한국산학기술학회논문지
Vol. 10, No. 11, pp. 3087-3093, 2009

이진수의 최소 디지트 표현과 공통 부분식 소거법을 이용한 디지털 필터의 성능 개선에 관한 연구

이영석[1*]
[1]청운대학교 디지털방송공학과

Study on Performance Improvement of Digital Filter Using MDR of Binary Number and Common Subexpression Elimination

Youngseock Lee[1*]
[1]**Dept. of Digital Broadcasting and Electronic Engineering, Chungwoon University**

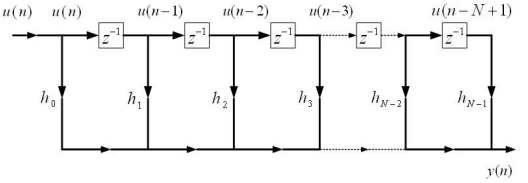
요 약 디지털 필터는 다양한 디지털 신호처리 분야에서 필수 불가결하게 사용되는 기본 요소이다. 디지털 필터는 이진수의 덧셈과 곱셈을 기본 연산으로 하기 때문에 이진수로 나타낸 필터의 계수 및 차수에 의해 연산 속도, 전력 소비 등의 성능이 결정 될 뿐만 아니라 VLSI 기술을 이용하여 디지털 필터가 반도체 칩으로 제작되는 경우, 칩의 면적에 영향을 미치게 된다. 본 연구에서는 디지털 필터의 성능을 개선하기 위하여 2의 보수로 표현되는 이진 필터 계수 데이터들에 대하여 0 디지트의 개수를 최대로 표현할 수 있도록 하는 두 가지 알고리즘을 적용하여 필터의 연산 속도를 증가 시키고, 공통 부분식 소거법을 적용하여 필터의 덧셈 연산을 간소화 시키며 곱셈 연산을 shift 연산으로 대체하여 디지털 필터 설계를 간단히 할 수 있는 방법을 제시하였다. 제안한 방법은 FPGA를 이용한 디지털 필터로 구현하여 성능을 평가하였다.

Abstract Digital filters are indispensible element in digital signal processing area. The performance of digital filter based on adding and multiplying operation, such as computational speed and power consuming is determined by the orders and coefficients of filter which has on effect area of semiconductor chip when it is implemented by VLSI technology. In this research, in order to performance improvement of digital filter, we proposed the algorithm to speed-up the operation of digital filter associated with the minimum signed digit representation of binary number system and method to simplify the digital filter design associated with common subexpression elimination. The performance of proposed method is evaluated by the computational speed and design-simplicity by experimental implemented digital filter on FPGA.

Key Words : Digital filters, Digital signal processing, Minimum signed-digit representation, Common subexpression elimination, FPGA

1. 서론

디지털 필터는 디지털 신호 처리 분야에서 필수불가결하게 쓰이는 요소이다. 일반적인 디지털 필터의 구조는 필터의 안정성(stability)을 고려하여 그림 1과 같이 입력 이진신호 $u[n]$이 필터에 인가되면, 필터의 계수인 h_n과 곱해지고 더해진 후 이진신호 $y[n]$을 출력하는 FIR(finite impulse response) 필터의 형태로 나타낸다[1,2].



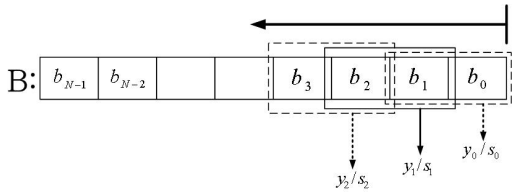
[그림 1] 일반적인 디지털 FIR 필터의 구조

*교신저자 : 이영석(yslee@chungwoon.ac.kr)
접수일 09년 06월 25일 수정일 (1차 09년 10월 27일, 2차 09년 11월 04일) 게재확정일 09년 11월 12일

