# Concept of Fan Regulator using MINI-2440

Amanpreet Kaur
Department of CSE/IT
Jaypee Institute of Information Technology
Noida, India

Surabhi Agarwal
Student of IT
Jaypee Institute of Information Technology
Noida, India

Tushar Gupta
Student of IT
Jaypee Institute of Information Technology
Noida, India

**Abstract**

**- In this paper, an approach is being proposed for the design and implementation of Concept of Fan Regulator using Mini-2440. A GUI based application named "Concept of Fan Regulator using Mini 2440" is being designed which is done by attaching external hardware (which was created) to this device's General Purpose Input Output (GPIO) port to make useful real time applications. It's an application where user will designate encoder the pulses per round and set the maximum speed of motor. The Mini 2440 displays the status of motor and receive the set speed from users. We have used HyperTerminal, DNW and Linux commands to implement the entire project. Qt Creator, Qt Designer platforms have been used to build the program files.**

**Keywords - DC Motor; DNW; GPIO; GUI; HyperTerminal; Qt Creator; Qt Designer**

## 1 INTRODUCTION

**Mini2440** is a practical low-cost development board, is currently the highest in a Cost effective learning board. It is for the Samsung S3C2440 processor and the use of professional power stable core CPU chip to chip and reset security permit system stability.

Arm 9 board is a 32-bit reduced instruction set computer (RISC) instruction set architecture. The relative simplicity of ARM processors makes them suitable for low power applications. It supports operating systems such as Linux 2.6, Windows CE and Android 1.5. The device has memory limitation having a memory size of only 64MB.

**Embedded system** is a combination of computer hardware and software, and perhaps additional mechanical or other parts, designed to perform a dedicated function. Examples of some basic use of embedded systems used in our daily life includes washing machine, wrist watches etc. Mini 2440 is compatible with some of the external devices such as mouse, keyboard and works normal on the windows platform.

**Cross compilation** is a process in which the compiler allows a computer program written on one type of computer to be used on another type. It the process of one machine excepting a source program as input and producing an object code that is executable on another machine.

The Friendly ARM Mini2440 is a single board computer based on a Samsung S3C2440 ARM9 microprocessor. The board is ideal for learning about ARM systems or integrating into numerous products. The remainder of this paper is organized as follows: Section 2 discuses about the background and related work. Section 3 discusses about the proposed methodology. Section 4 discusses about the experimentation and associated results. Finally, Section 5 concludes the paper.

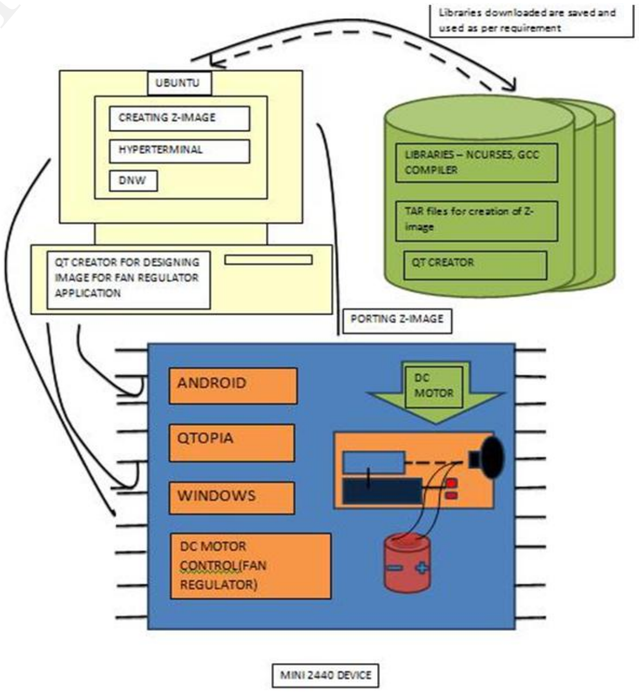

Libraries downloaded are saved and used as per requirement
UBUNTU
CREATING Z-IMAGE
HYPERTERMINAL
DNW
QT CREATOR FOR DESIGNING IMAGE FOR FAN REGULATOR APPLICATION
LIBRARIES – NCURSES, GCC COMPILER
TAR files for creation of Z-image
QT CREATOR
PORTING Z-IMAGE
ANDROID
QTOPIA
WINDOWS
DC MOTOR CONTROL(FAN REGULATOR)
DC MOTOR
MINI 2440 DEVICE

