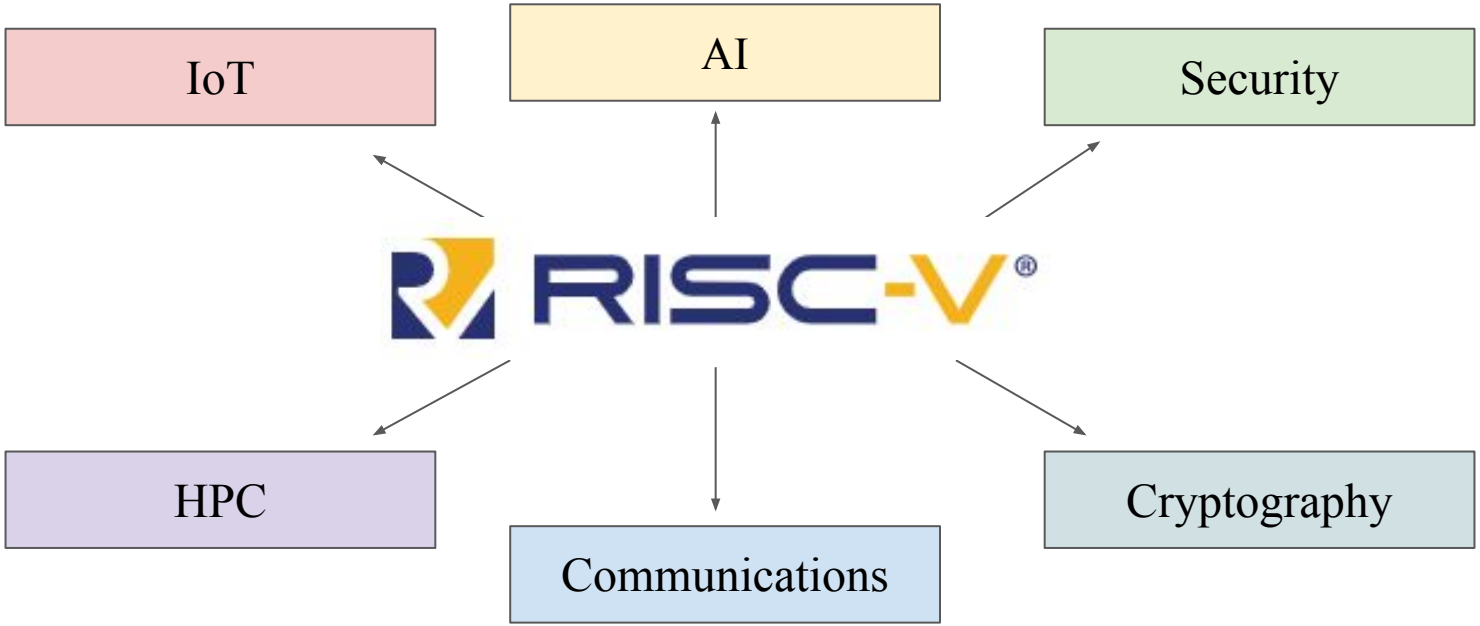
RGen: A Tool for Generating RISC-V Compiler, Simulator, and Application Support

