

shift-and-add 구조와 연산 하드웨어 공유를 이용한 효율적인 FIR 필터 구현

고 방 영, 한 호 산, 송 태 경
서강대학교 전자공학과
전화 : 02-707-3007 / 핸드폰 : 016-350-8907

Implementation of efficient FIR filter using shift-and-add architecture and shared hardware

Bang Young Goh, Ho San Han, Tae Kyong Song
Dept. of EE, Sogang University
E-mail : tksong@ccs.sogang.ac.kr

Abstract

In this paper, we present an area-efficient programmable FIR digital filter using canonic signed-digit(CSD) coefficients, in which the number of effective nonzero bits of each filter coefficient is reduced by sharing the shift and add logics for common nonzero bits between adjacent coefficients. Also, unused shift and add logics for a low-magnitude coefficient are reassigned to an appropriate high-amplitude coefficient.

In consequence, the proposed architecture reduces the hardware area of a programmable FIR filter by about 24% and improves performance about 6-7dB compared to other multiplierless FIR filters with powers-of-two coefficients.

I. 서론

디지털 신호 처리 분야에서 빈번하게 사용되는 FIR 필터는 구현에 있어서 곱셈기의 하드웨어 면적이 많은 영역을 차지한다는 단점이 있다. 이러한 단점을 개선하기 위해 필터 계수를 부호 붙은 2의 승수(signed power-of-two)의 조합을 이용하여 표현함으로써 곱셈기를 shift-and-add 연산으로 대체하는 방식이 사용되

었다. 이때 시프트 연산은 부호 붙은 2의 승수로 표현되는 필터 계수의 비제로 비트(nonzero bit)에 대해서만 수행함으로써 제로 비트에 대한 불필요한 연산을 줄일 수 있다. 또한, 하드웨어 면적을 많이 차지하는 프로그램 가능한 시프터를 고정된 시프터와 간단한 디지털 로직으로 대체하여 프로그램 가능한 구조를 유지하면서 하드웨어 면적을 감소시키는 방식이 제안되었다. 기존의 구조를 좀 더 효율적으로 구현하기 위해 본 논문에서는 큰 값을 갖는 필터 계수의 연산에 작은 값의 연산에서 사용되지 않은 하드웨어가 있는 경우, 이를 사용하여 성능을 개선하였다. 그리고, 이웃하는 필터 계수의 시프트 연산이 하나 이상 동일하게 나타나는 특성을 이용하여 각 필터 계수의 연산 결과를 공유하였다. 따라서, 각 계수에서 동일한 연산이 많이 중복되면 하드웨어 면적을 줄일 수 있으며, 동일한 하드웨어 면적을 사용할 때 기존의 방법에 비해 필터 성능이 개선된다. 이렇게 설계된 필터에 사용된 디지털 로직의 개수를 고려할 때, 기존의 필터보다 하드웨어 면적이 약 1/5정도 감소하게 되고, 필터 성능도 약 6-7dB정도 개선된다. 이러한 결과는 제안된 구조로 설계된 필터를 기반으로 컴퓨터 시뮬레이션을 통해 기존의 필터 성능과 비교함으로써 확인하였다.

II. shift-and-add 필터 구조

2.1 필터 계수의 코딩

필터의 임펄스 응답 $h(n)$ 은 잘 알려진 CSD(canonic signed digit)방식을 이용하여 식 (1)과 같이 부호 붙은 2의 승수의 조합으로 표현될 수 있다.[1]

$$h(n) = \sum_{k=1}^{m-1} s_k \cdot 2^k \quad (1)$$

이때, $s_k \in \{-1, 0, 1\}$, m 은 필터 계수의 양자화 비트 수이다. CSD방식은 식 (2)와 같은 조건이 요구되므로 제약이 없는 경우보다 통계적으로 필터 계수의 비제로 비트 수가 적게 발생한다. 이에 따라 연산 횟수가 감소하므로 효율적으로 필터를 구성할 수 있다.

