FT64F0AX

USART Application note

目录

FT64F0Ax 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
  - 1.5 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

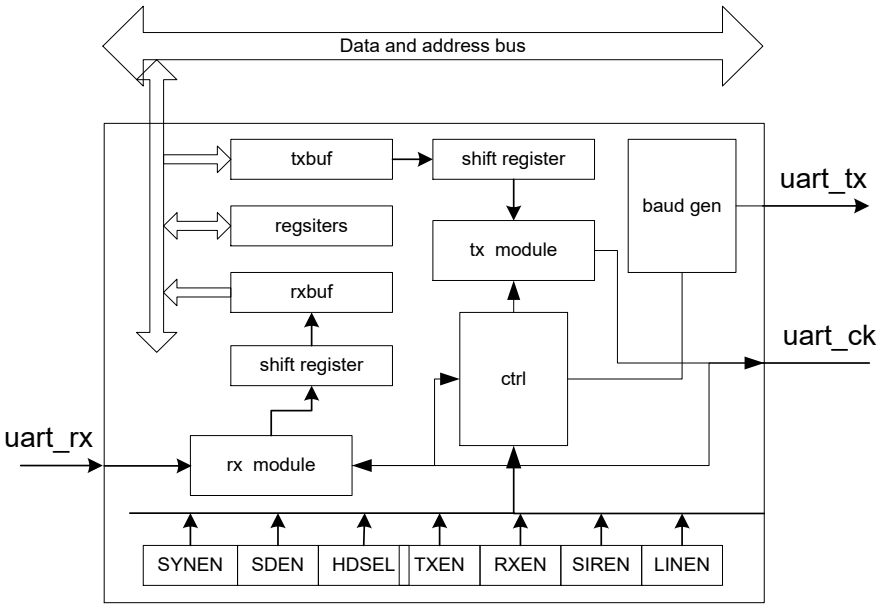

图 1-1 USART 结构框图

1.1. USART 接口相关寄存器汇总

名称	状态		寄存器	地址	复位值
UR1DATAL	数据发送/接收 BUF 低 8 位 (不适宜位操作)		UR1DATAL[7:0]	0x48C	RW-0000 0000
UR2DATAL			UR2DATAL[7:0]	0x50C	RW-0000 0000
UR1DATAH	当 URxEXTEN=1：数据发送/接收 BUF 高 1 位 注：要先写 URxDATAL，再写 URxDATAH 当 URxEXTEN=0： 1 = URxDATAL 为地址 0 = URxDATAL 为数据		UR1DATAH[0]	0x48D	RW-0
UR2DATAH			UR2DATAH[0]	0x50D	RW-0
UART2EN	USART 模块时钟	1 = 打开 0 = 关闭	PCKEN[7]	0x9A	RW-0
UART1EN			PCKEN[5]		
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
UR1BKREQ	发送断开帧 1 = 使能，或正在发送中 (完成后自动清 0) 0 = 关闭，或已发送完成 注意：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		UR1LCR[6]	0x48F	RW-0
UR2BKREQ			UR2LCR[6]	0x50F	RW-0
UR1EVEN	奇/偶校验	1 = 偶校验 0 = 奇校验	UR1LCR[4]	0x48F	RW-0
UR2EVEN			UR2LCR[4]	0x50F	RW-0
UR1PEN	校验位	1 = 使能 0 = 关闭	UR1LCR[3]	0x48F	RW-0
UR2PEN			UR2LCR[3]	0x50F	RW-0
UR1STOP	停止位长度 1 = 1.5 bit (智能卡模式) 或 2 bit 0 = 1 bit		UR1LCR[2]	0x48F	RW-0
UR2STOP			UR2LCR[2]	0x50F	RW-0
UR1LTH	通信数据长度 (不包括校验位) 1 = 8 bit 0 = 7 bit		UR1LCR[0]	0x48F	RW-0
UR2LTH			UR2LCR[0]	0x50F	RW-0
UR1RWU	多处理器模式下，接收唤醒进入哑模式 1 = 使能 0 = 关闭，或已退出		UR1LCREXT[1]	0x490	RW-0
UR2RWU			UR2LCREXT[1]	0x510	RW-0
UR1EXTEN	通信数据长度 1 = 9 bit 0 = 7 bit 或 8 bit (由 URxLTH 决定)		UR1LCREXT[0]	0x490	RW-0
UR2EXTEN			UR2LCREXT[0]	0x510	RW-0
UR1SIRLP	红外低功耗模式	1 = 使能 0 = 关闭	UR1MCR[5]	0x491	RW-0
UR2SIRLP			UR2MCR[5]	0x511	RW-0

