
Fix Intel CPU Throttling on Linux

This tool was originally developed to fix Linux CPU throttling issues affecting Lenovo T480 / T480s / X1C6 as described here.

The CPU package power limit (PL1/2) is forced to a value of **44 W** (29 W on battery) and the temperature trip point to **95 °C** (85 °C on battery) by overriding default values in MSR and MCHBAR every 5 seconds (30 on battery) to block the Embedded Controller from resetting these values to default.

On systems where the EC doesn't reset the values (ex: ASUS Zenbook UX430UNR), the power limit can be altered by using the official intel_rapl driver (see Static fix for more information)

Warning!

The latest commit (30 Oct 2021) switched from the legacy name `lenovo_fix` for the tool/config/system to a more uniform `throttled`. The install script was updated, but please report back if anything breaks.

Tested hardware

Other users have confirmed that the tool is also working for these laptops: - Lenovo T470s, T480, T480s, X1C5, X1C6, X1C8, T580, L590, L490, L480, T470, X280, X390, ThinkPad Anniversary Edition 25, E590 w/ RX 550X, P43s, E480, E580, T14 Gen 1, P14s Gen 1, T15 Gen 1, P15s Gen 1, E14 Gen 2, X1 Extreme Gen 4 - Dell XPS 9365, 9370, 9550, 7390 2-in-1, Latitude 7390 2-in-1, Inspiron 16 Plus 7620 - Microsoft Surface Book 2 - HP Probook 470 G5, Probook 450 G5, ZBook Firefly 15 G7

I will keep this list updated.

Is this tool really doing something on my PC??

I suggest you to use the excellent **s-tui** tool to check and monitor the CPU usage, frequency, power and temperature under load!

Undervolt

The tool supports **undervolting** the CPU by configuring voltage offsets for CPU, cache, GPU, System Agent and Analog I/O planes. The tool will re-apply undervolt on resume from standby and hibernate by listening to DBus signals. You can now either use the `UNDERVOLT` key in config to set global values

or the `UNDERVOLT.AC` and `UNDERVOLT.BATTERY` keys to selectively set undervolt values for the two power profiles.

===== **Notice that undervolt is typically locked from 10th gen onwards!** =====

IccMax (EXPERTS ONLY)

The tool now supports overriding the **IccMax** by configuring the maximum allowed current for CPU, cache and GPU planes. The tool will re-apply IccMax on resume from standby and hibernate. You can now either use the `ICCMAX` key in config to set global values or the `ICCMAX.AC` and `ICCMAX.BATTERY` keys to selectively set current values for the two power profiles. **NOTE:** the values specified in the config file are the actual current limit of your system, so those are not a offset from the default values as for the undervolt. As such, you should first find your system default values with the `--monitor` command.

HWP override (EXPERIMENTAL)

I have found that under load my CPU was not always hitting max turbo frequency, in particular when using one/two cores only. For instance, when running prime95 (1 core, test #1) my CPU is limited to about 3500 MHz over the theoretical 4000 MHz maximum. The reason is the value for the HWP energy performance hints. By default TLP sets this value to `balance_performance` on AC in order to reduce the power consumption/heat in idle. By setting this value to `performance` I was able to reach 3900 MHz in the prime95 single core test, achieving a +400 MHz boost. Since this value forces the CPU to full speed even during idle, a new experimental feature allows to automatically set HWP to performance under load and revert it to balanced when idle. This feature can be enabled (in AC mode *only*) by setting to `True` the `HWP_Mode` parameter in the throttled config file : <https://github.com/erpalma/throttled/blob/master/etc/throttled.conf#L41> .

I have run **Geekbench 4** and now I can get a score of 5391/17265! On `balance_performance` I can reach only 4672/16129, so **15% improvement** in single core and 7% in multicore, not bad ;)

setting cTDP (EXPERIMENTAL)

On a lot of modern CPUs from Intel one can configure the TDP up or down based on predefined profiles. This is what this option does. For a i7-8650U normal would be 15W, up profile is setting it to 25W and down to 10W. You can lookup the values of your CPU at the Intel product website.

