基于单 FPGA 的可伸缩高速 IP 查找设计

FPGA based Highly Efficient and Scalable IP Lookup

袁　晶　（丹麦科技大学信息数学建模学院 2800）
陆颖迪　（丹麦科技大学 COM 研究中心 2800）

摘要：现今对 IP 查找的解决策略中，很少同时考虑了经济性和有效性。本文提供了一个经济有效：基于 FPGA（现场可编程序门阵列）而且可伸缩（scalable）的设计。通过扩展内部的快速 IP 查找引擎，可以保证在最差情况下每秒 20,000,000 次查找，平均情况下每秒 36,231,002 次。实验模拟所用的路由表信息来自于 Mae West 的路由表快照。

关键词：IP 查找　FPGA　路由器

1　引言

IP 查找的核心问题是如何高效的使用内存和快速查找转发表（forward table）。随着无类别的域间路由的广泛应用，物理连接速度的提高以及高性能路由器断口数目的增加，对高效路由算法的应用需求越来越迫切。在可接受的成本下，对大容量的路由表查找的可伸缩性的设计恰恰能满足这种需求。

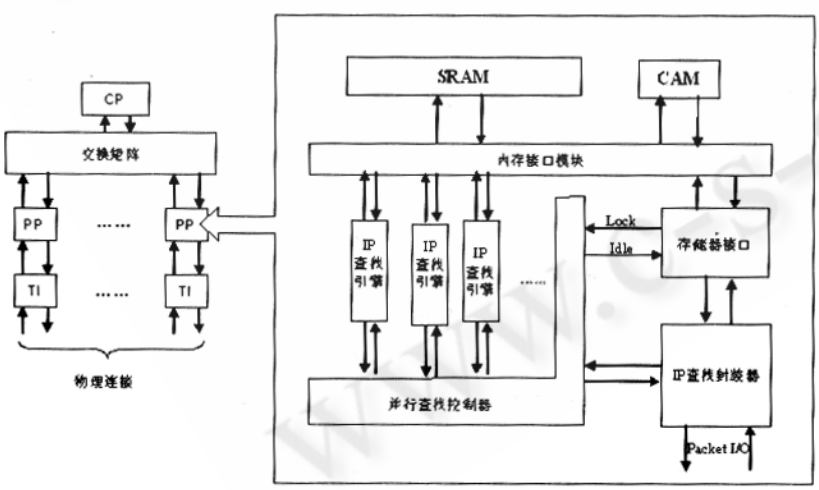

图 1　左：路由器模块示意图　　右：PP 系统整体构架图

FPGA（Field Programmble Gate Array）现场可编程门阵列。可编程逻辑的价值在于其缩短电子产品制造商开发周期以及帮助他们更快地将产品推向市场的能力。FPGA 被应用于广泛的应用中，从数据处理和存储直到仪器仪表、电信和数字信号处理。

对于 FPGA，设计人员可利用价格低廉的软件工具快速开发、仿真和测试其设计。然后，可快速将设计编程到器件中，并立即在实际运行的电路中对设计进行测试。原型中使用的可编程逻辑器件与正式生产最终设备（如网络路由器、DSL 调制解调器、DVD 播放器、或汽车导航系统）时所使用的可编程逻辑器件完全相同。

另一个关键优点是在设计阶段中客户可根据需要修改电路，直到对设计工作感到满意为止。一旦设计完成，客户可立即投入生产，只需要利用最终软件设计文件简单地编程所需要数量的可编程逻辑器件就可以了。

2　系统设计

2.1　系统构架

为了提供可伸缩性的功能，模块化设计方法贯穿于整个设计之中。图 1 的左半部分是路由器的构架。PP 是端口处理器（port processor），TI 是传送接口（Transmission Interface），CP 是控制处理器（Control processor）。图 1 的右半部分具体描述了 PP 部分，即整个 IP 查找系统的构架和模块。主要由以下模块组成：

