VHDL을 이용한 속도 독립 회로의 기술과 합성

정 성 태†

요 약

기존의 속도 독립 회로 합성 시스템에서 사용되는 기술 방법들은 각각 특정한 설계 양식과 합성 방법에 적합하도록 만들어졌기 때문에 표준화된 기술 방법으로 채택되지 못하고 있다. 본 논문에서는 하드웨어 기술을 위한 표준 언어인 VHDL을 이용하여 속도 독립 회로를 기술하고 합성하는 방법을 제안한다. VHDL은 광범위한 언어이므로 본 논문에서는 속도 독립 회로의 기술과 합성에 이용될 수 있는 VHDL 부집합을 정의한다. 그리고 VHDL로 기술된 회로 명세를 신호 전이 그래프로 변환한 다음에 기존의 합성 알고리즘을 이용하여 속도 독립 회로로 합성한다. 이를 위하여 각각의 VHDL 문을 부분적인 신호 전이 그래프로 변환하고 부분적인 신호 전이 그래프들을 합병함으로써 VHDL 프로그램을 신호 전이 그래프로 변환하는 체계적인 방법을 제안한다. VHDL을 이용함으로써 시뮬레이션, 테스팅 등 기존의 VHDL 기반의 다양한 설계 프로그램들과 속도 독립 회로 합성 프로그램을 통합하는 프레임워크 개발이 가능하게 되고 기존의 회로 설계자들이 쉽게 비동기 회로에 접근할 수 있게 되는 장점이 있다.

Specification and Synthesis of Speed-independent Circuit using VHDL

Sung-Tae Jung†

ABSTRACT

There are no standard language for the specification of speed-independent circuits because existing specification methods are designed appropriately to each synthesis methodology. This paper suggests a method of using VHDL, a standard hardware description language, for the specification and synthesis of speed-independent circuits. Because VHDL is a multi-purpose language, we define a subset of VHDL which can be used for the synthesis.

We transform the VHDL description into a signal transition graph and then synthesize speed-independent circuits by using a previous synthesis algorithm which uses a signal transition graph as the specification method. We suggest a systematic transformation method which transforms each VHDL statement into a partial signal transition graph and then merges them into a signal transition graph. This work is a step towards to the development of an integrated framework in which we can utilizes the existing CAD tools based on VHDL. Also, this work will enable a easier migration of the current circuit designers into asynchronous circuit design.

1. 서 론

반도체 공정 기술이 발전함에 따라 고속의 대규모 디지탈 시스템이 출현되고 있다. 그런데, 근래에 들어 동기 방식의 대규모 디지탈 시스템의 설계에 있어서 전역 클럭(global clock)이 제한 요소로 대두되고 있다. 이는, 반도체 소자의 속도가 빨라지는 것에 비례하여 선의 신호 전달 속도가 빨라지지 않고 클럭이 시스템의 각 구성 요소에 동시에 전달되지 않는 클럭 스큐

※ 이 논문은 1997년 한국학술진흥재단의 공모과제 연구비에 의하여 수행되었음.
† 정 회 원 : 원광대학교 컴퓨터공학과 교수
논문접수 : 1998년 9월 30일, 심사완료 : 1999년 5월 25일

(clock skew) 현상을 제거하기가 어렵기 때문이다. 이러한 문제를 해결하기 위한 한 방법으로 전역 클럭을 사용하지 않는 비동기 회로(asynchronous circuit)에 대한 연구가 근래에 들어 활발히 진행되고 있다.

비동기 회로는 클럭을 사용하지 않으므로 클럭 스큐 문제가 없고 평균 속도로 동작할 수 있으며 전력 소모가 적은 회로 구현에 이용될 수 있는 등 여러 가지 장점을 가지고 있다. 그러나 비동기 회로는 의도하지 않은 신호의 변화인 해저드(hazard)로 인하여 비동기 회로의 설계가 동기 회로(synchronous circuit)에 비해 훨씬 더 어렵다는 문제를 가지고 있다. 따라서, 최근에 들어 상위 단계의 기술로부터 비동기 회로를 자동으로 합성하는 여러 가지 방법들이 제안되고 있다[1,2,3,4,5].

기존의 비동기 회로 합성 시스템에서는 신호 전이 그래프[6], 비동기 유한 상태기[7], CSP(Communicating Sequential Processes)[8] 등이 회로 기술에 사용되고 있다. 그러나 이들은 각각 특정한 설계 양식과 합성 방법에 적합하도록 만들어졌기 때문에 통일된 표준 기술 방법이 없는 실정이다.

이에 반하여 동기 회로의 합성과 시뮬레이션 등에 사용되는 대부분의 상용 프로그램들은 표준 하드웨어 기술 언어인 VHDL(Very high speed integrated circuit Hardware Description Language)[9]을 사용하고 있고 일관성있는 회로 기술 방법을 채택하고 있다. 이로 인하여 회로 설계가 용이해졌고 설계 도구들 사이의 호환성이 높아져서 많은 설계 기술들이 축적될 수 있게 되었다.

