Massachusetts Institute of Technology

Department of Electrical Engineering and Computer Science

Differential Amplifier Script User Manual

Ethan Crain and Michael Perrott

http://www-mtl.mit.edu/research/perrottgroup/tools.html

November 30, 2004

---

1 INTRODUCTION

A simple numerical procedure has been implemented as a Matlab script to allow straightforward design of high speed, resistor loaded, differential amplifiers in modern CMOS processes. The amplifier, shown in Figure 1, exhibits device characteristics that dramatically depart from traditional square law characterisitics making designs based on hand calculations grossly inaccurate. The analytical form of the procedure allows for an intuitive perspective of the varying gain-bandwidth product for such amplifiers.

2 QUICK START

To begin using the amplifier scripts, simply download the file *diffamp.tar.gz* from http://www-mtl.mit.edu/research/perrottgroup/tools.html to a local directory and expand it by typing:

```
gunzip < diffamp.tar.gz | tar xvf -
```

Add the location of the script to your path by typing the following line in Matlab:

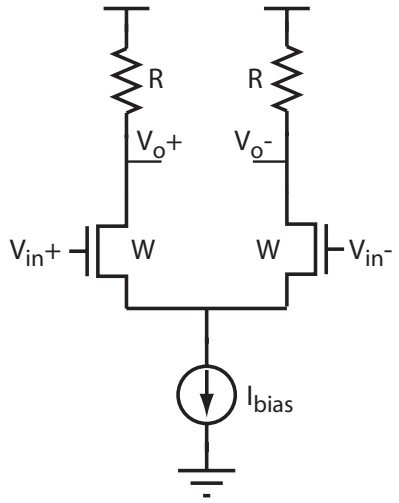

Figure 1: High-Speed Resistor Loaded Differential Amplifier

addpath('<path>/diffamp');

After completing this step, the scripts are ready to use. The syntax for calling the scripts is:

1. diffampPWR : $[W, R, BW] = diffampPWR(vsw, av, ibias, scale, cfix, corner)$;
2. diffampBW : $[W, R, Ibias] = diffampBW(vsw, av, bw, scale, cfix, corner)$;

For further help on using the scripts, refer to Sections 3, 4 and 5.

3 BACKGROUND

CMOS analog design techniques have traditionally assumed square law characteritics for device I-V curves when calculating the impact of device properties on circuit performance. However, the square law assumption is quickly becoming highly inaccurate with the introduction of finer line width processes due to nonideal effects such as velocity saturation. As a result, the accuracy of traditional design equations is steadily degrading, and analog designers are in need of alternate approaches to such formulations.

Thus far, there have been two responses to dealing with changing device characteristics in the analog design community. The first has been to assume square law I-V characteristics in calculations, and then utilize a simulator such as SPICE to tweak final device parameter

adjustments. Unfortunately, the square law is rapidly becoming inaccurate to the point that the analytical calculations are practically useless — all design time is then spent iterating SPICE simulations. Such an approach removes intuition from the designer's grasp, leads to a lengthy design process (since many tweaks are required), and often leads to sub-optimal performance. The second approach is to completely automate the analog design process — the user simply specifies performance specifications and some possible topologies, and customized software takes care of the rest. Unfortunately, while very useful for the design of standard analog blocks, such an approach removes creativity from the designer's grasp and offers little intuition for the creation of new circuit topologies.

We propose an alternate approach to this issue — develop numerical procedures for designing specific classes of circuits which resemble hand analysis, but use simulated device characteristics in place of analytical expressions. By sticking with procedures similar to hand analysis, much intution can be gained about design tradeoffs. By using simulated device characteristics, the results are made accurate so that little or no tweaking is required in SPICE.

The Differential Amplifier script applies the above philosophy, in the form of a Matlab script, to the design of high speed, resistor-loaded, differential amplifiers. These structures are tremendously useful in circuit applications whose speed requirements exceed the abilities of full-swing logic circuits. The script is very easy to use and provides the designer of high-speed circuits to accurately determine device circuit parameters for a specific set of performance parameters in a fraction of time of current design methodologies. The amplifier is assumed to be loaded by an identical amplifier stage scaled by α, where α is a user defined ratio of the loading stage to the driving stage, and a fixed interconnect parasitic capacitance, C_{fix}, as shown in Figure 2. Given a set of design constraints, gain, output voltage swing and either power dissipation or desired bandwidth, the script will return the required transistor width, load resistance and either resulting bandwidth, in the former case, or bias current, in the later case. The following section describes the steps required to setup the script.

