

基于 CPLD 的 UART 设计

Designing UART based on CPLD

(重庆大学) 张莉 杨永明

Zhang, Li Yang, Yongming

摘要: 串行通信在数字通信及控制系统中得到了广泛应用, 本文介绍一种采用可编程逻辑器件 CPLD 实现 UART 的方法, 将 UART 的核心功能集成到 CPLD 上, 使整体设计紧凑, 小巧, 实现的 UART 功能稳定、可靠。所有功能的实现全部采用 VHDL 进行描述。

关键词: 串行通信; CPLD; UART; VHDL

Abstract: Serial communication is widely used in digital communication and control system. This article introduces a method to design UART based on Programmable Logic Device CPLD. The core function is integrated in CPLD, which makes the whole design compact and cabinet, the achieved UART stable and reliable. All functions are given by VHDL.

Keywords: Serial communication; CPLD; UART VHDL

1 引言

随着微机系统的广泛应用和微机网络的极大发展, 串行通信在数据通信及控制系统中得到了广泛应用。常见的串行接口芯片如 8250、16450、16550 等, 这些芯片速度比较慢, 难以满足一些需要高速应用场合, 新型的 UART 如 16650 和 16750, 他们可以满足较高速度的要求。但无论是哪种专用的 UART 芯片, 其引脚较多、体积较大, 与其它器件的接口较为复杂。专用 UART 芯片能够实现比较全面的串行通信功能, 而在实际应用中我们往往只需要使用到 UART 的部分功能, 在设计中如果用到了高密度可编程逻辑器件 (CPLD/FPGA), 那么在剩余资源充足的情况下我们可以充分利用剩余资源实现所需的 UART 的功能, 这样就无需再外接专门的 UART 芯片, 从而简化了电路、缩小了体积、提高了可靠性、并且具有了更大的灵活性。基于以上考虑, 本文提出了一种采用 CPLD 来实现 UART 功能的方法。

2 功能概述

本设计实现的 UART 的功能为: 传送的一帧数据或一个字符包含了一位起始位, 5~8 位数据位, 一个停止位, 停止位后面是不定长度的空闲位; 数据传送的位数 5 位至 8 位四种可选; 波特率可调。传送数据时, 数据的低位在前, 高位在后。传送开始之前, 把采用的字符数据位宽度和数据传输速率作出规定。接收开始后, 接收器不断地检测串行数据输入端 (RXD), 看是否有起始位到来。起始位经确认后就开始接收所规定的的数据位以及停止位。经过处理将起始位和停止位去掉, 把数据位拼装成一个并行字节, 然后送到数据线上。一个字符接收完毕, 接收器又继续检测 RXD 端。发送之前首先将要发送的数据通过数据总线由 CPU 写入发送寄存器, 发送时同样要在数据位前增加一个低电平起始位, 在数据位末增加一个高电平停止位。

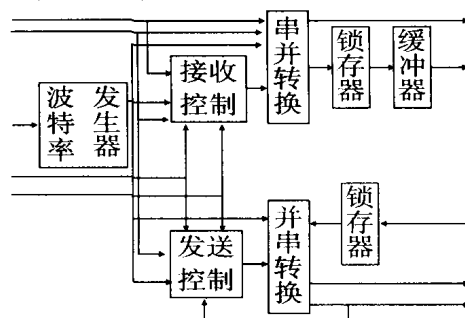


图 1 UART 结构图

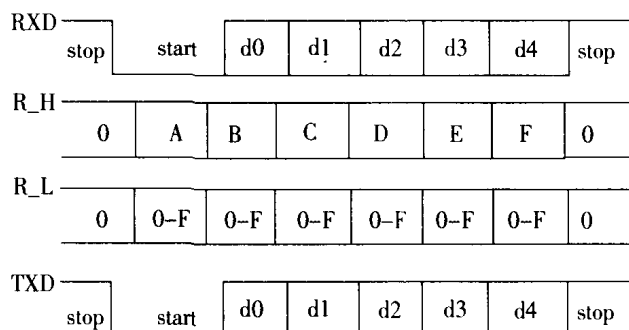


图 2 串行数据收发过程波形图

3 整体结构及各部分功能详述

UART 结构图如图 1 所示, 它是由波特率发生器,

接收和发送控制模块,串并、并串转换器,锁存器和三态缓冲器组成。

1. 波特率的设置

波特率的产生主要是通过一个定时器来完成的,设计的定时器具有自动恢复初值(初值自动再装入)功能,从初值开始到定时器所能计数的最大值重复计数,每次计满溢出信号作为输出即产生了内部时钟信号(INCLK),再将这一信号经过 16 分频就得到了所传送数据的波特率。因此可以通过设置不同的定时器初值改变定时器的溢出率从而达到改变波特率的目的。波特率与定时器的溢出率之间的关系表示为:波特率=定时器的溢出率/16。而定时器的溢出率又可表示为:定时器的溢出率=输入时钟频率/(2^K -计数初值),其中 K 为定时器位数。

