MF RC500-高集成 ISO14443A 读卡芯片

1 通用信息

1.1 范围

该文档讲述了 MF RC500 的功能。包括功能及电气规格，并给出了如何从系统和硬件的角度使用该芯片进行设计的细节。

1.2 概述

MF RC500 是应用于 13.56MHz 非接触式通信中高集成读卡 IC 系列中的一员。该读卡 IC 系列利用了先进的调制和解调概念，完全集成了在 13.56MHz 下所有类型的被动非接触式通信方式和协议。

MF RC500 支持 ISO14443A 所有的层。

内部的发送器部分不需要增加有源电路就能够直接驱动近操作距离的天线（可达 100mm）。

接收器部分提供一个坚固而有效的解调和解码电路，用于 ISO14443A 兼容的应答器信号。

数字部分处理 ISO14443A 帧和错误检测（奇偶&CRC）。此外，它还支持快速 CRYPTO1 加密算法用于验证 MIFARE 系列产品。

方便的并行接口可直接连接到任何 8 位微处理器，这样给读卡器/终端的设计提供了极大的灵活性。

1.3 特性

- 高集成度模拟电路用于卡应答的解调和解码。
- 缓冲输出驱动器使用最少数目的外部元件连接到天线
- 近距离操作(可达 100mm)
- 支持 MIFARE 双接口卡 IC 和 ISO14443A1－4 部分
- 加密并保护内部非易失性密匙存储器
- 并行微处理器接口带有内部地址锁存和 IRQ 线
- 灵活的中断处理
- 自动检测微处理器并行接口类型
- 方便的 64 字节发送和接收 FIFO 缓冲区
- 带低功耗的硬件复位
- 软件实现掉电模式
- 可编程定时器
- 唯一的序列号
- 用户可编程的启动配置
- 位和字节定位帧
- 数字、模拟和发送器部分各自独立的电源输入脚
- 内部振荡器缓冲连接 13.56MHz 石英晶体，低相位抖动
- 时钟频率滤波
- 短距离应用中发送器（天线驱动器）为 3.3V 操作。

2 方框图

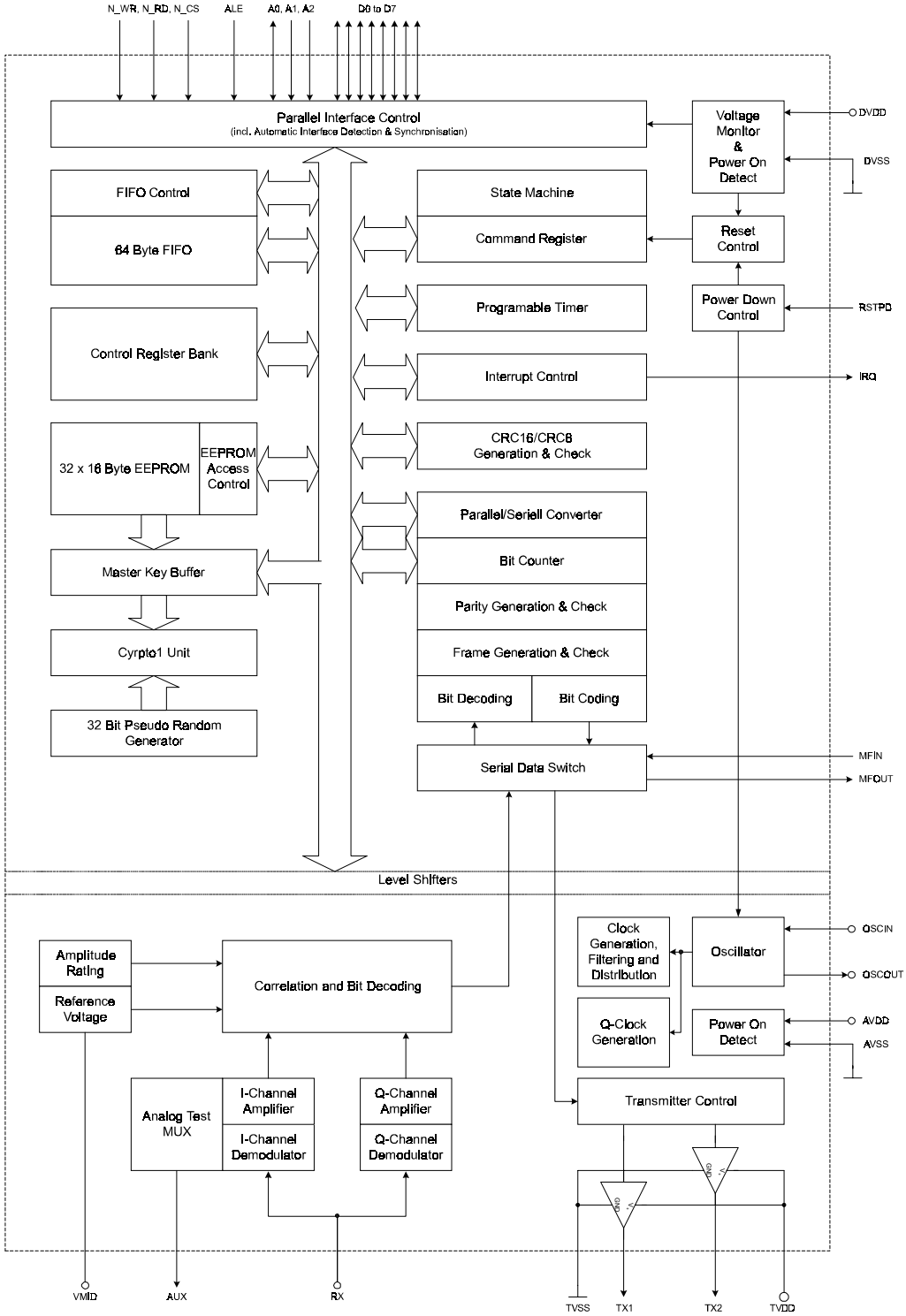


图 2-1 MF RC500 方框图

3 管脚信息

3.1 管脚配置

下图所示用黑体字母标注的管脚由 AVDD 和 AVSS 供电。黑线所标的管脚由 TVSS 和 TVDD 供电。其它管脚由 DVDD 和 DVSS 供电。

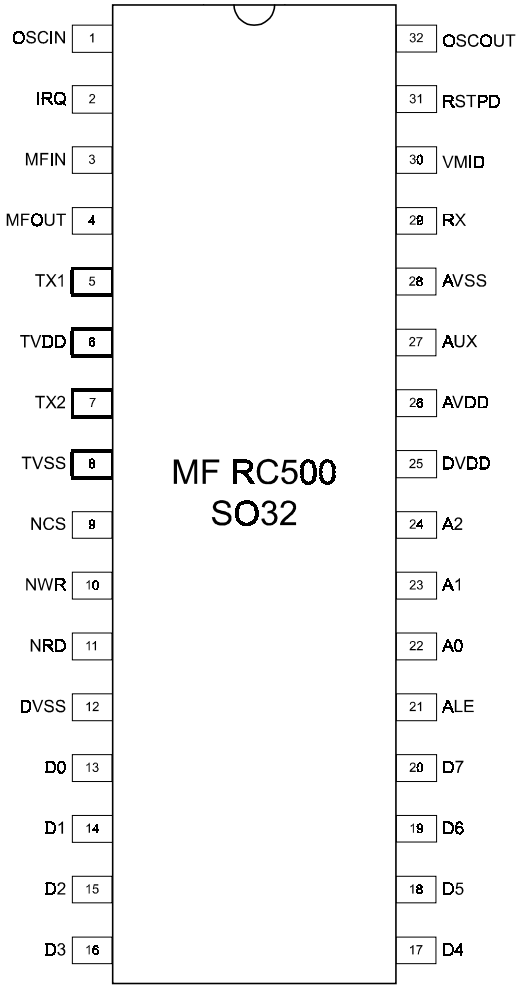


图 3-1　MF RC500 SO32 封装管脚配置

3.2　管脚描述

管脚类型：I：输入　O：输出　PWR：电源

管脚	符号	类型	描述
1	OSCIN	I	晶振输入：振荡器反相放大器输入。该脚也作为外部时钟输入(fosc=13.56MHz)
2	IRQ	O	中断请求：输出中断事件请求信号
3	MFIN	I	MIFARE 接口输入：接受符合 ISO14443A(MIFIRE)的数字串行数据流
4	MFOUT	O	MIFARE 接口输出：发送符合 ISO14443A(MIFIRE)的数字串行数据流
5	TX1	O	发送器 1：发送经过调制的 13.56MHz 能量载波
6	TVDD	PWR	发送器电源：提供 TX1 和 TX2 输出电源
7	TX2	O	发送器 2：发送经过调制的 13.56MHz 能量载波
8	TVSS	PWR	发送器地：提供 TX1 和 TX2 输出电源
9	NCS	I	/片选：选择和激活 MF RC500 的微处理器接口
10	NWR	I	/写：MF RC500 寄存器写入数据（D0~D7）选通
	R/NW	I	读//写：选择所要执行的是读还是写
	nWrite	I	/写：选择所要执行的是读还是写
11	NRD	I	/读：MF RC500 寄存器读出数据（D0~D7）选通
	NDS	I	/数据选通：读和写周期的选通
	nDStrb	I	/数据选通：读和写周期的选通
12	DVSS	PWR	数字地

管脚	符号	类型	描述
13	D0~D7	I/O	8 位双向数据总线
20	AD0~AD7	I/O	8 位双向地址和数据总线
21	ALE	I	地址锁存使能：为高时将 AD0~AD5 锁存为内部地址
	AS	I	地址选通：为低时选通信号将 AD0~AD5 锁存为内部地址
	nAStrb	I	/地址选通：为低时选通信号将 AD0~AD5 锁存为内部地址
22	A0	I	地址线 0：寄存器地址位 0
	nWait	O	/等待：信号为低可以开始一个存取周期，为高时可以停止
23	A1	I	地址线 1：寄存器地址位 1
24	A2	I	地址线 2：寄存器地址位 2
25	DVDD	PWR	数字电源
26	AVDD	PWR	模拟电源
27	AUX	O	辅助输出：该脚输出模拟测试信号。该信号可通过 TestAnaOutSel 寄存器选择
28	AVSS	PWR	模拟地
29	RX	I	接收器输入：卡应答输入脚，该应答为经过天线电路耦合的调制 13.56MHz 载波
30	VMID	PWR	内部参考电压：该脚输出内部参考电压。注:必须接一个 100nF 电容。
31	RSTPD	I	复位和掉电：当为高时，内部灌电流关闭，振荡器停止，输入端与外部断开，该管脚的下降沿启动内部复位。
32	OSCOUT	O	晶振输出：振荡器反向放大器输出

表 3-1　MF RC500 管脚描述

4　并行接口

4.1　所支持的微处理器接口概述

MF RC500 支持与不同的微处理器直接接口，可与个人电脑的增强型并口（EPP）直接相连。

下表所示为 MF RC500 所支持的并口信号

总线控制信号	总线	独立的地址和数据总线	复用的地址和数据总线
独立的读和写选通信号	控制	NRD，NWR，NCS	NRD，NWR，NCS，ALE
	地址	A0,A1,A2	AD0,AD1,AD2,AD3,AD4,AD5
	数据	D0 … D7	AD0 … AD7
共用的读和写选通信号	控制	R/NW,NDS,NCS	R/NW,NDS,NCS,AS
	地址	A0,A1,A2	AD0,AD1,AD2,AD3,AD4,AD5
	数据	D0 … D7	AD0 … AD7
带握手的共用读和写选通信号（EPP）	控制	—	nWrite,nDStrb,nAStrb,nWait
	地址		AD0,AD1,AD2,AD3,AD4,AD5
	数据		AD0 … AD7

表 4-1　所支持的并口信号

4.2　微处理器接口类型自动检测

在每次上电或硬复位后， MF RC500 也复位其并行微处理器接口模式并检测当前微处理器接口的类型。

MF RC500 在复位阶段后根据控制脚的逻辑电平识别微处理器接口。这是由固定管脚连接的组合（见下表）和一个专门的初始化程序实现的（见 11.4）

4.3　与不同微处理器类型的连接

如下表所示：

MF RC500	并行接口类型				
	独立读/写选通		共用读/写选通		
	专用地址总线	复用地址总线	专用地址总线	复用地址总线	带握手的复用地址总线
ALE	HIGH	ALE	HIGH	AS	nAStrb
A2	A2	LOW	A2	LOW	HIGH
A1	A1	HIGH	A1	HIGH	HIGH
A0	A0	HIGH	A0	LOW	nWait
NRD	NRD	NRD	NDS	NDS	NDStrb
NWR	NWR	NWR	R/NW	R/NW	NWRite
NCS	NCS	NCS	NCS	NCS	LOW
D7 … D0	D7 … D0	AD7 … AD0	D7 … D0	AD7 … AD0	AD7 … AD0

表 4-2　检测并行接口类型的连接配置

4.3.1　独立的读/写选通信号

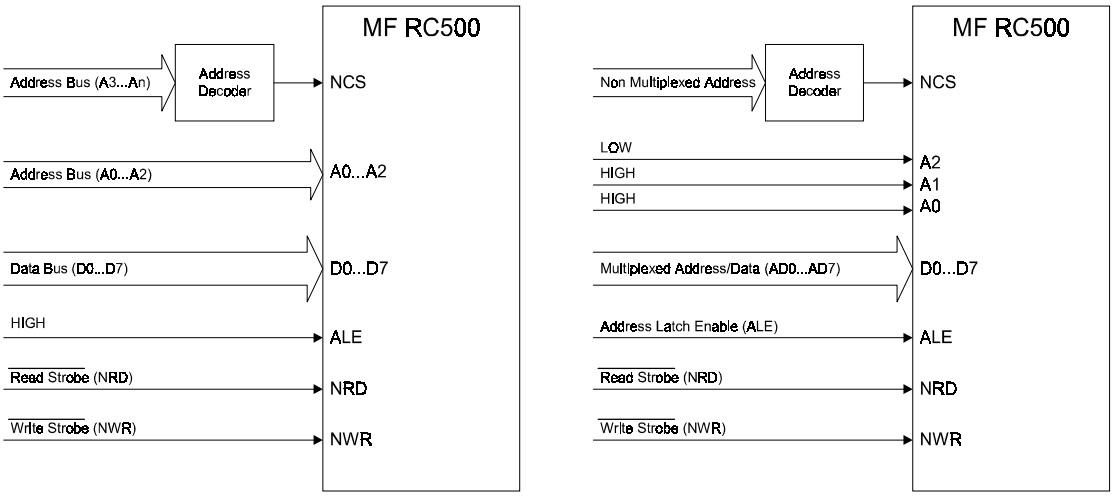


图 4-1　独立的读/写选通连接到微处理器

时序规格参见 20.5.2.1 章节。

4.3.2　共用的读/写选通信号

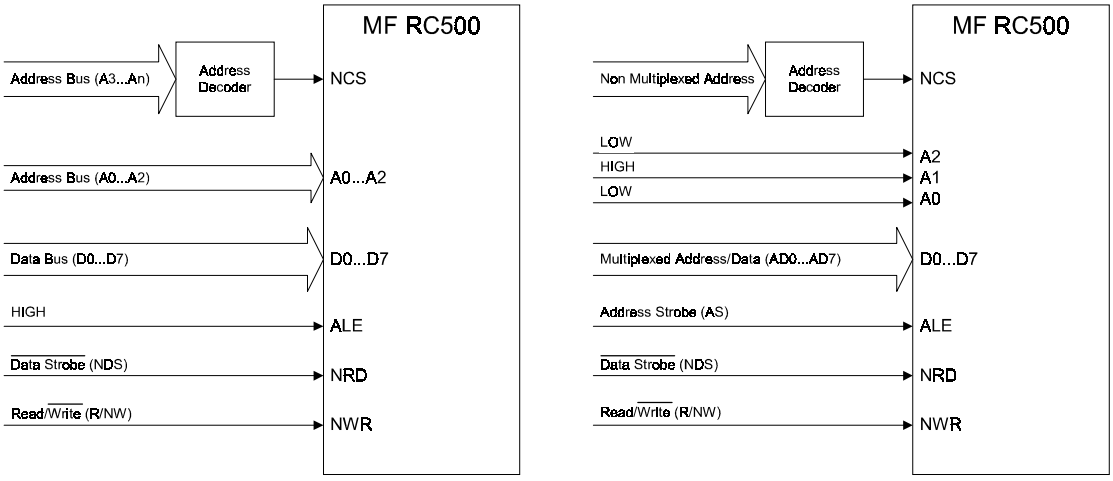


图 4-2　共用的读/写选通连接到微处理器

时序规格参见 20.5.2.2 章节。

4.3.3　带握手机制的共用读/写选通信号：EPP

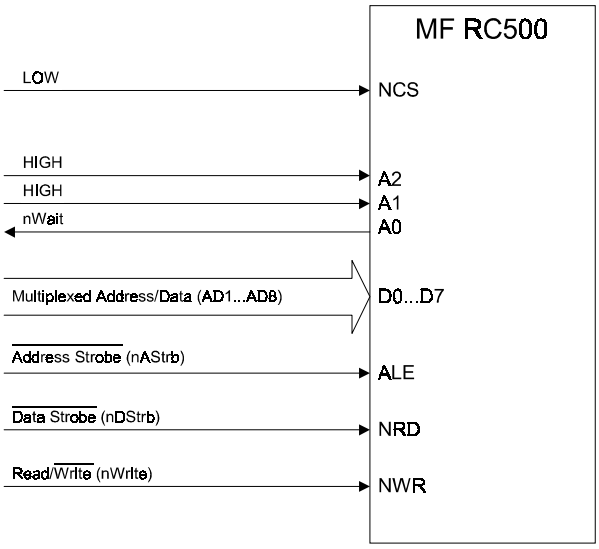


图 4-3　带共用读/写选通和握手机制连接到微处理器

时序规格参见 20.5.2.3 章节。

EPP 备注：

尽管在 EPP 的标准中无片选信号的定义，MF RC500 的 N_CS 允许禁止 nDStrb 信号。如果不用，应将其接到 DVSS。

在每次上电或硬复位后，nWait 信号（由 A0 脚发出）为高阻态。nWait 将在复位后 nAStrb 上的第一个下降沿时定义。

MF RC500 不支持读地址周期

5　MF RC500 寄存器集合

5.1　MF RC500 寄存器概述

页	地址(hex)	寄存器名	功能
页 0:命令和状态	0	Page	选择寄存器页
	1	Command	启动（和停止）命令的执行
	2	FIFOData	64 字节 FIFO 缓冲区输入和输出
	3	PrimaryStatus	接收器和发送器以及 FIFO 缓冲区状态标志
	4	FIFOLength	FIFO 中缓冲的字节数
	5	SecondaryStatus	不同的状态标志
	6	InterruptEn	使能和禁止中断请求通过的控制位
	7	InterruptRq	中断请求标志
页 1:控制和状态	8	Page	选择寄存器页
	9	Control	不同的控制标志：例如：定时器，节电
	A	ErrorFlag	显示上次命令执行错误状态的错误标志
	B	CollPos	RF 接口检测到的第一个冲突位的位置
	C	TimerValue	定时器的实际值
	D	CRCResultLSB	CRC 协处理器寄存器的最低位
	E	CRCResultMSB	CRC 协处理器寄存器的最高位
	F	BitFraming	位方式帧的调节

