

FT62F08X

USART Application note

目录

1. USART 接口	3
1.1. USART 接口相关寄存器汇总	4
1.2. 功能描述	9
2. 应用范例	16
联系信息	20

FT62F08x USART 应用

1. USART 接口

- 同步模式
 - ✓ 产生同步时钟输出
- 多芯片通信模式
 - ✓ 哑模式唤醒之后，才可以接收数据
 - ✓ 可以通过地址匹配和 IDLE 帧唤醒哑模式
- 异步模式
 - ✓ 可编程的 7, 8, 9 比特数据模式
 - ✓ 支持 1, 2, 1.5 bit 停止位
 - ✓ 支持红外 1.0 模式
 - ✓ 单线半双工
 - ✓ 发送接收使能控制
 - ✓ 16bit 波特率设置
 - ✓ RXNE 中断，TXE 中断，IDLE 帧中断，break 帧中断，奇偶校验错误，overrun 中断，发送完成中断
- 智能卡模式
 - ✓ bit 停止位
 - ✓ 时钟输出
 - ✓ guard time
- LIN 主机模式
 - ✓ 支持断开帧的发送与检测
- 自动波特率检测

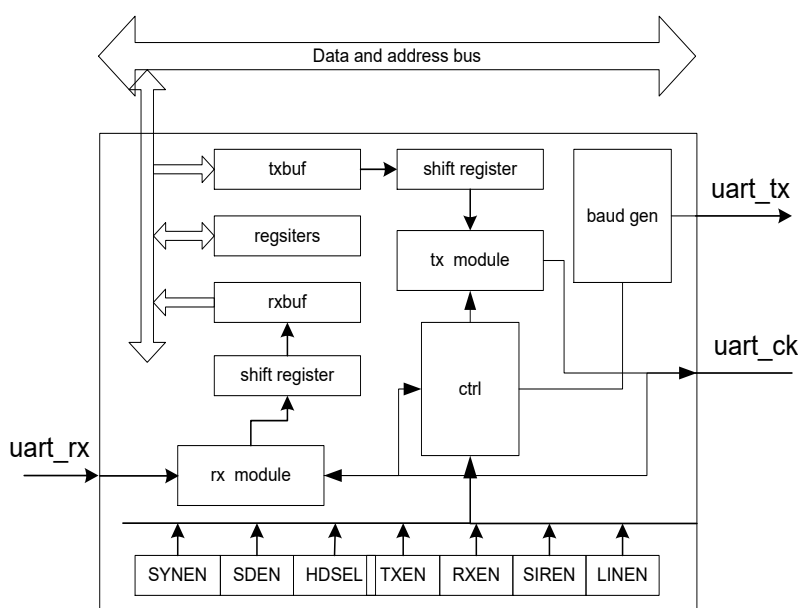


图 1-1 USART 结构框图

1.1. USART 接口相关寄存器汇总

名称	状态		寄存器	地址	复位值
DATAL	数据发送/接收 BUF 低 8 位 (不适宜位操作)		URDATAL[7:0]	0x48C	RW-0000 0000
DATAH	当 EXTEN=1: 数据发送/接收 BUF 高 1 位 注: 要先写 URxDATAL, 再写 URxDATAH 当 EXTEN=0: 1 = URxDATAL 为地址 0 = URxDATAL 为数据		URDATAH[0]	0x48D	RW-0
UARTEN	USART 模块时钟	1 = 打开 0 = 关闭	PCKEN[6]	0x9A	RW-0
SYSON	睡眠模式下, 系统时钟控制	1 = 保持运行 0 = 关闭	CKOCON[7]	0x95	RW-0
BKREQ	<u>发送断开帧</u> 1 = 使能, 或正在发送中 0 = 关闭, 或已发送完成 注意: 发送完成后该位自动清零, 不能在断开帧发送过程中对该位写 0, 发送断开帧之前请先设置断开帧的长度		URLCR[6]	0x48F	RW-0
EVEN	奇/偶校验	1 = 偶校验 0 = 奇校验	URLCR[4]		RW-0
PEN	校验位	1 = 使能 0 = 关闭	URLCR[3]		RW-0
URSTOP	<u>停止位长度</u> 1 = 1.5 bit (智能卡模式) 或 2 bit 0 = 1 bit		URLCR[2]		RW-0
LTH	<u>通信数据长度 (不包括校验位)</u> 1 = 8 bit 0 = 7 bit		URLCR[0]		RW-0
RWU	<u>多处理器模式下, 接收唤醒进入哑模式</u> 1 = 使能 0 = 关闭, 或已退出		URLCREXT[1]	0x490	RW-0
EXTEN	<u>通信数据长度</u> 1 = 9 bit 0 = 7 bit 或 8 bit (由 LTH 决定)		URLCREXT[0]		RW-0
SIRLP	红外低功耗模式	1 = 使能 0 = 关闭	URMCR[5]	0x491	RW-0

