以太网到多路 E1 适配电路设计及 FPGA 实现

摘要：介绍了一种基于现场可编程门阵列（FPGA）的以太网数据-多路 E1 反向复用器同步电路设计，分析了 FPGA 具体实现过程中的一些常见问题。该设计采用 VHDL 硬件描述语言编程，可以实现以太网数据在多路 E1 信道中的透明传输，适配电路芯片内置 HDB3 编解码器和数字时钟提取电路。

关键词：FPGA 反向复用 以太网数据 以太网络变压器 EI 信道 通信变压器 适配电路

伴随着 Internet 的迅速发展，IP 已经成为综合业务通信的首选协议，其承载的信息量也在成倍增长，如何利用现有的电信资源组建宽带 IP 网络是近年来研究的热点。目前，比较成熟的技术主要有 IP over SDH(POS)和 IP over ATM（POA）。POS 将 IP 包直接装入 SDH 的虚容器中，通道开销少、实现简单，具有自动保护切换功能；POA 的复接过程比较复杂，可以通过高系统开销提供较好的服务质量保证（QOS）。从目前的市场看，各大通信设备商都推出了基于 POS/POA 的产品，但总体成本较高，主要面向的是一些高端应用。对于带宽需求在十几兆以下的点对点通信而言，上述两种技术的优势并不明显。本文介绍的适配电路将以太网数据适配到 E1 信道传输，通过配置 E1 信道数量控制带宽，针对这类应用提供了一种经济灵活的解决方案。

