무선 센서 노드상의 저가형 플래시 메모리를 위한 하드웨어 추상화 구조†

(Hardware Abstraction Architecture for Low Cost Flash Memories in Wireless Sensor Nodes)

김 창 훈*, 권 영 직*
(Chang Hoon Kim, Young Jik Kwon)

요 약 본 논문에서는 무선 센서 노드에 사용 가능한 저가형 플래시 메모리를 위한 하드웨어 추상화 구조(Hardware Abstraction Architecture: HAA)를 제안한다. 제안된 HAA는 3개의 계층으로 이루어져 있으며, 세 개의 계층은 HIL(Hardware Interface Layer), HAL(Hardware Adaption Layer), HPL(Hardware Presentation Layer)로 구성된다. 여기서 HIL은 상위 계층의 어플리케이션에 대해 플랫폼 독립적인 인터페이스를 제공하고, HAL은 하드웨어 추상계층에서 가장 핵심적인 부분으로서 하드웨어 자원 제어, 상태관리, 논리적 명령어를 생성하며, HPL은 하드웨어 초기화 및 플래시 메모리와의 통신 부분을 담당한다. 제안된 HAA는 무선 센서노드에 가장 많이 사용되고 있는 Atmel사의 AT45DB 계열의 플래시 메모리에 적용되었으며, 4,384 바이트의 프로그램 메모리와 195 바이트의 데이터 메모리를 사용한다. 따라서 본 논문에서 제안된 HAA 구조는 3계층으로 설계되었기 때문에 소프트웨어 개발 측면에서 높은 유연성, 확장성, 재사용성을 제공하며, 낮은 메모리를 사용하기 때문에 무선 센서 노드용으로 적합하다 할 수 있다.

핵심주제어 : 무선 센서 네트워크, 하드웨어 추상화, AT45DB041B, 플래시 메모리

Abstract In this paper, we propose a hardware abstraction architecture(HAA) for low cost flash memories that can be applicable to wireless sensor nodes. The proposed HAA consists of three layers. The three layers are 1) HIL(Hardware Interface Layer), HAL(Hardware Adaption Layer), and HPL(Hardware Presentation Layer), where HIL provides a platform independent interface to applications of upper layers, HAL performs hardware resource management, program status control, and generation of logical instructions as main core of the HAA, and HPL initializes hardware and communicates data between MCU and flash memory. We implemented our HAA on AT45DB flash memory, and the HAA used 4,384 bytes program memory and 195 bytes data memory respectively. Since the proposed HAA is composed of well defined three layers and shows a low utilization of memory, it can provides a high efficiency in terms of flexibility, scalability, and re-usability, and thus the HAA is well suited for wireless sensor nodes.

Key Words : Wireless Sensor Network, Hardware Abstraction Layer, AT45DB, Flash Memory

1. 서 론

무선 센서 네트워크(Wireless Sensor Network: WSN) 환경에서 센서노드는 제한된 전력과 무선 통신, 그리고 센싱 능력을 가지는 여러 개의 작은 디바이스들로 구성된다[1]. 이러한 센서 네트워크

† 이 논문은 2008년도 정부재원(교육인적자원부 학술연구조성사업비)으로 한국학술진흥재단의 지원을 받아 연구되었음(KRF-2008-331-D00453).

* 대구대학교 컴퓨터·IT공학부

기술은 주변의 빛, 소리, 온도, 습도 등의 물리적 데이터의 수집을 바탕으로 환경감시, 재난감시, 군사, 홈 네트워크 의료 등의 다양한 분야에 적용되고 있다.

다양한 응용 분야에 사용되는 센서 노드들의 가장 큰 제약은 소모되는 전력이며 센싱한 데이터를 전송할 때 가장 큰 전력을 소모한다. 따라서 측정된 데이터를 계속해서 전송하는 것보다 저장하는 것이 에너지 소모의 측면과 데이터의 안전성을 위해 보다 효율적이다. 따라서, 데이터의 수집 분야가 다양해지고 장기간 수집을 목적으로 사용되는 경우가 늘어나면서 대량의 데이터를 저장하고 관리하기 위한 효율적인 저장장치가 요구되고 있다.

