論文91-28A-11-9

60MHz Clock 주파수의 IEEE 표준 Floating Point ALU

(IEEE Standard Floating Point ALU with 60MHz Clock Frequency)

李 溶 錫*

(Yong Surk Lee)

要 約

본 연구 논문에서는 1.0μm의 CMOS를 사용한 60MHz의 clock 주파수에서 32ns(2 clock) 내에 IEEE 표준 단정도와 배정도 부동 소숫점 숫자를 계산할 수 있는 ALU를 제안한다. 종래의 2의 보수 방식이 아닌 지연시간이 9ns인 1의 보수 방식의 54bit 자리올림 선택 가산기(carry select adder)를 사용하고 이 가산기를 첫번 cycle에서는 덧셈 또는 뺄셈에 사용하고 두번째 cycle에서는 같은 가산기로 반올림 반내림에 사용하여 두개의 가산기의 필요성을 한개로 줄여서 회로 지연시간을 극소화하여 32ns의 속도를 얻을 수가 있었다. 이것은 종래의 40MHz 이하의 설계보다 2-5배 빠른 속도이며 설계 목표는 60MHz였으나 상온에서는 80MHz에서도 동작을 확인하였다.

Abstract

This research paper presents an ALU unit using 1.0μm CMOS technology capable of doing IEEE standard single and double precision floating point calculation within 32ns (2 clocks) at 60MHz clock speed. This 32ns speed was achieved by using 9ns 1's complement arithmetic 54 bit carry select adder instead of previous 2's complement adders. On the first cycle, this adder is used for addition or subtraction and the second cycle uses this adder for rounding. This reduces the number of required adders from two to one. Speed improvement is 2 to 5 times compared with previous 40MHz design. Design goal was 60MHz, however, this unit is functioning at 80MHz at room temperature.

I. 서 론

요즘의 고성능 마이크로 프로세서 시스템은 고속의 부동 소숫점 unit를 요구한다. 현재 사용되고 있는 부동 소숫점 unit내에는 숫자 계산을 위한 ALU, 승산기, 제산기가 포함되어 있으며 ALU와는 별도 unit으로 설계되어 있다. 삼각함수 계산 unit도 따로 되어 있으며 승산기와 제산기를 사용하여 계산한다. ALU는 가산, 감산, AND, OR, XOR의 기능을 하며 곱셈과 나눗셈은 따로 승산기와 제산기가 있어서 모든 숫자 계산의 기능을 고속으로 할 수 있다. 이 모든 기능과 설계를 이 논문에서 자세히 논하는 것은 무리이므로 고속의 ALU만을 논하고져 한다. 과거의 발표된 또는 1990년대말 현재 사용되고 있는 ALU는 주파수가 40MHz이하이며 부동 소숫점 연산을 3

*正會員, 인텔株式會社
(Intel Corporation)
接受日字 : 1991年 3月 19日

clock cycle 이상에서만 할 수 있다.[1][2][3][4] 속도를 빠르게 하기 위하여 clock 주파수를 올리려면 트랜지스터의 channel length를 줄여야 하며 설계를 향상하여 회로의 지연시간을 줄임으로써 속도를 더욱 빠르게 할 수 있다. 이 논문에서는 1.0μm의 CMOS를 사용한 60MHz에 동작하는 ALU의 설계와 그 결과를 논하고자 한다.

Ⅱ. IEEE Floating Point 표준과 계산예

1985년에 IEEE의 committee에서 정한 이 표준[5]은 그 후 설계된 대표적인 부동 소숫점 unit인 Intel사의 80387, Motorola사의 68881등 거의 모든 부동 소숫점 unit에 사용되고 있으며 이 표준에서 단정도 또는 배정도 숫자는 그림1에 보인 바와 같이 다음의 3가지로 구성되어 있다.

(1) 1bit 부호 s.

(2) Bias된 지수(e) E=e−bias.

(3) 가수 f=. $b_1b_2\cdots b_{p-1}$.

p는 단정도의 경우 24, 배정도의 경우는 53이다. 실제의 값은(v) 다음과 같이 표시된다.

$$v=(-1)^s\cdot 2^E\cdot(1.f)$$

Bias의 값은 단정도 부동 소숫점의 경우 127, 배정도에서는 1023이며 1과 −1사이의 숫자를 나타내기 위해 필요하다. 단정도의 경우 표시할 수 있는 가장 작은 숫자는 $v_{min}=\pm 2^{-127}$이며 이보다 더 작은 숫자를 나타내기 위해서는 비정규(denormalized) 숫자를 사용해야 한다. 비정규로 표시할 수 있는 가장 작은 숫자는 단정도의 경우 $v_{min}=\pm 2^{-149}$이다. 이 외에도 확장 표준이 있어서 64bit 이상을 사용하여 더 넓은 숫자 범위를 표시할 수 있다.

