NoC 용 고속 데이터 패킷 할당 회로 설계

김정현 · 이재성*

한국교통대학교

Design of a High-Speed Data Packet Allocation Circuit for Network-on-Chip

Jeonghyun Kim · Jaesung Lee*

Korea National University of Transportation

E-mail : hanagod2015@gmail.com / jaesung.lee@ut.ac.kr

요 약

Network-on-Chip (NoC) 이 오프칩 네트워크 기반의 기존 병렬처리 시스템과 가장 크게 다른 점은 데이터 패킷 라우팅을 중앙 제어 방식(Centralized control scheme)으로 수행한다는 점이다. 이러한 환경에서 Best-effort 패킷 라우팅 문제는 데이터 패킷이 해당 코어에 도달 및 처리되는 시간을 Cost 로 하는 실시간 최소화 할당 문제(Assignment problem)가 된다. 본 논문에서는 할당 문제의 선형 대수 방정식에 대한 대표적인 연산 복잡도 저감 알고리즘인 헝가리안 알고리즘을 하드웨어 가속기 형태로 구현하였다. TSMC 0.18um 표준 셀라이브러리를 이용하여 논리 합성한 결과 헝가리안 알고리즘의 연산과정을 그대로 구현한 하드웨어 회로보다 Cost 분포에 대한 Case 분석을 통하여 구현한 것이 면적은 약 16%, Propagation delay는 약 52% 감소한 것으로 나타났다.

ABSTRACT

One of the big differences between Network-on-Chip (NoC) and the existing parallel processing system based on an off-chip network is that data packet routing is performed using a centralized control scheme. In such an environment, the best-effort packet routing problem becomes a real-time assignment problem in which data packet arriving time and processing time is the cost. In this paper, the Hungarian algorithm, a representative computational complexity reduction algorithm for the linear algebraic equation of the allocation problem, is implemented in the form of a hardware accelerator. As a result of logic synthesis using the TSMC 0.18um standard cell library, the area of the circuit designed through case analysis for the cost distribution is reduced by about 16% and the propagation delay of it is reduced by about 52%, compared to the circuit implementing the original operation sequence of the Hungarian algorithm.

키워드

MPP, NoC, On-Chip Router, Allocation Method, Hardware Accelerator

Ⅰ. 서 론

NoC 가 오프칩 네트워크 기반의 기존 병렬처리 시스템과 가장 크게 다른 점은 데이터 패킷 라우팅을 중앙 제어 방식(Centralized control scheme)으로 수행한다는 점이다. 그 이유는 Network Interface (NI), Router 등 기존 Peer-to-peer 네트워크 통신용 컴포넌트의 오버헤드가 너무 커서 단일 칩안에 많은 수의 NI, Router를 집적시키기가 어렵기 때문이다. 중앙 제어 방식으로 제어할 경우 Router의 내부 구조는 매우 단순해지며 그에 따라 NI도 Bus wrapper 정도 수준으로 간단해진다. 한편, NoC 의 경우 각 코어(Core)에게 배정할 수 있는 전용 메모리의 크기도 매우 한정적이어서 코어들이 처리해야 할 데이터의 대부분을 외부 오프칩 메모리(일종의 전용 스토리지)로부터 직접 공급을 받는다는 점도 기존 오프칩 네트워크와 다른 점이다.

이러한 환경에서 Best-effort 패킷 라우팅 문제는 데이터 패킷이 해당 코어에 도달 및 처리되는 시간을 cost로 하는 실시간 최소화 할당 문제(Assignment problem)가 된다. 중앙 제어기에 데이

* corresponding author

터 패킷의 크기(또는 코어에서 처리 소요 시간), 온칩으로 진입시 포트 위치, 그리고 할당 코어까지의 거리(Hopping count) 등의 정보가 미리 제공된다면 코어까지 도달 시간 및 데이터 처리 시간을 cost로 하여 모든 가능한 코어-데이터 맵핑 조합들에 대한 합산 cost 들을 비교하여 최소 cost 를 소모하는 맵핑 조합에 따라 라우팅을 하면 된다. 실제로 쓰레드 데이터들은 대부분 헤더와 데이터 페이로드로 구성된 패킷 형태로 전송되며 헤더에 발송 위치, 데이터의 크기 등에 대한 정보가 담겨 있어 온칩 진입과 동시에 그 정보들을 미리 알 수 있다.

