Evaluation of Software Architectures in the Automotive Domain for Multicore Targets in regard to Architectural Estimation Decisions at Design Stage

Master-Thesis Defence
André Christian Roßbach
Automotive Software Engineering
Content

1. Motivation
2. Topic & Tasks
3. Multicore Systems Model
4. Estimation-Procedure
5. Application & Simulation
6. Results
Typical Driver-Assistance Functions

Motivation
- Topic
- Model
- Procedure
- Application
- Simulation
- Results

- Adaptive Cruise Control
- Traffic Sign Recognition
- Emergency Braking
- Pedestrian Detection
- Collision Avoidance
- Lane Departure Warning
- Cross Traffic Alert
- Park Assist
- Blind Spot Detection
- Rear Collision Warning
- Surround View

[TI TDA2x 2013]

Long-Range Radar
LiDAR
Camera
Short-/Medium Range Radar
Ultrasound
Massive Data Throughput
Emerging Multicore Technology

Motivation
- Topic
- Model
- Procedure
- Application
- Simulation
- Results

"1 ECU = 1 Application"

Multicore Domain Units

- Number of processor-cores per ECU
- Number of applications per ECU
- Number of ECUs per vehicle
- Number of applications per vehicle


Time
What is Multicore?

Motivation
• Topic
• Model
• Procedure
• Application
• Simulation
• Results
Example Question

“Is it possible to evaluate the suitability of a multicore hardware platform for a given software architecture even at design time?”
Software Architecture (SA)

Motivation • Topic • Model • Procedure • Application • Simulation • Results
Topic: Scope at the Development Cycle
2 Tasks

1. Multicore System Model

2. Estimation-Procedure
Title

**Evaluation of Software Architectures**

*in the Automotive Domain for Multicore Targets*

*in regard to Architectural Estimation Decisions at Design Stage*
1. Task
Multicore System Model
Collecting System Attributes
Creating System Model

- Communication
- Memory
- Processor Cores

Graph-Model

Motivation - Topic - Model - Procedure - Application - Simulation - Results
Graph-Models

Motivation • Topic • Model • Procedure • Application • Simulation • Results
Creating System Model

Communication

Memory

Processor Cores

Hardware Tests

Motivation • Topic • Model • Procedure • Application • Simulation • Results
## Parameter Influence

<table>
<thead>
<tr>
<th>Neglectable</th>
<th>Important</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPC Intra-ECU</td>
<td>External Inter-ECU</td>
<td><strong>CAN vs. NoC</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>308 µs vs. 0,01 µs (30 000:1)</td>
</tr>
<tr>
<td>Local Memory Small Data</td>
<td><strong>Shared Memory</strong> Big Data</td>
<td><strong>Image vs. Variable</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1,46 mio vs. 145 Cycles (10 000:1)</td>
</tr>
<tr>
<td>Short Execution Timings</td>
<td><strong>Short Realtime Deadlines</strong></td>
<td><strong>Streaming vs. Control Loop</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>20 mio vs. 20k Cycles (1000:1)</td>
</tr>
</tbody>
</table>

**Communication** (IPC vs. External)

**Memory** (Shared Memory Access)

**Processor Cores** (Execution Timings)
2. Task

Estimation-Procedure

PROCEDURE
Estimation-Procedure

Input

Modelling

Verification

Results

Motivation • Topic • Model • Procedure • Application • Simulation • Results
Example Project
Online Camera Calibration
OCC System Structure

Load Tile

Tiles

Find Reference Points

Ref-Points

Egomotion Estimation

Load Tile

Delay

Tiles

ReFind Points

vector

Compare Values

vector

Calculate Camera Position

Camera Pos.

Minimize Position Error

Receive Camera Image

image

Store Image

05.11.2014

Master-Thesis (ASE)
OCC System Structure

Motivation • Topic • Model • Procedure • Application • Simulation • Results
OCC System Structure

Load Tile
- Tiles
- Find Reference Points
  - Ref-Points
  - Egomotion Estimation

Load Tile
- Delay
- Tiles
- ReFind Points
  - vector
  - Compare Values
  - vector
  - Calculate Camera Position
  - Cam-Pos
  - Minimize Position Error

Receive Camera Image
- Store Image

Find Reference Points
- vector
- Compare Values
- vector
- Calculate Camera Position
- Minimize Position Error

Receive Camera Image
- Store Image

Motivation • Topic • Model • Procedure • Application • Simulation • Results

Shared Memory
- L3 RAM

DSP
- Local Memory1
- Local Memory2
- Local Memory3
- Local Memory4

ARM A8
- Local Memory1
- Local Memory2
- Local Memory3
- Local Memory4

Display
- EVE

Cortex M3
- Cortex M3

Local Memory
- ARM A8
- DSP
- Display

Bus
- Video-In
- Video-In

Receive Camera Image
- Store Image

Minimize Position Error
- Delay

Compare Values
- Egomotion Estimation
- DSP
- ARM A8
- L3 RAM

Calculate Camera Position
- Cam-Pos
- DSP
- ARM A8
- L3 RAM

Minimize Position Error
- DSP
- ARM A8
- L3 RAM
OCC System Structure

Motivation • Topic • Model • Procedure • Application • Simulation • Results
Simulation Tasks

1. Modelling
   - SW Structure
   - HW Mapping

2. Simulation
   - Runtime Behaviour
   - Realtime Constraints

3. Optimisation
   - SW Structure
   - Load Chart
   - Verification

4. Tool Usability
Complexity of Underlying Model

Motivation • Topic • Model • Procedure • Application • Simulation • Results

05.11.2014

Master-Thesis (ASE)

www.tu-chemnitz.de
Modelling Software Structure

Communication Graph
Runnable Sequence
Path-Graph
Modelling Mapping

TaskGraph

SystemChart

Mapping Tree
Modeling Timings

Motivation • Topic • Model • Procedure • Application • Simulation • Results
Runtime Behaviour

SymTA/S Distribution

INCHRON Schedule
Realtime Constraints

**INCHRON chronVAL**

**TA Tool Suite Event-Chains**

Motivation • Topic • Model • Procedure • Application • Simulation • Results

05.11.2014 Master-Thesis (ASE)
## Optimisation
### Online Camera Calibration

<table>
<thead>
<tr>
<th>Original</th>
<th>4000s</th>
</tr>
</thead>
<tbody>
<tr>
<td>SW Structure</td>
<td></td>
</tr>
<tr>
<td>Analysis</td>
<td></td>
</tr>
<tr>
<td>Scheduling</td>
<td></td>
</tr>
<tr>
<td>Verification</td>
<td></td>
</tr>
<tr>
<td><strong>Result</strong></td>
<td><strong>?</strong></td>
</tr>
</tbody>
</table>

- **Motivation**
- **Topic**
- **Model**
- **Procedure**
- **Application**
- **Simulation**
- **Results**
Optimisation
Software Structure

Motivation • Topic • Model • Procedure • Application • Simulation • Results

05.11.2014
Master-Thesis (ASE)
Optimisation
INCHRON Load Diagram

Cortex M3
Cortex M3
Shared Memory
Display
ARM A8
DSP

Motivation • Topic • Model • Procedure • Application • Simulation • Results
Optimisation
SymTA/S Load Bar Chart

Load Bar Chart for 6 resources
7 columns: CORE1_ARM_Cortex_M3_32Bit_200MHz, CORE2_ARM_Cortex_M3_32Bit_200MHz, CORE_ARM_Cortex_A8_32B...

