gnucap – recent work and directions

Felix Salfelder

FOSDEM 2017

UNIVERSITY OF LEEDS

gnucap – recent work and directions

- About
- Getting Started
- gnucap-geda (gEDA interoperability)
- gnucap-adms (ADMS-va model compiler)
- outlook

About gnucap – GNU Circuit Analysis Package

- 1983. First traces (Albert Davis)
- 1990. *ACS*, Al's Circuit Simulator
- 1992. GPL
- 2001. Renamed to *gnucap*, a GNU project
- 2013. Source repos at `git.savannah.gnu.org`
- since 2015: gnucap-uf unforking and new extensions

About gnucap – features

- (single engine) mixed signal kernel
  - scalable efficient algorithms
  - queues, bypassing
  - cross events, curve fitting
  - automatic step control
- interactive user interface
- stable C++ library, fully pluggable
- Multi-language
  - spice, verilog, spectre
  - more as plugins

About gnucap – pluggability

- What is a "Plugin"?

About gnucap – pluggability

- What is a "Plugin"?
  - *Run time* extension (see `dlopen(3)`)
  - Register to *dispatcher* (dictionary) upon loading
  - Reduce need for time bombs and forks
  - unlimited customization
  - Increased code quality and flexibility

About gnucap – pluggability

- What is a "Plugin"?
- Plugin classes
  - Components, models
  - Commands, algorithms
  - Functions
  - Measurements, post-processing
  - Netlist/schematic languages
  - Interactive help

Getting started

- install gnucap
  - currently: from git, unstable branch
  - `./configure; make install`
  - or use `autotools` branch
  - distro packages (arch)

Getting started

- install gnucap
- install extension
    - see `README`, `INSTALL`
    - usually: (`configure`, `make install`)
    - or install distro packages

Getting started

- install gnucap
- install extension
- load extensions
  - `$ gnucap -a extension.so -a more.so -a ...`
  - e. g. startup wrapper
  - or: interactive `load` command
  - `rc` file

Getting started

- install gnucap
- install extension
- load extensions
- with `gnucap-make`
    - handcode `plugin.cc`
    - `$ gnucap -a gnucap_make.so`
    - `> load plugin.cc`

Now available

- gnucap-geda (gEDA interoperability)[†] [*]
- gnucap-adms (ADMS-va model compiler)[†] [*]
- gnucap-random (random variables)[*]
- gnucap-make (compile plugins on-demand)[†] [*]
- gnucap-qucs (qucsator replacement) [†]
- gnucap-jack (rt audio processor) [†] [*]
- model packs (spice, bsim, ..)

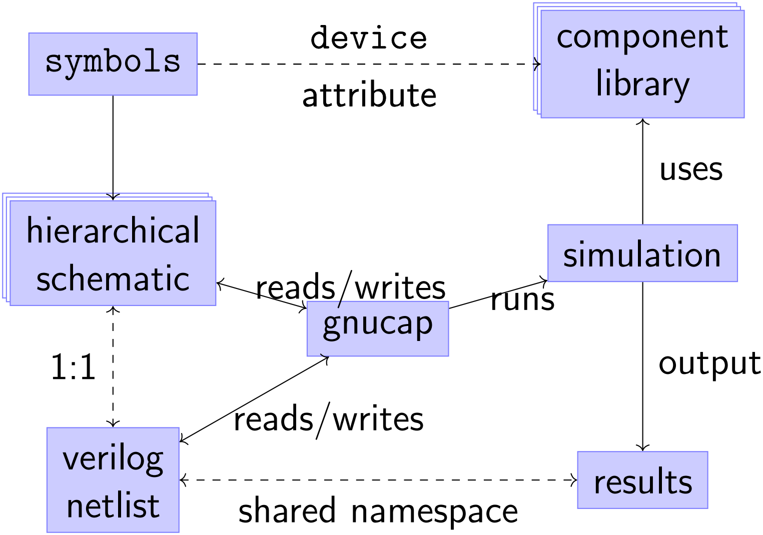
[†] ported from gnucap-uf

[*] packaged for arch linux (AUR)

