GDS30C6001기반 플래쉬 메모리 및 카드 시뮬레이터 구현

정회원 김 동 현*, 신 필 순*, 곽 윤 식**, 김 백 기***

Implementation of Flash Memory and Card Simulator using GDS30C6001

Dong-Hyun Kim*, Phil-Soon Shin*, Yoon-Sik Kwak**, Baek-Ki Kim*** ***Regular Members***

요 약

본 논문은 GDS30C6001이라는 USB콘트롤러을 기반으로 하는 플래쉬 메모리 및 카드 시뮬레이터에 관한 것이다. 시스템 구현에 사용하는 GDS30C6001은 MMC, SSFDC카드 인터페이스 콘트롤러로 최고속도 12Mbit/s, USB Ver 1.1 사양/윈도우 95, 98 그리고 MacOs 8.1에서 구동되며 플래쉬 메모리로는 8M, 16M 그리고 32M SMC에 대해서 자동 읽기기능, 메모리 블록 읽기/쓰기 기능, 메모리 덤프 기능, 파일 ｐｒｏｐｅｒｔｙ기능을 구현하였으며 이를 이용하여 시스템설계 및 디버깅과정을 효과적으로 수행할 수 있었다.

ABSTRACT

In this paper, we present a simulator of flash memory and its card which using USB controller named GDS30C6001. GDS30C6001 for system implementation is the interface controller of MMC and SSFCD card which operated with limit speed 12Mbit/s in USB Ver.1.1 and under circumstances windows 95, 98 and Mac Os 8.1. Using this controller, we implemented functional design such as automatic reading, memory block read/write, memory dump and file property of 8M/16M flash memory and 32M SMC. And with this results, we could effectively process the system design and debugging.

Ⅰ. 서 론

컴퓨터 기술의 발전과 더불어 컴퓨터 기능 및 디바이스에 대한 수요자의 욕구가 증대됨에 따라 컴퓨터 응용 기술은 획기적으로 발전되고 있다.

USB(Universal Serial Bus)는 PC와 외부 디바이스를 연결시켜 주는 외부버스로 제안되어 기존의 여러 문제점을 해결할 수 있다. 즉, 직렬 또는 병렬포트와 같은 저속포트는 그 응용에 많은 제약이 있다. 둘째, 기존의 컴퓨터 또는 사용가능한 포트수의 제약성이 있다. 셋째, 외부 디바이스의 관리가 복잡하다. 넷째, 전원관리 문제가 발생할 수 있다는 문제점이 있으며 이와같은 문제점을 해결할 수 있는 방안으로 제시되고 있는 것이 USB이다.[1][2]

USB의 특징은 외부 디바이스의 연결에 융통성이 있으며, 디바이스 수의 제약을 해결할 수 있고, 자체 전원을 사용할 수 있다는 장점이 있다.

본 시스템은 USB를 기반으로 화상 또는 음성 레코드 및 응용 시스템의 설계과정 및 디버깅 과정에 요구되는 플래쉬 메모리 및 플래쉬 메모리 카드에 대한 I/O포트 테스트(GPIO), SSFDC에 대한 읽기/쓰기 기능의 File Save Test기능, SSFDC Check카드, 모드선택, 블록 읽기 / 쓰기, 메모리 덤프가 가능한 플래쉬 메모리 제어기능을 주기능으로 설정

* 경희대학교 대학원 전자공학과 ** 충주대학교 컴퓨터공학과
*** 원주대학 전자통신과
논문번호 : T01015-0530, 접수일자 : 2001년 5월 30일

구현하였다.

Ⅱ. 본 론

Ⅱ-1. GDS30C6001 USB 컨트롤러

GDS30C6001은 MMC(Multimedia Card) 그리고 SSFDC(Solide State Floppy Disk Card)에 대한 USB 콘트롤러이다. 이는 USB 호스트 PC 그리고 메모리 카드(SSFDC/MMC)의 인터페이스 콘트롤러로 USB 버전 1.1 사양에 기반을 하고, 메모리 카드의 엑세스 위한 신호를 생성시키는 어뎁터이다. 외부의 부가적인 전원 지원 없이 USB버스의 전원을 공급한다. 호스트 PC상에서 동작되는 USB 디바이스 드라이버는 - MP3 또는 디지털 카메라등에 저장되어 있는 영상 또는 음성 데이터 등 - 메모리 카드의 제어가 가능하다.

