·电路与控制·

嵌入式串行通信接收处理的可靠设计方法

汪高勇，常晓玲

（中国电子科技集团公司光电研究院，天津 300000）

摘　要：描述了嵌入式系统串行通信接收处理的可靠设计方法。方法借用ISO模型的分层概念。通过定义广义帧格式建立了一种模块化编程架构，并依此提出了三种循环截断方法和新的环形缓冲参数构建方式，增强了环行缓冲的闭环可靠性，采用了定时及逻辑监控的办法抑制了通信过程中出现的意外故障，增强了容错特性。构建的软件具有更好的适应性，移植、维护及测试简单。

关键词：嵌入式系统；ISO模型；状态解析；环形缓冲；自然循环截断；模块化

中图分类号：TP302　　文章标识码：A　　文章编号：1673-1255（2014）-03-0038-06

Reliability Design Method for Embedded Serial Communication Receiving Processing

WANG Gao-yong, CHANG Xiao-ling

(Academy of Opto-Electronics, China Electronics Technology Group Corporation (AOE CETC), Tianjin 300000*, China)*

Abstract: A reliability design method for embedded system serial communication receiving processing is described. The layer concept of ISO model is introduced. A modularization programming architecture is built by defining generalized frame format. Based on this, three cycle truncation methods and a new ring buffer parameters building mode are proposed to enhance the closed-loop reliability of the ring buffer. The unexpected failure occurred in communication is suppressed and the fault-tolerant features are enhanced using timing and logic monitoring method. The software has better adaptability, transplantation and easy to be maintained and tested.

Key words: embedded system; ISO model; state analysis; circle buffer; natural circle truncation; modularization

嵌入式系统应用中，RS232类（含RS422/485）串行通信方式占据了重要地位。实际应用中，若没有严格、规范的协议，难以保证数据传输的可靠性[1]。通过合理地定义通信协议，使用严密的协议解析方法，改进环行缓冲参数的设计，可简化协议处理过程，提高接收处理可靠性，且使接收过程具有较强的容错能力。由于发送是可控的过程，相对简单，因此文中着重于讲述接收处理设计。

1　RS232类通信的缺陷

如图1所示，基本RS232类串行通信模型在ISO模型[0]中仅使用了最简单的结构。实际应用需要开发人员做不少附加处理保证信息可靠交互。而CAN通信[0]比RS232类通信多了硬件数据链路层，使设备具有统一的通信标准和纠错能力，开发人员便可简化通信处理，将更多精力放在协议内容的解析与执行，而不是在通信过程的建立及传输处理上。

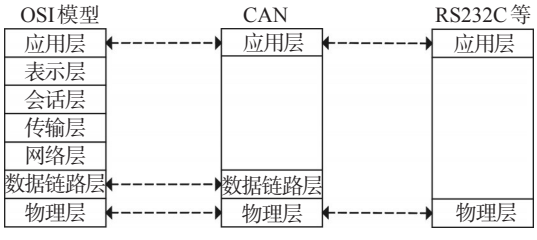


图1　串行通信与ISO模型

收稿日期：2014-03-18

作者简介：汪高勇（1980-），男，江苏南京人，工程师，学士，研究方向为信号处理.

CAN的硬件数据链路层用于寻址、通信纠错，具有完善的帧格式、校验方法、优先级判定及扩展规则。可保证数据无差错的在两个相互连接系统的数据链路层之间传递，而不用关心字节流传输的具体细节。与CAN不同，由于RS232类架构在这方面的天生缺陷和自由性，实际嵌入式应用中依据项目特点便催生了各种不同的RS232类通信协议格式定义方式，由此带来的解析方法也是多种多样，编程方法也依人而异。容易出现如下问题：

(1)帧格式定义较为自由带来处理方法千差万别；(2)由于协议的严密性容易导致数据组合漏洞且不易查出；(3)容错及纠错能力随处理算法不同而迥异；(4)可调式性、移植性、扩展性差；(5)模块化程度低。

若RS232类处理过程也借鉴CAN数据链路层的特性，便可有效解决上述问题。

2　处理设计

为增强RS232类串行通信的可靠性，除采用模块化编写外，主要需解决帧格式(格式协议)及协议处理算法。

2.1　帧格式

不严密的帧格式设计，不利于协议的扩展，易导致处理算法漏洞，在设备运行中造成某些不可追溯的随机故障，增加维护难度。处理算法依据帧格式[9]定义，即数据链路层传输格式，通常可概括为图2所示的广义帧格式。