현재 대부분의 센서노드를 위한 MCU는 매우 작은 크기의 메모리(RAM : 4 ~ 10KByte)를 지원하고, 응용의 크기가 on-chip 메모리의 수용량을 넘어설 수 있기 때문에 추가적인 메모리를 요구한다. 따라서 센서 노드들은 on-chip메모리의 적은 수용량을 해결하기위해 추가적인 예비 저장장치로 외부 플래시 메모리(512KB ~ 1MB)를 가지고 있다[3]. 플래시 메모리는 최근에 주목받는 반도체 기반의 데이터 저장매체로서 비휘발성(non-volatile)이고 하드디스크와 같은 보조기억 장치보다 실행속도가 훨씬 빠르다는 장점이 있다. 이러한 외부 플래시 메모리는 데이터로그, 마이크로 파일 시스템, over-the-air re-programming와 같은 다양한 응용을 지원한다.

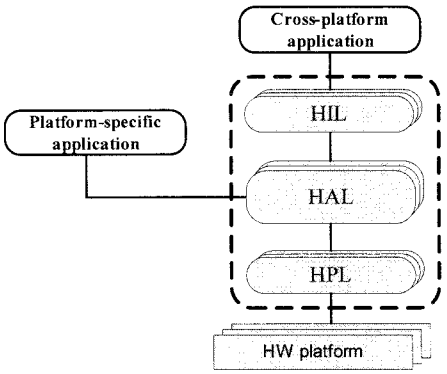
플래시 메모리는 NAND와 NOR로 두 가지의 종류가 있다[2, 3]. 현존하는 센서 노드용 저장시스템의 대부분이 NOR 플래시 메모리용으로 개발되어 있으며 그 중 가장 많이 사용되는 AT45DB NOR 플래시 메모리를 이용해 HAA를 설계한다. AT45DB 디바이스를 위한 HAA의 설계는 하드웨어의 능력을 최대한 발휘 하기위해 확장성과 효율성을 고려해서 설계되어야 한다[5]. 따라서 본 논문에서는 컴포넌트 모델의 TinyOS 2.x의 구조를 참고하여 이식성과 효율성을 고려한 AT45DB 플래시 메모리 디바이스 드라이버의 메모리 스택을 구성한다.

2. 관련연구

현재까지 디바이스 제어를 위한 효율적인 소프트웨어 구조에 대해 많은 연구가 이루어져 왔다. 일반적으로 HAA는 최하위 계층에서 하드웨어 의존적인 기능을 수행하고 상위 계층은 독립성과 이식성을 고려하여 하드웨어 구조의 세부사항(메모리 관리 장치, 타이머, 인터럽트, I/O 장치 등)을 숨김으로서 디바이스의 접근을 용이하게 한다[5, 6].

2.1 TinyOS 2.x의 HAA

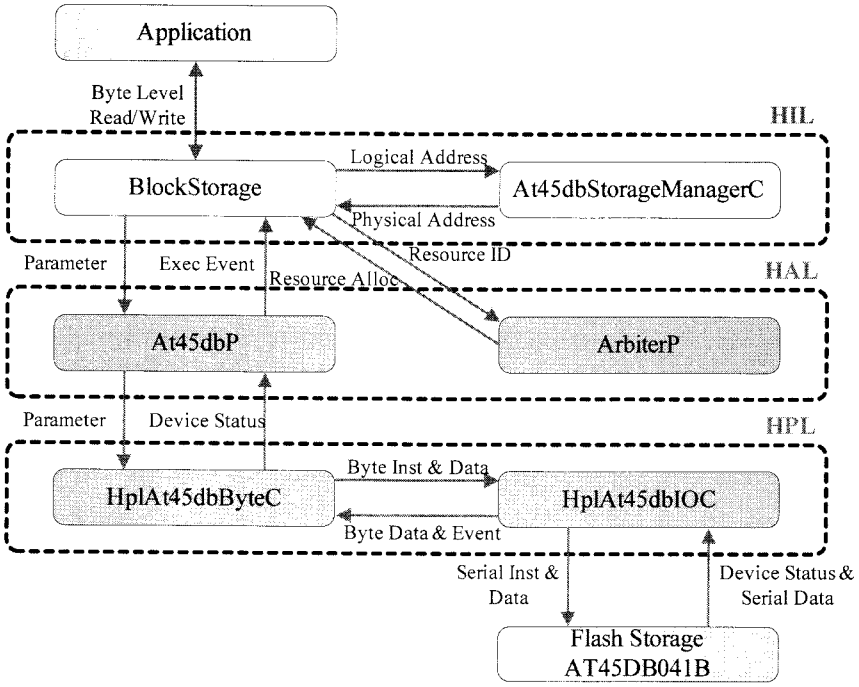
TinyOS 2.x의 HAA는 그림 1과 같이 세 개의 계층으로 구성되어 있으며, 각 계층별 주요 컴포넌트는 그림 2와 같다. 각 계층은 하위 계층에 의해 제공되는 인터페이스에 의존적이다. 분리된 각 계층은 명확하게 정의된 기능을 가지며 계층의 특성에 따라 정의된 기능을 수행한다. TinyOS 2.x는 하드웨어 추상화 구조를 위해 수직 및 수평으로 분리했다[7, 10]. 수직으로 계층을 분리함으로써 메모리 스택과 같이 최상위 계층은 이식성을 고려하려 하드웨어 독립적으로 설계 되었으며, 최하위 계층은 하드웨어에 의존적인 기능을 구현하였다.



