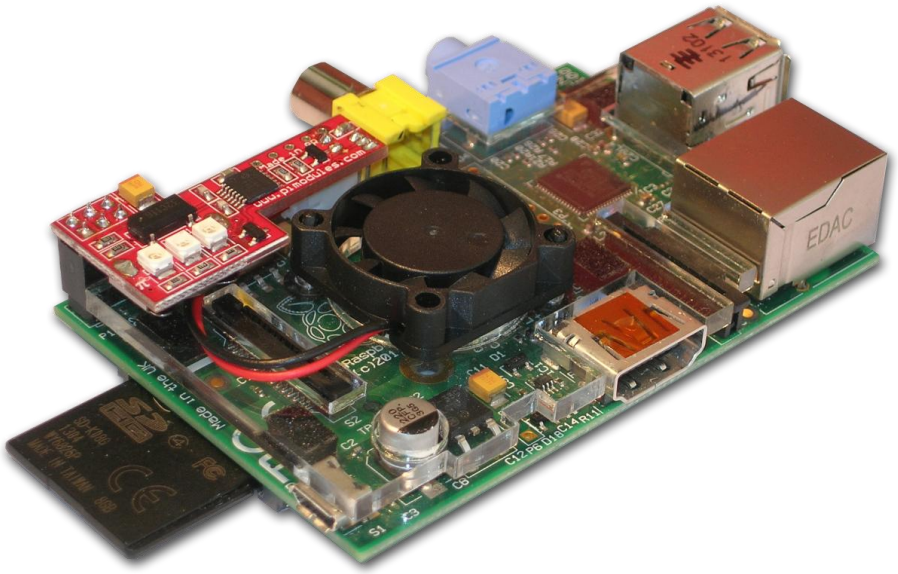
PiCoolFan

with Real Time Clock

Advanced Pi Cooling Fan System

for use with

RaspberryPi®



User Manual

Version 1.04

Firmware Release 1

Hardware Release 1

it is Pi - it is Cool - it is Fun

Credits

Our Company would like to thank the following people that reviewed and, many times, commented and corrected this document before we released it to the public domain.

Marcello Antonucci - Italy

Mathew E. Rivers - USA

Table of Contents

PiCoolFan – Advanced Pi Cooling Fan System

Introduction

The **PiCoolFan** is an Advanced Cooling Fan System for the RaspberryPi®, one that offers total temperature control over the RaspberryPi®. It is addressed to all RaspberryPi® users whose board reaches a high temperature, for whatever reason. There is no need for any additional cabling or Power Supply, as the **PiCoolFan** is powered by the same Power Supply of your original RaspberryPi® board; just insert the **PiCoolFan controller** on top of the **P1** connector of your RaspberryPi®. The **PiCoolFan is equipped with** an embedded temperature measurement system that continuously checks the RaspberryPi® PCB temperature with a dedicated sensor, and according to the measured temperature, the **PiCoolFan micro controller** starts, stops, or regulates the rotational speed of the embedded micro-fan. Thanks to the Air Distribution Plate, the **PiCoolFan** cools not only the microprocessor on the RaspberryPi® board but all the heat-generating devices, as well as the whole RaspberryPi® PCB. The embedded microcontroller is easily accessible by the user trough the **I^2C PiCO** interface and allows reading of the measured temperature, setting of temperature threshold, as well as starting/stopping the embedded micro fan. In addition the **PiCoolFan** offers on the same PCB a battery backed RTC and Real Time Powering Voltage Monitoring.

Applications

The **PiCoolFan** as an add-on module is addressed to all RaspberryPi® users that need to have a total control over the RaspberryPi® board temperature especially when running outdoor or in high temperature environments. The embedded micro-fan is of high quality, produces with very low noise (also thanks to the embedded PWM control technology), and is long lived.

System

The **PiCoolFan module** consists of the following units:

- The PWM Temperature Adjustment Fan Controller unit
- The micro-Fan
- Battery (not included) backed RTC
- The Air Distribution Plate (fit-in to most of the cases)

Features

The features of the **PiCoolFan module** are the following:

- Powering from the P1 connector of the RaspberryPi®
- Micro-controller supervised
- PWM FAN speed regulation

- Plug and Play
- 3 LED based information system (Red - hot, Blue - cold, Flashing Green – powering status)
- Full control over the system via **I²C** interface (The **PiCO** Interface):
    - Unconditional Fan ON/OFF
    - PWM FAN speed regulation
    - Temperature threshold set/get
    - Current System Temperature read
- Supports Celsius and Fahrenheit scale
- Real Time powering voltage monitoring with programmable threshold
    - Flashes continuously Green if power is within thresholds
    - Flashes fast if power is higher than threshold
    - Flashes slow if power is lower than threshold
- Can be used inside of the most of already existing cases
- Embedded RTC with separated Battery (not included)

What is in the BOX?

This package comes with everything you need to start using the **PiCoolFan** right out of the box. It is assembled, tested and contains all required accessories. A little work is necessary in order to setup the complete RaspberryPi® and **PiCoolFan** in a single full operating system.

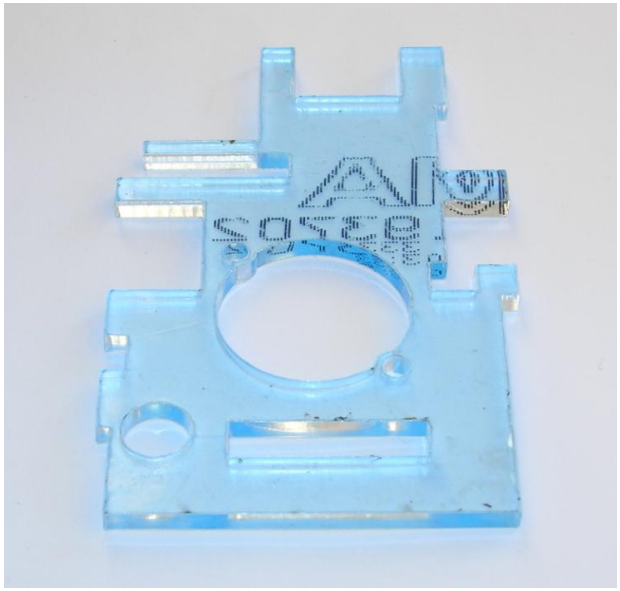
Each Package contains the following parts:

- The **PiCoolFan** module (PCB) assembled and tested
- Air Distribution plate
- FAN with cable
- Set of 2 pcs of tree clips required for FAN mounting onto Air Distribution Plate
- Set of 5 pcs of Rubber spacers required for the Air distribution mounting

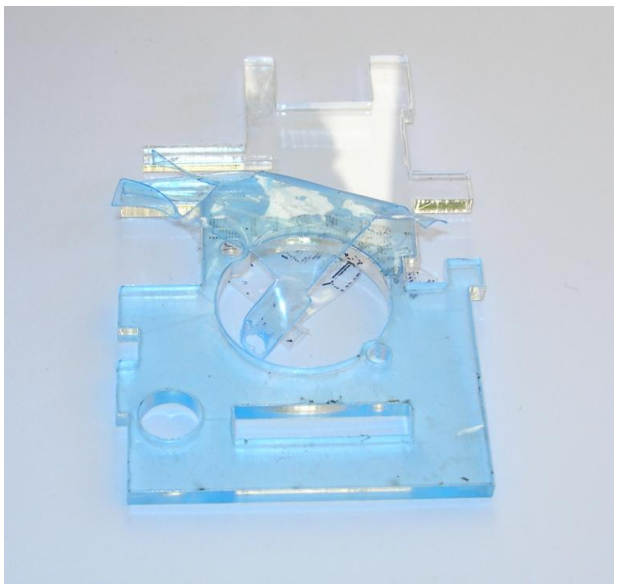
Hardware Installation

Before assembly, please download a print out of the PDF file with guidelines where to place the rubber spacers on the Air Distribution Plate.

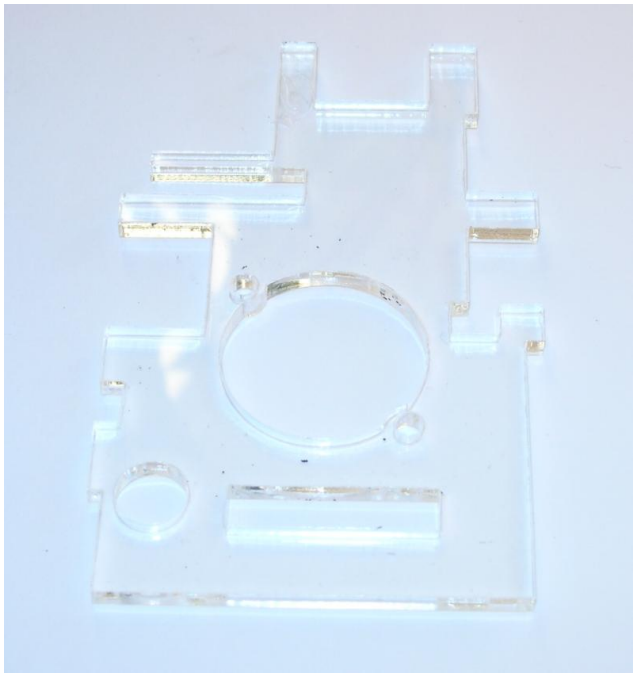
Please follow the following steps in order to assemble and run the **PiCoolFan.**



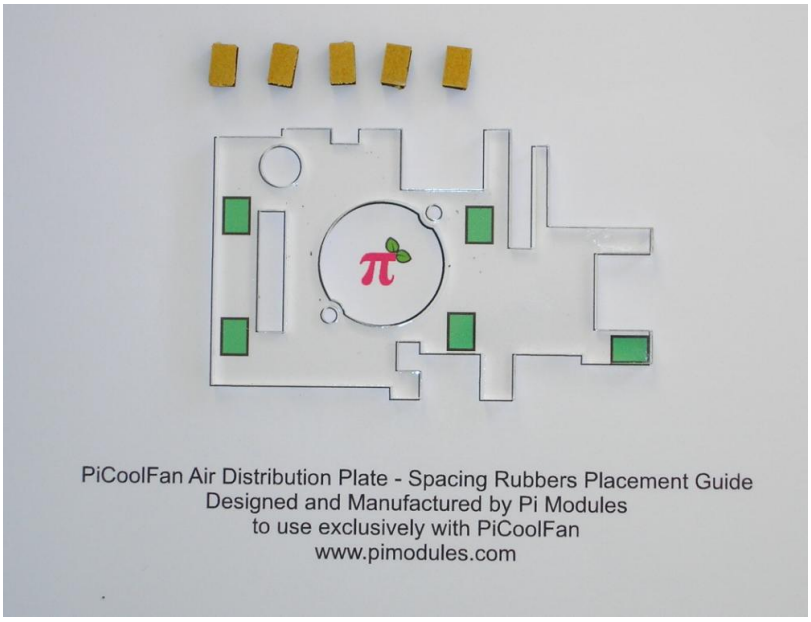
Place the Air Distribution Plane on the desktop



