임베디드 시스템 설계를 위한 FPGA 콘트롤러 자동 생성

손춘호° 윤정한 한태숙
한국과학기술원 전자전산학과 전산학전공
{chson°, dolgam}@pllab.kaist.ac.kr, han@cs.kaist.ac.kr

Automatic FPGA Controller Generation in Embedded System Design

Choonho Son° Jeong-Han Yun Taisook Han
Division of Computer Science, Dept. of Electrical Engineering & Computer Science,
KAIST

요 약

임베디드 시스템은 소프트웨어와 특수한 기능을 추가한 하드웨어의 긴밀한 결합으로 이루어진다. 이러한 임베디드 시스템 설계는 빠른 시간 안에 안전하면서 다양한 하드웨어와 소프트웨어를 동시에 설계하여 테스트하는 것이 중요하다. 이를 위해서 개발 초기에는 다양한 하드웨어를 설계할 수 있는 FPGA가 사용된다. 이러한 FPGA 를 소프트웨어와 연결시켜주는 하드웨어 콘트롤러와 디바이스 드라이버 설계는 임베디드 시스템 개발에 있어서 중요한 부분을 차지한다. 본 논문은 FPGA를 포함하는 임베디드 시스템의 구조를 모델링하여, 하드웨어의 기능에 따라 각 하드웨어를 제어하는 콘트롤러를 자동으로 생성하는 방법을 제안하였다.

1. 서 론

임베디드 시스템이 널리 사용되고 관련 기술들의 빠른 발전에 따라, 빠른 시간 안에 안전하면서 다양한 기능의 임베디드 시스템을 개발할 수 있는 능력이 중요시 되고 있다.[1] 이러한 임베디드 시스템의 사용 영역이 넓어진 것에 비하여 아직까지 임베디드 시스템 개발 지원 도구[3][4]에 대한 연구는 부족한 실정이다.

임베디드 프로세서가 제공하는 기능 이외의 다양한 기능을 개발하기 위해서 대부분의 임베디드 개발 보드에는 하드웨어를 설계하고 테스트 할 수 있는 FPGA (Field Programmable Gate Array)가 부착되어 있다. FPGA를 이용하여 다양한 기능의 하드웨어는 소프트웨어와 연동하여 임베디드 시스템으로 설계된다.

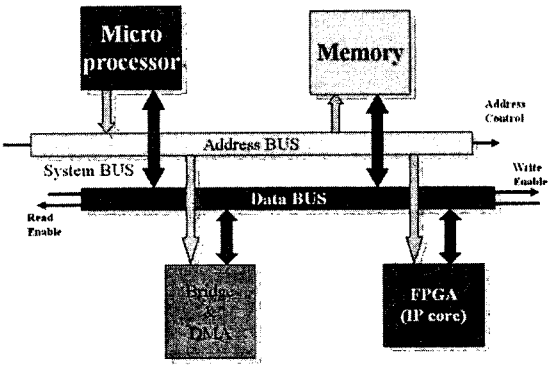
하지만 이러한 하드웨어와 소프트웨어를 연결시켜주기 위해서는 하드웨어의 특징에 따른 다양한 콘트롤러를 만들어 주어야 한다.

본 논문은 2장에서 FPGA를 포함하는 임베디드 시스템의 구조를 모델링하고, FPGA에서 구현되는 하드웨어의 종류에 따른 콘트롤러의 구성을 3장에서 분류하였다. 이러한 정보를 바탕으로 자동으로 하드웨어 콘트롤러를 만들어주는 방법에 대해서는 4장에서 설명하였다. 5장과 6장에서는 본 자동 생성 시스템을 바탕으로 인텔 XScale[2] 임베디드 프로세서와 연동한 실험과 결론을 제시하겠다.

본 연구는 대학 IT연구센터 육성.지원 사업의 연구 결과로 수행 되었음.