- Cortex M3: 92.16%
- Cortex M3: 92.16%
- ARM A8: 92.16%
- DSP: 69.22%
- Display: 69.22%
- Shared Memory: 8.65%

Motivation • Topic • Model • Procedure • Application • Simulation • Results
Optimisation Formal Verification
INCHRON chronVAL WCRT
Optimisation
Online Camera Calibration

<table>
<thead>
<tr>
<th>Optimisation Steps</th>
<th>Original</th>
<th>4000s</th>
</tr>
</thead>
<tbody>
<tr>
<td>SW Structure</td>
<td>Period</td>
<td></td>
</tr>
<tr>
<td>Scheduling</td>
<td>Load Diagram</td>
<td></td>
</tr>
<tr>
<td>Analysis</td>
<td>Load Chart</td>
<td></td>
</tr>
<tr>
<td>Verification</td>
<td>WCRT</td>
<td></td>
</tr>
</tbody>
</table>

**Result:** 167s

24x faster
Simulation Tools Usability

Motivation • Topic • Model • Procedure • Application • Simulation • Results
Results and Outlook

RESULTS
Example Question

“Is it possible to evaluate the suitability of a multicore hardware platform for a given software architecture?”

Yes

- Estimation-Procedure
- Behavioural Verification
- Structural Verification
- Optimisations
Results

1. Trend Analysis
2. Multicore-Project Analysis
   • Techniques
   • Question Catalogue
3. Multicore-System Attributes
4. Abstract Graph-Models (SW, HW, Mapping)
5. Estimation-Procedure
6. SA Simulation Tool Evaluation
Outlook

• Transformation:
  • Project-Analysis → System Description → Graph-Model → Tool

• Domain-specific Decision Guideline:
  • Domain → Decomposition → Impact of Parameter → Possible Bottlenecks → Test-Approaches

• Modelling:
  • Graph-Algorithms
  • SysML
Sources

[IAV_(1)] https://intranet.iavgroup.local/mo-g/de/news/ak-test-teststandard-iso29119-und-iav-testtools.html

[IAV_(2)] https://intranet.iavgroup.local/content/divisions-v/vi/de/leistungen0/fahrerassistenzsysteme-und-aktive-sicherheit.html


[VDC] VDC Research, “Next Generation Embedded Hardware Architectures: Driving Onset of Project Delays, Costs Overruns, and Software Development Challenges”, - September 2010
Thank You!
BACKUP-SLIDES
Emerging Multicore Technology

Motivation

- Topic
- Model
- Procedure
- Application
- Simulation
- Results

"1 Application = 1 ECU"
Domain Units


Number of applications per vehicle

Number of applications per ECU

Number of processor-cores per ECU
The European automotive market takes great benefit in leading research in the software technology for heterogeneous, embedded multicore ECUs.
HPC vs. Embedded Systems

Desktop/Server, High Performance Homogenous Multicore Platforms:

- Intel Xeon E5-2600
- Up to 20MB CACHE
- QPI 1
- QPI 2

Automotive, Embedded, Safety Critical, Heterogeneous Multicore Platforms:

- Checker Core
- FPU
- TriCore™ 1.6P
- DMI Overlay
- Data Flash
- BROM
- Key Flash

- Checker Core
- FPU
- TriCore™ 1.6P
- DMI Standby Overlay

- GTM
- CCUx
- GTP1.2x
- STM
- SCU
- BCU
- HSSL
- PLL & PLL EAY

- Ethernet
- PCIe
- MPCore
- MESA
- ASC LINK
- QSPI
- SENT
- POSI
- PC
- PCE
- IM1

bladesmadesimple.com, 2013 – “Intel Xeon E5-2600 Dual CPU”

## Programming Paradigm Shift

<table>
<thead>
<tr>
<th></th>
<th>Conventional</th>
<th>New</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>HW</strong></td>
<td>Microcontroller</td>
<td>Mixed, heterogeneous, specialized MPSoC</td>
</tr>
<tr>
<td><strong>SW</strong></td>
<td>“bare metal” Independent resources accesses</td>
<td>Distributed Apps</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Dependencies (Shared resources)</td>
</tr>
<tr>
<td><strong>Application-Distribution</strong></td>
<td>„one function per ECU“</td>
<td>MDCU</td>
</tr>
<tr>
<td></td>
<td>Easy mapping + scheduling</td>
<td>Scheduling-Tools</td>
</tr>
<tr>
<td><strong>Implementation-Support</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Data-size</strong></td>
<td>Small (\textbf{message})</td>
<td>Big (\textbf{Images})</td>
</tr>
<tr>
<td><strong>Algorithmic complexity</strong></td>
<td>(\textbf{(Micro-)Controller})</td>
<td>Image-Processing, Complex Algorithms</td>
</tr>
<tr>
<td><strong>HW-Dependencies</strong></td>
<td>Highly optimized (no OS)</td>
<td>Several Runnables (RTOS)</td>
</tr>
</tbody>
</table>

→ Analyzed Projects: No AUTOSAR
Topic: Scope at the Development Cycle

“Multicore and multiprocessor software projects are 4,5x more expensive, have 25% longer schedules, and require almost 3x as many software engineers.”

[VDC]
Backup-Slides
Model & Tests
Software Function-Graph

Motivation • Topic • Basics • Multicore • Model • Procedure • Results

Function Block
- Execution Timings in Cycles (for each core)
- Local Memory-Usage
- Real-time Requirements (Deadline/Period)
- Activation-Pattern/Stimuli
- Main Computation
- Resource Access

Message
- Occurrence (Period vs. Event-triggered)
- Message Size
- Protocol

Dependency
(IPA, IPC, Stimuli, Activation, Event)
Hardware Architecture-Graph

- **Local Memory**
  - Access Timing
  - Size

- **Resource/Interface**
  - Input Period
  - Data Size
  - Throughput
  - Buffer

- **Processor Core**
  - Instructions/Cycles per Second
  - Kind of Core (for Computation)

- **Communication Channel**
  - Shared Memory vs. Message Passing
  - Effective Bandwidth
  - Latency

- **Shared Memory**
  - Access Timing
  - Size

- **Core**

- **Binding**
  (Bandwidth)
# Schedule Representations

<table>
<thead>
<tr>
<th>Power-point</th>
<th>UML Timing Diagram</th>
<th>UML Sequenz Chart</th>
<th>GANTT Chart</th>
<th>LaTeX</th>
<th>Excel</th>
<th>SysML?</th>
</tr>
</thead>
</table>

**Figure 1:** Scheduling Example
# System Model

## Functional Requirements

<table>
<thead>
<tr>
<th>SOFTWARE MODEL</th>
<th>Processor execution timings on the particular processor cores</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ability to model the software in an abstract way</td>
<td>Realtime constraints, like deadlines and periods</td>
</tr>
<tr>
<td></td>
<td>Dependencies to other FBs</td>
</tr>
<tr>
<td></td>
<td>Preloaded resource usage of other applications running on the hardware target</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>HARDWARE MODEL</th>
<th>Processor-Cores (quantity, kind and speed)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ability to model the hardware in an abstract way</td>
<td>Global and local memory (for shared memory accesses)</td>
</tr>
<tr>
<td></td>
<td>Arrangement and Bindings of all the hardware components</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TIMING MODEL</th>
<th>Modelling of timing requirements like execution times, deadlines, jitter, communication time, synchronization points, etc.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scheduling of the FB according to the realtime-requirements</td>
<td>Deadline satisfaction</td>
</tr>
<tr>
<td></td>
<td>End-to-end latency of event-chains</td>
</tr>
<tr>
<td></td>
<td>Stimulation of events and tasks in a periodic, sporadic and singular way</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>COMMUNICATION MODEL</th>
<th>Modelling of Intra- and Inter-ECU networks (with speed, bandwidth, etc.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Utilization of the communication channels</td>
<td>Modelling of the usage of communication time and bandwidth</td>
</tr>
<tr>
<td></td>
<td>End-to-end latency (including waiting time at collision)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MEMORY MODEL</th>
<th>Memory usage, accessing, bindings, access and waiting timings</th>
</tr>
</thead>
<tbody>
<tr>
<td>Behaviour of memory-access in timing and size</td>
<td>Defining of restricts of size for mapping considerations</td>
</tr>
<tr>
<td></td>
<td>Modelling of semaphores or limited access channels</td>
</tr>
</tbody>
</table>
### Function Block (FB)