（1）IP 查找引擎实现了单线程，改进型 Prefix 查找算法。

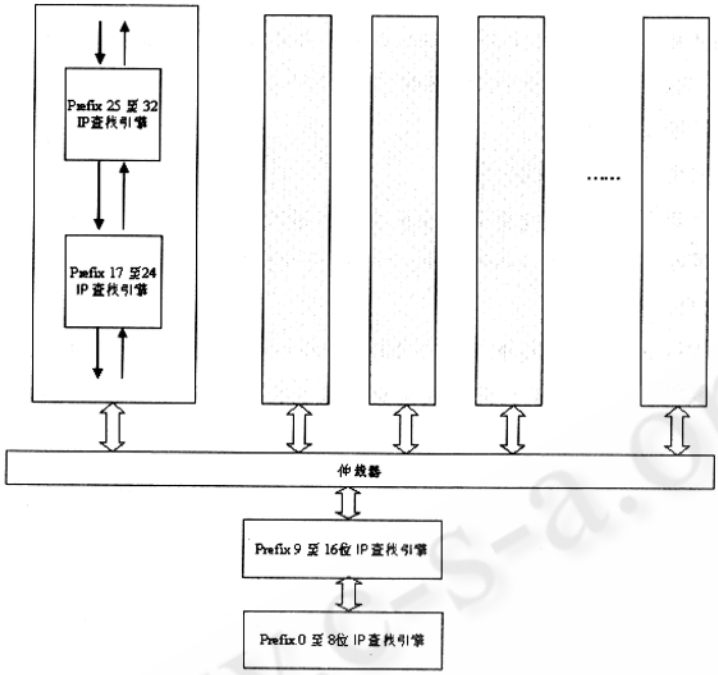

图 2　IP 查找引擎结构图

(2) 并行查找控制器的功能是控制多个 IP 查找引擎并行运算。这使得本系统可以根据不同的吞吐量要求,配置不同数目的 IP 查找引擎,具有极强的可伸缩性。

(3) 控制处理器接收路由信息更新命令(删除,修改和新加 prefix)和对内存进行更新处理。如果控制处理器和 IP 查找引擎同时工作的话,可能会产生冲突,导致 IP 查找失败。控制处理器和并行查找控制器之间的 idle 和 lock 信号是一对 4 象限握手(four phase handshake)信号,用于防止 IP 查找与内存更新的冲突。

(4) 存储器接口存在于 CAM,SRAM 和 IP 查找引擎,控制处理器之间,它有两个功能。一是对 IP 查找引擎的内存访问(SRAM 和 CAM)进行仲裁。注意由于 idle 和 lock 两个控制信号的作用,控制处理器和 IP 查找引擎不会同时工作。所以它不用在控制处理器和 IP 查找引擎之间进行仲裁。二是完成内存分配(assign memory)和内存释放(release memory)。

(5) IP 查找封装器从输入的数据包中抽取 IP 地址信息,写入一个 FIFO 队列,并行查找控制器从中读取 IP 地址,分配给某一个 IP 查找引擎。并行查找控制器得到查询结果后写入另一个队列,IP 查找封装器从中读出数据。

和 David E. Taylor 提供的解决方案相比,本系统在系统结构上有以下改进:

① 对多个 IP 查找引擎的内存访问的仲裁任务,由并行查找控制器转移到内存接口模块完成。首先,这样做简化了并行查找控制器的设计,使其能够工作于更高的主频下;其次 IP 查找引擎直接与内存接口通信,无须通过并行查找控制器的传递,提高了内存的访问效率,在最差情况下,减少了一半的内存访问量。

