Adaptive Embedded Systems Challenges of Run-Time Resource Management

Jan Madsen

DTU Informatics
Department of Informatics and Mathematical Modeling

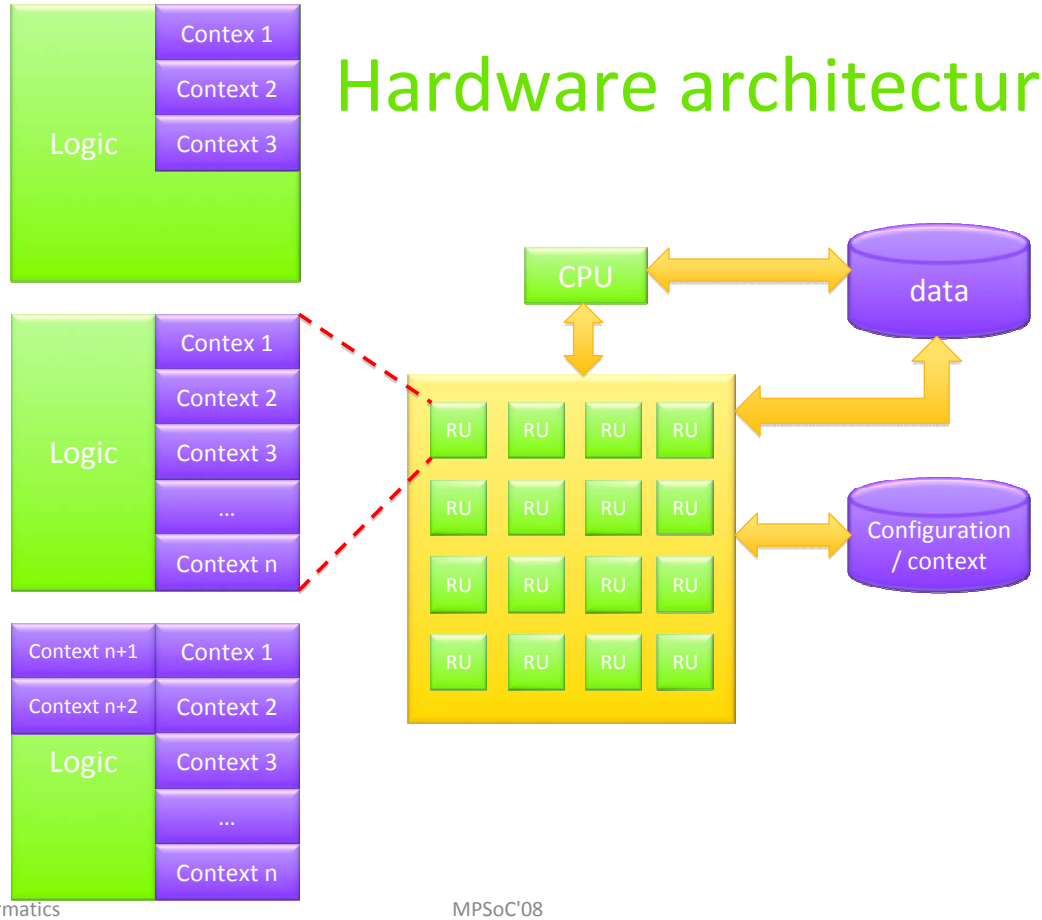
Adaptivity

- An embedded system is adaptive, if it can modify its behavior and/or architecture to changes in requirements, objectives, and/or external conditions
- Adaptivity is increasingly important as the complexity and autonomy of embedded systems increases
- Improve performance and resource utilisation
- Improve reliability and fault-tolerance

Issues

- Hardware architecture
- Application mapping
- Run-time resource management

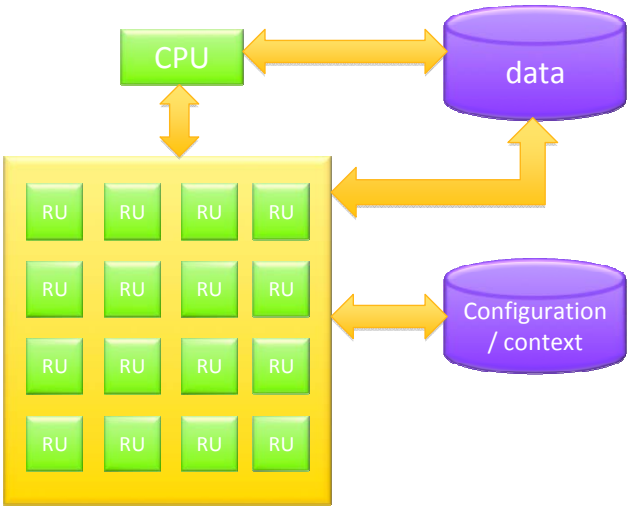
Hardware architecture



Hardware architecture

Issues

- Size of RU?
- Size of logic?
- Granularity of logic?
- Number of contexts?

