

Главная

Новости

Уроки по программированию МК

Устройства и интерфейсы

Ссылки

Ф

Свежие комментарии

- Сергей к записи [PIC Урок 3. Бегущие огни](#)
- Narod Stream к записи [PIC Урок 5. Таймеры](#)
- Артем к записи [PIC Урок 5. Таймеры](#)
- Narod Stream к записи [AVR Урок 13. ШИМ. Мигаем светодиодом плавно. Часть 1](#)
- Narod Stream к записи [STM Урок 10. HAL. Изучаем PWM \(ШИМ\). Мигаем светодиодами плавно](#)

Форум. Последние ответы

- [Narod Stream](#) в [Программирование МК STM32](#)
2 дн., 3 час. назад
- [Zandy](#) в [Программирование МК STM32](#)
2 дн., 10 час. назад
- [Narod Stream](#) в [Программирование МК STM32](#)
1 неделя, 6 дн. назад
- [Narod Stream](#) в [Программирование МК STM32](#)
1 неделя, 6 дн. назад
- [fireweb](#) в [Программирование МК STM32](#)
2 нед., 2 дн. назад

Январь 2018

Пн	Вт	Ср	Чт	Пт	Сб	Вс
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				
« Дек						

Архивы

- Январь 2018
- Декабрь 2017
- Ноябрь 2017
- Октябрь 2017
- Сентябрь 2017
- Август 2017
- Июль 2017
- Июнь 2017
- Май 2017

Главная > Программирование AVR > AVR Урок 33.
SPI. Карта SD. FAT. Часть 3

AVR Урок 33. SPI. Карта SD. FAT. Часть 3

Posted on [Январь 11, 2017](#) by [Narod Stream](#)
Опубликовано в [Программирование AVR](#)
— 2 комментария ↓

Яндекс.Директ

Печатные Платы на
Изготовление печатных плат партий. Звони
[pcb.electropribor-penza.ru](#) Ад

Яндекс.Директ

Печатная плата - Тайвань
Изготовление печатных плат и контрактное производство под ключ в Тайване
[explus.com.tw](#) Адрес и телефон

Урок 33 Часть 3

SPI. Карта SD. FAT

Продолжим начатое дело по программированию и использованию флеш-карты SD. В [предыдущей части](#) нашего урока мы написали инициализацию нашей карты, теперь пришло время что-то с неё считать, а может даже и записать, пока не обращая внимание на структурирование данных в виде файловой системы.

Поэтому создадим функцию для записи блока с карты SD выше функции main(). Почему сразу записи, да потому, что читать нам ещё нечего, так как сначала надо что-то записать, чтобы потом прочитать и убедиться, что это именно то, что мы записали

```
unsigned char SD_Write_Block (char*  
bf, unsigned char dt1, unsigned char  
dt2, unsigned char dt3, unsigned  
char dt4)  
{  
}
```

Вот сколько у нас входных параметров. Почему много, потому что во-первых мы передаём указатель на

[Программирование МК PIC](#)

[Тесты устройств и аксессуаров](#)

Яндекс.Директ

**Программируе
мые
логические
модули**
Сравните лучшие
ПЛК! Тех.поддержка,
скидки, уникальный
фильтр и гарантия!
[ipc2u.ru](#)
Адрес и телефон

**Стержневой
фторопласт
выгодно!**
Фторопласт Ф-4 в
стержнях. Д. 6-
300 мм. Выгодные
цены! Отличное
качество!
[Каталог материалов](#)
[Таблица весов](#)
[Цены](#) [Контакты](#)
[pezn.ru](#)
Адрес и телефон

Заходите на канал
Narod Stream

- Март 2017
- Февраль 2017
- Январь 2017
- Декабрь 2016
- Ноябрь 2016

место в памяти, откуда мы будем писать данные в блок флеш-памяти карты. Во-вторых, чтобы инициировать передачу блока данных во флеш-карту, мы будем передавать определённую команду, в которой предусмотрено 4 основных параметра.

Создадим в функции две переменных `unsigned char SD_Write_Block (char* bf, unsigned char dt1, unsigned char dt2, unsigned char dt3, unsigned char dt4)`

```
{
    unsigned char result;
    long int cnt;
```

Теперь команда. Посмотрим техническую документацию и найдём команду для записи блока

CMD INDEX	type	argument	resp	abbreviation	command description
CMD16	ac	[31:0] block length	R1	SET_BLOCKLEN	See description in Table 4-19
CMD24	adc	[31:0] data address*	R1	WRITE_BLOCK	In the case of a Standard Capacity SD Memory Card, this command writes a block of the size selected by the SET_BLOCKLEN command. In the case of a High Capacity Card, block length is fixed 512 Bytes regardless of the SET_BLOCKLEN command.

