

UNIVERSITATEA DIN BUCURESTI

FACULTATEA DE FIZICA

**COMPUTER OPERATED
RECONNAISSANCE ENTITY**

Graduation Paper

Student: ANDREI-LUCIAN BOGZA

Coordinator: Lect. Dr. CORNEL MIRONEL NICULAE

2011

Content

Content	1
Introduction.....	3
SOFTWARE INTERFACE – THE LINUX OPERATING SYSTEM	5
Core components of the Linux operating system.....	7
The boot loader	7
The Kernel	8
The Linux file system	9
Ubuntu Linux	11
The Android Operating System.....	13
DATA TRANSMISSION PROTOCOLS	15
IEEE 802.11	15
802.11a	15
802.11b/g.....	16
TCP/IP transmission protocol	17
PERSONAL CONTRIBUTION.....	19
C.O.R.E v1.0	19
Operating system installation	23
Installation and configuration of the video streaming server	26
Movement and remote control.....	28
C.O.R.E v2.0	32
Android and video streaming configuration.....	34
Movement and remote control.....	37

Any attempt to reproduce or distribute this document or parts of it without mentioning the source website (<http://android.webook.ro>) is forbidden.

Conclusions.....	39
Appendixes	40
Appendix 1. webcam-server startup script /etc/init.d/webcam-server	40
Appendix 2. webcam-server configuration file /var/www/webcam.html	41
Appendix 3. TDA1557Q data sheet	43
Bibliography	52

Introduction

From the beginning of time the human society has been seeking for ways of limiting the amount of effort and work while increasing productivity. Starting with the invention of the wheel, man was able to carry materials easier, faster and for longer distances. Afterwards, the creation of tools to work the earth improved food production. As time passed technological discoveries have made man's work easier, faster and more efficient. The industrial revolution showed mankind that technology can replace an individual in specific areas of industry and thus workers were transformed into supervisors.

In the modern and contemporary era machines also received a certain degree of autonomy thus minimizing the need of careful supervision. With the invention of transistors the information age had begun and computers took the place of the human brain in computing calculations that otherwise would have taken a large amount of time. It also enabled humans to keep track and store a huge amount of information in a particularly small space (a standard hard disk dimensions are $101.6 \text{ mm} \times 25.4 \text{ mm} \times 146 \text{ mm} = 376.77344 \text{ cm}^3$). For example the human brain is estimated to have a capacity in the order of Terabytes, depending on the number of neurons each individual has while a standard hard drive already has 3 Terabytes of data storage capability. Although machines can exceed the human brain in certain tasks they still lack a very important characteristic – intelligence. No matter how fast or how much information it can store, a machine still needs a human to be operated by or to write lines of code and pass them as instructions in order for it to operate autonomously. This means that a machine can be made smart but not intelligent[11].

The latest discoveries in technology have allowed man to send machines to explore environments that would be otherwise lethal. Starting from deep ocean explorations to conquering space man stood at the safety of his desk while machines explored and sent back information about those environments. Nowadays unmanned vehicles roll on the surface of Mars, are exploring the outer edge of the solar system, operate in areas with chemical, biological and nuclear hazards, defuse bombs, execute reconnaissance missions in armed conflicts or even

execute surgical precision strikes on enemy positions while the pilot is thousands of km away without any threat to his life.

This graduation paper describes the author's idea of building a reconnaissance entity that could be operated at distance and has the capability of sending back a video feed of the surrounding environment while maintaining cost effectiveness. To achieve this goal a laptop was used to power a series of motors that would set the device in motion. Two prototypes are presented in this paper, C.O.R.E v1.0 and C.O.R.E v2.0. C.O.R.E v2.0 is the outcome of the analysis made after the 1st prototype was finished in order to slim it down, improve the capabilities of the device and further reduce the cost. For C.O.R.E. v2.0 a smart phone was used to power the motors and allow the device to move.

The paper is structured in 3 main chapters:

The first chapter entitled **SOFTWARE INTERFACE – THE LINUX OPERATING SYSTEM** which describes the software part integrated in the two prototypes and the way the operating system works.

The chapter entitled **DATA TRANSFER PROTOCOLS** describes the protocols used by the devices to communicate with the user and send information to him.

The chapter **PERSONAL CONTRIBUTION** comprehends a detailed description with block diagrams of the two prototypes but also the methods used to achieve a fully functional device.

SOFTWARE INTERFACE – THE LINUX OPERATING SYSTEM

An operating system is a set of software that manages all the hardware and other software at a user's command. The primary job of an operating system is to load software from the internal storage into the memory and send it to the CPU for execution. The use of the computer is made possible because of the operating system.

Linux is an open source operating system created by Linus Torvalds, a student at the University of Helsinki in 1991. The main source of inspiration for Linux was UNIX, a proprietary operating system used in business and academic environment. He began writing Linux on the Minix operating system (a minimal Unix-like operating system) and later as the project matured the project could be developed under itself. The design for Linux follows the basic principles of UNIX as a modular operating system. This type of system uses a kernel which handles the file system, peripheral, network and process control access. The drivers for the system's devices are either directly integrated in the kernel or loaded as separate modules. After a part of his work was shared on the internet a community of developers interested in Linux had been created and a large number of people started to contribute to the development. Open source applications like a shell for command line interface (later the X-Window system was added for graphical user interface), compilers and common Unix tools to carry basic system tasks were added to the Linux kernel and so the fully functional open source, multi user and multitasking operating system was born [6].

After a few years Linux became a mature and very popular operating system and in our days it runs on about 1/3 of the world's servers. It is used on almost every domain, from supercomputers (it holds a 98% share) to embedded devices like smartphones. It had also becomes very popular in the governmental and research areas due to the operating system's free and open source policy [6].

Today a very large number of Linux distributions are available under the open source license. A Linux distribution consists of the Linux kernel (the operating system) and a package of applications that allows the user to adapt the operating system to his specific needs. The distribution is responsible for the software installed in the operating system, the general configuration of the kernel, security, networking and the integration of applications into a usable

Any attempt to reproduce or distribute this document or parts of it without mentioning the source website (<http://android.webook.ro>) is forbidden.

operating system. The development of Linux distributions is supported by an individual, a volunteer community or a commercial entity. The most popular Linux distributions are Ubuntu, Slackware, SUSE, Fedora Core, Gentoo, Knoppix, Linspire, and Mandriva [3].

Linux is able to run on almost all type of processors although Linus Torvalds and the other programmers developed it under the Intel 80x86 and compatible family of processors. Today Linux also runs on systems based on other architectures like AMD64 processors, Motorola 6800, PowerPC, Sun SPARC and UltraSPARC, HP PA-RISC and ARM making it a very portable operating system [6].

Core components of the Linux operating system

The boot loader

The Central Processing Unit of any computer can only execute instructions found in the read-only memory (ROM) or random access memory (RAM). Operating systems are stored on devices like floppy disks, hard disks, CD/DVD, flash memory or USB flash drives. When a computer is powered on the operating system is not loaded in RAM or ROM and thus it cannot start. A small program stored in ROM must be executed at system startup for the computer to find the storage device where the operating system is located. This small program is less than 512 bytes – a single sector called Master Boot Record and its job is to load the second-stage boot loader. Usually the system startup is managed by BIOS, EFI, OpenBIOS, OpenBoot and SLOF [5].

The second-stage boot loader is loaded in RAM by the Master Boot Record executed. From now on the Linux image (a compressed version of the Linux kernel) is loaded into the memory, decompressed and initialized. The next step is for the boot loader to check the system hardware, mount the root device and load the kernel modules. After this step high-level system initialization begins and by running `init`, the first user-space program parent of all processes (Figure 1). The most popular second-stage boot loaders are Grub, LiLo and SysLinux [5].

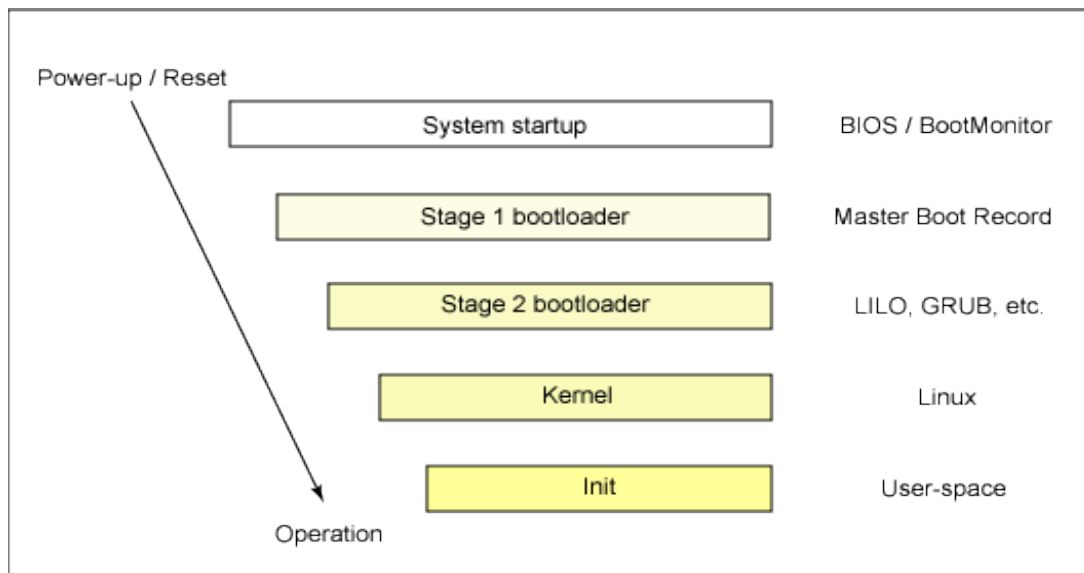


Figure 1. Linux boot order [5].

The Kernel

The Linux kernel is a monolithic kernel. This means that it is one large program where all its components can access the internal structures and routines. The alternative would be to have a microkernel where the functional parts would be split into separate pieces with a strict communication between them.

The problem is that because of the monolithic architecture adding new components in the kernel via the configuration process is time consuming. To use a device that is not built into the kernel you would have to configure and build a new kernel before using it. Unlike traditional monolithic kernels the Linux kernel has the ability to dynamically use Loadable Kernel Modules to configure the device drivers in the system that load and unload into the kernel at any point after the system has started. The dynamically loaded code is attractive because it makes the kernel flexible and its size minimal. Once a module is loaded it is considered a part of the kernel code and it has the same rights and permissions as normal kernel code. Another problem that affects the Loadable Kernel Modules is that once they are loaded and become a part of the kernel, programming errors also known as bugs left in the modules allows them to crash the kernel or leave security holes in the system for denial-of-service attacks or privilege escalation (giving a normal user root access). These programming errors can be corrected by adding

Any attempt to reproduce or distribute this document or parts of it without mentioning the source website (<http://android.webook.ro>) is forbidden.

patches or fixes that are released shortly after being reported thus keeping Linux one of the safest operating systems [7].

The Linux file system

Unlike Windows or Dos, where the file system was divided into partitions, mounted each at startup with a separate letter, Linux has a single hierarchy structure that starts from the root directory “/” that expands into various directories and subdirectories. Other partitions mounted at boot time or after are mounted under the root directory. This is known as a unified filesystem - Figure 2.

