DESIGN AND DEVELOPMENT OF GENETIC CIRCUIT OPTIMIZER

Y.L.Yap, Y.C.Wong, David F.W.Yap, S.S.S. Ranjit

Faculty of Electronics and Computer Engineering (FKEKK)
Universiti Teknikal Malaysia Melaka (UTeM)
Hang Tuah Jaya, 76100, Durian Tunggal, Melaka, Malaysia

ycwong@utem.edu.my

Abstract

Analog designers are interested in using optimization tools which may automate the process of transistor sizing. Genetic Algorithm (GA) uses the length and width variables of MOSFET to optimize the size of transistors in Matlab environment. The interface of few MOSFET parameters from circuit simulator PSpice to GA in Matlab was designed and demonstrated by a simple circuit.

Keywords: *Genetic algorithm, automated, MOSFET circuit, circuit optimization.*

I. INTRODUCTION

Transistor size will influence the speed of circuit, energy consumption, total area of circuit, and the delay constraints. Analog designers are tasked to taking large and small signal models of circuit components, and in order to fulfill certain performance requirements, deducing the components input parameters in accordance to output constraints. The process of translating the performance measurements into component parameters is called circuit sizing. In a modern analog design process, designers need to specify between 10 to 100 input parameters in order to achieve up to 20 output performance measurements [1]. Several automated and manual methods for circuit sizing exist in practice. The manual method involves a designer using his or her own accumulated knowledge of circuit behavior to iteratively adjust the component parameters such that they satisfy a set of first order transistor models, and then test the accuracy of these models.

During the design stages, simulating the circuit with Pspice yields a good approximation of circuit behavior. Pspice is a circuit simulator with highly complex physical device models state by R. Kraus [2]. Its results reliably confirm whether a circuit meets its performance specifications given its sizing parameters. In the field of automated circuit design, GA has been previously used as a standalone method in generating both the topology and components of a circuit [3]. J.R. Koza [4] had shown that, given the number of inputs and outputs of the circuit, the set of available components, and a fitness measurement in terms of performance, GA tended to perform satisfactorily in synthesizing eight different analog circuits.

Problem statement

Many designers struggle in optimizing the transistor's width and length. Usually, optimization process is done by trial and error technique. Therefore, fast and reliable artificial intelligent (AI) tools have become a demand for analog designers. Moreover, reduction of time needed for circuit simulation in PSpice is the main problem. The time used for PSpice to simulate each set of circuit parameter was time consumed. So, a simple and fastest mode of Pspice simulator is an essential element to reduce the simulation period.

II. LITERATURE REVIEW

Genetic circuit optimizer follows a general GA flow from N. Fujii and H. Shibatu [5] as Figure 1. The first step is to create the first generation population. Usually, the first population is generated randomly. However, in transistor sizing, the first population is populated with parameters that are known to have a range close to the desired value. This will lead to a quicker convergence in the appropriate range by reducing the search space. Next, chromosomes will be decoded into a format recognizable by Pspice. Pspice will simulate each parameter, and then fitness function is applied to the Pspice output data to produce a fitness value for each parameter. Hence, as in the general GA, the steps of selection, crossover, mutation, and the re-evaluation of fitness measures are applied iteratively until a good transistor parameter that meets the specification is found.

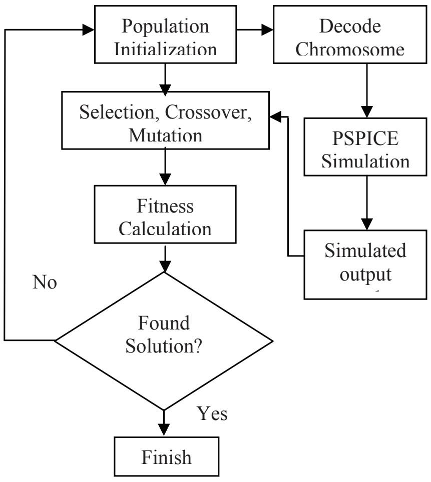


Fig 1. GA flow for Transistor sizing optimization.

Chromosome encryption

One of the important aspects of GA is the choice of how to encode a solution. In this case, the transistor parameter will be represented as a chromosome. The encoding can directly affect the ability of the iterative process to converge on an appropriate solution. GA will process information of component values such as resistor values and transistor sizes. So the values or parameters of a transistor can be fitted into a chromosome as in Figure 2. Each gene represented one of the transistor parameter and had two fields: width and length of transistor which will be multiply with micro meter, μm. Finally, output current, I_{out} will be inserted in to the chromosome as the last variable.

