論文98-35C-12-3

부분 버스 반전 부호화를 이용한 시스템 수준 전력 최적화

(Partial Bus-Invert Coding for System Level Power Optimization)

辛 英 洙 *, 蔡 洙 翊 *, 崔 起 榮 *

(Youngsoo Shin, Soo-Ik Chae, and Kiyoung Choi)

요 약

본 논문에서는 시스템 수준에서의 전력 최적화를 위한 한가지 방법으로 부분 버스 반전 부호화를 제안한다. 제안하는 방법에서는 버스 부호화가 버스선의 일부분에만 행해지는데, 이것은 버스선 전체를 부호화 함으로써 필요 없는 버스반전이 생기는 것을 피하기 위해서이다. 부분 버스 반전화를 위해서는 어떠한 버스선들을 선택하여 부호화하는가가 중요한데, 본 논문에서는 이러한 선택을 위한 알고리즘을 제안한다. 벤치마크 예제를 통한 모의 실험에서 부분 버스 반전 부호화 방법을 사용해 버스의 천이수를 평균 62.6% 줄일수 있다는 것을 보였다. 또한 제안한 알고리즘의 성능을 평가하기 위해 시뮬레이티드 어닐링 방법과 비교하였다.

Abstract

We present a partial bus-invert coding scheme for system-level power optimization. In the proposed scheme, we select a sub-group of bus lines involved in bus encoding to avoid unnecessary inversion of bus lines not in the sub-group thereby reducing the total number of bus transitions. We propose a heuristic algorithm that selects the sub-group of bus lines for bus encoding. Experiments on benchmark examples indicate that the partial bus-invert coding reduces the total bus transitions by 62.6% on the average, compared to that of the unencoded patterns. We also compare the performance of the proposed heuristic algorithm with that of simulated annealing, which shows that it is highly efficient.

I. 서 론

최근에 들어 휴대전화와 PDA와 같은 휴대용 시스템의 수요가 급증하면서 저전력에 대한 관심이 점점 높아지고 있다. 시스템의 전력을 감소시키는 일은 설계의 여러 단계에서 행해질 수 있지만 상위수준으로 갈수록 시스템 전체를 바라볼 수 있다는 면에서 보다 많은 전력감소를 가져올 가능성이 높다고 할 수 있다. 시스템의 구조적인 측면에서 볼 때 "버스"는 여러 부분 시스템을 연결해주는 중요한 요소인 동시에 많은 전력을 소모하는 부분이라고 할 수 있다. 특히, 버스가 칩과 칩을 연결하는 경우 전력 소모는 시스템 전체 전력소모의 상당부분을 차지한다. 그 이유는 칩 외부의 버스를 구동하기 위해 크기가 큰 구동소자가 필요하고 버스에 큰 용량성 부하가 관계되기 때문이다. 따라서, 버스의 전력소모를 줄이는 일은 시스템의 총 전력을 줄일 수 있는 중요한 방법중의 하나라고 할 수 있다.

버스 부호화 (bus encoding)를 통한 전력감소 방법에서는 버스를 통해 전달되는 정보를 천이 (transi-

* 正會員, 서울大學校 電氣工學部
(Shool of Electrical Engineering, Seoul National University)

※ 이 논문은 1997년도 한국학술진흥재단의 대학부설 연구소 과제 연구비에 의하여 연구되었음
接受日字:1998年8月10日, 수정완료일:1998年11月23日

tion)가 적게 되도록 새로운 형태로 부호화 하게 된다. 그러나, 버스 부호화 방법을 사용하게 되면 부호화/복호화를 위해 부가의 하드웨어가 필요하기 때문에 부가되는 하드웨어에 의한 전력소모가 버스 부호화를 통해 얻을 수 있는 전력감소를 상쇄하지 않도록 주의할 필요가 있다.

본 논문에서는 데이터 버스에 효과가 큰 기존의 버스 반전 부호화[1] (Bus-Invert Coding, BI 부호화)의 개념을 응용하여 데이터 어드레스 버스에 효과가 큰 새로운 버스 부호화 방법인 부분 버스 반전 부호화 (Partial Bus-Invert Coding, PBI 부호화) 방법을 제안한다. 이 방법에서는 기존의 BI 부호화가 버스 전체를 반전시키는데 비해 천이가 같이 발생할 확률이 높은 버스의 일부분만을 반전시킴으로써 보다 높은 천이 감소효과를 얻는다. 따라서, PBI 부호화에서는 버스의 어느 부분을 반전시킬 것인지를 결정하는 것이 대단히 중요하다고 할 수 있는데, 본 논문에서는 이러한 결정을 위해 버스선의 천이 확률과 버스선 사이의 상관성을 고려하는 heuristic 알고리즘을 제안한다. PBI 부호화를 위한 버스 선을 선택하기 위해서는 데이터 주소에 대한 사전 정보가 필요하기 때문에 제안하는 방법은 신호처리, 영상처리와 같이 ASIC과 메모리로 구성되는 시스템을 설계할 때 유용하게 사용될 수 있는 방법이다. PBI 부호화는 데이터 어드레스 버스에 대단히 큰 효과를 나타낸다. 그러나, 개념자체가 일반성을 띠고 있기 때문에 다른 종류의 버스에도 상당한 효과를 나타낼 수 있다.

