Micro Lab

PROJECT REPORT

Mana Poustizadeh - Fateme Hashemi – Farzan Ravaghi – Soroush Sarabandi

AMIRKABIR UNIVERSITY OF TECHNOLOGY | HAFEZ AVENUE

گزارش اول پروژه

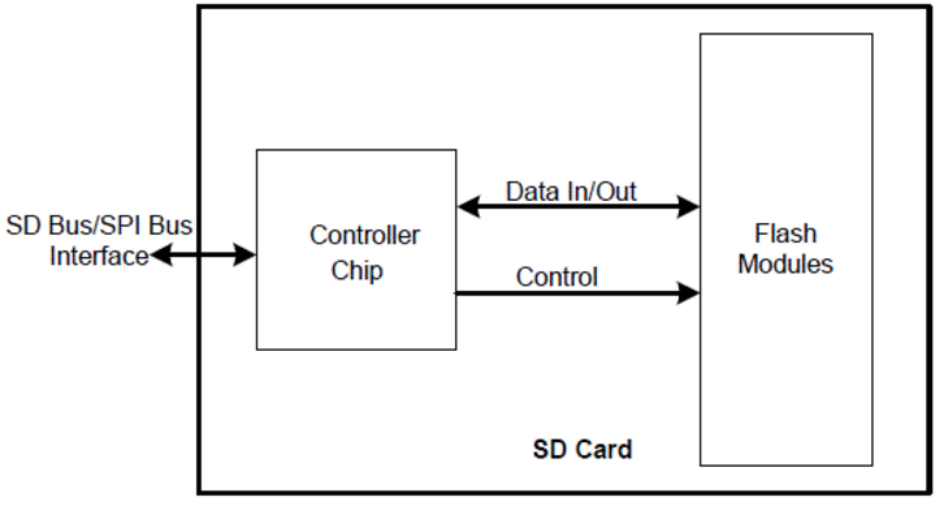
توضیح چگونگی کار uSD

همانطور که می‌دانید Micro SD قطعه‌ای برای ذخیره و خواندن داده‌هاست. داده‌ها بر روی SD card به صورت دیجیتال ذخیره می‌شوند. تراشه‌ای که درون بخش پلاستیکی قرار گرفته است از تعداد زیادی مدارهای الکتریکی ساخته شده‌اند. وقتی که در حال استفاده از SD card نیستیم، مدارهای آن مقادیر قبلی درون SD card را بدون نیاز به منبعی برای شارژ، حفظ می‌کنند. هنگامی که SD card در وسیله‌ای مثل دوربین یا موبایل قرار می‌گیرد، جریان الکتریکی کمی از وسیله، باعث حرکت الکترون‌های تراشه می‌شود. الگوی دیجیتالی که بر روی SD card ذخیره می‌شود، با داده‌های ذخیره شده روی آن مرتبط است. وارد شدن ولتاژ کمی بیشتری به SD card باعث می‌شود داده‌های روی آن پاک شوند و این امر امکان نوشتن مجدد روی SD card را فراهم می‌آورد. در اینجا می‌توان تنها بخش‌ها و قسمت‌هایی از SD card را پاک کرد.

SD card شامل دو بخش اصلی می‌شود: "هسته‌ی حافظه" و "کنترلر SD card "

هسته‌ی حافظه قسمتی از حافظه است که داده‌ی فایل در آن ذخیره می‌شود. با فرمت کردن SD card یک فایل سیستم در این قسمت نوشته می‌شود.

