

論文2001-38SD-2-7

Markov 확률모델을 이용한 저전력 상태할당 알고리즘 (FSM State Assignment for Low Power Dissipation Based on Markov Chain Model)

金 鍾 洙*
(Jongsoo Kim)

요 약

본 논문은 디지털 순서회로 설계시 상태할당 알고리즘 개발에 관한 연구로, 동적 소비전력을 감소시키기 위하여 상태변수의 변화를 최소화 하는 코드를 할당하여 상태코드가 변화하는 스위칭횟수를 줄이도록 하였다. 상태를 할당하는 데는 Markov의 확률함수를 이용하여 hamming거리가 최소가 되도록 상태 천이도에서 각 상태를 연결하는 edge에 weight를 정의한 다음, 가중치를 이용하여 각 상태들간의 연결성을 고려하여 인접한 상태들간에는 가능한 적은 비트 천이를 가지도록 모든 상태를 반복적으로 찾아 계산하였다. 비트 천이의 정도를 나타내기 위하여 cost 함수로 계산한 결과 순서회로의 종류에 따라 Lakshmikanth의 알고리즘보다 최고 57.42%를 감소시킬 수 있었다.

Abstract

In this paper, a state assignment algorithm was proposed to reduce power consumption in control-flow oriented finite state machines. The Markov chain model is used to reduce the switching activities, which closely relate with dynamic power dissipation in VLSI circuits. Based on the Markov probabilistic description model of finite state machines, the hamming distance between the codes of neighbor states was minimized. To express the switching activities, the cost function, which also accounts for the structure of a machine, is used. The proposed state assignment algorithm is tested with Logic Synthesis Benchmarks, and reduced the cost up to 57.42% compared to the Lakshmikanth's algorithm.

I. 서 론

기존의 Computer Aided Design(CAD)은 최적화 과정에서 주로 면적, 속도 및 Testability에 관하여 연구하였으나, 이동 통신 장비수요가 증가함에 따라서 전력 소비량이 문제가 대두되었다. 이러한 통신장비는 비메모리분야의 개발을 주도하게되었고, 반도체의 전력소비는 설계수준의 척도를 나타낼 정도로 중요한 성능측정 방법의 한 요인이 되었다. 종래에 수행하던 저전력 연구들은 하위수준에서 회로의 지연시간을 작게 유지하

면서 회로면적을 줄이는 방법과^[1~4], 상위수준에서 스위칭 변화를 감소시키려는 연구들이 제안되었다^[2,5,6~8]. 그러나, 아직 상위수준에서의 회로합성결과 자체도 만족할 만한 수준이 되지 못하지만 여러 가지 장점으로 인하여 저전력을 구현하려는 방법에 대한 연구가 필요하게 되었다.

일반적인 디지털 순서회로에서 상태수가 n 개일 때 코드의 길이는 $\log_2 n$ 에서 n 이 된다. 참고문헌 [8]에서 이용한 방법은 One Hot Code를 사용하여 상태할당을 하고 다음에 상태 천이확률을 근거로 상태 비트 동일화 및 비트 수정 단계를 거쳐 최종 상태코드를 할당하는 방법으로써 최대 길이에서 최소길이의 접근법이다. 이외에 상태를 할당하는 알고리즘에는 모든 변수를 포함하는 방정식을 계산하는 Integer Linear

* 正會員, 蔚山大學校 電氣電子情報시스템工學部
(Dept. of Elec. Eng., Univ. of Ulsan)

接受日字: 2000年5月8日, 수정완료일: 2001年1月16日

Programming(ILP)에서부터 인공지능 방법인 유전자 알고리즘을 사용하여 상태할당을 하는 방법 등이 제안되었다^[10]. 또한 입력의 형태가 정해지는 신호처리와 같은 특정 적용 분야에서는 그 입력을 분석하여 일정한 기준을 만들어 내어 Simulated Annealing 방식으로 코드를 변환시키는 방법도 있다^[5]. 비동기회로에서는 Racing으로 인하여 여분의 상태할당을 이용하는 알고리즘에 관한 연구도 발표되었다^[7].

본 논문은 순서회로 상태 천이도를 Markov Model을 이용하여 계산한 후 상태가 천이할 때 이들의 코드의 변화를 최소화함으로써 저전력을 구현하기 위한 상태할당 알고리즘에 관한 연구로, 상태 값이나 상태를 연결하는 에지들의 값들을 이용하여 상태 코드를 할당하는 3가지 방법을 제안하였으며 본 논문의 구성은 다음과 같다. 2장에서는 전력소비모델을 이용하여 상위수준에서 저전력을 구현하기 위한 요소를 살펴보고, 3장에서는 스위칭 정도를 표현하는 확률 모델에 대하여 언급하였다. 4장에서는 확률모델을 바탕으로 가중치(weight)를 계산하기 위한 과정을, 5장에서는 스위칭의 변화를 나타내기 위한 Cost 함수의 정의를, 6장에서는 3 가지 상태 할당 알고리즘을 제안하였다. 7장에서는 실험결과를 벤치마크회로로 실험한 결과를 기존의 알고리즘과 비교 분석하였으며, 끝으로 8장에서는 결론과 앞으로의 연구방향에 대하여 기술하였다.