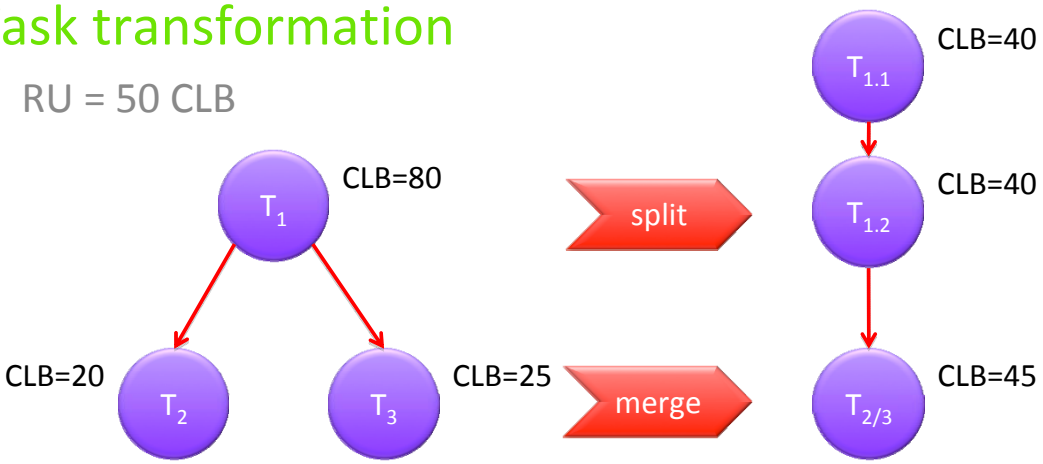


- $A = \#RU * (A_{logic} + \#context * A_{context}) + A_{...C}$

Application mapping

Task transformation

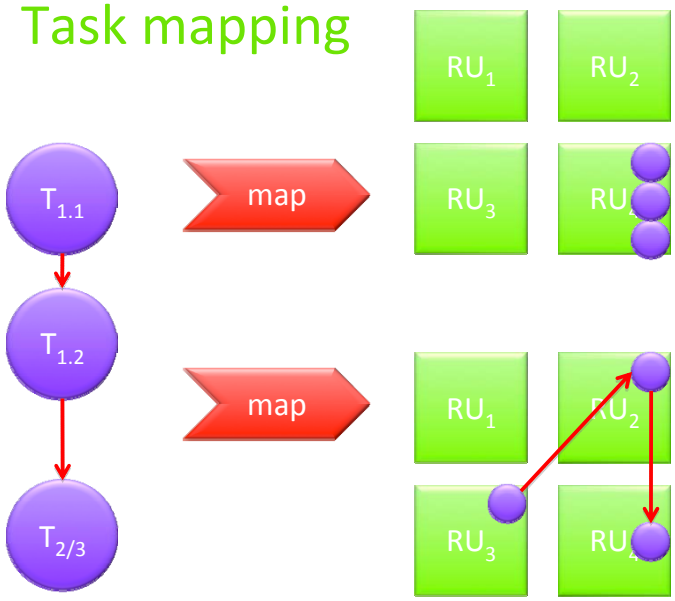
- RU = 50 CLB



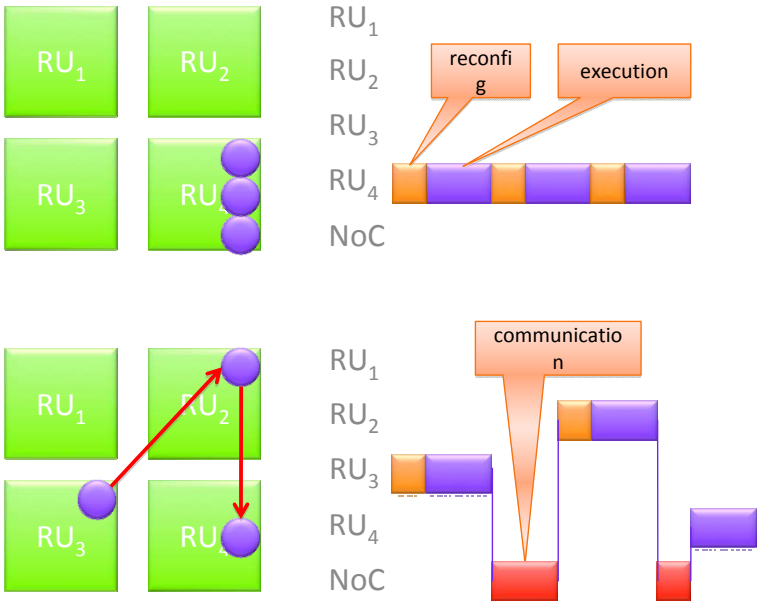
- Task may be mapped to any RU

Application mapping

