Year 3 Review
Brussels, Dec 14, 2007

Achievements and Perspectives :

Compilers and Timing Analysis

Cluster leader : Rainer Leupers (Y3)
RWTH Aachen
Peter Marwedel (Y4)
TU Dortmund

High-Level Objectives

- Accept the challenges for compilers for embedded systems
  - new architectures
  - power efficiency
  - run time efficiency
  - dependability,
  - …
- Combine expertise of leading European research institutes and companies in embedded compilers
- Build on common platforms
- ARTIST2 as an infrastructure

ACE

AbsInt Angewandte Informatik

imec

RWTH AACHEN
RHEINISCH-WESTFÄLISCHE TECHNISCHE HOCHSCHULE AACHEN

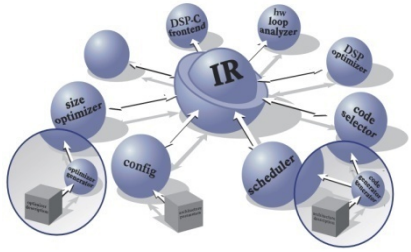
Technische Universität

TU WIEN
TECHNISCHE UNIVERSITÄT WIEN
VIENNA UNIVERSITY OF TECHNOLOGY

TU berlin

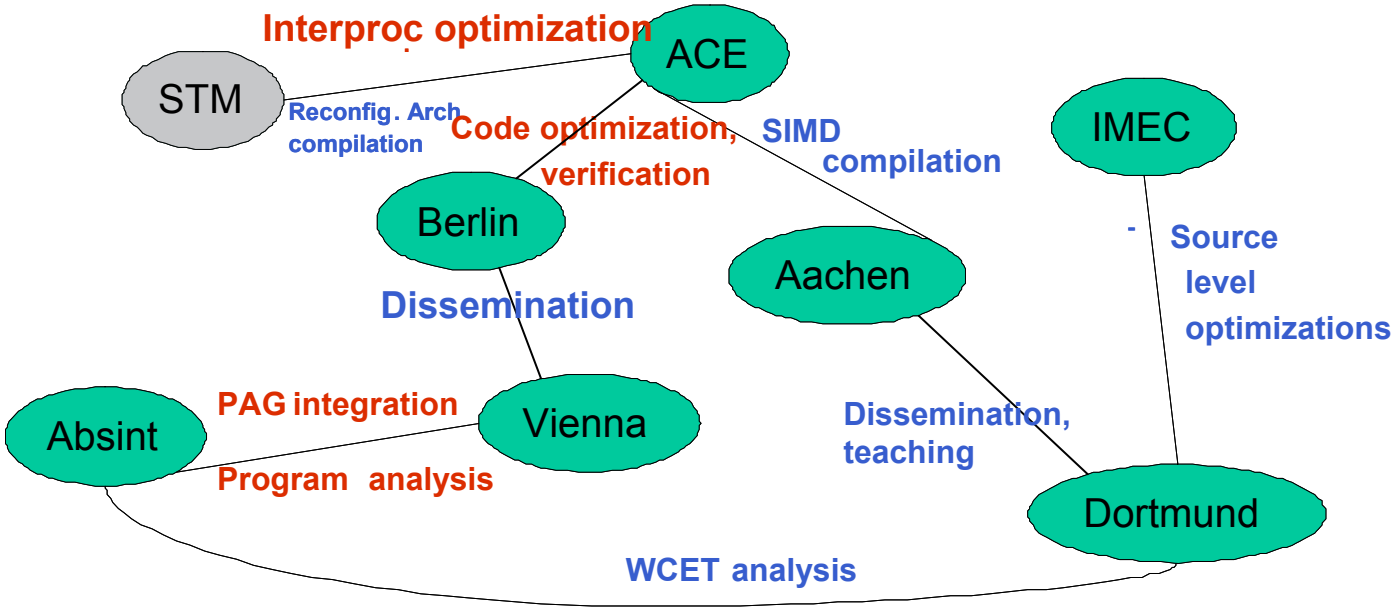
Approach

- Common platform for compiler development
  - CoSy key platform for new compiler optimizations
- Source-level code optimizations
  - Complementing capabilities of traditional compilers
  - Large degree of portability
  - Compatible with gcc approaches



