

Laboratory: “Designing Application Specific Embedded Processors”

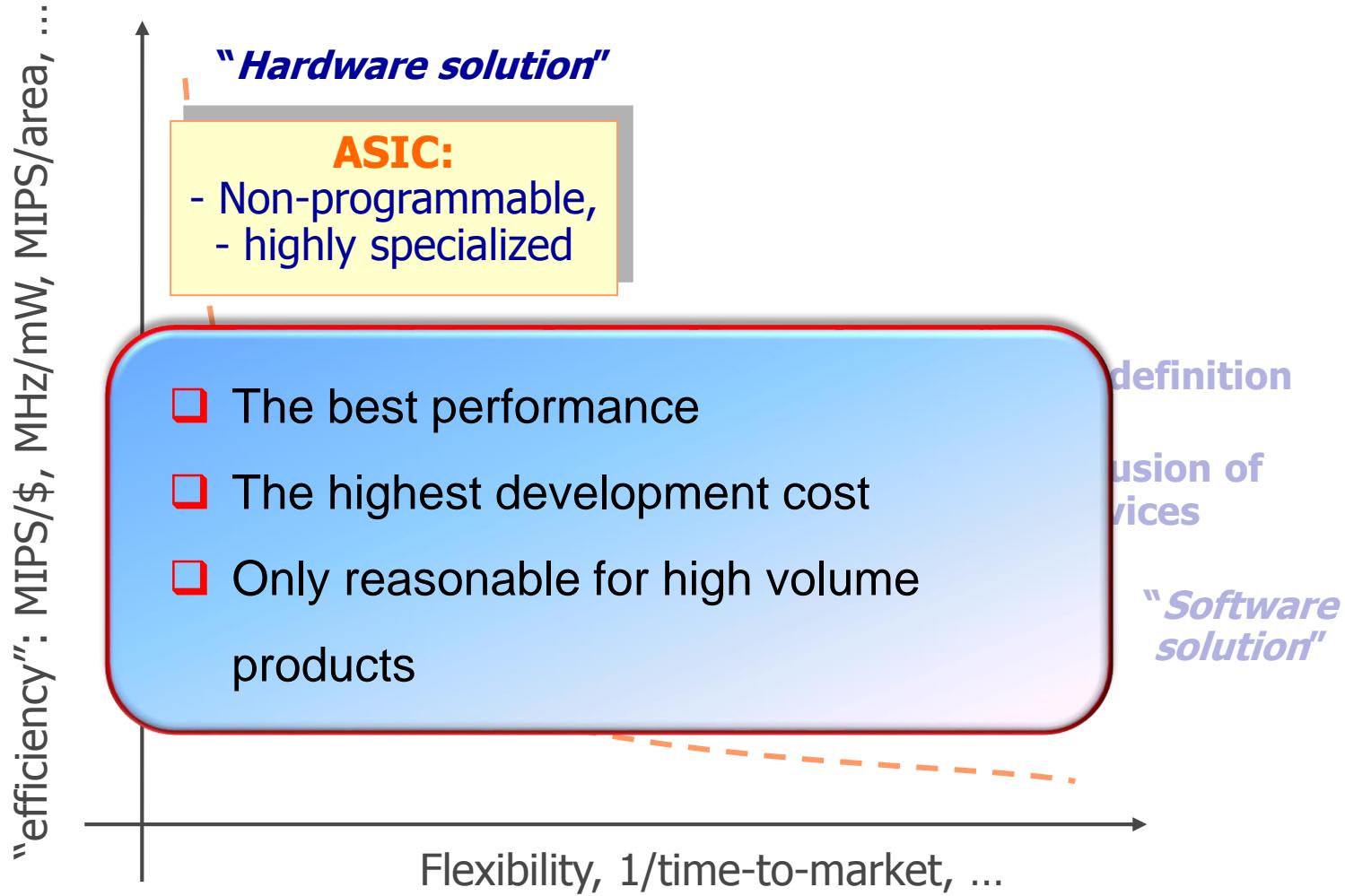
Dr.-Ing. Hussam Amrouch

(Lehrstuhl Prof. Dr. J. Henkel)

