Java Backend for ESSENT

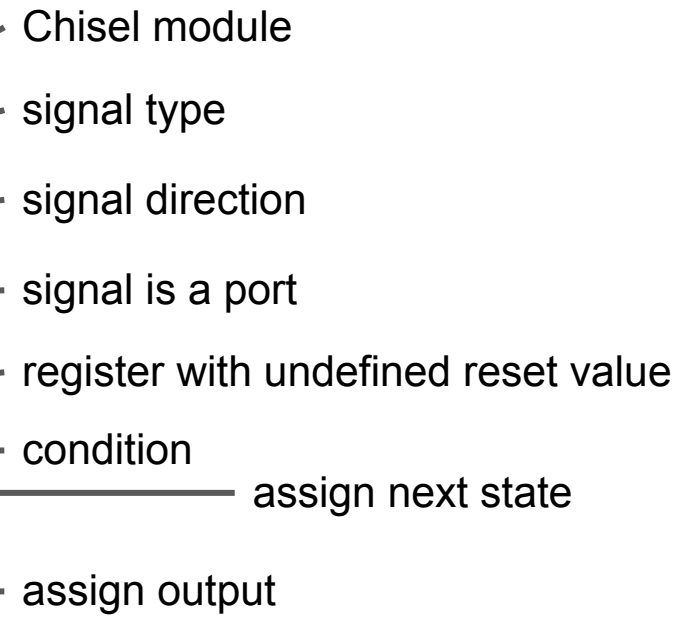
Augie Eriksson, Maanuj Vora

What is Chisel? — Overview

- Hardware Construction Language Embedded in Scala
- Allows you to write a Scala program that *generates* the description of a synchronous digital circuit
- Chisel is commonly compiled into FIRRTL (Flexible Internal Representation for RTL) or Verilog for simulation

What is Chisel? — Basics

```
class Inverter extends Module {
 val in = IO(Input(Bool()))
 val out = IO(Output(Bool()))
 val hold = IO(Input(Bool()))

 val delay = Reg(Bool())
 when(!hold) {
   delay := !in
 }
 out := delay
}
```



What is Chisel? — FIRRTL

```
circuit Inverter :
  module Inverter :
    input clock : Clock
    input reset : UInt<1>
    input in : UInt<1>
    output out : UInt<1>
    input hold : UInt<1>

    reg delay : UInt<1>, clock with :
      reset => (UInt<1>("h0"), delay)
    node _T = eq(hold, UInt<1>("h0"))
    when _T :
      node _delay_T = eq(in, UInt<1>("h0"))
      delay <= _delay_T
    out <= delay
```

Simulating Circuits

What is circuit simulation?

> Software replication of behavior of an actual electronic circuit

Simulations are:

- Cheap
- Fairly Accurate
- Can be slow

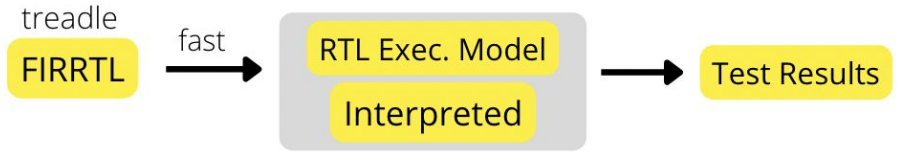
Fabrication is expensive and slow — must simulate as much as possible

Simulating Chisel Circuits

```
val dut = EssentTester(firrtlSource)

dut.poke("in", 1)
dut.poke("hold", 0)

dut.step()

assert(dut.peek("out") == 0)
```