**Function + Non-Functional Req.**

<table>
<thead>
<tr>
<th>Hierarchical structure and restricted size of a FB</th>
<th>Timing (budget)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Dependency</strong></td>
<td>(Estimated) maximal (budget) execution runtime</td>
</tr>
<tr>
<td>Clear dependencies at the beginning and end to other FBs</td>
<td></td>
</tr>
<tr>
<td>No dependency to other FBs within the FB</td>
<td></td>
</tr>
<tr>
<td><strong>Orthogonality</strong></td>
<td>WCET</td>
</tr>
<tr>
<td>The FBs need to be able to run independent from other FBs except the described dependencies (for concurrent execution)</td>
<td></td>
</tr>
<tr>
<td><strong>Modularity Amount</strong></td>
<td>AVGET Timings</td>
</tr>
<tr>
<td>The number of FBs should be far more than the number of processor cores</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Restricted size of interfaces and restricted coupling between the FBs</th>
<th>Realtime constraints</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Information flow</strong></td>
<td>Deadline, period-length, hard- vs. soft-realtime, priority, schedule advice</td>
</tr>
<tr>
<td>No information flow outwardly the local scope of the FB (no external memory access necessary for communication between two instructions of the same FB)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Communication Encryption Memory Access</th>
<th>Memory (budget)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Small communication overhead (very few information exchanges between the FBs)</td>
<td>(Estimated) maximal (budget) memory usage</td>
</tr>
<tr>
<td>All instructions access the same (internal and external) memory</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>High cohesion within the FB to reach an abstract view</th>
<th>Kind of computation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Close functionality</strong></td>
<td>Group of functionality of the instructions</td>
</tr>
<tr>
<td>The functionality within a FB serve the same goal (e.g. image processing)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Same Core Simplicity Classification</th>
<th>Possible Mapping</th>
</tr>
</thead>
<tbody>
<tr>
<td>All instructions are able to run on the same instruction set (processor-core)</td>
<td>Ability (e.g. instructions, bindings, etc.) to be mapped to different processor-cores</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Occurrence Functionality</th>
<th>57</th>
</tr>
</thead>
<tbody>
<tr>
<td>The FB as such can be classified in one of these groups: Processing, Algorithmic executions, Shared Memory Access (for big data)</td>
<td></td>
</tr>
</tbody>
</table>

05.11.2014

Master-Thesis (ASE)
# Kind of Computation + Dependencies

<table>
<thead>
<tr>
<th>Kind of Computation</th>
<th>Degree of Parallelism</th>
<th>Communication Load</th>
<th>Advised Core Type</th>
<th>Advised Parallel Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mathematical Calculation</td>
<td>Mostly very high, because of completely independent calculations</td>
<td>In the best case only to distribute the computation and collect the results</td>
<td>Homogenous multi-processor system with message passing</td>
<td>?</td>
</tr>
<tr>
<td>Image Processing</td>
<td>Mostly very high, because of parallel vector calculations</td>
<td>Mostly very high, because the Calculations are mutual dependent</td>
<td>Multi-core system with shared memory</td>
<td>Pipelining for sequences</td>
</tr>
<tr>
<td>Signal Processing</td>
<td>Very low, because only single instructions can be parallelized (which is mostly already done in HW)</td>
<td>Generally every signal-stream need to be processed and communicated from the input to the output</td>
<td>DSP (HW parallelism) or Single Core GPU</td>
<td>Pipelining</td>
</tr>
<tr>
<td>Interrupt Service</td>
<td>Very low, because an ISR should not be interrupted again - Only several interrupt catches can be done</td>
<td>Every interrupt needs to be thrown and caught – at the end the result needs to be communicated</td>
<td>Single Core GPU</td>
<td>-</td>
</tr>
<tr>
<td>Inter-ECU Communication</td>
<td>None, except several busses are given</td>
<td>Very high - Collect and send intra and inter-ECU data</td>
<td>Single core ASIC with big buffers</td>
<td>Producer-Consumer</td>
</tr>
<tr>
<td>I/O</td>
<td>Mostly only serial execution, maybe filled buffers can processed simultaneously</td>
<td>High, the reason is to communicate the input/output to other parts.</td>
<td>Single core ASIC with specified buffers - Otherwise parallel hardware I/O is required</td>
<td>-</td>
</tr>
</tbody>
</table>

**Functional**
- The functional continuing of former FBs
- Call from one FB to start another FB

**Data**
- Access to global variables
- Shared data (memory regions) (including locks, semaphores, mutexes, etc.)
- Synchronisation points
Double Roof Model

- Function Blocks
- Functional Model
- Software Architecture
- System
- Architecture
- Multicore HW target
- Resources

- UML Sequence diagram
- Dependencies between FBs
- Structure + Bindings
- Communication channels bandwidth + latency
- Processor Cores velocity
- Shared + Local Memory access timing

- Realtime-Req.
- Messages
- Execution Timings
- Memory usage
- Processor Cores velocity
- Communication channels bandwidth + latency
- Shared + Local Memory access timing

Adaption Double Roof Model
[Jürgen Teich]
5 Step Analysis Technique

- Understanding
- Structure Description
- Runtime Behaviour
- Mapping
- Alternations
System View - Parameters

Main Targets

- **Computations**
  - Algorithms
  - Processing
  - Timings

- **Safety critical**
  - Monitoring
  - Error Checking
  - Redundancy
  - Realtime Constraints

- **Controlling**
  - External Signals/Sensor Data
  - Control Loop Period

- **Parallelization**
  - Computational (Big Data)
  - Functional (FB → Cores)

Main Problems

- **Bottlenecks**
  - Communication Architecture
  - Memory Access (Overflow, Race Condition, Lost Update)
  - Execution Timings

- **Memory**
  - Buffer Overflow
  - Shared Resource Access Timings

- **Communication**
  - Throughput Reduction
  - Latency

- **Parallelization**
  - Computational (Big Data)
  - Functional (FB → Cores)

Main Problems

- **Safety critical**
  - Monitoring
  - Error Checking
  - Redundancy
  - Realtime Constraints

- **Controlling**
  - External Signals/Sensor Data
  - Control Loop Period

- **Parallelization**
  - Computational (Big Data)
  - Functional (FB → Cores)

Main Problems

- **Bottlenecks**
  - Communication Architecture
  - Memory Access (Overflow, Race Condition, Lost Update)
  - Execution Timings

- **Memory**
  - Buffer Overflow
  - Shared Resource Access Timings

- **Communication**
  - Throughput Reduction
  - Latency

- **Parallelization**
  - Computational (Big Data)
  - Functional (FB → Cores)

Main Problems

- **Safety critical**
  - Monitoring
  - Error Checking
  - Redundancy
  - Realtime Constraints

