**实验题目**

**实验一**

实验1：系统的频域和z域分析

设计计算机程序，产生序列并计算序列的DTFT，绘制其幅频特性和相频特性曲线；根据系统的单位脉冲响应和差分方程，计算系统的频率响应，绘制系统频率响应的幅频特性和相频特性曲线；根据系统的单位脉冲响应和差分方程，计算系统的系统函数、零极点分布；改变系统的零极点分布，观察系统频率响应的变化。

**实验二**

实验2：信号的频谱分析

设计计算机程序，产生序列并计算序列的FFT和IFFT，绘制其幅频特性和相频特性曲线；模拟产生离散系统的输入序列和单位脉冲响应，利用FFT和IFFT算法计算系统的输出响应，分析FFT的计算长度对系统输出响应的影响；模拟产生连续时间信号，选取适当的采样频率对其采样，并用FFT算法计算其频谱，分析信号的观测时间长度、FFT的计算长度对信号频谱计算结果的影响。

**实验三**

实验3：IIR数字滤波器设计及结构

设计计算机程序，根据滤波器的主要技术指标设计IIR数字巴特沃斯和切比雪夫低通、高通、带通和带阻滤波器；绘制滤波器的幅频特性和相频特性曲线，验证滤波器的设计结果是否达到设计指标要求；画出数字滤波器的直接型、级联型、并联型结构信号流图。

课外学习：课外利用编程工具，设计和分析IIR数字滤波器；设计实验内容和实验步骤，编写上机实验程序，撰写实验报告。

**实验四**

实验4：FIR数字滤波器设计及结构

设计计算机程序，根据滤波器的主要技术指标设计线性相位FIR数字低通、高通、带通和带阻滤波器；绘制滤波器的幅频特性和相频特性曲线，验证滤波器的设计结果是否达到设计指标要求；画出线性相位FIR数字滤波器的结构信号流图。

数字信号处理

实验报告

|  |  |
| --- | --- |
| 班级： |  |
| 姓名： |  |
| 学号： |  |
| 联系方式： |  |

西安电子科技大学

电子工程学院

目录

[摘要 4](#_Toc151906979)

[绪论 4](#_Toc151906980)

[1. 发展历史 4](#_Toc151906981)

[2. 现状 4](#_Toc151906982)

[3. 研究意义 5](#_Toc151906983)

[正文 5](#_Toc151906984)

[实验目的 5](#_Toc151906985)

[实验原理与方法 6](#_Toc151906986)

[实验原理 6](#_Toc151906987)

[实验方法 6](#_Toc151906988)

[实验内容及步骤 6](#_Toc151906989)

[实验结果分析及结论总结 7](#_Toc151906990)

[第一部分 7](#_Toc151906991)

[第二部分 8](#_Toc151906992)

[第三部分 8](#_Toc151906993)

[心得及展望 10](#_Toc151906994)

[心得 10](#_Toc151906995)

[展望 10](#_Toc151906996)

[附件 11](#_Toc151906997)

# **摘要**

本次实验完成了FIR数字滤波器的设计与分析,主要包括低通、高通、带通和带阻四种类型滤波器。根据设定的技术指标,采用计算机编程实现了滤波器参数计算、频率响应分析、结构实现以及对信号的滤波处理。

通过频率响应曲线,验证了所设计FIR滤波器的参数确实满足了设计指标;结构图展示了信号在滤波器内不同抽头之间的流动过程。最后,对测试信号应用FIR滤波器,通过波形和频谱比较,佐证了理论设计的准确性。ssss

本实验让我掌握了FIR滤波器设计方法,加深了频域分析理解,并体会到了数字信号处理理论与应用的联系。后续还会继续学习更复杂滤波器和DSP硬件的设计与实现。

# **绪论**

数字信号处理（DSP）是一门涉及数字信号的采集、处理和分析的领域，具有广泛的应用，包括通信、音频处理、图像处理、生物医学工程、雷达、控制系统等。以下是数字信号处理的发展历史、现状和研究意义的概述：

