摘　　要

目前Internet的发展趋势是具有更高的带宽和提供有差别业务的能力，传统的面向无连接的“尽力传送”模型已经无法满足这一需求。多协议标签交换(MPLS：Multi-Protocol Label Switching)作为下一代Internet骨干网的核心技术，为各种网络层协议和数据链路层协议提供了一种有效的多协议解决方案，能够满足日益增长的服务质量要求并提供流量工程支持。

传统的IP网络的核心器件——路由器已无法满足日益增长的网络需求，采用光电混合技术的Tbps多协议标签交换路由器正成为各研发机构的开发热点。本实验室开发研制了光电混合技术的MPLS路由器。其中具有硬件转发功能的接口模块是该路由器的关键技术，本文提出了一种高速分布式路由器接口模块的设计方法，详细讨论了接口模块的具体实现过程，完成了接口模块的逻辑和电路板的设计工作，并实验分析了在100M以太网中该接口模块的性能。

本文先简单地分析了传统IP网络的缺陷，介绍了IP网络的发展趋势、路由器的发展概况、多协议标签交换（MPLS）技术和所设计的高速MPLS路由器的具体组成。然后讨论了高速分布式路由器接口模块的总体设计方案，将接口模块划分为数据收发模块、数据处理模块、转发处理模块、接口控制模块和SAR（Segmentation And Reassemble）模块五部分。接下来对各个模块的具体设计过程与实验结果进行了详细的说明与分析。此接口模块采用FPGA实现数据的硬件转发，采用内容可寻址存储器(CAM：Content Addressable Memory)可实现千万次量级的路由/标签查找，从而可以实现单端口达到Gbps的数据接入速率。

关键词：多协议标签交换，内容可寻址存储器，Tbps路由器，光电混合，接口模块，转发

ABSTRACT

Higher bandwidth and the capability of providing different service are the current trend of development of Internet, but the traditional connectionless oriented "Best of Effect" module can't meet this expectation. As a key technology for the next generation Internet, MPLS (Multi-Protocol Label Switching) introduces an effective multi-protocol solution to both network layer protocol and data link layer protocol. It meets the increasing needs for Quality of Service (QoS) and traffic engineering.

Traditional router as the key equipment of IP networks can not meet the fast increasing need of the network. The research of terabits/s hybrid optical-electronic router has become a hotspot in the data communication field. And then, a terabits/s hybrid optical-electronic router was made in our lab. The interface module with hardware forwarding engine is the key component of the router. In this paper, a design method of the interface in the high speed distributed router is provided, and also with the logic and PCB design in detail. The interface module has been tested in the 100M Ethernet successfully.

In this paper, the limitation of the traditional IP network is introduced firstly. And then, the current trend of IP network development, the general development of the router, the MPLS architecture and the components of the router made in our lab are described consequently. The design scheme of the interface module is also provided in this paper. In the design scheme, the interface module is divided into five parts: data receiving and sending module, data processing module, forwarding processing module, interface control module and SAR module. At last the detailed design procedure of each part for the interface module and the analysis of the test data to the

engine. Embedded CAM (Content Addressable Memory) was used in the interface module to achieve route/label lookup rate of the order of magnitude of 10Mpps. Therefore the interface module can access Gbps data in a single port.

Keywords: Multi-Protocol Label Switching, Content Addressable Memory, Tbps router, hybrid optical-electronic, Interface module, Forward

1 绪 论

1.1 发展第三层交换技术的必要性

目前 Internet 的发展趋势是：用户数正以 165%的年增长率在全球扩展；Web 设置每 57 天增加一倍；带宽每 5 个月要求加倍；每 30 分钟增加新的网络连接。Internet 上的各种应用日益广泛，从数据、语音到视频，人们对信息内容有了更为广泛的需要，通信数据化、实时化的趋势越来越明显，这导致了对网络性能的要求的提高，在这种背景下传统的 Internet 已经在很多方面暴露出其先天的不足，集中体现在以下几个方面：

1） 带宽容量不足

在传统 IP 网络中，有个著名的 80/20 规则[1]。所谓的 80/20 规则，即 80%的网络流量发生在域内部，20%的网络流量跨域进行。但是现在网络环境已经发生巨变，具体表现在 Web 应用呈爆炸性增长，网络流量的分布模式已变得无法预测，用户数量呈指数曲线增长等等。现在，80/20 的规则已转换为 20/80。传统路由器缓慢的查找与转发速度已无法满足现在网络业务的高速增长，成为了 IP 网络带宽增长的瓶颈。

2） 缺乏服务质量 QoS(Quality of Service)保证。

在网络业务日益增多的情况下，对网络资源的要求也必然提高，对可靠性、灵活性等特性也提出了更高的要求。为了保证对各种业务的满足，引入 QoS 的概念。QoS[2]是指多媒体用户要求网络传输系统所必须保证的关于传输多媒体信息的质和量的特征集，定义一系列服务参数，如信元丢失率、平均信元延迟、延迟抖动、平均速率、峰值速率等等，对于不同的业务流相应作出规定。而传统的 IP 网络属于 IP 协议的应用初期，因此更多的是考虑利用 IP 协议来达到传输多种业务的目的，采用尽力传送[3](best effort)方式，未能就融合传输的质量等方面进行全方位的考虑。

3） 流量工程[4](TE:Traffic Engineering)能力弱。

网络拥塞可能会是因为网络资源不足或者是流量分布不均匀造成的。在前一种情况下，所有路由器和链路都会过载，唯一的解决办法是升级基础设施，提供更多的网络资源。在第二种情况下，网络中的部分地方过载而其它地方的负载却较轻。

流量工程就是安排数据流如何通过 IP 网络，以避免不均匀地使用网络而导致的拥塞的过程。流量工程提供可预测的网络流量控制。可以实现在普通的应用运行期间，网络可获得最大的容量，确保在故障或流量重新路由期间可以优化利用所有可用的网络资源，使网络可以检测到故障或拥塞

目前的路由协议从本质上讲是无连接的，路由选择只是基于目的地 IP 地址和最短路径进行的，忽略了网路可用链路容量和分组流本身的要求。这是传统 IP 技术的局限性决定的。

为了克服 IP 网络的上述缺陷，需要构建一个更完善的通信网络来满足未来所需的服务.目前主要提出三种新的传送技术：IP over ATM[5]，IP over SDH[6]，IP over WDM[7]。

1.2 第三层交换技术的比较

IP over ATM,融合了 IP 和 ATM 技术特点，发挥 ATM 支持多业务、提供 QoS(服务质量保证)的技术优势。IP over SDH，直接在 SDH 上传送 IP 业务，对 IP 业务提供了完善支持，提高了效率。而 IP over WDM，采用高速路由交换机设备和 DWDM (密集波分复用)技术，极大地提高了网络带宽，对不同码率、数据帧格式的业务提供全面支持。

1.2.1 IP over ATM

IP over ATM 的基本原理和工作方式为：将 IP 数据包在 ATM 层全部封装为 ATM 信元，以 ATM 信元的形式在信道中传输。当网络中的交换机接收到一个 IP 数据包时，它首先根据 IP 数据包的 IP 地址通过某种机制进行路由地址处理，按路由转发。随后，按已计算的路由在 ATM 网上建立一条虚电路（VC：Virtual Circuit)。以后的 IP 数据包将在此虚电路 VC 上以直通（Cut－Through）方式传输而不再经过路由器，从而有效地解决了 IP 路由器的瓶颈问题，并将 IP 包的转发速度提高到交换速度。

ITU－TSG－13 组把 IP over ATM 的解决方案分为两种模型：重叠模型和集成模型。

重叠模型的实现方式主要有 IETF（Internet Engineering Task Force）推荐的 IPOA（IP Over ATM）、CIPOA（Classic IP Over ATM）以及 ATM 论坛推荐的 LAN(Local Area Network)仿真（LANE－LAN Emulation）和 MPOA（Muti-Protocol over ATM）等。

集成模型的实现技术主要有 Ipsilon 公司提出的 IP 交换（IP Switching）技术、Toshiba 公司的信元交换路由器（Cell Switched Router），Cisco 公司提出的标记交换（Tag Switching）技术和 IETF 推荐的 MPLS 技术。

IP over ATM 具有以下优点：（1）利用 ATM 技术本身能提供 QoS 保证的特点，可提高 IP 业务的服务质量。（2）具有良好的流量控制均衡能力以及故障恢复能力，网络可靠性高。（3）适应于多业务，具有良好的网络可扩展能力。（4）对其它多种网络协议能提供支持。

但是它也存在 以下缺点：（1）IP 数据包分割加入大量头信息，造成很大的带宽浪费（20%～30%）。(2)由于 ATM 本身技术复杂，导致管理复杂。

1.2.2 IP over SDH

IP over SDH，也称 Packet over SDH（PoS），它以 SDH 网络作为 IP 数据网络的物理传输网络。它使用链路及 PPP 协议对 IP 数据包进行封装，把 IP 分组根据 RFC 1662 规范简单地插入到 PPP 帧中的信息段。然后再由 SDH通道层的业务适配器把封装后的IP数据包映射到SDH的同步净荷中，然后向下，经过 SDH 传输层和段层，加上相应的开销，把净荷装入一个 SDH 帧中，最后到达光层，在光纤中传输。

IP over SDH 具有以下优点：（1）对 IP 路由的支持能力强，具有很高的 IP 传输效率。（2）符合 Internet 业务的特点，如有利于实施多路广播方式。（3）能利用 SDH 技术本身的环路。（4）省略了不必要的 ATM 层，简化了网络结构，降低了运行费用。

但是它也存在一些缺点，即（1）仅对 IP 业务提供好的支持，不适于多业务平台。（2）不能像 IP over ATM 技术那样提供较好的服务质量保障

（QoS）。

1.2.3 IP over WDM

IP over WDM，也称光因特网。其基本原理和工作方式是：在发送端，将不同波长的光信号组合（复用）送入一根光纤中传输，在接收端，又将组合光信号分开（解复用）并送入不同终端。

IP over WDM 是一个真正的链路层数据网。在其中，高性能路由器通过光 ATM 或 WDM 耦合器直接连至 WDM 光纤，由它控制波长接入、交换、选路和保护。

IP over WDM 的帧结构有两种形式：SDH 帧格式和千兆以太网帧格式。

IP over WDM 具有以下优点：（1）充分利用光纤的带宽资源，极大地提高了带宽和相对的传输速率。（2）对传输码率、数据格式及调制方式透明。可以传送不同码率的 ATM、SDH / SONET（Synchronous Optical NETwork）和千兆以太网格式的业务。（3）不仅可以与现有通信网络兼容，还可以支持未来的宽带业务网及网络升级，并具有可推广性、高度生存性等特点。

但是它也存在一些缺点：（1）目前，对于波长标准化还没有实现。（2）WDM 系统的网络管理应与其传输的信号的网管分离。但在光域上加上开销和光信号的处理技术还不完善，从而导致 WDM 系统的网络管理还不成熟。（3）目前，WDM 系统的网络拓扑结构只是基于点对点的方式，还没有形成“光网”。（4）当前 DWDM 的设计是用于长途传输的，仅提供终端复用功能，上下复用还不能动态进行。

通过以上的分析比较，我们可以发现，在高性能、宽带的 IP 业务方面，IP over SDH 技术由于去掉了 ATM 设备，投资少、见效快而且线路利用率高。对于 IP over WDM 技术，它能够极大地拓展现有的网络带宽，最大限度地提高线路利用率，并且在外围网络以千兆以太网成为主流的情况下，这种技术能真正地实现无缝接入。但在目前，还有些关键技术还没有突破，只能说，IP over WDM 将代表着宽带 IP 主干网的明天。而 IP over ATM 技术则充分利用已经存在的 ATM 网络和技术，发挥 ATM 网络的技术优势，适合于提供高性能的综合通信服务，因为它能够避免不必要的重复投资，

提供 Voice、Video、Data 多项业务，是传统电信服务商目前最好的选择。

由于 ATM 技术已经比较成熟，只需对上层软件作部分修改即可支持第三层交换，成本小，风险低，因此，国外很多公司都推出了基于 ATM 的第三层交换技术，如 Toshiba 公司的信元交换路由器 CSR（Cell Switching Router）、Ipsilon 公司的 IP 交换机、Cisco 公司的 Tag 交换机、IBM 公司的基于累积路由的 IP 交换技术等。

1.3 第三层交换技术的研究进展

近年来出现了许多用于提高传统 Internet 速率，使其适用于宽带多媒体通信应用的交换技术。如 TOSHIBA 公司的信元交换路由器(Cell Switching Router: CSR)、Ipsilon 公司的 IP 交换机、CISCO 的 Tag 交换机、IBM 公司的基于累积路由的 IP 交换技术(Aggregate Route based IP Switching)。IETF 的 MPLS(Multiple Protocol Label Switching)则是正在制定的定位于大型网络的 IP 交换标准。在许多方面这些 IP 交换技术综合了 ATM 和 IP 的最好方面，即 ATM 的快速、简单交换；IP 的普遍性、可扩展性和灵活性。值得注意的是所有这些 IP 交换机均可以通过增加适合的软件，使 ATM 交换的硬件设备用作快速路由器。IP 交换目前极具吸引力，但已有了更广泛的涵义[8-19]。

1）IP Switching。Ipsilon Network 公司的 IP Switching 是一种高速路由器。它将转发功能映射到硬件交换机，如 ATM 交换机。从逻辑上可以看作是一个附有第三层转发功能的第二层交换设备，与第三层的数据转发模块高速互连。IP 交换机由 ATM 交换机和 IP 交换控制器组成。

IP Switching 采用低层流交换。在 IP Switching 中，所有的流被分为两类，一类是持续时间长、业务量大的数据流，在 ATM 交换机硬件中直接进行交换，快速、低延时；另一类是持续时间短、业务量小、是突发分布的数据流类型，通过 IP 交换控制器中的路由软件进行 hop-by-hop 转发。流分类过程动态选择流。流在交换前，必须标记。一个流只有在上行、下行链路都标记过后，才能直接通过 ATM 交换机进行交换。

2）Tag Switching。Tag Switching 由转发部分和控制部分组成的，两者互相独立。转发机制是一种简单的标记交换机制，通过使用定长的标记

来作出决定，并对标记重写。控制机制通过一组模块来维持保留正确的标记传播信息，以第三层协议为基础，每个模块具有一定的控制功能，它解决了 IP 与 B-ISDN 之间不一致的问题。

Tag Switching 系统中处于边缘的路由器将每个输入帧的第三层地址映射为简单的标记(Tag)，然后把帧转化为打了标记的 ATM 信元；打了标记的信元被映射到 VC(Virtual Channel)上，在网络核心，由支持 Tag Switching 的 ATM 交换机进行标记交换。目的地边缘路由器去掉信元中的标记，把信元转换为帧并将其送往接收者。

3）多标记交换(MPLS)。上述几种 IP 交换技术虽然存在不少不同之处，但是它们的出发点和目的是相同的。IETF 结合这些 IP 交换技术的特点，主要以 Tag Switching 为基础成立了 MPLS 标准化工作组来将网络层路由标记交换算法技术标准化。

MPLS 采用标记的包转发技术来实现简单、高性能的包转发机制。它通过用标记转发代替标准的基于目的端的 hop by hop 转发，从而简化了包转发机制，这种标记交换是第三层交换，却具有第二层的速度。

1997 年由美国千兆位以太网的后起之秀 Foundry 公司率先推出了第三层交换机，随后各大网络厂商如 Bay、3Com、Cisco、HP 等纷纷推出自己的第三层交换机，掀起了一股第三层交换的浪潮，不少已获得了实际应用．国内也已经开始采用．

1.4 光互连在第三层交换技术的应用

各种通信技术的发展为信息的传输提供了方便，从而促进了信息的进一步生成，导致社会信息量的爆炸式增加和对信息传输的爆炸性需求。目前实现计算机在局部区域内互连的局域网的数据速率大多已达到 100Mbps(Million Bits Per-Second)．如果 A 和 B 之间若同时有 10 个 LAN 之间需要交换数据，那么，总数据速率将达 1Gbps。如果三网合一，上述情况下的总数据率将高达 1.94Tbps。随着 Internet 网用户每六个月翻一番的爆炸式发展，千兆以太网技术的引入，不久的将来，人们将需要几十到几百 Gbps 甚至 Tbps 的信道传输容量和交换速率。

目前的电交换系统(基于电的交换背板)是由大规模集成电路和互连电

线组成的，由于电线不可避免存在 R.L.C 参数，使得传输带宽受到限制，当高速信号通过这种线路时，信号会严重失真畸变，线路间会有严重的串话，并有系统的时钟歪斜引起的严重误码和高功耗等缺点[20]。故宽带、高速、大容量的 MPLS 交换系统设备中的高速信息数据传输仍然采用电互连作为信息载体是十分困难的。因此近些年，开始研究自由空间微光学互连网，用光作为信息载体传输高速信息[21-29]。光波作为信息载体，具有极高的时空带宽积、高度并行性和无干扰性等特点，使得凭借于光波，高性能计算机系统中普遍存在的瓶颈效应能够得以解决，通信系统中大容量、高速率数据交换得以实现。

光互连从互连所采用的信道来看，可分为光纤互连、波导互连[30-31]和自由空间互连[27-29]等。光纤互连适用于电路板之间或计算机之间的连接[28][31]，与电互连相比，其优点是扇出量大、系统功耗低等，采用分离的光源和探测器。

波导互连可以提供高密度互连通道，适用于芯片内或芯片之间的互连，采用集成光源和探测器，由集成光路来连接。

相对于其他光互连方式而言，自由空间光互连具有更好的实际性能。自由空间光互连是利用自由空间作为光的传输媒体，不需要物理通道，以光速，无干扰地完成信息的高速并行式传送。自由空间光互连适用于不在同一平面内处理器之间的互连，如芯片之间或电路板之间的连接，可以使互连密度接近光的衍射极限，不存在信道对带宽的限制，易于实现重构互连。主要有使用全息光学元件、空间光调制器、透镜和反射镜的几种，是目前的一个研究热点。最近出现的几项实用化技术如：垂直腔表面发射激光器 VCSEL（Vertical Cavity Surface Emitting Laser）[32-36]，MCM[37]以及光电和光机构封装技术等使光互连技术在芯片级、多芯片组件级以及系统板级之间的连接集成得以实现。

随着相关技术的发展，光互连技术已经越来越成熟，为采用光互连实现高速、大容量的交换系统奠定了基础。

1.5 国外最新研究成果

当前欧美各国及东亚地区有关通信厂商及研究部门都在加紧进行

MPLS 的研究工作，并已推出部分具有 MPLS 功能的设备，有些厂家还宣称已设计出实现 MPLS 功能的专用芯片组，只待商机成熟便迅速推向市场。

当前推出的具有 MPLS 功能的交换机或路由器有：1）Lucent Ascend CBX500，提供 IP MPLS DS3 端口，当整机升级为全 MPLS 交换机时，每个交换模块可支持 20 多万条路由。此外还有 GX550、B-STDX9000 系列 ATM 交换机，可提供 MPLS 功能。2）韩国迅通公司 CellinX-6070 具有 MPLS 功能的 ATM 交换机（第三层以软件方式支持）。3）Cisco 公司 GSR12016 交换式路由器，容量 20G，本身具备 MPLS 功能，能直接与 MPLS 标签交换路由器互通（只支持 PVC：Permanent Virtual Circuit 方式）；BPX8650 ATM 交换机以专用接口方式支持 MPLS，目前 Cisco 称其现在出产的 ATM 交换机或高端路由器 70%支持 MPLS 功能。4）Newbridge 公司 670 路由交换式平台集 ATM。5）爱立信的新一代 AXD 301 ATM 交换机、AXI540 交换机等支持 MPLS 功能[38]。6）推出 MPLS 软件包的有 Cisco、Harris、future MPLS、Nortel、Data connect、Junipper 公司等。其中 Nortel 称可提供全面的 MPLS 解决方案。

对 MPLS 的大规模实验主要在美国和欧洲进行。Lucent 在全美 23 个大城市间的 NET2000 实验网上进行 MPLS 技术实验，以测试 MPLS 与其它 ATM 网、帧中继网等的互连互通、MPLS 对语音、实时视频传输的支持能力和 MPLS VPN（Virtual Private Network）性能等。欧洲国家 1999 年在其欧洲国家研究网（European National Research Networks）上实现 ATM、SDH、Gbps 级以太网、DWDM 等传输网络技术统一在 MPLS 下进行互连互通，并作了语音传输、视频服务和多媒体业务等实时业务传输和 VPN 增值服务的实验，实验名称为 MPLS TF-TANT（Task Force for Testing of Advanced Networking Technologies）。

1.6 本课题的研究目的和意义

最近十年，由于因特网的迅猛发展，改变了人们的生活方式，人们对网络也不断提出更高的要求，传统的网络体系结构的局限性日益明显。MPLS 技术在 IP 网络中引入了标签交换，一方面减少了路由器的转发时延，另一方面也提供了服务质量的功能。同时，由于 MPLS 将第三层路由与第

二层交换很好的结合起来，具有良好的扩展性。以 MPLS 路由器为基础构筑下一代因特网是非常有前途的。

对于网络而言，关键的瓶颈不在于光纤物理层数据的传输，例如，光纤通道的速率从 OC-3(155Mbps)一直上升到 OC-192(10Gbps)，而且通过密集波分复用技术(DWDM)，可进一步利用光纤潜在的带宽。网络的瓶颈主要是路由器的处理速度。在单端口速率达到 10Gbps 量级时，需要处理速度达到 Tbps 量级的路由器，相应单端口分组查找速率要达到每秒几十兆次查找(MLPS：Million Lookups Per-Second)速度，同时交换背板带宽容量也需要是 Tbps 量级。T 量级路由器不仅可以用于目前的网络构架(校园网、城域网和骨干网)满足各种带宽需求，而且在未来的全光网络中也会是重要的组成部分，因此，研究 Tbps 量级的路由器具有极大的现实意义。

同时，随着业务提供者和大型 ISP 不断升级骨干网，路由器的高速接口速率从最初的 OC-12 升级到 OC-48，最近又掀起了向 OC-192 升级的浪潮。具有高速 OC-192 光接口的高速路由器是网络向 IP over DWDM 结构发展的关键。对运营商来说，高的端口密度具有重要意义。因为大型骨干网的核心节点一般在大城市，中心局和入网点的空间资源不但有限而且成本很高，业务提供者必须考虑节约空间，而具有高密度端口的高速路由器是解决空间压力的一个重要举措。

在这样的背景下，高端路由器中接口模块的设计是个关键，也是一个难点，传统路由查找的速率最高只能达到几个 Mpps（Million Packets Per-Second）这样的量级，如果接口模块以这样的速率转发分组，要实现 Tbps 量级的路由器是不现实的。因此，对接口模块的设计，要求具有极高的分组转发速率。

本文提出了一种利用 ATM 交换硬件实现高速、可扩展的 MPLS 体系结构，接口模块采用内容可寻址存储器实现了千兆个分组每秒量级的高速转发，并且采用流水线设计尽量减少接口模块的处理开销。提出的设计方法对 MPLS 路由器和高端 IP 路由器的设计具有一定的参考价值。

通过本课题的研究，力求对 MPLS 技术的某些方面进行详细的剖析，为推动下一代 IP 网络的发展尽一点力。

2 基于光电混合 Tb/s 量级 MPLS 路由器

2.1 前言

在目前的因特网中，路由器的基本功能是把分组从信源转发到目的地。为了实现这一目标，每一个路由器都必须通过路由协议如路由信息协议 RIP(Routing Information Protocol)，开放最短路径优先协议 OSPF(Open Shortest Path First)，边界网关协议 BGP(Border Gateway Protocol)等[39-45]来获得关于网络的路由信息和状态。路由器通过路由信息来构建路由信息库 RIB(Routing Information Base)和转发信息库 FIB(Forward Information Base)。当一个分组进入路由器时，路由器提取 IP 分组头中的 IP 地址，然后按照最长匹配法则，在路由信息库中寻找与该分组地 IP 地址最为接近的表项索引，确定 IP 分组所属的 FEC（Forward Equivalence Class），再根据表项里记录的下一站地址转发 IP 分组。这种方法存在的问题是为了找到最长匹配，要对路由表进行多次访问，并且由于是不定长匹配，较难用硬件实现。这就限制了分组在路由器中的转发速度。

为了克服传统路由器的最长匹配法则的缺陷，MPLS 引入了一种称为标签的固定长度的标识符。当 IP 分组进入网络时，边缘路由器分析 IP 分组头，决定该分组所属的转发等价类 FEC（Forward Equivalent Class），然后分配一个标签给此分组。通常一个标签只能分配给一个 FEC。通过检查标签，就可以确定特定分组所属 FEC。在 MPLS 网络中，路由查找则转变为对定长的标签的查找，省去了对 IP 分组头的访问和 IP 地址的最长匹配的查找过程，方便用硬件实现，大大地提高了路由器的吞吐量。

另外，MPLS 在无连接的 IP 网络中引入了一种面向连接的机制。在一个 MPLS 网络中，对于每一个路由器或者一条通路都建立一条标签交换式通路 LSP(Label Switched Path)。边缘路由器分析 IP 分组头，决定使用哪一条 LSP，并且在转发分组到下一跳之前，按照标签的格式在分组中增加一个相应的标识符。一旦完成以上过程，则在由分组头部的标签标识的 LSP

