Giao tiếp thẻ nhớ MMC & SD Card

Thẻ nhớ MMC

MultiMediaCard (MMC) là thẻ nhớ có bộ nhớ flash chuẩn. Ra mắt năm 1997 bởi Siemens AG và SanDisk, nó dùng công nghệ bộ nhớ dựa trên NAND của Toshiba, và như vậy là nó sẽ nhỏ hơn bộ nhớ dựa trên NOR của Intel như CompactFlash. Kích thước của thẻ MMC bằng một con tem thư: 24 mm x 32 mm x 1.4 mm. Thẻ MMC gốc thường dùng giao diên serial 1-bit, nhưng các phiên bản mới hơn chỉ định cho phép trao chuyên 4 hoặc 8 bit trong một thời điêm. Thẻ MMC được dùng nhiêu hoặc ít hơn bởi có các thẻ Secure Digital card (SD card), nhưng vân có ý nghĩa bởi thẻ MMC vân dùng được trong các thiêt bị sử dụng các thẻ SD.
Tóm lại, một thẻ MMC sử dụng như một bộ nhớ lưu trữ cho thiêt bị di động, có thê dê dàng gỡ bỏ khi kêt nối với máy tính cá nhân. Lây ví dụ, một máy ảnh số sử dụng MMC đê lưu trữ các tệp ảnh. Với một đầu đọc thẻ MMC (thường là một hộp nhỏ kêt nôi qua công USB hoặc qua kêt nôi serial, hoặc được găn sãn ở máy tính), người dùng có thê lây các bức ảnh đã chụp vào máy tính. Các máy tính hiện đại, cả xách tay và đê bàn, thường có khe cắm thẻ SD, có thê đọc thẻ MMC nêu trình điều khiên của hệ điều hành hỗ trợ.
Thẻ MMC có dung lượng lớn như 4 GB và 8 GB . Chúng được dùng phần lớn trong những thiêt bị sử dụng thẻ nhớ, như điện thoại di động, máy nghe nhạc kỹ thuật số, máy ảnh số và thiêt bị trợ giúp kỹ thuật số cá nhân. Từ khi xuất hiện thẻ Secure Digital card và thẻ Secure Digital SDIO rất ít công ty cho sản xuất thiêt bị có khe cắm MMC (ngoại trừ Nokia 9300 communicator, dùng thẻ MMC có kích thước nhỏ), nhưng vì mỏng hơn , có chân cắm tương thích nên thẻ MMC có thê sử dụng trên phần lớn thiêt bị hỗ trợ thẻ SD nêu phần sụn / phần mềm được hỗ trợ.

2: Thẻ nhớ SD

Secure Digital (SD) là loại thẻ nhớ được phát triên bởi Matsush*te, SanDisk và Toshiba đê sử dụng trong các thiêt bị cầm tay. Hôm nay nó được sử dụng rộng rãitrong máy ảnh kỹ thuật số, máy quay phim kỹ thuật số, máy tính cầm tay, PDA, các máy nghe nhạc, điện thoại di động, GPS receivers, và trò chơi video Tiêu chuẩn SD dung lượng thẻ khoảng từ 1 MB đên 2 GB Phạm vi năng lực cho công suất cao SDHC chồng lên thẻ hơi, bắt đầu lúc 4 GB nhưng đạt cao đên 32 GB như của mid-2009. Các SDXC (Extended Capacity), một đặc điêm mới công bô tại 2009 Consumer Electronics Show, sẽ cho phép cho đên 2 thẻ suất lao.
Định dạng này đã được chứng minh rất phô biên. Thay đôi giao diện của các định dạng được thành lập đã thực hiện một số thiêt bị cũ được thiêt kê cho chuẩn SD card (≤ 4GB) không thê xử lý các định dạng mới hơn như SDHC (≥ 4GB). Mọi SD-tử có hình dạng giống vật chất và yêu tố hình thức tuy nhiên, gây nhầm lân cho một số người tiêu dùng
1.3: giao tiêp giữa 8051 và thẻ nhớ MMC/SD
Có rất nhiêu cách đê giao tiêp giữa 8051 và MMC/SD memory . Sau đây em xin giới thiệu một số cách :
- Kêt nối qua SPI
- Giao tiêp MMC qua PIC

