논문 2007-44SD-10-4

모바일 멀티미디어의 효율적 처리를 위한 재구성형 병렬 프로세서의 구조

(A Reconfigurable Parallel Processor for Efficient Processing of Mobile Multimedia)

유 세 훈*, 김 기 철**, 양 일 석***, 노 태 문***

(Sehoon Yoo, Kichul Kim, Yil Suk Yang, and Tae Moon Roh)

요 약

본 논문에서는 3D 그래픽스(graphics), H.264/H.263/MPEG-4 같은 동영상 코덱, JPEG 혹은 JPEG2000 같은 정지영상 코덱, MP3 같은 오디오 코덱 등 다양한 멀티미디어 관련 기술을 효율적으로 구현하기 위한 재구성형 병렬 프로세서 구조가 제안된다. 제안된 구조는 메모리와 프로세서를 직접 연결하여 메모리 접근 시간과 소비전력를 감소시키고, 3D 그래픽스 처리 과정 중 기하 단계의 부동소수점 연산을 지원한다. 또한 분할 SIMD(partitioned SIMD) 방식을 사용하여 하드웨어 비용을 줄이고, 명령어(instruction)의 조건부 실행(conditional execution)을 지원하여 알고리듬 개발이 용이하다.

Abstract

This paper proposes a reconfigurable parallel processor architecture which can efficiently implement various multimedia applications, such as 3D graphics, H.264/H.263/MPEG-4, JPEG/JPEG2000, and MP3. The proposed architecture directly connects memories and processors so that memory access time and power consumption are reduced. It supports floating-point operations needed in the geometry stage of 3D graphics. It adopts partitioned SIMD to reduce hardware costs. Conditional execution of instructions is used for easy development of parallel algorithms.

Keywords: Parallel processor, multimedia, mobile, SIMD, conditional execution

I. 서 론

휴대폰, PDA, PMP 등 다양한 휴대용 단말기에서 동영상, 정지영상, 오디오, 3D 그래픽스 등 다양한 매체에 대한 실시간 서비스를 제공해야 할 필요성이 증대되고 있다. 휴대용 단말기에 내장된 범용 마이크로프로세서는 그 성능이 일반 PC에 비해 낮고 여러 가지 일을 수행해야 하므로 실시간 수행이 필요한 경우 각각의 매체에 대한 전용 하드웨어를 단말기에 탑재시키는 방법을 사용할 수 있다. 이러한 방법을 사용하면 처리해야 하는 매체의 수가 늘어날수록 전용 하드웨어가 증가된다. 이러한 전용 하드웨어의 증가는 비용의 증가는 물론이

* 학생회원, ** 정회원, 서울시립대학교 전자전기컴퓨터공학부
(School of Electrical and Computer Engineering, University of Seoul)
*** 정회원, 한국전자통신연구원 U-단말 연구팀
(U-Terminal Research Team, Electronics and Telecommunications Research Institute)

※ 본 논문은 정보통신부 및 정보통신연구진흥원의 IT신성장동력핵심기술개발사업의 부분 지원으로 수행하였습니다.[2006-S-006-01, 유비쿼터스 단말용 부품/모듈]
※ 본 논문은 정보통신부 출연금으로 ETRI, SoC산업진흥센터에서 수행한 IT SoC 핵심설계인력양성사업의 부분 지원으로 이루어진 연구결과입니다. 본 논문의 내용을 발표할 때에는 반드시 ETRI, SoC산업진흥센터, IT SoC 핵심설계인력양성사업의 부분 지원으로 이루어진 결과임을 밝혀야 합니다.
※ 본 논문은 IDEC의 부분 지원으로 이루어졌습니다.
접수일자: 2007년5월26일, 수정완료일: 2007년9월19일

며 휴대용 단말기의 효용성을 감소시키는 소비전력의 증가를 가져오게 된다.

각 매체의 서비스를 위한 전용 하드웨어들을 사용하는 방법에 대한 대안으로 병렬 프로세서를 사용할 수 있다. 이는 하나의 병렬 프로세서를 이용하여 필요에 따라 다양한 매체에 대한 서비스를 제공하는 것이다. 이런 방법을 사용하면 동영상 매체에 대한 서비스를 수행할 때는 병렬 프로세서에서 동영상 매체를 위한 알고리듬이 수행되며, 3D 그래픽스에 대한 서비스를 제공할 때는 병렬 프로세서에서 기하 변환(geometry transform)이나 래스터라이제이션(rasterization)[1]을 위한 알고리듬들이 수행된다. 이렇게 병렬 프로세서를 사용하는 방법은 각 매체마다 전용 하드웨어를 사용하는 방법에 비하여 저비용, 저전력, 유연성, 고성능의 장점을 가질 수 있다.

병렬 프로세서의 기능과 성능은 연산을 수행하는 PE(Processing Element)와 각 PE들을 연결시키는 내부 연결망(interconnection network)의 기능 및 성능에 의하여 결정된다. 일반적으로 병렬 프로세서는 수행하는 알고리듬에 따라 그 성능이 크게 변한다. 즉 어떠한 알고리듬을 효율적으로 수행하는 병렬 프로세서가 또 다른 알고리듬을 수행할 때는 비효율적일 수 있다. 따라서 동영상, 정지영상, 오디오, 3D 그래픽스 등 다양한 매체에 대한 서비스를 제공해야 할 필요가 있는 경우에 각각의 매체에 대한 서비스를 효율적으로 수행할 수 있는 병렬 프로세서의 구조는 서로 다를 수 있다. 그러므로 동영상, 정지영상, 오디오, 3D 그래픽스에 대하여 넓게 사용될 수 있는 재구성형 병렬 처리 구조와 다양한 병렬 알고리듬들이 개발되어야 한다.

제안된 재구성형 병렬 프로세서는 여러 멀티미디어 관련 기술을 하나의 휴대용 기기에서 효율적으로 지원하기 위한 구조를 가지고 있다. 메모리와 PE를 직접 연결하여 메모리 접근 시간과 전력 소모를 감소시키고, 3D 그래픽스 처리 과정의 기하 단계(geometry stage)에서 사용되는 부동소수점 연산에 대한 지원을 한다. 또한 분할 SIMD와 조건부 실행을 지원해 저비용으로 효과적인 알고리듬 개발이 가능하게 한다.