2. FPGA를 포함하는 임베디드 시스템 구조

임베디드 시스템은 어떤 프로세서를 사용하고, 어느 정도의 기능이 필요하냐에 따라 운영체제가 있는 시스템과 펌웨어를 통해서 돌아가는 시스템 등의 다양한 방법으로 구현된다. 하지만 이러한 소프트웨어적인 차이에도 불구하고 대부분의 하드웨어는 일반적인 컴퓨터 구조를 따라 설계되고 있다. <그림 1>은 일반적인 FPGA를 포함하는 임베디드 시스템을 보여주고 있다.



<그림 1> FPGA를 포함하는 임베디드 시스템 구조

위의 그림과 같이 일반적인 임베디드 시스템에서의 FPGA는 시스템 버스에 연결되어 임베디드 프로세서가 처리할 수 없는 일들을 도와준다. 따라서 이러한 시스템 버스에 연결되어 있는 하드웨어를 사용하기 위해서 어드

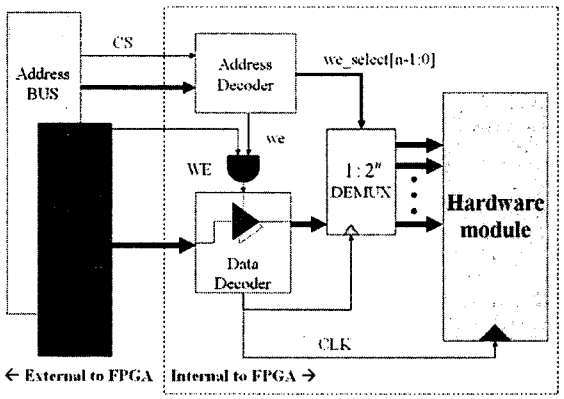
레스 버스 디코더와 데이터 버스 디코더로 이루어진 콘트롤러가 임베디드 프로세서와 FPGA 로직을 연결 시켜 주어야 한다. [5]

3. 임베디드 하드웨어 분류

임베디드 시스템 설계시 FPGA에 구현되는 하드웨어는 임베디드 프로세서의 관점에서 크게 3가지로 구분 할 수 있다.

3.1 Write Only 하드웨어

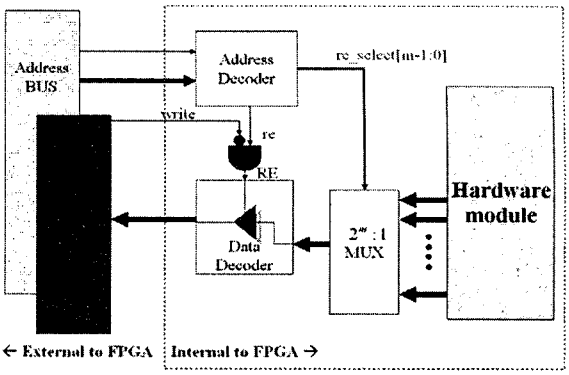
이 하드웨어는 임베디드 프로세서에서 FPGA 하드웨어 모듈에게 데이터를 전달하기만 한다. 임베디드 프로세서는 어드레스 버스에 FPGA의 물리 주소를 실어주고, 어드레스 버스 제어 시그널을 설정한다. 이와 동시에 데이터 버스에 하드웨어로 보낼 데이터를 실고 데이터 버스 제어 시그널을 설정한다. FPGA 콘트롤러는 어드레스 버스의 제어 시그널과 어드레스 버스의 데이터를 통해서 자신의 하드웨어 모듈에 전달할 데이터가 있음을 알고, 데이터 버스로부터 데이터를 읽어서 해당 하드웨어 입력 핀에 전달하게 된다. 이러한 방식으로 동작하기 위해서는 <그림 2>와 같은 어드레스 디코더와 데이터 디코더를 생성해야 한다.



