



南京理工大学
NANJING UNIVERSITY OF SCIENCE & TECHNOLOGY

学士学位论文

基于 FPGA 和 MEMS 麦克风的 声源测向系统设计

文志彬

(学生姓名)

9201080N0133

(学号)

指导教师

赵兆

副教授

学生学院

电子工程与光电技术学院

专业

微电子科学与工程

研究方向

嵌入式系统

论文提交时间

2024 年 5 月

学士学位论文

基于 FPGA 和 MEMS 麦克风的 声源测向系统设计

作 者：文志彬

指导教师：赵兆 副教授

南京理工大学

2024 年 5 月

Bachelor Dissertation

**Design of Sound Source Localization System
Based on FPGA and MEMS Microphone**

By

Zhibin Wen

Supervised by Prof. Zhao Zhao

Nanjing University of Science & Technology

May, 2024

声 明

本学位论文是我在导师的指导下取得的研究成果，尽我所知，在本学位论文中，除了加以标注和致谢的部分外，不包含其他人已经发表或公布过的研究成果，也不包含我为获得任何教育机构的学位或学历而使用过的材料。与我一同工作的同事对本学位论文做出的贡献均已在论文中作了明确的说明。

学生签名：_____ 年 月 日

学位论文使用授权声明

南京理工大学有权保存本学位论文的电子和纸质文档，可以借阅或上网公布本学位论文的部分或全部内容，可以向有关部门或机构递交并授权其保存、借阅或上网公布本学位论文的部分或全部内容。对于保密论文，按保密的有关规定和程序处理。

学生签名：_____ 年 月 日

摘要

随着消费电子技术的高速发展，越来越多的小型嵌入式设备，如智能音箱、无线耳机和笔记本电脑等，都开始集成微型麦克风阵列。其中，基于麦克风阵列的声源测向应用是此类产品实现干扰抑制、语音识别等相关应用功能的重要步骤之一。同时，近年来麦克风小型化的技术也越来越先进，以微机械系统技术 MEMS 为代表的数字式集成麦克风在大量嵌入式产品中得到应用，使得构建成本低、集成度高、性能优异、小体积的麦克风阵列成为可能。针对相应嵌入式系统中的多路音频信号处理任务，基于 FPGA 的软硬件开发方案因具有高速并行处理和灵活设计能力等特点而备受关注。

面向上述现状，本文设计并实现了一种基于 FPGA 和 MEMS 麦克风的声源测向系统，该系统构建了 MEMS 数字麦克风阵列实现音频采集，并采用改进权重的广义互相关（GCC）算法来实现声源测向，充分利用了 FPGA 和 MEMS 麦克风的低功耗和小体积特性，为声源测向提供了一种高效解决方案。首先对声源测向及其目前的研究现状进行了概述，比较了几种常见的测向方法，通过分析相应的优缺点，选择了适合 FPGA 实时实现的基于广义互相关法（GCC）的时延估计（TDOA）类方法，并探讨了不同的权重选择对定位精度的影响。然后，针对麦克风阵列方案，选择平面正十字阵作为本系统的麦克风阵列阵型。详细说明了本文所研系统采用的 Zynq SoC 硬件开发平台，并用 Verilog HDL 语言实现了系统的各个软件功能模块，并对各个模块进行了仿真。最后，对所研声源测向系统进行了实际测试验证。测试结果表明，在四路麦克风输入的情况下，系统仅使用了 37 个 DSP 单元，系统时钟频率为 50MHz 时，FPGA 的功耗为 0.44W，完成一帧计算的时间仅为 1.2ms，在开阔的室外环境中远场声源目标测向误差小于 4° 。

关键词：声源测向，麦克风阵列，时延估计，FPGA，MEMS

Abstract

With the rapid development of consumer electronics technology, an increasing number of embedded devices, such as smart speakers, wireless earphones, and laptops, are beginning to integrate miniature microphone arrays. These microphone arrays play a crucial role in enabling advanced functionalities like interference suppression and voice recognition in such products. Sound source localization based on microphone arrays is a key step in achieving these functionalities. Concurrently, the technology for miniaturizing microphones has also advanced significantly in recent years. Digital integrated microphones, represented by MEMS (Micro-Electro-Mechanical Systems), are being widely used in numerous embedded products. This technological advancement makes it possible to construct low-cost, highly integrated, high-performance, and compact microphone arrays. For multi-channel audio signal processing tasks in corresponding embedded systems, FPGA (Field-Programmable Gate Array)-based hardware-software development solutions are gaining significant attention due to their high-speed parallel processing and flexible design capabilities. FPGAs offer the advantage of real-time processing and customization, which are essential for applications requiring quick and accurate sound source localization.

In this context, this paper designs and implements a sound source localization system based on FPGA and MEMS microphones. The system uses a MEMS digital microphone array commonly found in embedded devices for audio acquisition. It employs an improved weighted Generalized Cross-Correlation (GCC) algorithm for sound source localization. The GCC algorithm is known for its effectiveness in estimating the time difference of arrival (TDOA) between signals received at different microphones, which is crucial for accurately determining the direction of the sound source. The system fully leverages the low power consumption and small size features of FPGA and MEMS microphones, providing an efficient and practical solution for sound source localization. Firstly, the paper provides an overview of sound source localization and its current research status, comparing several common localization methods. By analyzing their respective advantages and disadvantages, a TDOA method based on the Generalized Cross-Correlation (GCC) approach, suitable for real-time FPGA implementation, is selected. The impact of different weight selections on localization accuracy is also discussed in detail. The choice of weights in the GCC algorithm can significantly influence the precision of the localization, and this paper explores various weighting strategies to optimize performance.

Subsequently, for the microphone array scheme, a planar cross array is chosen as the II

microphone array configuration for this system. The planar cross array provides a balanced distribution of microphones, which enhances the accuracy and reliability of the sound source localization. The paper details the Zynq SoC (System on Chip) hardware development platform used in the system. The Zynq SoC integrates FPGA with a processor, offering a versatile platform for implementing complex signal processing tasks. Each software functional module is implemented using Verilog HDL (Hardware Description Language), followed by simulations of each module to ensure their correct operation. Finally, the designed sound source localization system is subjected to practical testing and validation. The test results show that, with four microphone inputs, the system uses only 37 DSP (Digital Signal Processing) units. At a system clock frequency of 50MHz, the FPGA's power consumption is 0.44W. The time taken to complete one frame of computation is only 1.2ms, demonstrating the system's efficiency. In practical scenarios, the sound source localization error for far-field targets in an open outdoor environment is less than 4° , indicating a high level of accuracy. These results validate the effectiveness of the proposed system and its potential for integration into various consumer electronic products requiring sound source localization capabilities.

Keywords: Sound Localization, Microphone Array, TDOA, FPGA, MEMS

目 录

摘要	I
Abstract	II
目录	IV
1 绪论	1
1.1 课题背景及研究意义	1
1.2 国内外研究现状	3
1.3 本文的主要研究内容与结构	5
2 基于麦克风阵列的声源测向基础	7
2.1 麦克风阵列结构	7
2.1.1 麦克风介绍	7
2.1.2 阵列的几何结构	8
2.2 声音传播模型	9
2.2.1 远场模型	9
2.2.1 近场模型	10
2.3 现有麦克风阵列声源测向方法	11
2.3.1 基于子空间类的方法	11
2.3.2 基于波束形成的方法	12
2.3.3 基于时延估计的方法	13
2.4 常用的时延估计方法	14
2.4.1 基本互相关	14
2.4.2 广义互相关	15
2.5 本章小结	16
3 声源测向系统硬件设计	17
3.1 总体设计方案	17
3.2 麦克风阵列设计	18
3.2.1 麦克风选型	18
3.2.2 阵列结构设计	19
3.2.2 阵列电路设计	19
3.3 FPGA 平台选择	20
3.3.1 主控芯片的选择	20
3.3.2 开发平台功能介绍	21

3.4 本章小结.....	24
4 声源测向系统软件设计	25
4.1 音频采集模块	25
4.2 音频预处理模块	26
4.2.1 音频滤波	28
4.2.2 音频插值	30
4.3 有声检测模块	33
4.4 时延估计模块	34
4.5 方位角估计模块	39
4.6 本章小结.....	41
5 系统测试及验证.....	42
5.1 系统硬件性能评估	42
5.1.1 系统资源占用分析	42
5.1.2 系统音频采集性能评估	43
5.2 系统测向功能测试	44
5.3 本章小结.....	45
6 总结与展望	46
6.1 本文总结.....	46
6.2 本文展望.....	46
致 谢	47
参考文献	48
附 录	50

图表目录

图 1.1 “回旋镖”反狙击手声探测仪器.....	1
图 1.2 科大讯飞声学成像仪.....	2
表 1.1 不同声源测向方法的对比.....	5
图 1.3 本文研究内容框图.....	5
表 2.1 MEMS 麦克风与驻极体麦克风的性能比较.....	7
图 2.1 常见的麦克风阵列摆型.....	8
图 2.2 远场模型示意图.....	9
图 2.3 近场模型示意图.....	10
表 2.2 常见的广义互相关加权函数.....	16
图 3.1 硬件系统框图.....	17
图 3.2 MSM26S4030H0 数字 MEMS 麦克风.....	18
图 3.3 MSM26S4030H0 频率响应图.....	19
图 3.4 正十字阵结构示意图.....	19
图 3.5 MSM26S4030H0 参考设计图.....	20
图 3.6 Zynq 7000 SoC 架构图.....	21
图 3.7 FLASH 和 SDIO 与 FPGA 的连接图.....	22
图 3.8 Zynq 系统模式引脚与启动模式的关系.....	23
图 3.9 UART 传输协议示意图.....	23
图 3.10 上位机串口终端示意图.....	24
图 4.1 软件模块的整体框图.....	25
图 4.2 I2S 左采集时序图.....	26
图 4.3 音频采集模块仿真结果.....	26
图 4.4 预处理模块整体框图.....	27
图 4.5 AMBA AXI4-Stream 接口定义.....	27
图 4.6 直接形式的 N 阶 FIR 滤波器.....	28
图 4.7 FIR 滤波器量化后的频率响应.....	29
图 4.8 FIR 滤波器前后时域波形图.....	29
图 4.9 FIR 滤波器仿真结果.....	30
图 4.10 不同采样率对单对线阵麦克风声源测向角度颗粒度的影响.....	31
图 4.11 CIC 滤波器的组成.....	31
图 4.12 插值模块 CIC 滤波器频率响应（量化后）.....	32

图 4.13 插值模块仿真结果.....	32
图 4.14 有声检测流程图.....	34
图 4.15 有声检测模块仿真结果.....	34
图 4.16 时延估计模块内部框图.....	34
图 4.17 时延估计 FFT 模块仿真结果.....	35
图 4.18 相同数据在 NumPy 中的仿真结果.....	35
图 4.19 随机取整效果展示.....	36
图 4.20 Divider Generator IP 输出格式.....	37
图 4.21 频域加权模块整体框图.....	37
图 4.22 时延估计模块整体处理耗时仿真.....	37
图 4.23 无加权（结果正确）和 PHAT 加权（结果错误）表现对比.....	38
图 4.24 无加权和改进的 PHAT 加权表现对比.....	39
图 4.25 时延估计模块仿真结果.....	39
图 4.26 方位角估计模块运行流程图.....	40
图 4.27 片上通讯架构图.....	40
图 4.28 测向结果发送仿真结果.....	41
图 5.1 系统 Block design 图.....	42
表 5.1 FPGA 硬件资源使用量.....	42
图 5.2 系统功耗估计报告.....	43
图 5.3 系统时序报告.....	43
图 5.4 ILA 采集结果（1000Hz 音频）.....	44
图 5.5 ILA 采集结果（2000Hz 音频）.....	44
图 5.6 测试平台原型.....	45
表 5.2 测向系统功能测试结果.....	45

1 绪论

1.1 课题背景及研究意义

声音这个概念从诞生开始，就是信息的重要载体^[1]，从语言的发展开始，人类就可以通过互相交流来传递信息。现在人类虽然可以通过非常多的渠道获取信息，但大部分信息的流动还是基于语音、文字和图像这三个主要载体。并且，与其他的信息载体不同的是，声音有着自身的独特优势，人类在说话时，可以把自己的情绪包含其中，使得声音所传达的意思更为丰富，让听者更好地理解表达的意思。这一切都足以说明，声音在我们生活中的重要地位，在科技日新月异的今天，如何利用与提取声音中的海量信息就成了一个重要的课题。

然而在现实生活中，声音信号往往伴随着大量噪声信号一起出现，在这样的情况下，麦克风阵列应运而生，在麦克风阵列的帮助下，可以实现有选择地对声音信号进行处理，有效地提取出里面的信息。而声源定位技术就是麦克风阵列处理技术里重要的一环，作为一种被动检测技术，可以通过分析声音到达不同麦克风的时间差、声音的能量信息和声音的频率信息来获取声源在空间中的信息，已实现声源的定位^[2]。



图 1.1 “回旋镖”反狙击手声探测仪

声源定位技术的发展很大程度上和军事技术的发展挂钩，尤其是在二战时期，因为声波的性质与电磁波有几分相似，人们便基于雷达技术的拓展，创造出了各类不同用途的声源探测器用于预警^[3]。进入二十一世纪之后，许多系统级别的商业产品也开始出现，例如瑞典 Swctron 公司开发的基于飞行噪音的 Helisearch 直升机定位系统^[4]，而这些系统中最有名的莫过于 2003 年美国 BNN 公司在美国国防部支持下开发的“回旋镖”反狙击手声探测仪^[5]，如图 1.1 所示，一个系统单元通过桅杆安装在车辆上，利用七个小型麦

克风检测枪口的爆炸声和超音速子弹穿过空气产生的超音速声波，基于到达时间差计算出子弹的来向、射手的距离和高度，整体定位精度为正负 15 度，探测范围 3 到 150 米，“回旋镖”在极端天气、开阔地和城市环境中均能工作，无论是静止还是移动。BBN 表示，在车速低于每小时 50 英里（80 公里）时，误报率低于每千小时不到一次。但目前这套系统也存在成本高、处理时间过长和复杂噪声下效果不佳的缺点，这些问题也是大数声源定位系统的通病。

在工业领域，声源定位技术应用的领域也有很多，其中最为常见几个领域噪声源定位、仪器故障排查声纳探测等等。2001 年，美国国家仪器公司开发了一套基于麦克风阵列的声源定位系统^[6]，对跑道上的客机进行噪声记录分析，成功帮助波音公司判断出了飞机起降阶段噪声的来源。2009 年，来自石家庄铁道学院的司春棣等人开发了一套基于声阵列技术的汽车内噪声源识别系统^[7]，通过将基于波束形成的噪声分析技术和光学图像的结合分析，有效分析出了车内噪声的来源，为汽车内的噪声控制提供了科学依据。2024 年，张中盘等人基于对 VMD 和包络谱峭度法的声音信息进行广义互相关时延估计^[8]，成功实现了一套托辊故障声源定位系统并投入现场运行。2023 年，中国科学院声学研究所设计了一套基于水下声学滑翔机联合的水下声源定位系统^[9]，利用水下滑翔机在东印度洋北部海域获取的声传播数据，对宽带脉冲信号的多途传播特性进行了分析，随后提出了一种基于脉冲波形结构匹配的声源距离估计方法，仅仅使用了一个听水器，就实现了对于声源距离的估计。在此基础上，通过两台水下声学滑翔机联合定位的方式，成功实现了水下声源距离和方位的同步估计。实现了双机的同步估计，对于 200 米深度的声源，估计误差小于 4%。



图 1.2 科大讯飞声学成像仪

在民用领域，声源定位系统主要在会议系统、智能音箱和助听器领域应用较多。2006 年，Polycom 推出了一套适用于小型会议室的智能会议系统，可以根据内置麦克风矩阵，

自动判断发言人的位置并对发言人的人声进行增强^[10]。2009 年，微软发布的 Kinect 系统通过结合麦克风矩阵和深度相机的数据，实现了对于使用者语音指令和肢体动作的识别^[11]。2021 年，科大讯飞推出了声学成像仪，硬件层面包括 64 路的麦克风阵列和一个摄像头，利用高采样率的数据及超声波声源定位技术，通过声像图与可见光图像叠加的成像方式，实时展示声源在空间的分布状态。

近年来，随着消费电子市场的迅速扩张，配备麦克风阵列的嵌入式设备，如无线耳机、语音助手以及智能音箱等，已变得越来越常见。这些设备中的声源定位技术起着至关重要的作用。而声源测向，就是声源定位系统的一个重要功能。以智能音箱为例，智能音箱在响应使用者的需求时，一般是先对使用者的声音进行测向，然后根据使用者的方向信息进行语音增强，最后完成语音的识别。从上述例子可以看出，声源测向是音频信号处理中一个很重要的步骤。

然而，与传统的大型麦克风阵列相比，这些嵌入式设备的麦克风阵列通常具有较小的孔径、有限的麦克风数量和间距，以及有限的计算能力。这些限制因素往往会影响声源定位的准确性，进而影响用户的实际使用感受。基于以上因素，本文研究了一种结合数字 MEMS 麦克风和 FPGA 的声源测向系统，通过 FPGA 的高拓展性、高能效和数字 MEMS 麦克风的高集成度，可以有效弥补上面介绍中提到的缺点，具有较好的研究意义和研究价值。

1.2 国内外研究现状

声源测向算法经过长久的发展，已经是一门较为成熟的技术，现有基于麦克风矩阵的声源测向方法大致可以分为三类，分别是基于子空间类、基于波束形成和基于时延估计三种方法^[12-14]。

（一）基于子空间类的声源测向方法

基于子空间类的估计方法本质原理是基于获取到的阵列信号，通过子空间分解的方式，来计算空间谱的自相关矩阵^[15]。其的运用场景大多集中在远场（例如雷达和天线）信号的窄带的时延估计中，之后拓展到语音信号的定位上。

1973 年，Pisarenko 首先提出了 MUSIC (Multiple Signal Classification) 算法^[16]，后由 Schmidt 于 1977 年在 Northrop Grumman 工作期间独立开发^[17]，这可以说是最为经典的子空间分解算法。MUSIC 算法假设信号向量由若干未知频率的复指数信号组成，在高斯白噪声的影响下，通过线性模型描述。该算法的关键假设是信号源的数量少于测量向量的元素数量。通过分析自相关矩阵的特征向量和特征值，MUSIC 算法能够将信号空间和噪声空间分开，并利用噪声子空间的特征向量来提高频率估计的性能。

1986 年，来自斯坦福大学的 R. Roy 及 T Kailath 等人提出了 ESPRIT (Estimation of

signal parameters via rotational invariance techniques) [18], 即旋转不变技术估计信号参数技术，是一种用于确定背景噪声中一系列正弦波的参数的技术。ESPRIT 最初是为频率估计提出的，但随着相控阵系统在日常技术中的引入，它也被用于到达角估计。其的基本原理是通过将接收到的信号数据分解为子阵列，并利用子阵列之间的几何关系来估计信号的参数。该方法利用信号的旋转不变特性，通过构建和操作特定的选择矩阵来实现对信号参数的估计。ESPRIT 的优点包括其对模型参数少的依赖，以及能够提供比传统方法更高的分辨率和估计精度。然而，ESPRIT 需要知道输入信号的数量，并且对阵列的几何结构有一定的要求。

基于子空间类估计的算法都有一个特点，那就是优秀的空间分辨率，这就让测向的精度变得很高，但这种算法最早是针对窄带信号（比如雷达信号）进行设计的，而声音信号严格意义上并不算是一种窄带信号，这就为处理带来了障碍。一般将高分辨率谱估计应用于声音信号时，需要先进行频带分解，会引入更多的计算资源，在本身矩阵分解就需要大量算力的情况下，整体算法在边缘设备的部署就会变得很困难。

（二）基于波束形成的声源测向方法

波束形成技术的本质是利用一个麦克风阵列捕捉信号，每个麦克风接收到的信号会因为声源到麦克风的距离不同而存在时间延迟。波束形成算法通过对这些信号进行时间延迟的调整和加权组合，形成一个指向特定方向的“波束”，再通过遍历和能量峰值搜索，确定最大能量的声音方向，认为这是算法估计的最佳位置^[19]。

波束形成目前在音频指向采集方面已经被很多设备和系统所使用，但这种技术进行声源测向效果比较一般。本质上基于波束形成的声源定位技术其实是一种最大似然估计^[20]，得到的是一个区间内的最佳估计结果，实现依赖于声源和噪声的先验信息^[21]，这就导致了算法很难去获得一个全局的最优解。如果需要利用网格搜索或其他方式来进一步从局部最优解来获得全局最优解，计算量也会增加^[22]。

