

**《单片机原理与实践》**

**Cortex-M4 TM4C1294XL微处理器**

**UART模块的介绍**

**姓 名 林宇航**

**指导教师**  **郭方洪老师**

**学 院 信息工程学院**

**专业班级 自动化1901班**

**报告提交日期**  2021年1月11日

**Cortex-M4 TM4C1294XL微处理器**

**UART模块的介绍**

摘要：数据传输的接线方式大体上就是两种：一种是并行接口，一种是串行接口。本组就串行接口展开研究，所谓串行接口是指数据在有限的几个IO上按照顺序，一位一位的进行传输。UART、IIC、SPI、CAN、USB等只要是串行传输的接口，都是串口的一种，本文重点讨论的是其中的UART口。

一.UART定义

通用异步收发器(Universal Asynchronous Receiver/ Transmitter, UART)是一个异步的串行通信接口。UART模块将处理器内部的并行数据转换为串行数据，通过串行总线UnTX以异步通信的方式发送出去；另一方面它也可以接收UnRX总线上的串行数据，转换为并行数据后返回给处理器进行处理。异步模式是一种常用的通信方式，相对于同步模式，异步模式不需要一个专门的时钟信号来控制数据的收发，因此发送数据时位与位的间隙可以任意改变。UART总线采用双向通信，可以实现全双工的发送和接收。嵌入式设计中，UART用来与计算机或其他设备进行通信。

二.UART的优点

这种通信方式使用的数据线少，在远距离通信中可以节约通信成本，但其传输速度比并行传输低。它很简单并且能够实现远距离通信。比如IEEE488定义并行通行状态时，规定设备线总长不得超过20米，并且任意两个设备间的长度不得超过2米;而对于串口而言，长度可达1200米。

通信使用3根线完成：(1)地线，(2)发送，(3)接收。由于串口通信是异步的，端口能够在一根线上发送数据同时在另一根线上接收数据。串口通信最重要的参数是波特率、数据位、停止位和奇偶校验。对于两个进行通信的端口，这些参数必须匹配。

三.UART与IIC、SPI的区别

1、UART就是两线，一根发送一根接收，可以全双工通信，线数也比较少。数据是异步传输的，对双方的时序要求比较严格，通信速度也不是很快。在多机通信上面用的最多。

2、SPI接口和UART相比，多了一条同步时钟线，UART的缺点也就是它的优点了，对通信双方的时序要求不严格，不同设备之间可以很容易结合，而且通信速度非常快。一般用在产品内部元件之间的高速数据通信上面，如大容量存储器等。SPI是三线制,网上可找到其通信协议和用法的3根线实现数据双向传输串行外围接口，收发独立、可同步进行。

3、I2C接口也是两线接口，它是两根线之间通过复杂的逻辑关系传输数据的，通信速度不高，程序写起来也比较复杂。一般单片机系统里主要用来和24C02等小容易存储器连接。

