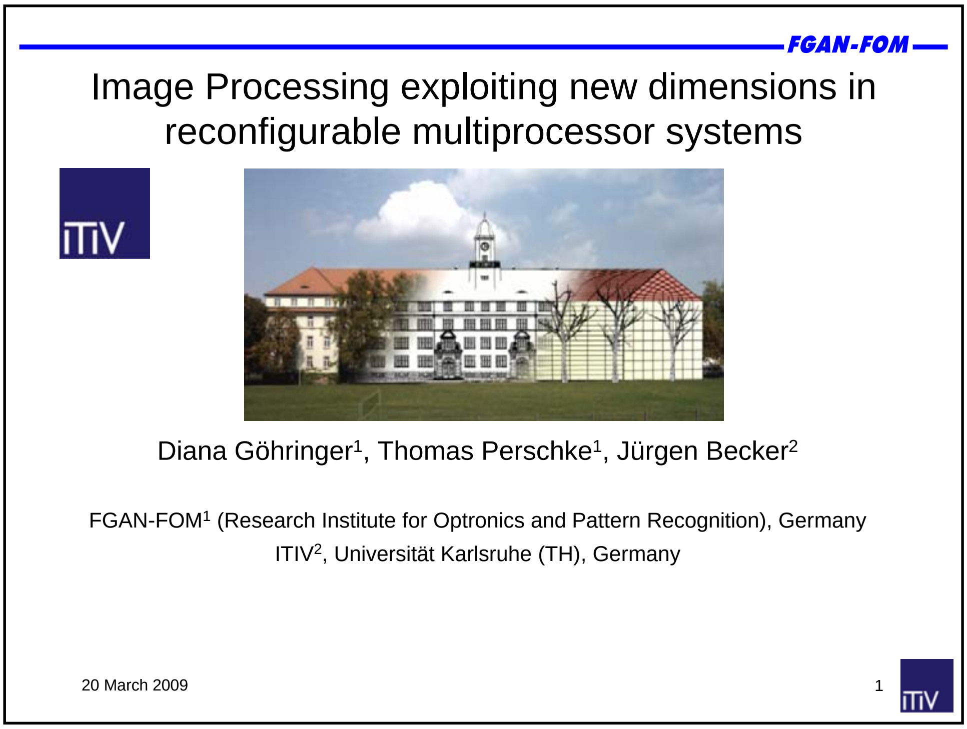
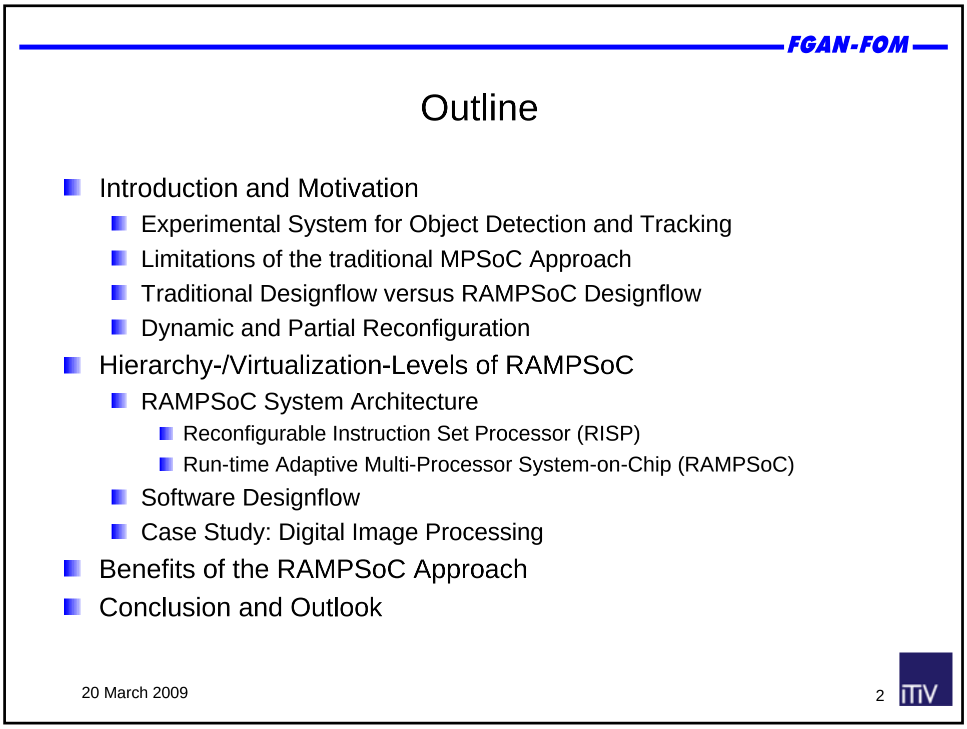


FGAN-FOM

Image Processing exploiting new dimensions in reconfigurable multiprocessor systems

Diana Göhringer[1], Thomas Perschke[1], Jürgen Becker[2]

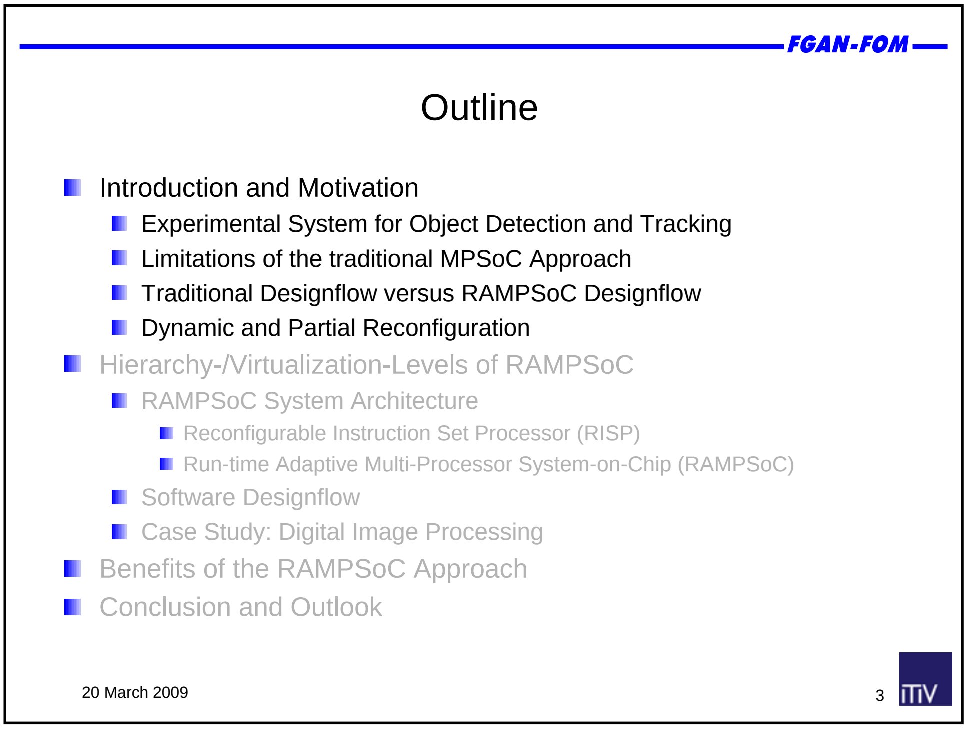
FGAN-FOM[1] (Research Institute for Optronics and Pattern Recognition), Germany
ITIV[2], Universität Karlsruhe (TH), Germany



