논문 2008-45SD-9-5

SoC를 위한 고성능 NAWM 버스 아키텍처

(NAWM Bus Architecture of High Performance for SoC)

이 국 표*, 윤 영 섭*

(Kook-Pyo Lee and Yung-Sup Yoon)

요 약

전형적인 공용버스 아키텍처는 동일시간에 하나의 데이터 전송을 처리할 수 있다. 본 논문에서는 동일시간에 여러 데이터 전송을 할 수 있는 NAWM (No Arbitration Wild Master) 버스 아키텍처를 제안하고 있다. AMBA 시스템에 대하여 NAWM 버스아키텍처의 마스터 래퍼와 슬레이브 래퍼를 설계해 보았으며, AMBA 시스템의 대부분 IP들을 수정없이 적용하는 것이 가능하다는 사실과 추가되는 타이밍 지연은 무시가능하다는 것을 확인하였다. 시뮬레이션을 통하여 NAWM 버스 아키텍처에서 여러 마스터들이 슬레이브에 접근할 때, 50% 이상 병렬처리가 가능함을 알 수 있었다.

Abstract

The conventional shared bus architecture is capable of processing only one data transaction in same time. In this paper, we propose the NAWM (No Arbitration Wild Master) bus architecture that is capable of processing several data transactions in same time. After designing the master and the slave wrappers of NAWM bus architecture about AMBA system, we confirm that most of IPs of AMBA system can be applied without modification and the added timing delay can be neglected. From simulation we deduce that more than 50% parallel processing is possible when several masters initiate slaves in NAWM bus architecture.

Keywords: bus architecture, architecture performance, SoC, AMBA

I. 서 론

여러 가지 기능을 갖는 컴포넌트를 하나의 칩에 집적화시킨 SoC칩은 계속적으로 발전하고 있으며, 더욱더 복잡하게 진화하고 있다.[1~5] 컴포넌트들을 함께 사용하기 위해서 공용버스(Shared Bus) 아키텍처가 제안되고 있는데, 성능을 향상시키기 위해 구조가 다양화되고 있으며 새로운 구조도 계속 개발되고 있다.[1~5] 세계적인 회사에서는 AMBA[6], Core Connect[7], Silicon MicroNetworks[8] 등의 버스 아키텍처를 개발하여 상용화하고 있다. 전 세계적으로 70% 이상 사용되고 있는 AMBA 시스템에서는 여러 버스를 계층적으로 구분하고 버스 사이에 브릿지로 연결하여 사용하고 있다. 이와 같은 방법을 사용할 경우, 컴포넌트의 증가에 따른 공용버스의 성능저하를 일부 감소시킬 수 있다. 그러나 버스 간의 데이터 전송이 발생할 때, 브릿지 블록에서 레이턴시가 증가하는 단점을 가지고 있다.

버스성능을 증가시키기 위해서 버스 중재장치인 아비터 부분의 개선이 이루어지고 있다. 우선순위 기반 중재(priority-based arbitration)[6], 라운드-로빈 중재(round-robin arbitration)[6], TDMA(time-division multiple access)[8], CDMA(code-division multiple access)[9] 방법 등이 개발되고 있다. 그러나 버스 중재의 개선으로 버스를 효율적으로 사용할 수는 있지만, 실질적인 성능향상으로 발전하기에는 한계가 있다.

획기적인 성능향상을 위해서는 여러 컴포넌트가 동시에 데이터를 전송하는 병렬처리 방법이 도입되어야

* 정회원, 인하대학교 전자공학과
(Dept. of Electronics Engineering, Inha University)
접수일자: 2008년7월8일, 수정완료일: 2008년8월28일

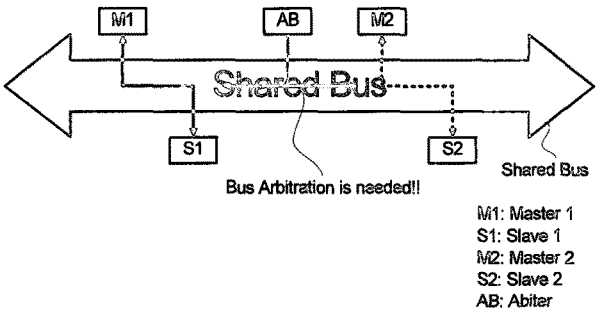
하지만, 동일시간에 하나의 데이터전송만 가능한 공용 버스 구조 하에서는 원칙적으로 불가능하다. 기존 공용 버스에 대한 대안으로 슈퍼컴퓨터나 서버 제작에 사용되는 네트워크 구현법을 도입한 NoC (Network on Chip)가 연구되고 있으며, 이를 보다 간단하게 구조화한 SNP(SoC Network Protocol)가 제안되고 있다.[9~10] 그러나 아직까지 기존 공용버스에 비해 복잡하고 오버헤드가 비교적 크다.