그러나 기존의 속도 독립 회로 합성 시스템에서는 독자적인 회로 기술 방법을 사용함으로써 VHDL 기반의 잘 개발된 소프트웨어를 활용할 수가 없었다. 따라서, 본 논문에서는 기존의 VHDL 기반 소프트웨어의 활용을 가능하게 하기 위하여 VHDL을 이용하여 속도 독립 회로를 기술하고 합성하는 방법을 제안한다. VHDL은 시뮬레이션, 테스팅, 합성 등 다양한 용도로 사용될 수 있는 언어이므로 본 논문에서는 속도 독립 회로의 기술과 합성에 이용될 수 있는 VHDL 부집합을 정의한다. 그리고 VHDL로 기술된 회로 명세를 신호 전이 그래프로 변환한 다음에 기존의 합성 알고리즘을 [14,15] 이용하여 속도 독립 회로로 합성한다. 이 알고리즘은 신호 전이 그래프를 입력받아 그래프의 신호 전이들 사이의 관계를 이용하여 속도 독립 회로를 합성한다. VHDL 프로그램으로부터 신호 전이 그래프로의 체계적인 변환 방법으로 본 논문에서는 각각의 VHDL 문을 부분적인 신호 전이 그래프로 변환하고 부분적인 신호 전이 그래프들을 합병하는 방법을 제안한다. VHDL을 이용함으로써 시뮬레이션, 테스팅 등 기존의 VHDL 기반의 다양한 설계 프로그램들과 속도 독립 회로 합성 프로그램을 통합하는 프레임워크 개발이 가능하게 되고 기존의 회로 설계자들이 쉽게 비동기 회로에 접근할 수 있게 되는 장점이 있다.

관련된 연구로는 미국 Utah 대학의 비동기 그룹에서 진행하고 있는 비동기 회로의 합성을 위한 통합된 프레임워크 개발이 있다[10]. 이 시스템에서는 하드웨어 기술 언어로서 널리 사용되고 있는 Verilog에 채널에 대한 사양을 추가한 Verilog+를 입력 언어로 사용하였다. 그러나 이 언어는 표준 언어가 아니고 VHDL에 비하여 이용도가 낮다.

영국의 Menchester 대학의 AMULET[11] 비동기 마이크로 프로세서 연구팀은 Sutherland에 의해 제안된 마이크로 파이프라인[12] 구조의 회로 모델에 기반하여 행위 단계의 VHDL 모델로부터 구조적 VHDL 모델을 생성하는 합성기를 개발하였다[13]. VHDL을 이용하여 마이크로 파이프라인 기반의 비동기 시스템을 효과적으로 기술할 수 있음을 보였고 행위 단계의 VHDL 모델이 단순한 변환 규칙에 의하여 구조적 VHDL 모델로 합성될 수 있음을 보였다. 그러나 이 방법에서는 시뮬레이션을 위하여 구조적 VHDL 모델을 생성하였지만 회로를 합성하는데 VHDL을 이용하지는 못하였다.

이와 같이 본 논문에서는 기존의 합성 시스템[14, 15]을 확장하여 VHDL로부터의 합성할 수 있도록 함으로써 기존의 축적된 기술과 설계 프로그램들을 비동기 회로의 설계에도 활용할 수 있도록 하였다. 예를 들어, 회로를 VHDL을 이용하여 기술함으로써 기존의 VHDL 시뮬레이션 프로그램을 그대로 활용할 수 있게 된다. 또한 기존의 VHDL에 익숙한 회로 설계자들이 전혀 새로운 신호 전이 그래프, 비동기 유한 상태기 등을 배울 필요 없이 보다 쉽게 비동기 회로의 설계에 접할 수 있게 함으로써 자연스럽게 비동기 회로 설계로 이동할 수 있을 것이다. 궁극적으로는 현재의 동기 회로 설계 소프트웨어들이 하나의 프레임워크로 통합된 것과 같이 비동기 회로의 설계 환경도 서로 유기적으로 통합됨으로써 보다 편리한 설계 환경을 제공할 수 있을 것이다. 본 논문의 결과는 그러한 통합 환경을 구축하기 위한 토대를 제공할 수 있을 것이다.

2. 합성 가능한 VHDL 부집합

VHDL은 매우 방대하고 복잡한 하드웨어 기술 언어로서 회로의 시뮬레이션에는 유용하지만 합성 시스템에서는 사용할 수 없는 요소들을 많이 포함하고 있다. 이러한 이유로 동기 회로 합성 시스템들은 나름대로의 합성 가능한 VHDL 부집합을 사용하고 있다. 본 논문에서도 합성 가능한 VHDL 부집합을 정의하여 사용한다. VHDL 회로 기술은 크게 두 부분으로 구성된다. 하나는 ENTITY 선언으로서 회로의 외부 인터페이스에 대한 정보를 기술한다. 본 논문에서 사용하는 ENTITY 선언의 구문은 (그림 1)과 같다. 여기에서는 프로그램 언어를 기술하는데 널리 사용되는 BNF (Bachus-Naur Format) 형식에 따라 구문을 기술하였다. 구문에 나타나 있듯이 ENTITY 선언에서는 회로의 입출력 포트에 대한 정보를 기술하는데, 본 논문에서는 입출력 포트의 데이터 유형으로 한 비트의 값만을 가질 수 있는 BIT와 STD_LOGIC 두 가지만을 허용한다.

```
entity_declaration ::=
  ENTITY identifier IS
    entity_header
    {entity_declarative_item}
  END [entity_simple_name];

entity_header ::= [PORT (port_element {; port_element} );]
port_element ::= identifier_list : [mode] type_indication [:= expression]
mode ::= IN | OUT | INOUT
type ::= BIT | STD_LOGIC
entity_declarative_item ::= constant_declaration | signal_declaration
```