名称	状态		寄存器	地址	复位值
TXEN	串口发送 1 = 使能 (相应 IO 会被用作 TX 引脚) 0 = 关闭		URMCR[4]		RW-0
RXEN	串口接收 1 = 使能 (相应 IO 会被用作 RX 引脚) 0 = 关闭		URMCR[3]		RW-0
WAKE	哑模式唤醒方式	1 = 地址匹配 0 = IDLE 帧	URMCR[2]		RW-0
HDSEL	半双工	1 = 使能 0 = 关闭	URMCR[1]		RW-0
SIREN	红外模式	1 = 使能 0 = 关闭	URMCR[0]		RW-0
RAR	多处理器模式下的本机地址[3:0]		URRAR[3:0]	0x493	RW-0000
DLL	波特率分频计数器低 8 位和高 8 位 波特率 = $F_{master} / (16 * \{DLH, DLL\})$		URDLL[7:0]	0x494	RW-0000 0000
DLH	注: $F_{master} = Sysclk$ 。{DLH, DLL} 最小值 = 0x0001, 当其为 0x0000 时, USART 不工作;		URDLH[7:0]	0x495	RW-0000 0000
ABRE	波特率检测溢出标志	1 = 溢出 0 = 正常	URABCR[3]	0x496	RW-0
ABRM	波特率检测模式 1 = 检测长度为 [(起始位+第 1bit 数据) / 2] (数据的第 1bit 必须为 1, 第 2bit 必须为 0) 0 = 只检测起始位长度 (第 1bit 数据必须为 1)		URABCR[2]		RW-0
ABRF	检测到波特率标志位 1 = 检测到 0 = 未检测到 注: 写 0 清零, 该位清零后, 会立即再次进入波特率检测, 为了保证每次检测到的都是起始位, 建议在 RXNEF 被置位后, 再清零此位		URABCR[1]		RW-0
ABREN	自动波特率检测	1 = 使能 0 = 关闭	URABCR[0]		RW-0
LBCL	同步模式下, 发送最后 1bit 数据(MSB)对应的时钟输出 1 = 使能 0 = 关闭		URSYNCR[3]	0x497	RW-0

名称	状态		寄存器	地址	复位值
URCPHA	同步模式时钟相位 (数据采样点) 1 = 第 2 个时钟转换沿 0 = 第 1 个时钟转换沿		URSYNCR[2]	0x498	RW-0
URCPOL	同步模式时钟极性 (总线空闲时, SCK 的状态) 1 = 高电平 0 = 低电平		URSYNCR[1]		RW-0
SYNEN	同步模式 1 = 使能 (相应 IO 会被用作同步时钟输出) 0 = 关闭		URSYNCR[0]		RW-0
LINEN	LIN Master 模式	1 = 使能 0 = 关闭	URLINCR[4]	0x498	RW-0
BLTH	断开帧长度 (bit) 注: BLTH>0 有效, 建议设置为 12 或 13bit 长度, 太短会被认为接收到的为正常帧		URLINCR[3:0]		RW-0000
NACK	智能卡模式, 检测到奇偶校验出错时回复 NACK 1 = 发送 NACK 0 = 不发送 NACK		URSDCR0[6]	0x499	RW-0
CKOE	智能卡时钟源输出 1 = 使能 (需配置 PSC 寄存器为有效值) 0 = 关闭		URSDCR0[5]		RW-0
SDEN	智能卡模式	1 = 使能 (停止位必须为 1.5bit) 0 = 关闭	URSDCR0[4]		RW-0
GT	智能卡模式, 保护时间 (两字符之间的波特时钟间隔) 注意: 最小值为 1 (即使设置 GT=0, 两个字符之间也有 1 个波特周期的间隔), 保护时间过后, 发送完成标志才被置位		URSDCR1[7:0]	0x49A	RW-0000 0000
PSC	对系统时钟进行分频, 给智能卡或红外低功耗提供时钟		URSDCR2[7:0]	0x49B	RW-0000 0000
		智能卡时钟源			
	0	无效			
	1	2 分频			
	2	3 分频			
	3	4 分频			
			
	255	256 分频			

表 1-1 USART 相关寄存器

名称	状态		寄存器	地址	复位值
AFP0[7]	<u>USART_CK</u>	1 = PD1 0 = <u>PA5</u>	AFP0[7]	0x19E	RW-0
AFP2[1]	<u>USART_RX</u>	1 = PA2 0 = <u>PA7</u>	AFP2[1]	0x11D	RW-0
AFP2[0]	<u>USART_TX</u>	1 = PB6 0 = <u>PA6</u>	AFP2[0]		RW-0
UROD	<u>USART_TX 开漏输出</u>	1 = 使能 0 = <u>关闭</u>	ODCON0[0]	0x21F	RW-0

表 1-2 USART 接口引脚控制

名称	状态		寄存器	地址	复位值
GIE	全局中断	1 = 使能 (PEIE, URTE, URRXNE, TCEN, IDELE, RXSE 适用) 0 = <u>全局关闭</u> (唤醒不受影响)	INTCON[7]	Bank 首地址 +0x0B	RW-0
PEIE	外设总中断	1 = 使能 (URTE, URRXNE, TCEN, IDELE, RXSE 适用) 0 = <u>关闭</u> (无唤醒)	INTCON[6]		RW-0
URTE	发送 BUF 为空中断	1 = 使能 0 = <u>关闭</u>	URIER[1]	0x48E	RW-0
TXEF	发送 BUF 状态	1 = 空 0 = <u>非空</u> 注: 写 DATAL(8bit) / DATAH(9bit)清 零	URLSR[5]	0x492	RO-1
URRXNE	接收 BUF 非空中断	1 = 使能 0 = <u>关闭</u>	URIER[0]	0x48E	RW-0
RXNEF	接收 BUF 状态	1 = 非空 0 = <u>空, 或已被清零</u> 注: 读 DATAL(8bit) / DATAH(9bit)清 零	URLSR[0]	0x492	RO-0
TCEN	发送完成中断	1 = 使能 0 = <u>关闭</u>	URIER[5]	0x48E	RW-0

