論 文

Systolic Array를 이용한 2-Dimension Convolution 설계 및 제작에 관한 연구

正會員 朴 健 約* 正會員 文 大 哲*

A Study on the Design of 2-Dimension Convolution with Systolic Array and One-chip IC

Gun Pyo PARK*, Dai Tchul MOON* *Regular Members*

要 約 본 논문에서는 실시간 디지털 신호처리에 사용할 수 있는 고속의 CMOS 2-D 콘볼루션을 설계하였고, 또한 VLSI 실현에 적합하도록 회로를 설계하였다.

알고리즘 MAC(multiplier accumulator)를 기본 단위로 modified Booth 알고리즘과 Ling's approach를 이용하여 설계하였으며, 설계된 MAC는 4개의 가산기로 구성된 Booth 디코더로 구현하였다. MAC는 규칙적인 셀의 반복 어레이 형태로 구성하였으며, VLSI 실현에 적합하도록 modularity하고 regularity하게 회로를 설계하였다. 입력되는 데이타와 계수 모두를 2's complement 연산이 되도록 하였다.

그리고 중간 처리 단어를 26비트로 하고 최종 출력을 25비트로 반올림하여 오차없이 수행할 수 있도록 2-D convolution을 설계하였다.

ABSTRACT In this paper we present an architecture for a high speeed CMOS 2 dimension convolution can be for real time digital signal processing. And a synthesis method for designing highly parallel algorithms in VLSI is presented. A parallel MAC(multiplier accumulator) design based on the modified Booth's algorithm and Ling's approach. Each multiplier accumulator consist of four rows adder and four Booth decoder. MAC consist of an iterative array of sequential cells, and are well suited to VLSI implementation as a results of this modularity and regularity. This research adderesses the design of multiplier capable of accepting data in 2's complement notation and coefficient in 2's complement notation.

And a 256 stage filter can be designed without any rounding or truncation errors, with an output wordlength of 26 bits.

I. 서 론

최근의 컴퓨터와 통신 시스템 분야가 급속도로 발전함에 따라서 DSP 또는 image processing과 같은 신호처리 기술이 중요하게 되었고, 그리고 많은 디지털 신호처리 알고리즘은 매우 규칙적이기 때문에 대규모 집적회로로 구현 가능하게 되었다. 또한 실시간 처리 및 많은 양의 데이타 처리를 위하여 디지털 시스템의 고속처리 능력이 필요하게 되었다[1][2].

*湖西大學校 情報通信工學科
Dept. of Information Telecommunication Eng., Hoseo University
論文番號 : 90-83(接受1990. 7. 23)

본 연구는 통신용 DSP 집적회로의 일종인 2 D convolution을 선형 파이프라인 어레이를 이용하여 고속의 연산을 수행할 수 있도록 convolution 에 대한 이론과 알고리즘을 설정하여 실시간 신호처리 시스템에 응용이 가능하도록 회로를 설계하였다.

2-D 콘볼루션은 영상처리, noise 필터링, feature enhancement등에 널리 사용되고, 1 D 콘볼루션은 레이더(radar)와 통신 시스템에서 이용되는 transversal 필터등에 사용되어지고 있다[2][3]. 또한 콘볼루션은 디지탈 신호처리 시스템의 초단부와 종단부에 사용되므로 고속의 실현도가 요구된다. 따라서 본 연구에서 구현된 다바이스는 3×3 mode 2 D와 1×10 mode 1 D의 콘볼루션 또는 correlation을 수행할 수 있도록 dimension selector 회로와 10개의 MAC(multiplier accumulator)를 선형 파이프라인 어레이 형태로 구성하였다.

데이타와 워드 길이는 각각 10 bit와 8 bit이며, 다바이스는 3×3 mode에서 3개의 연속되는 image line들로 입력된 3개의 10 bit path를 갖게 되며, image 데이타를 raster scan하기 위한 2개의 line delay를 사용하였다. 1 D 콘볼루션에 대해서는 3개의 input 데이타를 동시에 입력하게 되도록 설계되었다. Output 워드 비트는 over flow를 방지하고 시스템 확장시 instruction cycle 확장을 용이하도록 26bit로 하였다.

그림 1에서 output section은 하나 이상의 다바이스를 사용하여 콘볼루션 크기를 확장하기 위하여 설정하여 놓았으며, PD(programmable delay)와 고속의 26 bit adder로 구성된다. 또한 PD는 internal convolution 계산값 또는 외부의 26-bit 데이타값을 지연시킬 수 있도록 하였다. 이렇게 설계된 다바이스는 출력 워드길이를 26 비트로 확장하여 256차 필터링시 rounding이나 truncation 에러 없이 사용되어질 수 있다.