gnucap-geda

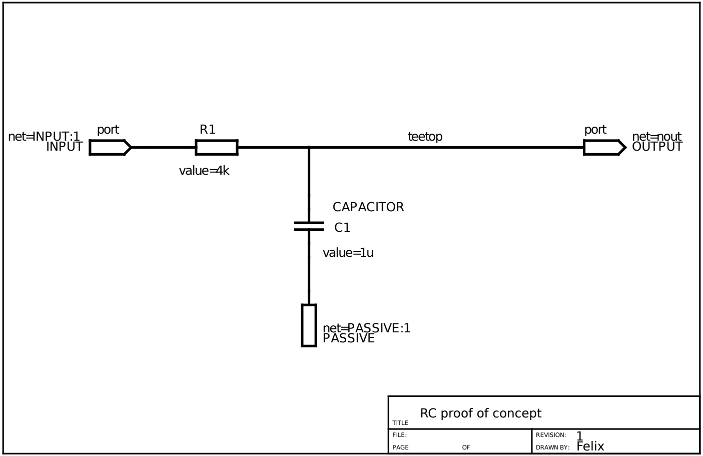
- GSoC project (2012?, Savant Krishna)
- idea: augmented netlist as schematic
- gEDA file exchange
- analyse/simulate schematic + component library
- sckt plugin, main line (new)
- full gEDA hierarchy support
- default port values (new, experimental)
- reduce need for spice-sdb

gnucap-geda architecture



gnucap-geda, 1:1 translation

- gEDA schematic (`rc.sch`)



gnucap-geda, 1:1 translation

- gEDA schematic (`rc.sch`)

```
C 45500 47200 1 0 0 resistor-2.sym
{
T 45900 47550 5 10 0 0 0 0 1
device=RESISTOR
T 45700 47500 5 10 1 1 0 0 1
refdes=R1
T 45400 46900 5 10 1 0 0 0 1
value=4k
}
N 45500 47300 44900 47300 4
```

gnucap-geda, 1:1 translation

- gEDA schematic (`rc.sch`)

```
C 45500 47200 1 0 0 resistor-2.sym
{
T 45900 47550 5 10 0 0 0 0 1
device=RESISTOR
T 45700 47500 5 10 1 1 0 0 1
refdes=R1
T 45400 46900 5 10 1 0 0 0 1
value=4k
}
N 45500 47300 44900 47300 4
```

- Verilog netlist representation

```
RESISTOR #(.basename(resistor-2.sym),.value(4k)) R1 (.1(x_cn
place #(.x(45500),.y(47300)) 45500:47300 (.port(x_cn_4));
place #(.x(46400),.y(47300)) 46400:47300 (.port(x_cn_5));
net #() net1 (.p(x_cn_4),.n(x_cn_2));
```

gnucap-geda, more

- component library supplementing gEDA symbols
- custom modules easy to integrate e. g.
  - spice macrocells
  - modelcards etc.
  - verilog-a models
- live examples included (new)
  - opamp analysis
  - frequency divider
  - comparator simulation

gnucap-geda in practice

```
load gnucap_geda.so

include analog.v                                  // gEDA component
include switch.v                                  //       libraries
geda "myfile.sch" module device="mydevice" // fetch schematic

verilog                                 // switch language
list                                    // print schematic

mydevice #(.xyz(3)) m1(1 2);            // instanciate

// used to spice?
// run some simulation
.spice
V1 1 0 ac 1
C1 2 0 1p

.print ac v(2)
.ac 1 1024 *2
```

about gnucap-adms

- Turns verilog-a models into component plugins
- finally ported to upstream gnucap
- Uses admsXml, but more futuristic
  - custom rules and templates
  - towards modelgen inspired architecture
  - controlled sources, no direct jacobian
  - essentially static C++

about gnucap-adms

- Turns verilog-a models into component plugins
- finally ported to upstream gnucap
- Uses admsXml, but more futuristic
  - custom rules and templates
  - towards modelgen inspired architecture
  - controlled sources, no direct jacobian
  - essentially static C++
  - still band-aid