(그림 1) TinyOS 2.x의 HAA

2.1.1 HIL

TinyOS 2.x에서 HAA의 최상위 계층은 HAL에 의해 제공되는 플랫폼에 의존적인 추상화를 가지는 HIL 컴포넌트로 구성된다. 또한, 추상화를 크로스-플랫폼 어플리케이션을 통해 하드웨어 독립적인 인터페이스로 변환하는 역할을 수행한다. 이러



(그림 2) TinyOS 2.x의 AT45DB HAA

한 인터페이스는 상위 어플리케이션 계층에 하드웨어의 차이점을 숨김으로써 어플리케이션 소프트웨어의 개발을 간단하게 한다.

2.1.2 HAL

HAL 컴포넌트들은 HAA의 핵심 요소이다. HAL은 하드웨어 자원의 사용과 관련된 복잡성을 숨김으로써 유용한 추상화를 만드는 HPL 컴포넌트들에 의해 제공된 가공되지 않은 인터페이스를 사용한다. HAL 컴포넌트들은 하드웨어 자원을 제어하고 중재하기 위해 사용되는 상태를 유지한다.

2.1.3 HPL

HPL에 속한 컴포넌트들은 하드웨어와 소프트웨어 인터페이스에 직접적으로 연결된다. HPL 컴포넌트들은 메모리 혹은 포트(Port)에 맵핑된 I/O를 통해 디바이스에 접근하며, 반대로 하드웨어는 인터럽트를 이용하여 서비스를 요청한다. 따라서 내부적으로 이러한 통신 채널을 이용함으로서 HPL은 하드웨어의 복잡성을 숨길 수 있다.

2.2 TinyOS 2.x의 AT45DB HAA

AT45DB 디바이스의 메모리 스택은 세 개의 계층을 가진다. HIL에서는 Application에 대한 일관된 접근 인터페이스와 논리주소를 페이지단위의 물리적인 주소로 변경해준다. HAL에서는 메모리 연산 결정 및 논리적인 명령어를 생성한다. 마지막으로 HPL에서는 디바이스 상태 제어 및 물리적인 명령어 구성 및 실행을 담당하며 AT45DB 디바이스와 통신을 한다[8]. 그림 2는 TinyOS 2.x의 AT45DB HAA 컴포넌트 구조를 나타내며 HIL, HAL, HPL에서 수행되는 컴포넌트의 기능은 다음과 같다.

• BlockStorageP

Application에 대해 논리적 메모리 명령어를 제공(Erage, Write, Sync, Read)하며, Byte단위의 데이터 접근 인터페이스를 제공한다.

• At45dbStorageMangeC

Volume 단위의 접근 인터페이스를 제공하며

Application의 논리적인 주소로부터 디바이스의 절대적인 주소 및 offset를 생성한다.

• At45dbP

상위계층의 페이지주소, offset으로부터 메모리 연산의 타당성 체크 및 플래시 메모리의 상태(사용가능 여부)를 체크한다. 그리고 상위 계층의 파라메타로부터 메모리 연산을 결정한다.

• ArbiterP

사용자 ID에 대한 유일한 resource ID를 할당하며 자원이 Free일때 사용자에게 자원을 할당한다. 즉 AT45DB 디바이스의 자원을 관리한다.

• HplAt45dbByteC

At45DB의 논리적인 명령어 및 주소로부터 물리적인 명령어 및 주소를 생성하고, HplAt45dbIOC 컴포넌트에 바이트 단위의 명령어 및 주소를 전송한다. 그리고 AT45DB 디바이스 상태 레지스터 값을 At45dbP 컴포넌트에 전달한다.

• HplAt45dbIOC

데이터 전송 및 수신을 위한 CS핀의 상태 결정, 핀의 방향 설정 및 초기화, 인터럽트를 제어 하고 AT45DB 디바이스와 시리얼 통신을 한다.

