Real Time Clock(DS1307) with AVR

Contents

(Redirected from A6.AVR Interfacing: RTC DS1307 (/wiki/index.php?title=A6.AVR_Interfacing:_RTC_DS1307&redirect=no))

Basics

The Real time clock DS1307 IC basically is stand alone time clock. Well, basically we can use a micrcontroller to keep time, but the value would go off as soon as it is powered off.

The RTC DS1307 is a handy solution to keep time all the way, when it is powered by a coin cell.

It is uses **I²C** (**Inter-Integrated Circuit**) protocol, referred to as *I-squared-C*, *I-two-C*, or *IIC* for communication with the micrcontroller.

Check the Basics of I2C (/wiki/Serial_protocol_I2C) here, if you are not familiar with it. For details of I2C in AVR, go through AVR Communication Protocols (/wiki/A5.AVR_Communication_Protocols:I2C,_SPI) tutorial.

The first thing that the MCU sends to the slave (RTC) is the device ID. The device ID for DS1307, shown below. It also tells weather we want to write to or read from the RTC.

7	6	5	4	3	2	1	0
1	1	0	1	0	0	0	R/W

- bit-0 is **0** than we **Write** to RTC
- bit-0 is **1** we **Read** from RTC.

This is defined in the code as:

```
1.  #define C_Ds1307ReadMode_U8   0xD1u  // DS1307 ID
2.  #define C_Ds1307WriteMode_U8  0xD0u  // DS1307 ID
```

The RTC keeps the date and time arranged in it's memory as shown below:

ADDRESS	FUNCTION	RANGE
00h	Seconds	00–59
01h	Minutes	00–59
02h	Hours	01-12/00-24
03h	Day	01–07
04h	Date	01–31
05h	Month	01–12
06h	Year	00–99
07h	Control	
08h to 3Fh	RAM	00h–FFh

Write to the addresses above we can set the time, and once we set it, we can read it any time we need.

The address 0x07 is a control registered as described below:

7	6	5	4	3	2	1	0
OUT	0	0	SQWE	0	0	RS1	RS0

We write 0x00 to Control register to disable SQW-Out. We do not use any other bits from it, so you need not worry.

Initialize

Now we can initialize the RTC with the code below

```
1.  void RTC_Init(void)
2.  {
3.      I2C_Init();                             // Initialize the I2c module.
4.      I2C_Start();                            // Start I2C communication
5.
6.      I2C_Write(C_Ds1307WriteMode_U8);        // Connect to DS1307 by sending its ID on I2c Bus
7.      I2C_Write(C_Ds1307ControlRegAddress_U8);// Select the Ds1307 ControlRegister to configure Ds
8.
9.      I2C_Write(0x00);                        // Write 0x00 to Control register to disable SQW-Out
10.
11.     I2C_Stop();                             // Stop I2C communication after initializing DS1307
12. }
```

Set Date and Time

```
1.  void RTC_SetDateTime(rtc_t *rtc)
2.  {
3.      I2C_Start();                          // Start I2C communication
4.
5.      I2C_Write(C_Ds1307WriteMode_U8);      // connect to DS1307 by sending its ID on I2c Bus
6.      I2C_Write(C_Ds1307SecondRegAddress_U8); // Request sec RAM address at 00H
7.
8.      I2C_Write(rtc->sec);                  // Write sec from RAM address 00H
9.      I2C_Write(rtc->min);                  // Write min from RAM address 01H
10.     I2C_Write(rtc->hour);                  // Write hour from RAM address 02H
11.     I2C_Write(rtc->weekDay);              // Write weekDay on RAM address 03H
12.     I2C_Write(rtc->date);                  // Write date on RAM address 04H
13.     I2C_Write(rtc->month);                  // Write month on RAM address 05H
14.     I2C_Write(rtc->year);                  // Write year on RAM address 06h
15.
16.     I2C_Stop();                             // Stop I2C communication after Setting the Date
17. }
```