## 1. 发展历史

20世纪初，DSP的基础开始奠定，主要用于通信领域。

20世纪60年代，数字计算机的普及使DSP技术更加实际可行。

20世纪70年代和80年代，DSP在音频、图像处理和通信领域取得显著进展。

20世纪90年代，DSP应用领域不断扩展，包括生物医学、雷达、声纳、控制系统等。

21世纪以来，DSP在无线通信、多媒体处理、机器学习等领域得到广泛应用，技术不断演进。

## 2. 现状

- DSP技术已经深入到许多领域，如5G通信、自动驾驶、智能音箱、数字音乐、医学成像等。

- 近年来，DSP在人工智能和机器学习中的应用迅速增长，包括图像识别、语音识别、自然语言处理等。

- 嵌入式DSP处理器的性能不断提高，使得实时信号处理变得更加高效。

- DSP技术也与硬件加速器（如GPU和FPGA）结合，提供更大的计算能力。

## 3. 研究意义

改善信号质量：DSP可以用于降噪、滤波和增强信号，提高信息的质量和可用性。

实时处理：DSP可以在实时应用中处理信号，如通信系统、自动控制和传感器网络。

数据分析：DSP可以用于提取信号中的有用信息，帮助决策制定和模式识别。

通信：DSP在调制、解调、错误检测和纠正等方面对通信领域至关重要。

医疗应用：DSP用于医学成像、心电图分析、生物信号处理等，有助于医疗诊断和疾病监测。

人工智能：DSP与机器学习结合，推动了图像、语音和文本处理等领域的发展。

总的来说，数字信号处理在现代科技和工程领域扮演着至关重要的角色。它不仅帮助改善了现有技术的性能，还推动了新技术的发展，为人类社会的进步和创新提供了强大的工具。

# 正文

## 实验目的

1. 熟悉FIR数字滤波器的设计方法,能够根据技术指标,设计低通、高通、带通、带阻FIR滤波器。
2. 掌握利用计算机编程实现滤波器参数计算和频率响应分析的方法,并通过绘制幅频特性和相频特性曲线验证滤波器设计的正确性。
3. 通过跟踪数字滤波器的信号流程,了解并绘制出各类型线性相位FIR滤波器的结构图,理解其信号处理流程。
4. 应用所设计的FIR滤波器对信号进行滤波处理,并通过分析对比滤波前后效果验证滤波器的设计指标和实际滤波特性。
5. 在计算机上利用编程的方式实现FIR滤波器的设计、分析和应用,提高设计数字滤波器和信号分析处理的能力。
6. 通过完成一次完整的FIR滤波器设计、仿真和应用的过程,熟悉数字信号处理流程,为后续课程的学习和研究奠定基础。

## 实验原理与方法ss

### 实验原理

1. FIR滤波器的基本原理: 根据设计指标,通过选取滤波器的脉冲响应序列 $h(n)$来实现不同频率下的增益或衰减,从而达到滤波的目的。
2. 数字滤波器频率响应分析的方法:通过DFT可以将脉冲响应转换到频域,得到滤波器的传递函数在频域上的特性。
3. 线性相位FIR滤波器结构与信号流处理过程。

### 实验方法

1. 编程设计低通、高通、带通、带阻FIR滤波器,根据设计指标确定滤波器系数。
2. 分析滤波器的幅频特性和相频特性,绘制其频率响应。
3. 跟踪信号流,绘制不同FIR滤波器的流程图。
4. 应用所设计的FIR滤波器对实际信号进行滤波,比较滤波前后信号的变化。

