



Version	Date	Author	Description of Change
V1.0	2012-09-05	Robert	Initial Version For Bluetooth Uart
V1.1	2014-08-21	Robert	Add RTS/CTS Description
V1.2	2014-10-09	Sean	Add Baud Rate Noting



## Bluetooth Uart 模块

### 1.1 简介

根据不同的蓝牙模组 UART 接口协议，BUART 支持 1200bsp~3,000,000bps 波特率(如果是 RC48MHz，波特率支持范围减半，可能有一定的偏差)。8/7/6/5 位数据，1/2 位停止位，奇偶校验且支持外部 FIFO 最大可达 12KB 深度,支持双工工作模式.详细的 Feature List 如下：

- 全双工传输。
- 波特率可调，96M 时钟频率下支持 1200bps~3Mbps。
- 8 倍过采样。
- 数据位宽支持 5~8bit。
- 停止位支持 1~2bit。
- 支持奇偶校验。
- 支持双向流控 RTS/CTS。
- 支持帧格式错误检查、溢出错误检查和校验位错误检查。
- 支持接收中断和发送中断以及中断屏蔽。
- 支持接收 FIFO 和发送 FIFO，可使用内部深度为 4 字节的 FIFO 或者 PMEM 中的作为外部 FIFO。



## 1.2 接口函数

```
int BuartInit(unsigned int BaudRate,unsigned char DatumBits,unsigned char
Parity,unsigned char StopBits);
int BuartExFifoInit(unsigned short FifoStartAddr,unsigned short RxFifoDepth,
    unsigned short TxFifoDepth,unsigned short RxFifoTrgDep);
int BuartIOctl(int Cmd,unsigned int Opt);
int BuartRecv(unsigned char* Buf,unsigned int BufLen,unsigned int Wait);
int BuartSend(unsigned char* Buf,unsigned int BufLen);
void GpioBuartRxIoConfig(unsigned char ModeSel);
void GpioBuartTxIoConfig(unsigned char ModeSel);
void GpioBuartCtsIoConfig(unsigned char ModeSel);
void GpioBuartRtsIoConfig(unsigned char ModeSel);
```

## 1.3 接口函数描述

### 1.3.1 BuartInit

原型:

```
int BuartInit(unsigned int BaudRate,unsigned char DatumBits,unsigned char
Parity,unsigned char StopBits);
```

参数:

BaudRate: 波特率, 96MHz 主频下, 取值[1200:3000000], 48MHz 最大 1,500,000

DatumBits: 数据位数, 取值{5,6,7,8}

Parity: 奇偶校验, 取值{0=无校验, 1=奇校验, 2=偶校验}

StopBits: 停止位, 取值{1=1bit 停止位, 2=2bit 停止位}

描述:

初始化 BUART 硬件模块

返回值:

0: 成功

EINVAL:参数不正确

### 1.3.2 BuartExFifoInit

原型:

```
int BuartExFifoInit(unsigned short FifoStartAddr,unsigned short RxFifoDepth,
    unsigned short TxFifoDepth,unsigned short
    RxFifoTrgDep);
```

参数:

FifoStartAddr: 外部 FIFO 在 PMEM 中偏移地址, 取值[0:0x3FFF]

RxFifoDepth: Rx FIFO 深度, 取值[1:0x4000]

TxFifoDepth: Tx FIFO 深度, 取值[1:0x4000]

RxFifoTrgDep: Rx 中断触发深度, 取值[1:0x4000]



注:

1. RxFifoDepth+ TxFifoDepth<=0x4000
2. RxFifoTrgDep< RxFifoDepth

描述:

当 BUART 使用外部 PMEM 作个外部 FIFO 时, 配置 UART 外部 FIFO

返回值:

- 0: 成功  
EINVAL: 参数不正确

### 1.3.3 BuartIOctl

原型:

int BuartIOctl(int Cmd,unsigned int Opt);

参数:

Cmd: I/O 控制命令(以 SDK 中 uart.h 命令为准),取值 {  
UART\_IOCTL\_BAUDRATE\_GET, //获得当前波特率  
UART\_IOCTL\_BAUDRATE\_SET, //设置波特率  
UART\_IOCTL\_RXFIFO\_TRGRDEP\_GET, //获得 RXFIFO 触发深度  
UART\_IOCTL\_RXFIFO\_TRGRDEP\_SET, //设置 RXFIFO 触发深度  
UART\_IOCTL\_INTR\_ENABLE, //使能 UART 中断  
UART\_IOCTL\_INTR\_DISABLE, //禁止 UART 中断  
//BT-UART 专用于外部 FIFO  
BUART\_IOCTL\_RXFIFO\_DEPTH\_GET, //获取外部 RXFIFO 深度  
BUART\_IOCTL\_RXFIFO\_DEPTH\_SET, //设置外部 RXFIFO 深度  
BUART\_IOCTL\_RXFIFO\_TRGR\_DEPTH\_GET, //获取外部 RXFIFO 触发深度  
BUART\_IOCTL\_RXFIFO\_TRGR\_DEPTH\_SET, //设置外部 RXFIFO 触发深度  
  
