西 南 交 通 大 学

本科毕业设计（论文）

GSM-R 空口协议软件-物理层模块设计实现

年　　级：2014 级

学　　号：2014111810

姓　　名：曹 云

专　　业：通信工程

指导教师：刘 林

二零一八年六月

西南交通大学

本科毕业设计（论文）学术诚信声明

本人郑重声明：所呈交的毕业设计（论文），是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的作品成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律结果由本人承担。

作者签名：

日期：　　年　　月　　日

西南交通大学

本科毕业设计（论文）版权使用授权书

本毕业设计（论文）作者同意学校保留并向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅。本人授权西南交通大学可以将本毕业设计（论文）的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存和汇编本毕业设计（论文）。

保密□，在_____年解密后适用本授权书。

本论文属于

不保密□。

（请在以上方框内打“✓”）

作者签名：　　指导教师签名：

日期：　　年　　月　　日　　日期：　　年　　月　　日

院系 信息科学与技术学院 专 业 通信工程

年级 2014 级 姓 名 曹 云

题目 GSM-R 空口协议软件-物理层模块设计实现

指导教师

评 语

指导教师 (签章)

评 阅 人

评 语

评 阅 人 (签章)

成 绩

答辩委员会主任 (签章)

年 月 日

毕业设计（论文）任务书

班　　级　通信 2014-02 班　学生姓名　曹 云　学　　号　2014111810

发题日期：　2017 年 11 月 26 日　　完成日期：　2018 年 6 月 1 日

题　　目　GSM-R 空口协议分析软件-物理层模块设计实现

1、本论文的目的、意义

目前，现有高速铁路 GSM-R 网络接口监测系统尚缺少对 Um 接口进行监测的设备，因此高铁线路出现 CTCS-3 无线超时和降级问题时，由于缺失 Um 接口监测数据，无法准确判断是车载 MT 问题，还是 GSM-R 网络问题。Um 接口通信质量易受无线信号的传输特性和无线环境影响。目前网络对出现突发干扰，无线环境变化导致故障的情况没有记录，无从考究。对 Um 接口进行监测，可实现跟踪车载设备到地面终端（RBC 无线闭塞设备）之间的所有信令和业务过程，为列控数据无线超时的准确分析和定位故障提供依据，对保障高速铁路列车的安全运营具有重要意义。本课题将完成 GSM-R 空口协议分析软件的设计实现。

2、学生应完成的任务

1 学习 GSM-R 物理层协议

2 物理层模块软件设计实现

3 翻译。

4 撰写论文

3、本论文与本专业的培养目标达成度如何？（如在知识结构、能力结构、素质结构等方面有哪些有效的训练。）

1. 具有运用通信工程专业知识，针对复杂工程问题进行科学描述，建立表达该工程问题的数学模型的能力（指标点 2.2）。

2. 具有通过文献研究，分析复杂通信工程问题解决过程中的影响因素，认识到解决复杂通信工程问题有多种方案可供选择，并能归纳相关解决方案，论证解决方案的合理性，获得有效结论（指标点 2.3）

3. 综合运用通信工程专业理论和技术手段，开发满足特定需求的通信系统硬件部件和软件。开发过程中具有追求创新的态度、意识和能力（指标点 3.3）。

4. 针对复杂工程问题，能基于通信工程专业理论、科学原理并采用科学方法拟定可行的实验方案，根据实验方案构建实验系统，进行实验研究(指标点 4.2)。

5. 能够正确采集、整理数据，对实验数据进行关联、建模，分析和对比、解释实验数据、并通过信息综合修正实验方案，得到合理有效结论（指标点 4.3）。

6. 掌握主要网络和文献检索工具的使用方法，了解本专业重要资料来源及获取方法，具备收集工程问题的相关技术信息的能力。能够针对复杂问题，选择与使用恰当的资源、现代工程工具和信息技术工具（指标点 5.1）。

7. 能够选择恰当的技术、资源、工具，对通信工程领域复杂工程问题进行模拟和预测，并能够理解其局限性（指标点 5.3）。

8. 具备良好的口头表达能力和人际交往能力，能够控制自我并了解、理解他人需求和意愿，能够通过口头或书面方式就复杂工程问题与业界同行及社会公众进行有效沟通和交流（指标点 10.1）。

9. 能够正确、熟练、规范的运用汉语撰写工程文件，如可行性分析报告、技术报告、设计文稿等，并可进行说明、解释，清晰表达或回应指令（指标点 10.2）。

10. 针对通信工程项目的多学科特点，具备在工程项目管理过程中运用工程管理原理与经济决策方法的能力（指标点 11.2）。

4、论文各部分内容及时间分配：（共 17 周）

第一部分 收集资料与阅读,开题报告,完成译文 （ 2 周）

第二部分 学习 GSM-R 物理层协议 （ 5 周）

第三部分 协议分析软件设计 （ 5 周）

第四部分 软件测试 （ 2 周）

第五部分 完成论文写作 （ 2 周）

评阅及答辩 审定,装订,准备答辩 (1 周)

备 注

指导教师：_____ 年 月 日

审 批 人：_____ 年 月 日

摘 要

目前对 GSM-R 网络进行监测和分析比较成熟的工具主要集中在对 GSM-R 后端有线网络的监测分析，即现有高速铁路 GSM-R 网络接口监测系统主要是对 Abis 接口、A 接口、PRI 接口、Gb 接口、Gi 接口以及车载设备端的接口等进行监测，尚缺少对 Um 接口的监测设备，无法对 Um 接口问题再现和定位故障。因此，对 UM 接口进行监测具有重要意义，本文实现了对 GSM-R Um 接口物理层协议解析的设计与实现。

本论文依据 3GPP 的 GSM 空中接口标准协议，设计实现了基于 Visual Studio2017 和 MATLAB2014 的、具有 GSM-R 空口协议物理层发送与接收消息功能的软件。通过编程，本文实现了信道编解码、交织、映射、调制解调、接收信号同步等功能，并基于此进行了数据测试。

本文讨论的物理层模块设计，主要分为发送端与接收端两部分。在发送端主要用 C 语言进行编程，信道编码用到了（2,1,4）卷积编码，交织分别进行了块内交织和块间交织，在块间交织的过程中完成了信道映射。在 MATLAB 上，通过相位表进行 GMSK 调制，解调使用了一比特差分解调法。在接收端，首先利用频率校正突发脉冲数据部分全零的特性进行粗同步，再根据同步突发脉冲训练序列的自相关性完成时间精确同步。解交织可以看作是交织的逆过程，信道解码用到的是 Viterbi 算法。最后得到 23 字节的数据格式交付给数据链路层进行进一步分析。对于犀浦镇的实测数据进行了解析，测试结果与理论预期相符。

关键词：GSM-R 系统；空中接口协议；物理层模块；

Abstract

At present , the Current analysis tools for GSM-R network are mainly focused on the back-end wired networks of GSM-R. The existing monitoring tools for high-speed railways GSM-R network system are mainly used in Abis interface, A interface, PRI interface, Gb interface, Gi interface or the interface of the on-board device. It lacks of Um interface monitoring tools which make it difficult to solve Um interface problems and locate the faults. Therefore, the purpose of this paper is to implement the GSM-R Layer 1 data analysis software.

According to 3GPP protocols, this paper proposes the analysis software for GSM-R Um air interface in Physical Layer (L1) based on Visual Studio 2017 and MATLAB 2014, which has the function of sending and receiving messages. This paper realizes channel coding and decoding, interleaving and de-interleaving, channel mapping, modulation and demodulation, , synchronization of received signals and implements data test based on these functions.

The Physical Layer module discussed in this paper is mainly divided into two parts: sending end and receiving end. The transmitter is programmed mainly in the C language, and the channel coding uses (2,1,4) convolution coding method. Inner interleaving and inter-block interleaving are designed respectively. The channel mapping is completed during inter-block interleaving. On the MATLAB platform, GMSK modulation is realized through the phase table, and 1- bit difference method is used for demodulation. At the receiving end, the full zero data characteristic of FCCH is used for synchronization step 1, and the timing accurate synchronization can be finished by using the autocorrelation of the SCH training sequence. De-interleaving can be regarded as an inverse process of interleaving, and the Viterbi algorithm is used for channel decoding. Finally, 23 bytes of data format are delivered to the Data Link Layer (L2) for further analysis. The test data of Xipu town are analyzed by the software, the results satisfy the theoretical expectation.

Keywords: GSM-R system; Um interface protocol; Physical Layer;

目 录

第 1 章 绪 论

1.1 背景与意义

GSM 即全球移动通信系统，作为第二代移动通信技术使用的主要标准，于 20 世纪 90 年代开始投入运营，开辟了数字移动通信时代。GSM-R（Global System for Mobile Communications－Railway）是一项由国际铁路联盟支持开发，基于 900MHz 频段的，用于铁路通信及应用的国际无线通信标准。在该标准中，引入了优先级、广播呼叫、组呼叫和位置寻址等功能。GSM-R 在欧洲已经有了十多年的运用，并形成了一套比较完善的标准体系[1]。

现有的高速铁路 GSM-R 网络接口监控系统主要监控 Abis 接口、A 接口、Gb 接口、Gi 接口和车载终端的接口。目前，缺乏对空中接口（Um）的监控设备，不能再现 Um 接口问题和定位故障，无法监控 GSM-R 网络的电磁环境和干扰的动态变化。Um 接口是 GSM-R 网络中的一种无线接口。通信质量极易容易受到无线信号及无线环境的传输特性的影响。在 CCTS-3 无线超时和降级的分析和处理中，现有的接口监控系统由于缺少 Um 接口监控数据，不能准确地断定是车辆 MT 问题或 GSM-R 网络问题。首先，Abis 接口和 A 接口监控数据仅包括基站和交换机之间的信令数据，不能对车载设备和交换机之间的语音和数据服务进行监控；其次，移动设备中的一些信令仅能通过移动进程中的 Um 接口传输，所以当这部分信令传输失败时，现有的 Abis 接口和 A 接口监控系统不能重现和故障定位。目前，没有关于网络突发性干扰和无线环境破坏导致故障的记录[2]。在铁路环境中实际运营时，网络容易出现突发干扰，比如出现切换后无邻区、突发质差等因不能判断准确原因而无线超时。对于这些情况，现在的处理办法是：等待故障再次出现。原因未明的故障率很高，问题点也很分散，同样的故障二次出现并在现场及时进行网优测试并不现实；安排人工上道排查故障。这种方式难以避免得效率和质量都不高，且是对人力资源的浪费。所以，为了提高 GSM-R 系统维护效率和故障检修水平，保障高铁高效与安全地正常运行，必须加快铁路沿线 Um 接口监测的部署。开发系统检测 Um 接口，可以跟踪车辆设备和地面终端（RBC 无线闭塞设备）之间的所有信令和业务流程。给列控数据无线超时的准确分析和定位提供了依据。这对保证高铁列车的安全运行有很重要意义。

本项目将完成 GSM-R 空中接口协议分析软件的设计与实现。

1.2 国内外发展（应用）现状

随着软件技术的快速发展，基于软件平台的协议分析系统也得到发展，但是现有高速铁路 GSM-R 网络接口监测系统还缺少对 Um 接口进行完备的监测与分析的系统。

对于 GSM-R 中 Um 接口的监测已经初步能够得到 Um 接口的监测信令和数据，但没有实现用于分析监测得到的信令和数据的软件分析平台。

关于 GSM-R 空中接口物理层，文献[3]中提到， GSM-R 的频点下行分布在 930-934 MHz，该频段即接收机覆盖的频段。GSM-R Um 接口检测仪先将收到的信号从高频转换到中频，在中频段完成数模转换，然后中频数字信号到基带信号的转换是在 FPGA 上进行的[4]。

在其文献[5]中提到了 GSM 发射机的组成，主要有 5 个步骤，分别是随机比特发生器，信道编码，交织，形成突发，调制 5 个步骤；在接收机同步方面，文献[6]提到了 GSM 系统同步所采用的方法，包括了 FCCH(频率校正突发脉冲)时间粗同步和 SCH(同步突发脉冲)时间精确同步，并且对具体的算法做出了分析。初同步是通过 FCCH 进行 FFT 来实现的，而时间精确同步是要根据突发中的同步序列进行自相关运算。在接收机的均衡与解码算法方面，Proakis 的文献[7]提出了最大似然序列均衡(MLSE)，G．D．Fomey 在文献[8]提出的维特比算法(VA)来消除接收序列中的码间干扰，实现无差错解码。本文也将根据上述文献提到的方法来进行 GSM-R 空中接口软件物理层的实现。

1.3 论文所做工作及思路

本文主要分三步实现了 GSM-R 空口协议软件-物理层模块设计实现：

（1）首先是分析协议，会查阅相关的 3GPP 协议，先自己理解物理层用到的理论知识，包括帧结构、逻辑信道与物理信道的映射关系、控制信道和业务信道的区别等。结合理论知识自己编程实现用卷积码进行信道 编码、交织和映射、GSMK 调制等。

（2）系统设计与实现，在软件的物理层，主要实现解调、同步、解交织、信道解码等步骤，分别用 MATLAB 完成解调，用 VS2017 完成同步、解交织、信道解码。同步会先对接收到的信号进行频谱分析，找到频点、帧号等信息，然后用 1 比特差分法得到解调后的信号，接着进行解交织、Viterbi 解码得到 184bit 数据，交付 L2 进行进一步分析。

（3）最后用自己设计的程序来分析和处理老师给的实测数据，具体对数据的分析会在高层实现，物理层主要就是解析信号的频点、完成同步等内容。

1.4 论文章节安排

第 1 章介绍了本文的研究背景和意义，GSM-R 空中接口协议分析软件的发展状况等。

第 2 章分析了设计 GSM-R 空中接口协议分析软件物理层所涉及到的协议内容。

第 3 章基于 3GPP 协议，提出了对软件系统设计的需求，模块划分，以及开发环

境和平台简单介绍，并对软件的代码实现，验证系统功能。

第 4 章对老师给的数据，用自己设计的软件进行测试分析。

第 2 章 GSM-R 空中接口物理层协议

2.1GSM-R Um 接口

GSM-R 通信系统中基站和移动台间的空中接口被称为 Um 接口。本文实现的是 GSM-R 空中接口上的分析软件设计。Um 接口的分层结构及功能如下：

2.1.1 物理层

物理层是 GSM-R 空中接口协议的最底层，物理层提供了在有限逻辑信道集合上的传输服务。逻辑信道的定义和特点将会在 2.3 节中详细描述。

逻辑信道在物理信道上进行复用，而物理信道是在无线电介质上调度的单元，用来支持传输于话音信道以及控制信道的比特流。物理层的主要功能是完成物理信道到逻辑信道的相互映射，并提供一些技术，使得有用的比特流信息适合在实际的信道中进行传输，最终向数据链路层传输各种功能的逻辑信道信息。

2.1.2 数据链路层

由下往上，链路层是 Um 协议的第二层，主要功能是在 BS（基站）与 MS（移动台）之间建立可靠的数据链路。在 GSM-R 系统中，建立了 LAPDm 协议，以更好地适应无线路径传播。物理层与链路层之间，主要通过服务源语在接口进行数据交换。

2.1.3 网络层

网络层是最高层，主要功能是完成具体话务控制和业务管理的。按照具体功能，网络层可分为三个子层，分别是无线资源管理（RR），接续管理（CM），移动性管理（MM）。网络层主要实现协议解析。

2.2 频率资源规划

在现行的 GSM-R 系统中，主要有两个频段：GSM900 频段和 GSM1800 频段，这两个频段主要分布在 900MHZ 和 1800MHZ 左右。每个频段被等间隔地分割成若干频道，且各频段分有成对存在的上下行频段。

GSM900MHZ 的工作频段：上行 890-915MHZ，下行 935-960MHZ。

GSM1800MHZ 的工作频段：上行 1710-1785MHZ，下行 1805-1880MHZ。

每个频道载波中心之间的间距是 200MHZ，以此作为频道间隔。每一个频点采用 TDMA 技术分为 8 个时隙。

以 GSM900 为例，按照频率间隔 200MHZ，GSM900 上下行频段分别可分为 125

个频道，分别标记为 0-124 频点序号，被称为绝对频点序号（ARFCN）。其中，63 号为该频段的中心频点。

频道划分情况如表 2-1 所示：

表 2-1 频道划分

	$ARFCN(n)$	$F_u(n)$	$F_l(n)$
GSM900	$0 \le n \le 124$	$890.2MHZ + 0.2*n$	$F_l(n) = F_u(n) + 45MHZ$
GSM1800	$512 \le n \le 885$	$1710.2MHZ + 0.2*(n\text{-}512)$	$F_l(n) = F_u(n) + 95MHZ$

注：$F_u(n)$ 为上行频道频率，$F_l(n)$ 为下行频道频率。

2.3 时分多址技术 TDMA

目前常用的多址方式主要有三种，分别是 FDMA，TDMA，CDMA。FDMA 是为每一条链接通路配一条固定频率的信道，考虑到频带保护间隔等因素，频带利用率实际非常低。而 GSM-R 用的是 FDMA/TDMA 混合方式。TDMA 技术将每一个频点划分为 8 个时隙，每个时隙分别对应一个信道，通常每个频点的第一个时隙包含的是控制信道的信息，第二到第八个时隙为业务信道的信息。

2.3.1 物理信道与逻辑信道

GSM-R 系统有两种信道，分别是物理信道与逻辑信道，物理信道是由多址方式等其他传输的物理特性决定的，在 TDMA 时分多址技术中，一个物理信道就对应了一个时隙。而逻辑信道则会根据基站与移动台之间发送信令的功能来划分。在物理层中，要完成物理信道和逻辑信道的相互映射。

根据信息功能，逻辑信道可以分为两大类，一类是用来传递控制信令的控制信道，另一类是用来传递加密业务信息的业务信道，逻辑信道的分类如图 2-1 所示。

业务信道（TCH）：主要用于传输用户语音和业务数据。根据业务种类，可以分为话音数据信道与业务数据信道；根据传输速度，可分为全速率业务信道和半速率业务信道。

控制信道（CCH）：主要用于基站信号同步和传输控制信令。可以被分为三类，CCCH 公共控制信道，DCCH 专用控制信道和 BCH 广播信道。

公共控制信道主要用于管理小区内移动台接入过程。分为 PCH 寻呼信道，AGCH 接入许可信道和 RACH 随机接入信道。其中除了随机接入信道为上行信道，其余信道均为下行信道。PCH 的作用是当一端需要与移动台建立呼叫连接时，根据 MS 登记的 LAC 等寻呼信息，网络在 PCH 上向所有满足条件的 MS 发送寻呼信息。当基站接到

