ANALOG DEVICES

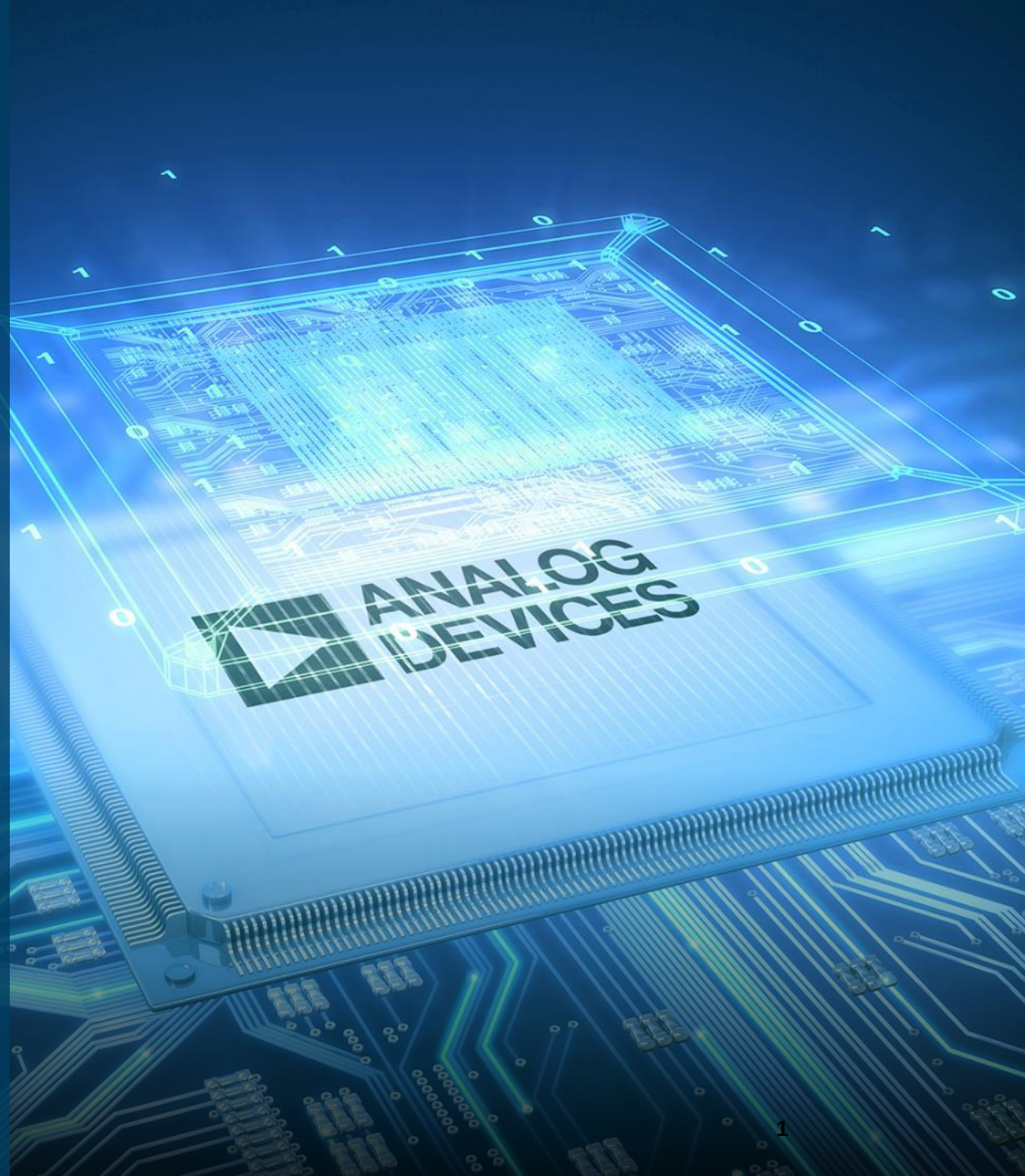
AHEAD OF WHAT'S POSSIBLE™

VAMPyRE
Verilog-A Model
Pythonic Rule Enforcer

Geoffrey Coram

Analog Devices, Inc.

