RISC-V Dataflow Extension

Authors: Martin Cowley and Lina Sawalha
Western Michigan University

CARRV 2021

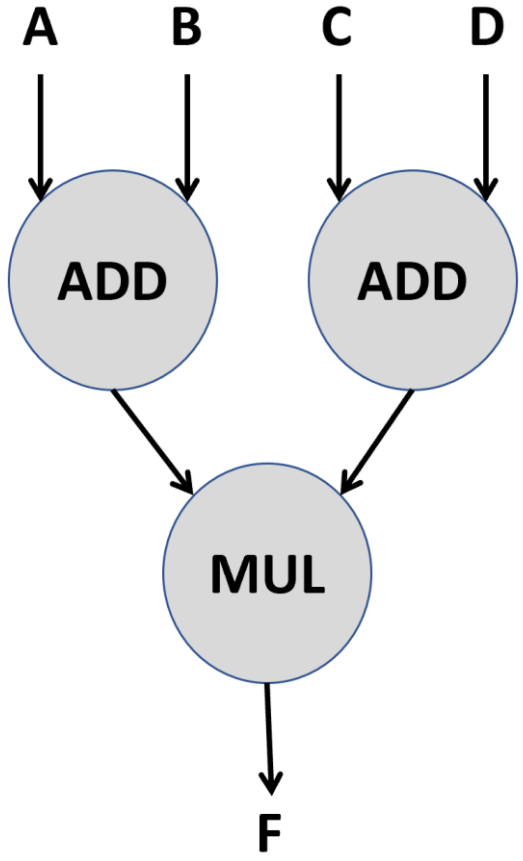
Background

Control Flow

- Algorithm represented as a sequence of instructions
- Instructions fetched in-order
- Instructions might be executed out-of-order
- Maintain control flow (commit in-order)

Dataflow

- Program is a graph of instructions
- Arcs between instructions are true-data dependencies
- Increase Instruction-level-parallelism (ILP): instructions are executed based on the availability of data



Control Flow Challenges

Challenges with out-of-order CPUs

- Extract ILP from control-flow programs
- Dataflow Execution of control-flow programs (limits window size)
- Complex hardware to extract data-dependencies

How to improve performance and minimize energy consumption?

Explicit Dataflow Motivation

True-data dependencies are exposed to the ISA

- Increase window of instructions
- Simplify hardware
- Better at handling irregular applications

Related Work

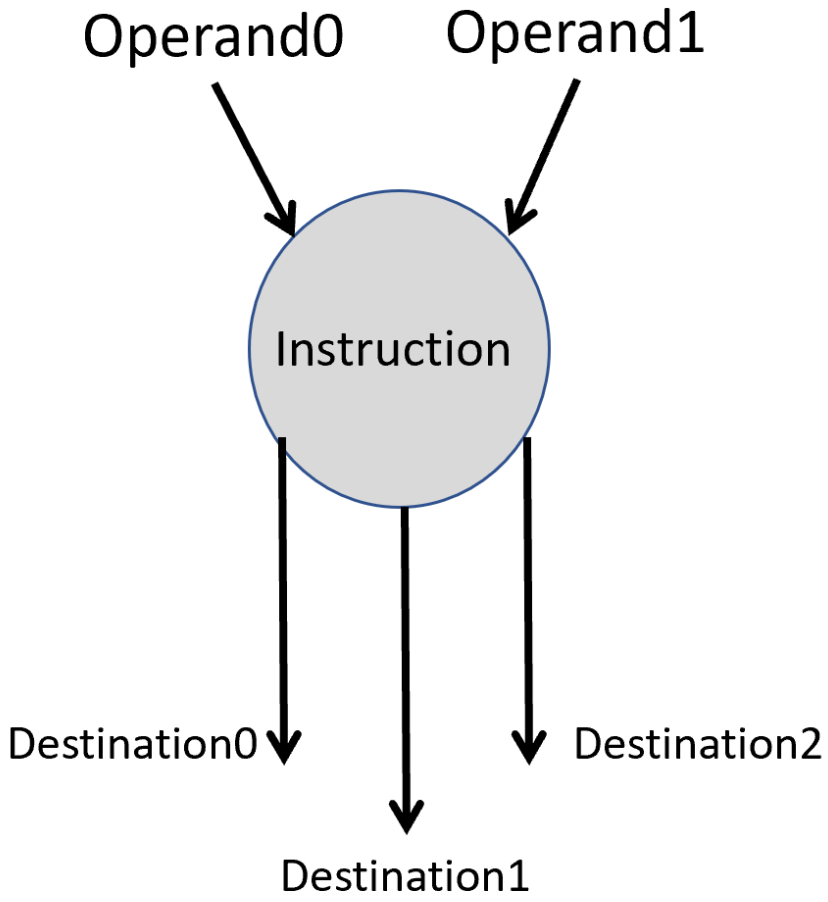
- Explicit Architectures
  - Tagged-token machines (Gurd 1985, Arvind 1990, Papadopoulos 1990)
  - Wavescalar (Swanson 2007)
- Hybrid Architectures
  - TRIPS (Burger 2004)
  - Dyser (Govindaraju 2012)
- Heterogeneous
  - SEED (Nowatzki 2015)

