AirPlay using Raspberry Pi 3

T. Chandralekha[1], Aditya Radhakrishnan[2], Nrusinha Prasad M.[3]

[1]*Assistant Professor(OG) ,Department of Information Technology, SRM University, Chennai India*
[2]*Under Graduate Student ,Department of Information Technology, SRM University, Chennai India*
[3]*Under Graduate Student ,Department of Information Technology, SRM University, Chennai India*

1.INTRODUCTION

I would like to introduce Raspberry Pi as a world's most inexpensive and powerful Single Board Computer. Ever since the launch of Raspberry Pi from 2012, we have seen several version of it. This is world's cheapest microprocessor unit specially built for learner and makers. We can easily learn how software and hardware work together without been worrying about damage/cost. We can buy Raspberry Pi board with just somewhere around 35$. The cost of Pi allows newbies to celebrate mistakes and learn most out of it. Also Raspberry Pi has a huge community and plenty of online resources which make learning smooth.

AirPlay is a proprietary protocol stack/suite developed by Apple Inc. that allows wireless streaming between devices of audio, video, device screens, and photos, together with related metadata. Originally implemented only in Apple's software and devices, it was called AirTunes and used for audio only. Apple has since licensed the AirPlay protocol stack as a third-party software component technology to manufacturer partners for them to use in their products in order to be compatible with Apple's devices.

Thus the proprietary feature of Apple makes it a problem for many people, and an economic issue for others. So our project is about combining Raspberry Pi with Airplay so that any speaker of any cost can be used to connect to Apple product to play audio and video. It is very economic and easy as to say.

1.1 Objective

The main objective is to reduce cost for audio and video mirroring in Apple products as it can only be done through an Apple TV or Apple supported product. This setup makes sure any iOS device can play audio and video to any speaker or monitor using the AirPlay feature given by Apple. This way it is very convenient, consumer efficient and more affordable.

1.2 Organization of the report

The report is divided into 4 parts and each part deals with the different aspects of the system.

(i) **Design:** This part talks about the existing system, how they are designed and the issues associated with them. Furthermore, it describes the features of the system proposed and the requirements for operating it.

(ii) **Module Description:** This part describes each module implemented in the system, i.e., how the data is processed in each and what are the steps involved from the user's point of view.

(iii) **Implementation:** This part deals with an overview of the platform for which the system is developed for. It also talks about the parameters needed for running the system and provides a sample of code used, along with screenshots of the output.

(iv) **Conclusion:** This part concludes the report and discusses the possible enhancement that can be implemented in the future improve the quality.

2. LITERATURE REVIEW

2.1 Existing System

The current system which is among the popular products sold in the market branded by the tech giants **Apple Inc.** is a proprietary protocol This exiting product itself is very popular among people in the market ,and doesn't need any second moment of recognition in particular. But the drawback there is it is very costly, and moreover it is only compatible with apple supported speakers and Apple TV only.

2.2 Proposed System

Our product will however provide an easier and economic way of achieving this system with added benefits and overcomes a few deficiencies by providing a platform with simple and eased codes.

The system which we make would be a speaker that would go in favor of a lot more customers as it is way cheaper than the **Apple inc**. product and coming hand in hand with every aspect the existing system offers. This will be different from the existing system because the product will rebuilt using a **Raspberry Pi** micro controller with **Linux terminal** instructions. It can be used to convert any wired speaker into a wireless speaker and also play it with apple without worrying about the apple compatibility.

2.3 Modules

- **Raspberry Pi:** A Raspberry Pi is a micro controller that is only compatible with Debian based OS. It is real cheap, very light and can be used for many creative projects such as the one done here.
- **USB Wi-Fi adapter:** It is the main source for the AirPlay mirroring between Raspberry Pi and the Apple device. It send wireless signals detected by the Apple product and the products mirror using that signal.
- **iOS device:** For this to work we will need an iOS device that can connect to using the AirPlay feature provided.0
- **Speakers:** Well the basic component required here is a speaker to play the audio to as we are focusing on audio output now and not video.