하지만, 최소 cost의 할당 조합을 찾기 위해 소요되는 연산량이 매우 많아 실시간 라우팅을 실현하는데 많은 어려움이 있다. 이에 본 논문에서는 할당 문제의 선형 대수 방정식에 대한 대표적인 연산 복잡도 저감 알고리즘인 헝가리안 알고리즘을 하드웨어 가속기 형태로 구현하였다. 특히, 신호 전달 지연(propagation delay)를 최소화하여 실시간 맵핑이 가능하도록 하였다.

II. 본 문

헝가리안 알고리즘은 할당 문제의 연산 복잡도를 대폭 감소시켜줄 수 있는 알고리즘으로 널리 알려져 있다.[1] 그림 1의 예시와 같이 코어와 데이터 패킷간 cost 값으로 구성된 행렬에 대하여 헝가리안 알고리즘을 적용하면 다음과 같은 흐름으로 처리된다. 각 행의 최소값을 찾아 각 행렬의 원소에서 빼준다. 그러면 주어진 행렬에 0의 값을 가진 원소가 최소 3개가 존재하게 된다. 이후 각 열의 최소값을 찾아 앞선 연산을 반복한다. 이때 해당 열의 최소값이 0이라면 연산을 수행하지 않는다. 이렇게 얻은 축소된 cost 값 행렬의 원소 중 같은 행과 열을 공유하지 않는 0이 3개 있다면 최적 할당이 완료되고 알고리즘은 종료된다. 그렇지 않다면 0을 없애는 최소의 간선을 그어 포함되지 않는 원소 중 최소값을 찾은 후 앞선 연산을 반복한다. 이러한 헝가리안 알고리즘의 방법을 그대로 적용하여 중앙 제어기 회로를 설계한다면 필연적으로 많은 수의 덧셈기와 비교기가 필요할 것이다. 또한, 최소값을 빼주는 연산이 여러 번 반복될 수 있기 때문에 실시간성 확보도 어려워지게 된다. 따라서, 회로 합성시 면적(Area)와 신호 전달 지연(Delay)을 증가시킬 것이다. 따라서, 본 논문에서는 헝가리안 알고리즘 연산 절차를 그대로 회로로 구현한 회로와 미리 0 값을 갖는 원소의 분포를 Case 별로 정리한 다음 비교기만을 이용해 최적 할당을 수행하는 회로를 둘 다 구현해 비교해본다.

Core \ Data Packet	Data Packet0	Data Packet1	Data Packet2
Core0	C0	C1	C2
Core1	C3	C4	C5
Core2	C6	C7	C8

그림 1. 3 x 3 cost 행렬의 예시

III. 설 계

본 논문에서는 3 x 3 할당 회로를 Verilog HDL을 이용하여 설계하였다. 회로의 입출력 신호는 그림 2와 같다.

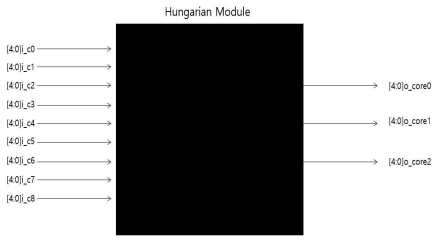

그림 2. 헝가리안 최적 할당 회로 모듈

cost 값을 0~9 단위로 normalize 하였기 때문에 입력신호를 5 bit로 선언하였다. 3 x 3 행렬의 C0부터 C8까지 순차적으로 4, 1, 2, 3, 6, 7, 1, 3, 9의 값을 입력하였을 때의 ModelSim 시뮬레이션 결과는 그림 3과 같다. 결과를 통해 Core0, Core1, Core2 에 각각 Packet2, Packet0, Packet1 이 할당되었으며 출력신호 o_core0, o_ core1, o_core2에 각각 소요되는 cost 값 2, 3, 3이 출력됨을 알 수 있다.

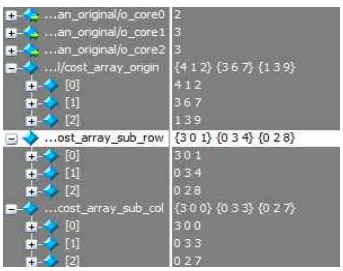

그림 3. ModelSim 시뮬레이션 결과

하지만, 합성 결과로부터 다수의 덧셈기가 합성되었고 신호 전달 지연도 큰 것으로 나타났다. 이러한 오버헤드를 줄이기 위하여 1차 연산 후 0의 분포 패턴을 분석해보았다. 그 결과 총 15가지 Case로 분류할 수 있었으며 0을 기준으로 매핑할 수 있는 6가지 Case (그림 4)와 1을 기준으로 매핑할 수 있는 9가지 Case (그림 5)로 구분하여 처리가 가능하다는 것을 알게 되었다. 그림에서 1은 0이 아닌 cost 값을 의미한다.