FGAN-FOM

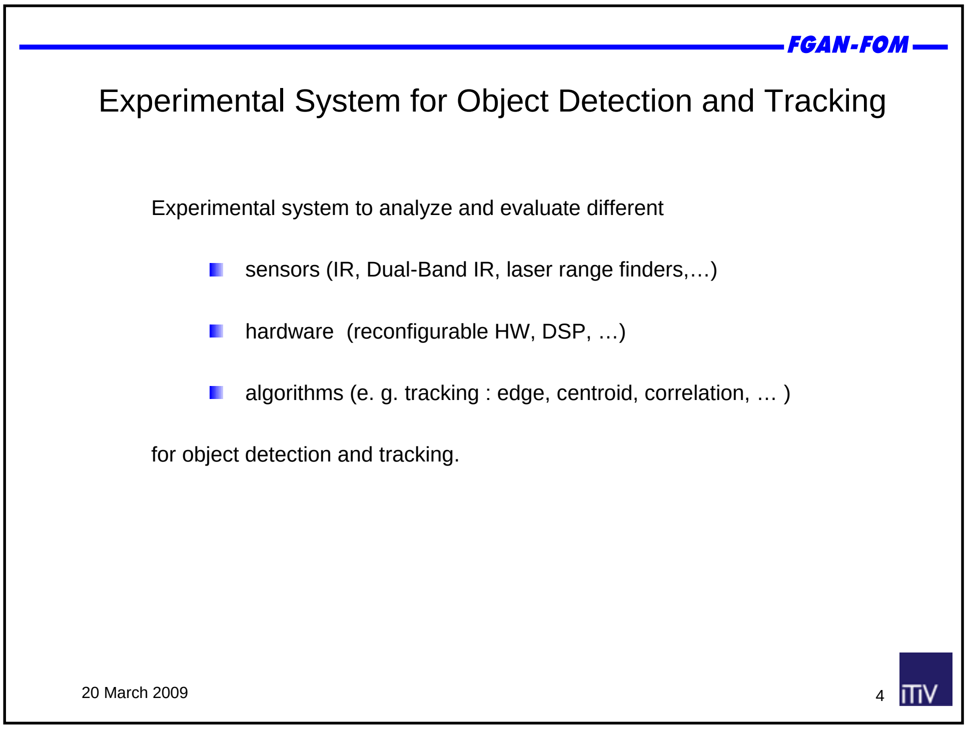
Outline

- Introduction and Motivation
  - Experimental System for Object Detection and Tracking
  - Limitations of the traditional MPSoC Approach
  - Traditional Designflow versus RAMPSoC Designflow
  - Dynamic and Partial Reconfiguration
- Hierarchy-/Virtualization-Levels of RAMPSoC
  - RAMPSoC System Architecture
    - Reconfigurable Instruction Set Processor (RISP)
    - Run-time Adaptive Multi-Processor System-on-Chip (RAMPSoC)
  - Software Designflow
  - Case Study: Digital Image Processing
- Benefits of the RAMPSoC Approach
- Conclusion and Outlook



Outline

- Introduction and Motivation
  - Experimental System for Object Detection and Tracking
  - Limitations of the traditional MPSoC Approach
  - Traditional Designflow versus RAMPSoC Designflow
  - Dynamic and Partial Reconfiguration
- Hierarchy-/Virtualization-Levels of RAMPSoC
  - RAMPSoC System Architecture
    - Reconfigurable Instruction Set Processor (RISP)
    - Run-time Adaptive Multi-Processor System-on-Chip (RAMPSoC)
  - Software Designflow
  - Case Study: Digital Image Processing
- Benefits of the RAMPSoC Approach
- Conclusion and Outlook

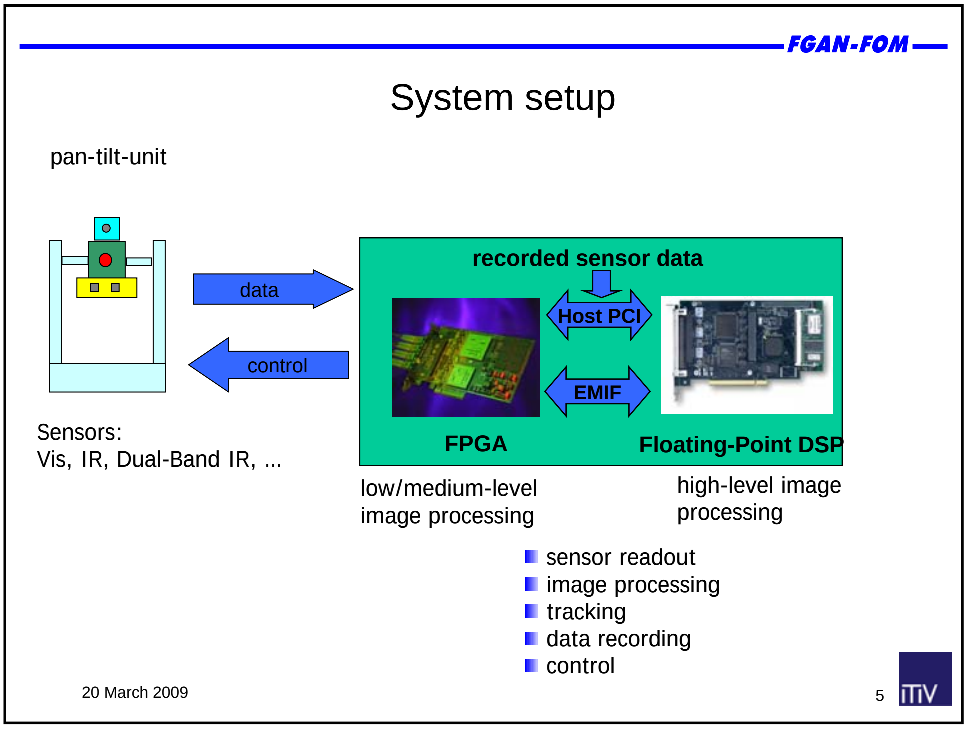


Experimental System for Object Detection and Tracking

Experimental system to analyze and evaluate different

- sensors (IR, Dual-Band IR, laser range finders,…)
- hardware (reconfigurable HW, DSP, …)
- algorithms (e. g. tracking : edge, centroid, correlation, …)

for object detection and tracking.