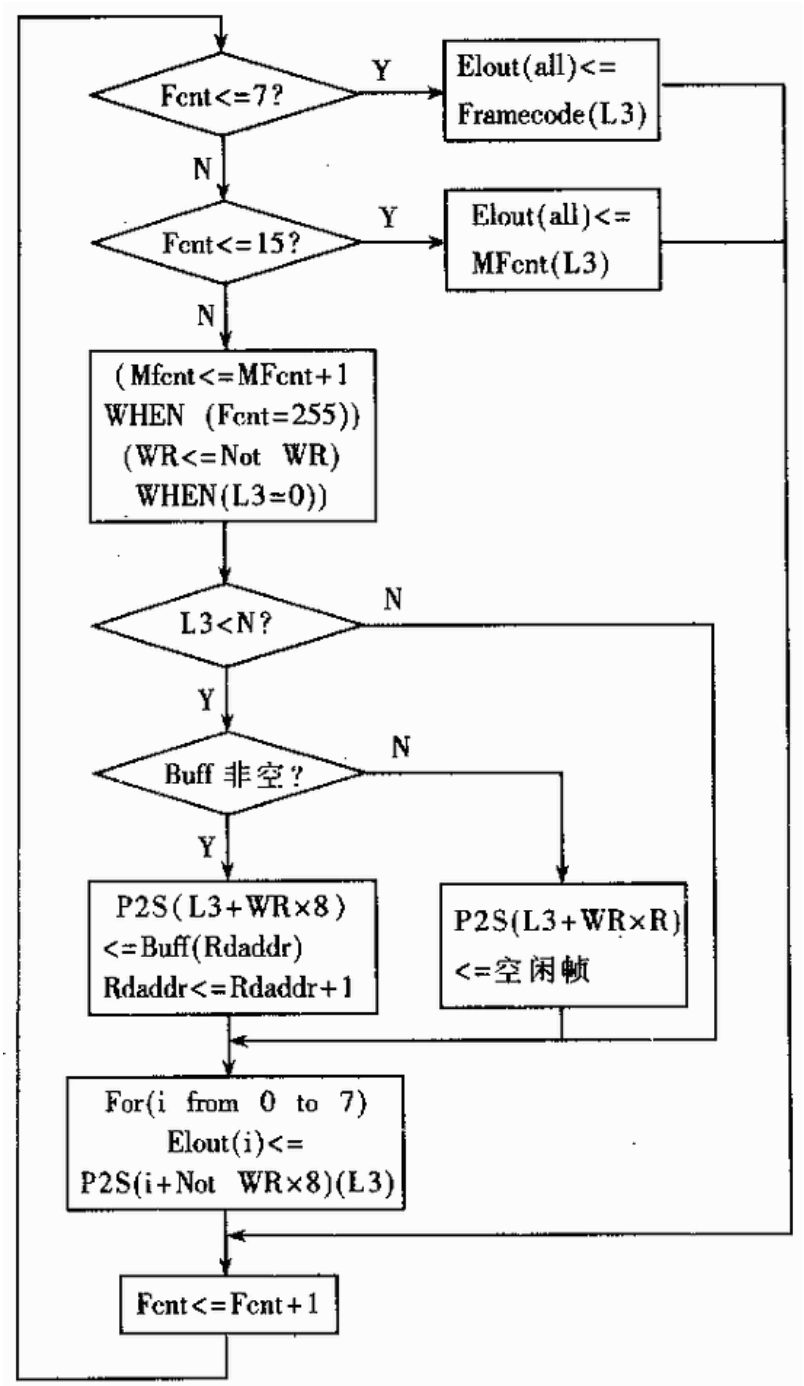

图 1 反向复用流程

适配电路的一侧为 MII 标准以太网 MAC 接口，采用 100MHz/全双工模式及网络变压器 YL18-2050S，另一侧是 8 路 E1（HDB3）差分接口及通信变压器 YL26-2413S 或 YL26-2416S。发送方向将以太网数据封装为 HDLC 帧，反向复用到 1～8 路可配 E1 信道传输，接收方向同步多路 E1，还原出以太网数据。带宽从 2MHz 到 16MHz（1～8 路 E1）可配，接收侧多路 E1 之间可以容纳 16 毫秒的延时。

鉴于目前国内类似产品较少，而 ASIC 开发成本较高，本电路采用 VHDL 编程→FPGA 实现设计流程。

1 反向复用定义

反向复用的基本概念就是把一路高速数据适配到多路低速信道中传输，提供相当于多路低速之和的传输带宽。对于点对点通信，主要关心以下几点性能。

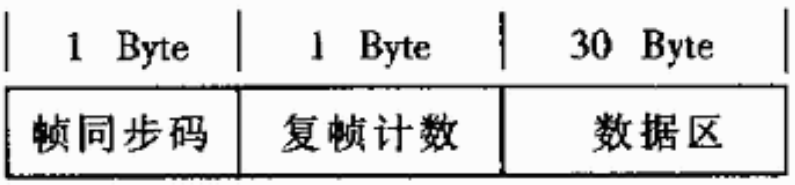

图 2 E1 帧结构

1.1 带宽利用率

低速信道在传输高速数据的同时，必然有附加的开销，会影响带宽利用率。在本设计中，以太网包进入适配电路后封装成 HDLC 帧需要四字节的附加信息。E1 帧在传输 HDLC 数据的同时要携带的同步信息，占用 6.25%带宽。综合上述两点，有效数据平均带宽利用率大约在 90%。

1.2 容纳延时

由于各路低速信道在传输过程中经历的路径不同，到达对端后各路间会有延时，设计中要考虑如何消除这部分延时。

1.3 线路故障处理

算法设计中必需有同步保护状态机，保证传输信道出现错误时，算法不会产生过高的附加误码率，并且在信道错误恢复时，保证数据传输同步恢复。

1.4 数据包突发性

以太网的满发送速率为 100MHz，而低速信道的最大可配带宽为 16MHz，这将导致即使平均流量小于配置带宽，一个短时间内，接收的数据量还有可能远大于传输带宽。所以在以太网数据的输入侧必须设计高容量的缓存队列以容纳一定的突发数据包。

