TECHNISCHE UNIVERSITÄT DRESDEN

Dresden Database Systems Group

Overview on Hardware Optimizations for Database Engines

Annett Ungethüm, Dirk Habich, Tomas Karnagel, Sebastian Haas, Eric Mier, Gerhard Fettweis, Wolfgang Lehner

BTW 2017, Stuttgart, Germany, 2017-03-09

HAEC

cfaed CENTER FOR ADVANCING ELECTRONICS DRESDEN

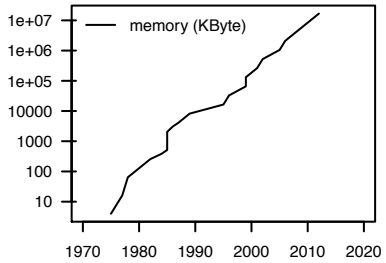
Interaction DB-Engine and Hardware

Applications/Database Engines

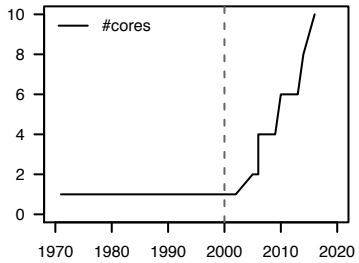
Modern Hardware

Well-Known Challenge:
Exploit hardware technology by specific data management techniques (indexing, data storage, query & transaction processing)

Main Memory



CPU

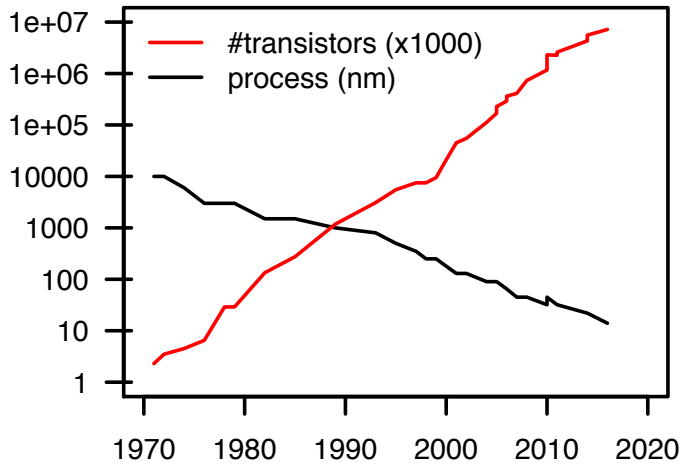


Era of Dark Silicon

Dresden Database Systems Group

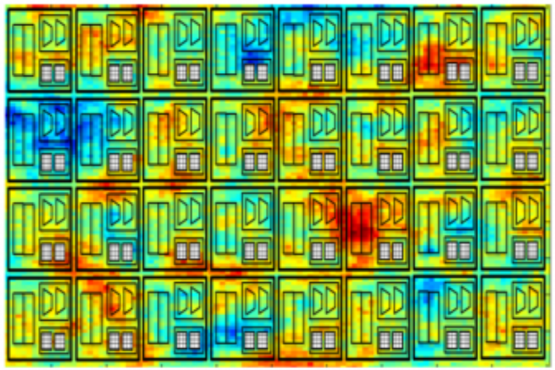
Moore's Law

- Number of transistors in a dense integrated circuit doubles approximately every two years.



Dark Silicon

- We can no longer power the transistors that Moore is giving us



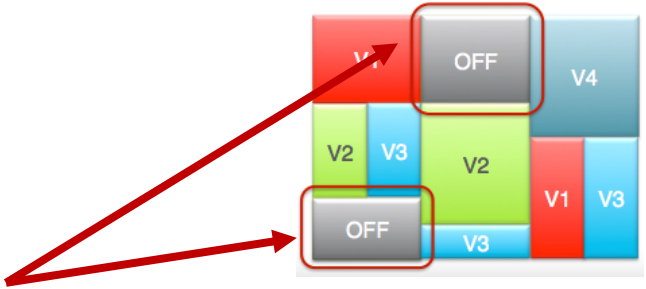
http://engineering.nyu.edu/garg/node/31

32-core (50% dark)

HW/SW Co-Design for DB-Engines

Dresden Database Systems Group

Applications/Database Engines



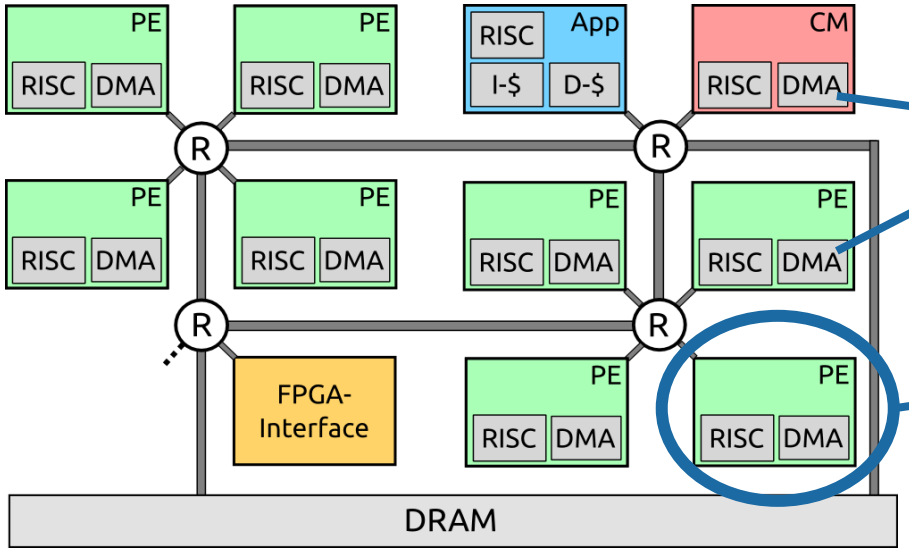
Challenge:
HW/SW Co-Design for Database Engines
Specialization of Hardware to overcome Dark Silicon

