Making gadgets really cool

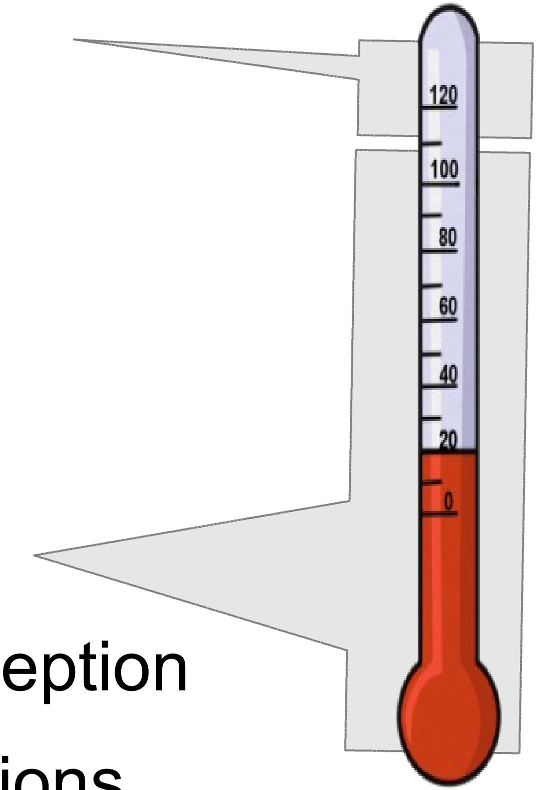
Noor ul Mubeen
Intel Corporation

Agenda / Focus

- Handheld smartphones & tablets
- Running Android Linux OS
- passive cooling software solution
- “Drawing board” focus but,
- builds on Linux Thermal framework
  - Some upstreamed
    - some in review
      - *some in making*.

Response to Thermal

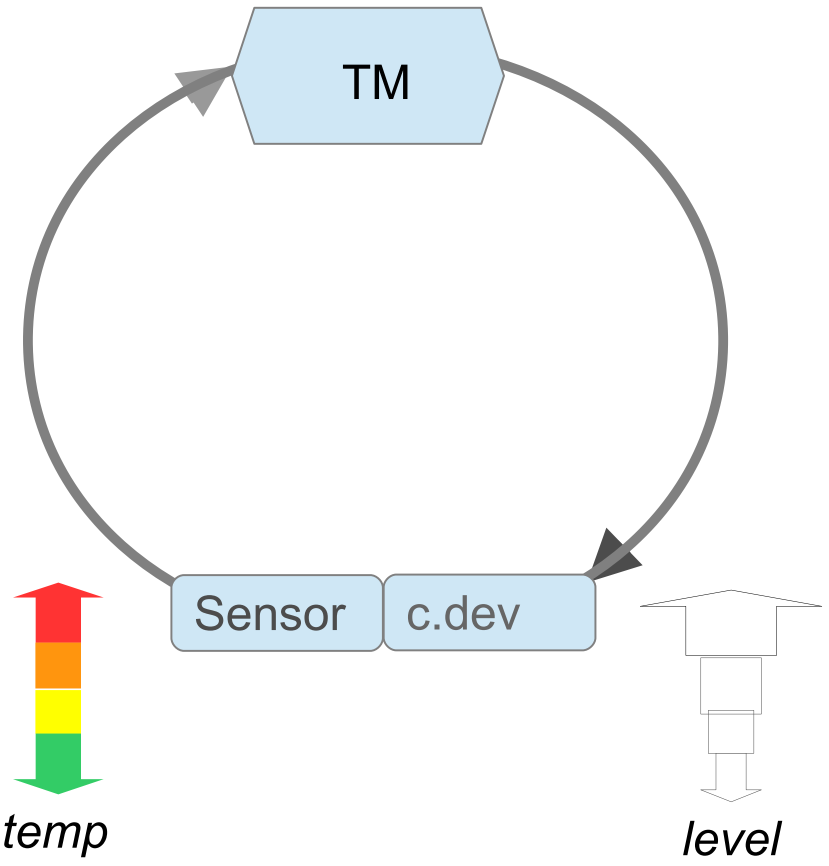
- User/Device safety range:
  - Extreme die temp > 100 °C
  - SoC, charger IC, PMIC etc.
  - Action: easy, shutdown!
  - Generally hardware assisted
- User comfort range (say > 35°C)
  - Importantly for user comfort/perception
  - Achieved by complex throttle actions
  - Defers cirtical actions



