



Главная

Новости

Уроки по программированию МК

Устройства и интерфейсы

Ссылки

Ф

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 1

AVR Урок 17. Часы реального времени DS1307. Часть 1

Posted on Декабрь 25, 2016 by Narod Stream
Опубликовано в I2C, Программирование AVR — 1 комментарий ↓



Урок 17 Часть 1


Часы реального времени DS1307

Продолжаем занятия по программированию **МК AVR**.
И сегодня мы познакомимся с очень хорошей микросхемой **DS1307**. Данная микросхема представляет собой **часы реального времени (real time clock** или **RTC**).
Также, благодаря тому, что общение микроконтроллера с данной микросхемой будет происходить с применением интерфейса **I2C**, мы ещё лишний раз на деле закрепим тему программирования данной шины.
Данная микросхема представлена компанией **Dallas**, вот её распиновка и основные технические характеристики

Яндекс.Директ



Программируемые логические модули
Сравните лучшие ПЛК! Тех.поддержка, скидки, уникальный фильтр и гарантия! ipc2u.ru
Адрес и телефон



Сильный мороз в Минской области?
Смотрите прогноз погоды на декабрь. yandex.by

Заходите на канал
Narod Stream

- 2/7

BIT7																				BIT0	
00H	CH		10 SECONDS						SECONDS						00-59						
	0		10 MINUTES						MINUTES						00-59						
	0		12		24		10 HR AP		10 HR		HOURS						01-12 00-23				
	0		0		0		0		0		DAY						1-7				
	0		0		10 DATE						DATE						01-28/29 01-31				
	0		0		0		10 MONTH		MONTH						01-12						
	10 YEAR										YEAR						00-99				
07H	OUT		0		0		SQWE		0		0		RS1		RS0						

Назначение данных регистров:

00h — секунды. Секунды хранятся в двоично-десятичном виде. То есть в младших 4 битах хранятся единицы секунд, а в более старших трёх — десятки. Также есть бит SH — это бит запуска микросхемы.

01h — минуты. Хранятся аналогично.

02h — более универсальный регистр. Здесь хранятся часы. В четырех младших битах — единицы чаов, в следующих более старших двух — десятки, в следующем 6 бите — флаг того, после полудня сейчас время или до полудня, в 7 бите — режим хранения — 12- часовой или 24-часовой.

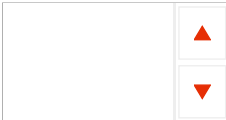
03h — день недели. Хранится в младших 3 битах, остальные биты не используются.

04h — здесь хранится день месяца, также в двоично-десятичном формате. В четырёх малдших битах — единицы, в двух следующих постарше — десятки, остальные биты не используются.

05h — номер месяца в году — хранится в двоично-десятичном формате точно также, как и часы.

06h — номер года, причём не полный четырёхзначный, а только двузначный. В младших четырех битах — единицы, в старших — десятки.

Вот этими семью регистрами мы и будем пользоваться. Последний регистр предназначен для конфигурирования частоты импульсов на импульсном выходе микросхемы, это делается в младших двух битах регистра. по умолчанию он будет 1 гц частотой, нам этого достаточно, чтобы помигать двоеточием, поэтому мы не будем пользоваться данными битами. Биты SOWE и OUT также применяются для настройки и включения формирователя даннх квадратных импульсов.



Проект для работы с дан микросхемой был создан обычн образом с именем **MyClock1307**, фай связанные с EEPROM оттуда убраны, а добавлены файлы **RTC.c** и **RTC.h**.

Содержание файла main.h у нас теперь вот такое

```
#ifndef MAIN_H_
#define MAIN_H_
#define F_CPU 8000000UL
```



Яндекс.Директ



Нужно программир-е контроллеров?

Комплексное обучение. Доступные цены. Минимальные сроки. Большой опыт!
[О компании](#)
[Услуги](#)
[Продукция](#)
[Преимущества](#)
[festo.com](#)
[Адрес и телефон](#)



Программируемые логические модули

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

```
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>
#include <stdio.h>
#include <stdlib.h>
#include "usart.h"
#include "twi.h"
#include "RTC.h"
#endif /* MAIN_H_ */
```

В главном файле **MyClock1307.c** создадим глобальные переменные для хранения показаний времени, даты и дня недели и после этого полное содержание после удаления всего лишнего в нём будет вот таким

```
#include "main.h"
unsigned char
sec,min,hour,day,date,month,year;
int main(void)
{
    I2C_Init();
    USART_Init (8);
    while(1)
    {
    }
}
```

От прошлого кода останется лишь инициализация I2C и USART.

