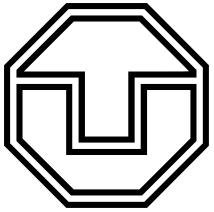
OpenVAF Verilog-A Compiler

P. Kuthe, M. Müller, M. Krattenmacher, M. Schröter

Chair for Electron Dev. & Integr. Circ., TU Dresden, 01062 Dresden, Germany
SemiMod UG (haftungsbeschränkt), Dresden, Germany

MOS-AK Feb.25 2022



Contents

- Compiler Overview
- Architectural Overview
- Capabilities
- Current Developments
- Future Developments

Compiler Overview (1)

- initially OpenVAF was developed as an interface for Verilog-A source files in the context of parameter extraction [1, 2]

  - ➢ get information on model parameters
  - ➢ evaluate model equations
  - ➢ analyze structure of model equations
  - ➢ generate derivatives of model equations

⇒These features are also needed for use in a circuit simulator context!

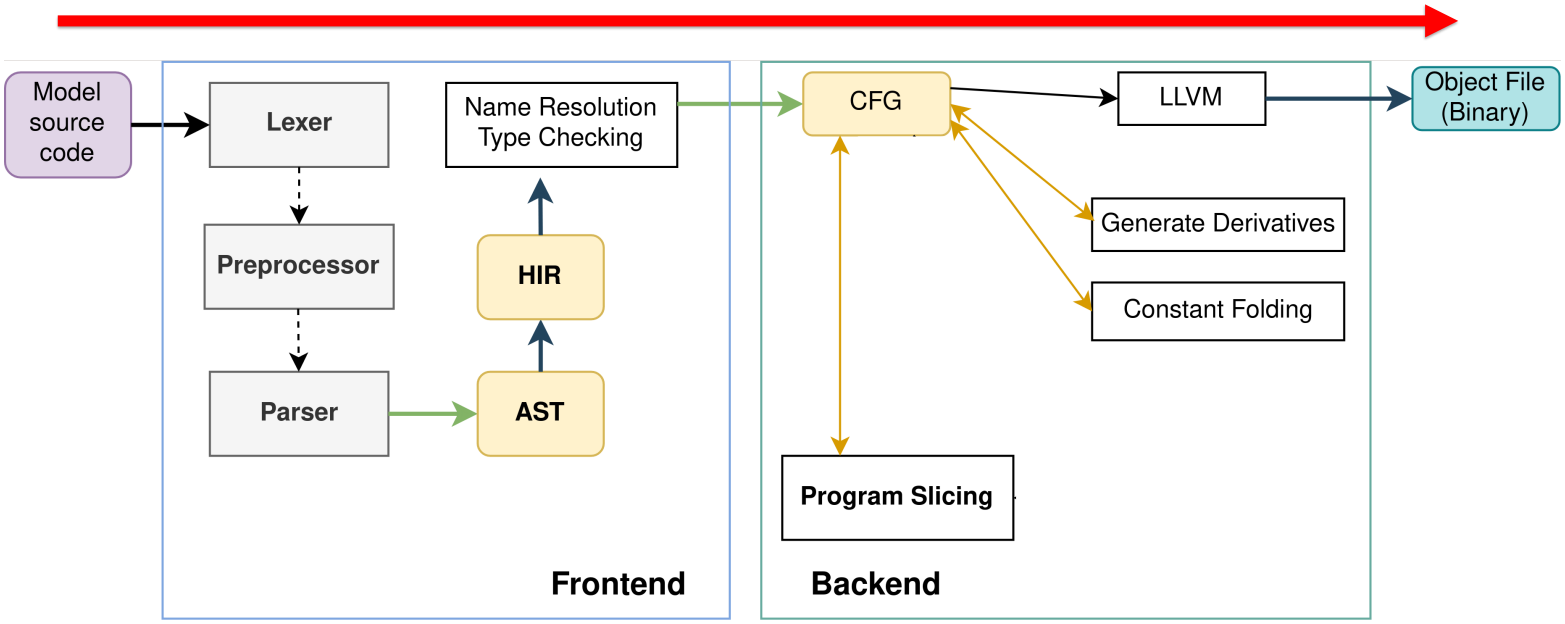
Compiler Overview (2)

