논문 2007-44SD-10-7

모바일 3차원 그래픽을 위한 기하변환 엔진 설계

(Design of Transformation Engine for Mobile 3D Graphics)

김 대 경*, 이 지 명**, 이 찬 호***

(Daekyoung Kim, Jeemyong Lee, and Chanho Lee)

요 약

최근 많은 디지털 콘텐츠들이 3차원 그래픽을 기반으로 제작됨에 따라 모바일 기기에 적용 가능한 저 전력 3차원 그래픽 하드웨어에 대한 관심이 증가하고 있다. 본 논문에서는 이러한 시대 흐름에 맞추어 모바일 기기에 적용 가능한 3차원 그래픽 기하변환 엔진을 설계하였다. 설계된 기하변환 엔진은 매핑 변환 유닛을 투영 변환 유닛에 통합하고 클리핑 유닛을 선별 유닛으로 대체하여 구조를 단순화하고 면적을 줄였다. 설계된 엔진은 IEEE-754 표준을 만족하는 32 bit 부동소수점 형식과 데이터 폭을 줄인 24 bit 부동소수점 형식의 연산을 수행할 수 있으며 이는 파라미터의 변환으로 선택할 수 있도록 하였다. 또한 파이프라인 방식을 설계에 적용하여 초기 지연을 제외하고는 매 사이클 입력되는 정점의 좌표 성분(x, y, z, w)을 연산하여 4 사이클 마다 하나의 변환된 정점 좌표 성분을 출력할 수 있도록 하여 동작의 속도 및 효율을 높였다. 설계된 기하변환 엔진은 FPGA 를 이용한 시스템으로 구현되었으며 설계된 엔진을 통해 변환된 3차원 객체가 TFT-LCD에 정상적인 3차원 그래픽 영상을 출력하는 것을 통해 검증하였다.

Abstract

As digital contents based on 3D graphics are increased, the requirement for low power 3D graphic hardware for mobile devices is increased. We design a transformation engine for mobile 3D graphic processor. We propose a simplified transformation engine for mobile 3D graphic processor. The area of the transformation engine is reduced by merging a mapping transformation unit into a projective transformation unit and by replacing a clipping unit with a selection unit. It consists of a viewing transformation unit, a projective transformation unit, a divide by w unit, and a selection unit. It can process 32 bit floating point format of the IEEE-754 standard or a reduced 24 bit floating point format. It has a pipelined architecture so that a vertex is processed every 4 cycles except for the initial latency. The RTL code is verified using an FPGA.

Keywords: 3D 그래픽, 기하변환, 클리핑, 저전력, 선별 유닛

I. 서 론

최근 그래픽 하드웨어의 발달로 인하여 많은 디지털 콘텐츠들이 3차원 그래픽을 기반으로 제작되고 이들이 2차원 그래픽 기반의 콘텐츠에 비하여 좋은 반응을 보임에 따라 많은 사용자들을 확보하고 있는 휴대폰, PMP, PDA 등의 모바일 기기에서도 3차원 그래픽 시스템에 대한 관심이 증가하고 있다. 3차원 그래픽 처리 과정은 2차원 그래픽에 비하여 많은 양의 연산을 필요로 하고 그 연산 또한 복잡하기 때문에 유한한 자원과 작은 크기를 특징으로 하는 모바일 기기에 적용하는 것은 매우 어려운 일이었다. 그러나 최근 SoC (system on chip) 기술의 발전에 따라 작은 면적을 가지고 보다 적은 전력을 소모하며 고속으로 동작하는 모바일 환경 기반의 3차원 그래픽 하드웨어 연구가 활발히 진행되고 있다. 그 중 3차원 그래픽을 구성하는 파이프라인의 일부인 기하변환 단계는 3차원 객체 대해 크기, 회전, 이동, 투영 등의 변환 연산을 수행하여 3차원 객체가 2차

* 학생회원, *** 정회원, 숭실대학교
(Dept. of Electronic Engineering, Soongsil University)
** 정회원, (주)넥스트칩
(NEXTCHIP CO.)
※ 본 연구는 서울시 산학연 협력사업과 ETRI SoC산업진흥센터에서 수행한 IT SoC 핵심설계인력양성 사업의 지원으로 수행되었으며 IDEC의 SW 툴을 지원받았습니다.
접수일자: 2007년5월26일, 수정완료일: 2007년9월11일