Derek Zijie Tu and Zhangxi Tan

RIOS Lab

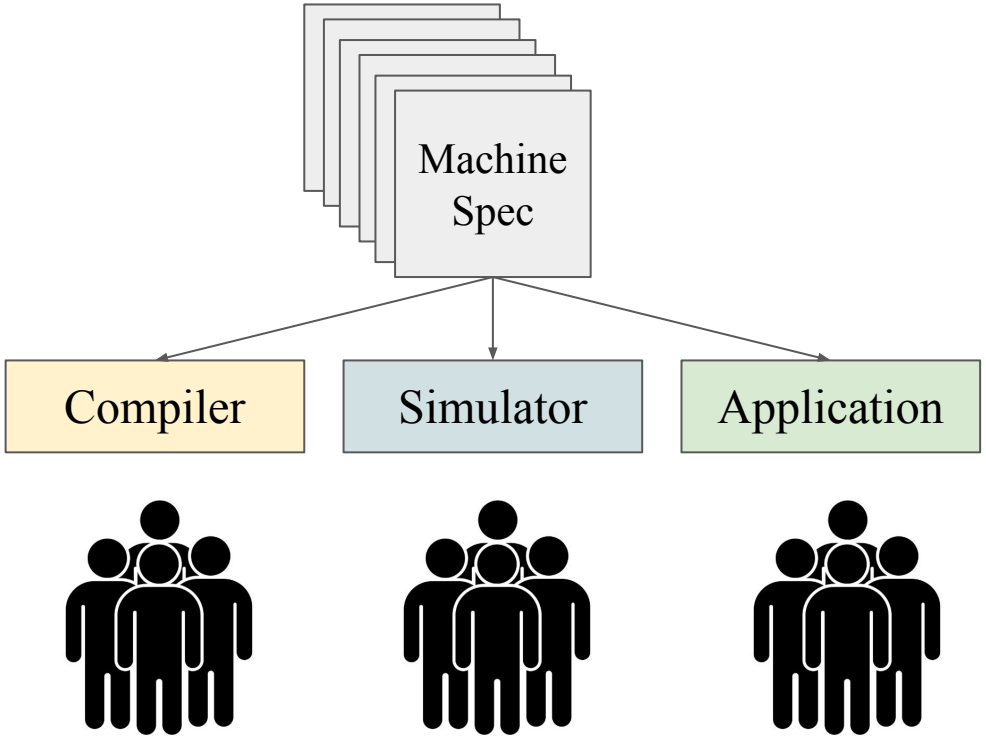
2023.06.17

Instruction Extensions are Important



Adding Instruction Extensions is Painful

- Easy to make mistakes
- Big time sink



Overview

- **RGen**: a RISC-V infrastructure for instruction extension support
- Artifact discussion
- RGen Architecture
  - **RGenIR**: A language extension to Sail
  - Support for targeted artifacts
- Testing
- Discussion

Choosing Artifacts

1) Machine Specification
2) Compiler
3) Simulator
4) Application

Discussion on Sail

Concerns

- Focused on instruction semantics
- Lacks some key target backends
- Simulator not as powerful as others

Our Solution

- Add a new isolated description
- Add the key backends
- Generate support for another simulator

Discussion on LLVM

Concerns

- Large amount of TableGen files
- Hard for new users to understand
- Very prone to mistakes

Our Solution

- Automatically generate support
- Provide support to help guide users in LLVM

Discussion on QEMU

Concerns

- Complex

Our Solution

- Automatically generate support
- Provide support to help guide users in QEMU

Discussion on PyTorch

Concerns

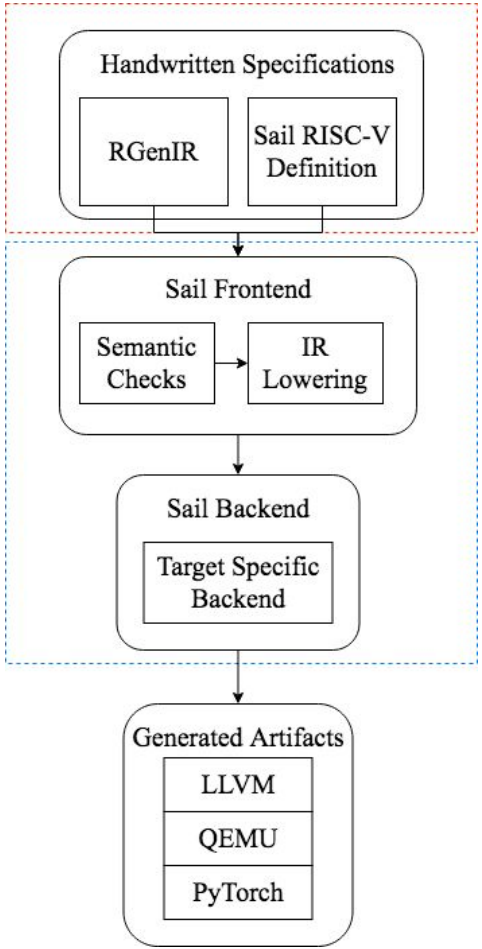
- In regards to RISC-V
- Hard to compile new instructions into PyTorch