BUART\_IOCTL\_TXFIFO\_DEPTH\_GET, //获取外部 TXFIFO 深度  
BUART\_IOCTL\_TXFIFO\_DEPTH\_SET, //设置外部 TXFIFO 深度  
BUART\_IOCTL\_TXFIFO\_SETCMD, //发起 TXFIFO SET 命令  
BUART\_IOCTL\_RXFIFO\_DATLEN\_GET, //获取外部 RXFIFO 内有效数据长度  
  
BUART\_IOCTL\_BTMODE\_SET, //设置 BUART 为 BT 工作模式  
BUART\_IOCTL\_RXFIFO\_CLR, //清空外部 RXFIFO 数据  
}

Opt: 如果命令包括参数, 则为命令参数, 否则为 0

描述:

控制 BUART 各种 I/O 行为。

返回值:

- 0 或其它值: 成功  
EINVAL: 参数不正确  
ENOSYS: 无此命令



### 1.3.4 BuartRecv

原型:

```
int BuartRecv(unsigned char* Buf,unsigned int BufLen,unsigned int Wait);
```

参数:

Buf: 数据接收缓冲区

BufLen: 缓冲区长度

Wait: 等待时间, 时间单位为在某波特率下, 传输一个 BYTE 需要的时间

描述:

BUART 数据接收

返回值:

>0: 成功(可能部份接收,即 RecvData>0&&RecvData<=BufLen)

ETIME: 超时失败

### 1.3.4 BuartSend

原型:

```
int BuartSend(unsigned char* Buf,unsigned int BufLen);
```

参数:

Buf: 数据发送缓冲区

BufLen: 缓冲区长度

描述:

BUART 数据发送

返回值:

>0: 成功

### 1.3.5 RX/TX IO 复用

原型:

```
void GpioBuartRxIoConfig(unsigned char ModeSel);
```

```
void GpioBuartTxIoConfig(unsigned char ModeSel);
```

参数:

ModeSel: GPIO RX/TX 复用关系索引,RX/TX 两两可任意组合

RX 取值

{0=GPA13,1=GPA24,2=GPA8,3=GPA29,ff=disable gpio function}

TX 取值

{0=GPA16,1=GPA25,2=GPB9,3=GPB28,ff=disable gpio function}

描述:

BUART RX/TX GPIO 引脚复用设置



返回值:

无

### 1.3.6 CTS/RTS IO 复用

原型:

```
void GpioBuartRtsIoConfig (unsigned char ModeSel);
```

```
void GpioBuartCtsIoConfig (unsigned char ModeSel);
```

参数:

ModeSel:       GPIO CTS/RTS 复用关系索引, CTS/RTS 两两可任意组合  
                  RTS 取值 {0=GPA12,1=GPB31,2=GPC1, ff=disable gpio function}  
                  CTS 取值 {0=GPA11,1=GPB30,2=GPC0, ff=disable gpio function}

描述:

BUART RTS/CTS GPIO 引脚复用设置

返回值:

无

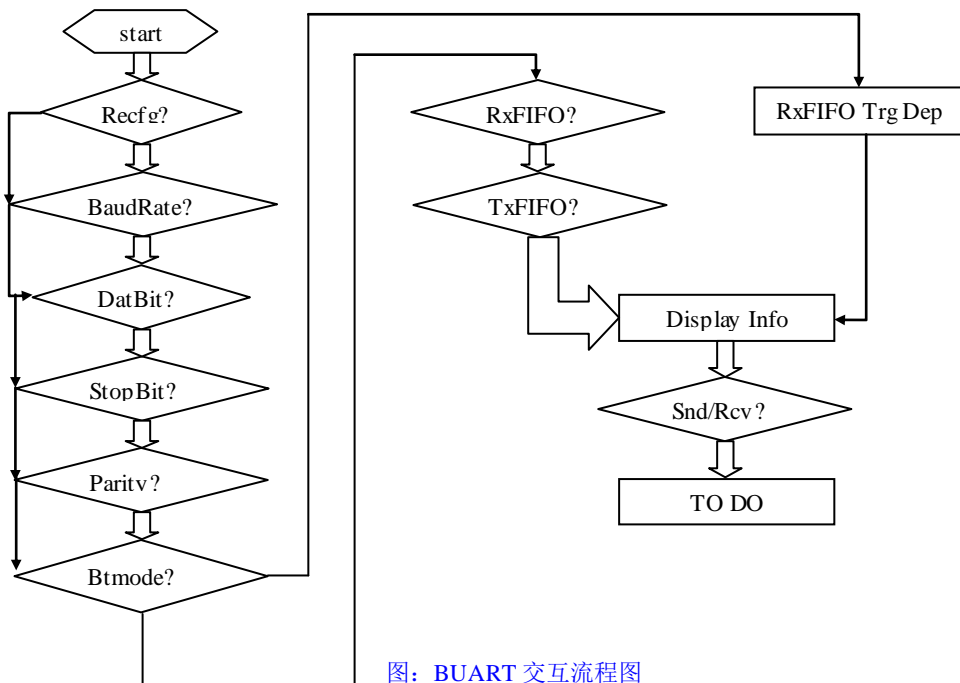
## 1.4 BUART 模块测试



### 1.4.1 测试 概述

为了测试的简单、方便、重现的目标，BUART 模式的测试条件和用例采用了交互式的动态条件模式，即测试者不用修改测试程序、重新编译程序，就可以根据串口输入测试条件，从而得到不同的测试案例。