名称	状态		寄存器	地址	复位值
TCF	发送完成标志	1 = 完成 0 = <u>未完成</u> 注：写 1 清零，或写 DATAL(8bit) / DATAH(9bit)后清零	URTC[0]	0x49C	R_W1C-1
IDELE	空闲帧中断	1 = 使能 0 = <u>关闭</u>	URIER[3]	0x48E	RW-0
IDLEF ¹	检测到空闲帧标志	1 = 检测到 0 = <u>未检测到</u>	URLSR[6]	0x492	RW0-0
RXSE	接收状态中断	1 = 使能 0 = <u>关闭</u> 接收状态中断产生条件： BKF = 1 FEF = 1 PEF = 1 OVERF = 1	URIER[2]	0x48E	RW-0
BKF ¹	接收到断开帧标志	1 = 接收到 0 = <u>未接收到，或已被清零</u>	URLSR[4]	0x492	RW0-0
FEF ¹	接收帧错误标志	1 = 错误 0 = <u>正确，或已被清零</u>	URLSR[3]	0x492	RW0-0
PEF ¹	接收奇偶校验错误标志	1 = 错误 0 = <u>正确，或已被清零</u>	URLSR[2]	0x492	RW0-0
OVERF ¹	接收 BUF 溢出标志	1 = 溢出 0 = <u>正常，或已被清零</u>	URLSR[1]	0x492	RW0-0
WAKE	哑模式唤醒方式选择	1 = 地址匹配 0 = <u>IDLE 帧</u>	URMCR[2]	0x491	RW-0
ADDRF	哑模式地址匹配标志	1 = 匹配 0 = <u>未匹配</u>	URLSR[7]	0x492	RO-0

表 1-3 USART 中断使能和状态位

¹ 写 0 清零，写 1 无效。

名称	地址	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	复位值
PCKEN	0x9A	TKEN	I2CEN	UARTEN	SPIEN	TIM4EN	TIM2EN	TIM1EN	ADCEN	0000 0000
CKOCON	0x95	SYSON	CCORDY	DTYSEL		CCOSEL[2:0]			CCOEN	0010 0000
URDATAL	0x48C	DATA[7:0]								0000 0000
URDATAH	0x48D	—							DATAH	---- ---0
URIER	0x48E	—		TCEN	—	IDELE	RXSE	URTE	URRXNE	--0- 0000
URLCR	0x48F	—	BKREQ	—	EVEN	PEN	URSTOP	—	LTH	-0-0 00-0
URLCREXT	0x490	—						RWU	EXTEN	---- --00
URMCR	0x491	—		SIRLP	TXEN	RXEN	WAKE	HDSEL	SIREN	---0 0000
URLSR	0x492	ADDRF	IDLEF	TXEF	BKF	FEF	PEF	OVERF	RXNEF	0010 0000
URRAR	0x493	—				RAR[3:0]				---- 0000
URDLL	0x494	DLL[7:0]								0000 0000
URDLH	0x495	DLH[7:0]								0000 0000
URABCR	0x496	—				ABRE	ABRM	ABRF	ABREN	---- 0000
URSYNCR	0x497	—				LBCL	URCPHA	URCPOL	SYNEN	---- 0000
URLINCR	0x498	—			LINEN	BLTH[3:0]			---0 0000	
URSDCR0	0x499	—	NACK	CKOE	SDEN	—				-000 ----
URSDCR1	0x49A	GT[7:0]								0000 0000
URSDCR2	0x49B	PSC[7:0]								0000 0000
URTC	0x49C	—							TCF	---- ---1

表 1-4 USART 相关寄存器地址

1.2. 功能描述

1.2.1. 一般描述

串口模块总共有三只引脚：

1. USART_RX：用作串口数据的输入引脚
2. USART_TX：用作串口数据的输出引脚，在用作半双工模式时也用作串行数据的输入引脚
3. USART_CK：在同步模式时用作同步时钟输出，在智能卡模式时用系统作分频时钟输出

该模块支持同步模式，异步模式，半双工模式，LIN Master 模式，红外模式和智能卡模式，默认的状态是工作于异步全双工模式，在确定使用一种模式后，请确保其他模式的相关使能位已关闭。

1.2.2. 异步工作模式

异步工作模式的串口采用异步的方式进行通信，配置的步骤如下：

1. 配置 DLH/DLL 产生相应的波特率进行通信，DLH 和 DLL 共同组成 16 位的波特率分频器，通信的波特率 = $f_{master} / (16 * (DL*))$ ，其中 f_{master} 为系统时钟，16 位的波特率分频器的值最小位 1，DL*表示的是 DLL 和 DLH 的组合，设置为零时串口不工作；

2. 配置 LCR 寄存器中的 LTH 位和 LCREXT 寄存器中的 EXTEN 来设置通信的数据长度，配置 LCR 寄存器中的 STOP 位来配置停止位的长度，配置 LCR 寄存器中的 PEN 和 EVEN 来配置奇偶校验位，配置 IER 寄存器中的中断使能位来允许中断；
3. 配置 MCR 寄存器中的 TXEN 和 RXEN 来使能允许发送和接收；