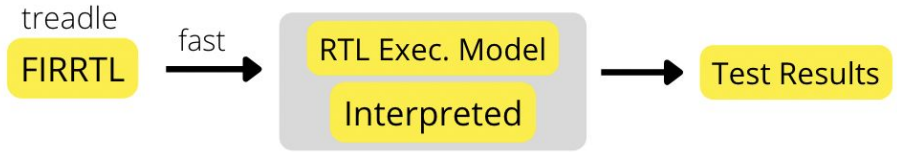
Methods of Circuit Simulation

Interpreted



Methods of Circuit Simulation

Interpreted



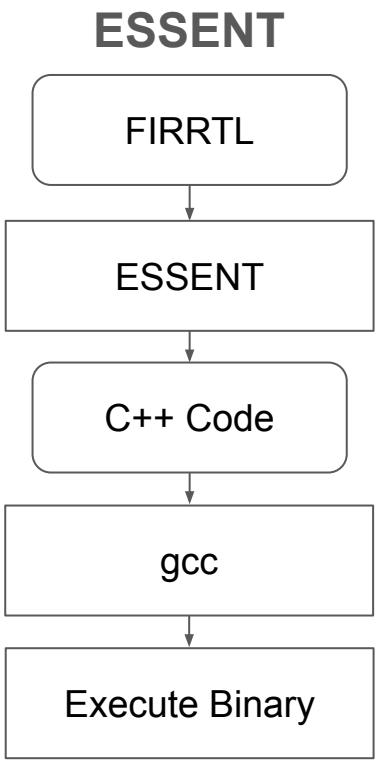
Compiled

What is ESSENT?

ESSENT is an existing compiled simulator for FIRRTL

Similar to Verilator — takes in FIRRTL instead of Verilog

ESSENT assumes that not all circuit components will change each cycle



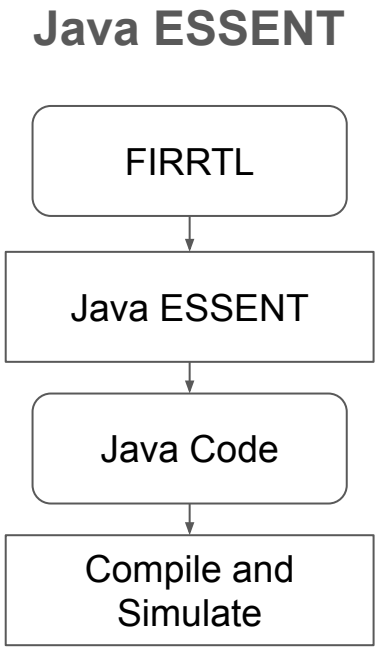
Hybrid Approach: Java Backend

Idea: Take advantage of existing Java Virtual Machine

- Testbench already runs on the JVM
- ESSENT would emit Java code instead of C++

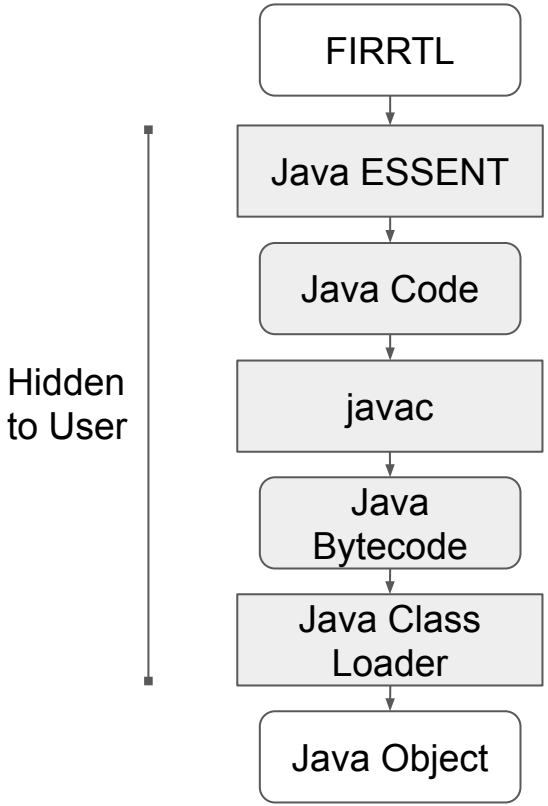
Potential Benefits:

- Faster simulation speed than Treadle
- Faster compilation speed than Verilator
- Hot spots in simulation can be just-in-time compiled



Hybrid Approach: Java Backend

```
val dut = EssentTester(source)

dut.poke("in", 1)
dut.poke("hold", 0)

dut.step()

assert(dut.peek("out") == 0)
```



Challenges Faced

Signals wider than 64 bits

- currently using immutable `BigInteger` class
- working on replacement due to memory inefficiencies

Use of Java primitives (`boolean` and `long`)

- takes better advantage of built-in javac optimizations
- less overhead than using objects (`BigInteger`)

Benchmarking Methodology

Testbenches:

- GCD
- RISCV-Mini
- TileLink Ram
- Serv

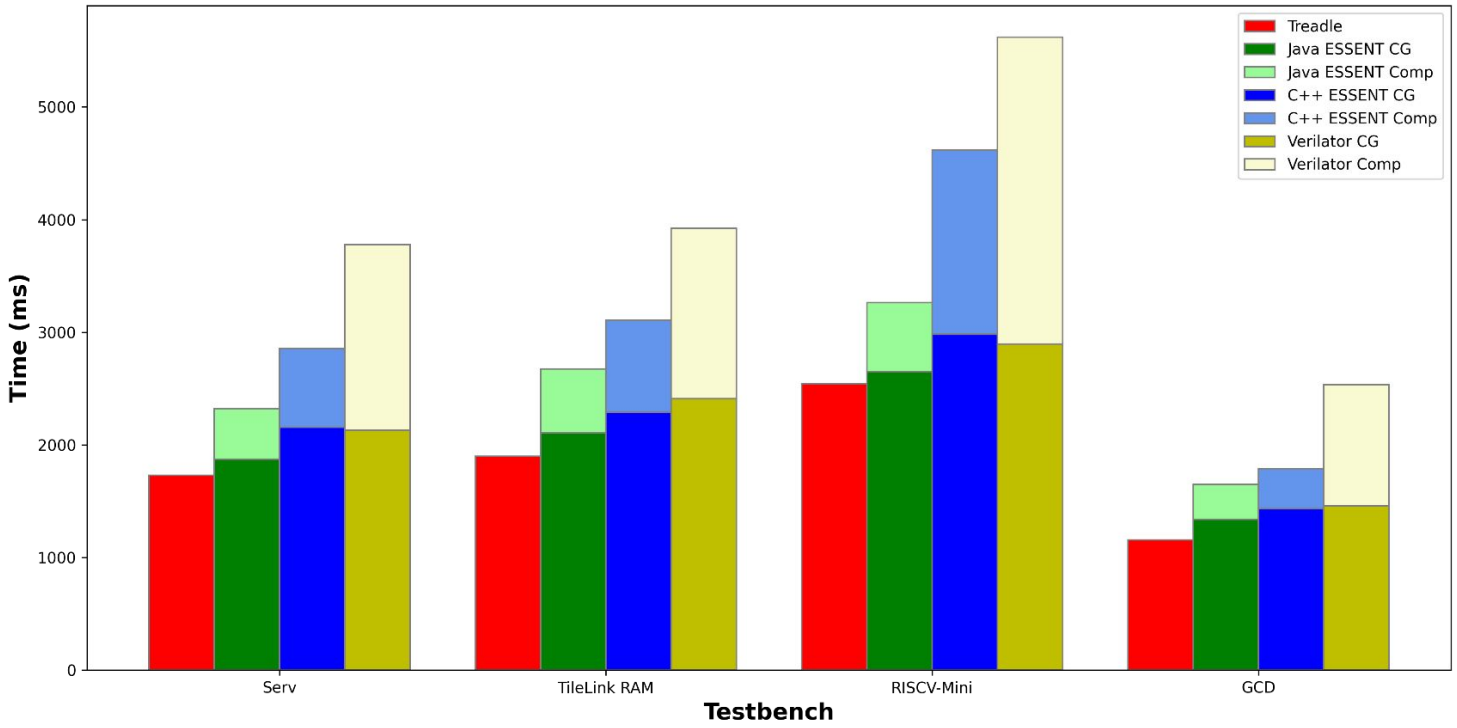
Simulators:

- Java ESSENT
- C++ ESSENT
- Verilator
- Treadle

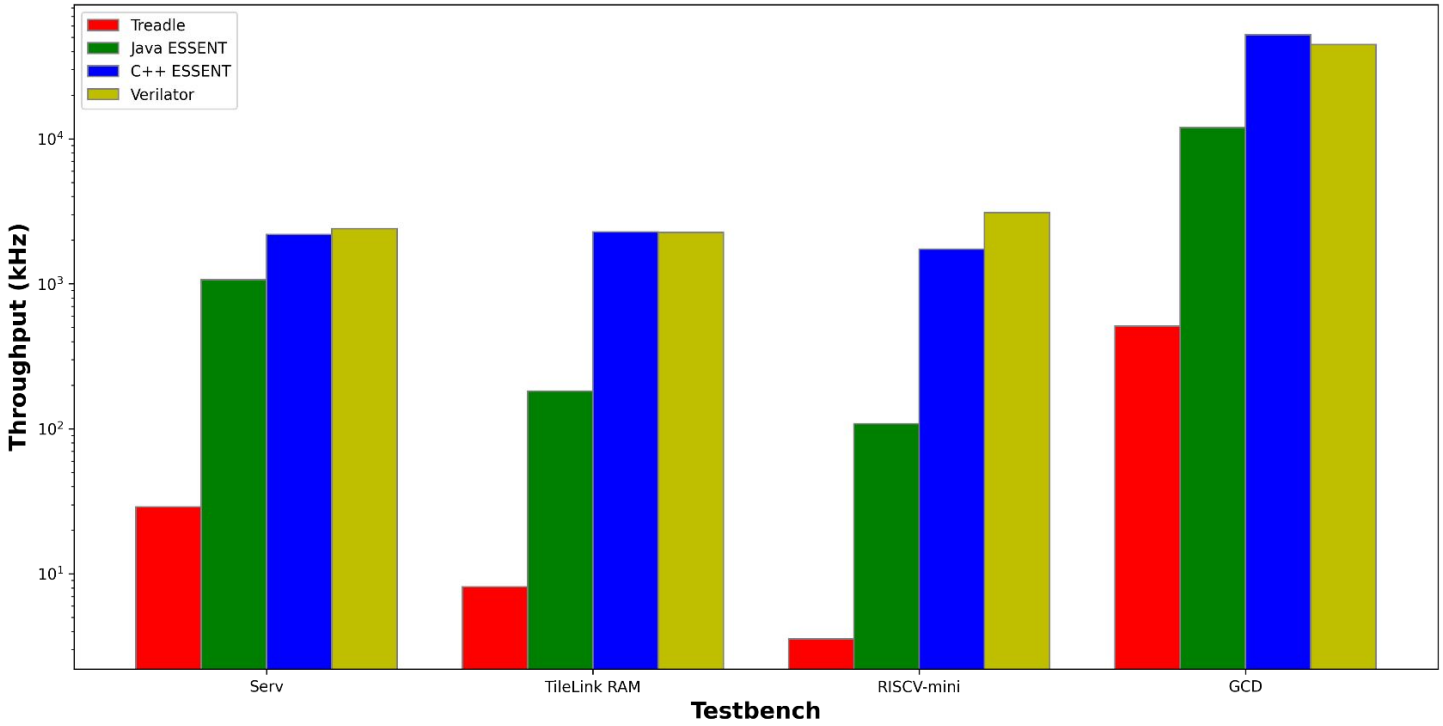
Measured:

- Code Gen (FIRRTL → C++/Java, N/A for Treadle)
- Compilation (C++/Java → Binary/Bytecode, or startup time)
- Simulation (time during actual circuit simulation)

