論文98-35C-3-3

역추적 시스토릭 어레이 구조 비터비 복호기의 파이프라인 합성

(A Pipeline Synthesis for a Trace-Back Systolic Array Viterbi Decoder)

鄭 熙 渡 *, 金 鍾 兌 *

(Hee-Do Jung and Jong-Tae Kim)

요 약

본 논문은 역추적 시스토릭 어레이 비터비 복호기를 설계함에 있어서 파이프라인 상위수준합성도구를 제시한다. 이 합성도구는 데이터흐름도 생성기와 상위수준합성기로 구성된다. 먼저 길쌈 부호기의 구성인자를 사양으로 하여 데이터흐름도를 VHDL로 생성한 후 파이프라인 방식의 스케쥴링과 할당을 수행한다. 이 합성도구는 설계영역을 효율적으로 탐사하여 주어진 제약조건을 만족하는 다양한 설계를 자동 합성하고 그 중 최적의 설계를 선택한다. 다양한 사양의 비터비 복호기를 합성하여 본 합성도구를 검증하였다.

Abstract

This paper presents a pipeline high-level synthesis tool for designing trace-back systolic array viterbi decoder. It consists of a data flow graph(DFG) generator and a pipeline data path synthesis tool. First, the DFG of the viterbi decoder is generated in the form of VHDL netlist. The inputs to the DFG generator are parameters of the convolution encoder. Next, the pipeline scheduling and allocation are performed. The synthesis tool explores the design space efficiently, synthesizes various designs which meet the given constraints, and choose the best one.

Ⅰ. 서 론

디지털 통신 시스템은 아날로그 시스템에 비해 통화의 고밀도화가 가능하여 이용자 수와 통화밀도를 개선할 수 있으며, 전력 소비량이 현저히 작기 때문에 단말기의 소형, 경량화가 가능하고, 보안의 확보와 데이터 전송 능력을 향상시킬 수 있다. 그러나 데이터 전송 중 전원의 순간적인 중단, 주파수의 혼란, 감쇠, 잡음 혹은 전송매질의 불규칙적인 전송 특성에 의해 발생되는 신호의 손상 현상인 페이딩(fading) 현상 등에 의한 채널 장애로 인하여 전송 데이터의 오류가 자주 발생하는데, 수신 측에서 수신한 정보가 송신 측이 전송한 정보와 같도록 하기 위해서 오류가 있는 정보를 수신했을 때, 수신 측이 그 오류를 검출하고, 정확한 정보를 얻을 수 있는 방법이 필요하다. 이것은 송신 측에서 디지털 데이터를 부호화하여 적절한 여분의 데이터를 첨부하여 전송하면, 수신 측에서는 이를 복호화 함으로써 채널 장애에 의한 오류 데이터의 정정이 가능하다. 따라서 디지털 이동통신 시스템에서는 전송 측에서의 채널 부호화 기술과 수신 측에서의 채널 복호화 기술이 중요한 문제가 되고 있는데, 이것을 해결하기 위한 채널의 부호화 방법으로는 기억(memory)의 존재 유무에 따라 블록 부호화(block encoding)와

* 正會員, 成均館大學校 電氣電子 및 컴퓨터工學部
(School of Electrical Engineering Sung Kyun Kwan University)

※ 본 연구는 서울대학교 반도체공동연구소의 교육부 반도체분야 학술연구조성비(과제번호 : ISRC96-E-2019)에 의해 수행되었습니다.

接受日字:1997年8月29日, 수정완료일:1998年2月26日

길쌈 부호화(convolution encoding)가 있으며 디지털 통신 시스템의 채널 부호화 방식으로서는 길쌈 부호가 많이 사용되고 있다. 이는 복호화에 많은 양의 메모리를 필요로 하지만, 블록 부호화에 비해 오류의 정정능력이 뛰어나기 때문이다.[1]

