

DS1307 RTC Module with BAT for Raspberry Pi SKU: EP-0059



Description

The RTC module is specifically designed for Raspberry Pi. It communicates with Raspberry Pi through I2C bus. There is a Maxim DS1307 and CR1220 button cell on the board to keep the real time for a long time after the Raspberry Pi has its power off. In order to offer a convenient way to debug, there are five pins set up which are 5V, 3.3V, Rxd, Txd on board.

Feature

- Use Maxim DS1307 chip
- Extends CR1220 button cell backup
- Can be operated with shell command
- Include a serial port connector
- Programmable square-wave output signal
- Consumes Less than 500nA in Battery-Backup Mode
- Automatic Power-Fail Detect and Switch Circuitry
- Programmable Square-Wave Output Signal

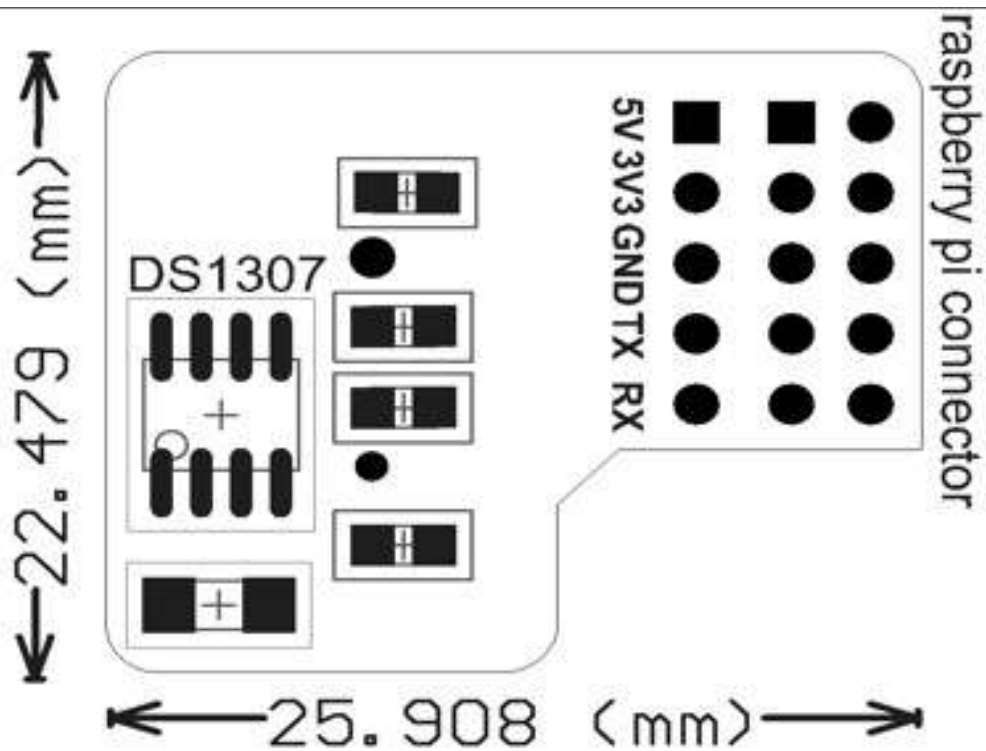
Parameters:

- Accuracy ± 20 ppm from 0°C to $+40^{\circ}\text{C}$
- Work voltage 5V
- Battery Backup Input for Continuous Time keeping
- Real-Time Clock Counts Seconds, Minutes, Hours, Day, Date, Month, and Year with Leap Year Compensation Valid Up to 2100
- Work temperature 0°C to $+70^{\circ}\text{C}$
- Ports:
 - Raspberry Pi A+/B+/2 module 2X13 connection port
 - Raspberry Pi 3, Mode B, 1GB RAM 2x20 connection port
- 2x5pin 2.54mm connector

Packages

- 1 x RTC Module
- 1x CR1220 Battery

Mechanical Drawing:



How to Connect:

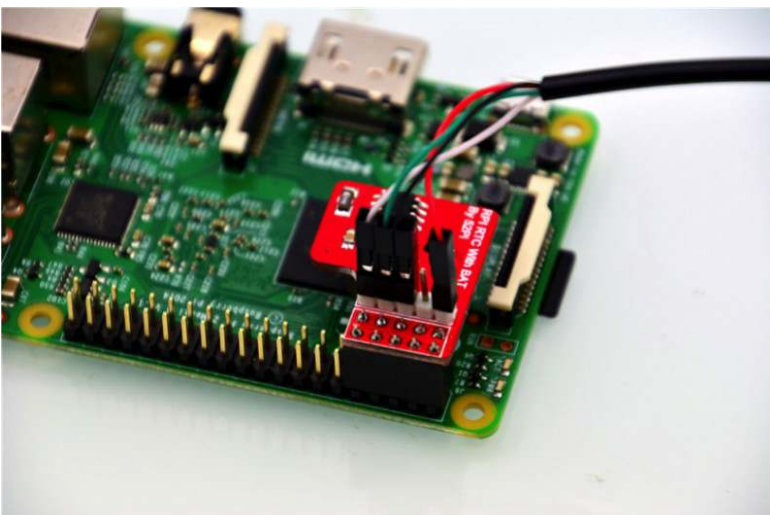
1. Software Requirement: Base Raspbian Operating System



2. Connection: Just insert the module into Raspberry Pi



3. USB-to-TTL cable wire connect to RTC



4. Finally

Note RTC 5v pin connect to USB-to-TTL Red wire (5v)
RTC GND pin connect to USB-to-TTL black wire (GND)
RTC TX pin connect to USB-to-TTL Green wire (TX)
RTC RX pin connect to USB-to-TTL white wire (RX)

How to configure in terminal

Here we assume that you have already burned the Raspbian Image into TF card and connect to your PC and logged in. Open a terminal and modify /boot/config.txt file using what you favorite editor such as vim.tiny or nano, add parameters as following picture:

```
pi@raspberrypi:~$ !grep
grep -v "#" /boot/config.txt |grep -v "^$"
hdmi_safe=1
disable_overscan=0
overscan_left=16
overscan_right=16
overscan_top=16
overscan_bottom=16
hdmi_force_hotplug=1
dtparam=i2c_arm=on
dtparam=spi=on
device_tree=bcm2710-rpi-3-b.dtb
dtoverlay=seeed-tft28b,speed=32000000,rotate=180
dtoverlay=pi3-miniuart-bt-overlay
dtoverlay=i2c-rtc,ds1307
force_turbo=1
dtoverlay=ads7846,penirq=22
dtparam=audio=on
```

You can read /boot/overlay/README and find this info to add support for ds1307 I2C Real Time Clock device.

Name: i2c-rtc
Info: Adds support for a number of I2C Real Time Clock devices
Load: dtoverlay=i2c-rtc,<param>=<val>
Params: ds1307 Select the DS1307 device

Please ensure that /boot/config.txt file include those three parameters:

```
device_tree=bcm2710-rpi-3-b.dtb
dtoverlay=i2c-rtc,ds1307
dtparam=i2c_arm=on
```

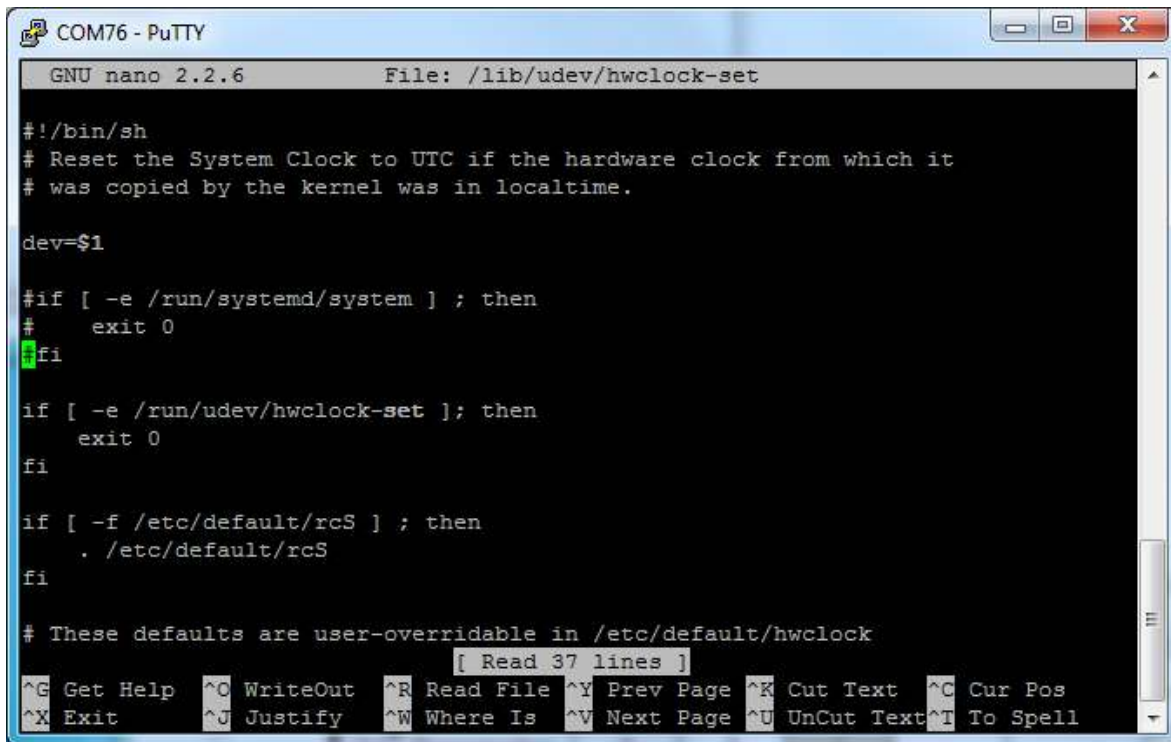
After that, please make sure you have disabled the "fake hwclock" which interferes with the 'real' hwclock

```
sudo apt-get -y remove fake-hwclock
sudo update-rc.d -f fake-hwclock remove
```