디지털 필터의 계수는 일반적으로 음수를 포함하고 있으며 이진수 시스템에서 음수는 2의 보수 형태로 나타내는 것이 일반적이다. 또한 음수를 나타내는 2의 보수는 양수를 나타내는 2의 보수 표현에 비하여 1의 개수를 많이 포함하고 있는 것이 일반적이다. 디지털 필터의 계수가 임의의 입력 이진신호 $x[n]$과 연산을 수행하는 경우 입력 신호가 필터의 이진 계수 가운데 1이 존재하는 부분에 shift하며 복사된 값들의 합으로 필터의 이진출력 $y[n]$이 얻어진다. 따라서 필터의 이진 계수에서 1의 개수를 줄일 수 있도록 이진수를 다른 형태로 표현 할 수 있으면 연산량을 줄여 연산 속도를 향상시키는 결과를 갖게 될 것이다. 이와같은 이진수의 표현 방식 가운데 가장 대표적인 숫자 표현 시스템이 CSD (canonical signed digit) 표현 방식이다[3]. 본 연구에서는 이진수의 CSD 표현 방식을 이용한 디지털 필터의 연산 속도를 증진시킬 뿐만 아니라 CSD 표현 방식의 단점인 올림수 전파 지연(carry propagation delay)을 보완하기 위한 이진수의 MSD 표현을 조합하여 빠른 속도로 이진수의 최소 디지트 표현이 가능한 방법을 제안하였고 그 결과를 디지털 필터에 적용하여 성능을 평가하였다.

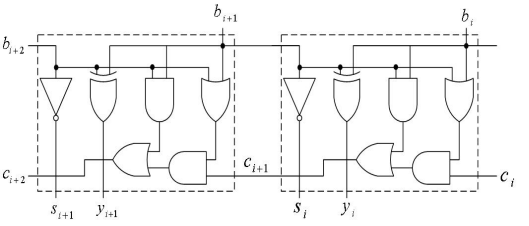
2. 이진수의 CSD 표현

이진수의 CSD 표현 방식은 최소 해밍 거리(minimum Hamming distance)를 갖는 부호 있는 디지트 표현(signed-digit representation)이다[4]. 이진수를 CSD로 표현 할 경우 이진수에서 1의 개수를 최소화할 수 있으며 인접한 비트들 사이에 non-zero 값을 허용하지 않는다. 2의 보수를 CSD 표현으로 나타낼 경우에 각 코드 값은 그림 2와 같이 인접한 비트들 간에 그림 3과 같은 연산 회로를 통하여 생성될 수 있다[5].



[그림 2] CSD 알고리즘의 구성도

그러나 CSD 알고리즘의 문제점은 그림 2와 같은 과정을 통하여 2비트씩 쌍을 이루어 오른쪽에서 왼쪽으로 자리를 이동하여 수행 하는 과정 중에 올림수의 전파(propagation)로 인하여 발생하는 리플(ripple)이 디지털 필터의 입력으로부터 출력이 나오기까지의 지연시간 즉, critical path에 영향을 미친다는 점이다. 그러므로 입력 비트 수가 증가하면 증가할수록 올림수 리플에 의한 전파 지연이 증가하게 된다.

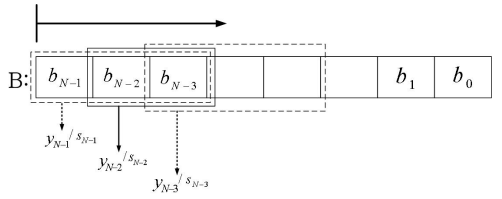


[그림 3] CSD 알고리즘을 수행하는 논리회로

이와 같은 CSD 알고리즘의 단점은 그림 3의 논리 회로에서 점선으로 나타낸 박스 부분이 그림 2의 윈도우에 해당하는 부분으로서 윈도우가 우측에서 좌측으로 이동하는 경우 그림 3의 회로에서 올림수 c_{i+1}이 왼쪽의 점선으로 나타낸 박스로 이동해야만 왼쪽의 회로에서 출력을 나타낼 수 있다. 그러므로 이와 같은 올림수에 의한 전파 지연을 최소화 하면서 CSD 표현 방법에 의해 non-zero 성분을 최소화 할 수 있는 알고리즘의 구성이 요구된다.

3. 이진수의 MSD 표현

Lim 등에 의해 연구된 MSD 표현 방법은 CSD 알고리즘과 달리 그림 4와 같이 이진수의 MSB로부터 LSB로 3비트를 한 쌍으로 하여 1비트씩 이동하면서 변환을 수행한다[5].



