

课程设计

课程名称： 嵌入式系统课程设计

设计名称： MP3播放器的设计与实现

专业班级： 物联网1502 学号：

学生姓名：

指导教师： 兰方鹏

2018年6月20日

目录

[一、【问题描述】 1](#_Toc518770320)

[二、【设计需求及分析】 1](#_Toc518770321)

[1、背景简介 1](#_Toc518770322)

[2、Madplay主程序流程图 2](#_Toc518770323)

[三、【设计功能的实现】 2](#_Toc518770324)

[1、Madplay的安装和测试 2](#_Toc518770325)

[2、Madplay程序的编写 3](#_Toc518770326)

[3、Madplay移植ARM平台 7](#_Toc518770327)

[4、Linux环境下安装QT 12](#_Toc518770328)

[5、QT实现MP3 播放器 13](#_Toc518770329)

[5.1在QT Creator创建MP3播放器的QT项目 13](#_Toc518770330)

[5.2 MP3播放器的ui界面的设计 15](#_Toc518770331)

[5.3 UI相关控件槽函数的设计 16](#_Toc518770332)

[5.4 UI界面的美化 20](#_Toc518770333)

[5.5成员变量和头文件说明 21](#_Toc518770334)

[四、【运行结果】 21](#_Toc518770335)

[1、Madplay命令行字符程序 21](#_Toc518770336)

[2、Qt MP3播放器的运行 23](#_Toc518770337)

[五、【心得体会】 26](#_Toc518770338)

[六、【附录】 27](#_Toc518770339)

[1、小组分工 27](#_Toc518770340)

[2、程序清单 27](#_Toc518770341)

**MP3播放器的设计与实现**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

专业： 物联网工程 班级： 1502 姓名：刘港 学号：2015003588 完成日期：2018-07-06

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

# 一、【问题描述】

⑴ 制作一个可以在Linux环境下运行的MP3播放器。要求实现的MP3主要功能包括：播放、暂停、继续、停止、循环播放、上一曲、下一曲、退出、歌曲显示等功能。

本设计是基于madplay库实现的，程序运行时会显示一个字符界面，在该字符界面下输入相应的提示字符即可实现音乐的播放、暂停、继续、停止、上一曲、下一曲、歌词的显示等功能，也可以加入额外的一些功能。编写一个MakeFile文件，make命令来实现程序的编译。

⑵ 对madplay源码进行移植，安装交叉编译工具链，搭建交叉编译环境，使得该程序可以在ARM平台上运行。

⑶ 在Linux平台（Redhat或Ubuntu）环境下配置QT环境，设计一个基于图形化界面的MP3播放器。

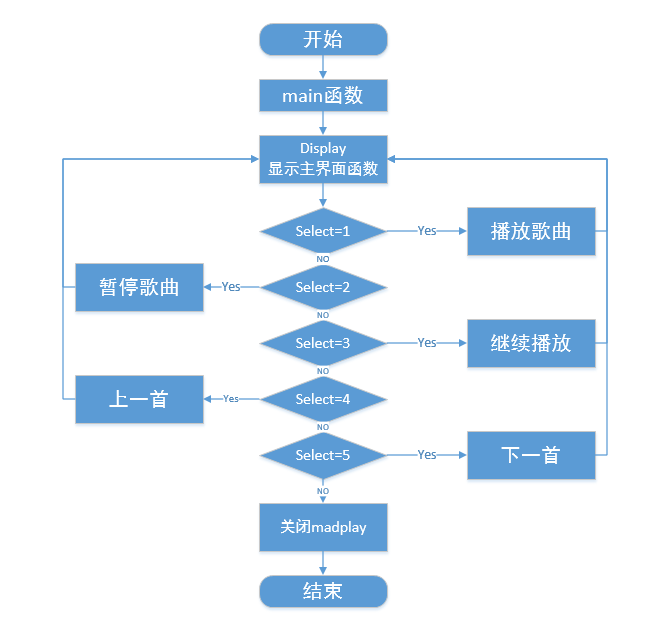
# 二、【设计需求及分析】

## 1、背景简介

随着嵌入式行业的日新月异,嵌入式系统的应用越来越受人喜爱,自由可移植的优点促使人们更倾向于关注嵌入式技术的研究,媒体类的产品被人们广泛而快速的使用,尤其是音乐播放器的软件开发成为了众多企业争宠的对象。

Madpaly 是 linux 上的一个开源 MP3 播放器,是一个高精度 MPEG 音频解码器。支持MPEG-1 Layer,Layer II 和 Layer III(也就是 MP3)标准,利用 libmad 库进行 MP3 解码。本课程设计利用 Madplay 库实现一个简单的音乐播放器,可以播放本地的音频文件并具备基本的音乐操作处理功能,播放暂停恢复停止音量操作等。

## 2、Madplay主程序流程图



二 图2 1

# 三、【设计功能的实现】

注：系统采用ubuntu16.04

## 1、Madplay的安装和测试

**1.1、madplay的安装**

在ubuntu环境下，直接使用在线安装方式

sudo apt-get install madplay

**1.2、madplay的配置**

在链接 madplay 共享库时只是指定了动态链接器和该程序所需要的库文件,并没有真的做链接,可执行文件 madplay 中调用的 madplay 库函数仍然是未定义符号,要在运行时做动态链接。

可设置环境变量,动态连接器就从这个指定的路径下搜索共享库

export LD\_LIBRARY\_PATH=xxx/mp3lib:$ LD\_LIBRARY\_PATH

修改/etc/profile，将madplay加入系统环境变量 vim /etc/profile

加入 export PATH=/usr/local/madplay/bin:$PATH后保存。

1.3、madplay的测试

拷贝一个 mp3 文件 1.mp3，执行命令madply 1.mp3

若可以播放表示 MadPlay 安装并配置成功。

## 2、Madplay程序的编写

**2.1、程序编写框架**

1）将歌曲名称目录进行查询并将结果重定向到歌单文件中。

2）读取歌单文件并将其存入数组

3）循环进行功能的选择，功能选择包括（播放、暂停、继续、上一首、下一首、退出）。

4）退出循环结束程序。

**2.2、程序各模块功能的实现**

2.2.1、将歌曲名称目录进行查询并将结果重定向到歌单文件中

(1)创建一个目录用于存放mp3歌曲，并将歌曲存入此目录

mkdir music

(2) 将歌曲名称目录重定向到歌单文件中

在cpp程序中调用system(“ls music > music\_list”);

system("ls /home/liugang/Desktop/music/ > /home/liugang/Desktop/music\_list");