的所有后续节点上，只需对分组进行与 ATM 交换机类似的简单的转发。MPLS 的这种面向连接的机制大大地提高了分组的转发速度。

本章从 MPLS 网络入手，首先介绍 MPLS 的工作原理和体系结构，然后提出了一种新的基于光电混合的大容量高速 MPLS 路由器体系结构，为后续章节的展开打下基础。

2.2 多协议标签交换(MPLS)网络体系结构

近几年的发展已清楚表明 IP 将是下一个世纪网络的主宰。因此，如何使 ATM 技术融入 IP,如何将路由和交换结合起来，如何解决 IP 无连接和 ATM 面向连接的矛盾，以支持规模日益增长的 Internet 和多媒体业务，成为目前研究的热点。众多厂商和学者提出了许多新方案、概念和名词，如 IP 交换、CSR、Tag 交换、ARIS（Aggregate Route based IP Switching）等[9-19]。这些新方案都有一个共同点，即在路由中引入交换，实现线速率的分组转发。然而由于各个研究机构具有各自的技术优势，因而提出的解决方案会有不同之处。IETF 作为 Internet 协议的起草组织，考虑各方面的因素，决定以 Cisco 的标签交换为主构造多协议标签交换（MPLS）的框架[19][46-48]。并专门建立了 MPLS 标准化工作组，起草 MPLS 相关协议。

2.2.1 MPLS 的概念

MPLS 的出现是源于早期的 IP 交换，其目的是将目前的各种 IP 路由技术和 ATM 交换技术兼容并蓄，以提供一种更具有弹性、扩张性以及效率更高的宽带交换网络。MPLS 包含了许多熟悉的 IP 交换概念，一个 MPLS 网络采用标准分组处理方式对第三层的分组进行转发，采用标签交换对第二层分组进行交换。网络工作是基于对等模型的，也就是网络中的每一个交换路由器(称之为标签交换路由器 LSR)都运行单一的 IP 选路协议、交换路由表更新信息；并维护一个拓扑结构和一个地址空间。

MPLS 网络如图 2.1 所示，一个 MPLS 网络的核心结构组成为：标签边缘路由器 LER(Label Edge Router) 和标签交换路由器 LSR(Label Switch Router)。网络中的 MPLS 路由器在控制器的控制下将 FEC(转发等价类)映射为标签(Label)，并由 LDP(标签发布协议)完成从入口到出口的标签交换

路径的建立。标签边缘路由器 LER 处于 MPLS 网络边缘，和非 MPLS 网络相连。当分组(IP 信包、帧中继或 ATM 信元)进入 MPLS 网络时，入口 LER 根据输入分组头查找路由表以确定通向目的地的标签交换路径 LSP，把查找到的对应 LSP 的标签插入到分组头中，然后将分组输出到标签标识的路径，按交换方式在网络中进行传送，后续的节点 LSR 只需完全根据分组标签进行标签交换式转发，无需再查路由表，从而大大简化了转发过程。当分组到达出口 LER 时，出口 LER 将标签剥去，并按第三层转发来处理分组。因此，MPLS 的实质是将路由器移到网络的边缘，将快速、简单的交换机置于网络中心，对一个连接请求实现一次路由、多次交换，由此提高网络的性能。

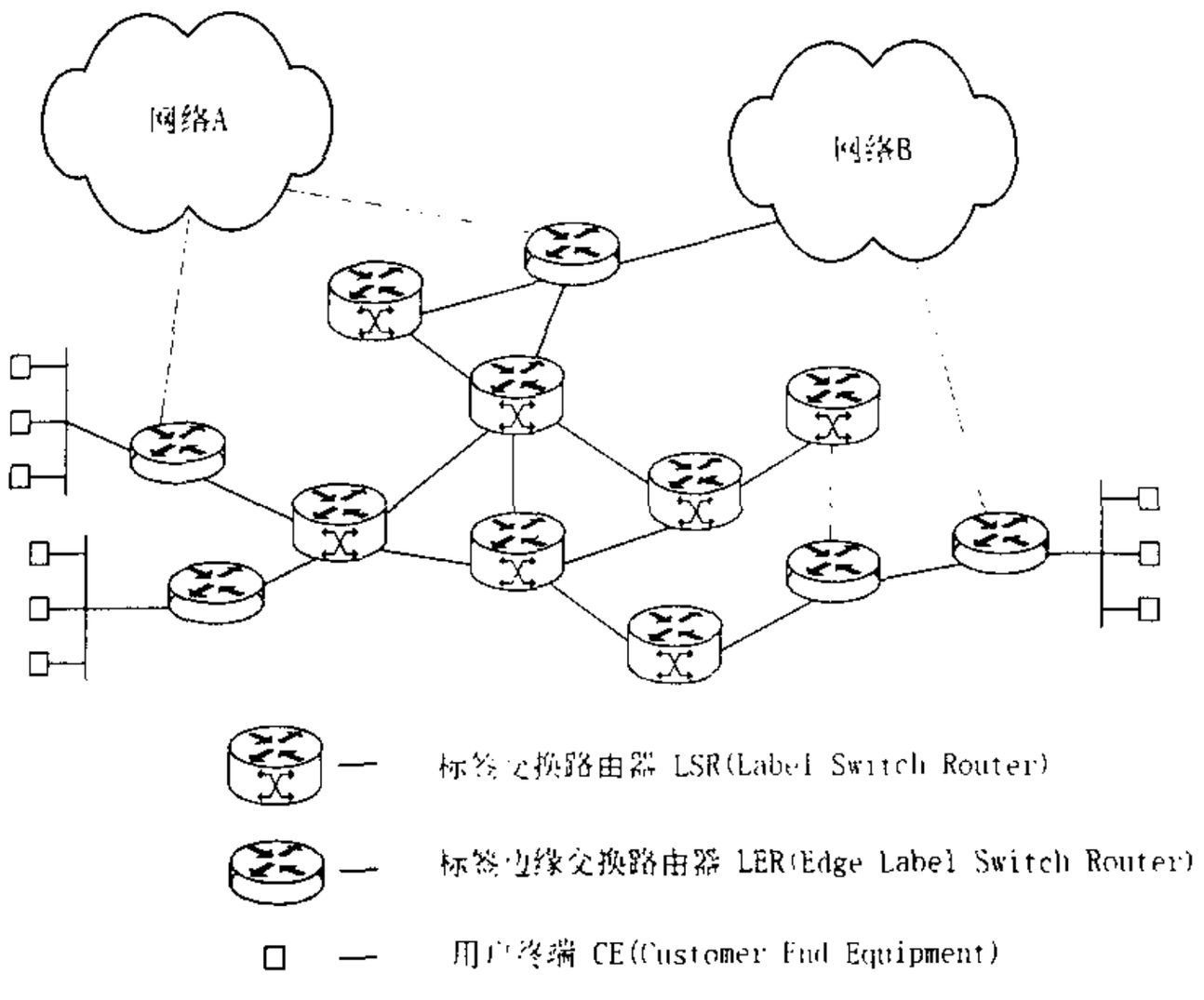


图 2.1　MPLS 网络结构示意图

一个 MPLS 网可由多个支持 MPLS 的 MPLS 域组成，但 MPLS 域的划分与路由域的划分是互不相关的。在一个路由域内可以同时包含 MPLS 节点和非 MPLS 节点，标签信息只在 MPLS 的相邻节点间传递。通常 LSP 的建立使用如 OSPF、BGP 等常规的 IP 选路协议。另外 MPLS 可运行在任何链路层上，例如 ATM、帧中继或点到点（PPP：Point-to-Point Protocol）

协议。

2.2.2 MPLS 的技术优势

以 MPLS 设备构成的骨干网相较于以传统路由器设备构成的骨干网来说，网络性能在以下各个方面得到了有效改进：

1) MPLS 简化了分组转发

MPLS 标签交换分组转发是基于定长短标签的完全匹配，而不是像路由器那样需要运行最长匹配算法。另外，MPLS 使用的标签头比典型的数据报协议（如 IP）的分组头要简单。这意味着 MPLS 允许比数据报更简单的转发规范，也意味着采用 MPLS 能够更容易地制造出高速路由器。而且这个转发的硬件基础是便宜、成熟的 ATM 交换技术，这大大减少了设备制造商的研发投资，加快了 MPLS 设备的面市时间和产品的成熟稳定性。

2) MPLS 隔离了路由和交换功能

由于 MPLS 将路由与分组转发从 IP 网中分隔开来，这使得在 MPLS 网中可以通过修正转发方法来推动路由技术演进；同时新的路由技术可以在不间断网络运行的情况下直接应用到网络中，而不必改动现有路由器上的转发技术，这是以前的各种网络技术不易做到的。

3) MPLS 支持有效的显式路由

显式路由技术是一种很有效的骨干网路由技术，尤其是在实现网络负荷调节、保证用户需求的 QoS 要求、提供差分服务等方面起着重要的作用。然而，在纯数据报路由中，每个分组头都携带显式路由是不可能的。但是 MPLS 只是在标签交换路径建立时所用的信令信息分组中允许携带显式路由，而不是在数据传输分组中携带。这意味着 MPLS 使得显式路由切实可行；也意味着 MPLS 可以利用显式路由带来的许多好处。

4) MPLS 能很好地实现流量工程

流量工程指根据用户数据业务量及当前网络状态选择数据传输路径的

过程，它主要用来平衡网络中不同链路、路由器和交换机的流量负荷。

在数据报路由方式下，流量工程的实现非常困难。通过调整与网络链路相关的度量能实现一定程度的负载平衡。然而，这种办法效果有限。在大型网络中，每两个节点之间都有多条路径，仅仅靠调整一跳接一跳(hop-by-hop)的路由度量是难以实现所有链路间的业务量平衡。

MPLS 允许从任意规定入口节点到任意规定出口节点的径流被单独标识。这使得 MPLS 能提供一种直接的机制来测量每个入口和出口对之间的流量，此外，由于 MPLS 可采用显式路由的标签交换路径，因此它还能保证任何特殊数据径流能按规定的路径传输。

2.2.3 MPLS 技术的核心技术及组件

在 MPLS 技术方案中有一些核心技术及组件：

1) 标签[49]（label）

标签是一个简短的，具有固定长度的，具有本地意义的标识符，它用以识别转发等价类 FEC。它是 MPLS 网络中的一项核心技术。MPLS 的许多优点都直接或者间接地来自于标签的使用。在 MPLS 网中进行分组转发的过程实际上就是标签交换操作的过程。

标签的格式取决于分组封装所在的介质。例如，ATM 封装的分组（信元）采用 VPI/VCI（Virtual Path Identifier/ Virtual Channel Identifier）数值作为标签，而帧中继 PDU(Protocol Data Unit)采用 DLCI(Data Link Circuit Identifier）作为标签。对于那些没有内在标签结构的介质封装，则采用一个特殊的数值填充。

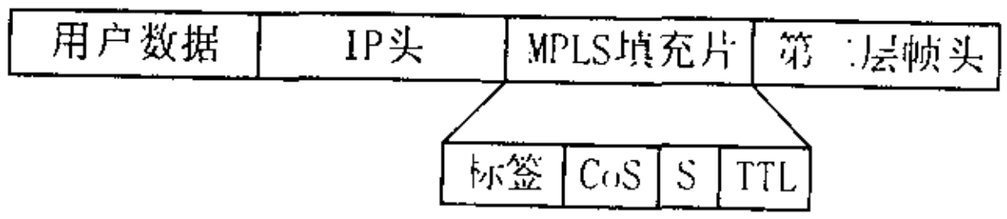


图 2.2 MPLS 标签格式

图 2.2 给出 4bytes 填充标签的格式，它包含一个 20bits 的标签值、一个 3bits 的 CoS（Class of Service）值、一个 1bit 的堆栈标识符和一个 8bits 的 TTL（Time-To-Live）值。此外，如果填充值被插入到一个 PPP 或以太网帧中，包含在各帧头中的一个协议 ID（或以太网类型）表示一个帧或者一个 MPLS 单播或组播帧。

2) 标签交换路由器（LSR）和标签边缘路由器（LSR）

如图 2.1 所示，标签交换路由器（LSR）和标签边缘路由器(LER)是 MPLS 网络中的节点。LSR 位于 MPLS 网络的中部，主要运行 MPLS 控制协议和第三层路由协议，并负责与其它标签交换路由器交换路由信息来建立路由表，实现转发等价类与 IP 分组头的映射，建立 FEC 和标签之间的绑定，分发标签绑定信息，建立和维护标签转发表等工作。LSR 除了支持标签交换以外，还支持第三层的 IP 分组逐跳式转发。

标签边缘路由器 LER 主要完成连接 MPLS 域和非 MPLS 域以及不同 MPLS 域的功能。并实现对业务进行分类、分发标签(作为出口 LER 时)、剥去标签等；它甚至可确定业务类型，实现策略管理，接入流量工程控制等工作。LER 是实现 MPLS 网络的关键功能设备之一。

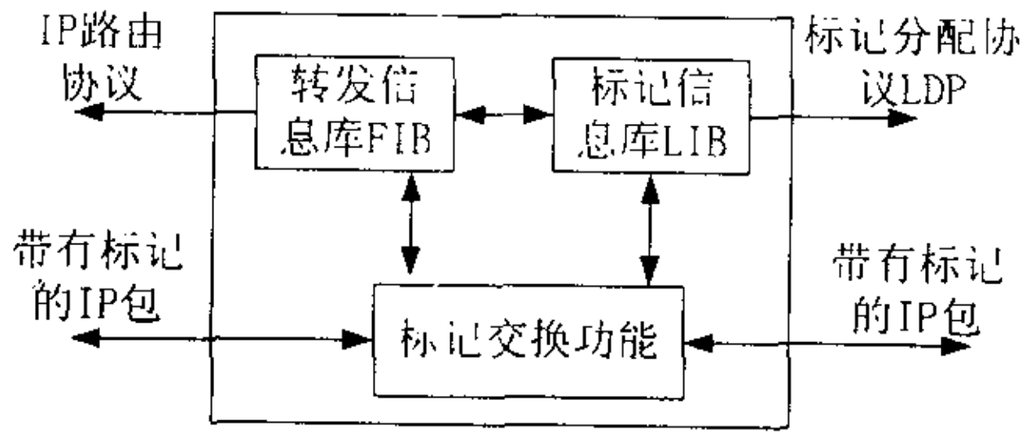


图 2.3 标签交换路由器的基本组成示意图

标签交换路由器 LSR 的基本组成如图 2.3 所示。由图可知，标签交换路由器是由 IP 选路的转发信息库 FIB（Forward Information Base）、标签信息库 LIB（Label Information Base）以及标签交换功能模块等组成。

3) 标签交换

MPLS 采用的一个核心技术是转发机制，即标签交换。实现标签交换是个快速和简单的转发过程，因为它不必像传统 IP 选路那样分析分组头中的变长部分。标签作为一个整体（也可能是标签中附加的字段，如 TTL（Time-To-Live）和 CoS（Class of Service））由交换机组件处理，即使一个分组包含一个多级的标签栈，MPLS 设备只负责处理栈中的顶部标签。

4) 标签发布协议：LDP[50]

标签发布协议 LDP（Label Distribution Protocol）是控制标签交换路由器之间交换标签与 FEC 绑定信息，协调 LSR 间工作的一系列规程。

LDP 的主要功能是让 LSR 实现 FEC 与标签的绑定，并将这种绑定通知给相邻的 LSR，以使各 LSR 间对收到的标签绑定达成共识。LDP 的最终目的是完成标签分发，并让各 LSR 对等体间在标签语义上达成一致理解，从而建立整条标签交换路径 LSP。目前 IETF 规定的 LDP 标签分发处理过程主要有：标签分配处理、标签赋值(Label Assignment)与分发控制处理、标签维护处理和标签请求处理等。

5) 标签交换路径[51]：LSP

标签交换路径是指在某逻辑层次下由多个 LSR 组成的交换式分组传输通路。LSP 与转发等价类 FEC 相对应。对一 FEC F，可以有多个入口 LSR。每条以这些 LSR 为起点的 FEC F 所对应的 LSP 将形成以出口 LSR 为根，以入门 LSR 为叶的“LSP 树”，我们称这颗树为 FEC F 的专有 LSP 树。

在 MPLS 网络中标签交换路径 LSP 的形成可分为三个过程：

(1) 网络起动后在路由协议如 OSPF、BGP、IS-IS（Intermediate System-Intermediate System）等的作用下在各节点中建立路由转发表。

(2) 根据路由转发表，各节点在 LDP 控制下建立标签交换转发信息基 LIB。

(3) 从入口 LSR、中间 LSR 和出口 LSR 的输入输出标签相互映射拼接起来后，就构成了从不同入口点到不同出口点的 LSP。

2.3 基于光电混合 Tbps 量级 MPLS 路由器

随着 IP 网络的飞速发展和宽带技术的不断出现，Internet 互连的核心设备——路由器也走过了三代的发展历程。传统的基于总线和中央处理器结构的路由器由于其体系结构上的局限，已经无法满足组建高速主干网络的需求。同时，各种应用和技术(如千兆/万兆以太网)的出现对网络带宽的要求越来越高。对于网络而言，关键的瓶颈不在于光纤物理层数据的传输，例如，光纤通道的速率从 OC-3(155Mbps)一直上升到 OC-192(10Gbps)，而且通过密集波分复用技术(DWDM)，可进一步利用光纤潜在的带宽。路由器成为了网络的主要瓶颈。在单端口速率达到 10Gbps 量级时，需要处理速度达到 Tbps 量级的路由器，相应单端口分组查找速率要达到几十兆次查找每秒(MLPS：Million Lookups Per-Second)，同时交换背板带宽容量也需要是 Tbps 量级。Tbps 量级路由器不仅可以用于目前的网络构架(校园网、城域网和骨干网)满足各种带宽需求，而且在未来的全光网络中也会有一席之地，因此，研究 Tbps 量级的路由器具有极大的现实意义。

大体来说，建立一个 Tbps 量级的 MPLS 路由器的主要困难是：1）如何提供高速互连来实现一个大容量的交换结构；2）由于 IP 分组的长度是可变的，如何设计宽带的交换体系机构来快速转发分组；3）如何设计转发引擎来快速转发分组。

为此，我们提出了一种可扩展的基于光电混合大容量 MPLS 路由器，内部采用可扩展的 ATM 光交换机构来支持高速的交换，如图 2.4 所示。它主要由主控板、高速光背板、交换板、接口板组成。此路由器使用高速光背板提供高速互连，并将路由引擎与转发引擎分离，以一种分布的方式执行分组转发。每一个接口模块都包含一个转发处理模块，执行标签的查找以及分组转发。运行路由协议、标签发布协议以及其它控制协议的任务主要由主控板承担，以加速在每个接口上的数据传送。

这种结构的 MPLS 路由器具有以下特点：第一，它不需要额外的网络协议来支持流分类以及虚通道连接，其次，由于采用 ATM 交换芯片以及可扩展的交换结构，交换机构规模可以扩展到 1024×1024，端口速率为 2.5 Gbps，甚至更高。所以该 MPLS 路由器具有良好的可扩展性和灵活性。另外，该 MPLS 路由器既可以支持 ATM 网络，也可以支持其它类型的网络。

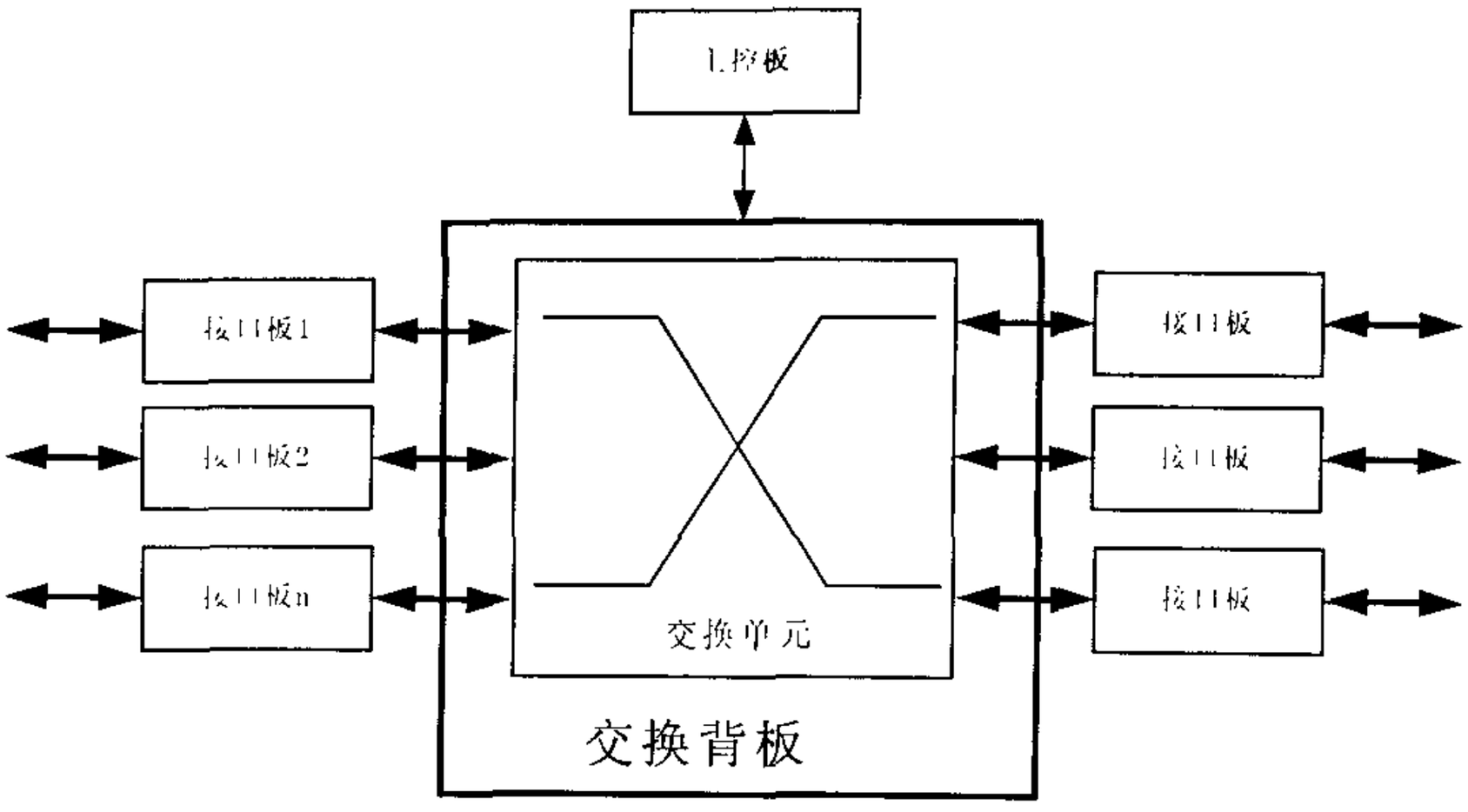


图 2.4　MPLS 路由器结构示意图

2.3.1 主控板

主控板是路由器的控制中心，主 CPU 和存储器就在主控板中。主控板负责整个路由器的管理和控制，IP 路由协议在主控板上运行。主控板直接接收来自网管中心的指令，并下发到各接口板执行指令，同时各接口板把运行状态和统计数据传送到主控板，由主控板进行必要的处理，需要时发给网管中心。网络管理员配置的静态路由以及通过运行路由协议生成的动态路由由主控板进行管理，并下发到各接口板，使各接口板可以独立地进行数据包的转发工作。主控板的作用举足轻重，一旦它发生故障，整个路由器将不能正常工作。对于电信网的核心网络设备来说，要求可用率达到 99.999%，即 1 年的停机时间不能超过 5 分钟。所以主控板通常配有两块，一般以主备的方式工作。主备板周期性地交换握手信号，一旦备用板收不到主用板的握手信号，则会启动倒换流程，接替主用板工作。

2.3.2 交换板

高速路由器的整机吞吐量很大，早期路由器的基于背板共享总线传递数据的方式已不能满足高速数据传递的需要。首先，共享总线不能避免内部冲突；第二，共享总线的负载效应使得高速总线的设计难度很大，总线上的多个端口及连接件所引起的电气负载、干扰和反射等都是极大的限制了总线的传输能力。目前，共享总线所能达到的最大传输速率为 20Gbps，这在只有几个 100Mbps 的以太网端口时还能勉强满足要求，但对于连线速率为 OC－48（2.5Gbps）以上的端口时，共享型总线就显得力不从心了。交换结构的引入逐步克服了共享总线的以上缺点。

交换结构 成为路由器设计中的最重要的部分之一。它影响到路由器的价格、性能、吞吐量、可扩展性以及路由器设计的复杂性。目前交换结构总的说来可以分为三大类：时分交换结构、空分交换结构和波分交换结构。

时分式交换结构是指所有的输入/输出端口共享一条高速的信元流通路，在一个时间片内，只有一条输入/输出路径使用该信元通道。这条共享的高速通路可以是共享介质型的，也可以是共享存储器型的。整个交换矩阵的交换容量由这个共享通路的吞吐量（如总线速度、总线仲裁速度、存储器容量和存取速度等）所限制，扩展不好。

空分式交换结构是指在输入和输出端之间有多条通路，不同的分组可以在不同通路上同时通过交换结构，空分结构一般具有较好的硬件扩展性，可以增加端口而不影响交换机的吞吐量，端口不必竞争单一的共享资源。由于交换机性能可随端口的增加而提高，所以在理论上空分交换机应能容纳很多的端口。典型的空分交换结构有：Crossbar 结构、Knockout 结构和 Banyan 类结构[52-53]。