Modern Hardware

Outline

Dresden Database Systems Group

Hardware Foundation



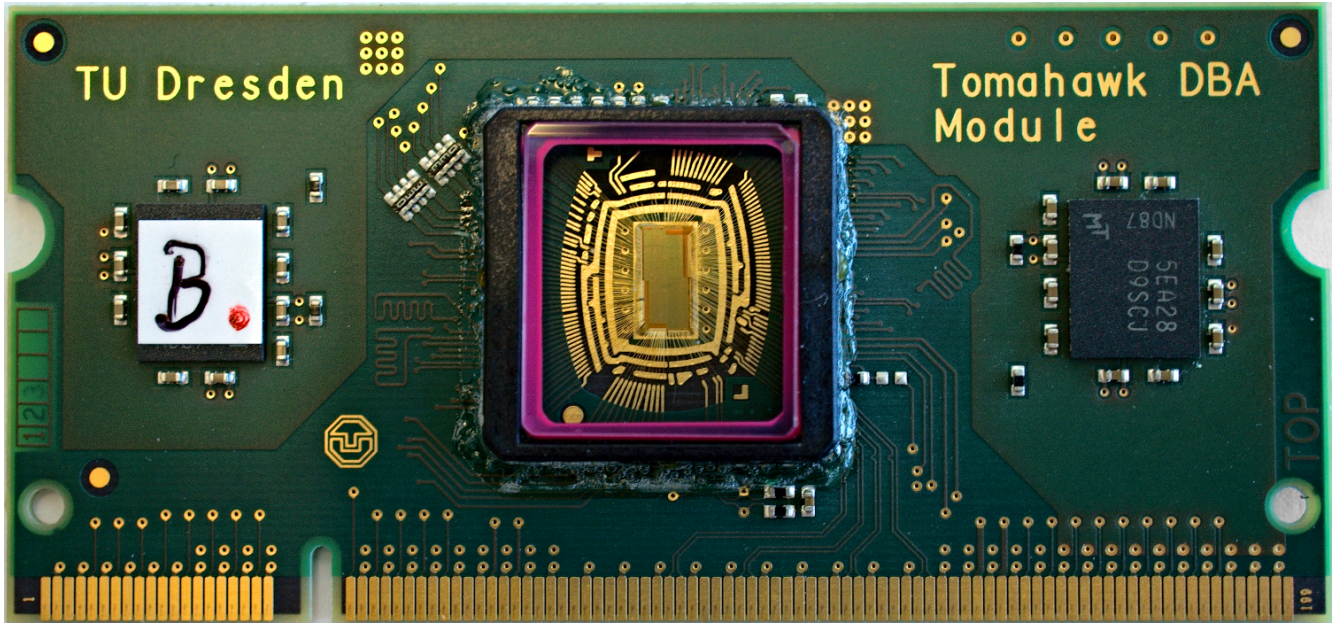
Intelligent DMA Controller

Extensions For Processing Elements

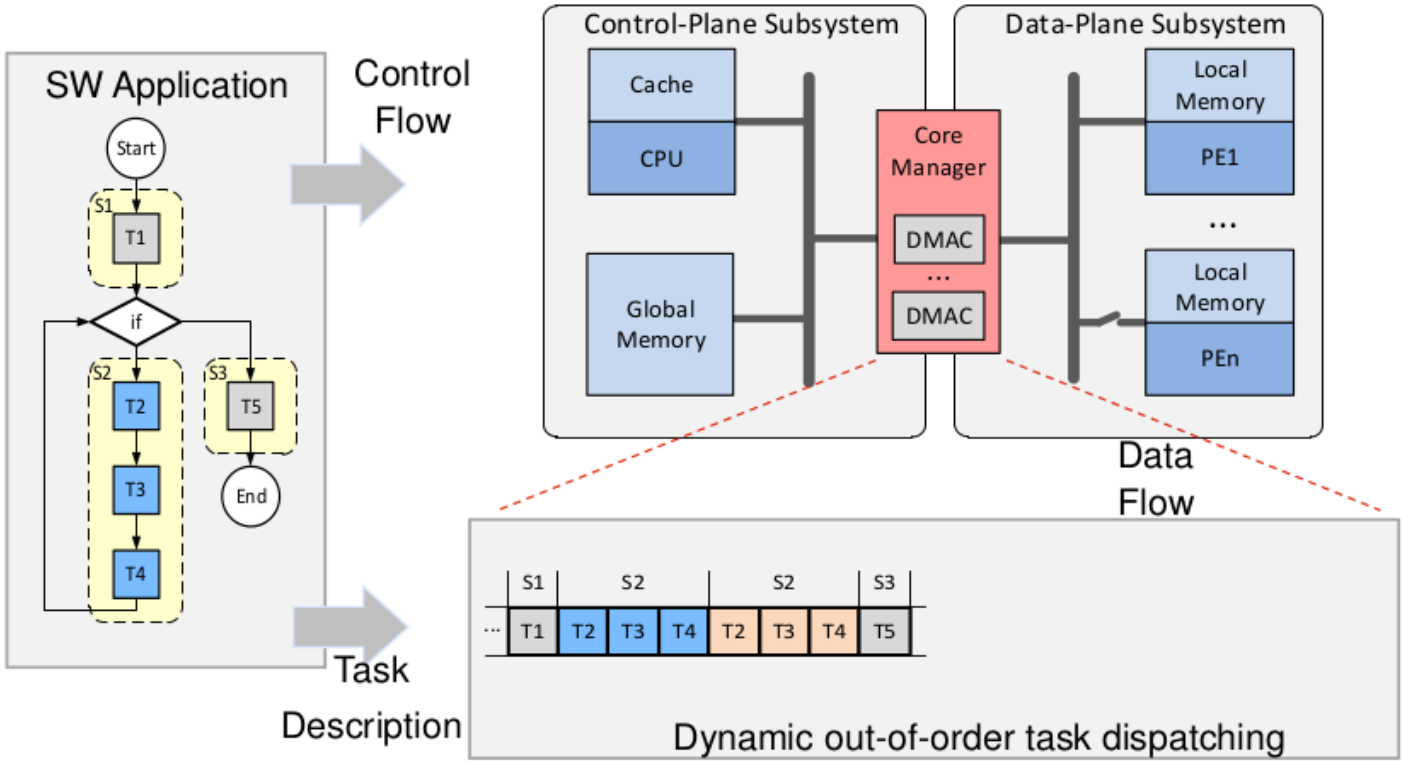
Hardware Foundation

Dresden Database Systems Group

Tomahawk Platform

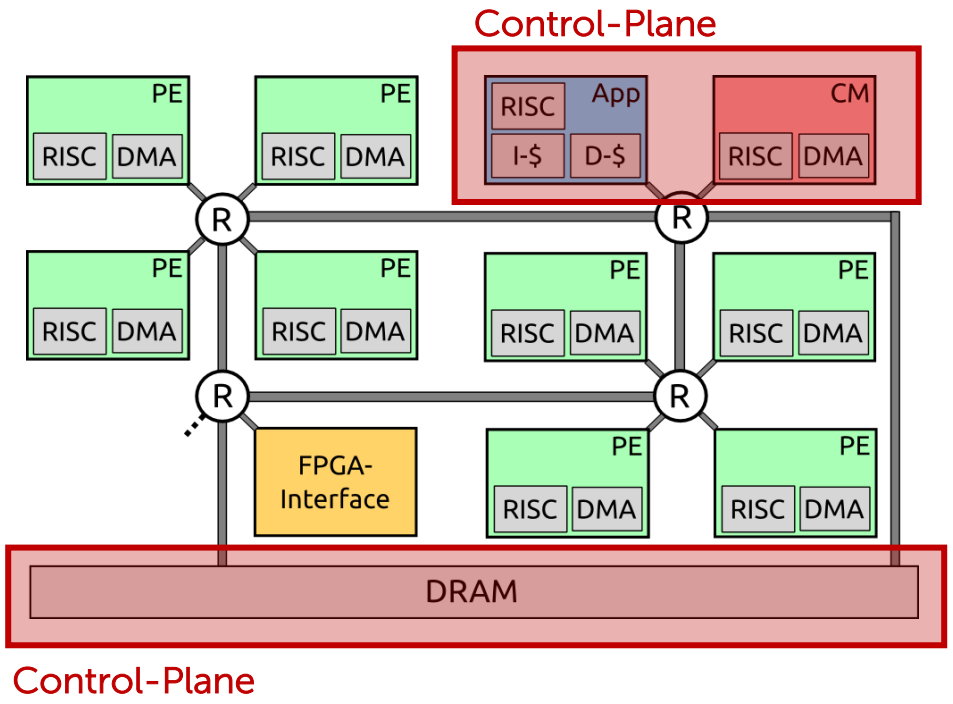


Hardware Foundation – Zoom In



Hardware Foundation – Zoom In (2)

Dresden Database Systems Group



Core Manager (CM)

- Extended Xtensa-LX5 from Tensilica (now Cadence)
- 32KB for code
- 64KB for data

