



Свежие комментарии

- Narod Stream к записи AVR Урок 14. USART. Связь МК с ПК. Часть 3
- Андрей к записи AVR Урок 14. USART. Связь МК с ПК. Часть 3
- Narod Stream к записи STM Урок 16. HAL. ADC. Regular Channel
- Narod Stream к записи STM Урок 44. SDIO. FATFS
- maxvalin к записи STM Урок 16. HAL. ADC. Regular Channel

Форум. Последние ответы

- Narod Stream в Программирование МК STM32  
2 нед., 3 дн. назад
- Zandy в Программирование МК STM32  
2 нед., 3 дн. назад
- Narod Stream в Программирование МК STM32  
4 нед., 1 день назад
- Narod Stream в Программирование МК STM32  
4 нед., 1 день назад
- fireweb в Программирование МК STM32  
1 месяц назад

Февраль 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				
« Янв						

Архивы

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

Главная > I2C > AVR Урок 17. Часы реального времени DS1307. Часть 2

# AVR Урок 17. Часы реального времени DS1307. Часть 2

Posted on Декабрь 25, 2016 by Narod Stream  
Опубликовано в I2C, Программирование AVR — 2 комментария ↓

## Урок 17 Часть 2

# Часы реального времени DS1307

В предыдущей части занятия мы познакомились с отличной микросхемой реального времени **DS1307**, подключили её, и начали писать исходный код для установки времени, даты и дня недели.

Продолжим писать наш код, используя тот же самый проект

Используя написанную функцию перевода из десятичного формата числа в двоично-десятичный, занесём данные в регистры микросхемы, начиная с нулевого адреса (также мы знаем что это и адрес самого первого регистра микросхемы)

```
I2C_SendByte(0); //Переходим на 0x00
I2C_SendByte(RTC_ConvertFromBinDec(0)); //секунды
```

Мета

- Регистрация
- Войти
- RSS записей
- RSS комментариев
- WordPress.org

искать здесь ...

Фильтровать

Уроки по программированию МК

- Программирование МК AVR
- Программирование МК STM32
- Программирование МК PIC
- Тесты устройств и аксессуаров

## Заходите на канал Narod Stream



narod stream Просмотреть канал: Владелец

Главная Видео Плейлисты Каналы Обсуждение 0 канале

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

```
I2C_SendByte(RTC_ConvertFromBinDec(3
1)); //минуты
I2C_SendByte(RTC_ConvertFromBinDec(2
0)); //часы
I2C_SendByte(RTC_ConvertFromBinDec(5
)); //день недели
I2C_SendByte(RTC_ConvertFromBinDec(2
9)); //дата
I2C_SendByte(RTC_ConvertFromBinDec(1
)); //месяц
I2C_SendByte(RTC_ConvertFromBinDec(1
6)); //год
```

Ну, у вас конечно будут другие данные, в зависимости от того момента времени, в который вы будете запускать данную программу.

Ну, и в конце, конечно условие STOP

```
I2C_SendByte(RTC_ConvertFromBinDec(1
6)); //год
I2C_StopCondition();
while(1)
```

Соберем код, установим в скобки правильное время и прошьём контроллер.

Пока мы никак не можем проверить, как это всё записалось и ходят ли часы. Для этого нужно написать код для чтения.

Можно конечно было бы посмотреть статусы, но раз уж нам всё равно читать показания регистров, то особого смысла в этом не вижу.

Пока данный код записи в регистры полностью закомментируем. Мы будем его раскомментировать тогда, когда будем программировать новую микросхему и заносить данные в регистры, каждый раз это делать на нужно.

Теперь в бесконечный цикл начнем писать код считывания данных из регистров

Точно также всё начинается с условия СТАРТ.

