第**11**章异步串口原理和通信的实现

实现异步串行通信是计算机系统最基本的外设功能之一，也是读者学习更复杂计算机外 设的基础。本章将详细介绍BS-232标准，以及龙芯1B处理器内建的异步申行收发器模块 的原理，并通过设计实例说明该模块的使用方法。

本章主要内容包括RS-232协议规范、龙芯1B处理器中UART模块原理。通过这些内 容的介绍，读者能够掌握异步申行通信的原理，并能够掌握使用汇编语言和C语言编写异 步串口通信实现代码的方法。

1. RS-232协议规范

RS-232是美国电子工业联盟(Electronic】 Industries Association, EIA)制定的串行数据 通信的接口标准，原始编号的全称是EIA-RS-232 (简称232, RS-232),它被广泛用于计 算机串行接口外设的连接。

在RS-232C标准中，232是标识号，C代表RS-232的第三次修改(1969年)，在这之 前，还有 RS-232B 和 RS-232A0

目前的最新版本是由EIA所发布的TIA-232-F,它同时也是美国国家标准ANSI /TIA- 232-F-1997 (R2002),此标准于2002年确认。在1997年由TIA/EIA发布当时的编号则是 TIA/EIA-232-F 与 ANSI/TIA/EIA-232-F-1997,在此之前的版本是 TIA/EIA-232-E。

RS-232标准规定了传输数据所使用的连接电缆和机械、电气特性、信号功能及传送过 程。基于这个标准基础，派生出其他电气标准，包括EIA-RS-422-A、EIA-RS-423A、 EIA-RS-485。

目前，在计算机上的COM1和COM2接口就是RS-232C接口。

注：在最新的计算机和笔记本电脑中，均不再提供这种接口，用户必须通过USB转 串口芯片，在计算机和笔记本电脑上虚拟出一个RS-232串行接口。

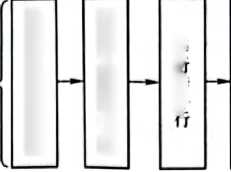
由于RS-232C的重大影响，即使自IBM PC/AT开始改用9针连接器起，目前几乎不再 使用RS-232中规定的25针连接器，但大多数人仍然普遍使用RS-232C来代表此接口。

1. RS-232传输特点

在RS-232标准中，有下面显著的特点。

1. 字符是按一个比特接着另一个比特的方式，使用一根信号线进行传输的。这就是 通常所说的以串行方式传输数据，这种传输方式的优点是传输线少、连线简单、传送距离 较远。
2. 对于信源(发送方)来说，需要将原始的并行数据封装(也称为打包)，然后转

换成一位一位的申行比特流数据进行发送；对于信宿(目的方)来说，当接收到申行比特 流数据后，对接收到的数据进行“拆解”，即从所接收到的申行比特流数据中找出原始比特 流数据的信息，将原始的比特流数据转换成并行数据，如图11.】所示。



原始字节数据

打包后的数据

串行电缆

本地时钟 信宿

提取有%的位数据

检測到比特流数据

得到并行有效敖据

*Ft //* CJ X4/

本地时钟

信源

ra ii.i

异步申行通信原理

1. 在从信源(发送方)通过串行电缆发送数据给信宿(目的方)的时候，并不需要 传输时钟信号。当信宿接收到申行数据的时候，会使用信宿本地的时钟对接收到的串行数据 进行采样和解码，然后将数据恢复出来。
2. 此外，通过RS-232在传送数据时，并不需要额外使用一个信号来传送同步信息, 只需在数据头部(header)和尾部(end)加上识别标志，这样就能将数据从“信源”正确 地传送到“信宿”。

用于实现RS-232通信功能的专用芯片，如8251和16550,称为通用异步接收发送器 (Universal Asynchronous Receiver Transmitter, UART) o

11.1.2 RS-232数据传输格式

RS-232中使用的编码格式是异步起停数据格式，如图11.2所示。在该数据格式中：

起止式异步协议的帧格式



第(Al)个字符

奇校

停止位

偶验

第"个字符

**7**位数据

第**(\*1)**个字符

奇偶**5**

校验位起始位 **7**位数据

0/1 0/1 1 0 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 I 0 0/1 0/1 0/1 0/1

低位 高位 下降边指出下

一字符的开始

图11.2 RS-232数据格式

1. 首先有一个逻辑0标识的起始位，该位标识新的一帧数据的开始。
2. 在起始位后面紧跟以串行格式存在的5-8个数据位(常见的是8位数据)，数据 的起始位对应于原始字节数据的LSB,数据的结束位对应于原始字节数据的MSB。
3. 最后一个数据位后面跟随奇偶校验位(可选)。可以在发送数据的时候设置是否需 要奇偶校验位，且发送方和接收方使用相同的奇偶校验设置。
4. 在可选的奇偶校验位后面跟着以逻辑“1”标识的1~2个停止位。发送方在发送

数据之前设置停止位的个数，旦发送方和接收方使用相同的停止位设置。

从上面的描述可知，在KS-232的数据格式中，如果发送一个有效字符数据字节(8位 表示)，则至少需要10位(至少需要一个数据位和一个停止位)。

在该数据格式中，每一位持续的时间与发送数据的时钟频率有关，即发送方以多快的速 度发送一个比特位。通常，将发送数据的时仲称为波特率时钟，用波特率表示，即每秒发送 位的个数。在基于KS-232的异步申行通信中，要求发送方和接收方使用相同的波特率 时钟。

11.1.3 RS-232电气标准

在RS-232标准中，其分别定义了逻辑“1”和逻辑“0”的电压范围:

1. 逻辑1的电压范围为-15--3V；
2. 逻辑0的电压范围为+3~ + 15V。

在RS-232中，接近零的电平是无效的。显然,RS-232中对逻辑“0”和逻辑“1”的 定义与数字逻辑对逻辑“1”和逻辑“0”的定义不同。为了让遵守RS-232电气标准的电 平信号与标准数字逻辑电平信号进行连接，需要执行电气标准转换，即把满足数字逻辑标准 的7TI/CMOS电平转换为RS-232电平，以及将RS-232电平转换为TTL/CMOS电平。例 如，美信(maxim)公司的MAX232芯片可以实现TTL/CMOS也平与RS-232电平之间的双 向转换，如图11. 3所示。



+5V INPUT

C3

卄

| ci+EL | | + U |
| --- | --- | --- |
| **V** | *2* |  |
| **C1-** | **3** |  |
| **C2・** | **4** | **MAX220**  **MAX232** |
| **C2-** | **5** | **MAX232A** |
| **V-** | **L±** |  |
| **T2OUT**巨 | |  |
| **R21N** |  |  |

E Vcc 回GND 回 T1OUT

0] RUN 可 R1OUT

7T| T1IN 可 T2IN

可 R2OUT

|  | **Vet**  **♦5V** 到 **HOV V+** |
| --- | --- |
| **C1-** | 电压倍増器 |
| **C2+** | +伽到-啊 |
| **C2-** | 电压反相器 |

16

6

♦5V- 400kQ tun

C电

+ I0V

II

TTL/CMOS.  
输人

♦5V.