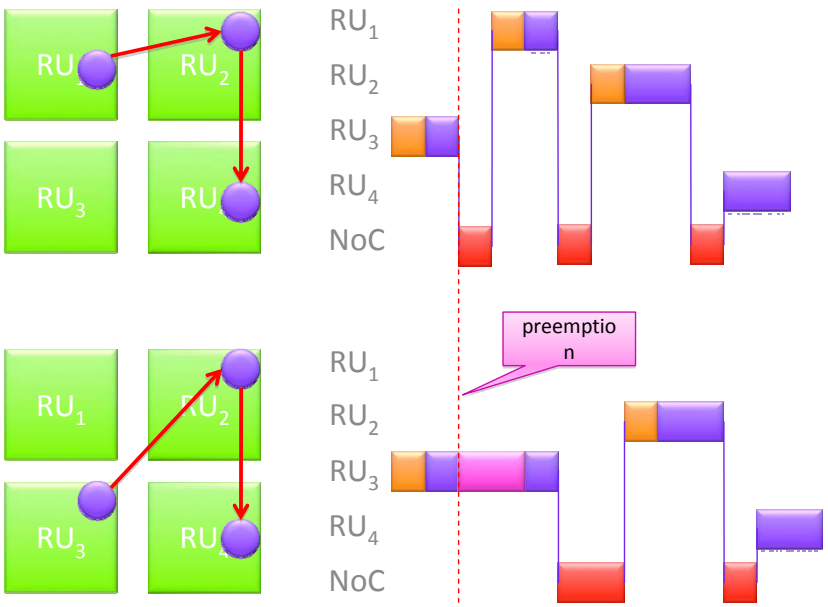
Task mapping



Run-time resource management



Run-time resource management



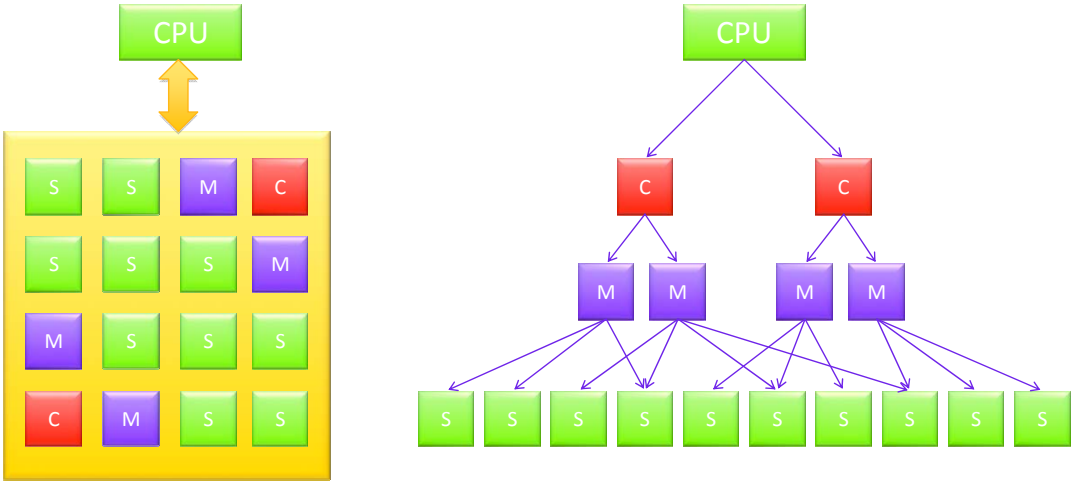
Run-time resource management

Issues

- Task dependency monitoring
- Task scheduling
- Task allocation and free-context management
- Task reallocation