（三）基于时延估计的声源测向方法

基于时延估计的声源测向方法和上述的两个方法都不一样，是一个两步定位方法，先对收到的声音信号进行延迟估计，之后用得到的时延信息结合阵列本身的结构信息来求解声源的方位角。

1984 年，C Knapp 和 G Carter 等人将广义互相关方法 GCC (Generalized Cross Correlation) 应用于声源定位^[23]，通过在频域加权的方式来减少噪声信号对于声源定位的影响。1999 年，Brandstein 等人研究了相位变换加权 PHAT (Phase Coherence Transform) 在广义互相关方法上的应用。我国方面也有不少的研究，2011 年，来自山东大学的夏阳和张元元等学者提出了 MPHAT 方法，提高了矩形麦克风矩阵在噪声环境下的定位精度^[25]。2022 年，长安大学的李登峰等人提出了基于 GCC-PHAT- β 的改进算法，有高斯白

噪声的两路单频正弦波信号下, 利用原本的 PHAT 的指数权重和二次相关, 成功提升了系统在低信噪比情况下的定位精度^[25]。

基于时延估计的测向方法有几个优点。首先, 它只需要少量麦克风就能实现, 这使得它比其他方法更经济。其次, 它的算力要求比较低, 实时性好, 这使得它适用于各种嵌入式应用, 如会议系统、智能机器人和安全监控。第三, 它是一种被动方法, 这意味着它不需要传感器主动发射信号, 这在能量消耗和复杂性方面是一个优势。

三种方法的主要特性如表 1.1 所示:

表 1.1 不同声源测向方法的对比

声源测向方法	计算量	抗噪能力	麦克风数量	定位精度	稳定性	实时性
子空间类估计	大	优秀	多	良好	良好	一般
波束形成	大	良好	多	优秀	优秀	一般
时延估计	小	一般	少	一般	一般	优秀

1.3 本文的主要研究内容与结构

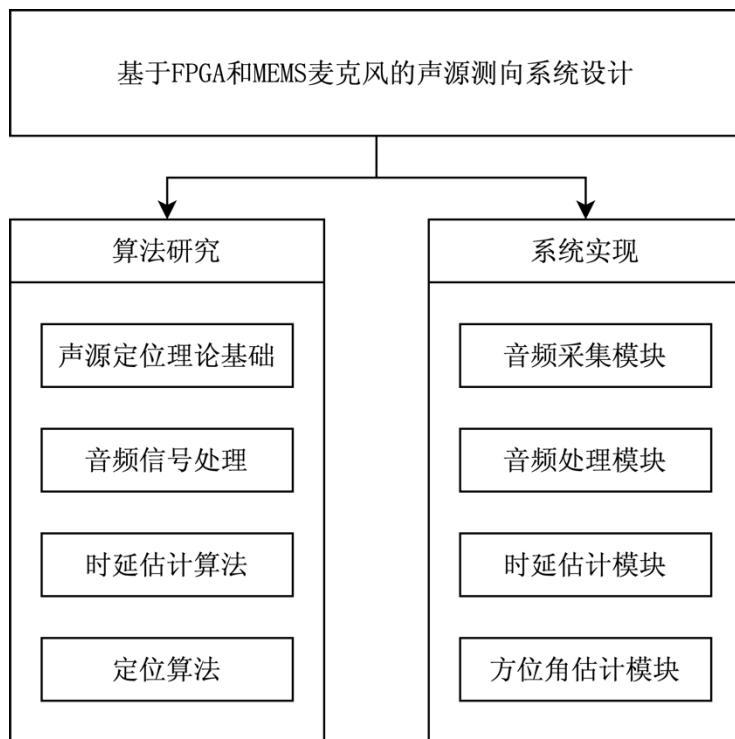


图 1.3 本文研究内容框图

本文的主要研究内容如图 1.3 所示, 对基于 FPGA 和 MEMS 麦克风的声源测向系统的算法和系统实现展开研究。在算法研究部分主要对声源定位理论基础、音频信号处理、时延估计算法和定位算法进行研究; 在系统实现部分主要对音频采集模块、音频预

处理模块（插值和滤波）、有声检测模块、时延估计模块和方位角估计模块进行数字电路的设计与实现。

本文的结构安排如下：

第一章为绪论，主要介绍麦克风阵列中声源定位算法的相关背景与声源定位系统的发展过程，大致总结了三种不同类型的声源定位算法并分析了不同算法的不同特性，阐述了本文的研究内容。

第二章为基于麦克风阵列的声源测向理论基础。介绍麦克风阵列中声源测向相关的理论基础，包括麦克风阵列的几何模型、声音传播的模型、现有声源测向算法的介绍，基于时延估计算法的测向原理，为后面的系统硬件和软件设计打下理论基础。

第三章为系统的硬件设计，主要介绍整体系统的硬件实现方案，包括 MEMS 麦克风的工作原理、麦克风矩阵的几何与电路设计、所用 FPGA 开发平台的介绍，最后给出了系统的整体结果。

第四章为系统的软件设计，主要介绍了声源定位算法在 FPGA 平台上的 Verilog HDL 实现，以及音频采集模块、音频预处理模块、有声检测模块、时延估计模块、片上通讯模块和方位角估计模块的仿真验证结果。

第五章为系统的测试与实验，包含了系统平台的搭建、系统整体的测试、MEMS 麦克风音频采集模块的测试和测向功能的测试，并结合测试结果对系统性能做出评价。

第六章为整体的总结与展望，主要总结了本文所完成的工作与存在的不足，并提出对应的改进可能。

2 基于麦克风阵列的声源测向基础

本章主要介绍麦克风阵列的声源测向的相关理论，包括麦克风阵列的几何模型、声音信号的模型、时延估计算法和基于时延估计算法的测向原理。

2.1 麦克风阵列结构

2.1.1 麦克风介绍

在麦克风阵列设计中，麦克风的选型至关重要，其的性能将会直接影响到麦克风阵列接收到信号的质量。作为一种重要的传感器器件，麦克风已经在发展中衍生出多种类型，其中最主要类型为圈动式和电容式。

动圈式麦克风是最古老的麦克风类型之一，它的原理是利用电磁感应记录音频^[26]。其以其坚固耐用而著称，通常是舞台麦克风的首选，因为动圈式麦克风通常可以承受巨大的声音、摇摆的动态和湿度的变化，同时还能有效地捕捉声音。这种麦克风在高声压级下表现出色，它的缺点是不如电容式麦克风灵敏，但在嘈杂环境中使用时，它能够有效地过滤掉噪音。

电容式麦克风是一种灵敏的设备，它使用薄薄的导电膜片来捕捉声音。因为整个介质的体积小，所以电容式麦克风的灵敏度较高，瞬态响应优秀。但电容式麦克风对压力、电平和湿度相关变化更为敏感，在嘈杂环境下表现不如动圈式麦克风，一般电容式麦克风大多在录音棚中使用。

而在麦克风阵列的应用中，则主要分为 MEMS 麦克风和驻极体麦克风，他们的特点如表 2.1 所示：

表 2.1 MEMS 麦克风与驻极体麦克风的性能比较

	MEMS 麦克风	驻极体麦克风
优点	尺寸小、集成度高、功耗低、兼容数字输出、稳定性和可靠性高	声音质量优异、宽频响、灵敏度高、结构简单
缺点	声音质量有限、会因为体积牺牲频响范围	成本较高、对环境条件敏感、整体功耗较高

因为 MEMS 本身的功耗较低，加之其可以支持数字输出，使得其和数字 I/O 较多的 FPGA 极为般配，FPGA 可以用自己的并行处理能力同时处理多路输入进来的音频信号，因此本系统的音频采集部分使用的集成的 MEMS 数字麦克风。

2.1.2 阵列的几何结构

系统的定位准确性不只受到音频采集质量的影响，还与麦克风阵列的几何布局密切相关。阵列性能受到多种因素的影响，包括整体的几何设计、麦克风的数量以及麦克风之间的距离设置。通常情况下，麦克风的数量越多，信号处理的效果越佳。然而，这也会导致处理的复杂性增加；阵元的尺寸越大，测向估计的分辨率越高，但是整体的截止频率却会被限制^[27]。两个麦克风并排摆放，就可以实现一个平面内 0 到 180 度内的定位，如果有三个麦克风，那就可以实现二维平面内的定位，也就是测向。理论上，只需要一个阵列有大于三个麦克风，就可以实现声源在三维空间的定位。

按照麦克风的数量和空间结构，麦克风阵列可以分为空间型和平面型。一般常见的有一维线性阵列，二维平面阵列和三维空间阵列，如图 2.1 所示：

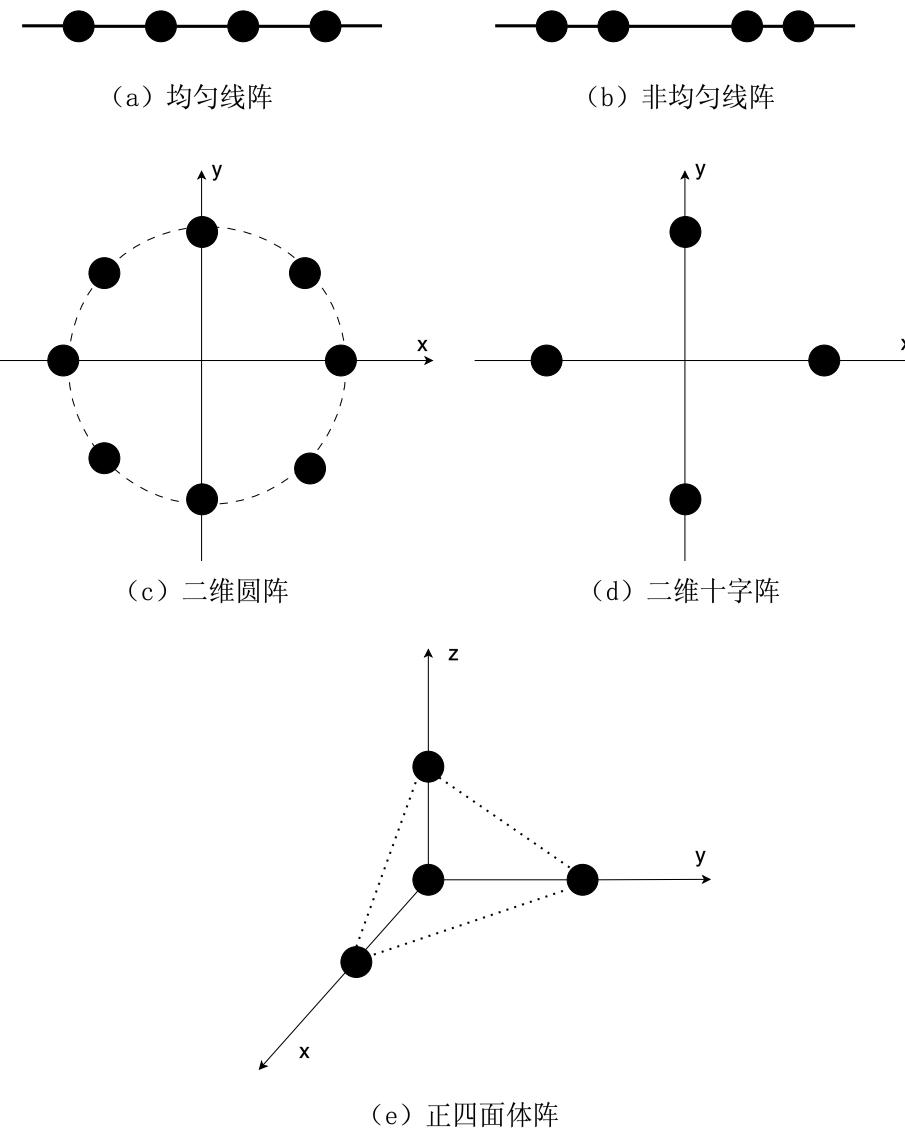


图 2.1 常见的麦克风阵列摆型

一维阵列是所有阵列里最简单的，如图 2.1 中的图（a）和（b）所示，所有麦克风阵列都在一条直线上，不受声源定位算法的限制，计算复杂度也最低，但是能定位的区域有限。二维阵列常见的主要是图 2.1（c）和图 2.1（d）中展示的二维圆阵列和二维十字阵，理论上来说可以估计声源的位置，但是和只有两个麦克风的二维线阵无法分辨声源是来自正半边和负半边一样，这种二维麦克风矩阵在对空间中的声源进行定位时也没有办法判断出声源是来自平面的上面还是下面，如果需要空间定位的话，就需要使用图 2.1（e）的这种的三维阵列，但这种阵列复杂度高，计算量也显著增加。

2.2 声音传播模型

在麦克风阵列中，由于声源离阵列的距离不确定，又由于声波传播的特性，就需要用不同的方式来分析。当声源离阵列的距离较远时，声源到阵列的距离就会显著大于阵元之间的距离，这样以来阵列离每个麦克风接收到信号的差别很小，可以近似认为声波是以一个平面的形式到达麦克风矩阵的；当声源离矩阵的距离较近的时候，每个阵元接收到的信号差别很大，这个时候就必须将声波视为球面波。因此在进行声源定位算法相关研究的时候，必须区分声音的传播模型，也就是近场模型和远场模型^[28]。对于一个均匀线阵来说，可以用公式（2.1）来区分近场和远场：

$$r < \frac{2d^2}{\lambda} \quad (2.1)$$

其中 r 为声源到阵列的距离， d 为阵列中阵元之间的距离， λ 为声源信号的波长，当声源到阵列的距离满足式（2.1）时为近场模型，否则为远场模型。人类在说话时，语音频率主要集中在 300Hz 到 3400Hz 之间^[29]，声波在常温空气中的传播速度约为 342m/s，易得，人声信号的波长通常在 0.1 米到 1.1 米之间。

2.2.1 远场模型

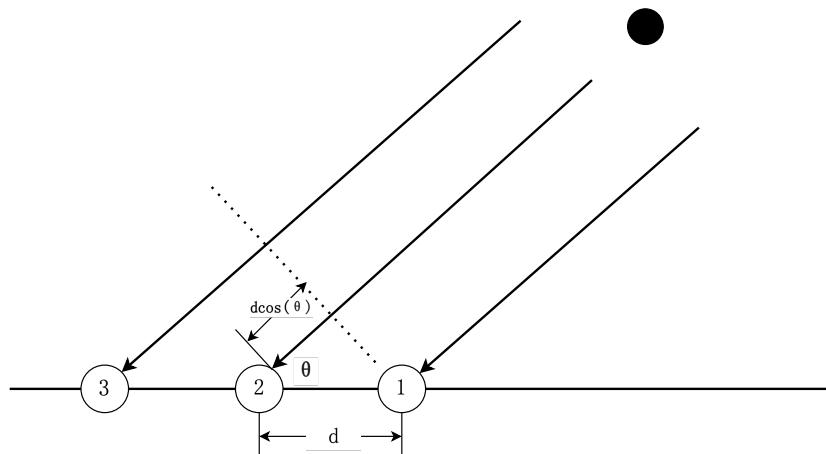


图 2.2 远场模型示意图

拿均匀线阵作为例子，当声源足够远（远场）时，传播模型如图 2.2 所示。在远场模型中，可以利用阵元接收到的音频信号实现对于声源到达方向的估计。声源发出的声波以近似平面波的方式到达不同阵元，方向与线阵成 θ 角。最右侧的麦克风的编号为 1，中间的编号为 2，麦克风 1 与麦克风 2 之间信号延迟差实际上就是声波走过 $d \cos(\theta)$ 的时间，所以信号到麦克风 1 和麦克风 2 的到达时间差可以表示为：

$$\tau_{12} = \frac{d \cos(\theta)}{c} \quad (2.2)$$

对于这两个麦克风，假设已经知道声源位于线阵的具体侧时，由式 (2.2) 可得，声源角度为：

$$\theta = \arccos\left(\frac{\tau_{12} \cdot c}{d}\right) \quad (2.3)$$

2.2.1 近场模型

本部分同样以均匀线阵作为例子，当声源位于近场时声音传播模型如图 2.3 所示：

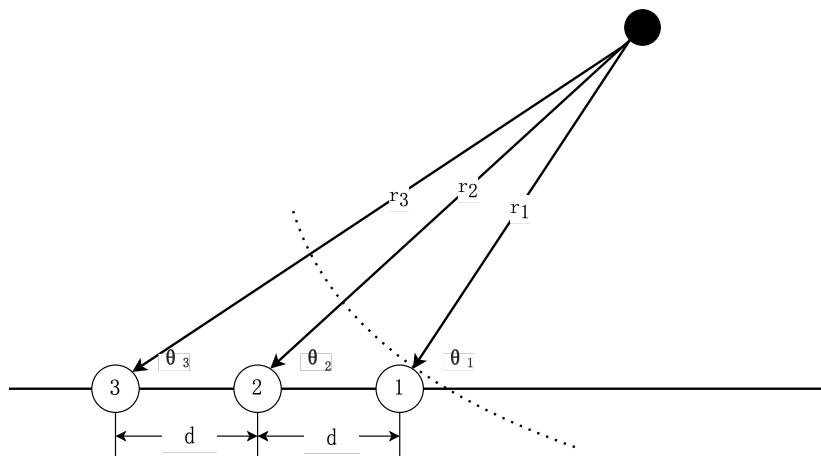


图 2.3 近场模型示意图

在近场模型下，可以声源到达方向的估计，同时由于声波的扩散方式近似与一个圆形，推导的时候可以借助 r_1 来计算出声源到线阵的距离。在本次计算中把麦克风 1 作为参考点，则可以得到对应的到达时间差：

$$\tau_{12} = \frac{r_2 - r_1}{c} \quad (2.4)$$

$$\tau_{13} = \frac{r_3 - r_1}{c} \quad (2.5)$$

经过简单的三角函数计算可得：

$$r_2 = \sqrt{r_1^2 + 2r_1 d \cos(\theta_1) + d^2} \quad (2.6)$$

$$r_3 = \sqrt{r_1^2 + 4r_1 d \cos(\theta_1) + 4d^2} \quad (2.7)$$

在通过时延估计算法得到 τ_{12} 和 τ_{13} 之后，就可以通过式 (2.4) 到式 (2.7) 得到声源的相关信息。

2.3 现有麦克风阵列声源测向方法

2.3.1 基于子空间类的方法

基于子空间类的声源测向方法有很多，这里以经典的 MUSIC (MULTiple SIgnal Classification) 算法为例进行详细介绍。

MUSIC 算法的原理对麦克风接受信号的协方差矩阵进行特征分解，从而得到噪声的子空间和信号的子空间，之后利用两个子空间的正交性来构建空间谱函数，并通过搜索空间谱函数的能量谱峰来实现对于声源的测向。

假设麦克风阵列的阵元数量为 M_s ，则其接收到的信号可以表示为：

$$\begin{bmatrix} x_1(t) \\ x_2(t) \\ \vdots \\ x_{M_s}(t) \end{bmatrix} = \begin{bmatrix} e^{-j\omega_0\tau_{11}} & e^{-j\omega_0\tau_{12}} & \dots & e^{-j\omega_0\tau_{1N_s}} \\ e^{-j\omega_0\tau_{21}} & e^{-j\omega_0\tau_{22}} & \dots & e^{-j\omega_0\tau_{2N_s}} \\ \vdots & \vdots & \ddots & \vdots \\ e^{-j\omega_0\tau_{M_s1}} & e^{-j\omega_0\tau_{M_s2}} & \dots & e^{-j\omega_0\tau_{M_sN_s}} \end{bmatrix} \begin{bmatrix} s_1(t) \\ s_2(t) \\ \vdots \\ s_{N_s}(t) \end{bmatrix} + \begin{bmatrix} n_1(t) \\ n_2(t) \\ \vdots \\ n_{M_s}(t) \end{bmatrix} \quad (2.8)$$

简化一下就可以得到式 (2.9)：

$$\mathbf{X}(t) = \mathbf{A}\mathbf{S}(t) + \mathbf{N}(t) \quad (2.9)$$

其中 $\mathbf{X}(t)$ 为麦克风阵列接收到信号的矩阵， \mathbf{A} 为阵列的导向矢量， $\mathbf{S}(t)$ 为原始声源信号的矢量， $\mathbf{N}(t)$ 为环境噪声矢量，由式 (2.9) 可以得到信号的协方差矩阵为：

$$\mathbf{R} = E(\mathbf{X}(t)\mathbf{X}^H(t)) \quad (2.10)$$

假设信号与噪声不相关，将式 (2.10) 代入式 (2.9) 得：

$$\mathbf{R}_{XN} = \mathbf{A}E(\mathbf{S}\mathbf{S}^H)\mathbf{A}^H + \sigma_N^2 \mathbf{I} = \mathbf{A}\mathbf{R}_S\mathbf{A}^H + \sigma_N^2 \mathbf{I} \quad (2.11)$$