본 연구에서는 기존의 IP와 버스 아키텍처를 그대로 사용하면서, 간단하게 버스의 병렬처리가 가능한 새로운 방법에 대해서 제안하려고 한다.

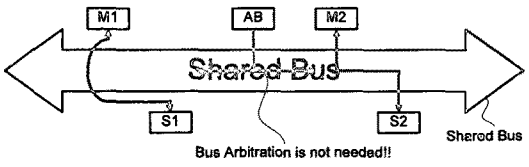
Ⅱ. 본 론

1. NAWM 버스 아키텍처 제안

버스 아키텍처 컴포넌트에서 마스터는 데이터 전송을 발생시키는 역할을 하고, 슬레이브는 마스터에서 보낸 데이터 전송을 응답하는 역할을 한다. 그림 1(a)에 일반적인 공용버스에 대한 마스터와 슬레이브의 데이터 전송이 나타나 있다. 간단하게 특징을 비교하기 위해 마스터와 슬레이브의 개수를 2개만 나타내었다. 마스터 M1이 슬레이브 S1에 데이터 전송하고, 마스터 M2가 슬레이브 S2에 데이터 전송을 할 경우, 공용버스에서 동시에 전송을 수행할 수 없으므로 아비터 AB의 중재



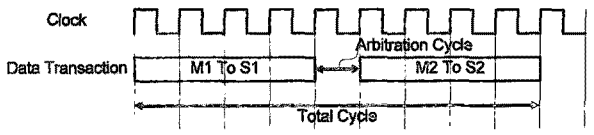
(a)



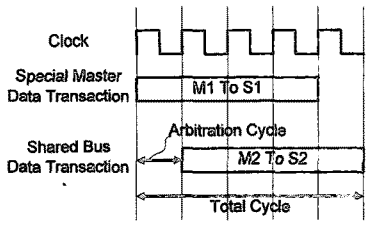
(b)

그림 1. (a) 전형적인 데이터 전송 버스구조,
(b) 제안하는 데이터 전송 버스구조

Fig. 1. (a) Bus architecture of conventional data transaction, (b) Bus architecture of proposed data transaction.



(a)



(b)

그림 2. (a) 전형적인 데이터 전송의 타이밍도,
(b) 제안하는 데이터 전송의 타이밍도

Fig. 2. (a) Timing Diagram of conventional data transaction, (b) timing Diagram of proposed data transaction.

를 받게 된다. 그림 1(a)에서는 마스터 M1이 전송되고 마스터 M2는 전송을 하지 못하고 있다. 마스터 M1의 전송이 완료된 후, 비로소 마스터 M2의 전송이 시작될 수 있다.

그림 1(b)에는 본 논문에서 제안하는 데이터 전송방법이 나타나 있다. 마스터 M1에서 슬레이브 S1으로 데이터 전송될 경우에는 공용버스를 거치지 않고 직접 처리되고, 마스터 M2에서 슬레이브 S2으로 데이터 전송될 경우에는 공용버스를 거치게 된다. 이 경우, M1과 M2의 데이터 전송을 동시에 병렬 처리할 수 있어서 성능을 크게 향상시킬 수 있다.

그림 2에 보듯이, 일반적으로 데이터 전송할 경우 9 클럭 사이클이 소요되지만, 제안하는 방법으로 데이터 전송할 경우 5 클럭 사이클이 소요된다. 결국 마스터 M1과 다른 마스터가 동시에 다른 슬레이브에 전송될 경우, 마스터 M1의 전송과 다른 마스터의 전송이 병렬로 이루어질 수 있으며, 마스터 M1의 전송 사이클만큼 전체 사이클이 감소될 수 있다. 그러므로 가장 많이 사용되는 마스터를 M1 마스터로 설정하고 시스템을 구성한다면 성능을 획기적으로 향상시킬 수 있다.