MF RC500 寄存器集(续)

页	地址(hex)	寄存器名	功能
页 2:发送器和编码器控制	10	Page	选择寄存器页
	11	TxControl	天线驱动脚 TX1 和 TX2 的逻辑状态控制
	12	CWConductance	选择天线驱动脚 TX1 和 TX2 的电导率
	13	PreSet13	该值不会改变
	14	PreSet14	该值不会改变
	15	ModWidth	选择调整脉冲的宽度
	16	PreSet16	该值不会改变
	17	PreSet17	该值不会改变
页 3:接收器和解码控制	18	Page	选择寄存器页
	19	RxControl1	控制接收器状态
	1A	DecodeControl	控制解码器状态
	1B	BitPhase	选择发送器和接收器时钟之间的位相位
	1C	RxThreshold	选择位解码器的阀值
	1D	PreSet1D	该值不会改变
	1E	RxControl2	控制解码器状态和定义接收器的输入源
	1F	ClockQControl	控制时钟产生用于 90°相移的 Q 信道时钟
页 4:时序和信道冗余	20	Page	选择寄存器页
	21	RxWait	选择发送后，接收器启动前的时间间隔
	22	ChannelRedundancy	选择 RF 信道上数据完整性检测的类型和模式
	23	CRCPresetLSB	CRC 寄存器预设值的低字节
	24	CRCPresetMSB	CRC 寄存器预设值的高字节
	25	PreSet25	该值不会改变
	26	MFOUTSelect	选择输出到管脚 MFOUT 的内部信号
	27	PreSet27	该值不会改变
页 5:FIFO,定时器和 IRQ 脚配	28	Page	选择寄存器页
	29	FIFOLevel	定义 FIFO 上溢和下溢警告界限
	2A	TimerClock	选择定时器时钟的分频器
	2B	TimerControl	选择定时器的起始和停止条件
	2C	TimerReload	定义定时器的预装值
	2D	IRQPinConfig	配置 IRQ 脚的输出状态
	2E	PreSet2E	该值不会改变
	2F	PreSet2F	该值不会改变
页 6:RFU	30	Page	选择寄存器页
	31	RFU	保留将来之用
	32	RFU	保留将来之用
	33	RFU	保留将来之用
	34	RFU	保留将来之用
	35	RFU	保留将来之用
	36	RFU	保留将来之用
	37	RFU	保留将来之用

页 7:测试控制	38	Page	选择寄存器页
	39	RFU	保留将来之用
	3A	TestAnaSelect	选择模拟测试模式
	3B	PreSet3B	该值不会改变
	3C	PreSet3C	该值不会改变
	3D	TestDigiSelect	选择数字测试模式
	3E	RFU	保留将来之用
	3F	RFU	保留将来之用

表 5-1　MF RC500 寄存器汇总

5.1.1　寄存器位状态

不同的寄存器的位和标志的状态是不同的，这取决于它们的功能。原则上具有相同状态的位都归类到共用的寄存器中。

缩写	状态	描述
r/w	读和写	这些位可通过微处理器读和写。由于它们仅用于控制方式，因此它们的内容不会被内部状态机影响，例如，TimerReload 寄存器可通过微处理器读写。还会被内部状态机读取，但不会改变它们。
dy	动态	这些位可通过微处理器读和写。但是它们也可被内部状态机自动写入，例如，Command 寄存器在执行完实际的命令后自动改变它的值。
r	只读	这些寄存器保存标志，其值仅由内部状态决定，例如，ErrorFlag 寄存器显示内部状态但不能通过外部写入。
w	只写	这些寄存器仅用于控制方式。它们可通过微处理器写入但不能读出。读这些寄存器会返回不确定的值。例如，TestAnaSelect 寄存器用于确定 AUX 脚的信号，但是不可能读出它的内容。

表 5-2　寄存器位的状态和设计

5.2　寄存器描述

5.2.1　页 0：命令和状态

5.2.1.1　Page 寄存器

选择寄存器页

名称：Page　　地址：0x00,0x08,0x10,0x18,0x20,0x28,0x30,0x38　　复位值：1000000,0x80

7	6	5	4	3	2	1	0
UsePageSelect	0	0	0	0	PageSelect		
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

位描述

位	符号	功能
7	UsePageSelect	如果设置为 1，PageSelect 的值作为寄存器地址 A5,A4 和 A3.寄存器地址的最低位由地址脚或内部地址锁存单独定义。 如果设置为 0，内部地址所处的整个内容定义寄存器地址。地址脚的用途见表 4-2。
6~3	0000	保留将来之用
2~0	PageSelect	仅当 UsePageSelect 设置为 1 时才使用 PageSelect 的值。此情况下，它指定寄存器页（寄存器地址 A5,A4 和 A3）

5.2.1.2 Command 寄存器

启动和停止命令的执行

名称：Command　　地址：0x01　　复位值：x0000000，0xx0

7	6	5	4	3	2	1	0
IFDetect Busy	0	Command					
r	r	dy	dy	dy	dy	dy	dy

位描述

位	符号	功能
7	IfDetectBusy	显示接口检测逻辑的状态：设置为 0 表示“接口检测成功完成”；设置为 1 表示“接口检测正在进行”
6	0	保留将来之用
5~0	Command	根据命令代码激活命令。读该寄存器显示实际执行的命令。

5.2.1.3 FIFOData 寄存器

64 字节 FIFO 缓冲区输入和输出

名称：FIFOData　　地址：0x02　　复位值：xxxxxxxx，0xxx

7	6	5	4	3	2	1	0
FIFOData							
dy	dy	dy	dy	dy	dy	dy	dy

位描述

位	符号	功能
7~0	FIFOData	用于内部 64 字节 FIFO 缓冲区的数据输入和输出口.FIFO 缓冲区对所有输入和输出数据流起到并入/并出的作用。

5.2.1.4 PrimaryStatus 寄存器

接收器、发送器和 FIFO 缓冲区状态标志。

名称：PrimaryStatus　　地址：0x03　　复位值：00000101，0x05

7	6	5	4	3	2	1	0
0	ModemState			IRq	Err	HiAlert	LoAlert
r	r	r	r	r	r	r	r

位描述

位	符号	功能
7	0	保留将来之用
6~4	ModemState	ModemState 显示发送器和接收器状态机的状态。 状态　状态名　描述 000　Idle　由于发送器和接收器都未启动或输入数据，因此它们都不处于操作中。 001　TxSOF　发送“帧起始”模式 010　TxData　从 FIFO 缓冲区发送数据（或冗余检测位） 011　TxEOF　发送“帧结束”模式 100　GoToRx1　中间状态，当接收器启动时 GoToRx2　中间状态，当接收器停止时 101　PrepareRx　等待直到 RxWait 寄存器中所选择的时间周期 110　AwaitingRx　计数器激活：等待管脚 Rx 的输入信号 111　Receiving　接收数据
3	IRQ	
2	Err	如果 ErrorFlag 寄存器中任何错误标志置位，该位设置为 1。
1	HiAlert	当保存在 FIFO 缓冲区内的字节数满足下面的等式：HiAlert=（64－FIFOLength）≤WaterLevel 时，该位设置为 1。 例如，　FIFOLength=60，WaterLevel=4　HiAlert=1 FIFOLength=59，WaterLeval=4　HiAlert=0
0	LoAlert	当保存在 FIFO 中的字节数满足下面的等式：HiAlert=（64－FIFOLength）≤WaterLevel 时，该位设置为 1。 例如，　FIFOLength=4，WaterLevel=4　HiAlert=1 FIFOLength=5，WaterLeval=4　HiAlert=0

5.2.1.5 FIFOLength

FIFO 中的缓冲字节数

名称：FIFOLength　地址：0x04　复位值：00000000，0x00

7	6	5	4	3	2	1	0
0	FIFOLength						
r	r	r	r	r	r	r	r

位描述

位	符号	功能
7	0	保留将来之用
6~0	FIFOLengh	指示保存在 FIFO 缓冲区的字节数，写入 FIFOData 寄存器增加，读减少 FIFOLength

5.2.1.6 SecondaryStatus 寄存器

不同的状态标志

名称：SecondaryStatus　　地址：0x05　　复位值：01100000，0x60

7	6	5	4	3	2	1	0
TRunning	E2Ready	CRCReady	0	0	RxLastBits		
r	r	r	r	r	r	r	r

位描述

位	符号	功能
7	TRunning	如果为 1，MF RC500 的定时器单元正在运行，例如，计数器会在下个定时器时钟将 TimerValue 寄存器值减一。
6	E2Ready	如果为 1，MF RC500 已经完成对 E2PROM 的编程
5	CRCReady	如果为 1，MF RC500 已经完成 CRC 的计算
4~3	00	该值不会被改变
2~0	RxLastBits	显示最后接收字节的有效位个数。如果为 0，整个字节有效。

5.2.1.7 InterruptEn 寄存器

使能和禁止中断请求通过的控制位

名称：InterruptEn　　地址：0x06　　复位值：00000000，0x00

7	6	5	4	3	2	1	0
SetIEn	0	TimerEn	TxIEn	RxIEn	IdleIEn	HiAlertIEn	LoAlertIEn
w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	符号	功能
7	SetIEn	该位置位定义在 InterruptEn 寄存器中置位的位，该位清零将清除标记的位
6	0	保留将来之用
5	TimerEn	允许将定时器中断请求（由 TimerIRq 指示）传递给脚 IRQ。除了通过 SetIEn 外，该位不能直接置位或清零。
4	TxIEn	允许将发送器中断请求（由 TxIRq 指示）传递给脚 IRQ。除了通过 SetIEn 外，该位不能直接置位或清零。
3	RxIEn	允许将接收器中断请求（由 RxIRq 指示）传递给脚 IRQ。除了通过 SetIEn 外，该位不能直接置位或清零。
2	IdleIEn	允许将 Idle 中断请求（由 IdleIRq 指示）传递给脚 IRQ。除了通过 SetIEn 外，该位不能直接置位或清零。
1	HiAlertIEn	允许将 HiAlert 中断请求(由 HiAlertIRq 指示)传递给脚 IRQ。除了通过 SetIEn 外，该位不能直接置位或清零。
0	LoAlertIEn	允许将 LoAlert 中断请求(由 LoAlertIRq 指示)传递给脚 IRQ。除了通过 SetIEn 外，该位不能直接置位或清零。

5.2.1.8 InterruptRq 寄存器

中断请求标志

名称：InterruptRq　　地址：0x07　　复位值：00000000，0x00

7	6	5	4	3	2	1	0
SetIRq	0	TimerIRq	TxIRq	RxIRq	IdleIRq	HiAlertIRq	LoAlertIRq
w	r/w	dy	dy	dy	dy	dy	dy

位描述

位	符号	功能
7	SetIRq	设置为 1 定义在 InterruptRq 寄存器中置 1 的位 设置为 0 清除 InterruptIRq 中标记的位
6	0	保留将来之用
5	TimerIRq	当定时器 TimerValue 寄存器值减为 0 时置位
4	TxIRq	当下列条件之一发生时置位： Transceive 命令：所有数据都已发送 Auth1 和 Auth2 命令：所有数据都已发送 WriteE2 命令：所有数据都已编程 CalcCRC 命令：所有数据都已处理
3	RxIRq	当接收终止时该位置位
2	IdleIRq	当命令由其自身终止时该位置位。例如，当命令寄存器的值从任何寄存器变为 Idle 寄存器的值时。 如果一个未知的命令启动，IdleIRq 置位。 由微处理器启动 Idle 命令不置位 IdleIRq。
1	HiAlertIRq	当 HiAlert 置位时，该位置位。与 HiAlert 相反，HiAlertIRq 保存该事件并只能通过 SetIRq 复位。
0	LoAlertIRq	当 LoAlert 置位时，该位置位。LoAlertIRq 保存该事件并只能通过 SetIRq 复位。

5.2.2 页 1：控制和状态

5.2.2.1 页寄存器

选择寄存器页。见 5.2.1.1

5.2.2.2 控制寄存器

不同的控制标志：例如：定时器，节电

名称：Control　　地址：0x09　　复位值：00000000，0x00

7	6	5	4	3	2	1	0
0	0	StandBy	PowerDown	Crypto1On	TStopNow	TStartNow	FlushFIFO
r/w	r/w	dy	dy	dy	w	w	w

位描述

位	符号	功能
7~6	00	保留将来之用
5	StandBy	将该位置 1 进入软件掉电模式。这表示内部电流消耗模块关闭，晶振保持运行。
4	PowerDown	将该位置 1 进入软件掉电模式。这表示内部电流消耗模块包括晶振在内关闭。
3	Crypto1On	该位指示 Crypto1 单元打开，因此与卡的所有数据通信都被加密。
2	TStopNow	将该位置 1 立即停止定时器，读该位总是返回 0。
1	TStartNow	将该位置 1 立即启动定时器，读该位总是返回 0。
0	FlushFIFO	将该位置 1 立即清除内部 FIFO 缓冲区的读/写指针和 FIFOOvfl 标志。 读该位总是返回 0。

5.2.2.3 ErrorFlag 寄存器

Error 标志指示上一个执行命令的错误状态

名称：ErrorFlag　　地址：0x0A　　复位值：00000000，0x00

7	6	5	4	3	2	1	0
0	KeyErr	AccessErr	FIFOOvfl	CRCErr	FramingErr	ParityErr	CollErr
r	r	r	r	r	r	r	r

位描述

位	符号	功能
7	0	保留将来之用
6	KeyErr	如果 LoadKeyE2 或 LoadKey 命令识别出输入数据不是根据密匙格式定义编码，则将该位置位。启动 LoadKeyE2 或 LoadKey 命令时该位清零
5	AccessErr	如果对 E^2PROM 的访问权限被禁止，该位置位。 启动与 E^2PROM 相关的命令时该位清零。
4	FIFOOvfl	如果微处理器或 MF RC500 内部状态机（例如接收器）试图将数据写入 FIFO 缓冲区而 FIFO 缓冲区已满时，该位置位。
3	CRCErr	如果 RxCRCEn 置位且 CRC 失败，该位置位；该位在 PrepareRx 状态中接收器的启动阶段自动清零
2	FramingErr	如果 SOF 不正确，该位置位；该位在 PrepareRx 状态中接收器的启动阶段自动清零
1	ParityErr	如果奇偶校验失败，该位置位；该位在 PrepareRx 状态中接收器的启动阶段自动清零
0	CollErr	如果检测到一个位冲突，该位置位；该位在 PrepareRx 状态中接收器的启动阶段自动清零

5.2.2.4 CollPos 寄存器

RF 接口上检测到的第一个位冲突的位置。

名称：CollPos　　地址：0x0B　　复位值：00010011，0x00

7	6	5	4	3	2	1	0
CollPos							
r	r	r	r	r	r	r	r

位描述

位	符号	功能
7~0	CollPos	该寄存器指示在接收到的帧中第一个检测到的冲突位的位置。 例： 0x00 指示在起始位的位冲突 0x01 指示在第 1 位的位冲突 0x08 指示在第 8 位的位冲突

5.2.2.5 TimerValue 寄存器

定时器的实际值

名称：TimerValue　　地址：0x0C　　复位值：XXXXXXXX，0xXX

7	6	5	4	3	2	1	0
TimerValue							
r	r	r	r	r	r	r	r

位描述

位	符号	功能
7~0	TimerValue	该寄存器显示定时器计数器的实际值。

5.2.2.6 CRCResultLSB 寄存器

CRC 协处理器寄存器低字节

名称：CRCResultLSB　　地址：0x0D　　复位值：XXXXXXXX，0xXX

7	6	5	4	3	2	1	0
CRCResultLSB							
r	r	r	r	r	r	r	r

位描述

位	符号	功能
7~0	CRCResultLSB	该寄存器显示 CRC 寄存器低字节的实际值。它只在 CRCReady 设为 1 时有效

5.2.2.7 CRCResultMSB 寄存器

CRC 协处理器寄存器高字节

名称：CRCResultMSB　　地址：0x0E　　复位值：XXXXXXXX，0xXX

7	6	5	4	3	2	1	0
CRCResultMSB							
r	r	r	r	r	r	r	r

位描述

位	符号	功能
7~0	CRCResultMSB	该寄存器显示 CRC 寄存器高字节的实际值。它只在 CRCReady 设为 1 时有效。对于 8 位 CRC 校验，该寄存器值未定义。

5.2.2.8 BitFraming 寄存器

位方式帧的调节

名称：BitFraming　　地址：0x0F　　复位值：00000000，0x00

7	6	5	4	3	2	1	0
0	RxAlign			0	TxLastBits		
r/w	dy	dy	dy	r/w	dy	dy	dy

位描述

位	符号	功能
7	0	保留将来之用
6~4	RxAlign	用于位方式帧的接收：RxAlign 定义了接收的第一个位存储到 FIFO 的位置。更多的位存储到后面的位位置。 在接收后，RxAlign 自动清零。 例：RxAlign＝0：接收的最低位存在位 0，接收的第二个位存在位 1。 RxAlign＝1：接收的最低位存在位 1，接收的第二个位存在位 2。 RxAlign＝3：接收的最低位存在位 3，接收的第二个位存在位 4。 RxAlign＝7：未定义 注：强烈建议不要使用 RxAlign＝7 以防止数据丢失。在位位置 6，14，22，30，38（CollPos）检测到的位冲突不能通过 RxAlign 解决，需要软件来实现。
3	0	保留将来之用
2~0	TxLastBits	用于位方式帧的发送：TxLastBits 定义要发送的最后一个字节的位数目。000 指示最后字节的所有位都要发送。在发送之后 TxLastBits 自动清零。

5.2.3 页 2：发送器和控制

5.2.3.1 页寄存器

选择寄存器页。见 5.2.1.1

5.2.3.2 TxControl 寄存器

控制天线脚 TX1 和 TX2 的逻辑状态

名称：TxControl　　地址：0x11　　复位值：01011000，0x58

7	6	5	4	3	2	1	0
0	ModulatorSource		1	TX2Inv	TX2Cw	TX2RFEn	TX1RFEn
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	符号	功能
7	0	该值不会被改变
6~5	Modulator Source	调制器的输入源： 00：低 01：高 10：内部编码器 11：管脚 MFIN
4	1	该值不会被改变
3	TX2Inv	设置为 1，管脚 TX2 上的输出信号将传递一个反相的 13.56MHz 能量载波
2	TX2Cw	设置为 1，管脚 TX2 上的输出信号将连续传递未调制的 13.56MHz 能量载波
1	TX2RFEn	设置为 1，管脚 TX2 上的输出信号将传递由发送数据调制的 13.56MHz 能量载波
0	TX1RFEn	设置为 1，管脚 TX1 上的输出信号将传递由发送数据调制的 13.56MHz 能量载波

5.2.3.3 CwConductance 寄存器

选择天线驱动脚 TX1 和 TX2 的电导率

名称：CwConductance　　地址：0x12　　复位值：00111111，0x3F

7	6	5	4	3	2	1	0
0	0	GsCfgCW					
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	符号	功能
7~6	00	该值不会被改变
5~0	GsCfgCW	该寄存器值定义输出驱动器的电导率.可用于调整输出功率以及电流消耗和操作距离.