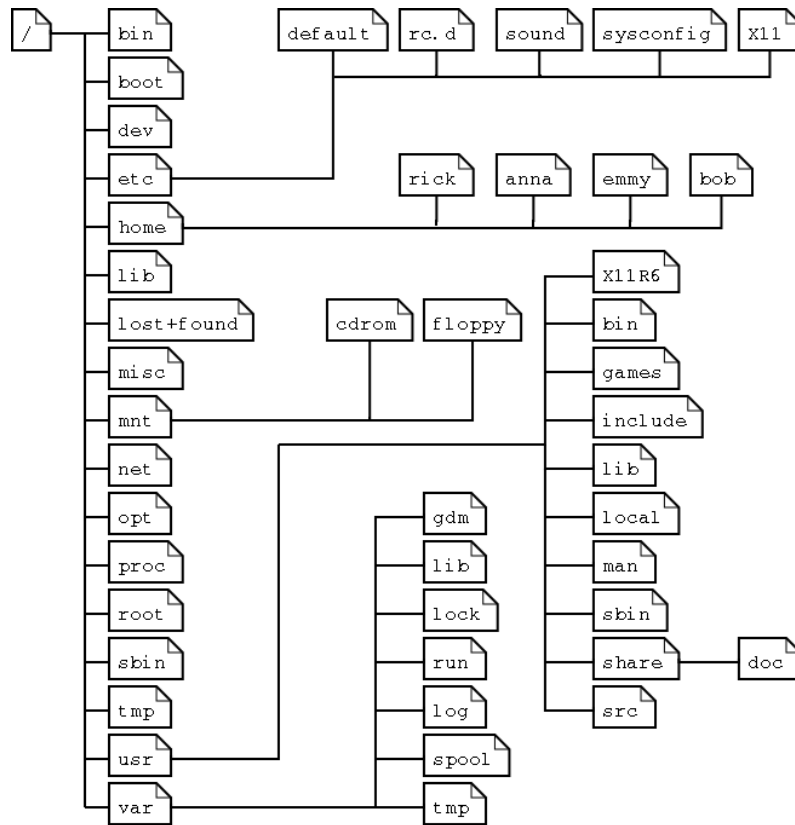


Figure 2. The Linux filesystem [6].

In Linux there are 2 types of partitions: data partitions (the root partition where the operating system's files are contained, various user partitions, partitions with other operating systems) and swap partitions (partitions that expand the computer's physical memory by storing parts of the RAM onto the hard disk). The Linux specific filesystem partitions are ext2, ext3 and ext4 although numerous others are supported like fat16, fat32, ReiserFS, MINIX. Ext3 and ext4 are journalized filesystems that offer protection for data in the case of power failure or unclean shutdown but it requires disk access more often and may cause a high wear level if they are used on solid state disks [8].

Every partition filesystem is divided into inodes. The inode is a entity that contains information about the files stored in the computer (where are located on the disk, to whom they belongs, date and time when they were created, file size, permissions, file type and number of links to them). The inodes are created when the partitions are formatted and a fix number is

Any attempt to reproduce or distribute this document or parts of it without mentioning the source website (<http://android.webook.ro>) is forbidden.

assigned depending on the maximum number of files that can be stored. Usually an inode is created for 2 to 8 Kb of storage [8].

This filesystem structure makes Linux very fast and flexible. It allows better disk space management, faster disk access due to the inode structure that have precise information about where data is stored lowering the number of I/O operations and allows protection for data against unforeseen failures when journalizing is used. It also allows the user to mount parts of the operating systems from remote locations (for example the /opt applications directory from a remote server) and use them giving him very high portability and availability.

Ubuntu Linux

Ubuntu is one of the most popular free Linux distributions developed by the UK based company Canonical Ltd primarily designed for desktop use and offering a user-friendly environment. It is named after a South African humanist philosophy focusing on people's

alliances and relations with each other. The word Ubuntu has its origin from the Bantu language and means “humanity towards others” [6].

Ubuntu is based on the Debian GNU/Linux distribution and statistics suggest it is present on 50% of the Linux desktops although there are also versions for netbooks and servers. The first version was released in 20 October 2004 and since then a new version is released every 6 months with support for 18 months providing patches, bug fixes and updates to programs. Every 4th version receives a long time support (LTS) for 3 years for desktop and 5 years for servers. Each version is composed of a variety of free software packages (except proprietary hardware drivers) which the user can download from the official repositories granting him a wide area of choice for his specific needs. Also a migration tool is present to import various settings from an existing Microsoft Windows installation to the current Ubuntu operating system [6].

Ubuntu is currently supported by x86 32bit and 64bit architectures but unofficially can also run on ARM mobile processors, SPARC, IA-64 and PowerPC. The minimal system requirements make it a perfect candidate for low-end systems and servers by maintaining the functionality without the need of costly hardware upgrades (Figure 1. Linux boot ordeFigure 3). Ubuntu can run without being installed on the host computer (with a performance loss) from a Live CD or Live USB to test the hardware compatibility and support for drivers before installation but also for data recovery in case of hardware or operating system failure. This feature awards it a high degree of portability allowing the user to carry the operating system with him and access it from any computer [6].

Minimum requirements	Server	Desktop
Processor speed	300 MHz	1 GHz
Memory	128	384
Hard Drive	1GB	5GB

Figure 3. Ubuntu minimum system requirements [7].

Due to its popularity Ubuntu enjoys a vast technical support worldwide in more than 55 languages from Canonical but also very active user created communities on forums, IRC channels and chat rooms. In these communities users share their problems, encountered bugs and solutions keeping Ubuntu in a continuous state of development.

Canonical currently supports several versions derived from Ubuntu towards user-specific needs:

- Edubuntu – a version dedicated to school environments and education,
- Kubuntu – a desktop distribution that use the KDE interface instead of Gnome,
- Mythbuntu – a version designed to create a Home Theater Persona Computer,
- Ubuntu studio – a version dedicated to multimedia processing,
- Xubuntu – a distribution for low-end computers that use the Xfce desktop environment to run more efficiently [6].

This makes Ubuntu a great candidate for a wide field of use starting from hosting web servers to scientific research offering a zero cost and high performance alternative to proprietary operating systems. It also gives the user a great amount of power in configuring and adapting to new demands while keeping older technologies working efficiently.

The Android Operating System

Android is an open-source Linux-based operating system designed mobile devices that runs a Java virtual machine (Dalvik) on top of the kernel instead of the standard Linux system. Android was initially developed by Android Inc. and in 2005 was acquired by Google, becoming one of the most popular operating system found on the majority of handheld devices today (about 33% market share, becoming one of the top selling platform). Newer versions include support for the x86 platforms although a device with touch screen capabilities is required to run properly [12].

The Android kernel is derived from the Linux kernel but with several changes differing from the standard architecture, lacking support for the standard GNU libraries and without official support for the X Window System (it can be installed unofficially but depends strongly by the availability of device specific drivers from the manufactures) [12].

The base of the Android Operating System is the Android Market, a portal from where the users can download and install applications to customize the device according to their needs. The majority of the applications found on the Android Market are free of charge and their total

Any attempt to reproduce or distribute this document or parts of it without mentioning the source website (<http://android.webook.ro>) is forbidden.

number is over 300.000 with approximately 5,000,000,000 downloads. Google also added support for 3rd party applications giving arise to numerous Android Market alternatives [12].

Applications are usually developed in Java language but there is also support for C and C++ and even an environment for novices with little programming skills called Google App Inventor. The use of Java means that the applications created are more compact and efficient given the limited memory and storage space found on most mobile devices. The Linux kernel components like device drivers, power management, networking and process management are controlled by Android native libraries that are called through Java interfaces. The advantage is that once an application is written and compiled it can run on any architecture (for example an application written for the ARM architecture can run on the x86 architecture with no modification to the original code) offering a high degree of portability [12].

Another advantage is that due to its open-source availability the end user can modify the entire operating system according to its needs and tastes or even modify the working parameters of the device (modify GPS, CPU frequency scaling, CPU overclocking). The only problem is that in order to gain high level access to the Android operating system the device needs to be rooted (gain superuser privileges), a practice that although Google is not against the majority of the device manufactures are resulting in voiding the warranty. However, workarounds have been found by users to prevent the loss of warranty by a practice called unroot, reverting the device to its initial state, eliminating any trace of the modifications.

DATA TRANSMISSION PROTOCOLS

IEEE 802.11

IEEE (Institute of Electrical and Electronic Engineers) is one of the three groups in the networking industry that set standards on how network equipments communicate with each other. 802.11 is a set of standards that use over-the air modulation techniques of the same protocol which provide the basis for Wi-Fi networks in the 2.4GHz, 3.6GHz and 5GHz frequency bands. The most popular and widespread protocols are 802.11a, 802.11b and 802.11g with 802.11n catching on quickly due its increased security and better transfer speeds but still not cost effective. Each protocol operates in a specific frequency range 802.11a using the unregulated 5GHz band, 802.11b/g the 2.4GHz band and 802.11n both 2.4GHz and 5GHz. Furthermore each frequency range is divided into a number of channels, specific to each protocol with a set of non overlapping channels (channels that don't use any part of their neighbor's frequencies making them unable to cause interferences with each other) [1].

802.11a

Due to the limited number of non-overlapping channels in 802.11b and g protocols and the large amount of cordless devices that operate in the 2.4GHz frequency range make 802.11a a good alternative. Relatively few devices operate in the unregulated 5GHz frequency spectrum making it perfect for overcrowded Wi-Fi areas. The advantages of using this set of frequencies are the increased number of channels (24 channels versus 11), increased number of non-overlapping channels (12 versus 3) and more room for data as on each channel as the channels assigned for 802.11a are wider(channel separation is 40MHz versus 20Mhz). This advantages leave 802.11a networks with a lower change of interference with other devices. Also 802.11a can achieve speeds up to 54Mbps making them adequate for bandwidth-consuming applications video streaming, data transfers) [1].

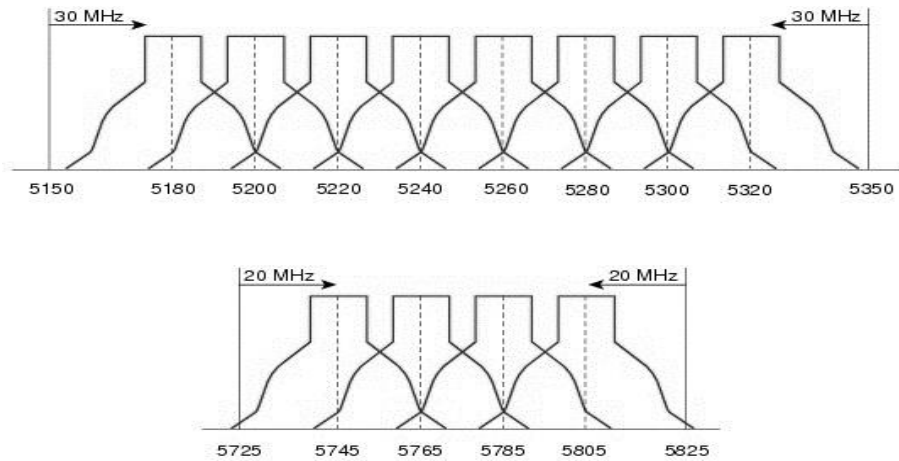


Figure 4. Non overlapping channels for 802.11a networks

The high carrier frequency comes however with a disadvantage: The range of 802.11a devices is slightly less than 802.11b/g due to the fact that the signals can't penetrate as far and they are absorbed by walls and other solid objects found in their paths. Also 802.11a is not compatible with 802.11b/g networks.