- OpenVAF is a Verilog-A compiler that …
  - … directly generates executable ***machine code***
  - … offers ***fast compilation*** without the need for another compiler (gcc)
  - … implements the language standard in a ***clear and unified*** way
  - … has **great ux** (error messages)
  - … is ***open source and licensed under GPL***
- commercial partners can request commercial license, software integration services into circuit simulators and support from SemiMod

Architectural Overview

- design inspired by modern compilers (clang, rustc, swift compiler)
- full name resolution and type checking
- focus on helpful error messages
- Back-end: state of the art algorithms to support efficient code generation
- directly generate shared objects -> can load model at runtime

from Verilog-A directly to binary



Capabilities (1) –UX

```
error: unexpected token 'an identifier'; expected ';'
    ┌─ /home/dspom/Projects/OpenVAF/integration_tests/EKV/ekv.va:578:17
    │
577 │                 I(dpb) <+ MOStype*idb
    │                                     - expected ';'
578 │                 I(dpb) <+ gmin*V(dpb);
    │                 ^ unexpected token

warning[L012]: reserved keyword 'nmos' was used as an identifier
    ┌─ /home/dspom/Projects/OpenVAF/integration_tests/EKV/ekv.va:101:55
    │
101 │     (* desc="MOS, channel, type" *) parameter integer nmos=1 from [0:1];
    │                                                       ^^^^ 'nmos' is a keyword
    │
    = 'nmos' will likely never be used in the implemented language subset so this use is allowed
    = to maintain compatibility with the VAMS standard this should be renamed
    = vams_keyword_compat is set to warn by default

warning[L012]: reserved keyword 'pmos' was used as an identifier
    ┌─ /home/dspom/Projects/OpenVAF/integration_tests/EKV/ekv.va:102:55
    │
102 │     (* desc="MOS, channel, type" *) parameter integer pmos=1 from [0:1];
    │                                                       ^^^^ 'pmos' is a keyword
    │
    = 'pmos' will likely never be used in the implemented language subset so this use is allowed
    = to maintain compatibility with the VAMS standard this should be renamed
    = vams_keyword_compat is set to warn by default
```

Capabilities (2) –UX

```
error: type missmatch: expected real value but found string parameter ref
    ┌─ /home/dspom/Projects/OpenVAF/integration_tests/EKV/ekv.va:317:47
    │
317 │             ucrit_a = ucrit_p*pow(TempK/TnomK,ucex);
    │                                               ^^^^ expected real value

error: 'TnomK_' was not found in the current scope
    ┌─ /home/dspom/Projects/OpenVAF/integration_tests/EKV/ekv.va:318:79
    │
318 │             phi_a   = phi_p*TempK/TnomK-3*Vt*ln(TempK/TnomK)-`EG(TnomK)*TempK/TnomK_+`EG(TempK);
    │                                                                             ^^^^^^ not found

error: contribute to branch with input ports
    ┌─ /home/dspom/Projects/OpenVAF/integration_tests/EKV/ekv.va:612:13
    │
 69 │     input           s;
    │     ----------------- info: 's' was declared input here
    .
 80 │     branch (s,sp)  ssp;
    │     ------------------ info: 'ssp' was declared here
    .
612 │             I(ssp) <+ V(ssp)/RSeff;
    │             ^^^^^^ illegal contribution
    │
    = help: change direction of 's' to inout

error: could not compile `ekv.va` due to 4 previous errors; 2 warning emitted
```

Capabilities (3) –Derivatives

- **arbitrary order** derivatives are supported
- efficient code generation:
  - utilize SSA -> re-use expressions
  - exploit mathematical identities
- very fast compile and run-time time
- can generate minimum number of Jacobian entries

$a = \exp(x)\sin(x)$
$b = ddx(ddx(a))$

```
block0:
    v12 = sin v10
    v13 = exp v10
    v16= fmul v12, v13
    v14 = call fn0 (v16)
    v15 = call fn0 (v14)
```

auto diff

```
block0:
    v12 = sin v10
    v101 = cos v10
    v102 = sin v10
    v103 = fneg v102
    v13 = exp v10
    v16 = fmul v12, v13
    v104 = fmul v101, v13
    v105 = fadd v104, v16
    v106 = fmul v103, v13
    v107 = fadd v106, v104
    v108 = fadd v107, v105
```