异步模式通信的数据格式是先发送低位数据位，最后发送高比特位，如下图所示的 8 比特数据格式不带奇偶校验和带奇偶校验的帧格式。

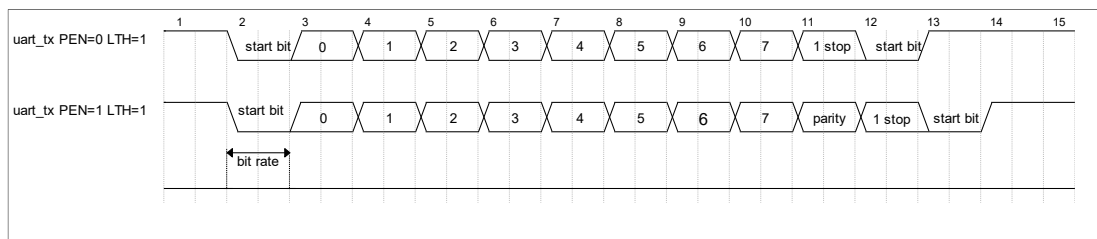


图 1-2 异步模式时序图

异步模式的数据处理流程包括阻塞模式处理和非阻塞模式处理，大致的处理流程如下：

1. 配置完波特率和相关控制位以后，发送端可以向 DATAL/H 发送 buf 寄存器写入数据，在阻塞模式下可以查询 TXEF 标志位，如果查询到 TXEF 为 1，则可以继续向 DATAL/H 写入需要发送的数据；在非阻塞模式下，使能发送为空中断，则在 TXEF 为 1 时，就会自动进入中断，向 DATAL/H 写入数据就可清除 TXEF 标志位，当在向 txbuf 写入最后一个要发送的数据时，禁用发送为空中断；
2. 接收端在阻塞模式下可以查询 RXNEF 标志位，在查询到该标志位为 1 时，表示接收到了数据，通过读取 DATAL/H 来清零 RXNEF 标志位；采用非阻塞模式接收数据时，需要使能 RXNE 中断，在串口接收到数据后，直接进入中断，读取 rxbuf 后清零 RXNEF 标志位；在采用非阻塞模式接收数据时，建议打开 RXSE 中断使能，在接收数据的过程中如遇到接收错误就会直接进入中断进行相关的处理；
3. 在串口发送数据的时候也可以使用 TCF 标志位来处理，在 TCF 标志位为 1 时，表示当前的数据发送已经完成，可以向 txbuf 写入下一个要发送的数据，这时 TCF 标志位会自动清零；

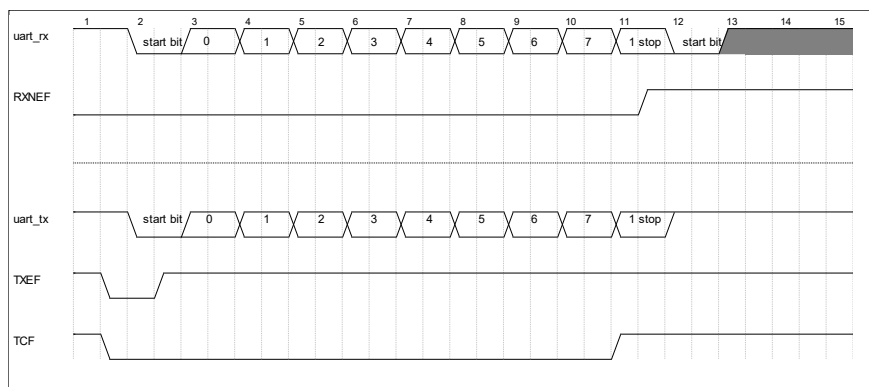


图 1-3 异步模式标志位时序图

1.2.3. 同步工作模式

同步工作模式用于串口模拟 SPI 通信的功能，在串口数据输出的同时，输出一个与数据相关的同步时钟，同步时钟的极性和相位可以通过 URSYNCR 寄存器中的 CPOL 和 CPHA 来配置；URSYNCR 寄存器中的 LBCL 控制的是最后一比特数据的时钟是否输出，该位为零时，只输出数据长度减一个有效同步时钟，

最后一个有效时钟不会输出；SYNEN 就是同步时钟输出使能位，在该位为 1 时相应的 IO 会用作同步时钟输出；该模块只能模拟 SPI 主机模式，数据输出是先发送低位数据，然后发送的高位数据，并且时钟引脚只能输出同步时钟，并不能用作时钟输入；

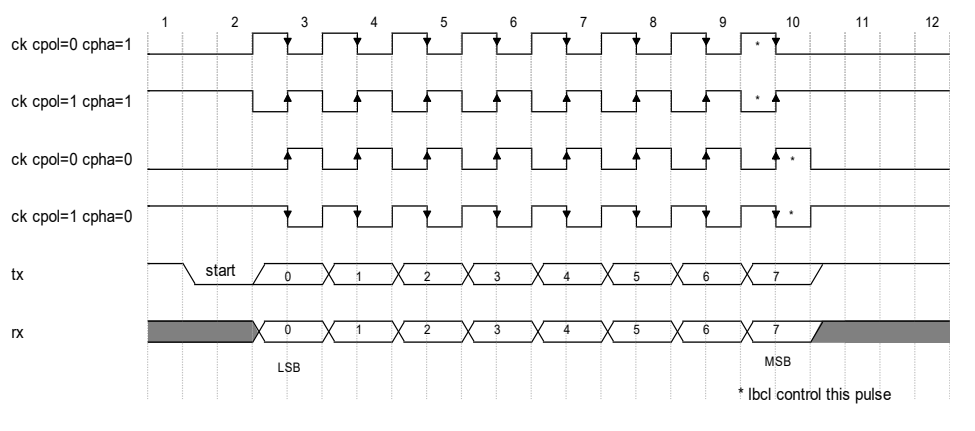


图 1-4 同步模式时序图

如图所示八比特数据格式的同步时钟输出，同步模式中如果没有使能 TXEN 也会产生同步脉冲输出，这时候同步模式只用于接收数据，写入到 DATAL/H 寄存器中的数据会发送到内部的移位寄存器中，用于产生同步脉冲输出，tx 引脚的值一直保持为 1，在同步发送的同时如果使能了 RXEN 接收位，则可以同步接收数据。