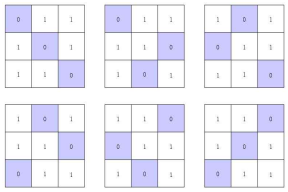

그림 4. 0을 기준으로 분류

1차 연산 후 그림 4 에 속한다면 0 인 지점들이 매핑 패턴이 된다. 반면 그림 5 에 속하는 경우라면 한번 더 연산을 수행하게 되면 결국 그림 4 의 그룹에 속하는 패턴이 나오게 된다. 그렇게 때문에 다량의 추가 덧셈기는 필요하지 않고 소수의 비교기만을 있으면 회로를 구현할 수 있게 된다.

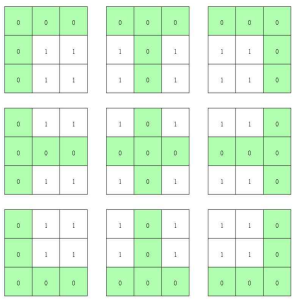

그림 5. 1을 기준으로 분류

Case 분류 방법을 이용한 최적 할당 회로 역시 Verilog HDL을 이용해 구현하였고 ModelSim을 이용한 시뮬레이션 결과 예시는 그림 6과 같으며 할당 결과는 그림 3에서의 할당 결과와 동일하였다.

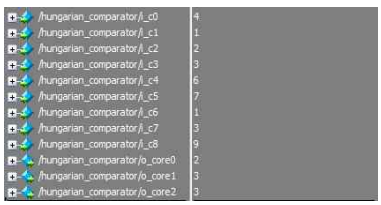

그림 6. Case 분류를 통해 설계된 회로의 ModelSim 시뮬레이션 결과

Ⅳ. 성능평가

TSMC 0.18μm cell 라이브러리를 이용하여 지금까지 설계한 두 종류의 회로를 논리 합성한 결과는 다음과 같다. 헝가리안 알고리즘의 연산 절차를 충실하게 구현한 하드웨어 회로(Original) 보다 Case 분석을 통하여 구현한 것이 면적은 약 16%, Propagation delay는 약 52% 감소한 것으로 나타났다.

표 1. 면적, 신호 전달 지연 비교

	Area	Delay
Original	71956	9467.27ps
Case 분류	60641	4553.25ps

Ⅴ. 결 론

본 논문은 멀티코어 프로세서에서 실시간으로 코어와 데이터 패킷간 최적 할당을 수행하기 위하여 헝가리안 알고리즘을 하드웨어 회로로 구현하였다. 구현 결과 알고리즘의 연산 절차를 가감없이 구현한 경우보다 Case 분석을 통해서 최적화 설계 구현을 하여야만 한 클럭 주기 내에 최적 할당 조합을 찾을 수 있다는 것을 알게 되었다. 만약 여러 클럭 주기에 거쳐 할당 조합을 찾게 되면 매 클럭 싸이클마다 데이터 패킷의 위치가 변하고 Busy 코어도 남은 수행 싸이클 수가 계속 변하기 때문에 그 조합은 최적 결과라 할 수 없을 것이다. 최근 양산되는 MPP 에 내장된 코어의 개수는 본 논문에서 가정한 코어의 개수보다 월등히 많기 때문에 향후에는 본 연구 내용을 토대로 좀 더 확장된 연구를 수행할 예정이다.

Acknowledgement

2022년 한국교통대학교 지원을 받아 수행하였음

References

[1] H. W. Kuhn, “The Hungarian method for the assignment problem,” *Naval Research Logistics Quarterly*, Vol. 2, Iss 1-2, pp. 83-97, Mar. 1955.

[2] X. Hu, Y. Zhang, L. Mao, J. Shen, and Z. Xing, “A Noc Centric Low Overhead Multi-chip Interconnection Technology,” *2021 IEEE 23rd Int’l Conf on High Performance Computing & Communications*, pp. 603-610, 2021.