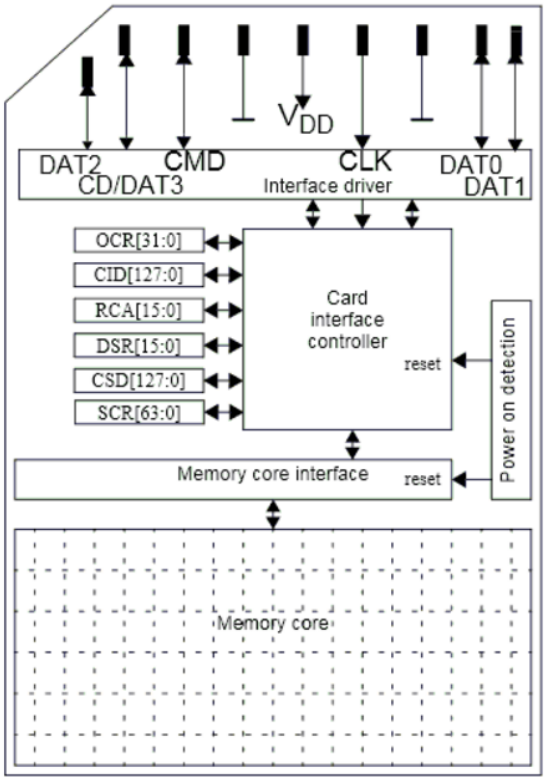
کنترلر SD card باعث می‌شود تا بتوانیم بین هسته‌ی حافظه و دستگاه‌های خارجی (مانند میکروکنترلرها) ارتباط برقرار کنیم.



ظرفیت SD card، همان اندازه‌ی هسته‌ی حافظه است. علاوه بر آن رجیسترهایی با بخش کنترلر همراه هستند که وظیفه‌ی نگهداری وضعیت SD card را برعهده دارند و از نوع فقط خواندنی هستند.

SD card می‌تواند به وسیله‌ی باس سریال با میکروکنترلر ارتباط برقرار کند و از باس‌های SD یا باس‌های SPI در این جهت استفاده کند. باس SD مختص ارتباطات با سرعت زیاد است و باس SPI با سرعت کم کار می‌کند. میکروکنترلر می‌تواند داده‌ها

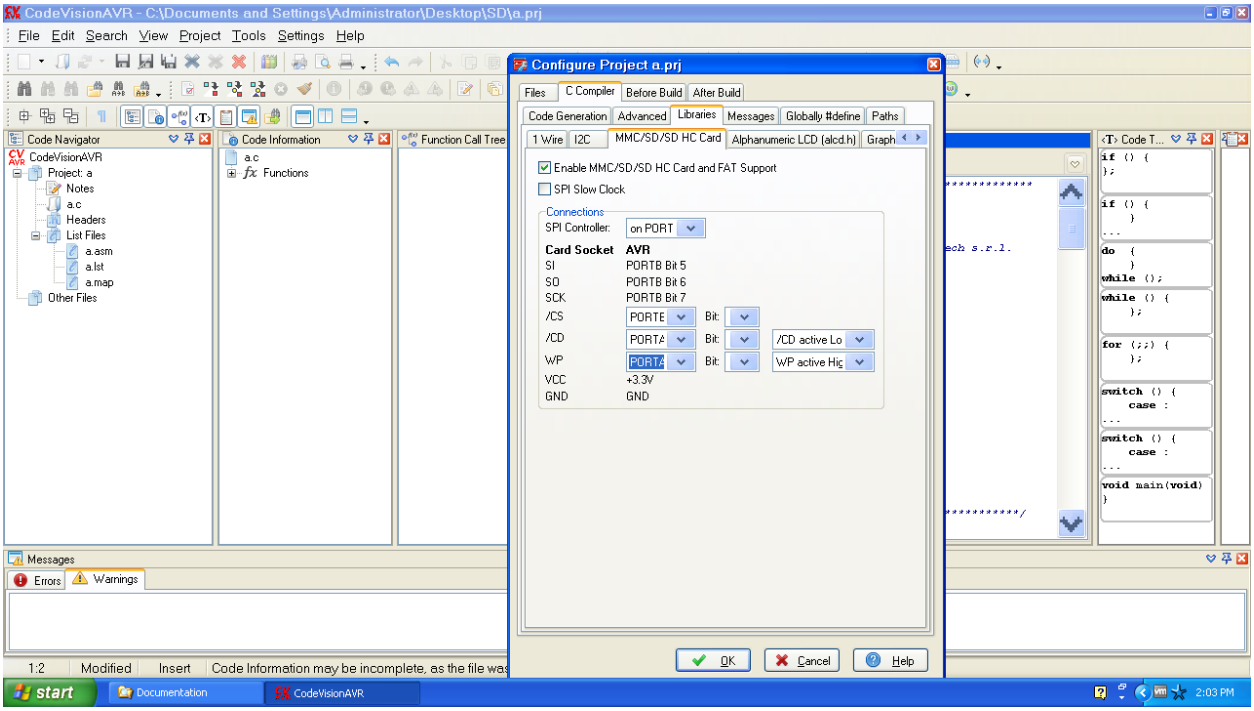
را در حافظه بنویسد یا از آن بخواند و همچنین رجیسترها (موجود در کنترلر) را با استفاده از دستورهای SD که از طریق این باس‌های سریال ارسال شده‌اند، می‌خواند.



