Automatic Tool Generation from Structural Processor Models

Institute of Computer Languages
Vienna University of Technology

Christian Doppler Laboratory
Compilation Techniques for Embedded Processors

Florian Brandner

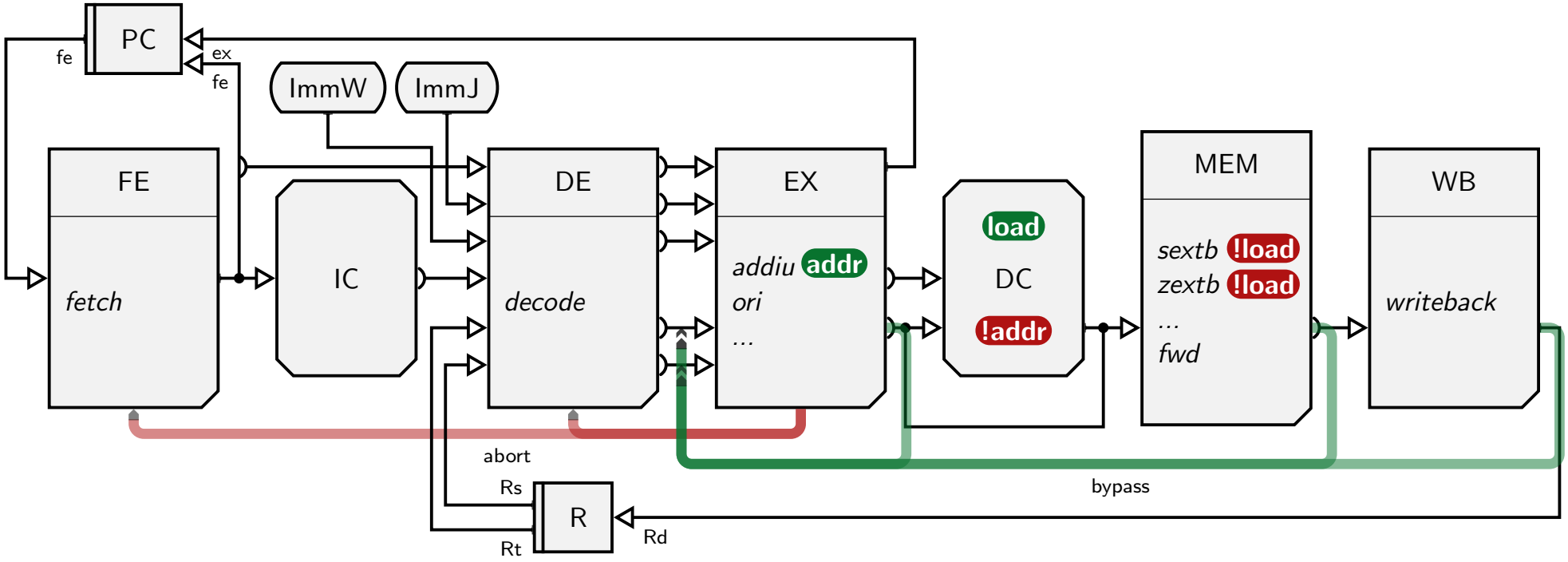
This work is supported by OnDemand Microelectronics and the Christian Doppler Forschungsgesellschaft.

xADL Processor Description Language

- Structural description of processor features
  - Components interconnected by *links*
    - Functional units, caches, memories, registers
    - Based on extensible types
    - Support for generics
  - Binary encoding, assembly syntax, programming conventions
  - Instruction set architecture
    - Automatically extracted from the structural model
    - Along *instruction paths*

- Generator tools
  - Compiler backend
  - Instruction set simulator
  - Prototype: VHDL hardware model
  - ...

