Summer Internship Diary 2020

Thermal Aware Adaptive Governor

Apaar Kamal
2K17/MC/21

Name : Apaar Kamal

Branch : Mathematics and Computing

Year : 2020

Roll No : 2K17/MC/21

Full Address (Permanent) :

House Number - 29 , Friends Enclave , Rudrapur , Udham Singh Nagar , Uttarakhand, India

Telephone No. : (+91) 8279798102

Name and address of the Trainer Organisation:

Samsung R&D Bangalore, No.2870, Phoenix Building, Bagmane Constellation Business Park Doddanekundi Circle Post, Outer Ring Rd, Doddanekundi, Marathahalli, Bengaluru, Karnataka - 560037

Telephone No. : 080-4681-3000

Training Started on (Date): 18/05/2020

Completed on (Date): 17/07/2020

Trained under the guidance of:

Mr. Sanjay Ramarao Shinde

Staff Engineer / CP Systems

Samsung R&D Institute Bangalore

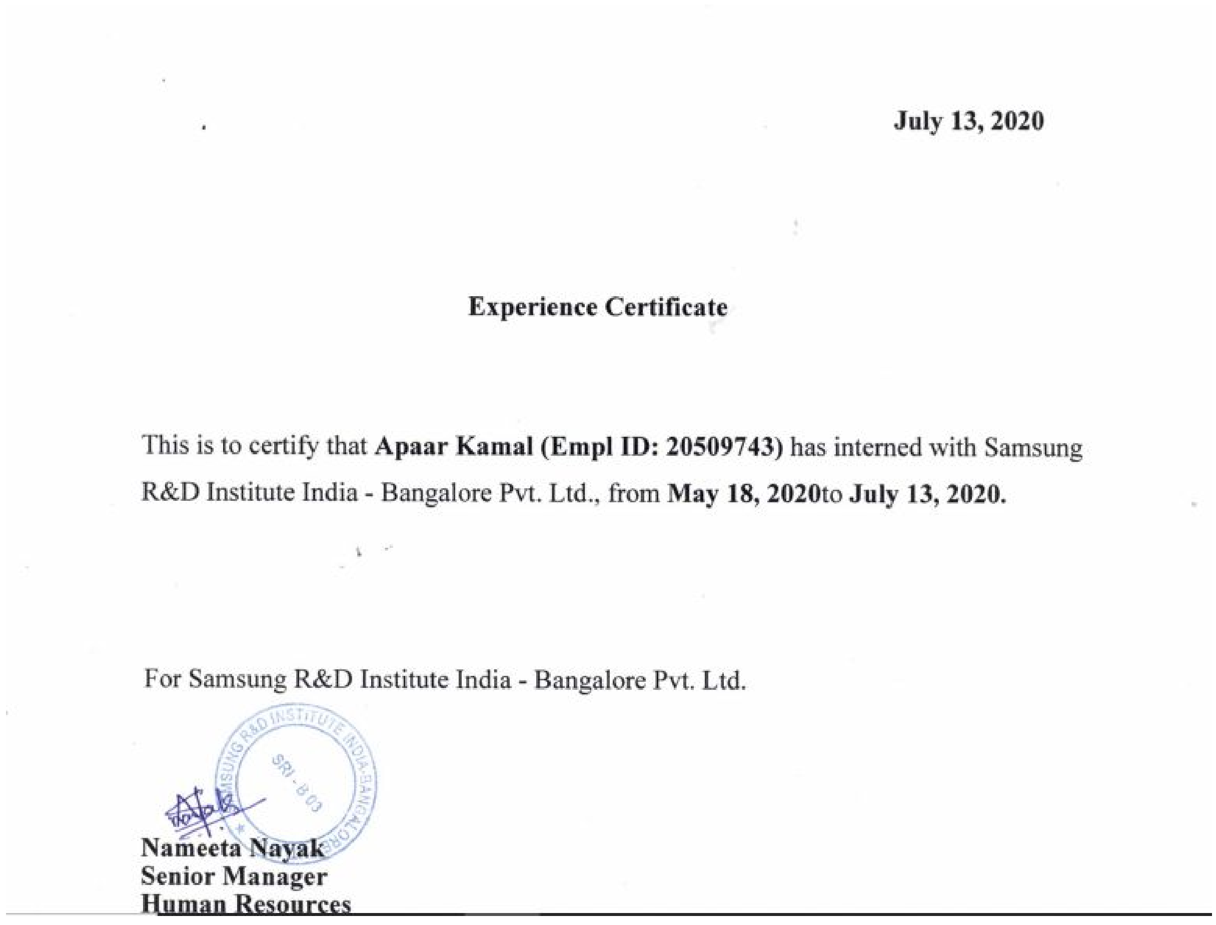
CONTENTS OF THE DIARY

ACKNOWLEDGEMENT

The internship experience I had at Samsung Research Institute, Bangalore was a great leap in my professional development. Hence, I would like to express my most sincere gratitude to Samsung Electronics for giving me this opportunity. I am also grateful for having a chance to meet so many wonderful and enthusiastic professionals who led me through this internship Period.

I am immensely thankful to my mentors Mr. Sanjay Ramarao Shinde, and every member of my team who led me through this awesome experience of working on the project. I worked with the Samsung Research Institute, Bangalore in the Networks and Communication Team, on “Thermal Aware Adaptive governor”.

I would also like to thank my parents and friends for the constant support during the course of my internship.



July 13, 2020

Experience Certificate

This is to certify that **Apaar Kamal (Empl ID: 20509743)** has interned with Samsung R&D Institute India - Bangalore Pvt. Ltd., from **May 18, 2020**to **July 13, 2020.**

For Samsung R&D Institute India - Bangalore Pvt. Ltd.

SAMSUNG R&D INSTITUTE INDIA-BANGALORE
SRI - B 03

[illegible]

Nameeta Nayak
Senior Manager
Human Resources

ABOUT SAMSUNG

Samsung is a South Korean multinational conglomerate headquartered in Samsung Town, Seoul. Samsung was founded by Lee Byung-chul in 1938 as a trading company. Over the next three decades, the group diversified into areas including food processing, textiles, insurance, securities, and retail. Samsung entered the electronics industry in the late 1960s and the construction and shipbuilding industries in the mid-1970s; these areas would drive its subsequent growth. Since 1990, Samsung has increasingly globalised its activities and electronics; in particular, its mobile phones and semiconductors have become its most important source of income. As of 2017, Samsung has the 6th highest global brand value. Its affiliate companies produce around a fifth of South Korea's total exports. Samsung's revenue was equal to 17% of South Korea's $1,082 billion GDP.

