논문 2007-44SD-10-8

패킷 스케줄러를 위한 빠르고 확장성 있는 우선순위 큐의 하드웨어 구조

(A Fast and Scalable Priority Queue Hardware Architecture for Packet Schedulers)

김 상 균*, 문 병 인**

(Sanggyun Kim and Byung In Moon)

요 약

본 논문에서는 QoS를 보장하면서 빠른 네트워크 속도를 지원해 줄 수 있는 우선순위 큐의 구조를 제안한다. 제안한 큐의 구조는 하나의 큐로 여러 개의 출력부에 출력을 보낼 수 있어 면적을 줄일 수 있고, 제어 블록을 추가함으로써 기존의 multiple systolic array 우선순위 큐보다 더 빠른 속도로 동작할 수 있기 때문에 높은 패킷 처리 속도를 요구하는 패킷 스케줄러 등에 적합한 구조이다. 또한, 이 구조는 높은 확장성을 지원한다.

Abstract

This paper proposes a fast and scalable priority queue architecture for use in high-speed networks which supports quality of service (QoS) guarantees. This architecture is cost-effective since a single queue can generate outputs to multiple out-links. Also, compared with the previous multiple systolic array priority queues, the proposed queue provides fast output generation, which is important to high-speed packet schedulers, using a special control block. In addition, this architecture provides the feature of high scalability.

Keywords: Quality of Service, Priority Queue, Packet Scheduler

I. 서 론

요즘 많이 이용되고 있는 실시간 동영상이나 실시간 음악 서비스 등의 real-time 통신에서는 기존의 웹서핑과 같은 일반적인 통신보다 높은 속도와 QoS(Quality of Service)의 보장을 요구한다. QoS란 사용자 또는 어플리케이션의 중요도에 따라 서비스 수준을 차등화 하여 한정된 대역폭에서 트래픽과 대역폭을 정책적으로 관리하는 것을 말한다. 즉, 끊김이 없는 동영상을 보거나 음악을 듣기 위해서는 QoS가 보장되어야 한다.

이전의 네트워크 장비들은 QoS를 보장하기 위해 소프트웨어적인 기술들을 사용하였으나 근래의 높아진 네트워크의 속도에 맞춰 QoS를 보장하기 위해서는 하드웨어적인 기술이 필요하다.

QoS를 보장하기 위한 하드웨어 구조로 각 데이터에 알맞은 우선순위를 부여하고 높은 우선순위 순으로 데이터를 처리하는 방법이 가장 많이 사용되고 있다. 예를 들면, binary tree based of comparator 우선순위 큐[4], shift register 우선순위 큐[1], systolic array 우선순위 큐[2~3] 등이 있다. 하지만 이런 방법들은 결점이 존재한다. binary tree based of comparator 우선순위 큐의 경우 큐의 용량이 늘어남에 따라 출력이 나오는 시간이 지연된다. shift register 우선순위 큐와 systolic

* 학생회원, ** 평생회원, 경북대학교 전자전기컴퓨터학부
(School of Electrical Engineering & Computer Science, Kyungpook National University)

※ 이 논문은 2005년도 경북대학교 학술진흥연구비에 의하여 연구되었음.

접수일자: 2007년5월26일, 수정완료일: 2007년9월21일

array 우선순위 큐는 확장성은 만족하지만 여러 개의 출력부를 사용할 때는 면적이 커지는 단점이 있다.

본론에서 소개할 modified multiple systolic array 우선순위 큐는 위에서 말한 우선순위 큐의 단점들을 보안한 구조로 여러 개의 출력부를 가지고 높은 속도를 요구하는 패킷 스케줄러에 알맞은 우선순위 큐이다.

II. 본 론

1. 여러 우선순위 큐의 구조들

현재 사용되고 있는 여러 우선순위 큐의 구조들에 대해 알아보고 각각의 장단점을 파악해보자.

가. Shift Register 우선순위 큐

Shift register 우선순위 큐는 그림 1과 같은 구조로 각 블록들이 바로 옆(좌, 우) 블록들과 서로 연결되어 있다.[1] 입력된 데이터는 버스를 통해 모든 블록에 동시에 입력되고 우선순위를 비교하여 한 블록만 데이터를 저장하고 나머지 블록의 데이터들은 오른쪽 또는 왼쪽으로 순차적으로 이동하게 된다. 가장 오른쪽 블록에 가장 높은 우선순위를 가지는 데이터가 항상 존재하므로 읽기 신호가 들어오면 가장 오른쪽 블록의 데이터가 출력된다. 하지만 입력되는 데이터가 모든 블록에 동시에 입력되어야 하기 때문에 블록들이 많아지면 버스 로딩 문제가 발생하게 된다.