(그림 1) ENTITY 선언의 구문

회로 내부의 기능은 ARCHITECTURE BODY에 의해 기술된다. ARCHITECTURE BODY는 (그림 2)에 나타나 있듯이 먼저 ARCHITECTURE의 이름과 ENTITY의 이름이 기술되고 그 다음에 선언문들과 병렬 문장들이 기술된다. 선언문으로는 상수 선언문, 신호 선언문, COMPONENT 선언문이 허용된다. BEGIN과 END 사이의 문장들은 서로 병렬로 수행되는 문장들인데, 본 논문의 합성에서는 PROCESS 문과 COMPONENT 문만을 허용한다.

```
architecture_body ::=
  ARCHITECTURE identifier OF entity_name IS
    {architecture_declarative_item}
  BEGIN
    {concurrent_statement}
  END [architecture_simple_name];

architecture_declarative_item ::=
  constant_declaration | signal_declaration | component_declaration
constant_declaration ::= CONSTANT identifier_list : type_indication
                         [:= expression];
signal_declaration ::= SIGNAL identifier_list : type_indication
                       [:=expression];
component_declaration ::=
    COMPONENT identifier
      [PORT ( port_element {; port_element} ); ]
    END [component_simple_name];
concurrent_statement ::= process_statement | component_instantiation_
                         statement
```

(그림 2) ARCHITECTURE BODY의 구문

PROCESS 문은 (그림 3)에 나타나 있듯이 순차문들의 집합으로 구성되는데, WAIT 문, 신호 치환 문, IF 문, WHILE 문이 사용될 수 있다.

```
process_statement ::=
    [process_label:]
    PROCESS
      {process_declarative_item}
    BEGIN
      {sequential_statement}
    END process [process_label];

sequential_statement ::= wait_statement
                         | signal_assignment_statement |
                         if_statement | while_loop_statement
wait_statement ::= WAIT UNTIL boolean_expression ;
signal_assignment_statement ::= target <= value_expression ;
if_statement ::=
    IF condition THEN
      sequence_of_statements
    {ELSIF condition THEN
      sequence_of_statements}
    [ELSE
      sequence_of_statements]
    END IF;
while_loop_statement ::=
            WHILE condition LOOP
              sequence_of_statements
            END LOOP;
```

(그림 3) PROCESS 문의 구문

WAIT 문에는 세 가지 유형이 있는데 본 논문의 합성에서는 WAIT UNTIL 문을 허용한다. 프로세스의 수행 중에 WAIT 문을 만나게 되면 프로세스의 수행이 일시 정지된다. 그 다음에 UNTIL 다음의 논리 수식에 포함되어 있는 신호에 이벤트가 발생하면 논리

수식이 계산되어서 그 결과가 참이면 WAIT 다음 문장부터 프로세스의 수행이 재개된다. 신호 치환 문에서는 신호의 상향 전이 또는 하향 전이를 기술한다.

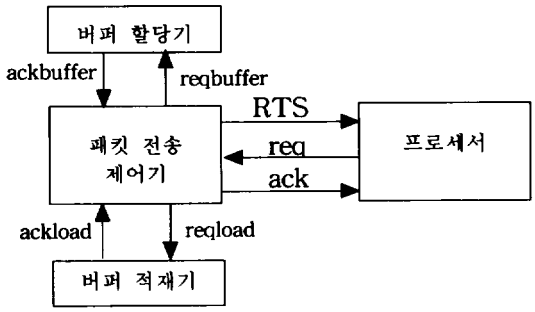
(그림 4)에는 위에서 정의한 VHDL 부집합을 이용하여 HP 회사에서 제작한 통신용 칩[16]에 포함되는 회로인 패킷 전송 제어기를 기술한 프로그램 예가 나타나 있다.

```
ENTITY mp_forward_pkt IS
    PORT (ackbuffer, ackload, req : IN BIT;
        reqbuffer, reqload, ack, RTS : OUT BIT);
END mp_forward_pkt;

ARCHITECTURE behavior OF mp_forward_pkt IS
BEGIN
    PROCESS
    BEGIN
        reqbuffer <= '1';
        WAIT UNTIL ackbuffer = '1';
        RTS <= '1';
        reqbuffer <= '0';
        WAIT UNTIL req = '1' AND ackbuffer = '0';
        reqload <= '1';
        RTS <= '0';
        WAIT UNTIL ackload = '1';
        ack <= '1';
        reqload <= '0';
        WAIT UNTIL req = '0' AND ackload = '0';
        ack <= '0';
    END PROCESS;
END behavior;
```