II. 전력소비모델

다음 식 (1)은 CMOS회로에서의 전력소비 모델이다^{[6][11]}.

$$P_{total} = \frac{1}{2} V_{dd}^2 \cdot f \sum_{nodes} C_i \cdot P_i \quad (1)$$

P_{total} 은 소모되는 전체 전력을 나타내며, f 는 주파수, V_{dd} 는 공급전압, C_i 는 i 번째 노드에서의 부하용량, 그리고 P_i 는 CMOS회로에서 입력의 스위칭에 따라 공급전압으로부터 충전되거나 접지로 방전될 때 발생하는 스위칭 전력소모를 나타낸다. 여기에서 P_i 를 스위칭 확률이라 부른다.

일반적으로 CMOS 회로에서 소모되는 전력은 스위칭 전력, 단락회로 전력, 그리고 누설전력이다. 이 중에서도 전력소모의 대부분을 차지하는 것이 스위칭 전력이다.

상위 수준에서는 자원공유를 최소화하여 데이터 상관관계를 보존하며 스위칭 변화를 감소시키는 방법으로 저전력을 구현하거나, 파이프라인 등을 사용하여 회로의 속도나 시간당 처리량을 높인 다음 공급 전압을 낮추어 처리량을 유지하는 방법도 있다^[9]. 잡음에 의한 오동작으로 전압이 고정된 경우 소비전력이 주파수와 직접 비례하기 때문에 주파수를 감소시키는 방법도 있으나 고정된 타이밍 제한조건을 가지는 실시간 응용 분야에서는 주파수를 감소시킬 수가 없는 경우 스위칭 변화율을 감소시켜 저전력을 구현할 수 있다.

III. 확률모델

CMOS 회로에서 전력소비의 주된 요인은 앞에서 지적한 바와 같이 스위칭 변화율이다. 평균 스위칭 변화는 스위칭 확률이나 스위칭 천이 확률로 나타낸다^[2]. 현재 대부분의 전력 측정방법은 패턴 의존성(pattern-dependence) 문제를 해결하기 위하여 확률개념을 사용하는데 보통 Zero Delay 모델이 사용되고 공간적, 시간적으로 서로 독립적이라고 가정한다^[11]. 따라서, 본 논문에서는 확률모델로서 Markov Model을 사용하였다.

시스템에 관한 정보와 입력에 대한 확률이 주어지면 상태 천이도에 대한 천이확률을 계산할 수 있는데, 입력에 대한 확률은 시스템에 대한 상위수준의 기술에서 얻을 수도 있고, 설계자로부터 직접 정보를 얻을 수도 있다. 입력에 대한 확률을 얻은 다음 이 확률을 Markov Chain의 형태로 변환하는데 상태 천이도가 사용된다^{[2][12]}. 이들은 다음과 같은 식으로 표현된다.

상태 S_i 에서 상태 S_j 로 천이할 조건부 확률은 식 (2)와 같이 표현할 수 있다.

$$p_{i,j} = Prob \left(\begin{array}{l} \text{다음상태} = S_j \\ \text{현재상태} = S_i \end{array} \right) \quad (2)$$

상태의 천이는 입력이 변화할 때 발생하므로 천이조건 확률은 입력신호의 변화를 이용하여 계산한다. 상태 천이조건 확률은 입력 값의 변화만을 고려하여 계산한 값이기 때문에 시스템의 스위칭 정도를 정확하게 나타낼 수 없다. 따라서 입력 변화에 의한 상태 천이뿐만 아니라 내부의 천이를 동시에 고려한 총 상태 천이확률을 다음과 같이 정의할 수 있다.

$$P_{i,j} = p_{i,j} \cdot P_i \quad (3)$$

여기서, P_i 는 상태 i 의 확률이다. 식 (3)에서 높은 값의 총 상태 천이확률을 얻기 위해서는 조건부 상태 천이확률 $p_{i,j}$ 와 상태 i 에 머무는 상태 확률 P_i 가 모두 높아야 함을 알 수 있다. 그리고 입력변화에 의한 상태의 천이만을 고려하는 조건부 상태 천이확률만 가지고는 잘못된 확률 값을 갖고 계산할 수도 있음이 입증되었다^[2].