[그림 4] MSD 알고리즘의 구성도

Lim의 MSD 표현 방법은 식 (2)와 같이 표현되는 2의 보수를 식 (3)과 같은 표현으로 나타내고자 하는 것으로서 회귀적인(recursive) 방법을 이용하여 구할 수 있으며 표 1과 같은 입력 및 결과 식을 갖는다.

$$x = -b(B-1)2^{B-1} + \sum_{r=0}^{B-2} b(r)2^r,\ b(r) = 0,1 \qquad (2)$$

$$y_m = -c_m(B-1)2^{B-1} + \sum_{r=0}^{B-2} c_m(r)2^r,\ c_m(r) = 0,1 \quad (3)$$

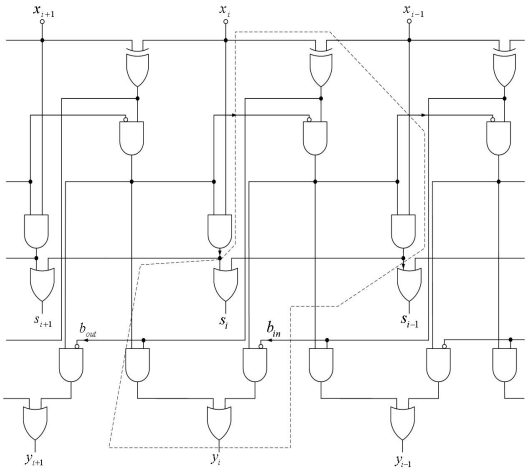
그림 5는 표 2의 논리식을 바탕으로 하여 3비트의 입력 신호 x_{i-1}, x, x_{i+1}에 대한 출력을 나타내고 있다. 출력 신호 s_i는 각 비트에서의 부호를 나타내며 $s_i = 0$이면 음수를, $s_i = 1$이면 양수를 나타낸다. y_i는 0 또는 1로 표현된다. 따라서 입력 신호 x_{i-1}, x, x_{i+1}에 대한 출력은 부호를 나타내는 s_i와 값을 나타내는 y_i로 나타낼 수 있다. 즉, s_i 값이 1이면 해당 비트의 부호는 +이고 0이면 -를 나타낸다.

[표 1] 2의 보수의 MSD 표현

d_i	b_i	b_{i-1}	b_{i-2}	y_i	d_{i-1}
0	0	0	x	0	0
0	0	1	0	0	1
0	0	1	1	1	1
0	1	0	0	-1	1
0	1	0	1	0	1
0	1	1	x	0	0
1	0	0	x	0	0
1	0	1	x	-1	0
1	1	0	x	1	0
1	1	1	x	0	0

예를 들어 $s_i = 0$ 이고 $y_i = 1$이면 해당 비트는 -1을 나타내며 $s_i = 1$ 이고 $y_i = 1$ 이면 해당 비트는 +1을 의미한다. 또한 점선으로 나타낸 부분은 1비트를 MSD를 이용하여 변환시키기 위한 회로 셀을 나타내고 있다.

그러나 MSD 표현 방식을 구현한 그림 5의 회로는 CSD 표현 방식을 구현한 그림 3의 회로와 비교하여 지연 시간이 짧은 장점이 있지만 그림 3의 회로에 비하여 회로의 구성이 복잡한 단점이 있으며 이는 반도체로 회로를 구현하기 위한 VLSI 설계에서 전체 회로의 면적을 증가시키는 결과를 초래하기 때문에 생산 비용이 증가하며 구현된 회로의 전력 소모량 또한 증가하는 단점을 갖고 있다. 따라서 앞서 표현한 두 이진수 표현 시스템의 단점을 보완하고 장점만을 구현하여 이진수의 최소 디지트 표현 방식을 나타내면서도 연산 속도 및 설계 시 연산부의 면적을 최소화하기위한 논리 회로의 구성이 요구된다.



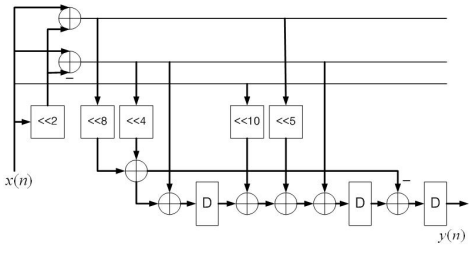
[그림 5] MSD 알고리즘의 논리 회로도

4. 공통부분 표현식 소거법

공통부분 표현식 소거법의 개념은 상수들이 갖고 있는 공통 구조(common structure)를 찾아 재사용(reuse)하는데 있다.

CSD를 이용한 이진수의 최소 디지트 표현 방식에서 가장 많이 발생하는 비트 패턴은 $10\bar{1}$, 101 및 1이므로 임의의 필터 계수가 앞의 3가지 공통 성분을 갖는 다고 가정하면 공통부분 표현식 소거법을 이용하여 필터의 구조를 간단히 할 수 있다. 식 (3)은 3개의 탭을 갖는 FIR 필터의 계수로서 각 필터의 계수는 $10\bar{1}$, 101 및 1의 비트 패턴을 갖고 있는 것을 관측할 수 있다. 그러므로 필터의 계수가 공통 성분을 갖고 있다면 공통성분 표현식 제거법에 의하여 필터의 복잡도를 줄일 수 있다.