Samsung Research is the advanced research & development hub of Samsung's Consumer Electronics (CE) Division and IT & Mobile Communications (IM) Division. Samsung R&D Institute India-Bangalore (SRI-B) is the largest R&D Center outside of South Korea and a key innovation hub in the Samsung group. The organisation has executed close to 300 projects since its formation. SRI-B works with many organisations involved in the field of software and technology in India.

The specific purpose of SRI-B in the Samsung family is two-fold: to create USPs for global flagship devices by creating significant advancements in Modem, Multimedia, Artificial Intelligence, Internet of Things, and to make for India by catering to the specific needs of Indian consumers. With a pivotal role in maximizing synergy from technological convergence and securing state-of-the-art technologies, Samsung Research actively conducts research to identify new future growth areas for Samsung, creating new value to improve people's lives. Core research themes at Samsung Research include artificial intelligence, data intelligence, Internet of Things, Tizen, smart machine, security, and next generation media. In collaboration with worldwide universities, research institutes, and partner companies, Samsung Research aims to reach its goal to "Shape the Future with Innovation and Intelligence".

INTRODUCTION

I did my summer internship at Samsung Research Institute, Bangalore in the months of May, June and July ‘20. I was assigned to the Networking R&D department of the Samsung Research Institute, Bangalore which is a critical department in production and research.

Our main problem is to control the temperature of the CPU/systems. So, we are trying to do that by controlling the clock frequency.
Power drawn (i.e. heat generated) is determined by CPU utilization.

In a digital, synchronous CMOS circuit (such as your processor), the power consumption is computed by: $P = C \times V^2 \times f$ Where C is the capacitance of the digital circuit (changes based on what instructions are being executed), V is the voltage of the CPU, and f is the clock frequency.

Some instructions draw more power than others, so we will assume it is fixed here. So, basically the temperature of CPU/systems depends on the power consumption. In order to reduce the temperature, one fine way is to decrease the clock frequency as it is directly proportional to the power consumption. When the clock frequency is high, a certain program is executed fast i.e. less time is required. But due to high frequency CPU utilization is also high which leads to rising of the CPU/system temperature. So, when we lower the clock frequency the same program is executed in a longer time period, thereby reducing the power consumption and CPU utilization is also reduced, thus temperature is lowered. To resolve this problem, we can implement a governor which changes the clock frequency to minimum (i.e. powersave) when threshold temperature is hit, otherwise it will follow the default maximum frequency (i.e. performance).

WEEK 1

There were four webinars in which various heads of the organization addressed all the interns.

VDI SETUP AND SOFTWARE INSTALLATION

VDI was set up and other required software was installed.

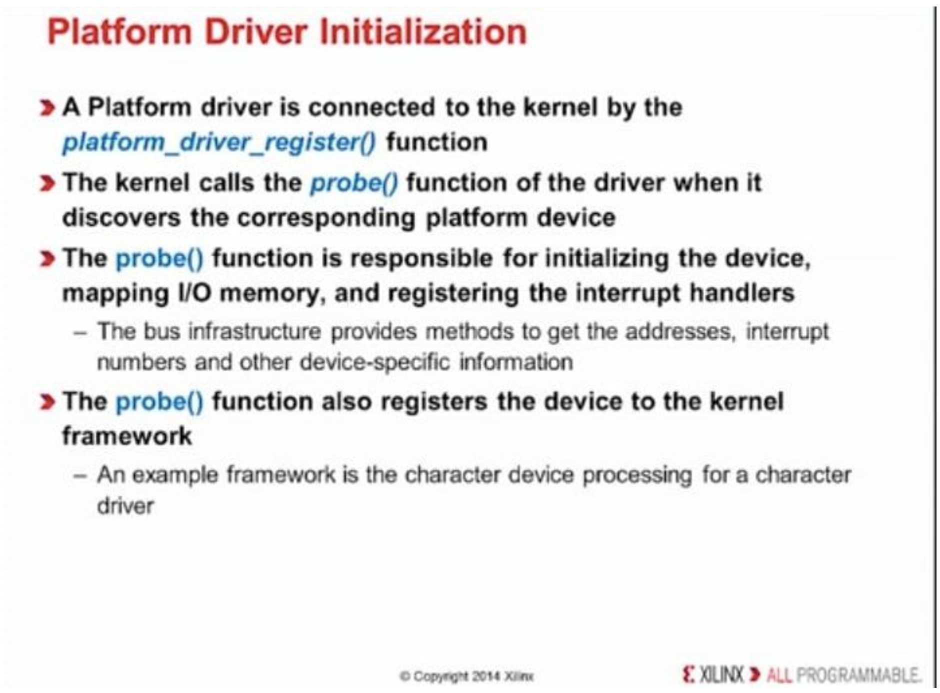
UNDERSTANDING PROBLEM STATEMENT

Problem statement was introduced by the mentor. worked on the problem statement and understood it using use cases. The Aim of the project was to implement a thermal aware adaptive governor.

WEEK 2 & 3

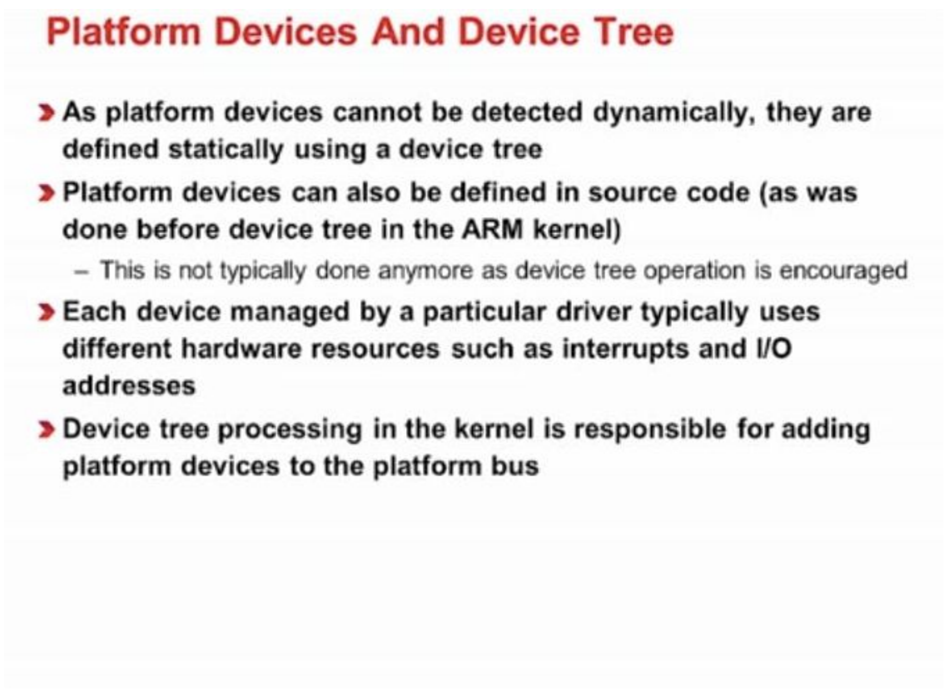
In week 2 and week 3, I was required to learn some concept about the which were :

