在K64上运行RNN进行音频降噪

目录

[一、简介 1](#_Toc61863911)

[二、实验 1](#_Toc61863912)

[三、训练 5](#_Toc61863913)

# 一、简介

普通mcu受限于资源，很难做一些复杂的深度学习。不过虽然难，但是仍然可以做。

上次使用CNN进行手写数字识别。这次使用RNN进行音频降噪。本次音频降噪使用的是定点降噪，将MFCC和增益作为输入参数进行训练。

MFCC，梅尔频率倒谱，常用来作音频特征提取。

实际进行语音降噪（滤波）的部分，是一个均衡器，也就是大家播放器内调节低音高音。而均衡器(Equalizer)的本质是很多个平行的带通滤波器(Bandpass Filter). 我们神经网络的输出，就是均衡器内各个filter band的增益。RNN通过调节不同频率的增益来达到降噪的效果。

# 二、实验

1 所需工具：FRDM-K64，Python3.7，pip，IAR

2 下载深度学习框架源码，<https://github.com/majianjia/nnom>这是个纯C框架，并不依赖硬件结构，移植很方便

3 移植，我们选择bubble这个例程，将inc，port和src文件夹添加到工程中，如图

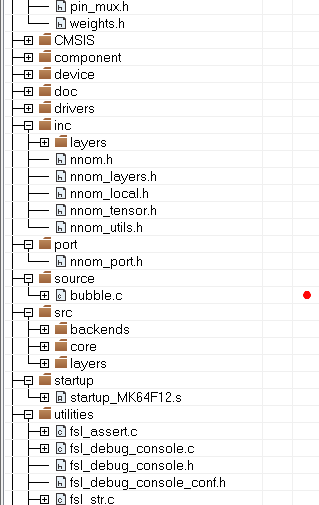
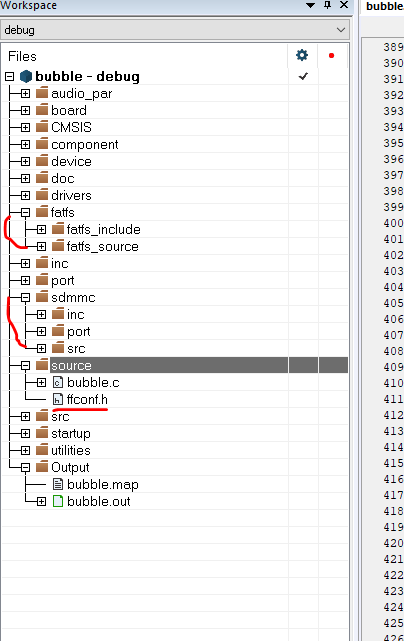


图1

打开port.h，将NNOM\_LOG的串口改成PRINTF(\_\_VA\_ARGS\_\_)，打开icf文件，将堆大小改成0x5000，define symbol \_\_size\_heap\_\_ = 0x5000;

这个库用的malloc，就是从这里分配内存，小了的话，就没办法运行网络。

4 移植fatfs文件系统到bubble工程



5 在bubble.c文件里，添加头文件

#include "nnom\_port.h"

#include "nnom.h"

#include "weights.h"

#include "denoise\_weights.h"

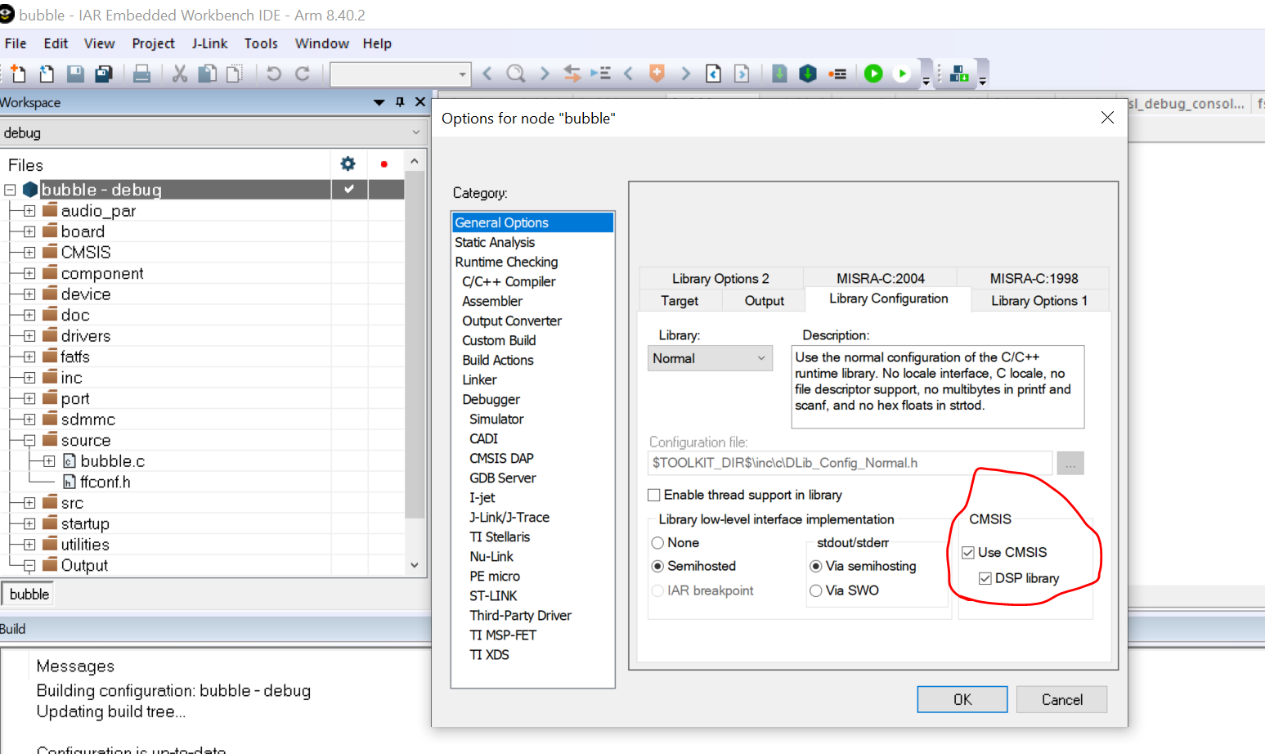
#include "mfcc.h"