Ⅱ. 시스템 구현

다바이스에 적용된 1 D 및 2-D convolution 함수는 다음과 같이 쓸 수 있다.

$$Y(k)=\sum_{i=0} h(i)\ X(k-i) \tag{1}$$

$$Y(k,\ l)=\sum_{i}\sum_{j} h(i,\ j)\ X(k-i,\ l-j) \tag{2}$$

위에서 식(1)은 1×10-stage 1-dimensional convolution에 대한 함수식이고, 식(2)는 3×3 stage 2 dimensional convolution에 대한 함수식이다. Convolution 회로는 그림 1에서 처럼 크게 Dimension Selector, ALU, Control Unit, Buffer and Shift Register Unit, Output Section으로 이루어져 있다.

그림 2에서 Dimension Selector는 1 D와 2 D 동작을 선택하는 부분이고, Buffer & Register는 데이타 재수정과 입력과 처리를 위한 부분이고, Output Section은 고차 kernel로 확장이 용이하도록 설계된 부분이다. ALU는 10개의 MAC (multiplier accumulator) 셀을 파이프라인 어레이로 구성되어 있다. 각 MAC는 10 bit 데이타와 8 bit의 계수로 승산되며 전단 MAC의 partial convolution 22 bit 결과를 현재단의 MAC에서 더해나가도록 되어 있다. MAC의 partial product 복잡도를 줄이기 위해서 (8×10) bit multiplier와 accumulator는 modifled Booth's 알고리즘과 Wallace tree를 이용한 carry save adder 어레이 방식을 사용하였다[4][5][6][7][11]. 각각의 MAC는 4열의 full adder와 4개의 Booth's decoder로 구성되어 있다.

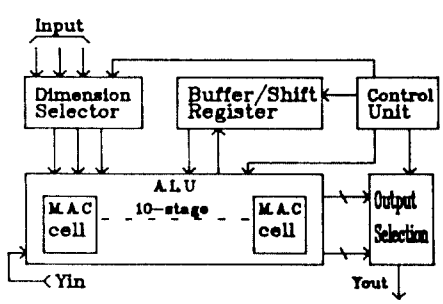


그림 1. 2 D convolution 블럭도.
Bolck diagram of 2 D convolution

Ⅲ. 회로 설계

3.1. Multiplier-Accumulator 설계

디지탈 시스템의 convolution이나 여러 signal processor들의 하드웨어 구현은 입력 sequence와 variable coefficient와의 multiplier 연산시 데이타의 처리방법에 따라 bit serial multiplier, serial parallel multiplier, parallel multiplier등의 방법으로 구현할 수 있다.[4X7X8] Serial한 경우는 전체적으로 space를 감소시킬 수 있으며 parallel multiplier의 경우 연산과정시 pipelining 방법을 이용 partial product에 대한 덧셈 연산을 신속히 처리할 수 있으므로 고속연산에 적합한 장점을 갖는다.

chip의 개발 목적에 부응하기 위하여 ALU 단은 parallel multiplier로 설계하였으며 10bit의 데이타와 8bit의 계수와의 multiplication 연산시 발생되는 partial sum의 수를 줄이기 위해서 modified Booth's 알고리즘을 사용하였다.[5] 또한 효율적인 partial sum 연산을 위해 Wallace tree 알고리즘을 이용한 adder 연산을 취하였고, CLA 설계에서는 Ling's approach를 이용하여 게이트 수를 감소시키고 속도를 증가시킴과 동시에 설계와 레이아웃 및 테스팅을 용이하게 할 수 있도록 modularity하고 regularity하게 회로들을 설계하였다.[9X10] 설계된 MAC 블럭도는그림 2와 같다.

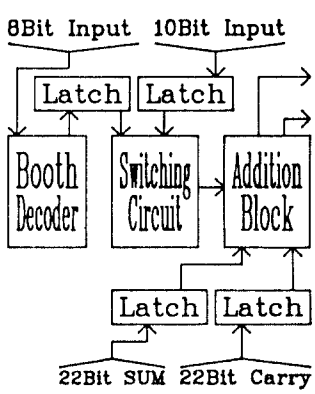


그림 2. MAC 블럭도
Block diagram of MAC

1) Modified Booth's 알고리즘

이 알고리즘은 multiplier의 세개의 bit 씩을 묶어서 (±2, 0, ±1)의 sequence로서 계수를 recoding하는 방법으로, 회로의 복잡도는 증가하나 연산과정은 1/2로 줄어든다[7].

Y를 계수, Y_i를 recoded number라고 할 때

$$Y=\sum_{\substack{i=0\\ odd}}^{n} Y_i \cdot 2^{i-1} \qquad (3)$$

의 식이 성립된다.

여기서 $Y_i = Y_{i-1}+Y_i-2Y_{i+1} \in \{-2, -1, 0, 1, 2\}$

관계가 얻어진다.

따라서 3개의 multiplier를 동시에 test함으로써 multiplication 속도를 증가시킬 수 있다.

식(3)으로부터 Y_i를 recoder로 generation함으로써 얻어지는 결과는 표1과 같다.

