Access FAT on SPI mode SD Card

Example Code Introduction for 32-bit NuMicro® Family

Information

Application	This sample code accesses the FAT file system on the SD card by the SPI interface
BSP Version	M451 Series BSP CMSIS V3.01.003
Hardware	NuTiny-EVB-M451-LQFP100 V1.3

The information described in this document is the exclusive intellectual property of Nuvoton Technology Corporation and shall not be reproduced without permission from Nuvoton.

Nuvoton is providing this document only for reference purposes of NuMicro microcontroller based system design. Nuvoton assumes no responsibility for errors or omissions.

All data and specifications are subject to change without notice.

For additional information or questions, please contact: Nuvoton Technology Corporation.

www.nuvoton.com

1 Function Description

1.1 Introduction

This sample code accesses the FAT file system on the SD card by the SPI interface. It also demonstrates how to read and write data to a file on the SD card.

1.2 Principle

SD card support two operating modes, SD card mode and SPI mode. The SD card SPI mode is compatible with SPI hosts and consists of the following four signals:

Pin #	SD Mode			SPI Mode		
	Name	Type[1]	Description	Name	Type[1]	Description
1	CD/DAT3[2]	I/O/PP[3]	Card Detect/ Data Line [Bit 3]	CS	I[3]	Chip Select (neg true)
2	CMD	PP	Command/Response	DI	I	Data In
3	V_{SS1}	S	Supply voltage ground	VSS	S	Supply voltage ground
4	V_{DD}	S	Supply voltage	VDD	S	Supply voltage
5	CLK	I	Clock	SCLK	I	Clock
6	V_{SS2}	S	Supply voltage ground	VSS2	S	Supply voltage ground
7	DAT0	I/O/PP	Data Line [Bit 0]	DO	O/PP	Data Out
8	DAT1[4]	I/O/PP	Data Line [Bit 1]	RSV		
9	DAT2[5]	I/O/PP	Data Line [Bit 2]	RSV		

This sample code provides a set of source code to support the SPI mode SD card commands and initialization flow.

This sample code also includes a third party simple FAT file system FatFs. It demonstrates how to read and write data to a file on the SD card by call the APIs of FatFs.

1.3 Demo Result

```
PLL Clock = 144000000 Hz
Core Clock = 72000000 Hz

+-----------------------------------------------------+
|       M451 Series SPI + SD Card Sample Code         |
+-----------------------------------------------------+
SDCard Open success
SDCard Open success
rc=0 FR_OK
D-HS- 2019/08/01 15:09         0  SYSTEM~1
----A 1980/00/00 00:00        66  test.txt
D---- 2015/09/09 09:39         0  FF8
----A 2019/09/09 19:32      7388  main.c
   2 File(s),       7454 bytes total
   2 Dir(s),  3594223616 bytes free
Read data from file test.txt :
[hello hello hello hello hello hello hello hello hello hello hello hello ]
```

2 Code Description

Configure SPI interface and initialize SD card:

```
/* Initializes the physical disk drive */
res = (FRESULT)disk_initialize(DRIVE_NUMBER);

if (res != STA_OK)
{
    put_rc(res);
    printf("\n\nInitialize SD card fail.\n");
}
```

Initialize FatFs file system:

```
/* Registers a work area to the FatFs module */
res = f_mount(DRIVE_NUMBER, &g_FatFs[0]);

if (res != FR_OK)
{
    put_rc(res);
    printf("\n\nMount file system fail.\n");
}
```

Read and write data to a file on the SD card by call the APIs of FatFs.

```
/* Open file and set it to readable / writable mode */
res = f_open(&file, "test.txt", FA_OPEN_ALWAYS | FA_READ | FA_WRITE);

if (res == FR_OK)
{
    /* Move file pointer to end of file (offset = file.fsize) */
    f_lseek(&file, file.fsize);
    /* Write something to file pointer */
    f_write(&file, "hello ", 6, &count);
    /* Move file pointer to begin of file (offset = 0) */
    f_lseek(&file, 0);
    /* Read something from file pointer */
    f_read(&file, g_buff, file.fsize, &count);
    printf("Read data from file test.txt : \n[%s]\n", g_buff);
    /* Close file */
```

```
        f_close(&file);
    }
    else
        printf("Open file fail. Return %d\n", res);
```