Now with the fake-hw clock off, you can start the original 'hardware clock' script.
Edit the script file `/lib/udev/hwclock-set` with nano or vim editor and comment out these three lines:

```
if [ -e /run/systemd/system ]; then
    exit 0
fi
```

Finally result like this:



```
COM76 - PuTTY
GNU nano 2.2.6 File: /lib/udev/hwclock-set
#!/bin/sh
# Reset the System Clock to UTC if the hardware clock from which it
# was copied by the kernel was in localtime.

dev=$1

#if [ -e /run/systemd/system ] ; then
#   exit 0
#fi

if [ -e /run/udev/hwclock-set ]; then
    exit 0
fi

if [ -f /etc/default/rcS ] ; then
    . /etc/default/rcS
fi

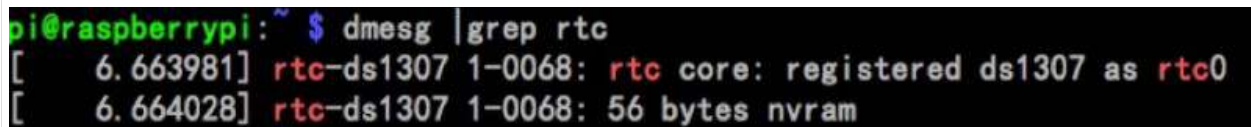
# These defaults are user-overridable in /etc/default/hwclock

[ Read 37 lines ]
^G Get Help  ^O WriteOut  ^R Read File  ^Y Prev Page  ^K Cut Text    ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is  ^V Next Page  ^U UnCut Text ^T To Spell
```

save and reboot your Rpi.

How to Check it

After reboot and log in, open a terminal and typing this command to check if RTC module is functional. `dmesg |grep rtc` if you can see this picture means that your RTC module is working properly.



```
pi@raspberrypi:~$ dmesg |grep rtc
[ 6.663981] rtc-ds1307 1-0068: rtc core: registered ds1307 as rtc0
[ 6.664028] rtc-ds1307 1-0068: 56 bytes nvram
```

and then you can adjust your system clock as following command:



```
pi@raspberrypi:~$ sudo date 051014302016.20
Tue May 10 14:30:20 UTC 2016
```

Note: 051014302016.20 is equal to mmDDHHMMYYYY.ss, more information please using 'man date' command.

Last step, set the Hardware Clock to the current System Time.

```
pi@raspberrypi: ~ $ sudo hwclock -w step 1
pi@raspberrypi: ~ $
pi@raspberrypi: ~ $
pi@raspberrypi: ~ $ date && sudo hwclock -r step 2
Tue May 10 14:37:54 UTC 2016
Tue May 10 14:37:54 2016 -0.010701 seconds
pi@raspberrypi: ~ $
```

FAQ

Question: Why does my RTC module running a wrong time when i reboot my Pi?

Answer: Please do remember to disable the "fake hwclock" which interferes with the 'real' hwclock as following commands:

```
sudo apt-get -y remove fake-hwclock
sudo update-rc.d -f fake-hwclock remove
```