System setup

pan-tilt-unit

data

control

recorded sensor data

Host PCI

EMIF

FPGA

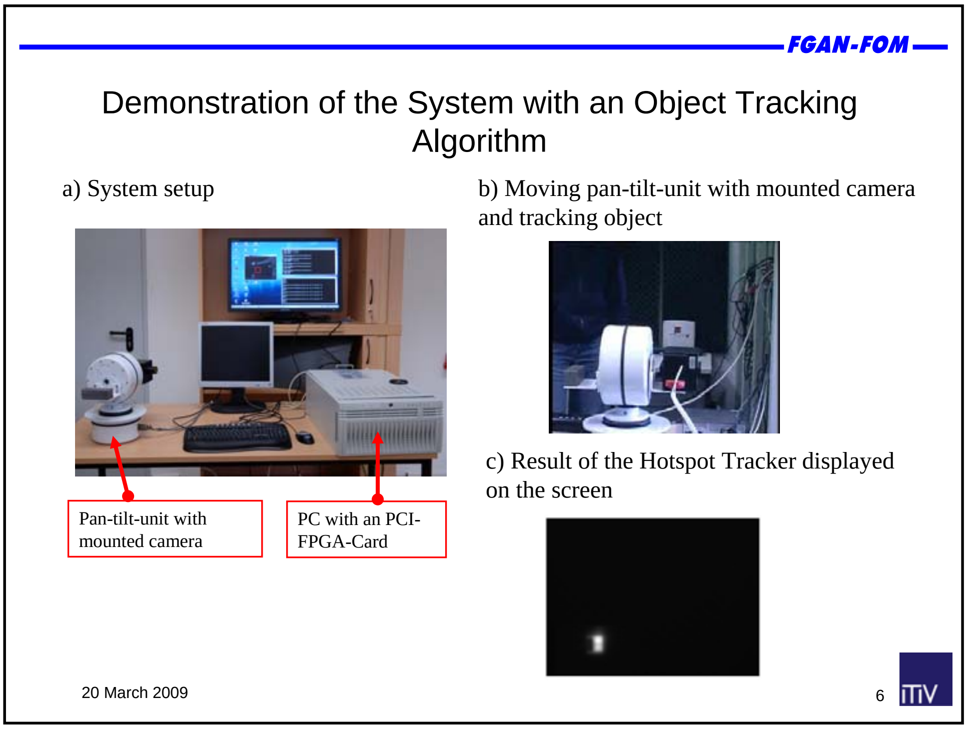
Floating-Point DSP

Sensors:
Vis, IR, Dual-Band IR, ...

low/medium-level image processing

high-level image processing

- sensor readout
- image processing
- tracking
- data recording
- control



Demonstration of the System with an Object Tracking Algorithm

a) System setup

Pan-tilt-unit with mounted camera

PC with an PCI-FPGA-Card

b) Moving pan-tilt-unit with mounted camera and tracking object

c) Result of the Hotspot Tracker displayed on the screen

Challenge

Experimental system „failed“ with respect to serve as an

easy to use rapid prototyping system

Why?

The implementation of complex software tracking algorithms on the FPGA needs a lot of expertise and time

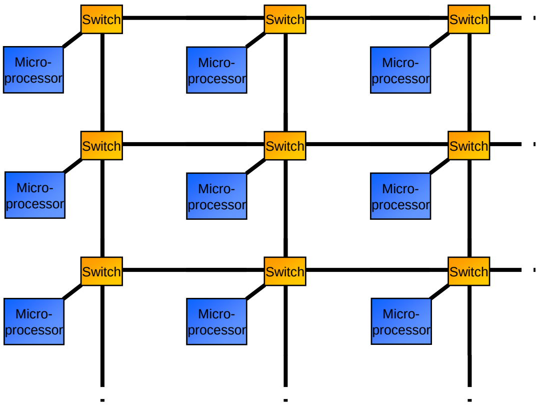
Full system functionality is only reached after hardware implementation

Possible solution?

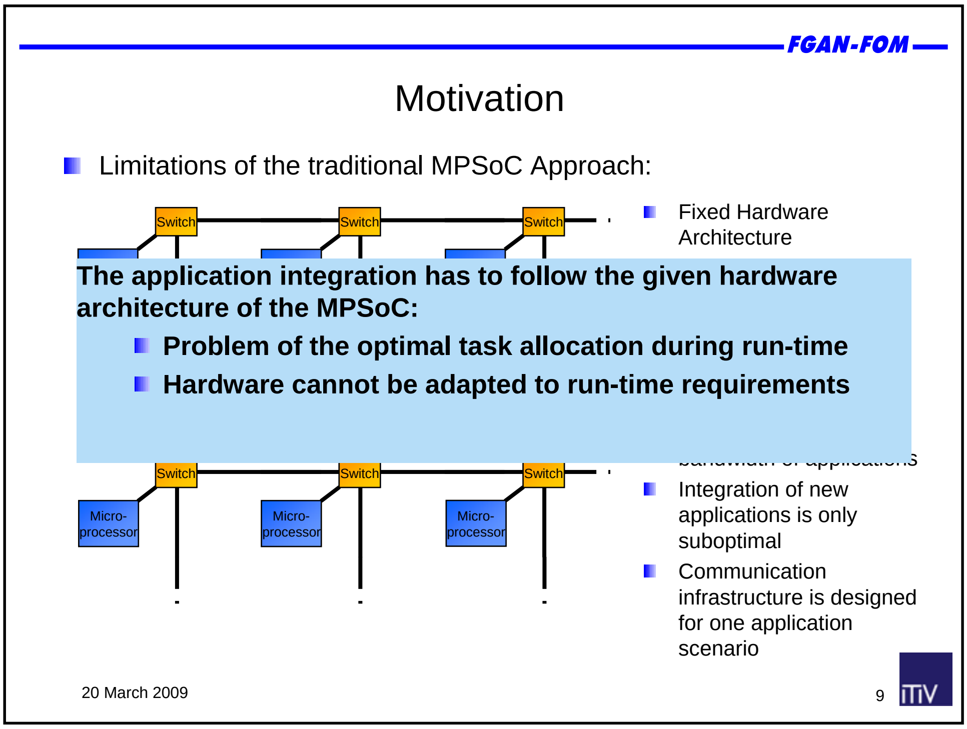
Rapid Prototyping with RAMPSoC system on FPGA

