

文章编号: 1005-8451 (2006) 10-0001-04

## 基于FPGA/CPLD的高速和低速UART的设计及其应用

王永州, 范多旺

(兰州交通大学 光电技术与智能控制教育部重点实验室, 兰州 730070)

**摘要:** 利用计算机软件技术(EDA技术)和FPGA/CPLD的灵活性可以方便快速地设计高速和低速的UART。高速的UART可以用在光纤通信上, 低速的UART可以用在FPGA/CPLD和单片机的通信上。设计中包含UART的发送模块、接收模块和波特率发生器, 所有功能的实现全部采用VHDL硬件描述语言来进行描述。设计、综合、仿真在QUARTUS II软件开发环境下实现。

**关键词:** UART; VHDL; FPGA/CPLD; 设计

**中图分类号:** TP273; TN402 **文献标识码:** A

### Design and application of high speed and low speed UART based on FPGA/CPLD

WANG Yong-zhou, FAN Duo-wang

(Automatic Control Research Institute, Key Laboratory of Opto-Electronic Technology and Intelligent Control Ministry of Education, Lanzhou Jiaotong University, Lanzhou 730070, China)

**Abstract:** Using EDA and the agility of FPGA/CPLD could easily and quickly design the high speed and low speed UART. High speed UART could be used on the optical fibre communication and low speed UART could be used on the communication of FPGA/CPLD with SCM. This design included the transmitter module, the receiver module and the Baudrate Generator. All function were described by VHDL. The implementation of design, simulation and synthesis were completed by the development software of QUARTUS II.

**Key words:** UART; VHDL; FPGA/CPLD; design

FPGA(Field-Programmable Gate Array,现场可编程阵列)在现代数字电路设计中发挥着越来越重要的作用。从设计简单的接口电路到设计复杂的状态机,甚至设计System On Chip(片上系统),FPGA所扮演的角色已经不容忽视。

随着EDA技术的发展,FPGA/CPLD已经在许多方面得到了广泛应用,而UART(Universal Asynchronous Receiver Transmitter 通用异步收发器)是广泛使用的串行数据传输协议。UART允许在串行链路上进行全双工的通信。串行外设用到RS232-C异步串行接口,一般采用专用集成电路即UART实现,常见的有芯片8250、8251等,但这些芯片速度较慢,难以满足一些需要高速应用的场合,虽然新型的UART如16550、16750,可以满足高速的要求。但是所有的UART芯片都存在引脚较多、体积较大、与其它器件的接口较为复杂的缺点。因此,利用VHDL语言将需要的UART的核心功能集成到FPGA/CPLD内部,不仅解决传统芯片的缺点,也使整个设

计更加紧凑、稳定且可靠。

### 1 UART的数据传输协议

UART发送/接受数据的帧格式如图1所示。包括线路空闲状态(idle,高电平)、起始位(start bit,低电平)、5 bit~8 bit数据位(data bits)、校验位(parity bit,可选)和停止位(stop bit,位数可为1, 1.5, 2 bit)。这种格式是由起始位和停止位来实现字符的同步。UART内部一般有配置寄存器,可以配置数据位数(5 bit~8 bit)、是否有校验位和校验的类型、停止位的位数(1, 1.5, 2)等设置。

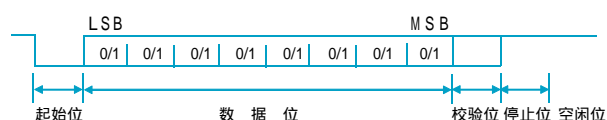


图1 基本UART的帧格式

UART的数据传输协议可以根据需要而改变,本设计中采用的协议是:总线由高到低跳变,宽度为一个波特率时间的负脉冲为开始位,然后是8 bit

收稿日期: 2006-05-10

作者简介: 王永州,在读硕士研究生;范多旺,教授。

的数据位,数据位后面是1 bit的停止位。

## 2 低速 UART 的设计

### 2.1 UART 发送模块

发送部分的用途是将准备输出的并行数据,按照图1的基本帧格式,转为txd信号串行输出,这里的串行化时的时钟信号只要使用对应要求的波特率的本地时钟即可。

CPU要写数据到发送缓冲寄存器(Tbuff)时,必须判断当前是否可写得状态,很明显,如果不判断此条件,发送的数据就会出错,当CPU写入数据到Tbuff后,Tbuff的数据会传送到发送移位寄存器(TReg)并移位。