- **Controlling**
  - External Signals/Sensor Data
  - Control Loop Period

- **Parallelization**
  - Computational (Big Data)
  - Functional (FB → Cores)

Main Problems

- **Bottlenecks**
  - Communication Architecture
  - Memory Access (Overflow, Race Condition, Lost Update)
  - Execution Timings

- **Memory**
  - Buffer Overflow
  - Shared Resource Access Timings

- **Communication**
  - Throughput Reduction
  - Latency

- **Parallelization**
  - Computational (Big Data)
  - Functional (FB → Cores)

Main Problems

- **Safety critical**
  - Monitoring
  - Error Checking
  - Redundancy
  - Realtime Constraints

- **Controlling**
  - External Signals/Sensor Data
  - Control Loop Period

- **Parallelization**
  - Computational (Big Data)
  - Functional (FB → Cores)

Main Problems

- **Bottlenecks**
  - Communication Architecture
  - Memory Access (Overflow, Race Condition, Lost Update)
  - Execution Timings

- **Memory**
  - Buffer Overflow
  - Shared Resource Access Timings

- **Communication**
  - Throughput Reduction
  - Latency

- **Parallelization**
  - Computational (Big Data)
  - Functional (FB → Cores)
System View – Question Catalogue

Main Targets:

- What kind of computations exists?
- What functional safety requirements are necessary (also as hardware modules)?
- Are the sensor-inputs periodical or event-driven?
- Which components shall be parallelized?

Main Problems:

- Where are the bottlenecks?
- Are any realtime deadlines violated?
- Is the communication bandwidth too small?
- Is the shared memory access too high?

What is going on data/information into the system?

- Package size
- Cycle Time
- Quality (Trustworthiness)

How will the software modules be partitioned?

- Are design patterns used for the parallel execution?

What are the external visible result values?

- Dependency from Input time
- Geometrical Decomposition

05.11.2014
Master-Thesis (ASE)
Communication - Parameter

Communication

- Timings
  - Realtime
  - Synchronisation

Realtime
- Hard/Soft Deadlines
- Period Length
- Occurrence of Events

Synchronisation
- Between FB/Modules
- Between Data/Signals/Messages

Architectural
- Communication-Mechanism
- Periodically vs. Event-triggered
- Central Points/Pattern
- Shared Memory vs. Message Passing
- Synchronicity
- Dependencies of Messages

Messages
- Type
- Amount
- Packet Size
- Data-path
- Latency
- Protocol

Runtime
- Throughput
- Occurring of Events
- Bus Load

- Architecture
- Latency
- Bandwidth
- Average Load
- Buffer Size

Bus system
Communication – Question Catalogue

- What are the hard/soft deadlines?
- What are the occurring events?
- Where and at what time are the synchronisation points between the messages and the computations?
- Need the messages/computations wait for each other?

<table>
<thead>
<tr>
<th>Architectural</th>
<th>Messages</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- What are the structure and patterns of the communication?
- What messages are expected?
- What is the communication load on runtime?

<table>
<thead>
<tr>
<th>Runtime</th>
<th>Throughput</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- What is the bandwidth between the components?
- What is the logical and physical link between the components?

<table>
<thead>
<tr>
<th>Buffer size</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
Communication Decision Tree

Scope

Objective

Pattern

Memory/Communication-Pattern
Processor Core - Parameters

Execution
- Runtime (WCET, Average, Dependence on Input)
- Memory Usage (Average, WC)

Properties
- Priority
- Realtime Constraints
- Time and Memory Budget

Dependencies
- Kind of Computation
- Functional (Result)
- Timing
- Structural

Properties
- Kind of Cores
- Main Purpose of Cores
- Connections (Bindings: Shared Memory, Bus)
Processor Core – Question Catalogue

- What are the timing budgets of the FBs?
- What is the runtime behaviour of the FBs?
- What is the main kind of computation for each particular FB?
- Which hardware resources and instruction set are necessary for the particular FB?

- What is the main purpose and specialized kind of computation for each core?
- What is the instruction set and hardware binding for each core?
Computation Decomposition

<table>
<thead>
<tr>
<th>Decomposition</th>
<th>Occurrence</th>
<th>Pattern</th>
<th>Main Purpose</th>
<th>Realtime</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task</td>
<td>Static</td>
<td>Pipeline</td>
<td>Controlling/Functional</td>
<td>Hard</td>
</tr>
<tr>
<td></td>
<td>Regular</td>
<td></td>
<td>Processing</td>
<td>Hard</td>
</tr>
<tr>
<td></td>
<td>Irregular</td>
<td></td>
<td>Signal Handling</td>
<td>?</td>
</tr>
<tr>
<td></td>
<td>Linear</td>
<td>Geometrical Decomposition</td>
<td>Algorithmic/Computational</td>
<td>Soft</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data Flow</td>
<td></td>
<td>Producer/Consumer</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data</td>
<td></td>
<td>Geometrical Decomposition</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Memory – Parameters

Architectural
- Kind of Data (Image, Variables, Sensor-Data, etc.)
- Data Structure (List, Array, etc.)
- Access Mechanism
- Decomposition (Data-flow, Data Structure)

Memory Usage
- Memory Access (Timings, Periodically vs. Event-driven)
- Memory Size
- Expected Waiting Timings and Latencies

- Kind of Memory
- Architecture (Register, Flash, Cache, MMU)
- Memory Size
- Access Timings
- Bindings and Bandwidth
- Average Load

Data

Architecture

Memory Usage

Memory
Memory – Question Catalogue

- What are the kind and structures of the data?
- What is the expected amount of data and memory accesses?
- How will the data be stored and accessed?
  - Memory Access (time, periodically vs. spontaneous)
  - Average Memory Size
  - Access Time
  - Waiting Time

- What are the bindings and access mechanisms of the memory modules?
- What is the throughput and speed of the memory modules?
Data Scope

- Message
- Synchronous
- Shared Memory

Big Data Structures

- Common address space
- Many participants
- Distributed Array

Memory Model
3 Investigation Domains

Communication
- Timings
  - Realtime
  - Synchronisation
- Communication
  - Architecture
  - Runtime

Processor Cores
- Functional execution
  - Properties
  - Execution

Memory
- Data
  - Architecture
  - Memory Usage
- Memory

Bus system
- Processor Cores
- Processor Cores
Backup-Slides
Model & Tests

TESTS

1 2 3 4 5
Execution Time (RT-Period)

• Project 1: Top View
  • 33ms RT-Period @550Mhz $\Rightarrow$ 18150k Cycles Execution-time
    • Bottleneck: Memory Access (Waiting) Time

• Project 3: Battery Management System
  • 5ms (Simulink) + 10/100/1000ms RT-Period @200Mhz
    $\Rightarrow$ 1000k Cycles Execution Time
    • Bottleneck: 70Hz of Voltage measuring

• Project 2: Power Electronics
  • 100 $\mu$s RT-Period @200Mhz $\Rightarrow$ 20k Cycles Execution-time
    • Bottleneck: Execution Time
      • 70% PWM + 5% OS + 10% A/D-time + 30-40% Periphery
        + 70-80% Monitoring = 195% (runs on 2 cores)

Ratio = (1000:1)
Communication-Time

200Mhz Clock-Cycle:

- CAN-Bus
  - Baud-Rate: 500k bps (200m)
  - Net: **25,974 kByte/s** (max @ 8Bytes/extended-Message)
  - Latency > 308µs = **61600 Cycles**