본 논문에서는 특별하게 정해진 마스터에게 버스 중재와 독립적으로 슬레이브에 접근할 수 있게 규정하고, 이를 NAWM (No Arbitration Wild Master) 버스 아키텍처라고 새롭게 정의하였다. NAWM 버스 아키텍처를 적용한 IEEE 802.11 네트워크 SoC의 예가 그림 3에 나타나 있다.[12] 버스 중재가 필요없는 마스터를 ARM 프로세서로 설정하였으며, DMA(Direct Memory

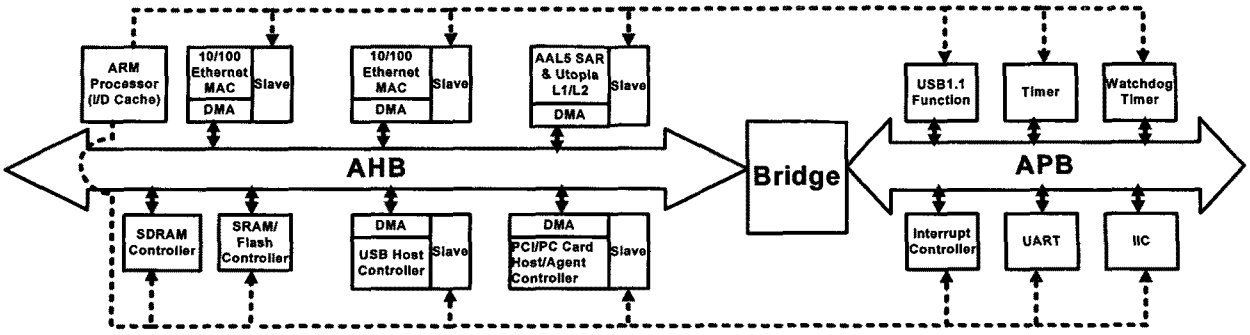


그림 3. NAWM 버스 아키텍처를 적용한 SoC의 예
Fig. 3. SoC example applying NAWM bus architecture.

Access) 로 데이터 전송하는 Ethernet MAC, SAR (Segmentation and Reassembly), PCI/PC Card, USB Host 등의 마스터는 공용버스 AHB에 연결되어 있다. DMA 등으로 데이터 전송하는 마스터는 데이터 전송 방법 등을 입력받아야 하므로 반드시 슬레이브가 포함된다. 그래서 그림 3에서 보듯이 마스터의 경우 내부에 슬레이브를 포함시켰다.

일반적으로 SoC에서 프로세서는 가장 많은 연산을 수행하고, 공용버스의 사용률이 다른 마스터보다 높을 뿐만 아니라, 신속하게 처리해야 한다.

NAWM 버스 아키텍처에서 그림 3과 같이 ARM 프로세서를 최우선적으로 처리해야 하는 마스터로 규정하고, 공용버스의 아비터 중재와 무관하게 SDRAM, SRAM/Flash, USB, IIC 등의 슬레이브에 접근할 수 있도록 하였다. 결국, ARM 프로세서는 와일드 마스터로서 가장 신속하게 명령을 처리할 수 있다. 그리고 DMA를 이용한 마스터와 동시에 서로 다른 슬레이브에 접근할 경우 동시에 병렬 수행이 가능하므로, 프로세서의 클럭 사이클만큼 전체 사이클이 감소하고, 성능도 크게 향상시킬 수 있다.

다음 장에서 구체적으로 NAWM 버스 아키텍처를 설계해 보고, 기존의 방식과 비교하여 구체적인 성능분석을 할 것이다.

2. NAWM 버스 아키텍처 설계

간단하게 NAWM 버스 아키텍처를 파악하기 위해, 마스터 3개와 슬레이브 3개의 경우에 대한 NAWM 버스 아키텍처의 데이터 전송 흐름도를 그림 4에 나타내었다. 모든 마스터는 3개의 슬레이브에 모두 접근할 수 있으며, 마스터 M2, M3는 공용버스를 통하여 슬레이브에 접근할 수 있다.

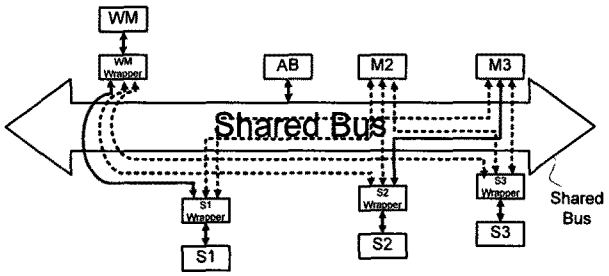


그림 4. NAWM 버스 아키텍처의 데이터 전송
Fig. 4. Data transaction of NAWM bus architecture.