표 1. Five level 승산기 레코딩
Five level multiplier recoding

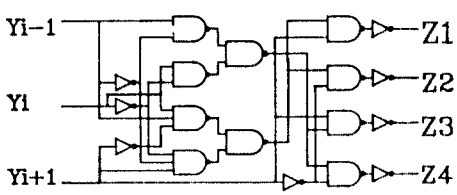
original multiplier			recoded multiplier	action	
Y_{i-1}	Y_i	Y_{i+1}	$Y_i'=Y_{i-1}+Y_i-2Y_{i+1}$		
0	0	0	+0	add	0
0	0	1	+1	add	X
0	1	0	+1	add	X
0	1	1	+2	add	2X
1	0	0	-2	add	-2X
1	0	1	-1	add	-X
1	1	0	-1	add	-X
1	1	1	+0	add	-0

그리고 Y_i를 FLR(five level recoder)로 구성하기 위하여 표 1로부터 modified Booth's 알고리즘의 action에 대하여 sign magnitude number Y_i^3, Y_i^2, Y_i^1(Y_i^3 sign, Y_i^2 '2' magnitude, Y_i^1 '1' magnitude)로 표시할 수 있다. 따라서 코딩된 계수(±2, ±1, 0)를 발생하기 위한 디코더 구성에 대한 디코딩 진리표를 작성하면 표 2와 같다.

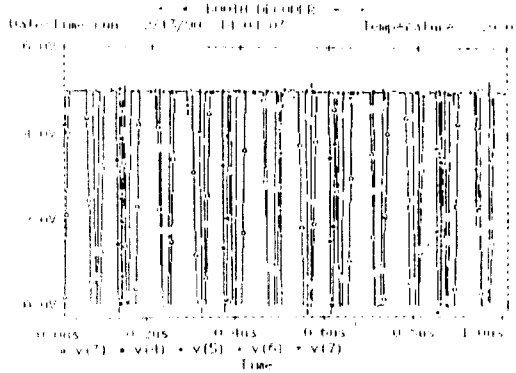
표 2. [illegible]
Decoding truth table

Y_{i+1}	Y_i	Y_{i-1}									
0	0	0	0	0	0	0	0	0	0	0	1
0	0	1	1	0	0	1	1	0	0	0	0
0	1	0	1	0	0	1	0	0	1	0	0
0	1	1	2	0	1	0	X	X	X	X	X
1	0	0	-2	1	1	0	0	1	0	0	0
1	0	1	-1	1	0	1	0	0	0	1	0
1	1	0	-1	1	0	1	X	X	X	X	X
1	1	1	-0	1	0	0	0	0	0	0	1

[illegible] 이용하여 Booth 디코더를 설계하면 그림 3과 같다.



(a) 논리도



(b) 시뮬레이션

그림 3. Booth 디코더
Booth decoder

표 3. recoded 승수에 따른 multiplicand 변화도
Table of multiplicand variation with recoded coefficient

multiplier	multiplicand 변화 (상위 비트부터)										
(예 : 3)	D_{n}	D_{n-1}	D_{n-2}	D_{n-3}	[illegible]	D_2	D_1	D_0	D_{-1}	D_{-2}	D_{-3}
+1 : Z1	$\overline{D}_{n}$	D_{n-1}	D_{n-2}	D_{n-3}	[illegible]	D_2	D_1	D_0	D_{-1}	D_{-2}	D_{-3}
+2 : Z2	D_{n-1}	$\overline{D}_{n-1}$	$\overline{D}_{n-2}$	$\overline{D}_{n-3}$	[illegible]	$\overline{D}_2$	$\overline{D}_1$	$\overline{D}_0$	$\overline{D}_{-1}$	$\overline{D}_{-2}$	$\overline{D}_{-3}$
+3 : Z3	D_{n-1}	D_{n-2}	D_{n-3}	[illegible]	[illegible]	D_2	D_1	D_0	D_{-1}	D_{-2}	0
-2 : Z4	$\overline{D}_{n-1}$	$\overline{D}_{n-2}$	$\overline{D}_{n-3}$	[illegible]	[illegible]	$\overline{D}_2$	$\overline{D}_1$	$\overline{D}_0$	$\overline{D}_{-1}$	$\overline{D}_{-2}$	1
-1 : Z3	1	0	0	0	0	0	0	0	0	0	0

2) Multiplication

Booth 디코더로부터 발생한 recoded 숫자(±2, 0, ±1)의 5개 항과 같은 동작을 수행하는 10 bit multiplicand에 대한 발생회로를 설계하면 표 3과 같다.

위의 term 중에서 −1과 −2는 2's complement 연산을 필요로 하므로 표 3의 항중에서 LSB 부분에 +1을 해주는 과정이 필요하다. 또한 D_i bit의 +2, −2의 경우는 +1, −1의 값을 MSB 방향으로 1 bit 시프트 해줌으로써 2배의 값을 얻도록 하였으며 맨 마지막 D_{11} bit는 D_{10} bit를 sign-extention한 bit이다.

