2020年中国研究生数学建模竞赛A题（华为公司命题）

**ASIC芯片上的载波恢复DSP算法设计与实现**

光数字信号处理（DSP）芯片是光传输领域里的“心脏”，这种芯片往往是基于专用集成电路（ASIC）实现的。例如，采用7nm芯片工艺制造的光传输芯片容量可以达到800Gbps，相当于单光纤可实现**48T**bps的容量，保障了网络流量的爆发型增长。ASIC芯片的DSP算法设计通常包含两个主要步骤，第一步是根据信道损伤的物理模型设计补偿算法，此时只需要考虑浮点计算；第二步是根据芯片资源和功耗约束，将算法改造成ASIC芯片可实现的定点形式，此时需要将算法细化为芯片上最基本的乘、加等运算，并考虑定点量化噪声的影响。怎样权衡性能和资源，实现具体场景下的最优设计，是DSP芯片算法工程领域持久不变的课题。本题以oDSP中一种关键的载波恢复算法为例，探讨算法与芯片的最优工程设计。

首先介绍关于基本通信系统和ASIC芯片上算法设计的基本知识。

**一、通信系统模型**

本题考虑一个简化的数字通信系统性能评估模型，如图1所示。发送端编码后的二进制序列映射调制为星座点上的符号并向外发送，每秒发送的符号个数称为波特率fBaud。信号在信道中受到色散和相位噪声的影响，并人为加入加性高斯白噪声，噪声量用信号和噪声功率的比值表示。接收端先补偿色散，再由载波恢复(Carrier Recovery, CR)算法补偿相位噪声，最后信号进行判决后逆映射为二进制比特序列。受信道中损伤和噪声影响，星座图会发生扩散，从而导致信号判错，使接收到的二进制序列与发端不一致，从而带来误码。错误二进制比特占总二进制比特的比率称为误码率（BER）。只要BER小于某个门限，那么纠错编码后的BER就能够小于1e-15次方量级，达到工程意义上的“无误码”传输。本题中不考虑纠错编码，BER均指直接判决后的BER。

算法评估中常用RSNR（Required SNR）代价来评估算法性能。 SNR(Signal-to-Noise Ratio)指的是信号功率和噪声功率的比例。例如图1中，若只有加性高斯白噪声存在时，对于给定的调制格式，SNR和BER存在关系也就确定了。将BER达到门限时信道中施加的SNR定义为Required SNR (RSNR)，它可理解为系统能够容忍的噪声量。当存在相位噪声、色散等干扰后，相同纠前门限点对应的SNR值会提高，说明系统能够容忍的噪声量减少了，将RSNR的增加部分称为RSNR代价。RSNR代价是衡量系统和算法性能的常用指标。例如CR算法性能越好，那么RSNR代价也应该越低。图1 模型计算RSNR的噪声是“人为”加入的，目的是用于评估系统性能。现实光传输系统中，噪声可能有来自于电器件、光器件、光放大器等各种系统组件。

下面对模型中的术语做进一步解释。

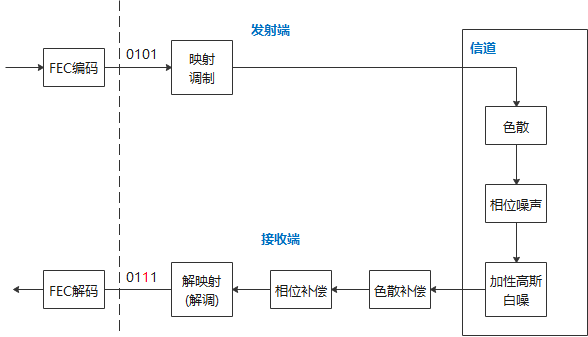


图1 简化数字通信系统性能评估模型

**【调制、星座图与误码率BER】**

二进制序列通常需要将K个比特作为一个“符号”进行传输，每个符号有个不同状态。光传输利用光波的复振幅承载信号，因此可用复平面上不同的点来对应不同的符号状态，这种将符号状态画在复平面上的图称为“星座图”，图上的点称为“星座点”。如图2(a)所示的QPSK（Quadrature Phase Shift Keying）调制，经过信道叠加噪声和接收机处理后，接收端的星座图不再是理想的四个点，而是会出现扩散。当接收机收到1个符号时，就将发送的符号判定为离该符号最近的星座点。显然，如果噪声过大，接收到的符号可能被判错从而产生误码，如图2(b)中的蓝点。误码率（Bit Error Ratio, BER）定义为错误的比特数占总传输比特数的比例，例如传输了50个符号共100个比特，其中有1个符号被误判为相邻的符号，错误了1个bit，则误码率为0.01。BER是衡量通信系统性能的最根本指标。