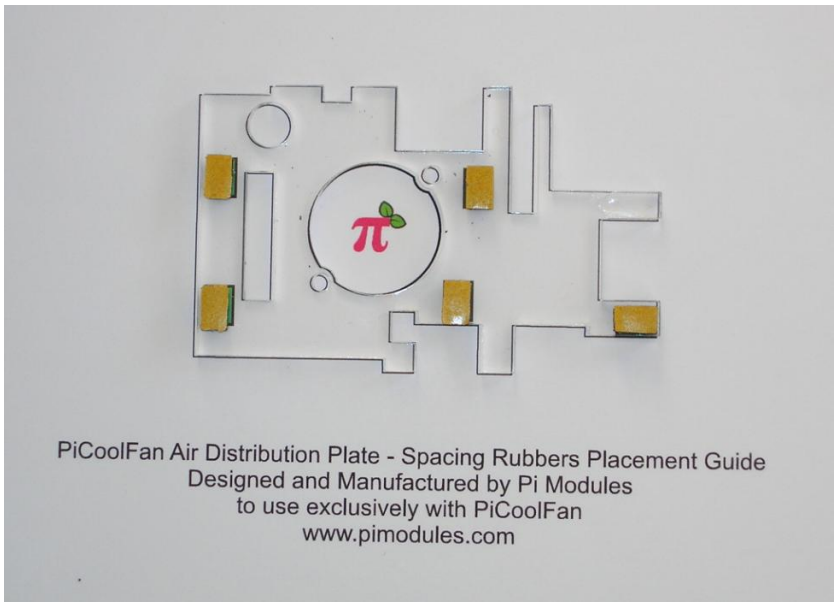
Remove the protective tapes from the Air Distribution Plate



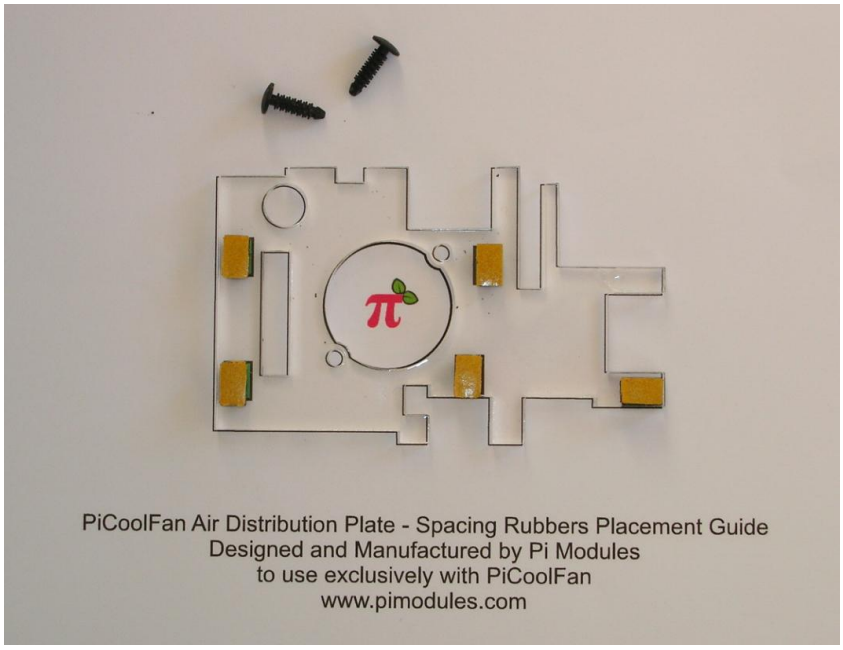
After removing the protective tape from the Air Distribution Plate, you will have the above picture.



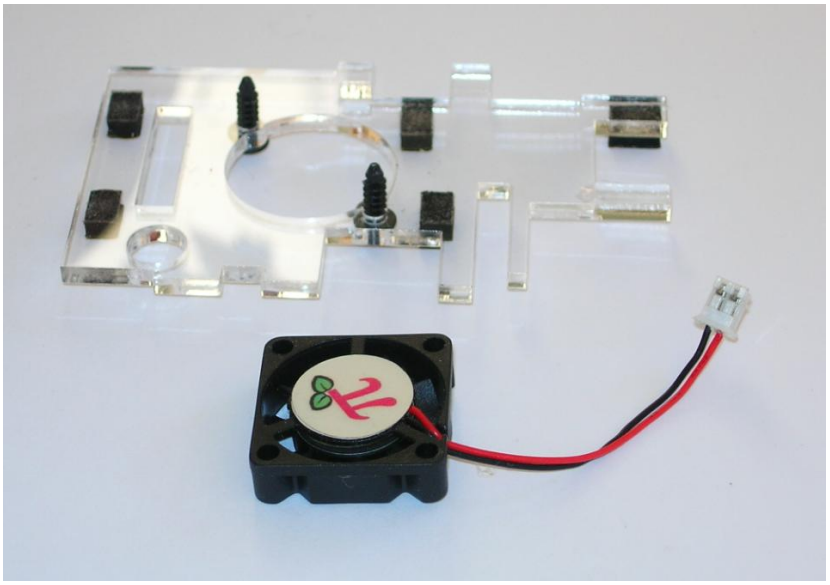
Lay the Air Distribution Plate on top of the Spacing Rubbers Placement Guide, and glue the self adhesive Spacing Rubbers to the Air Distribution Plate as shown.



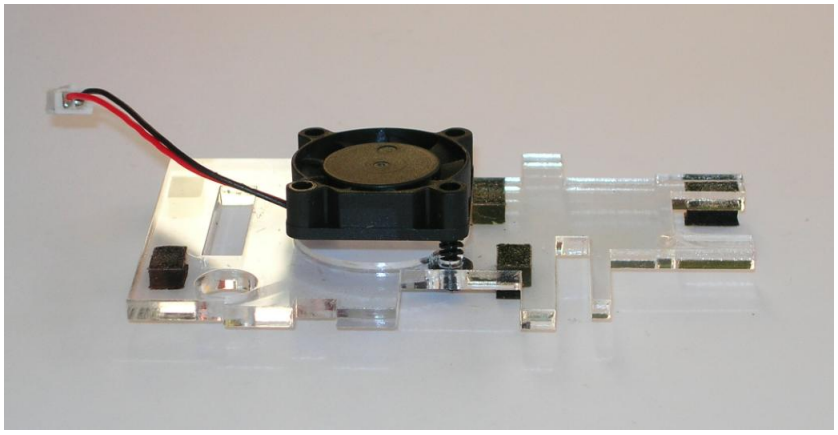
Remove the protective (brown) tape only from one side. The second self-adhesive side is used for permanent assembly of the Air Distribution Palate on the RPi. However it is not necessary.



