정보학연구 제8권 제3호 2005년 9월(pp.1~10)
The Journal of Information Technology, Vol.8, No.3

그래픽 디스프레이에 적합한 Cosine, Sine함수 발생기 설계에 관한 연구

김용성*

목 차

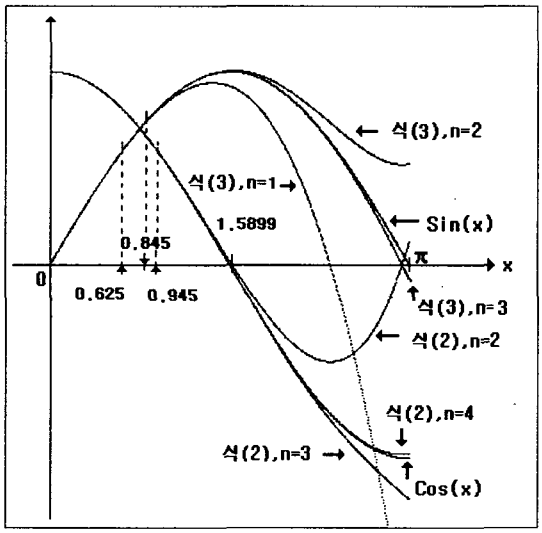
Ⅰ. 서론

Cosine 및 Sine함수는 일반적인 수학적 결과뿐 만아니라, 이 공학의 연구 분야의 시뮬레이션 및 컴퓨터 그래픽의 도형발생, 변형, 애니메이션 등의 다양한 연구 분야에 널리 사용되고 있으며[1], [2], [3], 이 에 대한 Cosine 및 Sine함수 발생기의 고속화에 대한 연구가 계속되고 있다.

일반적으로 Cosine 및 Sine함수 발생은 복잡한 연산 과정을 통하여 얻어지므로, 연산 량이 증가하며 이에 따라 전체 시스템의 연산속도를 저하 시키거나 하드웨어의 크기를 증가 시키는 주요 요인이 되며, 반복적인 연산을 사용하는 코딕(Cordic) 연산기로 설계를 하는 경우가 많다.[4], [5] 그러나 코딕 연산기는 파이프라인화로 설계한 경우에도 기본적인 잠재시간(Latency Time)으로 인하여 전체 연산 속도가 저하되며, 특히 그래픽 등에서 일정한 각도의 회전이나, 투영법에서 요구된 임의의 각도로 조정하는 경우 매번 많은 잠재시간이 소요되는 단점을 갖는다. 그러므로 본 논문에서는 제한 오차 범위에서 컴퓨터 그래픽 디스플레이에 적합한 Cosine 및 Sine함수 발생기를 설계하기 위하여 입력에 따라 $\pi/4$ 구간별 영역으로 분류하는 분류기를 설계하고, Tayler급수의 저차식을 사용한[6], [7] $0 \sim \pi/4$에서 동작하는 기본 Cosine 및 Sine함수 발생기를 설계하며, 이를 구간별 함수 발생이 가능하도록 설계하고자 한다. 또한 코브분할기법을 사용하여 제수 3에 대한 전용 제산기를 제안하여 속도를 향상시킴으로써 그래픽 디스플레이에 적합한 Cosine 및 Sine함수 연산기를 제안하고자 한다.

* 여주대학 컴퓨터 계열 부교수

II. 그래픽 도형 디스플레이와 Cosine 및 Sine함수 근사화



(그림 1) Cosine함수, Sine함수와 근사화식의 비교

Tayler급수는 $e^x, \sqrt{x}$ 등과 같은 함수를 급수의 형태로 근사 화하는데 사용한다. 나머지를 포함한 Tayler급수의 일반식은 식(1)과 같이 되며, Cosine 및 Sine함수의 근사화식은 식(2),식(3)과 같이 표현된다.

