

DS1302 - Arduino library support for the DS1302 Trickle Charge Timekeeping Chip

Copyright (C)2010 Henning Karlsen. All right reserved

You can find the latest version of the library at <http://www.henningkarlsen.com/electronics>

This library has been made to easily interface and use the DS1302 RTC with the Arduino.

If you make any modifications or improvements to the code, I would appreciate that you share the code with me so that I might include it in the next release. I can be contacted through <http://www.henningkarlsen.com/electronics/contact.php>

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA

Version:	1.0	6 Aug 2010	initial release
	2.0	23 Aug 2010	Added functions to use on-chip RAM
	2.1	17 Nov 2010	Added <code>setTCR()</code> ;
	2.2	26 Jan 2012	Added support for Arduino 1.0 IDE

Structures:

Time;	
Structure to manipulate time- and date-data.	
Variables:	hour, min, sec: For holding time-data date, mon, year: For holding date-data dow: Day-of-the-week with monday being the first day
Usage:	Time t; // Define a structure named t of the Time-class

DS1302_RAM;	
Buffer for use with readBuffer() and writeBuffer().	
Variables:	Cell[0-30]: Array of 31 bytes to hold the data read from or to be written to the on-chip RAM.
Usage:	DS1302_RAM ramBuffer; // Declare a buffer for use

Defined Literals:

Weekdays	
For use with setDOW() and Time.dow	
	MONDAY: 1 TUESDAY: 2 WEDNESDAY: 3 THURSDAY: 4 FRIDAY: 5 SATURDAY: 6 SUNDAY: 7

Select length	
For use with getTimeStr(), getDateStr(), getDOWStr() and getMonthStr()	
	FORMAT_SHORT: 1 FORMAT_LONG: 2

Select date format	
For use with getDateStr()	
	FORMAT_LITTLEENDIAN: 1 FORMAT_BIGENDIAN: 2 FORMAT_MIDDLEENDIAN: 3

Select Trickle-Charge values	
For use with setTCR()	
	TCR_D1R2K: 165 TCR_D1R4K: 166 TCR_D1R8K: 167 TCR_D2R2K: 169 TCR_D2R4K: 170 TCR_D2R8K: 171 TCR_OFF: 92

Functions:

DS1302(CE, IO, SCLK);	
The main class of the interface.	
Parameters:	CE: CE-pin of the DS1302 (Pin 5) IO: I/O-pin of the DS1302 (Pin 6) SCLK: SCLK-pin of the DS1302 (Pin 7)
Usage:	DS1302 rtc(2, 3, 4); // Start an instance of the DS1302 class

getTime();	
Get current data from the DS1302.	
Parameters:	None
Returns:	Time-structure
Usage:	t = rtc.getTime(); // Read current time and date.

setTime(hour, min, sec);	
Set the time.	
Parameters:	hour: Hour to store in the DS1302 (0-23) min: Minute to store in the DS1302 (0-59) sec: Second to store in the DS1302 (0-59)
Returns:	Nothing
Usage:	rtc.setTime(23, 59, 59); // Set the time to 23:59:59
Notes:	Setting the time will clear the CH (Clock Halt) flag. See the datasheet for more information on the CH flag.

setDate(date, mon, year);	
Set the date.	
Parameters:	date: Date of the month to store in the DS1302 (1-31) *1 mon: Month to store in the DS1302 (1-12) year: Year to store in the DS1302 (2000-2099)
Returns:	Nothing
Usage:	rtc.setDate(6, 8, 2010); // Set the date to August 6., 2010.
Notes:	*1: No cheking for illegal dates so Feb 31. is possible to input. <i>The effect of doing this is unknown.</i>

setDOW(dow);	
Set the day-of-the-week.	
Parameters:	dow: Day of the week to store in the DS1302 (1-7) *1
Returns:	Nothing
Usage:	rtc.setDOW(FRIDAY); // Set the day-of-the-week to be friday
Notes:	*1: Monday is 1, and through to sunday being 7.

