CGRA 를 위한 전력이 고려된 어플리케이션 매핑에 관한 연구

윤종희, 김용주, 박상현, 조두산, 이종원, 김경원, 백윤흥
서울대학교 공과대학 전기컴퓨터 공학부
e-mail : jhyoon, yjkim, shpark, dscho, jwlee, kwkim@compiler.snu.ac.kr, ypaek@sun.ac.kr

A Study on Power-aware Application Mapping for CGRA

Jonghee W. Yoon, Yongjoo Kim, Sanghyun Park, Doosan Cho, Jongwon Lee, Kyungwon Kim, Yunheung Paek
School of Electrical Engineering, Seoul National University

요 약

최근에 응용프로그램의 복잡도가 증가함에 따라 이를 빠르게 처리하기 위하여 각종 멀티미디어 SoC 에서 Coarse Grained Reconfigurable Architecture (CGRA) 들이 사용되고 있다. CGRA 가 제공하는 병렬성을 극대화하기 위한 많은 어플리케이션 매핑 알고리즘이 연구되어 왔으나 CGRA 에서 소모되는 전력을 줄이기 위한 노력은 거의 없는 상태이다. 이러한 문제를 극복하기 위해 본 논문에서는 기존의 매핑 알고리즘을 기반으로 누설전력을 줄이기 위한 방법에 대해 다루고자 한다.

1. 서론

오늘날 소비자의 욕구를 만족시키기 위해 멀티미디어, 통신 등과 같은 어플리케이션의 복잡도가 빠르게 증가하면서 이를 수행하는 임베디드 시스템에 높은 성능을 요구하고 있다. 단일 프로세서를 사용할 경우 높은 주파수로 인한 전력소모 증가가 문제되고 다수의 프로세서를 이용하여 설계하기에는 어플리케이션의 효과적인 포팅과 넓은 실리콘 면적이 문제가 된다. 이러한 문제를 해결하기 위해 ASIC 을 이용한 방식이 많이 이용되고 있으나 높은 디자인 비용으로 인해 빠르게 업그레이드 되고 있는 어플리케이션에 대응하기 어렵다.

프로세서와 ASIC 방식의 대안으로 제시된 것이 바로 CGRA (Coarse Grained Reconfigurable Architecture)이다. CGRA 는 일반 프로세서에 비해 많은 수의 FU (Functional Unit)을 갖고 있기 때문에 높은 수준의 병렬 연산을 수행할 수 있다는 특징을 가지고 있으며 Configuration Memory 를 통해 각 FU 가 프로그램가능하기 때문에 ASIC 보다 유연성이 높아서 어플리케이션의 표준이 바뀌는 경우 하드웨어를 새로 디자인 할 필요가 없다는 장점이 있다. 그림 1은 일반적인 CGRA 의 구조를 보여준다. 여기서 FU 간의 interconnection 은 FU 사이에 직접적인 데이터 교환이 가능한 연결을 의미한다. 이종(heterogeneous) CGRA 인 경우 일부 FU 들은 multiplier 나 load/store unit 을 포함하게 된다.

CGRA 를 모바일 기기에 사용하게 될 경우 성능을 높이는 것뿐만 아니라 전력소모를 줄이는 것이 매우 중요하다. 지금까지는 CGRA 의 많은 FU 을 이용하여 효과적으로 병렬성을 높이기 위해 많은 어플리케이션 매핑 알고리즘[1], [2], [3], [4]이 연구되어 왔지만 전력소모를 줄이기 위한 노력은 거의 없는 상황이다. 논문[4]에서 유일하게 전력소모를 고려하였으나 각 FU 에 configuration 메모리의 크기가 1 word 씩만 할당되어 context 를 수행 중에 FU 이 한가지 operation 만을 수행 할 수 있기 때문에 어플리케이션의 operation 수가 FU 개수를 넘는 경우 매핑이 불가능한 spatial 매핑 방식을 기반으로 하고 있다는 제약이 있다.

