

Circuitos Elétricos II – 2º semestre de 2016 - Trabalho

Prof. Antônio Carlos Moreirão de Queiroz

Escrever um programa que analise circuitos no domínio do tempo, contendo elementos lineares e não lineares, usando análise nodal modificada e o método dos trapézios junto com o método de Newton-Raphson.

O programa deverá analisar um circuito composto pelos elementos:

- Fontes de corrente e de tensão independentes.
- Resistores, capacitores e indutores.
- As quatro fontes controladas.
- Amplificadores operacionais ideais, de 4 terminais.
- Transformadores ideais.
- Diodos ideais
- Chaves ideais

O programa deve ler um netlist descrevendo o circuito, inicializar a análise no domínio do tempo com cálculo de ponto de operação, e então fazer a análise com o tempo total e o tamanho do passo, fixo, dados. Os resultados devem ser salvos em uma tabela em arquivo, de forma que possam ser lidos por outro programa que plote as curvas, como o MNAE. Uma linha de comando no netlist define os parâmetros necessários. A primeira linha desta tabela deve listar os nomes de todas as variáveis na tabela, iniciando pelo tempo “t”, com correntes citadas como “j” seguidas do nome do elemento onde estão. Ex: t 1 2 3 jH1 jF2.

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>

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-> <parâmetros>

Fonte de tensão: V<nome> <nó+> <nó-> <parâmetros>

Amplificador operacional ideal: O<nome> <nó saída+> <nó saída-> <nó entrada+> <nó entrada->

Diodo ideal: D<nome> <nó anodo> <nó catodo>

Chave ideal: \$<nome> <nó a> <nó b> <nó controle c> <nó controle d> <V_{lim}>

Transformador ideal: K<nome> <nó a> <nó b> <nó c> <nó d> <n>

Comentário: *<comentário>

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

Os parâmetros para as fontes são de acordo com o formato do SPICE, como implementado no programa MNAE. Devem ser suportadas fontes contínuas, senoidais e em pulsos.

DC <valor>

SIN <nível contínuo> <amplitude> <frequência em Hz> <atraso> <amortecimento> <defasagem em graus>
<número de ciclos>
PULSE <amplitude 1> <amplitude 2> <atraso> <tempo de subida> <tempo de descida> <tempo ligada>
<período> <número de ciclos>

A fonte senoidal vale:

$$x(t) = A_0 + Ae^{-\alpha(t-t_a)} \operatorname{sen}\left(2\pi f(t-t_a) + \frac{\pi}{180}\varphi\right)$$

onde A_0 é o nível contínuo, A a amplitude, f a frequência, t_a o atraso, α o amortecimento e φ a defasagem. Antes de $t = t_a$ ou após o número de ciclos, tem o valor inicial ou final respectivamente, de forma a não criar discontinuidades.

A fonte pulsada começa na amplitude 1, e fica aí até o fim do tempo de atraso. Então muda para a amplitude 2 variando linearmente dentro do tempo de subida, fica na amplitude 2 durante o tempo ligada, volta à amplitude 1 dentro do tempo de descida, e repete tudo com o período e o número de ciclos especificados. Termina na amplitude 1. Os tempos de subida e de descida podem ser nulos. O programa pode usar o tempo do passo então.

Os diodos e chaves ideais devem ter resistência infinita quando cortados e zero quando conduzindo. Deve ser necessário calcular as correntes nesses elementos como incógnitas. A chave conduz se $v_{cd} > V_{lim}$. Pode ser implementado um parâmetro V_d para o diodo, de forma que ele conduza quando $V_d > 0$.

O transformador ideal deve implementar $v_{cd} = n v_{ab}$ e $j_{ab} = -n j_{cd}$. A corrente j_{cd} deve ter que ser calculada. O transformador pode ser sempre ideal, funcionando mesmo com sinais contínuos, inclusive na análise do ponto de operação.

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.

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.

.TRAN <tempo final> <passo> TRAP <passos por ponto na tabela>

No método de Newton-Raphson, caso não ocorra convergência em um número razoável de iterações, as variáveis que não estão convergindo devem ser reinicializadas com valores randômicos. Se a reinicialização for feita mais que um número razoável de vezes, a análise deve ser interrompida, e o arquivo de saída fechado. A aproximação inicial é a solução do tempo anterior. No cálculo do ponto de operação, pode ser usada uma aproximação inicial nula.

Opcionalmente, o próprio programa pode plotar seus gráficos. O programa MNAE pode ser usado para plotar os gráficos. Este programa faz exatamente a mesma análise.

<http://www.coe.ufrj.br/~acmq/programs/mnae.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 requirem extensas bibliotecas instaladas.

Sugere-se partir do programa exemplo MNA1, que já tem o algoritmo completo da análise nodal modificada em C, implementar a análise no tempo com as fontes de sinal, e então implementar os elementos reativos. Por fim os não lineares com o ciclo de Newton-Raphson.

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. Trabalhos incompletos serão devolvidos para serem completados, só sendo aceitos completos.