이러한 길쌈 부호의 채널 복호화 방법으로서는 길쌈 부호의 최대 유사 복호 알고리즘(Maximum likelihood decoding algorithm)인 비터비(Viterbi) 알고리즘[2]이 주로 이용되고, 이를 구현하는데는 일반적으로 레지스터 교환 방식(register-exchange method)과 역추적 방식(trace-back method)의 2가지 방법이 사용된다. 역추적 방식은 구속장(constraint length)이나 부호율(code rate)과 같은 길쌈부호기의 상태에 따라 그 구조가 결정되어지며, 현재 고집적회로(VLSI) 기술의 발전으로 ASIC(Application Specific Integrated Circuit)구현이 용이한 구조가 발표되고 있고[3][4], 데이터의 실시간 처리를 갖는 병렬처리 하드웨어가 설계되어 있다.[5][6] 이 비터비 복호기의 ASIC은 보통 전형적인 설계방법인 schematic editor를 사용하여 구현[7]되는데, 이것은 성능향상을 위한 수정이 요구될 때, 규격의 확장에 따른 설계 확장이 필요할 때, 새로운 통신 방식으로 인한 재설계 등이 요구될 때 능동적으로 대처할 수 없어 비효율적이며, 이러한 상황에서는 처음부터 다시 설계를 시도하여야 한다. 따라서 다양한 설계 방식과 설계 기술에 의해 구현될 수 있는 여러 가지 설계를 비교하고, 설계 변경이 용이하며, 설계 목적에 가장 적합한 설계를 할 수 있도록 상위 수준 합성(High Level Synthesis) 방식의 도입이 요구되어 진다. 일반적으로 상위 수준 합성은 시스템 단계에서 결정된 시스템 사양과 성능 분석된 알고리즘 또는 데이터 흐름도(data flow)로부터 주어진 제약조건들과 설계 목표들을 충족시키는 최적의 레지스터 전송 수준의 구조를 구현하는 것이다. 여기서 제약 조건들은 허용된 칩의 면적 즉 비용, 최대 전력, 또는 최대 지연 시간 등이며, 설계 목표들은 비용의 최소화, 지연 시간의 최소화, 설계 시간의 단축 등이 될 수 있다.

본 논문에서 제안되는 비터비 복호기 상위 수준 합성기의 기본구조로 고려되는 시스토릭 어레이 방식[5]은 레지스터의 배열 안에서 한 클락 주기의 입력지연 시간을 가지고 연속적으로 파이프라인 방식의 역추적 연산이 이루어지므로 다른 구조보다 빠른 복호가 이루어지나, 자원의 공유가 불가능하여 비용이 많이 든다. 따라서 자원을 제한하여 비용의 감소가 요구될 때는 사용되는 모듈의 수를 줄여야 하는데, 이는 입력지연 시간을 높여 모듈을 공유시킴으로서 해결할 수 있다.

본 논문에서는 비터비 복호기를 설계함에 있어서 이러한 제한 하에서 최저 비용이 드는 설계를 lower bound로 최고 속도의 설계를 upper bound로 하는 설계 영역을 효율적으로 탐사하여 설계자에게 가장 적합한 파이프라인 방식의 비터비 복호기를 합성하는 도구를 제시한다. 비터비 복호기의 합성기는 길쌈 부호기의 레지스터 수, 부호율, 생성계열, 각 층의 레지스터 수를 입력받아 역추적 방식 비터비 복호기의 구조에 대한 데이터흐름도를 생성하는 데이터 흐름도 생성기와 데이터흐름도를 입력받아서 합성을 수행하는 합성기의 두 부분으로 구성한다.

Ⅱ. 역추적 시스토릭 어레이 방식 비터비 복호기

본 논문에서 합성되는 비터비 복호기의 구조는 역추적 시스토릭 어레이 방식[5]을 사용하는데, 이러한 비터비 복호기의 구조가 그림 1에 나타나 있다. 이것은 구속장이 2이고 부호율이 1/2인 길쌈 부호기에 대한 비터비 복호기를 나타내는데, 이 구조에서 나타난 바와 같이 역추적 시스토릭 어레이 방식은 크게 선택모듈(selection unit)과 역추적모듈(trace-back unit)로 나눌 수 있으며, 선택모듈은 각 상태에서의 가지 메트릭 연산 모듈과 최소 메트릭을 가진 상태를 선택하는 모듈, 그리고 각 상태의 메트릭의 값을 정규화(normalization) 하는 모듈들로 구성되고, 역추적 모듈은 일련의 역추적 유니트로 구성되어져 있다.

