論文93-30A-9-9

신경회로망을 이용한 조합 논리회로의 테스트 생성

(Test Generation for Combinational Logic Circuits Using Neural Networks)

金 榮 禹*, 林 寅 七*

(Young Oo Kim and In Chil Lim)

要 約

본 논문에서는 모듈 구조의 신경회로망을 이용하여 조합 논리회로의 stuck-at 고장에 대한 테스트 패턴을 생성하는 새로운 테스트 생성 방식을 제안한다. 하드웨어 기술 언어로 표현된 테스트 대상회로는 신경회로망 컴파일러에 의해 ATPG 신경회로망(Automatic Test Pattern Generation Neural Network)으로 구성된다. 즉, 테스트 대상회로의 각 논리 게이트는 양방향의 이산 Hopfield 네트워크로 이루어지는 신경회로망 모듈로 나타내어지며, 모든 신경회로망 모듈들은 각 모듈의 상태를 인접 모듈로 전달할 수 있도록 서로 연결되어 대상회로에 대한 ATPG 신경회로망을 이룬다. 고장 주입은 해당 신호선에 대응되는 뉴런에 특정 활성치 (Activation Value)를 고정하고, 필요한 신경회로망 모듈간의 연결을 무효화함으로써 이루어진다. 이때 ATPG 신경회로망의 모든 신경회로망 모듈들이 상호작용을 통하여 안정화되면 주어진 주입 고장에 대한 테스트 패턴을 구하게 된다. 제안된 테스트 생성 방식은 대량의 병렬성에 기초를 두고 있어 NP-complete한 것으로 알려진 테스트 생성에 효과적이며, 여러 조합 논리회로를 사용한 실험을 통하여 그 유효성이 입증된다.

Abstract

This paper proposes a new test pattern generation methodology for combinational logic circuits using neural networks based on a modular structure. The CUT (Circuit Under Test) is described in our gate level hardware description language. By conferring neural database, the CUT is compiled to an ATPG (Automatic Test Pattern Generation) neural network. Each logic gate in CUT is represented as a discrete Hopfield network. Such a neual network is called a gate module in this paper. All the gate modules for a CUT form an ATPG neural network by connecting each module through message passing paths by which the states of modules are transferred to their adjacent modules. A fault is injected by setting the activation values of some neurons at given values and by invalidating connections between some gate modules. A test pattern for an injected fault is obtained when all gate modules in the ATPG neural network are stabilized through evolution and mutual interactions. The proposed methodology is efficient for test generation, known to be NP complete, through its massive parallelism. Some results on combinational logic circuits confirm the feasibility of the proposed methodology.

I. 서론

VLSI 기술이 발전함에 따라 단일 칩상에 집적되는 소자수의 증가로 인하여 VLSI 테스트가 큰 문제로

* 正會員, 漢陽大學校 電子工學科
(Dept of Elec. Eng., Hanyang Univ.)
接受日字: 1992年 10月 15日

대두되고 있다. 이것은 많은 소자들이 서로 복잡하게 연결되어 있어서 제한된 수의 외부 핀을 통하여 칩 내부 신호선을 제어하고 관측하는 데 한계가 있기 때문이다. [1]

이러한 문제를 해결하기 위하여 LSSD(Level Sensitive Scan Design) [5] 와 같은 설계 방식이 사용되어, 칩 내부 신호선의 제어 및 관측을 용이하게 하고, 테스트시에 순서 논리회로(sequential logic circuit)를 조합 논리회로(combinational logic circuit)로 변환 할 수 있도록 한다. 그러므로 일반 논리회로에 대한 테스트 문제를 조합 논리회로 테스트 문제로 국한시킬 수 있다.

일반적으로 조합 논리회로에 대한 테스트 패턴 생성(TPG : Test Pattern Generation)은 대상회로(CUT : Circuit Under Test)의 탐색공간(search space)에서 정상회로와 고장회로의 출력이 서로 다르게 되어 고장 여부를 구분할 수 있는 입력 조합을 찾아내는 탐색 문제(search problem)로 생각할 수 있다. [2] 과거 수년간 조합 논리회로에 대한 탐색 속도를 빠르게 하기 위하여 많은 테스트 생성 알고리듬이 제안되고 개선되어 왔으나, [2] NP-complete 문제로 알려진 [3] 테스트 생성 문제를 단일 프로세서를 갖는 컴퓨터로 푸는 것은 여전히 상당한 시간과 계산량이 필요하다. [2,4]

본 논문에서는 대량의 병렬성을 갖는 신경회로망을 이용하여 가정된 고장에 대한 테스트 패턴을 생성하는 새로운 테스트 생성 방식을 제안한다. 테스트 대상회로의 각 논리 게이트는 게이트의 각 입출력 신호선에 대응되는 뉴런과 뉴런 상호간의 연결강도를 갖는 신경회로망 모듈로 모델링될 수 있다. [7,8] 이때 한 신경회로망 모듈의 연결강도(connection weight) 및 뉴런의 임계치(threshold)는 대응 논리 게이트가 취할 수 있는 상태에서 신경회로망 모듈이 에너지 극소점을 가져 안정화될 수 있도록 제반 제한조건들을 고려하여 결정된다.