본 논문의 나머지 부분은 다음과 같이 구성되어 있다. Ⅱ장에서 재구성형 병렬 프로세서의 구조를 제안하고 Ⅲ장에서 제안되는 재구성형 병렬 프로세서의 기능적 특성을 설명한다. Ⅳ장에서 명령어에 대해 살펴보고 명령어를 사용한 간단한 예제를 보인다. Ⅴ장에서 제안된 재구성형 병렬 프로세서를 기반으로 개발된 병렬 알고리듬에 대해서 언급한다. Ⅵ장에서 기존에 연구된 병렬 프로세서와 구조 및 알고리듬 성능을 비교한다. Ⅶ장에서는 제안되는 재구성형 병렬 프로세서의 함수적 검증 및 합성 결과에 대해서 살펴본다. 마지막으로 Ⅷ장에서 본 논문의 결론을 맺는다.

Ⅱ. 재구성형 병렬 프로세서 구조

1. 재구성형 병렬 프로세서 전체 구조

제안된 재구성형 병렬 프로세서는 크게 PE 어레이(PE array)와 그와 직접 연결된 로컬 메모리(local memory), 그리고 그 블록들을 제어하는 제어 유닛(control unit)으로 구성된다. 그림 1에 제안된 재구성형 병렬 프로세서의 전체 구조가 나타나 있다. PE 어레이는 4x24의 PE의 배열로 이루어져 있고 PE들은 그물형 망(mesh network)[2]으로 서로 연결되어 있다. 제어 유닛은 외부 메모리로부터 명령어를 받아 PE 어레이의 모든 PE에 브로드캐스팅(broadcasting)하는 역할과 PE가 로컬 메모리와 통신을 할 때 로컬 메모리에 주소를 인가하는 역할을 한다. PE 어레이 위쪽에는 부동소수점 덧셈을 가속하기 위한 부동소수점 누산기(floating-point accumulators) 24개가 한 행으로 배치되어 있다. 각 부동소수점 누산기는 PE 어레이의 가장 위쪽 행의 PE들 중 같은 열의 PE와 데이터를 주고받을 수 있도록 연결되어 있다.

2. 내부 연결망

모든 PE는 그물형 망 구조를 통해 이웃한 PE들과 연결되어 있다. 그림 2에 내부 연결망 구조가 나타나

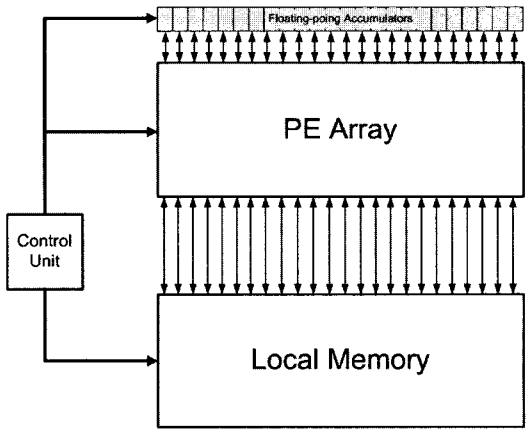


그림 1. 재구성형 병렬 프로세서 전체 구조
Fig. 1. Architecture overview of the proposed reconfigurable parallel processor.

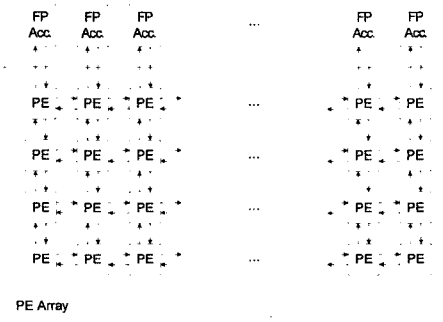


그림 2. 내부 연결망
Fig. 2. Interconnection network.

있다. 각 부동소수점 누산기는 PE 어레이의 첫 번째 행에 있는 PE들과 연결되어 있다. PE와 부동소수점 누산기의 데이터 통신 방법은 이웃한 PE 간의 데이터 통신 방법과 비슷하다.

3. 로컬 메모리

로컬 메모리는 PE들과 직접 연결되어 있다. 그러므로 하나의 칩에 로컬 메모리와 PE 어레이가 구현되면 PE가 로컬 메모리에 접근할 때 적은 전력으로 데이터 통신이 가능하다. 그림 3에 로컬 메모리와 PE 간의 연결 구조와 단일 주소 방식이 나타나 있다. 그림 3에 나타나는 바와 같이 각각의 PE는 독립적으로 로컬 메모리에 있는 메모리 뱅크(memory bank)와 각각 연결되어 있다. 각 메모리 뱅크는 2 킬로 바이트(kilo-bytes)의 데이터 저장 공간을 가지고 있으므로, 4x24=96개의 메모리 뱅크를 포함하는 로컬 메모리는 총 192 킬로 바이트의 저장 공간을 가지고 있다. 또한 모든 메모리 뱅크는 항상 같은 주소값을 사용하는 단일 주소 방식(uniform addressing)을 사용한다.

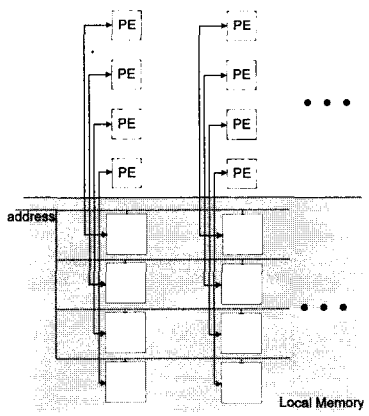


그림 3. 로컬 메모리와 PE의 연결과 단일 주소 방식
Fig. 3. Connection of the local memory with PEs and uniform addressing.

4. PE

본 절에서는 PE와 PE에 포함된 하드웨어 기능 블록들의 구조에 대해서 설명한다.