其中， $\mathbf{A}\mathbf{R}_S\mathbf{A}^H$ 为信号的子空间， $\sigma_N^2 \mathbf{I}$ 噪声的子空间， \mathbf{R}_S 为信号的协方差矩阵， σ_N^2 为噪声的功率， \mathbf{I} 为 $M_s \times M_s$ 的单位矩阵。信号矩阵 \mathbf{R}_{XN} 在特征值分解之后会得到 M_s 个特征向量，将这些特征向量的特征值从小到大进行排序，前 N 个特征值的对应的特征向量组成 \mathbf{V}_S ，剩下的特征向量（即噪声）组成 \mathbf{V}_N ，得：

$$\mathbf{R}_{XX} = \mathbf{U}_S \mathbf{V}_S \mathbf{U}_S^H + \mathbf{U}_N \mathbf{V}_N \mathbf{U}_N^H \quad (2.12)$$

其中 \mathbf{U}_S 为信号矩阵， \mathbf{U}_N 为噪声矩阵。理想情况下，噪声的特征向量和声源信号的特

征向量相互正交，即 $\mathbf{V}_S \mathbf{V}_N = \mathbf{0}$ 。但实际情况下有所不同，为了得到估计结果，需要对信号进行最大似然估计：

$$\widehat{\mathbf{R}} = \frac{1}{L} \sum_{i=1}^L \mathbf{X}_i \mathbf{X}_i^H \quad (2.13)$$

其中 L 为信号采样的样本数，在对式 (2.13) 作特征值分解时，由于噪声信号的存在，式 (2.13) 无法严格推导，就需要基于式 (2.14) 来获得声源的方位角：

$$\theta = \arg_{\theta} \min a^H(\theta) \widehat{\mathbf{U}}_N \widehat{\mathbf{U}}_N^H a(\theta) \quad (2.14)$$

因为 $a(\theta)$ 与 U_N 列正交，阵列信号的功率谱就会在相关处出现一个峰值，峰值位置对应的方位角就是声源相对于阵列的方向，MUSIC 算法计算出的功率谱可以表示为：

$$P = \frac{1}{a^H(\theta) \widehat{\mathbf{U}}_N \widehat{\mathbf{U}}_N^H a(\theta)} \quad (2.15)$$

2.3.2 基于波束形成的方法

最小均方无畸变响应 MVDR (Minimum Variance Distortionless Response) 是波束形成中最为经典的算法之一，该算法基于最小均方误差的标准，在保持所需检测方向增益不变的情况下，将波束形成的输出总功率降到最低，从而抑制干扰和噪音信号，并恢复出需要检测的成分。

在 MVDR 中，每个阵元接收到的信号可以写作：

$$\mathbf{x}_m(t) = \mathbf{a}_m(\theta_d) s(t) + \sum_{j=1}^J \mathbf{a}_m(\theta_{ij}) i_j(t) + \mathbf{n}_m(t) \quad (2.16)$$

其中 $\mathbf{a}_m(\theta_d)$ 为麦克风阵列在 θ_d 方向的响应， $s(t)$ 为声源发送的信号， $\mathbf{a}_m(\theta_{ij}) i_j(t)$ 为来自其他方向信号的干扰， $\mathbf{n}_m(t)$ 为阵列收到的噪声信号。假设接收到波束的赋形向量为 ω ，则这个方向的波束形成的平均功率为：

$$P(\omega) = \frac{1}{N} \sum_{t=1}^N |\mathbf{y}(t)|^2 = \frac{1}{N} \sum_{t=1}^N |\mathbf{w}^H \mathbf{x}(t)|^2 \quad (2.17)$$

当 N 足够大时，式 (2.17) 可以写作式 (2.18) 的形式：

$$P(\omega) = E[|\mathbf{y}(t)|^2] = \mathbf{w}^H E[\mathbf{x}(t) \mathbf{x}^H(t)] \mathbf{w} = \mathbf{w}^H \mathbf{R} \mathbf{w} \quad (2.18)$$

其中 $\mathbf{R} = E[\mathbf{x}(t) \mathbf{x}^H(t)]$ 就是阵列接收信号的协方差矩阵，式 (2.18) 的结果可以等效于：

$$\mathbf{w}^H \mathbf{R} \mathbf{w} = \mathbf{w}^H \left(\sigma_d^2 \mathbf{a}(\theta_d) \mathbf{a}(\theta_d)^H + \sum_{j=1}^{M-1} \sigma_j^2 \mathbf{a}(\theta_j) \mathbf{a}(\theta_j)^H + \sigma_n^2 \mathbf{n}_m(t) \right) \mathbf{w} \quad (2.19)$$

为了保证信号结果被完整保留，减小噪声和干扰噪声的影响，所以希望整体的平均功率可以尽量小。但从频域的角度看，要保证信号无失真输出，要保持 $\mathbf{w}^H \mathbf{a}(\theta_d) = 1$ 。由于零均值信号的方差等于平均功率，也就是说将平均功率降到最低就是等价于将方差降到最低，这也就是 MVDR 名称的由来。

最小化平均功率并保持无失真输出的方式是利用拉格朗日乘子法，将问题引入拉格朗日函数，得：

$$L(\mathbf{w}, \lambda) = \mathbf{w}^H \mathbf{R} \mathbf{w} + \lambda (\mathbf{w}^H \mathbf{a}(\theta_d) - 1) \quad (2.20)$$

令 \mathbf{w} 的偏导为零，代入回 $\mathbf{w}^H \mathbf{a}(\theta_d) = 1$ 得最优的最小平均功率，即得到对应角度下的加权向量计算式：

$$\mathbf{w}(\theta_d) = \frac{\mathbf{R}^{-1} \mathbf{a}(\theta_d)}{\mathbf{a}^H(\theta_d) \mathbf{R}^{-1} \mathbf{a}(\theta_d)} \quad (2.21)$$

有了式 (2.21) 的加权向量式，得式 (2.22) 的功率谱式，就通过遍历所有的 θ_d 来找到功率最强的方向，也就是声源方向。

$$P(\theta_d) = \frac{1}{\mathbf{a}^H(\theta_d) \mathbf{R}^{-1} \mathbf{a}(\theta_d)} \quad (2.22)$$

2.3.3 基于时延估计的方法

基于时延估计的声源测向方法又称到达时间定位 TDOA (Time Difference of Arrival)，顾名思义，就是利用声音信号到达阵列中不同阵元的时间差来确定声源的位置，本方法为两步方法，即时延估计和几何解算，具体实现的方法有很多，但是大多实现都很简洁，在这里不介绍具体的实现，而是概括一下整个 TDOA 方法的原理。

假设声音信号到达阵元的时间差为 Δt ，乘上声音传播的速度 c ，就可以求得声音到达不同阵元之间的路程差 Δd 。给定两个阵元，如果知道声音到这两个阵元之间的路程差 Δd ，就可以通过几何关系推算出声源的位置。因为本系统主要面向的是嵌入式场景，阵列尺寸较小，满足式 (2.1) 中对远场模型的要求，并且本文设计的是测向系统，所以之后的测向估计都将基于平面阵和远场模型进行介绍。

假设声源单位平面波的传播向量为 $\vec{u} = (u, v)$ ，阵列中有 N 个麦克风 ($N \geq 3$)，则所有麦克风之间两两组合共有 $N(N - 1)/2$ 对，记第 k 个麦克风的坐标为 (x_k, y_k) ，则可以得到麦克风 k 和 j 之间的延时 τ_{kj} ：

$$c\tau_{kj} = -(\mathbf{k} - \mathbf{j})\mathbf{u} \quad (2.23)$$

转换成标量形式如下：

$$c\tau_{kj} = u(x_k - x_j) + v(y_k - y_j) \quad (2.24)$$

当麦克风阵列中有多个麦克风时，每两个阵元就可以得到一组式（2.24）， N 个麦克风就会有 $N(N - 1)/2$ 个等式，声源单位传播向量 $\vec{u} = (u, v)$ 有两个未知数，因此最少需要两组等式就可以计算出声源方向，假设系统有三个麦克风，即 $N = 3$ ，可以得到以下等式：

$$\begin{aligned} c\tau_{21} &= u(x_2 - x_1) + v(y_2 - y_1) \\ c\tau_{31} &= u(x_3 - x_1) + v(y_3 - y_1) \\ c\tau_{23} &= u(x_2 - x_3) + v(y_2 - y_3) \end{aligned} \quad (2.25)$$

转换为矩阵形式：

$$\begin{bmatrix} (x_2 - x_1) & (y_2 - y_1) \\ (x_3 - x_1) & (y_3 - y_1) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} c\tau_{21} \\ c\tau_{31} \end{bmatrix} \quad (2.26)$$

假设阵列中心为坐标原点，记最左侧由阵元坐标组成的矩阵为矩阵 \mathbf{A} ，声源坐标组成的矩阵为矩阵 \mathbf{x} ，右侧由时延和声速组成的矩阵为矩阵 \mathbf{b} ，有：

$$\hat{\mathbf{x}} = \mathbf{A}^\dagger \mathbf{b} = (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T \mathbf{b} \quad (2.27)$$

记矩阵 \mathbf{A} 的广义逆矩阵 \mathbf{A}^\dagger 为 $\mathbf{A}^\dagger = [\mathbf{c}_1, \mathbf{c}_2]^T$ ， \mathbf{c}_1 和 \mathbf{c}_2 均为矩阵 \mathbf{A} 的列向量，带入式(2.32)可求出 $\mathbf{u} = (u, v)$ ，之后就可以得到声源的最小二乘方位角估计 φ ：

$$\varphi = \arctan \left(\frac{\mathbf{c}_1^T \mathbf{b}}{\mathbf{c}_2^T \mathbf{b}} \right) \quad (2.28)$$

其中 \arctan 为 Matlab 中的 atan2 函数实现，函数输入为两个变量，利用两个变量的正负号来实现 $[-\pi, \pi]$ 的输出。

2.4 常用的时延估计方法

因为 2.3.1 节提到的子空间类的声源测向方法和 2.3.2 节提到的基于波束形成的声源测向方法都更适用于麦克风数量较多并且计算资源较为丰富的系统，和本文研究的嵌入式场景相背，基于时延估计的声源测向方法更适合本文的研究。接下来的介绍将会分为两个部分，对时延估计中常见的基本互相关方法和广义互相关进行介绍。

2.4.1 基本互相关

基本互相关法通过对不同麦克风阵元接收的信号做互相关运算，通过运算的结果来

判断两个信号之间的延迟。假设有两个麦克风 $x_1(t)$ 和 $x_2(t)$ ，都在捕获来自声源的信号 $s(t)$ ，声音在传播的过程被附加上了噪音信号 $n_1(t)$ 和 $n_2(t)$ ，这两个噪音信号与声源和彼此之间都是互不相关的，两个麦克风之间的信号延迟表示为 τ_0 。则两个麦克风信号的时域方程可以表达为式（2.29），其中 a_0 为常数：

$$\begin{aligned} x_1(t) &= s(t) + n_1(t) \\ x_2(t) &= a_0 s(t - \tau_0) + n_2(t) \end{aligned} \quad (2.29)$$

之后将两个处于时域的信号转换到频域，可得：

$$\begin{aligned} \mathbf{X}_1(\omega) &= \mathbf{S}(\omega) + \mathbf{N}_1(\omega) \\ \mathbf{X}_2(\omega) &= a_0 \mathbf{S}(\omega) e^{-j\omega\tau_0} + \mathbf{N}_2(\omega) \end{aligned} \quad (2.30)$$

对得到的两个麦克风信号的频域结果做互相关运算得：

$$\begin{aligned} \mathbf{R}_{x_1, x_2}(\tau) &= E[x_1(t)x_2(t - \tau)] \\ \mathbf{R}_{x_1, x_2}(\omega) &= \mathbf{X}_1(\omega)\mathbf{X}_2^*(\omega) \\ &= |\mathbf{S}(\omega)|^2 a_0 e^{j\omega\tau_0} \end{aligned} \quad (2.31)$$

其中 E 表示求两个信号相关结果的期望。因为声源信号 $s(t)$ 和 噪音信号 $n_1(t)$ 和 $n_2(t)$ 都不相关，所以噪音项 $\mathbf{N}_1(\omega)$ 和 $\mathbf{N}_2(\omega)$ 都被消除。为了进一步知道信号延迟 τ_0 的值，需要求取两个麦克风信号的互相关功率谱，即：

$$\tilde{\mathbf{R}}_{x_1, x_2}(l) = \frac{1}{2\pi} \int_{-\pi}^{\pi} \mathbf{R}_{x_1, x_2}(\omega) e^{j\omega\tau} d\omega \quad (2.32)$$

通过寻找互功率谱的峰值位置，就可以得到信号在阵元之间传播的时间差 τ 。

2.4.2 广义互相关

但是在实际的应用场景下，麦克风收音不仅会受到噪声的影响，空间混响也是不可忽视的一部份，甚至麦克风收到的噪声也是部分相关的，这就会导致基本互相关的结果中出现大量的干扰旁瓣，特别是当信噪比较低时，想要得出正确的结果就会很困难。为了提高时延估计整体的精度，就需要对式（2.31）中的互相关频域结果做进一步的处理，以达到凸显峰值的效果，这就是广义互相关法的原理。

由式（2.32）的互相关功率谱得，为其加上一个变换权重，就可以得到对应的广义互相关函数：

$$\tilde{\mathbf{R}}_{x_1, x_2}(l) = \frac{1}{2\pi} \int_{-\pi}^{\pi} \psi_{12}(\omega) \mathbf{R}_{x_1, x_2}(\omega) e^{j\omega\tau} d\omega \quad (2.33)$$

而加权函数 $\psi_{12}(\omega)$ 对广义互相关的结果影响很大，怎么去选择合适的加权函数就变得十分重要，常见的加权函数由 SCOT、Roth、PHAT 等，如表 2.2 所示：

表 2.2 常见的广义互相关加权函数

名称	$\psi_{12}(\omega)$
CC	1
SCOT	$\frac{1}{\sqrt{\mathbf{R}_{x_1,x_1}(\omega)\mathbf{R}_{x_2,x_2}(\omega)}}$
Roth	$\frac{1}{\mathbf{R}_{x_1,x_2}(\omega)}$
PHAT	$\frac{1}{ \mathbf{R}_{x_1,x_2}(\omega) }$
HB	$\frac{ \mathbf{R}_{x_1,x_2}(\omega) }{\mathbf{R}_{x_1,x_1}(\omega)\mathbf{R}_{x_2,x_2}(\omega)}$
ML/HL	$\gamma(\omega) = \frac{\frac{ \gamma(\omega) ^2}{ \mathbf{R}_{x_1,x_2}(\omega) (1 - \gamma(\omega) ^2)}}{\frac{\mathbf{R}_{x_1,x_2}(\omega)}{\sqrt{\mathbf{R}_{x_1,x_1}(\omega)\mathbf{R}_{x_2,x_2}(\omega)}}}$

2.5 本章小结

本章主要从麦克风阵列的设计、声音的传播模型和声源测向算法三个方面展开了介绍。首先介绍了麦克风阵列的几何模型，指出不同类型的阵列使用的场景，平面阵和立体阵的区别，并且分析了声音波长和麦克风阵列最大尺寸之间的关系。之后分析了声源特性，对声音的近场模型和远场模型进行了介绍。最后阐述了时延估计算法的推导，详细介绍了广义互相关法的实现细节。最后得出结论，对于麦克风阵列来说，平面阵对于本系统的测向应用来说完全足够；对于声源测向算法来说，时延估计算法的计算量小，实时性好，符合 FPGA 声源测向系统的要求。

3 声源测向系统硬件设计

基于 FPGA 和 MEMS 麦克风的硬件设计是整套声源测向系统的核心部分。本章在 FPGA 开发平台的基础上，选用数字式 MEMS 麦克风搭配平面正十字阵进行声源测向，后续内容详细展开了系统的硬件规格与各项细节。

3.1 总体设计方案

本系统采用了 Xilinx 公司 Zynq 系列 FPGA 开发版，搭建了一套四麦克风阵列声源测向系统。本系统使用的 FPGA 有 85k 个逻辑单元、220 个乘法器以及 4.9Mbit 的片上 RAM，并且集成有一颗双核 ARM Cortex-A9 处理器；阵列的麦克风是在手机、笔记本电脑和无线耳机等随身设备中常见的全指向麦克风。

系统总体的硬件框图如图 3.1 所示，主要包括 PL（Programmable Logic）、PS（Processing System）和麦克风阵列三个模块，如图 3.1 所示。

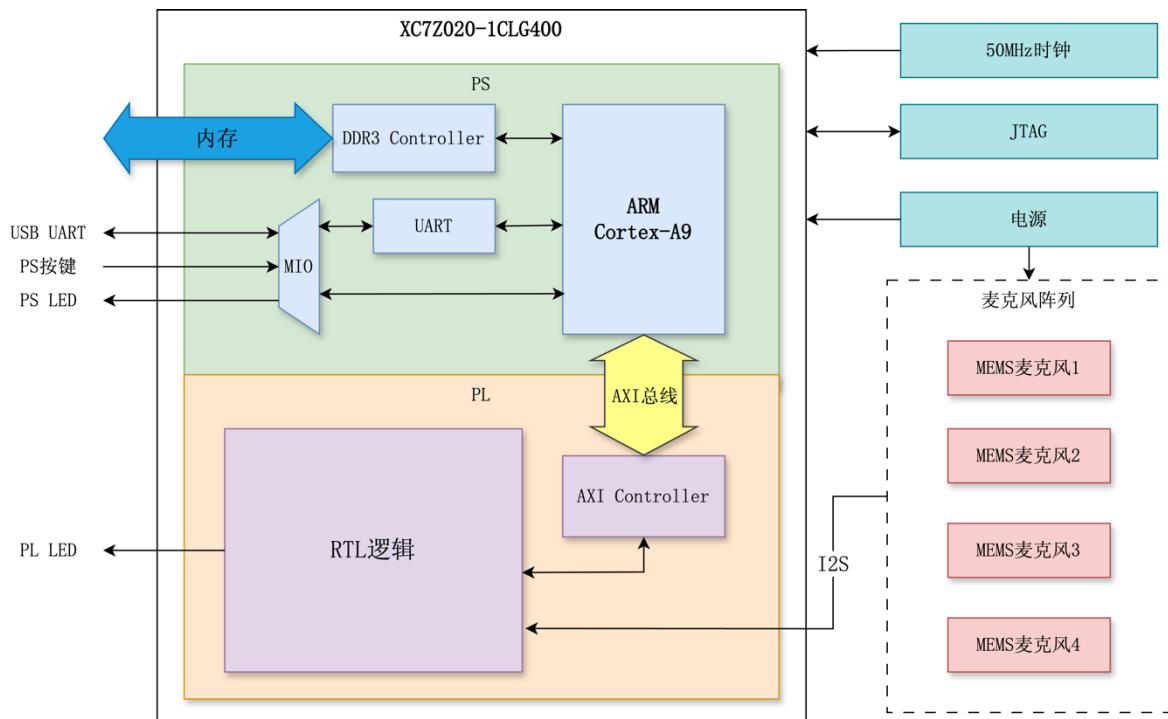


图 3.1 硬件系统框图

系统通过四个 MEMS 麦克风接收外界的声音信号，通过 I2S 协议将音频数据并行送入 PL 部分，PL 部分完成音频的处理后进行时延估计计算，时延估计完成后通过 AXI 总线把互功率谱结果发送给 PS，由 PS 端完成测向估计，并将计算得出的方位角通过 USB 串口发送到上位机，上位机通过串口终端对结果进行显示。

本文设计的声源测向系统的具体参数如下：

- (1) 通道数：4 通道；
- (2) 采样频率区间：300Hz 到 3000Hz；
- (3) 音频采样频率：48.8kHz；
- (4) 采样窗口长度：20ms；
- (5) 每秒可处理音频帧数：> 40；
- (6) 整体功耗：< 3W；
- (7) 麦克风阵列：平面十字阵；
- (8) 测向算法：TDOA