와일드 마스터 WM은 공용버스와 무관하게 3개의 슬레이브에 접근할 수 있으며, 이를 조절하기 위해서 마스터 WM 부분에 WM 래퍼와 슬레이브 부분에 슬레이브 래퍼 블록이 추가되었다. WM 래퍼는 슬레이브와 직접 데이터 전송을 위한 신호를 조절하고, 슬레이브 래퍼는 마스터 WM에 의해 전송되는 데이터와 공용버스에 의해 전송되는 데이터를 처리한다. 마스터 WM은 모든 슬레이브에 대해 공용버스 없이 데이터를 전송하므로 모든 슬레이브는 반드시 슬레이브 래퍼를 추가해야 하며, 슬레이브를 포함하고 있는 마스터에도 슬레이브 래퍼를 슬레이브와 연결해 주어야 한다. 본 논문에서는 전 세계의 70% 이상이 사용하고 있는 AMBA 시스템으로 NAWM 버스 아키텍처를 설계하려고 한다.

그림 5에 AMBA 시스템에 대한 마스터 래퍼의 블록도가 나타나 있다. 마스터 래퍼는 마스터로부터 발생하는 AMBA 신호들과 슬레이브로부터 발생하는 AMBA 신호를 처리해야 한다. 회로를 최대한 간단히 구성하여 블록 간의 지연시간을 최대한 줄여 고주파수에서 동작하도록 고려하였다. 마스터에서 슬레이브로 가는 모든 신호는 bypass시켰다. 그리고 주소값 "HADDR" 신호로 해당 슬레이브 블록을 선택하는 "HSEL" 신호는 AMBA 버스에서 공용버스의 디코더 블록에서 발생된

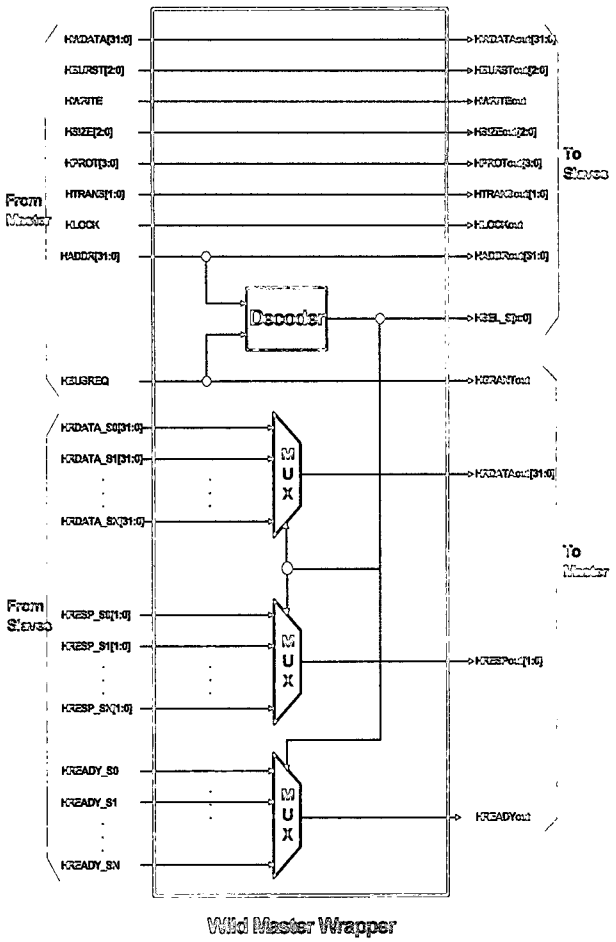


그림 5. 와일드 마스터 래퍼 블록도
Fig. 5. Block diagram of wild master wrapper.

```
.....
HSEL[0] = (HADDR[31:28]==4'b1111) && HBUSREQ;
HSEL[1] = (HADDR[31:28]==4'b1110) && HBUSREQ;
HSEL[2] = (HADDR[31:28]==4'b1101) && HBUSREQ;
HSEL[3] = (HADDR[31:28]==4'b1100) && HBUSREQ;
.....
```

그림 6. 와일드 마스터 래퍼의 디코더에 대한 Verilog code의 일부분
Fig. 6. A part of Verilog code about decoder of wild master wrapper.