원 평면상에 표현될 형태 및 위치를 결정하는 역할을 하며 이러한 변환 연산들은 3차원 객체를 구성하고 있는 정점(vertex) 단위의 연산을 통하여 이루어진다. 따라서 기하변환 단계의 연산 양은 최종 2차원의 영상을 표현하기 위해 사용되는 3차원 객체의 수와 질 (객체를 구성하는 정점의 수)에 의해 결정되며 보다 사실에 가까운 영상이 요구되어질수록 그 연산 양은 기하급수적으로 증가하게 된다. 이러한 연산을 모바일 환경에서 사용되는 저 전력 중앙처리장치(CPU)를 이용하여 소프트웨어로 처리할 경우 3차원 그래픽의 기하변환에 사용되는 모든 연산이 부동소수점 연산이라는 것을 감안하면 중앙처리장치가 과중한 연산 양을 처리해야 하므로 제 역할을 수행할 수 없어 고품질의 3차원 그래픽을 실시간으로 표현하는 것은 불가능에 가깝다고 할 수 있다. 따라서 모바일 환경에서 실시간으로 보다 사실적인 고품질의 3차원 그래픽 영상을 얻기 위해서는 고성능의 기하변환 하드웨어 엔진은 필수적이다.

본 논문에서는 기존의 기하변환 과정을 수정하여 전체적인 연산을 단순화하고 전력 소모를 줄인 기하변환 엔진 구조를 제안한다. 제안하는 구조는 매핑 변환 연산 순서를 바꾸어 투영 변환 연산에 통합하고 연산 양과 사이클이 많은 클리핑 연산은 훨씬 간단한 선별 연산으로 대체하여 전체적인 기하변환 연산을 보다 단순화 하여 전체 연산의 양을 감소시켰으며 엔진의 입출력 데이터 폭을 최소화하여 저 전력으로 동작할 수 있도록 하였다. 이러한 구조로 설계된 기하변환 엔진은 합성 시 파라미터 변환을 통하여 IEEE-754 표준의 32 bit 단정도 형식과 이를 축소한 형태인 24 bit 부동소수점 형식을 선택 가능하도록 하여 응용 환경에 따라 입력 데이터를 결정할 수 있게 하였으며 파이프라인 방식을 적용해 보다 고속으로 동작할 수 있다. 또한 제안하는 구조는 독립 삼각형 방식의 입력뿐 아니라 팬(fan)과 스트립(strip) 방식의 입력도 처리 할 수 있도록 하여 동일한 3차원 객체를 보다 적은 수의 정점으로 처리할 수 있게 함으로서 연산 사이클과 전력 소모를 줄이는 효율적인 연산이 가능하게 하였다. 설계된 기하변환 엔진은 FPGA를 이용한 시스템에 적용하여 동작을 검증하였다.

II. 효율적인 기하변환 엔진 구조

1. 제안하는 기하변환 엔진의 특징

가. 투영 변환 및 매핑

투영 변환은 3차원으로 모델링 된 객체를 2차원 화면에 나타내기 위하여 2차원으로 변환하는 과정이며 매핑 변환은 최종 출력될 2차원 영상의 크기를 결정하는 변환 과정이다. 제안하는 기하변환 엔진에서는 이러한 투영 변환에 매핑 과정을 포함하여 한번의 4x4 행렬과 1x4 벡터간의 곱셈으로 이 두 과정을 동시에 수행하도록 한다. 이러한 연산은 내부적으로 연산 순서를 바꿔 기존의 투영 변환 결과로 출력되는 x, y, z, w 의 인자 중 w를 먼저 구하여 매핑에 영향을 주는 x, y 값에 적용함으로서 가능해 진다. 다음 수식 (1)은 제안하는 투영변환 유닛의 행렬 연산을 나타내며 p 인자는 기존 투영변환 연산에 사용되던 값을 m 인자는 기존 매핑 변환에 사용되던 값을 나타낸다.