名称	状态		寄存器	地址	复位值
UR1TXEN	串口发送 1 = 使能 (相应 IO 会被用作 TX 引脚) 0 = 关闭		UR1MCR[4]	0x491	RW-0
UR2TXEN			UR2MCR[4]	0x511	RW-0
UR1RXEN	串口接收 1 = 使能 (相应 IO 会被用作 RX 引脚) 0 = 关闭		UR1MCR[3]	0x491	RW-0
UR2RXEN			UR2MCR[3]	0x511	RW-0
UR1WAKE	哑模式唤醒方式	1 = 地址匹配 0 = IDLE 帧	UR1MCR[2]	0x491	RW-0
UR2WAKE			UR2MCR[2]	0x511	RW-0
UR1HDSEL	半双工	1 = 使能 0 = 关闭	UR1MCR[1]	0x491	RW-0
UR2HDSEL			UR2MCR[1]	0x511	RW-0
UR1SIREN	红外模式	1 = 使能 0 = 关闭	UR1MCR[0]	0x491	RW-0
UR2SIREN			UR2MCR[0]	0x511	RW-0
UR1RAR	多处理器模式下的本机地址[3:0]		UR1RAR[3:0]	0x493	RW-0000
UR2RAR			UR2RAR[3:0]	0x513	RW-0000
UR1DLL	波特率分频计数器低 8 位和高 8 位 波特率 = Fmaster / (16 * {URxDLH, URxDLL}) 注：Fmaster = Sysclk。{URxDLH, URxDLL}最小值= 0x0001，当其为 0x0000 时, USARTx 不工作；		UR1DLL[7:0]	0x494	RW-0000 0000
UR1DLH			UR1DLH[7:0]	0x495	RW-0000 0000
UR2DLL			UR2DLL[7:0]	0x514	RW-0000 0000
UR2DLH			UR2DLH[7:0]	0x515	RW-0000 0000
UR1ABRE	波特率检测溢出标志	1 = 溢出 0 = 正常	UR1ABCR[3]	0x496	RW-0
UR2ABRE			UR2ABCR[3]	0x516	RW-0
UR1ABRM	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)		UR1ABCR[2]	0x496	RW-0
UR2ABRM			UR2ABCR[2]	0x516	RW-0
UR1ABRF	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 URxRXNEF 被置位后，再清零此位		UR1ABCR[1]	0x496	RW-0
UR2ABRF			UR2ABCR[1]	0x516	RW-0
UR1ABREN	自动波特率检测	1 = 使能 0 = 关闭	UR1ABCR[0]	0x496	RW-0
UR2ABREN			UR2ABCR[0]	0x516	RW-0
UR1LBCL	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		UR1SYNCR[3]	0x497	RW-0
UR2LBCL			UR2SYNCR[3]	0x517	RW-0

名称	状态	寄存器	地址	复位值
UR1CPHA	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	UR1SYNCR[2]	0x497	RW−0
UR2CPHA		UR2SYNCR[2]	0x517	RW−0
UR1CPOL	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	UR1SYNCR[1]	0x497	RW−0
UR2CPOL		UR2SYNCR[1]	0x517	RW−0
UR1SYNEN	同步模式 1 = 使能 (相应 IO 会被用作同步时钟输出) 0 = 关闭	UR1SYNCR[0]	0x497	RW−0
UR2SYNEN		UR2SYNCR[0]	0x517	RW−0
UR1LINEN	LIN Master 模式：1 = 使能；0 = 关闭	UR1LINCR[4]	0x498	RW−0
UR2LINEN		UR2LINCR[4]	0x518	RW−0
UR1BLTH	断开帧长度 (bit) 注: URxBLTH>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	UR1LINCR[3:0]	0x498	RW−0000
UR2BLTH		UR2LINCR[3:0]	0x518	RW−0000
UR1NACK	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	UR1SDCR0[6]	0x499	RW−0
UR2NACK		UR2SDCR0[6]	0x519	RW−0
UR1CKOE	智能卡时钟源输出 1 = 使能 (需配置 URxPSC 寄存器为有效值) 0 = 关闭	UR1SDCR0[5]	0x499	RW−0
UR2CKOE		UR2SDCR0[5]	0x519	RW−0
UR1SDEN	智能卡模式 1 = 使能 (停止位必须为 1.5bit) 0 = 关闭	UR1SDCR0[4]	0x499	RW−0
UR2SDEN		UR2SDCR0[4]	0x519	RW−0
URxGT	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 URxGT=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	URxSDCR1[7:0] x = 1, 2	0x49A 0x51A	RW−0000 0000
URxPSC	对系统时钟进行分频，给智能卡或红外低功耗提供时钟	URxSDCR2[7:0] x = 1, 2	0x49B 0x51B	RW−0000 0000

	智能卡时钟源	红外低功耗时钟源
0	无效	无效
1	2 分频	1 分频
2	3 分频	2 分频
3	4 分频	3 分频
…	…	…
255	256 分频	255 分频

表 1-1 USART1/USART2 相关寄存器

名称	状态		寄存器	地址	复位值
GIE	全局中断	1 = 使能 (PEIE, URxTE, URxRXNE, URxTCEN, URxIDELE, URxRXSE 适用) 0 = 全局关闭 (唤醒不受影响)	INTCON[7]	Bank 首地址 +0x0B	RW-0
PEIE	外设总中断	1 = 使能 (URxTE, URxRXNE, URxTCEN, URxIDELE, URxRXSE 适用) 0 = 关闭 (无唤醒)	INTCON[6]		RW-0
UR1TE	发送 BUF 为空中断	1 = 使能 0 = 关闭	UR1IER[1]	0x48E	RW-0
UR2TE			UR2IER[1]	0x50E	RW-0
UR1TXEF	发送 BUF 状态	1 = 空 0 = 非空 注：写 URxDATAL(8bit) / URxDATAH(9bit)清零	UR1LSR[5]	0x492	RO-1
UR2TXEF			UR2LSR[5]	0x512	RO-1
UR1RXNE	接收 BUF 非空中断	1 = 使能 0 = 关闭	UR1IER[0]	0x48E	RW-0
UR2RXNE			UR2IER[0]	0x50E	RW-0
UR1RXNEF	接收 BUF 状态	1 = 非空 0 = 空，或已被清零 注：读 URxDATAL(8bit) / URxDATAH(9bit)清零	UR1LSR[0]	0x492	RO-0
UR2RXNEF			UR2LSR[0]	0x512	RO-0
UR1TCEN	发送完成中断	1 = 使能 0 = 关闭	UR1IER[5]	0x48E	RW-0
UR2TCEN			UR2IER[5]	0x50E	RW-0
UR1TCF	发送完成标志	1 = 完成 0 = 未完成 注：写 1 清零，或写 URxDATAL(8bit) /URxDATAH(9bit)后清零	UR1TC[0]	0x49C	R_W1C-1
UR2TCF			UR2TC[0]	0x51C	R_W1C-1
UR1IDELE	空闲帧中断	1 = 使能 0 = 关闭	UR1IER[3]	0x48E	RW-0
UR2IDELE			UR2IER[3]	0x50E	RW-0
UR1IDLEF [1]	检测到空闲帧标志	1 = 检测到 0 = 未检测到	UR1LSR[6]	0x492	RW0-0
UR2IDLEF [1]			UR2LSR[6]	0x512	RW0-0