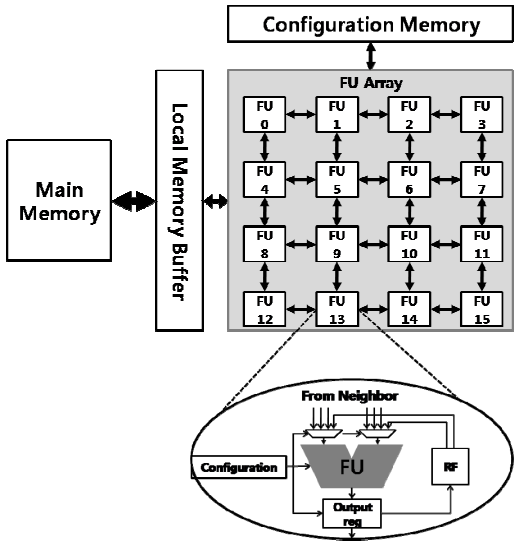

그림 1 CGRA 구조

본 논문에서는 더 큰 configuration 메모리를 이용하여 CGRA 에서 수행되는 보다 큰 어플리케이션을 매핑할 수 있는 temporal 매핑을 대상으로 한다. 다른 차이점은 기존 논문에서는 성능을 최대로 유지하는

한도 내에서 최소의 FU 에 매핑하여 누설전력소모를 최소화 하는 것에 비해 본 논문에서는 목표전력소모량을 만족시키도록 실시간 제약을 만족시키는 한도 내에서 사용되는 FU 의 수를 줄이는 방식을 취한다.

2. Power aware application mapping

이 논문에서 사용되는 기본 매핑 알고리즘은 가장 최근에 개발되었고 기존의 매핑 알고리즘에 비해서 더 빠르고 효율적인 방식인 EMS (Edge centric Modulo Scheduling)[3]이다. EMS 는 다양한 cost function 을 이용하여 최적의 매핑 값을 찾는 알고리즘이기 때문에 이 알고리즘자체에 새로운 cost function 을 추가하여 좀 더 효과적으로 누설 전력을 줄일 수 있으나 본 논문에서는 생략하도록 한다.

그림 2는 이 논문에서 제안하는 어플리케이션 매핑 방식을 나타낸다. 우선 일반적인 매핑 알고리즘에서와 같이 어플리케이션 C 코드로부터 추출한 DFG (Data Flow Graph)와 FU 배열 크기나 FU 간의 interconnection topology 와 관련된 CGRA 의 정보를 입력으로 받게 된다. 전력모드에 따라 목표전력소모량을 기초로 계산된 사용 가능한 FU 의 수가 주어지는 TPC (Target Power Consumption) 테이블을 이용하여 EMS 에 입력으로 들어가는 CGRA 의 FU 배열 크기가 조절된다. 이렇게 정의된 FU 수를 새로운 constraint 로 하여 EMS 매핑 알고리즘이 수행된다.

동종(homogeneous) CGRA 인 경우에는 필요한 개수의 임의의 FU 를 선택한 후 power gating 을 이용하여 누설전력을 제거할 수 있다. 그러나 이종 CGRA 의 경우에는 어떤 FU 를 선택하는지에 따라 전력소모감소량의 차이가 생기며, resource MII (Minimum Initiation Interval) 에도 영향을 주어 성능에 영향을 미친다. 따라서 누설 전력을 줄이기 위한 cost (C_l)가 최대가 되면서 resource MII 을 증가시키지 않는 범위 내에서 power gating 할 FU 를 선택해야 한다. c, c_m, $c_{l/s}$ 는 각각 일반 FU, multiplier 가 포함된 FU, load/store unit 이 포함된 FU 의 누설전력을 나타낸다. N_FU, N_FU_m, $N\_FU_{l/s}$ 는 각각 power gating 될 일반 FU, multiplier 가 포함된 FU, 그리고 load/store unit 이 추가된 FU 의 개수를 나타낸다. N_{m_t}, N_{l/s_t} 는 CGRA 가 포함하는 전체 multiplier 와 load/store unit 의 개수를 나타낸다. 마지막으로 N_m, $N_{l/s}$ 는 resource MII 를 증가시키지 않는 최소한의 multiplier 와 load/store unit 의 수이다.

