내장형 시스템 설계 언어인 SystemC로 부터의 Blif-MV 변환 규칙 연구*

김민숙° 안영정 방기석 최진영

고려대학교 컴퓨터학과

{mskim, yjahn, kbang, choi} @formal.korea.ac.kr

A Study on the Translation Rule from SystemC to Blif-MV : SC2MV

Min-Suk Kim° , Young-Jung Ahn, Ki-Seok Bang , Jin-Young Choi

Dept. of Computer Science and Engineering, Korea University

요 약

내장형 시스템의 개발에 있어서 자원의 효율적인 활용과 정확한 설계를 위해 SystemC를 이용한 통합설계 방식이 많이 사용되고 있다. 하지만 시스템이 점점 복잡해 지면서 단순한 언어차원에서의 개발 뿐 아니라 개발 이전에 시스템의 정확성을 검증해야 할 필요성이 대두되었다. 이를 위해 정형기법 및 테스팅과 같은 방법을 사용하게 되었다. 본 논문에서는 SystemC로부터 정형기법 도구인 VIS의 입력 언어인 BLIF-MV로 자동 변환하는 알고리즘을 제시하고, SystemC 코드로부터의 자동 검증 방법을 제안하고자 한다.

1. 서 론

미래의 휴대용 정보기기를 포함한 여러 내장형 시스템은 반도체 설계 및 생산 공정의 급속한 발달로 그 복잡도가 더해가고 있다. 과거의 내장형 시스템의 설계과정을 보면 모델링 단계 후 하드웨어와 소프트웨어 partitioning을 거쳐 각각 개별적으로 설계를 한 후 하드웨어 부분은 합성단계를 거치고 소프트웨어 부분은 구현 단계를 거쳐 내장형 시스템을 구축하였다. 하지만 아무런 의사소통이 없이 설계되는 방법은 각각의 설계를 끝냈을 때 서로 맞지 않아 재설계를 해야 하는 등의 시간과 비용의 낭비를 가져오고 시스템의 안전성과 신뢰성을 극도로 떨어뜨렸다. 이로 인해 내장형 시스템의 통합 설계 기법[1]이란 새로운 분야의 선두 주자로 SystemC[2]란 새로운 내장형 시스템 통합 설계 언어가 탄생하게 되었다. SystemC는 상위 단계에서 기술된 고성능의 복잡한 시스템을 하드웨어와 소프트웨어 부분을 적절히 혼합하여 최적의 성능을 가지도록 설계하는 언어로써 시스템의 성능 향상 및 설계 비용의 최소화 면에서 많은 기여를 하고 있다. 내장형 시스템에 대한 통합설계가 가능해짐으로써 전체 시스템을 하나의 칩 안에 구축하는 SoC(System On a Chip)가 현재 내장형 시스템의 연구가 많은 분야에서 진행 되고 있다. 하지만 내장형 시스템이 복잡해지면서 테스팅 단계에서 테스팅이 좀 더 어려워지고, 비용도 증가하고 있다. 사용자의 요구는 날로 커지고 개발의 라이프 사이클은 줄어들자 asserted-driven formal verification이라 하여 복잡한 시스템의 테스팅에 대한 대안으로 시스템의 정형 기법[3]에 의한 정형 검증이 많은 벤더들의 눈길을 끌고 있다. 이를 대표하는 모델 체킹[4]은 비록 적용분야는 유한 시스템으로 제한되지만, 상태 전이 시스템과 특성이 주어지면, 모델 체킹 알고리즘은 주어진 시스템이 검증하고자 하는 특성을 만족하는지를 알아보기 위해서 전체 상태 공간을 검사하고 그 결과를 알려줄 수 있다 본 연구에서는 시스템 기술 언어인 SystemC를 입력 언어로 받아들여 VIS[5]라는 모델 체킹 도구의 입력 언어인 BLIF-MV[6]로 변환하여 검증을 수행하는 방법을 논하고자 한다. 이 방법을 통해 다음과 같이 두 가지 장점을 얻을 수 있다. 첫째 시스템 수준 명세언어를 이용하여 시스템을 기술할 경우 복잡한 시스템의 세부 사항들을 나타내지 않고 구현과 독립적인 기술을 할 수 있어 최상위 수준에서 조기에 시스템의 결함을 찾아 낼 수 있다. 둘째, 모델 체킹과 같은 정형 검증을 이용함으로써 명세에 대하여 요구 조건이 만족하는지의 결과를 알아내어 시스템의 안정성과 신뢰성을 높일 수 있다.