CES - Chair for Embedded Systems

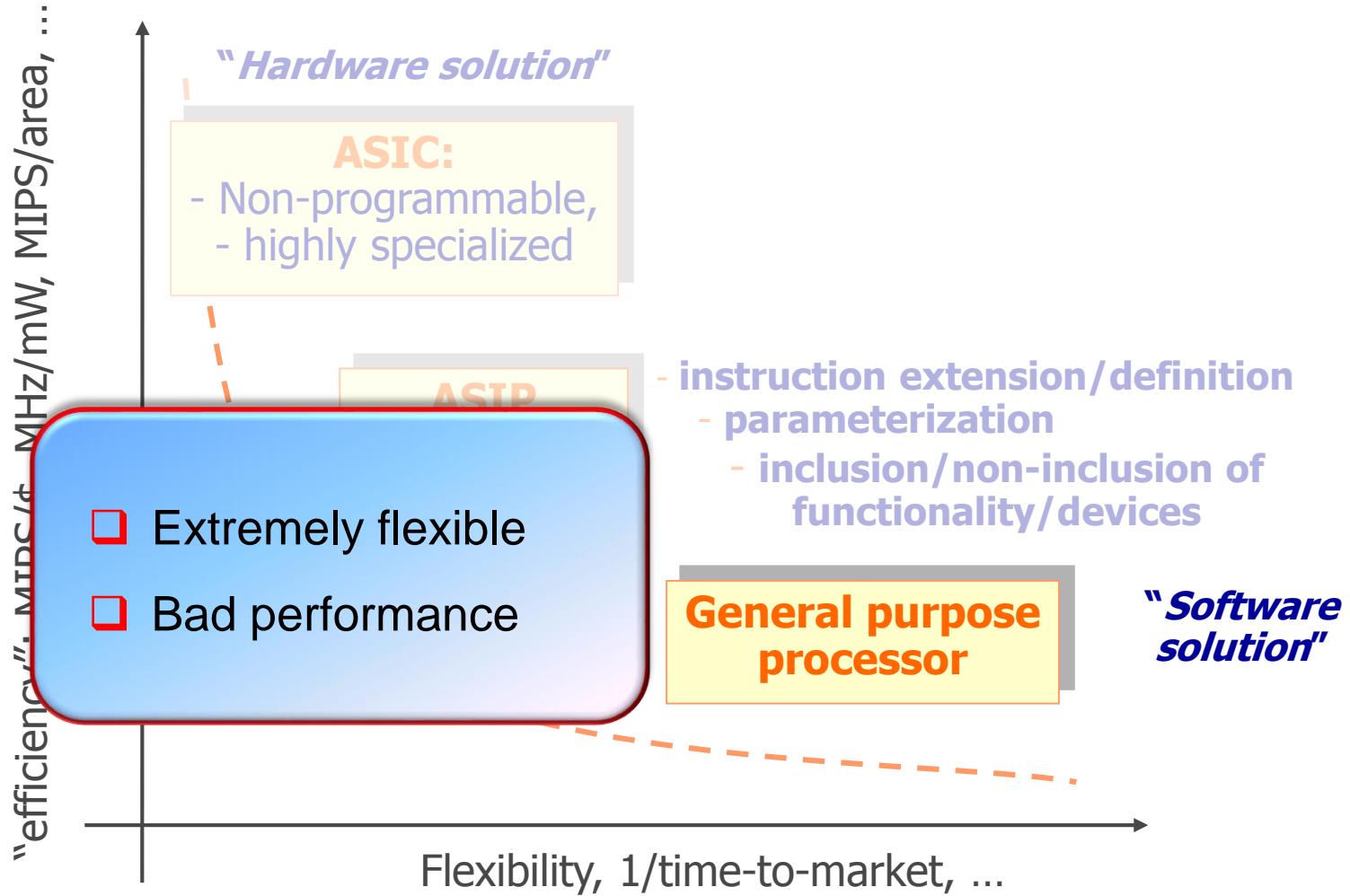
KIT - Karlsruhe Institute of Technology, Germany