다음은 Homogeneous Markov Chain에 대하여 살펴본다.

$$B = \begin{bmatrix} p_{0,0} & p_{0,1} & p_{0,2} & \dots \\ p_{1,0} & p_{1,1} & p_{1,2} & \dots \\ \dots & \dots & \dots & \dots \\ p_{i,0} & p_{i,1} & p_{i,2} & \dots \\ \dots & \dots & \dots & \dots \end{bmatrix} \quad (4)$$

B 는 조건부 상태 천이확률 행렬 $p_{i,j}$ 를 원소로 갖는 행렬이며, 확률의 공리로부터 다음의 관계를 얻을 수 있다.

$$\sum_j p_{ij} = 1 \quad (5)$$

A 를 상태확률 P_i 를 원소로 갖는 정상 상태 확률 벡터라 하면 다음과 같은 관계를 얻을 수 있다^[2].

$$A^T \cdot B = A^T \quad (6)$$

$$\sum_{i=0}^{\text{상태수}} P_i = 1 \quad (7)$$

위의 식 (6)과 식 (7)을 연립으로 풀어서 정상 상태 확률을 구한 후, 식 (3)에 대입하면 필요로 하는 총 상태 천이확률을 구할 수 있다.

IV. 상태천이도(State Transition Diagram)

VHDL과 같은 상위수준으로 기술된 입력을 분석하여 표 1과 같은 상태천이표를 얻었다고 가정하면 상태 천이도는 그림 1과 같다.

그림 1에서 현재 상태에서 다음 상태로 천이하는 경우의 수는 입력 비트의 수가 2비트이므로 총 4가지이다. 그 중에서 State 1에서 State 2로의 천이는 입력이 총 4가지 중에서 01/10과 같은 2 가지 경우에 발생하므로 조건부 상태 천이 확률은 2/4가 되고 State 2에서 State 1로의 천이는 입력이 11인 경우 발생하므로 그림 2와 같이 1/4가 된다.

표 1. 상태 천이표
Table 1. State Transition Table

입력	현재상태	다음상태
11	S1	S1
01/10	S1	S2
00	S1	S3
11	S2	S1
01/10/00	S2	S3
00	S3	S3
10	S3	S1
01/11	S3	S4
10/11	S4	S2
01/00	S4	S3

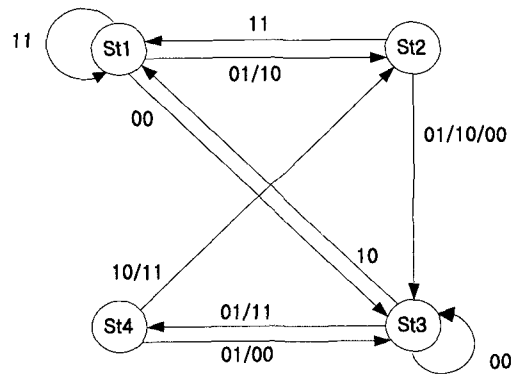


그림 1. 상태천이도
Fig. 1. State Transition Graph.

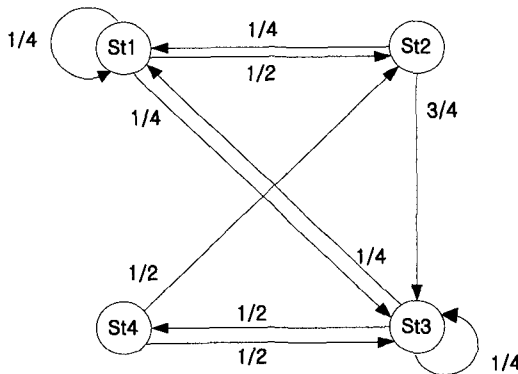


그림 2. 조건부 상태천이확률
Fig. 2. Conditional State Transition Probabilities.

식 (8)은 그림 2의 조건부 상태 천이확률 원소를 행렬 B 로 표기한 것이다.

$$B = \begin{bmatrix} \frac{1}{4} & \frac{2}{4} & \frac{1}{4} & 0 \\ \frac{1}{4} & 0 & \frac{3}{4} & 0 \\ \frac{1}{4} & 0 & \frac{1}{4} & \frac{2}{4} \\ 0 & \frac{2}{4} & \frac{2}{4} & 0 \end{bmatrix} \quad (8)$$

식 (5)(6)(7)에 행렬 B 를 대입하면 다음의 방정식을 얻을 수 있다.