Integration and Building Excellence

- Not all partners have same research interests ☞
  - 1 major activity: platform based code optimization and verification (S. Glesner)
  - 2 global cluster meetings per year
  - "Mini-cluster" structure for day-to-day cooperation
  - Each partner himself has a large network beyond ARTIST2



Integration and Building Excellence

- Available results with ARTIST2 sponsoring include:
  - ACE community meeting (Oct 2006, AMS)
  - ACE CoSy academic workshop with ARTIST2 participants (AC, TUB, Mar 2007)
  - ACE guest lectures in compiler classes at Aachen
  - IMEC training on DTSE methodology
  - Aachen/ACE code optimization engines in productization
  - Berlin/ACE: advanced code optimization verification in place
  - Vienna/Absint: PAG/ROSE infrastructure integration (SATIrE)
  - Dortmund/Absint: WCET aware compiler tool chain + optimizations
  - Joint publications at leading conferences, e.g. DATE, CODES/ISSS:
    - Dortmund/Bologna, Dortmund/Absint, IMEC/Bologna, Aachen/ACE ,..

Integration and Building Excellence

- Dissemination with ARTIST2 support (subset):
  - SCOPES: leading European event in code generation for embedded systems (Dortmund, ACE,...)
  - ESWeek/Salzburg: leading international event on embedded systems and architectures
  - CASA workshop
  - HiPEAC/ACACES: compiler courses for >100 students
  - SHAPES/CASTNESS: winter school
  - KDUbiq workshop tutorial P.Marwedel
  - ALARI: ES courses (P. Marwedel, R. Leupers)
  - ACE univ booth (open Cosy) at DATE 2007
  - EPFL: Programmable Digital Systems Design course for industry
  - COCV ETAPS 2007 Workshop (S. Glesner, J. Knoop)

Assessment at Y0+3

- What is going well
  - Participation of key industrial players (Absint, ACE)
  - Common compiler platform established
  - Mini-cluster structure is very effective
  - TU Berlin as new core partner
  - Many visible results (integrated SW prototypes, successful meetings, staff mobility, joint papers, dissemination activities, high visibility in US, Asia)
  - Reduced reporting overhead over Y2
  - Improved cooperation between European academic & industrial players
- What is not going well
  - Low ratio funding/(management overhead)

Structural changes

- Peter Marwedel takes over as cluster leader for Y4
- (… and Björn Lisper for TA)

Specific future work

- ACE/Aachen: architecture aware code optimization engines built around the CoSy platform.
- ACE/Berlin: improving the Itanium compiler platform
- Absint/USaar: WCET analysis
- Absint/Vienna: whole program source code analysis; generation of WCET annotations.
- AbsInt/Dortmund: studies on the influence of standard compiler optimizations on the program's WCET
- Dortmund/IMEC: placement of data in distributed/shared data memory hierarchy of a MPSoC platform.
- Dissemination activities by all partners as in Y1, Y2, Y3

Scientific Highlights

- SIMD-enabling loop transformations
  - Extended interface
  - Strip mining (new SIMD-able inner loop created)
  - Scalar expansion (Replace scalar by array to reduce dependencies)
  - Loop peeling (Loop split into two, main loop can be SIMDfied
  - Tools added to CoSy tool set

Scientific Highlights

- Conditional execution
  - Hampering within ITE blocks: Function calls, Non cond. executable code
  - Split statement into one convertible and one non convertible statement.

```
if (x==0) {
        A;
        B;
        C;
} else {
        D;
        E;
        F();
}
```

```
bool tmp = (x==0);
if (tmp) {
        A;
} else {
        D;
        E;
}
if (tmp) {
        B;
        C;
} else {
        F();
}
```

```
    p4 = (x == 0);
    p5 = !p4;
[p4] goto Then;
[p4] A;
[p5] D;
[p5] E;
    nop;
    …
    nop;
```

