高效 FIFO 串口双机通信在 ARM7 上的实现

马明龙，黄春梅，张 瑞

（长春工业大学 计算机科学与工程学院，吉林 长春 130021）

摘 要： 详细介绍了高效 FIFO 串口通信的基本原理和实现方法，并在两台基于 ARM7TDMI 微处理器的目标机上，用 FIFO 串口通信模式实现了两机之间的高效通信。整个工程分寄存器配置模块、串口接收模块、串口发送模块和容错模块。

关键词： 嵌入式系统；串口通信；FIFO；ARM；模块

中图分类号： TP368.1 **文献标识码：** B

Implementation of efficient serial communication between two targets based on ARM7 by using FIFO

MA Ming Long，HUANG Chun Mei，ZHANG Rui

（School of Computer Science and Engineering，Changchun University of Technology，Changchun 130021，China）

Abstract： This paper illustrates the key principle and method of serial communicating with FIFO buffer，and implements communicate efficiently between two targets based on ARM7TDMI core in FIFO-UART made. The whole project consists of four modules，which are registers configuring module，receiving module，transferring module and error tolerance module.

Key words： embedded system；UART；FIFO；ARM；module

S3C44B0X（时钟频率为 60 MHz）的 UART 单元提供 2 个独立的异步串行 I/O 口，每个通信口均可工作于中断或 DMA 模式。即 UART 能产生内部中断请求或 DMA 请求，在 CPU 和串行 I/O 口之间传送数据。它支持高达 115.2 Kb/s 的传输速率，每 1 个 UART 通道包含了 2 个 16 位的分别用于接收和发送信号的先进先出（FIFO）通道。S3C44B0X UART 包括可编程波特率、红外发送/接收、1 个开始位、1 个或 2 个停止位、5/6/7/8 位数据宽度和奇偶校验。每个 UART 包含 1 个波特率发生器、接收器、发送器和控制单元，其构成如图 1 所示[1]。

1 FIFO 概述

1.1 FIFO 概念

先入先出 FIFO（First In First Out），即先被写入到 FIFO 的数据将会先被读出。它是一片用来缓存数据的存储单元，可以把需要处理的数据先暂存在这片存储单元中，在数据量达到一定数量时再集中处理，以提高系统性能。FIFO 可以集成在芯片中，而当系统需要的缓冲区较大时，也可以用单独的 RAM 实现。S3C44B0X 串口

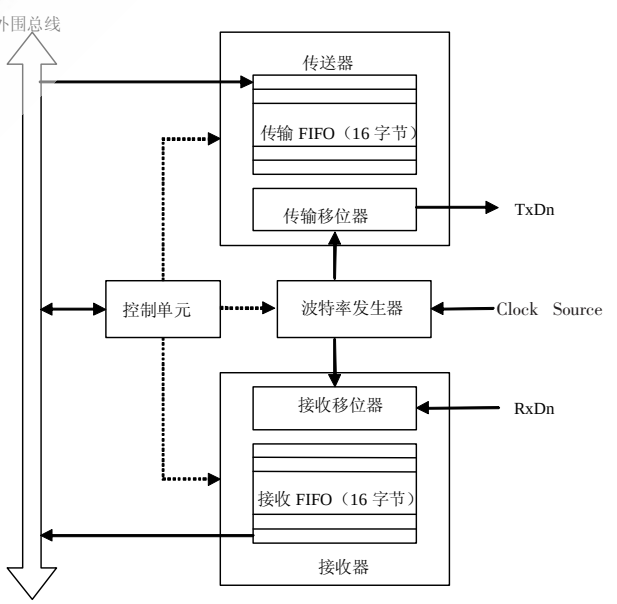

图 1 配置 FIFO 的 UART 框图

收发器包含了 16 B 的 FIFO 和数据移位器，将要传输的

数据写进 FIFO，然后赋值到发送移位器，最后从发送的引脚移位发送出去，达到利用 FIFO 缓冲区高效通信的目的。

1.2 FIFO 意义

FIFO 是数据传输系统中极其重要的一环，特别是在 2 个处于不同时钟域的系统接口部分，FIFO 的合理使用，不但能使接口处数据传输的输入输出速率进行有效的匹配，不使数据发生复写、丢失和读入无效数据的情况，而且还会有效地提高系统中数据的传输效率。使用 FIFO 进行串口通信，较之传统的串口通信有更高的效率。它将要发送和已经接收的数据集中起来进行操作，避免了频繁的总线操作，减轻了 CPU 的负担。因此，使得基于 FIFO 方式的串口通信目前应用十分广泛。