일반 논리회로에 대한 신경회로망은 각 신경회로망 모듈이 서로 연결된 뉴런간에 일치된 활성치(activation value)를 유지하도록 값을 주고 받을 수 있는 경로를 설정함으로써 구성된다. 각 신경회로망 모듈이 서로간의 제한조건하에 모두 안정화될 때 전체 신경회로망이 안정화되며 각 뉴런의 활성치는 논리회로의 해당 신호선의 신호 값이 된다.

CUT에 대한 자동테스트생성 신경회로망(Automatic Test Pattern Generation Neural Network : 이하 ATPG 신경회로망)은 무고장부(fault-free part), 고장부(faulty part), 입력부(input part), 출력 인터페이스부(output interface part)의 네 부분으로 구성된다. 무고장부는 회로에 고장이 없을 경우의 각 신호선의 상태를 나타내며, 고장부는 회로에 고장이 있을 경우의 신호선의 상태를 나타내도록 한다. 이 두 부분에서는 고장 주입시에 입력부로부터 고장 주입 위치에 고장의 효과가 나타나도록 하며(fault sensitization), 출력측으로 그 효과를 전달하도록 한다(fault propagation). 출력 인터페이스부에서는 출력측으로 고장 전파가 일어날 수 있는 조건을 제공한다.

고장이 주입된 ATPG 신경회로망의 각 모듈들이 서로연결된 모듈간의 제한조건을 유지하면서 안정화될 때 주입 고장에 대한 테스트 패턴을 생성한다. 이때 각 신경회로망 모듈들은 독립적으로 시간발전을 이룰 수 있으므로 대량의 병렬성을 얻을 수 있다. 만약 주입 고장이 검출할 수 없는 고장이라면 고장 주입에 따른 제반 제한조건하에서는 모든 신경회로망 모듈이 안정화되지 못하게 된다. 따라서 본 논문에서는 주어진 시간 내에 모든 신경회로망 모듈을 안정화시키지 못하는 고장은 검출할 수 없는 것으로 간주한다. 본 논문에서 제안한 신경회로망을 이용한 테스트 생성 방식은 SUN4 SPARC station1에서 C 언어로 구현하며, 여러 예제 회로에 적용하여 그 유효성을 입증한다.

Ⅱ. 게이트 모듈

논리 게이트의 각 신호선은 뉴런으로 표현될 수 있으며, 이때 뉴런의 활성치는 해당 신호선의 신호 값을 나타낸다. [7,8] 본 논문에서는 각 논리 게이트를 이산 Hopfield 네트워크 [10] 로 표현한 것을 게이트 모듈(gate module)이라 부르기로 한다.

뉴런의 활성치는 -1(논리 0), +1(논리 1), 0(무정의조건 : don't care condition)의 세가지 값 중 하나를 갖는 것으로 하며, 초기에는 게이트 모듈의 각 뉴런의 활성치는 모두 0인 것으로 한다. 각 게이트 모듈은 외부에서 뉴런의 활성치를 1이나 +1로 변화시키도록 자극받지 않는 한 시간발전을 하지 않고 모든 뉴런의 활성치는 0값을 유지한다. 하나의 게이트 모듈이 외부에 의해 변화되면 모듈 내의 각 뉴런은 다음과 같은 시간발전 규칙에 의해 활성치가 변화하여 모듈을 에너지 기저 상태로 안정화시킨다.

$$net_i = \sum_{j \neq i} T_{ij} V_j + \theta_i \begin{cases} > 0\text{일 때 } V_i(t+1) = +1, \\ < 0\text{일 때 } V_i(t+1) = -1, \\ = 0\text{일 때 } V_i(t+1) = V_i(t). \end{cases} \quad (1)$$

여기에서 V_i는 i번째 뉴런의 활성치, θ_i는 i번째 뉴

런의 임계치, T_{ij}는 j번째 뉴런에서 i번째 뉴런으로의 연결강도를 나타낸다.

하나의 게이트 모듈은 여러개의 일치상태(consistent state)를 가질 수 있다. 예를 들면 AND 게이트 모듈의 경우는 ((X2, X3, X1) : (-1, -1, -1), (-1, 0, -1), (-1, 1, -1), (0, -1, -1), (1, -1, -1), (1, 1, 1))의 6가지 일치상태가 있다. 게이트 모듈을 일치상태에서 안정화시키기 위해서는 일치상태에서 에너지 함수가 기저상태가 되어야 하며 일치상태가 아닌 상태에서는 그보다 높은 에너지를 가져야 한다. 에너지 함수는 식 (2)와 같다. [10,11]

$$E=-\frac{1}{2}\sum_{i}^{N}\sum_{j\neq i}^{N}T_{ij}V_iV_j-\sum_{i}^{N}I_iV_i+K \tag{2}$$

여기에서 N 은 신경회로망 내의 뉴런의 수, I_i는 뉴런 i의 외부 바이어스, K는 상수를 나타낸다.

각 게이트 모듈에 대한 퍼래미터들은 결정 하이퍼평면(decision hyperplane)이 각 뉴런에서 존재하여야 하며, 일치상태에서만 에너지 기저상태를 가지도록 하는 제한조건하에서 결정된다. [7,8]

그림 1은 AND 게이트 모듈에 대한 퍼래미터의 예를 보인 것이다. 각 퍼래미터는 다음과 같은 과정을 통하여 얻어진다.

