# RTCC – Real Time Clock & Calendar

## For the Raspberry Pi

## Introduction

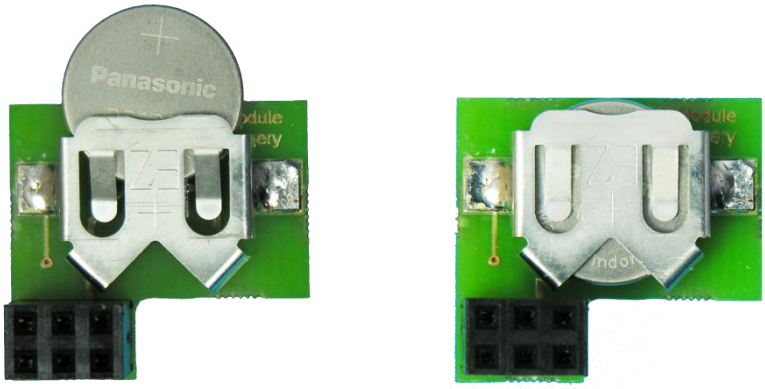
Thank you for purchasing this small module, designed to keep the time when no internet connection is present. Hopefully the information provided in this document will be all that you need, however I welcome comments, questions and constructive criticism via email at colin@circuitsurgery.com

## Description

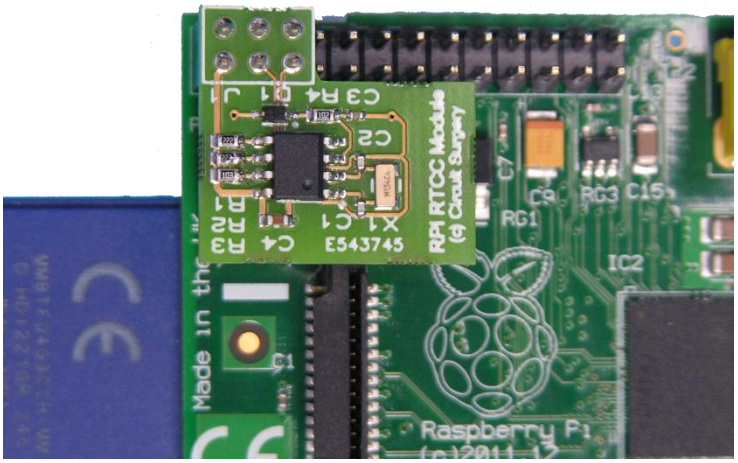
This is a small add-on board designed to fit directly onto the GPIO port of the Raspberry Pi and interface via the I2C bus, providing the facility to keep track of time independently of the internet. A small battery (type number BR1225 or equivalent – NOT included with the module due to shipping restrictions) may also be fitted in order to retain the time when the Raspberry Pi is switched off.

## Fitting

If the back-up facility is required, then a battery should be fitted into the holder on the back of the module. Make sure that the face of the battery with the "+" symbol is facing away from the board when inserting it.




The module can now be fitted onto GPIO pins 1-6 of the Raspberry Pi.



## Setup and use

Before you can access the Real Time Clock chip it will be necessary to install the software tools for accessing the I2C bus if you've not already done so.

Make sure your Pi is connected to the internet and from the command line, type:

- ***sudo apt-get update***
- ***sudo apt-get install i2c-tools***

This will ensure the operating system is up to date, and the second line will install "i2c-tools" software library

For the Raspian operating system, run Nano and modify the "Modules" file by entering:

- ***sudo nano /etc/modules***

Add these two lines to the end of the code:

- ***i2c-bcm2708***
- ***i2c-dev***

Now reboot the Pi so that the new library can "settle in"!

If there is a file called ***"/etc/modprobe.d/raspi-blacklist.conf"*** then this needs to be edited and the two lines:

- ***blacklist spi-bcm2708***
- ***blacklist i2c-bcm2708***

should be commented out by adding a "#" in front of them. So again at the command line prompt, enter:

- ***sudo nano /etc/modprobe.d/raspi-blacklist.conf***

...and then edit the two lines to read:

- ***#blacklist spi-bcm2708***
- ***#blacklist i2c-bcm2708***

Save the file, then type:

- ***sudo i2cdetect -y 1*** (***sudo i2cdetect -y 0*** if you have an early version Raspberry Pi)

This will show a table with any i2c addresses that are in use. The RTCC module uses address 0x6f .

In order to be able to access i2c-tools via Python, you will need to install the "python-smbus" package. Type:

- ***sudo apt-get install python-smbus***

That completes the basic setup for accessing the i2c bus and allowing direct access to the RTCC chip, which may now be tested using the Python script at http://www.circuitsurgery.com/sw/rtcctest.zip

This program firstly reads all the registers, prints them to the screen and also saves them to a file. The "ST" bit is then checked and if clear the program sets it to start the clock running. Likewise the "VBATEN" bit is set in order to enable the battery backup facility.

The user is then prompted to enter the current date and time and having collected the input, the registers are loaded with the information.

- **Note that for testing purposes I suggest entering a fictitious time at this stage, the reason will become clear later!**

The list of registers and their associated contents are displayed for a second time and then the current time (or, more accurately, the time entered by the user) is displayed continuously, updating every second.

Escape back to the system prompt by using "Ctrl + Z"

This is a quick (and dirty!) way to check the functionality of the RTCC chip, and also serves as a starting point for your own software.

## "hwclock"

In order to be able to use the "***hwclock***" command, a driver will need to be installed. There is not a specific driver for the MPC7941x chip, but the one for a different chip (the DS1307) is suitable.

From the command prompt, type:

- ***sudo modprobe rtc-ds1307***
- ***sudo bash*** (enters Bash as Root)
- ***echo mcp7941x 0x6f > /sys/bus/i2c/devices/i2c-1/new_device*** (for Rev.2 Pi's) OR...
- ***echo mcp7941x 0x6f > /sys/bus/i2c/devices/i2c-0/new_device*** (for Rev.1 Pi's)
- ***exit*** (Return to standard user)

Now you will be able to issue the command:

- ***sudo hwclock -r***

which will read the contents of the clock, which should correspond with the time entered at the first testing stage, and:

- ***sudo hwclock -w***

which will write the system time to the RTCC. Now, repeat the "***hwclock -r***" command and this time you should see that the output has changed to reflect the system time. The following steps allow the various modules to be loaded, so that the RTCC is available at system boot. Type:

- ***sudo nano /etc/modules***

Add the following line:

- ***rtc-ds1307***

Save and close the file. Now open /etc/rc.local for editing:

- ***sudo nano /etc/rc.local***

Immediately before the line – "***exit 0***" add the following lines:

- ***echo mcp7941x 0x6f > /sys/class/i2c-dev/i2c-1/device/new_device***
- ***hwclock -s***

These two lines will create the device on boot and then write the time from the RTCC to the system.

Your Pi is now set up with a hardware Real Time Clock and can tell the time regardless of whether it's connected to the internet or not!

## Odds and Ends

Full details of the MCP79410 RTCC chip can be found at:
http://www.circuitsurgery.com/docs/mcp79410-rtcc.pdf . Section 5 of which deals with the various registers and addresses.

For further information or support for this module, please email colin@circuitsurgery.com

Comments, suggestions and constructive criticism are always welcomed!