发送模块的功能可以由以下程序实现:

```
process(clk,reset,enable,load,data0,Tbuff,TReg,
tmpTRegE,tmpTBuffE)
    variable tmp_TRegE: std_logic;
    constant CntOne : Unsigned(3 downto 0) :
="0001";
begin
    if clk'event and clk='1' then 在时钟clk的上升沿
端触发执行下列数据转移
        if reset='0' then          重设reset=0时
            tmpTRegE<='1';        令临时 tmpTRegE=1
            tmpTBuffE<='1';       令临时 tmpTBuffE=1
            TxD<='1';             处于1的停止态
            bitcnt<="0000";       发送位计数器为00位
        elsif load='1' then       若是开始送入数据(load=1)
            Tbuff<=Data0;        将外部Data0送入Tbuff内
            tmpTBuffE<='0';      令缓冲存储器有数据非空着
        elsif Enable='1' then    若是启动开始传输数据
            if (tmpTBuffE='0') and (tmpTRegE='1') then 若是在
Tbuff 内容数据未转送入 TReg 的状态下
                TReg<=Tbuff;    Tbuff送入TReg
                tmpTRegE<='0';   令tmpTRegE='0'告知
TReg 有数据准备发送
                tmpTBuffE<='1';  tmpTBuffE='1'告知Tbuff
没有数据可送入
            elsif tmpTRegE='0' then 表明已有数据则开
始发送
                case bitcnt is    开始作发送位的判别计数
                    when "0000"=> 当计数位为0时
```

```
                TxD<='0'; 开始发送TxD=0启动信号
                bitcnt<=bitcnt+CntOne; 令计数器加1
            when "0001"|"0010"|"0011"|"0100"|"0101"|"0110"|"0111"|"
"1000"=>    当计数位处于1-8时
                TReg<='1'&TReg(7 downto 1); TReg以1
右移
                TxD<=TReg(0); 将TReg(0)的内容输出
到 txd 发送
                bitcnt<=bitcnt+CntOne; 令计数器加一
            when "1001"=>    当位计数器处于9的第9位发送时
                TxD<='1';   TxD=1发送停止信号
                TReg<='1'&TReg(7 downto 1);
Treg 以 1 右移
                bitcnt<="0000";    位计数器
bitcnt=0;
                tmpTRegE<='1'; Treg已发送完成
                let tmpTRegE=1
                when others => null; 在其他态则不作
任何运算
            end case;
        end if;
    end if;
end if;
end process;
```

发送模块仿真波形如图2所示。

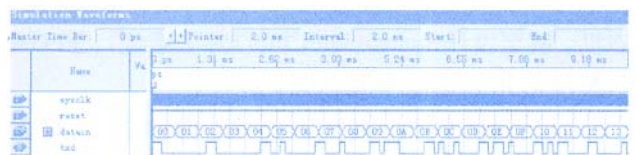


图2 发送模块仿真波形图

### 2.2 UART 接收模块

UART接收部分接收 rxd 串行信号,并将其转化为并行数据。但串并转换的时钟同发送模块一样处理。

串行数据帧和接收时钟是异步的,发送来的数据由逻辑1变为逻辑0可以视为一个数据帧的开始。接收器先要捕捉起始位,确定 rxd 输入由1到0,逻辑0要8个CLK16时钟周期,才是正常的起始位,然后在每隔16个CLK16时钟周期采样接收数据,移位输入接收移位寄存器ShtReg,最后输出数据Dout。还要输出一个数据接收标志信号DRdy标志数据接收完。

对于接收同样存在9、10、11位3种串行数据长度的问题,必须根据所设置的情况而将数据完整地

取下来。接收还有一个特别的情况,那就是它的移位时钟不是一直存在的,这个时钟必须在接收到起始位的中间开始产生,到停止位的中间结束。接收到停止位后,必须给出中断,并提供相应的校验出错、FRAME 错以及溢出等状态。