Requirements

A stripped down version of the python module `python-periphery` is now built-in and it is used for accessing the MCHBAR register by memory mapped I/O. You also need `dbus` and `gobject` python bindings for listening to dbus signals on resume from sleep/hibernate.

Writing to MSR and PCI BAR

Some time ago a feature called Kernel Lockdown was added to Linux. Kernel Lockdown automatically enables some security measures when Secure Boot is enabled, among them restricted access to MSR and PCI BAR via `/dev/mem`, which this tool requires. There are two ways to get around this: You can either disable Secure Boot in your firmware settings, or disable the Kernel Lockdown LSM.

The LSM can be disabled this way: Check the contents of the file `/sys/kernel/security/lsm` (example contents: `capability,lockdown,yama`). Take the contents of the file, remove `lockdown` and add the rest as a kernel parameter, like this: `lsm=capability,yama`. Reboot and Kernel Lockdown will be disabled!

As of Linux 5.9, kernel messages will be logged whenever the script writes to MSR registers. These aren't a problem for now, but there's some indication that future kernels may restrict MSR writes from userspace by default. This is being tracked by issue #215. The messages will look something like:

```
1 [ 324.833543] msr: Write to unrecognized MSR 0x1a2 by python3
2               Please report to x86@kernel.org
```

Note that some kernels (e.g. linux-hardened) will prevent from writing to `/dev/mem` too. Specifically, you need a kernel with `CONFIG_DEVMEM` and `CONFIG_X86_MSR` set.

Thermald

As discovered by *DEvil0000* the Linux Thermal Monitor (thermald) can conflict with the purpose of this tool. In particular, thermald might be pre-installed (e.g. on Ubuntu) and configured in such a way to keep the CPU temperature below a certain threshold (~80 °C) by applying throttling or messing up with RAPL or other CPU-specific registers. I strongly suggest to either disable/uninstall it or to review its default configuration.

Update

The tool is now running with Python3 by default (tested w/ 3.6) and a virtualenv is automatically created in `/opt/throttled`. Python2 should probably still work.

Installation

Arch Linux community package:

```
1 pacman -S throttled
2 sudo systemctl enable --now throttled.service
```

Thanks to *felixonmars* for creating and maintaining this package.

Artix Linux

```
1 makepkg -si
2 sudo rc-update add throttled default
3 sudo rc-service throttled start
```

Debian/Ubuntu

```
1 sudo apt install git build-essential python3-dev libdbus-glib-1-dev
   libgirepository1.0-dev libcairo2-dev python3-cairo-dev python3-venv
   python3-wheel
2 git clone https://github.com/erpalma/throttled.git
3 sudo ./throttled/install.sh
```

If you own a X1C6 you can also check a tutorial for Ubuntu 18.04 [here](#).

You should make sure that **thermald** is not setting it back down. Stopping/disabling it will do the trick:

```
1 sudo systemctl stop thermald.service
2 sudo systemctl disable thermald.service
```

If you want to keep it disabled even after a package update you should also run:

```
1 sudo systemctl mask thermald.service
```

In order to check if the service is well started, you could run:

```
1 systemctl status throttled
```

Fedora

A copr repository is available and can be used as detailed below. You can find the configuration installed at `/etc/throttled.conf`. The issue tracker for this packaging is available [here](#).

```
1 sudo dnf copr enable abn/throttled
2 sudo dnf install -y throttled
3
4 sudo systemctl enable --now throttled
```

If you prefer to install from source, you can use the following commands.

```
1 sudo dnf install python3-cairo-devel cairo-gobject-devel gobject-
  introspection-devel dbus-glib-devel python3-devel make libX11-devel
2 git clone https://github.com/erpalma/throttled.git
3 sudo ./throttled/install.sh
```

Feedback about Fedora installation is welcome.

Fedora Silverblue

Download the `.repo` file matching your Fedora on the copr repository page then copy it to `/etc/yum.repos.d/`.