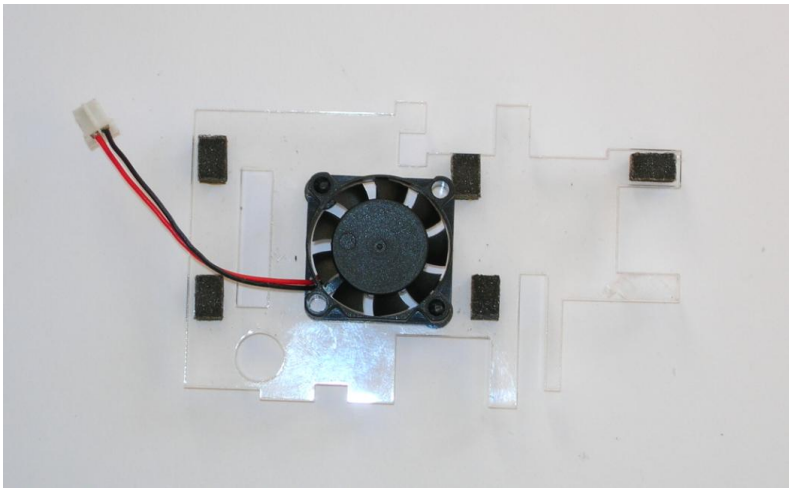
Prepare the tree clips and pass them over the 2 holes. Please take care to have them on the proper side.



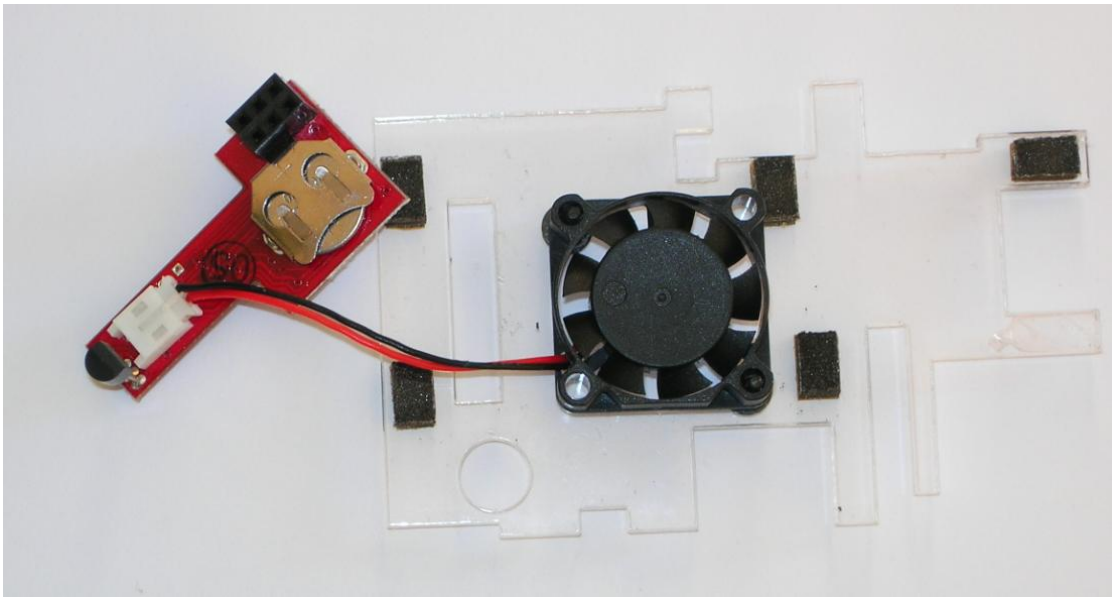
Prepare the FAN to be mounted over the Air Distribution Plate using the already assembled tree clips.



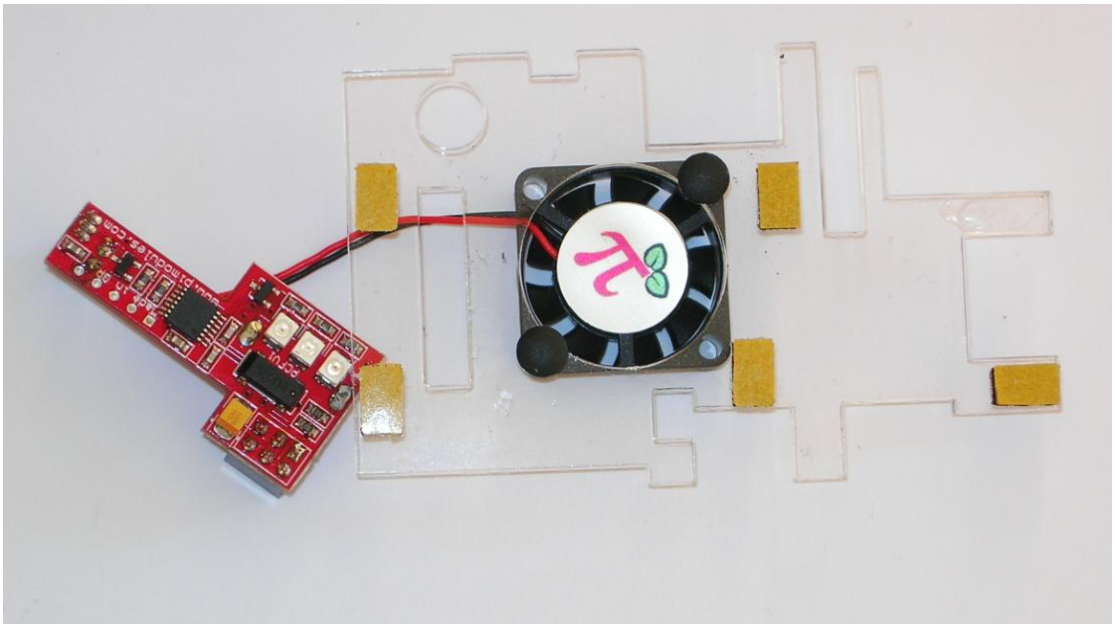
Put the FAN on the tree clips and press it to the Air Distribution Plate in order to stabilize the whole construction.