Motivation

- Limitations of the traditional MPSoC Approach:



- Fixed Hardware Architecture
- Processor architecture mostly optimized for a special application
- Often inefficient task allocation
  → Unequal workload
- Designed for a small bandwidth of applications
- Integration of new applications is only suboptimal
- Communication infrastructure is designed for one application scenario



Motivation

- Limitations of the traditional MPSoC Approach:

Switch
Switch
Switch
Switch
Switch
Switch
Micro-processor
Micro-processor
Micro-processor

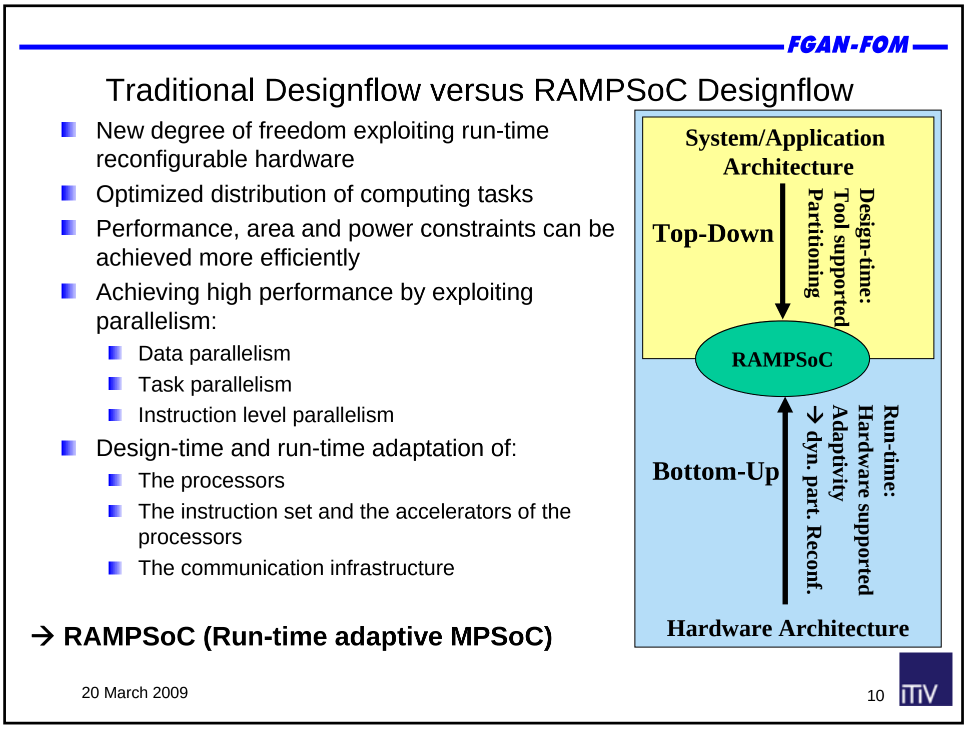
The application integration has to follow the given hardware architecture of the MPSoC:

- **Problem of the optimal task allocation during run-time**
- **Hardware cannot be adapted to run-time requirements**

- Fixed Hardware Architecture

bandwidth of applications

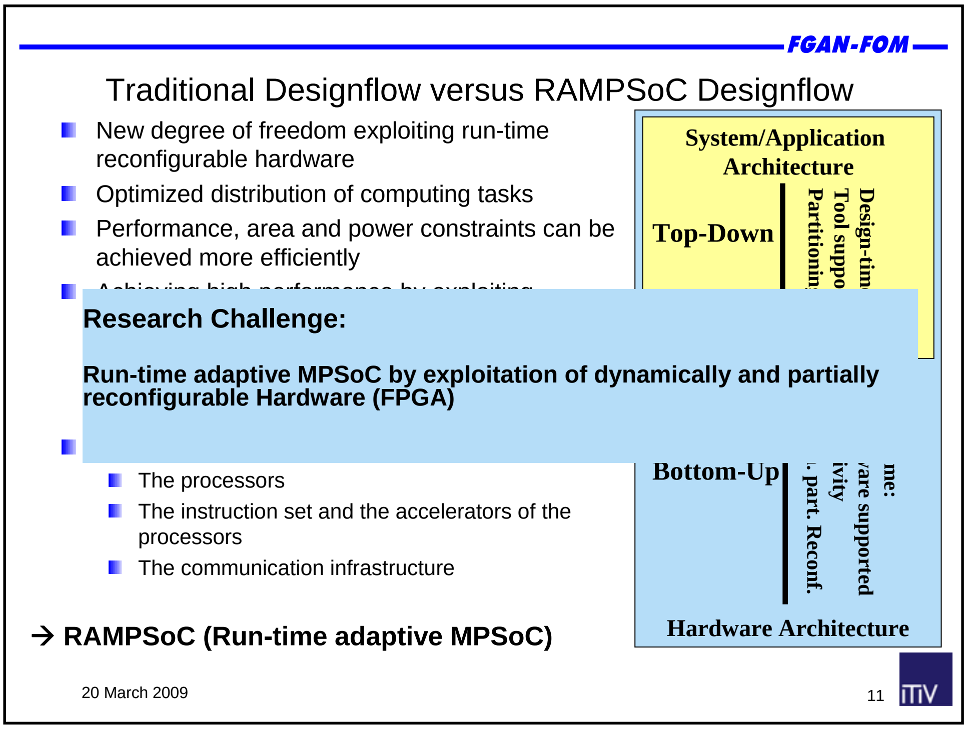
- Integration of new applications is only suboptimal
- Communication infrastructure is designed for one application scenario



Traditional Designflow versus RAMPSoC Designflow

- New degree of freedom exploiting run-time reconfigurable hardware
- Optimized distribution of computing tasks
- Performance, area and power constraints can be achieved more efficiently
- Achieving high performance by exploiting parallelism:
  - Data parallelism
  - Task parallelism
  - Instruction level parallelism
- Design-time and run-time adaptation of:
  - The processors
  - The instruction set and the accelerators of the processors
  - The communication infrastructure

→ RAMPSoC (Run-time adaptive MPSoC)