四. UART功能与特点

TM4C1294NCPDT 控制器包括8个UART模块，每个UART模块都有以下特点。

1)波特率可以通过编程设定,普通模式可以达到7.5 Mbit/s (16分频),高速模式可以达到15 Mbit/s（8分频）。

2）独立的16×8位发送（TX)缓冲区（FIFO)和接收（RX)缓冲区,可降低中断服务对CPU的占用。

3) FIFO的长度可以通过编程设定,包括提供传统双缓冲接口的1字节深的操作。

4）FIFO触发深度可设为1/8、1/4、1/2、3/4和7/8。

5）标准的异步通信,包括起始位、停止位和奇偶校验位。

6）线中止(Line-break）的产生和检测。

7）完全可编程的串行接口特性。

·有5、6、7或8个数据位。

·偶校验、奇校验、奇偶校验或无奇偶校验位的产生与检测。

·产生1或2位停止位。

8）IrDA串行红外SIR编码器/解码器模块。

·可选择采用 IrDA串行红外（SIR)输人/输出或者UART 输入/输出。

·支持 IrDA SAR编码/解码功能,半双工使数据速率高达115.2kbit/s。

·支持标准的3/16位持续时间和低功耗位持续时间（1.41～2.23us）

·可编程的内部时钟产生器，允许对参考时钟进行1~256的分频以得到低功耗模式的位持续时间。

9）支持与符合ISO7816标准的智能卡(smart cards）通信。

10）在以下 UART模块中提供调制解调器功能。

·UART0(调制解调器流控制和调制解调器状态)。

·UART1 (调制解调器流控制和调制解调器状态)。

·UART2(调制解调器流控制）

·UART3（调制解调器流控制）

·UART4（调制解调器流控制）

11）标准的FIFO深度和传送结束中断。

12）采用微型直接内存访问控制器（LDMA) 进行高效传输。

·有独立的通道用于发送和接收。

· 当接收FIFO有数据时产生单次接收请求,在接收FIFO到达预设的触发深度时产生猝发请求。

·当发送 FIFO中有空闲空间时产生单次接收请求，在接收FIFO到达预设的深度时产生猝发请求。

13）全局备用时钟(ALTCLK)资源或者系统时钟(SYSCLK) 可以用来产生波特率时钟。

五.UART功能的实现

主处理器与可编程模块的连接见图3。上电之后,主处理器必须送给可编程模块有效的/RESET信号, 确保模块回到初始状态。主处理器通过数据线送出控制信息和要发送的数据。如果D/C=0, 表明主处理器送来的是控制信息; 如果D/C=1, 表明主处理器 送来的是要发送的数据。而可编程模块RXD端接收到的数据也通过数据线送到主处理器。接收到有效的数据或数据发送完毕时,READY端口会送出低脉冲通知主处理器。

由图2可以看出, UART的功能主要有四块: 发送、接收、状态信息和控制信息。如果开始就考虑控制信息, 程序会过于复杂,加大对发送、接收功能仿真的难度。所以,我们先假设控制信息已经确定,如: 数据位数是8 位,停止位位数是1位,不加校验位, 波特率选为9600bps(这意味着可编程器件需要一个频率等于9600,单位为Hz的时钟)。等发送和接收功能实现后再逐步加入控制功能。