x의 라디안 값이 0 ~π에서, n=2, 3, 4인 경우 x에 대하여 식(2)는 각각 4차식, 6차식, 8차식으로 표현되고, n=1인 경우 식(3)은 x에 대하여 3차식으로 표현된다. Cosine 및 Sine함수와 근사화식에 대한 그래프를 그림 1에 표시하였다. Cosine 함수는 4차식으로 근사화한 경우 x=0~0.945에서 10^{-4}이하의 오차를 갖으며, Sine 함수를 3차식으로 근사화한 경우 x=0~0.655에서 10^{-4}이하의 오차를 갖는다. x=0~π의 모든 값에 대해서 10^{-4}이하의 오차를 갖으려면, 식(2)는 n=4이상이 되어야 하며, 식(3)의 경우는 n=3, x의 7차식 이상이어야 한다. 그러나 x^5이상의 차수가 포함된 연산을 수행하는 연산기를 설계하는 경우, 연산의 복잡도가 증가 하고, 속도의 지연 및 연산기의 크기가 커지는 단점을 갖는다. 그래픽 도형을 디스플레이하는 경우, 해상도가 1024x768~1280x1024 이 많이 사용되는데, 원 도형 발생점 시 Cosine, Sine함수와 반지름과의 승산결과는 1개의 좌표 점에 표시되고, 연산 결과는 정수 값으로 적용되므로, 원 함수와 근사화식과의 오차는 10^{-3}~ 10^{-4}이고, rms(Root Mean Square Error)가 1보다 작으면 된다. 그러므로 본 논문에서는 x가 0 ~ $\pi/2$에서 오차 범위를 만족하는 근사화식을 사용한다. 즉, Cosine함수의 근사화 경우 n=2인 4차식을 사용하면 x=0~1.160에서 오차조건을 만족하며, Sine함수의 경우는 n=1인 3차식을 사용하면, x=0~0.845에서 오차조건을 만족하므로, 허용오차 범위에서 함수 발생

$$f(x)=f(a)+f'(a)(x-a)+\frac{1}{2!}f'(a)(x-a)^2+\cdots+\frac{1}{n!}f^{(n)}(a)(x-a)^2+R_{n+1} \quad (1)$$

$$(n\to\infty,\ R_{n+1}\to 0)$$

$$\cos(x)=1-\frac{1}{2!}x^2+\frac{1}{4!}x^4-\frac{1}{6!}x^6-+\cdots+(-1)^n\frac{x^{2n}}{(2n)!}+\cdots \quad (\infty<x<\infty) \quad (2)$$

$$\sin(x)=x-\frac{1}{3!}x^3+\frac{1}{5!}x^5-+\cdots+(-1)^n\frac{x^{2n+1}}{(2n+1)!}+\cdots \quad (\infty<x<\infty) \quad (3)$$

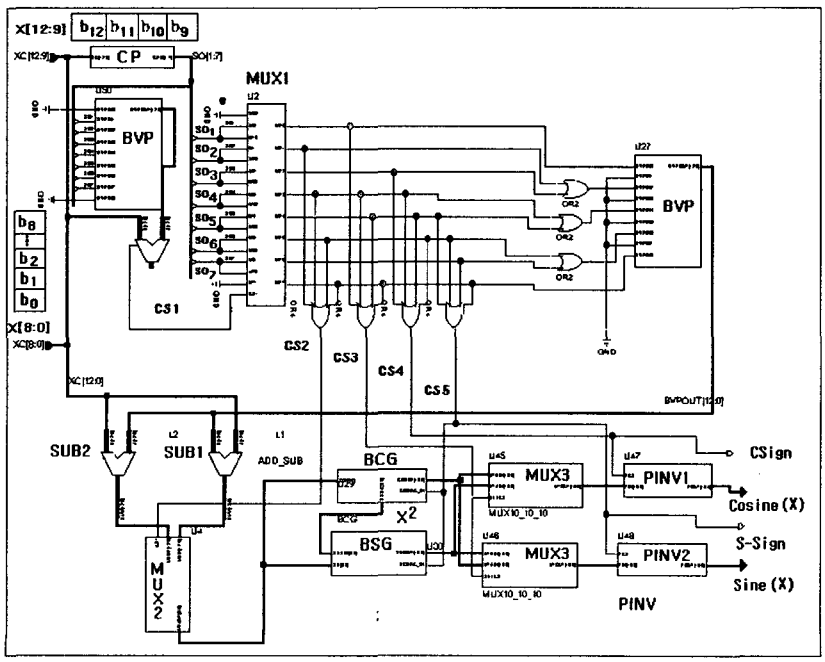
기를 기본으로 하여 작은 크기의 연산기로 Cosine 및 Sine함수를 발생기를 설계하고자 한다.