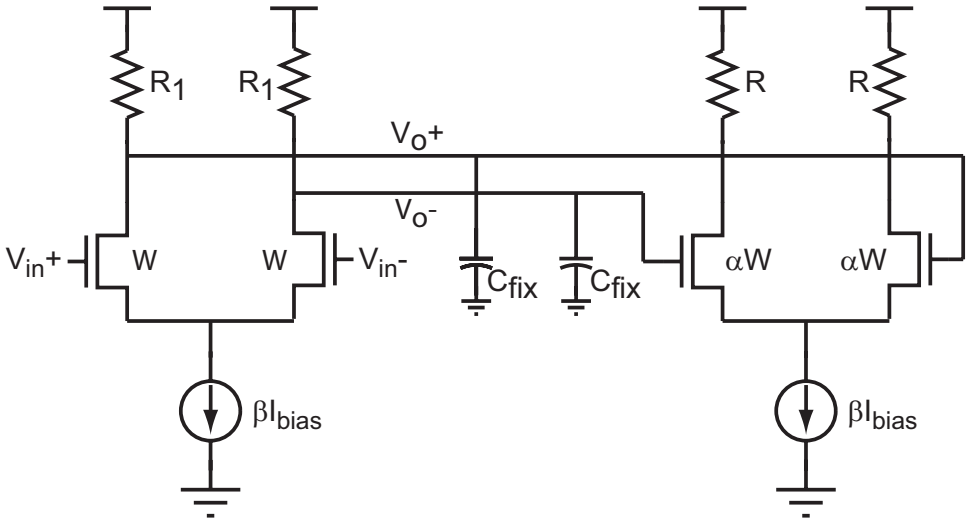

Figure 2: Configuration of Differential Amplifier Showing Loading

4 SETUP

To begin using the amplifier script, simply download the file *diffamp.tar.gz* from http://www-mtl.mit.edu/research/perrottgroup/tools.html to a local directory and expand it by typing:

```
gunzip < diffamp.tar.gz | tar xvf -
```

(Note: this command is one line). This will create a folder called **./diffamp** that contains two separate scripts, a SPICE netlist (found in **./diffamp/spicefiles**), SPICE model files (found in **./diffamp/spicefiles/models** - see note below), and the accompanying documentation that can be found in **./diffamp/Doc**. Add the location of the script to your path by typing the following line in Matlab:

```
addpath('<path>/diffamp');
```

The first script, called *diffampPWR.m*, calculates the transistor width and load resistance required to meet a set of gain, swing and power specifications. The second script, named *diffampBW.m*, returns the width, resistance and bias current required to satisfy a set of desired gain, swing and bandwidth specifications. Both scripts require that the HspiceToolbox, which can be downloaded at the link given above, is installed. Unix users need to compile the *loadsig.c* file that is found in **./diffamp/HspiceToolbox** in their HspiceToolbox folder

(original one). PC users need to replace the loadsig.dll file in their HspiceToolbox folder with the copy found in **./diffamp/HspiceToolbox**. Please note that this version of *loadsig* is different from the one downloaded with the full HspiceToolbox. Keep the old version in case this version causes problems.

Due to licensing issues, proprietary process information can not be distributed with this toolbox. Therefore, the characterization files for a specific process will have to be generated from process models. On a positive note, since the end user generates the process characterization files, any CMOS process can be used. For illustrative purposes, the BSIM3 V3.1 (SPICE 3f5 Level 8) models for the MOSIS (TSMC) T47R 0.18μm process have been included in this package. These models are non-proprietary and are freely available from the MOSIS website (www.mosis.org - Getting Started/Select a Process/TSMC/CL018/Test Results for TSMC_018 runs/T47R_MM_NON_EPI). Please note that all models files in **./diffamp/spicefiles/models** are *TYPICAL* despite the labeling. Slow and fast models are not freely available. To generate the required files for a different process, update the process and model path information in the header of the included SPICE netlist, *test.sp* in **./diffamp/spicefiles**, and run Hspice. The independent variable for both the ac and dc simulations is device width. Adjust the minimum, maximum and step values appropriatly for the process to be used. Also, for more accurate results, adjust the ibias parameter to a value at or near the expected bias current for the design (note that this value corresponds to half the bias current of the differential amplifier). The specific parameters to be extracted are $g_m, g_{d0}, C_{gs}, C_{gd}, C_{gb}$ and C_{db}. By default, Hspice uses an alter statement to generate both the dc and ac characterization files for 9 corners in **./diffamp/spicefiles**. The combinations of process corners (slow, typical, fast) and temperature (-40°C, 25°C, 85°C) can be adjusted to meet individual needs.

