特輯 技術解說

Analog 회로 설계를 위한 시뮬레이션 Tool

全 永 鉉, 金 春 慶
(株)金星일렉트론 半導體研究所

I. 서 론

전자회로의 설계는 그 설계된 회로의 성능을 정확히 측정하기 위하여 정확한 측정 방법을 요구하고 있다. 개별 소자로 설계된 회로의 성능 측정은 전형적인 breadboard를 사용하여 전기적인 특성을 측정할 수 있었다. 그러나 집적회로(integrated circuit)의 설계는 그와는 전혀 다른 문제를 가지고 있다. Breadboard에 존재하는 기생 소자(parasitic component)는 집적 회로에 존재하는 기생 성분과는 완전히 다른 특성을 나타내고 있다. 이러한 이유때문에, breadboard를 사용하여 집적 회로의 성능을 정확히 측정하기는 힘들 뿐만 아니라 집적회로의 각 소자를 각각의 대응하는 개별소자로 모델링(modeling) 하기는 매우 힘들다. 또한 현재의 집적회로는 하나의 칩위에 수만개의 소자를 집적하고 있는데 이와 같은 수만개의 소자를 breadboard상에 배열하기는 불가능하다.

그러나 집적회로의 제조전의 성능 측정은 필요 불가결하므로, 이를 위하여 컴퓨터 프로그램이 개발 되었으며, 이러한 프로그램을 일반적으로 회로 해석프로그램(circuit simulation program)이라고 부르고 있다. 사실, 회로 시뮬레이션 프로그램은 집적회로의 설계시 CUP-시간면에서 가장 널리 사용되는 컴퓨터를 이용한 회로 설계도구(CAD tool)이다. 전자회로의 전기적 특성을 해석하는 회로 해석 프로그램은 회로 특성에 나타나는 많은 실질적 문제를 다룰 수 있게 되어 있다. 설계되어진 회로는 내부적으로 수학적 수식으로 나타내어지며, 그러한 수식은 수치해석(numerical analysis) 방법을 사용하여 풀게 된다. 그래서 회로 해석 프로그램의 출력은 주어진 회로가 제조된 이후에 나타내어 질 수 있는 회로 특성을 시뮬레이션한 결과를 보여 주게 된다.

회로 해석 프로그램은 1970년대 이래로 집적 회로 설계시 가장 널리 사용되어 왔다. 그러나, 하나의 칩 위에 집적된 소자의 수가 증가함에 따라 이러한 복잡성을 해결하기 위하여 일련의 높은 단계의 해석 프로그램(high level simulation program) 개발의 필요성이 요구되어 왔다. 이러한 프로그램들은 행동단계 시뮬레이터(behavioral simulator), 함수 단계 시뮬레이터(functional level simulator), 게이트 단계 시뮬레이터(gate level simulator)를 포함하며, 최근에는 스위치 단계 시뮬레이터(switch level simulator)도 개발되어 사용되고 있다.

이러한 프로그램들은 회로의 기능 검증 및 일차적인 시간 특성(first order timing characteristics)을 검증하기 위하여 사용되고 있다. 그러나 이러한 높은 단계의 해석 프로그램들은 주로 논리회로(digital circuit)의 논리 검증에 국한되어 사용되어지고 있으므로, 사실 회로 해석 프로그램만이 전체적인 회로의 동작 특성이나 시간검증(timing verificatior)에 대한 충분한 정보를 제공하는 유일한 도구가 되고 있다.

현 단계에서 가장 널리 사용되는 회로 해석 프로그램은 SPICE(simulation program integrated circuit emphasis) 프로그램이다.[1] 지금 현재 거의 모든 회로 설계회사에서는 이 프로그램을 사용하여 회로를 설계하고 있으며 또한 상업적으로 개발된 많은 변형된 SPICE 프로그램(HSPICE, PSPICE, IGSPICE)들이 사용되고 있다. 이러한 프로그램 모두는 DC 동작점 해석(operating point analysis), 시간 영역 과도 해석(time comain transient analysis), AC 해석(AC analysis), 잡음해석(noise analysis)과 변형

해석(distortion analysis)등과 같은 다양한 해석을 할 수 있게 제공되어 진다. 이중에서 시간 영역 과도해석이 CPU 시간 측면에서 가장 시간이 많이 걸리는 해석이 된다. SPICE 프로그램이 처음 개발 되었을 당시에는 대략 200개 이하의 개별 소자를 해석하기 위하여 개발되었으나, 현재 많은 회사에서는 10,000개 이상의 소자를 포함한 회로해석을 위하여도 많이 사용되고 있으며, 이의 해석을 위하여 많은 시간을 필요로 하고 있다.