Ⅲ. 그래픽 디스플레이에 적합한 Cosine, Sine 함수발생기 설계

3.1. Tayler 급수의 저 차식을 사용한 Cosine, Sine 함수발생기 설계

설계하려는 Cosine, Sine함수 발생기는 0~ $\pi/2$에서 동작하는 Cosine, Sine함수 발생기를 기본으로 하여 설계하며, 이를 BCG(Basic Cosine Generator) BSG(Basic Sign Generator). 기본 함수 발생기로 한다. Ⅱ절에 의해 Cosine함수의 경우는 n=2인 4차식을 사용하고, Sine함수의 경우는 n=1인 3차식을 사용하며, 두 개의 오차조건을 만족하는 x의 범위는 0~0.845이므로, 0~ $\pi/4$ (=0.785)를 기준으로 한다. 0.785~ $\pi/2$경우는 Cos($\pi/2$-x)=Sin(x), Sin($\pi/2$-x)=Cos(x)를 사용한다. 본 논문에서는 구간별 Cosine, Sine함수를 표1.(a),(c)에 따라 식(4)와 같이 표현하며, 그림 2와 같이 0~ $\pi/4$를 기준으로 하여 구간 분할하고 기본 함수 발생기를 설계하여 Cos(x), Sin(x) 함수 발생기를 설계한다.

그림 2에서 입력은 x를 2^n배한 정수 값 X를 사용하며, n=10인 경우 0~ 2π을 $\pi/4$씩 8개의 구간으로 분할하고, 입력은 0~ $1922h$의 범위를 갖는다. X입력에 따라 표 1과 같이 0~ 2π에 따라 구역 구분 및 기준 값을 정하며, 다음의 단계



(그림 2) 저 차수 식을 사용한 Cosine, Sine 함수발생기

$$Cos(x) = \cos\{(-1)^{n+1}\pi/2 \lceil n/2 \rceil + (-1)^{n}x\}p \cdot s1 +$$
$$\sin\{(-1)^{n+1}\pi/2 \lceil n/2 \rceil + (-1)^{n}x\}\bar{p} \cdot s2$$
$$Sin(x) = \sin\{(-1)^{n+1}\pi/2 \lceil n/2 \rceil + (-1)^{n}x\}p \cdot s1 +$$
$$\cos\{(-1)^{n+1}\pi/2 \lceil n/2 \rceil + (-1)^{n}x\}\bar{p} \cdot s2$$

(p=1 if n=0,3,4,7 , s1=1, s1=-1 if n=2,3,4,5 , s2=1, s2=-1 if n=4,5,6,7, n:구역 값, 표 1 참조) (4)

별 연산을 수행한다.

단계 1. 표 1의 8개 구간의 기준에 따라 분할의 구분을 위한 1차 분류를 수행한다.

단계 2. 구역 1차 비교 결과에 따라 구역 기본값(BVP)의 비교를 하여 구역을 결정한다.

단계 3. 구역에 따라 구역 기준 값과 현재 입력 X의 차를 구한다.

단계 4. BCG, BSG를 사용하여 구역별 Cosine, Sine함수 값 및 부호를 결정한다.

구역 별 분류는 구역 기준 값으로 입력 X를 하나씩 비교하면 되지만, 8번의 비교연산이 소요된다. 그러므로 본 논문에서는 그림 2와 같이 연

〈표 1〉 입력 X에 대한 구역분할 및 기준 값

(a)

x의 입력범위	구역	1차분류기준(CP)	분류 기준값 (BVP기본값)
0~π/4	0	-	0
π/4~π/2	1	$\overline{b_{12}}\ \overline{b_{11}}\ \overline{b_{10}}$	324h
π/2~3π/4	2	$\overline{b_{12}}\ \overline{b_{11}}\ b_{10}$	649h
3π/4~π	3	$\overline{b_{12}}\ b_{11}\ \overline{b_{10}}$	96Eh
π ~5π/4	4	$\overline{b_{12}}\ b_{11}\ b_{10}\ \overline{b_9}$	C91h
5π/4~3π/2	5	$\overline{b_{12}}\ b_{11}\ b_{10}\ b_9$	FB5h
3π/2~7π/4	6	$b_{12}\ \overline{b_{11}}\ \overline{b_{10}}$	12DAh
7π/4~2π	7	$b_{12}\ \overline{b_{11}}\ b_{10}$	15FFh
2π	8	-	1922h

CP: comperator for partition, BVP: Basis Value of Partition

(b)

구역1차비교	2차비교 기준값 비교결과	
	-	0, +
0	-	-
1	0	1
2	1	2
3	2	3
4	3	4
5	4	5
6	5	6
7	6	7

(c)