[1] 写 0 清零，写 1 无效。

名称	状态		寄存器	地址	复位值
UR1RXSE	接收状态中断	1 = 使能 0 = 关闭 接收状态中断产生条件： URxBKF = 1 URxFEF = 1 URxPEF = 1 URxOVERF = 1	UR1IER[2]	0x48E	RW-0
UR2RXSE			UR2IER[2]	0x50E	RW-0
UR1BKF [1]	接收到断开帧标志	1 = 接收到 0 = 未接收到，或已被清零	UR1LSR[4]	0x492	RW0-0
UR2BKF [1]			UR2LSR[4]	0x512	RW0-0
UR1FEF [1]	接收帧错误标志	1 = 错误 0 = 正确，或已被清零	UR1LSR[3]	0x492	RW0-0
UR2FEF [1]			UR2LSR[3]	0x512	RW0-0
URxPEF [1]	接收奇偶校验错误标志	1 = 错误 0 = 正确，或已被清零	UR1LSR[2]	0x492	RW0-0
URxPEF [1]			UR2LSR[2]	0x512	RW0-0
UR1OVERF [1]	接收 BUF 溢出标志	1 = 溢出 0 = 正常，或已被清零	UR1LSR[1]	0x492	RW0-0
UR2OVERF [1]			UR2LSR[1]	0x512	RW0-0
UR1WAKE	哑模式唤醒方式选择	1 = 地址匹配 0 = IDLE 帧	UR1MCR[2]	0x491	RW-0
UR2WAKE			UR2MCR[2]	0x511	RW-0
UR1ADDRF	哑模式地址匹配标志	1 = 匹配 0 = 未匹配	UR1LSR[7]	0x492	RO-0
UR2ADDRF			UR2LSR[7]	0x512	RO-0

表 1-2 USART1/USART2 中断使能和状态位

名称	状态		寄存器	地址	复位值
AFP0[7:6] [2]	USART1_TX	00 = PA6　10 = PA7 01 = PB6　11 = PA2	AFP0[7:6]	0x19E	RW-00
AFP1[7:6] [2]	USART1_RX	00 = PA7　10 = PA6 01 = PA2　11 = PB6	AFP1[7:6]	0x19F	RW-00
AFP2[2:0] [2]	USART2_TX	000 = PB4　011 = PA4 001 = PA3　1xx = PA7 010 = PB5	AFP2[2:0]	0x198	RW-000
AFP2[5:3] [2]	USART2_RX	000 = PB5　011 = PA3 001 = PA4　1xx = PA7 010 = PB4	AFP2[5:3]		
UR2OD [2]	USART2_TX 开漏输出	1 = 使能 0 = 关闭	ODCON0[3]	0x21F	RW-0
UR1OD [2]	USART1_TX 开漏输出		ODCON0[0]		