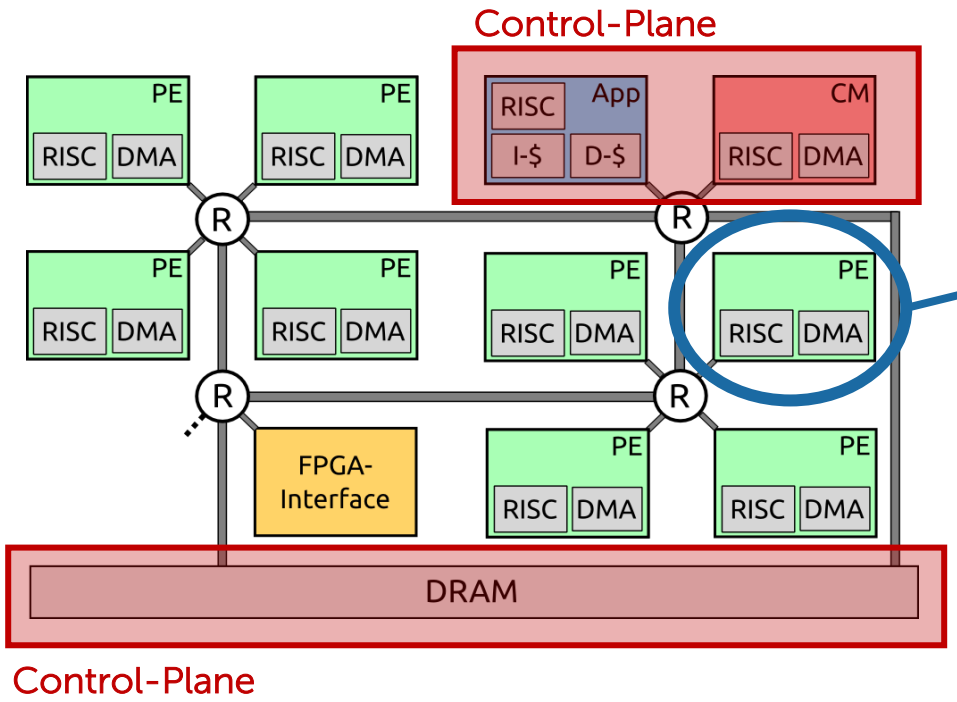
Processing Elements (PE)

- Xtensa-LX5 from Tensilica (now Cadence)
- 32KB for code
- 2x32KB for data on PE

Application Core (APP)

- 570T core from Tensilica (now Cadence)

Outline



PART I:

EXTENSIONS OF PROCESSING ELEMENTS

Development Flow

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

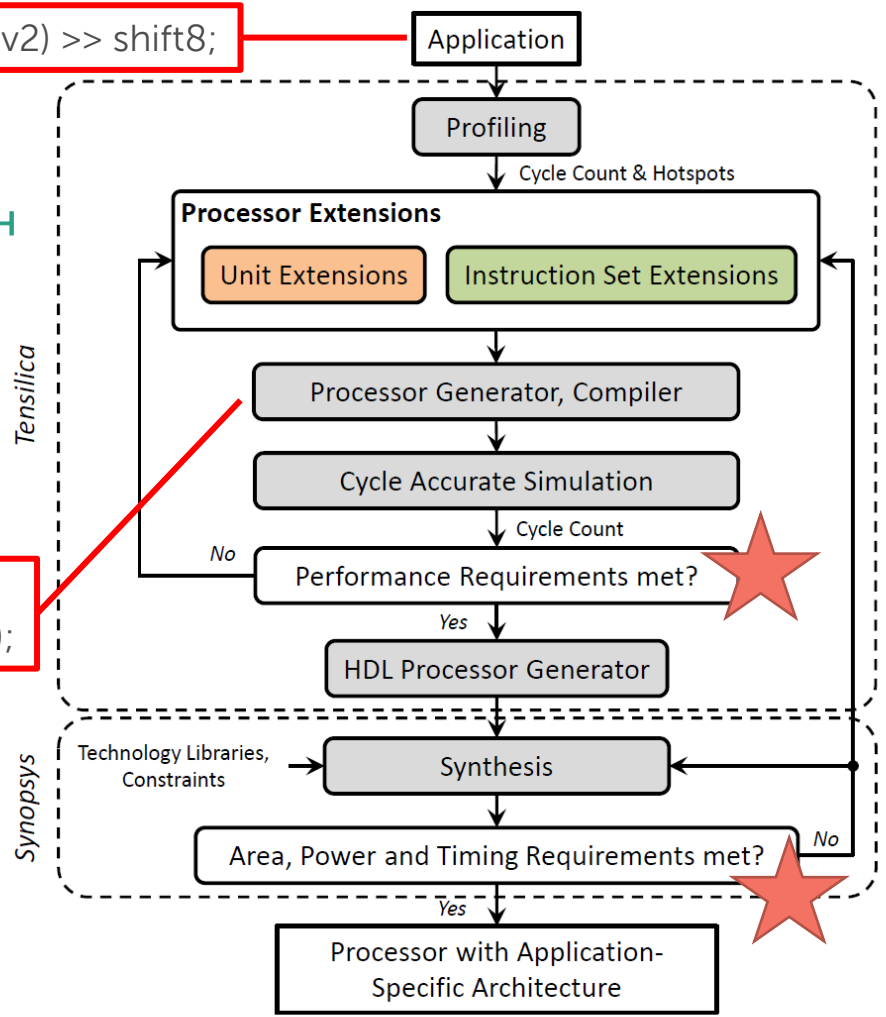
Development of instruction set extensions with Tensilica tools

- Tensilica Instruction Extension (TIE) language
- C/TIE compiler
- Cycle accurate simulator/debugger
- Processor generator

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

Synthesis of RTL code

- Synopsys Design Compiler, PrimeTime PX
- TSMC CMOS LP 65nm libraries

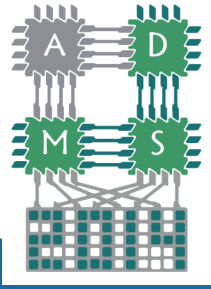


Investigated Database Primitives

Dresden Database Systems Group

ASAP 2016

The 27th Annual IEEE International Conference on Application-specific Systems, Architectures and Processors