Индекс команды у нас 24 или шестнадцатеричному — 0x18, первый бит, как мы знаем 0, второй всегда 1, поэтому будет 0x58.

Основным аргументом будет адрес данных. Поэтому напомним вызов функции передачи команды

```
long int cnt;
result=SD_cmd(0x58,dt1,dt2,dt3,dt4,0x95); //CMD24 даташит стр 51 и 97-98
```

Так как все параметры у нас во входе функции, то мы их пока не увидим, увидим, когда будем нашу функцию записи вызывать.

Выйдем, если что-то не то, если всё то, то вычистим мусор из регистра сдвига карты

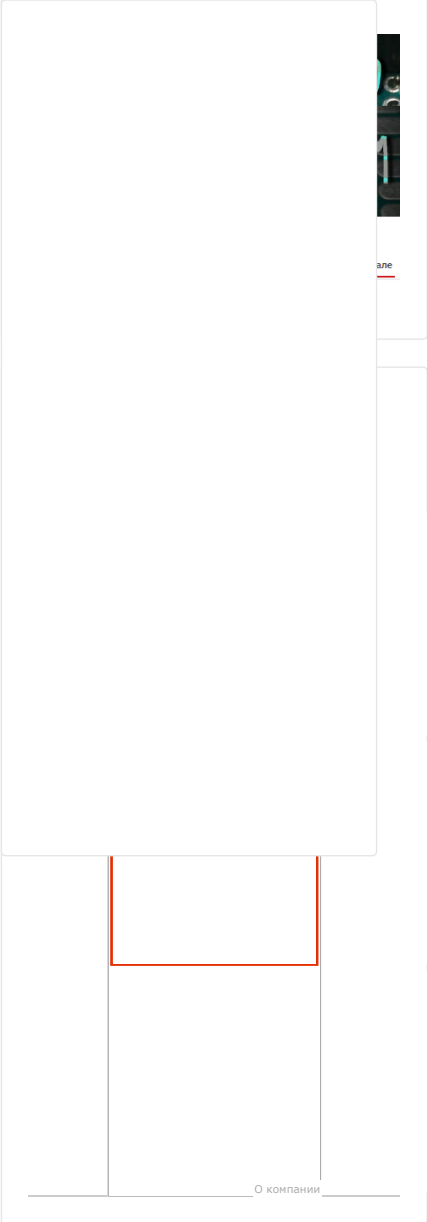
```
result=SD_cmd(0x58,dt1,dt2,dt3,dt4,0x95); //CMD24 даташит стр 51 и 97-98
if (result!=0x00) return 6; //Выйти, если результат не 0x00
SPI_SendByte (0xFF);
```

Вообще мы будем пробовать писать данные по адресу 0x400, так как если мы вдруг будем позже пробовать 32-битную файловую систему, то в первые байты мы писать не можем, так как там служебная информация файловой системы FAT32. В 0x300 мы писать не можем также, потому что блоки у нас по 512 байт и мы просто с такого адреса писать не можем, это не адрес блока.

Передадим начало буфера

```
SPI_SendByte (0xFF);
SPI_SendByte (0xFE); //Начало буфера
```

Начало буфера — это своего рода метка, она входит в пакет передачи данных, состоящий из метки начала и собственно самих данных

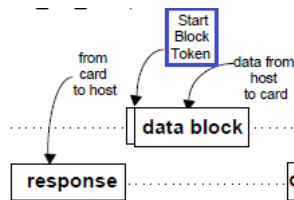


Рубрики

- 1-WIRE (3)
- ADC (6)
- DAC (4)
- GPIO (25)
- I2C (19)
- SPI (13)
- USART (8)
- Программирование AVR (131)
- Программирование PIC (6)
- Программирование STM32 (211)
- Тесты устройств и аксессуаров (1)

➤

31 ДЕНЬ	113 936
	12 222
07 ДНЕЙ	30 497
	4 195
24 ЧАСА	5 599
	1 152
СЕГОДНЯ	2 187
	529
НАПЯНШ	204
	40



Именно такая метка (0b11111110) должна быть для CMD 24, CMD17 и CMD18

First byte: Start Block

7								0
1	1	1	1	1	1	1	1	0

Ну, и теперь непосредственно передаём данные из буфера и в конце контрольную сумму и любой байт

```
SPI_SendByte (0xFE); //Начало буфера
for (cnt=0;cnt<512;cnt++)
SPI_SendByte(bf[cnt]); //Данные
SPI_SendByte (0xFF); //Контрольная
сумма
SPI_SendByte (0xFF);
```

Контрольная сумма любая, поэтому по этому поводу не заморачиваемся.

