

# Gnucap – Schaltungssimulation mit freier Software

Felix Salfelder

IGER 23



- ▶ Was ist Gnucap und wozu?
- ▶ Zwei Beispiele
- ▶ Hardwarebeschreibung
- ▶ Simulation
- ▶ Call for Participation

# Was ist Gnuicap?

- ▶ GNU Circuit Analysis Package
  - ▶ 1990 ACS, AI's Circuit Simulator
  - ▶ 2001 C++ Port, umbenannt in Gnuicap
- ▶ Modellierung und Simulation von Schaltungen
  - ▶ analog, kontinuierliche Signale
  - ▶ digital, diskrete Logik
- ▶ Motivation und Features
  - ▶ Erweiterbarkeit, Plug-ins
  - ▶ Neue Algorithmen
  - ▶ Hardwarebeschreibungssprache

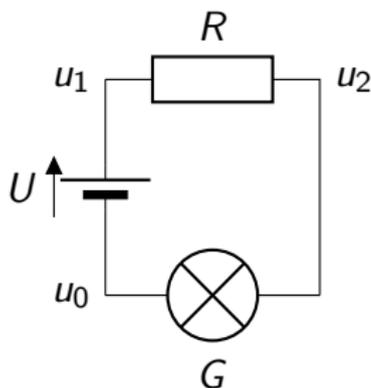
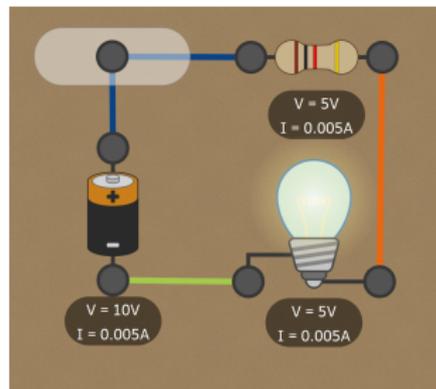
# Wozu: Modellierung und Simulation

- ▶ Schaltung: miteinander verbundene Bauteile



- ▶ Elektrisches Verhalten der Bauteile z.B.
  - ▶ Ohmsches Gesetz,  $I = U/R$ .
  - ▶ Gesteuerte Quellen,  $V_{\text{out}} = 100 * V_{\text{in}}$
  - ▶ Ereignisse, on input change: `delay()`; `out = !in`
- ▶ Simulation: berechne
  - ▶ Arbeitspunkt
  - ▶ Zeitverhalten
  - ▶ Übertragungsfunktion
  - ▶ ...

## Beispiel: Analogschaltung zum Nachrechnen



- ▶ Spannungsquelle:

$$(u_1 - u_0) = U$$

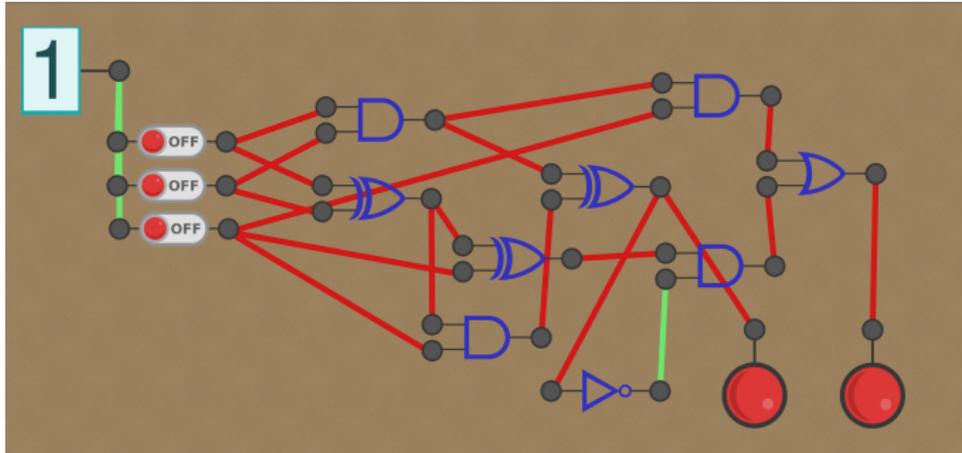
- ▶ Knotenregel: Ströme addieren sich zu Null.

$$\text{In } u_2: (u_1 - u_2)/R - (u_2 - u_0) * G = 0$$

- ▶ Referenz:  $u_0 = 0$

$$\begin{pmatrix} -1 & 1 & 0 \\ -G & G + 1/R & -1/R \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} u_0 \\ u_1 \\ u_2 \end{pmatrix} = \begin{pmatrix} U \\ 0 \\ 0 \end{pmatrix}$$

# Beispiel: Kombinatorische Schaltung

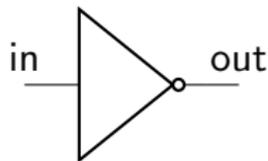


- ▶ Hier: 0/1 Rot/Grün statt Spannungen
- ▶ Keine Ströme notwendig (aber: Ereignisse, Verzögerung)
- ▶ Nachrechnen mit Stack und Funktionstabellen:  
Selective-trace Algorithmus

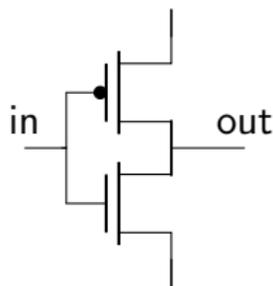
# Beschreibungssprachen (HDL)

