# 基于 FPGA 的高速同步 HDLC 通信控制器设计

陈 晨[1,2]，李志来[1]，徐 伟[1]，金 光[1]
（1. 中国科学院 长春光学精密机械与物理研究所，吉林 长春 130033；2. 中国科学院 研究生院，北京 100039）

**摘要**：高级数据链路控制 HDLC 协议是一种面向比特的链路层协议，具有同步传输数据、冗余度低等特点，是在通信领域中应用最广泛的链路层协议之一。提出实现 HDLC 通信协议的主要模块——CRC 校验模块及‘0’比特插入模块的 FPGA 实现方法。CRC 校验模块采用状态机设计方法，而‘0’比特插入模块是利用 FIFO 实现，为 HDLC 通信控制器的设计提供新的思路。该方法已在 Spartan3s400 开发板上实现，并能正确传输。

**关键词**：HDLC 协议；CRC 校验；‘0’比特插入；FPGA
**中图分类号**：TP302 **文献标识码**：A **文章编号**：1674-6236(2010)08-0175-04

## Design of high-speed synchronous HDLC protocol controller based on FPGA

CHEN Chen[1,2], LI Zhi-lai[1], XU Wei[1], JIN Guang[1]
(1. *Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences, Changchun* 130033, *China*; 2. *Graduate University, Chinese Academy of Sciences, Beijing* 100039, *China*)

**Abstract:** High-level Data Link Control (HDLC)protocol is a bit-oriented synchronous data link layer protocol, it is synchronous and its low degree of redundancy, it is one of the most extensively applied data link control protocols. In this paper, CRC verification module and ‘0’ insert module based on FPGA as the most important function module in the HDLC protocol are introduced. CRC verification module used state machine and ‘0’ insert module used FIFO as the primary module, which offered some new ideas of the design of HDLC protocol controller. The method was realized and correctly transmitted on Spartan3s400 DevKit.

**Key words:** HDLC protocol; CRC verification; ‘0’ bit insert and delete; FPGA

高级数据链路控制 HDLC (High-level Data Link Control)广泛应用于数据通信领域，是确保数据信息可靠互通的重要技术。实施 HDLC 的一般方法通常是采用 ASIC 器件和软件编程等。HDLC 的 ASIC 芯片使用简易，功能针对性强，性能可靠，适合应用于特定用途的大批量产品中。但由于 HDLC 标准的文本较多，ASIC 芯片出于专用性的目的难以通用于不同版本，缺乏应用灵活性。例如 CCITT、ANSI、ISO/IEC 等都有各种版本的 HDLC 标准，器件生产商都还有各自的标准，对 HDLC 的 CRC 序列生成多项式等有不同的规定。况且，专用于 HDLC 的 ASIC 芯片其片内数据存储器容量有限，通常只有不多字节的 FIFO 可用。对于某些应用来说，当需要扩大数据缓存的容量时，只能对 ASIC 芯片再外接存储器或其他电路，ASIC 的简单易用性就被抵销掉了[1]。

FPGA 是现场可编程门阵列，属于超大规模集成电路，具有丰富的系统门、逻辑单元、块 RAM 和 IO 引脚等硬件资源。由于 FPGA 具有重装载功能，可以在其内部灵活实现各种数字电路设计，甚至可以动态改变其内部设计，动态实现不同的功能[2]。

因此，采用 FPGA 实现 HDLC 是一种可行的方法。HDLC 通信控制器主要是对数据进行 CRC 校验、‘0’ 比特插入和加帧头帧尾操作。

## 1 “0”比特插入模块

HDLC 规程规定信息的传送以帧为单位，每一帧的基本格式如图 1 所示。


图 1 HDLC 规程格式

Fig. 1 HDLC protocols format

HDLC 规程指定采用 8 bit 的 01111110 为标志序列，称为 F 标志。用于帧同步，表示 1 帧的开始和结束，相邻 2 帧之间的 F，既可作为上一帧的结束，又可作为下一帧的开始。标志序列也可以作为帧间填充字符，因而在数据链路上的各个数据站都要不断搜索 F 标志，以判断帧的开始和结束[3]。