2장에서는 관련 연구를, 3장에서는 통합 설계를 위한 방법론인 SystemC에 대해 논하고, 4장에서는 모델 체킹을 이용한 정형 검증 도구인 VIS에 대해서, 그리고 5장에서는 SystemC를 입력으로 받아들여 모델 체커의 입력 언어인 BLIF-MV 생성 기법에 대해서 논한다. 끝으로 6장에서는 결론 및 향후 연구로 본 논문을 마치고자 한다.

2. 관련 연구

내장형 시스템의 통합 설계는 다양한 곳에서 다양한 측면으로 연구되고 있다. Berkeley를 중심으로 구성된 Polis그룹[7]은 하드웨어-소프트웨어 통합 설계를 하는 Polis라는 도구를 개발하고 있다. 이 도구는 Esterel[8]을 입력 언어로 받아 들여 이를 Codesign Finite State machine[7]으로 바꾸어 VIS라는 도구로 행위적 검증을 거친 후, 하드웨어와 소프트웨어 부분을 자동으로 분할 한 후 각각을 자동화된 도구로 합성하

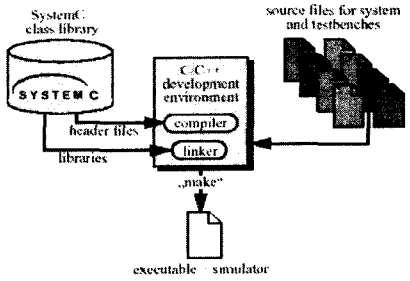
* 본 연구는 한국과학재단 목적기초연구(R01-2000-00287)지원으로 수행되었음

고 있다. 특히 이 도구에서는 Ptolemy[9]를 이용하여 다양한 시뮬레이션을 수행하고 있다. 이 외에도 하드웨어-소프트웨어 통합 설계는 다양하게 연구되고 있다.

2. SystemC

2.1 개요

SystemC는 [그림1]에서 볼 수 있듯이 C++ 라이브러리이고 소프트웨어 알고리즘, 하드웨어 아키텍처, 시스템 레벨 디자인, Soc를 효과적으로 생성하는데 사용될 수 있는 방법론이다. SystemC 클래스 라이브러리는 표준 C++에서는 찾아볼 수 없는 하드웨어 타이밍, 병렬 동작, 반응 동작(reaction behavior) 등을 포함하는 시스템 아키텍처를 모델하기 위한 필요한 구조를 제공해준다. 이러한 구조를 C에 포함시키는 것은 언어에 대한 독점적인 확장을 요구한다. C++ 객체 지향 프로그래밍 언어는 새로운 문법적 구조를 추가하지 않고도 언어를 클래스들을 통해서 확장할 수 있는 능력을 제공하고 있다. SystemC는 이러한 필요한 클래스들을 제공하고 설계자들에게 친숙한 C++언어와 개발 장비를 계속 사용할 수 있도록 해 만약 독자들이 C ++ 프로그래밍 언어에 친숙해 있다면 추가적인 문법을 배우지 않고도 SystemC 클래스에서 소개되는 추가적인 의미를 이해함으로 해서 SystemC로 프로그램하는 방법을 배울 수 있다.



[그림1] SystemC Design Flow

2.2 특징

SystemC는 하드웨어-소프트웨어 통합설계를 지원한다. 하드웨어와 소프트웨어 컴포넌트로 구성되는 복잡한 시스템 구조를 지원할 수 있도록 다음과 같은 특징을 갖는다.

1) 모듈(Module): 다른 모듈이나 모듈 안에 포함되는 프로세스를 가지는 계층적인 객체
2) 프로세스(Process): 기능을 표현하면 모듈에 포함
3) 포트(Port): 다른 모듈과의 연결 통로 역할
4) 클럭(Clock): 클럭 표시법이 존재

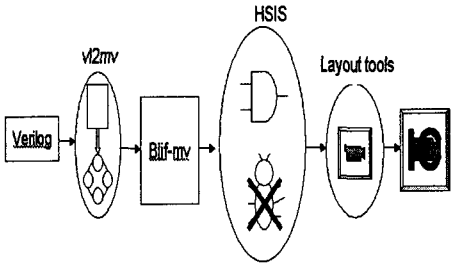
2.3 설계 방법론

[그림2]와 같은 SystemC 설계 접근 방식은 전통적인 접근 방식인 HW와 SW를 서로 다른 언어로 기술하는 것에 비해 시스템 레벨에서 RTL(Register Transfer Level) 수준까지 모델링 할 수 있게 해준다. 또한 하드웨어와 타이밍 구조가 추가되었기 때문에 작은 부분으로 나누어 조금씩 정제하는 방법을 사용할 수 있다.

