論文91-28B-9-3

한글 문자의 생성을 위한 하드웨어 가속기 개발

(Development of a Hardware Accelerator for Generation of Korean Character)

李泰炯*** 黃奎哲* 李潤泰* 裵鍾洪** 慶宗旻*

(Tae Hyung Lee, Gyu Cheol Hwang, Yun Tae Lee, Jong Hong Bae, and Chong Min Kyung)

要 約

본 논문은 탁상 출판시스템에서 PostScript등의 외곽선 폰트 자료로부터 한글 bitmap 폰트를 고속으로 생성해내기 위한 그래픽 시스템의 개발에 관한 것이다. 그리고 3차 Bezier 곡선의 제어점으로부터 곡선 궤적의 고속 계산을 수행하는 VLSI 칩인 KAFOG의 개발에 대하여 설명 하였다. KAFOG는 1.5μm CMOS gate array로 구현하였으며 사용된 gate의 수는 17,000개이다. KAFOG는 3차 Bezier 곡선을 직선들의 집합으로 근사화 하며, 최대 성능은 40MHz에서 초당 250K개의 곡선을 계산한다.

제작된 그래픽 시스템은 두개의 MC68000 마이크로 프로세서와 KAFOG로 구성된다. 두 프로세서는 master-slave 관계로 동작하며, 두 프로세서간의 통신의 handshaking 방식에 의해 이루어진다. KAFOG는 slave에 의해 제어되며, 외곽선 문자 계산을 위한 보조 프로세서로 사용된다. 그래픽 시스템은 초당 40개의 64×64 외곽선 문자를 발생시킨다.

Abstract

In this paper, we propose a graphic system for high speed generation of bitmap font data from the outline font data such as PostScript, etc. in desk-top publishing system. A VLSI chip called KAFOG was designed for the high-speed calculation of a cubic Bezier curve, which was implemented in 1.5 μm CMOS gate array using 17,000 gates. A cubic Bezier curve is approximated by a set of line segments in KAFOG at the throughput of 250K curves per second with the clock frequency of 40 MHz.

A prototype graphic system was developed using two MC6800 microprocessors and the KAFOG chip. Two microprocessors cooperate in a master and slave mode, and handshaking is used for commuication between two processors. KAFOG chip, being controlled by the slave processor, operates as a coprocessor for the calculation of the outline font. The throughput of the prototype graphic system is 40 64x64 outline fonts per second.

*正會員, **準會員, 韓國科學技術院 電氣 및 電子工學科
(Dept. of Electrical Eng., KAIST)

***正會員, 三星電子(株)家電部門 綜合研究所
(Samsung Elec. Co., ASIC Design Lab.)

接受日字 : 1991年 5月 14日

I. 서 론

사용자의 요구에 따라 다양한 형태의 글자체, 그래픽 기능을 제공하는 전자 출판 시스템이 LBP(laser beam printer)등의 개발에 힘입어 컴퓨터 응용 분야

로 많이 사용되고 있다. 특히 전자출판 시스템에서 다양한 글자체의 제공은 문서 편집기 등에서 매우 중요한 부분을 차지한다. 전자 출판 시스템에서 다양한 글자체의 제공을 위해서, 외곽선 글자체[1] (Outline font)가 글자체 정보를 표현하는 방법으로 널리 사용된다. 외곽선 글자체는 문자용 터미날이나 점행렬 프린터에서 사용되는 점행렬 글자체(bitmap font)와는 달리, 글자체 형태를 직선, 원, 그리고 자유곡선과 같은 그래픽 프리미티브등을 사용하여 묘사하여 고화질의 여러 형태 글자체를 다양하게 발생시킬 수 있는 글자체 정보화 방법이다. 그러나 외곽선 글자체로부터 bitmap 데이타를 계산하기 위해서는, 좌표변환이나 자유곡선 (일반적으로 외곽선 글자체에서는 3차 Bezier 곡선[4]이 사용됨) 계산과 같은 산술연산과 직선, 원 그리고 임의의 폐곡선 filling과 같은 그래픽 기능 수행이 요구된다. 특히 복잡한 다항식으로 나타내어지는 자유곡선의 계산은 많은 계산 시간이 소요된다.