1. Platform Driver Initialisation.
2. Platform Devices Driver Basics.
3. Platform Devices and Device Tree.
4. Platform Driver.
5. Platform Device Driver Memory / Interrupt Details.
6. Platform Device Driver in A kernel Module.
7. Character Device Driver.
8. Character Device Driver File Operations.
9. We need to get the basic training done on device drivers.
10. At the end of week 3 i was contacted by the manager and helped me by providing me the basic roadmap.



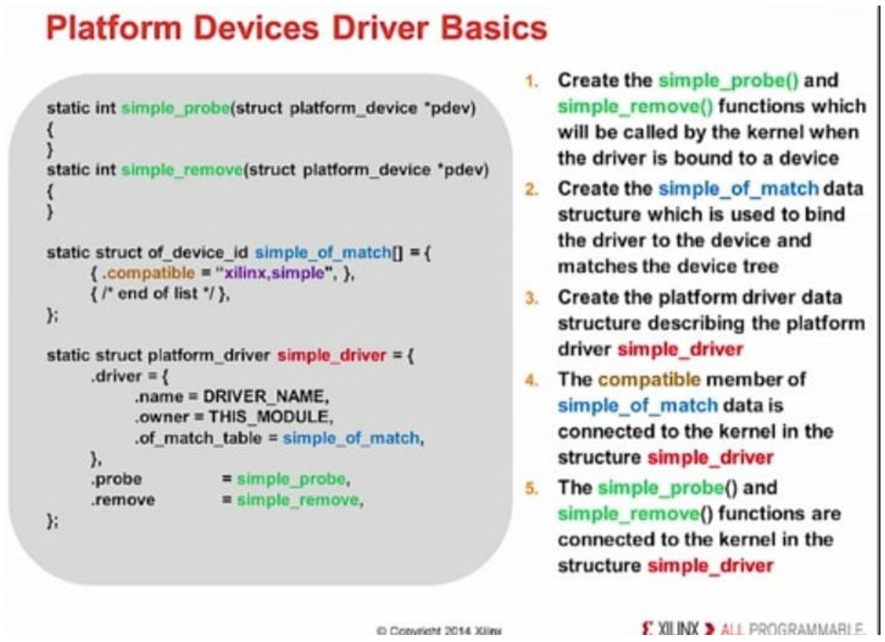
Platform Driver Initialization

- A Platform driver is connected to the kernel by the *platform_driver_register()* function
- The kernel calls the *probe()* function of the driver when it discovers the corresponding platform device
- The probe() function is responsible for initializing the device, mapping I/O memory, and registering the interrupt handlers
  - The bus infrastructure provides methods to get the addresses, interrupt numbers and other device-specific information
- The probe() function also registers the device to the kernel framework
  - An example framework is the character device processing for a character driver



Platform Devices And Device Tree

- As platform devices cannot be detected dynamically, they are defined statically using a device tree
- Platform devices can also be defined in source code (as was done before device tree in the ARM kernel)
  - This is not typically done anymore as device tree operation is encouraged
- Each device managed by a particular driver typically uses different hardware resources such as interrupts and I/O addresses
- Device tree processing in the kernel is responsible for adding platform devices to the platform bus



Platform Devices Driver Basics

```
static int simple_probe(struct platform_device *pdev)
{
}
static int simple_remove(struct platform_device *pdev)
{
}

static struct of_device_id simple_of_match[] = {
        { .compatible = "xilinx,simple", },
        { /* end of list */ },
};

static struct platform_driver simple_driver = {
        .driver = {
                .name = DRIVER_NAME,
                .owner = THIS_MODULE,
                .of_match_table = simple_of_match,
        },
        .probe          = simple_probe,
        .remove         = simple_remove,
};
```

1. Create the simple_probe() and simple_remove() functions which will be called by the kernel when the driver is bound to a device
2. Create the simple_of_match data structure which is used to bind the driver to the device and matches the device tree
3. Create the platform driver data structure describing the platform driver simple_driver
4. The compatible member of simple_of_match data is connected to the kernel in the structure simple_driver
5. The simple_probe() and simple_remove() functions are connected to the kernel in the structure simple_driver

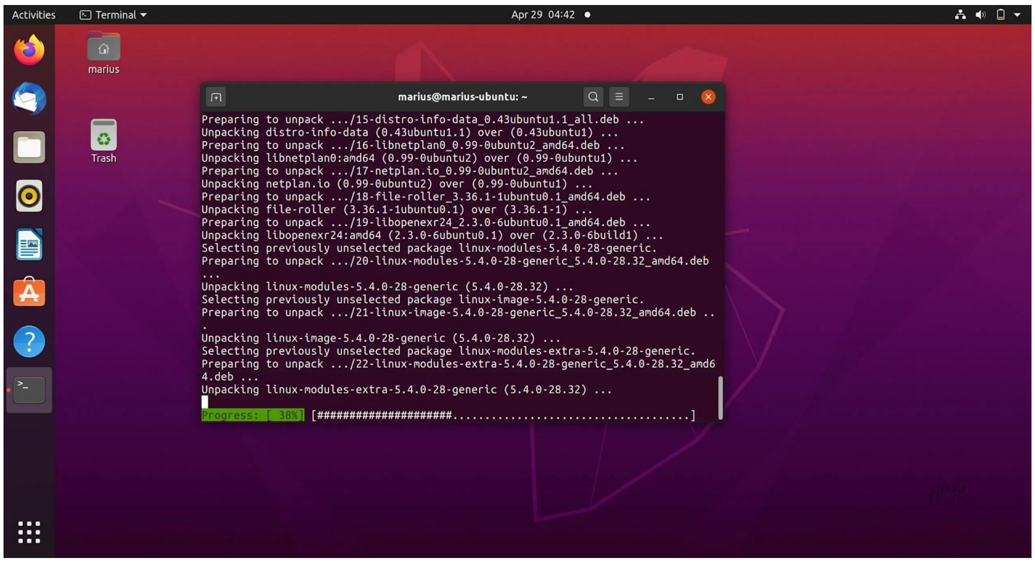
XILINX ALL PROGRAMMABLE.