2.2.2、读取歌单文件并将其存入数组

(1)、定义一个string类型的数组用于存放歌单，

string music\_list[N];

(2)、打开生成的歌单文件music\_list，按行读取文件的数据。定义一个缓冲string存放该行的歌曲名称并将其加入歌单数组music\_list[]，定义变量music\_number表示歌曲的数目。

int music\_number=0;

fstream fin("/home/liugang/Desktop/music\_list");

while(getline(fin,buffer))

{

music\_list[music\_number]=buffer;

music\_number++;

}

2.2.3、循环进行功能的选择，功能选择包括（播放、暂停、继续、上一首、下一首、退出）。

(1)、动态生成命令功能

此函数存在三个参数，第一个参数是歌曲清单数组，第二个参数是歌曲数目，第三个参数是当前歌曲的序号（0开始）。

函数返回为生成的播放命令字符串const char \*command。

const char\* produce\_command(string music\_list[],int music\_number,int current\_number=0)

{

string com="madplay";

for(int i=current\_number;i<music\_number;i++)

{

com=com+" "+"/home/liugang/Desktop/music/"+music\_list[i];

}

com=com+" < /dev/null > /dev/null 2>1& ";

const char \*command=com.c\_str();

return command;

}

(2)、播放功能

1）判断标志位是否为true，如果不是，则说明当前有歌播放，执行

if(start==false) system("killall -KILL madplay");

2）生成相应的播放命令执行，输出歌曲名称，并将标志位置为false。

system(command);

start=false;

cout<<music\_list[current\_number]<<" is playing---"<<endl;

(3)、暂停功能

1）生成暂停命令。

system("killall -STOP madplay");

2）输出歌曲名称并显示暂停。

cout<<music\_list[current\_number]<<" is puase---"<<endl;

(4)、继续功能

1）生成继续命令。

system("killall -CONT madplay");

2）输出歌曲名称并显示继续状态。

cout<<music\_list[current\_number]<<" is replaying---"<<endl;

(5)、下一首功能

1）关闭正在执行的歌曲。

system("killall -KILL madplay");

2）根据当前执行的歌曲序号，进行加一操作。如果序号大于等于歌曲数目，则将序号置为0，通过数组的下标来实现歌曲的循环播放。

current\_number++;

if(current\_number>=music\_number) current\_number=0;

3）生成相应的播放命令，执行播放。

const char \*nextcommand=produce\_command(music\_list,music\_number,current\_number);

system(nextcommand);

4）输出歌曲名称并显示播放状态。

cout<<music\_list[current\_number]<<" is playing---"<<endl;

(6)、上一首功能

1）关闭正在执行的歌曲。

system("killall -KILL madplay");

2）根据当前执行的歌曲序号，进行减一操作。如果序号小于0，则将序号置为歌曲数目-1，通过数组的下标来实现歌曲的循环播放。

current\_number--;

if(current\_number<0) current\_number=music\_number-1;3）生成相应3）生成相应的播放命令，执行播放。

const char \*precommand=produce\_command(music\_list,music\_number,current\_number);

system(precommand);

4）输出歌曲名称并显示播放状态。

cout<<music\_list[current\_number]<<" is playing---"<<endl;

(7)、退出功能

关闭歌曲并显示退出

system("killall -KILL madplay");

cout<<"--------play end---------"<<endl;

(8)、循环显示字符界面

void display()

{

cout<<" 1.Play"<<endl;

cout<<" 2.Pause"<<endl;

cout<<" 3.Continue"<<endl;

cout<<" 4.MoveNext"<<endl;

cout<<" 5.MovePrivious"<<endl;

cout<<" 6.Exit"<<endl;

cout<<"Please input the choose:";

}

**2.3、程序主要函数说明**

播放歌曲 system(" madplay 1.mp3 < /dev/null > /dev/null 2>1&");

"< /dev/null > /dev/null 2>1&" 表示只播放歌曲，不进行输入输出，即歌曲在后台进行播放。

暂停歌曲 system("killall -STOP madplay");

继续歌曲 system("killall -CONT madplay");

结束歌曲 system("killall -KILL madplay");

循环操作采用数组的下标进行实现。

**2.4、编写Makefile文件进行编译并运行**

1）编写Makefile文件

play:play.cpp

g++ play.cpp -o play

clean:

rm -rf play

2）运行Makefile文件 # make

## 3、Madplay移植ARM平台

Madplay移植ARM平台后,可以在ARM板上利用Madplay播放mp3音乐。

**3.1 建立 madplay 源码目录 madplay-src 存放 madplay 源码**

liugang@ubuntu:~/Desktop$ mkdir madplay\_src

**3.2 解压 madplay 相关源码到所建立的源码目录 madplay-src**

(1) 解压 zlib 库到源码目录 madplay-src

liugang@ubuntu:~/Desktop/课设软件/madplay移植软件$ tar -zxvf zlib-1.2.3.tar.gz -C /home/liugang/Desktop/madplay\_src/

(2) 解压 libid3tag 库到源码目录 madplay-src

liugang@ubuntu:~/Desktop/课设软件/madplay移植软件$ tar -zxvf libid3tag-0.15.1b.tar.gz -C /home/liugang/Desktop/madplay\_src/

(3) 解压 libmad 库到源码目录 madplay-src

liugang@ubuntu:~/Desktop/课设软件/madplay移植软件$ tar -zxvf libmad-0.15.1b.tar.gz -C /home/liugang/Desktop/madplay\_src/

(4) 解压 madplay 库到源码目录 madplay-src

liugang@ubuntu:~/Desktop/课设软件/madplay移植软件$ tar -zxvf madplay-0.15.2b.tar.gz -C /home/liugang/Desktop/madplay\_src/

**3.3 编译 zlib 库,为 libid2tag 提供库函数支持**

3.3.1 进入 zlib 目录:

liugang@ubuntu:~/Desktop/madplay\_src$ cd zlib-1.2.3/

3.3.2 执行 configure 脚本进行配置

liugang@ubuntu:~/Desktop/madplay\_src/zlib-1.2.3$ ./configure -shared --prefix=/home/liugang/Desktop/madplay\_src/zlib-1.2.3/