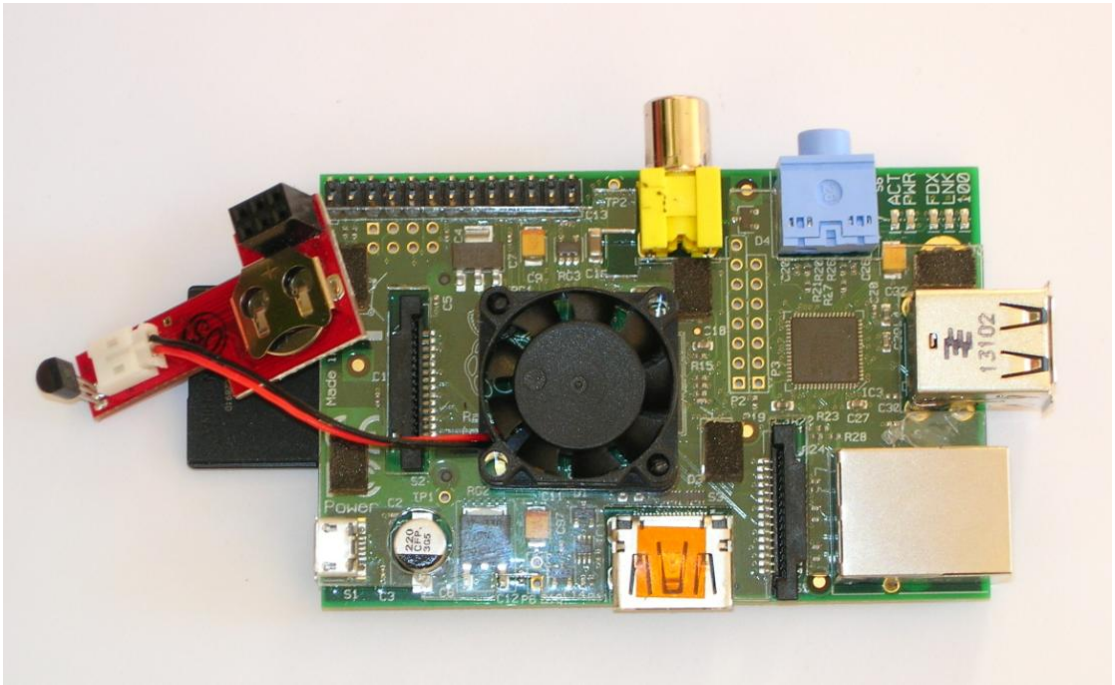
After Assembly the above construction will be available.



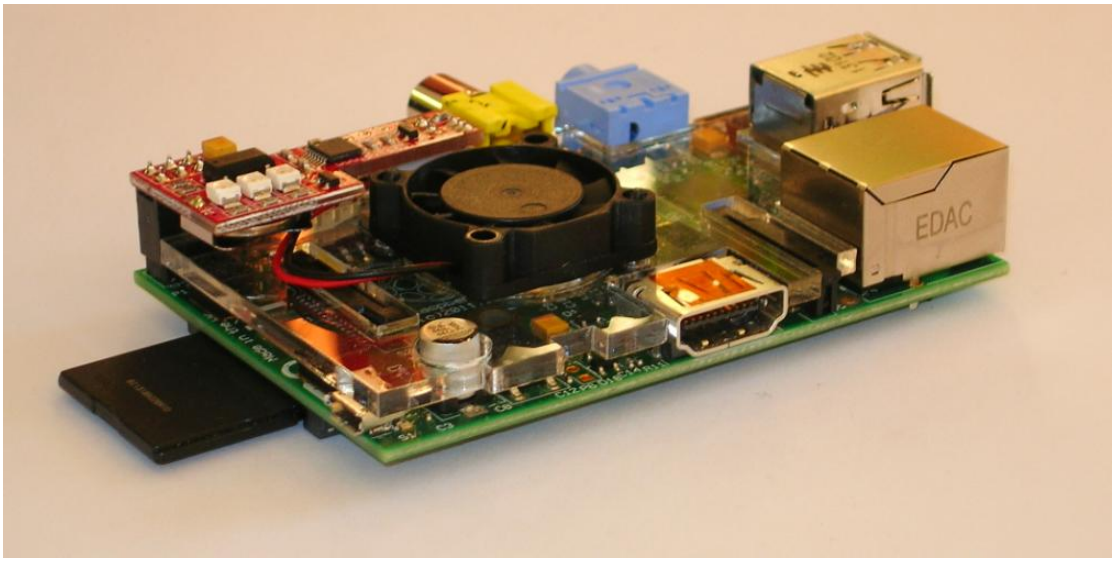
Please insert the coin battery to the battery holder if planning= to use the RTC. However, it is not necessary if you are not planning to use the RTC. Fit in the FAN connector to the **PiCoolFan** PCB socket.