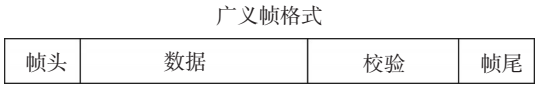


图2　广义帧格式

帧头：广义的帧头含有同步符(或前导码)、机号、地址、控制信息(例如CAN协议)；对简单的RS232类协议，根据实际需求，同步符字段通常为1~n个连续字节，以便接收方识别帧起始位置。

数据：数据结构由应用层定义，由应用层软件解析，数据可以为纯数据信息，也可能为控制命令或混合。

校验：为纠错提供依据，常用有异或、累加和、CRC等，通常为1~n字节。

结束符：结束符设置为1字节且采用与同步符相异的字符防止首尾不分的错误。

透明符[9]：为了避免解析程序在数据段、校验段中将与同步符、结束符相同的数据组合识别为同步符、结束符的错误增加了透明符。与透明符相同的组合前也增加透明符。虽然会增加通信时间开销，但可以提高解析算法的严密性，适当选择透明符在提高可靠性的同时兼顾较低的传输消耗。

帧尾(也即结束符)：标识本帧结束，通常为1~n字节。此外还有透明符作为辅助信息以标识数据与同步符、结束符、透明符的区别。

具体情况，可定义广义帧格式的各分段。例如图3所示帧协议仅包含同步符、数据、校验、结束符，并隐含透明符。

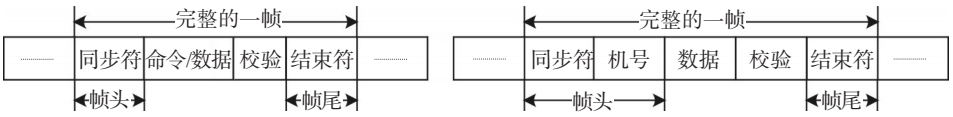


图3　简单帧格式及增加机号的帧格式

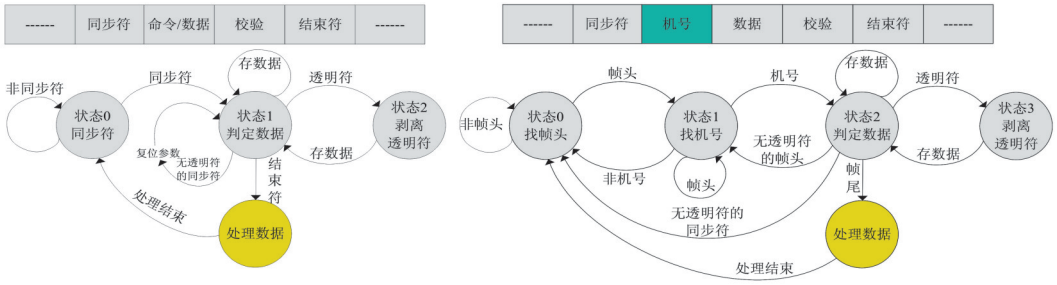
精简的情况，同步符、校验、结束符、透明符均为1字节。若帧头内增加机号信息，则如图3所示。更复杂的协议可依此添加如地址、控制域等信息。

2.2　协议处理算法

2.2.1　状态解析法

状态解析法[9-10](即状态机法)是串行通信软件中使用较为成熟的算法，可实现序列严密解析。简单帧与带机号帧的解析状态图如图4a和图4b所示。

未找到同步符时，状态参数处于状态0；找到同步符后，系统转移至状态1；状态1判定数据并存储，若收到透明符，则转移至状态2并丢弃透明符，再次收到1个字节时直接存储数据，并转移至状态1；若状态1时收到结束符(真正的结束符前面没有透明符)，则认为一帧数据结束，继而调用数据处理函数



(a)简单帧格式解析状态图　　含机号的帧协议剥离状态图　　(b)含机号的帧格式解析状态图

图4　简单帧格式解析状态图

校验并上报数据。结束后,返回状态0,继续下一帧数据处理。需注意,在状态1的过程中,若遇同步符(前面无透明符),则复位已有状态参数,重新存储数据,即从该同步符开始重新计算数据量,防止帧嵌套错误。