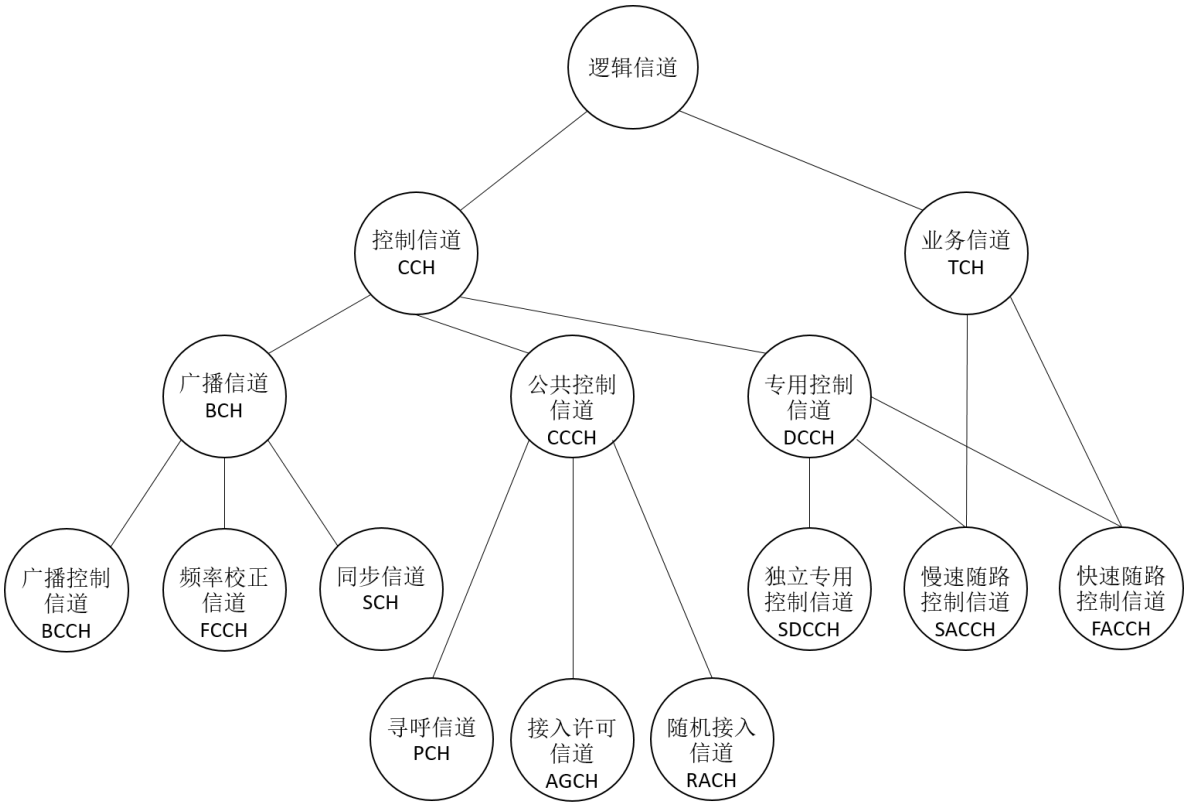

图 2-1 逻辑信道的分类

移动台的呼入请求时，便分配给 MS 一个专用的控制信道，AGCH 即该信道的描述信息。RACH 是 BS 需要主动与 BTS 接入时进行广播所需的服务信道。

专用控制信道与公共控制信道的区别在于是与某一个 BS 单独进行通信，分为 SDCCH 独立专用控制信道，SACCH 慢速随路控制信道和 FACCH 快速随路控制信道。SDCCH 是双向专用信道，用于传输建立连接所需信令。SACCH 和 FACCH 均与业务信道有关，SACCH 的上下行链路分别用来传输当前网络质量报告和系统消息，FACCH 则是为了适应高速传输过程中可能发生的偷帧现象而使用的控制信令。

广播信道就是基站与小区内移动台发送广播信息的单向下行信道，广播信道包括 BBCH 广播控制信道，FCCH 频率校正信道和 SCH 同步信道。BCCH 主要负责向小区内 MS 发送系统消息（SI），内容主要包括本小区及相邻小区的频率列表，LAI 号以及其他信道的参数等。FCCH 主要用于接收端和基站的频率同步，而 SCH 主要通过获取 TDMA 帧号（FN）和基站编号（BSIC）等信息完成位同步。

2.3.2TDMA 帧结构

由于 GSM-R 系统采用了时分多址的接入方式，一个物理信道就可以由 TDMA 帧号，时隙号和跳频序列号来定义，每一个时隙就对应了一个物理信道。

TDMA 帧组成格式从大到小可以分为超高帧，超帧，复帧，帧和时隙。

超高帧：TDMA 帧一般以一个超高帧为周期，持续时间为 3 小时 28 分 53 秒，由 2048 个超帧或者 2715648 个帧组成。

超帧：一个周期为 6.12s，有两种组成形式，分别由 26 个 51 复帧或者 51 个 26 复帧组成。

复帧：有两种复帧格式，51 复帧主要用于 BCCH、CCCH 和 SDCCH 等控制信道，周期为 3060/13≈235.385ms，专用于控制信道；26 复帧主要用于 FACCH 和 TCH 等业务信道，持续时长 120ms，用于业务信道及其随路控制信道。其中 24 个突发序列用于业务，2 个突发序列用于信令。

帧：一个帧包含 8 个时隙，26 或 51 个帧构成一个复帧，每一个 TDMA 帧含 8 个时隙，共占 60/13≈4.615ms。每个时隙含 156.25 个码元，占 15/26≈0.557ms。

具体的 TDMA 帧结构如图 2-2 所示。

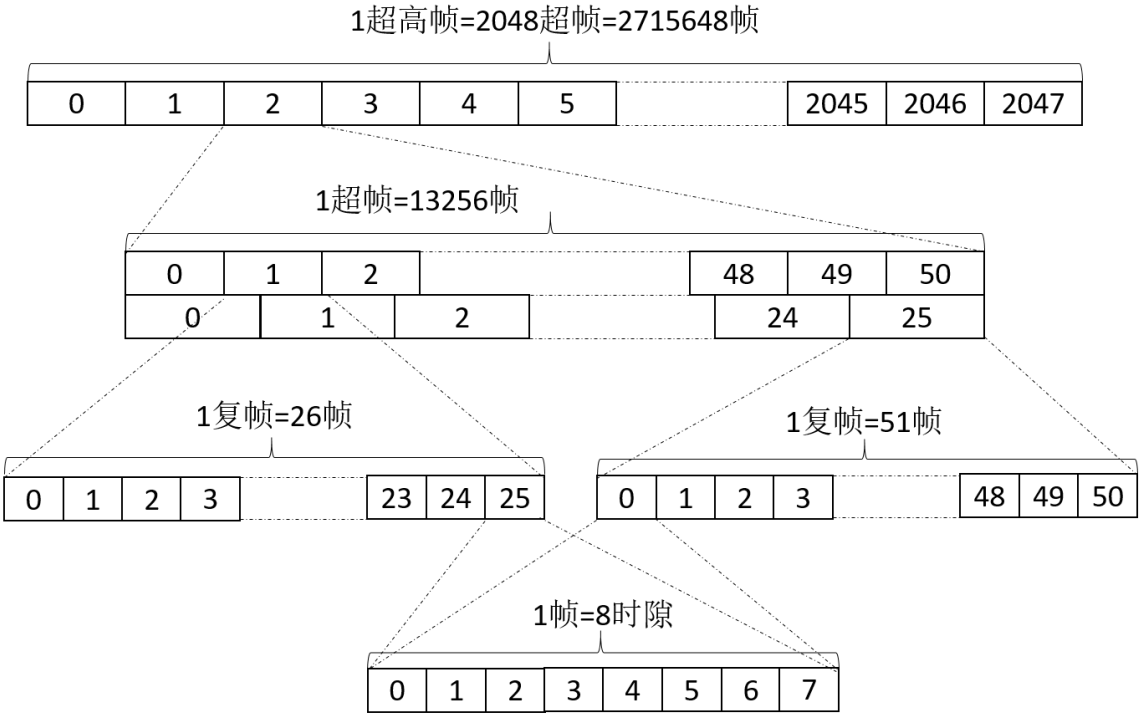

图 2-2 TDMA 帧结构

在每个 TDMA 帧时隙上，数据发送的格式被称为突发脉冲序列（Burst），分别为普通突发脉冲序列（NB），接入突发脉冲序列（AB），频率校正突发脉冲序列（FB），同步突发脉冲序列（SB）和空闲突发脉冲（Dummy Burst）。每个突发的长度都是 156.25bit，具体突发格式如图 2-3 所示。

普通突发序列（NB）主要包括 2 个 58 比特的传输信息和 26 比特的训练序列，2 个 58 比特里分别由 2 位用来做偷帧标志以区分 TCH 和 FACCH。根据 GSM 05-02 协议规定，训练序列有 8 种不同的表示来进行信道均衡。解码后可以得到链路层的传输帧或者网络层的消息帧。携带的消息类型包括 BCCH、PCH、SACCH、FACCH、

SDCCH、AGCH、TCH 等信道的消息。

接入突发序列（AB），是 MS 随机向网络请求接入时用到的突发，主要携带 41 比特的同步序列和 36 比特的加密信息，特点是保护间隔特别长，以适应移动台首次接入网络时不确定的时间提前量。

频率校正突发序列（FB），主要功能是携带 FCCH 上的消息实现载波同步，它由 142 个固定的全 0 比特组成，在时域上我们相当于一段频率 67.7KHZ 的正弦波。MS 将此正弦波与基站频率同步，才能继续在该信道上读取其他突发序列。

同步突发序列（SB），完成 MS 与 BTS 之间的时间（位）同步，主要是帧号（FN）的同步。该突发序列主要由 2 个 39 比特加密信息和 64 比特拓展训练序列组成。78 比特信息经过信道解码之后可以分离出 19 比特 TDMA 帧号（FN）和 6 比特基站识别码（BSIC）。

空闲突发序列（DB），一般与 NB 的格式相同，只是中间 141 比特为固定值，也就是说不携带任何有用的消息，只在基站没有要给移动台具体消息的时候发送。

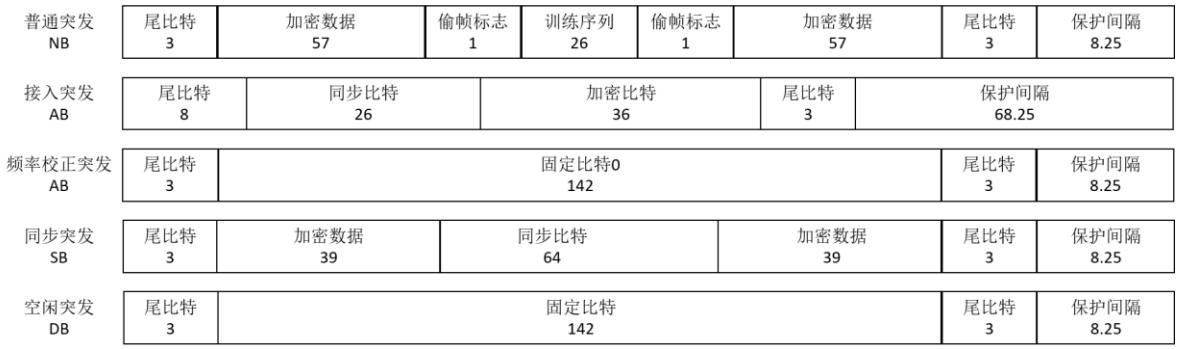

图 2-3 各类型突发脉冲结构

2.3.2 信道映射

在 GSM-R 系统中，共有 7 种物理信道与逻辑信道的映射方式。

（1） FACCH/F+TCH/F+SACCH/TF

（2） FACCH/H(0,1)+ TCH/H(0,1)+SACCH/TH(0,1)

（3） FACCH/H(0,1)+ TCH/H(0,1)+SACCH/TH(0,1)+TCH/H(1,1)

（4） CCH+BCCH+FSCH+CCCH

（5） SCH+BCCH+ FCCH+CCCH+SACCH/4(0,..,3)+ SDCCH/4(0,..,3)

（6） SACCH/8(0,…,7)+SDCCH/8(0,…,7)

（7） BCCH+CCCH

其中，控制信道的帧结构都是 51 复帧的格式,具体映射情况如图 2-4 所示。

业务信道的帧结构都是 26 复帧的格式，具体映射情况如图 2-5 所示。

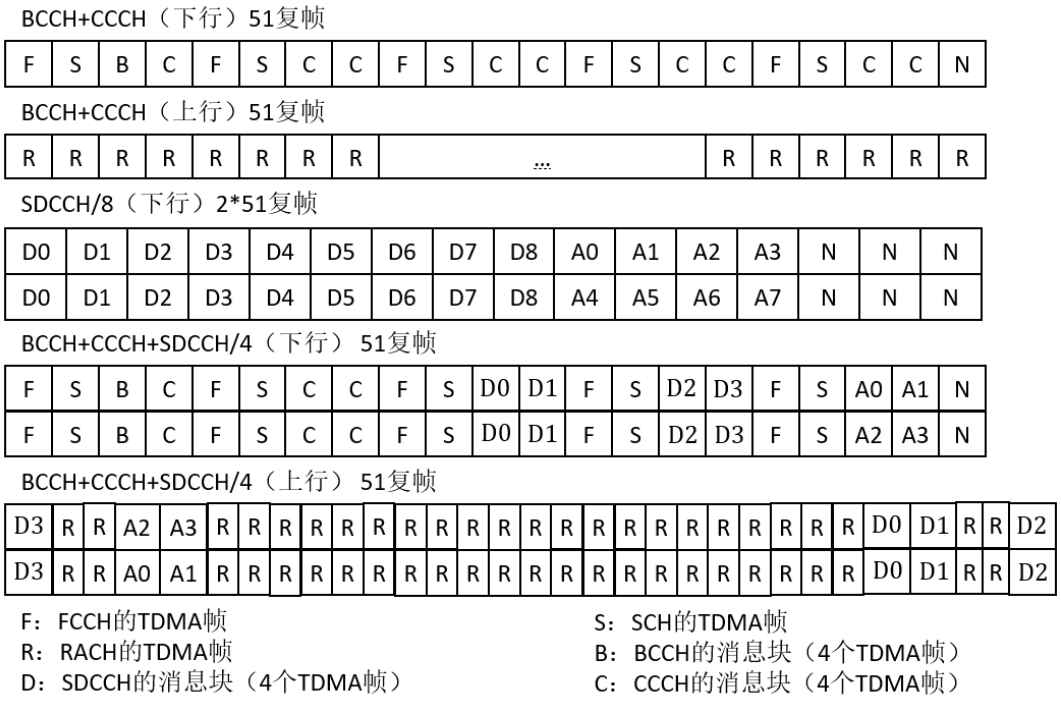

图 2-4 控制信道映射图

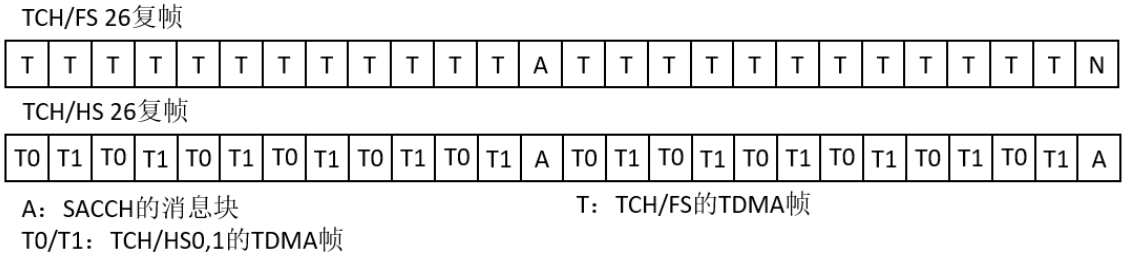

图 2-5 业务信道映射图

2.4 信道编解码与交织

GSM-R 物理层另一个重要的任务就是要提高数据传输的可靠性，为此，网络层和数据链路层的数据在物理层中传输时必须进行数字化的处理。为了将比特流转化为适合在信道中传输的电信号，需要以下几个步骤的处理。本文主要讨论了 GSM-R 控制信道的信令消息的传输过程，由于 TCH 信息大多经过加密处理，所以本文没有对业务信息进行深入探讨。对于 CCH 信令数据，在物理层发送时要经过信道编码（Channel Coding）、交织（Interleaving）、信道映射（Channel Mapping）和调制（Modulation）共四个步骤。相对地，在接收端要进行解调（Demodulation）、信道映射（Channel Mapping）、解交织（De-Interleaving）、信道解码（Channel Decoding）四个步骤。具体过程如图 2-6 所示。

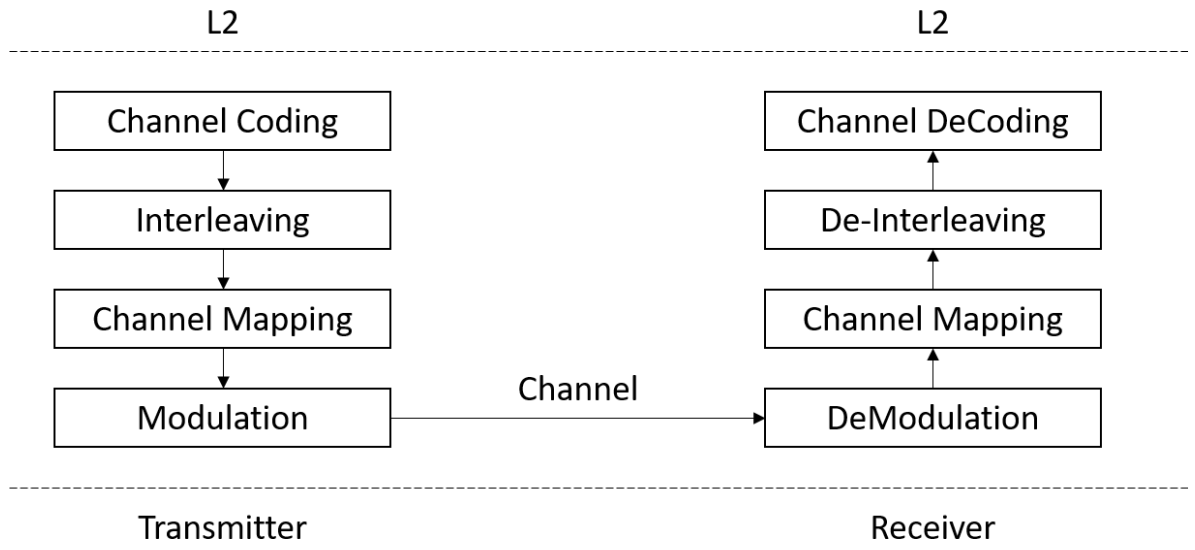

图 2-6 GSM-R 物理层传输过程

2.4.1GSM-R 信道编码

GSM-R 系统共有三种信道编码方法：奇偶码、分组码和卷积码，主要作用是用于检错和纠错。

1. 奇偶码

GSM-R 共有三种奇偶码，均是线性分组码，分别是 3 位、6 位和 10 位冗余码，各自被应用于话音消息、RACH 和 SCH 突发上。它们的生成多项式和因子分解如下：

3 位冗余码（话音）：

$$g(x) = x^3 + x + 1 \tag{2-1}$$

6 位冗余码（RACH）：

$$x^6 + x^5 + x^3 + x^2 + x + 1 = (x+1)(x^5 + x^2 + 1) \tag{2-2}$$

10 位冗余码（SCH）：

$$x^{10} + x^8 + x^6 + x^4 + x^2 + 1 = (x^4 + x^3 + x^2 + x + 1)(x^3 + x + 1)(x^2 + x + 1)$$
$$= (x^5 + 1)(x^7 + 1)/[(x+1)(x+1)] \tag{2-3}$$

