韓國 OA學會 論文誌
第7卷 第2號, 2002. 6.

2002-7-2-1-5

Interconnection 최적화를 위한 연산 스케줄링에 관한 연구

신 인 수*

A Study on Operation Scheduling for Interconnection Optimization

In-Soo Shin*

요 약

본 연구는 상위수준합성에서의 데이터패스합성을 위한 스케줄링을 다루었다. 특히, 연산자 할당에 필요한 상호연결선을 최적화하기 위한 스케줄링 방법을 제안하였다. 또한 최적의 스케줄링 결과를 얻기 위하여 ILP 수식을 이용하였고 본 연구의 효용성을 검증하기 위하여 5차 디지털 웨이브 필터를 대상으로 실험한 결과 기존의 결과 보다 나은 결과를 나타내었다.

Abstract

In this paper, we deals with scheduling problem for data-path synthesis in high-level synthesis. We proposed a new method to optimize interconnection cost for operator allocation. Especially, we focus to optimize for buses to transfer data channel and register to restore operation result. Also we take account ILP formulation in order to get optimal scheduling result. To verify the effectiveness of this study, we select 5th Digital Wave Filter. The experimental result show that purposed method was better then the general methods.

* 주성대학 인터넷가상현실학과 전임강사

논문접수 : 2002. 4. 11
심사완료 : 2002. 6. 8

Ⅰ. 서론

상위수준합성(High-Level Synthesis)과정 중의 하나인 스케줄링(Scheduling)은 DFG(Data Flow Graph) 상의 연산을 주어진 조건이 만족되는 제어스텝으로 지정(Assign)하는 과정으로서 스케줄링 방식에 따라, Iterative/Constructive형 스케줄링과 Transformation 형 스케줄링으로 크게 구분되며 스케줄링의 목적에 따라 성능제약 스케줄링과 자원제약 스케줄링으로 구분된다.

일반적으로 스케줄링 방법들은, 스케줄링 단계에서 연산의 최적화에 우선 순위를 두어 연산의 최소화를 위한 스케줄링을 하게 된다.

데이터 전송에 이용되는 버스와 레지스터의 상호연결선(Interconnection) 비용을 감안하지 않는 스케줄링 결과는 결코 최적화 된 것이라 볼 수 없다. 따라서, 본 논문에서는 스케줄링 시 연산과 버스의 가용자원에 제약이 가해진 상태에서의 연산 스케줄링을 위한 방법을 제안한다.

Ⅱ. 스케줄링 알고리즘

ASAP/ALAP 스케줄링은, 아무런 제약조건이 가해지지 않은 상태에서 최대의 성능을 얻기 위해 기본적으로 행해지는 스케줄링으로써, 연산간의 선후관계가 유지되는 범위 내에서 DFG상의 모든 연산을 가능한 한 앞쪽 또는 뒤쪽 제어스텝에 할당시키는 방법이다.

1. 기본 관계식

1-1. 선후관계식

연산이 할당될 수 있는 스케줄링 범위를 나타내는 것으로서 연산 O_j의 각 요소가 할당될 수 있는 제어스텝이, 연산 O_i가 할당된 제어스텝 보다도 최소한 연산 O_i의 수행시간인 D_i 사이클타임 이상 떨어진 제어스텝 이어야만 한다.

$$\sum_{P=S_i}^{L_i} P*X_i(P-S_i+1) - \sum_{q=S_j}^{L_j} q*X_j(q-S_j+1) \le -D_i \quad \cdots\cdots (1)$$

1-2. 범위관계식

연산이 할당될 수 있는 스케줄링 범위를 나타내는 것으로, ASAP/ALAP 스케줄링에 의해 산출된 연산의 스케줄링 범위에서, 연산 O_i는 상한 제어스텝 S_i와 하한 제어스텝 L_i의 범위에 걸쳐 오직 한번만 할당되어야 한다.