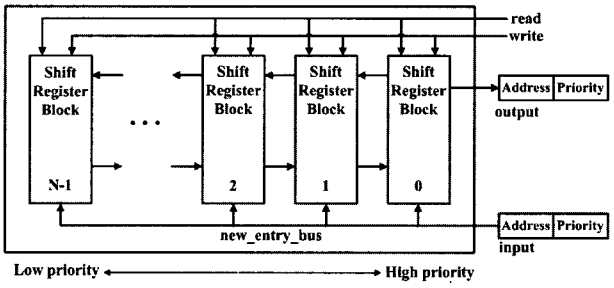

그림 1. Shift register 우선순위 큐의 구조
Fig. 1. Structure of the shift register priority queue.

나. Systolic Array 우선순위 큐

Systolic array 우선순위 큐는 그림 2와 같이 shift register 우선순위 큐와 비슷한 구조를 가지고 있다.[2~3] 하지만 모든 블록에 동시에 입력되는 것이 아니라 가장 오른쪽 블록이 입력을 받고 우선순위에 따라 데이터가 왼쪽으로 이동한다. shift register 우선순위 큐와 같이 가장 오른쪽 블록이 가장 높은 우선순위의 데이터를 가지게 된다. 하지만 각 블록의 내부를 보면 저장하는 레지스터와 우선순위 비교 후 왼쪽 블록으로 보낼 레지스터 두 개를 가지므로 shift register 우선순위 큐보다는 많은 면적을 차지한다.

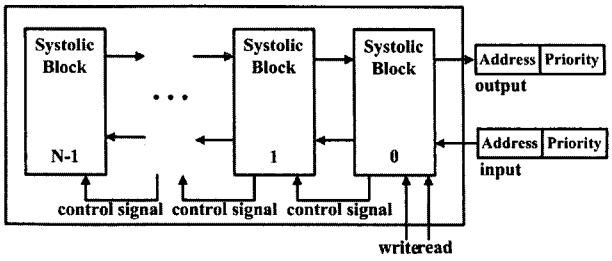

그림 2. Systolic array 우선순위 큐의 구조
Fig. 2. Structure of the systolic array priority queue.

다. Modified Systolic Array 우선순위 큐

위에서 설명한 두 우선순위 큐의 장점을 모아서 modified systolic array 우선순위 큐를 구성하였다.[5] 확장성을 위해 그림 3과 같이 여러 개의 modified systolic array 우선순위 큐 블록이 systolic array와 같은 형태로 연결되어 있고 각 블록의 내부는 shift register와 같은 구조를 가지고 있다. shift register 내부에 연결되어 있는 블록들은 버스 로딩 문제를 일으키지 않을 만큼의 수만큼 만 연결한다.

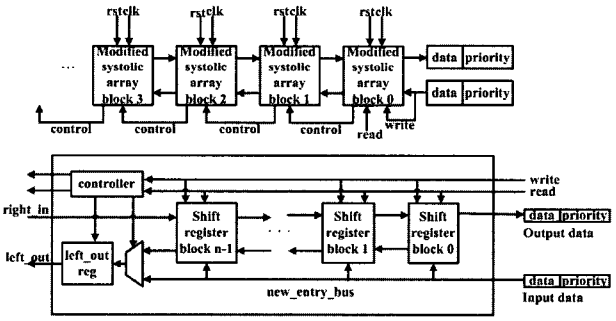

그림 3. Modified systolic array 우선순위 큐의 구조
Fig. 3. Structure of the modified systolic array priority queue.