3.2 麦克风阵列设计

3.2.1 麦克风选型

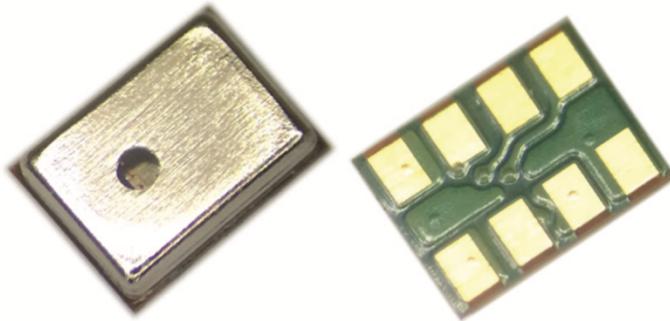


图 3.2 MSM26S4030H0 数字 MEMS 麦克风

通过 2.2.1 节和 3.1 节的介绍可知，MEMS 麦克风比较符合本系统面向的嵌入式场景。因此，本文选用了苏州敏芯微公司的 MSM26S4030H0 数字 MEMS 麦克风，如图 3.2 所示。该 MEMS 麦克风具有成本低、可选低功耗模式、数字 I2S 输出和高灵敏度的特点。

其他的关键参数如下：

- (1) 灵敏度： $-26\text{dB} \pm 2\text{dB}$ ；
- (2) 工作电压：1.6V 到 3.6V；
- (3) 消耗电流典型值：750 μA ；
- (4) 频率响应：见图 3.3；
- (5) 总谐波失真：1%（最大声压级为 100dB 时）；
- (6) 信噪比：57dB（20kHz 带宽，A 加权）；

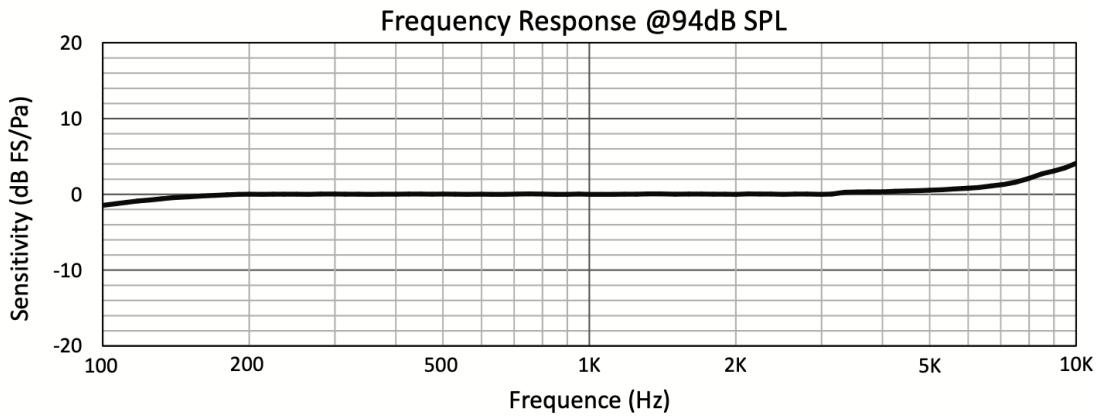


图 3.3 MSM26S4030H0 频率响应图

3.2.2 阵列结构设计

对于一套声源测向的应用中，应用最多且效果最好的的就是平面圆阵，因为在一定数量的麦克风下，平面圆阵能保证从各个方向到达的声波都能被尽可能平等地处理，这种对称性确保了声源定位的一致性和可靠性，不会因为声源的方位不同而有所偏差，并且这种对称性还有额外的优势，可以更容易地实现基于相位差或时间差的定位算法。本系统出于 FPGA 资源量的考量，选择了平面圆阵中的一种简单变体，即平面正十字阵，正十字阵的阵元最大间距为 18cm，具体结构如图 3.4 所示。

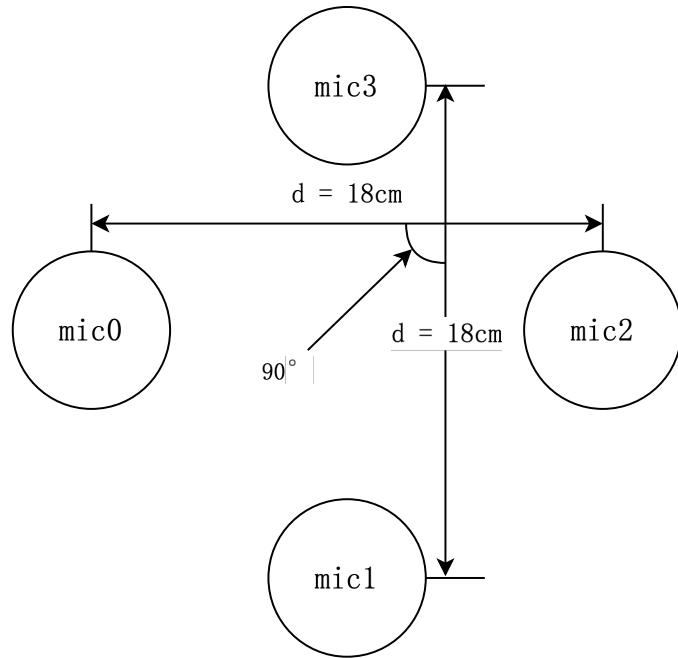


图 3.4 正十字阵结构示意图

3.2.2 阵列电路设计

本部分主要介绍了麦克风阵列的电路设计，根据图 3.5 的参考设计推荐，本系统

将所有麦克风都设置到 I2S 的左声道输出, 不启用立体声格式, 以保证阵列的同步采样, 这样总计就会有四个引脚向 FPGA 输出音频信息。电源输出选择直接与板载的 3.3V 引脚连接, 因为四个麦克风的工作电流很小, 所以可以满足麦克风阵列的供电需求。

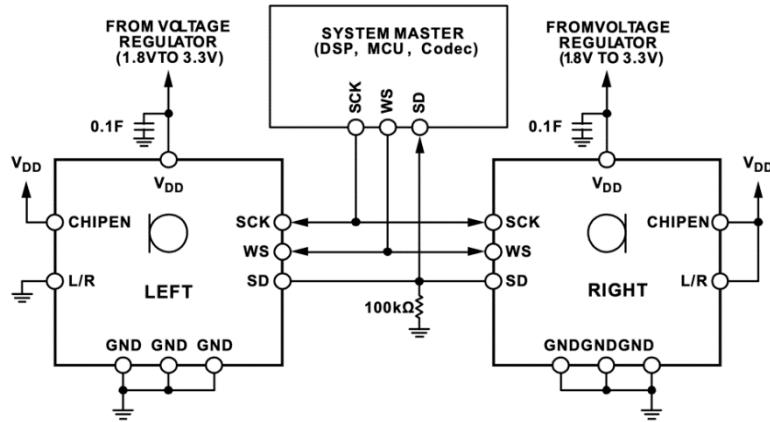


图 3.5 MSM26S4030H0 参考设计图

3.3 FPGA 平台选择

3.3.1 主控芯片的选择

目前市场上主流的 FPGA 厂商包括 Xilinx、Altera、高云和紫光同创等厂商。单论市场占用率和产品丰富度来说, Xilinx 是市场中的主流选择。因为本系统在算法部署时会需要用到一定量的乘法器, 工程较为复杂, 出于成本和性能的考虑, 选择了 Xilinx 公司的 Zynq-7000 系列 SoC 集成 FPGA。本系列属于成品系列中成本优化的可拓展 SoC 集成 FPGA 平台, 该系列集成了单核或双核的 Arm Cortex-A9, 28nm 的 7 系列可编程逻辑, 以及速率高达 12.5Gbps 的收发器。

FPGA 作为一种可高度自定义电路逻辑的器件, 在高速并行数据处理上有着得天独厚的优势。由于 FPGA 使用的是最直接的硬件数字逻辑, 和串行执行程序指令的 CPU 相比运行效率要高很多, 比如现在需要完成三个数据的乘加运算, 在 FPGA 上可以通过搭配加法器和乘法器, 只需要一个时钟周期, 就可以计算得出结果。对于 CPU 来说, 就需要把这个过程分解为乘法和加法, 无论如何都会需要两个时钟周期。同时, 基于 Verilog 的模块化特性, 使用者可以在一块 FPGA 芯片上实现大量相同的功能。

然而, 纯 FPGA 系统也有着自己的劣势, 最受人诟病的就是 FPGA 设计通常缺乏灵活性, 当应用中需要灵活的交互或者复杂的 TCP/IP 通信时, 纯 FPGA 的设计就会变得极为复杂, 或者需要消耗大量的资源。而 ARM 处理器, 特别是运行了实时操作系统 RTOS 的平台, 在图形界面显示和网络传输方面就会特别方便。比如在示波器制作一个逻辑分析仪, 可以利用 FPGA 实现多路高速数据采集, 再通过 DMA 发送给 ARM 端, ARM

端利用 LVGL 或者 QT 完成显示和交互，整体的代码量和性能都会比纯 ARM 或者纯 FPGA 平台好上不少。

可以看出，相较于传统的单 ARM 处理器或者纯 FPGA 的嵌入式芯片，Zynq-7000 SoC FPGA 既有 ARM 处理器灵活高速的数据计算和高层级的 C 语言编程平台，又拥有 FPGA 的高速并行数据处理能力。同时，Xilinx 提供了专有的互联架构，让用户可以将 FPGA 上的通过可编程逻辑通过封装，通过 AXI 总线暴露给 ARM 处理器，以实现灵活的系统设计。

3.3.2 开发平台功能介绍

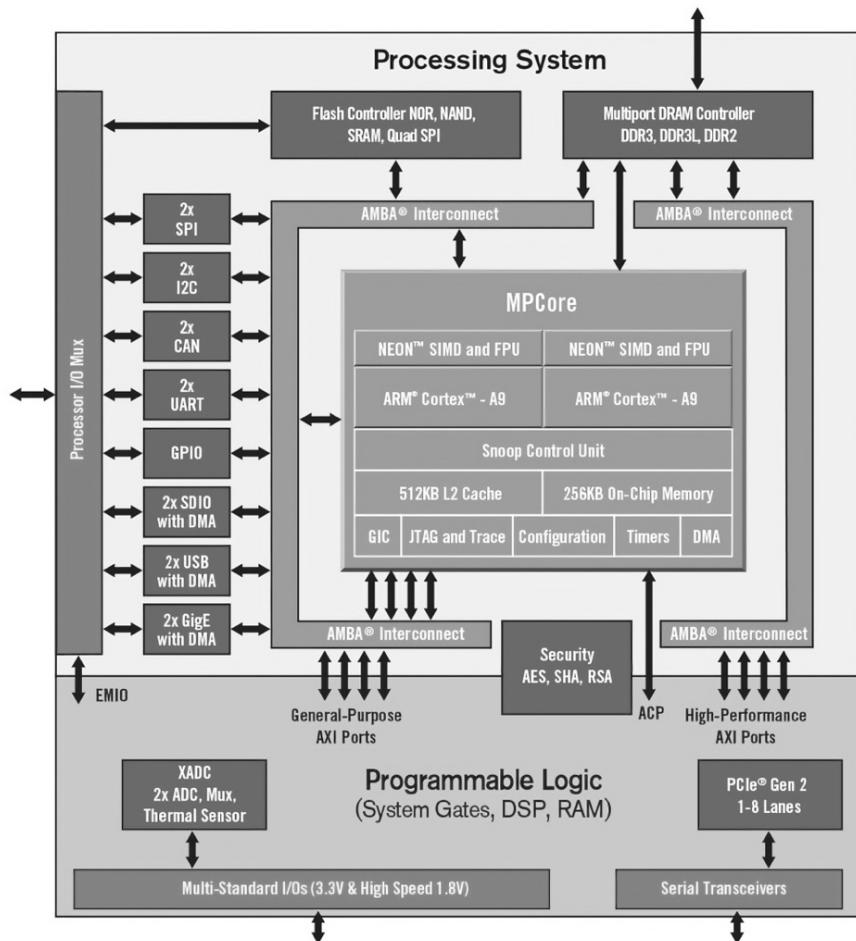


图 3.6 Zynq 7000 SoC 架构图

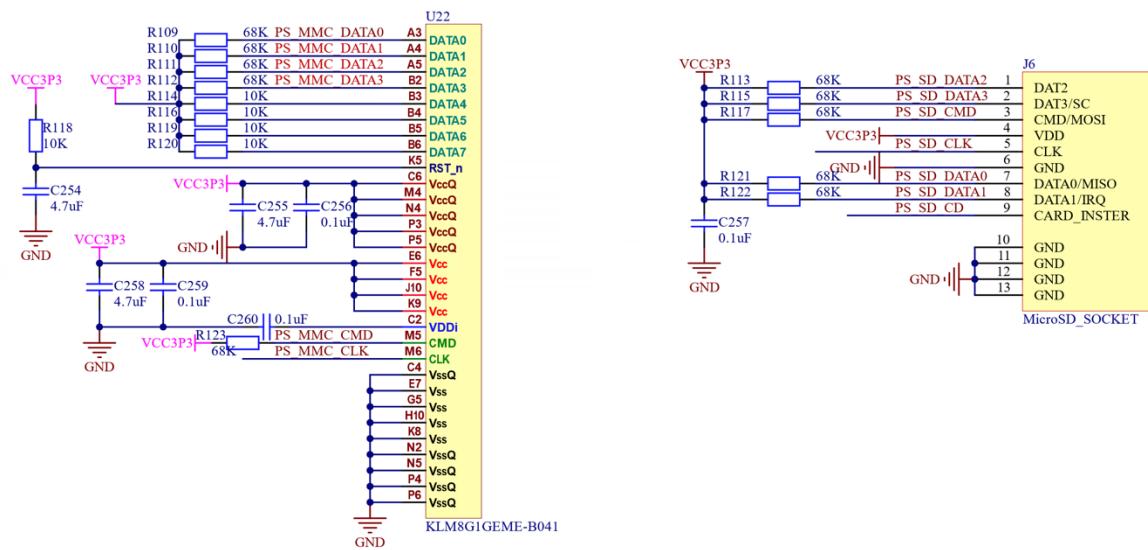
本系统基于 Xilinx Zynq 7000 系列中的入门级别器件 XC7Z020-1CLG400，该器件具有如下资源：

- (1) 85k 的逻辑资源；
- (2) 4.9Mb 的片上 RAM；
- (3) 220 个乘法器单元；

- (4) 4 个模拟锁相环 PLL;
 - (5) 4 个数字锁相环 MMCM;
 - (6) 双核 Cortex-A9 处理器，最高主频为 866MHz;
 - (7) 1 个 DDR3 控制器;
 - (8) 2 个 NAND/QSPI FLASH 控制器;
 - (9) 2 个 SD/SDIO 控制器;
 - (10) 2 个 SPI 主/从机控制器;
 - (11) 2 个 I2C 控制器;
 - (12) 2 个 UART 控制器;

其中(1)到(4)为PL(Programmable Logic)侧的资源,(5)到(12)为PS(Processing System)侧的资源,整体系统框图如图3.6所示。

FPGA 开发平台有三种程序的启动方式, 即 FLASH、SDIO 和 JTAG 模式。从 FLASH 启动时, 系统先初始化 FLASH 控制器, 之后从 FLASH 中拷贝比特流文件到 FPGA, 再将 PS 侧的应用代码拷贝至 DDR3 中, 最后跳转到代码执行; SDIO 整体启动流程和 FLASH 相比大同小异, 只是存储介质从 NAND FLASH 变为了 SD 卡; JTAG 模式则是直接通过 JTAG 接口将对应的代码烧录到 FPGA 和 PS DDR, 但系统掉电之后程序会丢失。FPGA 与 FLASH 和 SD 卡的连接如图 3.7 所示:



3.7 FLASH 和 SDIO 与 FPGA 的连接图

Zynq 平台在启动时先运行的是 BootROM 代码，类似于现代计算系统中的 BIOS，之后是 FSBL 系统代码和用户侧程序，系统的启动方式有五种，分别是 JTAG、NOR FLASH、NAND、QSPI FLASH 和 SD 卡，通过 MIO 引脚[5:2]来进行模式的选择，如图 3.8 所示：

Pin-Signal / Mode	MIO[8]	MIO[7]	MIO[6]	MIO[5]	MIO[4]	MIO[3]	MIO[2]
	V MODE[1]	V MODE[0]	BOOT_MODE[4]	BOOT_MODE[0]	BOOT_MODE[2]	BOOT_MODE[1]	BOOT_MODE[3]
Boot Devices							
JTAG Boot Mode; cascaded is most common ⁽¹⁾	0	0	0				
NOR Boot ⁽³⁾	0	0	1				
NAND	0	1	0				
Quad-SPI ⁽³⁾	1	0	0				
SD Card	1	1	0				
Mode for all 3 PLLs							
PLL Enabled		0	Hardware waits for PLL to lock, then executes BootROM.				
PLL Bypassed		1	Allows for a wide PS_CLK frequency range.				
MIO Bank Voltage⁽⁴⁾							
	Bank 1	Bank 0	Voltage Bank 0 includes MIO pins 0 thru 15. Voltage Bank 1 includes MIO pins 16 thru 53.				
2.5 V, 3.3 V	0	0					
1.8 V	1	1					

Notes:

1. JTAG cascaded mode is most common and is the assumed mode in all the references to JTAG mode except where noted.
2. For secure mode, JTAG is not enabled and MIO[2] is ignored.
3. The Quad-SPI and NOR boot modes support execute-in-place (this support is always non-secure)
4. Voltage Banks 0 and 1 must be set to the same value when an interface spans across these voltage banks. Examples include NOR, 16-bit NAND, and a wide TPIU test port. Other interface configuration may also span the two banks.

图 3.8 Zynq 系统模式引脚与启动模式的关系

在本系统中，系统与外界的交互基于串口，包括结果的上传与配置的进行，理论上任何支持解析串口输入输出的设备都可以搭配本系统使用。

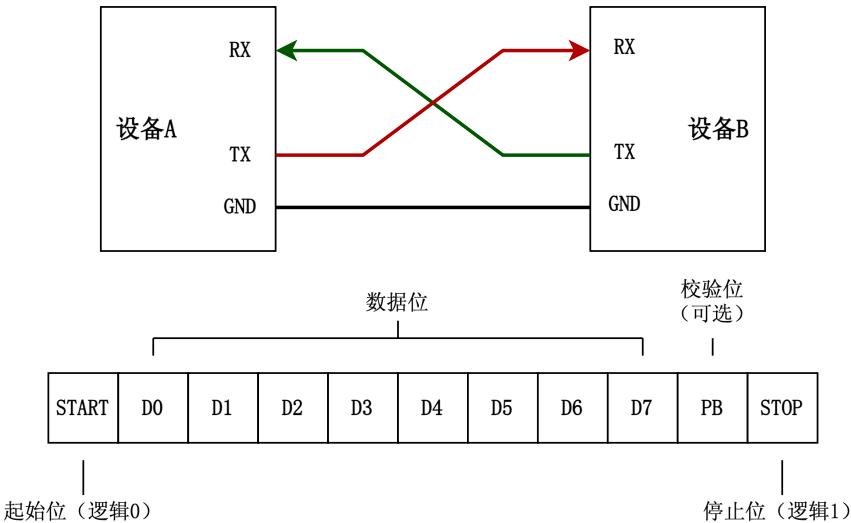
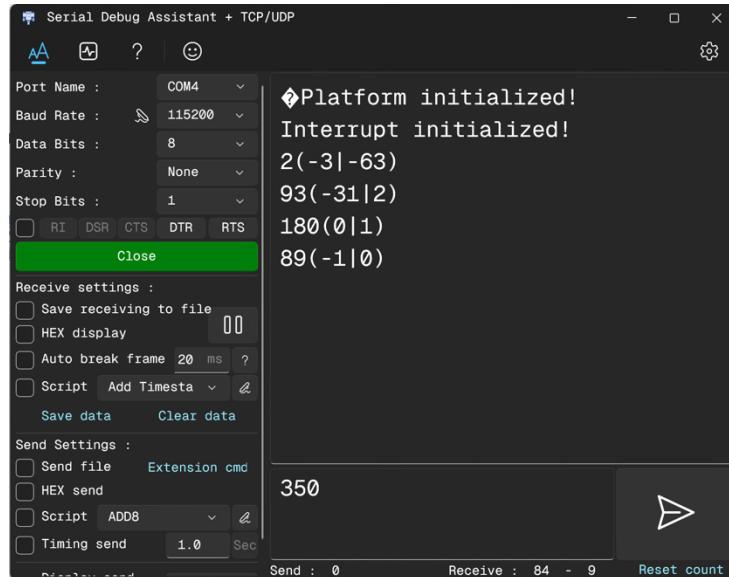


图 3.9 UART 传输协议示意图

UART (Universal Asynchronous Receiver/Transmitter, 通用异步收发传输器) 是一种广泛使用的硬件通信协议，它被广泛应用于嵌入式设备的串行通信中。UART 协议主要用于微控制器与其他设备（如计算机、另一个微控制器、外围设备等）之间的通信。它是异步的，意味着在通信过程中不需要时钟信号来同步发送和接收的数据。对于本系统