Contributions

- Dataflow sub-ISA
- Simple LLVM-based compiler
- Gem5 simulation model
  - Explicit dataflow model
  - Heterogeneous: existing out-of-order (O3) combined with new dataflow model

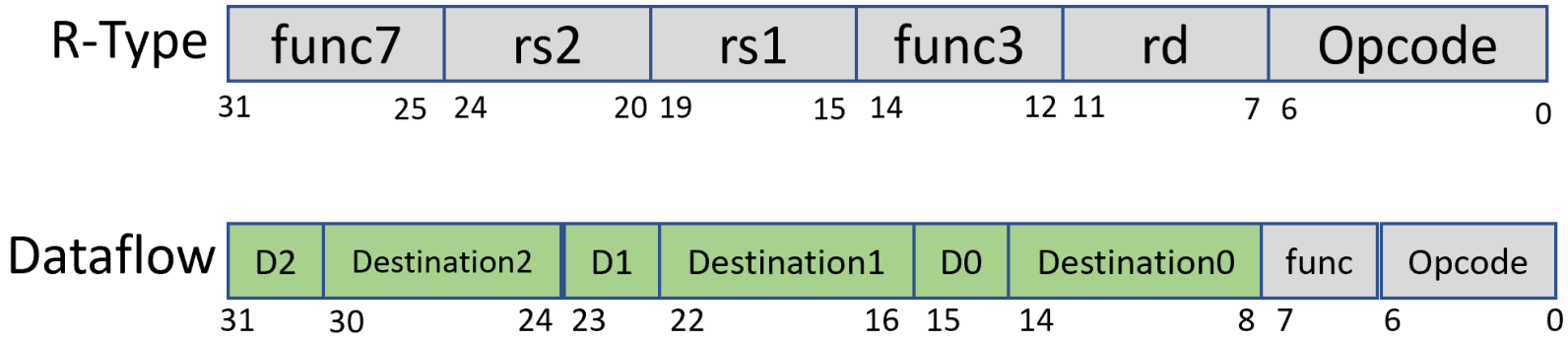
RISC-V Extension

- Max Operands: 2
- Max dependencies: 3
- Dependencies are encoded in the bitfields for the instruction
- Control Dependencies are converted to data dependencies



Instruction Bitfields

- Destination: pointer to dependent instruction
- D2,D1,D0: specifies the left or right operand



Arithmetic and Logic Instructions

Name	Description
df_add	addition
df_sub	subtraction
df_sl	shift left
df_and	bitwise AND
df_not	bitwise NOT
df_mul	multiplication
df_cmpLT	Compare Less Than
df_cmpEQ	Compare Equal
df_cmpGT	Compare Greater Than

