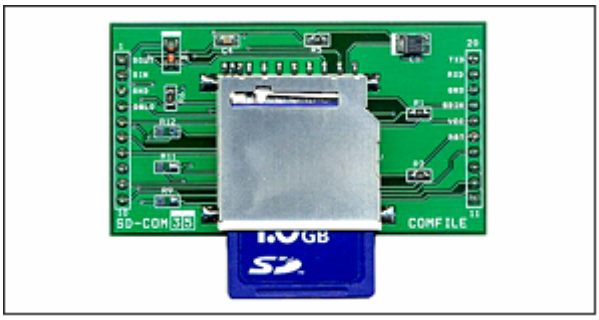
CUBLOC Peripheral

Serial SD/MMC

Serial SD/MMC Card Module User Manual



1. Intro

The Serial SD/MMC Card Module allows the user to read SD/MMC cards using TTL level serial communication. You will be able to interface with CUBLOC, CuTOUCH, or any other control devices that supports TTL serial.

2. Features

- Simple commands through TTL serial communication for creating files, writing and reading data.
- Industrial Standard FAT16 supported (FAT32 is not supported)
- Up to 2Gigabytes of SD/MMC Card supported.
- Read/Write Text and Binary data
- Korean Language supported
- Terminal Mode for using with MCUs or PC's Hyperterminal
- Automatic Card detection
- Status pins for Card detection
- 2 Wire (RX and TX) for 5V or 3V RS232C
- No parity, 8bit Data, 1 stop bit
- Set baud rates using commands
- Baud rates supported (bps) : 4800, 9600, 19200, 38400, 57600, 115200 (Default: 9600bps)
- Current Consumption
  - SD-COM5 (5V): 30mA(Idle)
  - SD-COM3 (3V): 9mA(Idle)
  - Read/Write Operation (Add ~ 30mA)
- Firmware Upgrade through Internet

3. Specifications

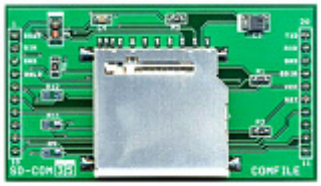
Model	SD-COM5	SD-COM3
Voltage	4.5~5.5V	2.7~5.5V
Read Speed	- 115200 bps: 10KB/s - 9600 bps: 8KB/s	- 115200 bps: 8KB/s - 9600 bps: 2KB/s
Write Speed	- 115200 bps: 8KB/s - 9600 bps: 0.4KB/s	- 115200 bps: 2KB/s - 9600 bps: 0.8kB/s

* Warning

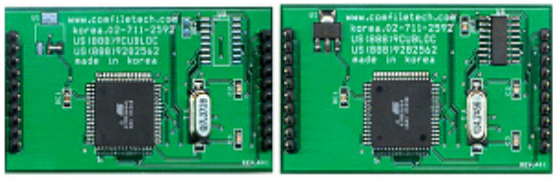
- The SD/MMC card comes in FAT16 as factory default. Please do not re-format the SD/MMC card in FAT32.
- Please format the SD/MMC card as FAT16 for SD/MMC cards in FAT32 before using.
- Please do not eject SD/MMC card during read/write operations. This can cause loss/error to your files.
- Depending on the SD/MMC Card model, read/write speed may differ slightly.
- SD/MMC card is slightly different from the CF card because 5V cannot be used directly to it. The SD-COM5 contains a 3.3V regulator and a logic-level converter.

4. Dimensions

■ Front



■ Back



SD-COM3 SD-COM5

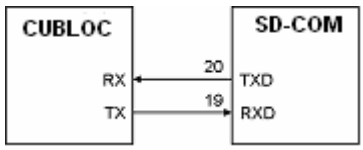
■ Pin Specification

Pin	Name	Description
1	SOUT	Serial Output pin for Firmware Upgrade
2	SIN	Serial Input pin for Firmware Upgrade
3, 18	GND	GROUND
4	DNLD	Firmware Upgrade Status Pin - During Firmware Upgrade: LOW - Normal: HIGH
15	RST	RESET (Pull up with 10K resistor)
16	VCC	Power (5V: 4.5~5.5V, 3V: 2.7~5.5V).
17	SDIN	SD/MMC Card Detection - SD/MMC Inserted: LOW - SD/MMC Not Inserted: HIGH
19	RXD	TTL Input pin for 5V(SD/MMC-COM5) or 3V(SD/MMC-COM3) level
20	TXD	TTL Output pin for 5V(SD/MMC-COM5) or 3V(SD/MMC-COM3) level
5~10	reserved	Reserved for future I/O ports