的简单应用场景来说实现简单，应用广泛，也能满足传输数据的要求。

拿常用的 PC 端串口终端作为例子，可以打印系统串口的输出，整体运行效果如图 3.10 所示，启动会显示平台和终端完成初始化，之后打印声源的方位角。



3.10 上位机串口终端示意图

3.4 本章小结

本章主要从声源测向系统硬件设计的角度入手，首先介绍了系统的总体设计方案，之后再详细介绍了所用 MEMS 的麦克风选型和阵列的设计。之后对 Zynq SoC FPGA 开发平台做了介绍，详细列出了 SoC FPGA 平台的优势，并对重要的硬件电路模块做了介绍，最后引出了配套上位机，介绍了系统向外发送数据和上位机解析数据的方式。整体证明了系统在硬件上的可实现性。

4 声源测向系统软件设计

在第三章中，声源测向系统完成了对应的硬件设计，我们需要通过软件来控制不同模块之间的正常运行，从而实现声源测向的目标。本章将主要介绍声源测向系统各个软件模块，并展示他们的仿真结果，其中时延估计模块是整个设计的核心，也是 FPGA 声源测向系统实现的难点所在。

软件模块的整体框图如图 4.1 所示，所有 PL 内的模块连接都基于标准的 AXI4-Stream 协议，PL 和 PS 间通讯基于 AXI4 协议。

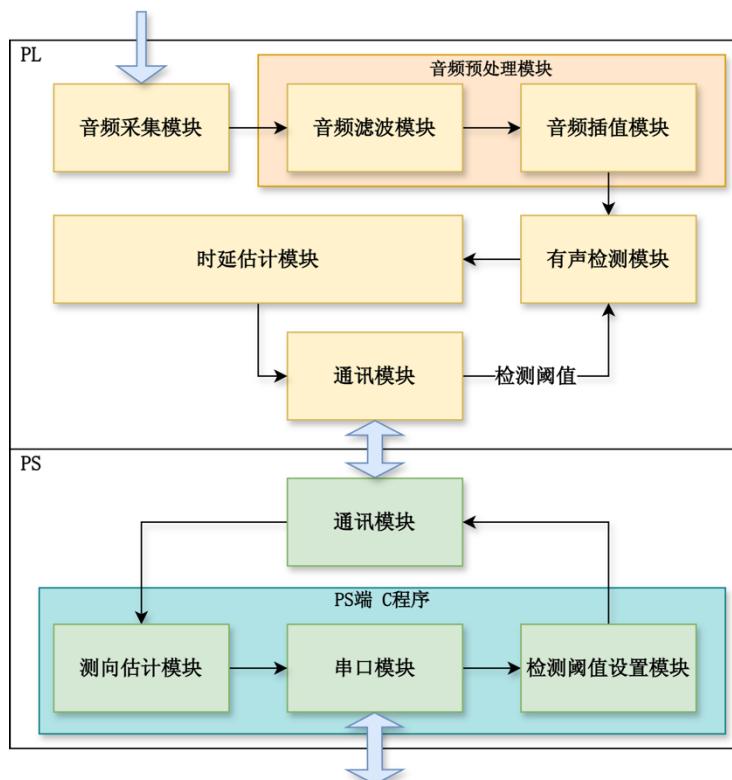


图 4.1 软件模块的整体框图

4.1 音频采集模块

音频采集模块主要实现了对于 MEMS 麦克风时钟信号的输出和音频信号的采集工作。在硬件系统的介绍中，可以知道本系统使用的 MEMS 麦克风时钟工作时钟频率典型值为 3.0MHz，为了设计的便利，本系统中选择将 50MHz 的系统时钟做 16 分频，输出 3.125MHz 的时钟给 MEMS 麦克风，此时麦克风的等效音频采样率可以用式（4.1）计算得，为 48.8kHz。

$$CLK_{sample} = \frac{CLK_{sys}}{64} \quad (4.1)$$

在音频数据的格式上，I2S 采用的是 32bit 有符号数输出，本麦克风支持 24bit 精度 MSB 输出，LSB 端将会自行补 1，具体时序如图 4.2 所示，为了节省后续需要使用的硬件资源，在这里将输出的音频数据截取高 16 位进行使用。

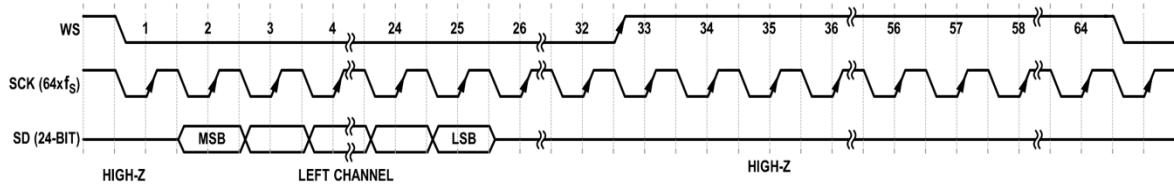


图 4.2 I2S 左采集时序图

采集模块的仿真结果如图 4.3 所示：

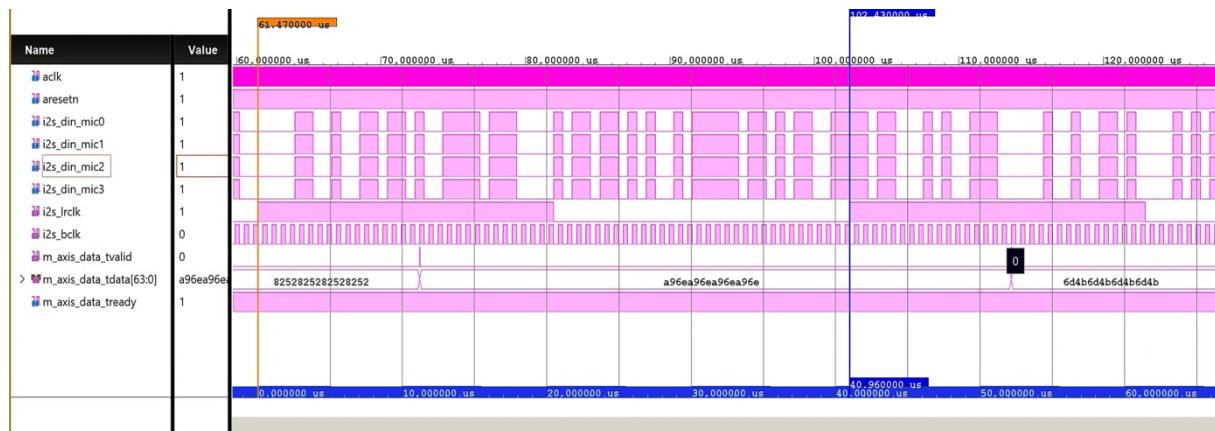


图 4.3 音频采集模块仿真结果

按照输入时钟 3.125MHz 来看，结合图 4.4 提供的时序，这个周期应该可以通过式 (4.2) 计算，正是仿真图中显示的 40.96us。

$$T_{target} = 2 \times \frac{1}{3.125MHz / 64} \quad (4.2)$$

同时截取 i2s_din_mic0，可以发现在 I2S 左侧输入的数据按十六进制来表示的话是 A96A，和第二个浮标处的 m_axis_data_tdata 匹配。

4.2 音频预处理模块

音频预处理模块包含音频滤波模块和音频插值模块，模块整体输入输出均为 64 位，也就是 16 位有符号整数音频信号。在本系统设计中，使用了 Xilinx 提供的 AXI4-Stream Broadcaster 和 AXI4-Stream Combiner，这两个 IP 核都是 Xilinx 提供的 AXI4 Infrastructure 得一部份，其中 AXI4-Stream Broadcaster 的作用是将一路（Master）的 AXI4-Stream 信号分发给多个 Slave，每一个 Slave 都提供一个独立的 TVALID 和 TREADY 握手，最高支持驱动 16 个 Slave；AXI4-Stream Combiner 则可以将多路 Slave 信号组合成一路 Master

信号输出给下一级。结合这两个 IP 核，就可以将四路音频信号分发给四个相同的模块进行并行处理，如图 4.4 所示，最后再把四路处理完毕后的信号汇成一路 64bit 的信号给下一级。

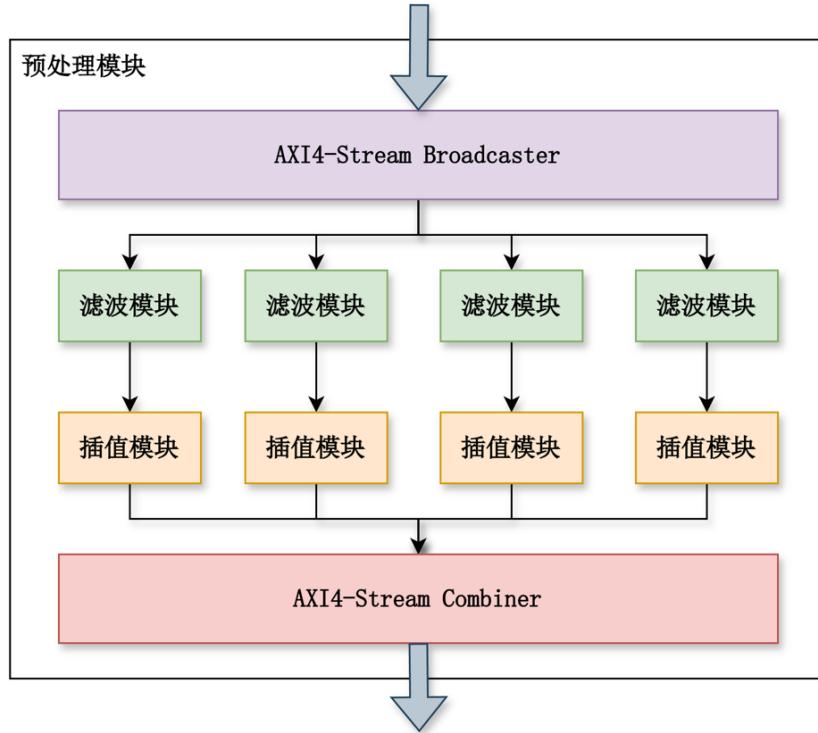


图 4.4 预处理模块整体框图

AXI4-Stream 协议属于 ARM 公司 AMBA(Advanced Microcontroller Bus Architecture)规范的一部份。AXI4-Stream 主要用于点对点的数据流传输，适用于高速数据流处理，如视频、音频和其他类型的串行数据流，它的接口定义如图 4.5 所示，其中重要的信号是 TVALID、TDATA、TLAST 和 TREADY，这几个信号和 AXI4-Stream 的握手协议有着重要的联系。

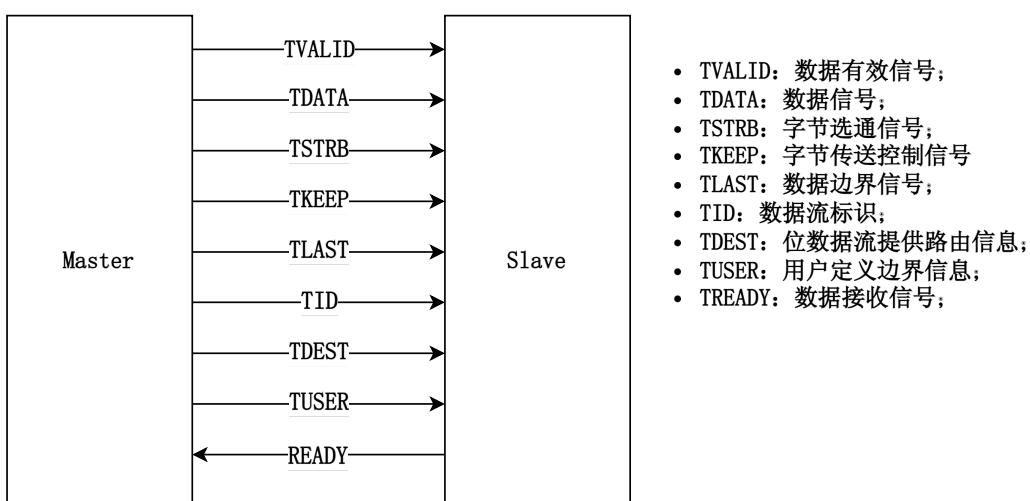


图 4.5 AMBA AXI4-Stream 接口定义

对于 AXI4-Stream 协议来说，握手机制可以分为三种：

(一) TVALID 在 TREADY 之前拉高

这种情况是相当常见的。在某些情况下，由于各种原因，后续模块无法及时处理新数据，因此会对前级模块施加反压，前级模块必须保持 TVALID 和 TDATA 不变，直到握手成功才能允许更新，握手成功意味着下级模块已经接收到前级传递的数据。

在实际应用中，通常不允许反压。例如在 DSP 系统中，要求实时持续地处理数据，一旦出现反压，新数据到来时就会丢失数据。一般情况下，系统出现这种情况通常是因为工作异常，需直接报中断进行处理。

(二) TREADY 在 TVALID 之前拉高

在这种情况下，后级模块随时准备接收数据，始终等待前级模块提供数据。例如，前级模块可能还未开始工作，只是在配置后才开始采集数据，这种情况下后级模块会持续等待前级模块传递数据。

(三) TREADY 和 TVALID 同时拉高

拉高当拍直接握手成功，不需要其他的步骤。

4.2.1 音频滤波

为了消除采集信号中噪声信号的影响，就需要对输入的语音信号进行滤波。由 2.2 节中的介绍可知，语音信号的带宽为 300Hz 到 3400Hz。为此本系统设计了一个低通滤波器来过滤掉语音信号带宽外的噪声信号， F_{pass} 设置为 3500Hz， F_{stop} 设置为 5500Hz。为了让处理后的信号相位不产生变化，系统本次采用的是有限冲激响应 FIR (Finite Impulse Response) 滤波器，结构示例可以参考图 4.6:

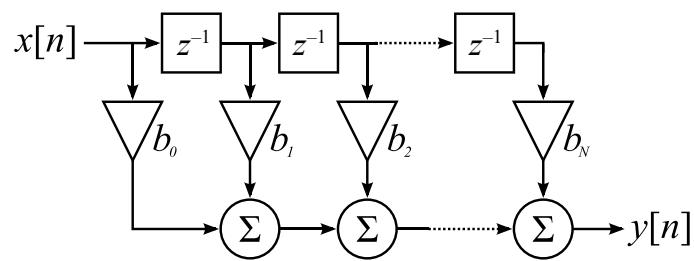


图 4.6 直接形式的 N 阶 FIR 滤波器

对于时域离散的 N 阶 FIR 滤波器，输出序列的每个值都是最近输入的加权之和：

$$\begin{aligned}
 y[n] &= b_0 x[n] + b_1 x[n-1] + \cdots + b_N x[n-N] \\
 &= \sum_{i=0}^N b_i \cdot x[n-i]
 \end{aligned} \tag{4.3}$$

其中， $y[n]$ 为输入信号， $x[n]$ 为输出信号， N 为 FIR 滤波器的阶数， b_i 为 N 阶 FIR

滤波器在时间 i ($0 \leq i \leq N$) 上的脉冲响应。上诉项中的 $x[n - i]$ 也被称为抽头项，在许多实现中，会将输入延迟，再做乘法运算，以减少使用乘法器的数量。滤波器的冲激响应被定义为有限区间内的非零值，具体如下：

$$h[n] = \sum_{i=0}^N b_i \cdot \delta[n - i] = \begin{cases} b_n & 0 \leq n \leq N \\ 0 & \text{otherwise.} \end{cases} \quad (4.4)$$

滤波器设计基于 Matlab Filter Designer 和 Vivado 提供的 FIR Compiler IP 核，实现基于最小二乘，滤波器阶数为 60 阶，具体量化后的频率响应如图 4.7 所示：

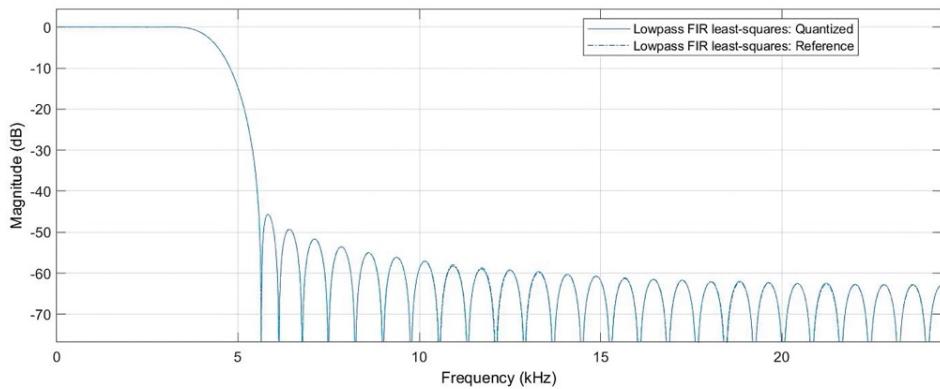


图 4.7 FIR 滤波器量化后的频率响应

导入采集的声音样本进行测试，结果如图 4.8 所示，可以看到滤波器有效地去除掉了声音中的高频毛刺：

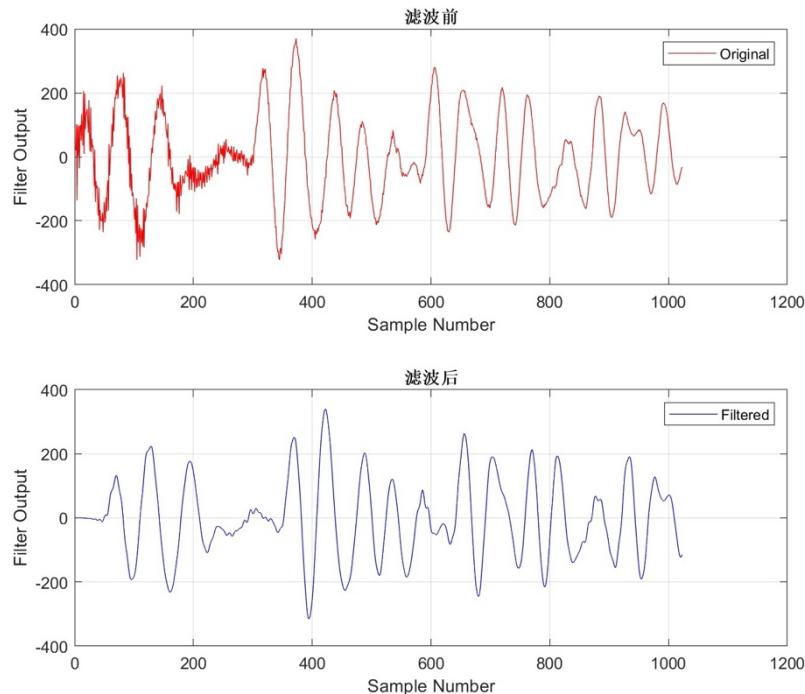


图 4.8 FIR 滤波器前后时域波形图

在将量化的系数导入 IP 核之后，整体的输出延迟为 41 个时钟周期（50MHz），与音频采集的时钟周期相比非常小，不足以对系统产生延迟的影响。整体的仿真结果如图 4.9 所示，IP 核的输出为 40 位（图中的 axis_fir_0_tdata），不可能直接送入后级处理，所以需要对输出进行截位处理，同时要保证滤波之后的数值不能和以前相差太多，根据图中的仿真结果，可以发现，选择输出的 32 到 17 位最为合适，能保证和输入信号的数量级一致。

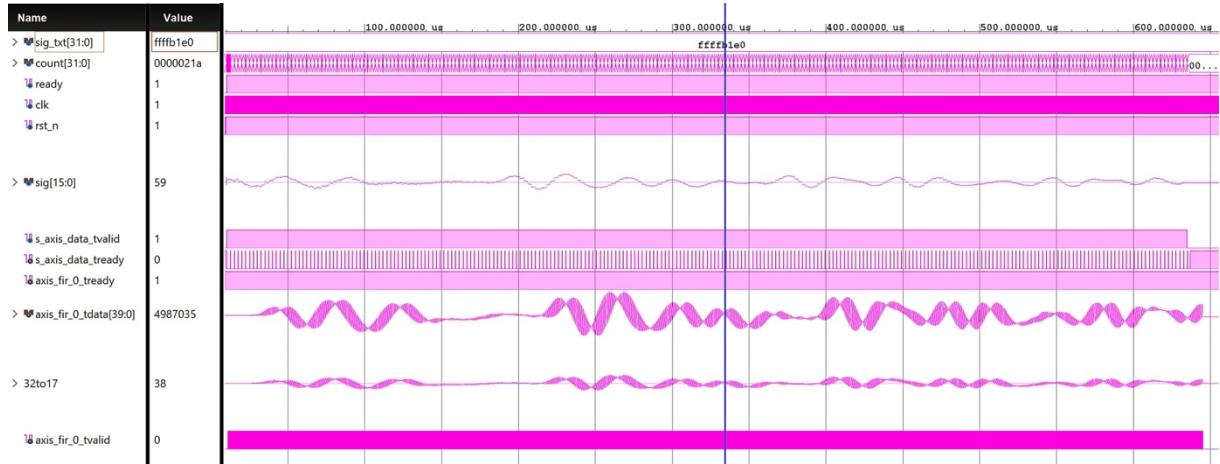


图 4.9 FIR 滤波器仿真结果

同时，因为 IP 核的处理需要时间，也就是之前所说的 41 个时钟周期延迟，所以波形图不是连续的。为了避免前面提到的 AXI4-Stream 的反压问题，对滤波器的输出进行了改进，使得输出的数据不会受到 TREADY 的阻塞。