TIOUT

RS-232

输出

| **12** | **R1OUT /** | **RUN** | **13** |
| --- | --- | --- | --- |
| *9* | **R2OUT /** | **A**  **IJskn**  **~ R2IN** | d |
|  | **GND** | **|J5kQ** |  |

T2OUT

10

7

400kQ T2IN

TTL/CMOS 输出

RS-232

'输入

图11.3在TTL/C0MS与RS-232之间进行电平转换的芯片

当把龙芯1B处理器的专用申行通信接口引脚连接到MAX232芯片上时，就可以实现龙 芯1B处理器通过串口电缆与其他设备进行通信。

11.1.4 RS-232参数设置

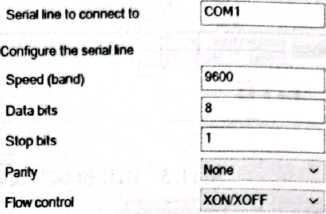
打开PulTY软件，单击左侧的Serial选项,在右侧窗口中可以看到串口的参数设置界 面,如图11. 4所示。在该界面中,可以设置波特率［Speed(band)］、奇偶校验(Parity), 停止位(Stop bits)，以及流量控制(Flow control)等参数。下面对这些参数的含义进行简 单介绍。

PulTY Configuration

Category

P Session logging o Terminal Keyboard Bel Features

S Window Appearance Behaviour Translation

i ^.Selection  
Colours  
g Connection  
r Data  
Proxy  
田SSH

Options controlfcng focal serial Ines

Seied a senal lv)e

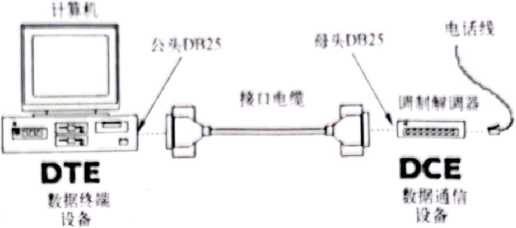
r Telnet -Rlogin SUPOUP

图11.4串口的参数设置界面

1. 波特率。它是指将数据从一个设备发送到另一个设备的速率，使用每秒钟发送位 的个数来度量，单位为波特率(bits per second, bps)o例如，可选择的波特率有300、 1200、2400、9600、19200、115200 等。
2. 奇偶校验。奇偶校验用于验证接收数据的正确性。一般不使用奇偶校验，如果使 用，那么既可以选择设置为奇校验也可以选择设置为偶校验。在偶校验中，要求所有发送数 据的位(包括校验位)中“1”的个数是偶数。根据这个校验标准，将奇偶校验位置“1” 或置“0”；在奇校验中，要求所有发送数据的位(包括校验位在内)中“1”的个数是奇 数。根据这个校验标准，将校验位置“1”或置“0”。
3. 停止位。停止位是在发送完数据的最后一位或奇偶校验位之后发送的，用于帮助 实现串口通信的接收方实现硬件的重新同步。例如，在以申行方式传输8位原始数据 “11001010”时，数据的前后加入起始位(以逻辑“0”表示)和停止位(以逻辑 T 表 示)。就需要在串行停止位可以是1位、1.5位或2位。
4. 流量控制。当需要发送握手信号或对数据完整性进行检测时，需要额外的信号进 行协助，这些额外的信号包括RTS/CTS和DTR/DSR,通常这些信号用于帮助RS-232实现 硬件流量控制。当串行通信需要流量控制时就要在发送方和接收方之间连接这些信号絞，以 保证通信过程的可靠性。一般情况下为了简化硬件信号的连接和降低控制的复杂度，不使用 用于硬件流量控制的信号。

11. 1.5 RS-232 连援器

RS-2A2 i'Jil之机1 HH.ii拄调，伸朔器做他输之用的.囚此它的引脚定 5常也和阙 3伸调狀!{•输 ” y HS - 2 V 的 i'J 备 nJ 以分为 招终罚 i7 备（Ihta krtninal Kqiiipmpnt. DTI'.如「（：）科傲IK讷俏）备（**I）Ma** Comnwinic^tinn Equipment, ME.如调—解调群）两 \*. UH-分r定z r 4、同的n.略t无送和u收（,,；■ **b** .般\*说,ii w帆和终端役*备*使用 im途拭舞.调制解调詐和，I印机ft **hi** rx.r.连搏詳.如图n.5*所示*



R 11.5 DTE和OU：设备的连枝

KS-232指定了 20个不同的信号连接，由25个D-sub （■型D类）引如构成的DR-25 违按器很多设备只使用了其中的部分引脚，出于节省资金和空间的貯虑,不少机器采用较 小的迁技器.特别是9引脚的或若是DB-9型的连接器.其广泛使用在绝大多" IIIM的AT机之后的计算机和J\*他许多设备上.DB-25和DB-9型的连接器在大部分设备上 是母头（插孔），但并不一定都是这样，有些设备上就是公头（插针）,

图11.6给出了【）11 - 9连接器公头和母头连接器的信号定义顺序 毎个信号的定飞盆 A 11.1所示。

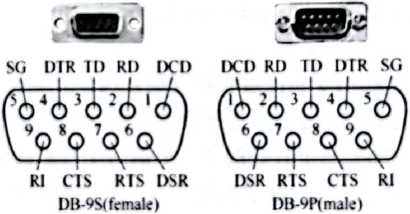


图11.6 RS-232串口连接群-母头（female）和公头（male）

表11.1 DB-9连接器信号的定义

|  |  |  |
| --- | --- | --- |
| 引脚名字 | 序母 | 功 能 |
| 公**Mttte**（、，） | **5** | 地线 |
| 发送锻据**（IIMTXIJ）** | **3** | 发送牧据 |
| 技受段据**（HD/RXD）** | *2* | 推收牧据 |
| 數据縱\*帝备**([k»U Trnmibil Ke^y, DTK)** | **4** | 終端设备通**M**周制解调器叮以进行故据传输 |
| **tlft**(市备好**(1UI< Sri H“d>, D'K)** | **6** | 阔机解凋器通知终编汶备准备就绪 |

第11章 异步串口原理和通信的实现

请求发送(Request To Send, RTS)

7 终端设备要求调制解调器提交散据

清除发送(Clear To Send, CTS)

8 调制解*调*器通知终端设备可以传敎据过来

数据载波检测(Carrier Detectt CD)

调制解调器通知终端设备侦听到我波信拈

振铃指示(Ring Indicator, R[)