You can then install the package:

```
1 rpm-ostree override remove thermald
2 rpm-ostree install throttled
3 systemctl reboot
4
5 sudo systemctl enable --now throttled
```

openSUSE

User *brycecordill* reported that the following dependencies are required for installing in openSUSE, tested on openSUSE 15.0 Leap.

```
1 sudo zypper install gcc make python3-devel dbus-1-glib-devel python3-
  cairo-devel cairo-devel python3-gobject-cairo gobject-introspection-
  devel
2 git clone https://github.com/erpalma/throttled.git
3 sudo ./throttled/install.sh
```

Gentoo

The ebuild is now in the official tree!

```
1 sudo emerge -av sys-power/throttled
2 # when using OpenRC:
```

```
3 rc-update add throttled default
4 /etc/init.d/throttled start
5 # when using systemd:
6 systemctl enable --now throttled.service
```

Solus

```
1 sudo eopkg it -c system.devel
2 sudo eopkg it git python3-devel dbus-glib-devel python3-cairo-devel
   libcairo-devel python3-gobject-devel
3 git clone https://github.com/erpalma/throttled.git
4 sudo ./throttled/install.sh
```

Void

The installation itself will create a runit service as throttled, enable it and start it. Before installation, make sure dbus is running `sv up dbus`.

```
1 sudo xbps-install -Sy gcc git python3-devel dbus-glib-devel
   libgirepository-devel cairo-devel python3-wheel pkg-config make
2
3 git clone https://github.com/erpalma/throttled.git
4
5 sudo ./throttled/install.sh
```

Uninstall

To permanently stop and disable the execution just issue:

```
1 systemctl stop throttled.service
2 systemctl disable throttled.service
```

If you're running runit instead of systemd:

```
1 sv down throttled
2 rm /var/service/throttled
```

If you're using OpenRC instead of systemd:

```
1 rc-service throttled stop
2 rc-update del throttled default
```

If you also need to remove the tool from the system:

```
1 rm -rf /opt/throttled /etc/systemd/system/throttled.service
2 # to purge also the config file
3 rm /etc/throttled.conf
```

On Arch you should probably use `pacman -R lenovo-throttling-fix-git` instead.

Update

If you update the tool you should manually check your config file for changes or additional features and modify it accordingly. The update process is then as simple as:

```
1 cd throttled
2 git pull
3 sudo ./install.sh
4 sudo systemctl restart throttled.service
5 OpenRC: sudo rc-service throttled restart
```

Configuration

The configuration has moved to `/etc/throttled.conf`. Makefile does not overwrite your previous config file, so you need to manually check for differences in config file structure when updating the tool. If you want to overwrite the config with new defaults just issue `sudo cp etc/throttled.conf /etc`. There exist two profiles `AC` and `BATTERY` and the tool can be totally disabled by setting `Enabled: False` in the `GENERAL` section. Undervolt is applied if any voltage plane in the config file (section `UNDERVOLT`) was set. Notice that the offset is in *mV* and only undervolting (*i.e.* negative values) is supported. All fields accept floating point values as well as integers.

My T480s with i7-8550u is stable with:

```
1 [UNDERVOLT]
2 # CPU core voltage offset (mV)
3 CORE: -105
4 # Integrated GPU voltage offset (mV)
5 GPU: -85
6 # CPU cache voltage offset (mV)
7 CACHE: -105
8 # System Agent voltage offset (mV)
9 UNCORE: -85
10 # Analog I/O voltage offset (mV)
11 ANALOGIO: 0
```

IMPORTANT: Please notice that *my* system is stable with these values. Your notebook might crash even with slight undervolting! You should test your system and slowly increasing undervolt to find the maximum stable value for your CPU. You can check this tutorial if you don't know where to start.