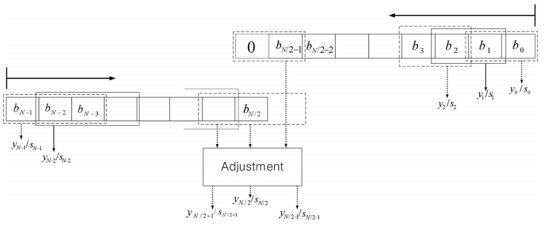
$$\begin{aligned} C_2 &= 0.10101\bar{1}0101\bar{1} \\ C_1 &= 0.10010100101\bar{1} \\ C_0 &= 0.10\bar{1}010\bar{1}0000 \end{aligned} \qquad (3)$$



[그림 6] 공통성분 표현식 제거 법에 의해 구현된 3탭 FIR 필터

5. CSD와 MSD를 이용한 양방향 최소 디지트 표현 알고리즘의 제안

Critical path를 감소시켜 변환 속도를 높이기 위하여 본 연구에서는 양방향 알고리즘을 적용하였다. 적용된 양방향 알고리즘(bidirectional algorithm)은 MSB로부터 LSB 방향으로 진행되는 MSD 표현 방법과 LSB로부터 MSB 방향으로 진행하는 CSD 표현 방법을 조합하여 계수를 변환하는 것으로 정의 할 수 있으며 그림 7과 같이 동시에 두 종류의 변환이 동시에 진행되도록 함으로서 CSD 알고리즘으로부터 발생하는 올림수의 전파 지연 시간을 감소시킬 뿐 만 아니라 디지털 필터 설계 시 필터 계수의 최소 디지트 표현을 위한 회로의 면적을 줄일 수 있는 효과를 가질 수 있다.



[그림 7] 양방향 알고리즘의 구성도

CSD 및 MSD 표현 방식은 모두 최소 해밍 거리를 갖는 변환이기 때문에 두 종류의 변환을 진행하면서 non-zero 성분의 redundancy는 발생하지 않는다. 또한 CSD 알고리즘을 진행하면서 발생하는 올림수 리플의 양을 반으로 줄일 수 있기 때문에 CSD 알고리즘에 의해 발생하는 critical path를 감소시킬 수 있다. 양방향 알고리즘은 다음의 단계에 의하여 진행된다.

(1) 입력을 $B=\{b_{N-1},\cdots,b_1,b_0\}$로 나타내고, 출력 을 $Y=\{y_{N-1},\cdots,y_1,y_0\}$로 나타낸다. 이때, 입력은 N 비트 2의 보수이고, 출력은 N 비트 MSD 표현 방식으로 나타난다고 가정한다.

(2) 입력 B를 B_L과 B_R로 나누어 B_L은 B의 상위 절반 비트들로, B_R은 하위 절반 비트들로 구 성된다. 즉, $B_L=\{b_{N-1},\cdots b_{N/2}\}$ $B_R=\{b_{N/2-1},\cdots,b_0\}$.

(3) B_R의 MSB에 0을 삽입하여 양의 2의 보수로 나타내도록 한다. 즉, $B_R=\{0,b_{N/2-1},\cdots,b_0\}$로 수정한 다음 B_L은 Lim의 MSD 알고리즘을 수행하고, B_R은 CSD 알고리즘을 수행하여 B_L과 B_R의 출력을 각각 Z_L과 Z_R로 나타낸다.

(4) Z_R의 비트 중 MSB를 제외한 값을 출력 Y의 오른쪽 출력 $Y_R^{'}$으로 나타낸다.

(5) $Y_L^{'}$에 Z_L을 할당하고 $Y^{'}=\{Y_L^{'},Y_R^{'}\}$으로 나타낸다.

(6) $Y_L^{'}$의 하위 2비트가 $\{-1,1\}$이면 $\{0,-1\}$로 할당한다.

(7) (6)에 의해 갱신된 $Y_L^{'}$의 LSB와 $Y_R^{'}$의 MSB가 $\{-1,1\}$이면 $\{0,1\}$을 할당한다.

(8) $Y=\{Y_L^{'},Y_R^{'}\}$로 출력을 나타낸다.

