ASIP Design with Automatic C/C++ Compiler Generation

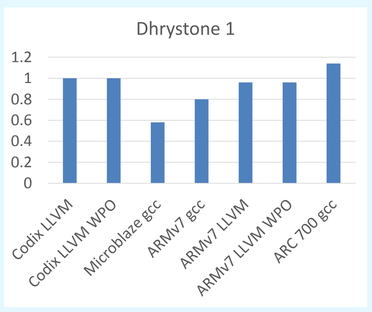
Adam Husár, Zdeněk Přikryl, Luďek Dolíhal, Karel Masařík, and Tomáš Hruška
Faculty of Information Technology, Brno University of Technology, Czech Republic
{ihusar, iprikryl, idolihal, masarik, hruska}@fit.vutbr.cz

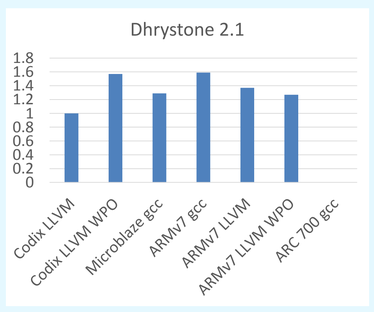
Introduction

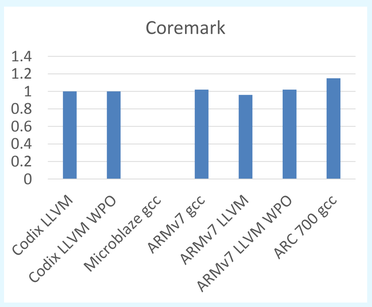
- Embedded systems issues
  - increasing complexity,
  - programmability,
  - one or more processors placed on a single chip
- Application Specific Instruction-set Processors (ASIPs)
  - optimized for a given task,
  - low power consumption,
  - tools for theirs design and, programming, testing and verification are needed,
  - automatic toolchain generation from an ASIP model in Architecture Description Language

Experimental Results

- 32-bit processor *Codix*
  - 7 pipeline stages
  - RISC instruction set
- Simulated instruction counts



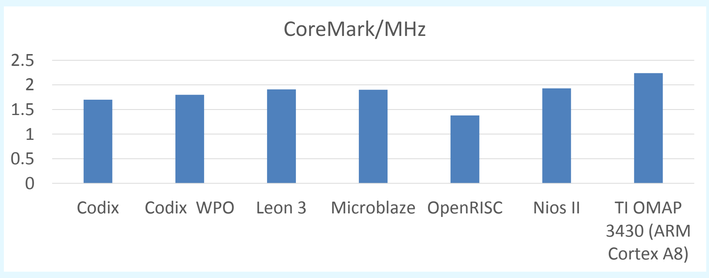




All benchmarks were compiled with -O3 optimization level.
Compilers used:
- arm-gcc 4.8.1
- microblaze-gcc 4.8.1
- arc-gcc 4.8.0
- codix-llvm 3.2 (codasip)
- arm-llvm 3.2 (official)

Simulators used:
- Codasip intersim 3.0.1
- Open Virtual Platforms build 20130630

