고속 라우터에 대한 고찰(I) - STC104의 레이블링 알고리즘

이 효 종†

요 약

고속 라우팅 스위치는 고성능 병렬처리의 통신망이나 VOD와 같은 멀티미디어 전송망에서 필수적으로 필요한 소자이다. 최대 100Mbps의 성능을 지니는 32개의 양방향 링크와 웜홀 라우팅, 구간 레이블링 및 적응라우팅 방법 등의 새로운 기술을 접목하여 개발된 고속 라우팅 스위치인 STC104는 전형적인 그 예이다. 이 고속 라우팅 스위치는 일부 병렬 컴퓨터에서 단일 칩으로 사용되고 있으나, 여러 개를 연결한 다양한 프로세서망에서 어떠한 성능을 나타날 지는 아직 연구되지 않았다. 본 연구에서는 STC104의 성능을 평가하기 위하여 구조와 특성을 먼저 고찰하였다. 임의 개수의 고속 라우터로 구성된 그물구조, 원환체 및 초입방체의 망을 설정하고, STC104가 지니는 구간 레이블링과 그룹 적응 라우팅 방법을 이용하여 패킷의 전송 알고리즘을 구현하였다. 또한 점대점 통신만을 할 수 있도록 제작된 STC104를 이용하여서 프로세서망과 방송 등에서 자주 요구되는 멀티캐스트를 지원할 수 있도록 변화된 U-mesh와 U-torus의 알고리즘을 제안하였다.

Study on High Speed Routers(I) - Labeling Algorithms for STC104

Hyo Jong Lee†

ABSTRACT

A high performance routing switch is an essential device to either the high performance parallel processing or communication networks that handle multimedia transfer systems such as VOD. The high performance routing chip called STC104 is a typical example in the technical aspect which has 32 bidirectional links of 100Mbps transfer speed. It has exploited new technologies, such as wormhole routing, interval labeling, and adaptive routing method. The high speed router has been applied into some parallel processing systems as a single chip. However, its performance over the various interconnection networks with multiple routing chips has not been studied. In this paper, the structures and characteristics of the STC104 have been investigated in order to evaluate the high speed router. Various topology of the STC104, such as meshes, torus, and N-cube are defined and constructed. Algorithms of packet transmission have been proposed based on the interval labeling and the group adaptive routing method implemented in the interconnected network. Multicast algorithms, which are often requited to the processor networks and broadcasting systems, modified from U-mesh and U-torus algorithms have also been proposed overcoming the problems of point-to-point communication.

키워드 : STC104, 레이블링알고리즘(labeling algorithm), 멀티캐스트(multicast), 고속라우터(highspeed router)

1. 서 론

현대 정보화 사회에서 통신 기술의 발달과 통신 시장의 급속한 성장은 기존의 문자 정보뿐만이 아니라 음성이, 정지 화상 및 동화상 정보까지 대용량의 복합적인 정보를 고속으로 처리할 수 있는 기술을 요구하게 되었다. 또한 지진 활동 분석이나 기상 예보와 같이 강력한 컴퓨터 처리 능력을 요구하는 분야에 이용되는 병렬 처리 기술 분야나 일반 상용 네트워크 분야에서도 방대한 정보의 신속한 처리를 위해 끊임없이 기술의 도전을 받고 있다. 따라서 짧은 시간 안에 정보를 가공하고 처리하는 기술의 요구에 대하여 다양한 하드웨어적인 개발과 더불어 프로세서간의 통신과 프로세서에 적합한 알고리즘 개발과 같은 소프트웨어적인 연구도 병행되고 있다. STC104나 기가비트를 전송하는 Rcube[13] 등은 하드웨어적인 처리의 대표적인 예이다.

STC104라는 통신 전용의 라우팅 스위치 칩은 첨단 VLSI 기술을 이용하여 개발되었다. 이 칩은 32개의 양방향 링크를 동시에 이용하여 메시지를 송·수신할 수 있으며 각 링크의 전송 속도는 최대100Mbps까지 가능하여 최대 대역폭 300

※ 본 논문은 "고속 라우터에 대한 고찰(II) - STC104 망 구성에 따른 성능 분석"으로 계속 됩니다.

† 종신회원 : 전북대학교 교수 · 공업기술연구소

논문접수 : 2000년 7월 10일, 심사완료 : 2001년 5월 29일

MByte/s에서 200Mpackets/s의 처리 능력을 가진다. 이러한 고속 라우터 칩은 고성능 병렬 시스템용 프로세서인 T9000[1]의 프로세서간 연결망에 사용되어 병렬시스템의 통신 작업을 처리하거나, 멀티미디어 자료의 고속 입출력을 다루는 시스템이나 수용자 요구 비디오(VOD : Video On Demand)시스템과 같이 방대한 자료의 고속 전송을 요구하는 광범위한 영역에 이용될 수 있다.

라우팅 스위치 자체의 특성 및 성능과 더불어 이를 이용해 임의의 망을 구성할 때 나타내는 성능 또한 중요한 변수가 된다. 기존의 STC104에 관한 자료들은 STC104 단일 성능에 대한 조사가 주를 이루고 있어 실제 응용시 여러 개의 라우팅 칩을 사용하는 망 구조의 선택을 위해서는 복합적인 연구가 이루어져야 한다. 따라서 망을 구성할 때 일반적으로 가장 널리 사용되는 직접 네트워크와 간접 네트워크를 중심으로 STC104의 특성을 충분히 반영하여 각 망의 through-put, latency 및 패킷 정체 시간을 중점적으로 연구할 필요가 있다. 본 논문에서는 이미 사용되고 있는 STC104 칩에 대한 구조와 성능 자료[2]를 근거로 모델링 하고 그 기능을 시뮬레이션 하는 소프트웨어 패키지를 개발하였다. 그리고 STC104에 내장되어있는 주요 특성인 웜홀 라우팅[3], 구간 레이블링 라우팅[4], 그룹 적응 라우팅 등의 다양한 기능을 적용시킨 상태에서 그물, 원환체(torus) 및 초입방체 망을 구성한 후 각 구조에 따른 전송알고리즘을 구현하였으며, 이에 따른 망의 성능을 측정하고 그 결과들을 비교 분석하였다. 또한, 점대점 네트워크의 취약점을 극복하고 멀티캐스트 기능을 부가하기 위하여 소프트웨어적으로 U-mesh[5], U-torus[6]등의 알고리즘을 STC104를 이용한 망에 적응하여 그 성능을 비교하고 주어진 망의 특성에 최적화될 수 있는 알고리즘을 제시하고자 한다.