결정 구역	BSG, BCG 입력 값	Cosine(x) 기본함수		Sine(x) 기본함수	
		연산기	결과부호	연산기	결과부호
0	X-BVP	BCG	+	BSG	+
1	BVP-X	BSG	+	BCG	+
2	X-BVP	BSG	-	BCG	+
3	BVP-X	BCG	-	BSG	+
4	X-BVP	BCG	-	BSG	-
5	BVP-X	BSG	-	BCG	-
6	X-BVP	BSG	+	BCG	-
7	BVP-X	BCG	+	BSG	-

산의 횟수를 감소시키기 위하여 각 기준 값에 대하여 표 1의 (a)와 같이 상위 bit를 사용하여 1차 분류를 CP에서 수행하며, 구간 값 SOi(i=1....7)와 입력 X의 대소 비교치를 MUX1의 제어입력인 CS1 신호로 사용하여 결과가 양수인 경우는 1차 분류의 구간으로 하고, 음수인 경우는 1차 분류 구역보다 하나적은 구간으로 결정함으로써 표 2. (b)의 2차 구간분류를 수행한다. 기준 값과의 차는 표 1. (c)와 같이 연산하며, 구역 1, 3, 5, 7의 "BVP-X" 감산은 SUB1에서 수행하고, 다른 구역의 감산"X-BVP"는 SUB2에서 SUB1과 동시에

수행하며, 제어신호 CS2에 의해 MUX2에서 선택된다. 또한 구역 0, 3 ,4 ,7인 경우에 BCG와 BSG는 각각 Cosine함수 및 Sine함수로 출력되며, 다른 구역에서는 제어신호 CS3에 따라 MUX3에서 교환되어 출력된다. 부호의 경우도 표 1.의 (C)의 조건에 의한 제어신호 CS4, CS5에 따라 출력 부호를 검출하여 CSign(Cosine부호 비트)과 SSign(Sine부호 비트)로 출력한다.

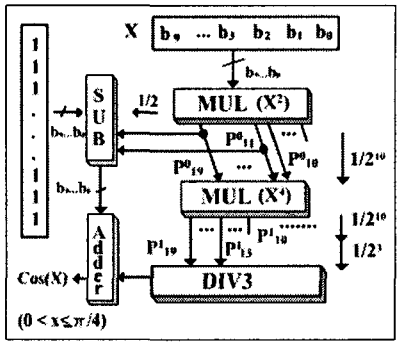
3-2. BCG, BSG 기본 함수 발생기 설계

그림 2에서 함수 발생기의 기본이 되는 BCG와 BSG는 Ⅱ절의 식(3)의 4차식과 식(4)의 3차식을 사용하므로 식 (5), (6)과 같이 표현되며, BCG와 BSG의 기본 구조는 그림 3(a),(b)와 같이 설계할 수 있다.

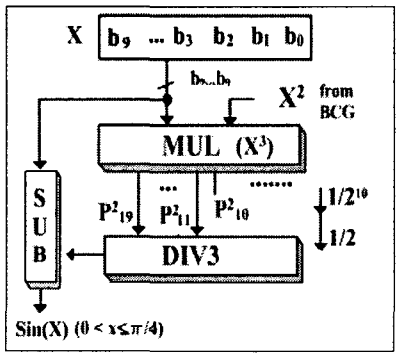
$$\cos(x) = 1 - \frac{1}{2!}x^2 + \frac{1}{4!}x^4 \quad (0 < x \le \pi/4) \tag{5}$$

$$\sin(x) = x - \frac{1}{3!}x^3 \quad (0 < x \le \pi/4) \tag{6}$$

그림 3 (a)의 BCG에서 두 승산기의 승수 및 피승수는 동일한 값으로 한다. 입력 X($b_0 \sim b_9$)의 첫 번째 승산기 결과는 X^2($P^0_{00} \sim P^0_{19}$)인데, X는 x를 2^{10}배하여 사용하므로 결과는 2^{20}배로 표현되므로 하위 10비트를 제외하고 $P^0_{10} \sim P^0_{19}$까지 사용하고 두 번째 승산기의 입력으로 사용한다. 식(5)에서 X^2결과의 1/2을 수행하여야 하므로 감산기(SUB)의 감수는 $P^0_{11} \sim P^0_{19}$으로 한다. 두 번째 승산기의 결과도 첫 번째와



