Linux Thermal Daemon

V0.6

03/15/13

Contents

Abstract

The demand for high performance desktop/laptop systems resulted in higher power dissipation. This problem is compounded by increase in demand for small form factor systems. This is increasingly difficult to deal performance and thermal in isolation. This is important to deal with thermal issues without significantly impacting performance proactively.

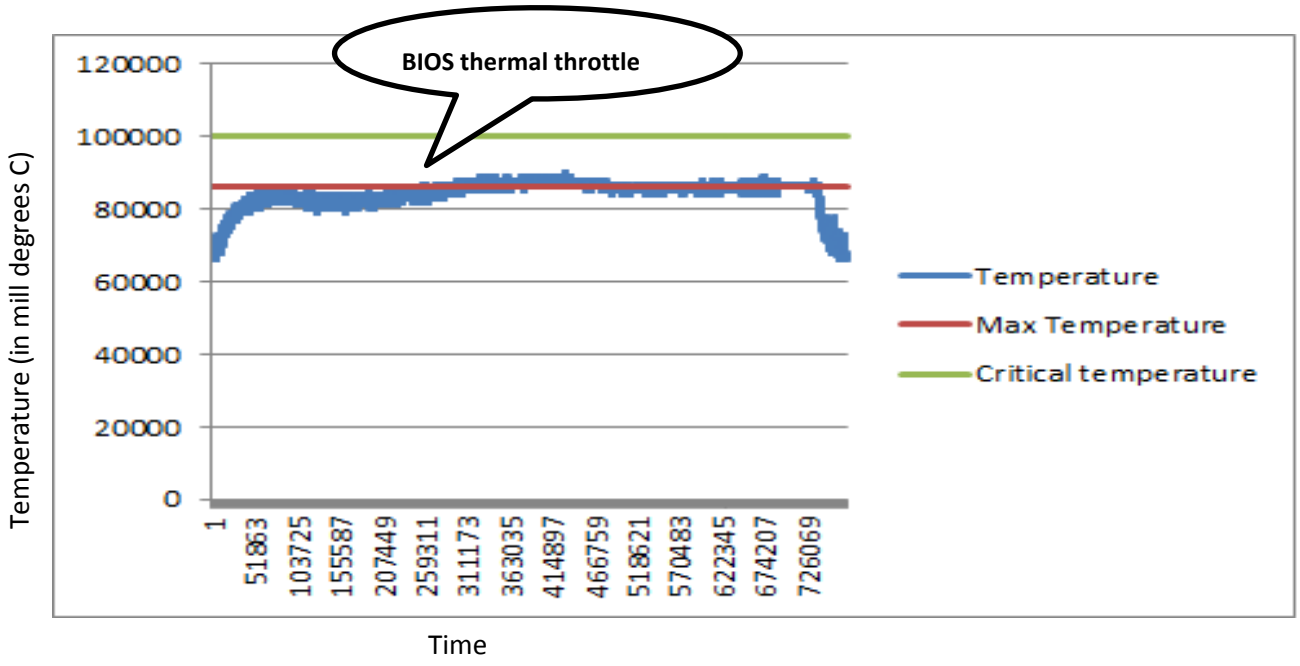
The solution used in this document is to develop a thermal daemon, which proactively control thermal using P-states, T-states and using Intel power clamp driver.

Problem Statement / Introduction

As processor executes at higher clock speeds, dynamic power consumption increases. The heat generated by this must be efficiently dissipated to improve reliability of the system. Data sheets normally contain a TDP (Thermal design power), which is the maximum amount of power the cooling system is required to dissipate. Common techniques for heat dissipation can include heat sinks, fans and other form of cooling devices. But as the form factor of systems reduces, it is not efficient to just rely on hardware/BIOS/OSPM to cool the system. This problem will be more eminent as fan less systems becomes more popular.

Based on experiments on small form factor devices, it was observed that systems reach near maximum temperature with relatively less load. This also can cause performance issues depending on the efficiency of cooling method used in the BIOS.

After detailed analysis of the problem, it was observed that increase in CPU temperature is causing BIOS to do thermal throttling, which is impacting performance.



Temperature trend in the problem systems

Current thermal control methods

Current systems use two approaches:

- Reactive approach
    - Bios starts taking action once it reaches certain temperature
    - ACPI thermal configuration driving cooling device activation
- Proactive approach
    - Using ACPI thermal configuration data proactively activate the configured cooling devices

Bios Approach

Bios can control FAN in the system. Based on temperature it can increase or decrease fan speed. It also can do thermal throttling. Thermal throttling is done by adjusting duty cycle of the processor clock or reduces the operating frequency and voltage. These controls can greatly impact performance.