본 논문의 구성으로 제2장에서는 STC104의 구조와 특성을 소개하고 이를 이용하여 구성할 망 구조에 대한 소개와 함께 각 망 구조에서 사용되는 구간 레이블링 방법에 대해 설명하였다. 제3장에서는 널리 사용되고있는 프로세서망과 범용 통신망에서 필수 요건으로 제시되고 있는 멀티캐스트를 구현하기 위한 변환된 소프트웨어 알고리즘을 소개하였다. 지면 관계 상 분리된 논문[14]에서는 STC104를 컴퓨터 시뮬레이션을 통해 구현하고 각 구조별 성능을 분석하고 제안된 소프트웨어 멀티캐스트 알고리즘을 시뮬레이션하고, 마지막으로 총괄적인 결론을 기술하였다.

2. STC104

라우터 칩의 성능을 정량적으로 분석하기 위해서는 STC104의 기능 구조와 사용되는 링크 프로토콜인 DS-Link [7]와 STC104를 구성하는 구성 요소들인 크로스바 스위치와 링크부 그리고 버퍼의 구성 및 특징을 이해하여야 한다. 또한 패킷 라우팅 기법과 헤더 제거, 그룹 적응 라우팅, universal 라우팅 등의 STC104에서 패킷의 라우팅에 사용되는 기술들의 특징을 그대로 구현하였다.

2.1 STC104의 구조

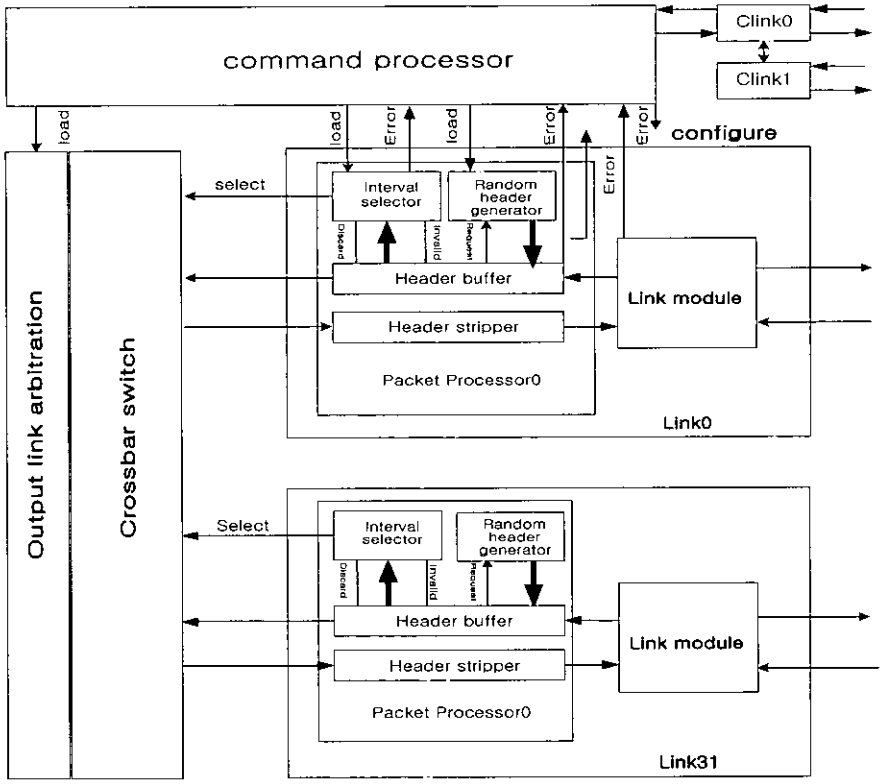
STC104의 하드웨어적인 측면보다 소프트웨어적으로 모델링 하는데 있어서 필요한 기능적 관점에서 구조를 살펴볼 필요가 있다. STC104는 라우팅 및 통신 제어 기능을 가지는 통신 전용의 스위칭 소자로 (그림 1)과 같은 구조를 가진다. 데이터들은 32개의 입력포트로부터 STC104의 핵심부인 크로스바 스위치로 들어가며 여기에서 입력된 데이터가 원하는 경로를 따라 32개의 출력 포트로 나간다. 각각의 링크는 이후 설명할 DS-Link를 사용하며 각 DS-Link는 하나의 입력포트와 출력포트를 구성한다.

DS-Link는 버퍼링을 제공하는 소자와 라우팅 기능을 수행하는 소자들과 함께 하나의 완전한 입력 경로와 출력 경로를 제공하는 링크부(linkslice)라고 하는 단위 구조를 이룬다.

STC104는 독립적으로 동작하는 32개의 링크부와 링크 입력부를 통해 받아들인 패킷의 라우팅 정보에 따라 경로를 만든 후 각 패킷을 원하는 출력단으로 내보내는 기능을 하는 크로스바 스위치 그리고 각 주요 요소들을 제어하는 제어부로 구성된다. 링크부는 패킷 프로세서와 링크 모듈로 구성되며 링크 모듈은 외부 포트와의 입·출력을 담당하며 패킷의 처리는 패킷 프로세서에 의해 행해진다. 명령 처리기는 각 요소들의 에러에 대한 처리 명령이나 환경 설정 명령 등을 처리하는 장치이다. Clink0와 Clink1은 일종의 DS-Link 모듈로서 STC104의 제어를 네트워크를 통해 행할 수 있도록 해주는 것으로 STC104로 구성된 망은 이 모듈을 통해 구성된 제어 네트워크를 통하여 전체 네트워크상의 STC104들을 제어할 수 있도록 되어있다.

2.1.1 DS-Link

STC104는 상호 연결망에서 통신 효율을 위하여 DS-Link를 사용한다. 하나의 DS-Link는 4개의 선으로 구성되며 입력과 출력단 각각 하나씩 데이터 라인(data line)과 스트로브 라인(strobe line)으로 구성되는데 DS-Link라는 이름은 바로 이 데이터 라인과 스트로브 라인으로 구성되는 데이터 스트로브 링크에서 기인한 것이다. DS-Link는 전이중(full-duplex), 비동기 통신을 지원하는 점대점(Point-to-point)연결을 제공하



(그림 1) STC104의 구조

며 링크 수준의 흐름 제어를 제공하며 링크 수준의 흐름 제어를 재공하는 통신 속도는 100MBit/s까지 지원한다.

DS-Link는 높은 전송속도를 지원하기 위해 특별한 방법의 비트 수준 프로토콜을 사용한다. 정보를 담은 데이터 라인과 제어신호인 스트로브 라인을 이용하여 clock을 전송하는 효과를 내게 된다. 데이터의 수신단에서는 데이터와 스트로브 라인의 EXOR 연산에 의해 clock 신호를 추출하기 때문에 신호 수신시 PLL(phase-locked loop)에 사용되는 면적을 줄일 수 있다. 더욱이 Gray coding은 간단하고 빠른 논리 회로로서 구현되어 질 수 있어 링크가 100Mbit/s 이상의 속도를 낼 수 있도록 해준다. 또한 DS-Link가 clock 신호를 함께 전송함으로써 발생되는 이점은 링크가 비동기이면서 수신 장치가 입력받는 데이터에 동기화 한다는 점이다. 이것은 전송 속도에 있어서 수신 장치의 최대 수신 속도를 초과하지 않는 범위내에서 임의의 전송속도로 전송이 가능하다는 것을 의미하기도 한다.

