FM1702SL 通用读卡机芯片

1 概述

FM1702SL 是复旦微电子股份有限公司设计的基于 ISO14443 标准的非接触卡读卡机专用芯片，采用 0.6 微米 CMOS EEPROM 工艺，支持 13.56MHz 频率下的 typeA 非接触通信协议，支持多种加密算法，兼容 Philips 的 MF RC530(SPI 接口)读卡机芯片

产品特点：

- 高集成度的模拟电路，只需最少量的外围线路
- 操作距离可达 10cm
- 支持 ISO14443 typeA 协议
- 内部带有加密单元
- 支持 SPI 接口模式
- 包含 512byte 的 EEPROM
- 包含 64byte 的 FIFO
- 数字电路具有 TTL/CMOS 两种电压工作模式
- 软件控制的 power down 模式
- 一个可编程计时器
- 一个中断处理器
- 一个串行输出输入口
- 启动配置可编程
- 数字，模拟和发射模块都有独立的电源供电，电压范围从 3V 到 5V
- 封装形式为 SOP24 小型封装

2　结构图

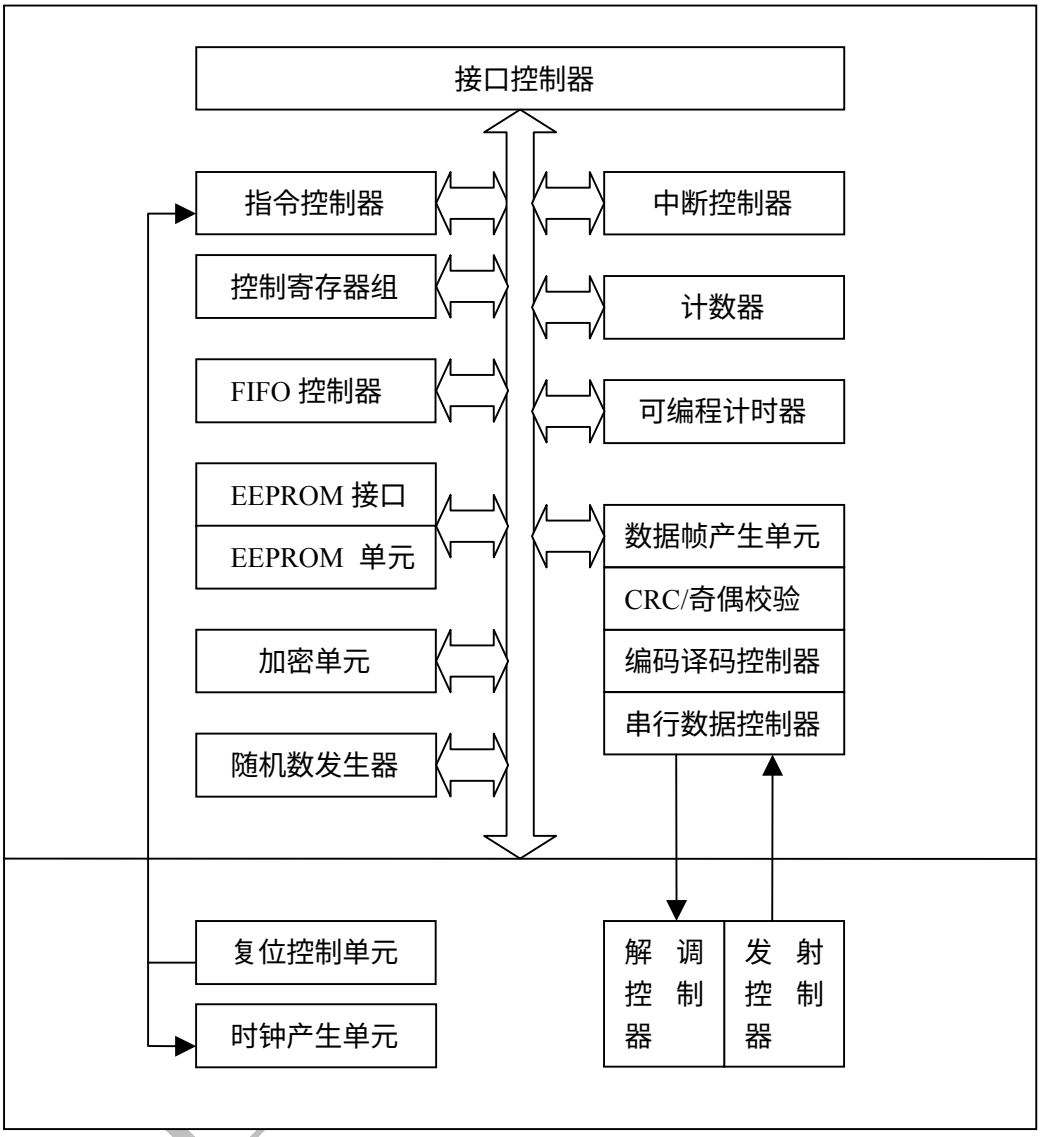


图 2－1：FM1702SL 结构图

3　管脚信息

3.1管脚配置

粗体字标识的管脚由 AVDD 和 AVSS 电源组供电
粗体框标识的管脚由 TVDD 和 TVSS 电源组供电
其他管脚由 DVDD 和 DVSS 电源组供电

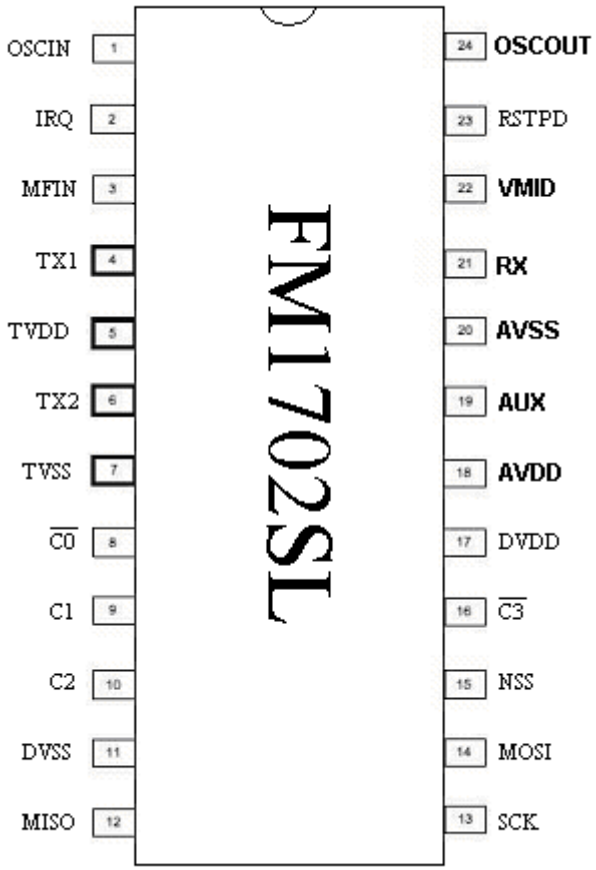


图 3－1：FM1702SL 管脚配置（SOP24 封装）

3.2 管脚描述

引脚序号	引脚名称	类型	引脚描述
1	OSCIN	I	**晶振输入**：fosc = 13.56MHz
2	IRQ	O	**中断请求**：输出中断源请求信号
3	MFIN	I	**串行输入**：接收满足 ISO14443A 协议的数字串行信号
4	TX1	O	**发射口 1**：输出经过调制的 13.56MHz 信号
5	TVDD	PWR	**发射器电源**：提供 TX1 和 TX2 的输出能量
6	TX2	O	**发射口 2**：输出经过调制的 13.56MHz 信号
7	TVSS	PWR	**发射器地**
8	CO	I	固定接低电平
9	C1	I	固定接高电平
10	C2	I	固定接高电平
11	DVSS	PWR	**数字地**
12	MISO	O	**主入从出**：SPI 接口下数据输出

管脚描述（续上页）

13	SCK	I	**串行时钟（SCK）**：SPI 接口下时钟信号
14	MOSI	I	**主出从入**：SPI 接口下数据输入
15	NSS	I	**接口选通**：选通 SPI 接口模式
16	C3	I	固定接低电平
17	DVDD	PWR	**数字电源**
18	AVDD	PWR	**模拟电源**
19	AUX	O	**模拟测试信号输出**：输出模拟测试信号，测试信号由 TestAnaOutSel 寄存器选择
20	AVSS	PWR	**模拟地**
21	RX	I	**接收口**：接收外部天线耦合过来的 13.56MHz 卡回应信号
22	VMID	PWR	**内部参考电压**：输出内部参考电压 注意：该管脚必须外接 68nF 电容
23	RSTPD	I	**复位及掉电信号**：高电平时复位内部电路，晶振停止工作，内部输入管脚和外部电路隔离；下沿触发内部复位程序
24	OSCOUT	O	**晶振输出**

表 3－1：FM1702SL 管脚描述

4 数字接口

4.1 支持的微处理器接口概述

FM1702SL 支持 SPI 微处理器接口
读写时序参照 22.5.2.1
在 SPI 通信方式下，FM1702SL 只能作为 slave 端，SCK 时钟需由 master 端提供。

4.2 自动侦测微处理器接口类型

在每一次上电或硬件复位后，FM1702SL 会复位微处理器接口处理模块，并且通过检测控制管脚上的电平来设置 SPI 接口

4.3 SPI 接口说明

FM1702SL 的 SPI 接口符合标准的 SPI 接口协议，并且只作为 slave 使用。

4.3.1. 读数据

按以下格式可以读出 n byte 的数据：

	byte 0	byte 1	byte 2	……..	byte n	byte n+1
MOSI	adr 0	adr 1	adr. 2	…….	adr n	00
MISO	XX	data 0	data 1	……	data n-1	data n

地址 byte 必须符合以下格式。第一 byte 的最高位定义了模式。若要从 FM1702SL 读出数据，则该位置‘1’。6－1 比特定义了地址，最后一比特必须置‘0’。
最后一 byte 应置‘0’。

address (MOSI)	bit 7, MSB	bit 6 - bit 1	bit 0
byte 0	1	address	RFU (0)
byte 1 to byte n	RFU (0)	address	RFU (0)
byte n+1	0	0	0

4.3.2. 写数据

按以下格式可以写 n byte 的数据到 FM1702SL：

	byte 0	byte 1	byte 2	…….	byte n	byte n+1
MOSI	adr	data 0	data 1	…….	data n-1	data n
MISO	XX	XX	XX	…….	XX	XX

地址 byte 必须符合以下格式。第一 byte 的最高位定义了模式。若要向 FM1702SL 写入数据，则该位置‘0’。6－1 比特定义了地址，最后一比特必须置‘0’。

该 SPI 写模式将所有数据都写入同一个给定的地址，这样可以高效的将数据写入 FM1702SL 的 FIFO 中。

Address line (MOSI)	MSB	bit 6 - bit 1	bit 0
byte 0	0	address	RFU (0)
byte 1 to byte n+1	data		

5 FM1702SL 寄存器组

5.1 FM1702SL 寄存器组概述

FM1702SL 的内部寄存器按功能不同分成 8 组，每组为一页，包含 8 个寄存器：
Page0：指令和状态寄存器组
Page1：控制和状态寄存器组
Page2：发射及编码控制寄存器组
Page3：接收及解码控制寄存器组
Page4：时间及校验控制寄存器组
Page5：FIFO，Timer 及 IRQ 控制寄存器组
Page6：预留寄存器组
Page7：预留寄存器组

Page	地址（hex）	寄存器名	功能
0	0	Page	选择寄存器组
	1	Command	指令寄存器
	2	FIFOData	64byte FIFO 的输入输出寄存器
	3	PrimaryStatus	发射器，接收器及 FIFO 的标识位寄存器
	4	FIFOLength	当前 FIFO 内 byte 数
	5	SecondaryStatus	各种状态标识寄存器
	6	InterruptEn	中断使能/禁止控制寄存器
	7	InterruptRq	中断请求标识寄存器
1	8	Page	选择寄存器组
	9	Control	各种控制标识寄存器
	A	ErrorFlag	上一条指令结束后错误标识
	B	CollPos	侦测到的第一个冲突位的位置
	C	TimerValue	当前 Timer 值
	D	CRCResultLSB	CRC 协处理器低 8 位
	E	CRCResultMSB	CRC 协处理器高 8 位
	F	BitFraming	调整面向 bit 的帧格式
2	10	Page	选择寄存器组
	11	TxControl	发射器控制寄存器
	12	CWConductance	选择发射脚 TX1 和 TX2 发射天线的阻抗
	13	PreSet13	预设寄存器，不要改变内容
	14	PreSet14	预设寄存器，不要改变内容
	15	ModWidth	选择载波调制宽度
	16	PreSet16	预设寄存器，不要改变内容
	17	PreSet17	预设寄存器，不要改变内容

FM1702SL 寄存器组（续上页）

3	18	Page	选择寄存器组
	19	RXControl1	接收器控制寄存器
	1A	DecoderControl	解码控制寄存器
	1B	BitPhase	调整发射器和接收器时钟相差
	1C	Rxthreshold	选择 bit 解码的阈值
	1D	PreSet1D	预设寄存器，不要改变内容
	1E	RxControl2	解码控制及选择接收源
	1F	ClockQControl	时钟产生控制寄存器
4	20	Page	选择寄存器组
	21	RxWait	选择发射和接收之间的时间间隔
	22	ChannelRedundancy	选择数据校验种类和模式
	23	CRCPresetLSB	CRC 预置寄存器低 8 位
	24	CRCPresetMSB	CRC 预置寄存器高 8 位
	25	PreSet25	预设寄存器，不要改变内容
	26	RFU	预留寄存器
	27	PreSet27	预设寄存器，不要改变内容
5	28	Page	选择寄存器组
	29	FIFOLevel	定义 FIFO 溢出级别
	2A	TimerClock	选择 Timer 时钟的分频
	2B	TimerControl	选择 Timer 启动/停止条件
	2C	TimerReload	Timer 预置值
	2D	IRQPinConfig	IRQ 输出配置
	2E	PreSet2E	预设寄存器，不要改变内容
	2F	PreSet2F	预设寄存器，不要改变内容
6	30	Page	选择寄存器组
	31	CryptoSelect	认证模式选择
	32	RFU	预留寄存器
	33	RFU	预留寄存器
	34	RFU	预留寄存器
	35	RFU	预留寄存器
	36	RFU	预留寄存器
	37	RFU	预留寄存器
7	38	Page	选择寄存器组
	39	RFU	预留寄存器
	3A	RFU	预留寄存器
	3B	RFU	预留寄存器
	3C	RFU	预留寄存器
	3D	RFU	预留寄存器
	3E	RFU	预留寄存器
	3F	RFU	预留寄存器

表 5－1：FM1702SL 寄存器组

5.1.1 寄存器位权限

每一个寄存器里的每一位按其功能都有不同的读写权限

缩写	权限	描述
r/w	读和写	这些位可以被微处理器读出和写入，他们只是用作控制，所以不会被内部状态机改写
dy	动态	这些位可以被微处理器读出和写入，并且他们可以被内部状态机自动改写
r	只读	这些位由内部状态机控制，只能被微处理器读出
w	只写	这些位用作控制，只能被微处理器写入。读取这些位得到的没有意义的数值

表 5－2：寄存器位的权限及描述

5.2 寄存器描述

5.2.1 PAGE0：指令和状态寄存器组

5.2.1.1 Page 寄存器

名字	Page							
功能	选择寄存器组							
地址	0x00，0x08，0x10，0x18，0x20，0x28，0x30，0x38							
复位值	10000000，0x80							
位	7	6	5	4	3	2	1	0
位名	UsePageSelect	0	0	0	0	0	0	0
位权限	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	位名	功能
7	UsePageSelect	初始化 SPI 接口时使用，见 11.4
6-0	0000000	预留值

5.2.1.2 Command 寄存器

名字	Command							
功能	开始或结束指令的执行							
地址	0x01							
复位值	X0000000，0xX0							
位	7	6	5	4	3	2	1	0
位名	IFDetectBusy	0	Command					
位权限	r	r	dy	dy	dy	dy	dy	dy

位描述

位	位名	功能
7	IFDetectBusy	接口检测的状态： 0：表示接口检测结束 1：表示接口检测正在进行
6	0	预留值
5-0	Command	运行并保存当前的指令

5.2.1.3 FifoData 寄存器

名字	FIFOData							
功能	64byte FIFO 的输入输出寄存器							
地址	0x02							
复位值	XXXXXXXX，0xXX							
位	7	6	5	4	3	2	1	0
位名	FIFOData							
位权限	dy	dy	dy	dy	dy	dy	dy	dy

位描述

位	位名	功能
7-0	FIFOData	64byte FIFO 数据接口，通过该寄存器将数据写入或读出 FIFO

5.2.1.4 Primary Status 寄存器

名字	PrimaryStatus							
功能	表示发射器，接收器及 FIFO 的状态							
地址	0x03							
复位值	XXXXXXXX，0xXX							
位	7	6	5	4	3	2	1	0
位名	0	ModemState			IRQ	Err	HiAlert	LoAlert
位权限	r	r	r	r	r	r	r	r

位描述

位	位名	功能
7	0	预留值
6-4	ModemState	显示当前发射器或接收器的状态：
3	IRQ	表示 InterrupEn 寄存器使能的任何一个中断源是否有请求
2	Err	1：表示 ErrorFlag 寄存器里有错误状态位置 1
1	HiAlert	1：表示 FIFO 中的数据数满足下列公式： HiAlert = (64-FIFOLength) ≤ WaterLevel
0	LoAlert	1：表示 FIFO 中的数据数满足下列公式： LoAlert = FIFOLength ≤ WaterLevel

