oriented Architecture

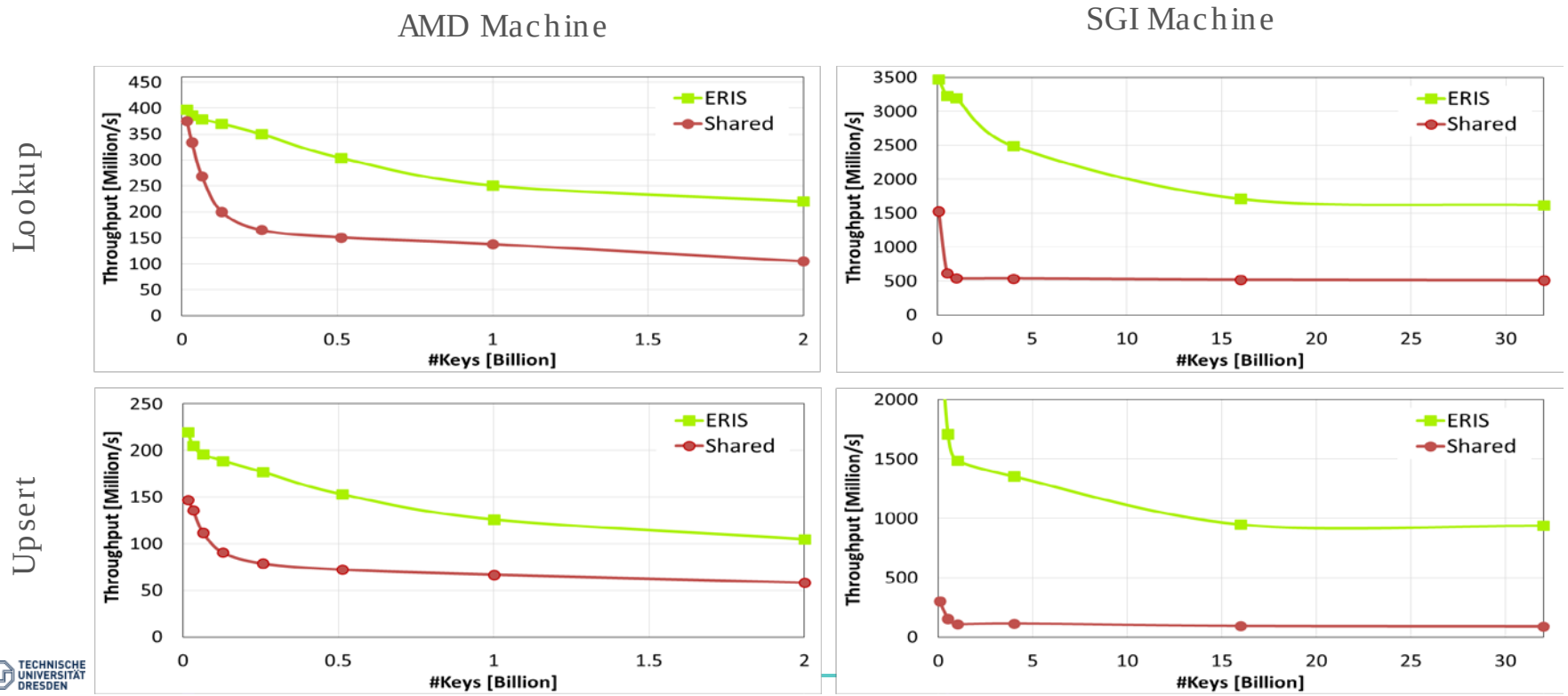
- Follows MVCC principle
- Distribution based on logical partitioning
- Aggressive re-partioning using copy as well as link strategies



