
# Generation of a QEMU-Based Instruction Set Simulator from a Processor Description in OpenVADL

**Johannes Zottele**, Matthias Raschhofer, Benedikt Huber and Andreas Krall

June 30, 2025

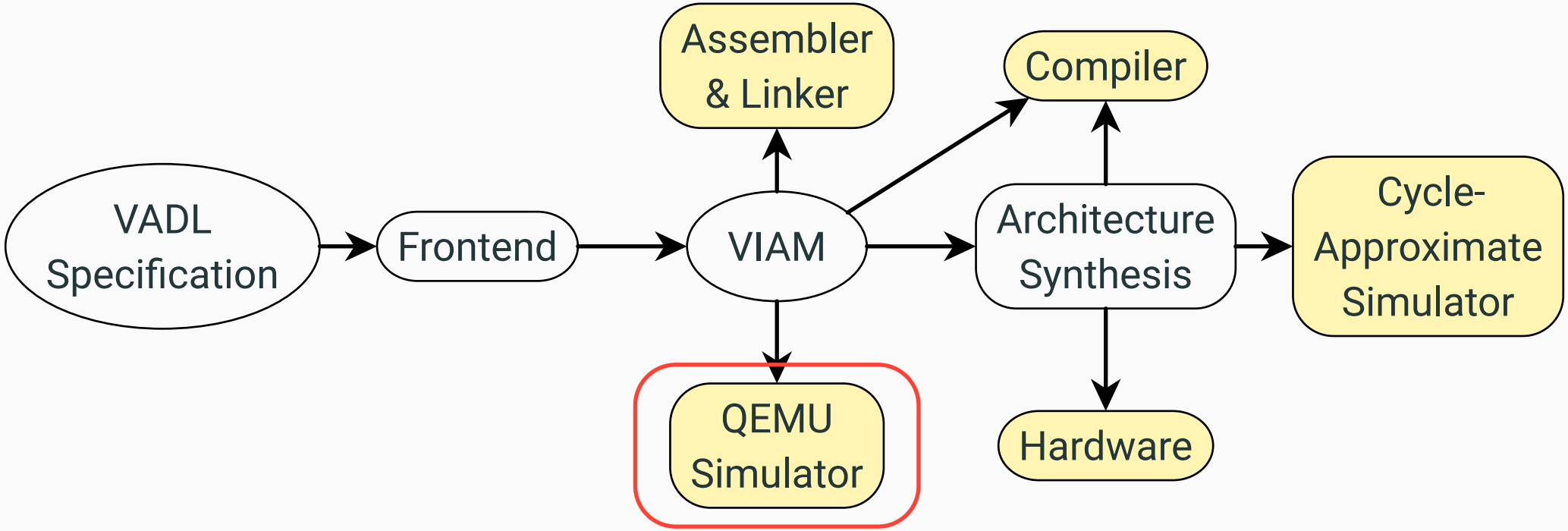
Technische Universität Wien, Vienna, Austria

# Vienna Architecture Description Language (VADL)

VADL

```
1   instruction set architecture RV64I = {
2     register   X : Bits<5> -> Bits<64>
3     format Itype : Bits<32> =
4     { imm    : Bits<12>
5     , rs1    : Bits<5>
6     , rd     : Bits<5>
7     , opcode : Bits<7>
8     , ...
9     , immS = imm as SInt<32>
10    }
11    instruction ADDI : Itype = X(rd) := X(rs1) + immS
12    encoding ADDI = {opcode = 0b001'0011, funct3 = 0b000}
```

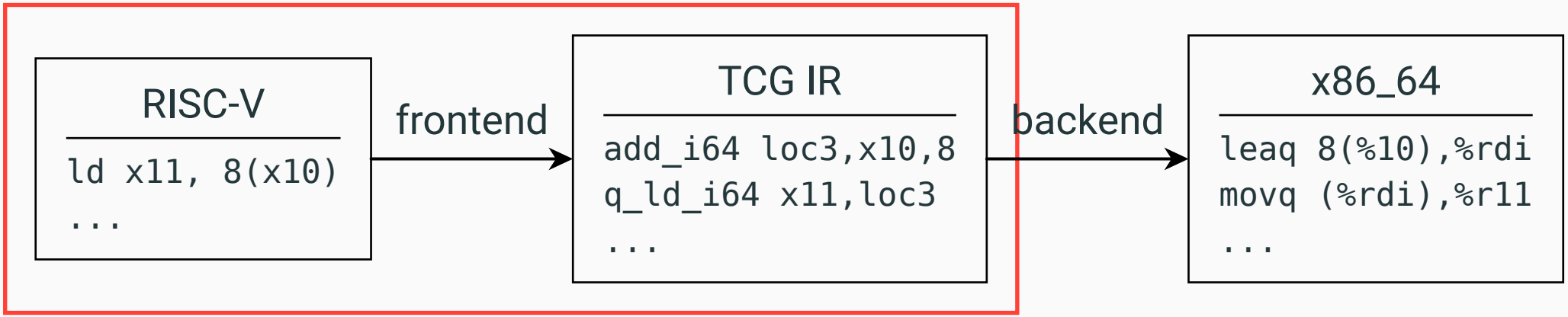
# OpenVADL Overview



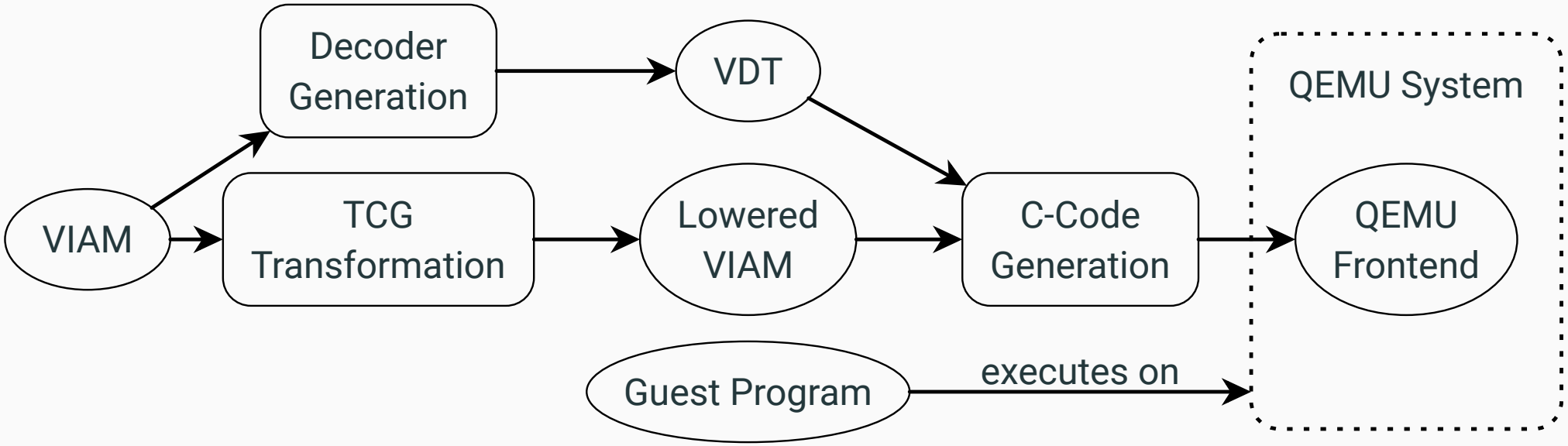
# QEMU

- Open source machine emulator
- Uses **dynamic binary translation (DBT)**
- Modular architecture
  - Simplifies support for new architectures
  - Employ an **architecture-agnostic IR (TCG)**
  - Includes reusable infrastructure (e.g. **GDB stub**)

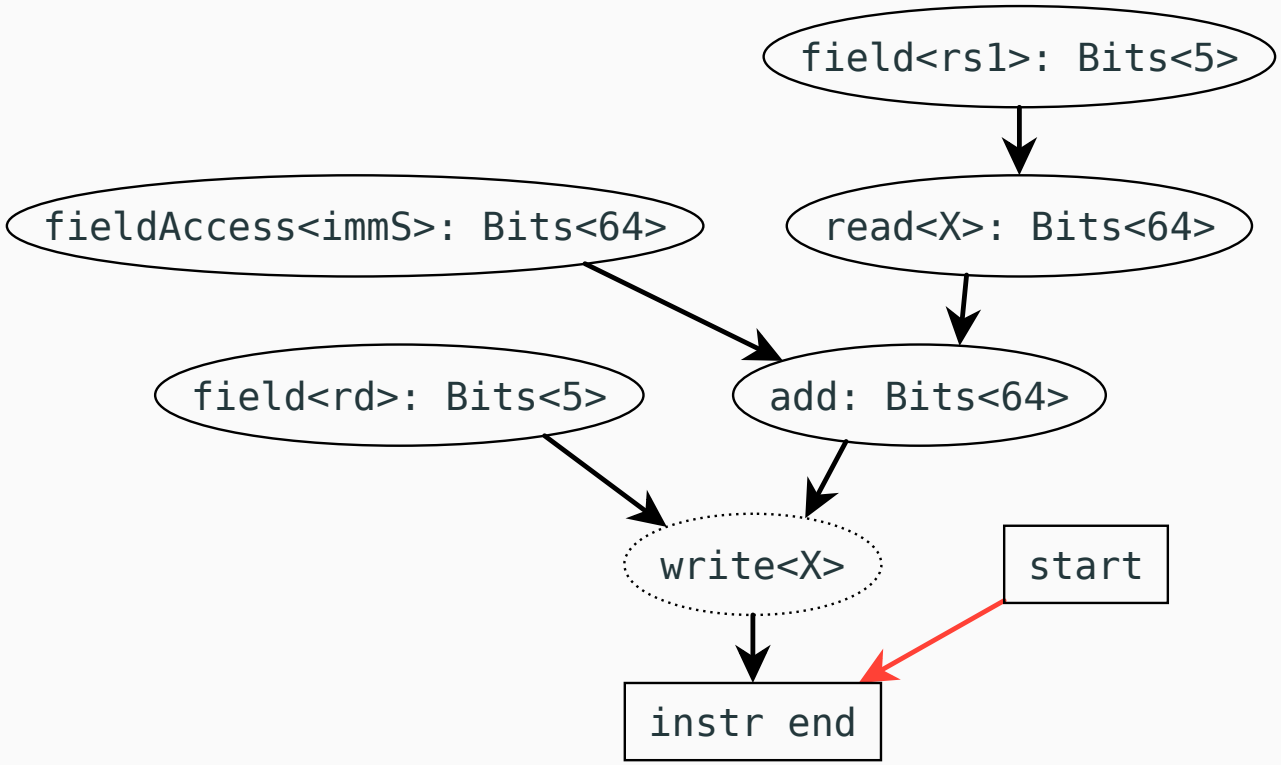
# QEMU - TCG Translation



# QEMU Generation

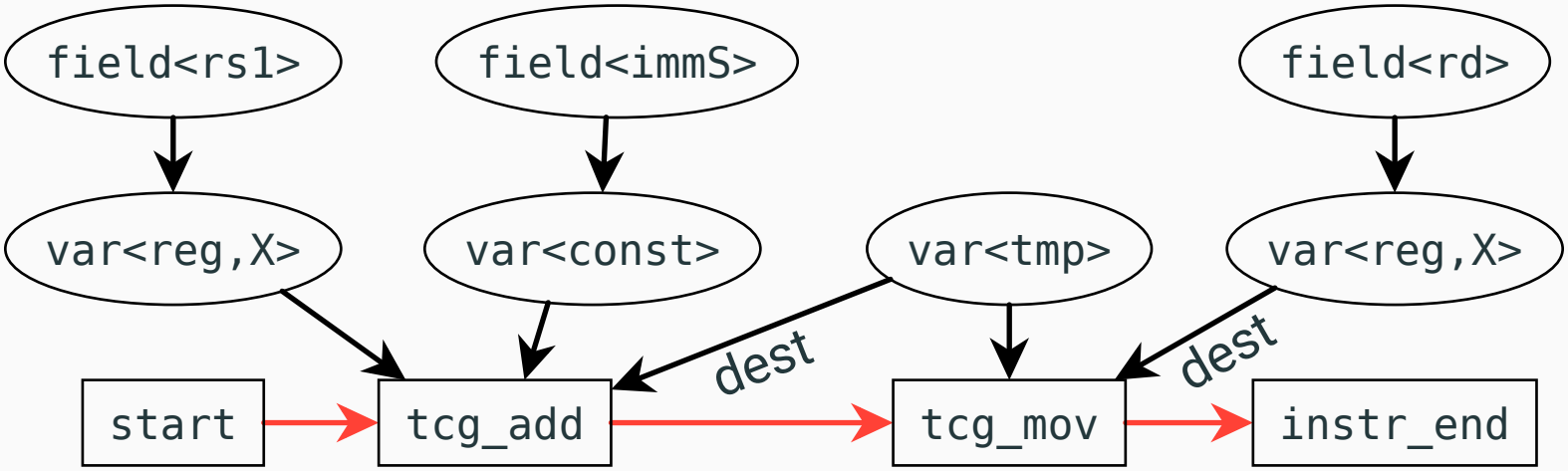


# VADL Intermediate Architecture Model (VIAM)




```
RISC-V 64 ADDI
  X(rd) := X(rs1) + immS
```

# Lowered VIAM

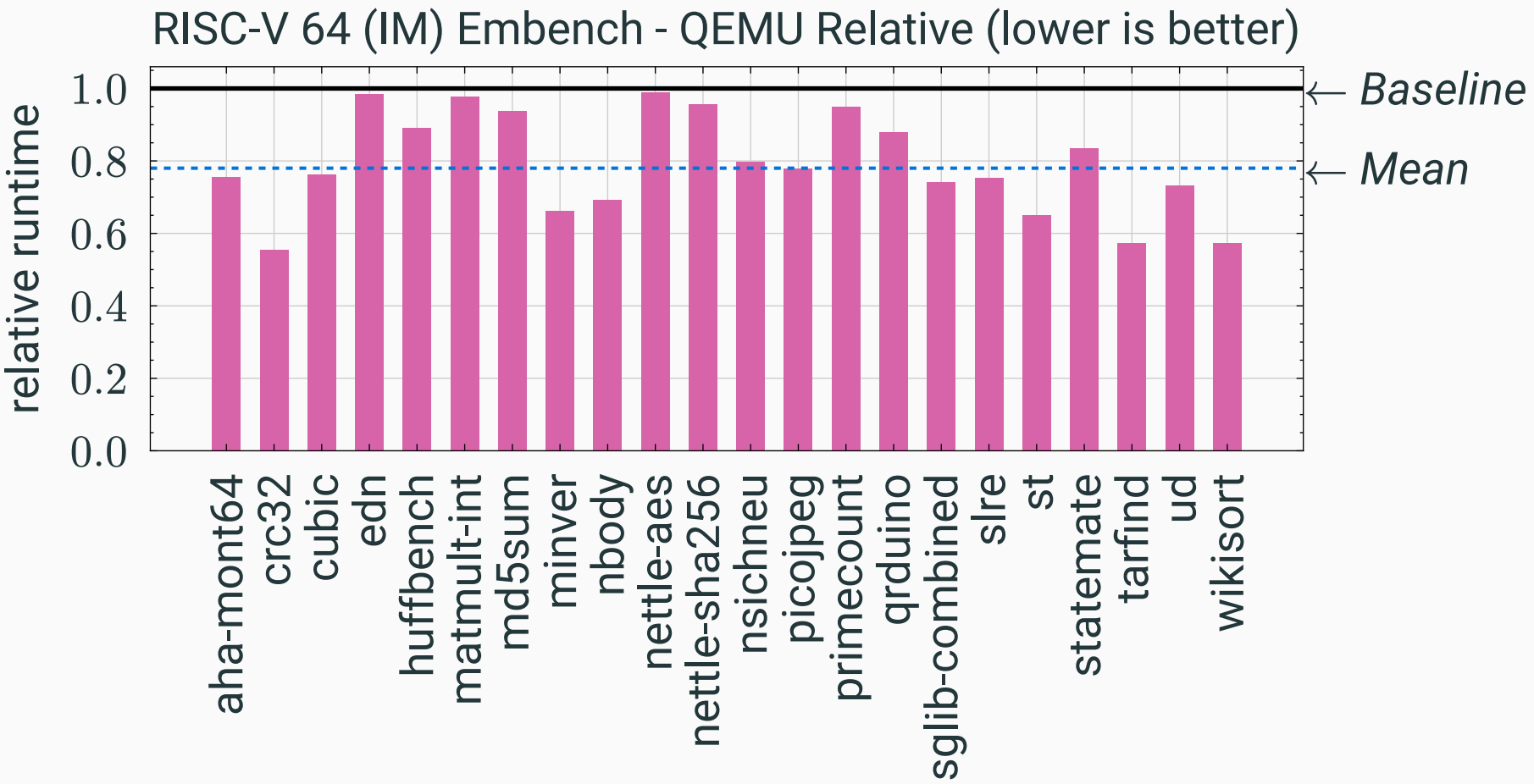


# Generated C-Code

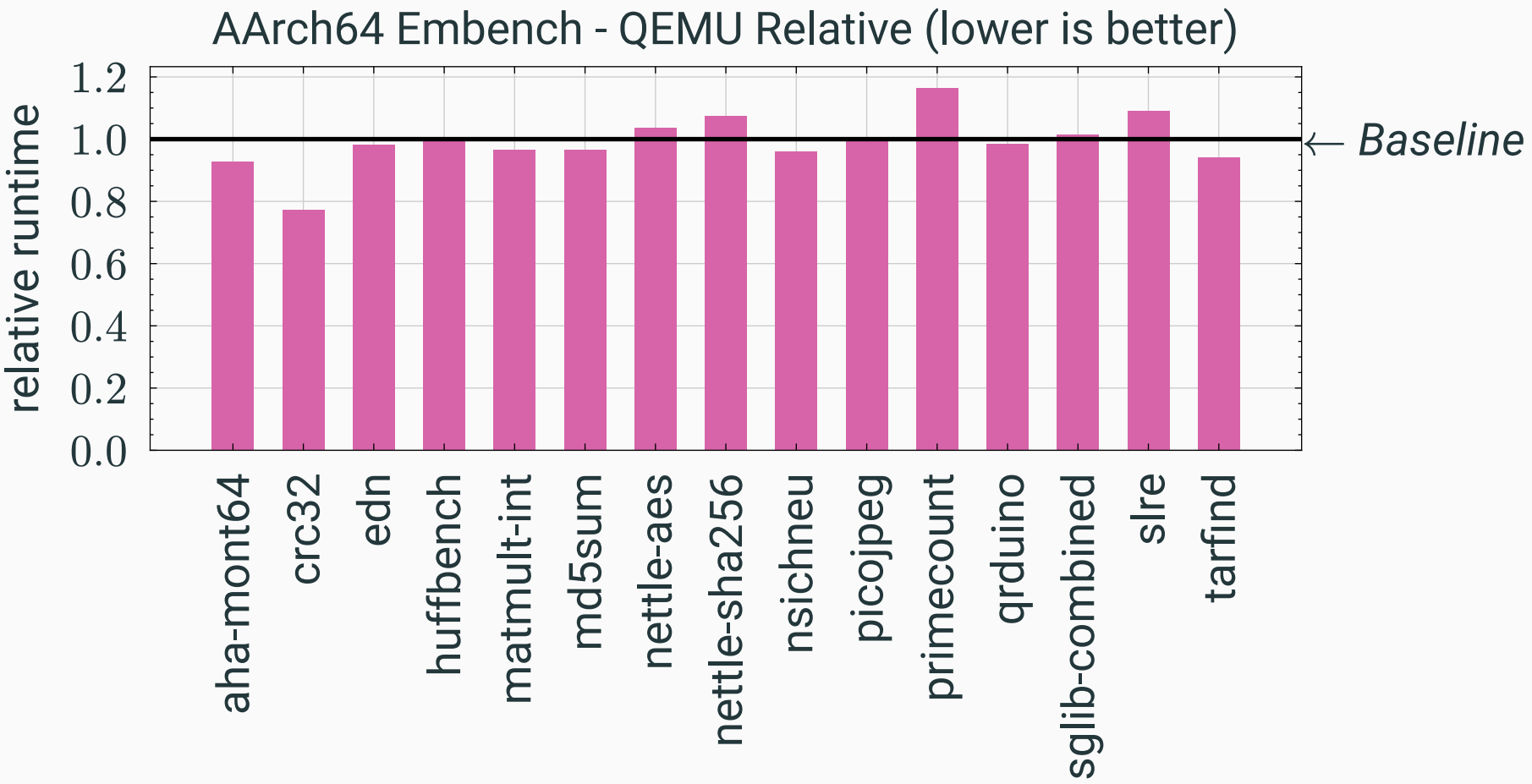
TCG Translation Function for RISC-V 64 ADDI

```c
static bool trans_addi(DisasContext *ctx, arg_addi *a) {
    TCGv_i64 reg_x_rd_dest = dest_x(ctx, a->rd);
    TCGv_i64 reg_x_rs1 = get_x(ctx, a->rs1);
    TCGv_i64 tmp_n4_0 = tcg_temp_new_i64();
    TCGv_i64 const_immS_n3 = tcg_constant_i64(a->immS);

    tcg_gen_add_i64(tmp_n4_0, reg_x_rs1, const_immS_n3);
    tcg_gen_mov_i64(reg_x_rd_dest, tmp_n4_0);

    return true;
}
```

# Evaluation Results



# Evaluation Results



# Conclusion & Future Work

- **OpenVADL** enables automatic generation of **QEMU frontends** from VADL specs
- Achieved by lowering the intermediate representation (**VIAM**) to **TCG operations**
- The generated frontend achieves up to **44% lower runtime** than upstream

---

## Future Work

- TCG vector support for tensor instructions
- User-mode simulation
- Floating-point instruction support
- Cycle Approximate Simulator based on the ISS

## More Information

- github.com/openvadl
- openvadl.org