계수의 자연 발생을 위한 기본 MUX 회로의 동작은 표 4와 같다. 그리고 전체적인 멀티플레서 회로는 그림 4와 같이 구성되어진다.

표 4. 입력 계수에 대한 멀티플레서의 동작
Multiplexer operation for input coefficients

Z1	Z2	Z3	Z4	Z5	S
1	0	0	0	0	b_{i-1}
0	1	0	0	0	$\overline{b_{i-1}}$
0	0	1	0	0	b_i
0	0	0	1	0	$\overline{b_i}$
0	0	0	0	0	0

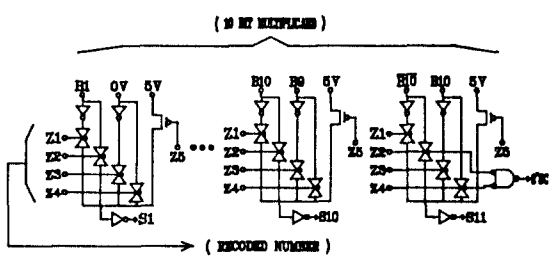


그림 4. 멀티플레서 회로도
Circuit diagram of multiplexer

실제적인 자연 계수발생 회로에서 MSB의 MUX 회로는 sign extention을 위한 회로 구성을 위하여 그림 4에서의 Z1과 Z3 제어에 의해 D_{10}가 그리고 Z2와 Z4 제어에 의해서는 D_{10}이 발생되는 회로를, LSB에서는 B_{i-1}에 D_i이 B_i에 '0' volt 값을 주어야만 실제적용되는 multiplicand가 생성된다. 그러므로 앞에서 설명한 자연계수 발생회로에서는 MSB가 반전된 형태로 발생되도록한 회로이므로 이를 MSB 값으로 변환하여 부호를 확장시키는 회로가 필요하다. 즉 부호확장을 고려한 연산결과를 얻기 위해서는 $\overline{MSB}$ 값을 MSB 값으로 변환시켜 주어야하며, MSB의 값이 (MSB가 n bit 일 때) n+1 bit 이후에 partial sum 계산에 나타나야 하므로 이를 위한 회로가 필요하다.

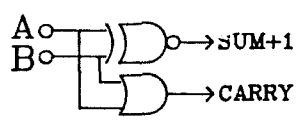
Sign-extetion을 위한 HA+1 회로를 표 5으로부터 구현하면 그림 5와 같다.

표 5. HA+1 진리표
Truth table of HA+1

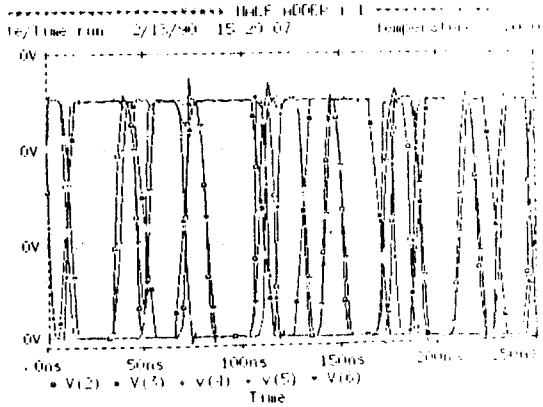
a	b	S1	C1	S2	C2
0	0	0	0	1	0
0	1	1	0	0	1
1	0	1	0	0	1
1	1	0	1	1	1

여기서 S1, C1은 HA 회로에서의 sum과 carry 값이고 S2, C2은 HA+1 회로에서의 sum과 carry 값이다. 따라서 HA+1에 대한 부울리안 함수를 표현하면 다음과 같다.

$$SUM+1=\bar{a}b+a\bar{b}=a\textcircled{\scriptsize\oplus}b=\overline{a\oplus b}$$
$$CARRY=\bar{a}b+a\bar{b}+ab=(\bar{a}+a)b+ab=b+\bar{a}b$$
$$=b+ab+\bar{a}b=a+b$$



(a) HA+1 회로



(b) 시뮬레이션 결과

그림 5. HA+1회로 및 시뮬레이션 결과
Circuit of HA+1 and result of simulation

3) CLA 단의 설계

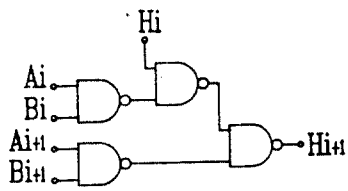
가산은 carry save 방법으로써 좀더 효율적인 partial sum 연산과 속도개선을 위하여 Wallace tree를 이용한 가산기 연산을 취하였다. 입력 워드의 크기를 갖는 가산기의 캐리 지연은 그 캐리들은 CSA 어레이의 마지막 단에서 발생되어 각 단에서 계산해 줌으로써 계산이 가능하다[9][12]. 여기서 CLA 부분은 sum 발생이 좀더 복잡하지만 캐리 생성에 있어서 게이트의 한 단계를 줄임으로써 carry 발생을 효과적으로 할 수 있으므로, 소자 감축 및 속도를 증가시킬 수 있도록 Ling's approach를 사용하였다.[9][13] Ling's 가산기는 기존의 CLA의 G_i항 대신에 carry in 이나 carry out이 발생했다는 의미를 갖는 H_i을 정의하여 H_i와 sum에 대한 순환식을 조합하면 다음과 같다[9][13].