(a) BCG



(b) BSG

(그림 3) BCG, BSG 기본 함수 발생기($0 < x \le \pi/4$)

동일하게 하위 10비트는 제거한다. 식(5)에서 $\frac{1}{4!}x^4$은 $\frac{1}{2^3 \cdot 3}x^4$이므로 제수 3을 갖는 전용 제산기(DIV3)의 입력은 $P^1_{13} \sim P^1_{19}$로 하고, 제산 및 감산 결과는 가산기를 통해 출력을 생성한다. BSG는 BCG의 X^2결과($P^0_{10} \sim P^0_{19}$) 및 X의 입력으로 식(6)의 X^3연산을 수행 하며, 결과의 $1/3! = 1/(2 \cdot 3)$을 수행하여야 하므로 $P^2_{11} \sim P^2_{19}$을 DIV3의 입력으로 하고, X와의 차를 구하여 출력하며, 제수 3인 전용 제산기는 다음 절과 같이 설계한다.

3-3. 코드 분할 기법을 사용한 제수 3인 전용 제산기(DIV3) 설계

일반적인 제산에서는 제수의 1이 시작되는 비트에서 0비트까지 비트 수와 동일한 비트수만큼 피제수의 상위부터 비교하며, 감산을 통한 비교 연산 후 양수 이면 몫을 1로 판별하고 음수이면 몫을 0로 판별하며 피제수를 원래의 수로 복귀시킨다. 다음 단계는 피제수의 하위 1비트를 추가 시킨 다음 이전의 연산과정을 반복 수행하며, 피제수가 제수보다 작을 때까지 반복 수행하므로 많은 연산시간이 소요된다. 감산기 대신 비교기를 사용하는 경우는 피제수를 원래의 수로 복원하는 과정은 생략할 수 있으나, 몫이 1인 경우는 감산을 수행하여야 하므로 연산 속도의 향상은 없으며 연산기의 크기만 증대하게 된다. 식(5)와 식(6)의 제산은 피제수가 $2^m \cdot 3$(m : 정수)의 형태를 가지므로 2^m의 제산은 m bit이동으로 가능하며, 그림 의 함수 발생기에 적용 시에 비트 이동은 단순히 비트 선의 연결을 m bit를 제외한 상위 비트만을 연결하면 되므로 실제의 논리소자는 사용되지 않는다. 그러므로 제산기는 고정된 3의 제수로 제산을 수행하면 되므로 본 논문에서는 코드 분할법을 Ⅲ. 2.절에서 표시한 제수 3인 전용 제산기를 설계하여 연산기의 속도를 향상시키고자 한다. $X = b_n 2^n + b_n 2^n + \ldots + b_0 2^0$ 일 때, $X_k = b_k 2^k$ (k : 정수)라 하면, X는 식(7)과 같이 분할하여 표현할 수 있으며, $\{X^n,\ X^{n-1},\ \ldots,\ X_0\}_2$의 코드는 그림 4. (a)와 같이 표현된다.

$$X = X_n + X_{n-1} + \ldots + X_0,$$

$$X_n = b_n 2^n + 0 + \ldots + 0,\ \ldots,$$

$$X_1 = b_1 2^1 + 0,\ X_0 = b_0 2^0 \quad (7)$$

식 (7)을 3으로 제산하는 경우, 식(8)과 같이 X_k에 대한 몫 Q_{X_k}와 나머지로 R_{Xk}로 표현한다.

$$Q_X = \sum_{k=0}^{n} Q_{Xk},\ R_X = \sum_{k=0}^{n} R_{Xk}$$

$$X_k/3 = (b_k 2^k + 0 + \ldots + 0) /(1 \cdot 2^1 + 1 \cdot 2^0) = Q_{X_k}(1 \cdot 2^1 + 1 \cdot 2^0) + R_{Xk}\ (\text{k:정수}) \quad (8)$$