Si2 CMC Technical Advisor for Verilog-A



13th International MOS-AK Workshop, Silicon Valley, Dec. 10-11, 2020

Motivation

- Interest from Compact Model Coalition (CMC) model developers for a basic checker to help in code development
- Interest from EDA vendors in consistent coding style for optimization
  - eg, consistent syntax for switch branches/node collapse
- Interest from CMC members in verifying policy compliance

- Existing tools have various drawbacks

Coram: VAMPyRE, MOS-AK 4Q2020

Available Tools

- VALint developed as part of NEEDS (http://needs.nanohub.org)
  - Presented at MOS-AK Washington, DC in 2015 (Xufeng Wang, Purdue)
  - Development apparently abandoned after NEEDS project was not renewed
- ADMS (https://sourceforge.net/projects/mot-adms/)
  - Open-source Verilog-A tool
  - Not user-friendly, requires additional packages, last update 3 years ago
- PyVerilog (https://pypi.org/project/pyverilog/)
  - Digital-central (IEEE Std 1364, not VAMS)
  - Requires Icarus Verilog

Available Tools

- Commercial simulators
  - Most major simulators compile and run Verilog-A models on the fly
  - Generally don’t warn users about problems
  - Require a license
- Cadence AHDL LINT
  - Presented at MOS-AK Silicon Valley in 2017 (Jushan Xi et al.)
  - Performance tips mainly

Coram: VAMPyRE, MOS-AK 4Q2020

VAMPyRE: What is it?

- Verilog-A Model Pythonic Rule Enforcer

- Stand-alone parser and checker
  - Single file vampyre.py (~5700 lines)
  - Written in python (2.7.x or 3.x.y), uses os, sys, argparse modules
  - No compilation or installation of other packages
- Written by Geoffrey Coram
  - Si2/CMC Technical Advisor for Verilog-A since 2016
  - Chairman of Verilog-AMS subcommittee that added compact modeling extensions in 2004

VAMPyRE: What does it do?

- Checks implemented:
  - Hidden state (variables used before assigned, or assigned conditionally)
  - Bias-dependent switch branches
  - Integer division (1/2 = 0 in Verilog-A)
  - Division by zero for parameters (1/parm if parm's range allows 0)
  - Unused parameters or variables
  - Incorrect access function for discipline (V(dt) if dt is thermal)
  - Ports without direction and/or discipline
  - Unnamed noise sources
  - Use of various features not appropriate for compact models (@events, log() instead of ln(), z-transforms, etc.)

VAMPyRE: What does it do?

- Style checking:
  - Proper indentation of blocks
  - No spaces at ends of lines
  - No TAB characters

- Style fixing:
  - Can write out files with corrected indentation

Digression: Hidden State

- What is hidden state?
  - Variable used before it is set
  - Simulator must remember value from iteration to iteration
- How does it happen?
  - Sometimes intentional: counter

```
if (V(in) > vth) count = count + 1;
```

  - Sometimes unintentional:
    - Assigned "`if`" but not "`else`"
    - Assigned under conditionA, but used under conditionB
      - Simulator may not realize that B is a subset of A

Digression: Hidden State

- `asrh` assigned in else block of one condition, used in else block of a second condition
- Tricky to correlate these in all cases; safer to consider `asrh` hidden state

```
if ((CSRH == 0.0) && (CTAT == 0.0)) begin \
    isrh          =  0.0; \
end else begin \
    // assign asrh


if (CTAT == 0.0) begin \
    itat          =  0.0; \
end else begin \
    // use asrh
```