Теперь примем результат команды из шины и произведём первичную проверку

```
SPI_SendByte (0xFF);
result=SPI_ReceiveByte();
if ((result&0x05)!=0x05) return 6;
//Выйти, если результат не 0x05
(Даташит стр 111)
```

Аналогично, как и в предыдущей функции, создадим цикл и дождёмся свободного состояния карты, и, если всё нормально, то возвращаем 0

```
if ((result&0x05)!=0x05) return 6;
//Выйти, если результат не 0x05
(Даташит стр 111)
cnt=0;
do { //Ждем окончания состояния
BUSY
result=SPI_ReceiveByte();
cnt++;
} while ( (result!=0xFF)&&
(cnt<0xFFFF) );
if (cnt>0xFFFF) return 6;
return 0;
}
```

Вызовем нашу функцию в main() и отобразим результат на дисплее в следующей строке

```
str_lcd(str);
result=SD_Write_Block(buffer,0x00,0x
00,0x04,0x00); //Запишем блок из
буфера
sprintf(str,"%d",result);
setpos(0,1);
str_lcd(str);
```

В первом параметре у нас буфер, в котором у нас есть строка, с которой мы

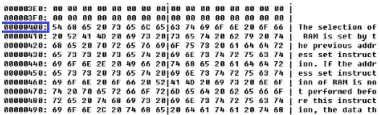
игрались в прошлом занятии.

Далше идёт у нас адрес, просто разбитый по 8 бит. А полностью получается 0x00000400, то есть именно тот адрес в который мы и будем пытаться писать блок.

Проверим сначала в протее



Хоть у нас пока нет функции чтения с карты, но проверить мы можем, открыв наш файл образа карты памяти



Мы видим, что вся наша информация записана в правильное место.

Теперь поиграем с чтением. Создадим функцию опять же над функцией main() и сразу напишем в неё локальные переменные

```
unsigned char SD_Read_Block (char*
bf, unsigned char dt1, unsigned char
dt2, unsigned char dt3, unsigned
char dt4)
{
    unsigned char result;
    long int cnt;
}
```

Дальше всё скопируем с предыдущей функции записи и будем потихоньку вносить изменения

Команда будет у нас уже CMD17

CMD17	sdhc	[31:0] data address ¹	R1	READ_SINGLE_BLOCK	In the case of a Standard Capacity SD Memory Card, this command, this command reads a block of the size selected by the SET_BLOCKLEN command. In the case of a High Capacity Card, block length is fixed 512 bytes regardless of the SET_BLOCKLEN command.
-------	------	----------------------------------	----	-------------------	---



Соответственно, CMD17 у нас превращается в 0x51, остальные параметры аналогичные

```
long int cnt;
result=SD_cmd
(0x51,dt1,dt2,dt3,dt4,0x95); //CMD17
даташит стр 50 и 96
if (result!=0x00) return 5; //Выйти,
если результат не 0x00
```

Затем также передача 0xFF с целью выждать время и заодно прочистить регистр SPI у карты

```
if (result!=0x00) return 5; //Выйти,
если результат не 0x00
```

```
SPI_SendByte (0xFF);
```

Посмотрим, как читаются данные с карты

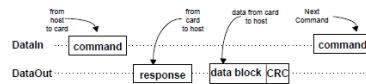


Figure 7-3: Single Block Read Operation

Мы видим, что никаких меток у нас нет, поэтому сразу принимаем ответ и проводим первичную его проверку

```
SPI_SendByte (0xFF);
cnt=0;
do{ //ждем начала блока
    result=SPI_ReceiveByte();
    cnt++;
} while ( (result!=0xFE)&&
(cnt<0xFFFF) );
if (cnt>=0xFFFF) return 5;
```

А вот теперь уже читаем данные

```
if (cnt>=0xFFFF) return 5;
for (cnt=0;cnt<512;cnt++)
    bf[cnt]=SPI_ReceiveByte(); //
получаем байты блока из шины в буфер
```

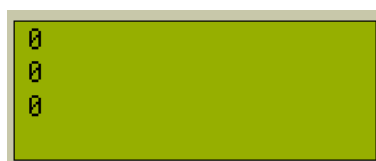
Затем мы обязаны принять контрольную сумму, поэтому примем её символически, никак не проверяя результат, даже не видя его, и возвращаем 0

```
for (cnt=0;cnt<512;cnt++)
    bf[cnt]=SPI_ReceiveByte(); //
получаем байты блока из шины в буфер
    SPI_ReceiveByte(); //Получаем
контрольную сумму
    SPI_ReceiveByte();
    return 0;
}
```

Теперь вызовем нашу функцию в main() и отобразим результат в третьей строке дисплея

```
str_lcd(str);
result=SD_Read_Block(buffer,0x00,0x0
0,0x04,0x00); //Считаем блок в буфер
sprintf(str,"%d",result);
setpos(0,2);
str_lcd(str);
```

Попробуем собрать код и запустить его в протее



Всё у нас считалось.