- Giao tiếp FAT32
- Giao tiếp qua FAT16 filesystem

-giao tiếp 8051 và mega 32

PHẦN 2: NỘI DUNG CỦA CÁCH PHƯƠNG PHÁP TRÊN

2.1: GIAO TIẾP 8051 VA MMC/SD BẰNG FAT16 FILESYSTEM

2.1.1: chuyển và nhận byte trên bus SPI

chức năng cơ bản nhất là chuyên 1 byte duy nhất trên bus SPI. Chức năng này được sử dụng trong tài liêuj sau đây, và là cách chúng gửi và nhận thông tin trên bus SPI. Các vi điều khiên upsd3334D-bus có một giao diện đầy đủ song song, do đó, khi bạn gửi một bytethì bạn sẽ nhận được 1 byte . Chức năng này thực hiện các công handshaking cần thiết đê gửi và nhận được một byte.

Mã:

```
BYTE SPI_Byte( BYTE ThisByte )
{
  while( !(SPISTAT&TISF) );
  while( (SPISTAT&BUSY) );
  SPITDR = ThisByte;
  while( !(SPISTAT&RISF) );
  while( (SPISTAT&BUSY) );
  return( SPIRDR );
}
```

2.1.2: Chip SPI

Một chức năng cơ bản là tạo điều kiện cho dòng CS vào thẻ SD. Kê từ khi thẻ SD có một CS phủ định (các tín hiệu thấp đê chọn thẻ), chúng thực hiện lệnh 0 khi chọn thẻ, và '1 'khi chúng tôi không chọn thẻ. Hai chức năng sau thực hiện quá trình này, nhưng một lần nữa - chúng được cụ thê cho việc thiêt kê phần cứng, và sẽ khác hơn này.

Mã:

```
void SPI_EnableCS()
{
  P4 &= 0x7F;       /* bật CS  */
}

void SPI_DisableCS()
{
  P4 |= 0xF0;       // Tắt CS  */
}
```

2.1.3. Thiêt lập tần số cho SPI

PI ClockThiêt lập tần số Clock cho Bus SPI được thực hiện bởi chức năng này. Lưu ý, dựa trên những định nghĩa trên, đạt tốc độ tôi đa cho vi điều khiên này băng cách sử dụng tinh thê thạch anh 22Mhz là 5MHz.

Mã lệnh:

```
void SPI_Init( enum SPI_FREQUENCIES ThisFrequency )
{
  SPI_DisableCS();     /* chip chon vô hiệu hóa nêu nó được kick hoạt  */

  /* thiêt lâp SPI kiêm soát đăng kí */
  SPICON0 = SPIEN | TE | RE;          /* cho phê SPI , Tx và Rx  */
  SPICON1 = 0x00;         & amp; amp; nbsp;         & amp; amp; nbsp; // không ngắt
  switch( ThisFrequency )          ; ; ;   // cài đặt tần số
  {
    Trường hợp 10Mhz:
      SPICLKD = SPI_FREQUENCY_10MHz;
      break;
    Trường hợp 25 Mhz:
      SPICLKD = SPI_FREQUENCY_5MHz;
      break;
    Trường hợp 1 Mhz:
      SPICLKD = SPI_FREQUENCY_1MHz;
```

```
      break;
    Trường hợp 400 Mhz:
    default:
      SPICLKD = SPI_FREQUENCY_400KHz;
      break;
  }
}
```

2.1.4: Gửi lệnh tới SPI

Gửi một lệnh đơn, và đọc các phản hồi từ thẻ này là tiếp theo khối một cách hợp lý. Đối với một mô tả chi tiết về câu trúc lệnh, bạn phải tham khảo các Đặc điểm kỹ thuật sản phẩm băng tay, có tiêu đề "SanDisk Secure Digital card, sản phẩm băng tay, Phiên bản 1.9, Văn bản số 80-13-00169, Tháng 12 năm 2003". Chức năng tôi sử dụng để làm điều này là như sau:

Mã lệnh:

```
#define CMD_GO_IDLE_STATE       & amp; amp; nbsp; 0
#define CMD_SEND_OP_COND        &a mp;a mp;n bsp;  1
#define CMD_SEND_CSD            9
#define CMD_SEND_CID            10
#define CMD_STOP_TRANSMISSION   12
#define CMD_SEND_STATUS         &am p;am p;nb sp;  13
#define CMD_SET_BLOCKLEN        &a mp;a mp;n bsp;  16
#define CMD_READ_SINGLE_BLOCK   17
#define CMD_READ_MULTIPLE_BLOCK 18
#define CMD_WRITE_SINGLE_BLOCK  24
#define CMD_WRITE_MULTIPLE_BLOCK 25
#define CMD_PROGRAM_CSD         &am p;am p;nb sp;  27
#define CMD_SET_WRITE_PROT      28
#define CMD_CLR_WRITE_PROT      29
#define CMD_SEND_WRITE_PROT     30
#define CMD_TAG_SECTOR_START    32
#define CMD_TAG_SECTOR_END      33
#define CMD_UNTAG_SECTOR        &a mp;a mp;n bsp;  34
#define CMD_TAG_ERASE_GROUP_START 35
#define CMD_TAG_ERASE_GROUP_END 36
#define CMD_UNTAG_ERASE_GROUP   37
#define CMD_ERASE               &am p;am p;nb sp;  38
#define CMD_LOCK_UNLOCK         &am p;am p;nb sp;  42
#define CMD_APP_CMD             & amp; amp; nbsp;  55
#define CMD_READ_OCR            58
#define CMD_CRC_ON_OFF          & ; ;nbs p;  59
#define ACMD_SEND_OP_COND       & amp; amp; nbsp; 41
```

Tập lệnh chung

```
{
  BYTE Index[6];
  struct
    {
    BYTE lệnh;
    ULONG đối số;
    BYTE Cksum;
  } CA;
} CommandStructure;
```

tập lệnh chung

```
{
  BYTE b[4];
  ULONG ul;
} b_ul;

BYTE SD_Command( BYTE ThisCommand, ULONG ThisArgument )
{
  b_ul Temp;
  BYTE i;
```

```
  /* cho phép các thiết bị */
  SPI_EnableCS();

  /* Gửi đồng hồ đệm để đảm bảo không có hoạt động đang chờ giải quyết */
  SPI_Byte( 0xFF );

  /* gửi lệnh */
  SPI_Byte(0x40 | ThisCommand);

  /* gửi đối số*/
  Temp.ul = ThisArgument;
  for( i=0; i<4; i++ )
    SPI_Byte( Temp.b[ i ] );

  /* gửi CRC */
  SPI_Byte((ThisCommand == CMD_GO_IDLE_STATE)? 0x95:0xFF);

  /*Gửi đồng hồ đệm để đảm bảo thẻ đã hoàn tất các hoạt động */
  SPI_Byte( 0xFF );
  return( 0 );
}
```

2.1.5: Đọc phản hồi từ thẻ SD

Mỗi lệnh gửi đến thẻ SD sẽ có phản hồi từ thẻ. Kích cỡ của phản hồi này, cùng với nội dung của các phản hồi này, phụ thuộc vào lệnh gửi đi được. Dưới đây là hai các chức năng mà có thể được sử dụng để đọc byte phản hồi từ thẻ SD.

Mã lệnh:

```
BYTE SD_GetR1()
{
  BYTE i, j;

  for( i=0; i<8; i++ )
  {          ; ; ;          ; ; ;    /* trả lời sau 1-8s*/
    j = SPI_Byte( 0xff );
    if(j != 0xff)        /* nếu nó không phải là 0xff, nó là một phản ứng */
      return(j);
  }
  return(j);
}

WORD SD_GetR2()
{
  idata WORD R2;

  R2 = ((SD_GetR1())<< 8)&0xff00;
  R2 |= SPI_Byte( 0xff );
  return( R2 );
}
```

2.1.6: Delay và Thời gian chức năng

Cuối cùng, hướng dẫn này đòi hỏi một chức năng Delay và chức năng một thời gian - Chức năng sử dụng trong suốt được gọi là Delay (), và giá trị được thông qua trong mili giây. Bạn có thể đạt được điều này theo những cách khác nhau, nhưng cách đơn giản nhất là sử dụng một ngắt.