802.11b/g

802.11b/g networks operate in the unregulated 2.4GHz frequency spectrum and can be used in environments like offices and buildings due to their penetration power and higher signal range. The 2.4GHz frequency spectrum is divided into 11 channels 22 MHz wide with 3 non-overlapping channels (channels 1, 6 and 11) as shown in Figure 5. The problem is that each channel is placed 5 MHz apart from each other meaning that each channel is using the some of the same radio frequencies as the channels near it causing interferences [1].

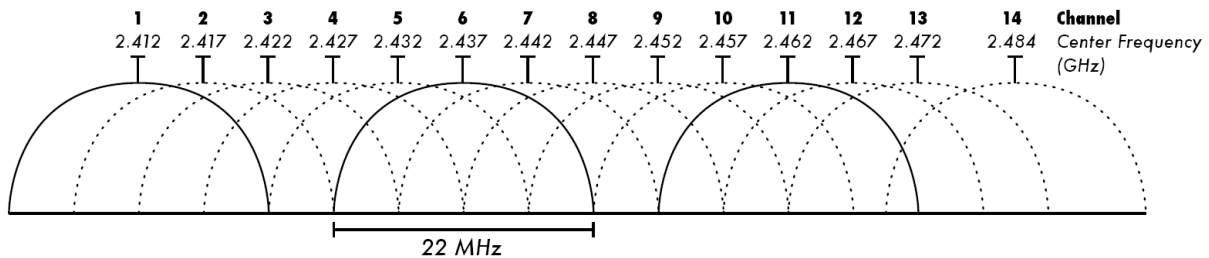


Figure 5. Channels in the 2.4GHz spectrum

802.11b networks can achieve theoretical maximum speeds up to 11Mbps although the real-world throughput is no more than 5Mbps. 802.11b's most common use is for internet connection sharing from a modem (a cable or DSL modem maxes out at 3 Mbps) for home or small office networks where there is no demand for high bandwidth [1].

802.11g is the next step in wireless networking after 802.11b using a new modulation scheme, and while it uses the same set of frequency and channels the maximum speed is greatly improved. 802.11g networks can achieve theoretical speeds up to 54Mbps (about 20Mbps real-world speed) thus making it 5 times faster than 802.11b networks [1].

802.11b and g networks can be interconnected due to 802.11g's backwards compatibility by slowing down the overall network speed to accommodate 802.11b devices. That means when an 802.11b client connects to an 802.11g access point, the access point speed is reduced (to about 22Mbps) making all b and g devices to operate at the same speeds [1].

The main disadvantages of 802.11b/g networks are the interferences with other devices that operate in the same frequency spectrum like cordless phones, Bluetooth devices and even microwave ovens. Other sources of interference are networks of the same type operating in the vicinity of each other due to the reduced channel separation making the devices unreliable in crowded Wi-Fi areas.

TCP/IP transmission protocol

The name TCP/IP refers to the most important suite of data communications protocols: Transmission Control Protocol and Internet Protocol. Its roots are in the 1969 Advanced Research Projects Agency ARPANET network built initially to study techniques for data communications. Due to its success many companies attached to it and started to use it for everyday data communications. Thus the Internet was born [10].

In order for computers to communicate efficiently between them protocols had to be used. A protocol is a set of rules for data communications.

Any attempt to reproduce or distribute this document or parts of it without mentioning the source website (<http://android.webook.ro>) is forbidden.

The TCP/IP protocol suite consists in four layers: the Link Layer, the Internet Layer, the Transport Layer and the Application Layer.

The Link Layer contains the technology to communicate with the network the host is primary attached providing basic connectivity functions by interacting with the networking hardware in the computer [10].

The Internet Layer facilitates the interconnection of network by providing communication between multiple links of a computer. It contains the Internet Protocol which allows hosts on a network to be identified and located [10].

Data transmission between hosts is handled by the third layer, the Transport Layer by using protocols like Transmission Control Protocol and User Datagram Protocol.

Application-based interaction between hosts is handled by the Application Layer. This layer contains a set of protocols for a vast array of data communication services like Telnet, FTP, DNS, SMTP, POP or IMAP.

PERSONAL CONTRIBUTION

The main purpose of this project was to create a low cost reconnaissance electronic system (a spy robot) with easy to acquire hardware from devices used in everyday life. Any practical implementation of this family of electronic devices needs two components: 1) a mobile unit (the robot) and 2) a remote control unit. These subsystems use a bidirectional communication channel for transmitting information between them. Both components are usually developed around a computing platform such as PC, laptop, smartphone, iPhone, etc.

In order for the robot (the mobile unit) to function properly several aspects had to be taken into consideration:

- A computer like platform that would allow control of the robot functions.
- A set of motors that would allow the movement of the robot.
- A method to power the motors according to the instructions received
- A reduction gearbox to allow fine movement over terrain.
- A Radio Transmitter/Receiver unit, natively coupled to the mobile unit.
- A set of sensors (including complex devices such as cameras, GPS receiver unit, gravitational sensor, etc.) to receive data from the surrounding environment.

In both projects described in the following sections we use the Wi-Fi communication channel (native channel of radio communication for laptops and smartphones).

C.O.R.E v1.0

C.O.R.E v1.0 is the name of the 1st device built by taking into considerations the points mentioned above. The robot platform (that would receive commands from the control computer), is built around an ASUS laptop. This computer (unfortunately with a burned video chip) has 1 GB of RAM, a 1.6GHz dual-core processor, a sound card and an 802.11a/b/g Wi-Fi card. Everything else was stripped down leaving just the housing for the motherboard and the battery. All other components were attached to the housing as described in Figure 7.

To move the device 4 CD-ROM motors with their gearboxes were attached powered from a car stereo amplifier with medium performances tied to an 11.1V laptop battery and receiving signals from the laptop's soundcard. The device is able to move forward or backwards and also steer left or right by using a tracked vehicle design by turning 2 motors in a direction and the other 2 in the opposing direction.

To receive data from the surrounding environment a 640x480 USB webcam was mounted on top of the device to stream video back to the computer used for control. The webcam has an FPS average of about 15fps.

The control computer can be any computer or mobile device that has installed a remote desktop client program for control and a web browser to receive streaming media.

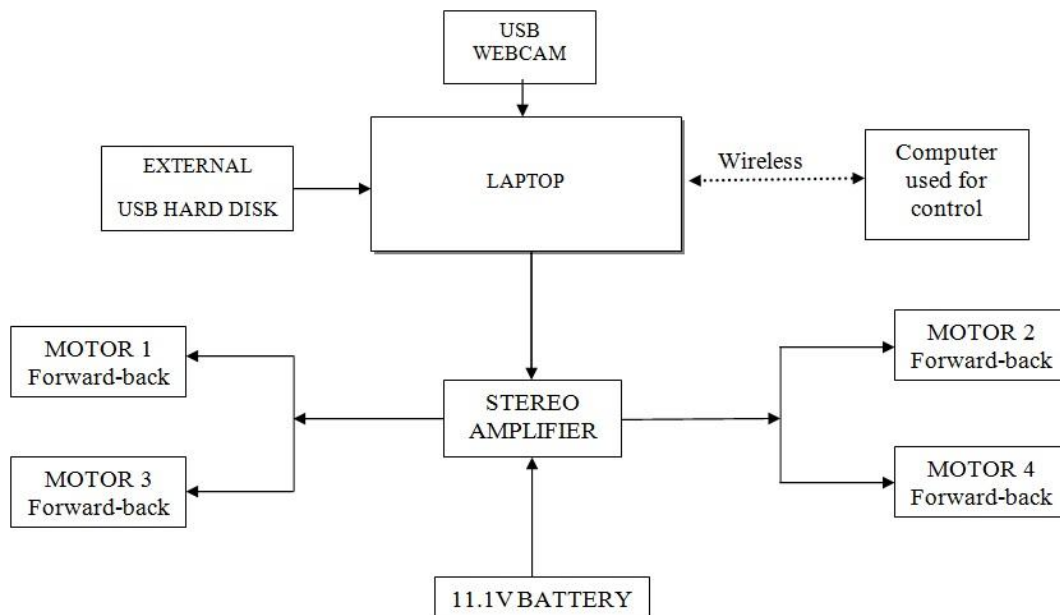


Figure 6. C.O.R.E v1.0 hardware block diagram

In Figure 6 we have the hardware block diagram explaining the working principle of C.O.R.E v1.0. The laptop boots the operating system from the external USB hard disk. After the operating system is started a connection between the laptop and the computer used for control is established via ad-hoc wireless network. Images from the webcam connected to the laptop are

Any attempt to reproduce or distribute this document or parts of it without mentioning the source website (<http://android.webook.ro>) is forbidden.

streamed through the network and received on the control computer. Commands are received by means of wireless network communication by the laptop (in this case a sound file is played). The sound transformed into an electric signal by the soundcard inside the laptop and passed out to the amplifier which in turn powers the motors allowing the whole device to move.

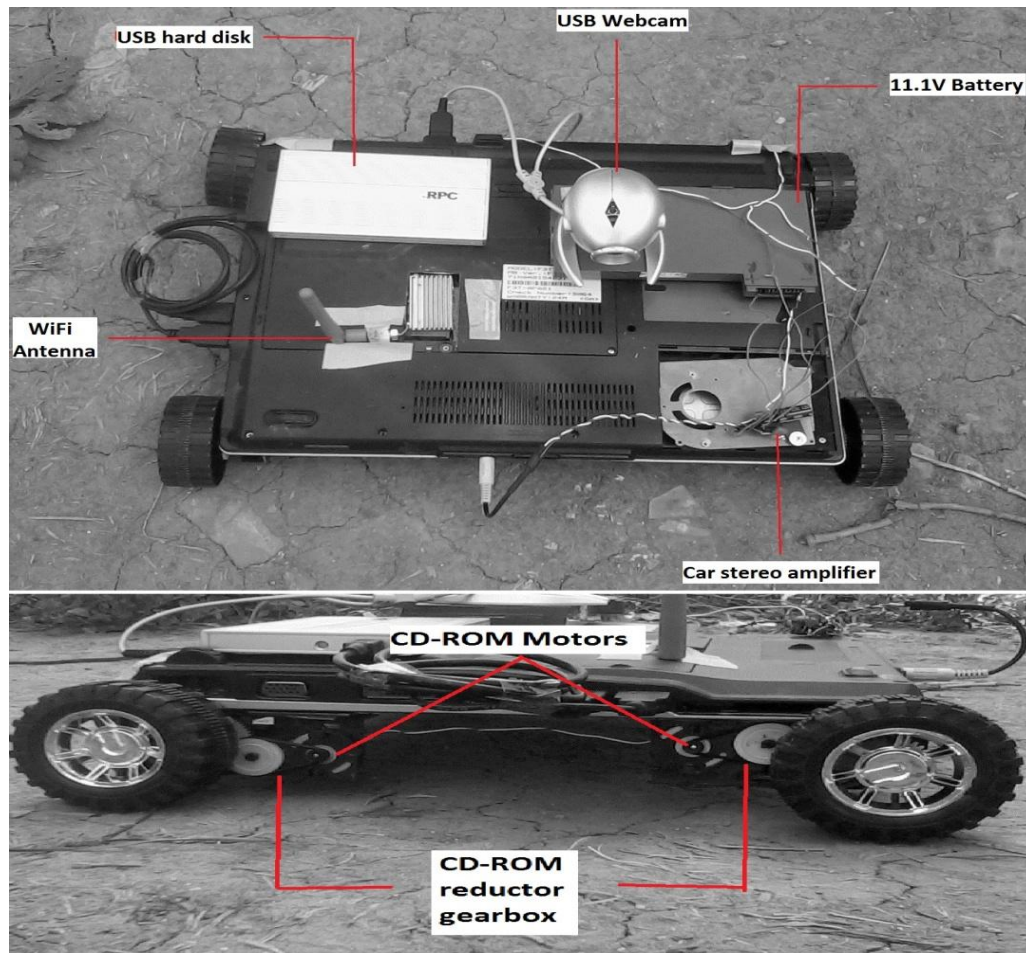


Figure 7. C.O.R.E v1.0 diagram

The operating system used to manage the device functions and facilitate communications was the Linux distribution Ubuntu 10.4 along with all updates and patches [9]. The main reason in choosing this operating system was its free availability, long time support, vast documentation, user friendly interface, and easy configuration steps. To assure portability the operating system was installed on a USB hard disk. This way any laptop with the minimum

required components could be used by simply plugging the USB hard disk and booting the operating system. Other advantage is lack of driver installation routine by automatically detecting and starting kernel modules that contain device drivers at system startup. There is also a high interoperability between Linux and Windows (the operating system used on the control computer) allowing services to communicate between the 2 operating systems as described in the software diagram from Figure 8.