波分结构是指信号通过不同的波长，选择不同的网络通路来实现。波分交换网络由波长复用/解复用器、波长选择空间开关和波长转换器组成。这种方式的优势十分明显，就是充分挖掘了光纤的带宽潜力，当然，其缺陷也很突出，原因在于实用的器件在一段时间内还很难实现。

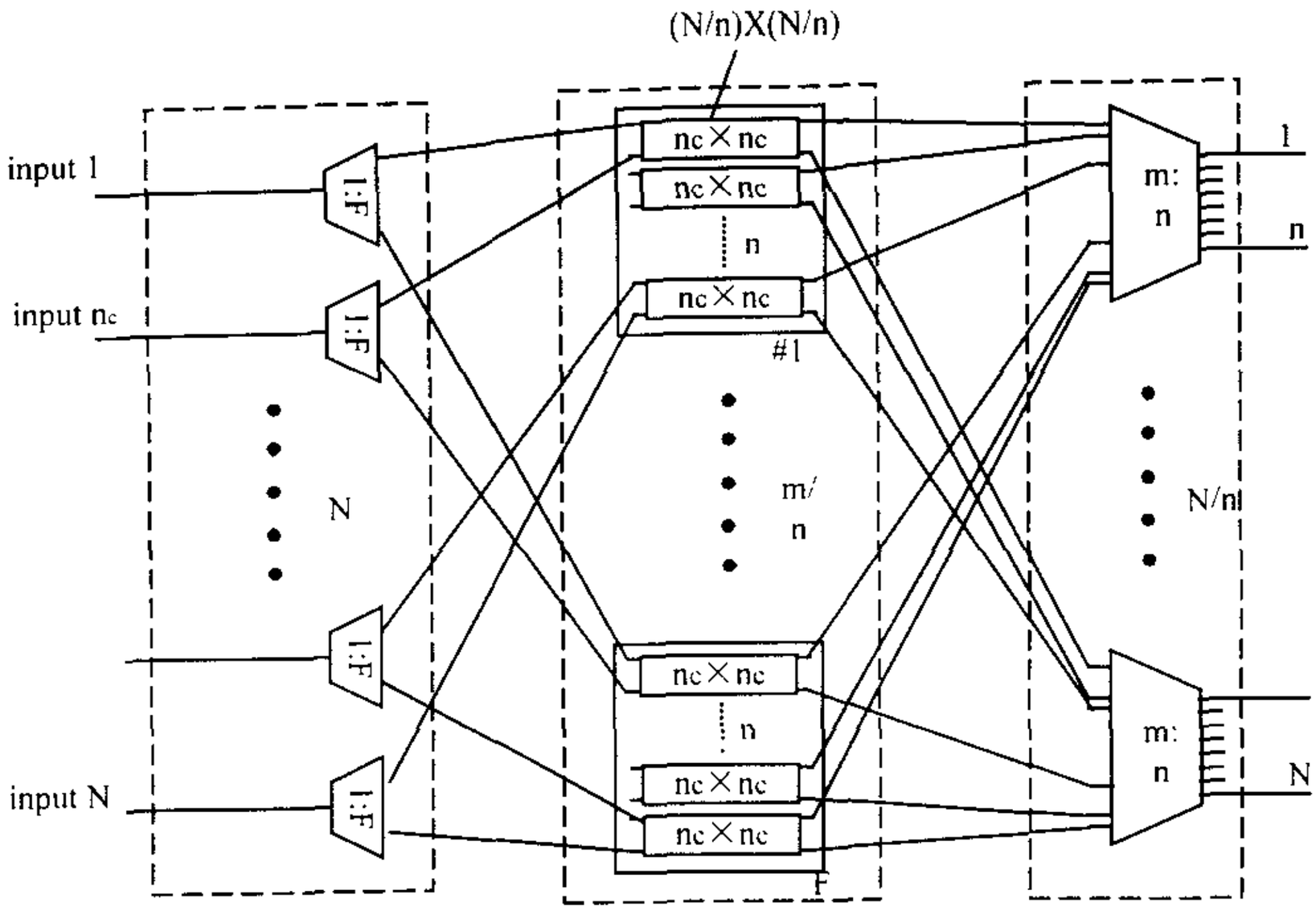


图 2.5 基于小 crossbar 的分组互连交换系统

本实验室采用多输入端口共享链路的可扩展的分组交换结构，如上图所示。从图中可以看出，连接到第一个 pipe 的第一个 crossbar 上的 n_c 个输入端，在其它几个 pipe 中也都连接到相应 pipe 中的第一个 crossbar，其余的输入端与此类似，因此可以把互连机构分为互不相关的 N/ n_c 个组，每组 n_c 个端口，连接到所有 pipe 中的同一个 crossbar 中。因此，只有在相同 crossbar 输入端口上的 n_c 个输入端来的信元间才有可能发生争用路径的情况。

2.3.3 高速光背板

传统路由器一般使用共享式总线传递数据，并在近几年引入了越来越高速的共享式总线，从 ISA 到 EISA 和现在的 PCI.但依然存在上文所述的缺点。在高速路由器中引入了交换网后，在背板上用点到点的高速连线把用户板连接到交换网，进行高速数据传递。由于交换网是备份的，背板上的数据线也是双套的。同时背板上仍存在控制总线，用于交换控制信息，当然控制信息也可以在数据通道的带内传送。

由于 Tbps 量级的 MPLS 路由器要求有极高的端口速率，完全采用纯电子的技术来实现这样的大容量路由器是非常困难。采用纯电子技术来实现大容量的路由器时，必须采用高度并行的方法把输入输出线的高速信号在交换机内部降为电互连线能够有效传输的低速并行信号，并通过互连大量的小容量模块来实现较大的总交换容量。但采用这种方法，一方面会引起交换机体积、成本的迅速增长，另一方面也会引起整体性能的下降。同时，由于高速连线众多，背板的设计和加工也非常复杂。

因此，为了克服电互连线传输速率难以提高的固有缺点，在我们所设计的 Tbps 量级的 MPLS 路由器中引入了光互连技术。光互连具有极高的空间时间带宽积，抗干扰能力强、互连通道等程、低功耗等优点。采用光互连技术是高速大容量路由器发展的必然趋势。

2.3.4 接口板

高速接口板是高速路由器设计中的关键与难点。每个接口板主要由数据收发模块、数据处理模块、转发处理模块、接口控制模块和 SAR 模块组成。数据收发模块与 OC3、OC12、OC-48 或 Gigabit 以太网连接。SAR 模块将不定长的 IP 数据包分割成定长的内部交换信元流用于交换。转发处理模块执行标签查找和替换。

传统的路由器采用集中的转发控制可以减少接口板的开销。但是，采用集中路由时，所有的标签都必须通过交换结构进出转发处理模块来获得路由判决。在输入接口和转发处理模块之间通过交换结构传输标签信息和判决信息，这大约占用了 25%的交换机构的带宽资源。另外，标签查找判决必须足够快，这就需要在路由引擎中采用功能强大的处理器。否则，接口板中将产生阻塞。

而我们提出的 MPLS 交换结构采用分布式控制的策略，在每一个接口板中都有一个本地转发处理模块来执行标签查找和转发判决。高速转发引擎的设计是高速路由器设计中的关键与难点。在后面章节中，将详细的论述高速接口板的设计方法与过程。

2.4 本章小结

在本章的开始，简单地介绍了 MPLS 技术的基础知识。揭示了 MPLS 的核心技术——标签。整个 MPLS 协议都是建立在这一基础上的，围绕标签的一些操作——映射、绑定、封装、分发、交换、保持、合并以及清除而形成了 MPLS 的一整套协议规范。

然后描述了一种基于可扩展 ATM 交换结构的 Tbps 量级 MPLS 路由器。该 MPLS 路由器采用每接口模块一个转发表的分布控制方式来实现分组的快速标签查找和转发，而路由查找则交给主控制模块进行。同时，为了保持 MPLS 路由器对外部节点的透明，在每一个接口上将分组打包成固定长度的信元进行交换，交换过的信元重组后再转发到下一跳。同样，为了保持对外部节点的透明性，我们采用了内部标签来标识分组的目的输出端口，因此在接口上进行标签查找时将添加一个表示目的输出端口的内部标签到分组上。当交换完成后，将此内部标签剥离。另外，为了支持 VC 合并，也采用内部标签的一部分来标识分组的输入端口信息。

在后面的章节中，将详细地论述高速接口模块的设计。

3 MPLS 路由器接口模块的总体方案

3.1 前言

现代高速路由器设计的基本思想主要包括 4 个方面：(1)将路由引擎 (routing engine)和转发引擎 (forwarding engine)分开，将局部转发表从全局路由表中独立出来；(2)用快速的硬件实现 IP 报文的报头处理、寻径和转发；(3)用多个分布式的接口模块加中央控制器的模式取代中央处理器加接口卡的模式；(4)用交换结构 (switch fabric)提高各接口模块之间的数据通信速度。

这些基本思想中前三个方面直接涉及到接口模块。现代高速路由器设计中，都采用每接口模块一个转发引擎和局部转发表的分布控制方式来实现分组的快速查找和转发。因此，高速接口模块的设计成为了现代高速路由器设计中的关键和难点，这还基于如下的事实：

传统路由查找的速率最高只能达到几个 Mpps 这样的量级，如果接口线卡以这样的速率转发分组，要实现 Tbps 量级的路由器是不现实的。

在高速接口模块设计中的关键技术主要是：1） 高速转发引擎的设计；2）将变长的 IP 数据报分割成定长的内部交换信元流，便于交换模块的高速交换；3）用快速的硬件实现 IP 报文的报头处理、寻径和转发；

3.2 接口模块的设计

在本课题中，MPLS 路由器接口模块包括数据收发模块、数据处理模块、转发处理模块、接口控制模块和 SAR 模块五部分，接口模块示意图如图 3.1 所示。

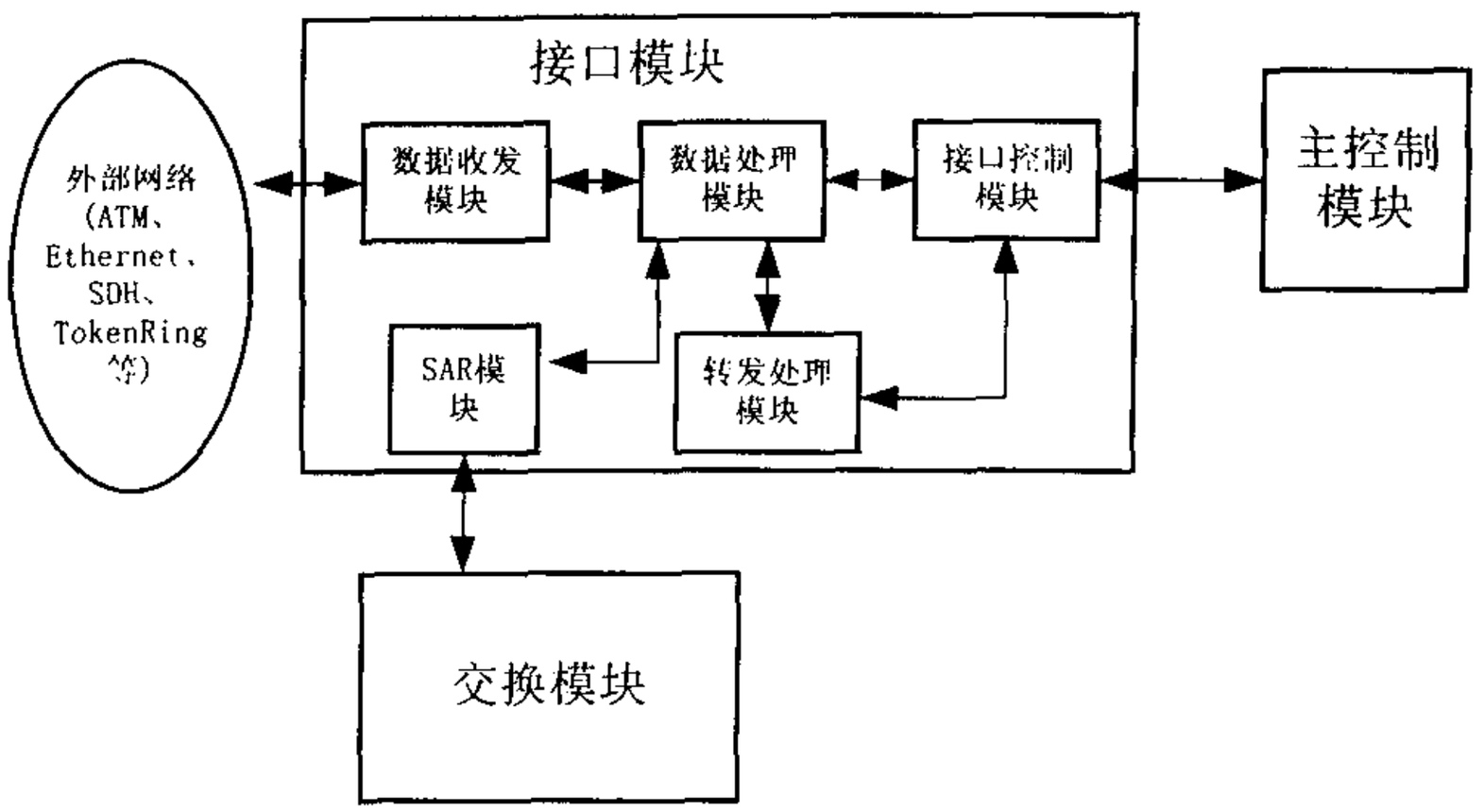


图 3.1　MPLS 路由器接口模块示意图

3.2.1 数据收发模块

现有的计算机网络中的物理设备和传输媒体的种类非常繁多，而通信手段也有许多不同的方式。接口所用接线器的形状和尺寸、引线数目和排列、固定和锁定装置等；数据线上电压的范围，‘1’或‘0’电平的电压表示；以及数据传输时所用的信道编码等等。这些都因接口的不同而有所不同。

数据收发模块的作用正是要尽可能地屏蔽掉这些差异，使其上面的协议处理单元感觉不到这些差异，这样就可以使上面的协议处理单元只需要考虑如何完成本层的协议和服务，而不必考虑网络具体的传输媒体是什么。

数据收发模块主要完成的是物理层的适配：机械特性、电气特性、性能特性以及规程特性的适配，并将物理层的数据提取出来，经过适当的电平、码型变换后，上传给数据处理单元处理，同时，将来自数据处理单元的数据进行相应的变换后，传入外部网中。

3.2.2 数据处理模块

数据处理模块完成从不同的物理层和数据链路层信息中提取出 IP 分

组提交给转发处理模块。它包括协议控制器和 IP 头处理器，其具体逻辑结构图如图 3.2。

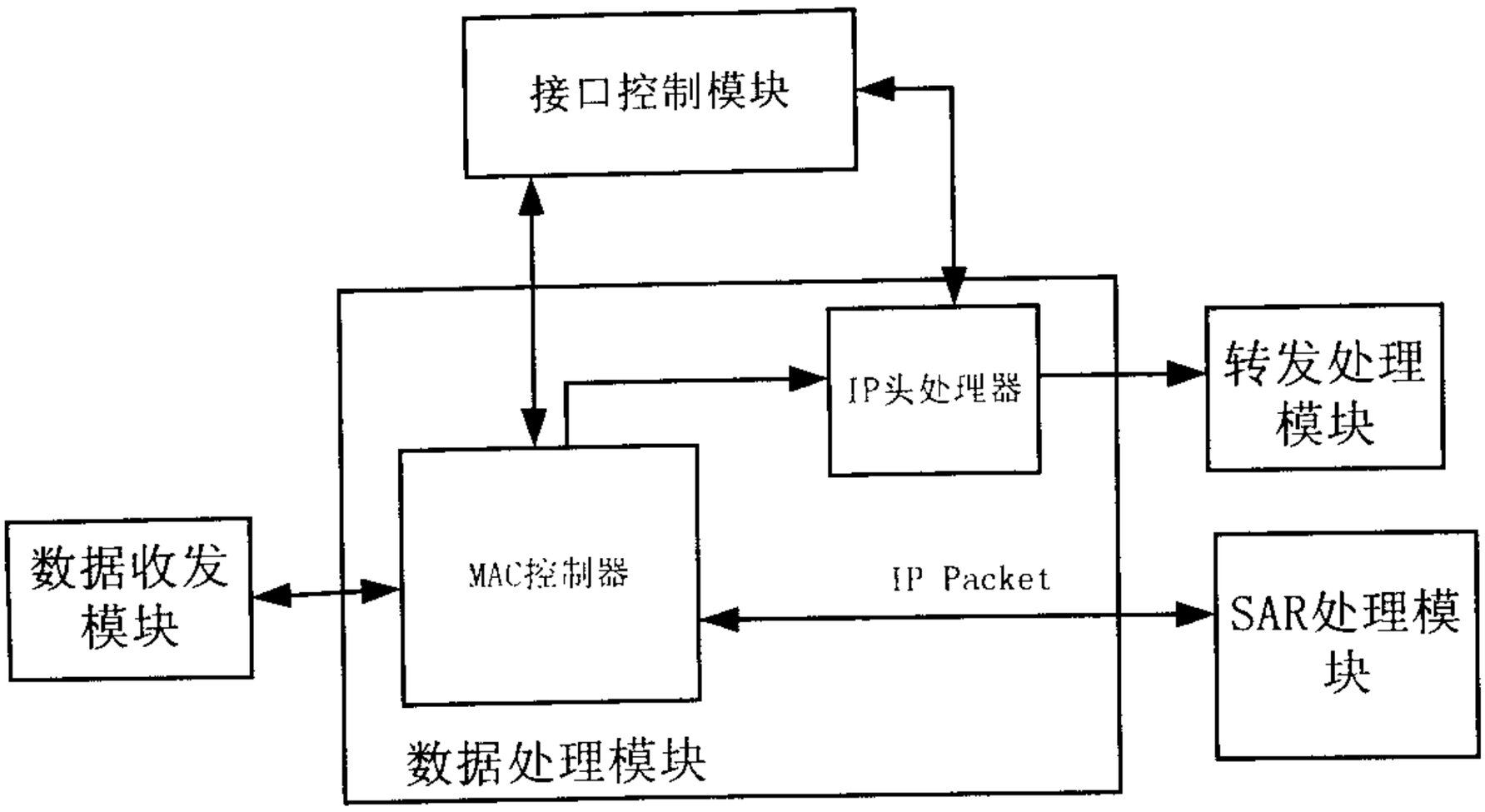


图 3.2　数据处理模块示意图

由于需要作数据包的处理，实时性的要求较高，因而应避免包的排列，力争以线速完成处理，这就要求各模块的功能尽可能采用并行处理实现。

3.2.3 转发处理模块

转发处理模块是接口模块的核心部分，也是整个路由器的核心部分之一。在高速路由器设计中，将路由与转发引擎分开是发展的趋势。现在，网络的接入线速不断提高，从 10M 以太网接口到 155M ATM 接口，再到现在的 OC－192(10Gbps)POS(Packect Over SDH)、10GbE(Giga bit Ethernet)接口。传统的软件实现转发已经无法满足目前需要，在新一代路由器结构中，转发引擎采用分布式设计，在接口模块中直接用硬件实现数据的转发。但传统网络中转发算法较为复杂，用硬件实现转发十分复杂。

大体上，高速路由器转发引擎设计的困难主要是：1）高速的路由表查找；2）各部件的高速实现。

a) 高速的路由表查找

随着网络规模的急速膨胀,路由表容量相应增长,如 CISCO 公司的 GRF 系列有 15 万行,7500 系列有 25 万行,GSR-12000 有 100 万行。对于查找路由表,虽然采取了一些措施,如对经常查找的地址采用缓存的方式,对子网进行总结性归类以缩小路由表项等,但仍未找到根本性解决措施。庞大复杂的路由表查找困难,是路由器发展中遇到的最大难点。

高速路由表查找的主要性能方面问题有两个:一是存储器的存取速度,主要由存储器访问数量和访问速度决定,这个是出现在的制造技术水平所决定的;二是路由器查表算法。

线速率查表是高速骨干路由器中转发引擎的一项关键技术。路由器根据到达数据包的目的 IP 地址通过路由转发表查找下一跳路由端口号,检查该 IP 包是否合法,重新计算校验和,然后将该数据包复制到相应输出端口。

衡量转发引擎的性能,通常用转包率表示每秒转发引擎处理的 IP 包个数。针对不同的输入端口速率,转发引擎对应有最小的转包率。若转发引擎的转包率小于此值,当满负荷情况下,将出现丢包。不同的输入端口速率,对查表的时间要求也不同,以 2.5Gbps 端口为例,满负荷情况下 (数据包长为 50 字节)查表时间是:

$$(50X8)/2.5=160\ ns$$

当前,实现线速率查表的方法主要有软件和硬件两类,但在 160 ns 的时间内根据目的 IP 地址查到该 IP 包的下一跳地址,是单靠软件方式难以胜任的。

传统 IP 路由器中采用的路由器查表算法是最长前缀匹配算法[54]。最长匹配查找算法过程为:将目的 IP 地址与路由表的各个入口表项中的掩码相与,如果其结果与该入口表项对应的子网地址和掩码相与的结果相同,那么表示该入口是对该目的 IP 地址的一个匹配,记录目的地址与该入口匹配所对应的 1 的个数;对路由表的所有入口表项重复该过程,找到其中具有最长的 1 的个数的那个入口,该入口即为对该目的 IP 地址的最长匹配表项。常见的保存路由表的数据结构是树,在树中,每条从根节点到叶子节

点的路径就对应着路由表中的一项。这样，寻找最长的前缀就转化成为寻找最长的路径.一般来说，基于树的算法从树的根节点开始，使用目的地址中的下若干位来匹配当前节点的子节点，直到找到一个匹配为止。因此，在最坏情况下查找路由表所花费的时间和找到的最长前缀匹配的长度成正比。基于树的算法的主要思想是大多数节点只需要保存很少的子节点而不用保存所有可能的值。这类算法节约了内存，付出的代价是需要进行更多次的内存查找。随着内存价格的下降，这种设计方法已经越来越不常用了

当前，提高路由器查表速度的三种技术如下：

首先是面向硬件的技术：对于较小的路由表可采用高速缓存方式，能有效地提高查表速度。具体做法是将组合电路与存储器集成在一起，形成智能存储器。常用的基于硬件的技术是用相联存储器 CAM (Content Addressable Memory)和高速缓存来提高查找速度。CAM 的优点是实现简单，有商用芯片可用，但其表项容量不大，成本高，而且很难随着路由表的变化而扩展，因此不能用在需要使用大路由表的主干路由器上。

第 2 种基于硬件的技术是通过增大使用的内存来存储路由表[55]，设计并行算法进行查.通过空间换取时间。这种方法可以减少存储器访问次数，也能提高其速度。这样做虽然提高了路由表的查找速度，但却带来了路由表更新的不便。 1 条路由的改变就需要更新大量的路由表项。

第二是表压缩技术[56]：通过某种算法将较大的路由表压缩变小，建立更复杂、紧缩的数据结构，然后存入高速缓存。表紧缩技术是使用复杂但是紧缩的数据结构来保存路由表。这样，路由表就可以放在处理器的第 1 级高速缓存中。这种方案可以支持千兆比特速率的路由表查找。使用文献[56]中提出的数据结构，可以把具有 40000 条路由的路由表压缩到只有 150K 字节。这样做虽然压缩了路由表，提高了路由表的查找速度，但却带来了路由表更新的不便。

第三是 Hashing 技术[57]：哈希表技术也可以用于路由表查找。寻找最长前缀匹配的需求限制了哈希表技术的使用。在实际使用中，我们并不知道一个目的地址对应的最长前缀匹配是多长。解决这一问题的方法是尝试不同长度的掩码，从中选择最长的匹配。掩码的选择既可以使用迭代方式，也可以使用层次方式，还可以使用地址的前几位指向一个前缀长度列表。

但是，这些方案的可扩展性都很差。

采用最长前缀匹配算法时，一个报文可能会同时匹配多个路由表项，我们必须在这多个路由表项中寻找最长的匹配。这样，就增加了路由表查找的时间，包转发速率因此而下降。因此，我们如果能不使用最长前缀匹配算法，包转发速率将得到很多的提高，同时，更是大幅度的消除了硬件实现时的复杂度。而在 MPLS 网络中，使用定长标签查找替换了传统的最长前缀匹配，这样消除了路由器查找算法中的一个难点。

在“传统的”路由体系结构中，由控制功能器件（比如单播路由、组播路由和有服务类型的单播路由）提供的不同功能需要在转发功能器件中具备多种转发算法[58-62]（如表 3.1 所示）。比如，单播分组的转发需要在网络层目的地地址基础上的最长匹配算法；组播分组转发需要在网络层源地址基础上的最长匹配算法以及在网络层源地址和目的地地址基础上的精确匹配算法；而带服务类型的单播分组的转发则需要在网络层目的地地址基础上的最长匹配算法以及由网络层头携带的在服务类型比特基础上的精确匹配算法。