2. 分组码

分组码采用的是法尔码（Fire），主要是用于检错。法尔码是一种线性码，而 GSM-R 系统用到的分组码，是缩短了的循环码，生成多项式如下：

$$g(x) = (x^{23} + 1)(x^{17} + x^3 + 1) = x^{40} + x^{26} + x^{23} + x^{17} + x^3 + 1 \tag{2-4}$$

分组码的多项式最高次数是 40，所以余数系数和荣誉比特为 40 个。

3. 卷积码

卷积码的主要功能是用来纠错。

对于不同的传输信道，卷积码分别有不同的生成多项式，如表 2-2 所示：

表 2-2 不同信道卷积码生成多项式

生成多项式	对应信道
$G_0 = 1 + D^3 + D^4$	TCH/FS, TCH/F9.6, TCH/F4.8, SDCCH, BCCH, PCH, SACCH, FACCH, AGCH, RACH, SCH
$G_1 = 1 + D + D^3 + D^4$	TCH/FS, TCH/F9.6, TCH/F4.8, SACCH, FACCH, SDCCH, BCCH, PCH, AGCH, RACH, SCH, TCH/F4.8
$G_2 = 1 + D^2 + D^4$	TCH/F4.8, TCH/F2.4, TCH/H2.4
$G_3 = 1 + D + D^2 + D^3 + D^4$	TCH/F4.8, TCH/F2.4, TCH/H2.4

信道编码分为以下两个步骤：分组编码加检错位，卷积编码输出。经过分组编码得到的 184 比特，加入 40 比特用于检错，再进行卷积编码，输出 456 比特。卷积编码用到的是 1/2 码率卷及编码，编码器采用二进制（2,1,4）卷积码编码器，如图 2-7 所示。

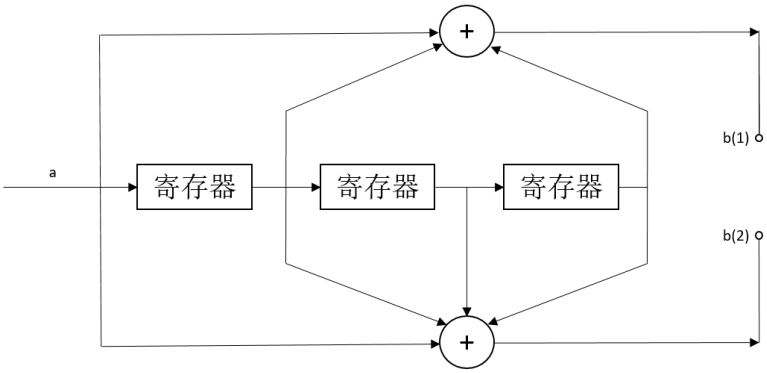

图 2-7 二进制（2,1,4）卷积码编码器

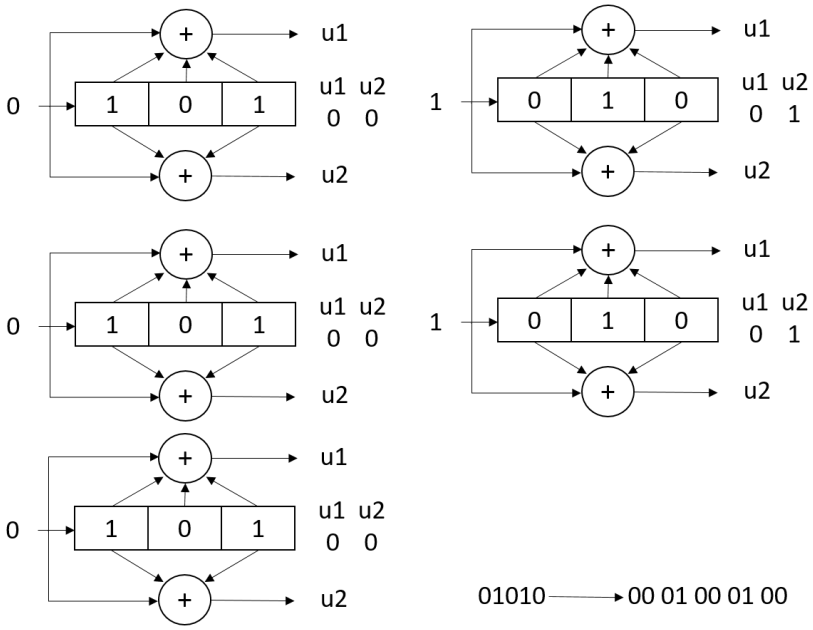

图 2-8 二进制（2,1,4）卷积码编码器编码过程

具体算法实现过程如图 2-8 所示例如，经过二进制（2,1,4）卷积码编码器卷积编码输入序列 10100 被编为 1110001011。编码器的唯一输入序列 a，经过两条不同路径达到输出端，对应两个响应序列，分别是 $g^{(1)}=(1111), g^{(2)}=(1101)$。对任意的输入序列 a，对应两个输出序列分别是 a 与 g 的离散卷积 $b^{(1)}=a*g^{(1)}, b^{(2)}=a*g^{(2)}$。

其生成多项式是

$$u^{(1)}(D)=1+D+D^3 \tag{2-5}$$

$$u^{(2)}(D)=1+D+D^2+D^3 \tag{2-6}$$

2.4.2 交织编码

交织的主要功能是把较长的连续突发差错离散成分散的随机差错，再用前向纠错编码（FEC）消除随机错误。交织深度越大，那么离散度越大，检错纠错能力也就越强，但相对的，交织编码处理时间越长，数据传输时延增大。也就是说，交织技术是以牺牲时间为代价的。

在 GSM-R 系统中，交织编码把 b 比特码字分散到 n 个突发中去。由于 NB 有 2 个 57bit 的有用信息，所以 b 和 n 最好取和 114 有关的数字进行简单运算，以 b=456bit 为例，可以按如下方式展开：

（1）4×114，填充整个序列；

（2）8×57，填充整个序列的一半；

（3）24×19，填充整个序列的 1/6;

（4）76×6，填充整个序列的 1/9；

表 2-3 概括了不同信道和传输类型的编码和交织方法。

表 2-3 不同信道和传输类型的编码与交织

信道和传输类型		输入速率（kbps）	输入码块（bit）	编码			输出码块（bit）	交织度（突发序列）
				校验比特	尾比特	卷积码率		
TCH/FS	I_a	13	50	奇偶, 3	4	1/2	456	8(半速率)
	I_b	13	132		4	1/2	456	8(半速率)
	II	13	78				456	8(半速率)
TCH/F9.6 TCH/H4.8		12 6	240 (4×60)		4	1/2	456	22 (复合)
TCH/F4.8		6	120 (2×60)		32 (2×16)	1/3	456	22 (复合)
TCH/F2.4		3.6	72		4	1/6	456	8 (半序列)

续表

TCH/H2.4	3.6	144 (2×72)		8 (2×4)	1/3	456	22 (复合)
SCH		25	奇偶, 10	4	1/2	78	1(SB)
RACH		8	奇偶, 6	4	1/2	36	1(AB)
FACCH		184	法尔码, 40	4	1/2	456	8(半序列)
SACCH SDCCH BCCH AGCH PCH		184	法尔码, 40	4	1/2	456	4(完整序列)

2.4.3 业务信道的编码交织

业务信道通常采用全速率话音业务信道（TCH/FS），其输入速率为 13kbps，每码块 260 比特持续周期 20ms。根据进入编码器前做的重要性次序评估，可以把话音消息分为I_a、I_b和II三类，分别采用不同的保护方法：

I类数据共有 182bit，I_a类有 50bit，另外 132bit 为I_b类。由于I类数据对低传输误差的需求很大，为其添加编码效率为 1/2 的卷积码保护。其中I_a类对传输误差最敏感，还要再添加 3 位冗余检测码作保护。而另外 78bit 为II类，对误差敏感度低，不需要额外保护。

1. 信道编码

（1）奇偶比特与尾比特

I_a类的错误检测尤其重要，因为只要其中任一比特的信息有差错，用户收到的噪音都会很大，可以用被干扰程度小的码块来代替坏的码块，比如将先前的码块外插。

长度为 50 比特的I_a类消息进入循环码编码器，进行（53，50，2）截短循环码编码。编码器的输入 49 项，由生成多项式$g(x)=x^3+x+1$产生系统码，得到多项式：

$$d(0)x^{52}+d(1)x^{51}+...+d(49)x^3+p(0)x^2+p(1)x+p(2) \qquad (2\text{-}7)$$

上述多项式中$p(0)$、$p(1)$和$p(2)$就是奇偶校验位。系统码除以生成多项式的余式是x^2+x+1。

将 182 比特 I 类信息$\{d(0),d(1),...,d(181)\}$和 3bit 奇偶比特$p(0)$、$p(1)$和$p(2)$进行重排。偶数和奇数信息分别位于$p(0)$、$p(1)$和$p(2)$两边，再加上 4 个“0”作尾比特，可以得到输出比特，共 189bit，表达式为：$\{u(0),u(1),...,u(188)\}$，其中：

$u(k)=d(2k)$, $u(184-k)=d(2k+1)$, $k=0,1,...,90$；

$u(91+k)=p(k), k=0,1,2$； $u(k)=0, k=185,186,187,188$。

（2）卷积编码

完成奇偶比特与尾比特的重排之后，用图 2-9 所示的编码器对这 189 比特进行码率为 1/2 的卷积码编码。

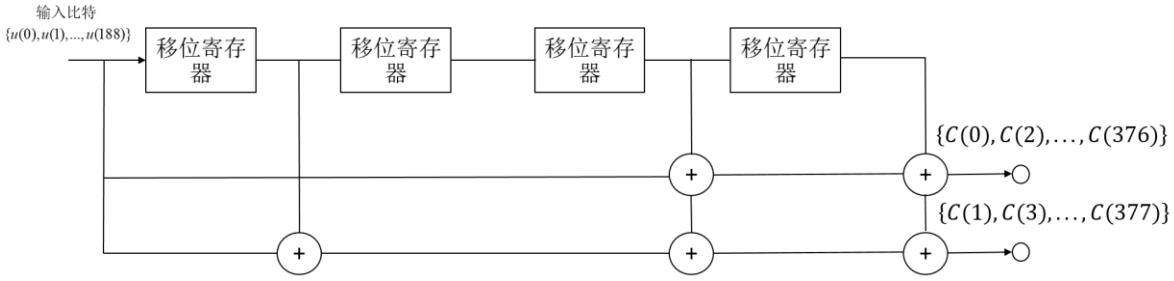

图 2-9 话音信道卷积码编码原理

卷积编码码的生成多项式为：

$$G_0 = 1 + D^3 + D^4 \tag{2-8}$$

$$G_1 = 1 + D + D^3 + D^4 \tag{2-9}$$

经过卷积编码器的输出一共 378 比特，再将 II 类的 78 比特加入，最后得到的输出编码为$\{c(0), c(1), ..., c(455)\}$，其中：

I 类：$c(2k) = u(k) \oplus u(k-3) \oplus u(k-4)$，

$c(2k+1) = u(k) \oplus u(k-1) \oplus u(k-3) \oplus u(k-4)$，$k = 0, 1, ..., 188$，

如果$k < 0$，有$u(k) = 0$；

II 类：$c(378 + k) = d(182 + k), k = 0, 1, ..., 77$。

经过比特重排和卷积编码，20ms260 比特的话音帧变成了 456 比特，速率为 22.8kbps。完整的话音业务信道误差保护如图 2-10 所示。

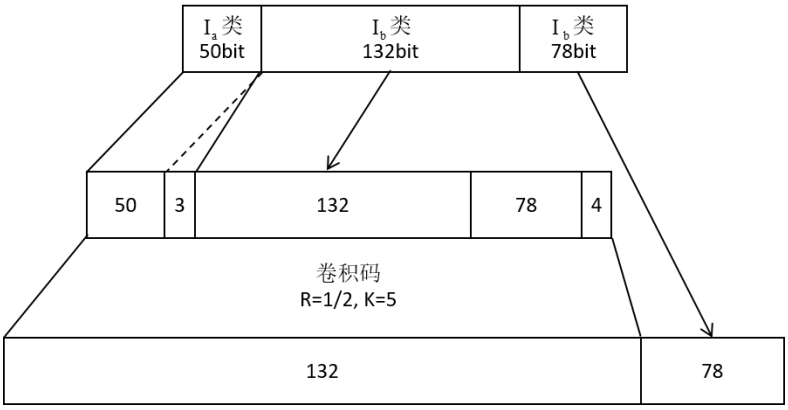

图 2-10 全速率话音的误差保护

2.交织

经过信道编码的 456bit 码块，被分成 8 组，每组 57 比特，在 8 个突发上进行交织，一个突发包含两个相邻的码块 A 和 B。为了进一步提升交织深度码块 A 占据了

突发中的奇数位置，相对的，码块 B 则占据了偶数位置。最后，一个码块被均匀得分到 8 个突发中，交织深度为 8。表 2-4 解释了交织的过程。

表 2-4 全速率话音交织规则

0	8	16	24	…	448	突发序列 N 的偶数比特
1	9	17	25	…	449	突发序列 N+1 的偶数比特
2	10	18	26	…	450	突发序列 N+2 的偶数比特
3	11	19	30	…	451	突发序列 N+3 的偶数比特
4	12	20	28	…	452	突发序列 N+4 的偶数比特
5	13	21	29	…	453	突发序列 N+5 的偶数比特
6	14	22	30	…	454	突发序列 N+6 的偶数比特
7	15	23	31	…	455	突发序列 N+7 的偶数比特
共 57 列						

因此，一个突发由以下 116 比特的编码数据组成：

（1）57 比特，取自码块 A（奇数位置）；

（2）1 比特，偷帧标志，表示数据段 1 是否用于快速随路信令方式或包含用户数据；

（3）57 比特，取自码块 A+1（偶数位置）；

（4）1 比特，偷帧标志。

偷帧标志（SF）为“0”时表示占用的是业务信道，为“1“时表示占用的是控制信道。图 2-11 和 2-12 分别表示的是话音帧交织到 TDMA 帧的过程和话音帧的分组交织。

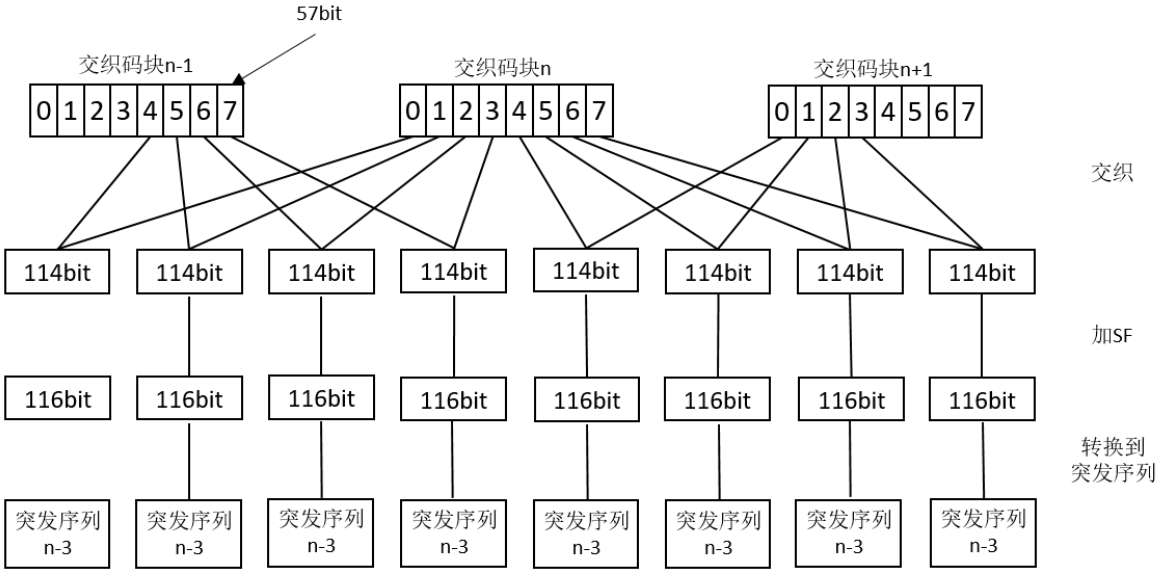

图 2-11 话音帧交织到 TDMA 帧

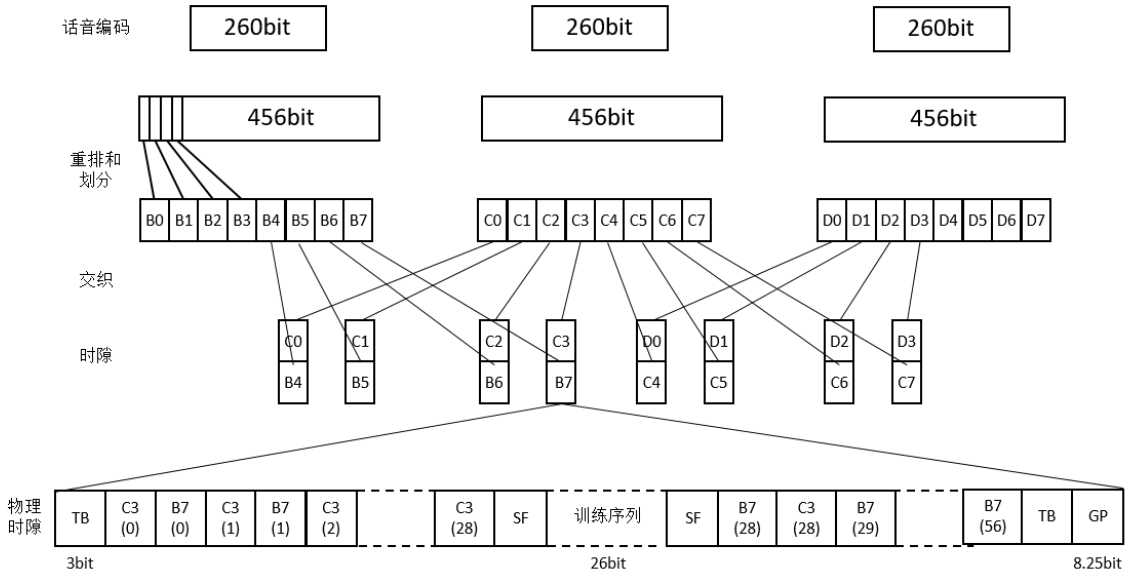

图 2-12 话音帧的分组交叉交织

2.4.4 控制信道的编码和交织

与业务信道不同的是，控制信道需要更多的 bit 用于检错。经过分组编码得到的 184 比特，加入 40 比特用于检错，再进行卷积编码，输出 456 比特。交织过程又分为块内交织和块间交织。不同的信道对应的交织方式也有所不同。图 2-13 为 BCCH、CCCH、SDCCH、FACCH、SACCH 的编码与交织过程。