가지 메트릭 연산 모듈은 격자도에서 각 상태에 연결된 가지의 부호어와 수신 심볼과의 차이를 구하고 이 차이와 이전의 값들을 누적하여 가지 메트릭을 계산하는데, 이는 부호율에 따라 그 복잡도가 결정되는데, 즉 한 상태에 들어오는 가지의 갯수는 부호율 R=m/n인 경우 2^m개가 되며, 부호율이 2/3일 경우 한 상태에는 2^2=4개의 가지가 들어오고 각 가지는 3비트의 정보를 가지고 있게 되는데 이러한 연산은 sel 모듈에서 수행되어지게 된다. 최소 메트릭의 상태를 결정하는 모듈은 각 상태의 가지 메트릭 값을 비교하여 최소의 메트릭을 가진 상태를 선택하고 그 상태의

값을 역추적 모듈에 전해주게 되는데, 선택 모듈의 mtdec모듈이 각 상태의 메트릭 값을 입력받아서 최소의 값을 갖는 상태를 결정하여 출력하게 된다. 부호화되는 정보 비트는 거의 반무한장이므로 레지스터에 저장되는 상태 메트릭의 값도 시간이 증가함에 따라 커지게 된다. 따라서 각 상태 메트릭 값을 조정하는 모듈이 필요한데, 정규화 모듈은 이전 단계의 상태 메트릭값 중에서 가장 작은 값을 이용하여 모든 상태 메트릭 값을 정규화 함으로서 상태 메트릭 값의 증가에 따른 레지스터의 데이터 비트 증가를 방지하여 회로의 규모를 줄일 수 있다. 이것은 i0 길쌈 부호기의 각 레지스터의 값은 0으로 초기화되어 부호화가 시작되므로 복호기도 상태 0에서 시작될 수 있도록 상태 0을 제외한 다른 모든 상태에 있어서 초기에는 상태 0의 상태 메트릭값 보다 훨씬 더 큰 값을 가져야 한다. 따라서 이러한 초기 값들이 각 상태에 할당될 수 있도록 하기 위해 i0norm에서는 0의 값으로 초기화된 값이 출력되며, i8norm에서는 7로 초기화된 값이 출력된다.

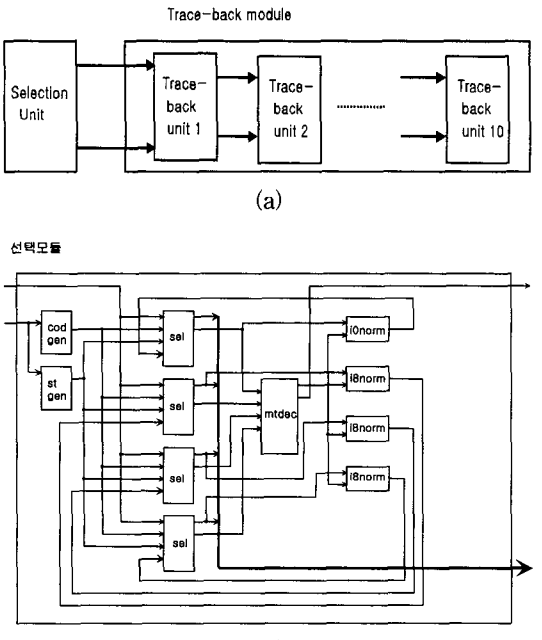

그림 1. (a) 시스토릭 비터비 복호기의 구조 (b) 선택모듈의 구조

Fig. 1. (a) The structure of systolic Viterbi decoder. (b) The structure of selection unit.

역추적모듈은 선택모듈에서 받은 역추적 값을 받아 그것을 역추적하여 복호된 정보 비트를 찾아내는데, 선택모듈에서 계산된 역추적 정보를 역추적 레지스터에 저장하고, 이것과 선택모듈에서 제공한 최소의 경로 메트릭을 갖는 상태를 이용하여 격자도의 상태중에서 가장 작은 상태 메트릭의 값을 갖는 한 시점전의 상태를 역추적하게 된다. 이 과정을 역추적 깊이만큼 순차적으로 반복하므로서 생존자를 역추적하게 되고 오류가 적은 원래의 정보를 복호한다. 역추적 방식은 초기 역추적 깊이만큼의 시간동안 역추적 정보를 저장하게되고, 그 후 역추적 깊이만큼 생존자를 역추적하기 때문에 초기 지연시간은 있으나 이 시간이 경과한 후에는 지연없이 매 클락마다 복호된 값을 출력하게된다. 이 역추적모듈은 파이프라인 형태의 레지스터의 배열로 구성되어서 각 반복 구간마다 역추적이 시행되는데, 역추적 길이의 결정은 에러의 정정 효율과 하드웨어 구현 효율을 동시에 고려하여 이루어져야 한다. 역추적 깊이 길이의 제한은 최대 유사 복호 출력의 결정을 저해하는 요인이 되는데, 보통 5K의 역추적 깊이의 길이가 많이 사용되고 있으며, 이 때 레지스터 배열의 갯수도 5K가 되어야 한다.

