Institut für Technische Informatik
Chair for Embedded Systems – Prof. Dr. J. Henkel

Vorlesung im SS 2016

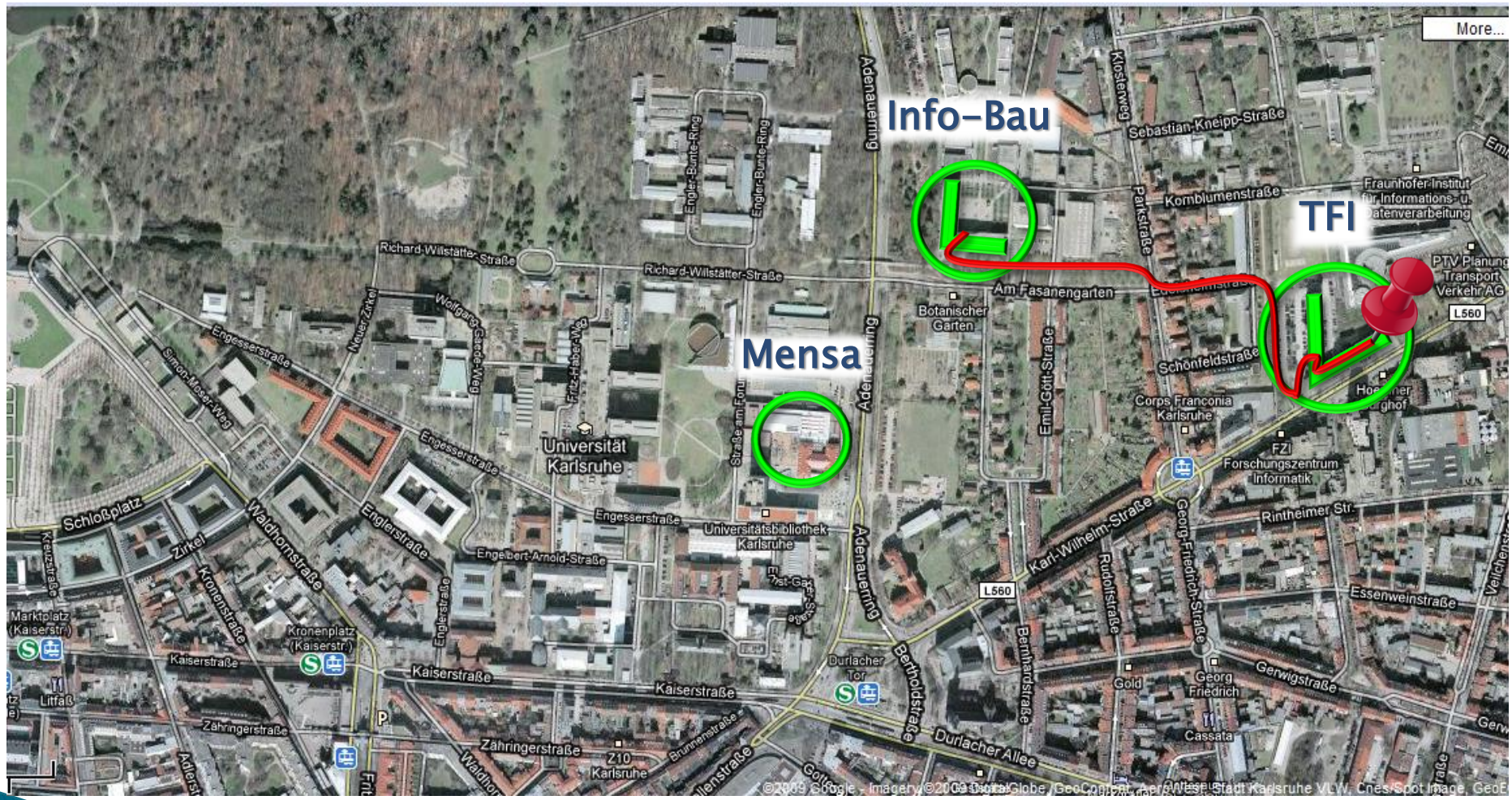
Reconfigurable and Adaptive Systems (RAS)

Marvin Damschen, Lars Bauer, Jörg Henkel

Organisation

- Lecture time: Mi., 15.45 – 17.15
  Bld. 50.34, HS -102
- Homepage: http://ces.itec.kit.edu/1317_1320.php
  You can find the slides from previous years on our old homepage: http://cesweb.itec.kit.edu/teaching/
- Slides Login: Login: “student”
  Passwd: “CES-Student”
- Contact: **marvin.damschen@kit.edu**
  Haid-und-Neu-Str. 7
  Bld. 07.21, Rm. B2-314.4
  (B-Wing, 2nd Floor)

CES @ Technologiefabrik (TFI)



Questions during the lecture

- Simply let me know / interrupt me

RAS Exam

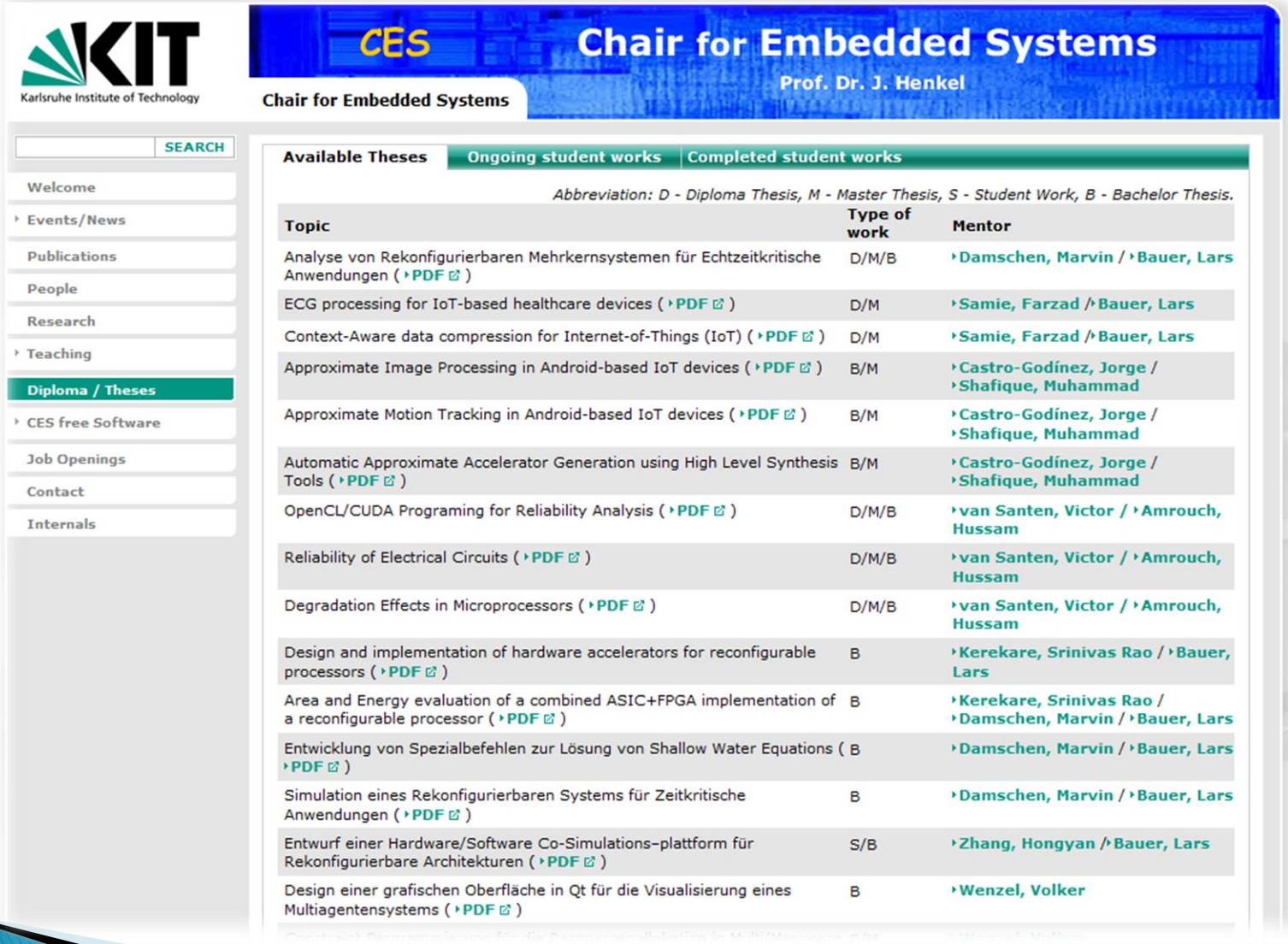
- CS Diploma:
  - Vertiefungsfach 8: Entwurf eingebetteter Systeme und Rechnerarchitekturen
- CS Master:
  - Modul: Rekonfigurierbare und Adaptive Systeme [IN4INRAS] (3 ECTS)
  - Modul: Eingebettete Systeme: Weiterführende Themen [IN4INESWTN] (10 ECTS)
  - Modul: Advanced Computer Architecture [IN4INACA] (10 ECTS)
- Other Study Courses (e.g. EE): ask individually