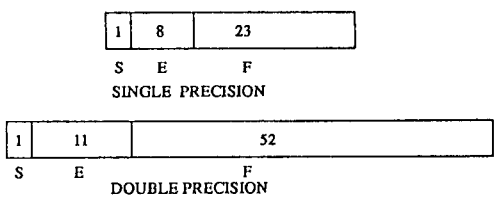

그림 1. IEEE 표준 single과 double precsion floating point
S=sign, E=exponent, F=fraction

Fig. 1. IEEE standard single and double precision floating point.
S=sign, E=exponent, F=fraction.

2개의 부동 소숫점 숫자를 정확히 계산한 후 가수의 숫자가 표준 보다 많아질 경우 표준과 같도록 하기위해 반올림 또는 반내림 (rounding)하는 4가지 방식의 표준이 있다.

(1) Round to nearest. 가까운 쪽으로 반올림 또는 반내림하는 방법이며 양쪽으로 똑같이 가까운 경우에는 least significant bit이 0이 되도록 하는 방법이다. 이 방법이 가장 많이 사용하는 방법으로 이렇게 하기 위해서는 barrel shifter에서 guard, round, sticky bit[6]을 검출하여야 한다.

(2) Direct rounding. $+\infty$, $-\infty$ 또는 0을 향해서 반올림 또는 반내림하는 방법이다.

위의 4가지 방법이 사용자가 부동 소숫점 unit의 명령을 사용하여 선택하도록 되어있다.

두개의 단정도 부동 소숫점 숫자인 (A), (B) 에서

$1.00000000001110010000101\times 2^{15}$ (A)

$1.00000010000000000001001\times 2^{10}$ (B)

(A)−(B)를 계산하기 위하여는 지수 숫자가 같아야 하므로 그 차이만큼 (B)를 5bit 우로 shift해서 (자리맞춤) (A)에서 빼야한다.

```
                                  ---guard bit
                                  ---round bit
                                  -sticky bit
  1.00000000001110010000101|||    ×2^15 (A)
−)0.00001000000100000000000001001 ×2^15 (B)
-------------------------------------------
  0.11111000001010010000010010111   ×2^15 (C)
```

자리맞춤을 할 때 guard, round와 sticky bit을 검출하여 다음 cycle에서 반올림 또는 반내림할 때 사용한다. 위의 계산결과 (C)가 0.111…이 되었으므로 IEEE 표준에 맞추기 위하여 1bit 좌로 shift 하면 (정규화) 다음과 같이 소숫점 왼쪽에 1이 오게된다.

$$1.11110000010100100001001011\underbrace{11}_{27\text{bit}}\times 2^{14}$$

좌로 shift 한 결과 가수가 27bit(IEEE 표준은 23 bit)이 되었으므로 반올림 또는 반내림해서 23bit으로 만들어야 한다. 실제로 이 뺀 결과 (C)가 단정도인 경우 표준의 2배인 46bit까지 나올 수 있으므로 46bit 가산기가 필요하나 이것을 최소화하기 위하여 자리맞춤을 할 때 소숫점 아래 23bif 이하는 guard, round, sticky bit만을 보존하고 실제로는 계산하지 않는다. 이 방법으로 가수가 23bit인 단정도의 경우 25bit 가산기이면 충분하고 가수가 52bit인 배정도의 경우 54

bit 가산기이면 가능하다. 배정도에서 지수값의 차이를 계산하고 정규화할 때 좌로 shift한 만큼 지수 값을 빼 주기 위해서 지수 unit는 11 bit의 가산기가 필요하다. 자리맞춤과 정규화를 하기 위하여 barrel shifter는 좌 우로 0내지 52bit까지 shifte할 수 있도록 설계되어 있다. Leading one detector는 뺄셈의 결과 (C)에서 most significant bit 쪽에 몇개의 0이 있는가를 검출하여 barrel shifter로 보내도록 그림2와 같은 구성도로 설계 되었다.