논문의 구성은 다음과 같다. 다음 장에서는 버스 부호화에 관한 관련 연구에 대해 정리하고 본 논문의 연구동기를 제안하고자 한다. III 장에서는 PBI 부호화에 대해 설명하고 버스 선을 선택하기 위한 알고리즘을 제시한다. IV 장에서는 벤치마크 예제와 실제로 설계된 audio decoder에서 추출한 패턴을 사용한 실험결과를 제시하고, V 장에서는 결론을 맺는다.

II. 관련연구와 연구동기

버스 반전 부호화[1]는 마이크로프로세서를 사용하는 시스템과 같이 범용 시스템의 데이터 버스에 적용될 수 있는 방법이다. 프로세서에서 생성되는 명령어는 메모리 상에 연속적으로 위치해 있는 경우가 많으므로 Gray 부호화[2]와 같이 연속된 숫자사이의 천이수가 1인 방법이 효과적으로 사용될 수 있다. T0[3] 부호화에서는 어드레스 버스의 값이 연속적인 값이라고 판단될 경우 새로운 버스 값을 보내지 않음으로써 천이 수를 더욱더 줄일 수 있다. 수신부쪽에 이러한 정보를 전달하기 위해 추가의 버스 선이 필요하며, 수신부쪽에서는 추가된 버스선의 정보를 이용하여 새로운 주소를 계산하게 된다. 일반적으로 마이크로프로세서의 어드레스 버스는 명령어 주소와 데이터 주소가 같이 사용하기 때문에 Gray 부호화나 T0 부호화가 큰 효과를 나타내지 못하는데, T0_BI, Dual_T0, Dual_T0_BI 부호화[4]와 같이 여러 가지 부호화를 같이 사용하여 효과를 얻을 수 있다.

범용 시스템과는 달리 전용 시스템에서는 버스를 통해 전달되는 정보를 사전에 알 수 있는 경우가 많으므로 정보를 통계적으로 분석하면 천이 수를 효과적으로 줄일 수 있다. The Beach Solution[5]에서는 주소 패턴들의 상관성을 이용하여 버스의 천이수가 최소화되는 방향으로 부호화/복호화하는 방법을 사용한다. 이 방법은 본 논문에서 제시하는 방법과 패턴의 통계적인 정보를 이용한다는 면에서 유사하지만 PBI 부호화가 버스의 일부분만을 부호화 하는데 비해 버스 전체를 부호화 한다는 면에서 차이점이 있다.

데이터 어드레스 버스는 데이터 버스나 명령어 어드레스 버스와는 다른 성질을 나타낸다. 첫째, 데이터 어드레스 버스를 통해 전달되는 패턴들은 명령어 어드레스 버스의 경우와는 달리 연속적인 경우가 많지 않다. 영상 처리 알고리즘과 같이 메모리를 많이 사용하는 응용에서는 연속적인 경우가 극히 드물다. 따라서 Gray 부호화나 T0 부호화 방법은 데이터 어드레스 버스에 적합하지 않다. 둘째, 데이터 버스의 경우 패턴들이 임의의 값을 가진다고 가정하는 것이 타당성을 지니는 반면, 데이터 어드레스 버스의 경우 특정한 버스선의 값이 0이나 1의 값을 가지는 확률이 큰 경우가 많다.

지금까지 제안된 버스 부호화 방법들은 버스 전체를 부호화 하기 때문에 부호화/복호화에 필요한 부가 하드웨어를 고려하게 되면 패턴당 천이수가 작을 경우 부가 하드웨어에서 소모하는 전력이 상당한 비중을 차지할 수가 있다. 예를 들어, 8-bit 버스에서 버스선당 관계되어 있는 용량이 30 pF이라고 가정하자. 칩 내부의 노드당 평균 용량이 0.2 pF이고 버스 부호화 방법을 사용해 패턴당 천이수가 평균 3에서 2로 줄었다

고 가정하자. 만일 부가 하드웨어에서 100번의 천이가 발생했다면 전력소모는 버스 부호화를 이용해 줄어드는 전력의 67%를 차지한다. PBI 부호화 방법의 동기는 버스에서 발생하는 천이 수를 줄이는 동시에 부호화/복호화를 위해 필요한 부가 하드웨어에서 발생하는 천이 수까지 같이 줄이자는 데 있다.