뉴런 2와 3이 서로 대칭인 조건으로부터

$$T_{12}=T_{13} \tag{3}$$
$$I_2=I_3 \tag{4}$$

을 얻을 수 있다. AND 게이트의 일치상태를 (X2, X3, X1)로 표시하면 (1, 1, 1), (-1, d, -1), (d, -1, -1)의 세가지 일치상태를 갖는다. 여기에서 d는 무정의조건(don't care condition)이며, -1, 0, +1의 어느 값도 가질 수 있음을 나타낸다. 뉴런 1로 들어오는 모든 신호의 합은 식(1)에 의해

$net_1 = T_{12}(X2 + X3) + I_1$

이므로 뉴런 1에서 하이퍼 평면이 존재하기 위한 조건으로부터 일치상태 (1, 1, 1)에 대하여

$2T_{12} + I_1 > 0$

일치상태 (-1, d, -1)에 대하여

$T_{21}(-1+d)+I_1 < 0$

이 된다. 따라서 위의 두 식으로부터

$$-2T_{21} < I_1 < 0 \tag{5}$$
$$T_{21} > 0 \tag{6}$$

임을 알 수 있다. 뉴런 2로 들어오는 모든 신호의 합은

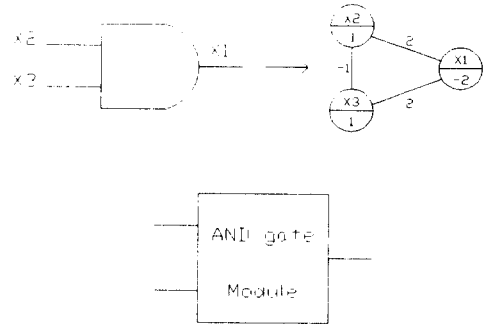


그림 1. AND 게이트 모듈의 예
Fig. 1. An example of AND gate module.

$net_2 = T_{12}X1 + T_{23}X3 + I_2$

이 되므로, 뉴런 2에서 하이퍼 평면이 존재하기 위한 조건으로부터 (1, 1, 1)에 대하여

$T_{12} + T_{23} + I_2 > 0$

(-1, d, -1)에 대하여

$-T_{12} + T_{23}d + I_2 < 0$

이 된다. 위의 두 식으로부터

$$T_{12} > -T_{23} \tag{7}$$
$$I_2 < T_{12} \tag{8}$$

을 얻을 수 있다. (1, 1, 1)및 (-1, d, -1)에서 같은 기저 에너지 값을 갖도록 하기 위하여

$$E=-\frac{1}{2}\sum_{i}^{3}\sum_{j\neq i}^{3}T_{ij}V_iV_j-\sum_{i}^{3}I_iV_j+K$$
$$=-(T_{12}X1X2+T_{13}X1X3+T_{23}X2X3+I_1X1+I_2X3)+K$$
$$-(2T_{12}+T_{23}+I_1+I_2+I_3)+K$$
$$=-(T_{12}-T_{13}d-T_{23}d-I_1-I_2+I_3d)$$
$$d(T_{12}d+T_{23}-I_3)+T_{12}+T_{23}+2I_1+2I_2+I_3=0$$

의 식이 d의 값에 무관하게 성립하여야 하므로

$$T_{12} + T_{23} - I_3 = 0 \tag{9}$$
$$T_{12} + T_{23} + 2I_1 + 2I_2 + I_3 = 0 \tag{10}$$
$$I_1 + 2I_2 = 0 \tag{10}$$

이 된다. (3), (4), (5), (6), (7), (8), (9), (10)' 에서 $I_2=k$, $T_{32}=l$로 놓으면

$$I_2 = I_3 = k > 0 \tag{11}$$

$$I_1 = -2k \quad (12)$$

$$T_{32} = I < 0 \quad (13)$$

$$T_{21} = T_{31} = k - I \quad (14)$$

의 조건이 구해진다. 그림 1에 보인 AND 게이트 모듈에 대한 퍼래미터는 k=1, I= 1인 경우의 예이다. 이와 같이 얻어진 AND 게이트 모듈은 각 뉴런의 활성치가 일치상태일 때 각 net값이 그 뉴런의 활성치를 변화시키지 않는 범주에 속하게 된다.

예를 들어 그림 1의 AND 게이트에서 처음 상태 (X2, X3, X1)이 일치상태인 (-1, 1, 1)이라면, $net_1 = -2 < 0$, $net_2 = -2 < 0$, $net_3 = 0$이므로 이 상태를 유지하게 되며, 불일치상태인 (-1, 1, 1)이라면, $net_1 = 2 > 0$, $net_2 = net_3 = -2 < 0$이므로 뉴런 1, 2, 3중 임의의 뉴런의 활성치가 바뀌고 다시 각 net값이 계산되어 일치상태가 될 때까지 상태의 변화가 계속된다.

다른 게이트 모듈에 대한 퍼래미터는 AND 게이트 모듈의 경우와 같은 방법에 의해 구할 수 있으며 그림 2는 본 논문에서 사용하고 있는 각 게이트에 대한 퍼래미터의 예를 보인 것이다. 본 논문에서는 그림 2에 보인 것과 같은 2입력 게이트에 대한 게이트 모듈을 사용하며, 3입력 이상인 게이트들은 2입력 게이트의 조합으로 구성한다.