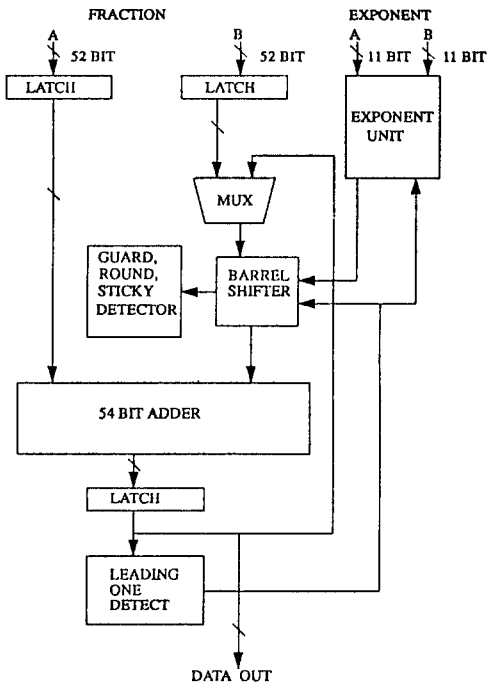

그림 2. 설계된 floting point ALU의 block diagram.
Fig. 2. Designed block diagram of the floation point ALU.

Ⅲ. 회로구성 및 동작원리

1. 구성도

그림2는 이와같은 기능을 할 수 있는 ALU의 요약한 구성도이다. 첫번 clock cycle은 A, B data를 bus line에서 latch하여 barrel shifter와 54bit 가산기를 거쳐 다음 latch까지이며 두번째 cycle에서는 이 data를 다시 barrel shifter에 넣어서 정규화하고 다시 54bit 가산기로 반올림 또는 반내림하도록 설계되었다. barrel shifter는 첫번 cycle에 최고 52bit까지 우로 shift 할 수 있어야 한다. 54bit 가산기는 덧셈과 뺄셈을 할 수 있도록 설계되었다. 대부분의 computer의 ALU는 2의 보수 산술을 사용하여 뺄셈을 하나 이 부동 소숫점 ALU는 1의 보수 산술을 사용하였다. 그 이유는 IEEE 부동 소숫점 표준에 의하면 모든 숫자를 부호-절대값으로 나타내도록 되어 있어 2의 보수를 사용하면 MSB가 1인 경우 부호-절대값으로 변환할 때 1을 빼어 주어야 하는 자리올림 전파 감산기(carry propagation subtractor)가 필요하기 때문이다. 이러한 감산기 회로는 복잡하고 계산이 느리다. 1의 보수 산술에서는 반전하는 것만으로 음수를 간단하고 빠르게 후보-절대값으로 변환할 수 있다. 54bit 가산기에 나온 값은 다시 latch된다. 그러므로 60MHz에서 이 ALU가 충분히 동작하도록 16ns안에 첫번 latch부터 두번째 latch까지 mux, barrel shifter를 거쳐 54bit 가산기로 덧셈이 끝나도록 설계되었다. 두번째 cycle에서는 다시 barrel shifter에 의해서 좌우 shift해야 한다. 54bit 가산기의 출력은 leading one detector가 몇 bit를 좌로 shift해야 할 것인가를 결정하여 barrel shifter에 보내어 진다. 반올림 또는 반내림은 +0또는 +1을 하는 것이므로 54bit 가산기에 의하여 한다. 이때 첫번 cycle에서 결정된 guard, round, sticky bit과 IEEE 표준의 4가지 반올림 또는 반내림 방법 중 선택된 방법에 의하여 +0 또는 +1이 결정되어 진다. 이 ALU는 barrel shifter와 가산기를 두번 사용할 수 있도록 하여 종래의 2개의 가산기의 필요성을 1개로 줄여서 60MHz에서도 충분히 동작하도록 설계되었다.