Teaching @ CES, SS 2016

- Lectures
  - RAS
  - Low Power Design
- Labs
  - Entwurf eingebetteter Systeme
  - Entwurf von eingebetteten applikationsspezifischen Prozessoren
  - Low Power Design and Embedded Systems
- Seminars
  - Rekonfigurierbare Eingebettete Systeme
  - Dependability in Embedded Systems
  - Stereo Video Processing
  - Multicore for Multimedia Processors
  - Low Power Design for Embedded Systems
  - Internet of Things for Healthcare

More Info: http://ces.itec.kit.edu/26.php

Theses @ CES



Karlsruhe Institute of Technology

CES Chair for Embedded Systems

Prof. Dr. J. Henkel

Chair for Embedded Systems

SEARCH

- Welcome
- Events/News
- Publications
- People
- Research
- Teaching
- Diploma / Theses
- CES free Software
- Job Openings
- Contact
- Internals

Available Theses | Ongoing student works | Completed student works

Abbreviation: D - Diploma Thesis, M - Master Thesis, S - Student Work, B - Bachelor Thesis.

Topic	Type of work	Mentor
Analyse von Rekonfigurierbaren Mehrkernsystemen für Echtzeitkritische Anwendungen (PDF)	D/M/B	Damschen, Marvin / Bauer, Lars
ECG processing for IoT-based healthcare devices (PDF)	D/M	Samie, Farzad / Bauer, Lars
Context-Aware data compression for Internet-of-Things (IoT) (PDF)	D/M	Samie, Farzad / Bauer, Lars
Approximate Image Processing in Android-based IoT devices (PDF)	B/M	Castro-Godínez, Jorge / Shafique, Muhammad
Approximate Motion Tracking in Android-based IoT devices (PDF)	B/M	Castro-Godínez, Jorge / Shafique, Muhammad
Automatic Approximate Accelerator Generation using High Level Synthesis Tools (PDF)	B/M	Castro-Godínez, Jorge / Shafique, Muhammad
OpenCL/CUDA Programing for Reliability Analysis (PDF)	D/M/B	van Santen, Victor / Amrouch, Hussam
Reliability of Electrical Circuits (PDF)	D/M/B	van Santen, Victor / Amrouch, Hussam
Degradation Effects in Microprocessors (PDF)	D/M/B	van Santen, Victor / Amrouch, Hussam
Design and implementation of hardware accelerators for reconfigurable processors (PDF)	B	Kerekare, Srinivas Rao / Bauer, Lars
Area and Energy evaluation of a combined ASIC+FPGA implementation of a reconfigurable processor (PDF)	B	Kerekare, Srinivas Rao / Damschen, Marvin / Bauer, Lars
Entwicklung von Spezialbefehlen zur Lösung von Shallow Water Equations (PDF)	B	Damschen, Marvin / Bauer, Lars
Simulation eines Rekonfigurierbaren Systems für Zeitkritische Anwendungen (PDF)	B	Damschen, Marvin / Bauer, Lars
Entwurf einer Hardware/Software Co-Simulations-plattform für Rekonfigurierbare Architekturen (PDF)	S/B	Zhang, Hongyan / Bauer, Lars
Design einer grafischen Oberfläche in Qt für die Visualisierung eines Multiagentensystems (PDF)	B	Wenzel, Volker

Theses @ CES

- http://ces.itec.kit.edu/69.php
- **Note:** Info on homepage is typically not up-to-date
  - If you are interested in a particular topic: better ask individually
- There are often SADABAMA theses or Hiwi jobs available in the scope of reconfigurable systems
- Main projects:
  - *i*-Core: invasive Core
  - Real-time: Analysis and design of predictable reconfigurable architectures
  - OTERA: Online Test Strategies for Reliable Reconfigurable Architectures
- Topics:
  - Algorithms for Runtime System, Operating System, …
  - Static Program Analysis, Toolchain, Compiler, Synthesis, …
  - Architecture, Hardware Prototype, Simulation Environment, …

Beneficial Previous Knowledge

- Rechnerstrukturen
  - Prerequisites
- Eingebettete Systeme
  - ES1: Optimierung und Synthese Eingebetteter Systeme
  - ES2: Entwurf und Architekturen für Eingebettete Systeme
  - The core topics (e.g. details about FPGA architectures) will be recapitulated in the scope of this lecture
  - Thus, the contents of ES1 and ES2 are beneficial but not required in full detail

General Literature

- “Fine- and Coarse-Grain Reconfigurable Computing”, S. Vassiliadis and D. Soudris, Springer 2007.
- “Runtime adaptive extensible embedded processors – a survey”, H. P. Huynh and T. Mitra, SAMOS, pp. 215–225, 2009.
- “Reconfigurable computing: architectures and design methods”, T.J. Todman et al., IEE Proceedings Computers & Digital Techniques, vol. 152, no. 2, pp. 193–207, 2005.
- “Reconfigurable Instruction Set Processors from a Hardware/Software Perspective”, F. Barat et al., IEEE Transactions on Software Engineering, vol. 28, no. 9, pp. 847–862, 2002.

Karlsruhe Institute of Technology
Institut für Technische Informatik
Chair for Embedded Systems – Prof. Dr. J. Henkel

Reconfigurable and Adaptive Systems (RAS)

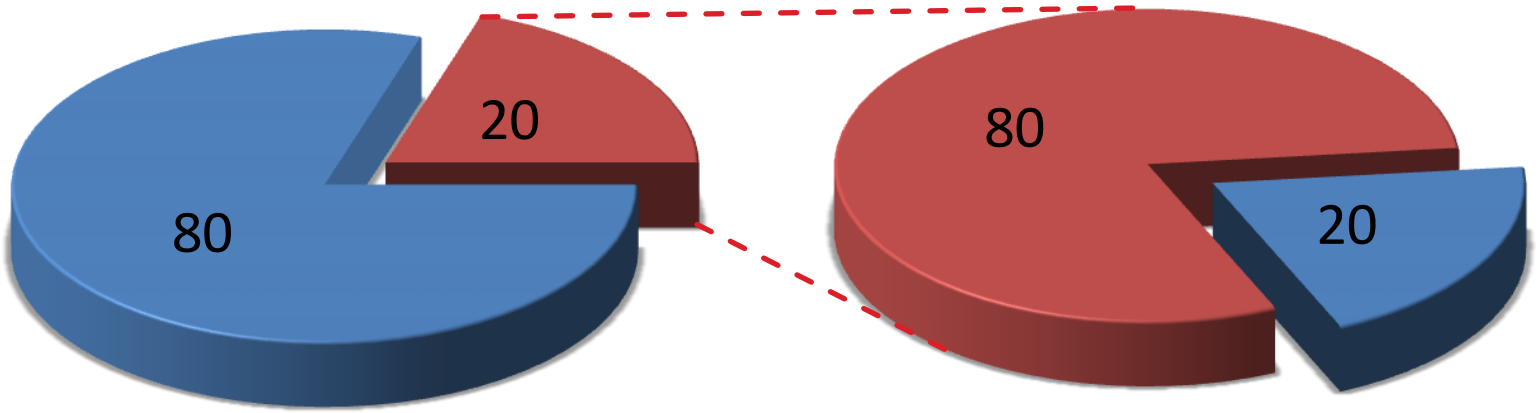
1. Introduction and Motivation: The Demand for Adaptivity