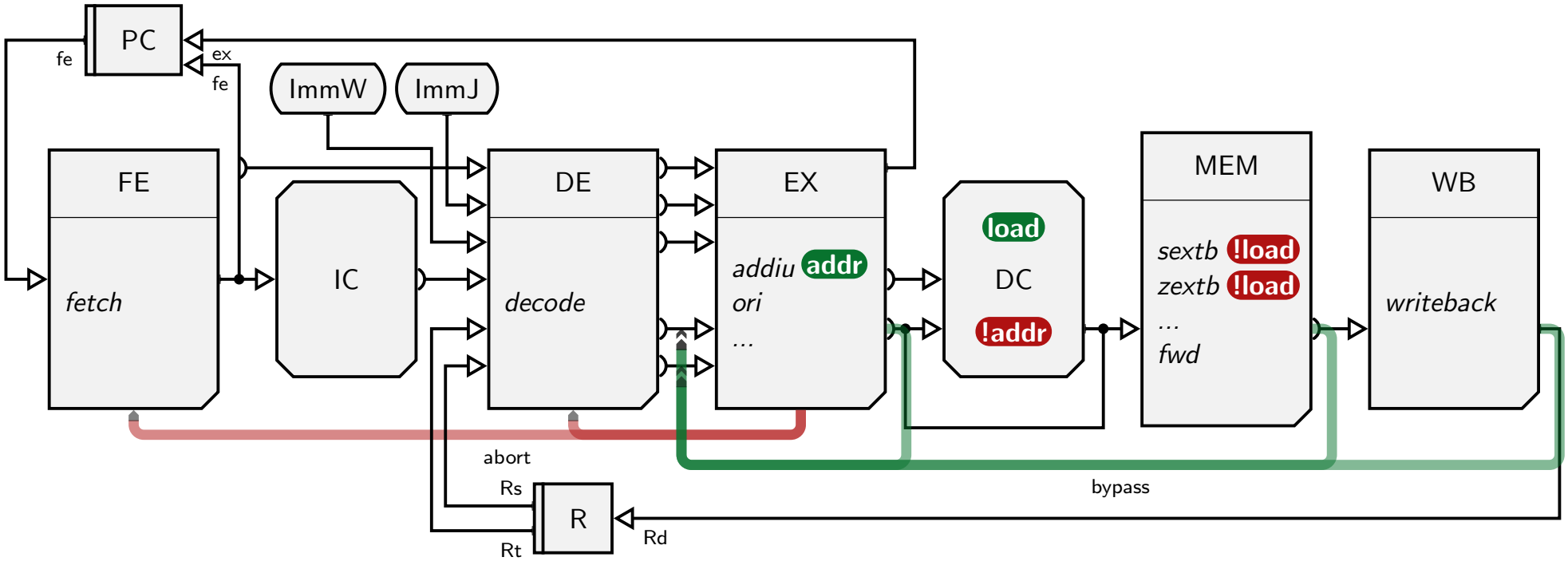
Example: MIPS Model



Instruction paths:

- FE:IC:DE:EX:DC:MEM::WB
- FE:IC:DE:EX:MEM::WB
- FE:IC:DE:EX

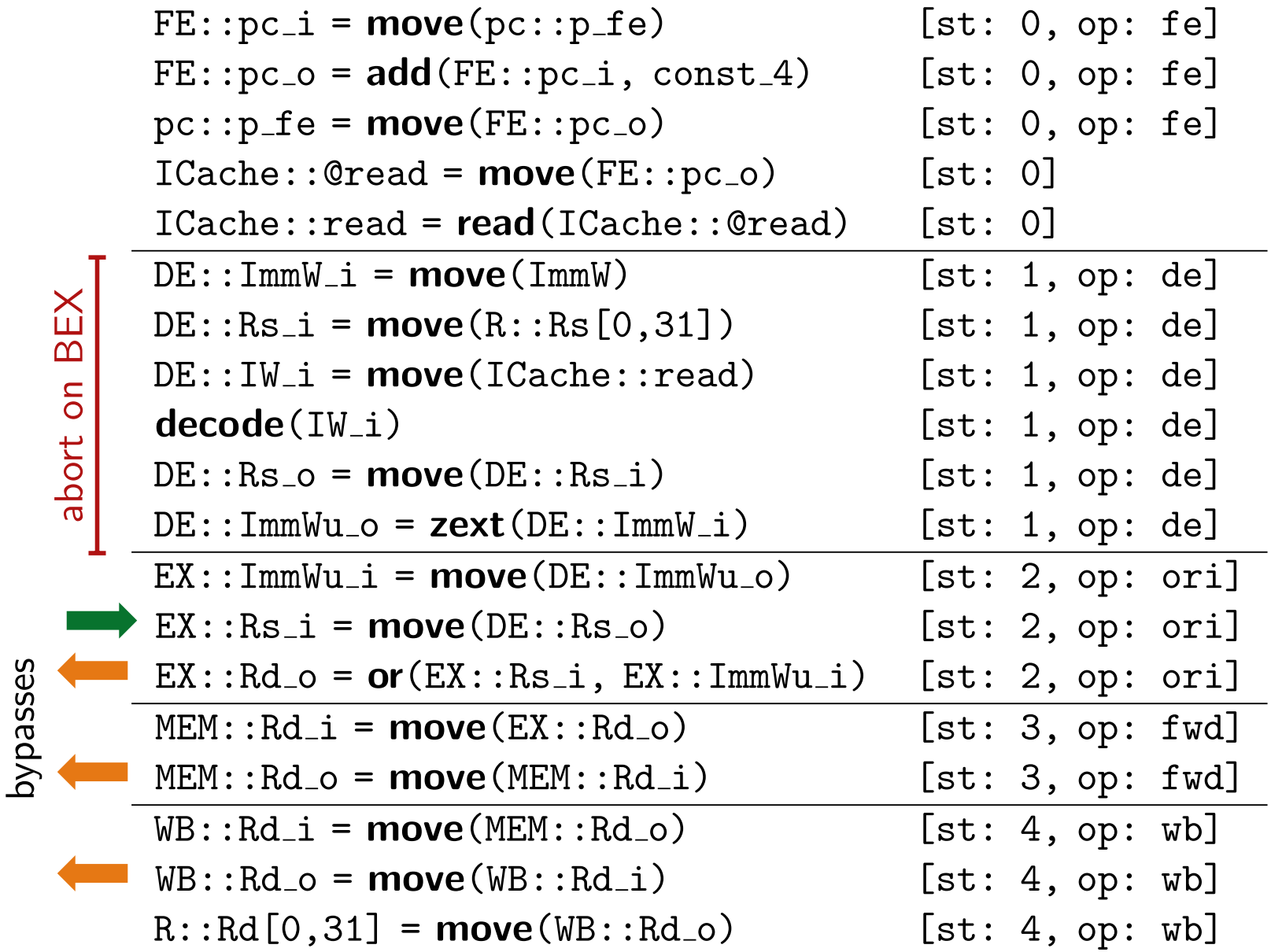
Example: MIPS Model



Instructions:

- FE:fetch IC DE:decode EX:ori MEM:fwd WB:writeback
- FE:fetch IC DE:decode EX:addiu MEM:fwd WB:writeback
- FE:fetch IC DE:decode EX:addiu DC MEM:zextb WB:writeback
- ...