본 논문은 외곽선 글자체로부터 bitmap 데이타 발생을 가속시키기 위한 그래픽 가속기 개발에 관한 연구이다. 특히 자유곡선 (3차 Bezier Curve) 계산을 하드웨어로 구현하여 외곽선 글자체 계산 시간을 단축 시켰다. 3차 Bezier 곡선 계산용 VLSI 칩은 1.5μm CMOS Gate Array로 제작 되었으며 (이하 이 칩의 이름을 KAFOG라 함), 사용된 gate 수는 약 17,000개이다. 외곽선 및 2차원 그래픽 처리를 위한 그래픽 시스템은 2개의 MC68000 마이크로 프로세서와 KAFOG 그리고 TMS34061 그래픽 콘트롤러로 구성된다.

본 논문의 구성은 다음과 같다. 2장에서는 3차 자유곡선 계산 전용 하드웨어인 KAFOG의 구조 및 동작원리, 3장에서는 KAFOG를 이용한 그래픽 가속기의 구조, 4장에서는 그래픽 시스템의 동작 환경, 그리고 5장에서 결과 및 결론으로 구성된다.

Ⅱ. KAFOG의 구조 및 동작 원리

KAFOG는 외곽선 문자에서 사용되는 3차 Bezier 곡선 계산 전용 하드웨어로, KAFOG에서 구현된 Bezier 곡선 계산 알고리듬은 파라메타 증가 방법(Parameter increasing method)이다. 파라메터 증가 방법은 forward differencing 방법[3]이나 subdivision 방법[3]에 비해 파이프라인 구조를 갖는 하드웨어로 쉽게 구현될 뿐만 아니라, 특히 subdivision 방법에 비해 계산 오차가 적은 장점이 있다. 일반적인 매개변수를 이용한 3차 Bezier 곡선식은 식(1)과 같이 표현된다.

$$P(t)=P_0(1-t)^3+3P_1(1-t)^2t+3P_2(1-t)t^2+P_3t^3 \quad (0\leq t\leq 1) \qquad (1)$$

P_0, P_1, P_2과 P_3는 곡선의 곡률 및 크기등을 결정하는 제어점이다. 매개변수 t의 다항식으로 표현된 식(1)을 파라메터 증가 방법을 이용하여 계산하는 과정이 그림1에서 설명된다.

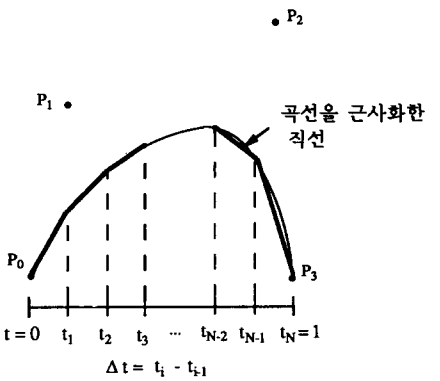


그림 1. 파라메터 증가 방법을 이용한 3차 Bezier 곡선 계산 알고리듬

Fig. 1. Algorithm for cubic bezier curve calculation based on parameter increasing method.

파라메터 증가 방법은 곡선을 임의의 갯수의 직선으로 근사화 시키는 방법이다. Bezier 곡선을 근사화 하는 직선의 갯수는 곡선의 크기에 따라 변화하고, 곡선의 크기는 곡선의 제어점을 포함하는 최소 크기의 사각형으로 정의된다. 그림1은 곡선을 N개의 직선으로 근사한 예를 보여준다. 곡선을 근사화하는 각 직선의 양 끝점은 $\triangle$t의 값으로 증가하는 매개 변수 t의 값에 의해 식(1)로부터 계산된 P(t)의 값이다. $\triangle$t의 값에 의해 0으로부터 1까지 증가하는 t에 의해 계산된 P(t) 대하여 각각 이웃하는 P(t)의 값을 연결한 직선들이 곡선을 근사화한 직선의 집합이 된다.