getTimeStr([format]);	
Get current time as a string.	
Parameters:	format: <Optional> FORMAT_LONG "hh:mm:ss" (default) FORMAT_SHORT "hh:mm"
Returns:	String containing the current time with or without seconds.
Usage:	Serial.print(rtc.getTimeStr()); // Send the current time over a serial connection

getDateStr([sformat[, eformat[, divider]]]);	
Get current date as a string.	
Parameters:	sformat: <Optional> *1 FORMAT_LONG Year with 4 digits (yyyy) (default) FORMAT_SHORT Year with 2 digits (yy) eformat: <Optional> *2 FORMAT_LITTLEENDIAN "dd.mm.yyyy" (default) FORMAT_BIGENDIAN "yyyy.mm.dd" FORMAT_MIDDLEENDIAN "mm.dd.yyyy" divider: <Optional> Single character to use as divider. Default is '.'
Returns:	String containing the current date in the specified format.
Usage:	Serial.print(rtc.getDateStr()); // Send the current date over a serial connection (in Little-Endian format)
Notes:	*1: Required if you need eformat or divider. *2: Required if you need divider. More information on Wikipedia (http://en.wikipedia.org/wiki/Date_format#Date_format).

getDOWStr([format]);	
Get current day-of-the-week as a string.	
Parameters:	format: <Optional> FORMAT_LONG Day-of-the-week in English (default) FORMAT_SHORT Abbreviated Day-of-the-week in English (3 letters)
Returns:	String containing the current day-of-the-week in full or abbreviated format.
Usage:	Serial.print(rtc.getDOWStr(FORMAT_SHORT)); // Send the current day in abbreviated format over a serial connection
getMonthStr([format]);	
Get current month as a string.	
Parameters:	format: <Optional> FORMAT_LONG Month in English (default) FORMAT_SHORT Abbreviated month in English (3 letters)
Returns:	String containing the current month in full or abbreviated format.
Usage:	Serial.print(rtc.getMonthStr()); // Send the current month over a serial connection
halt(value);	
Set or clear the CH ^{*1} flag.	
Parameters:	value: true: Set the CH flag false: Clear the CH flag
Returns:	Nothing
Usage:	rtc.halt(true); // Set the CH flag
Notes:	*1: CH: Clock Halt flag. See the datasheet for more information.
writeProtect(enable);	
Set or clear the WP ^{*1} bit.	
Parameters:	enable: true: Set the WP bit false: Clear the WP bit
Returns:	Nothing
Usage:	rtc.writeProtect(false); // Clear the WP bit
Notes:	*1: WP: Write-Protect bit. See the datasheet for more information.
setTCR(value);	
Set the Trickle-Charge Register. Use the defined literals to set the correct value.	
Parameters:	value: Use the defined literals to set the number of diodes and resistance used.
Returns:	Nothing
Usage:	rtc.setTCR(TCR_D1R4K); // Set the Trickle-charge register to support 1 diode and a 4K ohm resistor.
Notes:	The literals are defines as TCR_DxRyK where x is the number of diodes used (1 or 2), and y is the resistance used (2, 4 or 8 Kohm). TCR_OFF turns of the Trickle-Charge function.
writeBuffer(buffer);	
Burst-write the buffer to on-chip RAM.	
Parameters:	buffer: DS1302_RAM buffer
Returns:	Nothing
Usage:	rtc.writebuffer(ramBuffer); // Write the 31 bytes of ramBuffer to the on-chip RAM
readBuffer();	
Burst-read the on-chip RAM to the buffer.	
Parameters:	None
Returns:	DS1302_RAM buffer
Usage:	ramBuffer=rtc.readBuffer(); // Read all 31 bytes of on-chip RAM and store the in ramBuffer
poke(address, value);	
Write one single byte to on-chip RAM.	
Parameters:	address: address of byte to write (0-30) value : value to write to <address> (0-255)
Returns:	Nothing
Usage:	rtc.poke(15, 160); // Write 160 to address 15

`peek(address);`

Read one single byte from on-chip RAM.

Added in v2.0

Parameters: `address:` address of byte to read (0-30)

Returns: Byte containing data read from on-chip RAM

Usage: `b=rtc.peek(18);` // Read a single byte from address 18 and put the result in b