Notes on building technology LUTs for the Gm/ID method with ELDO

gm_ID_starter_kit_v2.3 (Boris Murmann Rev. 20190921)
(as of 27 July 2021, this is the most recent version)

Reference:
P.G.A. Jesper and B. Murmann, Systematic Design of Analog CMOS Circuits Using Pre-Computed Lookup Tables, Cambridge University Press 2017
The textbook is based on gm_ID_starter_kit_v2.1

Procedure to generate the .mat technology files for Stanford's CMOS 180nm process with ELDO

The .mat files generation is performed running the top-level matlab script:

- /usr/class/gmidLUTs/sweepfiles/techsweep_eldo_run.m

The top-level script "calls" an auxiliary matlab configuration script that setups all required tools and paths and that contains the netlist template necessary to auto-generate the simulations that allows collecting the technology data saved in the .mat files.

- /usr/class/gmidLUTs/sweepfiles/techsweep_cfg_bsim3_180_eldo.m

The spice model of Stanford's CMOS 180nm technology is in:

- /usr/class/models/ee214_hspice.mod

Since, running the top-level script can take quite a long time, it is advisable to first run the following utility script:

- /usr/class/gmidLUTs/sweepfiles/techsweep_eldo_debug.m

The utility script is designed for debugging the configuration script. Basically, this is a simplified version of techsweep_eldo_run.m that runs just one single simulation sweep and displays the parameters at mid supply. Once this work, the full generation of the .mat files can be kicked off and should work without any problem.

The .mat files saved running the script techsweep_eldo_run.m are:

- /usr/class/gmidTECHs/180nch_e.mat
- /usr/class/gmidTECHs/180pch_e.mat

NOTE:
To run the top-level matlab script directly from the Linux shell create a shell script atfile.sh with the following content

```
unset DISPLAY
nohup nice matlab > matlab_techsweep_eldo_run.out 2>&1 << EOF
techsweep_eldo_run
exit
EOF
```

and run it as follows:

```
$ at –f atfile.sh now
```

To monitor the execution progress do as follows:

```
$ tail –f atfile.sh now
```

Example to test the .mat files and to illustrate how to use the look_up and the look_upVGS functions

Run the following script:

- /usr/class/gmidLUTs/test_lookup_eldo.m
- /usr/class/gmidLUTs/test_lookupVGS_eldo.m

test_lookup_eldo.m

```
% File name: test_lookup_eldo.m
% Test script for function "look_up"
clearvars;
close all;

addpath('/usr/class/gmidLUTs;/usr/class/gmidTECHs')
% addpath('/usr/local/MATLAB/HspiceToolbox')

% ----------------------------------------
% all nmos data is contained in structure nch
% ----------------------------------------
load('/usr/class/gmidTECHs/180nch_e.mat');
device = nch;
L = device.L;

%Plot ID versus VDS
vds = device.VDS;
vgs = 0.4:0.05:0.6;
ID = look_up(device, 'ID', 'VDS', vds, 'VGS', vgs);
figure;
plot(vds, ID)
ylabel('I_D [A]')
xlabel('V_D_S [V]')
grid;

% Plot Vt against L
vt = look_up(device, 'VT', 'VGS', 0.6, 'L', L);
figure;
plot(L, vt)
ylabel('V_t [V]')
xlabel('L [um]')
grid;

% Plot ft against gm_id for different L
gm_id = 5:0.1:20;
ft = look_up(device, 'GM_CGG', 'GM_ID', gm_id, 'L', min(L):0.05:0.3)/2/pi;
figure;
plot(gm_id, ft)
xlabel('g_m/I_D [S/A]')
ylabel('f_T [Hz]')
grid;

%Plot id/w against gm_id for different L
gm_id = 5:0.1:20;
id_w = look_up(device, 'ID_W', 'GM_ID', gm_id, 'L', min(L):0.05:0.3);
figure;
semilogy(gm_id, id_w)
xlabel('g_m/I_D [S/A]')
ylabel('I_D/W [A/m]')

%Plot id/w against gm_id for different VDS (at minimum L)
gm_id = 5:0.1:20;
id_w = look_up(device, 'ID_W', 'GM_ID', gm_id, 'VDS', [0.8 1.0 1.2]);
figure;
semilogy(gm_id, id_w)
```

```
xlabel('g_m/I_D [S/A]')
ylabel('I_D/W [A/m]')

%Plot gm/gds against gm_id (at minimum L and default VDS)
gm_id = 5:0.1:20;
gm_gds = look_up(device,'GM_GDS','GM_ID', gm_id);
figure;
semilogy(gm_id, gm_gds)
xlabel('g_m/I_D [S/A]')
ylabel('g_m/g_d_s')

%Plot Thermal noise factor gamma
KB = 1.38e-23;
vgs = 0.4:25e-3:1.8;
gm_id = look_up(device, 'GM_ID', 'VGS', vgs, 'L', min(L));
gamma = look_up(device,'STH_GM', 'GM_ID', gm_id)/(4*KB*device.TEMP);
figure;
semilogy(gm_id, gamma);
xlabel('g_m/I_D [S/A]')
ylabel('\gamma_n')

%Plot Flicker noise Corner frequency at minimum L
vgs = 0.4:25e-3:1.8;
gm_id = look_up(device, 'GM_ID', 'VGS', vgs, 'L', min(L));
fco = look_up(device, 'SFL_STH', 'VGS', vgs, 'L', min(L));
figure;
semilogy(gm_id, fco);
xlabel('g_m/I_D [S/A]')
ylabel('f_c_o [Hz]')

%try invalid syntax
% ID = look_up(nch, 'ID', 'GM_ID', 8, 'GM', 0.00345)

%try invalid syntax
% wt = look_up(nch, 'CGD', 'GM_ID', 10);

% ------------------------------------------
% all pmos data is contained in structure pch
% ------------------------------------------

load('/usr/class/gmidTECHs/180pch_e.mat');
device = pch;
L = device.L;

%Plot ID versus VDS
vds = device.VDS;
vgs = 0.4:0.05:0.6;
ID = look_up(device, 'ID', 'VDS', vds, 'VGS', vgs);
figure;
plot(vds, ID)
ylabel('I_D [A]')
xlabel('V_D_S [V]')
grid;

% Plot Vt against L
vt = look_up(device, 'VT', 'VGS', 0.6, 'L', L);
figure;
plot(L, vt);
```

```
ylabel('V_t [V]')
xlabel('L [um]')
grid;

% Plot ft against gm_id for different L
gm_id = 5:0.1:20;
ft = look_up(device, 'GM_CGG', 'GM_ID', gm_id, 'L', min(L):0.05:0.3)/2/pi;
figure;
plot(gm_id, ft)
xlabel('g_m/I_D [S/A]')
ylabel('f_T [Hz]')
grid;

%Plot id/w against gm_id for different L
gm_id = 5:0.1:20;
id_w = look_up(device, 'ID_W', 'GM_ID', gm_id, 'L', min(L):0.05:0.3);
figure;
semilogy(gm_id, id_w)
xlabel('g_m/I_D [S/A]')
ylabel('I_D/W [A/m]')

%Plot id/w against gm_id for different VDS (at minimum L)
gm_id = 5:0.1:20;
id_w = look_up(device, 'ID_W', 'GM_ID', gm_id, 'VDS', [0.8 1.0 1.2]);
figure;
semilogy(gm_id, id_w)
xlabel('g_m/I_D [S/A]')
ylabel('I_D/W [A/m]')

%Plot gm/gds against gm_id (at minimum L and default VDS)
gm_id = 5:0.1:20;
gm_gds = look_up(device,'GM_GDS','GM_ID', gm_id);
figure;
semilogy(gm_id, gm_gds)
xlabel('g_m/I_D [S/A]')
ylabel('g_m/g_d_s')

%Plot Thermal noise factor gamma
KB = 1.38e-23;
vgs = 0.4:25e-3:1.8;
gm_id = look_up(device, 'GM_ID', 'VGS', vgs, 'L', min(L));
gamma = look_up(device,'STH_GM', 'GM_ID', gm_id)/(4*KB*device.TEMP);
figure;
semilogy(gm_id, gamma);
xlabel('g_m/I_D [S/A]')
ylabel('\gamma_p')

%Plot Flicker noise Corner frequency at minimum L
vgs = 0.4:25e-3:1.8;
gm_id = look_up(device, 'GM_ID', 'VGS', vgs, 'L', min(L));
fco = look_up(device, 'SFL_STH', 'VGS', vgs, 'L', min(L));
figure;
semilogy(gm_id, fco);
xlabel('g_m/I_D [S/A]')
ylabel('f_c_o [Hz]')
```

Results obtained running test_lookup_eldo for Stanford's 180nm process

nch

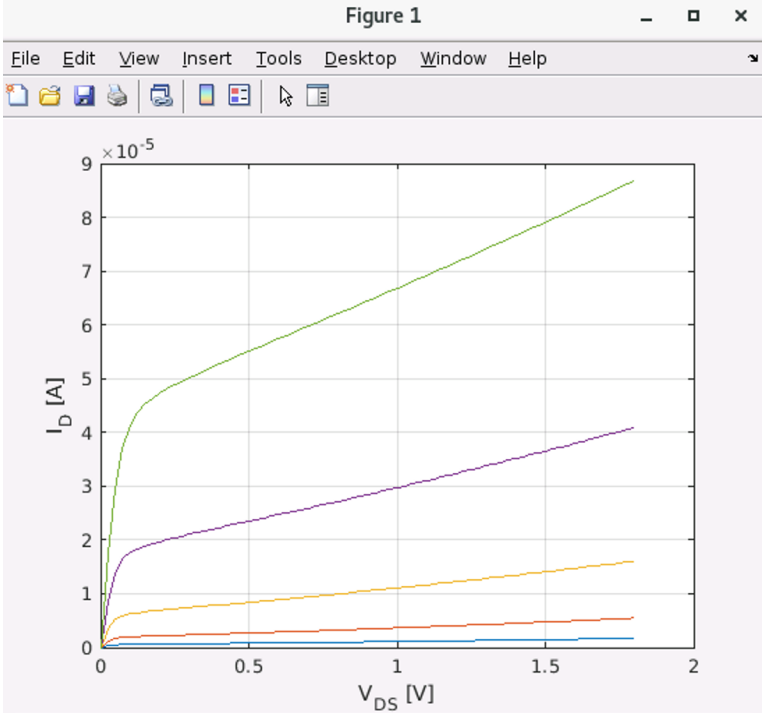


Figure 1. Plot ID vs. VDS

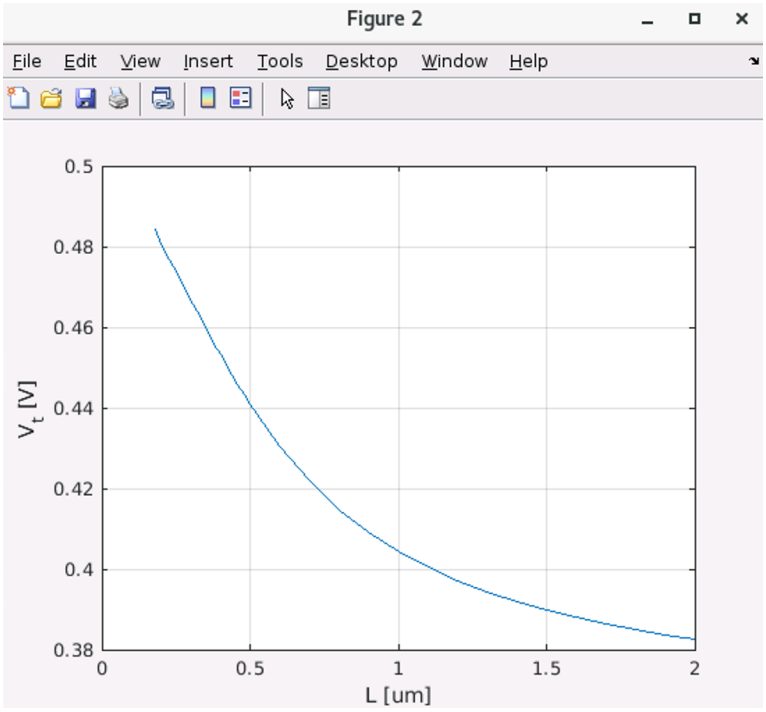


Figure 2. Plot Vt vs. L

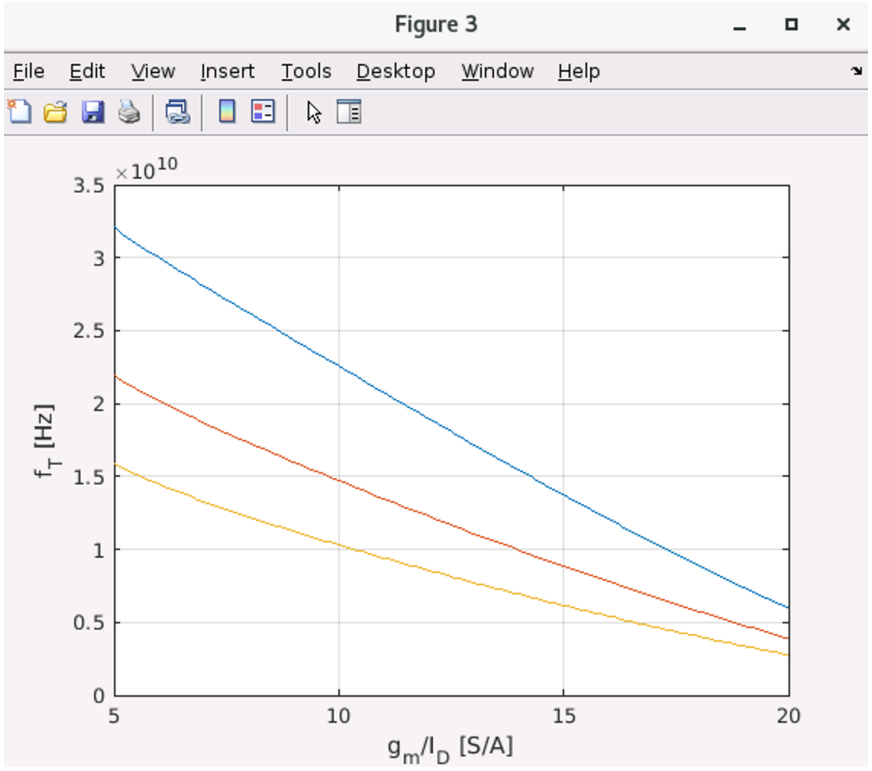


Figure 3. Plot fT vs. gm/ID for different L

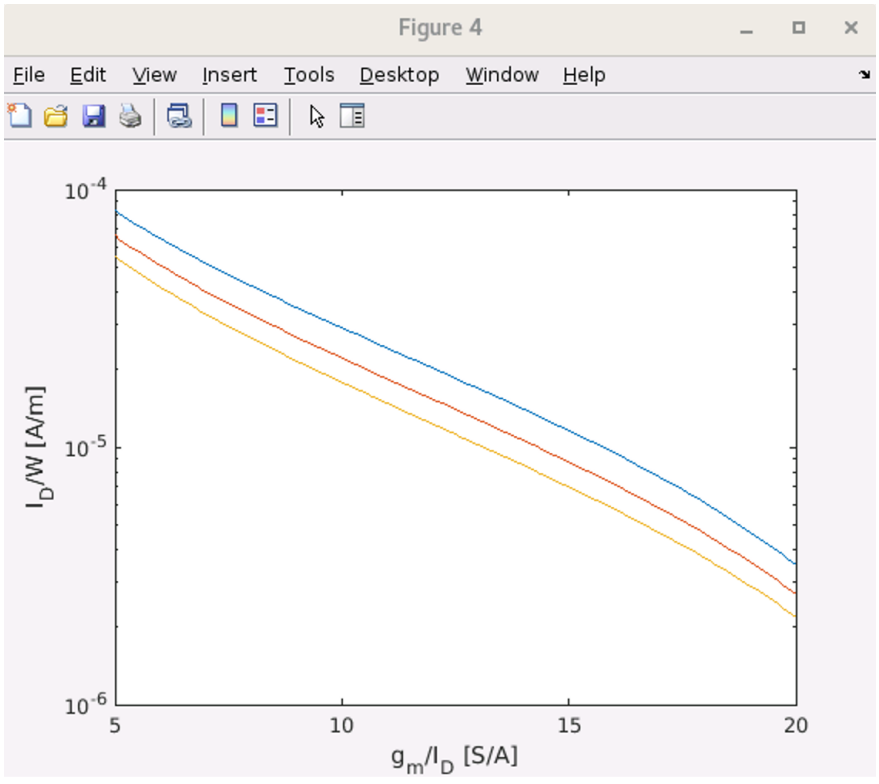


Figure 4. Plot ID/W vs. gm/ID for different L

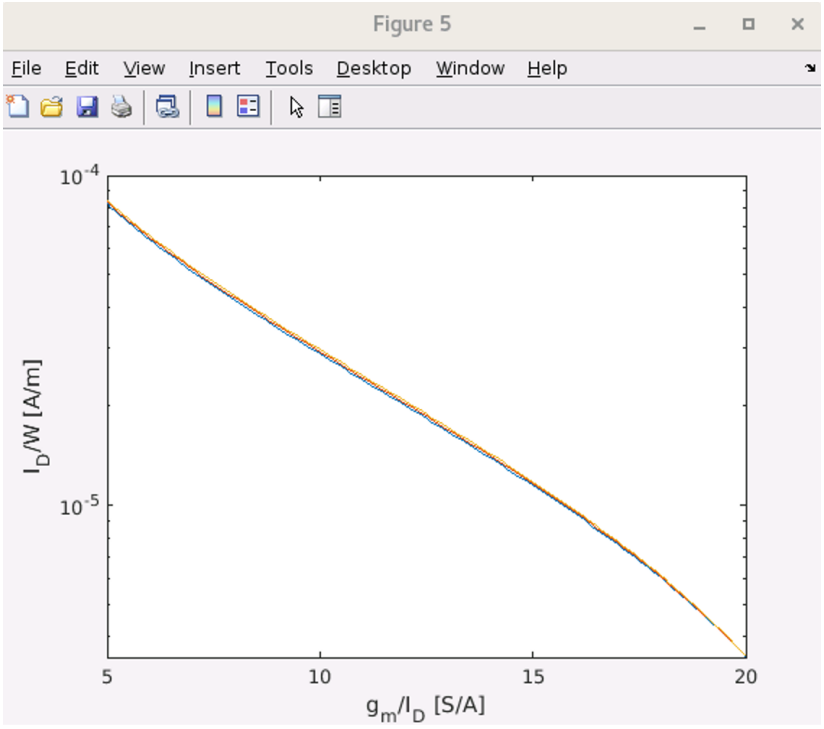


Figure 5. Plot ID/W vs. gm/ID for different VDS (at minimum L)

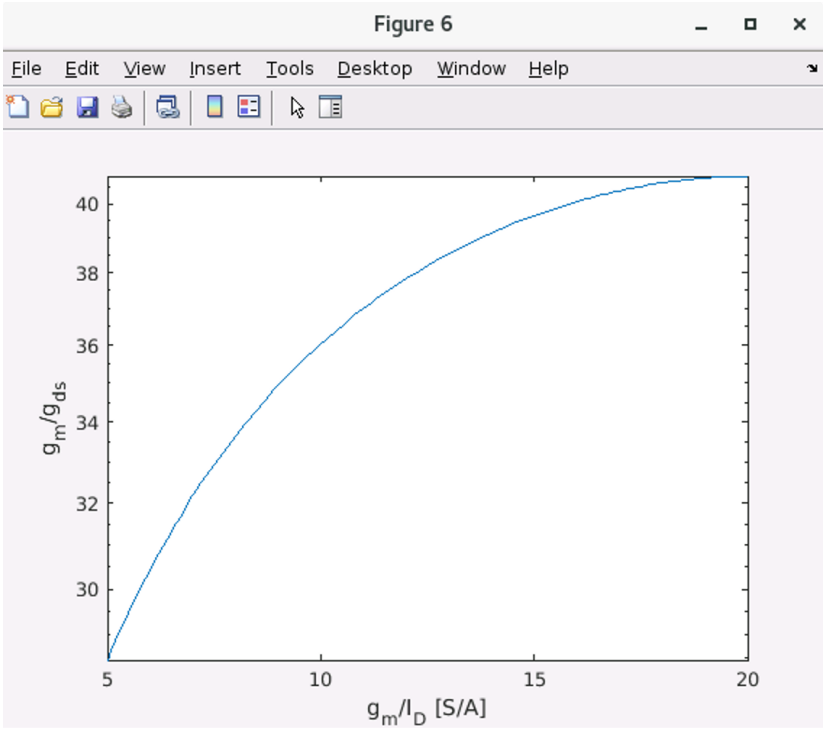


Figure 6. Plot gm/gds vs. gm/ID (at minimum L and default VDS)

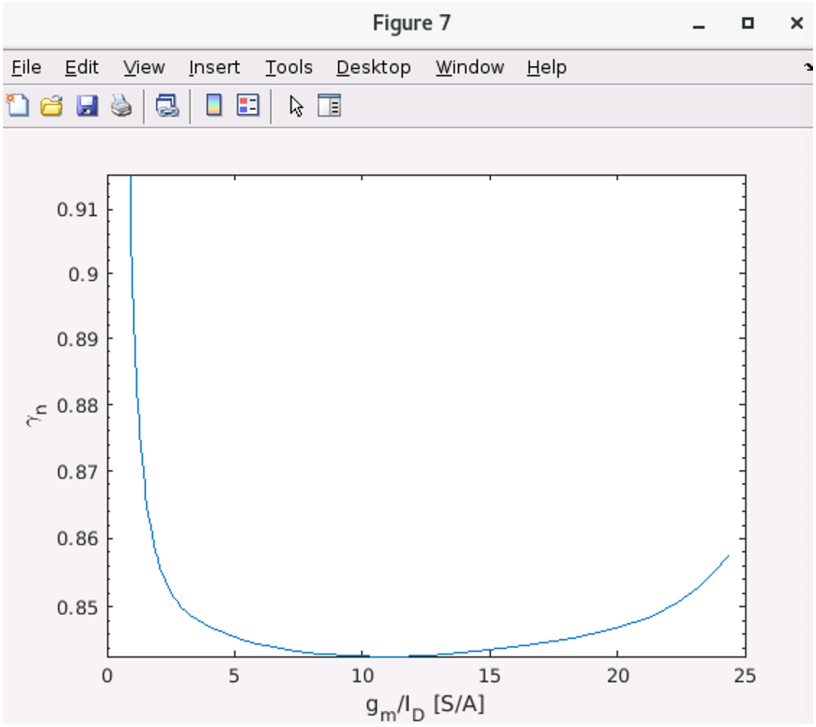


Figure 7. Plot thermal noise factor gamma vs. gm/ID (at minimum L)

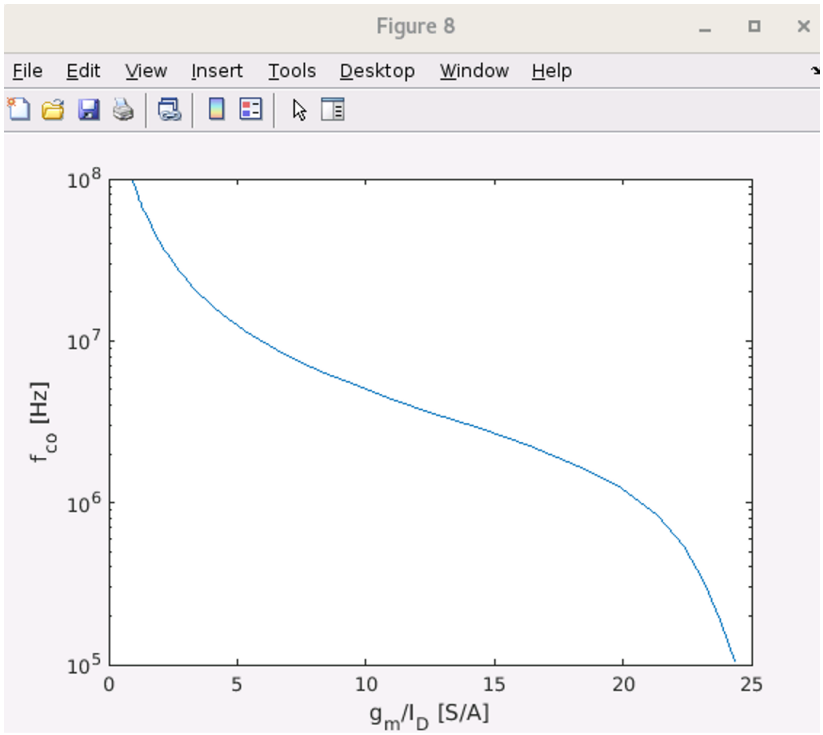


Figure 8. Plot Flicker noise corner frequency fco vs. gm/ID (at minimum L)

pch

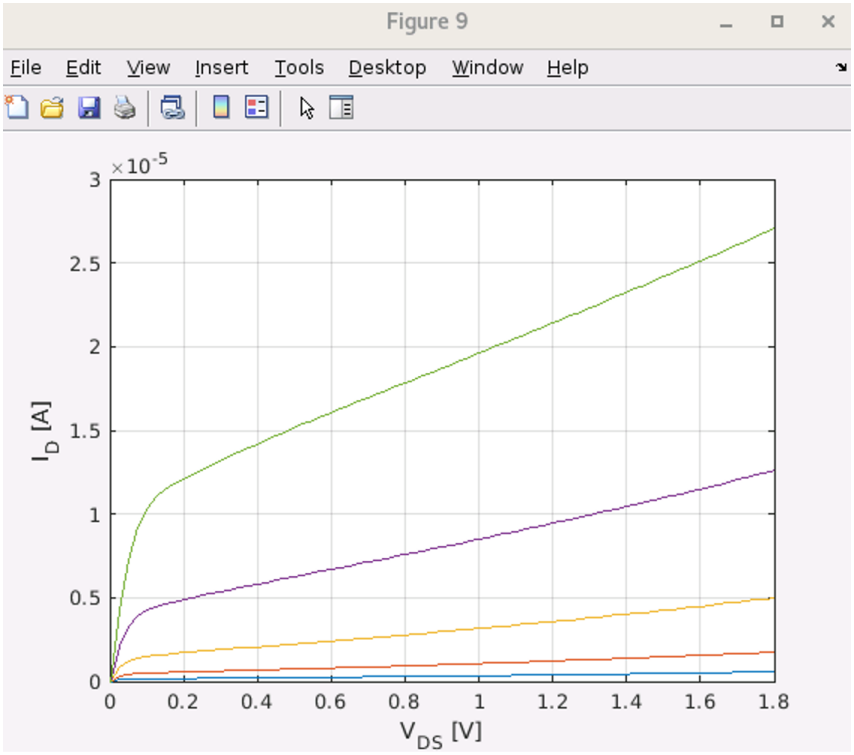


Figure 9. Plot ID vs. VDS

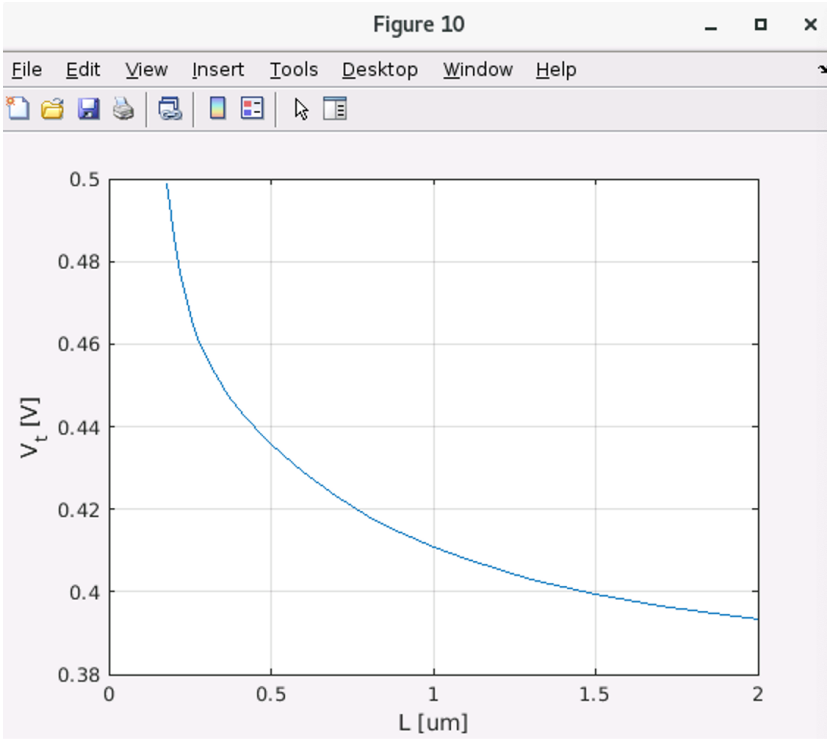


Figure 10. Plot Vt vs. L

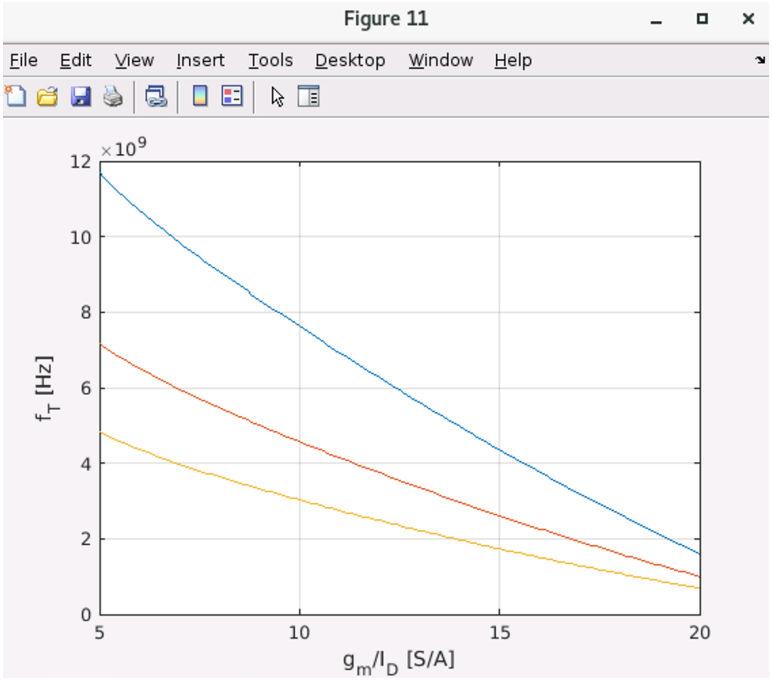


Figure 11. Plot fT vs. gm/ID for different L

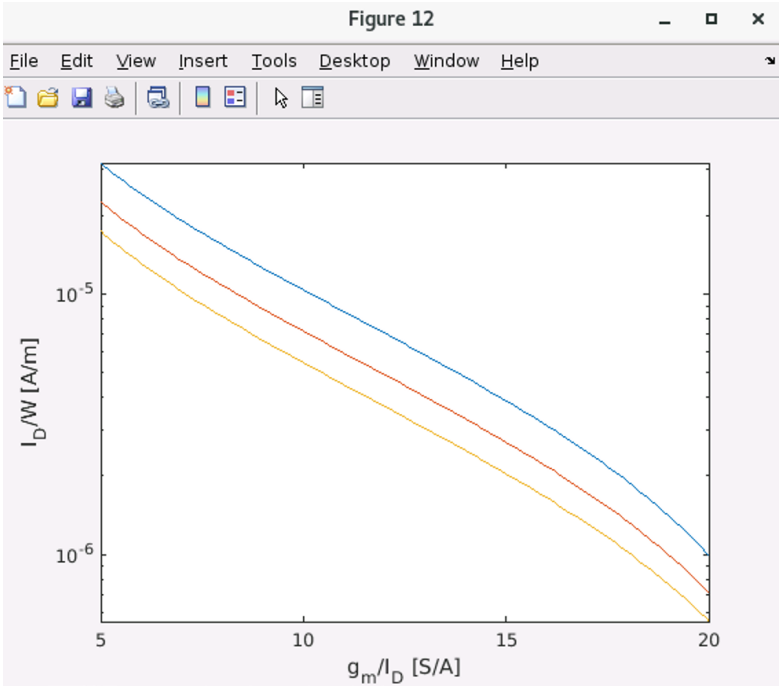


Figure 12. Plot ID/W vs. gm/ID for different L

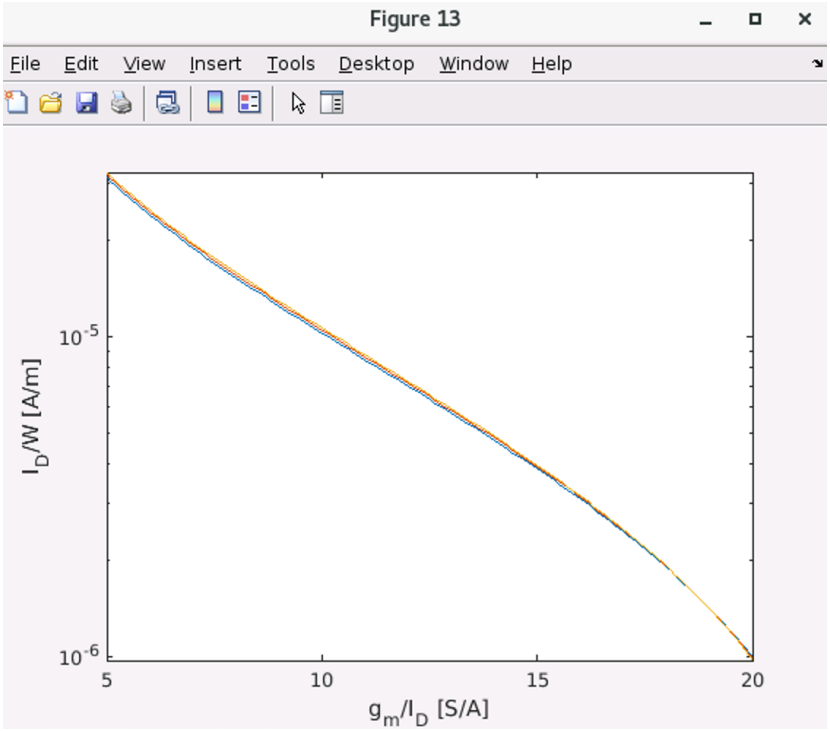


Figure 13. Plot ID/W vs. gm/ID for different VDS (at minimum L)

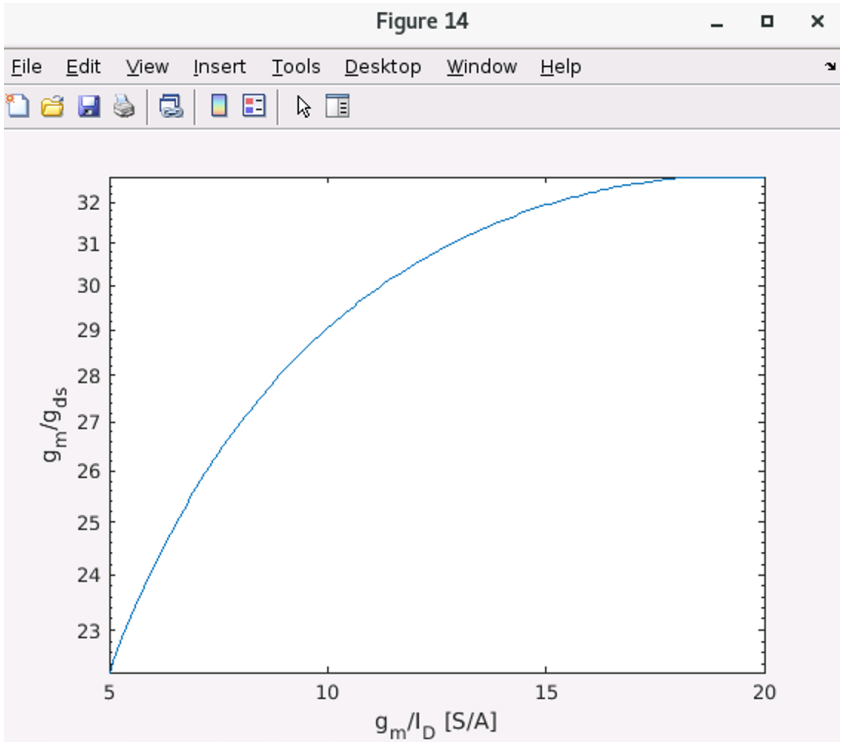


Figure 14. Plot gm/gds vs. gm/ID (at minimum L and default VDS)