- ▶ 1973: L. Nagel: SPICE, Analogsimulation
  - ▶ Schaltpläne, Netzlisten: Punch Cards
  - ▶ Modelle: Fortran, später C.
- ▶ 1988-1996: EDIF, „vendor-neutral format“
  - ▶ Schaltpläne und Netzlisten
- ▶ 1984: P. Moorby: Verilog. Frei seit 1995.
  - ▶ Simulation, Verifikation, Synthese *digitaler* Schaltungen
  - ▶ Vereinheitlicht Bauteilmodellierung und Schaltpläne
- ▶ ab 1998: Verilog-AMS: Analog/Mixed-Signal Erweiterung
  - ▶ Ersetzt SPICE, definiert Kompatibilität
  - ▶ Hat sich durchgesetzt (gegen VHDL etc.)
  - ▶ Industriestandard (!)
- ▶ Unterstützung in *open source* Tools
  - ▶ seit 1998: Icarus Verilog: digital
  - ▶ seit 2001: GnuCap: Verilog Schaltpläne
  - ▶ ADMS, OpenVAF: Verilog-A für SPICE Simulatoren
  - ▶ Dez. 2022: GnuCap wieder aufgenommen: Verilog-AMS

# Modellierung in Verilog/Verilog-A



```
module inverter(out, in);  
    output out;  
    input in;  
  
    assign  
        #'delay out = in;  
endmodule
```



```
module inverter(out, in, vdd, gnd);  
    inout out, in, vdd, gnd;  
    electrical out, in, vdd, gnd;  
  
    // NEU: gnucap-modelgen-verilog  
    // instanziiere Modelle  
    pch #() p(out, in, vdd, vdd);  
    nch #() n(out, in, gnd, gnd);  
endmodule
```

# Modellierung in Verilog-AMS

```
module inverter(out, in, vdd, gnd);
    output out;
    input in, vdd, gnd
    electrical out, in, vdd, gnd;

    analog begin
        real v;
        v = V(in, gnd) - V(vdd, gnd)/2.;
        @(cross(v));
        V(out, gnd) <+ transition(v<0., 0., 'delay)*V(vdd, gnd);
    end
endmodule
```

- ▶ Ereignisbasiert, aber analog
- ▶ Sprachunterstützung: geplant 2024

## Konversion von Signalpegeln

```
connectmodule voltage_to_ttl5(in, out);
    input in;
    output out; reg out;
    electrical in;
    ttl5 out;
    always @(cross(V(in) - 2.5, 1))
        out = 1;
    always @(cross(V(in) - 2.5, -1))
        out = 0;
endmodule
connectmodule ttl5_to_voltage(in, out);
    [..]
endmodule
```

- ▶ Übergang zwischen *disciplines*: benutzerdefiniert
- ▶ Essentiell für Analog/Mixed-Signal Modellierung
- ▶ Unterstützung: geplant 2024

# Simulation: Stand der Technik

- ▶ 1989 T. Quarles: SPICE3, aka. ngSPICE
  - ▶ Portiert von Fortran nach C
  - ▶ BSD lizenziert, unzählige Forks
  - ▶ Globaler Gleichungslöser, analog
- ▶ 1991: Al Davis: ACS
  - ▶ Al's Dissertation: Skalierender Löser
  - ▶ Erster funktionierender "fast SPICE" Algorithmus
  - ▶ Ereignisse, mixed-signal, multi-rate, ..
  - ▶ bis 2001: Portiert von Fortran über C nach C++, *Gnucap*
- ▶ Seither
  - ▶ "Fast SPICE" setzt sich in der Industrie durch
  - ▶ Keine quelloffenen Weiterentwicklungen
  - ▶ Gnucap: noch nicht fertig

# Simulation: Bedarf

- ▶ 2019: Free Silicon Foundation gegründet  
Ziel: Freie Hardware mit F/OSS tools, FSi Conference
- ▶ ab Dez. 2022: NLnet finanziert "Verilog-AMS in Gnuicap"  
Projekt  
Fast fertig: Verilog-A Unterstützung in modelgen-verilog.
- ▶ April 2023: EU Chips Act:  
43 Milliarden Euro für Innovationsförderung etc.
- ▶ FSiC'23: Erste Analog/Mixed Signal Chips mit quelloffenen  
Tools
  - ▶ SAR-ADC aus Linz: ngSpice und Xyce am Anschlag.
  - ▶ "IHP aims to enable analog circuit design with open source software"

Keine freie Hardware ohne freien Simulator. Wer baut ihn?

# Call for Participation

- ▶ Es gibt beliebig viel zu tun
- ▶ Nächste NLnet Runde: 1. August
- ▶ Voraussichtlich: Weitere Entwickler im Team
- ▶ Geplant: Kollaboration mit Forschungseinrichtung
- ▶ Möglich: Praktikum für Studenten
- ▶ Jeder darf mitmachen, jetzt einlesen!

# Call for Participation

Wir könnten gebrauchen:

- ▶ Diverse Bauteilmodelle
- ▶ Befehle, Algorithmen
- ▶ Tutorials, Dokumentation
- ▶ Kompatibilität mit anderen Projekten
- ▶ Unterstützung für Dateiformate
- ▶ Fertigstellung der Python Bibliothek
- ▶ Graphische Benutzeroberfläche

Finanzierung ist möglich.

Vielen Dank.