(그림 4) 패킷 전송 제어기에 대한 VHDL 기술

(그림 5)에는 패킷 전송 제어기에 대한 블록 다이어그램이 나타나 있다. 패킷 전송 제어기는 reqbuffer 신호에 1을 보내어 버퍼 할당기로부터 데이터 패킷을 적재할 수 있는 빈 버퍼를 요구한다. 버퍼 할당기는 버퍼를 할당한 다음에 ackbuffer 신호의 값을 1로 만든다. 패킷 전송 제어기는 버퍼를 할당받은 다음에 RTS 신호의 값을 1로 만들어 프로세서에 데이터 패킷을 보낼 준비가 되어 있음을 알리고 reqbuffer 신호를 0으로 만들어 버퍼 할당 요구를 취소한다. 그러면 프로세서는 req 신호의 값을 1로 만들어 데이터 패킷을 보내라고 요구하고 버퍼 할당기는 ackbuffer 신호의 값을 0으로 만든다. 그 다음에는 패킷 전송 제어기가 reqload 신호의 값을 1로 만들어 버퍼에 데이터를 적재하라고 요구하고 RTS 신호를 0으로 만든다. 그러면 패킷 적재기는 버퍼에 데이터를 적재한 다음에 ackload 신호를 1로 만든다. 그러면 패킷 전송 제어기는 ack 신호의 값을 1로 만들어서 프로세서에 패킷이 전송되었음을 알리고 reqload 신호의 값을 0으로 만든다. 그러면 프로세서는 req 신호의 값을 0으로 만들고 패킷 적재기는 ackload 신호의 값을 0으로 만든다. 그 다음에 패킷 전송 제어기가 ack 신호의 값을 0으로 만듦으로써 초기 상태로 돌아간다.



(그림 5) 패킷 전송 제어기의 블록 다이어그램

3. VHDL 기술로부터의 합성

본 논문에서는 VHDL 기술을 신호 전이 그래프로 변환한 다음에 기존의 신호 전이 그래프로부터의 속도 독립 회로 합성 알고리즘[15]을 이용하여 회로를 합성한다.

3.1 신호 전이 그래프

신호 전이 그래프는 Chu[6]에 의해 제안된 비동기 회로의 기술 방법으로서 Petri 네트[17]의 한 부 집합을 비동기 회로의 기술에 적합하도록 해석한 것이다. Petri 네트는 다음과 같이 정의된다.

정의 1 Petri 네트는 $\langle T, P, F, M_0 \rangle$의 튜플로서 정의된다. T는 전이(transition)의 집합을 나타내고 P는 장소(place)의 집합을 나타낸다. F는 전이와 장소 사이의 흐름 관계를 나타내며 $F \subseteq (P \times T) \cup (T \times P)$의 관계를 갖는다. M_0는 초기 표식(marking)을 나타내는데, 표식이란 네트의 각 장소에 토큰이라 불리는 0이상의 정수를 할당하는 함수로 정의된다.

Petri 네트는 표현력이 매우 강하여 광범위한 시스템을 기술할 수 있지만 이를 합성에 이용하기는 어렵

기 때문에, Petri 네트의 한 부 집합으로서 병렬성과 비결정성을 모두 갖는 시스템을 기술할 수 있는 자유 선택 네트에 기반하여 신호 전이 그래프를 정의하였다. 자유 선택 네트는 다음과 같이 정의된다.

정의 2 자유 선택 네트는 두 전이 t_1과 t_2가 같은 입력 장소 p를 공유할 경우에 언제나 p가 t_1과 t_2의 유일한 입력 장소가 되는 Petri 네트로 정의된다.

신호 전이 그래프는 자유 선택 네트의 전이를 기술하려는 회로에서의 신호의 물리적인 전이로 해석한 것이다. 신호 s의 전이는 $s+$ 또는 $s-$와 같이 나타낸다. $s+$는 신호 s의 값이 0에서 1로 바뀌는 것을 나타내고, $s-$는 s의 값이 1에서 0으로 바뀌는 것을 나타낸다. $s*$는 $s+$ 또는 $s-$를 나타내고 $\overline{s*}$는 $s*$의 역방향 전이를 나타낸다. 기술하려는 회로의 전체 신호 집합을 S라 하자.

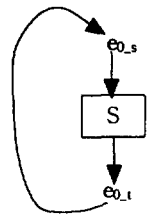
정의 3 신호 전이 그래프는 전이의 집합 T가 신호 전이의 집합 $S \times \{+,-\}$으로 해석된 자유 선택 네트로 정의된다.

3.2 신호 전이 그래프로의 변환

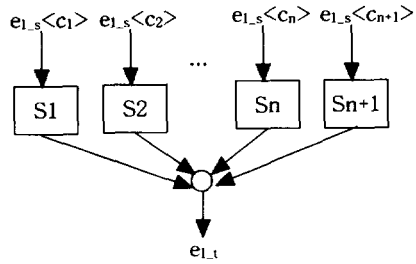
3.2.1 문장의 변환

본 논문에서는 먼저 VHDL 문장을 부분적인 신호 전이 그래프로 먼저 변환한 다음에 이를 합병해 나감으로써 VHDL 기술로부터 신호 전이 그래프로 변환해 나간다. VHDL 기술이 여러개의 PROCESS 문으로 구성되어 있으면 먼저 각각의 프로세스를 신호 전이 그래프로 변환한 다음에 이를 합병한다. 하나의 프로세스는 계속해서 반복 수행되는 무한 루프와 같으므로 (그림 6)과 같이 변환된다. 여기에서 e_{i_s} $(i=0,1,\ldots,n)$는 시작 임시 전이라 하고 e_{i_t}는 종료 임시 전이라 하자. 이러한 임시 전이는 변환된 신호 전이 그래프들을 합병을 용이하게 하기 위해 사용되는데, 나중에 합병 과정에서 제거될 수 있다.

(그림 7)에는 IF 문의 변환 방법이 나타나 있다. IF 문의 각 조건에 따라 수행되는 문장들은 서로 배타적으로 수행되므로 이들 각각은 그림과 같이 서로 독립적인 가지로 나뉜 다음에 하나의 장소에서 만나도록 변환된다. 하나의 장소에 여러 개의 입력 전이들이 있으면 이들은 서로 배타적으로 일어난다는 것을 의미한다.