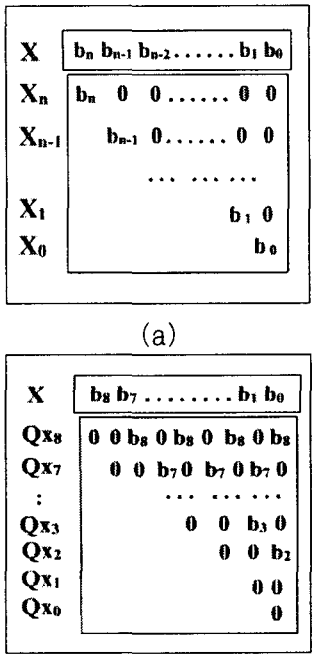
몫 Q_{Xk}는 식(9)와 같이 표현되며 b_k의 2비트 하위부터 2비트마다 b_k로 몫의 비트열을 구성하며, R_{Xk}는 k(k :정수)가 홀수인 경우는 나머지가 1이 되고, 짝수인 경우는 10_2이 된다. Q_{X1}과, Q_{X0}은 0 이므로, X_1과 X_0가 나머지가 된다.

$$Q_{Xk} = 0 + 0 + b_{k-2} 2^{k-2} + 0 + b_{k-4} 2^{k-4} + 0 + \ldots + b_{k-2m} 2^{k-2m} (\text{k,m: 정수},\ k - 2m \geq 0)$$

$$R_{Xk} = 1 \cdot 2^1 \cdot (k \bmod 3) + 2^0 \cdot (k \bmod 2),\ R_{X1} = b_1 \cdot 2^1,\ R_{X0} = b_0$$

$$R_X = Q_{Rx}/3 + R,\ Q = Q_{Xk} + Q_{Rx} \quad (9)$$

피제수를 9비트로 하는 경우 분할 된 몫을 그림 4의 (b)에 표현하고 있는데, 홀수 번 분할 비트의 몫과 짝수 분할비트의 몫은 한 비트 씩 서로 교대로 표현되므로, 연속된 두 개의 분할 비트의 몫은 연산과정 없이 분할 몫의 해당 비트만을 연결하면 된다. 피제수가 9비트인 경우 그림

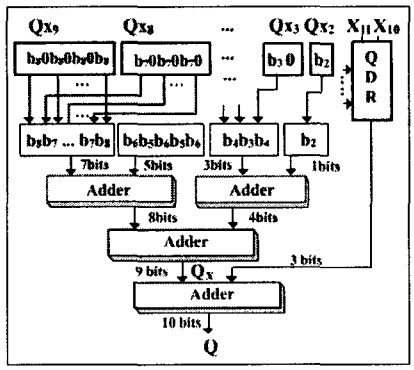


(그림 4) n bit 피제수의 코드분할 및 분할된 몫(제수=3)
(a) 코드분할(n bit) (b) 분할된 몫

5에 분할 코드를 이용한 제수3 용 제산기를 설계하였다. 예를 들어 k=8, 6, 4, 2일 때, $Q_{xk}+Q_{x(k-1)}$의 결과는 $\{b_k, b_{(k-1)}, ...\}$로 형태로 표현되며 그림 5의 (a)와 같이 가산을 수행하여 Q_X를 산출하고 , Q_{Rx}는 그림 5. (b)와 같이 QDR(Quotient for Division of accumulated residue) 연산기에서 수행하며, 발생된 나머지의 합을 3으로 나눈 몫을 연산한다. 식 (9)와 같이 k가 짝수인 경우와 홀수인 경우의 나머지의 "1" 표현 비트가 서로 배타적이므로 $\{R_{Xk}, R_{Xk-1}\}_2$ (k=9,7,5,3,2)로 2비트 씩 배열한 후 가산을 수행한다. 가산의 결과가 S_0, S_1, S_2, S_3일 때, 식 (10)의 논리를 수행하는 변환기를 통하여 몫을 Q_{Rx}을 구하고, Q_X와 가산에 의해 몫 Q를 생성한다.

$$QDR_0 = S_3S_2 + S_3S_1S_0 \quad QDR_1 = \overline{S_3}S_2S_0 + \overline{S_3}S_2S_1 + S_3\overline{S_2S_1} + S_3\overline{S_2S_0}$$

$$QDR_2 = \overline{S_3}S_2\overline{S_1S_0} + S_3\overline{S_2S_1} + \overline{S_3S_2}S_1 + S_3S_2S_1 + S_3S_1\overline{S_0} \quad (10)$$