Example: *or immediate* instruction



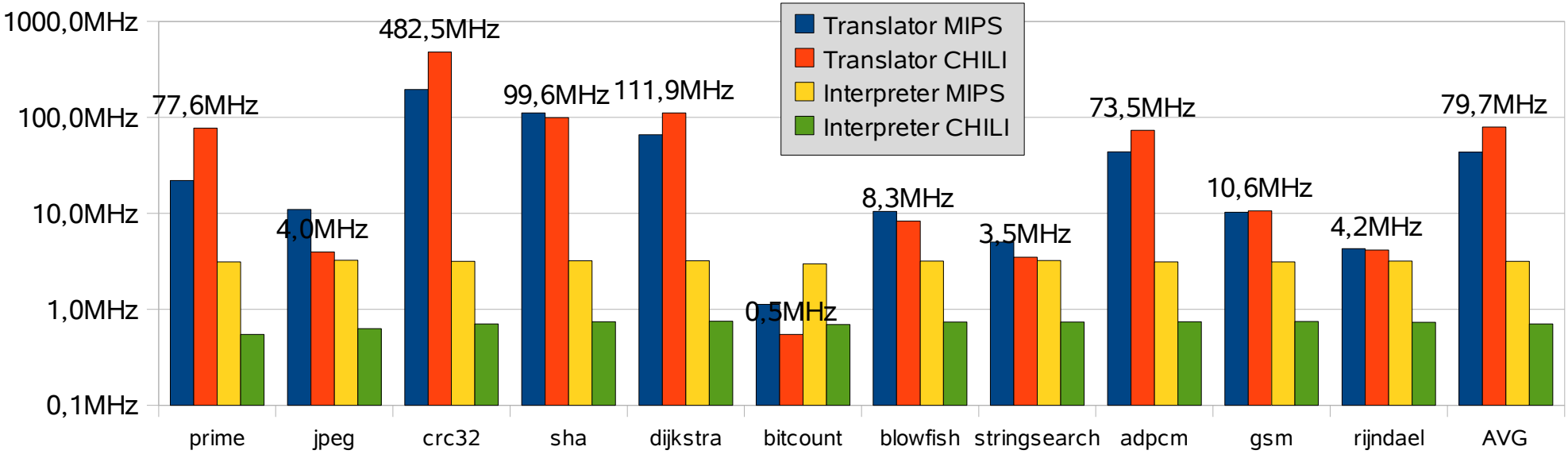
Simulator Generator

Derive software emulator of the modeled processor

- Mixed interpretation/compilation
- Split instructions according to pipeline stages
- Derive simulation and compilation functions for each stage
- Code generation
  - Using LLVM just-in-time compiler (version 2.4)
  - Compile basic blocks
    - *Linear* sequence of instructions
    - Simple optimizations only
  - Recompile *regions*
    - Arbitrary set of basic blocks
    - *Non-linear* control flow, i.e. loops
    - Aggressive optimizations

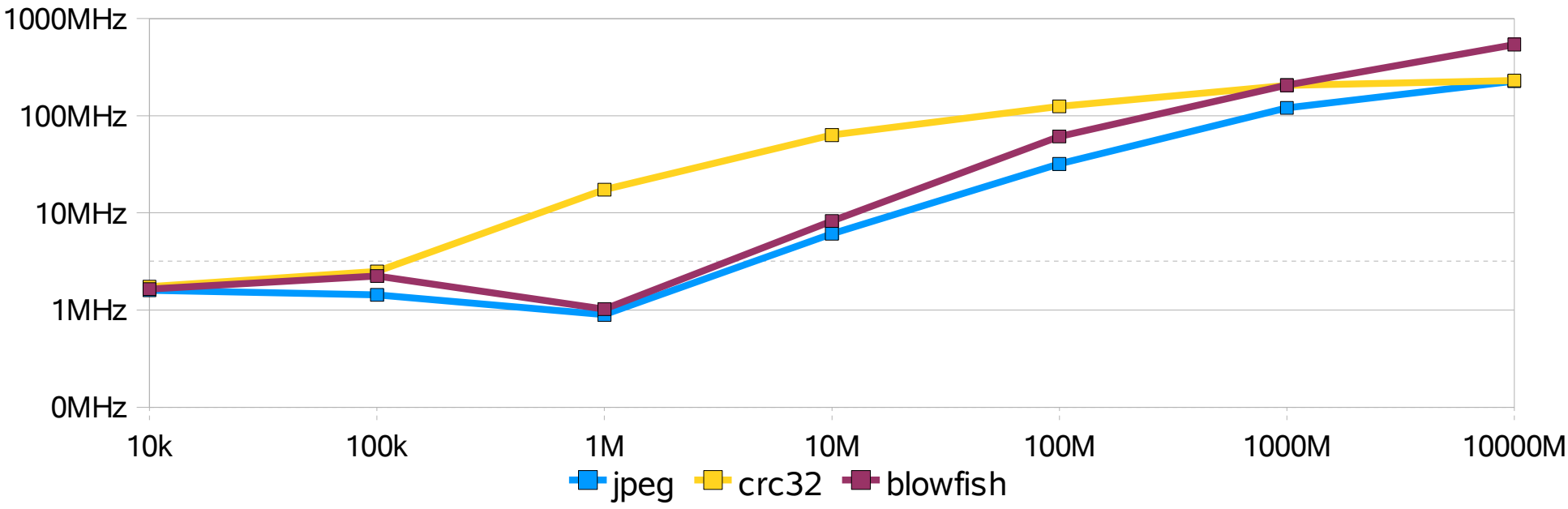
Simulator Generator - Results (1)

Running on an Athlon 64 3500+, 1 GB RAM, 32-bit Linux



Simulation speed for CHILI and MIPS processor models.
Compiled simulation reaches up to **480** Mhz!

Simulator Generator - Results (2)



Simulation speed over time for the MIPS processor model.