9 调制解调器通知终端设备有电话进来

11.2.2 UART寄存器组的功能

本小节将介绍每个UART模块中寄存器的定义及功能。

11.2龙芯1B处理器中UART模块原理

龙芯1B处理器集成的通用异步申行收发器(Universal Asynchronous Receiver/

Transmitter, UART)模块兼容标准的16550A芯片，该UART模块的内部结构如图11. 7 所示。

图11.7龙芯1B处理器内建的UART模块的内部结构

11.*2.*1 UART寄存器组的基地址

龙芯1B处理器中内建了 12个可并行工作的UART接口，这些接口的内部寄存器完全相 同，但是它们的基地址不同，如表11.2所示。

表11.2 UART接口寄存器的基地址

分頻器

总线接口

中断仲裁模块

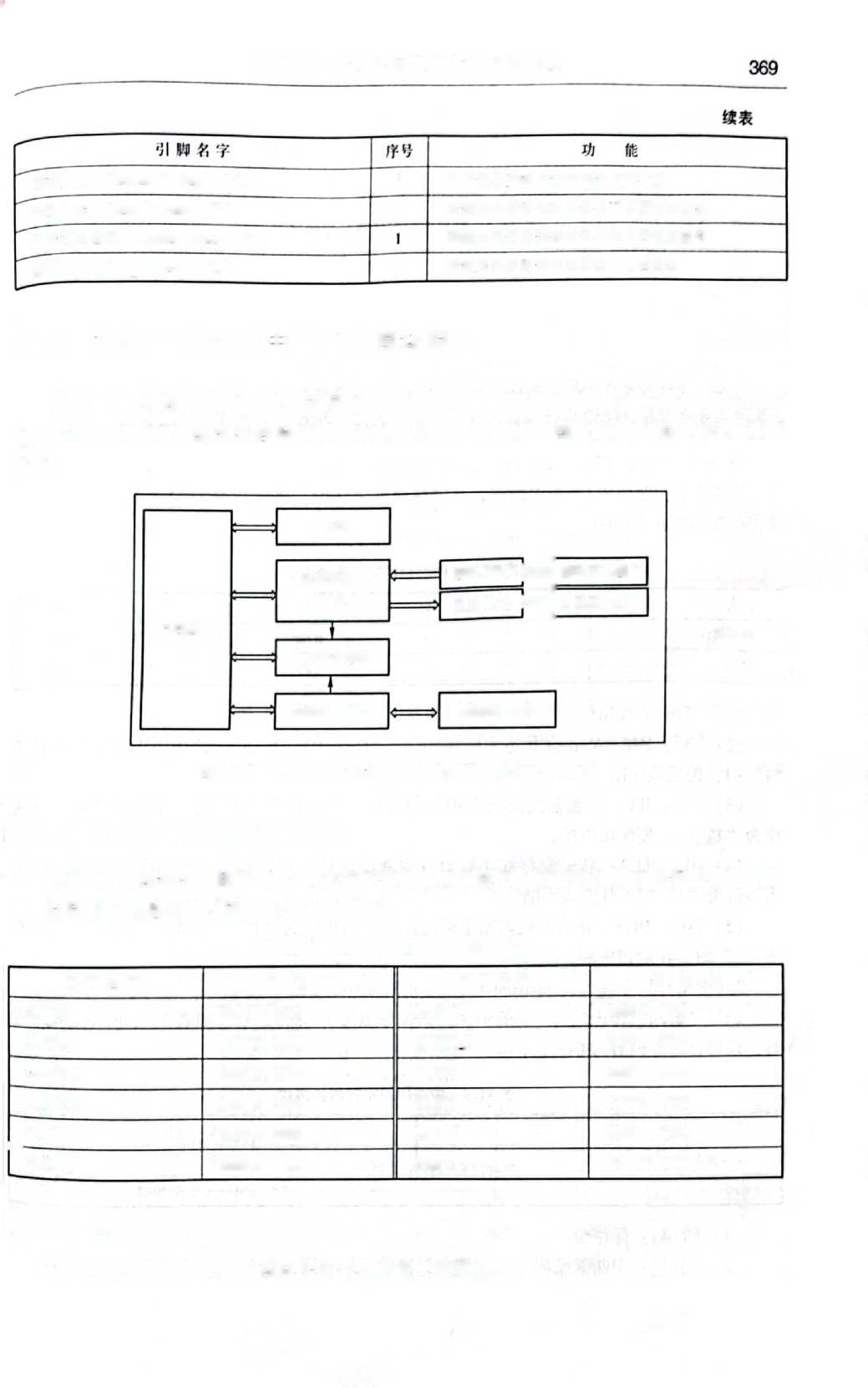
Modem 接 口

Modem寄存器

访何寄存器 模块

|  |  |
| --- | --- |
| 接收**FIFO** | =|接收模块援口 |
| 发送**FIFO** | =可发送模块接口 |

|  |  |  |  |
| --- | --- | --- | --- |
| 接口名字 | 破地址 | 接口名字 | 任地址 |
| **UART0** | **0xBFE4 0000** | **UARTI.2** | **0xBFE4 6000** |
| **UART0**」 | **0xBFE4 1000** | **UART1 3** | **0xBFE4 7000** |
| **UART0\_2** | **0xBFE4 2000** | **UART2** | **0xBFE4 8000** |
| **VART0J** | **0xBFE4 3000** | **UART3** | **0xBFE4 C000** |
| **\_ L'ARTl** | **0xBFE4 4000** | **UART4** | **OxBFE6 C000** |
| **UART1J** | **0.BFE4 5000** | **UART5** | **OxBEF7 C000** |



1. 敦据传稔保持寄存器(Data Transfer Hold Register, DTHR)

该寄存器保存着要发送/接收的数据，其偏移地址为0x00,格式如表11.3所示，当复 位时，该寄存器的内容为0x00。

表11.3数据传输保持寄存器的格式

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 位索引 | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| 读写厲性 |  |  |  | **W/R** | |  |  |  |
| 名字 | **Tx FIFO/Rx FIFO** | | | | | | | |

注：当对■该寄存器执行写操作时，写到该寄存器的内容为要发送的数据(TxFIFO)； 当对■该寄存器执行读操作时，读取该寄存器的内容为接收到的歎据(RxFIFO)。

1. 中断使能寄存器(Interrupt Enable Register, IER)

该寄存器控制使能/屏蔽中断源，其偏移地址为0x01,格式如表11.4所示。当复位时, 该寄存器的内容为0x00。

表11.4中断使能寄存器的格式

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 位索引 | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| 读写属性 | —— | **——** | — | **——** | **R/W** | **R/W** | **R/W** | **R/W** |
| 名字 | — | **—** | —— | **—** | **IME** | **ILE** | **ITxE** | 【臨 |

1. [7：4]：保留位。
2. [3]： IME0 Modem状态中断使能位。当该位为“0”时，禁止该中断；当该位为 “1”时，使能该中断。
3. [2]： ILE。接收器线路状态中断使能位。当该位为“0”时，禁止该中断；当该 位为“1”时，使能该中断。
4. [1]： ITxE。传输保存寄存器为空中断使能位。当该位为“0”时，禁止该中断； 当该位为“1”时，使能该中断。
5. [0]： IRxE。接收有效数据中断使能位。当该位为“0”时，禁止该中断；当该位 为“1”时，使能该中断。
6. 中断标识寄存器(Interrupt Indication Register, HR)

该寄存器用于标识当前产生的中断，其偏移地址为0x02,格式如表11.5所示。当复位 时，该寄存器的内容为Oxcl。

表11.5中断标识寄存器的格式

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 位索引 | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| 读写属性 | **—** | —— | —— | **—** | **R** | | | **R** |
| 名字 | **—** | —— | —— | **—** | 中断源编码(见表**10.7)** | | | **INTp** |