表 3.1 传统的路径体系结构

路径功能	单播路径	具有服务类型的单播路径	组播路径
转发算法	在目的地地址上的最长匹配	在目的地地址上的最长匹配+在服务类型上的精确匹配	在源地址上的最长匹配+在源地址、目的地地址和输入接口上的精确匹配

标签交换的一个重要特点是在其转发功能器件中没有多种转发算法：在标签交换的转发功能器件中只包含一种转发算法——在标签交换技术基础之上的算法（如表 3.2 所示）。这就在标签交换和传统的路由体系结构之间形成了一个重要的差别。这种单一的转发算法在实现上比以前相应简单了不少，能更容易的用硬件实现。

表 3.2 标签交换的体系结构

路径功能	单播路径	具有服务类型的单播路径	组播路径
转发算法	公用的转发（标签交换方法）		

在路由表查找算法上，我们采用相联存储器 CAM 和高速 RAM（Random Access Memory）的配合使用，完成高速查找。

CAM 是一种专用存储器件，可进行快速大量并行搜索。搜索的时候，存储器中所有的数据同时与搜索关键字比较，搜索结果就是匹配项的物理地址。它可以在硬件中完成数据表查询，需要使用专用比较电路，对每个存储位进行比较。

为了保证快速搜索，CAM 通常采用管线结构，每个时钟周期都能启动搜索，运行速度可以维持在每时钟周期搜索一次。目前 CAM 的最高搜索速度为每秒 1 亿次，即可达到 100MPPS（Million Packets Per-Second）的查找速度，该速度可在 OC-768 中对每个信息包进行一次搜索或在 OC-192 中对每个信息包进行四次搜索。也就是说，对于 10G 以太网单个设备可以支持每个信息包六次搜索，或每个信息包在两个合计起来 10G 的以太网端口进行三次搜索。

前面已经提到过，CAM 的缺点在于其表项容量不大。因此，为了更好的发挥 CAM 高速查找的优势，局部转发表只是主路由表的一部分高速缓冲。

b) 各部件的高速实现

在传统的路由转发过程中，路由表的查找的延时是转发过程中最主要的延时。采用 CAM 器件实现 MPLS 标签的定长查找后，路由表的查找延时有了大幅度下降。这时，使用软件实现的 IP 报头处理过程，成了转发过程中的瓶颈所在。为了解决这个问题，我们采用了 FPGA 硬件设计，快速的实现了这一部分的处理。在用硬件快速实现的前提下，为了更快地支持

数据转发处理模块的高速运转，对这一部分进行流水线设计，实现了并行处理，提高了包转发率，如下图所示。分组头的校验、路由/转发表的查找、TTL 减一以及校验和的更新是同时并行进行的，当然只有在通过分组头的校验后，才能发出信号触发使能路由/转发表的输出和分组头的更新输出。

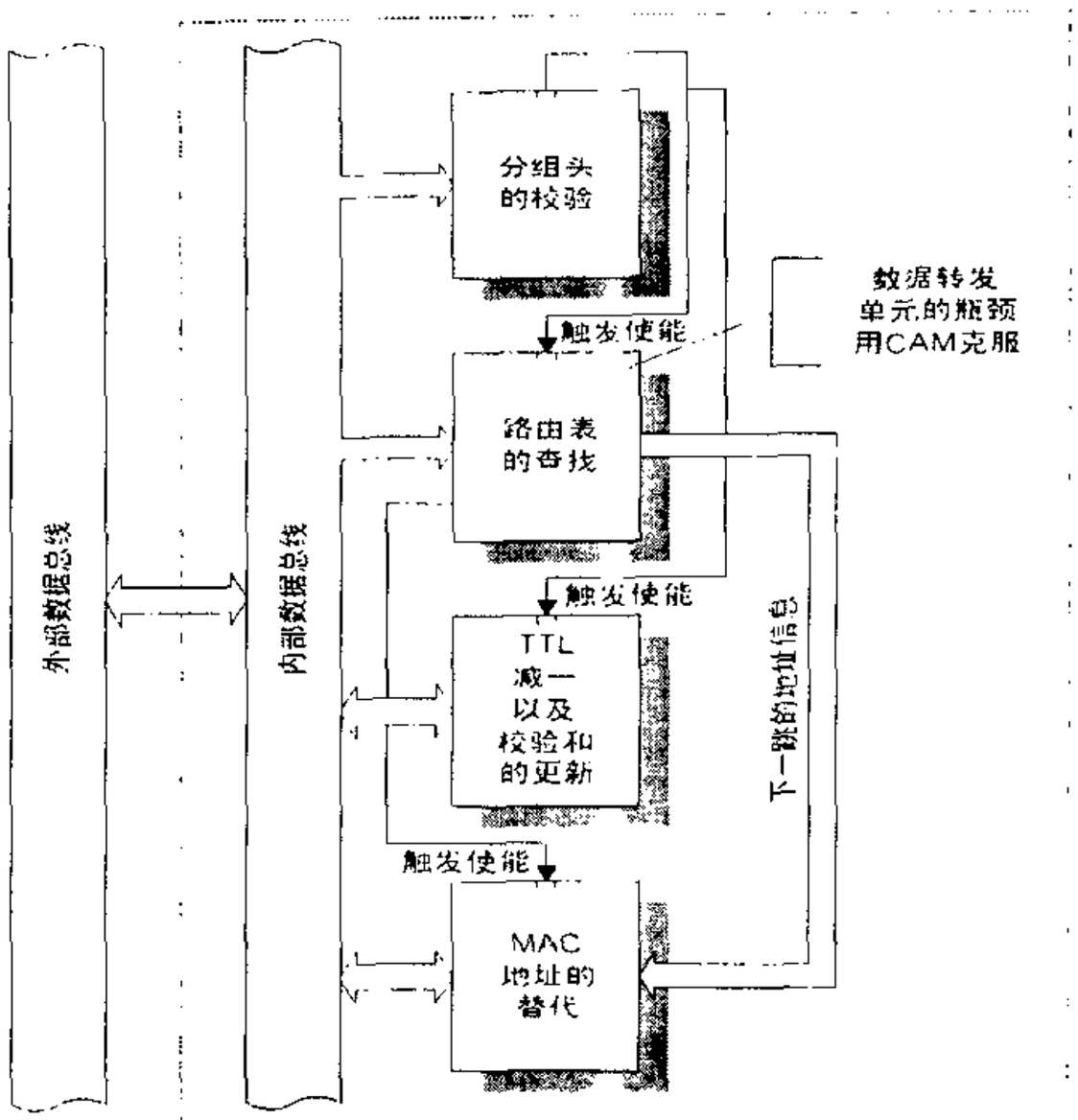


图 3.3 转发处理模块的流水线设计

3.2.4 SAR（Segmentation And Reassemble）模块

SAR 模块是高速 MPLS 路由器接口模块设计中的关键和难点。

IP 数据包通过交换结构的时候，可以是以定长模块的形式（通过数据包的定长分割），也可以不进行分割直接进行变长交换。一般高性能的交换结构都采用了定长交换的方式，在数据包进入交换结构以前把它分割为固定长度的 cell，这些 cell 通过交换结构以后被按照原样组织成原来的变长包（packet）。定长交换方式更利于交换网的控制，分组长度一样，判断其传输和离开的时刻就很容易。在时隙结束时，调度表检查等待传送的分组，

决定下一个时隙哪个输入与哪个输出相连，避免输出或输入端的空闲，保持交换机的高效率。而且从硬件设计的角度讲，处理固定长度分组比处理不同长度的分组更简单、快速。同时定长交换可以避免某些业务流的大长度包长时间占用交换网，影响高优先级业务和实时业务的交换。

SAR（Segmentation And Reassembly；分割与组装）模块它包括分割与组装两部分，完成对 IP 数据包的分割和组装。

3.2.5 接口控制模块

接口控制模块主要完成对转发表的刷新与维护，同时处理相关 IP 控制信息。

由于外部网络的拓扑的变化是不可预知的，标签/路由表也不是固定不变的，需要定期刷新与维护。同时，接口的标签/路由表是此路由器的主标签/路由表的一部分，当输入的数据流在接口的标签/路由表中查找不到所对应的项时，接口控制模块就将此 IP 包的整个 IP 头（包括标签）直接封装后，送入交换模块在主标签/路由表中进行查找，这个信元拥有比一般数据高的优先级。结果送回 IP 头处理逻辑，同时更新标签/路由表。在实际设计中，主 CPU 占交换模块的一个端口，具有较高的优先级，它和接口控制模块间的通信是通过交换模块进行的，先 SAR，再交给交换模块。将根据输入数据所查表项的频度对标签/路由表进行相应的调整。

可以看到接口控制模块从数据处理模块和转发处理模块中取得相应控制信息：

a. 该信息可能是 IP 网络控制流。如 ARP 报文，边缘路由器一般都需要在接口中实现 ARP。

b. 若是 MPLS 网络管理信息，接口控制模块对此信息进行分析，并将其转送给主控制模块。

c. 若是路由消息，接口控制模块直接将此消息转送给主控制模块。

3.3 本章小结

本章从整体上论述了接口模块的设计方案，同时详细地讨论了高速 MPLS 路由器接口模块的关键技术：1） 高速转发引擎的设计；2）将变长

的 IP 数据报分割成定长的内部交换信元流，便于交换模块的高速交换；3）用快速的硬件实现 IP 报文的报头处理、寻径和转发；并提出了相应的解决方法。详细地论述了路由表的查找算法，给出了一种高速的查找方法——CAM 硬件查找。并将接口模块划分为数据收发模块、数据处理模块、转发处理模块、接口控制模块和 SAR 模块五部分。其中数据收发模块负责以太网数据帧的收发，采用了 LEVEL ONE 公司（现已被 INTEL 公司收购）的 LXT971 芯片来实现这部分功能。数据处理模块、转发处理模块和 SAR 模块这三个模块都是使用 FPGA 设计实现的。在设计上，采用了 VHDL 语言设计与图形设计相结合的方法，灵活的运用了两者的优点；在芯片的选用上，采用了 ALTERA 公司的 APEX20K100EQC－2X 芯片。

在后面的章节中，将详细地论述各模块的硬件实现。

4 接口模块的硬件实现

4.1 前 言

网络在短短的几十年内获得了飞速的发展，但在这发展的几十年中，由于各种的原因，当今的网络世界是多种网络协议并立。要淘汰的网络技术虽已日暮西山，但由于以前投资的设备，不可能马上丢弃。同时，各通信设备供应商都有一套自己的协议，这样，现在 internet 上的协议仍是群雄纷争。如接入网就有 ADSL（Asymmetrical Digital Subscriber Line）、FDDI（Fiber Distributed Data Interface）、cabel modem、千兆以太网等。而作为连接不同网络之间的路由器也相应的提供若干类型的端口，以实现网与网之间的连接。

早在 60 年代，夏威夷大学研究的 ALOHA 系统是以太网的雏形，最初设计的传输速度只有 4800bps。经过 10M 以太网，到快速以太网、千兆以太网，再到现在的 10G 以太网，以太网已不再是纯粹的局域网了。在城域网，广域网方面，以太网的规范与标准正在不断的完善之中。以太网无疑是以后的发展趋势。在此，我们针对现在主流的快速以太网，做出了高速 MPLS 路由器的 DEMO 系统。

在 DEMO 系统中，数据收发模块负责以太网数据帧的收发，采用了 LEVEL ONE 公司（现已被 INTEL 公司收购）的 LXT971 芯片来实现这部分功能。数据处理模块、转发处理模块和 SAR 模块这三个模块都是使用 FPGA 设计实现的。在设计上，采用了 VHDL 语言设计与图形设计相结合的方法，灵活的运用了两者的优点；在芯片的选用上，采用了 ALTERA 公司的 APEX20K100EQC－2X 芯片。

4.2 数据收发模块的实现

数据收发模块完成对网络数据流的接收与发送，并在进行相应的光电转换与数据处理后，向数据处理模块提供 MII（Media Independent Interface）

数据流。数据收发模块可通过电或光接口与外部网络相连，其结构如图 4.1 所示。

数据收发模块主要完成的是物理层的适配：机械特性、电气特性、性能特性以及规程特性的适配，并将物理层的数据提取出来，经过适当的电平、码型变换后，上传给数据处理模块处理，同时，将来自数据处理模块的数据进行相应的变换后，传入外部以太网中。

以太网中，传输媒质通常有同轴电缆、双绞线、光纤。发展趋势是放弃使用同轴电缆(10Base5 与 10Base2)，双绞线是现在流行的介质选择，而随着千兆以太网、10G 以太网的普及，光纤也越来越受欢迎。在本路由器接口中，就提供了双绞线与光纤这两种接口。

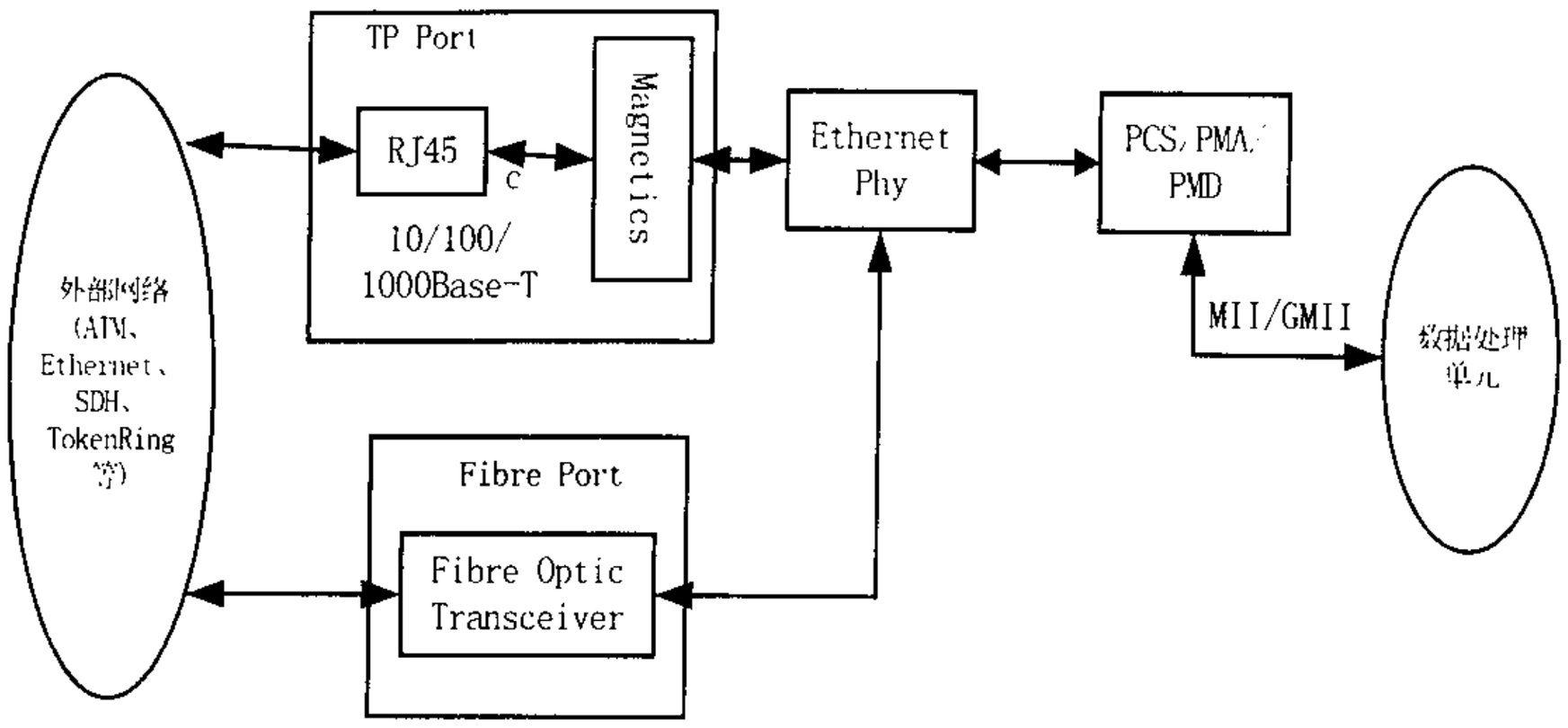


图 4.1　数据收发模块示意图

在传输基带数字信号时，为了更好的传输，都会对信号进行相应的编码。通常对不同的协议、传输速率或传输介质，采用不同的编码方法，以达到最佳的传输效果。因此，在使用双绞线时，10M 以太网使用的是曼彻斯特(Manchester)编码，而 100M 以太网是经过 4B/5B 变换后，使用 MLT3 编码。在使用光纤时，是经过 4B/5B 变换后，用 PECL 电平直接进行传输。

在这里，选取了 LEVEL ONE 公司的 LXT971 芯片作为 MPLS 路由器接口数据收发模块的以太网 PHY 层收发芯片。该芯片提供了一个 RJ45 接

口和一个 100Base－FX 的光纤接口。RJ45 接口同时支持 10Base－T 和 100Base－TX 的自适应网络。

使用 LXT971 实现接口数据收发模块，以 100Base－TX 为例，其图如图 4.2 所示。从双绞线上传来数据，经变压器隔离、转换后，送给 LXT971，在经过 LXT971 变换后形成 MII（Media Independent Interface）数据流 RXD[3..0]，上传给 MAC 层进行处理。同样，从 MAC 层传下的数据流 TXD[3..0]，经过逆变换后，送入以太网中。

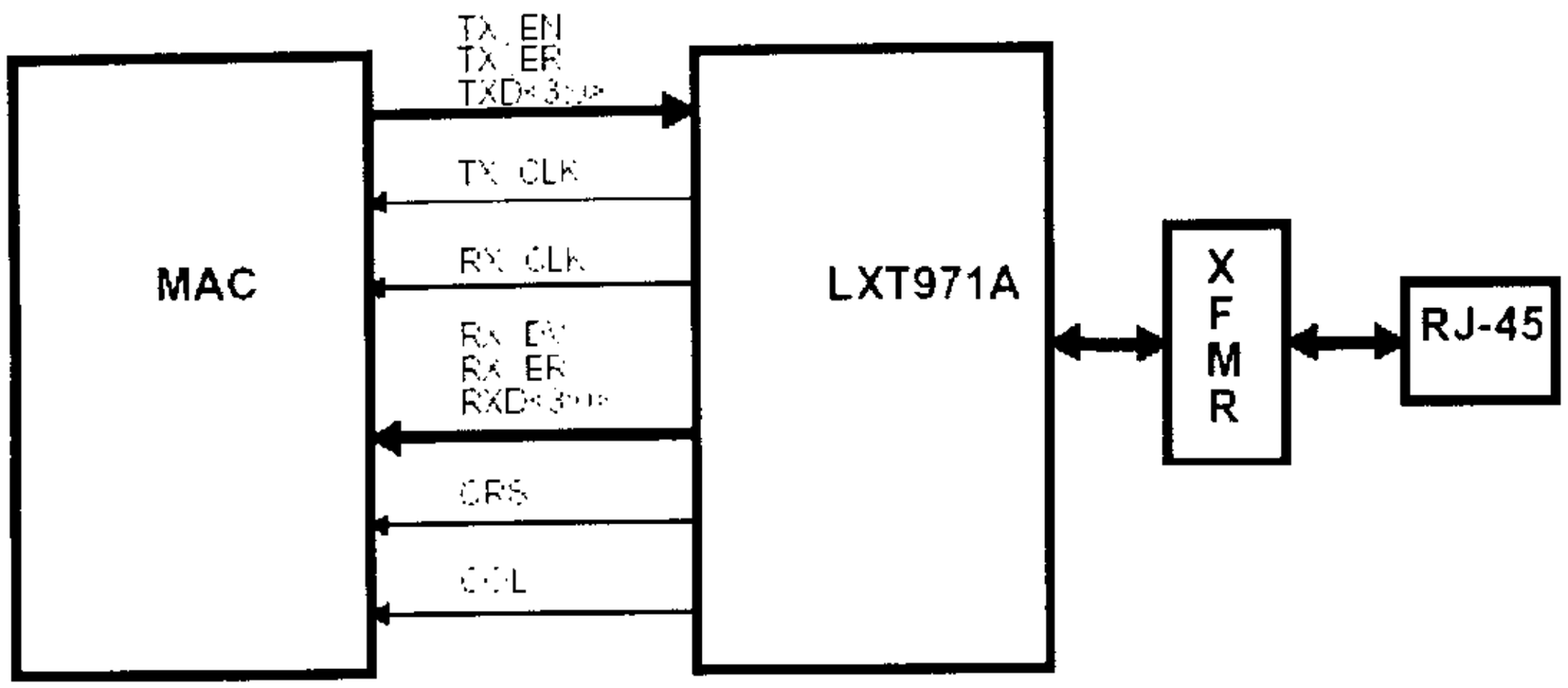


图 4.2　LXT971 接口图

图 4.2 清楚的显示了数据收发模块与数据处理模块之间的接口，也即是 LXT971 芯片与 MAC 层之间的 MII 接口。MII 接口有两个数据通道：发送数据通道 TXD 和接收数据通道 RXD。这两个数据通道都有自己的时钟线、数据线、错误控制信号线和使能控制信号线。它们分别是 TX_CLK、TXD[3..0]、TX_ER、TX_EN 和 RX_CLK、RXD[3..0]、RX_ER、RX_DV。剩下的两根信号线是 CRS（Carrier Sense）用于载波侦听、COL（Collision）用于冲突检测。

数据流经过 LXT971 的具体操作图 4.3 所示。网络数据进来，经过 MLT3 解码，然后解扰（DeScramble）、再进行 4B/5B 变换后，就形成了 MII 的串行数据流，经过串并转换，就是 MII 数据流了。而从 MAC 层过来的数

据进行相应的逆变换，就可送入以太网中传输。在这里，扰码的目的是为了扩展信号的功率谱和更有效的减小电磁干扰（EMI：Electromagnetic Interference）。扰码器使用一个 11bit 的数据不相关的生成式，自动的扰码和解扰。

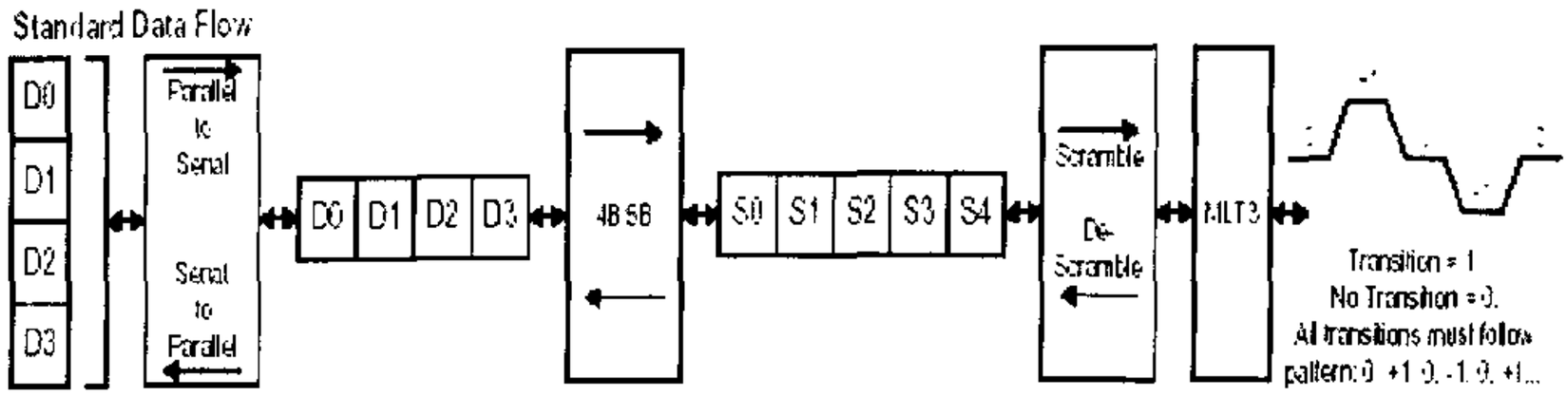


图 4.3 LXT971 数据通道

出于光能量不能为负，而且编码效率太低等原因，曼彻斯特码和差分曼彻斯特编码都不适用于光纤 LAN。所以引入了 4B/5B 码，因为它具有便于提取定时，低频分量小，可实时监测，迅速同步等优点。4B/5B 码的编码效率约为 80%。在 4B/5B 码型中，每 4 位二元输入信息被编码成一个 5 位二元输出码组。由于 4 位二元码组只有 16 种组合，而 5 位二元码组有 32 种组合，因此可以充分利用这种冗余度来实现线路传输码应当具有的性能。在 100M 以太网中，数据空闲时，IDLE 信号被连续地发送。流开始标志 SSD（Start－of－Stream Delimiter）被至于每个数据流的开始，J 标志和 K 标志总是成对出现的，K 标志总是紧跟在 J 标志的后面。而与此相同，流结束标志 ESD（End－of－Stream Delimiter）被至于每个数据流的结束处，T 标志和 R 标志总是成对出现的，R 标志总是紧跟在 T 标志的后面。

在双绞线方式下，LXT971 的网络接口需要一个 1:1 的变压器进行隔离。变压器隔离来自双绞线的静态电压，用以保护内部电路。通常，要求变压器所隔离的静态电压为 2KV。同时，变压器还可以滤除共模噪声，在 1～60MHZ，要求变压器的共模抑制比最小为 40dB；在 60～100MHZ，要求变压器的共模抑制比最小为 35dB。RJ45 接口的接线方式有两种，一种是交换机方式，一种是网卡方式。在这里，路由器接口是交换机方式。其