状态	状态名	描述
000	Idle	发射器和接收器空闲
001	TxSOF	正在发射数据帧帧头
010	TxData	正在发射数据
011	TxEOF	正在发射数据帧帧尾
100	GoToRx1	接收器开始的中间状态
	GoToRx2	接收器开始的中间状态
101	PrepareRx	接收器工作前等待 RxWait 寄存器定义的时间
110	AwaitingRx	接收器打开，准备接收数据
111	Receiving	正在接收数据

5.2.1.5 FIFOLength 寄存器

名字	FIFOLength							
功能	FIFO 中数据的 byte 数							
地址	0x04							
复位值	00000000，0x00							
位	7	6	5	4	3	2	1	0
位名	0	FIFOLength						
位权限	r	r	r	r	r	r	r	r

位描述

位	位名	功能
7	0	预留值
6-0	FIFOLength	标识 FIFO 中数据的 byte 数，每写一个数据到 FIFO，FIFOLength 加 1，每读出一个数据，FIFOLength 减 1

5.2.1.6 Secondary Status 寄存器

名字	SecondaryStatus							
功能	不同的状态位							
地址	0x05							
复位值	01100000，0x60							
位	7	6	5	4	3	2	1	0
位名	TRun-ning	E2-Ready	CRC-Ready	0	0	RxLastBits		
位权限	r	r	r	r	r	r	r	r

位描述

位	位名	功能
7	TRunning	1：表示 Timer 正在运行
6	E2Ready	1：表示 EEPROM 擦写过程结束
5	CRCReady	1：表示 CRC 计算结束
4-3	00	预留值
2-0	RxLastBits	表示接收到的最后一 byte 数据中正确的 bit 数。若 0，表示整个 byte 都是正确的

5.2.1.7 InterruptEn 寄存器

名字	InterruptEn							
功能	使能或禁止中断请求							
地址	0x06							
复位值	00000000，0x00							
位	7	6	5	4	3	2	1	0
位名	SetlEn	0	TimerlEn	TxlEn	RxlEn	IdlelEn	HiAlertlEn	LoAlertlEn
位权限	w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	位名	功能
7	SetlEn	1：使能 InterruptEn 寄存器里置 1 的中断源 0：禁止 InterruptEn 寄存器里置 1 的中断源 如： 写 0xA0 到 InterruptEn，表示使能 Timer 中断，若有 Timer 中断请求，Timer 中断请求会反应到 primaryStatus 寄存器的 IRQ 位 写 0x20 表示禁止 Timer 中断，Timer 中断请求不会反应到 PrimaryStatus 寄存器的 IRQ 位
6	0	预留值
5	TimerlEn	1：表示允许 Timer 中断请求（标识位为 TimerlRq）反应到 IRQ 管脚。这一位不能直接置成 1 或 0，只能通过 SetlEn 来置
4	TxlEn	1：表示允许发射器中断请求（标识位为 TxlRq）反应到 IRQ 管脚。这一位不能直接置成 1 或 0，只能通过 SetlEn 来置
3	RxlEn	1：表示允许接收器中断请求（标识位为 RxlRq）反应到 IRQ 管脚。这一位不能直接置成 1 或 0，只能通过 SetlEn 来置
2	IdlelEn	1 表示允许 idle 中断请求（标识位为 IdlelRq）反应到 IRQ 管脚。这一位不能直接置成 1 或 0，只能通过 SetlEn 来置
1	HiAlertlEn	1：表示允许 high alert 中断请求（标识位为 HiAlertlRq）反应到 IRQ 管脚。这一位不能直接置成 1 或 0，只能通过 SetlEn 来置
0	LoAlertlEn	1：表示允许 low alert 中断请求（标识位为 LoAlertlRq）反应到 IRQ 管脚。这一位不能直接置成 1 或 0，只能通过 SetlEn 来置

5.2.1.8 InterruptRq 寄存器

名字	InterruptRq							
功能	中断源请求标识							
地址	0x07							
复位值	00000000，0x00							
位	7	6	5	4	3	2	1	0
位名	SetlRq	0	TimerlRq	TxlRq	RxlRq	IdlelRq	HiAlertlRq	LoAlertlRq
位权限	w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	位名	功能
7	SetlRq	1：将 InterruptRq 寄存器里置 1 的标识位置 1 0：将 InterruptRq 寄存器里置 1 的标识位清 0 如： 写 0xA0 到 InterruptRq，表示将 Timer 中断源标识初始为 1 写 0x20 表示将 Timer 中断源标识初始为 0
6	0	预留值
5	TimerlRq	1：表示 TimerValue 寄存器里的值已经减为 0
4	TxlRq	1：表示有下列事件发生： Transceive 指令：所有的数据都发送了 Auth1 和 Auth2 指令：所有数据都发送了 WriteE2 指令：所有数据都擦写结束 CalcCRC 指令：所有数据都计算结束
3	RxlRq	1：表示接收数据结束，接收器停止
2	IdlelRq	1：表示指令寄存器被内部状态机清 0 如果任何未知的指令写入指令寄存器，该位也置 1 但是微处理器写 Idle 指令到指令寄存器不影响该位
1	HiAlertlRq	1：表示 HiAlert 置 1。和 HiAlert 不同，HiAlertlRq 保存这个状态并只能被 SetlRq 复位
0	LoAlertlRq	1：表示 LoAlert 置 1。和 LoAlert 不同，LoAlertlRq 保存这个状态并只能被 SetlRq 复位

5.2.2 PAGE1：控制和状态寄存器组

5.2.2.1 Page 寄存器

选择寄存器组，见 5.2.1.1Page 寄存器

5.2.2.2 Control 寄存器

名字	Control							
功能	各种控制标识							
地址	0x09							
复位值	00000000，0x00							
位	7	6	5	4	3	2	1	0
位名	0	0	Stand-By	Power Down	Crypto 1On	TStop-Now	TStart-Now	Flush-FIFO
位权限	r/w	r/w	dy	dy	dy	w	w	w

位描述

位	位名	功能
7-6	00	预留值
5	StandBy	置 1 则进入软件 PowerDown 模式。这种模式下，内部电路停止工作，晶振不停振
4	PowerDown	置 1 则进入软件 PowerDown 模式。这种情况下，内部电路和晶振都停止工作
3	Crypto1On	1：表示加密单元打开，并且所有的数据传输都经过加密 这一位只有在 Authent2 指令通过后才被内部状态机置 1
2	TStopNow	置 1 则立刻停止 Timer 读该位返回 0
1	TStartNow	置 1 则立即启动 Timer 读该位返回 0
0	FlushFIFO	置 1 则清空 FIFO 以及读写指针（FIFOLength 置 0），并且清 FIFOOvfl 标识位 读该位返回 0

5.2.2.3 ErrorFlag 寄存器

名字	ErrorFlag							
功能	上一条指令结束后错误标识							
地址	0x0A							
复位值	01000000，0x40							
位	7	6	5	4	3	2	1	0
位名	0	KeyErr	Access Err	FIFO-Ovfl	CRC-Err	FramingErr	Parity-Err	CollErr
位权限	r	r	r	r	r	r	r	r

位描述

位	位名	功能
7	0	预留值
6	KeyErr	1：表示 LoadKeyE2 或 LoadKey 指令中输入的数据不符合规定的密钥格式 0：开始 LoadkeyE2 或 LoadKey 指令
5	AccessErr	1：违反 PROM 读写权限 0：开始 EEPROM 相关指令
4	FIFOOvfl	1：表示 FIFO 数据已满，微处理器或内部状态机仍然往 FIFO 里写数据，FIFO 溢出
3	CRCErr	1：表示 RxCRCEn 置 1 的情况下 CRC 校验出错。这一位在接收器开始（PrepareRx 状态）时自动清 0
2	FramingErr	1：表示 SOF 出错。这一位在接收器开始（PrepareRx 状态）时自动清 0
1	ParityErr	1：表示数据奇偶校验出错。这一位在接收器开始（PrepareRx 状态）时自动清 0
0	CollErr	1：表示有冲突位。这一位在接收器开始（PrepareRx 状态）时自动清 0

5.2.2.4 CollPos 寄存器

名字	CollPos							
功能	侦测到的第一个冲突位的位置							
地址	0x0B							
复位值	00000000，0x00							
位	7	6	5	4	3	2	1	0
位名	CollPos							
位权限	r	r	r	r	r	r	r	r

位描述

位	位名	功能
7-0	CollPos	表示接收数据过程中侦测到的第一个冲突位的位置

5.2.2.5 TimerValue 寄存器

名字	TimerValue							
功能	Timer 当前值							
地址	0x0C							
复位值	XXXXXXXX，0xXX							
位	7	6	5	4	3	2	1	0
位名	TimerValue							
位权限	r	r	r	r	r	r	r	r

位描述

位	位名	功能
7-0	TimerValue	显示当前 Timer 值

5.2.2.6 CRCResultLSB 寄存器

名字	CRCResultLSB							
功能	CRC 协处理器低 8 位							
地址	0x0D							
复位值	XXXXXXXX，0xXX							
位	7	6	5	4	3	2	1	0
位名	CRCResultLSB							
位权限	r	r	r	r	r	r	r	r

位描述

位	位名	功能
7-0	CRCResultLSB	显示 CRC 寄存器低 8 位，该寄存器的数据仅在 CRCReady 为 1 时有效

5.2.2.7 CRCResultMSB 寄存器

名字	CRCResultMSB							
功能	CRC 协处理器高 8 位							
地址	0x0E							
复位值	XXXXXXXX，0xXX							
位	7	6	5	4	3	2	1	0
位名	CRCResultMSB							
位权限	r	r	r	r	r	r	r	r

位描述

位	位名	功能
7-0	CRCResultMSB	显示 CRC 寄存器高 8 位，该寄存器的数据仅在 CRCReady 为 1 时有效

5.2.2.8 BitFraming 寄存器

名字	BitFraming							
功能	调整面向 bit 的帧格式							
地址	0x0F							
复位值	00000000，0x00							
位	7	6	5	4	3	2	1	0
位名	0	RxAlign			0	TxLastBits		
位权限	r/w	dy	dy	dy	r/w	dy	dy	dy

位描述

位	位名	功能
7	0	预留值
6-4	RxAlign	定义收到的第一个 bit 在 FIFO 中存放的位置，接收结束后，RxAlign 自动清 0 如： RxAlign ＝ 0：收到的第一个 byte 的最低位存放在 FIFO 中的第 0 位 RxAlign ＝ 1：收到的第一个 byte 的最低位存放在 FIFO 中的第 1 位
3	0	预留值
2-0	TxLastBits	定义了最后一个 byte 中要发送出去的 bit 数，000 表示最后一个 byte 中所有 bit 都发送出去。 发送结束后，TxLastBits 自动清 0

5.2.3 PAGE2：发射及编码控制寄存器组

5.2.3.1 Page 寄存器

选择寄存器组，见 5.2.1.1Page 寄存器

5.2.3.2 TxControl 寄存器

名字	TxControl							
功能	控制发射器逻辑行为							
地址	0x11							
复位值	01011000，0x58							
位	7	6	5	4	3	2	1	0
位名	0	ModulatorSource		1	TX2Inv	TX2Cw	TX2RF En	TX1RF En
位权限	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	位名	功能
7	0	预留值，不要更改
6-5	ModulatorSouce	选择调制源： 00：LOW 01：HIGH 10：内部编码器 11：MFIN 管脚
4	1	预留值，不要更改
3	TX2Inv	1：TX2 管脚输出 13.56MHz 反相能量载波
2	TX2Cw	1：TX2 管脚输出持续的 13.56MHz 非调制能量载波 0：TX2 管脚输出持续的 13.56MHz 调制能量载波
1	TX2RFEn	1：TX2 管脚输出 13.56MHz 经发送数据调制过的能量载波 0：TX2 管脚输出固定的电平
0	TX1RFEn	1：TX1 管脚输出 13.56MHz 经发送数据调制过的能量载波 0：TX1 管脚输出固定的电平

5.2.3.3 CwConductance 寄存器

名字	CwConductance							
功能	选择发射脚 TX1 和 TX2 发射天线的电导							
地址	0x12							
复位值	00111111，0x3F							
位	7	6	5	4	3	2	1	0
位名	0	0	GsCfgCW					
位权限	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	位名	功能
7-6	00	预留值，不要更改
5-0	GsCfgCW	定义了输出电导，用来调整输出能量以及电流消耗和操作距离 注意：详细信息见 13.3

5.2.3.4 PreSet13 寄存器

名字	PreSet13							
功能	预留寄存器							
地址	0x13							
复位值	00111111，0x3F							
位	7	6	5	4	3	2	1	0
位名	PreSet13							
位权限	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	位名	功能
7-0	PreSet13	预留值，不要更改

5.2.3.5 PreSet14 寄存器

名字	PreSet14							
功能	预留寄存器							
地址	0x14							
复位值	00011001，0x19							
位	7	6	5	4	3	2	1	0
位名	PreSet14							
位权限	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	位名	功能
7-0	PreSet14	预留值，不要更改

5.2.3.6 ModWidth 寄存器

名字	ModWidth							
功能	定义调制宽度							
地址	0x15							
复位值	00010011，0x13							
位	7	6	5	4	3	2	1	0
位名	ModWidth							
位权限	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	位名	功能
7-0	ModWidth	定义调制宽度 Tmod ＝ 2（ModWidth＋1）/fc

5.2.3.7 PreSet16 寄存器

名字	PreSet16							
功能	预留寄存器							
地址	0x16							
复位值	00000000，0x00							
位	7	6	5	4	3	2	1	0
位名	PreSet16							
位权限	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	位名	功能
7-0	PreSet16	预留值，不要更改

5.2.3.8 PreSet17 寄存器

名字	PreSet17							
功能	预留寄存器							
地址	0x17							
复位值	00000000，0x00							
位	7	6	5	4	3	2	1	0
位名	PreSet17							
位权限	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	位名	功能
7-0	PreSet17	预留值，不要更改

5.2.4 PAGE3：接收及解码控制寄存器组

5.2.4.1 Page 寄存器

选择寄存器组，见 5.2.1.1Page 寄存器

5.2.4.2 RxControl1 寄存器

名字	RxControl1							
功能	控制接收器行为							
地址	0x19							
复位值	01110011，0x73							
位	7	6	5	4	3	2	1	0
位名	0	1	1	1	0	0	Gain	
位权限	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	位名	功能
7-2	011100	预留值，不要更改
1-0	Gain	定义接收器放大器增益： 00：27 dB 01：30 dB 10：38 dB 11：42dB

5.2.4.3 DecoderControl 寄存器

名字	DecoderControl							
功能	控制解码器行为							
地址	0x1A							
复位值	00001000，0x08							
位	7	6	5	4	3	2	1	0
位名	0	0	ZeroAfterColl	0	1	0	0	0
位权限	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	位名	功能
7-6	00	预留值，不要更改
5	ZeroAferColl	1：所有接收到的冲突位之后的数据都置 0
4-0	01000	预留值，不要更改

5.2.4.4 BitPhase 寄存器

名字	BitPhase							
功能	调整发射器和接收器时钟相差							
地址	0x1B							
复位值	10101101，0xAD							
位	7	6	5	4	3	2	1	0
位名	BitPhase							
位权限	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	位名	功能
7-0	BitPhase	定义发射器和接收器时钟的相位差，该值影响数据接收质量

5.2.4.5 RxThreshold 寄存器

名字	RxThreshold							
功能	选择 bit 解码的阈值							
地址	0x1C							
复位值	11111111，0xFF							
位	7	6	5	4	3	2	1	0
位名	MinLevel				CollLevel			
位权限	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	位名	功能
7-4	MinLevel	定义解码器能够接收的最小信号，若信号强度低于该值，则不被处理
3-0	CollLevel	定义 Manchester 编码的信号里弱半 bit 相对强半 bit 产生冲突必须达到的最小值

5.2.3.8 PreSet1D 寄存器

名字	PreSet1D							
功能	预留寄存器							
地址	0x1D							
复位值	00000000，0x00							
位	7	6	5	4	3	2	1	0
位名	PreSet1D							
位权限	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	位名	功能
7-0	PreSet1D	预留值，不要更改

5.2.4.7 RxControl2 寄存器

名字	RxControl2							
功能	解码控制及选择接收源							
地址	0x1E							
复位值	01000001，0x41							
位	7	6	5	4	3	2	1	0
位名	RcvClkSell	RxAutoPD	0	0	0	0	DecoderSource	
位权限	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	位名	功能
7	RcvClkSell	1：选择 I 时钟作为接收器时钟 0：选择 Q 时钟作为接收器时钟
6	RxAutoPD	1：接收器在接收数据前自动打开和接受数据后自动关闭，用于节省功耗 0：接收器始终打开
5	0000	预留值，不要更改
1-0	DecoderSource	选择解码器源： 00：LOW 01：内部解调器 10：MFIN 输入的 Manchester 编码的副载波 11：MFIN 输入的 Manchester 编码的基带信号