통계적으로 알려진 바에 의하면 많은 설계회사에서 평균적으로 매달 50,000번 이상 SPICE 프로그램을 사용하고 있으며 SPICE 수행의 80%이상(CPU 시간 20% 이하)이 수백개 이하의 소자를 포함하는 회로의 해석에 사용되고 있으며 20%이하 (CPU 시간 80% 이상)가 그 이상의 소자를 포함하는 회로의 해석에 사용되고 있다. 이러한 이유에 의하여 보다 빠른 시간안에 정확하게 회로를 해석할 수 있는 회로해석 프로그램의 개발이 추진되고 있으나 아직 SPICE보다 나은 해석 프로그램이 개발되고 있지는 않다고 볼 수 있다.

Ⅱ. 회로 시뮬레이션 (Circuit Simulation)

SPICE에서 전기적 과도현상(electrical transient analysis) 문제는 비선형 상 미분 방정식(nonlinear ordinary differential equation)을 형성하여 푼다는 의의미를 가지고 있다. 이러한 방정식들은 가해진 외부 입력전압과 주어진 초기조건(initial conition) 하에서 회로의 동적 특성(dynamic characteristics)을 모델한 것이다. 이러한 방정식을 푼 결과는 각 회로 노우드(node)에서의 전압 파형이나 각 회로 소자(element)에 흐르는 전류 파형이다.

전형적인 회로 시뮬레이터들은 그 회로 방정식을 풀기 위하여 직접 해석 방법(direct method)을 사용한다. 여기서 직접 해석 방법이란 비선형 상미분 방정식을 수치적분을 이용하여 비선형 차분 방정식(nonlinear difference equation)으로 변형한 다음 이 방정식을 반복적인 N-R 방법(newton raphson method)을 사용하여 해를 구하는 방법이다. 이러한 과정에는 주어진 방정식의 선형화와 선형화된 방정식을 LU 분해 방법에 의하여 해를 구하는 과정을 포함한다.[2] 이러한 과정에 의하여 큰 회로를 해석하는 데는 2가지 제약이 따르게 된다. 그 한가지는 LU 분해 방법에 의하여 큰 회로의 선형 방정식을 풀때 많은 시간이 소요되는 것이고, 다른 하나는 주어진 각 시간점(time point)에서 모든 변수들이 새로이 저장되어지며 풀리어져야 하므로 변하지 않은 변수들 조차 새로이 풀게 됨에 의한 시간의 낭비가 따르게 된다.

회로 시뮬레이터의 성능을 향상 시키기 위하여 많은 연구가 계속 되어왔다. 이러한 분야에 있어서의 초기의 작업은 시간 시뮬레이터(timing simulator) 라고 명명 되어진다.[3,4,5] 시간 시뮬레이터는 회로를 해석할 때 분할(tearing)과 반복(relaxation) 방법을 사용하여 회로를 해석함에 의하여 해석시간을 단축하였다. 최근에는 반복 해석 방법에 기본을 둔 많은 방법이 제안되었다. 특히 파형 이완 방법(waveform relaxation method)을 사용한 많은 시뮬레이션 프로그램들이 만들어 졌는데 이러한 프로그램으로는 RELAX[6], SWAN,[7] TOGGLE,[8] KTIME[9]등이 있다. 다른 법으로는 ITA(iterated timing analysis) 방법을 사용한 것으로서 SPLICE[10]와 ELDO[11]와 같은 프로그램들이 만들어져 사용되고 있다.

그러나 이와 같은 새로운 시뮬레이션 프로그램들은 적용되는 회로의 범위가 대부분 논리회로(digital circuit)에 국한 되어짐에 의하여 사용범위가 일반적이지 못하고 아직 이렇다할 상업용 시뮬레이션 프로그램들이 아직 개발되지 못하고 있는 실정이다. 이러한 이유에 의하여 아직까지 범용의 회로해석 프로그램인 SPICE를 가장 널리 사용하고 있으므로 본 논문에서는 SPICE에 사용되는 일반적인 회로 해석 방법에 대하여 알아보기로 한다.

Ⅲ. 회로 해석 방법

일반적으로 회로해석 프로그램의 구조는 주어진 IC 회로망에 대한 정보들을 효과적으로 정리, 저장하는 부분과 저장된 정보들로 부터 회로 해석 방정식을 형성하고 푸는 부분으로 나누어 진다. 이를 위하여 여러가지 기법들이 복합적으로 사용되는데, 이때 고려하여야 할 요소들은 다른 일반 컴퓨터 프로그램에서 처럼 사용하는 기법에 따른 효과적인 정보저장구조(data structure)의 결정, 계산 횟수의 최소화, 사용된 기법의 복잡도와 결과의 정확도 등이다. 회로 시뮬레이터에서는 이상의 모든 요소들이 최상의 방법으로 고려되고 가능한한 신뢰도가 높고, 결과가 정확한 기법들이 선택 사용된다.[12,13,14]