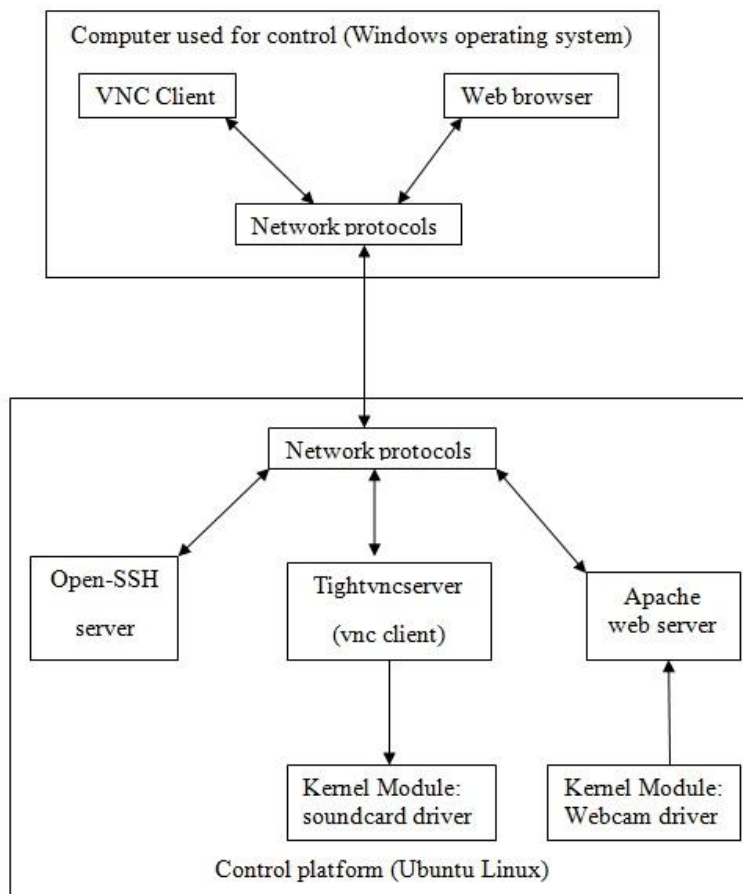


Figure 8. Software diagram

Operating system installation

Due to the fact that the video chip on laptop used on C.O.R.E is burned the installation had to be done on another computer which was very easy to achieve due to the nature of Linux which allows it to be installed on external media and to auto configure the kernel modules responsible for hardware at boot. The installation process is fast and straightforward. The user is asked to select a partition scheme for the storage media, select a time zone, a keyboard layout and type in a user and password. After these steps the system installs the necessary file and asks for a reboot. After the first boot the operating system is ready to use. The next step is to update the system with the latest updates and patches. To do this we can either use the system update function in Ubuntu or the terminal for a better insight of any problems encountered. To update using the terminal the following commands:

```
sudo apt-get update
```

```
apt-get upgrade
```

The command **sudo** is used to obtain superuser access rights required to install programs on any Linux operating system. Once typed before the command the access rights will remain until the terminal window is closed.

In order to have remote access to the C.O.R.E v1.0 2 services must be installed: TightVNC server and Open-SSH. TightVNC server will be used for remote control and is very useful because it creates a virtual framebuffer in the RAM memory, allowing the X Window Server to start even if a video card is not present (or in this case burned). The X Window System is required in order to obtain a remote desktop connection to the robot. The role of TightVNC is to create a virtual desktop and export it to the computer used for control. Open SSH is a console-based remote administration application that will be used in low-bandwidth situation and also for configuring and installing applications faster than in a remote desktop environment. These programs are installed by typing the following commands in the terminal window:

```
apt-get install tightvncserver
```

```
apt-get install Open-SSH
```

Any attempt to reproduce or distribute this document or parts of it without mentioning the source website (<http://android.webook.ro>) is forbidden.

The programs are downloaded from the Ubuntu repositories and installed on the operating system along with the required dependencies (set of libraries required by the programs in order to run). After the installation is completed **tightvncserver** needs to be provided with a password in order to restrict access to the system. To create a password for **tightvncserver** the following command needs to be typed in a terminal window:

```
vncpasswd
```

A dialog line will appear asking for the password and retype for confirmation and the password is created. In order to connect to the C.O.R.E v1.0 a VNC client is required to be installed on the computer used for control. Before connecting, the IP of C.O.R.E v1.0 and the password created by **vncpasswd** are required. After they are typed in on the computer used for control a window with the desktop of C.O.R.E v1.0 appears and the user can interact with the operating system as it was in front of it.

Connection to Open-SSH requires a ssh client on the computer used for control. In the SSH client window the IP, user and password are asked and after typing them a terminal on the C.O.R.E v1.0 opens. From now on any commands typed in the SSH terminal are executed C.O.R.E v1.0.

The next step is to configure **tightvncserver** to start automatically at boot time and create a virtual desktop to which the computer used for control would connect. In order to do so the following lines must be added at the end of the configuration file `/etc/rc.local`:

```
su core -c "tightvncserver"
```

```
exit 0
```

The final step is to create a WiFi ad-hoc network which would start at boot time and would ensure communication with the computer used for control. This is achieved by typing in the terminal window:

```
sudo ifconfig wlan0 down
```

```
ifconfig wlan0 channel 1 essid C.O.R.E mode ad-hoc
```

Any attempt to reproduce or distribute this document or parts of it without mentioning the source website (<http://android.webook.ro>) is forbidden.

ifconfig wlan0 up

After the last step is completed the network C.O.R.E is created. The local host receives the address 10.42.43.1, the address that will be used in order to connect to the VNC server and the video streaming server. The system's remote desktop capabilities are now configured and the hard disk can be moved on C.O.R.E v1.0.

Installation and configuration of the video streaming server

In order to receive a video stream on the computer used for control a video streaming server must be installed on the device. The video streaming server is composed of 2 programs: the *apache web server* and *webcam server*. Both programs are installed by typing in a terminal window:

```
apt-get install webcam-server
```

```
apt-get install apache2
```

In order for webcam-server to start at boot a file called webcam-server must be placed in `/etc/init.d`. The file webcam-server contains a startup script that will allow the control of the program as a daemon. The startup script's contents are described in Appendix 1. webcam-server startup script `/etc/init.d/webcam-server`. The script is made executable by typing in a terminal window:

```
sudo chmod +x /etc/init.d/webcam-server
```

```
sudo update-rc.d webcam-server defaults
```

Webcam-server uses a Java applet installed on C.O.R.E v1.0 to show the images received from the webcam in a browser window. The Java applet is located in `/usr/share/doc/webcam-server/applet/`. For the control computer to connect and receive a video stream the contents of this folder must be moved into the root folder of the Apache web server of the device (`/var/www`). The applet is configured by default to stream video at 1 frame per second at 320x240 pixels and to use the localhost as the streaming domain (streaming domain refers to the IP or class of IP addresses that can connect to the server and receive the video feed). In order to fix this the file `/var/www/webcam.html` must be modified as indicated in Appendix 2. webcam-server configuration file `/var/www/webcam.html`

Any attempt to reproduce or distribute this document or parts of it without mentioning the source website (<http://android.webook.ro>) is forbidden.

The connection to the video streaming server is made by typing in any browser window the IP address and port of the streaming server: <http://10.42.43.1:8888> from any computer connected to the network created by C.O.R.E v1.0.

Movement and remote control

A cheap and easy to build system was needed for movement and control of the device. The solution came by using a Phillips TDA1557Q car stereo amplifier that would receive electrical signals from the laptop soundcard, amplify them and further on power the motors that would turn the wheels and move the device (Figure 9). A gearbox from a CD-ROM drive was used in order to slow down the speed of the motors, obtain a fine control over the device and also to increase the torque.

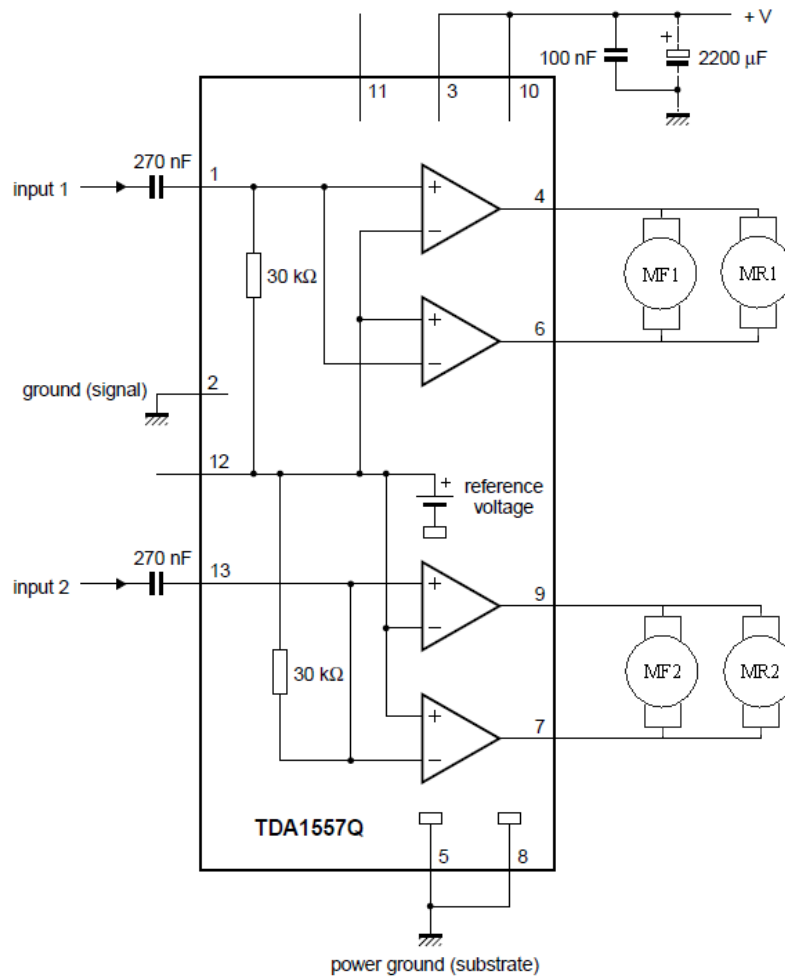


Figure 9. Functional diagram of the amplifier used to control the motors

For control signal received by the amplifier is created by playing a sound file on C.O.R.E v1.0. By sending positive or negative mono audio signals the device moves forward or backwards as shown in Figure 10.