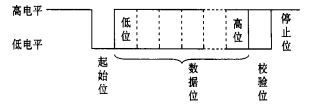


图1串行异步通信字符代码传输格式

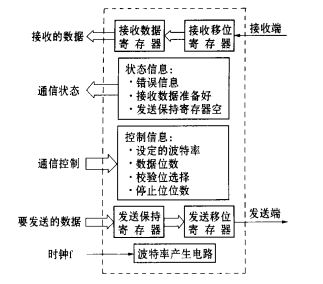


图2 UART功能框图

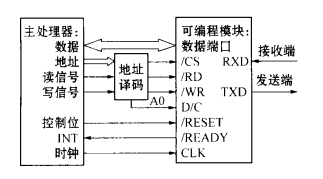


图3 可编程器件与主处理器连接框图

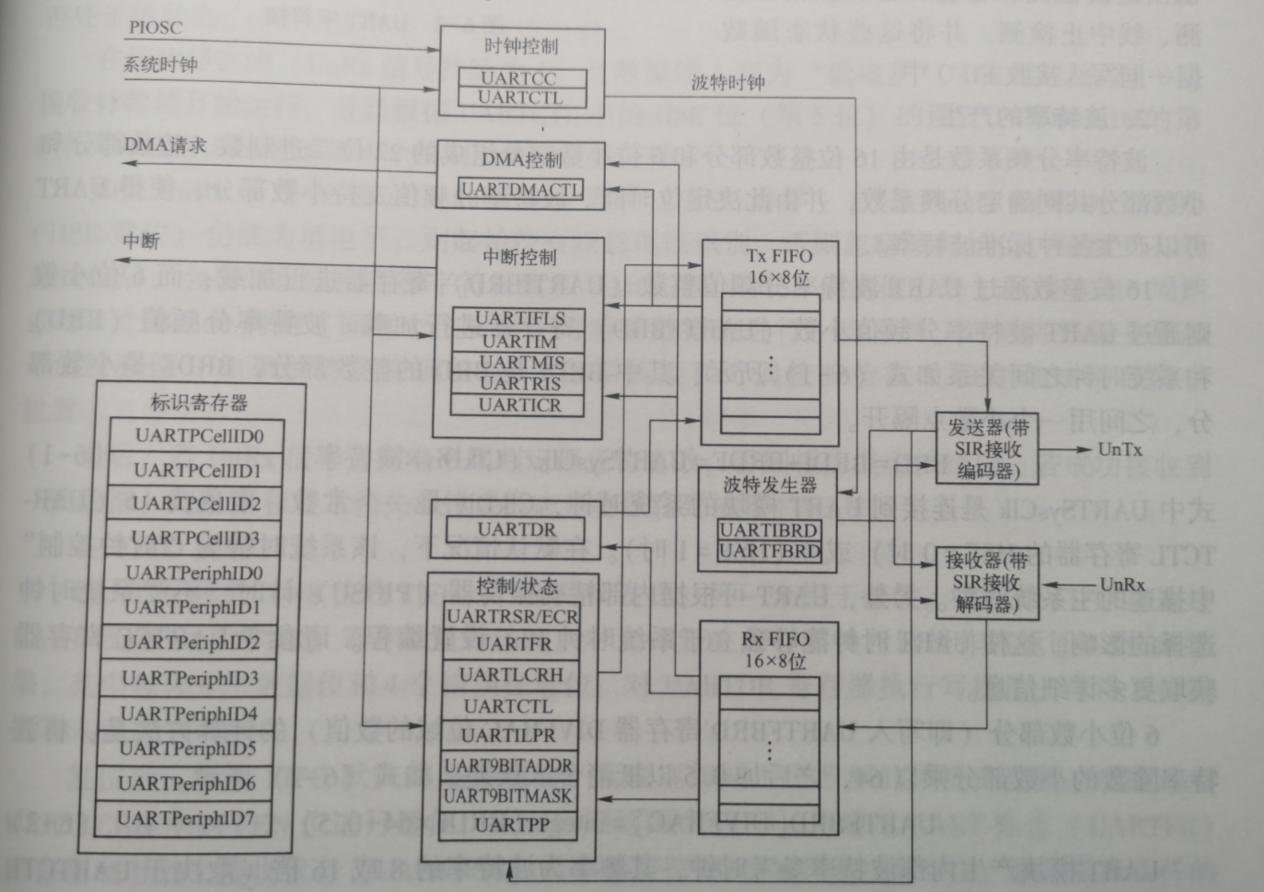


图4 UART内部结构

六.数据传输

数据在接收或发送时各保存在16字节深的FIFO中,接收FIFO的每个单元还有额外4位保存状态信息。当需要进行发送时，先将数据写入发送FIFO。若UART模块已经使能，则将按UARTLCRH 寄存器所配置的参数开始发送数据帧。UART模块会持续发送数据，直到发送 FIFO中没有可发数据为止。数据一经写入发送FIFO（如果该FIFO不为空)，UART标志（UARTFR）寄存器中的BUSY位即会生效,并且在数据发送期间一直保持有效。只有当发送FIFO已空，并且最后1个字符（包括停止位)已经从移位寄存器中发出后，BUSY位才会失效。即使UART模块不再使能,此标志位也能指示出UART是否处于忙状态。

在接收器空闲（UnRx信号持续为1)且数据输入变为“低电平”(收到起始位）时，接收计数器开始运行，并且根据UARTCTL中的HSE位(第5位)的设置,在Baud16的第8个周期或者Baud8的第4个周期对数据进行采样。

如果UnRx信号在Baud16的第8个周期（HSE清零)或者Baud8的第4个周期(HSE置位）仍然为低电平，则起始位有效且可以识别,否则忽略该起始位。检测到有效起始位后，会按照设定的数据字符长度和UARTCTL中HSE位的值，每16个Baud16周期或每8个Baud8周期（即每个位周期）对后续数据位进行一次采样。之后将捕捉并校验奇偶校验位（如果使能了奇偶校验)。数据长度和奇偶校验位在UARTLCRH寄存器中设置。