2.4 Use Case Representation

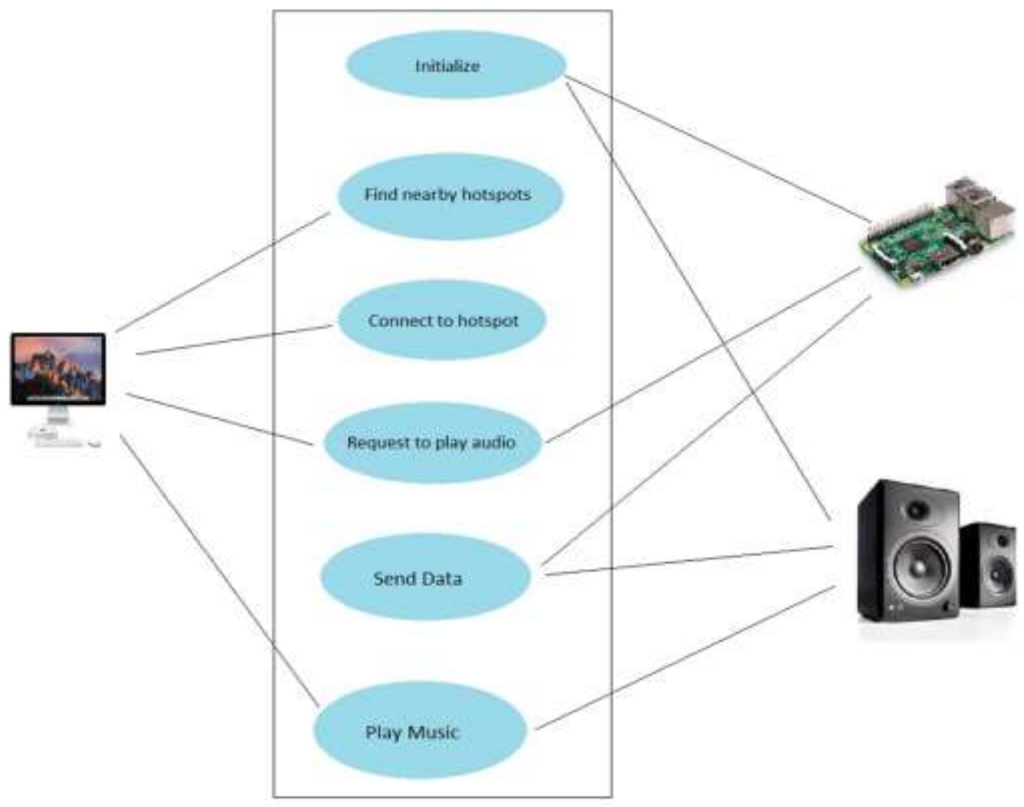
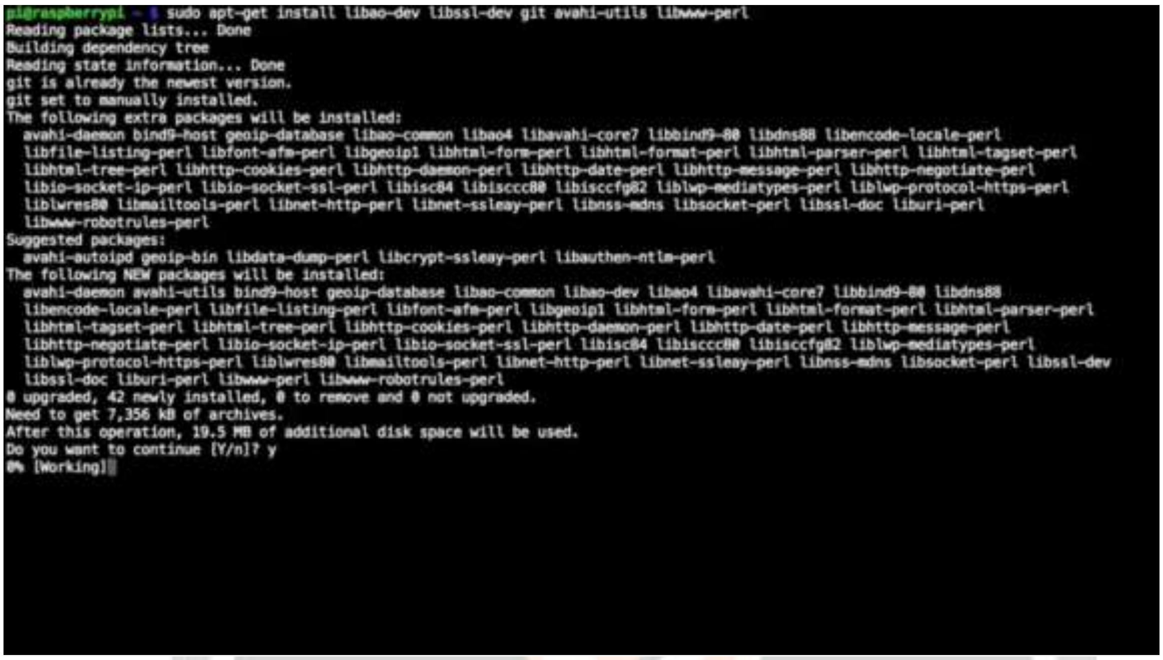