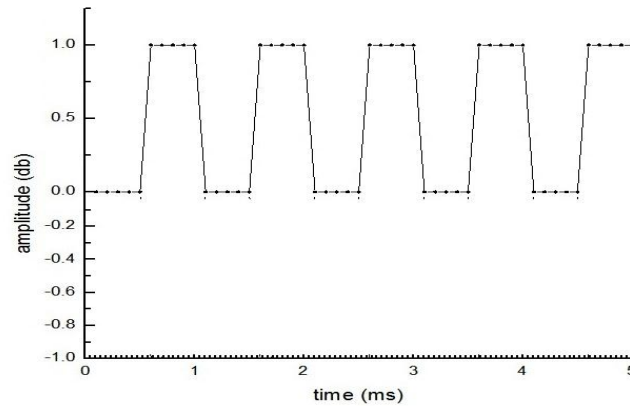


Figure 10. A mono signal used to move the device forward

To rotate the device left or right a positive signal must be sent to one channel and a negative signal to the other in order to obtain a tank-like steering. The stereo signal generated is described below, in Figure 11.

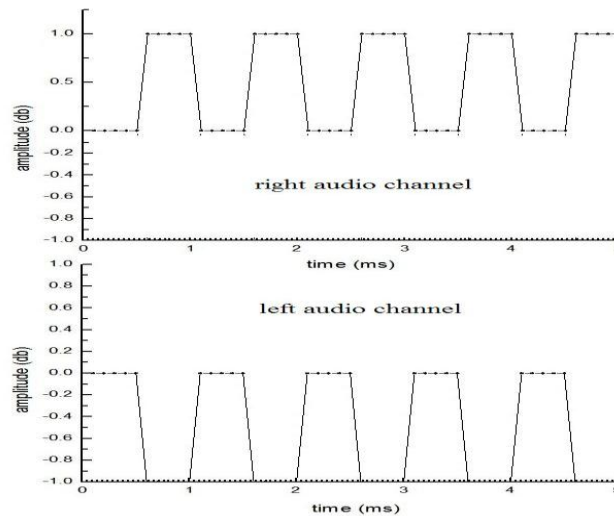


Figure 11. A stereo signal used to turn the device to the left

The speed of the device is controlled by using different frequencies for the signal, the lower the frequency the faster the device moves. The signals were generated using the program

Any attempt to reproduce or distribute this document or parts of it without mentioning the source website (<http://android.webook.ro>) is forbidden.

Audacity 1.3 Beta and saved as .mp3 files [2]. A set of 2 signals were created for each type of movement in order to move the device fast when needed or to slowly maneuver. The 2 signals have frequencies of 100Hz and 200Hz.

The remote control system is simple and intuitive. The .mp3 files containing the signals for control were placed on the hard disk of the C.O.R.E v1.0. By connecting to the C.O.R.E v1.0 via the VNC client and opening the .mp3 files in any music player the device will move according to the signals received from the specific .mp3 file that is played at the given time. A built in function in Ubuntu allows any user to preview any audio file by hovering the mouse cursor over it, allowing a fast and very comfortable method of control. Once the mouse is moved from the icon of the audio file the playback ends making the device to stop. Another possibility for control is by SSH and using a command line media player in order to play the given audio files.

The overall movement speed of the device is limited by the webcam that can only stream at 15 frames per second making it very hard to control at high speeds because of the latency that occurs. The motors are powerful enough to propel the device on almost any flat surfaces being able to overcome small obstacles like rocks and grass. The 4 wheel traction system used diminishes the chances to remain stuck from the unevenness of the ground.

The maximum range for wireless control is about 140m outdoor and 125m indoor. The range can be expanded by upgrading the antenna used by the C.O.R.E v1.0 or by using a 3G USB modem making it accessible from any computer connected to the Internet anywhere in the world. When a 3G USB modem is used, access to the device is made by using the IP address provided by the modem.

Any attempt to reproduce or distribute this document or parts of it without mentioning the source website (<http://android.webook.ro>) is forbidden.

C.O.R.E v2.0

After C.O.R.E v1.0 was finished the author was looking for ways to lower the overall costs and make the device smaller in order to be used in hard to access areas and less easy to be spotted. A better, low cost camera was needed in order to allow maneuvers at higher speeds and a better streaming image and a way to determine the position of the device in case of network failures due to external factors. The cost of an additional GPS device was high so alternatives had to be found. In the search for a device that would accomplish the same functions as the laptop but smaller and lighter and with the capabilities described earlier, the solution found was the use of an Android-powered smartphone.

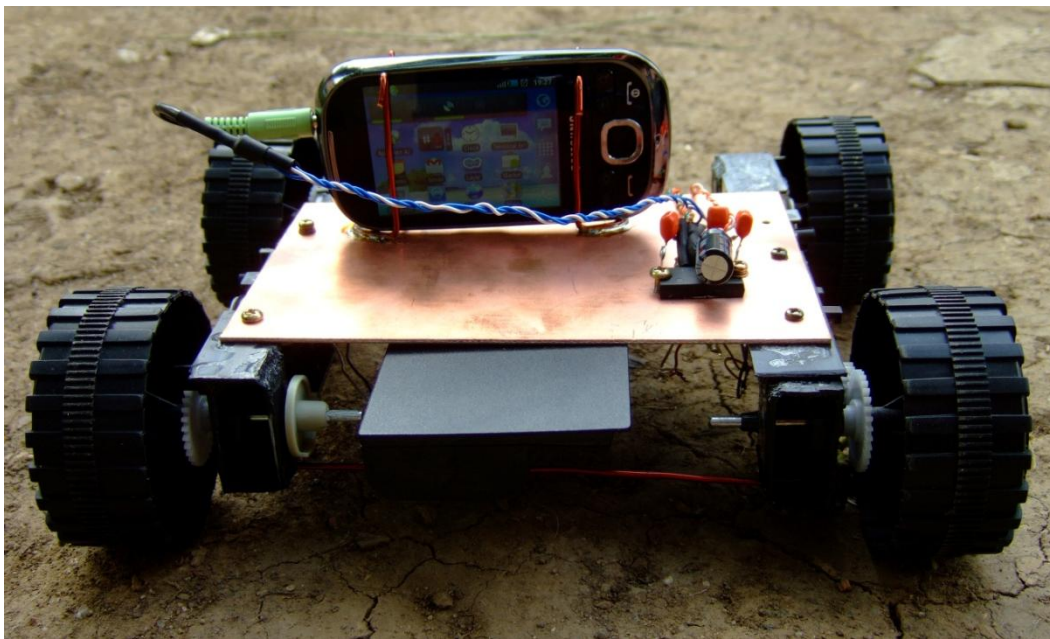


Figure 12. C.O.R.E v2.0 - rear view

In order for the device to remain cost effective the cheapest Android-powered smartphone was used: Samsung Galaxy I5500. The device has a 600MHz ARM v11 processor, 256 MB of RAM, 802.11b/g wireless, 170MB internal storage + 1GB MicroSD card (upgradable up to 16GB), a 2 MP camera, GPS and accelerometer running Android 2.2 (Froyo). The autonomy is about 520h in stand-by and 9h when in use which is a huge step from the maximum autonomy that the laptop could achieve.

For C.O.R.E v2.0 the working principle is the same as C.O.R.E v1.0, a stereo amplifier is used to power the motors that in turn move the device (Figure 13). The only difference is that the laptop, external hard disk and webcam are replaced by the smartphone. This makes C.O.R.E v2.0 about ¼ the size of its predecessor. The amplifier receives the signals needed for control from the phones jack on top of the smartphone.

Data from the surrounding environment is received from the integrated 2MP camera by the same means as C.O.R.E v1.0 – video streaming. Also other data sets can be gathered such as magnetic fields information, sound levels, ground movements and vibrations.

The control method is the same as used for the first device built by connecting via wireless network and using any web browser to connect to an interface similar to the VNC server.

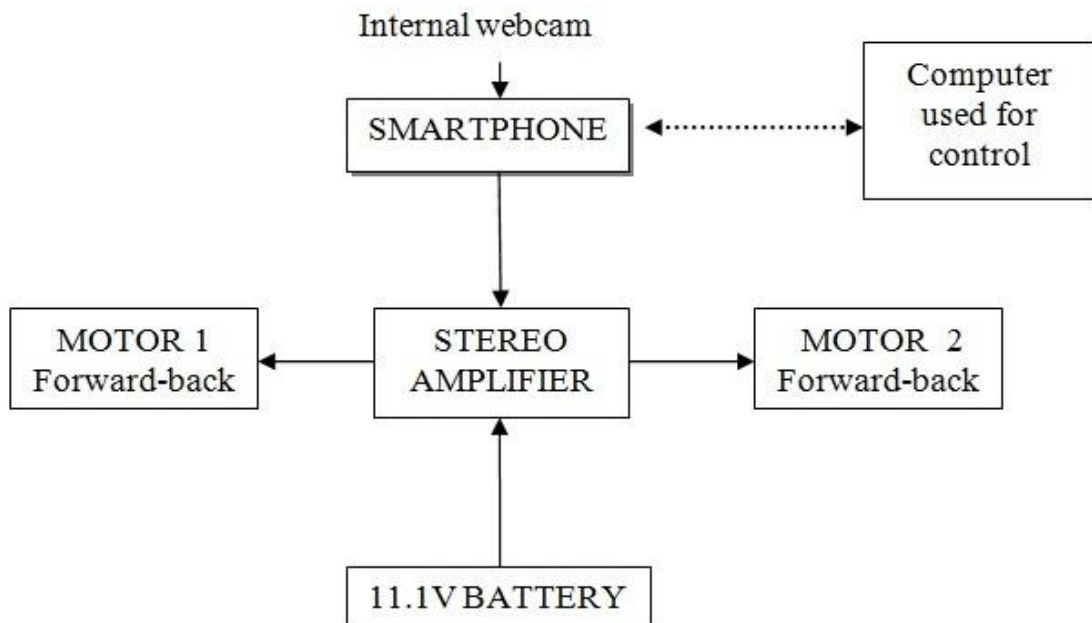


Figure 13. Hardware block diagram

Android and video streaming configuration

The configuration needed for the Android operating system to adapt it to the needed specification is minimal. The only requirement to the system is to be rooted in order for the remote administration applications to work. To root the device an application called Universal Androot is needed. The application can be downloaded directly to the smartphone by visiting the following address in the internal browser:

<http://forum.xda-developers.com/attachment.php?attachmentid=377352&d=1281451779>

After the application is installed root access is obtained by simply running the program and selecting the “Root this device” option. To prevent loss of warranty an “Unroot” option is available to be used in the case that the device needs to be returned to the producer for repairs.

The rest of the configurations needed for the Android operating system are made from the internal settings menu of the smartphone.

In order to connect to the smartphone a wireless access point is needed. The access point is created and configured by navigating to

Menu > Wireless & networks > Tethering & portable hotspot > Portable Wi-Fi hotspot > on

The access point created will allow for other devices to connect to the smartphone to receive the video feed or for control purposes. The standard network security option is open but it can be changed to a password protected encrypted network to obtain a limited access communication environment.