② 使用'Lock'和'Idle'两个信号来协调 IP 查找与更新,避免二者冲突。

③ 根据改进型 Prefix 树算法的要求,在内存接口中增加了对 CAM 的访问和仲裁功能。

2.2 IP 查找引擎

如图 2 所述,prefix 按长度被分成 4 个部分,一是 32 位至 25 位,二是 24 位至 17 位,三是 16 位至 9 位,四是 8 位至 0 位。其中第一、四部分在 CAM 中查询,第二、三部分在 SRAM 中查询。四个部分分别由四个模块共 600 行 VHDL 代码实现。系统结构如图 4 所示。四个模块串行查找,顺序依次是一、二、三、四部分。根据最长匹配原则,如果前一级模块查找成功的话,则不用进入下一级。和并行查找相比(四个模块同时查找),每次查找占用的内存带宽降低了将近一半。注意模块三、四是在多个 IP 查找引擎中共享,这样做的好处,一是在不影响性能的前提下,大约节约 1/3 的 FPGA 的器件资源。因为在 IPMA 提供的路由表中,仅有 1.2% 的 prefix 属于第三、四部分,本系统在应用中一般不会多于 16 个 IP 查找引擎,远远小于 83(1/1.2%)。所以共享模块三、四不会影响性能;二是降低了存储器接口的负担,因为访问 Block RAM 的只有一个模块(模块三),所以不需要对 block Ram 的访问进行仲裁。

2.3 并行查找控制器

并行查找控制器的功能是控制多个 IP 查找引擎并行运算。它有两个基本任务,一是分配 IP 给 IP 查找引擎,二是将查找结果输出给 IP 查找封装器。第一个任务十分简单,只要将从封装器传入的 IP 地址写入先入先出队列(FIFO)。然后从 FIFO 中读出地址并分配给空闲的 IP 查找引擎。第二个任务是将查找结果按 IP 地址读入的顺序输送到封装器。因为对于不同地址,IP 查找引擎需要的查找时间不同,所以并行查找控制器需要对输出结果排序,使其与输入的 IP

地址的顺序相对应。

针对第二个任务，本文设计思想是，采用查寻结果队列，一种新的随机存储先入先出的数据结构(RAFIFO)和 IP 查找引擎存储地址阵列(简称存储地址阵列)来解决排序问题。RAFIFO 和普通的 FIFO 一样，RAFIFO 分别用头和尾指针来指向队列的头和尾，并且和随机存储器一样，它可以根据地址存储。存储地址阵列中的每一个存储单元与一个 IP 查找引擎相对应，它记录该 IP 查找引擎在查找完成后所要写入的查寻结果队列的地址。当某个 IP 查找引擎被分配 IP 后，RAFIFO 中的 Head 指针指向的那一项置成无效(invalid)，其地址存入查寻结果队列中对应的 IP 查找引擎的存储单元，并且 head 移位指向下一个地址。当 IP 查找引擎得到结果后，它将结果存入查寻结果队列，其写入地址为存储地址阵列中的地址。如果 RAFIFO 中的 tail 所指的数据有效且队列不为空，数据输出到封装器，同时 tail 指向下一地址。

在 Xilinx Virtex 1000E 的硬件环境下，支持 16 个 IP 查找引擎的并行查找控制器的时钟频率最高可以达到 102MHz，超过了 100MHz 的系统时钟频率。在性能测试中，在满负荷的情况下 IP 查找引擎的平均空闲率仅为 7%，证明并行查找控制器对 IP 查找引擎的调度是高效的。

2.4 存储器接口

存储器接口有 SRAM 接口和 CAM 接口组成。SRAM 接口由内存分配模块，内存释放模块，内存访问仲裁模块组成；而 CAM 仅由内存访问仲裁模块。

内存分配模块和内存释放模块的效率在很大程度上取决于内存的组织。本设计使用位表记录每位内存的状态(是否空闲)，使用最高支持 16Mbit 的三级索引表来查找空闲内存。对于每次内存分配和内存释放操作，只需三次内存访问。效率高而且易于硬件实现。