$$\sum_{k=1}^{r_i} X_i(k) = 1, (i=1,2,..,n_t) \cdots\cdots (2)$$

1-3. 자원관계식

특정 제어스텝에 할당될 수 있는 연산의 개수와의 관계를 나타내는 자원관계식은, 제어스텝 P에 할당될 수 있는 연산유형 t인 연산의 총 개수는 사용 가능한 자원의 개수 N_t를 초과해서는 안된다.

$$\sum_{i=1}^{n_t} X_i(P-S_i+1) \le N_t, (P=1,2,..,T_{max}-Di+1) \quad \cdots\cdots (3)$$

1-4. 시간관계식

DFG 상의 연산 O_i의 각 요소가 할당되는 제어스텝은, 제어스텝의 최대치 Tmax를 결코 초과할 수 없다.

$$\sum_{P=S_i}^{L_i} P*X_i(P-S_i+1) \le Tmax-Di+1, (i=1,2,..,n_t) \quad \cdots\cdots (4)$$

2. 상호연결선과 레지스터

연산들 간에 데이터 전달을 위해 사용되어지는 버스와 저장장치는 스케줄링 과정에서 필히 고려하여야 할 사항 중 하나이다. 내부연결선은 와이어(Wire), 멀티플렉서(Multiplexer)와 버스(Bus)로 구성되고 저장 장치는 레지스터(Register), ROM 및 RAM 등의 요소로 구성되어 있다.

연산들 상호간에 직접적인 데이터 이동을 위해 사용되어지는 버스는 자원공유를 목적으로 사용되어진다.

임의의 제어스텝 P-1에서 P로 이동하는 연산 $O_i(k)$의 각기 다른 입력을 $B_i(k)$라 할 때, 제어스텝 P에서 요구되는 버스의 수 N_{bus}는 그 스텝에 할당되는 모든 연산의 입력 변수들의 수와 같아야 한다. 그러므로 스케줄링

과정에서 요구되는 버스의 수는 다음과 같은 관계식으로 표현할 수 있다.

$$\sum_{i=1}^{n_i} B_i(k) \le N_{bus}, (i=1,2,\ldots,T_{\max}-D_i+1) \quad \cdots\cdots\cdots\cdots\cdots\cdots (5)$$

그림 1은 스케줄링이 완료된 DFG의 일부분을 나타내었다. 그림 1의 (a)와 (b)는 동일한 수행을 하는 모델로, 요구되는 연산의 수는 ALU 2개, 곱셈기 1개로서 스케줄링의 결과는 같다. 그러나 연산간의 데이터 이동을 고려할 경우, 그림 (a)는 제어스텝 2로부터 제어스텝 3으로 이동하는 데이터의 수는 3개이고, 그림 (b)는 2개로서 모든 구간에 걸쳐 균등하게 분포되어 있다.

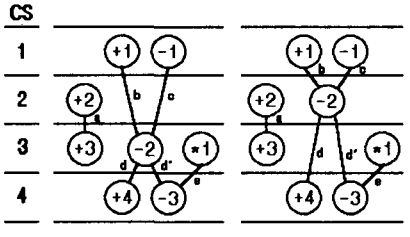


그림 1. DFG 모델
Fig. 1. A DFG model

스케줄링 단계에서 연산에 대한 버스의 최소화와 함께 고려할 사항으로, 데이터 저장을 위한 레지스터를 들 수 있다. 연산의 입력으로 사용되어 지는 버스의 자원공유를 위하여 변수를 임시로 저장하는 레지스터는, 하나 또는 여러 제어스텝에 걸쳐 존재하게 된다. 레지스터의 할당은 연산의 출력이 존재하는 제어스텝으로부터 후행 연산의 입력으로 사용하기 위해 데이터를 버스에 전달할 때까지이다. 이 기간을 변수의 라이프타임(Lifetime)이라 한다.