3차 Bezier 곡선을 파라메터 증가 방법을 이용하여 파이프라인 구조를 갖는 하드웨어를 구현하기 위해 식(1)을 식(2)로 변형시켰다.

$$P(t)=At^3+Bt^2+Ct+D=[(At+B)t+C]t+D$$
$$A=-P_0+3P_1-3P_2+P_3$$
$$B=3P_0-6P_1+3P_2$$
$$C=-3P_0+3P_1$$
$$D=P_0 \qquad (2)$$

식(2)는 Xt+Y 형태식의 반복적인 형태로 표현된 것과 같다. 위와 같은 특성으로 인하여 식(2)는 그림 2와 같은 3단 파이프라인 구조를 갖는 하드웨어로 쉽게 구현될 수 있다.

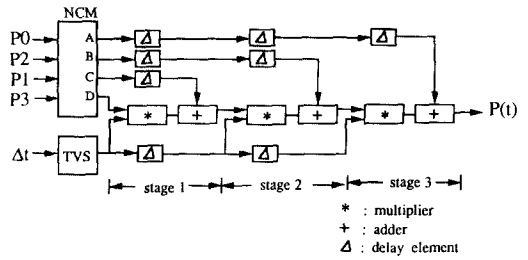


그림 2. 3차 Bezier 곡선 계산을 위한 3단 파이프라인 구조를 갖는 하드웨어 구조

Fig. 2. A hardware architecture consisting of 3 stage pipeline for the calculation of cubic Bezier curves.

3차 Bezier 곡선 하드웨어는 NCM(new coefficient maker), TVS(t value scheduler) 그리고 연산부로 구성된다. NCM 블럭은 3차 Bezier 곡선의 제어점인 P_0, P_1, P_2와 P_3로부터 식(2)의 새로운 계수 A, B, C 그리고 D를 계산하는 하드웨어이다. TVS는 Δt의 값을 입력으로 받아서 파이프라인의 각 계산 순간에 파라미터 t값을 계산하여 연산부에 전송하는 하드웨어이다. 나머지 연산부는 실제 Bezier 곡선을 계산하는 부분으로 bit-serial 구조를 갖는 곱셈기와 덧셈기로 구성되고 자세한 동작원리는 다음 절에서 설명된다.

1. 파이프라인 Bit-Serial 곱셈기의 설계

곡선을 계산하는 하드웨어에서 가장 큰 비중을 차지하는 것은 곱셈기이다. 본 논문에서는 곱셈기에 대한 하드웨어 비용을 가능한 줄이는 반면, 동작 속도를 상당한 수준으로 유지하기 위해 파이프라인 구조를 갖는 bit-serial 곱셈 알고리듬[6]을 사용하였다. Bit-serial 방식은 연산을 bit단위로 수행하기 때문에 각 곱셈기의 연산소자에서 소요되는 시간이 적고, 데이타 룰이 stream 형태로 파이프 라인을 통하여 연속적으로 처리된다. 또한 이 방식은 한 chip내에 여러개의 연산자를 수용할 수 있고 이들을 동시에 동작시킬 수 있기 때문에 연산자와 데이타가 많은 경우에 특히 유용하다. 그리고 bit-serial방식의 하드웨어에서는 각 연산자의 동기방법이 중요한 관건이 된다.

그림3에서 연산부의 입력 데이타 A, B, C, D는 곱셈이나 덧셈에서 소요되는 지연시간만큼 shift된 시간에 다음 단의 입력으로 들어가도록 하여야 한다. A, B, C, D는 M-bit 2's complement 정수이고, t는 TVS 블럭에서 계산된 '0'과 '1' 사이의 값을 갖는 소수이다.