Chức năng chậm trễ của tôi sử dụng một ngắt có cập nhật một biến hệ thống gọi là "Mã CK" mỗi millisecond. Để tính toán sự chậm trễ, tôi chờ đợi ticker để đếm số mili giây chờ đợi. Chức năng trễ của tôi trông như thế này

Mã lệnh:

```
 void Delay( WORD MilSec )
{
  ULONG xdata DelayTickValue;

  /* tính toán giá trị đánh dấu từ đây đến mili giây sau đó*/
  DelayTickValue = Ticker + MilSec *( (float)(TICKS_PER_SECOND) / 1000.0 );

  /* chờ đến khi đánh dấu vào điểm chuẩn */
  while( Ticker < DelayTickValue );
```

```
}
```

Chức năng thời gian là một yêu cầu của thư viện Chân, và được sử dụng cho thời gian dán tem tập tin được tạo ra trong FAT này. Kê từ phần cứng của tôi hỗ trợ một đồng hồ thời gian thực, chức năng thời gian của tôi trông như thê này:

Mã lệnh:

```
DWORD get_fattime ()
(
RTC_CURRENT rtc;
RTC_read (& rtc);
trở lại ((DWORD) ((WORD) (rtc. năm) + 20) <<25)
| ((DWORD) rtc Tháng <. <21)
| ((DWORD) rtc Ngày <. <16)
| ((DWORD) rtc Giờ <. <11)
| ((DWORD) rtc Phút <. <5)
| ((DWORD) rtc Giây.>>
```

2.1.7: Thiết lập công SPI trong statup A51

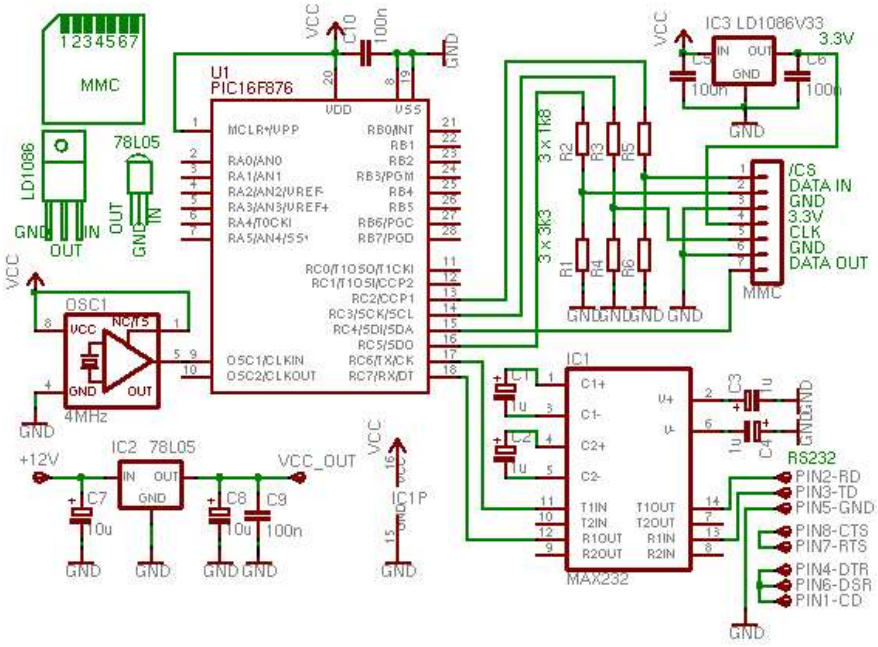
Đối với các upsd3334D, các SFR của răng đặt các khía cạnh chức năng cho các công SPI là P4SFS0 và P4SFS1. Vi điều khiển của bạn sẽ khác nhau ở đây, nhưng ý tưởng chính là đê thiêt lập các chân SPI đê bạn có một M0SI, miso, SPICLK, và CS - Những tên này tương ứng với "Micro NGOÀI TRONG SIave", "Micro TRONG SIave OUT", "SPI Clock ", và" chip Chọn "tương ứng.