### 1.4.2 测试 流程



当流程走到最后一步时，BUART 基本参数就配置完成了，最后将串口调试工具与 BUART 连接好后，就可以真正的测试 BUART 了，根据发送或接收流程，查看收发双方的数据是否一致来判断 BUART 是否工作正常。下面是在调试串口用以设置 BUART 串口的截图：



```
SSCO3.2 (作者:聂小猛(丁丁), 主页http://www.mcu51.co...
@@@@@@@@@@@@AutoTest Start@@@@@@@@@@@@
Please send 'Y' or 'y' to test bt uart, send 'N' or 'n' to skip
BUART configuration:
    BaudRate      :115384 bps
    Frame Len     :8 Bits
    Parity        :No
    Stop Len      :1 Bit(s)
    BTmode        :Yes
    BTRx FIFO Depth: 1
    BTRx FIFO Trigger Depth: 1
    BTTx FIFO Depth: 1
Do you want to configure Buart, Yes(Y) or No(N), please give out your choice

[BAUDRATE]
01: 1200
02: 2400
03: 4800
04: 9600 (default)
05: 19200
06: 38400
07: 57600
08: 76800
09: 115200
10: 230400
11: 460800
12: 921600
13: 1382400
14: 1500000
15: 1843200
16: 2764800
17: 2764000
18: 3000000
[01-18], CR default, [1200:3000000], XX for customization

[DATUM BIT]
1: 5 Bits
2: 6 Bits
3: 7 Bits
4: 8 Bits (default)
Select 1-4, CR default:
*** Use The Default

[STOP BITS]
1: 1 Bit (default)
2: 2 Bits
Select 1-2, CR default:
*** Use The Default

[PARITY]
1: NONE (default)
2: ODD
3: EVEN
Select 1-3, CR default:
*** Use The Default

[BT MODE]
1: Use BT mode (default)
2: Not BT mode
Select 1-2, CR default:
*** Use The Default

[RX FIFO]
1: RX FIFO depth 1 byte(default)
[00001~12288]:others(RX+TX <= 12288)
RX FIFO depth, or CR default
*** Use The Default

[TX FIFO]
1: TX FIFO depth 2 byte(default)
[00002~12288]:others(RX+TX <= 12288)
TX FIFO depth, or CR default
*** Use The Default
BUART configuration:
    BaudRate      :9600 bps
    Frame Len     :8 Bits
    Parity        :No
    Stop Len      :1 Bit(s)
    BTmode        :Yes
    BTRx FIFO Depth: 6144
    BTRx FIFO Trigger Depth: 1
    BTTx FIFO Depth: 6144
Which BUART function do you want to test, S(send), R(receiv), C(cancel)
Do you want to re-configure Buart, Yes(Y) or No(N), please give out your choice
Strike any key to start transmission test
```

注: 为了方便测试, 如果采用默认值则, 直接回车即可, 否则输入选择值, 然后再回车确认。





## 1.5 BUART 波特率精度说明

不同的波特率需要设置不同的寄存器，这些操作已经被封装在了 lib 里面。不同的波特率下也有不同的理论误差。

系统时钟 fclk 为 96M，假设需要的波特率为 x，则 baud\_clk 频率应该为 8x。  
baud\_clk 的频率为 fclk 的分频时钟，分频系数为  
 $(\text{clk\_div}+1)+(\text{clk\_div\_frac}*0.125)=\text{fclk}/8x$ 。假设所需波特率为 1382400，则  
 $\text{fclk}/8x=8.68$ ，则 clk\_div 应该为 7，clk\_div\_frac 应该为 5。

典型波特率对应的参数配置以及允许的时钟频率偏差。其中，“+”表示比标准频率高(快)，“-”表示比标准频率低(慢)。

Baud rate	clk_div	clk_div_frac	Difference	Tolerance
1200	9999	0	0%	-2.5% ~ +3.4%
2400	4999	0	0%	-2.5% ~ +3.4%
4800	2499	0	0%	-2.5% ~ +3.4%
9600	1249	0	0%	-2.5% ~ +3.4%
19200	624	0	0%	-2.5% ~ +3.4%
38400	311	5	0%	-2.5% ~ +3.4%
57600	207	2	+0.05%	-2.45% ~ +2.85%
76800	155	2	0%	-2.5% ~ +3.4%
115200	103	1	+0.05%	-2.45% ~ +2.85%
230400	51	0	+0.17%	-2.33% ~ +3.57%
460800	25	0	0%	-2.5% ~ +3.4%
921600	12	0	0%	-2.5% ~ +3.4%
1382400	7	5	+0.7%	-1.8% ~ +3.5%
1843200	5	4	+0.17%	-2.33% ~ +2.97%
2764800	3	2	+2.13%	-0.37% ~ +4.93%
2764800	3	3	-0.8%	-3.3% ~ +2%