5.2.4.8 ClockQControl 寄存器

名字	ClockQControl							
功能	控制 Q 时钟产生							
地址	0x1F							
复位值	000XXXXX，0xXX							
位	7	6	5	4	3	2	1	0
位名	ClkQ180Deg	ClkQCalib	0	ClkQDelay				
位权限	r	r/w	r/w	dy	dy	dy	dy	dy

位描述

位	位名	功能
7	clkQ180Deg	若 I-Q 时钟相位差超过 180 度，该位置 1，否则为 0
6	ClkQCalib	0：复位及接收过程结束后自动校正 Q 时钟相位 1：不校正 Q 时钟相位
5	0	预留值，不要更改
4-0	ClkQDelay	用于产生 Q 时钟的预置值

5.2.5 PAGE4：时间及校验控制寄存器组

5.2.5.1 Page 寄存器

选择寄存器组，见 5.2.1.1Page 寄存器

5.2.5.2 RxWait 寄存器

名字	RxWait							
功能	选择发射和接收之间的时间间隔							
地址	0x21							
复位值	00000110，0x06							
位	7	6	5	4	3	2	1	0
位名	RxWait							
位权限	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	位名	功能
7-0	RxWait	数据发送后，接收器等待 RxWait 定义的 bit 时钟数，在这段时间内，Rx 上收到的任何信号都被忽略

5.2.5.3 ChannelRedundancy 寄存器

名字	ChannelRedundancy							
功能	选择数据校验种类和模式							
地址	0x22							
复位值	00000011，0x03							
位	7	6	5	4	3	2	1	0
位名	0	CRCMSBFirst	CRC3309	CRC8	RxCRCEn	TxCRCEn	Parity-Odd	Parity-En
位权限	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	位名	功能
7	0	预留值，不要更改
6	CRCMSBFirst	1：数据流高位先进 CRC 协处理器 0：数据流低位先进 CRC 协处理器
5	CRC3309	1：CRC 算法切换为 ISO/IEC3309 0：CRC 算法切换为 ISO1443A
4	CRC8	1：计算 8bit CRC 0：计算 16bit CRC
3	RxCRCEn	1：对接收数据进行 CRC 校验，接收数据最后两 byte CRC 不送入 FIFO。如果校验出错，CRCErrFlag 置 1 0：接收过程不进行 CRC 校验
2	TxCRCEn	1：对发射数据计算 CRC 校验码，并附加在数据流尾一起发送 0：不发送 CRC 校验码
1	ParityOdd	1：选择奇校验 0：选择偶校验
0	ParityEn	1：每个发送的数据 byte 后都插入 1bit 校验位，同样每个接收到的数据 byte 都进行奇偶校验 0：不发送及检查奇偶校验位

5.2.5.4 CRCPresetLSB 寄存器

名字	CRCPresetLSB							
功能	CRC 预置值低 8 位							
地址	0x23							
复位值	01100011，0x63							
位	7	6	5	4	3	2	1	0
位名	CRCPresetLSB							
位权限	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	位名	功能
7-0	CRCPresetLSB	CRC 协处理器预置值低 8 位

5.2.5.5 CRCPresetMSB 寄存器

名字	CRCPresetMSB							
功能	CRC 预置值高 8 位							
地址	0x24							
复位值	01100011，0x63							
位	7	6	5	4	3	2	1	0
位名	CRCPresetMSB							
位权限	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	位名	功能
7-0	CRCPresetMSB	CRC 协处理器预置值高 8 位

5.2.5.6 PreSet25 寄存器

名字	PreSet25							
功能	预留寄存器							
地址	0x25							
复位值	00000000，0x00							
位	7	6	5	4	3	2	1	0
位名	PreSet25							
位权限	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	位名	功能
7-0	PreSet25	预留值，不要更改

5.2.5.7 RFU 寄存器

名字	RFU							
功能	预留寄存器							
地址	0x26							
复位值	0x00							
位	7	6	5	4	3	2	1	0
位名	RFU							
位权限	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	位名	功能
7-0	RFU	预留值

5.2.5.8 PreSet27 寄存器

名字	PreSet27							
功能	预留寄存器							
地址	0x27							
复位值	XXXXXXXX，0xXX							
位	7	6	5	4	3	2	1	0
位名	PreSet27							
位权限	w	w	w	w	w	w	w	w

位描述

位	位名	功能
7-0	PreSet27	预留值，不要更改

5.2.6 PAGE5：FIFO，Timer 及 IRQ 控制寄存器组

5.2.6.1 Page 寄存器

选择寄存器组，见 5.2.1.1Page 寄存器

5.2.6.2 FIFOLevel 寄存器

名字	FIFOLevel							
功能	定义 FIFO 溢出级别							
地址	0x29							
复位值	00000100，0x08							
位	7	6	5	4	3	2	1	0
位名	0	0	WaterLevel					
位权限	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	位名	功能
7-6	00	预留值，不要更改
5-0	WaterLevel	定义了 FIFO 向上或向下溢出的级别： 若 FIFO 里剩下的空间小于等于 WaterLevel，则 HiAlert 置 1 若 FIFO 里已有的数据小于等于 LoAlert，则 LoAlert 置 1

5.2.6.3 TimerClock 寄存器

名字	TimerClock							
功能	选择 Timer 时钟的分频							
地址	0x2A							
复位值	00000111，0x07							
位	7	6	5	4	3	2	1	0
位名	0	0	TAutoRestart		TPreScaler			
位权限	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	位名	功能
7-6	00	预留值，不要更改
5-4	TAutoRestart	1：Timer 递减到 0 后，自动重新从 TReloadValue 开始递减
3-0	TPreScaler	定义 Timer 时钟频率 f_{Timer}，范围从 0 到 21： $f_{Timer} = 13.56MHz/2^{TPreScaler}$

5.2.6.4 TimerControl 寄存器

名字	TimerControl							
功能	选择 Timer 启动/停止条件							
地址	0x2B							
复位值	00000110，0x06							
位	7	6	5	4	3	2	1	0
位名	0	0	0	0	TStop-RxEnd	TStopRxBegin	TStartTxEnd	TStartTxBegin
位权限	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	位名	功能
7-4	0000	预留值，不要更改
3	TStopRxEnd	1：数据接收完毕后 Timer 自动停止 0：Timer 不受该条件影响
2	TStopRxBegin	1：第一个正确的 bit 接收完 Timer 自动停止 0：Timer 不受该条件影响
1	TStartTxEnd	1：数据发送结束后 Timer 自动启动。若此时 Timer 已经在运行，则 Timer 重新启动，从 TReloadValue 开始递减 0：Timer 不受该条件影响
0	TStartTxBegin	1：第一个 bit 发送后 Timer 自动启动。若此时 Timer 已经在运行，则 Timer 重新启动，从 TReloadValue 开始递减 0：Timer 不受该条件影响

5.2.6.5 TimerReload 寄存器

名字	TimerReload							
功能	定义 Timer 预置值							
地址	0x2C							
复位值	00001010，0x0A							
位	7	6	5	4	3	2	1	0
位名	TReloadValue							
位权限	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	位名	功能
7-0	TReloadValue	Timer 若启动，则从该值开始倒计数。改变该寄存器值只能影响下一次 Timer 启动的初值

5.2.6.6 IRQPinConfig 寄存器

名字	IRQPinConfig							
功能	配置 IRQ 输出行为							
地址	0x2D							
复位值	00000010，0x02							
位	7	6	5	4	3	2	1	0
位名	0	0	0	0	0	0	IRQInv	IRQPushPull
位权限	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	位名	功能
7-3	000000	预留值，不要更改
1	IRQInv	1：IRQ 管脚为 IRq bit 的反相输出 0：IRQ 管脚为 IRq bit 的同相输出
0	IRQPushPull	1：IRQ 管脚为标准 CMOS 输出 0：IRQ 管脚为开漏输出

5.2.6.7 PreSet2E 寄存器

名字	PreSet2E							
功能	预留寄存器							
地址	0x2E							
复位值	XXXXXXXX，0xXX							
位	7	6	5	4	3	2	1	0
位名	PreSet2E							
位权限	w	w	w	w	w	w	w	w

位描述

位	位名	功能
7-0	PreSet2E	预留值，不要更改

5.2.6.8 PreSet2F 寄存器

名字	PreSet2F							
功能	预留寄存器							
地址	0x2F							
复位值	XXXXXXXX，0xXX							
位	7	6	5	4	3	2	1	0
位名	PreSet2F							
位权限	w	w	w	w	w	w	w	w

位描述

位	位名	功能
7-0	PreSet2F	预留值，不要更改

5.2.7 预留寄存器组

5.2.7.1 Page 寄存器

选择寄存器组，见 5.2.1.1Page 寄存器

5.2.7.2 CryptoSelect 寄存器(FM1705)

名字	CryptoSelect							
功能	选择认证模式							
地址	0x31							
复位值	00000000，0x00							
位	7	6	5	4	3	2	1	0
位名	0	0	0	0	0	0	0	Crypto Select
位权限	w	w	w	w	w	w	w	w

位描述

位	位名	功能
7-1	0000000	预留值，不要更改
0	CryptoSelect	1：兼容 SH 标准的认证模式 0：兼容 MIFARE 标准的认证模式

5.2.7.3 RFU 寄存器

名字	RFU							
功能	预留寄存器							
地址	0x32，0x33，0x34，0x35，0x36，0x37							
复位值	XXXXXXXX，0xXX							
位	7	6	5	4	3	2	1	0
位名	RFU							
位权限	w	w	w	w	w	w	w	w

位描述

位	位名	功能
7-0	RFU	预留值

5.2.8 预留寄存器组

5.2.8.1 Page 寄存器

选择寄存器组，见 5.2.1.1Page 寄存器

5.2.8.2 RFU 寄存器

名字	RFU							
功能	预留寄存器							
地址	0x39，0x3A，0x3B，0x3C，0x3D，0x3E，0x3F							
复位值	XXXXXXXX，0xXX							
位	7	6	5	4	3	2	1	0
位名	RFU							
位权限	w	w	w	w	w	w	w	w

位描述

位	位名	功能
7-0	RFU	预留值

6 EEPROM 结构

6.1 EEPROM 存储器结构

块号	块地址	byte地址	读写权限	内容
0	0	00…0F	r	产品信息区
1	1	10…1F	r/w	寄存器复位初始值存放区
2	2	20…2F	r/w	
3	3	30…3F	r/w	寄存器初始值存放区
4	4	40…4F	r/w	
5	5	50…5F	r/w	
6	6	60…6F	r/w	
7	7	70…7F	r/w	
8	8	80…8F	w	密钥存放区
9	9	90…9F	w	
10	A	A0…AF	w	
11	B	B0…BF	w	
12	C	C0…CF	w	
13	D	D0…DF	w	
14	E	E0…EF	w	
15	F	F0…FF	w	
16	10	100…10F	w	
17	11	110…11F	w	
18	12	120…12F	w	
19	13	130…13F	w	
20	14	140…14F	w	
21	15	150…15F	w	
22	16	160…16F	w	
23	17	170…17F	w	
24	18	180…18F	w	
25	19	190…19F	w	
26	1A	1A0…1AF	w	
27	1B	1B0…1BF	w	
28	1C	1C0…1CF	w	
29	1D	1D0…1DF	w	
30	1E	1E0…1EF	w	
31	1F	1F0…1FF	w	

表6－1：EEPROM存储器结构

6.2 寄存器初始值存放区

10hex 到 2Fhex 寄存器在*初始阶段*（见 11.3 章）会被自动初始成*寄存器复位初始值存放区*内的值。用户也可以使用 LoadConfig 指令（见 16.6.1 章）将这些寄存器初始化成*寄存器初始值存放区*里的值。
注意：

- Page 寄存器（10_{hex}，18_{hex}，20_{hex}，28_{hex}）不会被初始化
- 不要更改任何 PreSet 寄存器的内容
- 所有预留的寄存器或寄存器位都为‘0’，不要更改内容

6.2.1 寄存器复位初始值存放区

EEPROM 第一和第二块存放初始阶段 10_{hex} 到 $2F_{hex}$ 寄存器的初始值，默认值见 6.3.2。对应关系如下：

EEPROM Byte地址	寄存器地址	注释
10_{hex} (块1，Byte0)	10_{hex}	跳过
11_{hex}	11_{hex}	复制
…	…	…
$2F_{hex}$(块2，Byte15)	$2F_{hex}$	复制

表6－4：寄存器地址对应表

6.2.2 寄存器复位初始值

寄存器复位初始值在产品测试过程中被初始化，在每次上电或复位过程中，这些值被用来初始化内部寄存器 10_{hex} 到 $2F_{hex}$。

EEPROM Byte地址	寄存器地址	值	描述
10	10	00	Page
11	11	58	TxControl
12	12	3F	CwConductance
13	13	3F	PreSet13
14	14	19	PreSet14
15	15	13	ModWidth
16	16	00	PreSet16
17	17	00	PreSet17
18	18	00	Page
19	19	73	RxControl1
1A	1A	08	DecoderControl
1B	1B	AD	BitPhase
1C	1C	FF	RxThreshold
1D	1D	00	PreSet1D
1E	1E	41	RxControl2
1F	1F	00	ClockQControl
20	20	00	Page
21	21	06	RxWait
22	22	03	ChannelRedundancy
23	23	63	CRCPresetLSB
24	24	63	CRCPresetMSB
25	25	00	PreSet25
26	26	00	RFU
27	27	00	PreSet27
28	28	00	Page
29	29	08	FIFOLevel
2A	2A	07	TimerClock
2B	2B	06	TimerControl
2C	2C	0A	TimerReload
2D	2D	02	IRQPinConfig
2E	2E	00	PreSet2E
2F	2F	00	PreSet2F

表 6－5：寄存器复位初始值地址对应表

6.2.3 寄存器初始值

EEPROM 块 3 到块 7 保存着寄存器初始化值，用户可以用 LoadConfig 指令（见 16.6.1）将内部寄存器 10hex 到 20hex 初始化为这些值。对应关系如下：

E²PROM Byte 地址	寄存器 Address	注释
EEPROM起始byte地址	10hex	跳过
EEPROM起始byte地址 + 1	11hex	复制
…	…	…
EEPROM起始byte地址 + 31	2Fhex	复制

表6－6：寄存器初始值地址对应表

注意： 寄存器初始值模块是可读＋可写权限，用户可以用它来初始化寄存器，同样也可以用来来存放数据

6.3 密钥存放区

6.3.1 密钥格式

密钥必须以规定的格式存放在 EEPROM 内。每一 byte 的密钥分为低四位（k0 到 k3）和高四位(k4 到 7)，每四位按位及位反存放在一个 byte 内。LoadKeyE2 和 LoadKey 指令会按此来检查密钥是否满足这种格式。使用这种格式，6byte 的 key 需存放在 12byte EEPROM 内：

KEY Byte	0 (LSB)		1		…	5 (MSB)	
KEY	$\overline{k7k6k5k4}k7k6k5k4$	$\overline{k3k2k1k0}k3k2k1k0$	$\overline{k7k6k5k4}k7k6k5k4$	$\overline{k3k2k1k0}k3k2k1k0$	…	$\overline{k7k6k5k4}k7k6k5k4$	$\overline{k3k2k1k0}k3k2k1k0$
E²PROM Byte 地址	n	n+1	n+2	n+3	…	n+10	n+11
例子	$5A_{hex}$	$F0_{hex}$	$5A_{hex}$	$E1_{hex}$		$5A_{hex}$	$A5_{hex}$

表6－7：KEY存放格式

例子：key 为 A0A1A2A3A4A5，则在 EEPROM 中存放格式为：$5AF05AE15AD25AC35AB45AA5_{hex}$

6.3.2 密钥的存放

FM1702SL 保留了 384byte EEPROM 空间用于保存密钥。密钥可以在这些 byte 中任意位置开始存放。
例子：如果一条密钥的 byte0 存放在 $12F_{hex}$，则 byte1 存放在 130_{hex}，byte2 存放在 131_{hex}，一直到 byte11 存放　在 $13A_{hex}$

7 FIFO

7.1 概述

FM1702SL 包含一个 8x64 的并行 FIFO，保存微处理器和 FM1702SL 之间通信的数据

7.2 访问规则

FIFO 通过 FIFOData 寄存器输入和输出数据。向这个寄存器里写一 byte 数据即向 FIFO 里添加一 byte 数据，同时 FIFO 写指针加一。从这个寄存器读一 byte 数据即从 FIFO 里读出一 byte 数据，同时 FIFO 读指针加一。FIFOLength 寄存器记录读/写指针之间的长度。

当 FM1702SL 执行一条指令时，内部状态机可能会对 FIFO 进行内部读/写操作，所以除了指令本身要求外，微处理器在 FM1702SL 指令执行过程中不要对 FIFO 执行不正确的访问

下列表格给出 FM1702SL 指令执行过程中对 FIFO 的访问情况：

指令	微处理器允许		注释
	写数据到FIFO	从FIFO读数据	
StartUp	-	-	
Idle	-	-	
Transmit	√	-	写指令参数或追加发射数据
Receive	-	√	读取接收数据
Transceive	√	√	写指令参数，接收过程中读接收到的数据
WriteE2	√	-	写指令参数或追加要写入EEPROM的数据
ReadE2	√	√	写指令参数，在指令执行过程中读取从EEPROM读出的数据
LoadKeyE2	√	-	写指令参数
LoadKey	√	-	写指令参数
Authent1	√	-	写指令参数
Authent2	-	-	
LoadConfig	√	-	写指令参数
CalcCRC	√	-	写指令参数