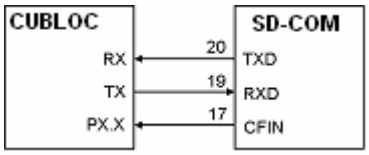
***Do not connect TXD/RXD directly to PC's serial port as PC uses 12V RS232 levels. You need to use a MAX232 chip to convert 5V TTL to 12V RS232C level before doing so.**

5. How to connect data lines

■ Processor Connection (CUBLOC, etc)

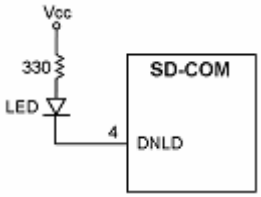


■ SDIN Pin Connection



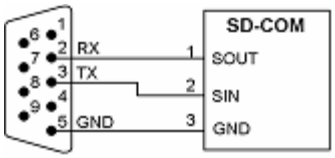
You can check the status of SD/MMC card insertion by connecting SDIN pin to one of your processor's input pins.

■ How to connect SD/MMCIN pin to an LED

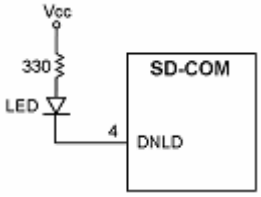


You will be able to see the LED light up when the SD/MMC card inserted.

■ How to connect data lines for Firmware Upgrade



■ How to connect Firmware Upgrade Status Pin to LED



6. Communication Protocol

■ Basics

Command [Filename] [Option] Data] CR LF

Command, Filename, Option, and data are separated by a space (HEX 0x20). Depending on the command, Filename, Option, and Data can be required. All Commands must be followed by a CR (Carriage Return, 0x0D) and LF (Line Feed, 0x0A).

For example, to store "Hello World" into text.txt, you would do:

In C:

printf("fputs test.txt /w Hello World \r\n");

In CUBLOC:

Putstr 1, "fputs test.txt /w Hellow World",cr,lf

■ Return Values

The SD/MMC-COM will reply as follows:

<Message>

- Normal

When command is processed successfully, the SD/MMC-COM will return a capitalized 'O' or in hex, 0x4F.

- Error

When command is not processed successfully, an error will occur and the SD/MMC-COM will return a capitalized 'E' or in hex, 0x45.

<Data>

When reading a file, the SD/MMC-COM will return the data after the message

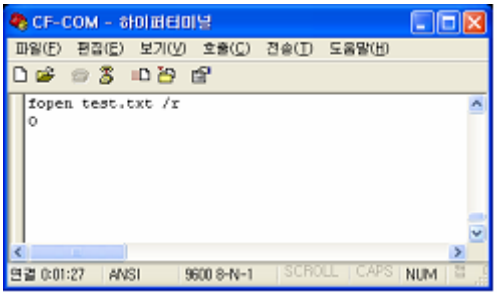
■ MCU Mode and Terminal Mode

<MCU Mode> is used when you are connecting SD/MMC-COM to another control device such as CUBLOC.

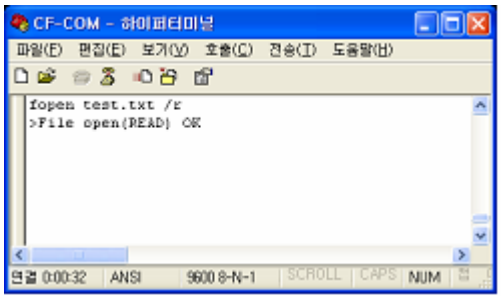
The factory default is set to MCU Mode.

<Terminal Mode> is used when you are connecting SD/MMC-COM to PC's Hyperterminal or similar program for testing and debugging. The returned messages are in a more detailed format.