2. 串行数据接收采样

为了能对位进行正确的操作,以设定的波特率的 16 倍的速率采样 RXD 端的电平。这就把接收一位的时间等分成 16 份,接收是在每位时间的第 8 个计数状态采样 RXD 的值,由于采样时刻总是在接收位的中间位置,这样可以避开信号两端的边沿失真。

3. 串行数据收发的控制

(1) 串行数据接收控制

串行数据接收控制功能的实现是通过一个 8 位的计数器 COUNT 来完成的,计数器的高 4 位为 R_H,低 4 位为 R_L。计数器的状态与串行数据接收过程的波形关系如图 2 所示,以 5 位数据位为例。

RXD 端的起始位作为数据接收的启动信号,在起始位到达以前计数器的 R_H 和 R_L 都保持为“0”。因为停止位和空闲位都规定为高电平(逻辑 1),这样就保证起始位开始处一定有一个下降沿。当接收控制器在收到一系列的“1”(停止位或空闲位)之后,检测到一个下降沿,说明起始位出现。若数据传送位数控制线 A1、A0 选择的传送位数是 8 位则立即将 R_H 置为 7H,是 7 位则立即将 R_H 置为 8H,是 6 位则立即将 R_H 置为 9H,是 5 位则立即将 R_H 置为 AH;R_L 在 4 种情况下均置为 0H。此后计数器启动,每 16 个时钟周期接收一位数据,当计数到 R_H 和 R_L 均为 FH 时,一个数据接收过程结束,计数器又翻转到 0 状态,等待下一个起始位的到来。

(2) 串行数据发送控制

串行数据发送控制过程中启动发送数据的条件不是起始位信号,而是发送锁存器状态标志 TEMPTY

和写信号 WR。当前一个数据发送完毕将 TEMPTY 置“1”,CPU 就可将下一个待发送的数据写入发送锁存器,TEMPTY 置“0”,启动发送。串行数据发送控制具有与串行数据接收控制类同的工作过程,此处不再复述。

4 仿真结果

本设计完全采用 VHDL 语言进行描述。实现的 UART 设置了时钟输入 CLK、复位输入 RESET、串行数据输入 RXD、读写信号输入 RD 和 WR、片选输入 CS、传送数据位数选择输入 A1 和 A0、定时器初值输入 COUNT、串行发送数据输出 TXD、接收寄存器状态输出 RDFULL、发送寄存器状态输出 TEMPTY 以及数据总线 DATA。

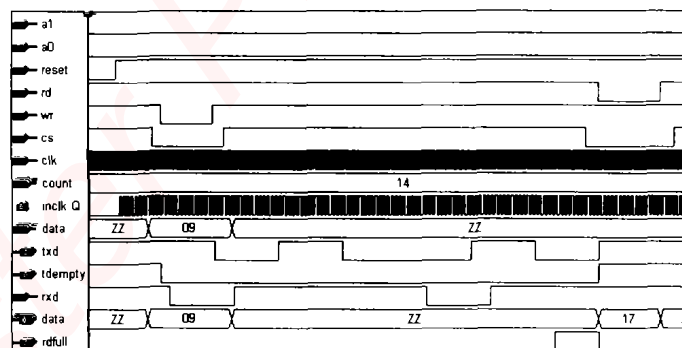


图 3 UART 模块仿真波形图

用 MAX+PLUS II 对 UART 模块进行了仿真,仿真波形如图 3 所示。a0、a1 设置的为“00”,选择 5 位数据传送宽度。复位后,发送、接收计数器均清零,TEMPTY 为“1”,RDFULL 为“0”。在 RXD 信号线上设置一串数据“0111011”,在 DATA 上与读脉冲对应处接收到 5 位数据值 17H。从 DATA 上输入数据 09H,在 TXD 上发送一串脉冲“0100101”。仿真表明实现的 UART 工作过程正确。下载到 FLEX10K 芯片中,经试验功能正常。实现的 UART 所需触发器为 48 个,仅占整体资源的一小部分。

参考文献

- [1]刘乐善,叶济忠,叶永坚. 微型计算机接口技术原理及应用. 华中理工大学出版社. 1996
- [2]侯伯亨,顾新. VHDL 硬件描述语言与数字逻辑电路设计. 西安电子科技大学出版社. 2000

作者简介:张莉,女,77 年 1 月生,四川人,目前在重庆大学电气工程学院攻读硕士学位。

(400044 重庆大学电气工程学院)张莉 杨永明

嵌入式资源免费下载

总线协议:

1. [基于 PCIe 驱动程序的数据传输卡 DMA 传输](#)
2. [基于 PCIe 总线协议的设备驱动开发](#)
3. [CANopen 协议介绍](#)
4. [基于 PXI 总线 RS422 数据通信卡 WDM 驱动程序设计](#)
5. [FPGA 实现 PCIe 总线 DMA 设计](#)
6. [PCI Express 协议实现与验证](#)
7. [VPX 总线技术及其实现](#)
8. [基于 Xilinx FPGA 的 PCIE 接口实现](#)
9. [基于 PCI 总线的 GPS 授时卡设计](#)
10. [基于 CPCI 标准的 6U 信号处理平台的设计](#)
11. [USB30 电路保护](#)
12. [USB30 协议分析与框架设计](#)
13. [USB 30 中的 CRC 校验原理及实现](#)

VxWorks:

1. [基于 VxWorks 的多任务程序设计](#)
2. [基于 VxWorks 的数据采集存储装置设计](#)
3. [Flash 文件系统分析及其在 VxWorks 中的实现](#)
4. [VxWorks 多任务编程中的异常研究](#)
5. [VxWorks 应用技巧两例](#)
6. [一种基于 VxWorks 的飞行仿真实时管理系统](#)
7. [在 VxWorks 系统中使用 TrueType 字库](#)
8. [基于 FreeType 的 VxWorks 中文显示方案](#)
9. [基于 Tilcon 的 VxWorks 简单动画开发](#)
10. [基于 Tilcon 的某武器显控系统界面设计](#)
11. [基于 Tilcon 的综合导航信息处理装置界面设计](#)
12. [VxWorks 的内存配置和管理](#)
13. [基于 VxWorks 系统的 PCI 配置与应用](#)
14. [基于 MPC8270 的 VxWorks BSP 的移植](#)

Linux:

1. [Linux 程序设计第三版及源代码](#)
2. [NAND FLASH 文件系统的设计与实现](#)
3. [多通道串行通信设备的 Linux 驱动程序实现](#)
4. [Zsh 开发指南-数组](#)
5. [常用 GDB 命令中文速览](#)
6. [嵌入式 C 进阶之道](#)
7. [Linux 串口编程实例](#)

Windows CE:

1. [Windows CE.NET 下 YAFFS 文件系统 NAND Flash 驱动程序设计](#)
2. [Windows CE 的 CAN 总线驱动程序设计](#)
3. [基于 Windows CE.NET 的 ADC 驱动程序实现与应用的研究](#)
4. [基于 Windows CE.NET 平台的串行通信实现](#)
5. [基于 Windows CE.NET 下的 GPRS 模块的研究与开发](#)
6. [win2k 下 NTFS 分区用 ntldr 加载进 dos 源代码](#)
7. [Windows 下的 USB 设备驱动程序开发](#)
8. [WinCE 的大容量程控数据传输解决方案设计](#)
9. [WinCE6.0 安装开发详解](#)
10. [DOS 下仿 Windows 的自带计算器程序 C 源码](#)
11. [G726 局域网语音通话程序和源代码](#)
12. [WinCE 主板加载第三方驱动程序的方法](#)
13. [WinCE 下的注册表编辑程序和源代码](#)
14. [WinCE 串口通信源代码](#)
15. [WINCE 的 SD 卡程序\[可实现读写的源码\]](#)

PowerPC:

1. [Freescale MPC8536 开发板原理图](#)
2. [基于 MPC8548E 的固件设计](#)
3. [基于 MPC8548E 的嵌入式数据处理系统设计](#)
4. [基于 PowerPC 嵌入式网络通信平台的实现](#)

5. [PowerPC 在车辆显控系统中的应用](#)
6. [基于 PowerPC 的单板计算机的设计](#)

ARM:

1. [基于 DiskOnChip 2000 的驱动程序设计及应用](#)
2. [基于 ARM 体系的 PC-104 总线设计](#)
3. [基于 ARM 的嵌入式系统中断处理机制研究](#)
4. [设计 ARM 的中断处理](#)
5. [基于 ARM 的数据采集系统并行总线的驱动设计](#)
6. [S3C2410 下的 TFT LCD 驱动源码](#)
7. [STM32 SD 卡移植 FATFS 文件系统源码](#)
8. [STM32 ADC 多通道源码](#)
9. [ARM Linux 在 EP7312 上的移植](#)

Hardware:

1. [DSP 电源的典型设计](#)
2. [高频脉冲电源设计](#)
3. [电源的综合保护设计](#)
4. [任意波形电源的设计](#)
5. [高速 PCB 信号完整性分析及应用](#)
6. [DM642 高速图像采集系统的电磁干扰设计](#)