Digression: Hidden State

- `pterm` assigned under conditions `flsh == 1` or `2`, used in else of `flsh == 0 || rth < `MIN_R`
- Hard to correlate these along with range of `flsh` that prevents `flsh > 2`

```
if (flsh == 1 && rth >= `MIN_R) begin
    pterm   =  Vciei*it + (vdci_t-Vbici)*iavl;
end else if (flsh == 2 && rth >= `MIN_R) begin
    pterm   =  Vciei*it + (vdci_t-Vbici)*iavl + ibei*Vbiei
             + ibci*Vbici + ibep*Vbpei + ijbcx*Vbpci + ijsc*Vsici;
    …
end
if(flsh == 0 || rth < `MIN_R) begin
    V(br_sht)       <+ 0.0;
end else begin
    I(br_sht)       <+ V(br_sht)/rth_t-pterm;
    I(br_sht)       <+ ddt(cth*V(br_sht));
end
```

Digression: Hidden State

- How can I avoid hidden state?
  - Initialize variables properly
  - Ensure variable is set in all `if/else` blocks (or initialize right before the `if` statements)
  - Ensure `case` statements have `default:`

Digression: Bias-dependent switch branches

- What is a switch branch?

```
if (V(in) < vth) begin
  I(sw) <+ 0; // open switch (no current)
end else begin
  V(sw) <+ 0; // closed switch
end
```

- Described in the Verilog-AMS LRM (version 2.4, section 5.6.5)

Digression: Bias-dependent switch branches

- Switch branch may also have finite resistances:

```
if (V(in) < vth) begin
  I(sw) <+ V(in) / Roff;
end else begin
  V(sw) <+ I(in) * Ron;
end
```

- Ron may be zero

Digression: Bias-dependent switch branches

- How is it relevant to compact models?
  - Parasitic resistors implemented with switch branch

```
if (Rdrain > 0) begin
  I(d,di) <+ V(d,di) / Rdrain;
end else begin
  V(d,di) <+ 0;
end
```

  - Optimizing compilers may collapse (d,di) to reduce matrix size

Digression: Bias-dependent switch branches

- What can go wrong?
  - If condition is bias-dependent, simulator must be able to dynamically switch formulations during the simulation
  - Node “di” is retained, and extra matrix row for branch current

```
if (Rd_t > 0) begin
  I(d,di) <+ V(d,di) / Rd_t;
end else begin
  V(d,di) <+ 0;
end
```


Coram: VAMPyRE, MOS-AK 4Q2020

Digression: Bias-dependent switch branches

- How do you fix it?
  - If condition must be bias-independent (RdGeo depends only on parameters)
  - But resistance Rdrain may depend on bias (or self-heating temperature)
  - Ensure that Rdrain is always non-zero when RdGeo is

```
if (RdGeo > 0) begin
  I(d,di) <+ V(d,di) / Rdrain;
end else begin
  V(d,di) <+ 0;
end
```

VAMPyRE: Does it work?

- Tested against CMC standard models
  - BSIMCMG
  - BSIMIMG
  - Hicum
  - MVSG_CMC
  - ASM-HEMT
  - PSP
  - (and others)

VAMPyRE: BSIMCMG 111.1.0-beta3

Reading bsimcmg.va
Reading bsimcmg_macros.include
Reading bsimcmg_parameters.include
NOTICE in file …, line 1306: Parameter 'LCGISL' with units 'm*V^3' has default 'LCGIDL' with units 'm*(V^3)'
NOTICE in file …, line 1308: Parameter 'PCGISL' with units 'm*V^3' has default 'PCGIDL' with units 'm*(V^3)'
NOTICE in file …, line 1309: Parameter 'WCGISL' with units 'm*V^3' has default 'PCGIDL' with units 'm*(V^3)'
NOTICE in file …, line 1310: Parameter 'P2CGISL' with units 'm*V^3' has default 'PCGIDL' with units 'm*(V^3)'
NOTICE in file …, line 1624: Parameter 'NOIA2' with no units has default 'NOIA' with units 's^(1-EF)/(eV*m^3)'
NOTICE in file …, line 1798: Parameter 'WSSP0' with no units has default 'WDIM0' with units 'm'
NOTICE in file …, line 1799: Parameter 'WSSPR' with no units has default 'WDIMR' with units 'nm'
Reading bsimcmg_variables.include
Reading bsimcmg_body.include
Reading bsimcmg_initialization.include
WARNING in file …, line 182: Unexpected default 'PDVT1' for binning parameter 'WDVT1SS'
WARNING in file …, line 182: Unexpected default 'PDVT1' for binning parameter 'P2DVT1SS'
WARNING in file …, line 193: Unexpected default 'PK0SI' for binning parameter 'WK2SI'
WARNING in file …, line 193: Unexpected default 'PK0SI' for binning parameter 'P2K2SI'