Ⅲ. 파이프라인 상위 수준 합성기

비터비 복호기의 합성기는 길쌈 부호기의 레지스터 수, 부호율, 생성계열, 각 층의 레지스터 수를 입력받아 역추적 방식 비터비 복호기의 구조에 대한 데이터 흐름도를 생성하는 데이터 흐름도 생성기와 데이터흐름도를 입력받아서 합성을 수행하는 합성기의 두 부분으로 구성되어 있다.

1. 비터비 복호기의 데이터 흐름도 생성

본 논문에서 구현하는 비터비 복호기 합성기의 입력으로 들어가는 데이터 흐름도는 역추적 시스토릭 어레이 구조를 기본 구조로 하여, 각 모듈을 structural design decomposition 방법으로 표현하였다. 이것은 그림 2에 나타난 역추적 시스토릭 어레이 구조의 비터비 복호기의 계층적 구조에서 가장 하위 레벨의 기본 모듈들을 먼저 설정하고, 이 모듈들의 behavioral 모델을 VHDL로 기술하여 라이브러리로 구현한 후, 이 기본 모듈들의 연결을 계층적 구조로 기술하므로서 구현될 수 있는데, 이러한 기술을 VHDL netlist로 표현함으로서 데이터흐름도를 구현하였다.

이렇게 구현된 데이터흐름도를 살펴보면, 각 모듈들

은 길쌈 부호기의 구속장, 부호율, 생성 계열, 그리고 각 층의 레지스터 수와 같은 인자들에 따라 그 구조가 유동적이며 그것들의 연결 구조도 변하게 된다. 따라서 이러한 인자들에 따라 각 모듈의 구조와 모듈들 간의 연결 관계에 대한 규칙성을 파악하여 이러한 인자들을 가지고 있는 길쌈 부호기를 복호화하는 적절한 비터비 복호기의 VHDL 모델을 구현할 수 있으므로, 비터비 복호기의 자동화가 가능하게 되는데 이것은 비터비 복호기의 VHDL 모델 자동 생성기로 구현되었다. 이 자동 생성기는 C언어를 이용하였으며, 길쌈 부호기의 구속장, 부호율, 생성 계열, 그리고 각 층의 레지스터 수와 같은 길쌈 부호기의 구성 인자를 입력으로 받아서 이 길쌈 부호기를 복호화하는 비터비 복호기의 각 모듈의 behavior와 모듈간의 netlist를 VHDL 모델로 출력한다. 이 자동 생성기는 길쌈 부호기의 전체 레지스터 수 K가 2개에서 9개까지, 부호율 R은 최대 7/8이 되는 길쌈 부호기에 대한 비터비 복호기의 VHDL 모델을 생성할 수 있으며, 이렇게 생성된 VHDL 모델은 VHDL 시뮬레이터를 이용하여 시뮬레이션 함으로서 역추적 시스토릭 어레이 구조 비터비 복호기의 데이터흐름도를 검증하였다. 사용된 VHDL 시뮬레이터는 Mentor Graphics의 quick HDL을 이용하였으며, 또한 이 모델은 합성이 가능하도록 구현되었다.

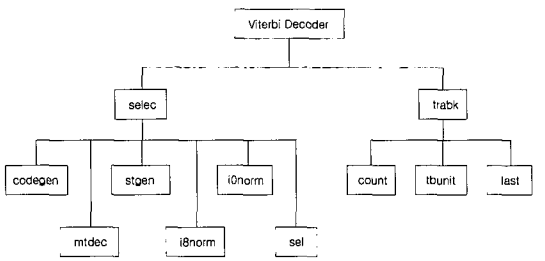

그림 2. 비터비 복호기의 계층적 구조
Fig. 2. Structural design hierarchy for the Viterbi decoder.

비터비 복호기의 VHDL 모델 자동 생성기의 출력 결과가 표 1에 나타나 있는데, 이 출력은 입력에 따르는 이름을 가진 1개의 파일로 나타나는데, 예를 들면 출력이 k2r12.vhd로 나타나면 이는 길쌈 부호기의 레지스터 수 K가 2이고, 부호율이 1/2인 경우를 나타낸다. 이 출력은 그림 2의 모든 계층 구조를 포함하고 있으며, 역추적 깊이는 기본적으로 5K를 사용하였고, m, n은 R=m/n으로 나타나는 부호율을 나타내며, 생성 계열의 입력은 나타내지 않았다.