2.5 内存访问仲裁

内存访问仲裁模块对 IP 查找引擎的内存访问(SRAM 和 CAM)进行仲裁。这个模块也是实现系统可扩展性的关键模块。为了保证公平，高效和易于硬件实现，系统使用轮转调度机制(Round－Robin Scheduling)的时钟调度法(Clock Policy)均衡各个 IP 查找引擎的内存访问。工作原理是，首先创建一个环状帧队列，每一帧对应一个 IP 查找引擎，每一帧里保存 IP 查找引擎的内存请求状态。每个时钟周期开始后，帧指针沿顺时针方向转动，如果在转动过程中，帧指针所指向帧的内存请求状态是请求内存访问，则允许其进行内存访问，同时帧指针停止转动。如果没有 IP 查找引擎请求内存访问，则帧指针在转动一圈后，停止转动。

根据综合报告，在 Xilinx Virtex 1000E 的硬件环境下，支持 18 个 IP 查找引擎的内存访问仲裁模块的时钟频率最高可以达到 120MHz，明显高效。

3 系统性能

理论最差情况系统性能是可推算的。以下采用真实的路由表(Mae West 的路由表快照)进行系统平均性能的测试。

系统测试基于 Xilinx Virtex 1000E，运行于 100MHz，共有 16 个 IP 查找引擎。其中每个 IP 查找引擎使用 2% 的逻辑门资源，并行查找控制器使用 3% 的逻辑门资源，存储器接口使用 3.5% 的逻辑门资源，并行查找控制器使用 10% 的逻辑门资源，IP 查找封装器使用 2% 的逻辑门资源。整个系统共使用了 50.5% 的逻辑门资源。

测试所用的路由信息来自于 Mae West 的 2001 年 7 月的路由表快照。其中 29587 条路由信息存于 4 兆片外零总线转变(ZBT)静态随机存取存储器(SRAM)，38 条路由信息存于片内 CAM。Shortcut 表存于 Block RAM。为了方便测试和性能评估，2048 条 IPv4 目的地址存于 FPGA 的 Block RAM 中供 IP 查找封装器读取，其目的地址随机从路由表中产生。

当测试开始时，控制单元首先决定用于查找的 IP 查找引擎的数目，路由信息更新频率。IP 查找引擎的数目从 1 到 16。路由信息更新频率从 1000 次变化到 100000 次。

图 3 是没有更新操作的测试结果。对于配置了 16 个 IP 查找引擎的系统，其吞吐量高达每秒 36,231,002 次。但相对于单引擎而言，其吞吐量随着 IP 查找引擎的增加而下降，10 个引擎的系统下降了 16.7%，16 个引擎的系统下降了 37.5%。这主要有两个原因，首先因为随着内存带宽使用率提高，IP 查找引擎的内存访问冲突提高，其对内存的访问需要等待仲裁。特别是在内存带宽耗尽的情况下，系统吞吐量不再随着 IP 引擎数的增加而增加。其次是随着引擎数目的提高，并行查找控制器的负担也增大，IP 查找引擎的平均空闲率也随着增大。如图所示，16 个引擎已接近系统吞吐量极限，再增加引擎数目，系统的吞吐量增加已不显著。

图 4 反映出不同更新频率下的系统平均吞吐量。从图中可见，小于每秒更新 1,000 次的情况下，系统吞吐量基本不受影响；每秒更新 10,000 次，系统吞吐量大约降低 2%，影响十分小；每秒更新 100,000 次，系统吞吐量大约降低 21%，