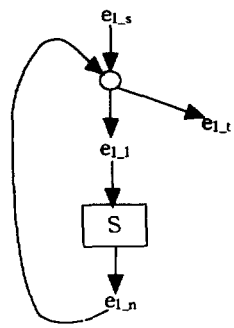


(그림 6) 프로세스의 변환 : PROCESS BEGIN S END PROCESS



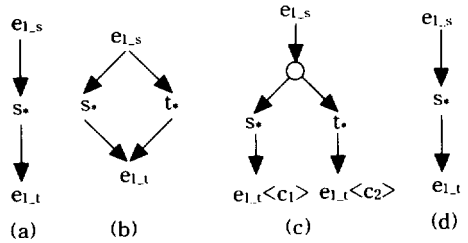
(그림 7) IF 문의 변환 : IF (C1) THEN S1 ELSIF (C2) THEN S2 ... ELSIF (Cn) THEN Sn ELSE Sn+1 END IF

(그림 8)에는 WHILE 문의 변환 방법이 나타나 있다. WHILE 문의 내부는 주어진 조건이 참인 경우에 반복 수행되므로 그림과 같이 루프가 생성되도록 변환한다. 그리고 조건이 거짓일 경우에 루프를 빠져나갈 수 있도록 하기 위하여 입력 전이와 출력 전이가 여러 개인 장소가 사용되고 그림과 같이 임시 전이와 아크가 삽입된다. 신호 전이 그래프에서 장소의 출력 아크에 연결되어 있는 신호 전이가 여러 개인 경우에 이들은 서로 배타적으로 발생한다는 것을 의미한다.



(그림 8) WHILE 문의 변환 : WHILE (Cond) LOOP S END LOOP

(그림 9)에는 WAIT 문장과 신호 치환문의 변환 방법이 나타나 있다. WAIT 문에서 기다리는 이벤트가 하나인 경우에는 (그림 9(a))와 같이 주어진 신호 전이와 임시 전이 두 개를 생성하고 신호 전이 사이에 아크를 삽입한다.



(a) WAIT UNTIL s ='*' (b) WAIT UNTIL s='*' AND t='*'
(c) WAIT UNTIL s='*' OR t='*' (d) s<= '*'

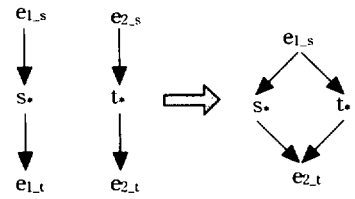
(그림 9) WAIT 문의 변환

WAIT 문에서 기다리는 이벤트가 AND로 연결되어 있으면 신호 전이들이 병렬로 일어날 수 있다는 것을 의미하므로 (그림 9) (b)와 같이 병렬로 발생할 수 있는 신호 전이로 변환하고 그림과 같이 임시 전이와 아크가 삽입된다. WAIT 문에서 기다리는 이벤트가 OR로 연결되어 있으면 신호 전이들이 서로 배타적으로 일어난다는 것을 의미하므로 (그림 9) (c)와 같이 변환한다. 그리고 종료 임시 전이에는 그 전이가 일어날 수 있는 조건을 기록해두어 나중에 합병에 사용될 수 있도록 한다. 신호 치환 문은 (그림 9) (d)와 같이 변환된다. WAIT 문과 신호 치환문이 IF 문의 각 문장 블록의 첫 번째 문이면 시작 임시 전이에도 해당 조건을 추가하여 나중에 해당 조건의 신호 전이와 합병될 수 있도록 한다.

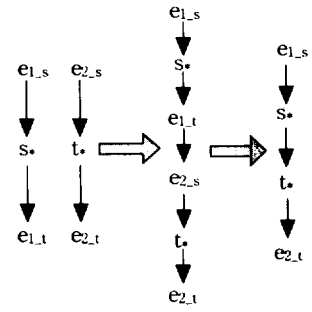
3.2.2 신호 전이 그래프의 합병

신호 치환문이 연속해서 오는 경우에는 (그림 10)과 같이 합병될 수 있다. VHDL에서 신호 치환문이 연속해서 나타날 경우에 이는 순차적인 동작을 의미하지만 속도 독립 회로에서는 입력의 변화 없이 두 개의 출력이 순차적으로 변하는 것은 두 신호 전이가 병렬로 일어나는 것과 동일한 의미를 가진다. 따라서 그림과 같이 합병함으로써 회로의 병렬성을 높일 수 있다.

WAIT 문이 연속해서 나타나거나 WAIT 문과 신호 치환문이 연속해서 나타나는 경우에는 두 문장이 순서대로 수행되어야 한다. 따라서 각각의 문장에 대한 신호 전이가 하나씩 발생하는 경우에는 (그림 11)과 같이 앞 문장에 대한 종료 임시 전이로부터 뒷 문장에 대한 시작 임시 전이로의 아크를 삽입하면 된다. 그런데, 중간의 두 임시 전이는 그림과 같이 제거될 수 있다.