Our Solution

- Provide quick and dirty method

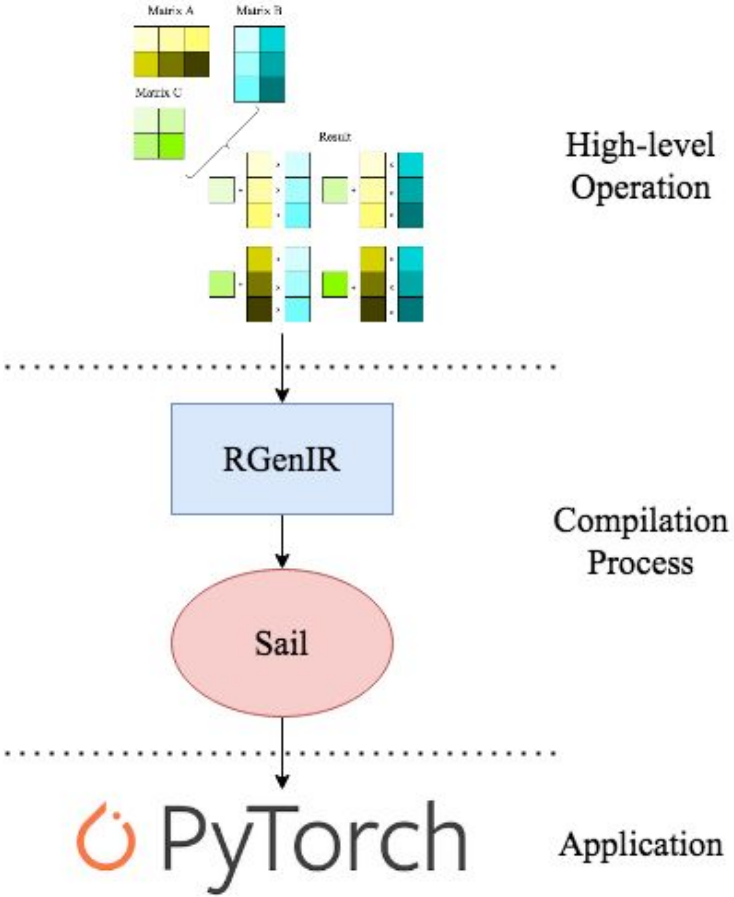
RGen Architecture

- Two handwritten specifications:
  1) RGenIR
  2) Sail RISC-V definitions
- Generate support for:
  1) LLVM
  2) QEMU
  3) PyTorch