Fig. 1. MINI-2440 overall workflow

## 2 BACKGROUND AND RELATED WORK

To continue with our project, research was done on Mini-2440, understanding its current problems and identifying their solutions.Mini-2440 user manual and research papers like Linux Kernel Development, Speed Control of DC Motor using Neural Network Configuration, etc were studied and helped in finding out the current problems and issues with this device.

Current Problems which we came across included:

- Very Low Power
- Limited Battery Backup
- Touch Screen Issues with Screen Calibration

1. Formatting and erasing the Nand Flash and use Nor Flash while booting and compiling.
2. How to compile and boot all supported Operating Systems in Ubuntu and Windows using HyperTerminal and DNW.
3. How to boot Android, Linux, and Wince using serial port in Mini2440.
4. Creating z-image and yaffs image for OS such as Android, Linux and Windows CE.
5. Cross-compile Android and Qtopia on Ubuntu.
6. Making applications using Qt Creator.
7. Creating GUI based applications which will have a control from device as well as laptop.
8. Connecting external hardware to device by enabling the ARM-9 board kernel and thus making z-image and ensure compatibility with same.
9. Web connectivity on Mini2440.
10. Controlling hardware of the device (LED's and Buttons).
11. Mini 2440 Test applications like Touch panel and PWM (Buzzer).
12. Real time applications such as Concept of Fan Regulator, Temperature Sensors and displaying temperature value on web server, Smoke Sensors, LED ON/OFF Test.

By studying all research papers and current problems, it was understood that it is first necessary to install Operating system into device and then build application using Qt Creator and Qt Designer and then use DNW, Hyperterminal and Linux commands to run the project.

## 3 METHODOLOGY

This section is divided into two subsections: Section 3.1 and Section 3.2. Section 3.1 describes the creation of hardware and Section 3.2 describes the proposed algorithms and software used.

### 3.1 *Creation of Hardware*

First of all, creation of hardware was done and then connected to Mini-2440. It was built using DC Motor, connecting wires, breadboard, and battery. A 7-pin and 10-pin wires were used to connect it to GPIO port of device.
Mini2440 has a 34 Pin 2.0mm pitch GPIO interface on CON4. CON4 contains the GPIO pins and a number of other CPU-pins, such as AIN0-AIN3, SPI interface, I2C interface, GPB0 and GPB1. CON4's pin-1 and pin-9 were used to connect the hardware to GPIO port of Mini-2440.