스케줄된 DFG로부터 요구되는 레지스터의 수는, DFG에서 변수의 이동이 가장 많은 제어스텝에서 변수의 합으로 구할 수 있다. 그러므로 레지스터의 최소화는 DFG 상에서 연산의 할당을 모든 제어스텝에 걸쳐 균등하게 분포된 결과가 최적의 해가 된다.

변수는 라이프타임 동안 하나의 레지스터에 할당되어야만 한다. 레지스터는 먼저 사용된 변수의 라이프타임과 중복되지 않는 범위 내에서는 하나의 레지스터를 다수의 변수가 공유할 수 있다.

그림 2는 그림 1에 대한 변수의 라이프타임을 나타낸 것이다. 그림 1에서, 요구되는 자원의 수와 제어스텝이 같은 동일한 스케줄링 결과를 얻었지만, '-2'의 연산을 제어스텝 2 또는 제어스텝 3으로 이동함에 따라 변수의 라이프타임이 바뀌게 된다.

표 1과 2는 그림 2의 라이프타임에 대한 레지스터의 할당과 자원공유를 나타내었다. 레지스터는 제어스텝을 경유하는 라이프타임이 중복(overlap)되지 않는 것을 묶어서 공유할 수 있다.

표 1과 2에서 요구되는 레지스터의 수는 각각 3개와 2개이다. 그러므로 그림 1에서, 그림 (a) 보다 (b)가 더 나은 스케줄링의 결과이다. 이와 같이 동일한 자원의 수가 요구되는 스케줄링의 결과를 얻어도, 특정 연산이 할당되는 제어스텝에 따라 라이프타임의 변화로 인하여 요구되는 버스나 레지스터의 수가 달라 지게된다. 따라서, 스케줄링 과정에서 내부연결선과 저장장치를 함께 고려한 스케줄링이 요구된다.

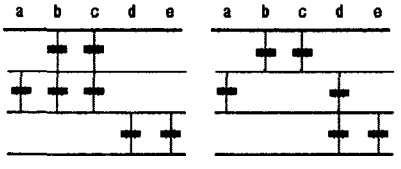


그림 2. 변수의 라이프타임
Fig. 2. Lifetime of variables

표 1. 그림 1(a)의 레지스터 할당
Table 1. Register assignment of Fig 1 (a)

레지스터	변 수	자원의 수
1	a	3개
2	b , d	
3	c , e	

표 2. 그림 1(b)의 레지스터 할당
Table 2. Register assignment of Fig 1(b)

레지스터	변 수	자원의 수
1	a , b , e	2개
2	c , d	

Ⅲ. 버스의 최적화

스케줄링 단계에서 제약조건을 연산의 최소화에 국한하지 않고, 연산의 할당에 따른 버스 및 레지스터의 수에 대한 사항도 충분히 고려해야만 목적함수를 최소화 할 수 있다.

버스는 임의의 연산 O_i의 결과로부터 최근접 후행 연산 O_j로 값의 전달을 위해 사용되어 지며, 선행연산으로부터 후행 연산의 입력 시점에서는 하나의 버스를 사용하여야 한다. 이때 해당 제어스텝에서 변수의 전달을 위해 사용되는 버스의 수는 오직 하나 뿐이며, 입력이 있는 버스는 연산의 입력으로 사용되기 이전에는 중복(overlap)되게 사용되어 질 수 없다.

연산간의 변수 전달을 위해 요구되는 버스 수의 최소화는 임의의 제어스텝에 할당되어 있는 연산자의 수와 데이터의 이동에 직접적으로 관계된다.

스케줄링 과정에서 요구되는 연산에 대한 버스의 최소화는, 모든 제어스텝 중, 연산의 입력이 가장 많은 제어스텝의 연산자의 수와 같다.