$$H_i = g_i + t_{i-1} \cdot H_{i-1} \qquad (4)$$

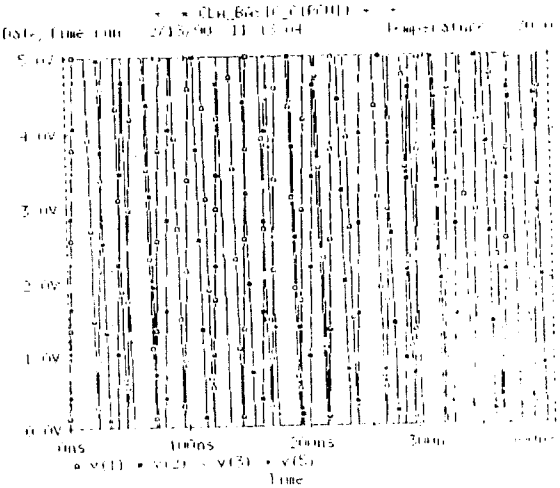
$$S_i = t_i \oplus H_i + g_i \cdot t_{i-1} \cdot H_{i-1} \qquad (5)$$

여기서, $t_i = a_i + b_i$이다.

식(4)로 부터 CLA의 기본회로를 설계하면 그림 6과 같다.



(a) 기본회로



(b) 시뮬레이션 결과

그림 6. CLA의 기본 회로 및 시뮬레이션 결과
Simulation result and basic circuit of CLA

실제적으로 Ling's 가산기는 $(a_i+b_i,\ a_i \oplus b_i)$로 설계하게 시작되나, 복잡한 설계를 가지게 된다. 그러나 한 단계의 1의 순서가 감소됨으로써 표 6 비교표에서 처럼 캐리전파 시간의 감소와 사용되어지는 게이트가 감소하므로 전체적인 칩 면적이 감소한다.

또한, 식 (4)와 (5)의 recurence 관계로 부터 회로를 설계할 때 기존의 CLA처럼 사이즈의 문제가 발생한다. 그래서 4단을 기본으로 하여 recursive하게 다음 단에 이용될 수 있도록 modularity 하게 설계하였다. 그리고 기본 4단의 CLA에서 발생한 마지막 캐리를 다음 단의 초기 캐리로 넘겨서 좀더 효과적인 계산을 하기 위해서 종속 캐리 발생 블럭을 설계하였다[9][12]. 설계된 회로는 그림 7과 같다.

표 6. 기존의 CLA와 Ling's 가산기를 적용한 CLA와의 비교
The contrast of conventional CLA and Ling's CLA

기존 CLA의 carry 발생 수식
$C_0=C_0$
$C_1=g_1+P_1C_0$
$C_2=g_2+P_2g+P_2P_1C_0$
$C_3=g_3+P_3g_2+P_3P_2g_1+P_3P_2P_1C_0$
$C_4=g_4+P_3g_3+P_4P_3g_2+P_4P_3P_2P_1C_0$
Ling's CLA의 carry 발생 수식
$H_0=g_0$
$H_1=g_1+g_0$
$H_2=g_2+g_1+t_1g_0$
$H_3=g_3+g_2+t_2g_1+t_2t_1g_0$
$H_4=g_4+g_3+t_3g_2+t_3t_2g_1+t_3t_2t_1g_0$

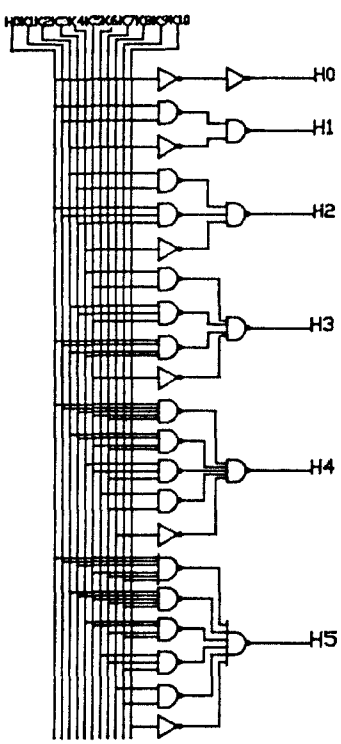


그림 7. 캐리 발생 블럭도
Carry generation block diagram

그리고 최종 sum을 형성하는 기본회로는 그림 8과 같으며 최종 sum 출력에서 나온 각 bit slice sum S_i는 다음 단의 누산기로 전달되고, 이 출력은 다시 CSA의 전가산기로 궤환되어 누산 동작이 가능하게 한다. 또한 10-stage convolution 에 맞는 동작을 하기 위하여 첫번째 MAC에서 계산된 convolution 값이 다음 단의 MAC에서 계산된 convolution 값에 더하여지기 위한 회로를 구현하면 그림 9와 같다.