表 1-3 USART1/USART2 接口引脚控制

[2] USART 的 TX 与 RX 引脚可以互换，且 TX 映射管脚才能配置开漏输出

名称	地址	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	复位值
PCKEN	0x9A	UART2EN	I2CEN	UART1EN	SPIEN	TIM4EN	TIM2EN	TIM1EN	ADCEN	0000 0000
CKOCON	0x95	SYSON	CCORDY	DTYSEL		CCOSEL[2:0]			CCOEN	0010 0000
UR1DATAL	0x48C	UR1DATAL[7:0]								0000 0000
UR1DATAH	0x48D	—							UR1DATAH	---- ---0
UR1IER	0x48E	—		UR1TCEN	—	UR1IDELE	UR1RXSE	UR1TE	UR1RXNE	--0- 0000
UR1LCR	0x48F	—	UR1BKREQ	—	UR1EVEN	UR1PEN	UR1STOP	—	UR1LTH	-0-0 00-0
UR1LCREXT	0x490	—						UR1RWU	UR1EXTEN	---- --00
UR1MCR	0x491	—	—	UR1SIRLP	UR1TXEN	UR1RXEN	UR1WAKE	UR1HDSEL	UR1SIREN	--00 0000
UR1LSR	0x492	UR1ADDRF	UR1IDLEF	UR1TXEF	UR1BKF	UR1FEF	UR1PEF	UR1OVERF	UR1RXNEF	0000 0000
UR1RAR	0x493	—				UR1RAR[3:0]				---- 0000
UR1DLL	0x494	UR1DLL[7:0]								0000 0000
UR1DLH	0x495	UR1DLH[7:0]								0000 0000
UR1ABCR	0x496	—				UR1ABRE	UR1ABRM	UR1ABRF	UR1ABREN	---- 0000
UR1SYNCR	0x497	—				UR1LBCL	UR1CPHA	UR1CPOL	UR1SYNEN	---- 0000
UR1LINCR	0x498	—			UR1LINEN	UR1BLTH[3:0]				---0 0000
UR1SDCR0	0x499	—	UR1NACK	UR1CKOE	UR1SDEN	—				-000 ----
UR1SDCR1	0x49A	UR1GT[7:0]								0000 0000
UR1SDCR2	0x49B	UR1PSC[7:0]								0000 0000
UR1TC	0x49C	—							UR1TCF	---- ---1
UR2DATAL	0x50C	UR2DATAL[7:0]								0000 0000
UR2DATAH	0x50D	—							UR2DATAH	---- ---0
UR2IER	0x50E	—		UR2TCEN	—	UR2IDELE	UR2RXSE	UR2URTE	UR2RXNE	--0- 0000
UR2LCR	0x50F	—	UR2BKREQ	—	UR2EVEN	UR2PEN	UR2STOP	—	UR2LTH	-0-0 00-0
UR2LCREXT	0x510	—						UR2RWU	UR2EXTEN	---- --00
UR2MCR	0x511	—		UR2SIRLP	UR2TXEN	UR2RXEN	UR2WAKE	UR2HDSEL	UR2SIREN	-000 0000
UR2LSR	0x512	UR2ADDRF	UR2IDLEF	UR2TXEF	UR2BKF	UR2FEF	UR2PEF	UR2OVERF	UR2RXNEF	0000 0000
UR2RAR	0x513	—				UR2RAR[3:0]				---- 0000
UR2DLL	0x514	UR2DLL[7:0]								0000 0000
UR2DLH	0x515	UR2DLH[7:0]								0000 0000
UR2ABCR	0x516	—				UR2ABRE	UR2ABRM	UR2ABRF	UR2ABREN	---- 0000
UR2SYNCR	0x517	—				UR2LBCL	UR2CPHA	UR2CPOL	UR2SYNEN	---- 0000
UR2LINCR	0x518	—			UR2LINEN	UR2BLTH[3:0]				---0 0000
UR2SDCR0	0x519	—	UR2NACK	UR2CKOE	UR2SDEN	—				-000 ----
UR2SDCR1	0x51A	UR2GT[7:0]								0000 0000
UR2SDCR2	0x51B	UR2PSC[7:0]								0000 0000
UR2TC	0x51C	—							UR2TCF	---- ---1

表 1-4 USART1/USART2 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 URxDLH/URxDLL 产生相应的波特率进行通信，URxDLH 和 URxDLL 共同组成 16 位的波特率分频器，通信的波特率 = fmaster / (16 * (DL*))，其中 fmaster 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 URxDLL 和 URxDLH 的组合，设置为零时串口不工作；
2. 配置 LCR 寄存器中的 URxLTH 位和 LCREXT 寄存器中的 URxEXTEN 来设置通信的数据长度，配置 LCR 寄存器中的 URxSTOP 位来配置停止位的长度，配置 LCR 寄存器中的 URxPEN 和 URxEVEN 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 URxTXEN 和 URxRXEN 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。

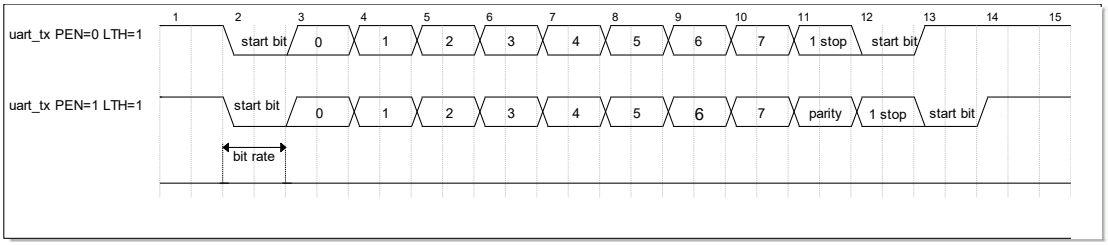

图 1-2 异步模式时序图

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 URxDATAL/H 发送 buf 寄存器写入数据，在阻塞模式下可以查询 URxTXEF 标志位，如果查询到 URxTXEF 为 1，则可以继续向 URxDATAL/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 URxTXEF 为 1 时，就会自动进入中断，向 URxDATAL/H 写入数据就可清除 URxTXEF 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 URxRXNEF 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 URxDATAL/H 来清零 URxRXNEF 标志位；采用非阻塞模式接收数据时，需要使能 URxRXNE 中断，在串口接收到数据后，直接进入中断，读取 rxbuf 后清零 URxRXNEF 标志位；在采用非阻塞模式接收数据时，建议打开 URxRXSE 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 URxTCF 标志位来处理，在 URxTCF 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 URxTCF 标志位会自动清零；