表7－1：FIFO访问规则

7.3 控制 FIFO

除了读写 FIFO 外，用户可以通过设置 FlushFIFO 位来复位 FIFO 指针。在这种情况下，FIFO 被清空，FIFOLength 置 0，FIFOOvfl 标识位被清除，FIFO 内原有的数据不再有效。

7.4 FIFO 状态信息

微处理器可以通过下列寄存器获得 FIFO 状态：
FIFO 中数据长度：FIFOLength
FIFO 渐满警告：HiAlert
FIFO 渐空警告：LoAlert
FIFO 溢出（FIFO 满的情况下继续写数据到 FIFO）：FIFOOvfl

FIFO 可以产生两个中断请求：
如果 LoAlertRq 置 1 且 LoAlert 变为 1，会激活 IRQ 管脚
如果 HiAlertRq 置 1 且 HiAlert 变为 1，会激活 IRQ 管脚

LoAlert 变为 1 的条件：
LoAlert ＝ FIFOLength ≤ WaterLevel

HiAlert 变为 1 的条件：
HiAlert ＝ （64－FIFOLength）≤ WaterLevel

7.5 FIFO 相关寄存器

标识	寄存器	地址 寄存器，bit位
FIFOLength	FIFOLength	0x04, bits 6-0
FIFOOvfl	ErrorFlag	0x0A, bit 4
FlushFIFO	Control	0x09, bit 0
HiAlert	PrimaryStatus	0x03, bit 1
HiAlertIEn	InterruptIEn	0x06, bit 1
HiAlertIRq	InterruptIRq	0x07, bit 1
LoAlert	PrimaryStatus	0x03, bit 0
LoAlertIEn	InterruptIEn	0x06, bit 0
LoAlertIRq	InterruptIRq	0x07, bit 0
WaterLevel	FIFOLevel	0x29, bits 5-0

表7－2：FIFO相关寄存器

8 中断请求系统

8.1 概述

如果有中断请求事件发生，FM1702SL 会将 PrimaryStatus 寄存器里的 IRq 位置 1，同时激活 IRQ 管脚。IRQ 上的信号可以用来向微处理器发出中断请求。

8.1.1 中断源概述

下表列出了各个中断标识对应的中断源及其标识建立的条件。
TimerIRq 表明 TIMER 有中断请求。当 TIMER 自减到 0（TAutoRestart 置 0）或 TpreLoad 值（TautoRestart 置 1）时 TimerlRq 标识建立
TxIRq 标识在三种情况下会被建立：
1．所有数据发送完毕
2．CRC 协处理器将 FIFO 中所有数据计算完毕，此时 CRCReady 也被置 1
3．所有 FIFO 中的数据都被写入 EEPROM，此时 E2Ready 也被置 1
RxIRq 表明接收器接收数据完毕
IdleIRq 表明指令执行完毕
HiAlerIRq 表明 HiAlert 被置 1，见 7.4 章
LoAlertIRq 表明 LoAlert 被置 1，见 7.4 章

中断标识	中断源	建立条件
TimerIRq	TIMER	Timer自减到0或TpreLoad值
TxIRq	发射器	数据发送结束
	CRC协处理器	FIFO中所有数据CRC计算结束
	E²PROM	FIFO中所有数据已被写入EEPROM
RxIRq	接收器	接收器接收数据结束
IdleIRq	指令寄存器	指令执行结束
HiAlertIRq	FIFO	HlAlert置1，FIFO渐满
LoAlertIRq	FIFO	LoAlert置1，FIFO渐空

表8－1：中断源

8.2 中断应用

8.2.1 控制中断请求及其标识

FM1702SL 通过设置 InterrupRq 寄存器中的 bit 向微处理器指明相应的中断请求，并且通过 InterrupEn 寄存器来打开或禁止这些中断请求。

寄存器	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
InterruptEn	SetIEn	rfu	TimerIEn	TxIEn	RxIEn	IdleIEn	HiAlertIEn	LoAlertIEn
InterruptRq	SetIRq	rfu	TimerIRq	TxIRq	RxIRq	IdleIRq	HiAlertIRq	LoAlertIRq

表8－2：中断控制寄存器

只要有任何一个中断请求标识被置1并且相应的中断使能位打开，则PrimaryStatus寄存器中的IRq位被置1。如果有多个中断标识同时置1，则所有的中断请求位做“或”运算后激活IRq标识及IRQ管脚

8.2.2 访问中断寄存器

中断请求位由内部状态机来设置。此外微处理器也可以使用 SetIEn 或 SelIRq 位来设置或清除他们。如果想要某个特定的中断位要置 1，则将该位置 1 并置 SetIxx 位为 1。如果想要没个特定的中断位置 0，则将该位置 1 并置 SetIxx 位为 0。

8.3 IRQ 管脚配置

用户可以通过 IRQPinConfig 寄存器来控制 IRQ 管脚的输出行为

IRQInv： 若置 0，则 IRQ 管脚电平与 IRq bit 位电平相同
若置 1，则 IRQ 管脚电平与 IRq bit 位电平相反

IRQPushPull： 若置 1，IRQ 管脚为标准 CMOS 输出特性
若置 0，IRQ 管脚为开漏输出，且必须外接上拉电阻

注意：在复位过程（见 11.2 章）IRQInv 置 1 且 IRQPushPull 置 0，此时 IRQ 管脚为高阻状态

8.4 中断相关寄存器

标识	寄存器	地址 寄存器，bit位
HiAlertIEn	InterruptEn	0x06, bit 1
HiAlertIRq	InterruptRq	0x07, bit 1
IdleIEn	InterruptEn	0x06, bit 2
IdleIRq	InterruptRq	0x07, bit 2
IRq	PrimaryStatus	0x03, bit 3
IRQInv	IRQPinConfig	0x07, bit 1
IRQPushPull	IRQPinConfig	0x07, bit 0
LoAlertIEn	InterruptEn	0x06, bit 0
LoAlertIRq	InterruptRq	0x07, bit 0
RxIEn	InterruptEn	0x06, bit 3
RxIRq	InterruptRq	0x07, bit 3
SetIEn	InterruptEn	0x06, bit 7
SetIRq	InterruptRq	0x07, bit 7
TimerIEn	InterruptEn	0x06, bit 5
TimerIRq	InterruptRq	0x07, bit 5
TxIEn	InterruptEn	0x06, bit 4
TxIRq	InterruptRq	0x07, bit 4

表8－3：中断相关寄存器

9 TIMER

9.1 概述

FM1702SL 包含一个 TIMER，选择芯片 13.56MHz 时钟的不同分频作为计时时钟。微处理器可以按需要将其配置为：

- Timeout－Counter
- Watch－Dog Counter
- Stop Watch
- Programmable One－Shot
- Periodical Trigger

TIMER 可以用来计算两个事件的时间间隔或标识某一事件在某一精确的时间后发生。TIMER 可以被若干事件触发，但不会影响任何事件的进行。TIMER 相关的标识位也可以被用来产生中断请求

9.2 TIMER 应用

9.2.1 控制 TIMER

TIMER 的主要部分是一个自减计数器。只要该计数器值不为 0，就会在时钟控制下做自减操作。如果 AutoRestart 置 1，则 TIMER 不会自减到 0。当 TIMER 计数到 1 时会在下一个时钟自动加载 TimerReolad 寄存器的值

TIMER 将 TimerReload 寄存器的值加载后自动开始运行，这可由下列事件触发：
- 开始发送第一个 bit（TxBegin 事件）并且 TStartTxBegin 置 1
- 发送完最后一个 bit（TxEnd 事件）并且 TStartTxEnd 置 1
- TStartNow 被微处理器置 1

注意：TIMER 被触发后都会从 TimerReload 寄存器装载初始值，重新开始倒计数

TIMER 可以被下列事件停止：
- 收到第一个 bit（RxBegin 事件）并且 TStopRxBegin 置 1
- 接收器接收结束（RxEnd 事件）并且 TStopRxEnd 置 1
- TIMER 自减到 0 并且 TAutoRestart 置 0
- TStopNow 被微处理器置 1

写一个新的数据到 TimerReload 寄存器并不立即影响计数器，TimerReload 寄存器只在 TIMER 下一次重新开始时改变 TIMER 的初始值。所以 TimerReload 寄存器可以在 TIMER 运行过程中被赋值。

如果 TIMER 是被 TstopNow 位停止的，则不会发出 TimerIRq 请求

9.2.2 TIMER 时钟周期

TIMER 时钟由芯片 13.56MHz 时钟分配得到。由 TPreScaler 寄存器决定分频数：

$$T_{TimerClock} = \frac{1}{f_{TimerClock}} = \frac{2^{T\,\mathrm{Pre}Scaler}}{13.56MHz}$$

TPreScaler 寄存器范围从 0 到 21，对应 $T_{TimerClock}$ 从 74ns 到 150ms

从上一个开始事件到目前的时间范围为 74ns 到 40s，计算公式如下：

$$T_{Timer} = \frac{T\,\mathrm{Re}loadValue - TimerValue}{f_{TimerClock}}$$

9.2.3 TIMER 状态

SecondaryStatus 寄存器里的 TRunning bit 位标识了当前 TIMER 的状态。任何事件触发 TIMER 启动后，TRunning 标识置 1，TIMER 停止后，TRunning 置 0。Running 标识置 1 的下一个时钟起 TimerValue 开始自减。当前 TIMER 值可从 TimerValue 寄存器读出。

9.3 TIMER 用途

9.3.1 TIMER-OUT 和 WATCH-DOG-COUNTER

TIMER 启动后自动从 TimerValue 开始递减，如果定义好的结束事件（如接收到第一个 bit）发生，TIMER 就停止，没有 TIMER 中断产生。
另一方面，如果结束事件没有发生（如卡在规定的时间内没有回发数据），则 TIMER 一直计数到 0 并产生中断请求，这个中断可以通知微处理器预期的事件没有在规定的时间内发生

9.3.2 STOP WATCH

微处理器可以计算 TIMER 开始到结束之间的时间：

$$VT = (TReload_{value} - Timer_{value}) * T_{Timer}$$

9.3.3 PROGRAMMABLE ONE-SHOT TIMER

微处理器启动 TIMER 后等待 TIMER 的中断，在规定的时间后中断会出现

9.3.4 PERIODICAL TRIGGER

若微处理器将 TAutoRestart 置 1，则会周期性（T_{Timer}）的产生中断请求

9.4 TIMER 相关寄存器

标识	寄存器	地址 寄存器，bit位
TautoRestart	TimerClock	0x2A, bit 5
TimerValue	TimerValue	0x0C, bits 7-0
TimerReloadValue	TimerReload	0x2C, bits 7-0
TpreScaler	TimerClock	0x2A, bits 4-0
Trunning	SecondaryStatus	0x05, bit 7
TstartNow	Control	0x09, bit 1
TstartTxBegin	TimerControl	0x2B, bit 0
TstartTxEnd	TimerControl	0x2B, bit 1
TstopNow	Control	0x09, bit 2
TstopRxBegin	TimerControl	0x2B, bit 2
TstopRxEnd	TimerControl	0x2B, bit 3

表 9－1：TIMER 相关寄存器

10 省电工作模式

10.1 Hard Power Down 模式

在 RSTPD 上加高电平就进入到 Hard Power Down 模式。这将关闭所有的内部电流消耗包括振荡器。所有的数字输入驱动器与输入管脚分离而由内部决定电平（RSTPD 自身除外）。所有输出管脚电平被固定在某一值。

如下表所示：

符号	管脚位置	类型	描述
OSCIN	1	I	没有与输入分离，被拉到 AVSS
IRQ	2	O	高阻
MFIN	3	I	与输入分离
TX1	5	O	高
TX2	7	O	低
C0	9	I	与输入分离
C1	10	I	与输入分离
C2	11	I	与输入分离
MISO	12	I	与输入分离
SCK	13	I	与输入分离
MOSI	14	I/O	与输入分离
NSS	15	I	与输入分离
C3	16	I	与输入分离
AUX	19	O	高阻
RX	21	I	不变
VMID	22	A	拉到 AVDD
RSTPD	23	I	不变
OSCOUT	24	O	高

表 10－1： 在 Hard Power Down 方式下管脚信号

10.2 Soft Power Down 模式

当 Control 寄存器的 PowerDown 位被置为 1 之后，芯片立即进入 Soft Power Down 模式。所有的内部电流都被关断（包括振荡器驱动）。

与 Hard Power Down 模式不同，数字输入驱动没有与输入管脚分离，依然保持工作状态。数字输出管脚保持原有状态。

当 Control 寄存器的 PowerDown 位复位之后，需要 512 个时钟周期退出 Soft Power Down 模式，由 PowerDown 标志来标示。对这一位复位并不能立即清除它，当退出 Soft Power Down 模式后它会自动清零。

注意：如果使用内部振荡器，必须考虑到它由 AVDD 供电，到振荡器稳定且时钟周期能被内部逻辑电路检测到必定会有一定的延迟 tOSC。

10.3 Stand By 模式

当 Control 寄存器的 StandBy 位被置为 1 之后，芯片立即进入 Stand By 模式。所有的内部电流都被关断（包括除振荡器驱动外所有的内部时钟驱动）。

与 Hard Power Down 模式不同，数字输入驱动没有与输入管脚分离，依然保持工作状态。数字输出管脚保持原有状态。

与 Soft Power Down 模式不同，振荡器不需要恢复时间。

当 Control 寄存器的 StandBy 位复位之后，OSCIN 管脚上需要经过 4 个时钟周期才能退出 Stand By 模式，由 StandBy 标志来标示。对这一位复位并不能立即清除它，当退出 Stand By 模式后它会自动清零。

10.4 接收器关闭

在不需要的时候关闭接收器，在数据接收之前打开接收器可以降低功耗。在将 RxAutoPD 设置成 1 之后可以自动实现这一功能。如果将这一位设置成 0，则接收器一直处于工作状态。

11 启动过程

启动过程如下图所示：

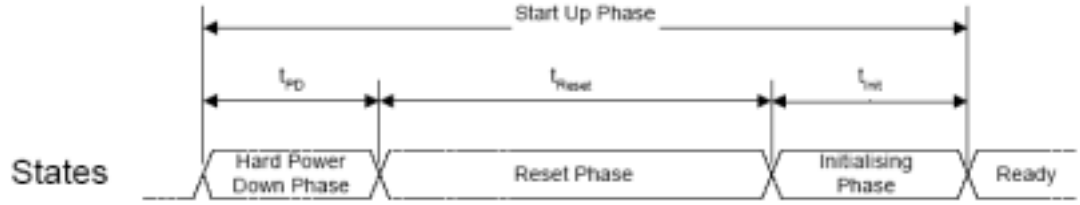


图 11－1 启动过程

11.1 Hard Power Down 阶段

在下列情况下会进入 Hard Power Down 阶段

- 由于 DVDD 管脚上加电引起的上电复位（在 DVDD 低与数字电路复位阀值电压时启动）
- 由于 AVDD 管脚上加电引起的上电复位（在 AVDD 低与模拟电路复位阀值电压时启动）
- 在 RSTPD 管脚上加高电平（当 RSTPD 管脚上电平为高时启动）

11.2 复位阶段

复位阶段自动跟随 Hard Power Down 阶段。一旦振荡器工作稳定，需要 512 个时钟周期完成该状态。在复位阶段，一些寄存器的值由硬件预置。相应的寄存器复位值在介绍寄存器时已经给出（见 5.2 章）。

注意：如果使用内部振荡器，必须考虑到它由 AVDD 供电，到振荡器稳定且时钟周期能被内部逻辑电路检测到必定会有一定的延迟 tOSC。

11.3 初始化阶段

初始化阶段自动跟随复位阶段，需要 128 个时钟周期。在初始化阶段，EEPROM 的第 1 和第 2 扇区内容被复制到 10hex 至 2Fhex 寄存器。

注意：在产品测试时，FM1702SL 被初始化成默认值，这有助于最大限度的减少微处理器对芯片的配置工作。

11.4 初始化 SPI 接口方式

芯片复位后，必须进行一次初始化程序以便初始化 SPI 接口模式，而且可以同步微处理器和 FM1702SL 的启动工作。

在整个启动过程中，Command 寄存器的值始终为 3Fhex。在初始化阶段结束后，FM1702SL 自动进入 Idle 状态，Command 寄存器的值随之变成 00hex。

执行下列程序确保初始化 SPI 接口：

- 读 Command 寄存器，直到 6 比特值变成 00hex。此时内部初始化阶段已经结束，芯片准备好接收外部指令。
- 往 Page 寄存器写 80hex 初始化 SPI 接口
- 读 Command 寄存器，如果它的值为 00hex，则 SPI 接口已经初始化成功
- 往 Page 寄存器写 0hex 开始使用 SPI 接口

