Tuning the Factory Default Temperature Threshold

The **PiCoolFAN4** is ready to be used without any software installation, making use of the temperature sensor on the **PiCoolFan4** PCB.

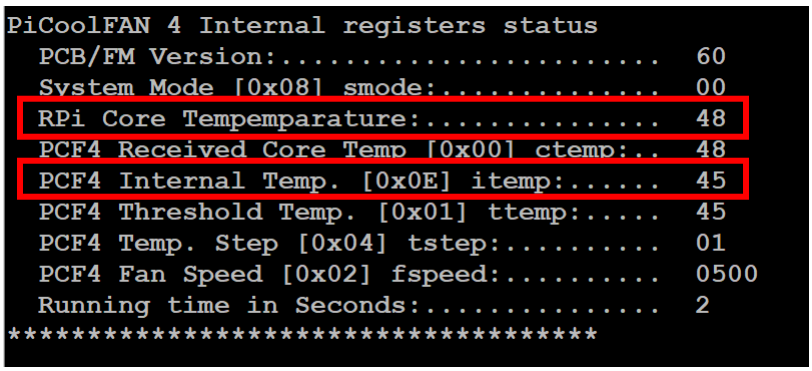
This factory default setup (threshold) has been tuned for use inside an official Raspberry Pi case (in the local environment of the production facility), however if you are using a different case, in a different environment, or just simply want to tune the settings - this can be achieved easily with the provided software.

Users who are not using a case with their Raspberry Pi are also strongly advised to tune the fan threshold.

Remember: The **PiCoolFAN4** temperature sensor is 3.5mm above the Raspberry Pi CPU. It is not a direct reading from the Raspberry Pi's CPU temperature sensor.

Different ambient temperatures, draughts or enclosures usually require tuning of the temperature threshold to ensure the fan is activated at the right time.

1. Install the **PiCoolFAN4**
2. Close your system in desired case, with heatsink (*see page 8*) or without
3. Place your system in the desired location where it will be used
4. Run your system for 5 – 10 minutes
5. Run the **pcf4_status.py** script (*in your home directory as per page 10)*
6. Observe both the **PCF4 Internal Temp itemp** (*PiCoolFAN4 temperature sensor*) and the **RPi Core Temperature** (*Raspberry Pi CPU temperature sensor*) readings as indicated in the image below:



These readings give an indication of the link between the direct temperature of the Raspberry Pi CPU and the **PiCoolFAN4** temperature sensor readings.

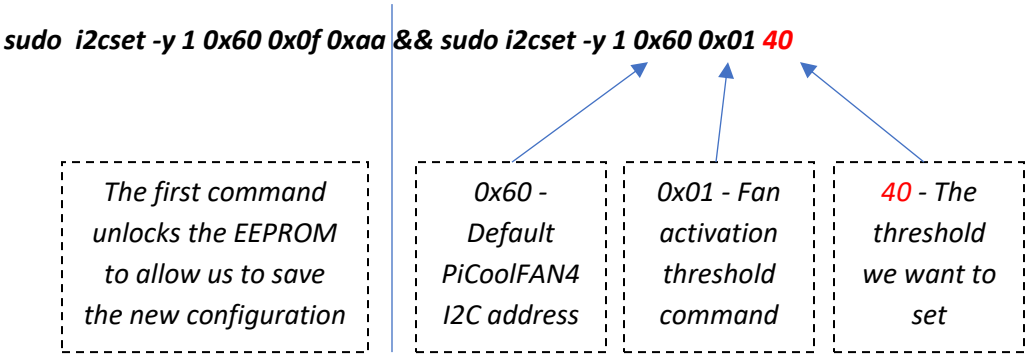
For ***example***, the Raspberry Pi temperature ***might*** be 60°C and the PCF4 temperature ***may*** be 50°C.

This indicates that the PCF4 sensor will likely always be around 10°C lower that the Raspberry Pi sensor – so to set a threshold for the Raspberry Pi sensor to 50°C, we would set the PCF4 threshold to 40°C.