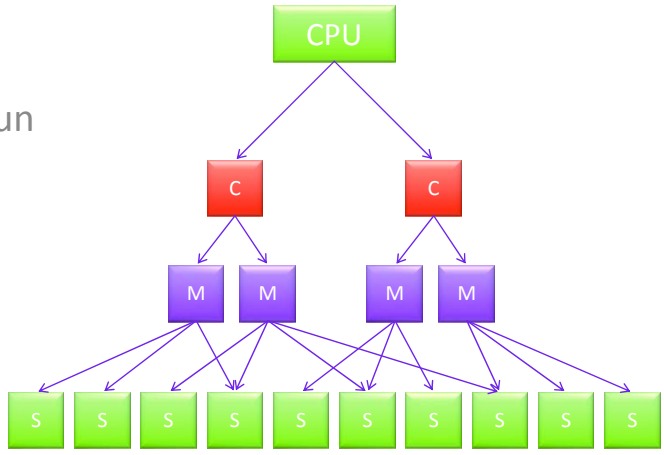
➡ *Time critical!*

Run-time resource management



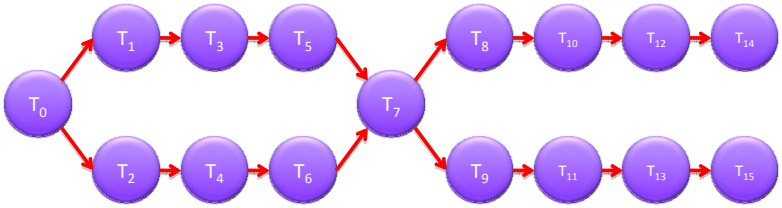
Run-time resource management

- C-node
  - Collect resource distribution information
  - Decides on M-nodes to run application
- M-node
  - Allocate application to S-nodes
  - Monitor synchronization
- S-node
  - Run tasks of application
  - Multiple contexts requires local scheduling

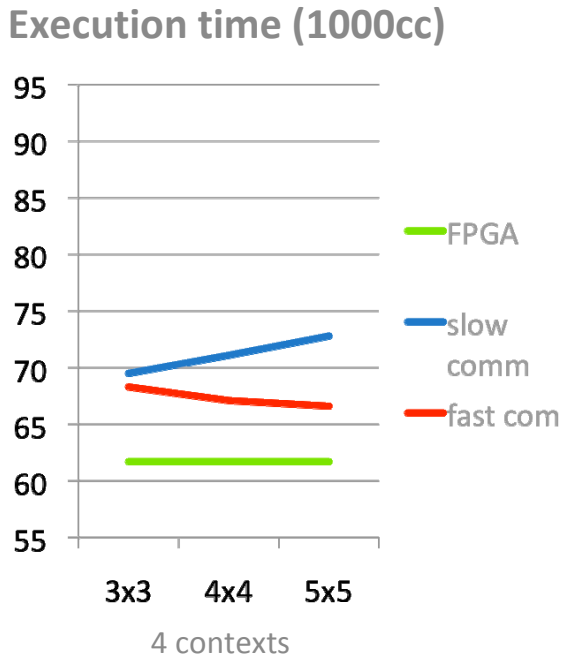


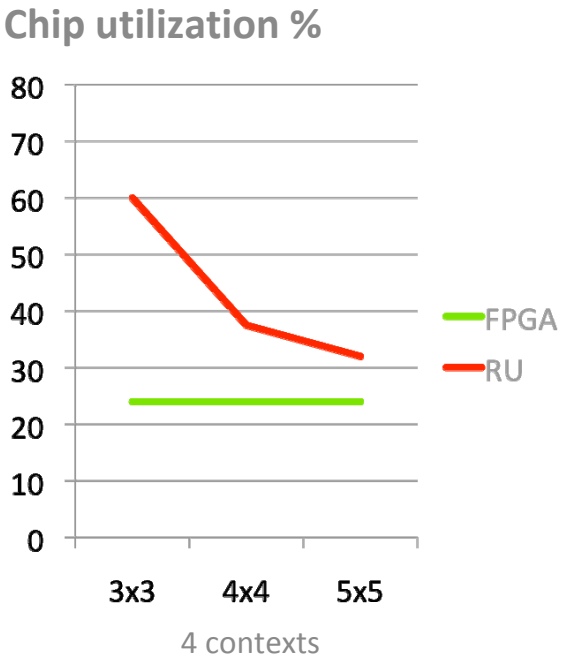
Experiment: MP3 case

- Assumptions
  - Area = 10.000 CLBs
- FPGA implementation
  - Area = 2408 CLBs (24% of chip)
  - Execution time: 61739 cc