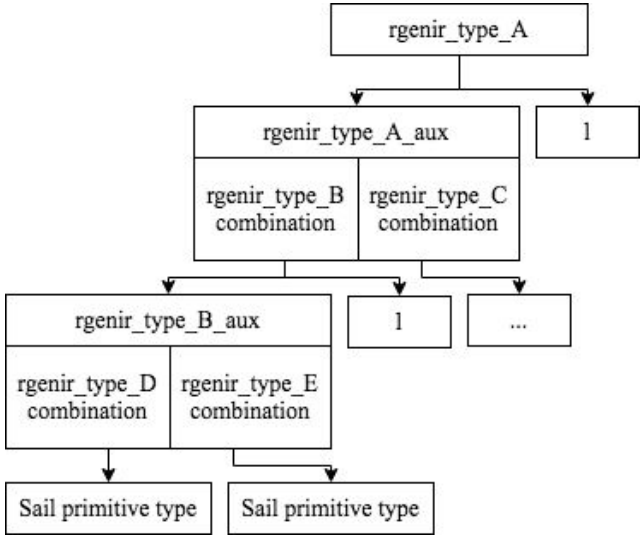
RGenIR

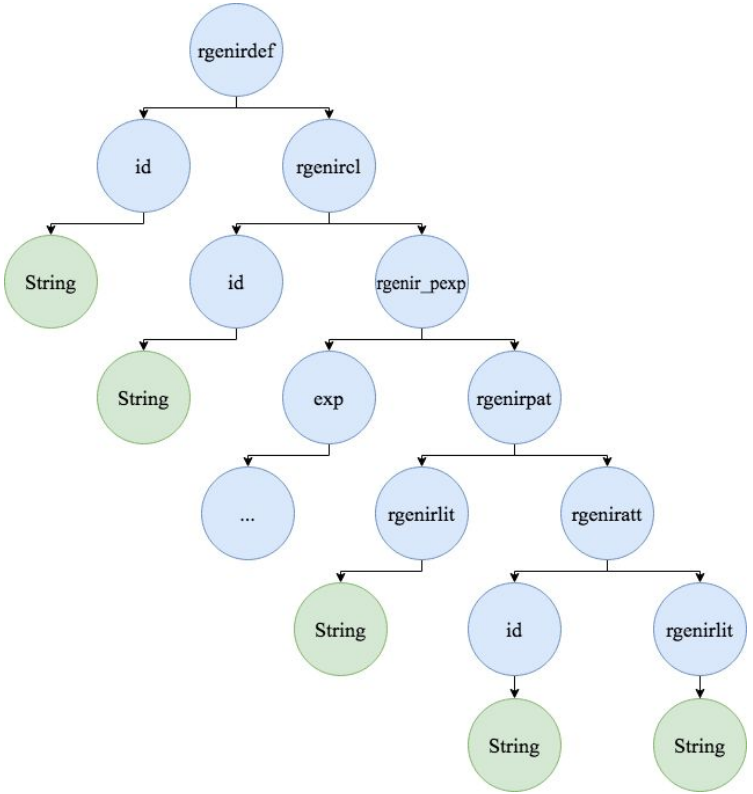
- Language extension to Sail
- High-level descriptor
  - Instruction semantics not always needed
- For high-level operations
  - Combination of assembly-level instructions