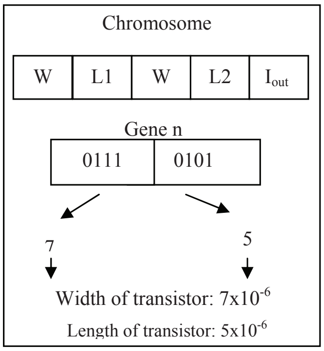
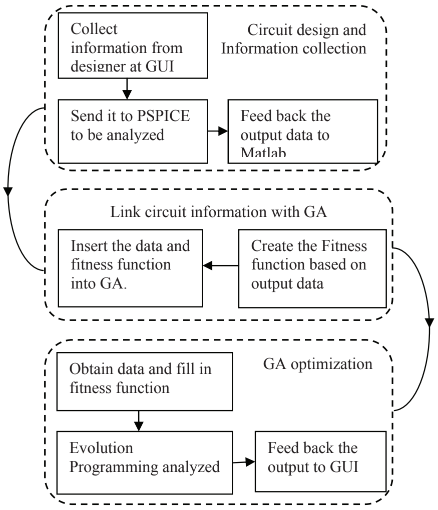


Fig 2. Encoding for a transistor width as 7μm and length as 5μm and using binary encoding technique.

III. METHODOLOGY

Optimization process shown in Figure 3 consists of three main stages: circuit design, link circuit information with GA, and GA optimization [6]. The circuit design stage uses Pspice coding to draw and insert the required parameter information. Secondly, the GA optimization stage uses GA to explore the best circuit parameters that correspond to the given width and length. The information keyed in via Graphics User Interface (GUI) will be linked using Matlab command to the GA to optimize it and eventually display the result. In the link stage of this system, the fitness function would be calculated based on the input and output data provided by Pspice simulation. These data's will be matched to create the fitness function and will be used in GA optimization.


Collect information from designer at GUI
Circuit design and Information collection
Send it to PSPICE to be analyzed
Feed back the output data to Matlab
Link circuit information with GA
Insert the data and fitness function into GA.
Create the Fitness function based on output data
Obtain data and fill in fitness function
GA optimization
Evolution Programming analyzed
Feed back the output to GUI

IV. SYSTEM DESIGN

The design starts with the PSpice circuit analysis. PSpice coding such as netlist or command code are necessary for us to modify transistor parameter when interfacing with Matlab [7]. Interfacing Pspice and Matlab to create the flow of data are focused. Matlab will collect the data of Direct Current (DC) swap from Pspice then fit into chromosome in GA algorithm. After that, fitness value will be evaluated base on the parameter in chromosome [8].

A. simple PSpice circuit

Circuit in Figure 4 is a simple MOSFET inverter circuit. Output of the inverter is amplified using cascaded amplifier setup that is transistors M3 and M4. Output current at resistor, R2 will be taken as circuit output to be analyzed in GA.

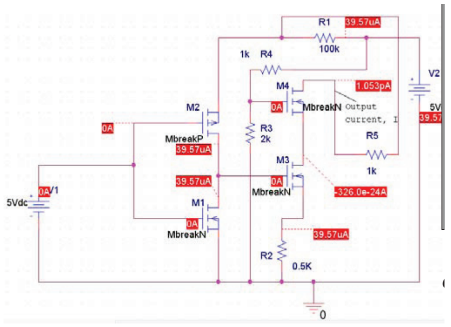
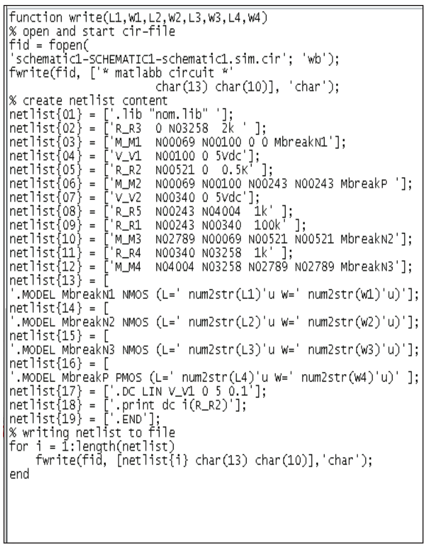


Fig 4. Schematic of circuit in PSPICE where output current is to be analyzed.

B. Create Netlist

Pspice read out and analyze the netlist. Then, parameters of transistors in the circuit will be setup according to the value in nestlist. The transistor model will be recognized and value inserted to in Pspice. Example, value hold by the variable name num2str(L1) will transfer to MbreakN1 transistor model. Value of Num2str(L1) refer to the random generated chromosome parameters. Next, PSpice will simulate the circuit with DC swap by 0.5 increments from 0 to 5V to the transistor which is stated in the netlist in Figure 9. Lastly, Pspice will print the output to be analyzed at i(R_R2) as the result.