4.2.2 音频插值

由式 (2.3) 可知，对于一对麦克风（线阵）来说，远场模型定位角度的结果数量其实是有限的，具体取决于时延 τ 的区分度，我们可以对式 (2.3) 做一个拓展，假设时延为 $\tau = n/f_s$ ， n 为时延估计中得出的峰值点， f_s 为采样频率，相应可以得到角度的计算表达式，即

$$\theta = \arccos\left(\frac{\tau_{12} \cdot c}{d}\right) = \arccos\left(\frac{c \cdot n}{f_s \cdot d}\right) \quad (4.5)$$

上式表明，在一个固定的时延区间 $\tau \in [-\tau_{max}, \tau_{max}]$ 内，时延 τ 的区分度，或者说测向算法能计算出的角度颗粒度，取决于峰值点在互功率谱中的位置区间范围 $n \in [-n_{max}, n_{max}]$ ， n_{max} 越大，理论上在-90°到 90°这个角度区间内能得到的可能角度结果越多，也就是结果的颗粒度/区分度更好。在一个麦克风阵列测向系统中，想要提高 n_{max} 的值，就需要提高高效的采样率 f_s ，或者增加阵列的尺寸 d ，但后者对于嵌入式应用来说基本不可能。图 4.10 所示为阵列尺寸固定的情况下，不同采样率对测向结果颗粒度的影响。

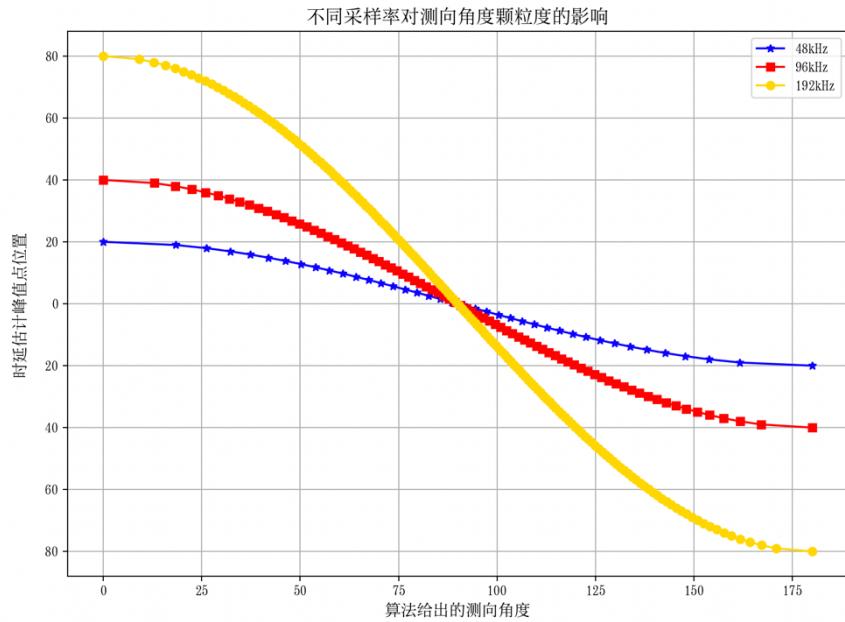


图 4.10 不同采样率对单对线阵麦克风声源测向角度颗粒度的影响

因此，本系统设计了一个音频插值模块，对输入的信号进行四倍插值，让等效采样率从采集模块的 48.8kHz 提升到 195.2kHz。模块的具体实现基于级联积分梳状 CIC (Cascade Integrator Comb) 滤波器，其又被称为霍根诺尔滤波器，是多率信号处理中一种十分常见的滤波器，通常在数字信号处理中实现大范围采样率的变化。CIC 滤波器通常用于具有一些有着大量“剩余样本率”的系统中，也就是系统样本率远大于被处理信号的带宽或者相反的情形，如数字下变频器（下采样）和数字上变频器（上采样）。CIC 滤波器的实现结构仅仅需要使用加法器、减法器和延迟元件，这让 CIC 滤波器在硬件电路系统中较为容易实现，在 FPGA 的场景中十分合适。

CIC 滤波器由两部分组成，积分器的级连和梳状滤波器的级连，如图 4.11 所示：

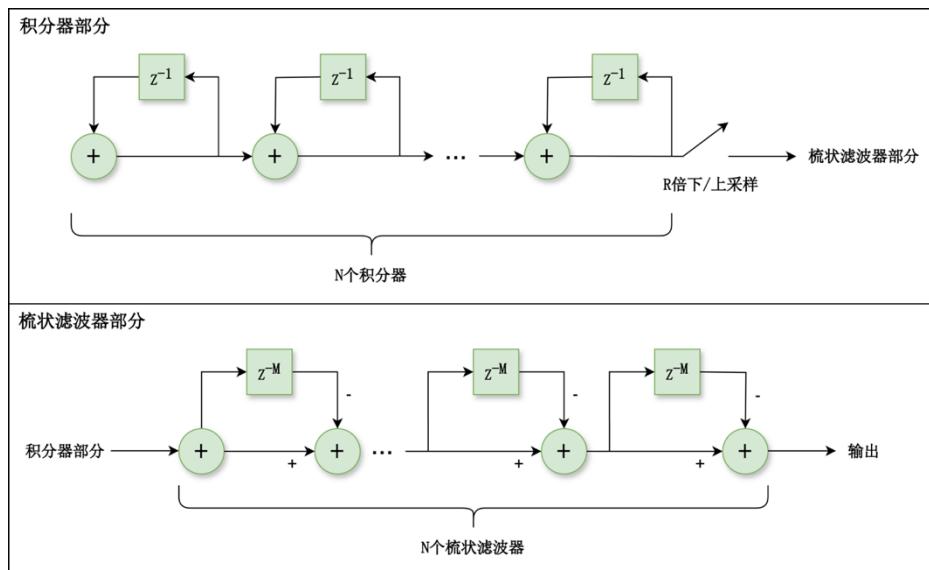


图 4.11 CIC 滤波器的组成

CIC 滤波器的基本概念是，把输入信号经过 N 个长度为 $R * M$ 的矩形窗的单位幅度滤波，就会得到 CIC 滤波器的系统响应方程，即式 (4.6)：

$$H(z) = \frac{(1 - z^{-R*M})^N}{(1 - z^{-1})^N} \quad (4.6)$$

其中 N 为 CIC 滤波器的阶数， R 为采样频率的变化（上采样或者下采样）， M 为梳状滤波器阶段的滤波延迟。当 CIC 滤波器做下采样时，数据先经过积分滤波器部分，之后在经过梳状滤波器部分，对于上采样则相反。本次系统的实现整体阶数为 5，输出会从输入的 16bit 提高到 24bit，具体可以用式 (4.7) 计算 (B 为输入位数)：

$$B_{max} = \left\lceil \log_2 \frac{(RM)^N}{R} + B \right\rceil \quad (4.7)$$

为了后续的处理，模块直接截取高 16 位进行输出，整体输出延迟为 20 个时钟周期，总计使用两个乘法器，整体量化后具体的频率响应如图 4.12 所示：

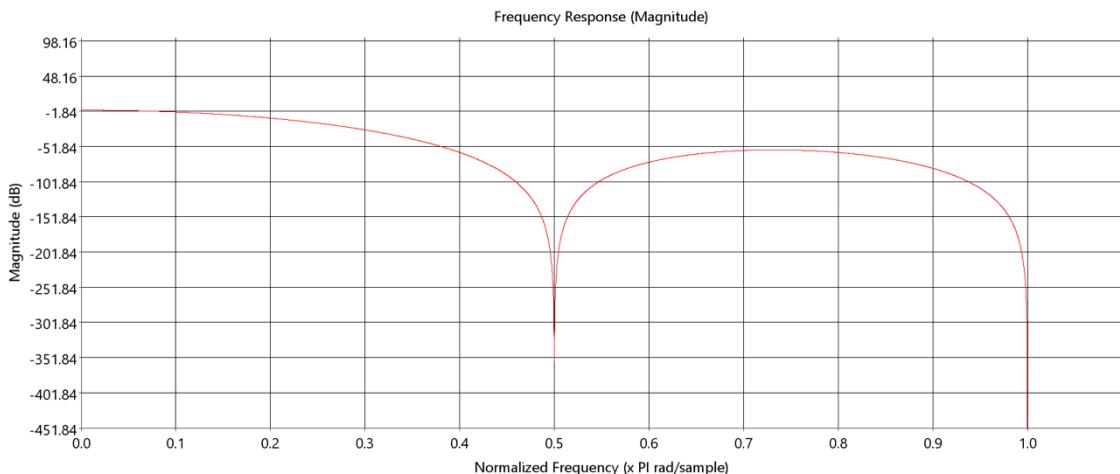


图 4.12 插值模块 CIC 滤波器频率响应（量化后）

模块仿真结果如图 4.13 所示，在输入-169 的数据之后，生成了-162、-165、-178 和-201 四个结果，符合四倍插值的设计要求。

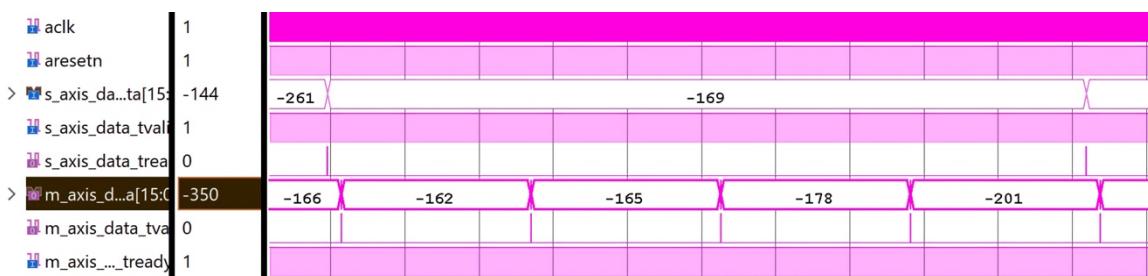


图 4.13 插值模块仿真结果

4.3 有声检测模块

有声检测模块的目的是为了让系统只在真正有目标声音出现时进行处理，避免一直对噪声场信号进行处理，不仅浪费算力还会产生无效结果，降低系统的效率。在有声检测或语音信号处理领域，短时能量法和短时过零率法是两种常用的方法，用于从背景噪声中检测和区分语音信号，下面将对两种方法进行简短的介绍：

(一) 短时能量法

短时能量法是一种基于能量的特征提取方法，用于区分语音和非语音段。这种方法基于的原理是，语音段的能量通常高于静音段或背景噪声段。短时能量通常通过以下步骤计算，先将信号分割成短时帧，每一帧通常包含几毫秒到几十毫秒的数据，之后对于每一帧，计算其能量，通常是求帧内所有样本的平方和来实现，如式 (4.8) 所示。其中 $x[n]$ 是第 n 个样本， N 为帧中样本的总数。

$$E = \sum_{n=0}^{N-1} |x[n]|^2 \quad (4.8)$$

一般在应用中，会设定一个阈值，比较每一帧的能量，如果能量超过阈值，则认为该帧包含语音；否则，认为是静音或背景噪声。阈值的设定是一个关键问题，可能需要根据不同的环境噪声水平动态调整。

(二) 短时过零率法

短时过零率是另一种用于语音检测的技术，它测量的是信号波形在一定时间内穿过零点的次数。语音信号的过零率与它的频率成分有关，通常在语音段和静音段之间有明显的不同。短时过零率可以通过以下步骤计算，首先同样对语音信号进行分帧，对于每一帧，计算穿过零点的此数，如式 (4.9) 所示，其中 $sgn(x)$ 为符号函数， $x \geq 0$ 则 $sgn(x) = 1$ ，否则 $sgn(x) = -1$ 。

$$Z = \frac{1}{2} \sum_{n=0}^{N-1} |sgn(x[n]) - sgn(x[n-1])| \quad (4.9)$$

一般在应用中只需要一个静态阈值，比较每帧的过零率，可以区分语音和非语音。语音通常具有更高的过零率，特别是含有高频成分的语音。

在实际应用中，短时能量法和短时过零率法常常结合使用，以利用两种方法的优点，提高语音活动检测的准确性和鲁棒性。例如，可以同时使用这两种方法的结果来决定一个帧是否包含语音，一个基于静态阈值，一个基于动态阈值这样可以在不同的信号条件下更好地适应环境噪声和信号变化。

本次模块应用选择结合两种方法一起使用，整体流程如图 4.14 所示：

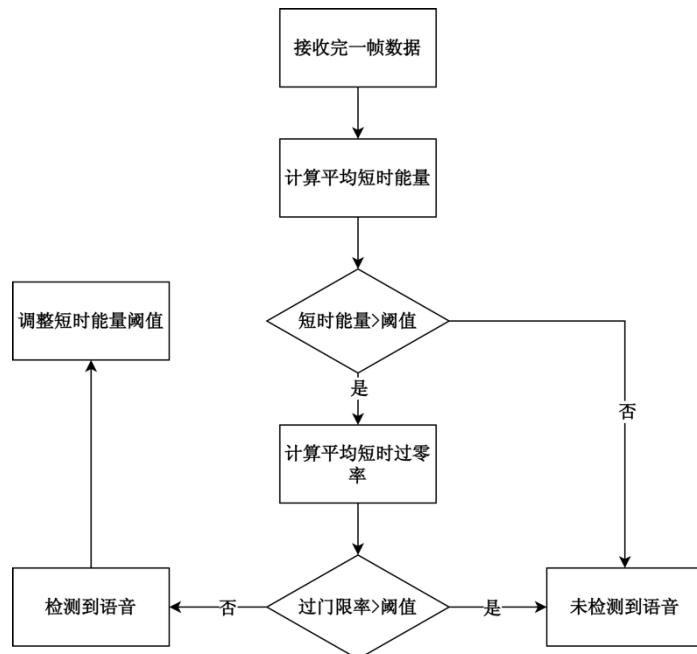


图 4.14 有声检测流程图

有声检测的仿真结果如图 4.15 所示，可以看出，在高分贝的情况下，vad_result 输出为 1。



图 4.15 有声检测模块仿真结果

4.4 时延估计模块

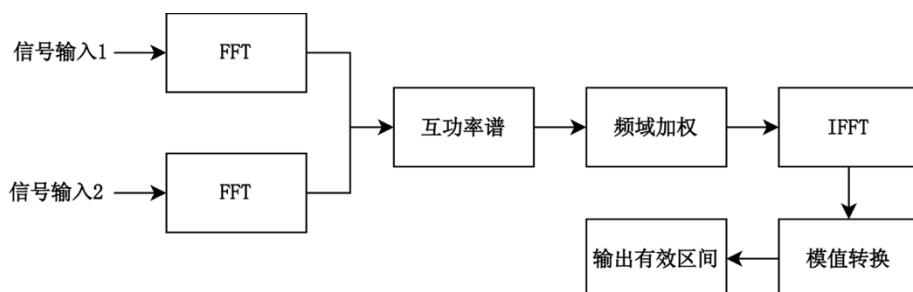


图 4.16 时延估计模块内部框图

时延估计模块实现的是对 4 个通道的 16bit 信号做时延估计，输入信号的等效采样率为 195.2kHz，共需要计算 4096 点进行计算，整体的结构如图 4.16 所示。

时延估计的第一步是把 4 个通道的 4096 个 16bit 数据通过 FFT 将其转换为 4 路频域信息。对于 FFT，Xilinx 贴心地提供了对应的 IP 核，可以直接调用，其支持 4 种运算架构，Pipelined Streaming I/O、Radix-4 Burst I/O、Radix-2 Burst I/O 和 Radix-2 Lite Burst I/O。其中 Radix-2 Burst I/O 在性能和资源之间取得一个比较不错的平衡，采用 Radix-4 Burst I/O 相同的迭代方法，但需要的阶数更少。这意味着它比 Radix-4 Burst I/O 消耗资源更少，但转换时间比 Radix-4 Burst I/O 更长，所以本次设计将会选用 Radix-2 Burst I/O，最后整体消耗了 6 个乘法器，17 个 BRAM，整体处理时间为 659.46us。

同时因为在 FFT 运算中每次蝶形运算都会增加数据的位数，如果不选用缩放模式，16bit 的输入将会得到 27bit 的输出，因为后面还需要对接乘法器，不对位数进行限制的话将会迎来位数的指数增长，所以本次设计启用缩放模式，经过测试，选用缩放系数为 [00 00 00 00 00 00 00 01 01 01 01 01 01 01]，总计缩放 32 倍，为一个输出精度的平衡点。



图 4.17 时延估计 FFT 模块仿真结果

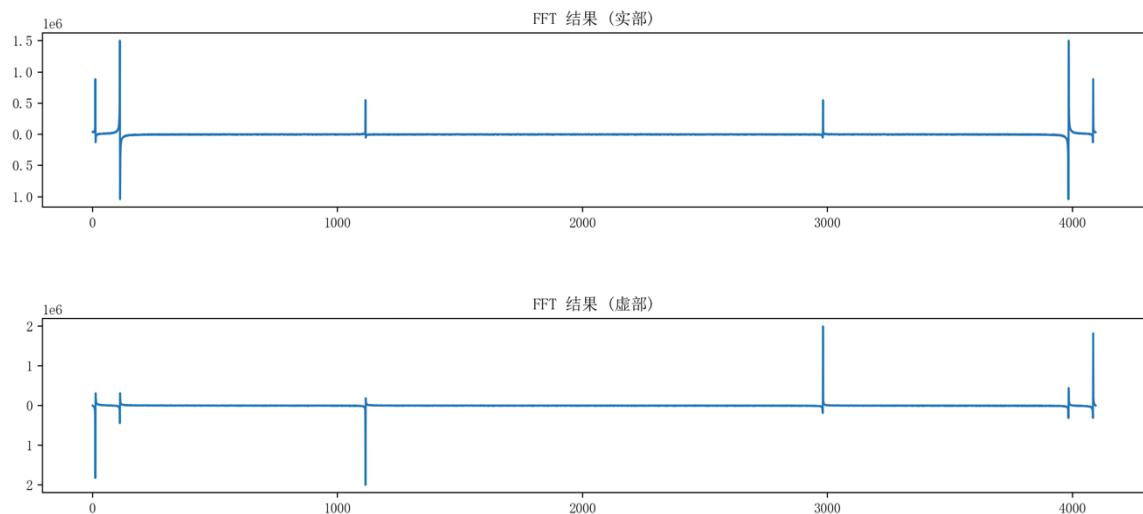


图 4.18 相同数据在 NumPy 中的仿真结果

FFT 模块的仿真结果如图 4.17 所示，上面为 FFT 输出的实部，下面的为 FFT 输出的虚部，将相同的数据带入 NumPy 中进行计算，结果如图 4.19 所示，可以发现，硬件 FFT 模块与 Python 仿真的结果基本一致。

在完成 FFT 的频域转换后，处理完的频域数据将送入互功率谱模块计算。互功率谱模块说简单一点就是一个共轭乘，本次采用 Xilinx 自带的 Complex Multiplier，将其中一侧的复数用 Verilog HDL 的\$signed 语法转为有符号数然后取负送入 Complex Multiplier 中。对于普通的乘法器，16 位的数据之间进行乘法，将会输出一个 33 位的数据，但 Complex Multiplier 允许 Random rounding，即随机取整，和普通的截位相比效果更好，具体效果如图所示，但这个模式下的 Complex Multiplier 需要用户通过 TUSER 输入一个随机信号，这里选择直接选用时钟信号的二分频输入。在使用随机取整之后，输出被降低到 24 位。