라. Multiple Systolic Array 우선순위 큐

위에서 설명한 modified systolic array 우선순위 큐를 이용하여 여러 개의 출력부를 하나의 큐로 대체할 수 있는 multiple systolic array 우선순위 큐를 그림 4와 같이 구성하였다.[5] 기존의 여러 큐 구조에서는 각각의 출력부에 큐가 하나씩 있어야 했으나 이 구조를 이용하면 하나의 큐로 여러 개의 출력부로 출력을 보낼 수 있으므로 면적이 절약이 된다. 기본적인 구조는

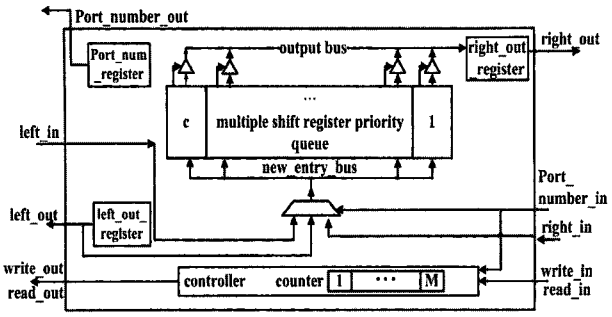

그림 4. Multiple systolic array 우선순위 큐의 구조
Fig. 4. Structure of the multiple systolic array priority queue.

modified systolic array 우선순위 큐와 비슷하지만 여러 출력부에 데이터를 보낼 수 있도록 하기 위해 몇 가지 블록들이 추가되어 있다.

2. Modified Multiple Systolic Array 우선순위 큐

기존의 multiple systolic array 우선순위 큐는 각 출력부에 대한 입력과 출력에 여러 사이클이 소요된다. 데이터가 입력되면 한 블록에만 저장되고 나머지 블록의 데이터들은 순차적으로 이동하기 때문에 데이터가 정렬될 때 까지 기다린 뒤 출력하거나, 이동 중인 데이터를 제외하고 저장되어 있는 데이터 중 가장 높은 우선순위의 데이터를 출력하게 된다. 즉, 출력하는데 시간이 많이 걸리거나 잘못된 데이터가 출력 될 수도 있다. 우리는 제어를 위한 블록을 추가하여 이 문제를 해결할 수 있는 새로운 구조를 제안한다.

가. 제안하는 구조

그림 5와 같이 내부 각 블록들이 서로 연결되어 있는 구조는 multiple systolic array 우선순위 큐와 같지만 제어블록이 추가 되어 있다. 제어블록은 좀 더 정확하고 빠른 속도로 입출력을 하기 위해 추가된 블록으로 각 shift register 블록들에 연결되어 있다.

확장성을 만족시켜주기 위해 그림5와 같이 systolic array의 형태로 여러 개의 modified multiple systolic array 우선순위 큐를 연결한다.

나. 동작

각 블록들은 제어블록의 신호에 따라 read, write, shift_right, shift_left의 네 가지 동작을 한다.

기존의 multiple systolic array 우선순위 큐는 shift register의 각 블록에 동시에 데이터가 입력되어 각 블록에서 우선순위를 판단 한 뒤 저장하거나 오른쪽 또는 왼쪽으로 데이터를 보낸다. 하지만 modified multiple systolic array 우선순위 큐는 제어블록이 어느 블록에 데이터를 쓸 것인가와 어느 블록으로부터 읽을 것인가를 판단하여 각 블록에 신호를 보낸다. 순차적으로 데이터가 이동 되는 것이 아니라 한꺼번에 이동되므로 빠르게 입출력이 가능하다.

(1) 입력

데이터가 입력되면 제어블록에서 데이터가 저장될 블록을 파악하여 해당 블록에 write 신호를 보내고 그 왼쪽 블록들 중 출력부 번호가 다른 블록에는 동시에 shift_left 신호를 보내 저장되어 있던 데이터를 한 블록씩 왼쪽으로 이동시킨다.

그림 6의 (b)를 보면, (a)의 정렬된 상태에서 출력부 번호 1을 가진 data가 입력이 된다. data가 입력될 블록은 write 신호가 입력이 되고, 나머지 왼쪽 블록들에게는 shift left 신호가 입력이 되어 1 cycle 후에는 그림과 같이 정렬이 된다.