Capabilities (4) – Compile Time

*.va → compilation of all branches + derivatives → model.o

```
test tests::integration::ekv ......... ok; finished in 0.03s
test tests::integration::bsim4 ....... ok; finished in 0.24s
test tests::integration::hicuml2 ..... ok; finished in 0.23s
test tests::integration::diode_cmc ... ok; finished in 0.27s
test tests::integration::mvsg_cmc .... ok; finished in 0.55s
test tests::integration::asmhemt ..... ok; finished in 0.38s
test tests::integration::bsim3 ....... ok; finished in 0.41s
test tests::integration::bsimimg ..... ok; finished in 0.43s
test tests::integration::bsimcmg ..... ok; finished in 0.46s
test tests::integration::bsimsoi ..... ok; finished in 0.56s
test tests::integration::bsim6 ....... ok; finished in 0.63s
test tests::integration::bsimbulk .... ok; finished in 0.77s
test tests::integration::hisimsotb ... ok; finished in 0.71s
test tests::integration::psp ......... ok; finished in 1.02s
test tests::integration::hisimhv ..... ok; finished in 3.58s
test tests::integration::hisim2 ...... ok; finished in 15.75s

test result: ok. 16 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 25.32s
```

this a bug

Capabilities (5) – Standard Compliance

- handles all public CMC models **without modifications**
- large parts of the standard are already covered in front-end
- missing:
  - vector nets
  - arrays
  - genvar
  - complex events
  - nested modules

```
test tests::integration::ekv ......... ok
test tests::integration::bsim4 ....... ok
test tests::integration::hicuml2 ..... ok
test tests::integration::diode_cmc .... ok
test tests::integration::mvsg_cmc ..... ok
test tests::integration::asmhemt ..... ok
test tests::integration::bsim3 ....... ok
test tests::integration::bsimimg ..... ok
test tests::integration::bsimcmg ..... ok
test tests::integration::bsimsoi ..... ok
test tests::integration::bsim6 ....... ok
test tests::integration::bsimbulk ..... ok
test tests::integration::hisimsotb .... ok
test tests::integration::psp ......... ok
test tests::integration::hisimhv ..... ok
test tests::integration::hisim2 ....... ok
```

- problem: some uncertainties in the language

Current Developments (1) – Compiler

- separate resistive and reactance related (idt/ddt) Jacobians
- polish some details (bugfixes etc.)
- generate code for noise simulations

Current Developments (2) – Interface

- goal: simulator independent interface for **compact models**
- seperate charge/current rhs and matrix entries
  - ➢ easily support tran, AC, harmonic balance etc.
- seperate function for noise
- requires some restrictions
  - restrict analog operators (ac_stim, laplace,...)
  - ddt/idt must be linear

Current Developments (3) – Integration

- integration into ngspice
  - goal: compile and simulate first compact model with Ngspice in 2022
  - we are looking for funding
- integration into commercial simulators
  - talks with EDA vendors

Future Developments

- behavioural models / full circuits in Verilog-A
- funding will be required long-term
  - ➢ cooperation with EDA vendors is high priority
  - ➢ funding for integration with open-source simulators?
- Work on Verilog-A standard compliance
  - ➢ the language standard can be ambiguous
  - ➢ exchange about interpretation of the standard
  - ➢ cooperation on a shared Verilog-A test-suite would be of high priority

Acknowledgments

Tommy Rosenbaum and Martin Claus (Infineon)

Wladek Grabinski (MOS-AK)

Dietmar Warning and Holger Vogt (NGspice)

References

- [1] M. Krattenmacher, M. Müller and M. Schröter, “DMT—Device-modeling-toolkit,” in Proc. HICUM Workshop, Munich, Germany, 2019. [Online]. Available: https://www.iee.et.tu-dresden.de/iee/eb/forsch/Models/workshop_2019/contr_2019/dmt.pdf
- [2] P. Kuthe, M. Müller and M. Schröter, "VerilogAE: An open source Verilog-A compiler for compact model parameter extraction“, in IEEE Journal of the Electron Devices Society, vol. 8, pp. 1416-1423, 2020.