VAMPyRE: BSIMCMG 111.1.0-beta3

bsimcmg_parameters.include:

[LNPWP2]DVT1SS take defaults from [LNPPP]DVT1

`MPRnb(DVT1SS, DVT1, "", "Subthreshold swing exponent coefficient. After binning it should be within (0 : inf)")

`MPRnb(LDVT1SS, LDVT1, "m", "L-term of DVT1SS")

`MPRnb(NDVT1SS, NDVT1, "", "N-term of DVT1SS")

`MPRnb(PDVT1SS, PDVT1, "m", "P-term of DVT1SS")

`MPRnb(WDVT1SS, PDVT1, "m", "W-term of DVT1SS")

`MPRnb(P2DVT1SS, PDVT1, "m", "WL-term of DVT1SS")

VAMPyRE: BSIMCMG 111.1.0-beta3

bsimcmg_parameters.include:

[LNPWP2]DVT1SS take defaults from [LNPPP]DVT1SS

`MPRnb(DVT1SS, DVT1, "", "Subthreshold swing exponent coefficient. After binning it should be within (0 : inf)")

`MPRnb(LDVT1SS, LDVT1, "m", "L-term of DVT1SS")

`MPRnb(NDVT1SS, NDVT1, "", "N-term of DVT1SS")

`MPRnb(PDVT1SS, PDVT1, "m", "P-term of DVT1SS")

`MPRnb(WDVT1SS, PDVT1, "m", "W-term of DVT1SS")

`MPRnb(P2DVT1SS, PDVT1, "m", "WL-term of DVT1SS")

VAMPyRE: BSIMIMG 102.9.3 beta

Reading bsimimg.va
NOTICE in file bsimimg/bsimimg_body.include, line 720: $error() preferred over $finish()
NOTICE in file bsimimg/bsimimg_body.include, line 729: $error() preferred over $finish()
NOTICE in file bsimimg/bsimimg_body.include, line 741: $error() preferred over $finish()
NOTICE in file bsimimg/bsimimg_body.include, line 750: $error() preferred over $finish()
NOTICE in file bsimimg/bsimimg_body.include, line 1012: $error() preferred over $finish()
    Further notices of this type will be suppressed

WARNING in file bsimimg/bsimimg_body.include, line 1537: Bias-dependent '==': if (spD==0.0)
WARNING in file bsimimg/bsimimg_body.include, line 1582: Bias-dependent '==': if (Rdss==0.0)
WARNING in file bsimimg/bsimimg_body.include, line 1606: Bias-dependent '==': if (spD==0.0)
WARNING in file bsimimg/bsimimg_body.include, line 1713: Bias-dependent '!=': if (K0_t!=0.0)

WARNING in file bsimimg/bsimimg_body.include, line 2183: Switch branch (d,di) with bias-dependent condition
WARNING in file bsimimg/bsimimg_body.include, line 2190: Switch branch (s,si) with bias-dependent condition
WARNING in file bsimimg/bsimimg_body.include, line 2194: Switch branch (d,di) with bias-dependent condition
WARNING in file bsimimg/bsimimg_body.include, line 2195: Switch branch (s,si) with bias-dependent condition