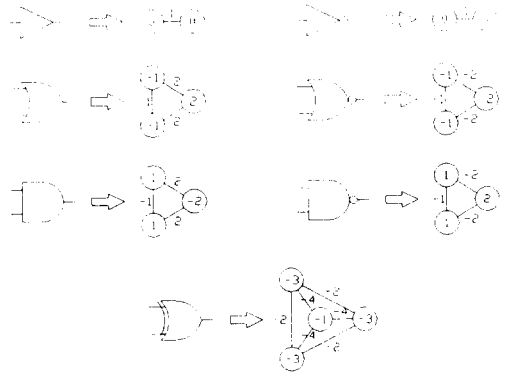
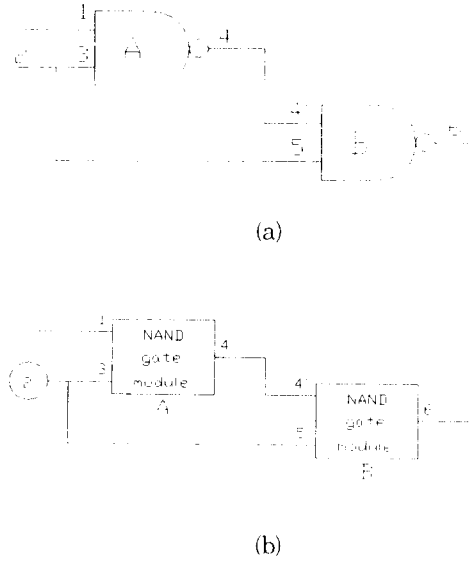


그림 2. 각 게이트 모듈의 예
Fig. 2. Examples of gate modules.

Ⅲ. 논리회로에 대한 신경회로망

그림 3(a)의 논리회로에서는 6개의 고장 가능 위치가 있다. Fan-out이 이루어진 2, 3, 5의 신호선은 각각 별도의 고장이 존재할 수 있다. 이것을 고려하기 위하여 본 논문에서는 신호선 2, 3, 5를 각각 다른 뉴런에 할당한다. Fan out에 대한 고려를 일반화하면 fan out이 일어나지 않은 4와 4'의 신호선에 대해서도 각각 다른 뉴런을 할당할 수 있다. 이때 뉴런간의 연결선은 상호 뉴런의 활성치를 동일한 값으로 유지하기 위하여 활성치를 전달하는 경로로 한다. 즉, 그림 3(a)의 논리회로에 대한 신경회로망은 그림 3(b)와 같이 구성한다. 여기에서 각 게이트 모듈의 내부에서는 (1)의 식과 같은 시간발전 규칙이 적용되며, 모듈간의 연결선은 연결된 뉴런간에 활성치를 동일하게 유지하기 위한 전달 경로로서 작용한다. 이렇게 함으로써 신호선 2, 3, 5는 각각 별도의 고장이 존재할 수 있는데 이에 대해서는 제5절에서 설명한다.



(a)

(b)

그림 3. (a) 논리회로의 예
(b) 대응 신경회로망
Fig. 3. (a) An example of logic circuit,
(b) The corresponding neural networks.

일반적으로 대상 논리회로에 대한 신경회로망은 각 논리 게이트에 대응하는 게이트 모듈과 대상 논리회로의 연결 상태와 동일하게 게이트 모듈간을 연결한 연결 경로로써 구성된다. 만약 하나의 게이트 모듈이 시간발전 규칙에 의해 안정화되면 그 모듈 내의 각 뉴런의 활성치는 모듈 외부의 활성치 전달 경로를 통해 다른 모듈의 뉴런으로 전달되어 그뉴런의 활성치를 동일하게 유지하도록 한다.

활성치를 전달받은 뉴런의 활성치가 변화하였다면

그 뉴런을 포함한 게이트 모듈은 다시 이 조건하에서 시간발전을 하게 된다. 즉 안정화된 모듈의 각 뉴런 중 그 활성치가 변화한 뉴런은 외부 전달 경로를 통해 인접 뉴런의 활성치를 변화시키고, 그 모듈은 시간 발전을 하게 된다. 이와 같이 시간발전과 활성치 전달이 계속되어, 외부 전달 경로로 연결된 뉴런간의 활성치를 같은 값으로 유지하면서 모든 모듈이 에너지 기저 상태가 되면 일반 논리회로에 대한 전체 신경회로망이 안정화된다. 이때의 각 뉴런의 활성치는 대상 논리회로내 각 신호선의 신호값이 된다. 전체 신경회로망의 에너지는 각 모듈들의 에너지의 합이 된다.

Ⅳ. ATPG 신경회로망

회로 내에 발생한 고장은 어떤 입력 패턴에 대하여 고장이 없는 회로에 의한 출력과 고장이 있는 회로의 출력이 서로 같지 않을 때 외부에서 검출될 수 있다. 이때의 입력 패턴을 테스트 패턴이라 하며, 테스트 패턴을 발생시키는 과정을 테스트 생성이라 한다. 일반적으로 테스트 생성 과정은 고장활성화(fault sensitization) 과정과 고장전파(fault propagation) 과정으로 이루어진다.[6] 고장활성화는 고장이 가정된 신호선에 고장 신호를 만들어 내는 것이며, 고장전파는 생성된 고장을 주출력(primary output)까지 전파되도록 하는 것이다. 본 논문의 방식에서는 고장활성화와 고장전파가 원활히 일어날 수 있도록 ATPG 신경회로망을 구성하여 테스트 패턴을 생성한다.