由于 HDLC 具有固定的帧格式，以 7EH 为帧头和帧尾，为了保证透明传输，即只有帧头和帧尾出现连续的 6 个‘1’，其他地方不允许有连续 5 个以上的‘1’出现，否则就要进行‘0’比特插入，即只要遇到连续 5 个‘1’，就在其后插入 1 个‘0’。根据传输数据量的大小可采用以下 2 种思路实现‘0’比特插入操作。

收稿日期：2009-12-04 稿件编号：200912011
作者简介：陈 晨(1986—)，女，山西长治人，硕士研究生。研究方向：FPGA 嵌入式系统。

### 1.1 遇‘0’缓冲实现法

由于数据中出现多少个连续的‘1’是不可控的，故最终插入‘0’的个数也是不可控的。例如要发送的“有效数据”（包含地址字段、控制字段、信息字段、帧校验字段）为 320 bit，则最多会插入 64 个‘0’。数据是串行输入，每插入 1 个‘0’，则数据由 5 位变成 6 位，则插‘0’后要输入的数据就被“积压”来，插入的‘0’越多，“积压”的数据就越多。如果采用文献[4]中的插‘0’方法，简单的将‘0’插入，将会丢失 1 位数据，设置 1 个 64 位的缓冲，每插入 1 个‘0’就把后面数据做为 1 位延时，插‘0’后在把已经延时 1 个时钟周期的数据加进来，就保证不丢失数据。实际利用 VHDL 语言编译时，其 VHDL 代码为：

```
if(clk' event and clk='1')then
  b<=((std_logic_vector'(b(3),b(2),b(1),b(0),datain)));
  for i in 64 down to 1 loop
  a(i)<=a(i-1);
  a(0)<=datain;
  end loop;
  if(b="11111")then
    dataout<='0'; i<=i+1;
  else
    dataout<=a(conv_integer(i(6 downto 0)));
  end if;
end if;
```

上述 VHDL 的思路：矢量 $a$ 的第 64 位到第 1 位分别为 datain 延时 64 个时钟周期的串行数据到延时 1 个时钟周期的串行数据，i 的初始数据为 0，每遇到连续 5 个‘1’，插入 1 个‘0’后，dataout 输出为 datain 延时 i 个时钟周期的数据。这样就做到了不丢失数据。图 2 是利用 ISE 9.1i 仿真的波形图。

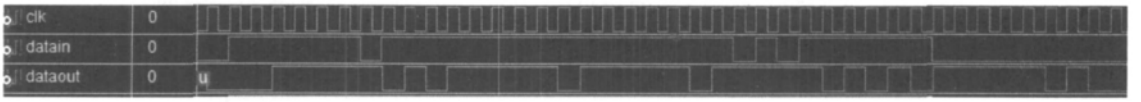


图 2 遇‘0’缓冲法仿真图

Fig. 2 Simuration of meating ‘0’ buffering

由图 2 可看到插‘0’操作后，数据比未插‘0’前变长了，而且变长了多少位是由数据内容决定的。

该方法编程简单，占用 FPGA 资源少，在一个模块内就能完成‘0’比特插入操作。

### 1.2 利用 FIFO 实现

遇‘0’缓冲实现法在传输大容量数据时，需要设置许多位缓冲，这样就耗费大量的 FPGA 内部资源，而且随着延时位数增加，门延时呈指数增长，累积到一定程度就会产生误差，所有当数据量大时，上述的方法就不再适用，可以利用 FIFO 实现。

当数据量大时，“积压”的数据相应也变大，可以利用 FPGA 内部资源 FIFO 节省逻辑资源，提高逻辑速度。选择异步 FIFO，即读/写时钟不是同一个，这样可高速写入数据，再通过控制读时钟控制读的信息。

利用 FPGA 实现的 VHDL 代码为：

```
if(clk'event and clk='1')then
y<=((std_logic_vector'(y(3), y(2), y(1), y(0),
datain)));
if(y="11111")then
dataout<='0';
y<="00000";
a<='0';
else
dataout<=datain;
a<='1';
end if;
end if;
clkout<=clk and a;
```