가. PE의 구조

제안된 재구성형 병렬 프로세서에서 PE는 데이터 통신, 연산의 동작을 16비트 단위로 수행한다. PE 어레이를 구성하는 각 PE의 구조가 그림 4에 나타나 있다. 그림에 나타난 것과 같이 PE는 위(north), 아래(south), 오른쪽(east), 왼쪽(west)에 위치하는 이웃 PE와 데이터를 주고 받을 수 있는 입출력 포트를 가지고 있다. 연산은 곱셈기(multiplier), 산술논리장치(arithmetic/logic unit), 시프터(shifter)를 포함하는 기능 유닛(functional unit)에서 이루어진다. 이 때 연산의 입력은 레지스터 파일(register file)로부터 얻어지고, 출력 또한 레지스터 파일로 저장된다. 레지스터 파일은 두 개의 입력 포트를 가지고 있어서 한 번에 두 개의 데이터를 레지스터 파일 안의 서로 다른 레지스터에 각각 저장할 수 있다. 이러한 구조는 기능 유닛에서 연산을 수행하고 그 결과를 저장하는 동시에, 이웃한 PE로부터 데이터를 받거나 로컬 메모리로부터 데이터를 불러오는 것을 가능하게 한다. 로컬 메모리에 데이터를 저장하거나 로컬 메모리로부터 데이터를 불러오기 위해서 M.W.R.(Memory Write Register)와 M.R.R. (Memory Read Register)를 사용한다. PE 안의 모듈들의 동작은 명령어 디코더(instruction decoder)의 출력에 의해 결정된다. 명령어 디코더는 PE 외부의 제어 유닛으로부터 명령어를 입력

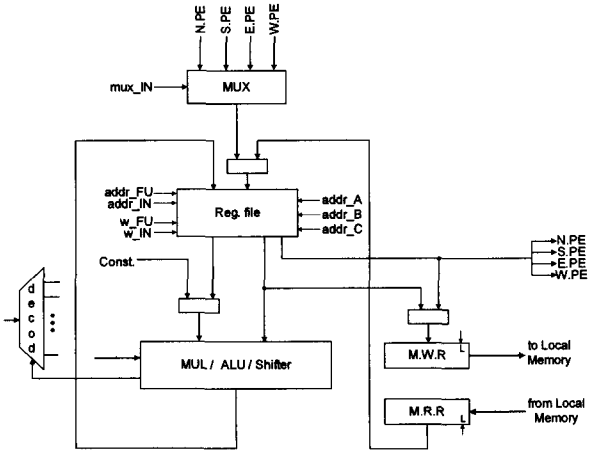


그림 4. PE의 구조
Fig. 4. Processing element.

으로 받고 PE 안의 모듈들의 동작을 결정하기 위한 모든 제어 신호를 발생시키는 역할을 한다.

나. 기능 유닛

기능 유닛은 PE의 동작 중 연산에 관련된 부분을 담당한다. 포함되는 모듈은 곱셈기, 산술논리장치, 시프터이다. 기능 유닛에 포함된 곱셈기는 18비트 2의 보수 배열 곱셈기(two's complement array multiplier)를 사용한다. 재구성형 병렬 프로세서의 기본 처리 단위가 16비트인데 18비트 2의 보수 배열 곱셈기를 사용한 이유는 16비트 2의 보수 곱셈, 16비트 정수 곱셈, 16비트 가수(mantissa) 곱셈을 모두 지원하기 위함이다. 16비트 가수 곱셈은 24비트 부동소수점 형식을 사용하는 3D 그래픽스의 부동소수점 곱셈을 위한 것으로, 24비트 부동소수점 형식의 가수 부분이 16비트로 정의되기 때문에 필요하다. 16비트 가수 곱셈을 수행하고 나서 결과값 중 일부를 부동소수점 곱셈용 플래그(flag)에 세팅하고, 이 플래그 값을 산술논리장치와 시프터에서 부동소수점 곱셈의 정규화(normalization) 과정에 이용한다.

산술논리장치는 논리곱(AND), 논리합(OR), 배타적 논리합(XOR)의 논리 연산 모듈과 수학 연산 모듈인 덧셈/뺄셈기(adder/subtractor)를 포함하고 있다. 위에 설명한 바와 같이 부동소수점 곱셈의 정규화 과정을 위한 특별한 동작이 정의되어 있다.

시프터는 멀티플렉서(multiplexer) 기반으로 동작한다. 왼쪽으로 혹은 오른쪽으로, 0비트(no-shift)부터 15비트까지 시프트(shift) 동작을 수행할 수 있다. 산술논리장치와 마찬가지로 부동소수점 곱셈의 정규화 과정을 위한 특별한 동작이 정의되어 있다.

다. 레지스터 파일

레지스터 파일에는 16비트 레지스터 32개가 포함된다. 0번째 레지스터는 항상 0값을 저장하고 있는 영 레지스터(zero register)로 고정되어 있다. 레지스터 파일은 입력 포트가 두 개, 출력 포트가 세 개다. 입력 포트는 각각 기능 유닛, 이웃 PE 또는 로컬 메모리로부터 데이터를 받을 수 있다. 출력 포트는 각각 기능 유닛의 입력 A, 기능 유닛의 입력 B, 이웃 PE 또는 로컬 메모리로 데이터를 전송 할 수 있다. 이런 구조는 PE 간의 데이터 통신과 연산을 동시 수행할 수 있도록 한다. 이러한 데이터 통신과 연산의 중첩(overlapping)을 통해 빠른 성능의 알고리듬을 개발할 수 있다.

5. 부동소수점 누산기

부동소수점 덧셈 연산의 경우 많은 연산량을 필요로 하므로 제안된 재구성형 병렬 프로세서에서는 24비트 부동소수점 누산기를 사용한다[3]. 그림 1에 나타난 바와 같이 부동소수점 누산기 24개가 PE 어레이 위쪽에 연결되어 있다. PE 어레이 안에 포함된 PE는 입출력을 16비트 단위로 하기 때문에, 부동소수점 누산기의 입력단과 출력단에는 지연 레지스터(delay register)를 각각 포함하고 있어서, 두 클럭에 걸쳐 데이터를 주고받도록 설계되어 있다.

III. 기능적 특성

1. 부동소수점 연산의 지원

제안된 재구성형 병렬 프로세서에서는 3D 그래픽스 처리 과정의 기하 변환 단계에서 쓰이는 부동소수점 덧셈 연산 및 부동소수점 곱셈 연산을 가속하기 위해 하드웨어 지원을 한다.

부동소수점 덧셈 연산의 경우, 많은 연산량을 필요로 하므로 추가적으로 24개의 부동소수점 누산기를 PE 어레이 위쪽에 배치하여 사용한다.

부동소수점 곱셈 연산의 경우, 기능 유닛 안에 부동소수점 곱셈용 플래그를 두어 부동소수점 곱셈과 관련된 명령어를 실행할 때 이 플래그를 사용하여 부동소수점 곱셈 연산의 정규화 과정을 처리한다.

2. 분할 SIMD

PE 어레이의 PE들은 4그룹으로 나누어지고 각 그룹이 조금씩 다른 동작을 하게 할 수 있다. 이를 분할 SIMD 방식이라고 한다. 그룹을 나누는 방법에는 두 가지 모드가 있다. 그룹을 나누는 방법이 그림 5에 나타나 있다. 하나는 행 색인(index)을 그룹 색인으로 사용하는 방법으로, 0행 PE, 1행 PE, 2행 PE, 3행 PE로 나

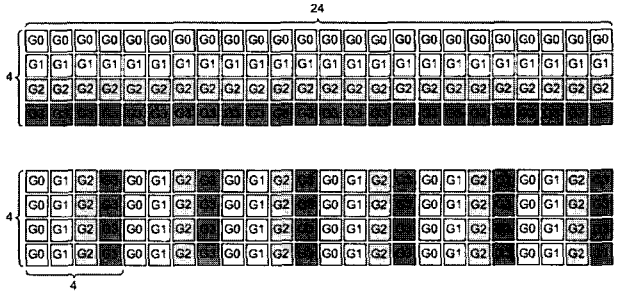


그림 5. 분할 SIMD 방식의 그룹 분할 방법
Fig. 5. Group-seperation method of partitioned SIMD.

누는 방법이다. 또 하나의 방법은 열 단위로 나누되 열 색인에 모듈로 4(modulo 4)한 색인을 그룹 색인으로 사용하는 방법이다. 즉, 0열/4열/8열/... PE가 0그룹이고, 1열/5열/9열/... PE가 1그룹, 2열/6열/10열/... PE가 2그룹, 3열/7열/11열/... PE가 3그룹이 되는 방법이다. 이 때 서로 다른 그룹은 그림 4에 나타난 mux_IN, addr_IN, addr_C 제어 신호를 다르게 가질 수 있다. 이런 방법을 사용하면 각 그룹이 다른 방향의 이웃 PE로부터 데이터를 받을 수 있고(mux_IN), 이웃 PE로부터 받은 데이터 또는 로컬 메모리로부터의 데이터를 레지스터 파일의 서로 다른 레지스터에 저장할 수 있으며(addr_IN), 레지스터 파일의 서로 다른 레지스터에 있는 데이터를 이웃 PE 또는 로컬 메모리로 보낼 수 있다(addr_C).

3. 조건부 실행

제안된 재구성형 병렬 프로세서에서 사용되는 모든 명령어들은 조건부 실행된다. PE에서는 조건부 실행을 위해서 플래그를 사용한다. 이 플래그는 부동소수점 곱셈 연산을 위한 플래그와는 다른 플래그로, 명령어를 실행할 때 항상 참조되는 플래그이다. 플래그에는 다섯 가지 종류가 있다. 플래그의 종류와 역할이 표 1에 나타나 있다.[1] 플래그는 기능 유닛에 포함된 산술논리장치의 출력에 상관없이 값이 변하지 않는 플래그로 특별한 조건이 없이 산술논리장치의 계산 결과를 레지스터 파일에 저장하는 경우에 사용된다. <Z>, <N> 플래그는 매 클럭산술논리장치의 출력이 변할 때마다 그에 따라 업데이트 되는 플래그이다. 반면에 [f_1], [f_2]는 각각 <Z>, <N> 플래그의 값을 유지시킬 필요성이 있을 때 사용된다. 즉 <f_1>, <f_2>는 매 클럭 산술논리장치의 출력이 변할 때마다 업데이트되지 않고 사용자가 업데이트 여부를 결정할 수 있다.

조건부 실행을 사용하면 PE 어레이에 포함된 각 PE들이 참조하는 플래그에 따라 기능 유닛의 연산 결과, 이웃 PE로부터의 데이터 또는 로컬 메모리로부터의 데이터를 레지스터 파일의 레지스터에 저장할지 여부

표 1. 플래그의 종류와 역할
Table 1. Types and roles of flags.

종류	역 할
<1>	항상 1
<Z>	산술논리장치의 출력이 영(Zero)
<N>	산술논리장치의 출력이 음수(Negative)
<f1>	<Z> 플래그를 유지; 사용자가 저장 여부 결정 가능
<f2>	<N> 플래그를 유지; 사용자가 저장 여부 결정 가능

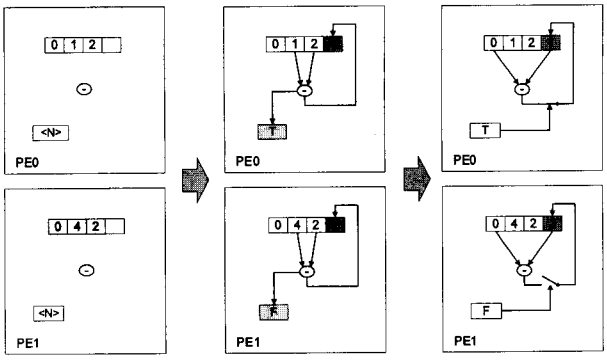


그림 6. 조건부 실행의 예
Fig. 6. An example of conditional execution.

를 결정할 수 있다. 따라서 서로 다른 PE에서 조건에 따라 서로 다른 연산을 수행할 수 있다. 조건부 실행의 간단한 활용 예가 그림 6에 나타나 있다. 그림 6에서 PE0은 1 - 2의 절대값을 구하고, PE1은 4 - 2의 절대값을 구하는 과정을 보이고 있다. PE0에서 결과값 1-2=-1이 레지스터에 저장되고 플래그 <N>이 참(true)이 된다. PE1에서는 결과값 4-2=2가 레지스터에 저장되고, 플래그 <N>이 거짓(false)이 된다. 그 후 두 PE에서 "만일 <N>이 참이면, C := 0 - C" 명령어를 수행하면, <N>이 참인 PE0에서는 0-(-1)=1을 계산하고, <N>이 거짓인 PE1에서는 아무 계산도 하지 않는다. 결과적으로 두 PE에서는 |A - B|를 연산한 것이다.

Ⅳ. 명령어

본 장에서는 제안된 재구성형 병렬 프로세서에서 쓰이는 명령어와 명령어를 사용하는 예에 대해서 설명한다.

1. 명령어 종류

명령어는 크게 산술논리 명령어, 시프트 명령어, 곱셈 명령어로 분류된다. 산술논리 명령어에는 AND, OR, XOR, ADD, SUB, ADDF가 있으며 각각 그 이름에 나타난 연산을 수행할 때 사용된다. ADDF는 부동소수점 곱셈의 정규화 과정에 사용되는 덧셈 명령어이고 이 명령어가 수행될 때는 Ⅱ장에서 설명한 부동소수점 곱셈용 플래그를 참조한다.

시프트 명령어에는 VSHFT_INI, VSHFT_CNT, VSHFT_END, VSHFTF_INI, VSHFTF_CNT, VSHFTF_END가 있으며 시프트 연산을 수행한다. 명령어 이름에 있는 _INI, _CNT, _END 접미어는 시프터의 2단계 파이프라인(2-stage pipeline) 특성에 기인한다. 즉 모든

명령어는 한 클럭 단위로 수행되는 반면 시프트의 연산 결과값은 두 클럭 후에 얻을 수 있기 때문에 _INI 명령어는 첫 번째 파이프라인 단계에서만 연산할 때 사용되고, _CNT 명령어는 첫 번째와 두 번째 파이프라인 단계에서 연산할 때 사용되고, _END 명령어는 두 번째 파이프라인 단계에서만 연산할 때 사용된다. VSHFTF_ 명령어는 부동소수점 곱셈의 정규화 과정에 사용되는 시프트 명령어이고 이 명령어가 수행될 때는 II장에서 설명한 부동소수점 곱셈용 플래그를 참조한다. 곱셈 명령어에는 MUL_INI, MUL_END, MULF_INI, MULF_END가 있으며 곱셈 연산을 수행한다. 명령어가 _INI, _END 두 개로 나누어진 이유는 곱셈의 경우 입력에 비해 출력이 두 배의 비트 크기를 갖기 때문에 곱셈 연산을 완료하려면 두 클럭이 필요하기 때문이다. 즉 곱셈의 입력이 두 개의 16비트 피연산자인 반면 결과값은 32비트이기 때문에 상위 16비트와 하위 16비트를 두 클럭에 나누어 각각 레지스터에 저장해야 하기 때문이다.

2. 부동소수점 곱셈

본 절에서는 제안된 재구성형 병렬 프로세서에서 수행되는 부동소수점 곱셈 과정에 대해서 설명한다.

가. 부동소수점 곱셈 과정

제안되는 재구성형 병렬 프로세서 사용하는 부동소수점 형식은 24비트로서, 부호 1비트, 지수 7비트, 가수 16비트를 포함한다. 이러한 부동소수점 형식을 사용하면 32비트의 IEEE 754 단정도 형식(single precision)에 비해 비용을 적게 사용하면서도 3D 그래픽스 표준인 OpenGL에서 요구하는 10-5 정확도를 만족할 수 있다. 제안되는 재구성형 병렬 프로세서는 16비트 단위로 값을 레지스터에 저장하기 때문에 24비트 부동소수점 형식을 저장하기 위해서 그림 7과 같이 두 개의 16비트 레지스터에 나누어 저장하는 방법을 사용한다. 제안된 재구성형 병렬 프로세서에서의 부동소수점 곱셈 알고리듬이 아래에 나타나 있다.

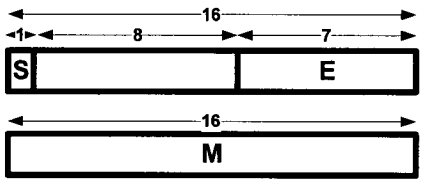


그림 7. 24비트 부동소수점 형식의 저장 방법
Fig. 7. Storing method of a 24-bit floating-point format.

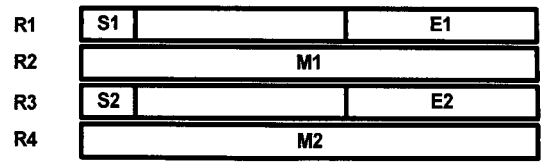


그림 8. PE의 레지스터에 저장된 입력값
Fig. 8. Operands stored in registers of a PE.

부동소수점 곱셈을 위해 두 개의 입력이 PE의 R1~R4 레지스터 위치에 저장되어져 있다고 가정한다. 그림 8에 레지스터에 저장된 입력값이 나타나 있다.

[STEP 1]

R1과 R3를 더하여 R5에 저장한다. 부동소수점 곱셈에서 지수 비트는 더해지고, 부호 비트도 R1과 R3를 더하는 것으로 얻어진다.

[STEP 2]

R2와 R4를 곱하고 결과를 R2와 R3에 저장한다. 제안된 구조에서 사용하는 부동소수점 형식에서 가수가 16비트로 이루어지므로 17비트 곱셈이 이루어져야 한다. 17비트 가수부 곱셈이 이루어졌을 경우, 출력으로 34비트가 출력되어지는데 아래와 같이 세 가지 경우로 나올 수 있다.

01.xxxxxxxxxxxxxxxx
10.xxxxxxxxxxxxxxxx
11.xxxxxxxxxxxxxxxx

곱셈 결과는 최상위 비트에 따라 정규화 과정이 수행되야 한다. 이를 처리하기 위해서 부동소수점 곱셈용 플래그에 가수부 곱셈 결과의 상위 두 비트가 저장된다.

[STEP 3]

플래그 상위 비트 [FH]이 1이면 R2를 그림 9와 같이 오른쪽으로 1비트 시프트한다. [FH]가 0이면 R2를 그대로 둔다.

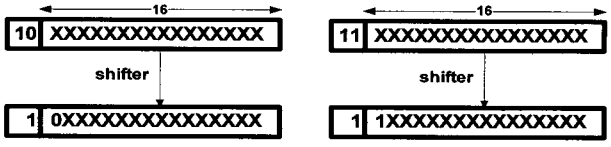


그림 9. 부동소수점 플래그를 참조한 시프트 연산
Fig. 9. Shift operation with floating-point flags.

[STEP 4]

[FH]이 1이면 가수 부분이 1비트 오른쪽 시프트되었

으므로 지수 부분인 R5에 1을 더하고 R5에 저장한다. 만일 [FH]이 0이면 그대로 둔다.

[STEP 5]

[STEP 1]에서 바이어스(bias)가 포함된 두 값이 더해졌으므로 R5에서 바이어스 63을 빼고 R1에 저장한다.

[STEP 6]

만일 지수부가 음수면 언더플로우(underflow)가 발생된 경우이므로, 이를 확인하기 위해서 R1과 0x4000 을 논리합하고 [f_1] 플래그를 저장한다.

[STEP 7]

[f_1]가 0일 경우, 즉 지수부가 음수일 경우 R1, R2 레지스터에 예외(exception) 처리를 한다.

[STEP 8]

만일 지수부가 7비트를 넘으면 오버플로우(overflow)가 발생된 경우이므로, 이를 확인하기 위해서 먼저 R1의 값과 0x7FFF 값을 논리합 함으로써 부호 비트를 제거한다. 그 후에 0x007E 값을 빼고 [f_2] 플래그를 저장한다.

[STEP 9]

[f_2]가 0일 경우, 즉 오버플로우가 발생된 경우 R1, R2 레지스터에 예외 처리를 한다.

나. 부동소수점 곱셈에 대한 명령어 예제

부동소수점 곱셈 과정을 명령어로 표현하면 아래와 같다. 아래에 나타난 바와 같이 부동소수점 곱셈 연산을 수행하는 데에 총 15 클럭이 소요된다.

```
ADD R5, R1, R3          // R1 + R3 → R5
MULF_INI R2, R2, R4     // R2 * R4 → R2
MULF_END R3, R2, R4     // R2 * R4 → R3
VSHFTF_INI R2, R2       // if [FH]=1,R2 》1→R2
VSHFTF_END R2, R2
ADDF R5, R5             // if [FH]=1,R5 + 1→R5
SUB R1, R5, 63          // R5 - 63 → R1
AND R3, R1, 0x4000 // R1 & 0x4000 → R3 <f1>
AND R1, R5, 0x8000 // if <f1>=0,R5 & 0x8000→R1
AND R2, R2, 0x0000 // if <f1>=0,R2 & 0x0000→R2
AND R3, R1, 0x7FFF // R1 & 0x7FFF → R3
SUB R3, R3, 0x007E  // R3 - 0x007E → R3 <f2>
AND R3, R1, 0x8000 // if <f2>=0,R1 & 0x8000→R3
OR R1, R3, 0x007E  // if <f2>=0,R3 | 0x007E→R1
OR R2, R2, 0xFFFF // if <f2>=0,R2 | 0xFFFF→R2
```

V. 멀티미디어 병렬 알고리듬

본 장에서는 제안된 재구성형 병렬 프로세서를 기반으로 개발된 멀티미디어 알고리듬에 대해 설명한다.

1. 움직임 추정(motion estimation)

움직임 추정은 H.264 표준에서 가장 계산량이 많은 연산이다[4~5]. 제안된 재구성형 병렬 프로세서에서는 전역 탐색 블록 정합(Full Search Blocking Matching) 움직임 추정을 위한 SAD(Sum of Absolute Differences) 계산 알고리듬을 수행한다. 수행된 알고리듬에서는 동시에 세 개의 후보 블록(candidate blocks)에 대해 16x16, 16x8, 8x16, 8x8, 8x4, 4x8, 4x4의 가변 블록 크기(variable block size)의 SAD 연산을 지원한다. 제안된 재구성형 병렬 프로세서에서 수행된 알고리듬은 표 2에 나타난 성능 목표(target performance)를 만족시킨다.

2. 2D 정수 변환(integer transform)

4x4 2D 정방향/역방향 정수 변환은 H.264 표준에서 기본적으로 사용되는 압축 기술이다[4~6]. 수행된 정수 변환 알고리듬에서는 각각의 4x4 입력 블록이 4x4 2D PE 행렬의 16개의 PE에 매핑(mapping)되기 때문에 4x24 PE 어레이에서 동시에 6개의 정수 변환이 수행된다. 하나의 입력 블록에 대한 정수 변환은 8 클럭에 수행된다.

3. 2D 이산 여현 변환(DCT)

H.263과 MPEG-4, JPEG 등의 압축 표준에서 자주 쓰이는 8x8 2D 정방향/역방향 이산 여현 변환 알고리듬이 수행된다[5, 7~8]. 정수 변환의 경우와 비슷하게 6개의 이산 여현 변환이 PE 어레이에서 동시에 수행된다. 하나의 8x8 입력 블록에 대한 이산 여현 변환은 43 클럭에 수행된다.

4. 고속 푸리에 변환(FFT)

1024포인트 고속 푸리에 변환은 MP3에서 기본적으로 사용되는 기술이다[9]. 제안된 재구성형 병렬 프로세서에서는 재귀적 나비 연산(recursive butterfly operation)을 이용한 1024포인트 4진(radix-4) 고속 푸

표 2. 움직임 추정 알고리듬의 목표 성능
Table 2. Target performance of the ME algorithm.

화면 형식	CIF (352x288)
탐색 범위	[-8, +8]
블록 크기	7가지의 가변 블록 크기
동작 주파수	100 MHz
초당 프레임 수	30

리에 변환 알고리듬이 수행된다. 1024포인트 4진 고속 푸리에 변환 알고리듬은 약 950 클럭에 수행된다.