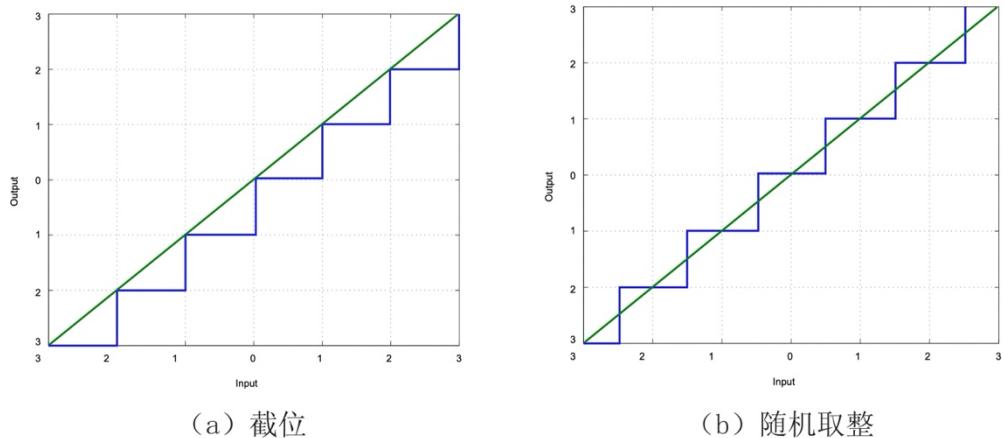


图 4.19 随机取整效果展示

频域加权模块完成了两个工作，加权系数的计算和加权的进行。对于加权系数的计算，对于 GCC-PHAT 来说，就是计算这个复数的模值，为了提高计算的效率，在这里选用 Xilinx 官方提供的 Cordic IP 核，选择 translate 模式，也就是将直角坐标转换为极坐标，然后只取模的部分就可以完成 PHAT 系数的计算。同时因为 Cordic IP 启动需要 34 个时钟周期，在这里为了保持同步，在两者之间加入了一个移位寄存器，将互功率谱结果和权值一并送入后级的除法器中。

后级除法器基于 Xilinx 的 Divider Generator IP 核，两组分别并行对实数和虚数进行加权，除法器 IP 核支持余数和小数输出，具体数据格式如图 4.20 所示，IP 核支持自定义余数/小数宽度，同时商（Quotient）、余数（Remainder）和小数（Fractional）都是独立的带符号数，不可以直接拼接使用，如果想要得到完整的数必须去掉小数的符号位，所以为了方便使用，我们让加权后的理论输出均小于 1，既可以用小数输出精度来做输出位数控制，又避免了两个有符号数的拼接。



图 4.20 Divider Generator IP 输出格式

频域加权模块的整体设计如图 4.21 所示：

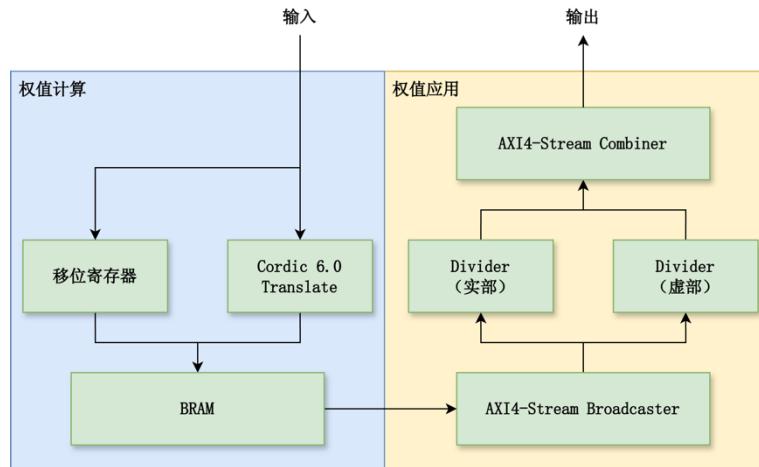


图 4.21 频域加权模块整体框图

最后选择让除法器输出 16 位宽度的小数，这样对于频域加权模块来说，输出就是 4 通道 16bit 数据，之后这些数据将会送到 IFFT 模块，把频域的功率谱转换回时域。IFFT 模块使用的也是 Xilinx 提供的 FFT IP 核，同时输出也选择缩放 16bit 输出，整体缩放系数为[00 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01]，总计缩放 2048 倍。IFFT 完成处理后，利用 Xilinx Cordic IP 核，选择 translate 模式，输出 16bit 的极坐标，只取模部分，送入后级模块输出有效区间，至此完成对于时延估计模块的介绍。

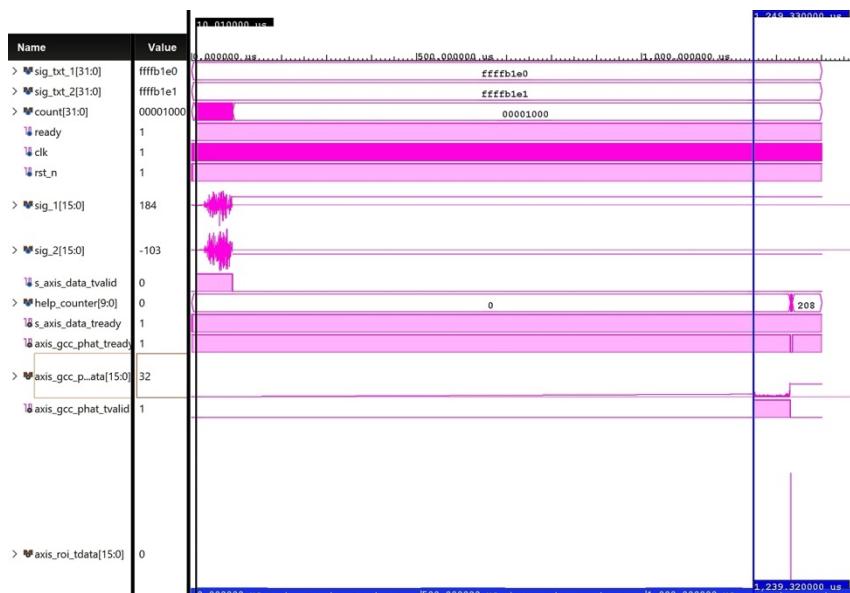


图 4.22 时延估计模块整体处理耗时仿真

整体模块的耗时如图 4.22 右下角所示，延迟约 1.2ms，与采集 48.8kHz 采集 1024 点所需的 20.9ms 相比足够地小，可以完成对采集的每一帧的处理。

不过在实验和仿真中，对于实际的音频信号，PHAT 加权的效果往往不尽如人意，与普通互相关相比，虽然互功率谱的尖峰更加明显，但尖峰位置的多样性却随之显著增加，或者说，尖峰的位置变得更加不确定了，整体产生的效果如图 4.23 所示，其中正确峰位置为 $x=78$ 处，无加权预测正确，而 PHAT 加权却在零延迟处产生了一个极高的错误峰。PHAT 在对整个互相关谱进行白化滤波时，虽然成功将大值缩小，但同时也将模较小的复数提升到了一个较高的方式，而这些小的频率成分对整体测向结果没有什么贡献，增强他们反而削弱了有效的频率成分，因此造成了较多错误峰的出现。

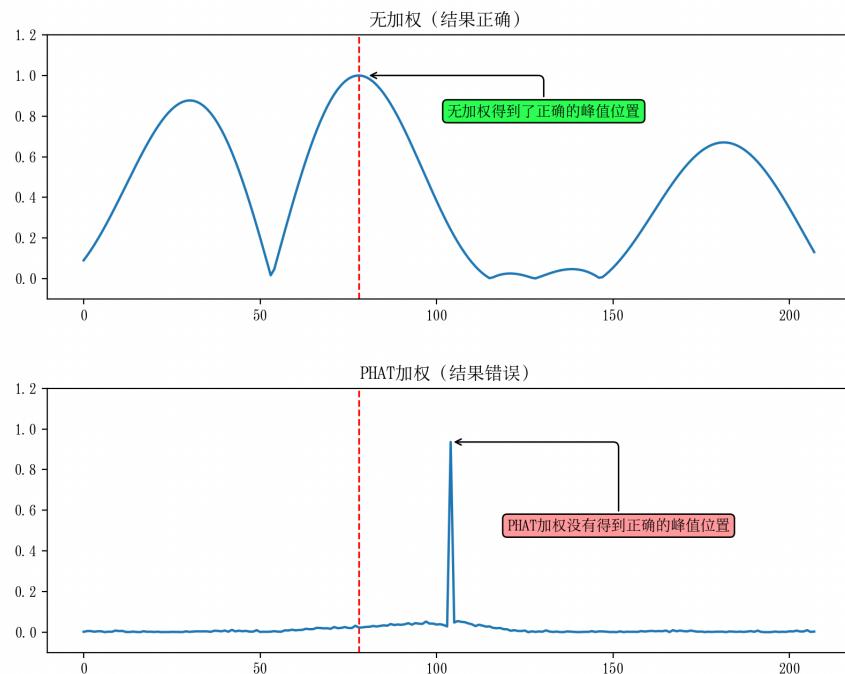


图 4.23 无加权（结果正确）和 PHAT 加权（结果错误）表现对比

为此，本文使用了一种改进的 PHAT 加权，通过为给权值添加一个常数来缓解错误峰现象，即将表 2.2 中的 PHAT 加权更改为式 (4.10) 的形式：

$$\frac{1}{|\mathbf{R}_{x_1, x_2}(\omega)| + C} \quad (4.10)$$

为了硬件上的实现简单，我们选择基于 4096 个数据的平均模作为基础去设计常数 C ，这样算法在数学上也较为简单，能在 FPGA 上进行快速部署。其实这就模块内保留了一块 BRAM 的原因，这块 BRAM 让可以将数据进行缓存，之后计算模的平均值，对这个值做一定的处理，之后 BRAM 输出的时候把输出值经过一个加法器加上计算出来的常数 C 即可。经过 NumPy 的仿真测试，发现 0.1 倍的平均模效果较好，可以在不产

生错误峰的同时有效增强正确峰和噪音峰的对比，整体效果如图 4.24 所示，在与图 4.23 相同输入音频数据的情况下，可以看到改进的 PHAT 加权显著地抑制了旁瓣。

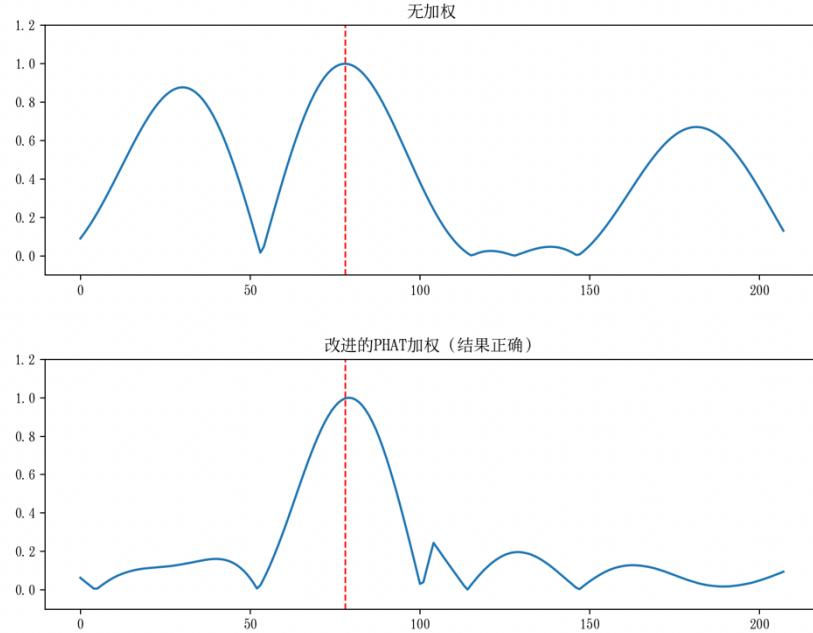


图 4.24 无加权和改进的 PHAT 加权表现对比

将算法部署到 FPGA 侧后，输入相同的音频数据，时延估计模块整体的仿真结果如图 4.25 所示，可以见到结果基本正确，因为算法在部署的过程中有量化和取整的存在，所以结果不是百分百匹配。

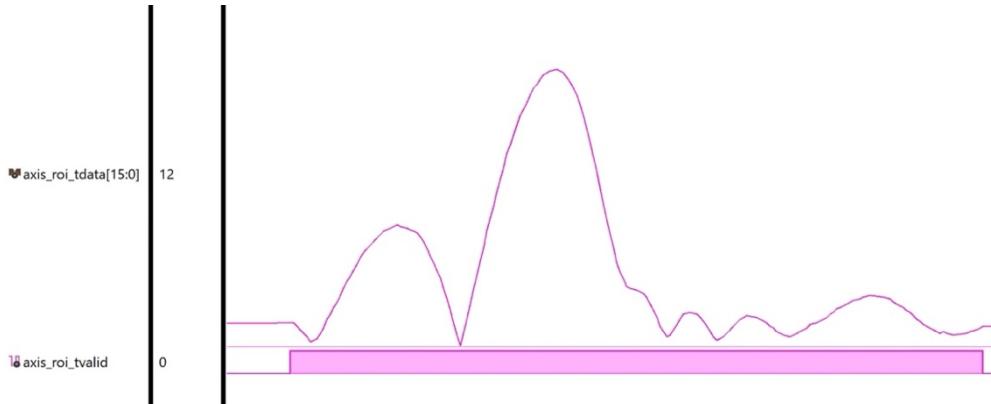


图 4.25 时延估计模块仿真结果

4.5 方位角估计模块

本模块运行在 PS 端，在基于 c 语言的帮助下，设计者可以实现灵活的系统设计和复杂的系统调度，如果阵列设计有所更改，只要阵列的最大尺寸不超过现有设计的大小，就可以只更改 PS 端的 c 代码，不仅整体实现更快，还可以避免 PL 端冗长的综合和仿真过程，极大地提高了设计整体的灵活性。

具体的实现主要基于 2.3.2 节介绍的内容。由式 (2.16) 得, 可以得到平面正十字阵的测向估计矩阵, 即式 (4.12), 其中 d 为正十字阵中麦克风之间的距离。

$$\begin{bmatrix} d & 0 \\ 0 & d \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} c\tau_{20} \\ c\tau_{31} \end{bmatrix} \quad (4.12)$$

将式 (4.12) 带入式 (2.18) 中, 可得方位角的计算式, 即式 (4.13), 其中 τ_{20} 为图 3.4 所示 mic0 与 mic2 之间的时延估计结果, τ_{31} 同理。

$$\varphi = \arctan\left(\frac{\tau_{20}}{\tau_{31}}\right) \quad (4.13)$$

而由式 (4.5) 可知, 时延 τ 和互功率谱峰值位置成正比, 所以最后方位角可以通过直接将两个峰值位置送入 atan2 函数得到, 运行流程图如图 4.26 所示, 整体被包装成一个触发函数。

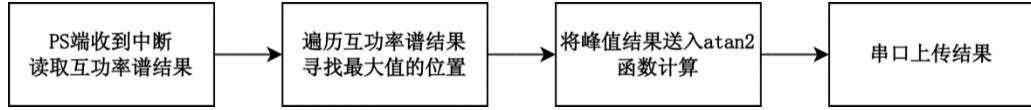


图 4.26 方位角估计模块运行流程图

PL 端的时延估计结果通过映射的内存 (BRAM) 由 PL 发向 PS, 整体的传输结构如图 4.27 所示。其中 AXI SmartConnect 是 Xilinx 提供的系统连接生成器, 可以实现将一个或者多个内存映射主设备和一个或多个内存映射从设备进行连接, 具体的处理细节由 Vivado 的主动连线负责, 操作起来不需要关心实现细节。AXI GP 即 AXI 通用总线, Zynq7020 平台提供了两条 AXI GP 和两条 AXI HP (高性能) 总线供使用, 由于本系统在 PL 与 PS 之间的数据传输量并不大, 所以实现选择使用 AXI GP 总线。

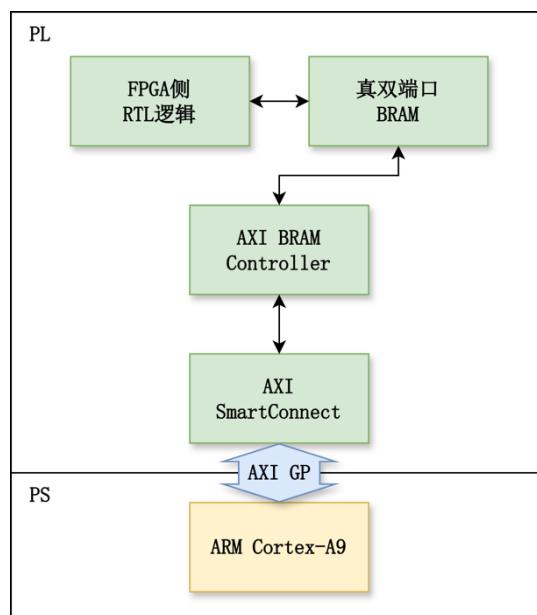


图 4.27 片上通讯架构图

整体传送的数据主要是时延估计模块的结果，共 208 个 16bit 数据，数据的个数可以用式(4.11)计算得，其中 d_{max} 为两麦克风之间的法向距离，在本系统的设计即 18cm； f 为等效采样率，在本系统中即 195.2kHz； c 为声速，设为 342m/s。

$$N_{max} = 2 \times \frac{d_{max} \cdot f}{c} \quad (4.11)$$

从存储深度来看，8192x32bit 的 BRAM 完全足够；从传输速度来看，AXI GP 总线支持的最大传输速度为 600MB/s，本系统一秒共处理约 50 帧数据，整体需要的传输速度不到 500KB/s，不会有传输速度上的瓶颈。结果发送的仿真结果如图 4.28 所示：

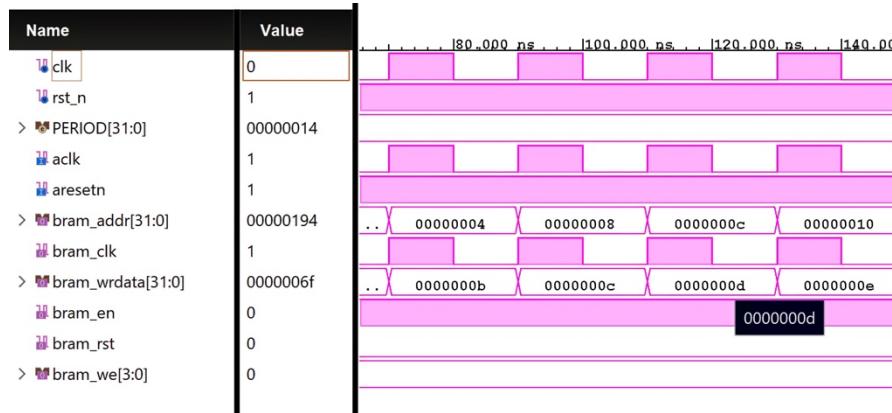


图 4.28 测向结果发送仿真结果

4.6 本章小结

本章以第二章介绍的理论基础和第三章介绍的 FPGA SoC 硬件平台为基础，结合 FPGA 与 MEMS 设计了一套基于平面正十字阵的声源测向系统。针对 MEMS 采集的麦克风信号进行 FIR 滤波和 CIC 插值，利用有声检测触发系统的运行，并提出了一种改进的 PHAT 加权方法，提高了整体测向的精度，在 50MHz 时钟频率下，完成一帧计算的时间约为 1.2ms，最后利用最小二乘的方法实现了测向估计算法的设计，完成了整体系统的软件设计。

5 系统测试及验证

第三章和第四章分别就系统的硬件设计和软件设计进行了介绍，本章在前面硬件设计和软件的基础上，对系统整体的资源消耗、功耗、时序和音频采集性能进行分析，再基于 FPGA 开发平台和平面麦克风阵列原型，搭建测试平台并分析测试结果。

5.1 系统硬件性能评估

系统顶层的 Block design 如图 5.1 所示，其中橙色的即 PL 侧主要逻辑部分。

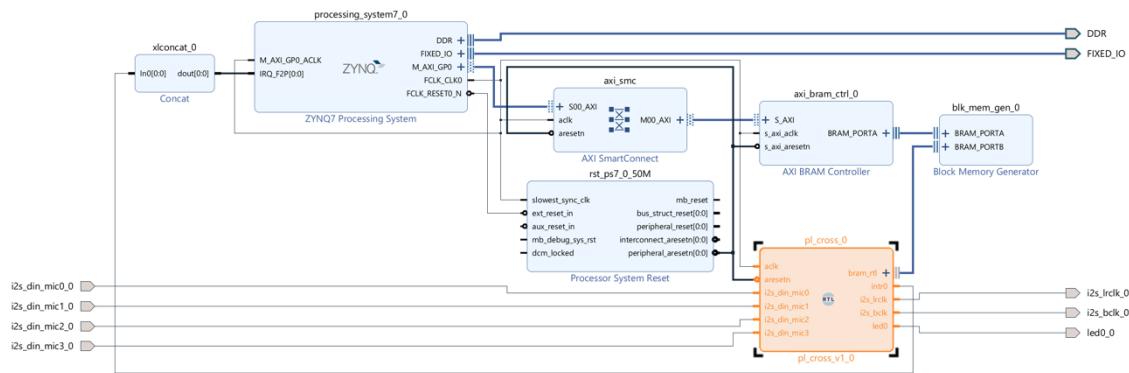


图 5.1 系统 Block design 图

5.1.1 系统资源占用分析

本次设计的测向系统所消耗的硬件资源如图 5.1 所示。其中 BRAM 的消耗较多，这个主要是因为插值之后数据量加大了四倍，所有不能就地处理的数据都需要缓存到 BRAM 中。整体查找表的使用率为 34.72%，寄存器的使用率为 27.38%，DSP 单元的使用量为 16.82%。

表 5.1 FPGA 硬件资源使用量

硬件资源	查找表	查找表 RAM	寄存器	BRAM	DSP
总资源数	53200	17400	106400	140	220
占用资源	18469	1416	29130	58.50	37
占用比例	34.72%	8.14%	27.38%	41.79%	16.82%

本文设计的系统的工作时钟频率为 50MHz，在 Vivado 中进行 Implementation 可以得到功耗估计报告，如图 5.2 所示，系统的总功耗为 1.971W，其中 PS (ARM) 侧的功

耗为 1.531W，PL 侧（FPGA）的功耗为 0.44W。其中 PL 侧的动态功耗为 0.28W，静态功耗为 0.16W，动态功耗占比为 63%。

Summary

Power analysis from Implemented netlist. Activity derived from constraints files, simulation files or vectorless analysis.