$$s_i \cdot s_{i+1} = 0, \quad 0 \leq i \leq m-1 \quad (2)$$

2.2 기존의 shift-and-add를 이용한 필터 구조

그림 1은 일반적인 필터 구조이다. 기존에는 이러한 필터의 곱셈기를 고정된 시프터와 멀티플렉서를 이용하여 shift-and-add 연산으로 그림 2와 같이 대체하였다.[2]

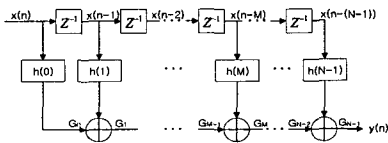


그림 3. 필터의 일반적인 구조(direct form)

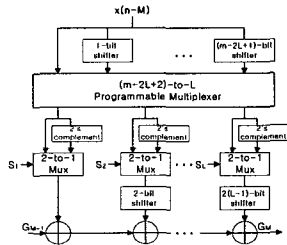


그림 4. M번째 계수에 대한 shift-and-add연산

그림 2의 구조는 식 (3)과 같이 각 비제로 비트가 발생할 수 있는 구간을 나누고 이를 고려한 것이다. 여기서 Z_i 는 각 비제로 비트가 발생할 수 있는 구간이고, 이것의 원소는 식 (1)의 k 를 의미한다.

$$\begin{aligned} Z_1 &= \{0, 1, 2, \dots, (m-1)-2(n-1)\} \\ Z_2 &= \{2, 3, 4, \dots, (m-1)-2(n-2)\} \\ &\vdots \\ Z_L &= \{2(L-1), 2(L-1)+1, \dots, m-1\} \end{aligned} \quad (3)$$

이때, $S_k \in \{-1, 1\}$ ($1 \leq k \leq L$)이고, L 은 각 계수에 허용 가능한 비제로 비트의 개수이다. N 은 필터의 길이이고, 본 논문에서는 이를 짝수로 가정하였다.

III. 제안한 shift-and-add 필터 구조

3.1 두 계수의 연산 하드웨어를 공유하는 구조

FIR 필터의 임펄스 응답은 일반적으로 sinc파형의 형태를 갖는다. 즉, 중앙값은 크지만, 중앙에서 멀어질수록 매우 작은 값을 갖게 된다. 그림 3에서 보듯이 작은 값의 표현에는 비제로 비트가 큰 수에 비해 적게 필요하게 되는데, 이러한 필터의 특성을 하드웨어 구조에 적용할 수 있다.

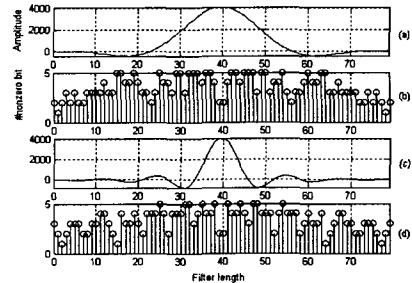


그림 5. (a),(c)임의의 필터에 대한 임펄스 응답. (b),(d)L=5일 때, (a),(c)의 각 필터 계수가 가지는 비제로 비트 수.

그림 4에서 $h(M)$ 은 $h(N/2-M)$ 보다 작은 값이라고 가정하고, 이때 M 은 $0 \leq M \leq N/4$ 를 만족하는 정수라고 정의한다.