$$\begin{aligned} P_1 &= \frac{1}{4} P_1 + \frac{1}{4} P_2 + \frac{1}{4} P_3 \\ P_2 &= \frac{2}{4} P_1 + \frac{2}{4} P_4 \\ P_3 &= \frac{1}{4} P_1 + \frac{3}{4} P_2 + \frac{1}{4} P_3 + \frac{2}{4} P_4 \\ P_4 &= \frac{2}{4} P_3 \\ P_1 + P_2 + P_3 + P_4 &= 1 \end{aligned} \quad (9)$$

결국, 식 (9)의 방정식을 풀면, 상태확률 P_i 를 얻을 수 있는데, $P_1 = \frac{1}{5}$, $P_2 = \frac{1}{5}$, $P_3 = \frac{2}{5}$, $P_4 = \frac{1}{5}$ 으로 이 값들을 식 (3)에 대입하면 그림 3과 같이 총 상태 천이확률을 얻는다.

각 상태들 간의 천이 정도만을 고려하기 때문에 그림 3에서 상태 1에서 상태 2로 천이와 상태 2에서 상태 1로 천이를 동일한 것으로 간주한다. 따라서 각 상태들간의 천이 정도는 방향성 상태 천이확률의 합을 정수화 하면 상태 1과 상태 2사이의 가중치(weight)는 상태 1에서 상태 2로의 천이확률 $1/20$ 과 상태 2에서 상태 1로의 천이확률 $1/10$ 을 합하여 그림 4와 같이 정수화된 3으로 방향성이 없게된다.

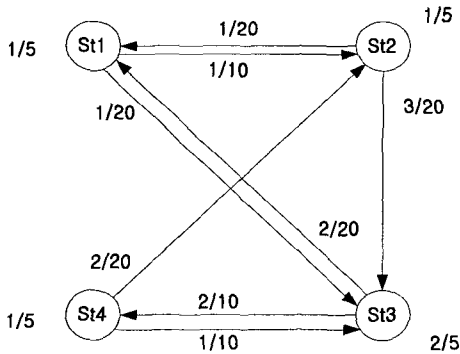


그림 3. 상태확률과 총 상태천이확률
Fig. 3. State Probabilities and Total State Transition Probabilities.

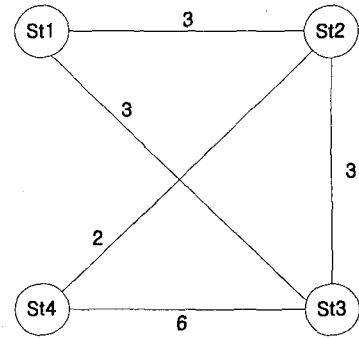


그림 4. 가중치로 표시된 상태 천이도
Fig. 4. Weighted State Transition Graph.

V. Cost 함수

이 절에서는 가중치가 표시된 상태 천이도를 이용하여 각각의 상태에 상태 코드를 할당한 후 이들의 성능을 계산하는 측정함수에 대하여 설명한다. 상태 천이시 할당된 상태코드의 변화는 동적 전력소모의 주된 원인이 되기 때문에 가능하면 각각의 상태간의 Hamming Distance가 2이상 되지 않게 처리하여야한다. 가중치를 고려하지 않고 단순히 코드의 변화만을 이용하면 의미가 없으므로 다음과 같은 Cost 함수를 정의한다.

Cost 함수는 식 (10)과 같이 상태들 사이의 스위칭 정도를 나타내는 가중치와 상태 i 와 상태 j 간의 코드 비트 변화의 곱으로 표시한다.

$$cost(i, j) = weight \times \sum_{k=1}^{state\ code\ bit} \text{mod}(C_i(k), C_j(k)) \quad (10)$$

전체 상태 천이도의 Cost는 각 상태의 연결성을 고려하여 부분 Cost들로부터 얻을 수 있다.

$$cost = \sum_{edge\ count} cost(i, j) \quad (11)$$

식 (11)을 이용하여 얻은 Cost는 공급전압이나 주파수와 부하용량 등과 같이 하위수준에서 결정되는 요인들이 고정된 경우 전력소비량을 나타내므로 상태 천이시 코드의 변화를 감소시키면 저전력을 구현할 수 있게된다.

VI. 알고리즘

본 논문에서는 시작 상태의 선택은 최종 Cost에 큰 영향을 미치지 않기 때문에 초기 상태를 우선 할당하는 기

존의 알고리즘들과는 다르게 여러 상태들 중에서 가장 천이확률이 높은 상태를 시작 상태로 선택한다. 시작상태를 결정하는데는 두 가지 방법이 있는데 하나는 어떤 상태로 향하는 에지들에 부여된 상태간 천이확률의 합이 가장 큰 상태를 선택하는 방법이고 다른 하나는 단순히 상태간의 천이확률이 가장 높은 에지를 선택하는 방법이다.