To install a video streaming server on the Android operating system an application called IP Webcam is needed. The application can be downloaded and installed from the internal Android Market application. After the application is started a window appears asking for a login user and password (optional) and several quality options. These options will be selected and adapted as needed at a certain time or operation. By selecting start server the video streaming

Any attempt to reproduce or distribute this document or parts of it without mentioning the source website (<http://android.webook.ro>) is forbidden.

server is active and the control computer can connect to the IP address of the smartphone and port 8080 in order to receive the video feed. Multiple options to receive the video feed are available: directly in web browser, in a media player with stream support such as VLC or to use Skype to send the feed further over the Internet. For commodity and to have the video feed on top of all the other windows on the control computer VLC media player was used.

The next software component needed to have a functional device is an application that would allow remote administration. For this the application needed is Webkey. Webkey can be downloaded and installed from the Android Market. Once installed the remote administration platform needs to be configured by adding a user from the user settings menu and setting the permissions required. Access to the remote host is made by opening a browser window on the control computer with the IP address of the device (Figure 14). Besides the window with the device screen the application control interface allows the sending of commands and button inputs directly to the phone. The video from C.O.R.E v2.0 can be adjusted depending on the network speed or signal for a better control in low bandwidth situation. Connection to the Webkey remote administration interface can be made over the internet via the website <http://androidwebkey.com> when mobile internet is used. The device sends an identification number to the website and access is made using the Google account registered with the smartphone.

Any attempt to reproduce or distribute this document or parts of it without mentioning the source website (<http://android.webook.ro>) is forbidden.

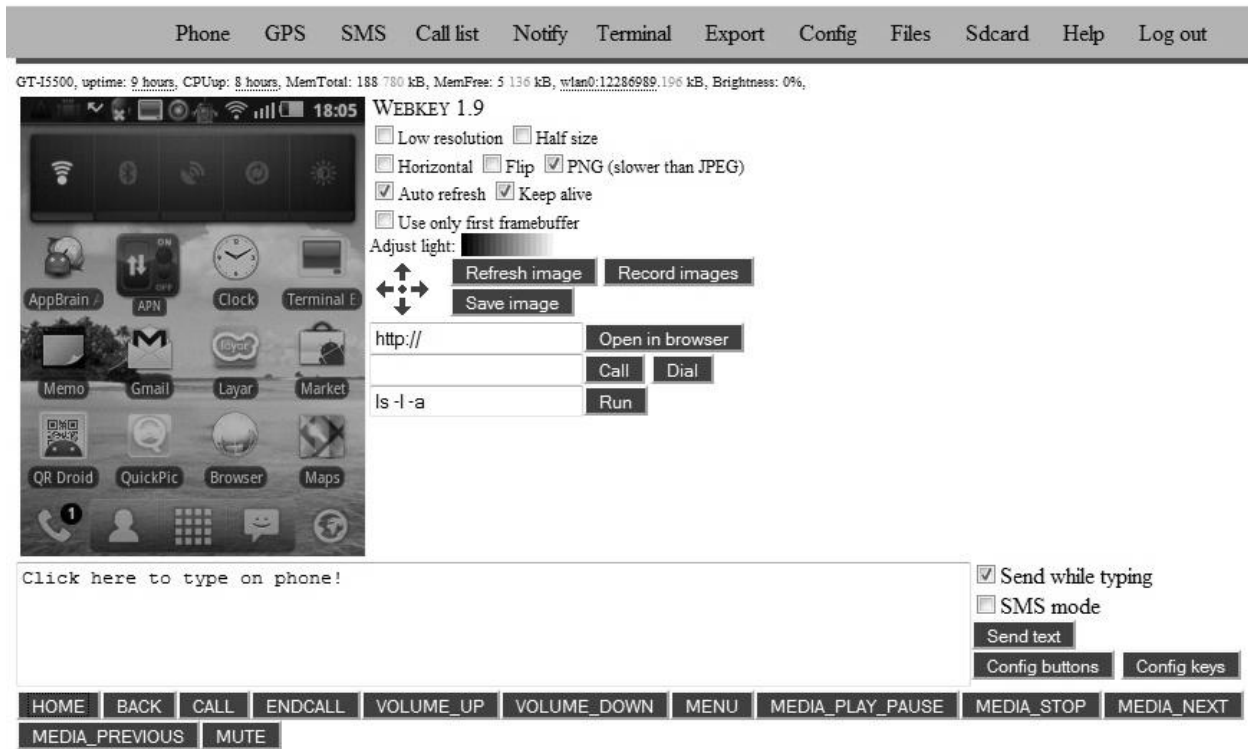


Figure 14. The Webkey remote administration interface

Another useful application is GPS Position Logger for Android, an application that stores the GPS position of the device at a certain time interval defined by the user. The application stores GPS data in a .kml file on the external SD card, allowing the recreation of the path followed by the device in applications like Google Earth or Google Maps.

All the software necessary to have a functional device have been installed and it is ready to receive remote administration requests. The working principle of the software part of the device is almost the same as in C.O.R.E v1.0 except some minor differences due to the nature of the Android operation system which uses Dalvik (Java virtual machine) on top of the Linux kernel to manage and launch applications (Figure 15).

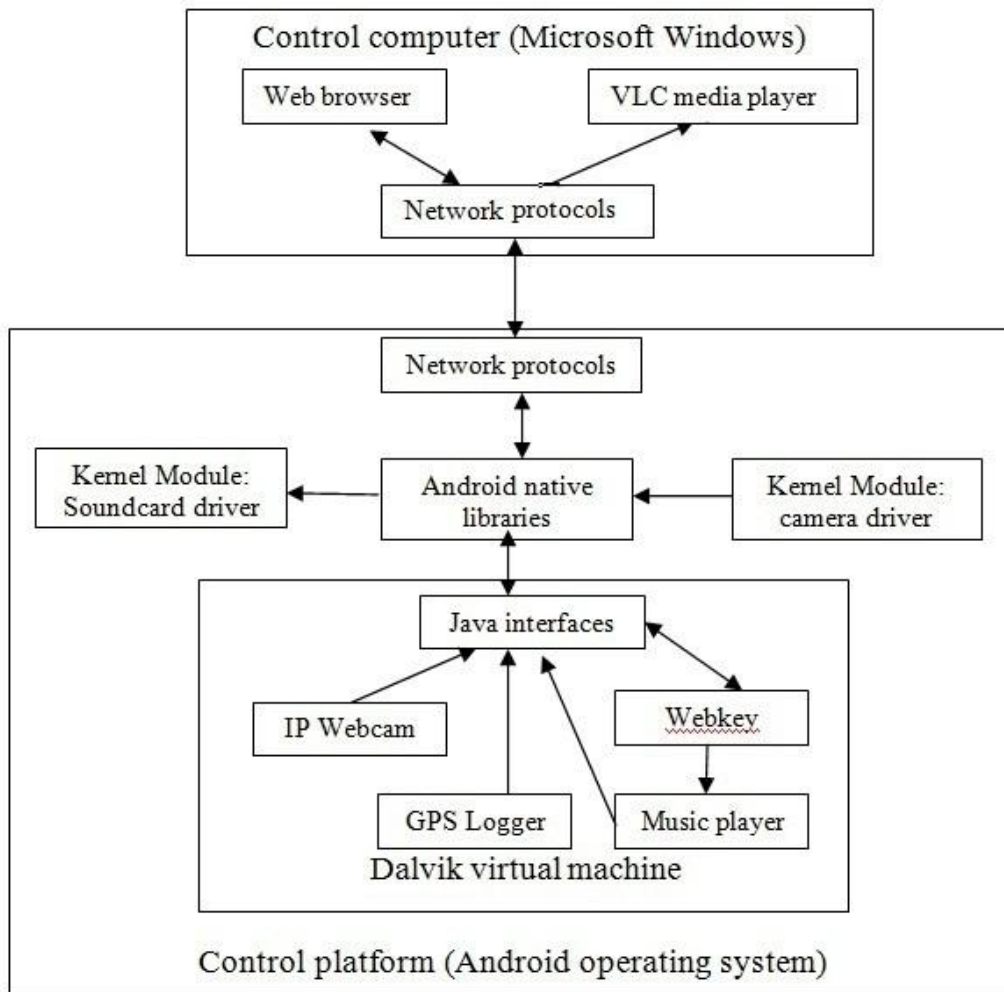


Figure 15 Software diagram

Movement and remote control

The movement of the device is similar to C.O.R.E v1.0. When a sound file is played on the smartphone the resulting signal from the internal soundcard is sent to the amplifier to control the motors and their speed. Instead of using different set of audio files for speed the volume function is used to achieve 3 different speeds for maneuvering. The motors and reduction gearboxes are the same as use in the first device (from a CD-ROM drive).

Any attempt to reproduce or distribute this document or parts of it without mentioning the source website (<http://android.webook.ro>) is forbidden.

In order to control the device a user has to connect to the AP of the smartphone and type its IP address in a browser. Next a control interface will appear allowing a remote desktop session along with other functions like the ability to upload and download files to and from the smartphone, gain access to the internal Linux terminal and phone functions like call and send SMS (call function may be used to eavesdrop on the surrounding environment). Movement commands are given by playing through the remote desktop function the proper audio file in any media player on the smartphone.

With the use of live video streaming the device can be guided in real time very easily by following the video feed the VLC player window and issuing commands to play .mp3 files through the remote desktop interface. The integrated camera, besides video streaming can also take quality photos of the environment and due to a built in function of the Android operating system a GPS tag can be added to indicate the exact location where the photos were taken.

The current device customization is minimal in order to demonstrate the working principles but there are enough application designed for it to adapt it for any situation needed. Applications like Tricoder can detect ground movement and vibrations, G forces, acoustic levels, magnetic field strength around the device, geographical data and wireless signals. Other applications can analyze wireless networks and decode information packets send between hosts. By using the Android Lost application the device can be recovered in case the communication with the control computer is interrupted and it can no longer be established. The application allows control the internal smartphone functions by SMS or from www.androidlost.com. Thus, if the device is lost a SMS can be sent to locate the phone and send back the GPS coordinates. Data wipe of the internal memory and SD card options are available as well as several lock functions in the case the device is in danger of falling into the wrong hands or the internal data acquired by the device has a secret nature. All the applications needed to adapt the device to certain needs can be found on the Android Market.

The range of C.O.R.E v2.0 is about 100m outdoor by using the internal wireless. By using the mobile internet available for the smartphone SIM card the range will only be limited by the placement of the cell phone towers giving it almost a country-wide coverage. When mobile internet is used the IP used to connect is the IP provided by connecting to the internet.

Conclusions

All of the aspects suggested by the author were achieved and it was found that the project had indeed achieved the desired performances and low building costs.

The configuration of the Ubuntu and Android led to better knowledge and understanding in the domain of the Linux-based operating systems and communication protocols. Because of the high degree of information available the installation and modification of the target operating systems and integration of the newest technologies for communication had become extremely easy allowing the deployment of complex reconnaissance devices in a very short time.

The only major difficulties encountered was during the building of C.O.R.E v1.0 because of the burned video chip that wouldn't allow standard VNC servers to run in order to obtain a remote desktop environment. The solution was found after a period of documentation by using TightVNC server, a VNC server that uses a virtual framebuffer in the RAM memory to create a virtual desktop on the host computer. Another problem encountered was the overheating of the Wi-Fi card that would lead to frequent disconnects after a few minutes from boot. The problem was solved by using a heat sink and thermo conducting silicone paste between them.

In the future the author aims to build a flying reconnaissance device with a higher degree of performance incorporating multiple functions programmable by the final user according to his specific needs and maintaining the costs as low as possible.