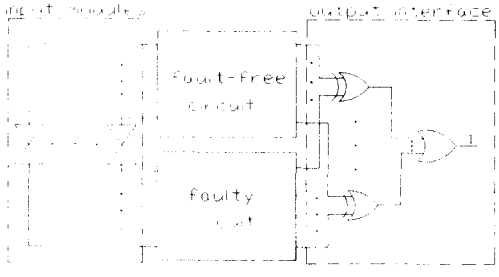


그림 4. ATPG 신경회로망의 구조
Fig. 4. The structure of ATPG neural network.

ATPG 신경회로망의 일반적인 구조는 그림 4와 같다. 무고장부(fault-free part)는 내부의 모든 모듈이 안정화되면 고장이 없는 상태에서의 각 신호선의 신호값이 내부 뉴런들의 활성치로 나타나게 되며, 고장부(faulty part)는 고장이 있는 상태에서의 각 신호선의 값이 뉴런들의 활성치로 나타나게 된다. 이 두 부분의 주출력 신호 중 최소한 하나 이상의 신호가 다르게 나타나야 가정된 고장에 대한 테스트 패턴을 찾을 수 있으며, 출력인터페이스부에서 이 조건을 제공해준다.

출력인터페이스부는 그림 4에 나타난 바와 같이 고장부와 무고장부의 주출력 쌍을 XOR하여 그 중 최소한 하나가 1값이 되도록 XOR 게이트의 모든 출력을 OR 할 때 1값을 가지도록 구성한다. 입력부는 고장부와 무고장부 간의 활성치 전달을 중계하며, 동시에 대상 고장에 대한 테스트 패턴을 나타나게 한다.

Ⅴ. 고장주입 및 테스트 생성

하나의 신호선에 고장을 가정하면 그 고장은 ATPG신경회로망의 대응 뉴런들의 활성치를 고장활성화를 위한 값으로 고정시키고, 고장부에서의 고정된 활성치를 역방향(backward direction)으로 전달되지 않도록 활성치 전달 경로를 차단함으로써 ATPG 신경회로망에 주입된다. 정상적인 동작을 할 때는 가정된 고장의 반대 값(s-a-0이면 +1, s-a-1이면 -1)이 고장 가정 신호선에 나타나야 하며, 고장값은 고장시에 나타난다. 따라서 고장의 반대 값을 무고장부의 대응 뉴런의 활성치로 고정하고, 고장 값을 고장부의 대응 뉴런의 활성치로 고정한다. 또한 논리회로의 신호 전달은 단방향성이므로 고장이 발생하여도 그것은 역방향으로는 전달되지 않으므로, 이를 나타내기 위해

고장부에서는 고장이 주입된 뉴런의 역방향 활성치 전달 경로를 차단한다.

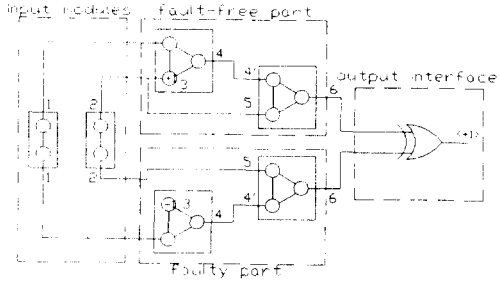


그림 5. 그림 3(a)에 대한 초기 ATPG 신경회로망
Fig. 5. Initial ATPG neural network for Fig. 3(a).

그림 5는 그림 3(a)의 논리회로에서 신호선 3에 s-a-0 고장을 가정했을 때의 ATPG 신경회로망의 초기 상태이다. 무고장부의 뉴런 3에는 고장의 반대 값인 +1을 활성치로 고정하고, 고장부의 뉴런 3에는 고장

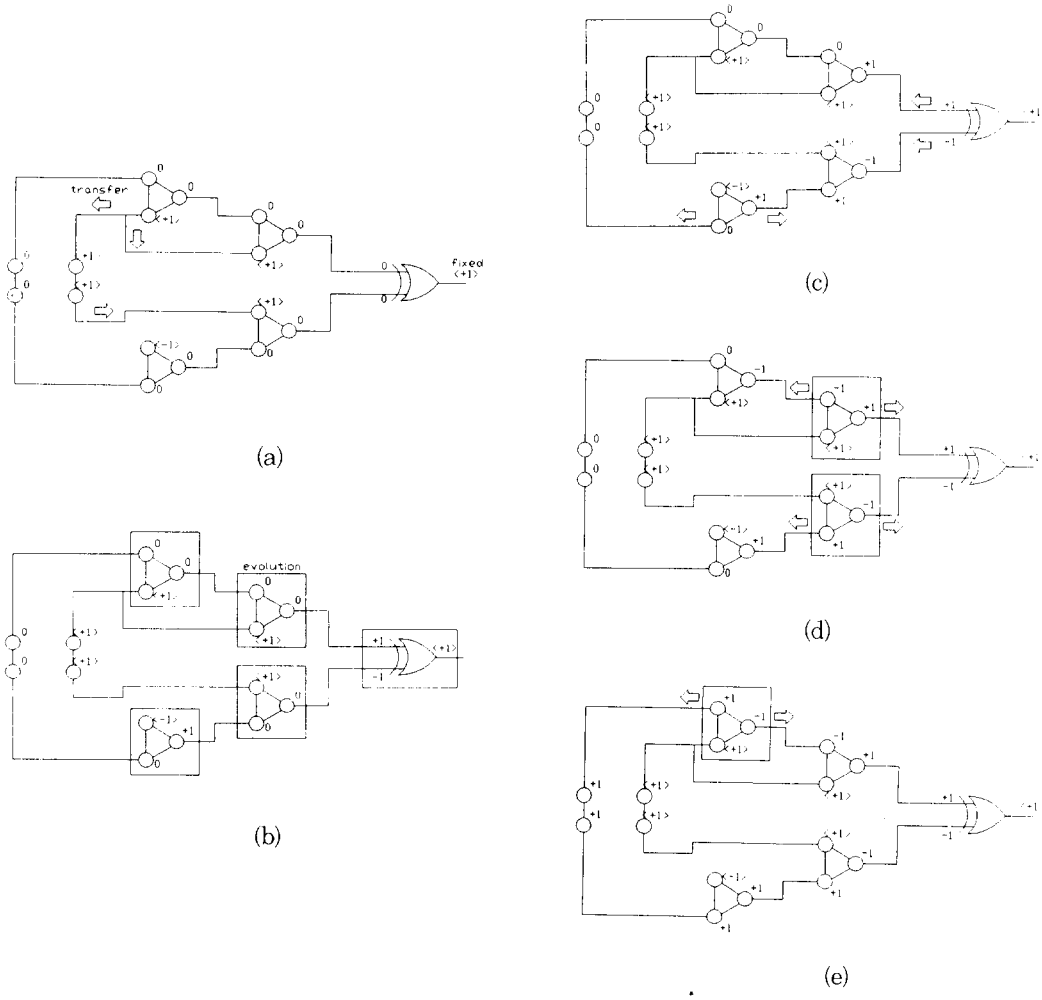


그림 6. 그림 3(a)의 논리회로에서 신호선 3에 s-a-0 고장이 발생하였을 경우 대응 ATPG 신경회로망이 테스트 패턴을 생성하는 과정

Fig. 6. The test pattern generation process for stuck-at-0 fault on signal line 3 of the logic circuit in Fig. 3(a) by ATPG neural network.

값인 -1을 활성치로 고정하였다. 또한 고장부의 뉴런 3에서의 역방향 활성치 전달 경로는 차단하였다. 이 논리회로는 주출력이 하나이므로 출력인터페이스부에서는 무고장부와 고장부의 출력을 XOR하고 그 출력을 +1로 고정한다.

고장 주입에 의해 ATPG 신경회로망이 초기상태로 정립되면, 각 모듈들은 제한조건하에서 시간발전을 통해 안정화 상태로 진행한다. 여기에서 제한조건이라 함은 활성치가 고정된 뉴런은 그 값을 유지하면서 활성치 전달 경로로 연결된 뉴런들이 모두 같은 값을 가져야한다는 것을 의미한다. ATPG 신경회로망의 시간발전 규칙은 다음과 같다.

(1) 초기 상태의 ATPG 신경회로망 각 뉴런의 활성치는 고장이 주입된 뉴런을 제외하고는 모두 0으로 한다.

(2) 활성치 전달 경로로 연결된 뉴런들은 모두 같은 활성치를 가져야 한다. 즉, 한 모듈 내의 뉴런이 활성치가 변화되면 변화된 활성치는 활성치 전달 경로를 통해 인접 뉴런들로 전달된다.

(3) 한 모듈 내의 뉴런들의 활성치중 최소한 하나 이상이 기존의 값에서 변화되면 그 모듈은 시간 발전을 한다. 시간발전 규칙은 식(1)에 따른다.

(4) ATPG 신경회로망 내의 모든 모듈이 안정화되

면 ATPG 신경회로망은 시간발전을 멈추고 테스트 패턴을 발생한다.

그림 3(a)의 신호선 3에 s-a-0를 가정하였을 경우 신호선 3에 대응하는 뉴런에 각각 활성치를 고정하고 그 값을 인접 뉴런에 전파한 ATPG 신경회로망은 그림 6(a)와 같다. 이 경우 시간발전을 하여야 할 신경회로망 모듈은 고장부의 4번 게이트, 무고장부의 4번 게이트, 고장부의 5번 게이트, 출력 인터페이스 등이 된다. 이러한 게이트들이 시간발전을 하여 안정화하면 그림 6(b)와 같이 된다. 이때 무고장부 4번 게이트, 무고장부의 5번 게이트, 고장부의 5번 게이트 등은 시간발전 이전의 뉴런 상태가 이미 안정화된 상태이므로 그 활성치들이 변화하지 않는다. 또한 출력 인터페이스부의 XOR 게이트는 그 입력 뉴런의 활성치가 각각 +1, -1로 되거나 -1, +1로 될 수 있는데 그 선택은 랜덤 함수에 의해 이루어진다. 상태의 변화가 생긴 모듈인 고장부 4번 게이트와 출력 인터페이스 게이트는 그 상태를 인접 모듈로 전파하여 그림 6(c)와 같이 된다. 이 경우 시간발전을 요하는 신경회로망 모듈은 무고장부 6번 게이트와 고장부 6번 게이트이다. 이들이 시간발전을 이루고 그 변화된 활성치를 인접 모듈로 전파하면 그림 6(d)와 같이 된다. 이러한 과정을 거쳐 최종적으로 그림 6(e)와 같이 되면 모든 모듈이 안정화되므로 테스트 패턴은 +1, +1(논리값으로 1, 1)이며, 고장이 없을 경우에는 +1(논리 1) 값이 출력되고 3번 신호선에 s-a-0 고장이 발생하였을 경우에는 -1(논리 0)값이 출력함을 알 수 있다.

만약 가정된 고장이 검출될 수 없는 고장(undetectable fault)이라면 제한조건하에서는 모든 모듈이 안정화될 수 없다. 그러므로 이 경우에는 ATPG 신경회로망은 위의 시간 발전을 무한 반복하게 된다. 따라서 본 논문의 방식에서는 일정 횟수나 일정 시간 동안의 시간발전을 통해 ATPG 신경회로망이 안정화하지 않으면 가정된 고장은 검출할 수 없는 것으로 간주한다. 예를 들어 그림 5에서 신호선 3의 s-a-1 고장은 검출할 수 없는 고장이며, ATPG 신경회로망은 주어진 제한조건하에서는 안정화되지 않는다. 따라서 이 고장은 일정한 반복이 진행된 후 검출될 수 없는 고장으로 간주된다.

Ⅵ. 구현 및 결과

본 논문에서 제안된 ATPG 시스템은 SUN4 SPARC station1에서 C 언어로 구현되었다. 그림 7은 구현된 ATPG 시스템의 흐름도이다.

CUT는 하드웨어 기술 언어로 기술되어 ATPG 시스템의 입력으로 제공된다. 하드웨어 기술 언어는 YACC[14]를 이용하여 구성하였다. 그림 8(b)는 그림 8(a)의 CUT에 대한 회로기술이다.

CUT에 대한 고장리스트나 고장 주입 위치 및 여타의 정보들은 고장주입 단계나 고장 시뮬레이션 단계에서 사용된다. 신경회로망 데이타베이스에는 2절에서 다룬 게이트모듈 등의 신경회로망 모듈들이 저장되며, 신경회로망 컴파일러에서는 CUT에 대한 회로기술을 신경회로망 데이타베이스를 참조하여 ATPG 신경회로망으로 구성한다.

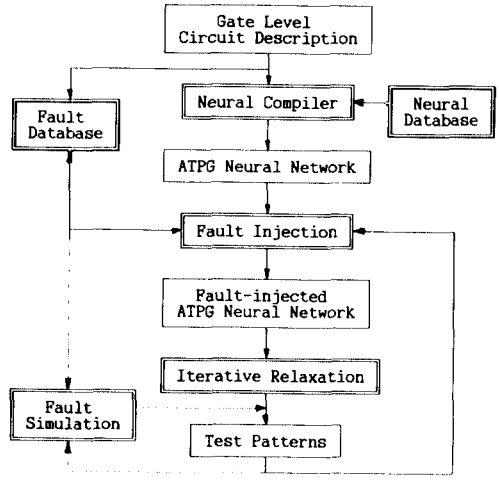
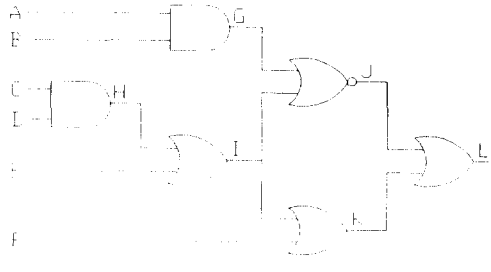


그림 7. ATPG 시스템의 흐름도
Fig. 7. The flow of ATPG system.

ATPG 시스템은 구성된 ATPG 신경회로망에 고장을 주입하고, 5절에서 기술된 바와 같은 반복적 시간 발전을 통해 테스트 패턴을 생성한다. 하나의 테스트 패턴이 생성되면 가정된 고장은 고장 데이타베이스에서 제외되며, 또한 고장 시뮬레이션을 통해 동일한 테스트 패턴으로 검출할 수 있는 여타의 고장을 찾아 고장 대상에서 제외한다. 모든 대상 고장이 고려되면 테스트 생성은 멈추게 된다.



(a)

```
# C14 circuit : example in [5]
C14(A, B, C, D, E, F : OUT_L)
{
        AND
                G(A, B : J)
                H(C, D : I);
        OR
                I(H, E : J, K)
                K(I, F : L)
                L(J, K : OUT_L);
        NOR
                J(G, I : L);
}
```

(b)

그림 8. (a) 예제 회로[2], (b) 회로 기술
Fig. 8. (a) An example circuit,
(b) its circuit description.

표 1은 ECAT 회로[15], Full adder, Schenider circuit[16], SN7480 gated full adder[9], C17[9], C14[2], 2비트 ALU, BCD-to-Excess 3 converter 등을 본 시스템에 적용한 결과이다.

표 1. 실험 결과
Table 1. Experimental results.