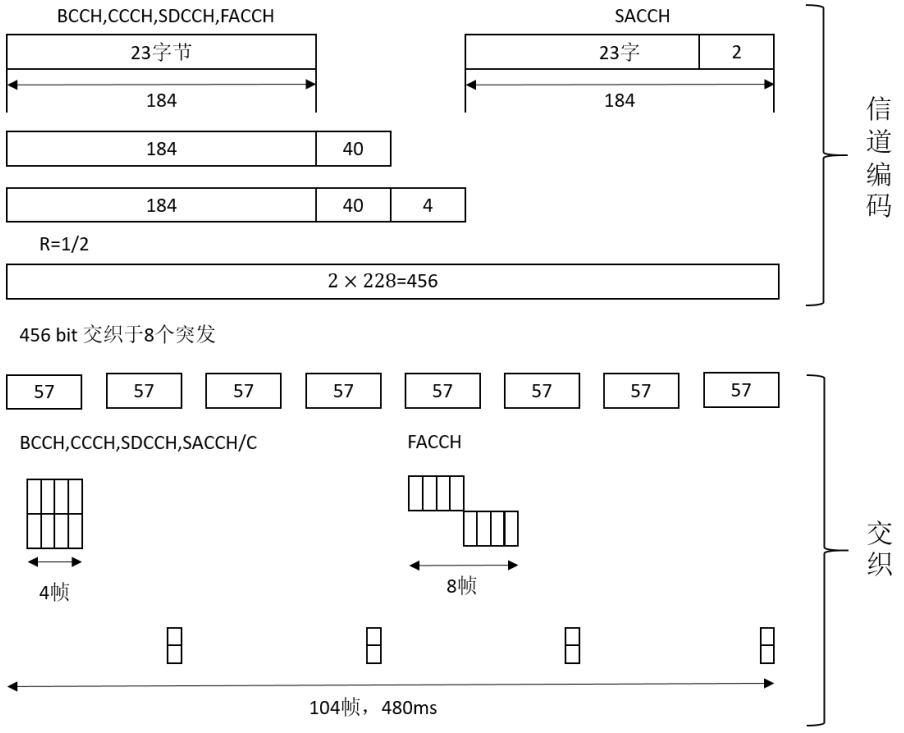

图 2-13 BCCH、CCCH、SDCCH、FACCH、SACCH 的信道编码与交织

2.5 调制与解调

2.5.1GMSK 调制

GSM-R 系统的调制方式 GMSK，是调制指数为 0.3 的二进制调频，基带信号为矩形波。通常会在调制前加入高斯低通滤波器。基带波形在滤波后，生成一种新的基带波形，具有较好的频谱特性。GMSK 调制原理方框图如图 2-14 所示

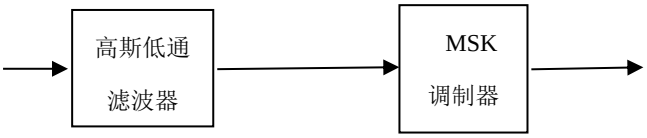

图 2-14 GMSK 调制原理方框图

为了更好地抑制 MSK 的带外辐射，高斯滤波器一般具有以下特点：

（1）带宽窄并且具有陡峭的截止特性，可以抑制高频分量；

（2）冲激响应的过冲较小，防止顺势频偏过大；

（3）调制指数为 0.3，对应的一个码元内的载波相移为 π /2。

高斯低通滤波器的传输函数为

$$H(f)=\exp(-a^2 f^2) \tag{2-10}$$

式中，a 是与高斯滤波器的 3dB 带宽 B_b 有关的常数。由 3dB 带宽定义有

$$H^2(B_b)=\frac{1}{2} \tag{2-11}$$

即

$$\exp(-2a^2 B_b^2)=2^{-1} \tag{2-12}$$

所以

$$aB_b=\sqrt{\frac{1}{2}\ln 2}\approx 0.5887 \tag{2-13}$$

由此可见，改变 a, B_b 将随之改变。

滤波器的冲激响应为

$$h(t)=\frac{\sqrt{\pi}}{a}\exp\left[\left(-\frac{\pi}{a}t\right)^2\right] \tag{2-14}$$

$h(t)$ 是非时限的，相邻脉冲会产生重叠，它随 t^2 按负指数规律迅速减小，所以近

似认为冲击宽度是有限的。

可以看出 GMSK 滤波器最重要的参数是 3dB 基带带宽 B_b 和基带码元间隔 T_b。如果输入为双极性不归零矩形脉冲序列 $s(t)$：

$$s(t)=\sum_n a_n b(t-nT_b), a_n=\pm 1 \tag{2-15}$$

式中，

$$b(t)=\begin{cases}\dfrac{1}{T_b}, 0\le|t|\le\dfrac{T_b}{2}\\ 0, \quad \text{其他}\end{cases} \tag{2-16}$$

其中，T_b 为码元间隔。高斯滤波器的输出为

$$x(t)=s(t)*h(t)=\sum_n a_n g(t-nT_b) \tag{2-17}$$

式中，$g(t)$ 为高斯滤波器的脉冲响应：

$$g(t)=b(t)*h(T)=\frac{1}{T_b}\int_{T_b-T_b/2}^{T_b+T_b/2}h(\tau)d\tau=\frac{1}{T_b}\int_{T_b-T_b/2}^{T_b+T_b/2}\frac{\sqrt{\pi}}{a}\exp\left[-\left(\frac{\pi\tau}{a}\right)^2\right]d\tau \tag{2-18}$$

GMSK 信号的表达式为

$$s_{GMSK}(t)=\cos\left\{\omega_c t+\frac{\pi}{2T_b}\int_{-\infty}^{t}\left[\sum a_n g\left(\tau-nT_b-\frac{T_b}{2}\right)\right]d\tau\right\} \tag{2-19}$$

式中，a_n 为输入数据。

高斯滤波器的输出序列经过 MSK 调制得到 GMSK 信号，经过高斯滤波后的脉冲信号边缘更为平滑，无陡峭沿，也无拐点。

由于相邻脉冲间有重叠，因此，在决定一个码元内的脉冲面积时，要考虑相邻码元的影响。这样，在不同的码流图案下，会使一个码元内脉冲面积不同，因而对应的相位路径也不同。

GMSK 信号可以表示为正交形式，即

$$s_{GMSK}(t)=\cos[\omega_c t+\varphi(t)]=\cos\varphi(t)\cos\omega_c t-\sin\varphi(t)\sin\omega_c t \tag{2-20}$$

式中

$$\varphi(t)=\frac{\pi}{2T_b}\int_{-\infty}^{t}\left[\sum a_n g\left(\tau-nT_b-\frac{T_b}{2}\right)\right]d\tau \tag{2-21}$$

2.5.2GMSK 解调

GMSK 的解调方式有两种，分别是相干解调和 1 比特差分解调，由于 1 比特差分解调的实现相对来说要简单的多，所以本设计采用了 1 比特差分解调的方法。根据 1 比特差分检测算法，可以找到在 1 比特周期内接收到的信号相位的变化量。这种相位方面的改变量为：

$$\Delta\varphi(t)=\frac{\pi}{2}\int_{t-T_b}^{t}\left[d(t)*h(t)\right]dt \tag{2-22}$$

从上式可以看出，$\Delta\varphi(t)$ 的值没有超过 T_b，所以在 1 比特周期内相位变化的最大值为 $\left[\Delta\varphi(t)\right]_{\max}=\frac{\pi}{2}$。

如果 $z(t)=I(t)+jQ(t)=A_r e^{j\varphi(t)}$，$A_r$ 是接收到信号矢量的幅值，信号相位的改变量为 $D(t)=\Delta\varphi(t)=img\left[z(t)*(t-T_b)\right]$，D(t)表示解调的波形。对接收到的 I、Q 支路分量的基带信号通过模数转换后，可以采用 1 比特差分检测算法。

通过 1 比特差分检测算法，我们可以找到传输的码元，在 1 比特周期时间内的相位变化量。这种相位的改变量可以表示为：

$$\Delta\varphi(t)=\varphi(t)-\varphi(t-T_b) \tag{2-8}$$

上式可以用图 2-15 所示的原理来实现：

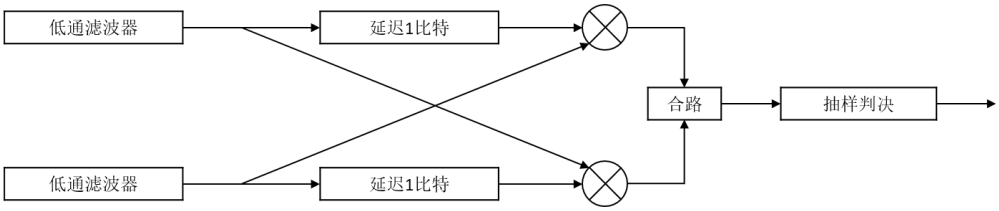

图 2-15 1 比特差分检测

当 $\Delta\varphi(t)$ 大于等于 0 时，接收到的数据是“1”；当 $\Delta\varphi(t)$ 小于 0 时，接收到的数据是“0”。1 比特差分检测算法的解调过程如图 2-16 所示：

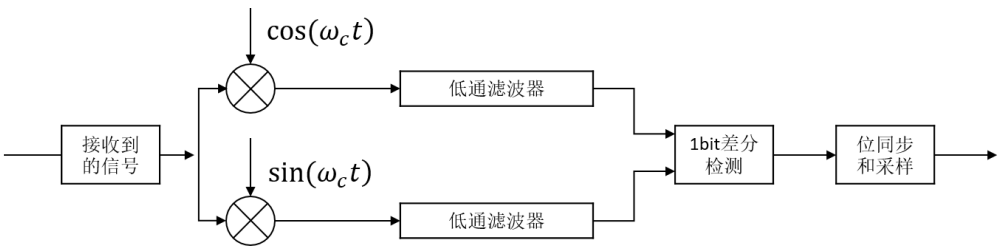

图 2-16 GMSK 信号解调框图

2.6 物理层与高层间关系

2.6.1 物理层接口与原语

通过接口和服务原语（PH-Primitives），物理层（Physical Layer）可以访问数据链路层（Data Link Layer）、无线链路控制和介质访问控制层（Radio Link Control and Medium Access Control Layer）以及应用程序的支持功能单元（Function Units）。图 2-17 是物理层与高层的接口关系。

（1） 与 L2（数据链路层）的接口

物理层与数据链路层接口，在此接口上支持传输控制信道消息。物理层和数据链路层之间的通信是通过服务原语来执行的抽象方式，在 3GPP 中并没有对它们实现限制。在 L1 和 L2 之间交换的服务原语用于 L2 帧的传输，还可以用于指示建立到 L2

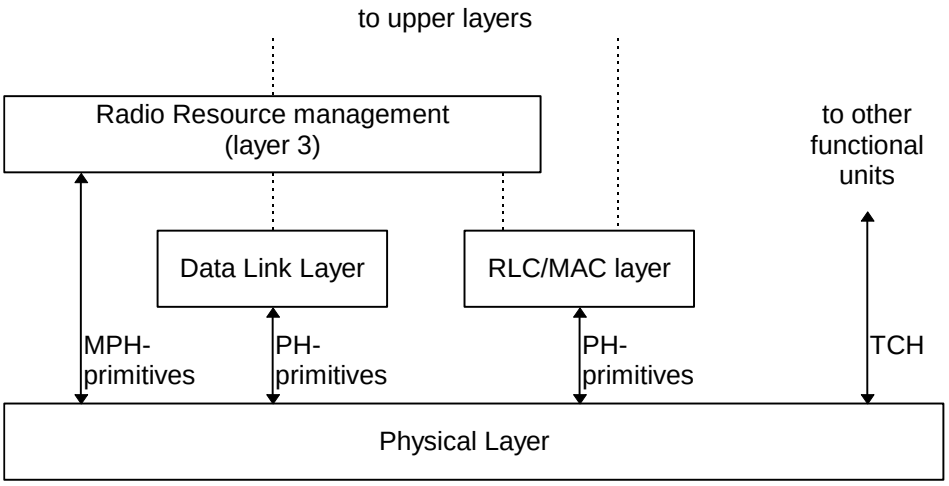

图 2-17 物理层接口

的信道。在 GSM-R 空口协议软件设计中，我们通过调用函数传输数据快来实现服务原语的功能。

（2） 与 RLC/MAC 的接口

物理层与无线链路控制和介质访问控制（RLC/MAC）层的接口，支持分组数据控制信道和分组数据业务信道。在物理层和 RLC/MAC 层之间交换的服务原语用于 RLC/MAC 块（Block）的传输，并指示将分组数据物理信道建立到 RLC/MAC 层。

（3） 与 L3 的接口

物理层在移动台 MS 和层 3 的无线资源管理（RR 子层）实体，通过服务原语（MPH-Primitives）来实现。与 RR 子层交换的原语与信道的分配、物理层系统信息（包括测量结果）等有关。

物理层与数据链路层 Dm 信道的交互以及与 RLC/MAC 层的分组数据物理信道交互以原语的形式表示，其中原语表示物理层和相邻层之间的信息和控制的逻辑交换。

它们不指定或约束实现。对于物理层，定义了两组原语：

PH：物理层与数据链路层和 RLC/MAC 层之间的原语；

MPH：L1 与 L3 RR 子层之间的原语。

2.6.2 物理层服务

在 OSI 参考模型中，层的服务接入点（SAPS）被定义为门，通过该门将服务提供给相邻的更高层，如图 2-18 所示。通过 SAP，物理层为数据链路层提供服务。SAP 既用于控制服务提供实体（在这是物理层的情况下，也与信道的建立和释放相关的命令）和数据传输（在物理层内为比特的传输）。L3 的 RR 子层代替数据链路层控制信道的建立和释放。在 GSM 系统的物理层上，SAP 被定义在每个控制信道的物理层和数据链路层之间，如图 2-19 所示。此外，在 GSM 系统的物理层上，SAP 被定义在物理层和 RLC/MAC 层之间，用于分组数据控制信道和分组数据业务信道，如图 2-20 所示。

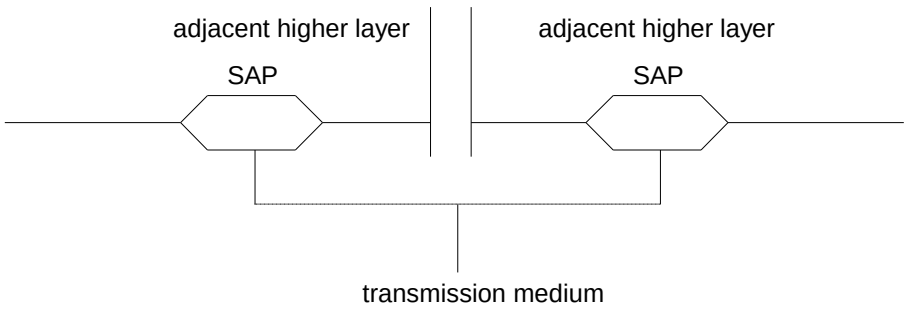

图 2-18 服务接入点原理

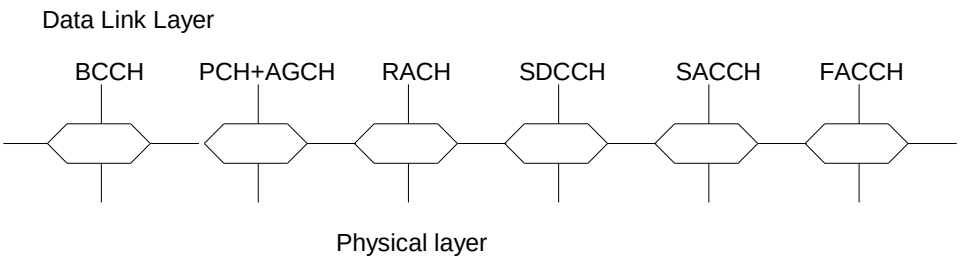

图 2-19 移动台 MS 中物理层与数据链路层之间的服务接入点

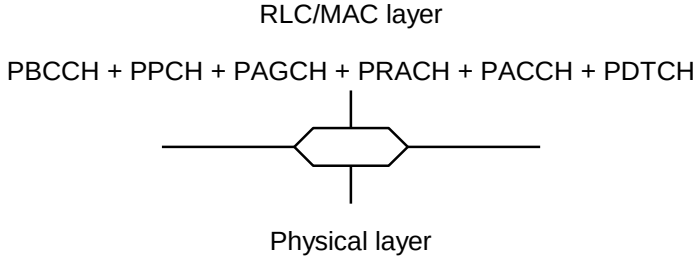

图 2-20 移动台 MS 中物理层与 RLC/MAC 层之间的服务接入点

物理层给高层提供的服务包括：

（1）访问能力（Access capabilities）。物理层在有限的逻辑信道集合上提供传输

服务，逻辑信道在物理信道上复用。一些物理信道由网络保留用于共同使用（例如CCH 和 BCH 的组合），其他被分配到与 MSS（专用物理信道）的专用连接，或者被分配给 MSS 之间用于分组交换数据业务（分组数据物理信道）共享使用。在时间上，在分配的物理信道上使用的逻辑信道的组合可以改变，在控制信道的 SAPS 上的数据以离散的块交换，其大小为 23 或 21（SACCH）八位字节。分组数据业务信道的 SAP 上的数据以 184, 271, 315 或 431 位的大小的离散块交换。

（2）错误检测（Error Detection）。物理层提供了一个错误保护的传输服务，它包括错误检测功能和较低级别的纠错功能。错误的接收帧不提供给数据链路层或 RLC/MAC 层。

（3）加密。对业务信道数据进行加密。

2.7 本章小结

本章首先引出了 GSM-R 系统组成，引出本文重点——空中接口，即 MS 和 BTS 之间的通信接口。重点介绍了物理层用到的各项技术，主要是 TDMA 时分多址，尤其是信道的分类和映射等，还有信道编码和交织技术等，结合 3GPP 发布的协议标准，简单分析了物理层与高层的通信。

第 3 章 GSM-R 空中接口协议软件物理层设计与实现

3.1 系统设计要求

3.1.1 功能需求

GSM-R 系统物理层的功能如表 3-1 所示。

表 3-1 物理层功能描述

功能需求	内容
基于系统消息的小区配置（业务功能）	1） 小区参数的配置：小区频率列表、邻小区频率列表、CCCH 的配置参数、小区重选参数以及位置区码等参数的设置 2） 小区参数的设置
无线资源的规划	1） 帧同步：SB 和 NB 帧号的计算 2） 信道映射：对于不同小区配置映射不同的逻辑信道
信道解码和交织	1） 完成物理层基带突发序列的组合与交织 2） 信道解码，包括 BCCH、CCCH、SDCCH、SACCH 和 FACCH 的信道解码

3.1.2 开发平台环境介绍

本文所介绍的 GSM-R 空口协议软件物理层模块的设计，基于 Visual Studio 2017 和 MATLAB R2014a 平台编程。其中，信道编码和信道解码两个模块主要是处理的比特流，即“0”、“1”数组，且为了和高层直接更好的匹配，对于我也更好理解，所以采用了 C 语言来编写代码；而调制解调模块，因为在载波同步时要调用 FFT（快速傅里叶变化）函数来做频谱估计，以及读取采样数据更方便的原因，用到了 MATLAB。

3.2 系统模块划分

在 3GPP 协议中，规定了 GSM-R 空口协议的模型，按层从下往上可以将系统分为四个模块：物理层模块、数据链路层模块、网络层模块和应用层模块。其中物理层是 OSI 七层模型的最底层，物理层基带处理模块在接收端主要完成的功能如下：