일반적으로 M-bit의 X값과 N-bit의 Y값을 곱할때, 보통 곱셈 결과는 (M+N)-bit이지만, KAFOG 연산부의 두입력은 2's complement 정수와 0과 1사이의 소수이고, 이 소수부분이 N-bit으로 정규화 되어짐에 따라, 0이상의 정수만이 결과로서 유효한 KAFOG의 특성에 의해 하위 N-bit은 반올림 오차정도의 영향을 주기 때문에 무시하여도 된다. 따라서 KAFOG의 연산부는 M×N 곱셈기이지만 M-bit만을 출력한다.

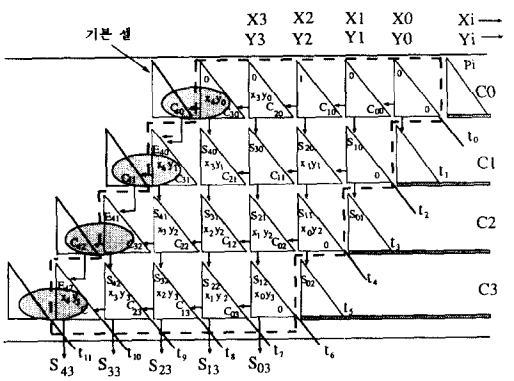


그림 3. 곱셈기의 파이프라인 동기와 기본 셀

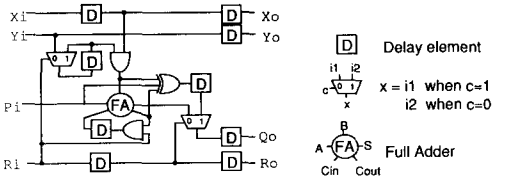
Fig. 3. Pipeline synchronization and basic cell of multiplier.

그림3은 파이프라인 구조를 갖는 bit-serial 곱셈기의 timing 관계를 보이고 있다. 그림에서 삼각형은 full adder를 의미하고 타원으로 표시된 부분은 부호확장을 위한 exclusive-OR를 의미한다. 점선의 내부는 현재 입력되고 있는 X, Y 입력 데이타에 대하여 연산이 진행되고 있는 부분이고, 점선의 오른쪽은 이미 연산이 끝난 결과이다. 그리고 점선의 왼쪽은 다음에 들어오는 데이타에 대한 연산을 나타낸다. 또한 full adder를 나타내는 삼각형의 빗변의 연장선상에 있는 연산들은 같은 시각에 이루어짐을 나타낸다. 즉, 시간이 지남에 따라 연산이 왼쪽으로 진행되는 것을 알 수 있다.

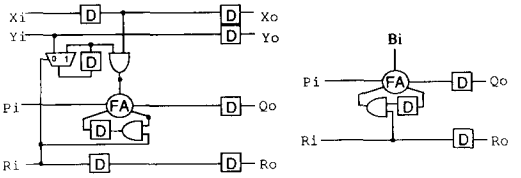
그림3을 보면 전체 cell의 갯수는 M의 값에는 관계없이 N개만 있으면 된다는 것을 알 수 있다. 그리고 주어진 입력에 대한 곱셈의 결과는 2N clock이후에 출력된다는 것을 알 수 있다. 이 bit-serial 곱셈기는 연속적인 입력을 받아들이면서 계속 출력을 내보낼 수 있는 것이 특징인데, 그렇게 하기 위해서는 앞에서 언급했듯이 M-bit 입력에 대해 다음 단으로 M-bit 만의 출력을 넘겨 주어야 하는데 하위 (N−1) -bit의 결과에 대한 효과를 살려야 한다. 따라서 하위 (N−1)bit의 적당한 곳에서 반올림을 시키기 위해 반올림 bit를 더해 주어야 한다. 그림3에서 X는 2's complement 정수이고 Y는 '0'와 '1'사이의 소수라고 하면 X 데이타의 가장 오른쪽에 Y 데이타의 가장 왼쪽에 소숫점이 있다고 할 수 있고, 이런 경우 연산결과는 소숫점이하 첫째 자리까지 구해지게 된다.

