

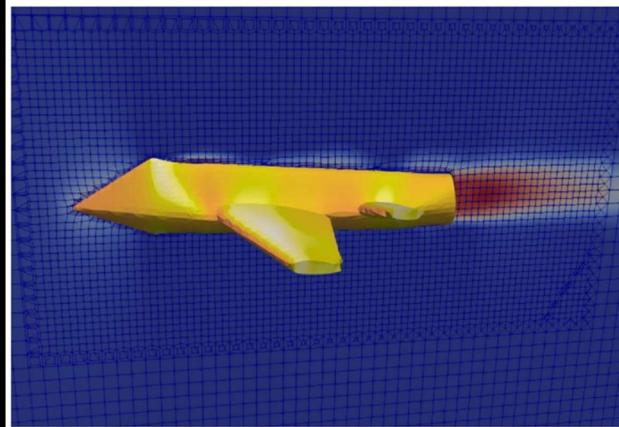
# (第2回)ngspiceのpython操作方法検討 他

8Dec 2025

AirtomoR

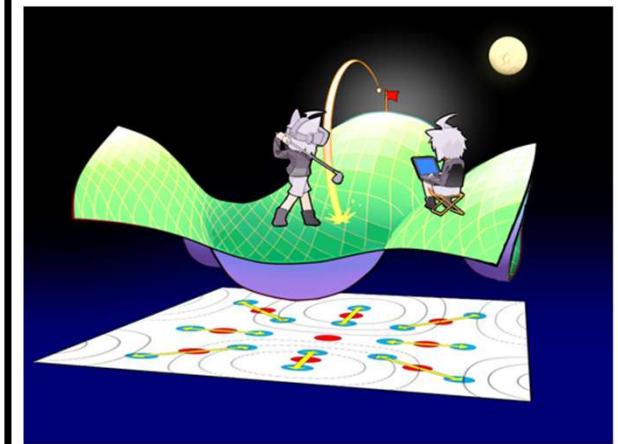
Kindle unlimited  
で読んでね

3D 物理シミュレーションで実践できる  
多パラメータ最適化の新手法  
( アップデート版 )  
パラメータ分配スコア - パラメータ分析ユーティリティソフト ( GUI,python3-pyside2 )  
3D 形状最適化シミュレーションモデル 有



3年前位

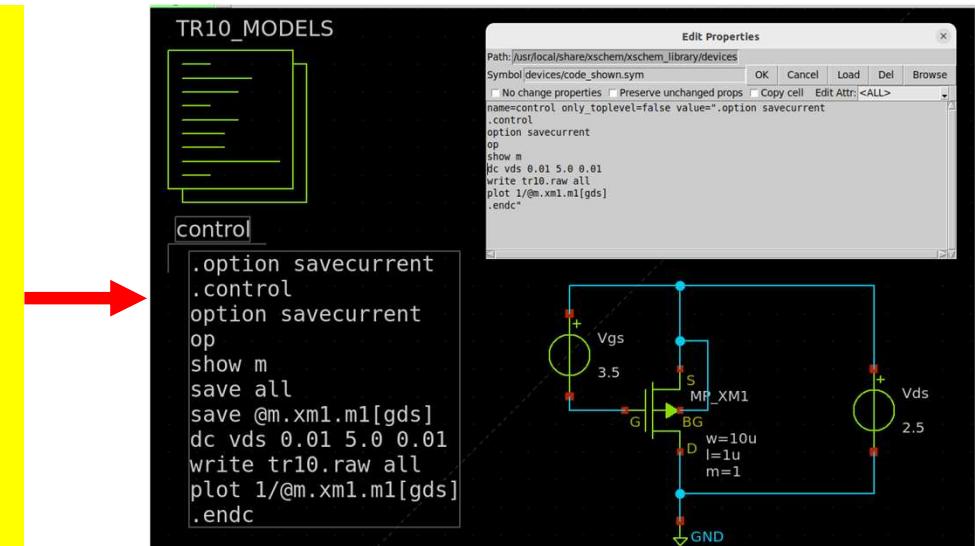
グリッドサーチ  
多目的値多パラメータ最適化  
ソフトウェアの拡張機能解説と  
物理シミュレーション例示



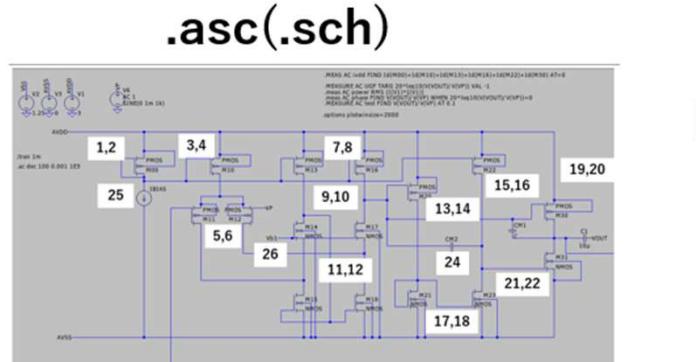
今年5月

# .rawファイル出力できずは解決

writeを入れても  
rawファイルが出力されなかつたが  
bashからの実行として  
ngspice -i 2stage\_ac.spice  
とすれば  
write行で指定した名前の  
rawファイルが生成