3. 효율적인 AT45DB HAA 설계

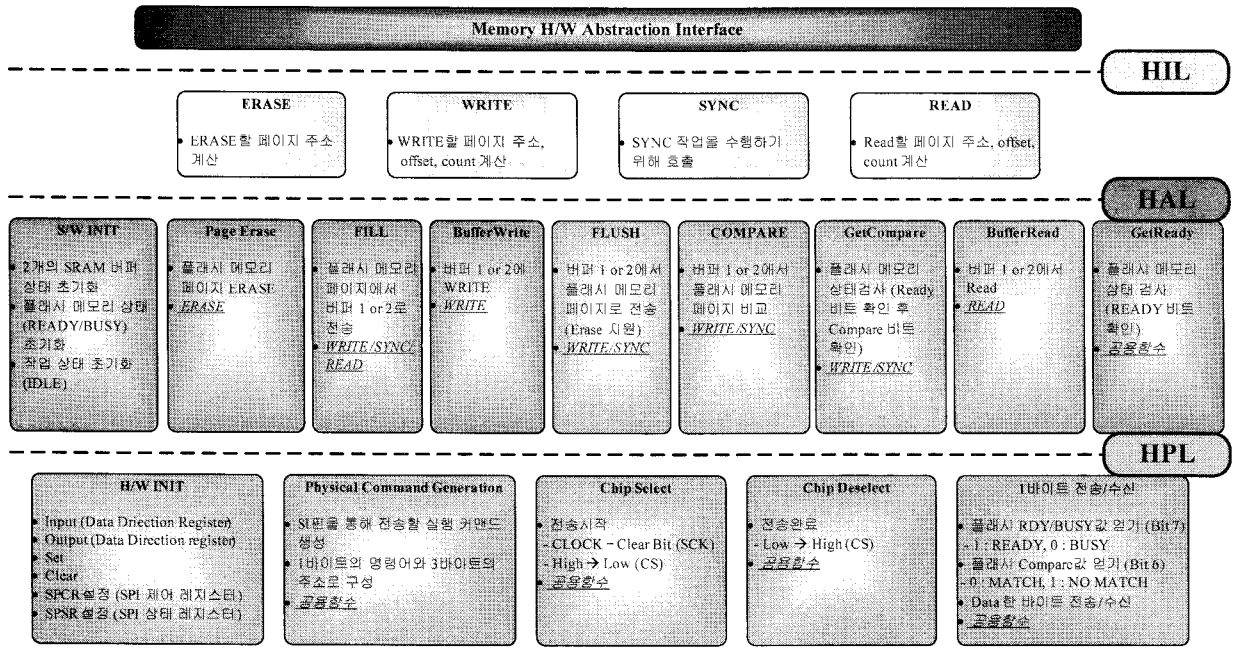
3.1 AT45DB 디바이스

AT45DB는 SPI에 적합한 NOR 플래시 메모리이며 digital voice, image program code, data -storage, over-the-air reprogramming 응용에 다양하게 사용된다. 표 1은 센서노드 SN100보드에 사용되는 AT45DB 디바이스의 특징을 나타낸다.

<표 1> AT45DB의 특성[9]

속성	값
총 사이즈	4,325,376 Bit
페이지 사이즈	264-byte
페이지의 총 수	2048
SRAM데이터 버퍼 수	2

표 2는 AT45DB의 핀 구성을 나타내며 본 논문에서는 HAA의 구현을 위해 SN100 센서노드[11]



(그림 3) AT45DB HAA 구조

를 사용한다.

