Circuitos Elétricos II – 1º semestre de 2017 - Trabalho

Prof. Antônio Carlos Moreirão de Queiroz

Escrever um programa que analise circuitos lineares contendo transistores bipolares, encontrando o ponto de operação e a resposta em frequência para pequenos sinais.

O programa deverá analisar um circuito composto pelos elementos:

- Fontes de corrente e de tensão independentes.
- Resistores, capacitores, indutores e transformadores.
- As quatro fontes controladas.
- Amplificadores operacionais ideais, de 4 terminais.
- Transistores bipolares, com modelo de Ebers-Moll com efeito Early.

O programa deve ler um netlist, e realizar uma análise nodal modificada de ponto de operação, e após isto deve realizar análises no estado permanente senoidal do modelo de pequenos sinais, em uma faixa de frequências especificada. Uma linha de comando no netlist define os parâmetros necessários. O resultado deverão ser duas tabelas em arquivos, uma listando o ponto de operação encontrado (o programa pode apenas listar os valores do ponto de operação sem salvar a tabela, mas de forma que todos possam ser examinados), e outra tabela tendo a frequência como primeira coluna, e todas as tensões nodais e correntes nas fontes de tensão e curtos-circuitos nas outras colunas, em módulo e fase, plotável com outro programa. A primeira linha desta tabela deve listar os nomes de todas as variáveis na tabela, com correntes citadas como "j" seguidas do nome do elemento onde estão e sufixos m e f para indicar módulo e fase. Ex: f 1m 1f 2m 2f 3m 3f jH1m jH1f jF2m jF2f.

O programa pode ser baseado no programa exemplo MNA1, que implementa a análise nodal modificada para um circuito resistivo linear. Uma versão gráfica (para o Borland C++ builder 6) está também disponível:

http://www.coe.ufrj.br/~acmq/cursos/mna1.zip
http://www.coe.ufrj.br/~acmq/cursos/mna1gr.zip

Formato para o netlist:
O netlist pode ser gerado pelo programa Edfil. Veja os programas em http://www.coe.ufrj.br/~acmq/cursos

Primeira linha: Comentário (o editor Edfil coloca o número de nós nesta linha).
Linhas seguintes: Descrição do circuito, com um elemento por linha. A primeira letra determina o tipo de elemento.

Resistor: R<nome> <nó1> <nó2> <Resistência>
Indutor: L<nome> <nó1> <nó2> <Indutância>
Acoplamento entre indutores: K<nome> <La> <Lb> <*k*> (La e Lb nomes de indutores já declarados.)
Capacitor: C<nome> <nó1> <nó2> <Capacitância>
Fonte de tensão controlada a tensão: E<nome> <nóV+> <nóV-> <nóv+> <nóv-> <*Av*>
Fonte de corrente controlada a corrente: F<nome> <nóI+> <nóI-> <nói+> <nói-> <*Ai*>
Fonte de corrente controlada a tensão: G<nome> <nóI+> <nóI-> <nóv+> <nóv-> <*Gm*>
Fonte de tensão controlada a corrente: H<nome> <nóV+> <nóV-> <nói+> <nói-> <*Rm*>
Fonte de corrente: I<nome> <nó+> <nó-> <módulo> <fase (graus)> <valor contínuo>
Fonte de tensão: V<nome> <nó+> <nó-> <módulo> <fase (graus)> <valor contínuo>
Amplificador operacional ideal: O<nome> <nó saída+> <nó saída-> <nó entrada+> <nó entrada->
Transistor bipolar: Q<nome> <nóc> <nób> <nóe> <tipo> <α> <αr> <$I_s$be> <$V_T$be> <$I_s$bc> <$V_T$bc> <VA> <C_0be> <C_1be> <C_0bc> <C_1bc>
Comentário: *<comentário>

(Notar que <xxx> significa colocar o valor xxx sem <>.)

O programa exemplo permite nomes nos nós. O programa feito pode continuar permitindo usando o mesmo algoritmo (código no programa MNA1), ou admitir apenas números. Neste caso a primeira linha gerada pelo editor Edfil pode ser usada.

Atualizado em 3/5/2017

As direções para fontes são de acordo com a ordem dos nós e as direções convencionais associadas, sendo o primeiro nó o positivo.