完成接口初始化之后，可以通过往 Page 寄存器写 00hex 切换到线性寻址方式。

12　振荡器电路

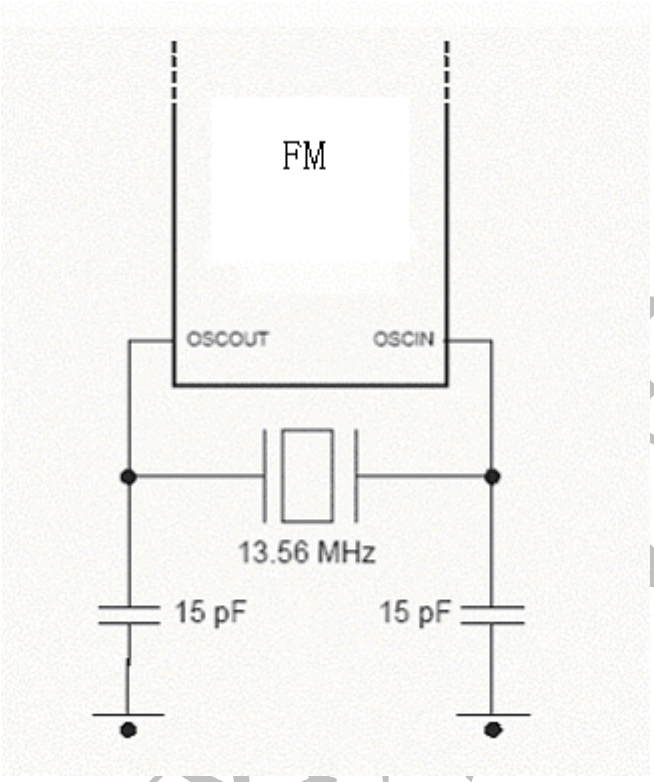


图 12－1：晶振连接方法

在同步时序电路中，提供给 FM1702SL 的时钟作为编码和解码的时基。因此，稳定的时钟频率对于正常的工作至关重要。最好使用内部时钟振荡器以及推荐的电路来满足这一要求。如果使用外部时钟源，时钟信号必须加到 OSCIN 管脚。在这种情况下，必须注意时钟的占空比以及抖动，时钟信号的品质必须得到保证。时钟信号应该满足 19.5.2 中所要求的。

注意：我们不推荐使用外部时钟源。

13 发射管脚 TX1 和 TX2

从 TX1 和 TX2 发射出去的是调制过的 13.56MHz 载波信号。辅以几个无源器件来匹配和滤波，它就可以直接驱动天线（见 18 章）。为此，输出电路的内部阻抗被设计得非常小。通过配置 TxControl 寄存器可以控制 TX1 和 TX2 的输出信号。

13.1 配置 TX1 和 TX2

下表说明了 TX1 的可能配置情况：

TxControl 配置情况	**包络信号**	**TX1 上的信号**
TX1RFEn		
0	X	低
1	0	低
1	1	13.56 MHz 载波

表 13－1：配置管脚 TX1

下表说明了 TX2 的可能配置情况：

Register Configuration in TxControl			**包络信号**	**TX2 上信号**
TX2RFEn	***TX2CW***	***InvTX2***		
0	X	X	X	低
1	0	0	0	调制后 13.56 MHz 的信号
			1	13.56 MHz 载波
		1	0	调制后 13.56 MHz 的信号，与 TX1 有 180° 相移
			1	13.56 MHz 载波，与 TX1 有 180° 相移
	1	0	X	13.56 MHz 载波
		1	X	13.56 MHz 载波，与 TX1 有 180° 相移

表 13－2：TX2 管脚的配置

13.2 工作距离与功耗的关系

用户可以通过调整天线驱动电压 TVDD，在最大工作距离和使用不同的天线匹配电路的功耗之间找到一个平衡点。

13.3 脉冲宽度

通过将数据按照 Miller 码编码，数据信息包含在包络中传输到卡。而且，Miller 码的每一个停顿也被编码成特定长度的脉冲。通过配置 ModWidth 寄存器可以调整这一脉冲的宽度。参考下面的公式：

$$T_{Pulse} = 2\frac{ModWidth + 1}{f_c}$$

其中，fc=13.56MHz。

14 接收电路

14.1 概述

FM1702SL 采用了正交解调电路来解调 RX 脚上的 ISO 14443 标准的副载波信号。ISO 14443-A 副载波信号是 Manchester 编码、ASK 调制信号。正交解调器使用两个不同的时钟：Q 时钟和 I 时钟(相差 90 度)。两路副载波信号被放大、滤波后经相关/求值/数字化电路解调后送入数字模块。

在信号处理过程中可以进行各种调整获得最佳性能。

14.2 信号接收过程

接收过程包括以下几个步骤：
首先对 13.56MHz 的载波信号进行正交解调，建议 Q 时钟进行自动校准(见 14.3.1)获得最佳效果，解调后的信号被增益可调的放大器放大。相关电路计算了接收信号与预期值的相似度，Bit phase 寄存器决定了发射器和接收器时钟的相差。在求值和数字化电路中检测有效比特且把结果送入 FIFO 寄存器。

14.3 接收器操作

通常，缺省设置可以使 FM1702SL 与卡进行数据传输。然而，在一些特定环境里用户设置可以获得更好的效果。

14.3.1 Q 时钟自动校准

接收器正交解调的概念是产生 I 时钟和与之相差 90 度的 Q 时钟。为了获得最佳的解调性能，Q 时钟和 I 时钟必须相差 90 度。FM1702SL 复位后自动进行校准。

ClkQCalib 值为 0 时, Transceive 命令结束后可以进行自动校准; *ClkQCalib* 值为 1 时, 禁止了除复位外所有的自动校准。也可以由软件通过设置 *ClkQCalib* 为 1 来进行自动校准。

Q 时钟自动校准持续 65 个振荡周期(约 4.8μs), *ClkQDelay* 值正比于 Q 时钟与 I 时钟的相差, 状态标识 *ClkQ180Deg* 表明 Q 时钟与 I 时钟的相差超过 180 度。

注意：

- 复位初始值设置了复位后 Q 时钟为自动校准。
- *ClkQCalib* 设置为 1 可以永久禁止自动校准。
- 可以通过微处理器向 *ClkQDelay* 写数据，目的是为了禁止自动校准且通过软件进行预设置。注意，通过软件设置延迟值时 *ClkQCalib* 必须已经被设置为 1 且至少经过了 4.8μs 的时间间隔。如果 *ClkQCalib* 为 0 时，设置的延迟值将在下一个自动校准阶段被刷新。

14.3.2 放大器

解调信号经过可调放大器的放大可以进行性能优化。通过 Gain[1:0]寄存器调整放大器的增益，下面是可选的增益因子:

寄存器设置	增益因子 [dB] (模拟结果)
0	20
1	24
2	31
3	35

表14－1：内部放大器的增益因子

14.3.3 相关电路

相关电路计算了接收信号与期望信号之间的匹配度，包括Q和I两路通道。对于每一路输入信号,相关器有两路输出，所以共有四个输出信号。相关电路需要卡信号的相位信息来进行性能优化，该信息由微处理器通过寄存器*BitPhase [7:0]*来设置，该值确定了发射器与接收器时钟的相位关系(tBitPhase =1/13.56 MHz的倍数)。

14.3.4 求值及数字化电路

对 Manchester 码信号的每个 bit-half，相关结果被求值。求值及数字化电路从两个 bit-half 的信号强度确定该比特是否有效，如果有效则确定它的值或是否包含冲突。

为了优化，用户可以选择下面的标准:

- *MinLevel*：确定了强 bit-half 有效的最低信号强度。
- *CollLevel*：确定了弱 bit-half 产生冲突的最低信号强度。如果信号强度低于该值，1 和 0 可以确定。
  CollLevel 确定了相应于强 half-bit 幅度的最低信号强度。

数据传输结束后，卡必须要在一定时间间隔(帧保护时间, ISO14443 标准)后才可以发送响应，通过 *RxWait* 寄存器设置该时间长度，*RxWait* 寄存器确定数据传输给卡后几个比特时间打开接收机。

如果 *RcvClkSelI* 寄存器设置为 1,相关器和求值电路采用 I 时钟；如果设置为 0,则采用 Q 时钟。

注意： 建议使用默认的 Q 时钟。

15 串行信号开关

15.1 概述

FM1702SL 包括两个主要模块：数字模块(包括状态机、编码器和解码逻辑等)，模拟模块(包括调制器、天线驱动器、接收机和放大电路)。这两模块的接口设计可以将接口信号送入 MFIN

15.2 时序信号开关的相关寄存器

DecoderSource 标识确定内部 Manchester 解码器输入信号的方式：

DecoderSource	**解码器输入信号**
0	常数 0
1	模拟部分的输出，缺省配置。
2	直接连接到MFIN, Manchester 码信号调制的847.5 kHz副载波信号。
3	直接连接到MFIN, Manchester 码信号。

表15－1：DecoderSource值

ModulatorSource 确定了所发射的 13.56MHz 载波的调制信号，该信号驱动 TX1 及 TX2 脚：

ModulatorSource	**调制器的输入信号**
0	常数 0 (TX1及TX2脚无载波).
1	常数 1 (TX1及TX2脚有持续的载波).
2	内部编码器的调制信号(包络), 缺省配置.
3	直接连接到MFIN, Miller 码信号.

表15－2：ModulatorSource值

16 FM1702SL 指令集

16.1 概述

FM1702SL 的行为由一个内部状态机决定，该状态机可执行一组专门的指令集。将某条指令代码写入指令寄存器可启动一个相应的命令的执行。

某些指令执行需携带参数和（或）数据，这些参数和数据主要通过 FIFO 进行交换。

16.2 命令行为简介

- 需要将数据流作为输入的命令会直接处理它在 FIFO 缓存找到的数据。
- 需要特定数量的参数的命令只有在接收到数量正确的参数后，才会开始执行。
- 在开始执行命令时 FIFO 缓存不会自动清空，因而，可以先将部分参数和（或）数据写入 FIFO 缓存，然后再开始命令的执行，同时写入剩余的参数和（或）数据。
- 由微处理器写入指令寄存器新的指令代码例如：*Idle-Command* 可以中断当前正在执行的除了 *StartUp-Command* 之外的任何命令.

16.3 FM1702SL 命令简介

16.3.1 基本说明

命令	指令代码 (16 进制)	功能	通过 FIFO 传递的参数和数据	返回的数据	见相关的章节
StartUp	3F	运行复位和初始化过程 注意：软件不能执行这条命令，只能通过上电或冷复位来做。	-	-	16.3.2
Idle	00	空指令，用来取消当前命令执行。	-	-	16.3.3
Transmit	1A	发送 FIFO 缓存数据	数据流	-	16.4.1
Receive	16	激活接收电路。 注意：内部状态机等待配置在 *RxWait* 寄存器中的时间过去后，才能真正启动接收 注意：这条命令和 *Transmit-Command* 没有时序关系，所以可以单独用于测试目的。	-	数据流	16.4.2
Transceive	1E	发送 FIFO 缓存数据，发送完后自动激活接收电路。 注意：内部状态机等待配置在 *RxWait* 寄存器中的时间过去后，才能真正启动接收 注意：这条命令是发送和接收的组合。	数据流	数据流	16.4.3
WriteE2	01	从 FIFO 缓存读取数据，并且写入内部 E^2PROM。	起始地址 LSB 起始地址 MSB 数据流	-	16.5.1
ReadE2	03	从内部 E^2PROM 读取数据，并且写入 FIFO 缓存。 注意：密钥不能读出。	起始地址 LSB 起始地址 MSB 字节数	数据流	16.5.2

基本说明（续上页）

LoadKeyE2	0B	将密钥从 E^2PROM 复制到 key 缓存。	起始地址 LSB 起始地址 MSB	-	16.8.1
LoadKey	19	将密钥从 FIFO 缓存复制到 key 缓存。	Byte0(LSB) Byte1 … Byte10 Byte11(MSB)	-	16.8.2
Authent1	0C	执行 Crypto1 算法的认证过程的第一步。	卡认证命令 卡的块地址 卡序列号的 LSB 卡序列号的 Byte1 卡序列号的 Byte2 卡序列号的 MSB	-	16.8.3
Authent2	14	执行 Crypto1 算法的认证过程的第二步。	-	-	16.8.4
LoadConfig	07	从 E^2PROM 读取数据，用于初始化寄存器。	起始地址 LSB 起始地址 MSB	-	16.6.1
CalcCRC	12	激活 CRC 协处理器。 注意：CRC 计算结果可从 *CRCResultLSB* 和 *CRCResultMSB* 寄存器中读到。	数据流	-	16.6.2

表 16－1：FM1702SL 命令说明

16.3.2 STARTUP COMMAND $3F_{HEX}$

命令	指令代码 (16 进制)	功能	通过 FIFO 传递的参数和数据	返回的数据
StartUp	3F	运行复位和初始化过程。 注意：软件不能执行这条命令，只能通过上电或硬件复位来做。	-	-

StartUp-Command 运行复位和初始化过程，无需发送或接收数据。不能被微处理器激活，只能在发生下列事件后自动启动：

- DVDD 管脚上引起上电复位
- DVDD 管脚上引起上电复位
- RSTPD 管脚上下降沿

在复位过程中，异步复位定义特定寄存器位；初始化过程中，用 E^2PROM 中的值定义特定寄存器。

当 *StartUp-Command* 结束后，*Idle-Command* 自动插入。

注意：

- 微处理器不准在 FM1702SL 正在执行 *StartUp-Command* 时进行写入操作。微处理器需读取命令寄存器 (*Command-Register*) 中是否为 *Idle-Command* 以确保 *StartUp-Command* 操作结束。（见 11.4 章）
- 当 *StartUp-Command* 执行时，只能读 page0 中的寄存器。
- *StartUp-Command* 不能被微处理器中断。

16.3.3 IDLE COMMAND 00_{HEX}

命令	指令代码 (16 进制)	功能	通过 FIFO 传递的参数和数据	返回的数据
Idle	00	空指令，用来取消当前命令执行。	-	-

Idle-Command 将 FM1702SL 切换到空闲状态，在这个状态中等待下一条命令。无需发送或接收数据。当前指令执行完成后，FM1702SL 自动进入空闲状态，同时将 *IdleIRq* 位置起来发出中断申请。如果由微处理器执行，可以取消当前正在执行的除了 *StartUp-Command* 之外的任何命令，但此时中断申请不会发出。

注意：用 *Idle-Command* 中断命令不会同时清 FIFO 中数据。

16.4 通讯命令

FM1702SL 全兼容 ISO14443，因此该读卡机芯片指令集非常灵活和全面。以下几个章节会讨论和卡通讯的命令，最后是三重认证过程。

16.4.1 TRANSMIT COMMAND $1A_{HEX}$

命令	指令代码 (16 进制)	功能	通过 FIFO 传递的参数和数据	返回的数据
Transmit	1A	发送 FIFO 缓存数据	数据流	-

Transmit-Command 从 FIFO 缓存中去数据然后传递到发送器。无需返回数据。只能由微处理器启动

16.4.1.1 使用 Transmit-Command

从下列中选择发送数据的顺序：

1. 当处于空闲状态时，将所有需要发送的数据写入 FIFO，而后，将 *Transmit-Command* 的指令码写入命令寄存器，启动发送。
   注意：这种方法最大可传输 64 字节。

2. 先将 *Transmit-Command* 的指令码写入命令寄存器。由于 FIFO 中无数据，则虽然发送使能，但发送过程只有等 FIFO 中写入第一字节数据才被启动。微处理器必须及时写入下一数据以保证在 RF 界面上有数据流传递。
   注意：此方法可以传递任意长度数据流，不过要求数据及时写入 FIFO 缓存。

3. 当处于空闲状态时，将部分需要发送的数据写入 FIFO，而后，将 *Transmit-Command* 的指令码写入命令寄存器，启动发送。如果当前 FM1702SL 处于 *Transmit-Command* 工作状态中，则可以由微处理器向 FIFO 写入数据，发送器则可以将这些数据添加到发送的数据流中。
   注意：此方法可以传递任意长度数据流，不过要求数据及时写入 FIFO 缓存。

如果发送器为保证 RF 界面上数据流的连续性，请求发送下一数据，但 FIFO 中为空，则 *Transmit-Command* 自动停止，内部状态机由发送状态跳回空闲状态。

注意：如果微处理器将写入 *Idle-Command* 和其他命令写入命令寄存器，覆盖了 *Transmit-Command*，下一个时钟发送即停止。这会导致不符合 ISO14443-A 的信号产生。

16.4.1.2 RF 通道冗余校验及帧格式

每次发送的帧由 SOF（帧头）、数据流、EOF（帧尾）三个阶段依次构成，可以通过 *PrimaryStatus-Register* 的 *ModemState* （见 16.4.4 章）来观察这些不同阶段。

如果 *ChannelRedundancy-Register* 的 *TxCRCEn* 位被置起，则 FM1702SL 计算出 CRC 添加到数据流后。CRC 计算方法由 *ChannelRedundancy-Register* 中设置决定。产生奇偶校验也由 *ChannelRedundancy-Register* 中的设置（*ParityEn* 位和 *ParityOdd* 位）决定。

16.4.1.3 发送面向位的帧

通过将 *TxLastBits* 位设为非 0 值，发送器可以配置成最后发送一个不完整的字节。见下图：

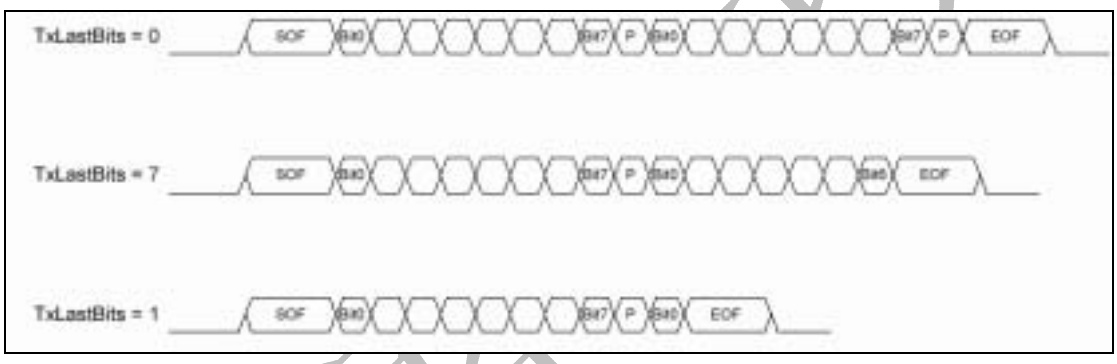


图 16－1 ：发送面向位的帧

上图也同时展示了如果如果 *ChannelRedundancy-Register* 的 *ParityEn* 位被置起，所由完整传送的字节都由奇偶校验位，但最后一个不完整的字节没有。发送结束后，*TxLastBits* 位被自动清成 0。