Но для того, чтобы нам всю эту рутину каждый раз не писать, давайте напомним функцию для передачи по адресу устройства байта адреса памяти

```
void I2C_SendByteByADDR(unsigned
char c,unsigned char addr)
{
    I2C_StartCondition(); // Отправим
    условие START
    I2C_SendByte(addr); // Отправим в
    шину адрес устройства + бит чтения-
    записи
    I2C_SendByte(c);// Отправим байт
    данных
    I2C_StopCondition();// Отправим
    условие STOP
}
```

Также заодно напомним функцию чтения обычного байта из шины и чтения последнего байта из шины. У нас такие функции уже были, но они были в особенном файле и были с префиксом EE\_, а также там была обработка ошибки. Скопируем их себе теперь в

JavaScript Diagrams

Click Here for GoJS

Build Interactive JavaScript Diagrams

### Рубрики

- 1-WIRE (3)
- ADC (6)
- DAC (4)
- GPIO (26)
- I2C (19)
- SPI (13)
- USART (8)
- Программирование AVR (131)
- Программирование PIC (7)
- Программирование STM32 (216)
- Тесты устройств и аксессуаров (1)

7

31 ДЕНЬ	136	109
	14	139
07 ДНЕЙ	32	840
	4	422
24 ЧАСА	5	059
	1	141
СЕГОДНЯ	2	956
		129
НАПЛИШУ	143	29

стандартный обычный TWI.c, убрав префиксы и всё лишнее

```

unsigned char I2C_ReadByte(void)
{
    TWCR = (1<<TWINT)|(1<<TWEN)|
    (1<<TWEA);
    while (!(TWCR & (1<<TWINT)));//
    ожидание установки бита TWIN
    return TWRD;//читаем регистр
    данных
}

unsigned char I2C_ReadLastByte(void)
{
    TWCR = (1<<TWINT)|(1<<TWEN);
    while (!(TWCR & (1<<TWINT)));//
    ожидание установки бита TWIN
    return TWRD;//читаем регистр
    данных
}

```

Напишем в TWI.h на все эти функции прототипы

```

void I2C_SendByte(unsigned char c);
//передача байта в шину
void I2C_SendByteByADDR(unsigned
char c,unsigned char addr); //
передача байта в шину на устройство
по адресу
unsigned char I2C_ReadByte(void); //
читаем байт
unsigned char
I2C_ReadLastByte(void); //читаем
последний байт

```

И теперь можно начинать писать код в бесконечный цикл функции main().

Отправим адрес 0 (адрес первого регистра) по адресу устройства

```

while(1)
{
    //Читаем время
    I2C_SendByteByADDR(0,0b11010000);
    //переходим на адрес 0

```

Затем вставим задержку на 300 миллисекунд. Данную задержку можно и в конце кода вставить, ну давайте попробуем здесь, поэкспериментируем, так сказать

```

I2C_SendByteByADDR(0,0b11010000); //
переходим на адрес 0
_delay_ms(300);

```

Затем посылаем в шину адрес устройства с битом чтения, отправив перед этим условие СТАРТ

```

_delay_ms(300);
I2C_StartCondition(); //Отправим
условие START
I2C_SendByte(0b11010001); //отправим
в устройство бит чтения

```

По идее мы вообще так не должны делать, так как у нас в функции, которую мы вызвали перед задержкой было в конце условие СТОП, а это не требуется. Обычно сразу СТАРТ. Но можно и так, всё работает. Если указатель установлен

уже туда, куда надо, то можно сразу адрес чтения и начинать читать, а не так, как было в случае с EEPROM.

---

Дальше читаем все регистры

```
I2C_SendByte(0b11010001); //
отправим в устройство бит
чтения
sec = I2C_ReadByte();
min = I2C_ReadByte();
hour = I2C_ReadByte();
day = I2C_ReadByte();
date = I2C_ReadByte();
month = I2C_ReadByte();
year = I2C_ReadLastByte();
```

Помним, что последний байт читается из шины без подтверждения и для этого у нас есть соответствующая функция.

В конце чтения отправим в шину условие СТОП

```
year = I2C_ReadLastByte();
I2C_StopCondition(); //Отправим
условие STOP
```

Далее, используя функцию преобразования из двоично-десятичного формата в десятичный, преобразуем считанные показания, так как мы их забрали из регистров именно в двоично-десятичном виде.

```
I2C_StopCondition(); //Отправим
условие STOP
sec = RTC_ConvertFromDec(sec); //
Преобразуем в десятичный формат
min = RTC_ConvertFromDec(min); //
Преобразуем в десятичный формат
hour = RTC_ConvertFromDec(hour); //
Преобразуем в десятичный формат
day = RTC_ConvertFromDec(day); //
Преобразуем в десятичный формат
year = RTC_ConvertFromDec(year); //
Преобразуем в десятичный формат
month = RTC_ConvertFromDec(month);
//Преобразуем в десятичный формат
date = RTC_ConvertFromDec(date); //
Преобразуем в десятичный формат
```