RGenIR Pattern and Type Matching

- Simple patterns
- Tree-based traversal

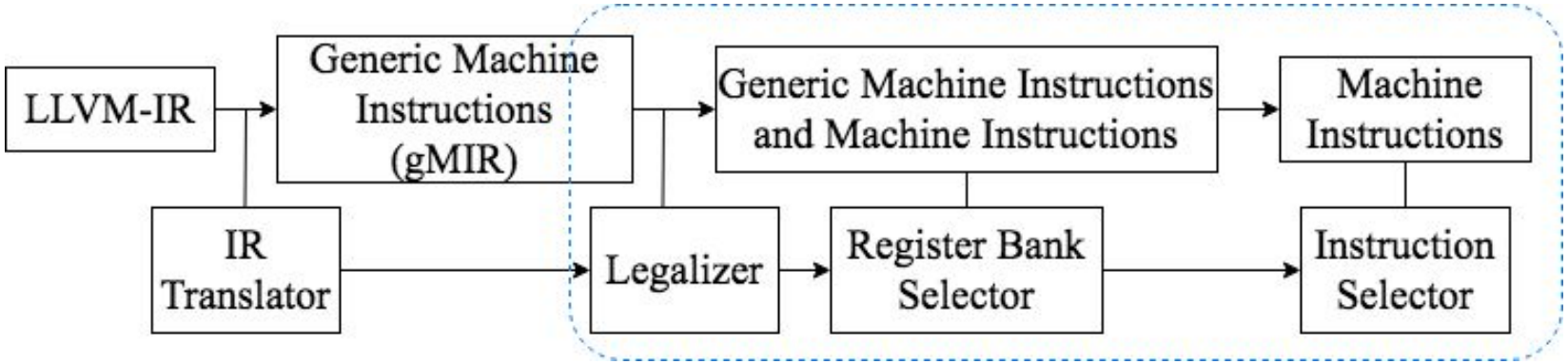




Generating Support for Artifacts

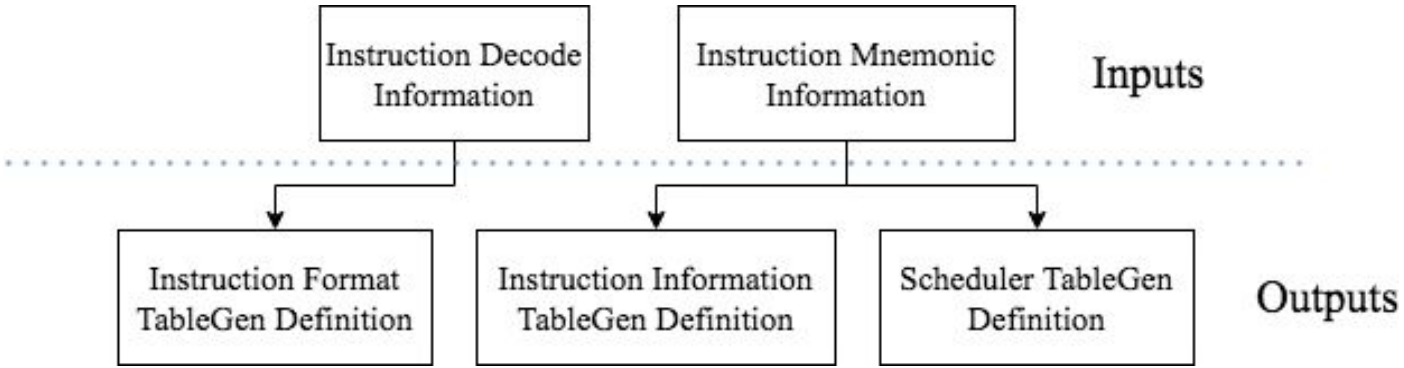
RGen Support for LLVM

- Sail RISC-V definitions
- Help translates gMIR to Machine Instructions
  - Generate instruction semantics for LLVM's TableGen



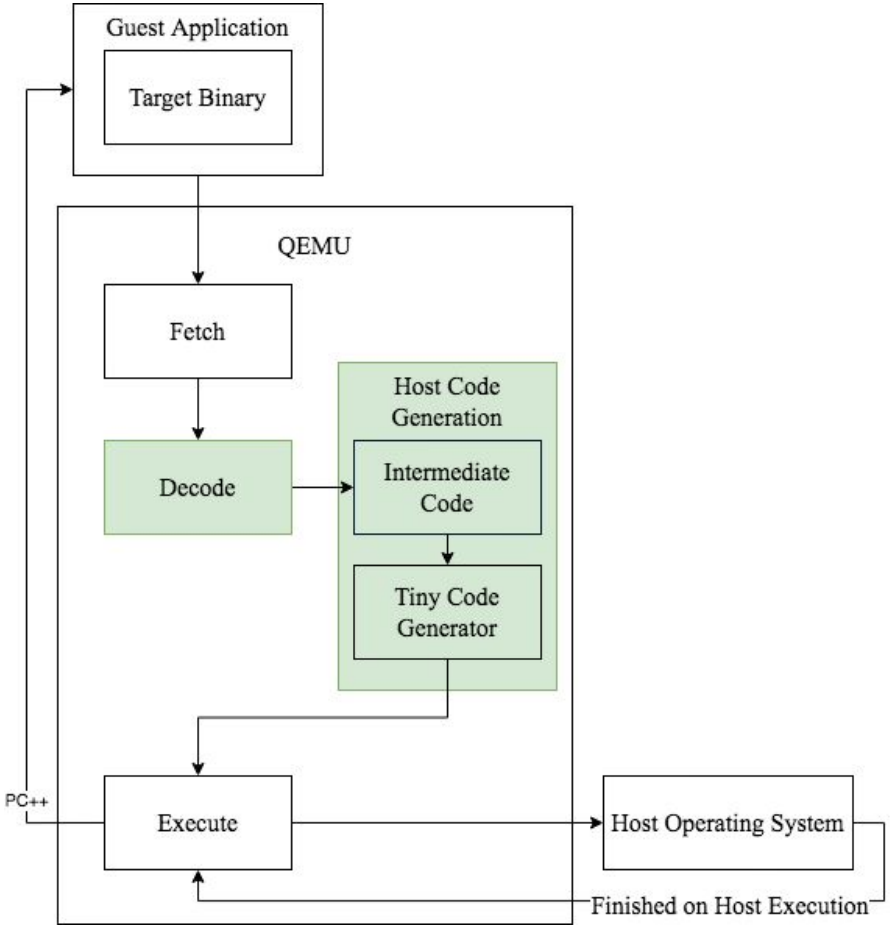
Generated Components for LLVM

- Instruction Format TableGen definition
- Scheduler TableGen definition
- Instruction Information TableGen definition