변수의 이동은 연산과 직접적인 관계가 있으므로, 각각의 입출력 수의 조합에 의해, 제어스텝에 할당된 연산수의 배수로 구할 수 있다. 임의의 제어스텝 P의 연산 O_i에 필요한 입력단자는 항상 두개의 입력을 가지므로 제어스텝 P에서 요구되는 버스의 수는, 그 스텝에 할당되어 있는 연산의 합에 두 배가된다. 그러므로, 스케줄링 과정에서, 변수의 이동에 따른 버스의 수는 다음의 관계식을 만족하여야 한다.

$$2*\sum_{i=1}^{n_1} X_i(P-S_i+1)-N_{bus}\le 0,(P=1,\ 2,\ \ldots,\ T_{max}-D_i+1) \quad \cdots\cdots (6)$$

앞에서 기술한 바와 같이, 변수가 최소한 하나 또는 여러 제어스텝에 걸쳐 존재하게 되므로, 연산 Oi의 출력으로부터 연산 O_j의 입력으로 사용되기까지, 변수의 라이프타임의 합인 LT(LifeTime)에 대한 관계식은 다음과 같이 나타낼 수 있다.

$$LT_i(j)=\max_{o_i\rightarrow o_j}(T_j-T_i-D_i) \quad \cdots\cdots (7)$$

여기서, T_i와 T_j는 ASAP/ALAP에 의해 주어진 연산 O_i와 O_j에 대한 스케줄링의 범위이다.

위에서 기술한 변수의 라이프타임인 LT와, 앞장에서 기술한 수식 표현을 이용하여 변수의 라이프타임의 최소화를 위한 관계식은 아래와 같이 나타낼 수 있다.

$$\sum_{P=S_i}^{L_i} P*X_i(P-S_i+1)-\sum_{q=S_j}^{L_j} q*X_j(q-S_j+1)+LT_i(j)=-D_i \quad \cdots\cdots (8)$$

스케줄링 단계에서 중요한 문제는 연산과 버스의 최소화에 있으므로, 레지스터의 최소화로 인한 연산과 버스에 대한 자원의 증가는 피해야 한다.

연산과 버스에 대한 레지스터의 비용함수(cost function)에 대한 관계는 다음과 같이 나타낼 수 있다.

$$C_1*FU+C_2*BUS-C_3*\sum_{i=1}^{n_1}(LT_i(j))\ >\ 0 \quad \cdots\cdots (9)$$

관계식 9에서 FU과 BUS는 스케줄링에서의 연산자와 버스를 의미하며, C1과 C2는 연산자와 버스의 비용이다. 그리고 C3는 변수의 이동에 필요한 요소인 레지스터의 비용이다.

관계식 6, 8 그리고 9에서, 버스의 할당은 연산의 지정과 직접적인 관계가 있고, 레지스터의 할당은 연산이 지정된 이후에 가능한 조건이 된다. 그러므로, 관계식 6에서 의미하듯이 연산과 버스의 자원 관계는 상호의존적이므로 스케줄링 과정에서 이 두 가지를 동시에 최소화 할 수 있으며, 관계식 9에 의해 레지스터에 비해 연산자와 버스의 비중이 높기 때문에 레지스터의 최소화로 인한 연산자와 버스의 증가는 피해야 한다.

그림 3은 26개의 가산기와 8개의 곱셈기로 구성된 5차 디지털 웨이브필터의 DFG를 나타낸 것이다.

각 연산의 왼쪽과 오른쪽의 숫자는 각각 ASAP과 ALAP스케줄링에 의해 산출된 스케줄링 범위를 의미한다. 그림 3의 DFG는 후행 연산의 결과가 선행 연산의 입력으로 사용되는 반복적인 루프 구조를 갖는다.

그림 4는 파이프라인형 곱셈기로 구현한 멀티사이클 연산에서 버스의 최소화와 라이프타임의 최소화에 대한 스케줄링의 결과이다. 그림 옆의 숫자는 해당 제어스텝에서 요구되는 곱셈기, 가산기, 버스, 그리고 레지스터의 수이다. 각 제어스텝에서 요구되는 곱셈기의 경우에는 파이프라인형 연산자를 사용하여 멀티사이클 연산을 수행함

으로 2 사이클타임일 때도 하나의 연산으로 간주하였다.