WEEK 4

As the system requirements were huge and we had to install ubuntu and download linux source code. We had multiple IT failures and we had to install and uninstall it multiple times before we were good to go.
In week 4. I need to do the following tasks:

1. Install the ubuntu(LINUX).
2. Basic loading and unloading of drivers on the kernel level.
3. Learned the use of basic commands of ubuntu.
4. Learned the basic command like cat, echo.
5. Learned about how to make "Make file".

6. Made a driver which simply printed the “hello world” on the kernel.
7. I was having some problems with my ubuntu. I was unable to do some calls through my terminal, resolving them took the rest of the week.



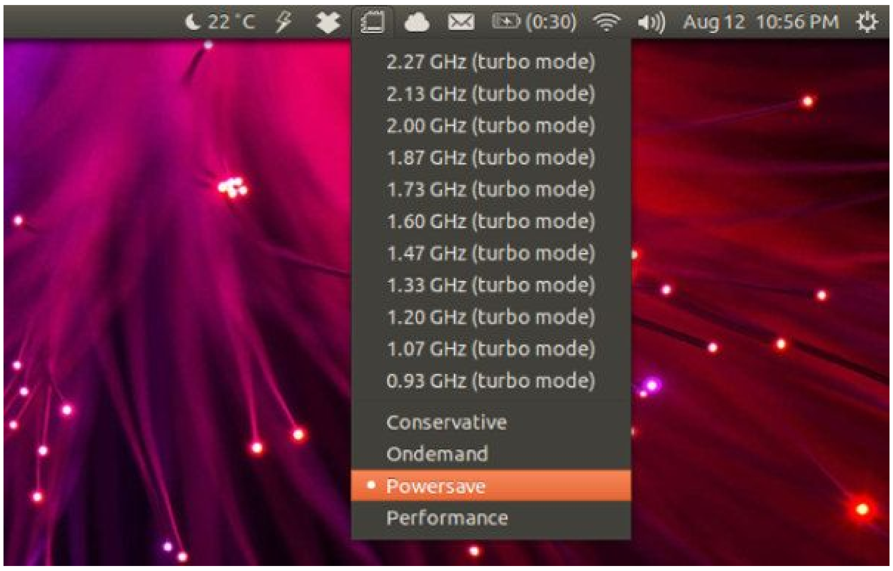
```
Preparing to unpack .../15-distro-info-data_0.43ubuntu1.1_all.deb ...
Unpacking distro-info-data (0.43ubuntu1.1) over (0.43ubuntu1) ...
Preparing to unpack .../16-libnetplan0_0.99-0ubuntu2_amd64.deb ...
Unpacking libnetplan0:amd64 (0.99-0ubuntu2) over (0.99-0ubuntu1) ...
Preparing to unpack .../17-netplan.io_0.99-0ubuntu2_amd64.deb ...
Unpacking netplan.io (0.99-0ubuntu2) over (0.99-0ubuntu1) ...
Preparing to unpack .../18-file-roller_3.36.1-1ubuntu0.1_amd64.deb ...
Unpacking file-roller (3.36.1-1ubuntu0.1) over (3.36.1-1) ...
Preparing to unpack .../19-libopenexr24_2.3.0-6ubuntu0.1_amd64.deb ...
Unpacking libopenexr24:amd64 (2.3.0-6ubuntu0.1) over (2.3.0-6build1) ...
Selecting previously unselected package linux-modules-5.4.0-28-generic.
Preparing to unpack .../20-linux-modules-5.4.0-28-generic_5.4.0-28.32_amd64.deb
...
Unpacking linux-modules-5.4.0-28-generic (5.4.0-28.32) ...
Selecting previously unselected package linux-image-5.4.0-28-generic.
Preparing to unpack .../21-linux-image-5.4.0-28-generic_5.4.0-28.32_amd64.deb ..
.
Unpacking linux-image-5.4.0-28-generic (5.4.0-28.32) ...
Selecting previously unselected package linux-modules-extra-5.4.0-28-generic.
Preparing to unpack .../22-linux-modules-extra-5.4.0-28-generic_5.4.0-28.32_amd6
4.deb ...
Unpacking linux-modules-extra-5.4.0-28-generic (5.4.0-28.32) ...

Progress: [ 38%] [######################...................................]
```

WEEK 5

We started working on the design of cpufreq governor and marked files important to us from the ocean of files in linux source code. Also we identified the functions in the files that are responsible for the work we want to implement. After that we submitted our design proposal.Basically the steps were :