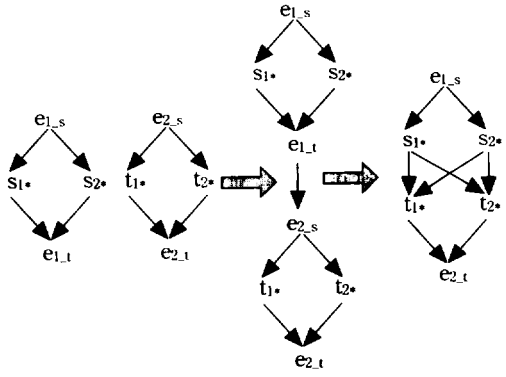


(그림 10) 연속된 신호 치환문의 합병



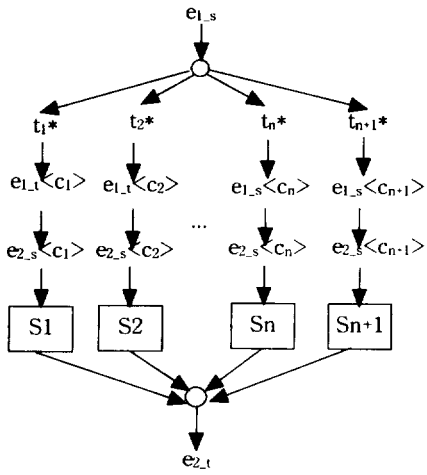
(그림 11) WAIT 문과 WAIT 문 또는 WAIT 문과 신호 치환문의 합병 : 신호 전이가 하나씩인 경우

또한 각각의 문장에 여러 개의 신호 전이가 병렬로 발생하는 경우에는 (그림 12)와 같이 임시 전이의 모든 선행 전이로부터 후행 전이로의 아크를 삽입함으로써 중간의 임시 전이를 제거할 수 있다.



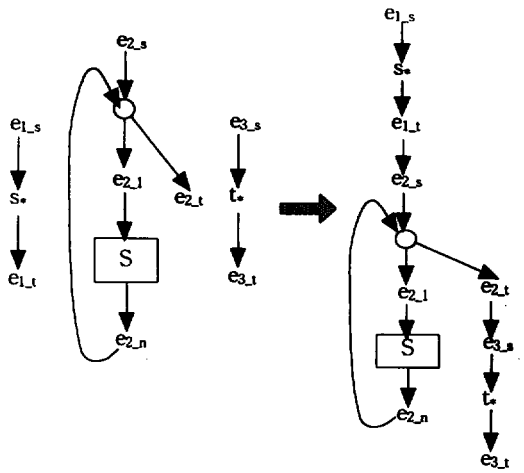
(그림 12) WAIT 문과 WAIT 문 또는 WAIT 문과 신호 치환문의 합병 : 신호 전이가 병렬로 발생하는 경우

WAIT 문과 IF 문과의 합병은 (그림 13)과 같이 종료 임시 전이와 시작 임시 전이의 조건이 같은 것끼리 합병함으로써 이루어진다.



(그림 13) WAIT 문과 IF 문의 합병

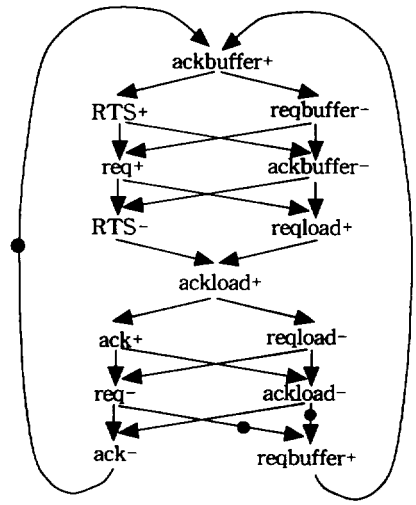
WHILE 문과의 합병의 한 예가 (그림 14)에 나타나 있다. WHILE 문의 종료 임시 전이와 WHILE 문 다음 문의 시작 임시 전이를 합병함으로써 WHILE 문의 루프를 빠져나갈 때 WHILE 문 다음 문이 수행되도록 한다. 합병된 다음에 중간에 있는 시작 임시 전이와 종료 임시 전이들은 앞에서 설명한 방법에 의하여 제거된다.



(그림 14) WHILE 문과의 합병 : s <= '*' ; WHILE (cond) LOOP S END LOOP; t <= '*';

앞에서 설명한 방법에 의하여 합병해 나감으로써 각 프로세스는 하나의 완성된 신호 전이 그래프로 변환된다. 프로세스가 신호 전이 그래프로 변환된 다음에는 신호 전이들을 검사하여 프로세스 사이에 공통적인 신호 전이를 찾아서 이들을 합병함으로써 신호 전이 그래프들을 합병한다. 공통된 신호 전이가 없으면 각 신호 전이 그래프는 독립적으로 합성된다.

(그림 4)의 패킷 전송 제어기에 대한 VHDL 프로그램을 본 논문에서 제안한 변환 방법에 의하여 신호 전이 그래프로 변환한 결과가 (그림 15)에 나타나 있다.