VHDL netlist의 데이터 흐름도로서 비터비 복호기의 검증을 완료한 후, 합성기의 입력으로 들어가는 데이터 흐름도 그래프를 작성하였다. 이것은 directed acyclic graph, G = (V, E)의 형식을 갖는데, 여기서 node, V는 역추적 시스토릭 어레이 비터비 복호기의 기본모듈을 나타내며, directed edge, E는 기본모듈 사이의 연결과 의존성을 나타낸다. 이것도 역시 C언어를 이용하여 길쌈 부호기의 구속장, 부호율, 생성 계열, 그리고 각 층의 레지스터 수와 같은 인자들을 입력으로 받아서 자동적으로 이 입력에 적합한 directed acyclic graph의 형식이 출력되는 프로그램을 작성하여 구현되었는데, VHDL 모델 자동 생성기와 같은 자료구조를 가진다.

표 1. 입력에 따른 출력 VHDL 모델의 주요 형태
Table 1. The main form of VHDL model according to the input.

입력 (K, m, n)	블록의 수			라인 수	모델 생성 시간 (sec)
	i8norm	sel	tbunit		
2, 1, 2	3	4	10	600	0.017
3, 2, 3	7	8	15	770	0.017
4, 1, 3	15	16	20	920	0.033
4, 3, 4	15	16	20	1210	0.133
5, 3, 5	31	32	25	1870	0.283
7, 2, 5	127	128	35	4240	0.367
8, 1, 2	255	256	40	5800	0.250

2. 파이프라인 방식 비터비 복호기의 상위 수준 합성기

상위 수준의 합성은 스케줄링과 할당(allocation)의 두 단계로 나눌 수 있는데 스케줄링은 주어진 behavior의 연산이 수행될 제어구간(control step)을 결정하는 과정이고 할당은 연산을 연산자에 할당하고, 변수를 레지스터에, 그리고 연산기들과 레지스터들을 버스나 멀티플렉서를 이용하여 연결하는 과정을 의미한다. 이러한 합성에 있어서, 파이프라인은 하나의 연산처리 과정을 여러 단계로 나누고, 각 단계를 처리하기 위한 하드웨어 유니트들을 독립적으로 구성하여 동시에 동작하도록 하는 것으로 한 입력 데이터는 파이프라인의 첫 번째 유니트로 들어와서 지정된 연산이

수행된 후에 다음 유니트로 보내진다. 이 과정이 각 유니트를 통하여 순서대로 처리되고, 마지막 유니트에서의 연산까지 수행되면 그 데이터에 대한 전체 연산 과정이 완료되어지며, 파이프라인이 아닌 연산 장치에 비해 많은 속도 향상을 이룰 수 있다. 본 논문에서는 이러한 파이프라인 방식의 스케줄링 알고리즘으로 주어진 비용과 입력지연시간을 이용하여 자원을 할당하고 스케줄링을 수행하는 Sehwa[8]와 이것을 이용한 기존의 방법[9]을 기반으로 하여 역추적 시스토릭 어레이 방식 비터비 복호기의 상위 수준 합성기를 구현하였다.

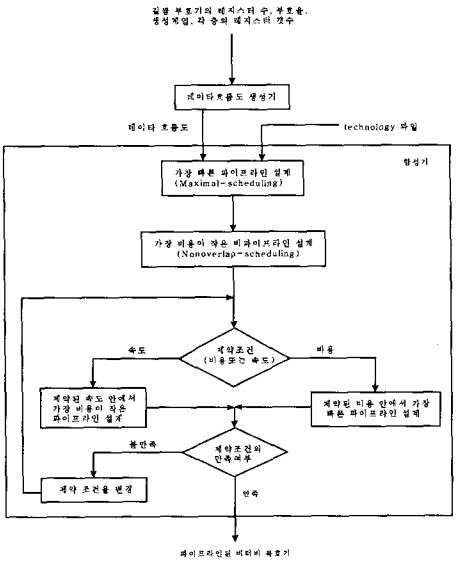

그림 3. 파이프라인 방식 비터비 복호기의 상위 수준 합성기의 흐름도
Fig. 3. The flowchart of pipeline high-level synthesizer for Viterbi decoder.