2.2.2　数据结构定义

使用状态解析处理算法,需定义一个结构体变量,程序依此实现处理。

```
struct uartclass{
uchar rx_buff[RX_LENGTH];//接收缓冲区
uchar fr_buff[FR_LENGTH]; /帧剥离缓冲区
uchar tx_buff[TX_LENGTH]; //发送缓冲区
uchar rsp;          //接收缓冲区未读数据起始序号读计数器
 uchar rxn;         //接收数据长度
 uchar frn;         //帧有效数据长度
 uchar tsp;         //发送缓冲区待发数据起始序号读计数器
 uchar txn;         //剩余待发送数据长度
 uchar state;       //状态参数
}uart;
```

uart变量的成员rx_buff为接收环行缓冲;tx_buff为发送环形缓冲;fr_buff为处理后的帧数据缓冲,为线性缓冲区,其长度为协议中最大原始数据字节数,该缓冲也是向上报数据的接口。

2.2.3　环行缓冲设计

构成环行缓冲区[0-0]的参数通常为读指针、写指针及缓冲区,修改时需判定读、写指针相对位置,再计算修改参数,当处理程序与中断程序同时修改时,易造成指针修改冲突,且有额外的逻辑判定及运算量。文中采用新的环行缓冲参数构成方法,即使用数据读序号(指针)、数据量及自然循环截断实现缓冲区闭环,原理如图5所示。

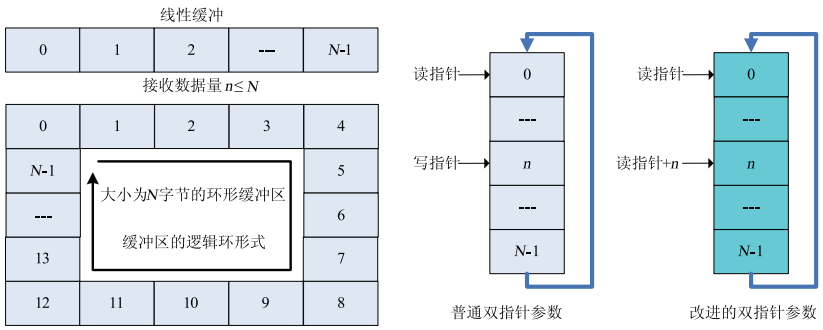


图5　新环形缓冲参数改进示意

参数rsp存储接收数据读序号(可理解为数据序号指针),rxn为接收数据长度。接收数据时,若rxn为0,则:

当前数据写序号=rsp+rxn;

中断每接收1字节,rxn+1→rxn,处理程序每读取1字节,rxn-1→rxn,且rsp+1→rsp。处理程序启动依据为rxn>0。当rxn=RX_LENGTH时,判定缓冲区满。rsp的内容并非数据判定对象,仅记录剩余数据起始位置。当rsp计到最大值RX_LENGTH(rsp从1算起)时,rsp自动复位,实现环形缓冲序号闭环。

2.2.4　参数闭环设计

环行缓冲区参数的闭环是环行缓冲方式比较重要的运算。为提高可靠性,这里采用改进的算法,即“循环截断”有三种方式。RX_LENGTH为10的情况见图6。

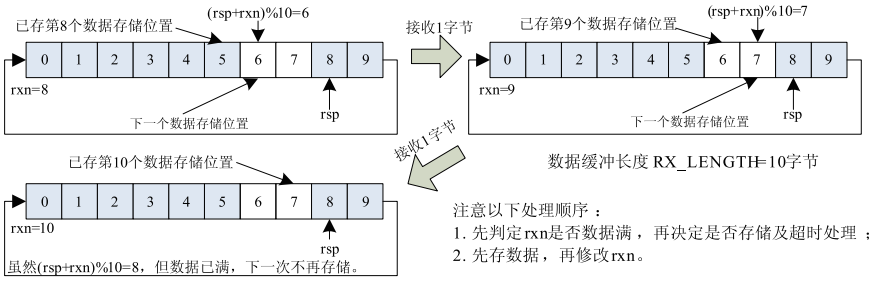


图6　RX_LENGTH为10的情况

(1)取模方式(%)

随着不断读取数据,rsp终会计到接收缓冲最大长度RX_LENGTH,而对rsp+rxn的情形,当rsp>0,rxn>0时rsp+rxn也会因此出现越界。为使缓冲区连续,可对rsp或rsp+rxn取模,使rsp或rsp+rxn返回环行缓冲区起点。例如:当RX_LENGTH=10,当前读数据起始序号rsp=8,已有数据长度rxn=8,则:当前数据写序号(临时变量)=(rsp+rxn)% RX_LENGTH=13%10=6;可知,若新接收1字节,则应在数组序号为6的地方存数据,并修改rxn+1→rxn,rxn=9,若再接收1字节,运算后,数据写序号为7,存入数组后,rxn=10。当再接收数据时,因rxn=RX_LENGTH,可判定缓冲区满,不再处理或报出错。当数据写序号=7时,与数据读序号=8刚好构成环形连续。