#include "wav.h"

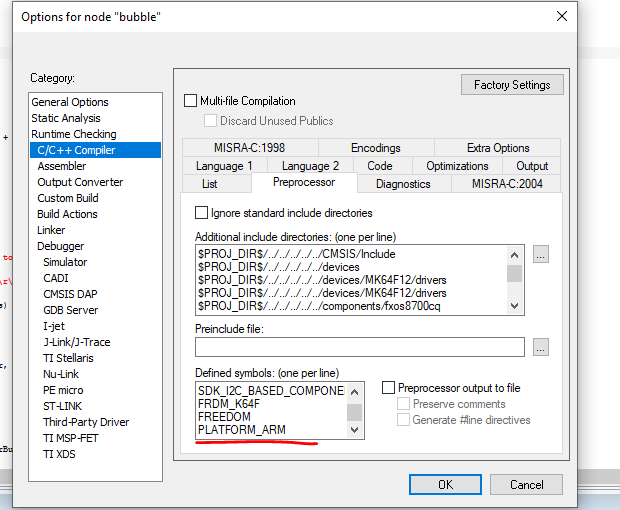
#include "equalizer\_coeff.h"

6 找到框架里examples\rnn-denoise下的main.c，main\_arm.c是为stm32使用的。Main.c是为windows运行例程提供的，这个代码是通过打开音频文件，然后运行网络，最后生成降噪音频。用这个代码方便进行实验。这个文件里的关于文件操作的api都需要我们手动改成mcu的api。我增加了进度条显示，可以看附件。