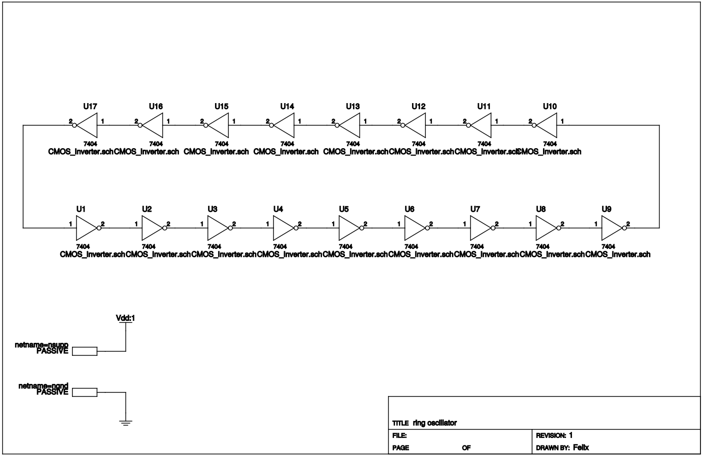
gnucap-adms features

- Voltage sources, current probes
- Custom subcircuit components
- Linear operators (`ddt`, `idt`, `ddx`)
- on-demand compilation
- Needs work
    - admsXml use is limited
    - evaluation routines need untangling
    - shift to modelgen (or icarus)
    - compile/distribute IP blocks

gnucap-adms features

- Voltage sources, current probes
- Custom subcircuit components
- Linear operators (`ddt`, `idt`, `ddx`)
- on-demand compilation
- Needs work
    - admsXml use is limited
    - evaluation routines need untangling
    - shift to modelgen (or icarus)
    - compile/distribute IP blocks
    - full -ams support

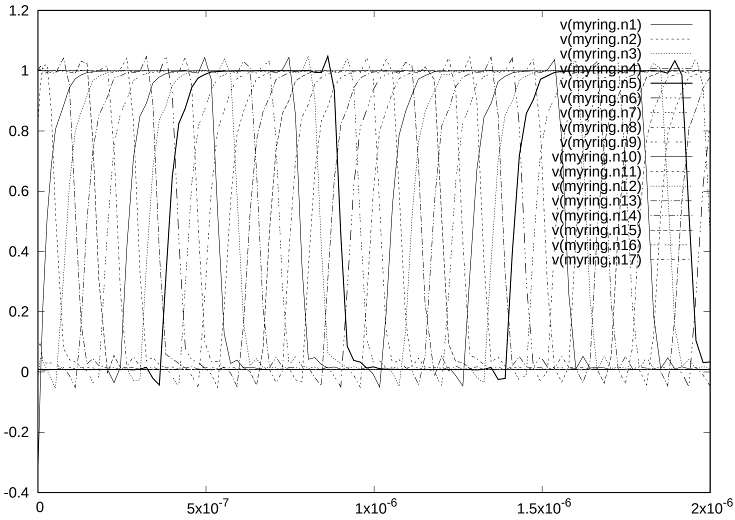
gnucap-adms in practice



gnucap-adms in practice

```
load lang_geda.so
load lang_adms.so

// compile and load verilog-a
ahdl_include bsim6.va
include "modelcard.nmos"
include "modelcard.pmos"

geda "ring17.sch" module device="ring"

verilog
vsource #(.v(1)) vsupp(vdd 0);
ring myring(vdd 0);

print tran v(myring.n*)

// writes into ascii table
tran 0 2u > tran.out
end
```

gnucap-adms in practice



outlook

- Cooperation
  - QUCS engine, `gnucsator`?
  - KiCad, file exchange?
- improved usability, packageing
- unforking gnucap-uf
  - more analysis, algorithms
  - more scripting
  - more devices
  - more spice support
  - tons of unfinished drafts

Thank You.