Then:
B;
C;

Else:
F();

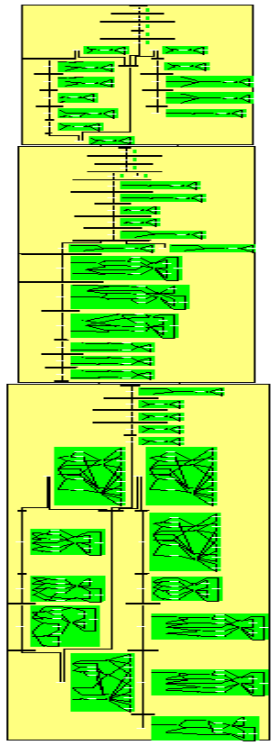
End:
…

RWTH AACHEN
RHEINISCH-WESTFÄLISCHE TECHNISCHE HOCHSCHULE AACHEN

Information Society

TU Vienna

- Implemented shape analysis for C++ subset
- Computed may/must alias pairs from shape graphs, allowing comparison with pointer analyses w.r.t. precision
- Automatically generated may/must alias pairs as C++ program annotations
- Generated external program representation for tool interoperability



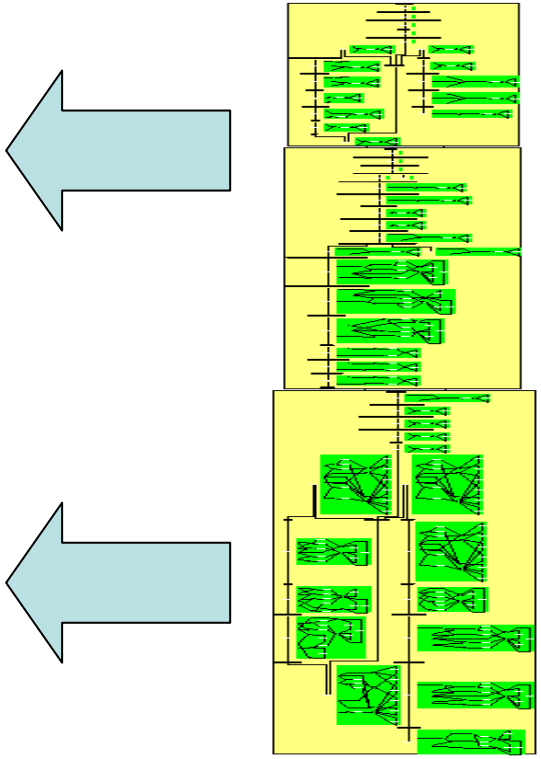
TU WIEN
Technische Universität Wien
Vienna University of Technology

AbsInt
Angewandte Informatik

Information Society

Example: Aliasing Information as Annotations

```
// pre must_aliases : {}
// pre may_aliases : {(l,t),(l,x),(l,y),(l,y->next),(t,
while(x != ((0))) {
  // pre must_aliases : {}
  // pre may_aliases : {(l,t),(l,x),(l,y),(l,y->next),(
  t = y;
  // post,pre must_aliases : {(t,y)}
  // post,pre may_aliases : {(l,t),(l,x),(l,y),(l,y->n
  y = x;
  // post,pre must_aliases : {(x,y)}
  // post,pre may_aliases : {(l,t),(l,x),(l,y),(x,y)}
  x = (x -> next);
  // post,pre must_aliases : {(x,y -> next)}
  // post,pre may_aliases : {(l,t),(l,y),(x,y->next)}
  y -> next = t;
  // post must_aliases : {}
  // post may_aliases : {(l,t),(l,y),(l,y->next),(t,y->next)}
}
```