양방향의 CMD, DAT핀이 지원된 MMC 인터페이스가 가능하여 MMC모드 엑세스를 위한 PLD가 요구되지 않는다. 또한 32Mbyte 이상의 SSFDC가 지원된다.

이 같은 콘트롤러의 블록 다이어그램은 그림 1과 같으며 세부적인 특성은 다음과 같다.

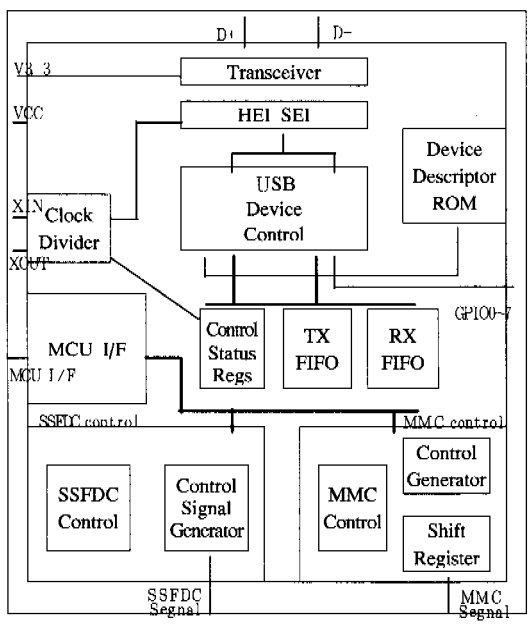


그림 1. USB콘트롤러에 대한 블럭도

① MMC/SSFDC USB 콘트롤러
② 직렬 데이터 통신으로 최고속도 12Mbit/s지원
③ plug and play 지원
④ 3.3V 전원동작으로 외부전원이 불필요
⑤ USB 버전 1.1 지원
⑥ On-Chip 트랜서버
⑦ USB 디스크립터(Descriptor)에 대한 프로그램이 가능
⑧ 4개의 endpoint(양방향의 Endpoint 0이 1개, Bulk endpoint가 2개, Status endpoint 1개)
⑨ Windows 95, 98 MacOs 8.1 지원
⑩ 외부 MCU (Micro Controller Unit)을 위한 3핀의 직렬 인터페이스 기능을 갖고 있다.

이 같은 기능은 외부 카드리더, 디지털 카메라, PDA, MP3 플레이어, 그리고 화상·음성레코더 등에 활용된다.

Ⅱ-2. 플래쉬 메모리 및 카드 구조

본 시스템에서 사용한 메모리 및 카드는 8, 16M의 K9F6408U0A, K9F2808U0M, 32M SMC의 K9D5608V0M으로 그림 2와 같은 메모리 구조를 가지고 있다.

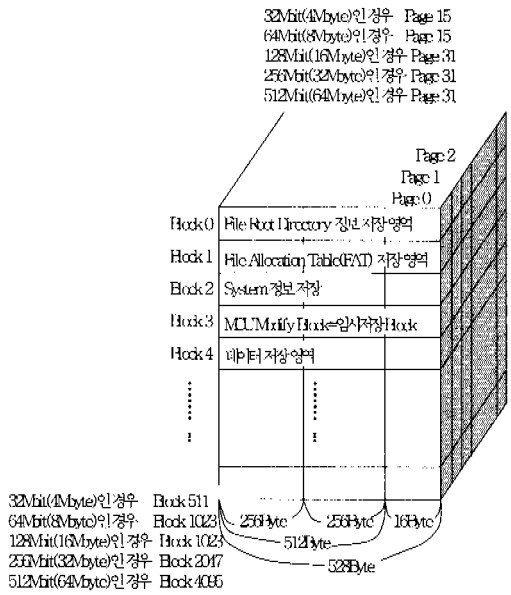


그림 2. 메모리 구조

Ⅲ-3. 구현된 시스템 구조

구현된 시스템에서는 플래쉬 메모리내 데이터를 PC상으로 Upload/Download가 가능하고 기존 USB 테스트 프로그램을 upgrade시키는 방향으로 설계되었으며, 그 파일구조는 그림 3과 같다. 파일 시스템

구조는 3레벨로 구성되며, 저 레벨의 H/W부분에서는 GDS3DC6001등과 관련된 인터페이스 부분이며, 커널 레벨의 Usbfmc.sys파일과 유저레벨의 Cwusbfmc.dll, P2MMC.dll파일로 구성되어 있다.