그림 3은 이 합성기의 합성 과정을 나타내고 있는데, 길쌈 부호기에 대한 인자들을 입력하면, 데이터 흐름도가 생성되고 이것과 데이터 흐름도의 각 모듈, 멀티플렉서, 레지스터의 비용과 지연시간에 대한 정보를 가지고 있는 technology 파일이 파이프라인 합성기에 입력되면 설계 영역이 결정되며, 제약조건이 입력되면 합성기는 이 제약 조건을 만족시키는 파이프라인 구조를 탐색하여 그 결과를 출력한다. 여기서는 비용과 입력지연시간이 제약조건으로 입력될 수 있다. 합성기는 각 모듈을 최대한으로 중첩하여 수행되어 질 수 있도록 하기 위해 먼저 적절한 클락 주기를 선택하고, 그 후 maximal-scheduling, nonpipeline-scheduling, feasible-scheduling의 스케줄링 알고리즘[8]을 사용하여 스케줄링을 수행한다.

Maximal-scheduling 알고리즘은 최대 stage-time의 제한만을 가지고 가능한 짧게 데이터흐름도를 스케줄하는데 비용의 제한을 고려하지 않기 때문에 각 연산은 각각의 연산자에 할당되며, 따라서 데이터 흐름도의 파이프라인 구현시 입력지연시간이 1이 되므로 가장 빠른 수행 속도를 갖는다.

Nonpipeline-scheduling 알고리즘은 최대 stage-time과 전체 사용 모듈 수의 제한 하에서 가능한 짧게 데이터흐름도를 스케줄하며 어떠한 중첩도 이루어지지 않기 때문에 모든 time-step에서 자원의 공유가 가능하다.

Feasible-scheduling 알고리즘은 정해진 입력지연시간과 최대 stage-time 제한을 가지고 데이터흐름도를 스케줄하고, 파이프라인 구현시의 전체 비용이나 또는 최소한의 요구되는 성능에 제한을 가진다. 비용이 제한된 합성의 경우에는 요구되는 모듈의 각 타입의 적절한 수를 선택하는데 여기서 사용되는 모듈들의 비용이 제한된 비용을 초과해서는 안된다. 그러나 최종적인 비용은 스케줄링이 수행될 때까지 결정할 수가 없으며, 최적의 합성 결과는 비용이 제한된 스케줄링이 수행된 모든 설계들을 비교하여 결정된다. 비용이 제한된 합성에서 가장 작은 입력지연시간은 더 큰 입력지연시간을 가진 설계보다 더 빠른 성능을 가지기 때문에 가능한 한 최소의 입력지연시간부터 출발하게 되는데, 만약 적절한 설계가 발견되지 않으면, 입력지연시간은 증가되고 새로운 입력지연시간에 따라 가능한 stage-time은 다시 고려한다. 입력지연시간을 늘이면 더 많은 자원의 공유가 가능하고, 전체의 비용은 줄어들게 되며, 이러한 과정이 적절한 설계를 찾을 때까지 반복된다. 성능이 제한된 합성은 제한되는 최소한의 성능을 충족하면서 가장 비용이 작은 설계를 합성한다. 이것은 먼저 최대한의 입력지연시간을 가지고 출발하는데 만약 적절한 설계가 발견되지 않으면, 즉 제한되는 성능을 충족시키지 못하면 입력지연시간은 감소된다. 입력지연시간이 줄어들면 스케줄링시에 더 많은 모듈이 사용되며, 성능은 향상된다. 이 과정을 반복하여 제한 조건을 만족하는 설계를 찾게 되는데, 이러한 설계들이 찾아지면 다시 입력지연시간을 증가시키고, 더 작은 비용을 갖는 해를 찾기 위해 설계된 비

용을 제한 조건으로 하여 비용이 제한된 합성을 수행한다. 현재 설계된 비용이 제한조건으로 사용되기 때문에 비용이 제한된 합성을 사용하면 더 비용이 작은 설계를 발견할 수도 있다.

C언어로 Sehwa를 구현한 합성기[9]는 모든 합성 과정이 끝난 후에 합성 결과를 비교하기 때문에 많은 메모리가 필요로 하며, 모든 stage 시간에 대해 합성을 수행하므로 합성기의 실행 시간이 길었다. 따라서 이러한 합성기를 구속장이 큰 비터비 복호기의 합성에 적용하기는 많은 메모리와 시간을 요구한다. 본 논문에서 제시한 합성기는 한번의 합성이 수행될 때마다 이전의 결과와 비교하여 좋지 못한 합성결과를 버림으로서, 메모리의 사용을 줄였고, 적절한 합성 결과가 나올 수 있는 stage 시간에 대해서만 합성을 수행하도록 수정하여서 합성기의 실행 속도를 높일 수 있었다.