TECHNISCHE UNIVERSITÄT WIEN

VIENNA UNIVERSITY OF TECHNOLOGY

Information Society Technologies

Berlin

- Formalized the scheduling phase in the theorem prover Isabelle/HOL
  - Discovered a bug in the scheduler of the popular GNU assembler
- Developed a backend specification for the Intel Itanium processor
  - Optimizations to increase parallelism by Block Extension and Speculative Techniques

IMEC

- Multiple fine-grain dynamic memory allocation design options
- Extraction of application specific information
- Extraction of memory hierarchy specific information
- ☞ Unique dynamic memory allocator, which is compiled with the software application

Dortmund

- ☞ See TA subcluster
- ☞ See resource aware design

Q&A ?

Year 3 Review
Paris, December 14th, 2007

Achievements and Perspectives :

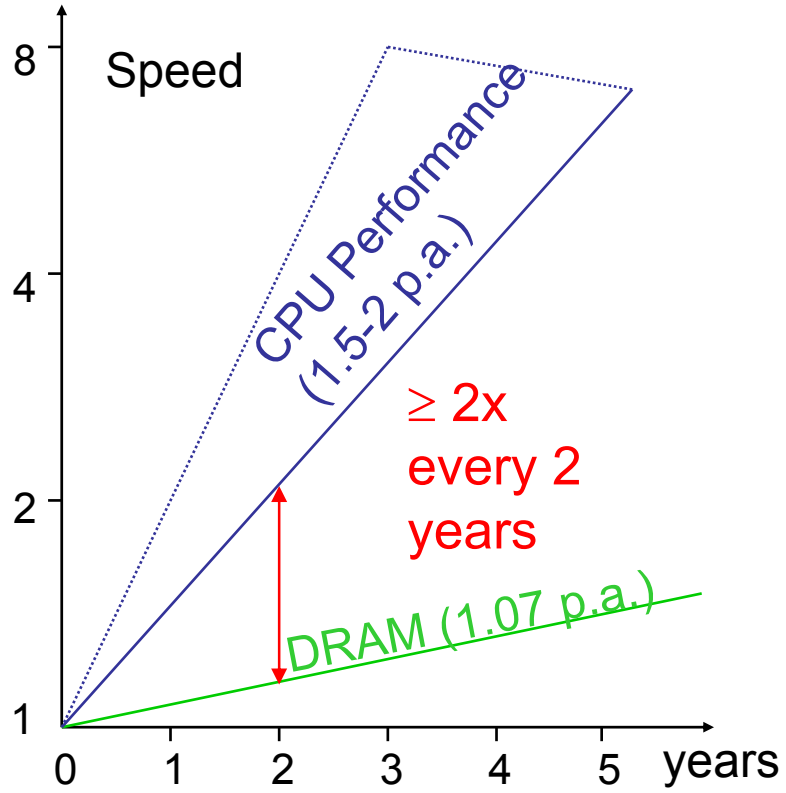
Timing Analysis activity

Activity leader: Björn Lisper

Affiliation: MDH

Timing variations getting larger

Speed gap between processor and main DRAM increases



[P. Machanik: Approaches to Addressing the Memory Wall, TR Nov. 2002, U. Brisbane]

Similar problems also for embedded systems & MPSoCs

☞ In the future:

Memory access times >> processor cycle times

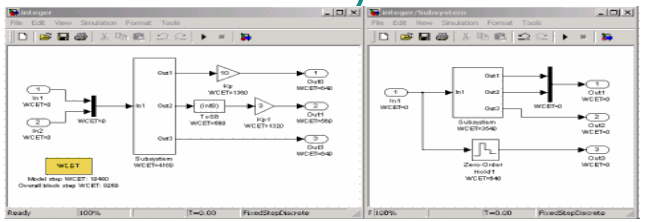
☞ "Memory wall" problem

High-Level Objectives

- To achieve:
  - integration of components for timing analysis of different partners
  - basis: a common representation for an intermediate exchange format
- Synergy between compilation and timing analysis
  - Compilers need TA information to optimize for worst case
  - TA needs information from compilers to deliver precise
    ates