3.3.3 修改 Makefile 文件

修改 Makefile 第 19 行

19 CC=arm-linux-gcc

修改 Makefile 第 28 行

28 LDSHARED=arm-linux-gcc -shared -Wl,-soname,libz.so.1

修改 Makefile 第 29 行

29 CPP=arm-linux-gcc -E

修改 Makefile 第 36 行

36 AR=arm-linux-ar rc

修改 Makefile 第 37 行

37 RANLIB=arm-linux-ranlib

3.3.4 安装交叉编译工具链,并配置环境变量

安装arm-linux-gcc

liugang@ubuntu:~/Desktop/课设软件$ tar -zxvf arm-linux-gcc-4.3.2.tgz

liugang@ubuntu:~/Desktop/课设软件$ cd usr/local/

liugang@ubuntu:~/Desktop/课设软件/usr/local$ sudo mv arm /usr/local/

配置环境变量

liugang@ubuntu:/usr/local$ export PATH=/usr/local/arm/4.3.2/bin/:$PATH

3.3.5 编译并安装 zlib 库

liugang@ubuntu:~/Desktop/madplay\_src/zlib-1.2.3$ make

liugang@ubuntu:~/Desktop/madplay\_src/zlib-1.2.3$ make install

**3.4 编译 libid3tag 库**

3.4.1 进入 libid3tag-0.15.1b 目录

liugang@ubuntu:~/Desktop/madplay\_src/zlib-1.2.3$ cd ../ libid3tag-0.15.1b

3.4.2 配置 configure 脚本

liugang@ubuntu:~/Desktop/madplay\_src/libid3tag-0.15.1b$ ./configure --prefix=/home/liugang/Desktop/madplay\_src/libid3tag-0.15.1b CC=arm-linux-gcc --host=arm-linux CPPFLAGS=-I/home/liugang/Desktop/madplay\_src/zlib-1.2.3/include/LDFLAGS=-I/home/liugang/Desktop/madplay\_src/zlib-1.2.3/lib/

3.4.3 编译并安装 libid3tag 库

liugang@ubuntu:~/Desktop/madplay\_src/libid3tag-0.15.1b$ make

liugang@ubuntu:~/Desktop/madplay\_src/libid3tag-0.15.1b$ make install

**3.5 编译音频解码库 libmad,该解码库的编译依赖于 libid3tag**

3.5.1 进入 libmad-0.15.1b 目录

liugang@ubuntu:~/Desktop/madplay\_src/libid3tag-0.15.1b$ cd ../ libmad-0.15.1b

3.5.2 执行脚本配置

liugang@ubuntu:~/Desktop/madplay\_src/libmad-0.15.1b$ ./configure --prefix=/home/liugang/Desktop/madplay\_src/libmad-0.15.1b/ CC=arm-linux-gcc --host=arm-linux CPPFLAGS=-I/home/liugang/Desktop/madplay\_src/libid3tag-0.15.1b/include/LDFLAGS=-I/home/liugang/Desktop/madplay\_src/libid3tag-0.15.1b/lib/

3.5.3 编译并安装 libmad 库

liugang@ubuntu:~/Desktop/madplay\_src/libmad-0.15.1b$ make

cc1: error: unrecognized command line option "-fforce-mem"

Makefile:383: recipe for target 'version.lo' failed

make[2]: \*\*\* [version.lo] Error 1

make[2]: Leaving directory '/home/liugang/Desktop/madplay\_src/libmad-0.15.1b'

Makefile:424: recipe for target 'all-recursive' failed

make[1]: \*\*\* [all-recursive] Error 1

make[1]: Leaving directory '/home/liugang/Desktop/madplay\_src/libmad-0.15.1b'

Makefile:249: recipe for target 'all' failed

make: \*\*\* [all] Error 2

//make 错误时删除 MakeFile 当中的-fforce-mem

CFLAGS = -Wall -g -O -fforce-mem -fforce-addr -fthread-jumps -fcse-follow-jumps -fcse-skip-blocks -fexpensive-optimizations -fregmove -fschedule-insns2 -fstrength-reduce

删除 -fforce-mem

liugang@ubuntu:~/Desktop/madplay\_src/libmad-0.15.1b$ make install

**3.6 编译 madplay 应用程序,该应用程序依赖于zlib,libid3tag和libmad 库**

3.6.1 进入 madplay-0.15.2b 目录

[liugang@ubuntu:~/Desktop/madplay\_src/libid3tag-0.15.1b$ cd ../ madplay-0.15.2b

3.6.2 执行 configure 脚本配置

liugang@ubuntu:~/Desktop/madplay\_src/madplay-0.15.2b$ ./configure --prefix=/home/liugang/Desktop/madplay\_src/madplay-0.15.2b CC=arm-linux-gcc --host=arm-linux CPPFLAGS=-I/home/liugang/Desktop/madplay\_src/libid3tag-0.15.1b/include/-I/home/liugang/Desktop/madplay\_src/libmad-0.15.1b/include/ LDFLAGS=-L/home/liugang/Desktop/madplay\_src/zlib-1.2.3/lib/-L/home/liugang/Desktop/madplay\_src/libid3tag-0.15.1b/lib/-L/home/liugang/Desktop/madplay\_src/libmad-0.15.1b/lib

3.6.3 编译并安装madplay

liugang@ubuntu:~/Desktop/madplay\_src/madplay-0.15.2b$ make

liugang@ubuntu:~/Desktop/madplay\_src/madplay-0.15.2b$ make install

3.6.4 查看 madplay所依赖的库文件

liugang@ubuntu:~/Desktop/madplay\_src/madplay-0.15.2b$ arm-linux-readelf -d madplay|grep Shared

结果如下所示:

0x00000001 (NEEDED) Shared library: [libmad.so.0]

0x00000001 (NEEDED) Shared library: [libid3tag.so.0]

0x00000001 (NEEDED) Shared library: [libm.so.6]

0x00000001 (NEEDED) Shared library: [libc.so.6]

**3.7 移植 madplay所需库文件到开发板**

madplay 要正常运行需要 libmad.so.0,libid3tag.so.0,libm.so.6 和 libc.so.6,其中 libm.so 和libc.so.6 在开发板的/lib 目录中存在,因此只需要将 lib-mad.so.3 和 libid3tag.so.0 移植到开发板即可,但libmad.so.0 和 libid3tag.so.0 是两个软链接文件分别链接 libmad.so.0.2.1 和libid3tag.so.0.3.0,可用下面命令查看:

liugang@ubuntu:~/Desktop/madplay\_src/madplay-0.15.2b$ ls -l /home/liugang/Desktop/madplay\_src/libmad-0.15.1b/lib/libmad.so.0

显示结果如下:

lrwxrwxrwx 1 liugang liugang 15 7月 4 02:38 /home/liugang/Desktop/madplay\_src/libmad-0.15.1b/lib/libmad.so.0 -> libmad.so.0.2.1