Throttle: negative feedback loop

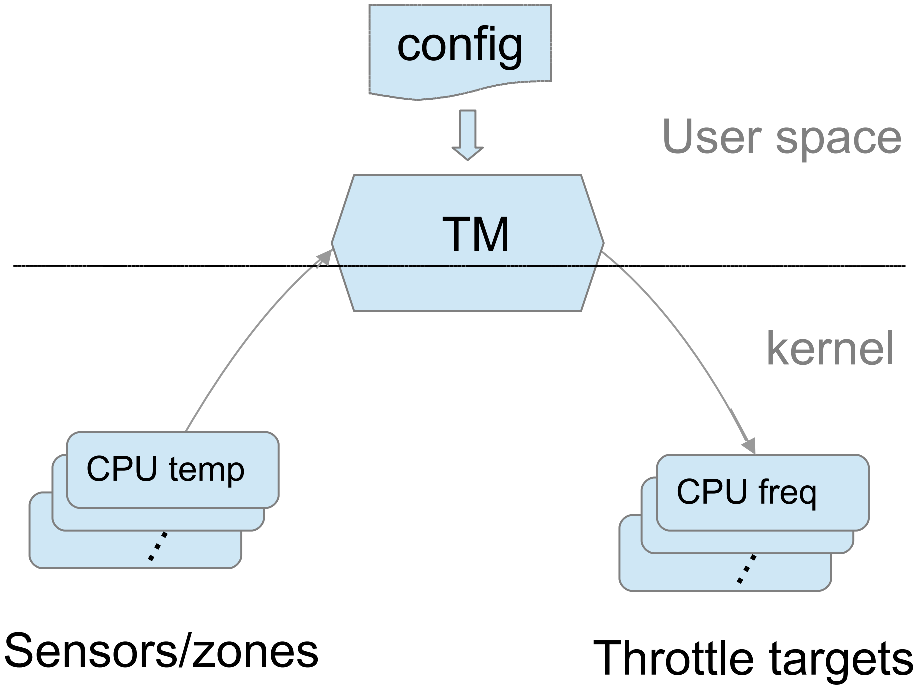
In principle:

- Sensor
  - In a zone abstract
- Throttle targets
  - a.k.a cooling dev
- TM Software
  - completes the loop
  - Controls / policy



TM framework

- Thermal zones:
  - Battery, CPU, GPU...
- Throttle targets:
  - CPU-freq
  - GPU-freq
  - Display brightness
  - Charge current...
- TM policy

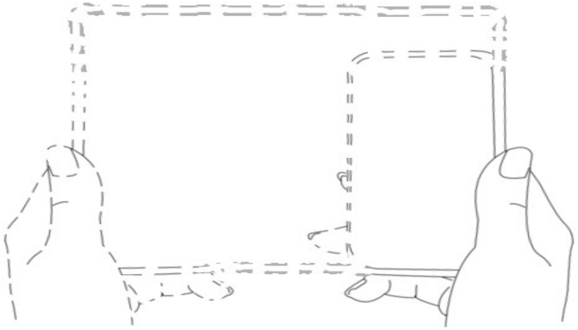


```
thermal_sys.c
    thermal_zone_device_register()
    thermal_cooling_device_register()
    thermal_zone_bind_cooling_device()
```