DS-Link가 처리할 수 있는 링크상의 오류는 두 가지로 패리티 에러와 단선(disconnection)에러이다. 먼저 패리티 에러는 DS-Link의 토큰 수준에서 한 비트의 에러가 일어났을 때 감지된다. 둘째로 다른 토큰이 없을 때 빈 토큰을 전송하는 프로토콜을 사용하여 링크의 단선 에러를 감지할 수 있도록 한다. 단선 에러는 실제로 링크가 물리적으로 단절되었을 때 일어나거나 링크의 다른 쪽 끝에서 오류가 일어나서 전송이 중지되었을 때 일어난다.

크로스바 스위치는 STC104의 중심이 되는 요소로 32개의 입력포트와 32개의 출력포트를 가진다. 크로스바 스위치에서는 어느 순간이나 32개의 점대점 연결이 임의의 입력포트와 출력포트 사이에서 이루어질 수 있으며 50MHz에서 싸이클 당 32바이트의 전송이 가능하다. 이것은 대략 초당 1.6 GBytes의 전송이 이루어질 수 있음을 의미한다. 크로스바 스위치내의 제어가 분산되어 있으므로 한 싸이클 내에 동시에 32개의 연결이 이루어지거나 해제될 수 있다.

2.1.2 크로스바 스위치

크로스바 스위치는 링크 부로부터 각 패킷의 앞부분에 출력포트를 결정하는데 필요한 값을 특별한 형태의 제어 토큰으로 넘겨받는다. 그리고 출력포트의 사용이 가능할 때 크로스바 스위치는 넘겨받는 제어 토큰을 이용하여 스위칭 망에 패킷이 지나갈 통로를 열고 패킷의 끝을 나타내는 종결 토큰을 받고 다음 단으로 전송할 때까지 개방된 경로를 유지한다.

연결이 이루어지면 데이터 선 외에 11 비트 버스의 핸드

쉐이크 신호선도 연결되고 따라서 크로스바 스위치를 통한 흐름 제어가 유지되며 이것은 칩 전체에 대한 흐름 제어 또한 가능하게 하다.

크로스바 스위치는 여러 개의 입력이 있을 때 각 입력들이 동시에 동일한 출력을 요구하는 경우 하나의 입력에 대한 처리가 진행되고 잇는 동안에 다른 입력들은 출력이 가능해질 때까지 대기시키는 입력들 간의 중재(arbitration)를 수행한다. 중재는 대기중인 입력이 제한된 시간 내에 모두 수행 가능하도록 행해진다. 크로스바 스위치는 또한 그룹핑 기법을 지원하여 요구한 입력에 대해 하나 이상의 출력단이 가능할 때 이들 사이에서 다른 단계의 중재를 수행한다.

2.1.3 버퍼링

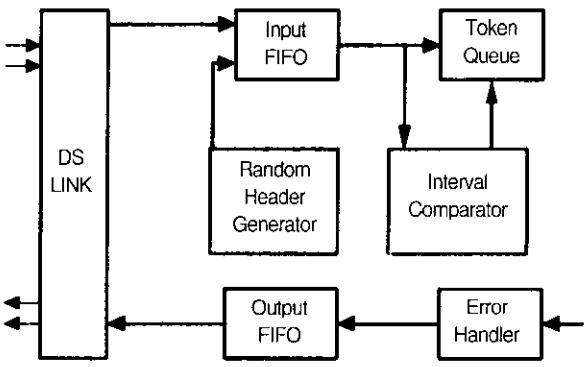
STC104의 주요 기능 중의 하나가 바로 각 요소마다 들어있는 버퍼이다. STC104는 32개의 경로 각각에 대해 모두 70개의 토큰을 저장할 수 있는 버퍼를 지니고 있다. 칩 전체적으로 볼 때 총 2240개의 토큰을 저장할 수 있는 용량을 지닌다. <표 1>은 STC104내 하나의 경로가 가지고 있는 버퍼를 각 요소들로 구분하여 나타낸 것이다.

〈표 1〉 한 경로내의 버퍼 분포

입력측 DS-Link	20
입력 FIFO	20
토큰 큐	3
크로스바 스위치	4
출력 FIFO	20
출력측 DS-Link	3
합 계	70

2.1.4 링크부

링크부는 라우팅 경로에서 크로스바 스위치를 제외한 입력 DS-Link로부터 크로스바 스위치까지의 경로에 해당하는 입력 부분과 크로스바 스위치에서 출력 DS-Link까지의 출력 부분을 합한 것으로 그 구성은 (그림 2)와 같다. 링크부의 구성은 그림에서 보이는 바와 같이 입력측에 입력 FIFO, 랜덤 헤더 발생기, 구간 비교기, 토큰 큐의 네 기능 블록으로 구성되고 출력측에 에러 처리기와 출력 FIFO 2개의 블록으로 구성된다.



(그림 2) 링크부의 기능 조직

입력 DS-Link로부터 데이터는 먼저 링크부 입력측의 마치 2단계의 파이프라인처럼 동작하는 입력 FIFO와 토큰 큐의 두 블록을 거치게 된다. 각 블록에는 랜덤 헤더 생성기와 구간 비교기가 있어서 데이터 흐름에 영향을 미치지는 않으나 입력 FIFO와 토큰 큐의 동작에 필요한 정보를 제공하는 역할을 한다.

입력 FIFO는 입력되는 모든 토큰이 거치게 되는 블록으로 장치가 한가할 때는 토큰에 대해 단지 한 사이클의 latency만을 반영한다. 그러나, 출력 경로가 차단되었을 때, 즉 두 개의 패킷이 동시에 같은 출력단을 요구하여 크로스바 스위치에서 다른 포트에 의해 경로가 사용되고 있을 경우 FIFO는 DS-Link로부터 최대 버퍼 크기인 20개까지의 토큰을 받아들여 버퍼에 저장한다. 토큰 큐가 토큰을 받을 준비가 되는 순간, 즉 해당 입력에 대한 포트에로의 출력이 허용되었을 때 FIFO는 토큰 큐로 데이터를 입력받은 순서대로 내보낸다. 여기에서 입력 FIFO의 처리 능력은 한 사이클 당 한 토큰의 비율로 데이터를 입·출력 할 수 있다.

