ANALOG DEVICES

AN-334
应用笔记

One Technology Way • P.O. Box 9106 • Norwood, MA 02062-9106, U.S.A. • Tel: 781.329.4700 • Fax: 781.461.3113 • www.analog.com

数字信号处理技术

数字滤波

实时数字滤波是DSP最强大的工具之一。它最明显的优势是几乎消除了无源元件随时间和温度的波动、运算放大器漂移(有源滤波器)等造成的滤波器误差，除此之外，数字滤波器能够提供非常高的性能规格，如果采用模拟方案来实施，虽说不无可能，但要实现同样的性能将极其困难。而且，数字滤波器的特性可以通过软件控制轻松改变。因此，数字滤波器广泛用于自适应滤波应用，如调制解调器、数字音频、数字移动无线电和语音处理等。

设计数字滤波器的基本步骤与设计模拟滤波器相同。首先确定所需的滤波器响应特性，然后计算滤波器参数。传递函数和相位响应等特性的使用方式相同。

模拟滤波器与数字滤波器的主要差别在于：前者计算的是电阻、电容和电感值，而后者计算的是系数指。对于数字滤波器，数值取代了模拟滤波器的电阻和电容等物理元件。这些数值作为滤波器系数保存在存储器中，与来自ADC的数据值一起用于执行滤波计算。

数字滤波器实际上是一个离散函数，采用数字化数据而不是连续波形工作，每个采样周期获取一个数据点。

由于这种离散性质，我们可以用编号来引用数据样本，例如：样本1、样本2、样本3等。图7.1显示基本滤波功能：滤除一个低频信号中的高频噪声。此波形必须利用一个ADC数字化，产生样本x(n)。然后将这些数据值馈入数字滤波器，本例中为一个低通滤波器。输出数据样本为y(n)，DAC利用它重建一个模拟波形。

数字滤波

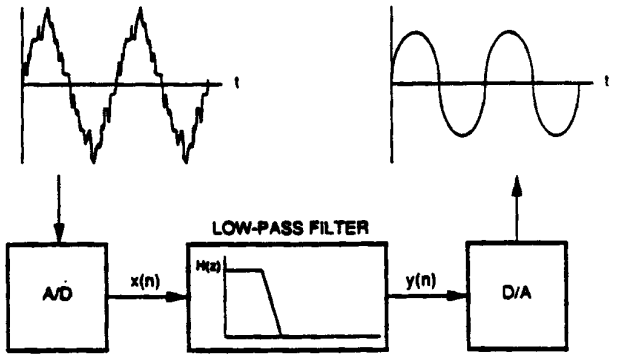

图7.1

然而，数字滤波器并不能满足所有信号处理滤波要求。为了维持实时操作，DSP处理器必须能够在一个采样时钟周期(1/f)内执行完滤波程序的所有步骤。因此，目前它的应用主要是在语音和音频带宽领域。不过，可以牺牲软件控制和灵活性，设计特殊硬件数字滤波器，以视频速度的采样速率工作。在另一些情况下，可以通过如下方法来克服速度限制：首先将高速ADC数据存储在缓冲存储器中，然后以与DSP数字滤波器速度兼容的速率读取缓冲存储器。这样就能维持伪实时操作，例如在雷达系统中，信号处理通常是对每发送一个脉冲后收集到的突发数据执行。即使在高过采样率的数据采样系统中，ADC之前和DAC之后通常也需要一个简单的模拟抗混叠滤波器。最后，当信号频率提高到足够高的程度，超过可用ADC的处理能力时，数字滤波也就无从谈起，因为我们没有ADC，数据采样系统不再成立。在极高频率下，由于运算放大器带宽和失真限制，有源模拟滤波是无法实现的，滤波要求必须通过纯无源器件来满足。下面将重点讨论可以在DSP程序控制下实时运行的滤波器。

数字滤波优势

- 高精度
- 高性能
- 线性相位、恒定群延迟(FIR滤波器)
- 无器件变化导致的漂移
- 灵活，可实现自适应滤波
- 易于仿真和设计

图7.2

数字滤波限制

- 计算必须在采样周期内完成
- 若要维持实时操作，则应用仅限于语音和音频带宽信号
- 视频频率处理需要硬连线的数字滤波器
- 仍然需要模拟滤波器：抗混叠和高频处理
- 缺少高速ADC进行采样

图7.3

有限脉冲响应(FIR)数字滤波器

最简单的数字滤波器是有限脉冲响应(FIR)滤波器，FIR滤波器的最基本形式是“移动平均”滤波器，如图7.4所示。

简单的移动平均FIR滤波器

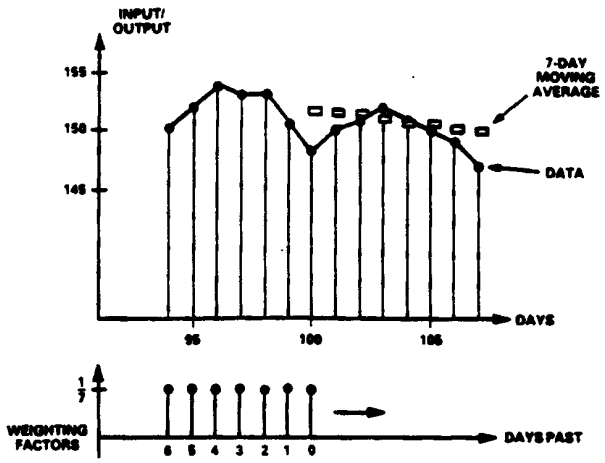

图7.4

图中显示了一名节食者体重的7天移动平均值和每日的体重。获得7天的数据样本之后，移动平均曲线上的第一点计算如下：将7个数据样本相加再除以7。观察分析该过程的另一种方法是将各数据样本乘以权重系数1/7，然后求和。为了获得移动平均曲线上的第二个点，从和值中减去第1个加权数据样本，然后加上第8个加权数据样本。这一过程持续进行，可以看作是对每日读数的非常粗糙的低通滤波。该过程的数字实现如图7.5所示，其中显示了各种乘法、延迟和求和。之所以得名为有限脉冲响应(FIR)滤波器，是因为脉冲响应的持续时间是有限的，也就是说：在7个零值输入样本后，滤波器输出变为0。处理实际的电信号时，移动平均曲线类似于图7.6。