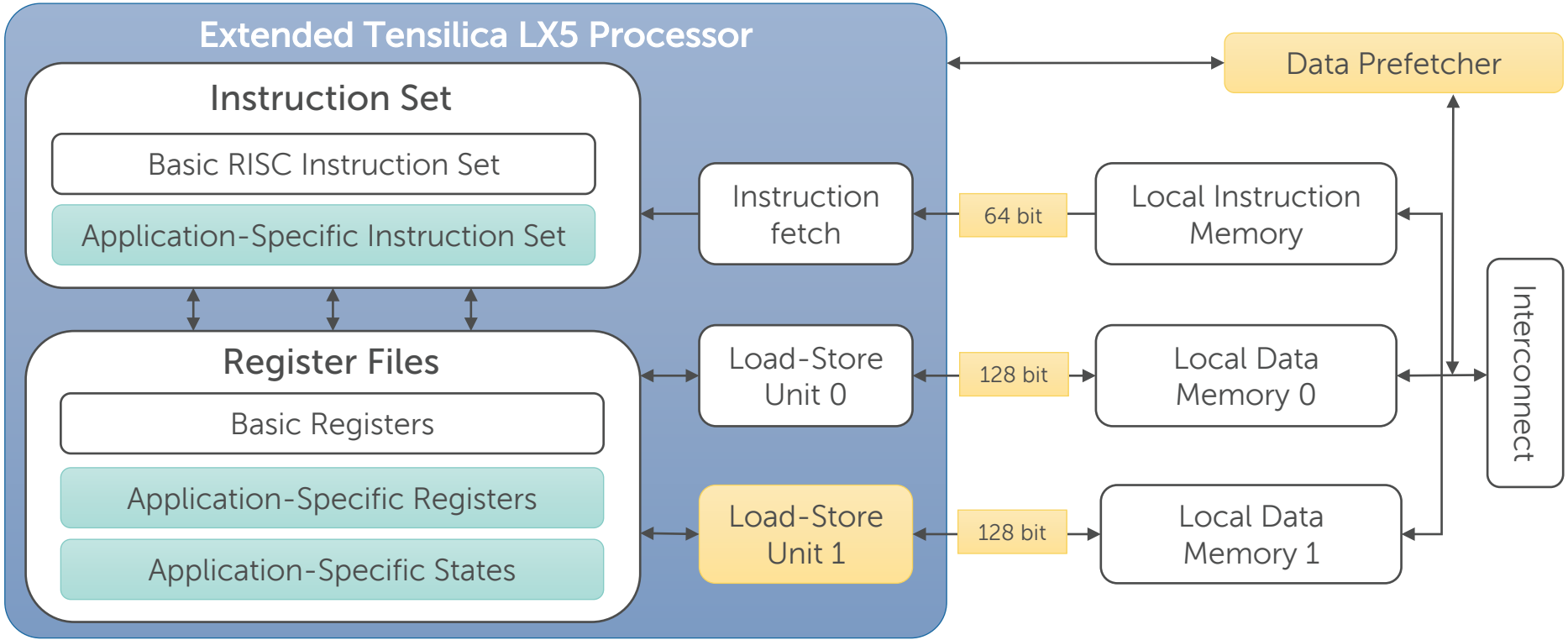


ACM SIGMOD 2014

	Bitmap Compression and Processing (AND, OR, XOR)			Hashing						Sorted Set Operations				
Primivites	WAH	PLWAH	COMPAX	Hash + Lookup	Hash + Insert	Hash Keys	Hash Sampling	CityHash32	Merge Sort	Intersection	Union	Difference	Sort-Merge Join	Sort-Merge Aggregation (SUM)

General Approach for all Extensions

Dresden Database Systems Group



Bitmap Primitives

Dresden Database Systems Group

Bitmaps are a Special Kind of Index

Table T

OID	X	bitmap index =0	=1	=2	=3
1	0	1	0	0	0
2	1	0	1	0	0
3	3	0	0	0	1
4	2	0	0	1	0
5	3	0	0	0	1
6	3	0	0	0	1
7	1	0	1	0	0
8	3	0	0	0	1
		b_1	b_2	b_3	b_4

Bit-wise OR

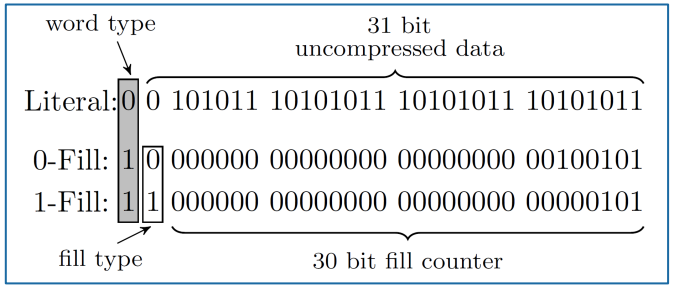
select * from T where X < 2

Bitmaps Compression

- bit length equals number of tuples

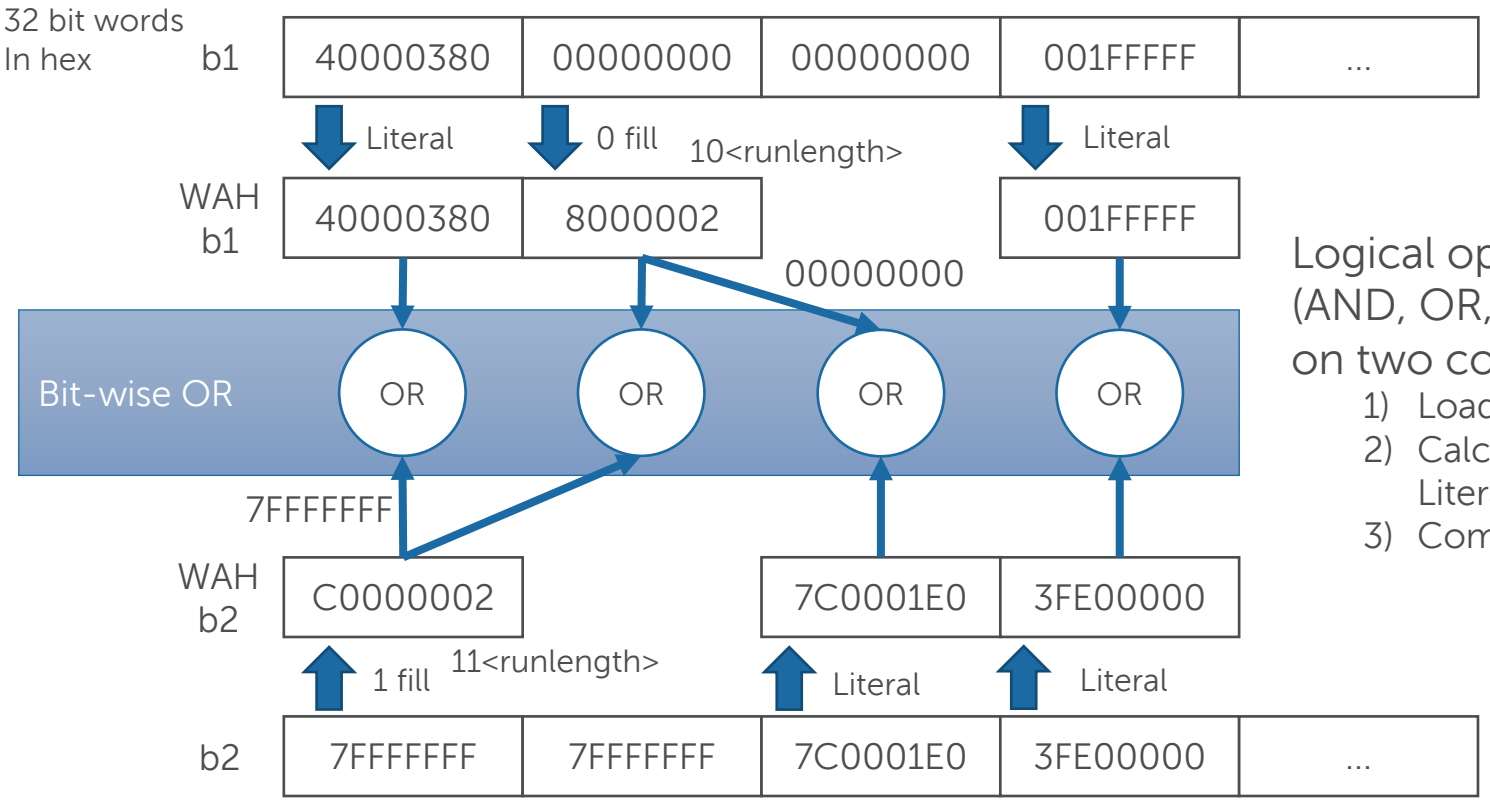
Word-Aligned Hybrid (WAH) Code

- Stateless compression
- Run-length-encoding (RLE)
  - run of 0's and 1's
- WAH bitmaps contain RLE
  - compressed fills and
  - uncompressed literals



Bit-Wise OR on Compressed Bitmaps

Dresden Database Systems Group



Logical operations (AND, OR, XOR) on two compressed bitmaps

1) Load WAH word(s)
2) Calculate output (Fill-Fill, Literal-Fill, Literal-Literal)
3) Combine output

C-Code

Dresden Database Systems Group

```
WHILE(XIDX!=XSIZE && YIDX!=YSIZE)     {
   //new X or Y? Calculate new fill count …

   if(XisFill==1 && YisFill==1) { //2 fills
     if(XfillWords<YfillWords)
     min=XfillWords;
     else
     min=YfillWords;
     writeFill(comprResultBI,&Zidx,X[Xidx]|Y[Yidx],min);
     XfillWords-=min;
     YfillWords-=min;
   }
   else if((XisFill==1 && YisFill==0) || (XisFill==0 &&
   YisFill==1)) {
     if(XisFill==1){
     XfillWords--;
     if((X[Xidx]&0xC0000000)==0xC0000000)
           writeFill(comprResultBI, &Zidx, 0xC0000000, 1);
     else { comprResultBI[Zidx]=Y[Yidx]; Zidx++; }
   }
```