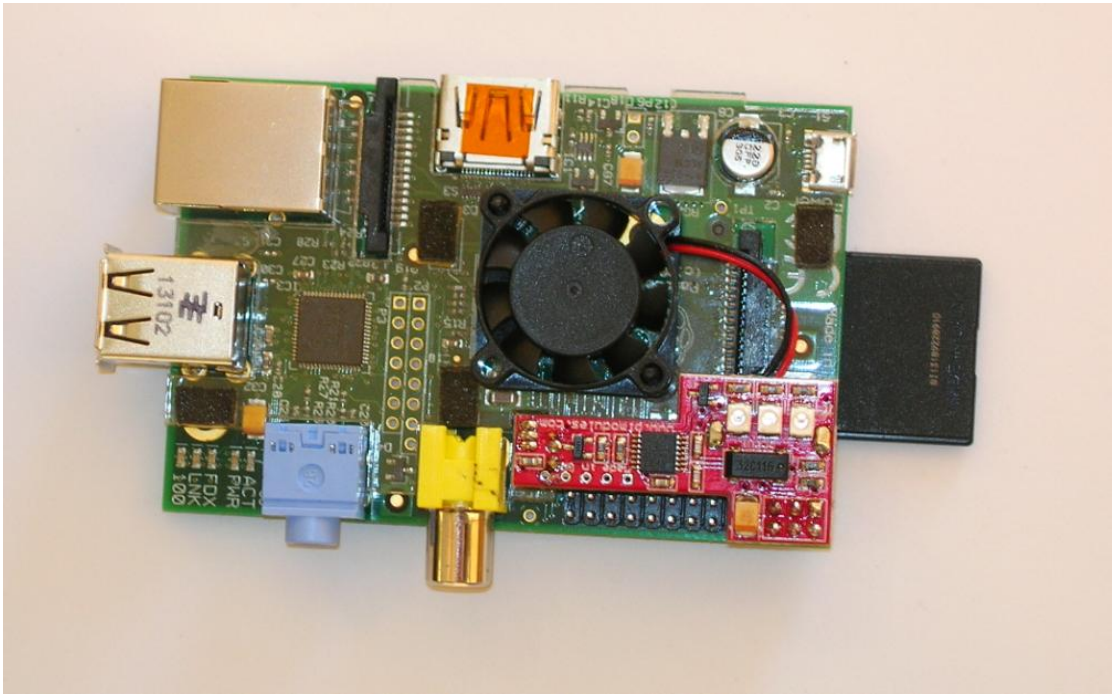
The **PiCoolFan** FAN, PCB and the Air Distribution Plate after the assembly will looks like above picture.



Place the Air Distribution Plate on the RPi, if needed remove the brown tape from the rubber spacers in order to use self adhesive facility to glue it on the RPi.



Fit in the **PiCoolFan** PCB on the P1 connector taking into account that the Temperature Sensor needs to be placed inside the hole of the Air Distribution Plate and need to touch the RPi PCB near the C15. Please be sure that the FAN supply cable is placed in a free area.



Now you can start Using **PiCoolFan**. The system is ready to be used and do not need any interface from the user. In order to use more advanced functionality please follow the next chapter of the manual.

Software Installation

This section describes how to program, read, set and handle all parameters of the **PiCoolFan** including the RTC. This also specifies the PiCO interface.

Setting-up the I²C interface and RTC

The I²C Ports on the RaspberryPi® is not enabled by default. Follow these steps to enable the I²C port and then the RTC communicating through **I²C** with RaspberryPi®.

First it is necessary to edit the config file that sets the the **I²C** port to default disabled. This setting is stored in /etc/modprobe.d/raspi-blacklist.conf. Use nano to edit this but you can also use any other editor you are comfortable with.

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

Once this file is open find this line *blacklist i2c-bcm2708* and comment it out by adding# to the front of it.

#blacklist spi-bcm2708

#blacklist i2c-bcm2708

Edit /etc/modules

$sudo nano /etc/modules

And add the following:

i2c-bcm2708
i2c-dev
rtc-ds1307

Add the modules to the kernel (they will automatically be added on subsequent boots from /etc/modules):

$sudo modprobe i2c-bcm2708
$sudo modprobe i2c-dev
$sudo modprobe rtc-ds1307

Reboot the system

$sudo reboot

Install **I²C** tools

$sudo apt-get install i2c-tools

Look for ID #68 with i2cdetect

On a 256MB Raspberry Pi Model A:

$sudo i2cdetect –y 0

On a 512MB Raspberry Pi Model B:

$sudo i2cdetect –y 1

The result should look like:

```
pi@raspberrypi ~ $ sudo i2cdetect -y 1
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- 68 -- -- -- 6c -- -- --
70: -- -- -- -- -- -- -- --
pi@raspberrypi ~ $
```

I²C PiCoolFan Simulated DS1307 Clock detection

Then as roots do the following for model of RaspberryPi® you have

On a 256MB Raspberry Pi Model A:

$sudo bash
 # echo ds1307 0x68 > /sys/class/i2c-adapter/i2c-0/new_device
 # exit

On a 512MB Raspberry Pi Model B:

$sudo bash
 # echo ds1307 0x68 > /sys/class/i2c-adapter/i2c-1/new_device
 # exit

The result should look like:

```
pi@raspberrypi ~ $ sudo bash
root@raspberrypi:/home/pi# echo ds1307 0x68 > /sys/class/i2c-adapter/i2c-1/new_device
root@raspberrypi:/home/pi# exit
exit
pi@raspberrypi ~ $
```

PiCoolFan Simulated DS1307 Clock sudo bash commands execution

Then check for time from the clock (which will show Sat 01 Jan 2000 if it is the first time it is used):

$sudo hwclock -r

Then write the current system time to the clock:

$sudo hwclock -w

Then edit /etc/rc.local:

$sudo nano /etc/rc.local

and add the following before exit 0:

On a 256MB Raspberry Pi Model B:

echo ds1307 0x68 > /sys/class/i2c-adapter/i2c-0/new_device
hwclock -s

On a 512MB Raspberry Pi Model B:

echo ds1307 0x68 > /sys/class/i2c-adapter/i2c-1/new_device
hwclock -s

The Peripheral I²C Control – PiCO – Interface

The **PICO** is an implementation of **I²C** interface adapted for easy control of the peripheral connected to the RaspberryPi® via command line. Values over the **PiCO** interface could be presented as HEX or BCD coded numbers. These give the user the ability to handle supported devices just by human recognized commands. In the **PiCO** interface, due to the mask feature, there are implemented multiple I2C addresses, that are spited to each peripherals. It allows co-existing multiple devices over a single I2C interface (i.e. FAN control system with RTC). Thanks to human-understandable and simple commands, control of peripherals is extremely simple. Control from programming language is also possible and as easy. The core concept of the **PiCO** interface is that all peripheral device control and data exchange between it and RaspberryPi® variables are common for the **I²C** interface as also for the peripheral itself. Therefore, any change of them by either the RaspberryPi® or the peripheral causes immediate update and action.