다음절에서 언급되는 Universal 라우팅 기능이 적용되는 경우 입력 FIFO는 'randomize' 환경 플래그가 설정되어 있으면 랜덤 헤더 발생기가 생성한 헤더를 DS-Link로부터 받은 패킷의 앞부분에 결합시켜 토큰 큐로 보낸다. 임의로 발생되는 헤더는 프로그램 가능한 변수 base와 range에 의해 식 (1)과 같은 범위 내에서 선택되어진다.

$$base \le header \le base + range \qquad (1)$$

구간 비교기는 미리 프로그램 되어진 구간 라우팅표와 입력되는 헤더를 비교하여 출력 링크와 상태 플래그를 발생시킨다. 구간 비교기는 병렬적으로 한 사이클 내에 라우팅 표에 있는 모든 구간의 비교를 행할 수 있다.

토큰 큐는 구간 비교기에 의해 생성된 데이터를 해석하고 입력되는 데이터 스트림의 구조적 에러를 찾아내는 역할을 한다. 토큰 큐로 입력된 패킷은 크로스바로 보내어지거나 또는 토큰 큐에서 소모되고 에러를 발생시킨다. 에러처리기는 링크부의 출력부에 있는 두 블록 중 하나로 에러가 일어났을 경우의 처리를 제외하고는 일종의 0 사이클 latency 버퍼로 동작한다.

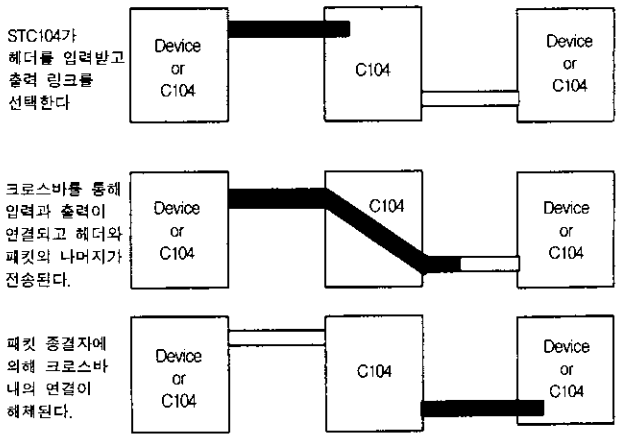
출력 FIFO는 20개까지의 토큰을 저장할 수 있는 버퍼이며 헤더 제거 기능을 가지고 있다는 것을 제외하고는 입력 FIFO와 동일한 방식으로 동작한다. 만일 헤더 제거 플래그가 설정되어 있으면 패킷의 앞쪽에 있는 보통 헤더에 해당되는 하나 또는 두 개의 데이터 토큰이 제거된다. 헤더의 크기는 프로그램 가능한 플래그에 의해 설정되며 헤더를 제거한 이후 패킷에 남아있는 데이터 토큰이 없다면 패킷은 두 번째 헤더를 가지지 않으며 그런 경우 패킷의 끝을 나타내는 종결 토큰이 제거되고 "null packet" 에러가 발생된다.

2.2 STC104의 특성

STC104는 VLSI 기술을 이용한 라우팅 소자로 32 방향 크로스바 스위치와 DS-Link를 사용하며 라우팅 지연 시간을 최소화하기 위한 wormhole 라우팅, 헤더 제거법, 구간표시 라우팅, 적응 라우팅, universal 라우팅 등의 기술을 이용하고 있다. 또한 이러한 기능들을 제어하기 위한 프로그래밍까지 가능한 별도의 제어부를 가지고 있다.

2.2.1 패킷 라우팅

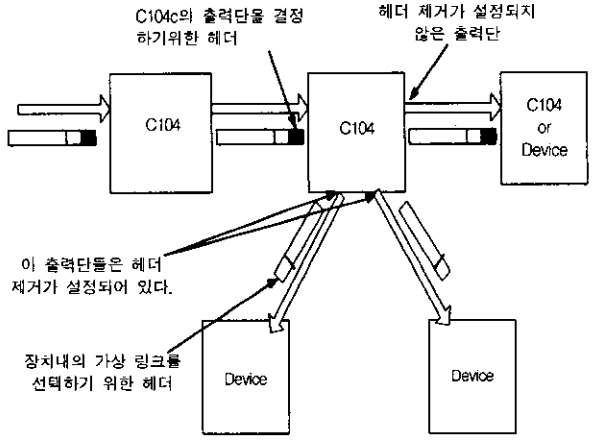
패킷 라우팅은 네트워크상의 스위치와 링크들을 통과하는 데이터의 흐름을 제어하는 링크 프로토콜을 따른다. 대부분의 라우팅 프로토콜에서 데이터는 플릿(flits)이라 불리는 여러 개의 작은 데이터 조각들로 구성된 패킷들로 이루어진다. 플릿은 두 개의 스위치 사이에서 한 사이클에 전송되어질 수 있는 가장 작은 단위의 데이터 조각으로 고정된 크기를 갖는다. 머리 플릿은 출력단을 결정하기 위해 스위치가 사용하는 라우팅 정보를 가지고 있다. 일단 패킷이 네트워크로 들어가면 그 패킷의 머리 플릿이 가장 먼저 가고 이 머리 플릿을 받는 스위치는 담겨진 라우팅 정보에 따라 출력단을 설정하고 패킷이 통과할 경로를 설립한다. 뒤이어 라우팅 정보를 가지지 않는 플릿들이 머리 플릿을 따라 스위치로 들어오고 머리 플릿에 의해 설립된 경로를 따라 전송된다. 패킷의 마지막 플릿은 해당 패킷의 꼬리 플릿이라 불리며 각각의 스위치는 이 꼬리 플릿이 통과한 후 출력단을 다른 패킷에 의해 사용될 수 있는 상태로 만들어 지나간 패킷에 의해 설립되었던 경로를 닫는다. 패킷의 라우팅은 store-and-forward 방식, virtual-cut-through[8], 그리고 wormhole 라우팅 방식이 있는데 STC104는 32 x 32 비-블로킹 크로스바 스위치를 가지며 임의의 링크에서 다른 임의의 링크로 다른 링크들에 대한 간섭 없이 라우팅 될 수 있도록 wormhole 라우팅을 사용한다. (그림 3)에 보인 바와 같이 패킷의 헤더가 STC104에 받아들여지는 즉시 크로스바를 통한 연결이 설정되며 패킷의 헤더와 그 나머지가 바로 출력 링크로 전송되어지기 시작한다. 이렇게 스위치를 통해 설립된 경로는 'end of packet' 플릿이나 'end of message' 플릿이 통과하고 나면 사라지게 된다.