<MCU Mode return Message in Hyperterminal>



<Terminal Mode return Message>



■ Commands For Terminal Mode

mode

mode [Option] CR LF

- Operation: Select MCU Mode or Terminal Mode.
- Option

/t Terminal Mode

/m MCU Mode

- Example

```
mode /t
>Mode: terminal(full message)
mode /m
>Mode: MCU(simple message)init
```

init

init CR LF

- Operation: Initialize the SD/MMC Card.
- Example:

```
init
>Initialize OK
```

cd

cd [Change Directory] CR LF

- Operation: Change directory. Directory name must be within 40 characters.
- Example 1: Change directory to mydir\sub1

```
cd mydir\sub1
>Change directory OK: mydir1\sub1
```

- Example 2: Change directory to root directory

```
cd \
>Change directory OK: \
```

dir

dir CR LF

- Operation: List Directory
- Return Value: Filenames are returned with size of file inside (). Directory names are return with brackets [].
- Example

```
dir
ROOT\\
SINE.DAT (210)
TEST.TXT (7618)
[MYDIR1]
[MYDIR2]
```

fsize

fsize [Filename] CR LF

- Operation: Display File size.
- Return Value: File Size
- Example (Terminal Mode)

```
fsize test.txt
  >File Size: 7618 bytes
```

- Example (MCU Mode)

```
fsize test.txt
  7618
```

dsize

dsize CR LF

- Operation: Display Total Disk Space of the SD/MMC Card.
- Return Value: Total Disk Space of SD/MMC Card
- Example (Terminal Mode)

```
dsize
  >Total size: 128032768 bytes (125 MB)
```

- Example (MCU Mode)

```
dsize
  128032768
```

ftime

ftime [Filename] CR LF

- Operation: Display File creation and Last-Modified times.
- Return Value: File creation and Last-modified times.
- Example(Terminal Mode)

```
fsize test.txt
  >File created:   08/01/2005   15:37:13
   File modified: 07/21/2005   11:10:08
```

- Example(MCU Mode)

```
  ftime test.txt
  08/01/2005   15:37:13
07/21/2005   11:10:08
```

* This product does not have a real time clock, therefore when doing a file write, the Creation and Last-Modified times are not recorded. Only files created or modified in the PC will show Creation and Last-Modified times.

md

md [Directory] CR LF

- Operation: Make Directory.
- Example

```
md mydir1
>Make directory OK
md mydir1\sub1
>Make directory OK
```

rd

rd [Directory] CR LF

- Operation: Remove Directory.
- Example

```
rd mydir1
>Remove directory OK
rd mydir1\sub1
>Remove directory OK
```

* Please remove the files within the directory before removing. This command only will remove empty directories.

del

del [Filename] CR LF

- Operation: Delete File.
- Example

```
del test.txt
>Delete file OK
del mydir1\test.txt
>Delete file OK
```

fcreate

fcreate [Filename] CR LF

- Operation: Create a new file with size 0.
- Example:

```
fcreate test.txt
>File create OK
fcreate mydir1\test.txt
>File create OK
```

* fcreate command will create a file with size 0. Therefore all read/write commands with options Open File (/r) or Append File (/a) cannot be used.

rename

rename [Source Filename] [Destination Filename] CR LF

- Operation: File의 이름을 바꿉니다.
- Example: test.txt File을 test2.dat 라는 이름으로 바꾸기

```
rename test.txt test2.dat
  >Rename OK
```

fopen

fopen [Filename] [/Option] CR LF

- Operation: Open File.
- Option
  - **/r** File Read
  - **/w** File Write
  - **/a** File Append
- Example: File Read

```
fopen test.txt /r
  >File open(READ) OK
```

- Example: File Overwrite

```
fopen test.txt /w
  >File open(WRITE) OK
```

- Example: File Append

```
fopen test.txt /a
  >File open(APPEND) OK
```

* After opening a File, you must use File Close (fclose) command to close the File.
* Only 1 File may be opened at one time.
* /r and /a Option cannot be used with Files of size 0.

fclose

fclose CR LF

- Operation: Close File.