1.2.4. 半双工模式

半双工模式属于异步工作方式的一种，只是在通信时只用到了 tx 引脚，tx 引脚 IO 应该配置成开漏模式，发送与接收的处理由软件控制 RXEN 和 TXEN 来实现；需要注意的是如果在发送过程中同时使能了接收，则发送的数据也会被本机接收到；置位 HDSEL 即可启用半双工模式。

1.2.5. 红外工作模式

红外模式用于红外通信，置位 SIREN 位可以使能红外模式，同时 LTH 位置位为 1，启用八比特数据格式；通信的波特率设置跟异步串口的配置方法相同。

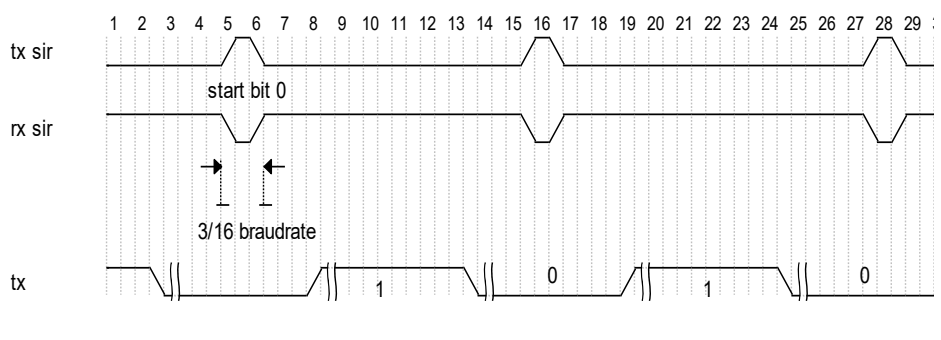


图 1-5 红外模式时序图

如图所示红外模块发送的脉冲宽度是比特周期的 3/16，当发送的数据为零时会产生一个高脉冲；接收时的低脉冲会被解释成零；接收与发送的总线极性是相反的，发送空闲时总线保持低电平，接收空闲时总线保持为高电平。

红外模式可以工作在低功耗模式，红外模式通常工作在系统的时钟频率下，红外的通信波特率

$=f_{master}/(16*DL^*)$; 当使能了 SIRLP 以后, 红外的通信波特率 $= f_{master} / (PSC * 16 * DL^*)$; 这里的 DL^* 表示 DLL 和 DLH 的组合, 在 psc 设置为 0 或 1 时, psc 分频模块无效, 波特产生模块直接使用 F_{master} , 如下图所示。



图 1-6 红外低功耗模式原理框图

1.2.6. 智能卡模式

智能卡模式属于半双工模式, 支持 ISO7816-3 标准, 置位 SDEN 来启用智能卡模式, 除此之外根据协议要求需要使能 1.5 比特停止位 STOP 和奇偶校验位 PEN, 同时需要配置相应的 IO 为开漏模式。

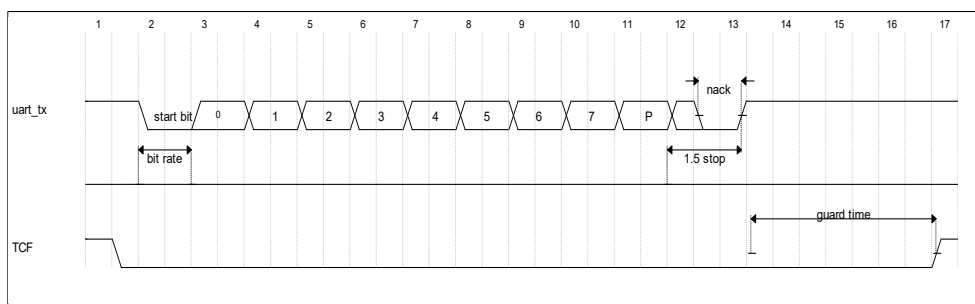


图 1-7 智能卡模式时序图

在使能了 NACK 位以后, 接收方在检测到奇偶校验出错以后, 会在 0.5 个停止位之后拉低总线一个比特周期, 同时发送方会在停止位处检测总线是否被拉低, 若检测到总线被拉低, 则会产生帧错误标志 FEF, 发送方根据要求可以选择重发当前的数据, 发送次数由用户决定。在没有使能 NACK 位时, 接收方在检测到奇偶校验错误时, 不会拉低总线而是会产生一个奇偶检验错误标志位 PEF。

智能卡模式的发送方发送完成数据后, TCF 标志位会在经过 GT 个波特周期后置位; 发送与接收的处理由软件控制 TXEN 和 RXEN 来处理。

智能卡模式中, 可以通过使能 CKOE 来输出一个时钟供给使能卡使用, 输出的时钟频率详见 URSDCR2 寄存器说明; 需要注意的是在置位 CKOE 后, 请配置 PSC 的值为有效值, 否则不应该置位 CKOE。