다. 마스터 WM은 AMBA 버스를 거치지 않으므로 디코더 블록을 새로 만들어야 한다. 그림 6에 Verilog로 코딩된 디코더 블록의 일부분이 나타나 있다. 주소 HADDR 신호가 32'hFXXX_XXXX, 32'hEXXX_XXXX, 32'hDXXX_XXXX, 32'hCXXX_XXXX 일 때, 각각 HSEL[0], HSEL[1], HSEL[2], HSEL[3]이 선택되며, 버스요청신호 HBUSREQ 신호에 의해서 조절된다. 마스터 래퍼는 AMBA 버스를 거치지 않고 슬레이브로부터 직접 HRESP, HREADY, HRDATA[31:0]의 신호를 받게 되는데, 이는 HSEL[x] 신호를 기준으로 먹스 로직에 의해 선택되거나 AND 연산을 거쳐서 마스터 블록에 전달된다.

본 논문에서 설계한 마스터 래퍼는 대부분의 신호를 bypass시켰으며, 디코더, 먹스, AND 등 일부 신호지연이 적은 게이트가 추가되었다. 이는 슬레이브 신호가 마스터 블록에 전달될 때 AMBA 버스에서 원래 만들어 주어야 하는 부분이므로, 새롭게 신호지연이 발생한 것은 아니다. 결국 마스터 래퍼를 추가로 연결해도, 실제적으로 발생하는 신호지연은 거의 없으며, 전체 주파수 감소에 영향을 크게 주지 않는다.

그림 7과 그림 8에 슬레이브 래퍼에 대해 나타나 있다. 마스터 WM에서 전송되는 모든 데이터는 공용버스를 거치지 않고 슬레이브에 접근하므로, 모든 슬레이브는 반드시 슬레이브 래퍼를 추가해야 한다. 슬레이브 래퍼는 마스터 WM과 공용버스에서 AMBA 신호를 받으므로, 슬레이브에 전달해야할 신호를 선택해야 한다. "HSEL_Selected" 신호를 기준으로 먹스 회로에서 슬레이브에 전달할 신호를 선택하며, "HSEL_Selected" 신호는 다음과 같다.

$$HSEL\_Selected = (HTRANS[1:0] \neq 'SEQ) \wedge HSEL\_S[x] \quad (1)$$

여기서, "HTRANS[1:0]"는 공용버스에서 발생하는 AMBA transaction 신호이고, "HSEL_S[x]"는 마스터 WM에서 발생하는 슬레이브 선택신호이고, "'SEQ"는 버스에서 데이터를 전송 중인 경우를 의미하는 파라미터이다. 공용버스의 데이터를 전송하지 않을 경우 ("HTRANS[1:0]"가 SEQ가 아닐 때)에 마스터 WM에 의해 해당 슬레이브가 선택("HSEL_S[x]"가 "1"일 때) 되면, "HSEL_Selected"가 "1"의 값을 갖게 되며 마스터 WM의 전송이 선택된다.

슬레이브에서 발생하는 "HREADY" 신호는 현재 접근하고 있는 마스터가 마스터 WM일 경우와 다른 마스터일 경우에 따라 "HREADYoutWM" 신호 또는 "HREADYoutBus" 신호로 나뉘어지며, "HREADYout WM" 신호의 값은 다음과 같이 결정된다.

1) HTRANS[1:0]=='SEQ) ∧ HSEL_S[x] 이면 0을 갖는다.
2) 1)이 아닐 경우, "HSEL_Selected" 신호가 "1"이면 HREADY 값을 갖는다.
3) 1),2)가 아니면 "1"을 갖는다.

그리고 "HREADYoutBus" 신호의 값은 다음과 같이 결정된다.

1) "HSEL_Selected" 신호가 "0"이면 HREADY 값을 갖는다.
2) 1)이 아닐 경우, HTRANS[1:0]=='NONSEQ 이면 "0"을 갖는다.
3) 1),2)가 아니면 "1"을 갖는다.

HRESP[1:0], HRDATA[31:0]은 bypass시켰으며, 슬레이브가 선택되고 있을 의미하는 "HSELout" 신호는 다음과 같다.

$$HSELout = HSEL\_x\_Bus \vee HSEL\_S[x] \quad (2)$$

슬레이브 래퍼 블록은 마스터 WM과 다른 마스터의 신호를 선택하기 위한 먹스회로와 마스터에게 전달하는 "HREADY" 신호 생성을 위해 수 개의 게이트가 소요되었다. 그러나 마스터 래퍼와 마찬가지로 실제 신호 지연시간은 미미한 수준이며, 전체 칩 사이즈에도 큰 영향을 주지 않는다.