**.sch**から**.spice**も**bash**から実行できました



## 変数編集コード

.net(.spice)

```

IBIAS N001 AVSS
CM2 N007 N002 C
CM1 N002 N006 C
V1 AVDD 0 3
V2 Vb1 0 1.25
V3 AVSS 0 0
V6 VP 0 SINE(0 1m 1k) AC 1
M00 N001 N001 AVDD AVDD PMOS
M10 N003 N001 AVDD AVDD PMOS
M11 N009 N004 N003 N003 PMOS
M12 N008 VP N003 N003 PMOS
M13 N005 N001 AVDD AVDD PMOS
M16 N002 N001 AVDD AVDD PMOS
M14 N005 Vb1 N009 AVSS NMOS
M15 N009 N005 AVSS AVSS NMOS
M17 N002 Vb1 N008 AVSS NMOS
M18 N008 N005 AVSS AVSS NMOS

```

.log(.raw経由からも可)

```
test: v(vout)/v(vp)=(105.868dB,-3.84881°) at 0.1
phase: v(vout)/v(vp)=(0.000558095dB,-94.7432°) at 286041
power: RMS(i(v1))=0.000256367 FROM 0.001 TO 1e+009
ugf=319960 FROM 0.001 TO 319960
ivdd: i(v1)=(-20.9287dB,-3.84987°) at 0.1
```

```
1 with open("./temp/amp3_1.log",encoding='CP932') as f:
2     s = f.read()
3     lines = s.split("\n")
4     test_result_list1 = []
5     print(lines[7],lines[8])
6     for i in range(len(lines)):
7         if lines[i][:6] == "power:":
8             print("power found")
9             test_result_list1= re.findall(pattern, lines[i] )
10            print(test_result_list1)
11
12 test_result1 = test_result_list1[1]
13 #test_result1=1000.0*(10**-(0.05*float(test_result1)))
14 test_result1=1000*float(test_result1)
15 test result1=str(test result1)
```

## スコア取得コード

LTS spiceなら spicelib で左のような GUI モデルとして素子値にアクセス化(但し一部未実装) xschem はどうやらできなさそう。KiCAD だと python 対応可能らしい(KiCAD 公式で対応化?)

.schから.spiceは  
xschem your\_circuit.sch --command "xschem netlist; simulate"  
で生成可能(ただ出力フォルダ先はデフォルトだとプログラムフォルダ?)  
.schさえ作ればbashで.rawまで出せる=pythonプログラムに載せられる

# (先週と同じ)パラメータ更新はGrokでルーチン化可能

明日夜の勉強会でそのまま投げて使ってください！

5分で決まるように「選択肢+チェック欄」形式にしてあります。

【チーム名 / プロジェクト名】 \_\_\_\_\_

1. 対象プロセス（複数選択可）  
 sky130A  GF180MCU  IHP SG13G2  TR10 (東海理化180nm BiCD)  その他: \_\_\_\_\_

2. シミュレーション作業ディレクトリ  
 xschem/ の直下に sim/ を作る（推奨）  
 各回路ごとに sim\_<回路名>/ を作る  
 回路名+cornerで sim\_<回路名>\_<corner>/ を作る（例: sim\_opamp\_tt）

3. ファイル命名規則（絶対に変えない）  
netlist → {回路名}.cir

4. xschem CLI コマンド（固定）  
xschem -q -n -s --netlist\_filename {cirファイル} {schファイル}

5. ngspice CLI コマンド（固定）  
ngspice -b -r {rawファイル} -o {logファイル} {cirファイル}

6. Pythonスクリプトの必須構成（私が毎回この通りに書く）  
【必須関数】

```
def run_simulation(sch_file: str | Path, corner: str = "tt") -> dict:  
    # 戻り値は必ず dict で以下を含む  
    return {  
        "sch": Path,  
        "cir": Path,  
        "raw": Path,  
        "log": Path,  
        "work_dir": Path
```

}

【必須インポート】

```
import subprocess  
from pathlib import Path
```

【エラー処理】

失敗したら必ず raise RuntimeError(エラーメッセージの最後20行)

7. コーディングスタイル（私が厳守するもの）  
・変数名は snake\_case  
・日本語コメント可（むしろ推奨）  
・spicelib.RawRead で読む例を最後に必ず付ける  
・バックアップは作らない（gitで管理する前提）

8. モデルファイルのinclude方法（チームで統一）  
 .sch の一番下に .include "../models/tr10\_tt.lib" のように書く  
 xschemrc に全局includeを書く（上級者向け）

9. 追加で欲しい便利関数（チェックでON/OFF）  
 自動で matplotlib プロットしてPNG保存  
 corner一括実行 (tt/ss/ff/mc) を並列で回す  
 W/L/m を引数で変えてスイープ実行  
 測定値 (gm, id, vthなど) を自動抽出してcsv吐き出し

10. その他チーム特有ルール（自由記入）  
\_\_\_\_\_

Netlist段階でパラメータ更新は人間の見通しが悪いけど  
LLMにやらせると決めれば特に悩みもないかも？(テンプ  
レ化しておけば再現性のある出力をしてもらえるとの事)

# 今日の相談

・Iueさん作のVCOプロジェクトを動かせるはずの環境をセットした  
はずがエラー。EDAを再確認して、もし間違っていたら  
再セット方法を御教授願います。

(ただIueさん作のVCOが載っているgitリンク見失った。

下記リンクのsinby社のVCOや3zkiさんのクロックダブラーを実施するかも?)

[https://github.com/ishi-kai/ISHI-KAI\\_Multiple\\_Projects\\_OpenMPW\\_PTC06-1](https://github.com/ishi-kai/ISHI-KAI_Multiple_Projects_OpenMPW_PTC06-1)

```
Virtual-Platform:~$ bash eda-setup.sh
```

```
Virtual-Platform:~$ bash pdk_PTC06-setup.sh
```

エラー  
GrokはPDKが合っていないのでは?  
といった回答