### 2.3 波特率发生模块

UART 的接收和发送是按照相同的波特率进行收发的(当然也可以实现成对的不同波特率进行收发)。波特率发生器产生的时钟频率不是波特率时钟频率,而是波特率时钟频率的16倍,目的是为在接收时进行精确地采样,送出异步的串行数据。根据给定的晶振时钟和要求的波特率算出波特率分频数。即假定当前按照9 600 bps进行收发,那么波特率发生器输出的时钟频率应为 $9600 \times 16 \text{ Hz}$ ,当然这也是可以改变的。

假定提供时钟为40 MHz,如果希望输出9 600 Hz的波特率,那么 $40\,000\,000 / (26 \times 10) = (16 \times 9\,615)$ ,误码率 $< 0.01\%$ ,也就是说40 MHz的时钟周期经过26 F和10 F之后就得到了接收端需要的时钟频率 $\text{EnableRx} = 9600 \times 16 \text{ Hz}$ ,而发送端需要的时钟频率就是 $\text{EnableTx} = 9600 \text{ Hz}$ 。

## 3 低速UART的应用

随着基于FPGA的NIO S II 嵌入式处理器的出现,低速UART(波特率300~115 200)在集成电路中的应用越来越广泛。特别是把UART模块及其它一些组件和NIO S II 嵌入式处理器集成在一片FPGA芯片上的方案成为许多厂家设计新型产品的首选解决方案。基于NIO S II 嵌入式处理器的GPS/GSM车辆监控系统就是利用集成在NIO S II 处理器中的低速UART和GPS模块及GSM模块进行通信的。系统综合设计的硬件组成框图如图3所示。系统设计的整体框图如图4所示。

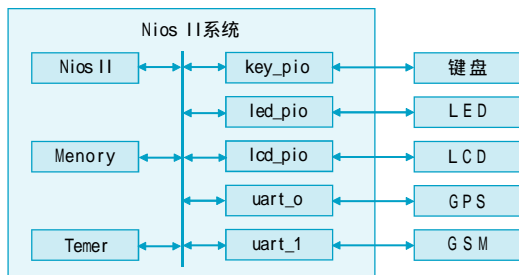


图3 车辆监控系统硬件组成

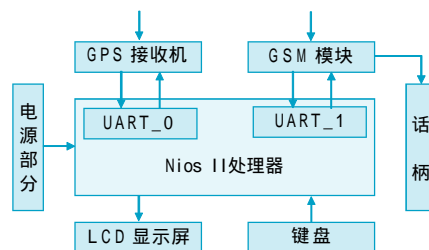


图4 车辆监控系统整体框图

## 4 高速UART的设计及应用

高速UART和低速UART在设计的结构上是相同的,在低速UART的基础上利用FPGA/CPLD可以设计传输速度为1 M~10 M甚至传输速度更高的UART。要设计高速的UART,必须使用FPGA内部的锁相环PLL,经过PLL倍频后最高可输出320 MHz的时钟频率。由于FPGA为高速运作的芯片,所以在设计中为FPGA提供的外部时钟一般不应低于50 MHz。下面利用50 MHz的外部时钟和FPGA内部的锁相环PLL来设计5 MHz和10 MHz的高速UART。首先利用锁相环PLL把50 MHz倍频到200 MHz,如图5所示,PLL输出的时钟频率sysclk为200 MHz。由于 $200 \text{ MHz} / (5 \times 8) = 5 \text{ MHz}$ ,即200 MHz经过5 F后就得到了接收端的采样频率40 MHz,为了降低干扰,接收端应该以第4或第5个采样点的值来作为实际接收到的值。在设计10 MHz的高速UART时,可以先将50 MHz的外部时钟倍频到300 MHz,则 $300 \text{ MHz} / (5 \times 6) = 10 \text{ MHz}$ ,即发送端的时钟频率为10 MHz,而接收端的采样频率为60 MHz,为了降低干扰,接收端应该以第3或第4个采样点的值来作为实际接收到的值。