Fill-Fill

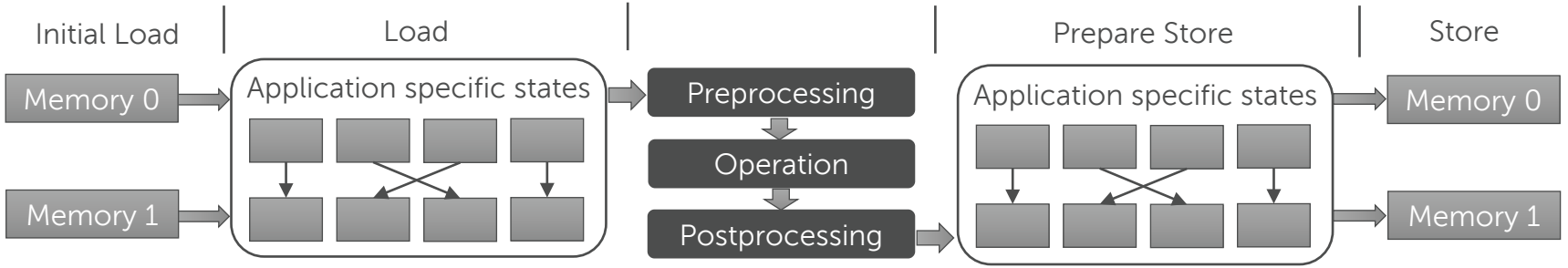
Literal-Fill

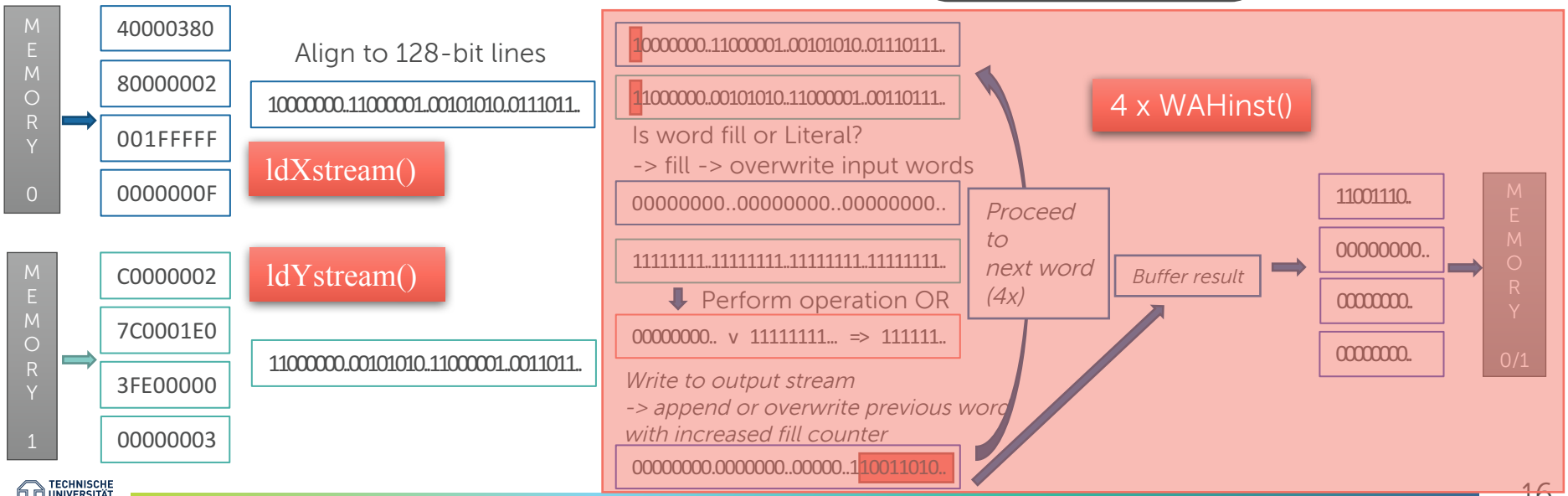
```
if(YisFill==1){
  YfillWords--;
if((Y[Yidx]&0xC0000000)==0xC0000000)
        writeFill(comprResultBI, &Zidx, 0xC0000000, 1);
                else {comprResultBI[Zidx]=X[Xidx]; Zidx++; }
    }
 }
 else {
    result=X[Xidx]|Y[Yidx];
    if((result&0x7FFFFFFF)==0x7FFFFFFF)
        writeFill(comprResultBI, &Zidx, 0xC0000000, 1);
    else if((result&0x7FFFFFFF)==0)
        writeFill(comprResultBI, &Zidx, 0x80000000, 1);
    else { comprResultBI[Zidx]=X[Xidx]|Y[Yidx]; Zidx++; }
 }
}
```

Literal-Literal

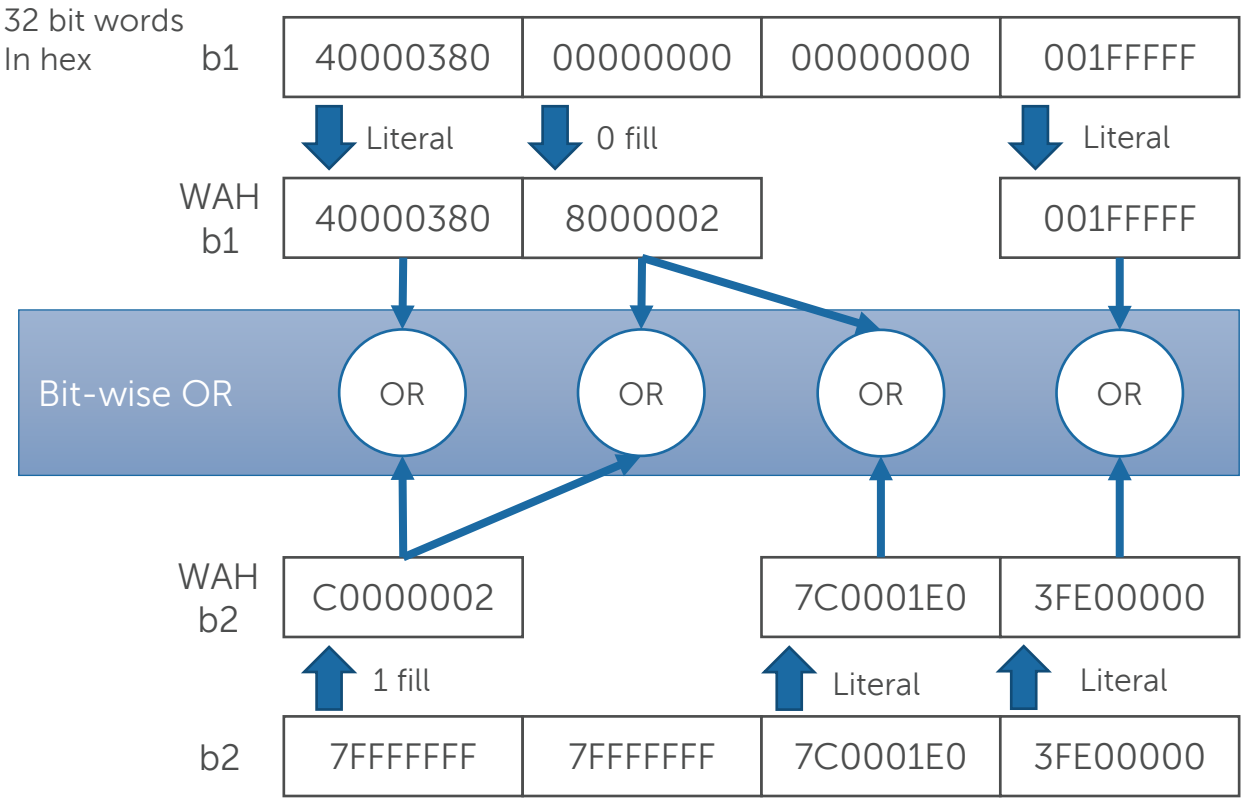
Processing with PE Extension

Dresden Database Systems Group





Bit-Wise OR on Compressed Bitmaps



Code with Extension

```
do{
        ldXstream();
        ldYstream();
        WAHinst();
        WAHinst();
        WAHinst();
} while(WAHinst());
```