注意：如果 *TxLastBits* 位不为 0，则必须通过清 *ChannelRedundancy-Register* 的 *TxCRCEn* 位来禁止 CRC 产生。

16.4.1.4 发送长于 64 字节的帧

为发送长于 64 字节的帧，微处理器需在 *Transmit-Command* 执行时向 FIFO 继续写入数据，内部状态机在开始发送数据流的最后一前时会检查 FIFO 的状态。如下图：

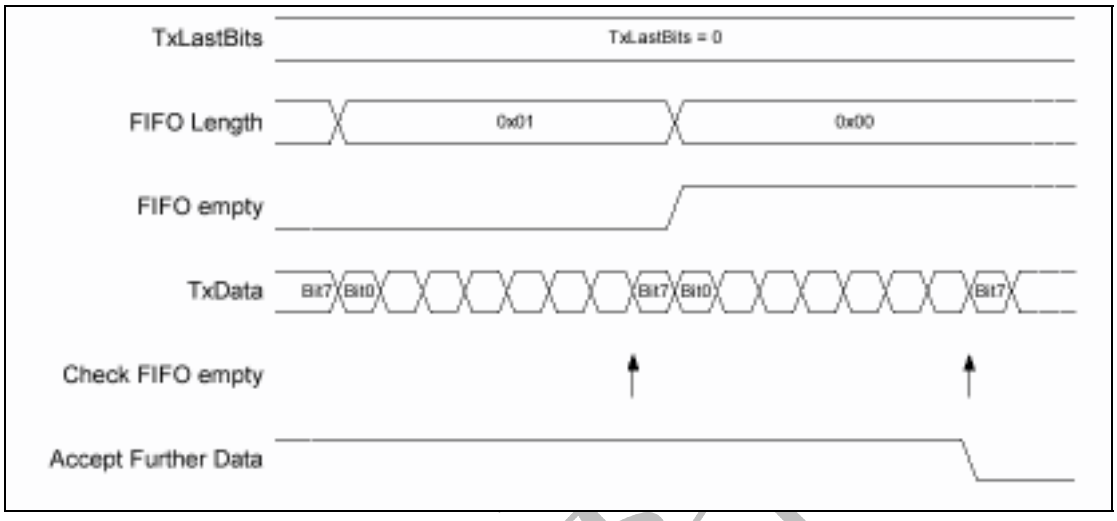


图 16－2：发送面向字节的帧的时序

只要内部信号‘Accept Further Data’为 1，写入 FIFO 的数据就会被添加到发送数据流后。如果为 0，则发送结束，此时才写入的数据不会被发送，而只保留在 FIFO 缓存中。

注意：如果使能奇偶校验位的产生（即 *ParityEn* 位被置起），奇偶校验位时是最后一位被发送的，这样会使‘Accept Further Data’信号多持续一位时间。

如果 *TxLastBits* 位不为 0，最后一字节不被完整发送，而只发送 *TxLastBits* 位定义的起始于最低有效位的位数。
这样，内部状态机会提前检查 FIFO 状态，如下图：

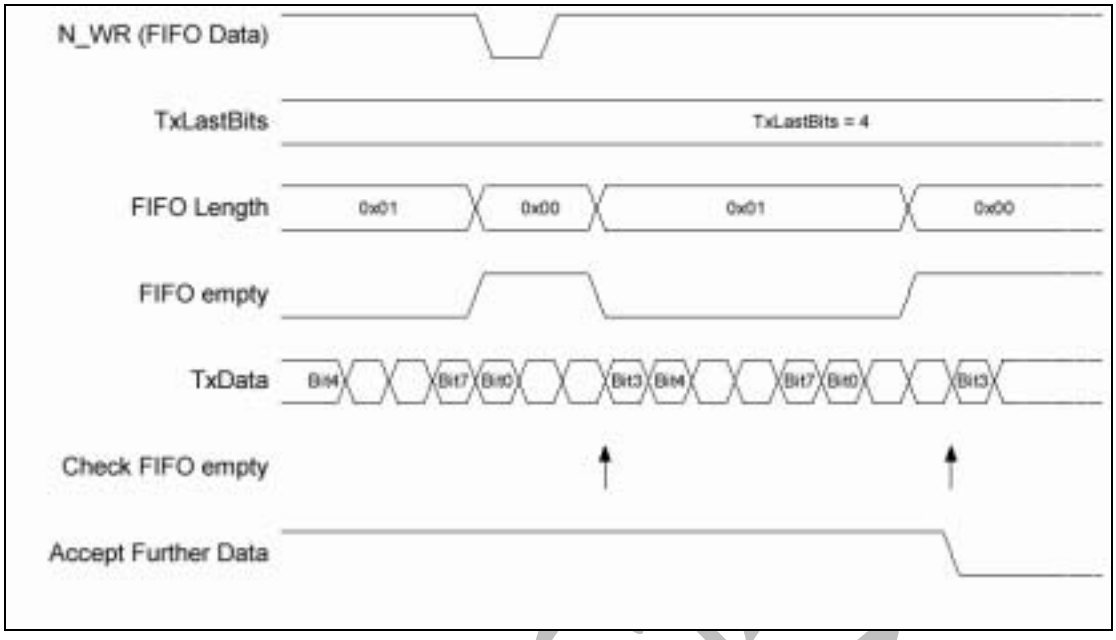


图 16－3：发送面向位的帧的时序

在上图的例子中，发送 Bit 3 后停止。如果经过配置，会加上 EOF。

上图还显示了在 FIFO 状态在被检测前 *FIFOData-Register* 正好有一个写操作，这就造成了‘FIFO empty’信号再次跳为 0，由此内部信号‘Accept Further Data’维持为 1，刚写的一个字节则通过 RF 界面被发送出去。

信号‘Accept Further Data’只能由‘Check FIFO empty’功能改变。这个功能作用是在预期发送的最后一位前一位检测‘FIFO empty’信号。

帧定义	检测位置
8 位，带奇偶校验	第 8 位
8 位，不带奇偶校验	第 7 位
x 位，　不带奇偶校验	第（x - 1）位

16.4.2 RECEIVE COMMAND 16_{HEX}

命令	指令代码 (16 进制)	功能	通过 FIFO 传递的参数和数据	返回的数据
Receive	16	激活接收电路。	-	数据流

Receive-Command 激活接收电路，所有从 RF 界面接收的数据可通过 FIFO 返回。可由微处理器启动，也可由执行 *Transceive-Command* 自动启动。

注意：这条命令和 *Transmit-Command* 没有时序关系，所以可以单独用于测试目的。

16.4.2.1 使用 Receive-Command

Transmit-Command 启动后，每一个位时钟会使 *RxWait-Register* 中的值减一。当从 3 减到 1 使，模拟部分的接收电路准备好并被激活。当计数值到 0，接收电路开始监视从 RF 界面接收的信号。如果信号电平超过 *MinLevel-Register* 中定义的水平时，接收电路于是开始译码。如果在接收电路探测到输入管脚 Rx 再也没有信号，则结束接收，译码器会置起 *RxIRq* 位表示申请中断。

检测 *PrimaryStatus-Register* 的 *ModemState*，可监视发送过程的不同阶段。（见 16.4.4 章）

注意：由于 *RxWait-Register* 中必须计数从 3 到 0 以有时间初始化模拟部分的接收电路，所以该寄存器至少置 3。

16.4.2.1 RF 通道冗余校验及帧格式

译码器在开始接收数据流等待 SOF。当检测到，它就激活串变并转换器并收集输入的数据位。每一个完整的字节都被传递到 FIFO。如果检测到 EOF 或者输入信号电平低于 *RxThreshold-Register* 中定义的 *MinLevel* 值时，接收和译码停止，*Idle-Command* 自动插入,随之产生给微处理器的中断申请并置起相应状态标志。

如果 *ChannelRedundancy-Register* 的 *RxCRCEn* 位被置起，接收到的数据中应包括 CRC 块，块的大小由 *ChannelRedundancy-Register* 的 *CRC8* 位决定是一个或两个字节。

注意：如果接收到的 CRC 块正确，就不会将它传递到 FIFO 缓存。这通过将收到的数据串行移过一个一字节或两字节（由定义的 CRC 方法决定）长的内部缓存实现。CRC 块将保留在内部缓存中，最后在延迟一个或两个字节时间后，FIFO 中含有除了 CRC 块的所有数据。如果 CRC 校验失败，则将所有数据包括错误的 CRC 块送入 FIFO。

ChannelRedundancy-Register 中的 *ParityEn* 位置起，收到的每一字节后应当跟一个奇偶校验位。如果 *ParityOdd* 位置起，预期的校验为奇校验，否则为偶校验。

16.4.2.2 冲突检测

在选卡阶段，如果有不止一张的卡，就会同时应答。FM1702SL 支持 ISO14443-A 规定的算法，通过所谓防冲突循环来解决卡序列号的数据冲突问题。该算法的基础是检测位冲突。

现在使用的位编码机制即 Manchester-coding 支持位冲突检测。如果在某位的前半和后半部分都检测到副载波调制，就会发送不是一个 1 或一个 0 而是一位冲突信号。FM1702SL 使用 *CollLevel* 的设置来区分 1 或 0 和一个位冲突，即当一位收到的数据中较小的半位大于 *CollLevel* 的定义，则说明检测到一个位冲突。

如果数据位检测到一个位冲突，则错误标志 *CollErr* 置起；如果检测到奇偶校验有一个位冲突，则错误标志 *ParityErr* 置起。

接收电路不受检测到的冲突位的制约，继续接收输入的数据，译码器会发送 1 给检测到冲突的位。

注:作为例外，如果 *ZeroafterColl* 位置起，所有第一个冲突位后接收到的位无论冲突或不冲突，都被强制为 0。这个特点可以简化软件实现兼容 ISO14443-A 防冲突机制。

当帧的第一位冲突被检测到，该位位置会存放在 *CollPos-Register*。

冲突位对应值如下：

冲突位	*CollPos-Register* 的值
SOF	0
LSByte 的 LSBit	1
…	…
LSByte 的 MSBit	8
第二个 Byte 的 LSBit	9
…	…
第二个 Byte 的 MSBit	16
第二个 Byte 的 LSBit	17
…	…

表 16－ 1：位冲突位置的返回值

由于奇偶校验位的冲突根据定义跟随在数据位的冲突后，所以奇偶校验位不包括在 *CollPos* 中。如果在 SOF 中检测到位冲突，则报告帧错误，而且没有数据传递到 FIFO，在这种情况下，接收电路继续监视输入信号直到收到错误输入数据流的结尾，产生正确的通知信号给微处理器。这将帮助微处理器决定什么时候允许发送数据给卡。

16.4.2.3 接收面向位的帧

接收电路可以处理含有不完整字节的数据流，支持面向位的帧需使用以下值：

- *RxAlign* 给第一个输入字节选择一个位偏移量，例如：如果 *RxAlign* 被设为 3，前 5 位被传递到 FIFO 缓存，接下来的位按字节打包并传递。完成接收后，*RxAlign* 被自动清空。
- *TxLastBits* 位返回最后接收一字节的有效位数，例如：如果在接收命令结束后 *TxLastBits* 位等于 5，说明最低 5 位有效。

只有帧错误标志 *FrameErr* 没有置起，*TxLastBits* 位才有效；如果 *RxAlign* 不为 0 且 *ParityEn* 有效，第一个奇偶校验位会被忽略而不被校验。

16.4.2.4 通讯错误

下表列出了错误标志建立条件：

原因	置起的标准位
接收的数据没有 SOF	FramingErr
CRC 块不等于计算值	CRCErr
接收到数据长度小于 CRC 块	CRCErr
奇偶校验位不等于计算值	ParErr
检测到冲突	CollErr

表 16－ 2：通讯错误

16.4.3 TRANSCEIVE COMMAND $1E_{HEX}$

命令	指令代码 (16 进制)	功能	通过 FIFO 传递的参数和数据	返回的数据
Transceive	1E	发送 FIFO 缓存数据，发送完后自动激活接收电路。	数据流	数据流

Transceive-Command 先执行 *Transmit-Command*（见 16.4.1 章），然后自动启动 *Receive-Command*（见 16.4.2 章），所有需发送的数据传递到 FIFO 缓存，所有接收的数据也传递到 FIFO 缓存。它只能由微处理器启动。

注意：*RxWait* 寄存器定义发送最后一位到激活接收电路的时间，可用来调整发送和接收之间的时序关系，此外，*BitPhase* 寄存器定义发送电路和接收电路时钟之间的相移。

16.4.3.1 与卡通讯的状态

发送器和接收器的实际状态可从通过 *PrimaryStatus-Register* 的 *ModemState* 得知。
下列是 *ModemState* 内部状态分配如下表：

ModemState	状态名	描述
000	Idle	由于发送器和接收器被启动或接收器没有输入数据，所有都不在工作状态。
001	TxSOF	发送 SOF
010	TxData	发送 FIFO 里的数据（或冗余检查位）
011	TxEOF	发送 EOF
100	GotoRx1	当接收器启动，通过中间状态
	GotoRx2	当接收器结束，通过中间状态
101	PrepareRx	等待 *RxWait* 寄存器定义的时间。
110	AwaitingRx	接收器激活；在 Rx 引脚等待输入信号
111	Receiving	接收数据

表 16－3：ModemState 含义

16.4.3.2 与卡通讯的状态机

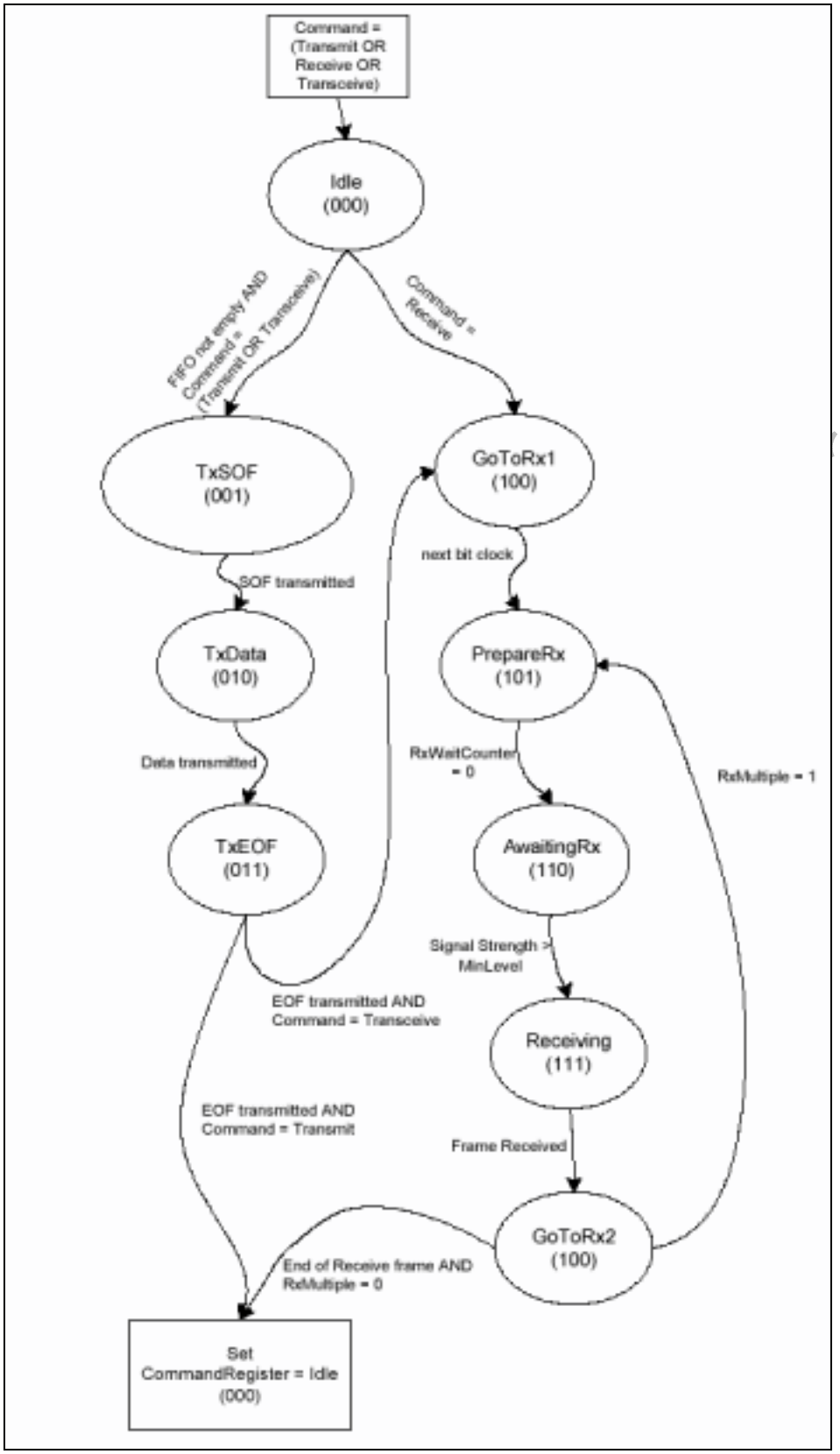


图 16－4：与卡通讯的状态机

16.5 E2PROM 访问命令

16.5.1 WRITEE2 COMMAND 01_{HEX}

16.5.1.1 概述

命令	指令代码 (16 进制)	功能	通过 FIFO 传递的参数和数据	返回的数据
WriteE2	01	从 FIFO 缓存读取数据，并且写入内部 E^2PROM。	起始地址 LSB 起始地址 MSB 数据流	-