其基本思想是，一旦遇到 5 个连续的‘1’，就“抹掉”1 个时钟，利用 ISE 9.1i 仿真的波形图如图 3 所示。

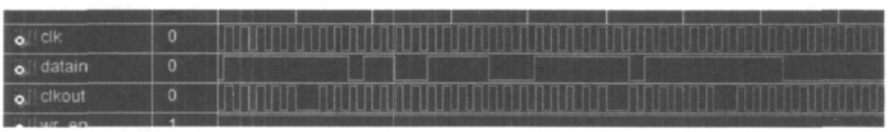


图 3 对时钟处理的仿真图

Fig. 3 Simulation of clock processing

设计一个 FIFO 与上述 VHDL 代码产生的模块相连，电路图如图 4 所示。

利用 ISE 9.1i 仿真得到的波形图如图 5 所示。可看到对 datain 进行了‘0’比特插入操作，保证数据不丢失。而且该方法可根据所选器件的片内资源设置任意大容量的 FIFO，并且当片内 FIFO 的存储量不够时，可先存入一部分数据，等 FIFO 读取一部分后，不满时再存入一部分数据。

‘0’比特删除操作是‘0’比特插入操作的反过程。在接收时为了还原原本的信息，就要删除发送时插入的‘0’。以逐位延时法为例，dataout 最一开始输出延时了 64 个时钟周期的

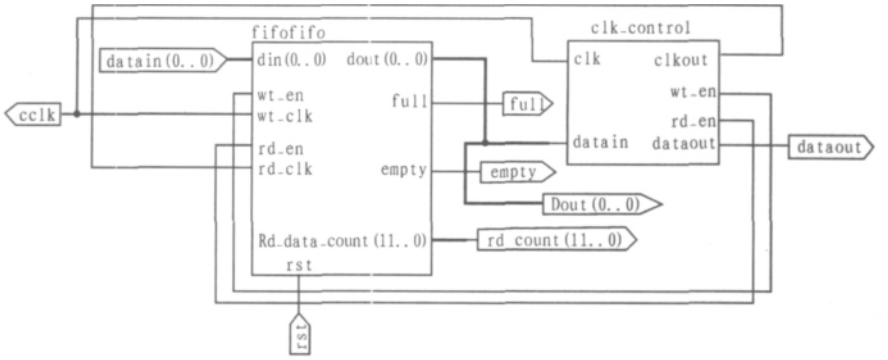


图 4　利用 FIFO 方法的电路图

Fig. 4　Circuit diagram of FIFO method

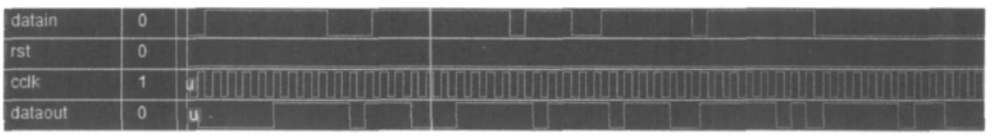


图 5　利用 FIFO 实现的仿真图

Fig. 5　Simulation FZFO method realization

串行数据,i 的初始值为 64,当遇到‘11111’时,i 减 1,输出延时了 i 个时钟周期的串行数据。而利用 FIFO 的方法就是遇到‘11111’,抹去 1 个写时钟,将数据写入 FIFO,再按规定的时钟把数据读取,当然写入的时钟可用较高的时钟周期。

## 2　CRC 校验模块

帧校验字段用于对帧进行循环冗余校验,校验的范围从地址字段的第 1 个比特到信息字段的最后 1 个比特,但为了透明传输而插入的‘0’比特不在校验范围内。

CRC 原理实际上就是在一个 $p$ 位二进制数据序列之后附加一个 $r$ 位二进制校验码，从而构成一个总长为 $n=p+r$ 位的二进制序列,例如,$p$ 位二进制数据序列 $D=[d_{p-1}d_{p-2}...d_1d_0]$，$r$ 位二进制校验码 $R=[r_{r-1}r_{r-2}...r_1r_0]$，所得到的二进制序列就是 $M=[d_{p-1}d_{p-2}...d_1d_0\ r_{r-1}r_{r-2}...r_1r_0]$，附加在数据序列之后的这个校验码与数据序列的内容之间存在着某种特定的关系。如果因干扰等原因使数据序列中的某一位或某些位发生错误,这种特定关系破坏,因此,通过检查这一关系,实现对数据正确性的检验[5]。