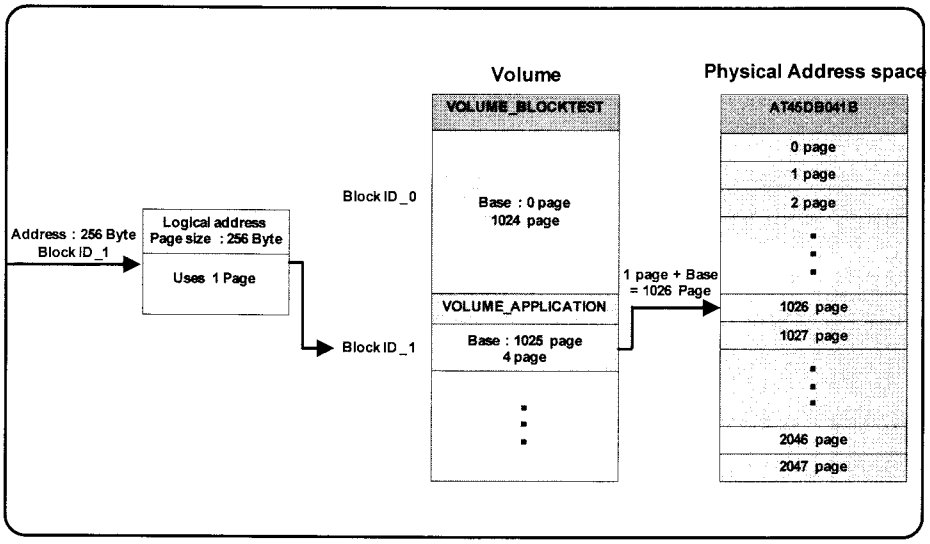
<표 2> AT45DB의 핀 구성

핀 이름	SN100
CS (Chip Select)	*
SCK (Serial Clock)	*
SI (Serial Input)	*
SO (Serial Output)	*
WP (Write Protect Pin)	
RESET (Chip Reset)	*
RDYBUSY (Ready/Busy)	

모리 적용이 가능하며, 다양한 센서노드용 운영체제에 이식이 가능하도록 설계 되었다.

3.2.1 AT45DB의 HIL

HIL은 하드웨어에 독립적이고 응용에 관한 일관된 접근 인터페이스를 제공한다. 바이트 단위인 주소를 인자로 받으며 파라 메타의 유효성 검사 및 논리적인 주소를 물리적인 페이지 주소로 계산한다. HAA의 최상위 계층을 나타내며 ERASE/WRITE/SYNC/READ가 호출 되었을때 block ID



(그림 4) Physical address mapping

3.2 AT45DB HAA 설계

본 절에서는 AT45DB 플래시 메모리를 위한 HAA를 설계한다. 본 논문에서 제안한 플래시 디바이스 드라이버의 HAA역시 세 개의 계층으로 구성되어 있지만 그림 3에 나타나듯이 HAA와 Memory H/W Abstraction Interface를 분리 함으로서 효율적인 메모리 사용이 가능하다. Memory H/W Abstraction Interface는 다수의 메모리 중 특정 메모리에 접근하기 위한 기능을 제공하며, 선택된 메모리 디바이스의 추상화 계층으로, 해당 메모리 디바이스에 접근하기 위한 서비스를 제공한다. 따라서 제안된 AT45DB의 HAA는 다양한 메

를 포함한다. block ID는 블록에 접근한 클라이언트의 상태를 저장하고 공유된 자원을 중재하며 칩의 물리적인 절대 주소를 계산하는데 사용된다.

예를 들어, 어플리케이션에서 HIL의 WRITE함수를 호출 했을 때 입력 파라메타가 논리적인 주소 300byte, 데이터 버퍼 포인터, 데이터 길이 350, block ID 1, 한 페이지의 크기를 256-byte로 가정하자.

- 논리적 페이지 주소 : 1page (300 = (1 × 256) + 44)
- offset : 44 (300 = (1 × 256) +44)
- count : 212 (256 - offset)
- 물리적 페이지 주소 : 1026page (논리적 페이지 주소 + Base)

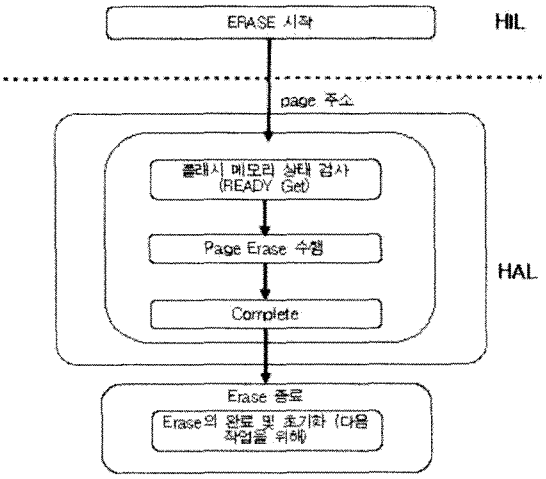
따라서 HIL에서는 위 예의 경우, 그림 4와 같이 1026page에 접근 하게 된다.

3.2.2 AT45DB의 HAL

HAL은 2개의 SRAM 데이터 버퍼와 플래시 메모리 상태에 따라 논리 적인 메모리 명령을 수행한다. 그림 3에 HAL계층에서 수행하는 다양한 기능을 나타내었으며, Page Erase, FILL, BufferWrite, FLUSH, COMPARE, BufferRead, Flash Status Read(GetReady)로 구성된다. 또한 2개의 SRAM 버퍼는 캐시의 역할을 수행하며 빠른 데이터 접근을 가능하게 한다.