Netlist of the circuit from Figure 5 is shown as below.

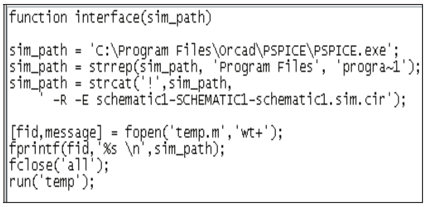


```
function write(L1,W1,L2,W2,L3,W3,L4,W4)
% open and start cir-file
fid = fopen(
'schematic1-SCHEMATIC1-schematic1.sim.cir'; 'wb');
fwrite(fid, ['* matlabb circuit *'
                    char(13) char(10)], 'char');
% create netlist content
netlist{01} = ['.lib "nom.lib" '];
netlist{02} = ['R_R3  0 N03258  2k ' ];
netlist{03} = ['M_M1  N00069 N00100 0 0 MbreakN1'];
netlist{04} = ['V_V1  N00100 0 5Vdc'];
netlist{05} = ['R_R2  N00521 0  0.5K' ];
netlist{06} = ['M_M2  N00069 N00100 N00243 N00243 MbreakP '];
netlist{07} = ['V_V2  N00340 0 5Vdc'];
netlist{08} = ['R_R5  N00243 N04004  1k' ];
netlist{09} = ['R_R1  N00243 N00340  100k' ];
netlist{10} = ['M_M3  N02789 N00069 N00521 N00521 MbreakN2'];
netlist{11} = ['R_R4  N00340 N03258  1k' ];
netlist{12} = ['M_M4  N04004 N03258 N02789 N02789 MbreakN3'];
netlist{13} = [
'.MODEL MbreakN1 NMOS (L=' num2str(L1)'u W=' num2str(W1)'u)'];
netlist{14} = [
'.MODEL MbreakN2 NMOS (L=' num2str(L2)'u W=' num2str(W2)'u)'];
netlist{15} = [
'.MODEL MbreakN3 NMOS (L=' num2str(L3)'u W=' num2str(W3)'u)'];
netlist{16} = [
'.MODEL MbreakP PMOS (L=' num2str(L4)'u W=' num2str(W4)'u)' ];
netlist{17} = ['.DC LIN V_V1 0 5 0.1'];
netlist{18} = ['.print dc i(R_R2)'];
netlist{19} = ['.END'];
% writing netlist to file
for i = 1:length(netlist)
    fwrite(fid, [netlist{i} char(13) char(10)],'char');
end
```

Fig 5. Netlist of the schematic from Figure 4.

C. Interfacing Pspice with Matlab

Function of interface (Sim_path) will activate PSpice program and data in the file of Schematic-SCHEMATIC1-schematic1.sim.cir containing the netlist of the circuit will be imported. The Temporary m-file, temp will be created as the workspace. Find and Replace, Strrep function will replace all the occurrence of Sim_path and progra~1 to Program Files. String concatenate, Strcat function will horizontally concatenate Matlab with Pspice and the file name Schematic-SCHEMATIC1-schematic1.sim.cir inside it. This code will be executed in temp.m to create the interface process of Pspice and Matlab.

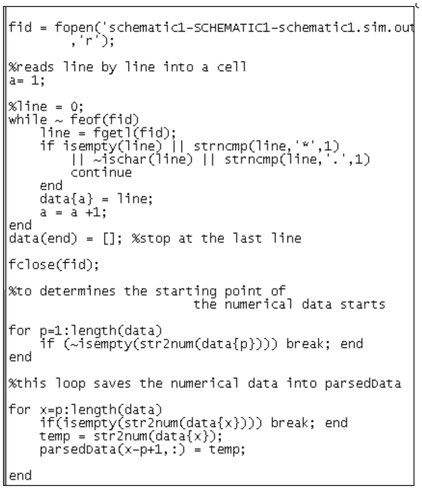


```
function interface(sim_path)

sim_path = 'C:\Program Files\Orcad\PSPICE\PSPICE.exe';
sim_path = strrep(sim_path, 'Program Files', 'progra~1');
sim_path = strcat('!',sim_path,
     ' -R -E schematic1-SCHEMATIC1-schematic1.sim.cir');

[fid,message] = fopen('temp.m','wt+');
fprintf(fid,'%s \n',sim_path);
fclose('all');
run('temp');
```

Fig 6. MATLAB will interface with PSPICE to do circuit simulation.