5.2.3.4 PreSet13 寄存器

名称：PreSet13　　地址：0x13　　复位值：00111111，0x3F

7	6	5	4	3	2	1	0
0	0	1	1	1	1	1	1
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

注:该寄存器值不会被改变!

5.2.3.5 PreSet14 寄存器

名称：PreSet14　　地址：0x14　　复位值：00011001，0x19

7	6	5	4	3	2	1	0
0	0	0	1	1	0	0	1
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

注:该寄存器值不会被改变!

5.2.3.6 ModWidth 寄存器

选择调制脉冲的宽度

名称：ModeWidth　　地址：0x15　　复位值：00010011，0x13

7	6	5	4	3	2	1	0
ModeWidth							
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	符号	功能
7~0	ModeWidth	该寄存器根据 Tmod=2*(ModeWidth+1)/fc 定义调制脉冲宽度。

5.2.3.7 PreSet16 寄存器

名称：PreSet16　　地址：0x16　　复位值：00000000，0x00

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

注:该寄存器值不会被改变!

5.2.3.8 PreSet17 寄存器

名称：PreSet17　　地址：0x17　　复位值：00000000，0x00

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

注:该寄存器值不会被改变!

5.2.4　页 3:接收器和解码器控制

5.2.4.1　页寄存器

选择寄存器页。见 5.2.1.1

5.2.4.2　RxControl1 寄存器

控制接收器状态

名称：RxControl1　　地址：0x19　　复位值：01110011，0x73

7	6	5	4	3	2	1	0
0	1	1	1	0	0	Gain	
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	符号	功能
7~2	011100	该值不会被改变
1~0	Gain	该寄存器定义接收器信号电压增益因素: 00:　27db 01:　30db 10:　38db 11:　42db

5.2.4.3　DecodeControl 寄存器

名称：RxControl1　　地址：0x1A　　复位值：00001000，0x08

7	6	5	4	3	2	1	0
0	0	ZeroAfer Coll	0	1	0	0	0
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	符号	功能
7~6	00	该值不会被改变
5	ZeroAferColl	如果设置为 1，在一个位冲突之后的任何位都屏蔽为 0。这就很容易由 ISO14443A 中定义的防冲突处理进行处理。
4~0	01000	该值不会被改变

5.2.4.4　BitPhase 寄存器

选择发送器和接收器时钟之间的位相位。

名称：RxControl1　　地址：0x1B　　复位值：10101101，0xAD

7	6	5	4	3	2	1	0
BitPhase							
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	符号	功能
7~0	BitPhase	定义发送器和接收器时钟之间的位相位 注:该寄存器的正确值对正常操作是非常必要的.

5.2.4.5 RxThreshold 寄存器

选择位解码器的阀值

名称：RxThreshold　　地址：0x1C　　复位值：11111111，0xFF

7	6	5	4	3	2	1	0
MinLevel				CollLevel			
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	符号	功能
7~4	MiniLevel	定义解码器输入端可接受的最小信号强度。如果信号小于该值将不进行计算
3~0	CollLevel	定义解码器输入的最小信号强度，该信号必须被 Manchester 编码信号的弱半位达到以产生相对于强半位幅度的位冲突。

5.2.4.6 PreSet1D 寄存器

名称：PreSet1D　　地址：0x1D　　复位值：00000000，0x00

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

注:该寄存器值不会被改变!

5.2.4.7 RxControl2 寄存器

控制解码器的状态并定义接收器的输入源。

名称：RxThreshold　　地址：0x1E　　复位值：010000011，0x41

7	6	5	4	3	2	1	0
RcvClkSell	RxAutoPD	0	0	0	0	DecoderSource	
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	符号	功能
7	RcvClkSell	如果设置为 1，I-时钟用作接收器时钟；为 0 表示使用 Q-时钟。I-时钟和 Q-时钟之间有 90º相移。
6	RxAutoPD	如果设置为 1，接收器电路在接收前自动打开并在完成后关闭。这样可减少电流的消耗。 如果设置为 0，接收器始终有效
5~2	0000	该值不会被改变
1~0	DecoderSource	选择解码器输入源： 00：低 01：内部解调器 10：管脚 MFIN 输入的副载波调制 Manchester 编码信号 11：管脚 MFIN 输入的基带 Manchester 编码信号

5.2.4.8 ClockQControl 寄存器

控制时钟产生用于 90º相移的 Q 信道时钟

名称：ClockQControl　　地址：0x1F　　复位值：000XXXXX，0xXX

7	6	5	4	3	2	1	0
ClkQ180Deg	ClkQCalib	0	ClkQDelay				
r	r/w	r/w	dy	dy	dy	dy	dy

位描述

位	符号	功能
7	ClkQ180Deg	如果 Q-时钟与 I-时钟的相移超过 180º,该位置 1,否则为 0。
6	ClkQCalib	如果该位为 0，Q-时钟在复位后和从卡接收数据后自动校准
5	0	该值不会被改变
4~0	ClkQDelay	该寄存器显示实际用于产生 I-时钟的 90º相移以获得 Q-时钟的延迟元素的数目。它可由微处理器直接写入或在校准周期自动写入。

5.2.5 页 4：RF 时序和信道冗余

5.2.5.1 页寄存器

选择寄存器页。见 5.2.1.1

5.2.5.2 RxWait 寄存器

选择发送后，接收器启动前的时间间隔

名称：ClockQControl　　地址：0x21　　复位值：00000101，0x06

7	6	5	4	3	2	1	0
RxWait							
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	符号	功能
7~0	RxWait	在数据发送后，接收器的启动由于 RxWait 位时钟而延迟。在这段“帧保护时间”内管脚 Rx 上的任何信号都被忽略。

5.2.5.3 ChannelRedundancy 寄存器

选择 RF 信道上数据完整性检测的类型和模式

名称：ChannelRedundancy　　地址：0x22　　复位值：00000011，0x03

7	6	5	4	3	2	1	0
0	CRCMSB First	CRC 3309	CRC8	RxCRCEn	TxCRCEn	ParityOdd	ParityEn
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	符号	功能
7	0	该值不会被改变
6	CRCMSBFirst	如果设置为 1，CRC 计算首先将最高位移入 CRC 协处理器。 如果设置为 0，CRC 计算从最低位开始。 注：根据 ISO14443A 该位必须为 0
5	CRC3309	如果设置为 1，CRC 计算根据 ISO/IEC3309 执行 注：根据 ISO14443A 该位必须为 0
4	CRC8	如果设置为 1，计算任何 8 位 CRC 如果设置为 0，计算一个 16 位 CRC
3	RxCRCEn	如果设置为 1，接收帧的最后字节被解释为 CRC 字节。如果 CRC 是正确的，CRC 字节不放入 FIFO。在有错误的情况下，CRCErr 标志置位。 如果设置为 0，无 CRC
2	TxCRCEn	如果设置为 1，对发送数据进行 CRC 计算并将 CRC 字节加到数据流中。 如果设置为 0，不发送 CRC
1	ParityOdd	如果设置为 1，单独产生或者出现奇数的奇偶校验 如果设置为 0，单独产生或者出现偶数的奇偶校验 注：根据 ISO14443A 该位必须为 1
0	ParityEn	如果设置为 1，奇偶校验位在每个字节后插入发送数据流中并会出现在接收数据流的每个字节后。 如果设置为 0，不会产生或者出现奇偶校验位

5.2.5.4 CRCPresetLSB 寄存器

CRC 寄存器预设值的低字节

名称：CRCPresetLSB　　地址：0x23　　复位值：01010011，0x63

7	6	5	4	3	2	1	0
CRCPresetLSB							
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	符号	功能
7~0	CRCPresetLSB	CRCPresetLSB 定义 CRC 计算的起始值。如果 CRC 计算使能，该值在发送、接收和 CalcCRC 命令开始时装入 CRC。

5.2.5.5 CRCPresetMSB

CRC 寄存器预设值的高字节

名称：CRCPresetMSB　　地址：0x24　　复位值：01010011，0x63

7	6	5	4	3	2	1	0
CRCPresetMSB							
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	符号	功能
7~0	CRCPresetMSB	CRCPresetMSB 定义 CRC 计算的起始值。如果 CRC 计算使能，该值在发送、接收和 CalcCRC 命令开始时装入 CRC。 注：如果 CRC8 为 1，该寄存器无效

5.2.5.6 PreSet25 寄存器

名称：PreSet25　　地址：0x25　　复位值：00000000，0x00

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

注:该寄存器值不会被改变!

5.2.5.7 MFOUTSelect 寄存器

选择输出到管脚 MFOUT 的内部信号

名称：MFOUTSelect　　地址：0x26　　复位值：00000000，0x00

7	6	5	4	3	2	1	0
0	0	0	0	0	MFOUTSelect		
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	符号	功能
7~3	00000	该值不会被改变
2~0	MFOUTSelect	MFOUTSelect 定义输出到脚 MFOUT 的信号： 000　恒为低 001　恒为高 010　来自内部编码器的调制信号（包络），Miller 编码 011　串行数据流，非 Miller 编码 100　能量载波解调器输出信号（卡调制信号） 101　副载波解调器输出信号（Manchester 编码卡信号） 110　RFU 111　RFU

5.2.5.8 PreSet27 寄存器

名称：PreSet27　　地址：0x27　　复位值：00000000，0x00

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

注:该寄存器值不会被改变!

5.2.6 页 5：FIFO、定时器和 IRQ 管脚配置

5.2.6.1 页寄存器

选择寄存器页。见 5.2.1.1

5.2.6.2 FIFOLevel 寄存器

定义 FIFO 上溢和下溢警告界限

名称：FIFOLevel　　地址：0x29　　复位值：00001000，0x08

7	6	5	4	3	2	1	0
0	0	WaterLevel					
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	符号	功能
7~6	00	该值不会被改变
5~0	WaterLevel	该寄存器定义了 MF RC500 用于微处理器的 FIFO 上溢和下溢警告界限： 如果 FIFO 缓冲区剩余空间等于或小于 FIFO 缓冲区中的 WaterLevel 字节，HiAlert 设置为 1。 如果等于或小于 FIFO 缓冲区中的 WaterLevel 字节，LoAlert 设置为 1。

5.2.6.3 TimerClock 寄存器

选择定时器时钟的分频值

名称：TimerClock　　地址：0x2A　　复位值：00000111，0x07

7	6	5	4	3	2	1	0
0	0	TAutoRestart	TPreScaler				
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	符号	功能
7~6	00	该值不会被改变
5	TAutoRestart	如果设置为 1，定时器从 TReloadValue 处自动重新开始向下计数，而不是向下计数到零。 如果设置为 0，定时器减少到零并且 TimerIRq 置位。
4~0	TPreScaler	定义定时器时钟 f_{Timer}。TPreScaler 可以从 0 调整到 21。下面的公式用于计算 f_{Timer}：$f_{Timer}=13.56MHz/2^{TPreScaler}$

5.2.6.4 TimerControl 寄存器

选择定时器的起始和停止条件

名称：TimerControl　　地址：0x2B　　复位值：00000110，0x06

7	6	5	4	3	2	1	0
0	0	0	0	TStopRxEnd	TStopRxBegin	TStartTxEnd	TStartTxBegin
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	符号	功能
7~4	0000	该值不会被改变
3	TStopRxEnd	如果设置为 1，当数据接收结束时定时器自动停止。0 表示定时器不受该条件影响。
2	TStopRxBegin	如果设置为 1，当接收到第一个有效位时定时器自动停止。0 表示定时器不受该条件影响。
1	TStartTxEnd	如果设置为 1，当数据发送结束时定时器自动停止。0 表示定时器不受该条件影响。
0	TStartTxBegin	如果设置为 1，当第一个字节发送时定时器自动停止。0 表示定时器不受该条件影响。

5.2.6.5 TimerReload 寄存器

定义定时器的当前值

名称：TimerReload　　地址：0x2C　　复位值：00001010，0x0A

7	6	5	4	3	2	1	0
TReloadValue							
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	符号	功能
7~0	TReloadValue	启动时定时器装入 TReloadValue。改变该寄存器只在下次启动事件影响定时器。 如果 TReloadValue 设置为 0，定时器将不能启动。

5.2.6.6 IRQPinConfig 寄存器

配置管脚 IRQ 的输出状态

名称：IRQPinConfig　　地址：0x2D　　复位值：00000010，0x02

7	6	5	4	3	2	1	0
0	0	0	0	0	0	IRQInv	IRQPushPull
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

位描述

位	符号	功能
7~2	000000	该值不会被改变
1	IRQInv	如果设置为 1，管脚 IRQ 上的信号与对应的位 IRq 状态相反。 如果为 0，表示相同
0	IRQPushPull	如果设置为 1，管脚 IRQ 为标准 CMOS 输出。 如果为 0，IRQ 为开漏输出。

5.2.6.7 PreSet2E 寄存器

名称：PreSet2E　　地址：0x2E　　复位值：00000000，0x00

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

注:该寄存器值不会被改变!

5.2.6.8 PreSet2F 寄存器

名称：PreSet2F　　地址：0x2F　　复位值：00000000，0x00

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

注:该寄存器值不会被改变!

5.2.7 页 6：RFU

5.2.7.1 页寄存器

选择寄存器页。见 5.2.1.1

5.2.7.2 RFU 寄存器

名称：RFU　　地址：0x31,0x32,0x33,0x34,0x35,0x36,0x37　　复位值：00000000，0x00

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

注:这些寄存器保留将来之用。

5.2.8 页 7：测试控制

5.2.8.1 页寄存器

选择寄存器页。见 5.2.1.1

5.2.7.2 RFU 寄存器

名称：RFU　　地址：0x39　　复位值：00000000，0x00

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0
w	w	w	w	w	w	w	w

注:该寄存器保留将来之用。

5.2.8.3 TestAnaSelect 寄存器

选择模拟测试信号

名称：TestAnaSelect　　地址：0x3A　　复位值：00000000，0x00

7	6	5	4	3	2	1	0
0	0	0	0	TestAnaOutSel			
w	w	w	w	w	w	w	w

位描述

位	符号	功能	
7~4	0000	该值不会被改变	
3~0	TestAnaOutSel	该寄存器选择输出到管脚 AUX 的内部模拟信号。细节参见 19.3	
		值	信号名称
		0	V_{mid}
		1	$V_{bandgap}$
		2	$V_{RxFolll}$
		3	$V_{RxFollQ}$
		4	V_{RxAmpl}
		5	V_{RxAmpQ}
		6	V_{CorrNI}
		7	V_{CorrNQ}
		8	V_{CorrDI}
		9	V_{CorrDQ}
		A	V_{EvalL}
		B	V_{EvalR}
		C	V_{Temp}
		D	RFU
		E	RFU
		F	RFU

5.2.8.4 PreSet3B 寄存器

名称：PreSet3B　　地址：0x3B　　复位值：00000000，0x00

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

注:该寄存器值不会被改变!

5.2.8.5 PreSet3C 寄存器

名称：PreSet3C　　地址：0x3C　　复位值：00000000，0x00

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

注:该寄存器值不会被改变!