Ⅲ. 부분 버스 반전 부호화

1. 문제 정의

BI 부호화에서는 부호화된 정보를 받는 수신부에 버스가 반전되었는지 그렇지 않은 지의 정보를 송신하기 위해 추가의 버스 선이 필요한데, 이를 invert 선이라고 부른다. 만일 invert 선이 0의 값을 가지면 버스는 반전되지 않은 원래 패턴 값을 나타내며, 1의 값을 가지면 버스는 반전된 값으로 이루어져 있다는 것을 나타낸다. 송신부쪽에서는 버스로 보낼 정보를 반전할것인지의 여부를 결정하기 위해 버스에 현재 실려 있는 패턴과 다음에 보낼 패턴과의 천이 수 (Hamming distance)를 계산하게 된다. 만일 이 값이 버스 폭의 반보다 크면 패턴을 반전하여 송신하고, 그렇지 않으면 반전하지 않는다. 이를 보다 정형화된 형태로 표현해 보기로 하자. 시간 i에서의 n-bit 패턴을 B_i로 표시하기로 하면 BI 부호화 방법에 의해 부호화된 패턴 B_i'은 다음의 식에 의해 주어진다.

$$(B_i', inv)=\begin{cases}(B_i, 0), & \text{if } HD[(B_{i-1}', inv), (B_i, 0)] \le \frac{n}{2} \\ (\overline{B_i}, 1), & otherwise\end{cases} \quad (1)$$

여기서 inv는 invert 선의 값을, $HD(x, y)$는 패턴 x와 y간의 천이 수를 나타낸다. 이 부호화 방법은 패턴당 천이 수를 $n/2$으로 제한시켜 주기 때문에 순간 최대 전력소모 또한 반으로 줄여주는 특징이 있다.

이제, n개의 버스선 중에서 m개의 버스선 만을 BI 부호화하고 나머지 선은 그대로 두는 경우를 생각해보자. 만일 패턴이 시간에 따라 임의의 값을 가지고 버스선 들이 상호 독립이라면 m값이 커지면 커질수록 패턴당 천이수가 줄어든다는 것을 보일 수 있다. 즉, m개의 버스 선만이 BI 부호화된 패턴의 패턴당 천이수의 기대값을 $E(m)$이라고 하면 m이 n보다 작거나 같을 때, 다음 식이 성립한다. 유도과정은 부록을 참조하기 바란다.

$$E(m)=\frac{n}{2}-\sum_{i=\frac{m}{2}+1}^{m}(2i-m-1)C_m^i\left(\frac{1}{2}\right)^m \quad (2)$$

m을 증가시키면서 $E(m)$의 변화를 보면, $E(m)$이 감소하는 것을 볼 수 있는데, 이것은 결국 BI 부호화가 최상의 방법이라는 것을 나타낸다. 그러나, 이 성질은 패턴이 시간에 대해 임의의 값을 가지고 버스 선들이 상호 독립인 경우에만 성립하며, 이 조건으로부터 조금만 벗어나면 $E(m)$은 m에 대해 감소하지 않는다는 것을 알 수 있다. 따라서, 주어진 어떤 패턴들의 집합에 대해 $E(m)$이 최소화되는 m의 값과 m개의 버스 선이 어떻게 구성되는지를 찾아내는 것은 BI 부호화와는 다른 새로운 문제를 형성한다고 볼 수 있다.

2. 개요

PBI 부호화에서는 버스 B를 두 부분으로 분할한다. 즉, n-bit 버스 $B=(b^0, b^1, \ldots, b^{n-1})$가 있고 시간 i에서의 버스선 b^j의 값을 b_i^j, 시간 i에서의 패턴을 $B_i=(b_i^0, b_i^1, \ldots, b_i^{n-1})$라고 하자. PBI 부호화에서는 버스를 선택된 부분 버스 선들의 집합 S와 나머지 버스 선들의 집합 R로 분할한다. 이 때, S에 포함되는 버스 선들은 천이될 확률이 높고 천이가 일어날 상관성 (transition correlation)이 높은 선들로만 구성된다. 따라서, BI 부호화를 S에 포함되는 선택된 버스 선들에 대해서만 시행함으로써 버스전체에 대해 BI 부호화를 시행하는 것보다 높은 천이감소효과를 얻을 수 있다.

일단 버스가 분할되고 난 후, PBI 부호화는 다음과 같은 순서로 이루어진다. S_i와 S_{i-1}' (시간 $i-1$에서의 부호화된 S_{i-1})과의 천이 수를 계산한다. 이 때, invert 선의 천이 수도 포함시킨다. 만일 이 값이 $|S|/2$보다 크면 invert 선을 1로 두고 S_i를 반전시킨다. 이때 R_i는 반전시키지 않는다. 그렇지 않으면 invert 선을 0으로 두고 B_i를 반전시키지 않는다. 이상의 과정을 수식을 이용하여 나타내면 다음과 같다.

$$(S_i', R_i, inv)=\begin{cases}(S_i, R_i, 0), & \text{if } HD[(S_{i-1}', R_{i-1}, inv), (S_i, R_i, 0)] \le \frac{|S|}{2} \\ (\overline{S_i}, R_i, 1), & otherwise.\end{cases} \quad (3)$$