Note: The date and time read from Ds1307 will be of BCD format, like:

- 0x12,0x39,0x26 for 12hr,39min and 26sec.
- 0x15,0x08,0x47 for 15th day,8th month and 47th year

Get Date and Time

```
void RTC_GetDateTime(rtc_t *rtc)
{
    I2C_Start();                            // Start I2C communication

    I2C_Write(C_Ds1307WriteMode_U8);        // connect to DS1307 by sending its ID on I2c Bus
    I2C_Write(C_Ds1307SecondRegAddress_U8); // Request Sec RAM address at 00H

    I2C_Stop();                                // Stop I2C communication after selecting Sec Reg

    I2C_Start();                            // Start I2C communication
    I2C_Write(C_Ds1307ReadMode_U8);            // connect to DS1307(Read mode) by sending its ID

    rtc->sec = I2C_Read(1);                // read second and return Positive ACK
    rtc->min = I2C_Read(1);                 // read minute and return Positive ACK
    rtc->hour= I2C_Read(1);               // read hour and return Negative/No ACK
    rtc->weekDay = I2C_Read(1);           // read weekDay and return Positive ACK
    rtc->date= I2C_Read(1);              // read Date and return Positive ACK
    rtc->month=I2C_Read(1);            // read Month and return Positive ACK
    rtc->year =I2C_Read(0);             // read Year and return Negative/No ACK

    I2C_Stop();                             // Stop I2C communication after reading the Date
}
```

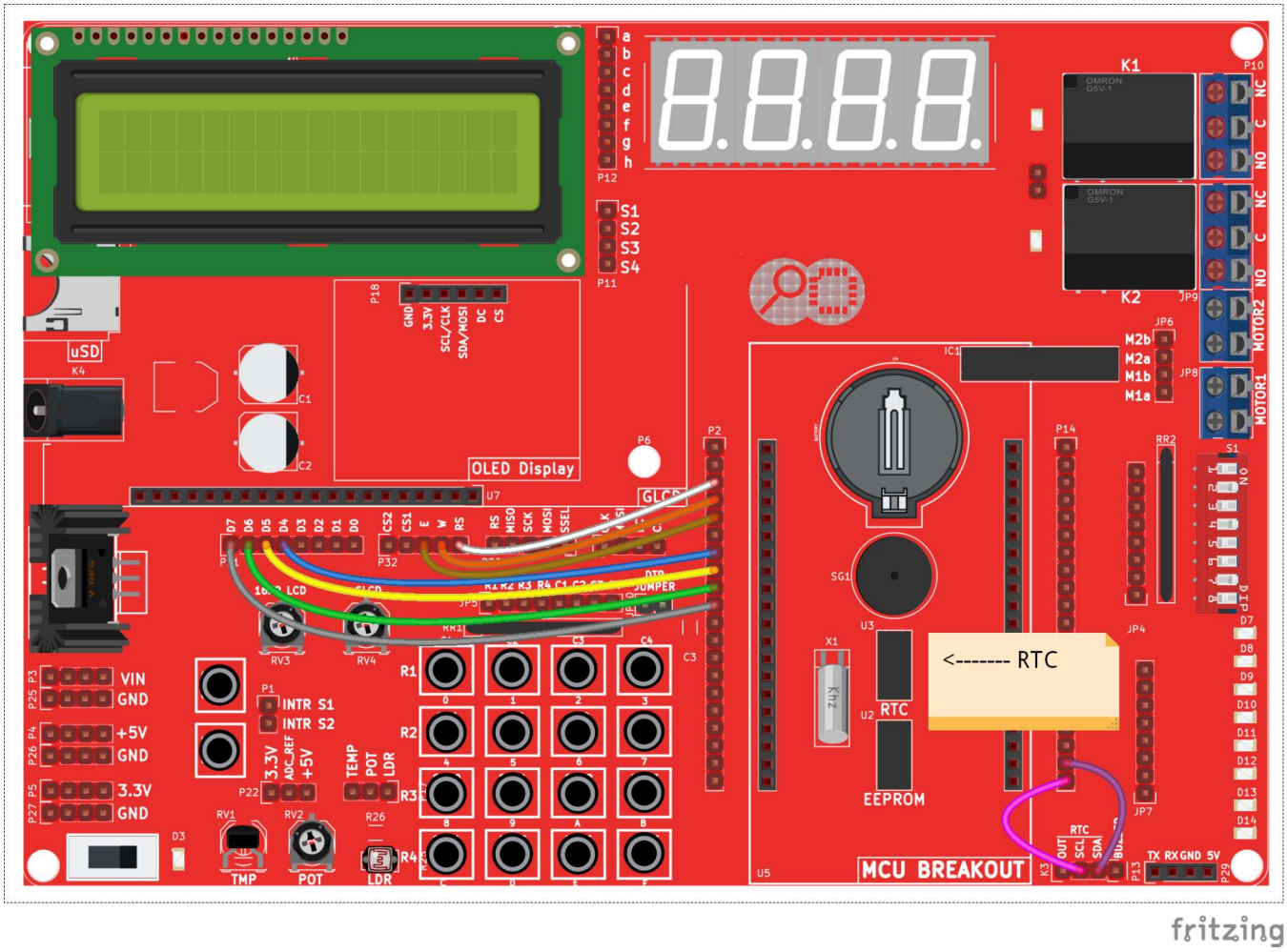
Both the above functions use a simple structure shown below for easy access

```
typedef struct
{
  uint8_t sec;
  uint8_t min;
  uint8_t hour;
  uint8_t weekDay;
  uint8_t date;
  uint8_t month;
  uint8_t year;
}rtc_t;
```