Теперь проверим в живой схеме, прошив перед этим контроллер

Яндекс.Директ



**Печатные
Платы
на Заказ.
Звоните!**

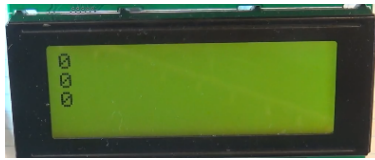
Изготовление
печатных плат на
заказ. От прототипов
до крупных партий.
Звони

Изготовление плат
Сборка и монтаж
Оборудование
Контакты

pcb.electropribor-
penza.ru
Адрес и телефон

Печатная плата - Тайвань

Изготовление печатных **плат**
и контрактное производство под
ключ в Тайване
explus.com.tw Адрес и телефон



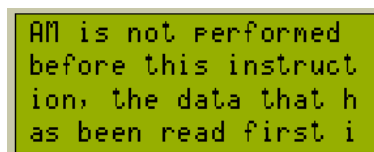
Только мы не видим, что именно у нас считалось. Если мы отобразим наш буфер, в который мы считали, то толку от этого не будет, так как там уже есть этот текст. Поэтому мы должны назначить буфер для чтения, так как наш буфер с текстом нам уже не подойдёт, ибо в нём уже есть текст и мы просто не проверим работу функции. Поэтому создадим ещё один буфер, а предыдущий закомментируем, так как у нас не хватит на два буфера оперативной памяти и компилятор даст ошибку и не будет собирать код

```
//char buffer[512] ="The..."; //Буфер
данных для записи/чтения
char buffer2[512] ={}; //Буфер
данных для чтения
```

И также закомментируем код вызова и отображения результата функции записи в main(), а также исправим имя буфера в вызове функции чтения, затем добавим задержку, раскомментируем функцию вывода буфера на экран дисплея, исправив там также имя буфера

```
//
result=SD_Write_Block(buffer,0x00,0x
00,0x04,0x00); //Запишем блок из
буфера
// sprintf(str,"%d",result);
// setpos(0,1);
// str_lcd(str);
result=SD_Read_Block(buffer2,0x00,0x
00,0x04,0x00); //Считаем блок в
буфер
sprintf(str,"%d",result);
setpos(0,2);
str_lcd(str);
_delay_ms(1000);
for (i=0;i<=22;i++)
{str80_lcd(buffer2+i*20);_delay_ms(1
000);}
while(1)
```

Сначала соберём код и проверим в протеусе



Затем прошьём контроллер и посмотрим результат вживую



Как видим, всё у нас работает.

Таким образом, мы научились работать по шине SPI с флеш-картой SD, научились туда писать блоки, читать блоки, что очень неплохо. А со **следующей части** нашего занятия мы начнём уже работать с определением типа карты, а также будем учиться работать с файловой системой на карте.

Предыдущая
часть

Программирование
МК AVR

Следующая
часть

Исходный код

**Техническая документация на
Secure Digital**

Программатор, модуль SD и дисплей можно приобрести здесь:

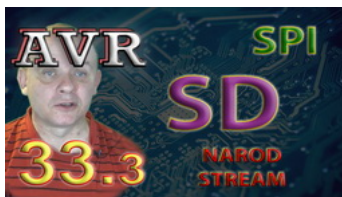
Программатор (продавец надёжный)

USBASP USBISP 2.0

Модуль карты SD SPI

Дисплей LCD 20×4

Смотреть ВИДЕОУРОК
(нажмите на картинку)



👁 Post Views: 311

◀ AVR Урок 33.

SPI. Карта SD.

2 комментария на "AVR Урок 33. SPI. Карта SD. FAT. Часть 3"

AVR Урок 33.



михаил: SPI. Карта SD.

Июнь 5, 2017 в 4:35 пп

Спасибо огромное за Ваши статьи. Неделю пытался проинициализировать SD карту аппаратным SPI, а тут в течение 3х часов (с перерывами) и все заработало! Удачи!

[Ответить](#)



михаил:

Июнь 13, 2017 в 2:06 пп

Уважаемый admin! Нет ли у Вас в планах разжевать для широких масс тему USB MS class именно с mega8 + SD карта? В разделе stm32 есть, а здесь пока нет.

[Ответить](#)

Добавить комментарий

Ваш e-mail не будет опубликован.


Обязательные поля помечены *

Комментарий

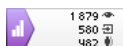
Имя *

E-mail *

Сайт

× 2 = **4** 

[Главная](#) | [Новости](#) | [Уроки по программированию МК](#)
| [Программирование микроконтроллеров AVR](#) | [Программирование микроконтроллеров STM32](#)
| [Программирование микроконтроллеров PIC](#) | [Тесты устройств и аксессуаров](#)
| [Устройства и интерфейсы](#) | [Ссылки](#) | [Форум](#) | [Помощь](#)



© 2018 Narod Stream

[Наверх](#)