Figure-1: Use Case Diagram

2.5 Sample Code and Screenshots

-The below code is to install some of the **shairports'** dependencies.

>sudo apt-get install git libao-dev libssl-dev libcrypt-openssl-rsa-perl libio-socket-inet6-perl libwww-perl avahi-utils libmodule-build-perl



```
pi@raspberrypi ~ $ sudo apt-get install libao-dev libssl-dev git avahi-utils libwww-perl
Reading package lists... Done
Building dependency tree
Reading state information... Done
git is already the newest version.
git set to manually installed.
The following extra packages will be installed:
  avahi-daemon bind9-host geoip-database libao-common libao4 libavahi-core7 libbind9-80 libdns88 libencode-locale-perl
  libfile-listing-perl libfont-afm-perl libgeoip1 libhtml-form-perl libhtml-format-perl libhtml-parser-perl libhtml-tagset-perl
  libhtml-tree-perl libhttp-cookies-perl libhttp-daemon-perl libhttp-date-perl libhttp-message-perl libhttp-negotiate-perl
  libio-socket-ip-perl libio-socket-ssl-perl libisc84 libisccc80 libisccfg82 liblwp-mediatypes-perl liblwp-protocol-https-perl
  liblwres80 libmailtools-perl libnet-http-perl libnet-ssleay-perl libnss-mdns libsocket-perl libssl-doc liburi-perl
  libwww-robotrules-perl
Suggested packages:
  avahi-autoipd geoip-bin libdata-dump-perl libcrypt-ssleay-perl libauthen-ntlm-perl
The following NEW packages will be installed:
  avahi-daemon avahi-utils bind9-host geoip-database libao-common libao-dev libao4 libavahi-core7 libbind9-80 libdns88
  libencode-locale-perl libfile-listing-perl libfont-afm-perl libgeoip1 libhtml-form-perl libhtml-format-perl libhtml-parser-perl
  libhtml-tagset-perl libhtml-tree-perl libhttp-cookies-perl libhttp-daemon-perl libhttp-date-perl libhttp-message-perl
  libhttp-negotiate-perl libio-socket-ip-perl libio-socket-ssl-perl libisc84 libisccc80 libisccfg82 liblwp-mediatypes-perl
  liblwp-protocol-https-perl liblwres80 libmailtools-perl libnet-http-perl libnet-ssleay-perl libnss-mdns libsocket-perl libssl-dev
  libssl-doc liburi-perl libwww-perl libwww-robotrules-perl
0 upgraded, 42 newly installed, 0 to remove and 0 not upgraded.
Need to get 7,356 kB of archives.
After this operation, 19.5 MB of additional disk space will be used.
Do you want to continue [Y/n]? y
0% [Working]
```