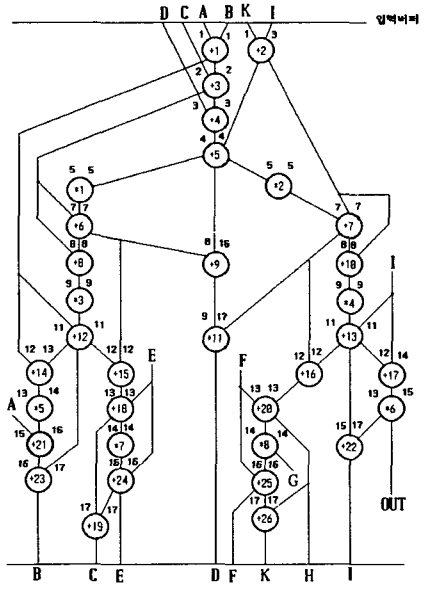
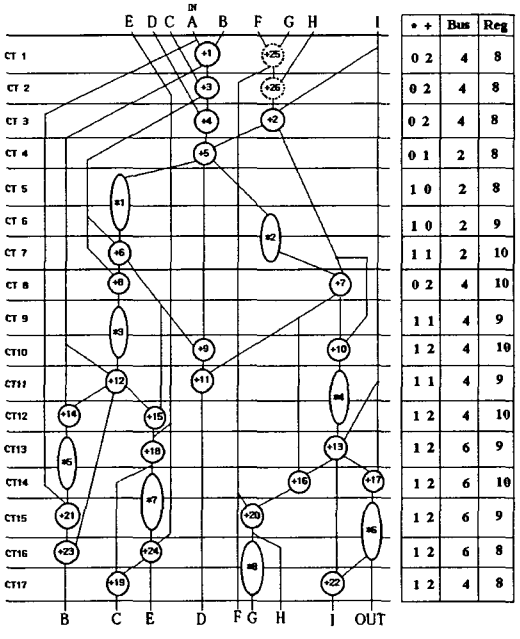


그림 3. 5차 디지털 웨이브필터
Fig. 3. 5th-Order Digital Wave Filter



* +	Bus	Reg
0 2	4	8
0 2	4	8
0 2	4	8
0 1	2	8
1 0	2	8
1 0	2	9
1 1	2	10
0 2	4	10
1 1	4	9
1 2	4	10
1 1	4	9
1 2	4	10
1 2	6	9
1 2	6	10
1 2	6	9
1 2	6	8
1 2	4	8

그림 4. 파이프라인 곱셈기에 대한 스케줄링 결과
Fig. 4. Pipelined Multipler Implementation

버스의 수는 각 제어스텝에 할당된 연산의 수에 두 배이며, 레지스터의 수는 해당 제어스텝에서 연산의 출력으로부터 다음 제어스텝으로 데이터의 이동이 있는 변수들의 합이다.

Ⅳ. 실험 및 결과

본 논문에서 제안한 스케줄링 방법의 효율성을 검증하기 위해서 벤치마크 모델(benchmark model)에 대한 스케줄링을 행하였다. 사용된 벤치마크 모델로서는 1988년 *High-Level synthesis workshop* 에서 표준 벤치마크 모델로 채택된 그림 3의 5차 디지털 웨이브필터(5th-order digital wave filter)를 택하였다.

실험결과는 공개된 자료가 가장 풍부하고 스케줄링 성능이 우수한 것으로 판정된, ALPS 시스템과 Spaid 시스템의 결과와 비교를 하였다. 모든 ILP 수식은 branch-and-bound방법을 사용함으로써 최적의 정수 해를 구하는, 선형프로그램(linear program)인 Lindo팩키지를 이용하여 SUN Spark 시스템에서 결과를 구하였다.

