

論文98-35C-4-1

조합논리회로의 고장 검출을 개선을 위한 회로분할기법

(Circuit Partitioning to Enhance the Fault Coverage for Combinational Logic)

盧貞鎬*, 金相鎭**, 李昌熙**, 尹泰鎭**, 安光善**

(Jeong-Ho Noh, Sang-Jin Kim, Chang-Hee Lee, Tae-Jin Yun, and Gwang-Seon Ahn)

요 약

검출 불가능한 고장을 가지는 조합논리회로의 분할문제에 대하여 본 논문에서는 고장 검출율을 높이는 BPT(Bipartition for Testability) 분할알고리즘을 제안한다. 검출 불가능한 고장을 가진 논리회로는 여분 회로, 분지-재수렴회로, 그리고 구조적으로 동등한 회로이다. BPT알고리즘은 검출불가고장 위치를 고려하여 절단되는 신호선을 정하므로 고장검출율을 향상시킨다. 또한 신호선이 절단될 때 이 신호선을 통하여 제어성이나 관찰성이 향상됨을 이용한다. 알고리즘은 회로의 초기 분할에서 각 노드를 다른 부분으로 이동시킬 때 절단되는 신호선 수의 변화와 노드의 이동에 의해 검출가능해지는 검출 불가능한 고장의 수에 바탕을 둔 이득값을 이용하여 이동시킬 노드를 선택한다. 제안된 알고리즘을 ISCAS85 벤치마크회로에 대해 실험한 결과 일반적인 이분 분할 기법에 비해 절단되는 신호선의 수는 다소 증가하였으나 고장 검출율은 향상됨을 보인다.

Abstract

Partitioning problem of large combinational logic has been studied in real world. Most of logic include undetectable faults from the structure of it's redundant, fan-out-reconvergent, and symmetrical feature. BPT algorithm is proposed to enhance the fault coverage for combinational logic partitioning. This algorithm partitions the logic by cut the lines related to undetectable structure when seperating. Controllability and observability are considered in the process of partitioning. This algorithm is evaluated effective by testing ISCAS85 circuits.

I. 서 론

반도체 제작 기술의 발전으로 칩의 집적도가 10^8 의 트랜지스터 크기로 증가하고 있으며, 다양한 기술에 의한 시스템의 설계가 실현되고 있다. 근래에는 반주 문용 반도체인 FPGA(Field Programmable Gate

Array) 그리고 여러 종류의 IC를 하나의 칩에 구현하는 MCM(multi-chip module)의 연구가 활발하다. 대용량의 FPGA를 기반으로 하는 시스템은 라이프 사이클이 빠른 시제품 생산에 널리 사용되며, 대용량의 MCM은 기판상의 집적도를 높이고 신호 지연의 감소, 그리고 낮은 열 방출의 장점을 가지므로 여러 분야의 시스템 구현에 사용되고 있다. 그러나 디지털 시스템이 대형화됨에 따라 하드웨어가 복잡하게 되어 시스템의 고장 발생 확률이 높아지고 있으며, 높은 신뢰도를 요구하는 대형화 시스템의 개발이 증가됨에 따라 고장을 검출하고 진단하는 것은 매우 중요한 일이 되었다.

* 正會員, 國防科學研究所

(Agency for Defense Development)

** 正會員, 慶北大學校 컴퓨터工學科

(Kyungpook Nat'l University, Dept. Computer Eng.)

接受日字:1997年8月29日, 수정완료일:1998年3月31日

설계에 있어서 신뢰도를 높이기 위해서 대용량의 회로를 분할하는 연구는 1972년부터 활발하게 진행되어 왔으며^[1], 관련연구로는 분할되는 에지의 가중치를 최소화할 수 있도록 하면서 균형화된 분할을 해나가는 2분분할법(2-way partitioning)^[2], 각 소자의 크기와 핀 수의 최대 최소 값에 제한을 가하여 두고 이것을 만족시키는 범위 내에서 각 소자 사이의 연결을 최소화 할 수 있도록 하는 다중분할(multi-way partitioning)^[3,4], 소자간의 연결선의 수 보다 시스템의 성능을 최대화하기 위한 성능지향분할(performance-driven partitioning)^[5], 소자의 배치와 전체적인 모양의 갖추기 위해서 배선의 길이를 최소화 할 수 있게 하는 레이아웃지향분할(layout-driven partitioning)^[6], 시스템의 성능을 향상시키고 핀수와 선의 길이를 줄이기 위해 복사기법을 병행하는 반복분할(partitioning with replication)^[7] 등이 있다.

디지털시스템은 일반적으로 조합논리회로와 순서논리회로로 구성되며, 본 연구에서는 조합논리를 분할 설계하는 경우, 회로내의 고장이 있는 부분을 중심으로 분할하는 방법을 제시한다. 대상 고장은 단일고착고장(s-a-f; stuck-at-fault)으로 한정하며, 분할 위치를 결정하는 요소로 redundant이거나 aborted인 검출불가고장(UDF; undetectable fault)의 위치를 중심으로한 연관 신호선들의 절단에 대한 가중치를 계산하여 이 값에 따라 분할 위치를 결정하는 알고리즘을 제안한다. 분할은 먼저 대상 회로에서의 검출불가 고장을 가진 신호선을 찾고, 다음에 회로내의 관련 신호선에 대한 절단이득을 계산하여 회로 분할에 대한 위치를 결정한다. 대상회로의 검출불가 신호선을 찾는 도구는 고장검출 시뮬레이터인 ATLANTA^[8]를 이용하였으며, 본 연구에서 제시한 테스트를 고려한 2분분할 알고리즘인 BPT(bipartition for testability)로 절단이득과 고장검출 이득에 의해 결정된 노드에 대해 분할 과정을 수행한다. 또한 본 알고리즘은 UDF를 갖는 특수한 구조의 회로들과 ISCAS85 벤치마크 회로들에 대하여 실험되어지며, 그 결과 분할에 의한 고장 검출율이 향상됨을 보인다.

II. 관련연구

본장에서는 분할의 목적과 기존의 연구 내용을 소개하고 UDF를 줄이기 위한 분할의 필요성을 제시하였

다. 또한 UDF가 발생하게 되는 회로의 구조를 설명하였다.

1. 고장 검출과 분할문제

회로의 고장을 찾는 연구는 반도체의 설계에서 중요한 요소가 되었다. 일반적으로 회로에서는 시뮬백터를 입력에 인가하였을 때 출력에 나타난 결과가 정상치와 같은지 여부를 비교하여 고장을 검출한다. 고장을 검출하기 위해서는 첫째 고장이 존재하는 위치에 고장값을 만들고, 둘째 고장값을 적어도 하나의 경로를 따라 출력으로 전달하며, 셋째 고장이 전파되는 경로상에 존재하는 게이트들은 고장값이 출력에 나타날 수 있도록 모든 입력에 제어값(control value)이 설정되어야 한다. 따라서 회로가 대화화됨에 따라 고장 검출에 걸리는 시간은 기하급수적으로 증가하며 또한 입출력선수의 제약에 따라 UDF의 숫자도 증가된다.

회로의 분할기법은 큰 시스템을 나누어서 상호작용하는 작은 부분으로 나누는 것을 말한다. 시스템의 설계에 있어서 분할 기법이 필요한 이유는 다음과 같다. 첫째로 시스템의 결합구성과 시뮬레이션에 사용되고 있는 현재의 설계 도구의 기능 제한으로 인하여 복잡한 시스템 전체를 수용할 수 없는 경우이며, 둘째로 소자의 크기제한으로 인하여 하나의 소자안에 설계된 전체 시스템을 내장할 수 없는 경우, 셋째로 시스템 수행의 최적화, 제작비의 저렴화 및 설계 변화에 대처하기 위한 경우이다. 그러므로 현재의 설계여건상 분할은 반드시 필요한 설계과정의 한 요소가 된다.

분할을 위해 여러가지 방향으로 연구가 수행되어 왔다. Two-way partitioning은 대상회로를 정해진 균형 비율내에서 이분하는 방법으로 분할에의해 절단되는 에지의 수를 최소화 한다든지 다른 목적을 위해 균형화된 분할을 해나가는 방법이다^[2]. Multi-way partitioning은 둘 이상의 블럭으로 분할을 수행하는 것으로서 각 구성요소의 크기와 핀수의 최대 최소값에 대한 제한을 가하여 두고 이것을 만족시키는 범위 내에서 각 구성요소 사이의 연결을 최소화 할 수 있도록 하는 기법이다^[3,4]. Performance-driven partitioning은 구성요소간의 연결선이 수 보다 시스템의 성능을 최대화 하기 위한 방법이다. 최근에는 주로 시간을 개선하고 있지만 최소 전력 소비량도 연구의 대상이 되고 있다^[5]. Layout-driven partitioning은 소자의 배치와 전체적인 모양을 갖추도록 분할하는 방

법으로 배선의 길이를 최소화 하는 것을 목적으로 한다^[6]. Partitioning with replication은 시스템의 성능을 향상시키고 핀수와 선의 길이를 줄이기 위하여 복사기법을 병행하여 수행하는 기법이다^[7].

이러한 기존의 대부분의 회로 분할 기법들은 회로의 고장검출율과는 무관하게 회로 분할을 하며 절단 에지의 수가 줄어들어 고장검출율은 더 증가하게 된다.^[2] 회로의 테스트를 위한 분할이라는 것은 고장 검출율을 높이는 것에 중점을 둔 분할기법을 말한다. 테스트를 고려한 분할 접근법은 크게 침해적(invasive) 접근법과 비침해적(noninvasive) 접근법이 있다.^[10] 침해적 접근법은 회로간 연결선의 일부를 끊어 멀티플렉서를 삽입하고 적당한 제어신호를 이용하여 테스트 벡터를 입력하고 결과값을 출력하도록 하는 것이다. 일반적으로 많이 사용되는 방법은 시스템의 레치를 이용하여 분할을 위해 자연스러운 경계를 형성하는 것으로 시스템 레치는 BILBO(Built-in Logic Block Observer)나 LSSD(Level Sensitive Scan Design)와 같은 구조로 대체된다. 비침해적 접근법은 하드웨어의 추가없이 회로를 나누도록 하는 것으로 나누어진 부회로들의 입출력으로 직접 제어 및 관찰할 수 있도록 하는 것이다. 이러한 접근법은 부가적인 시간지연이 비교적 적으며 테스트를 위하여 추가되는 하드웨어도 없다.

고장검출율을 높이면서 회로를 분할하는 방안을 실제에 적용하기 위해서는 분할 대상 회로를 분석하여 UDF를 찾고 회로를 분할하여 검출불가 고장을 검출 가능하게 만들기 위한 최적 신호선들을 찾아 이를 절단하여 분할하는 기법이 필요하다. 본 연구에서는 비침해적 접근법인 분할방법을 택하였으며 회로에 존재하는 UDF를 없애는 방법으로 대상회로의 UDF의 위치를 중심으로한 연관 신호선들의 절단에 대한 가중치를 계산하여 이 값에 따라 분할 위치를 결정함으로써 고장 검출율을 높일 수 있는 알고리즘을 제안한다.

2. UDF를 가진 회로의 구조

디지털회로상에서 입력가능한 모든 테스트 패턴을 입력신호선에 인가하여도 검출할 수 없는 고착고장이 존재할 수 있다. 경로감지를 기본으로 하는 알고리즘을 이용하여 검출할 수 없는 고장인 UDF는, 여분 회로, 분지-재수렴(fan-out-reconvergence)회로, 그리고 구조적으로 동등한 회로에서 발생한다.^[9]

그림 1은 여분회로에서의 UDF의 예이다. 신호선 z에 대한 기능은 신호선 g에 대한 기능과 동일하므로 게이트 1과 5가 여분회로가 된다. 신호선 d상의 s-a-0(stuck-at-0) 고장이 존재한다면 이를 검출하기 위해 입력신호선 a, b에 논리 1을 가져야 한다. 또한 주출력신호선 Z로 이 고장을 전파하려면 신호선 g에 논리 0가 출력되어야 한다. 따라서 신호선 e, f는 논리 1이 되어야 한다. 그런데 a, b는 이미 1의 값을 가졌으므로 e의 값은 0이 되어 있기 때문에, 앞의 신호선 e, f가 1이 되어야하는 조건을 만족시키지 못하게 된다. 그러므로, 신호선 d상의 s-a-0 고장은 검출할 수 없다.

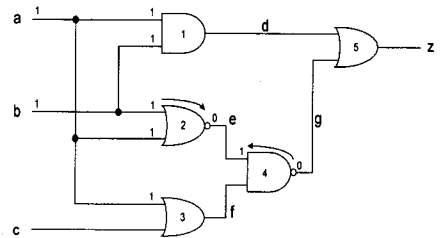


그림 1. 여분회로에서의 검출불가고장을 가진 회로 Fig. 1. Redundant circuit with UDF.

조합회로의 분지-재수렴회로상에서 분지게이트와 재수렴게이트간에 고장값이 서로 다른 경로로 전파된 후 재수렴게이트에서 만날 때, 재수렴게이트를 통해 고장 값을 전파시키지 못하는 경우에 UDF가 발생한다.

또 s-a-x 고장을 검출하기 위해 고장이 있는 선상에 /x를 인가하기 위한 정당화 과정에서의 입력값들이 고장선의 출력을 주 출력단으로 이끌어내지 못하도록 하는 경우도 있다. 또 다른 예로, 구조상으로 기능이 같은 경우 둘 이상의 고장을 검출하기 위한 테스트 벡터와 기대되는 결과값이 동일하여 어느 고장에 의하여 잘못된 출력이 있었는지를 구분할 수 없는 경우도 있다.

위에서 설명한 3가지 구조의 회로상에 존재하는 UDF를 검출하기 위한 방법은 2가지로 나눌 수 있다. 첫 번째 방법은 대상회로의 중간 게이트입력에 선을 추가하여 제어성을 높이거나 출력선을 추가하여 관측성을 높이는 방법이며, 두 번째 방법은 대상 회로의 중간 신호선들을 절단하여 분할함으로써 UDF요소를 제거하여 고장검출율을 높이는 방안이 적용될 수 있다. 본 연구는 두 번째 방법인 회로분할기법을 도입하여

UDF를 제거하는 방법을 제안한다.

III. 고장검출율을 높이는 분할 방법

본 장에서는 BPT알고리즘의 세부사항을 기술하고 UDF를 포함한 예제회로를 대상으로 BPT알고리즘을 적용하여 분할이 이루어지는 과정을 보이고 시간 복잡성을 분석한다.

1. BPT 분할 알고리즘

FM 분할^[2,13]은 한 노드를 현재의 블록으로부터 다른 블록으로 이동시킬 때, 절단집합(cutset)의 크기를 얼마나 줄일 수 있는가를 노드 이동을 위한 척도로 삼으며, 이를 절단이득(cut gain)이라 한다. 그림 2에서 각 노드를 다른 블록으로 이동할 때의 절단 이득을 나타내었다.

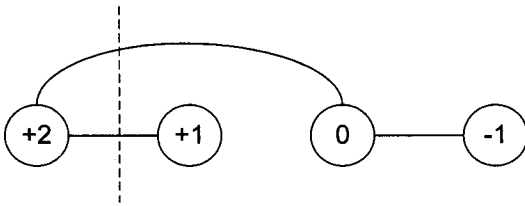


그림 2. 절단 이득
Fig. 2. Cut Gain.

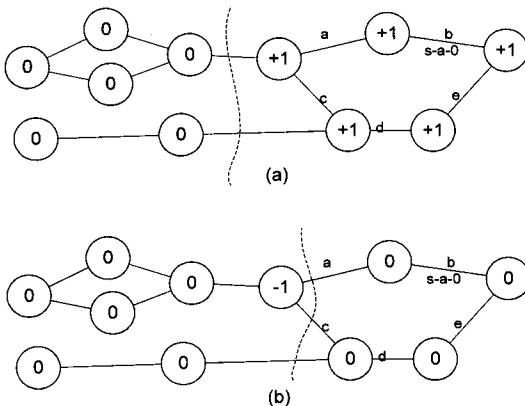


그림 3. 고장 이득
Fig. 3. Fault gains.

절단이득값이 크면 그 노드는 다른 블록으로 이동하게 될 가능성이 높다. 여기에 앞절에서 설명된 분할 값을 적용하여 이득값을 조정하면 검출 불가능한 고장을 검출할 수 있도록 해줄 수 있는 에지를 통하여 두 블록으로 분할될 확율이 높아져서 고장 검출율이 높아지

게 된다. 이때, 한 노드를 다른 블록으로 옮김으로써 검출가능해지는 UDF의 수를 고장이득(fault gain)이라고 한다. 그림 3에서 각 노드를 다른 블록으로 이동할 때의 고장 이득을 나타내었다.

에지 b에 존재하는 s-a-0가 UDF라 하고, 에지 a, b, c, d, e 중 하나가 절단되면 이 고장이 검출가능해진다고 가정하여 각 노드의 고장 이득은 나타내었다. 그림 3 (a)에서 한 노드를 이동한 후 절단 이득은 (b)와 같이 다시 계산된다.

고장 이득값을 구하는 방법은 어떤 노드에 연결된 한 에지가 절단집합에 속하여 있고 그 에지의 분할되지 않은 팬아웃 부분이 분할값을 가지거나 절단집합에 속하지 않은 다른 에지가 분할값을 가진다면 이 노드의 전체 이득을 높여주어 분할 값을 가진 에지가 절단 집합에 속할 수 있도록 한다.

분할이득과 고장이득을 함께 고려하여 회로를 이분하는 알고리즘 BPT를 그림 4에 나타내었다.

BPT(c)

begin

$r = \text{valance rate}$

$\text{CutValue}()$

randomly divide all nodes into block A,B with valance criterion

for all nodes n **do**

$n_{\text{gain}} = \text{Gain}(n)$

repeat

$\text{Pass}()$

until no nodes are moved

end

그림 4. BPT 알고리즘

Fig. 4. BPT algorithm.

먼저 균형 비율 $0 < r < 1$ 을 설정하고 각 에지에 대해 절단값을 구한다. 그 후 균형비율이 유지되는 범위 내에서 전체 회로를 임의로 두 블록 A, B로 나눈다. 이때 블록의 크기와 r과의 관계는 다음과 같이 나타낼 수 있다.

$$|A| / (|A| + |B|) \cong r$$

각 노드에 대해 초기 이득값을 구한 후 더 이상 이동할 노드가 없을 때까지 pass를 반복 수행한다.

대상회로에 존재하는 UDF는 분할에 의해 주입력 혹은 주출력단에 의해 제어 및 관찰 가능해질 필요가 있으므로 먼저 대상회로로부터 테스트 불가능한 UDF

부분을 검출해 내고 각 신호선이 분할되었을때 제거되는 UDF의 수를 나타내는 절단값을 계산하여 회로분할 위치를 결정할때 사용한다.^[11,12]

먼저 절단값을 구하는 알고리즘은 그림 5와 같다. 이 알고리즘은 모든 UDF에 대하여 정당화(justify)와 전파(propagate)의 두 과정을 거쳐 그와 관계된 입출력 선들에 대한 절단값을 계산한다. 정당화는 고장위치에 대한 고장발생값을 PI측에 대하여 정해주는 과정으로 이 과정을 통해 입력측의 관련 신호선들에 대한 절단값이 결정된다. 전파는 고장위치에서 출력측으로 고장값을 전달하는 과정으로, 이 과정을 통하여 출력측에 관련된 신호선들의 절단값이 결정된다.

각 게이트와 입출력 패드를 노드로 각 신호선을 에지로 나타낼 때, 논리회로는 노드의 집합 V와 에지의 집합 E로 나타내지는 그래프 G=(V, E)로 볼 수 있다. 이때, 에지 m에서 발생할 수 있는 UDF x를 $f_{m,x}$ 으로 나타낸다. 예를들어, 에지 3에 존재하는 UDF s-a-0(D)는 f_3 / a 로 나타낸다. 또, 에지 m에 존재하는 UDF $f_{m,x}$ 가 에지 l의 절단에 의해 제거된다면 이 UDF $f_{m,x}$ 는 집합 fs_l 의 원소가 되며, fs_l 의 원소수는 절단값 cv_l 이 된다. fs_l 와 cv_l 은 다음과 같은 식이 된다.

$$fs_l = \{f_{m,x} \mid \text{iff fault } v \text{ on edge } m \text{ is detectable by cutting edge } l\}$$

$$cv_l = |fs_l|$$

fs_l 과 cv_l 을 구하기 위하여 아래와 같은 알고리즘을 사용한다.

```

CutValue()
begin
  for every undetectable fault  $f_{m,x}$  do
    begin
      Justify(m, /x,  $f_{m,x}$ )
      Propagate(m, x,  $f_{m,x}$ )
    end
  for every edge l in G do
     $cv_l = |fs_l|$ 
  end

```

그림 5. 절단값 구하는 알고리즘
Fig. 5. Cut-value algorithm.

정당화의 과정은 임의의 에지에 특정 값을 할당하기 위한 입력측의 값들을 결정해나가는 과정으로 그림 6과 같다.

```

Justify(l, exp, f)
begin
  if l is already assigned with /exp then
    insert f into  $fs_l$ 
  set l to exp
  Implication(l,exp)
  if l is a PI then return
  c = controlling value of l
  i = inversion of l
  inval =  $exp \oplus i$ 
  if (inval = /c) then
    for every input j of l do
      Justify(j, inval, f)
  else
    begin
      select one input j of l
      Justify(j, inval, f)
    end
  end
end

```

그림 6. 정당화(justify) 알고리즘
Fig. 6. Justify algorithm.

고장 f를 정당화하기 위하여 임의의 에지 l을 기대값 exp로 할당하는 과정 중 에지 l에 이미 다른 값이 할당되어 있거나 암시되어 있는 경우 정당화를 시도하게 되면 에지 l은 기대값 exp로 설정될 수 없게 된다. 그러므로 이 에지를 절단하게 되면 외부에서 직접 원하는 값을 입력할 수 있으므로 고장집합 fs_l 에 고장 f를 추가하게 된다. 그 후 에지 l에 값 exp를 할당하고 이로인해 암시되는 이웃 에지의 값을 각각 할당한다.

고장값들을 주 출력단쪽인 PO를 향하여 감지경로를 따라 전파시키는 전파 과정은 그림 7과 같다.

```

Propagate(l, err, f)
begin
  set l to err
  insert f into  $fs_l$ 
  if l is PO then return
  k = the fanout of l
  c = controlling value of k
  i = inversion of k
  for every input j of k other than l do
    Justify(j, /c, f)
  Propagate(k, err $\oplus$ i, f)
end

```

그림 7. 전파 알고리즘
Fig. 7. Propagation algorithm.

이 과정을 통해 고장을 주 출력단쪽으로 출력할 수 있도록 감지경로내의 모든 게이트 입력단에 제어값들

을 할당한다. 이때 원하는 제어값을 할당할 수 없게 될 경우 고장은 주출력단에서 관찰할 수 없게 되므로 이 값을 관찰할 수 있도록 하기 위해서 고장이 전파 되는 감지경로중의 한 에지를 절단하든지, 아니면 고장을 전파시킬 수 없게하는 게이트의 다른 에지에 제어값을 인가할 수 있도록 그 에지나 그 에지를 제어할 수 있는 그 이전의 에지를 절단하도록 한다.

각 Pass는 그림 8과 같다.

```

Pass()
begin
  set all nodes free
  repeat
    select a max gain free node  $u$  from a side which
    satisfy valance criterion
    if selection exist then
      begin
        move  $u$  into other side of  $u_{side}$  and block it
         $u_{gain} = Gain(u)$ 
        for all neighbors  $v$  of  $u$ 
           $v_{gain} = Gain(v)$ 
        for all edges  $l$  which connected to  $u$  do
          begin
            set  $path$  to all edge  $m$  which contain
             $f_{i,x}$  in  $fs_m$ 
            for all node  $v$  connected to  $path$  do
               $v_{gain} = Gain(v)$ 
          end
        end
      until no selection exist
    end
  end
end

```

그림 8. Pass의 처리절차
Fig. 8. Pass procedure.

각 Pass는 먼저 모든 노드를 free상태로 설정하며 시작된다. 균형을 유지할 수 있는 범위 내에서 free 상태이면서 이득이 양수인 노드중에 가장 큰 이득을 갖는 노드를 선택한다. 이를 만족하는 노드가 없다면 현 Pass는 종료하게 된다. 선택된 노드는 다른 블럭으로 이동시키고 그 블럭에 머물도록 block상태로 만든다. 이 노드와 이웃노드 그리고 이 노드에 연결된 에지에 존재하는 UDF를 가지고 있는 모든 에지에 대해 다시 이득을 구한다.

앞서 설명한 바와같이 분할이득은 절단집합의 크기를 줄이는데 참고되고 고장이득은 UDF의 수를 줄이는데 참고된다. 이는 양자간 약간의 상충성을 가지므로 어느정도의 선에서 절충을 보아야 한다. 여기에서는 그림 9에서 보는 바와같이 두 이득이 같은 중요도

를 가지도록 하여 전체 이득을 결정하도록 하였다. 여러가지 응용시 요구에 따라 절단집합의 수와 UDF의 수 간에 비율을 조정하여 적용할 수 있을 것이다.

```

Gain(n)
begin
  return  $CutGain(n) + FaultGain(n)$ 
end

```

그림 9. 전체적 이득값 구하기
Fig. 9. Overall gain.

절단이득은 한 노드를 다른 블럭으로 옮겼을 때, 절단집합의 크기가 얼마나 줄어드는 가를 나타내는 값으로 아래의 알고리즘을 통해서 구할 수 있다. 노드 n 에 연결된 에지 l 이 노드 n 이 속한 블럭에 다른 노드를 가지지 않는다면 노드 n 의 이동에 의해 절단집합의 크기는 1 감소하게 된다. 반면, 노드 n 에 연결된 에지 l 이 절단집합에 속하지 않았다면 노드 n 의 이동에 의해 절단집합의 크기는 1증가하게 된다. 이와같은 규칙에 따라 노드 n 에 연결된 모든 l 에 대해 이득을 합하여 절단이득값을 구한다.

```

CutGain(n)
begin
  gain = 0
  for all edges  $l$  which connected to  $n$  do
    begin
      if  $l$  has no nodes in  $n_{side}$  except  $n$  then
        increment gain
      if  $l$  is included in  $n_{side}$  completely then
        decrement gain
    end
  return gain
end

```

그림 10. 절단이득
Fig. 10. Cut gain.

고장이득은 한 노드를 다른 블럭으로 옮겼을 때, UDF가 얼마나 감소하는가를 나타내는 값으로 그림 11의 알고리즘을 통해 구할 수 있다. 노드 n 에 연결된 모든 에지 l 에 대해, 고장집합 fs_l 에 속한 각 고장 $f_{i,x}$ 를 자신의 fs 에 포함하는 에지로 구성된 에지집합을 $path$ 라 할때, $path$ 내의 모든 에지가 노드 n 이 속한 블럭에만 존재한다면 이 노드의 이동으로 검출될 수 없었던 고장 $f_{i,x}$ 이 검출가능하게 되므로 고장이득이 증가하게 된다. 반면, l 을 제외한 $path$ 가 노드 n 이 속하지 않은 블럭에 완전히 속한다면 노드 n 의 이동으로

검출 가능하던 고장 $f_{i,v}$ 가 검출 불가능하게 되므로 고장이득은 감소하게 된다. 노드 n 에 연결된 모든 에지 l 에 대해, 이러한 이득을 합하여 고장이득값을 구한다.

```

FaultGain(n)
begin
    gain = 0
    for all l which connected to n do
        for all  $f_{i,v}$  which included in  $fs_l$  do
            begin
                set path to all edge  $m$  which contain  $f_{i,x}$ 
                in  $fs_m$ 
                if path only exist in  $n_{side}$  then
                    increment gain
                if path except l is only exist in other
                side of  $n_{side}$  then
                    decrement gain
            end
        end
    return gain
end
    
```

그림 11. 고장이득
Fig. 11. Fault gain.

2. 적용예제

본 절에서는 임의의 회로에 대하여 48% 균형비를 가지는 BPT알고리즘을 적용한 예를 보인다. 그림 12는 내부에 4개의 UDF를 포함하고 있는 회로이다. 16개의 게이트와 6개의 입력 포트, 3개의 출력 포트를 가지고 있어 총 25개의 노드로 구성된다. 48%의 균형비를 만족하기 위하여 블록내에 최소한 12개의 노드를 포함해야 한다.

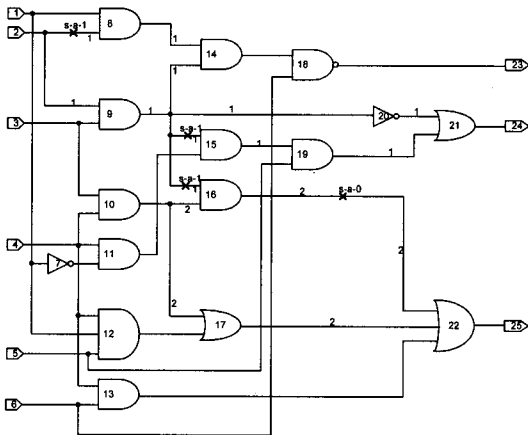


그림 12. 예제회로
Fig. 12. Example.

이 회로를 그래프로 나타내고 BPT알고리즘을 적용한 결과 수 번의 pass를 거쳐 그림 13과 같은 분할을

얻었다. 이때의 절단집합의 크기는 6이고 4개의 UDF 중 하나만이 검출 가능한 상태이다. 에지 위의 숫자는 각 상태에서 그 에지의 절단값이다. 이 상태에서 절단이득과 고장이득을 각 게이트의 좌측 상단에 나타내었다(절단이득/고장이득). 이때 전체 이득은 절단이득과 고장이득의 합이다. 블록A에는 13개의 노드가 있고 블록B에는 12개의 노드가 있으므로 블록A로부터 가장 큰 이득을 가지는 노드를 택하여 블록B로 이동시킨다. 이때, 노드 8, 14, 18만이 이득값 0을 가지고 있고 나머지는 모두 음의 이득값을 가진다. 세계의 노드 중 14번 노드가 임의로 선택되어 블록B로 이동되면 절단선 ②를 통하여 회로가 분할되게 된다. 이에따라 새로이 이득이 계산되어지고 이를 각 게이트의 왼쪽아래에 나타내었다. 블록B가 13개의 노드를 가지므로 여기에서 이동될 노드가 선택되어야 하고 15번 노드만이 이득값 0을 가지고 나머지는 음의 이득을 가지므로 이 노드가 선택되어 블록A로 옮겨진다.

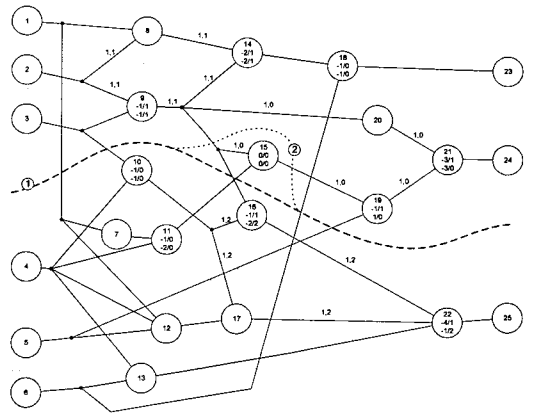


그림 13. 예제회로의 절단 1,2
Fig. 13. Example circuit partitioning 1,2.

15번 노드가 옮겨진 이후의 회로를 그림 14에 나타내었다. 절단선 ③을 통해 절단된 이 상태에서는 블록 A의 19번 노드가 이득값 1을 가지므로 선택되어 블록 B로 옮겨진다. 절단선은 ④로 조정되고 이어 블록 B에서 이득값 0을 갖는 10번 노드가 블록 A로 옮겨진다. 절단선이 ⑤로 조정된 후에는 균형비를 유지하면서 양의 이득을 갖는 노드가 없으므로 BPT알고리즘은 종료하게 된다.

BPT알고리즘이 종료한 후 절단집합의 크기는 7이 되었고 4개의 UDF중 3개가 검출가능한 상태가 되었다. 결과를 분석해 볼 때, 임의로 분할했을 때에 비해

절단집합의 크기는 다소 증가할 수 있으나 UDF의 수는 감소하였음을 알 수 있다.

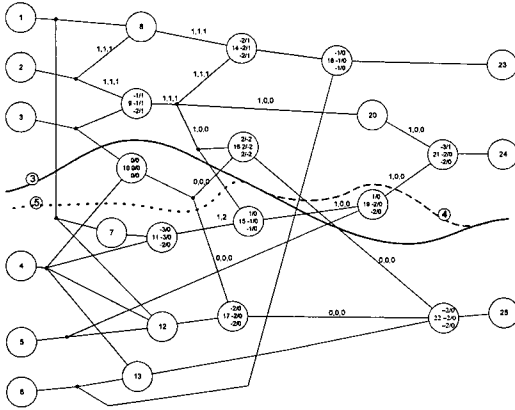


그림 14. 예제회로의 절단 3,4,5
Fig. 14. Example circuit partitioning 3,4,5

3. 복잡성 분석

FM 알고리즘은 linear time 알고리즘으로 회로내의 핀수의 총합을 p 라 할 때 각 Pass의 시간 복잡성은 $O(p)$ 로 알려져 있고 각 노드에 핀수의 제한이 있을 때 일반적으로 노드수에 비례한다. BPT알고리즘은 FM알고리즘을 기본으로 하고 있으며 최초 절단값을 구하기 위하여 각 UDF에 대하여 propagate와 justify를 수행하는데 필요한 시간이다. propagate와 justify를 수행하기 위하여 최악의 경우 모든 노드를 모두 방문할 수 있으므로 UDF수를 u , 노드의 수를 n 이라 할때, $O(un)$ 이다. 그리고 각 pass에서 노드를 이동할 때마다 집합 fs 를 참조하여 path를 설정하고 path를 형성하는 각 노드의 neighbor의 gain을 수정해 주어야 한다. 그러므로 path의 길이는 최악의 경우 n 이 될 수 있으나 관찰에 의하면 그래프에서의 한 cycle크기 정도에 해당된다. 한 path의 평균 크기를 c 라하고 한 노드의 평균 핀수를 d 라 할때 한 pass의 수행 시간은 $O(upcd)$ 이므로 전체의 수행시간은 $O(u(pcd+n))$ 로 볼 수 있다.

IV. 실험 및 고찰

본 장에서는 몇 가지 회로를 사용하여 BPT 알고리즘을 테스트하였다. UDF를 찾아내어 고장 검출율을 구하기 위하여 ATPG 프로그램인 ATALANTA ver 1.1이 사용되었다. BPT 알고리즘은 C++로 구현되었

으며 PENTIUM PC상에서 수행하였다. 본 실험에서는 3장에서 든 예제회로와 ISCAS85 회로 중 UDF를 포함하는 9개의 회로에 대해서 균형 비율을 45%로 설정하여, 대표적인 2분분할알고리즘인 FM알고리즘과 본 연구에 의한 BPT알고리즘을 수행시키었으며 그 결과는 표 1과 같다.

표 1. BPT 알고리즘을 대상회로에 실험한 결과

Table 1. Testing result for BPT algorithm.

Source circuit				cut-set		time (sec)		Block A		Block B		BPT-UDF / FM-UDF
Circuit	Nodes	UDF	Fault coverage	FM BPT		nodes	UDF	nodes	UDF			
example	25	4	92.308	5	0.18	15	1	18	2		0.33	
				7	0.26	15	0	19	1			
c432	203	4	99.237	22	0.21	132	1	131	0		0	
				24	1.15	130	0	132	0			
c499	275	8	98.945	18	0.54	160	0	161	4		0.25	
				21	1.95	162	1	163	0			
c1355	619	8	99.492	47	1.37	336	0	336	0			
				48	3.33	345	0	389	0			
c1908	938	9	99.521	60	2.29	510	0	509	7		0.57	
				64	16.98	521	1	592	3			
c2670	1566	117	95.741	84	4.06	818	29	980	46		0.61	
				94	63.46	831	15	972	31			
c3540	1741	137	96.004	105	4.15	1089	66	923	43		0.51	
				116	76.14	948	25	1073	31			
c5315	2608	59	98.897	43	6.52	1240	36	1490	23		0.52	
				45	66.34	1243	20	1494	11			
c6288	2480	34	99.561	45	5.92	1146	2	1411	32		0.74	
				51	59.18	1322	18	1240	6			
c7552	3827	131	98.265	31	7.48	1983	8	1931	88		0.62	
				36	98.43	1953	22	1954	38			