查看 libid3tag.so.0

liugang@ubuntu:~/Desktop/madplay\_src/madplay-0.15.2b$ ls -l /home/liugang/Desktop/madplay\_src/libid3tag-0.15.1b/lib/libid3tag.so.0

显示结果如下:

lrwxrwxrwx 1 liugang liugang 18 7月 4 01:09 /home/liugang/Desktop/madplay\_src/libid3tag-0.15.1b/lib/libid3tag.so.0 -> libid3tag.so.0.3.0

因此,把 libmad.so.0.2.1 和 libid3tag.so.0.3 移植到开发板/lib 目录下并创建两个软链接即可。

(1) 将 PC 机 /home/liugang/Desktop/madplay\_src/libmad/lib 目 录 下 的 libmad.so.0.2.1 和

/home/liugang/Desktop/madplay\_src/libid3tag/lib 目录下的libid3tag.so.0.3.0 两个库文件拷贝到开发板的/lib 目录下

(2) 在开发板终端上创建软链接文件,命令如下:

ln -s libid3tag.so.0.3.0 libid3tag.so.0

ln -s libmad.so.0.2.1 libmad.so.0

(3) 在开发板上查看是否移植成功

ls -l libmad.so.0 libid3tag.so.0

(4) 移植 madplay 应用程序到开发板

把 /home/liugang/Desktop/madplay\_src/madplay-0.15.2b/bin/目录下的可执行文件 madplay 复制到开发板的/usr/bin 目录,传一首歌曲 1.mp3,进行测试:

使用命令“madplay 1.mp3”能播放说明移植成功。

**3.8 编写一个 Makefile 文件,用 make 命令将之前设计的 mp3 播放器程序编译成可以在ARM 板上运行的程序,然后将该程序传到 ARM 板上,测试程序。**

makefile 文件目录：/home/liugang/Desktop/musiccode/Makefile

Makefile文件如下：

arm-play:play.cpp

arm-linux-g++ play.cpp -o arm-play

clean:

rm -rf arm-\*

## 4、Linux环境下安装QT

Qt 是一个跨平台的 C++应用程序框架,支持 Windows、Linux、Mac OS X、Android、iOS、Windows Phone、嵌入式系统等。即 Qt 可以同时支持桌面应用程序开发、嵌入式开发和移动开发,覆盖了现有的所有主流平台。

这次课程设计我们在 QT 平台上开发一个 MP3 播放器,首先需要在 Linux 平台上安装并配置 QT 开发环境,配置安装如下。

**4.1 下载 QT 源码软件,并解压**

http://download.qt.io/archive/qt/4.7/qt-everywhere-opensource-src-4.7.3.tar.gz.mirrorlist

我们统一下载的版本连接为:

http://mirrors.tuna.tsinghua.edu.cn/qt/archive/qt/4.7/qt-everywhere-opensource-src-4.7.3.tar.gz

**4.2 执行 QT 配置**

进入 qt-everywhere-opensource-src-4.7.3 目录,执行配置文件 configure

Ubuntu 环境下需要超级账号执行,即:sudo ./configure

当出现系统出现以下信息时:

Which edition of Qt do you want to use ?

Type 'c' if you want to use the Commercial Edition.

Type 'o' if you want to use the Open Source Edition.

键入:o

Type '3' to view the GNU General Public License version 3.

Type 'L' to view the Lesser GNU General Public License version 2.1.

Type 'yes' to accept this license offer.

Type 'no' to decline this license offer.

Do you accept the terms of either license?

键入:yes

**4.3 编译源码**

执行命令:make -j4。加上-j4 选项表示开了 4 个线程进行编译,时间会大大缩短。

**4.4 安装程序**

Ubuntu 环境下命令为:sudo make install

安装成功后系统会生成这个文件: /usr/local/Trolltech/Qt-4.7.3/bin/qmake

**4.5 配置 QT 的环境变量**

ubuntu 环境下:

vim /etc/profile ，加入：

export QTDIR=/usr/local/Trolltech/Qt-4.7.3

export PATH=$QTDIR/bin:$PATH

export LD\_LIBRARY\_PATH=$QTDIR/lib:/usr/local/lib:$LD\_LIBRARY\_PATH

保存之后 source /etc/profile立即生效.

**4.6 测试**

输入命令:qmake –v

若输出以下信息表示 QT 安装并配置成功。

liugang@ubuntu:~$ qmake -v

QMake version 2.01a

Using Qt version 4.7.3 in /usr/local/Trolltech/Qt-4.7.3/lib

**4.7 安装 QCreator**

至此,qt-x86 环境已搭建好,但是这样编写代码开发 GUI 应用程序不方面,我们通过Qt Creator 可以进行快速开发。Qt Creator 是一个用于 Qt 开发的轻量级跨平台集成开发环境。

(1) 下载 QCreator

下载地址: http://download.qt.io/official\_releases/qtcreator/2.5/2.5.2/

下载版本:qt-creator-linux-x86-opensource-2.5.2.bin

(2) 安装:

执行命令:./ qt-creator-linux-x86-opensource-2.5.2.bin

根据系统提示选择即可,默认 QCreator 的安装路径是:/opt/qtcreator-2.5.2

安装成功后,Linux 系统桌面会出现 QCreator 图标。

## 5、QT实现MP3 播放器

### 5.1在QT Creator创建MP3播放器的QT项目

(1)打开Qt Creator，点击文件-新建文件或工程，选择Qt Gui应用

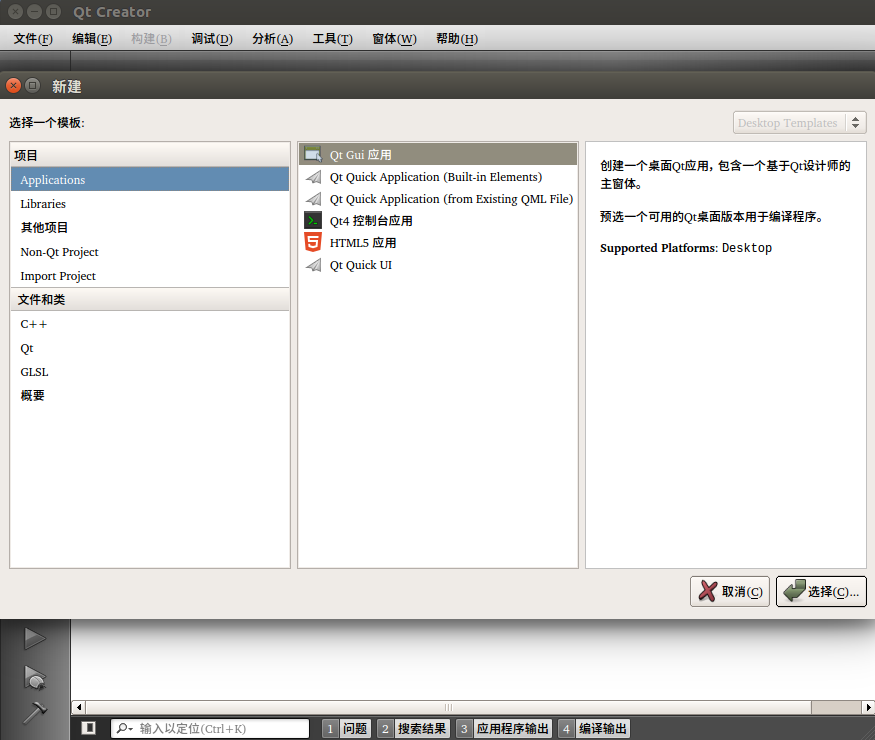


图5.1 1

(2)输入项目名称和项目创建路径

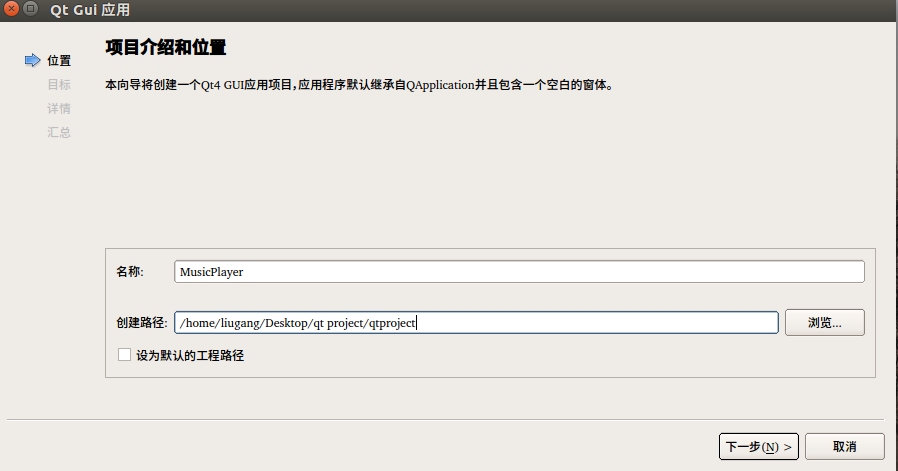


图5.1 2

(3)设置qt项目的基类为QWidget，并勾选创建界面

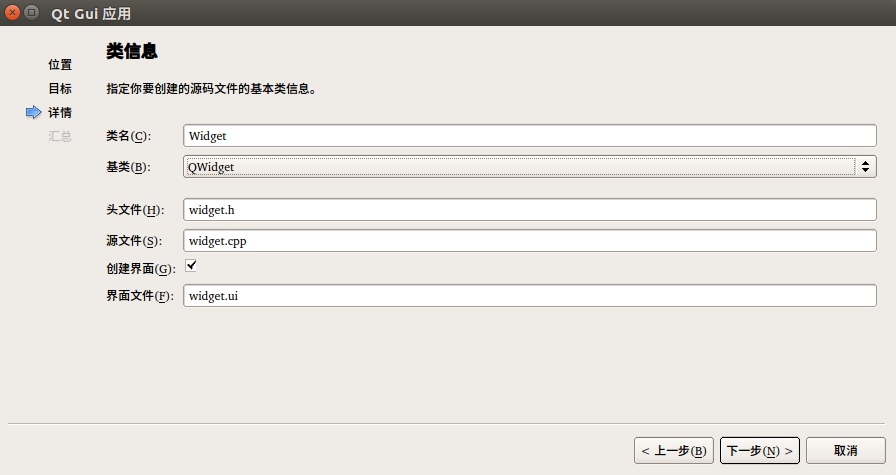


图5.1 3

(4)完成qt项目的创建

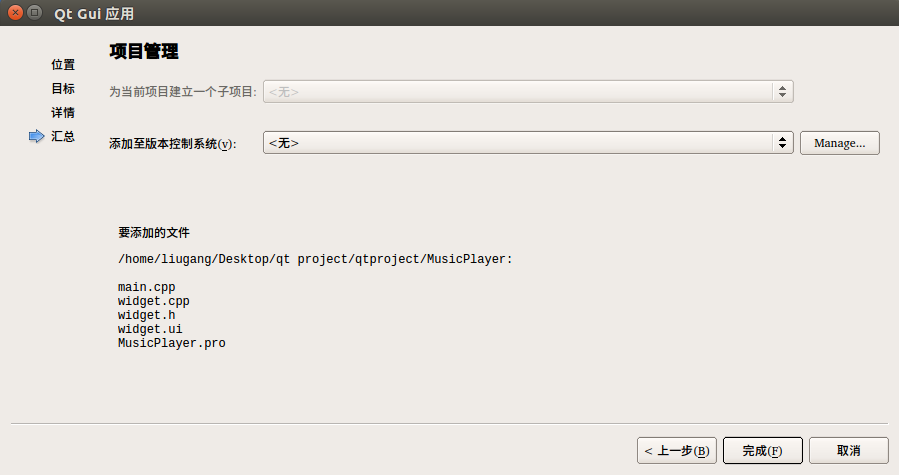


图5.1 4

### 5.2 MP3播放器的ui界面的设计

点击widget.ui进入ui界面，在界面的画板中添加若干label、push button、list widget、horizontal spacer（滑动条）等控件。设计的ui界面如下图：