<그림2> Write Only 콘트롤러 블록 다이어그램

3.2 Read Only 하드웨어

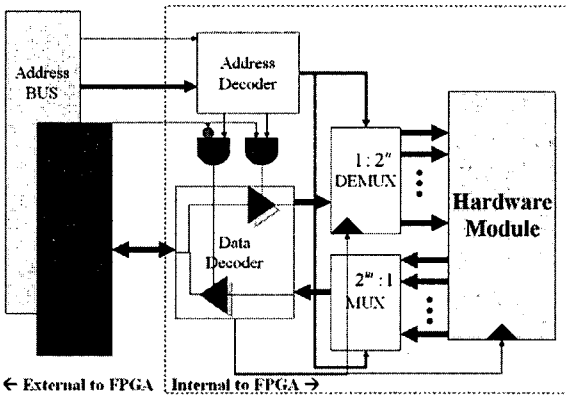
Read Only 하드웨어는 Write Only 하드웨어와는 반대로 하드웨어 모듈에서 발생한 데이터를 임베디드 프로세서가 읽기를 수행해야 한다. 이는 어드레스 디코더는 Write Only 로직과 동일하게 작동하고, 데이터 디코더는 반대로 하드웨어에서 데이터를 읽어서 데이터 버스에 실어주어야 한다. 이러한 방식으로 동작하기 위해서는 <그림 3>과 같은 회로를 설계하여야 한다.



<그림 3> Read Only 콘트롤러 블록 다이어그램

3.3 Read / Write 하드웨어

Read/Write 하드웨어는 앞에서 설명한 두가지의 콘트롤러의 조합으로 이루어진다. 하지만 데이터 버스를 임베디드 프로세서와 하드웨어 모듈 모두가 사용할 수 있기 때문에 둘 사이의 충돌을 막기 위한 방법이 필요하다. 이는 <그림4>와 같이 어드레스 버스 디코더에 의해서 데이터 버스 디코더는 write 또는 read 중에 한번에 하나씩만을 수행하게 된다.



<그림4> Read/Write 하드웨어

4. 임베디드 시스템 정보 및 자동 생성 도구

시스템 버스와 연결되는 하드웨어 콘트롤러를 설계하기 위해서는 각각의 버스에 대한 정보를 가지고 있어야 한다. 본 자동 생성 도구는 하드웨어-소프트웨어 동시 설계 지원 도구의 일부분으로 설계되었기 때문에 다른 도구와의 원활한 정보 공유를 위해서 각 입력 정보들은 XML 형식으로 저장한다. 필요한 정보는 어드레스 버스의 폭과 해당 FPGA를 제어하는 제어 시그널의 이름과 타입이다. 데이터 버스 역시 버스의 폭과 제어 시그널의 이름과 타입이 필요하다. 또한 어드레스 버스 정보에는 하드웨어 모듈이 쓰기 전용,읽기 전용,읽기쓰기 용인지를 구분 해 줄 수 있는 정보를 포함하게 된다.

본 자동 생성 도구는 이러한 XML로 저장된 임베디드 시

스템 정보를 바탕으로 FPGA에 합성 가능한 Verilog 코드를 생성하게 된다.

5. 임베디드 시스템 예제 및 결과

본 시스템은 Intel PXA255 임베디드 프로세서와 ALTERA Cyclone FPGA가 장착된 한백 전자의 EMPOS II 보드[6]에서 실험 및 테스트를 하였다.