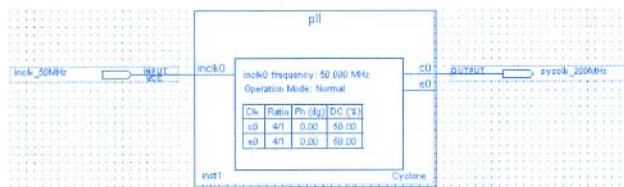


图5 FPGA内部的锁相环PLL

由于低速的信号在光纤传输中会产生很大的误码率,要使用光纤传输串行信号,必须借助于FPGA/CPLD来发送和接收光纤中传输的串行信号。如图6所示的结构为工程设计中的一种常用的解决方案。

文章编号: 1005-8451 (2006) 10-0004-03

## 基于 API 函数的模拟终端串行通信设计

丁 函, 张海明, 安永祥

(兰州交通大学 机电工程学院, 兰州 730070)

**摘 要:** 通过对模拟终端串口通信设计的介绍, 阐述在 Windows 9X/NT 操作系统中, API 函数和多线程技术在串行通信中的应用。该方法能更加灵活、高效地控制串口操作。

**关键词:** 模拟终端; 串行通信; 多线程技术; API 函数

**中图分类号:** TP274+.2; TP311.1 **文献标识码:** A

### Designing of analog terminal serial-port communication based on API function

DING Han, ZHANG Hai-ming, AN Yong-xiang

(School of Mechatronic Engineering, Lanzhou Jiaotong University, Lanzhou 730070, China)

**Abstract:** By introducing the design of analog terminal serial-communication, it was elaborated the application of the API function and multithread technology in serial-port communication in Windows 9X/NT. This method could control the operation of serial-port more flexibly and efficiently.

**Key words:** analog terminal; serial communication; multithread technology; API function

本文介绍的串口通信按照 RS-232 标准串行通信协议与外围设备进行数据通信。

VC++6.0 是 Windows 环境下功能强大的可视化应用程序开发工具, 在这种环境下编写的程序具有设备无关性和可移植性的优点。因此, 本文将介绍基于 VC++6.0 中 Windows API (Windows Application Programming Interface, Windows 应用程序编程接口)

收稿日期: 2006-04-14

作者简介: 丁 函, 在读硕士研究生; 张海明, 副教授。

函数和多线程思想进行模拟终端通信程序设计的方法, 该程序可以实现从外部设备读写数据功能。

### 1 系统整体结构

控制计算机为整个系统重要组成部分, 应用程序通过 API 函数像通常访问文件一样进行通信, 利用串口采集外围设备中数据。系统结构如图 1 所示。

#### 1.1 基于 Win32API 函数的串口创建

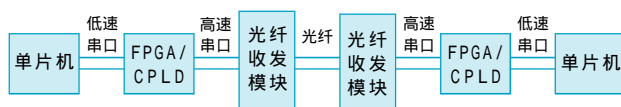


图6 高速串口的应用

### 5 结束语

设计中的逻辑功能模块完全采用 VHDL 语言进行描述, 在 ALTERA 公司生产的 Cyclone 系列的 EP1C12Q240I7 芯片上实现了下载测试, 并制作了用光纤连接两块电路板, 在电路板上测试了 FPGA 和 AVR 单片机之间 9 600 Hz 的串口通信以及两块电路板间利用 FPGA 和光纤收发模块设计的 5 M 和 10 M

的串口通信, 完成了在 FPGA 上实现高速和低速 UART 的全过程。测试的结果验证了 UART 的逻辑功能以及时序关系均符合设计要求。

#### 参考文献:

- [1] 林容益. cpu/soc及外围电路应用设计(基于fpga/cpld)[M]. 北京: 北京航空航天大学出版社, 2004.
- [2] 潘 松, 黄继业. eda技术实用教程 [M]. 北京: 科学出版社, 2002, 10.
- [3] 姚燕南, 薛钧义. 微型计算机原理[M]. (第3版) 西安: 西安电子科技大学, 1994.
- [4] ALTERA. a8251 programmable communication interface datasheet [A]. www.altera.com.cn. 2004/2005.9.