Designing Embedded Systems

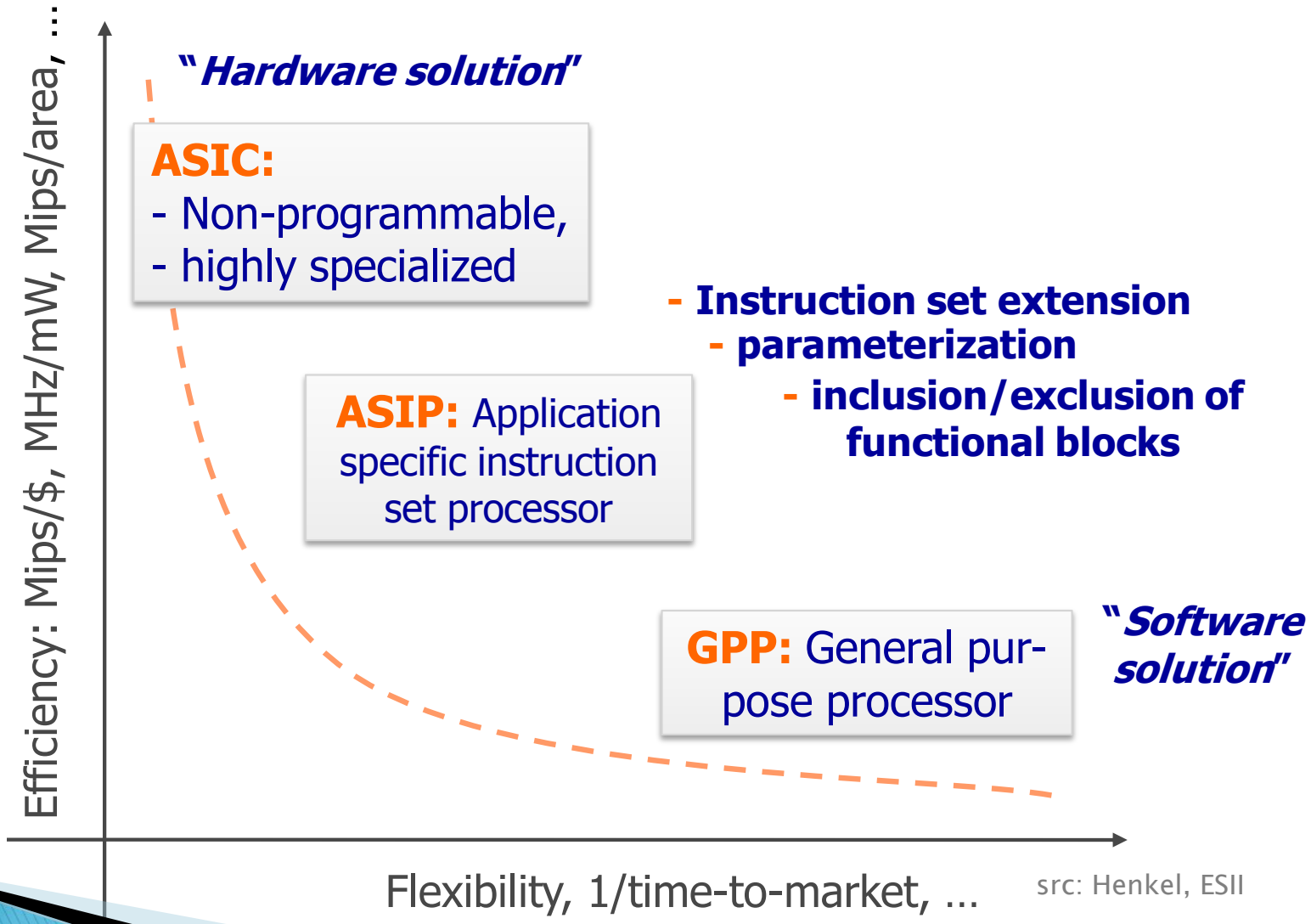
- Typical approach:
  - Static analysis of system requirements (e.g. com-putational hot spots)
  - Build optimized system
- Today's requirements:
  - More functionality
  - Increasing complexity
  - Non-functional constraints
- **Problem:**
  - Statically chosen design point has to match all requirements
  - Typically inefficient for individual components (e.g. tasks or hot spots)

Definition 'Computational Hot Spot'

- A rather small part of the application that corresponds to a rather large part of the execution time
  - Also called 'Computational Kernel'
  - Typically: inner loop
  - 80/20 rule (90/10 rule etc.)