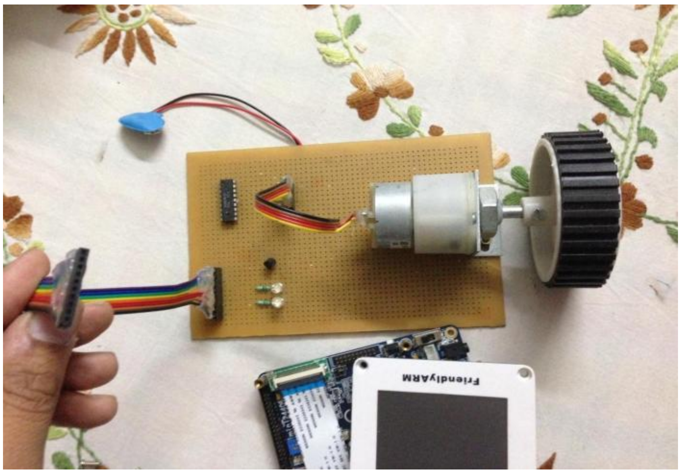


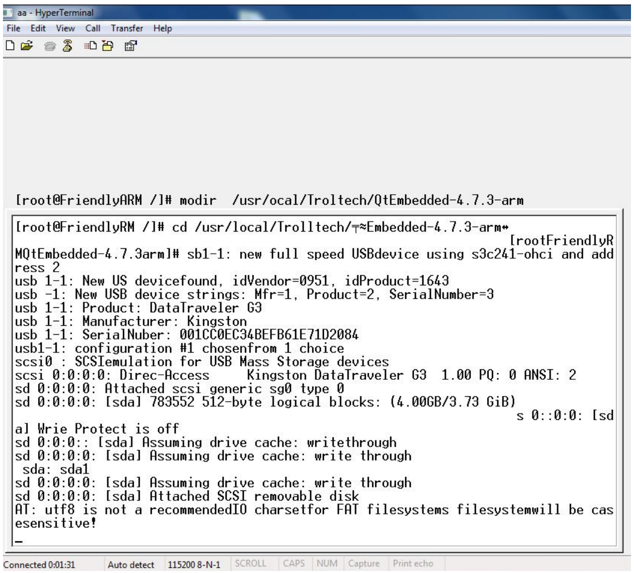
Fig. 2. Hardware attached to Mini-2440

### 3.2 *Algorithms and Software Used*

This section depicts the workflow of proposed approach and list all the software and algorithms used.
Mini-2440 device needs software installations to be able to work properly. A 32 bit OS Laptop is needed for Mini-2440 as this device works with 64bit OS only if each and every software is certified and licensed.
We needed:

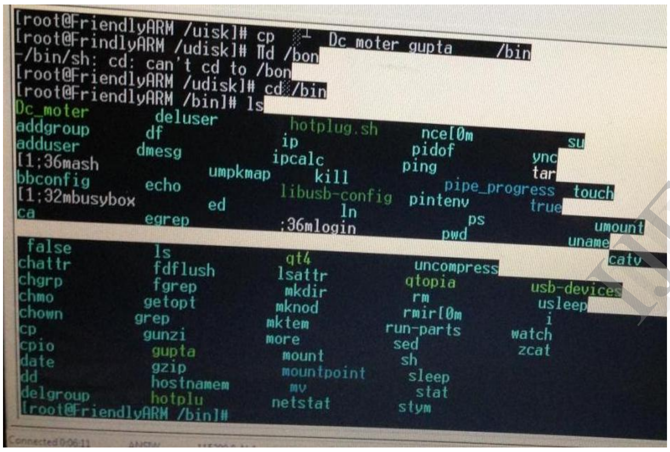
- DNW
- HyperTerminal for booting OS and executing GUI based application.
- Serial to USB Convertor Driver
- S3C USB Host Driver
- USB Download Driver
- Supervivi-128M-bootloader
- Ser2pl Driver
- We needed UBUNTU 12.04 to create the z-image of OS
- Ubuntu Qt Creator
- Hardware for "Concept of Fan Regulator" was connected to GPIO port of MINI-2440 to test run and control.

After successful deployment of Operating System into Mini2440, an application was designed on Qt Designer to run on this device.
gupta file which is a script file was first created in Linux environment and then Dc_Moter file was created in Ubuntu 12.04 using Qt Creator for GUI based application. Then all supported libraries were installed in this device for cross compiling bin files for motion.
All these files were booted into device by inserting pen drive into Mini2440. After successful run and testing of all files, we run our application "Concept of Fan Regulator" using HyperTerminal and DNW. After successful run of our application file, GUI appears on the device. This application enables users to control the speed of motor and allows users to see the speed of motor on web server.