ACPI Active and Passive Control

ACPI defines configuration and interfaces that allow OSPM to implement thermal control. Current Linux kernel thermal ACPI module, implements these controls. So based on the validity of configuration data, this can be a very efficient method for thermal controls. But it was observed that many systems don't have this configuration data or invalid data, preventing kernel module to take timely action.

Proposed Solution

The main objectives while considering possible solutions are:

- Prevent BIOS from thermal throttling as much as possible by proactively controls
- Don't rely on validity of ACPI configuration data. If there is a good valid data, the existing Linux kernel module will take necessary actions anyway.

This solution defines a Linux user mode daemon "thermal daemon", which provides a

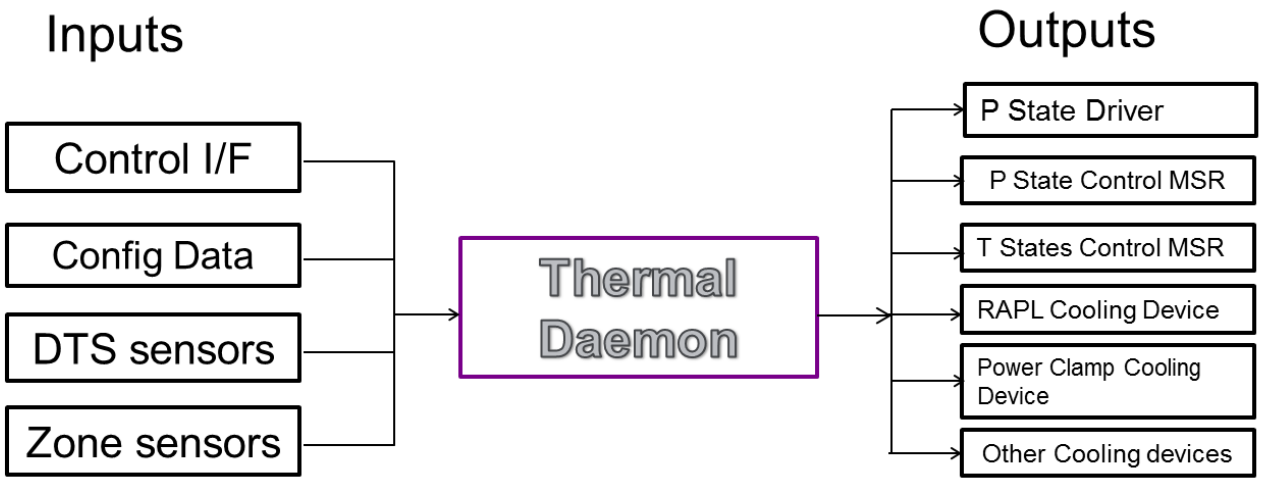
- Short time to market
- Proactively control temperature
- Use existing kernel infra-structure
- Defines an architecture, which can be easily enhanced.

Thermal Daemon Overview

The primary function of this daemon:

- Use existing I/F to read temperature sensors
- Calculate set point dynamically
- Alternatively use pre-configured or stored set points
- Once set point is reached use the best cooling method
- Allow user to set preferences using DBUS I/F

Block Diagram



Input to the system

Configuration data

The idea is not to rely on any configuration data. But if someone decides to tune a particular system, an ACPI style configuration can be specified using an XML file. The system matching is done by DMI UUID. Refer to thermal-conf.xml for DTD and example.

DTS Sensors

DTS sensors refer to digital temperature sensors. The output of DTS sensors can be read from /sys/devices/platform/coretemp.x interface.

Zone sensors

Zone sensors can be read from /sys/class/thermal/ interface. These define temperature sensors using ACPI style configuration data. By default this daemon doesn't use these sensors unless specified in XML configuration.

Outputs from the system

P States

P states refer to performance states. This daemon uses either Intel P state driver, if available or MSRs defined in Intel Software Architecture document to control p-states. The advantage of using MSRs is more fine grain control for turbo states. If the system can't be cooled by downgrading turbo state, turbo is disengaged and non-turbo frequencies are used to control thermal.

RAPL Cooling device

If there is a RAPL (Running average power limit) cooling device driver is present, it will use it to set limits to reduce temperature.

Power Clamp driver

Once P states are not enough to control, it used Intel power clamp driver to cool the system. This uses idle injections to cool the system.

T States

Using T states, clock duty cycle can be modified. This daemon uses T states control using MSRs, once the system can't be cooled by any other method.

Cooling devices

If the XML configuration defines ACPI style cooling device paths, this daemon activates cooling devices once on reaching configured set point.