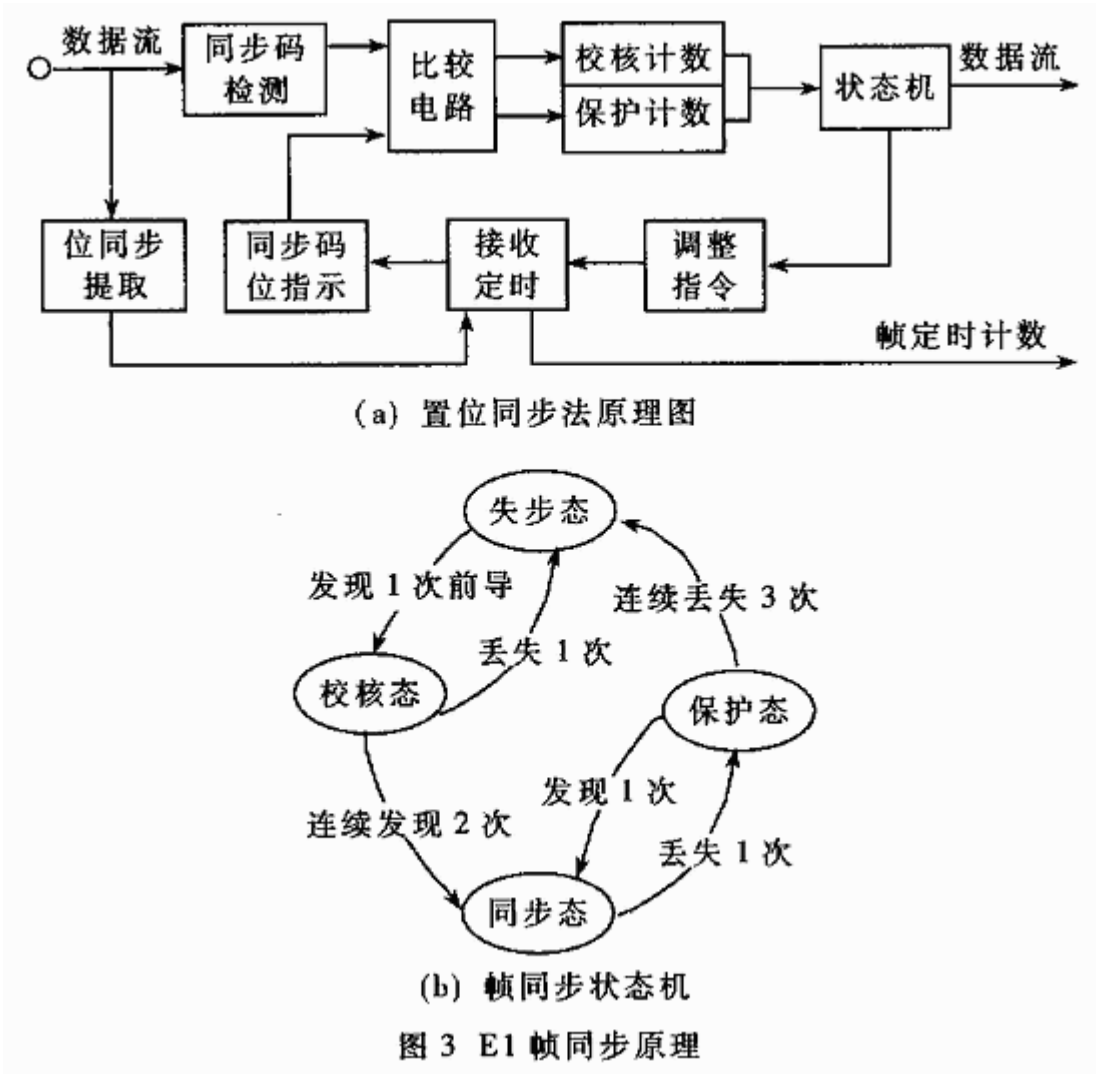

图 3 E1 帧同步原理

2 算法设计

以下就设计中一些核心电路的算法做具体说明。

2.1 以太网数据→多路 E1 反向复用

数据反向适配一般可以采用三种方式：包间插、比特间插和字节间插。所谓包间插就是指数据包到达后，连续检测各个 E1 信道，在第一个查到的空闲信道上顺序传输整个数据包，下一包到达后再重复这一过程。这种方式的优点是设计简单，在对端也不需要对多路 E1 进行同步，各路数据可单独处理。缺点是由于各路 E1 传输过程中经历延时不同，导致对端接收数据包的顺序与发端顺序有较大差别。考虑到发包比较稀疏的情况，一个长包完全可以在一路 E1 中传递，而其它 E1 通道没有数据包传送。这样，一方面造成带宽浪费，另一方面也引入较大的转发延时。

比特间插在传输过程中没有包的概念，只是顺序从以太网数据输入缓存区内读出比特流并按 1→n(n 路 E1)循环编号，编号为 i 的比特在第 i 路 E1 中传输。这种设计电路十分复杂，对端要对多路 E1 同步到比特单位才能还原出有效数据。其优点就是没有带宽的浪费和输入输出包顺序的变化，转发延时也是固定的。

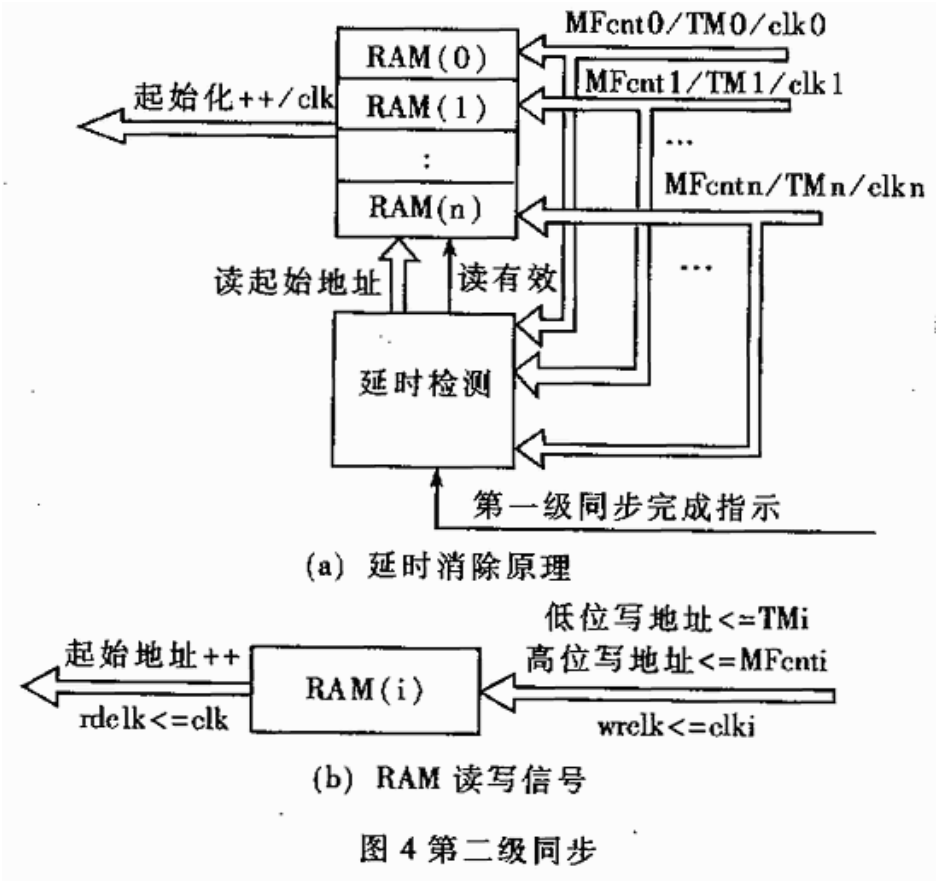

图 4 第二级同步

本设计采用的字节间插是比较折中的解决方案。它的基本原理与比特间插类似，但从缓存区读出的数据和编号都是以字节为单位，每个字节经过串并转后在对应编号的E1信道中传输。它继承了比特间插的优点，同时由于对端只需同步到字节单位上，处理时钟较为宽裕，同步电路设计也就相对简单。其具体的算法实现如下：

以太网数据包进入适配电路后封装成HDLC帧存入缓存区Buff，Buff的出口侧速率与多路E1信道传输速率匹配，一个2MHz时钟周期内完成的操作如图1所示。

图中，

Fcnt：8比特帧计数（E1帧有256个bit）；

MFcnt：8比特复帧计数；

L3：Fcnt的低三位比特；

E1out：输出8路E1数据（8bit）；

Framecode：E1帧同步码“10011011”；

N：配置E1路数（1～8）；

Rdaddr：Buff读地址；

P2S：2×8字节并串转换存储区，写一组8字节时，读另一组8字节；

WR：P2S读写区域指示（0，1）。

输出的E1帧格式如图2所示，帧同步码和复帧计数都是为接收端提供同步信息。

2.2 E1 接收数据同步

E1 接收侧要完成的功能是从多路 E1 数据中还原出以太网数据包。简单看就是发送侧反向复用的逆过程，通过高速时钟循环从 1→n 路 E1 中各读出一字节合成一路数据。但由于各路 E1 在传递过程中经过延时不同，同一时刻到达字节不对齐，在合并前必须对多路 E1 进行同步。考虑到同步过程的复杂性，在设计算法时采用了分步处理，以降低复杂度。

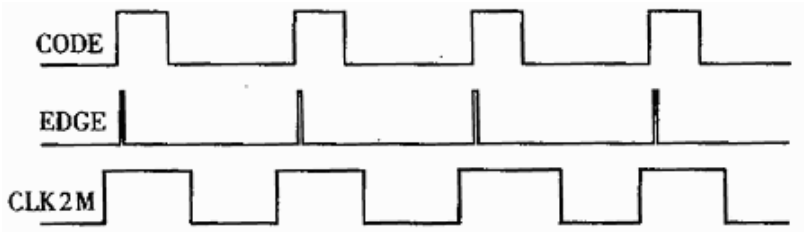

图 5 HDB3 时钟提取原理

第一步处理是根据帧结构中帧同步码，通过置位同步法完成单路 E1 帧同步，同步原理图和状态机如图 3 所示。同步后每路输出数据产生独立的帧定时计数器 TM，从 0→31 标记 E1 帧对应的 32（256bit）字节，其中第 0 字节即帧同步码 Framecode，第一字节即复帧计数器 MFcnt。这一步并未涉及多路间的延时消除。

根据状态转移图确定的状态参数，α（搜索保护帧数）=3、β（同步保护帧数）=4。设线中误码率 Pe=10-3，L（帧同步码长度）=8，Ts（E1 帧周期）=125μs，得出同步机的性能指标：

平均同步时间≈[1/（L×Pe）β-(α-1/2)] ×Ts≈8.5 小时

同步失帧误码率≈1/2（α-1/2）（L×Pe）β≈5×10 -9

可见在较高的误码率下（10 -3），同步机还是能够保证比较好的同步质量，平绝每 8.5 小时出现一次失步，引入的附加误码率也只有 10 -9 量级。

第二步处理根据 E1 帧同步产生的帧计数和每感数据中的复帧计数消除各之间的延时及时钟相位差。其基本原理见图 4。在第 0 路数据延时最小的情况下，设第一步完成同步后到达 RAM（0）的数据为第 M 复帧第 T 字节，数据以各自计数为写地址存入 RAM，延时检测通过计数不停检测，直到所有路第 M 复帧 T 字节到达后，再统一以 clk 为读时钟，以 M&T 为起始地址顺序从 RAM 中读出数据。由于 MFcnt 最大为 255,第二步同步能容纳的线路时延在（-128，+128）帧之间，超过这个范围 MFcnt 所代表的时延就可能是（n(任意整数) ×128+MFcnt）帧，所以该算法能承受的最大线路延时为：

128×Ts（帧周期）=16ms

在线路误码率较低时，E1 帧同步丢失较少，帧定时计数 TM 可以确定，影响第二步处理的只有数据中的复帧计数字节。在没有保护的条件下，假设目前某一路 MFcnt 由于误码使其读数由正确值 A 变为 B，因为下一轮开始时 A、B 位置帧会重写，不会对数据整体逻辑产生较大影响，其它位置的误码会在后续的 HDLC 帧同步处理中检测。如果线睡误码率较高，由于 E1 会频繁失步使第二步处理根本无法进行。出于述考虑该步算法设计没有采用状态机保护。

2.3 HDB3 时钟提取

从接收的 E1 信号 HDB3 编码中提取时钟的原理如图 5 所示。首先通过一个高速时钟采样 HDB3 的码流（CODE）得到数据变化沿（EDGE），再根据 EDGE 位置由高速时钟分频出对应的 2MHz 时钟。

3 电路设计

整体电路结构如图 6 所示。在输入数据缓存和消延时两部分处理中，由于需要较大存储空间，采用两块外挂的 SSRAM。内部处理以字节为单位，全同步电路设计，对应以太网侧处理速率为 12.5MHz，对尖 E1 侧时钟为 256kHz（2.048MHz/8）。两侧速率匹配通过高速时钟采样低速时钟完成。