Monitoring

With the flag `--monitor` the tool *constantly* monitors the throttling status, indicating the cause among thermal limit, power limit, current limit or cross-origin. The last cause is often related to an external event (e.g. by the GPU). The update rate can be adjusted and defaults to 1 second. Example output:

```
1 ./throttled.py --monitor
2 [I] Detected CPU architecture: Intel Kaby Lake (R)
3 [I] Loading config file.
4 [I] Starting main loop.
5 [D] Undervolt offsets: CORE: -105.00 mV | GPU: -85.00 mV | CACHE:
   -105.00 mV | UNCORE: -85.00 mV | ANALOGIO: 0.00 mV
6 [D] IccMax: CORE: 64.00 A | GPU: 31.00 A | CACHE: 6.00 A
7 [D] Realtime monitoring of throttling causes:
8
9 [AC] Thermal: OK - Power: OK - Current: OK - Cross-domain (e.g. GPU):
   OK || VCore: 549 mV - Package: 2.6 W - Graphics: 0.4 W - DRAM: 1.2
   W
```

Static Fix

You can alternatively set the power limits using intel_rapl driver (modifying MCHBAR values requires Linux 5.3+). Bear in mind, some embedded controllers (EC) control the power limit values and will reset them from time to time):

```
1 # MSR
2 # PL1
3 echo 440000000 | sudo tee /sys/devices/virtual/powercap/intel-rapl/intel-
   -rapl:0/constraint_0_power_limit_uw # 44 watt
4 echo 280000000 | sudo tee /sys/devices/virtual/powercap/intel-rapl/intel-
   -rapl:0/constraint_0_time_window_us # 28 sec
5 # PL2
6 echo 440000000 | sudo tee /sys/devices/virtual/powercap/intel-rapl/intel-
   -rapl:0/constraint_1_power_limit_uw # 44 watt
7 echo 2440 | sudo tee /sys/devices/virtual/powercap/intel-rapl/intel-
   -rapl:0/constraint_1_time_window_us # 0.00244 sec
8
9 # MCHBAR
10 # PL1
11 echo 440000000 | sudo tee /sys/devices/virtual/powercap/intel-rapl-mmio/
   intel-rapl-mmio:0/constraint_0_power_limit_uw # 44 watt
12 # ^ Only required change on a ASUS Zenbook UX430UNR
13 echo 280000000 | sudo tee /sys/devices/virtual/powercap/intel-rapl-mmio/
   intel-rapl-mmio:0/constraint_0_time_window_us # 28 sec
14 # PL2
```

```
15 echo 44000000 | sudo tee /sys/devices/virtual/powercap/intel-rapl-mmio/
   intel-rapl-mmio:0/constraint_1_power_limit_uw # 44 watt
16 echo 2440 | sudo tee /sys/devices/virtual/powercap/intel-rapl-mmio/
   intel-rapl-mmio:0/constraint_1_time_window_us # 0.00244 sec
```

If you want to change the values automatic on boot you can use systemd-tmpfiles:

```
1 # /etc/tmpfiles.d/power_limit.conf
2 # MSR
3 # PL1
4 w /sys/devices/virtual/powercap/intel-rapl/intel-rapl:0/
   constraint_0_power_limit_uw - - - - 44000000
5 w /sys/devices/virtual/powercap/intel-rapl/intel-rapl:0/
   constraint_0_time_window_us - - - - 28000000
6 # PL2
7 w /sys/devices/virtual/powercap/intel-rapl/intel-rapl:0/
   constraint_1_power_limit_uw - - - - 44000000
8 w /sys/devices/virtual/powercap/intel-rapl/intel-rapl:0/
   constraint_1_time_window_us - - - - 2440
9
10 # MCHBAR
11 # PL1
12 w /sys/devices/virtual/powercap/intel-rapl-mmio/intel-rapl-mmio:0/
   constraint_0_power_limit_uw - - - - 44000000
13 # ^ Only required change on a ASUS Zenbook UX430UNR
14 w /sys/devices/virtual/powercap/intel-rapl-mmio/intel-rapl-mmio:0/
   constraint_0_time_window_us - - - - 28000000
15 # PL2
16 w /sys/devices/virtual/powercap/intel-rapl-mmio/intel-rapl-mmio:0/
   constraint_1_power_limit_uw - - - - 44000000
17 w /sys/devices/virtual/powercap/intel-rapl-mmio/intel-rapl-mmio:0/
   constraint_1_time_window_us - - - - 2440
```