(2)数据与方式(&)

若RX_LENGTH取2的整数倍,可简化为数据与运算,如RX_LENGTH=16,当前读数据起始序号rsp=8,已有数据长度rxn=8时,运算可简化为:

当前数据写序号(临时变量)=(rsp+rxn) & (RX_LENGTH-1)=0x10 & 0x0F=0;可知,新接收数据应在数组序号为0的地方存储,若再接收8字节,数据存入数组后rxn+8→rxn,rxn变为16(先存再rxn++,则最后一个数据的数组序号为7)。当rxn等于RX_LENGTH,再接收数据时,可判定缓冲区满。这时,数据写序号=7与数据读序号=8刚好构成环形连续。

(3)自然循环截断方式

若内存充裕,rsp、rxn定义为unsigned char类型可实现256字节自然循环截断,可更快地实现环形参数闭环,同样还是上述范例,当RX_LENGTH=256时,数据写序号运算可简化为:

当前数据写序号(临时unsigned char变量)= rsp+rxn;由于rsp、rxn、数据写序号均为临时unsigned char变量,对rsp+rxn超出255范围的高位数据将被舍弃,即(rsp+rxn)与0xFF相与的结果,即模256。而rsp计满255后由于自然溢出而复位,避免了运算及比较,由于仅有一次加法运算,进一步减少了代码并缩短了处理时间。

对需要设定缓冲区大小的场合,取模可实现通用设计;对2的整数倍缓冲区大小,使用数据与可实现较快速的运算;对资源比较充裕的系统,则完全可以考虑自然循环截断,以简化运算,加快处理速度。

需注意:在数组定义中实际寻址序号最大值为RX_LENGTH- 1,采用数据与运算时,应使用RX_LENGTH-1取与,这与取模不同,自然循环截断的原理与数据与类似。除(rsp+rxn)外,rsp也可使用循环截断实现闭环。

以上三种环形参数运算方法均不需要较多的运算、比较,可使缓冲区严格闭环并缩减了处理过程及时间。以自然循环截断效率最高且简单,其次为数据与及取模。对目前的系统硬件处理速度而言,这种运算差异,可以忽略不计。

2.2.5　参数修改冲突处理

应注意中断程序调用与非中断程序之间的参数修改冲突。中断与协议解析处理均使用环行缓冲区,修改参数rxn,易导致rxn修改冲突。可在协议解析处理中使用局部屏蔽中断的方法实现参数安

全修改，如图7所示。协议解析处理修改rxn时，先暂时屏蔽接收中断使能，并以简短语句完成参数修改，之后恢复中断使能。鉴于串行通信传输速率相对较慢，几个处理周期的延迟不会对中断造成影响。

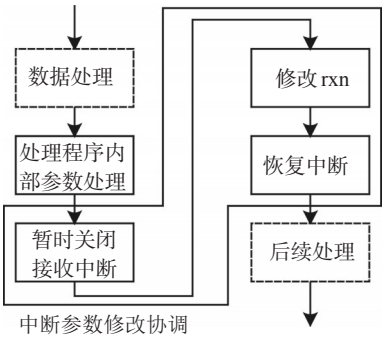


图7 中断参数修改协调

做其他中断处理程序时，尽量避免在中断中写入大量的处理代码，中断仅为数据捕获及缓存，这有利主程序任务及数据处理的连续执行而不会因为中断处理超时造成各种未知阻塞。

2.2.6 协议解析处理

使用状态机实现解析，流程如图8所示。主程序定时调用协议解析处理程序，协议解析处理程序判定urat.rxn>0时启动，按照状态图不断查找符合要求的帧头、帧尾。判定帧接收完整、校验正确后，上报应用层处理程序，之后复位状态参数重新处理。对超时及校验错误的情况，直接复位参数，继续搜索。处理过程中，超时由监控程序单独控制，使软件具有帧残缺、数据错误的自恢复特性。