Impacts on Industry

- TA tools are in use in the aeronautics, aerospace, and automotive industries
- TA is relevant for all sectors using Embedded Real-Time Systems
- Industry trends: integrated architectures (IMA, AUTOSAR)
- TA needs to be a part of the development process

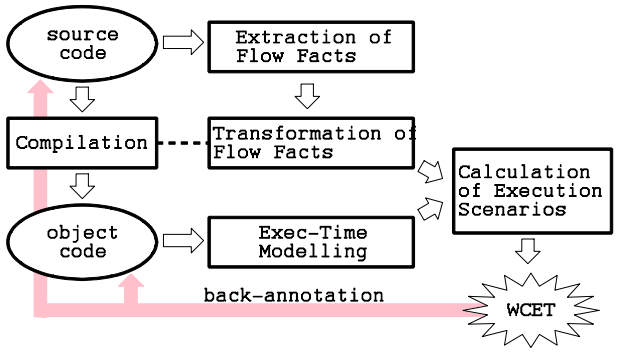


State of the Integration in Europe

- Handling of complex HW architectures, cache replacement strategies (USaar, AbsInt)
- Parametric WCET analysis (MDH, USaar)
- Integration of Bound-T binary decoder with Rapitime WCET tool (York, Tidorum)
- Interchange formats (USaar, AbsInt, MDH, Tidorum, TU Vienna):
  - CRL2 usage experiments
  - AIR format: syntax specification extended with attribute database
  - ALF flow analysis input format: draft specification
  - Conversion AIR to current SWEET input format
  - Analysis of existing languages for user-defined program flow constraints

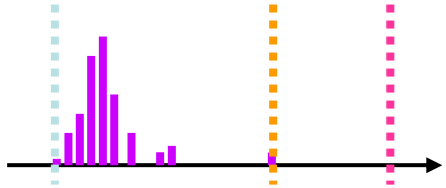
State of the Integration in Europe (2)

- Certification of timing-critical subsystems of the A 380 using aiT
- Evaluation (MDH, USaar, TU Vienna, Tidorum, AbsInt):
  - Common TA benchmark suite arising from WCET Challenge
  - Evaluations of tools on industrial codes
- Compiler-TA integration (TU Vienna, USaar, Dortmund, AbsInt):
  - Transformation of program flow constraints through compiler optimizations
  - Compiler generated cache locking for improved WCET estimation
  - Influence of procedure cloning on WCET prediction



Overall Assessment and Vision at Y0+3

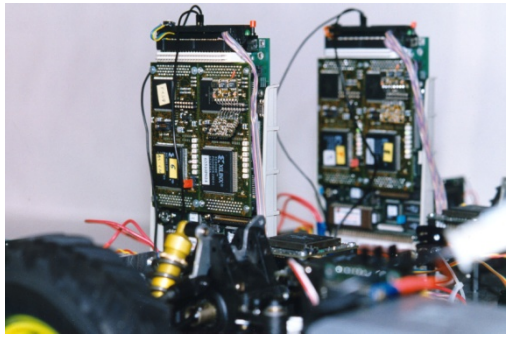
- Well:
  - Europe is still in front
  - Substantial progress TA – compiler integration
  - Interesting developments of TA: handling complex architectures, parametric analysis
  - Cooperation on measurement-based analysis
  - Evaluation: benchmarks, industrial case studies



- Not well:
  - Definitions of interchange and interface formats (AIR, ALF) is taking more time than expected

Plans for Year 4

- Milestone: integration of existing components
- Planned activities Y4:
  - Integration of tool components: finalization of ALF, bridge ALF-AIR
  - Common flow constraint format
  - WCET Tool Challenge 2008
  - Design for Timing Predictability – handling complex HW architectures



