DS1307

DS1307 I^{2}C Real-Time Clock Arduino library

Manual

Rinky-Dink Electronics

Introduction:

This library has been made to easily interface and use the DS1307 RTC with the Arduino without needing the Wire library.

This library uses a software-based, TWI-/I^{2}C-like protocol which *will* require exclusive access to the pins used.
You will not be able to share pins with other I^{2}C devices.

From the DS1307 datasheet:

> The DS1307 serial real-time clock (RTC) is a lowpower, full binary-coded decimal (BCD) clock/calendar plus 56 bytes of NV SRAM. Address and data are transferred serially through an I2C, bidirectional bus. The clock/calendar provides seconds, minutes, hours, day, date, month, and year information. The end of the month date is automatically adjusted for months with fewer than 31 days, including corrections for leap year. The clock operates in either the 24-hour or 12-hour format with AM/PM indicator. The DS1307 has a built-in power-sense circuit that detects power failures and automatically switches to the backup supply. Timekeeping operation continues while the part operates from the backup supply.

Please note that this library only makes use of the 24-hour format.

You can always find the latest version of the library at **http://www.RinkyDinkElectronics.com/**

For version information, please refer to **version.txt**.

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`

DS1307_RAM;	
Buffer for use with readBuffer() and writeBuffer().	
Variables:	`Cell[0-56]:  Byte-array to hold the data read from or to be written to the on-chip RAM.`
Usage:	`DS1307_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 Square Wave Output rate

For use with setSQWRate()

SQW_RATE_1:	0
SQW_RATE_4K:	1
SQW_RATE_8K:	2
SQW_RATE_32K:	3

Functions:

DS1307(SDA, SCL);	
The main class of the interface.	
Parameters:	SDA: Arduino pin connected to the SDA-pin of the DS1307 (Pin 5, Serial Data) SCL: Arduino pin connected to the SCL-pin of the DS1307 (Pin 6, Serial Clock)
Usage:	`DS1307 rtc(2, 3); // Start an instance of the DS1307 class`

getTime();	
Get current data from the DS1307.	
Parameters:	None
Returns:	Time-structure
Usage:	`t = rtc.getTime(); // Read current time and date.`

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([slformat[, eformat[, divider]]]);	
Get current date as a string.	
Parameters:	slformat: **<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`

setTime(hour, min, sec);	
Set the time.	
Parameters:	hour: Hour to store in the DS1307 (0-23) min: Minute to store in the DS1307 (0-59) sec: Second to store in the DS1307 (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 datesheet for more information on the CH flag.

setDate(date, mon, year);	
Set the date.	
Parameters:	date: Date of the month to store in the DS1307 (1-31) *1 mon: Month to store in the DS1307 (1-12) year: Year to store in the DS1307 (2000-2099)
Returns:	Nothing
Usage:	rtc.setDate(4, 10, 2010); // Set the date to October 4., 2010.
Notes:	*1: No checking for illegal dates so Feb 31. is possible to input. *The effect of doing this is unknown.*

setDOW(dow);	
Set the day-of-the-week.	
Parameters:	dow: Day of the week to store in the DS1307 (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.

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.`

setOutput(enable);	
Set the SQW/OUT pin (pin 7) on the DS1307 to HIGH or LOW. This command has no effect if enableSQW() has been set to TRUE.	
Parameters:	`enable: TRUE sets the output to HIGH, and FALSE sets it to LOW.`
Returns:	`Nothing`
Usage:	`rtc.setOutput(true); // Set SQW/OUT to HIGH`

enableSQW(enable);	
Enable or disable Square Wave output on the SQW/OUT pin (pin 7).	
Parameters:	`enable: TRUE enables Square Wave output, and FALSE disables it.`
Returns:	`Nothing`
Usage:	`rtc.enableSQW(true); // Enable Square Wave output on SQW/OUT`

setSQWRate(rate);	
Set the Square Wave output rate.	

Parameters:	`rate: SQW_RATE_1   sets a 1Hz rate`
	`      SQW_RATE_4K  sets a 4.096KHz rate`
	`      SQW_RATE_8K  sets a 8.192KHz rate`
	`      SQW_RATE_32K sets a 32.768KHz rate`

Returns:	`Nothing`
Usage:	`rtc.setSQWRate(SQW_RATE_1); // Sets the rate for SQW to 1 Hz`

writeBuffer(buffer);

Burst-write the buffer to on-chip RAM.

Parameters:	buffer: DS1307_RAM buffer
Returns:	Nothing
Usage:	rtc.writebuffer(ramBuffer); // Write the 56 bytes of ramBuffer to the on-chip RAM

readBuffer();

Burst-read the on-chip RAM to the buffer.

Parameters:	None
Returns:	DS1307_RAM buffer
Usage:	ramBuffer=rtc.readBuffer(); // Read all 56 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 (0-55)
	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.

Parameters:	address: address of byte to read (0-55)
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