1. [7：4]；保留位。
2. [3：1]：中断源编码字段，其含义如表11. 6所示。

裏11.6中断源協码字用的含义

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 而i | 2 1 | |  | 中 Kft t V | 中斯拂 | 中♦垠位曾耕 |
| **Q** | **1** | **1** | 护’■ | 河收线酬状态 | 有偶、湛出成軸**VUV.**或“断中商 | **K I5R** |
| **0** | **1** | **0** | 第 | It收到“我做此 | **HK)**中"，针个敬达利触寇的俏 |  |
| **1** | **1** | **0** | 弟 | 传收縮时 | 在**FIFO**中¥少有-个宁竹.倒在」个**7** | —— |
| **0** | **0** | **1** |  | 传輸仰"寄"關为十 | 传■供**4**有**“t»**为空 | 读|出 |
| **0** | **0** | **0** | 第四 |  | **CT**、、**IrM. Rl *A* IKJI** | 读**MW** |

(3) [0]：中断表示位。当该位为“0”时，正在挂起一个中断;当该位为-I"时,

没有挂起的中断。

1. FIFO 控制寄存器(HFO Control Register, FCR)

该寄存器用于设***K*** FIFO触发中断的门限，H偏样地址为0x02,格式如占11.7所示 当促位时，该寄存器的内容为0xc0o在UART模块中,FIFO的深度为16。

表11.7 FIFO控制寄存器的格式

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 位索引 | **7** | **6** | **3** | **4** | **3** | **2** | **1** | **0** |
| 该"属性 | **W** | | — | — |  | w | W |  |
| 名字 | **TL** | | — | — | — |  | Riwi | **——** |

1. [7：6]; TL。设置用于触发FIFO中断的门限。当该字段为“00”时，触发FIFO 中断的门限为1个字节；当该字段为“01”时，触发FIFO中断的门限为4个字节；'，该字 段为“10”时，触发FIFO中断的门限为8个字节；当该字段为“II”时，触发FIFO中断 的门限为14个字节。
2. [5：3]：保留位。
3. [2]： Txset。当该位置“1”时，淸除发送FIFO的内容，并复位其逻辑,
4. ⑴：Rxseto当该位置“1”时，淸除接收FIFO的内容，并夏位其逻辑，
5. [0]：保留位。

注：中断标识寄存器和FIFO控制寄存器共用一个存储空间地址，使用读/写操作来 区分这两个寄存器。当执行写寄存器操作时，将值写入FIFO控制寄存為；当执行读寄 存器操作时，将从中断标识寄存器读取中断状态标志。

1. 线路控制寄存器([Jne Control Register, LCK)

该寄存器用于设于线路的工作状态，其偏移地址为0x03 .格式如式11.8所示。当复位 时，该寄存器的内容为0x03。

表11.8线路控制寄存器的格式

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 位索引  ,― | **7** | **6** | **5** | **4** | **3** | *2* | **1** | **0** |
| 读写**M**性  — | **K/W** | **K/W** | **R/W** | **H/W** | **R/U** | **R/W** | **K/W** | |
| \_名字 | **IJLAB** | **BCB** | **SPB** | **EPS** | **PE** | **SB** | **HEC** | |

1. [7]： DLAB。分頻锁存器访问控制位。当该位为 T 时,访问操作分频锁存器; 哗该位为“0”时，访问操作正常寄存器。
2. [6]； BCB。打断控制位。当该位为 T 时,将申口的输出设置为0 (打断状 态)；当该位为“0“时,正常操作。
3. [5]： SPB0指定奇偶校验控制位。当该位为“0”时，不用指定奇偶校验位;当 该位为 T 时,如果LCR[4]位是 T,则传输和检査奇偶校验位为“0"；如果LCR[4] 位为“0”，则传输和检査奇偶校验位为T。
4. [4]： EPS。奇偶校验选择控制位。当该位为“0”时,在每个字符中有奇数个 T (包括数据和奇偶校验位);当该位为“1”时,在每个字符中有偶数个 T。
5. [3]： PE。奇偶校验位使能控制位。当该位为“0”时，没有奇偶校验位；当该位 为“1”时，在输出时生成奇偶校验位，输入则判断奇偶校验位。
6. [2]： SB。定义生成的停止位的个数。当该位为“0”时，1个停止位；当该位为 “1”时，在5位字符长度时是1.5个停止位，其他长度是2个停止位。
7. [1：0]: BEC。设置每个字符的位数。当该字段为“00”时，数据K度为5位；当 该字段为“01”时，数据长度为6位；当该字段为“10”时，数据长度为7位；当该字段 为“11”时，数据长度为8位。

6) Modem 控制寄存器(Modem Control Register, MCR)

该寄存器用于设置线路的工作状态，其偏移地址为0x04,格式如表11.9所示。当复位 时，该寄存器的内容为0x00。

表11.9 Modem控制寄存器的格式

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 位索引 | **7** | **6** | **5** | **4** | **3** | *2* | **1** | **0** |
| 读写届性 | — | — | —— | **W** | **W** | **W** | **W** | **W** |
| 名字 | —— | —— | —— | **LOOP** | **OIT2** | **OUT1** | **RTSC** | **DTRC** |

1. [7：5]：保留位。
2. [4]： LOOP。回路模式控制位。当该位为“0”时,正常操作；当该位为T 时, 为环路模式。在环路模式中，TXD输出持续为T,输出移位寄存器丸接连接到输出移位寄 存器中,其他信号连接关系为DTR连接到DSR、RTS连接到CTS、OUT1连接到RI、OUT2连 接到DCD。
3. [3]： OUT2。在环路模式中连接到DCD的输入。
4. [2]： OUT10在环路模式中连接到R1的输入。
5. [1]： RTSC0 RTS信号控制位控制RTS的输出。
6. [0] ： DTRC。DTR信号控制位控制DTR的输出。

7)线路状态寄存器(Line Status Register, LSR)

该寄存器用于査看线路的工作状态，其偏移地址为0x05,格式如表11. 10所示。当复 位时，该寄存器的内容为0x00。

表11.10线路状态寄存器的格式

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 位索引 | **7** | **6** | **5** | **4** | **3** | *2* | **1** | **0** |
| 读写属性 | **R** | **R** | **R** | **R** | **K** | **R** | **R** | **R**  .-二—-— |
| 名字 | **ERROR** | **TE** | **TFE** | **Bl** | **FE** | **PE** | **OE** | **DR** |