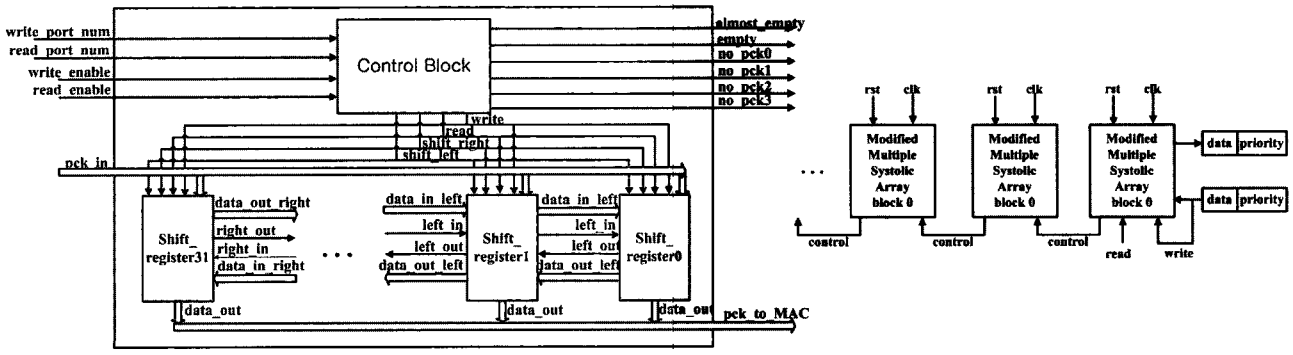

그림 5. Modified multiple systolic array우선순위 큐의 구조
Fig. 5. Structure of the modified multiple systolic array priority queue.

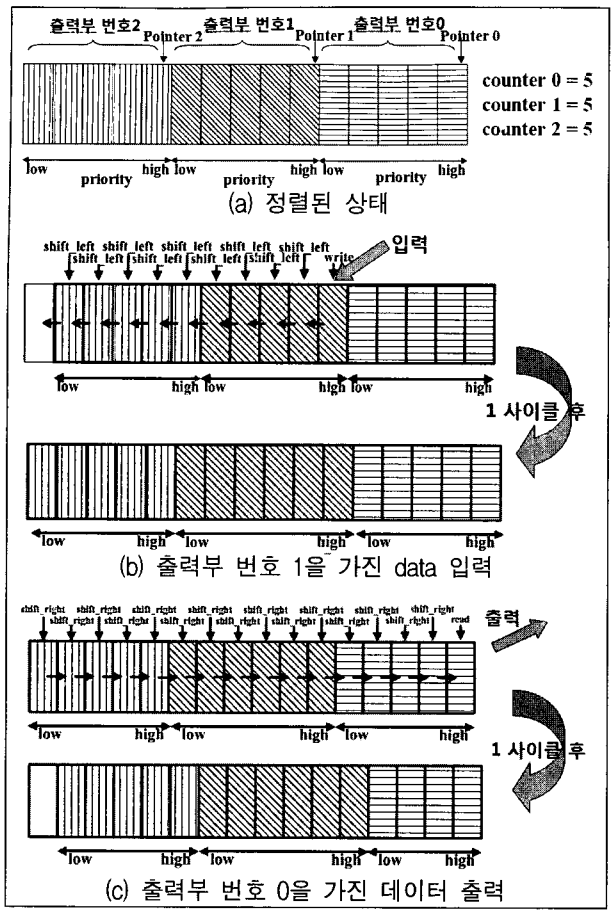

그림 6. 입출력 동작 예시
Fig. 6. Examples of input and output operations.

(2) 출력

출력신호가 들어오면 해당 출력부 번호를 가지는 데이터들 중 가장 순위가 높은 블록이 read 신호를 받아 데이터를 출력시키고 나머지 왼쪽 블록들은 shift_right 신호를 받아 데이터를 한 블록씩 오른쪽으로 이동시킨다. 그림 6의 (c)를 보면, 출력부 0을 가진 data의 출력을 요청하는 신호가 입력되면 가장 오른쪽 블록에 read 신호가 입력되고 나머지 왼쪽의 블록들은 shift right 신호를 입력받아 1 cycle 후 정렬이 된다.

다. 제어블록

제어블록은 그림 7과 같이 포인터, 카운터 그리고 시그널 컨트롤러로 구성되어 있다. 카운터와 포인터는 출력부의 수만큼 존재한다.

카운터는 저장되어 있는 데이터의 수를 헤아리고 있다가 데이터가 없을 경우, 연결되어있는 modified multiple systolic array 블록에 해당 출력부의 데이터를 요구한다. 포인터는 저장되어있는 데이터 중 가장 우선순위가 높은 데이터가 저장되어 있는 블록을 가리킨다.