（1）解调信号。现有的数据是经过抽样放大的离散信号，要将该数据转换为“0”、“1”比特，才方便编程处理。

（2）完成同步，包括载波同步和位同步。先通过 FFT 算法找到 FB 所在的大概位置，实现载波同步，然后解析基站（BTS）下行发送的 SB 中的 帧号（FN）和基站号

（BSIC），同步基站的 FN；

（3）完成信道映射，即对接收到的突发序列做物理信道到逻辑信道的映射；

（4）处理比特流形式的信道解码与解交织。

任务设计中要完成接收端的信号处理，将 I、Q 支路的离散信号转换为含有 184bit 的 LAPDm 数据链路层数据格式，通过服务原语传给数据层进行进一步的分析。为了更好得完善物理层的整个系统，除了接收端，本文还编程仿真了发送端的整个过程，并在 MATLAB 上实现了调制解调过程。物理层可以划分为以下子模块，如图 3-1 所示。

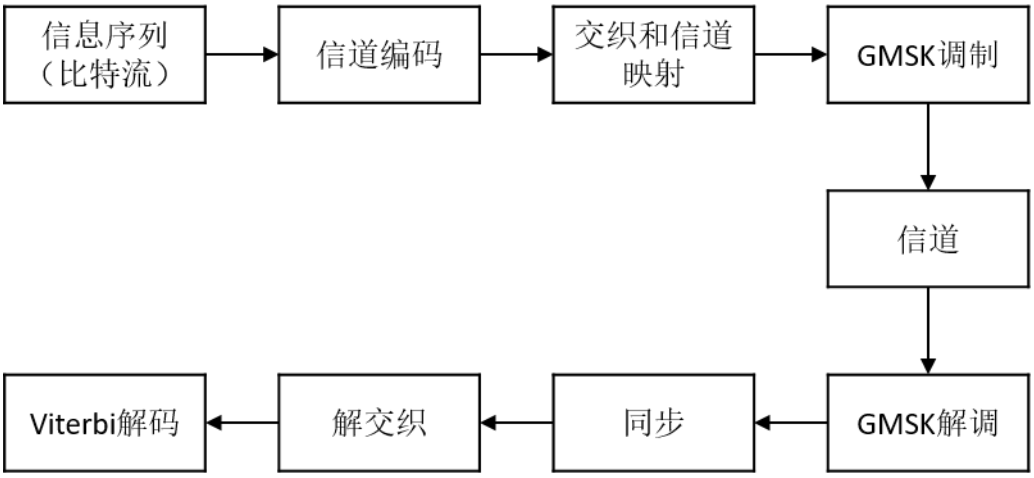

图 3-1 物理层子模块划分

接收端可以看作是发送端的逆向过程，所以在具体的程序设计时，可以将信道编码和解码、交织和解交织的模块合并。另外，交织分为块内交织和块间交织，信道映射会在块间交织的部分完成。同步搜索又分为两步，分别是 FB 搜索实现初定位，SB 搜索实现精确定位，为了减小噪声干扰，会在解调之前进行 FB 搜索。具体的代码结构如图 3-2 所示

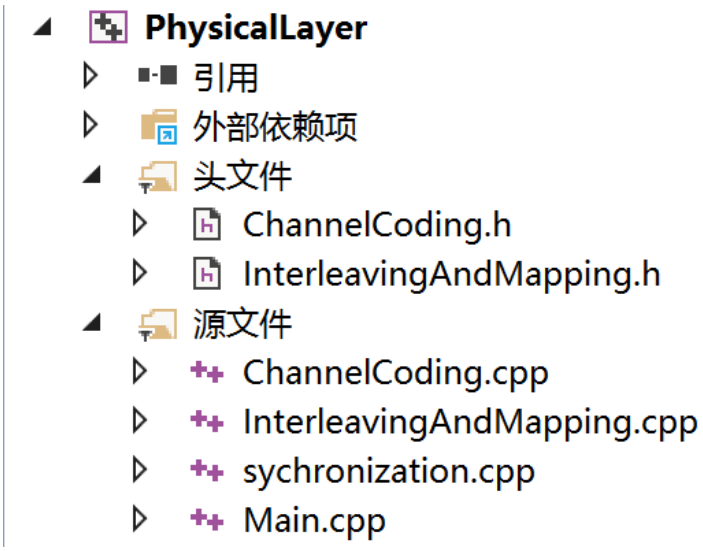

图 3-2 代码结构

3.3 发送端

3.3.1 信道编码模块

卷积编码函数 encode 参数设置如表 3-2 所示：

表 3-2 卷积编码函数参数设置

变量名	参数类型	作用
*symbols	unsigned int	编码输出
*data	unsigned int	编码输入
nbytes	unsigned int	nbytes=n/16,n 为实际输入码字的数目
startstate	unsigned int	定义初始化状态
a0	unsigned int	初始 a0=0，寄存器 a0 用来存放下一个输入量
a1	unsigned int	初始 a1=0，寄存器 a1 用来存放 a0 上一个输入量
a2	unsigned int	初始 a2=0，寄存器 a2 用来存放 a1 上一个输入量
a3	unsigned int	初始 a3=0，寄存器 a3 用来存放 a2 上一个输入量

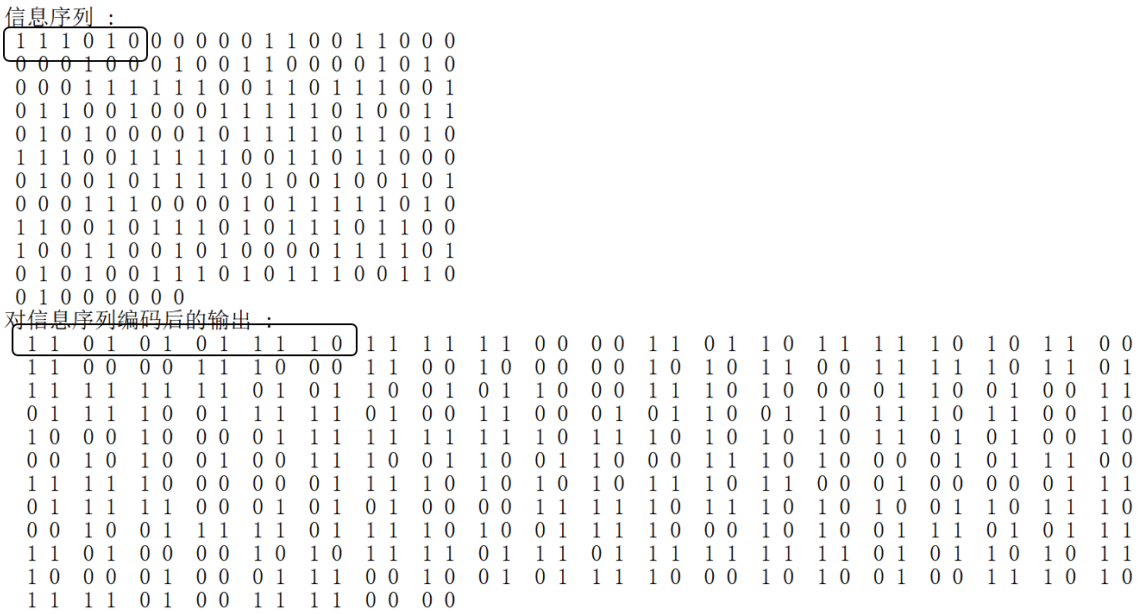

图 3-3 编码输出序列程序运行图

图 3-3 是编码输出的程序运行图，信息序列中，前 184bit 是有用信息，后面的 40bit 是检错位，最后 4bit 用于奇偶校验。例如，经过二进制（2,1,4）卷积码编码器卷积编码输入序列 111010 被编为 11 01 01 01 11 10，与理论分析的结果一致。

3.3.2 交织与信道映射模块

本系统设计的交织分为两个步骤，分别是块内交织和块间交织。下面以 BCCH 信道消息为例，块内交织先将经过信道编码得到的 456bit 交织于 8 个突发块，在程序中就是得到 8 个1×57 的数组，如图 3-4 所示。

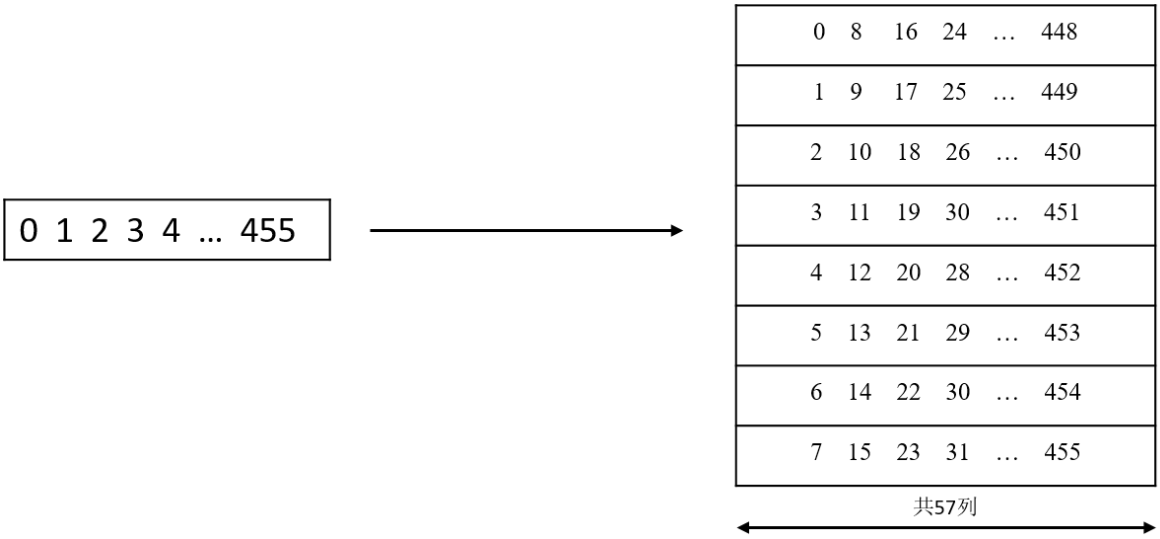

图 3-4 块内交织

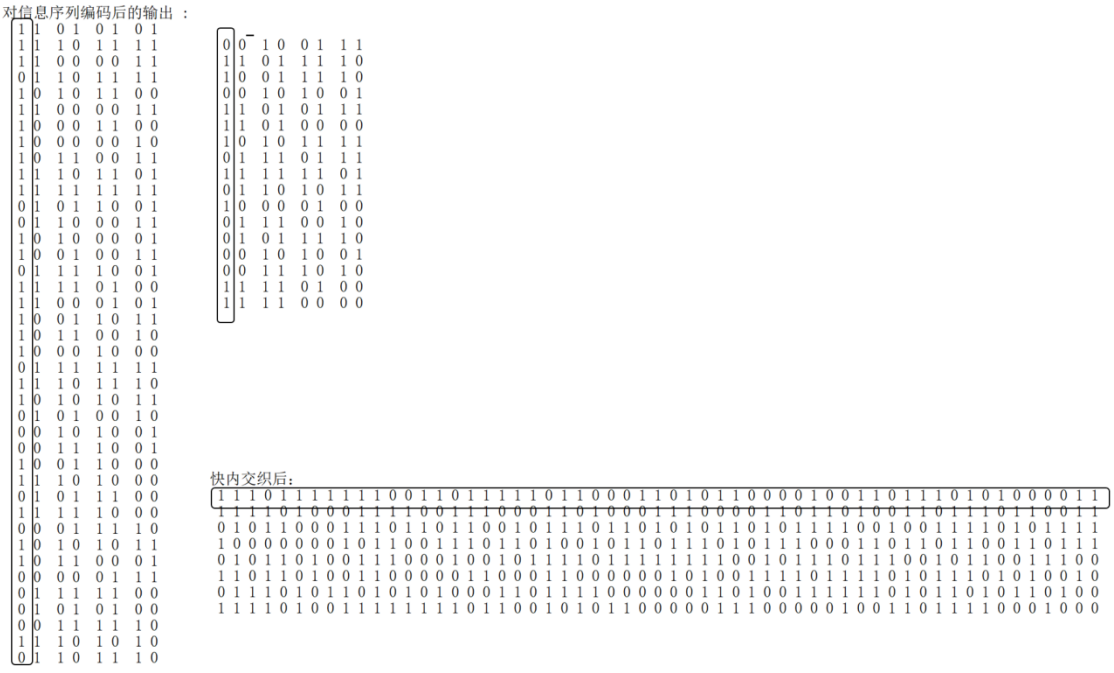

图 3-5 块内交织程序运行图

图 3-5 所示结果，信息序列每隔 8 个 bit 被交织到同一个块中，运行图中被圈起来的是 Block1 中的内容。完成块内交织后，进行块间交织，块间交织得到 4 个 2×57 的数据块分别为块内交织得到的 n 和 n+4 块，这两块正好对应了 NB 突发中的 114（两

个 57）bit 的数据位，如图 3-6 所示。然后按照 NB 的格式，填充尾比特、训练序列、偷帧标志、保护间隔等，构成 156.25bit 的一个突发，图 3-7 为块间交织运行结果，图中圈起来的内容对应的是 NB 的各个结构。最后，每个突发放在每一个帧的 0 时隙（TS0）中，其余 TS1-TS7 存放业务信息。一个 456bit 的 BCCH 数据要占用 4 帧的长度，CCCH、SDCCH 和 SACCH/C 都是如此，只有 FACCH 比较特殊，需要占用 8 帧的长度。

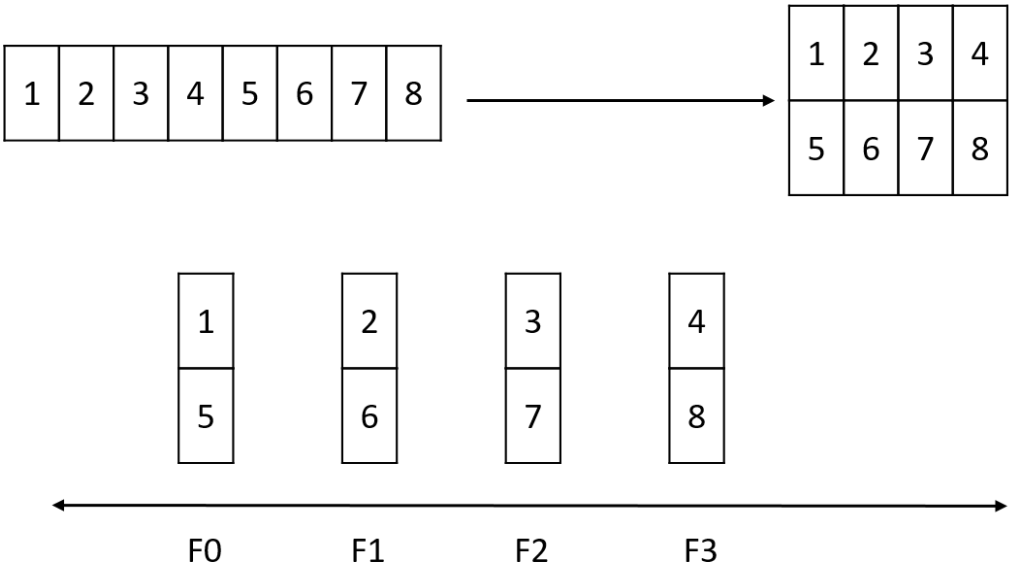

图 3-6 BCCH 块间交织过程

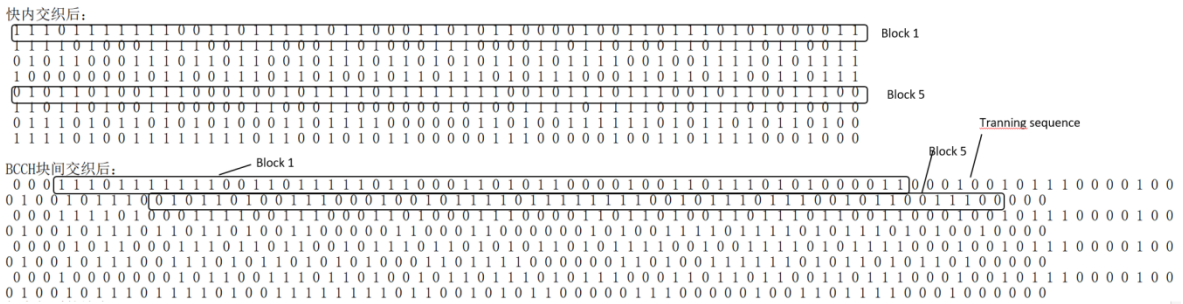

图 3-7 BCCH 块间交织程序运行结果

完成交织，以及各信道的消息映射之后，把所有控制信息放在 TS0 上，所有业务信息放在 TS1-TS7 上，封装成帧，按照第二章中提到的不同信道组合的方式，组成比特流进行调制。

3.3.3GMSK 调制模块

在 GMSK 调制仿真过程中，本设计一共要用到如表 3-3 所示的仿真参数设置，其中 3dB 基带带宽 B_b 和基带码元间隔 T_b 主要用来设计高斯滤波器。仿真中对基带信号码元逐一进行相位计算，然后通过相位和载波频率得到 GMSK 调制波形，图 3-8 为 GMSK

调制处理流程。

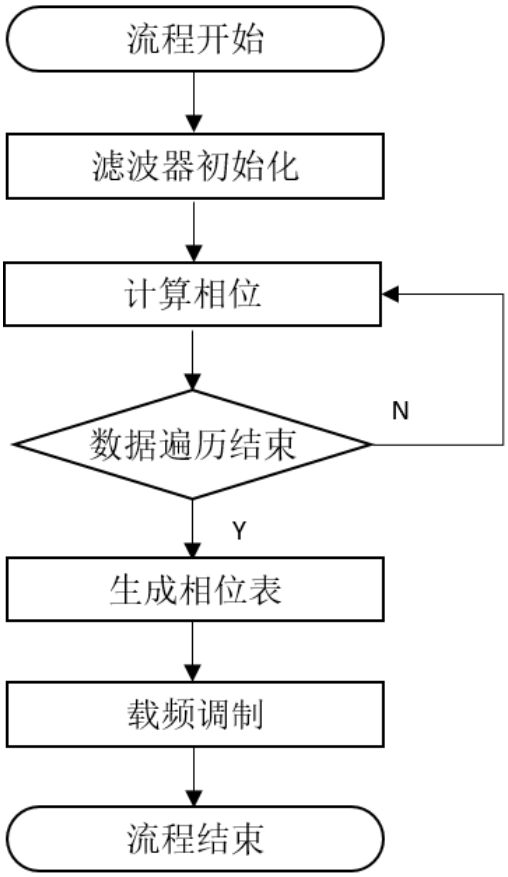

图 3-8 GMSK 信号调制流程

表 3-3 GMSK 调制参数设置

参数	设定值	含义
Ak	textread('C:\Users\MAC\Desktop\GSML1\data\interleaving.txt');	基带信号
Ts	1/16000	基带信号周期，16KHz
Tb	1/32000	输入信号周期，32KHz
BbTb	0.3	GMSK 调制系数
Bb	BbTb/Tb	3dB 带宽
Fc	32000	载波频率
F_Sample	64	每载波采样点数
B_num	Length(Ak)	基带信号码元数
B_sample	F_sample*Fc*Tb	每基带码元采样点数 B_sample=Tb/Dt
Dt	1/Fc/F_sample	采样间隔