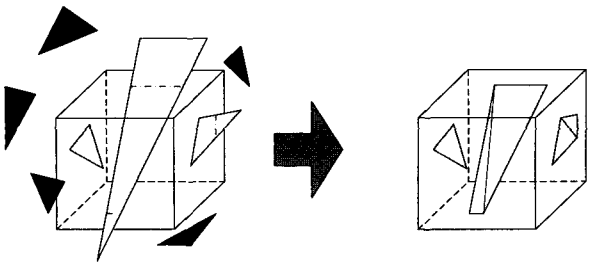
$$\begin{bmatrix} x' \\ y' \\ z' \\ w \end{bmatrix} = \begin{bmatrix} m_{sx} * p_{11} & m_{sx} * p_{12} & p_{13} & p_{14} + m_{tx} / w \\ m_{sy} * p_{21} & m_{sy} * p_{22} & p_{23} & p_{24} + m_{ty} / w \\ p_{31} & p_{32} & p_{33} & p_{34} \\ 0 & 0 & p_{43} & p_{44} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \quad (1)$$

투영 변환과 매핑 변환 과정을 통합함으로서 매핑 변환 연산에 소비되던 사이클 및 연산기를 제거하는 효과를 얻어 동작속도의 증가 및 전체 면적 감소의 효과를 얻을 수 있으며 파이프라인 스톨이 발생할 수 있는 클리핑 연산을 기하변환 연산의 마지막 단계로 보내는 장점이 있다.

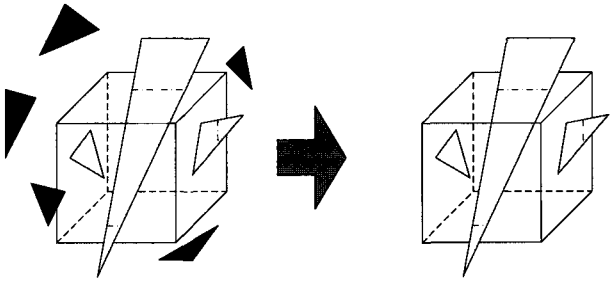
나. 클리핑

클리핑은 관측자 시점에서 화면에 보이지 않는 부분을 제거하는 과정으로 이때 화면에 보이는 부분까지를 클리핑 영역(Clipping region)이라고 한다. 기존의 클리핑 처리에는 Cohen-Sutherland와 Liang-Barsky의 클리핑 알고리즘을 혼합하여 3차원 형태로 확장한 형태의 알고리즘이 주로 사용되어 졌다. 이 방식은 클리핑 범위에 걸쳐있는 객체들을 연산하기 위하여 많은 연산 양을 필요로 한다[1]. 또한 이 과정에서 클리핑 범위에 걸쳐있는 삼각형은 분할하여 클리핑 범위내의 새로운 삼각형으로 재구성하는데 이 과정에서 새로운 정점이 생성될 수 있다.

그림 1.(a)에서는 회색 삼각형을 구성하는 9개의 정점이 클리핑 수행결과 15개로 증가한 것을 보여준다. 이는 기하연산 엔진의 파이프라인에 스톨을 발생시키면서 래스터라이저에 전달해야하는 정점의 수를 증가시키고 클리핑 연산을 복잡하게 만드는 원인이 되기도 한다. 따라서 제안하는 기하변환 엔진에서는 그림 1.(b)와 같이 클리핑 범위에 걸쳐있는 객체들은 제거하지 않고



(a) 기존 클리핑 연산



(b) 제안하는 선별유닛 연산

그림 1. 기존 클리핑 연산과 선별유닛 연산의 결과 비교

Fig. 1. Comparison of result with general clipping.

간단한 비교 연산을 통하여 클리핑 범위를 완전히 벗어하는 객체들만을 제거하는 선별 유닛을 제안하고 클리핑 과정을 대체하였다. 클리핑 범위에 걸쳐져 있는 객체에 대하여 기하변환 엔진에서 처리하지 않는 이유는 래스터라이저의 변 처리 및 스팬 처리 과정에서 클리핑 범위에 걸쳐져 있는 부분을 연산하는 과정과 유사한 연산이 진행되므로 이 때 나머지 클리핑 과정을 수행할 수 있기 때문이다. 이러한 구조적 개선을 통해 기하변환 엔진은 클리핑 범위에 걸쳐져 있는 객체에 대한 처리를 위한 하드웨어 제거 및 연산 시간의 단축, 파이프라인 스톨 방지 등의 효과를 얻을 수 있다.