WriteE2-Command 将 FIFO 中前两个字节看作 E^2PROM 开始地址，余下的任何数据被看作数据并对 E^2PROM 从给定的开始地址开始编程。该命令不返回数据。它只能由微处理器启动，而且不能自动停止，只能用微处理器插入 *Idle-Command* 中止。

16.5.1.2 编程过程

在一个编程周期内可对 E^2PROM 的一个到 16 个字节进行编程，一个编程周期所需时间为 8ms。

内部状态机将在 FIFO 中准备的数据复制到 E^2PROM 数据缓存，E^2PROM 数据缓存为 16 个字节，正好等于 E^2PROM 块的大小。当 E^2PROM 数据缓存写满或 FIFO 中数据全部读出，启动一个编程周期。

只要在FIFO缓存中有未处理的字节或处于一个编程周期中，标志*E2Ready*为0。如果FIFO中所有数据都编程进E^2PROM，*E2Ready*置为1，同时中断申请标志*TxIRq*也置1，用来产生编程结束的中断。

E2Ready 置为 1，可用用微处理器插入 Idle-Command 中止 WriteE2-Command。

注意：当*E2Ready*为0时，E^2PROM还处在编程过程中，*WriteE2-Command*不可以被任何命令中断。

16.5.1.3 时序图

下图以写 5 个字节为例说明了编程过程：

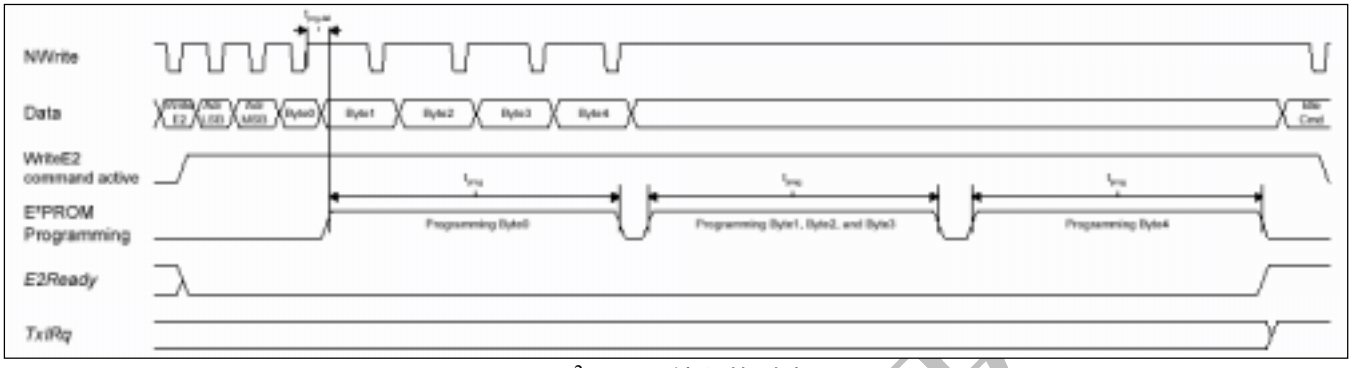


图 16－5：E^2PROM 编程的时序图

解释：假定 FM1702SL 在微处理器写入 Byte 1 之前只读到 Byte 0（$t_{prog,del}$ = 300 ns），FM1702SL 开始了一次 8ms 的编程过程，同时微处理器将 Byte 1 到 Byte 4 写入 FIFO。假定开始地址 $16C_{HEX}$，Byte 0 存放在此；然后 FIFO 中的数据再复制到 E^2PROM 数据缓存，拷贝到 Byte 3 时，FM1702SL 发现此时 E^2PROM 地址为 $16F_{HEX}$，已经到了 E^2PROM 块的最后一字节了。所以下次，Byte 4 被编程至地址 170_{HEX}。最后一个字节编程完后，标志位 *E2Ready* 和 *TxIRq* 置起，表示当前编程过程结束。

虽然所有数据都已写入 E^2PROM，但 *WriteE2-Command* 不会自动退出。如果此时写入新的数据到 FIFO，编程过程就会在地址 171_{HEX} 继续。使用 *Idle-Command* 可以使其退出。

16.5.1.4 WriteE2-Command 相关错误标志

E^2PROM 第 0 块（地址 00_{HEX} ～ $0F_{HEX}$）禁止编程，如果对这些地址编程就会造成错误标志 AccessErr 置起，也不会有编程过程启动。若地址超过 $1FF_{HEX}$，则将其和 $1FF_{HEX}$ 相与后作为实际的操作地址（要了解 E^2PROM 组织结构，参见第 6 章）。

16.5.2 READE2 COMMAND 03_{HEX}

16.5.2.1 概述

命令	指令代码 (16 进制)	功能	通过 FIFO 传递的参数和数据	返回的数据
ReadE2	03	从内部 E^2PROM 读取数据，并且写入 FIFO 缓存。	起始地址 LSB 起始地址 MSB 字节数	数据流

ReadE2-Command 将 FIFO 中前两字节作为 E^2PROM 读操作的起始地址。下一个字节指定要读出的字节数。
当在 FIFO 中有以上三个参数后，从起始地址开始指定字节数从 E^2PROM 读出到 FIFO 缓存。
ReadE2-Command 只能由微处理器启动，当指定数据读出后自动中止执行。

16.5.2.2 READE2-Command 相关错误标志

禁止读 E^2PROM 地址 08_{HEX} ～ $1F_{HEX}$（密钥 key 保存区），如果读这些地址，就会造成错误标志 AccessErr 置起。。若地址超过 $1FF_{HEX}$，则将其和 $1FF_{HEX}$ 相与后作为实际的操作地址（要了解 E^2PROM 组织结构，参见第 6 章）。

16.6 其他命令

16.6.1 LOADCONFIG COMMAND 07_{HEX}

16.6.1.1 概述

命令	指令代码 (16 进制)	功能	通过 FIFO 传递的参数和数据	返回的数据
LoadConfig	07	从 E^2PROM 读取数据，用于初始化寄存器。	起始地址 LSB 起始地址 MSB	-

LoadConfig-Command 将 FIFO 中前两个字节作为 E^2PROM 读操作的起始地址，当 FIFO 中的两个参数有效时，从 E^2PROM 该起始地址开始的 32 字节会复制到 FM1702SL 控制和配置寄存器内。*LoadConfig -Command* 只能由微处理器启动，当所有相关寄存器配置后自动中止执行。

16.6.1.2 寄存器分配

从 E^2PROM 该起始地址开始的 32 字节会复制到 FM1702SL 地址为 10_{HEX} ～ $2F_{HEX}$ 的寄存器内（要了解 E^2PROM 组织结构，参见第 6 章）。

注意：寄存器分配的过程和芯片初始化的过程是相同的（见 11.3 章）。不同点在于，芯片初始化时 E^2PROM 起始地址固定为 10_{HEX}（第 1 块，Byte 0），而 *LoadConfig-Command* 可以选择起始地址。

16.6.1.3 LOADCONFIG-Command 相关错误标志

合法的 E^2PROM 起始地址为 10_{HEX} ～ 60_{HEX}。

禁止复制 E^2PROM 块 08_{HEX} ～ $1F_{HEX}$（密钥 key 保存区）的内容，如果读这些地址，就会造成错误标志 AccessErr 置起。若地址超过 $1FF_{HEX}$，则将其和 $1FF_{HEX}$ 相与后作为实际的操作地址（要了解 E^2PROM 组织结构，参见第 6 章）。

16.6.2 CALCCRC COMMAND 12_{HEX}

16.6.2.1 概述

命令	指令代码 (16 进制)	功能	通过 FIFO 传递的参数和数据	返回的数据
CalcCRC	12	激活 CRC 协处理器。	数据流	-

CalcCRC-Command 把 FIFO 中所有数据作为 CRC 协处理器输入字节，所有在命令启动前存放在 FIFO 中的数据都会被处理。该命令不返回通过 FIFO 任何数据，CRC 寄存器的内容可从 *CRCResultLSB* 和 *CRCResultMSB* 寄存器中读到。它只能由微处理器启动，而且不能自动停止，只能用微处理器插入 *Idle-Command* 中止。如果 FIFO 中没有数据，在停止前 *CalcCRC-Command* 会一直等待有新的数据写入 FIFO 作为输入。

16.6.2.2 CRC 协处理器设置

使用 CRC 协处理器前，需配置下列参数：

参数	值	位	寄存器
CRC 寄存器长度	8 位或 16 位 CRC	CRC8	ChannelRedundancy
CRC 算法	符合 ISO14443-A 或 ISO/IEC3309 的算法	CRC3309	ChannelRedundancy
CRC 复位值	任意	CRCPresetLSB CRCPresetMSB	CRCPresetLSB CRCPresetMSB

表 16－4 ：CRC 协处理器相关参数

8 位 CRC 多项式为：$x^8 + x^4 + x^3 + x^2 + 1$。
16 位 CRC 多项式为：$x^{16} + x^{12} + x^5 + 1$。

16.6.2.3 CRC 协处理器的状态标志

状态标志 *CRCReady* 表示 CRC 协处理器已处理完所有 FIFO 中的数据。当 *CRCReady* 置起，*TxIRq* 会置起产生中断申请，支持 CRC 协处理器的中断驱动的使用方法。

当 *CRCReady* 和 *TxIRq* 置为 1，*CRCResultLSB*、C*RCResultMSB* 寄存器和标志 *CRCErr* 分别有效。

CRCResultLSB、C*RCResultMSB* 寄存器保存 CRC 寄存器的内容，标志 *CRCErr* 指示处理数据的正确性。

16.7 命令执行过程中的错误处理

如果命令执行中产生错误，通过将 *PrimaryStatus* 寄存器内的状态标志 *Err* 置起来表示，微处理器通过读 *ErrorFlag* 中的状态标志可以知道错误产生的原因。

ErrorFlag 寄存器中的错误标志	相关命令
KeyErr	*LoadKeyE2, LoadKey*
AccessError	*WriteE2, ReadE2, LoadConfig*
FIFOOvl	没有特定的命令
CRCErr	*Receive，Transceive，CalcCRC*
FramingErr	*Receive，Transceive*
ParityErr	*Receive，Transceive*
CollErr	*Receive，Transceive*

表 16－5：错误标志纵览

16.8 安全命令

16.8.1 LOADKEYE2 COMMAND $0B_{HEX}$

16.8.1.1 概述

命令	指令代码 (16 进制)	功能	通过 FIFO 传递的参数和数据	返回的数据
LoadKeyE2	0B	将密钥从 E^2PROM 复制到 key 缓存。	起始地址 LSB 起始地址 MSB	-

LoadKeyE2-Command 将 FIFO 中前两字节作为 E^2PROM 读操作的起始地址。E^2PROM 起始地址开始的字节作为密钥，需符合 6.4.1 规定的正确密钥格式。当 FIFO 中有两个字节数据后，该命令则启动。*LoadKeyE2-Command* 只能由微处理器启动，当整个密钥复制从 E^2PROM 到密钥缓存区后自动中止执行。

16.8.1.2 LOADKEYE2-Command 相关错误标志

如果密钥的格式不正确（见 6.4.1），密钥缓存区保存的是一个未定义的值，同时 *KeyError* 置起。

16.8.2 LOADKEY COMMAND 19_{HEX}

16.8.2.1 概述

命令	指令代码 (16 进制)	功能	通过 FIFO 传递的参数和数据	返回的数据
LoadKey	19	将密钥从 FIFO 缓存复制到 key 缓存。	Byte0(LSB) Byte1 … Byte10 Byte11(MSB)	-

LoadKey-Command 将 FIFO 中前十二个字节作为密钥，需符合 6.4.1 规定的正确密钥格式。

当 FIFO 中有十二个字节后，如果检查后格式合法则被拷贝到密钥缓存区。（见 17.2）。

LoadKey-Command 只能由微处理器启动，当整个密钥复制从 FIFO 到密钥缓存区后自动中止执行。

16.2.2.2 LOADKEY-Command 相关错误标志

所有所需的数据被从 FIFO 复制到密钥缓存区后，如果密钥的格式不正确（见 6.4.1 章），密钥缓存区保存的是一个未定义的值，同时 *KeyError* 置起。

16.8.3 AUTHENT1 COMMAND $0C_{HEX}$

16.8.3.1 概述

命令	指令代码 (16 进制)	功能	通过 FIFO 传递的参数和数据	返回的数据
Authent1	0C	执行 Crypto1 算法的认证过程的第一步。	卡认证命令 卡的块地址 卡序列号的 LSB 卡序列号的 Byte1 卡序列号的 Byte2 卡序列号的 MSB	-

Authent1-Command 是一条特殊的 *Transceive-Command*：它需要 6 个参数，前两个参数被发送到卡，卡的回发不传递给微处理器，而用作检验卡的真实性和证明 FM1702SL 对卡的真实性。

Authent1-Command 只能由微处理器启动，该命令状态转换同 *Transceive-Command*（见 16.4.3 章）。

16.8.4 AUTHENT2 COMMAND 14_{HEX}

16.8.4.1 概述

命令	指令代码 (16 进制)	功能	通过 FIFO 传递的参数和数据	返回的数据
Authent2	14	执行 Crypto1 算法的认证过程的第二步。	-	-

Authen2-Command 是一条特殊的 *Transceive-Command*。无需任何参数，FM1702SL 内部产生需发送给卡的数据。卡的回发不传递给微处理器，而用作检验卡的真实性和证明 FM1702SL 对卡的真实性。

Authent2-Command 只能由微处理器启动，该命令状态转换同 *Transceive-Command*（见 16.4.3 章）。

16.8.4.2 AUTHENT2-Command 的作用

如果 *Authen2-Command* 通过，卡和 FM1702SL 的认证都通过，于是控制位 *Crypto1On* 自动置位。该位置起后，所有与卡的通讯都使用 Crypto1 算法加密后进行。如果 *Authen2-Command* 失败，*Crypto1On* 自动清零。

注：标志 *Crypto1On* 不能用软件置位，只有通过 *Authen2-Command* 后硬件置位才行。软件可以清该位使以后的通讯恢复成明文。

注意：*Authen2-Command* 必须在 *Authent1-Command* 成功通过后执行（见 16.8.3 章）,此外，密钥缓存区的密钥必须和卡上的相同。

17 认证及数据加密传输

17.1 概述

FM1702SL 使用的认证算法称为三重认证。它基于密钥长度为 48 比特的私有加密数据流。如欲获取标准卡片的数据，有关相应密要的知识是必需的。为了能够成功进行卡的认证以及后续对储存于卡 EEPROM 中的数据进行操作，FM1702SL 必须能够获得正确的密钥。当一张卡按照 ISO14443A 协议被选中后，用户可以按照标准协议继续操作。这种情况下，必须执行卡片认证。这一过程在执行 Authent1（参考 16.8.3）和 Authent2（参考 16.8.4）指令时自动完成。在卡认证的过程中，加密算法被初始化，在成功认证之后与卡的通讯处于加密状态。

17.2 密钥处理

在认证指令执行过程中，FM1702SL 从内部密钥缓冲器中读取密钥。密钥总是从密钥缓冲器中获取。因此认证指令无需指明密钥存储地址。当然，在认证指令开始之前，用户必须保证在密钥缓冲器中已经准备好了密钥。