1. 회로 해석 방정식의 형성

대부분의 회로 해석 프로그램에서 사용하는 회로 해

석 방정식의 형성방법은 노우드 해석 방법(nodal analysis:NA)에 기초를 둔 변형 노우드 해석 방법(modified nodal analysis:MNA)이다. 노우드 해석방법은 회로내의 노우드 전압들을 회로 변수로 취하고 각각의 노우드를 중심으로 KCL(Kirchhoff's current law)을 적용하여 회로 해석 방정식을 구한다. 이 방법은 각 소자별로 해석 방정식의 기여도를 찾을 수 있으며 회로 해석 방정식의행렬의 대각선 항들이 '0' 이 아닌 값들을 갖는 잇점이 있다.

그러나 전원 전압이나 전류를 변수로 하는 소자들에 대해서는 취급하기가 곤란하다. 이러한 문제점들을 해결하기 위하여 변형 노우드 해석 방법에서는 일부 필요한 소자 전류들을 노우드 전압과 함께 회로 변수로 첨가하고, 첨가된 전류가 흐르는 소자의 전류 전압 관계식을 부수적인 회로 해석 방정식으로 사용함으로서, 노우드 해석 방식의 잇점을 그대로 유지할 수 있다. 변형 노우드 해석방법(MNA)은 노우드 해석방법(NA)의 일반화된 방법이라고 생각되어질 수 있다. 우선 MNA를 설명하기 전에 NA에 대해서 간략하게 설명한다.

NA는 기존의 여러 회로 해석 프로그램에 사용되는 방법으로 다음과 같은 논리적인 단계로서 구성되어 진다. 먼저 주어진 회로의 각 노우드에서 KCL을 적용하기 위하여, 독립 전류원의 전류와 가지 전압(branch voltage)으로 표현되는 가지 전류(branch current)를 수식화한 가지 방정식(branch equation)을 사용한다. 그리고 각 노우드 전압을 인수로 갖는 가지 전압을 나타내기 위하여 KVL(Kirchhoff's voltage law)을 사용한다. 이상에 설명한 단계를 그림 1의 예제에 각 단계별로 적용하면 다음과 같다.

각 노우드에 KCL을 적용한다.

노우드 1 : i1+i2+i3= 0

노우드 2 : −i3+i4−i5= 0

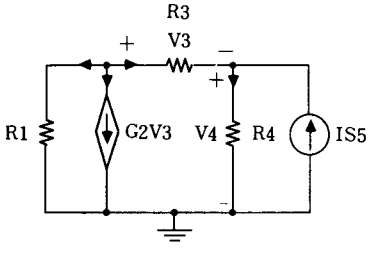


그림 1. 선형 저항 회로

각 가지 전류를 가지 전압으로 나타낸다.

노우드 1 : 1/R1. V1+G2. V3+1/R3. V3= 0

노우드 2 : −1/R3. V3+1/R4. V4−IS5= 0

각 가지 전압을 노우드 전압으로 나타낸다.

노우드 1 : 1/R1. e1+G2. (e1−e2)+1/R3. (e1−e2) = 0

노우드 2 : −1/R3. (e1−e2)+1R4. e2=IS5

위의 회로 수식을 행렬(matrix) 형태로 나타내면 다음과 같다.

$$Yn.e=\begin{bmatrix} 1/R1+G2+1/R3 & -G2-1/R3 \\ -1/R3 & 1/R3+1/R4 \end{bmatrix}\begin{bmatrix} E1 \\ E2 \end{bmatrix}$$

$$=\begin{bmatrix} 0 \\ IS5 \end{bmatrix}=b$$

여기서 계수 행렬(coefficient matrix) Yn은 노우드 어드미턴스 행렬(node admittance matrix)이라고 말하고 b는 우변 벡터(right hand side vector)라고 말한다. n개의 노우드를 가진 회로에 대해서는 n개의 미지 전압을 갖는 n개의 노우드 방정식이 형성되므로 A는 n×n 행렬이 되고 b는 n벡터가 된다.

이러한 NA의 장점은 계수 행렬에 있어서 대각선 항에 영을 포함하지 않으며 또한 대각선 지배적(diagonally dominant)이므로 선형 방정식을 풀 때 매우 간단해 진다는 것이다. 그러나 이러한 NA는 KCL에 기본을 둔 방법이므로 독립 전압 원, CCCS(current controlled current source), CCVS(current controlled voltage source) 와 VCVS(voltage controlled voltage source)등을 포함하는 회로에는 적용 시킬 수 없다. 그 이유를 알아보면, 노우드 방정식에서 위의 소자들의 가지 전압을 가지 전류로서 직접 대치할 수 없기 때문이다. 또한 NA에서 구해지는 것은 각 노우드의 미지 전압값이므로 만일 사용자가 각 가지에 흐르는 전류의 값을 알고자 할때는 각 노우드 전압으로부터 전류를 다시 계산하여야 하는 단점이 있다. NA에서의 이러한 단점을 보완하기 위하여 도입된 것이 MNA이다.