다. 정점 데이터 입력 방식

제안하는 기하변환 엔진은 정점 데이터를 독립 삼각형, 스트립, 팬 방식으로 입력 받아 처리 할 수 있도록 하였다. 그림 2는 각 입력 방식 별로 3개의 삼각형 생성에 대한 정점 입력을 나타내며 표시된 숫자는 입력되는 정점의 순서 및 수를 나타낸다. 일반적으로 3차원 모델링 소프트웨어에서 모델링 데이터를 추출할 경우 독립 삼각형 방식으로 데이터가 출력되며 이러한 형태로 데이터가 입력될 경우 이웃한 두 번째 이후의 삼각형은 두 개의 정점이 이전 삼각형과 중복되어 입력되므로 같은 수의 삼각형을 생성할 경우 스트립 방식이나 팬 방식에 비하여 입력되는 정점의 수가 많아지게 된다.[2] 예

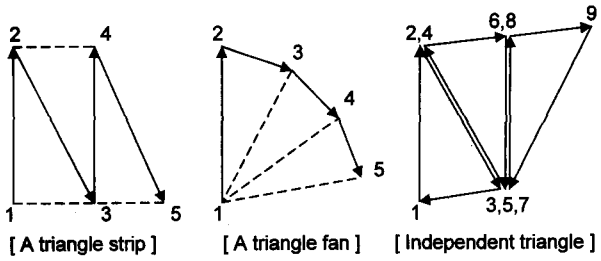


그림 2. 정점 데이터 입력 방식

Fig. 2. Various methods of input sequences of vertices.

표 1. 독립 삼각형 방식과 스트립 방식의 입력 정점 수 비교

Table 1. Comparison of triangle formation methods using independent triangles and triangle strip.

	# of vertex	# of face	# of input vertex		
			independent triangle	triangle strip	
				strip	triangle
Sphere	86	168	504	15	272
Torus	170	340	1020	34	543
Chess	1036	2068	6204	160	3343

를 들어 스트립 방식일 경우 100 개의 삼각형이 하나의 스트립으로 입력된다면 102 개의 정점 데이터만을 연산하게 되지만 독립 삼각형 방식의 데이터가 입력된다며 300 개의 정점에 대한 연산이 필요하게 된다. 따라서 스트립이나 팬 방식을 이용할 경우 독립 삼각형 방식에 비해 연산 사이클을 최대 1/3로 줄일 수 있다. 표 1에는 검증에 사용된 3차원 객체에 대해 독립 삼각형 방식과 스트립 방식에 따른 정점 수의 차이를 보여준다.

이러한 입력 방식에 의한 입력 정점 수의 차이는 입력되는 3차원 객체가 복잡해질수록 더욱 증가하게 되며 정점 단위로 연산을 수행하는 기하변환 엔진의 경우 일반적으로 독립 삼각형 방식의 정점 입력은 연산 양을 증가시키는 요인이 된다. 이러한 이유로 제안하는 기하변환 엔진은 독립 삼각형 형태의 입력 뿐 만아니라 스트립, 팬 형태의 입력 데이터도 처리할 수 있도록 설계하여 기하변환 엔진의 연산 효율을 높이고 다양한 입력 방식의 적용이 가능하도록 하였다. 스트립과 팬 형태로 데이터가 입력될 경우 클리핑 단계와 삼각형 형성 단계에서 독립 삼각형 방식과의 연산 차이가 발생하게 된다. 특히 클리핑의 단계의 경우 새로운 정점의 생성으로 인하여 입력되지 않은 삼각형이 발생하기 때문에 기존의 기하변환 엔진은 스트립이나 팬 구조를 처리하는 데 어려움이 있었다. 그러나 제안하는 구조에서는 클리핑 단계를 선별 단계로 대체하여 이러한 문제의 발생 요인을 제거하고 입력 방식에 따른 별도의 제어신호

를 엔진에 전달하여 삼각형 형성 단계의 연산 시 입력 방식에 적합한 처리를 할 수 있도록 하였다.

2. 제안하는 기하변환 엔진 구조

본 논문에서 제안하는 기하변환 엔진은 투영변환과 매핑 연산을 통합하고 복잡한 클리핑 연산을 선별 유닛으로 대체하여 그림 3과 같은 구조를 갖는다. 매핑 유닛이 투영 유닛에 통합되고 클리핑 유닛이 선별 유닛으로 바뀌었음을 알 수 있다. 연산 사이클은 1개의 삼각형에 대해 기존 구조에서 30 에서 63 사이클이 소요되면서 파이프라인 스톨이 발생할 수 있으나 제안된 구조는 21사이클로 감소하면서 파이프라인 스톨도 발생하지 않는다.