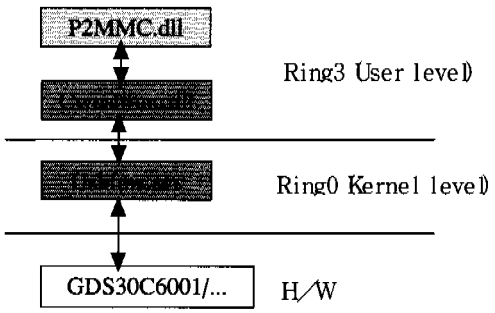


그림 3. 파일 시스템 구조

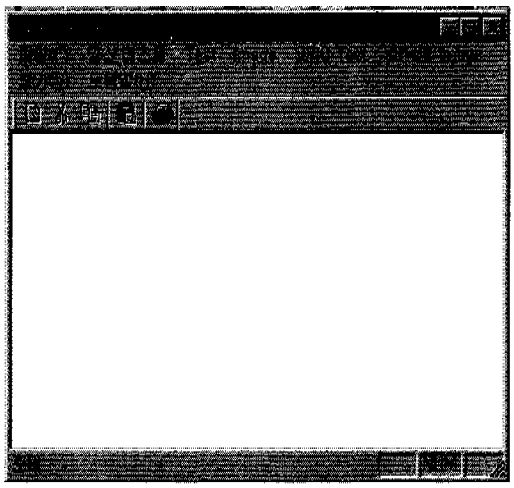
① 플래쉬 메모리에 대한 자동 읽기 기능

기존 시스템에 대해서 플래쉬 메모리 내용의 자동 읽기 기능으로 메모리 구조에 대한 내용을 확인할 수 있다. 즉, 제시된 그림 4(b)와 같이 메모리의 블록단위 데이터를 확인할 수 있다. 이를 위한 알고리즘은 다음과 같다.

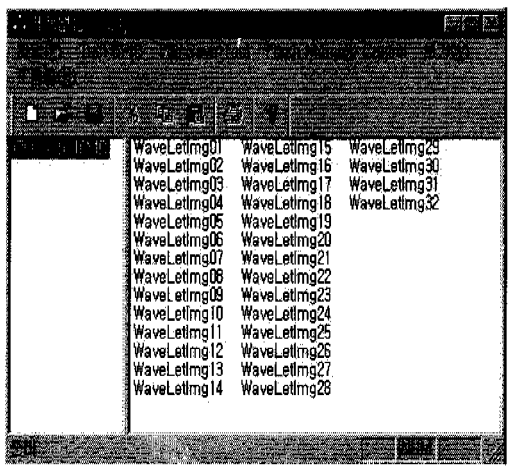
```
BYTE* FAT::ssfdc_READ_1st()
{
BYTE Buff[512+16];
DWORD stAddress, Size, Block;

stAddress = 0x00;
Size = 512;
Block = 1;

if(!SSFDCReadMultiBlock(Buff,        stAddress,
Size+32, Block)) {
        AfxMessageBox("Read error!.",
            MB_OK | MB_ICONHAND);
          return NULL;
          }
          return Buff;
}
```