- UART
  - Baud-Rate: 115 200 bps
  - Net: **6,4 kByte/s** (4 Bytes Data + 2 Bytes Checksum)
  - Latency > 625 µs = **125000 Cycle**

- SPI
  - 10Mbit/s
  - Net: **1,25 MByte/s**
  - Latency > 3,2 µs = **640 Cycles**

- Internal Peripheral-BUS System (between Cores)
  - 200Mhz/2 Clock-Cycle @ 32Bit
  - Net: **2,5 GByte/s**
  - Latency = 0,01µs = **2 Cycles**

**Ratio = (60 000:1)**
Memory Access Time

• 660 Mhz Shared Memory
  • Cache with 64 Bits (no usage at storage)
  • Stores 16Bit(2Bytes) in 1 Cycle +1 extra Cycle new line access (assumption)
  •

• Storage for a value
  ➞ 1 Word (Int) = 4 Bytes ➞ 2 + 1 Cycle
  ➞ 4,545ns (0,00454µs)

• Storages of a picture
  ➞ 1280*800*3Byte ➞ (3Bytes/2)*1280*800 Cycles
  ➞ 2,327ms (2327,3µs)

Ratio = (500 000:1)
Test-Procedure

- Functional Model
- ECU Target TMDXEV8148

For each Domain
- Communication (Ethernet)
- Processor Cores (Execution Timings)
- Memory (Shared Memory Access)

- Definition of Testcases
- Implementation of Testcases
- Exposition of Results

- Estimation Validation
- Process of FB Testing
- Detect Possible Bottlenecks
TI Vision MID EVE
<table>
<thead>
<tr>
<th>T</th>
<th>Testcase Description</th>
<th>Goals/Benefits for Scientific Work</th>
<th>Options</th>
<th>Expected Results</th>
<th>Expectation [Simulated]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Load Images (just in time over Ethernet)</td>
<td>Ground zero example and tests for external communication interfaces.</td>
<td>minor: (De)Activation of caches o (De)Activation of DMA</td>
<td>External communication interface speed is comparable slow (estimated 30 Mbit/s (\rightarrow) for each image 273ms + 86ms only for communication). Execution &gt; 273 + 86ms + x (\approx) 370ms</td>
<td>Execution: Access time: Mem-copy &lt; Word-wise &lt; Byte-wise</td>
</tr>
<tr>
<td>2</td>
<td>Memory access with different strategies 1. For-Loop byte-wise 2. For-Loop half-word/word-wise 3. Use MemCopy</td>
<td>This shows that the implementation and the usage of predefined support of memory access can make a huge difference.</td>
<td>o (De)Activation of caches o Data packets &gt;&gt; comm.-alignments o (De)Activation of DMA</td>
<td>The Caching-Effects may make a huge difference in the reading of the memory. Access time: Mem-copy &lt; Word-wise &lt; Byte-wise</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Latency and Delay of Communication 1. IPC over NoC (Send signal (1 Byte message between 2 processor cores)) 2. Use shared memory to communicate</td>
<td>This helps to get a magnitude of communication latencies.</td>
<td>o (De)Activation of caches o Data packets &gt;&gt; comm.-alignments minor: o (De)Activation of DMA</td>
<td>The runtime execution differs because “image-processing” has a different execution (and HW support on the different processor cores). Runtime: DSP &lt; ARM &lt; EVE [100\ us - 1\ ms] (\leq) 10 cycles</td>
<td>Run-time: TC6 &gt; TC5</td>
</tr>
<tr>
<td>4</td>
<td>Implement down-sampling FB for DSP, ARM, EVE (to test same Alg. on different cores)</td>
<td>Test the behaviour (runtime execution time) of the same algorithm on different kind of cores.</td>
<td>minor: (De)activation of caches o (De)activation of DMA</td>
<td>The Ethernet connection is the slowest part, so a huge speedup is expected. Runtime: (&lt; n*(273+86\ ms)) (\leq) 10ms (\approx) 10ms</td>
<td>Execution &lt; n*(273+86 ms) (\leq) 10ms</td>
</tr>
<tr>
<td>5</td>
<td>Load n images and execute in a row</td>
<td>Get an example of end-to-end time without external communication latency.</td>
<td>minor: (De)activation of caches o (De)activation of DMA</td>
<td>Probably a small loss of end-to-end execution timings of a single run, but a total speedup of the sum of executions, because of parallel execution. Runtime: TC6 &gt; TC5</td>
<td>Run-time: TC6 &gt; TC5</td>
</tr>
<tr>
<td>6</td>
<td>Load n images Fill pipeline and execute in parallel</td>
<td>See how the execution is influences (compared to TC5) through parallel executions.</td>
<td>minor: Data packets &gt;&gt; comm.-alignments o (De)Activation of caches o (De)Activation of DMA</td>
<td>The memory access could take longer, because of IPC communication over the shared memory. Also, caching effects could influence the timings a lot. Runtime: TC7 &gt; TC6</td>
<td>Run-time: TC7 &gt; TC6</td>
</tr>
<tr>
<td>7</td>
<td>TC6 + interrupts through IPC/global variables (exchange via shared memory)</td>
<td>See behaviour, if massive data stream is interrupted by comparable small communication/shared memory accesses.</td>
<td>o (De)Activation of caches minor: o Data packets &gt;&gt; comm.-alignments o (De)Activation of DMA</td>
<td>A context switch should only take some cycles. Context-switch: &lt; 1000 cycles</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Measure a simple context-switch of two task on the same processor core</td>
<td>This will show if context-switches need to be considered at estimation time or if they are to less in resource usage</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## General Testing Results

### Δ-Timings

<table>
<thead>
<tr>
<th>Processor Cores (Execution Timings)</th>
<th>Min %</th>
<th>Max %</th>
</tr>
</thead>
<tbody>
<tr>
<td>Communication (Ethernet)</td>
<td>33</td>
<td>13975</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Memory (Shared Memory Access)</th>
<th>Min %</th>
<th>Max %</th>
</tr>
</thead>
<tbody>
<tr>
<td>Communication (Ethernet)</td>
<td>2</td>
<td>38</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Estimation vs. Measuring</th>
<th>Min %</th>
<th>Max %</th>
</tr>
</thead>
<tbody>
<tr>
<td>Communication (Ethernet)</td>
<td>47</td>
<td>64408</td>
</tr>
<tr>
<td>Communication (Ethernet)</td>
<td>-5</td>
<td>228</td>
</tr>
<tr>
<td>Communication (Ethernet)</td>
<td>-55</td>
<td>2506</td>
</tr>
</tbody>
</table>
General Testing Results

Difference: Min. – Max. Timings
• Processor Cores Execution: 0,0 – 2,2 %
• Memory Access: 2,1 – 38,1 %
• External Communication: 33,1 – 13974,6 %

Difference: Max. Estimation to Measuring
• Processor Cores Execution: 47 – 64408 %
• Memory Access: -5 – 228 %
• External Communication: -55 – 2506 %
Abstraction

\[ \text{MinCycles}_{FB} = \frac{\text{MinPeriod}_{FB} \times f}{\#FB \times C} \]

\[ \text{MinCycles}_{FB} = \frac{33333 \, \mu s \times 200 \, MHz}{5 \times 100} \approx 13333 \]

\[ \text{MinCycles}_{FB}(C = 100) \]
## Test-Results

<table>
<thead>
<tr>
<th>Decomposition</th>
<th>Specifics</th>
<th>Bottleneck</th>
<th>Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.) Data Flow</td>
<td>Short realtime-periods</td>
<td>Processor execution time</td>
<td>1000:1 Realtime-Periods</td>
</tr>
<tr>
<td>2.) Function/Task</td>
<td>Heterogeneous cores</td>
<td>Mapping + Synchronization + Communication</td>
<td>30 000:1 (Intra- vs. Inter-ECU)</td>
</tr>
<tr>
<td>3.) Data</td>
<td>Shared memory</td>
<td>Memory access (waiting) time</td>
<td>500 000:1 (Integer vs. Image)</td>
</tr>
</tbody>
</table>
Memory Accesses