그림 7의 양방향 알고리즘의 구성도로부터 B_L의 MSB와 B_R의 LSB가 서로 만나는 부분을 표현하기 위한 보정(adjustment) 부분은 위에서 서술한 양방향 알고리즘의 5단계에서 7 단계에 해당하는 부분으로서 B_L의 MSB와 B_R의 LSB는 총 8개의 경우의 수를 가질 수 있으며 이때 B_L의 MSB와 B_R의 LSB가 각각 $10,01,\bar{1}0,0\bar{1}$의 경우에는 과정 (5)를 통하여 서로 더하여 최종적인 출력으로 나타내는 것이 가능하다.

그러나 B_L의 MSB와 B_R의 LSB가 11의 경우는 알고리즘의 특성상 나타나지 않으며 $1\bar{1},\bar{1}\bar{1},\bar{1}1$의 경우에는 최소 디지트로 이진수를 나타내기 위하여 non-zero 성분이 서로 인접하지 않아야 하기 때문에 $1\bar{1}$은 01, $\bar{1}1$은 $0\bar{1}$ 및 $\bar{1}\bar{1}$은 B_L의 MSB-1 비트를 $\bar{1}$로 하고 B_L의 MSB와 B_R의 LSB 비트에 각각 0을 할당하여 더한 결과로 나타낼 수 있다. 표 2는 VHDL 코드로 양방향 알고리즘을 구현하기 위한 프로그램 흐름도를 나타내고 있다.

[표 2] 양방향 알고리즘의 프로그램 흐름도

START:
1. Assign binary input B into B_R and B_L
2. Insert 0 into MSB+1 bit position og B_R
3. Process MSD algorithm to B_L and CSD algortithm to B_R
4. Assign results of step 3 into Z_L and Z_R
5. Assign Z_R into right side output $Y_R^{'}$ of output Y, except MSB bit of Z_R
6. Assign Z_L into right side output $Y_L^{'}$ of output Y
7. Output $Y=Y_L^{'},Y_R^{'}$
8. If LSBs of $Y_L^{'}$ is $\bar{1}1$, then assign $0\bar{1}$
9. If the updated values of LSB of $Y_L^{'}$ and MSB of $Y_R^{'}$ by Step 8 is $\bar{1}1$, then assign 01
10. Output $Y=Y_L^{'},Y_R^{'}$

END

표 2의 결과로부터 그림 7에 나타낸 양방향 알고 리즘 보정 과정의 $y_{N/2-1}/s_{N/2-1}$, $y_{N/2}/s_{N/2}$및 $y_{N/2+1}/s_{N/2+1}$은 각각 다음과 같은 논리식을 이용하여 나타낼 수 있으며 이때 C 및 D는 각각 양방향 알고리즘을 수행하면서 최종적으로 발생하는 MSD 및 CSD 올림수를 나타낸다.

$$y_{N/2-1} = b_{N/2-1} \oplus D \tag{4}$$
$$s_{N/2-1} = b_{N/2} + b_{N/2+1} + D$$

$$y_{N/2} = \overline{b_{N/2}} \cdot b_{N/2-1} \cdot C + \overline{b_{N/2}} \cdot \overline{D} \cdot (\overline{C} + \overline{b_{N/2-1}}) + b_{N/2} \cdot \overline{b_{N/2-1}} \cdot C$$
$$s_{N/2} = \overline{b_{N/2+1}} + \overline{D} + b_{N/2-1} \cdot D \tag{5}$$

$$y_{N/2+1} = \overline{b_{N/2+1}} \cdot b_{N/2}(b_{N/2-1} \cdot C + D) + b_{N/2+1} \cdot \overline{b_{N/2}} \cdot (C \cdot + \overline{b_{N/2-1}} + D) \tag{6}$$
$$s_{N/2+1} = \overline{b_{N/2+1} \oplus D}$$

6. 실험 및 결과 분석

Lim의 MSD 알고리즘과 CSD 알고리즘을 결합한 2의 보수에 대한 양방향 알고리즘은 16비트의 2의 보수에 적용할 수 있도록 27차의 FIR 필터를 VHDL로 구현하였다. 알고리즘은 은 Xilinx사의 ISE 통합 개발 환경에서 개발하였으며, VHDL로 구현된 알고리즘은 Xilinx사의 XC4VLX100 FPGA칩을 통하여 구현하였다[8,9,10].