Address	Name	Type	R/W	Explanation
0 or **0x00**	mode	byte	Write	0 – unconditional FAN OFF 1 – unconditional FAN ON 2 – FAN AUTO ON/OFF controlled by PCF **Default value: 2**
1 or **0x01**	speed	byte	Write	0 – FAN speed 00% (OFF) 1 – FAN speed 100% 2 – FAN speed 25% 3 – FAN speed 50% 4 – FAN speed 75% **Default value: 3**
2 or **0x02**	ctemp	word	Read	Contains actual system temperature If Celsius selected then 2 digits If Fahrenheit selected then 3 digits
4 or **0x04**	ttemp	word	Write	Threshold temperature **Default value: 42 DEG Celsius**
6 or **0x06**	scale	byte	Write	Celsius or Fahrenheit 0 – Celsius 1 - Fahrenheit **Default value: 0**
7 or **0x07**	status	byte	Read	Running FAN Status 0 – is not running OFF 1 – is running ON
8 or **0x08**	vcc_pi	word	Read	Actual value of VCC supplying RPI on P1 5V Pin in 10^{th} of mV
10 or **0x0A**	vcc_upi	word	Write	Up Limit of VCC supplying RPI on P1 5V Pin in 10^{th} of mV **Default value: 520 (5.2 V DC)**
12 or **0x0C**	vcc_dpi	word	Write	Down Limit of VCC supplying RPi on P1 5V Pin in 10^{th} of mV **Default value: 480 (4.8 V DC)**
14 or **0x0E**	version	byte	R/W	**Read:** Firmware and Hardware Version

				0xHF (Hardware, Firmware) **Write:** 0x00 then restore factory defaults **Write:** 0xee then read pcf data from flash memory **Write:** 0xff then store pcf data to flash memory

In order to handle the PiCO command a write or read action should be performed to the I2C on the address where the PiCO interface has been assigned. For the PiCoolFan the address is set to **0x6C** and any communication should be done though this address. For reading of the word variable the **w** should be placed at the end of line.

Examples of commands

$sudo i2cget –y 1 0x6C 2 w

Will generate the result of the current temperature in Celsius and in BCD format in order to simplify the reading

0x0034

This means that system temperature is 34 degrees in Celsius

$sudo i2cset –y 1 0x6C 0 1

Will switch unconditional ON the FAN with the speed defined in the variable '1'

$sudo i2cset –y 1 0x6C 0 0

Will switch unconditional OFF the FAN with the speed defined in the variable '1'

$sudo i2cget –y 1 0x6C 8 w

Will read the actual voltage on the P1 5 V pin and give value of 0XXX in 10^{th} of mV