- Write Byte-wise
- Write Word-wise
- Write via Memcopy
- Read Byte-wise
- Read Word-wise
- Read via MemCopy
- Writing Integer
- Writing Long
- Writing Long Long
- Writing Long Double
- Reading Integer
- Reading Long
- Reading Long Long
- Reading Long Double
# FB Measurements

<table>
<thead>
<tr>
<th>Processor Type (MHz)</th>
<th>Task Name</th>
<th>Execution Cycles</th>
<th>Measured</th>
<th>Diff: Min-Max in %</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARM A8 (600)</td>
<td>Receive Camera Image</td>
<td>Cycles</td>
<td>T vs. Δt</td>
<td>Min</td>
</tr>
<tr>
<td></td>
<td>T1_R1_ReceiveCameraImage_A8</td>
<td>μs</td>
<td>30335808</td>
<td>4269643469</td>
</tr>
<tr>
<td></td>
<td>Image Transfer to Store</td>
<td>Cycles</td>
<td>1451351</td>
<td>1483916</td>
</tr>
<tr>
<td></td>
<td>T1_R2_TransferImageToStore_A8</td>
<td>μs</td>
<td>2419</td>
<td>2473</td>
</tr>
<tr>
<td></td>
<td>Store Image</td>
<td>Cycles</td>
<td>1444658</td>
<td>1474529</td>
</tr>
<tr>
<td></td>
<td>T1_R3_StoreImage_SM</td>
<td>μs</td>
<td>2408</td>
<td>2458</td>
</tr>
<tr>
<td>DSP (500)</td>
<td>Manipulate Image</td>
<td>Cycles</td>
<td>233606937</td>
<td>322541557</td>
</tr>
<tr>
<td></td>
<td>T2_R1_ManipulateImage_DSP</td>
<td>μs</td>
<td>467214</td>
<td>645083</td>
</tr>
<tr>
<td></td>
<td>Write on Image</td>
<td>Cycles</td>
<td>701391</td>
<td>702617</td>
</tr>
<tr>
<td></td>
<td>T2_R2_WriteOnImage_SM</td>
<td>μs</td>
<td>1403</td>
<td>1405</td>
</tr>
<tr>
<td>EVE-ARP (250)</td>
<td>Load Image</td>
<td>Cycles</td>
<td>538880</td>
<td>571520</td>
</tr>
<tr>
<td></td>
<td>T3_R1_LoadImage_SM</td>
<td>μs</td>
<td>2156</td>
<td>2286</td>
</tr>
<tr>
<td></td>
<td>Image Downsizing</td>
<td>Cycles</td>
<td>293411</td>
<td>294258</td>
</tr>
<tr>
<td></td>
<td>T3_R2_ImageDownsizing_EVE</td>
<td>μs</td>
<td>1174</td>
<td>1177</td>
</tr>
<tr>
<td></td>
<td>Store Mini Image</td>
<td>Cycles</td>
<td>134720</td>
<td>142880</td>
</tr>
<tr>
<td></td>
<td>T3_R3_StoreMinImage_SM</td>
<td>μs</td>
<td>539</td>
<td>572</td>
</tr>
<tr>
<td>DSP (500)</td>
<td>Image Transfer</td>
<td>Cycles</td>
<td>699431</td>
<td>699431</td>
</tr>
<tr>
<td></td>
<td>T4_R1_ImageTransfer_DSP</td>
<td>μs</td>
<td>1399</td>
<td>1399</td>
</tr>
<tr>
<td>ARM A8 (600)</td>
<td>Load Mini Image</td>
<td>Cycles</td>
<td>1444658</td>
<td>1474529</td>
</tr>
<tr>
<td></td>
<td>T5_R1_LoadMiniImage_SM</td>
<td>μs</td>
<td>2408</td>
<td>2458</td>
</tr>
<tr>
<td></td>
<td>Image Transfer to Send</td>
<td>Cycles</td>
<td>15338124</td>
<td>21366315</td>
</tr>
<tr>
<td></td>
<td>T5_R2_ImageTransferToSend_A8</td>
<td>μs</td>
<td>25564</td>
<td>35611</td>
</tr>
<tr>
<td></td>
<td>Send Image to Client</td>
<td>Cycles</td>
<td>13754623</td>
<td>18308939</td>
</tr>
<tr>
<td></td>
<td>T5_R3_SendImageToClient_A8</td>
<td>μs</td>
<td>22924</td>
<td>30515</td>
</tr>
</tbody>
</table>
# FB Estimations