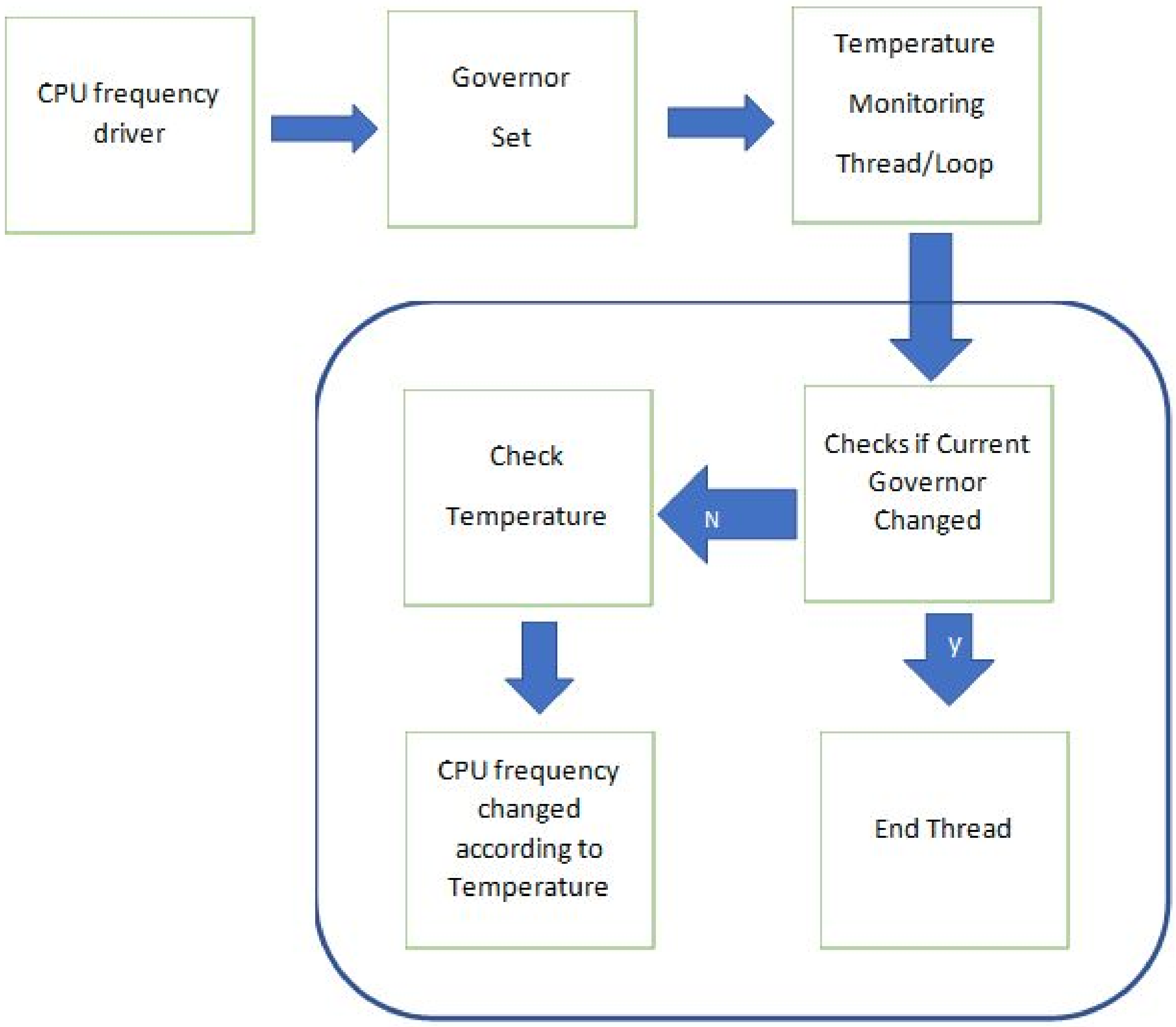
1. First i was required to learn how i can change the current frequency governor,

2. Then I have to trace back all the calls from the terminal to the kernel.
3. Before the above task i was required to learn how echo and cat command works.
4. For that I read code from cpufreq.c and many other files.
5. Since, the code consists of 50,000 lines so it took the rest of the week.



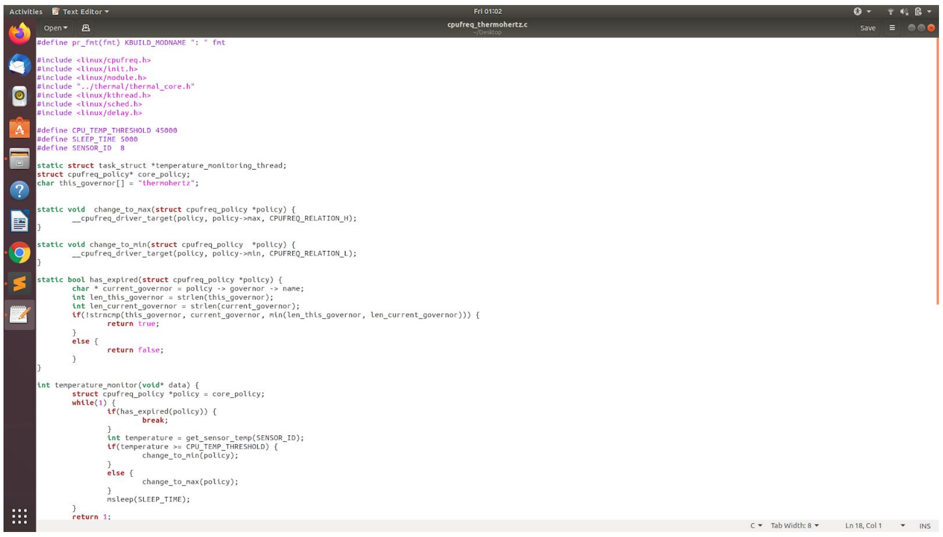
WEEK 6

This week I built a kernel for testing purposes. We got out design approved and started its implementation and after a good amount of brainstorming sessions, we got the design implemented and now it was good to go.

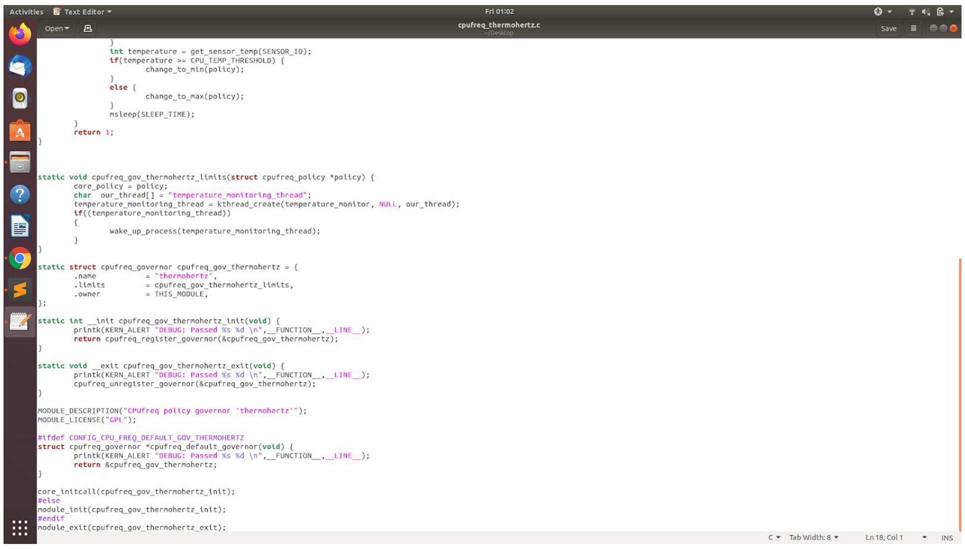

CPU frequency driver
Governor Set
Temperature Monitoring Thread/Loop
Checks if Current Governor Changed
N
Check Temperature
y
CPU frequency changed according to Temperature
End Thread

WEEK 7

This week we modified the implementation as suggested by our mentor and this made our governor more efficient and later we prepared our final project report and pitched. We got good appreciation for our work.