본 논문에서는 단지 가중치만을 고려해서 높은 가중치를 가지는 에지들 순서로 상태할당을 하는 Lakshmikanth^[13]의 알고리즘과는 달리 가중치와 연결성을 동시에 고려하여 상태할당을 하였다.

그림 5는 제안된 알고리즘을 간단히 표현한 Flow Chart다. 입력 변화에 대한 상태 천이정보를 입력받아서 상태 천이확률을 계산하고 방향성 상태 천이도를 구한다. 방향성 상태 천이도에서 방향성을 제거하면서 얻은 가중치를 각각의 상태에 대한 기본적인 정보로 이용한다. 상태 코드의 비트 범위는 앞에서 언급하였듯이 $\log_2 n$ 에서 상태수 n 으로 코드 비트를 변화시키면서 각 상태 비트 수에서의 할당을 수행하여 Cost가 가장

작은 비트 수의 상태할당을 선택한다. 연결된 노드를 찾아 처리하는데는 다음과 같은 3가지 방법이 있다.

1. Maximum Value Edge Searching(MVES)

MVES 알고리즘은 상태도를 읽어 들인 후, 상태 천이확률이 가장 높은 에지를 찾는다. 천이확률이 가장 높은 에지를 선택하여 그 에지에 연결된 상태 2개를 할당한다. 이 MVES 알고리즘은 이미 할당된 에지와 연결된 에지들을 고려 대상으로 하여 그 중에서 가장 천이확률이 높은 에지를 먼저 할당하는 방식이다. MVES의 처리과정을 그림 4의 예를 들어 설명한다.

그림 4에서는 State 3과 State 4사이의 에지가 6으로 가장 높은 가중치를 가진다. State 3과 State 4에 상태코드 00과 01을 할당하면, 에지 $e_{3,4}$ 는 소거되고 에지 $e_{2,3}$, $e_{1,3}$, $e_{2,4}$ 가 남는다. 가장 높은 가중치를 가지는 에지를 다음 할당할 에지로 선택해야 하므로 에지 $e_{1,3}$ 과 $e_{2,3}$ 이 3으로 동등한 가중치를 갖는다. 이렇게 동등한 에지값이 2개 이상일 경우 어느 것을 선택 하여도 상관없다. 할당할 상태가 정하여지면 할당할 상

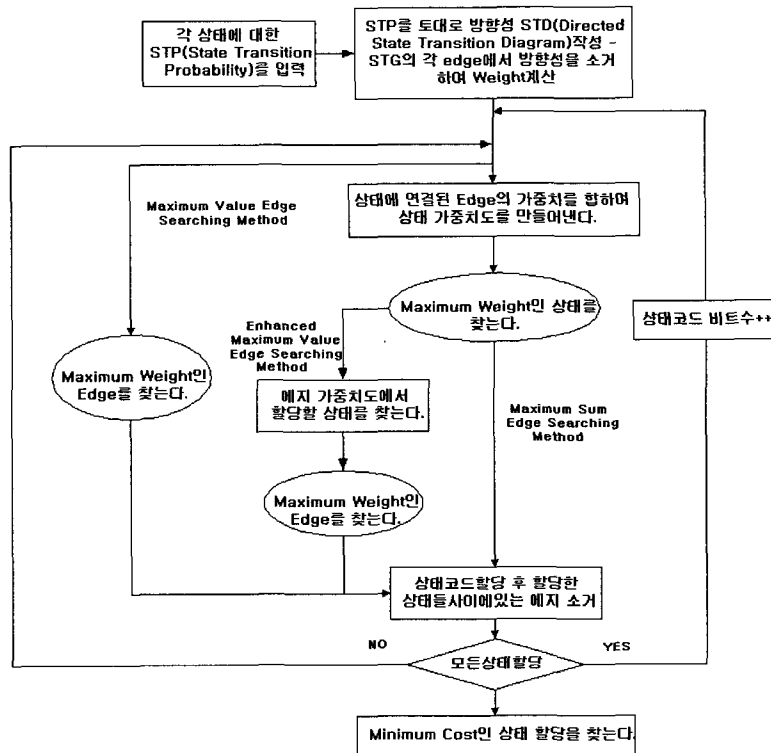


그림 5. 알고리즘 flow chart
Fig. 5. Proposed Algorithm.