Typical Implementation Alternatives

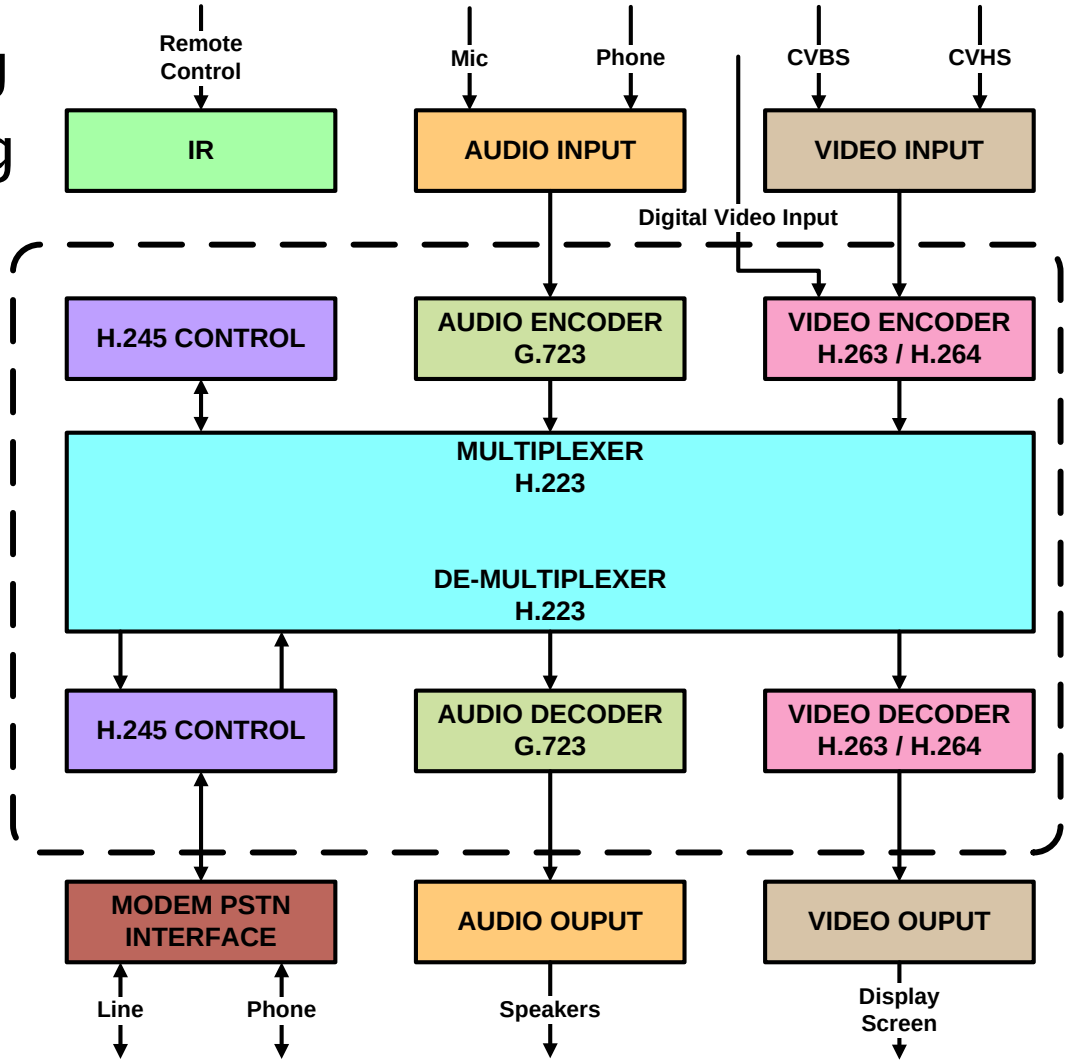


src: Henkel, ESII

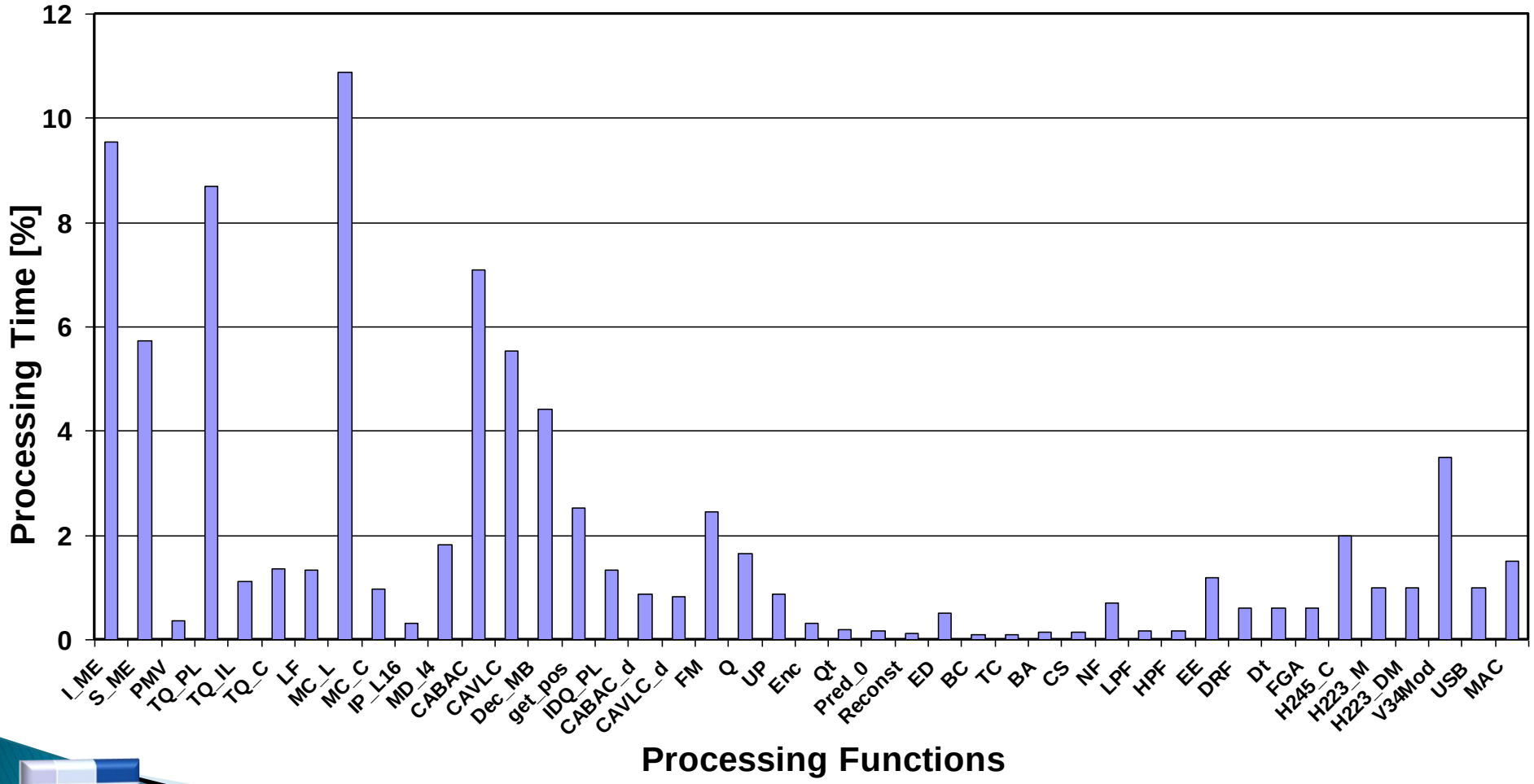
First Example: H.324 Video Conferencing

- Video En-/Decoding
- Audio En-/Decoding
- Data (De-)Multi-plexing
- Control protocol