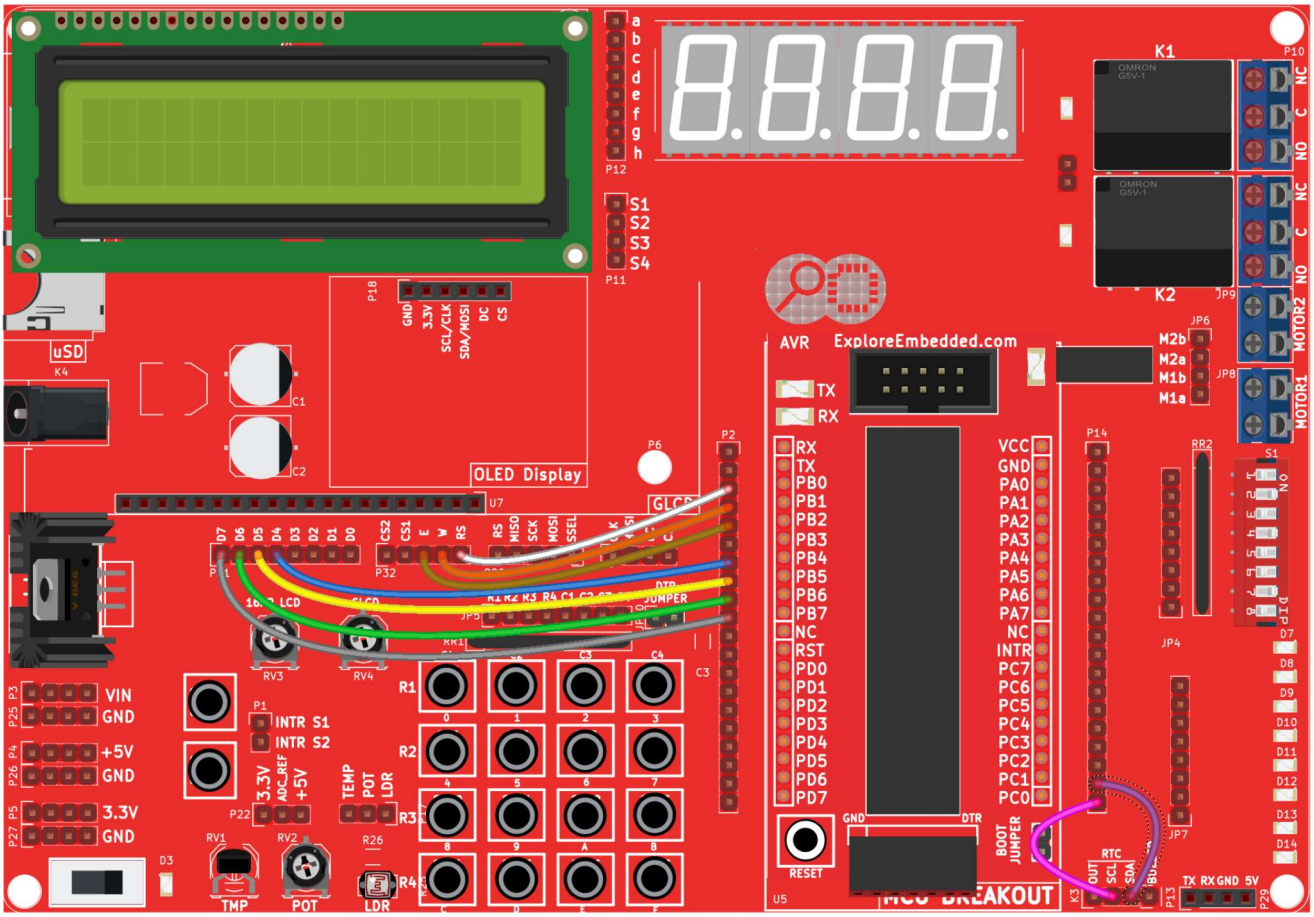
Example

Now, let's put together the all that we have discussed in a simple example to read and show the time on character LCD.

Hookup



(/wiki/File:Real_Time_Clock(DS1307)_with_AVR_LCD_RTC.png)



(/wiki/File:Real_Time_Clock(DS1307)_with_AVR_LCD_bb.png)

Code

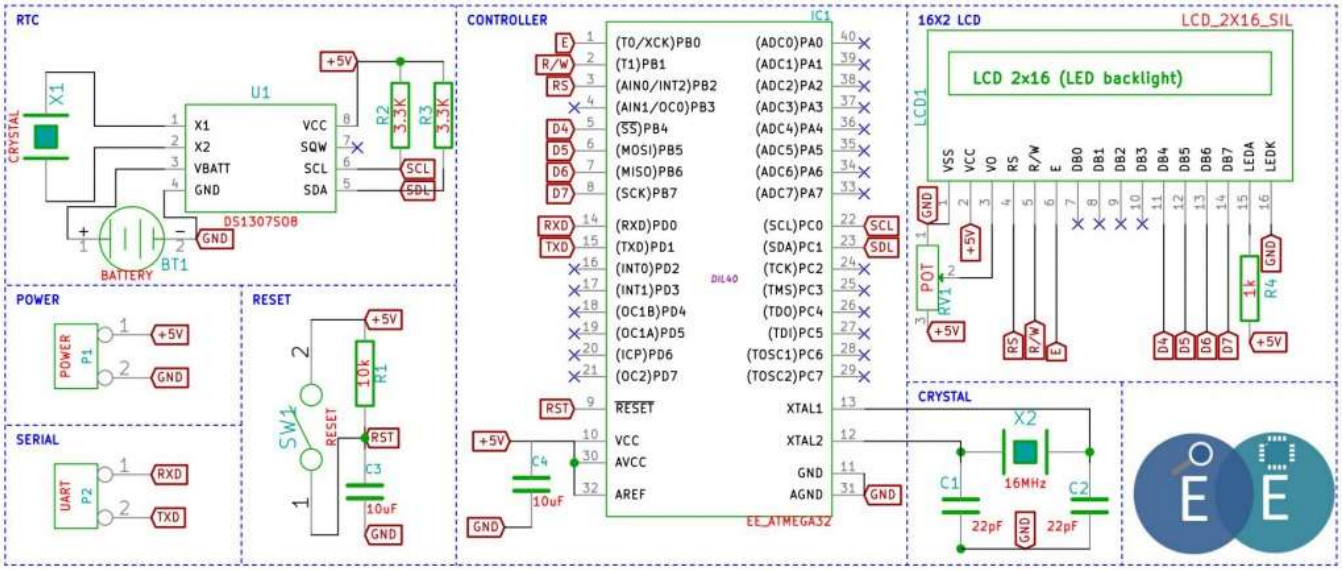
```
#include "rtc.h"
#include "lcd.h"
int main()
{
    rtc_t rtc;

    /*Connect RS->PB0, RW->PB1, EN->PB2 and data bus to PORTB.4 to PORTB.7*/
    LCD_SetUp(PB_0,PB_1,PB_2,P_NC,P_NC,P_NC,P_NC,PB_4,PB_5,PB_6,PB_7);
    LCD_Init(2,16);

    /*Connect SCL->PC0, SDA->PC1*/
    RTC_Init();
    rtc.hour = 0x10; //  10:40:20 am
    rtc.min =  0x40;
    rtc.sec =  0x00;