图8 协议解析处理执行流程

2.2.7 监控设计

监控软件[9]在处理条件尚不满足、处理过程因外部原因而中断死等或数据保鲜时间超过一定限度后强制退出。超时监控放置于系统定时中断内，采用单字节接收中断复位计时器数值的方法实现。逻辑监控在通信处理模块内依据错误条件判据放置，常见的几类错误见表1所示。

上述错误均便于在协议解析处理程序的各阶段加入判定条件。符合条件要求便执行相应处理，并由监控软件发出错误消息，将类型编号发送上报，便于程序调试。对系统联试，若开放自定义的测试程序段，可实现通信自测试功能。

3 实际编程

建立数据结构及模块化后，针对接收处理需构建以下两类程序组：

```
//对外接口程序组
void  Serial_Ini(uint,uchar, uchar (*)(uchar *));  //产口模块初始化—主程序调用
void  Serial_ Process(void);  //串口处理—主程序轮询调用
void  Serial_Watch(void);  //定时监视—定时中断调用
```

表1　串行通信错误及处理

错误类型	可能的原因	状态解析处理	监控处理
有头无尾	未知的通信中断、数据冲突或乱数	正常等待结尾	超时处理
无头有尾	未知的通信中断、数据冲突或乱数	正常处理	无
连头	上位机出错或乱数干扰	正常处理	无
无透明符	上位机出错、乱数干扰	校验处理	无
帧空	上位机出错、乱数干扰	正常处理	无
帧超长	上位机出错、乱数干扰	正常处理	超数据量处理
乱数阻塞	上位机出错、乱数干扰	正常处理	超数据量处理
帧嵌套	上位机出错	正常处理	无
连续*N*帧	上位机连发	正常处理	超数据量处理
上电乱数	硬件不稳定	超数据量处理	超数据量、超时处理
随机组合	乱数、受干扰	正常处理	无

```
void  Serial_Rx_Int(uchar);  //接收中断处理—串口硬件中断调用
//对内处理服务程序组
uchar (*Serial_Cmd)(uchar *);  //应用层接口外调函数指针
void  Serial_Data_ Process(void);  //数据校验及应用层外调处理
void  Serial_Watch_On(void);  //定时监视开启
void  Serial_R_Rst(void);  //接收参数复位
void  Serial_Error(uchar);  //串口报错
void  Serial_Rsp_Add(uchar);  //数据扫描参数修正
串口模块外部接口及参数初始化:
void  Serial_Ini(uint  baud, uchar stop, uchar even, uchar (*Process)(uchar  *))
{
 串口硬件参数初始化;
 Serial_Cmd = Process;  //外部应用层处理入口函数指针初始化。
}
主程序挂接实例:
main()
{
 System_Ini();
 Serial_Ini(115200,1,0, App_Ent);  //波特率,停止位,奇偶位,应用层处理入口
 …
 while(1)
{
 Task_0();
 …
 Task_n();
 Serial_Process();  //主程序轮询调用串口处理
}
}
中断程序挂接实例:
void  Serial_Int(void) interrupt  x
{
 if (RI)
{ Serial_Rx_Int(物理层接收缓冲);RI=0;}  //接收中断处理
 if (TI)
{发送调用略;TI=0;}
 //发送中断处理
}
系统定时中断程序挂接实例:
void  Time0_Int(void) interrupt  y
{
 Time_Task0();  //其他定时任务
 …
 Serial_Watch();  //串口定时任务
}
```

应用层由带参数的函数指针接口。串口通信处理模块得到被剥离了透明符、同步符、帧尾结束符的原始数据,经校验后调用指向应用层入口的函数指针。完成数据传递。其过程如图9所示。

4　效果检测

试验针对帧错误[9]及大数据量乱码测试,程序均正常运行,在连续的乱数序列中均可检出正常数据帧,这对于RS485总线类的通信场合非常有用。对采用本方法设计的接收处理程序进行大数据量连续

(下转第55页)

术出版社，2010：60-62.
[5] 李堡勇，王显军.码盘的原理及应用[J]. 伺服控制，2008，2：43-45.
[6] 杨俊志.单码道绝对式角度编码器的编码及解码原理[J]. 仪器仪表学报，2004，25（增刊）：139-141.
[7] 赵波.绝对式三级组合光电轴角编码器[J]. 微计算机信息，2008，24（5-1）：12-14.
[8] 苏海冰，刘恩海.单圈绝对式编码器的研制[J]. 光学精密工程，2002，10（1）：74-78.
[9] 郁有文，常健.绝对码编码器中一种新型的编码方法[J]. 仪器仪表学报，2004，25（4）：541-544.
[10] 汤天瑾，曹向群，林斌.光电轴角编码器发展现状分析及展望[J]. 光学仪器，2005，27（1）:90-95.