② 플래쉬 메모리 블록 읽기/쓰기 기능

전자에서 설명한 기능①에서 확인된 블록에 대해서 임의의 자료에 대한 읽기/쓰기 기능을 제공함으로써 플래쉬 메모리상에 데이터를 시뮬레이션 할

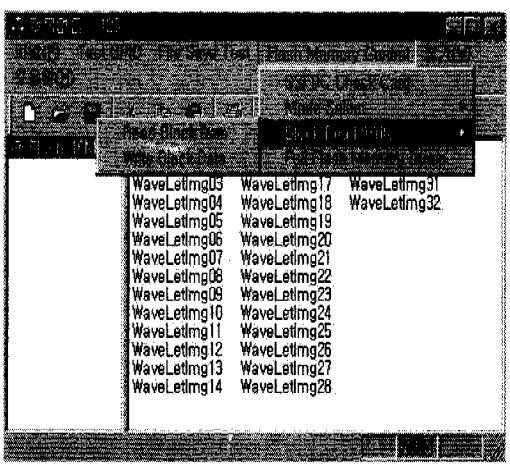


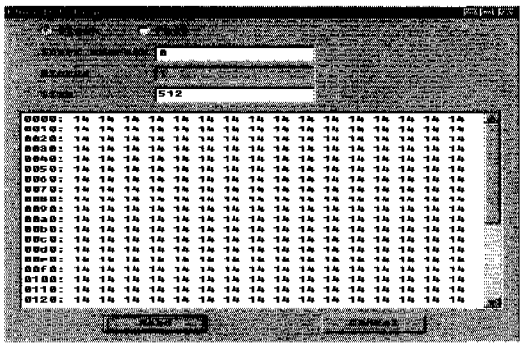
(a) 기존 시스템



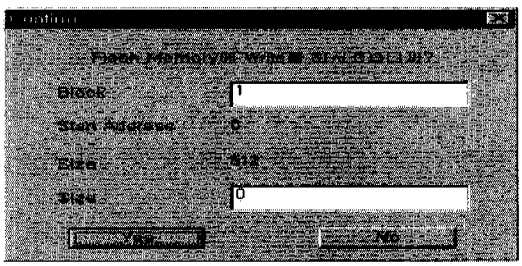
(b) 구현 시스템

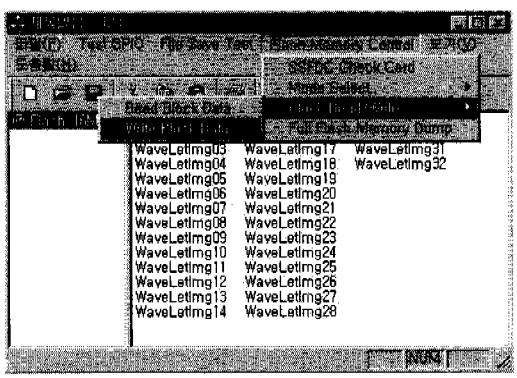
그림 4. 자동 메모리 읽기 기능





(a) 읽기기능





(b) 쓰기기능

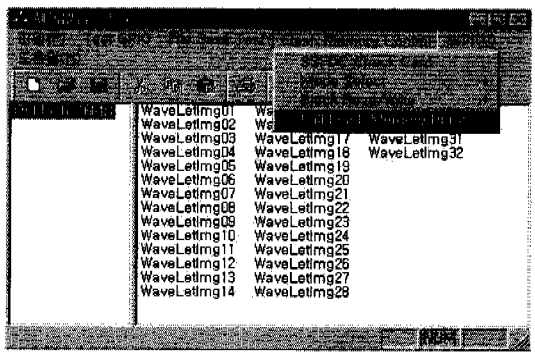
그림 5. 블록 단위의 데이터 읽기/쓰기 기능

수 있도록 하였다. 즉, 블록단위의 저급레벨의 소스 데이터를 확인하여 메모리 디버깅 작업이 가능하도록 하였다. 이를 위한 알고리즘은 다음과 같다.

```
void CReadBlockData::OnOK() {
char str[100];
char temp[10];
BYTE Buff2[512+16];
UpdateData(TRUE);

if (!IOOpenPort(NULL, NULL)) {
    AfxMessageBox("Cannot open port.",
                MB_OK | MB_ICONHAND);
            return;
}
// Samsung Flash Memory Select
DEVICEWriteGPIO(3, 0);
if( !SSFDCCheckCard() ) {
            DEVICEWriteGPIO(4, 0);
            if( !SSFDCCheckCard() )

        AfxMessageBox("Not Found.(SSFDC
                or Flash Memory).", MB_OK |
                            MB_ICONHAND);
            }
BYTE Device = SSFDCGetDeviceCode();
switch (Device) {
            case SSFDCDEVICE_4MB:
            wsprintf(str, "%02x", Device);
                            break;
                            .
                            .
            default:
            AfxMessageBox("Card verification
                    error, at OnSsfdcRead #2",
                MB_OK | MB_ICONHAND);
            IOClose();
            return;
            }
if( ((CButton*)GetDlgItem
            (IDC_BLOCK))->GetCheck() ) {
    if(!SSFDCReadMultiBlock(Buff2,
        DWORD(m_stAddress),
        DWORD(m_Size+32),
        DWORD(m_Block))) {
            AfxMessageBox("Read error!.",
                MB_OK | MB_ICONHAND);
            return;
                }
            }
if( ((CButton*)GetDlgItem
            (IDC_PAGE))->GetCheck() ) {
    AfxMessageBox("Donot Support!",
            MB_OK | MB_ICONHAND);
            return;
            }
m_BlockData = "";
```

```
temp[3] = 0;
for(DWORD i = 0 ; i < 512 ; i+=16){
        sprintf(temp, "%04x: ", i);
        m_BlockData += temp;
        for(DWORD j=0; j<16; j++){
        sprintf(temp,"%02x",Buff2[i+j]);
          m_BlockData += temp;
          }
        m_BlockData += "\r\n";
        }
UpdateData(FALSE);
IOClose();
}
```