FIR滤波器的数字形式

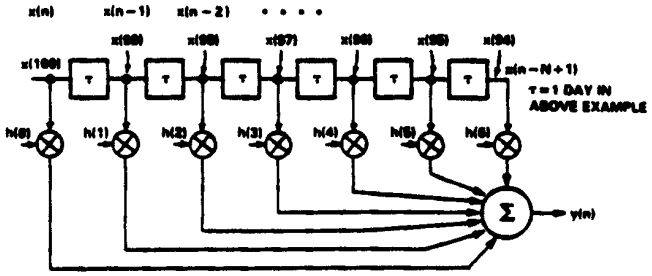

图7.5

应用于模拟信号的移动平均FIR滤波器

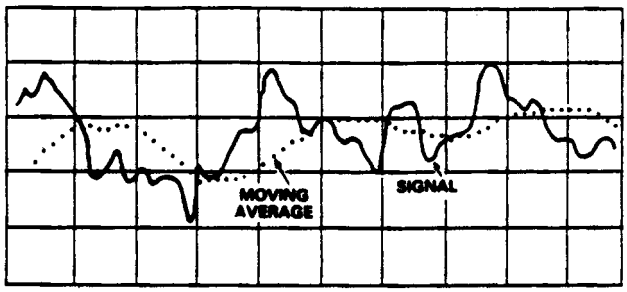

图7.6

从数学角度看，移动平均滤波器就是滤波器脉冲响应h(t)与采样数据点x(t)的"卷积"，结果为输出y(t)，如图7.7所示。对于线性卷积，运算涉及到将x(t)乘以h(t)的逆转且线性偏移版本，然后对乘积中的值求和。

移动平均系数与采样波形的卷积

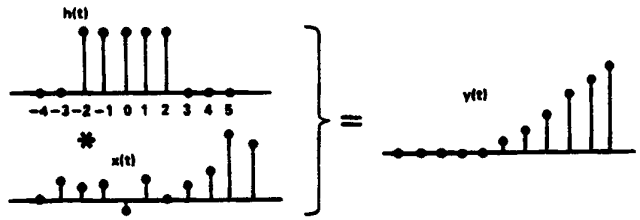

图7.7

针对不同的抽头数N，移动平均滤波器的sin(x)/x频率响应如图7.8所示。(注：本部分中，N指的是采样点数，而不是ADC或DAC的分辨率位数！)注意，提高抽头数可以使移动平均滤波器的滚降特性发生更急剧的变化，但无法减少不需要的旁瓣。

移动平均系数与采样波形的卷积

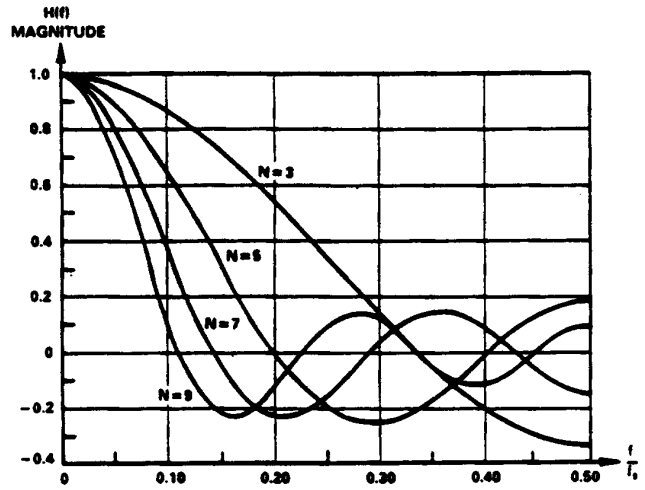

图7.8

通过适当地选择各个权重或系数，而不是给予同样的权重，可以大幅改善简单FIR移动平均滤波器的性能。滚降的急剧变化程度可以通过增加更多级(抽头)来改善，阻带衰减特性可以通过适当地选择滤波器系数来改善。FIR滤波器设计的本质在于选择合适的滤波器系数和抽头数，从而实现所需的传递函数H(f)。有多种算法可将频率响应H(f)转换成一组FIR系数，绝大部分此类软件可以在市场上获得，并且可以在PC上运行。FIR滤波器设计的关键法则是：FIR滤波器的系数h(n)等于频率传递函数H(f)的脉冲响应的量化值。反之，脉冲响应为H(f)的傅里叶变换。

决定FIR滤波器传递函数H(f)的因数

- **抽头数**
- **选择适当的加权滤波器系数**

图7.9

时域和频域的对偶性

不妨暂时岔开话题，讨论时域与频域的关系，从而更好地了解FIR等数字滤波器背后的原理。在数据采样系统中，卷积运算可以通过一系列乘法和累加来实现。时域/频域中的卷积运算相当于频域/时域中的点与点乘法。例如，时域中的卷积相当于频域中的乘法，如图7.10所示。可以看出，频域中的滤波可以通过如下方式完成：将通带中的所有频率成分乘以1，并将阻带中的所有频率成分乘以0。反之，频域中的卷积相当于时域中的点与点乘法。

时域和频域的对偶性

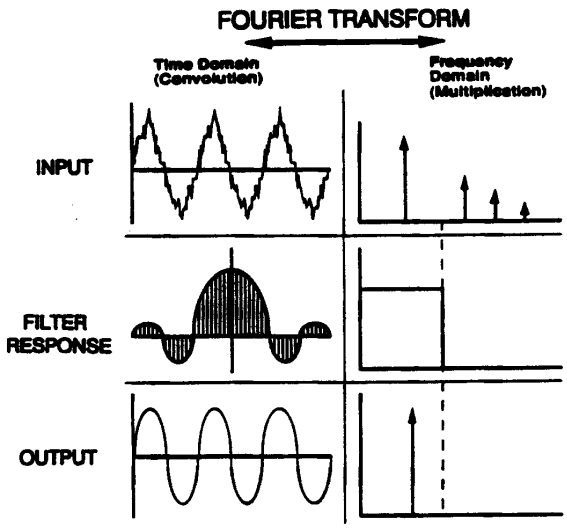

图7.10

频域(1或0)中的传递函数可以通过傅里叶变换转换到时域，这种变换会在时域中产生一个脉冲响应。由于频域中的乘法(信号频谱乘以传递函数)相当于时域中的卷积(信号与脉冲响应卷积)，因此可以将信号与脉冲响应卷积，从而对信号进行滤波。FIR滤波正是这样一个过程。由于是数据采样系统，因此信号和脉冲响应经过时间和幅度量化，产生离散样本。构成脉冲响应的离散样本就是FIR滤波器系数。

滤波器设计(模拟或数字)涉及到的数学必定会用到变换。在连续时间系统中，拉普拉斯变换可以看作是傅里叶变换的一般情形。同样，可以对离散时间数据采样系统的傅里叶变换进行一般化处理，得到所谓z变换。在数字滤波器设计中使用z变换的详情参见参考文献1、2和3。

使用循环缓冲在DSP硬件中实现FIR滤波器

如上所述，FIR滤波器(如图7.11所示)必须执行如下卷积运算：

$$y(n) = h(n)*x(n) = \sum_{i=0}^{N-1} h(i)x(n-i)$$

其中，h(i)为滤波器系数阵列，x(n-i)为滤波器的输入数据阵列。公式中的N表示滤波器的抽头数，与滤波器的性能有关，如上文所述。

直接形式的FIR滤波器

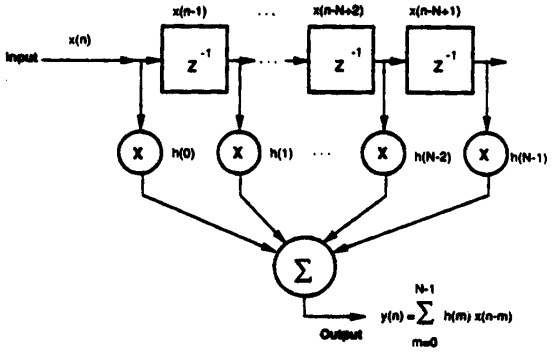

图7.11

在一系列FIR滤波器公式中，N系数位置始终是按顺序访问，从h(0)到h(N-1)。相关的数据点循环通过存储器；每次计算一个滤波器输出后，新增样本便取代最旧的样本。可以使用一个固定边界的RAM来实现这种循环缓冲机制，图7.12显示的是一个4抽头FIR滤波器的情况。