（上接第43页）

灌入（约1 GB随机数据及隐藏其中的若干正确序列以115 200 bps速率24 h连续灌入）测试后，软件仍运转正常，并成功检出隐藏其中的正确帧序列，且能上报乱数中错误的数据组合错误类型。

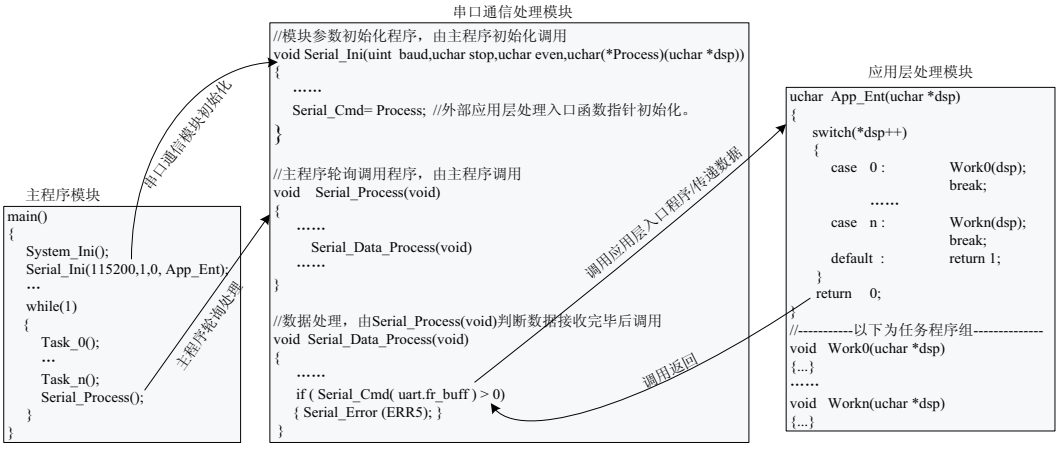


图9　接口函数调用关系

5　总　结

提出用于嵌入式串行通信处理的可靠设计方法并用C语言实现。采用新的环行缓冲参数设计避免有双指针参数运算易出错的缺点；提出三种循环截断方法改善了环行缓冲的闭环可靠性；采用定时及逻辑监控抑制了通信中的意外错误，并具有防阻塞、可调式及自测试能力。所述方法可提高嵌入式串行通信软件接收处理的可靠性，使设备软件具有更好的适应性，移植、维护及测试简单。

参考文献

[1] 祝军生.现代通信系统软件可靠性设计技术[J]. 电子产品可靠性与环境试验，2005（3）.
[2] Benhrouz Forouzan（美）. 数据通信与网络[M]. 潘仑，译. 北京：机械工业出版社，2000：25-35，204-225.
[3] 江玲.一种简单可靠的串行通信处理方案[J]. 西南科技大学学报，2004，19（4）.
[4] 李莹，贾彬.一种基于状态机的串口通信协议的设计与实现[J]. 电子设计工程，2012，20（7）.
[5] 夏桂华.基于状态机的远程串行通讯技术研究[J]. 计算机测量与控制，2012.
[6] 刘洪斌.采用状态机和消息机制的串口接收程序[J]. 单片机与嵌入式系统应用，2001（10）：72-73.
[7] 魏先民.有限状态机在嵌入式软件中的应用[J]. 潍坊学报，2007，6（4）：24-25.
[8] 陈乐.嵌入式环境下串行帧通信的设计与实现[J]. 现代电子技术，2010（23）.
[9] 宋国明.嵌入式系统串口通信分层结构设计与实现[J]. 微计算机信息，2006，22（4-2）.
[10] 刘辉.嵌入式实时系统CAN通信软件设计方案[J]. 计算机仿真，2008，25（4）.
[11] 马玉春，宋瀚涛.串行通信协议的研究及应用[J]. 计算机应用研究，2004，21（4）：228-229.
[12] 杨国志.DOS与Windows环境下串行通信方法的研究[J]. 计算机工程与设计，2004，25（8）.
[13] 杨飞然.循环缓冲机制在DSP异步数据访问中的应用[J]. 电声技术，2013，37（1）.