제안하는 기하변환 엔진은 정점 좌표 및 각 변환 유닛별 연산 인자, 그리고 제어 정보를 입력 받아 기하변환 연산을 수행하고 변환된 정점 좌표를 출력하는 과정을 파이프라인 방식으로 설계하여 초기 지연 시간을 제

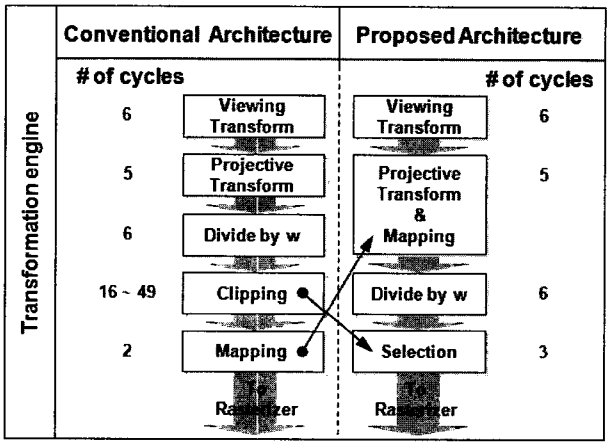


그림 3. 제안하는 기하변환 엔진 구조
Fig. 3. Architecture of the proposed transformation engine.

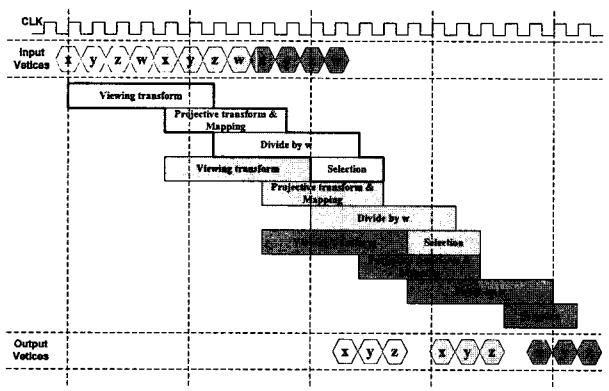


그림 4. 제안하는 기하변환 엔진의 파이프라인
Fig. 4. Pipeline of the proposed transformation engine.

표 2. 제안하는 기하변환 엔진의 유닛 별 연산기 수
Table 2. Numbers of arithmetic units for the proposed transformation engine.

	Viewing Transform	Projective Transform	Divide by w	Selection
Adder	3	3	1	one comparator
Multiplier	3	4	1	
Inversion	0	0	1	

외하고는 매 4 사이클 마다 하나의 변환된 정점 좌표를 래스터라이저로 전달한다.

그림 4는 제안하는 구조의 파이프라인 구조로 한 개의 삼각형을 이루는 3개의 정점 입력에 대한 내부 연산 진행을 나타낸다. 그림 4의 파이프라인 구조에서 하나의 변환된 정점 좌표가 4사이클 마다 출력되는 이유는 제안하는 구조가 모바일 환경을 목표로 설계하였기 때문에 메모리에서 데이터를 읽어오거나 래스터라이저로 데이터를 전달할 때 사용하는 버스의 대역폭을 줄여 보다 적은 전력을 소모하게 하기 위함이다. 설계된 기하변환 엔진은 파라미터 설정에 따라 IEEE-754 표준의 32 bit 단정도 및 이를 축소한 24 bit 부동소수점 연산을 각각 지원할 수 있도록 설계하여 전체 엔진의 입출력 데이터 또한 32 bit 또는 24 bit의 데이터 폭을 갖게 된다. 기하변환 엔진에 사용된 부동소수점 연산기들은 고속의 연산을 위해 동작 지연시간을 최소화 하였으며 그 결과 덧셈과 곱셈은 각각 1 사이클, 역수기(inversion)는 3 사이클의 동작 지연 시간을 갖는다. 표 2는 제안하는 기하변환 엔진의 각 유닛별로 사용되는 부동소수점 연산기의 수를 나타내고 있다.