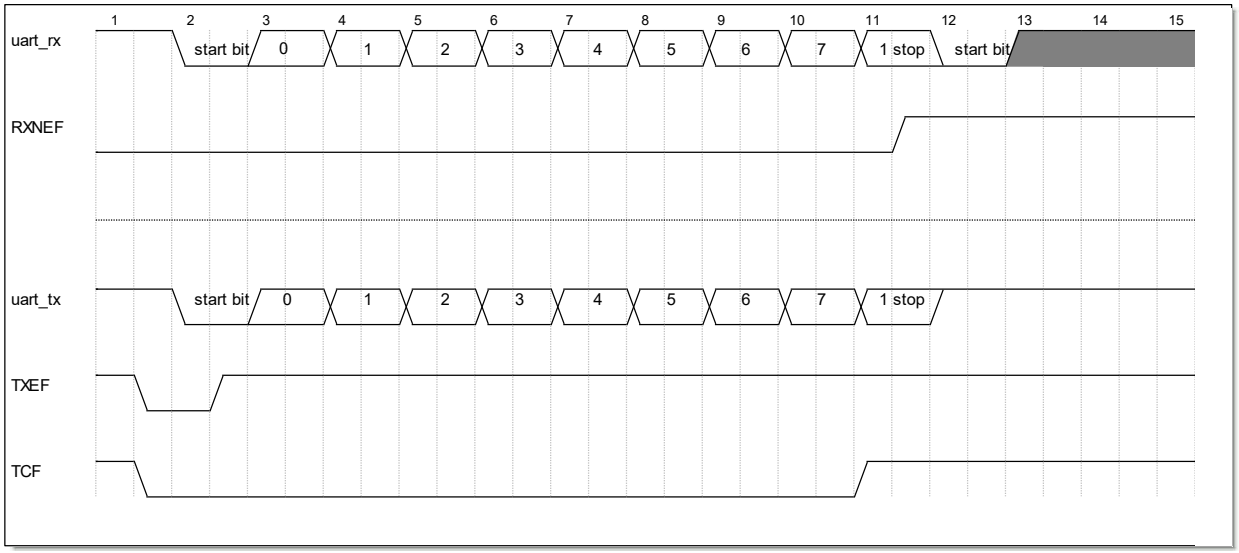

图 1-3 异步模式标志位时序图

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 URxSYNCR 寄存器中的 URxCPOL 和 URxCPHA 来配置；URxSYNCR 寄存器中的 URxLBCL 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；URxSYNEN 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；

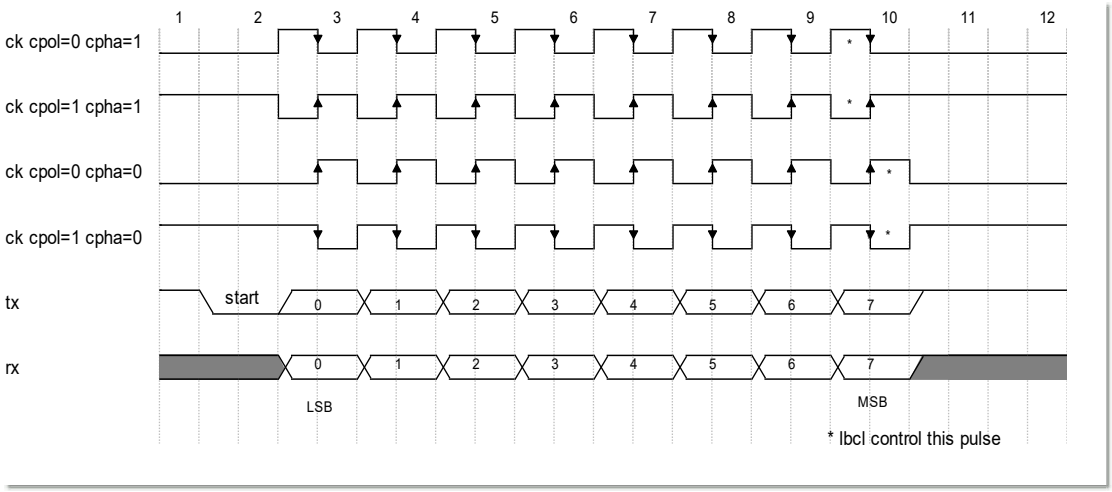

图 1-4 同步模式时序图

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 URxTXEN 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 URxDATAL/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 URxRXEN 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 URxRXEN 和 URxTXEN 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 URxHDSEL 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 URxSIREN 位可以使能红外模式，同时 URxLTH 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。

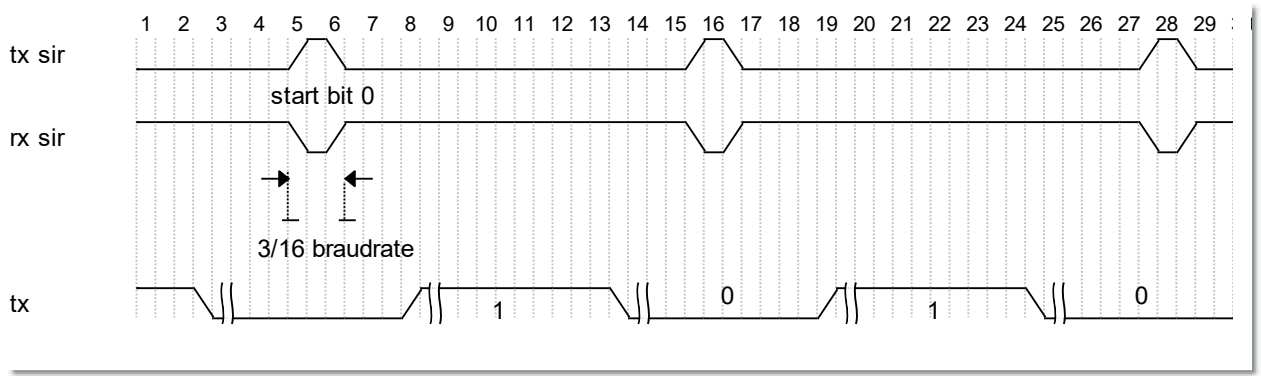

图 1-5 红外模式时序图

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 =fmaster/(16*DL*)；当使能了 URxSIRLP 以后，红外的通信波特率 = fmaster / (URxPSC * 16 * DL*)；这里的 DL*表示 URxDLL 和 URxDLH 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 Fmaster，如下图所示。