Many More Extensions

Dresden Database Systems Group

	Bitmap Compression and Processing (AND, OR, XOR)			Hashing						Sorted Set Operations				
Extension / Processor	WAH	PLWAH	COMPAX	Hash + Lookup	Hash + Insert	Hash Keys	Hash Sampling	CityHash32	Merge Sort	Intersection	Union	Difference	Sort-Merge Join	Sort-Merge Aggregation (SUM)
BitiX	X	X	X											
HASHI				X	X	X	X	X						
Titan3D						X			X	X	X	X	X	X
Tomahawk DBA	X					X	X		X	X		X		

Evaluation

Dresden Database Systems Group

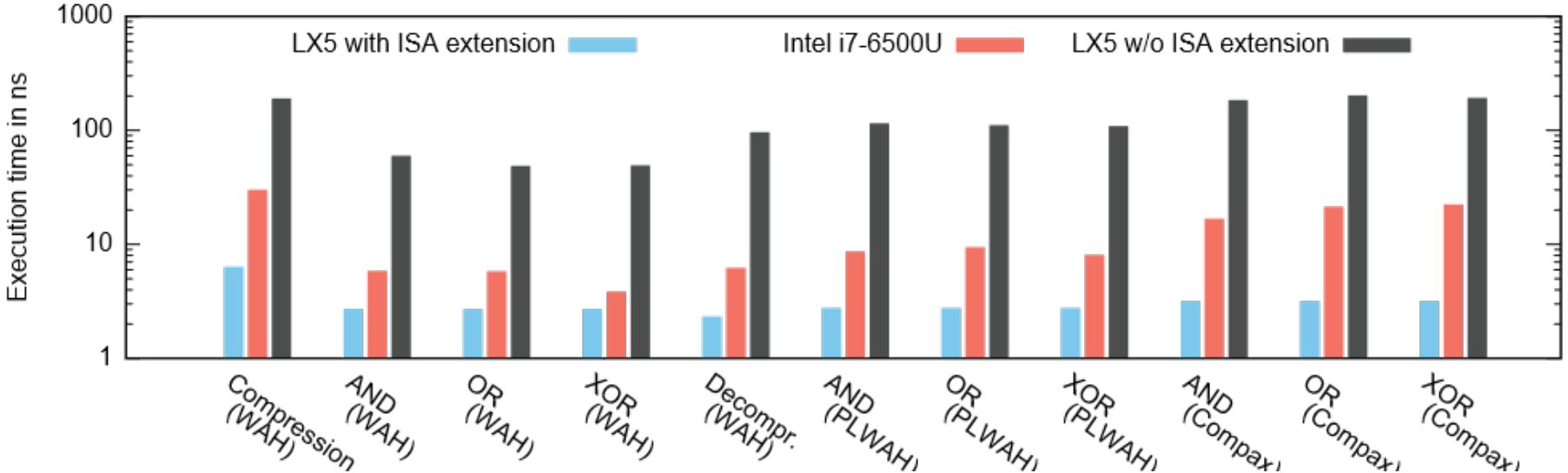
Reference Processors

- Tomahawk DBA Processor --> Set of different DB-Extensions for WAH-Compression, Hashing, and Sortes-Set Operations

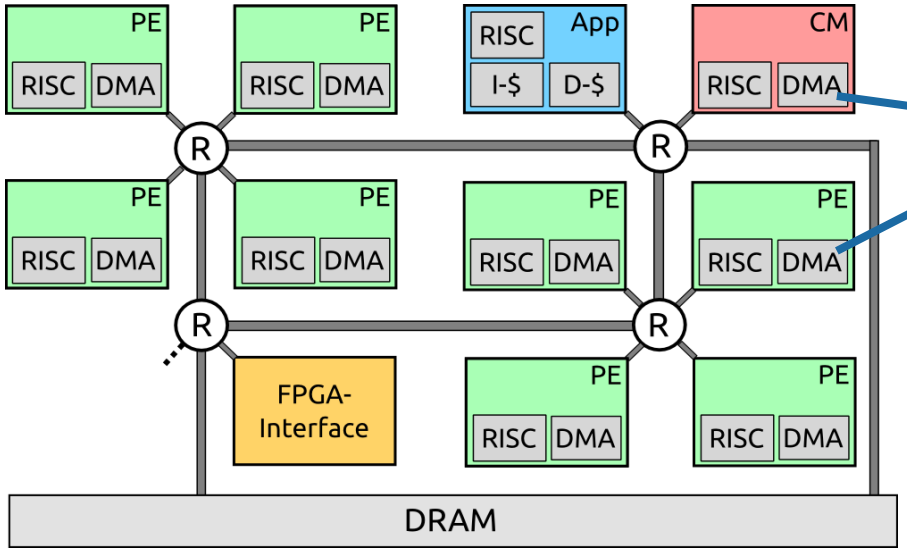
Processor	Description	Technology [nm]	A_{total} [mm²]	f_{MAX} [GHz]	P_{MAX} [W] @ f_{MAX}
Tomahawk without DBA	Basic Xtensa LX5 without instruction set extensions, 1 LSU, 32-bit memory interface	28	15.92	0.555	0.7
Tomahawk with DBA	Set of different DB-Extensions for WAH-Compression, Hashing and Sorted-Set Operations	28	↑ 18	↓ 0.5	↑ 0.753
Comparison					
Intel i7-6500U	Low-power Intel 2-core processor based on Skylake architecture, 4MB L3 cache	14	99*	3.1	25