GMSK 调制过程中的基带波形、相位波形和调制波形如图 3-9 所示。

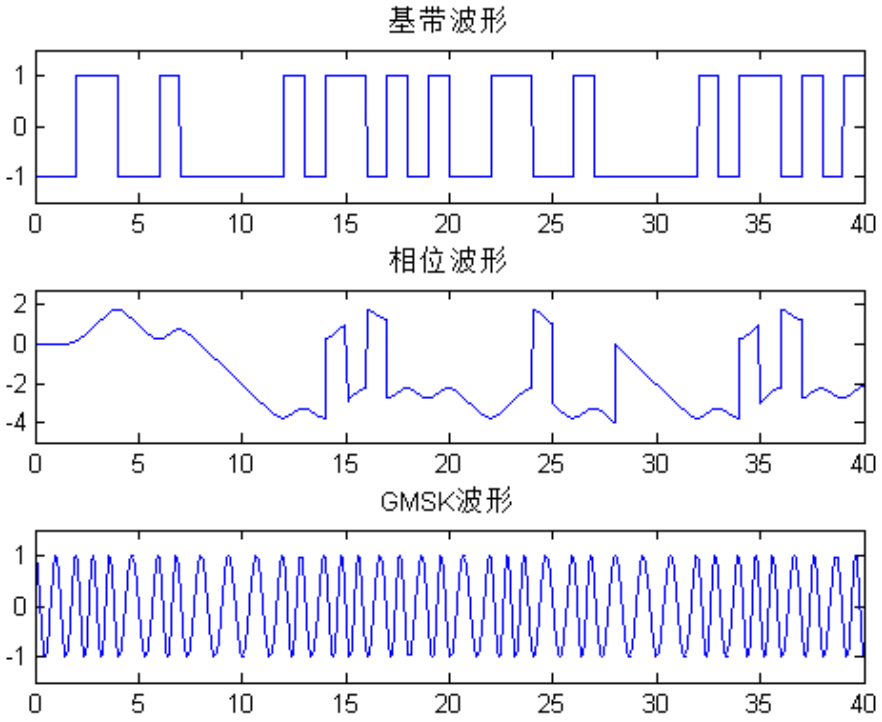

图 3-9 调制过程波形

经过交织和映射得到的数据，存放在 data 文件夹中，然后在 MATLAB 中读取。为了清楚得分析调制过程，图 3-9 给出了片段的基带波形、相位波形和调制波形。

3.4 接收端

3.4.1GMSK 解调模块

在接收端，调制后的 GMSK 信号经过数字下变频后恢复成 I、Q 两路信号后，可以运用 1 比特差分检测进行解调。在解调模块的仿真中，本文先设计了一个 n 阶巴特沃斯模拟低通滤波器，参数设计如表 3-4 所示。

表 3-4 低通滤波器设计参数

参数	设定值	含义
Fs	10000	采样频率（由奈奎斯特准则得到）
rp	3	通带纹波不超过 rp dB
rs	50	阻带衰减至少 rs dB
wp	$2\times pi\times 50$	通带截止频率
ws	$2\times pi\times 800$	阻带截止频率

根据 1 比特差分解调法，设计了如图 3-10 的解调模块。对于调制脉冲序列，延迟 1bit,移相 pi/2，两个信号相乘之后进行低通滤波、抽样判决，得到解调信号。

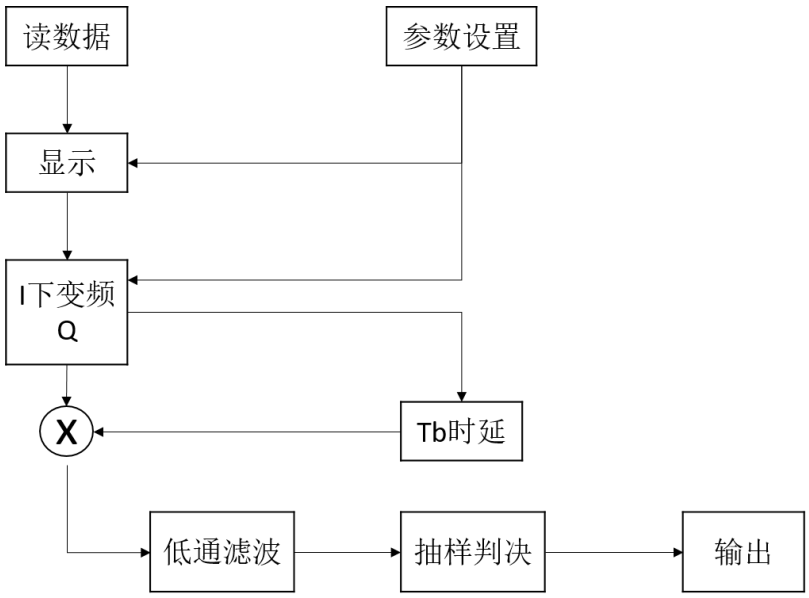

图 3-10 1 比特差分解调流程

图 3-11 和图 3-12 为解调整个过程各步的波形图。分析结果可以看出仿真的低通滤波器性能好，解调得到的信号能还原得到 3.3.3 中输入的基带信号。

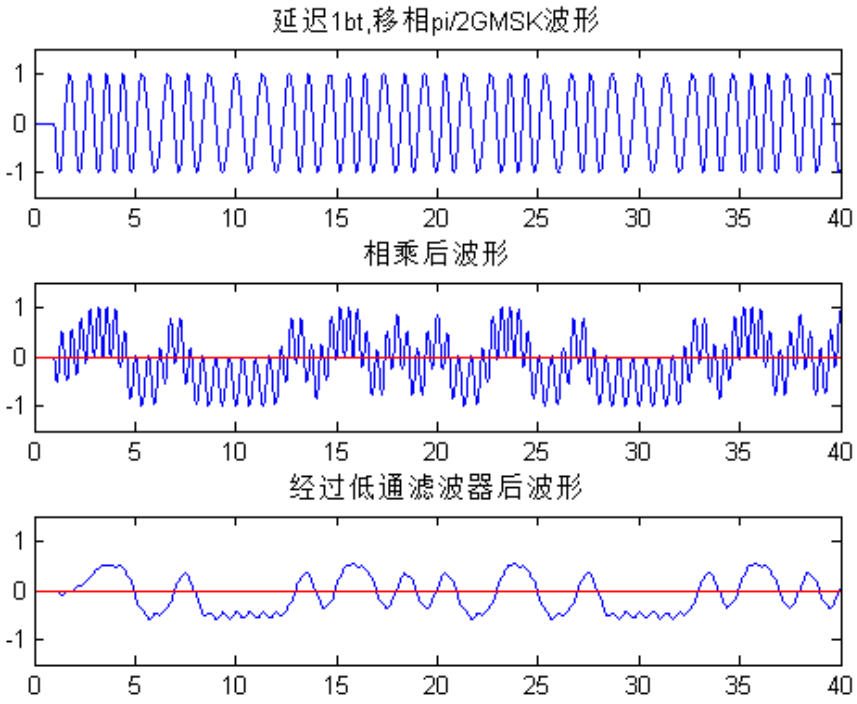

图 3-11 解调过程波形图 1

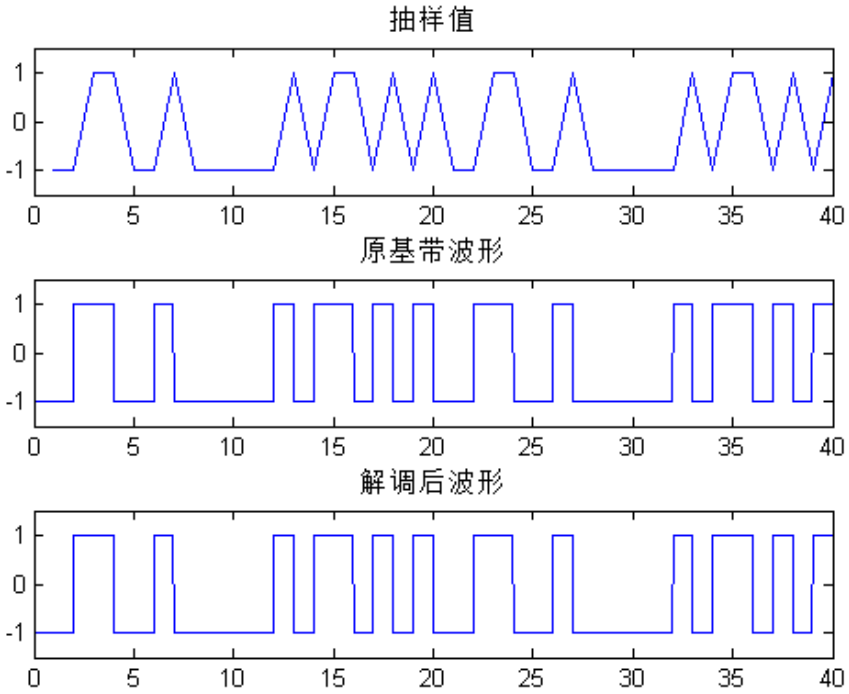

图 3-12 解调过程波形图 2

3.4.2TDMA 帧同步模块

同步搜索主要分成两步，先通过 FB 获得粗时间同步，并进行频偏矫正；再由初同步估计 SB 所在位置，进行精确时间同步，通过 SB 的加密数据波分获得基站号 BSIC，帧号。

3.4.2.1 载波同步

为实现载波同步，就要用到频率校正突发 FB。FB 分别位于含 51 帧的复合帧中第 0, 10, 20, 30, 40 帧的 0 时隙处，下面以 BCCH 和 CCCH 的复合信道为例，完整的时序结构图如图 3-13 所示，其中 F,S,B,B,B,B,C,C,C,C 分别表示每帧的第一个时隙。

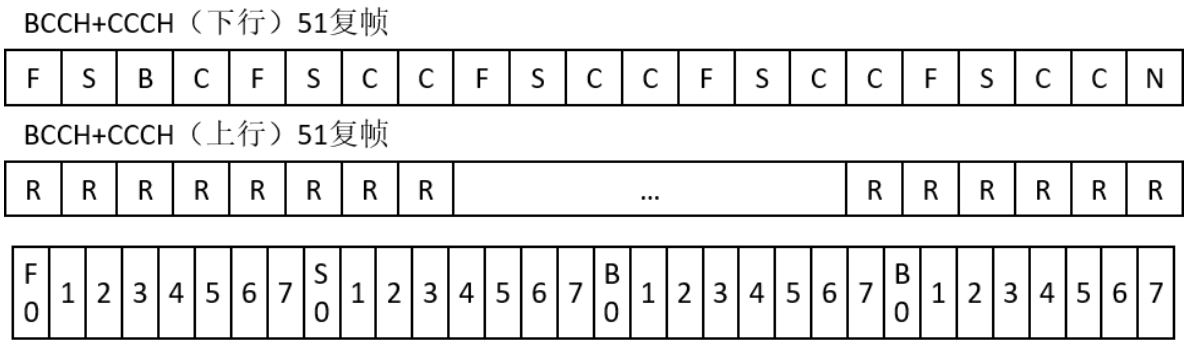

图 3-13 公共信道（IV）时序结构图

由于 GSM-R 系统采用的调制方式是 GMSK，根据上文所述，FB 数据部分发送的是 142bit 的“0”。因此在时域上看到的是一段频率为 67.7083KHZ 的正弦波。我们开窗对 FIR 输出的经过两倍采样的 I，Q 数据分别做 256 的点 FFT 变换，求出频域 256 点的 I 和 Q 的平方和，记为能量。在每 25000 个采样点求出能量最大值及应的窗口值，把对应的点的序号看成是 FB 的大概起始点，完成初定位。上式中各变量的含义如表 3-5 所示，粗同步算法流程图如图 3-14 所示。

表 3-5 粗同步参数设置

参数	设置值	含义
B_length	156.25	每个突发对应 bit 数
n	2	奈奎斯特采样率
t	8	每帧有 8 个时隙
d	10	每两个 FB 至少间隔 10 帧

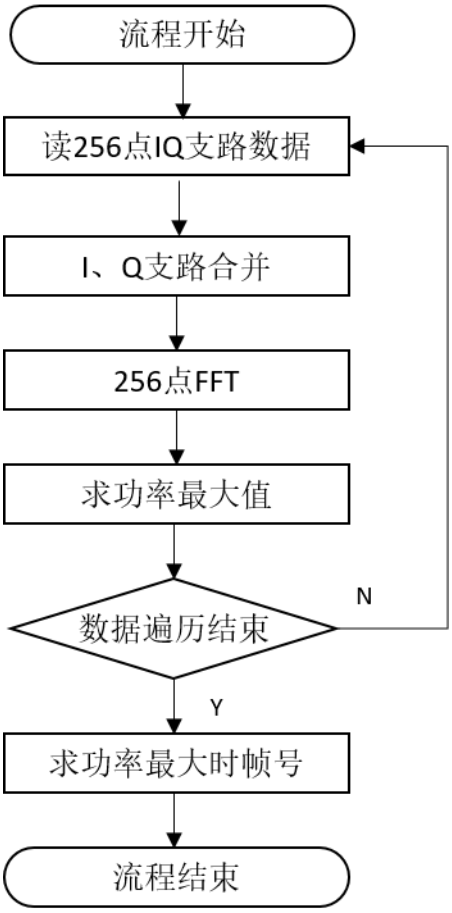

图 3-14 粗同步处理流程

每个窗口值都可以对应得到一个 256 点的频谱图，如图 3-15 所示，可以看到频谱图是对称的，且有一个明显的冲击响应。对应的，把每个窗口中对应的最大值取

出来，然后在整个抽样点上描出来，会发现有明显的峰值，此处就是 FB 所在位置。本文会在第四章数据分析中进一步分析。

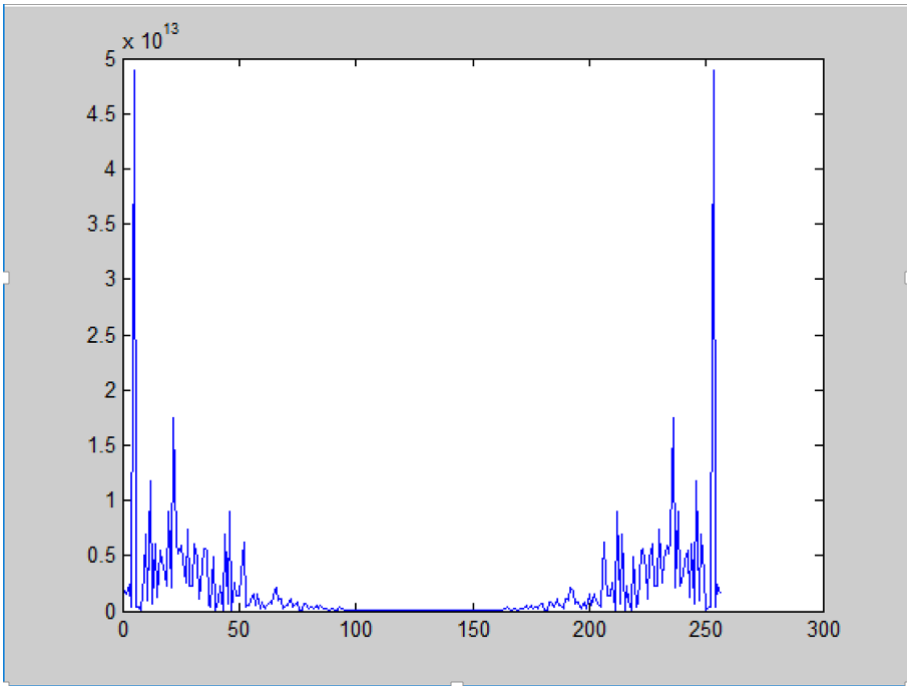

图 3-15 256 点 FFT 频谱

3.4.2.2 帧同步

利用 FB 完成初定位后，根据上节所述帧结构，SB 在 FB 之后一帧的地方，可以粗略估计 SB 的位置。然后结合 SB 同步比特的特点完成精确定位。

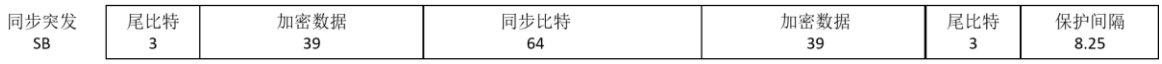

图 3-16 同步突发结构

图 3-16 中，64 位同步比特是已知的序列，在 3GPP 0502 中做了规定，它具有比较强的自相关能力。本文利用这个特性，开窗进行同步比特相关的方法就可以大概找到 SB 的起始点了。图 3-17 为滑窗操作的示意图。

图 3-17 滑窗操作

如图 3-16 所示，每个 SCH 信道有 25bit 的加密消息字段，其中 19 比特是帧号，6 比特用于 BSIC 基站号。每个单独的 SCH 时隙都携带着一个完整的同步消息，而且 SCH 的突发脉冲的消息位的字段是 78bit，因而需要将这 25 比特的数据编码成 78bit。具体的方法是将这 25 个比特的数据再加上 10 个奇偶校验比特和 4bit 的尾位（TB），得到 39bit。再将这 39 个比特按照 1：2 的卷积编码速率编码，就得到了 78 个比特的

消息。

在 SB 的起始点后，我们就可以得到 156 点 SB 的数据。再送入解调模块（GMSK 解调）就可以解调出上图中 78bit（39+39）的同步信息比特，再经过解交织和 CRC 校验就可以得到 25 比特 SCH 信道的消息字段，进而知道帧号以及 BSIC 号。在得到这些同步消息后，就可以进行 BCCH CCCH,SDCCH 的解析了。

图 3-18 是同步处理过程。具体步骤如下：

（1）得到经过粗同步的数据，通过和 SB 拓展训练序列的自相关运算，得到 SB 的确切位置；

（2）从第一个有效的 SB 开始，进行帧同步；

（3）对于接收到的 SB，有两种可能，第一种是 SB 有效，那么就将所有小区频点的 FN 和本地 SB 的 FN 更新；第二种是 SB 无效，那么根据 BCCH 的特点，得到本地 SB 的 FN，再去更新小区内其他频点的 FN；解析 SB 要用到 Viterbi 解码和 CRC 校验。计算方法满足，若上个 SB 满足 $FN\%51==41$，那么当前 FN 等于 FN+11，否则为 FN+10；