## 实验内容及步骤

一、FIR滤波器设计

1. 根据技术指标,编写程序设计低通、高通、带通、带阻FIR滤波器,确定滤波器的脉冲响应序列。

二、FIR滤波器频率响应分析

1. 对FIR滤波器执行频率响应分析,得到幅频特性和相频特性。
2. 绘制幅频响应和相频响应曲线。

三、FIR滤波器结构绘制

1. 根据设计的4种FIR滤波器类型,分别绘出其信号流程结构图。

四、FIR滤波器应用

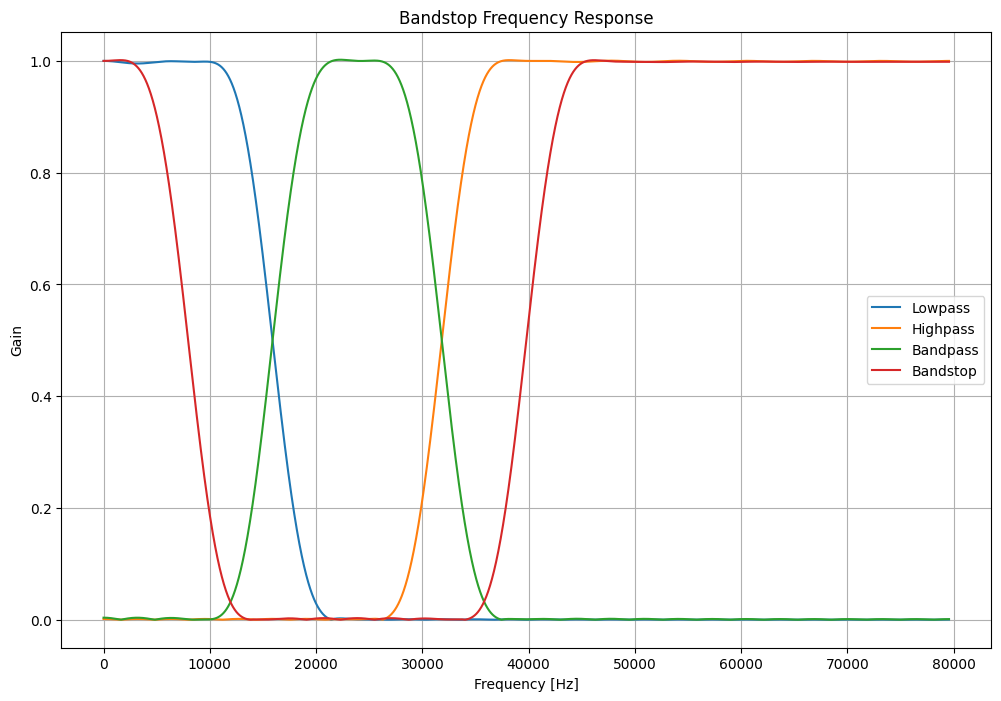
1. 采集或生成实验用的信号。
2. 对信号应用步骤一所设计的FIR滤波器,实施滤波处理。
3. 比较滤波前后信号的波形和频谱,分析滤波效果。