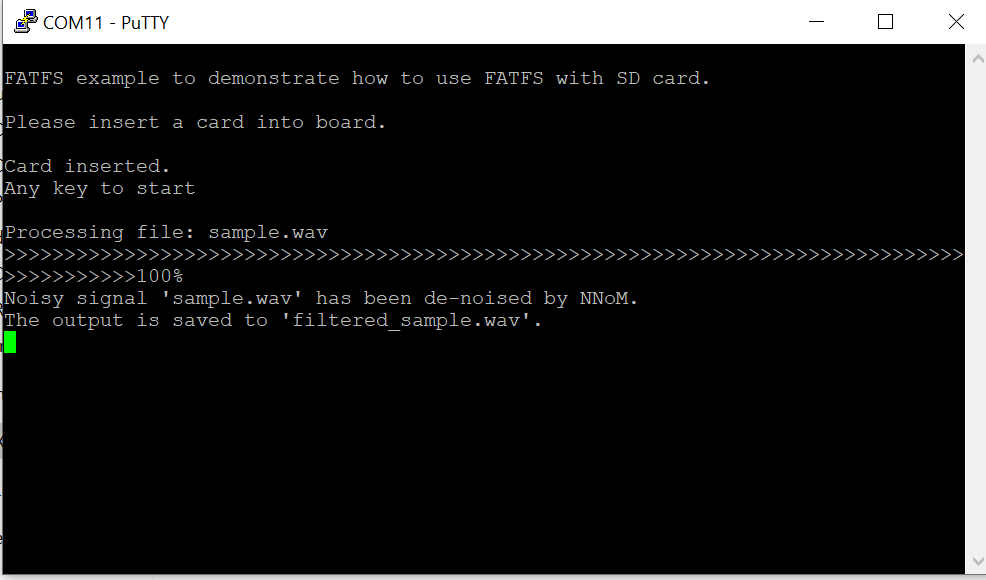
7 这个框架还使用了dsp，所以需要开启dsp支持，如图



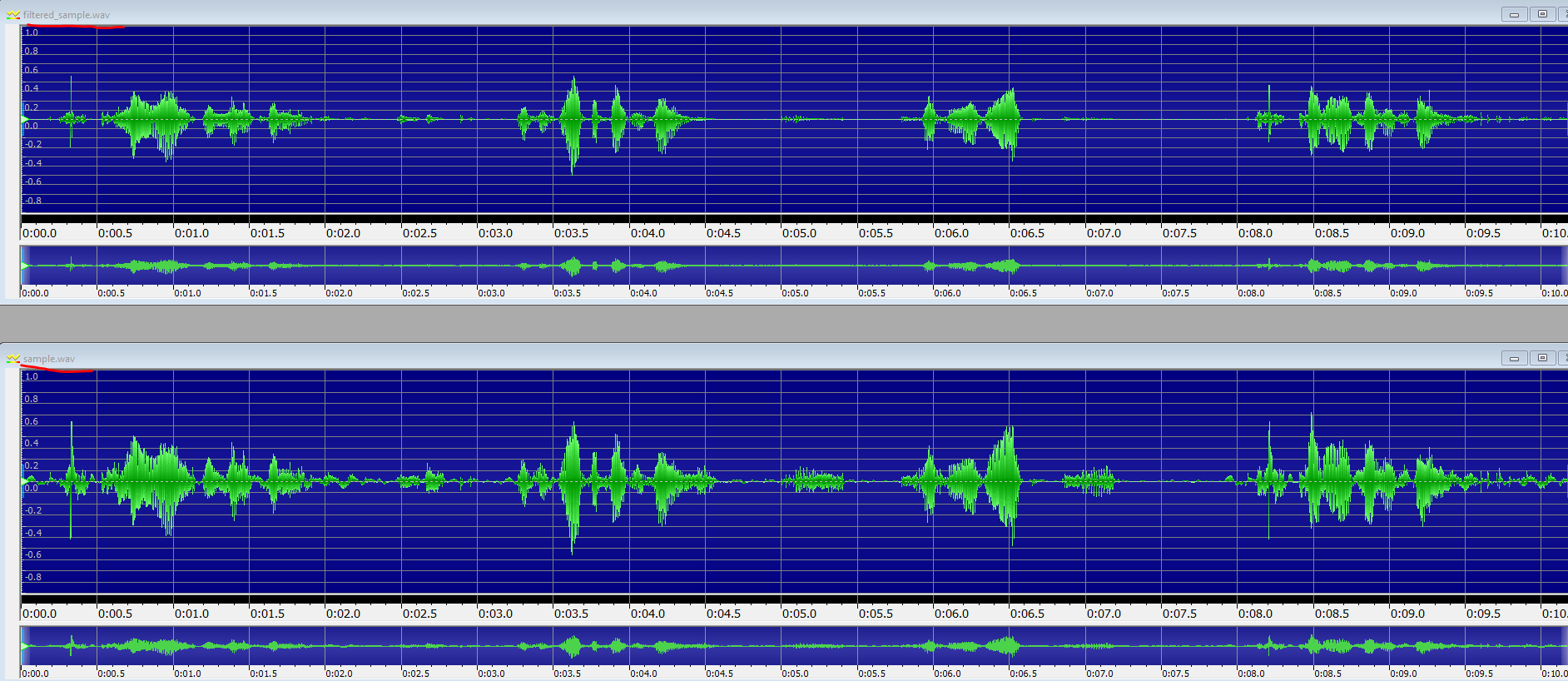
8 添加宏定义



9 编译通过后，下载。测试的wav，需要取名叫sample.wav，mcu会对他降噪，然后生成filtered\_sample.wav，把sd卡插入电脑就可以听降噪的音频。



这是音频分析软件分析的音频，上面是降噪后的，下面是原声。可以看到很多杂音被抑制了。



# 三、训练

通过以上步骤，我们就实现了音频降噪，那么这个数据是怎么训练的？你需要安装tensorflow，以及keras。具体怎么训练的在README\_CN.md文件有写。这个tensorflow需要安装gpu版本，并且需要安装cuda。请自行查阅安装

1 下载语音数据<https://github.com/microsoft/MS-SNSD>，放入MS-SNSD文件夹下

2 使用pip3 安装requirements.txt里的工具

3 直接运行noisyspeech\_synthesizer.py，这个会报错，需要把15行的float改成int类型

这样就不回报错了

4 运行脚本gen\_dataset.py，生成mfcc和增益gain

5 运行main.py，这样就会生成sample的噪音文件，可以给我们用来测试，同时也生成

weights.h

denoise\_weights.h

equalizer\_coeff.h