1. ［7］： ERROR。错误标识位。当该位为“1”时，表示发生奇偶校验位错误，或帧 備误或打断中断。当该位为“0”时，表示没有错误。
2. ［6］： TE。传输为空标识位。当该位为“1”时，表示传输FIFO和传输移位寄存 器都为空。给传输FIFO写数据时将清0。当该位为“0”时，表示有数据。
3. ［5］： TFE。传输FIFO为空标识位。当该位为T”时，表示当前传输FIFO为空, 绐传输FIFO写数据时清零；当该位为“0”时，表示当前传输FIFO中有数据。
4. ［4］： BI。打断中断标识位。当该位为“1”时，表示接收到的起始位+数据+奇偶 位+停止位都有0,即有打断中断；当该位为“0”时，表示没有打断中断。
5. ［3］： FE0帧错误标识位。当该位为“1”时，表示接收的数据没有停止位；当该 位为“0"时,表示没有错误。
6. ［2］： PE。奇偶校验位错误标识位。当该位为“1”时，表示当前接收数据有奇偶 構误；当该位为“0”时，表示没有奇偶错误。
7. ［1］： OE。数据溢岀标识位。当该位为“1”时，表示有数据溢出；当该位为“0” 时，表示没有数据溢出。
8. ［0］： DR。接收数据有效标识位。当该位为“1”时，表示在接收FIFO中有数据; 当该位为“0”时，表示在接收FIFO中无数据。

注：在对该寄存器进行读操作时，将LSR［4：1］和LSR［7］清0, LSR［6：5］在给传输 FIFO写数据时清0, LSR［0］则时接收FIFO进行判断。

8) Modem 状态寄存器(Modem Status Register, MSR)

该寄存器用于査看Modem的工作状态，其偏移地址为0x06,格式如表11. 11所示。当 复位时，该寄存器的内容为0x00。

% 11.11 Modem状态寄存器的格式

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 位娠引 | **7** | **6** | **5** | **4** | **3** | *2* | **1** | **0** |
| 读写属性 | **R** | **R** | **R** | **R** | **R** | **R** | **R** | **R** |
| 名字 | **CDCD** | **CR!** | **CDSR** | **CCTS** | **DDCD** | **TERI** | **DDSR** | **DCTS** |

1. ［7］： CDCDO对DCD输入值取反，或者在环路模式中连接到OUT2。
2. ［6］： CBI0对RI输入值取反，或者在环路模式中连接到OUT1。
3. ［5］： CDSR0对DSR输入值取反，或者在环路模式中连接到DTR。
4. ［4］： CCTSO对CTS输入值取反，或者在环路模式中连接到RTS。
5. ［3］： DDCD。DDCD 指示位。
6. ［2］； TERIe R［边沿检测。RI状态从低到高变化。
7. ［1］： DDSR。DDSR 指示位。
8. ［0］： DCTS0 DCTS 指示位。

9)分频锁存器

分频锁存器由分频锁存器1和分配锁存器2两个寄存器构成，用于生成指定的波特率时 钟，其偏移地址分别为0x00和0x01,格式如表11. 12和表11.13所示。当复位时，这两个 寄存器的内容均为0x00。

表**11. 12**分频锁存器**1**的格式(低**8**位)

|  |  |
| --- | --- |
| 位索引 | 7 6 ~~5~~ |—TZLZ-L ~~2~~ ~~1 '~~-LA- |
| 读写属性 | **- R/W** |
| 名字 | 存放分频锁存器的低**8**位 **J** |

表**11.13**分频锁存器**2**的格式(高**8**位)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 位索引 | **—7** |  | **is ~r~~r 3\_\_\_\_?\_\_1—** | '1 ° |
| 读写属性  字 |  |  | **R/W** 一  存放分頻做存器的髙**8**位 |  |

注：UART模块的时钟频率是DDR.clk频率的一半。

11.3 PuTTY工具的下载和安装

PuTTY是一个集成虚拟终端、系统控制台和网络文件传输为一体的自由及开放源代码 的程序。它支持多种网络协议，包括SCP、SSH、Telnet、rlogin和原始的套接字连接。它也 可以连接到串行端口。

该工具的早期版本仅支持Windows平台，之后陆续增加对各类Unix平台和Mac OS X的 支持。

下载和安装PuTTY工具的主要步骤如下。

1. 在 IE 浏览器中，输入 <https://www>. putty, org/ 网址。
2. 弹出新的页面，如图11. 8所示。在该页面中，找到Download PuTFY标题。在该标 题下面，找到 You can download PuTTY here。单击 here。

Download PuTTY

PuTTY is an SSH and telnet client, developed originally by Simon Tatham for the Windows platform. PuTTY is open source software that is available with source code and is developed and supported by fi group of volunteers.

You can download PuTTY here.

图**11, 8 PuTTY**工具下载入口

1. 自动跳转到 https：//www. chiark. greenend. org. uk/-sgtatham/putty/latest. hlml,并打 开新的页面，如图11.9所示。

**Package files**

You probably want one of These. They indude versions of all the PuTTY utilities. (Not swe whether you want the 32 bit or the 64 bit vers ton? Read the FAQ f ntiy)

MSI( "Windows InsUlter\* )

64 bit x86 Q.；。山uju a

64-bit Arm: uiufSP,

32-bit x86: 。邃 qL匚 a

Unix sourct trchiv\*

**a n： tuurflx^ u；. tx**

(or by FTP) for by FTP! (or by FTP)

I yig nature! (ZgoaMff] (ygnilyrfil

(or by FTP)

細皿Ml

图**11.9 PuTTY**工具下载页面

1. 在该页面中，鼠标右键单击putty-64hit-0. 76 - installer, msi,弹出浮动菜单。在浮 効菜单内，选择将链接另存为…选项。
2. 弹出另存为对话框。在该对话框中，选择保存文件的位置，该文件的名字为putty- 54bit-O. 76-installer. msi0
3. 在保存该文件的文件夹中，找到并双击putty-64bit-0. 76-installer. msi,启动安装 it程。
4. 弹出PulTY release 0.76 (64-bit) Setup对话框,如图11. 10所示。在该对话框中, 提示"Welcome to the PulTY release 0. 76 (64-bit) Setup Wizard"信息。

*B* i^uTTY release 0.76 (64-b由 Setup — X



Welcome to the PuTTY release 0.76 (64-bit) Setup Wizard

The SeU) you to change tw way PuTTIf

rHeave 0 75 (645) *art* hstaled on you Etvtw or to rmove it from yoJ cortiuter. Ock Next to contrue or Caned Io ewt fre SeS

Nrrt ]

图 **11. 10 PuTTY rrlraM- 0. 76 (M-bil) Setup** 对话桐**(1)**

1. 单击Nel按钮。
2. 弾出 PulTY release 0. 76 (64-bit) Setup-Destination Folder 对话框，如图 11. 11 所

示。在该对话在中■提不"Click Next to inMall Io the default folder or click Change to choose an- other0 选择默认的安装路径。

号 **PuTTY** *0 76* **(64 btl) Setup**

Dcsttnatton FoMer

Nett to tmtal to the (MwK folder or d»d( to dmosc another.

PuTTY mm 0 % (M M) to:

Change一

图 11.11 PuTTY release 0, 76 ( 64-bit) Setup-Destination Folder 对话框



1. 单击Next按钮。
2. 弹'll Pul'I'Y release 0. 76 (64-bit) Setup-Product Features 对活框，如图 Il.|2 所 示。在该对活框中，单击Add shortcut to PulTY cn the Desktop前面的按钮亘，出现浮动荣 单。在浮动菜单内，选择 Will be installed on local hard drive。

PuTTY release 0.76 (64-bit) Setup 一 x

Product Features

Select the way you want features to be Installed.