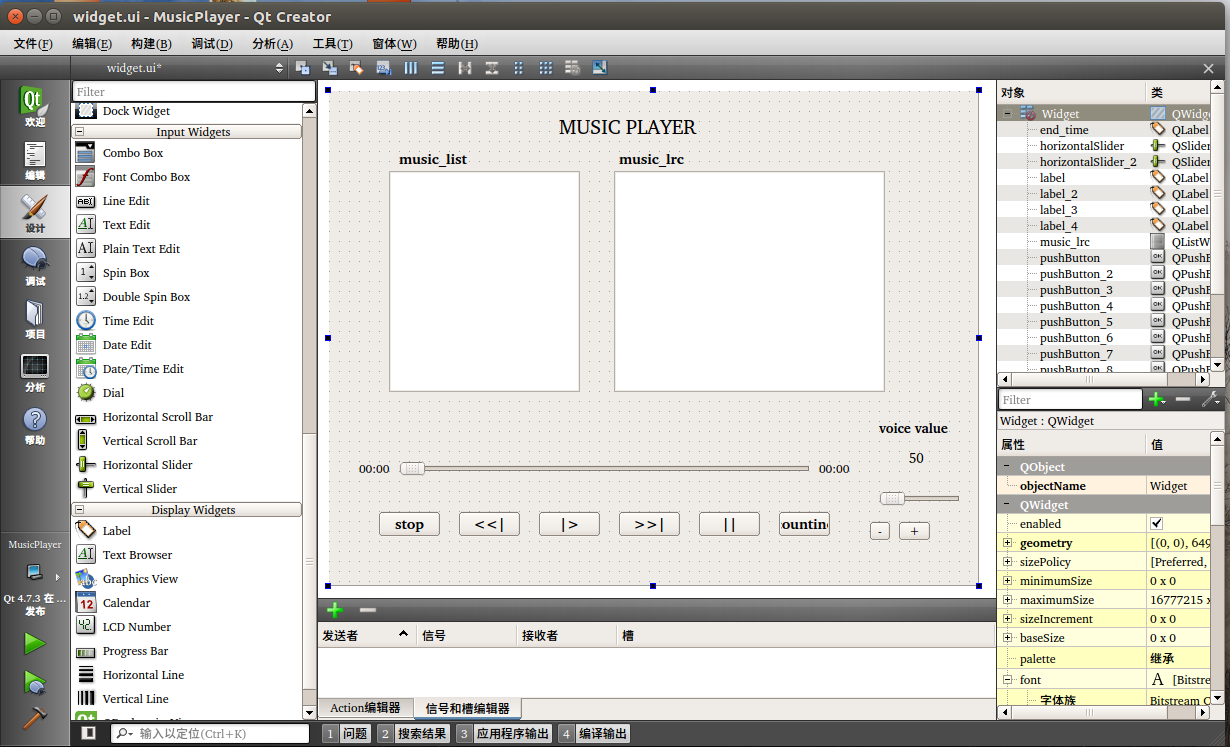


图5.2 1

### 5.3 UI相关控件槽函数的设计

5.3.1、槽函数的声明（widget.h）

private slots:

//---------单击播放按钮---------

void on\_pushButton\_3\_clicked();

//---------单击下一首按钮--------

void on\_pushButton\_4\_clicked();

//---------单击上一首按钮--------

void on\_pushButton\_2\_clicked();

//---------单击继续按钮---------

void on\_pushButton\_clicked();

//---------单击暂停按钮---------

void on\_pushButton\_5\_clicked();

//---------单击停止按钮---------

void on\_pushButton\_8\_clicked();

//---------双击歌曲--------------------

void on\_ui\_music\_list\_doubleClicked();

//---------声音进度条值改变------------------------

void on\_horizontalSlider\_2\_valueChanged(int value);

//---------单击声音-按钮---------

void on\_pushButton\_7\_clicked();

//---------单击声音+按钮---------

void on\_pushButton\_6\_clicked();

//---------定时器操作---

void start\_counting();

//---------歌曲进度条拖到释放---------------

void on\_horizontalSlider\_sliderReleased();

5.3.2、关键槽函数的实现

**5.3.2.1、实现歌单的展示**

首先调用系统命令生成歌曲清单存入文件中，之后打开文件进行一行一行地读取，将读取到的歌名添加到list widget中实现歌词的展示。

//----------------------------加载歌曲清单-----------------------

system("ls /home/liugang/Desktop/music/ > /home/liugang/Desktop/music\_list");

**fstream fin("/home/liugang/Desktop/music\_list");**

std::string buffer;

int music\_number=0;

while(getline(fin,buffer))

{

const char\* music=buffer.c\_str();

**ui->ui\_music\_list->addItem(tr(music));**

music\_number++;

}

**5.3.2.2、歌曲进度的时间、歌词动态显示和进度条的动态改变**

采用定时器来实现动态的操作，设置定时一秒，当超过一秒后，系统将产生一个timeout()信号，通过connect关联来执行相应的槽函数更改时间和进度条，从而实现进度条、歌词和时间的动态显示。

//----------------------------定时器连接函数--------------------- **QObject::connect(timer,SIGNAL(timeout()),this,SLOT(start\_counting()))**;

//-----------------------------定时器槽函数--------------------------

void Widget::start\_counting()

{

playtime++;

//--------------时间动态-----------

//caltime()函数将播放时间(int)转换成Qstring字符串

QString qplaytime=caltime(playtime);

**ui->start\_time->setText(qplaytime);**

//--------------进度条和歌词-----------------

**ui->horizontalSlider->setValue(playtime);**

int music\_row=ui->ui\_music\_list->currentRow();

if(playtime%3==0&&playtime>=15&&music\_row==0){

**ui->music\_lrc->setCurrentRow((playtime-16)/3);**

//此次歌词只做了一个简单的定位实现，因为时间和lrc文件的问题，此代码只作为测试使用，并不能实现歌词动态定位功能。

}

}

**5.3.2.3、单击播放按钮功能函数**

首先判断是否有歌曲播放，如果有，将播放的歌曲KILL掉。之后取出播放的歌曲在list widget中所在行的歌名来载入歌词，将歌词载入music\_lrc中。将歌名使用label进行显示，并使用madplay播放歌曲，开启定时器。

//------------------------------单击play按钮------------------------

void Widget::on\_pushButton\_3\_clicked()

{

if(!firstflag) system("killall -KILL madplay ");

int curMusicRow=ui->ui\_music\_list->currentRow();

**QString currentMusic=ui->ui\_music\_list->currentItem()->text();**

//---------读取载入歌词------------

fstream finj("/home/liugang/Desktop/lrc/JoelHanson-TravelingLight");

std::string buffer;

while(getline(finj,buffer))

{

const char\* musiclrc=buffer.c\_str();

**ui->music\_lrc->addItem(tr(musiclrc));**

}

//------------lable显示歌名--------------

**ui->show1->setText(currentMusic)**;

string com="madplay /home/liugang/Desktop/music/"+currentMusic.toStdString()+" < /dev/null > /dev/null 2>1&";

const char\* command=com.c\_str();

system(command);

firstflag=false;

//-------------开启定时器，定时1秒------------

**timer->start(1000);**

**5.3.2.3、音量调节进度条函数**

如果进度条的值改变，将会触发这个槽函数。首先将进度条的值通过label进行显示，通过调用系统命令amixer来改变音量。

//-------------------------------音量活动slider---------------------

void Widget::on\_horizontalSlider\_2\_valueChanged(int value)