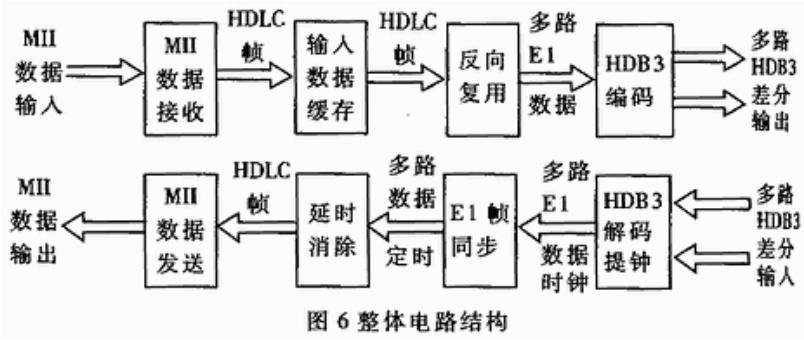

图 6 整体电路结构

4 时序分析

从逻辑验证到 FPGA 实现主要区别是增加了实现布线、引脚间的时延。使系统失效的时序问题主要有以下几点：

（1）输入经内部逻辑到输出的建立时间、保持时间和引脚时延大于一个时钟周期。

（2）并行处理的信号彼此之间时延过大，不能同时采样。

（3）在内部对时钟信号进行过多操作，引入时钟毛刺。

（4）对同一时钟，既使用上升沿，又使用下降沿触发，使时钟最高频率损失一半。

（5）在交叉时钟域中，直接采样由另一时钟作为触发的信号，引入不确定态。

对于这些常见问题，设计中采用如下相应对策：

（1）所有输入、输出引脚信号都经过时钟采样，减少引脚的时延。

（2）内部信号操作增加 D 触发器，两级触发器之间尽量减少组合逻辑，比较复杂的处理经多个时钟周期完成，减小信号保持时间。

（3）并行逻辑经过相同的处理流程，时分复用完成，并行引脚也尽量分配在一起。

（4）内部处理由单一系统时钟完成，低速时钟经高速时钟采样统一到高速时钟上，减少交叉时钟域。

（5）不对系统时钟进行操作，只使用上升沿触发器。

（6）必需进行时钟转换时，通过双端口 RAM 或 FIFO 完成，不直接操作交叉时钟。

在设计中注意上述问题后，QUARTUS II 的后仿真结果能够很好地满足时序要求，其中系统时钟（12.5MHz）最在能够满足 20.59MHz，2MHz 时钟能够满足 41.03MHz。此结果在实际硬件测试中得到了验证。

5 实际产品性能分析

本文介绍的以太网/多路 E1 适配电路设计已实际应用在华环公司 HOEL-1100 E1/100 Base-TX 适配器中，表 1 是实际产品的吞吐量测试结果。由于以太网吞吐量包括 IEEE802.3 规定的导和 SFD 字节，而这部分信息是固定值，不需要经过 E1 信道传输到对端，所以测试值可能大于实际 E1 信道容量。表 2 是在 8 路 E1 配置下，以 15MHz 速率发包测得的以太网数据两端设备的传输延时。该设计在 APEX II 20K100 器件中占用的逻辑单元为 3608 个（共 4160 个逻辑门）。**表 1 吞吐量测试结果（单位 Mbit/s）**

路数/包长	64	128	256	512	1.24	1280	1518
1	2.44	2.19	2.05	1.99	1.97	1.97	1.97

2	4.85	4.33	4.06	4.00	3.98	3.93	3.93
3	7.25	6.41	6.09	5.93	5.75	5.75	5.75
4	9.57	8.57	8.06	7.97	7.83	7.80	7.80
5	11.88	10.76	10.25	9.88	9.75	9.75	9.75
6	14.47	12.72	12.06	11.75	11.74	11.43	11.43
7	16.82	15.07	14.25	13.87	13.50	13.50	13.50
8	19.27	17.26	16.13	15.75	15.37	15.37	15.37

表 2 以太网口传输时延（单位 μs）

包长	64	128	256	512	1024	1280	1518
Cut Trough	400.4	437.0	516.9	670.4	976.9	1130.1	1278.1
Store and Forward	395.3	426.8	496.5	629.5	895.0	1027.7	1156.7

在开发过程中由于采用高级硬件编程语言→编程器件的设计实现过程，大大缩短了开发周期，增加了硬件设计的灵活性和可移植性，也避免了专用集成电路设计的高风险。采用逻辑仿真与后时序仿真相结合的验证方法，基本可以保证设计的可靠性。基于上述优点，这种开发方式在中小指集成电路开发中已得到广泛的应用。尤其是近年来，硬件方面伴随着微电子工艺的迅速发展，编程器件的集成度正在成倍增长，越来越多的 ASIC 单元如微处理器、专用接口等嵌入编程器件中，使其适用范围更广；软件方面 EDA 开发商提供了众多的 Ipcore 及仿真工具，使得编程过程进一步简化，可靠性也不断增强；在此基础上 SYS On Programmable Chip 技术也开始走向商业化，为编程器件的发展提供了更为广阔的空间