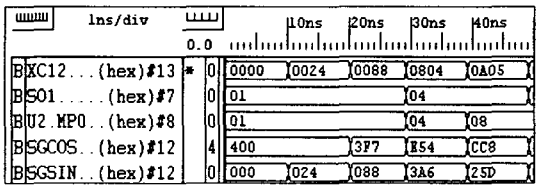


(b)
(그림 5) 코드 분할법을 사용한 제수 3 전용 제산기(n=9)
(a) 제산기 (b) 누적된 나머지의 제산

Ⅳ. 실험 및 고찰

본 논문에서 그래픽 디스플레이에 적합한

Sine, Cosine함수 발생기에 대한 검증은 HLL와 Xilinx FPGA용 논리회로 시뮬레이터를 사용하였다. 그림 2의 1차 구간 비교용 CP는 표 1(a)에 따라 구성하였으며, BVP의 비교용 기본 값은 2의 보수를 사용하고 감산기 SUB1과 SUB2는 가산기로 대치하여 수행할 수 있다. 또한, 그림 3에서 음수의 경우 PINV(Programmable Inverter)를 1의 보수 형태로 표현되는데, 2의 보수로 표현하기 위하여 PINV의 제어입력을 그림 4의 (a)BCG, (b)BSG의 캐리입력에 연결하며, 부호비트를 포함하여 12비트로 출력된다. 그림 6에 제안된 Cosine, Sine함수 발생기의 입출력 신호를 표시하였으며, x=0.1328일 때 X는 2^{10}배로 입력하므로 88_{16}이 XC에 입력되면, 구간 선택신호는 01이 되고, Cosine 값 0.991933의 2^{10}배인 $3F7_{16}$(12비트)이 출력되며, X=A05인 경우는 구역 1차 비교신호 SO는 04_{16}가 되어 SO2=1선택되며, 구역 2차 비교 시는 MP신호가 08_{16}이 되어 SO3=1이 되므로 구간이 3이 되고, 2의 보수로 표현된 $CC8_{16}$(12비트)가 출력된다.



(그림 3) 제안된 Consune, Sine함수 발생기의 출력 신호

AND, OR 등의 기본 논리 게이트 연산시간을 t_{blg}라 하고, 가산기 및 감산기의 연산시간을 t_{aop}라 하는 경우 전체 연산시간은 1차 구간 선택 시간 t_{fsec}, 2차 구간 선택 및 입력 조정시간을 t_{ssec}, BVP는 구역 개별 입력에 따른 OR논리로 구성이 가능하므로e)소요시간은 t_{blg}로 한다. BCG연산시간은 t_{BCG}, BSG연산시간은 t_{BSG}라 하고, 승산기의 시간을 t_{mlt}, 제수 3 전용 제산기의 연산시간을 t_{3divn}이라하는 경우, Cosine 함수 전체 연산시간 $t_{\cos}$ 및 t_{Sin} 는 식(11)과 같이 표현된다.

$$t_{\cos} = t_{fsec} + t_{ssec} + t_{BCG} + 2\,t_{blg} = 8\,t_{blg} + 6\,t_{aop} + 2\,t_{mlt},\quad t_{\cos} = t_{Sin}$$

$$t_{fsec} = 2\,t_{blg},\quad t_{ssec} = 5\,t_{blg} + 2\,t_{aop},\quad t_{BCG} = t_{BSG} = 2\,t_{mlt} + t_{3divn} + t_{aop},\quad t_{3divn} = 3\,t_{aop} + t_{blg}$$

BCG연산에서 감산은 X^4연산과 동시에 이루어지며, BSG연산은 연산기 크기는 BCG보다 작지만 BCG의 X^2을 전달 받으므로 BCG와 동일한 연산시간이 소요된다. 참고문헌 [5]의 경우 연산시간은 $(29+n)(t_{aop} + t_{Reg})$인데, 파이프라인화가 되어있어서 $29(t_{aop} + t_{Reg})$이후에는 1 clock 마다 출력이 되지만 기본지연시간이 $29(t_{aop} + t_{Reg})$이 되므로, 결과가 연속적으로 요구되지 않는 경우에는 지연시간이 많이 소요되는 단점을 갖는다. 제안된 함수 발생기의 경우, BCG연산 후 MUX 및 PINV의 소요시간을 $3\,t_{blg}$라 하고, 승산기의 속도가 $5\,t_{aop}$라 하면, $t_{\cos} = 16\,t_{aop} + 11\,t_{blg}$이므로 연산시간이 향상되었음을 알 수 있다. 또한, 본 논문에서 설계된 구조에 레지스터를 부가하여 파이프라인 설계로 전화하는 경우에도 기본지연시간이 줄어드는 장점은 있으나, 정밀한 값을 요구하는 연산에는 참고 문헌[5]가 유리하므로, 제안된 함수 발생기는 그래픽 디스프레이에 적합함을 알 수 있다.