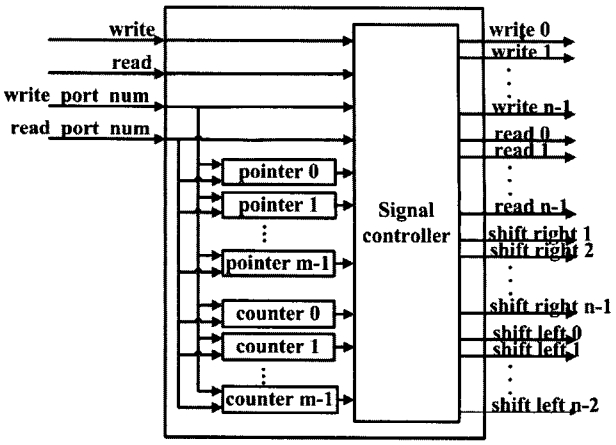

그림 7. 제어블록의 구조
Fig. 7. Structure of the control block.

출력부의 개수만큼 포인터도 존재하므로 각 출력부의 가장 높은 순위의 데이터를 가진 블록 위치를 가리키고 있다. 이것은 출력 시에 이용된다. 시그널 컨트롤러는 각 포인터와 카운터의 값, 입력으로 들어오는 write, read 신호를 받아서 어느 블록에 shift_right, shift_left, write, read 신호를 보낼 지 결정하는 역할을 한다.

III. 실 험

본 논문에서 제안한 modified multiple systolic array 우선순위 큐는 RTL수준으로 구현하여 시뮬레이션 및 합성하였다.

기존의 우선순위 큐들과의 크기를 비교하기 위하여, 앞에서 소개한 shift register 우선순위 큐, systolic array 우선순위 큐, modified systolic array 우선순위 큐, multiple systolic array 우선순위 큐의 저장소 크기를 32개로 정하고 합성 한 후, modified multiple systolic array 우선순위 큐와 transistor의 수를 비교하였다. 그림8과 같이 출력부가 한 개 일 때는 multiple systolic array 우선순위 큐와 modified multiple systolic array 우선순위 큐의 면적이 크지만 출력부의 수가 많아질수록 shift register 우선순위 큐, systolic array 우선순위 큐 그리고 modified systolic 우선순위 큐는 출력부의 수에 비례하여 면적이 넓어지는 반면 multiple systolic array 우선순위 큐와 modified multiple systolic array 우선순위 큐는 출력부의 수와 상관없이 일정한 크기를 가진다. 대부분의 패킷 스케줄러는 1개 이상의 출력부를 가지기 때문에 multiple systolic array 우선순위 큐와 modified multiple systolic array 우선순위 큐는 기존의 다른 우선순위 큐

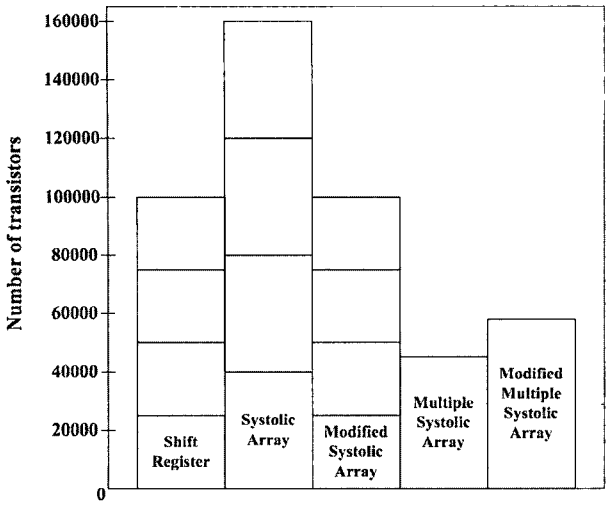

그림 8. Modified multiple systolic array 우선순위 큐와 기존의 큐들의 transistor 수의 비교

Fig. 8. Comparison of the number of transistors used in modified multiple systolic array priority queue and the previous priority queues.

표 1. 출력부의 수를 다르게 했을 경우, 입출력에 걸리 는 cycle 수 비교

Table 1. Comparison of input/output cycle counts (for a single output links and four output links) by changing out-link numbers.