5.2.8.6 TestDigiSelect 寄存器

选择数字测试模式

名称：TestDigiSelect　　地址：0x3D　　复位值：00000000，0x00

7	6	5	4	3	2	1	0
SignalTo MFOUT	TestDigiSignalSel						
w	w	w	w	w	w	w	w

位描述

位	符号	功能
7	SignalToMFOUT	设置为 1，MFOUTSelect 中的设定无效，取而代之的是输出到管脚 MFOUT 的 TestDigiSignalSel 中定义的数字测试信号 设置为 0 时，由 MFOUTSelect 定义输出到管脚 MFOUT 的信号
6~0	TestDigiSignalSel	选择输出到管脚 MFOUT 的数字信号 细节请参阅 19.4

TestDigiSignalSel	信号名称
F4hex	s_data
E4hex	s_valid
D4hex	s_coll
C4hex	s_clock
B5hex	rd_sync
A5hex	wr_sync
96hex	int_clock

5.2.8.7 RFU 寄存器

名称：RFU　　地址：0x3E,0x3F　　复位值：00000000，0x00

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

注:这些寄存器保留将来之用。

5.3 MF RC500 寄存器标志位汇总

标志	寄存器	地址 寄存器,位的位置
AccessErr	ErrorFlag	0x0A, bit 5
BitPhase	BitPhase	0x1B, bits 7:0
ClkQ180Deg	ClockQControl	0x1F, bit 7
ClkQCalib	ClockQControl	0x1F, bit 6
ClkQDelay	ClockQControl	0x1F, bits 4:0
CollErr	ErrorFlag	0x0A, bit 0
CollLevel	RxThreshold	0x1C, bits 3:0
CollPos	CollPos	0x0B, bits 7:0
Command	Command	0x01, bits 5:0
CRC3309	ChannelRedundancy	0x22, bit 5
CRC8	ChannelRedundancy	0x22, bit 4
CRCErr	ErrorFlag	0x0A, bit 3
CRCMSBFirst	ChannelRedundancy	0x22, bit 6
CRCPresetLSB	CRCPresetLSB	0x23, bits 7:0
CRCPresetMSB	CRCPresetMSB	0x24, bits 7:0
CRCReady	SecondaryStatus	0x05 , bit 5
CRCResultMSB	CRCResultMSB	0x0E, bits 7:0
CRCResultLSB	CRCResultLSB	0x0D, , bits 7:0
Crypto1On	Control	0x09, bit 3
DecoderSource	RxControl2	0x1E, bits 1:0
E2Ready	SecondaryStatus	0x05, bit 6
Err	PrimaryStatus	0x03, bit 2
FIFOData	FIFOData	0x02, bits 7:0
FIFOLength	FIFOLength	0x04, bits 7:0
FIFOOvfl	ErrorFlag	0x0A, bit 4
FlushFIFO	Control	0x09, bit 0
FramingErr	ErrorFlag	0x0A, bit 2
Gain	RxControl1	0x19, bits 1:0
GsCfgCW	CWConductance	0x12, bits 5:0
HiAlert	PrimaryStatus	0x03, bit 1
HiAlertIEn	InterruptEn	0x06, bit 1
HiAlertIRq	InterruptRq	0x07, bit 1

标志	寄存器	地址 寄存器,位的位置
IdleIEn	InterruptEn	0x06, bit 2
IdleIRq	InterruptRq	0x07, bit 2
IFDetectBusy	Command	0x01, bit 7
IRq	PrimaryStatus	0x03, bit 3
IRQInv	IRQPinConfig	0x2D, bit 1
IRQPushPull	IRQPinConfig	0x2D, bit 0
KeyErr	ErrorFlag	0x0A, bit 6
LoAlert	PrimaryStatus	0x03, bit 0
LoAlertIEn	InterruptEn	0x06, bit 0
LoAlertIRq	InterruptRq	0x07, bit 0
MFOUTSelect	MFOUTSelect	0x26, bits 2:0
MinLevel	RxThreshold	0x1C, bits 7:4
ModemState	PrimaryStatus	0x03 , bit 6:4
ModulatorSource	TxControl	0x11, bits 6:5
ModWidth	ModWidth	0x15, bits /:0
PageSelect	Page	0x00, 0x08, 0x10, 0x18, 0x20, 0x28, 0x30, 0x38, bits 2:0
ParityEn	ChannelRedundancy	0x22, bit 0
ParityErr	ErrorFlag	0x0A, bit 1
ParityOdd	ChannelRedundancy	0x22 , bit 1
PowerDown	Control	0x09, bit4
RcvClkSelI	RxControl2	0x1E, bit 7
RxAlign	BitFraming	0x0F, bits 6:4
RxAutoPD	RxControl2	0x1E, bit 6
RxCRCEn	ChannelRedundancy	0x22, bit 3
RxIEn	InterruptEn	0x06, bit 3
RxIRq	InterruptRq	0x07, bit 3
RxLastBits	SecondaryStatus	0x05, bits 2:0
RxWait	RxWait	0x21, bits 7:0
SetIEn	InterruptEn	0x06, bit 67
SetIRq	InterruptRq	0x07, bit 7
SignalToMFOUT	TestDigiSelect	0x3D, bit 7
StandBy	Control	0x09, bit 5
TAutoRestart	TimerClock	0x2A, bit 5

标志	寄存器	地址 寄存器,位的位置
TestAnaOutSel	TestAnaSelect	0x3A, bits 6:4
TestDigiSignalSel	TestDigiSelect	0x3D, bit 6:0
TimerIEn	InterruptEn	0x06, bit 5
TimerIRq	InterruptRq	0x07, bit 5
TimerValue	TimerValue	0x0C, bits 7:0
TPreScaler	TimerClock	0x2A, bits 4:0
TReloadValue	TimerReload	0x2C, bits 7:0
TRunning	SecondaryStatus	0x05, bit 7
TStartTxBegin	TimerControl	0x2B, bit 0
TStartTxEnd	TimerControl	0x2B, bit 1
TStartNow	Control	0x09, bit 1
TStopRxBegin	TimerControl	0x2B, bit 2
TStopRxEnd	TimerControl	0x2B, bit 3
TStopNow	Control	0x09, bit 2
TX1RFEn	TxControl	0x11, bit 0
TX2Cw	TxControl	0x11, bit 3
TX2Inv	TxControl	0x11, bit 3
TX2RFEn	TxControl	0x11, bit 1
TxCRCEn	ChannelRedundancy	0x22, bit 2
TxIEn	InterruptEn	0x06, bit 4
TxIRq	InterruptRq	0x07, bit 4
TxLastBits	BitFraming	0x0F, bits 2:0
UsePageSelect	Page	0x00, 0x08, 0x10, 0x18, 0x20, 0x28, 0x30, 0x38, bit 7
WaterLevel	FIFOLevel	0x29, bits 5:0
ZeroAfterColl	DecoderControl	0x1A, bit 5

5.4 寄存器寻址方式

可通过 3 种机制对 MF RC500 进行操作：

- 通过执行命令初始化功能和控制数据操作
- 通过一系列的可配置位配置电气和功能状态
- 通过读取状态标志监控 MF RC500 的状态

命令、配置位和标志都可通过微处理器接口访问。MF RC500 可内部寻址 64 个寄存器。这需要 6 条地址线。

5.4.1 分页机制

MF RC500 寄存器集被分成 8 页，每页 8 个寄存器。不管当前所选是哪一页，页寄存器总是可以寻址的。

5.4.2 专用的地址总线

使用 MF RC500 专用地址总线，微处理器通过地址脚 A0,A1 和 A2 定义 3 条地址线。这允许在一页内进行寻址。要在不同页的寄存器之间进行切换就需要用到分页机制。

下表列出了寄存器地址的组合状况：

寄存器位： UsePageSelect	寄存器地址					
1	PageSelect2	PageSelect1	PageSelect0	A2	A1	A0

表 5-3 专用地址总线：组合寄存器地址

5.4.3 复用的地址总线

使用 MF RC500 复用的地址总线，微处理器可以一次定义所有的 6 条地址线。这种情况下，可以既可以使用分页机制也可使用线性寻址。

下表列出了寄存器地址的组合状况：

接口总线类型	寄存器位： UsePageSelect	寄存器地址					
复用地址总线（分页模式）	1	PageSelect2	PageSelect1	PageSelect0	AD2	AD1	AD0
复用地址总线（线性寻址）	0	AD5	AD4	AD3	AD2	AD1	AD0

表 5-4 复用地址总线：组合寄存器地址

6 E²PROM 存储器结构

6.1 E²PROM 存储器结构图

块编号	块地址	字节地址	访问权限	存储器内容	相关章节
0	0	00 ... 0F	r	产品信息区	6.2
1	1	10 ... 1F	r/w	启动寄存器初始化文件	6.3.1
2	2	20 ... 2F	r/w		
3	3	30 ... 3F	r/w	寄存器初始化文件	6.3.3
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	Crypto1密匙	6.4
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 E²PROM 存储器结构图

6.2 产品信息区（只读）

字节	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
含意	产品类型标识					RFU			产品序列号				内部			CRC

表 6-2 产品信息区

产品类型标识:

MF RC500 是产品系列的第一个产品。产品系列的每一个成员都有其唯一的产品类型标识。产品类型标识的值如下表所示：

	产品类型标识				
字节	0	1	2	3	4
值	30hex	88hex	F8hex	00hex	XXhex

表 6-3 产品类型标识定义

产品序列号：

MF RC500 有一个 4 字节的序列号，每个器件的序列号都是唯一的。

内部：

3 个字节保存内部微调参数。

CRC：

产品信息区的内容通过一个 CRC 字节保证安全，该 CRC 在启动时检测。

6.3 寄存器初始化文件（读/写）

从 10 到 2F 地址范围内的寄存器初始化在初始化阶段通过启动寄存器初始化文件自动完成。此外，用户可以通过执行 LoadConfig 命令初始化 MF RC500 寄存器。

注：

- 页寄存器（地址：10，18，20，28）跳过不进行初始化
- 确认所有的 PreSet 寄存器都没有改变
- 确认所有保留将来之用的寄存器位（RFU）都设为 0

6.3.1 启动寄存器初始化文件（读/写）

E^2PROM 存储器块地址 1 和 2 的内容用于在初始化阶段自动对 MF RC500 寄存器 10 到 2F 初始化。制造时写入 E^2PROM 的默认值见章节 6.3.2。

分配如下：

E^2PROM 字节地址	寄存器地址	备注
10hex(块 1，字节 0)	10hex	跳过
11hex	11hex	复制
…	…	…
2Fhex(块 1，字节 0)	2Fhex	复制

表 6-4 启动时寄存器初始化字节分配

6.3.2 启动寄存器初始化文件的装载内容

在产品测试阶段，启动寄存器初始化文件使用下表所列出的值进行初始化。每次上电的初始化阶段这些值都写入 MF RC500 寄存器。

E^2PROM 字节地址	寄存器地址	值	描述
10	10	00	页：用户自由使用
11	11	58	TxControl：发送脚 TX1 和 TX2 关闭，桥驱动器配置和调制器由内部数字电路驱动。
12	12	3F	CwConductance：TX1 和 TX2 内阻最小
13	13	3F	PreSet13
14	14	19	PreSet14
15	15	13	ModWidth：Miller 脉冲编码的脉冲宽度设置为标准配置
16	16	00	PreSet16
17	17	00	PreSet17
18	18	00	页：用户自由使用
19	19	73	RxControl1：放大器增益最大
1A	1A	08	DecoderControl：位冲突在数据流中总是表现为高
1B	1B	AD	BitPhase：BitPhase 设置为标准配置
1C	1C	FF	RxThreshold：MinLevel 和 CollLevel 设置为最大
1D	1D	00	PreSet1D
1E	1E	41	RxControl2：Q-时钟用于接收器，“接收器自动关闭”打开，解码器由内部模拟电路驱动。
1F	1F	00	ClockQControl：“Q-时钟自动校准”打开
20	20	00	页：用户自由使用
21	21	06	RxWait：帧保护时间设置为 6 位时钟
22	22	03	ChannelRedundancy：信道冗余根据 ISO14443A 设置
23	23	63	CRCPreSetLSB：CRC 预设值根据 ISO14443A 设置
24	24	63	CRCPreSetMSB：CRC 预设值根据 ISO14443A 设置
25	25	00	PreSet25
26	26	00	MFOUTSelect：管脚设置为低
27	27	00	PreSet27
28	28	00	页：用户自由使用
29	29	08	FIFOLevel：WaterLevel FIFO 缓冲区警戒值设置为标准配置
2A	2A	07	TimerClock:TPreScaler 设置为标准配置，定时器单元重启功能关闭
2B	2B	06	TimerControl:定时器在发送结束时启动，接收开始时停止
2C	2C	0A	Timerreload:TReloadValue：定时器单元预设值设置为标准配置
2D	2D	02	IRQPinConfig:管脚 IRQ 设置为高阻态
2E	2E	00	PreSet2F
2F	2F	00	PreSet2F

表 6-5 启动寄存器初始化文件的装载内容

6.3.3 寄存器初始化文件（读/写）

E^2PROM 存储器从块地址 3 到 7 的内容可用于 MF RC500 寄存器 10 到 2F 的初始化，通过执行 LoadConfig 命令实现。这需要一个 2 字节的变量用作初始化处理时 E^2PROM 的起始字节地址。

分配如下：

E^2PROM 字节地址	寄存器地址	备注
E^2PROM 起始字节地址	10hex	跳过
E^2PROM 起始字节地址+1	11hex	复制
…	…	…
E^2PROM 起始字节地址+31	2Fhex	复制

表 6-6 启动时用于寄存器初始化的字节分配

寄存器初始化文件大到足够装下两套初始化值并剩余一个块(16 字节)留给用户使用。

注：寄存器初始化文件可由用户读写。因此，这些字节可用于保存用户作其它用途的特定数据。

6.4 Crypto1 密匙（只写）

6.4.1 密匙格式

要在 E^2PROM 中保存一个密匙，必须以特定的格式写入。每个密匙字节必须分成从 k0 到 k3 的低 4 位（低半字节）和从 k4 到 k7 的高 4 位（高半字节）。每半个字节在一个字节中保存两次，两个半字节之一按位取反。该格式是 LoadKeyE2（见 16.8.1）和 LoadKey（见 16.8.2）命令连续执行的预处理。如下表所示：

主密匙字节	0(LSB)		1			5(MSB)	
主密匙位	$\overline{k7\ k6\ k5\ k4}$ k7 k6 k5 k4	$\overline{k3\ k2\ k1\ k0}$ k3 k2 k1 k0	$\overline{k7\ k6\ k5\ k4}$ k7 k6 k5 k4	$\overline{k3\ k2\ k1\ k0}$ k3 k2 k1 k0		$\overline{k7\ k6\ k5\ k4}$ k7 k6 k5 k4	$\overline{k3\ k2\ k1\ k0}$ k3 k2 k1 k0
E^2PROM 字节地址	n	n+1	n+2	n+3		n+10	n+11
例	5A	F0	5A	E1		5A	A5

表 6-7 密匙保存格式

例：实际的密匙 A0A1A2A3A4A5hex 的值 5AF05AE15AC35AB45AA5hex 必须写入 E^2PROM

注：尽管有可能将其它格式的数据装入 E^2PROM 的密匙存储区，但这样一个密匙不可能获得有效的卡验证。LoadKeyE2 命令将失败。

6.4.2 E^2PROM 中的密匙存储

MF RC500 在 E^2PROM 中保留了 384 字节用于保存 Crypto1 密匙。它不使用存储器分割，反映了密匙存储的 12 字节结构。因此，专用存储区的每个字节都可以是密匙的起始字节。

例：如果一个密匙从一个 E^2PROM 块的最后一个字节开始（例如：密匙字节保存在 12Fhex）。接下来的字节保存在下一个 E^2PROM 块（例如：密匙字节 1 保存在 130hex，字节 2 保存在 131hex 而字节 11 保存在 13Ahex）。

一个密匙占用 384 字节存储区中的 12 个字节，在 E^2PROM 中可以保存 32 个不同的密匙。

注：不可能将一个密匙装入超过 E^2PROM 字节地址 1FFhex 的位置。

7 FIFO 缓冲区

7.1 概述

MF RC500 具有一个 8×64 位的 FIFO 缓冲区，它起到一个并行-并行转换器的作用。它缓冲微处理器和 MF RC500 之间输入和输出的数据流。这样最高可以处理 64 字节长的数据流而不需要考虑时限。

7.2 访问 FIFO 缓冲区

7.2.1 访问规则

FIFO 缓冲区输入和输出数据总线连接到 FIFOData 寄存器。对该寄存器的写操作会将一个字节存入 FIFO 缓冲区并将内部 FIFO 缓冲区写指针加一。对该寄存器的读操作显示保存在 FIFO 缓冲区的内容并将 FIFO 缓冲区读指针加一。写和读指针之间的距离可通过读 FIFOLength 寄存器获得。

当微处理器启动一个命令，MF RC500 可以在命令处理时根据这条命令访问 FIFO 缓冲区。物理上只存在一个 FIFO 缓冲区，可用于输入和输出指向。因此微处理器必须考虑到不要采用不确定的方式访问 FIFO 缓冲区。

下表所示为在命令处理时对 FIFO 访问的汇总：

有效命令	微处理器允许的操作		备注
	写 FIFO	读 FIFO	
StartUp	-	-	
Idle	-	-	
Transmit		-	
Receive	-		
Transceive			微处理器必须知道命令的实际状态(发送还是接收)
WriteE2		-	
ReadE2			微处理器必须准备变量,只允许读出
LoadKeyE2		-	
LoadKey		-	
Authent1		-	
Authent2	-	-	
LoadConfig		-	
CalcCRC		-	

表 7-1　FIFO 缓冲区允许的操作

7.3　控制 FIFO 缓冲区

除了对 FIFO 缓冲区进行读和写之外，FIFO 缓冲区的指针可以通过置位 FlushFIFO 复位。结果就是 FIFOLength 变为 0，FIFOOvfi 清零，实际保存的值再不能被访问。FIFO 缓冲区可装入其它的 64 个字节。

7.4　FIFO 缓冲区的状态信息

微处理器可以获得关于 FIFO 缓冲区状态的数据：

- 已经保存在 FIFO 缓冲区中字节数：FIFOLength
- 警告，FIFO 缓冲区已经很满：HiAlert
- 警告，FIFO 缓冲区已经很空：LoAlert
- 指示，尽管 FIFO 缓冲区已满，仍写入字节：FIFOOvfl　FIFOOvfl 只能通过设置位 FlushFIFO 清零。