Ⅳ. 실험 결과

그림 4에는 표1에서 레지스터수가 4이고 부호율이 1/3인 길쌈 부호기에 대한 비터비 복호기의 VHDL 모델을 Mentor의 quickHDL을 이용한 시뮬레이션 결과를 나타내었다. 이것은 먼저 임의의 정보비트를 부호화한 부호어에 대해 무작위의 오류를 발생시켜서 이러한 오류가 포함된 부호어를 비터비 복호기의 입력으로 받아서 복호화하는 시뮬레이션을 수행하였다. 여기서 it는 오류가 포함된 수신 부호어이며, 복호되는 출력 값은 decd로 나타나 있다.

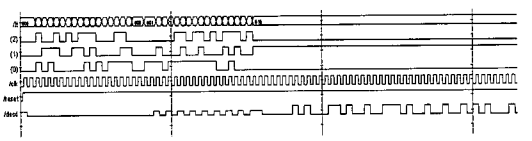

그림 4. K=4, R=1/3인 비터비 복호기의 시뮬레이션 결과

Fig. 4. Simlation result for the viterbi decoder. (K=4, R=1/3)

본 논문에서 제시한 합성기의 실험을 위해 모듈 타입에 대한 비용과 지연시간의 상대적인 값을 표 2와 같이 가정하였다. 이것은 합성기의 입력으로 들어가는 technology 파일의 내용을 나타내며, 멀티플렉서와 레지스터의 비용은 고려하지 않았다. 표 2에 나타난 비터비 복호기를 이루는 각 모듈들 중에서 sel, i8norm은 구속장에 따라 그 수가 결정되고, tbunit는 역추적 깊이에 따라 그 수가 결정되어지며, 나머지 모듈들은 1개만이 존재하므로, sel, i8norm, tbunit 모듈들만이 입력지연시간에 따라 공유가 가능하게 된다.

표 2. 비터비 복호기 각 모듈의 비용과 지연시간

Table 2. Cost and delay of each module of Viterbi decoder.

Level 1	Level 2	Level 3	cost	delay(ns)
Viterbi Decoder	selec	codegen	5	10
		stgen	5	10
		sel	10	30
		mtdec	10	20
		i0norm	8	20
		i8norm	8	20
	trabk	count	7	20
		tbunit	8	20
		last	5	10

표 3. 비터비 복호기의 합성 결과

Table 3. The result of synthesizing Viterbi decoder.

	입력	사용된 모듈의 수			입력지연시간 / 비용	stage 수
		sel	i8norm	tbunit		
K2R12	MAX	4	3	10	30 / 184	13
	NON	1	1	1	480 / 66	16
	100(s)	2	1	4	90 / 100	14
	170(s)	1	1	2	150 / 74	16
	100(c)	3	1	4	90 / 100	14
K3R23	MAX	8	7	15	30 / 296	18
	NON	1	1	1	750 / 66	25
	200(s)	2	2	3	180 / 110	21
	150(c)	3	3	5	90 / 134	20
	250(c)	4	4	8	60 / 176	19
K5R35	MAX	32	31	25	30 / 808	28
	NON	1	1	1	1950 / 66	65
	100(s)	11	11	9	90 / 310	30
	200(s)	6	6	5	180 / 188	33
	150(c)	4	4	4	240 / 144	35
K7R25	MAX	128	127	35	30 / 2616	38
	NON	1	1	1	7710 / 66	257
	100(s)	43	43	12	90 / 910	40
	300(s)	13	13	4	300 / 306	47
	1000(c)	43	43	12	90 / 910	43
K8R12	MAX	526	525	40	30 / 4960	43
	NON	1	1	1	15390 / 66	513
	100(s)	176	176	14	90 / 1692	45
	500(s)	33	33	3	480 / 352	58
	1000(c)	88	88	7	180 / 870	48

표 3에는 구속장이나 부호율에 따른 여러 가지의 비터비 복호기의 구조에서 제약 조건이 주어질 때 적절한 비터비 복호기의 파이프라인 합성 결과를 나타내고 있는데, 이 세 개 모듈들의 사용갯수를 나타내어 입력지연시간에 따른 모듈의 갯수를 비교하였다. 표 3에서 K2R12는 길쌈부호기의 레지스터수가 2, 부호율이 1/2임을 나타내며, 입력은 비용이나 입력지연시간이 되는데 괄호안의 기호가 c는 비용을 나타내고 s는 입력지연시간을 표시한다. maximal-scheduling시 합성되는 결과는 입력에 MAX로 나타내며, NON은 nonoverlap-scheduling시 합성되는 결과를 나타낸다.