이러한 MNA는 대부분의 기존 회로 시뮬레이터에서 사용하는 방법으로 NA와 비슷한 논리 단계를 가지고 회로 방정식을 형성한다. 그림 2의 회로에 MNA를 각 단계별로 적용하면 다음과 같다.

각 노우드에 KCL을 적용한다.

노우드 1 : i1+i2+i3= 0

노우드 2 : −i3+i4−i5−i6= 0

노우드 3 : i6+i8= 0

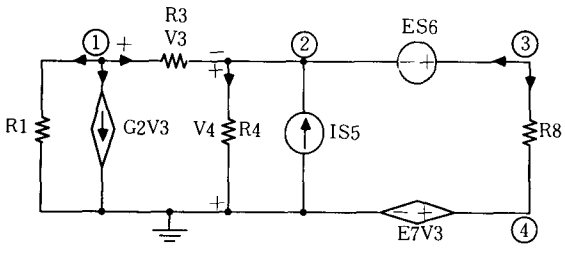


그림 2. 선형 저항 회로

노우드 4 : i7−i8= 0

각 가지 전류를 가지 전압으로 나타낸다.

노우드 1 : 1/R1. 1V1+G2. V3+1/R3. V3= 0

노우드 2 : −1/R3. V3+1/R4. V4−i6=IS5

노우드 3 : i6+1/R8. V8= 0

노우드 4 : −1/R8. V8+i7= 0

여기서 전류 i6와 i7은 대응되는 소자들의 가지 방정식 때문에 가지 전압으로 바뀌어 지지 않으므로 가지 방정식을 위의 수식에다 더 첨가 시킨다.

가지 6 : V6=ES6

가지 7 : V7−E7. V3= 0

KVL을 사용하여 각 노우드 전압으로 모든 가지 전압을 나타낸다.

노우드 1 : 1/R1. e1+G2. (e1−e2)+1/R3. (e1−e2) = 0

노우드 2 : −1/R3. (e1−e2)+1/R4. e2−i6 =IS5

노우드 3 : i6+1/R8. (e3−e4) = 0

노우드 4 : −1/R8. (e3−e4)+i7= 0

가지 6 : e3−e2=ES6

가지 7 : e4−E7. (e1−e2) = 0

위의 6개의 방정식은 4개의 노우드 방정식과 2개의 가지 방정식으로 구성되어 진다. 이러한 6개의 방정식을 행렬 형태로 나타내면 다음과 같다.

$$\begin{bmatrix} Yn & A \\ B & C \end{bmatrix} \begin{bmatrix} e \\ i' \end{bmatrix} =$$

$$\begin{bmatrix} 1/R1+G2+1/R3 & -G2-1/R3 & 0 & 0 & 0 & 0 \\ -1/R3 & 1/R3+1/R4 & 0 & 0 & -1 & 0 \\ 0 & 0 & 1/R8 & -1/R8 & 1 & 0 \\ 0 & 0 & -1/R8 & 1/R8 & 0 & 1 \\ 0 & -1 & 1 & 0 & 0 & 0 \\ -E7 & +E7 & 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} e1 \\ e2 \\ e3 \\ e4 \\ i6 \\ i7 \end{bmatrix}$$

$$= \begin{bmatrix} 0 \\ IS5 \\ 0 \\ 0 \\ ES6 \\ 0 \end{bmatrix}$$

여기서 Yn은 주어진 회로의 노우드 어드미턴스 행렬에 대응하는 것이고 나머지는 가지 방정식과 관계된 계수 행렬이다. 위의 회로에서 저항, 독립 전류원과 VCCS는 NA 방정식과 같이 우변항과 Yn 행렬에만 관계되고 독립 전압원과 VCCS는 MNA 계수 행렬의 나머지에 관계된다. CCCS와 CCVS를 위해서는 자동적으로 회로에 가지가 더 첨가 되어진다.

위의 예에서 보여준 것처럼 MNA는 회로의 노우드 전압 뿐만 아니라 가지 전류도 미지 변수로 취함에 의하여 전압원이나 전류원을 포함하는 모든 회로에 적용시킬 수 있다. 또한 NA에서는 가지 전류를 변수로 취할 수 없음에 의하여 직접 구할 수 없었던 가지 전류를 MNA에서는 직접 구할 수 있다. 그러나 MNA에서는 가지 전류를 변수로 취함에 의하여 계수 행렬의 대각선 항에 "0"이 생기는 경우가 있고 또한 대각선 지배적이지도 않기 때문에 선형 방정식을 풀기 전에 행렬의 열 변환(row interchange)을 행하여 이러한 경우를 없애야 하며 각 전원 전압의 전류를 부가 변수로 취함에 의하여 회로 방정식의 크기가 NA보다 커지게 되는 단점이 있다.