密钥缓冲器可以通过一下方式加载：

- 用LoadKeyE2指令从E²PROM中加载
- 直接由外部处理器通过LoadKey指令从FIFO中加载

17.3 操作三重认证指令

三重加密算法被用于执行标准认证。在密钥缓冲器中必须储存准确的密钥以便能够进行成功的认证操作。

步骤 1： 通过 LoadKeyE2（参见 16.8.1）或者 LoadKey（参见 16.8.2）加载密钥到内部密钥缓冲器；

步骤 2： 启动 Authent1 指令（参见 16.8.3），结束以后，检查错误标志来判断执行结果；

步骤 3： 启动 Authent2 指令（参见 16.8.4），结束以后，检查错误标志以及 Crypto1On 标志来判断执行结果。

17.4 认证算法

FM1702SL 分别支持 MIFARE 三重认证算法

18 典型应用

18.1 电路图

下面是典型应用电路图，天线直接连接到 FM1702SL

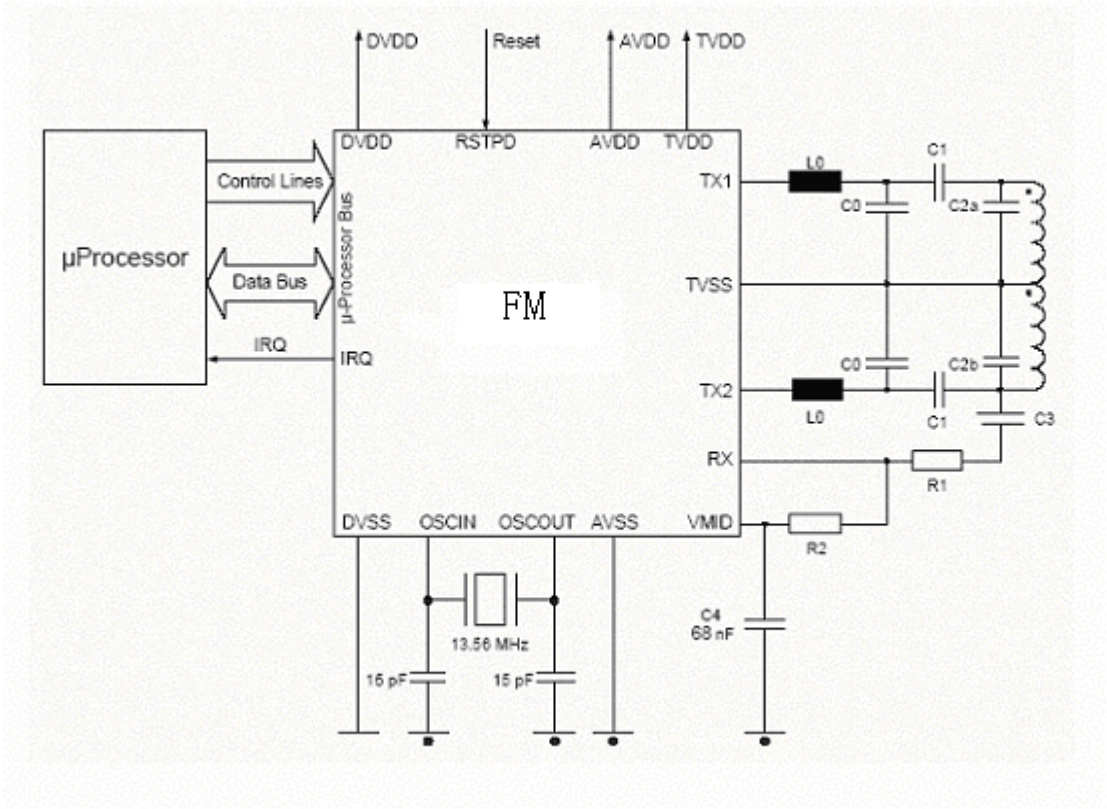


图 18－1：应用电路示例：直接天线连接

匹配电路包含一个 EMC 低通滤波器，一个接收电路，天线匹配电路以及天线

18.2 电路描述

18.2.1 EMC 低通滤波器

FM1702SL 系统工作于 13.56MHz 频率下,这一频率产生于一个石英晶体振荡器，用于驱动 FM1702SL 并且提供给天线 13.56MHz 的载波。但是除了 13.56MHz 以外，还会有能量以高次谐波的方式往外发射。国际 EMC 规则规定了在一个宽频范围内发射能量的大小，因此，必须要有一个合适的滤波器过滤输出信号以满足此规定。

我们强烈推荐使用多层板来实现上述电路的滤波器。低通滤波器由 L0 和 C0 组成，它们的值由下表给出。

注意：为了得到最好的效果，所有器件的质量至少要达到下表所推荐的范围。

18.2.2 接收电路

FM1702SL 的内部接收电路利用卡的回应信号在副载波的双边带上都有调制这一概念来进行工作。我们推荐时用芯片内部产生的 VMID 作为 RX 管脚输入信号的偏置。为了稳定 VMID 的输出，必须在 VMID 和 GND 之间连接一个电容 C4。接收电路需要在 RX 和 VMID 之间连接一个分压电路。上面的电路图展示了一个推荐的接收电路。这个接收电路由 R1、R2、C3 和 C4 组成。其值由下表给出。

器件	值	备注
L0	1.0uH±10%	如 TDK 1R0J
C0	136pF±2%	NP0 材料
R1	1kΩ±5%	
R2	820Ω±5%	
C3	68nF±2%	NP0 材料

表 18－1：EMC 滤波器和接收电路元件值

注意：不推荐适用 X7R 材料的电容

18.3 计算天线线圈的电感

精确计算天线线圈的电感值在实践上石非常困难的，但是可以用下面的公式估算。我们建议天线采用圆环状或者矩形设计。

$$L_1\,[nH] = 2 \cdot l_1[cm] \cdot \left(\ln\left(\frac{l_1}{D_1}\right) - K \right) N_1^{1.8}$$

l_1 …… 一圈导线的长度

D_1 …… 导线直径或者 PCB 板敷铜的宽度

K …… 天线形状系数（圆环状：K=1.07；矩形：K=1.47）

ln …… 自然对数

18.3.1 直接连接天线的阻抗匹配

我们推荐使用图 20-1 所示的电路设计天线直接连接的匹配电路。电容 C1 和 C2a、C2b 的取值依赖于天线的电性能以及环境影响。

下表所示电容值只是一个参考。事实上，它们只是用于调试时的起始值。

天线线圈电感 [uH]	C1 [pF]	C2a [pF]	C2b [pF]
0.8	15	270	330
0.9	15	270	270
1.0	15	220	270
1.1	15	180//22	220
1.2	15	180	180//22
1.3	15	180	180
1.4	15	150	180
1.5	15	150	150
1.6	15	120//10	150
1.7	15	120	150
1.8	15	120	120

表 18－2：匹配电路的电容取值

然而，为了优化性能，准确地取值必须通过调试来获得。

上面的表格假设天线线圈的分布电容有 15pF。电容 C1、C2a 和 C2b 应该采用 NP0 电介质而且误差在±2%之内。

实际的天线电感和电容值取决于很多情况比如：

- 天线的电阻（PCB 类型）
- 导体的厚度
- 线与线之间的距离
- 保护层材料
- 附近的金属或者铁氧体

19 电性能

19.1 极限参数

符号	参数	MIN	MAX	单位
$T_{amb,abs}$	存储温度	-40	+150	°C
DVDD AVDD TVDD	直流供电电压	-0.5	6	V
$V_{in,abs}$	所有数字管脚对 DVSS 绝对电压	-0.5	DVDD + 0.5	V
$V_{RX,abs}$	RX 管脚对 AVSS 绝对电压	-0.5	AVDD + 0.5	V

表 19－1：极限参数

19.2 工作条件

工作电压必须满足：DVDD<=AVDD<=TVDD

符号	参数	条件	MIN	TYP	MAX	单位
T_{amb}	环境温度	-	-25	+25	+85	°C
DVDD	数字电路供电电压	DVSS = AVSS = TVSS = 0V	3.0	3.3	3.6	V
			4.5	5.0	5.5	V
AVDD	模拟电路供电电压	DVSS = AVSS = TVSS = 0V	3.0	3.3	3.6	V
TVDD	发射电路供电电压	DVSS = AVSS = TVSS = 0V	3.0	5.0	5.5	V

表 19－2：工作条件

19.3 工作电流

符号	参数	条件	MIN	TYP	MAX	单位
IDVDD	数字电路工作电流	Idle 指令		6	9	mA
		Stand By 模式		3	5	mA
		Soft Power Down 模式		800	1000	μA
		Hard Power Down 模式		1	10	μA
IAVDD	模拟电路工作电流	Idle 指令, 接收器打开		25	40	mA
		Idle 指令, 接收器关闭		8	12	mA
		Stand By 模式		6.5	9	mA
		Soft Power Down 模式		1	10	μA
		Hard Power Down 模式		1	10	μA
ITVDD	发射电路工作电流	发射连续载波			150	mA
		TX1 和 TX2 悬空 *TX1RFEn*, *TX2RFEn*= 1		4.5	6	mA
		TX1 和 TX2 悬空 *TX1RFEn*, *TX2RFEn*= 0		65	130	μA

表 19－3：工作电流

19.4 管脚特性

19.4.1 输入管脚特性

管脚 MISO，MOSI 有 TTL 输入特性，如下表所示

符号	参数	条件	MIN	MAX	单位
I_{Leak}	输入漏电流		-1.0	+1.0	μA
V_T	阀值	CMOS: DVDD < 3.6 V	0.35 DVDD	0.65 DVDD	V
		TTL: 4.5 < DVDD	0.8	2.0	V

表 19－4：典型输入管脚特性

管脚 NSS，SCK 和 MFIN 具有 Schmitt 触发器特性，如下表所示

符号	参数	条件	MIN	MAX	单位
I_{Leak}	输入漏电流		-1.0	+1.0	μA
V_{T+}	正向阀值	TTL: 4.5 < DVDD	1.4	2.0	V
		CMOS: DVDD < 3.6 V	0.65 DVDD	0.75 DVDD	V
V_{T-}	负向阀值	TTL: 4.5 < DVDD	0.8	1.3	V
		CMOS: DVDD < 3.6 V	0.25 DVDD	0.4 DVDD	V

表 19－5：具有 Schmitt 触发器的输入管脚特性

RSTPD 具有 CMOS Schmitt 特性。此外，它内部有一个 RC 低通滤波器，因此输入的复位信号有相应延迟。

符号	参数	条件	MIN	MAX	单位
I_{Leak}	输入漏电流		-1.0	+1.0	μA
V_{T+}	正向阀值	CMOS: DVDD < 3.6 V	0.65 DVDD	0.75 DVDD	V
V_{T-}	负向阀值	CMOS: DVDD < 3.6 V	0.25 DVDD	0.4 DVDD	V
$t_{RSTPD,p}$	传输延迟			20	μs

表 19－6：RSTPD 输入特性

模拟输入管脚 RX 具有如下的电容特性

符号	参数	条件	MIN	MAX	单位
C_{RX}	输入电容			15	pF

表 19－7：RX 输入电容

20.4.2 数字输出管脚特性

管脚 MISO 和 IRQ 具有 TTL 输出特性，如下表所示

符号	参数	CONDITIONS	MIN	TYP	MAX	单位
V_{OH}	输出高电平	DVDD = 5 V, IOH= -1 mA	2.4	4.9		V
		DVDD = 5 V, IOH= -10 mA	2.4	4.2		V
V_{OL}	输出低电平	DVDD = 5 V, IOL= 1 mA		25	400	mV
		DVDD = 5 V, IOL= 10 mA		250	400	mV
I_O	输出电流或灌电流	DVDD = 5 V			10	mA

表 19－8：数字输出管脚特性

20.4.3 天线驱动管脚输出特性

天线驱动管脚 TX1 和 TX2 的输出电阻在输出高电平时可以通过配置 CwConductance 寄存器的 GsCfgCW 来更改，而在输出低电平时，输出阻抗是固定的。

在默认配置下，输出特性如下表所示

符号	参数	CONDITIONS	MIN	TYP	MAX	单位
V_{OH}	输出高电平	TVDD = 5.0 V, IOL = 20 mA		4.97		V
		TVDD = 5.0 V, IOL = 100 mA		4.85		V
V_{OL}	输出低电平	TVDD = 5.0 V, IOL = 20 mA		30		mV
		TVDD = 5.0 V, IOL = 100 mA		150		mV
I_{TX}	输出电流	连续载波			200	mA_{peak}

表 20－9：天线驱动管脚输出特性

19.5 交流电性能

19.5.1 SPI 接口交流工作说明

符号	参数	MIN	MAX	单位
t_{SCKL}	SCK 低电平宽度	100		ns
t_{SCKH}	SCK 高电平宽度	100		ns
t_{SHDX}	SCK 高到数据改变	20		ns
t_{DXSH}	d 数据改变到 SCK 变高	20		ns
t_{SLDX}	SCK 低到数据改变		15	ns
t_{SLNH}	SCK 低到 NSS 变高	20		ns

表 19－10：SPI 时序说明

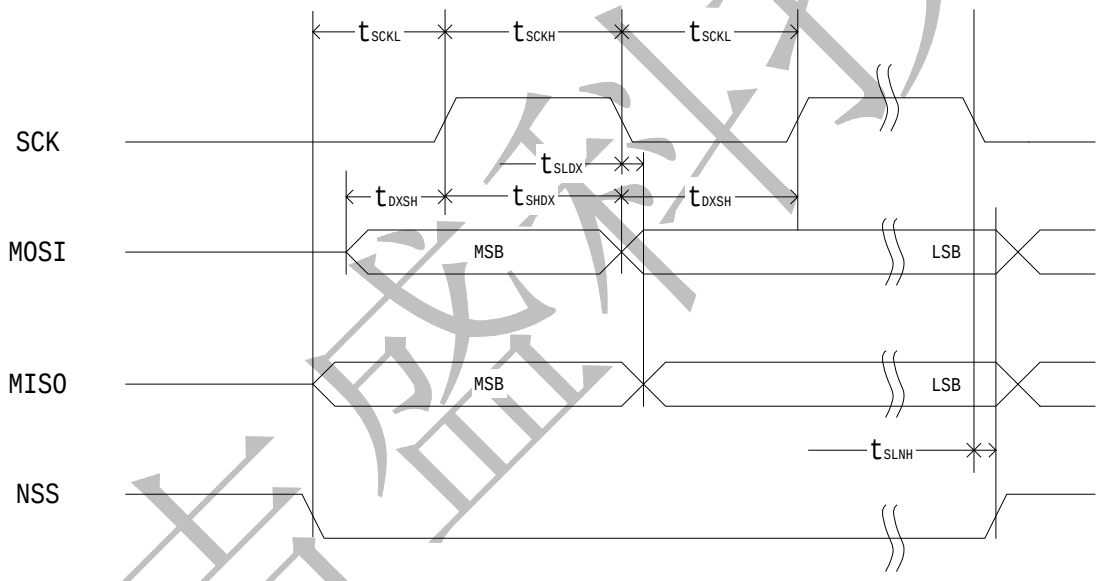


图 19－4：SPI 时序图

说明：如果要在一个数据流里传输更多 byte 的数据，NSS 必须始终为低。在传输多于一个的数据流时，在数据流之间 NSS 必须为高。

19.5.2 时钟频率

1 号管脚为时钟输入管脚，OSCIN

参数	符号	MIN	TYP	MAX	单位
时钟频率 (由时钟滤波器检测)	f_{OSCIN}		13.56		MHz
占空比	d_{FEC}	40	50	60	%
时钟沿波动	t_{jitter}			10	ps

21 E²PROM 特性

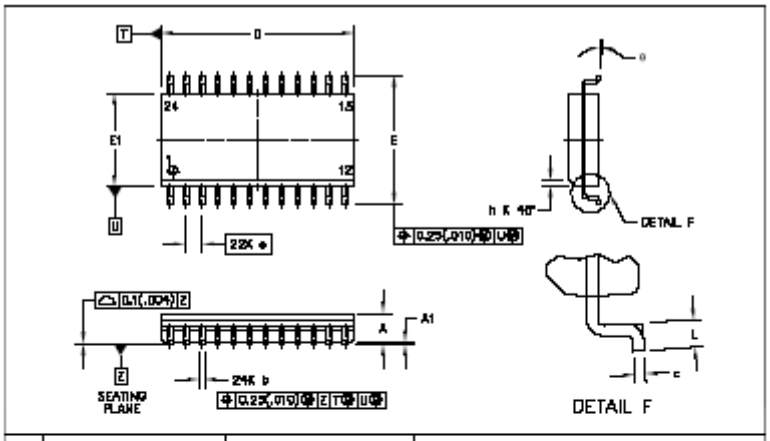
E²PROM 大小为 32x16x8 = 4096 bit.

符号	参数	条件	MIN	MAX	单位
$t_{EEEndurance}$	数据擦写次数		100.000		擦/写次数
$t_{EERetention}$	数据保存时间	Tamb ≤55°C	10		年
$t_{EEErase}$	擦时间			4	ms
$t_{EEWrite}$	写时间			4	ms

表 21－1：E²PROM 特性

22 封装

芯片采用 SOP24 封装



DIM	MILLIMETERS MIN	MAX	INCHES MIN	MAX	NOTES
A	2.35	2.65	.093	.104	1. CONTROLLING DIMENSION: MILLIMETER.
A1	0.13	0.29	.005	.011	2. INTERPRET DIMENSIONS AND TOLERANCES PER ASME Y14.5M–1994.
D	15.25	15.54	.601	.612	
E	10.05	10.55	.396	.416	3. DIMENSION D AND E1 DO NOT INCLUDE MOLD PROTRUSION.
E1	7.4	7.6	.292	.299	
b	0.35	0.49	.014	.019	4. MAXIMUM MOLD PROTRUSION 0.15 (.006) PER SIDE.
e	0.23	0.32	.009	.013	
e	1.27 BSC		.050 BSC		5. DIMENSION b DOES NOT INCLUDE DAM BAR PROTRUSION. ALLOWABLE DAM BAR PROTRUSION SHALL BE 0.13 (.005) TOTAL IN EXCESS OF THE b DIMENSION AT MAXIMUM MATERIAL CONDITION.
L	0.41	0.9	.016	.035	
h	0.25	0.75	.010	.029	
θ	0°	8°	0°	8°	

TITLE: SOIC 24 LD WIDE BODY 300 MIL PKG 1.27 PITCH POD	COMPANY	ASECL
	SHEET	1 OF 1

图 23－1：SOP24 封装