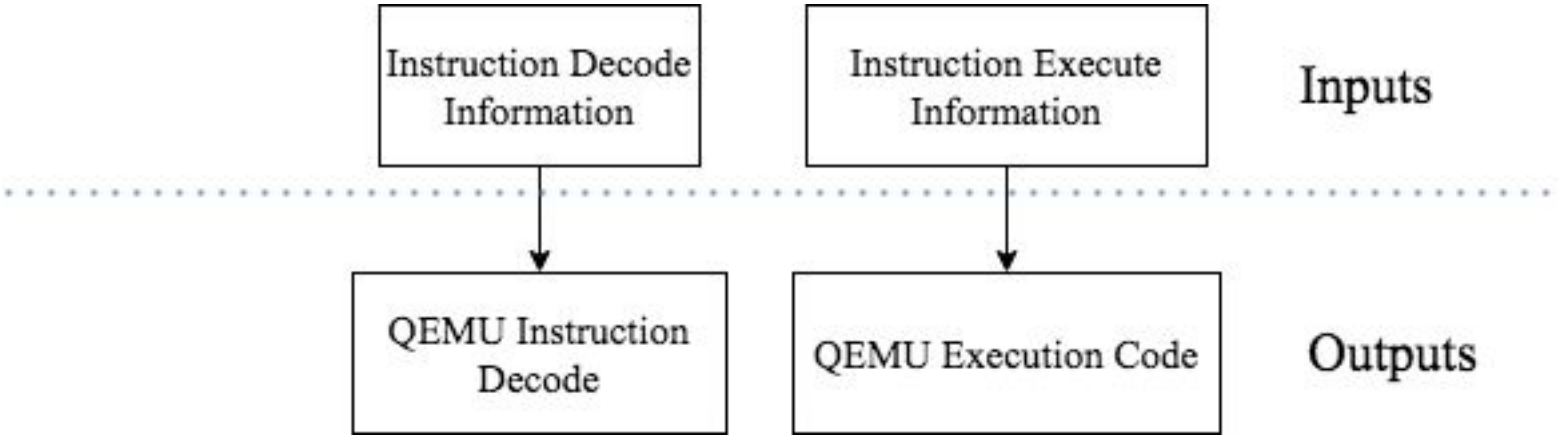
RGen Support for QEMU

- Sail RISC-V definitions
- Decoder
  - Instruction semantics
- Tiny Code Generator
  - Translation functions
  - CPU state manipulators