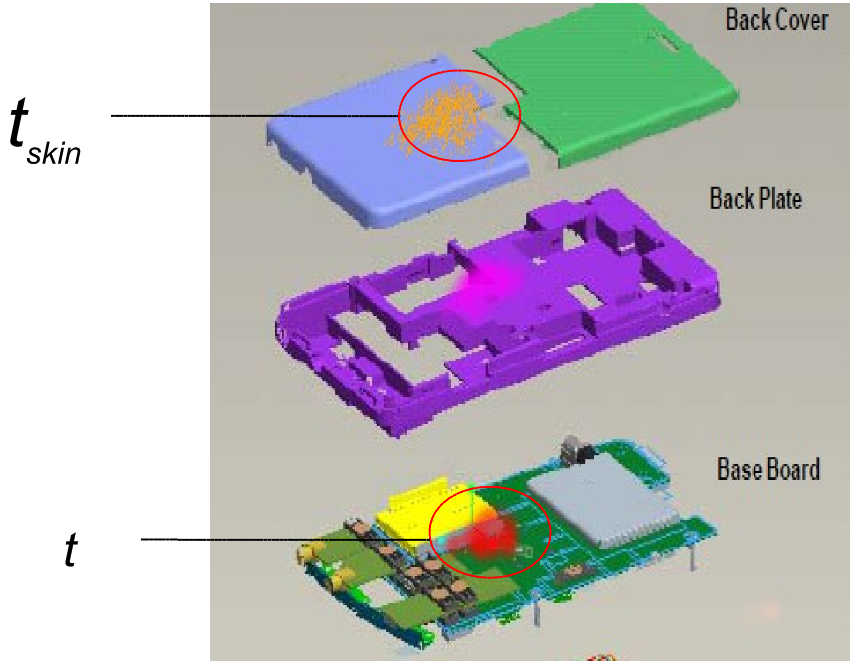
Skin zone temperature

User perception:

- Broad range, say > 40°C
- multiple touch points
- Including display side ear piece
- Net effect of different components on Skin
  - virtual skin sensor needed.



Tskin virtual sensor



$$t_{skin} = f(t)$$

Where f() is,

- Evaluated for given FF
- Prefer a simple formula
- Which gives least errors in operating temp. range
- Details beyond scope, but math is obvious.

Tskin virtual sensor…

T_{skin}

Sensors/zones

Throttle targets

- Plug into feedback loop
- Calculate f(t), apply throttle policy as applicable.
- For User space solution: via config file
- kernel space solution: via platform driver that knows the constituent sensors.

TM Policy

CPU temp

M

Sensors/zones

CPU freq

N

Throttle targets

- Mapping zone/sensor <-> cdev

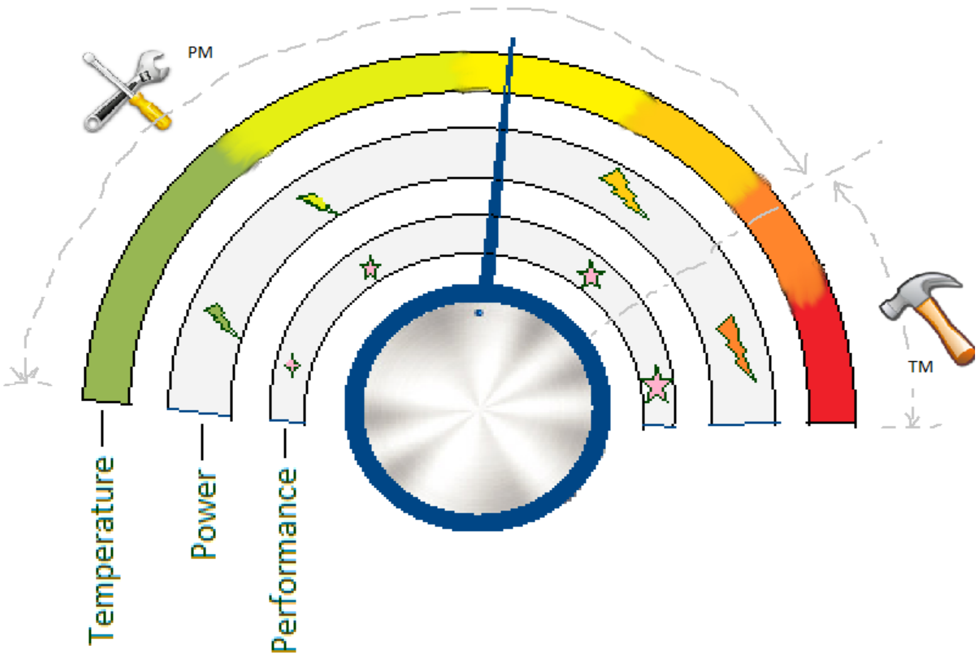
m:n where $(m \subset M)$ *to* $(n \subset N)$