3 Software and Hardware Environment

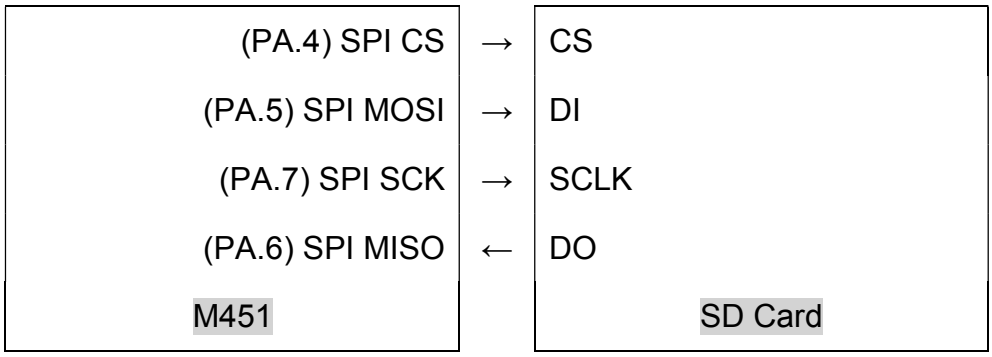
Software Environment

- BSP version
  - M451 Series BSP CMSIS V3.01.003
- IDE version
  - Keil uVersion 4.74

Hardware Environment

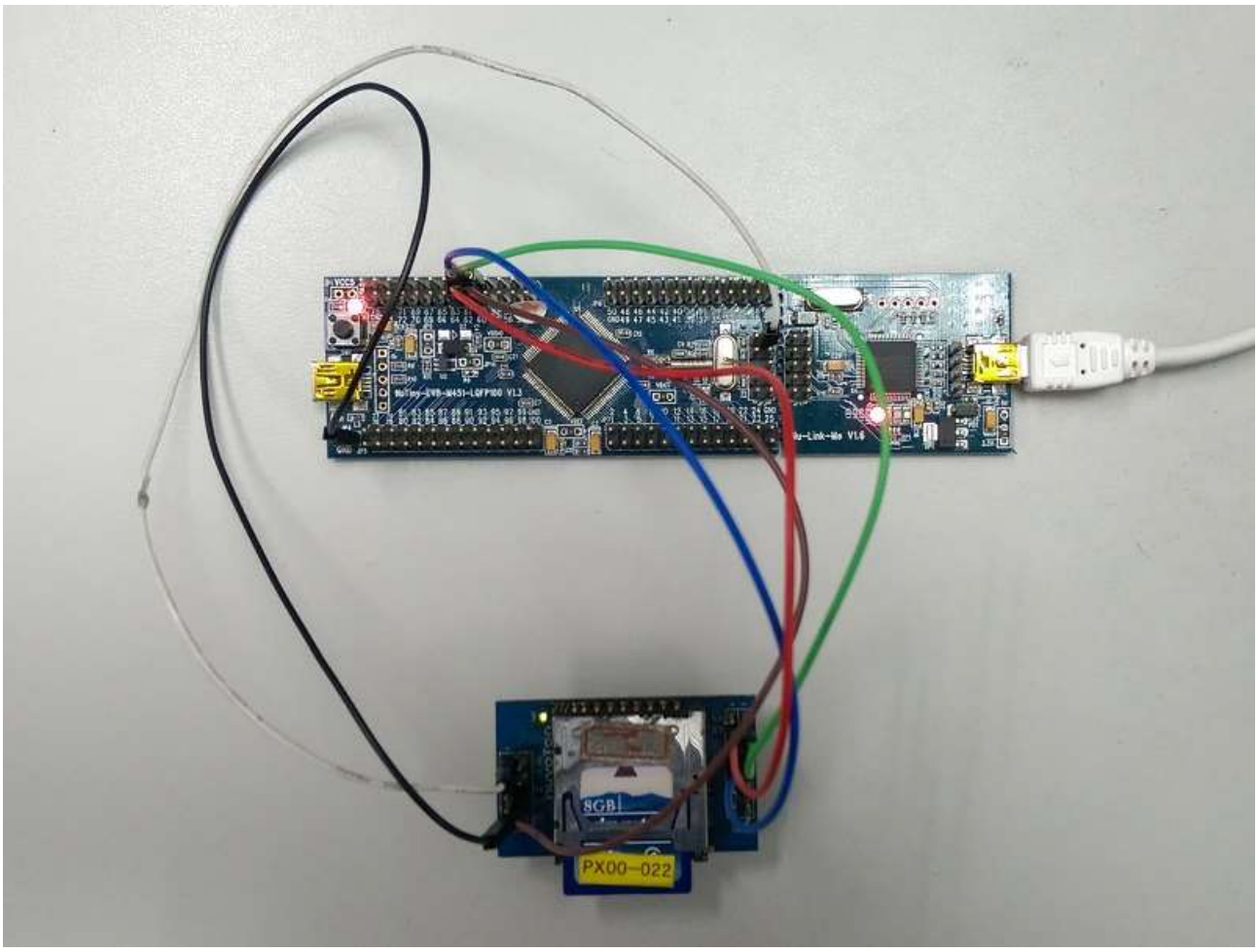
- Circuit components
  - NuTiny-EVB-M451-LQFP100 V1.3
  - SD card socket board
- Diagram