Outputs control order

Once temperature needs to be controlled, daemon starts activating the cooling devices using the following order:

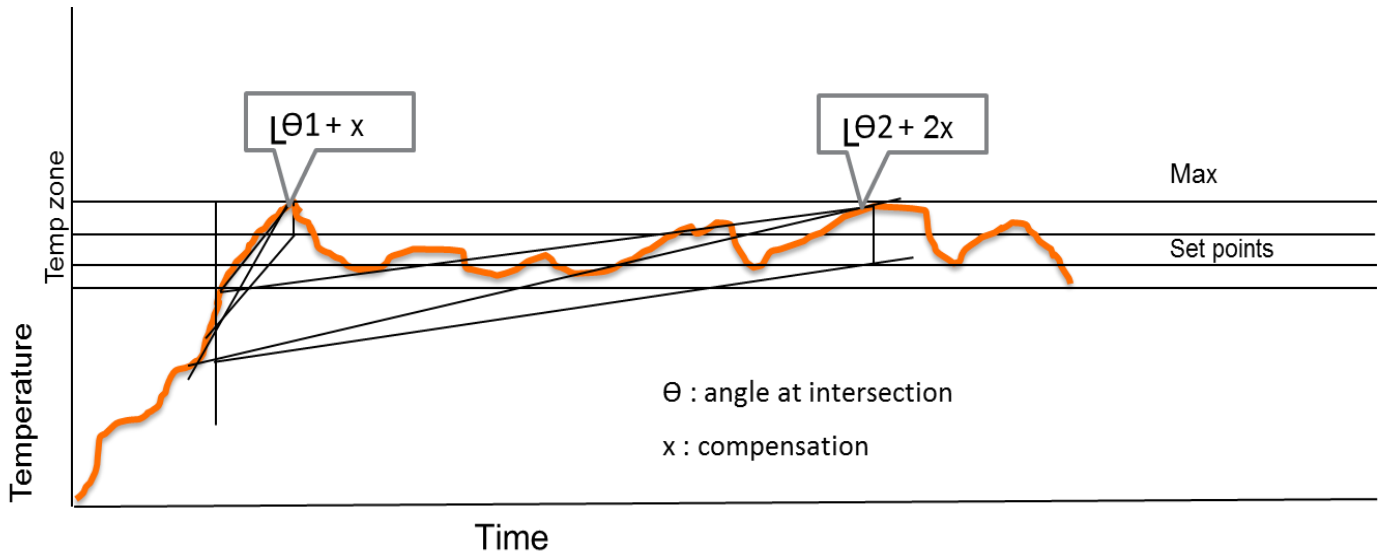
- RAPL cooling device driver
- P states control
    - Intel P state driver
      OR
    - Turbo sub states
    - Disengage turbo
    - Traverse half of the P states
- Intel power clamp driver
- T states

Dynamic set point adjustment

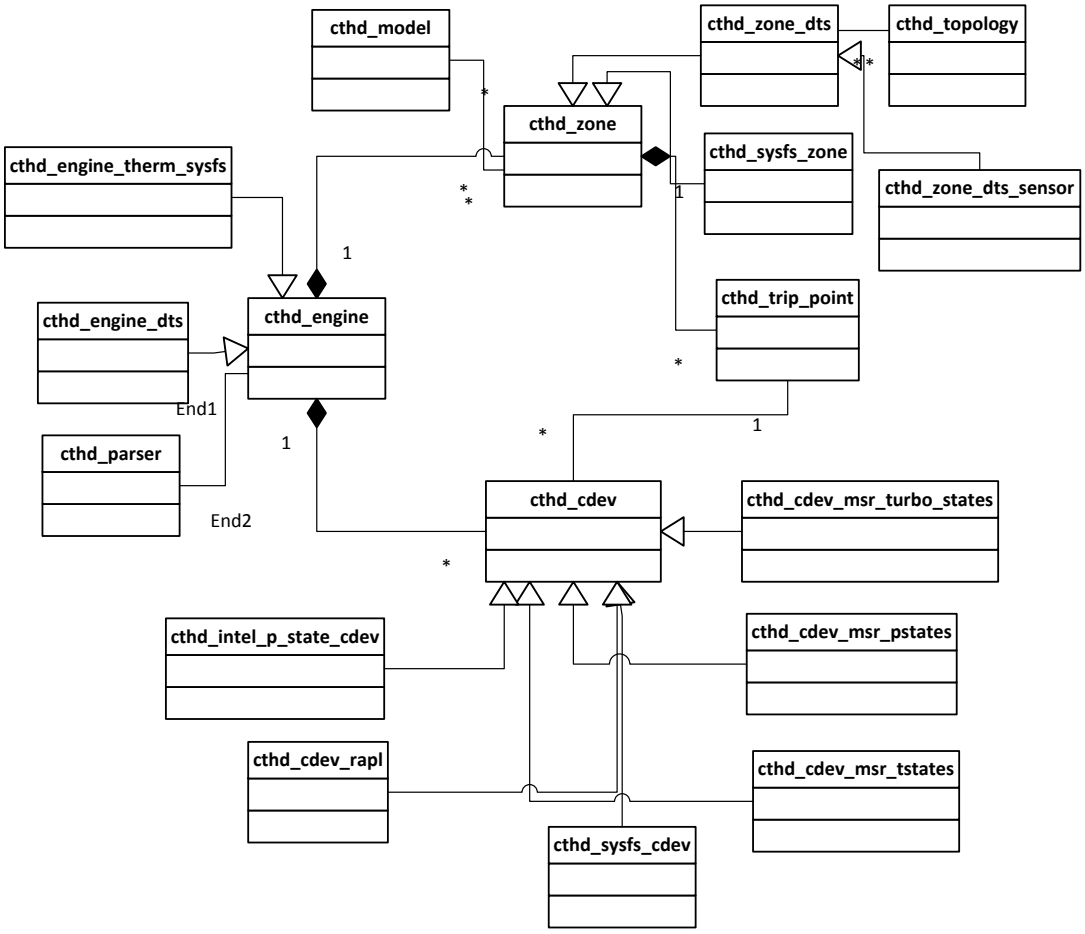
Set point refers to a temperature when a cooling action is activated. This daemon uses the maximum temperature, which can be read from /sys/devices/platform/coretemp.x as a reference. Very first time when the system reaches this temperature, it calculates a set point, at which cooling action starts on subsequent times.