2. Barrel Shifter와 Guard, Round, Sticky Bit Detector

Barrel shifter는 52bit을 좌우로 shift할 수 있으며 60MHz에서 동작하기 위해서는 회로의 지연 시간을 3.5ns이내로 제한하여야 한다. 두 단계의 mux로 설계되어 있으며 첫번 단계에서는 좌 또는 우로 0, 1, 2, 3, 4, 5, 6, 7bit까지 shift하고 두번째 단계에서는 좌 또는 우로 0, 8, 16, 24, 32, 40, 48bit을 shift하도록 설계되어 있다.(그림3, 4) 예를 들어 45bit을 우로 shift하기 위해서는 첫번 단계에서 5bit을 우로 shift 하고 두번째 단계에서 40bit을 우로 shift하여 도합 45bit을 shift하게 된다. 이것을 첫번 단계에서 40bit, 두번째 단계에서 5bit을 shift하도록 설계할 수도 있으나 이렇게 하면 guard, round, sticky bit detect회로가 복잡해지므로 전자의 방식이 채택되었다. 또한 이러한 선택을 고려할 때는 가장 layout이 편리하고 적은 면적을 차지하도록 하였다. 첫번 단계에서는 52개의

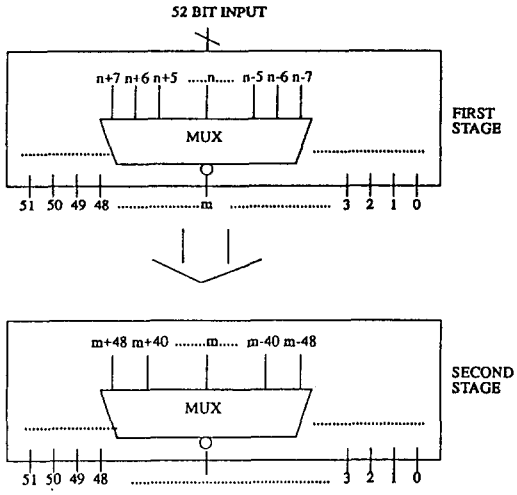

그림 3. 52bit을 좌 또는 우로 shift할 수 있는 barrel shifter
Fig. 3. Barrel shifter for 52bit left or right shift.

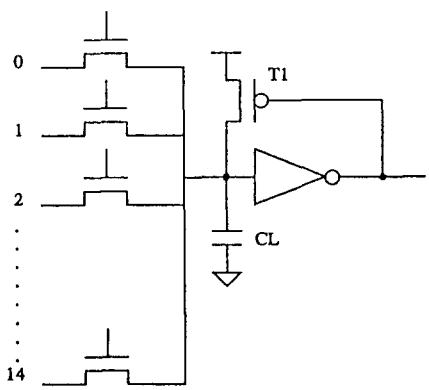

그림 4. Barrel shifter에 사용된 15 input MUX
Fig. 4. 15 input MUX used in barrel shifter.

15 입력 mux가, 두번째 단계에서는 52개의 7입력 mux가 사용되었다. 지연 시간을 최소화하기 위해서는 그림5의 C_L을 최소화하는 것이 중요하다. 특히 15입력 mux는 15개의 트랜지스터가 한 절점(node)에 연결되어 있으므로 접합 용량(junction capacitor)이 대단히 높다. 그러므로 NMOS와 그것을 구동하는 트랜지스터의 크기도 알맞아야 한다. C_L은 약 0.5pF으로 추정되었고 N_1까지의 delay를 0.8ns 이내로 하기 위해서 SPICE simulation을 한 결과 그림5와 같

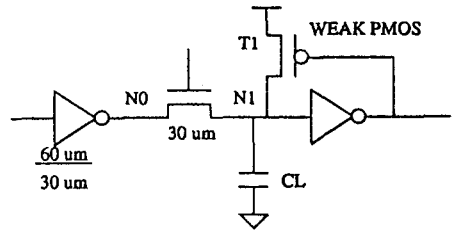

그림 5. Barrel shifter의 transistor 크기 결정
Fig. 5. Determinaion of the transistor size of barrel shifter.

이 트랜지스터의 크기가 결정되었다. 절점 N_0가 V_{cc}일 경우 N_1의 전압은 $V_{cc}-V_T$가 되나 (V_T=NMOS threshold) 이 경우 T_1이 켜져서 N_1의 전압이 V_{cc}까지 올라간다. T_1의 크기가 너무 크면 절점 N_1을 pull down할 경우 지연 시간이 너무 길어질 수가 있으므로 알맞아야 한다. 그림5에 보이는 바와 같이 구동 inverter의 크기가 60μm/30μm(PMOS/NMOS) 이고 NMOS 크기가 30μm인 경우 SPICE simulation에 의하여 T_1은 3μm/5μm (폭/길이)로 결정되었다.