Appendixes

Appendix 1. webcam-server startup script /etc/init.d/webcam-server

```
#!/bin/sh
SERVER_BIN=webcam-server
LOCK_FILE=/var/lock/$SERVER_BIN
RTRN=0
OPTIONS='-v -g 320x240 -p 8888 -c CORE'
start() {
[ -f $LOCK_FILE ] && echo "$SERVER_BIN already started"
[ -f $LOCK_FILE ] && return
echo -n "Starting $SERVER_BIN: "
export LD_PRELOAD=/usr/lib/libv4l/v4l1compat.so
nohup $SERVER_BIN $OPTIONS > /dev/null 2>/dev/null &
RTRN=$?
[ $RTRN -eq 0 ] && echo Started! || echo FAIL
[ $RTRN -eq 0 ] && touch $LOCK_FILE
}
stop() {
[ -f $LOCK_FILE ] || echo "$SERVER_BIN is not running"
[ -f $LOCK_FILE ] || return
echo -n "Stopping $SERVER_BIN: "
pkill -f "$SERVER_BIN $OPTIONS"
RTRN=$?
rm -f $LOCK_FILE
[ $RTRN -eq 0 ] && echo Stopped! || echo FAIL
}
case "$1" in
start)
start
;;
stop)
stop
;;
restart)
stop
start
;;
*)
echo "Usage: $0 {start|stop|restart}"
RTRN=1
```

Any attempt to reproduce or distribute this document or parts of it without mentioning the source website (<http://android.webook.ro>) is forbidden.

```
esac
exit $RTRN
```

Appendix 2. webcam-server configuration file /var/www/webcam.html

```
<html>
<head>
<title>WebCam</title>
</head>
<APPLET CODE = "WebCamApplet.class" archive="applet.jar" WIDTH =
"640" HEIGHT = "480">
<param name=URL value="http://10.42.43.1:8888">
<param name=FPS value="90">
<param name=width value="640">
<param name=height value="480">
</APPLET>
</body>
</html>
```

Any attempt to reproduce or distribute this document or parts of it without mentioning the source website (<http://android.webook.ro>) is forbidden.

Appendix 3. TDA1557Q data sheet

INTEGRATED CIRCUITS

DATA SHEET

TDA1557Q

2 x 22 W BTL stereo car radio
power amplifier with speaker
protection

Product specification
File under Integrated Circuits, IC01

May 1992

Philips
Semiconductors



PHILIPS

557Q data sheet

2 x 22 W BTL stereo car radio power amplifier with speaker protection

TDA1557Q

FEATURES

- Requires very few external components
- High output power
- Low offset voltage at output
- Fixed gain
- Good ripple rejection
- Mute/stand-by switch
- Load dump protection
- AC and DC short-circuit-safe to ground and V_P
- Thermally protected
- Reverse polarity safe
- Capability to handle high energy on outputs ($V_P = 0$)

- Protected against electrostatic discharge
- No switch-on/switch-off pop
- Flexible leads
- Low thermal resistance.

GENERAL DESCRIPTION

The TDA1557Q is a monolithic integrated class-B output amplifier in a 13-lead single-in-line (SIL) plastic power package. The device contains 2 x 22 W amplifiers in BTL configuration and has been primarily developed for car radio applications.

QUICK REFERENCE DATA

SYMBOL	PARAMETER	CONDITIONS	MIN.	TYP.	MAX.	UNIT
V_P	positive supply voltage range	operating	6.0	14.4	18	V
		non-operating	–	–	30	V
		load dump	–	–	45	V
I_{ORM}	repetitive peak output current		–	–	4	A
I_{tot}	total quiescent current		–	80	–	mA
I_{sb}	stand-by current		–	0.1	100	μ A
I_{sw}	switch-on current		–	–	60	μ A
$ Z_i $	input impedance		25	–	–	k Ω
T_{XTAL}	crystal temperature		–	–	+150	$^{\circ}$ C
Stereo application						
P_O	output power	THD = 10%; 4 Ω	–	22	–	W
SVRR	supply voltage ripple rejection	$R_S = 0$; $f = 100$ Hz to 10 kHz	45	–	–	dB
$ \Delta V_O $	DC output offset voltage		–	–	250	mV
α	channel separation		40	–	–	dB
$ \Delta G_V $	channel unbalance		–	–	1	dB
G_V	closed loop voltage gain		45	46	47	dB

ORDERING INFORMATION

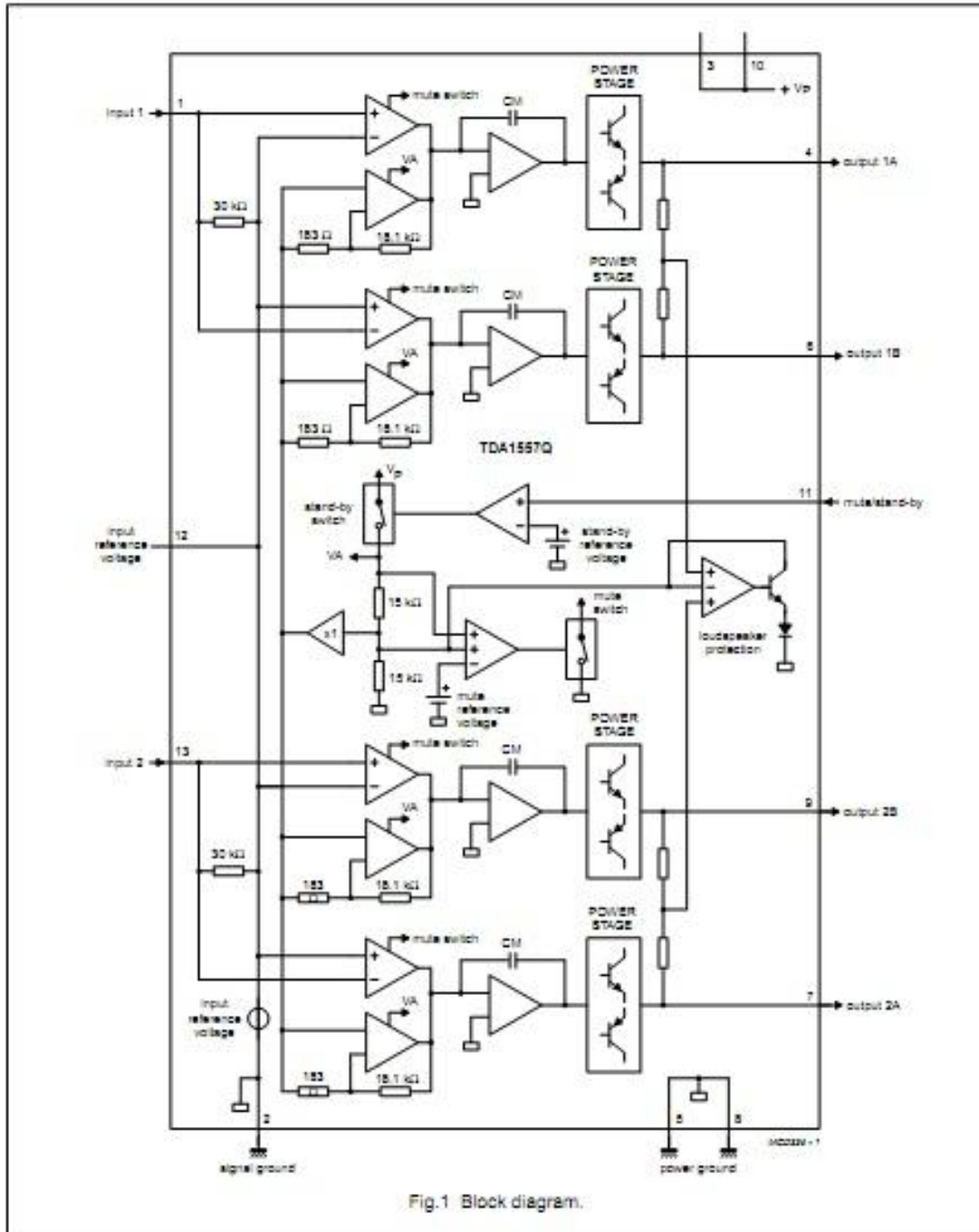
EXTENDED TYPE NUMBER	PACKAGE			
	PINS	PIN POSITION	MATERIAL	CODE
TDA1557Q	13	DIL	plastic	SOT141R

Note

1. SOT141-6; 1996 August 23.

2 x 22 W BTL stereo car radio power amplifier with speaker protection

TDA1557Q



2 x 22 W BTL stereo car radio power amplifier with speaker protection

TDA1557Q

PINNING

SYMBOL	PIN	DESCRIPTION
INP1	1	input 1
GND1	2	ground (signal)
V _{P1}	3	supply voltage 1
OUT1A	4	output 1A
GND	5	power ground 1
OUT1B	6	output 1B
OUT2A	7	output 2A
GND	8	power ground 2
OUT2B	9	output 2B
V _{P2}	10	supply voltage 2
M/SS	11	mute/stand-by switch
V _{ref}	12	input reference voltage
INP2	13	input 2

FUNCTIONAL DESCRIPTION

The TDA1557Q contains two identical amplifiers with differential input stages, and can be used for bridge applications. The gain of each amplifier is fixed at 46 dB. Special features of this device are:

- a. mute/stand-by switch
 - low stand-by current
 - low mute/stand-by switching current (low cost supply switch)
 - mute facility
- b. loudspeaker protection
 - when a short circuit to ground is made, which forces a DC voltage of ≥ 1 V across the loudspeaker, a built-in protection circuit becomes active and limits the DC voltage across the loudspeaker to ≤ 1 V
- c. the harmonic distortion at low frequencies can be decreased by connecting two diodes to ground at pin 12.

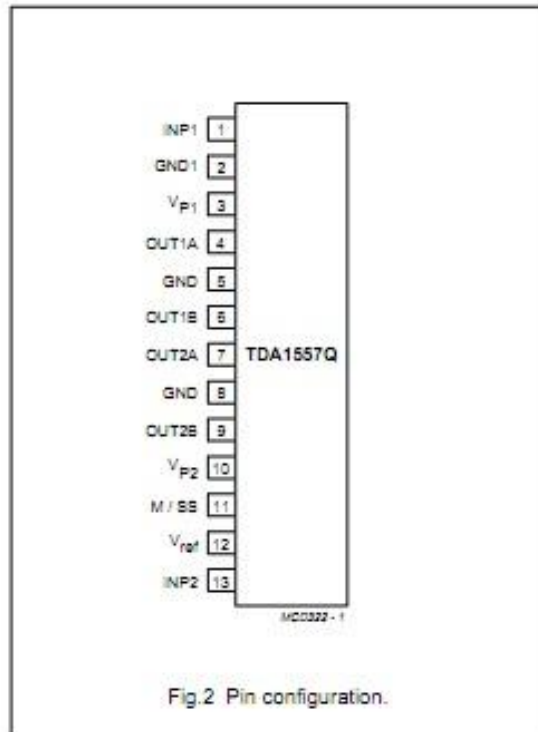


Fig.2 Pin configuration.