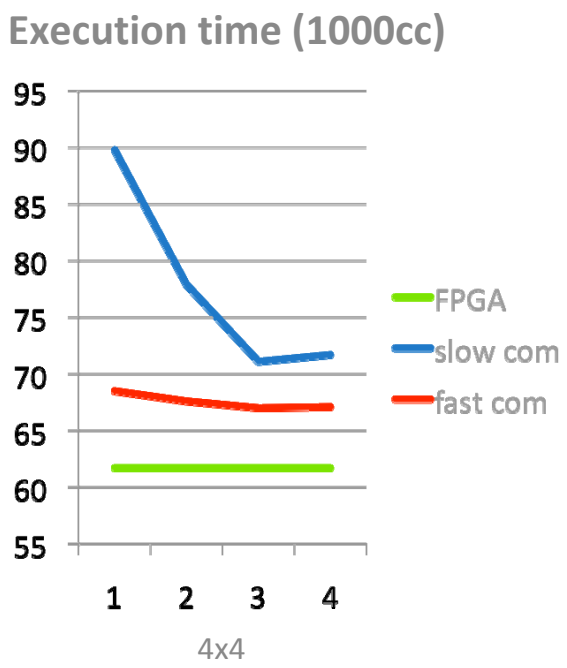


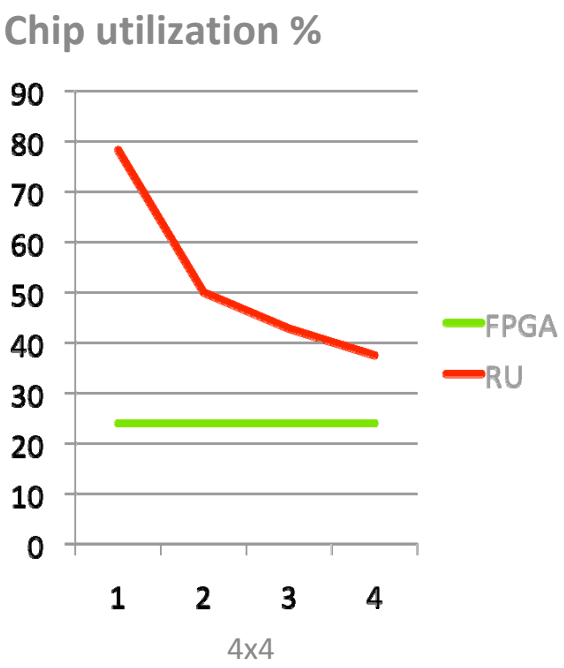
Experiments: RU size?