Memory Instructions

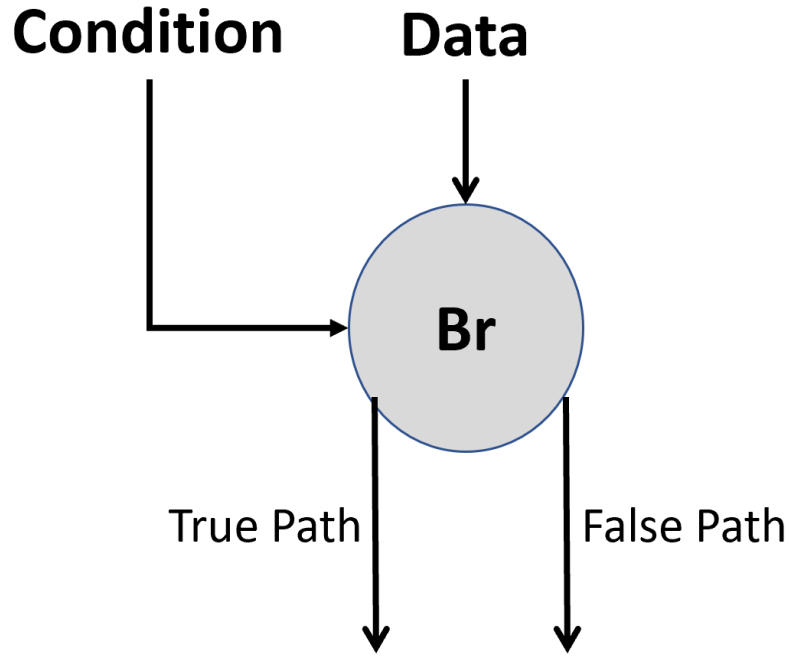
Memory aliasing

- Conservative approach
  - Memory instructions are forced to execute in-order
  - Compiler inserts data dependencies between memory instructions

Name	Description
df_lw	load 64-bit
df_sw	store 64-bit
df_sd	store 32-bit
df_ld	load 32-bit
df_lb	load single byte
df_sb	store single byte

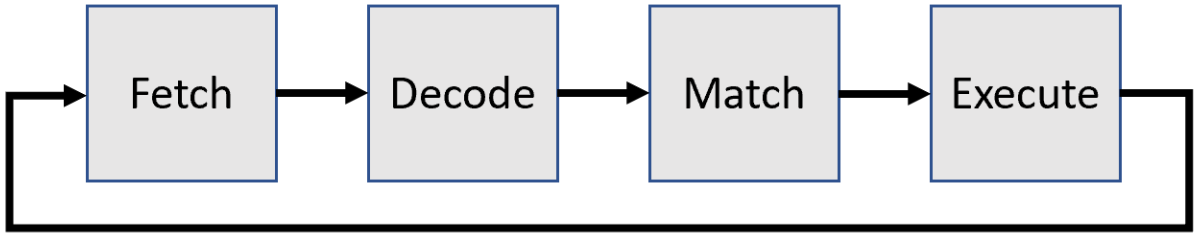
Conditional Instructions

- Control dependencies are converted into data dependencies
- df_br: conditional branch instruction (if-else)
- df_loop_br: used to implement loops
- df_switch: switch between dataflow and Von Neumann execution