VAMPyRE: BSIMIMG 102.9.3 beta

WARNING in file bsimimg_body.include, line 1713: Bias-dependent '!=': if (K0_t!=0.0)

```
// Lateral Non-uniform doping effect (IV-CV Vth shift) factor
if (K0_t != 0.0) begin
  T1   = K0_t / (K0SI_t * qia + 2.0 * nVtm);
  Mnud = lexp(-T1);
end else begin
  Mnud = 1.0;
end
```

VAMPyRE: BSIMIMG 102.9.3 beta

WARNING in file bsimimg_body.include, line 1713: Bias-dependent '!=': if (K0_t!=0.0)

```
// Lateral Non-uniform doping effect (IV-CV Vth shift) factor
if (K0_t != 0.0) begin
  T1   = K0_t / (K0SI_t * qia + 2.0 * nVtm);
  Mnud = lexp(-T1);
end else begin
  Mnud = 1.0;
end
```

lexp(0) = 1 so the value of Mnud matches

VAMPyRE: BSIMIMG 102.9.3 beta

WARNING in file bsimimg_body.include, line 1713: Bias-dependent '!=': if (K0_t!=0.0)

```
// Lateral Non-uniform doping effect (IV-CV Vth shift) factor
if (K0_t != 0.0) begin
  T1   = K0_t / (K0SI_t * qia + 2.0 * nVtm);
  Mnud = lexp(-T1);
end else begin
  Mnud = 1.0;
end
```

lexp(0) = 1 so the value of Mnud matches

BUT the derivatives of the constant 1.0 are all zero!

VAMPyRE: HICUM/L0

```
Reading hicumL0V2p0p0.va
WARNING in file hicumL0V2p0p0.va, line 750: Bias-dependent '==': if (cjcx0_t==0.0)
WARNING in file hicumL0V2p0p0.va, line 1067: $simparam("gmin", 1e-12) should use 0 for default value
WARNING in file hicumL0V2p0p0.va, line 1068: $simparam("gmin", 1e-12) should use 0 for default value
ERROR in file hicumL0V2p0p0.va, line 1073: Branch contribution depends on quantity with bad derivative (from bias-dependent == or !=)
ERROR in file hicumL0V2p0p0.va, line 1074: Branch contribution depends on quantity with bad derivative (from bias-dependent == or !=)
ERROR in file hicumL0V2p0p0.va, line 1088: Branch contribution depends on quantity with bad derivative (from bias-dependent == or !=)
ERROR in file hicumL0V2p0p0.va, line 1092: Branch contribution depends on quantity with bad derivative (from bias-dependent == or !=)
ERROR in file hicumL0V2p0p0.va, line 1093: Branch contribution depends on quantity with bad derivative (from bias-dependent == or !=)
   Further errors of this type will be suppressed
WARNING in file hicumL0V2p0p0.va, line 1177: Bias-dependent '!=': if (IB!=0.0)
WARNING in file hicumL0V2p0p0.va, line 1183: $simparam("gmin", 1e-12) should use 0 for default value
WARNING in file hicumL0V2p0p0.va, line 1185: $simparam("gmin", 1e-12) should use 0 for default value
WARNING in file hicumL0V2p0p0.va, line 1188: $simparam("gmin", 1e-12) should use 0 for default value
   Further warnings of this type will be suppressed
WARNING in file hicumL0V2p0p0.va, line 519: Parameter 'f2vg' was never used
WARNING in file hicumL0V2p0p0.va, line 1234: Module uses $temperature but does not have 'dtemp'
WARNING in file hicumL0V2p0p0.va, line 1234: Variable 'Cdummy' was never used
WARNING in file hicumL0V2p0p0.va, line 1234: Variable 'Qdummy' was never used
WARNING in file hicumL0V2p0p0.va, line 1234: Operating-point variable 'RCX' differs only in case from parameter 'rcx'
WARNING in file hicumL0V2p0p0.va, line 1234: Operating-point variable 'RE' differs only in case from parameter 're'
WARNING in file hicumL0V2p0p0.va, line 1234: Operating-point variable 'IS' differs only in case from parameter 'is'
```