5.1 8 bit LED 콘트롤러 예제

FPGA 보드에는 8개의 LED가 있다. 이 8개의 LED를 소프트웨어에서 제어할 수 있도록 해주는 Write Only 콘트롤러는 <그림5>와 같이 두개의 임베디드 시스템 정보를 입력으로 필요로 한다.

```
<AddrDecoder>
  <addr id=" CX_A"  width=" [21:1]" />
  <chip_select id=" NPX_CS5"  type=" active low" />
  <write from=" 0"  to=" 0" />
</AddrDecoder>

<DataDecoder>
  <data id=" CX_D"  width=" [15:0]" />
  <write_enable id=" NPX_PWE"  type=" active low" />
</DataDecoder>
```

<그림5> 8bit LED 어드레스, 데이터 디코더 정보

5.2 바이트 스왑퍼 예제

본 예제는 2바이트 little endian을 2바이트 big endian으로 변환시켜주는 예제에 대한 임베디드 시스템 정보이다. 소프트웨어는 little endian의 2바이트를 하드웨어의 입력으로 주면 2바이트의 big endian과 read_enable 시그널을 소프트웨어에 전달해 줄 수 있는 read/write 콘트롤러를 만들어 준다.

```
<AddrDecoder>
  <addr id="CX_A" width="[21:1]" />
  <chip_select id="NPX_CS5" type="active low" />
  <write from="0" to="1" />
  <read from="2" to="3" />
</AddrDecoder>

<DataDecoder>
  <data id="CX_D" width="[15:0]" />
  <write_enable id="NPX_PWE" type="active high" />
</DataDecoder>
```

<그림6> byte swapper 어드레스, 데이터 디코더 정보

5.3 실험 결과

위 두가지의 임베디드 시스템 정보를 바탕으로 자동 생성 도구가 만들어 주는 콘트롤러의 Verilog 코드의 라인수는 <그림7>에 나타나 있다. 임베디드 시스템의 설계시 다양한 하드웨어를 설계하고 테스트하는 데 있어 본 논문에서 제시하는 것과 같은 자동 콘트롤러 생성 도구가 없다면, 개발자들은 어드레스 / 데이터 디코더와 이를 연결하는 최상위 모듈 코드들을 일일이 수작업으로 작성하여야 한다.

종류	어드레스 디코더	데이터 디코더	최상위 모듈	총합
8 bit LED	11	66	21	98
바이트스왑퍼	17	87	30	134

<그림7> 자동 생성된 Verilog code의 라인수

6. 결론

본 논문은 FPGA를 포함하는 임베디드 시스템의 모델을 제시하고 다양한 임베디드 시스템을 설계시 여러가지 하드웨어들을 안전하고 빠르게 생성하고 테스트할 수 있는 방법을 제시하였다. 이를 바탕으로 향후에는 본 연구를 바탕으로 콘트롤러 뿐만 아니라, FPGA하드웨어를 위한 디바이스 드라이버 모델을 제시하고 이를 자동으로 생성하는 방법에 대한 연구도 이루어 져야겠다.

또한 하드웨어와 소프트웨어를 동시에 설계하고 테스트하는 것 뿐만 아니라, 설계 초기에 소프트웨어와 하드웨어의 안정성에 대해서도 검증할 수 있도록 하드웨어의 제약 조건, 상태 등을 받아서 검증해주는 시스템으로 발전도 기대해 볼 만 하다.

7. 참고 문헌

[1] Choonho Son. Jeong-Han Yun, Hyun-Goo Kang, and Taisook Han, " Hardware/Software Interface Generation for Embedded System using Hardware Interface Automata", *Proceedings of the 4th International Conference on Asian Language Processing and Information Technology*, 2005.

[2] " *Intel® PXA255 Processor Design Guide*", Intel, March 2003.

[3] F. Balarin, A. Jurecska, and H. Hsieh et al, " *Hardware-Software Co-Design of Embedded System: The Polis Approach*", Kluwer Academic Press, Boston, 1997.

[4] Cristiano C. de Araujo, and Edna Barros, " Interface Generation for Concurrent Processes During Hardware/Software Co-synthesis", *Proceedings of the 15th Symposium on Integrated Circuits and Systems Design*, 2002.

[5] Randy H. Katz, " *Contemporary Logic Design*", The Benjamin/Cummings Publishing Company.

[6] Hanback electronics homepage, http://www.hanback.co.kr