To set a new threshold (whatever you decide), we use an I²C command, changing the red value to your desired temperature threshold:

sudo i2cset -y 1 0x60 0x0f 0xaa && sudo i2cset -y 1 0x60 0x01 40

We have broken down the command below so that you can understand what each part does:



This example is setting the threshold to 40°C, however the user can change this to whatever value is right for their requirements.

Remember, in this scenario based on the factory default setup, we're setting the PiCoolFAN4's temperature sensor reading for the threshold. Should you wish to use the Raspberry Pi's internal temperature sensor as the threshold sensor, see page 18.

Advanced Functionality

This section will guide the user in using the advanced functionality of the PiCoolFAN4, such as configuring the software to use the Raspberry Pi's temperature sensor, changing the cooling mode (hard or mild), adjusting the temperature threshold, fan speed changes and more.

Cooling Profiles

There are two type of cooling profile available to users:

Hard Cooling

A simple 'on/off' cooling profile.

The fan is initially off, and turns on at 100% speed when the temperature threshold is hit. When then temperature is lower than the threshold, it turns off again.

This profile is extremely effective at avoiding over overheating, however comes at the cost of fan noise.

Mild Cooling (recommended)

This cooling profile is more advanced in that it gradually speeds the fan up and down depending on how far the temperatures is from the threshold.

Users can adjust the settings of this profile to change the scale of the fan speed changes.

The Mild Cooling Profile is the recommend cooling method as it guarantees the lowest noise, best cooling and lowest current consumption, and as such, is the default profile setting from the factory.

Cooling Profile Parameters

Each Cooling Profile contains a set of three parameters which control how the profile reacts to temperature changes. These parameters are:

- Sensor and cooling profile selection (***Smode***)
- Temperature threshold (***Ttemp***)
- Fan speed change rate I.e. how the speed of the fan will change depending on temperature changes (***Tstep***)

Users can change any one (or all) of these parameters to customize cooling to their exact needs. All changes are stored in the EEPROM i.e. the settings are saved, even when the Raspberry Pi is turned off and on again.

This section will describe the different options for each parameter, then describe how to structure and issue this command to the Raspberry Pi.

Smode

System Mode (smode) defines both the sensor to be used and the cooling profile to be used with that sensor.

The following values are available:

smode value	Temperature Sensor	Cooling Profile
0x00	PiCoolFan4	Mild
0x02	Raspberry Pi	Mild
0x10	PiCoolFan4	Hard
0x12	Raspberry Pi	Hard
0x20	Manual System - User needs to start/stop the fan manually (via commands) and check the temperature manually	

Note: In order to use the Raspberry Pi temperature sensor, some software setup is required. Please see page 18.

Note: 0x00 is the factory default smode setting

Ttemp and Tstep

Ttemp defines the temperature threshold to be used in the cooling profile.

Tstep defines the number of steps (temperature in degrees) it takes to increase/decrease fan speed to the next range.

When used with a Hard cooling profile, Ttemp is simply the point at which the fan turns on/off. Tstep is not used with Hard cooling profiles.

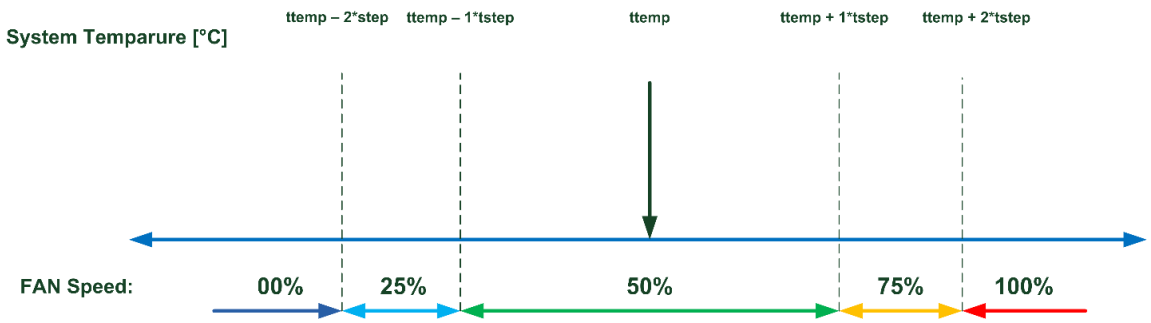
When used with a Mild cooling profile, the Ttemp is the center point at which the gradual fan changes are based on, which works in conjunction with Tstep.