Evaluation - Bitmaps

Dresden Database Systems Group



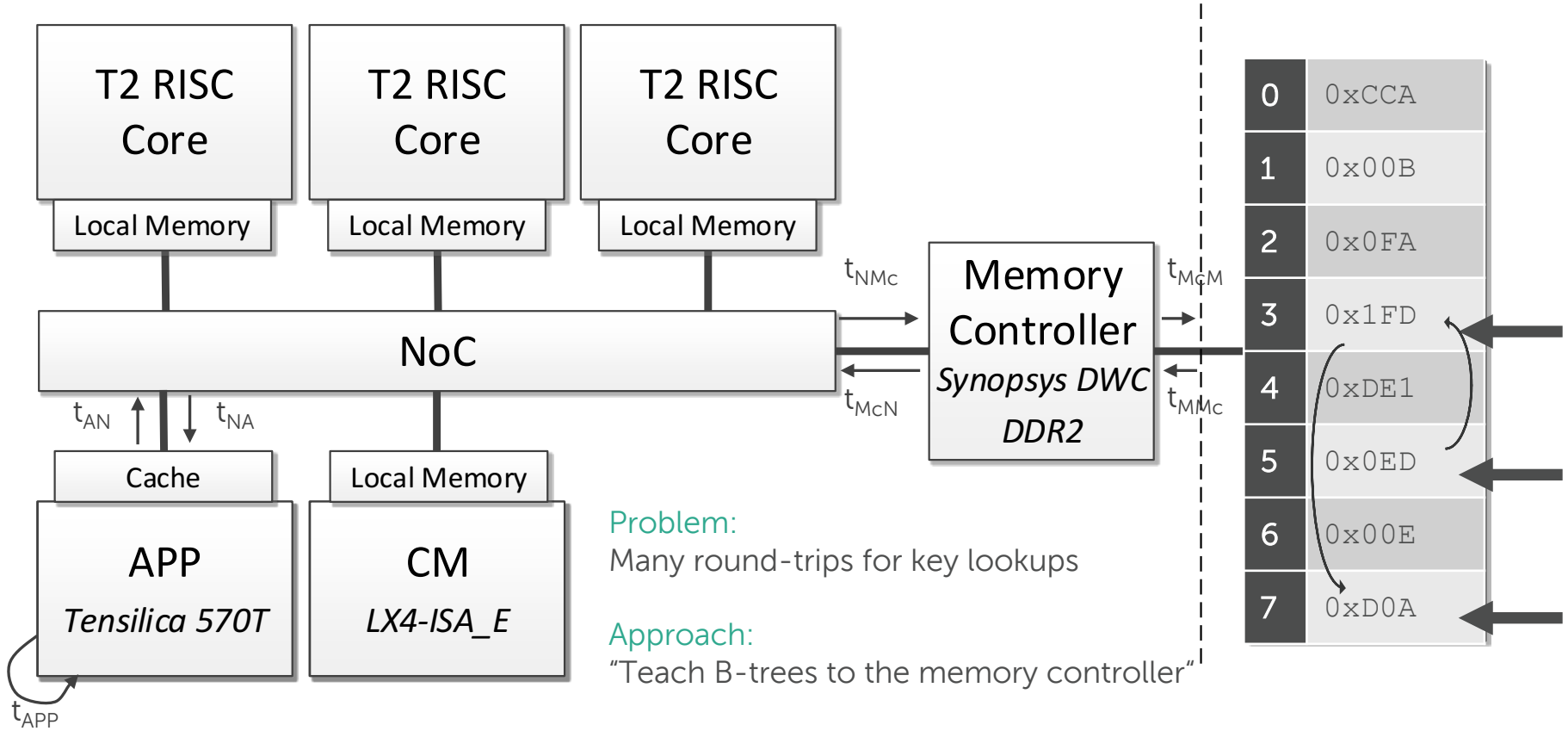
Outline



PART 2:

INTELLIGENT DMA CONTROLLER

Problem Statement

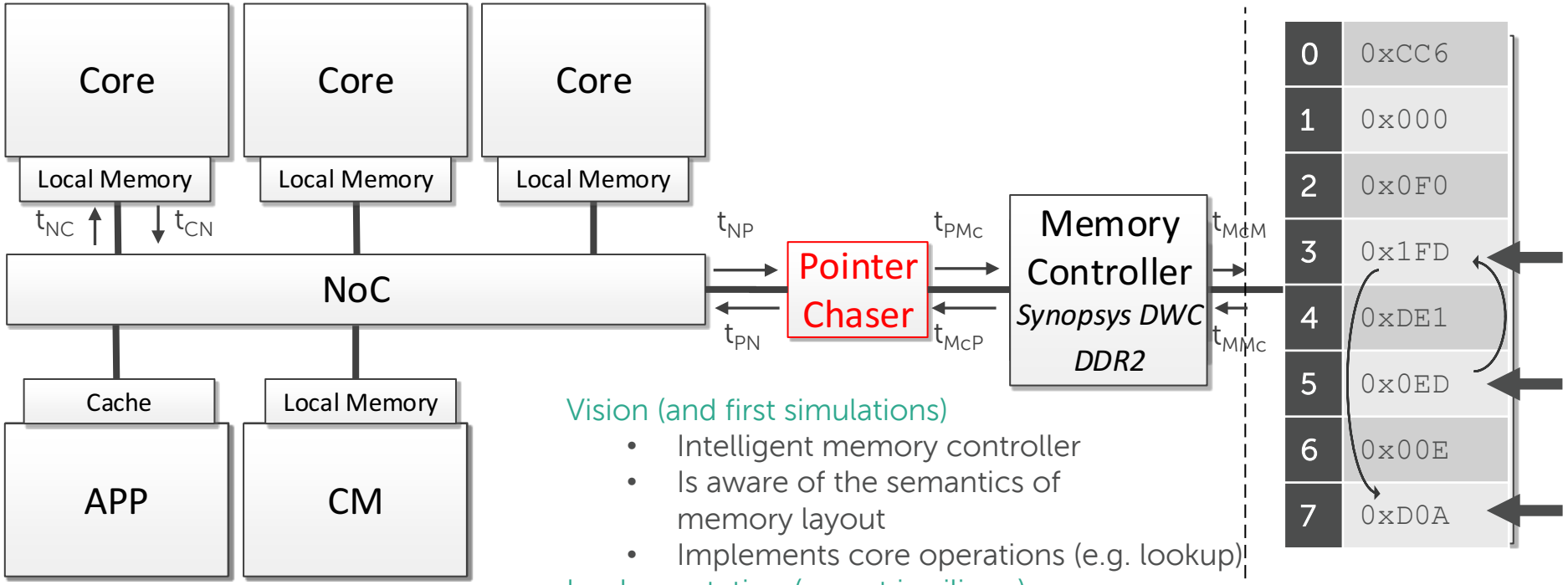


Problem:
Many round-trips for key lookups

Approach:
"Teach B-trees to the memory controller"

Intelligent Main Memory Controller (iDMA)

Dresden Database Systems Group



Vision (and first simulations)

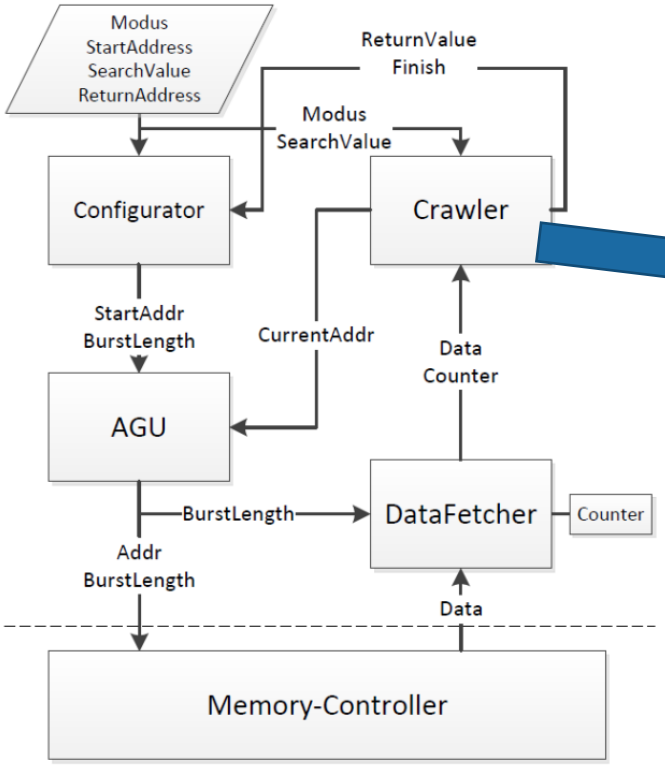
- Intelligent memory controller
- Is aware of the semantics of memory layout
- Implements core operations (e.g. lookup)