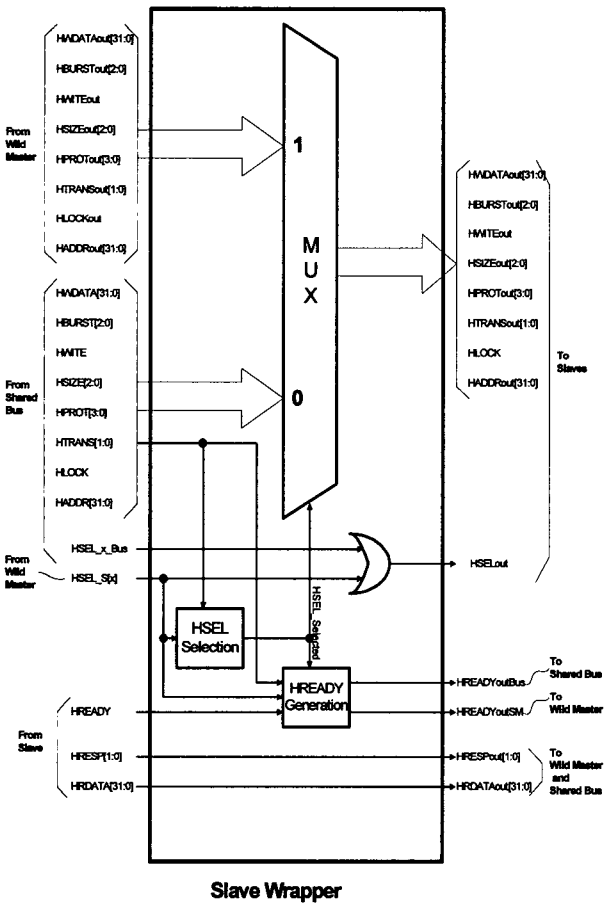


그림 7. 슬레이브 래퍼 블록도
Fig. 7. Block diagram of slave wrapper.

```
parameter NONSEQ 2'b10
parameter SEQ    2'b11
.....
HSEL_Selected = (HTRANS[1:0] !='SEQ) && HSEL_S[x];
.....
HREADYoutWM = HTRANS[1:0]=='SEQ) && HSEL_S[x] ? 1'b0 :
                                  HSEL_Selected ? HREADY
:
                                                  1'b1;
.....
HREADYoutBUS = !HSEL_Selected          ? HREADY :
                HTRANS[1:0]=='NONSEQ ? 1'b0 : 1'b1;
.....
```

그림 8. 슬레이브 래퍼의 "HSEL Selection"과 "HREADY Generation" 블록에 대한 Verilog code의 일부분
Fig. 8. A part of Verilog code about "HSEL Selection" and "HREADY Generation" blocks of slave wrapper.

3. NAWM 버스 아키텍처 성능분석

그림 9에 보듯이 SoC에 대한 성능을 파악하는 방법은 크게 동적인 시뮬레이션과 정적인 분석으로 나뉠 수 있다. 동적인 시뮬레이션은 정확도는 높일 수 있으나, 효율이 낮아서 성능을 파악하는데 비교적 장시간이 소요된다. 반면에 정적인 분석은 빠른 시간에 성능 예측이 가능하여 효율적이지만, 정확도가 상대적으로 낮다.

본 논문에서는 NAWM 아키텍처의 정적인 분석을 통해 성능에 대한 분석을 실시하여, 기존 아키텍처 대비 성능향상 여부를 파악하고 성능에 대한 주요 인자를 분석할 것이다. 그리고 다음 논문에 동적인 시뮬레이션 결과를 추가하고, NAWM 아키텍처의 성능을 종합적으로 분석하려고 한다.