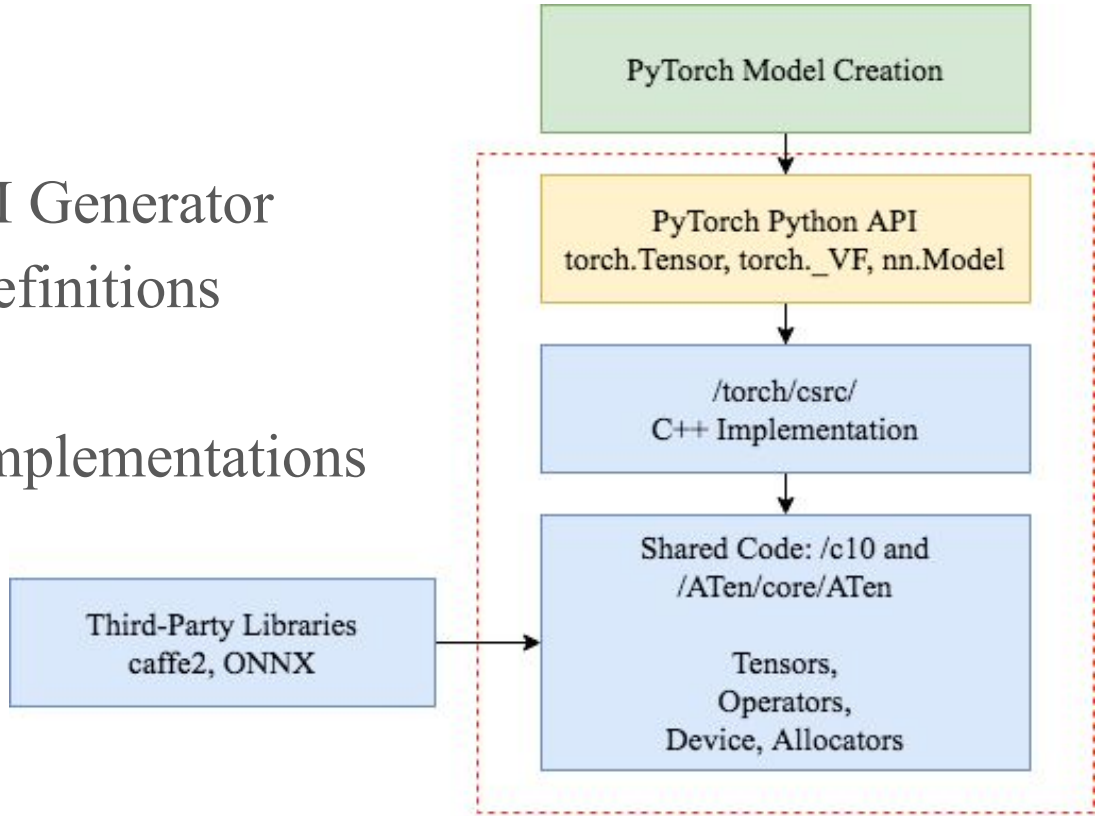
Generated Components for QEMU

- Instruction decode information
- Instruction functionality information



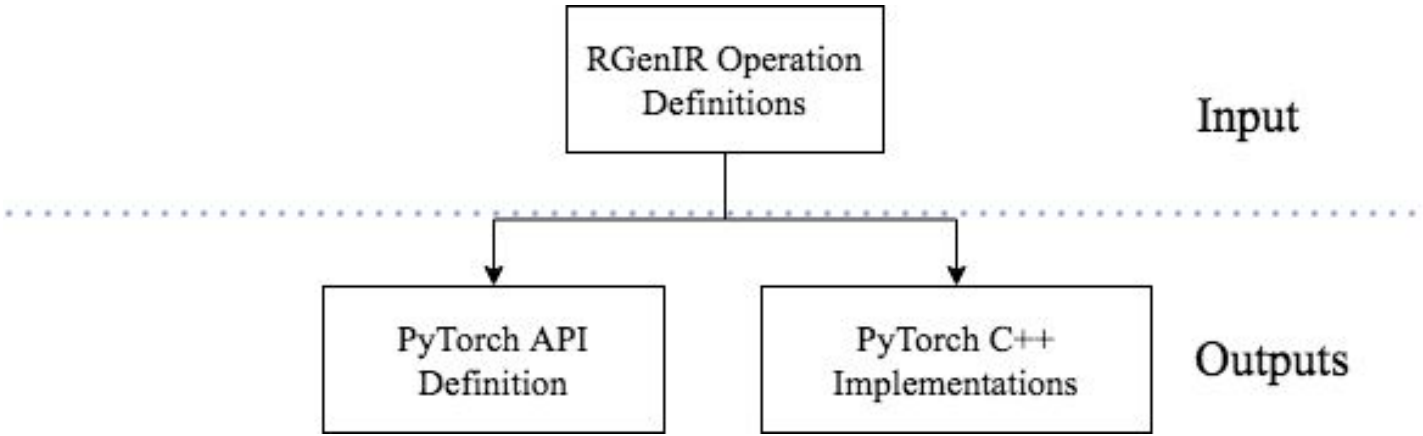
PyTorch Support

- RGenIR
- PyTorch Python API Generator
  - Generate Tensor definitions
- C++ backend
  - Generate Tensor implementations



Generated PyTorch Components

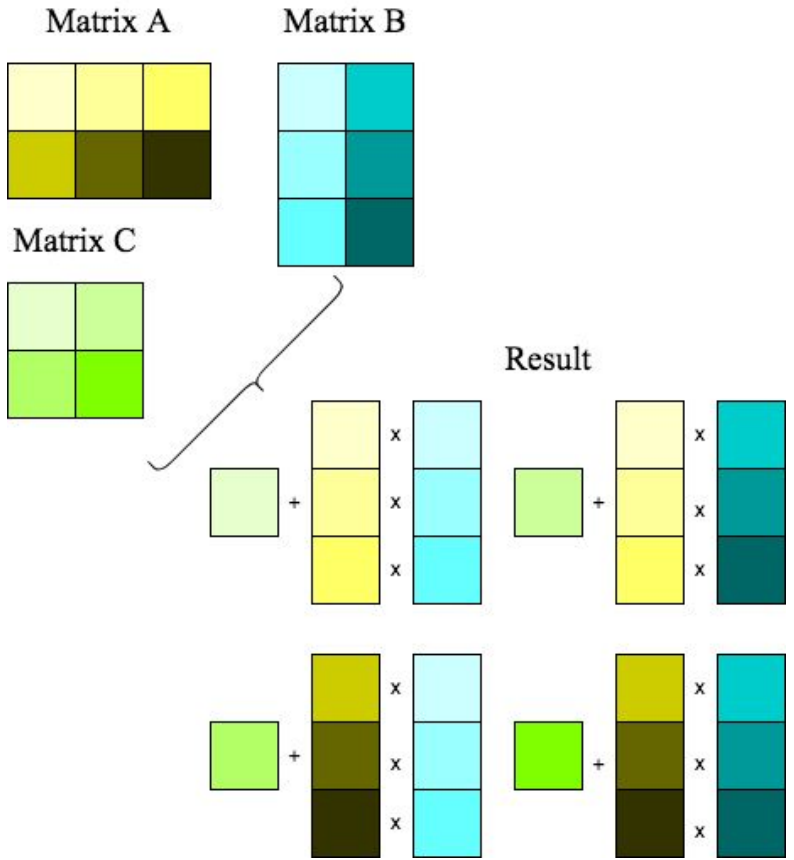
- YAML definitions
- C++ implementation
  - PyTorch API facing implementation
  - External Library