具体电路图如图 4.4 所示。在网络接口中，TPFINP 和 TPFIN 是差分输入端口，TPFOP 和 TPFON 是差分输出端口。

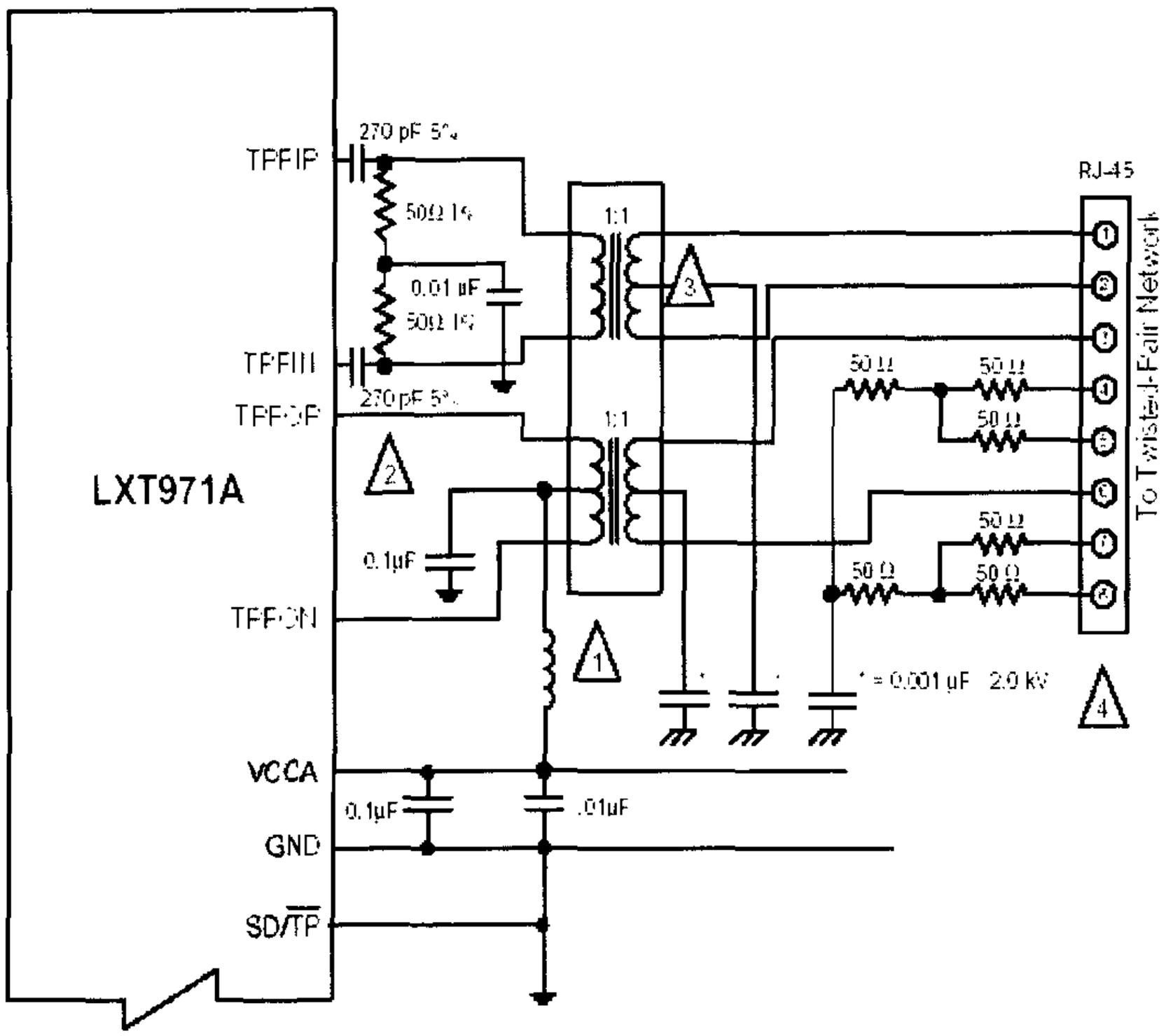


图 4.4 LXT971 网络接口电路图

4.3 现场可编程逻辑器件的选取

现场可编程器件可由用户配置，具有结构灵活、高密度、高性能、开发工具先进、开发成本低、质量稳定、可实时在线编程等特点。它的出现使许多数字系统可以采用微处理器、存储器和可编程器件这三类具有现场可编程特性的高容量器件组成，使数字系统的开发变得十分快速和灵活。

接口模块主要功能是由现场可编程逻辑器件实现的，为了选择最适当的器件进行设计，需首先进行器件的选型。接口模块在时序配合上要求较高、工作速度很快，综合考虑集成度、速率、开发工具等因素，选用了美国 ALTERA 公司的 APEX20K100EQC－2X 芯片进行硬件设计。开发工具为该公司的配套设计软件 QUARTUS II 2.0。

ALTERA 公司的 APEX20K100EQC－2X 芯片采用了 E^2PROM 编程技术，具有电可擦除功能，比普通的基于EPROM的FPGA（Field Programmable Gate Arrays）器件更方便。而且该系列芯片还具有在线可编程性 ISP（IN System Programmability），可通过串行下载电缆（Bit Blaster）或并行下载电缆（Byte Blaster）对已装配在印制板上的器件进行在线编程，省去了编程器、接插件，方便了系统的调试，而且还提高了可靠性。

一般，FPGA 设计周期由四个步骤组成：设计输入、设计实现、设计校验和芯片编程。设计输入的方法有：图形、文本、波形输入等多种。然后，借助于 QUARTUS II 2.0 开发软件完成设计实现和设计校验（包括功能仿真、定时仿真等）。最后，将产生的编程数据通过边界扫描接口下载到器件中。

4.4 数据处理模块

数据处理模块完成对网络接口协议——以太网 MAC（Media Access Control）层的处理，并提取出 IP 头交与转发处理模块。它包括 MAC 控制器和 IP 头处理器。

MAC 层是以太网协议的核心体现，具体说明可参见 IEEE802.3 协议。MAC 控制器完成对 IP 数据包的封装，并通过载波侦听多路访问/冲突检测（CSMA/CD）技术发送/接收。IP 数据包缓存在 fifo 中，同时由 IP 头处理器将 IP 头提取出来传给转发单元。

4.4.1 MAC 控制器

MAC 层的功能主要是完成数据的封装与介质访问的管理。CSMA/CD

的协议的主要功能都是在 MAC 层完成的。数据封装包括发送数据的封装与接收数据的拆装。在发送时完成封装操作，给帧信息装配帧同步的帧定界符，对地址段进行源和目的地址的编码，计算 CRC（Cyclic Redundancy Check）校验码序列。在接收时，去掉帧定界符，并对地址段进行判断是否与本站的地址相符合并对所接收的帧信息进行 CRC 校验，以确定接收是否正确。访问管理包括避免冲突的载波侦听判断、冲突发生后的处理、后退时间（back off time）的计算、冲突增强的人为干扰的发送控制与重发送控制等。而载波侦听信息和冲突信息则是由物理层提供的。

在接口板中，使用 VHDL 和图形的混合编程完成了 MAC 处理器的设计。可以将 MAC 处理器从功能上分成发送（TX）、接收（RX）和控制三个模块，如图 4.5 所示。在本设计中，将控制模块的功能分别放入 TX 模块和 RX 模块中，在后面不作单独详细说明。

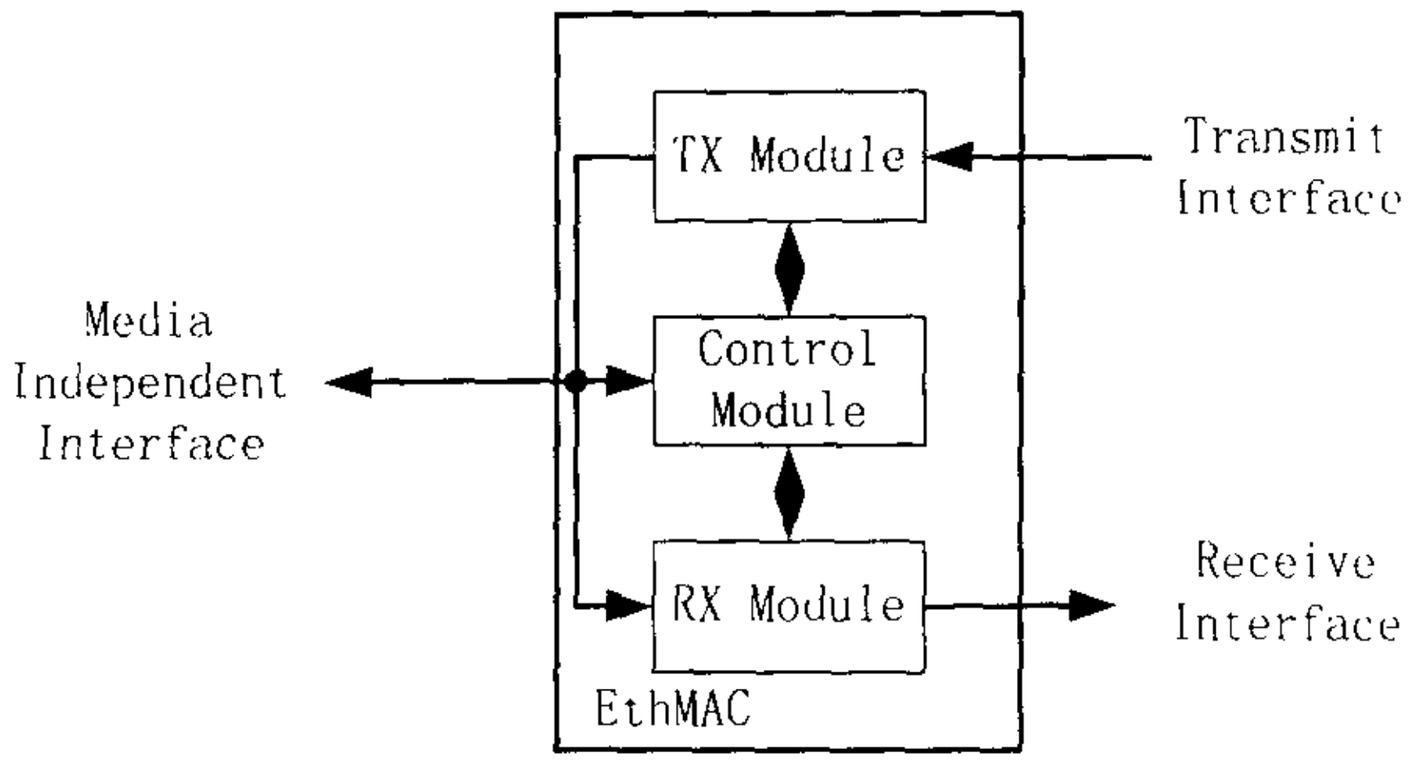


图 4.5　MAC 控制器设计原理图

1）　TX 模块

TX 模块负责将上层数据封装发送，同时实现避免冲突的载波侦听判断、冲突发生后的处理、后退时间的计算、冲突增强的人为干扰的发送控制与重发送控制等功能。

TX 模块的 VHDL 设计

TX 模块采用 VHDL 设计完成，整个设计程序过长（见附录），在这里只给出 TX 模块的实体说明：

```
library IEEE;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

ENTITY TXMac IS
  port(Byteclk,FrameEnd,Txdclk:in std_logic;
       Tx_valid: in std_logic;
       CRS,COL,FullDuplex:in std_logic;
        power_Rst: in std_logic; --the reset signal of the whole system.
Reset all ,when it='1';
        Txdata:in std_logic_vector(7 downto 0);
        DA_mac,SA_mac:in std_logic_vector(47 downto 0);
       TXD:out STD_LOGIC_VECTOR(3 downto 0);
        Tx_en,Tx_er:out std_logic);
END TXMac;
```

在设计过程中，从功能上将 Tx 模块划分成传输数据串并转换模块(TxData_P_S)、传输数据缓存模块（TxSync_FIFO）、冲突计数模块（TxCollision_Counter)、传输数据复用模块（TxData_MUX)、IFG 计时模块（IFG_Timer)、后退计时模块（BackOff_Timer)、随机数发生模块（Random_Number_Generator）、数据传输长度计数模块（TxLength_Couter)、CRC 校验码发生模块（TxCRC_Generator）和 Tx 状态机模块（Tx_State_Machine）十个模块。

传输数据串并转换模块(TxData_P_S)用来将 8 位的并行数据转换成 4 位并行数据。因为向下的 MII 数据接口所提供的接口是 4 位的半位元组（nibble)。当 Tx 状态机模块将 Transmit_Enable 信号置为有效时，传输数据串并转换模块开始工作。

传输数据缓存模块（TxSync_FIFO）用来存储要发送的完整的 MAC 数据帧。因为以太网并不是即到即发型的，要等网络空闲时才可以发送数据。同时，由于网络可能在传输时产生冲突，这时，如果冲突次数没有超过限定数值时，就需要重发。因此，需要传输数据缓存模块对 MAC 数据帧进行缓存。当 Tx 状态机模块将 Transmit_Enable 信号置为有效时，传输数据缓存模块开始工作，同时将 Tx_en 置为有效。如果 Transmit_err 有效时，Tx_er 也被置为有效。

数据传输长度计数模块（TxLength_Couter）计算已经向 MII 接口传输了的帧长度。当 Tx 状态机模块将 Transmit_Enable 信号置为有效时，数据传输长度计数模块开始工作，Transmit_Enable 信号无效时，数据传输长度计数模块将复位。

冲突计数模块（TxCollision_Counter）对 MAC 数据帧传输时所产生的冲突进行计数，如果冲突次数太多，超过了限定数值时，就通知 Tx 状态机模块将此 MAC 数据帧丢弃，准备发送下一数据帧，同时冲突计数模块复位，等待新的一帧发送。如果冲突次数没有超过限定数值时，就通知 Tx 状态机模块重发。如果接口工作在全双工模式下，冲突计数模块将忽略任何冲突信号。

传输数据复用模块（TxData_MUX）通过 Tx 状态机模块发送的数据选择信号 Data_select 对前导码、起始分界符、帧数据和帧校验码进行复用。复用后，整个数据帧缓存在传输数据缓存模块中等待传送。

IFG 计时模块（IFG_Timer）用来保证帧间间隔（IFG）为 96bit 数据间隔，或是 24 个发送时钟周期。如果接口工作在半双工模式下，IFG 计时模块将监听网络，从网络空闲开始对 IFG 计时。如果接口工作在全双工模式下，IFG 计时模块不再监听网络，IFG 计时从上一数据帧传输完毕开始。

随机数发生模块（Random_Number_Generator）将产生一个 10bit 的随机数。这个随机数产生的范围是 0 到（2^k-1）之间，K 的取值为所发生的冲突数和 10 两者之间的较小者。

后退计时模块（BackOff_Timer）决定后退延时数值。当冲突发生后，需要将数据帧进行重传。这时需要有一定的随机延时，来减小冲突的再次发生的概率。这个后退延时的大小由随机数发生模块产生的随机数决定。

CRC 校验码发生模块（TxCRC_Generator）产生的 CRC 校验码将被加在数据帧后面作为帧校验序列（FCS）。这个值是以从目的地址开始至 PAD 字段的所有字段的内容为函数进行计算的。发送和接收算法都使用循环冗余校验(CRC)来产生 FCS 字段的 CRC 值。该编码的生成多项式为 G(X)：

$$G(X)=x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^{8}+x^{7}+x^{5}+x^{4}+x^{2}+x^{1}+x^{0} \quad (4.1)$$

在发送过程中对每一帧信息产生一个 CRC 校验码，连同帧信息一起发送。在接收过程，对所接收的信息进行相同的 CRC 计算，并把计算的结果与接收的 CRC 码进行比较，以便发现传输中的错误，达到校验的目的。在这里，使用 8 位并行的 CRC 算法来计算 CRC。

Tx 状态机模块（Tx_State_Machine）的主要作用是控制传输进程。当上层有数据包要进行传输时，Tx_sof 信号被置位。状态机发出信号，传输数据复用模块、CRC 校验码发生模块和传输数据缓存模块开始工作，将数据包与前导码、起始分界符复用后形成 MAC 帧数据存储在传输数据缓存模块中。如果上层的数据包不足 46 字节，则添加 PAD 数据使其满足最小帧长度要求。CRC 校验码发生模块生成 CRC 校验码与此同时进行，当数据（包括 PAD 数据）全部到达传输数据复用模块时，CRC 校验码生成，CRC 校验码发生模块输出 CRC 校验码。传输数据复用模块将 CRC 校验码复用添加在数据后面作为 FCS。这样，MAC 帧就形成了。如果在这过程中，帧间间隔（IFG）满足同时网络空闲时，状态机发出信号通知传输数据串并转换模块、数据传输长度计数模块和冲突计数模块开始运作，MAC 数据帧开始传输。在此期间，可能会有冲突产生，Tx 状态机模块将作相应处理，同时随机数发生模块、后退计时模块将会有相应反应。

2） RX 模块

RX 模块负责将接收网络接口的数据包，然后对数据包进行有效性检查，如帧长度、目的地址匹配以及 CRC 校验等。如果是无效数据包，则抛弃，如果是有效数据包，则去掉 MAC 层封装，送到上层处理单元。

RX 模块的 FPGA 设计

Rx 模块的设计是采用图形设计与语言设计相结合的方法。图 4.6 是 Rx 模块的顶层设计的图形设计图。所采用的设计软件是 ALTERA 公司的 QUARTUS II 2.0。

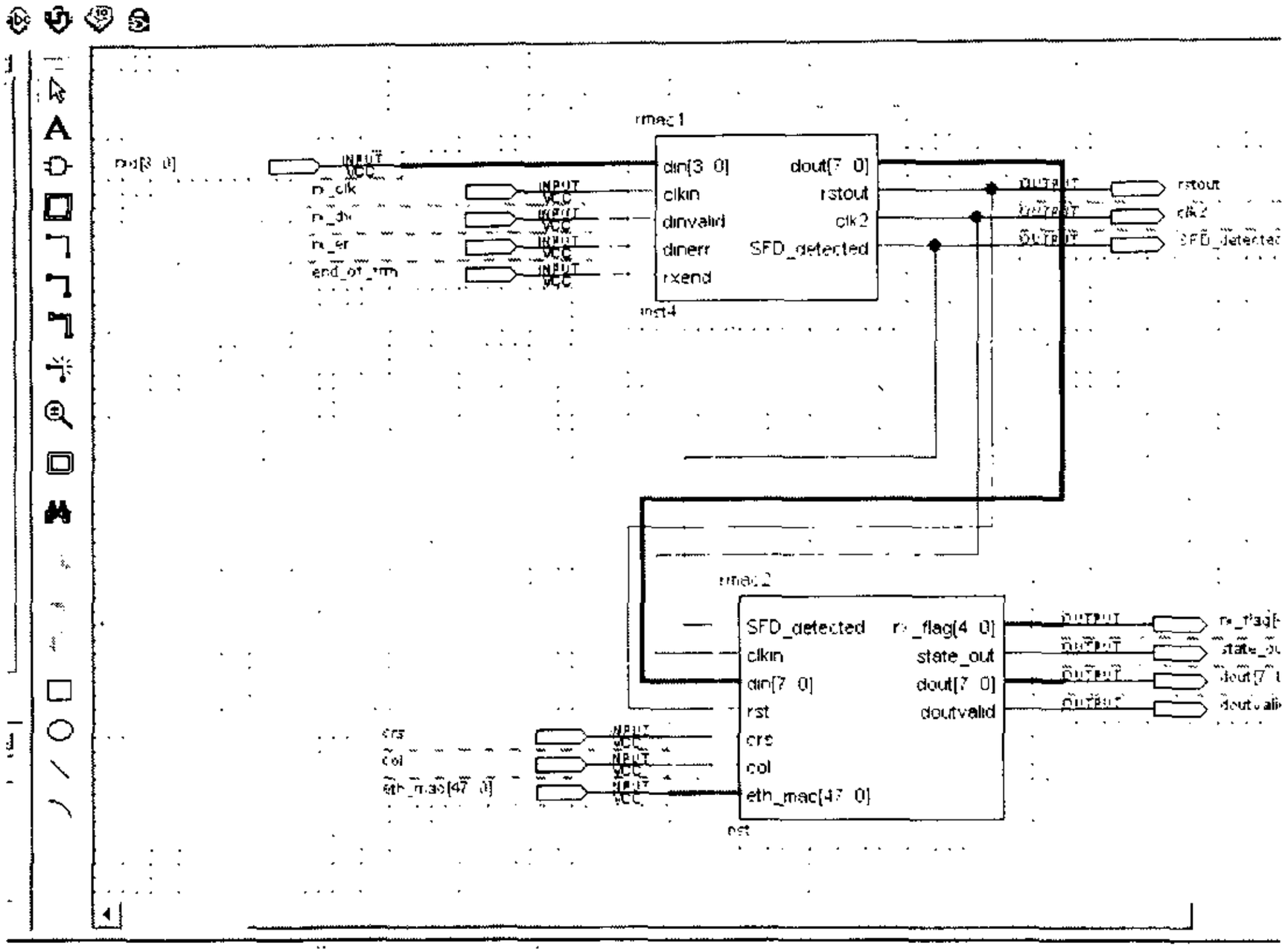


图 4.6 Rx 模块顶层设计图

图 4.6 中包含两个模块：rmac1 和 rmac2。其中 rmac1 模块负责对所接收的数据帧进行缓存，实现速率匹配。同时 rmac1 模块进行 SFD 探测，探测帧开始定界符，来确定数据帧的开始。当 SFD 被探测到时，SFD 探测模块将 SFD_detected 信号线置位，通知 rmac2 模块。在此同时，rmac1 模块进行串并转换，在高速设计中通常都是将高速的信号进行串并转换成多位并行处理，降低对处理速度的要求。这样，就能很好的在低速的处理器上完成高速的信号处理。在这里，所接收的 MII 数据都是 4 位的半元组（半个字节），为了方便后面处理，将数据都串并转换成 8 位并行数据。

rmac2 模块是实现 rx 模块功能的主要模块，它完成对输入数据包的有效性检查，其中包括帧长检测、目的 MAC 地址检测以及帧校验（FCS）检测。帧长检测，包括最小帧长度检测和 MAC 帧头的 LENGTH/TYPE 域的检测。目的 MAC 地址检测是对目的 MAC 地址进行分析，如果是广播地址或组播地址，给出状态信号。否则，将目的 MAC 地址与信号线 eth_address[47:0]上的本地 MAC 地址进行比较，如果不匹配，则表示此数据帧是无效数据帧，将数据帧丢弃。

帧校验检测是对数据帧进行循环冗余校验（CRC）。CRC 校验所采用的编码生成多项式与 Tx 模块中发送时是一样的，可参见式 4.1。对所接收的信息进行相同的 CRC 计算，并把计算的结果与接收的 CRC 码进行比较，以便发现传输中的错误，达到校验的目的。如果帧校验检测发现错误，则表示此数据帧是无效数据帧，将丢弃此数据帧。

4.4.2 IP 头处理器

IP 头处理器从 MAC 处理器中送过来的数据中提取出 IP 头，并将 IP 头送到转发单元进行转发处理。IP 头处理器的功能挺简单，实现也比较容易，在这里不再详细说明了。IP 头处理器的模块图可见图 4.7。

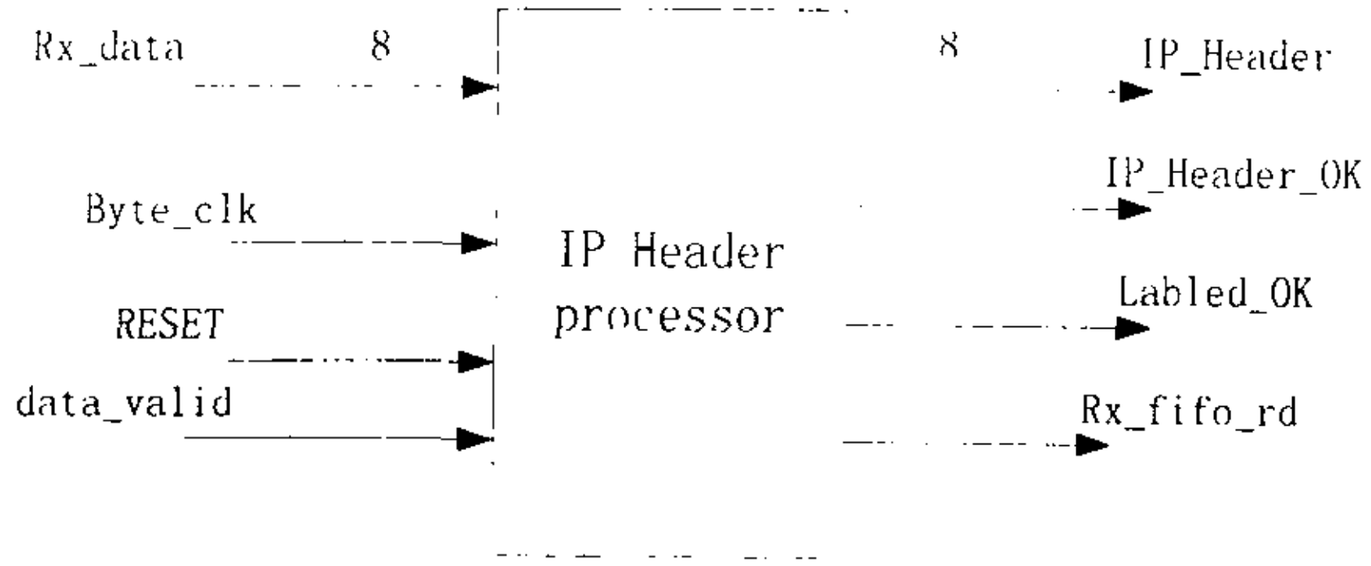


图 4.7 IP 头处理器

当 MAC 控制器将 data_valid 信号置为有效，说明已经收到有效地数据帧。IP 头处理器将 Rx_fifo_rd 信号置为有效，开始从 MAC 控制器中的 FIFO 中读取 IP 头。由于进入接口模块的数据帧可能是从 MPLS 网络中过来的加了 MPLS 标签的 IP 数据帧，也有可能是从外部网络过来的普通 IP 数据帧。