ASIPs: efficiency vs. flexibility



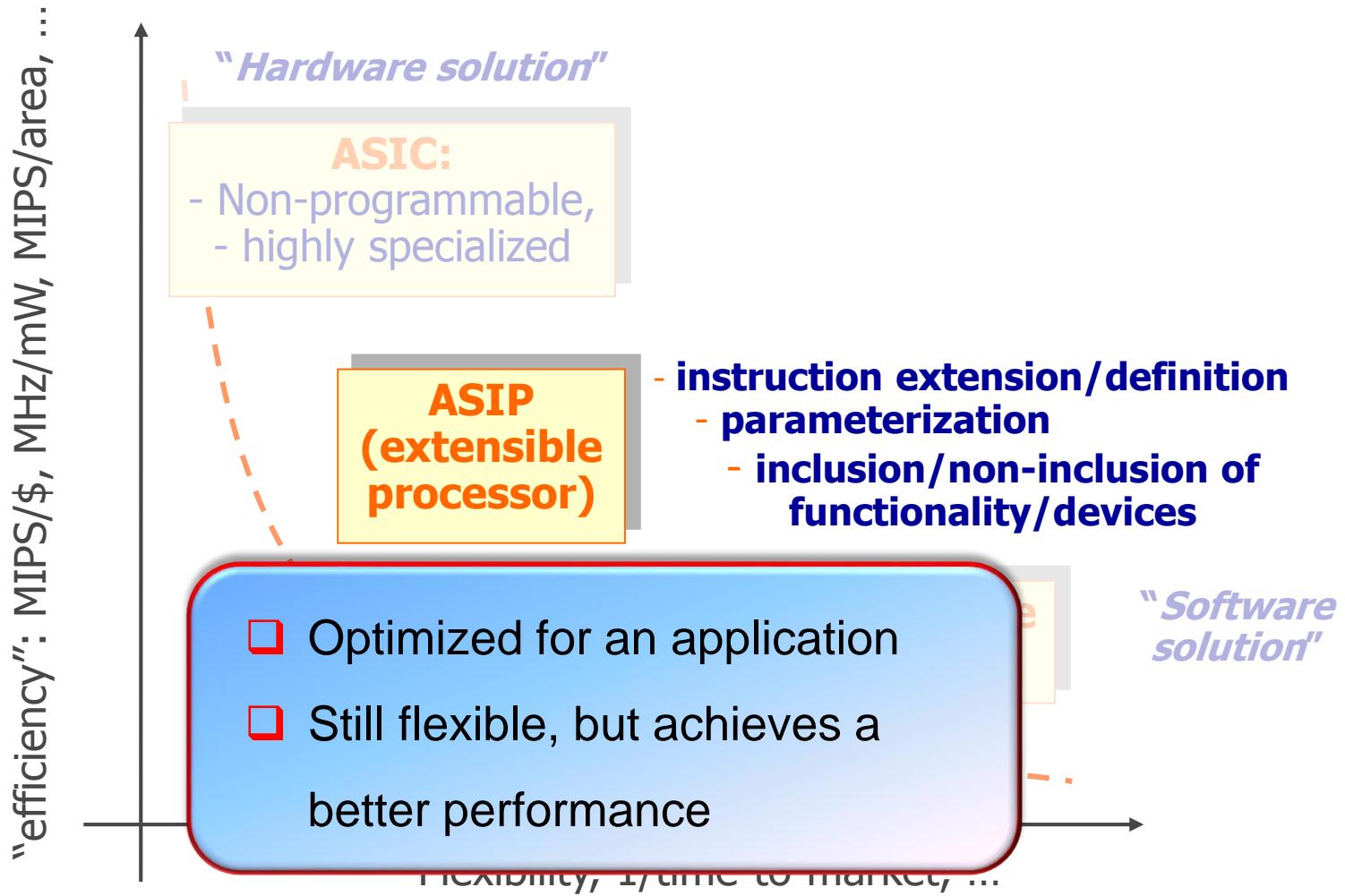
src: Henkel "Design and Architectures for Embedded Systems (ESII)"

