Designing Mixed Criticality Applications on Modern Heterogeneous MPSoC Platforms

Giovani Gracioli, Rohan Tabish, Renato Mancuso, Reza Mirosanlou, Rodolfo Pellizzoni, and Marco Caccamo

Technical University of Munich
University of Illinois at Urbana-Champaign
University of Waterloo
Boston University

ECRTS 2019 @ Stuttgart
Introduction
Multicore Processors

Contention!

sources of unpredictability
Multiprocessor System-on-a-Chip (MPSoC)

Define PL components to mitigate the non-determinism in traditional multicore CPUs

Ideal trade-off between processing power and RT guarantees
Contributions

High-performance and time-sensitive applications to co-exist under strict temporal isolation

Set of SW and HW Techniques:
- Hypervisor with coloring & code relocation,
- PL-side SPM,
- Variable TDMA slot size

Full-stack implementation on one of the latest-generation MPSoC

Hardware IP to prevent the problem of memory waste when cache coloring is used
System Model

Non-criticality Domain
- OS + Applications
- core 1
- \ldots
- core K
- Shared LLC
- LLC Partition 1
- DRAM + PL SPM
- Interconnect 1
- Local Memory 1 (DRAM)

Mid-criticality Domain
- RTOS + RT Tasks
- core K+1
- \ldots
- core H
- Interconnect 2
- Local Memory 2 (SPM)

High-criticality Domain
- RTOS + RT Tasks
- core H+1
- \ldots
- core C
- Interconnect 3
- Local Memory 3 (SPM)
Background

3-Phase Task Execution Model

3 stages, 2 resources

\[ \tau_i = \{L_i, C_i, U_i\} \]
Pipelining and Memory Bus Scheduling

Memory operations can be parallelized with respect to execution.

**PIPELINING**

Memory operations can be parallelized with respect to execution.

**Partition in space**

**Partition in time (TDMA)**

**Scratchpad**

**DMA slot size**

\[ \sigma \geq \max_i (L_i, U_i) \]
# Chosen Platform

<table>
<thead>
<tr>
<th>Feature</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CPU units</strong></td>
<td>4x A53 1.2 Ghz, 2x R5 600 Mhz</td>
</tr>
<tr>
<td><strong>A53 Memories</strong></td>
<td>32KB private I/D caches, 1MB LLC</td>
</tr>
<tr>
<td><strong>R5 Memories</strong></td>
<td>32KB private I/D caches, 128KB TCM</td>
</tr>
<tr>
<td><strong>PS-PL interfaces</strong></td>
<td>2x HPM, 1x LPD (PS→ PL) 2x HPC, 4x HP (PL→ PS)</td>
</tr>
<tr>
<td><strong>Memories</strong></td>
<td>DDR 4GB 64-bit (PS), OCM 256KB (PS) DDR 512MB 16-bit (PL), BRAM 3MB (PL)</td>
</tr>
</tbody>
</table>

**Diagram:**

- **Xilinx Ultrascale+ ZCU102**
- **PB Switches**
- **SD Card Slot**
- **2x Pmod I/O + I2C**
- **PCIe® Gen 2x4 slot (4 x GTR)**
- **CAN Header**
- **SysMon**
- **PM Bus**
- **JTAG**
- **DisplayPort (2 x GTR)**
- **USB UART**
- **USB JTAG**
- **12 Volt Power**
- **PM Bus**
Design Space Exploration

- MPSOcs allow many possible designs

Where to execute tasks?
Where to implement the commu. engine?
Which main memory?
SPM memory?
How to handle PS-PL communi.?
Proposed Design

Jailhouse Hypervisor

- R5 Core
- DMA
- Cache
- DRAM

- SPM 1
- SPM 2
- SPM 3

Legend:
- Green = High-Performance PS-to-PL Interfaces
- Blue = Low-Power Domain PS-to-PL Interface
Coloring Revisited

From cache controller’s perspective:
- Tag
- Index
- Offset

From OS’s perspective:
- Page Frame Number
- Offset
- Color

Physical Pages:
- N - 1
- .
- .
- .
Address Translator

Address Translator is transparent

A53
0xA0023456 (8MB address space)

PS-PL Interface (HPM)

Translator

Programmable Logic

Coloring bits (14-15) are dropped

Core 0

Core 3

0xA00B456

0xA0000000

0xA1FFFFFF

0x10000

SPM
Designs Evaluation

- We performed an experimental evaluation to evaluate the created designs. We used:
  - Two benchmarks from San Diego Visual Benchmark Suite (SD-VBS) – disparity and mser
  - Bandwidth benchmark (BW) to stress the memory subsystem
  - Main memory (DRAM in PS) and SPM (BRAM in PL)
    - PS-DRAM is faster than PL-DRAM
  - DMA on the PS-side
    - DMA on the PS-side is also faster than PL-side DMA
- Predictability for mixed criticality applications
Designs Evaluation