- Example

```
fopen test.txt /r
  >File open(OPEN) OK
  fclose
  >File close OK
```

fputc

fputc [Filename] [/Option] [1 Byte Data] CR LF

- Operation: Write 1 byte to the File.

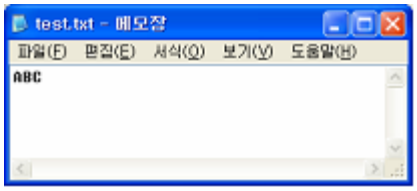
- Option

/w File Write)

/a File Append)

- Example

```
fputc test.txt /w A
>Put character OK
fputc test.txt /a B
>Put character OK
fputc test.txt /a C
>Put character OK
```



fputs

fputs [Filename] [/Option] [String] CR LF

- Operation: Write up to 256 cha\

racters of String data to the File.

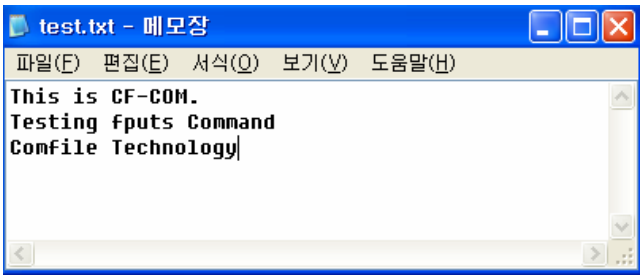
- Option

/w File Write)

/a File Append)

- Example

```
fputs test.txt /w This is SD/MMC-COM.
>Put string OK
fputs test.txt /a Testing fputs Command
>Put string OK
fputs test.txt /a Comfile Technology
>Put string OK
```



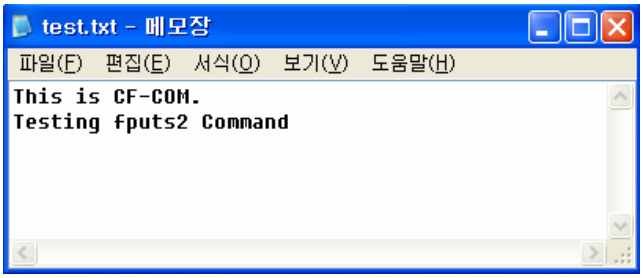
fputs2

fputs2 CR LF

- Operation: Write until ^Z (Ctrl + Z, ASCII Code 0x1A) is received. Up to 256 bytes or characters may be written.

- Example

```
fopen test.txt /w
>File open(WRITE) OK
fputs2
>File put string ready.
This is SD/MMC-COM.
>Put string OK (continue or ^Z)
Testing fputs2 Command
Ctrl + Z >fputs2 end
```



* fopen command must be used before using fputs2 command.

* <Ctrl + Z> will automatically close the file and fclose command does not have to be called.

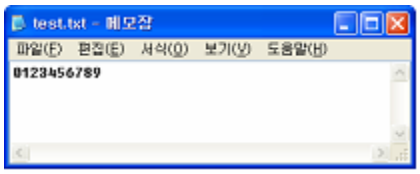
fwrite

fwrite [/ # of bytes to write] CR LF

- Operation: Write up to set # of bytes (Up to 512 bytes per Operation).

- Example

```
fopen test.txt /w
>File open(WRITE) OK
fwrite /4
>Packet size: 4 bytes
0123>Packet 4 bytes write OK
fwrite /6
>Packet size: 6 bytes
456789>Packet 6 bytes write OK
fclose
>File close OK
```



* fopen command must be used before using fwrite command.

fgetc

fgetc [/# of bytes to read] CR LF

- Operation: Read up to set # of bytes (Up to 256 bytes per Operation).

- Example

```
fputs test.txt /w 0123456789
>Put string OK
fopen test.txt /r
>File open(READ) OK
fgetc /4
0123
fgetc /6
456789
fclose
>File close OK
```

* fopen command must be used before using fgetc command.

fgets

fgets CR LF

- Operation: Read 1 line of string (CR LF = new line).