1.2.7. LIN Master 模式

串口模块支持 LIN Master 模式，使能 LINEN 后进入 LIN Master 模式，在发送断开帧之前请先配置一下断开帧的长度 BLTH；如图所示，在置位 BKREQ 后，tx 引脚会发送 BLTH 个连续的低电平，发送完成后自动清零，在使能该位后，可以查询 BKREQ 的状态，等到 BKREQ 为 0 时表示断开帧发送完成；在发送断开帧的过程中请勿手动清零 BKREQ。

接收端在接收到大于起始位+数据长度+停止位个数的连续低电平以后，会被认为接收到了断开帧，BKF 会被置 1。

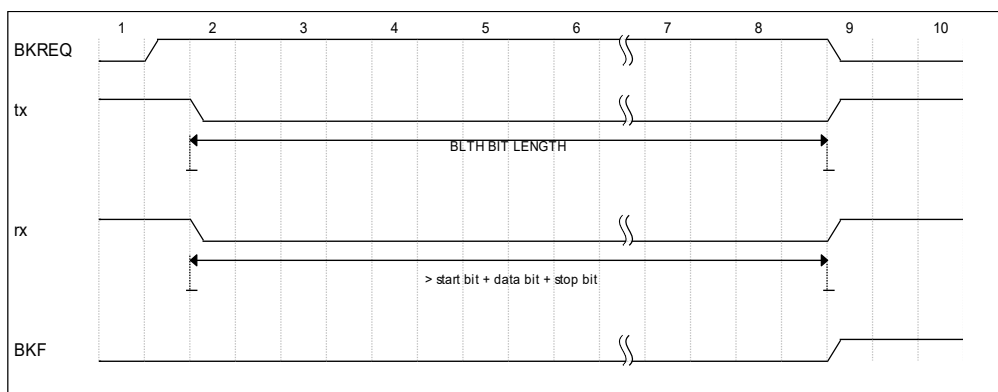


图 1-8 LIN Master 模式时序图

需要注意的是断开帧的接收与发送并不局限于 LIN mode，其他异步模式，红外模式等也是可以应用。

1.2.8. 多芯片通信模式

多处理器通信用于一个芯片用作主机模式，其他芯片用作从机模式，从机的发送引脚通过逻辑与的方式连接到主机的 RX 引脚，这种模式中主机希望接收特定的消息，只有在特定的条件触发后才会接收数据。

置位 RWU 后即可进入哑模式，屏蔽一切接收，根据 WAKE 的配置，可以唤醒接收主机接收数据：

1. WAKE 置零，在接收到起始位+数据位+停止位总个数波特周期后唤醒；
 2. WAKE 置一，在接收到匹配的地址后唤醒；
- 地址空闲唤醒，在置位 RWU 后，如果总线数据一致繁忙，则不唤醒接收，在检测到连续的一帧空闲时间 (起始位+数据位+停止位)后唤醒开始接收数据。

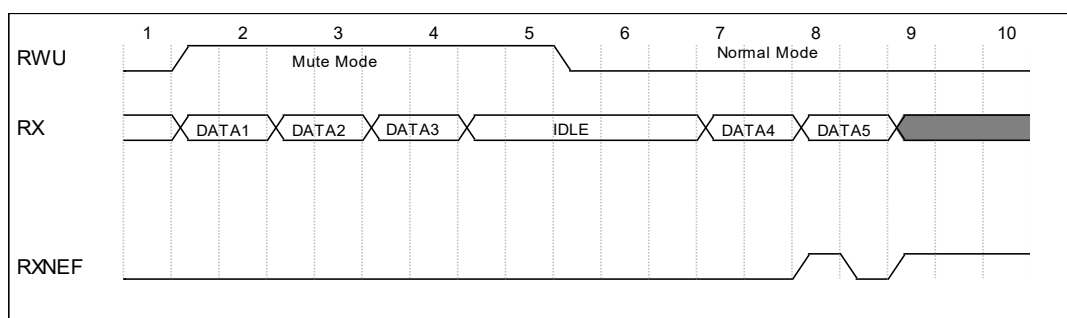


图 1-9 哑模式空闲唤醒时序图

- 地址匹配唤醒，在置位 RWU 后，每次接收到数据后都会判断数据的高位是否为 1，若为 1 则把数据的低四位与 URRAR 的值进行比较，若相等则退出哑模式开始接收之后的数据，后续如果再次接收到地址数据（该模式下数据的高位为 1 则表示接收到的数据为地址数据），则还是会与本机的地址 URRAR 进行比较，若不同立即进入哑模式；该模式下每次匹配到地址后 ADDF 会被置 1，反之没有匹配到地址时一直为零。