3. 부분 버스 S를 선택하기 위한 알고리즘

PBI 부호화의 성능은 부분 버스 S를 어떻게 선택하느냐에 의해 좌우된다. 그러나, 총 천이 수를 최소화시키는 S를 찾기 위해서는 2^n 가지의 경우를 살펴보

아야 하므로 n값이 커지면 현실적으로 불가능하다고 할 수 있다. 본 논문에서는 버스 선들의 천이 확률과 천이 발생 상관성을 살핌으로써 n가지의 경우 만으로부터 S를 선택하는 알고리즘을 제안한다. 우선, j번째 버스 선에 대해 천이 부호화 (transition encoding) 값(t_i^j)을 다음과 같이 정의한다.

$$t_i^j = \begin{cases} 1, & \text{if } b_{i-1}^j \neq b_i^j \\ 0, & otherwise. \end{cases} \quad (4)$$

즉, t_i^j는 j번째 버스 선이 시간 i에서 천이를 일으키면 1, 그렇지 않으면 0의 값이 된다. j번째 버스 선과 k번째 버스 선의 천이 발생 상관 계수 (transition correlation coefficient) 또는 줄여서 상관 계수 (correlation coefficient)(ρ_{jk})를 다음과 같이 정의한다.

$$\rho_{jk} = \frac{K_{jk}}{\sigma_j \sigma_k} \quad (5)$$

여기서, σ_j는 t^j의 표준편차 (standard deviation)이고, K_{jk}는 t^j와 t^k의 공분산 (covariance)이며 다음과 같이 정의된다.

$$K_{jk} = E\{t_j t_k\} - m_j m_k \quad (6)$$

Algorithm *Select_bus_lines_for_PBI_coding*
begin
L1: Compute the transition probability of each bus line, p_j;
L2: Compute the correlation coefficient of each pair of bus lines, ρ_{jk};
L3: $S = \{\}, R = \{b^0, b^1, \ldots, b^{n-1}\}$;
L4: Initialize the configuration set $C = \{\}$;
L5: Select b^j with the highest transition probability;
L6: $S = \{b^j\}, R = R - \{b^j\}, C = C \cup \{(S, R)\}$;
L7: **while** $R \neq \{\}$ **do**
L8: Select b^j such that $\frac{\sum_{k \in S} \rho_{jk}}{|S|} + p_j$ is a maximum;
L9: $S = S \cup \{b^j\}, R = R - \{b^j\}, C = C \cup \{(S, R)\}$;
L10: **end do**
L11: Count the number of total transitions after PBI coding for each configuration in C;
L12: Select the configuration that yields the minimum number of total transitions;
end

그림 1. 부분 버스 선택 알고리즘
Fig. 1. Selection algorithm of the sub-bus.

$E\{x\}$는 x의 기대값을 m_j는 t^j의 평균값을 의미한다. 즉, ρ_{jk}는 j번째 버스 선과 k번째 버스 선이 항상 같은 시간에 천이를 일으키면 1, j번째 버스 선에 천이가 발생하지 않을 때 마다 k번째 버스 선이 천이를 일으킨다면 -1의 값을 가진다.

부분 버스 선택 알고리즘은 그림 1과 같다.

우선 천이 확률이 가장 높은 버스 선을 선택하고 (L5) 이를 이용해 첫 번째 구성 (configuration, S와 R로 분할된 상태)을 생성한다 (L6). **While** 루프를 매번 반복할 때마다 새로운 버스 선을 선택하여 S에 추가하게 되는데, 이 때 이미 선택된 버스 선들과의 평균 천이 상관계수와 천이 확률의 합이 최대가 되는 버스선 b^j를 선택하고 (L8) 이를 이용해 새로운 구성을 생성한다 (L9). 이렇게 해서 생성된 n개의 구성 각각에 대해 PBI 부호화를 수행한 후 (L11) 총 천이 회수를 구하여 이 값이 최소가 되는 구성을 선택한다 (L12). 알고리즘의 적용 예를 다음 예제에 보였다.

예제 1: 그림 2(a)의 데이터 어드레스 패턴에 대해 식 (4)를 이용해 천이 부호화를 하게되면 그림 2(b)와 같이 된다. 우선 가장 천이가 많이 발생하는 버스선인 b^1(가장 왼쪽 버스 선을 b^0, 가장 오른쪽 버스 선을 b^7이라고 하자)을 선택하여 첫 번째 구성을 생성하고 알고리즘의 while loop내에서 각각 나머지 7개의 구성을 생성한다. 이렇게 생성된 8개의 구성을 그림 3에 보였으며, 이중에서 총 천이회수가 가장 적은 5번째 구성을 선택한다.