Теперь нам надо как-то вообще запустить микросхему. Если микросхема новая, либо никогда не использовалась, либо кто-то специально для каких-то целей изменил значение бита CH, то она ещё не "ходит".

Ну, вообще, как только мы установим все значения в регистрах микросхемы, так она и запустится и наши часы пойдут.

Подключение или схема использована также вся из прошлого занятия, то есть время смотреть мы будем посредством шины USART в терминальной программе.

Поэтому, собственно, используя наши знания предыдущего занятия, напишем писать функцию установки времени.

Первым делом мы, само собой, передадим условие СТАРТ

```
//Устанавливаем время
I2C_StartCondition();
```

Затем передаём адрес с битом записи 0

```
I2C_StartCondition();
I2C_SendByte(0b11010000);
```

Перейдём на адрес 0, а значит к той части памяти, где расположен самый первый регистр

```
I2C_SendByte(0b11010000);
I2C_SendByte(0); //Переходим на 0x00
```

Прежде чем писать какие-то значения в регистры микросхемы, мы вспомним, что числа мы сначала должны преобразовать в двоично-десятичный формат, который будет удобен для регистров. Для этого мы зайдём в файл

RTC.c и такую функцию и напишем. Она будет очень лёгкой и в объяснении не нуждается

```
unsigned char
RTC_ConvertFromBinDec(unsigned char
c)
{
    unsigned char ch = ((c/10)<<4)|
(c%10);
    return ch;
}
```

Ну и также давайте напишем и функцию обратного типа, переводящую число из двоично-десятичного формата в десятичный. С помощью неё мы, наоборот, будем считанные показания времени преобразовывать в вид, удобный нашему восприятию (ЧПИ — человеко-понятный интерфейс)

```
unsigned char
RTC_ConvertFromDec(unsigned char c)
{
    unsigned char ch = ((c>>4)*10+
(0b00001111&c));
    return ch;
}
```

Здесь также всё придельно ясно, мы сдвигаем вправо старшую тетраду байта, умножаем её на десять и прибавляем младшую тетраду (старшую отмаскировываем нулями)

Напишем прототипы данных функций в файле **RTC.c**

```
#include "main.h"
unsigned char
RTC_ConvertFromDec(unsigned char c);
//перевод двоично-десятичного числа
в десятичное
unsigned char
RTC_ConvertFromBinDec(unsigned char
c); //перевод десятичного числа в
двоично-десятичное
```

Соберём код, а прошивать контроллер пока не будем. Нам нужно ещё дописать код записи в регистры и написать в бесконечный цикл процедуру чтения времени и даты и отправку всего этого в USART, а затем уж прошьём полностью весь код, прописав правильные значения времени и даты в установку времени.

Сделаем мы всё это в **следующей части** занятия.

*Предыдущий
урок*

*Программирование
МК AVR*

*Следующая
часть*

**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: 711

STM Урок 40.

Знакомство с

Один комментарий на «AVR Урок 17. Часы реального времени DS1307. Часть 1»



Melamed: Часы реального времени DS1307.
Октябрь 4, 2017 в 11:39 дп

Проанализировав ваши функции `RTC_ConvertFromBinDec` и `RTC_ConvertFromDec`, пришел к выводу, что время и дата в микросхеме DS1307 хранятся в следующем формате: в младших четырех битах хранится десятичные единицы, а старших 4 битах — десятичные десятки. Поэтому достаточно для выделения десятичных единиц выполнить операцию побитного и с числом `0x0F`, а десятков с помощью побитного сдвига на 4 бита

```
ed = min & 0x0f;  
dec = min>>4;
```

И у меня это работает

[Ответить](#)

Добавить комментарий

Ваш e-mail не будет опубликован.
Обязательные поля помечены *

Комментарий

Имя *

E-mail *

Сайт




шесть -  = 




Отправить комментарий

- 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



2 302 
825 
625 

© 2018

Наверх