图2 星座图与噪声导致误码的示意图 图3 信号和噪声的相关定义示意图

### 图3中理想星座点用表示，接收到的符号用表示，则噪声为

### (1)

### 噪声通常服从均值为0的正态分布。噪声的方差等于噪声的平均功率，定义为

### (2)其中N为总共传输的符号数。信号平均功率定义为发送符号绝对值平方的均值：

### (3)

定义信号和噪声功率的比值为信噪比（Signal-to-Noise Ratio, SNR），

(4)

工程上通常用dB作为SNR的单位，定义为

(5)

**【相位噪声与CR算法】**

相噪会对信号叠加一个时变的相位，

(6)

其中是叠加相噪前的波形，是叠加相噪后的波形，是相噪。通信系统常用等间隔采样后的离散方式表示，每个采样点对应了某个时刻采样到的波形。对于相位噪声，k+1时刻和k时刻的相位差表示为：

(7)

其中LW是一个激光器线宽指标，单位为kHZ。是波特率，而是均值为0，方差为1的随机变量。一个典型的相噪随时间变化的规律如图4所示，相位变化也有可能演化到负值。

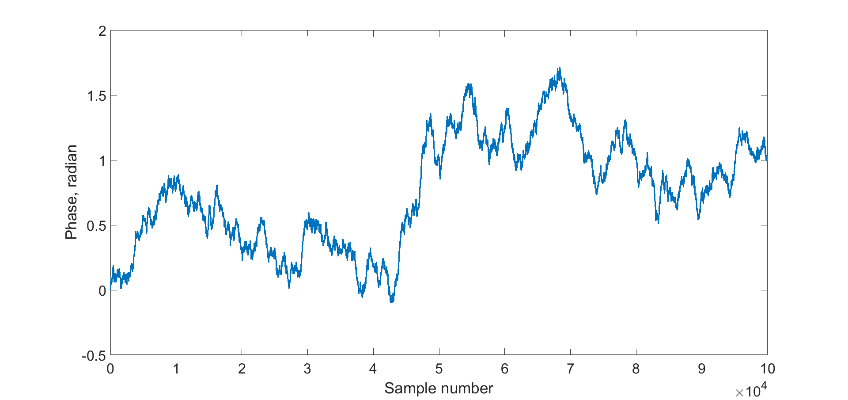


图4 典型相位噪声演化曲线

如图5所示，典型的CR算法是间隔性地插入已知的导频符号（Pilot），通过比较接收信号和已知符号的相位差，来估算出当前的相位噪声，再将此相位差反乘到接收端受影响的符号上，就实现了相位噪声的补偿。Pilot占总符号的比例称为Pilot开销，例如每N个符号中包含M个Pilot，则开销为M/N。在设计CR算法时可以有各种考虑：例如应该尽可能减小Pilot以降低系统开销，因为Pilot本身为已知量并不传递信息；间插Pilot之间净荷的相位可以通过各种插值方法来近似；加性白噪会影响相位估计精度，可以用2个连续的Pilot符号求平均以抑制白噪影响，也可以在间插的Pilot之间求平均。当然，以上仅是举例，实际CR算法不限于此。

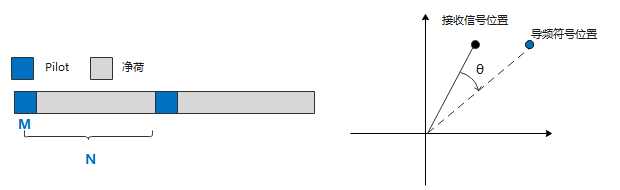


图5 Pilot的载波恢复算法

**【色散与色散补偿算法】**

光纤中色散的效应，可认为是对信号直接做傅里叶变换后的频域数据施加1个随频点平方关系变化的相位，如式8所示。

(8)

其中是波长，取1550nm，Dz是色散值，c是光速，f则是频点。本题中，信道中施加色散和算法中补偿色散如图6所示，假设色散值已知，则先FFT将接收到的数据转换到频域，再乘上式中的频域响应，随后再IFFT转换回时域即可。信道中色散的频域响应和算法中补偿色散的频域响应呈共轭关系。

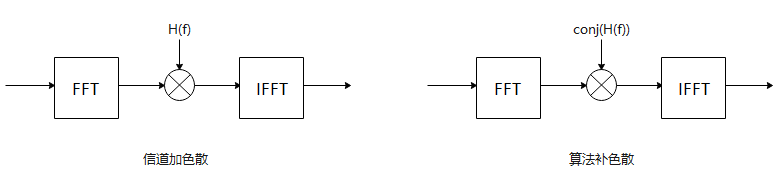


图6 色散和色散补偿方法

**二、ASIC芯片上的算法实现**

与我们通常在计算机上开展的基于软件的数学计算不同，在ASIC上的计算是基于硬件电路展开的。例如要计算某个公式，通用计算机上是转换为一条条逻辑指令，按时间顺序在相同的CPU中运行，最终输出结果。而在ASIC上开展的计算，则是把计算过程拆分为具体的加法、乘法等基本操作，每一个基本操作都对应了不同的专用逻辑电路，在芯片上各自占据了一定的面积。设计ASIC芯片上的DSP算法，需要考虑并行实现，定点量化，时序约束，和资源/功耗约束等约束，下面简要叙述。

**【并行实现】**

芯片上的计算都是在系统时钟下，以一拍一拍的节奏运行的。芯片计算的吞吐量必须要大于信号传输速度，才能确保信息不丢失。如果采用串行的处理方式，芯片时钟频率极高，而芯片的功耗与时钟主频率近似呈平方关系，显然不能通过一味提升主频的方式增加处理流量，必须采用并行处理的方法，以资源换取处理流量的提升。例如图7中所示的求平方操作，串行情况下1个时钟周期处理1个符号，100G的波特率至少需要100GHz的时钟主频，远远超出了现实能达到的水平。若付出N倍的资源，每次同时计算N个符号，则时钟主频仅为1/N，功耗大大降低。当前oDSP时钟主频在500MHz~1GHz量级，对应符号的并行度约为100~200量级。

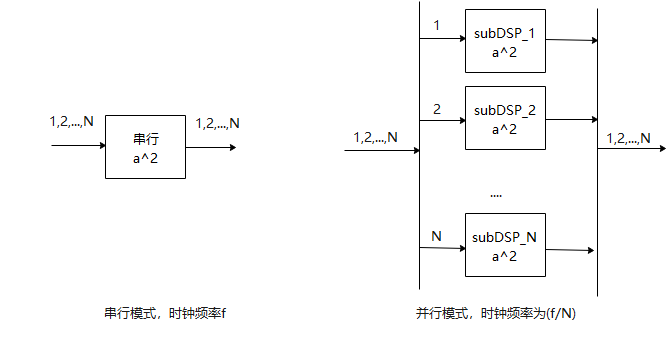


图7 ASIC芯片并行运算示意图

**【定点量化】**

计算机中常用双精度浮点数来定义参变量，而ASIC通常用定点数来表示参变量的大小。双精度浮点数在大多数计算情况下几乎没有精度损失，而ASIC定点数因为位数往往较小，舍入误差增大，带来量化噪声。表示定点数的二进制位数称为定点位宽。定点数常分为有符号和无符号位2种，如下图所示，例如S(8,4)表示有符号的8-bit定点数，其中小数位占4位；u（7,4）代表无符号的7-bit定点数，其中小数位占4位。单个定点数仅能代表实数，复数则用2个定点分别代表实部和虚部。在ASIC中任何计算都应量化为定点数的计算，量化噪声的影响也是算法设计的关键之一。例如CR相噪算法中，收端受影响的符号往往用6~9bit定点数表示，而计算相噪部分的位宽则根据实际情况而变化。

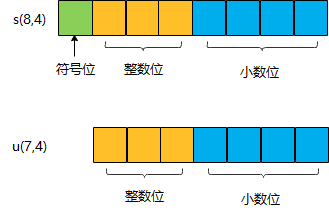


图8 定点表示法

**【基本操作、时序约束和资源/功耗约束】**

ASIC上的算法设计，需要将计算拆解为基本的操作。本题中考虑加法、乘法，查表和数据缓存四类操作，其表示如图9所示。加法和乘法均是对两个数开展，复数运算拆分实部和虚部各自计算。减法可认为是先做一次乘以-1的乘法，再做一次加法。ASIC上由于是二进制定点数表示，乘以或除以2的幂次方相当于对小数点进行移位，不会带来额外的资源代价。查表用于加法和乘法无法直接实现的操作。例如求sin函数，可以将输入-输出映射关系分别量化为离散输入-输出对应关系，通过查表的方式求得输出。例如求 d， 首先一级加法计算a+b, 再通过移位得到，随后一级乘法得到, 最后通过sin查表的方式得到d。每种运算都对应了芯片上的资源。显然，参与运算的数据位宽越大，占用的资源也会更多。



图9 ASIC芯片基本运算操作

另一方面，芯片是在时钟主频下一拍一拍运行，一拍运行中只能完成有限个连续的基本运算，但通过加缓存的方式，可以实现流水的结构。例如图10中，若运算1结果出来的时候，一拍内剩余的时间无法完成运算2，则需要在运算1的出口加入1级buff缓存以存储运算1的输出数据。每拍运算时，运算2读取buff中存储的运算1的上一拍输出开展计算，而运算1则和当前输入数据开展运算，并将结果在存入buff中，这样一来，每拍运行可以同时开展运算1和运算2，但会造成输出的结果会比运算1和运算2在同一拍完成的情况延迟1拍。



图10 缓存实现流水结构

不同的运算路径有着不同的时钟延迟，当算法存在不同的计算路径时，需要保证每个路径的延时对准。例如在CR算法中，计算出的相位噪声差需要反乘回数据，但相噪计算通常无法在1拍内完成，从而导致相位和数据有延时，需要用buffer将延时对准。本题中，假设1个时钟周期最多完成1级乘法，4级加法，以及1级查表操作。

加法、乘法、查表和缓存均需要占用资源，在算法设计也应有所考虑。加法和乘法均是两两操作，加法资源近似与最大位宽呈正比，乘法资源近似与位宽乘积成正比，延时资源和位宽以及延时深度近似呈正比。对于一个M-N的查表，资源与(2^M)\*N呈正比。M代表输入位宽，N代表输出位宽，例如：已知信号求相位的查表操作，信号位宽s(10,1)，相位位宽s(9,1)，此时的M=20，N=9。在查表时候的输入为信号的【实部 虚部】组合，需要遍历的地址有2^20，每个地址对应的相位值的位宽为s(9,1)。

**表1**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **基本操作** | **8+8bit**  **加法器** | **8\*8 bit**  **乘法器** | **8bit-8bit查表** | **8bit,**  **每2048符号延时** |
| **资源** | **1 U** | **8 U** | **128 U** | **1 U** |

**三、ASIC芯片算法设计的典型步骤**

综合上述内容，芯片上算法设计的通常包含以下具体步骤：

1) 根据物理模型和系数特性设计算法原型：例如CR算法中，采用Pilot估计相位差，并利用插值等方法计算出净荷的相噪，最终将估计的相位差乘回数据以补偿相噪。

2）在考虑并行度和时序约束的基础上，将算法细化为可实现的基本操作，并大致考虑算法的实现复杂度对算法实现的影响。此时可假设为浮点数，不需要考虑定点量化噪声。例如CR算法中，可能需要考虑净荷相噪的插值如何由基本操作实现，计算噪声路径的延时如何与补偿噪声路径的延时相匹配等。

3） 进一步考虑定点量化噪声的影响，尽量以更低的资源实现，此时位宽优化是一个关键步骤。

----------------------------------------问题---------------------------------------------

**问题1：考虑波特率为150Gbaud的标准16QAM信号，令线宽为100kHz，色散值为2万ps/nm，算法的并行度固定为128，不考虑定点量化。请以基本的加法、乘法、查表和缓存为基础，并以RSNR代价<0.3dB为目标，设计一套CR算法，使得Pilot开销最小？**

**问题2：考虑线宽从10kHz~10MHz，色散Dz从0~10,000 ps/nm变化场景，以RSNR代价<0.3dB为目标，定量挖掘色散、线宽与Pilot开销的关系。**

**问题3：在问题2的场景上，进一步将芯片实现的资源纳入考察，此时需考虑定点量化对性能和资源的影响，且导频开销可任意变化（但必须确保净荷的流量为>145Gbaud），如何设计资源最低的CR算法？**

**问题4：现实中性能和资源的权衡与具体场景有关。例如长距干线传输对性能要求往往比短距离要求更高，长距传输可付出更多的资源以降低RSNR代价。并选出问题3中你队认为有代表性的1种场景，给出统筹性的“性能-资源”综合考虑下的算法设计思路，构造性能和资源的综合代价函数，尝试给出一套自动优化位宽和实现性设计的方案，并给出定量结果，用以指导算法开发。**

**注：本题中均不考虑色散补偿和误码率计算的复杂度和资源，只需考虑CR算法（计算相噪+补偿相噪）相关的资源即可。**