■ HAL Erase 연산

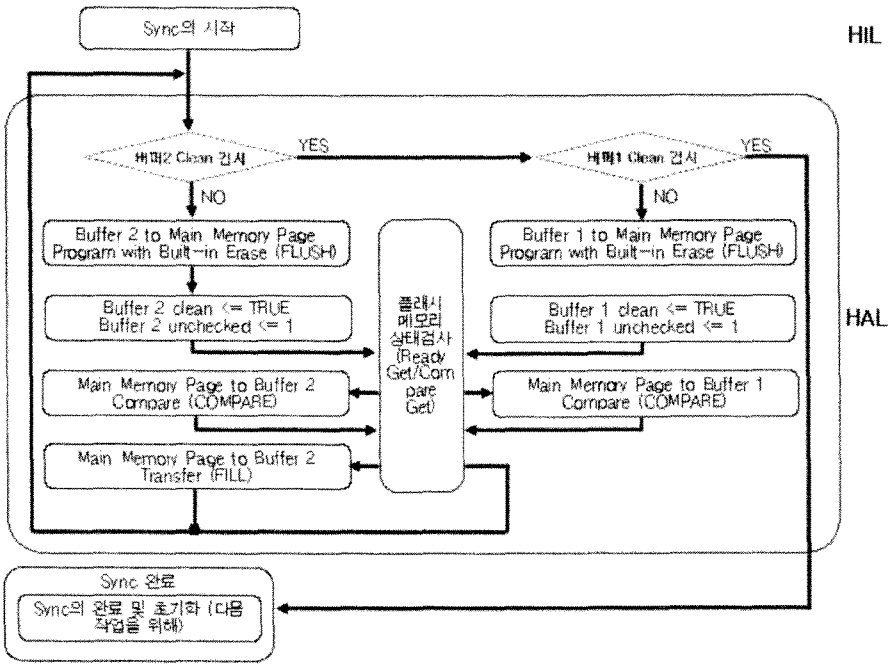
Erase연산은 그림 5와 같이 Flash Status Read와 Page Erase로 구성되며 플래시 메모리의 상태 레지스터가 READY이면 해당하는 AT45DB 플래시 메모리페이지의 데이터를 삭제하는 기능을 수행한다.



(그림 5) Erase Operation

■ HAL Sync 연산

Sync연산은 Flash Status Read, FLUSH, COMPARE로 구성되며, 마지막 쓰기 연산 후에 호출 되어야 한다. 이는 2개의 SRAM 버퍼에 마지막으로 써진 데이터가 전원이 OFF됨에 따라 삭제 될 수 있기 때문에 반드시 수행 되어야 한다. Sync연산은 채워진 버퍼를 플래시 메모리 페이지로 쓰기 작업을 수행하며 COMPARE연산은 SRAM 버퍼와 플래시 메모리에 써진 데이터의 유효성을



(그림 6) Sync Operation

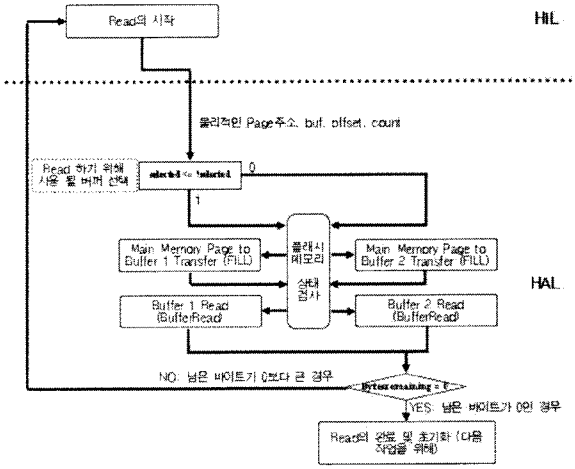
검사하기 위해 사용된다. 마지막으로 2개의 SRAM 버퍼가 모두 비어 있는 상태이면 Sync연산을 완료한다.