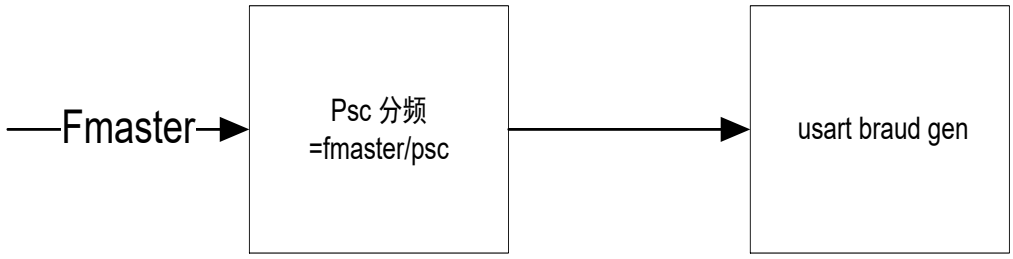

图 1-6 红外低功耗模式原理框图

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 URxSDEN 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 URxSTOP 和奇偶校验位 URxPEN，同时需要配置相应的 IO 为开漏模式。

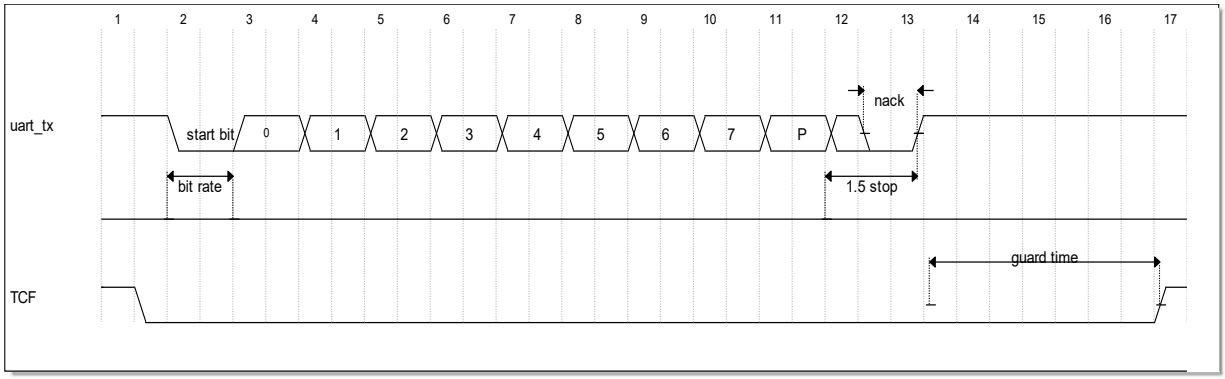

图 1-7 智能卡模式时序图

在使能了 URxNACK 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 URxFEF，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 URxNACK 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 URxPEF。

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

智能卡模式中，可以通过使能 URxCKOE 来输出一个时钟供给使能卡使用，输出的时钟频率详见 URxSDCR2 寄存器说明；需要注意的是在置位 URxCKOE 后，请配置 URxPSC 的值为有效值，否则不应该置位 URxCKOE。

1.2.7. LIN Master 模式

串口模块支持 LIN Master 模式，使能 URxLINEN 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 URxBLTH；如图所示，在置位 URxBKREQ 后，tx 引脚会发送 URxBLTH 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 URxBKREQ 的状态，等到 URxBKREQ 为 0 时则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 URxBKREQ。