4抽头FIR滤波器的数据存储器寻址

$$y(n) = h(n) * x(n) = \sum_{i=0}^{N-1} h(i)\,x(n-i)$$

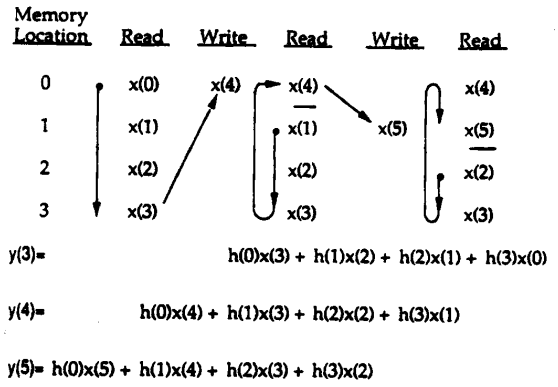

图7.12

每次卷积后，最新数据样本便会取代最旧数据样本。RAM中保存四个最新数据样本的"时间历史"。

如果将新数据值写入存储器，覆盖最旧的值，那么可以在DSP芯片的固定边界RAM中实现这一延迟线。为方便存储器寻址，从存储器读取旧数据值时，应从刚写入值的位置之后的一个位置开始。例如，假设将x(4)写入存储器位置0，则随后应从位置1、2、3、0读取数据值。可以将这个例子推广以支持任意抽头数。以这种方式寻址数据存储器位置时，地址发生器只需要按顺序提供地址，而不用考虑是寄存器读操作还是寄存器写操作。到达最后一个位置时，存储器指针必须复位到缓冲器的起始位置，因此这种数据存储缓冲器是循环的。

系数与数据同时取出。所选的寻址方案决定了最旧的数据样本必须最先取出。因此，最后一个系数必须最先取出。系数可以反向存储在存储器中：h(N-1)是第一个位置，h(0)是最后一个位置，地址发生器提供递增地址。或者，系数也可以正常方式存储，而对系数的访问则是从缓冲器的末端开始，地址发生器递减。在图7.12所示的例子中，系数以逆序存储。

FIR滤波器设计技术

FIR滤波器设计要求指定一组有限数量(N)的系数h(n)，用以模拟一个理想的滤波器。时域中的滤波器系数h(n)对应于滤波器传递函数H(f)的脉冲响应。

FIR滤波器设计关键原则

- **FIR滤波器的系数h(n)等于频率传递函数H(f)的脉冲响应的量化值。**
- **通过对H(f)进行傅里叶变换来计算脉冲响应。**

图7.13

图7.14比较了两种滤波器的传递函数：一是针对1dB带内纹波而优化的二阶、四阶、六阶理想切比雪夫低通滤波器，一是针对0.002dB通带纹波而优化的91抽头(即91个系数和91个顺序循环缓冲存储器位置)数字FIR滤波器。几乎没有与之对应的模拟滤波器，模拟硬件无法实现如此高的阶数(通过经验近似法可知它高于70极)。通带内的响应更平坦，再现的信号更忠实于原貌，通带内的相位失真可忽略不计，因为滤波器将所有频率成分均等延迟。这是FIR滤波器的另一个重要特性(线性相位响应和恒定群延迟)，对于数字音频应用极具吸引力。

91抽头FIR滤波器响应与切比雪夫模拟滤波器响应的对比

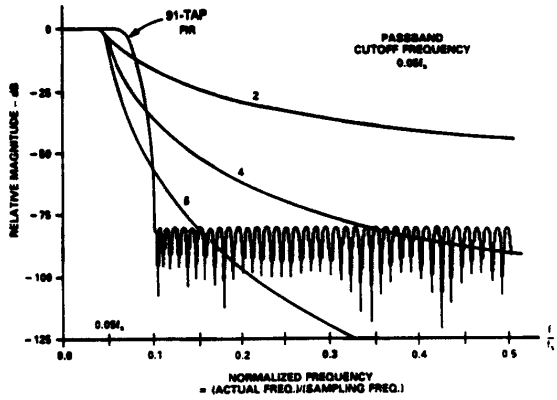

图7.14

如果在微电脑ADSP-2101中实现图7.14所示的91抽头FIR滤波器，则每个抽头需要一个处理器周期(80ns)，总处理时间为7.3μs。这意味着最高可以实现大约136kHz的采样速率，同时仍能维持实时操作。

91抽头FIR滤波器性能特征

- **通带纹波：0.002dB**
- **线性相位**
- **阻带衰减：80dB**
- **利用ADSP-2101处理器时，可以实现138kHz的采样速率（每个滤波器抽头需要80ns的周期时间）**
- **无对应的模拟滤波器(要求70个极点！)**

图7.15

使用CAD技术设计FIR滤波器

在实际应用中，上文讨论的原理已通过简单易用、可在大多数PC上运行的CAD程序实现。用户只需要指定所需的FIR滤波器特性(采样频率、通带频率、阻带频率、通带纹波和阻带衰减)，如图7.16所示。CAD程序计算所需的滤波器抽头数(N)、脉冲响应和滤波器系数。

滤波器设计关键参数

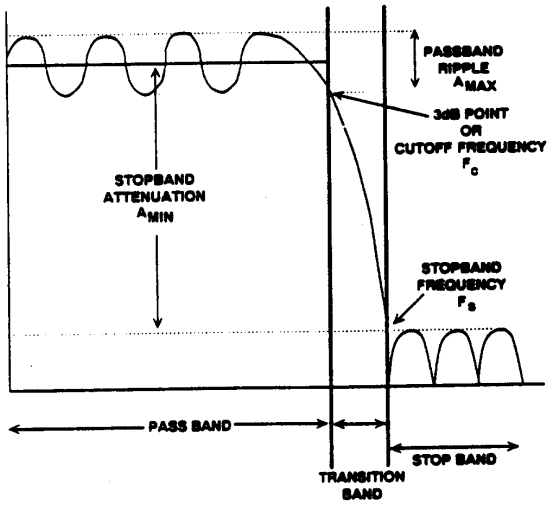

图7.16

FIR滤波器设计CAD程序输入

- **通带**
- **通带纹波**
- **阻带**
- **阻带衰减**
- **字长，即16位定点**

图7.17

输出还会提供频率响应H(f)、脉冲响应和阶跃函数响应的曲线。如果响应特性满足要求，就可以将滤波器系数下载到DSP处理器中。CAD程序还能模拟有限字长(即以16位定点算法执行计算)对传递函数的影响。