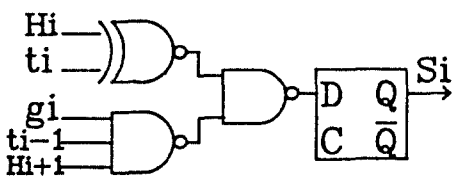


그림 8. 최종 sum 기본회로
Basic circuit of the last sum

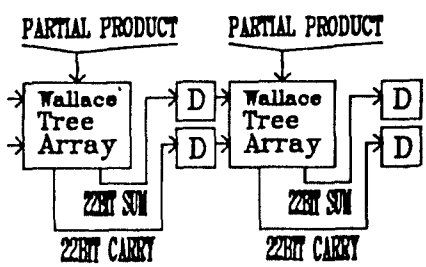


그림 9. Convolution 결과 누산동작 블럭도
Block diagram of accumulating operation for convolution results

최종 sum을 만드는 stage에서 ACC에 adder를 사용하는 대신에 전단의 sum들 즉, 전단에 저장된 sum(22bit)들을 다음단의 MAC에서 adder array로 전달함으로써 두번의 adder 연산을 하는 것을 한번의 연산으로 줄일 수 있다. 따라서 ALU에서의 전파지연은 Booth 디코더에서의 지연과 4단의 adder 어레이에서의 지연만을 포함하므로 전체 MAC 지연시간은 $\tau_{Booth}+4\tau_{add}$가 된다.

3.2. 입·출력 인터페이스

10비트의 데이타를 비트 serial 또는 parallel로 입력하여 parallel로 입력시키거나 그대로 인터페이스와 26비트의 parallel 출력을 비트 serial로 변환시키거나 그대로 parallel로 출력할 수 있게끔 설계하였다. 데이타의 형태는 sign bit와 magnitude bit로 구성되는 2의 보수 형태를 취하여 연산과정에서의 데이타 조작이 간편하게 하였다. 또한 tri-state 인버터를 MUX 구조로 연결하고 선택할 수 있는 기능을 부여하여, serial하게 입력되 데이타의 정렬순서를 MSB 혹은 LSB 중에서 선택할 수 있게 하였다.

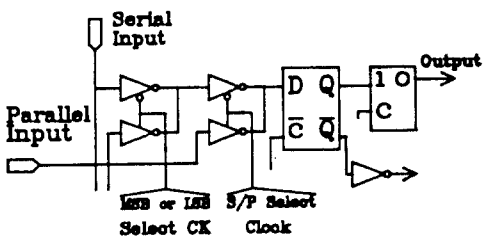


그림 10. 입력 인터페이스
Input interface

3.3. Output Section

Output section은 PD(programmable delay)를 두어 하나 이상의 device를 사용하여 convolution을 쉽게 확장시키기 위하여 외부의 입력신호에 따라 delay를 조작함으로써 여러개의 device 를 쉽게 조합할 수 있도록 설계하였다.

PD는 우선 22bit의 convolution 데이타와 외부의 26bit 데이타를 선택적으로 PD하기 위하여 MUX 회로를 사용하였으며, 선택된 데이타는 1×20 stage일 경우에 1×10 convolution window를 2개 사용하여 얻을 수 있으므로 10 cycle의 지연을 필요로 한다. 그러나 실제로는 입력·출력 및 convolution device 자체에서의 지연을 감안하면 실제적으로 필요한 지연은 7 cycle 정도이면, 1×20 stage convolution이 실현될 수 있으므로 PD는 7 cycle이면 된다. 그리고 6×6 stage로의 확장을 위해서 2, 3, 4, cycle의 delay를 필요로 하므로 PD에서는 총 4개의 control port와 DEMUX 그리고 1개의 MUX (5×1)와 7개의 DF/F 또는 latch를 필요로 한다. PD 전체 블럭도를 그림 11에 나타내었다.