因此，要对 IP 头进行相应的分析来分别是否加上了 MPLS 标签。如果是加了 MPLS 标签的 MPLS 数据帧，IP 头处理器将 Labled_OK 信号置为有效，否则，将 IP_Header_OK 信号置为有效。IP 头数据通过 IP_Header 信号线传送给转发单元。

4.5 转发处理模块

MPLS 的转发处理模块使用的交换/转发算法是基于标签技术的，其工作方式为：当一个 LSR 接收到一个分组时，路由器提取出分组头中的标签，然后利用该标签(入口标签)作为标签转发表的一个索引，查找转发表，获得相应表目(包括输出接口，出口标签及对其的操作如 Pop,Swap,Push 等，某些情况下如考虑 QoS 还可能包括输出队列)，对标签进行相应的操作后，把出口标签填入分组，然后将标签交换过的分组放入相应的输出接口/输出队列。

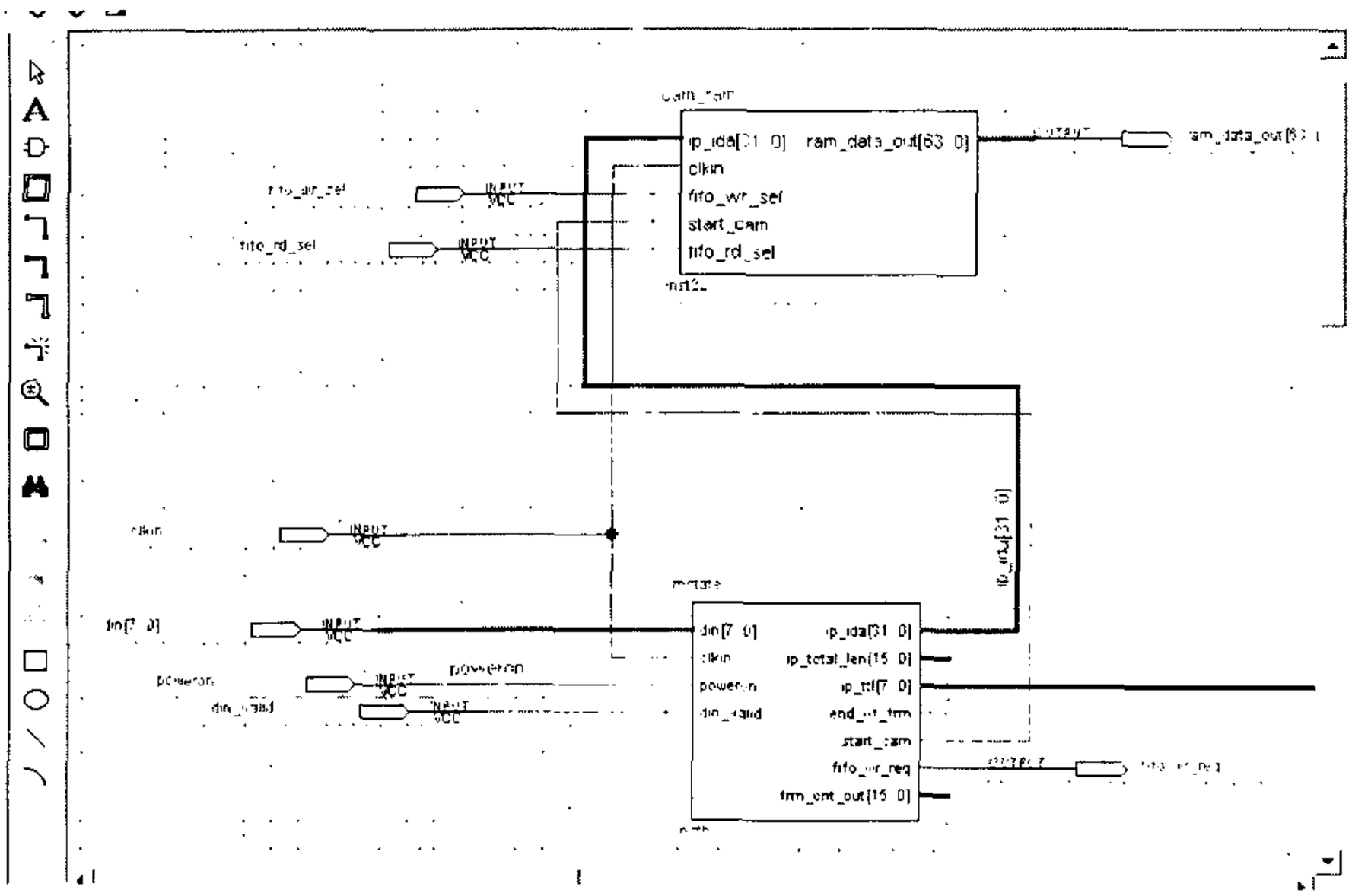


图 4.8 转发处理模块

转发处理模块是路由器接口模块的核心部分，也是整个路由器的核心

部分之一。它接收来自 IP 头处理器的信息，对 IP 数据头进行处理，通过数据帧的 FEC 分类、标签的判断、查找、交换以及绑定等操作，完成数据帧的转发。

转发处理模块由 FPGA 实现，图 4.8 是转发处理模块的顶层设计图。转发处理模块在设计上分成 mstate 模块和 cam_ram 模块。mstate 模块接收来自 IP 头处理器的信息，完成 IP 数据头检测与处理、数据帧的 FEC 分类、标签的判断以及绑定等操作。

mstate 模块在接受到来自数据处理模块的 IP 数据头后，要对 IP 数据头进行检测，对 IP 数据头的校验主要包括：

1) 验证协议版本的一致，Version 域和 Protocol 域一起指定网络层协议的版本；

2) IP 数据头的长度域 Hlen 必须≥5； 即 IP 数据头要不小于 20 字节

3) 域 Total Length 表示包括 IP 数据头在内的 IP 分组的总长度，该值必须≥域 Hlen 指定的值；

4) 域 Total Length 值不能超过 IP 协议协商指定的最大传输单元 MTU；

5) 对 IP 数据头的“头校验和”必须和 Header Checksum 一致；IP 数据头校验和的算法很简单：设“头校验和”初值为零，然后对头标数据每 16 位求异或，结果取反，便得到校验和。

如果校验 1、2、3、5 的任何一个未通过，那么就简单地丢弃该分组；

如果通过这些校验，则进行生存时间域 Time To Live(TTL)的判断，

a) 如果 TTL>1，那么路由器将 TTL 减 1，并更新 IP 数据头的校验和域 Header Checksum；

b) 否则，发出控制管理消息 ICMP 以通知发送方，同时简单地丢弃该分组。

如果未通过校验 4，即 IP 分组总长度>MTU，那么路由器将根据 Flags 域(不分片标志域，DF)决定是否将分组分为几个小片进行传输。

如果输入的 IP 数据头中包含了标签，那么提取该标签，对标签进行 TTL 检测，如果是有效标签，则送入 CAM/RAM 模块进行标签查找。同时

判断此 IP 数据头所绑定的标签是否是一个标签栈。

标签是 MPLS 网络中的一项核心技术。它是一个简短的，具有固定长度的，具有本地意义的标识符，它用以识别转发等价类 FEC。MPLS 的许多优点都直接或者间接地来自于标签的使用。在 MPLS 网中进行分组转发的过程实际上就是标签交换操作的过程。

MPLS 技术实际上就是围绕标签的一系列的操作。因此，如果输入的 IP 数据头是普通的 IP 数据头，其中不包含标签，就需要为此 IP 数据头绑定一个标签。MPLS 技术提供了 FEC（Forwarding Equivalent Class：转发等价类）与标签的映射。为了获得相应的标签映射，转发处理模块要对输入的 IP 数据头进行转发等价类（FEC）的分类。

转发等价类（Forwarding Equivalent Class）是 MPLS 中最重要的一个概念，甚至可以说是 MPLS 技术的基础。

在 MPLS 网络中，当分组到达 MPLS 网络入口时，它将按一定的规则被划归为不同的子集。从转发的角度来说，每个子集中的分组都由路由器以相同的方式来处理（比如它们都被发送到相同的下一跳），即使就这些分组的网络层头中的信息来说这些子集中的分组彼此互不相同。我们称这种子集为转发等价类（FEC）。路由器对属于一给定的 FEC 的所有分组以相同的方式转发，其原因是分组内的网络层头所携带的信息和转发表中的表目之间的映射是多到一的映射（一到一的映射是特殊的例子）。也就是说，网络层头中内容不同的分组可以映射到转发表中相同的表目内，而在此转发表中，每个表目都描述一类特定的 FEC。最简单的 FEC 划分依据是目的地址，除此之外，还可考虑 IP 分组的业务类型 TOS、源地址等。FIB 的检索关键项是 FEC 标识。在这里，所采用的 FEC 划分依据就是目的 IP 地址。

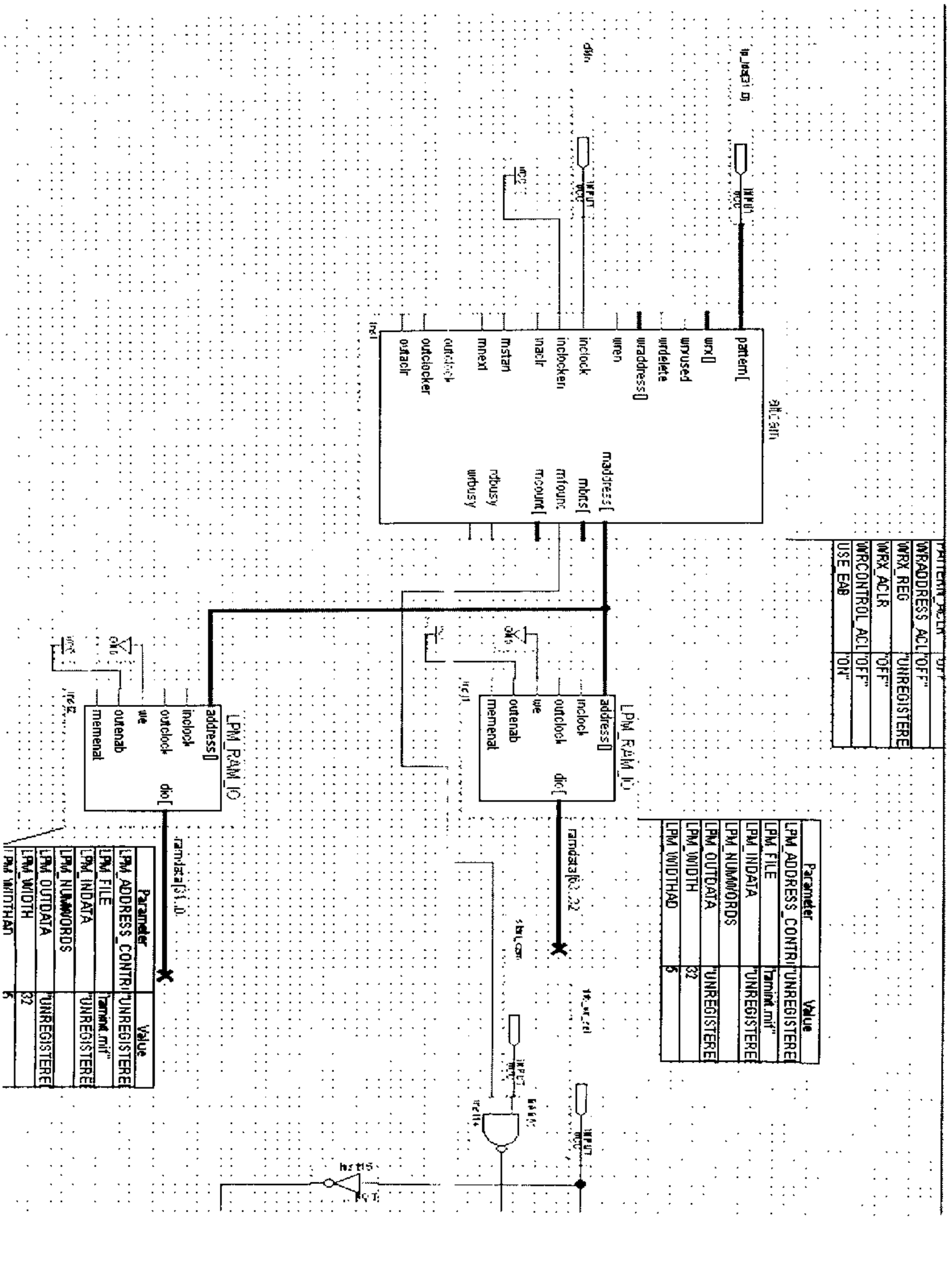
通常用转包率（表示每秒转发引擎处理的 IP 包个数）来衡量转发引擎的性能。由于转发引擎往往采用大规模高速 FPGA 实现，因此 IP 包的合法性检查以及校验和的计算等较易实现，影响转包率的主要因素为线速率查表的时间。线速率查表是高速骨干路由器中转发引擎的一项关键技术。

目前传统 IP 路由器中采用的路由器查表算法是最长前缀匹配算法。采用最长前缀匹配算法时，每次查找都必须将整个转发表至少遍历一次。当在骨干网中，若转发表庞大时，这将造成大的查找时延。采用 MPLS 技术，

使用标签进行高速全定长查找(容易用硬件实现)，避免了因低速的可变长度路由查找产生的瓶颈。

采用硬件查找是提高转发表查找速率的另一项有效的技术：对于较小的路由表可采用高速缓存方式，能有效地提高查表速度。常用的基于硬件的技术是用相联存储器 CAM (content- addressable memories)和高速缓存来提高查找速度，具体做法是将组合电路与存储器集成在一起，形成智能存储器。

本 DEMO 系统中，由于路由表较小，并且不考虑路由表的更新，因此，采用相联存储器 CAM 和高速 RAM 的配合使用，完成高速查找。在转发处理模块中，cam_ram 模块完成对转发表的查找。图 4.9 为 cam_ram 模块的设计图。


PATTERN_ACLR
WRADDRESS_ACL
WRX_REG
WRX_ACLR
WRCONTROL_ACL
USE_EAB
"OFF"
"OFF"
"UNREGISTERE
"OFF"
"ON"
pattern[
wrx[]
wrused
wrdelete
wraddress[]
wren
inclock
inclocken
inaclr
mstart
mnext
outclock
outclocken
outaclr
maddress[
mbits[
mfound
mcount[
rdbusy
wrbusy
inst
LPM_RAM_IO
address[]
inclock
outclock
we
outenab
memenal
dio[
LPM_RAM_IO
address[]
inclock
outclock
we
outenab
memenal
dio[
ramdata[63..32
ramdata[31..0]
Parameter
Value
LPM_ADDRESS_CONTRI
"UNREGISTERE
LPM_FILE
"ramint.mif"
LPM_INDATA
"UNREGISTERE
LPM_NUMWORDS
LPM_OUTDATA
"UNREGISTERE
LPM_WIDTH
32
LPM_WIDTHAD
6

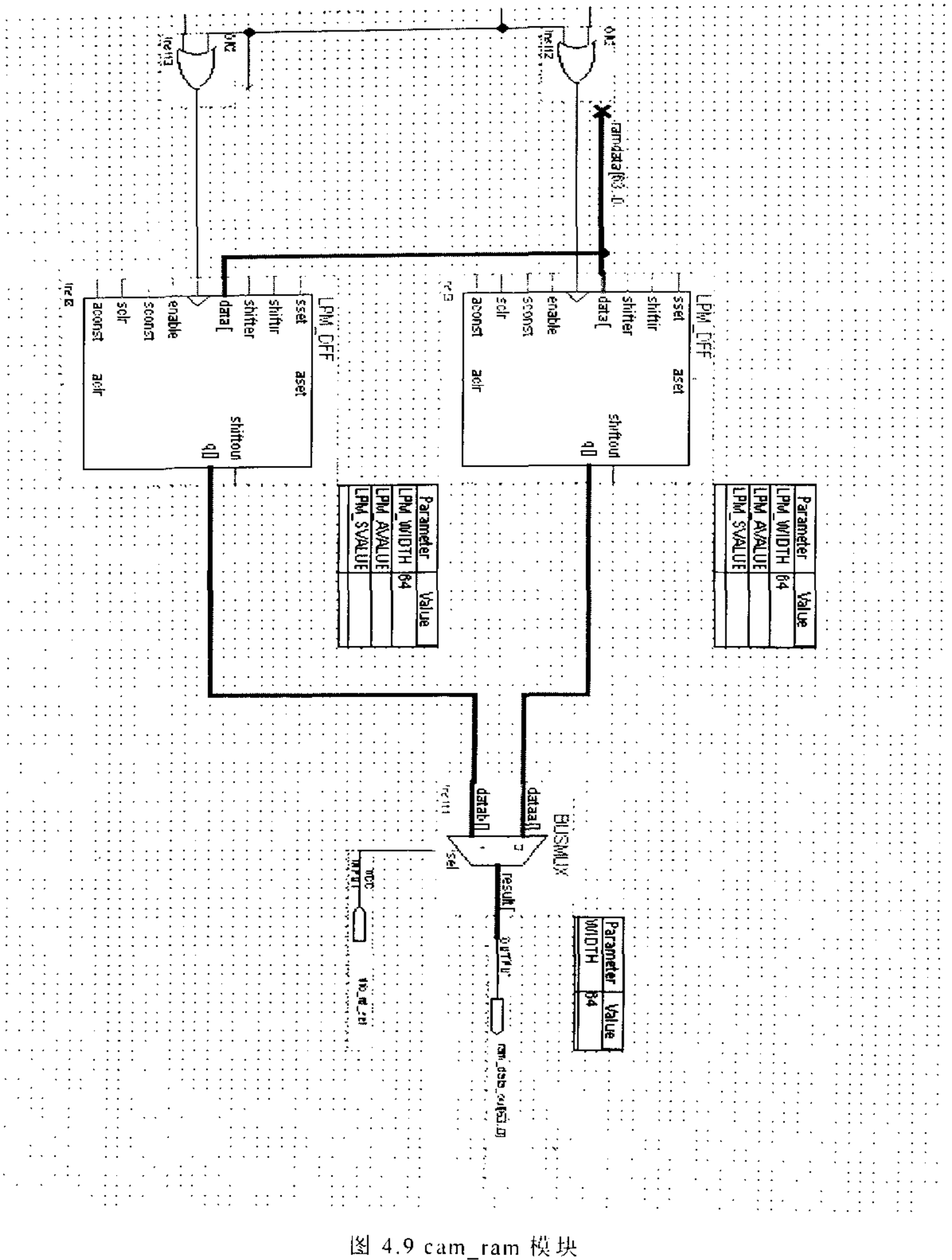
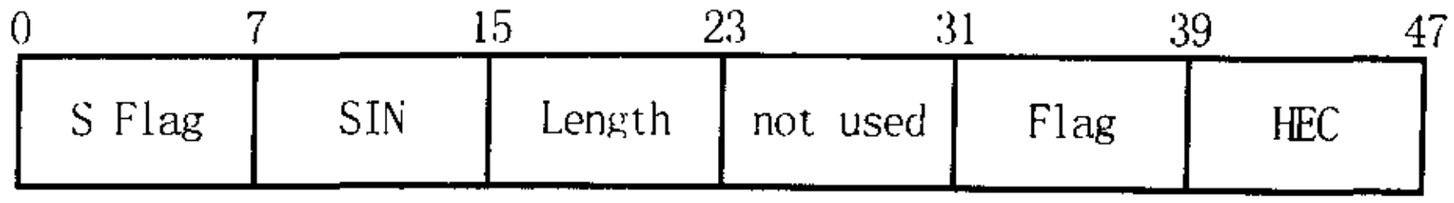


图 4.9 cam_ram 模块

4.6 SAR 模块

在路由器中，输入交换结构的数据格式有两种：一种是不定长数据格式。IP 数据帧是不定长的数据帧，在传统的 IP 路由器中，基本上都采用这种数据格式。另外一种是定长数据格式，现在的高速路由器中，都采用这种方式。在这里，我们所设计的 MPLS 路由器也是采用定长数据格式。因此，在数据进入交换结构之前，要将数据分割成内部交换信元，同样，当内部交换信元通过交换后，要将其组装还原。本章我们将对 SAR（分割与组装）模块的具体设计进行详细说明。

由于以太网数据帧的长度是可变的，最小 46 个字节，最长 1500 个字节。若让这样可变长度的以太网数据帧进入交换模块，将使调度器（scheduler）的设计变得很复杂。因为可变长度的数据包可能随时结束，调度器必须要时刻监视所有端口，看哪些输出端口空闲，且一旦某输出端口空闲，调度器就必须选择一个新的输入与之相连接。如果一个空闲的输入端口有几个包需要到不同的输出端口，调度器是让该端口等待一个忙输出端口变为空闲，还是立即选择一个空闲的输出端口呢？如果不等，则有可能是别的端口得不到处理；若等，则浪费了交换模块的带宽（经证明，这种浪费几乎达到了整个系统带宽的一半）。因此，选择一个定长的数据包是提高路由器效率的一种有效途径。本路由器定义了一种内部交换信元格式作为交换信元，来实现等长交换。同时，因为选用了定长的数据包来交换，调度器的设计大大简化，可以通过硬件来完成，在本路由器中采用 FPGA 来完成调度器的设计。在这里，所采用的内部交换信元类是与 ATM 信元。信元数据都是 48 个字节，不过信元头不是象 ATM 信元那样 5 个字节，而是 6 个字节。同时，抛弃了复杂的 ATM 信令。内部交换信元头的格式如图 4.10 所示。



S Flag: Switch Flag 交换标志

SIN: Source Interface Number 源接口号

Length: 信元数据区数据长度

not used: 未用

Flag: 标志

HEC: Header Checksum 头校验和

图 4.10 内部交换信元头格式

在内部交换信元头中，第一个字节 S Flag 是交换模块进行交换寻径控制使用的标志，如果为全零，表示此内部交换信元为空信元。第二个字节 SIN 是内部交换信元的源接口号——即输入接口号。因为在交换过程中对从同一个输入接口来的信元都是顺序交换的，因此，用源接口号能很好的区分同时到达输出接口的内部交换信元。第三个字节 Length 表示信元中的有效数据长度。当内部交换信元不是数据帧的结束信元时，这个域值都是 48。如果是结束信元，域值是除去填充后的有效数据长度。第四个字节暂时未用。第五个字节 Flag 是控制信息，它包括一些控制标志，其中包括结束标志，它用来标注信元是否是结束信元。第五个字节 HEC 是头校验和，用于保证头标数据的完整性。这里头校验和算法与 IP 数据头的头校验和算法一样：设“头校验和”初值为零，然后对头标数据每 16 位求异或，结果取反，便得到校验和。