- Example

```
fputs test.txt /w This is SD/MMC-COM.
>Put string OK
fputs test.txt /a Testing fgets Command
>Put string OK
fputs test.txt /a Comfile Technology
>Put string OK
fopen test.txt /r
>File open(READ) OK
fgets
This is SD/MMC-COM.
fgets
Testing fgets Command
fgets
Comfile Technology
fclose
>File close OK
```

* fopen command must be used before using fgets command.

fread

fread [Filename] CR LF

- Operation: Read all data in File

- Example

```
fputs test.txt /w This is SD/MMC-COM.
>Put string OK
fputs test.txt /a Testing fread Command.
>Put string OK
fputs test.txt /a Comfile Technology
>Put string OK
fread test.txt
This is SD/MMC-COM.
Testing fread Command.
Comfile Technology
```

reset

reset CR LF

- Operation: SD/MMC-COM을 하드웨어적으로 리셋 시킵니다.
- Example

```
reset
>System reset OK
```

baud

baud [Baud rate] CR LF

- Operation: Set the Baud rate for the serial communications.
- Example

```
baud /9600
>Baudrate: 9,600bps
```

card

card CR LF

- Operation: Return SD/MMC Card Status.
- Example: When SD/MMC Card Inserted

```
card
>SD/MMC card inserted
```

- Example: When SD/MMC Card Removed

```
card
>Error: SD/MMC card NOT inserted
```

* When SD/MMC Card is Removed and Re-inserted, the following message will appear.

<Terminal Mode>

```
card
>SD/MMC card inserting...
>SD/MMC card detected
```

<MCU Mode>

```
card
ID
```

MCU Mode returns 'O' (0x4F) for successful operations except when SD/MMC Card is Inserted, where a 'I' and a 'D' is returned.

help

help CR LF

- Operation: Show Help Menu.

■ Error Message

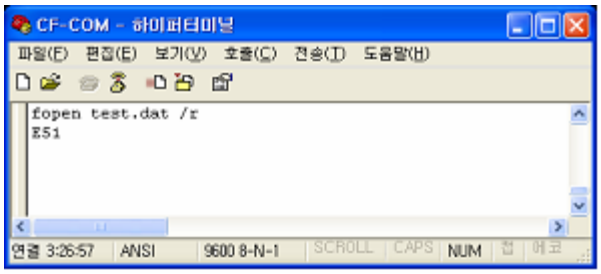
MCU Mode Error Codes.

Error Code	Explanation
E00	Command not recognized.
E10	Card not inserted.
E11	Card inserted but not initialized. (Please remove and re-insert the card)
E12	Error during Card Initialization.

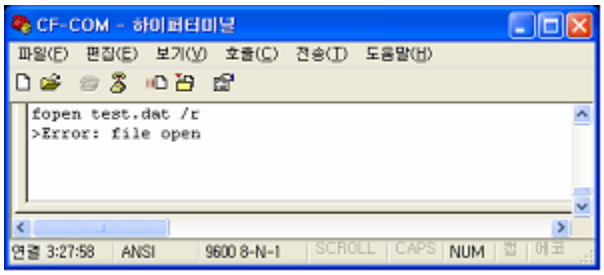
Error Code	Explanation
E20	File creation Error during fcreate.
E21	File Delete Error
E22	File Rename Error
E30	md command Error
E31	rd command Error
E32	cd command Error
E40	fsize command Error
E41	ftime command Error
E50	fopen command Error
E51	fopen command Error
E52	fopen command Error
E53	fopen command Error
E54	File open already
E55	fclose command Error
E56	Command not available during fopen
E57	fopen must be used beforehand
E58	During fopen command, only options /r, /w/ /a may be used
E60	fgetc option not between 1 and 256 bytes
E61	Fgetc command my only use option /r
E70	fputc command may only use options /w or /a
E71	Data Write error during fputc
E72	fputs command, only options /w or /a may be used

E73	fputs command Error
E74	fwrite option must be between 1 and 512 bytes
E80	baud rate setting Error