MF RC500 可产生一个中断信号

- 如果 LoAlertIRq 设为 1，当 LoAlert 变为 1 时将使脚 IRQ 有效
- 如果 HiAlertIRq 设为 1，当 HiAlert 变为 1 时将使脚 IRQ 有效

如果可以保存在 FIFO 缓冲区内的字节数只有 WaterLevel 或更少，标志 HiAlert 置位。由下列等式产生：

$$HiAlert = (64 - FIFOLength) \leq WaterLevel$$

如果实际保存在 FIFO 缓冲区内的字节数只有 WaterLevel 或更少，标志 LoAlert 置位。由下列等式产生：

$$LoAlert = FIFOLength \leq WaterLevel$$

7.5 FIFO 缓冲区寄存器概述

下表所示为与 FIFO 缓冲区有关的标志，以字母顺序排列：

标志	寄存器	地址 寄存器,位的位置
FIFOLength	FIFOLength	0x04，位 6~0
FIFOOvfl	ErrorFlag	0x0A，位 4
FlushFIFO	Control	0x09，位 0
HiAlert	PrimaryStatus	0x03，位 1
HiAlertIEn	InterruptIEn	0x06，位 1
HiAlertIRq	InterruptIRq	0x07，位 4
LoAlert	PrimaryStatus	0x03，位 0
LoAlertIEn	InterrupIEn	0x06，位 0
LoAlertIRq	InterruptIRq	0x07，位 0
WaterLevel	FIFOLevel	0x29，位 5~0

表 7-2 与 FIFO 缓冲区相关的寄存器

8 中断请求系统

8.1 概述

MF RC500 通过在 PrimaryStatus 寄存器中设置 IRq 位指示一定的事件并使 IRQ 脚有效。IRQ 脚上的信号可用于具有中断处理能力的微处理器产生中断。这就使微处理器的软件更为有效。

8.1 中断源概述

下表所示为完整的中断标志、相关的中断源和置位的条件。中断标志 TimerIRq 指示由定时器单元产生的中断。TimerIRq 置位的条件是定时器减到 0 或者 TPreLoad 值(如果 TAutoRestart 使能)。

TxIRq 位指示由不同中断源产生的中断。如果发送器有效而且状态从发送数据到帧结束，发送器单元自动将中断标志位置位。CRC 协处理器在处理完 FIFO 缓冲区内所有数据后将 TxIRq 置位，这通过标志 CRCReady=1 指示。如果 E^2PROM 编程结束，TxIRq 置位，通过位 E2Ready=1 指示。

当接收数据的结束被检测到时通过 RxIRq 标志指示中断。如果命令结束并且命令寄存器的内容变为 Idle 时，标志 IdleIRq 置位。如果 HiAlert 位置 0，标志 HiAlertIRq 置位，这表示 FIFO 缓冲区已到达由位 WaterLevel 指示的边界。见章节 7.4。

如果 LoAlert 位置 0，标志 LoAlertIRq 置位，这表示 FIFO 缓冲区已到达由位 WaterLevel 指示的边界。见章节 7.4。

中断标志	中断源	自动置位
TimerIRq	定时器单元	定时器从 1 计数到 0
TxIRq	发送器	数据流，发送到卡，结束
	CRC 协处理器	FIFO 缓冲区所有数据都已处理
	E^2PROM	FIFO 缓冲区所有数据都已编程
RxIRq	接收器	数据流，从卡接收，结束
IdleIRq	命令寄存器	命令执行完成
HiAlertIRq	FIFO 缓冲区	FIFO 缓冲区变满
LoAlertIRq	FIFO 缓冲区	FIFO 缓冲区变空

表 8-1 中断源

8.2 中断请求处理的实现

8.2.1 控制中断及其状态

MF RC500 通过置位 InterruptRq 寄存器中的相应位将中断请求源告知微处理器。每个中断相关的请求位可由 InterruptEn 寄存器中的中断使能位屏蔽。

寄存器	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 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

如果任意一个中断请求位置位(表示中断请求被挂起)并且对应的中断使能位置位，PrimaryStatus 寄存器中的状态标志 IRq 置位。此外，不同的中断源可同时设为有效。因此所有的中断请求位都相“或”并连接到标志 IRq 再到脚 IRq。

8.2.2　访问中断寄存器

中断请求位由 MF RC500 内部状态机自动置位。另外，微处理器可以对其进行访问以置位或清零。

InterruptRq 和 InterrupEn 寄存器的特殊实现允许改变其中单个位而不影响其它位。如果指定的中断寄存器要设置为 1，Setlxx 必须置位并同时将指定的位置位。反过来，如果指定的中断标志要清零，将 0 写入 Setlxx，同时中断寄存器的指定地址必须设置为 1。如果在设置或清零过程中不改变一个位的内容，必须将指定的位地址写入 0。

例：将 3Fhex 写入 InterruptRq 寄存器，由于 SetIRq 清零而其它位置位，这将清零所有位。写入 81hex 将位 LoAlertIRq 置位并保留其它位不变。

8.3　管脚 IRQ 的配置

状态标志 IRq 的逻辑电平通过管脚 IRQ 表现出来。此外，管脚 IRQ 的信号也可由 IRQPinconfig 寄存器的位进行控制：

- IRQInv：如果置 0，IRQ 脚上的信号等于 IRq 位的逻辑电平；如果置 1，IRQ 脚上的信号与 IRq 位的逻辑电平相反
- IRQPushPull：如果置 1，IRQ 脚具有标准的 CMOS 输出特性；否则为开漏输出，而且必须接一个外部上拉电阻使其为高电平。

注：在复位过程中（见 11.2），IRQInv 置 1 而 IRQPushPull 置 0。这使 IRQ 脚为高阻态。

8.4　中断请求系统寄存器概述

下表所示为中断请求系统的相关标志，以字母顺序排列。

标志	寄存器	地址 寄存器,位的位置
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　与中断请求系统有关的寄存器

9　定时器单元

9.1　概述

MF RC500 内部有一个定时器。它由片内 13.56MHz 时钟驱动。微处理器可使用该时钟管理与定时有关的任务。定时器单元可配置为以下几种方式之一：

- 超时计数器
- 看门狗
- 停止监视
- 可编程单次触发
- 周期触发

定时器单元可用于测量两个事件的时间间隔或指示一个发生在特定时间后的特定事件。定时器可由事件触发，但是定时器自身不会影响任何内部事件（例如，数据接收时的定时溢出不影响接收的处理）。此外，几个定时器相关的标志置位并可用于产生中断。

9.2　定时器单元的实现

9.2.1　方框图

下面所示为定时器模块的方框图：

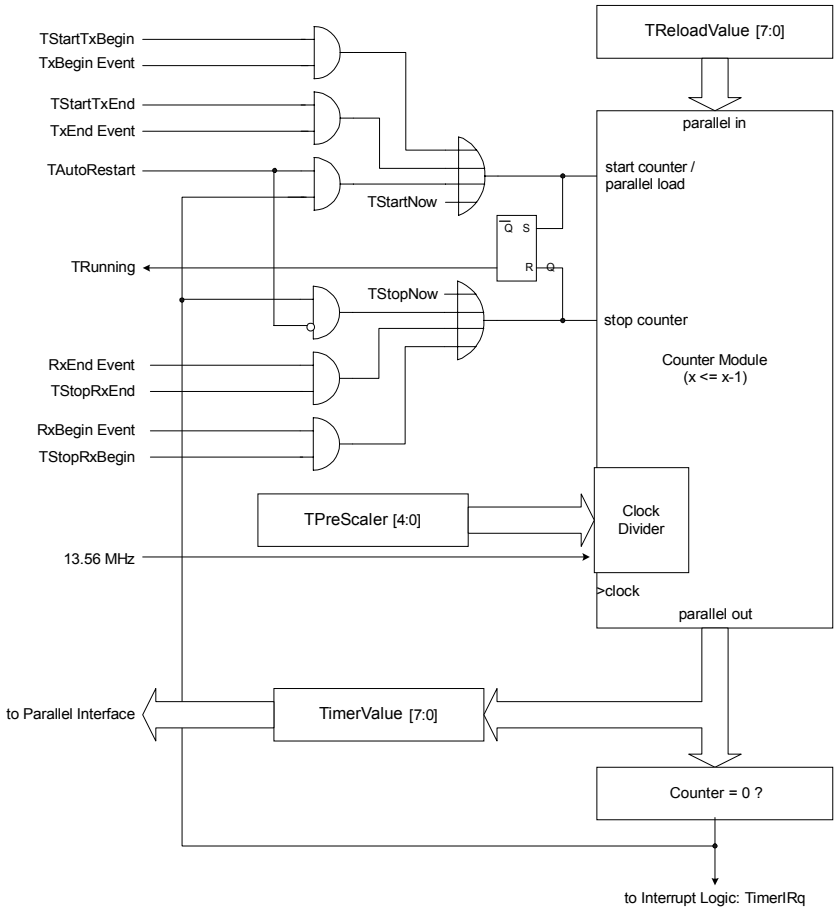


图 9-1　定时器模块方框图

定时器单元是以几个带有启动和停止计数器的使能标志的事件组合来设计的。例如，将位 TStartTxBegin 置位使能利用定时器单元对数据接收进行控制。此外，第一个接收位由 TxBeginEvent 指示。该组合在定义 TReloadValue 后启动计数器。如果计数器值为零或定义的停止事件发生了，定时器自动停止。

9.2.2 控制定时器单元

定时器单元的主要部分是一个倒计数器。只要倒计数器不等于零，它就在每个定时器时钟减一。如果 TAutoRestart 使能，定时器不会减到零。减到 1 时，定时器在下个时钟将 TimerReload 值重新装入。

定时器通过将 TimerReload 寄存器值装入计数器模块立即启动。可由下列的事件之一触发：

- 第一个位发送到卡（TxBegin 事件）且 TStartTxBegin 为 1
- 最后一个位发送到卡（TxEnd 事件）且 TStartTxEnd 为 1
- 位 TStartNow 设置为 1（通过微处理器）

注：每个启动事件都将 TimerReload 寄存器的值重装入定时器。因此，定时器是重复触发的。定时器可配置为下列事件之一停止：

- 来自卡第一个有效位的接收（RxBegin 事件）且 TStopRxBegin 设置为 1
- 来自卡最后一个有效位的接收（RxEnd 事件）且 TStopRxEnd 设置为 1
- 计数器模块减到 0 且 TAutoResetstart 设置为 1
- 位 TStopNow 设置为 1（通过微处理器）

装入新值，例如，将 0 装入 TimerReload 寄存器不会立即影响计数器，它只有到下次启动事件才会影响计数器。因此，TimerReload 寄存器即使在定时器单元计数当中也可改变。改变 TimerReload 寄存器的结果将在下次启动事件后显现出来。

如果置位 TStopNow 将计数器停止，TimerIRq 不会发出信号。

9.2.3 定时器单元时钟和周期

定时器单元的时钟通过一个可编程分频器由 13.56MHz 片内时钟驱动。时钟的选择由 TPreScaler 寄存器完成，它根据下面的公式定义定时器单元频率：

$$T_{TimerClock} = \frac{1}{f_{TimerClock}} = \frac{2^{TPreScaler}}{13.56MHz}$$

TPreScaler 寄存器的可能值从 0 到 21。时间 $T_{TimerClock}$ 的最小值大约为 74ns，最大 150ms。上次启动事件所经过的时间以下式计算：

$$T_{Timer} = \frac{TReLoadValue - TimerValue}{f_{TimerClock}}$$

最小时间约为 74，最大约为 40s。

9.2.4 定时器单元的状态

SeconddaryStatus 寄存器中的 TRunning 位显示定时器的当前状态。任何已配置的启动事件以 TReloadValue 启动定时器并将状态标志 TRunning 置 1，任何已配置的停止事件停止定时器并将 TRunning 重新置 0。只要状态标志 TRunning 置 1，TimerValue 寄存器就在下个定时器时钟改变。

实际的定时器单元内容只能通过 TimerValue 寄存器读出。

9.3 定时器单元的使用

9.3.1 超时和看门狗

在设定 TRloadValue 启动定时器后，定时器单元以一个特定的事件启动并减少 TimerValue 寄存器的值。如果发生特定的停止事件，例如从卡接收到一个位，定时器单元停止（不产生中断）。

另一方面，如果没有产生停止事件，例如卡没有在期望的时间内响应，定时器值减到 0 并产生一个定时器中断请求。这就告知微处理器期望的事件没有在给定的时间 T_{Timer} 内发生。

9.3.2 停止监视

在特定启动和停止事件之间的 T_{Timer} 可以由微处理器通过 MF RC500 定时器测量。定时器装入 TReloadValue 并启动计数。如果定义的停止事件发生，定时器停止。时间可由下式计算得到：

$$\Delta T = (TReload_{value} - Timer_{value}) * T_{Timer}$$

如果定时器没有减到 0。

9.3.3 可编程单次触发定时器

微处理器启动定时器单元并等待定时器中断。在指定的时间 T_{Timer} 后产生中断。

9.3.4 周期性触发

如果微处理器置位 TAutoReset，每经过时间 T_{Timer} 周期性产生中断请求。

9.4 定时器单元寄存器概述

下表所示为定时器单元相关的寄存器，以字母顺序排列

标志	寄存器	地址
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

10 节电模式

10.1 硬件掉电

硬件掉电通过在脚 RSTPD 置高电平使能。这将关闭所有的内部电流消耗，包括振荡器。所有的数字输入缓冲区都独立于输入端并由内部定义(RSTPD 除外)。输出脚在固定的值冻结。

如下表所示：

符号	脚	类型	描述
OSCIN	1	I	不独立于输入，拉低到 AVSS
IRQ	2	O	高阻
MFIN	3	I	独立于输入
MFOUT	4	O	低
TX1	5	O	高
TX2	7	O	低
NWR	9	I	独立于输入
NRD	10	I	独立于输入
NCS	11	I	独立于输入
D0~D7	13~20	I/O	独立于输入
ALE	21	I	独立于输入
A0	22	I/O	独立于输入
A1	23	I	独立于输入
A2	24	I	独立于输入
AUX	27	O	高阻

RX	29	I	不变
VMID	30	A	拉低到 AVDD
RSTPD	31	I	不变
OSCOUT	32	O	高

表 10　硬件掉电时管脚的信号

10.2　软件掉电

将 Control 寄存器中的位 PowrDown 置位后立即进入该模式.所有的内部电流消耗都关闭(包括振荡器缓冲区)。

与硬件掉电不同，数字输入缓冲区不独立于输入端，但保持其功能。数字输出脚不改变状态。重新设置 Control 寄存器中的 PowerDown 后，需要 512 个时钟才能退出软件掉电模式。这通过 PowerDown 位自身来指示。对其重新设置不会立即清除它，但是当离开软件掉电模式时，它由 MF RC500 自动清零。

注：如果使用内部振荡器，必须考虑到它由 AVDD 供电，将会花费一定的时间等待振荡器稳定下来。

10.3　待机模式

通过置位 Control 寄存器中的 StandBy 位立即进入该模式。所有内部电流消耗都关闭（包括内部数字时钟缓冲区但不包括振荡器缓冲区）。

与硬件掉电模式不同，数字输入缓冲区没有与输入端隔离而是保持其功能。数字输出脚不改变其状态。

与软件掉电模式不同，振荡器不需要时间唤醒。

在重新设置 Control 寄存器中的 StandBy 位后，需要脚 OSCIN 上的 4 个时钟退出待机模式。这通过 StandBy 位自身来指示。重新设置它不会立即清除它，但是当退出待机模式时，它由 MF RC500 自动清零。

10.4　接收器掉电

当接收器不需要时将其关闭以节电并在从卡接收数据之前重新打开。这通过 RxAutoPD 置位自动完成。如果将其设置为 0，接收器持续打开。

11　启动阶段

启动时执行的阶段如下图所示

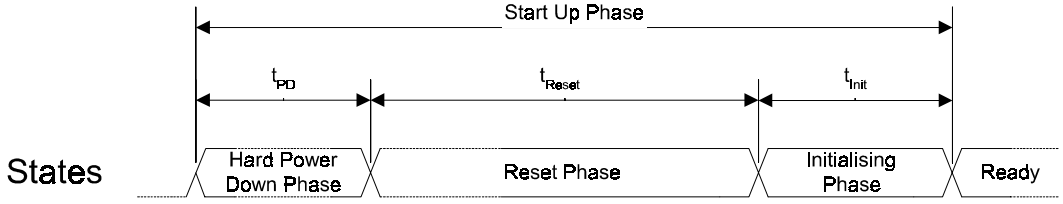


图 11-1　启动过程

11.1　硬件掉电状态

下列情况可激活硬件掉电模式：

- 由脚 DVDD 上电导致的上电复位（在 DVDD 低于数字复位阀值时激活）
- 由脚 AVDD 上电导致的上电复位（在 AVDD 低于模拟复位阀值时激活）
- 脚 RSTPD 上的高电平（在脚 RSTPD 为高时激活）

11.2　复位阶段

在硬件掉电阶段之后自动跟随复位阶段。这将花费 512 个时钟。在复位阶段中，一些寄存器位由硬件预置。其各自的复位值见每个寄存器的描述（见 5.2）。

注：如果使用内部振荡器，必须考虑到它由 AVDD 供电，将会花费一定的时间等待振荡器稳定下来。

11.3 初始化阶段

在复位阶段后自动跟随初始化阶段。这将花费 128 个时钟。在初始化阶段中 E2PROM 块 1 和 2 的内容复制到寄存器 10~2FH（见 6.3）。

注：在产品测试时，MF RC500 以默认配置值初始化。这将微处理器配置器件的错误减到最小。

11.4 初始化并行接口类型

由于不同的微处理器接口类型（见 4.3）具有不同的连接，所以执行一个特定时序使能检测正确的微处理器类型并同步微处理器和 MF RC500 的启动。

在整个启动阶段中，Command 值读出为 3FH。在初始化阶段的结束，MF RC500 自动输入 Idle 命令。结果 Command 值变为 00H。

为了确保对微处理器接口类型正确的检测，必须执行下面的时序：

- 读 Command 寄存器，直到 Command 的 6 位值为 00H。内部初始化阶段此时结束，MF RC500 准备接收控制
- 将 80H 写入 Page 寄存器以初始化微处理器接口
- 读 Command 寄存器。如果该值为 00H，微处理器接口初始化成功