SAR（Segmentation And Reassembly：分割与组装）模块具体框图如图 4.11 所示，它包括分割模块和组装模块两部分。

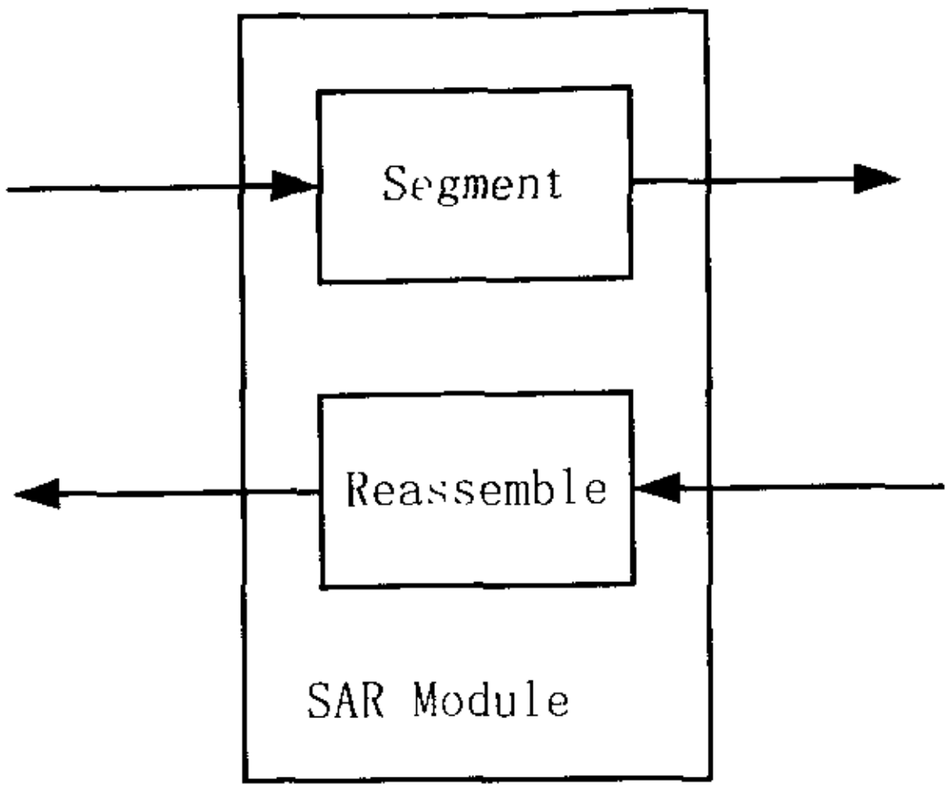


图 4.11 SAR 处理模块具体框图

分割模块

在交换模块中采用定长交换，因此，在数据帧送入交换模块之前，都要将数据帧分割成定长的内部交换信元。分割的方法及内部交换信元的格式如图 4.12 所示。

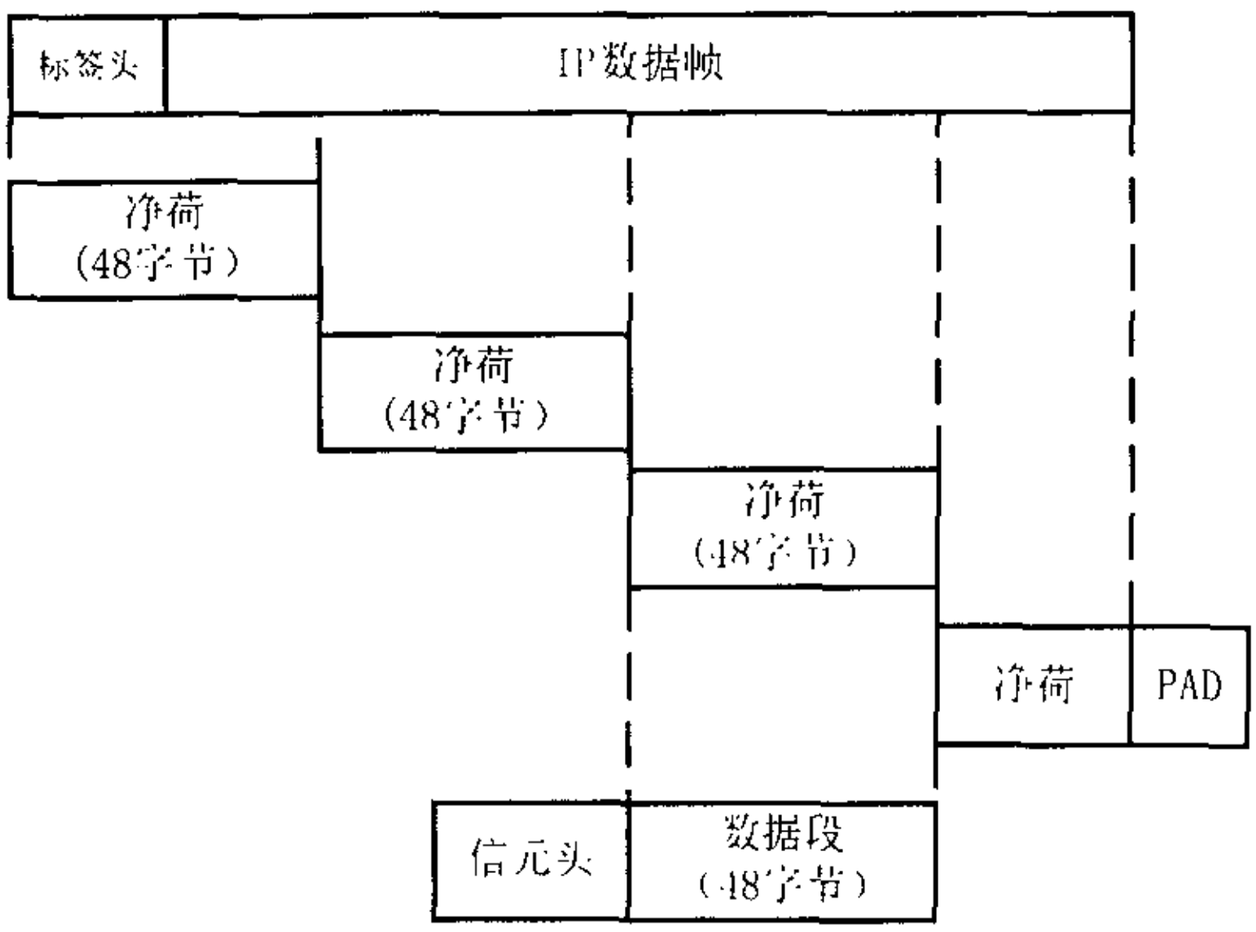


图 4.12 分割方法及内部交换信元格式

内部交换信元的长度为 54 字节，类似与 ATM 信元，信元数据也是 48 字节长。不过不同于 ATM 信元，内部交换信元头为 6 字节，而不是 5 字节。内部交换信元头中包含着数据帧分割和组装的信息。内部交换信元头的格式见图 4.6。分割模块采用 VHDL 设计完成，下面是分割模块的实体说明：

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
LIBRARY altera_mf;
USE altera_mf.altera_mf_components.all;

entity tcell is
port
(poweron:in std_logic;
 fifo_rdusedw:in std_logic_vector(7 downto 0);
 fifo_data_in:in std_logic_vector(7 downto 0);
 ram_data_in:in std_logic_vector(63 downto 0);
 fifo_wr_sel:in std_logic;
 fifo_rd_sel:in std_logic;
 fifo_wr_end:in std_logic;
 fifo_rd_req:out std_logic;
 cellclk_byte_syn:in std_logic;
 cellclk_frm:in std_logic;
 celldata:out std_logic_vector(7 downto 0)
);
end tcell;
```

poweron 是个全局复位信号，高电平有效。fifo_rdusedw 表示 IP 数据包缓存中缓存的字节数。经过标签绑定的 IP 数据包从 fifo_data_in 信号线传递给分割模块，在分割模块中被分割成内部信元流，从 celldata 信号线输出，经并串转换后，传给交换模块进行处理。ram_data_in 是转发处理模块查表所得结果，分割模块根据这信号来创建内部信元头。

组装模块

数据帧被分割模块分割成内部交换信元流，传送到交换模块。经过交换模块交换后，要将内部交换信元流组装还原成数据帧。内部交换信元的组装是根据内部交换信元头中的组装标志完成的。组装模块采用 VHDL 设计完成，下面是组装模块的实体说明：

```
library IEEE;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

ENTITY atmcell IS
  port(Byteclk,Fclk,aclr:in std_logic;
       power_Rst: in std_logic; --the reset signal of the whole system.
                                Reset all when it='1';
       Atmcellin:in std_logic_vector(7 downto 0);
       AccessNum:in std_logic_vector(1 downto 0);      --decide which
access plan
       Fifo_count_in:out std_logic_vector(15 downto 0); --this include
DataCount and FifoSelect
       RamWren1,RamWren2,RamWren3,FrameEnd:out std_logic;
       AtmCell_out:buffer STD_LOGIC_VECTOR(7 downto 0)
);
END atmcell;
```

组装模块通过 Atmcellin 信号线接收来自交换模块的内部信元流，根据内部信元头的组装标志，进行组装。组装完成后，由 AtmCell_out 信号线送出。

4.7 接口控制模块

接口控制模块主要完成对转发表的刷新与维护，同时处理相关 IP 控制信息。

在 MPLS 网络中，数据的转发实际上就是标签的交换。作为标签的映射关系表——转发表，更是转发过程中的关键。由于外部网络的拓扑的变化是不可预知的，转发表中的数据表项不可能是一成不变。映射关系的有效性，决定着数据的路由转发是否成功。例如，路由器 A 到路由器 B 之间的网络路径因某种原因断了，而路由器 A 中的转发表没有相应的更新。而这时候，如果有数据帧到达路由器 A，要转发到路由器 B 中。由于路由器 A 中的转发表没有相应的更新，仍然按已无效的路径进行转发而导致转发失败。

在我们设计的高速路由器中，路由器的路由管理由主控制模块完成。接口模块中的转发表只是主路由表的一个局部高速缓存。由接口控制模块完成对转发表的刷新与维护。这个过程包括两项基本活动，即路由的更新(update)和废弃(flushing)。

更新——转发表在两种情况下进行更新操作。一种是当主路由表刷新的同时，主控制器模块向各个接口广播刷新消息。接口控制模块接收到刷新消息后，对转发表进行更新。第二种是当一个分组在接口模块的转发表中没有找到相应的转发表项时，接口控制模块将分组头传送给主控制模块，在主路由表中进行查找。如果，查找到相应的转发表项，则将其返回接口模块。此时，更新转发表。

废弃——该术语既代表了一种定时器，又代表了去除路由表中某个路由的处理动作。由于网络外部的 IP 数据流的随机性很大，是突发性的。CAM 的容量是不可能做的很大的。在转发过程中，有些转发表项可能在一段时间内会被经常查找，有些转发表项则长时间不会被查找。而表项数目是有限的，就有可能会出现这样的情况。当数据帧到达接口后，在转发表中没有查找到相应的转发项，而在主路由表中找到相应表项，可此时，CAM 中已经满了，无法再添加。这样，就只有频繁的查找主路由表，降低了转发效率，可能造成拥塞。为了解决这个问题，我们可以象主路由表中一样，设置废弃定时器。例如，如果废弃定时器的设定时间为 60 秒，只要废弃定时器超过 60 秒，则相应的转发表项将被即刻从转发表中删除。

除了刷新和维护转发表外，接口控制模块还处理相关 IP 控制信息。

可以看到接口控制模块从数据处理模块和转发处理模块中取得相应控

制信息：

a 该信息可能是 IP 网络控制流。如 ARP 报文，边缘路由器一般都需要在接口中实现 ARP。

b 若是 MPLS 网络管理信息，接口控制模块对此信息进行分析，并将其转送给主控制模块。

c 若是路由消息，接口控制模块直接将此消息转送给主控制模块。

在我们设计的路由器中，因为只是 DEMO 系统，在相应的地方作了一定的简化。我们在实验中所使用的路由表，是静态路由。因此，对于接口控制模块来说，对转发表的刷新和维护就是不必要的。故我们暂时没有对这部分进行设计。不过，在以后的工作中，这一部分是必不可少的。

4.8 实验测试

在实验中，接口模块所接的外部网络是百兆以太网，外部输入数据由计算机程序产生。接口模块从百兆以太网中接收数据，经过数据收发模块、数据处理模块、转发处理模块、接口控制模块以及 SAR 模块处理后，将生成的内部交换信元流送入交换模块。下图是接口板的实物照片。图 4.14 中所示的波形是内部交换信元流的串行信号。

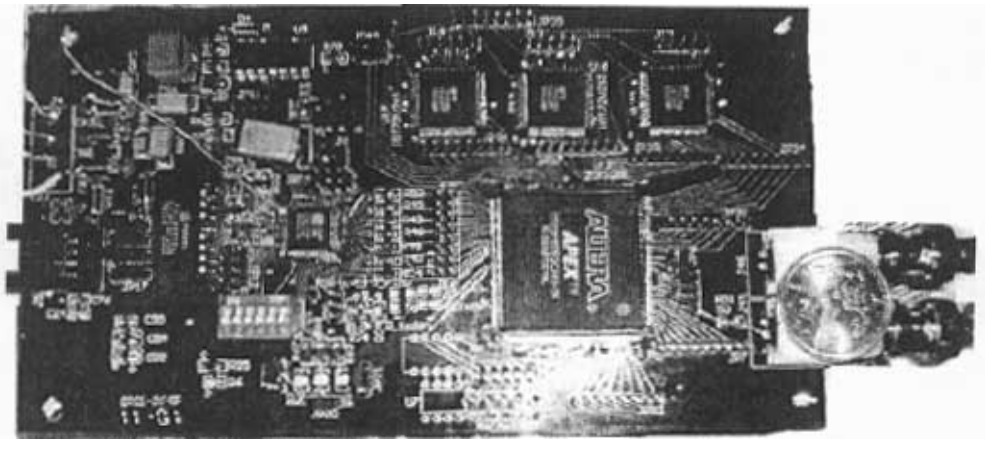


图 4.13 接口板实物照片

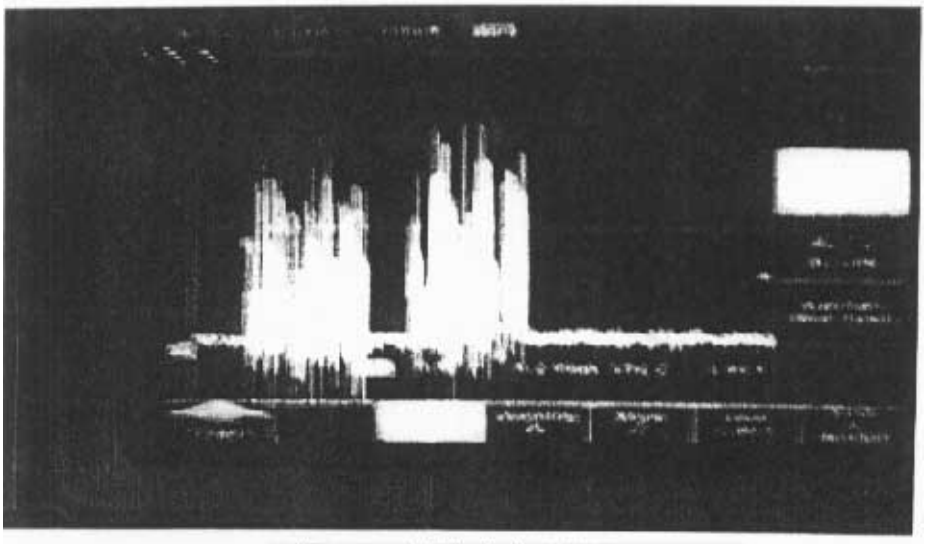
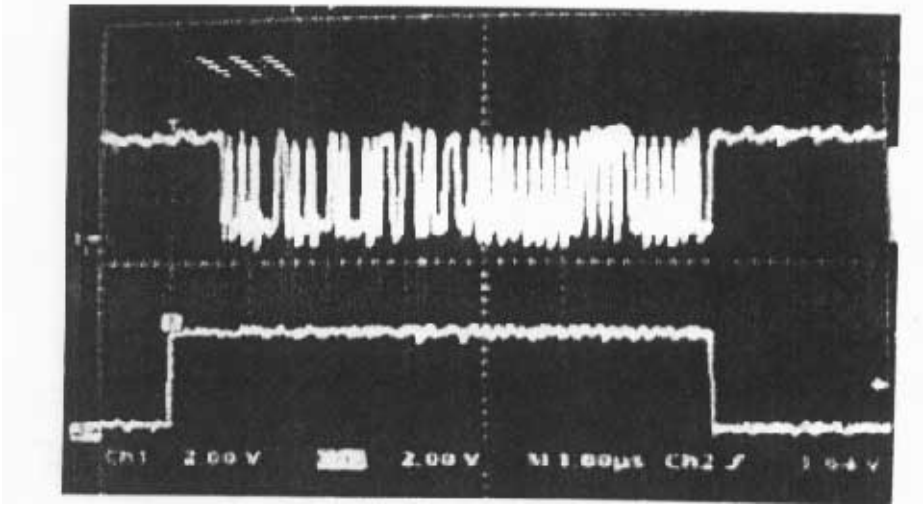
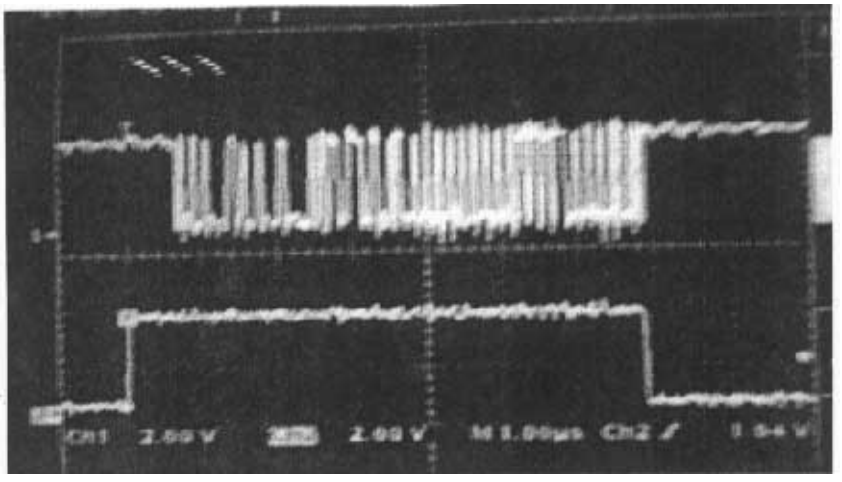


图 4.14 内部交换信元信号



(a) 接收信号



（b）外部自环发送信号

图 4.15 外部自环信号

由于交换模块还没有开发完成，因此，我们无法进行整机调试。因此，将接口板外部自环，对接口模块进行了相应的一致性测试和功能测试。图 4.15 是外部自环时所测的波形图。4.15(a)是从百兆以太网中传送过来的信号，经过数据收发模块处理后的接收信号，通道一是 RXD[0]信号，通道二是 RX_DV 信号。图 4.15(b)是接口模块外部自环后，准备发送到百兆以太网中的信号，通道一是 TXD[0]信号，通道二是 TX_DV 信号。

图 4.14 是接口模块对接收信号处理后，要传送到交换模块的串行内部信元信号，而交换模块不对信元进行任何处理，只是提供一个数据交换通道，因此，可以通过外部自环来模拟交换模块。接口模块接收到来自交换模块的信元流后，经过处理还原成网络信号，由图 4.15 中可以看出，接口模块所接收的信号和发送的信号完全一致，可以证明接口模块的在外部自环实验中，成功的完成了接口模块的处理。

同时，我们采用软件（所采用的软件是 LANEXPLOER V3.7 FOR WINDOWS2000/NT/95/98）抓取相应的包进行分析，如图 4.16 所示。

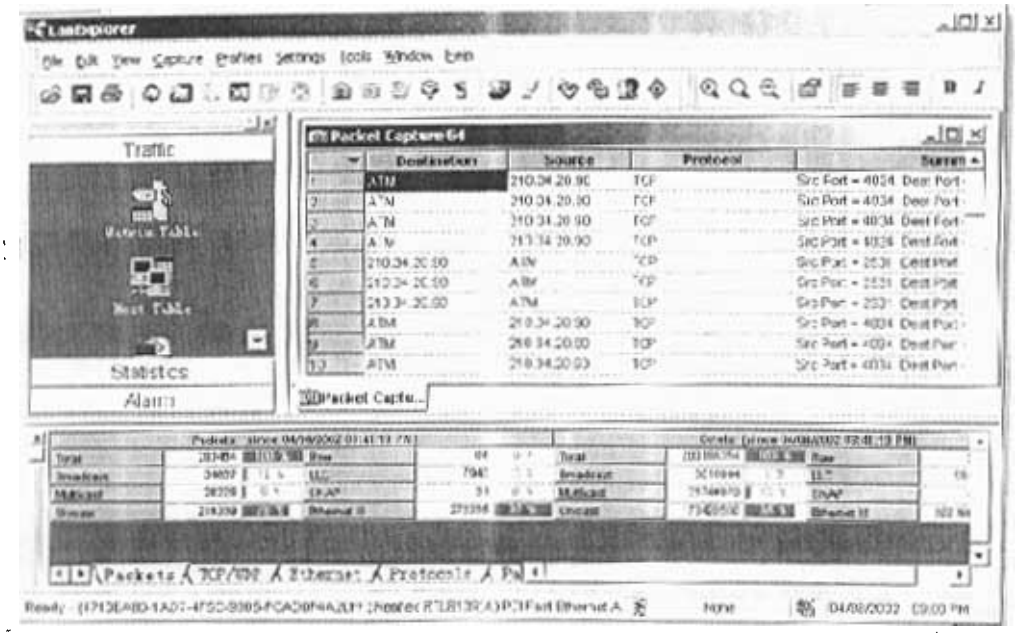


图 4.16 包的抓取

对端口吞吐量测试、丢包率测试中取得了理想的效果。其中，丢包率测试要求测出在满线速不同帧大小情况下的单端口丢包率。在百兆以太网中，以 64、128、256、512、1024、1280、1518Byte 的不同大小的帧进行测试，通过 LANEXPLOER 软件抓取相应的包进行分析，发现没有包丢失。由此，我们可以得出相应的结果：接口模块在线速处理下，丢包率很小，所设计的路由器的丢包率取决于后面的处理模块——交换模块的丢包率。

4.9 本章小结

将路由引擎 (routing engine)和转发引擎 (forwarding engine)分开，将局部转发表从全局路由表中独立出来，是现代高速路由器的设计的一个基本思路。

而我们提出的 MPLS 交换结构采用分布控制的策略，在每一个输入接

口中都有一个本地转发处理模块来执行标签查找和转发判决。而 MPLS 技术中只有一种转发算法——在标签转发技术基础上的算法，这是标签交换和传统路由体系结构之间的一个重要区别。这种设计使得 MPLS 转发处理模块的设计也相应简单了不少，可以用硬件实现。

本章对接口模块的数据处理模块、转发处理模块、SAR 模块以及接口控制模块进行了 FPGA 设计。

现代高速路由器的设计中，都采用定长交换来提高交换效率，同时也简化了调度器的设计。因此，SAR 模块成为了现代高速路由器设计中的一个必要部分。SAR 模块本来打算采用相应芯片(如 transwitch 公司的 SARA 系列)，但正如 Cisco,Juniper 等公司指出的，对于高速线卡而言，SAR 芯片是个瓶颈(目前比较高的是 622Mbps 的处理速度，2.5Gbps 的还比较少)，为了能达到 10Mpps 左右的线速转发，目前看来不适合使用 SAR 芯片，而且，在设计中，SAR 就是完成组/拆包的功能，不需要如 SARA 芯片中实现的其它加值功能，所以 SAR 也用 FPGA 实现。

一般而言，对比较大的网络，接口的 CAM/RAM 应该是外接，对于我们的 Demo 系统，入口表项可以取得较小，如果用 Altera 的 20k 系列，就可以把 CAM/RAM 集成实现在系统里面，同时逻辑上也比较简单。因此，CAM/RAM 也用 FPGA 来实现而不用外接式的(这里列举 Altera 提供的 CAM 大小：EP20K160E 可以提供 40Kbits 的 CAM，EP20K300E 为 72K....EP20K1500E 可提供 21Kbits 的 CAM，可以按需要配置成不同宽度和深度的 CAM)。对于 IP 包缓冲的选择，由于 20k 系列实现 CAM(Altera 的 APEX 20KE CAM)时，查找速率为 10ns(外接式 CAM 的典型速率为 20ns)，即：100Mpps。可见 IP 头的查找不会成为本接口设计中的瓶颈。

5 全文总结

随着互联网使用用户和接入的网络的数量逐渐增多，因特网承载的信息也向实时化、可视化、多样化的方向发展。为了用 IP 更好地传送图像、语音、视频等业务，需要对路由技术进行改进。把交换机和路由器相结合，产生了第三层交换技术，用固定长度的标签查找替代第三层地址的最长匹配算法，大大提高了分组转发的速度。多协议标签交换（MPLS）将多种第二层与第三层技术结合起来，毫无疑问，MPLS 将成为下一代 Internet 骨干网标准。多协议标签交换作为一种崭新的技术，其开发是近来高端路由器研发的热点，特别是接口模块的设计，要求具有极高的分组转发速率，这对 MPLS 路由器的设计提出了很高要求。

本文对高速 MPLS 路由器的接口模块进行了研究，主要工作包括以下几个方面：

1）全面分析传统 IP 网络的缺陷，比较了三种 IP 传输的方法（IP over ATM, IP over SDH/SONET, IP over WDM）的优缺点。

2）介绍了 MPLS 技术的基础知识。揭示了 MPLS 的核心技术——标签。整个 MPLS 协议都是建立在这一基础上的，围绕标签的一些操作——映射、绑定、封装、分发、交换、保持、合并以及清除而形成了 MPLS 的一整套协议规范。

3）讨论了一种新的基于光电混合的 Tbps 量级的 MPLS 路由器。采用分布式的转发方式，在每个输入接口上都设计一个转发模块，由接口模块完成数据包的转发功能。与一般路由器的基于软件查找方案不同，接口模块利用硬件来进行快速路由查找和标签交换，通过在接口模块中使用嵌入式内容可寻址存储器，路由/标签查找速率可高达千万次分组每秒量级。

4）提出了一种分布式高速路由器接口模块的具体设计方法，并将接口模块

收发模块、数据处理模块、转发处理模块、接口控制模块和 SAR 模块五部分。

5）采用 CAM 硬件查找方法实现了线速查找，解决了接口模块中路由表查找的瓶颈。

6）使用流水线设计，并行处理，实现了各部件的高速处理。

7）完成了整个接口模块的研制，做出了接口模块的实验板。在文中详细地阐述了各个模块的具体实现方法与过程，并给出了相关的实验数据。

致　　谢

三年的硕士学习生活马上就要结束了。在这里我想对所有关心和帮助过我的人说一声“非常感谢”。

首先，我要感谢的是我的导师曹明翠老师和罗风光老师。在整个硕士学习阶段，从选题、实验、到论文的完成，都倾注了导师的心血。曹老师和罗老师严谨的治学态度、广博的知识修养、无私奉献的敬业精神给我留下了深刻的印象，使我受益非浅。在此对导师表示衷心的感谢和敬意。

在研究工作中，曾经得到很多老师的热心帮助。特别要感谢的是徐军老师、周新军老师，在他们的共同关心下，我的研究工作得以顺利的进行。他们孜孜不倦的教诲经常给我重大的启示，使我的知识水平有了较大的提高，在此对他们表示感谢。

感谢本实验室的全体在读博士生、硕士生。特别感谢博士生罗志祥、刘儿儿，陈春汉、胡巧燕，硕士生汪浩然，方助军，陈涛，谭伟，裘菁，谢胡，黄平，冯勇华在课题研究中对我提供无私的帮助以及和我进行的许多有益的讨论。

感谢杨振宇、赵茗、胡鹏、季杭峰、张晟以及激光研究院99级研究生的全体同学。

在此我要感谢我的女友，学业的顺利完成与她的关心与支持是分不开的。最后，要感谢的是我的家人，家人在背后默默的支持是我求学的动力。

在以后的日子里，我会记住成长道路上关心和帮助过我的所有人，并不断提高自己，为社会做出更大奉献。

李峰

2002年4月于华工园

参考文献

1. A. Asthana, C. Delph, H. V. Jagadish, and P. Krzyzanowski, Toward a gigabit IP router, *J. High-Speed Networks*, vol. 1, pp. 281–288, 1992.
2. Xipeng Xiao, L.M.Ni, "Internet QOS: a big picture", IEEE Network, Vol.13, No.2, pp.8-18, 1999
3. C. 胡伊特马, 因特网路由技术, 清华大学出版社, 北京, 1998
4. P. Bhaniramka, W. Sun and R. Jain, "Quality of Service using Traffic Engineering over MPLS: An Analysis," *Globecom'99*.
5. Laubach, M. Classical IP and ARP over ATM. RFC 1577, January 1994
6. 姚宝富，戈玲，钟培军，基于SDH的宽带IP网，*电信技术*，1999年第4期，4-7
7. Srinivasan Seetharaman, "IP over DWDM", http://www.cis.ohio-state.edu/~jain/cis788-99/ip_dwdm/index.html
8. C.Y. 梅兹，IP交换技术协议与体系结构，*机械工业出版社*，北京，1999
9. P. Newman, "IP switching and gigabit routers," *IEEE Commun. Mag.*, **30:** 64–69, Feb 1997.
10. P. Newman, G. Minshall and T. Lyon, "IP switching: ATM under IP," *IEEE/ACM Trans. on Networking*, **6(2):** 117-129, 1998.
11. P. Newman, W. L. Edwards, et al., "Transmission of Flow Labelled IPv4 on ATM Data Links," RFC 1954, May 1996.
12. Y. Katsube, K. Nagami, H. Esaki, "Toshiba's Router Architecture Extensions for ATM: Overview," RFC 2098, Apr 1997.
13. K. Nagami, et al., "Toshiba's Flow Attribute Notification Protocol (FANP) Specification," RFC 2129, Apr 1997.
14. P. Newman, W. Edwards, et al., "Ipsilon's General Switch Management Protocol Specification Version 2.0," RFC 2297, Mar 1998.

15. P. Newm an, W. Edwards, et al., "Ipsilon Flow Management Protocol Specification for IPv4 Version 1.0." RFC 1953, May 1996.

16. Y. Rekhter, B. Davie, D. Katz, E. Rosen, and G. Swallow, "Cisco Systems' Tag Switching Architecture – Overview." RFC 2105, Feb 1997.

17. N. Feidman, A. Viswanathan, "ARIS Protocol Specification." *IBM Technical Report:* TR 29.2368, Mar 1998.

18. A. Viswanathan, N. Feldman, Z. Wang, R. Callon, "Evolution of multiprotocol label switching." *IEEE Communications Magazine*, **36(5)**, May 1998.

19. E. Rosen, et al., "Multiprotocol label switching Architecture." RFC 3031, Jan 2001.

20. K. C. Saraswat and F. Mohammadi, "Effect of scaling of interconnections on the time delay of VLSI circuits", IEEE Trans. Electron Devices , ED-29(4), pp. 645-650, 1982

21. D. M. Chiarulli, S. P. Levitan, P. Derr, *et al.*, Multichannel optical interconnections using imaging fiber bundles, *Int. Meet. Optical in Computing*, Aspen, 112-114, 1999

22. D. J. Goodwill, Free space optical interconnect for terabit network elements, *Int. Meet. Optical in Computing*, Aspen, 208-210, 1999

23. Y. Li, J. Ai and T. Wang, 100×100 opto-electronic cross-connector using OPTOBUS, *Proc. SPIE*, **3490**: 282-284, 1998

24. P. Lukowicz, S. Sinzinger, K. Dunkel, *et al.*, Design of an opto-electronic VLSI/Parallel fiber bus, *Proc. SPIE*, **3490**: 289-292, 1998

25. D. M. Chiarulli, S. P. Levitan, P. D. Raju, *et al.*, Multichannel optical interconnections using imaging fiber bundles, *Int. Meet. Optical in Computing*, Aspen, 112-114, 1999

26. J. Popelek and Y. Li, 256 channel fiber & free-space hybrid interconnect module, *Int. Meet. Optical in Computing*, Aspen, 106-108, 1999

27. K. Hirabayashi, Optical beam direction compensation system for board-to board free-space optical interconnection in high capacity ATM switch, *IEEE J. Lightwave Technol., 15(5)*: 158-182, 1997

28. Y. Liu, Smart pixel module development for free space optical interconnect, *Proc. SPIE*, **3490**: 528-531, 1998

29. K. Kato, M. Ishii, and Y. Inoue, Packaging of large-scale planar light-wave circuits, in *Proc. IEEE Components Technol. Conf.*, pp. 37–45, 1997

30. R. T. Chen, H. Lu, D. Robinson, *et al.*, Guided-wave planar optical interconnects using highly multiplexed polymer waveguide holograms, *IEEE J. Light. Technol.*, **10**: 888-897, 1992

31. I. Ogawa, *et al.*, Lossless hybrid integrated 8-ch optical wavelength selector module using PLC platform and PLC-PLC direct attachment techniques, in *Optical Fiber Commun. Conf. (OFC'98)*, Postdeadline Paper PD4.1-PD-4.4, 1998

32. H. Kosaka, M. Kajita, M. Yamada, *et al.*, A 16×16 optical full-crossbar connection module with VCSEL-array push/pull module and polymer-waveguide coupler connector, *IEEE Photon. Technol. Lett.*, **7**: 244-246, 1995

33. C. Duan and C. W. Wilmsen, Optoelectronic ATM switch using VCSEL and smart detector arrays, *Proc. SPIE*, **3490**: 103-106, 1998

34. O. Sjolund, D. A. Louderback, E. R. Hegblom, *et al.*, Free-space optical interconnect using flip-chip bonded microlensed arrays of monolithic vertical cavity lasers and resonent photodetectors, *Int. Meet. Optics in Computing*, Aspen, 215-217, 1999

35. Jack L. Jewell , etc. , "Vertical-Cavity Surface-Emitting Lasers: Design , Growth , Fabrication, Characterization", IEEE J. Of Quantum electronics , Vol.27, No.6, Jun. ,1991。

36. C. Duan and C. W. Wilmsen, Optoelectronic, "ATM switch using VCSEL

and smart detector arrays". Proc. SPIE. Vol.3490 pp.103-106. 1998

37. H. Tsuda. T. Sakamoto. M. Hikita. *et al.*. Hybrid-integrated smart pixels for MCM and board-level optical interconnects. *Int. Meet. Optics in Computing*. Aspen. 212-214. 1999

38. Staffan Blau. Jan Rooth. Jorgen Axell. Fiffi Hellstrand. Magnus Buhrgard . AXD 301: A new generation ATM switching system. *Computer Networks*. 31. 559–582. 1999

39. Perlman. R. Interconnections: Bridge and Routers. Reading. MA: Addison-Wesley. 1992

40. J. Moy. OSPF Version 2. RFC2328. April 1998

41. Hedrick. C.. "Routing Information Protocol". RFC 1058. Rutgers University. June 1988.

42. Bates. T. R.. Chandra. D. Katz. and Y. Rekhter. Multiprotocol Extensions for BGP-4. RFC 2283. February 1998.

43. Bates. T.. and R. Chandrasekeran. BGP Route Reflection: An Alternative to Full Mesh IBGR RFC 1966. June 1996.

44. Chandra. R.. P Ttaina. and T. Li. BGP Communities Attribute. RFC 1997. August 1996

45. S. Keshav and R. Sharma. Issues and trends in router design. *IEEE Communication. Magazine*. pp. 144–151. May 1998.

46. 王洵，张建峰，郑少仁，多协议标签交换技术（MPLS）在 ATM 网络中实现技术研究，*电讯技术*，1999 年，第 5 期，64－68

47. 李珂，顾尚杰，诸鸿文，MPLS 的框架及其关键技术，*数字通信*，2000 年，第 2 期，30－32

48. 魏颖琪，李晖晖，张光昭，ATM MPLS 系统设计，*数据通信*，1999 年，第 3 期，12－25

49. E. Rosen. Y. Rekhter. D. Tappan. et al.. "MPLS Label Stack Encoding." RFC 3032. Jan 2001.

50. L. Andersson. et al.. "LDP Specification." *RFC 3036. Jan 2001.*

51. Internet Draft, "Constraint-Based LSP Setup using LDP", draft-ietf-mpls-cr-ldp-03.txt, Sep., 1999

52. S. W. Fuhrmann, "Performance of a packet switch with crossbar architecture", IEEE Trans. Commun., Vol.41, No.8, pp.2131-2140, 1993

53. Y. Yeh, M. G. Hluchyj and A. S. Acampora, "The knockout switch: a simple, modulor architecture for high-performance packet switching", IEEE JSAC, Vol.5, No.8, pp.1274-1283, 1987

54. W. Doeringer, G. Karjoth, and M. Nassehi, Routing on longest matching prefixes, *IEEE/ACM Trans. Networking*, vol. 4, no. 1, pp. 86–97, Feb. 1996.

55. Gupta P.L in S,Mc Keown N. Routing lookups in hardware atmemory access speeds. In:Guerin R ed. Proceedings of theIEEE INFOCOM' 98. San Francisco,CA:IEEE Computer Society Press,1 998. 1 2 40～ 1 2 47

56. Brodnik A,Carlsson S.Degermark M et al. Small forwarding tables for fast route lookups. ACM Computer Communica-tion Review,1 997,2 7(4) :3～ 1 4

57. Waldvogel M,Turner J,Plattner B. Scalable high speed IP routing lookups. ACM Computer Communication Review,1 997,2 7(4) :2 5～ 36

58. S. Nilsson and G. Karlsson, Fast address lookup for Internet routers, in *Proc. IFIP 4th Int. Conf. Broadband Commun...* pp. 11–22, April, 1998

59. A. Brodnik, S. Carlsson, M. Degermark, and S. Pink, Small forwarding tables for fast routing lookups, in *Proc. ACM SIGCOMM*. pp. 3–14, Aug. 1997

60. P. Gupta, S. Lin, and N. McKeown, Routing lookups in hardware at memory access speeds, in *Proc. INFOCOM*, March 1998, Session 10B-1, 1998

61. M. Waldvogel, G. Varghese, J. S. Turner, and B. Plattner, Scalable high speed IP routing lookups, in *Proc. ACM SIGCOMM97*, pp. 25–36, August, 1997

62. Chandranmenon, C., and G. Varghese. Trading Packet Headers for Packet Processing. In Proceedings of ACM SIGCOMM 95, September 1995, 162—173

附录 1 攻读学位期间发表论文目录

1、 李峰，曹明翠，罗风光，刘儿兀. 高速 MPLS 路由器接口的设计. 华中科技大学学报，2002(6).

2、 李峰，曹明翠，罗风光，刘儿兀 陈春汉 IP 网络中的 Qos 保证 电子技术 2001，Vol.28，NO.6，9～12

3、 李峰，曹明翠，罗风光 全光通信网 电信过程技术与标准化 2001.4 44～47

4、 刘儿兀，曹明翠，陈春汉，李峰，“多协议标签交换(MPLS)路由器的设计.” *数据通信*. 2001(1).

5、 Liu Erwu. Cao Mingcui. Chen chunhan. Li feng. "Performance Comparison and Analysis of IP and MPLS networks." *The Journal Of China Universities Of Posts And Telecommunications*. 2001(2).

附录2 术 语

AAL	Asynchronous Transfer Mode Adaptation Layer	ATM 适配层
ADSL	Asymmetrical Digital Subscriber Line	非对称数字用户线
ARIS	Aggregate Route based IP Switching	基于汇聚路由的 IP 交换
ARP	Address Resolution Protocol	地址解析协议
AS	Autonomous System	自治系统
ASIC	Application Specific Integrated Circuit	专用集成电路
ATM	Asynchronous Transfer Mode	异步传输模式
BE	Best-Effort	尽力而为
BGP	Border Gateway Protocol	边界网关协议
CAM	Content Addressable Memory	内容可寻址存储器
CIDR	Classless Inter-Domain Routing	无类域间寻径
CoS	Class of Service	服务类
CRC	Cyclic Redundancy Check	循环冗余校验
CR-LDP	Constraint-based Routing Label Distribution Protocol	基于约束寻径的标签发布协议
DiffServ	Differentiated Services	区分服务
DLCI	Data Link Circuit Identifier	数据链路电路标识
DWDM	Dense Wavelength Division Multiplexing	密集波分复用
EMI	Electromagnetic Interference	电磁干扰
FPGA	Field Programmable Gate Arrays	现场可编程门阵列
FDDI	Fiber Distributed Data Interface	光纤分布式数据接
FEC	Forward Equivalence Class	转发等价类
FIB	Forward Information Base	转发信息库
FIFO	First-In-First-Out	先入先出
FR	Frame Relay	帧中继
FTN	FEC To NHLFE map	FEC 到 NHLFE 的映射
GSMP	General Switch Management Protocol	通用交换机管理协
GbE	Giga bit Ethernet	千兆以太网

HEC	Header Error Control	头错误控制
ICMP	Internet Control Message Protocol	因特网控制消息协议
IETF	Internet Engineering Task Force	因特网工程任务组
IFMP	Ipsilon Flow Management Protocol	Ipsilon 的流管理
IGP	Interior Gateway Protocol	内部网关协议
ILM	Incoming Label Map	入标签映射
IntServ	Integrated Services	集成服务
IP	Internet Protocol	因特网协议
IPSec	IP Security protocol	IP 安全协议
IPOA	IP Over ATM	ATM 上的 IP
IS-IS	Intermediate System-Intermediate System	中介系统-中介系统
LAN	Local Area Network	局域网
LDP	Label Distribution Protocol	标签发布协议
LER	Label Edge Router	标签边缘路由器
LIB	Label Information Base	标签信息库
LLC	Logical Link Control	逻辑链路控制
LSR	Label Switching Router	标签交换路由器
LSP	Label Switched Path	标签交换路径
MAC	Media Access Control	媒介接入控制
MIB	Management Information Base	管理信息库
MII	Media Independent Interface	媒介无关接口
MLPS	Million Lookups Per-Second	每秒百万次查找
MPLS	Multi-Protocol Label Switching	多协议标签交换
MPPS	Million Packets Per-Second	每秒百万个分组
MSM	Mental-Semiconductor-Mental	金属-半导体-金属
MTU	Maximum Transmission Unit	最大传输单元
NHLFE	Next Hop Label Forwarding Entry	下一跳标签转发入
NHRP	Next Hop Resolution Protocol	下一跳解析协议
NNI	Network-to-Network Interface	网络-网络接口
OSI	Open System Interconnection	开放系统互连
OSPF	Open Shortest Path First	开放最短路径优先
PDU	Protocol Data Unit	协议数据单元

PHB	Per-Hop Behavior	每一跳行为
PNNI	Private Network-to-Network Interface	专有网络到网络接口
POS	Packect Over SDH	SDH 上的包
PPP	Point-to-Point Protocol	点到点协议
PVC	Permanent Virtual Circuit	永久虚电路
QoS	Quality of Service	服务质量
RAM	Random Access Memory	随机存取存储器
RIP	Routing Information Protocol	寻径信息协议
RSVP	Resource ReserVation Protocol	资源预留协议
RTOS	Real-Time Operating System	实时操作系统
SAR	Segmentation And Reassemble	分段和重装
SDH	Synchronous Digital Hierarchy	同步数字体系
SNMP	Simple Network Management Protocol	简单网络管理协议
SONET	Synchronous Optical NETwork	同步光网络
TCP	Transmission Control Protocol	传输控制协议
TDP	Tag Distribution Protocol	标记发布协议
TE	Traffic Engineering	流量工程
ToS	Type of Service	服务类型
TTL	Time-To-Live	存活时间
UDP	User Datagram Protocol	用户数据报协议
UNI	User-to-Network Interface	用户到网络接口
VC	Virtual Circuit	虚电路
VC	Virtual Channel	虚信道
VCI	Virtual Channel Identifier	虚信道标识符
VCSEL	Vertical Cavity Surface Emitting Laser	垂直腔表面发射激光器
VHDL	Very High-speed Hardware Description Language	极高速硬件描述语言
VPI	Virtual Path Identifier	虚通道标识符
WAN	Wide Area Network	广域网
WDM	Wavelength Division Multiplexing	波分复用

附录 3 Tx 模块部分 VHDL 程序

下面是 tx 模块的 VHDL 程序：