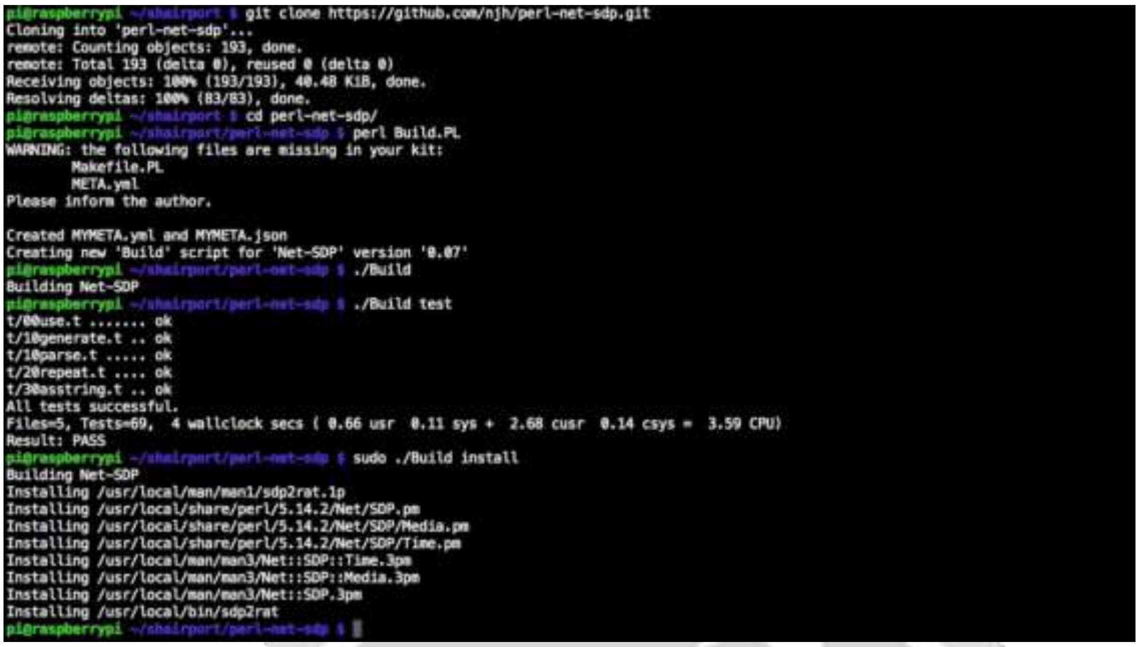
-The below code are to update and install those updates to on your Raspberry Pi.

>sudo apt-get update

Sudo apt-get upgrade

-The below codes are to download and install an update so that the software **shairport** that is supposed to help in connecting to the iOS devices is compatible with the latest version of iOS.

```
>git clone https://github.com/njh/perl-net-sdp.git perl-net-sdp
  cd perl-net-sdp
  perl Build.PL
  sudo ./Build
  sudo ./Build test
  sudo ./Build install
```