ASIPs: efficiency vs. flexibility



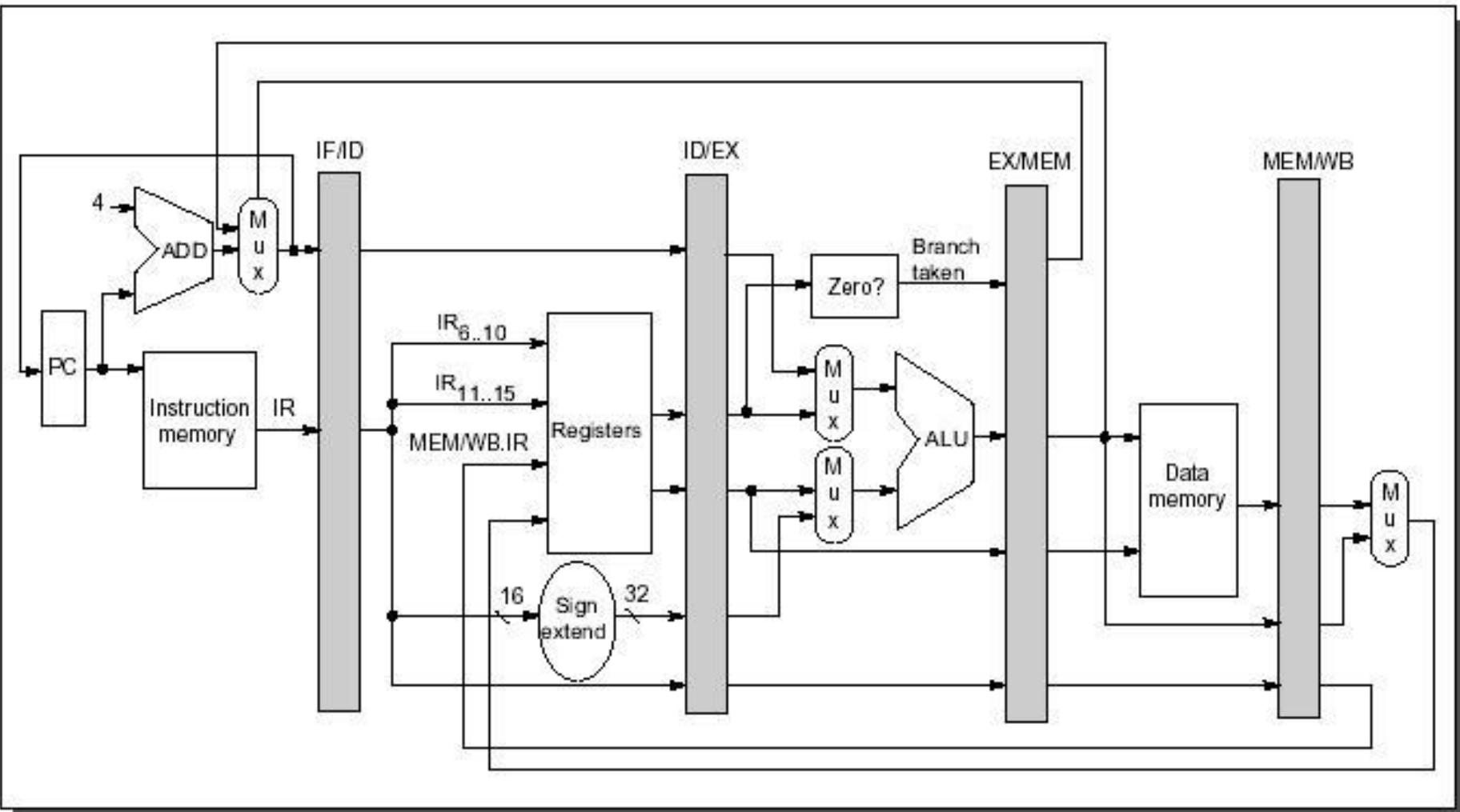
src: Henkel "Design and Architectures for Embedded Systems (ESII)"

ASIPs: efficiency vs. flexibility



src: Henkel "Design and Architectures for Embedded Systems (ESII)"