0x0496
It means 4.96 V DC in BCD format

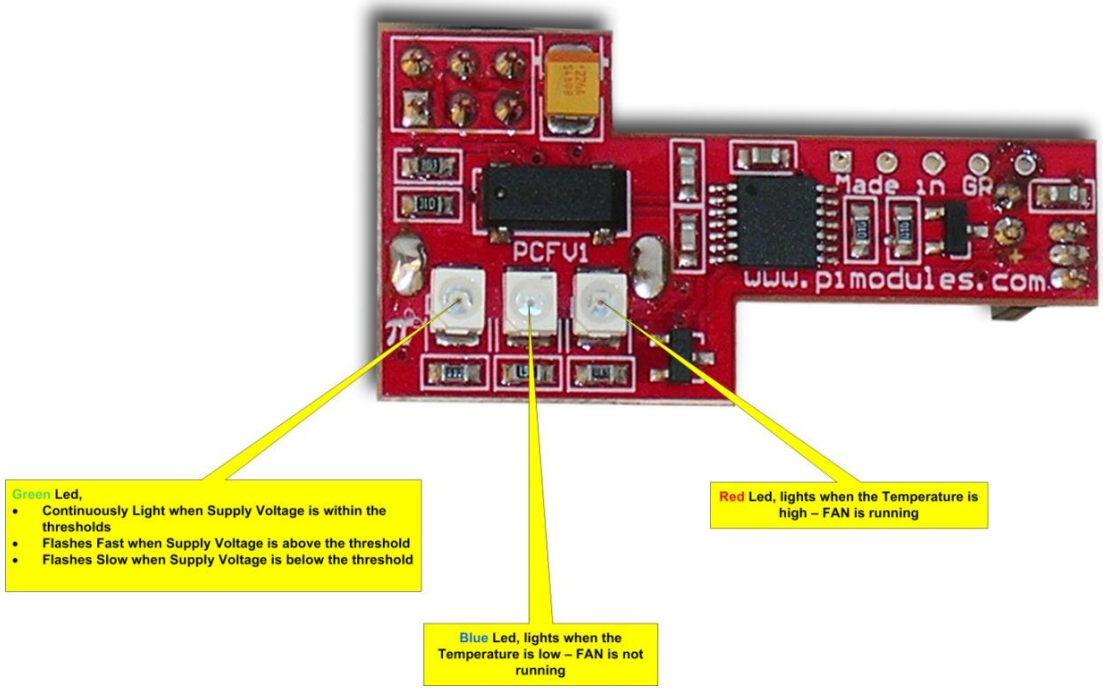
LEDs based User Simple Interface

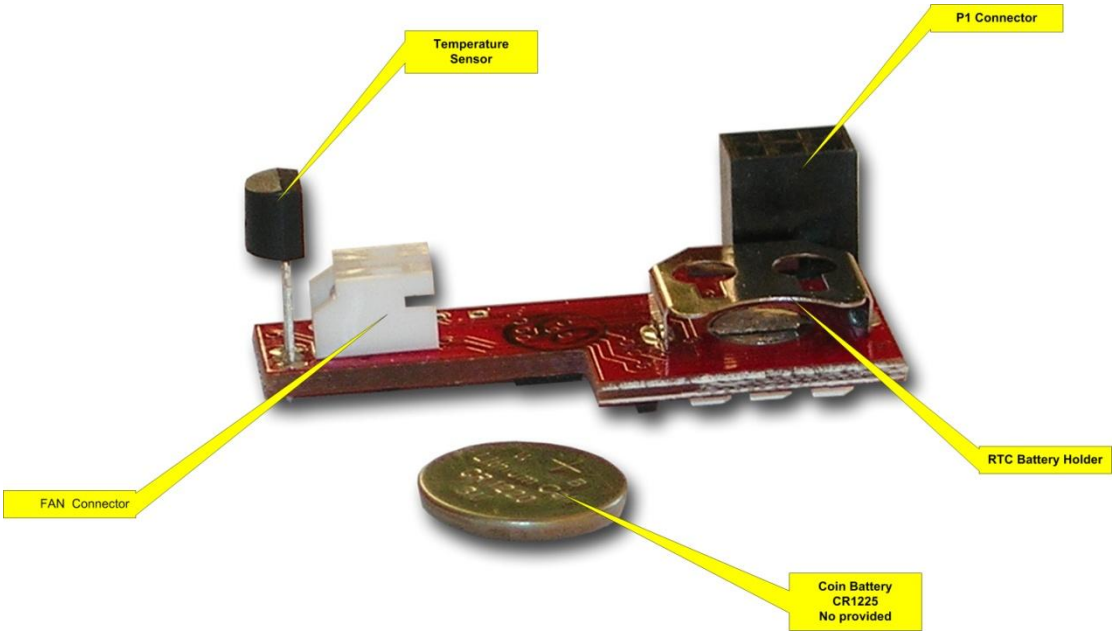
There are 3 single color LEDs that helps the user to see what happens with the system without accessing any command.

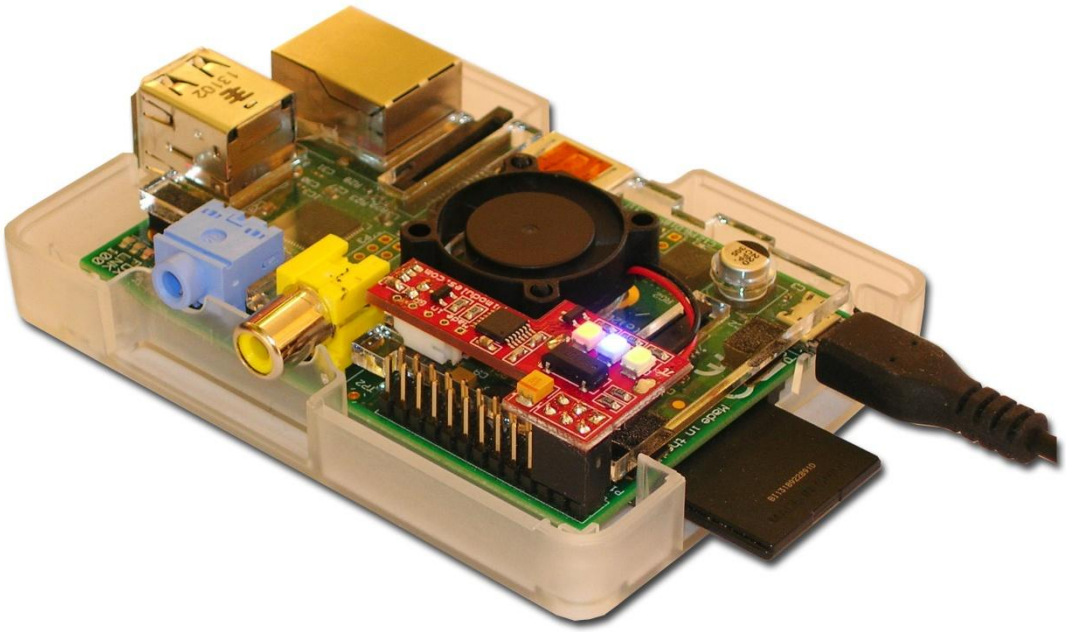
- **GREEN LED**
  Lights when P1 5V is within the requested range
  Flashes fast if P1 5V is above threshold
  Flashes slow if P1 5V is below threshold

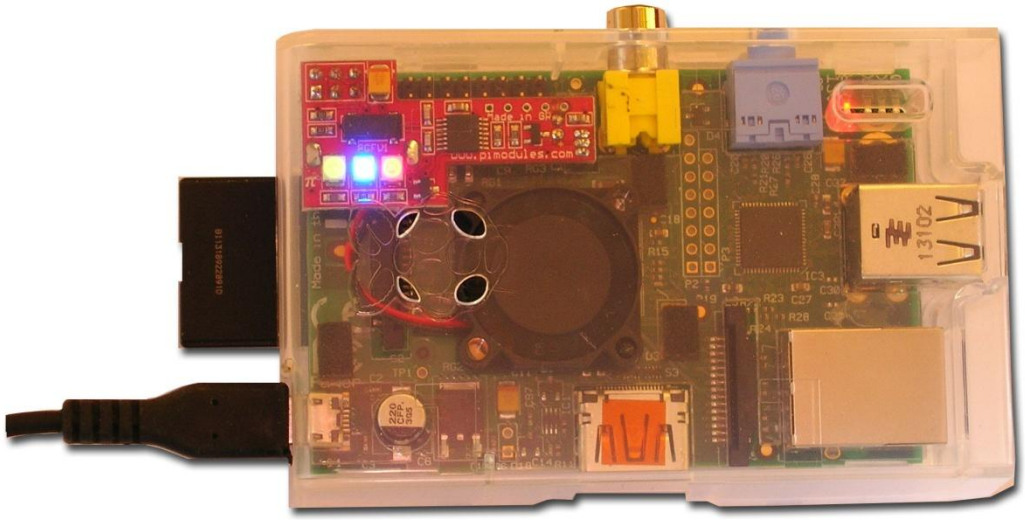
- **BLUE LED**
  Lights when temperature of the system is below the threshold (factory default 42 Celsius). Fan is not running.

- **RED LED**
  Lights when temperature of the system is above the threshold (42 Celsius). Fan is running.




Temperature Sensor
P1 Connector
FAN Connector
RTC Battery Holder
Coin Battery CR1225 No provided




www.pimodules.com