2. 선형 방정식의 해

앞절에서 형성된 선형 회로 방정식은 다음과 같이 나타내어 진다.

$$\begin{aligned} &a11 \cdot x1+a12. x2+\cdots+a1n. xn=b1 \\ &a21. x1+a22. x2+\cdots+a2n. xn=b2 \\ &\quad \vdots \quad \vdots \qquad \vdots \qquad \vdots \\ &an1. x1+an2. x2+\cdots+ann. xn=bn \end{aligned}$$

이와같은 수식을 벡터 방정식으로 나타내면 다음과 같다.

$$A. X=b \qquad (1)$$

여기서 행렬 A는 n×n 행렬이며 X는 n개의 미지 변수 벡터, b는 n개의 값을 갖는 우변항 벡터이다.

회로 시뮬레이터에서 선형 방정식(1)의 해를 구하는데 가장 많이 사용하는 방법은 LU분해에 의하여 해를 얻는 방법이므로 본 절에서는 LU분해 방법에 대하여 설명한다. LU분해는 주어진 계수 행렬 A를

상과 하의 삼각 행렬(lower and upper triangular matrix) L과 U로 각각 분해하여 해를 구한다.

$$A = L \cdot U$$
$$A \cdot x = L \cdot U \cdot x = b \qquad (2)$$

일단 A가 L과 U로 분해되면 첫번째로 $U \cdot x = Y$ 라고 두고 $L \cdot y = b$를 풀어서 y의 값을 구한다음 $U \cdot x = y$를 풀어서 미지 변수 벡터 x를 구한다. 여기서 $L \cdot y = b$를 푸는 것을 전반대치(forward substitution)이라고 하고 $U \cdot x = y$를 푸는 것을 후반대치(backward substitution)이라고 부른다. 이와 같은 과정을 알고리즘으로 나타낸 것이 그림 3에 있다.

LU Decomposition Algorithm
LU Decomposition: Given : A, a nonsingular n×n matrix. Step 0: Set k←1. Step 1: Set the k^{th} row of U equal to the k^{th} row of the reduced matrix $A^{(k)}=[a_{ij}^{(k)}]$. where $A^{(1)}=A$. Thus $u_{kj} \leftarrow a_{kj}^{(k)}$: j=k, k+1, ⋯, n Step 2: If k=n then stop. Step 3: Obtain the k^{th} column of L by setting $l_{ik} \leftarrow a_{ik}^{(ka)}/u_{kk}$: i=k+1, ⋯, n Step 4: Update the reduced matrix from $A^{(k)}$ to $A^{(k+1)}$ by setting $a_{ij}^{(k+1)} \leftarrow a_{ij}^{(k)} - l_{ik}u_{kj}$ where i, j=k+1, k+2, ⋯, n Step 5: Increment k and return to step 1.
Forward Substitution: $y_k \leftarrow b_k - \sum_{j=1}^{k-1} l_{kj}y_j$: k=1, 2, ⋯, n
Backward Substitution: $x_k \leftarrow \frac{y_k - \sum_{j=k+1}^{n} u_{kj}x_j}{u_{kk}}$: k=n, n−1, ⋯, 1

그림 3. LU분해 알고리즘

그러나 실질적으로 시뮬레이터에서 사용할 때는 이와 같은 직접 해석 방법(direct method)을 사용하면서 계산 시간의 단축 및 저장 공간(memory space)을 줄이는 스파스 매트릭스(sparse matrix) 방법을 사용한다. 이와 같은 성긴 선형 방정식(sparse linear squation)을 풀기 위하여 사용하는 다른 방법으로는 이완 방법(relaxation method)이 있다.

이완 방법은 주어진 선형 방정식들은 분해한 다음 각 분해된 방정식을 독립적으로 반복적 과정을 통하여 해를 구하는 방법이다. 반복적 과정은 수렴 조건에 맞을때까지 각 방정식들 사이에 적용되어 진다.

이와 같은 이완 방법은 사실 n개의 변수를 가지는 하나의 큰 선형 방정식 행렬들을 하나의 변수만을 포함하는 n개의 작은 선형 방정식으로 변환하여 해를 구하는 방법이 된다. 그러나, 이완 방법은 수렴할 수 있는 조건에 맞지 않는 선형 방정식은 무한한 반복에 의해서도 해를 구할 수 없다는 단점이 있으므로 일반적인 회로 해석 프로그램에서는 사용하지 않는다.