91抽头FIR滤波器性能特征

- **频率响应曲线，显示有限字长算法的影响**
- **脉冲响应曲线**
- **阶跃函数响应**
- **所需的抽头数**
- **滤波器系数**

图7.18

对于CAD滤波器设计，还开发了其它算法，可以针对不同的特性优化滤波器性能。Parks-McClellan程序(参考文献1)就是一例，它利用近似理论的雷米兹交换算法，将所需特性与实际特性之间的最大误差降至最小。

使用CAD程序设计FIR数字音频滤波器的示例

本例中，我们将设计一个音频低通滤波器，其设计采样速率为44.1kHz(CD播放器的标准工作频率)。所用的程序来自Momentum Data Systems公司(参考文献5)。该程序为菜单驱动型，与IBM PC兼容。该滤波器将实现为图7.19所示的直接形式FIR滤波器。

直接形式的FIR滤波器

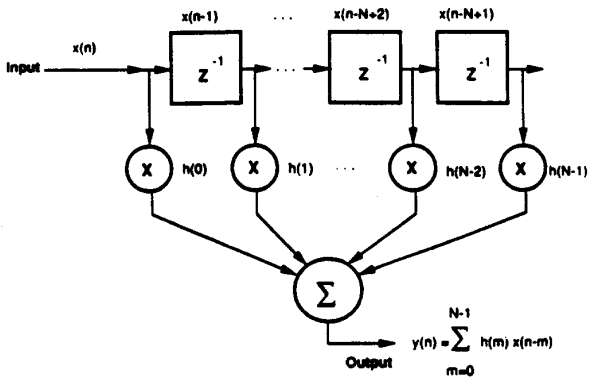

图7.19

首先，我们从图7.20所示的主菜单选择要设计的滤波器类型。我想选择的是“等纹波FIR设计(Parks-McClellan)”。

滤波器设计与分析系统主菜单(屏幕1)

- **IIR滤波器设计**
- **带窗口的FIR滤波器设计**
- **等纹波FIR设计(Parks-McClellan)**
- **读取滤波器规格文件**
- **系统分析(Z域输入)**
- **系统分析(s域输入)**
- **读取系统分析输入文件**
- **设置系统默认值**
- **退出到DOS**

图7.20

然后出现第二个屏幕，如图7.21所示。此屏幕用于选择FIR滤波器的类型(低通、高通、带通等)，以及指定频率模式、增益模式、是否使用sin(x)/x补偿。

有限脉冲响应滤波器设计菜单(屏幕2)

滤波器类型：	**1-低通**
	2-高通
	3-带通
	4-阻带
	5-差分器
	6-多带
频率模式：	**H-赫兹**
	R-弧度/秒
增益规格模式：	**1-最大增益 1.0**
	2-正常增益 1.0
滤波器补偿：	**输入X选择**

图7.21

下一个屏幕如图7.22所示，可输入采样速率、频段边沿以及关于通带纹波和阻带衰减的规格要求。我们选择低通滤波器，截止频率18 kHz。

FIR滤波器设计低通滤波器(屏幕3)

采样频率：	**44100.0**
通带频率：	**18000.0**
阻带频率：	**21000.0**
通带纹波：	**1.00000E.02**
阻带纹波：	**(衰减)96dB**

图7.22

然后，程序会计算所需的滤波器系数。完成计算后，出现图7.23所示的屏幕，告诉我们实现该滤波器需要多少个系数(抽头数)。如果抽头数与DSP处理器的吞吐速率和采样速率兼容，用户就可以让程序继续执行。

FIR设计示例(屏幕4)

估算的FIR滤波器抽头数：	**69**
输入所需的抽头数：	**69**

图7.23

如果在微电脑ADSP-2101中实现69抽头FIR滤波器，则每个抽头需要一个处理器周期(80ns)，因此总处理时间为5.5 µs。这意味着最高可以实现大约182 kHz的采样速率，同时仍能维持实时操作。

针对69抽头FIR滤波器的ADSP-2101处理器时间

- **每抽头80ns(一个处理器周期)**
- **69个抽头**
- **5.5µs处理时间(80ns × 69)**
- **182kHz采样速率，同时支持实时操作**

图7.24

下一步如图7.25所示，就是将系数量化为正确的位数，使系数与所用的DSP处理器兼容。本例使用ADSP-2101，它是一款16位定点处理器，因此系数量化为16位。

FIR设计示例(屏幕5)

选择量化所需的位数

位数(8至32)： 16

图7.25

注意，系数经过计算和正确量化后，我们必须知道量化过程对滤波器性能的影响。滤波器设计程序最初使用非常高的分辨率计算系数。当这些高精度系数被量化为较低分辨率的16位数时，会有一定的精度损失。这种精度损失可能会对滤波器的性能产生不利影响。为了验证性能合理，应当进行滤波器仿真。本例中，仿真利用16位数学形式执行。图7.26显示了仿真滤波器的响应，以便分析滤波器性能。输出还提供脉冲响应(如图7.27所示)和阶跃响应(如图7.28所示)。很显然，此滤波器没有对应的模拟滤波器。利用经验法则计算具有这种过渡带特性(18 - 21 kHz；85 dB)的模拟滤波器所需的极点数可知，滤波器阶数为65！(参考第三部分)

FIR滤波器设计示例频率响应

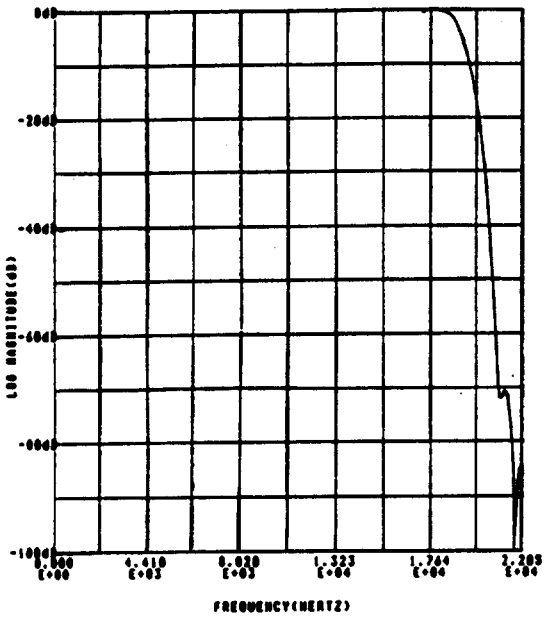

图7.26

如果滤波器性能满足要求，就可以将系数文件下载到DSP硬件以便实现滤波器。如果响应不能满足要求，可以更改抽头数或其他参数并重新执行设计过程，直到实现所需的响应为止。