五、总结

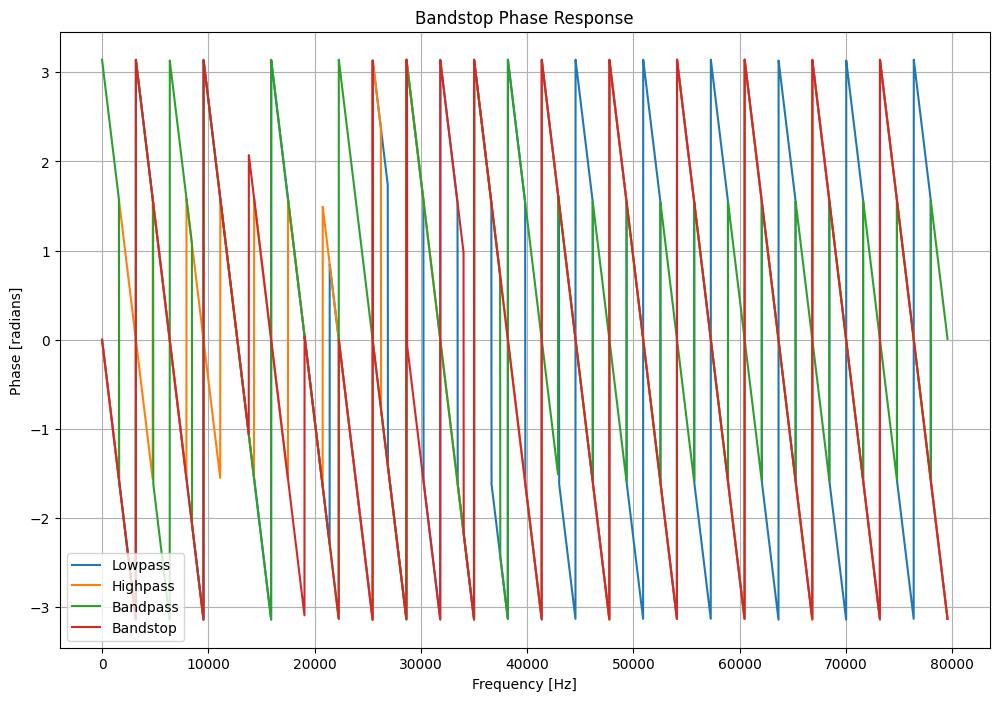
1. 总结实验过程,完成实验报告。

## 实验结果分析及结论总结

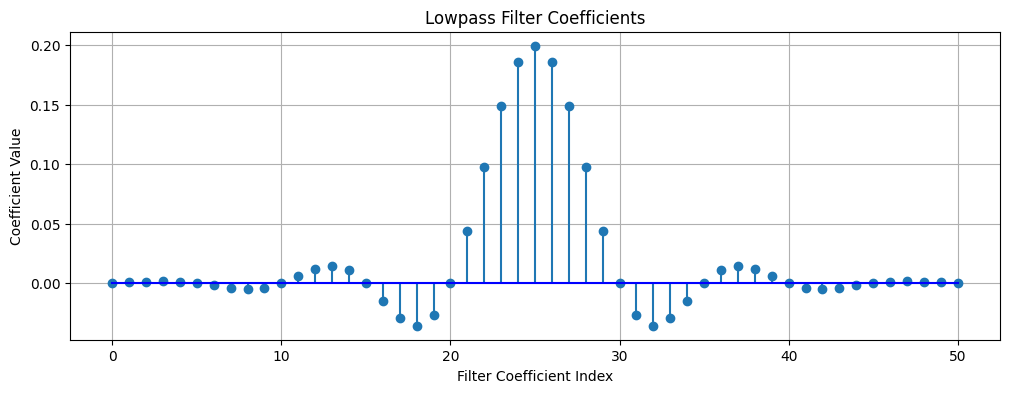
### 第一部分

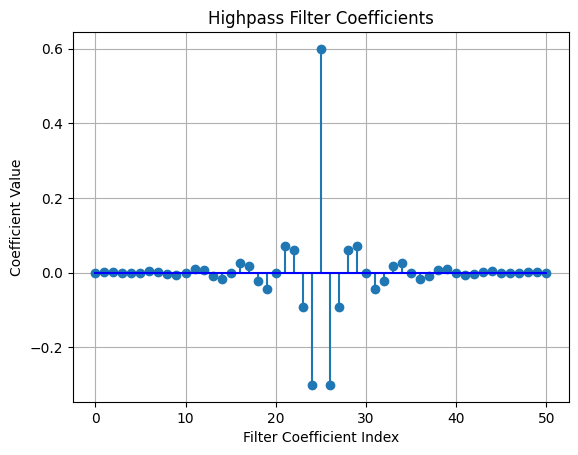


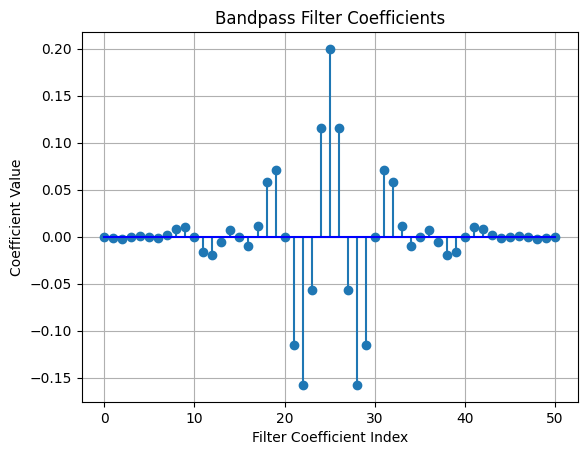
### 第二部分

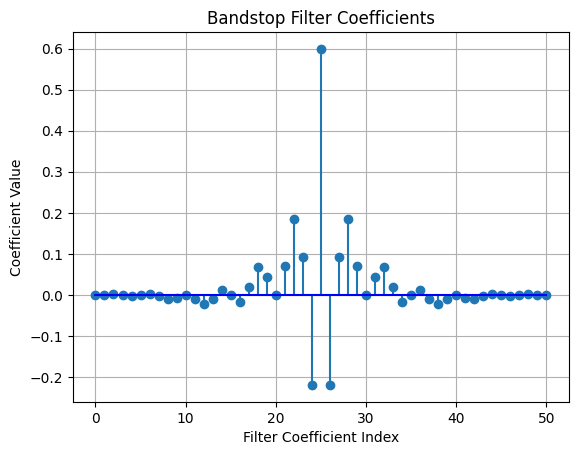


### 第三部分









## 心得及展望

### 心得

1. 通过本次实验,我掌握了FIR数字滤波器的设计方法,学会了编程实现滤波器参数的计算。
2. 频率响应分析让我直观理解到参数选择对滤波器频域特性的影响。
3. 绘出滤波器的结构流程图,加深了我对数字滤波器处理流程的理解。
4. 实际应用滤波器处理信号让我体会到理论运用于实际的重要性。