VAMPyRE: MVSG_CMC 2.0.0

```
Reading mvsg_cmc_2.0.0.va
WARNING: Non-standard encoding in file mvsg_cmc_2.0.0.va (windows-1252)
STYLE in file mvsg_cmc_2.0.0.va, line 11: Space character at end of line
STYLE in file mvsg_cmc_2.0.0.va, line 12: Space character at end of line
STYLE in file mvsg_cmc_2.0.0.va, line 14: Use of TAB characters discouraged
STYLE in file mvsg_cmc_2.0.0.va, line 18: Space character at end of line
STYLE in file mvsg_cmc_2.0.0.va, line 68: Space character at end of line
STYLE in file mvsg_cmc_2.0.0.va, line 77: Space character at end of line
    Further style comments of this type will be suppressed

…

WARNING in file mvsg_cmc_2.0.0.va, line 87: Parameter 'version' was never used
WARNING in file mvsg_cmc_2.0.0.va, line 154: Parameter 'flagfps1s' was never used
WARNING in file mvsg_cmc_2.0.0.va, line 173: Parameter 'flagfps2s' was never used
WARNING in file mvsg_cmc_2.0.0.va, line 192: Parameter 'flagfps3s' was never used
WARNING in file mvsg_cmc_2.0.0.va, line 211: Parameter 'flagfps4s' was never used
```

VAMPYRE: MVSG_CMC 2.0.0

WARNING in file mvsg_cmc_2.0.0.va, line 154: Parameter 'flagfps1s' was never used
WARNING in file mvsg_cmc_2.0.0.va, line 173: Parameter 'flagfps2s' was never used
WARNING in file mvsg_cmc_2.0.0.va, line 192: Parameter 'flagfps3s' was never used
WARNING in file mvsg_cmc_2.0.0.va, line 211: Parameter 'flagfps4s' was never used

flagfps1 should have been flagfps1s in this call:

idsfps1 = calc_iq(idsfps1,qgsfps1,qgdfps1,qcfps1,qbfps1,qsfps1,vtdiblfps1,vdsatfps1,vgsfps1,vdsfps1,1,vcfps1,vbfps1,flagfps1,tdut,tnomk,phit,w,lgfps1,cgfps1,cfps1s,ccfps1,cbfps1,vtofps1,sfps1,delta1fps1,0.0,ndfps1,alphafps1,vx0fps1,mu0fps1,betafps1,mthetafps1,vthetafps1,vtzeta,dibsat,epsilon,vzeta,lambda,ngf,type);

(compare use of “flagfp1s” and “flagfp1”)

VAMPyRE: ASM-HEMT 101.1.0

```
Reading asmhemt.va
STYLE in file asmhemt.va, line 8: Space character at end of line
STYLE in file asmhemt.va, line 15: Space character at end of line
STYLE in file asmhemt.va, line 357: Space character at end of line
STYLE in file asmhemt.va, line 367: Space character at end of line
STYLE in file asmhemt.va, line 417: Space character at end of line
    Further style comments of this type will be suppressed

STYLE in file asmhemt.va, line 736: Multiple assignments on a single line
STYLE in file asmhemt.va, line 736: Multiple assignments on a single line
STYLE in file asmhemt.va, line 736: Multiple assignments on a single line
STYLE in file asmhemt.va, line 736: Multiple assignments on a single line
STYLE in file asmhemt.va, line 736: Multiple assignments on a single line
    Further style comments of this type will be suppressed

STYLE in file asmhemt.va, line 790: Multiple contributions on a single line
WARNING in file asmhemt.va, line 1572: Call to ddx(), but Ids does not depend on V(b)
WARNING in file asmhemt.va, line 1587: Call to ddx(), but qgi does not depend on V(b)
WARNING in file asmhemt.va, line 1591: Call to ddx(), but qdi does not depend on V(b)
WARNING in file asmhemt.va, line 1595: Call to ddx(), but qsi does not depend on V(b)
WARNING in file asmhemt.va, line 1596: Call to ddx(), but qbi does not depend on V(b)
WARNING in file asmhemt.va, line 481: Parameter 'rigddio' was never used
```