    - Userspace policy
    - Kernel based policy

- Apply sensor weights by means of trip points
- Apply proportions by exposing correct step from cdev
- No golden rule*. Evaluate for given Form factor.

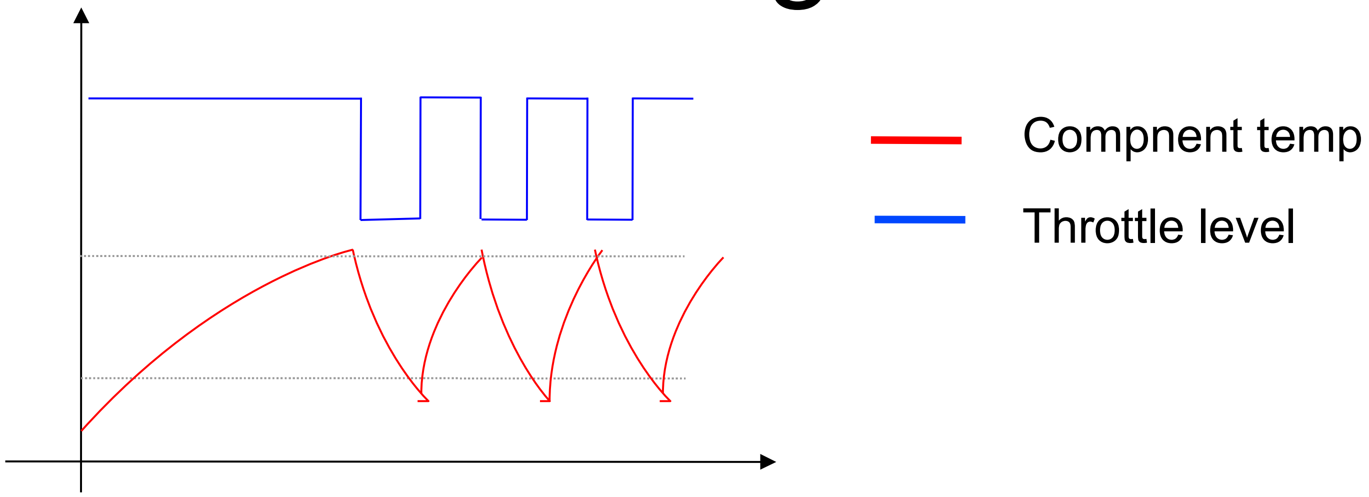
*But conditions apply.

The 3 scale knob



- beware of the knob [config file/trip points]
- more than one scale underneath the needle
- Power throttle: proactive spanner (any normal day)
- Thermal throttle: reactive hammer (abnormal temp.)
- Example: Characterize these over Component specific benchmark.

Classic “thrashing” issues



- some workloads (benchmarks) can get temp & throttle levels in lock step: thrashing each other.
- Add new throttle substate ?
- For every given substate band, there exists a...
- Simplistic light weight solution preferred for abnormal reactive problems!
- Unless perf is the only thing to address ;)

Thank you

Ack

Linux TM maintainers: Zhang Rui, Len Brown
TM Gurus: Hari, Ramesh
Contributors: Sujith, Durgadoss