<MCU Mode Error Message >

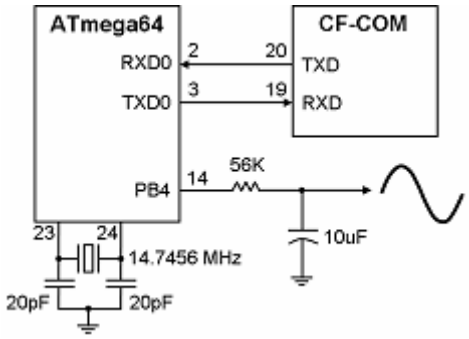


<Terminal Mode Error Message >



7. Example 1: Read Sine Frequency File from the SD/MMC card and output as analog signal.

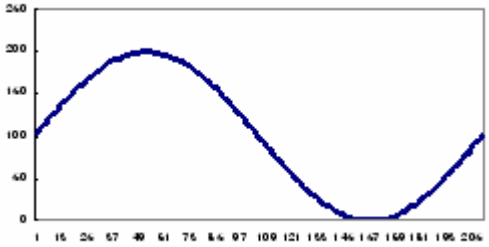
■ Circuit Schematics



Please connect the SD/MMC-COM and the processor and set the processor's PWM to output and create an RC filter.

■ How to

Using your PC, store sine.dat File in the SD/MMC card. This File is a binary File that stores a sine wave such as shown below.



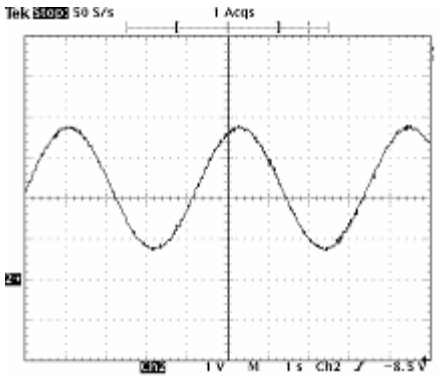
< sine.dat File – 210 bytes of data >

```
00000000h: 67 6A 6D 70 73 76 79 7C 7F 82 84 87 8A 8D 8F 92
00000010h: 95 97 9A 9C 9F A1 A4 A6 A8 AA AC AE B0 B2 B4 B6
00000020h: B8 B9 BB BC BE BF C0 C1 C2 C3 C4 C5 C6 C6 C7 C7
00000030h: C7 C8 C8 C8 C8 C8 C8 C7 C7 C7 C6 C5 C5 C4 C3 C2
00000040h: C1 C0 BF BD BC BA B9 B7 B5 B4 B2 B0 AE AC AA A8
00000050h: A5 A3 A1 9E 9C 99 97 94 91 8F 8C 89 86 84 81 7E
00000060h: 7B 78 75 72 6F 6C 69 66 63 60 5D 5A 57 54 51 4E
00000070h: 4B 49 46 43 40 3D 3A 38 35 32 30 2D 2B 28 26 24
00000080h: 21 1F 1D 1B 19 17 15 13 12 10 0E 0D 0B 0A 09 08
00000090h: 06 05 05 04 03 02 02 01 01 00 00 00 00 00 00 00
000000a0h: 01 01 02 02 03 04 04 05 06 07 09 0A 0B 0D 0E 10
000000b0h: 11 13 15 17 19 1B 1D 1F 21 23 26 28 2A 2D 2F 32
000000c0h: 35 37 3A 3D 40 42 45 48 4B 4E 51 54 57 5A 5D 60
000000d0h: 63 66
```

When the source code in the processor is executed, the sine.dat File from SD/MMC card is read and outputted as PWM signal. At this time, the PWM output is set as an RC filter, causing the digital signal to be outputted as analog signal. The RC filter acts as a generic DAC (digital-to-analog converter).

Depending on the data stored in the SD/MMC Card, the user is able to output various waves or even ECG signals, allowing it to act