1.3 FIFO 中断请求

S3C44B0X 的 UART 有 7 个状态(Tx/Rx/Error)信号：溢出错误、奇偶错误、帧错误、断点条件、接收 FIFO/Buffer 数据准备就绪、发送 FIFO/Buffer 空和发送移位寄存器空，这些状态信号由相应的 UART 状态寄存器(UTRSTATn/UERSTATn)声明[1]。

当处于接收错误状态时，如果在控制寄存器(UCONn)中接收错误状态中断使能位被置为 1，则溢出错误、奇偶校验错误、帧错误及断点错误，每 1 个作为 1 种错误状态都可发出错误中断请求。当 1 个接收错误状态中断请求被发现时，引起中断请求信号会被读 UERSTATn 所识别。如果控制器中的接收模式被选定为中断模式，则当接收器从接收移位寄存器向接收 FIFO 传输数据时，会激活接收 FIFO 的可引起接收中断的“满”状态信号。同样，如果控制器中的发送模式被选定为中断模式，则当发送器从发送 FIFO 向发送移位寄存器传输数据时，可引起发送中断的发送 FIFO“空”状态信号被激活。如表 1 所示。

表 1 与 FIFO 相连的中断

类型	FIFO 模式	非 FIFO 模式
Rx 中断	若每次接收的数据达到了接收 FIFO 的触发水平，则 Rx 中断产生；若 FIFO 为非空且在 3 字时间内没有接收到数据，则 Rx 中断也将产生(接收超时)	若每次接收数据变为“满”，则接收移位寄存器产生的 1 个中断
Tx 中断	若每次发送的数据达到了发送 FIFO 的触发水平，则 Tx 中断产生	若每次发送数据变为“空”，则发送保持寄存器产生 1 个中断
错误中断	帧错，若奇偶校验错和断点条件信号被发现且以字节为单位被接收，则将产生 1 个错误中断	所有的错误立即产生 1 个错误中断。如果同时另 1 个错误中断发生，则只会产生 1 个中断

2 FIFO 串口通信的实现

FIFO 重启时，输入和输出的指针都指向 FIFO 中的第 1 个存储位置。对 FIFO 的每次写入操作会使输入指针指向 FIFO 的下 1 个存储位置，相应地每次读取操作会使 FIFO 的输出指针指向 FIFO 的上 1 个存储位置。若指针需要从最后 1 个存储位置移动到第 1 个存储位置，则 FIFO 会自动实现这一过程而不需要任何对指针的重启操作。FIFO 内部除了包含输入和输出端口之外，通常还有其他状态标志输出，如空状态和满状态。当 FIFO 已空或者已满时，空状态和满状态标志位就会有相应的输出，即当 FIFO 已空时不能进行读取操作，当 FIFO 已满时不能进行写入操作[2]。

2.1 配置特殊寄存器

为了使目标系统能正常工作，必须配置相关的寄存器，如 I/O 口寄存器、串口控制寄存器和串口源/目的寄存器等。S3C44B0X 有 2 个串口，这里以串口 0 为例，进行相关寄存器的配置。

```
/*I/O 口配置，定义各相关引脚功能和上拉电阻状态 */
rPCONC  |=0xf0000000;
rPUPC   |=0xc000;
rPCONE=(rPCONE  &0x3ffeb)|0x28;
rPUPE   |=0x6;
rPCONF=(rPCONF  &0x3ff)+0x124800;
rPUPF   |=0x1e0;
/* 定义串口 0 工作寄存器组 */
rULCON0=0x3;   //正常模式，无奇偶校验，1 位停止位，
                //8 位数据位
rUCON0=0x245;   //Rx 为边沿触发，Tx 为电平触发，禁
                //止超时中断，产生接收错误中断，普通传送、
                //发送与接收为中断或轮询模式
rUFCON0=(2<<6)|(1<<4)|(6)|1;  //FIFO 启动需先复位
rUBRDIV0=(mclk/(baud*16));   //mclk 为 60000000，baud
                //为 115200
```