Ⅵ. 결론

컴퓨터 그래픽에서 변환, 전환 및 각 분야의 시뮬레이션 등의 수행 시 Cosine 및 Sine함수가 널리 사용되고 있으며, 일반적으로 삼각함수 발생기는 Cordic 알고리즘을 사용하며 반복적인 연산을 통하여 결과 값을 생성한다. 그러나 이러한 수행한 결과를 그래픽 도면에 디스플레이하는 경우 양자화된 값으로 표현하게 된다. 그러므로 본 논문에서는 그래픽 디스플레이에 적합하도록 Cosine 및 Sine함수 발생기의 처리 속도를 향상시키기 위하여 Tayler급수의 저차식의 사용하고, $0 \sim 2/\pi$사이의 Cosine 및 Sine 함수 발생기를 기본으로 하여 $0 \sim 2\pi$에서 구간별로 적용될 수 있도록 설계하고, 제수 3을 기본으로 한 전용 제산기를 설계하여 기본 함수 발생기 BCG 및 BSG를 설계 하였으며, rms오차<1 범위 내에서 처리속도를 향상시켰다. 설계된 함수 발생기는 입력 값이 연속이 아닌 경우에 참고분헌[5]보다 처리 속도가 향상되었고, 설계된 함수발생기를 파이프라인 처리하는 경우에는 연속적인 입력인 경우에도 처리속도가 향상될 수 있는 장정을 갖는다. 그러나, 승산기가 전체 처리 속도를 저해시키고 있으며, 그래픽 디스플레이보다 허용 오차 범위가 작은 경우에는 적합하지 않은 단점을 갖는다.

앞으로 설계된 함수 발생기의 속도를 향상시키기 위하여 자승 및 3승용 전용 승산기에 대한 설계가 필요하며, 정밀도를 높이도록 하여 다른 분야에도 활용이 가능하도록 하여야 할 것이다.

참고문헌

[1] Foley, Van Dam, Feiner, and Hugles, *Computer graphic principle and practice*, Addison-Wesley, Second Edition, 1990.

[2] William.M.Neuman, Robert F. Sproull, *Principle of Interactive Computer Graphics*, McGraw-Hill Inc, 1986.

[3] 김용성 외 1, "행렬·벡터 연산용 1차원 시스톨릭 어레이 프로세서를 이용한 그래픽 가속기의 설계", 「전자공학회논문지」, 제30권 B편, 제1호, 1993, pp.1-9.

[4] 이민석 외 1, "CORDIC알고리즘을 이용한 DFFS설계", 「대한전자공학회론 추계종합학술대회 논문집」, 제22권 제2호, 1999, pp.985-988.

[5] 이단균 외 1, "Global Positioning System응용을 위한 파이프라인 형 Cordic회로 설계", 「전자공학회논문지」, 제23권, 제11호, 1996, pp.89-101.

[6] Waston Fulks, Advanced Calculus-An Introduction to Analysis, John Wiely & Sons Inc, 1969.

[7] Erwin Kreyszig, Advanced Engineering Mathematics, 탑출판사, 1979.

A Study On the Design of Cosine, Sine Function Generator for the Display of Graphics

Yong-Sung Kim*

Abstract

Cosine and Sine function is widely used for the arithmetic, translation, object drawing, Simulation and etc. of Computer Graphics in Natural Science and Engineering. In general, Cordic Algorithm is effective method since it has relatively small size and simple architecture on trigonometric function generation. However profitably it has those merits, the problem of operation speed is occurred. In graphic display system, the operation result of object drawing is quantized and has the condition that is satisfied with rms error less than 1. So in this paper, the proposed generator is composed of partition operator at each $\pi/4$ and basic Cosine, Sine function generator in the range of $0\sim \pi/4$ using the lower order of Tayler's series in an acceptable error range, that enlarge the range of $0\sim 2\pi$ according to a definition of the trigonometric function for the purpose of having a high speed Cosine, Sine function generation. And, division operator using code partition for divisor three is proposed, the proposed function generator has high speed operation, but it has the problems in the other application parts with accurate results, is need to increase the speed of the multiplication.

Key Words: function generator, consine function, sine function

* Associate Professor, Dept. of Computer System, Yeojoo Institute of Technology