M451 uses SPI1 interface to access SD card. The connection between M451 and SD card as below:



- SPI CS is Chip Select. The SPI master is M451.
- SPI MOSI is Data Out of M451 and connect to Data In of SD card. It need pull-up resistors to avoid floating data.
- SPI SCK is clock. M451 provide clock to SD card to sync the data.

- SPI MISO is Data In of M451 and connect to Data Out of SD card. It need pull-up resistors to avoid floating data.



4 Directory Information

- EC_M451_FATFS_SPI_Mode_SDCard_V1.00
  - Library — Sample code header and source files
    - CMSIS — Cortex® Microcontroller Software Interface Standard (CMSIS) by Arm® Corp.
    - Device — CMSIS compliant device header file
    - StdDriver — All peripheral driver header and source files
  - SampleCode
    - ExampleCode — Source file of example code
      - M451_FATFS_SPI_Mode_SDCard — Source file of example code M451_FATFS_SPI_Mode_SDCard
        - FF8 — A generic FAT file system module, FatFs, for small embedded systems. Its official website is: http://elm-chan.org/fsw/ff/00index_e.html.

5 How to Execute Example Code

1. Browsing into sample code folder by Directory Information (section 4) and double click M451_FATFS_SPI_Mode_SDCard.uvproj.
2. Enter Keil compile mode
   a. Build
   b. Download
   c. Start/Stop debug session
3. Enter debug mode
   a. Run

6 Revision History

Date	Revision	Description
Sep. 10, 2019	1.00	1. Initially issued.

Important Notice

Nuvoton Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, "Insecure Usage".

Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, the control or operation of dynamic, brake or safety systems designed for vehicular use, traffic signal instruments, all types of safety devices, and other applications intended to support or sustain life.

All Insecure Usage shall be made at customer's risk, and in the event that third parties lay claims to Nuvoton as a result of customer's Insecure Usage, customer shall indemnify the damages and liabilities thus incurred by Nuvoton.

Please note that all data and specifications are subject to change without notice.
All the trademarks of products and companies mentioned in this datasheet belong to their respective owners.