2.2 FIFO 串口发送模块

串口数据发送帧格式是可编程的，它包含 1 个开始位，5~8 个数据位，1 个可选的奇偶位和 1~2 个停止位，这些都可以通过线控制寄存器(UCONn)来设置。发送器也能够产生发送中止条件。中止条件迫使串口输出保持在逻辑 0 状态，这种状态保持超过 1 个传输帧的时间长度。通常在 1 帧传输数据完整地传输完之后，再通过这个全 0 状态将中止信号发送给对方。中止信号发送之后，传送数据将持续地放入到输出 FIFO 中。要发送的数据被存放在定义的字符串指针 uart0TxStr 中，串口发送模块通过读该字符串中的字符进行数据发送，核心源代码如下：

```
void __irq Uart0_TxFifoInt(void)
{
```

```
    /* 判断 FIFO 发送缓冲区是否为满或字符串结束 */
    while( ! (rUFSTAT0 & 0x200) && (*uart0TxStr ! ='\0'))
    {
        rUTXH0=*uart0TxStr++;
        for(i=0;i<700;i++);   //延迟,防止 FIFO 误写
    }
    rI_ISPC=BIT_UTXD0;
    if(*uart0TxStr == '\0')
    {
        rINTMSK |= BIT_UTXD0;
        rI_ISPC=BIT_UTXD0;
    }
}
```

2.3 FIFO 串口接收模块

接收的数据帧格式与发送一样都是可编程的。它包括了 1 个起始位,5~8 个数据位,1 个可选的奇偶校验位和 1~2 个停止位,这些都可以通过线控制寄存器(UCONn)来设置。接收器还可以检测到溢出错误、奇偶校验错误、帧错误和中止状况,每种情况下都会将 1 个错误标志置位。

(1)溢出错误表示新的数据已经覆盖了旧的数据,因为旧的数据没有及时被读入。

(2)奇偶校验错误表示接收器检测到了意料之外的奇偶校验结果。

(3)帧错误表示接收到的数据没有有效的停止位。

(4)中止状况表示 RxDn 的输入被保持为 0 状态超过了 1 个帧传输的时间[3]。

(5)在 FIFO 模式下接收 FIFO 不为空,但接收器已经在 3 个字时间内没有接收到任何数据,就认为发生了接收超时状况。

接收模块将数据从接收移位寄存器中读出后,首先被存储到接收缓存数组 keyBuf[ ]中。变量 keyBufWrPt 和 keyBufRdPt 指向缓存数组中当前写数据和读数据,当接收模块往缓存数组中写入 1 个字节后,keyBufWrPt 加 1;当 Uart_IntGetKey 从缓存数组中读出 1 个字节后,keyBufRdPt 加 1。两变量最大值为 KEY_LEN,超过最大值时置零。接收模块的核心代码如下[4]:

```
/* 接收模块将移位寄存器中的数据读出到接收缓存数组中 */
void __irq Uart0_RxFifoInt(void)
{
    rI_ISPC=BIT_URXD0;
    if(rUFSTAT0==0)
    Uart_Printf("time out\n");
    while( (rUFSTAT0&0xf) >0 )          //循环直到 FIFO
                                        //发送缓冲区为空
    {
        keyBuf[keyBufWrPt++]=rURXH0;    //读取接收缓
                                        //冲区数据存入缓存数组
        if(keyBufWrPt==KEY_BUFLEN)
        keyBufWrPt=0;
    }
}
/* 定义 1 个函数从接收缓存数组中读取数据 */
char Uart_IntGetkey(void)
{
    if(keyBufRdPt==KEY_BUFLEN)
    keyBufRdPt=0;
    while(keyBufWrPt==keyBufRdPt);//等待直到 FIFO 被触发
    return keyBuf[keyBufRdPt++];
}
```