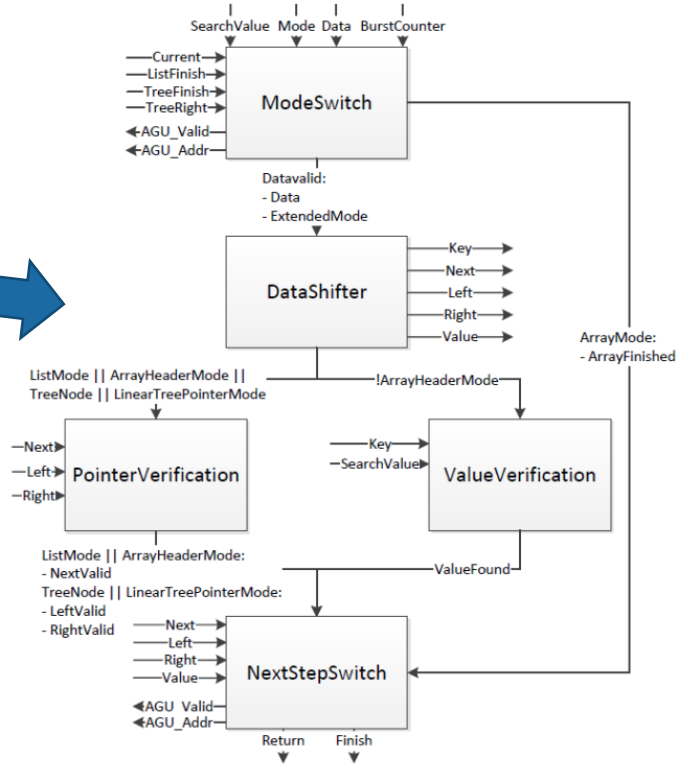
Implementation (no yet in silicon)

- 0,183mm² PE with 200Mhz

First iDMA Design

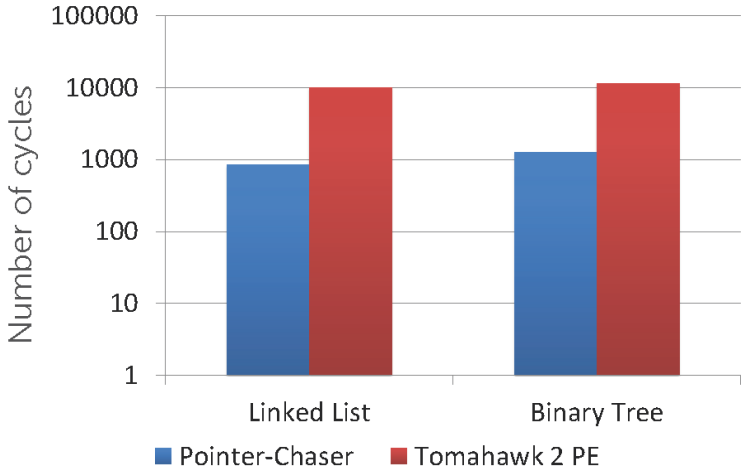
Dresden Database Systems Group



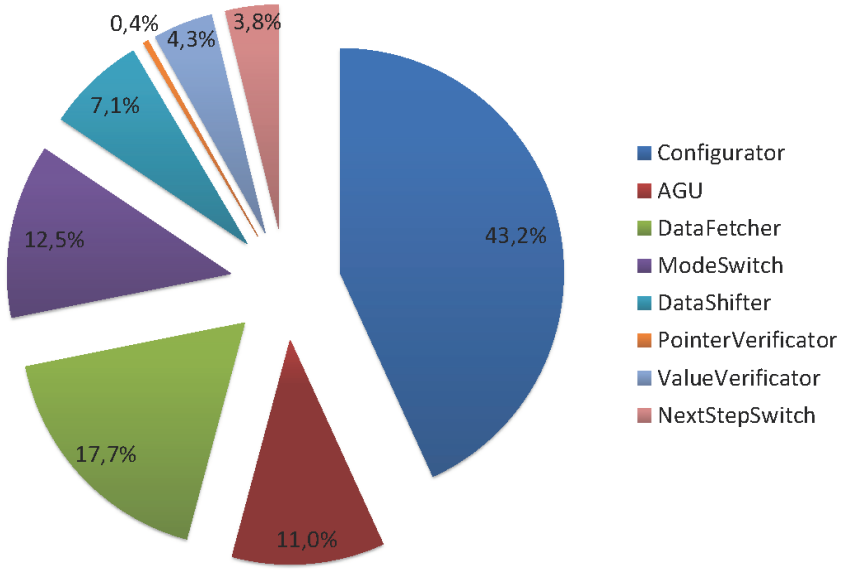


Evaluation using Simulator

Dresden Database Systems Group

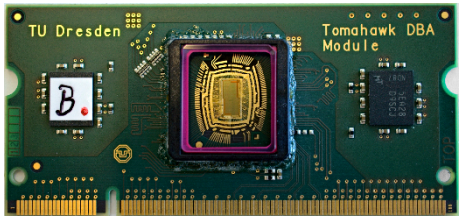


(a) Execution times in number of clocks of iDMA with pointer chaser and a processor with an SDRAM



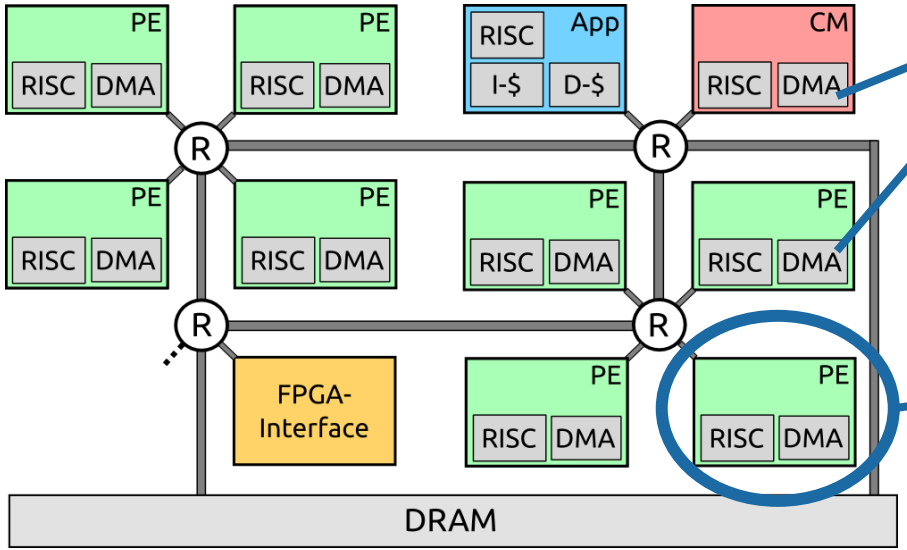
(b) Percentage area fraction of the submodules of the pointer chaser component

Summary

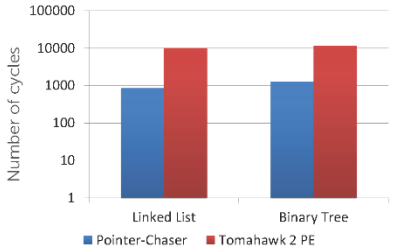


Dresden Database Systems Group

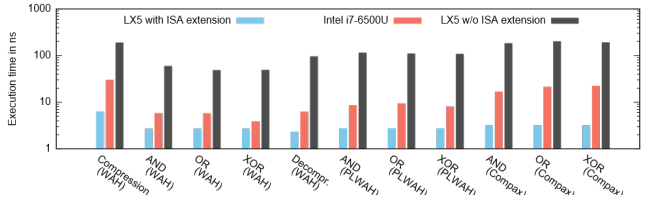
Hardware Foundation



Intelligent DMA Controller



Extensions For Processing Elements



TECHNISCHE UNIVERSITÄT DRESDEN

Dresden Database Systems Group

Overview on Hardware Optimizations for Database Engines

Annett Ungethüm, Dirk Habich, Tomas Karnagel, Sebastian Haas, Eric Mier, Gerhard Fettweis, Wolfgang Lehner

BTW 2017, Stuttgart, Germany, 2017-03-09

HAEC

cfaed CENTER FOR ADVANCING ELECTRONICS DRESDEN