This is based on

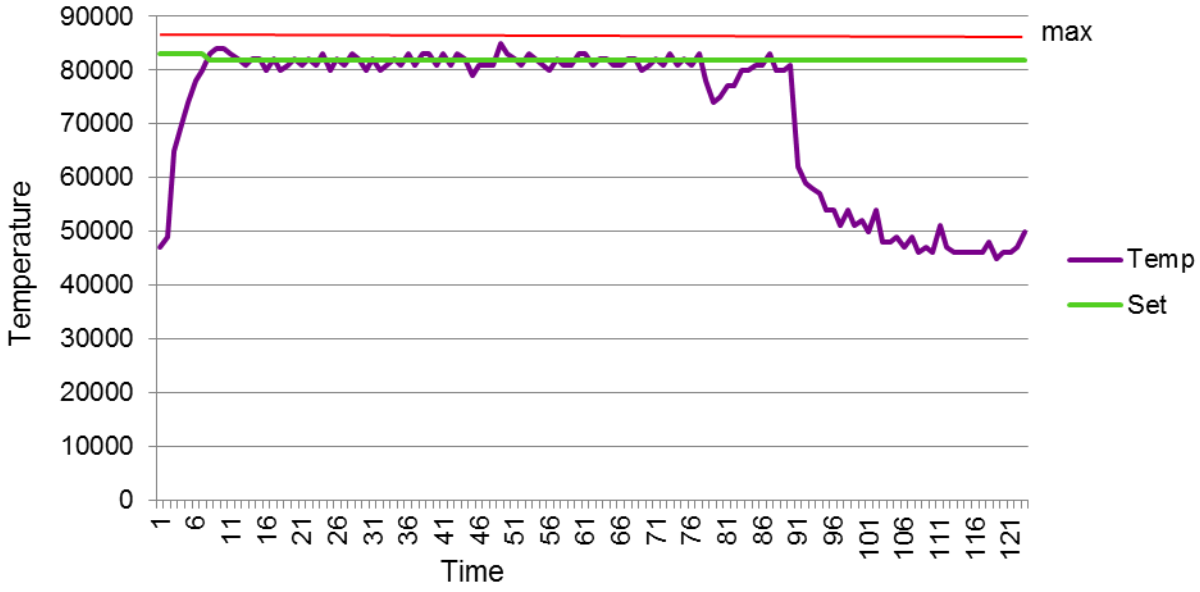
- Newton's law of cooling: Rate of heat transfer will reduce over time
- Longer in high temperature zone, more correction is required
- Account for time and slope relationship between temperature and required cooling



Software Architecture (class diagram)



Results


90000
80000
70000
60000
50000
40000
30000
20000
10000
0
Temperature
1
6
11
16
21
26
31
36
41
46
51
56
61
66
71
76
81
86
91
96
101
106
111
116
121
Time
max
Temp
Set