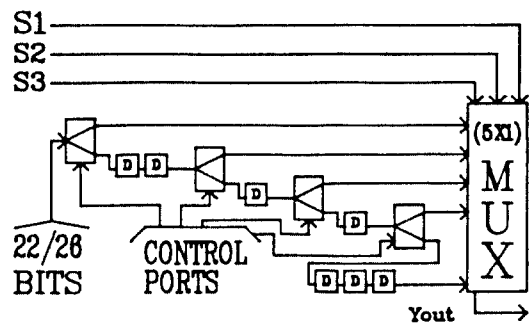


그림 11. PD 블럭도
Block diagram of PD

Output section에서 PD된 후 22bit와 26bit 데이타를 더하기 위하여는 고속의 26bit adder를 구성함에 있어서 CLA를 사용하였으며, MAC 구성 때와 마찬가지 방법으로 캐리 발생 블럭을 구성하여 좀더 효과적인 연산을 취하였다.

3.4. Dimension Selector

3개의 10bit 데이타 입들로 구성된 입력을 3×3 stage 2 D convolution 및 1×10 stage 1 D convolution을 수행할 수 있게 하기 위하여 1 D, 2 D를 위해 선택적으로 입력 데이타를 인접하게 할 수 있게끔 설계하였다. 2 D시에는 m, n값에 따라 3×3 stage가 지정되나 때에 따라 1×10 stage로 할 수 있도록 설계하였다. 그림 12는 dimension selector의 기본 블럭도이다. 1×10 stage 1 D convolution 일 때는 DS unit에서 어떠한 지연도 없이 그대로 MAC로 공급되게 되며, 3×3 stage 2 D convolution 때에서는 B입력을 3 clock cycle을 그리고 C 입력을 6 clock cycle의 delay가 되도록 하였다.

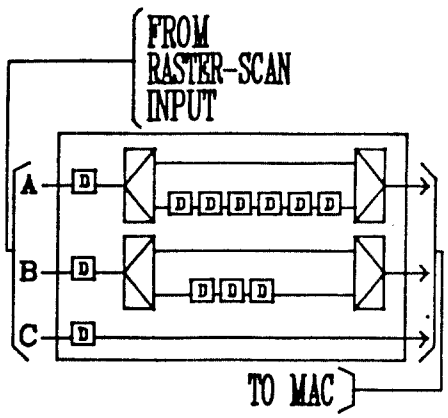


그림 12. Dimension selector의 블럭도
Block diagram of dimension selector

3.5. 제어회로

제어 구조는 machine이 어떤 shift 논리 상태들의 집합을 지원지하는 로직이다. 이들상태들은 레벨들의 계층구조를 갖는다. 즉 major state X는 관련된 minor state X_1, X_2,…,X_n을 갖고, 어떤 레벨로도 내려갈 수 있다. 또한, 활성화 신호를 만들기 위해 필요한 상태를 해석하기 위한 로직을 포함한다.

칩의 동작을 제어하기 위해 필요한 신호는 internal convolution을 제어하기 위한 신호 dimension 및 PD, 그리고 buffer & register를 제어하기 위한 신호등으로 구분되며 controller 회로는 랜덤 논리를 사용하여 구현하였다. 처음 리세트 신호로부터 칩 전체가 클리어된 후에 입력 인터페이스가 동작된다. 이후에 internal convolution을 수행하기 위한 제어 신호를 발생하게 된다. 이 때 사용되어지는 제어 신호들은 binary ripple counter를 사용하여 주기적인 파형을 만들고 그 파형들을 static gate들을 사용, 조합하여 원하는 제어 신호를 생성했다. 그림 13은 제어회로에 대한 논리도이다.