Circuits	ECT	F/A	SCH	SN7	C17	C14	ALU2	BCD
# inputs	6	3	4	9	5	6	8	4
# outputs	1	2	1	3	2	1	3	4
# signals	23	16	25	49	17	14	105	28
# gate modules	59	28	27	64	24	21	107	41
# total faults	46	32	50	98	34	28	210	56
# faults detected	38	32	43	98	34	27	209	56
# undetectable faults	8	0	7	0	0	1	0	0
# test patterns	5	25	9	38	19	8	140	37
ave. time/fault (sec)	2.49	0.03	0.33	0.43	0.03	0.06	17.8	0.14
ave. time/detected	0.16	0.03	0.08	0.43	0.03	0.02	13.6	0.14
fallut coverage (%)	100	100	100	100	100	100	99	100

cf: ECT ECAT circuit[15]
F/A Full adder circuit,
SCH Schenider circuit[16],
SN7 SN7480 gated full adder[9],
ALU2 A 2-bit Arithmetic Logic Unit,
BCD BCD to excess 3 code converter

Ⅶ. 결 론

본 논문에서는 모듈 구조를 갖는 신경회로망을 이용하여 디지탈회로의 논리회로 테스트에 필요한 테스트 패턴을 생성하는 새로운 방식을 제안하였다. 각 모듈들은 양방향의 이산 Hopfield 네트워크로 구성하며, 각 모듈간의 활성치를 일치시키기 위한 활성치 전달 경로를 설정하여 ATPG 신경회로망을 구성하였다. ATPG 신경회로망은 고장주입 및 시간발전을 통하여 각 신경회로망 모듈들이 안정화되도록 진행함으로써 고장활성화 및 고장전파 과정을 수행하여 테스트 패턴을 생성하였다.

제안된 테스트 생성 방식은 C 언어로 구현하여 여러 예제 회로에 적용함으로써 그 유효성을 입증하였다.

앞으로의 연구 과제는 대량의 병렬성을 얻기 위한 하드웨어 구현 및 뉴런의 수를 줄이기 위한 연구 등이다. 또한 테스트 패턴을 찾는 시간을 단축하기 위허 고장합병(fault collapsing), 지배자(dominator) 등의 개념을 도입한 전처리부에 대한 연구가 이루어져야 하겠다.

參 考 文 獻

[1] T. W. Williams and K. P. Parker, "Design for Testability A Survey," *Proc. IEEE*, vol. 71, 98-112, Jan. 1983.

[2] H. Klenke et al. "Parallel-Processing Techniques for Automatic Test Pattern Generation," *Computer*, vol. 25, no. 1, pp. 71-84, Jan. 1992.

[3] O. H. Ibrra and S. K. Shani, "Polynomially Complete Fault Detection Problems," *IEEE Trans. on Computers*, pp. 242-249, March 1985.

[4] H. Fujiwara and T. Shimono, "On The Acceleration of Test Generation Algorithms," *IEEE Trans. Computer* vol. C-32, pp. 1137-1144, Dec. 1983.

[5] E. B. Eichelberger and T. W. Williams, "A logic design structure for LSI testability," Proc. 14th Design Automation Conf., pp. 462-468, June 1977.

[6] T. Kirkland and M. R. Mercer, "Algorithms for Automatic Test Pattern Generation," *IEEE Design & Test*, pp. 43-55, June 1988.

[7] S. T. Chakradhar et al. "Automatic Test Generation Using Neural Networks," Proc. *IEEE Int'l Conf. Computer-*

Aided Design, pp. 416-420, 1988.

[8] S. T. Chakradhar et al. "Toward Massively Parallel Automatic Test Generation," *IEEE Trans. on Computer-Aided Design*, pp. 981-994, Sep. 1990.

[9] Y. H. Kim and I. C. Lim, "An Efficient Test Generation Algorithm for Digital Logic Circuits," Journal of KITE, vol. 28 A, no. 2, pp. 83-92, Feb. 1991.

[10] J. J. Hopfield, "Neural Networks and Physical Systems with Emergent Collective Computional Abilities," *Proc. Nat 'l. Acad.* Sci. USA, vol. 79, pp. 2554-2558, April 1982.

[11] J. J. Hopfield, "Neurons with Graded Response Have Collective Computational Properties Like Those of Two-State Neurons," *Proc. Nat, l. Acad. Sci.* USA, vol. 81, pp. 3088-3092, May 1984.

[12] J. J. Hopfield and D. W. Tank, "'Neural" Computation of Decisions in Optimization Problems," Biological Cybernetics, Vol. 52, pp. 158-159, 1985.

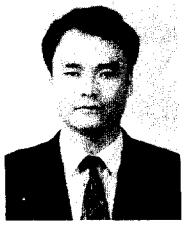
[13] R. P. Lippmann, "An Introduction to Computing with Neural Nets," *IEEE ASSP Magagine*, pp. 4-22, April 1987.

[14] A. T. Schreiner et al. Introduction to Compiler Construction with UNIX, Prentice-Hall, 1985.

[15] H. Fujiwara, Logic Testing and Design for Testability, The MIT Press, 1985.

[16] R. G. Bennetts, Design of Testable Logic Circuits, Addison-Wesley, 1984.

著 者 紹 介



金 榮 禹(正會員)

1962年 2月 18日生. 1981年 3月 ~ 1985年 2月 한양대학교 전자공학과(공학사). 1985年 3月 ~ 1987年 2月 한양대학교 전자공학과 (공학석사). 1989年 3月 ~ 현재 한양대학교 대학원 전자공학과 (박사과정). 주관심분야는 신경회로망, 논리회로 테스트, Broadband ISDN 교환제어, CAC(Call Admission Control) 및 routing 등임.

林 寅 七(正會員) 第 30卷 B編 第 2號 參照

현재 한양대학교 전자공학과 교수