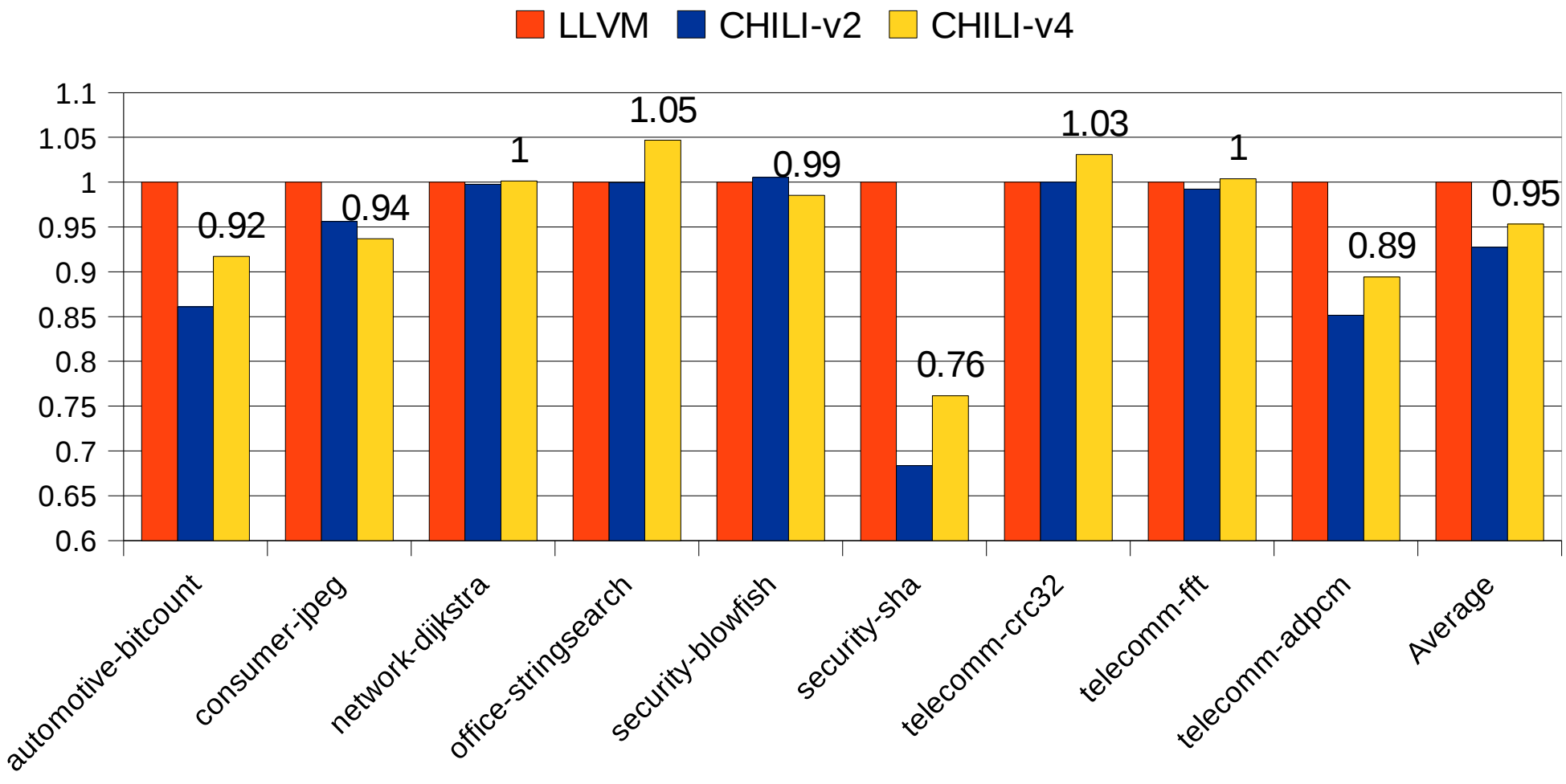
Peak performance $>$ **800** Mhz!