Traditional Designflow versus RAMPSoC Designflow

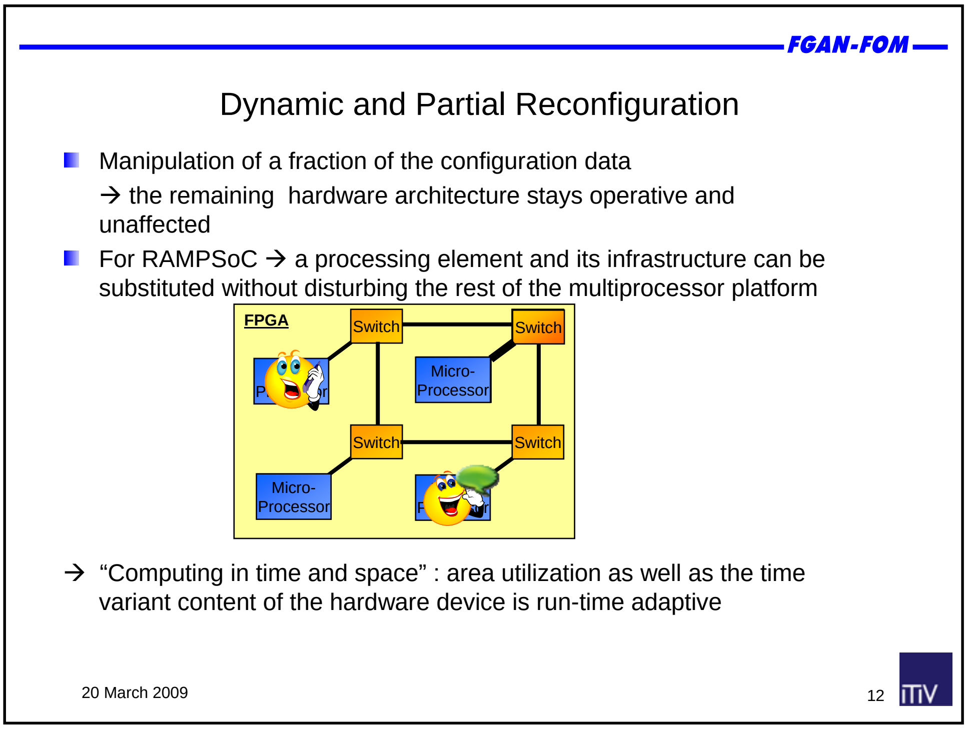
- New degree of freedom exploiting run-time reconfigurable hardware
- Optimized distribution of computing tasks
- Performance, area and power constraints can be achieved more efficiently
- Achieving high performance by exploiting

Research Challenge:

Run-time adaptive MPSoC by exploitation of dynamically and partially reconfigurable Hardware (FPGA)

-
  - The processors
  - The instruction set and the accelerators of the processors
  - The communication infrastructure

→ **RAMPSoC (Run-time adaptive MPSoC)**

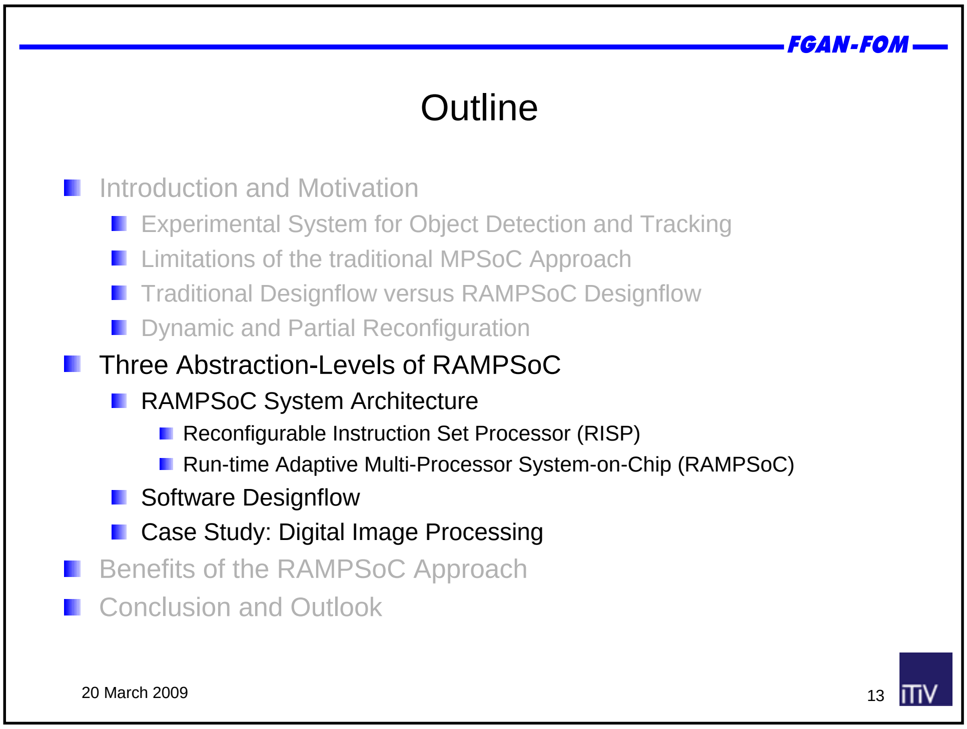


Dynamic and Partial Reconfiguration

- Manipulation of a fraction of the configuration data
  → the remaining hardware architecture stays operative and unaffected
- For RAMPSoC → a processing element and its infrastructure can be substituted without disturbing the rest of the multiprocessor platform

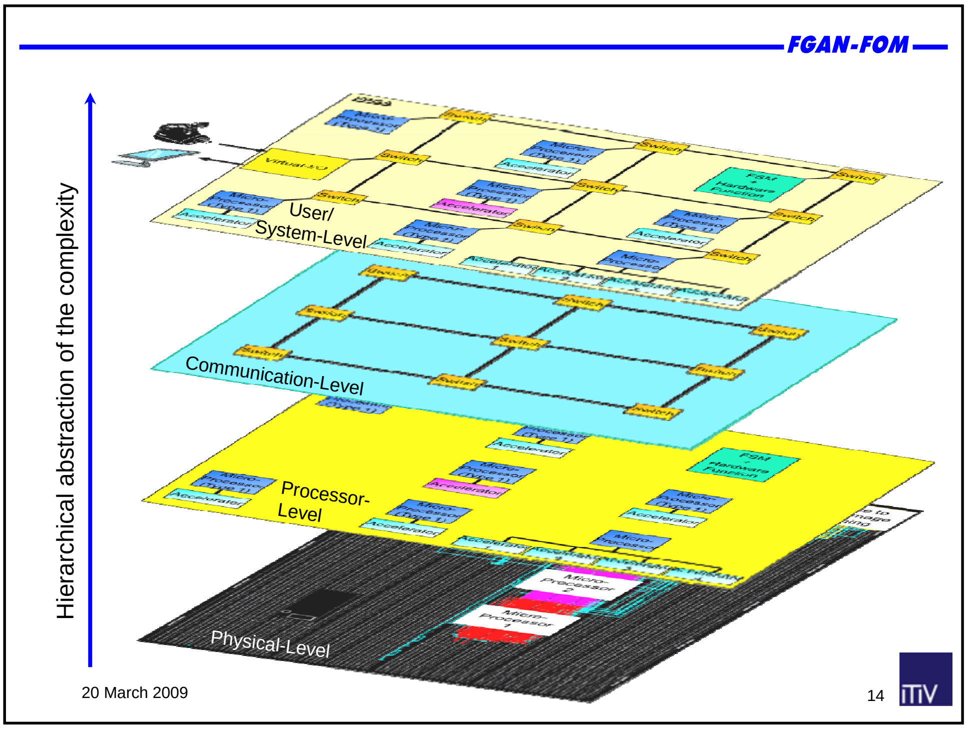
FPGA
Switch
Switch
Micro-Processor
Switch
Switch
Micro-Processor