Evaluation: Some MicroBenchmarking

Dresden Database Systems Group

Lookup/Upsert Throughput Depending on Index Size

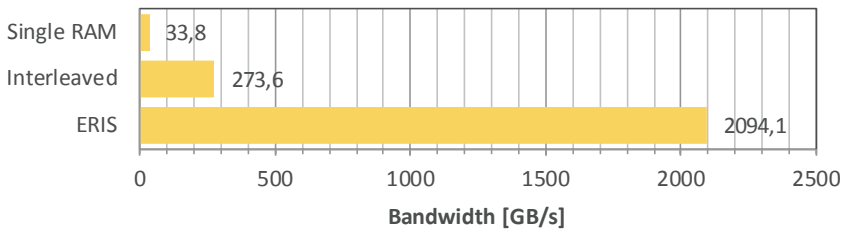


Evaluation: Scan Throughput

Dresden Database Systems Group

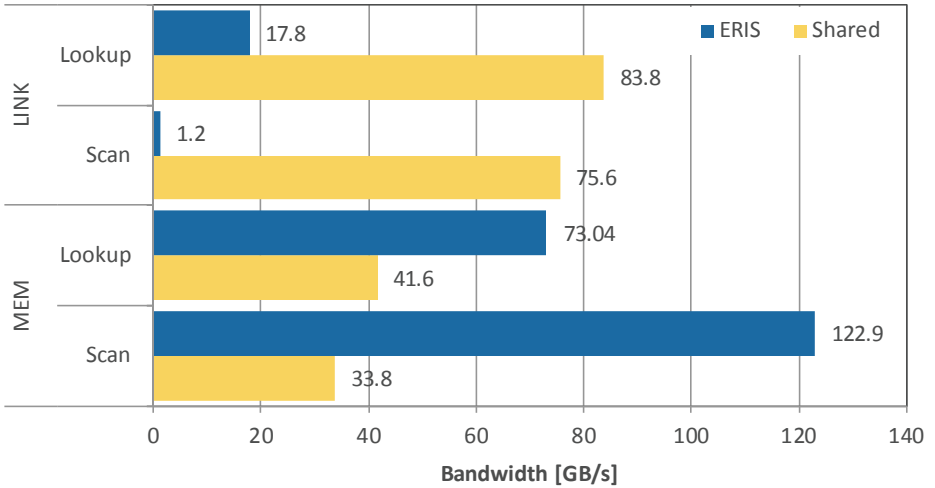
Scan Performance

- SGI Machine
- 488 cores – parallel scan
- 8 billion entries in the column store



Link and Memory Controller Activity

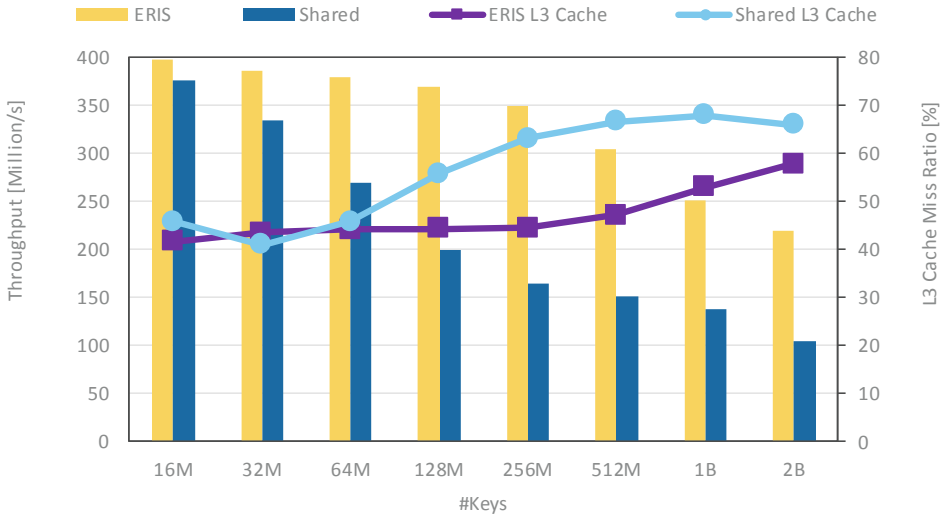
- AMD Machine
- Scan: 8B Keys
- Lookup: 1B Keys