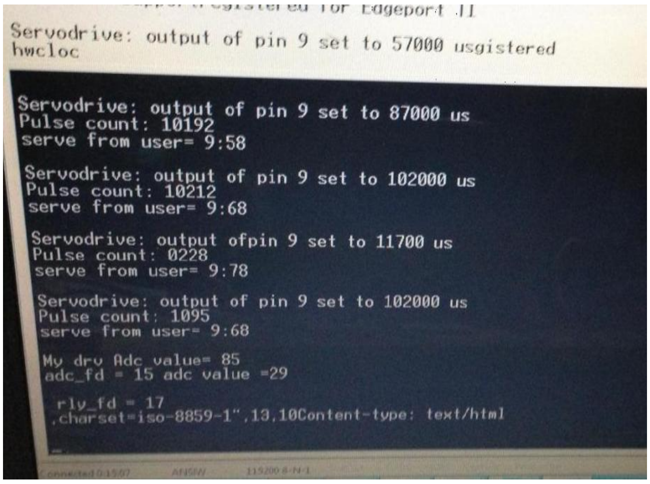
```
[root@FriendlyARM /]# modir  /usr/ocal/Troltech/QtEmbedded-4.7.3-arm
[root@FriendlyRM /]# cd /usr/local/Trolltech/╤≈Embedded-4.7.3-arm↔
                                                                 [rootFriendlyR
MQtEmbedded-4.7.3arm]# sb1-1: new full speed USBdevice using s3c241-ohci and add
ress 2
usb 1-1: New US devicefound, idVendor=0951, idProduct=1643
usb -1: New USB device strings: Mfr=1, Product=2, SerialNumber=3
usb 1-1: Product: DataTraveler G3
usb 1-1: Manufacturer: Kingston
usb 1-1: SerialNuber: 001CC0EC34BEFB61E71D2084
usb1-1: configuration #1 chosenfrom 1 choice
scsi0 : SCSIemulation for USB Mass Storage devices
scsi 0:0:0:0: Direc-Access     Kingston DataTraveler G3  1.00 PQ: 0 ANSI: 2
sd 0:0:0:0: Attached scsi generic sg0 type 0
sd 0:0:0:0: [sda] 783552 512-byte logical blocks: (4.00GB/3.73 GiB)
                                                                  s 0::0:0: [sd
a] Wrie Protect is off
sd 0:0:0:: [sda] Assuming drive cache: writethrough
sd 0:0:0:0: [sda] Assuming drive cache: write through
 sda: sda1
sd 0:0:0:0: [sda] Assuming drive cache: write through
sd 0:0:0:0: [sda] Attached SCSI removable disk
AT: utf8 is not a recommendedIO charsetfor FAT filesystems filesystemwill be cas
esensitive!
_
```

Fig. 3. Application Deployment (1) - attaching pen drive to device and copying all files to bin folder



```
[root@FriendlyARM /uisk]# cp       Dc moter gupta      /bin
[root@FrindlyARM /udisk]# Пd /bon
-/bin/sh: cd: can't cd to /bon
[root@FriendlyARM /udisk]# cd /bin
[root@FriendlyARM /bin]# ls
Dc_moter        deluser
addgroup       df                 hotplug.sh       nce[0m
adduser       dmesg              ip              pidof           su
[1;36mash                          ipcalc          ping            ync
bbconfig       echo    umpkmap       kill                          tar
[1;32mbusybox                     libusb-config  pintenv  pipe_progress  touch
ca                    ed              ln                          true
               egrep               :36mlogin        ps              umount
                                               pwd             uname
 false          ls                 qt4                             catv
chattr          fdflush           lsattr        uncompress
chgrp           fgrep              mkdir        qtopia          usb-devices
chmo           getopt             mknod          rm             usleep
chown          grep              mktem         rmir[0m          i
cp              gunzi             more          run-parts      watch
cpio            gupta              mount         sed            zcat
date            gzip               mountpoint    sh
dd              hostnamem           mv            sleep
delgroup        hotplu            netstat        stat
[root@FriendlyARM /bin]#                       stym
```

Fig. 4. Application Deployment (2) - Dc_moter file and gupta file in bin folder



```
registered for Edgeport .II
Servodrive: output of pin 9 set to 57000 usgistered
hwcloc

Servodrive: output of pin 9 set to 87000 us
Pulse count: 10192
serve from user= 9:58

Servodrive: output of pin 9 set to 102000 us
Pulse count: 10212
serve from user= 9:68

Servodrive: output ofpin 9 set to 11700 us
Pulse count: 0228
serve from user= 9:78

Servodrive: output of pin 9 set to 102000 us
Pulse count: 1095
serve from user= 9:68

My drv Adc value= 85
adc_fd = 15 adc value =29

rly_fd = 17
,charset=iso-8859-1",13,10Content-type: text/html
```