（4）对于收到的 NB，首先更新该频点的本地计数帧号，再得到 NB 的 FN。

同步过程用到的自相关函数 xcorr 的输入参数如表 3-6 所示。

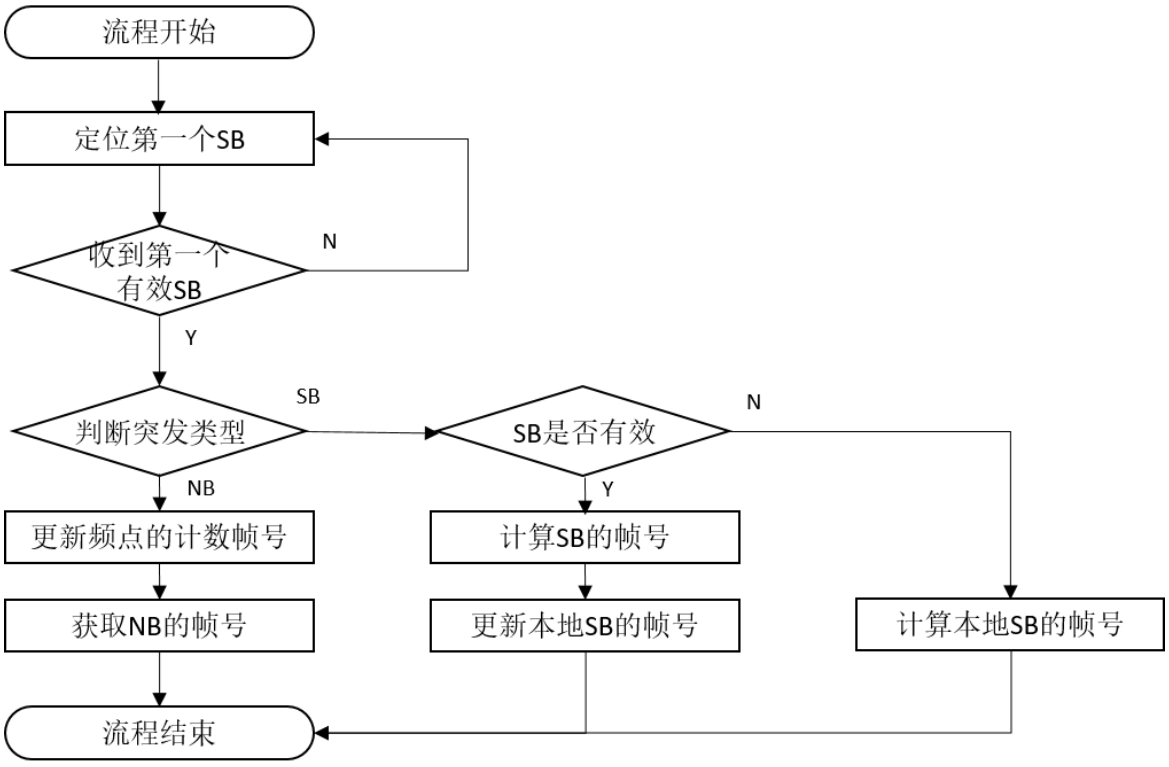

图 3-18 帧同步模块处理的过程

表 3-6 xcorr 输入参数

参数	参数类型	含义
*r	float	自相关度
*x	short	待测序列 x
*y	short	已知序列 y
N	int	序列长度

经过解调之后得到的数据，开窗与 64 位训练序列逐位进行自相关运算，得到的同步位置 rn 即 SB 中的自相关序列的起始位置，位同步的程序测试如图 3-19 所示。在点 310 前后按照 SB 的格式，可以找到 78 位数据位，就可以得到帧号、小区频点等数据。

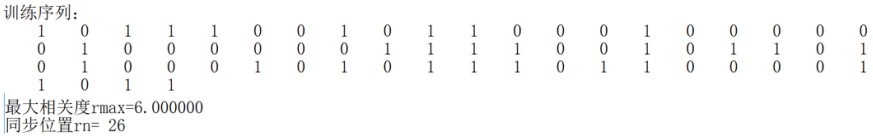

图 3-19 位同步程序测试

3.4.3 信道解码模块

3.4.3.1 解交织

在移动通信传输过程中，为了将连续出现的错误转换为随机突发的错误，以便于检错和纠错，我们往往要进行交织。GSM-R 系统中，控制信道和业务信道的交织方式不同，所以他们的交织深度也有所不同，分别是 4 和 8。解交织即按照交织的规则，逆向处理将交织块重新恢复成连续的比特流。

我们以一个交织深度为 8 的业务信道消息为例，分析解交织的过程。

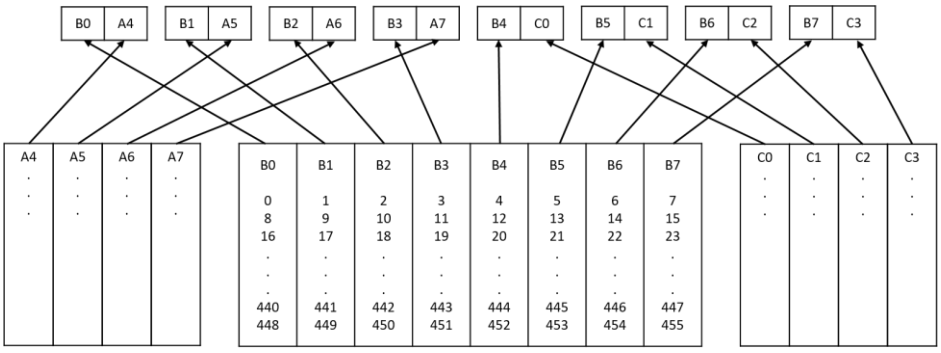

图 3-20 全速率话音业务交织图

图 3-20 为业务信道的交织过程图，分别进行了块内和块间交织，这样一个长为 20ms 的消息被分成 8 份，又被分配到 8 个不同的 NB 中。

本文用函数 deinterleaving 来实现解交织功能，其参数设置如表 3-7 所示。

表 3-7 函数 deinterleaving 参数

参数	参数类型	含义
*in	int	输入 456bit
*out	int	输出 8*57bit
*K	int	由 k 计算出块号、块长度等

交织与解交织互为逆过程，程序运行结果如图 3-21 所示，可以看出交织前后的码字关系是

$$
\begin{aligned}
&i(B,j)=c(n,k)\\
&k=0,1,...,455\\
&n=0,1,...,N,N+1,...\\
&B=B_0+4n+(k \bmod 8)\\
&j=2((49k) \bmod 57)+((k \bmod 8)/4)
\end{aligned}
$$

其中，n 为交织前消息块编号（BN），k 为 BN 中比特编号，B 为交织后突发（NB）编号，j 为 NB 中比特编号。

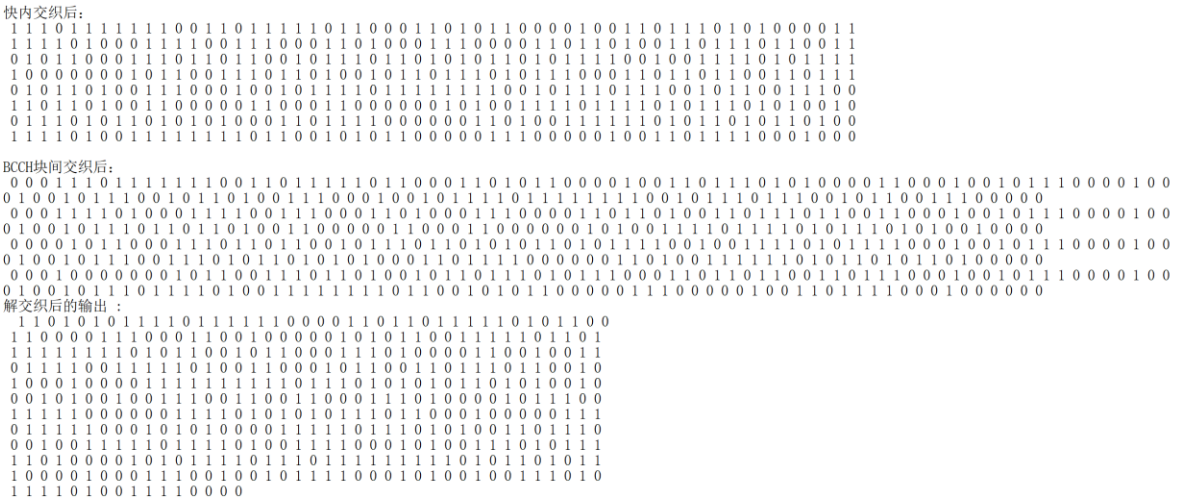

图 3-21 解交织程序运行图

3.4.3.2 信道解码

信道解码是对信道编码的一个逆过程，信道编码时增加了一些冗余比特达到差错控制的目的，在接收端解码时就要通过这些冗余比特进行纠错和检错，改善传输质量。其中，卷积编码的过程和原理已经在上文中解释过了，此处不再赘述，GSM-R 系统采用的是码率为 1/2，采用（2,1,4）卷积码。

对于卷积码编码，我们采用 Viterbi 算法来做解码。（2,1,4）卷积码共有 16 种状态，每状态都有两条输入路径。基于最大似然原则，在网格图中每个节点的两条路径中选择发送与接收码字距离最小的路径，最大化判决的正确率。

在进行编程实现 Viterbi 解码的时候，要用到四个函数，如表 3-8 所示。

表 3- 8 Viterbi 解码函数

函数名	参数	作用
trandistance	m, stat1, stat2	符号表示从 state1 到 state2 时输出符号的汉明距离
traninput	a, b	状态从 a 到 b 时输入卷积码的符号
tranoutput	a, b	状态从 a 到 b 时卷积码输出的符号
viterbi	initialstate, *viterbiinput, *viterbioutput	Initialstate: 定义解码器初始状态 Viterbiinput: 解码器输入码字序列 Viterbioutput: 解码器输出码字序列

在 Viterbi 函数中，定义网格图中每一点为一个结构体 sta，每个点包括三个元素，如表 3-9 所示。

表 3-9 结构体内元素

元素名	数据类型	含义
met	int	转移到此状态累计的度量值
value	int	输入符号
*last	struct sta	指向前一个状态的指针

具体的 Viterbi 解码步骤如下：

（1）定义网格图和指针。

struct sta state[16][N];

struct sta *g;

（2）初始化每个状态的度量值，包括网格图中每个点和指针。

（3）利用汉明距离找出经过 n 步后度量值最小的状态，准备回溯路由，这部分代码如图 3-22 所示。

```
for (t = 1; t<N; t++)
{   for (p = 0; p<8; p++)
        for (q = 0; q<16; q++)
        {   if ((trandistance(viterbiinput[t], s[p], s[q]) == 10000) || (trandistance(viterbiinput[t], s[p + 8], s[q]) == 10000))
            {       continue;       }
            else
            {   int met1 = state[p][t - 1].met + trandistance(viterbiinput[t], s[p], s[q]);
                int met2 = state[p + 8][t - 1].met + trandistance(viterbiinput[t], s[p + 8], s[q]);
                if (met1>met2)
                {   state[q][t].met = met2;
                    state[q][t].value = traninput(s[p + 8], s[q]);
                    state[q][t].last = &state[p + 8][t - 1];
                }
                else
                {   state[q][t].met = met1;
                    state[q][t].value = traninput(s[p], s[q]);
                    state[q][t].last = &state[p][t - 1];
                }
            }
        }
}

r = state[0][N - 1].met;                /*找出n步后度量值最小的状态，准备回溯路由*/
g = &state[0][N - 1];
```

图 3-22 寻找最小度量值代码

（4）向前递归找出最大似然路径。

无差错传输得到的 Viterbi 解码结果如图 3-23 所示。

```
无错误出现时对编码输出viterbi译码得到的序列:
 1 0 1 1 1 0 1 0 0 1 0 1 1 0 0 1 0 0 0 1
 0 0 1 0 1 0 0 1 0 0 1 0 0 0 1 1 1 0 1 0
 0 0 0 1 0 1 0 1 1 0 1 0 1 0 1 1 0 0 0 0
 0 1 1 0 0 1 1 0 0 0 1 0 0 1 0 1 0 0 1 0
 1 1 0 1 0 0 0 0 1 1 0 1 1 0 0 1 1 0 1 1
 1 0 1 0 1 1 0 1 1 1 0 0 1 0 0 1 1 0 0 0
 0 1 1 0 1 0 0 1 0 1 0 0 0 0 0 0 0 0 0 1
 0 1 0 1 1 1 0 0 0 0 1 1 1 0 1 1 1 0 1 1
 0 1 0 0 0 0 1 1 1 1 1 1 1 1 1 1 0 1 0 0 0
 0 0 0 0 1 1 1 1 0 1 0 0 0 0 1 0 1 0 1 1
 0 1 0 1 1 0 1 1 1 1 1 1 1 1 1 0 0 1 0 0
 0 1 0 0
经viterbi解码后得到的信息序列:
 1 0 1 1 1 0 1 0 0 1 0 1 1 0 0 1 0 0 0 1 0 0 1 0 1 0 0 1 0 0 1 0 0 0 1 1 1 0 1 0 0 0 0 1 0 1 0 1 1 0 1 0 1 0 1 1 0 0 0 0 0 1 1 0 0 1 1 0 0 0 1 0 0 1 0 1 0 0
 1 0 1 1 0 1 0 0 0 0 1 1 0 1 1 0 0 1 1 0 1 1 1 0 1 0 1 1 0 1 1 1 0 0 1 0 0 1 1 0 0 0 0 1 1 0 1 0 0 1 0 1 0 0 0 0 0 0 0 0 0 1 0 1 0 1 1 1 0 0 0 0 1 1 1 0 1 1 1
 0 1 1 0 1 0 0 0 0 1 1 1 1 1 1 1 1 1 0 1 0 0 0 0 0 0 0 1 1 1 1 0 1 0 0 0 0 1 0 1 0 1 1 0 1 0 1 1 0 1 1 1 1 1 1 1 1 1 1 0 0 1 0 0 0 1 0 0
出错的个数
1
出错所在的位数
25
译错的个数:0
```

图 3-23 Viterbi 解码无差错传输运行结果

例如，如果输入的信息比特是 1001，经过编码器输出信息比特是 11101111，经过信道干扰后接收到的信息比特是 11001111，根据（2，1，4）卷积译码的状态转移图，从T_0到T_4只能按箭头所示的方向进行状态变换，当卷积码的两位中有一位错误的时候，比如T_1到T_2，正确的编码是 10，传输有误得到的是 00，按照图 3-24 实线的走法，得到汉明距离最小为 1，其他方法汉明距离都大于 1，于是得到正确译码，可见 Viterbi 译码有一定的纠错能力。

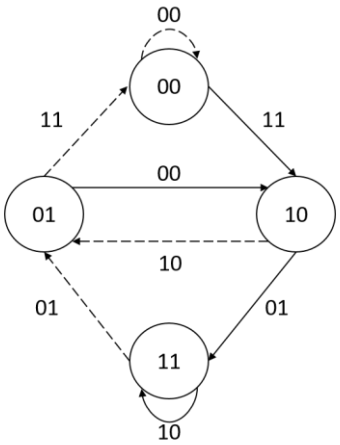

图 3-23 卷积编码状态转移图

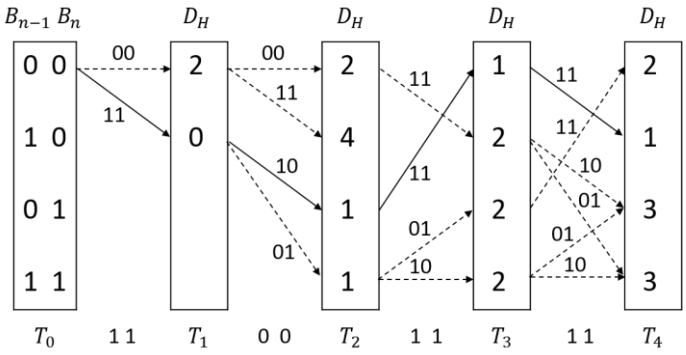

图 3-24 Viterbi 译码的过程

3.5 本章小结

本章主要介绍了软件的代码实现。虽然处理数据的时候只需要用到接收端部分，但是为了更好地理解和熟悉系统，本次设计还是编写了发送端和调制的部分。在发送端，本文完成了卷积编码、交织、映射和 GMSK 调制；在接收端，解调主要用到的是 1 比特差分解调方法；在进行同步之前，先对信号做 FB 搜索进行初定位，然后用解调得到的序列做精确定位和同步解析；然后提取每帧时隙 0 的控制信息，做反映射，解交织，信道解码，最后得到 23 字节的数据帧，交付给 L2 做进一步解析。

第 4 章 数据实测及分析

4.1 数据读取

数据文件由多帧连续的 IQ 数据合成，无文件头。文件格式如表 4-1 所示：

表 4-1 数据帧组成格式

IQ 数据帧 1
IQ 数据帧 2
...
IQ 数据帧 n

每帧 IQ 数据格式如表 4-2 所示：

表 4-2 IQ 数据格式

序号	描述	长度	备注
1	帧头	4 字节	固定为 0xAAAAAAAA
2	数据长度	2 字节无符号整数	不含帧头、帧尾及自身之外的总长度
3	通道号	2 字节无符号整数	采集数据的通道号
4	命令码	2 字节无符号整数	采集数据的命令码
5	采集的频率	4 字节无符号整数	单位为 Hz
6	频谱跨距	4 字节无符号整数	单位为 Hz
7	中频带宽	4 字节无符号整数	单位为 Hz
8	采样率	4 字节无符号整数	单位为 Hz
9	IQ 数据个数 n	2 字节无符号整数	
10	I 数据 1	2 字节有符号整数	
11	I 数据 2	2 字节有符号整数	
12	...	...	
13	I 数据 n	2 字节有符号整数	
14	Q 数据 1	2 字节有符号整数	
15	Q 数据 2	2 字节有符号整数	
16	...	...	
17	Q 数据 n	2 字节有符号整数	
18	帧尾	4 字节	固定为 0x55555555

4.2 解调数据测试分析

由于实测数据中，已经有了经过下变频得到的 I、Q 支路数据，GMSK 解调的核心就是根据 1 比特相位差来判决得到输出结果。根据欧拉公式和 MATLAB 中的 angle 函数来解析相位角，本文改进了第三章中提到的 1 比特差分解调方法，因为每 3 个采样点对应 2 个比特的信息，每采样点进行一次拓展，使每 3 个采样点对应一个比特，然后再抽样判决。

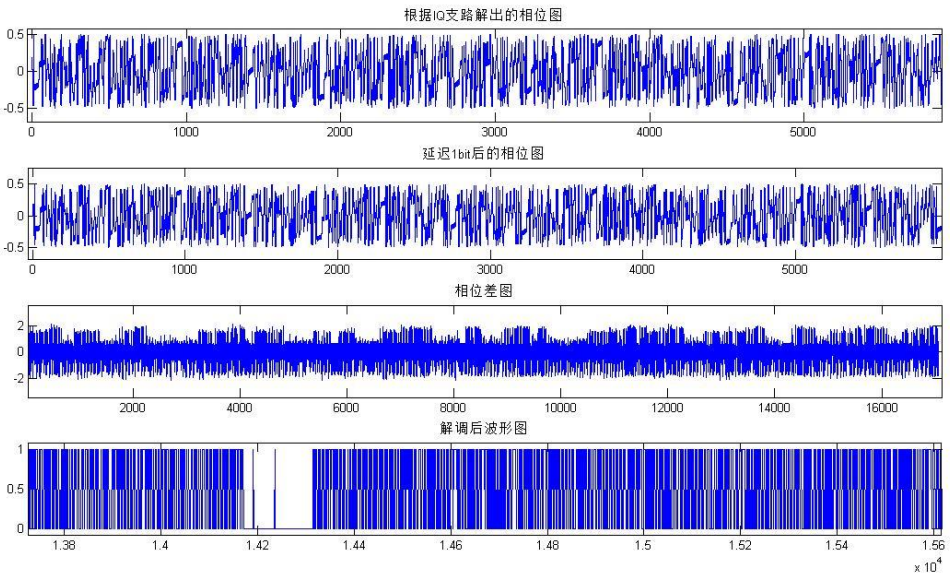

图 4-1 解调结果

如图 4-1 所示，在解调结果中得到了 146 个连续的 0，可以认为这是 FB 在解调之后的位置。

4.3 数据同步测试分析

4.3.1FB 搜索实现初同步

根据上文所述，FB 数据部分发送的是 142bit 的“0”。所以在时域上看到的将是一段频率为 67.7083KHZ 的正弦波。于是开窗对 FIR 输出的两倍采样的 I，Q 数据做 256 的点 FFT 变换。并求出频域 256 点的 I 的平方加上 Q 的平方之和记为能量。根据奈奎斯特采样定理，每 25000 个采样点求出能量最大值及对应的窗口值，可以看成是 FB 的大概起始点，完成初定位。