실험은 CSD 알고리즘 및 MSD 알고리즘을 단독으로 수행 하였을 때와 양방향 알고리즘을 수행하였을 때 지연 시간을 측정하고 비교하였으며 이때 사용된 FPGA 칩 내부 논리 요소들에 구성을 비교하여 시스템의 복잡도를 비교 분석하였다.

표 3은 실험을 위하여 사용한 디지털 필터의 계수 및 2의 보수, CSD 표현, Lim의 MSD 표현, 양방향 알고리즘 표현을 나타내며 $\bar{1}$은 -1을 나타낸다. 표3으로부터 제안한 양방향 알고리즘의 결과가 이진수의 CSD 표현 방식과 결과가 일치하는 것을 관찰할 수 있다.

[표 3] 디지털 필터 계수, 2의 보수, CSD, Lim의 MSD 및 양방향 알고리즘 표현

필터 계수	2의 보수	CSD	MSD	양방향
0.0004	0000 0000 0000 0100	0000 0000 0000 0010	0000 0000 0000 0010	0000 0000 0000 0010
-0.0006	1111 1111 1111 1010	0000 0000 0000 $\bar{1}$010	0000 0000 0000 0$\bar{1}\bar{1}$0	0000 0000 0000 $\bar{1}$010
-0.0012	1111 1111 1111 0100	0000 0000 000$\bar{1}$ 0100	0000 0000 0000 $\bar{1}\bar{1}$00	0000 0000 000$\bar{1}$ 0100
-0.0008	1111 1111 1111 1000	0000 0000 0000 $\bar{1}$000	0000 0000 0000 $\bar{1}$000	0000 0000 0000 $\bar{1}$000
0.0007	0000 0000 0000 0111	0000 0000 0000 100$\bar{1}$	0000 0000 0000 100$\bar{1}$	0000 0000 0000 100$\bar{1}$
0.0024	0000 0000 0000 1000	0000 0000 0010 $\bar{1}$000	0000 0000 0010 $\bar{1}$000	0000 0000 0010 $\bar{1}$000
0.0022	0000 0000 0001 0110	0000 0000 0010 $\bar{1}$0$\bar{1}$0	0000 0000 0001 10$\bar{1}$0	0000 0000 0010 $\bar{1}$0$\bar{1}$0
-0.0008	1111 1111 1111 1000	0000 0000 0000 $\bar{1}$000	0000 0000 0000 $\bar{1}$000	0000 0000 0000 $\bar{1}$000
-0.0046	1111 1111 1101 0010	0000 0000 0$\bar{1}$01 0010	0000 0000 00$\bar{1}\bar{1}$ 0010	0000 0000 0$\bar{1}$01 0010
-0.0051	1111 1111 1100 1101	0000 0000 0$\bar{1}$01 0$\bar{1}$01	0000 0000 0$\bar{1}$01 00$\bar{1}\bar{1}$	0000 0000 0$\bar{1}$01 0$\bar{1}$01
0	0000 0000 0000 0000	0000 0000 0000 0000	0000 0000 0000 0000	0000 0000 0000 0000
0.0076	0000 0000 0100 1100	0000 0000 0101 0$\bar{1}$00	0000 0000 0101 0$\bar{1}$00	0000 0000 0101 0$\bar{1}$00
0.0102	0000 0000 0110 0110	0000 0000 10$\bar{1}$0 10$\bar{1}$0	0000 0000 10$\bar{1}$0 10$\bar{1}$0	0000 0000 10$\bar{1}$0 10$\bar{1}$0
0.0026	0000 0000 0001 1010	0000 0000 0010 $\bar{1}$010	0000 0000 0010 0$\bar{1}\bar{1}$0	0000 0000 0010 $\bar{1}$010
-0.0112	1111 1111 1001 0000	0000 0000 $\bar{1}$001 0000	0000 0000 $\bar{1}$001 0000	0000 0000 $\bar{1}$001 0000
-0.0185	1111 1111 0100 0111	0000 000$\bar{1}$ 0100 100$\bar{1}$	0000 0000 $\bar{1}\bar{1}$00 100$\bar{1}$	0000 000$\bar{1}$ 0100 100$\bar{1}$
-0.0086	1111 1111 1010 1010	0000 0000 $\bar{1}$010 1010	0000 0000 0$\bar{1}$0$\bar{1}$ 0$\bar{1}\bar{1}$0	0000 0000 $\bar{1}$010 1010
0.0148	0000 0000 1001 0100	0000 0000 1001 0100	0000 0000 1001 0100	0000 0000 1001 0100
0.0139	0000 0001 0011 1111	0000 0001 0100 000$\bar{1}$	0000 0001 0100 000$\bar{1}$	0000 0001 0100 000$\bar{1}$
0.0214	0000 0000 1101 0110	0000 0001 00$\bar{1}$0 $\bar{1}$0$\bar{1}$0	0000 0001 00$\bar{1}$0 $\bar{1}$0$\bar{1}$0	0000 0001 00$\bar{1}$0 $\bar{1}$0$\bar{1}$0
-0.018	1111 1111 0100 1100	0000 0001 0101 0$\bar{1}$00	0000 0000 $\bar{1}\bar{1}$01 0$\bar{1}$00	0000 0001 0101 0$\bar{1}$00
-0.0574	1111 1101 1100 0010	0000 00$\bar{1}$0 0$\bar{1}$00 0010	0000 00$\bar{1}$0 0$\bar{1}$00 0010	0000 00$\bar{1}$0 0$\bar{1}$00 0010
-0.0534	1111 1101 1110 1010	0000 00$\bar{1}$0 0$\bar{1}$00 1010	0000 00$\bar{1}$0 0$\bar{1}$00 1010	0000 00$\bar{1}$0 0$\bar{1}$00 1010
0.0201	0000 0000 1100 1001	0000 0001 00$\bar{1}$0 1001	0000 0001 00$\bar{1}$0 1001	0000 0001 00$\bar{1}$0 1001
0.1452	0000 0101 1010 1100	0000 10$\bar{1}$0 0$\bar{1}$0$\bar{1}$ 0$\bar{1}$00	0000 0110 0$\bar{1}$0$\bar{1}$ 0$\bar{1}$00	0000 10$\bar{1}$0 0$\bar{1}$0$\bar{1}$ 0$\bar{1}$00
0.2641	0000 1010 0101 0001	0000 1010 0101 0001	0000 1010 0101 0001	0000 1010 0101 0001
0.3129	0000 1100 0011 1001	0001 0$\bar{1}$00 0100 $\bar{1}$001	0001 0$\bar{1}$00 0100 $\bar{1}$001	0001 0$\bar{1}$00 0100 $\bar{1}$001