src: cityrockz.com

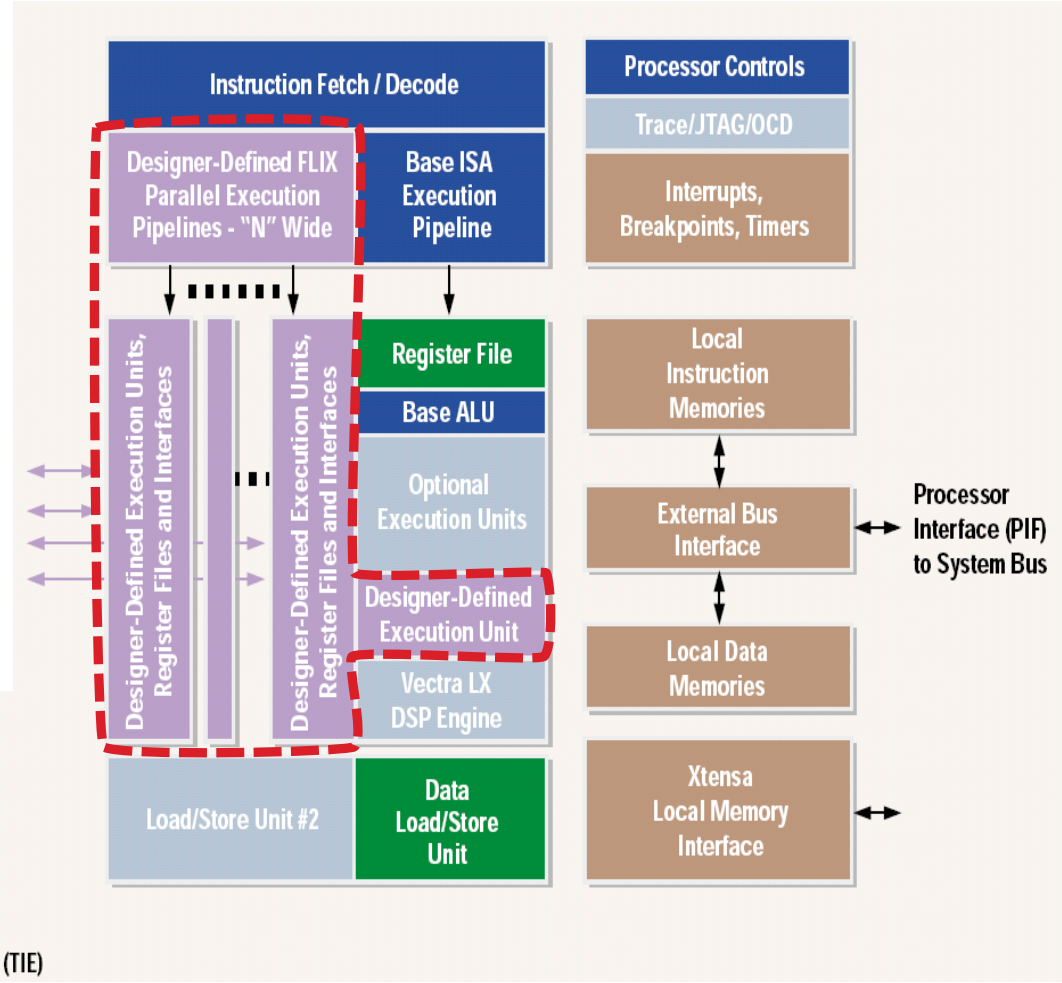


Hotspots in H.324 Video Conferencing



ASIP Implementation

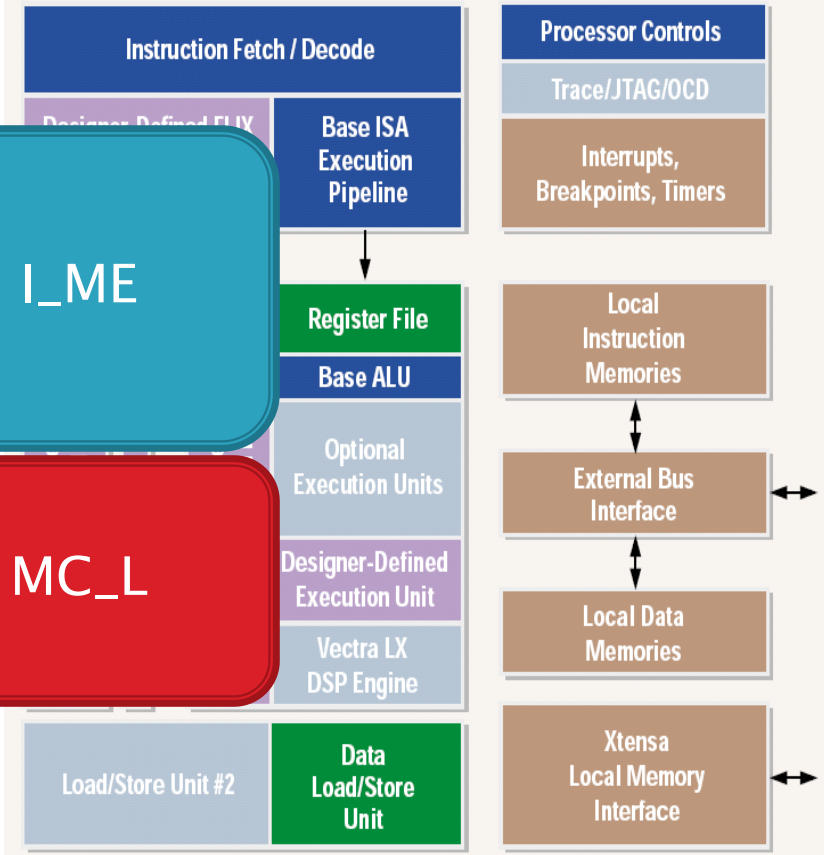
- Design accele-rators for the hot spots
- Connect them as Execution Units, Register Files, and Interfaces



src: Tensilica, Inc.: "Xtensa LC Product Brief"

ASIP Implementation (cont'd)

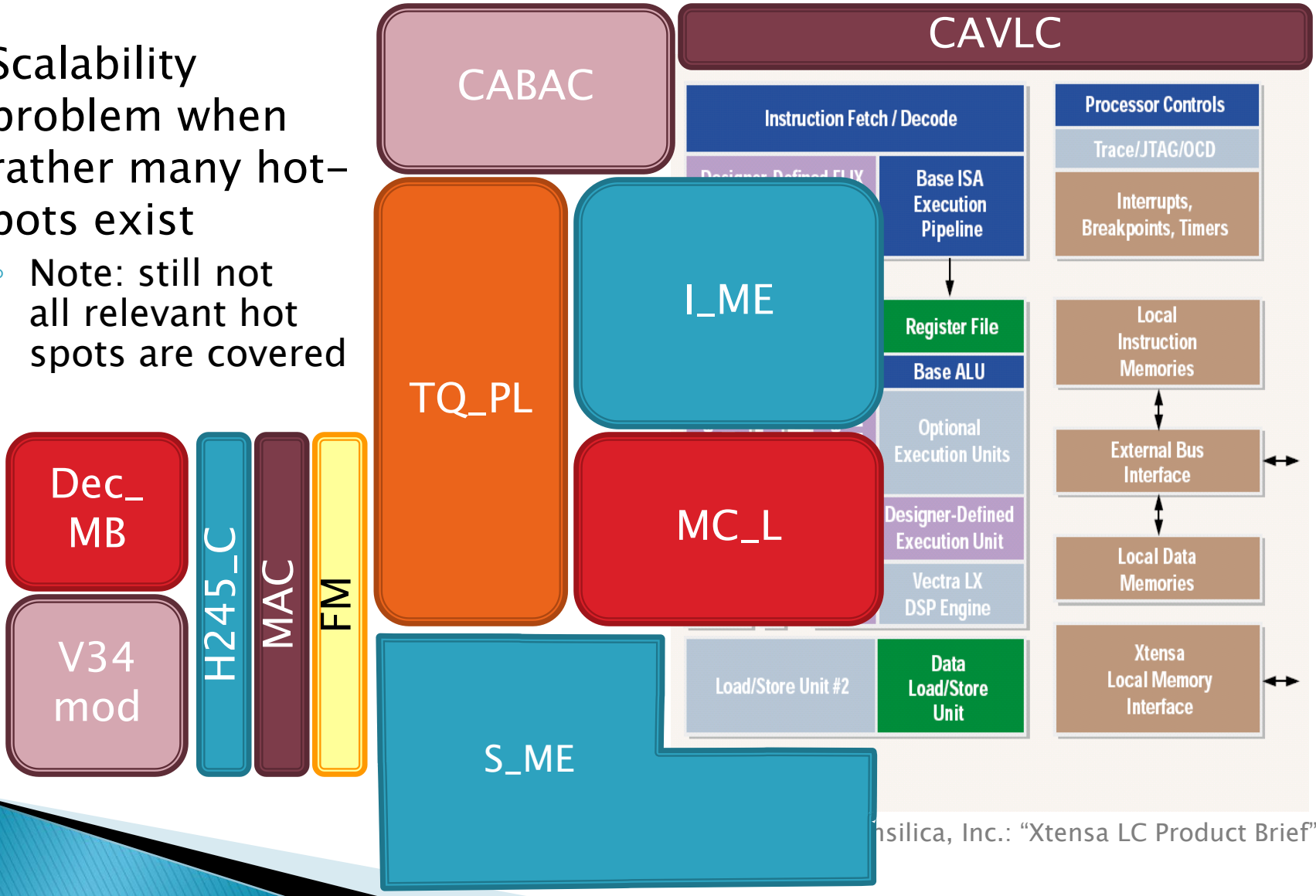
- Provides noticeably improved performance after targeting the major hot spots
- However, performance still not sufficient to achieve real-time requirements
  - More hot spots need to be accelerated



src: Tensilica, Inc.: "Xtensa LC Product Brief"

ASIP Implementation (cont'd)

- Scalability problem when rather many hot-pots exist
  - Note: still not all relevant hot spots are covered



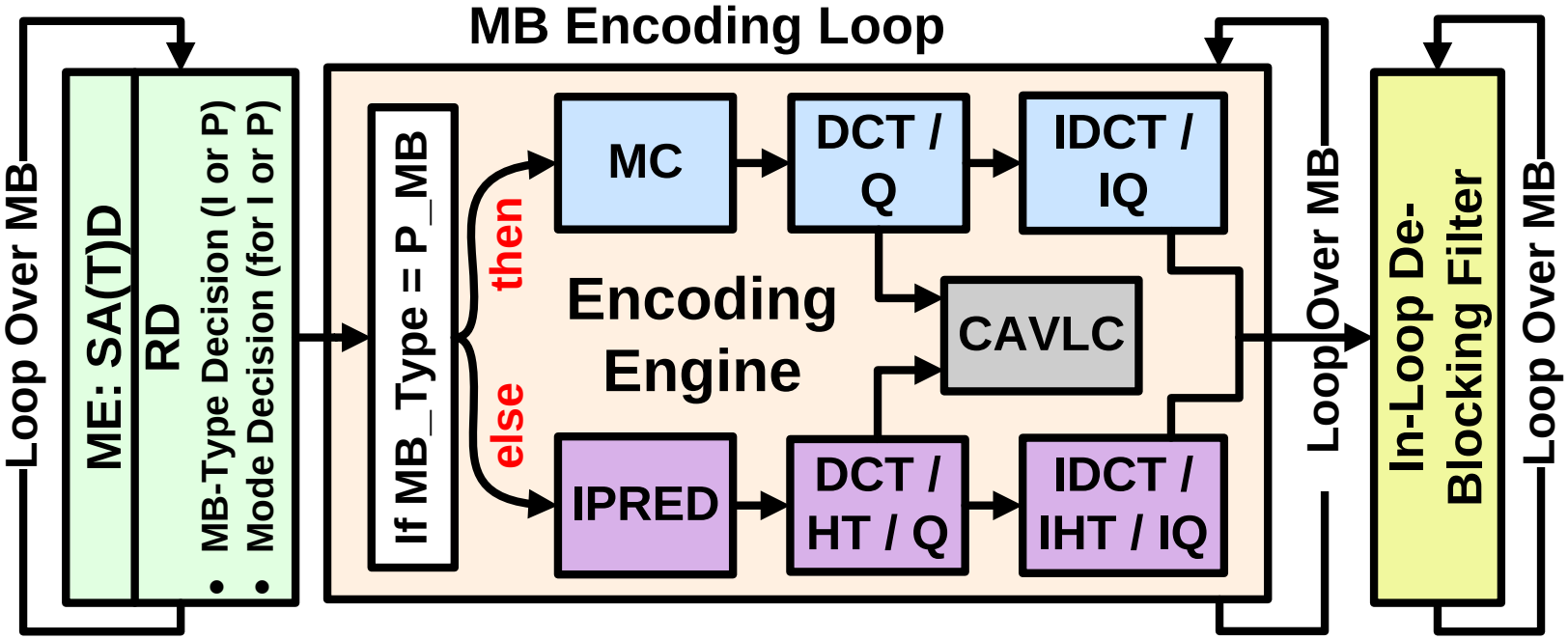
nsilica, Inc.: "Xtensa LC Product Brief"