III. 설계 및 검증

제안하는 구조의 3차원 그래픽 기하변환 엔진은 Verilog-HDL을 이용하여 설계되었으며 설계된 기하변환 엔진을 검증하기 위하여 그림 5와 같이 FPGA를 이용한 검증 시스템을 구성하였다. FPGA 에는 ARM 프로세서와의 통신을 위한 ARM Stripe, 설계된 기하변환 엔진, 영상 출력을 위한 TFT-LCD 컨트롤러 그리고 이러한 IP(intellectual property)들 간의 통신을 위한 AMBA AHB로 구성하였으며 기하변환 엔진 이외의 검증에 필요한 3차원 그래픽 시스템 구성요소들 및 전체 시스템의 제어 부는 소프트웨어로 구성하여 기하변환 엔진에서 변환된 정점 정보를 TFT-LCD를 통하여 확인할 수 있도록 하였다. 검증시스템 입력 데이터는

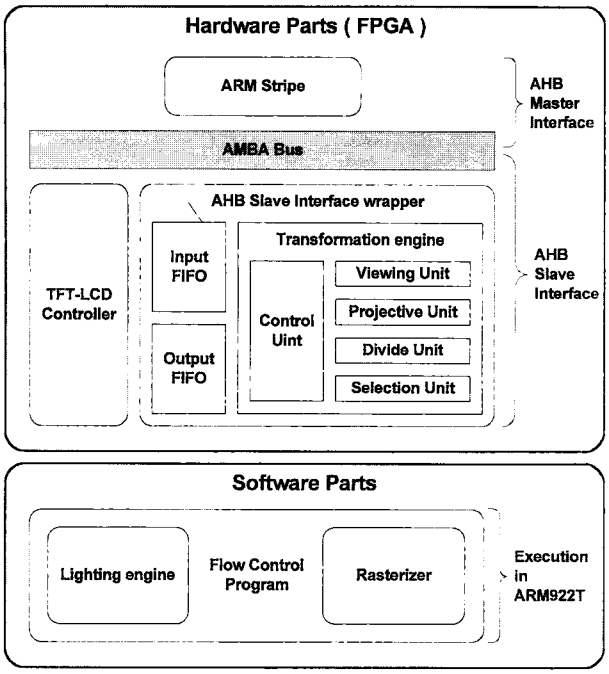


그림 5. ARM 프로세서와 FPGA를 이용한 검증 시스템 구성도

Fig. 5. Block diagram of verification system using an FPGA and an ARM processor.

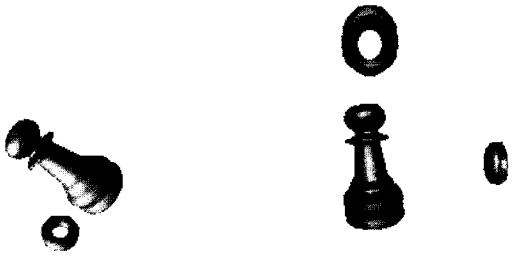


그림 6. FPGA 검증 시스템의 출력 결과

Fig. 6. Screen images of FPGA verification system.

1,036개의 정점을 갖는 체스 폰 모델, 170개의 정점을 갖는 토러스, 그리고 86개의 정점을 갖는 구등 다양한 3차원 객체를 사용하였고, 검증 시스템을 통하여 입력된 3차원 객체들이 기하변환 엔진을 통하여 이동, 신축, 회전 및 투영 변환되는 결과를 그림 6과 같이 TFT-LCD에 출력된 영상을 통해 확인하였다.

표 3에는 Synopsys design compiler와 0.25um CMOS 셀 라이브러리를 이용하여 기하변환 엔진을 합성한 결과가 나타나 있다. 합성은 100 MHz 동작 주파수를 기준으로 진행되었으며 32 bit 연산을 기준으로 약 96,000 게이트 크기, 24 bit 연산을 기준으로 약

표 3. 0.25um CMOS 셀 라이브러리를 이용한 합성 결과 (100 MHz 동작 주파수 기준)

Table 3. Synthesis results using 0.25um CMOS technology at 100MHz.

	projection & mapping	Clipping	Total
# of gate	52,600	28,000	129,000
	projection & mapping	Selection	Total
# of gate	38,600	8,300	96,000

표 4. 기존 설계 결과와의 성능 비교

Table 4. Comparison of performance with other transformation engines.