Pipeline Processor

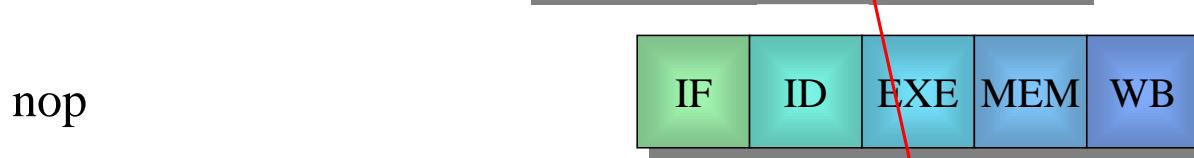
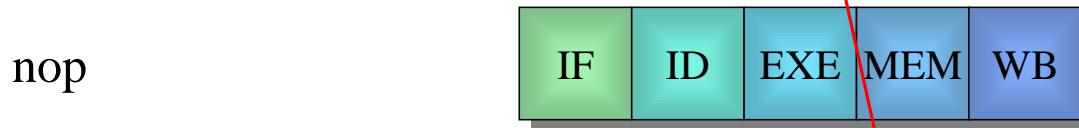
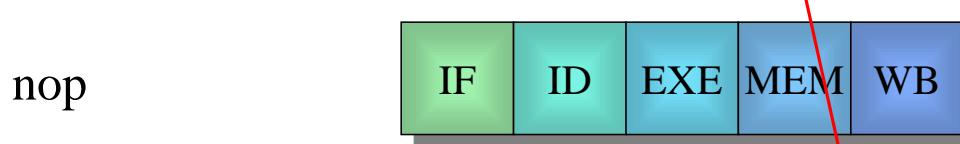
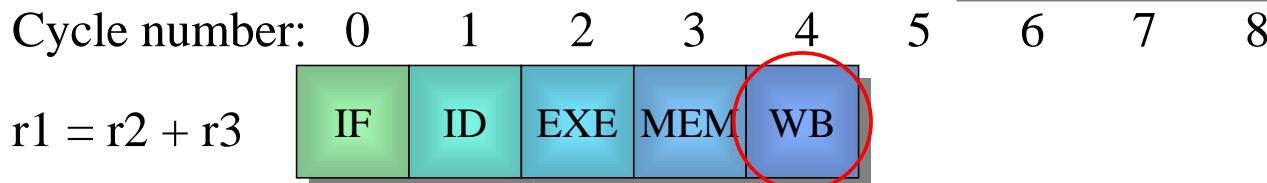


src: Muhammad Shaaban "The DLX Architecture", lecture on 'Computer Architecture'

Pipelining

Overlapping Execution of multiple Commands with a data dependency

IF	Instruction Fetch
ID	Instruction Decode
EXE	Execute
MEM	Memory Access
WB	Write Back