NAWM 아키텍처에서는 와일드 마스터와 일반 마스터가 서로 다른 슬레이브에 접근할 경우에 병렬처리 가능하고, 와일드 마스터의 처리 사이클만큼 성능을 개선시킬 수 있다. 식 (3)과 같이 전체 확률에서 와일드 마

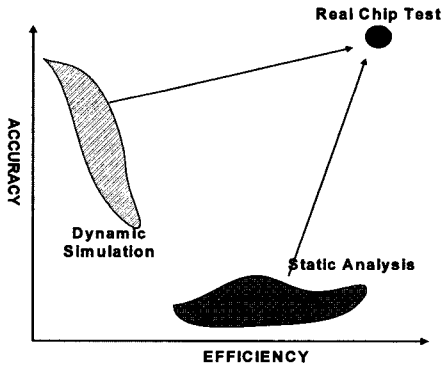


그림 9. SoC에 대한 정적인 분석과 동적인 시뮬레이션의 특징
Fig. 9. Difference of static analysis and dynamic simulation about SoC.

스터와 일반 마스터가 동시에 동일한 슬레이브에 접근할 확률을 제외하면 병렬처리 확률을 얻을 수 있다.

$$P = 1 - \sum_{n,p=1}^{n,p} P_{WMSp} \circ P_{MnSp} \quad (3)$$

여기서, P_{WMSp}는 와일드 마스터 WM이 슬레이브 Sp에 접근할 확률, P_{MnSp}는 와일드 마스터 외의 마스터 Mn이 슬레이브 Sp에 접근할 확률이다.

슬레이브가 13개, 마스터가 5개 존재하는 그림 3의 SoC 구조에서 와이드 마스터와 다른 마스터가 동시에 서로 다른 슬레이브에 접근할 확률은 그림 10와 같다. 실제 시스템을 고려하여 그림 3의 IEEE 802.11 네트워크 SoC에서 와일드 마스터 ARM 프로세서가 슬레이브 SDRAM에 접근할 확률이 제일 높고 다른 슬레이브들의 접근할 확률은 모든 슬레이브가 동일하다고 가정하였으며, 다른 슬레이브들의 접근할 확률은 동일하다고 가정하였다. 그리고 ARM 프로세서 외의 마스터들은 메모리 접근을 위해 DMA를 내장한 데이터 전송이므로, 슬레이브 SDRAM과 SRAM에만 접근한다고 가정하였다. 그림 10에서 보듯이 결국 마스터들이 SDRAM에 접근할 확률에 따라 와일드 마스터가 병렬처리될 확률이 정해진다. ARM 프로세서 또는 다른 마스터들의 SDRAM 접근확률이 0%에 접근하면 병렬처리 확률이 100%에 가까워지며, ARM 프로세서와 다른 마스터들의 SDRAM 접근확률이 높아짐에 따라 병렬처리 확률이 낮아짐을 알 수 있다. 결론적으로 와일드 마스터 WM과 다른 마스터의 SDRAM 접근확률에 차이를 두면서 시스템을 운영할 때, NAWM 아키텍처의 효율이 극대화될 수 있다. 예를 들어 DMA가 내장된 마스터들은 데이터양이 많으므로 용량이 큰 SDRAM을 통한 데이터 전송을 늘리고, 프로세서는 명령코드 (instruction code) 와 데이터 코드를 SRAM으로 리맵(remap)시켜서 SDRAM 보다 SRAM의 접근확률을 높이는 것이 성능을 향상시킬 수 있는 적절한 대안이 될 수 있다.

그림 11에 보면 슬레이브의 개수가 약 5개 이상에서 병렬처리의 확률이 포화됨을 알 수 있다. 결국 NAWM 아키텍처는 5개 정도의 기본 슬레이브만으로 병렬처리 확률을 최대로 높이면서, 성능을 향상시킬 수 있음을 알 수 있다.

III. 결 론

본 논문에서는 동일 시간에 한 번의 데이터 처리만

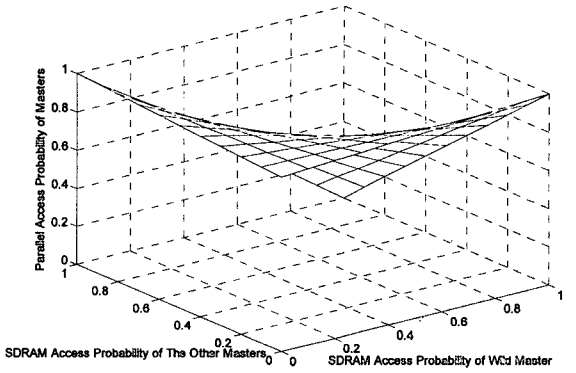


그림 10. SDRAM 접근에 따른 마스터들의 병렬처리 확률
Fig. 10. Parallel access probability of masters due to SDRAM access.