2.4 FIFO 容错模块

除了接收 FIFO 寄存器之外,UART 还具有 1 个状态 FIFO。状态 FIFO 表示了在 FIFO 寄存器中,哪一个数据被毫无错误地接收。假设 UART 的 FIFO 连续接收到 A、B、C、D、E 字符,并且在接收 B 字符时发生了帧错误(即该字符没有停止位),在接收 D 字符时发生了奇偶校验错。虽然 UART 错误发生了,但不会产生错误中断,因为含有错误的字符还没有被 CPU 读取。当字符被读出时错误中断才会发生,而且只有在读出 URXHn 和 UERSTATn 寄存器后,FIFO 错误状态寄存器才会被清除[5]。容错模块核心代码如下[6]。

```
void __irq Uart0_RxFifoErrorInt(void)
{
    rI_ISPC=BIT_UERR01;
    Uart_Printf("UERSTAT0=0x%x\n",rUERSTAT0& 0xf);
    while( (rUFSTAT0&0xf) >0 )
    {
        keyBuf[keyBufWrPt++]=rURXH0;
        if(keyBufWrPt==KEY_BUFLEN);
        keyBufWrPt=0;
    }
}
```

3 实验结果

本实验在 S3C44B0X 和 ADS1.2 平台上实现,取得了预期的效果。在同等条件下(忽略温度、电压等外部因素变化),在带 FIFO(FIFO)和不带 FIFO(Non-FIFO)时发送和接收所花时间如表 2 所示。

表 2 Non-FIFO/FIFO 发送和接收时间(单位:s)

状态 / 数据量	Non-FIFO		FIFO	
	发送	接收	发送	接收
1 KB	0.483 072	0.094 905	0.415 276	0.095 765
2 KB	1.086 025	0.189 933	0.831 108	0.167 113
4 KB	2.113 323	0.380 108	1.566 247	0.337 276

(下转第 48 页)

的利用率有所降低，降低了 1~2 个百分点。由此可以看出，XCP 协议对在拥塞情况下提高链路利用率达到了很好的效果，能充分地利用链路带宽，同时达到稳定的带宽利用率的时间也很快。

由图 5 可以看出在第 5 s 之前，没有发生拥塞，缓冲队列值为 0；第 5 s 时，第 2 个流开始发送数据，主干链路发生拥塞，队列值有所增加；第 10 s 开始随着拥塞的变大缓冲队列值又有所增大；缓冲队列值随着拥塞的加剧逐渐增大，但增加的值不大，说明 XCP 协议可以保持很小的排队队列值，这样相应的排队时延值也会很小。

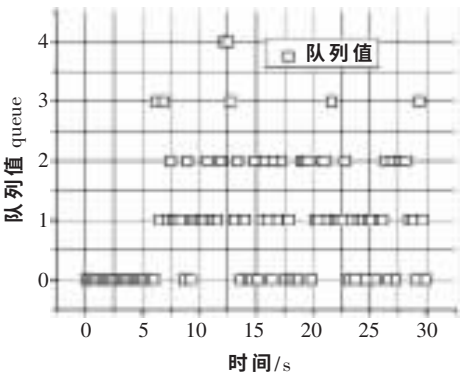

图 5 平均队列图

本文在 ns2 下面对 XCP 协议的公平性和稳定性做了相应的仿真分析。仿真结果表明，高带宽时延乘积网络中 XCP 协议不仅能够保持链路的公平性和稳定性，而且还能达到链路的高利用率，同时，路由器的平均队列值保持得很小。

参考文献

[1] FLOYD S, JACOBSON V. Random early detection gateways for congestion voidance [J]. IEEE/ACM Transactions on Networking, 1993, 1(4):397-416

[2] ATHURALIYA S, LI V H, LOW S H, et al. Rem: Active queue management[J]. IEEE Network, 2001,15(3):48-53.