Tests and Evaluations

Choosing an Extension

- Matrix Extension
  - New to RISC-V environment
  - State of the art
- Based on Apple-AMX

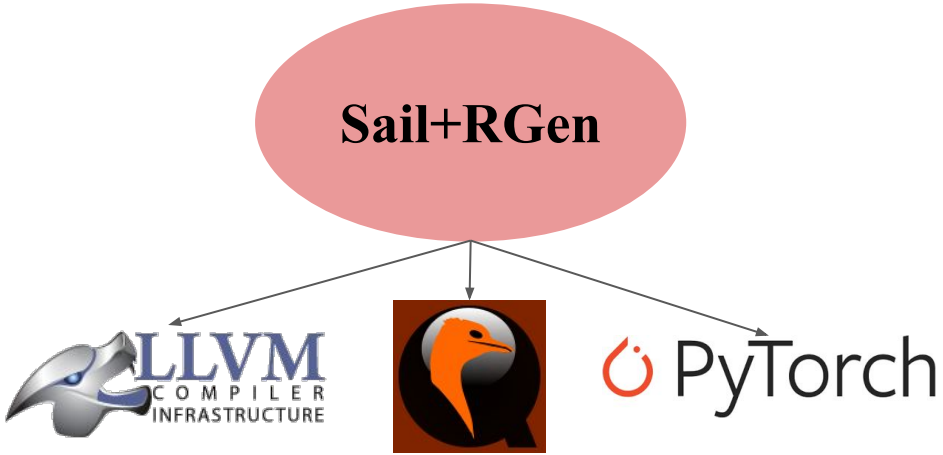


Methodology

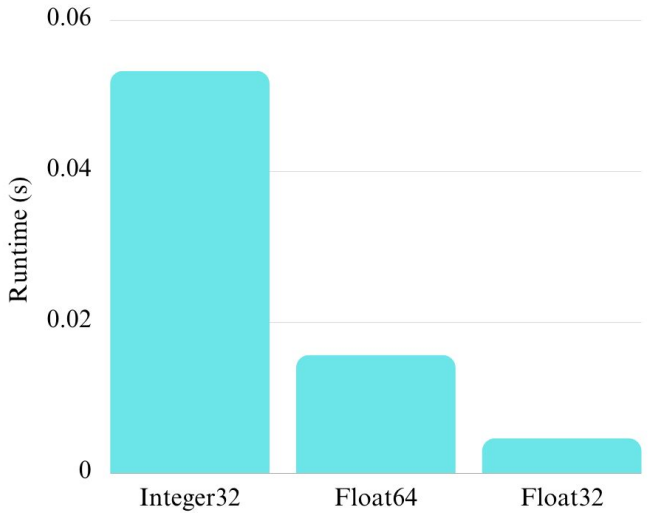
- Cross-compile generated PyTorch operations using LLVM15.0.1
- Run on QEMU 7.2.0
  - Fedora Rawhide 20200108 kernel
  - Fedora RISC-V 64-bit Rawhide drive
  - 64 GB of memory with 4 simulated cores
- Run generated PyTorch operations on QEMU

Testing

- End-to-end time < 20 mins



- Operations run < 1 second



- Promising results!

Discussion

- Future Work
  - Code generation improvements
  - More target artifacts

- How to show the work is useful?

- How to integrate into the open source community?

- Contact: derek.t@rioslab.org