(그림 3) STC104에서의 wormhole 라우팅

2.2.2 헤더 제거

네트워크의 구성을 단순하게 하기 위한 방법으로 각 패킷상의 헤더에 두 개의 단계를 제공하는 방법이 있다. 첫 번째 헤더는 목적 장치를 지정하는데 보통 라우팅 네트워크로부터의 출력 링크를 의미한다. 패킷이 라우팅 장치를 떠날 때 첫 번째 헤더가 제거되어 이 패킷을 처리할 프로세서에게 두 번째 헤더를 나타내게 된다. 이러한 방식을 지원하기 위하여 STC104는 임의의 길이의 패킷도 라우트 할 수 있다. STC104에 의해 사용되는 처음 헤더에 해당하는 몇 바이트 이후의 정보는 실제 시스템의 다른 부분에서 헤더가 패킷의 앞부분에서 제거되고 난 이후에 헤더로 사용된다. (그림 4)를 보면 STC104의 각 출력 링크는 패킷의 헤더가 라우팅 결정을 내리는 데 사용된 다음 각 패킷의 앞부분으로부터

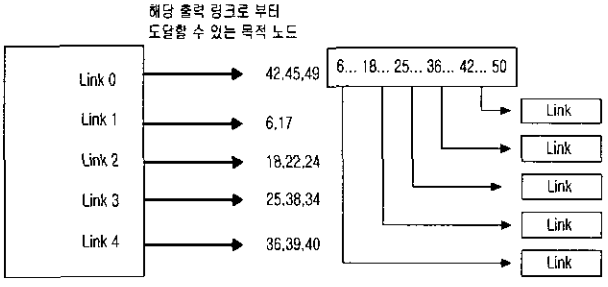


(그림 4) 헤더 제거

라우팅 헤더를 제거하도록 설정되어질 수 있다. 따라서 기존의 라우팅 헤더가 제거되고 난 이후 남은 데이터의 앞부분은 다음 장치에 의해 패킷이 받아들여질 때 헤더로서 다루어지게 된다.

2.2.3 구간 레이블링(Interval labeling)

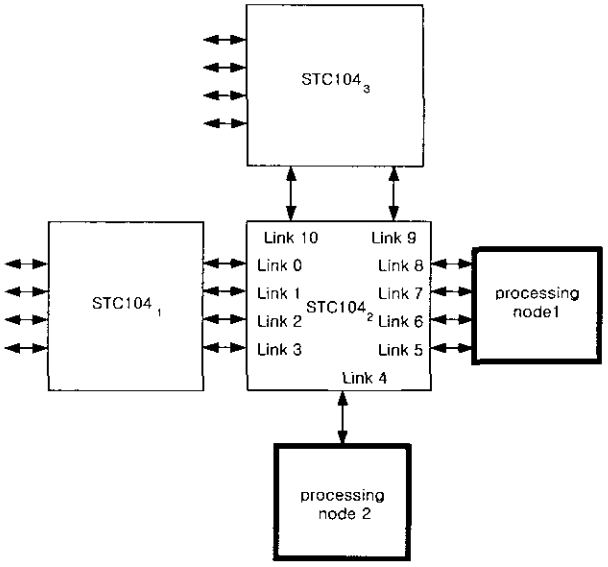
네트워크 상에서 라우터들은 각자 가지고 있는 링크를 통해 다수의 터미널들과 연결되어있다. 따라서, 각각의 패킷이 전송되기 원하는 목적지에 도착하기 위해 사용할 출력 링크를 결정하는 레이블링 방법을 채택하고 있다. 즉, 원홀 라우팅을 사용하는 STC104는 (그림 5)와 같이 각각의 출력 링크에 주소의 구간을 할당하여 그 링크를 통해 도달할 수 있는 모든 터미널 노드를 라우팅 할 수 있도록 한다. 패킷이 STC104에 도착했을 때 헤더는 구간 라우팅 표의 전체구간과 비교되어지며 이를 통해 출력 링크가 결정된다. 구간은 연속적으로 겹침이 없이 각 헤더가 반드시 하나의 구간에 해당하도록 설정된다. 구간 레이블링은 [m, n)으로 표기하며 그 의미는 해당 링크에 연결된 장치들의 주소범위가 m보다 크거나 같고 n보다 작다는 것을 나타낸다. 이 방법은 어떠한 네트워크에서도 모든 패킷이 목적지를 찾아가도록 보장한다. 또한 하드웨어로 구현하기에 간단하며 작은 실리콘 면적에 많은 수의 링크에 대한 레이블링이 가능하므로 비용과 전력 손실을 적게 할 수 있다.



(그림 5) 구간 레이블링을 사용하는 네트워크

2.2.4 그룹적응 라우팅(Grouped adaptive routing)

그룹적응 라우팅은 연속적인 번호를 갖는 일련의 링크들을 한 그룹으로 지정하여 그 그룹으로 라우팅 되어진 패킷이 동일 그룹내의 사용되지 않는 링크로 전송되도록 하여 latency와 throughput상에서 더 나은 성능을 가지도록 하는 방법이다. (그림 6)은 그룹 적응라우팅을 보여주며 패킷이 $STC104_1$에서 $STC104_2$를 거쳐 $node_1$로 설정된다고 가정할 때 패킷이 $STC104_2$로 들어가는데 있어서 패킷의 헤더는 패킷이 Link5를 통해 $node_1$로 출력되도록 확장한다. 만일 Link5가 이미 사용중이라면 패킷은 자동으로 가장 먼저 사용 가능한 링크를 찾아 Link6, Link7 또는 Link8로 전송되어진다. 링크들은 각각의 링크에 있는 비트를 설정함으로써 그룹화 되는데 만일 그 값이 0이면 그룹의 시작을 나타내고 1이면 이미 시작된 그룹에 포함되는 것을 의미한다.



(그림 6) 그룹적응 라우팅의 예

3. 네트워크 구성

일반적으로 네트워크의 구성은 사용하는 라우팅 알고리즘과 네트워크 구조에 따라 달라진다. 본 논문에서는 구간 레이블링 라우팅과 함께 원홀 라우팅과 그룹 적응 라우팅을 이용하는 네트워크로 설정하였다. 두 노드를 연결하는 링크가 여러 개의 링크로 구성되는 경우를 다중 링크(multiple links)라 하고 다중 링크에서 한 방향으로의 링크의 수를 다중 링크의 너비(width)라 하며 이 수는 모든 다른 노드를 향한 방향에 대해 동일한 것으로 제한한다.