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，URxBKF 会被置 1。

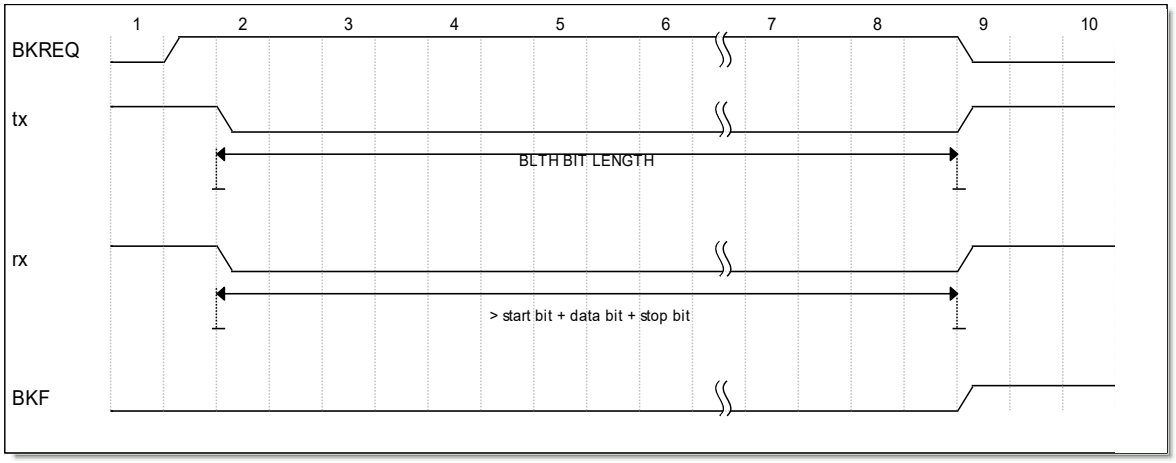

图 1-8 LIN Master 模式时序图

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

1. URxWAKE 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. URxWAKE 置一，在接收到匹配的地址后唤醒；

- 地址空闲唤醒，在置位 URxRWU 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。

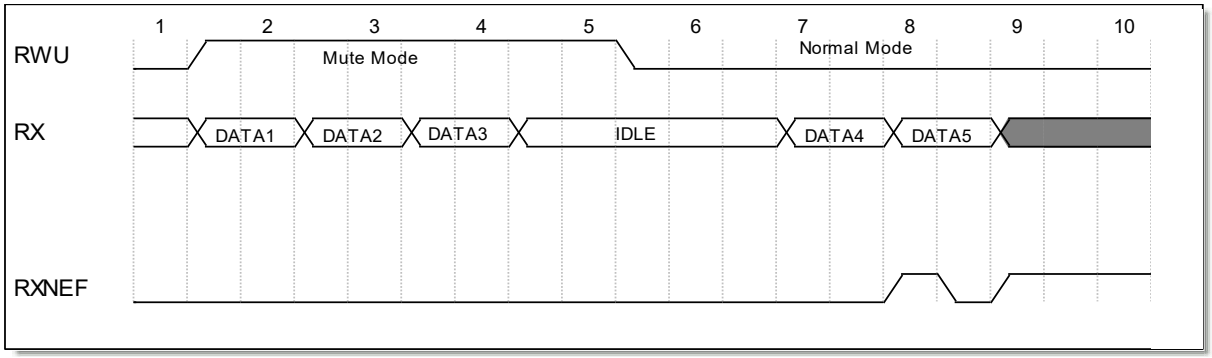

图 1-9 哑模式空闲唤醒时序图

- 地址匹配唤醒，在置位 URxRWU 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 URxRAR 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 URxRAR 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 URxADDRF 会被置 1，反之没有匹配到地址时一直为零。