일반 곱셈 알고리듬에서 2's complement 정수를 곱하는 경우에는 첫번째 열의 M-bit를 부호확장시켜 (M+1)-bit를 얻고 다음 열의 (M+1)-bit와 덧셈을 행하고 하는 동작을 N번 반복하여 결과를 얻게 된다. 그런데 파이프라인 bit-serial 곱셈기에서는 일련의 두 cell만이 연결되어 있으므로 각 cell에서 각각 한번씩 부호확장을 하여야 한다. 즉, C_0의 부호 확장에 대한 정보를 C_1에 넘겨주고 다시 그것을 C_2의 부호 확장시 고려하여 C_3에 넘겨주는 동작을 반복하여야 한다. 이것을 구현하기 위해서는 그림3에서와 같이 C_0의 출력중 하위 5bit는 각 bit 위치에서 partial sum, carry, partial product를 full addition 하여 얻어지고, 이것들은 다시 C_1의 paritial sum으로 입력되고, 그 중 부호 bit(MSB)로는 C_0의 MSB위치에서 partial sum 입력과 partial product, 그리고 발생되는 carry를 Exclusive OR하여 얻어진다.

여기서 사용되는 cell은 두가지 type으로 그림 4 에서 보듯이 cell-A는 부호확장 회로가 포함되어 있고, cell-B는 마지막 단으로 부호 확장회로가 제거된 형태이다. 즉, (N−1)개는 cell-A를 사용하고 마지막 단1개는 cell-B를 사용하여 전체 N개의 cell로 그림5와 같이 한단의 bit-serial 곱셈기가 구성되는 것이다. 그림5에서 입력 x_i는 M-bit의 2's complement 정수 데이타가 그리고 y_i에는 N-bit의 '0'와 '1' 사이의 값을 갖는 고정 소숫점 데이타가 stream으로 입력되고, r_i는 현재 들어오고 있는 데이타의 LSB를 나타내 주는 제어신호로 현재의 bit가 LSB이면 r_i는 '0'가 되고 그 외에는 '1'을 갖게 된다. 따라서 r_i가 '0'가 될 때 carry를 '0'로 reset 시켜 주는 역할을 하게 된다.



(a) Cell-A : 부호확장 기능을 가진 1bit 곱셈기



(b) Cell-B : 1bit 곱셈기 (c) 1bit 덧셈기

그림 4. Bit-serial 곱셈기에서 사용되는 연산 소자들
Fig. 4. Arithmetic elements used in bit-serial multiplier.

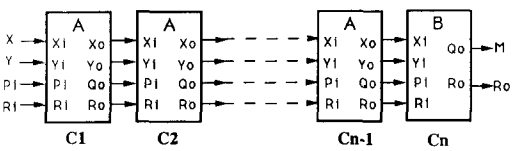


그림 5. 파이프라인 bit-serial 곱셈기의 구조
Fig. 5. The architecture of pipelined bit-serial multiplier.

그리고 r_i는 각 cell에서 적절한 시간에 y_i bit를 flip-flop에 저장하고 부호를 확장하는 역할도 한다. p_i는 이전까지 합한결과 (partial sum)가 들어오고, 가장 앞단의 p_i는 반올림된 곱셈 결과를 내주기 위한 신호로 데이타의 LSB로부터 (N−1)번째 bit를 '1'로 하여 이진 소숫점 첫째자리에서 반올림 시키는 역할을 한다. 그림6은 파이프라인 각 단의 결과가 완료되기 전에 중간결과가 다음단에 영향을 주지 않아야 함을 설명하고 있다. 그림에서 점선으로 된 대각선 부분은 인접한 곱셈과정의 결과들이 서로 다른 것에 영향을 미치지 않도록 분리되어야 함을 나타낸다. 즉, D_0단에서는 내부적으로 계산되는 (N−1)개의 bit를 내부에서만 전파시키고 D_1으로 넘어가지 않도록 하여 D_1에서 계산되고 있는 결과에 영향을 주지 않도록 하여야 한다. 그렇게 하기 위해서는 현재의 bit가 그 데이타의 LSB라는 것을 나타내 주는 제어신호가 있어서 그 신호가 지나는 각 cell을 초기화 시키는 역할을 하도록 하여야 한다. 이렇게 함으

로해서 2N bit의 지연후에 연속적으로 유효한 결과를 얻을 수 있게 되는 것이다.