{

QString voice\_value=QString("%1").arg(value);//int 转Qstring

**ui->voice->setText(voice\_value);** //设置label声音的值

string svoice\_value=intToString(value);

**string linux\_voice\_com="amixer -q sset PCM "+svoice\_value+"%";**

const char\* linux\_voice\_command=linux\_voice\_com.c\_str();

system(linux\_voice\_command);

}

**5.3.2.3、通过歌曲进度条来调节歌曲播放进度**

首先得到进度条的值，将这个int值转换为时间的表示方式。通过madplay的 -s 参数加入时间来控制歌曲播放进度。

//------------------------------歌曲进度条---------------------------

void Widget::on\_horizontalSlider\_sliderReleased()

{

system("killall -KILL madplay ");

**QString currentMusic=ui->ui\_music\_list->currentItem()->text();**

**int slidertime=ui->horizontalSlider->value();**

playtime=slidertime;

QString qplaytime=caltime(playtime);

**string com="madplay /home/liugang/Desktop/music/"+currentMusic.toStdString()+" -s "+qplaytime.toStdString()+" < /dev/null > /dev/null 2>1&";**

const char\* command=com.c\_str();

system(command);

timer->start();

}

其他槽函数的实现请参考所提交的qt代码文件。

### 5.4 UI界面的美化

5.4.1、编写qss代码并加入相应的资源文件

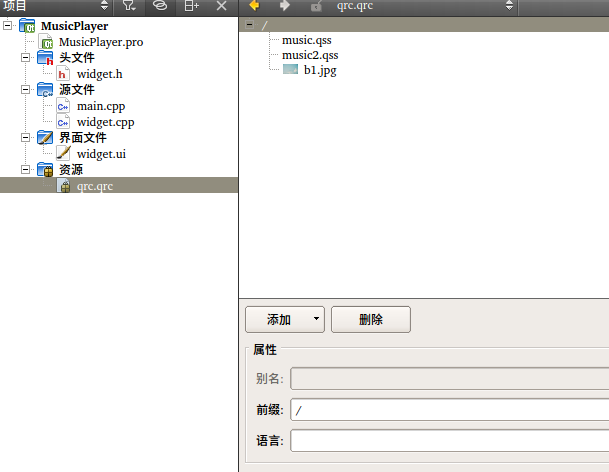


图5.4 1

5.4.2、在main函数载入qss文件

//------------加载qss-------------

QFile qss(":/music2.qss");

qss.open(QFile::ReadOnly);

qApp->setStyleSheet(qss.readAll());

qss.close();

### 5.5成员变量和头文件说明

5.5.1、成员变量

public:

explicit Widget(QWidget \*parent = 0);

int playtime; //播放时间

int playrow; //当前播放歌曲所处歌单的行数

bool firstflag; //播放标志位

QTimer \*timer=new QTimer(); //定时器

5.5.2、头文件信息

#include <QWidget>

#include <QTimer>

#include <QApplication>

#include <QFile>

#include <sys/types.h>

#include <fstream>

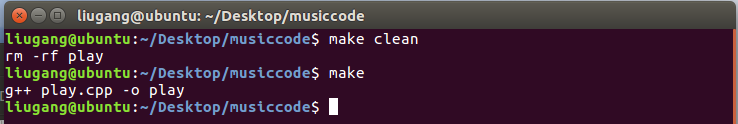
#include <QString>

#include "sstream"