5. 이산 웨이블렛 변환(DWT)

JPEG2000 표준은 전통적인 JPEG 베이스라인 프로파일(baseline profile)에 비해 비트량, 품질면에서 우수하다. 이산 웨이블렛 변환은 푸리에 변환과 같이 공간 영역(spatial domain)의 신호를 시간 영역(time domain)으로 변환하는 연산을 한다. 하지만 푸리에 변환과 달리 이미지에서 임의의 영역을 효과적으로 분석하는 것이 가능하다. 1D 이산 웨이블렛 변환에는 3/5 필터와 7/9 필터가 있다[10]. 제안된 재구성형 병렬 프로세서에서는 3/5 필터 알고리듬이 수행된다. 2D 이산 웨이블렛 변환은 1D 이산 웨이블렛 변환을 수평 방향과 수직 방향으로 수행함으로써 연산된다. 8개의 픽셀이 4x1 PE 행렬에 매핑되고 24개의 PE의 열이 동시에 연산을 수행하는 데에 걸리는 시간은 25 클럭이다. 따라서 8개의 픽셀의 1D 이산 웨이블렛 변환은 2 클럭에 수행된다.

6. 디블록킹 필터(deblocking filter)

H.264 영상 압축 표준에서 정수 변환을 수행한 블록 가장자리에는 왜곡이 발생한다. 이러한 블록 왜곡을 완화하기 위해서 디블록킹 필터 알고리듬이 수행된다[4~5]. 제안된 재구성형 병렬 프로세서에서 수행된 16x16 크기의 매크로블록에 대한 디블록킹 필터 알고리듬은 약 200 클럭에 수행된다.

7. 3D 그래픽스

3D 그래픽스의 기하 단계(geometry stage)에 포함된 변환 과정(transformation process)이 수행된다. 변환 과정을 수행하기 위해서는 4x4 변환 행렬과 4x1 입력 행렬의 곱에 대한 연산이 필요하다. 이 행렬들을 구성하는 성분은 부동소수점 형식이기 때문에 부동소수점 곱셈 및 덧셈 연산을 수행해야 한다[1]. 이런 연산은 연산량이 많기 때문에 일반적인 정수 연산기에서는 수행 시간이 오래 걸린다. 따라서 제안된 재구성형 병렬 프로세서에서는 Ⅲ장에 설명한 바와 같이 부동소수점 곱셈 및 덧셈을 가속하기 위해 하드웨어를 지원한다. 수행된 알고리듬에서는 4x1 입력 행렬 네 개를 모아서 4x4 행렬로 만들어 연산을 한다. 네 개의 4x1 입력 행렬에 대한 변환 과정은 약 100 클럭에 수행된다.

Ⅵ. 구조 및 알고리듬 성능 비교

1. 재구성형 병렬 프로세서 구조 비교

기존의 재구성형 병렬 프로세서와의 구조 비교가 표 3에 나타나 있다. 표 3에 나타나는 바와 같이 제안된 구조에서는 이웃한 PE끼리만 연결하는 망을 사용하고 다른 구조에서는 서로 멀리 떨어진 PE끼리 연결시키는 망을 추가적으로 사용하기 때문에, 제안된 구조에서 내부 연결망 면에서 상대적으로 적은 비용을 사용한다. 그물형 망은 유연성이 떨어지기 때문에 알고리듬 개발 시 어려움이 있을 수 있다. 제안된 재구성형 병렬 프로세서에서는 이를 해결하기 위해 2장과 3장에서 설명하는 바와 같이 데이터 통신과 연산의 중첩, 분할 SIMD, 조건부 실행 등의 방법을 지원한다.

2. 알고리듬 성능 비교

1024포인트 고속 푸리에 변환의 경우 [15]는 2613 클럭의 성능을 보이고, 제안된 재구성형 병렬 프로세서에서는 958 클럭의 성능을 보였다. 이런 차이를 보인 이유는 [15]에서는 2진(radix-2) 방식을 사용하고 제안된 구조에서는 4진 방식을 사용함으로써 연산 단계의 수를 줄였기 때문이다.

8x8 이산 여현 변환의 경우 [16]에서 23 클럭의 성능

표 3. 기존의 재구성형 병렬 프로세서와의 구조 비교
Table 3. Architecture comparison with previously researched reconfigurable parallel processors.

	[11]	[12]	[13]	[14]	제안된 구조
PE 배열 크기	8 x 8	24 x 32	16 x 16	8 x 8	4 x 24
PE 연결 구조	3단계 망	행단위/열단위/전역 망	행단위/전역 망	2단계 망 + 조건 분기용 망	그물형 망(1단계)
기본 연산 단위	16 비트	2 비트	8 비트	8 비트	16 비트

을 보이고 [17]에서 3736 클럭의 성능을 보였으며 제안된 구조에서 43 클럭의 성능을 보였다. 이런 차이가 발생한 이유는 [17]에서 곱셈기를 사용하지 않고 DA (Distributed Arithmetic)를 사용했고, 제안된 구조에서 곱셈 연산 및 로컬 메모리 접근에 각각 2 클럭이 소요되기 때문이다.

Ⅶ. 함수적 검증 및 합성 결과

1. 함수적 검증

정수 변환과 부동소수점 곱셈 알고리듬에 대한 명령어를 작성하여 테스트함으로써 함수적 검증이 이루어졌다. 정수 변환의 경우 H.264 표준의 참조 소프트웨어인 JM11[18]의 입출력과 비교하여 동일한 값이 얻어짐을 확인하였다. 부동소수점 곱셈의 경우 C언어로 부동소수점 곱셈을 수행한 입출력과 비교하여 OpenGL 표준에서 요구하는 10^{-5}의 정확도(precision)를 만족하는 것을 확인하였다.

2. 합성 결과

제안된 재구성형 병렬 프로세서는 VHDL로 설계되었다. 설계된 구조는 100MHz의 동작 주파수로 합성되었고, 합성 결과 부동소수점 누산기는 3,867 게이트카운트(gate count), PE는 14,902 게이트카운트로 산출되었다. 따라서 24개의 부동소수점 누산기와 4x24의 PE 어레이에 대한 게이트카운트는 약 150만으로 산출된다.