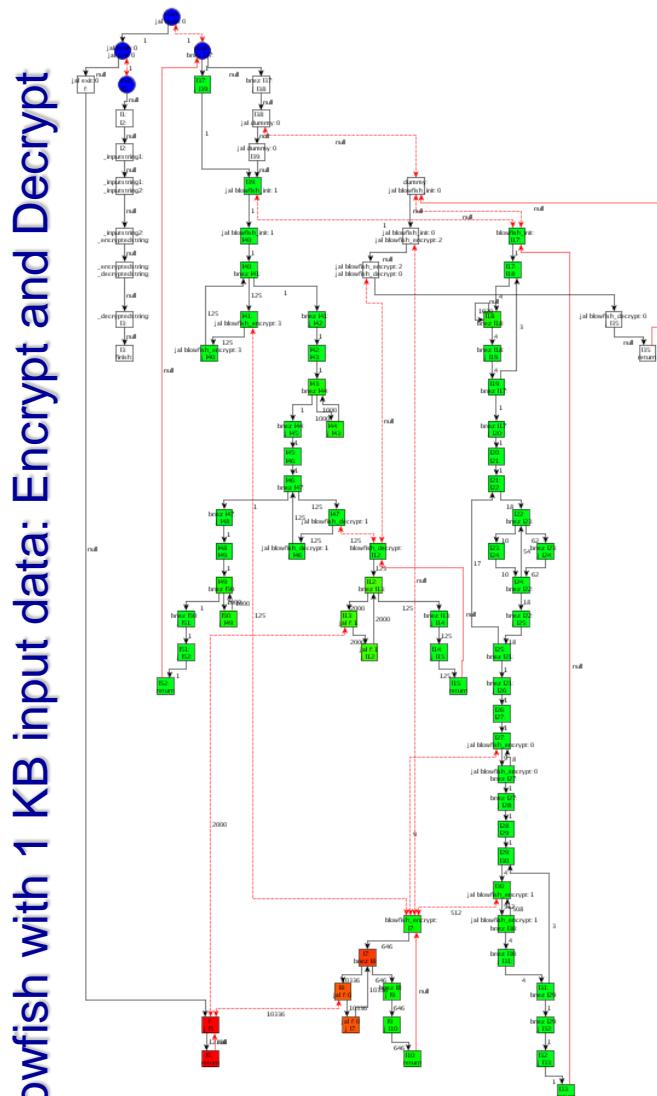


Goal of ASIP Lab

- ❑ Creating new CPUs with new instructions and implementing these new instructions in Hardware and Software with evaluation and testing
- ❑ The main goals are:
 - ❑ creating new ASIPs for special applications
 - ❑ benchmark those ASIPs to find out their benefits and drawbacks
 - ❑ and finally to interpret the benchmark results

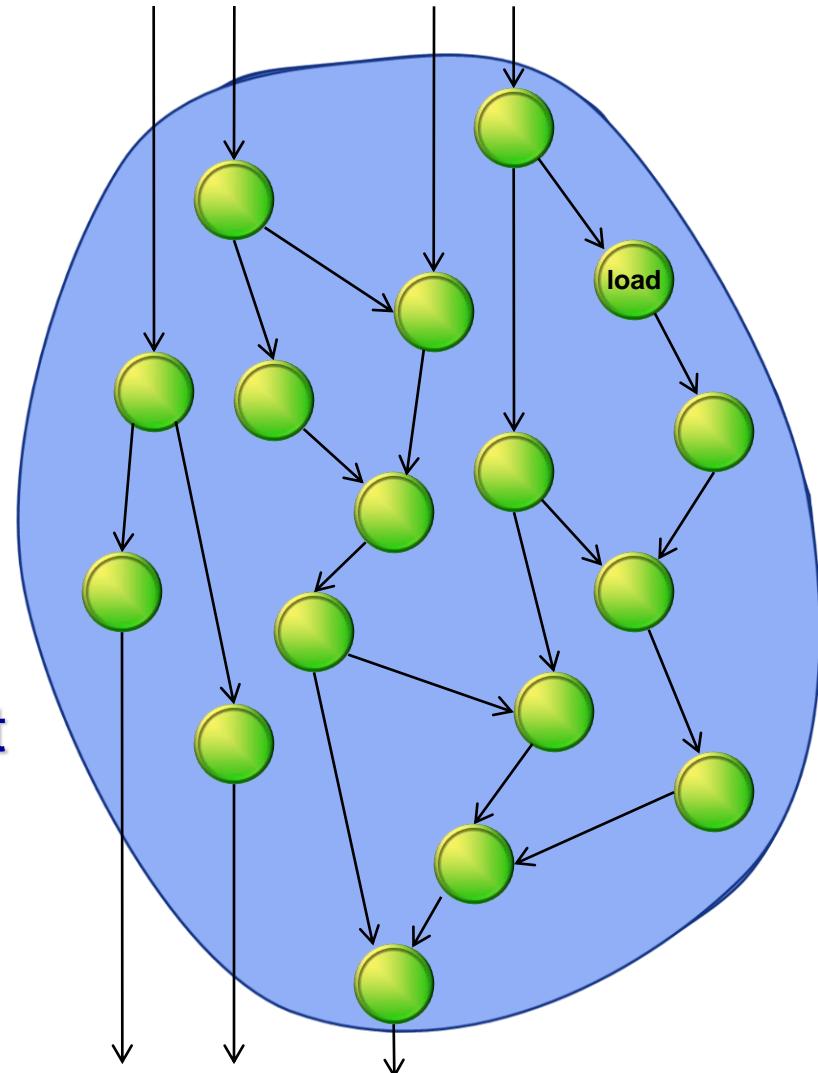
Profiling on Control-Flow Graph

- ❑ Each node is a **Base Block**:
 - ❑ The instructions in a Base Block are always executed together
 - ➔ No jumps in a Base Block, except the end
 - ➔ No jump targets in a Base Block, except the beginning