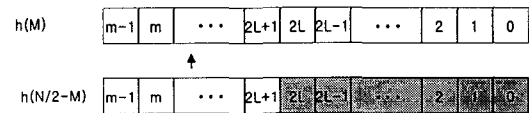


그림 6. 두 계수의 비제로 비트의 발생 구간

$h(M)$ 이 L 개 이하의 비제로 비트로 표현 가능한 경우, $h(M)$ 의 곱셈 연산에 할당된 하드웨어는 사용되지 않는 부분이 생기게 된다. 이러한 경우의 하드웨어 손실을 줄이기 위해 사용되지 않은 하드웨어를 $h(N/2-M)$ 의 곱셈 연산에 사용할 수 있다. 이렇게 하면 큰 수의 필터 계수에 비제로 비트를 더 많이 할당한 것과 동일한 효과를 얻을 수 있다. 이는 필터 계수의 비제로 비트 개수 제한에 따르는 에러를 감소시켜 필터의 성능을 높이게 된다. 이러한 구조로 그림 5를 제안한다.

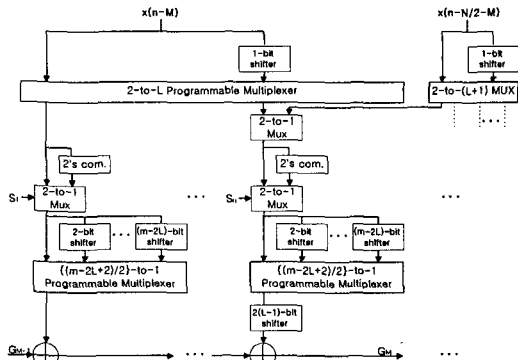


그림 7. 두 계수의 연산 하드웨어를 공유하는 구조

(1) 멀티플렉서 구성

입력을 공유하는데 따르는 하드웨어 증가를 최소화하기 위해 CSD연산의 특징을 이용하여 멀티플렉서를 구성한다. $h(N/2-M)$ 의 비제로 비트 중 L 개는 $h(N/2-M)$ 의 시프트 연산을 위해 할당된 하드웨어를 이용하여 연산하고 $(L+1)$ 번째 비제로 비트는 $h(M)$ 의 남은 하드웨어를 사용한다고 가정한다. $h(M)$ 은 작은 수이므로 $h(N/2-M)$ 의 상위 비트와 중복될 확률이 낮다.

CSD 코딩 방식은 연속적으로 비제로 비트를 허용하지 않기 때문에 $h(N/2-M)$ 에서 $(L+1)$ 번째 할당되는 비제로 비트는 점무늬 구간에서 발생하게 된다. 이를 고려하여 상위 비트의 비제로에 대한 시프트 연산을 수행하는 하드웨어에만 입력을 공유한다. 그리고, 기존에는 미리 시프트된 입력 데이터를 한번의 멀티플렉싱 과정을 통해 선택하여 $(m-2L+2)$ -to- L 의 멀티플렉서가 필요했었다. 그러나, $0,1,2,3,\dots,(m-2L),(m-2L+1)$ 시프트에 대해 $(0,1),2(0,1),\dots,(m-2L)(0,1)$ 과 같이 연산을 수행하면 2 -to- L 과 $(m-2L+2)/2$ -to- L 의 멀티플렉서가 필요하게 된다. 일반적으로 L 이 m 보다 훨씬 작은 값이므로, 멀티플렉싱 과정을 세분화하는 것이 더 적은 경우의 수를 갖게되어 하드웨어를 감소시킬 수 있다.

(2) 각 비제로 비트의 부호 결정

입력 데이터는 시프트 연산을 거치면 데이터 크기가 시프트 횟수만큼 증가하게 된다. 이를 연산 전과 같은 데이터 크기로 연산을 하려면 하위 비트의 데이터를 잘라내야 하고, 이렇게 되면 연산 결과에 에러가 발생한다. 에러를 감소시키기 위해 증가된 데이터 크기로 연산하는 경우에는 그 만큼 하드웨어의 부담이 커지게 된다.

이러한 단점을 보완하기 위해 각 비제로 비트의 부호 결정하는 부분을 그림 5와 같이 시프트 연산 이전에 수행하도록 한다.

3.2 동일한 비제로 비트의 연산을 공유하는 구조

그림 3.(a),(b)의 임펄스 응답에서 보듯이 각 필터 계수는 값의 변화가 크지 않기 때문에 이웃하는 필터 계수의 값이 유사하다는 특징을 갖는다. 따라서, 각 계수의 비제로 비트가 적어도 하나 이상 동일한 위치에 나타나게 된다. 이는 각 계수에서 동일한 시프트연산을 하게 되므로 이러한 중복 연산을 감소시키기 위해 이웃하는 두 필터 계수의 연산을 동시에 수행하고자 한다. 그림 6은 이와 같은 특성을 적용하기 위하여 본 논문에서 제안하는 구조이다.

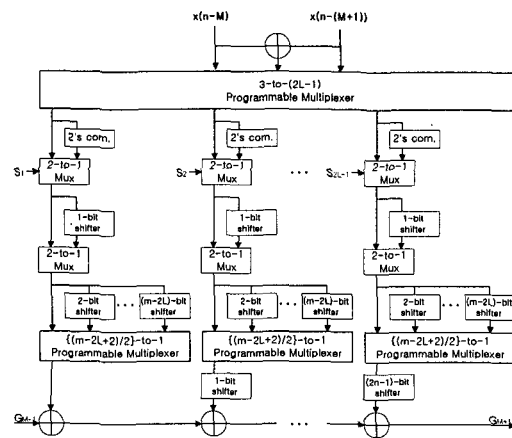


그림 6. 동일한 비제로 비트의 연산을 공유하는 구조

기존에는 두 계수의 연산에 $2L$ 개의 비제로 비트에 대해 연산할 수 있는 하드웨어가 필요했지만, 그림 6과 같이 제안된 방식을 이용하면 $(2L-1)$ 개의 비제로 비트에 대한 연산 하드웨어만으로도 기존과 동일한 연산을 할 수 있다.

또한, 이때 두 계수의 비제로 비트를 동시에 연산하므로 식 (2)와 같은 CSD 코딩의 제약 없이 더 많은 비제로 비트가 중복되도록 필터 계수를 코딩할 수 있다. 이웃하는 필터 계수의 비제로 비트가 많이 중복될수록, 각 계수에 더 많은 비제로 비트를 할당하여 연산을 한 것과 같은 효과를 얻을 수 있다. 따라서 기존보다 감소된 하드웨어로 더 높은 성능의 필터를 구현할 수 있다.

IV. 시뮬레이션 결과

앞에서 살펴본 여러 가지 새로운 방식들을 시뮬레이션을 통해 그 성능을 확인해 보았다. 제안된 구조들은 필터 계수의 변화에 적용가능하므로 시뮬레이션은 다양한 필터에 대해 수행하였다.

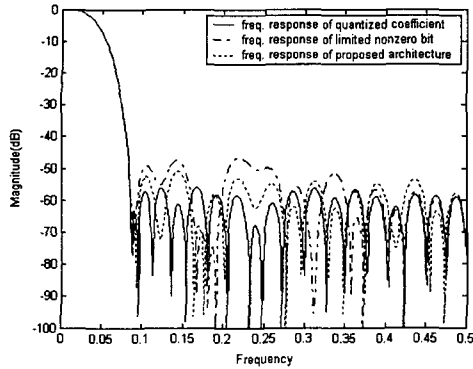
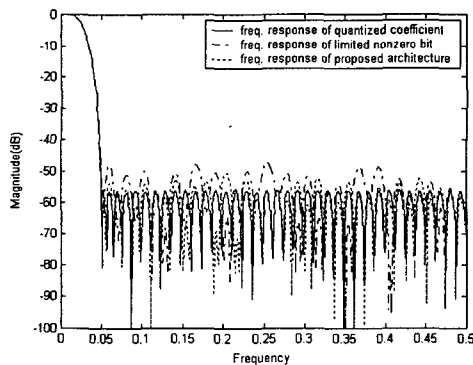