Ⅷ. 결론 및 향후 연구 과제

본 논문에서는 모바일 멀티미디어의 효율적 처리를 위한 재구성형 병렬 프로세서의 구조를 제안하였다. 제안된 시스템은 PE 어레이, 로컬 메모리, 제어 유닛, 그리고 부동소수점 누산기 24개로 구성되어 있다. PE 어레이는 4x24의 PE들이 그물형 망을 구성하고 있는 구조를 가지고 있다. PE 어레이의 0행 PE들은 부동소수점 누산기와 각각 연결되어 있다. 로컬 메모리는 PE와 직접 연결되어 있기 때문에 적은 전력 소모로 PE와 로컬 메모리의 데이터 통신이 가능하다. 제안된 재구성형 병렬 프로세서에서는 분할 SIMD와 조건부 실행 방식을 사용하기 때문에 적은 하드웨어 비용으로 효과적인 알고리듬 개발이 가능하다. 부동소수점 누산기 및 부동소수점 곱셈용 플래그를 사용함으로써 3D 그래픽스의 기하 변환에서 쓰이는 부동소수점 덧셈 및 곱셈을 효율적으로 지원한다.

향후 연구 과제로는 제안된 재구성형 병렬 프로세서의 멀티미디어 처리에 대한 기능성을 높이고, 개발된 알고리듬의 성능을 향상시키는 연구가 진행될 예정이다.

참 고 문 헌

[1] Tomas A. M., Eric H., *Real-Time Rendering*, A K PETERS, pp. 9-66, 2002.

[2] William J. D., Brian T., *Principles and Practices of Interconnection Networks, Morgan Kaufmann*, pp. 89-106, 2003.

[3] Israel K., *Computer Arithmetic Algorithms*, A K PETERS, pp. 59-89, 2001.

[4] G. Sullivan, A. Luthra, and T. Wiegand, "DRAFT OF VERSION 4 OF ISO/IEC 14496-10 (E)," in Joint Video Team(JVT) of ISO/IEC MPEG and ITU-T VCEG, April 2005.

[5] I. Richardson, *H.264 and MPEG-4, VIDEO COMPRESSION*, John Wiley&Sons, 2002.

[6] A. Hallapuro, M. Karczewicz, and H. Malvar, "Low Complexity transform and quantization," in Joint Video Team(JVT) of ISO/IEC MPEG and ITU-T VCEG, Docs. JVT-B038 and JVT-B039, Jan. 2002.

[7] ISO/IEC, Digital compression and coding of continuous tone still images: requirements and quidelines, International Standard Organization, 1994.

[8] ITU Telecom. Standardization Sector of ITU, "Video coding for low bitrate communication," ITU-T Recommendation H.263, Mar. 1996.

[9] Karlheinz B., "MP3 and AAC explained," AES 17th conference, 1999.

[10] JPEG-2000 Part 1 Final Draft International Standard, "ISO/IEC JTC1/SC29 WG1 N189R," Aug. 2000.

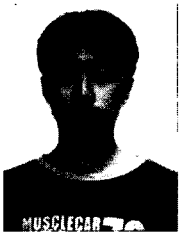
[11] H. Parizi, A. Niktash, A. Kamalizad, N. Bagherzadeh, "A Reconfigurable Architecture for Wireless Communication Systems," *Information Technology: New Generations*, pp. 250-255, Las Vegas, USA, April 2006.

[12] J. R. Hauster, J. Wawrzynek J., "Garp: a MIPS processor with a reconfigurable coprocessor," *FPGAs for Custom Computing Machines*, pp. 12-21, Napa Valley, USA, April 1997.

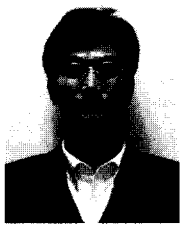
[13] H. Schmit, D. Whelihan, A. Tsai, M. Moe, B. Levine, R. Reed Taylor, "PipeRench: A virtualized programmable datapath in 0.18 micron technology," *Custom Integrated Circuits*

Conference, pp. 63-66, Orlando, USA, May 2002.
[14] 김윤진, 정진용, 강신원, 최기영, "재구성형 프로세싱 모듈의 설계," *대한전자공학회 학술회의*, pp. 312-317, 2004년 5월.
[15] A. H. Kamalizad, C. Pan, N. Bagherzadeh, "Fast parallel FFT on a reconfigurable computation platform," *Computer Architecture and High Performance Computing*, pp. 254-259, Sao Paulo, Brazil, Nov. 2003.
[16] H. Parizi, A. Niktash, N. Bagherzadeh, F. Kurdahi, "MorphoSys: A coarse grain reconfigurable architecture for multimedia applications," *8th International Euro-Par Conference*, pp. 844-848, Paderborn, Germany, Aug. 2002.
[17] 김남섭, 이상훈, 금민하, 김진상, 조원경, "멀티미디어 무선 단말기를 위한 재구성 가능한 코프로세서의 설계," *전자공학회 논문지 SD편*, 제44권, 제4호, 63~72쪽, 2007년 4월.
[18] JVT H.264/AVC Joint Model Reference Software version 11.0, http://iphome.hhi.de/suehring/tml/download/old_jm/jm11.0.zip, Aug. 2007.

저 자 소 개



유 세 훈(학생회원)
2006년 서울시립대학교 전자전기컴퓨터공학부 학사
2006년~현재 서울시립대학교 전자전기컴퓨터공학부 석사 과정
<주관심분야 : SoC설계, 멀티미디어, 병렬처리>



김 기 철(정회원)
1982년 서울대학교 전기공학과 학사
1984년 서울대학교 전기공학과 석사
1991년 University of Southern California 전기공학과 (컴퓨터공학 전공) 박사
1984년~1994년 한국전자통신연구원 선임연구원
1994년~현재 서울시립대학교 전자전기컴퓨터공학부 교수
<주관심분야 : SoC설계, 멀티미디어, 컴퓨터구조, 병렬처리>

양 일 석(정회원)
1989년 경북대학교 전자공학과 학사
1994년 경북대학교 전자공학과 석사
1999년~현재 한국전자통신연구원 선임연구원
<주관심분야 : 통신, 컴퓨터, 신호처리, 반도체>

노 태 문(정회원)
1984년 경북대학교 전자공학과 학사
1986년 경북대학교 전자공학과 석사
1998년 경북대학교 전자공학과 박사
1988년~현재 한국전자통신연구원 책임연구원
<주관심분야 : 신호처리, 반도체, SoC>