要传输 $p$=16 位数据 1001011010101011,选定的 $r$=16 的校验序列为 10001000000100001，对应的 FCS 帧校验列是用 10010110101010110000000000000000(共 $p+r$=32 位)对 2 取模整除以 10001000000100001 后的余数 1010100011000001(共有 $r$=16 位)。因此，发送方应发送的全部数据列为 10010110101010111010100011000001。接收方将收到的 32 位数据对 2 取模整除以 $r$ 校验二进制位列 10001000000100001,如余数非 0,则认为有传输错误位。

而多项式乘除法运算过程与普通代数多项式的乘除法相同。多项式的加减法运算以 2 为模,加减时不进位或错位,和逻辑异或运算一致,即加法和减法等价。则对上述例举的数据的 CRC 计算过程如图 6 所示。


图 6　CRC 计算实例

Fig. 6　CRC calculating instance

模拟上述计算 CRC 校验值的方法，不难想到可用状态机实现,设置一个 17 位的矢量,检验最高位是否为零。如果为零,则跳转到状态 1,即所有位左移,最低位补 1 位数据;如果不为零，则跳转到状态 0，最低位补 1 位数据,与“00010000001000010”异或,(以 CRC-CCITT 为例,由于 y16 与‘1’异或必为‘0’,datain 与‘0’异或还为 datain),这种思路的 VHDL 代码如下:

```
IF ( y16='0' ) THEN
next_sreg<=STATE1;
y <= (( std_logic_vector'(y15, y14, y13, y12, y11, y10,
y9, y8, y7, y6,y5, y4, y3, y2, y1, y0, datain)));
ELSE
next_sreg<=STATE0;
y<=((std_logic_vector'("00010000001000010")xor(( std_
logic_vector' (  y15,y14,y13,y12,y11,y10,y9,y8,y7,y6,y5,
```

```
y4,y3,y2,y1,y0,datain)))));
    END IF;
```

图 7 为该 VHDL 代码的仿真波形，可看到该方法模拟对 2 取模整除的一步步计算。

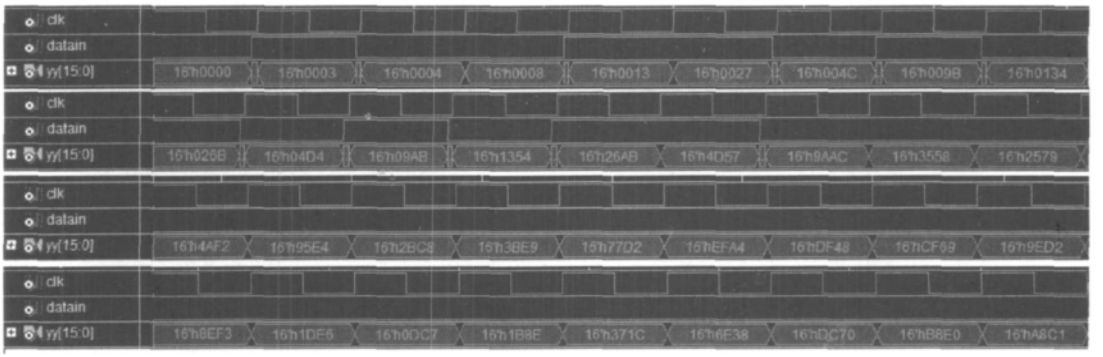


图 7　CRC 校验仿真图

Fig. 7　Simulation of CRC check

该方法思想简单，是对 2 取模整除方法的模拟，直观，易于理解，由于是串行输入，不受需要 CRC 计算的数据位数限制。由于 HDLC 通信协议的最大优点是对要传输的信息文电比特结构无任何限制[6]，也就是说，信息文电可以是任意的比特串，不会影响链路的监控操作。因此，这里给出的 CRC 串行算法符合 HDLC 传输文电比特结构任意的特点。