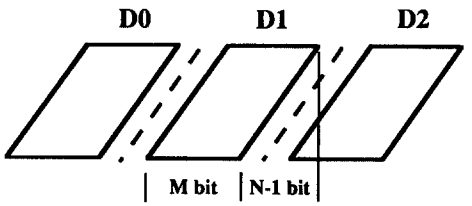


그림 6. 파이프라인 각 단의 계산중간 값이 다음단에 영향을 미치지 않아야 함을 나타낸다.

Fig. 6. Intermediate result of pipeline should not effect the next inputs of pipeline.

2. 연산기 각 단에서의 결과

파이프라인 각 단에서의 출력으로 나오는 결과의 bit 수를 미리 계산하여 보는 것은 실제 하드웨어 구성시 파이프라인 각 출력단에 놓여질 latch의 크기를 결정하는데 상당히 유용하게 쓰여질 수 있어, 본 절에서는 파이프라인 각 단 결과의 bit수를 계산한다.

식(2)에서 제어점에 해당하는 P_0, P_1, P_2, P_3가 각각 K bit로 표현된 데이타라고 하면, 식(2)에서 A는 (K+2) bit, B는 (K+3) bit, C는 (K+2) bit, 그리고, D는 K bits로 표현된다. 그러면 각 단에서의 계산 결과의 유효 bit수를 알아보자.

먼저 At+B를 계산하는 단의 결과 유효 bit 수는 다음과 같다. A×t의 결과가 K+2 bit이고 B가 K+3bit 이므로 At+B의 결과는 K+4를 넘을 수 없다.

(At+B)t+C를 계산하는 단은 (At+B)t가 K+4, C가 K+2 이므로 유효 결과 bit 수는 K+5이다. 마지막 단은 ((At−B)t+C)t가 K+5, D가 K+1이므로 최종 결과는 K+6이 된다. 위의 결과로부터 파이프라인의 첫단은 K+4 bit, 두번째 단은 K+5 bit 그리고 최종단은 K+6 bit latch가 필요하게 된다.

Ⅲ. 그래픽 시스템의 구성

그래픽 시스템은 그림7과 같이 Master MC 68000과 Slave MC68000과 KAFOG로 구성된다.

Master MC68000은 host 컴퓨터와의 통신, TMS 34061 비디오 콘트롤러 제어 그리고 slave MC68000로 각 문자의 코드와 크기에 대한 정보를 주고, 계산된 글자의 bitmap 데이타를 공유 메모리로부터 프레임 버퍼(video 회로에 포함)로 옮기는 등의 일을 수행한다.

Slave MC68000는 master MC68000로부터 글자의 계산이 요구되면 KAGFO를 이용하여 요구된 글자의 외곽선 데이타를 계산하고, 이 데이타에 대하여 filling 연산을 수행하여 글자의 bitmap 데이타를 공유 메모리에 쓴다. 그리고 font 메모리(local 메모리에 포함)에 있는 외곽선 글자체 정보를 관리한다.

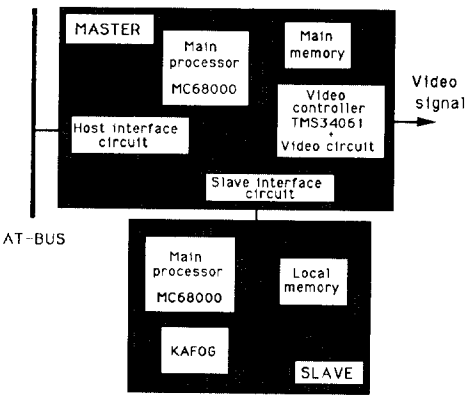


그림 7. 그래픽 시스템의 구조

Fig. 7. The architecture of the graphic system.