  - Measurement-based analysis: framework, and experiments
  - WCET-aware compilation

Scientific Highlights

- **WCET aware compilation:**
  - Bridging two subclusters (not in original proposal)
  - Compiler using Timing Model WCET Analysis
  - Avoiding time-consuming Iterations by the designer
- **Results**
  - Impact of compiler requirements on WCET intermediate languages
  - Reduced WCET's with procedure cloning
  - Reduced WCET's with instruction cache locking
  - Memory allocation with reduced WCET's
  - Transformation of flow facts

Technische Universität

AbsInt
Angewandte Informatik

imec

Information Society

Locking of I-Caches

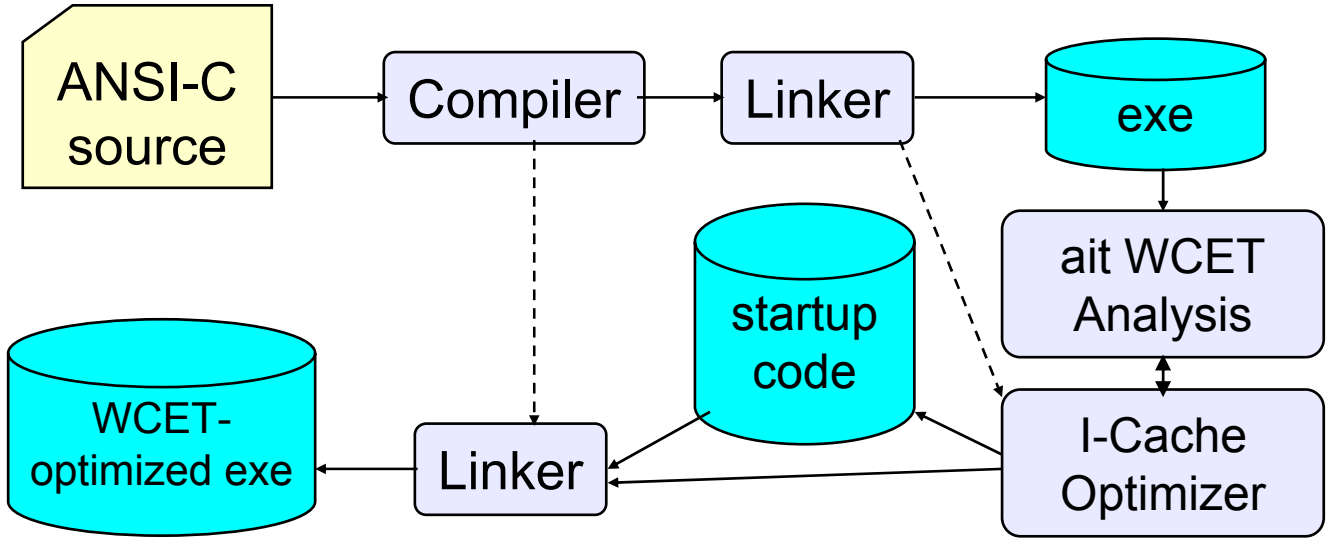
Many caches allow “locking” or “freezing” (no replacements).

Can be used to improve timing predictability:

- Load promising Functions into Cache

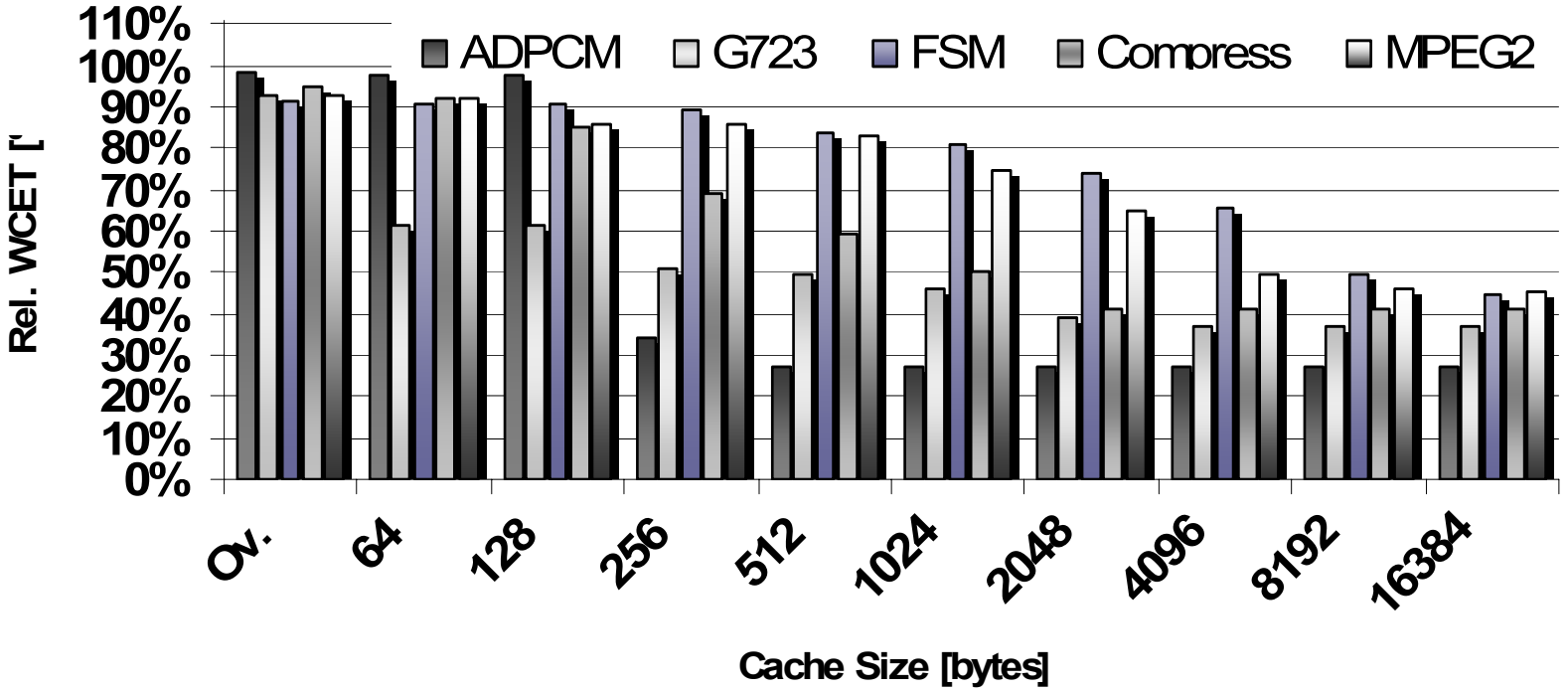
Optimizations:

- Worst case paths can change during optimization
- Requires coupling of timing analysis tool and compiler



[Heiko Falk, Sascha Plazar, Henrik Theiling: Compile-Time Decided Instruction Cache Locking Using Worst-Case Execution Paths, *CODES/ISSS*, 2007]

Example: Relative WCETs after I-Cache Locking



- 100% ≅ HW-Controlled I-Cache
- Monotonous WCET Decreases
- ARM 920T / 16kB: 54.6% - 73.1% WCET Reduction

Procedure Cloning (1)

- Procedure cloning reduces WCET estimates

```
int f(float *x, int n, int p) {
 for (i=1; i<=n; i++) {
  x[i] = pow(x[i], p);
  if (i==10) {...} }
 return x[n]; }

int main(void) {
 return f(a, 5, 2); }
```