SD Memory Card architecture

تنظیمات و توابع و libraryها

بعد از ساختن پروژه در قسمت Project>Configure رفته و از قسمت Tab C Compiler قسمت Libraries قسمت MMC/SD/SD HC Card را انتخاب می‌کنیم.



در اینجا با فعال سازی گزینه ی Enable MMC/SD/SD HC Card and FAT Support امکان اتصال Micro Controllerرا به SD Cardاز طریق SPI به عنوان protocol ارتباطی را فراهم می‌کنیم.

تنظیم دوم SPI Slow Clock است که با زدن آن Clock ما در ارتباط SPI، 2 برابر کند تر می‌شود (در نتیجه نرخ انتقال اطلاعات کمتر می‌شود.) از این گزینه برای سازگاری با سخت افزارهای قدیمی‌تر استفاده می‌شود.

گزینه‌های بعدی را می‌بایست با توجه به نحوه ی سیم کشی و اتصال Micro مان به SD Card Controller تعیین کنیم و در آن تعیین می‌کنیم که کدام bit از Port انتخاب شده برای ارتباط SPI هر یک از نقش‌های SPI را به عهده می‌گیرد.

نقش‌های وجود در protocol SPI عبارتند از :

- SCLK: Serial Clock (output from master).

- MOSI: Master Output Slave Input, or Master Out Slave In (data output from master).
- MISO: Master Input Slave Output, or Master In Slave Out (data output from slave).
- SDIO: Serial Data I/O (bidirectional I/O)
- SS: Slave Select (often active low, output from maste

در ضمن بسته به مشخصات سخت افزار SDCard ، Signal مورد استفاده برای Write protect کردن SD Card استفاده می‌شود که می‌تواند Active low یا Active high باشد.

توابع کار با SD Card

به دو روش می‌توان با SD Card کار کرد یکی Low Level Access است و دیگری FAT Access

Low Level Access

ما با استفاده از کتاب‌خانه‌ی sdcard.lib که توابع آن در sdcard.h، اعلان شده اند به sd card دسترسی خواهیم داشت.

توابع آن عبارتند از :

void disk_timerproc (void)

باید هر 10 ثانیه از طریق timer صدا شود.

unsigned char disk_initialize(unsigned char drv)

SD card را برای دسترسی آماده می‌کند

DRESULT disk_read (unsigned char drv, unsigned char* buff, unsigned long sector, unsigned char count)

در این

DRESULT disk_write (unsigned char drv, unsigned char* buff, unsigned long sector, unsigned char count)

FAT Access

در این حالت دسترسی که به کمک کتاب خانه ی ff.lib که توابع آن در ff.h اعلان شده اند ما دسترسی سطح بالای تری به SDCard داریم که در آن امکان ساخت فایل و نوشتن و خواندن در سطح فایل فراهم است.

کتاب خانه ی ff خود از کتاب خانه ی sdcard استفاده می‌کند.

توابع این کتابخانه عبارت اند از :

FRESULT f_mount(unsigned char vol, FATFS *fs);

با خواندن داده ساختارهای FAT ذخیره شده بر روی SD Card آن را برای کار اماده می‌کند

FRESULT f_open(FIL* fp, const char* path, unsigned char mode);

در مسیر path یک فایل با حالت دسترسی تعیین شده در mode باز می‌کند و یک file point بر می‌گرداند که از آن می‌توان برای read یا write استفاده کرد.

FRESULT f_write(FIL* fp, const void* buff, unsigned int btw, unsigned int* bw);

Buff را در فایل مشخص شده با fp می‌نویسد

FRESULT f_read(FIL* fp, void* buff, unsigned int btr, unsigned int* br);

Buff را به تعداد byte to read از فایل پر می‌کند

FRESULT f_lseek (FIL* fp, unsigned long ofs);

برای جا به جایی file pointer در داخل فایل استفاده می‌شود و آن را به byte مشخص شده در ofs می‌برد.