Debug

You can enable the `--debug` option to read back written values and check if the tool is working properly. At the startup it will also show the CPUs platform info which contains information about multiplier values and features present for this CPU. Additionally the tool will print the thermal status per core which is handy when it comes to figuring out the reason for CPU throttle. Status fields stands for the current throttle reason or condition and log shows if this was a throttle reason since the last interval. This is an example output:

```
1 ./throttled.py --debug
2 [D] cpu platform info: maximum non turbo ratio = 20
3 [D] cpu platform info: maximum efficiency ratio = 4
4 [D] cpu platform info: minimum operating ratio = 4
5 [D] cpu platform info: feature ppin cap = 0
```

```

6 [D] cpu platform info: feature programmable turbo ratio = 1
7 [D] cpu platform info: feature programmable tdp limit = 1
8 [D] cpu platform info: number of additional tdp profiles = 2
9 [D] cpu platform info: feature programmable temperature target = 1
10 [D] cpu platform info: feature low power mode = 1
11 [D] TEMPERATURE_TARGET - write 0xf - read 0xf
12 [D] Undervolt plane CORE - write 0xf2800000 - read 0xf2800000
13 [D] Undervolt plane GPU - write 0xf5200000 - read 0xf5200000
14 [D] Undervolt plane CACHE - write 0xf2800000 - read 0xf2800000
15 [D] Undervolt plane UNCORE - write 0xf5200000 - read 0xf5200000
16 [D] Undervolt plane ANALOGIO - write 0x0 - read 0x0
17 [D] MSR PACKAGE_POWER_LIMIT - write 0xcc816000dc80e8 - read 0
    xcc816000dc80e8
18 [D] MCHBAR PACKAGE_POWER_LIMIT - write 0xcc816000dc80e8 - read 0
    xcc816000dc80e8
19 [D] TEMPERATURE_TARGET - write 0xf - read 0xf
20 [D] core 0 thermal status: thermal throttle status = 0
21 [D] core 0 thermal status: thermal throttle log = 1
22 [D] core 0 thermal status: prochlor or forcepr event = 0
23 [D] core 0 thermal status: prochlor or forcepr log = 0
24 [D] core 0 thermal status: crit temp status = 0
25 [D] core 0 thermal status: crit temp log = 0
26 [D] core 0 thermal status: thermal threshold1 status = 0
27 [D] core 0 thermal status: thermal threshold1 log = 1
28 [D] core 0 thermal status: thermal threshold2 status = 0
29 [D] core 0 thermal status: thermal threshold2 log = 1
30 [D] core 0 thermal status: power limit status = 0
31 [D] core 0 thermal status: power limit log = 1
32 [D] core 0 thermal status: current limit status = 0
33 [D] core 0 thermal status: current limit log = 0
34 [D] core 0 thermal status: cross domain limit status = 0
35 [D] core 0 thermal status: cross domain limit log = 0
36 [D] core 0 thermal status: cpu temp = 44
37 [D] core 0 thermal status: temp resolution = 1
38 [D] core 0 thermal status: reading valid = 1
39 .....

```

Autoreload

Auto reload config on changes (unless it's deleted) can be enabled/disabled in the config

```

1 [General]
2 Autoreload = True

```

A word about manufacturer provided tooling

Tools provided by your notebook manufacturer like Dell Power Manager tend to persist their settings to the system board. If you ever had it running under Windows and activated a cool/quiet/silent/saving profile, this setting will still be active when running linux, throttling your system.

On my Dell Latitude 5591, not even a BIOS reset to manufacturer default killed the active **Quiet** profile

Disclaimer

This script overrides the default values set by Lenovo. I'm using it without any problem, but it is still experimental so use it at your own risk.