1. Master와 Slave간의 통신 규약

Master의 main 메모리 일부를 데이타 버퍼로 할당하며 데이타 버퍼가 비어 있으면 master는 host컴퓨터에게 데이타를 요구한다. 이때 host 컴퓨터와의 통신은 host interface 회로를 통해 수행한다. Host 컴퓨터로부터 데이타를 공급 받으면 master는 아래의 동작을 수행한다. 그래픽 시스템에서 처리되는 데이타는 문자 코드와 2차원 그래픽 함수이다. 문자 코드를 그래픽 시스템이 처리할 경우 master는 slave의 도움을 받게된다. 이때 master는 slave가 처리해야할 문자 코드와 크기에 대한 정보를 slave와 master 사이의 버퍼에 넣어둔다. 그리고 slave에서 계산이 완료되면 master는 공유 메모리로부터 데이타를 프레임 버퍼로 옮긴다.

Slave은 master에 의해 문자 계산을 요구 받으면 동작한다. Slave와 master 사이의 버퍼에 slave에 의해 처리 되어야 할 문자의 코드와 크기에 대한 데이타가 있다. Slave는 이 문자 코드를 받은 후에 이

에 해당하는 문자의 외곽선 정보를 font 메모리로 부터 읽어온다. Slave 프로세서는 KAFOG를 이용해서 외곽선 정보로부터 외곽선 데이타를 계산한다. 계산된 외곽선 데이타는 여러개의 line command 구성된다. 이 외곽선 데이타에 filling 동작을 수행하여 문자의 bitmap 데이타를 계산하여 공유 메모리에 써넣은 후에 master에 요구한 계산이 완료 되었음을 알려준다.

2. 전체 시스템의 동작 환경

제작된 그래픽 시스템은 IBM PC/AT slot의 장착 보드로 사용되며, 사용자와 interface를 위해 IBM PC/AT를 이용하고, 그래픽 시스템에 의해 처리된 데이타의 결과는 그래픽 시스템에 의해 제어되는 1024×768 해상도를 갖는 그래픽 모니터에 디스플레이 된다. 그래픽 시스템에서 수행되는 일은 외곽선 문자체의 데이타를 그래픽 모니터에 나타내기 위한 bitmap 데이타를 계산하는 것과, 2차원 그래픽 함수의 수행이다.

그래픽 시스템에서 수행되는 프로그램과 외곽선 문자 데이타는 그래픽 시스템의 초기화 시에 PC로부터 down loading된다. 그리고 그래픽 시스템에서 처리하는 데이타는 그래픽 시스템이 PC에 데이타를 요구할 때 PC가 그래픽 시스템의 데이타 버퍼에 데이타를 옮겨준다. 그림8은 그래픽 시스템의 동작 환경을 나타낸다.

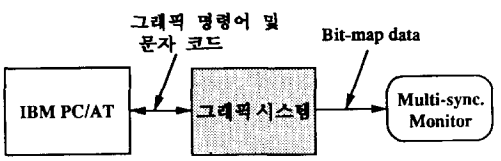


그림 8. 그래픽 시스템의 동작 환경
Fig. 8. Operating environment of the graphic system.

Ⅳ. 결과 및 결론

본 논문에서는 외곽선 문자 발생을 가속시키기 위한 그래픽 시스템을 개발하였다. 개발된 그래픽 시스템은 두개의 MC68000 프로세서가 master-slave 관계로 동작한다. Slave는 외곽선 문자 계산전용 프로세서로 동작하며 KAFOG를 3차 Bezier 곡선 계산 전용 보조 프로세서로 사용한다. Master와 slave 간의 통신으로 handshaking 방법이 사용된다. 그림 9는 본 연구에서 제작된 외곽선 문자계산 전용 그래픽 시스템에서 계산된 외곽선 한글 및 한자를 보여준다.