- We consider the following execution scenarios:

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Experiment</th>
<th>Accessed Memory</th>
<th>Coloring</th>
<th>PS-PL Interface</th>
<th>Contention Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCY-SOLO</td>
<td>Solo</td>
<td>PS DRAM</td>
<td>No</td>
<td>Not used</td>
<td>None</td>
</tr>
<tr>
<td>LCY-STRESS</td>
<td>Contention</td>
<td>PS DRAM</td>
<td>No</td>
<td>Not used</td>
<td>3x BW</td>
</tr>
<tr>
<td>OUR-SOLO</td>
<td>Solo</td>
<td>SPM</td>
<td>Yes</td>
<td>Dedicated</td>
<td>None</td>
</tr>
<tr>
<td>OUR-HIGH</td>
<td>Contention</td>
<td>SPM</td>
<td>Yes</td>
<td>Dedicated</td>
<td>1x BW from low-crit. 2x BW from mid-crit.</td>
</tr>
<tr>
<td>OUR-MID</td>
<td>Contention</td>
<td>SPM</td>
<td>Yes</td>
<td>Shared</td>
<td>1x BW from low-crit. 1x BW from mid-crit. 1x BW from high-crit.</td>
</tr>
</tbody>
</table>
Results for mser

(a) LCY-SOLO

(b) LCY-STRESS

Avg. = 4.84E + 05
WCET = 5.05E + 05
BCET = 4.82E + 05
Var. Win. = 4.68%

Avg. = 6.65E + 05
WCET = 8.72E + 05
BCET = 6.30E + 05
Var. Win. = 27.82%
Mser: LCY-SOLO vs. OUR-HIGH

(a) LCY-SOLO

(b) OUR-HIGH

Average = 4.84E+05
WCET = 5.05E+05
BCET = 4.82E+05
Var. Win. = 4.68%

Average = 4.92E+05
WCET = 5.50E+05
BCET = 4.85E+05
Var. Win. = 11.72%
DMA Evaluation

- DMA transfer time
  - Different data sizes
  - 1000 repetitions
  - AVG, STD, WCET
  - Programming overhead
DMA Evaluation

<table>
<thead>
<tr>
<th>Transfer Size</th>
<th>Transfer Time</th>
<th>Bandwidth (MB/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Average (μs)</td>
<td>STD</td>
</tr>
<tr>
<td>2 KB</td>
<td>4.92</td>
<td>0.057</td>
</tr>
<tr>
<td>4 KB</td>
<td>7.15</td>
<td>0.04</td>
</tr>
<tr>
<td>8 KB</td>
<td>11.63</td>
<td>0.01</td>
</tr>
<tr>
<td>9.1 KB</td>
<td>12.91</td>
<td>0.05</td>
</tr>
<tr>
<td>16 KB</td>
<td>20.62</td>
<td>0.08</td>
</tr>
<tr>
<td>22 KB</td>
<td>27.42</td>
<td>0.10</td>
</tr>
<tr>
<td>32 KB</td>
<td>38.52</td>
<td>0.05</td>
</tr>
<tr>
<td>1 MB</td>
<td>1149.44</td>
<td>0.05</td>
</tr>
</tbody>
</table>

- STD within range [0.057, 0.1]
- Programming overhead: **3.89 us**
- Programming overhead vs. small data size transfers
- Model behaves well as long as task execution times are longer than the time required to reload an SPM partition
Case study

- Videos frames captured from a camera are processed in a high-criticality domain

- Disparity: obtains relative positions of objects
  - Useful for cruise control, pedestrian tracking, and collision control

- Demonstrate how the system behaves in a realistic setup and show the limits in terms of achievable hard real-time guarantees
Case study

- Two image sizes
  - 64x48 (9.1KB) and 128x64 (22KB)
  - Limitations on the SPM size and benchmark
- Images from the KITTI vision benchmark suite dataset
- Code size
  - Disparity 64x48: 349KB
  - Disparity 128x64: 745KB
  - Erika RTOS: 294KB
- Four out of the five scenarios previously described
Case study: Disparity

Supported Frequency 64x48

![Frequency Chart]

- LCSY-SOLO
- LCSY-STRESS
- OUR-SOLO
- OUR-HIGH
Case study: Disparity

Supported Frequency 128x64

- LCY-SOLO
- LCY-STRESS
- OUR-SOLO
- OUR-HIGH

Frequency (Hz)
Summary and Future Work

Multiple criticality domains on MPSoCs

Software techniques (isolation, cache coloring, code/data relocation)

Hardware techniques (SPM, dedicated PS-PL interfaces, address translator IP)

Full-stack system implementation

Compiler Integration

Security

Schedulability Analysis
Thank you!