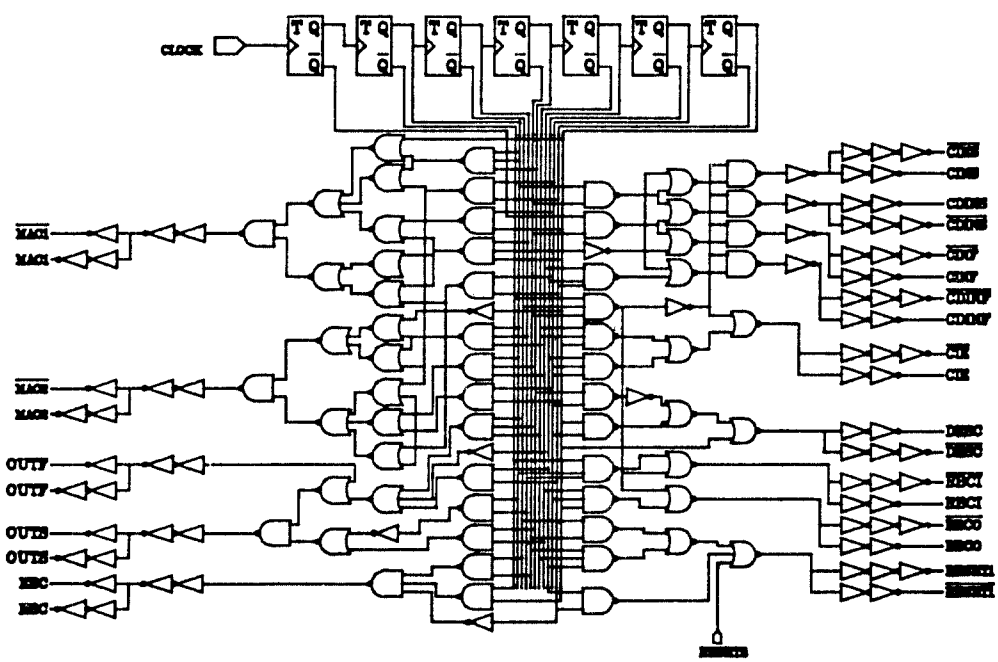


그림 13. 제어회로 논리도
Logic diagram of control circuit

V. 결 론

전체적인 시스템의 구성은 1 dimension convolution 또는 2 dimension convolution을 결정하기 위한 dimension selector를 두고 선택되어진 dimension에 따른 convolution을 실행하기 위하여 10-stage의 MAC 배열을 사용하였으며, 1 stage부터 9-stage의 MAC는 addition block에서 누산동작을 하도록 설계하였다. 또한 MAC에서는 리플 캐리 효과없이 연산 속도를 높이기 위하여 10번째 MAC에서 Ling's approach를 사용하여 CLA 및 CG 블럭을 설계하였다. 계수와 convolution 결과치는 buffer 및 shift register에 저장이 되도록 설계하였고, 또한 output section에서는 22bit의 convolution 계산치와 26bit의 외부 데이타와의 합 계산을 위하여 26bit의 CLA 및 CG 블럭을 설계하였다. 또한 output section단에 PD 회로를 설계하여 하나 이상의 다바이스를 종속연결하여 고차 kernel의 convolution 확장 및 필터링 응용에 용이하도록 설계가 되었다. 그리고 시스토릭 아레이를 이용한 MAC 구성으로 연산속도 및 칩 면적의 최적화를 실현하였으며, modularity하고 regularity하게 module들을 형성하였기 때문에 VLSI실현 및 testablity가 용이하도록 하였다. 현재 설계된 시스템은 레이아웃과 DRC 및 ERC를 끝내고 tape out 단계에 있다.

參 考 文 獻

1. H. T. Kung외 3인, "VLSI System and Computation (A Two-level Pipelined Systolic Array for Convolution)", Springer-Verlag, pp. 255~264, 1981.
2. King-Sun Fu, "VLSI for Pattern Recoginition and Image Processing (One-Dimension Systolic Arrays for Multidimensional Convolution)", Springer-Verlag pp. 9~24, 1984.
3. Ronald W. Schafer, "Discriate-Time Signal Processing , Prentice-Hall, Inc. 1989.
4. Richard J. Higgins, "Digital Signal Processing in VLSI", Prentice-Hall, Inc., 1990.
5. Weste Eshraghian, "Principles of CMOS VLSI Design", Addition Wesley, 1988.
6. Carver Mead & Lynn Conway, "Introduction to VLSI System", Addition Wesley, 1980.
7. N. Zafar, M. N. Konar, T. Oseth, "A 24×24 Bit Parallel Multiplier based a futher Modified Booth's Algorithm", IEEE CUSTOM INTEGRATED CIRCUITS Conference, pp. 289~291, 1985.
8. I Chen Wu, "A Fast 1-D Serial parallel Systolic Multiplier", IEEE Trans. on Comp., Vol. c-36, No. 10, Oct., pp. 1,243~1,247, 1987.
9. R. W. Doran, "Variants of an Improved Carry Look Ahead Adder", IEEE Trans. on Comp. Vol. 37, No. 9, Sep., 1988.
10. H. Ling, "High Speed Binary Adder", IBM J. Res. Develop., Vol. 25, pp. 156, May, 1981.
11. Charles R. Baugh, "A Two's Complement Parallel Array Multiplicaton Algorithm", IEEE Trans. on Comp., Vol. c-22, No. 12, Dec. 1973.
12. 박건표, 문대철, "실시간 디지탈신호 처리용 고속 MULTIPLIER 설계 및 제작에 관한 연구", 통신학회, 추계종합학술발표회 논문집, pp. 203~207, 1990.
13. 노희문, 문대철, "Ling's Approach를 이용한 Carry Look ahead adder 설계 및 제작에 관한 연구", 추계종합학술발표회 논문집, pp. 21~214, 1990.

朴 健 杓(Gun Pyo PARK) 正會員
1965年 3月15日生
1989年 2月 : 湖西大 通信工學科 卒業
1989年 3月~1990年 現在 : 湖西大 情報通信工學科 助工課程
※ 주관심분야 : VLSI 설계 및 CAD.

文 大 哲(Dai Chul MOON) 正會員
1955年 6月 7日生
1979年 2月 : 중전大 電子科 卒業
1981年 8月 : 高麗大 大學院 電子科(碩士)
1988年 2月 : 高麗大 大學院 電子科(工學博士)
1984年 3月~現在 : 湖西大情報通信工學科 助教授