→ “Computing in time and space” : area utilization as well as the time variant content of the hardware device is run-time adaptive

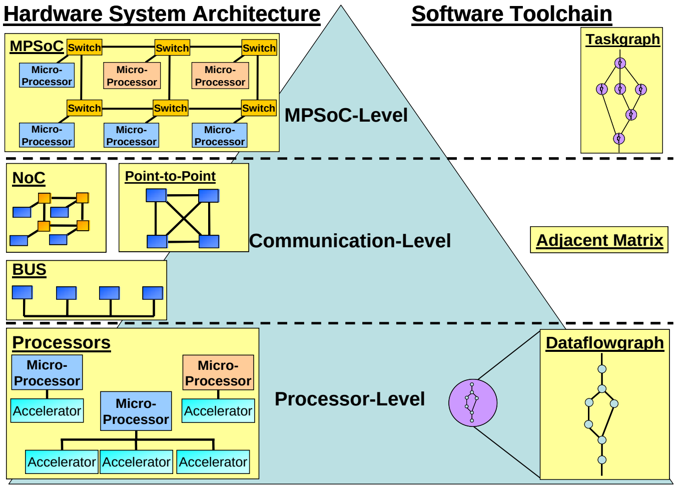


Outline

- Introduction and Motivation
  - Experimental System for Object Detection and Tracking
  - Limitations of the traditional MPSoC Approach
  - Traditional Designflow versus RAMPSoC Designflow
  - Dynamic and Partial Reconfiguration
- Three Abstraction-Levels of RAMPSoC
  - RAMPSoC System Architecture
    - Reconfigurable Instruction Set Processor (RISP)
    - Run-time Adaptive Multi-Processor System-on-Chip (RAMPSoC)
  - Software Designflow
  - Case Study: Digital Image Processing
- Benefits of the RAMPSoC Approach
- Conclusion and Outlook

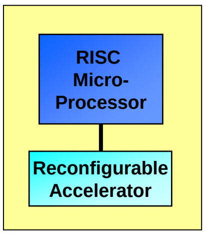


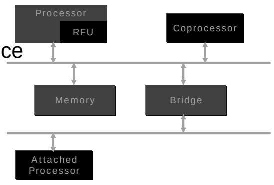
Three Abstraction-Levels of RAMPSoC



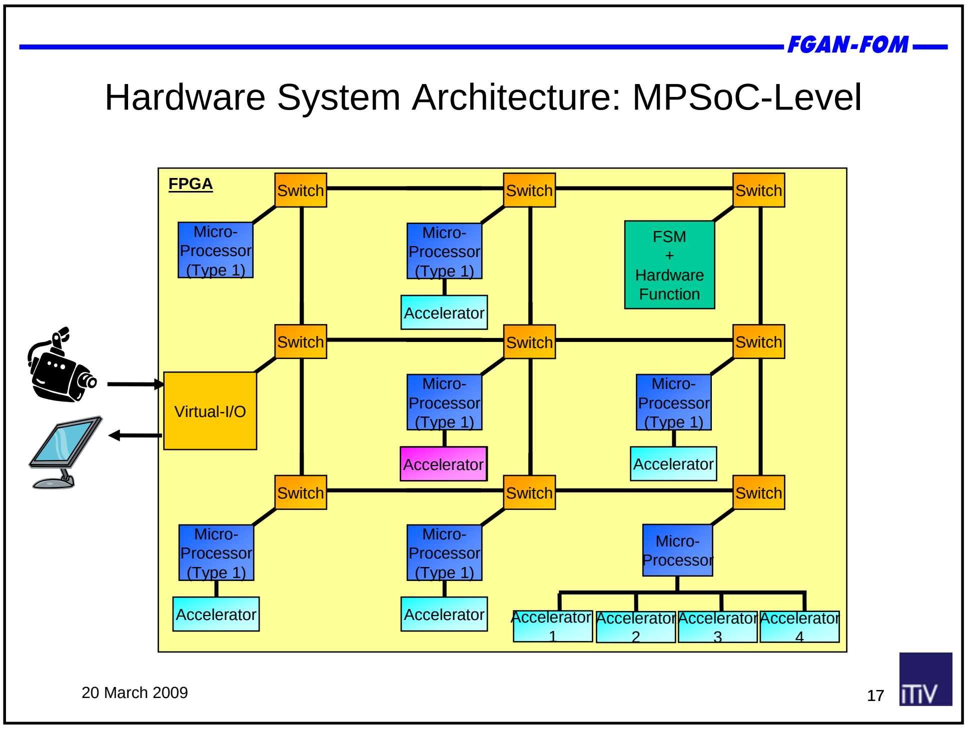
Processor-Level : RISP

- Reconfigurable Instruction Set Processor
- Like an ASIP (Application Specific Instruction Set Processor) with the extension to exchange the specialized functional units (FU) during run-time (reconfigurable)
- Benefits over ASIPs:
  - Versatile applicable for a wide range of applications (e.g. Image Processing, Telecommunications, …)
  - Particularly suitable for applications with diverse requirements (Algorithms)
  - Very flexible due to dynamic and partial reconfiguration
  - Lower power consumption
    → only actual required FUs are implemented on the device
  - Smaller Area
  - More cost-efficient and shorter Time-to-Market
    → only one mask required





Reference: Reconfigurable Instruction Set Processors: A Survey, F. Barat & R. Lauwereins (K.U.Leuven) , RSP 2000