Total On-Chip Power:	1.971 W
Design Power Budget:	Not Specified
Process:	typical
Power Budget Margin:	N/A
Junction Temperature:	47.7°C
Thermal Margin:	37.3°C (3.1 W)
Ambient Temperature:	25.0 °C
Effective θJA:	11.5°C/W
Power supplied to off-chip devices:	0 W
Confidence level:	Low

[Launch Power Constraint Advisor](#) to find and fix invalid switching activity

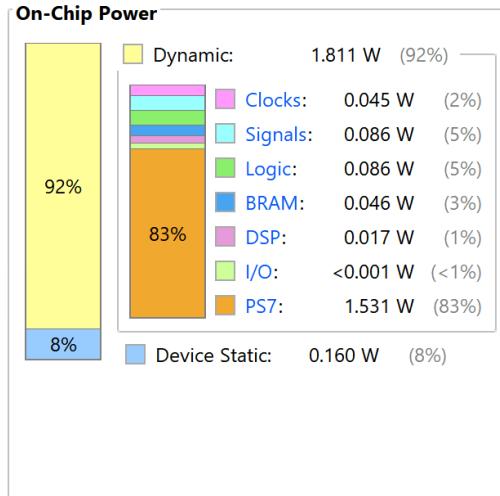


图 5.2 系统功耗估计报告

在时序方面，如图 5.3 所示，最差建立时间负时序裕量 WNS (Worst Negative Slack) 为 4.231ns，最差保持时序裕量 WHS (Worst Hold Slack) 为 0.007ns，最差脉冲宽度裕量 WPWS (Worst Pulse Width Slack) 为 8.750ns，整体都达到了 50MHz 下的时钟约束要求，保证了系统运行的稳定。

Design Timing Summary

Setup	Hold	Pulse Width
Worst Negative Slack (WNS): 4.231 ns	Worst Hold Slack (WHS): 0.007 ns	Worst Pulse Width Slack (WPWS): 8.750 ns
Total Negative Slack (TNS): 0.000 ns	Total Hold Slack (THS): 0.000 ns	Total Pulse Width Negative Slack (TPWS): 0.000 ns
Number of Failing Endpoints: 0	Number of Failing Endpoints: 0	Number of Failing Endpoints: 0
Total Number of Endpoints: 82355	Total Number of Endpoints: 82355	Total Number of Endpoints: 31517

All user specified timing constraints are met.

图 5.3 系统时序报告

5.1.2 系统音频采集性能评估

对于测向系统来说，精度一部份取决于运行的算法，一部份取决于阵列采集的精度，本部分主要是利用扬声器播放特定频率的测试音频，检查系统捕获的音频频率是否与播放音频的频率一致。

具体测试采用 1000Hz 和 2000Hz 的点频信号，信号获取采用 Xilinx 提供的 ILA (Integrated Logic Analyzer)，ILA 采集的时钟频率为系统时钟，即 50MHz，利用 ILA 截取有声检测模块后输出的信号，具体测试效果如下：

(一) 1000Hz 正弦波信号

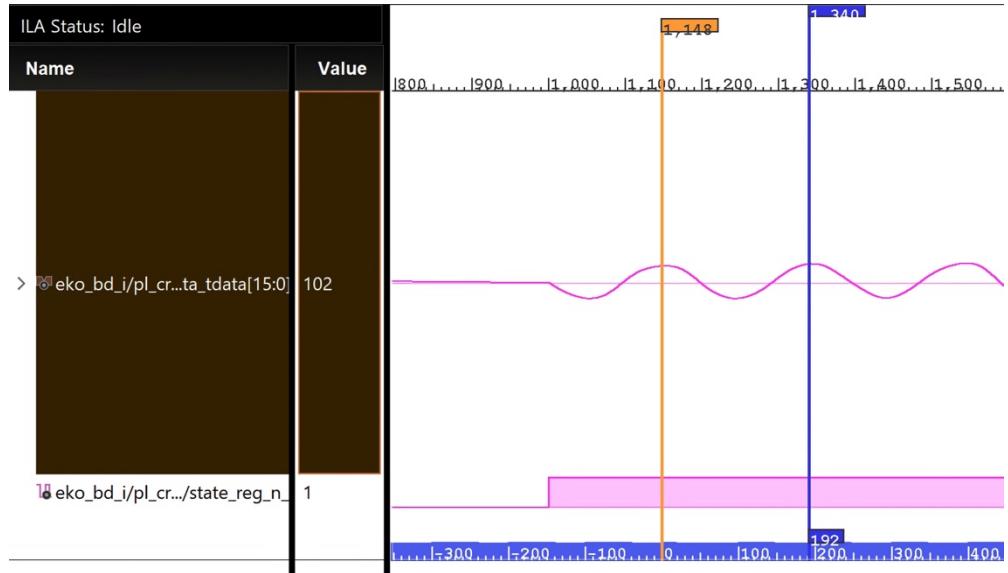


图 5.4 ILA 采集结果 (1000Hz 音频)

可以看到接收信号形状为正弦波，图中正弦波的周期约为 192 点，按照等效采样率 192.5Khz 来计算，有式 5.1，说明 1000Hz 的信号接收正常

$$f = \frac{195.2\text{kHz}}{192} = 1016.7\text{Hz} \quad (5.1)$$

(二) 2000Hz 正弦波信号

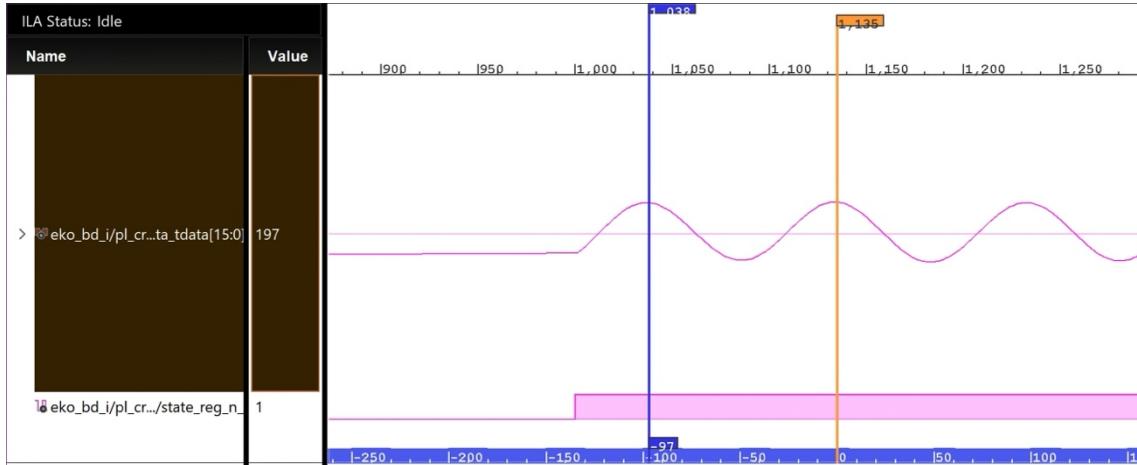


图 5.5 ILA 采集结果 (2000Hz 音频)

可以看到接收信号形状为正弦波，图中正弦波的周期约为 97 点，按照等效采样率 192.5Khz 来计算，有式 5.2，说明 2000Hz 的信号接收正常

$$f = \frac{195.2\text{kHz}}{97} = 2012.3\text{Hz} \quad (5.2)$$

5.2 系统测向功能测试

系统实验平台的原型如图 5.6 所示，整体测试环境在空旷的室外进行，减少室内混响对系统造成的干扰，声源离麦克风的距离分别为 1.5m，声源放置的方位角为 330°、180°、45°和 120°，声源为人咳嗽声的录音。上述每种测试情况进行 15 次重复试验，最后结果取平均值。

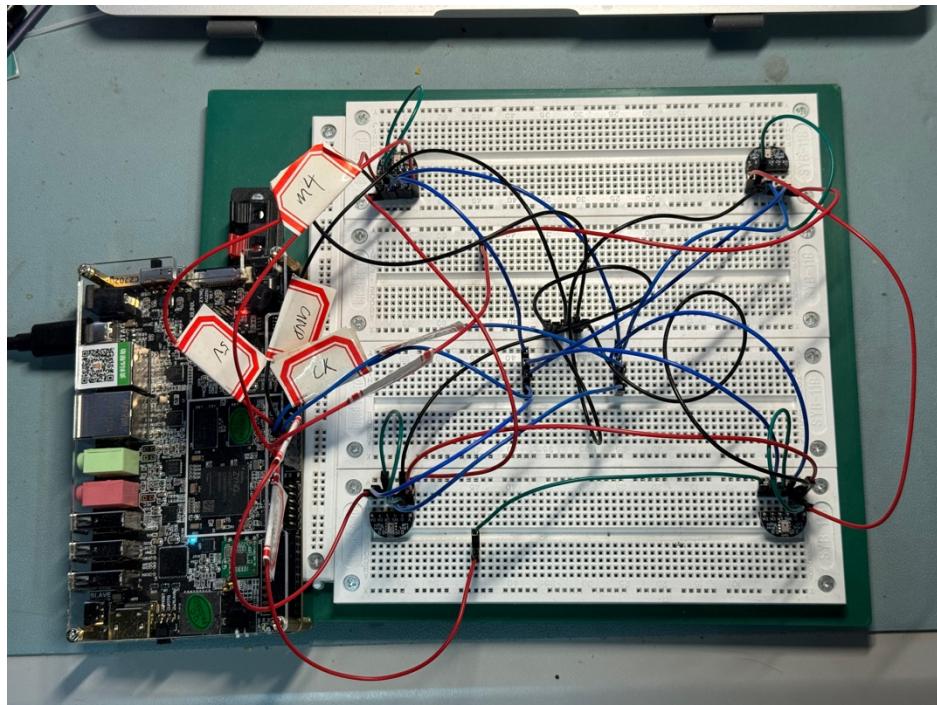


图 5.6 测试平台原型

具体测试结果如表 5.2 所示，整体的误差不超过 4°，可以实现准确的声源测向。

表 5.2 测向系统功能测试结果

实际方位角/°	330	180	45	120
方位角估计均值/°	328.73	176.26	47.63	117.36
估计误差的标准差/°	1.99	3.46	3.07	1.86

5.3 本章小结

本章是对系统总体的测试，首先分析了系统的资源消耗、功耗和时序，展示了系统设计的整体情况；然后对系统的音频采集准确性进行了分析，实验结果表明系统整体的音频采集和预处理模块工作正常；之后搭建了一套测向系统的原型，并基于这套原型系统进行测向准确性的测试，结果表明当声源离系统 1.5m 时，测向的方位角误差不超过 4°，测向功能可以正常运行。

6 总结与展望

6.1 本文总结

声源测向，即确定声源相对于观察者的方位角，是音频信号处理领域的一个关键应用。随着消费电子技术的飞速发展，越来越多的小型嵌入式设备，如智能音箱、无线耳机和笔记本电脑等，都开始集成微型麦克风阵列。然而，这些小型的麦克风音频处理系统常常面临着阵列尺寸小、采样频率和计算能力有限等挑战。为了解决以上问题，本文设计了一套基于 MEMS 麦克风和 FPGA 的声源测向系统，并对广义互相关法作出改进并应用，实现了一套基于平面正十字阵的声源测向系统。本文的研究内容和工作成果如下：

(1) 研究与分析了声源测向的基本概念及其研究现状，比较了多种常用的测向方法，分析了它们的优势与局限，并选择了适合于计算量较小、实时性好的时延估计方法作为本研究的基础。

(2) 完成了系统麦克风阵列的设计与软件在 FPGA 平台的部署。软件的整体设计包括 PL 和 PS 两部分，PL 部分包括音频采集模块，音频预处理模块、有声检测模块和方位角估计模块，PS 部分包括测向估计模块和用户交互，两个整体之间使用片上通信。算法应用上本文使用了一种改进的 PHAT 权重，基本消除了 PHAT 权重的错误峰值影响，最后基于 Xilinx Zynq-7020 平台完成了部署，完成一帧数据计算的时间为 1.2ms。

(3) 完成了系统整体的评估和测向的分析。对系统整体的硬件资源消耗量、功耗、时序做了分析，DSP 使用率仅 16.8%，在 50MHz 时钟频率下系统中 FPGA 侧功耗为 0.44W，最差建立时间负时序裕量 WNS (Worst Negative Slack) 为 4.231ns；在空旷的室外环境下，1.5m 处声源测向的平均误差小于 4°。

6.2 本文展望

在本文的研究过程中，发现有以下的问题可以加以改进：

(1) 本文的实现因为对于音频的插值，消耗的 BRAM 较多，如果将插值的步骤后移，比如对互功率谱的结果进行样条插值，所需的查找表和 BRAM 数量会大幅下降。

(2) 本文在求取互功率谱的时候采用的是单帧数据，如果通过加窗实现帧与帧之间的平滑过渡，效果会更好。

(3) 有声检测的实现较为简单，在复杂的声学环境下表现堪忧，如果可以使用其他更有效的方法，将能有效地减少误检率。

致 谢

首先感谢的是我的导师赵兆教授，在我论文写作的过程中为我答疑解惑，悉心教导。老师严谨的科研精神和负责的工作态度都给我留下了很深的印象，为我日后的学习和工作树立了榜样。

感谢我的朋友们，许洪康、赵天祺、钟一萌、张睿哲、连力、赵锦泽、...还有许多许多，有了你们，我大学四年的生活才充实而快乐，大家都帮助了我许多，尤其是在一些困难的时候，由衷地感谢你们。

感谢自己，没有放弃，准时交卷。

最后，还要感谢父母，是我求学路上坚强的后盾，感谢你们的鼓励与支持。

参考文献

- [1] 胡航. 语音信号处理[M]. 哈尔滨: 哈尔滨工业大学出版社, 2009.
- [2] 王松. 基于 TDOA 的声源定位算法研究与实现[D]. 山东大学, 2020.
- [3] Ganguly A. Noise-Robust Speech Source Localization and Tracking Using Microphone Arrays for Smartphone-Assisted Hearing Aid Devices[D]. The University of Texas at Dallas, 2018.
- [4] Jin C, Pang Z, Hou R, Qi G, Ou H, Yuan C. Application of Acoustic Detection Technology in Reconnaissance Warning Equipments[J]. In Man-Machine-Environment System Engineering: Proceedings of the 16th International Conference on MMESE 2016 (pp. 275-280). Springer Singapore, 2016.
- [5] Terzic M. One approach to the evaluation of the effectiveness of the boomerang system for acoustic source localization and identification[J]. Military Technical Courier, 2010, 58(3):78-87.
- [6] Lylloff O, E Fernández-Grande, Agerkvist F, et al. Improving the Efficiency of Deconvolution Algorithms for Sound Source Localization[J]. Journal of the Acoustical Society of America, 2015, 138(1): 172-80.
- [7] 司春棣,陈恩利,杨绍普,等.基于声阵列技术的汽车噪声源识别试验研究[J].振动与冲击,2009(6):171-174,190.
- [8] 张中盘, 张明, 时瑛, 李宁钢, 董帆, 万书亭. 皮带输送机托辊故障声源定位方法[J]. 噪声与振动控制, 2024, 44(1): 142-147.
- [9] 吴禹沈, 秦继兴, 李整林, 吴双林, 王梦圆, 顾怡鸣. 声学滑翔机联合的深海水下声源定位[J]. 声学学报, 2023, 48(3): 437-446.
- [10] Lylloff O, E Fernández-Grande, Agerkvist F, et al. Improving the Efficiency of Deconvolution Algorithms for Sound Source Localization[J]. Journal of the Acoustical Society of America, 2015, 138(1): 172-80.
- [11] 崔斌,陈亮,胡红梅,等.基于 Kinect 的声源定位时延获取及算法性能研究[J].信息技术,2015(10):103-107.
- [12] Crochiere R E, Flanagan J L. Speech Processing: An Evolving Technology[J]. Bell Labs Technical Journal, 1986, 65(5): 2-11.
- [13] Silverman B. A Practical Methodology for Speech Source Localization with Microphone Arrays[J]. Computer Speech & Language, 1997, 11(2): 91-126.
- [14] Cadzow J A, Kim Y S. General Direction-of-Arrival Estimation- A Signal Subspace

- Approach[J]. IEEE Transactions on Aerospace and Electronic Systems, 1989, 25(1): 31-47.
- [15] 赵文峰. 基于麦克风阵列的声源定位系统研究及实现[D]. 华中科技大学, 2009.
- [16] Akay M, Semmlow JL, Welkowitz W, Bauer MD, Kostis JB. Noninvasive detection of coronary stenoses before and after angioplasty using eigenvector methods[J]. IEEE Transactions on Biomedical Engineering. 1990 Nov;37(11):1095-104.
- [17] Schmidt R, Schmidt R O. Multiple Emitter Location and Signal Parameter Estimation[J]. IEEE Transactions on Antennas & Propagation, 1986, 34(3): 276-280.
- [18] Roy R, Kailath T. ESPRIT-estimation of signal parameters via rotational invariance techniques[J]. IEEE Transactions on acoustics, speech, and signal processing. 1989 Jul;37(7):984-95.
- [19] Hahn W, Tretter S. Optimum Processing for Delay-Vector Estimation in Passive Signal Arrays[J]. IEEE Trans.inf.theory, 1973, 19(5): 608-614.
- [20] Alvarado V M. Talker Localization and Optimal Placement of Microphones for A Linear Microphone Array Using Stochastic Region Contraction[M]. Brown University, 1990.
- [21] DiBiase J H. A High-Accuracy, Low-Latency Technique for Talker Localization in Reverberant Environments Using Microphone Arrays[M]. Brown University, 2000.
- [22] Li, Z, Duraiswami, R. Flexible and Optimal Design of Spherical Microphone Arrays for Beamforming[M]. IEEE Press, 2007.
- [23] Knapp C, Carter G. The generalized correlation method for estimation of time delay[J]. IEEE transactions on acoustics, speech, and signal processing. 1976 Aug;24(4):320-7.
- [24] Brandstein MS. Time-delay estimation of reverberated speech exploiting harmonic structure[J]. The Journal of the Acoustical Society of America. 1999 May 1;105(5):2914-9.
- [25] 夏阳, 张元元. 基于矩形麦克风阵列的改进的 GCC-PHAT 语音定位算法[J]. 山东科学. 2011 Dec 20;24(6):75-9.
- [26] 民用动圈麦克风 展现细腻动人的歌喉[J]. 家庭影院技术,2007(10):66-66.
- [27] 霍静茹, 宋文豪. 基于传声器阵列的声源定位[J]. 信息技术, 2016, 40(6): 136-138.
- [28] 杜功焕, 朱哲民, 龚秀芬. 声学基础. 第 3 版[M]. 南京大学出版社, 2012.
- [29] Traunmüller H, Eriksson A. The frequency range of the voice fundamental in the speech of male and female adults[J]. Unpublished manuscript. 1995 Jan;11.

附 录

本科期间发表的论文和出版著作情况:

本科期间参加的科学研究情况:

本科期间获得的荣誉与奖励情况: