Power Efficiency and Performance for Embedded and HPC Systems with Custom CGRAs

Nuno Paulino (nuno.m.paulino@inesctec.pt), João Canas Ferreira (jcf@fe.up.pt), João M.P. Cardoso (jmpc@fe.up.pt)
INESC TEC & University of Porto, Porto, PORTUGAL

João Lopes (joao.d.lopes@tecnico.ulisboa.pt), Mário Véstias (mvestias@deetc.isel.ipl.pt), José T. Sousa (jose.desousa@inesc-id.pt)
INESC ID & ISEL & University of Lisbon, Lisbon, PORTUGAL

1. Introduction

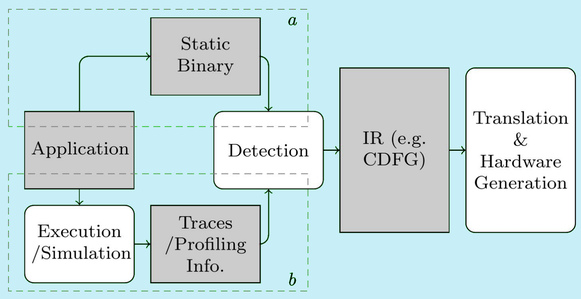
The embedded and HPC domains are distant, but some of their requirements are converging:

- Embedded systems run applications requiring ever increasing computational power
- HPC systems require new levels of power efficiency
- **Both domains require better compute performance per energy cost**

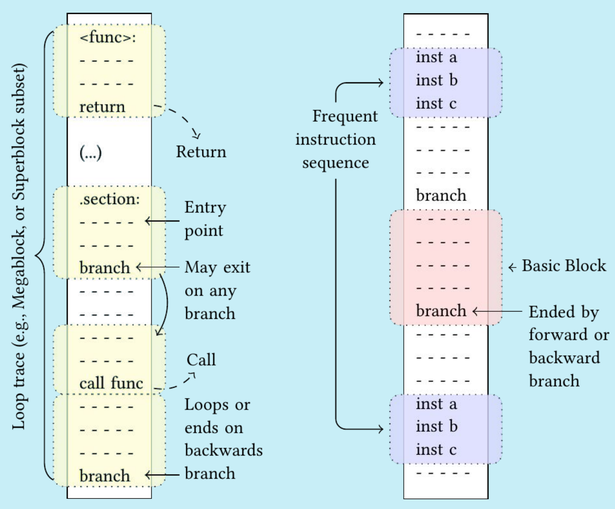
The goal of this project is to devise efficient techniques for dynamically mapping computations extracted from execution behavior to the resources of specialized reconfigurable accelerators.

2. Binary Workload in Embedded Applications

- Detect workload in instruction traces
- **Augment host processor with automatically generated specialized heterogeneity**



- **Binary Segments [3]**
  - Different types of instruction sequences
  - Detected automatically from profiling
  - Translated into **specialized hardware**



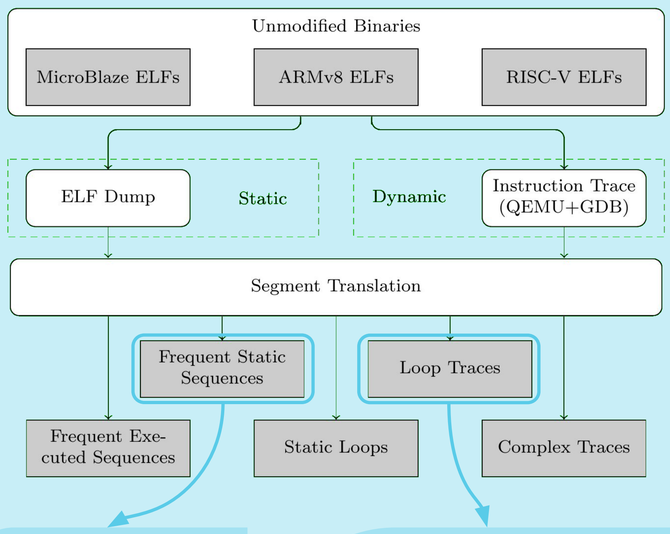
More on binary acceleration approaches:

- N. Paulino *et al.* 2020. *"Improving Performance and Energy Consumption in Embedded Systems via Binary Acceleration: A Survey"*, **ACM Computing Surveys** 53, 1, Article 6 (February 2020), 36 pages
- N. Paulino, (2020): A Breakdown of Binary Acceleration Approaches and Systems. INESC TEC. (DataPaper). https://doi.org/10.13140/RG.2.2.27223.62886

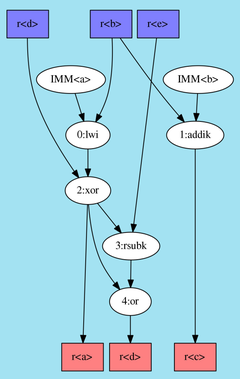
3. A Binary Translation Framework for Automated Hardware Generation

- Current Binary Translation Framework Features
  - Process ELF files or instruction traces
  - Decoding of MicroBlaze and ARMv8 instruction fields and operands
  - Detection of four types of segments
  - Detection of recurrent locations and iteration counts of repeating segments
  - Conversion to CDFG representations:
    - Further optimization
    - Retargeting to **heterogeneous hardware**

