《嵌入式系统》大作业实验报告：Part 2

陈俊哲　2020010964
梁　烨　2020080093
田正祺　2020080095

1　实验内容

实验背景：本次实验的硬件系统采用 MPU 和 MCU 双平台设计，其中 MPU 用于 Linux 操作系统开发，MCU 用于 ARM 体系结构与裸机编程。本次实验会在 Linux 系统下进行。

实验前置知识：本次实验需要使用 C 语言来编写文件读取函数，并利用虚拟机交叉编译代码后，在开发平台上运行。因此需要提前掌握一定的 C 语言基础。

实验目标：

2　实验部署

程序的测试是在运行 Ubuntu 20.04 虚拟机的 Windows 主机上进行。Ubuntu 中需要装测试以及交叉编译的依赖，可以通过运行 `setup-ubuntu.sh` 安装依赖。我们没有使用提供的交叉编译链。

开发板上需要将 STM32MP157 芯片启动拨码设为 EMMC 启动方式，即 101 状态，并插好电源开机。

开发板与主机可以直接通过以太网线链接（见章节 5），或通过以太网线链接到路由器。连到路由器的优点是可以通过外网访问，让所有组员可以在线上合作。开发板与主机之间的文件拷贝以及命令执行通过 Ubuntu 自带的 SCP 以及 SSH（没有使用 XShell 或 Xftp）（SSH 设置见章节 3.2）。若开放给公网建议在开发板上设置公钥认证（见章节 5）。

3　实验过程

3.1　源代码

程序由以下四个源代码文件组成：

- `waveheader.h`：定义了 Waveform Audio File Format (WAVE) 的头格式。参考的标准：https://datatracker.ietf.org/doc/html/draft-ema-vpim-wav-00。
- `audioplayer.h`：包含函数声明，以及定义了保存音频播放器的状态的结构体 `AudioPlayer`。
- `audioplayer.c`：实现了以下函数：
  - `ap_open`：读入 WAVE 文件并将信息写入 `AudioPlayer`。
  - `ap_get_header_string`：格式化 WAVE 文件的参数并写入字符串。
  - `ap_print_header`：将 `ap_get_header_string` 产生的字符串输出到命令行。
  - `ap_save_header`：将 `ap_get_header_string` 产生的字符串写入文件。
  - `ap_close`：释放 `ap_open` 所占用的资源。
- `main.c`：处理命令行参数，调用 `AudioPlayer` 相关的函数，输出信息以及错误。

3.2 编译与运行

测试编译（`make debug`）以及交叉编译（`make xc`）的命令都在 `Makefile` 中定义。使用的交叉编译器是 `arm-linux-gnueabihf-gcc`。由于 Part 1 没有使用第三方库，所以不需要做额外的链接。

将可执行文件拷贝到开发板上用 `make scp`，需要保证 `SXX_KEY`、`SXX_PORT`、`SXX_HOST` 变量与实际情况一致。在主机上和在开发板上正常运行程序的命令分别为 `make drun` 和 `make xrun`。

`Makefile` 中有些注释掉的命令是为了编译以及链接 Part 2 中的 ALSA 库，可以忽略。

4 实验结果

5 实验心得

在实验过程中遇到的问题及其解决方法如下：

- 编译 Alsa:为了得到开发板的平台信息,需要将 `config.guess` 拷贝到开发板上并运行,得到:`armv7l-unknown-linux-gnu`