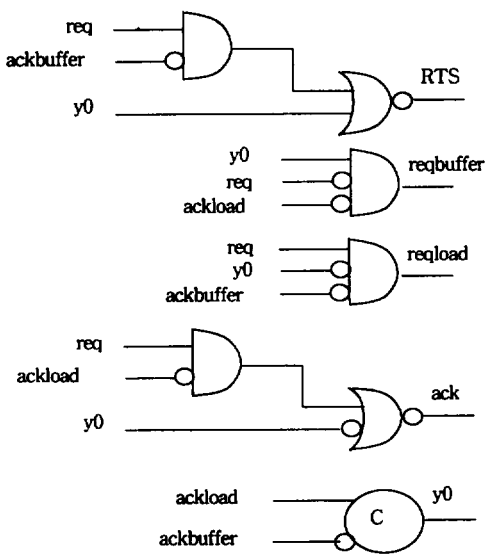
(그림 15) 패킷 전송 제어기에 대한 신호 전이 그래프

3.3 신호 전이 그래프로부터의 합성

본 논문에서 사용한 신호 전이 그래프로부터 속도 독립 회로 합성 알고리즘 적용되기 위해서는 신호 전이 그래프가 유일 상태 코딩 특성을 만족해야 한다. 유일 상태 코딩 특성이란 신호 전이 그래프에 대한 상태 그래프에서 임의의 두 상태가 서로 다른 이진 코드를 가지는 것을 의미한다. 즉, 각 표식에서의 신호들의 값이 서로 같은 경우가 없다는 것을 의미한다.

VHDL 기술로부터 변환된 신호 전이 그래프도 이 조건을 만족하도록 해야 한다. 따라서, 유일 상태 코딩 특성을 만족하지 못하는 경우에는 신호 전이 그래프에 새로운 신호 전이를 추가하여 이 특성을 만족하도록 만든다. 본 논문에서는 Enric[18]에 의하여 제안된 유일 상태 코딩 알고리즘을 이용하여 이 특성이 만족되도록 한다.

유일 상태 코딩 특성이 만족하는 신호 전이 그래프는 기존의 합성 방법에 의하여 속도 독립 회로로 변환된다. 이 속도 독립 회로는 AND, OR, NOT 게이트와 C-원소의 단순 게이트들로 구성된다. 패킷 전송 제어기에 대한 합성 결과가 (그림 16)에 나타나 있다.



(그림 16) 패킷 전송 제어기에 대한 합성 결과

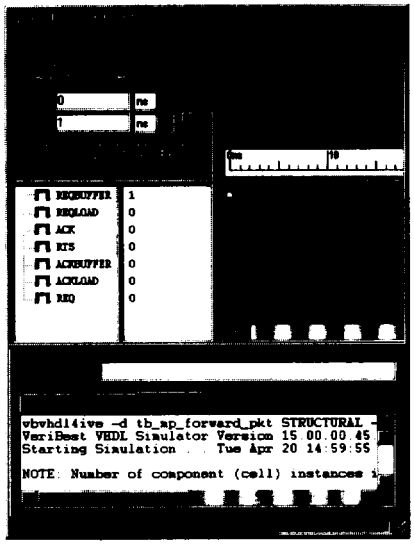
5. 실험 결과

본 논문의 합성 시스템은 Sparc 워크스테이션에서 C 언어를 사용하여 구현되었다. VHDL 언어를 사용하여 기존의 벤치마크 회로를 기술한 다음에 본 논문에서 제안한 변환 방법에 의하여 신호 전이 그래프로 변환하고 이로부터 속도 독립 회로를 합성한 결과가 <표 1>에 나타나 있다.

<표 1>에 나타나 있듯이 기존의 비동기 회로에 대한 벤치마크들은 본 논문에서 제안한 VHDL 부집합에 의하여 기술될 수 있었고 속도 독립 회로로 합성될 수 있었다. 합성된 회로의 면적은 기존의 신호 전이 그래프로 기술하였을 때와 동일하였다. 또한 VHDL로 기술된 회로 명세에 대하여 VHDL 시뮬레이터를 적용함으로써 회로 명세가 올바른지 검사할 수 있었다. (그림 17)에는 VHDL 시뮬레이터를 이용하여 패킷 전송 제어기에 대해 시뮬레이션을 수행하는 화면이 나타나 있다.

〈표 1〉 실험 결과

회 로	합성 결과	
	리터럴 수	C-원소 수
아날로그-디지털 변환기	10	2
FIFO 버퍼 제어기	10	1
full handshake	0	2
half handshake	5	2
2-4 phase handshake	20	3
4-2 phase handshake	14	2
alloc-outbound	24	2
패킷 전송 제어기	12	1
분산 상호 배제 제어 회로	29	3
master-read	34	7
nak-pa	17	1
nowick	12	1
ram-read-sbuf	20	3
wrdata	15	3
MMU 제어기	22	4
sbuf-read-ctl	13	2
sender-done	5	1
vme	6	1



(그림 17) 패킷 전송 제어기에 대한 VHDL 시뮬레이션 결과

6. 결 론

본 논문에서는 VHDL를 이용하여 비동기 회로를 기술하는 방법과 VHDL 기술을 신호 전이 그래프로 변환함으로써 속도 독립 회로를 합성하는 방법을 제안하였다. VHDL 프로그램으로부터 신호 전이 그래프의

체계적인 변환 방법으로 본 논문에서는 각각의 VHDL 문을 부분적인 신호 전이 그래프로 변환하고 부분적인 신호 전이 그래프들을 합병하는 방법을 제안하였다. 기존의 속도 독립 회로 합성 시스템에서는 표준화된 기술 방법을 사용하지 않음으로써 기존의 축적된 많은 CAD 소프트웨어를 이용하지 못할 뿐만 아니라 통합된 설계 환경을 제공하지 못하고 있다. 따라서 본 논문에서는 속도 독립 회로의 설계에 VHDL을 사용할 수 있도록 하여 기존의 축적된 기술과 설계 프로그램들을 속도 독립 회로의 설계에도 활용할 수 있도록 함으로써 통합된 설계 환경 구축의 기틀을 마련하였다.