```
library IEEE;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

ENTITY TXMac IS
  port(Byteclk,Txdclk:in std_logic;
       Tx_valid: in std_logic;
       CRS,COL,FullDuplex:in std_logic;
       power_Rst: in std_logic: --the reset signal of the whole system.
Reset all ,when it='1';
       Tx_data:in std_logic_vector(7 downto 0);
       TXD:out STD_LOGIC_VECTOR(3 downto 0);
       tx_ok: out std_logic;
       Tx_en,Tx_er:out std_logic
       );
END TXMac;
ARCHITECTURE TXMac1 OF TXMac IS

COMPONENT TxCRC_Generator
 port(
 power_rst:        in std_logic;        -- HDLC controller generated reset
BYTE_CLK:    in std_logic;          -- HDLC controller byte clock
DATA_BYTE:   in std_logic_vector(7 downto 0); -- data octets
EN:          in std_logic;          -- CRC enable synched to BYTE_CLK
CRC_out_EN:   in std_logic;
DATA_out:  out std_logic_vector(7 downto 0)
```

```
        );
END COMPONENT;
COMPONENT TxData_MUX
 port(
        txdata:   in std_logic_vector(7 downto 0); -- data octets
        crc_data:   in std_logic_vector(7 downto 0);
        mux_sel:   in std_logic_vector(1 downto 0);
         DATA_out: out std_logic_vector(7 downto 0)
          );
END COMPONENT;
COMPONENT TxData_P_S
 port(Byteclk,Txdclk_2:in std_logic;
        power_rst: in std_logic;
        Transmit_Enable: in std_logic;
        power_Rst: in std_logic; --the reset signal of the whole system.
Reset all ,when it='1';
        Txdata:in std_logic_vector(7 downto 0);
        TXD:out STD_LOGIC_VECTOR(3 downto 0);
         Tx_en,Tx_er:out std_logic
           );
END COMPONENT;
COMPONENT TxSync_FIFO
 port(
        power_rst: in std_logic;
        Transmit_Enable: in std_logic;
        Byteclk,Txdclk:in std_logic;
        Txdata:in std_logic_vector(7 downto 0);
        Txout:out STD_LOGIC_VECTOR(7 downto 0)
        );
END COMPONENT;
```

```
COMPONENT TxLength_Couter
 port(
        power_rst: in std_logic;
        Transmit_Enable: in std_logic;
        Txdclk:in std_logic;
        count_len:out std_logic_vector(11 downto 0)
          );
 END COMPONENT;
 COMPONENT Random_Number_Generator
  port(
        power_rst: in std_logic;
        Txdclk:in std_logic;
        col_attm:in std_logic_vector(3 downto 0);
        backoff_time:out STD_LOGIC_VECTOR(9 downto 0)
          );
 END COMPONENT;
 COMPONENT BackOff_Timer
  port(
        power_rst: in std_logic;
        Txdclk:in std_logic;
        start_backoff:in std_logic;
        backoff_time:in STD_LOGIC_VECTOR(9 downto 0);
        backoff_p: out std_logic
          );
 END COMPONENT;
 COMPONENT TxCollision_Counter
  port(
        power_rst: in std_logic;
        Transmit_Enable: in std_logic;
        Txdclk:in std_logic;
```

```
        crs,col,fullduplex:in std_logic;
        coll,start_backoff:out std_logic;
        col_attm:out std_logic_vector(3 downto 0)
          );
END COMPONENT;
COMPONENT IFG_Timer
 port(
        power_rst: in std_logic;
        Transmit_Enable: in std_logic;
        Txdclk:in std_logic;
        crs,fullduplex:in std_logic;
        Transmit_available:out std_logic
          );
END COMPONENT;
COMPONENT Tx_State_Machine
 port(
        power_rst: in std_logic;
        byteclk,Txdclk:in std_logic;
        crs,fullduplex:in std_logic;
        Transmit_available:in std_logic
        coll:in std_logic;
        backoff_p: in std_logic;
        count_len:in std_logic_vector(11 downto 0);
        crc_EN:out std_logic;          -- CRC enable synched to
BYTE_CLK
        CRC_out_EN:out std_logic;
        mux_sel:   out std_logic_vector(1 downto 0);
        Transmit_Enable: out std_logic;
        tx_ok: out std_logic;
        Txdclk_2:out std_logic
```

```
        );
END COMPONENT;
    signal CRC_data,Txout,DATA_out:std_logic_vector(7 downto 0);
    SIGNAL Txdclk_2,CRC_en,CRC_Out_en:std_logic;
    signal mux_sel: std_logic_vector(1 downto 0);
    signal Transmit_Enable,Transmit_available,backoff_p:std_logic;
    signal coll:std_logic;
    signal start_backoff:std_logic;
    signal count_len:std_logic_vector(11 downto 0);
    SIGNAL col_attm: std_logic_vector(3 downto 0);
    signal backoff_time: STD_LOGIC_VECTOR(9 downto 0);

BEGIN
crc32:TxCRC_Generator
        port map(BYTE_CLK=>Byteclk,
                 power_rst=>power_rst,
                 DATA_BYTE=>txdata,
                 EN=>crc_en,
                 CRC_out_EN=>crc_out_en,
                 DATA_out=>CRC_data
        );
mux:TxData_MUX
 port map(
        txdata=>txdata,
        crc_data=>crc_data,
        mux_sel=>mux_sel,
        DATA_out=>DATA_out,
         );
p_s:TxData_P_S
 port map(txCLK_2=>txclk_2,
```

```
        power_rst=>power_rst,
        Transmit_Enable=>Transmit_Enable,
        Txdata=>Txout,
        TXD=>TXD,
        Tx_en=>Tx_en,
        Tx_er=>Tx_er
            );
    fifo:TxSync_FIFO
     port map(
            power_rst=>power_rst,
            Transmit_Enable=>Transmit_Enable,
            Byteclk=>Byteclk,
            Txdclk=>Txdclk,
            Txdata=>DATA_out,
            Txout=>Txout
                );
    len:TxLength_Couter
     port map(
            power_rst=>power_rst,
            Transmit_Enable=>Transmit_Enable,
            Txdclk=>Txdclk,
            count_len=>count_len
                );
    rand: Random_Number_Generator
     port map(
            power_rst=>power_rst,
            Txdclk=>Txdclk,
           col_attm=> col_attm,
           backoff_time=>backoff_time
                );
```

```
backoff: BackOff_Timer
 port map(
        power_rst=>power_rst,
        Txdclk=>Txdclk,
        start_backoff=>start_backoff,
        backoff_time=>backoff_time
        backoff_p=>backoff_p
            );
    collision: TxCollision_Counter
     port map(
            power_rst=>power_rst,
            Transmit_Enable=>Transmit_Enable,
            Txdclk=>Txdclk,
            crs=>crs,
            fullduplex=>fullduplex,
            col=>col,
            coll=>coll,
            start_backoff=>start_backoff,
            col_attm=>col_attm
               );
    IFG: IFG_Timer
     port map(
            power_rst=>power_rst,
            Transmit_Enable=>Transmit_Enable,
            Txdclk=>Txdclk,
            crs=>crs,
            fullduplex=>fullduplex,
            Transmit_available=>Transmit_available
            );
    state_machine: Tx_State_Machine
```

```
port map(
        power_rst=>power_rst,
        Transmit_Enable=>Transmit_Enable,
        Txdclk=>Txdclk,
        Byteclk=>Byteclk,
        crs=>crs,
        fullduplex=>fullduplex,
        Transmit_available=>Transmit_available
        coll=>coll,
        backoff_p=>backoff_p
        count_len=>count_len
        crc_EN=>crc_en,
        CRC_out_EN=>crc_out_en,
        mux_sel=>mux_sel,
        tx_ok=>tx_ok,
        Txdclk_2=>Txdclk_2
          );
END ARCHITECTURE TXMac1;
```