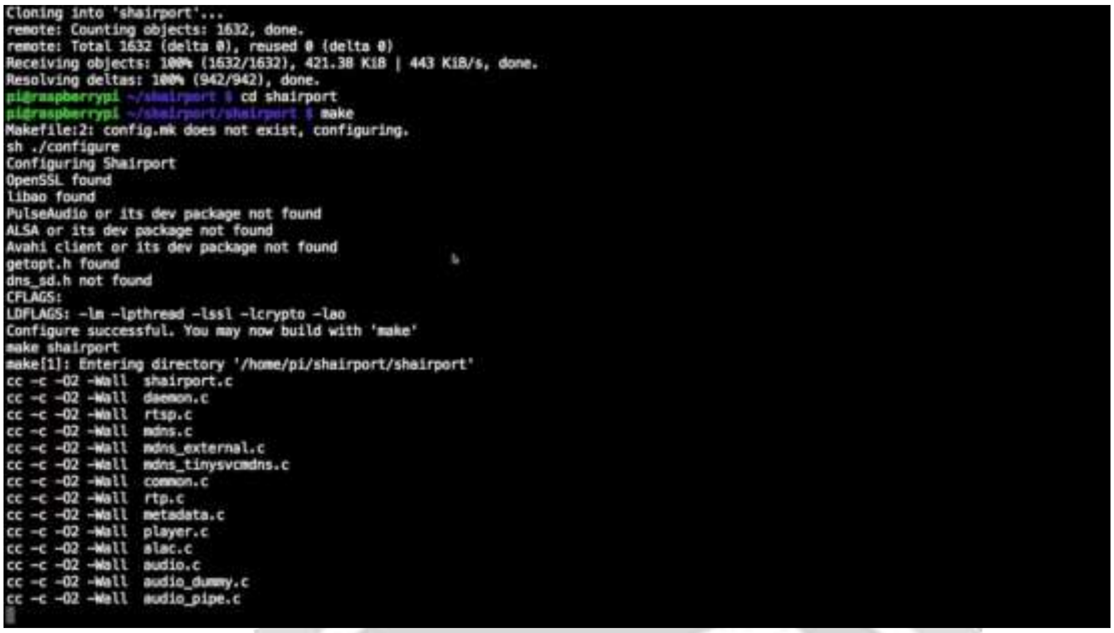


```
pi@raspberrypi ~/shairport $ git clone https://github.com/njh/perl-net-sdp.git
Cloning into 'perl-net-sdp'...
remote: Counting objects: 193, done.
remote: Total 193 (delta 0), reused 0 (delta 0)
Receiving objects: 100% (193/193), 40.48 KiB, done.
Resolving deltas: 100% (83/83), done.
pi@raspberrypi ~/shairport $ cd perl-net-sdp/
pi@raspberrypi ~/shairport/perl-net-sdp $ perl Build.PL
WARNING: the following files are missing in your kit:
        Makefile.PL
        META.yml
Please inform the author.

Created MYMETA.yml and MYMETA.json
Creating new 'Build' script for 'Net-SDP' version '0.07'
pi@raspberrypi ~/shairport/perl-net-sdp $ ./Build
Building Net-SDP
pi@raspberrypi ~/shairport/perl-net-sdp $ ./Build test
t/00use.t ....... ok
t/10generate.t .. ok
t/10parse.t ..... ok
t/20repeat.t .... ok
t/30asstring.t .. ok
All tests successful.
Files=5, Tests=69,  4 wallclock secs ( 0.66 usr  0.11 sys +  2.68 cusr  0.14 csys =  3.59 CPU)
Result: PASS
pi@raspberrypi ~/shairport/perl-net-sdp $ sudo ./Build install
Building Net-SDP
Installing /usr/local/man/man1/sdp2rat.1p
Installing /usr/local/share/perl/5.14.2/Net/SDP.pm
Installing /usr/local/share/perl/5.14.2/Net/SDP/Media.pm
Installing /usr/local/share/perl/5.14.2/Net/SDP/Time.pm
Installing /usr/local/man/man3/Net::SDP::Time.3pm
Installing /usr/local/man/man3/Net::SDP::Media.3pm
Installing /usr/local/man/man3/Net::SDP.3pm
Installing /usr/local/bin/sdp2rat
pi@raspberrypi ~/shairport/perl-net-sdp $
```

-Now these codes are to download and install the **shairport** software itself.