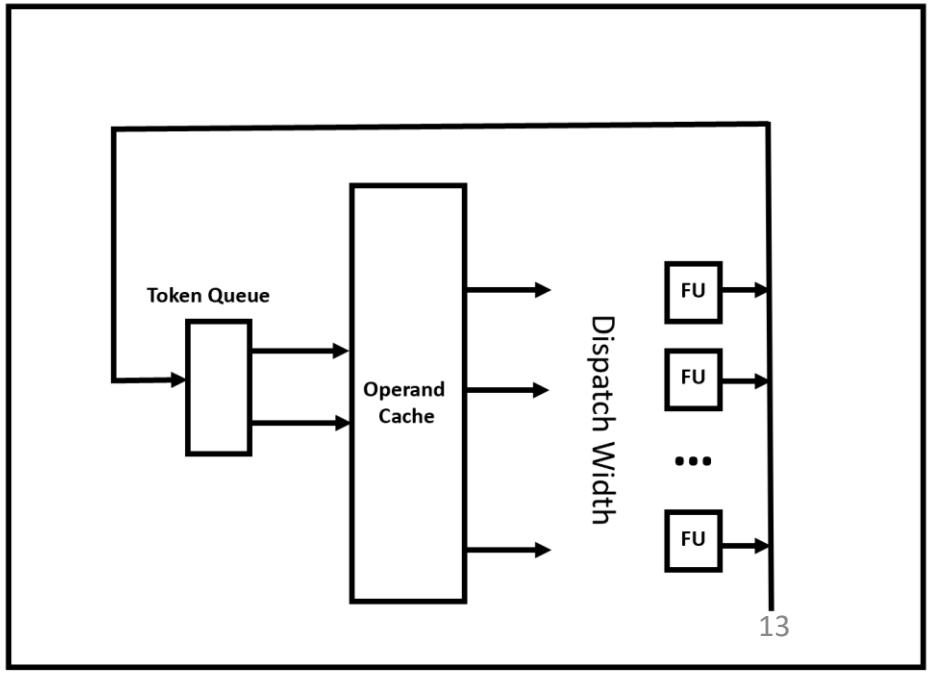
Dataflow Microarchitecture

- Circular pipeline
- Fetch stage: no PC
- Decode stage: no reg file
- Match Unit:
  - 450-entry token cache
- Simple loop predictor



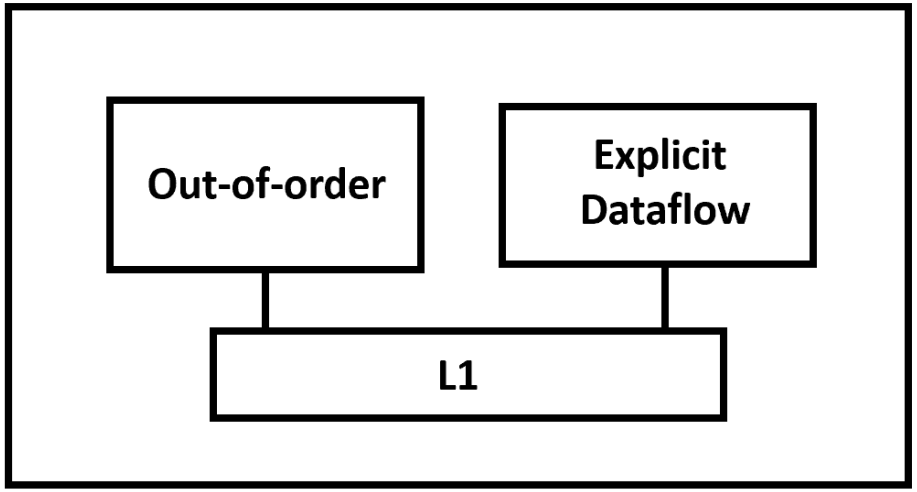
Instruction Scheduling

- Operands are stored in the cache until all operands have arrived for the corresponding instruction
- Dispatch: instructions are dispatched to functional units (FUs) when all operands have reached the token cache
- The output to the FUs are placed in the token queue to later be processed by the operand cache



Heterogeneous Microarchitecture

- Switches between OoO and Explicit dataflow execution
- Shared L1 Cache
- Special dataflow instructions allow the two architectures communicate



Experiments

- Added heterogeneous model to *gem5*
  - Combination of dataflow core + existing O3 model
- LLVM backend
  - Generates dataflow graphs from C/C++ code
  - more work required to optimize code (optimizations are currently done by hand).
- Microbenchmarks test common programming features
  - Loops, arrays, arithmetic, etc.

Microbenchmarks

Sum array:

- simple loop
- regular memory access

Indirect Sum:

- simple loop
- irregular memory access paterns

Matrix Multiplication:

- Regular memory access (matrix)
- More complected control structures

Results

- Up to **7.5%** improvement
- Worst case: **-2.36%** reduction

(mainly due to fine-grain switching overhead)

Benchmark	% Improvement
Sum Array	7.48%
Matrix Multiplication	-2.36%
Indirect Sum	3.21%

Conclusion and Future Work

- New extension for the RISC-V ISA
- Added a dataflow simulation model to gem5
- Up to 7% improvement over gem5's existing OoO core.
- Future work:
  - Improve compiler to better optimize code and target larger applications
  - Run many benchmarks to measure dataflow benefit for different applications and domains
  - Estimate energy consumption