3. 과도해석(transient analysis) 방정식의 형성

주어진 회로망에 변형 노우드 해석방법을 적용하여 얻어지는 회로 해석 방정식의 일반적인 형태는 식(3)과 같이 비선형 미분 방정식 벡터로 표시된다.

$$f(x, \dot{x}, t) = 0 \qquad (3)$$

식 (3)에서 x는 회로 변수 벡터이고 x의 시간미분 $\dot{x}$는 커패시터나 인덕터와 같은 시미분 소자들에 의하여 나타나게 된다. 식 (3)을 풀기 위하여는 수치 적분에 의하여 전체의 시간 구간을 일정한 간격으로 나누어 초기 시간부터 점진적으로 계산하는 시증분 적분(time incremental numerical integration)이 적용된다. 이와 같은 시증분 적분에는 여러가지가 있지만 가장 기본적인 방법으로는 Euler-Forward(E, F) 방법, Euler-Backward(E. B) 방법과 Trapezoidal 방법이 있다. 이와 같은 3가지 방법을 각각 사용하면 시간 미분 $\dot{x}$는 다음과 같이 표현된다.

Euler-Forward 방법 :

$$\dot{x}_n = \frac{(x_{n+1} - x_n)}{h_n} \qquad (4)$$

Euler-Backward 방법 :

$$\dot{x}_{n+1} = \frac{(x_{n+1} - x_n)}{h_n} \qquad (5)$$

Trapezoidal 방법 :

$$x_{n+1} = x_n + h/2\,(\dot{x}_{n+1} + \dot{x}_n) \qquad (6)$$

여기서 첨자 n, n+1은 계산되는 시점을 표시하며 h는 시간간격(time step)을 의미한다. 이러한 시증분 적분에서 시간간격을 결정하는 것은 결과의 정확도 면에서 매우 중요하므로, 이 결정 방법에는 여러가지 방법이 사용되고 있으나 주로 local truncation error(LTE) 방법을 사용하여 시간 간격을 결정하고 있다.

이 이외에 널리 사용되는 수치적분 방법으로는 Simpson 방법, Gear 방법 등이 있는데 정확도는 위에서 언급한 방법보다 높지만 저장 용량과 계산 시간이 더 많이 든다. 식 (3)에 Euler-Backward 수치 적분 방법을 적용하면 식 (7)과 같은 벡터 방정식을 얻게 된다.

$$f(x_{n+1}, \frac{(x_{n+1}-x_n)}{h_n}, t_{n+1}) = 0 \qquad (7)$$

이와 같은 벡터 방정식을 풀기 위하여 앞절에서 앞절에서 설명한 Gauss 소거법이나 LU 분해방법을 사용하면 각 노우드의 전압과 전류값을 구할 수 있게 된다.

예로서, 시 미분 소자인 커패시터에 Euler-Backward의 수치 적분을 적용하면 다음과 같다.

$$i = C \cdot \frac{dv}{dt}$$

$$i_{n+1} = C \cdot \frac{v_{n+1} - v_n}{h} \Rightarrow i_{n+1} = \frac{C}{h} \cdot v_{n+1} - \frac{C}{h} \cdot v_n$$

이와 같은 수치 적분 방법은 비선형 미분 방정식(식 (3))을 비선형 대수 방정식으로 변환한다. 이러한 비선형 대수 방정식을 매 시간점(time point)에서 선형화 시키는 방법을 사용하여 풀게 된다.

이와 같은 비선형회로를 선형회로로 바꾸는 선형화 방법에는 여러가지가 있지만 일반적으로 수렴 특성이 좋고 실현하기가 간편한 Newton-Raphson(N-R)반복 방법을 많이 사용하고 있다. 식(3)을 비선형 회로 방정식 벡터라고 가정하면 식 (3)에 E. B 수치적분 공식을 적용하여 얻어진 결과는 식 (8)와 같은 비선형 방정식이 된다.

$$f(x_{n+1}, \frac{x_{n+1}-x_n}{h_n}, t_{n+1}) = 0$$

$$\Rightarrow g(x) = 0 \qquad (8)$$

비선형 방정식 (8)을 풀기 위하여 N. R 반복 방법을 적용한다, N. R 반복 방법은 식 (9)에서 보듯이 미지변수 x의 가정치 $\hat{x}$을 중심으로 1차 Taylor 급수 전개하여 얻은 선형 방정식들을 반복적으로 수렴할 때까지 계산하여 실제 해를 구하는 것이다.