Summary of ASIP Implementation

- ASIPs perform well when
  1. rather few hot spots need to be accelerated and
  2. those hot spots are well known in advance
- ASIPs are less efficient when targeting many hot spots or unknown hot spots
  - All accelerators are provided statically (i.e. they require area and consume power) even though typically just a few of them are needed at a certain time
  - Even for a given application it may not necessarily be clear, which parts are 'hot' during execution as this may depend on input data (as demonstrated in the following)

➔ In such situations a different architecture might be preferable

Second Example: H.264 video Encoder



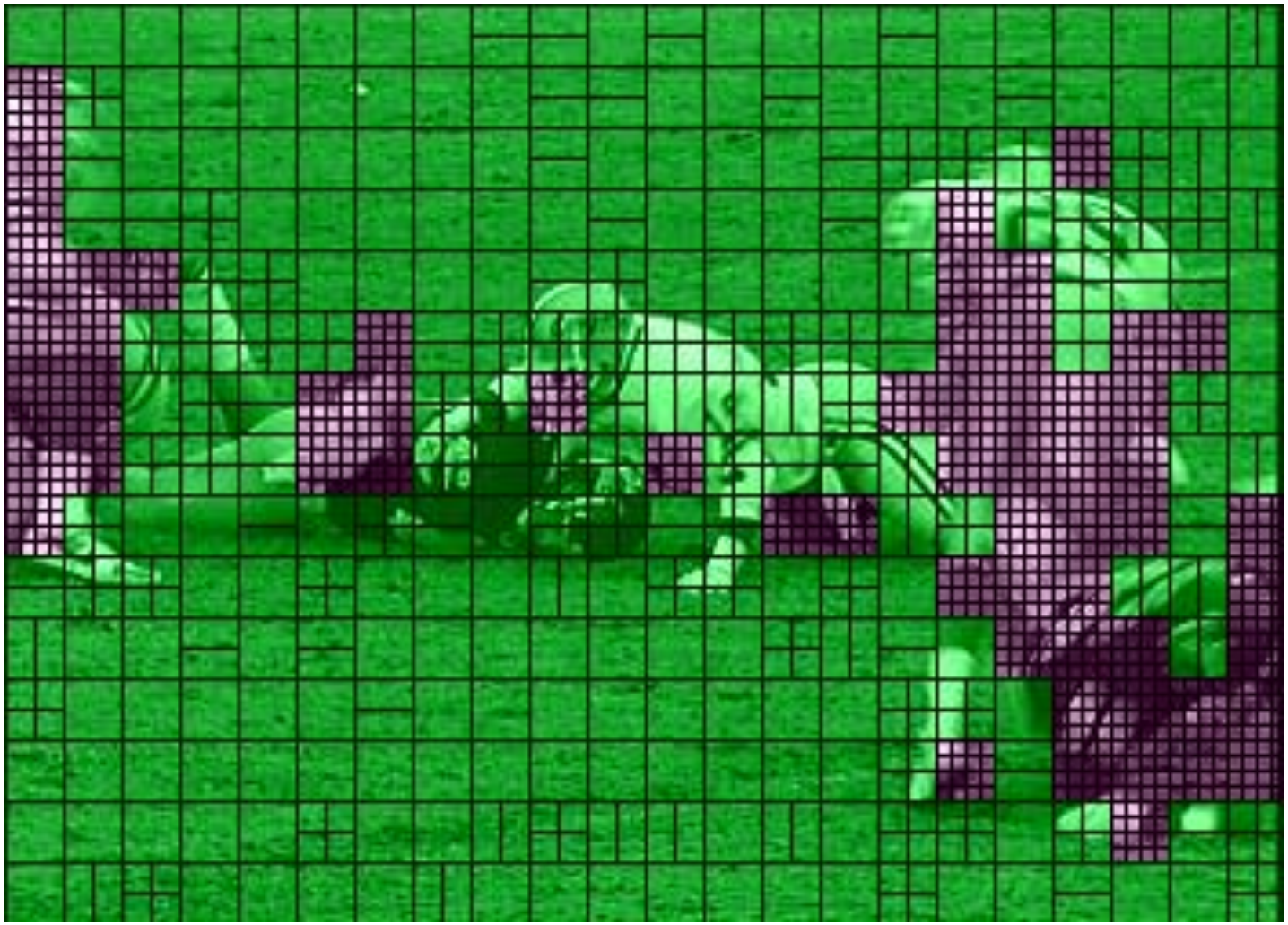
- Iterates on MacroBlocks (**MBs**, i.e. 16x16 pixels)
- 2 different MB-types
  → different computational paths with different computational requirements
  - I-MB (spatial prediction)
  - P-MB (temporal prediction)