3. 정형 검증 도구

3.1 VIS

VIS (Verification Interacting with Synthesis) 는 검증, 시뮬레이션, 유한 상태 하드웨어 시스템의 합성(synthesis)을 종합한 도구이다. 이 도구는 Verilog를 입력 언어로 사용하여 중간 포맷인 blif-mv를 사용하여 fair CTL 모델체킹, language emptiness 검사, 조합 순차 동등성 검사, 원형 기반 시뮬레이션(cycle-based simulation)과 계층적 체계화(hierarchical synthesis)를 지원한다. VIS는 Berkeley대학과 콜로라도 대학 과 협력하여 개발 되고 있으며 1세대 도구인 HSIS나 SMV에 비교하여 보다 개선된 프로그래밍 환경과 새로운 호환성을 제공하며 또한 성능을 향상 시켜주는 경우도 있다. VIS는 BLIF-MV라는 중간 형태에서 작동하며 BLIF-MV 파일은 VL2MV라 불리는 컴파일러에 의해 생성된다. VIS의 전체적인 동작은 [그림2]와 같이 Verilog-HDL을 받아들여 vl2mv라는 컴파일러를 이용하여 중간 코드인 BLIF-MV로 바꾸어 HSIS로 합성 및 디버그를 하며, HSIS를 이용한 결과를 layout tool을 이용하여 집적회로를 만든다.



[그림2] VIS 소개

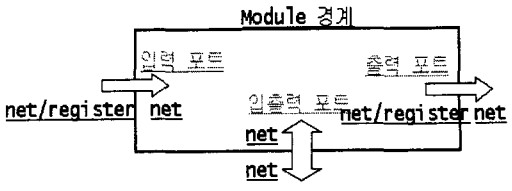
3.2 vl2mv

BLIF-MV는 비결정적인 계층적 순차 시스템을 묘사하기 위해 디자인된 언어이다. BLIF-MV 파일은 VL2MV라 불리는 컴파일러에 의해 생성된다. BLIF-MV은 오토마타로 묘사되며 이는 VIS로 읽혀져 계층적 트리 형태로 저장된다. 이 계층에 대한 순회는 UNIX의 디렉토리에 대한 순회와 유사하며 시뮬레이션과 검증 작업은 어느 계층에서나 가능하다. 이는 차례로 하위 모듈을 구성하고 모듈의 기능은 임의의 함수와 래치를 가진 게이트의 네트워크에 의해 나타내어진다. 각각의 표현된 오토마타의 스테이트 수는 메인 모델의 래치 수와 하위 모델의 래치 수를 더한 값을 n이라 할 경우 n^2개 만큼 된다.

4. SystemC에서 BLIF-MV 생성 방법

4.1 Net variables & Register variables

변수의 선언 및 사용 시에는 [그림3]과 같이 Verilog[10]에서 사용하는 포트 연결 규칙을 따른다. 넷은 하드웨어 요소 사이에 연결을 나타낸다. SystemC에서 넷은



[그림 3] 포트 연결 규칙

" sc_in<type> 이름" 또는 " sc_inout<type> 이름" 을 통해 정의된다. 레지스터는 데이터를 저장할 수 있다. 레지스터는 다른 논리값이 들어오기 전까지는 그 값을 유지할 수 있다. SystemC에서 레지스터는 " sc_out<type> 이름" 또는 일반 데이터 선언을 통해 정의 된다. SystemC(오른쪽)에 따른 BLIF-MV(왼쪽)의 변환 규칙을 나타낸다. 변수가 latch로 선언되었을 때 변수이름_ps는 현재 상태를 변수이름_ns는 다음 상태를 나타낸다.