태와 연결된 상태 중에 이미 할당한 상태의 코드들을 가지고 할당할 수 있는 코드의 집합에서 Cost가 가장 작은 코드를 찾는다. 여기서 가장 작은 비트 변화를 갖는 코드가 10이므로 State 1의 상태 코드로 10을 할당한다. State 1,3,4는 할당되었으므로 이들 사이에 있는 에지 $e_{1,3}$ 을 소거한다. 마지막으로 $e_{2,4}$ 와 $e_{2,3}$ 이 있는데 이는 State 2를 가리키므로 인접한 상태 코드와 가장 적은 Cost를 가지는 코드인 11을 할당한다. 즉 할당순서는 State 3, 4, 1, 2가 된다. 최종 Cost는 식 (10)과 식 (11)을 이용하면 다음과 같다.

$$\begin{aligned} \text{cost} &= 3 \times \text{mod}(11,10) + 2 \times \text{mod}(11,01) + \\ & 2 \times \text{mod}(11,00) + 6 \times \text{mod}(00,01) + \\ & 3 \times \text{mod}(10,00) = 20 \end{aligned}$$

2. Enhanced Maximum Value Edge Searching (EMVES)

EMVES 방식은 각각의 상태에 연결된 에지들의 가중치를 더하여 상태에 가중치를 부여한다. 이렇게 부여된 수치들을 상태 가중치라 하고, 가장 큰 상태 가중치를 가지는 상태를 초기 할당할 상태로 선택한다.

그림 4에서 State 1=6, State 2=8, State 3=12 State 4=8로 State 3이 12로 상태 가중치가 가장 높으므로 00이 할당되는데 동일한 값이 2개 이상일 경우 연결된 에지들의 수가 많은 상태로 선택한다. 가능한 에지 $e_{2,3}$, $e_{3,4}$, $e_{1,3}$ 가운데서 $e_{3,4}$ 가 6으로 높으므로 State 4를 01로 할당한다. 다음 이에 연결된 State 2를 할당하면 코드가 11이 되고 끝으로 State 1을 처리하면 MVES 알고리즘과 동일한 결과를 얻게되나 처리순서는 에지가 순차적으로 배열이 되어있다고 가정하면 State 3, 4, 2, 1로 된다.

3. Maximum Sum Edge Searching(MSES)

MSES 방법은 그림 4와 같이 에지의 가중치를 상태 가중치로 변환한 후, 초기 할당 상태의 선택뿐만 아니라 전체적인 할당 순서 역시도 상태 가중치를 이용하여 결정한다. State 3의 상태 가중치가 제일 높으므로 00이 할당되고 이에 연결된 에지 $e_{1,3}$, $e_{2,3}$, $e_{3,4}$ 에서 상태 가중치가 가장 높은 State 2를 할당하는데 Cost 계산을 하여 01로 한다. 다음에 State 4과 연결된 에지 $e_{2,4}$ 를 추적하여 State 4를 할당하고, 마지막으로 State 1를 할당한다. 처리는 State 3, 2, 4, 1 순서로 되어 코드값이 다르게 할당된다. 결국, MVES 방법에

서 에지의 가중치를 상태의 가중치로 바꾸면 MSES 방법과 동일한 알고리즘이 된다.

VII. 실험결과

Logic Synthesis Benchmarks^[14] 회로를 이용하여 MVES, EMVES 및 MSES 알고리즘으로 상태할당을 처리한 결과는 표 2, 3, 4와 같다. VI장의 알고리즘 작

표 2. Lakshmikant와 MVES 알고리즘
Table 2. Comparison Lakshmikant with MVES .

	상태수	에지수	(cost/bit)		침이 감소율
			Lakshmikant	제안	
bbtas	6	6	174/3bit	123/3bit	29.31%
dk15	4	6	2013/3bit	1701/3bit	15.5%
s8	5	7	1504/3bit	1354/3bit	10%
tav	4	4	24/2bit	16/2bit	33.33%
train11	11	14	59669/4bit	52896/4bit	11.35%
bbara	10	22	2901/4bit	2870/4bit	1.07%
modulol2	12	12	9174/4bit	5004/4bit	45.45%
train4	4	4	5003/2bit	5003/2bit	0%
planet1	48	63	2795/6bit	1190/6bit	57.42%
sse	16	25	121/4bit	109/4bit	9.1%
sand	32	46	1024/5bit	581/5bit	43.26%
log	17	22	27136/5bit	13944/5bit	48.06%
cse	16	31	4224/4bit	2688/4bit	36.4%

표 3. Lakshmikant와 EMVES 알고리즘
Table 3. Comparison Lakshmikant with EMVES.