Compiler Backend Generator

Derive compiler backend from processor models

- Retarget LLVM compiler infrastructure (version 2.4)
- Instruction selector
  - Derive tree patterns from instruction set model
  - Extend coverage
  - Verify completeness
- Instruction scheduler
  - Instruction paths resemble possible execution flow
  - Correspond to resource tables for scheduling
- Register allocator
  - Derive register classes from register files/ports

Compiler Backend Generator - Results

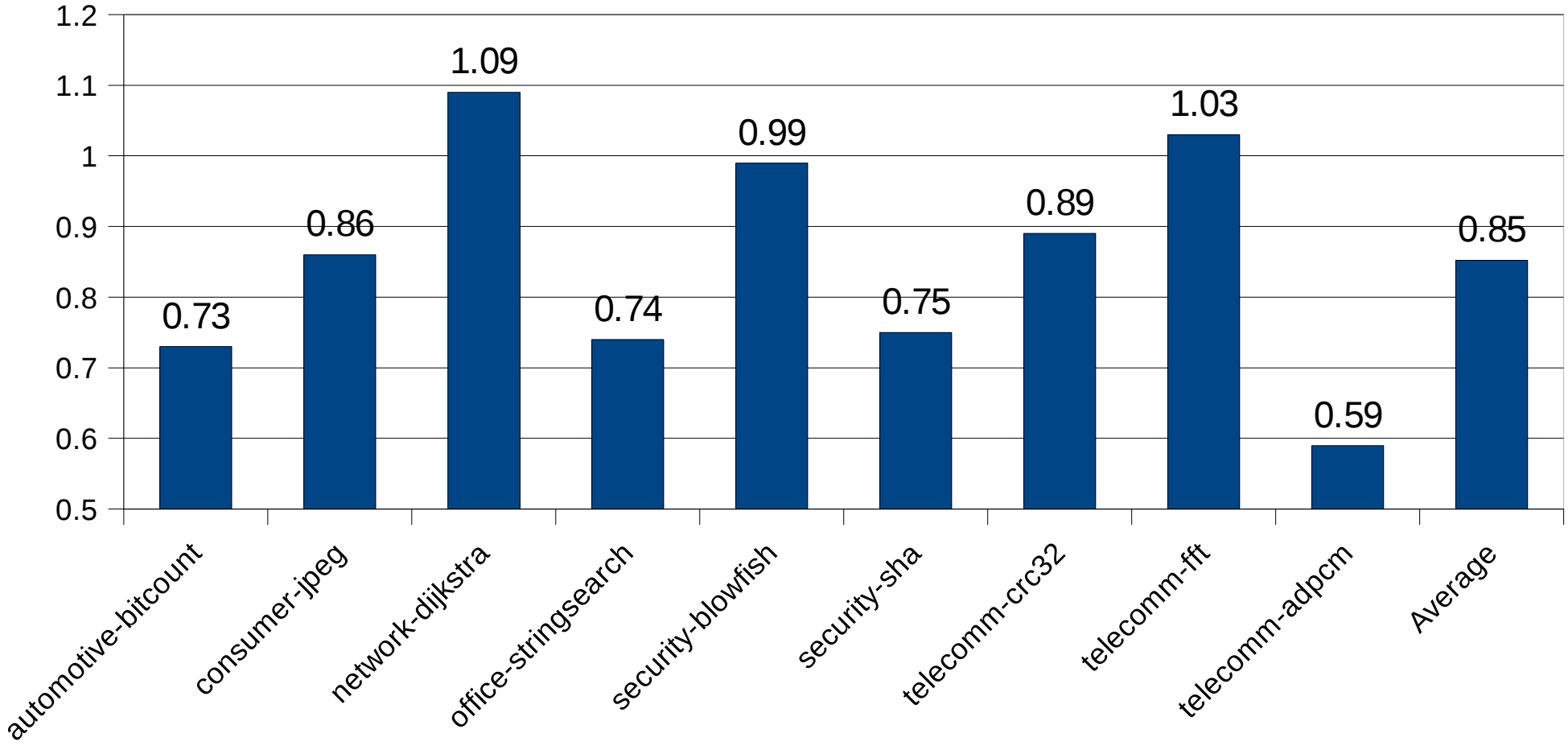


Relative performance results of LLVM-based vs. xADL-generated compilers for two configurations of the CHILI VLIW processor.

Conclusion

- Structural xADL processor description language
  - Extensible types
  - Compact and intuitive specifications
  - Instruction set extraction along instruction paths
  - Single behavior specification
- Generator tools
  - High-speed simulation
    - Mixed interpretation/compilation
    - Peak performance of $>$ 800 Mhz
  - High-quality code generation
    - Competitive with production compilers
    - Slight slower code by 5%
- Acknowledgments
  - David Rigler
  - Andreas Fellnhofer

Compiler Backend Generator - Results (2)



Relative performance results of GCC-based vs. xADL-generated compilers for the MIPS processor.