### 展望

1. 数字滤波器设计还有IIR、CIC、自适应滤波器等,我会继续学习扩展。
2. 更复杂的滤波器结构如并联、级联连接还需要掌握。
3. 函数发生器和oscilloscope的使用经验对我学后续模电实验有帮助。
4. DSP芯片中的固件滤波器实现也是我感兴趣想学习的方向。

## 附件

|  |
| --- |
| 代码 |
| import numpy as np  import matplotlib.pyplot as plt  from scipy.signal import freqz, firwin  # 指定滤波器的主要技术指标  fs = 1000  # 采样频率  cutoff\_low = 100  # 低通滤波器截止频率  cutoff\_high = 200  # 高通滤波器截止频率  cutoff\_band = [100, 200]  # 带通滤波器截止频率范围  cutoff\_stop = [50, 250]  # 带阻滤波器截止频率范围  filter\_order = 51  # 滤波器阶数 (odd)  # 设计FIR数字滤波器  lowpass\_filter = firwin(filter\_order, cutoff\_low, fs=fs, pass\_zero=True)  highpass\_filter = firwin(filter\_order, cutoff\_high, fs=fs, pass\_zero=False)  bandpass\_filter = firwin(filter\_order, cutoff\_band, fs=fs, pass\_zero=False)  bandstop\_filter = firwin(filter\_order, cutoff\_stop, fs=fs, pass\_zero=True)  # 绘制滤波器的幅频特性曲线  def plot\_filter\_response(filter\_coeff, title):      w, h = freqz(filter\_coeff, worN=8000, fs=fs)      plt.plot(0.5 \* fs \* w / np.pi, np.abs(h), label=title)      plt.title(title + ' Frequency Response')      plt.xlabel('Frequency [Hz]')      plt.ylabel('Gain')      plt.grid(True)      plt.legend(loc='best')  # 画出滤波器的幅频特性曲线  plt.figure(figsize=(12, 8))  plot\_filter\_response(lowpass\_filter, 'Lowpass')  plot\_filter\_response(highpass\_filter, 'Highpass')  plot\_filter\_response(bandpass\_filter, 'Bandpass')  plot\_filter\_response(bandstop\_filter, 'Bandstop')  plt.show()  # 画出滤波器的相频特性曲线  def plot\_phase\_response(filter\_coeff, title):      w, h = freqz(filter\_coeff, worN=8000, fs=fs)      plt.plot(0.5 \* fs \* w / np.pi, np.angle(h), label=title)      plt.title(title + ' Phase Response')      plt.xlabel('Frequency [Hz]')      plt.ylabel('Phase [radians]')      plt.grid(True)      plt.legend(loc='best')  plt.figure(figsize=(12, 8))  plot\_phase\_response(lowpass\_filter, 'Lowpass')  plot\_phase\_response(highpass\_filter, 'Highpass')  plot\_phase\_response(bandpass\_filter, 'Bandpass')  plot\_phase\_response(bandstop\_filter, 'Bandstop')  plt.show()  # 画出线性相位FIR数字滤波器的结构信号流图  def plot\_filter\_structure(filter\_coeff, title):      plt.stem(filter\_coeff, basefmt='b')      plt.title(title + ' Filter Coefficients')      plt.xlabel('Filter Coefficient Index')      plt.ylabel('Coefficient Value')      plt.grid(True)  plt.figure(figsize=(12, 4))  plot\_filter\_structure(lowpass\_filter, 'Lowpass')  plt.show()  plot\_filter\_structure(highpass\_filter, 'Highpass')  plt.show()  plot\_filter\_structure(bandpass\_filter, 'Bandpass')  plt.show()  plot\_filter\_structure(bandstop\_filter, 'Bandstop')  plt.show() |