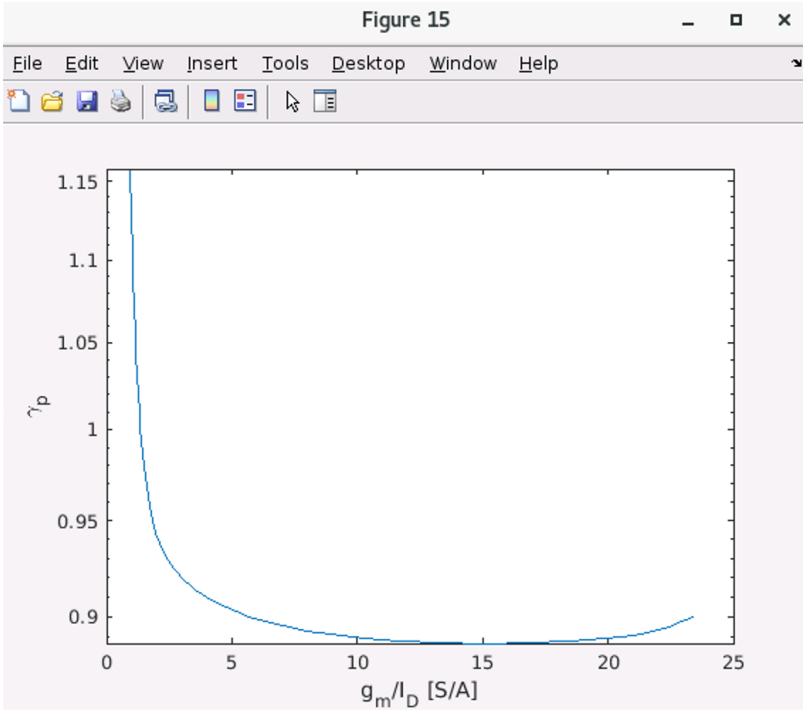


Figure 15. Plot thermal noise factor gamma vs. gm/ID (at minimum L)

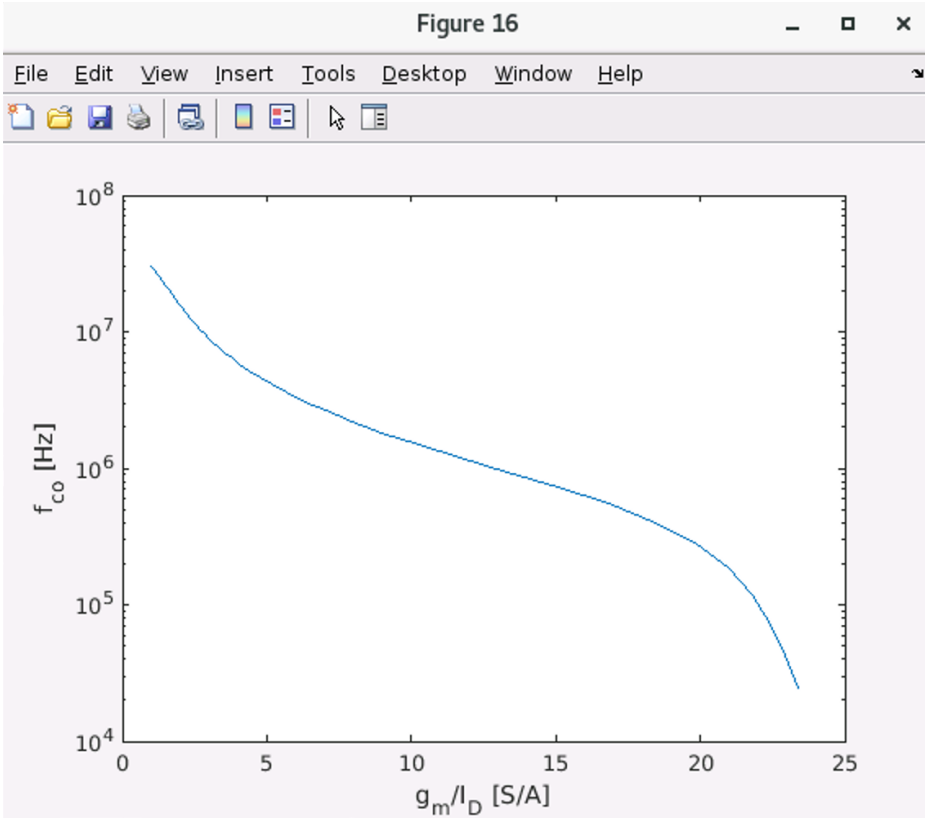


Figure 16. Plot Flicker noise corner frequency fco vs. gm/ID (at minimum L)

test_lookupVGS_eldo.m

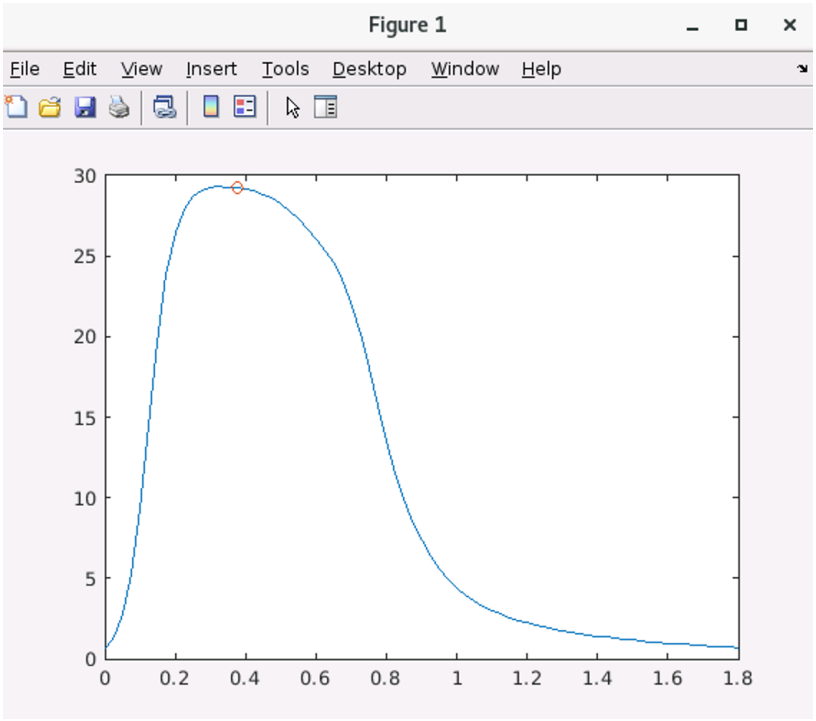
```
% File: test_lookupVGS_eldo.m
% Usage examples for lookup_VGS function

clearvars;
clc;
close all;

addpath('/usr/class/gmidLUTs;/usr/class/gmidTECHs')
addpath('/usr/local/MATLAB/HspiceToolbox')

%addpath ..
load('/usr/class/gmidTECHs/180nch_e.mat')

%% Example with unknown source voltage
gm_ID = 15;
VGB   = .9;
VDB   = .7;

% find VGS and VS
VGS = look_upVGS(nch, 'GM_ID', gm_ID,'VDB',VDB,'VGB',VGB)
VSB  = VGB - VGS  % VS does not change much with VDB

% now check to make sure the same gm/ID comes our from forward
% interpolation
gmID = look_up(nch,'GM_ID','VGS',VGS,'VDS',VDB-VSB,'VSB',VSB)

%% Example with vector input for L

gm_ID = 15;
L = min(nch.L):0.1:0.5;
VGS = look_upVGS(nch, 'GM_ID', gm_ID, 'L', L)

%% Example with vector input for gm/ID

gm_ID = 5:10;
VGS = look_upVGS(nch, 'GM_ID', gm_ID, 'L', 0.25)


%% Example with ID_W as input

gm_ID =10:12;
ID_W = look_up(nch,'ID_W','GM_ID', gm_ID)
VGS = look_upVGS(nch, 'GM_ID', gm_ID)
VGS = look_upVGS(nch, 'ID_W', ID_W)


%% Example with known and positive VSB
% The issue is that gm/ID drops again for VGS near zero and the function
% may then catch the wrong intercept. To fix this, the function looks only
% to the right of the maximum value for gm/ID

VSB = 0.8;
VDS = 0.2;
```

```
gm_ID = look_up(nch,'GM_ID','VGS',nch.VGS,'VSB',VSB, 'VDS', VDS);
gmid = max(gm_ID)-0.05;
VGS = look_upVGS(nch, 'GM_ID', gmid,'VSB',VSB,'VDS',VDS)
display('Plot Figure 1')
plot(nch.VGS, gm_ID, VGS, gmid, 'o')


%% Example with vector input for mode 2 -- L

VSB = 0.8;
VDS = 0.2;
VGS = look_upVGS(nch, 'GM_ID', gmid,'VSB',VSB,'VDS',VDS, 'L',
min(nch.L):0.1:0.2)


%% Example with vector input for mode 2 -- VDS

VSB = 0.8;
VDS = 0.2:0.1:0.5;
VGS = look_upVGS(nch, 'GM_ID', gmid,'VSB',VSB,'VDS',VDS)

%% Example with values exceeding maximum

gm_ID = 50;
VGS = look_upVGS(nch, 'GM_ID', gm_ID, 'L', 0.25)
```