	Performance	area [# of gate]
[3]	4.3M vertices/sec, 100MHz	N/A
[4]	20M vertices/sec, 80 MHz	139,190
Proposed engine	24M vertices/sec, 100 MHz	96,000

63,700 게이트 크기를 갖으며 이는 기존 방식에 비해 약 26% 감소된 면적이다. 구현된 기하변환 엔진은 최대 초당 24M 정점 처리 능력을 가진다.

표 4에는 다른 기하변환 엔진과의 성능을 비교한 내용이 나타나 있다. 그러나 비교 대상의 입력 형태 및 기능 등의 차이로 직접적인 비교는 어렵다.

IV. 결 론

본 논문에서는 모바일 기반의 3차원 그래픽 프로세서에 적용 가능한 3차원 기하변환 엔진의 새로운 구조를 제안하고 제안한 구조를 기반으로 기하변환 엔진을 설계 및 검증하였다. 설계된 기하변환 엔진은 전형적인 기하변환 연산 과정의 변경 및 통합을 통하여 면적 및 연산 양을 감소시켰으며 파이프라인 방식으로 설계하여 동작의 효율성을 증가 시켰다. 제안하는 구조는 투영 변환에 매핑 변환을 통합하고 복잡한 클리핑 연산을 간단한 선별 연산으로 대체하였으며 삼각형 형성을 선별 연산 시 미리 진행하여 래스터라이저의 연산을 줄일 수 있도록 하였다. 그 결과 연산 사이클을 기존 30~63에서 21로 감소시켰다. 또한 스트립, 팬 방식의 입력을 지원함으로써 동일한 3차원 객체를 최대 1/3로 감소한 적은 수의 정점 입력으로 처리될 수 있도록 하여 연산 사이클을 동일 비율로 감소시킬 수 있다. 내부 연산기들은

IEEE-754 표준을 만족하는 32 bit 단정도 형식 및 이를 축소한 형태의 24 bit 부동소수점 연산을 수행할 수 있도록 하여 설계된 기하변환 엔진이 다양하게 적용될 수 있도록 하였다. 설계된 기하변환 엔진은 FPGA와 ARM 프로세서를 이용한 검증시스템을 통해 동작을 검증하였고 0.25um CMOS 기술을 이용하여 100 MHz 동작주파수를 기준으로 합성했을 때 32bit 연산을 기준으로 약 96,000 게이트, 24bit 연산을 기준으로 약 63,700 게이트 정도의 면적을 가져 기존 구조에 비해 26% 감소하였다.

참 고 문 헌

[1] EDWARD ANGEL, "Interactive Computer Graphics", Addison Wesley, p340-343
[2] David Kornmann, "Fast and Simple Triangle Strip Generation", VMS Finland, Espoo, Finland, 1999.
[3] 이마음, "휴대단말기용 3차원 그래픽 기하프로세서 설계", 서울시립대학교 석사학위 논문, 2006년 2월
[4] 송인석, "SoC 플랫폼 기반 모바일용 3차원 그래픽 T&L Processor 구현", 서경대학교 석사학위 논문, 2007년 2월
[5] TOMAS AKENINE-MOLLER, ERIC HAINES, "REAL_TIME RENDERING", A K PETERS, p.27-30, p.95-96, 2002.
[6] Mark Segal, Kurt Akeley, "The OpenGL Graphic System : A Specification(Version 1.2.1)", Silicon Graphics, 1998.10
[7] Richard S. Writh. Jr, Benjamin Lipchak, "OpenGL SUPERBIBLE 3rd Edition", Que/Sams, 2004.

저 자 소 개

김 대 경(학생회원)
2006년 숭실대학교 정보통신전자공학부 학사졸업.
2006년~현재 숭실대학교 전자공학과 석사재학.
<주관심분야 : 3D 그래픽 프로세서 설계, SoC 설계방법론, SoC 플랫폼 설계>

이 지 명(정회원)
대한전자공학회 논문지
제42권 SD편 제6호 참조
현재 (주) 넥스트칩 주임연구원

이 찬 호(정회원)
대한전자공학회 논문지
제43권 SD편 제9호 참조
현재 숭실대학교 정보통신전자공학부 부교수