In the example below, the ttemp is on the center of temperature monitoring window and **tstep** is set to 1.

When the temperature moves one degree lower or one degree higher than **ttemp, the fan runs at 50% of its speed (lower noise). This is because tstep is set to 1.**

If the temperature continuous to increase, the fan speed will increase 75% of its speed, as this is another 1 'step' as defined by tstep (which is set to 1). The fan will eventually run at 100% if the temperature increases by another step.

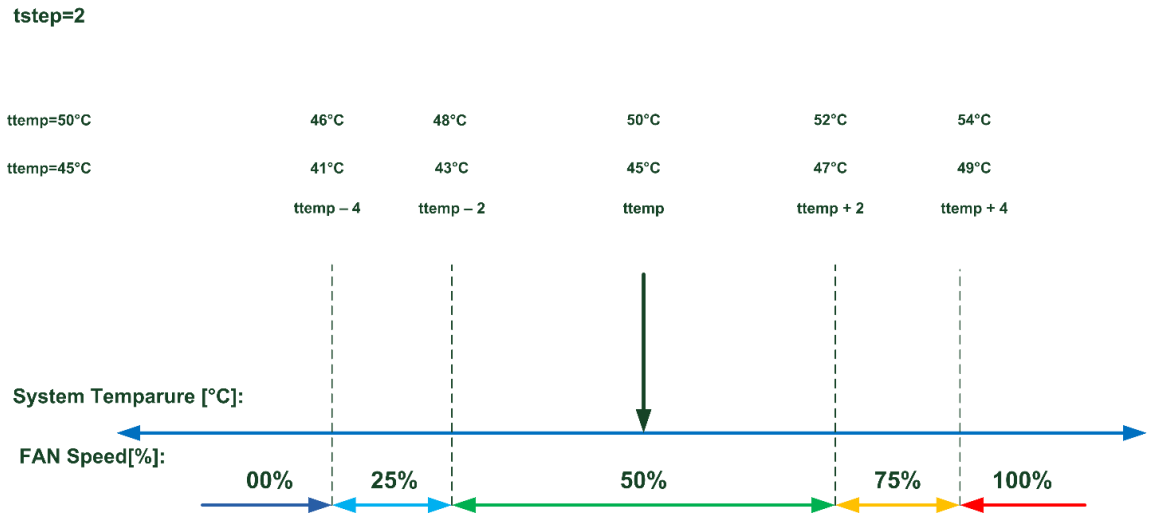
The same works for decreasing temperature with the same steps/range, lowering fan speed until it eventually turns off the fan when not required. This is ideal for minimizing noise output.



The user can change the ttemp to any value, effectively moving the window up or down in the temperature scale.

Tstep can also be changed, to increase/decrease the window of temperature increase/decrease before moving to the next fan speed.

Here is an example of Ttemp set to 45°C and Tstep set to 2:



How to set a Cooling Profile & Parameters

Each Cooling Profile parameter can be set using an I²C command, and users can choose to change one or many (or all) settings to meet their needs.

Most of the command is the same for each parameter, with just a different I²C address for each parameter entered at the end (along with the desired setting).

Setting Smode

Smode is set by using **0x08** in the I²C command followed by the desired **setting**:

sudo i2cset -y 1 0x60 0x0f 0xaa && sudo i2cset -y 1 0x60 0x08 0x10

In this example, the setting is 0x10 which uses the **PiCoolFAN4** sensor and a Hard cooling profile.

Other **smode** settings can be seen in the table on page 14.

Setting Ttemp

Ttemp is set by using **0x01** in the I²C command followed by the desired **temperature**:

sudo i2cset -y 1 0x60 0x0f 0xaa && sudo i2cset -y 1 0x60 0x01 45

In this example, we use 45 which sets the temperature threshold to 45 degrees.

Setting Tstep

Tstep is set by using **0x04** in the I²C command followed by the desired **setting**:

sudo i2cset -y 1 0x60 0x0f 0xaa && sudo i2cset -y 1 0x60 0x04 3

In this example, we are setting the step to 3. Step values permitted are 1, 2, 3, 4 or 5.