Hardware System Architecture: MPSoC-Level

FPGA

Switch

Switch

Switch

Micro-Processor (Type 1)

Micro-Processor (Type 1)

Accelerator

FSM + Hardware Function

Switch

Switch

Switch

Virtual-I/O

Micro-Processor (Type 1)

Accelerator

Micro-Processor (Type 1)

Accelerator

Switch

Switch

Switch

Micro-Processor (Type 1)

Accelerator

Micro-Processor (Type 1)

Accelerator

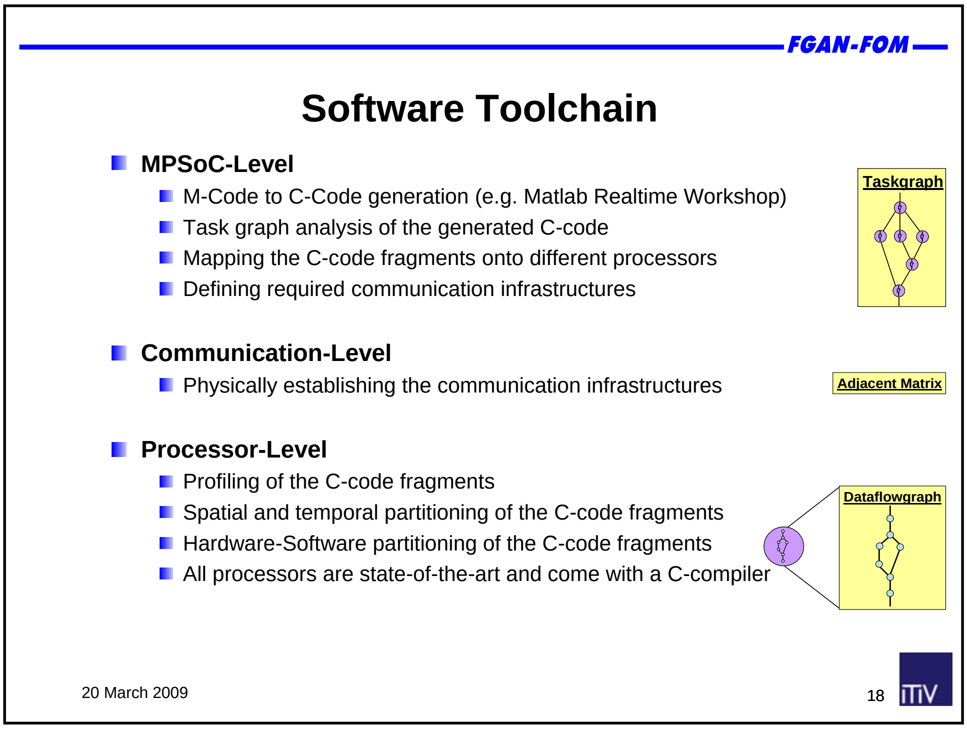
Micro-Processor

Accelerator 1

Accelerator 2

Accelerator 3

Accelerator 4



Software Toolchain

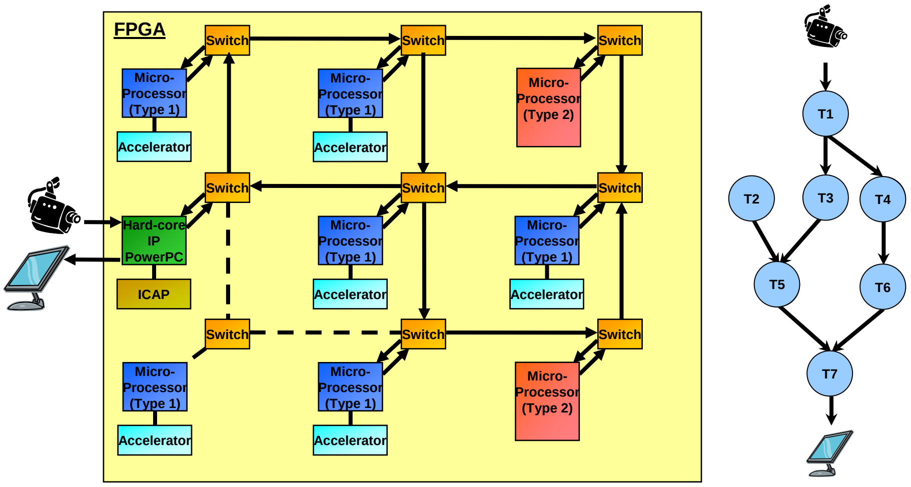
- **MPSoC-Level**
  - M-Code to C-Code generation (e.g. Matlab Realtime Workshop)
  - Task graph analysis of the generated C-code
  - Mapping the C-code fragments onto different processors
  - Defining required communication infrastructures
- **Communication-Level**
  - Physically establishing the communication infrastructures
- **Processor-Level**
  - Profiling of the C-code fragments
  - Spatial and temporal partitioning of the C-code fragments
  - Hardware-Software partitioning of the C-code fragments
  - All processors are state-of-the-art and come with a C-compiler

Taskgraph

Adjacent Matrix

Dataflowgraph

Case Study : Digital Image Processing



Outline

- Introduction and Motivation
  - Experimental System for Object Detection and Tracking
  - Limitations of the traditional MPSoC Approach
  - Traditional Designflow versus RAMPSoC Designflow
  - Dynamic and Partial Reconfiguration
- Hierarchy-/Virtualization-Levels of RAMPSoC
  - RAMPSoC System Architecture
    - Reconfigurable Instruction Set Processor (RISP)
    - Run-time Adaptive Multi-Processor System-on-Chip (RAMPSoC)
  - Software Designflow
  - Case Study: Digital Image Processing
- Benefits of the RAMPSoC Approach
- Conclusion and Outlook

Benefits of the RAMPSoC Approach

- Programming model available
  Designtools (Compiler, Tools, Processors) exist

- Advantages over pure hardware solutions:
  - More time-efficient designflow through software adaptation and the usage of existing processor cores
  - More flexible through Hardware-Software-Codesign
  - Adapting the system more easily to new tasks through software design

- Advantages over RISPs:
  - Better Performance due to extended parallelism
  - Higher computation power (expected ☺)
  - Possibility to execute several applications in parallel

Benefits of the RAMPSoC Approach

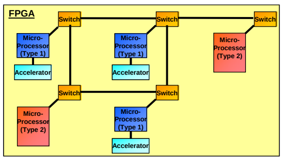
- Advantages over traditional MPSoCs
  - Better Performance due to parallelization of complex tasks in Hardware
  - Lower power consumption → Computing power on-demand
  - More flexible and versatile → optimizable during design-time and during run-time
  - Reduced costs and faster Time-to-Market

→ RAMPSoC inherits the advantages of RISPs and traditional MPSoCs and extends them by using the dynamic and partial reconfiguration feature in the MPSoC-level (Adaptation of the communication infrastructure, whole processors and instruction sets).