## 3　程序加载验证

经过逻辑综合和时序仿真后，利用 ISE 9.1i 集成开发软件将程序烧入 FPGA，利用示波器观测 FPGA 按 HDLC 通信协议标准发出的信号。如要发送的“有效信号”（不含帧头帧尾，未进行 CRC 校验及‘0’比特插入之前的原始数据）为“1111 1111”，则经过 FPGA 处理后应发出的数据为“0111 1110 1111 1011 1000 1111 0111 1000 0011 1111 0 ”，利用示波器检测到的信号波形如图 8 所示。

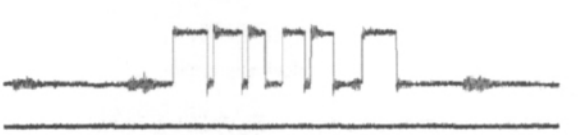

图 8　利用示波器检测到的波形

Fig. 8　Detection waveform of oscilloscope

由图 8 可知，对数据进行 CRC 校验，‘0’比特插入及加帧头帧尾操作，发送数据正确，符合 HDLC 通信协议标准。

## 4　结束语

由于 HDLC 通信协议具有透明传输、可靠性高等优点，在数据链路层应用广泛，而 FPGA 更具有灵活、高性能、低成本、平台化、可定制等优点，具有系统级能的复杂可编程逻辑器件和现场可编程门阵列实现可编程片上系统也成为今后的发展方向。本文提出的基于 FPGA 实现 HDLC/SDLC 协议方法采用 ISE 9.1i 编译、综合、仿真、布线、烧写，ISE 软件支持器件多，功能强大，操作更方便，因此，该实现方法具有很强的实用性，另外，程序加载入 FPGA 后发送数据正确，说明该实现方法实用、有效。

**参考文献**


[1] 王喜，吴祖民，魏武.HDLC 的 FPGA 实现方法[J].通信与广播电视，2005(3)：23-29.
WANG Xi,WU Zu-min,WEI Wu.Realization of FPGA with HDLC[J]. Communication & Audio and Video,2005(3):23-29.

[2] 应三从，张行.基于 FPGA 的 HDLC 协议控制器[J].四川大学学报：工程科学版，2008，40(3)：116-120.
YING San-cong,ZHANG Xing. New HDLC protocol controller based on the FPGA [J]. Journal of Sichuan University: Engineering Science Edition;2008,40(3):116-120.

[3] 娄景艺，王鲁平，李飚.HDLC 控制协议的 FPGA 设计与实现[J].电子设计工程，2005(5)：64-66.
LOU Jing-yi, WANG Lu-ping, LI Biao. Design and implementation of a HDLC protocol controller based on the field programmable gate arrays [J].Electronic Design Engineering, 2005(5):64-66.

[4] 王鲁平，李飚，胡敏霞.一种基于 FPGA 的 HDLC 协议控制器[J].电子产品世界，2003(11)：13-14.
WANG Lu-ping, LI Biao, HU Min-xia. A HDLC protocol controller based on FPGA[J]. Electronic Engineering & Product World,2003(06A): U013-U014.

[5] 范红旗，王胜，祝依龙.CRC 编解码器及其 FPGA 实现[J].数据采集与处理，2006，12(2)：97-100.
FAN Hong-qi, WANG Sheng, ZHU Yi-long.CRC coder-encoder algorithm and its FPGA implementation [J].Journal of Data Acquisition & Processing, 2006,21(B12):97-100.

[6] 李晓娟，黄翌.基于 FPGA 的 HDLC 设计实现[J].现代电子技术，2007(6)：35-37.
LI Xiao-juan，HUANG Yi. HDLC design realization based on FPGA[J]. Modern Electronics Technique, 2007,30(6): 35-37.