Results obtained running test_lookupVGS_eldo for Stanford's 180nm process

```
VGS =
    0.6600

VSB =
    0.2400

gmID =
   15.0109

VGS =
    0.5927
    0.5887
    0.5793
    0.5684

VGS =
    0.8236
    0.7744
    0.7381
    0.7095
    0.6860
    0.6659

ID_W =
   1.0e-04 *
    0.2898
    0.2420
    0.2022
```

```
VGS =
    0.6669
    0.6497
    0.6341

VGS =
    0.6669
    0.6497
    0.6341

VGS =
    0.3752

Plot Figure 1
```




```
VGS =
    0.3752

VGS =
    0.3752
    0.3737
    0.3717
    0.3698

look_upVGS: GM_ID input larger than maximum!

VGS =
    NaN
```

techsweep_cfg_bsim3_180_eldo.m

```
% File: techsweep_cfg_bsim3_180_eldo.m
% Configuration for techsweep_eldo_run.m
% Boris Murmann
% Stanford University
% September 13, 2017

% Patrick Munar & Claudio Talarico: 2021-07-26
% Gonzaga University
% Modified input netlist to match eldo syntax

function c = techsweep_cfg_bsim3_180_eldo

% Model info and file paths
c.modelfile = '/usr/class/models/ee214_hspice.mod';
c.modelinfo = 'Stanford EE214 models, 180nm CMOS, ELDO, BSIM3v3';
c.corner = 'NOM';
c.temp = 300;
c.modeln = 'nch';
c.modelp = 'pch';
c.savefilen = '180nch_e';
c.savefilep = '180pch_e';
c.simcmd = '/opt/mentor/amsv/amsv/bin/eldo -i techsweep.sp > techsweep.out';
c.outfile = 'techsweep.sw0';
c.outfile_noise = 'techsweep.ac0';

% Path to hspice toolbox (http://www.cppsim.com/download_hspice_tools.html)
addpath('/usr/local/MATLAB/HspiceToolbox')

% Sweep parameters
c.VGS_step = 25e-3;
c.VDS_step = 25e-3;
c.VSB_step = 100e-3;
c.VGS_max = 1.8;
c.VDS_max = 1.8;
c.VSB_max = 1.0;
c.VGS = 0:c.VGS_step:c.VGS_max;
c.VDS = 0:c.VDS_step:c.VDS_max;
c.VSB = 0:c.VSB_step:c.VSB_max;
c.LENGTH = [(0.18:0.02:0.5) (0.6:0.1:2.0)];
c.WIDTH = 5;
c.NFING = 1;

% Variable definitions
c.nvars = {'n_id', 'n_vt', 'n_vdsat', 'n_gm', 'n_gmb', 'n_gds', 'n_cgg', 'n_cgs', 'n_cgd',
'n_cgb', 'n_cdd', 'n_css'};
c.pvars = {'p_id', 'p_vt', 'p_vdsat', 'p_gm', 'p_gmb', 'p_gds', 'p_cgg', 'p_cgs', 'p_cgd',
'p_cgb', 'p_cdd', 'p_css'};
c.outvars = {'ID', 'VT', 'VDSAT', 'GM', 'GMB', 'GDS', 'CGG', 'CGS', 'CGD', 'CGB', 'CDD', 'CSS'};
c.nvars_noise = {'ni_mn', 'nf_mn'};
c.pvars_noise = {'ni_mp', 'nf_mp'};
c.outvars_noise = {'STH', 'SFL'};

% Simulation netlist
netlist = sprintf([...
'techsweep.sp \n'...
'.notrc \n'...
'.option COMPAT \n'...
'.option POST_VERSION=9601 POST=1 \n'...
'.option NOMOD NOASCII \n'...
'\n'...
'.inc %s \n'...
'.inc techsweep_params.sp \n'...
'.temp %d \n'...
'\n'...
'vnoi     vx  0         dc 0  ac 1 \n'...
'vdsn     vdn vx        dc ''ds''  \n'...
'vgsn     vgn 0         dc ''gs''  \n'...
'vbsn     vbn 0         dc ''-sb'' \n'...
'vdsp     vdp vx        dc ''-ds'' \n'...
'vgsp     vgp 0         dc ''-gs'' \n'...
```

```
'vbsp     vbp 0         dc ''sb''  \n'...
'h1       vn  0        ccvs  vnoi  1 \n'...
'r1       vn  0        100T nonoise \n'...
'mn       vdn vgn 0 vbn %s  L=''length*1e-6'' W=%d \n'...
'mp       vdp vgp 0 vbp %s  L=''length*1e-6'' W=%d \n'...
'\n'...
'***.dc gs 0 %d %d ds 0 %d %d \n'...
'.dc sweep data=data1 \n'...
'.ac lin 1 1 1 sweep data=data1 \n'...
'.noise v(vn) vnoi 80\n'...
'\n'...
'.probe dc n_id    = par(''id(mn)'') \n'...
'.probe dc n_vt    = par(''vth(mn)'') \n'...
'.probe dc n_vdsat = par(''vdsat(mn)'') \n'...
'.probe dc n_gm    = par(''gm(mn)'')   \n'...
'.probe dc n_gmb   = par(''gmbs(mn)'') \n'...
'.probe dc n_gds   = par(''gds(mn)'')  \n'...
'.probe dc n_cgg   = par(''cgg(mn)'') \n'...
'.probe dc n_cgs   = par(''-cgs(mn)'') \n'...
'.probe dc n_cgd   = par(''-cgd(mn)'') \n'...
'.probe dc n_cgb   = par(''-cbg(mn) '') \n'...
'.probe dc n_cdd   = par(''cdd(mn)'') \n'...
'.probe dc n_css   = par(''css(mn)'') \n'...
'\n'...
'.probe ac ni_mn   = par(''thnoise(mn)'') \n'...
'.probe ac nf_mn   = par(''flknoise(mn)'') \n'...
'\n'...
'.probe dc p_id    = par(''-id(mp)'') \n'...
'.probe dc p_vt    = par(''-vth(mp)'') \n'...
'.probe dc p_vdsat = par(''-vdsat(mp)'') \n'...
'.probe dc p_gm    = par(''gm(mp)'')   \n'...
'.probe dc p_gmb   = par(''gmbs(mp)'') \n'...
'.probe dc p_gds   = par(''gds(mp)'')  \n'...
'.probe dc p_cgg   = par(''cgg(mp)'') \n'...
'.probe dc p_cgs   = par(''-cgs(mp)'') \n'...
'.probe dc p_cgd   = par(''-cgd(mp)'') \n'...
'.probe dc p_cgb   = par(''-cbg(mp) '') \n'...
'.probe dc p_cdd   = par(''cdd(mp)'') \n'...
'.probe dc p_css   = par(''css(mp)'') \n'...
'\n'...
'.probe ac ni_mp   = par(''thnoise(mp)'') \n'...
'.probe ac nf_mp   = par(''flknoise(mp)'') \n'...
'.end \n'...
], c.modelfile, c.temp-273, ...
c.modeln, c.WIDTH*1e-6, ...
c.modelp, c.WIDTH*1e-6, ...
c.VGS_max, c.VGS_step, ...
c.VDS_max, c.VDS_step);

% Write netlist
fid = fopen('techsweep.sp', 'w');
fprintf(fid, netlist);
fclose(fid);

return
```