[3] HOLLOT C, MISRA V, TOWSLEY D, et al. On designing improved controllers for aqm routers supporting tcp flows[J]. In Proc. of IEEE INFOCOM, Apr. 2001.

[4] KUNNIYUR S, SRIKANT R. Analysis and design of an adaptive virtual queue [J]. In Proc. Of ACM SIGCOMM, 2001.

[5] KATABI D. Congestion control for high bandwidth-delay products networks [EB/OL]. In: Proc. ACM SigComm'02, Aug. 2002.

(收稿日期：2009-06-05)

(上接第 42 页)

以传输 4 KB 数据为例，由表 2 可知，使用 FIFO 时，发送和接收分别节省 0.547 076 s 和 0.042 832 s 时间。假定传输 1 bit 的数据用时为 θ s，传输数据量为 n，则可知使用 FIFO 和不使用 FIFO 两种情况下的用时差为 $15n\theta/16$s。由此可见，当传输数据量 n 越大时，采用 FIFO 的串口传输模式的用时越少、优越性越明显。这也显示了 FIFO 在串口传输较大数据量的工程应用中的重要性和必要性。

在串口通信应用越来越广的背景下，提高串口通信速度显得格外重要。本文以 S3C44B0X 微处理器为平台，介绍的基于 FIFO 的串口双机通信的原理和实现方法，该方法同时也适用于其他配置 FIFO 缓冲区的微处理器，具有很强的适用性和通用性，在学习、研究的同时，也为工程应用中的串口通信提供了参考模型。

参考文献

[1] 田泽.嵌入式系统开发与应用[M].北京：北京航空航天大学出版社，2005.

[2] Samsung. S3C44B0X datasheet[EB/OL]. http://pdf1.alldatasheet.com/datasheet-pdf/view/84253/SAMSUNG/S3C44-B0X.html, 2003.

[3] 马忠梅，英惠.ARM 嵌入式处理器结构与应用基础(第 2 版)[M].北京：北京航空航天大学出版社，2007.

[4] 严蔚敏，吴伟民.数据结构：C 语言版[M].北京：清华大学出版社，1996.

[5] 胥静.嵌入式系统设计与开发实例详解：基于 ARM 的应用[M].北京：北京航空航天大学出版社，2005.

[6] 王宇行.ARM 程序分析与设计[M].北京：北京航空航天大学出版社，2008.

(收稿日期：2009-05-12)

(上接第 45 页)

UWB 通信系统的性能进行分析，并与传统算法，以及各种放宽约束算法进行了比较。实验表明，基于 DPSO 的多路径选择算法在低信噪比情况下具有更好的择优性能，其抗多用户干扰的能力也优于其他同类算法。

参考文献

[1] CABRIELLA M, BENEDETTO D, GIANCOLA G. 超宽带无线电基础[M].葛利嘉，朱林，袁晓芳，等译.北京：电子工业出版社.2006.

[2] LEE C, CHO D, YOU Y, et al. A solution to improvement of DS-UWB system in the wireless home entertainment network[J]. IEEE Transactions on Consumer Electronics, 2005,51(2):529-533.

[3] GEZICI S. Design and analysis of impulse radio ultra wideband receivers for communications and geolocation[D]. Dissertation Abstracts International, 2006, 67(2).

[4] 杨红孺，高洪元，庞伟正，等.基于离散粒子群优化算法的多用户检测器[J].哈尔滨工业大学学报，2005，37(9)：1303-1306.

[5] 李俊，郝成民，刘湘伟.改进 PSO 算法在雷达干扰任务分配中的应用[J].计算机仿真，2008，25(12)：27-30.

[6] 徐珍霞，顾洁.离散粒子群优化算法在变电站选址中的应用[J].电气应用，2006，25(4)：35-38.

(收稿日期：2009-06-08)