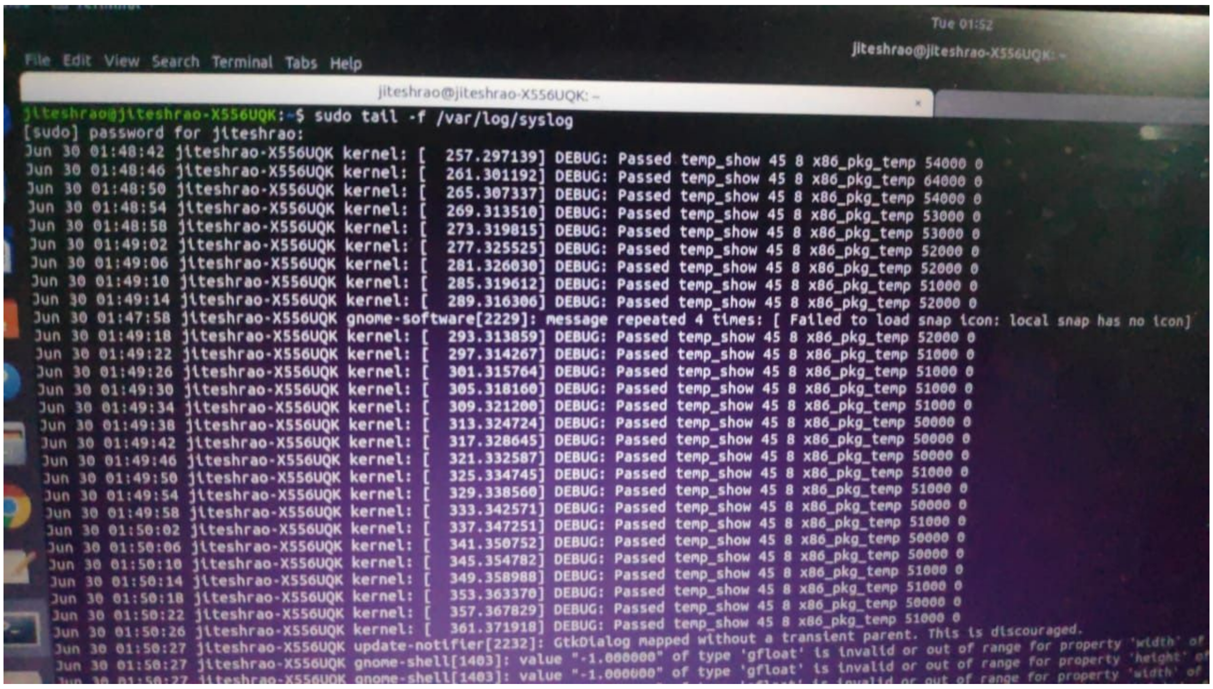
```c
#define pr_fmt(fmt) KBUILD_MODNAME ": " fmt

#include <linux/cpufreq.h>
#include <linux/init.h>
#include <linux/module.h>
#include "../thermal/thermal_core.h"
#include <linux/kthread.h>
#include <linux/sched.h>
#include <linux/delay.h>

#define CPU_TEMP_THRESHOLD 45000
#define SLEEP_TIME 5000
#define SENSOR_ID  8

static struct task_struct *temperature_monitoring_thread;
struct cpufreq_policy* core_policy;
char this_governor[] = "thermohertz";


static void  change_to_max(struct cpufreq_policy *policy) {
        __cpufreq_driver_target(policy, policy->max, CPUFREQ_RELATION_H);
}

static void change_to_min(struct cpufreq_policy  *policy) {
        __cpufreq_driver_target(policy, policy->min, CPUFREQ_RELATION_L);
}

static bool has_expired(struct cpufreq_policy *policy) {
        char * current_governor = policy -> governor -> name;
        int len_this_governor = strlen(this_governor);
        int len_current_governor = strlen(current_governor);
        if(!strncmp(this_governor, current_governor, min(len_this_governor, len_current_governor))) {
                return true;
        }
        else {
                return false;
        }
}

int temperature_monitor(void* data) {
        struct cpufreq_policy *policy = core_policy;
        while(1) {
                if(has_expired(policy)) {
                        break;
                }
                int temperature = get_sensor_temp(SENSOR_ID);
                if(temperature >= CPU_TEMP_THRESHOLD) {
                        change_to_min(policy);
                }
                else {
                        change_to_max(policy);
                }
                msleep(SLEEP_TIME);
        }
        return 1;
```



