OptimoDE: Programmable Accelerator Engines Through Retargetable Customization

Nathan Clark, Hongtao Zhong,
Kevin Fan, Scott Mahlke

CCCP Research Group
University of Michigan
http://cccp.eecs.umich.edu

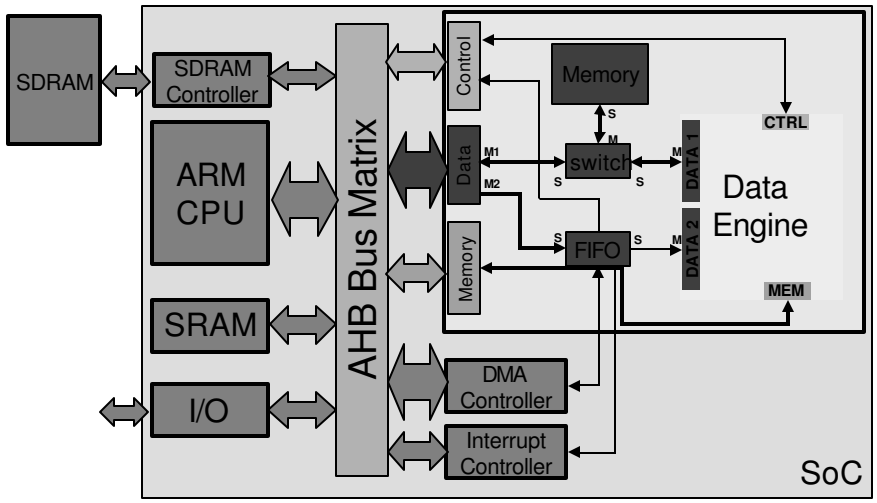
Krisztián Flautner,
Koen Van Nieuwenhove

ARM Limited

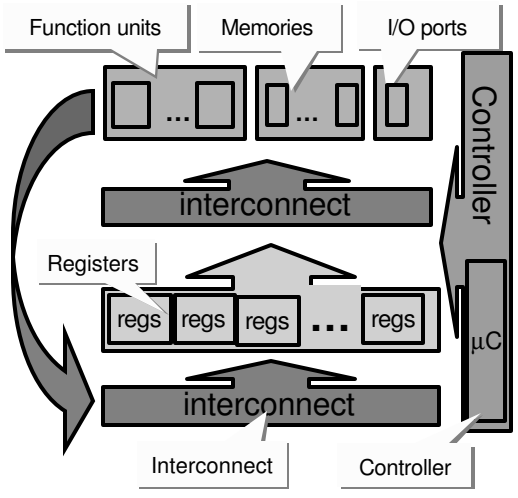
OptimoDE Overview

- OptimoDE
  - A configurable VLIW-styled Data Engine architecture
  - Targeted at intensive data processing
- Characteristics
  - Very wide performance envelope
    - Power / area / speed tradeoff
    - Exploiting parallelism in applications
  - Unlimited data path configuration options
  - User extensible through ISA customization
- Semi-automatic design system
  - User-in-the-loop design, retargetable compiler toolchain

OptimoDE in a System On Chip



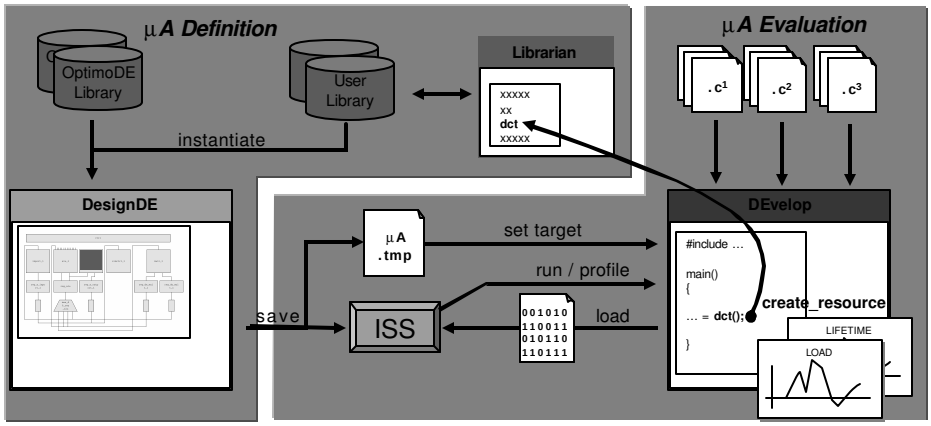
OptimoDE Architecture Model



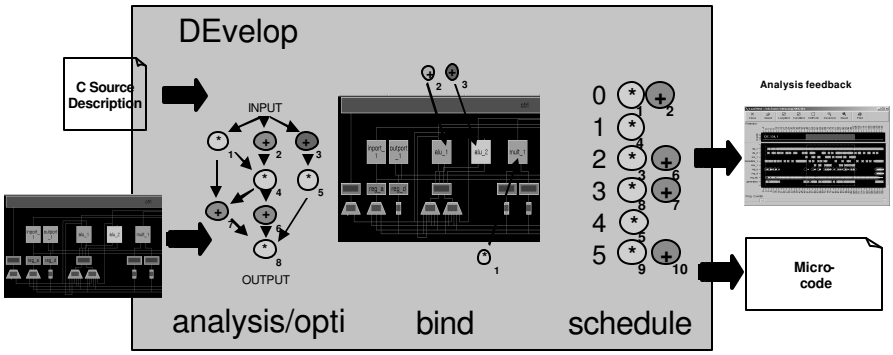
- Functional Units
  - ALU, ACU, Multipliers
  - Custom
- Memory
  - RAM (asynch / synch)
  - ROM
- I/O ports
  - addressable
  - handshake protocol
- Registers
  - Register files
- Interconnect
  - Direct connection
  - Shared bus
- Controller

- All layers required
- Intra-layer configuration

Design Toolchain

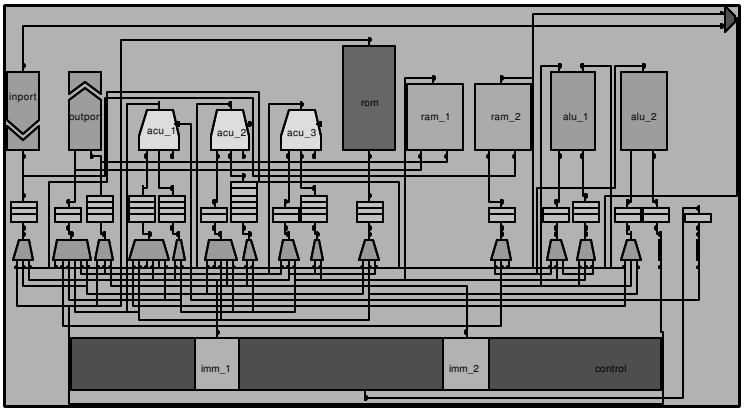


Compiler Toolchain



Syntax checks Dataflow analysis	Match architecture and dataflow graph	Optimize code and register use

32-point DCT Microarchitecture



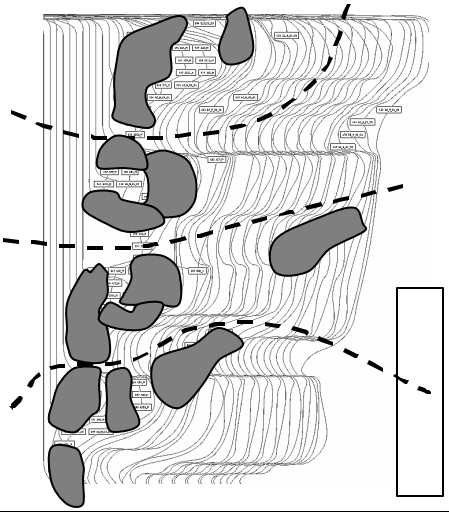
- 2 Custom FUs, 2 RAM, 1 ROM, 3 ACU, 2 I/O ports
- Designer responsible for creating custom units manually

Retargetable Customization

- Prototype 2 technologies in OptimoDE
  - Automated ISA customization
  - Retargetable customization to an "application-area"
- Customizing for 1 application
  - Programmability → Nominally programmable
  - **Critical problem – Cannot sustain performance across similar applications**
  - How well does a custom ISA generalize
    - 5 encryption algorithms, create custom design for each
    - Average loss >80% versus native [MICRO, 2003]
  - Proactive generalization creates a retargetable design

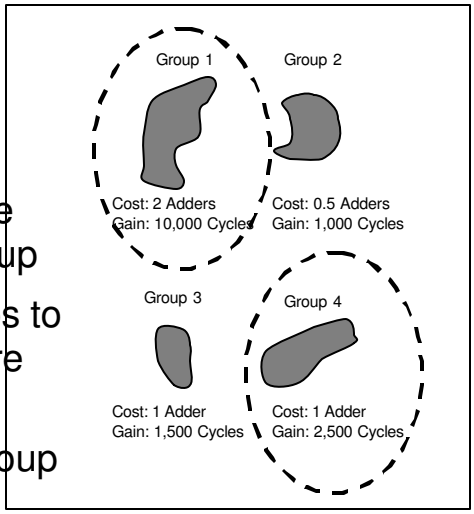
Creating Custom Instructions

- Candidate discovery
  - Identify customization opportunities
- Examine program DFG
- Partition DFG at:
  - Memory operations
  - Unprofitable edges
- Enumerate candidate subgraphs within each partition



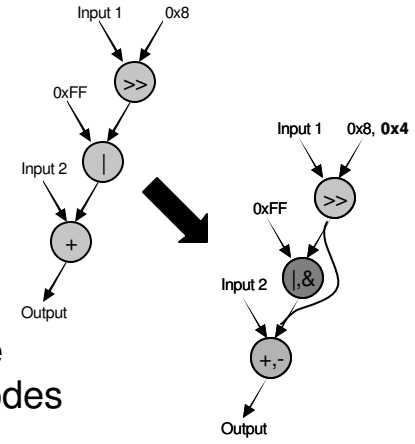
Grouping and Selection

- Group candidate subgraphs with same structure
- Estimate performance and cost for each group
- Greedily select groups to implement in hardware subject to budget
- 1 CFU created per group

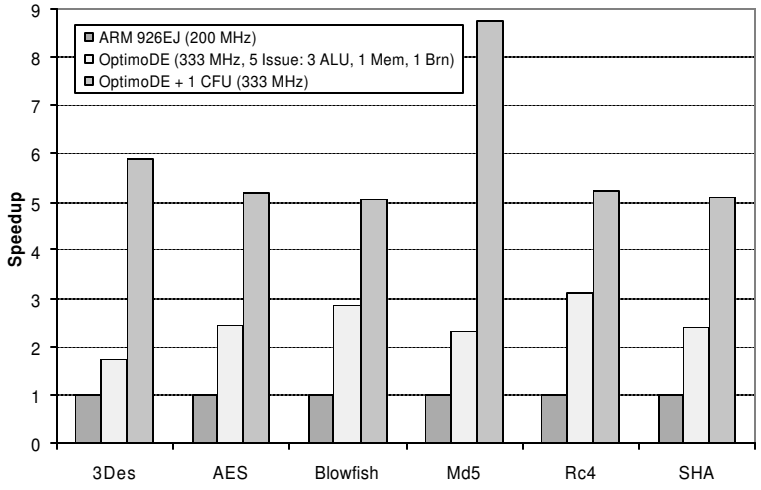


Proactively Generalize Groups

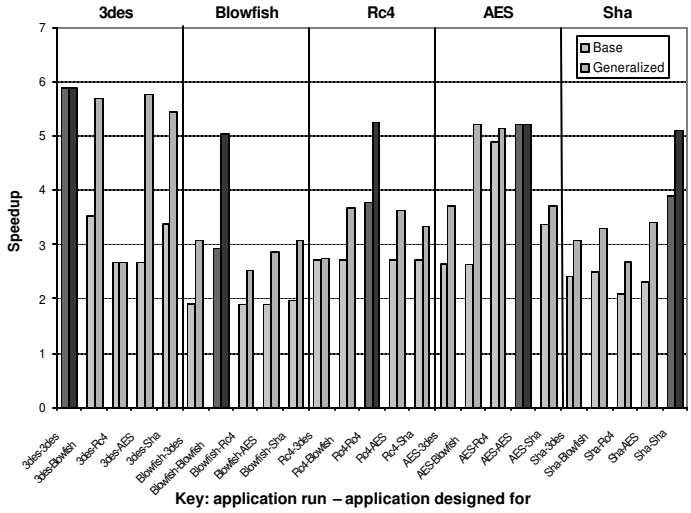
- Cost-effectively extend group functionality to enable reuse
- Wildcard – multiple functionality at nodes
- Subsumed – configurable interconnect to bypass nodes



Native Speedups

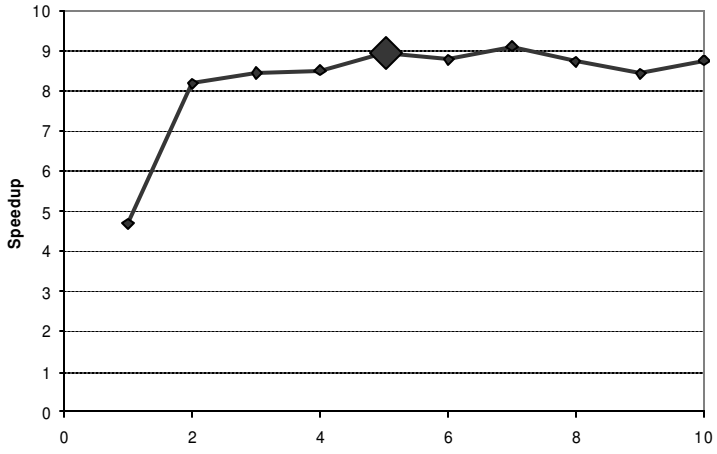


Importance of Generalization



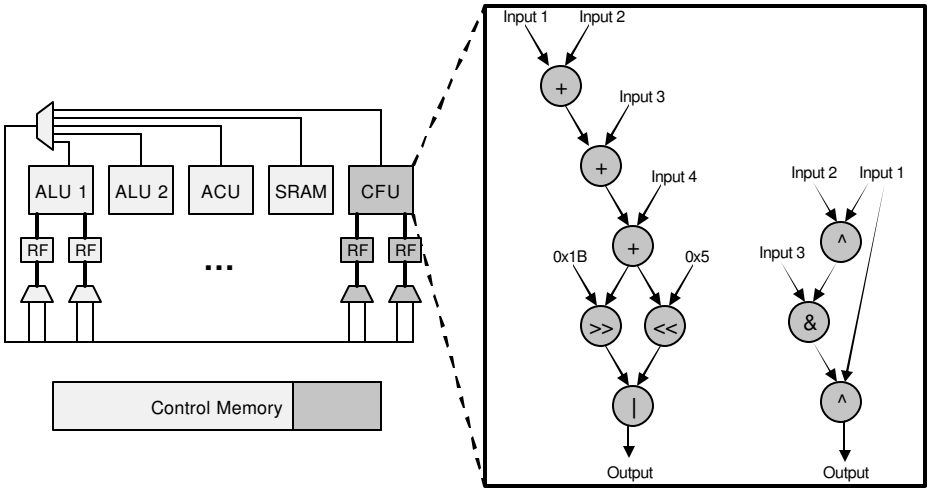
Key: application run – application designed for

Case Study - Md5



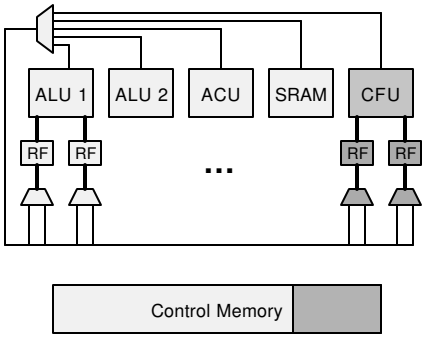
CFU Cost Budget (Relative to 32-bit ripple-carry adder)

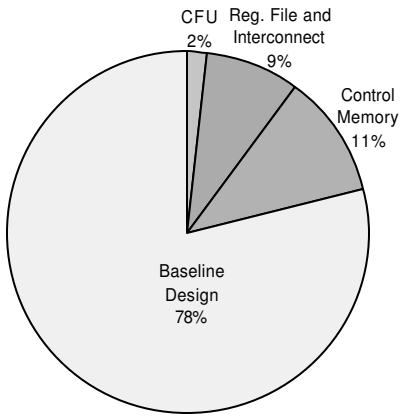
OptimoDE Design for this Point



Die Area Breakdown

OptimoDE = 5.5 mm^2 in 0.13 µ
ARM 926EJ = 5.0 mm^2 in 0.13 µ





Conclusions

- OptimoDE
  - Configurable VLIW-style data engine architecture
  - Automated tools for implementing embedded signal and data processing solutions
- Automatic retargetable customization
  - Customized design combined with cost-effective generalization
  - Performance programmability - Performance stability across a family of similar applications

For More Information

- CCCP group website
  - cccp.eecs.umich.edu
- ARM OptimoDE information
  - www.arm.com/products/CPUs/families/OptimoDE.html

Designing for a Domain