Example: Football Video

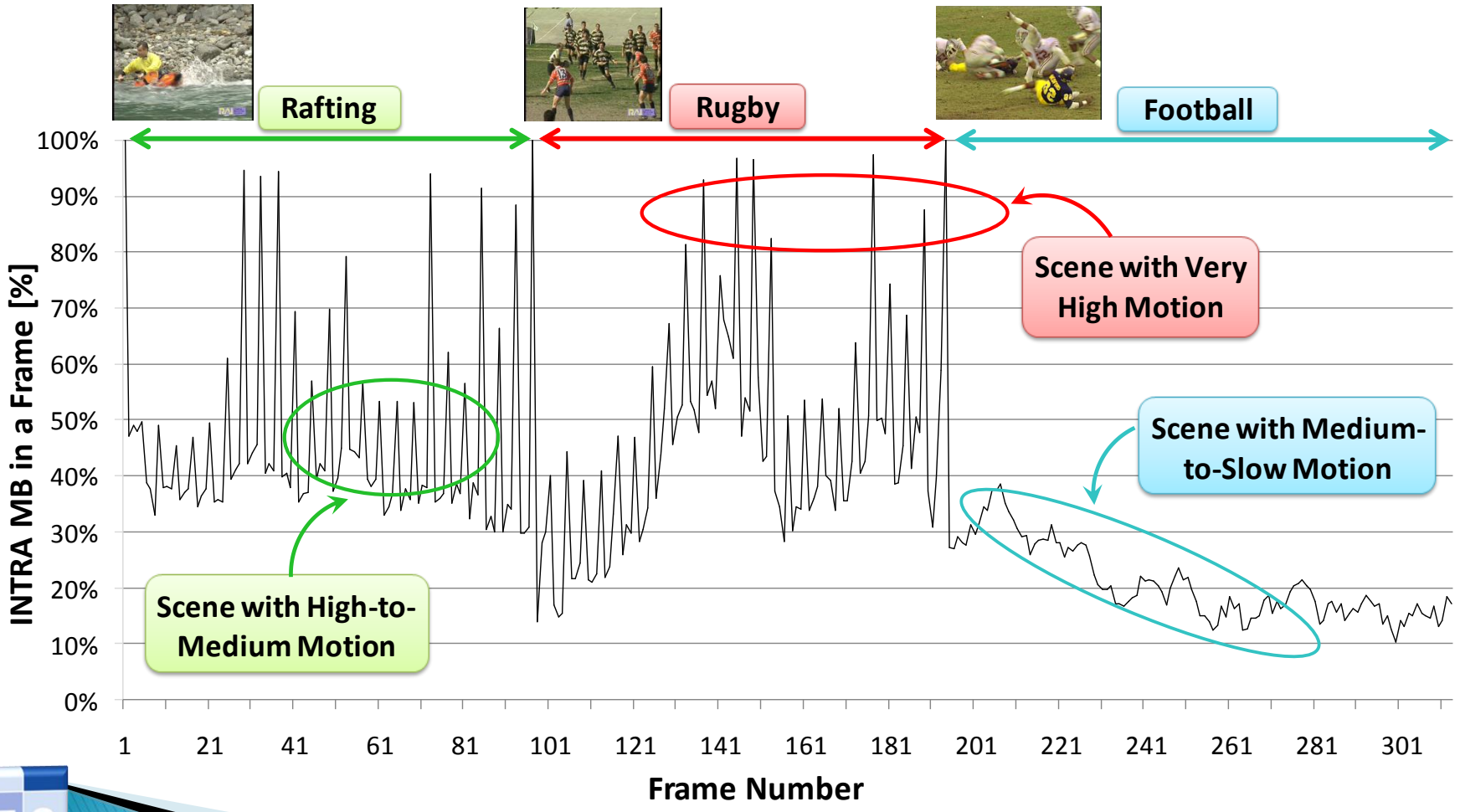
I-MB

P-MB

Note: 16x16 MBs can be partitioned into sub-MBs, e.g. 16x8, 8x8, down to 4x4



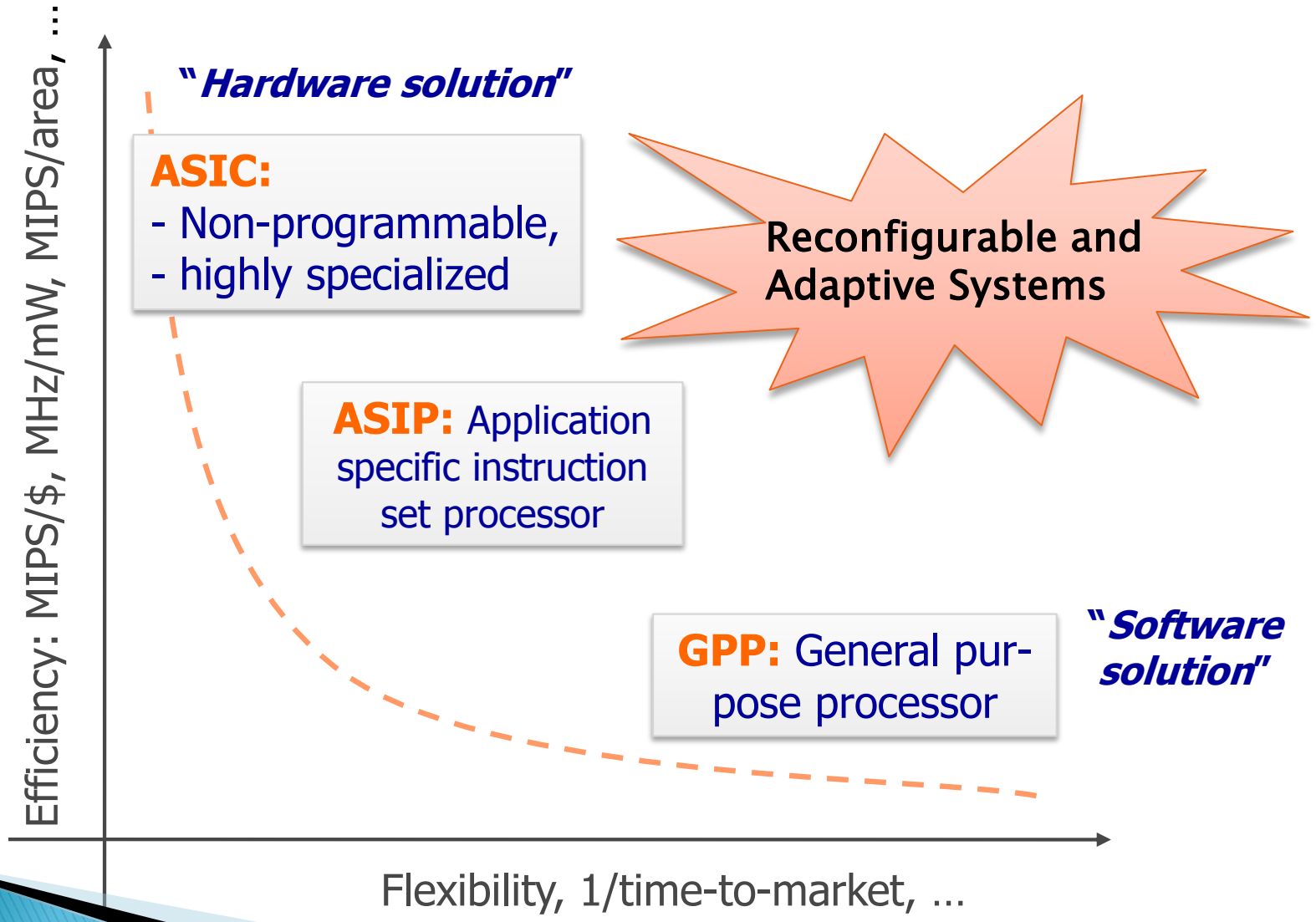
Example: Distribution of I-MBs in Medium-to-VeryHigh Motions



Conclusion: Demand for Adaptivity

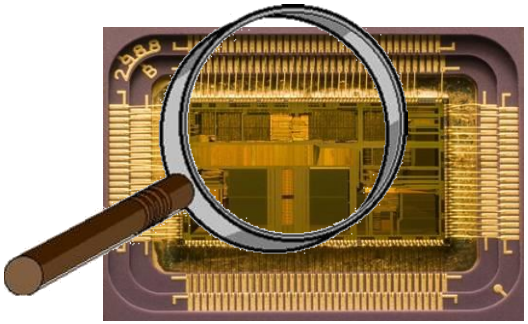
- Even for a well known application it is not always clear which parts will be 'hot' (e.g. according computational complexity) and thus benefit from accelerators
  - This depends on changing input data and control flow
- Even more complex: multi-tasking scenarios
  - Not clear, which applications will execute at the same time
  - Not clear, which applications will execute at all (user can download new applications)
  - This significantly increases the number of potential hot spots → hardly possible to address this with an ASIP
- Systems that fulfill the demand for adaptivity may lead to
  - Better performance (absolute criteria)
  - Higher Efficiency (relative criteria e.g. performance per area etc.)
  - Lower cost (no redesign if specifications change, no overdesign to cover all scenarios)

Potentials of RAS



Potentials of RAS (cont'd)

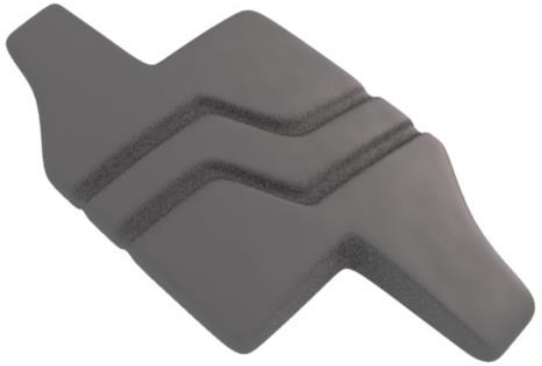
- Providing accelerators for hot spots on demand
- Efficient dependability/reliability and fault tolerance
  - Rather than providing static redundancy or hardened devices, use online monitoring (BIST: Build-in Self-Test) to detect faults and use reconfiguration and adaptation to react accordingly

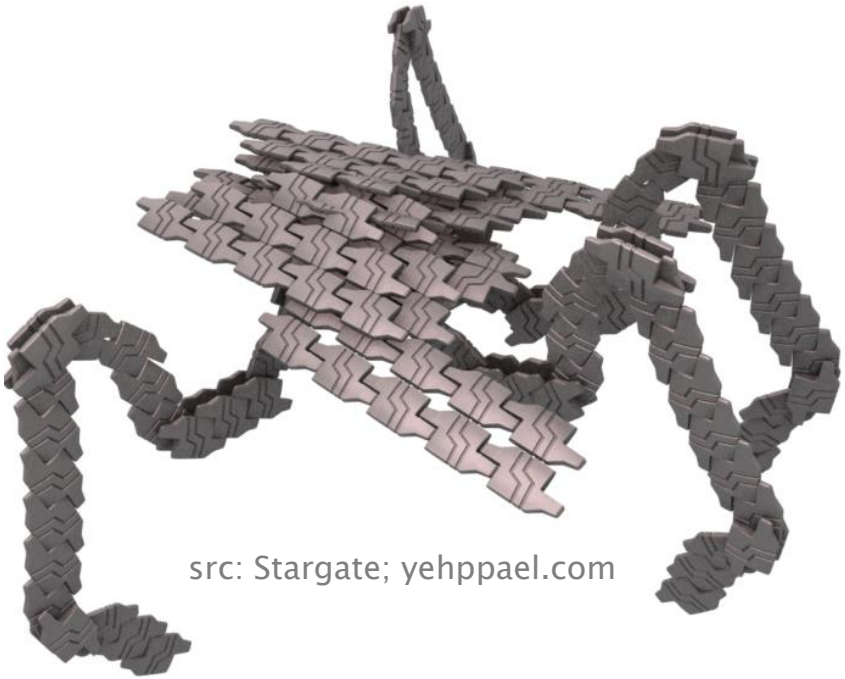


- Reducing the design/development costs
  - Hardware bug fixes, hardware updates
  - Avoids hardware redesign
- Shorter Time-to-market
  - The time between idea and product
- Improved efficiency
  - E.g. energy reduction due to better resource utilization
- So-called 'Self-x' properties (explained in the following)