Install PuTTYfiles

[Add shortcut to PUTTY on tM Destop—  
Will be installed on local hard drive

5 Entire feature will be installed on local hard drive

\* Entire feature will be unavailable

This feature requires 5KB on your hard drive.



Cancel

图 11. 12 PuTTY release 0. 76 (64-bit) Setup-Product Features 对话框

1. 单击Install按钮。
2. 弹出 PuTTY release 0. 76 (64-bit) Setup-Installing PuTTY release 0. 76 (64-bit) 对话框，如图11.13所示。在该对活框中，将要显示安装进度。

勘 PuTTY release 0.76 (64 bit) Setup

Installing PuTTY release 0.76 (64-btt)

Please wait while the Set叩 Wizard Installs 印TTY release 0.76 (64 顷).

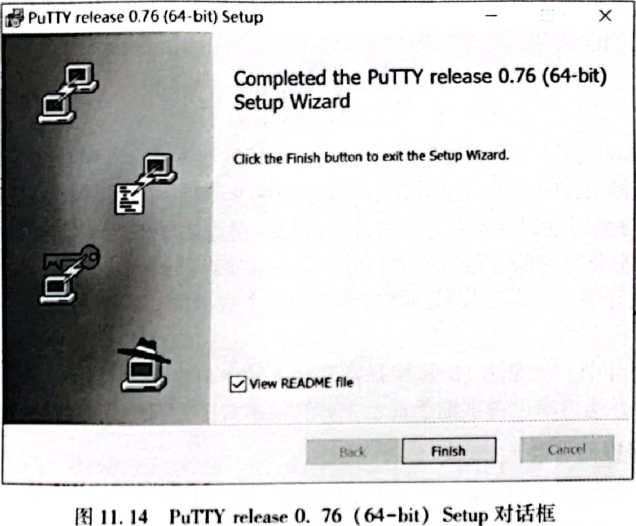
Back Kext [ Cancel

图 11. 13 PuTTY release 0. 76 ( 64-bit) Setup-Installing PuTD' release 0. 76 ( 64-bit)对话框

(】4)弹出用于账户控制对话框。在该对话框中，提示“你要允许来自未知发布者的此 应用对你的设备进行更改吗?”。

—

1. 单击是按钮，退出该对话框。
2. 安装完成后，自动弹出新的PuTTY release 0. 76 ( 64-bil) Setup对话框，如图11. 14 所示O 在该对话框中,提示"Completed the PuTTY release 0. 76 ( 64-bit) Setup Wizard"信息。



1. 单击Finish按钮,退出该对话框。
2. 当成功安装该软件工JI后，在Windows 10操作系统桌面上出现名字为“PuTTY (64-bit),•的图标。此外，通过Windows 10操作系统的开始->P->PuTTY(64-bit),也可以 找到该软件工具的人口。

**11.4**异步申口通信的设计和实现

本节将介绍使用汇编语汚和c i吾吉实现异歩申口通信的方法。在使用汇编语言程序实 现异步巾口通信时，側禾于对UAKT模块底层寄存器的操作和控制。通过直接操作底层寄存 器的方法，使得读者能够宜观地理解和掌握软件与外设之间的驱动和控制关系。在使用C 诲言实现异步申口通信时，侧重于调用应用程序接口 (Application Program Interface, API) 函数来实现申行通信。通过调用API函数，使得读者能理解和掌握软件的分层设计结构, 能高效率使用C语言中的API函数实现特定应用场景的开发。

**11.4.1**串口通信的汇编语言设计和实现

本节将使用汇编语言编写串口通信的实现代码，并在龙芯1B硬件开发平台上进行测试 和脸证。

1. 预备知识

在使用汇编语言设计申口通信代码之前，需要明确下面的一些信息。

1. 在该设计中，将使用龙芯】B处理器的UART3模块。通过11.2. 1节给出的表格可

378 微型计算机系统原理及应用

—. 知，UART3模块的基地址为0xBFE4C000o该模块中所有寄存器在存储空间中的有效地址都 是由基地址加上寄存器的偏移地址得到的。

1. 在该设计中，申口通信的波特率设置为9600。在11.2.2节介绍分频器寄存器时, 已经说明UART模块的基准时钟頻率是DDR2时钟频率的一半。因为DDR2时钟的频率或 100MHz,所以UART模块的基准时钟频率为100/2 = 50MHzo根据波特率计算公式：

A'ART presca，e=16^d 式中，n®为UART模块的基准时钟；Band为申口传输速率；prescale为分频因子。

因此，可以得到分频因子为(10x")/(i6x9600)= 325。十进制数325对应于十六进制 数145。因此，分频因子的低8位为十六进制数45,分频因子的高8位为十六进制数01。这 两个数分别作为分频时钟低寄存器的值和分频时钟高寄存器的值。

1. 在该设计中，将串口数据的格式设置为8个数据位、1个停止位，无奇偶校验，无 流址控制。
2. 在该设计中，从龙芯1B硬件开发平台上的串口发送字符申“Hell。World!”给主 计算机的串口，并通过串口调试助手显示主计算机申口接收到的数据。
3. 汇编语言程序设计

本部分将介绍如何在LoongIDE中使用汇编语言编写中口通信程序代码，主要歩骤如下°

1. 启动LoongIDE集成开发环境(以下简称ImngIDE)。
2. 在LoongIDE主界面主菜单中,选Project->Open Project,弹出Open对话框。
3. 在Open对话框中，将路径定位到\loongsonlB\_example\examp!e\_l 1\_1路径下。在该 路径下，找到并选中文件example」1\_1. Ixp。
4. 单击【打开】按钮，打开文件example\_il\_l. Ixpo
5. 在LoongIDE主界面左侧的Project Explorer窗口中，找到井双击l>sp\_start. S文件。
6. 在右侧窗口中，输入如代码清单11-1所示的汇编语苻代码。

代码清单**11-1**串行通信的汇编语言代码

/ \*下面为数据段，在数据段中保存了 **UART3**的基地址和耍发送的字符申**string \*/**

**.data** 〃数据段声明

UAHT3\_BASE\_ADDR : .long OxBFE4COOO

string : . asciiz " Hello World !"