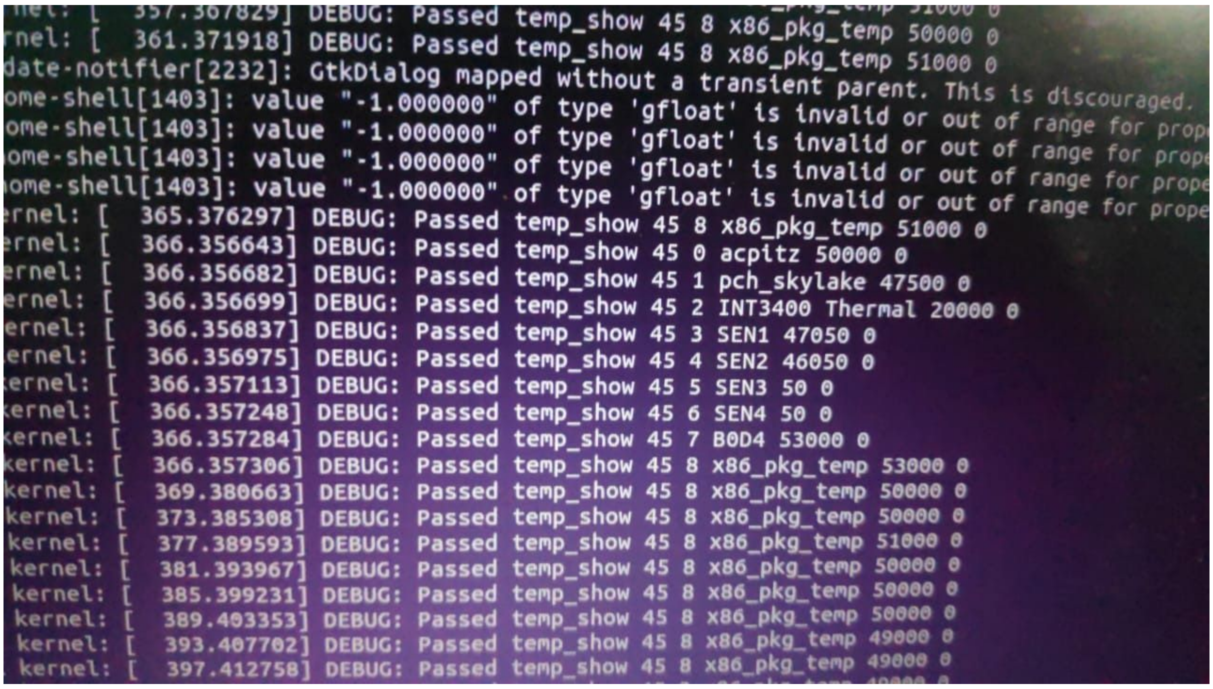
```c
                }
                int temperature = get_sensor_temp(SENSOR_ID);
                if(temperature >= CPU_TEMP_THRESHOLD) {
                        change_to_min(policy);
                }
                else {
                        change_to_max(policy);
                }
                msleep(SLEEP_TIME);
        }
        return 1;
}



static void cpufreq_gov_thermohertz_limits(struct cpufreq_policy *policy) {
        core_policy = policy;
        char  our_thread[] = "temperature_monitoring_thread";
        temperature_monitoring_thread = kthread_create(temperature_monitor, NULL, our_thread);
        if((temperature_monitoring_thread))
        {
                wake_up_process(temperature_monitoring_thread);
        }
}

static struct cpufreq_governor cpufreq_gov_thermohertz = {
        .name           = "thermohertz",
        .limits         = cpufreq_gov_thermohertz_limits,
        .owner          = THIS_MODULE,
};

static int __init cpufreq_gov_thermohertz_init(void) {
        printk(KERN_ALERT "DEBUG: Passed %s %d \n",__FUNCTION__,__LINE__);
        return cpufreq_register_governor(&cpufreq_gov_thermohertz);
}

static void __exit cpufreq_gov_thermohertz_exit(void) {
        printk(KERN_ALERT "DEBUG: Passed %s %d \n",__FUNCTION__,__LINE__);
        cpufreq_unregister_governor(&cpufreq_gov_thermohertz);
}

MODULE_DESCRIPTION("CPUfreq policy governor 'thermohertz'");
MODULE_LICENSE("GPL");

#ifdef CONFIG_CPU_FREQ_DEFAULT_GOV_THERMOHERTZ
struct cpufreq_governor *cpufreq_default_governor(void) {
        printk(KERN_ALERT "DEBUG: Passed %s %d \n",__FUNCTION__,__LINE__);
        return &cpufreq_gov_thermohertz;
}

core_initcall(cpufreq_gov_thermohertz_init);
#else
module_init(cpufreq_gov_thermohertz_init);
#endif
module_exit(cpufreq_gov_thermohertz_exit);
```

WEEK 8

Prepared for presentation and ran some final tests.