FIR滤波器设计示例脉冲响应

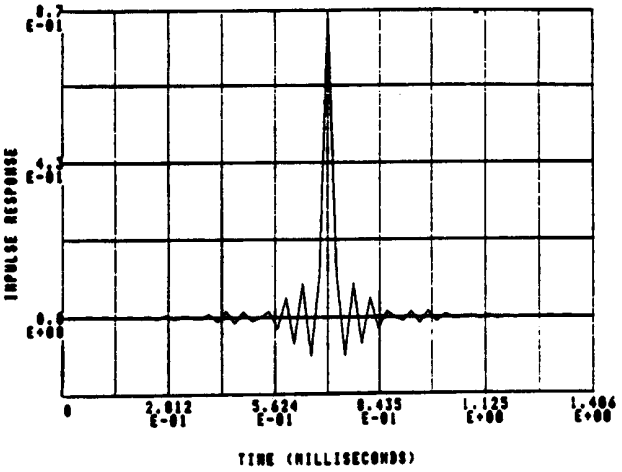

图7.27

FIR滤波器设计示例阶跃响应

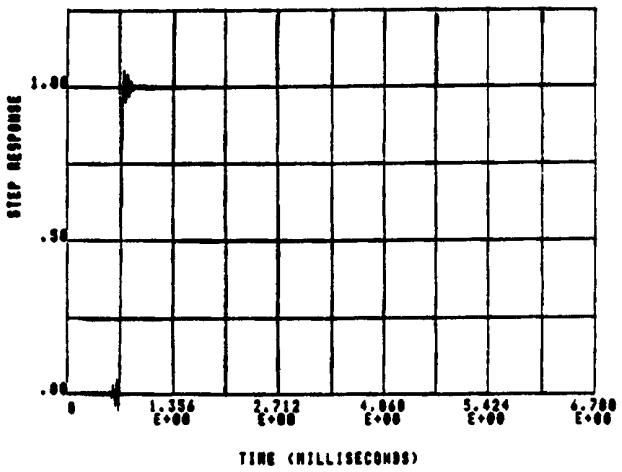

图7.28

确保FIR滤波器的线性相位

FIR滤波器的一大优势是始终具有线性相位响应特性，因而对音频和声纳应用极具吸引力。线性相位意味着所有输入频率都被滤波器延迟相同的量。在FIR滤波器中，这是信号通过N个抽头所需的时间。对于一个频率带，此延迟常被称作“群延迟”。线性相位FIR滤波器的群延迟恒定不变。

为了确保FIR滤波器的线性相位特性，要求滤波器系数对称，例如，图7.29所示的一个简单低通滤波器和图7.30所示的一个简单高通滤波器就是如此。此外，线性相位还要求使用奇数数量的抽头。

对称的滤波器系数产生线性相位响应
——低通滤波器

图7.29

对称的滤波器系数产生线性相位响应
——高通滤波器

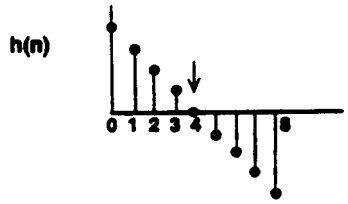

图7.30

使用FIR滤波器进行抽取处理

FIR滤波器可以用于需要数据速率抽取的应用，例如过采样Σ-Δ型ADC。假设我们想抽取一个FIR滤波器的输出数据速率，抽取系数为2，那么我们将隔一个采样点从滤波器获取一个采样点。这同时意味着，滤波器输出计算只需每隔一个采样时钟周期执行。换言之，DSP处理器现在有两个采样时钟间隔来完成卷积计算。这意味着可以使用更多的滤波器抽头，或者使用较慢的处理器。

FIR滤波器属性总结

- **始终稳定**
- **具有线性相位、恒定的群延迟**
- **可自适应**
- **低舍入噪声**
- **抽取输出时具有计算优势**
- **易于理解和实现**

图7.31

无限脉冲响应(IIR)数字滤波器

如上文所述，数字FIR滤波器在现实中没有与之对应的模拟滤波器，最接近的类比是加权移动平均。此外，FIR滤波器只有零点，没有极点。IIR滤波器则不同，它具有对应的传统模拟滤波器(巴特沃兹、切比雪夫和椭圆滤波器)，并且可以利用我们更熟悉的传统滤波器设计技术进行分析和合成。

图7.32显示了一个二阶低通有源滤波器，其对应的IIR数字滤波器如图7.33所示。该二阶IIR滤波器称为“双二阶滤波器”(因为它是利用z域中的一个双二阶方程式来描述)，是更高阶IIR设计的基本构建模块。图中还给出了描述具有5个系数的该滤波器特性的差分方程。

二阶模拟滤波器实现

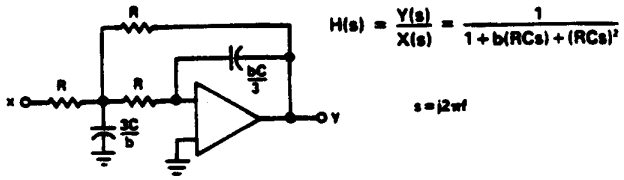

图7.32

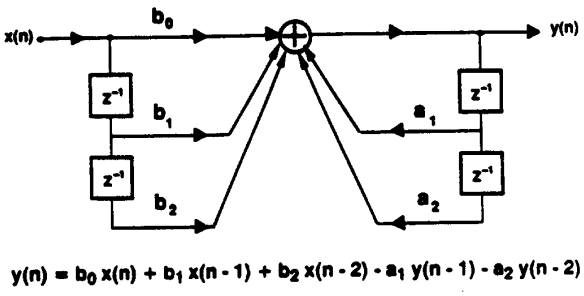

图7.33

数字滤波器的一般方程如图7.34所示，由此可导出一般传递函数H(z)，其分子和分母均包含多项式。分母的根决定滤波器的极点位置，分子的根决定零点位置。虽然可以直接根据此方程式构建一个高阶IIR滤波器(称为“直接形式”实现)，但量化误差(有限字长算法)引起的累计误差可能引起不稳定和较大的误差。因此，常常是级联几个具有适当系数的双二阶部分，而不是使用直接形式实现。双二阶部分可以独立缩放，然后级联起来，从而使系数量化和递归累计误差最小。级联双二阶滤波器比相应的直接形式滤波器执行得更慢，但更稳定，并且有限算法误差引起的误差影响降至最小。计算特定DSP IIR滤波器的吞吐时间时，应当检查双二阶滤波器部分的基准性能规格。对于ADSP-2101，单个双二阶部分的执行时间为560ns，对应于七个指令周期。

滤波器一般方程式

$$y(n) = \underbrace{\sum_{k=0}^{M} b_k x(n-k)}_{\text{FEEDFORWARD}} + \underbrace{\sum_{k=1}^{N} a_k y(n-k)}_{\text{FEEDBACK}}$$

GIVES RISE TO THE TRANSFER FUNCTION

$$H(z) = \frac{\sum_{k=0}^{M} b_k z^{-k} \quad (\text{ZEROS})}{1 - \sum_{k=1}^{N} a_k z^{-k} \quad (\text{POLES})}$$

图7.34

IIR滤波器属性总结

- **反馈(递归)**
- **可能不稳定**
- **通常实现为级联双二阶滤波器，而不是直接形式滤波器**
- **非线性相位**
- **效率高于FIR滤波器**
- **抽取输出时无计算优势**
- **具有类似的模拟滤波器**

图7.35

IIR滤波器的吞吐考虑

- **确定实现所需滤波功能需要多少双二阶部分**
- **乘以每个双二阶部分的执行时间(ADSP-2101为560ns)**
- **结果为实时操作允许的最短采样时间($1/f_S$)**

图7.36

总结：FIR滤波器与IIR滤波器

采用FIR还是IIR滤波器设计，有时可能难以抉择，下面给出几条基本原则以供参考。一般来说，IIR滤波器需要的存储器和乘法运算更少，因而效率高于FIR滤波器。IIR滤波器可以根据以前的模拟滤波器设计经验来设计。IIR滤波器可能会出现不稳定问题，但如果通过级联二阶系统来设计更高阶滤波器，则发生这一问题的可能性大大降低。

另一方面，对于给定的截止频率响应，FIR滤波器需要更多抽头和运算，但具有线性相位特性。FIR滤波器采用有限的数据历史工作，如果某些数据损坏(例如ADC闪烁码)，则FIR滤波器仅针对N-1个样本发生响铃振荡。然而，因为反馈，IIR滤波器的响铃振荡时间将显著延长。

如果要求陡峭的截止频率并且处理时间至关重要，则IIR椭圆滤波器是合适之选。如果乘法运算量不是非常大，并且要求线性相位，则应选择FIR滤波器。

IIR与FIR滤波器

IIR滤波器	FIR滤波器
效率较高	效率较低
有对应的模拟滤波器	无对应的模拟滤波器
可能不稳定	始终稳定
非线性相位响应	线性相位响应
对毛刺的响铃振荡较大	对毛刺的响铃振荡较小
CAD设计包可用	CAD设计包可用
无法通过抽取提高效率	可以通过抽取提高效率

图7.37

快速傅里叶变换

许多应用要求在频域中处理或分析信号。在模拟领域，利用模拟频谱分析仪可以轻松完成这一任务。从数学角度来看，这一过程可以通过求取连续时间模拟信号的傅里叶变换来复制。傅里叶变换产生模拟信号的频谱成分。但是，在数据采样系统中，这一过程必须通过DSP对ADC输出数据的处理来完成。此外，模拟频谱分析与数字频谱分析有两个截然不同之处。第一，ADC的输出是连续输入x(t)的离散量化样本。在数据采样系统中，离散傅里叶变换(DFT)执行时域样本到频域的变换。此外，DFT必须采用有限数量的数据采样点工作，而傅里叶变换则是处理连续波形。

连续和离散时间到频率变换

- **傅里叶变换处理连续时间波形**
- **离散傅里叶变换处理波形的有限数量离散时间样本**

图7.38

如果x(n)是N个输入数据样本的序列，则DFT产生一个在频率上均匀隔开的N样本序列X(k)。DFT由一系列乘法和加法组成，数据字乘以一个正弦值，然后将若干个此类乘积相加，如图7.39所示。

离散傅里叶变换(DFT)方程

$$X(k) = \sum_{n=0}^{N-1} x(n)e^{-j2\pi nk/N}, \text{ where}$$

$$e^{-j2\pi nk/N} = \cos(2\pi nk/N) - j\sin(2\pi nk/N)$$

图7.39

DFT可以看作是输入信号与许多正弦值的相关或比较，用以评估输入信号的频率成分。例如，1024点DFT需要输入信号的1024个样本和一个正弦波的1024点。采用在$-f_S/2$到$+f_S/2$范围内均匀隔开的1024个不同频率的正弦波。每次DFT均会对照输入信号检查正弦波，确定输入信号中存在多少该频率。对于1024个频率，均要重复执行这一过程。结果如图7.40所示，其中输出频谱中出现N/2个离散频率成分。如果采样频率为f_S，则频谱线之间的间隔为f_S/N或$1/Nt_S$，其中t_S为采样周期$1/f_S$。

频谱分析常常是针对复信号(具有实部和虚部)执行，从而获得相位、幅度和频率信息。在上例中，1024个复数数据值与1024个复数正弦值相乘并累加，这就需要1024次复数乘法运算。对于每个频率都要重复这一过程，总乘法次数为1024^2；一般意义上，需要N^2次复数乘法运算。即使对一个功能强大的DSP器件，这一运算次数也是相当庞大的，并且颇费时间。只有需要计算所有输出频率时，才会有如此大的运算量。如果只需要确定一个或几个频率的频率成分值，则运算量并不大。

不同记录长度的典型FFT输出

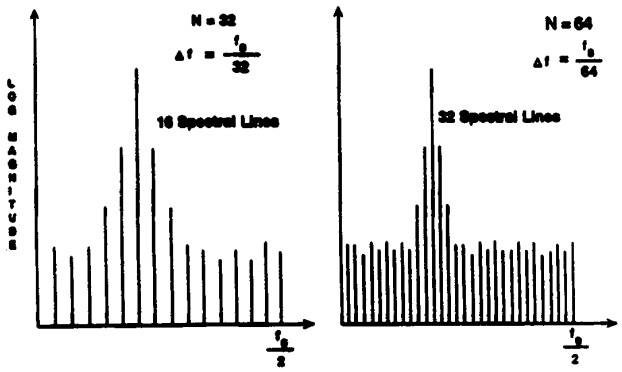

图7.40

然而，在大多数频谱分析情况下，要求计算最高为$f_S/2$的整个频谱，因此我们必须找到一种更快的办法。FFT是一种通过减少所需的乘法和累加次数，从而加速DFT计算的算法。它由J. W. Cooley在20世纪60年代推广开来，实际上是对Runge(1903)和Danielson、Lanczos (1942)等人思想的重新发现，最初出现在没有计算机和计算器的时期，那时数值计算依靠人工完成，费时费力。

FFT利用了DFT运算过程中的某些代数和三角对称性。例如，如果执行1024点DFT，则需要10242(1,048.567)次复数乘法。可以将1024点DFT分解为两个512点DFT，结果相同。这称为"抽取"。每个512点DFT需要5122(262,144)次复数乘法，总共需要524,288次复数乘法。与原来的1,048,567次乘法相比，这已经大大减少。图7.41显示一个N点DFT分解为两个N/2点DFT。水平线上有一个相位系数W(有时称之为"转动系数")，表示要乘以W。箭头与水平线相交的点表示求和。线上的-1表示符号反向。

8点DFT的时间上的第一次抽取

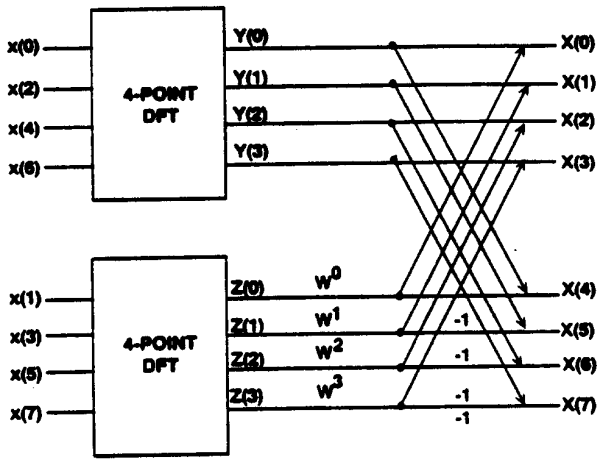

图7.41

既然可以将1024点DFT分解为两个512点DFT并得到同样的结果，那么为何不将各512点DFT分解为两个256点DFT，从而进一步减少运算量呢？确实可以这样做。这一抽取过程可以一直进行下去，直到原始DFT被分解为2点DFT(最小的DFT)。

8点时域抽取FFT
正常顺序输入、位反转输出

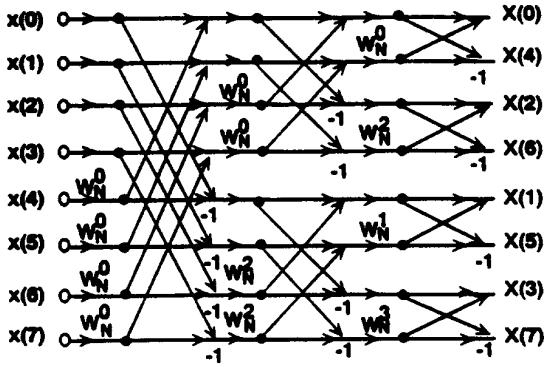

图7.42

抽取过程完成后，最终的运算系列就是FFT。这一过程如图7.42的8点DFT所示。由于该FFT的第一个抽取系数为2，因此称之为基2 FFT。如果初始DFT的抽取系数为4，则称为基4 FFT。注意，输入数据点是以正常顺序获得，但输出为位反转顺序。因此，位反转硬件在ADSP-2101等DSP处理器中很常见。基本计算本质上是2点DFT，常被称为"蝴蝶"计算。FFT由许多蝴蝶计算组成。图7.43显示了基2时域抽取FFT的基本蝴蝶计算，每次蝴蝶计算需要一次复数乘法运算。

基2时域抽取FFT蝴蝶计算

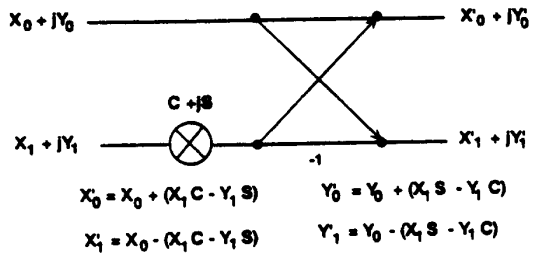

图7.43

FFT对减少DFT所需运算量的意义如图7.44所示。

N点FFT的计算效率

DFT	FFT
N^2次乘法	$(N/2 \log_2(N)$
对于N = 1024	对于N = 1024
1,048,578次乘法	5,120次乘法

200:1

图7.44

注意，FFT会计算所有N/2个频谱输出(全有或全无！)。如果只需要计算几个频谱点，则DFT的效率更高。使用DFT计算单个频谱输出只需要N次复数乘法。

FFT硬件实现

一般意义上，N点FFT的存储器要求是：N个位置用于实数数据，N个位置用于虚数数据，N个位置用于正弦数据(有时称为FFT系数或转动系数)。只要满足存储器要求，DSP处理器就必须在所需的时间内执行必要的计算。许多DSP供应商会给出针对具体FFT大小的性能基准或一次蝴蝶计算的时间。比较FFT规格时，必须确保所有情况下都使用相同类型的FFT。例如，1024点FFT基准可以从基2或基4 FFT获得；因为所需的运算次数不同，所以这些基准是不兼容的。

一旦基本硬件要求得到满足，剩下的工作就是通过软件来实现系统。硬件相同时，不同的软件程序可以不同的方式处理数据，从而实现不同的算法，例如：基2、基4、时域抽取或频域抽取。参考文献6给出了一个优化的基4 FFT算法。

DSP FFT硬件基准比较

- **基2、基4 FFT?**
- **蝴蝶计算执行时间?**
- **总FFT执行时间?**

图7.45

FFT设计考虑

设计FFT的第一步是确定所需的点数N或记录长度。解决这一问题有几种方法。采样速率f_S至少必须是目标最大输入信号频率的两倍。一旦知道采样速率，就可以由f_S/N得出FFT的频谱分辨率。FFT中的点数越多，则频谱分辨率越高。在频谱分析应用中，这是首要考虑。

例如，在实时语音分析中，信号带宽约为4kHz，意味着采样速率为8 kHz。语音的频谱不是固定不变的。信号必须分为许多窗口T_W，其长度足够短，以确保各自的特性不会因为FFT中的平均值计算而消失。在语音的长期FFT中，所有意义都会丢失。另一方面，T_W必须足够长，以提供适当的频谱分辨率。现已确定，对于人类语音现象，20ms较为适当，因此T_W = 20ms。

实时语音分析FFT示例

- **带宽 = 4kHz，采样速率 = 8kHz**
- **窗口 = 20ms**
- **N > 8kHz × 20ms = 160，因此使用N = 256**
- **处理器能否支持?**
- **ADSP-2101在N = 256时的基准性能为0.59ms**
- **可以！还有19.41ms用于其他计算**

图7.46

如何判断处理器能否支持FFT呢？窗口T_W中的采样点数等于$T_W f_S$，即20ms × 8kHz = 160点。将其舍入到最接近的2的幂，即256点。这意味着，DSP处理器必须在小于T_W(每个窗口的数据采集时间)的时间内完成256点FFT，否则就无法实现实时处理，计算必须离线进行。ADSP-2101可以在0.59 ms内完成256点FFT，留下19.41 ms用于其他计算。

大多数DSP处理器的基准FFT处理时间由制造商提供。图7.48显示了ADSP-2101的基4基准时间。该512点基准时间是相对于基2 FFT而言。评估不同DSP处理器时，必须在相同条件下进行比较。例如，基4 FFT会比基2 FFT稍快些。

图7.47还显示了与FFT执行时间相关的实时操作的最大采样速率。这些采样速率说明，ADSP-2101等现代DSP微电脑能够对带宽高达100 – 200 kHz的信号进行实时FFT分析。

ADSP-2101基准FFT性能和相关的实时操作采样速率

FFT大小	执行时间	最大采样速率
256	0.59ms	434kHz
512	1.3ms	394kHz
1024	2.9ms	353kHz
2048	6.5ms	315kHz
4096	14.2ms	258kHz

图7.47

窗口包含整数周期的正弦波的FFT

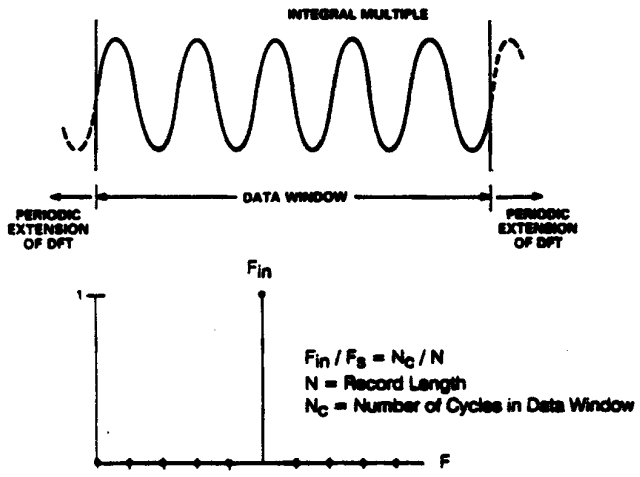

图7.48

频谱泄漏和窗口

了解FFT处理中的频谱泄漏的最佳方法是考虑对一个纯正弦波输入执行FFT的情况。考虑两种状况。在图7.48中，采样速率与输入正弦波频率的比值使得数据窗口(或记录长度)内恰好包含整数个周期。这就产生一个处于正弦波频率的单音FFT频谱响应，如图中所示。图7.49显示该正弦波的数据窗口内不含整数个周期的状况。端点的不连续性等效于将该正弦波乘以一个具有sin(x)/x频域响应的矩形窗口脉冲。时域中的不连续性会在频域中产生泄漏，因为需要许多频谱项来填补该不连续性。由于端点不连续，FFT频谱响应显示正弦波的主瓣被污损，并显示了大量具有矩形时间脉冲基本特性的相关旁瓣。

窗口包含非整数周期的正弦波的FFT

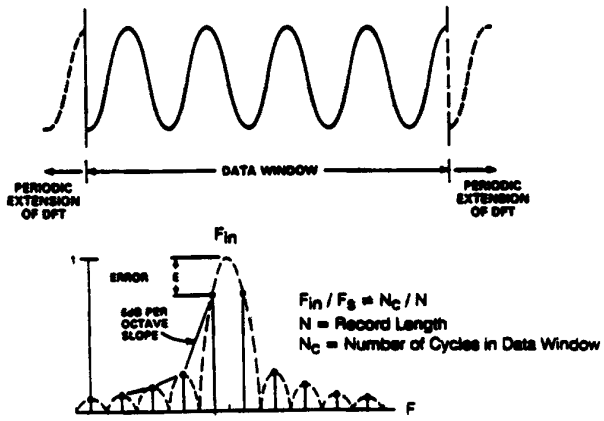

图7.49

在实际的FFT频谱分析应用中，确切的频率是未知的，因此必须采取措施将这些旁瓣减至最小。这可以通过选择窗口函数(而不是矩形窗口)来实现。输入时间样本乘以适当的窗口函数，从而将窗口边缘的信号调整为0。适当窗口函数的选择主要需考虑主瓣扩散与旁瓣滚降之间的关系。在数据中添0，从而执行更长时间的FFT，也可以减少泄漏。关于窗口的深入讨论，强烈建议参阅参考文献4。

图7.50显示了一个简单窗口函数(Hanning窗口)的时域和频域特性。图7.51比较了Hanning窗口与更复杂的最少4项Blackman-Harris窗口的频率响应。

窗口包含整数周期的正弦波的FFT

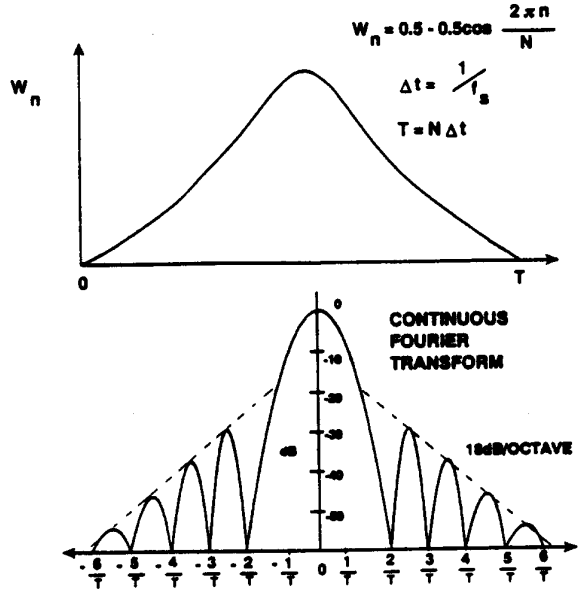

图7.50

蝴蝶计算的结果可能大于其输入，这种数据增长会给固定位数的DSP带来潜在的问题。为防止数据溢出，需要事先缩小数据，以便为增长留有足够的多余位数。或者，也可以在每次FFT后缩小数据。用于在每次FFT之后缩小数据的技术称为“块浮点”。它之所以得到这样的名称，是因为整个数组作为一个块缩放，无论块中的各元素需要缩放与否。整块缩放后，各数据字的相对关系保持不变。例如，如果各数据字右移一位(除以2)，则绝对值改变，但彼此的相对值保持不变。

加权函数的比较

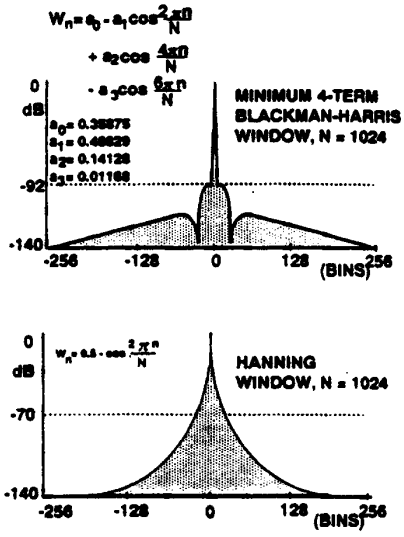

图7.51

FFT总结

- **FFT是一个算法，而不是近似计算**
- **高速运算的实现不以精度为代价**
- **FFT是DFT的快速实现方法**
- **FFT的频率分辨率为f_s/N，N = 记录长度**
- **时间中的端点不连续常常要求利用窗口函数使之平滑**
- **利用DSP微电脑，可以在高于100kHz的采样速率时实现实时FFT处理**

图7.52

参考文献

1. Richard J. Higgins, Digital Signal Processing in VLSI, Prentice-Hall, 1990.
2. A V. Oppenheim and R. W. Schafer, Digital Signal Processing, Prentice-Hall, 1975.
3. L. R. Rabiner and B. Gold, Theory and Application of Digital Signal Processing, Prentice-Hall, 1975.
4. Fredrick J. Harris, On the Use of Windows for Harmonic Analysis with the Discrete Fourier Transform, Proc. IEEE, Vol. 66, No. 1, 1978 pp. 51-83.
5. Momentum Data Systems, Inc. Costa Mesa, CA.
6. Fares Eidi, An Optimized Radix-4 Fast Fourier Transform (FFT), Analog Devices Application Note E1329-5-9/89. Available from Analog Devices.
7. High Speed Design Seminar, Analog Devices, 1990.
8. Amy Mar, Editor, Digital Signal Processing Applications Using the ADSP-2100 Family, Prentice-Hall, 1990.
9. C. S. Williams, Designing Digital Filters, Prentice-Hall, 1986.
10. R. W. Ramirez, The FFT: Fundamentals and Concepts, Prentice-Hall, 1985.