표 4는 CSD, Lim의 MSD 및 양방향 알고리즘을 VHDL로 구현하고 임의의 입력에 대한 모의실험을 수행하였을 때 입력에 대한 출력의 지연 시간을 나타내고 있다.

[표 4] CSD, Lim의 MSD 및 양방향 알고리즘의 지연 시간 비교

Method	CSD	MSD	양방향
Delay[ns]	7.2	5.5	3.7

표 4의 결과로부터 양방향에서 동시에 변환을 수행하는 알고리즘이 CSD나 Lim의 MSD에 비하여 지연 시간이 감소 한 것을 관측할 수 있다. 그러므로 양방향 알고리즘은 디지털 필터의 구현에 있어 지연 시간을 감소 시켜 필터 시스템의 critical path를 줄일 수 있다. CSD 표현 방법을 이용하여 2의 보수를 나타내는 시스템의 경우 설계 및 구현은 그림 3과 같이 간단하지만 올림수 리플에 의해 critical path가 길어지는 단점이 있고, Lim의 MSD 표현 방법은 그림 5에서 나타내는 바와 같이 올림수 리플의 영향을 CSD 표현 방법에 비하여 상대적으로 덜 받기 때문에 시스템의 critical path는 감소하는 장점을 갖고 있으나, 시스템이 복잡해지면서 발생하는 반도체 내부의 영역의 증가 및 전력 소비 문제가 나타날 수 있다. 따라서 critical path 문제를 해결하면서 시스템의 복잡도를 줄이는 방법으로서 양방향 알고리즘은 유용한 방법이라 할 수 있다. 또한 정확한 시스템의 복잡도를 비교 분석하기 위하여 위의 세 가지 알고리즘을 FPGA를 이용하여 구현하였을 경우 필요한 FPGA 내부의 논리 구성 요소는 표 5와 같이 나타났다.