	상태수	에지수	(cost/bit)		침이 감소율
			Lakshmikant	제안	
bbtas	6	6	174/3bit	123/3bit	29.31%
dk15	4	6	2013/3bit	1701/3bit	15.5%
s8	5	7	1504/3bit	1354/3bit	10%
tav	4	4	24/2bit	16/2bit	33.33%
train11	11	14	59669/4bit	52303/5bit	12.34%
bbara	10	22	2901/4bit	2829/4bit	2.48%
modulol2	12	12	9174/4bit	5004/4bit	45.45%
train4	4	4	5003/2bit	5003/2bit	0%
planet1	48	63	2795/6bit	1190/6bit	57.42%
sse	16	25	121/4bit	109/4bit	9.1%
sand	32	46	1024/5bit	581/5bit	43.26%
log	17	22	27136/5bit	13944/5bit	48.06%
cse	16	31	4224/4bit	2688/4bit	36.4%

표 4. Lakshmikant와 MSES 알고리즘
Table 4. Comparison Lakshmikant with MSES.

	상태수	에지수	(cost/bit)		천이 감소율
			Lakshmikant	제안	
bbtas	6	6	174/3bit	123/3bit	29.31%
dk15	4	6	2013/3bit	1701/3bit	15.5%
s8	5	7	1504/3bit	1354/3bit	10%
tav	4	4	24/2bit	16/2bit	33.33%
train11	11	14	59669/4bit	51992/5bit	12.87%
bbara	10	22	2901/4bit	2829/4bit	2.48%
modulo12	12	12	9174/4bit	7895/4bit	13.94%
train4	4	4	5003/2bit	5003/2bit	0%
planet1	48	63	2795/6bit	1190/6bit	57.42%
sse	16	25	121/4bit	109/4bit	9.1%
sand	32	46	1024/5bit	673/5bit	34.28%
log	17	22	27136/5bit	13824/5bit	49.06%
cse	16	31	4224/4bit	2688/4bit	36.4%

표 5. MVES, EMVES, MSES 알고리즘 비교
Table 5. Comparison Between MVES, EMVES, and MSES Algorithms.

	MVES	EMVES	MSES
bbtas	123/3bit	123/3bit	123/3bit
dk15	1701/3bit	1701/3bit	1701/3bit
s8	1354/3bit	1354/3bit	1354/3bit
tav	16/2bit	16/2bit	16/2bit
train11	52896/4bit	52303/5bit	* 51992/5bit
bbara	2870/4bit	* 2829/4bit	* 2829/4bit
modulo12	* 5004/4bit	* 5004/4bit	7895/4bit
train4	5003/2bit	5003/2bit	5003/2bit
planet1	1190/6bit	1190/6bit	1190/6bit
sse	109/4bit	109/4bit	109/4bit
sand	* 581/5bit	* 581/5bit	673/5bit
log	13944/5bit	13944/5bit	*13824/5bit
cse	2688/4bit	2688/4bit	2688/4bit

* 최소의 cost를 갖는 경우

성원리에서 알 수 있듯이 MVES는 상태간의 에지 가중치를 중심으로 할당했기 때문에 특정 에지 가중치가 높은 회로의 경우 효과적이고, EMVES 알고리즘은 시작 상태의 선택이 매우 중요하므로 많은 에지들이 향하는 상태를 먼저 고려함으로 특정 상태에 에지들이 많이 향하는 회로를 구현할 경우에 적용하면 된다. 마지막으로 기술된 MSES 알고리즘은 시작상태의 선택을 각 상태에 연결된 에지들이 많은 경우, 또 그 에지들의 가중치의 합이 하나의 에지 가중치보다 큰 경우에 유용하다. EMVES와 MVES는 Cost면에서는 거의

동일함을 표 5를 통해서 알 수 있다. 그러나 임의의 상태로 향하는 에지의 수가 많은 상태가 하나만 존재한다면 EMVES가 효과적일 것이다. 최종 상태할당 후 얻은 상태 코드를 이용하여 Cost를 구한 다음, Lakshmikant가 제안한 상태할당 알고리즘과 비교한 결과 그림 6과 같이 거의 동일한 코드 길이로 할당되었다. 각각의 표에 표시된 바와 같이 상태수가 많아짐에 따라 스위칭에 의한 전력소비를 더욱 줄일 수 있어 Planet 1과 같이 상태수가 48개인 경우 Lakshmikant의 알고리즘에 비하여 Cost를 57.42% 감소시킬 수 있었다. Train11과 같은 예외적인 경우에 MVES 알고리즘은 동일한 코드의 길이를 갖으며 11.35%의 스위칭 변화율을 감소시킨 반면에 EMVES와 MSES 경우에는 코드의 길이가 1비트 증가하나 12.34% 및 12.87%의 스위칭 변화율을 감소시켰다.