Guard, round, sticky bit detector는 barrel shifter에 의해 우로 shift했을 때 단정도인 경우 소숫점 이하 23bit만 (배정도는 52bit) 가산기로 보내고 나머지 bit은 guard, round, sticky bit만 보존하여 다음 cycle에서 반올림 또는 반내림할 때 사용한다. 단정도일 경우 guard bit은 소숫점 이하 24번째 bit이며 round bit은 25번째 bit이다. Sticky bit은 26번째 이하의 모든 bit 중에 하나라도 1이 있으면 1이고 모두 0이면 0이다. Guard와 round bit을 sticky bit에 포함시키지 않고 따로 알아야 하는 이유는 IEEE 반올림 mode 중 round to nearest의 경우를 위해서이다. 이 guard, round, sticky bit 회로의 이 bit들이 다음번 cycle에서 반올림 또는 반내림할 때 사용되므로 충분한 시간적 여유가 있어서 느린 회로로도 충분하다.

3. 54 bit Adder

이 가산기는 1의 보수 산술을 사용해서 부호-절대값 값을 빠르게 최소한의 회로로 계산하도록 되어 있다. 덧셈인 경우는 B입력이 반전되지 않고 그대로 A와 더해진다. 이것은 2의 보수 경우와 다를 바가 없다. 그러나 뺄셈인 경우 B입력이 XOR gate에 의해서 반전된 다음에 A와 더해진다. 만일 자리올림 출력이 0이면 더해진 값이 원하는 해답이고 자리올림 출력이 1이면 더해진 값에 1을 다시 더한 값이 올바른 해답이 된다. (그림6과 표1) 그림6에서 sum

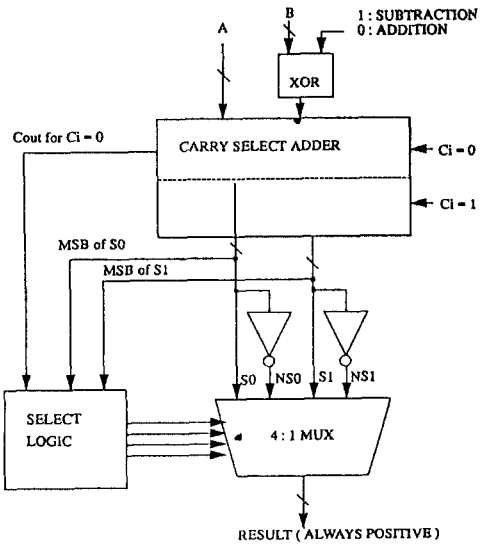

그림 6. 1's complement adder/subtractor
Fig. 6. 1's complement adder/subtractor.

표 1. 그림6의 select logic의 truth table
Table 1. Truth table of the select logic in fig. 6.

Cout	MSB of S0	MSB of S1	MUX SELECTS
0	0	–	S0
0	1	–	NS0
1	–	0	S1
1	–	1	NS0 1

의 MSB가 1인 경우는 /sum을 선택한다. 이렇게 하여 결과는 항상 부호-절대값을 얻을 수 있다. 54bit 덧셈이 9ns 이내에 이루어 져야만 60MHz의 clock속도를 얻을 수 있다. 그러므로 그림7의 자리올림 선택 가산기(carry select adder)[7] 방식을 사용하여 회로를 최소화하고 가장 빠르게 계산할 수 있는 가산기를 설계하였다. 자리올림 선택 가산기내에 사용될 8bit 가산기는 carry chain 가산기(그림8)를 사용했다. Carry look ahead 가산기도 사용이 가능하나 carry chain 가산기의 장점은 회로가 간단하고 A, B 입력의 NAND, NOR, XOR 기능을 동시에 얻을 수 있고 자리올림 입력=1, 자리올림 입력=0의 두개의 가산기가 쉽게 얻어져서 자리올림 선택 가산기로 사용하기가 최적이고 또 다음 cycle에서 반올림 또는 반내림할 때 +0또는 +1을 하여야 하는데 이 기능도 이미 포함되어 있다.