/ \*下面为代码段，代码段实现了将数据从龙芯**1B**开发板发送到主计算机的功能\*/

**・text** //代码段声明

FRAME( bsp\_start,sp,O,ra)

, set noreorder

move sO, ra 〃返回地址

/\*下面的代码将使能访问分频锁存器\*/

la vO, UART3\_BASE\_ADDR 〃将 UART3\_BASE\_ADDR 地址加载到寄存器、0 lw s0, (vO) 〃将(vO)指向的存储空间地址内容加载到寄存器s0

〃s0保存着UART3的基地址(KBFE4COOO

〃将线路控制寄存器**ler**中的内容加我到寄存器**tl**

Ibu tl, 3(s0) ori tl, 0x00000080 sb tl, 3(s0)

〃将ler中的dlab位置1

//将控制命令写入辰中，使能访问分频繊存器

/ \*下面的代码用于设置时钟分频锁存器\*/

//将计算得到的分频因子的低**8**位加载到寄存器**(2** 〃将该分频因子写入分频低锁存器

sb

li

sb

12, 0x45 t2, 0( s0) t2, 0x01

t2, l(s0)

//将计算得到的分频因子的髙**8**位加载到寄存器**(2**

〃将该分频因子写入分频高锁存器

/ \*下面的代码将禁止访问分频锁存器，而是正常访问其他寄存器\*/

li si, 0xFFFFFF7F 〃将立即数0xFFFFFF7F加载到寄存器SI

and tl, tl.sl 〃将dlab位重新设置为0

**sb tl, 3(sO)** 〃将控制命令写入**|cr**中,禁止访问分频锁存器

/\*下面的代码设置串行通信的数据格式\*/

ori tl, 0x00000003 //设置数据格式，无奇偶,8位数据,1位停止

sb tl, 3(s0) 〃将数据格式的控制字写入ler

/\*下面的代码将发送数据\*/

〃获取字符申所在的存储空间的起始地址

//标号

〃将所对应存储空间的字符加载到寄存器**t3** //判断是否字符申结尾，如果是，则退出发送过程 //分支指令后面加入**nop**指令，用于延迟隙

Ibu 13, 0( vl) lieq t3,zero,2f nop

sb t3, 0( sO) addu vl, vl, 1 b lb

nop

2：

//将要发送的字符写到数据寄存器**dat**

//地址加**1,**指向下一个要发送字符的存储地址

〃无条件跳转到标号**1,**继续发送

〃分支指令后面加入**nop**指令，川于延迟隙

//恢红返冋地址

move ra, sO j ra nop

〃跳转到恢夏地批的指令

//跳转指令后面加入**nop**指令，用于延退隙

・ set reonler

ENDFRAME(bsp\_start)

1. 保存设计文件。
2. 设计编译和调试

本部分将介绍如何对汇编语言代码进行编译，并将生成的代码下载到龙芯1B开发板进 行验证。主要步骤如下。

(])在LoongIDE主界面主菜单中，选择Project->Compile & Build,对设计进行编译和 链接，并生成可执行文件。

(2)使用Mini-USB电缆，将龙芯1B硬件开发平台上的USB接口与安装有LoongIDE软 件工具的PC/笔记本电脑的USB接口进行连接。

1. 通过龙芯IB开发板套件提供的USH-巾11电缆，将该电缆一端的9针申口( ”头) 连接到龙芯版开发板上标记为UART3的屮口上，将该电缆一端的USB接口连接到PC/% 记本电脑的USB接口上。
2. 给龙芯1B硬件开发平0上电。
3. 在Windows操作系统中，进入设备管理器界面。在该界面中，找到虚拟出来的申 口，如图11.15所示。在本书所使用的电脑中，虚拟出的串口号为C0M5。

\* 0 端口 **(COM** 和 **LPT)**

**@ USB Serial Port (COM3)  
g USB Serial Port (COM4)**

**I & USB-SERIALCH340(COM5j**

图**11. 15**虚拟出来的申口号

注：读者要根据自己计算机上虚拟出来的实际串口号，该串口号可能和书中给出的 并不相同O

1. 启动PulTY (64-bit)软件工具。自动弹Hl PuTTY Configuration对话框，如图11.16 所ZK。在该对话框左侧的Category窗口中，找到并选中Session选项。在该界面右侧的窗口 中，按如下设置参数。

燄 **PuTTY Configuration**

Category\* 曰 Session

L Logging  
B Tenmnal  
I Keyboard

Bell

' Features

S Window Appearance Behaviour

\ Translation  
卍 Selecbon  
Coioufs  
日 Connection  
Data  
Proxy  
fflSSH

Serial Telnet Rlogin SUPDUP

Basic options for your PuTTY session

Speafy the destination you want to connect to

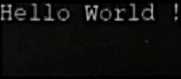
Close window on exit

O Always Q Never (•) Only on dean ext

图 11-16 PuTTY Configuration 对话框

**M'，I I** 早 屮 **u** 八

1. Connectiontype: Serial (通过夏选框选择)。
2. Serialline： C0M5 (通过文本框输入)。
3. Speed； 9600 (通过文本框输入)。
4. 单击图11. 16中的【Open】按钮，进入命令行模式。
5. 在LoongIDE主界面主菜单下,选择I)'ebug->Run,进入调试器模式°等开始运行 程序时，在PuTTY命令行界面中打印出计算机所接收到的字符巾，如图1】.17所示。



聋 COM 5- PuTTY

图**11-17 PuTTY**命令行模式下打印所接收到的字符申信息

**11.4.2**串口通信的**C**语言设计和实现

本节将介绍如何使用C语言并通过调用API编写串口通信的代码，并在龙芯1B硬件开 发平台上进行测试和验证。

1. 软件代码设计的分层结构

应用程序代码

在设计软件代码时，采用常用的分层和代码封装技术， ,

如图11. 18所示。釆用分层结构的好处是，各层之间相对独 应用程序接口(API)函敎

立，易于对代码进行维护和移植。 f

1)底层寄存器的定义和声明 读/写寄存器操作的封装

本书前面提到龙芯1B处理器采用的是MIPS架构，该 —T—

架构中存储器和外设釆用了统一编址的方法，且访问存储 ~— ~~丄~~

器和访问外设釆用相同的指令。因此，可以通过C语言中 ,层寄存器的定义和声明 的结构和联合数据类型来封装UART模块中的寄存器，如代 图**11-18**程序设计的分层结构 码淸单11-2所示。

代码清单**11-2 UART**模块中寄存器的封装

typeilef struct |

union |

struct I

unsigned char dat；

unsigned char ier；

I dal\_ier；

struct I

unsigned char divclk\_low； unsigned char divclk\_high； I divclk ；

I reg\_muxl;

union |

unsigned char iir;

unsigned char fcr；

I ；

unsigned char ler;

unsigned char mcr；

unsigned char Isr；

unsigned char msr;

I UART.REG.SET；

因为UART模块中的数据寄存器，以及中断使能寄存器和时钟分频寄存器共用一个存储 地址空间，因此使用联合数据结构进行处理，如代码清单11-3所示。

代码清单**11-3**嵌入联合数据类型封装公共存储类型**（1）**

union \

struct |

unsigned char dat;

unsigned char ier;

I datjer;

struct I

unsigned char divdkjow；

unsigned char divclk\_high；

I divclk;

I r<\*g„niuxl;

此外，UART模块中的中断标识寄存器和FIFO控制寄存器也采用共:用存储空间地址， 因此也釆用联合数据结构进行”装，如代码淸单11-3所示

代码清单**11-3**嵌入联合数据类型封裝公共存储类型**（2）**

union |

unsigvirtl char iir；

unsigned char fcr;

I;