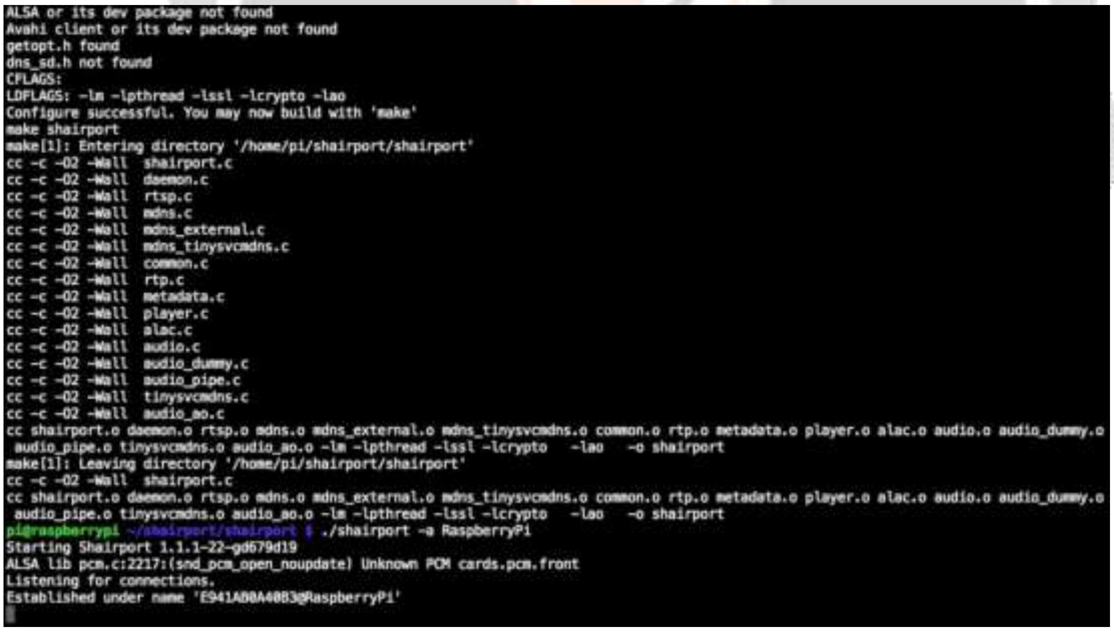
```
>git clone https://github.com/hendrikw82/shairport.git
  cd shairport
  make
  make install
  cp shairport.init.sample /etc/init.d/shairport
  cd /etc/init.d
  chmod a+x shairport
  update-rc.d shairport defaults
```



```
Cloning into 'shairport'...
remote: Counting objects: 1632, done.
remote: Total 1632 (delta 0), reused 0 (delta 0)
Receiving objects: 100% (1632/1632), 421.38 KiB | 443 KiB/s, done.
Resolving deltas: 100% (942/942), done.
pi@raspberrypi ~/shairport $ cd shairport
pi@raspberrypi ~/shairport/shairport $ make
Makefile:2: config.mk does not exist, configuring.
sh ./configure
Configuring Shairport
OpenSSL found
libao found
PulseAudio or its dev package not found
ALSA or its dev package not found
Avahi client or its dev package not found
getopt.h found
dns_sd.h not found
CFLAGS:
LDFLAGS: -lm -lpthread -lssl -lcrypto -lao
Configure successful. You may now build with 'make'
make shairport
make[1]: Entering directory '/home/pi/shairport/shairport'
cc -c -O2 -Wall  shairport.c
cc -c -O2 -Wall  daemon.c
cc -c -O2 -Wall  rtsp.c
cc -c -O2 -Wall  mdns.c
cc -c -O2 -Wall  mdns_external.c
cc -c -O2 -Wall  mdns_tinysvcmdns.c
cc -c -O2 -Wall  common.c
cc -c -O2 -Wall  rtp.c
cc -c -O2 -Wall  metadata.c
cc -c -O2 -Wall  player.c
cc -c -O2 -Wall  alac.c
cc -c -O2 -Wall  audio.c
cc -c -O2 -Wall  audio_dummy.c
cc -c -O2 -Wall  audio_pipe.c
```

-And the last code is to broadcast the signal for the iOS devices to detect and use to play wirelessly to the speaker.