< Output Wave>

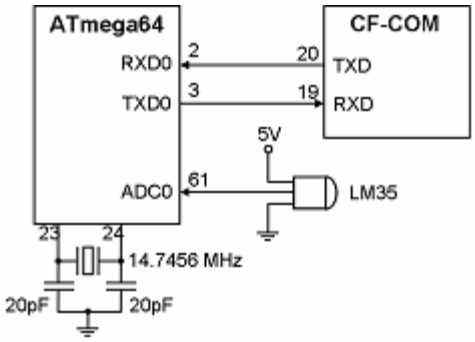


■ Source Code

```
/*****************************************
Project                 :   Example 1
Compiler                : CodeVisionAVR
Chip type               : ATmega64
Clock frequency         : 14.745600 MHz
Data Stack size         : 1024
*****************************************/

#include <mega64.h>
#include <delay.h>
#include <stdio.h>

unsigned char RX_buff[210];
unsigned char fread_end=0;
unsigned int RX_count=0;

void init(void)
{
  WDTCR=0x00;       // Watchdog Timer disable
  #asm("cli")       // global interrupts disable

  UCSR0B=0x00;      // Set UART0 to 9,600bps
  UCSR0A=0x00;      // SD/MMC-COM must be also set to 9,600bps
  UCSR0C=0x06;
  UBRR0H=0x00;
  UBRR0L=0x5F;
  UCSR0B=0x98;

  PORTB=0x00;
  DDRB=0x10;        // Set PB4 as PWM

  // Timer/Counter 0 initialization
  // OC0 output: Non-Inverted PWM
  ASSR=0x00;
  TCCR0=0x67;
  TCNT0=0x00;
  OCR0=0x00;

  #asm("sei")       // global interrupts enable
}

void main(void)
{
  unsigned int i;

  init();

  printf("fread sine.dat\r\n");      // Read File

  while(1) {
    if (fread_end==1) {
      OCR0=RX_buff[i];                    // OutputPWM
      delay_ms(20);
      i++;
      if (i==RX_count)
        i=0;
    }// end if
  }// end while
}

//////////////////////////////////////////
///   UART0 receive interrupt service routine
//////////////////////////////////////////
interrupt [USART0_RXC] void usart0_rx_isr(void)
{
  unsigned char status, data;

  data=UDR0;

  // Store Sine Wave values from SD/MMC-COM in a buffer
  RX_buff[RX_count]=data;
  RX_count++;
  if (RX_count>208)
    fread_end=1;

}
```

8. Example 2: Read Temperature Sensor and store current temperature values in a File of SD/MMC Card.

■ Circuit Schematics



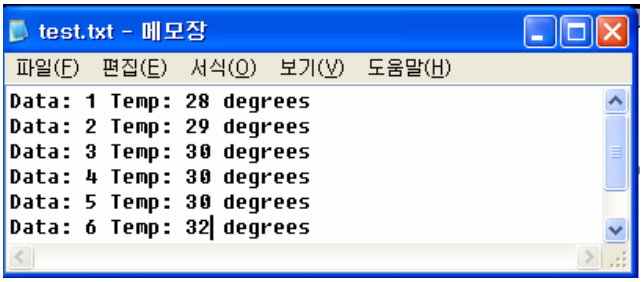
Connect the SD/MMC-COM and the processor. Connect the voltage output for temperature of LM35 to the ADC of the processor. LM35 output 0.01V per 1℃. For example, when LM35 voltage output is 0.26V, this means the current temperature is 26℃.

■ How to

When the source code in the processor is executed, thermo1.txt and thermo1.dat files are created in the SD/MMC card. Every 1 second, the Temperature is converted and stored in thermo1.txt as text and thermo2.dat as binary values.

After a few seconds of execution, you may open thermo1.txt file in PC and verify the written data. You can also verify the binary data using programs such as UltraEdit.

< thermo1.txt >



< Binary File, thermo1.dat >

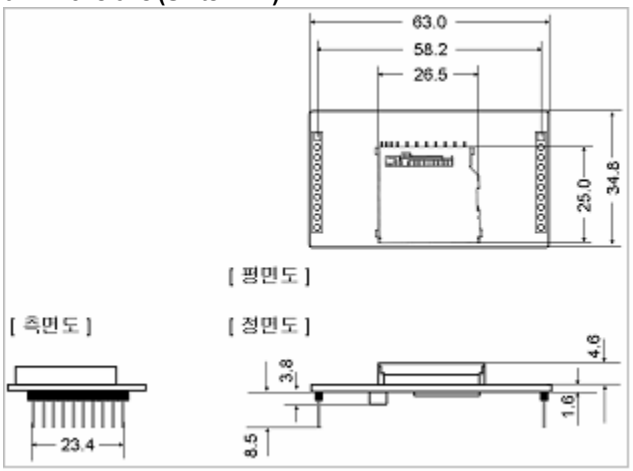
```
00000000h: 3C 54 45 4D 50 3E 0D 0A 1C 1C 1C 1D 1D 1D 1E 1E
00000010h: 1E 1E 1E 1E 1F 1F 1F 1F 1F 1F 1F 1F 1E 1E 1E 1E
00000020h: 1D 1D 1D 1D 1D 1C 1D 1C 1C 1C 1C 1C 1C 1C 1C 1C
```