■HAL Write 연산

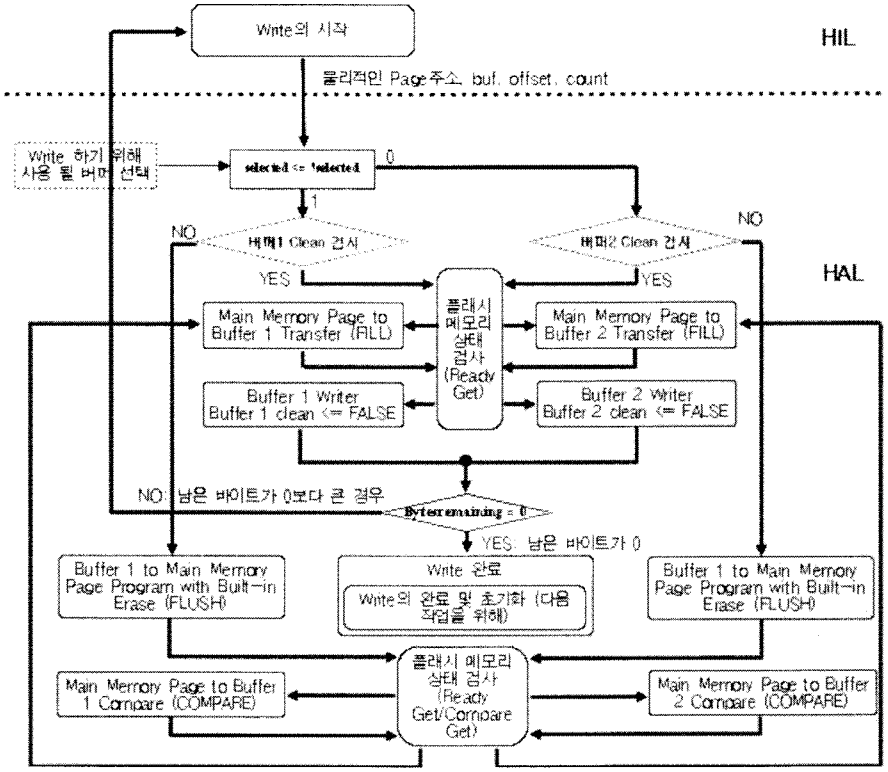
Write연산은 그림 7과 같이 Flash Status Read, FILL, BufferWrite로 구성되며 2개의 SRAM 데이터 버퍼 중 하나를 선택하고 FILL 연산을 수행한다. FILL 연산을 수행하는 이유는 플래시 메모리의 한 페이지에 저장된 다수의 데이터들이 수정될 수도 있기 때문이다. (플래시 메모리 페이지에 데이터 Write시 페이지 Erase를 먼저 수행) FILL 연산 종료 후 BufferWrite 연산을 수행(버퍼가 채워진 상태가 됨)하며 Write연산을 종료한다. 또한 멀티페이지 write를 지원해 다수의 페이지의 연속적인 쓰기도 가능하다. 만약, 채워진 버퍼에 쓰기 작업이 일어난 경우 FLUSH와 COMPARE 연산을 하고, 계속해서 BufferWrite 연산을 수행한다.

■HAL Read 연산

Read연산은 Flash Status Read, FILL, BufferRead로 구성되며 2개의 SRAM 데이터 버퍼중 하나를 선택하고 FILL 연산을 수행해서 해당하는 플래시 메모리 페이지의 데이터를 버퍼에 채우고 BufferRead 연산을 수행 후 Read 연산을 완료한다. 또한 멀티페이지 read를 지원해 다수 페이지에 대한 연속적인 Read 연산도 가능하다.



(그림 8) Read Operation

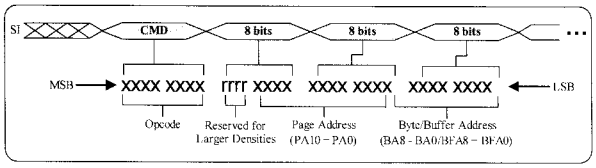


(그림 7) Write Operation

3.2.3 AT45DB의 HPL

HPL은 그림 3에서 최하위 계층을 나타내며 핀 방향설정과 같은 초기화와 연산 명령어를 실행하며 MCU와 SPI 통신을 수행한다. 그림 9와 같이 Read/Write 연산은 1-byte의 opcode와 3-byte의 주소로 구성되어 있다.

따라서 HPL은 Page Erase, FILL, BufferWrite, FLUSH, COMPARE, BufferRead, Flash Status Read를 위한 명령어를 구성하며 SPI 모드 0과 3을 지원한다.