```

```
18      rtc.date = 0x01; //1st Jan 2016
19      rtc.month = 0x01;
20      rtc.year = 0x16;
21      rtc.weekDay = 5; // Friday: 5th day of week considering monday as first day.
22      /*##### Set the time and Date only once. Once the Time and Date is set, comment these lines
23           and reflash the code. Else the time will be set every time the controller is reset*/
24      RTC_SetDateTime(&rtc);  //  10:40:20 am, 1st Jan 2016
25      /* Display the Time and Date continuously */
26      while(1)
27      {
28          RTC_GetDateTime(&rtc);
29          LCD_GoToLine(0);
30          LCD_Printf("time:%2x:%2x:%2x  \nDate:%2x/%2x/%2x",(uint16_t)rtc.hour,(uint16_t)rtc.min
31      }
32      return (0);
33  }
```

m/Xplorer001/dabc20ebd54d261f07f0/raw/329c35ef33b0537fcf6c8e40c58aecc0887d279e/Atmega32_DS1307RTC_main.c) Atmega32_DS1307RTC_main.c (https://gist.github.com/Xplorer001/dabc20ebd54d261f07f0#file-atmega32_ds1307rtc_main-c) hosted with ❤ by GitHub (https://github.com)

Schematic

[1] (http://exploreembedded.com/wiki/images/4/43/Schematic_AVR_Interfacing_RTC.pdf)



(/wiki/File:Schematic_AVR_Interfacing_RTC.JPG)

(/wiki/File:Real_Time_Clock(DS1307)_with_AVR.gif)

Video Tutorial

For those of you, who would like to watch instead of read we have made a video with all the gyan.

Downloads

Download the complete project folder from the below link:
https://github.com/ExploreEmbedded/ATmega32_ExploreUltraAvrDevKit/archive/master.zip
(https://github.com/ExploreEmbedded/ATmega32_ExploreUltraAvrDevKit/archive/master.zip)

Have a opinion, suggestion , question or feedback about the article let it out here!

ALSO ON EXPLOREEMBEDDED.COM/WIKI

6 years ago • 2 comments
Arduino ...

4 years ago • 1 comment
Interfacing ...

0 Comments

Start the discussion...

LOG IN WITH

OR SIGN UP WITH DISQUS

Name

Sort by Best 5

Be the first to comment.

Category (/wiki/Special:Categories): AVR Tutorials (/wiki/Category:AVR_Tutorials)

Subscribe to hear about our latest Explorations!

name@example.com SUBSCRIBE

 (https://twitter.com/exploreembedded)  (https://www.facebook.com/ExploreEmbedded/)

 (https://www.youtube.com/channel/UCvXGpvPuosEI-ALxvCrSbaA)  (https://github.com/ExploreEmbedded)

 PayPal

Now shipping worldwide from India with ♥