在接口初始化之后，通过将 0X00 写入页寄存器可激活线性地址模式。

12 振荡器电路

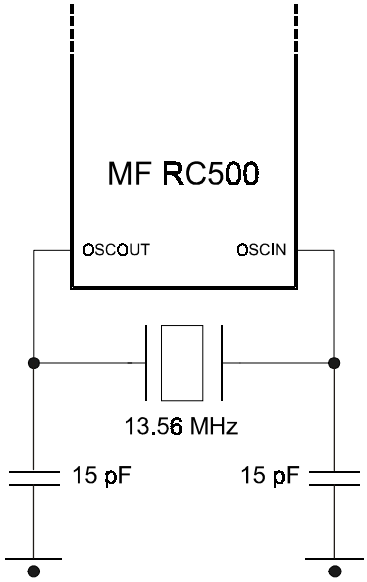


图 12-1 石英晶振的连接

提供给 MF RC500 的时钟作为同步系统编码器和解码器的时基。因此时钟的稳定性是实现正确操作的一个重要因素。要获得最佳性能，时钟的抖动必须尽可能小。可通过使用带有推荐电路的内部时钟缓冲区来实现。如果使用外部时钟源，时钟信号必须输入到 OSCIN。这种情况下要特别注意时钟的占空比和抖动，时钟的质量必须进行鉴定以符合在 20.5.3 中的规格。

备注：建议不使用外部时钟源。

13 发送器管脚 TX1 和 TX2

TX1 和 TX2 上传递的信号是由包络信号调制的 13.56MHz 能量载波。只需要很少的用于匹配和滤波（见 18）的无源元件就可以直接驱动天线。因此，输出电路设计成具有非常低的内阻。TX1 和 TX2 的信号可通过 TxControl 寄存器进行控制。

13.1 TX1 和 TX2 的配置

TX1 几种可能的配置见下表：

TxControl 中的寄存器配置	包络	TX1 上的信号
TX1RFEn		
0	X	低
1	0	低
1	1	13.56MHz 能量载波

表 13-1 管脚 TX1 的配置

TX2 几种可能的配置见下表：

TxControl 中的寄存器配置			包络	TX2 上的信号
TX2RFEn	TX2CW	InvTX2		
0	X	X	X	低
0	0	0	0	低
			1	13.56MHz 能量载波
		1	0	高
			1	13.56MHz 能量载波 相对 TX1 有 180º 相移
	1	0	X	13.56MHz 能量载波
		1	X	13.56MHz 能量载波 相对 TX1 有 180º 相移

表 13-2 管脚 TX1 的配置

13.2 操作距离 VS. 功耗

通过使用不同天线匹配的电路和/或改变天线驱动电压脚 TVDD 的电压，用户可能会发现在最大可操作距离和功耗之间有一个折衷方案。

13.3 脉冲宽度

包络传递要发送到卡的据信号信息。根据 Miller 码对数据信号进行编码来完成。此外，Miller 编码信号的每个暂停编码为一定长度的脉冲。脉冲宽度可通过 ModWidth 寄存器进行调节。计算如下：

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

此处 fc=13.56MHz

14 接收器电路

14.1 概述

MF RC500 集成了一个正交调制电路，该电路从输入到 RX 脚的 13.56MHz ASK 调制信号中解析出 ISO14443-A 副载波信号。正交调制器使用两个不同的时钟 Q-和 I-时钟，它们之间的相位差为 90º 得到的 。副载波信号经过放大，滤波然后输入到相关性电路。求出相关性结果，数字化后输入到数字电路。

所有的处理单元都可进行调节以获得最优性能。

14.2 方框图

图 14-1 所示为计数器电路方框图。接收处理包括几个步骤。首先完成 13.56MHz 载波信号的正交解调。要获得最优性能，建议使用时钟 Q 自动校准（见 14.3.1）。解调的信号由一个可调节的放大器进行放大。一个相关性电路计算期望的和接收的数据之间的相似度。在计算和数字化电路中有效位被检测出来然后送到 FIFO 寄存器。在此电路中可进行几个调节步骤。

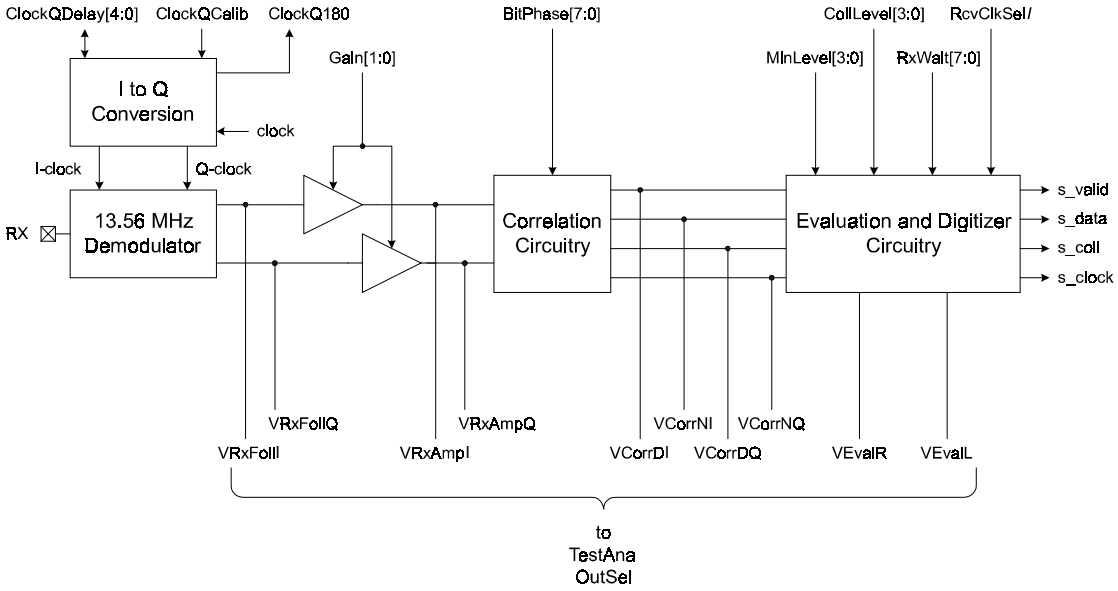


图 14-1　接收器电路方框图

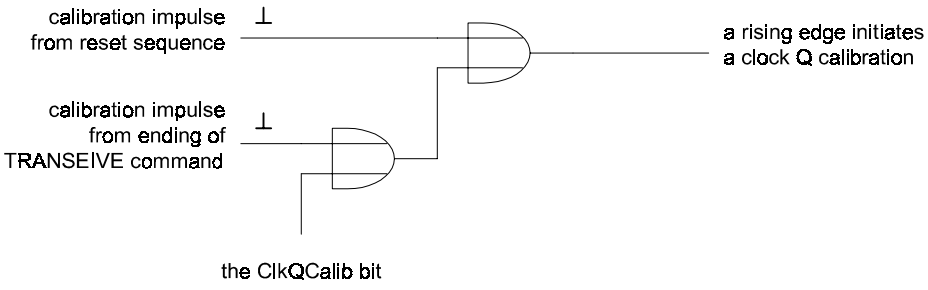
用户可以观察通过接收器的信号，如上图所示。使用 TestAnaSelect 寄存器一次可输出一个信号到 AUX 脚，详见 19.3。

14.3　对接收器进行操作

通常情况下，在启动初始化文件中的默认设定适用于 MF RC500 与 MIFARE 卡之间的数据通信。但是在有些环境下特定的用户设定会有更好的性能。

14.3.1　时钟 Q 自动校准

接收器的正交解调产生一个相位信号 I 时钟和一个 90º　　Q 时钟. 为了得到最佳的解调性能, Q-和 I-时钟的相位差必须为 90º　　MF RC500 的复位阶段之后自动进行校准处理。也可以在每次收发命令的结束进行自动校准。要实现这一点，必须将 ClkQCalib 位设置为 0。将该位配置为 1 将禁止所有的自动校准（复位阶段后除外）。也可以通过软件启动一次自动校准，将位 ClkQCalib 从 0 变为 1。



注：Q 时钟的自动校准过程最大为 65 个振荡周期，大约为 4.7us。

ClkQDelay 的值与 Q-和 I-时钟之间的相移成比例。状态标志 ClkQ180Deg 表示 Q-和 I-时钟之间的相移大于 180º　　。

注：

- 启动配置文件使能在复位后的 Q 时钟自动校准。
- 当 ClkQCalib 为 1，不执行自动校准。将该位保持 1 用于永远禁止自动校准。
- 可通过微处理器将数据写入 ClkQDelay。目的是禁止自动校准和由软件来预置延迟。但要注意，由软件配置延迟值要求在这之前 ClkQCalib 已经置位并且至少经过 4.8us。每次写入延迟值时都必须将 ClkQCalib 位设置为 1。如果 ClkQCalib 为 0，已配置的延迟值将会被下次自动校准所覆盖。

14.3.2 放大器

解调信号必须由可变的放大器放大以实现最佳性能。放大器的增益可以通过寄存器位 Gain[1:0]调节，见下表：

寄存器设定	增益因素 （仿真结果）	增益因素[dB] （仿真结果）
0	22	26.9
1	35	30.9
2	82	38.3
3	130	42.2

表 14-1 内部放大器的增益因素

14.3.3 相关性电路

相关性电路计算接收到的和期望的信号之间的匹配程度。输出是对接收信号中所期望的信号的振幅进行测量。Q-和 I-信道都会执行。相关器对每个输入信道传递两路输出，结果总共有 4 路输出信号。

为了优化性能，相关性电路需要相位信息用于来自卡的信号。该信息必须由微处理器利用寄存器 BitPhase[7:0]来定义。该值定义了发送器和接收器之间的相位是 $t_{BitPhase}$＝1/13.56MHz 的倍数。

14.3.4 计算和数字化电路

对每半位 Manchester 编码信号都计算相关性结果。计算和数字化电路根据两个半位的信号强度确定当前位是否有效。如果有效，该位的值或当前位间隔是否包含一个位冲突。

为了以优化的方式完成该任务，用户可选择下列级别：

- MinLevel：定义认为有效的较强半位信号的最小信号强度
- CollLevel：定义较弱半位 Manchester 编码信号必须超过的最小信号强度，用于产生一个位冲突。如果信号强度低于该值，就可以明确的决定是 1 还是 0。CollLevel 定义相对于较强半位幅值的最小信号强度。

在数据发送之后，一定时间内不允许卡发送它的响应，在 ISO14443 标准中这叫做帧保护时间。这段时间的长度在 RxWait 寄存器中设置。RxWait 寄存器定义在数据发送到卡的若干个位时间后打开接收器。

如果寄存器位 RcvClkSelI 置位，I-时钟作为相关器和计算电路的时钟。如果置 0 则使用 Q-时钟。

注：推荐使用 Q-时钟。

15 串行数据转换

15.1 概述

MF RC500 包含两个主要模块。一个数字电路（包括状态机、编码器和解码器逻辑等）和一个模拟电路（包括调制器、天线驱动器、接收器和放大电路）。这两个模块的接口可配置为这样一种方式，即接口信号可输出到管脚 MFIN 和 MFOUT。

这样的拓扑结构支持将 MF RC500 的模拟部分与其它器件的数字部分连接。

15.2 方框图

图 15-1 所示为串行数据转换。其中有三个不同的转换开关可以使 MF RC500 实现不同的配置。

串行数据转换还可在设计同步时使用或用于测试发送和接收的数据。19.2 讲述了在串行数据转换处的模拟测试信号和测量。

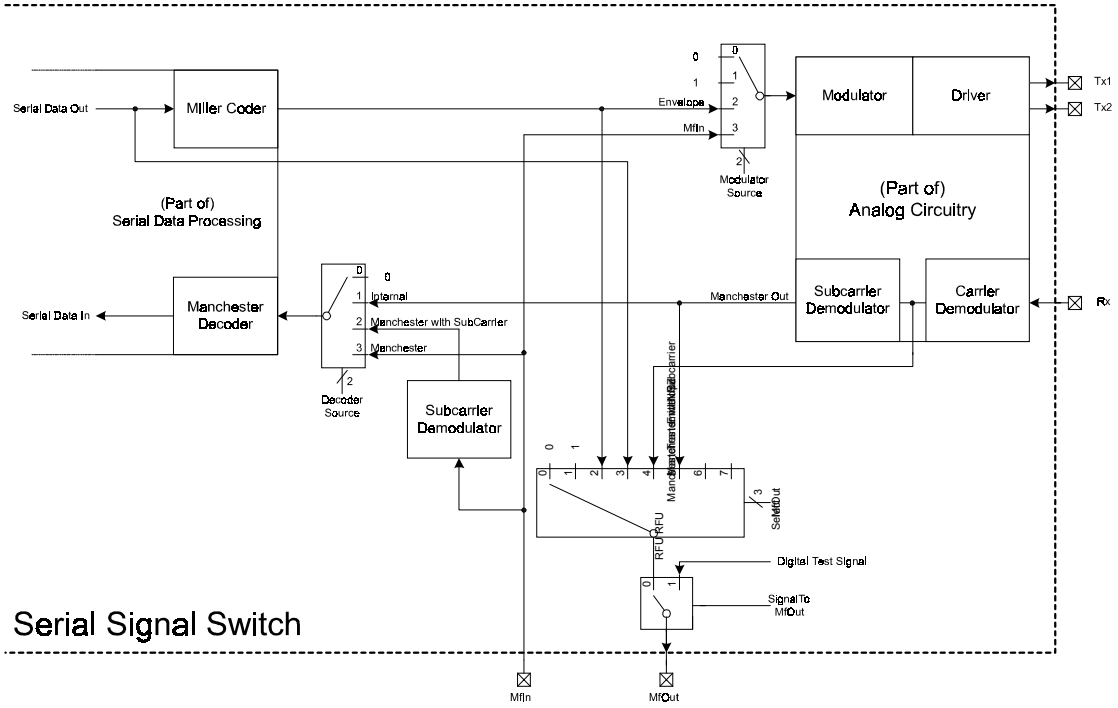


图 15-1　串行数据转换

下面的章节讲述了用于配置和控制串行数据转换的相关寄存器。

15.3　与串行数据相关的寄存器

标志 DecoderSource 以下列方式定义用于内部 Manchester 解码器的输入信号：

DecoderSource	解码器的输入信号
0	恒为 0
1	模拟部分的输出。此为默认配置
2	直接连接到 MFIN 脚，由 Manchester 编码信号调制的 847.5kHz 副载波信号除外
3	直接连接到 MFIN，Manchester 编码信号除外

ModulatorSource 定义调制发送 13.56MHz 能量载波的信号。调制信号驱动 TX1 和 TX2。

ModulatorSource	调制器的输入信号
0	恒为 0（脚 TX1 和 TX2 的能量载波关闭）
1	恒为 1（脚 TX1 和 TX2 持续传递的能量载波
2	来自内部编码器的调制信号（包络）。此为默认配置
3	直接连接到 MFIN，Miller 脉冲编码信号除外

MFOUTSelect 选择输出到脚 MFOUT 的信号。

ModulatorSource	调制器的输入信号
0	恒为低
1	恒为高
2	来自内部编码器的调制信号（包络）
3	要发送的串行数据流（与 MFOUTSelect=2 相同，单不由 Miller 脉冲编码器编码）
4	接收器电路的输出信号（卡调制信号重新产生和延迟）
5	副载波调制器的输出信号（Manchester 编码卡信号）
6	RFU
7	RFU

注：要使用 MFOUTSelect，测试信号控制位 SignalToMFOUT 必须为 0。

15.4 MFIN 和 MFOUT 的使用

15.4.1 有源天线的概念

MF RC500 模拟电路可经由脚 MFIN 和 MFOUT 使用。下面的寄存器必须进行一些设定:

寄存器	值	信号	MF RC500 脚
ModulatorSource	3	Miller 脉冲编码	MFIN
MFOUTSelect	4	带副载波的 Manchester 编码	MFOUT
DecoderSource	X	-	-

另一方面,MF RC500 数字电路

可经由脚 MFIN 和 MFOUT 使用。下面的寄存器必须进行一些设定:

寄存器	值	信号	MF RC500 脚
ModulatorSource	X	-	-
MFOUTSelect	2	Miller 脉冲编码	MFOUT
DecoderSource	2	带副载波的 Manchester 编码	MFIN

两个 MF RC500 器件(按照上面描述的方式配置)可通过脚 MFOUT 和 MFIN 直接相连。

15.4.2 驱动两个 RF 部分

有可能将一个无源天线连接到脚 TX1,TX2 和 RX（经由合适的滤波器和匹配电路）并同时将一个有源天线连接到脚 MFOUT 和 MFIN。在这样的配置下，两个 RF 部分可以由一个微处理器驱动（逐个地）。

16 MF RC500 命令集

16.1 概述

MF RC500 的状态由可执行特定的命令集的内部状态机决定。这些命令可通过将相应的命令代码写入 Command 寄存器来启动。处理一个命令所需要的变量和/或数据主要通过 FIFO 缓冲区进行交换。

16.2 常规的状态

- 每个需要数据流（或数据字节流）作为输入的命令会立即处理它在 FIFO 缓冲区中发现的数据。
- 每个需要一定数目变量的命令只会当它从 FIFO 缓冲区接收到正确数目的变量时才会启动处理。
- FIFO 缓冲区在命令启动时不会自动清零。因此，可以将命令变量和/或数据字节写入 FIFO 缓冲区，然后再启动命令
- 每个命令（StartUp 命令除外）都可被微处理器通过写入新命令所中断。例如：Idle 命令。