И в конце бесконечного цикла отправим всё это в определённом виде в USART

```
date = RTC_ConvertFromDec(date); //
Преобразуем в десятичный формат
USART_Transmit(date/10+0x30); //
Преобразуем число в код числа
USART_Transmit(date%10+0x30); //
Преобразуем число в код числа
USART_Transmit('.');
USART_Transmit(month/10+0x30); //
Преобразуем число в код числа
USART_Transmit(month%10+0x30); //
Преобразуем число в код числа
USART_Transmit('.');
USART_Transmit(year/10+0x30); //
Преобразуем число в код числа
```

```

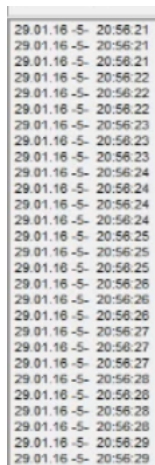
USART_Transmit(year%10+0x30);//
Преобразуем число в код числа
USART_Transmit(' ');
USART_Transmit('-');
USART_Transmit(day+0x30);//
Преобразуем число в код числа
USART_Transmit('-');
USART_Transmit(' ');
USART_Transmit(' ');
USART_Transmit(hour/10+0x30);//
Преобразуем число в код числа
USART_Transmit(hour%10+0x30);//
Преобразуем число в код числа
USART_Transmit(':');
USART_Transmit(min/10+0x30);//
Преобразуем число в код числа
USART_Transmit(min%10+0x30);//
Преобразуем число в код числа
USART_Transmit(':');
USART_Transmit(sec/10+0x30);//
Преобразуем число в код числа
USART_Transmit(sec%10+0x30);//
Преобразуем число в код числа
USART_Transmit(0x0d); //переход в
начало строки
USART_Transmit(0x0a); //перевод
каретки
}

```

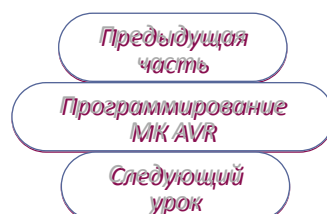
Смещение на 0x30 в вычисление кода символа — это преобразование самой цифры в код цифры. Именно такая разница и есть в таблице **ascii**.

Можно конечно не париться с таким преобразованием и использовать функцию **sprintf** и она прекрасно с этим справится, но так интересно, функция **sprintf** ещё себя покажет, это я уж вам обещаю точно.

Соберём код, откроем терминальную программу, нажмём там **Connect**, Прошьём контроллер и посмотрим результат



Наши часы отлично ходят.



**Исходный код**

### Документация на микросхему DS1307

Программатор, модуль RTC DS1307 с микросхемой памяти и переходник USB-TTL можно приобрести здесь:

Программатор (продавец надёжный)  
[USBASP USBISP 2.0](#)

[Модуль RTC DS1307 с микросхемой памяти](#)

[Переходник USB-TTL лучше купить такой \(сейчас у меня именно такой и он мне больше нравится\)](#)

### Смотреть ВИДЕОУРОК (нажмите на картинку)



Post Views: 614

< AVR Урок 17.

Часы реального

2 комментария на “AVR Урок 17. Часы реального времени DS1307. Часть 2”



Vlad:

Январь 23, 2017 в 10:40

Превосходно. Особенно помогли ошибки, которые Вы в ходе урока разбираете и устраняете. Мне это помогло в поиске своих ошибок гораздо больше чем многотомники по C++. Благодарю Вас за прекрасные уроки.

[Ответить](#)



admin:

Январь 24, 2017 в 4:15 дп

Вам также спасибо за внимание к моим занятиям!

[Ответить](#)

### Добавить комментарий

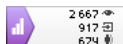
Ваш e-mail не будет опубликован.

Обязательные поля помечены \*

Комментарий

Имя \*

E-mail \*

**Сайт**  
восемь + 5 = [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