Outline

- Introduction and Motivation
  - Experimental System for Object Detection and Tracking
  - Limitations of the traditional MPSoC Approach
  - Traditional Designflow versus RAMPSoC Designflow
  - Dynamic and Partial Reconfiguration
- Hierarchy-/Virtualization-Levels of RAMPSoC
  - RAMPSoC System Architecture
    - Reconfigurable Instruction Set Processor (RISP)
    - Run-time Adaptive Multi-Processor System-on-Chip (RAMPSoC)
  - Software Designflow
  - Case Study: Digital Image Processing
- Benefits of the RAMPSoC Approach
- Conclusion and Outlook

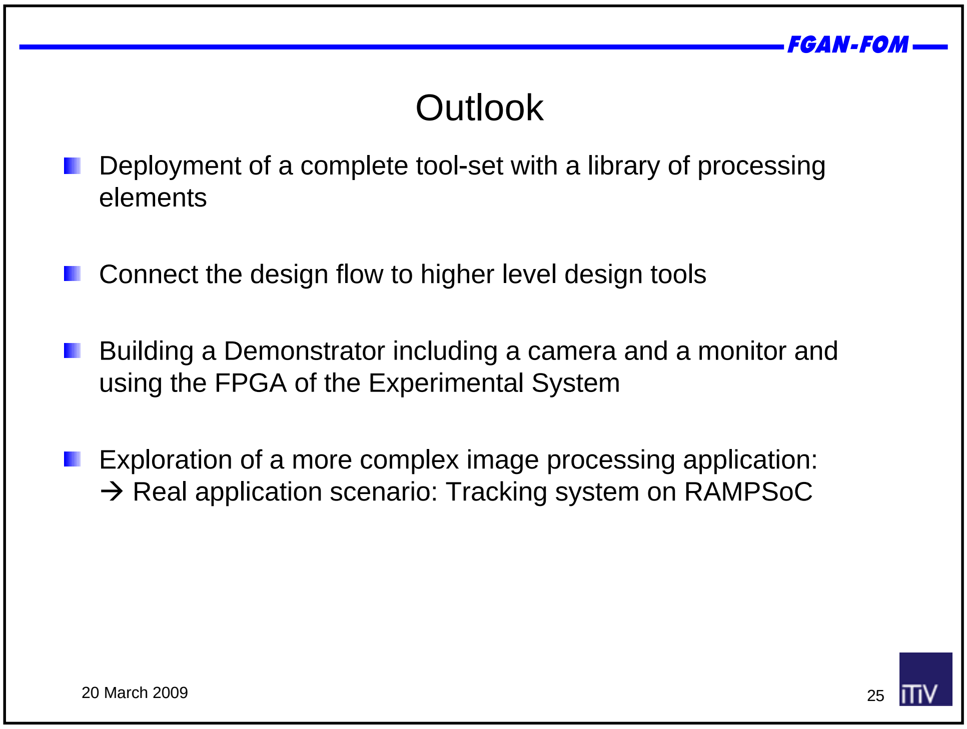
Conclusion



- Modular and adaptive MPSoC → RAMPSoC

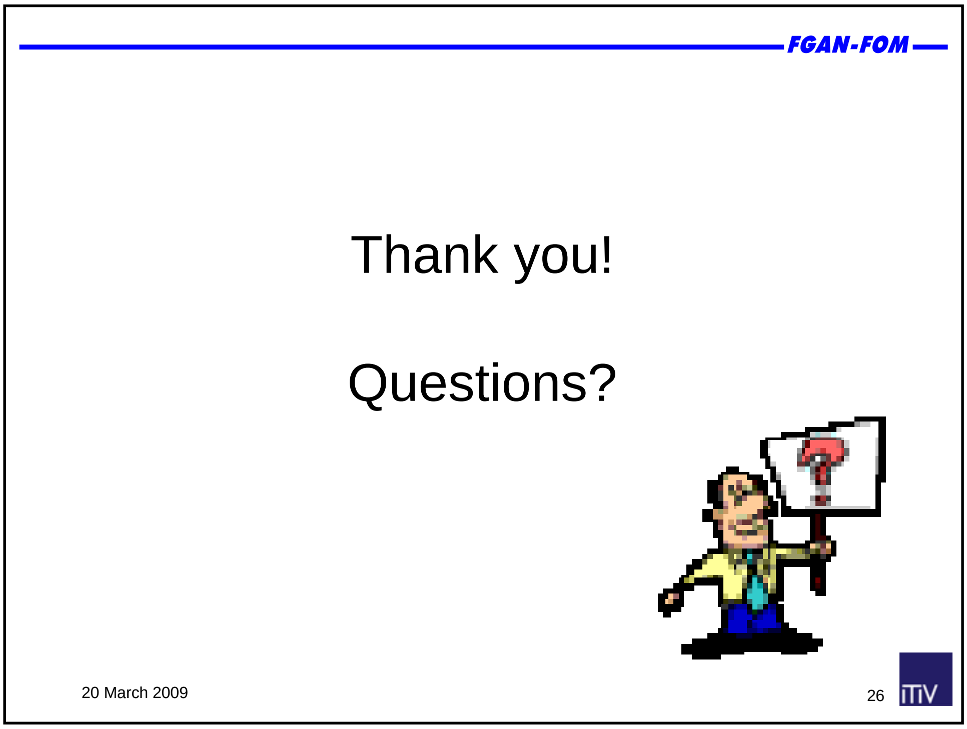
- Run-time adaptation through dynamic and partial reconfiguration:
  - Processor (bitwidth, architecture (VLIW, RISC, CISC))
  - Accelerator
  - Communication infrastructure (e.g. Network-on-Chip)
  - Optimizing the communication paths through component migration

- Advantages over traditional MPSoCs:
  - Lower power consumption → on-demand functionality
  - More flexibility → run-time adaptation
  - Reduced costs → re-use for a multitude of applications
  - Better performance → exploitation of parallelization and reconfiguration



Outlook

- Deployment of a complete tool-set with a library of processing elements
- Connect the design flow to higher level design tools
- Building a Demonstrator including a camera and a monitor and using the FPGA of the Experimental System
- Exploration of a more complex image processing application:
  → Real application scenario: Tracking system on RAMPSoC



Thank you!

Questions?