(그림 9) Read/Write 연산을 위한 명령어 시퀀스

4. 결 론

본 논문에서는 무선 센서 노드에 사용 가능한 저가형 플래시 메모리를 위한 하드웨어 추상화 구조를 설계하였다. 디바이스 드라이버의 하드웨어 추상화는 하드웨어의 복잡성을 감추는 것에 의해 응용프로그램 개발을 간단히 하고 이식성을 좋게 한다. 그러나 잘 설계 되지 않은 하드웨어 추상화는 센서 네트워크 응용의 요구 사항인 재 사용성과 에너지 효율성면에서 서로 상충된다. 따라서 제안된 AT45DB 플래시 메모리 디바이스 드라이버는 HAA와 Memory H/W Abstraction Interface를 분리함으로써 효율적인 메모리 사용이 가능하며, 높은 재 사용성을 제공한다. 제안된 HAA는 무선 센서노드에 가장 많이 사용되고 있는 Atmel사의 AT45DB 계열의 플래시 메모리에 적용되었으며, 유비쿼터스 신기술 연구센터의 SN100 센서노드 [11]에 구현한 결과 4,384 바이트의 프로그램 메모리와 195 바이트의 데이터 메모리를 사용하였다. 따라서 본 논문에서 제안된 HAA 구조는 3계층으로 설계되었기 때문에 소프트웨어 개발 측면에서 높은 유연성, 확장성, 재사용성을 제공하며, 낮은 메모리를 사용하기 때문에 무선 센서 노드용으로 적합하다 할 수 있다.

참 고 문 헌

[1] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," IEEE Communications Magazine, pp. 102-114, Aug. 2002.

[2] Memory Technology Device Subsystem for Linux,http://www.linux-mtd.infradead.org/doc/nand.html

[3] 송준영, 이기혁, 한형진, 최원철, 한경훈, 한지연, 손기락, "센서 노드를 위한 플래시 메모리 저장 시스템에 대한 고찰", 한국 정보과학회 가을 학술 발표논문집, vol. 34, No. 2, 2007.

[4] Andreas Lachenmann, Pedro José Marrón, Matthias Gauger, Daniel Minder, Olga Saukh, and Kurt Rothemel, "Removing the Memory Limitation of Sensor Networks with Flash-Based Virtual Memory," ACM SIGOPS Operating Systems Review, Proceedings of the 2007 conference on EuroSys EuroSys '07, Vol. 41, pp. 131-144, March 2007.

[5] Vlado handziski, Joseph Polastre, Jan-Hinrich Hauer, Cory Sharp, Adam Wolisz, and David culler, "Flexible Hardware Abstraction for Wireless Sensor Networks," Proceeding of the Second European Workshop on Wireless Sensor Network, pp. 145-157, Feb. 2005.

[6] Sungjoo Yoo and Ahemd A. Jerraya, "Introduction to hardware abstraction layers for SoC," Embedded Software for SOC, pp. 179-186, 2003.

[7] TEP 002: Hardware Abstraction Architecture, Sept. 2004. http://www.tinyos.net.

[8] TEP 103: Permanent Data Storage (Flash). http://www.tinyos.net.

[9] Atmel Corporation 4-megabit DataFlash AT45DB-041B Datasheet, 2005.

[10] P.Levis, TinyOS Programming, Oct.2006
[11] UTRC, 유비쿼터스 신기술 연구센터. http://www.utrc.re.kr

김 창 훈 (Chang Hoon Kim)

- 2001년 2월 : 대구대학교 컴퓨터정보공학부 (공학사)
- 2003년 2월 : 대구대학교 컴퓨터정보공학과 (공학석사)
- 2006년 8월 : 대구대학교 컴퓨터정보공학과 (공학박사)
- 2006년 9월 : 대구대학교 정보통신공학부, BK21 연구교수
- 2007년 8월 ~ 현재 : 대구대학교 컴퓨터·IT공학부, 전임강사
- 관심분야 : 암호시스템, Embedded System, RFID/USN 보안

권 영 직 (Young Jik Kwon)

- 종신회원
- 1976년 2월 : 경북대학교 수학과 (이학사)
- 1980년 2월 : 영남대학교 경영학과 (경영학석사)
- 1991년 2월 : 계명대학교 경영학과 (경영학박사)
- 1980년 3월 ~ 현재 : 대구대학교 컴퓨터·IT공학부 교수
- 2000년 1월 ~ 2001년 1월 : Washington State University 방문교수
- 관심분야 : 소프트웨어공학, 전자상거래

논문접수일 : 2009년 5월 29일
논문수정일 : 2009년 6월 16일
게재확정일 : 2009년 6월 25일