[표 5] FPGA로 구현된 세 가지 알고리즘의 복잡도 비교

Method	CSD	MSD	양방향
Number of LUT	1536	1893	1682
Number of slice	889	1026	845
Number of IOB	46	84	40

표 5로부터 FPGA 내부의LUT(Look-Up Table)의 사용빈도는 CSD 알고리즘이 나머지 두 개의 알고리즘보다 근소하게 덜 사용한다는 점에서 시스템의 복잡도가 작은 것을 알 수 있으나, 표 3에 나타낸 바와 같이 CSD를 이용한 시스템의 지연 시간이 다른 두 알고리즘에 비하여 길기 때문에 고속 시스템에는 적합하지 않은 특성임을 알 수 있다. 또한 slice의 수나 IOB(Input-Out Block)의 개수는 제안한 양방향 알고리즘이 다른 두 알고리즘에 비하여 작기 때문에 시스템의 지연 시간을 고려하면, 제안한 양방향 알고리즘이 시스템의 속도를 향상 시키면서 더 간단한 하드웨어적인 구조를 갖고 있음을 알 수 있다.

6. 결론

본 연구에서는 디지털 필터의 구현에 있어 필터의 지연 시간을 줄이면서 하드웨어의 복잡도를 줄일 수 있는 알고리즘을 제안하였다. 제안한 방법은 27차의 디지털 필터에 적용하여 기존의 디지털 필터의 계수를 구현하는 방법으로 사용되었던 CSD 알고리즘 및 Lim등 에 의해 제안된 MSD 알고리즘과 필터의 critical path로 인한 지연 시간을 비교하였고 하드웨어로 구현하였을 때의 시스템 복잡도를 FPGA의 하드웨어 복잡도를 나타내는 3가지 구성요소를 이용하여 비교 분석하였다. 분석 결과는 critical path로 인한 지연 시간에 있어서는 제안한 알고리즘이 다른 두 알고리즘에 비하여 약 1.7배에서 2배까지 지연 시간이 줄어드는 것을 확인할 수 있었으며 하드웨어의 복잡도 분석에서도 다른 두 알고리즘에 비하여 비교적 덜 복잡한 구조를 갖고 있는 것을 확인 할 수 있었다. 따라서 제안한 방법은 디지털 필터를 하드웨어적으로 구현하는 경우 작은 시간 지연을 갖고 낮은 복잡도를 갖는 시스템으로 구현할 수 있을 것으로 사료된다.

참고문헌

[1] Backenius, E. Sail, E. Gustafsson, "Bidirectional conversion to minimum signed-digit representation," Circuits and Systems, 2006. ISCAS 2006. Proceedings. 2006 IEEE International Symposium.

[2] Xilinx Corporation, Digital filter Product specifications, 2000.

[3] Y. C. Lim, J. B. Evans, and B. Liu, "Decomposition of binary integers into signed power-of-two terms," IEEE Trans. on Circuits and Systems, vol. 38, no. 6, pp. 667-672, June 1991.

[4] S. K. Das and M. C. Pinotti, "Fast VLSI circuits for CSD coding and GNAF coding," Electronics Letters, vol. 32, no. 7, pp. 632-634, Mar. 1996.

[5] Y. C. Lim and S. Parker, "FIR filter design over a discrete powers-of two coefficient space," IEEE

Trans. Acoust. Speech Signal Processing, vol. 31, no. 3, pp. 583-296, June 1983.

[6] A. G. Dampster and M. D. MacLeod, "Constant Integer Multiplication Using Minimum Adder," IEEE Proceedings G, Vol. 141, No. 5, pp. 407-413, Oct. 1994.

[7] I. Richard and Hartley, " Subexpression Sharing in Filters Using CSD Multipliers," IEEE Trans. on Circuits and Systems, Vol. 43, No. 10, Oct. 1996.

[8] Uwe Meyer Base, Hariharan Naratazan and Encarnacion Castillo, Antonio Garcia, " Faster than the FFT : The Chirp-z RAG-n Discrete Fast Fourier Transform," Frequenz, Vol. 60, 2008.

[9] Uwe Meyer Base, Digital Signal Processing with field Programmable Gate Array, Springer-Verlag, New York, 2nd ed., 2004.

[10] P. K. Dutta and P. B. Dutta Gupta, "Optimization Method for Broad Band Modem FIR Filter Design Using Common Subexpression Elimination," IEEE Instruments and Measurement Technology Conference, Vol. 3, 1994.

이 영 석(Young-Seock Lee) **[정회원]**

- 1993년 2월 : 서울시립대학교 전자공학과 (공학사)
- 1995년 2월 : 서울시립대학교 대학원 전자공학과 (공학석사)
- 1998년 2월 : 서울시립대학교 대학원 전자공학과 (공학박사)
- 1998년 3월 ~ 현재 : 청운대학교 디지털방송공학과 교수

<관심분야>
SOC, 임베디드시스템, 의용생체시스템,VLSI 신호처리