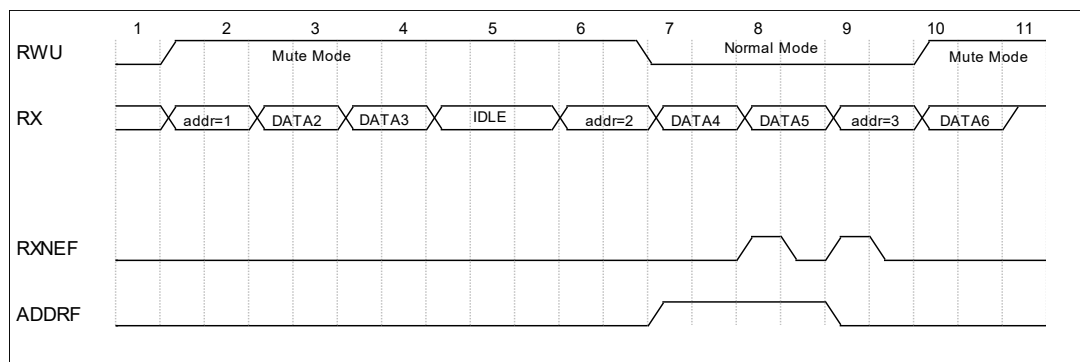


图 1-10 哑模式地址匹配唤醒时序图

1.2.9. 自动波特率检测

自动波特率检测功能用于接收端校准通信波特率，保持与发送端波特率相同，串口模块实现的波特率检测模块有两种模式：

1. 检测起始位的长度 (model0)；此模式要求数据的第一比特为一，例如数据 0x03、0x55 等；
2. 检测起始比特和第一比特的长度 (model1)；此模式要求第一比特的数据为 1，第二比特的数据为 0，例如数据 0x55，0x01 等；

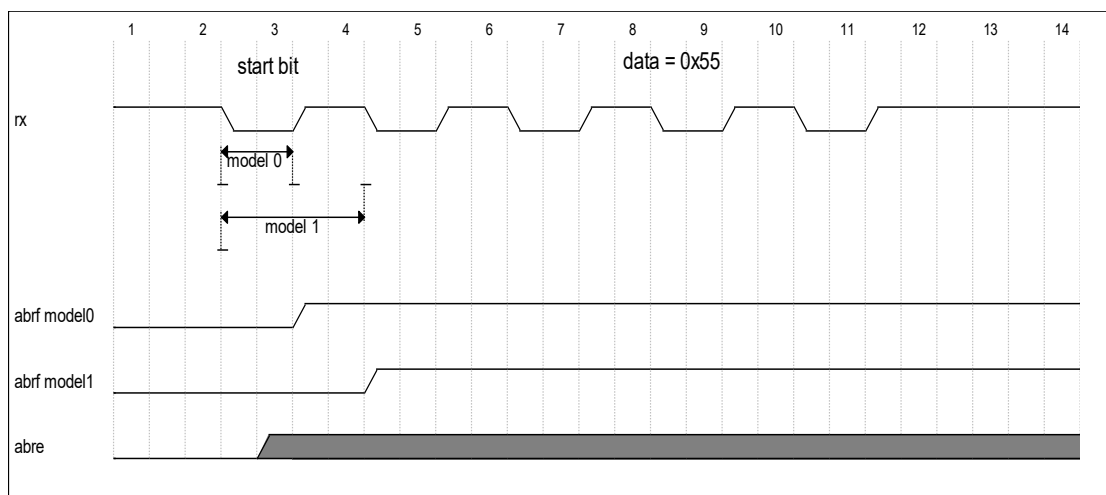


图 1-11 自动波特率检测时序图

使用波特率检测功能，首先使能 ABREN，然后根据自己使用的检测模式配置 ABRM，读取 ABRF 是否为 1 (上次使用后未清零)，如果为 1，则写零清零；然后开始接收数据，波特率检测完成后 ABRF 就会置 1，在 ABRF 置 1 后，不要立即清零 ABRF，因为清理 ABRF 会立即在当前传输的位置（可能已经不是起始比特的位置）进行波特率检测，这样会导致错误的结果；当前数据接收完成后会产生 RXNEF 标志位，然后可以清零 ABRF，开始下一次波特率的检测，假如不清零 ABRF 标志位，则下次接收数据时不

会启动波特率检测；如果波特率检测超出了范围则会产生 ABRE 标志位，表示波特率检测出错。

波特率检测完成后，如果后续还需要检测波特率则不需要立即清零 ABRF，只有在需要再次检测的时候清零 ABRF 即可。

需要注意的是波特率检测的数据是用来配置 DLL/DLH 寄存器用的，如果发送方波特率数据不靠近 $F_{baudrate} = F_{master} / (16 * \{DLH, DLL\})$ ，则波特率检测模块会自动配置本机为靠近支持的波特率，串口模块并不支持小数波特率，因此该模块的波特率检测存在误差。