对于每一个窗口值，都有一个 256 点的频谱图，对每个频谱图取最大值，在所有采样点上作图，得到如图 4-2。

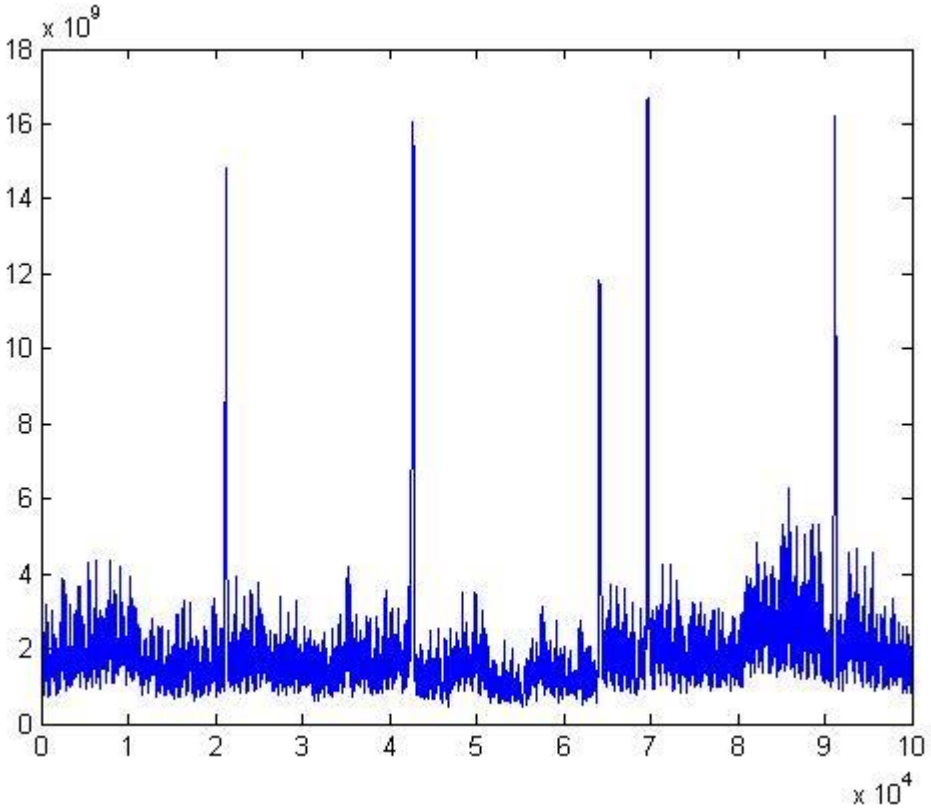

图 4-2 FB 搜索

可以从图 4-2 看到，每个峰值之间的间隔大概为 25000 个采样点，与理论分析相符合，通过求第一个峰值的序号，可以大概知道 FB 的起始位置，也就实现了初定位，然后要解调之后进行 SB 搜索，利用 SB 中训练序列自相关性很好的特点实现精确定位。解析到的序列位置为 21221，$21221\times\frac{2}{3}=14147$ 。

4.3.2SB 搜索实现精确定位

对于解调得到的数据，从 MATLAB 的工作区保存到 txt 文档，在 Visual Studio2017 里调用 synchronization 子程序，进行时间精确定位，结果如图 4-3 所示。

```
训练序列:
     1     0     1     1     1     0     0     1     0     1     1     0     0     0     1     0     0     0     0     0
     0     1     0     0     0     0     0     0     1     1     1     1     0     0     1     0     1     1     0     1
     0     1     0     0     0     1     0     1     0     1     1     1     0     1     1     0     0     0     0     1
     1     0     1     1

最大相关度rmax=6.000000
同步位置rn=14370
 请按任意键继续. . .
```

图 4-3 时间精确同步运行结果

同步比特开始的位置是 15720，结合前文的数据，解调得到的连续 0 比特从 14169 到 14315 共 146 比特，中间间隔 1405 比特。到此，可以认为完成了同步定位。

4.4 本章小结

本章主要测试分析了实测数据，用自己设计的软件来测试。其中，同步部分的数据和理论预期的基本相符合，虽然频谱中仍有旁瓣干扰，但不影响结果。SB 搜索定位到的位置与理论预期的也基本相同。

结 论

本文根据 3GPP 关于 GSM-R 系统空中接口（Um）标准协议，搭建了基于 Visual Studio 2017 和 MATLAB R2014a 平台的物理层分析软件的设计和实现。实现了 GSM-R 采样数据在物理层的处理，包括解调、同步、解交织和解码等功能，得到 184bit 的数据格式交付给 L2 进行进一步分析。本设计的主要内容有以下几点：

（1）本文研究了 GSM-R 系统空中接口物理层相关的协议，对协议中涉及的关键内容和技术做了较为详细的介绍，为软件设计打下了理论基础。

（2）本文基于 GSM-R Um 接口的三层协议，实现了软件的设计和实现，在软件中，物理层分为解调、同步、解交织和解码等模块，每个模块都进行了代码实现。

（3）本文中对于同步的分析方法采用了先进行频谱分析再解调的方法，相对于先解调再同步的方法，减小了噪声干扰及误码对结果的影响，从测试结果来看，这样的分析方法也是准确可靠的。

由于我本人这学期忙着准备面试和申请学校，本设计虽然实现了各个模块的功能，但没能将各模块组合起来实现一个简洁、完整的软件。以下是本设计需要改进的地方：

（1）用滑窗法寻找同步位置的方法虽然比较精确，但运算量确实比较大，出于 GSM-R 系统对低时延的高要求，算法需要继续优化.

（2）由于测试数据只涉及到控制信道的 BCCH、CCCH 消息，突发只涉及到 NB、FB 和 SB 三种类型，所以复杂度不大，需要进一步提高解析别的信道信息的能力。

致 谢

大学生活一眨眼间就要过完了，回首走过的岁月，心中倍感欣慰，在我即将写完这篇毕业论文的之际，谨以此向曾经指导帮助、支持和鼓励过我的老师和同学们表达最真诚的谢意。

首先，要感谢我的班导师刘林老师对我的毕业设计和论文的指导与帮助。尤其是我在选题以及开题答辩期间在国外交换学习。从毕业设计的开题到完成过程中的学习、研究，刘林老师都给了我悉心的指导和热情的帮忙，解答了我的各种疑惑，使我的毕业论文能够顺利的完成。刘林老师对工作的认真负责、对学术的钻研精神和严谨的学风，都是值得我终生学习的。

在我的大学本科四年的学习生活中，我能够取得现在的成绩还要感谢学院、学校的各位老师和领导以及所有职工的辛勤付出。正是由于他们的认真工作，尽职尽责，才给我们创造了一个美好校园环境，让每一个学子能够茁壮成长，在这四年里能够度过一个美好的大学生活。

另外也要感谢通信 2014-02 班的每位同学。在这四年里，大家一起生活，一起学习，共同维持着这个集体，度过了一段美好的大学时光。感谢你们的鼓励和帮忙，特别是我的室友黄润、赵亦岩，四年来，我们朝夕相处，共同进步，感谢你们给予我的所有关心和帮忙，同窗之谊，我将终生难忘!

最后我要感谢的是我的父母，你们一直是我最坚强的后盾，对我的支持鼓励是我一辈子都无以回报的。

参考文献

[1] 邵翔. GSM-R 系统欧洲标准简介[J]. 铁路通信信号工程技术(RSCE), 2008, 5(3),9-11

[2] 李昌根. GSM-R 系统 Um 接口监测与智能分析系统研究[J]. 铁路信号技术交流, 2017.

[3] 吴宁，韩蕾．基于 GSM-R 网络空中接口动态监测系统研究[J]．铁路通信信号工程技术，2014, 11(2)：26-31．

[4] 裘凯迪. 多通道 GSM-R Um 接口监测仪的研究与实现[D]. 硕士论文. 北京交通大学, 2016

[5] 向长波，李惠，王玉霞等. 基于 Matlab／Simulink 的 GSM 系统物理层仿真[J]. 中国新通信，2015，17(10)：104—105．

[6] 刘继武．GSM 空中信号截取与移动台定位技术研究[D]．硕士论文．东南大学，2004．

[7] Proakis J G. Adaptive equalization for TDMA digital mobile radio[J]．Vehicular Technology, IEEE Transactions on，1991，40(2)：333—341．

[8] Fomey Jr G D．Maximum-likelihood sequence estimation of digital sequences in the presence of intersymbol interference[J]．Information Theory, IEEE Transactions on，1972，1 8(3): 363-378．

[9] 郑沐. GSM 空口协议分析软件的实现及应用[D]. 硕士论文. 电子科技大学, 2016

[10] 孙杨杰. GSM 空口协议分析软件设计及实现[D]. 硕士论文. 电子科技大学, 2016

[11] 钟章队. 铁路 GSM-R 数字移动通信系统[M]. 中国铁道出版社, 2007

[12] 3GPP. TS 04.03 V8.0.2 Channel structures and access capabilities[S]. 3GPP, 2002

[13] 3GPP. TS 04.05 V8.0.2 General aspects[S]. 3GPP, 2002

[14] 3GPP TS 04.06, v5.3.0, Data Link(DL) layer specification[S].

[15] 3GPP. TS 05.02 v8.11.0 Multiplexing and multiple access on the radio path[S]. 3GPP, 2003

[16] 3GPP. TS 05.03 V8.9.0 Channel coding[S]. 3GPP, 2005

[17] 3GPP. TS 05.05 V8.20.0 Radio transmission and reception[S]. 3GPP, 2005

[18] 3GPP. TS 05.08 V8.23.0 Radio subsystem link control[S]. 3GPP, 2005

[19] 3GPP TS 05.10 V8.12.0 Radio subsystem synchronization[S]. 3GPP, 2003

[20] 3GPP TS 08.58 V8.6.0 Layer 3 specification[S]. 3GPP, 2000

附录 1 部分程序源码

卷积与 Viterbi 解码程序代码

```
//（2,1,4）卷积和维特比算法解卷积
#include <stdio.h>
#include "Conio.h"
#define N 228     //总编码长度
#include "math.h"
#include <stdlib.h>
#include<time.h>
#define randomize() srand((unsigned)time(NULL))
int s[16]={0,8,4,12,2,10,6,14,1,9,5,13,3,11,7,15};//四个寄存器，16 个状态

//卷积码编码
encode(
unsigned int *symbols,       /*编码输出*/
unsigned int *data,          /*编码输入*/
unsigned int nbytes,         /*nbytes=n/16,n 为实际输入码字的数目  */
unsigned int startstate      /*定义初始化状态*/
)
{
    int j;
    unsigned int input,a0=0,a1=0,a2=0,a3=0;
    for(j=0;j<nbytes;j++)
    {
        input=*data;
        data++;
        *symbols = input^a2^a3;//^为异或运算符（生成多项式为[1 0 0 1 1]和[1 1 0 1 1]）
        symbols++;
        *symbols = input^a0^a2^a3;
        symbols++;
        a3=a2;
        a2=a1;
```

```
            a1=a0;
            a0=input;
        }
        return 0;
}

/*********viterbi 译码相关*****************/
 int trandistance(int m, int stat1,int stat2)              /*符号 m 与从 state1 到 state2 时输出符号的汉明距离,
                                                                    如果 state1 无法到 state2 则输出度量值为 10000*/
 {
        int c;
        int sym,sym1,sym2;
        sym1=((stat2>>3)&1)^((stat1>>1)&1)^((stat1)&1);//1 0 0 1 1
        sym2=((stat2>>3)&1)^((stat1>>3)&1)^((stat1>>1)&1)^(stat1&1);//1 1 0 1 1
        sym=(sym1<<1) | sym2;

if((((stat1>>3)&1)==((stat2>>2)&1))&&(((stat1>>2)&1)==((stat2>>1)&1))&&( ( (stat1>>1)&1)==(stat2&1)))
            c=((m&1)^(sym&1))+(((m>> 1)&1)^((sym >> 1)&1));//计算欧式距离
        else
            c=10000;
        return(c);
}


int traninput(int a,int b)              /*状态从 a 到 b 时输入卷积码的符号*/
{
        int c;
        if
((((a>>3)&1)==((b>>2)&1))&&(((a>>2)&1)==((b>>1)&1))&&(((a>>1)&1)==(b&1)))
            c=((b>>3)&1);
        else
```

```
        c=-1;
    return(c);
}

int tranoutput(int a,int b)              /*状态从 a 到 b 时卷积码输出的符号*/
{
    int c,s1,s2;
    s1=((b>>3)&1)^((a>>1)&1)^(a&1);//1 0 0 1 1
    s2=((b>>3)&1)^((a>>3)&1)^((a>>1)&1)^(a&1);//1 1 0 1 1
    if((((a>>3)&1)==((b>>2)&1))&&(((a>>2)&1)==((b>>1)&1))&&(((a>>1)&1)==(b&1)
))
        c=(s1<<1)|s2;
    else
        c=-1;
    return(c);
}

void viterbi(int initialstate,                    /*定义解码器初始状态*/
                int *viterbiinput,                /*解码器输入码字序列*/
                int *viterbioutput)               /*解码器输出码字序列*/
{
    struct sta                                    /*定义网格图中每一点为一个
结构体,其元素包括*/
    {
        int met;                                  /*转移到此状态累计的度量值
*/
        int value;                                /*输入符号 */
        struct sta *last;                         /*及指向前一个状态的指针*/
    };
    struct   sta state[16][N];
    struct   sta *g;
    int i,j,p,q,t,r,u;//,l;

    for(i=0;i<16;i++)                             /* 初始化每个状态的度量值*/
```

```
    {
        for(j=0;j<N;j++)
            state[i][j].met=0;
    }

    for(int m=0;m<16;m++)
    {
        state[m][0].met=trandistance(*viterbiinput,initialstate,s[m]);
        state[m][0].value=traninput(initialstate,s[m]);
        state[m][0].last=NULL;
    }

    for(t=1;t<N;t++)
    {
        for(p=0;p<8;p++)
            for(q=0;q<16;q++)
            {

    if((trandistance(viterbiinput[t],s[p],s[q])==10000)||(trandistance(viterbiinput[t],s[p+8],s
[q])==10000))
                    {
                    //    state[q][t].last=NULL;0

                        continue;
                    }
                    else
                    {
                        int met1=state[p][t-1].met+trandistance(viterbiinput[t],s[p],s[q]);
                        int met2=state[p+8][t-1].met+trandistance(viterbiinput[t],s[p+8],s[q]);
                        if(met1>met2)
                        {
                            state[q][t].met=met2;
                            state[q][t].value=traninput(s[p+8],s[q]);
                            state[q][t].last=&state[p+8][t-1];
```

```
                    }
                    else
                    {
                        state[q][t].met=met1;
                        state[q][t].value=traninput(s[p],s[q]);
                        state[q][t].last=&state[p][t-1];
                    }
                }

            }
    }

    r=state[0][N-1].met;                    /*找出 n 步后度量值最小的状态，准备回溯路由*/
    g=&state[0][N-1];


  for(u=N;u>0;u--)                          /*向前递归的找出最大似然路径 */
    {viterbioutput[u-1]=g->value;
     g=g->last;
    }

    /* for(u=0;u<8;u++)
       *(viterbioutput+u)=state[u][2].met; */      /*此行程序可用于检测第 n 列的度量值*/
}

void decode(unsigned int *input, int *output,int n)
{
    int viterbiinput[500];
    int j;
    for(j=0;j<n+4;j++)
    {
        viterbiinput[j]=(input[j*2]<<1)|input[j*2+1];
```

```
        //printf("%3d",viterbiinput[j]);
    }
    viterbi(s[0],viterbiinput,output);
}

void main()
{
    int decodeoutput[250],n=N-4,i,m,j=0,jj=0;
    unsigned                                                                    int
encodeinput[250]/*={0,0,1,1,0,1,1,0,0,0,0,0,1,0,1,1,0,0,1,0,0,0,1,0,0,0,0,1,0,0,0,0,0,0}*/,wr
ong[50]={0},encodeoutput[500];

    randomize();
    for(i=0; i<n; i++)
        encodeinput[i]=rand()%2;
    encodeinput[n]=0;
    encodeinput[n+1]=0;
    encodeinput[n+2]=0;
    encodeinput[n+3]=0;

    encode(encodeoutput,encodeinput,n+4,s[0]);
    printf("信息序列 :\n");
     for(i=0;i<n+4; i++)
        printf("%2d",encodeinput[i]);

    printf("\n");
    printf("对信息序列编码后的输出 :\n");
    for(i=0;i<(n+4)*2;i++)
    {
        printf("%2d",encodeoutput[i]);
        if(i%20==19)
            printf("\n");
    }
    printf("\n");
```

```
decode(encodeoutput,decodeoutput,N);
/*
for(i=0;i<N;i++)
      encoded[i]=encodeoutput[i];
decode(encoded,decoded,N/2);
for(i=0;i<N/2;i++)
      decodeoutput[i]=decoded[i];
for(i=0;i<N;i++)
      encoded[i]=encodeoutput[i+N];
      decode(encode,decoded,N/2);
for(i=0;i<N/2;i++)
      decodeoutput[i+N/2]=decoded[i];
*/
printf("无错误出现时对编码输出 viterbi 译码得到的序列:\n");
for(int ii=0;ii<n;ii++)
      printf("%2d",decodeoutput[ii]);
printf("\n");
for(i=0;i<n;i++)
{
      if(encodeinput[i]!=decodeoutput[i])
            jj++;
}
 printf("译错的个数:%d\n",jj);

printf("请输入出错的个数\n");
scanf("%d",&m);
printf("请输入每个出错所在的位数\n");
for(i=0;i<m;i++)
{
      scanf("%d",&wrong[m]);
      if(encodeoutput[wrong[m]]==0)
            encodeoutput[wrong[m]]=1;
     else
           encodeoutput[wrong[m]]=0;
```

```
    }
    printf("输入 viterbi 解码器的待解码序列(即有误的序列) :\n");
    for(i=0;i<(n+4)*2;i++)
    {
        printf("%2d",encodeoutput[i]);
        if(i%20==19)
            printf("\n");
    }
    printf("\n");


    decode(encodeoutput,decodeoutput,n+4);
    printf("经 viterbi 解码后得到的信息序列:\n");
    for(i=0;i<n;i++)
        printf("%2d",decodeoutput[i]);
    printf("\n");

    for(i=0;i<n;i++)
    {
        if(encodeinput[i]!=decodeoutput[i])
            j++;
    }
     printf("译错的个数:%d\n",j);
    system("pause");
}
```