B_1	0	0	0	1	1	0	1	0	B_1	0	0	0	0	0	0	0	0
B_2	0	1	1	1	0	0	0	1	B_2	0	1	1	0	1	0	1	1
B_3	0	0	1	1	0	0	0	1	B_3	0	1	0	0	0	0	0	0
B_4	0	1	0	0	1	1	1	0	B_4	0	1	1	1	1	1	1	1
B_5	1	0	1	0	0	1	1	1	B_5	1	1	1	0	1	0	0	1
	(a)									(b)							

그림 2. 데이터 어드레스 패턴의 예 (a) 부호화되지 않은 패턴, (b) 천이 부호화된 패턴
Fig. 2. An example of data address patterns. (a) Unencoded patterns and (b) Transition encoding.

버스 선을 선택하는 기준으로서 이미 선택된 버스 선과의 평균 천이 상관계수와 천이 확률을 동시에 고려 (L8)하는 이유는 함께 천이가 발생할 확률이 높고 천이회수 또한 높을 때 버스 반전의 효과가 극대화되기 때문이다. 그림 4는 32-bit 데이터 어드레스 버스를 사용하는 C 프로그램으로 되어있는 lowpass filter에 대해 제안한 알고리즘을 적용한 예이다. 가장 왼쪽의

점은 부호화를 하지 않은 경우에 해당되고, 가장 오른쪽은 BI 부호화를 적용한 경우에 해당된다. 그림으로부터 제안한 알고리즘을 사용하여 선택된 버스선의 개수가 늘어남에 따라 총 천이회수가 줄어드는 현상을 관찰할 수 있다. 그러나, 약 27개의 버스선 이상을 선택하게 되면 총 천이회수는 반대로 증가하는 것을 볼 수 있는데, 이것은 이전에 선택된 버스 선들과 반대 방향으로 천이가 발생할 확률이 높은 버스 선이 선택되었거나 또는 천이할 확률 자체가 낮은 버스 선이 선택된 결과이다.

	S	R	총 천이회수
구성 1	$\{b^1\}$	$\{b^0, b^2, b^3, b^4, b^5, b^6, b^7\}$	18
구성 2	$\{b^1, b^2\}$	$\{b^0, b^3, b^4, b^5, b^6, b^7\}$	15
구성 3	$\{b^1, b^2, b^4\}$	$\{b^0, b^3, b^5, b^6, b^7\}$	12
구성 4	$\{b^1, b^2, b^4, b^7\}$	$\{b^0, b^3, b^5, b^6\}$	9
구성 5	$\{b^1, b^2, b^4, b^6, b^7\}$	$\{b^0, b^3, b^5\}$	8
구성 6	$\{b^1, b^2, b^3, b^4, b^6, b^7\}$	$\{b^0, b^5\}$	9
구성 7	$\{b^1, b^2, b^3, b^4, b^5, b^6, b^7\}$	$\{b^0\}$	10
구성 8	$\{b^0, b^1, b^2, b^3, b^4, b^5, b^6, b^7\}$	$\{\}$	11

그림 3. 생성된 구성들
Fig. 3. Constructed configurations.

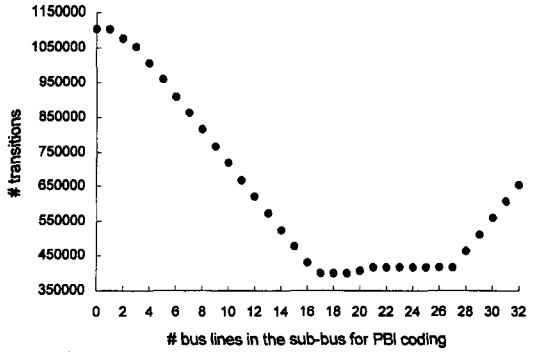


그림 4. PBI 부호화를 적용했을 때의 Lowpass filter의 총 천이회수 변화
Fig. 4. The number of total transitions versus the number of bus lines involved in the PBI coding in an example of lowpass filter.

또 하나 중요한 사실은 선택된 버스선의 개수가 늘어나더라도 총 천이회수가 그다지 변화하지 않는 영역이 존재한다는 사실이다. 따라서, 부호화/복호화에 필요한 부가 하드웨어에서 소모되는 전력을 고려할 경우, 최상의 선택은 PBI 부호화를 위해 선택된 버스선의 수가 작으면서 버스에서의 총 천이수도 상대적으로 작게 되는, 영역의 좌측지점에서 발생할 것이라는 것을 짐작할 수 있다. 이러한 점을 고려하기 위해, 이제 총 천이회수에 부호화/복호화에 필요한 부가 하드웨어가 소모하는 전력을 포함시키기로 하자.

CMOS 회로에서 동적 전력 (dynamic power)은 부하 용량과 천이 빈도 (switching activity)에 비례한다. 이 성질로부터, 총 유효 버스 천이회수 (total effective bus transition) T_{eff}를 다음과 같이 정의한다.

$$T_{eff} = T_{bus} + \frac{C_{\text{int}}}{C_{bus}} T_{\text{int}} \tag{7}$$

여기서, T_{bus}는 버스에서의 총 천이회수, T_{int}는 부호화/복호화 회로에서의 천이회수, C_{int}는 칩 내부에서의 평균 용량, C_{bus}는 버스선당 관계된 총 용량을 나타낸다. 위의 식을 사용하여 제안한 알고리즘의 L11에서 유효 버스 천이회수를 계산하게 되면 부호화/복호화에 필요한 부가 하드웨어의 전력소모를 고려한 천이회수를 구할 수 있게 된다.

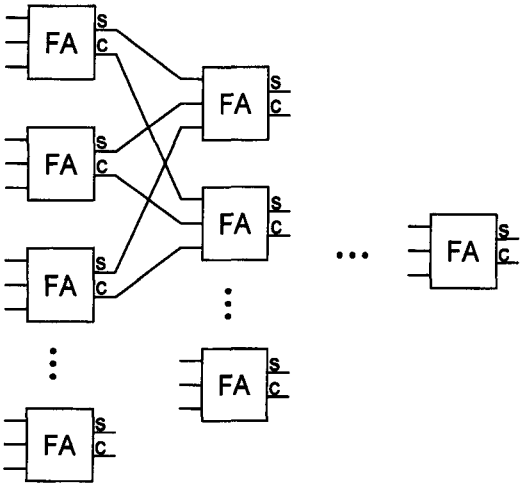


그림 5. Full adder의 tree로 구성된 Majority voter 회로
Fig. 5. Block diagram of a majority voter circuit implemented with a tree of full adders.

T_{int} 값을 예측하기 위해서는 부호화/복호화 하드웨어를 모델링할 필요가 있는데, m개의 버스 선을 반전하기 위해서는 $2m$개의 XOR 회로와 $m+1$개의 입력을 가지는 majority voter 회로가 필요하고, 복호화를 위해서는 m개의 XOR 회로가 필요하게 된다[1]. 그러나, majority voter에서 발생하는 천이회수가 XOR

회로들에서 발생하는 천이회수보다는 훨씬 많기 때문에 전체 천이회수를 majority voter에서 발생하는 천이회수로 근사시키기로 하자. Majority voter는 그림 5와 같이 full adder의 tree로 구성되고, full adder 당 게이트의 수를 x, x개의 입력을 가지는 majority voter를 구성하기 위해 필요한 full adder의 개수를 $N(x)$로 나타내면, T_{int}는 다음과 같이 주어진다.

$$T_{\text{int}} = xN(m+1)a_pL \tag{8}$$

여기서, a_p는 majority voter 입력들의 평균 천이 확률, L은 패턴의 개수를 나타낸다. $N(x)$는 다음과 같이 주어지는데, 유도과정은 부록을 참조하기 바란다.

$$N(x) = x-2 \tag{9}$$

Ⅳ. 모의 실험 결과

PBI 부호화의 타당성을 검증하기 위해 두 가지 종류의 실험을 하였다. 첫 번째는 영상처리와 신호처리에 사용되는 벤치마크 알고리즘들을 이용한 실험이고, 두 번째는 VHDL로 기술하고 LSI 10k gate library로 합성되어 있는 audio decoder[6]를 사용한 실험이다. C_{bus}로는 30 pF[7], C_{int}로는 0.2pF, x는 7을 가정하였다.

1. 영상처리와 신호처리 벤치마크 알고리즘을 이용한 실험

이 실험에서는 32-bit 데이터 어드레스 버스를 가정하였고, 실험을 위해 우선 SPARC 프로세서 상에서 각 알고리즘을 실행하여 생성되는 주소를 Shade라는 프로그램을 이용하여 추출하였다. 추출된 결과를 이용하여 제안한 알고리즘을 적용한 후 PBI 부호화와 BI 부호화를 적용한 실험 결과가 표 1에 있다. PBI 부호화에 의한 천이 감소는 부호화하기 전에 비해 평균 62.6%, 최대 71.8%를 나타낸다. 더욱이, 이러한 감소는 32개의 버스 선 중에서 평균 20개의 버스 선만을 부호화 함으로써 얻어진 결과이다. 표 1의 마지막 열에 있는 데이터는 제안한 알고리즘 대신 시뮬레이티드 어닐링 (SA)을 적용해서 버스 선을 선택한 후 PBI 부호화를 적용한 결과이다. 표로부터 제안한 알고리즘의 성능이 SA에 비해 결코 뒤지지 않는다는 것을 알 수 있다. 제안한 알고리즘의 실행시간은 Ultra 1 워크스테이션상에서 3분 미만이다.

표 1. 벤치마크 예제에 대한 총 천이회수 비교

Table 1. Comparison of the total bus transitions for benchmark examples.

Applications		Unencoded T_{bus}	BI coding		PBI coding		PBI coding with SA	
Name	n		T_{bus}	% Red.	T_{bus}	% Red.	T_{bus}	% Red.
Compress	32	1756468	1066266	39.3	722260	58.9	721864	58.9
Laplace	32	3928218	2377233	39.5	1603476	59.2	1603470	59.2
Linear	32	3948001	2420801	38.7	1227401	68.9	1227401	68.9
Lowpass	32	1101927	656119	40.5	399686	63.7	399622	63.7
SOR	32	2874978	1900735	33.9	1343694	53.3	1343654	53.3
Wavelet	32	2197	1394	36.6	620	71.8	617	71.9
Average				38.1		62.6		62.7

표 2는 부호화/복호화에 필요한 부가 회로에서 발생하는 천이회수를 총 버스 천이회수에 포함하여 실험한 결과이다. 이 경우 PBI 부호화와 BI 부호화에 의한 천이 감소 효과의 차이는 표 1보다 더 커지는데, 그 이유는 PBI 부호화의 경우 전체 버스선 중에서 일부분만을 부호화 하므로 그만큼 부가 회로에 대한 부담이 적기 때문이다.

표 2. 벤치마크 예제에 대한 총 유효 천이회수 비교

Table 2. Comparison of the total effective bus transitions for benchmark examples.

Applications		Unencoded T_{eff}	BI coding		PBI coding		PBI coding with SA	
Name	n		T_{eff}	% Red.	T_{eff}	% Red.	T_{eff}	% Red.
Compress	32	1756468	1145673	34.8	777984	55.7	773465	56.0
Laplace	32	3928218	2554821	35.0	1726454	56.0	1716934	56.3
Linear	32	3948001	2599283	34.2	1395489	64.7	1395489	64.7
Lowpass	32	1101927	706935	35.9	436525	60.4	433850	60.6
SOR	32	2874978	2030708	29.4	1433641	50.1	1433641	50.1
Wavelet	32	2197	1493	32.0	709	67.7	697	68.3
Average				33.6		59.1		59.3

2. Audio decoder를 사용한 실험

이 실험에서는 MPEG-2와 AC-3를 동시에 지원하는 audio decoder를 사용한 실험을 하였다. 설계된 decoder로부터 두 가지의 패턴을 추출하였는데, 첫 번째는 표 3에서 Parser라고 이름 붙여진 패턴이고, 두 번째는 FFT라고 이름 붙여진 패턴이다. 두 패턴 모두 설계된 decoder를 VHDL simulation하여 얻은 결과이다. BI 부호화의 경우 부호화하기 전에 비해 오히려 천이수가 증가하는 것을 볼 수 있는데, 이것은 버스에

서의 천이수가 전체 버스 폭에 비해 상대적으로 적은 데 비해, 부호화/복호화 회로에서는 계속 천이가 발생하기 때문이다.

표 3. Audio decoder에 대한 총 유효 천이회수 비교

Table 3. Comparison of the total effective bus transitions for audio decoder.

Applications		Unencoded	BI coding		PBI coding		
Name	n	T_{eff}	T_{eff}	% Red.	T_{eff}	% Red.	|S|
Parser	16	2563	2675	-4.4	741	71.1	7
FFT	7	1036	1049	-1.3	829	19.9	2

V. 결 론

본 논문에서는 데이터 어드레스 버스의 천이 수를 줄임으로써 시스템의 전력 감소효과를 얻고자 하는 목적으로 부분 버스 반전 부호화 방법을 제안하였다. 제안하는 방법에서는 버스 전체를 부호화 하는 기존 방법들과는 달리 버스의 일부분만을 부호화 함으로써 부호화/복호화에 필요한 부가 하드웨어를 최소화하도록 하였다. 실험결과를 통해 제안하는 방법이 버스의 천이 수를 상당히 줄일 수 있다는 것을 보였다. 또한, 버스 선을 선택하기 위해 제안한 알고리즘의 성능을 시뮬레이티드 어닐링 알고리즘과 비교하였다.

감사의 글

※ 본 논문의 실험을 위해 audio decoder를 제공해 주신 서울대학교 전기공학부 이석준과 성원용 교수님께 감사드립니다.

부 록

1. 식 (2)의 유도

우선 부호화하지 않은 패턴의 천이수에 대한 기대값 $E(0)$와 m개의 버스선을 부호화할때의 천이수에 대한 기대값 $E(m)$ 사이의 차이를 생각한다. 두 값의 차이는 m개의 버스선중에서 i개의 버스선에서 천이가 발생하고, 그 값이 $m/2<i\leq m$을 만족할 때 발생하며, 그 차이는 $i-(m-i+1)$이다. 그 이유는 m개의 버스선을 반전시키게 되면 천이수가 $m-i$가 되고 invert선의 천이수 1이 더해지기 때문이다. m개의 버스선중에서 i개의 버스선이 천이할 확률은 다음 식과 같다.

$$C_m^i(\frac{1}{2})^m\sum_{j=0}^{n-m}C_{n-m}^j(\frac{1}{2})^{n-m}=C_m^i(\frac{1}{2})^m \tag{10}$$

이제, $E(0)$와 $E(m)$의 차이는 다음 식에 의해 주어진다.