3.1 그물구조

2차원 그물구조는 각 열이 p개의 노드로 구성된 m개의 1차원 배열로 구성된 것으로 볼 수 잇다. 여기에서 각각의 노드는 네트워크에서의 라우터를 의미한다. 이들 노드는 (그림 7)과 같이 레이블링하며 일반화하면 첫행에 해당하는 노드부터 0, ⋯, p-1, p, ⋯, 2p, ⋯, 3p-1, ⋯, (m-1)p, ⋯, mp-1로 레이블링된다. 여기에서 각 노드에 할당된 번호는 실제로 노드로 표현된 라우터에 연결된 가사의 터미널에 대한 주소이며 따라서 노드 n은 실제로[n,n+1)로 레이블링된 라우터의 출력에 연결된 터미널과 라우터 자체를 표현한다. 레이블링 원하는 노드의 번호를 c라 하고 노드 c가 속한 열의 번호를 I라 하면 p개의 노드로 구성된 m개의 1차원 배

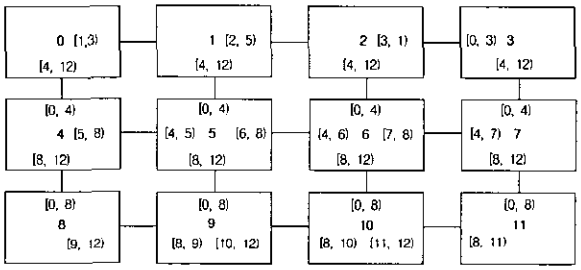
열에서의 레이블링 방식은 다음과 같다.

$$I_{U,C} = [0, (i-1)p),\ i = 0, \cdots p-1 \quad (2)$$

$$I_{D,C} = [ip, mp),\ i = 0, \cdots p-1 \quad (3)$$

$$I_{L,C} = [(i-1)p, c),\ i = 0, \cdots p-1\ ;\ c = 0, \cdots mp \quad (4)$$

$$I_{R,C} = [c+1, ip),\ i = 0, \cdots p-1\ ;\ c = 0, \cdots mp \quad (5)$$



(그림 7) 그물구조에서 구간 레이블링의 예

$I_{U,C}$는 그물구조에서 해당노드의 위쪽, 즉 행번호가 낮은 방향에로의 링크에 대한 레이블링이고, $I_{D,C}$는 아래쪽의 링크에 대한 레이블링이다. $I_{L,C}$는 한 행을 이루는 노드들의 1차원 배열상에서 낮은 주소를 가지는 쪽을 향하는 링크에 대한 레이블링이며 $I_{R,C}$는 높은 주소를 가지는 쪽으로 향한 링크에 대한 레이블링이다. 만일 하나의 라우터에 두 개 이상의 터미널이 연결되는 경우 한 개의 라우터 당 터미널 노드수를 a라 하면 각 링크의 구간을 나타내는 변수에 a를 곱한다.

3.2 원환체

2차원 그물구조에서 동일 차원상의 노드 각 종단을 반대편 종단과 연결시킴으로서 N차원의 순환형 그물구조를 구성할 수 있다. 이렇게 형성된 순환형의 그물구조를 원환체(torus) 구조라 한다. 원환체 구조의 노드는 그물구조의 노드와 같은 방법으로 번호가 할당된다. 그러나, 구간 레이블링에 있어서는 원환체가 링 구조에서처럼 서로 반대쪽을 위치하는 노드가 상호 연결되어 주회 효과를 고려해야 한다.

원환체에서의 레이블링은 최적 라우팅을 위해 패킷은 가장 짧은 경로를 따르도록 행해지며 노드들을 통한 경로는 링 구조에서와 같이 그 방향에 존재하는 전체 노드 수의 1/2보다 크지 않다.

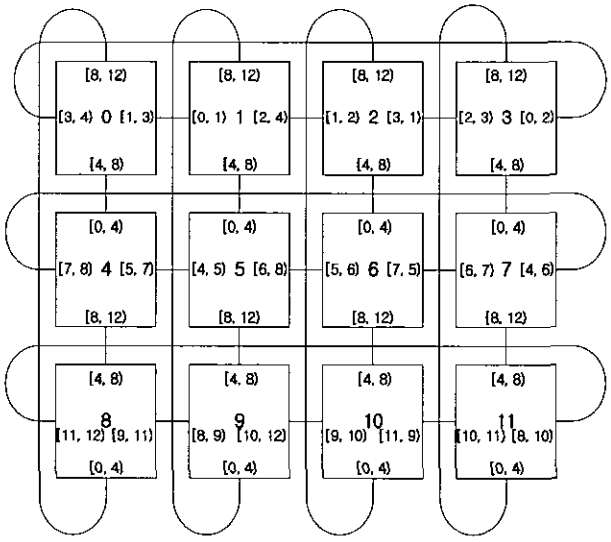
(그림 8)은 원환체의 레이블링을 예로 보인 것으로 각 노드의 링크에 대한 레이블링은 최단 경로를 갖는 최적 라우팅 레이블링을 따른 것이다. 일반적인 레이블링 방법은 m ×n 구성의 원환체에서 레이블링을 원하는 노드를 c라 하고 노드 c가 속한 행의번호를 i, 열 번호를 j라 하면 구간 레이블링은 다음과 같다.

$$I_{U,C} = [((i + 1 + 1 + n/2)\%n)m, im),$$
$$i = 0, \cdots, n-1 (im : \text{if } i = 0,\ im = nm) \quad (6)$$

$$I_{D,C} = [((i + 1)\%n)*m), ((i + 1 + n/2)\%n)m),$$
$$i = 0, \cdots, n-1 \quad (7)$$

$$I_{L,C} = [((j + 1 + m/2)\%m) + im, c),$$
$$i = 0, , p-1\ ;\ j = 0, \cdots, m-1 \quad (8)$$

$$I_{R,C} = [i * m + (j + 1)\%m, ((j + 1 + m/2)\%m) + im),$$
$$i = 0, \cdots, n-1\ ;\ j = 0, \cdots, m-1 \quad (9)$$