2 x 22 W BTL stereo car radio power amplifier with speaker protection

TDA1557Q

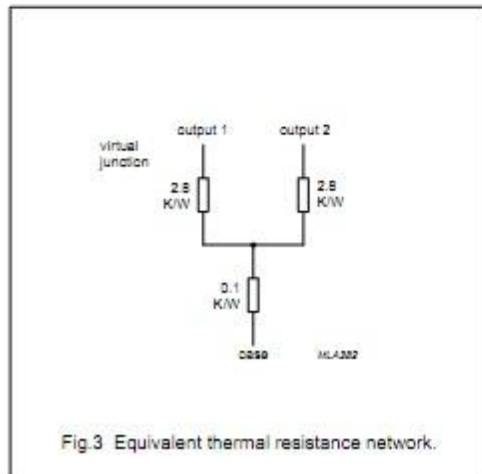
LIMITING VALUES

In accordance with the Absolute maximum System (IEC 134).

SYMBOL	PARAMETER	CONDITIONS	MIN.	MAX.	UNIT
V _F	positive supply voltage	operating	-	18	V
		non-operating	-	30	V
		load dump protected; during 50 ms; rise time ≥ 2.5 ms	-	45	V
V _{FSC}	AC and DC short-circuit safe voltage		-	18	V
V _{FR}	reverse polarity		-	6.0	V
	energy handling capability at outputs	V _F = 0	-	200	mJ
I _{OSM}	non-repetitive peak output current		-	6	A
I _{ORM}	repetitive peak output current		-	4	A
P _{tot}	total power dissipation		-	60	W
T _{stg}	storage temperature range		-55	+150	°C
T _J	junction temperature		-	+150	°C

THERMAL RESISTANCE

SYMBOL	PARAMETER	THERMAL RESISTANCE
R _{th vj-a}	from virtual junction to ambient in free air	40 K/W
R _{th vj-c}	from virtual junction to case (see Fig.3)	1.5 K/W



2 x 22 W BTL stereo car radio power amplifier with speaker protection

TDA1557Q

2 x 22 W BTL stereo car radio power amplifier with speaker protection

TDA1557Q

DC CHARACTERISTICS

$V_F = 14.4$ V, $T_{amb} = 25$ °C, unless otherwise specified. See note 1.

SYMBOL	PARAMETER	CONDITIONS	MIN.	TYP.	MAX.	UNIT
Supply						
V_F	positive supply voltage range	note 2	6.0	14.4	18	V
I_F	quiescent current		–	80	160	mA
V_O	DC output voltage	note 3	–	6.9	–	V
$ \Delta V_{OS} $	DC output offset voltage		–	–	250	mV
Mute/stand-by switch						
V_{sw}	switch-on voltage level		8.5	–	–	V
MUTE CONDITION						
V_{mute}	mute voltage		3.3	–	6.4	V
V_O	output signal in mute position	$V_I = 1$ V max; $f = 1$ kHz	–	–	20	mV
$ \Delta V_{OS} $	DC output offset voltage		–	–	250	mV
STAND-BY CONDITION						
V_{sb}	stand-by voltage		0	–	2.0	V
I_{sb}	DC current in stand-by condition	$V_{I1} \leq 0.5$ V $0.5 < V_{I1} \leq 2$ V	–	–	100 500	μ A μ A
I_{sw}	switch-on current		–	30	60	μ A
I_F	positive supply current	short-circuit to GND; note 4	–	5.5	–	mA
Loudspeaker protection						
$ \Delta V_{4-5, 7-8} $	DC voltage across R_L		–	–	1.0	V

Any attempt to reproduce or distribute this document or parts of it without mentioning the source website (<http://android.webook.ro>) is forbidden.

2 x 22 W BTL stereo car radio power amplifier with speaker protection

TDA1557Q

AC CHARACTERISTICS

$V_P = 14.4$ V; $R_L = 4$ Ω ; $f = 1$ kHz; $T_{amb} = 25$ °C; unless otherwise specified. See note 1.

SYMBOL	PARAMETER	CONDITIONS	MIN.	TYP.	MAX.	UNIT
P_O	output power	THD = 0.5%	15	17	–	W
		THD = 10%	20	22	–	W
		$V_P = 13.2$ V; THD = 0.5%	–	12	–	W
		$V_P = 13.2$ V; THD = 10%	–	17	–	W
THD	total harmonic distortion	$P_O = 1$ W	–	0.1	–	%
B	power bandwidth	THD = 0.5%; $P_O = -1$ dB with respect to 15 W	–	20 to 15 000	–	Hz
f_{low}	low frequency roll-off	-1 dB; note 5	–	25	–	Hz
f_{high}	high frequency roll-off	-1 dB	20	–	–	kHz
G_V	closed loop voltage gain		45	46	47	dB
SVRR	supply voltage ripple rejection	ON; note 6	34	–	–	dB
		ON; note 7	38	–	–	dB
		ON; note 8	45	–	–	dB
		MUTE; notes 6 and 7	45	–	–	dB
		stand-by; notes 6 and 7	80	–	–	dB
$ Z_i $	input impedance		25	30	36	k Ω
V_{no}	noise output voltage	ON; $R_S = 0$; note 9	–	325	500	μ V
		$R_S = 10$ k Ω ; note 9	–	350	–	μ V
		MUTE; notes 9 & 10	–	180	–	μ V
α	channel separation		40	–	–	dB
$ \Delta G_V $	channel unbalance		–	–	1	dB

Notes to the characteristics

- All characteristics are measured using the circuit shown in Fig.4
- The circuit is DC adjusted at $V_P = 6$ to 18 V and AC operating at $V_P = 8.5$ to 18 V
- At 18 V < V_P < 30 V, the DC output voltage $\leq V_P/2$
- Conditions: $V_{11} = 0$; short-circuit output to GND; switch V_{11} to MUTE or ON condition (rise time $V_{11} > 10$ μ s).
- Frequency response externally fixed.
- Ripple rejection measured at the output with a source-impedance of 0 Ω (max. ripple amplitude of 2 V) and a frequency of 100 Hz.
- Ripple rejection measured at the output with a source-impedance of 0 Ω (max. ripple amplitude of 2 V) and a frequency between 1 and 10 kHz.
- Ripple rejection measured at the output with a source-impedance of 0 Ω (max. ripple amplitude of 2 V) and a frequency between 100 Hz and 10 kHz. Pin 12 is decoupled with two diodes to ground.
- Noise voltage measured in a bandwidth of 20 Hz to 20 kHz.
- Noise output voltage independent of R_S ($V_{in} = 0$).

2 x 22 W BTL stereo car radio power amplifier with speaker protection

TDA1557Q

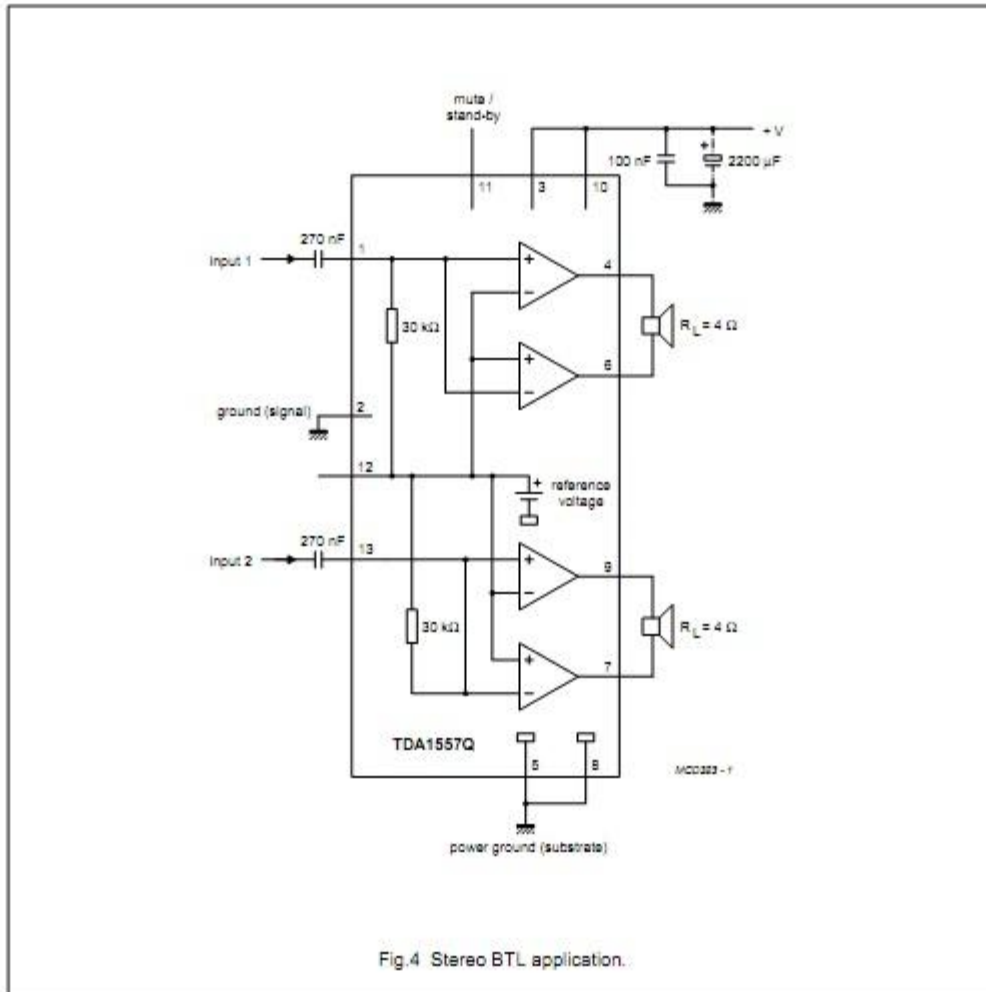


Fig.4 Stereo BTL application.

Bibliography

- [1] **Danny Briere, Pat Hurley.** *Wireless network hacks & mods for dummies*. s.l. : Wiley Publishing, 2005.
- [2] Download Audacity. [Online] [Cited: March 24, 2011.] <http://audacity.googlecode.com/files/audacity-win-unicode-1.3.13.exe>.
- [3] **Barkakati, Naba.** *Linux All-in-One Desk Reference for Dummies* . s.l. : For Dummies, 2006.
- [4] Linux. *Wikipedia.org*. [Online] [Cited: June 1, 2011.] <http://en.wikipedia.org/wiki/Linux>.
- [5] **IBM.** Linux Boot. *www.ibm.com*. [Online] IBM. [Cited: June 2, 2011.] <http://www.ibm.com/developerworks/library/l-linuxboot/index.html>.
- [6] Ubuntu Linux. *Wikipedia.org*. [Online] [Cited: June 11, 2011.] http://en.wikipedia.org/wiki/Ubuntu_linux.
- [7] **Rusling, David A.** The Linux Kernel. [Online] 1999. [Cited: June 3, 2011.] <http://www.cs.northwestern.edu/~pdinda/ics-f09/doc/linux-kernel.pdf>.
- [8] **Garrels, Machtelt.** The Linux Documentation Project. *The Linux Documentation Project*. [Online] 2008. [Cited: June 14, 2011.] <http://tldp.org/LDP/intro-linux/intro-linux.pdf>.
- [9] Ubuntu official download page. *Ubuntu.com*. [Online] [Cited: March 24, 2011.] <http://www.ubuntu.com/download/ubuntu/download>.
- [10] **Hunt, Craig.** *TCP IP Network administration*. s.l. : O'REILLY Media, 1997.
- [11] *What Is the Memory Capacity of the Human Brain?* **Reber, Paul**. s.l. : Scientific American Mind, 2010.
- [12] Android (operating system). *Wikipedia.org*. [Online] [Cited: July 20, 2011.] [http://en.wikipedia.org/wiki/Android_\(operating_system\)](http://en.wikipedia.org/wiki/Android_(operating_system)).