VAMPyRE: ASM-HEMT 101.1.0

WARNING in file asmhemt.va, line 481: Parameter 'rigddio' was never used

```
`MPRco( rigsdio      ,1.0e-15      ,"A/m^2"      ,0.0      ,inf …
`MPRco( rigddio      ,1.0e-15      ,"A/m^2"      ,0.0      ,inf …
```

```
rigsdio_t = rigsdio*exp(rktgs*(Tdev/Tnom-1));
```
OK

Copy & paste error:

```
rigddio_t = rigsdio*exp(rktgd*(Tdev/Tnom-1));
```

VAMPyRE: PSP103.8 beta

```
Reading psp103.va
Reading psp/psp103.va
Reading psp/discipline.h
Reading psp/Common103_macrodefs.include
Reading psp/JUNCAP200_macrodefs.include
Reading psp/PSP103_macrodefs.include
Reading psp/PSP103_module.include
Reading psp/PSP103_parlist.include
Reading psp/JUNCAP200_parlist.include
Reading psp/JUNCAP200_varlist.include
Reading psp/JUNCAP200_InitModel.include
Reading psp/PSP103_scaling.include
WARNING in file psp/PSP103_scaling.include, line 675: Possible division by zero for parameter SCREF
WARNING in file psp/PSP103_module.include, line 1798: Bias-dependent '==': if (sqid==0.0)
WARNING in file psp/PSP103_module.include, line 2112: Bias-dependent '==': if (sid==0.0)
WARNING in file psp/PSP103_parlist.include, line 31: Parameter 'LEVEL' was never used
WARNING in file psp/PSP103_parlist.include, line 875: Parameter 'LMIN' was never used
WARNING in file psp/PSP103_parlist.include, line 876: Parameter 'LMAX' was never used
WARNING in file psp/PSP103_parlist.include, line 877: Parameter 'WMIN' was never used
WARNING in file psp/PSP103_parlist.include, line 878: Parameter 'WMAX' was never used
WARNING in file psp/psp103.va, line 50: Variable 'initial_instance.JUNCAPexpressInit.qjunbot' was never used
WARNING in file psp/psp103.va, line 50: Variable 'initial_instance.JUNCAPexpressInit.qjunsti' was never used
WARNING in file psp/psp103.va, line 50: Variable 'initial_instance.JUNCAPexpressInit.qjungat' was never used

Module: PSP103VA

Instance parameters (32)

Model parameters (846)

Operating-point variables (236)
Possible hidden-state variables:
    asrh, idmult, vav, vbi_minus_vjsrh, vj, vjsrh, wdep, wsrh, zinv
```

VAMPyRE: Does it work?

- Tested against CMC standard models
  - BSIMCMG (111.1.0 beta): found issues with binning parameters
  - BSIMIMG (102.9.3 beta): found bias-dependent != (causes incorrect derivatives)
  - Hicum (L0/V2.0.0): found unused parameter F2VG and issues with CMC Verilog-A Code Standards
  - MVSG (2.0.0): found unused parameters (bugs! wrong parameters used)
  - ASM-HEMT (101.1.0): found unused parameter (copy&paste bug)
  - PSP (103.8.0 beta): found possible division by zero (also found unused “inorm” but only after NXP had reported it already)

Release History

- Released under Educational Community License (ECL-2.0)
- First release: v1.0 on July 25, 2020
- Version 1.1: add checks for compliance with CMC Verilog-A Code Standards
- Versions 1.2-1.5: minor revisions
- Current version: 1.6, November 25, 2020
- Distribution through the CMC / Silicon Integration Initiative web site `https://si2.org/cmc`