Đối với vi điều khiên của tôi, tôi đặt công cho các cài đặt sau, từ bên trong file STARTUP.A51 (trong lăp ráp):

Mã lệnh:

```
; Chương trình uPSD Port-4 đăng ký ... ;
;;
; P4SFS0 - tập hợp các chức năng chính của các chân;
; Mặc định là '0 ', mà là GPIO pin;
;;
; 0 - buzzer đầu ra (PWM) (đặt làm thay thê - 1);
; 1 - SPIADDRESS SELECT 0 - |;
; 2 - SPIADDRESS SELECT 1 | này được sử dụng như là 3-8;
; Bộ giải mã cho SPI chọn thiêt bị;
; 3 - SPIADDRESS SELECT 2 - | (đặt làm GPIO - 0);
; 4 spiclock - (đặt làm thay thê - 1);
; 5 - Miso (đặt làm thay thê - 1);
; 6 - MOSI (đặt làm thay thê - 1);
; 7 - spi dân sử dụng dòng chọn (đặt làm GPIO - 0);
;------------------------------------------------ -----------;
; P4SFS1 - tập hợp các chức năng thay thê,;
; Nêu tương ứng với bit trong P4SFS0 được thiêt lập;
;;
; 0 - PCA0 Module 0, TCM0 (đặt làm 0);
; 1 - bỏ qua, kê từ P4SFS0 đã là 0 (đặt là 0);
; 2 - bỏ qua, kê từ P4SFS0 đã là 0 (đặt là 0);
; 3 - bỏ qua, kê từ P4SFS0 đã là 0 (đặt là 0);
; 4 - SPI Clock, SPICLK (đặt làm 1);
; 5 - Nhận SPI, SPIRXD (đặt làm 1);
; 6 - SPI Transmit, SPITXD (đặt làm 1);
; 7 - bỏ qua, kê từ P4SFS0 đã là 0 (đặt là 0);
;;
;------------------------------------------------ -----------;
MOV P4SFS0, # 071H
MOV P4SFS1, # 070H
```

2.2: GIAO TIÊP MMC/SD & PIC



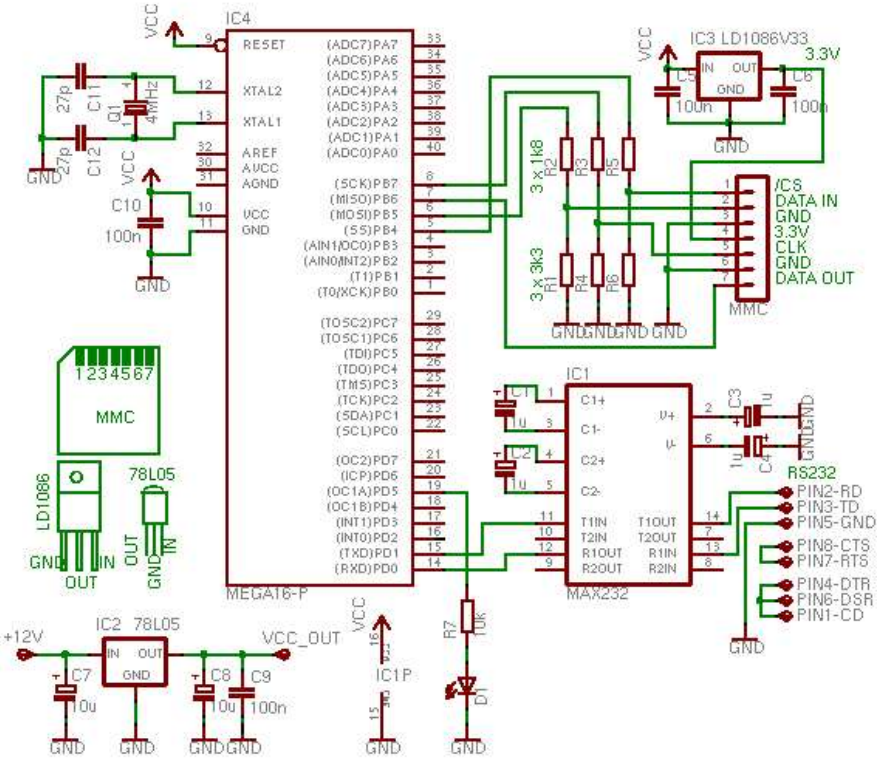
PIC - MMC (Multi Media Card) Flash Bộ nhớ mở rộng