③ 메모리 dump 기능



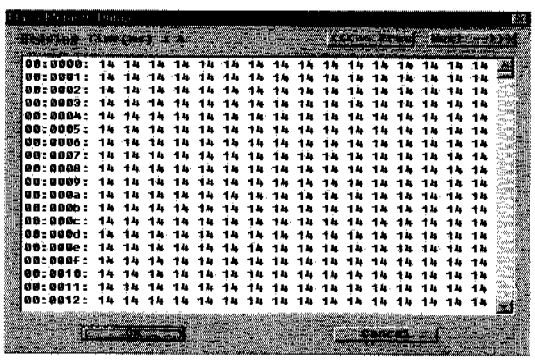


그림 6. 메모리 Dump 기능

메모리 Dump기능을 제공하여 메모리내의 데이터를 조사하는데 블록단위의 Dump기능을 활용할 수 있도록 하여 사용자로 하여금 쉽게 사용할 수 있는 부가적 기능을 추가하였다. 이를 위한 알고리즘은 다음과 같다.

```
void CMemoryDump::OnOK() {
char *temp = new char [10];
BYTE *Buff2 = new BYTE [512+16];
time_t start, finish;
start = clock();
UpdateData(TRUE);
DWORD FlashSize;
        if (!IOOpenPort(NULL, NULL)) {
        AfxMessageBox("Cannot open
        port.", MB_OK | MB_ICONHAND);
          return ;
        }
// Samsung Flash Memory Select
DEVICEWriteGPIO(3, 0);
if( !SSFDCCheckCard() ) {
        DEVICEWriteGPIO(4, 0);
        if( !SSFDCCheckCard() )
        AfxMessageBox("Not Found.
         (SSFDC or Flash      Memory).",
           MB_OK | MB_ICONHAND);
        }
BYTE Device = SSFDCGetDeviceCode();
switch (Device) {
        case SSFDCDEVICE_4MB:
                FlashSize = 4194304;
                        break;
                .
                .
default:
        AfxMessageBox("Card verification
      error,  at  OnSsfdcRead  #2",  MB_OK
                | MB_ICONHAND);
        IOClose();
        return ;
}
temp[3] = 0;
DWORD MemoryAddress = 0;
m_MemoryDump = "";
for(DWORD i = StartMemAddress;
    i < EndMemAddress /*32768/
    2 FlashSize/16*/ ; i+=512){

        if(!SSFDCReadMultiBlock(Buff2,
           DWORD(i), DWORD(512+32),
           DWORD(1))) {
          AfxMessageBox("Read error!.",
            MB_OK | MB_ICONHAND);
```

```
                    return;
            }
            for(DWORD k = 0 ; k < 512 ;
                k+=16) {
                sprintf(temp, "%2x:", BlockNum);
                m_MemoryDump += temp;
        sprintf(temp,"%4x:",MemoryAddress);
                MemoryAddress++;
                m_MemoryDump += temp;
                for(DWORD j=0;j <16;j++){
                  sprintf(temp,"%2x",Buff2[k+j]);
                  m_MemoryDump += temp;
                  }
                m_MemoryDump += "\r\n";
            }
}
finish = clock();
long duration = (finish - start) /
CLOCKS_PER_SEC;
m_duration = duration;
IOClose();
delete [] temp;
delete [] Buff2;
UpdateData(FALSE);
}
```

④ 파일 property 기능

파일 property 기능을 이용하여 메모리에 저장되어 있는 파일의 property를 확인할 수 있으며, 데이터의 속성에 따른 인터페이스 동작 시험이 가능하도록 하였다.

이를 위한 알고리즘은 다음과 같다.

```
voidCMainFrame::OnSsfdccheckcard()
{
    if (!IOOpenPort(usbDevice,
                    m_fEPPMode)) {
    AfxMessageBox("Cannot open
     port.", MB_OK | MB_ICONHAND);
     return;
     }
BOOL SSFDCCHECKCARD =
                    SSFDCCheckCard();
// Flash Memory
```