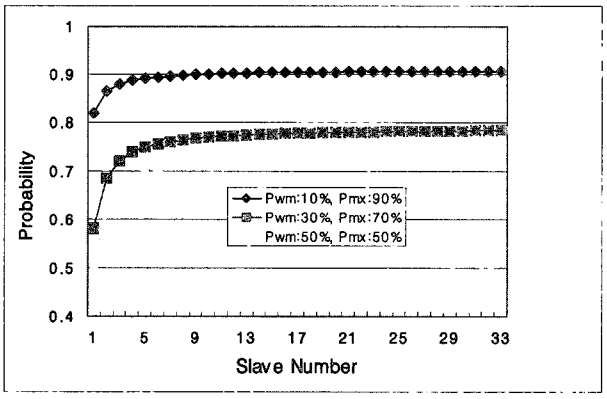


그림 11. 슬레이브 수에 따른 마스터들의 병렬처리 확률 (Pwm: 와일드 마스터의 SDRAM 접근확률, Pmx: 다른 마스터들의 SDRAM 접근확률)
Fig. 11. Parallel access probability of masters due to slave numbers. (Pwm: SDRAM access probability of wild master, Pmx: SDRAM access probability of the other masters)

가능한 기존 공용버스를 개선하여, 여러 데이터가 병렬처리될 수 있는 NAWM 버스 아키텍처를 제안하였다. 그리고 기존 IP를 수정없이 사용가능하며, 설계상의 타이밍 마진에 큰 영향을 미치지 않았음을 확인하였다. AMBA 시스템에 대하여 마스터, 슬레이브 래퍼를 설계해 보고 특징을 파악해 보았다. 마지막으로 성능에 대한 정적인 분석을 통해, NAWM 버스 아키텍처의 병렬처리가 효율적임을 입증하였다.

다음 논문에서는 NAWM 버스 아키텍처에 대한 동적인 시뮬레이션을 통해 세밀하게 성능을 분석해 보고, 특징들을 파악할 계획이다.

참 고 문 헌

[1] R. Lu and C.-K. Koh, "SAMBA-Bus: A High Performance Bus Architecture for System-on-Chips", IEEE Trans. on VLSI Systems, vol. 15, no. 1, pp.69-79, 2007.

[2] E. Salminen, V. Lahtinen, K. Kuusilinna, and T. Hamalainen, "Overview of bus-based system-on-chip interconnections", in Proc. IEEE Int. Symp. Circuits Syst., pp. II-372-II-375, 2002.

[3] K. Lahiri, A. Raghunathan, and G. Lakshminarayana, "The lotterybus on-chip communication architecture", IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 14, no. 6, pp.596-608, 2006.

[4] R. Lu and C.-K. Koh, "A high performance bus communication architecture through bus splitting", in Proc. Asia-South Pacific Design Autom. Conf., pp.751-755, 2004.

[5] K. Sekar, K. Lahiri, A. Raghunathan, and S. Dey, "FLEXBUS: A high performance system-on-chip communication architecture with a dynamically configurable topology", in Proc. Design Autom. Conf., pp.571-574, 2005.

[6] ARM, Limited, Cambridge, U.K., "AMBA Specification", 1999.

[7] IBM, Armonk, NY, "CoreConnect bus architecture", 1999.

[8] Sonics, Inc., Mountain View, CA, "Silicon micronetworks technical overview", 2002.

[9] S. Shimizu, T. Matsuoka, and K. Taniguchi, "Parallel bus systems using code-division multiple access technique", in Proc. IEEE Int. Symp. Circuits Syst., pp.II-240-II-243, 2003.

[10] F. G. Moraes, N. L. V. Calazans, A. V. deMello, L. H. Moller, L. C. Ost, "Hermes: an Infrastructure for Low Area Overhead Packet-switching Networks on Chip", Integration the VLSI Journal, 38(1), Oct. 2004, pp.69-93.

[11] 이상헌, 이찬호, "다중 채널과 동시 라우팅 기능을 갖는 고성능 SoC 온 칩 버스 구조", 대한전자공학회논문지, 제44권, SD편, 제4호, pp.332-339, 2007년.

[12] http://www.samsung.com/global/business/semiconductor/productInfo.do?fmly_id=234&partnum=S3C2510A

저 자 소 개

이 국 표(정회원)
대한전자공학회 논문지
제45권 SD편 제4호 참조

윤 영 섭(정회원)
대한전자공학회 논문지
제45권 SD편 제4호 참조