最后,若UnRx信号为高电平则判定停止位有效，否则视为发生帧错误。若成功接收到一帧数据，则数据和与之相关的错误标志都将保存到接收FIFO中。

七.FIFO操作

UART有两个16×8位的FIFO；一个用于发送，另一个用于接收。这两个FIFO都通过UART数据（UARTDR）寄存器进行访问。对UARTDR寄存器执行读操作将返回12位的结果，其中包含8个数据位和4个错误标志位；对UARTDR寄存器执行写操作，可将8位数据写入发送FIFO中。

复位后，两个FIFO默认都是禁用的，其表现如同1字节深的保持寄存器。可通过对UARTLCRH 中的FEN位进行置位，从而启用这两个FIFO。可通过UART标志（UARTFR）寄存器和UART 接收状态（UARTRSR）寄存器监控FIFO的状态。而对空、满和溢出条件的监控则是由硬件来完成的。UARTFR 寄存器包含空和满的标志（TXFE、TXFF、RXFE和RXFF位)，而UARTRSR寄存器则通过OE位指示溢出状态。如果FIFO被禁用,将根据1字节深的保持寄存器的状态设置空和满标志。

令FIFO产生中断的触发点是通过UART中断FIFO深度选择（UARTIFLS）寄存器来控制的。两个FIFO可分别配置为不同的触发深度。可选的触发深度包括1/8、1/4、1/2、3/4和7/8。举例来说，若设置接收FIFO的触发深度为1/4，则当UART连续收到4个数据字节后即会产生一个接收中断。复位后两个FIFO的默认触发深度都是1/2。

八.UART功能在TM4C1294板子上的应用

主要使用的函数有以下几个：

SysCtlClockFreqSet();//设置系统的晶振频率

SysCtlPeripheralEnable();//使能板子的相应外设

IntMasterEnable();//开启进程中断

GPIOPinConfigure();

GPIOPinTypeUART();//将引脚设置为uart模式

下面的函数可以用来配置板子的uart功能：

void uartconfigure(){

g\_ui32SysClock = SysCtlClockFreqSet((SYSCTL\_XTAL\_25MHZ |

SYSCTL\_OSC\_MAIN |

SYSCTL\_USE\_PLL |

SYSCTL\_CFG\_VCO\_480),120000000);

SysCtlPeripheralEnable(SYSCTL\_PERIPH\_UART0);

SysCtlPeripheralEnable(SYSCTL\_PERIPH\_GPIOA);//uart0对应的外设引脚为PA0，PA1

//设置PA0，PA1为uart引脚

GPIOPinConfigure(GPIO\_PA0\_U0RX);

GPIOPinConfigure(GPIO\_PA1\_U0TX);

GPIOPinTypeUART(GPIO\_PORTA\_BASE, GPIO\_PIN\_0 | GPIO\_PIN\_1);

//设置波特率115200，数据位 8 ，校验位 None ，停止位 1 ，8——N——1模式

UARTConfigSetExpClk(UART0\_BASE, g\_ui32SysClock, 115200,

(UART\_CONFIG\_WLEN\_8 | UART\_CONFIG\_STOP\_ONE |

UART\_CONFIG\_PAR\_NONE));

//开启uart0中断

IntEnable(INT\_UART0);

UARTIntEnable(UART0\_BASE, UART\_INT\_RX | UART\_INT\_RT);

}

在使用uart向串口发送数据接收数据时，一般可以使用以下几个函数：

UARTprintf(); //向串口打印字符串，类似C语言

UARTgets();

UARTCharPutNonBlocking();//向串口发送字符

UARTChargetNonBlocking();

使用上面前两个函数需要#include "utils/uartstdio.h"，并且将uartstdio.c复制到main()同级目录下。

在实际中，如果用上面的配置方式，UARTprintf()函数不能正常使用，在上面配置函数最后加入：

UARTStdioConfig(0, 115200, g\_ui32SysClock);

则可以正常使用，最后查看UARTStdioConfig()函数的声明，发现其内部是由UARTConfigSetExpClk()函数实现的。

组内分工情况：

陈腾朝：摘要，UART定义，UART的优点，UART与IIC、SPI的区别

周威：UART功能与特点，UART功能的实现

傅瞻远：数据传输，FIFO操作，UART功能在TM4C1294板子上的应用