검출 불가능한 고장을 가진 회로에 대한 시험에서 FM 방법과 BPT에 의한 결과를 비교해 볼 때, UDF 수는 FM 방법인 경우 평균 57%로 감소했으며, BPT는 32%로 감소했다. BPT 알고리즘은 FM에 의한 분할에 비해 UDF의 수에 있어서 평균적으로 46% 감소되었으므로 높은 고장 검출율을 보이고 있다. 그러나 고장 검출을 위해 임의의 신호선이 절단되도록 이득값을 수정하기 때문에 절단집합의 크기는 좀 더 커지게 된다. 10개의 회로에 의한 실험에서 FM에 비해 BPT의 절단집합의 크기는 110%로 증가되었다. 또한 분할에 의하여 절단집합 만큼의 추가 핀이 더 생기게 되어 overhead를 가지게 된다. 원시회로의 노드 수에 비해

나누어진 후의 노드수의 합이 더 큰 것은 절단집합의 크기가 커짐에 따라 입출력 단자가 많아진 것과 ATALANTA 1.1에서의 테스트를 위하여 임시 버퍼를 추가한 출력단이 있기 때문이다.

V. 결 론

본 논문에서는 조합 논리회로의 단일 stuck-at-fault에 대한 고장검출율을 높이기 위해 절단에 따른 cut의 수가 적게 증가되고 검출불가고장인 UDF가 최대한으로 감소하는 노드를 찾아 옮김으로써 고장검출율이 향상될 수 있는 분할 기법을 제안하였다. 이를 위해 절단값, 절단이득(cut gain), 고장이득(fault gain)을 산출하고 두 이득의 합이 가장 큰 노드를 이동하여 분할하므로 절단수의 증가를 줄이고 고장검출율을 높일 수 있게 하였다. 분할을 위한 노드의 이동은 이득값을 이용하기에 적합한 FM분할 방법을 기본으로 하였으며, 실험결과 분할 후 검출율이 향상되었음을 확인하였다. 이는 분할에 의해 회로내부에 감추어진 신호선들이 주입력단이나 주 출력단으로 바뀌면서 가시성이 높아짐에 따라 고장 검출율이 높아졌기 때문이다.

참 고 문 헌

[1] F. M. Johnnes, "Partitioning of VLSI Circuits and Systems," 33rd Design Automation Conf., pp. - , Las Vegas, USA, June 1996.

[2] S. Dutt, W. Deng, "A Probability-Based Approach to VLSI Circuit Partitioning," 33rd Design Automation Conf., pp. - , Las Vegas, USA, June 1996.

[3] N. C. Chou, L. T. Liu, C. K. Cheng, W. J. Dai, R. Lindelof, "Local ratio cut and set covering partitioning for huge logic emulation systems," Trans. on Computer-Aided Design of Integrated Circuits and Systems, no. 14, pp. 1085-1092, 1995.

[4] D. J. Huang, A. B. Kahng, "Multi-way system partitioning into single and multiple type FPGAs," Int'l Symp. on Field-Programmable Gate Arrays, pp.

140-145. ACM, 1995.

[5] Y. Katsura, T. Koide, S. Wakabayashi, N. Yoshida, "A new system partitioning method under performance and physical constraints for multi-chip modules," Asia and South pacific Design Automation Conf., pp 113-118, 1995.

[6] B. M. Riess, K. Doll, F. M. Johannes, "Partitioning very large circuits using analytical placement techniques," Design Automation Conf., pp. 646-651, 1994.

[7] L. J. Hwang, A. E. Gamal, "Min-cut replication in partitioned networks," Trans. on Computer-Aided Design of Integrated Circuits and Systems, no. 14, pp. 96-106, 1995.

[8] H. K. Lee and D. S. Ha, "On the Generation of Test Patterns for Combinational Circuits," Technical Report No. 12_93, Dep't of Electrical Eng., Virginia Polytechnic Institute and State University.

[9] T. W. Williams, Advances in CAD for VLSI Volume5 VLSI TESTING, North-Holland, 1985.

[10] B. R. Shaer, "Circuit partitioning to enhance VLSI testability," Doctoral Dissertation, Dep't of Electrical Eng. Univ. of South Florida, December 1995.

[11] M. Abramovici, M. A. Breuer, A. D. Friedman, Digital Systems Testing and Testable Design, Computer Science Press, 1990.

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

[13] C. M. Fiduccia, R. M. Mattheyses, "A Linear-Time Heuristic for Improving Network Partitions," 19th Design Automation Conf., pp. 175-181, 1982.

저 자 소 개



盧貞鎬(正會員)

1948년 1월 15일생. 1972년 2월 한국항공대학 전자공학과 졸업. 1987년 8월 경북대학교 대학원 전자공학과 석사학위 취득. 1976년부터 현재 국방과학연구소(현 책임연구원). 주관

fault 시뮬레이션임



金相鎭(正會員)

1994년 계명대학교 전자계산학과 졸업(공학사). 1996년 경북대학교 대학원 컴퓨터공학과 졸업(공학석사). 1998년 경북대학교 대학원 컴퓨터공학과 박사과정 수료. 관심분야는 VLSI 테스트, 회로분할

李昌熙(正會員) 第 35卷 C編 第 2號 參照

尹泰鎭(正會員) 第 35卷 C編 第 2號 參照

安光善(正會員) 第 35卷 C編 第 2號 參照