techsweep_eldo_run.m

```
% Matlab script for technology characterization
% Boris Murmann
% Stanford University
% September 12, 2017

% Claudio Talarico: 2020-06-27
% uncomment the configuration file corresponding
% to the technolgy to be characterized

clearvars;
close all;

% Load configuration
c = techsweep_cfg_bsim3_180_eldo;
% c = techsweep_cfg_bsim3_tsmc180_hsp;
% c = techsweep_cfg_bsim4_45_hsp;
% c = techsweep_cfg_bsim4_50_hsp;
% c = techsweep_cfg_bsim4_16_hsp;
% c = techsweep_cfg_bsim3_600_hsp;

% Simulation loop
for i = 1:length(c.LENGTH)
    str=sprintf('L = %2.2f', c.LENGTH(i));
    disp(str);
    tic
    for j = 1:length(c.VSB)
        % Write simulation parameters
        fid=fopen('techsweep_params.sp', 'w');
        fprintf(fid,'.param length = %d\n', c.LENGTH(i));
        fprintf(fid,'.param sb = %d\n', c.VSB(j));
        fprintf(fid,'.data data1 \n');
        fprintf(fid,'+ gs ds \n');
        for m=1:length(c.VDS)
            for n=1:length(c.VGS)
                fprintf(fid,'+ %d %d \n', c.VGS(n), c.VDS(m));
            end
        end
        fprintf(fid,'.enddata \n');
        fclose(fid);

        % Run simulator
        [status,result] = system(c.simcmd);
        if(status)
            disp('Simulation did not run properly. Check techsweep.out.')
            return;
        end

        %Read and store results
        h = loadsig(c.outfile);
        % 2021-07-26 (change the format of the ELDO results)
        % added lines denoted CT. 20210726
        % commented out original lines denoted with %%%
        for n = 1: length(c.outvars)
            values_vec = evalsig(h, c.nvars{n});          % CT. 20210726
            values = vec2mat(values_vec, length(c.VDS))'; % CT. 20210726
            %%% nch.(c.outvars{n})(i,:,:,j)  = evalsig(h, c.nvars{n}); % original line
            nch.(c.outvars{n})(i,:,:,j)  = values;        % CT. 20210726
            values_vec = evalsig(h, c.pvars{n});          % CT. 20210726
            values = vec2mat(values_vec, length(c.VDS))'; % CT. 20210726
            %%% pch.(c.outvars{n})(i,:,:,j)  = evalsig(h, c.pvars{n}); % original line
            pch.(c.outvars{n})(i,:,:,j)  = values;        % CT. 20210726
        end

        h = loadsig(c.outfile_noise);
        for n = 1: length(c.outvars_noise)
            values_vec = evalsig(h, c.nvars_noise{n});
            values = vec2mat(values_vec, length(c.VDS))';
            nch.(c.outvars_noise{n})(i,:,:,j)  = values;
```

```
            values_vec = evalsig(h, c.pvars_noise{n});
            values = vec2mat(values_vec, length(c.VDS))';
            pch.(c.outvars_noise{n})(i,:,:,j)  = values;
        end

    end
    toc
end
% Include sweep info
nch.INFO   = c.modelinfo;
nch.CORNER = c.corner;
nch.TEMP   = c.temp;
nch.VGS    = c.VGS';
nch.VDS    = c.VDS';
nch.VSB    = c.VSB';
nch.L      = c.LENGTH';
nch.W      = c.WIDTH;
nch.NFING  = c.NFING;
pch.INFO   = c.modelinfo;
pch.CORNER = c.corner;
pch.TEMP   = c.temp;
pch.VGS    = c.VGS';
pch.VDS    = c.VDS';
pch.VSB    = c.VSB';
pch.L      = c.LENGTH';
pch.W      = c.WIDTH;
pch.NFING  = c.NFING;

save(c.savefilen, 'nch');
save(c.savefilep, 'pch');
```

techsweep_eldo_debug.m

```
% File: techsweep_eldo_debug.m
% Matlab script for technology characterization
% Debug version (one ELDO run + display)
% Boris Murmann
% Stanford University
% September 12, 2017

% Claudio Talarico: 2020-06-27
% uncomment the configuration file corresponding
% to the technolgy to be characterized

% Patrick Munar and C. Talarico : 2021-07-26
% ELDO DC sweep data format is different than HSPICE DC sweep format
% modified script accordingly

clearvars;
close all;

tic
% Load configuration
c = techsweep_cfg_bsim3_180_eldo;

% Write simulation parameters
fid=fopen('techsweep_params.sp', 'w');
fprintf(fid,'.param length = %d\n', c.LENGTH(1));
fprintf(fid,'.param sb = %d\n', c.VSB(1));
fprintf(fid,'.data data1 \n');
fprintf(fid,'+ gs ds \n');
for i=1:length(c.VDS)
    for j=1:length(c.VGS)
        fprintf(fid,'+ %d %d \n', c.VGS(j), c.VDS(i));
    end
end
fprintf(fid,'.enddata \n');
fclose(fid);

% Run simulator
[status,result] = system(c.simcmd);
if(status)
    disp('Simulation did not run properly. Check techsweep.out.')
    return;
end

% Read and display results
h = loadsig(c.outfile);
lssig(h)
hn = loadsig(c.outfile_noise);
lssig(hn)


% Display data for middle of sweep
idx1 = round(length(c.VGS)/2)
idx2 = round(length(c.VDS)/2)
s = sprintf('Data for VGS = %d, VDS = %d, VSB = %d, L = %d', ...
    c.VGS(idx1), c.VDS(idx2), c.VSB(1), c.LENGTH(1));

% Read and display raw parameters and created output
for k = 1:length(c.nvars)
    % values = evalsig(h, c.nvars{k}); % original line (commented out)
    values_vec = evalsig(h, c.nvars{k});          % CT. 20210726 (added)
    values = vec2mat(values_vec, length(c.VDS))'; % CT. 20210726 (added)
    s = sprintf('%s = %d', c.nvars{k}, values(idx1, idx2));
    disp(s);
    nch.(c.outvars{k}) = values(idx1, idx2);
end

for k = 1:length(c.nvars_noise)
    values_vec = evalsig(hn, c.nvars_noise{k});
    values = vec2mat(values_vec, length(c.VDS))';
    s = sprintf('%s = %d', c.nvars_noise{k}, values(idx1, idx2));
```

```
    disp(s);
    nch.(c.outvars_noise{k}) = values(idx1, idx2);
end

disp(nch);

for k = 1:length(c.pvars)
    % values = evalsig(h, c.pvars{k}); % original line (CT. commented out)
    values_vec = evalsig(h, c.pvars{k});          % CT. 20210726 (added)
    values = vec2mat(values_vec, length(c.VDS))'; % CT. 20210726 (added)
    s = sprintf('%s = %d', c.pvars{k}, values(idx1, idx2));
    disp(s);
    pch.(c.outvars{k}) = values(idx1, idx2);
end

for k = 1:length(c.pvars_noise)
    values_vec = evalsig(hn, c.pvars_noise{k});
    values = vec2mat(values_vec, length(c.VDS))';
    s = sprintf('%s = %d', c.pvars_noise{k}, values(idx1, idx2));
    disp(s);
    pch.(c.outvars_noise{k}) = values(idx1, idx2);
end

disp(pch);
toc

% CT. 20210726 (ELDO DC sweep data format is different than HSPICE)
% check conversion from vector to array
% vgn = evalsig(h, 'V_vgn');
% vgn_vec = evalsig(h, 'sw_ds');
% vgn1 = vec2mat(vgn_vec, length(c.VDS))';
% find(vgn-vgn1)

% gm/id Plot for NMOS
gm = evalsig(h, 'n_gm');
id = evalsig(h, 'n_id');
figure;
plot(gm./id);

% gm/id Plot for PMOS
gm = evalsig(h, 'p_gm');
id = evalsig(h, 'p_id');
figure;
plot(gm./id)
```