16.3 MF RC500 命令汇总

命令	代码	动作	经由 FIFO 通过的变量和数据	经由 FIFO 返回的数据	所在章节
StartUp	3FH	运行复位和初始化阶段 注:该命令不能通过软件,只能通过上电或硬件复位启动	-	-	16.3.2
Idle	00H	无动作:取消当前执行的命令	-	-	16.3.3
Transmit	1AH	将数据从 FIFO 缓冲区发送到卡	数据流	—	16.4.1
Receive	16H	启动接收器电路 注：在接收器实际启动之前，状态机经过寄存器 RxWait 配置的时间后才结束等待。 注：由于该命令与 Transmit 命令无时序的关系，因此可以只用于测试。	-	数据流	16.4.2
Transceive	1EH	将数据从 FIFO 发送到卡并在发送后自动启动接收器。 注：在接收器实际启动之前，状态机经过寄存器 RxWait 配置的时间后才结束等待。 注：该命令是发送和接收的组合	数据流	数据流	16.4.3
WriteE2	01H	从 FIFO 缓冲区获得数据并写入内部 E^2PROM	起始地址低字节 起始地址高字节 数据字节流	-	16.5.1
ReadE2	03H	从内部 E^2PROM 读出数据并将其放入 FIFO 缓冲区 注：密匙不能被读出	起始地址低字节 起始地址高字节 数据字节个数	数据流	16.5.2
LoadKeyE2	0BH	将一个密匙从 E^2PROM 复制到密匙缓冲区	起始地址低字节 起始地址高字节	-	16.8.1
LoadKey	19H	从 FIFO 缓冲区读出密匙字节并将其放入密匙缓冲区 注：密匙必须以指定的格式准备	字节 0（低） 字节 1 … 字节 11（最高）	-	16.8.2
Authent1	0CH	执行 Crypto1 卡验证的第一部分	卡的 Auth 命令 卡的模块地址 卡的序列号最低字节 卡的序列号字节 1 卡的序列号字节 2 卡的序列号最高字节	-	16.8.3
Authent2	14H	使用 Crypto1 算法执行卡验证的第二部分	-	-	16.8.4
LoadConfig	07H	从 E^2PROM 读取数据并初始化 MF RC500 寄存器	起始地址低字节 起始地址高字节	-	16.6.1
CalcCRC	12H	启动 CRC 协处理器 注：CRC 计算结果可从寄存器 CRCResultLSB 和 CRCResultMSB 中读出。	数据字节流	-	16.6.2

图 16-1 MF RC500 命令汇总

16.3.1 基本状态

16.3.2 STARTUP 命令 3FH

命令	代码	动作	变量和数据	返回的数据
StartUp	3FH	运行复位和初始化阶段 注:该命令不能通过软件,只能通过上电或硬件复位启动	-	-

StartUp 命令运行复位和初始化阶段。不需要返回任何数据。它不能由微处理器启动，而是在下列事件之后自动启动：

- 脚 DVDD 上电引起的上电复位
- 脚 AVDD 上电引起的上电复位
- 脚 RSTPD 的下降沿

复位阶段通过异步复位定义一定的寄存器位。初始化阶段用取自 E^2PROM 的值定义一定的寄存器。当 StartUp 命令结束时自动输入 Idle 命令。

注：

- 当 MF RC500 正在执行 StartUp 命令时，微处理器不能对 MF RC500 进行写操作。为了确认这一点，微处理器应当查询 Idle 命令以确认初始化阶段的结束。（见 11.4）
- 当 StartUp 命令有效时，只能读出 MF RC500 的页 0。
- StartUp 命令不能被微处理器中断

16.3.3 Idle 命令 00H

命令	代码	动作	变量和数据	返回的数据
Idle	00H	无动作，取消当前执行的命令	-	-

Idle 命令将 MF RC500 切换到非活动状态。在 Idle 状态中它等待下一个命令。该命令不需要或返回任何数据。器件在完成一个命令时自动进入 Idle 状态。该情况下，MF RC500 同时通过置位 IdleIRq 初始化中断请求。Idle 命令可由微处理器触发用于停止其它命令的执行（StartUp 命令除外）。这种情况下不产生 IdleIRq。

备注：使用 Idle 命令停止一个命令不会清除 FIFO 缓冲区的内容。

16.4 与卡通信的命令

MF RC500 是一个完全兼容 ISO14443-A 的读卡芯片。因此，该芯片的命令集与 MIFARE 专用读卡芯片相比有更大的灵活性和普遍性。下面的章节讲述用于卡通信的命令集，以 MIFARE 相关验证过程结束。

16.4.1 Transmit 命令 1AH

命令	代码	动作	变量和数据	返回的数据
Transmit	1A	将数据从 FIFO 缓冲区发送到卡	数据流	-

Transmit 命令从 FIFO 缓冲区取出数据并将其送到发送器，该命令不返回任何数据。Transmit 命令只能通过微处理器启动。

16.4.1.1 Transmit 命令的工作

要发送数据必须满足下列条件之一：

1. 所有要发送到卡的数据在 Idle 命令有效时写入 FIFO。之后将 Transmit 命令代码写入 Command 寄存器。
   注：可用于最大长度为 64 字节的数据传输
2. 首先将 Transmit 命令代码写入 Command 寄存器，由于 FIFO 中没有可用的数据，命令有效但发送并未触发。当第一个数据字节写入 FIFO 时数据传输开始。为了在 RF 接口上产生一个连续的数据流，微处理器必须及时将下个数据送入 FIFO。
3. 在 Idle 有效时，将要发送到卡的一部分数据写入 FIFO。之后将 Transmit 命令代码写入 Command 寄存器。在 Transmit 命令有效时，微处理器可以将更多数据发送到 FIFO。发送器将其加入发送数据流中。

注：可实现任意长度的数据发送，但要求必须将数据及时送入 FIFO。

当发送器需要发送下一个字节以保持 RF 接口上数据流的连续而 FIFO 缓冲区为空时，Transmit 命令自动终止。这导致内部状态机将状态 Transmit 转为 Idle。

如果对卡的数据发送已结束，MF RC500 置位标志 TxIRq 以通知微处理器。

备注：如果微处理器用 Idle 命令或其它命令覆盖了 Command 寄存器中的发送命令，发送在下个时钟周期立即停止。这将产生与 ISO14443-A 不一致的输出信号。

16.4.1.2 RF 信道冗余和帧

每个发送的帧都包含一个 SOF（帧起始）格式，之后为数据流并由 EOF（帧结束）结尾。这些发送时序的不同阶段可通过观察 PrimaryStatus 寄存器中的 ModemState 进行监控(见 16.4.4)。将 ChannelRedundancy 寄存器中的 TxCRCEn 位置位，计算 CRC 并加到数据流中。CRC 的计算取决于 ChannelRedundancy 寄存器的设定。奇偶校验根据 ChannelRedundancy 寄存器（位 ParityEn 和 ParityOdd）的设定进行处理。

16.4.1.3 位方式帧的发送

发送器可以配置为发送一个不完整的最后字节。要实现这一点，必须将 TxLastBits 设置为一个不为 0 的值。如下图所示：

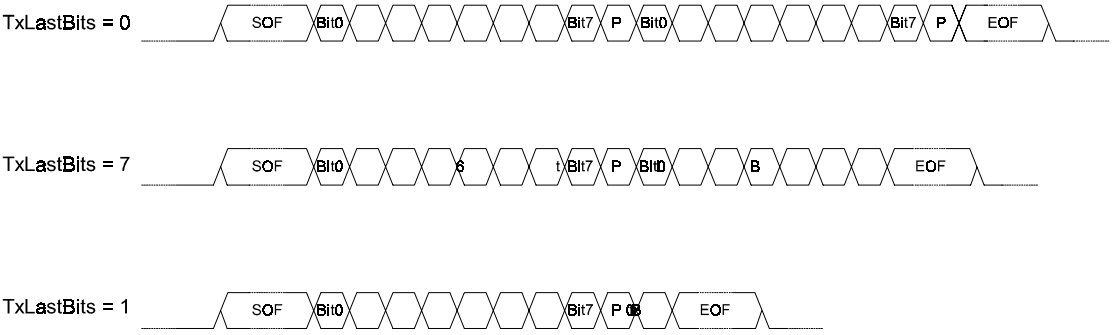


图 16-1 位方式帧的发送

上图显示了 ChannelRedundancy 寄存器中 ParityEn 置位时的数据流。所有完全发送的字节后面都带一个奇偶校验位，但不完整字节后没有奇偶校验位。在发送之后，TxLastBits 自动清零。

注：如果 TxLastBits 不等于 0，CRC 的产生必须禁止。可通过清零 ChannelRedundancy 寄存器中的 TxCRCEn 实现。

16.4.1.4 超过 64 字节帧的发送

要产生一个大于 64 字节的帧，微处理器必须在 Transmit 命令有效时将数据写入 FIFO 缓冲区。当启动实际数据流的最后一位的发送时（检测时间以下图的箭头标注），状态机检测 FIFO 的状态。

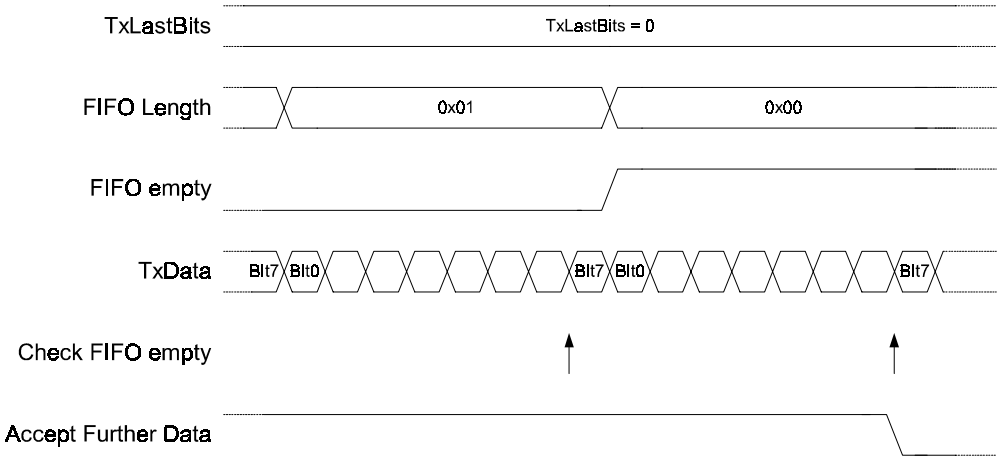


图 16-2 字节方式帧的发送时序

只要内部信号“Accept Further Data”为 1，就可以将更多的数据装入 FIFO。MF RC500 通过 RF 接口将该数据加入发送的数据流中。如果内部信号“Accept Further Data”为 0，发送将会终止。所有在“Accept Further Data”变为 0 后写入 FIFO 的数据将不再发送，但仍然保留在 FIFO 缓冲区。

备注：如果奇偶校验产生使能（Parity 位为 1），奇偶校验位为发送的最后一位。这将信号“Accept Further Data”延长了一个位的时间。

如果 TxLastBits 不为 0，最后一个字节不完全发送，只发送 TxLastBits 中设定的位的数目（从最低位开始）。这样内部状态机必须在较早的点及时检测 FIFO 的状态（如下图所示）。

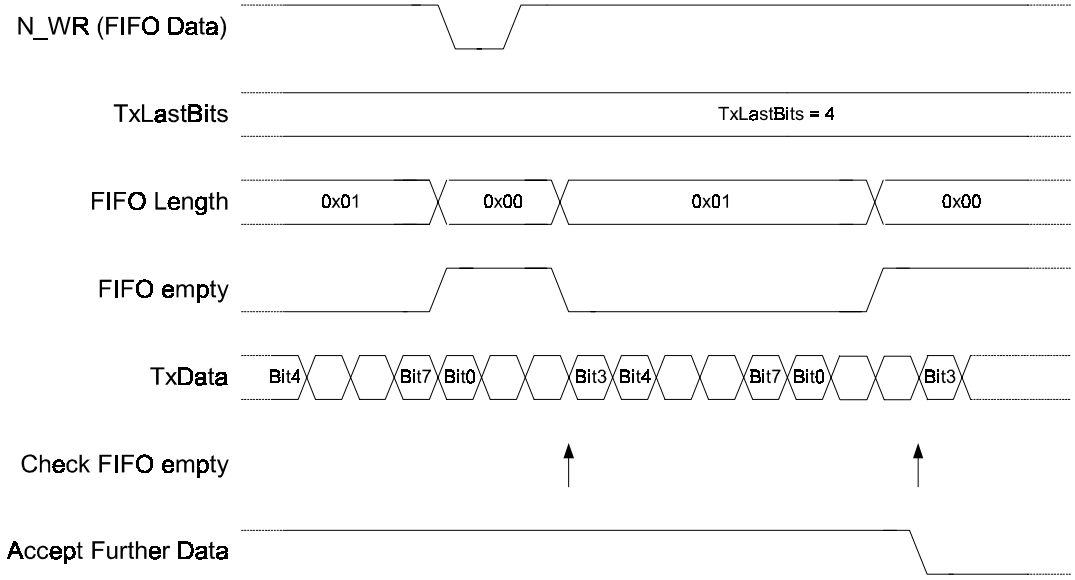


图 16-3　位方式帧的发送时序

由于该例中 TxLastBits＝4，在位 3 发送后，发送停止。如果已配置，帧为带有 EOF 的完整帧。

上图还显示了在检测 FIFO 状态之前对 FIFOData 寄存器有一个写操作.这使得“FIFO empty”重新变为 0 并使“Accept Further Data”保持有效。写入的新字节通过 RF 接口发送。“Accept Further Data”只由“Check FIFO empty”功能所改变。该功能在最后发送位的一个位时间之前检验“FIFO empty”。

帧定义	检验处
8 位带奇偶校验	8^{th} 位
8 位不带奇偶校验	7^{th} 位
x 位不带奇偶校验	$(x-1)^{th}$ 位

16.4.2　Receive 命令 16H

命令	代码	动作	变量和数据	返回的数据
Receive	16	启动接收器电路	-	数据流

Receive 命令启动接收器电路。所有从 RF 接口接收到的数据通过 FIFO 缓冲区返回。Receive 命令可通过微处理器或启动在执行 Transceive 命令时自动启动。

注：由于该命令与 Transmit 命令无时序的关系，因此可以只用于测试。

16.4.2.1　接收命令的工作

在启动 Receive 命令后，内部状态机每个位时钟将 RxWait 寄存器值减一。从 3 到 1，模拟接收器电路准备并启动，当计数器为 0 时，计数器开始监控 RF 接口进入的数据。如果信号强度大于 MinLevel 寄存器中设置的值，接收器开始解码。如果在接收器输入脚 Rx 检测不到信号，解码器停止。解码器通过置位 RxIRq 指示操作的终止。

接收时序的不同阶段都可通过 PrimaryStatus 寄存器中的 ModemState 进行监控(见 16.4.4)。

注：由于计数器值从 3 到 0 用于启动模拟接收器电路，因此 RxWait 的最小值为 3。

16.4.2.2 RF 信道冗余和帧

解码器在每个数据流的开始期望一个 SOF 格式。如果检测到 SOF，它启动串一并转换器并收集进入的数据位。每完成一个字节送入 FIFO。如果检测到 EOF 或信号强度低于 RxThreshold 寄存器中 MinLevel 的值，接收器和解码器停止，执行 Idle 命令并对微处理器产生一个适当的响应（中断请求使能，状态标志置位）。

如果 ChannelRedundancy 寄存器中的位 RxCRCEn 置位，数据流后会有一个 CRC 块。CRC 块可以是一个或者两个字节(取决于 ChannelRedundancy 寄存器中的 CRC8)。

备注：接收到的 CRC 块如果是正确的就不送往 FIFO 缓冲区。这通过将接收到的数据字节移入一个或两个字节（取决于已定义的 CRC）内部缓冲区实现。CRC 块保存在此内部缓冲区。因此 FIFO 缓冲区内所有的数据字节会延迟一到两个字节。

如果 CRC 失败，所有接收到的字节都送入 FIFO 缓冲区（包括错误的 CRC）。

如果 ChannelRedundancy 寄存器中的 ParityEn 置位，在每个字节后会有一个奇偶校验位。如果 ParityOdd 置位，期望的奇偶校验是一个奇数校验，否则为偶数校验。

16.4.2.3 冲突检测

如果在卡选择阶段内 RF 区域的卡超过一个，它们将会同时响应。MF RC500 支持 ISO14443-A 定义的算法，通过所谓的防冲突处理来解决卡序列号的数据冲突。该方法的基础是检测位冲突的能力。

位冲突检测由旧的位编码模式即 Manchester 编码支持。如果在一个位的第一个和第二个半位检测到的是副载波调制而不是传送的 1 或者 0，就通知出现一个位冲突。为了将一个 1 或 0 位与位冲突区别开，MF RC500 使用 CollLevel 的设定。如果较小振幅半位的振幅大于 CollLevel 定义的值，MF RC500 指示一个位冲突。

如果检测到一个位冲突，错误标志 CollErr 置位。如果在奇偶校验位检测到位冲突，标志 ParityErr 置位以指示一个奇偶校验错误。

接收器独立于检测到的位冲突，继续接收进入的数据流。在位冲突的情况下解码器在冲突的位置发送 1。

注：作为一个例外，如果 ZeroAfterColl 置位，第一个位冲突之后接收的所有位都强制为 0，不管检测到的是位冲突或是明确的状态。该特性使得通过软件实现 ISO14443-A 定义的防冲突处理变得容易。

当一个帧检测到第一个位冲突时，该位的位置保存在 CollPos 寄存器中。

冲突的位置如下表所示：

冲突位置	CollPos 的值
SOF	0
最低字节的最低位	1
…	…
最低字节的最高位	8
第二个字节的最低位	9
…	…
第二个字节的最高位	16
第三个字节的最低位	17
…	…