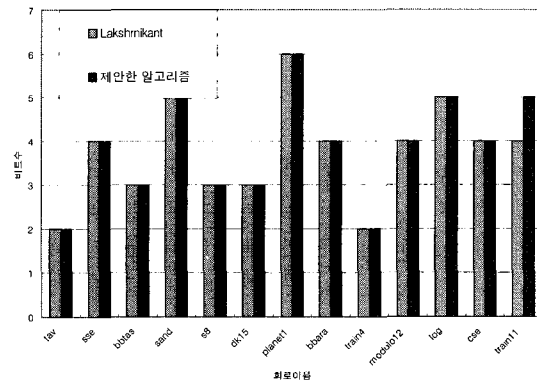


그림 6. 상태 코드의 비트수 비교
Fig. 6. Comparison of Bit Lengths.

VIII. 결론

본 논문에서 제안된 알고리즘은 시스템의 입력에 따라서 동작하는 순서회로에서 상태간의 연결성을 고려하여 스위칭이 작게 발생하도록 상태할당을 하는 것이다. 주어진 회로의 상태 개수가 증가할수록 Lakshmikant 알고리즘과 같이 동일한 코드 길이를 유지하면서 많은 스위칭 변화를 줄일 수 있었다. 그 이유는 Lakshmikant는 회로의 상태 천이 정도를 상태 가중치로 정의하여 단순히 상태 가중치의 크기 순서로 상태를 나열하여 상태코드를 할당하기 때문이다. 특별한 경우 Lakshmikant의 결과와 비교하면 코드길이가 증가하여 면적에 영향을 줄 수가 있으나 그 증가가 1

비트 정도로 미미하므로 실제 전력 소모면에서 효과적일 수 있다.

본 연구에서는 비동기회로에서 발생하는 레이스와 헤즈드 문제와 상태코드 변화에 따른 출력의 영향을 고려하지 않았다. 그러나 코드의 변화가 거의 최소로 변화하므로 출력에 미치는 영향은 거의 없을 것으로 예측할 수 있으며 최적의 면적을 유지하는 상태코드가 존재한다면 면적에 대한 비교가 향후 연구과제이다.

참 고 문 헌

- [1] J. Cortadella, M. Kishinevsky, A. Kondratyev, L. Lavagno, and A. Yakovlev, "A region-based theory for state assignment in speed independent circuits", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 16, no. 8, August 1997.
- [2] L. Benini, and G. De Micheli, "State assignment for low power dissipation", *IEEE Journal of Solid-State Circuits*, vol. 30, March 1995.
- [3] 이기중, 황선영, "다단 논리 회로로 구현된 FSM의 효율적인 상태할당 알고리즘", *한국 정보 과학회 논문지*, 제 18권, 제 2호, 1991년 3월
- [4] 양세양, 김진옥, "유한상태기의 면적 최적화를 한 상태할당 문제의 해석적 접근", *한국 정보 과학회 논문지*, 제 21권 제 7호, 1994년 7월
- [5] M. Koegst, G. Franke, and K. Feske, "FSM state assignment for low power and power estimation under user-specified input sequences", *BEC* 1996.
- [6] Paul Landman, Renu Mehra, and Jan M. Rabaey, "An integrated CAD environment for low-power design", *IEEE Design and Test of Computers*, vol. 13, no. 2, pp. 72-82, June 1996.
- [7] Vinit Kantabutra, and Andreas G. Andreou, "A state assignment approach to asynchronous CMOS circuit design", *IEEE Transactions on Computers*, vol. 43, no. 4, April, 1994.
- [8] 구경희, 조경록, "상태전이확률을 이용한 비동기회로의 저전력 상태할당 알고리즘", *전자공학회는 논문지*, 제 34권 C편, 제 12호, 1997년 12월
- [9] 임세진, 조준동, "스위칭동작 최소화를 통한 저전력 데이터 경로의 최적화" *전자공학회논문지*, 제 36권 C편 제 4호, 1999년 4월
- [10] A. E. A. Almaini and J. F. Miller, P. Thomson, and S. Billina, "State assignment of finite state machines using a genetic algorithm", *IEEE Proc. Computer Digital Techniques*, vol. 142, no. 4, July 1995.
- [11] R. Marculescu, D. Marculescu, and M. Pedram, "Sequence compaction for power estimation: theory and practice", *IEEE Transactions On Computer-Aided Design of Integrated Circuits and Systems*, vol. 18, no. 7, July 1999.
- [12] Akhilesh Tyagi, "Entropic bounds on FSM switching", *IEEE Transactions on VLSI Systems*, vol. 5, no. 4, December 1997.
- [13] Lakshmikanth Bhupathi and Liang-Fang Chao, "Exploiting skewed state probabilities for low power state assignment", *Proceedings of the IEEE International Symposium on Circuits and Systems*, vol. 4, May 1996.
- [14] *Proc. International Workshop on Logic Synthesis*, 1989-1994.

저 자 소 개

金 鍾 洙 (正會員)

Journal of Electrical Engineering and Information
Science vol. 3, no. 1, 1998 參照