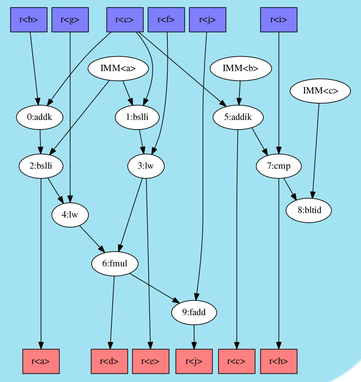
- **Binary Translation Framework repository**
  - https://github.com/specs-feup/specs-hw



Example #1
Segment type: acyclical
Number of nodes: 5
Number of memory reads: 1
Number of memory writes: 0
Maximum ILP of graph: 2
Critical Path Length: 4
Max IPC: 5.0



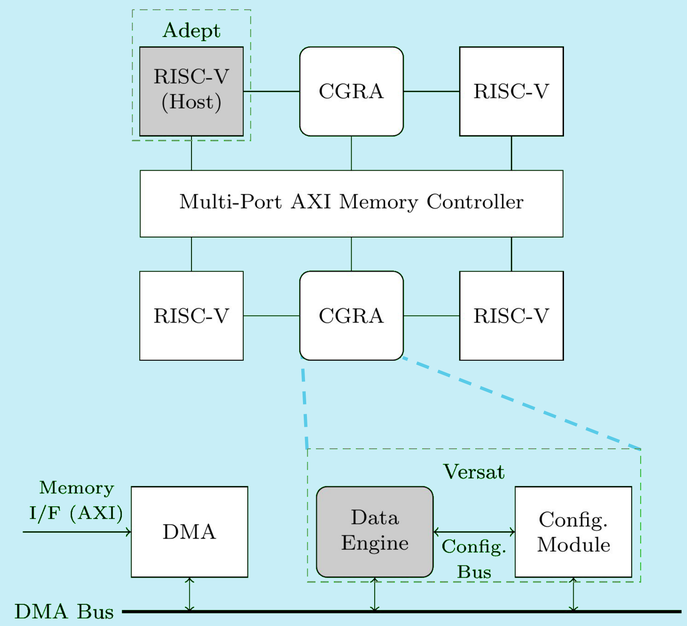
Example #2
Segment type: cyclical
Number of nodes: 10
Number of memory reads: 2
Number of memory writes: 0
Maximum ILP of graph: 3
Critical Path Length: 5
Initiation Interval: 3 → Max IPC: 3.3



4. Heterogeneous Computing with Multiple RISC-V and CGRA Cores

- For embedded HPC the use of GPUs and FPGAs as acceleration engines may exceed the area power/energy budgets

- CGRAs are an interesting alternative:
  - Customized engines
  - Enough performance at a fraction of the cost and energy efficiency



- Versat CGRA Features [5]
  - Targets runtime compilation of configurations
  - Easy to program by non-hardware experts via assembly or C++ API
  - Linear array of small full mesh CGRA nodes
    - Exploits loop techniques such as unrolling, tiling, and interchange
    - Frequency of operation is independent of configuration due to enforced pipelining

- **Open source MIT license repositories**
  - RISC-V System on Chip + Versat CGRAs
    - https://github.com/jjts/iob-soc
  - Versat CGRA
    - https://github.com/jjts/versat

5. Results & On-Going Work

- Acceleration of MicroBlaze loop traces [1,2]:
  - 5.6x geo. mean speedup vs. MicroBlaze
  - 1.8x geo. mean speedup vs. 4-issue VLIW

- Estimated ILP potential in ARMv8 applications
  - 4.8, in Basic Blocks from traces

- RISC-V SoC+Versat@65nm vs. ARM A9@40nm
  - ~2x less power consumption
  - ~3x less silicon area

On-Going Work

- Segment optimization and extraction:
  - Support for RISC-V ISA
  - Memory access pattern analysis
  - Segments representing nested loops, Segments representing multi-path loop traces

- Generating loop and subgraph accelerators:
  - At runtime by component assembly via DPR
  - For ARMv8 on UltraScale+ MpSoC devices
  - For standalone RISC-V cores
  - **For our RISC-V+CGRAs designs**

1. N. Paulino, J. C. Ferreira and J. M. P. Cardoso, "Generation of Customized Accelerators for Loop Pipelining of Binary Instruction Traces", in *IEEE Trans. on Very Large Scale Integration Systems*, vol. 25, no. 1, pp. 21-34, Jan. 2017
2. N. Paulino, J. C. Ferreira and J. M. P. Cardoso, "Dynamic Partial Reconfiguration of Customized Single-Row Accelerators", in *IEEE Trans. on Very Large Scale Integration Systems*, vol. 27, no. 1, pp. 116-125, Jan. 2019
3. N. Paulino, J. C. Ferreira and J. M. P. Cardoso, "Improving Performance and Energy Consumption in Embedded Systems via Binary Acceleration: A Survey", ACM Computing Surveys 53, 1, Article 6 (February 2020), 36 pages
4. Daniel Granhão, "Transparent control flow transfer between CPU and Intel FPGAs", 2019, Universidade do Porto, Faculdade de Engenharia
5. João D. Lopes and José T. de Sousa, "Versat, a Minimal Coarse-Grain Reconfigurable Array", in *Proc. of the 12th Int. Meeting on High Performance Computing for Computational Science*, VECPAR, Porto, Portugal, June 2016
6. L. Fiolhais *et al.*, "Low Energy Heterogeneous Computing with Multiple RISC-V and CGRA Cores", *2019 IEEE International Symposium on Circuits and Systems (ISCAS)*, Sapporo, Japan, 2019, pp. 1-5.

This work was supported by the PEPCC project, "PTDC/EEI-HAC/30848/2017," financed by Fundação para a Ciência e Tecnologia (Portuguese Foundation for Science and Technology).