After generating the Hspice files, all process variables need to be adjusted in the header of both Matlab amplifier scripts. The description to the right of each parameter is fairly self-explanatory, but a full description for each variable is available in the last section of this document. After completing this step, the scripts are ready to use. The syntax for calling the scripts is as follows:

1. diffampPWR : $[W, R, BW] = diffampPWR(vsw, av, ibias, scale, cfix, corner)$;

2. diffampBW : $[W, R, Ibias] = diffampBW(vsw, av, bw, scale, cfix, corner)$;

Refering to Figure 1, the outputs from the script are:

1. W: transistor width

2. R: load resistance

3. BW: resulting amplifier bandwidth if diffampPWR is used

4. Ibias: required bias current if diffampBW is used

The arguements to the script are:

1. V_{sw}: Output voltage swing ($I_{bias}R$)

2. A_v: Voltage gain (g_mR)

3. I_{bias}: Desired bias current for diffampPWR script

4. BW: Desired amplifier bandwidth for diffampBW script

5. scale: Ratio of loading stage to driving stage (α in Figure 2)

6. C_{fix}: fixed load capacitance due to interconnect, etc. (refer to Figure 2)

7. corner: Process and temperature corner

5 Examples

Two examples are presented to illustrate how the amplifier scripts can be used in the design of high-speed, resistor loaded, differential amplifiers.

Example 1

The first example explores the design of an amplifier driving both a fixed load capacitance and a scaled replica amplifier as encountered in a cascaded limit amplifier, as shown in Figure 3. Assume that we want to design a multi-stage limit amplifier with a 1V output

swing and gain and bandwidth of 2V/V and 5.0GHz, respectively, per stage. The fixed capacitance, due to interconnect, is 10fF and each stage of the limit amplifier is identical ($\alpha = 1$). Consider typical conditions at room temperature. Since the bandwidth is specified use diffampBW:

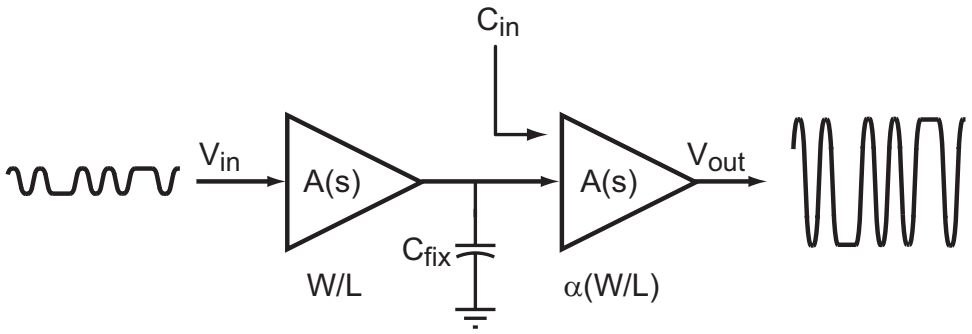

Figure 3: Differential amplifier loaded by fixed capacitance and scaled differential amplifier cell

$[W, R, I] = diffampBW(1.0, 2.0, 5.0e9, 1, 10e-15, 't25')$
$W = 1.285e-5$
$R = 437.4028$
$I = 0.0023$

The resulting transistor width, load resistance and bias current are 12.85um, 437.4ohms and 2.3mA, respectively. Alternately, if we wanted to limit the power dissipation to 1.5mA per stage:

$[W, R, B] = diffampPWR(1.0, 2.0, 1.5e-3, 1, 10e-15, 't25')$
$W = 8.4375e-6$
$R = 666.6667$
$B = 4.6921e9$

The resulting transistor width, load resistance and bandwidth are 8.44um, 667ohms and 4.69GHz, respectively. The script allows us to quickly investigate trade-offs between various design parameters.

Example 2

The second example explores the design of an amplifier driving a only a fixed capacitive load as encountered in the design of an output buffer (neglecting the package), as shown in Figure 4. Assume that we want to design an output buffer driving a lumped capacitive load of 500fF with a 1V output swing. Additionally, we would like to have a gain and bandwidth of 1.3V/V and 7.0GHz, respectively. To ensure that the design works well across temperature, check the design at $-40°$C, $0°$C and $85°$C. Only consider typical conditions. Since the bandwidth is specified use diffampBW:

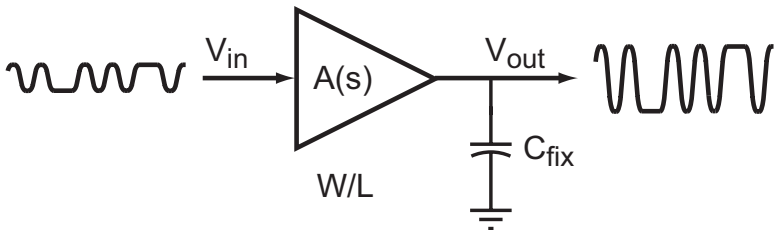

Figure 4: Differential amplifier loaded by fixed capacitance only

$[W, R, I] = diffampBW(1.0, 1.3, 7.0e9, 0, 5e-13,'t25')$

$W = 7.111e-5$

$R = 43.9398$

$I = 0.0228$

$[W, R, I] = diffampBW(1.0, 1.3, 7.0e9, 0, 5e-13,'t-40')$

$W = 5.912e-5$

$R = 44.3937$

$I = 0.0225$

$[W, R, I] = diffampBW(1.0, 1.3, 7.0e9, 0, 5e-13,'t85')$

$W = 8.609e-5$

$R = 43.5540$

$I = 0.0230$

To satisfy the worst case condition (85°C), the resulting transistor width, load resistance and bias current are 86.1um, 43.6ohms and 23.0mA, respectively. The script allows us to quickly investigate the impact of process and temperature variations on the design.

6 SUPPLEMENTAL INFORMATION

Using the HSPC toolbox for Hspice (http://www-mtl.mit.edu/research/perrottgroup/tools.html) there are 3 parameters that specify how the source/drain area and perimeter dimensions are calculated. The first two parameters, hdin and hdout, are process specific and depend on the design rules. Refering to Figure 5, hdin and hdout are the minimum source/drain extensions from the gate poly to composite edge with and without diffusion area sharing, respectively. Hdin is calculated as the sum of twice the minimum gate to contact space plus the mininum contact size. Hdout is calculated as the minimum gate to contact spacing plus the minimum contact size plus the minimum contact to composite edge space.

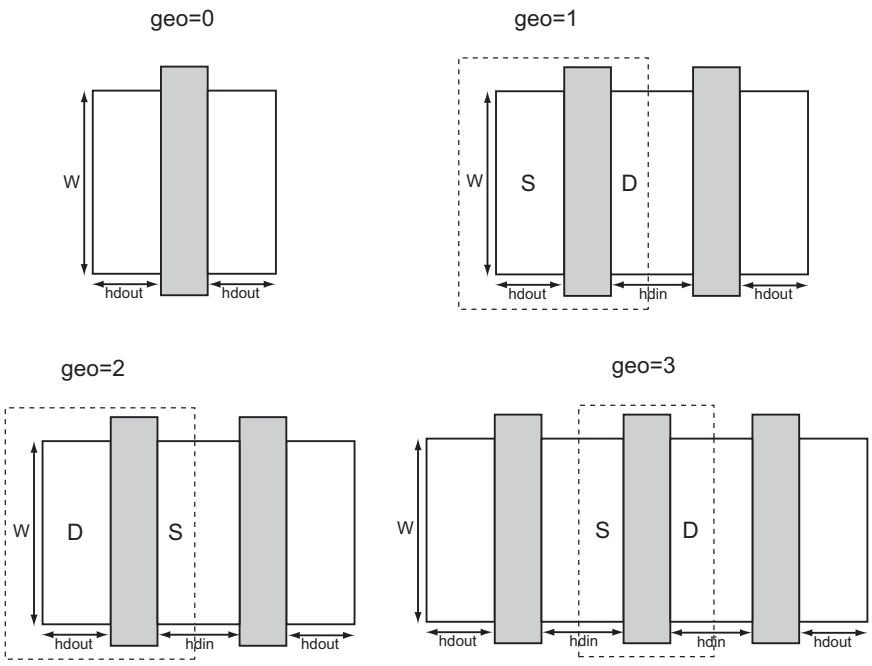

Figure 5: Diffusion parameters for different values of geo

The geo factor defines the topology of the transistor, as shown in Figure 5, and determines whether hdin or hdout is used in the diffusion area and perimeter calculations.

By default, both scripts assume minimum length devices for high-speed operation. However, the length can be increased for different bandwidth, noise or matching constraints. Simply change the length (l) in the spice netlist, *test.sp* in **./diffamp/spicefiles**, to the desired value, run the characterization file in Hspice and update the length, denoted lmin, in each of the Matlab scripts.

A constant current is used in the Hspice simulations that generate the process characterization data while the device width, W, is swept. If the diffampPWR script is used, so that power is fixed, the ibias parameter in the Hspice netlist should be set to the desired value for maximum accuracy. If the diffampBW script is used both the power and device size vary to satisfy the set of design constraints so ibias can not be set to the final value directly. In practice, the amount of error introduced due to this issue is minimal and the design parameters achieved with the default settings are satisfactory. However, if a higher degree of accuracy is desired, a first design iteration can be done with the ibias value set to an initial guess. The value of ibias for the spice netlist can be retargeted for subsequent iterations based on the results of the diffampBW script.

The process and temperature corners were set by default to be (slow, typical, fast) and ($-40°$C, $25°$C, $85°$C) but can be adjusted if both the netlist file and the scripts are updated.