>./shairport -a RaspberryPi



```
ALSA or its dev package not found
Avahi client or its dev package not found
getopt.h found
dns_sd.h not found
CFLAGS:
LDFLAGS: -lm -lpthread -lssl -lcrypto -lao
Configure successful. You may now build with 'make'
make shairport
make[1]: Entering directory '/home/pi/shairport/shairport'
cc -c -O2 -Wall  shairport.c
cc -c -O2 -Wall  daemon.c
cc -c -O2 -Wall  rtsp.c
cc -c -O2 -Wall  mdns.c
cc -c -O2 -Wall  mdns_external.c
cc -c -O2 -Wall  mdns_tinysvcmdns.c
cc -c -O2 -Wall  common.c
cc -c -O2 -Wall  rtp.c
cc -c -O2 -Wall  metadata.c
cc -c -O2 -Wall  player.c
cc -c -O2 -Wall  alac.c
cc -c -O2 -Wall  audio.c
cc -c -O2 -Wall  audio_dummy.c
cc -c -O2 -Wall  audio_pipe.c
cc -c -O2 -Wall  tinysvcmdns.c
cc -c -O2 -Wall  audio_ao.c
cc shairport.o daemon.o rtsp.o mdns.o mdns_external.o mdns_tinysvcmdns.o common.o rtp.o metadata.o player.o alac.o audio.o audio_dummy.o
 audio_pipe.o tinysvcmdns.o audio_ao.o -lm -lpthread -lssl -lcrypto   -lao   -o shairport
make[1]: Leaving directory '/home/pi/shairport/shairport'
cc -c -O2 -Wall  shairport.c
cc shairport.o daemon.o rtsp.o mdns.o mdns_external.o mdns_tinysvcmdns.o common.o rtp.o metadata.o player.o alac.o audio.o audio_dummy.o
 audio_pipe.o tinysvcmdns.o audio_ao.o -lm -lpthread -lssl -lcrypto   -lao   -o shairport
pi@raspberrypi ~/shairport/shairport $ ./shairport -a RaspberryPi
Starting Shairport 1.1.1-22-gd679d19
ALSA lib pcm.c:2217:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.front
Listening for connections.
Established under name 'E941A80A40B3@RaspberryPi'
```

2.7 Future Enhancements

- Switching from Raspbian to Arch Linux to decrease boot time from ~35seconds to 6 seconds.
- Adding an audio output socket and switch.
- Adding bluetooth support for non-Airplay devices.
- To support android devices.

2.8 Features of the setup

The most important feature of this setup are:

- Low cost
- Easy to develop.
- Doesn't require Apple supported devices only.

3. CONCLUSION

The Raspberry Pi is a powerful little beast and a great platform for building low-cost, but highly capable, embedded systems. The interfaces built into its GPIO connector make it easy to bolt on modules using simple low-cost electronics and a bit of configuration to create very functional and flexible systems. The inclusion of a dedicated camera interface and networking interfaces give you everything you could possible need for an Internet-connected home security system.

I've covered a lot of topics in this report, and I hope that what I have presented has been done in a structured and methodical way, and has given you the tools and techniques to carry on this journey so that you are able to create the perfect AirPlay device for your needs.

4. ACKNOWLEDGEMENT

We have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. I would like to extend my sincere thanks to all of them.

I am highly indebted to Mr.Thivakaran, our honorable Head of Department, and Ms.T. Chandralekha, our project guide, for their guidance and constant supervision as well as for providing necessary information regarding the project & also for their support in completing the project.

My thanks and appreciations also go to my colleague in developing the project and people who have willingly helped me out with their abilities.

5.REFERENCES

-www.lifehacker.com
-www.youtube.com
-www.en.wikipedia.org