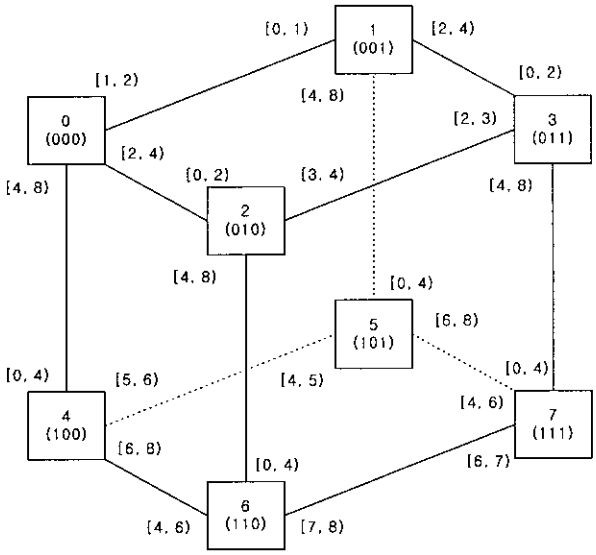
(그림 8) 원환체에서 구간 레이블링의 예

3.3 초입방체

일반적으로 N-cube라 불리며 노드의 확장이 2^n으로 증가되는 것이 특징이다. 망의 복잡성과 노드 확장의 제한에도 불구하고 다른 구조에 비해 우수한 성능을 보여주어 병렬 프로세서와 같이 통신의 고성능을 요하는 분야에서 이용된다. 초입방체에서의 노드 주소는 비트 표현에 기반을 두고 (그림 9)와 같이 할당된다. 일반적으로 N차 초입방체는 M = N-1인 두 개의 M차 초입방체 H_1과 H_2로 이루어진 것으로 간주되며 두 개의 M차 초입방체를 결합하는데 있어 M차 초입방체 H_1에서의 노드는 $0, \cdots, 2^n-1$로 레이블링 되며 H_2에서의 노드는 $2^n, \cdots 2^{n+1}-1$로 레이블링 된다. H_1에 있는 각각의 노드 H_1과 H_2의 대응하는 노드 H_2를 연결하는 링크는 H_1에서는 구간 $[2^n, 2^{n+1})$으로 레이블링 되며 H_2에서는 $[0, 2^n)$으로 레이블링 된다. 초입방체의 레이블링은 원래의 구조를 각 단계에서 더욱 간단한 구조로 분할시켜 나가면서 레이블링 하는 순환 반복적인 작업이다. (그림 9)에서 보인 3차 초입방체는 상대적으로 간단한 구조로 그 이상의 분해가 필요 없으므로 어떤 다른 높은 차수의 초입방체의 레이블링을 행하는데 있어서의 가장 적당한 기본

구조가 된다. 레이블링 알고리즘의 일반화를 위하여 각 노드에 해당하는 라우터를 $(c_0, c_1, \cdots, c_{n-1}) \in [0, 2^n)$이라 할 때 각 차원 $i \in [0, n)$에 대한 구간 표기 I_i는 다음과 같이 한다.

$$I_i(c_0, c_1, \cdots, c_{n-1}) = [(\sum_{i<l<n} c_l 2^l), (\sum_{i+1<l<n} c_l 2^l) + (c_{i+1}+1)2^{i+1}) - [(\sum_{i<l<n} c_l 2^l), \sum_{i<l<n} c_l 2^l + (c_i+1)2^i) \quad (10)$$



(그림 9) 초입방체의 레이블링의 예

초입방체도 다른 구조의 레이블링에서와 같이 하나의 라우터에 두 개 이상의 터미널이 연결되는 경우 한 개의 라우터당 터미널 노두수를 α라 할 때 레이블링은 각 링크의 구간을 나타내는 변수에 α를 곱한다.

3.4 멀티 캐스트

멀티캐스트는 하나의 발원노드로부터 여러 개의 확정된 목적지로 데이터를 전송하는 것을 의미한다. 이것은 네트워크 상의 모든 노드로 동일한 데이터를 전송하는 브로드캐스트(broadcast)의 일반화된 형태이다. 최근 MBone[9]이나 Stream Works[10]와 같이 IP(internet protocol)상에서 멀티캐스트를 수행하려는 많은 노력들이 행해져 왔다. 실시간 서비스들은 종단 지연 시간과 지터(jitter), 패킷손실 등에 대해 지정된 수준의 성능을 요구하며 네트워크는 어떤 수준의 그 연결 성능을 유지하기 위해 네트워크의 링크와 버퍼, 라우팅 자원들을 관리한다.

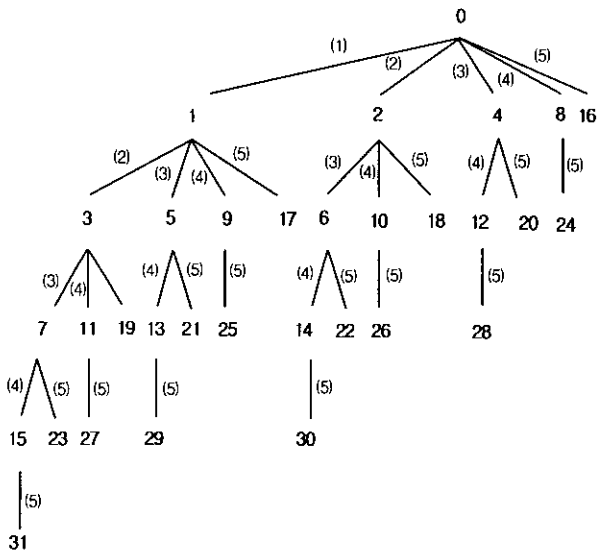
일반적으로 멀티 캐스트의 구현은 소프트웨어적인 방법과 하드웨어적인 방법으로 구분할 수 있다. 소프트웨어적인 방법은 멀티캐스트를 원하는 노드에 대한 소프트웨어 트리를 구성하여 각 노드에 멀티캐스트 트리의 하나 또는 그 이상의 복사본을 전송하는 방법이다. 하드웨어적인 방법은 네트워크를 구성하는 스위치에서 지원하는 멀티캐스트 기능을 사용하는 방법이다. 하드웨어적으로 점대점(point-to-point) 통신만을 지원하는 STC104에서는 소프트웨어멀티캐스트 방법을 이용하여야한다.

3.4.1 소프트웨어 멀티캐스트

멀티캐스트 소프트웨어 알고리즘들은 노드들에 패킷을 보내는 방식에 따라 그 특징을 달리하며 일반적으로 멀티캐스트 그룹내의 모든 노드에 패킷을 보내는 대신에 멀티캐스트 그룹 내의 몇몇 노드에 패킷을 보내고 패킷을 받은 노드들은 아직 패킷을 받지 못한 그룹내의 다른 노드로 패킷을 전송하는 방식을 취한다.

최적 멀티캐스트 트리의 구성 방법은 멀티캐스트가 적용되는 망의 구조와 라우팅알고리즘에 따라 달라지며 패킷이 동시에 동일한 링크를 사용하지 않도록 구성한다.

(그림 10)은 노드 0이 31개의노드(1,2,3,4,…,31)에 멀티캐스트를 한다고 할 경우의 멀티캐스트 트리의 예이다. 노드 0이 노드 1에 패킷을 보내고 다시 노드 2에 패킷을 보낼 때 노드 1은 그와 동시에 자신의 부분 집합중의 노드 3에 패킷을 보낸다. 그리고, 다시 노드 1이 노드 4에 패킷을 보낼 때 노드 1은 노드 5로, 노드 3은 자신의 부분 집합에 속하는 노드 7로 메시지를 보낸다. 물론 이 때 노드 0으로부터 패킷을 받았던 노드 2도 역시 자시의 부분 집합에 속하는 노드에 패킷을 보낸다. 이렇게 하여 모든 목적지 노드에 대한 멀티캐스트는 5단계에서 끝나게 된다.



(그림 10) 멀티캐스팅 트리의 예

3.4.2 멀티캐스트 알고리즘

멀티캐스트 알고리즘은 망의 구조와 패킷의 라우팅 방법에 따라 멀티캐스트 트리의 구성 방법과 운용방법을 달리한다. 특히 망의 구성 방법은 멀티캐스트를 원하는 목적 노드들의 분포에 영향을 미쳐 멀티 캐스트 트리의 분할 전송에 영향을 미친다. 웜홀 라우팅을 사용하지 않는 네트워크에서 다른 목적 노드들과 인접하지 않은 노드에 대한 패킷의 전송을 위해 중간 노드의 선택은 멀티캐스트 트리의 전체적 구조에 큰 영향을 미칠 수 있다.