$$E(0)-E(m)=\sum_{i=m/2+1}^{m}(2i-m-1)C_m^i(\frac{1}{2})^m. \tag{11}$$

$E(0)=n/2$이므로 이 값을 식 (11)에 대입하면 $E(m)$은 다음과 같다.

$$E(m)=\frac{n}{2}-\sum_{i=m/2+1}^{m}(2i-m-1)C_m^i(\frac{1}{2})^m \tag{12}$$

2. 식 (9)의 유도

그림 3으로부터 majority voter의 첫 번째 단에 사용되는 full adder의 개수는 $x/3$이며, 두 번째 단에 $x/3\cdot 2$개의 입력을 제공한다. 따라서 두 번째단에 사용되는 full adder의 개수는 $x/3\cdot 2/3$이 되고, 이를 반복하면 $N(x)$의 값은 다음과 같다.

$$\begin{aligned}N(x)&=\frac{x}{3}+\frac{x}{3}(\frac{2}{3})+\frac{x}{3}(\frac{2}{3})^2+\ldots+\frac{x}{3}(\frac{2}{3})^{k-1}\\&=x[1-(\frac{2}{3})^k]\end{aligned} \tag{13}$$

여기서 k는 tree의 단수를 나타낸다. 마지막 단은 한 개의 full adder로 구성할 수 있으므로,

$$\frac{x}{3}(\frac{2}{3})^{k-1}=1 \tag{14}$$

이 되고, 이 값을 식 (13)에 대입하면 다음 식을 얻는다.

$$\begin{aligned}N(x)&=x[1-(\frac{2}{3})^{1+\log_{3/2}x/3}]\\&=x-2.\end{aligned} \tag{15}$$

참 고 문 헌

[1] M. R. Stan and W. P. Burleson, "Bus-invert coding for low-power I/O," IEEE Trans. on VLSI Systems, vol. 3, no. 1, pp. 49-58, Mar. 1995.

[2] C. L. Su, C. Y. Tsui, and A. M. Despain,

"Low power architecture design and compilation technique for high-performance processors," in Proc. IEEE COMPCON, Feb. 1994, pp. 209-214.

[3] L. Benini, G. De Micheli, E. Macii, D. Sciuto, and C. Silvano, "Asymptotic zero-transition activity encoding for address busses in low-power microprocessor-based systems," in Proc. Great Lakes Symposium on VLSI, Mar. 1997, pp. 77-82.

[4] L. Benini, G. De Micheli, E. Macii, D. Sciuto, and C. Silvano, "Address bus encoding techniques for system-level power optimization," in Proc. Design, Automation and Test in Europe Conference and Exhibition, Feb. 1998, pp. 861-866.

[5] L. Benini, G. De Micheli, E. Macii, M. Poncino, and S. Quer, "System-level power optimization of special purpose applications: The Beach Solution," in Proc. Int'l Symposium on Low Power Electronics and Design, Aug. 1997. p. 24-29.

[6] S. Lee and W. Sung, "A parser processor for MPEG-2 audio and AC-3 decoding," in Proc. Int'l Symposium on Circuits and Systems, June 1997, pp. 2621-2624.

[7] D. Liu and C. Svensson, "Power consumption estimation in CMOS VLSI chips," IEEE Journal of Solid-State Circuits, vol. 29, no. 6, pp. 663-670, June 1994.

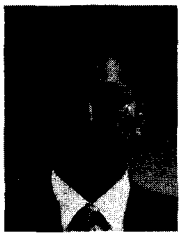
저 자 소 개

辛 英 洙(正會員)

1994年 서울대학교 전자공학과(공학사). 1996年 서울대학교 전자공학과(공학석사). 1996年 ~ 현재 서울대학교 전기공학부 박사과정 재학중. 주관심 분야는 system-level power management, scheduling of embedded real-time systems, system-level analysis/synthesis 등임

崔 起 榮(正會員)

1955年 8月 30日生. 1978年 서울대학교 전자공학과 졸업. 1980年 한국과학원 전기 및 전자공학과 석사. 1989年 미국 Stanford 대학 전기공학과 박사. 1978年 ~ 1983年 (주)금성사 중앙연구소 근무. 1989年 ~ 1991年 미국 Cadence Design System, Inc. 근무. 1991年 ~ 1995年 서울대학교 반도체공동연구소 및 전자공학과 조교수. 1995年 ~ 현재 서울대학교 전기공학부 부교수. 주관심 분야는 CAD, VLSI 설계 등임



蔡 洙 翊(正會員)

1976年 서울대학교 전기공학과 졸업. 1978年 서울대학교 전기공학과 석사. 1987年 미국 Stanford 대학 전기공학과 박사. 1978年 ~ 1982年 공군사관학교 교관. 1987年 ~ 1990年 ZyMos Corporation, 대우통신 근무. 1990年 ~ 현재 서울대학교 전기공학부 부교수. 주관심 분야는 영상 신호처리, VLSI 시스템 설계, 초저전력 회로 설계 등임