# 四、【运行结果】

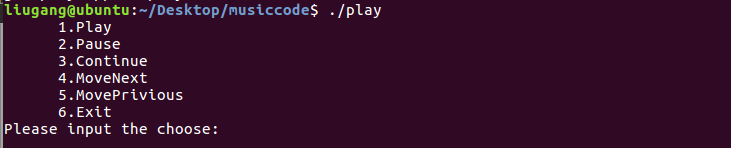
## 1、Madplay命令行字符程序

1.1、命令行make执行madplay的Makefile



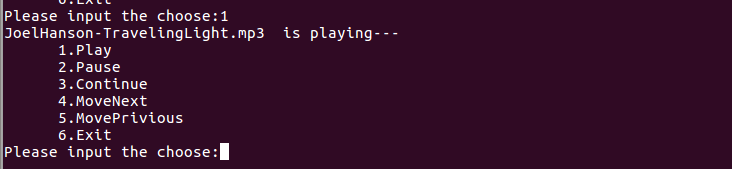
四 图1.1 1

1.2、执行play



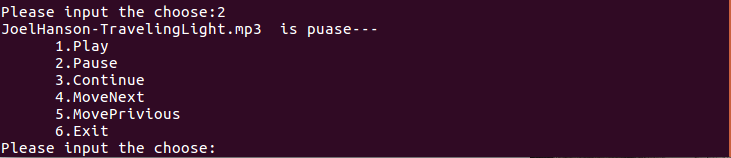
四 图1.2 1

选择1进行播放：



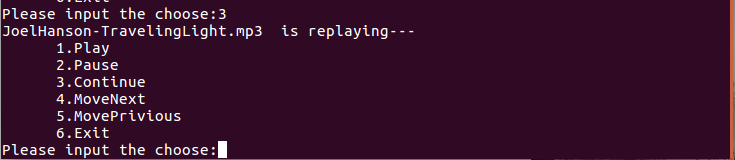
四 图1.2 2

选择2进行暂停：



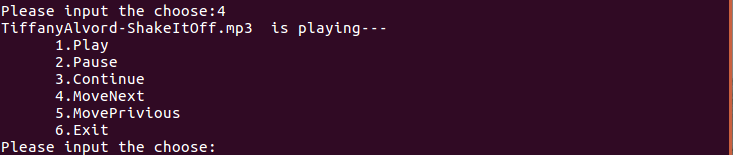
四 图1.2 3

选择3继续播放：



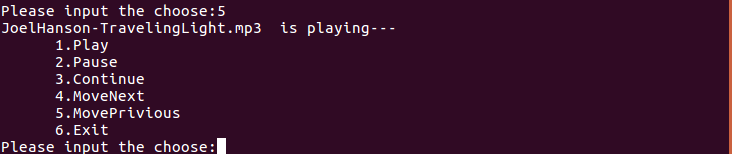
四 图1.2 4

选择4下一首播放：



四 图1.2 5

选择5上一首播放：



四 图1.2 6

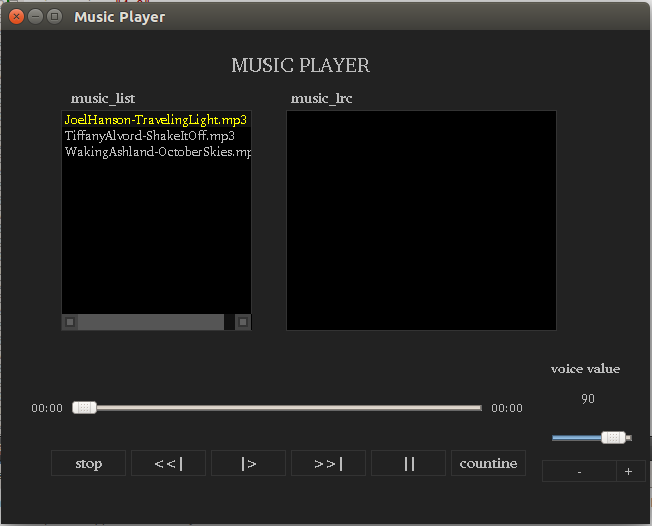
选择6退出：



四 图1.2 7

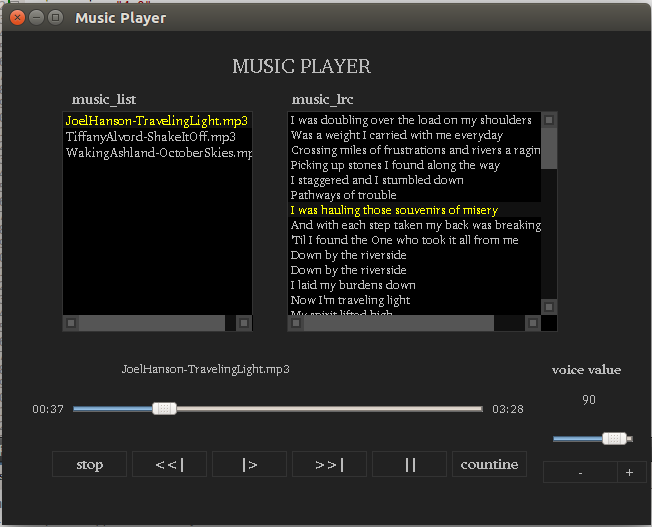
## 2、Qt MP3播放器的运行

2.1 初始界面



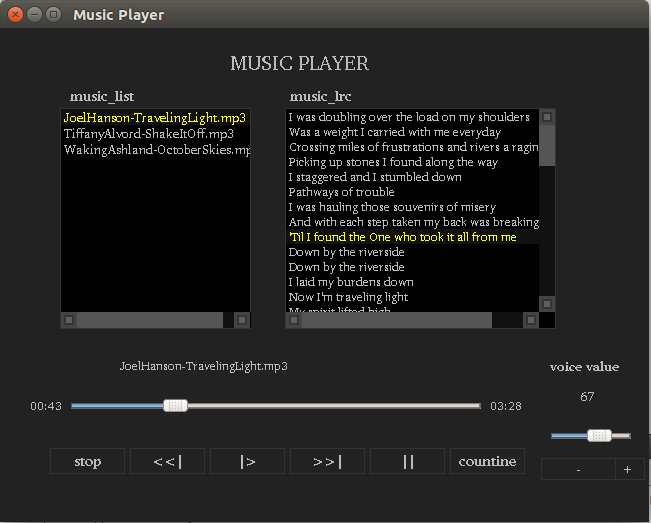
四 图2 1

2.2 点击播放，实现歌词、播放时间，播放进度条的动态显示



四 图2 2

2.3 使用进度条或者按钮实现声音的调节



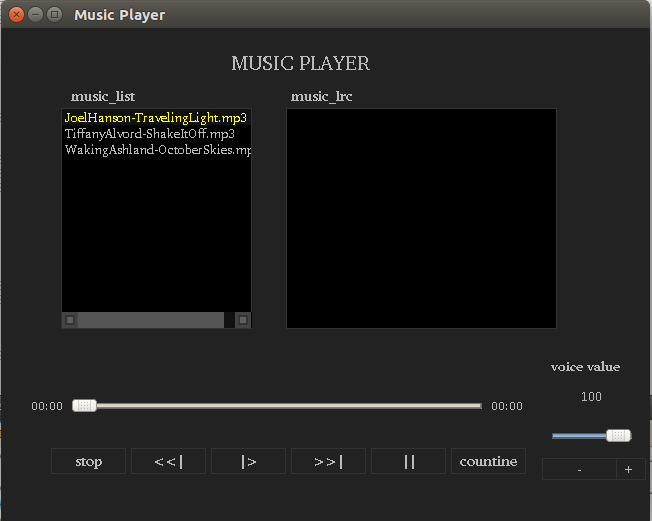
四 图2 3

2.4 单击上/下一首播放歌曲，实现暂停继续等



四 图2 4

2.5 点击stop停止歌曲播放



四 图2 5

# 五、【心得体会】

// 在设计过程中碰到的问题及最终解决方法

问题：在使用madplay时，使用system(“madplay 1.mp3 &”)不能正确地后台播放

解决办法：在播放命令后加入< /dev/null > /dev/null 2>1&

即使用system(" madplay 1.mp3 < /dev/null > /dev/null 2>1&")进行后台播放。

在播放命令后加入

问题：在安装arm-linux-gcc过程中遇到

sudo: arm-linux-gcc：找不到命令

解决办法：

由于Ubuntu16.04用的是64位的，在Ubuntu16.04版本当中应该安装执行：

sudo apt-get install lib32ncurses5 lib32z1

问题：在安装qt时出现系统缺少 QT 配置运行的相关库文件，导致无法安装QT

解决办法：安装相应的库文件

sudo apt-get install libxtst-dev libxrender-dev buid-essential

# 六、【附录】

## 1、小组分工

|  |  |  |
| --- | --- | --- |
| 学号 | 姓名 | 分工完成模块 |
|  | 任凯（组长） |  |
| 2015003588 | 刘港 |  |
|  | 张雅娴 |  |

## 2、程序清单

嵌入式课程设计源码：

arm移植：arm移植过程、Makefile、play.app

madplay字符界面：Makefile、play、play.app

qt project:

MusicPlayer:

Main.app、music2.qss、MusicPlayer.pro、MusicPlayer.pro.user、qrc.qrc、widget.cpp、widget.h、widget.ui

MusicPlayer-build-desktop-Qt\_4\_7\_3\_\_\_PATH\_\_Qt-4\_7\_3\_\_\_\_:……