Đơn giản là một giao diện MMC (MultiMediaCard) với PIC Microchip thông qua SPI (Serial Port Interface). Đây là một dữ liệu rất tiện khai thác gỗ mạch với nhiều bộ nhớ để lưu trữ dữ liệu I2C RAM.'s Hoặc EEPROM's là khó có sẵn tại các kích thước lớn hơn 256Kb, nhưng giải pháp này với 64MB Flash MMC không phải là để đánh bại trong cả hai hiệu quả chi phí và khối lượng lưu

Phiên bản này sử dụng một PIC16F876, nhưng nó dễ dàng cho ra những sơ đồ và phần mềm khác của PIC.
MMC được kết nối với các chân SPI của PIC qua ngăn điện áp điện trở đơn giản để chuyển đổi các +5 V cấp cao về 3.3V được sử dụng bởi MMC. Điện áp cung cấp cho MMC (2.7V - 3.6V) xuất phát từ một điều áp LD1086V33 (3.3V) hoặc tương đương. Có thể chạy PIC ở 3.3V, nhưng sau đó chuyển đổi AD không được ổn định như lúc 5V-Các dữ liệu. Ra pin từ MMC đi trực tiếp vào PIC, bởi vì 3.3V là cao cho các PIC.

2.3:GIAO TIẾP FAT32 GIAO TIẾP VỚI THẺ NHỚ MMC/SD

GIAO TIẾP MEGA32:



Dự án này triển khai một hệ thống thu dữ liệu tốc độ cao bằng cách sử dụng vi điều khiển và Mega32 Controller Area Network (CAN).

Ghi dữ liệu là điều cần thiết để thử nghiệm và phát triển một racecar. Ghi những gì từng cảm biến được thực hiện có thể cho biết một kỹ thuật làm thế nào chiếc xe đang hoạt động, và quan trọng nhất, làm thế nào để làm cho nó nhanh hơn. .Một chiếc xe outfitted cũng có thể có nhiều cảm biến, với Công thức Một trong những chiếc xe cũng có hơn 100 bộ cảm ứng. Xe FSAE Cornell đã trên 50 cảm biến trên đó, nhiều người trong đó có yêu cầu cao tỷ lệ lấy mẫu để có ích. Dữ liệu hệ thống thương mại được mua lại hoặc tốn kém, chậm, hoặc có vài yếu tố đầu vào. Một giải pháp cho vấn đề này đã được cố gắng bởi trước đó 476 học sinh (Karl Antle và Ryan McDaniel) sử dụng một PIC18F2585, nhưng nó chỉ có thể đăng nhập đáng tin cậy ở 150 Hz. Nhiều bộ cảm biến trên xe yêu cầu cao hơn nhiều mức lấy mẫu, như là các bộ cảm biến được ghi lại các sự kiện xảy ra trong một thời gian rất ngắn thời gian. Ví dụ, khi nhìn vào một vết sưng sắc nét bằng cách sử dụng vị trí rocker sự kiện có thể chỉ cuối 0,05 giây khi lái xe một cách nhanh chóng. Nếu lấy đạo hàm của các dữ liệu lấy mẫu một tỷ lệ rất cao là cần thiết để cung cấp cho dữ liệu hữu ích, ít nhất là 500 HZ. Nó được điều này cần thiết phải mua lại dữ liệu tốc độ cao mà chúng tôi có động cơ để tạo ra một hệ thống thu dữ liệu tốc độ cao để thay thế hiện một.

Tham khảo từ các trang nước ngoài & bài dịch được dịch bởi:

Họ và tên: Mai Trần Điều
Lớp : DHDT3TB

Comments

You do not have permission to add comments.