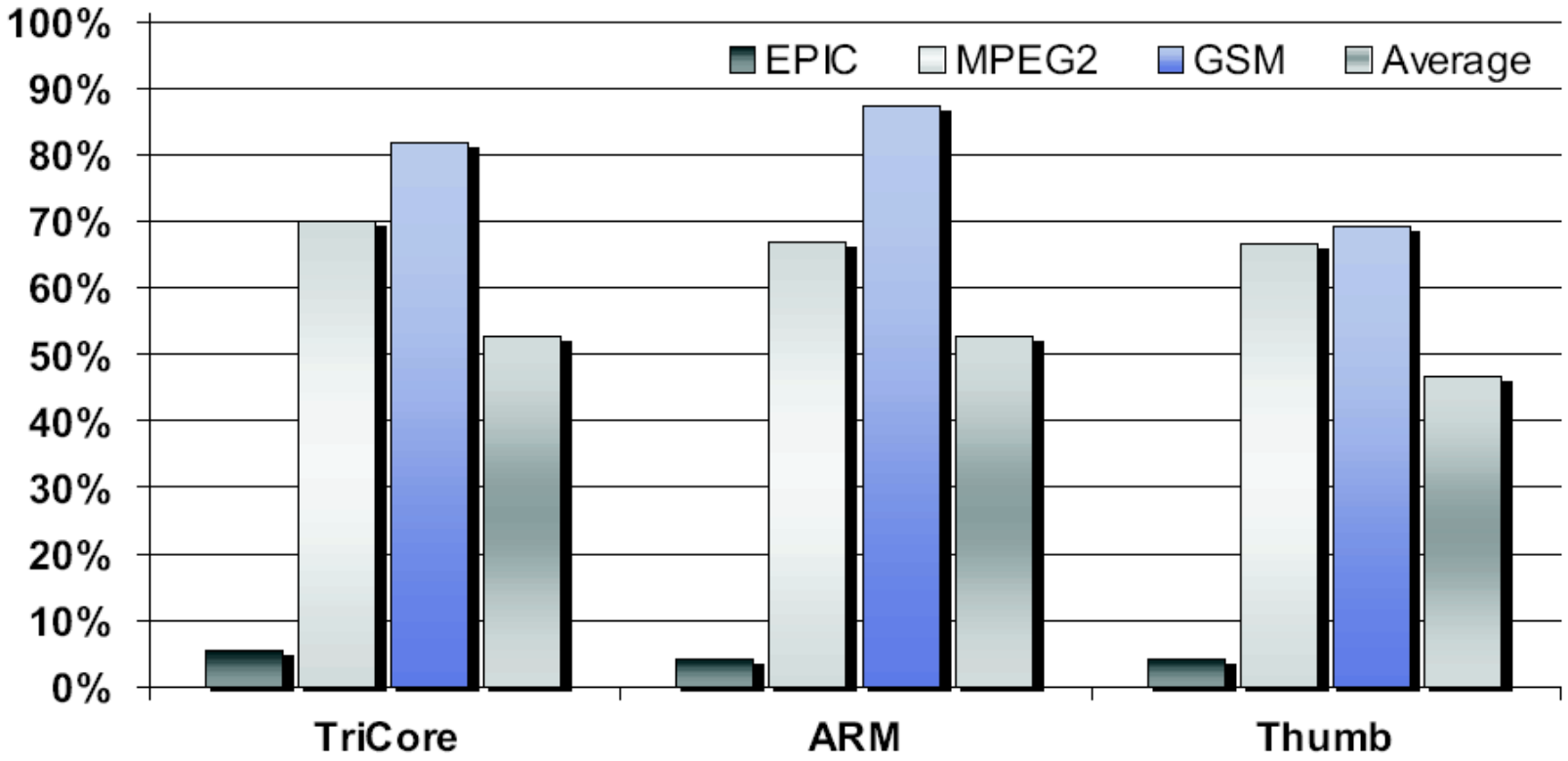
```
int f1(float *x) {
 for (i=1; i<=5; i++)
  x[i] = x[i]*x[i];
 return x[5]; }

int main(void) {
 return f1(a); }
```

Technische Universität

AbsInt
Angewandte Informatik

Information Society

Procedure Cloning (2)



Q&A ?