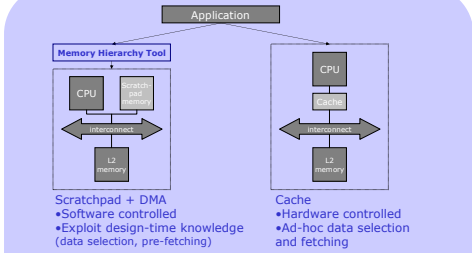
Source-to-source optimizations of statically allocated data mapping on MPSoC platforms*

Arindam Mallik, Maryse Wouters, Peter Lemmens, Eddy De Greef, Thomas J. Ashby
(arindam, woutersm, lemmensp, degreef, ashby)@imec.be
IMEC vzw, Smart Systems and Energy Technology

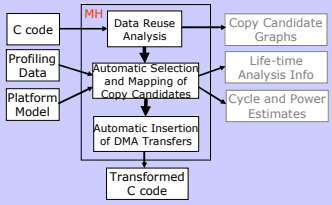


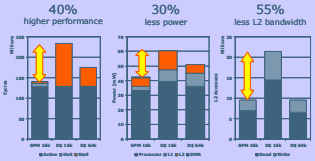
Scratchpad + DMA
•Software controlled
•Exploit design-time knowledge (data selection, pre-fetching)

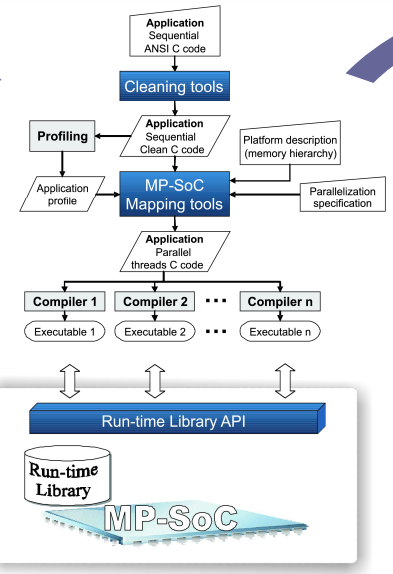
Cache
•Hardware controlled
•Ad-hoc data selection and fetching

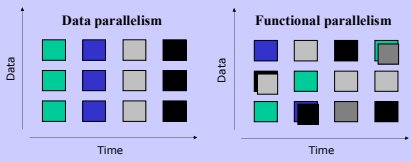
Scratchpad optimization with MH[1]:

- Designer provides profiling data and high-level platform description
  - •Array access profiling data
  - •Platform description
- MH tool analyzes the source code and inserts re-use buffers and DMA transfers
  - •Selection of re-use buffers
  - •Scheduling of DMA transfers
- Execute code
  - •On scratchpad based platform
  - •On high-level simulator (HLsim),







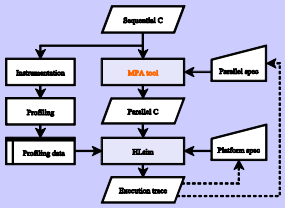


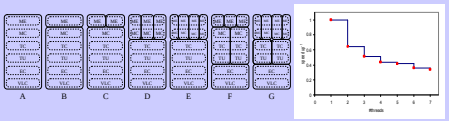
The dividing of a task into separate identical parts that are executed in parallel on different items of data.

The dividing of a task into separate (different) parts that are executed in parallel.

Automatic Parallelization with MPA[2]:

- Designer specifies parallelization
  - •Through a "ParSpec" file
  - •Assign code-blocks to a thread
  - •Both functional and data-level parallelism
- MPA tool applies the parallelization
  - •Create threads
  - •Insert communication channels
  - •Insert synchronization constructs
- Execute parallel code
  - •Using Pthreads-based RTlib on Linux
  - •Using HLsim with time-annotation





Based on instruction-accurate CoWare ARM11 model

MP-MH Tool

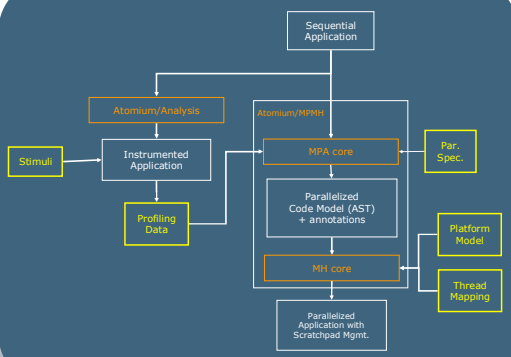
- Automatic parallelization of sequential code
- Scratchpad memory usage optimization
- Optimization of statically allocated data usage and allocation
- Tool for multi-processor platforms
- Natural order:
  - First parallelization (MPA)
  - Then scratchpad optimization (MH)

Motivations:

- Direct information flow possible between MPA and MH cores
- Fewer preconditions to check
- Facilitates synergies

Design Challenges:

- MH - No support for multi-threaded applications
- MH - No support for multi-processor platforms
- MPA - Cannot analyze block transfers
- MPA - Not platform-aware



Improvements:

- MH core has been made thread-aware
  - •Reuse opportunities for each thread separately
  - •Shared variables (arrays) are analyzed globally
- MH core can now model multi-processor platforms
- Sequential profiling data is 'parallelized' to have accurate estimations in each thread
- Automatic BT assignment for synchronization
- Shared arrays are correctly analyzed and copy sizes are correctly calculated

* This work is developed as a part of the contribution from IMEC in context of the MNEMEE project (European ICT FP7 Embedded Systems Design).

[1] R Baert, E De Greef, E Brockmeyer, G Vanmeerbeeck, P Avasare, J Mignolet, M Cupak, "An automatic scratch pad memory management tool and MPEG-4 encoder case study", in DAC, 2008
[2] R Baert, E Brockmeyer, T Ashby, S Wuytack, "Exploring parallelizations of applications for MPSoC platforms using MPA", in DATE, 2009
[3] J.-Y Mignolet, R Wuyts, "Embedded multi-processor systems-on-chip programming", To be published in Software, IEEE , vol.26, no.3, May-June 2009