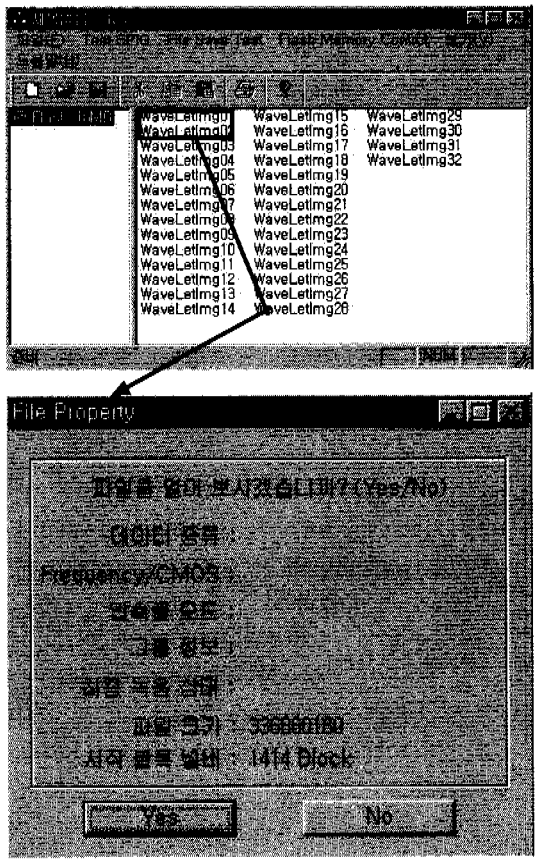


그림 7. 파일 property 기능

```
DEVICEWriteGPIO(3, 0);
if( !SSFDCCheckCard() ) {
            DEVICEWriteGPIO(4, 0);
            if( !SSFDCCheckCard() ) {
              AfxMessageBox("Not
              Found.(SSFDC or Flash
              Memory).", MB_OK |
              MB_ICONHAND);
                  return;
              }
}
BYTE DeviceCode;
BYTE MakerCode;
DeviceCode=
            SSFDCGetDeviceCode();
MakerCode =
            SSFDCGetMakerCode();
char str[100];
wsprintf(str, "Get Device Code =
```

```
 %02X \n\nGet Maker Code =
 %02X", DeviceCode, MakerCode);
AfxMessageBox(str, MB_OK);
IOClose();
}
```

Ⅲ. 시스템 구현 환경

본 시스템은 윈도우 환경에서 사용이 가능하도록 시스템을 설계하기 위해서 OS는 윈도 98, 그리고 펜티엄 Ⅱ PC를 사용 하였으며 구현 언어는 C++ 을 사용하였다. 사용된 USB 콘트롤러는 현대전자에서 생산되고 있는 GDS30C6001 USB 콘트롤러를 기반으로 플레쉬메모리는 삼성전자에서 생산되는 K9F6408U0A, K9F2808U0M 8, 16Mbyte 와 32Mbyte SMC인 K9D5608V0M 로 실험을 수행하였다.

Ⅳ. 결 론

본 논문에서는 GDS30C6001이라는 USB콘트롤러을 기반으로 하는 플래쉬 메모리 및 카드를 테스트하는 시뮬레이터에 관한 것이다.

기존의 시스템의 경우는 메모리에 대한 기능이 부족하고 부가적으로 제공되는 기능이 하드웨어 시스템의 설계 및 디버깅 과정에 비효율적인 관계로 이를 해결할수 있었다. 시스템 구현에 사용하는 GDS30C6001은 MMC, SSFDC카드 인터페이스 콘트롤러로 최고속도 12Mbit/s, USB Ver 1.1 사양/윈도우 95, 98 그리고 MacOs 8.1에서 구동되며 플래쉬 메모리로는 8M, 16M 그리고 32M SMC에 대해서 자동 읽기기능, 메모리 블록 읽기/쓰기 기능, 메모리 덤프 기능, 파일 property기능을 구현되었다.

참 고 문 헌

[1] CHRIS CANT, Writing Windows WDM Device Drivers, R&D Books.

[2] DeKKer Newcomer,Developing Windows NT Device Drivers, Addison Wesley

[3] Universal Serial Bus Specification Revision 1.1

[4] Universal Serial Bus SSFDC/MMC I/F Controller Data SHeet(GDS30C6001), HYUNDAI Electronics industries.co.LTD, 2000, Mar

김 동 현(Dong-Hyun Kim) 정회원

1996년 2월 : 경희대학교 전자공학과(공학석사)

1999년 2월 : 경희대학교 전자공학과 박사과정수료

신 필 순(Phil-Soon Shin) 정회원

1999년 2월 : 경희대학교 전자공학과(공학석사)

2001년~현재 : 경희대학교 전자공학과 박사과정

곽 윤 식(Yoon-Sik Kwak) 정회원

한국통신학회논문지 제 25권 8B호 참조

김 백 기(Baek-Ki Kim) 정회원

한국통신학회논문지 제 25권 12T호 참조