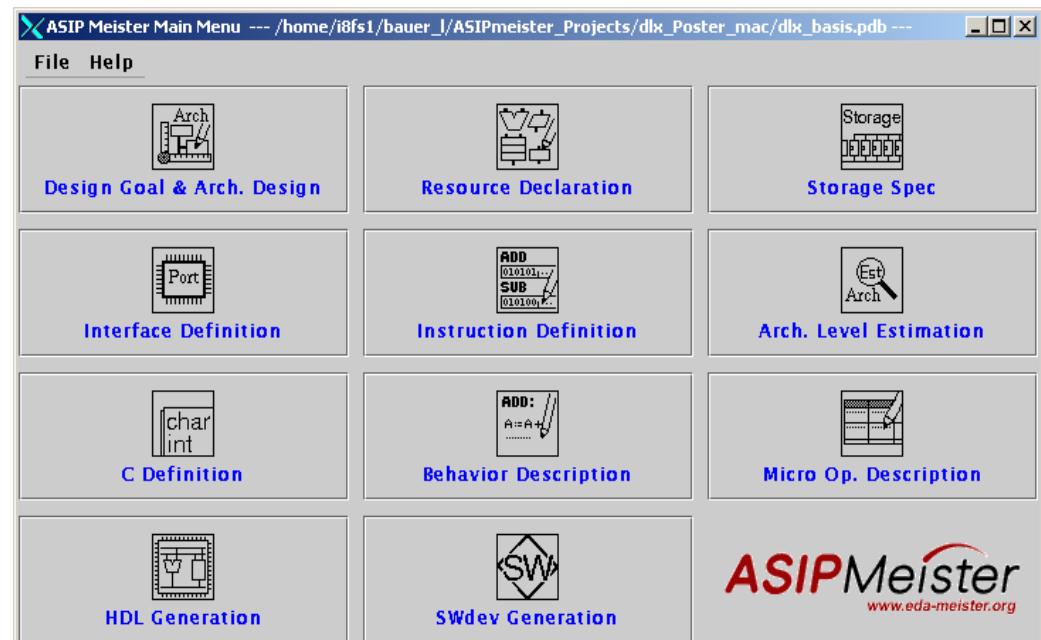
Constraints for Designing Custom Instructions

- ❑ Number of Input-/Output-Values to Custom Instruction
 - ❑ Limited by register file, e.g. 4 read and 2 write ports
- ❑ ‘Forbidden Nodes’, e.g. Load/Store
- ❑ Convex Graph
- ❑ Limited Frequency, Area, Power, and/or Energy Budget
 - ❑ An ASIP that consumes more power may consume less energy, if the application terminates much earlier



Tasks in ASIP Lab

- ❑ Programming in assembly language
- ❑ Implementing new instructions with ASIP Meister (+Simulation, +Hardware)
- ❑ Evaluating the results from the new CPU
- ❑ designing new instructions for large applications with testing
- ❑ Creating different CPU versions



Lab Environments

- ❑ dlxsim: Simulator for DLX-Assembler
- ❑ ASIP Meister: To create new processor
- ❑ CoSy: To compile C-code to assembly code
- ❑ ModelSim: simulator for VHDL-Code
- ❑ Xilinx ISE: Synthesis of VHDL-Code
- ❑ XPower: To estimate power consumption



Important Directories

- ❑ /Software/epp:
 - ❑ The required programs for the Lab are often used automatically by scripts
- ❑ ~asip00/ASIPMeisterProjects/TEMPLATE_PROJECT/
 - ❑ New version of Lab scripts
 - ❑ Project example with application example
- ❑ ~asip00/SessionX: The requirements for each session
 - ❑ For instance, you can find the slides and the Lab Script in ~asip00/Session0

Important Note

- ❑ The sessions need ALWAYS to be executed remotely on one of the following PCs and not locally:

i80labpc01, i80labpc02, ... i80labpc010

To avoid compatibility issues between the Software tools and the OS

General Information

- ❑ Computer-Accounts (Ubuntu 12.04)
 - ❑ For the first session, read chapter 3 (dlxsim; without 3.2.2, 3.2.3 & 3.3.2)
 - ❑ For the second session, read chapter 2 (working environment), 8.3 (using the compiler) and 5 (Modelsim)
-
- ❑ **Now:** Accounts & Linux-Tutorial (chapter 2.2)

Supervisors and the regular meeting

□ Dr. Hussam Amrouch

□ Sajjad Hussain

□ Regular Meeting:
what about Tuesday 14:00 ?



Questions??