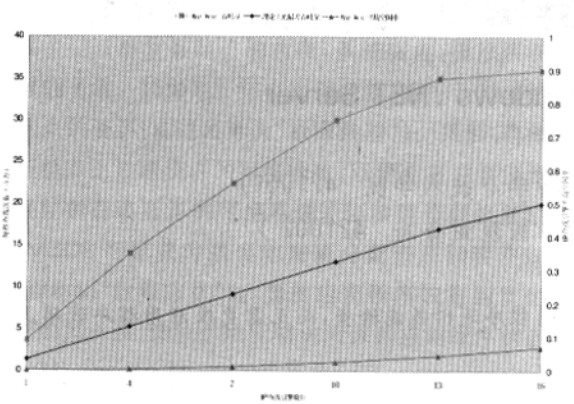

图 3　无更新情况下，IP 查找引擎数目与每秒查找数目和平均空闲率关系曲线

但是这种情况很少发生，因为实际情况是更新操作每秒低于1000 次。

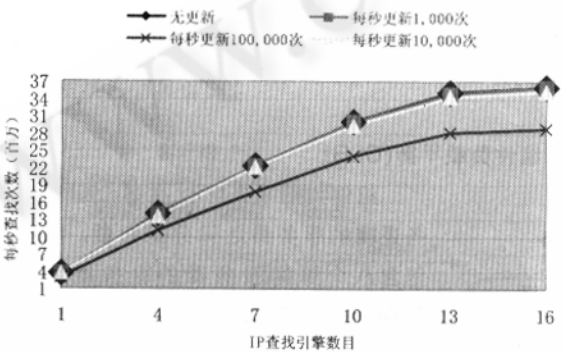

图 4　不同更新频率下，IP 查找引擎数目与每秒查找数目之关系曲线

下表是将本系统与华盛顿大学 Saint Louis 分校的 FIPL 系统进行比较。理论最差情况下，最高吞吐量，本系统高于后者 120%；Mae West 平均吞吐量，高于后者 149%。

表 1　系统性能比较

	理论最差情况下，最高吞吐量(百万)	平均情况下，最高吞吐量(百万)(Mae West)
FIPL	9.1	10.1
本系统	20.0	36.2

4　结束语

随着光通信网络速度的不断增加，以及对嵌入式网络服务高性能，高灵活性的需求，当今网络路由器必须更加高效率和具备可编程性。其中 IP 地址查询是路由器的一个最主要的功能，也是瓶颈所在。本文的快速 IP 查询设计采用 Michael Berger 的算法，即快捷表和多维树技术；在 FPGA 主板，Xilinx Virtex 1000E 上实现了一种可伸缩，高性能的 IP 查找引擎，其查找速度可达到至少 2000 万次/每秒，可满足 10Gbits 以太网的要求。相对于当今在系统芯片领域，普遍应用的昂贵的商业解决方案，即大容量的内容可寻址存储器(CAM)和半导体专业集成电路(ASICs)而言，本文提供的低成本，高性能设计方案具有很强的竞争力和很好的应用前景。

参考文献

1 Michael Berger. "IP Lookup With Low Memory Requirement and Fast Update", IEEE, Jun. 2003 HPSR, "High Performance Switching and Routing", P287－291.

2 Mae－West routing database, "The Internet performance Measurement and Analysis (IPMA) project", http://www.merit.edu/ipma/routing_table/, 10 Oct.2002.

3 BELENKIY, Andrey. Deterministic IP Table Lookup at Wire Speed. http://www.isoc.org/inet99/proceedings/4j/4j_2.htm. New Jersey Institute of Technology USA. Last access July 24,2000.

4 Dong－Kil Shin, et al. "A Compact Routing Lookups Schemes for Implementation In Single－chip FPGA", Journal of the Korean Physical Society (JKPS), Volume 40[Special Issue],No.4, pp. 759－764, April 2002.

5 Jean－Louis Brelet. Using Block RAM for High Performance Read/Write CAMs. http://www.xilinx.com/bvdocs/appnotes/xapp204.pdf. XAPP204 (v1.2) May 2, 2000.

6 David E. Taylor, John W. Lockwood et al, Scalable IP Lookup for Programmable Routers, IEEE Infocom 2002, New York NY, June 23－27, 2002.

7 William Stallings. Computer Organization And Architecture Designing for Performance, 4th ed. Prentice Hall. Inc. 1996.