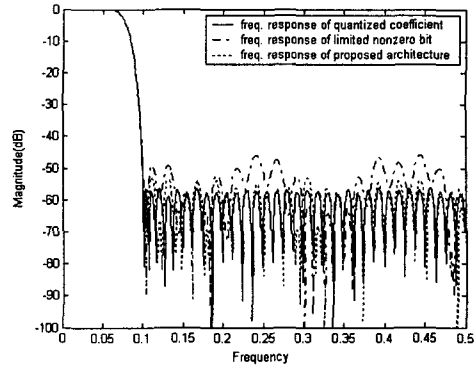
그림 9. 두 계수의 연산 하드웨어를 공유하는 구조에 적용된 필터의 주파수 응답(L=3, N=40, 양자화 9bit)

그림 7은 기존의 구조와 제안된 구조의 필터에 대한 주파수 응답을 비교한 것이다. 그림에서 보듯이 기존의 구조보다 제안된 구조의 필터는 스톱밴드 리플의 최대 값이 46.65dB에서 53.20dB로 6dB이상 개선되었다. 또한 개선된 구조를 이용하여 하드웨어 면적은 기존과 유사하게 구현할 수 있다.

그림 8은 필터 계수를 12bit으로 양자화 하고, 각 계수의 비제로 비트는 전체 필터 길이의 1/4에는 2개를 나머지 큰 계수에 대해서는 3개를 할당하여 구현한 필터의 주파수 응답이다. 결과적으로 (a)의 경우, 그림에서 보듯이 제안된 구조의 필터는 기존의 필터 성능보다 스톱밴드 리플의 최대치가 47.05dB에서 50.61dB로 약 3dB가량 개선되었고, (b)의 경우에도 45.97dB에서 53.15dB로 6dB이상 성능이 개선되었다. 또한 입력을 6비트로 가정할 때, 필터의 하드웨어 면적은 기존보다 약 24%감소하게 된다. 따라서, 기존의 필터 보다 하드웨어 면적을 작게 차지하면서 동시에 성능 향상된 필터를 구현할 수 있다.



(a) 그림 3.(a)의 임펄스 응답을 갖는 필터



(b) 그림 3.(b)의 임펄스 응답을 갖는 필터

그림 8. 동일한 비제로 비트의 연산을 공유하는 구조에 적용된 필터의 주파수 응답(N=80, 양자화 12bit)

IV. 결론

본 논문에서는 FIR 필터 구조에서 서로 같은 시프트 연산 결과를 공유하고, 데이터 크기가 작은 곳에서 연산을 수행함으로써 기존보다 필터 성능이 향상되고 하드웨어 면적도 감소함을 확인하였다. 제한된 비제로 비트만으로 시프트 연산을 수행하던 필터에서 동일한 연산의 반복을 제거함으로써, 이때 남게되는 하드웨어를 이용하여 제한된 비제로 비트수를 증가시켜 더 좋은 성능을 갖는 필터를 설계할 수 있다. 이는 시뮬레이션을 통해 제안된 구조와 기존의 구조를 비교함으로써 필터의 주파수 응답에서 스톱밴드 리플이 6-7dB정도 개선됨을 확인할 수 있었다. 또한, 본 논문에서는 제안된 여러 가지의 필터 구조의 구현에 사용되는 디지털 로직 개수를 고려하여 기존의 필터보다 하드웨어 면적도 적게 차지하는 것을 확인하였다.

참고문헌

- [1] H. Samuelli, "An improved search algorithm for the design of multiplierless FIR filters with powers-of-two coefficients", IEEE Trans. Circuits Syst., VOL. CAS-36, pp. 1044-1047, July 1989
- [2] Woo Jin Oh and Yong Hoon Lee, "Implementation of Programmable Multiplierless FIR Filters with Powers-of-Two Coefficients", IEEE Trans. Circuits Syst., VOL.42, NO. 8, August 1995