3.4.3 그물구조

먼저 그물구조에서의 멀티캐스트는 U-mesh 알고리즘[11]을 구현 대상으로 삼았다. U-mesh의 알고리즘은 (그림 11)에 기술한 바와 같다. 기본 개념은 메시지의 발신자를 중심으로 그물구조를 분할해 나가면서 각각의 대표 노드에 부분 집합노드로 구성된 트리와 데이터를 함께 보내는 방법이다. 이 알고리즘의 구현 및 성능은 [11]에서 Caltech 12 × 14 2차원 그물구조의 168-node Symult 2010컴퓨터에서 소프트웨어적으로 구현되어 발원 노드가 멀티캐스트 데이터를 보낼 모든 노드에 직접 데이터를 보내는 개별 전송 방법과 [12]에서 설명된 Symult 시스템 자체의 멀티캐스트 지원 명령인 Xmsend와 비교 실험 결과 더 나은 성능을 보이는 것으로 나타났다.

```
Inputs : Φ : dimension-ordered chain (Dleft, Dleft+1, ···, Dright)
             for source and destination
         Dsource : the address of source node
Output : Send [ log2(right - left +1)] Messages
Procedure :
while left < right do
   if source < (left+right)/2 then          /* send right */
      center = letf +[(right-left)/2] ;
      D = {Dcenter, Dcenter+1, ···, Dright} ;
      right =center -1 ;
   else if source > (left+right)/2 then      /*send left */
         center = left +[(right-left)/2] ;
         D = {Dleft, ...., Dcenter-1, ···, Dcenter} ;
         left = center +1 ;
   else                                      /*send left */
         center = source -1 ;
         D = {Dleft, Dcenter-1, ···, Dcenter} ;
         left = source ;
   endif
   send a message to node Dcenter with the address field D ;
endwhile
```

(그림 11) U-mesh 알고리즘

3.4.4 원환체

원환체의 경우 그물구조와 그 구성 형식이 비슷하나 그물구조에서의 각 행 또는 열의 종단에 있는 노드들이 서로 연결되어 회전의 특성을 지니고 있으므로 멀티캐스트 역시 이에 대한 고려가 있어야 한다. 웜홀 라우팅 특성이 고려된 원환체에서의 멀티 캐스트에 관한 연구[13]에서 원환체에서의 최적 멀티캐스트 방법에 대한 소개와 함께 수신 노드수의 변화에 따른 성능을 측정하고 개별 전송과의 성능 비교를 하여 더 나은 결과를 보여주었다. (그림 12)는 U-torus 알고리즘을 기술한 것이다. U-torus알고리즘에서는 U-mesh와 각 종단이 상호 연결되어서 링 구조와 동일한 방법으로 중심 노드를 결정하여 메시지를 보낸다.

```
Inputs : R-chain(Dleft, Dleft+1, ···, Dright),
         where Dleft is the local address

Output : Send [ log2(right - left +1)] messages
Procedure :
while left < right do
   center = letf +[(right-left+1)/2] ;
   D = {Dcenter, Dcenter+1, ···, Dright} ;
   Send a message to node Dcenter with the address field D ;
   right = center +1 ;
endwhile
```

(그림 12) U-torus 알고리즘

참 고 문 헌

[1] Simpson, M, Thompson, P. W., "The T9000 Communications Architecture," Networks, Routers & Transputers, Inmos Ltd. 1993.

[2] The STC104 Asynchronous Packet Switch, Preliminary Data sheet, SGS Thompson Microelectronics, June 1994.

[3] Ni, L. M. and Mckinley, P.K., "A Survey of Wormhole Routing Techniques in direct Networks," IEEE Computer, Vol.26, pp.62-76, February 1989.

[4] Van Leeuwen, J. and Tan, R. B., "Interval Routing," The Computer Journal, Vol.30, No.4, pp.298-307, 1987.

[5] Tsai, Yih-Jia and McKinley, Philip K., "An Extended Dominating Node Approach to Collective Communication in All-Port Wormhole-Routed 2D Meshes," Proc. 1994 Scalable High performance Computing Conference, pp.199-206, May, 1994.

[6] Robinson, David E., McKinley, Philip K., and Cheng, BettyH. C., "Optimal Multicast Communication in Wormhole-Routed Torus Networks," Proc. 1994 International Conference on Parallel Processing, August, 1994.

[7] M. Simpson, P. W. Thompson, "DS-Links and C104 Routers," Networks, Routers, & Transputers, INMOS Ltd., 1993.

[8] P. Kermani and L. Kleinrock, "Virtual cut-through : a new computer communication switching technique," Computer Networks, Vol.3, pp.267-286, 1979.

[9] Casner, "Frequently Asked Questions on the Multicast Backbone," ftp : //venera.ise.edu/mbone/faq.txt, May 6, 1995.

[10] Petri Koskelainen, "Report on Streamworks (tm), "http :

//www.cs.tut.fi/~petkos/streams.html, August 1995.

[11] Philip K. McKinley, Hong Xu, Abdol-Hossein, "Unicast-based Multicast Communication in Wormhole-routed Networks," IEEE Transactions on Parallel and Distributed Systems, Vol.5, No.12, pp.1252-1265, December 1994.

[12] Ametek Computer Research Division, Arcadia, Ametek System 14, Mars System Software User's Guide Version 1.0, 1987.

[13] B. Zerrouk, V. Reibaldi, F. Potter, A. Greiner and A. Derieux, "RCube : a message routing device using the OMI/HIC high speed link technology," Electronics, Circuits, and Systems, 1996, ICECS '96, Proceedings of the Third IEEE International Conference on, Vol.1, Page(s) : 343-345 Vol.1, 1996.

[14] 이효종, "고속라우터에 대한 고찰(II) - STC104의 망 구성에 따른 성능분석", 정보처리학회논문지 A, 제8-A권 제2호, 2001.

이 효 종

e-mail : hlee@mail.chonbuk.ac.kr

1982년 전북대학교 지구과학과 졸업(이학사)

1985년 University of Utah 기상학과 졸업(이학석사)

1986년 University of Utah 컴퓨터과학과 졸업(공학사)

1988년 University of Utah 컴퓨터과학과 대학원 졸업(공학석사)

1991년 University of Utah 컴퓨터과학과 대학원 졸업(공학박사)

1986년~1989년 University of Utah 컴퓨터과학과 조교

1989년~1991년 University of Utah 선임프로그래머

1991년 University of Utah 대기연구소 연구원

1991년~현재 전북대학교 교수

관심분야 : 병렬영상처리, 인공지능, 병렬인식 알고리즘 개발 및 소프트웨어 공학