출력부 개수 / 큐의 구조	1개 (입력/출력)	4개 (입력/출력)
multiple systolic array 우선순위 큐	5 / 7 cycle[5]	5 / 7 cycle[5]
modified multiple systolic array 우선순위 큐	1 / 1 cycle	1 / 1 cycle

보다 면적에서 효율적인 구조이다.

Multiple systolic array 우선순위 큐와 modified multiple systolic array 우선순위 큐의 성능을 비교하기 위해서 출력부가 1개 일 때와 4개 일 경우를 구분하여 시뮬레이션 하였다. 아래의 표 1은 하나의 modified multiple systolic array 우선순위 큐 블록만을 사용하였을 때의 경우이다. 만약, 확장을 위해 modified multiple systolic array 우선순위 큐 블록을 더 연결한다면 입력과 출력에 걸리는 cycle은 연결한 블록의 수만큼 늘어나게 된다.

Multiple systolic array 우선순위 큐와 modified multiple systolic array 우선순위 큐를 비교 했을 때, 표 1과 같이 출력부의 수가 달라져도 입출력 cycle은 변함이 없는 것은 동일했지만, modified multiple systolic array 우선순위 큐가 입출력 모두에 1 cycle만 소요됨으로써 더 빠른 속도로 동작함을 확인 할 수 있었다. 이 결과는 각각의 블록이 데이터의 우선순위를 판단하여 정렬하는 multiple systolic array 우선순위 큐보다 제어블록을 이용하여 데이터를 입출력하는 modified multiple systolic array 우선순위 큐가 좀 더 빠른 성능을 나타낸다고 할 수 있다.

또한 ISE를 이용하여 합성한 결과 제어장치는 전체 modified multiple systolic array 우선순위 큐 면적의 5% 만 차지함을 알 수 있었다.

Ⅳ. 결 론

본 논문은 패킷 스케줄러를 위한 빠르고 확장성 있는 우선순위 큐의 하드웨어 구조를 제안하였다. 출력부가 여러 개 일 경우 기존의 여러 우선순위 큐보다 적은 면적으로 높은 성능을 낼 수 있었으며, multiple systolic array 우선순위 큐의 단점을 보완하기 위하여 제어블록을 추가적으로 두어, 좀 더 빠르고 정확한 동작을 할 수 있도록 하였다. 실험을 통해 적은 면적의 제어블록을 추가한 modified multiple systolic array 우선순위 큐가 multiple systolic array 우선순위 큐보다 빠르게 동작함을 확인 할 수 있었다.

참 고 문 헌

[1] J. Chao, "A Novel Architecture for Queue Management in the ATM network," IEEE J. Selected Areas in Comm. vol. 9, no. 7, pp. 1,110-1,118, Sept. 1991.

[2] P. Lavoie and Y. Savaria, "A Systolic Architecture for Fast Stack Sequential Decoders," IEEE Trans. Comm, vol. 42, nos. 2/3/4, pp. 324-334, Feb./Mar/Apr. 1994.

[3] C.E. Leiserson, "Systolic Priority Queues," Proc. Caltech Conf. VLSI, pp. 200-214, Jan. 1979.

[4] D. Picker and R. Fellman, "A VLSI Priority Packet Queue with Inheritance and Overwrite," IEEE Trans. Very Large Scale Integration Systems, vol. 3 no. 2, pp. 245-252, June. 1995.

[5] S. Moon, J. Rexford, and K. Shin, "Scalable Haradware Priority Queue Architectures for High-Speed Packet Switches," IEEE Trans. Comp, vol. 49, no. 11, pp. 1215-1226. Nov. 2000.

저 자 소 개

김 상 균(학생회원)
2006년 경북대학교 전자전기 컴퓨터학부 학사 졸업
2007년 현재 경북대학교 전자 공학과 석사 과정
<주관심분야 : 네트워크 프로세서 설계>

문 병 인(평생회원)
1995년 연세대학교 전자공학과 학사 졸업
1997년 연세대학교 전자공학과 석사 졸업.
2002년 연세대학교 전기전자 공학과 박사 졸업.
2002년~2004년 하이닉스반도체 선임연구원
2004년~2005년 연세대학교 연구교수
2005년~현재 경북대학교 전자전기컴퓨터학부 조교수
<주관심분야 : 디지털 집적회로 설계, SoC 설계, 마이크로프로세서 구조 설계>