- Hardware performance comparison

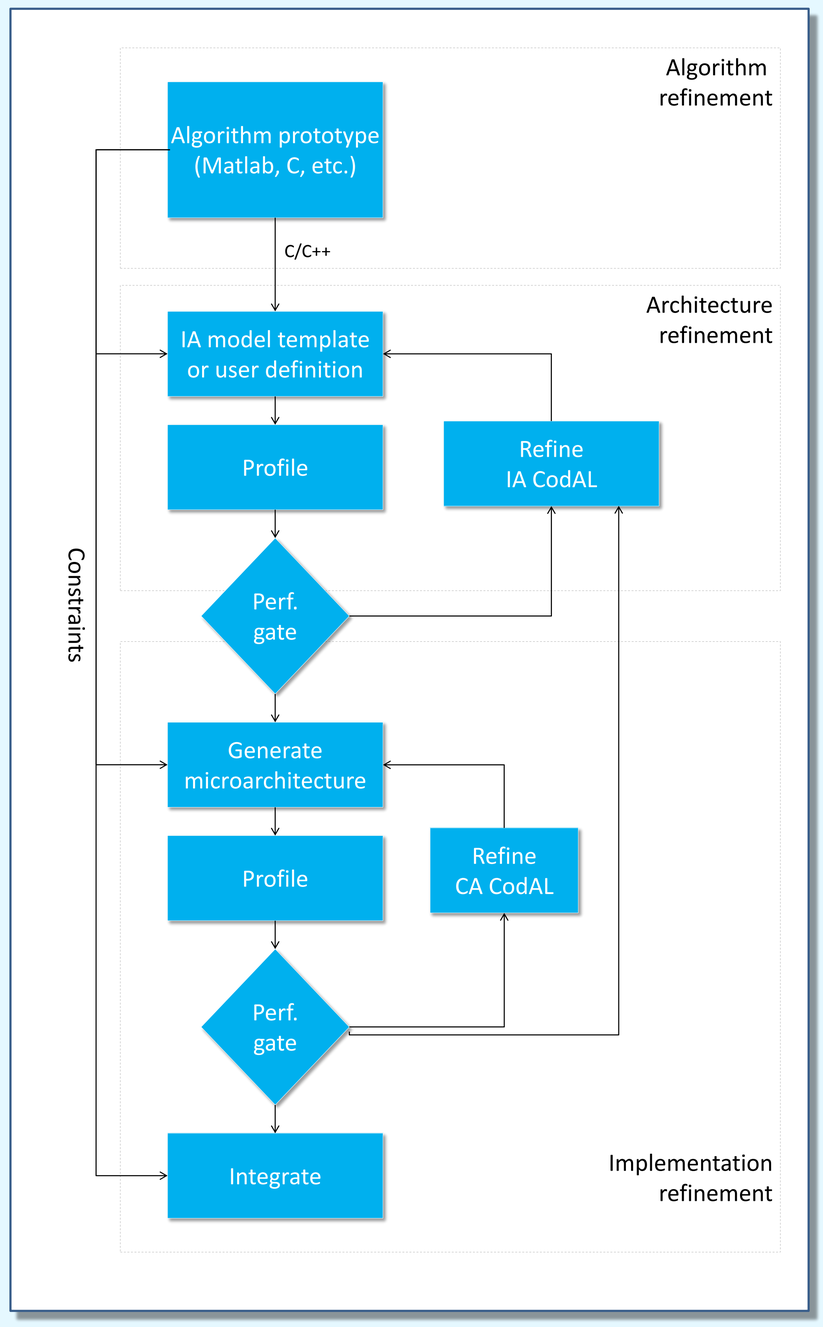


Coremark benchmark was compiled with -O3 optimization level.
Compilers:
- leon-gcc 4.4.2, microblaze-gcc 4.1.2, open-risc-gcc 4.5.1, nios-gcc 4.2.1
- codix-llvm 3.2

Codix hardware: 100 Mhz, Virtex 5, 16kB instruction and 16kB data caches
Results for Leon 3, Microblaze, OpenRISC, Nios II and TI OMAP 3430 are from [1]

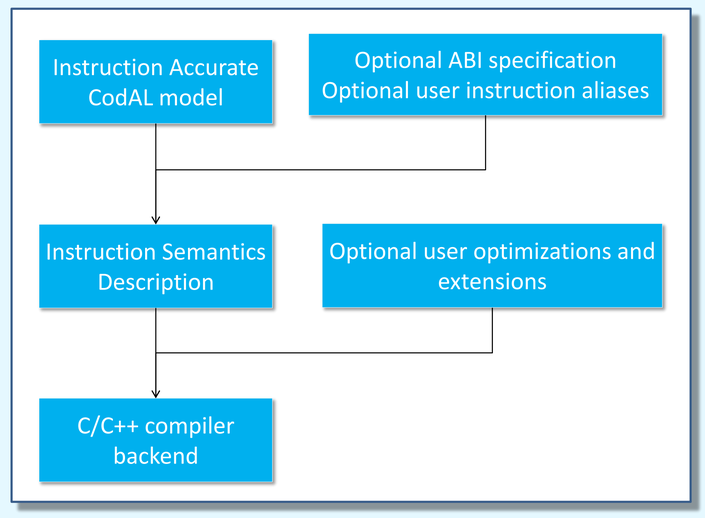
ASIP Design with Codasip® Framework

- *CodAL* architecture description language is used for ASIP design
- Automated generation of the tool-chain and the HDL description of the processor from *CodAL*
- One needs to verify that the C/C++ compiler generates correct executable files
- **Instruction-accurate model** allows generation of
  - programming tools with the C/C++ compiler,
  - simulation tools,
  - golden model for verification
- **Cycle-accurate model** allows generation of
  - programming tools without the C/C++ compiler,
  - simulation tools,
  - HDL description,
  - verification environment



Automatic Generation of C/C++ Compiler

- LLVM-based
- The same processor model is used to generate simulator and compiler backend
- Automatic generation of instruction selector, register allocator with spilling, scheduler and other passes
- Profile-guided superblock formation
- Scheduling and bundle formation for VLIW architectures



Conclusion

- Code quality comparable with hand-written compilers with many extensions
- gcc-compatible compiler driver with modified GNU binutils
- Fast and easy porting of the Newlib standard C library
- Production-quality Codasip® Framework for ASIP design can be obtained from www.codasip.com, free academic license

References

[1] Sven-Ake Andersson: Four soft-core processors for embedded systems, Realtime Embedded, 8th Jan 2013, http://www.eetimes.com/document.asp?doc_id=1280290&page_number=6

This work was supported by the European Social Fund (ESF) in the project Excellent Young Researchers at BUT (CZ.1.07/2.3.00/30.0039), the IT4Innovations Center of Excellence (CZ.1.05/1.1.00/02.0070), Brno Ph.D. Talent Scholarship Programme, the BUT FIT project FIT-S-11-2, research plan no. MSM0021630528.