$$0 = g(x) = g(\hat{x} + \Delta_x)$$

$$= g(\hat{x}) + \frac{\partial f}{dx} \Big|_{\hat{x}} \cdot \Delta x$$

$$= g(\hat{x}) + J(\hat{x}) \cdot (x - \hat{x})$$

$$\Rightarrow J(\hat{x}) \cdot x = -g(\hat{x}) + J(\hat{x}) \cdot \hat{x} \qquad (9)$$

여기서 J(x)은 g(x)의 1차 미분 벡터 즉, Jacobian 행렬을 의미한다. N. R 반복 방법의 문제점으로는 가정치 $\hat{x}$가 실제 해와 멀리 떨어져 있을 경우 발생하는 비 수렴(nonconvergence)이 있다. 그러나 실제 적분 과정에서는 바로 전 시점에서의 값들이 현시점에 대하여 좋은 가정치가 되므로 이들을 사용하면 대개의 경우 비수렴 문제가 해결된다. 위의 식 (9)에서 우변항은 모든 값을 미리 알고 있는 항들로 되어 있으므로 식 (10)과 같은 선형 방정식이 된다.

$$A \cdot x = b \qquad (10)$$

여기에서 $A = J(\hat{x})$, $b = -g(\hat{x}) + J(\hat{x}) \cdot \hat{x}$ 이다.
식 (10)과 같은 선형 방정식은 앞 절의 선형 방정식의 해석과 같은 Gauss 소거법이나 LU 분해방법을 사용하여 풀면 각 노우드의 전압과 전류값을 구할 수 있게 된다.

현재까지 우리는 회로 해석 방정식 (3)의 해를 구하는 일련의 방법들을 알아보았다. 이러한 방법들은 이미 형성된 식 (3) - (9)에 일괄적으로 적용하기 보다는 회로해석 방정식의 형성 단계에서 해당 소자별로 적용하는 것이 보다 효율적이다. 즉, 수치 적분 방법은 커패시터와 같은 시미분 소자 들에만 적용하고 N.R선형화 방법은 비선형 소자들에게만 적용한다.

이와 같이 각 소자별로 적당한 수치 해석 방법을 적용하는 것을 companion 모델 방법이라고 한다. 주어진 회로망 내의 모든 소자들에 대하여 companion 모델 방법을 적용하면, 그 결과 회로는 선형 회로만으로 구성되므로 컴퓨터로 처리하기가 편해진다.

지금까지 설명한 회로 해석 방법에 있어서 가장 많은 계산 시간을 필요로 하는 부분은 N. R 반복 부분이다. 이와 같은 N. R 반복은 방정식 형성 단계(formulation phase)와 해를 구하는 단계(solution phase)로 나뉘어 지며, 이 두 단계는 수렴 조건에 맞을때까지 매 시간 점에 있어서 반복 되어진다. 여기서 방정식 형성 단계는 앞에서 설명한 Jacobian 행렬을 만드는데 소요되는 시간으로서 각 주어진 회로 소자의 복잡한 방정식을 계산하므로서 많은 시간이 소요되게 된다. 해를 구하는 단계에서는 앞의 방정식 형성에 의하여 만들어진 선형 방정식을 직접 해석 방법(direct method)을 사용하여 푸는 과정으로서 만일 해석 하고자 하는 회로의 크기가 작으면 이 과정은 방정식 형성 시간에 비하여 무시할 수 있으나 회로의 크기가 커지면 이 과정이 회로 해석 시간의 대부분을 차지하게 된다. 이와같은 해석 시간과 회로 크기에 대한 관계를 그림 4에 보였다.

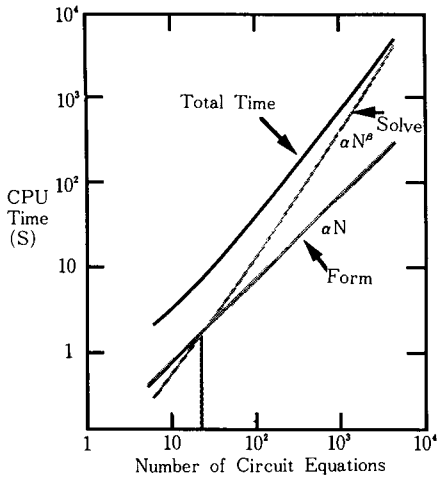


그림 4. 회로 방정식의 크기 대 해석 시간의 비교

Ⅳ. 결 론

이상에서 집적 회로 해석에 필요한 회로 시뮬레이터에서 일반적으로 사용하는 알고리즘에 대하여 간략히 소개 하였다. 앞으로 반도체 산업에서 집적 회로 설계 부분의 비중이 점진적으로 커지고 고집적화 됨에 따라 복잡한 회로 설계의 필요성은 더욱 증가할 것이고, 이에 따라 회로 시뮬레이터의 사용은 더욱 커질 것이다. 이러한 요구에 의해서 앞에서 언급한 것처럼 지금까지 많은 회로 해석 프로그램이 나왔으나, 거의 모든 시뮬레이터에서 사용하는 알고리즘은 앞에서 설명한 방법에 기본을 두고 약간 변형된 것이 대부분이다. 지금까지 많은 회로 설계자들이 이러한 해석 프로그램을 많이 사용하여 왔으나 그 프로그램의 해석 방법에 대하여 알지 못하고 사용하여 왔다고 생각되므로, 이 글이 이러한 설계자들의 설계 능력 향상에 조금이라도 도움이 되기를 바란다.