Fig. 5. GUI Application Deployment (3) – application successfully run on Mini-2440 displaying the speed of motor

Common errors during Application Deployment:

1. Do not forget touch screen calibration. If it is lost at one corner then command cannot work in that corner.
2. Check the gupta file, at last line should be Dc_moter -qws
3. libts-1.s0.1. .... if we get any this kind of error just modify the first number instead

# ln –s libts-0.0.so.0 libts-1.0.so.0 (the library is being changed of Mini-2440)

## 4 EXPERIMENTATION AND RESULTS

In this section, experimentation and results of this project are being discussed. This section is divided into three subsections: Section 4.1 to Section 4.3 as mentioned below.

### *4.1 Installation of Qtopia OS into Mini-2440*

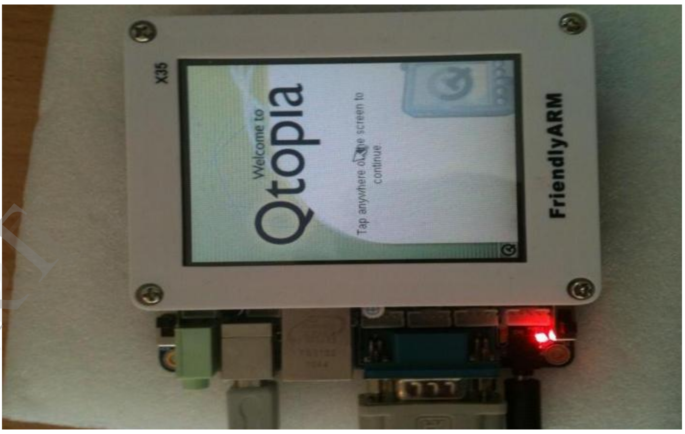
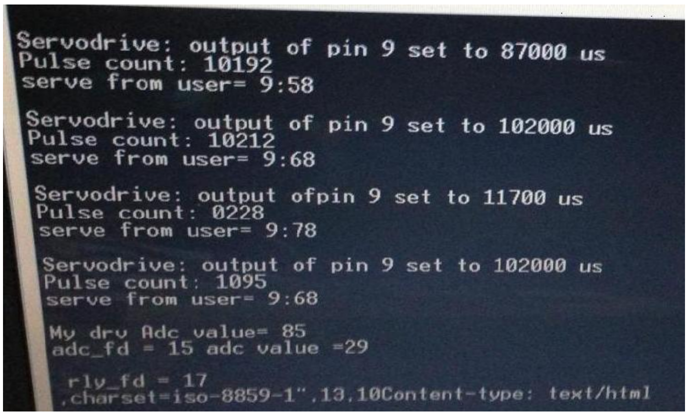


Fig. 6. Running Qtopia OS on Mini-2440

### *4.2 GUI Application building and running*



```
Servodrive: output of pin 9 set to 87000 us
Pulse count: 10192
serve from user= 9:58

Servodrive: output of pin 9 set to 102000 us
Pulse count: 10212
serve from user= 9:68

Servodrive: output ofpin 9 set to 11700 us
Pulse count: 0228
serve from user= 9:78

Servodrive: output of pin 9 set to 102000 us
Pulse count: 1095
serve from user= 9:68

My drv Adc value= 85
adc_fd = 15 adc value =29

rly_fd = 17
,charset=iso-8859-1",13,10Content-type: text/html
```

Fig. 7. Application running on HyperTerminal

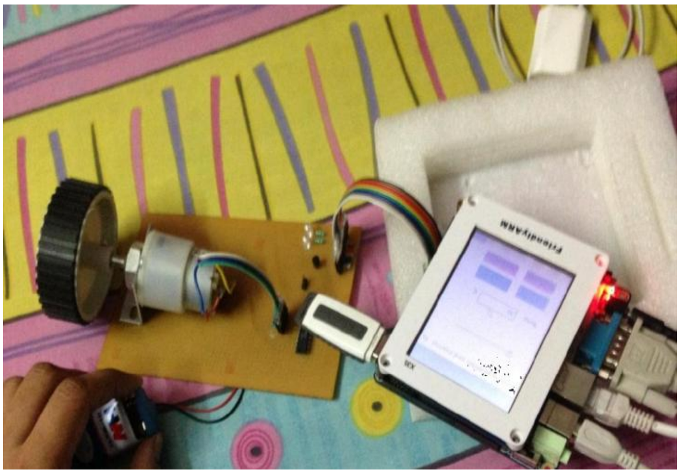


Fig. 8. Concept of Fan Regulator using Mini-2440

### *4.3 Testing GUI Application*

Testing has been performed by using following test cases:

| Test Cases | | Completed | % Completed | Errors |
|---|---|---|---|---|
| Performance | Prevention | Yes | 99 | Error Free |
| | Testing | Yes | 100 | |
| | Detection | Yes | 100 | |
| Compatibility | Detection | Yes | 100 | Error Free |
| Load | Prevention | Yes | 95 | Error free |
| | Detection | Yes | 100 | |
| Code Modules | Prevention | Yes | 100 | Error free |
| | Detection | Yes | 100 | |
| | | | | |

TABLE I- Testing using different test cases

| Layers | Basic Idea of Layer | Problems | Testing Technique |
|---|---|---|---|
| Presentation | User interaction with application | Display issue | After z-image booting of OS, wait for some time so that performance is enhanced |
| Business | Used for communication between other layers | Connectivity or Hardware Issue | Hardware testing or checking all connections required for any single failure |
| Logic | Algorithm application Layer | Changes in /bin file | Change in any file can stop running of application and its display on web server. |

TABLE II- Testing of GUI Application

## V CONCLUSION

After working on this device, one should first able to deploy operating system on an ARM processor. Although this device is good for android development and testing, but when it comes to application development, there is not enough support for the same. But still, it was managed to build the real time application named “Concept of Fan Regulator using Mini 2440” and successfully run on this device. .

For future perspective, hardware that was created now also consisted of two LEDs and a temperature sensor.

A real time application for LED On/Off Test through device as well as laptop and displaying the current temperature on web server can be built. Also, device supports smoke sensor applications which can be implemented.

The applications for this device are limited but effort is to build all related real time applications for this device.

## REFERENCES

1. Amit Atri and Md. Ilyas, “Speed Control of DC Motor using Neural Network Configuration”, International Journal of Advanced Research in Computer Science and Software Engineering, Volume 2, Issue 5, May 2012
2. Chandra Shekhar Gohiya, S.S.Sadistap, S.A.Akbar and B.A.Botre, “Design and Development of Digital PID Controller for DC Motor Drive System Using Embedded Platform for Mobile Robot”, 3rd IEEE International Advance Computing Conference,2013
3. Chee-Yee Chong and Srikanta P. Kumar, “Sensor Networks: Evolution, opportunities and Challenges”, Proceedings of the IEEE, Volume 91, Issue 8, August 2013
4. Ian McDonald, “Linux Kernel development”, 2013
5. Matthias Kalle Dalheimer and Klaralvdalens Datakonsult AB, “A Comparison of Qt and Java for Large- Scale, Industrial-Strength GUI Development”, ACM Digital Library, 2007
6. Miss Swati Malgonde and Prof. Mrs. P.S. Deshpande, “Implementation of Smartphone using ARM9 and Embedded Linux operating system: A survey approach”, International Journal of Engineering Research and Applications, Volume 3, Issue 1, 2013
7. M.Manoj Kumar and G.Srinivasa Raju, “Design and Implementation of the Lab Remote Monitoring and Controlling System Based on Embedded Web Technology”, International Journal of Scientific and Research Publications, Volume 3, Issue 3, March 2013
8. Sheetal K. Mahadik and Prof.P.G.Chilver, “On-Line Remote Data Acquisition and Control System for Embedded Real Time Applications”, International Journal of Engineering Research and Applications, Volume 3, Issue 2, 2010
9. Vaishali Patil and Avichal Kapur, “Real Time Embedded Web Server Based LED’s Control using ARM Processor and Ethernet”, International Journal of Engineering Research and Applications, Volume 3, Issue 2, 2013