<table>
<thead>
<tr>
<th>Processor</th>
<th>FB/Execution Cycles</th>
<th>Estimation</th>
<th>Diff: Measuring to Estimation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ARM A8</strong>&lt;br&gt;(600)</td>
<td><strong>Receive Camera Image</strong>&lt;br&gt;<code>T1_R1_ReceiveCameraImage_A8</code>&lt;br&gt;Cycles: 163840200 - 163840200&lt;br&gt;µs: 273067 - 273067</td>
<td>Min: 163840200&lt;br&gt;Max: 163840200&lt;br&gt;Min in %: -81&lt;br&gt;Max in %: 2506</td>
<td>Measurement: -96</td>
</tr>
<tr>
<td><strong>Image Transfer to Store</strong>&lt;br&gt;<code>T1_R2_TransferImageToFile_A8</code>&lt;br&gt;Cycles: 6000 - 6000&lt;br&gt;µs: 10 - 100</td>
<td>Min: 6000&lt;br&gt;Max: 6000&lt;br&gt;Min in %: -100&lt;br&gt;Max in %: -100</td>
<td>Measurement: -96</td>
<td></td>
</tr>
<tr>
<td><strong>Store Image</strong>&lt;br&gt;<code>T1_R3_StoreImage_SM</code>&lt;br&gt;Cycles: 600000 - 1800000&lt;br&gt;µs: 1000 - 3000</td>
<td>Min: 600000&lt;br&gt;Max: 1800000&lt;br&gt;Min in %: 141&lt;br&gt;Max in %: -18</td>
<td>Measurement: 22</td>
<td></td>
</tr>
<tr>
<td><strong>DSP</strong>&lt;br&gt;(500)</td>
<td><strong>Manipulate Image</strong>&lt;br&gt;<code>T2_R1_ManipulateImage_DSP</code>&lt;br&gt;Cycles: 50000 - 50000&lt;br&gt;µs: 100 - 1000</td>
<td>Min: 50000&lt;br&gt;Max: 50000&lt;br&gt;Min in %: -99&lt;br&gt;Max in %: -93</td>
<td>Measurement: -100</td>
</tr>
<tr>
<td><strong>Write on Image</strong>&lt;br&gt;<code>T2_R2_WriteOnImage_SM</code>&lt;br&gt;Cycles: 5000 - 5000&lt;br&gt;µs: 10 - 100</td>
<td>Min: 5000&lt;br&gt;Max: 5000&lt;br&gt;Min in %: -100&lt;br&gt;Max in %: -100</td>
<td>Measurement: -99</td>
<td></td>
</tr>
<tr>
<td><strong>EVE-ARP</strong>&lt;br&gt;(250)</td>
<td><strong>Load Image</strong>&lt;br&gt;<code>T3_R1_LoadImage_SM</code>&lt;br&gt;Cycles: 200000 - 600000&lt;br&gt;µs: 800 - 2400</td>
<td>Min: 200000&lt;br&gt;Max: 600000&lt;br&gt;Min in %: 169&lt;br&gt;Max in %: -5</td>
<td>Estimation: -63</td>
</tr>
<tr>
<td><strong>Image Downsizing</strong>&lt;br&gt;<code>T3_R2_ImageDownsizing_EVE</code>&lt;br&gt;Cycles: 20000 - 200000&lt;br&gt;µs: 80 - 800</td>
<td>Min: 20000&lt;br&gt;Max: 200000&lt;br&gt;Min in %: -93&lt;br&gt;Max in %: -32</td>
<td>Measurement: -32</td>
<td></td>
</tr>
<tr>
<td><strong>Store Mini Image</strong>&lt;br&gt;<code>T3_R3_StoreMiniImage_SM</code>&lt;br&gt;Cycles: 50000 - 150000&lt;br&gt;µs: 200 - 600</td>
<td>Min: 50000&lt;br&gt;Max: 150000&lt;br&gt;Min in %: 169&lt;br&gt;Max in %: -5</td>
<td>Measurement: -63</td>
<td></td>
</tr>
<tr>
<td><strong>DSP</strong>&lt;br&gt;(500)</td>
<td><strong>Image Transfer</strong>&lt;br&gt;<code>T4_R1_ImageTransfer_DSP</code>&lt;br&gt;Cycles: 5000 - 5000&lt;br&gt;µs: 10 - 100</td>
<td>Min: 5000&lt;br&gt;Max: 5000&lt;br&gt;Min in %: -99&lt;br&gt;Max in %: -93</td>
<td>Measurement: -99</td>
</tr>
<tr>
<td><strong>ARM A8</strong>&lt;br&gt;(600)</td>
<td><strong>Load Mini Image</strong>&lt;br&gt;<code>T5_R1_LoadMiniImage_SM</code>&lt;br&gt;Cycles: 1500000 - 4500000&lt;br&gt;µs: 250 - 750</td>
<td>Min: 1500000&lt;br&gt;Max: 4500000&lt;br&gt;Min in %: 863&lt;br&gt;Max in %: 228</td>
<td>Measurement: -90</td>
</tr>
<tr>
<td><strong>Image Transfer to Send</strong>&lt;br&gt;<code>T5_R2_ImageTransferToSend_A8</code>&lt;br&gt;Cycles: 6000 - 60000&lt;br&gt;µs: 10 - 100</td>
<td>Min: 6000&lt;br&gt;Max: 60000&lt;br&gt;Min in %: -100&lt;br&gt;Max in %: -100</td>
<td>Measurement: -100</td>
<td></td>
</tr>
<tr>
<td><strong>Send Image to Client</strong>&lt;br&gt;<code>T5_R3_SendImageToClient_A8</code>&lt;br&gt;Cycles: 40960000 - 40960000&lt;br&gt;µs: 68267 - 68267</td>
<td>Min: 40960000&lt;br&gt;Max: 40960000&lt;br&gt;Min in %: -66&lt;br&gt;Max in %: -55</td>
<td>Estimation: 124</td>
<td></td>
</tr>
</tbody>
</table>
# IPC-Communication Test

<table>
<thead>
<tr>
<th>IPC-Communication</th>
<th>between ARM Cortex A8 and DSP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cycles</td>
<td>µs</td>
</tr>
<tr>
<td>4229</td>
<td>7,05</td>
</tr>
<tr>
<td>3226</td>
<td>5,38</td>
</tr>
<tr>
<td>2009</td>
<td>3,35</td>
</tr>
<tr>
<td>1933</td>
<td>3,22</td>
</tr>
<tr>
<td>2652</td>
<td>4,42</td>
</tr>
<tr>
<td>2002</td>
<td>3,34</td>
</tr>
<tr>
<td>2631</td>
<td>4,39</td>
</tr>
<tr>
<td>2040</td>
<td>3,40</td>
</tr>
<tr>
<td>2509</td>
<td>4,18</td>
</tr>
<tr>
<td>AVG</td>
<td>3,87</td>
</tr>
<tr>
<td>MIN</td>
<td>3,22</td>
</tr>
<tr>
<td>MAX</td>
<td>7,05</td>
</tr>
<tr>
<td>Count</td>
<td>10</td>
</tr>
</tbody>
</table>

05.11.2014 Master-Thesis (ASE)
Backup-Slides
Estimation-Procedure

PROCEDURE
Estimation-Procedure

Input
- Project-Analysis

Modelling
- Functions-Model
- Possible ECU Target
- Architecture-Graph
- Domain specific Decision-Guide
- Software-Architecture (FB-Mapping)
- Modelling in Simulator
- Timing-Constraints
- Plausibility Validation
- Validated Software-Architecture

Verification
- Traces of previous projects
- Pre-Implemented FB
- Requirements
- Timing-Estimations
- Timing-Constraints
- Scheduling
- Mapping
- Possible Bottlenecks
- Test-Approaches

Results

Motivation • Topic • Basics • Multicore • Model • Procedure • Results
Alternative: “Correctness-by Construction”
Backup-Slides
Example Application (OCC)

EXAMPLE

05.11.2014
Master-Thesis (ASE)
91
www.tu-chemnitz.de
Results

Online Camera Calibration

Cameras * Tiles * Period * Images * Optimisations = Total Time

4 * 2 * 100ms * 100 * 50 = 4000 s
1 * 1 * 33,333ms * 100 * 50 = 167 s

24x faster
Software-Structure

**DSP**
- Find Reference Points
- Egomotion Estimation
- Load Tile
- ReFind Points
- Compare Values
- Load Speed

**M4**
- Receive Camera Image
- Store Image

**ARM**
- Calculate Camera Position
- Minimize Positioning Error
- Receiving CAN-Data
- Store Speed Values

**Video Client**
- Load Image
- Dewarp Image
- Send to HMI
OCC-Timing Estimations

- DSP
  - 40 µs
  - Traces of previous projects
  - ISS-Simulation
  - 4,136 µs
  - Calculated Timing-Estimation
  - 160 µs
  - Traces of previous projects
  - Expert Timing-Estimation
  - 0,4 µs

- M4
  - 4* 15360 µs
  - Example-Implementation
  - 4* 2327,27 µs
  - Calculated Timing-Estimation

- VC
  - 2327,27 µs
  - Calculated Timing-Estimation
  - 1,876 µs
  - Calculated Timing-Estimation
  - 2881,80 µs
  - Calculated Timing-Estimation

- ARM
  - 0,533 µs
  - Expert Timing-Estimation
  - 166,67 µs
  - Traces of previous projects
  - 0,01 µs
  - Calculated Timing-Estimation
  - 0,018 µs
  - Calculated Timing-Estimation
  - 0,4 µs
  - Calculated Timing-Estimation

Example of implementation and calculated timing estimation based on previous projects.
Software Architecture-Graph

Motivation • Topic • Basics • Multicore • Model • Procedure • Results

- Load Tile
  - Find Reference Points
    - Load speed
  - Egomotion Estimation
    - Implicit
    - Ref Point
  - Load Tile
    - Tiles
    - Refine Points
    - vector
  - Compare Values
    - vector
    - Calculate Camera Position
      - Cam Pos
      - Minimize Positioning Error