2. 应用范例

```
//*****
/* 文件名: TEST_62F08x_USART.c
* 功能:    FT62F08x-USART 功能演示
* IC:      FT62F088 LQFP32
* 内部:    16M
* empno:   500
* 说明:    串口上电发送 10 个字符, 然后等待接收 10 个字节数据 (通过串口助手发送接收)
*
* 参考原理图 TEST_62F08x_sch.pdf
*/
//*****
#include "SYSCFG.h"
//*****宏定义*****
#define uchar      unsigned char

#define DemoPortOut  PB3
#define DemoPortIn   PC3

volatile uchar      receivedata[10] = 0;
volatile uchar      senddata = 0;

volatile uchar      toSend[11] = {0x31,0x32,0x33,0x34,0x35,0x36,0x37,0x38,0x0d,0x0a};
uchar i=0;
uchar mmm=0;
/*-----
* 函数名: interrupt ISR
* 功能:    中断处理
* 输入:    无
* 输出:    无
*-----*/
void interrupt ISR(void)
{
    if(URRXNE && RXNEF) //接收中断
    {
        receivedata[mmm++] = URDATA1;
        if(mmm>=10)
        {
            mmm=0;
        }
        NOP();
    }
    //-----
    if(TCEN && TCF) //发送中断
```



```

{
    TCF=1;                //写 1 清 0

    if(i<10)
    {
        URDATA1 =toSend[i++];
    }
    else
    {
        i=0;
    }
    NOP();
}
}
/*-----
* 函数名: POWER_INITIAL
* 功能:   上电系统初始化
* 输入:   无
* 输出:   无
-----*/
void POWER_INITIAL (void)
{
    OSCCON = 0B01110001;    //IRCF=111=16MHz
    INTCON = 0;             //暂禁止所有中断

    PORTA = 0B00000000;
    TRISA = 0B00000000;    //PA 输入输出 0-输出 1-输入
    PORTB = 0B00000000;
    TRISB = 0B00000000;    //PB 输入输出 0-输出 1-输入  PB3-OUT

    PORTC = 0B00000000;
    TRISC = 0B00001000;    //PC 输入输出 0-输出 1-输入  PC3-IN
    PORTD = 0B00000000;
    TRISD = 0B00000000;    //PD 输入输出 0-输出 1-输入

    WPUA = 0B00000000;    //PA 端口上拉控制 1-开上拉 0-关上拉
    WPUB = 0B00000000;    //PB 端口上拉控制 1-开上拉 0-关上拉
    WPUC = 0B00001000;    //PC 端口上拉控制 1-开上拉 0-关上拉
    WPUD = 0B00000000;    //PD 端口上拉控制 1-开上拉 0-关上拉

    WPDA = 0B00000000;    //PA 端口下拉控制 1-开下拉 0-关下拉
    WPDB = 0B00000000;    //PB 端口下拉控制 1-开下拉 0-关下拉
    WPDC = 0B00000000;    //PC 端口下拉控制 1-开下拉 0-关下拉
    WPDD = 0B00000000;    //PD 端口下拉控制 1-开下拉 0-关下拉

```

```

    PSRC0 = 0B11111111;    //PORTA,PORTB 源电流设置最大
    PSRC1 = 0B11111111;    //PORTC,PORTD 源电流设置最大

    PSINK0 = 0B11111111;    //PORTA 灌电流设置最大 0:最小, 1:最大
    PSINK1 = 0B11111111;    //PORTB 灌电流设置最大 0:最小, 1:最大
    PSINK2 = 0B11111111;    //PORTC 灌电流设置最大 0:最小, 1:最大
    PSINK3 = 0B11111111;    //PORTD 灌电流设置最大 0:最小, 1:最大

    ANSELA = 0B00000000;    //全为数字管脚
}
/*-----
* 函数名: DelayUs
* 功能:    短延时函数 --16M-2T--大概快 1%左右.
* 输入:    Time 延时时间长度 延时时长 Time Us
* 输出:    无
-----*/
void DelayUs(unsigned char Time)
{
    unsigned char a;
    for(a=0;a<Time;a++)
    {
        NOP();
    }
}
/*-----
* 函数名: DelayMs
* 功能:    短延时函数
* 输入:    Time 延时时间长度 延时时长 Time ms
* 输出:    无
-----*/
void DelayMs(unsigned char Time)
{
    unsigned char a,b;
    for(a=0;a<Time;a++)
    {
        for(b=0;b<5;b++)
        {
            DelayUs(197); //快 1%
        }
    }
}
/*-----
* 函数名: UART_INITIAL

```

* 功能： 主函数

* 输入： 无

* 输出： 无

-----*/

void UART_INITIAL(void)

{

 PCKEN |= 0B00100000; //打开 UART 时钟

 URIER = 0B00100001; //使能发送接收完成中断

 URLCR = 0B00000001; //8 位数据，停止位 1，无奇偶校验

 URMCR = 0B00011000;

 URDLL = 104; //9600 波特率 = Fosc/16*[URDLH:URDLL]

 URDLH = 0;

 TCF = 1;

 INTCON = 0B11000000;

}

/*-----

* 函数名： main

* 功能： 主函数

* 输入： 无

* 输出： 无

-----*/

void main(void)

{

 POWER_INITIAL(); //系统初始化

 UART_INITIAL();

 DelayMs(100);

 if(TXEF) //上电发送 10+1 个数据

 {

 URDATAL = 0x41;

 }

 while(1)

 {

 NOP();

 }

}

联系信息

Fremont Micro Devices Corporation

#5-8, 10/F, Changhong Building
Ke-Ji Nan 12 Road, Nanshan District,
Shenzhen, Guangdong, PRC 518057

Tel: (+86 755) 8611 7811

Fax: (+86 755) 8611 7810

Fremont Micro Devices (HK) Limited

#16, 16/F, Block B, Veristrong Industrial Centre,
34-36 Au Pui Wan Street, Fotan, Shatin, Hong Kong SAR

Tel: (+852) 2781 1186

Fax: (+852) 2781 1144

<http://www.fremontmicro.com>

* Information furnished is believed to be accurate and reliable. However, Fremont Micro Devices Corporation assumes no responsibility for the consequences of use of such information or for any infringement of patents of other rights of third parties, which may result from its use. No license is granted by implication or otherwise under any patent rights of Fremont Micro Devices Corporation. Specifications mentioned in this publication are subject to change without notice. This publication supersedes and replaces all information previously supplied. Fremont Micro Devices Corporation products are not authorized for use as critical components in life support devices or systems without express written approval of Fremont Micro Devices Corporation. The FMD logo is a registered trademark of Fremont Micro Devices Corporation. All other names are the property of their respective owners.