또한 기존의 VHDL에 익숙한 회로 설계자들이 전혀 새로운 신호 전이 그래프, 비동기 유한 상태기 등을 배울 필요 없이 보다 쉽게 속도 독립 회로의 설계에 접할 수 있게 함으로써 자연스럽게 속도 독립 회로 설계로 이동이 가능하도록 하였다.

앞으로 본 논문에서 구현한 합성 시스템을 기반으로 기존의 설계 소프트웨어들과의 통합을 통하여 상위 단계의 하드웨어 기술로부터 칩 레이아웃에 이르기까지의 전 과정을 자동화하는 통합된 설계 환경에 대한 연구가 수행되어야 할 것이다.

참 고 문 헌

[1] P.A. Beerel and T. H.-Y. Meng, "Automatic Gate-Level Synthesis of Speed-independent Circuits," *Proceedings of International Conference on Computer Aided Design*, pp.581-586, Nov. 1992.

[2] V.I. Varshavsky, V.B. Marakhovshy, and V.V. Smolensky. "Designing self-timed devices using the finite automaton model," *IEEE Design & Test of Computers*, Vol.12, No.1, pp.14-23, Spring 1995.

[3] B. Lin and S. Devadas, "Synthesis of hazard-free multilevel logic under multi-input changes from binary decision diagrams," *IEEE Transactions on Computer-Aided Design*, Vol.14, No.8, pp.974-985, Aug. 1995.

[4] F.-C. Cheng, "Synthesis of high speed delay-insensitive combinational iterative tree circuits," *In Proc. International Conf. Computer Design (ICCD)*, pp.301-306, Oct. 1997.

[5] P.A. Beerel, C.J. Myers, and T.H.-Y. Meng, "Covering conditions and algorithms for the synthesis of speed-independent circuits," *IEEE Transactions on Computer-Aided Design*, Vol.15, No.3, March 1998.

[6] T.A. Chu, "Synthesis of Self-timed VLSI Circuits from Graph Theoretic Specifications," *Ph.D. Thesis*, Massachusetts Institute of Technology, 1987.

[7] K.Y. Yun, "Automatic synthesis of extended burst-mode circuits using generalized C-elements," *In Proc. European Design Automation Conference (EURO-DAC)*, pp.290-295, Sep. 1996.

[8] A.J. Martin, "Programming in VLSI: From communicating processes to delay-insensitive circuits," *In C. A. R. Hoare, editor, Developments in Concurrency and Communication, UT Year of Programming Series*, pp.1-64. Addison-Wesley, 1990.

[9] D.L. Perry, "*VHDL*," 2nd edition, McGraw-Hill Book Co., 1994.

[10] H. Jacobson, "ACK - An Asynchronous Circuit Design Framework," *Talk slides presented at Sun Microsystems*, San Francisco, U.S.A., March 1997.

[11] S.B. Furber, J.D. Garside, S.Temple, J. Liu, P. Day, and N.C. Paver, "AMULET2e: An asynchronous embedded controller," *In Proc. International Symposium on Advanced Research in Asynchronous Circuits and Systems*, pp.290-299. IEEE Computer Society Press, April 1997.

[12] I.E. Sutherland, "Micropipelines," *Communications of the ACM*, Vol.32, No.2, pp.720-738, June 1989.

[13] S.-Y. Tan, S. B. Furber, and W.-F. Yen. "The design of an asynchronous VHDL synthesizer," *In Proc. Design, Automation and Test in Europe (DATE)*, IEEE Computer Society Press, February 1998.

[14] 정성태, 전주식, "결정성 신호 전이 그래프로부터

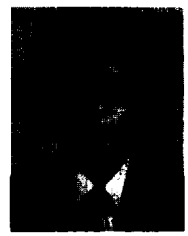
속도 독립 회로의 합성 알고리즘", *한국정보과학회 논문지*, 제21권, 6호, pp.1026-1036, 1994.

[15] 정성태, 전주식, "신호 전이 그래프로부터 속도 독립 회로의 합성 알고리즘", *한국정보과학회 논문지*, 제21권, 11호, pp.2026-2038, 1994.

[16] A. Davis, B. Coates, and K. Stevens, "The Post Office experience: Designing a large asynchronous chip," *In Proc. Hawaii International Conf. System Sciences*, Vol.I, pp.409-418. IEEE Computer Society Press, Jan. 1993.

[17] J.L. Peterson, *"Petri Net Theory and the Modeling of Systems,"* Prentice-Hall, 1981.

[18] E. Pastor and J. Cortadella, "An Efficient Unique State Coding Algorithm for Signal Transition Graphs," Proceedings of International Conference on Computer Design, pp.174-177, Oct. 1993.



정 성 태

e-mail : stjung@wonnms.wonkwang.ac.kr

1987년 서울대학교 컴퓨터공학과 졸업(학사)

1989년 서울대학교 대학원 컴퓨터 공학과(공학 석사)

1994년 서울대학교 대학원 컴퓨터 공학과(공학 박사)

1994년 9월~1995년 2월 한국전자통신연구소 박사후 연수 연구원

1995년~현재 원광대학교 컴퓨터공학과 교수

관심분야 : 비동기 회로 논리 합성, 비동기 회로 설계, 컴퓨터 그래픽스