Evaluation: L3 Cache Usage

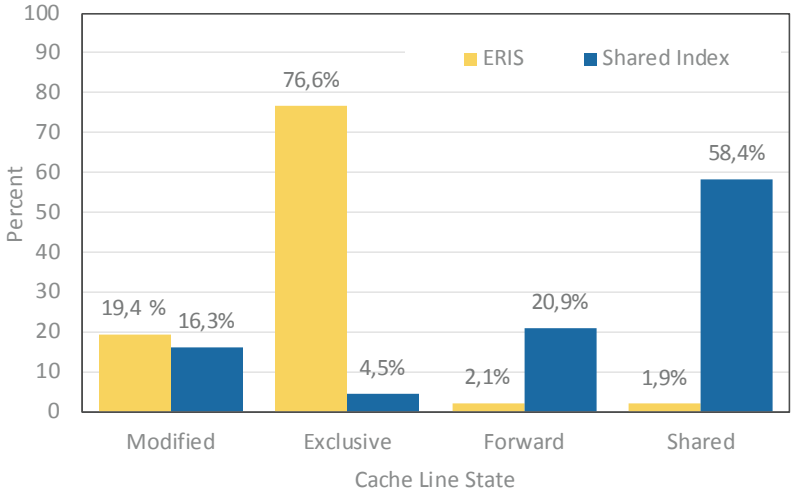
Dresden Database Systems Group

L3 Cache Usage – Index Lookup



L3 Cache Line State – Index Lookup

- Percentage of all hits
- 1B keys

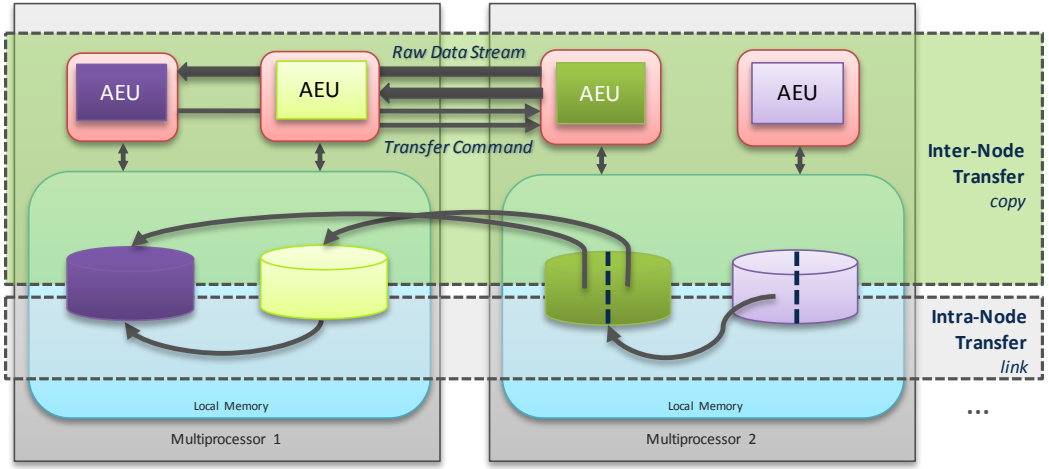


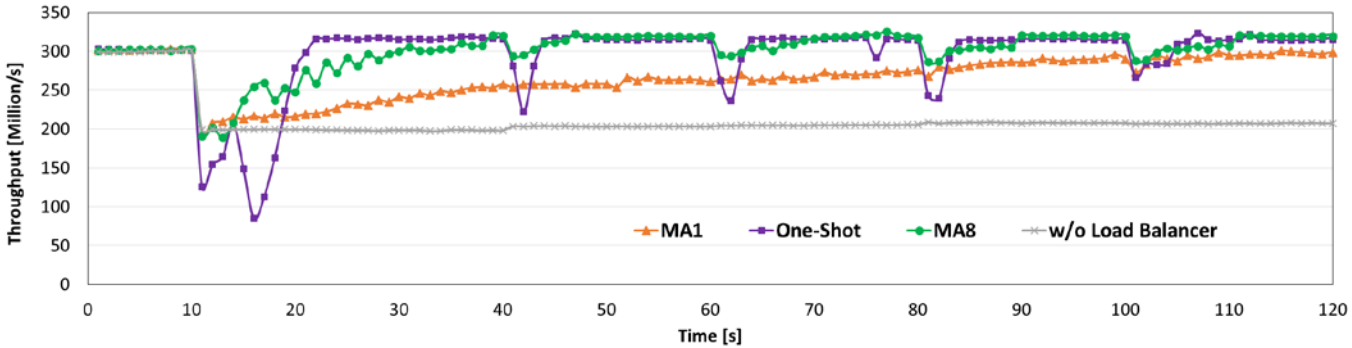
Key Component: Load Balancer

Dresden Database Systems Group

Balancing Strategies

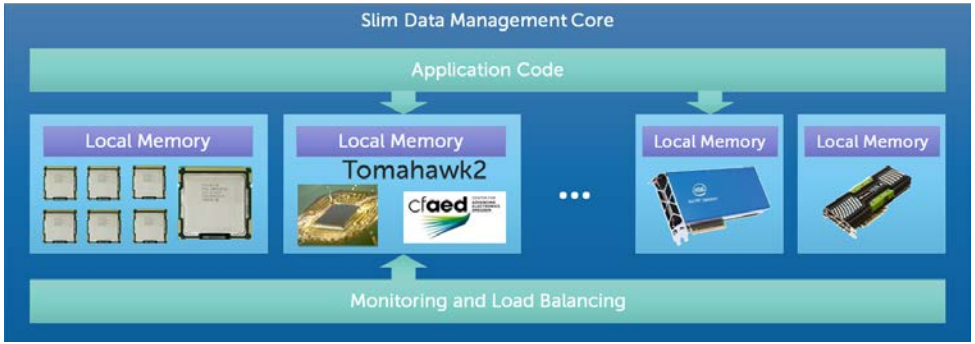
- Copy Strategy:
  copy data within NUMA systems between different sockets
- Link Strategy:
  delegate pointer, delay data movment





Summary and Conclusion

Conclusion



- MODERN HARDWARE DEMANDS SIGNIFICANT RE-THINKING OF DB ARCHITECTURES
  - extreme NUMA Systems: „distributed system" with common address space
  - customizable processors („dark silicon"): revival of the database machine?
  - communication: optical / wireless & RMDA:
    blurring the boundaries between scale-out and scale-up
  - Non-Volatile RAM: will be a game-changer!
- SLIM AND EXTENSIBLE DATA MANAGEMENT CORE TO
  - efficiently support today's applications requirements (mixed OLTP/OLAP workloads etc.)
  - embrace and exploit capabilities of modern hardware

Dresden Database Systems Group

HAEC
CRC 912
Highly Adaptive Energy-Efficient Computing

ScaDS DRESDEN LEIPZIG

DRESDEN concept

cfaed CENTER FOR ADVANCING ELECTRONICS DRESDEN

5G Lab Germany