最后通过代码,将结构体指向UART模块寄在器的首地址,如代码清单丨1-4所示。

代码清单**11-4**将结构体指向**UART**模块寄存器的首地址

■leGnr UART3\_BASE\_AD1）R 0xBEE4C000

执Wine UART\_3\_REC （（l'ART\_REG\_SET • ） L'ART3\_BASE\_ADDF｛）

2） 自定义数据类型

为了便于对数据类型的操作，在设计中对C语言原有的数据类型又进行了封装，如代 码清单11-5所示。

代码清单**11-5**对**C**语言原有数据类型的封装

typedef unsign^i char uchar；

tyjxnlef unsigned int uint;

3） 封装发送字符串操作的函数

在该设计中，对修改甲口波特率的操作和发送字符申的操作分别逬行了封装，如代码清

第11章 异步串口原理和通信的实现

单11-6和代码清单11-7所示。

代码清单11-6对发送字符串操作的封装函数

/\*发送单个字符的函数\*/

trans\_char( UART\_REG\_SET \* device, uchar c)

(device->reg\_mux 1). dat\_ier. dat = c;

//将字符写入**UART**模块的数据寄存器

/\*发送字符申的函数\*/

trans\_str( UART\_REG\_SET ♦ device, uchar \* str)

〃判断字符申是否结束，结束则退出

trans\_char( UART\_3\_REG, \* str)； str++;

〃调用发送単个字符的函数**trans\_char**

代码清单11-7对修改波特率操作的封装函数

change.band( UART\_REG\_SET ♦ device, uint bandrate)

uchar value;

uint band；

//根据给定的波特率，计算分频值

//读取**ler**寄存器的内容

//置**ler**寄存器中的**dlab**位为**T**

〃写**ler**寄存器，使能访问分頻器

〃写低**8**位分频寄存器

//写髙**8**位分頻寄存器

〃读取**ler**寄存器中的内容

〃置**ler**寄存器中的**dlab**位为**“0”** //写**ler**寄存器，禁止访问分頻器

band = (50000000/( 16 \* bandrate));

value = device->lcr；

value = value I 0x80； device->lcr = value ；

(device->reg\_muxl). divclk. divclk\_low = ( band & Oxff); (device->reg\_muxl). divclk. divclk\_high = (t>and»8); value = device->lcr； value = value & 0x7f； device->lcr-value;

注:对■串口模块中其他寄存器的初始化操作，是通过在main, c文件中调用初始化系 统函数lslx\_drv\_init()实现的,详见main, c史件。

1. C语言程序设计

本部分将介绍如何在LoongIDE中使用C语言和API函数编写申口通信程序代码，主要 步骤如下。

1. 启动LoongIDE软件工具。
2. 在 LoongIDE 主界面主菜单中，选择 New->NewProject Wizard...,弹出 New Project Wizanl-C Project 对话框。
3. 在New Project Wizard-C Project对话框中，按如下设置参数。

① Project Type： C Executable (通过复选框设置)。

384 很.型计算机系统原理及应用

■ ■ —〜

***2*** Pmjecl Nanir: c\aniplc\_l I\_2 (.通过义卜机物入) .

注：工程文件夹设置为:E 八looii”onl1l\_Eimpl(、'example」1\_2。

1. 弹出Ne Pmjeci WizaM・ MCU, TookhaiM RTOS对活框「在该对活框中，按如下 设置参tt

J Men Menial： 151B200 (15232)；

1. Too】 Chain： SDE Lite4. 9. 2for MIPS；
2. l ?ing BTOS： None ( barepn>gramming) •
3. 单击 Next 按钮.弹出 New Pmject Wizanl-Bare Program Components 对话框。
4. 在、**e** Project Wizanl-Bare Program Components r|\ 不勾选任何程序组件(Program Components)前面的复选框。
5. 单击 Next 按钮,弹出 New Project Wizard-New project summary 对话框。
6. 在Nev Project XVizanl-New project summary对活框中，单击OK按钮，完成工程的 建立。
7. 在l^oonglDE主界面左侧的Project Explorer窗口中，找到并双击main, c文件乡
8. 自动打开main.c*文件,*删除main ()函数，然后在该文件中添加如代码清单11-8 所示的C语言代码,该代码清单中只给出了主函数main中的代码。

代码清单11-8主函数main中的代码

int main(void )

|  |  |
| --- | --- |
| **int i；** |  |
| **unsigned char string：40] = | \*' Hello World!” I** | //要发送的字符串 |
| **unsigned char format]2] = j " \r\nw | ；** | 〃要发送的格式化符号 |
| **lslx\_dn\_init()；** | //初始化**1B**处理器中的外设模块 |
| **install\_3th\_librarie$()；** | //安装第三方库 |
| **change J^andC UART\_3\_REG ,9600)；** | **// UART3**模块波特率重新设置为**9600** |
| **trans\_str( E A RT\_3\_REG, format)；** | 〃发送格式化字符 |
| **tran>\_str( UART\_3\_REG,string)；** | 〃发送真正的字符串 |
| **return 0；** |  |

注：在该代码清单中只给出了主函数main中的代码，在main, c文件中新添加的其 他代码详见main.c文件。这些代码在前面的代码清单中都已经给出并进行了详细说明。

3.设计编译和调试

本部分将介绍如何对C语言代码进行编洋，并将生成的代码下载到龙芯1B开发板进行 验证，其过程与11.4. 1节中所介绍的设计编译和调试过程完全相同。

11.4.3总结

上血分别使用汇编语言和C语言编写程序代码实现申口通信。下面对这两种方法开发



应用程序的优点和缺点进行简爪的比较和说明。

1. 釆用汇编诺言开发应用，读者需要掌握处理器底层架构和指令集的知识，对读者 的计算机系统原理知识要求较髙。而采用C浴言开发应用，读者仅需要了解API函数实现 的功能，以及入口参数和返冋参数即可，不需要过多地掌握处理器底层架构和指令集的知 识,对读者进行应用开发的入门门槛要求较低。

读者会发现，在雅个设计工程中，最底层的启动引导代码都是由汇编语言实现的，这是 因为C语言是一个跨平台的厝言，有些最底层的机器细节使用C语言是无法进行描述的。

1. 在开发时间上，釆用汇编语言开发应用的时间要明显多于采用C语言开发应用的 时间。因此，对于要求快速上市的应用开发来说采用C语言是比较合适的。
2. 因为釆用C语言开发时，读者往往是调用经过层层封装的API函数，这对于对存 储器容At比较苛刻的嵌入式应用场合来说是不利的，而釆用基于汇编语言对硬件的直接操作 将显著改善对存储器资源的使用效率。
3. 在使用C语言开发应用程序代码时，通过优化C语言设计代码结构，以及合理设 置编译器的属性参数，就能让C语言所生成的代码效率過近使用汇编语言所达到的代码 效率。
4. 从调试代码的角度来说，显然采用汇编语言编写代码具冇优势，因为汇编语言直 接对应机器底层，而C语言代码经过层层封装并不容易找到设计缺陷。

总而言之，两种方法各冇优缺点。因此，在读者学习软件代码开发时，应该先学习汇编 语言程序开发，然后过渡到C语言程序开发。这样在编写C语言代码吋，就能充分高效地 利用底层硬件的各种资源。此外，在必要时可以通过在C语言中嵌入汇编代码的方法，以 满足实时性的要求。