그림 9. 그래픽 시스템에 의해 계산된 외곽선 한글 및 한자(글자의 크기는 64×64)
Fig. 9. Korean and hinenes outline fonts calculated by the graphic system. (font size is 64×64)

KAFOG는 1.5μm CMOS gate array로 구현되었으며 사용된 gate 수는 약 17000개이다. KAFOG의 최대 성능은 40MHz에서 250K개의 3차 Bezier 곡선을 계산할 수 있다. KAFOG의 곡선 계산 결과는 곡선을 근사화한 직선의 집합으로, 최대성능은 곡선의 크기와는 무관하게 각 곡선을 8개의 직선으로 근사화한 것이다. 그림10은 KAFOG의 레이아웃을 보여준다.

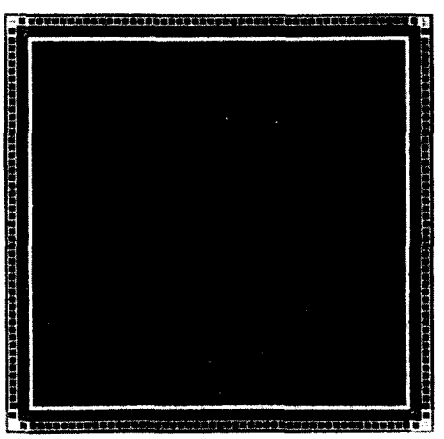


그림10. KAFOG의 레이아웃
Fig. 10. Layout of KAFOG.

KAFOG의 기능에 임의의 폐곡선에 대해 filling을 수행하는 부분을 포함되면 현재 개발된 그래픽 시스템중 slave 부분을 한개의 칩으로 집적화할 수 있을 것으로 생각된다.

參 考 文 獻

[1] Ph . Coueignoux, "Character Generation by Computer," Computer Graphics and Image Processing 16, pp. 240-269, 1981.

[2] Roger D. Hersch, "Character Generation under grid constraints," Computer Graphics, Volumn 21, Number 4, July, pp. 243-251, 1987.

[3] J.D. Foley and A. Van Dam, "Fundamentals of Interactive Computer Graphics," Adisson Wesley, 1984.

[4] Wolfgang Bohm, Gerald Farin, Jurgen Kanmann, "A Survey of Curve and Surface methods in CAGD," Computer Aided Geometric Design 1, pp. 1-60, 1984.

[5] William M. Newman, Robert F. Sproull, "Principal of Interactive Computer Graphics," McGraw Hill, Second edition, 1986.

[6] R.F. Lyon. "Two's Complement Principle Multipliers," IEEE Trans. Comm., pp. 418-425, April, 1976.

著 者 紹 介

李 泰 炯(正會員)

1960年 2月 2日生. 1988年 2月 건국대학교 전자공학과 졸업. 1990年 2月 한국과학 기술원 전기및 전자공학과 석사학위 취득. 현재 삼성전자(주) 가전부문 종합연구소 재직. 주관심분야는 Computer Architecture, Computer Graphics Hardware, DSP, ASIC 등임.

●

黃 奎 哲(正會員)

1964年 9月 16日生. 1987年 2月 경북대학교 전자공학과 졸업. 1989年 2月 한국과학기술원 전기 및 전자공학과 석사학위 취득. 현재 한국과학 기술원 박사과정 재학중. 주관심분야는 VLSI 설계 등임.

●

慶 宗 旻 (正會員) 第28卷 A編 第5號 參照
현재 한국과학기술원 전기 및 전자공학과 정교수

李 潤 泰(正會員)

1960年 7月 19日生. 1983月 2月 서울대학교 전기공학과 졸업. 1985年 2月 한국과학기술원 전기 및 전자공학과 졸업. 1985年 2月 삼성전자(주) 반도체부문 입사. 현재 한국과학기술원 박사과정 재학중. 주관심분야는 Computer Architecture의 VLSI 설계 등임.

●

裵 鍾 洪(準會員)

1966年 9月 15日生. 1989年 2月 한양대학교 전자공학과 졸업. 1991年 8月 한국과학기술원 전기 및 전자공학과 석사학위 취득. 현재 한국과학기술원 박사과정 재학중. 주관심분야는 VLSI 설계, Computer Architecture설계 등임.