표 3. 파이프라인 곱셈기를 이용한 스케줄링결과
Table 3. Scheduling Result with Pipelined Multiplier

시스템 / 소요자원	Spaid	ALPS	SHIN
Adders	3	2	2
Multipliers	2	1	1
Bus	-	6	6
Loop length	17	17	17
DFG excution time	-	19	19

실험결과는 표 3과 같으며, 본 논문에서 제시한 방법에 의한 결과를 SHIN 이라고 표현하였다.

5차 디지털 웨이브필터를 실험모델로 선택한 대부분의 연구에서와 마찬가지로 "+" 연산의 지연시간은 40nS, "*" 연산의 지연 시간은 80nS, 제어스텝의 시간 간격은 50nS인 것으로 간주하였다.

표 3에 나타낸 파이프라인형 곱셈기를 이용한 스케줄

링의 결과로부터 알 수 있듯이, 본 연구의 결과는 ALPS 시스템에서 구현한 결과와는 동일한 자원의 효과를 얻었으며, Spaid시스템에서 구현한 결과보다는 더 나은 결과를 얻을 수 있었다.

V. 결 론

본 연구에서는 데이터패스 스케줄링에서의 비용함수 최소화를 위한 버스의 최적화 문제를 다루었다. 특히, 연산과 버스와의 관계를 고려한 스케줄링 기법과 스케줄링의 결과에 따른 레지스터의 최소화에 대하여 제안하였다.

기술된 기법의 정확성을 검증하기 위하여 표준 벤치마크 모델인 5차 디지털 웨이브 필터에 적용시킨 결과 기존의 연구결과와 일치함으로써 제시된 비용함수 최소화기법과 ILP수식이 정확함을 알 수 있었다.

본 연구에서는 버스의 할당과정과 레지스터의 최적화 과정이 별개로 수행됨으로써, 연산과 버스 그리고 레지스터간의 관계가 동시에 고려되지 않았다.

앞으로의 연구 과제는 완전한 데이터패스 합성이 이루어지기 위해서, 연산과 버스 그리고 레지스터를 동시에 고려한 새로운 연구가 요망된다.

참고문헌

[1] Carver Mead, Lynn Conway, " Introduction to VLSI System " Addison wesly Co. 1980

[2] S.Y.KUNG,T.KAILATH," VLSI and Modern Signal Processing ", Printice Hall, Inc., 1985

[3] Daniel D.Gajski,Nikil D.Dutt and Barry M. Pangrle " SILICON COMPILATION (TUTORIAL) " IEEE CUSTOM INTEGRATEDCIRCUITS CONFERENCE 1986, pp. 102-110

[4] Michael C. McFarland, SJ Alice C. Parker,Raul Camposano " Tutorial on High-Level Synthesis " IEEE Design Automation Conference., 1988, pp. 330-336

[5] Jiahn-Hung Lee, Yu-Chin Hau, Youn-Long Lin " A NEW INTEGER LINEAR PROGRAMMING IN DATA PATH SYNTHESIS" IEEE. 1989, pp. 20-23

[6] Cheng-Tsung Hwang, Jiahn-Hurng Lee, and Yu-Chin Hsu, " A Formal Approach to the Scheduling Problem in High Level Synthesis " IEEE. 1991, pp. 464-475

[7] 이근만, 임인칠, " 자원제약 조건하에서의 데이터패스 스케줄링", 대한통신학회, 1994

[8] 신인수, "ILP를 이용한 상위수준합성에서의 자원할당에 관한 연구", 박사학위논문,1999

저 자 소 개

신 인 수

1991년 청주대학교 전자공학과 (공학사)

1993년 청주대학교 대학원 전자 공학과 (공학석사)

1999년 청주대학교 대학원 전자 공학과 (공학박사)

현재 주성대학 인터넷가상현실학과 전임강사

관심분야 : 디지털시스템, High Level Synthesis, VLSI & CAD, Web Server 및 Programming