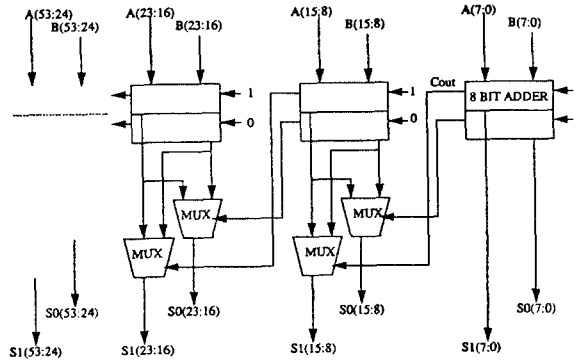

그림 7. Carry select adder
Fig. 7. Carry select adder.

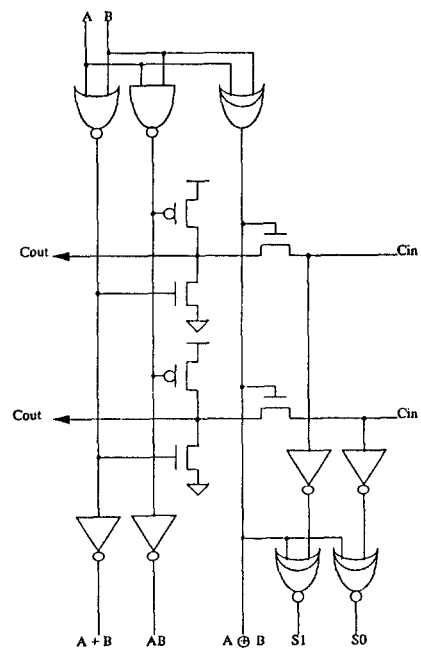

그림 8. 1 bit carry chain adder
Fig. 8. 1 bit carry chain adder.

4. Leading One Detect and Rounding

첫번 cycle에서 54bit 가산기를 거친 값은 IEEE 표준 값이 아닐 수가 있으므로 두번째 cycle에서 barrel shifter를 사용하여 좌로 shift하여 MSB에 1이 오도록 하여야 한다. 동시에 지수의 값은 좌로 shift된 값만큼 빼 주어야 한다. Leading one detector의 역할은 가장 MSB쪽에 있는 1의 위치를 알아내어 barrel shifter로 하여금 그만큼 좌로 shift하도록 하며 지수 unit에게 그 값을 빼 주도록 하여야 한다.

Leading one detector는 critical path에 있으므로 지연시간이 최소가 되도록 설계하는 것이 가장 중요하다. 이렇게 shift된 값은 다시 원하는 반올림 mode에 따라서 +0 또는 +1을 하여야 한다. 자리올림 선택 가산기는 마침 자리올림 입력 =0과 자리올림 입력 =1의 가산기가 이미 포함되어 있으므로 이러한 기능은 쉽게 얻어질 수 있다. 요즘의 VLSI 설계는 같은 면적의 실리콘에 될 수 있으면 더 많은 기능을 넣어야 하므로 회로를 극소화하여 적은 숫자의 트랜지스터로 많은 기능을 수행할 수 있는 것이 대단히 중요하다. 이 ALU에 사용된 54bit 가산기는 덧셈과 뺄셈 그리고 반올림과 반내림의 모든 기능을 한개의 가산기로써 함으로 두개의 가산기의 필요성을 하나로 줄여 최대한의 속도를 얻도록 설계 되었다.

Ⅳ. 회로설계 및 고찰

회로 설계는 daisy workstation, logic simulator는 daisy DLS, circuit simulator는 SPICE를 사용하였다. 사용된 1.0μm의 CMOS technology의 최소 크기 inverter는 PMOS=6μm, NMOS=3μm이었으며 같은 크기의 inverter를 구동하는 지연 시간은 250ps였다. 표2는 사용된 CMOS technology를 요약한 것이다. 60MHz의 clock 주파수를 얻기 위해서는 모든 회로가 최소한의 지연 시간을 갖도록 설계되었다. 그림 9는 첫 cycle과 두번째 cycle의 critical path를 보여준다. 한 cycle이 16ns이므로 적어도 15ns 정도에서 동작하도록 설계되었다. SPICE simulation으로 85℃, worst case process parameter에서도 동작되도록 하였다. 이와 같이 높은 주파수에서도 동작되도록 하려면 회로 설계의 모든 부분에서 지연 시간을 최소화하고 트랜지스터 크기를 알맞게 하는 것이 중요하다. 트랜지스터 크기가 크면 다음단을 빨리 구동할 수 있으나 전단에서의 부하용량이 커져서 느려지므로 전체적으로 가장 빠르게 할 수 있는 각 단계간의 적정한 비율이 있다. 그림10의 inverter chain의 경우 첫번 단계의 gate 용량이 C_G이고 부하용량이 C_L인 경우 $f=C_L/C_G$로 정의한다. Inverter가 같은 크기의 inverter를 구동하는 지연 시간을 t라고 하고 각 단계간의 크기의 비율을 n이라고 하면 전체 N단계가 있을 때 C_G로부터 C_L까지의 전체 지연시간은 $T=Nnt$가 된다. 그리고 $f=n^N$의 관계가 성립된다. 그러므로