1) net variable

```
sc_in<bool> x,y;          .names x y a
sc_out a;                 - - = x
a = x; a = y;             - - = y
```

2) register variable

```
                          .names c1 e1 c2 e2 a_ps
sc_out<sc_bit> a;         a_ns
if (c1) a = e1            1 - - - -= e1
if (c2) a = e2            - 1 - - -= e2
                          0 0 - - - =a_ps
```

4.2 Statements

Concatenation 은 결합 연산자로 여러 개의 피연산자들을 한데 묶는 메커니즘을 제공한다. 피연산자들은 크기가 항상 정해져 있어야 하며, (,)을 통해 표현된다. 특정 조건에 따라서 문장을 수행할지 말지를 결정하는 조건문은 C-언어에서와 같이 if-else 문을 사용하여 표현된다. 분기 조건이 3개 이상인 경우 if-else-if 문을 다루기가 불편하게 된다. 이 경우 case 문을 사용해서 간단하게 같은 결과를 얻을 수 있다. Statements 의 SystemC(위쪽)에 따른 BLIF-MV(아래쪽)의 변환 규칙을 나타낸다.

1) concatenation

```
sc_in<bool> b;
sc_in<bool> c, d;
sc_in<sc_int<4>> a;
a = ('2', b, c&d);
```

```
.names t<0>
0
.names t<1>
0
.names b t<2>
- =b
.names c<0> d<0> t<3>
.def 0
1 1 1
.names t<0> a<0>
- =t<0>
.names t<1> a<1>
- =t<1>
.names t<2> a<2>
- =t<2>
.names t<3> a<3>
- =t<3>
.names t<4> a<4>
- =t<4>
```

2) if statement

```
If (c1) a= e1
Else a = e2;
```

```
.names c1 e1 e2 a_ps a_ns
1 - - - =e1
0 - - - =e2
```

4.3 예제

위의 규칙을 적용시켜 D-flip flops의 SystemC코드를 다음과 같이 변환 시킬 수 있다.

```
#include "systemc.h"
SC_module(dff){
  Sc_in<bool> din;
  Sc_in<bool> clock;
  Sc_out<bool> dout;

  Void doit(){
    Dout = din;}

  SC_CTOR(dff){
    SC_METHOD(doit);
    Sensitive_pos << clock;}
};
```

```
.model dff
.input din
.input clock
.output dout
.names din dout_ps dout_ns
1 - 1
0 - 0
.r dout_ns dout
1 1
0 0
```

5. 결론 및 향후 연구 방향

본 연구에서는 시스템 기술 언어인 SystemC를 입력 언어로 받아들여 VIS라는 모델 체킹 도구의 입력 언어인 BLIF-MV로 변환하여 검증을 수행하는 방법을 제안하였다. 이 방법을 통해 다음과 같이 두 가지 장점을 얻을 수 있다. 첫째, 시스템 수준 명세언어를 이용하여 시스템을 기술할 경우 복잡한 시스템의 세부 사항들을 나타내지 않고 구현과 독립적인 기술을 할 수 있어 최상위 수준에서 조기에 시스템의 결함을 찾아 낼 수 있다는 장점과 둘째, 모델 체킹과 같은 정형 검증을 이용함으로써 명세에 대하여 요구 조건이 만족하는지의 결과를 알아 낼 수 있다.

향후 연구 과제로는 본 연구에서 하드웨어 부분으로 제한하여 모델 체킹을 수행하는 방법을 소프트웨어 부분까지 확대하고자 한다.

참고 문헌

[1] W.H.Wolf, *"Hardware-software co-design of embedded systems"*, Proc.IEEE, vol. 82., pp.967~989, July 1994

[2] SystemC.org. *SystemC Version2.0 User's Guide*, SystemC.org, 2002

[3] Andrew Harry, *Formal Methods-FACT FILE, VDM and Z, Wiley,* 1996

[4] Kenneth L.McMillan, *Symbolic Model Checking,* Kluwer Academic Publisher, 1993

[5] Tiziano Villa, Gitanjali Swamy, Thomas Shiple, *VIS User's Manual,* University of California, Berkeley, 1996

[6] Yuji Kukimoto, *BLIF-MV,* The VIS Group, University California, Berkely, May 1996

[7] http://www-cad.eecs.berkeley.edu/~polis./

[8] .Berry, *The Esterel v5 Language Primer Version v5_91,* INRIA,June 5, 2000

[9] Edward A. Lee, "*Overview of the Ptolemy Project,*" *Technical Memorandum UCB/ERL M01/11,* University of California, Berkeley, March 6, 2001

[10] Palnitkar Samir, *Verilog HDL*, Prentice Hall, 1996