File Edit View Search Terminal Tabs Help
Tue 01:52
jiteshrao@jiteshrao-X556UQK: ~

```
jiteshrao@jiteshrao-X556UQK:~$ sudo tail -f /var/log/syslog
[sudo] password for jiteshrao:
Jun 30 01:48:42 jiteshrao-X556UQK kernel: [  257.297139] DEBUG: Passed temp_show 45 8 x86_pkg_temp 54000 0
Jun 30 01:48:46 jiteshrao-X556UQK kernel: [  261.301192] DEBUG: Passed temp_show 45 8 x86_pkg_temp 64000 0
Jun 30 01:48:50 jiteshrao-X556UQK kernel: [  265.307337] DEBUG: Passed temp_show 45 8 x86_pkg_temp 54000 0
Jun 30 01:48:54 jiteshrao-X556UQK kernel: [  269.313510] DEBUG: Passed temp_show 45 8 x86_pkg_temp 53000 0
Jun 30 01:48:58 jiteshrao-X556UQK kernel: [  273.319815] DEBUG: Passed temp_show 45 8 x86_pkg_temp 53000 0
Jun 30 01:49:02 jiteshrao-X556UQK kernel: [  277.325525] DEBUG: Passed temp_show 45 8 x86_pkg_temp 52000 0
Jun 30 01:49:06 jiteshrao-X556UQK kernel: [  281.326030] DEBUG: Passed temp_show 45 8 x86_pkg_temp 52000 0
Jun 30 01:49:10 jiteshrao-X556UQK kernel: [  285.319612] DEBUG: Passed temp_show 45 8 x86_pkg_temp 51000 0
Jun 30 01:49:14 jiteshrao-X556UQK kernel: [  289.316306] DEBUG: Passed temp_show 45 8 x86_pkg_temp 52000 0
Jun 30 01:47:58 jiteshrao-X556UQK gnome-software[2229]: message repeated 4 times: [ Failed to load snap icon: local snap has no icon]
Jun 30 01:49:18 jiteshrao-X556UQK kernel: [  293.313859] DEBUG: Passed temp_show 45 8 x86_pkg_temp 52000 0
Jun 30 01:49:22 jiteshrao-X556UQK kernel: [  297.314267] DEBUG: Passed temp_show 45 8 x86_pkg_temp 51000 0
Jun 30 01:49:26 jiteshrao-X556UQK kernel: [  301.315764] DEBUG: Passed temp_show 45 8 x86_pkg_temp 51000 0
Jun 30 01:49:30 jiteshrao-X556UQK kernel: [  305.318160] DEBUG: Passed temp_show 45 8 x86_pkg_temp 51000 0
Jun 30 01:49:34 jiteshrao-X556UQK kernel: [  309.321200] DEBUG: Passed temp_show 45 8 x86_pkg_temp 51000 0
Jun 30 01:49:38 jiteshrao-X556UQK kernel: [  313.324724] DEBUG: Passed temp_show 45 8 x86_pkg_temp 50000 0
Jun 30 01:49:42 jiteshrao-X556UQK kernel: [  317.328645] DEBUG: Passed temp_show 45 8 x86_pkg_temp 50000 0
Jun 30 01:49:46 jiteshrao-X556UQK kernel: [  321.332587] DEBUG: Passed temp_show 45 8 x86_pkg_temp 50000 0
Jun 30 01:49:50 jiteshrao-X556UQK kernel: [  325.334745] DEBUG: Passed temp_show 45 8 x86_pkg_temp 51000 0
Jun 30 01:49:54 jiteshrao-X556UQK kernel: [  329.338560] DEBUG: Passed temp_show 45 8 x86_pkg_temp 51000 0
Jun 30 01:49:58 jiteshrao-X556UQK kernel: [  333.342571] DEBUG: Passed temp_show 45 8 x86_pkg_temp 50000 0
Jun 30 01:50:02 jiteshrao-X556UQK kernel: [  337.347251] DEBUG: Passed temp_show 45 8 x86_pkg_temp 51000 0
Jun 30 01:50:06 jiteshrao-X556UQK kernel: [  341.350752] DEBUG: Passed temp_show 45 8 x86_pkg_temp 50000 0
Jun 30 01:50:10 jiteshrao-X556UQK kernel: [  345.354782] DEBUG: Passed temp_show 45 8 x86_pkg_temp 50000 0
Jun 30 01:50:14 jiteshrao-X556UQK kernel: [  349.358988] DEBUG: Passed temp_show 45 8 x86_pkg_temp 51000 0
Jun 30 01:50:18 jiteshrao-X556UQK kernel: [  353.363370] DEBUG: Passed temp_show 45 8 x86_pkg_temp 51000 0
Jun 30 01:50:22 jiteshrao-X556UQK kernel: [  357.367829] DEBUG: Passed temp_show 45 8 x86_pkg_temp 50000 0
Jun 30 01:50:26 jiteshrao-X556UQK kernel: [  361.371918] DEBUG: Passed temp_show 45 8 x86_pkg_temp 51000 0
Jun 30 01:50:27 jiteshrao-X556UQK update-notifier[2232]: GtkDialog mapped without a transient parent. This is discouraged.
Jun 30 01:50:27 jiteshrao-X556UQK gnome-shell[1403]: value "-1.000000" of type 'gfloat' is invalid or out of range for property 'width' of
Jun 30 01:50:27 jiteshrao-X556UQK gnome-shell[1403]: value "-1.000000" of type 'gfloat' is invalid or out of range for property 'height' of
Jun 30 01:50:27 jiteshrao-X556UQK gnome-shell[1403]: value "-1.000000" of type 'gfloat' is invalid or out of range for property 'width' of
```



```
rnel: [  361.371918] DEBUG: Passed temp_show 45 8 x86_pkg_temp 51000 0
date-notifier[2232]: GtkDialog mapped without a transient parent. This is discouraged.
ome-shell[1403]: value "-1.000000" of type 'gfloat' is invalid or out of range for prop
ome-shell[1403]: value "-1.000000" of type 'gfloat' is invalid or out of range for prop
ome-shell[1403]: value "-1.000000" of type 'gfloat' is invalid or out of range for prope
ome-shell[1403]: value "-1.000000" of type 'gfloat' is invalid or out of range for prope
rnel: [  365.376297] DEBUG: Passed temp_show 45 8 x86_pkg_temp 51000 0
rnel: [  366.356643] DEBUG: Passed temp_show 45 0 acpitz 50000 0
rnel: [  366.356682] DEBUG: Passed temp_show 45 1 pch_skylake 47500 0
rnel: [  366.356699] DEBUG: Passed temp_show 45 2 INT3400 Thermal 20000 0
rnel: [  366.356837] DEBUG: Passed temp_show 45 3 SEN1 47050 0
rnel: [  366.356975] DEBUG: Passed temp_show 45 4 SEN2 46050 0
ernel: [  366.357113] DEBUG: Passed temp_show 45 5 SEN3 50 0
ernel: [  366.357248] DEBUG: Passed temp_show 45 6 SEN4 50 0
ernel: [  366.357284] DEBUG: Passed temp_show 45 7 B0D4 53000 0
kernel: [  366.357306] DEBUG: Passed temp_show 45 8 x86_pkg_temp 53000 0
kernel: [  369.380663] DEBUG: Passed temp_show 45 8 x86_pkg_temp 50000 0
kernel: [  373.385308] DEBUG: Passed temp_show 45 8 x86_pkg_temp 50000 0
kernel: [  377.389593] DEBUG: Passed temp_show 45 8 x86_pkg_temp 51000 0
kernel: [  381.393967] DEBUG: Passed temp_show 45 8 x86_pkg_temp 50000 0
kernel: [  385.399231] DEBUG: Passed temp_show 45 8 x86_pkg_temp 50000 0
kernel: [  389.403353] DEBUG: Passed temp_show 45 8 x86_pkg_temp 50000 0
kernel: [  393.407702] DEBUG: Passed temp_show 45 8 x86_pkg_temp 49000 0
kernel: [  397.412758] DEBUG: Passed temp_show 45 8 x86_pkg_temp 49000 0
```

Second Innovative Work

I am working as a mentor at Coding Blocks. I took 3 batches this summer and also developed a online course which is best selling course in the industry.

1) Interview Preparation
2) Bootcamp
3) Competitive Programming

Importantly I developed two courses with Prateek Narang, co founder at Coding Blocks.

1) Online Competitive Programming - Link
2) Graph Theory - Link

I also took 30 days workshop on Data Structures and Algorithms in Lovely Professional University at their Training and Placement Centre. Link

REFERENCES

1. https://tools.ietf.org/html/rfc6824
2. https://en.wikipedia.org/wiki/Multipath_TCP
3. https://tools.ietf.org/html/draft-hesmans-mptcp-socket-03
4. https://tools.ietf.org/html/draft-bonaventure-mptcp-backup-00
5. https://en.wikipedia.org/wiki/Multipath_TCP
6. https://multipath-tcp.org/pmwiki.php/Users/Android
7. https://blog.apnic.net/2019/03/04/a-quick-look-at-quic/
8. https://www.youtube.com/watch?v=ZF6-KmHz1sU
9. https://tools.ietf.org/html/draft-ietf-quic-transport-29
10. https://stackoverflow.com/questions/187655/are-https-headers-encrypted
11. https://www.chromium.org/quic