■ Source Code

```
/*****************************************
Project                 : Example 2
Compiler                : CodeVisionAVR
Chip type               : ATmega64
Clock frequency         : 14.745600 MHz
Data Stack size         : 1024
*****************************************/

#include <mega64.h>
#include <delay.h>
#include <stdio.h>

#define ADC_VREF_TYPE    0xC0    // AVREF=internal 2.56V

char fRX=0;

//////////////////////////////////////////
///       Read the AD conversion result
//////////////////////////////////////////


unsigned int read_adc(unsigned char adc_input)
{
  unsigned long data=0;

  ADMUX=adc_input|ADC_VREF_TYPE;
  ADCSRA|=0x40;                    // Start AD Conversion
  while ((ADCSRA & 0x10)==0);   // Wait until finished
  ADCSRA|=0x10;
  data=ADCW;
  return (data);
}

void init(void)
{
  WDTCR=0x00;        // Disable Watchdog
  #asm("cli")        // global interrupts disable

  UCSR0B=0x00;       // Set UART0 to 9,600bps
  UCSR0A=0x00;       // SD/MMC-COM must also be set to
9,600bps
  UCSR0C=0x06;
  UBRR0H=0x00;
  UBRR0L=0x5F;
  UCSR0B=0x98;

  // ADC initialization
  // ADC Clock frequency: 115.200 kHz
  // ADC Voltage Reference: Int., cap. on AREF
  ADMUX=ADC_VREF_TYPE;
  ADCSRA=0x87;

  #asm("sei")            // global interrupts enable
}
```

```
void wait_message(void)
{
  // Wait reply from SD/MMC-COM
  while(fRX==0);
  fRX=0;
}

void main(void)
{
  unsigned int a=0;
  unsigned int temper=0;

  init();

  // Create File
  printf("fcreate thermo1.txt\r\n");
  wait_message();

  // Create File
  printf("fcreate thermo1.dat\r\n");
  wait_message();

  while(1) {

// After using fcreate, you may use option /a after writing to it using option /w
 printf("fputs thermo1.txt /w <Temperature Example>\r\n");
 wait_message();
 printf("fputs thermo1.dat /w <TEMP>\r\n");
 wait_message();

  while(1) {

    a++;
    temper=read_adc(0)/1024.0*2.56*100;

    // Write to text file using fputs
    printf("fputs thermo1.txt /a Data:%d, Temp: %d 'C\r\n", a, temper);
    wait_message();

    // Write to binary file using fputc
    printf("fputc thermo1.dat /a %c\r\n",(unsigned char)temper);
    wait_message();

    delay_ms(1000);  // Delay 1 second
  }// end while
}

////////////////////////////////////////
///  UART0 receive interrupt service routine
////////////////////////////////////////
interrupt [USART0_RXC] void usart0_rx_isr(void)
{
  unsigned char status, data;

  status=UCSR0A;
  data=UDR0;

  if (data=='O')  // OK reply: ASCII Code (0x4F)
    fRX=1;
}
```

9. Dimensions (Units: mm)



Max's How-to using CUBLOC Study Board right from the package

1) Connect 5V and GND of the SD Card Module to 5V and GND of the study board.
2) Connect TXD and RXD of the SD Card Module to RX and TX pin of CB280(on the upper right corer) on the study board. (For CB280 connect to P10 and P11)
3) Connect your PC's serial cable to the bottom serial port of the study board.
4) Now you are ready to communicate to the SD Card
5) Run the Code: (you can download at www.cubloc.com under Max's Application Download under the title, "**Max's CF/SD Card Library**"