D. Data Extraction from PSpice to Matlab

Looping technique in Figure 7 will be used to extract numerical data generated by PSpice. Matlab will recognize all the necessary character in the log file and fit the corresponding data into matrix column in workspace. The algorithm starts with the import all the data line by line into a temporary variable call data (*a*) where *a* was continuously updated. Then, numerical data inside this variable will be searched and fitted into a new variable name parsedData.



```
fid = fopen('schematic1-SCHEMATIC1-schematic1.sim.out
        ,'r');

%reads line by line into a cell
a= 1;

%line = 0;
while ~ feof(fid)
    line = fgetl(fid);
    if isempty(line) || strncmp(line,'*',1)
        || ~ischar(line) || strncmp(line,'.',1)
        continue
    end
    data{a} = line;
    a = a +1;
end
data(end) = []; %stop at the last line

fclose(fid);

%to determines the starting point of
                        the numerical data starts

for p=1:length(data)
    if (~isempty(str2num(data{p}))) break; end
end

%this loop saves the numerical data into parsedData

for x=p:length(data)
    if(isempty(str2num(data{x}))) break; end
    temp = str2num(data{x});
    parsedData(x-p+1,:) = temp;

end
```

Fig 7. Looping technique to parse data from PSpice into Matlab.

V. EXPERIMENTAL RESULT

Data of MOSFET inverter circuit in Figure 4 are loaded into Pspice and simulated. Output of the simulation is collected by parsedData function and will be inserting to chromosomes to evaluate. Together with the eight inputs of length and width of the transistors value is obtained using GA. Figure 8 shows the effect of using various sizes of the transistor parameter on ID and V_{GS}. Varying the transistor parameters will also vary the output current, I and eventually lead to a change to the switching time of the circuit. Hence, the automated interfacing between Pspice and Matlab was created.

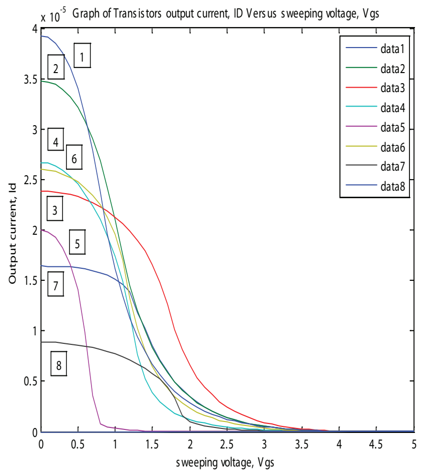


Fig 8. Graph of output current, IDS versus sweeping voltage, VGS

VI. CONCULSION

By empirically interfacing PSpice and Matlab, we have shown that the automated interfacing algorithm had successfully reduced the time consumed for import or export data between these software. It is clearly shown here that MOSFET parameters are difficult to be fitted by the designer manually into PSpice simulator. The optimization via GA frees up a lot of designing and circuit analysis time of the designer. The automated interfacing between PSpice and Matlab had shown great improvement to the analog circuit design area.

VII. REFERENCE

[1] Varun Aggarwal and Una-May O'Reilly, "Design of Posynomial Models for Mosfets: Symbolic Regression Using Genetic Algorithms", Genetic Programming: Theory and Practice IV : pp. 219-236, 2006.

[2] R. Kraus, P.Turkes, J.Sigg, "Physics-Based Models of Power Semiconductor Devices for the Circuit Simulator SPICE", 29th Annual IEEE Power Electronics Specialists Conference, Fukuoka, Japan, May 17-22, 1998.

[3] S.Z Ricardo, C. P Marco Aurelio, and Marley Maria B.R. Vellasco, "EVOLUTIONARY ELECTRONIC automatic design of electronic circuits and systems by genetic algorithm", pg 92-93, CRC Press, New York, 2002.

[4] J.R. Koza, HB Forrest, David Andre, Martin Keane, and Frank Dunlap, "Automated Synthesis of Analog Electrical Circuits by Means of Genetic Programming", IEEE Tracsactions on Evolutionary Computation, 1(2): pp. 109- 128, 1997.

[5] N. Fujii and H. Shibatu, "Analog Circuit Synthesis by Superimposing of Sub-Circuits," Proceedings of the International Symposium on Circuits and Systems, vol. 5, no. 1, pp. 427-430, 2001.

[6] JH Yu, ZG Mao, "A design method in CMOS operational amplifier optimization based on adaptive genetic algorithm", WSEAS Transactions on circuit and systems, vol. 8, pp 548-558,2009.

[7] O.A John, "PSPICE and MATLAB for Electronic: An integrated approach", CRC Press LLC, 2000.

[8] S.N.Sivanandam and S.N.Deepa, "Introduction to Genetic Algorithm", Springer – Verlag Berlin Heidelberg, 2008.