O programa deve ler as instruções de como tratar o netlist de uma linha de comando no próprio netlist, no formato abaixo. Não deve ser necessário fornecer qualquer outro parâmetro ao programa além do arquivo de entrada.

.AC <LIN ou OCT ou DEC> <pontos> <início> <fim>

Como no SPICE, o número de pontos é por década ou por oitava com as opções DEC e OCT. Senão é o total com a opção LIN. As frequências são em Hz. Podem haver opções internas no programa para usar frequências em rad/s e para gerar módulos em dB.

Opcionalmente, o próprio programa pode plotar seus gráficos. O programa RFNLIN pode ser usado para plotar os gráficos. Este programa faz exatamente a mesma análise, mas por enquanto sem tratar os transistores.
http://www.coe.ufrj.br/~acmq/cursos/rfnlin.zip

O programa deve ser escrito preferencialmente em uma linguagem compilada como C, C++ ou Pascal. O programa deve preferencialmente rodar em ambiente gráfico Windows. Evite usar apenas uma interface de console. Um arquivo .zip com tudo o que for necessário para o programa, inclusive fontes, arquivo executável, documentação, bibliotecas e exemplos não deve ter mais de 4 Mbytes. Evite sistemas de desenvolvimento que requeiram extensas bibliotecas instaladas.

Sugere-se partir do programa exemplo MNA1, que já tem o algorítmo completo da análise nodal modificada em C, implementar a análise do ponto de operação com transistores bipolares, e então a análise no estado permanente senoidal, operando com variáveis complexas. Note que a montagem do sistema de equações e sua resolução são similares em todas as análises, com as mesmas variáveis. Embora seja ineficiente, a solução do sistema complexo pode ser usada no cálculo do ponto de operação também. Em C, é simples usar variáveis complexas.

Os transistores NPN e PNP devem ser tratados pelo modelo de Ebers-Moll com controles pelas correntes nos diodos, com o efeito Early implementado para transistores não invertidos. É conveniente acompanhar a listagem do ponto de operação com uma lista dos estados dos transistores, com tensões, correntes e capacitâncias. Pode ser implementado, como alternativa, o modelo com controle pelas correntes i_c e i_e.

O programa deve tratar as capacitâncias dos transistores como as capacitâncias dos diodos be e bc. Eles tem duas capacitâncias em paralelo:

$$C_{reversa} = \frac{C_0}{\left(1-\frac{v}{\phi}\right)^n} \text{ (se } v > \frac{\phi}{2} \text{ usar } v = \frac{\phi}{2})$$

$$C_{direta} = C_1(e^{v/V_T} - 1) \text{ para } v > 0$$

Os parâmetros são dados diretamente no netlist. Seja por exemplo um diodo com: $C_{reversa}$ = 5 pF com v = 0 V e C_{direta} = 100 pF com i = 1 mA. O diodo tem I_s = 1 nA e v = 0.6 V com 1 mA. Os parâmetros do diodo são então: I_s = 1 nA, V_T = v/Ln(i/I_s+1) = 43.43 mV (seria ηV_T, já com a correção). Para os capacitores, seria C_0 = 5 pF, e C_1 = $C_{direta}I_s/i$ = 100 x 10^{-18} F. Os parâmetros do modelo dos diodos podem ser n = 0.5 e ϕ = 0.6 V. Notar que as duas capacitâncias aparecem se v>0. A descrição do transistor seria algo como:

Q1 1 2 3 NPN 0.995 0.5 1e-9 43.43e-3 1e-9 43.43e-3 100 5e-12 100e-18 5e-12 100e-18

O programa pode prover valores “default” se não forem dados todos os parâmetros. O diodo pode ser também implementado como elemento, descrito como D<nome> <nóa> <nók> <I_s> <V_T> <C_0> <C_1>.

Grupos de 3 alunos, no máximo. O programa deverá ser apresentado e demonstrado, completamente funcional, por todo o grupo, e entregue com um relatório com comentários e exemplos significativos e originais verificados, até (entenda-se antes de) duas semanas antes da segunda prova.

Note-se que o trabalho é bastante extenso, e deve ser começado imediatamente.

Atualizado em 3/5/2017