File: techsweep.sp

```
techsweep.sp
.notrc
.option COMPAT
.option POST_VERSION=9601 POST=1
.option NOMOD NOASCII

.inc /usr/class/models/ee214_hspice.mod
.inc techsweep_params.sp
.temp 27

vnoi     vx  0         dc 0  ac 1
vdsn     vdn vx        dc 'ds'
vgsn     vgn 0         dc 'gs'
vbsn     vbn 0         dc '-sb'
vdsp     vdp vx        dc '-ds'
vgsp     vgp 0         dc '-gs'
vbsp     vbp 0         dc 'sb'
h1       vn  0        ccvs  vnoi  1
r1       vn  0        100T nonoise
mn       vdn vgn 0 vbn nch  L='length*1e-6' W=5.000000e-06
mp       vdp vgp 0 vbp pch  L='length*1e-6' W=5.000000e-06

***.dc gs 0 1.800000e+00 2.500000e-02 ds 0 1.800000e+00 2.500000e-02
.dc sweep data=data1
.ac lin 1 1 1 sweep data=data1
.noise v(vn) vnoi 80

.probe dc n_id    = par('id(mn)')
.probe dc n_vt    = par('vth(mn)')
.probe dc n_vdsat = par('vdsat(mn)')
.probe dc n_gm    = par('gm(mn)')
.probe dc n_gmb   = par('gmbs(mn)')
.probe dc n_gds   = par('gds(mn)')
.probe dc n_cgg   = par('cgg(mn)')
.probe dc n_cgs   = par('-cgs(mn)')
.probe dc n_cgd   = par('-cgd(mn)')
.probe dc n_cgb   = par('-cbg(mn) ')
.probe dc n_cdd   = par('cdd(mn)')
.probe dc n_css   = par('css(mn)')

.probe ac ni_mn   = par('thnoise(mn)')
.probe ac nf_mn   = par('flknoise(mn)')

.probe dc p_id    = par('-id(mp)')
.probe dc p_vt    = par('-vth(mp)')
.probe dc p_vdsat = par('-vdsat(mp)')
.probe dc p_gm    = par('gm(mp)')
.probe dc p_gmb   = par('gmbs(mp)')
.probe dc p_gds   = par('gds(mp)')
.probe dc p_cgg   = par('cgg(mp)')
.probe dc p_cgs   = par('-cgs(mp)')
.probe dc p_cgd   = par('-cgd(mp)')
.probe dc p_cgb   = par('-cbg(mp) ')
.probe dc p_cdd   = par('cdd(mp)')
.probe dc p_css   = par('css(mp)')

.probe ac ni_mp   = par('thnoise(mp)')
.probe ac nf_mp   = par('flknoise(mp)')
.end
```

```
.param length = 2
.param sb = 1
.data data1
+ gs ds
+ 0 0
+ 2.500000e-02 0
+ 5.000000e-02 0
+ 7.500000e-02 0
+ 1.000000e-01 0
+ 1.250000e-01 0
+ 1.500000e-01 0
+ 1.750000e-01 0
+ 2.000000e-01 0
+ 2.250000e-01 0
+ 2.500000e-01 0
+ 2.750000e-01 0
+ 3.000000e-01 0
+ 3.250000e-01 0
+ 3.500000e-01 0
+ 3.750000e-01 0
+ 4.000000e-01 0
+ 4.250000e-01 0
+ 4.500000e-01 0
+ 4.750000e-01 0
+ 5.000000e-01 0
+ 5.250000e-01 0
+ 5.500000e-01 0
+ 5.750000e-01 0
+ 6.000000e-01 0
+ 6.250000e-01 0
+ 6.500000e-01 0
+ 6.750000e-01 0
+ 7.000000e-01 0
+ 7.250000e-01 0
+ 7.500000e-01 0
+ 7.750000e-01 0
+ 8.000000e-01 0
+ 8.250000e-01 0
+ 8.500000e-01 0
+ 8.750000e-01 0
+ 9.000000e-01 0
+ 9.250000e-01 0
+ 9.500000e-01 0
+ 9.750000e-01 0
+ 1 0
+ 1.025000e+00 0
+ 1.050000e+00 0
+ 1.075000e+00 0
+ 1.100000e+00 0
+ 1.125000e+00 0
+ 1.150000e+00 0
+ 1.175000e+00 0
+ 1.200000e+00 0
+ 1.225000e+00 0
+ 1.250000e+00 0
+ 1.275000e+00 0
+ 1.300000e+00 0
+ 1.325000e+00 0
+ 1.350000e+00 0
+ 1.375000e+00 0
+ 1.400000e+00 0
+ 1.425000e+00 0
+ 1.450000e+00 0
+ 1.475000e+00 0
+ 1.500000e+00 0
+ 1.525000e+00 0
+ 1.550000e+00 0
+ 1.575000e+00 0
+ 1.600000e+00 0
+ 1.625000e+00 0
+ 1.650000e+00 0
+ 1.675000e+00 0
+ 1.700000e+00 0
+ 1.725000e+00 0
+ 1.750000e+00 0
+ 1.775000e+00 0
+ 1.800000e+00 0
+ 0 2.500000e-02
+ 2.500000e-02 2.500000e-02
```

```
+ 5.000000e-02 2.500000e-02
+ 7.500000e-02 2.500000e-02
+ 1.000000e-01 2.500000e-02
+ 1.250000e-01 2.500000e-02
+ 1.500000e-01 2.500000e-02
+ 1.750000e-01 2.500000e-02
+ 2.000000e-01 2.500000e-02
+ 2.250000e-01 2.500000e-02
+ 2.500000e-01 2.500000e-02
+ 2.750000e-01 2.500000e-02
+ 3.000000e-01 2.500000e-02
+ 3.250000e-01 2.500000e-02
+ 3.500000e-01 2.500000e-02
+ 3.750000e-01 2.500000e-02
+ 4.000000e-01 2.500000e-02
+ 4.250000e-01 2.500000e-02
+ 4.500000e-01 2.500000e-02
+ 4.750000e-01 2.500000e-02
+ 5.000000e-01 2.500000e-02
+ 5.250000e-01 2.500000e-02
+ 5.500000e-01 2.500000e-02
+ 5.750000e-01 2.500000e-02
+ 6.000000e-01 2.500000e-02
+ 6.250000e-01 2.500000e-02
+ 6.500000e-01 2.500000e-02
+ 6.750000e-01 2.500000e-02
+ 7.000000e-01 2.500000e-02
+ 7.250000e-01 2.500000e-02
+ 7.500000e-01 2.500000e-02
+ 7.750000e-01 2.500000e-02
+ 8.000000e-01 2.500000e-02
+ 8.250000e-01 2.500000e-02
+ 8.500000e-01 2.500000e-02
+ 8.750000e-01 2.500000e-02
+ 9.000000e-01 2.500000e-02
+ 9.250000e-01 2.500000e-02
+ 9.500000e-01 2.500000e-02
+ 9.750000e-01 2.500000e-02
+ 1 2.500000e-02
+ 1.025000e+00 2.500000e-02
+ 1.050000e+00 2.500000e-02
+ 1.075000e+00 2.500000e-02
+ 1.100000e+00 2.500000e-02
+ 1.125000e+00 2.500000e-02
+ 1.150000e+00 2.500000e-02
+ 1.175000e+00 2.500000e-02
+ 1.200000e+00 2.500000e-02
+ 1.225000e+00 2.500000e-02
+ 1.250000e+00 2.500000e-02
+ 1.275000e+00 2.500000e-02
+ 1.300000e+00 2.500000e-02
+ 1.325000e+00 2.500000e-02
+ 1.350000e+00 2.500000e-02
+ 1.375000e+00 2.500000e-02
+ 1.400000e+00 2.500000e-02
+ 1.425000e+00 2.500000e-02
+ 1.450000e+00 2.500000e-02
+ 1.475000e+00 2.500000e-02
+ 1.500000e+00 2.500000e-02
+ 1.525000e+00 2.500000e-02
+ 1.550000e+00 2.500000e-02
+ 1.575000e+00 2.500000e-02
+ 1.600000e+00 2.500000e-02
+ 1.625000e+00 2.500000e-02
+ 1.650000e+00 2.500000e-02
+ 1.675000e+00 2.500000e-02
+ 1.700000e+00 2.500000e-02
+ 1.725000e+00 2.500000e-02
+ 1.750000e+00 2.500000e-02
+ 1.775000e+00 2.500000e-02
+ 1.800000e+00 2.500000e-02
+ 0 5.000000e-02
+ 2.500000e-02 5.000000e-02
+ 5.000000e-02 5.000000e-02
+ 7.500000e-02 5.000000e-02
+ 1.000000e-01 5.000000e-02
+ 1.250000e-01 5.000000e-02
+ 1.500000e-01 5.000000e-02
+ 1.750000e-01 5.000000e-02
```

• • •

gmid_lut_180.m

```
% File: gmid_lut_180.m
% analyzing the structure of the data of nch and pch
clearvars;
clear all;
close all;
clc;

addpath('/usr/class/gmidLUTs;/usr/class/gmidTECHs')
addpath('/usr/local/MATLAB/HspiceToolbox')

load('180nch_e.mat')
display(nch)
display('vtn1=')
display(nch.VT(1,1,1,1))

load('180pch_e.mat')
display(pch)
display('vtp1=')
display(pch.VT(1,1,1,1))
```

```
nch =

  struct with fields:

        ID: [32×73×73×11 double]
        VT: [32×73×73×11 double]
     VDSAT: [32×73×73×11 double]
        GM: [32×73×73×11 double]
       GMB: [32×73×73×11 double]
       GDS: [32×73×73×11 double]
       CGG: [32×73×73×11 double]
       CGS: [32×73×73×11 double]
       CGD: [32×73×73×11 double]
       CGB: [32×73×73×11 double]
       CDD: [32×73×73×11 double]
       CSS: [32×73×73×11 double]
       STH: [32×73×73×11 double]
       SFL: [32×73×73×11 double]
      INFO: 'Stanford EE214 models, 180nm CMOS, ELDO, BSIM3v3'
    CORNER: 'NOM'
      TEMP: 300
       VGS: [73×1 double]
       VDS: [73×1 double]
       VSB: [11×1 double]
         L: [32×1 double]
         W: 5
     NFING: 1

vtn1=
    0.4922
```

```
pch =

  struct with fields:

        ID: [32×73×73×11 double]
        VT: [32×73×73×11 double]
     VDSAT: [32×73×73×11 double]
        GM: [32×73×73×11 double]
       GMB: [32×73×73×11 double]
       GDS: [32×73×73×11 double]
       CGG: [32×73×73×11 double]
       CGS: [32×73×73×11 double]
       CGD: [32×73×73×11 double]
       CGB: [32×73×73×11 double]
       CDD: [32×73×73×11 double]
       CSS: [32×73×73×11 double]
       STH: [32×73×73×11 double]
       SFL: [32×73×73×11 double]
      INFO: 'Stanford EE214 models, 180nm CMOS, ELDO, BSIM3v3'
    CORNER: 'NOM'
      TEMP: 300
       VGS: [73×1 double]
       VDS: [73×1 double]
       VSB: [11×1 double]
         L: [32×1 double]
         W: 5
     NFING: 1

vtp1=
    0.5115
```