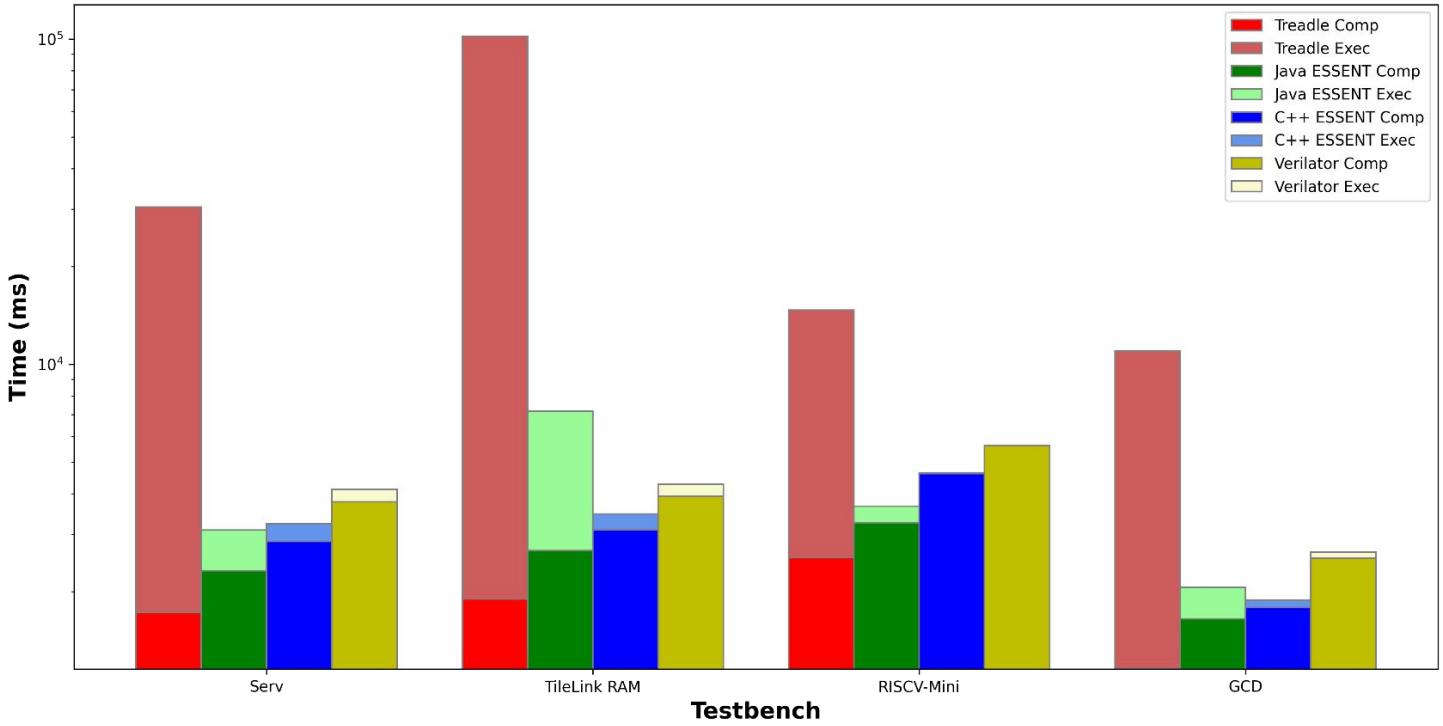
Compilation Performance Comparison



Simulation Performance Comparison



Aggregate Performance Comparison



Future Work

Better (longer) benchmarks

Faster simulation speed

VCD Dumping

`chiseltest` integration

Java ESSENT

- New open-source RTL simulator
- Outperforms interpreted simulators and has a low start-up time
- Future backend for `chiseltest` library
- Best suited for medium-sized circuits running on medium-length testbenches

GitHub:
https://github.com/ekiwi/essent/tree/java-backend

Augie Eriksson
<raeriksson@berkeley.edu>

Maanuj Vora
<maanujvora@berkeley.edu>