$$C_l = max(c \cdot N\_FU + c_m \cdot N\_FU_m + c_{l/s} \cdot N\_FU_{l/s})$$
$$N_m \leq N_{m\_t} - N\_FU_m$$
$$N_{l/s} \leq N_{l/s\_t} - N\_FU_{l/s}$$

목표전력소모를 만족시키다 보면 실시간 제약을 만족시켜야 하는 어플리케이션의 경우에 정상적인 수행이 이루어지지 않을 수 있다. 따라서 EMS 에 의해 매핑된 결과는 실시간 제약을 만족시킬 수 있는지를 테스하는 루틴이 필요하게 된다. 이를 위해 본 논문에서 제안한 매핑에는 각 어플리케이션에 따라 PB (Performance Bound)를 적용하여 PB 이상의 성능을 보이는 매핑 결과를 사용할 수 있도록 피드백 룹이 포함되어 있다.

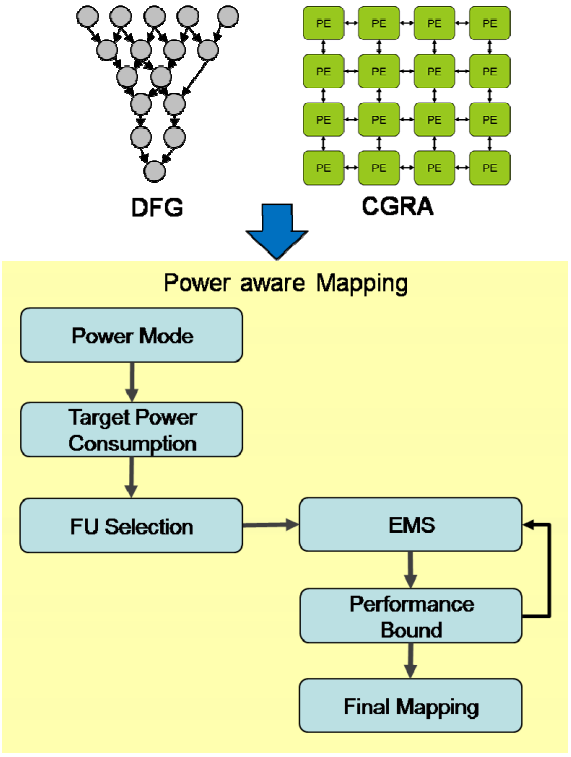

그림 2 전력을 고려한 어플리케이션 매핑

3. 결론

현재 많은 CGRA 를 위한 매핑 알고리즘이 개발되었다. 그러나 이들은 모두 CGRA 의 병렬성을 극대화하는 목적만을 가지고 있었기 때문에 전력소모를 줄이기 위한 노력이 없었다. 본 논문을 통해 실시간 제약을 만족시키는 한도 내에서 누설전력소모를 최소화하기 위한 매핑을 제시하였다. 앞으로 FU selection 과 관련하여 좀더 자세한 연구가 필요할 것으로 본다.

참고문헌

[1] J.-e. Lee, K. Choi, and N. D. Dutt. Compilation approach for coarsegrained reconfigurable architectures. IEEE Des. Test, 20(1):26–33, 2003.

[2] B. Mei, S. Vernalde, D. Verkest, H. Man, and R. Lauwereins. Dresc: A retargetable compiler for coarse-grained reconfigurable architectures, 2002.

[3] H. Park, K. Fan, S. A. Mahlke, T. Oh, H. Kim, and H.-s. Kim. Edge-centric modulo scheduling for coarse-grained reconfigurable architectures. In PACT '08: Proceedings of the 17th international conference on Parallel architectures and compilation techniques, pages 166–176, New York, NY, USA, 2008. ACM..

[4] J. W. Yoon, A. Shrivastava, S. Park, M. Ahn, R. Jeyapaul, and Y. Paek. Spkm: a novel graph drawing based algorithm for application mapping onto coarse-grained reconfigurable architectures. In ASPDAC'08: Proceedings of the 2008 conference on Asia and South Pacific design automation, pages 776–782, Los Alamitos, CA, USA, 2008. IEEE Computer Society Press.