Self-organisation/Self-configuration

- The ability to determine and establish feasible/ good setups
  - Composed out of predetermined elements
  - Or created from scratch (online-synthesis)
  - Or implicitly created (emergent behavior)

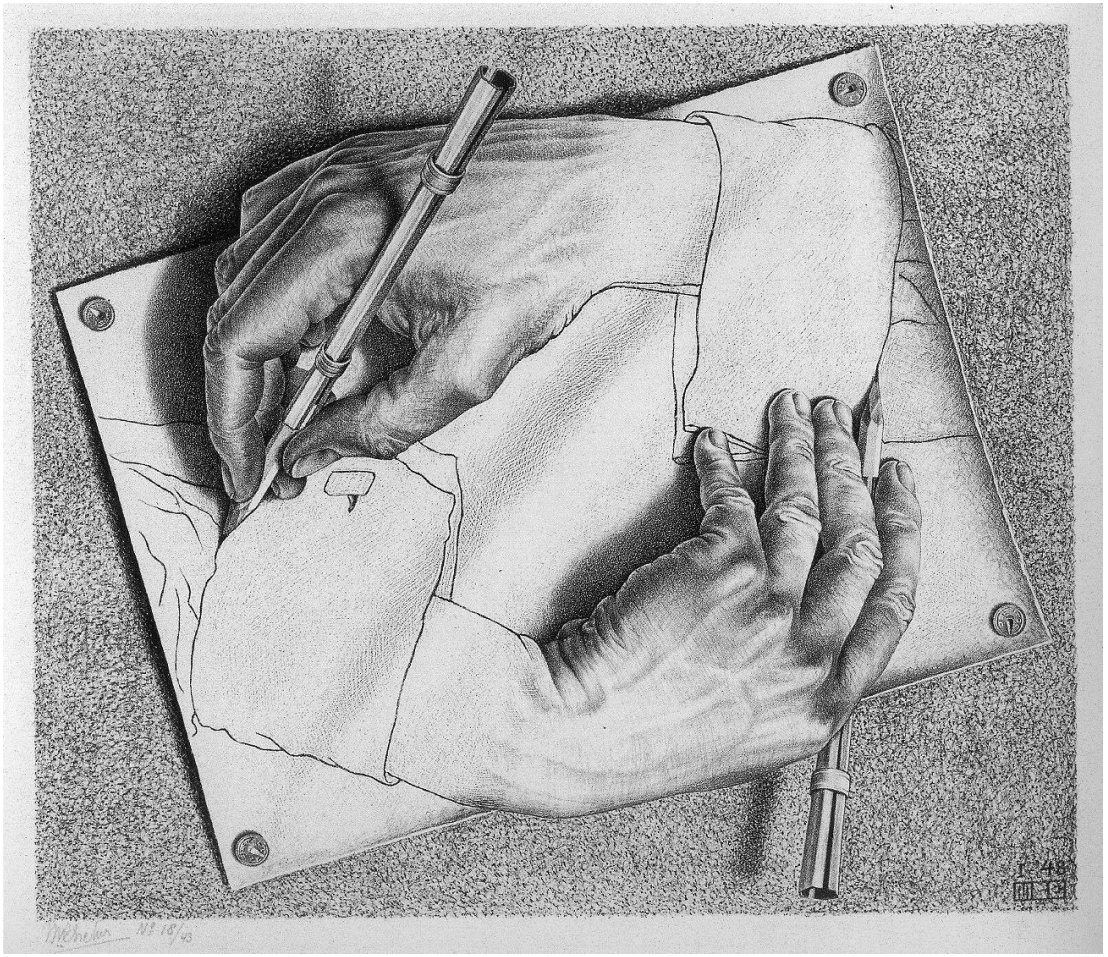




src: Stargate; yehppael.com

Self-adaptation/Self-optimization

- The ability to modify/improve the system setup towards maximizing a certain cost function (e.g. performance, energy saving, or efficiency)
- The cost function is not necessarily fixed, but it may vary, depending on external requirements, goals etc.



src: M. C. Escher

Self-healing

- The ability to resist, tolerate, or correct certain faults
- It is not necessarily required to explicitly detect them
- It is not necessarily required to operate with the same performance, efficiency etc. as before the fault
  - Graceful degradation

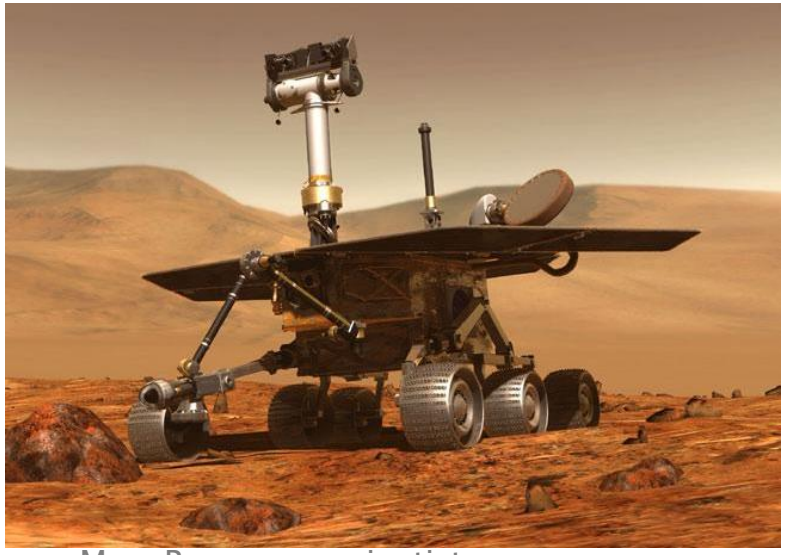
src: T-800; spill.com

src: T-1000; geekologie.com

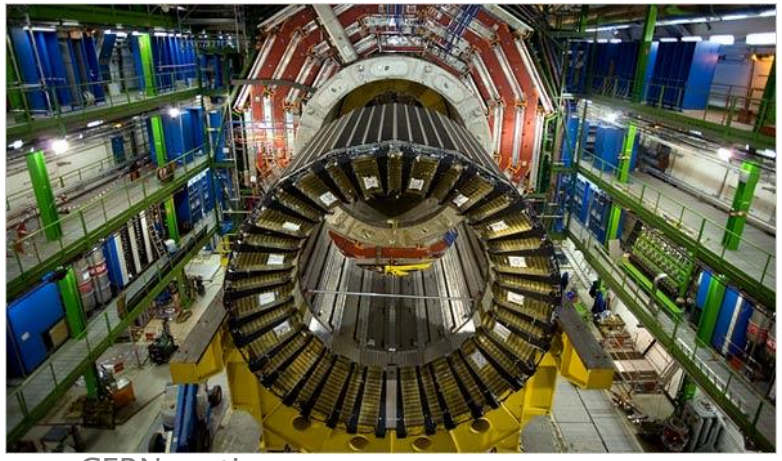
src: T-1000; movie-infos.net

Sneak Preview

- Techniques for (Self-) Reconf.
  - How to use/develop/reconfigure accelerators
  - Optimizations (compile time/run time)
- Different flavors of reconfigurable processors
  - Basic systems
  - Highly efficient/adaptive systems
  - Online synthesis
- New Technologies for reconfi-gurable devices and innovative products
- Improving system reliability by reconfiguration



src: Mars Rover, newscientist.com



src: CERN, nytimes.com