參 考 文 獻

[1] A. Vladimirescu, A.R. Newton, D.O. Pederson, and A. Sangiovanni Vincentelli, "SPICE Version 2G User's Guide," Univ. of California, Berkeley, Oct. 1980.

[2] D.A. Calahan, *Computer Aided Network* Design, McGraw-Hill, 1972.

[3] S.P. Fan, M.Y. Hsueh, A.R. Newton, and D.O. Pederson, "MOTIS-C: A new circuit simulator for MOS LSI circuits," in IEEE Proc. ISCAS, 1977.

[4] A.R. Newton, "The Simulation of Large-Scale Integrated Circuits," Ph.D. Dissertation, Univ. of California, Berkeley, ERL Memo. ERL-M78/52, July 1978.

[5] P. Yang, I.N. Hajji, and T.N. Trick, "SLATE: A circuit simulation program with latency exploitation and node tearing," in Proc. IEEE Intl. Conf. Circuits and Computers, Port Chester, NY, Oct. 1980.

[6] E. Lelarasmee and A. Sangiovanni-Vincentelli, "RELAX: A new circuit simulator for large scale MOS integrated circuits," Electronic Research Lab., Univ. of California, Berkeley, Memo. UCB/ERL M82/6, Feb. 1982.

[7] D. Dumlugol, "Segmented Waveform Relaxation Algorithms for Mixed-Mode Simulation of Digital MOS VLSI Circuits, Ph.D. Dissertation, Katholieke Univ. Leuven, Oct. 1986

[8] H.Y. Hsieh, A.E. Ruehli, P. Ledak, "Progress on Toggle: A Waveform Relaxation VLSI-MOSFET CAD Program," Proc. of the IEEE Int. Symp. of Circuits and Systems, 1985.

[9] Y.H. Jun, C.W. Lee, K.J. Lee, and S.B. Park, "Timing Simulator by Waveform Relaxation Considering Feedback Effect," Proc. of the IEEE Int. Symp. of Circuits and Systems, 1987.

[10] E. Acuna, J. Dervenis, R. Saleh, "iSPLICE3: A New Simulator for Mixed Analog/Digital Circuits," Custom Integrated Circuits Conference, May 1989.

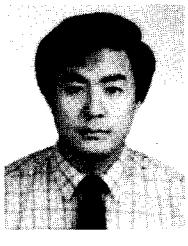
[11] H.E. Tahawy, A. Chianale, B. Wennion, "Functional Verification of Analog Blocks in FIDELDO: A Unified Mixed-Mode Simulation Environment," IEEE Int. Symp. ISCAS-89, pp. 2012-2015, May 1989.

[12] "Advanced statistical analysis program (ASTAP)," Program Reference Manual, Pub. No. SH20-1118-0, IBM Corp. Data Proc. Div., White Plains, NY 10604.

[13] P. Antognetti, D.O. Pederson, H. De Man, *Computer Design Aids for VLSI Circuits*, Sijthoff & Noordhoff, Netherlands, 1981.

[14] B.Carnahan, *Applied Numerical Methods*, John Wiley & Sons, 1969.

筆 者 紹 介



全 永 鉉

1960年 12月 20日生

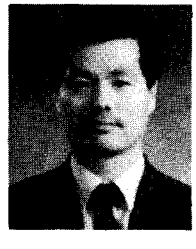
1984年 한양대학교 전자공학과 (공학사)

1986年 한국과학기술원 전기및전자공학과(석사)

1986年 한국과학기술원 전기 및 전자공학과(공학박사)

1989年～1991年 미국 Coordinated Science Lab. 연구원

1991年～현재 (주) 금성일렉트론 반도체연구소 (메모리 설계실)



金 春 慶

1952年 1月 4日生

1974年 서울대학교 전기공학과 (공학사)

1981年 미국 IOWA대학교(석사)

1984年 미국 Minnesota 대학교 (공학박사)

1974年～1979年 국방과학연구소, 연구원

1983年 미국 Metheus Corporation 연구원

1984年～1989年 미국 GE. VLSI Group 연구원

1989年～현재 (주) 금성일렉트론 반도체연구소 CAD 담당이사