$$\ln f=N\cdot\ln n \text{ 이 되고}$$

$$T=\ln f\cdot\frac{n}{\ln n}\cdot t=k\cdot\frac{n}{e\cdot\ln n},\ (k=constant,\ e=natural\ log)$$

표 2. 사용된 CMOS technology
Table 2. Used CMOS technology.

WELL STRUCTURE	: TWIN WELL CMOS
OXIDE THICKNESS	: 200AUNGSTROMS
PMOS THRESHOLD	: −0.7V
NMOS THRESHOLD	: 0.7V
GATE LENGTH	: 1.0μm
METAL LAYER	: DOUBLE

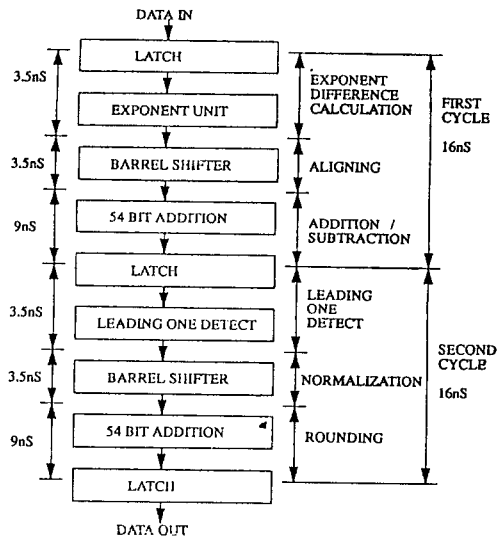

그림 9. Critical path
Fig. 9. Critical path.

표3에 보인 바와 같이 T의 최소값은 n=e=2.718일 때가 된다. 실제로는 2내지 5의 값을 사용하였다. 대부분의 회로에서 가장 빠른 속도를 위해서는 이 하나의 공식에 의하여 그 부하용량을 대등한 크기의 inverter로 환산하여 그 트랜지스터 크기를 결정할 수 있다.

금속 선(metal line)의 부유용량을 계산할 때는 friging 용량[8][9]을 염두에 두어야 한다. 요즘의 CMOS는 1μm 정도의 넓이와 거리를 사용함으로써 용량이 C=k·A/d 공식에 의한 값보다도 100% 정도나 많아질 수 있다. (A:면적, d:거리, k;유전율) 이러한 fringing 용량은 금속선의 넓이, 거리, 두께와 유전체 등에 관계되고 또 금속 식각 과정에서 사진공정(photo process)과 과도 식각(over etch)등에 따라서도 달라지므로 이론적으로 계산하기는 대단히 힘들고 test chip을 설계 측정하여 각 공정마다의 최대치의

표 3. 그림10의 가장 알맞는 inverter 크기 결정 (e=2.718)

Table 3. Determination of the best inverter size in fig. 10 (e=2.718)

n	$\frac{n}{e \ln n}$
1.0	infinite
1.5	1.36
2	1.06
e	1.0
3	1.0
4	1.06
5	1.14
10	1.6
50	4.70
100	7.99

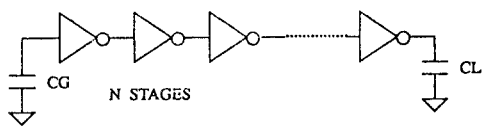

그림10. 가장 알맞는 inverter 크기 결정
Fig. 10. Determination of the best inverter size.

metal 용량을 사용하였다.

그림11은 3입력 NAND gate의 경우 3개의 NMOS 트랜지스터 중 가장 빠른 신호를 ground에서 가장 가까운 T_1에 넣고 가장 느린 신호를 T_3에 넣는 것을 보인다. 이렇게 함으로써 C_1과 C_2가 순차적으로 방전할 수 있도록 하였다. 이 경우 A와 C를 거꾸로 넣는것 보다 약50ps가 빨라진다. 대부분의 VLSI 설계에서는 이와 같은 차이를 무시하나 높은 주파수에서 동작되는 회로는 모든 부분에서 지연 시간을 최소화해야만 하므로 이와 같은 차이도 고려되었다.