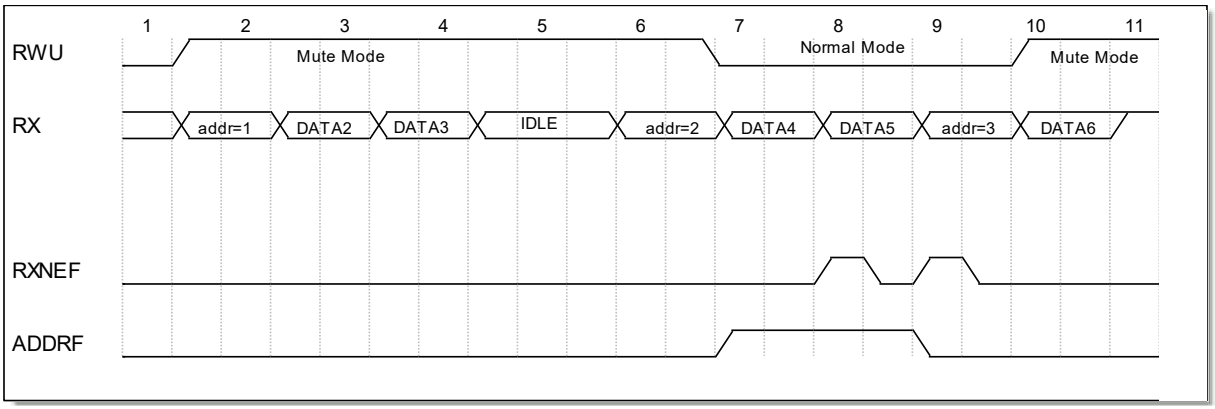

图 1-10 哑模式地址匹配唤醒时序图

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；

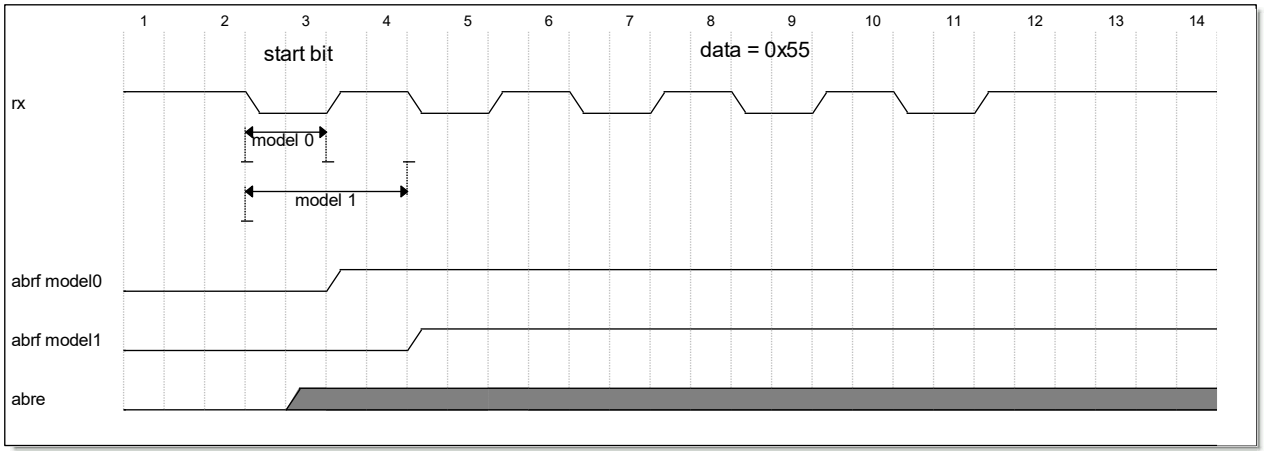

图 1-11 动波特率检测时序图

使用波特率检测功能，首先使能 URxABREN，然后根据自己使用的检测模式配置 URxABRM，读取 URxABRF 是否为 1 (上次使用过后未清零)，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 URxABRF 就会置 1，在 URxABRF 置 1 后，不要立即清零 URxABRF，因为清理 URxABRF 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 URxRXNEF 标志位，然后可以清零 URxABRF，开始下一次波特率的检测，假如不清零 URxABRF 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 URxABRE 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 URxABRF，只有在需要再次检测的时候清零 URxABRF 即可。

需要注意的是波特率检测的数据是用来配置 URxDLL/URxDLH 寄存器用的，如果发送方波特率数据不靠近 Fbraudrate = Fmaster / (16 * {URxDLH, URxDLL})，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_64F0Ax_USART.c
* 功能：    FT64F0Ax_USART 功能演示
*  IC：     FT64F0A5 TSSOP20
* 内部：    16M
* 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）
*
*              FT64F0A5   TSSOP20
*               --------------------------
* NC----------|1(PA5)       (PA4)20|----------NC
* TXIO--------|2(PA6)       (PA3)19|----------NC
* RXIO--------|3(PA7)       (PA2)18|----------NC
* NC----------|4(PC0)       (PA1)17|----------NC
* NC----------|5(PC1)       (PA0)16|----------NC
* NC----------|6(PB7)       (PB0)15|----------NC
* GND---------|7(GND)       (PB1)14|----------NC
* NC----------|8(PB6)       (PB2)13|----------NC
* VDD---------|9(VDD)       (PB3)12|----------NC
* NC----------|10(PB5)      (PB4)11|----------NC
*               --------------------------
*/
//*************************************************************
#include "SYSCFG.h";
#include      "FT64F0AX.h";
//**********************宏定义*****************************
#define  uchar    unsigned char

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：   接收中断和发送中断
 * 输入：   无
 * 输出：   无
 -------------------------------------------*/
void interrupt ISR(void)
{
    //中断处理程序
    if(UR1RXNE&&UR1RXNEF)                    //接收中断
```

```
    {
        receivedata[mmm++] =UR1DATAL;

        if(mmm>=10)
        {
            mmm=0;
        }
        NOP();
    }

    if(UR1TCEN&&UR1TCF)                    //发送中断
    {
        UR1TCF=1;

        if(i<10)
        {
            UR1DATAL =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;          //系统时钟选择为内部振荡器 16MHz,分频比为 1:1

    INTCON=0;                              //禁止所有中断

    PORTA=0B00000000;
    PORTB=0B00000000;
    PORTC=0B00000000;

    WPUA=0B00000000;                       //弱上拉的开关，0-关，1-开
    WPUB=0B00000000;
    WPUC=0B00000000;
```

```
    WPDA=0B00000000;                    //弱下拉的开关，0-关，1-开
    WPDB=0B00000000;
    WPDC=0B00000000;

    TRISA=0B00000000;                   //输入输出设置，0-输出，1-输入
    TRISB=0B00000000;
    TRISC=0B00000000;

    PSRC0=0B11111111;                   //源电流设置最大
    PSRC1=0B11111111;
    PSRC2=0B00001111;
    PSINK0=0B11111111;                  //灌电流设置最大
    PSINK1=0B11111111;
    PSINK2=0B00000011;
    ANSELA=0B00000000;                  //设置对应的 IO 为数字 IO
}
/*---------------------------------------------
* 函数名：DelayUs
* 功能：   短延时函数
* 输入：   Time 延时时间长度  延时时长 Time μs
* 输出：   无
---------------------------------------------*/
void DelayUs(uchar Time)
{
    uchar 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
* 输入：   无
---------------------------------------------*/
void DelayMs(uchar Time)
{
    uchar a,b;
    for(a=0;a<Time;a++)
    {
        for(b=0;b<5;b++)
        {
            DelayUs(197);
```

```
            }
        }
    }
    /*---------------------------------------------
    * 函数名：UART_INITIAL
    * 功能：   初始化串口
    * 输入：   无
    * 输出：   无
    ---------------------------------------------*/
    void UART_INITIAL(void)
    {
        PCKEN|=0B00100000;                  //使能 UART1 模块时钟
        UR1IER=0B00100001;                  //使能发送完成中断，使能接收数据中断
        UR1LCR=0B00000001;                  //8 位数据长度，1 位停止位，无奇偶校验位
        UR1MCR=0B00011000;                  //使能发送和接收接口

        UR1DLL=104;                         //波特率=Fmaster/(16*{URxDLH,URxDLL})=9600
        UR1DLH=0;
        UR1TCF=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(UR1TXEF)                             //上电发送 10+1 个数据
    {
        UR1DATAL=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