- Implicit: Done in previous period

- Load Image
  - Dewarp Image
    - image
    - Send to HMI
  - Receive Camera Image
    - image
    - Store Image
  - Receive CAN-Data
    - Speed
    - Store Speed values

05.11.2014 Master-Thesis (ASE) www.tu-chemnitz.de
OCC System Structure

Motivation • Topic • Basics • Multicore • Model • Procedure • Results

- Load Speed
- Find Reference Points
- Load Tile
- ReFind Points
- Receiving CAN-Data
- Receive Camera Image
- Load Image

- Speed
- Ref-Points
- Delay
- vector
- image
- image

- Compare Values
- Calculate Camera Position
- Minimize Position Error

- Store Speed Values
- Store Image
- Dewarp Image
- Send to HMI

- DSP
- EVE
- ARM A8
- Cortex M3
- Display
- Local Memory1
- Local Memory2
- Local Memory3
- Local Memory4

Shared Memory
L3 RAM

Bus

Delay
OCC System Structure

- Load Speed
- Find Reference Points
- Load Tile
- ReFind Points
- Receive CAN-Data
- Receive Camera Image
- Load Image
- Speed
- Ref-Points
- Delay
- Vector
- Compare Values
- Store Speed values
- Store Image
- Image
- Image
- Speed
- Load Image
- Image
- Dewarp Image
- Send to HMI
- Minimize Position Error
- Calculate Camera Position
- Compare Camera Position
- Minimize Position Error
- Speed
- Ref-Points
- Load Image
- Load Speed
- Find Reference Points
- Local Memory1
- Local Memory2
- Local Memory3
- Local Memory4
- DSP
- ARM A8
- Cortex M3
- Cortex M3
- Display
- EVE
- Send to HMI
- Video-In
- Video-In
- L3 RAM
- Bus
Modelling Software Structure

Communication Graph

Runnable Sequence

Path-Graph
OCC-Modellierung: PRECISION PRO
Optimisation
Software Structure
Optimisation Formal Verification
INCHRON chronVAL WCRT

Motivation • Topic • Model • Procedure • Application • Simulation • Results
Software Structure

AMALTHEA Runnable Sequence

SymTA/S Path-Graph

05.11.2014 Master-Thesis (ASE) www.tu-chemnitz.de
Software Structure

- **AMALTHEA Runnable Sequence**
- **SymTA/S Path-Graph**
- **TA Communication Graph**

05.11.2014

Master-Thesis (ASE) 105
Mapping

SymTA/S TaskGraph

INCHRON SystemChart
Optimisation 1
Software Structure

100ms

Load Tile
Load Reads
Find Reference Points
Calculate Nominal Value

100ms

Load Tile
Refine Points
Compare Values

Path: DSP 1 5 LoadTile -> Finish of DSP 1 7 CompareValues

BCRT 169.4 µs
20 ns

WCRT 99.31 ms
160 ns
Optimisation 1
Software Structure
Architecture-Graph (Vision MID EVE)
Architecture-Graph (Superset C66x ISA)
Architecture-Graph (Full TDA2x Superset)
Backup-Slides
Simulation (OCC) + Tools

SIMULATION

1 2 3 4 5
Realtime Constraints

TA Tool Suite Event Chains
System Analysis

Critical Path

⇒ Problem if Non-predictable memory Accesses
Simulation Results

5 Tools

- Fraunhofer FOKUS PRECISION PRO
- INCHRON
- SYMTA VISION
- AMALTHEA
- TA Timing Architects
Simulation Results
System Analysis

- INCHRON Load-Balancing
- SymTA/S Load Bar Chart
- TA Tool Suite Load Diagram

Motivation • Topic • Basics • Multicore • Model • Procedure • Results
<table>
<thead>
<tr>
<th>Phase</th>
<th>PRECISION PRO</th>
<th>INCHRON</th>
<th>SymTA/S</th>
<th>Timing Architects</th>
<th>AMAL-THEA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Partitioning &amp; Allocation</td>
<td>+</td>
<td>o</td>
<td>o1</td>
<td>+2</td>
<td>+3</td>
</tr>
<tr>
<td>Formal Validation</td>
<td>o</td>
<td>++</td>
<td>+</td>
<td>o</td>
<td>o</td>
</tr>
<tr>
<td>Simulation</td>
<td>o4</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>-5</td>
</tr>
<tr>
<td>Scheduling View/Representation</td>
<td>o4</td>
<td>++</td>
<td>+</td>
<td>++</td>
<td>-5</td>
</tr>
<tr>
<td>Timing Analysis</td>
<td>o</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>o</td>
</tr>
<tr>
<td>Optimization</td>
<td>o</td>
<td>o</td>
<td>o6</td>
<td>++</td>
<td>o</td>
</tr>
<tr>
<td>Re-Feed Traces</td>
<td>o</td>
<td>o</td>
<td>o7</td>
<td>o7</td>
<td>o</td>
</tr>
<tr>
<td>Model In/Export (e.g. AUTOSAR)</td>
<td>o</td>
<td>+</td>
<td>o</td>
<td>+</td>
<td>+8</td>
</tr>
<tr>
<td>Tasks</td>
<td>PRECISION PRO</td>
<td>INCHRON</td>
<td>SymTA/S</td>
<td>Timing Architects</td>
<td>AMAL-THEA</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------</td>
<td>---------</td>
<td>---------</td>
<td>------------------</td>
<td>-----------</td>
</tr>
<tr>
<td>Model HW</td>
<td>+</td>
<td>+</td>
<td>++</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Model SW</td>
<td>+</td>
<td>++</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Model Timings</td>
<td></td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Model Tsys</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Model OCC</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Update ECU</td>
<td></td>
<td>+ +</td>
<td></td>
<td>1</td>
<td>+</td>
</tr>
<tr>
<td>Compare Versions</td>
<td></td>
<td></td>
<td>2</td>
<td></td>
<td>+</td>
</tr>
<tr>
<td>Add OCC + Tsys</td>
<td></td>
<td>+ +</td>
<td></td>
<td>1</td>
<td>3, 4</td>
</tr>
<tr>
<td>Parameter</td>
<td>PRECISION PRO</td>
<td>INCHRON</td>
<td>SymTA/S</td>
<td>Timing Architects</td>
<td>AMAL-THEA</td>
</tr>
<tr>
<td>-------------------------</td>
<td>---------------</td>
<td>---------</td>
<td>---------</td>
<td>-------------------</td>
<td>-----------</td>
</tr>
<tr>
<td>Academic Licence</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>-</td>
<td>++</td>
</tr>
<tr>
<td>Company Support</td>
<td>+</td>
<td>++</td>
<td>++</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Scientific Background</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>+</td>
<td>++</td>
</tr>
<tr>
<td>Documentation</td>
<td>-</td>
<td>+</td>
<td>++</td>
<td>+</td>
<td>++</td>
</tr>
<tr>
<td>Model Coverage</td>
<td>+</td>
<td>+</td>
<td>++</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Multicore Focus</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>++</td>
</tr>
<tr>
<td>Usability</td>
<td></td>
<td>++</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Training time</td>
<td>+</td>
<td>++</td>
<td>+</td>
<td>+</td>
<td></td>
</tr>
<tr>
<td>Operability</td>
<td>+ 1</td>
<td>++</td>
<td>+</td>
<td>+</td>
<td></td>
</tr>
<tr>
<td>Performance</td>
<td></td>
<td>++</td>
<td>+ 2</td>
<td>+ 3</td>
<td></td>
</tr>
</tbody>
</table>
Backup-Slides
Results

1 2 3 4 5
END