이 ALU의 모든 layout은 자동 loyout CAD tool을 사용하지 않고 전부 일일이 손으로 layout을 하였다. 그 이유는 ALU는 부동 소숫점 unit에서 가장 속도가 요구되는 핵심 부분이므로 manual loyout에 의해서 모든 배선(metal line)의 길이를 최소화하여 최대의 속도를 얻기 위해서이다.

V. 결 론

본 설계 연구 논문에서는 IEEE 부동 소숫점 숫자

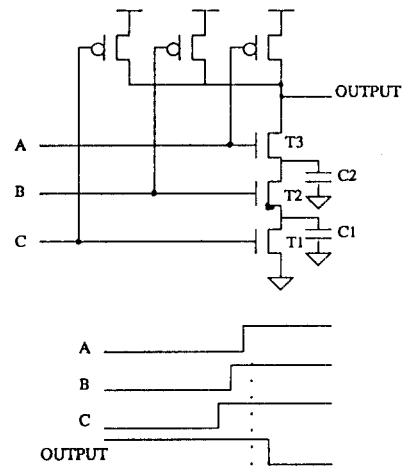

그림11. 3 input NAND gate
Fig. 11. 3 input NAND gate.

계산을 60MHz의 clock 주파수에서 2clock 이내에 할 수 있는 ALU를 제안하였다. 1.0μm CMOS를 사용하여 전체 chip의 크기는 200mil×200mil이었다. 종래의 2의 보수 방식의 가산기가 아닌 지연시간이 9 ns인 1의 보수 방식의 가산기를 사용하였으며 같은 가산기를 첫번째 cycle과 두번째 cycle에 반복하여 사용함으로 두개의 가산기의 필요성을 한개로 줄여서 회로를 극소화 함으로써 60MHz의 clock 주파수에 동작이 가능하였다. 이것은 현재 사용되고 있는 33MHz 정도의 (1990년 말 현재) ALU 보다 약 2-5배 향상된 것이며 수년내로 100MHz까지의 ALU가 가능할 전망이다. 부동 소숫점 unit 전체를 한 논문에서 자세히 논하는 것은 무리이므로 승산기, 제산기, pipeline, microcode등에 대해서도 앞으로 논하고자 한다.

參 考 文 獻

[1] Gerry Kane, *Mips R2000 Risc Architecture*, Prentice-Hall, 1987.

[2] Intel Corp., 80387 Programmer's Manual, 1987.

[3] Motrola Inc., *MC68881 Floating Point Corprocessor Manual*, 1985.

[4] M. Bonomi, P. Chandra, P. Wilson, C. Byington, "Floating point processing," *Byte* pp. 195-204, March, 1988.

[5] IEEE Standard Committee, "IEEE Standard for binary floating-point arithmetic," *ANSI/*

IEEE Std 754-1985.

[6] Timothy Hu, "Circuit design technology for a floating point processor," University of California, Berkeley, Department of Electrical Engineering and Computer Science 1987.

[7] A. Rothermel, B.J. Hosticka, G. Troster, J. Arndt, "Realization of transmission gate conditional sum adder with low latency time," *IEEE Journal of Solid-State Circuits,* vol. 24, no. 3, pp 558-561, Jume, 1989.

[8] P.E. Cottreel, E.MButurla, D.R. Thomas, "Multi-dimensional simulation of VLSI wiring capacitance," *IEDM, pp. 548-551, 1982.*

[9] R.L.M. Dang, N. Shigyo, "Coupling capacitance for two dimensional wires," *IEEE Electron Device Letters,* vol. EDL-2, no. 8, pp. 196-197, August, 1982.

著 者 紹 介

李 溶 錫(正會員)

1973年 연세대학교 전기공학과 졸업. 1981年 University of Michigan, Ann Arbor, 반도체 분야 박사학위 수여. 그후 Device Engineering, Gate Array, Memory, ISDN Telecommunication Chip, Microcontroller, RISC Microprocessor, Floating Point unit, cache Controller 등을 설계. 현재 Intel Corporation에서 Microprocessor System과 architecture를 하고 있음.