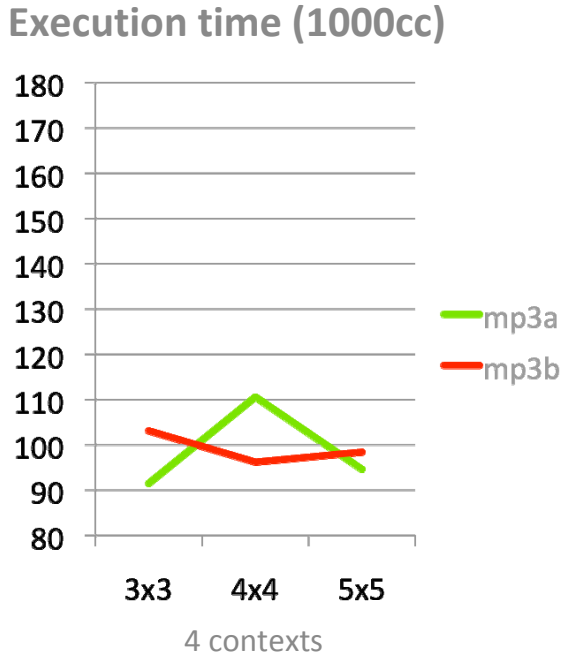


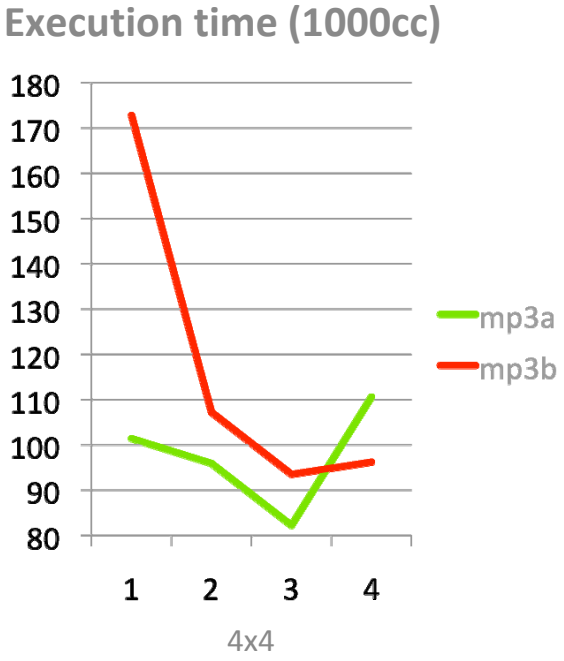
Experiments: number of contexts?





Experiments: reallocation?



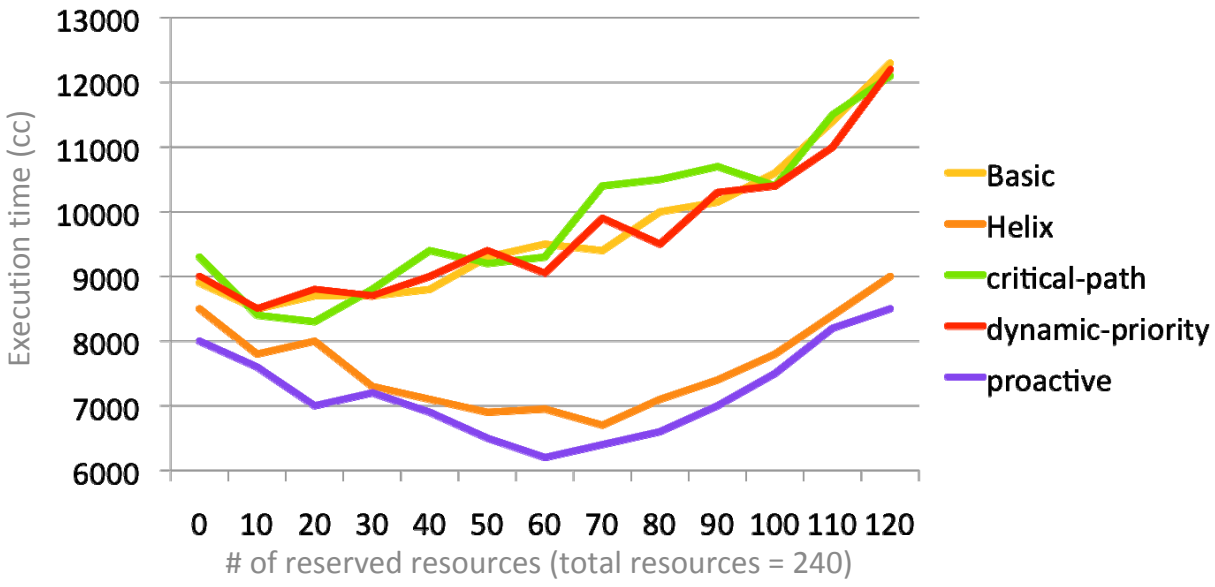


Allocation/reallocation strategies

- Basic
  - Form cluster around M-node to reduce communication
- Critical-path
  - Prioritize allocation of tasks on critical path
- Helix
  - Form local cluster for each application
- Dynamic-priority
  - Prioritize tasks that are close to finish
- Proactive
  - "Run-time clean-up"

Allocation/reallocation strategies

- Running 100 applications (of 5 different types)



Summary

- Understanding run-time behavior of dynamically reconfigurable systems is *hard*
- Being able to do run-time resource management is *even harder*
  - Analysis has to be fast to be feasible
  - What are the key issues to analyse?
  - Reallocation issues are complicated to analyse at run-time
- Essential to understand interplay between the different aspects of the dynamic behavior
- *Need tools to support this!*

Acknowledgements

- Kehuai Wu
- Esben Rosenlund Hansen
- Michael Reibel Boesen

- Kehuai Wu, Esben Rosenlund Hansen, Jan Madsen. *Towards Understanding and Managing the Dynamic Behavior of Run-Time Reconfigurable Architectures*, to appear in the proceedings of the Conference on Engineering of Reconfigurable Systems and Algorithms (ERSA'08), July 2008.
- Kehuai Wu and Jan Madsen. *COSMOS: A System-Level Modelling and Simulation Framework for Coprocessor-Coupled Reconfigurable Systems,* SAMOS VII: International Symposium on Systems, Architectures, Modelling and Simulation 2007.