V. 결 론

ASIC 설계에 있어서 가장 필수적으로 요구되는 것은 개념(idea)또는 시스템 사양으로부터 칩을 생산하기까지 설계에 필요한 시간을 최소한으로 줄이고, 또한 가능한 한 다양한 설계방식과 구현방법들을 상위수준에서 비교하여 보다 적절한 설계를 구현하는 것이다.

비터비 복호기를 설계함에 있어서는 매 클락마다 입력을 받는 시스토릭 어레이 구조는 비록 속도는 빠르나 비용이 많이 들므로, 작은 비용의 설계가 요구될 때는 사용되는 모듈 수를 줄어야 한다. 이것은 입력지연시간을 조정하므로서 제약조건을 만족하는 적절한 설계를 찾는 일이다. 본 논문에서는 이러한 주어진 제약조건에서 파이프라인 방식의 다양한 설계영역을 상위단계에서 효율적으로 탐사하여 입력지연시간과 비용이 다른 여러 가지 설계를 짧은 시간 안에 비교하므로서 최적의 설계를 가능하게 한다.

또한 길쌈 부호기의 여러 가지 사양에 따른 비터비 복호기의 데이터 흐름도인 VHDL 모델의 시뮬레이션 결과를 쉽게 비교할 수 있으므로 보다 효과적인 비터비 복호기의 설계가 가능할 수 있을 것으로 기대된다.

참 고 문 헌

[1] 김만영, *부호이론*, 희중당, 서울, pp. 277-315, 1979

[2] G. C. Jr and J. B. Cain, *Error-Correction Coding for Digital Communications*, Plenum, New York, pp. 227-266, 1981.

[3] Charles M. Rader. "Memory Management in Viterbi Decoder", *IEEE Trans. Comm.*, vol. COM-29, no. 9 pp. 1399-1401, Sept. 1981.

[4] Jens Sparso. Henrik N. Jorgenson, "An Area-Efficient Topology for VLSI Implementation of Viterbi Decoders and Other Shuffle-Exchange Type Structures", *IEEE Jr. Solid-State Circuit*, vol. SC-26, no. 2 pp. 90-96, Feb. 1991.

[5] T. Truong, M. Shih, I. Reed, E. H. Satorius, "A VLSI Design for a Trace-Back Viterbi Decoder", *IEEE Trans. Comm.*, vol. 40, no. 3 pp. 616-624, March 1992.

[6] C. Chang, K. Yao, "Systolic Array Processing of Viterbi Algorithm", *IEEE Trans. Inform. Theory*, vol. IT-35, no. 1 pp. 76-86, Jan. 1989.

[7] 차진종, 현진일, 강인, 김재석, 김경수, "CDMA 이동통신 시스템용 비터비 복호기 ASIC 설계 및 구현", *전자공학회 논문지*, 제33권 A편 제1호, pp. 139-152, January 1996

[8] N. Park and A. Parker "Sehwa: a software package for synthesis of pipelines from behavioral specification" *IEEE Trans. CAD*, vol. 7, no. 3 pp. 356-370, March 1988.

[9] Woohyun. David Lee "Design and Implementation of High-Level Synthesis of Pipelined Data-path and Clocking Scheme", *Master's thesis*, Dept. of ECE, University of California, Irvine, June 1992.

저 자 소 개

鄭 熙 渡(正會員)

1996년 성균관대학교 전기공학과졸업(공학사). 1998년 성균관대학교 대학원 전기공학과 공학석사. 1998년 1월 ~ 현재 삼성전자 재직. 주관심분야는 통신 및 DSP용 VLSI 설계

金 鍾 兌(正會員)

1982년 성균관대학교 전자공학과졸업(공학사). 1987년 미국 캘리포니아대학(Irvine) 전기 및 컴퓨터공학과 졸업(공학석사). 1992년 미국 캘리포니아대학(Irvine) 전기 및 컴퓨터공학과졸업(공학박사). 1991년 ~ 1993년 미국 The Aerospace Corporation 연구원. 1993년 ~ 1995년 전북대학교 컴퓨터공학과 교수. 1995년 ~ 현재 성균관대학교 전기전자 및 컴퓨터공학부 교수. 주관심분야는 VLSI CAD, ASIC 설계, 컴퓨터구조