表 16-2 位冲突位置的返回值

由于奇偶校验位中的位冲突定义，奇偶校验位在 CollPos 中没有计数。如果在 SOF 中检测到一个位冲突，将报告一个帧错误并不再将数据送入 FIFO 缓冲区。这种情况下，接收器继续监控进入的信号并在检测到错误输入数据流的结束时对微处理器产生正确的通知。这有助于微处理器确定什么时候允许对卡进行下一次发送。

16.4.2.4 接收位方式帧

接收器能够处理不完整字节的字节流，这就产生位方式帧。要支持该功能，必须使用下面的值：

- RxAlign 选择一个进入字节的位偏移。例如，如果 RxAlign 设置为 3，接收到的前 5 位送入 FIFO 缓冲区。后面的位打包成字节并输入。在接收之后 RxAlign 自动清零。如果 RxAlign 设置为 0，所有接收到的位都装入一个字节。
- RxLastBits 返回最后接收到的字节的有效位的数目。例如，如 RxLastBits 在接收命令的结束时值为 5，则低 5 位有效。如果最后字节完整，RxlastBits 为 0。

如果没有通过标志 FrameErr 指示的帧错误，RxLastBits 只能有效。如果 RxAlign 设置为一个不为 0 的值并且 ParityEn 有效，不检测第一个奇偶校验位而是将其忽略。

重点：如果 RxAlign 设置为 7，MF RC500 不会将第一个字节（包含从卡接收的单个位）送到微处理器，而是将其删除（见 5.2.2.8）。

注：在位位置 6,14,22,30,38（CollPos）检测到的位冲突不能由 RxAlign 的支持所解决，而是需要由软件完成。

16.4.2.5 通信错误

下表所示导致错误标志置位的事件：

原因	置位的位
接收到的数据没有以 SOF 格式起始	FramingErr
CRC 块的值与期望的不同	CRCErr
接收到的数据比 CRC 块短	CRCErr
奇偶校验位不等于期望的值（例，期望奇偶校验位的位置出现了一个位冲突	ParErr
检测到位冲突	CollErr

表 16-3 通信错误表

16.4.5　与卡通信的状态图

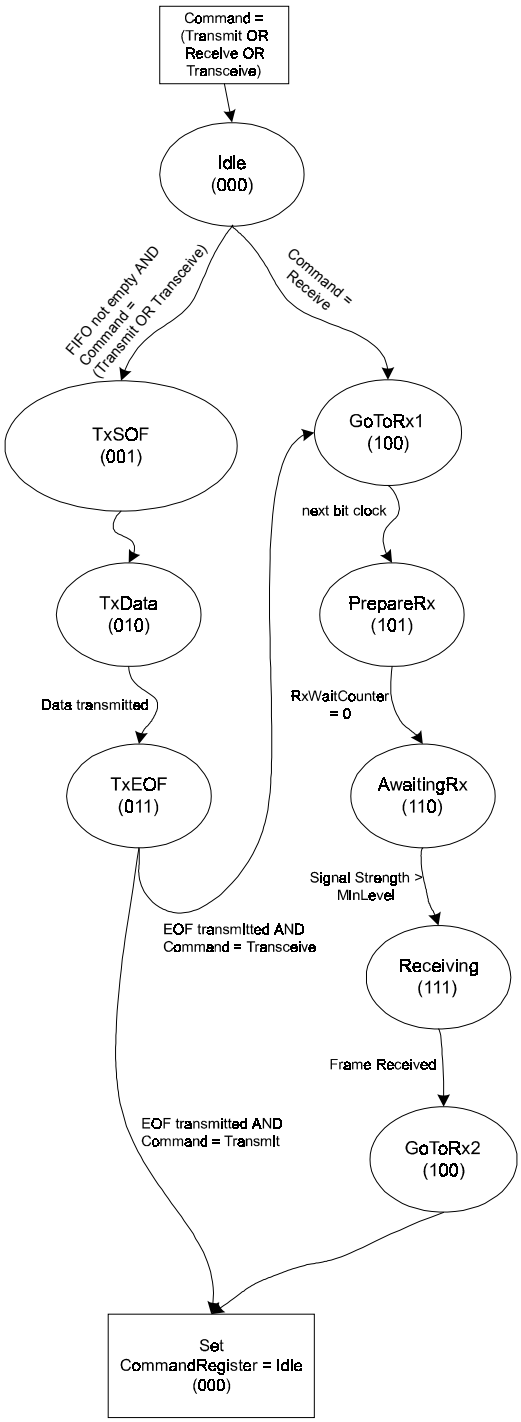


图 16-4　状态图：卡通信

16.5　访问 E²PROM 命令

16.5.1　WriteE2 命令 01H

16.5.1.1　概述

命令	代码	动作	变量和数据	返回的数据
WriteE2	01	从 FIFO 缓冲区获得数据并写入 E²PROM	起始地址低字节 起始地址高字节 数据字节流	-

WriteE2 命令将 FIFO 缓冲区的头两个字节解释为 E²PROM 的起始字节地址.之后的字节解释为数据字节并从给定的起始字节地址将其写入 E²PROM。该命令不返回任何数据。

WriteE2 命令只能由微处理器启动。它不会自动停止，必须由微处理器执行 Idle 命令将其终止。

16.5.1.2 编程的处理

在一个编程周期内可将 1 到 16 个字节写入 E²PROM。需要的时间约为 5.8ms。状态机将 FIFO 缓冲区内所有准备好的数据复制到 E²PROM 输入缓冲区。内部 E²PROM 输入缓冲区的长度为 16 字节，等于 E²PROM 块的规格。编程周期从写入 E²PROM 输入缓冲区的最后一个位置或者从读出 FIFO 缓冲区的最后一个字节开始。

只要在 FIFO 缓冲区内有未经处理的字节或者 E²PROM 编程周期仍在进行当中，标志 E2Ready 为 0。如果从 FIFO 缓冲区读出的所有数据都已写入 E²PROM，标志 E2Ready 设置为 1。在 E2Ready 的上升沿时 TxIRq 指示为 1。这用于产生一个中断指示所有数据的编程都已完成。

在 E2Ready 设置为 1 后，WriteE2 命令可通过微处理器写入 Idle 命令停止。

重点：在 E2Ready 为 0 时，不能通过启动其它命令来停止 WriteE2 命令。否则当前处理的 E²PROM 块的内容不会被定义，或者更糟糕的情况：MF RC500 的功能被不可逆地降低。

16.5.1.3 时序图

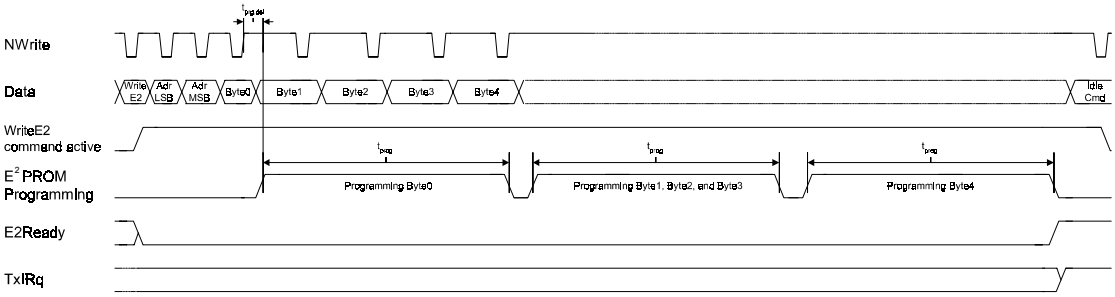


图 16-5 E²PROM 编程时序图

说明：假设 MF RC500 在微处理器可以写字节 1（tprog,del=300ns）之前找到并读出字节 0。这使得 MF RC500 开始编程周期，大概需要 2.9ms。同时微处理器将字节 1 到字节 4 保存到 FIFO 缓冲区。假设 E²PROM 起始字节地址为 16CH，字节 0 就保存在此处。MF RC500 将后续的数据字节复制到 E²PROM 输入缓冲区。复制字节 3 检测到该字节必须编程到 E²PROM 字节地址 16FH。由于这是存储器块的结束，MF RC500 自动启动一个编程周期。字节 4 将会编程到 E²PROM 字节地址 170H。由于这是最后一个数据字节，指示 E²PROM 编程结束的标志（E2Ready 和 TxIRq）置位。

尽管已将所有的数据写入 E²PROM，MF RC500 仍停在 WriteE2 命令。将更多的数据写入 FIFO 缓冲区将会继续从地址 171H 开始对 E²PROM 编程。使用 Idle 命令停止该命令。

16.5.1.4 WriteE2 命令的错误标志

对 E²PROM 块 0（E²PROM 字节地址 00H~0FH）的编程是被禁止的。对该地址的编程将置位标志 AcessErr。不会启动编程周期。超过 1FFH 的地址按 200H 取模（E²PROM 存储器的结构见第 6 章）

16.5.2 ReadE2 命令 03H

16.5.2.1 概述

命令	代码	动作	变量和数据	返回的数据
ReadE2	03	从 E²PROM 读出数据并将其放入 FIFO 缓冲区	起始地址低字节 起始地址高字节 数据字节流	数据字节

ReadE2 命令将 FIFO 缓冲区内的头两个字节解释为 E²PROM 的起始字节地址。下一个字节指定返回的字节数。当所有 3 个变量字节都存在于 FIFO 缓冲区时，就从给定的 E²PROM 起始字节地址开始将指定数目的数据字节从 E²PROM 复制到 FIFO 缓冲区。

ReadE2 命令只能由微处理器触发。当所有数据传输完毕，该命令自动停止。

16.5.2.2 ReadE2 命令的错误标志

对 E^2PROM 块 8 到 1FH 的读操作是被禁止的。对这些地址的读操作将会置位标志 AccessErr。超过 1FFH 的地址按 200H 取模（E^2PROM 存储器的结构见第 6 章）。

16.6 其它命令

16.6.1 LoadConfig 命令 07H

16.6.1.1 概述

命令	代码	动作	变量和数据	返回的数据
LoadConfig	07	从 E^2PROM 读出数据并初始化寄存器	起始地址低字节 起始地址高字节	-

LoadConfig 命令将 FIFO 缓冲区内的头两个字节解释为 E^2PROM 的起始字节地址。当两个变量字节存在于 FIFO 缓冲区时，就从给定的 E^2PROM 起始字节地址开始将 32 个字节从 E^2PROM 复制到 MF RC500 的控制和配置寄存器中。LoadConfig 命令只能由微处理器触发。当所有相关寄存器的复制完成时，该命令自动停止。

16.6.1.2 寄存器分配

E^2PROM 起始字节地址开始的 32 个字节内容写入 MF RC500 寄存器 10H 到 2FH（E^2PROM 存储器的结构见第 6 章）。

注：寄存器分配的过程与启动初始化是一样的（见 11.3）。不同的是，启动初始化时的 E^2PROM 起始地址固定为 10H（块 1，字节 0），而 LoadConfig 命令可以选择。

16.6.1.3 LoadConfig 命令相关的错误标志

有效的 E^2PROM 起始地址为 10H 到 60H。对块 8H 到 1FH 的复制是被禁止的。对这些地址的读操作将会置位标志 AccessErr。超过 1FFH 的地址按 200H 取模（E^2PROM 存储器的结构见第 6 章）。

16.6.2 CalcCRC 命令 12H

命令	代码	动作	变量和数据	返回的数据
CalcCRC	12	启动 CRC 协处理器	数据字节流	-

CalcCRC 命令将 FIFO 缓冲区内的所有数据取出作为 CRC 协处理器的输入字节。所有保存在 FIFO 缓冲区内的数据在命令启动之前都将进行处理。该命令不通过 FIFO 缓冲区返回任何数据，但 CRC 寄存器的内容可通过 CRCResultLSB 寄存器和 CRCResultMSB 寄存器读出。CalcCRC 命令只能由微处理器启动。它不会自动停止，必须由微处理器执行 Idle 命令将其终止。。如果 FIFO 缓冲区为空，CalcCRC 命令等待 FIFO 缓冲区更多的输入。

16.6.2.2 CRC 协处理器设定

用于 CRC 协处理器的参数可进行如下配置：

参数	值	位	寄存器
CRC 寄存器长度	8 位或 16 位 CRC	CRC8	ChannelRedundancy
CRC 算法	根据 ISO14443-A 或 ISO/IEC3309 的算法	CRC3309	ChannelRedundancy
位处理方向	将 MSBit 或 LSBit 先移入 CRC 寄存器	CRCMSBfist	ChannelRedundancy
CRC 预置值	任意	CRCPresetLSB CRCPresetMSB	CRCPresetLSB CRCPresetMSB

表 16-5 CRC 协处理器参数

8 位 CRC 的 CRC 多项式固定为：$x^8+x^4+x^3+x^2+1$

16 位 CRC 的 CRC 多项式固定为：$x^{16}+x^{12}+x^5+1$

16.6.2.3 CRC 协处理器的状态标志

CRCReady 的状态标志指示 CRC 协处理器已经完成对 FIFO 缓冲区中所有数据字节的处理。CRCReady 标志置位后，TxIRq 置位产生中断请求，这支持了使用 CRC 协处理器驱动中断。

当CRCReady和TxIRq分别设置为1时，CRCResultLSB和CRCResultMSB寄存器的内容和标志CRCErr有效。

CRCResultLSB 和 CRCResultMSB 寄存器保存 CRC 寄存器的内容，CRCErr 标志指示所处理的数据的 CRC 有效性。

16.7 命令执行时的错误处理

如果在命令执行时检测到任何错误，通过置位 PrimaryStatus 寄存器中的状态标志 Err 来指示。微处理器可以通过 ErrorFlag 寄存器中的状态标志得到错误原因的信息。

ErrorFlag 寄存器的错误标志	相关命令
KeyErr	LoadKeyE2,LoadKey
AccessError	WriteE2,ReadE2,LoadConfig
FIFOOvl	无特定的命令
CRCErr	Receive,Tranceive,CalcCRC
FramingErr	Receive,Tranceive
ParityErr	Receive,Tranceive
CollErr	Receive,Tranceive

表 16-6 错误标志汇总

16.8 MIFARE 典型的安全性命令

16.8.1 LoadKeyE2 命令 0BH

16.8.1.1 概述

命令	代码	动作	变量和数据	返回的数据
LoadKeyE2	0B	从 E^2PROM 读出一个密匙并将其放入内部密匙缓冲区	起始地址低字节 起始地址高字节	-

LoadKeyE2 命令将 FIFO 缓冲区内的头两个字节解释为 E^2PROM 的起始字节地址。从给定起始字节地址开始的 E^2PROM 字节被解释为密匙，并以正确的密匙格式保存（见 6.4.1）。当两个变量字节存在于 FIFO 缓冲区时，命令开始执行。LoadKeyE2 命令只能由微处理器启动。将 E^2PROM 的密匙复制到密匙缓冲区后，它自动停止。

16.8.1.2 与 LoadKeyE2 命令相关的错误标志

如果密匙格式不正确（见 6.4.1），会将一个未定义的值复制到密匙缓冲区同时标志 KeyErr 置位。

16.8.2 LoadKey 命令 19H

16.8.2.1 概述

命令	代码	动作	变量和数据	返回的数据
LoadKey	19	从 FIFO 缓冲区出一个密匙并将其放入密匙缓冲区	字节 0（LSB） 字节 1 … 字节 10 字节 11（MSB）	-

LoadKey 命令将 FIFO 缓冲区内的前 12 个字节解释为密匙，以正确的格式保存（见 6.4.1）。当 FIFO 缓冲区可提供 12 个变量字节时，对其检测如果有效则复制到密匙缓冲区（见 17.2）。

LoadKey 命令只能由微处理器启动。将密匙从 FIFO 缓冲区复制到密匙缓冲区后，它自动停止。

16.8.2.2 与 LoadKey 命令相关的错误标志

将所有需要的字节从 FIFO 缓冲区复制到密匙缓冲区。如果密匙格式不正确（见 6.4.1），会将一个未定义的值复制到密匙缓冲区同时标志 KeyErr 置位。

16.8.3 Authent1 命令 0CH

16.8.3.1 概述

命令	代码	动作	变量和数据	返回的数据
Authent1	0C	从 FIFO 缓冲区出一个密匙并将其放入密匙缓冲区	卡 Auth 命令 卡的块地址 卡的序列号低字节 卡的序列号字节 1 卡的序列号字节 2 卡的序列号高字节	-

Authent1 命令是一个特殊的 Transeive 命令：它占用 6 个字节用于发送到卡。卡的响应不送到微处理器，但用于检测卡的可靠性并向卡证实 MF RC500 的可靠性。

Authent1 命令只能由微处理器启动。该命令的状态时序与 Transceive 命令相同（见 16.4.3）。

16.8.4 Authent2 命令 14H

16.8.4.1 概述

命令	代码	动作	变量和数据	返回的数据
Authent2	14	使用 Crypto1 算法执行卡可靠性的第二部分	-	-

Authent2 命令是一个特殊的 Transeive 命令：它不需要任何变量字节，但所有要发送到卡的数据由 MF RC500 自身进行分配。卡的响应不送到微处理器，但用于检测卡的可靠性并向卡证实 MF RC500 的可靠性。

Authent2 命令只能由微处理器启动。该命令的状态时序与 Transceive 命令相同（见 16.4.3）。

16.8.4.2 Authent2 命令的作用

如果 Authent2 命令成功验证了卡和 MF RC500 的可靠性，控制位 Crypto1On 自动置位。当 Crypto1 置位后，所有进一步与卡的通信都使用 Crypto1 保密算法加密。如果 Authent2 命令失败，位 Crypto1On 清零。

注：标志 Crypto1On 不能由微处理器置位，只能通过成功执行 Authent2 命令置位。微处理器可以清零该位与卡继续进行明码通信。

注：Authent2 命令必须在 Authent1 命令成功后马上执行（见 16.8.3）。此外，保存在密匙缓冲区内的密匙必须与卡上的一致。

17 MIFARE 标准的验证和加密

17.1 概述

MIFIRE 标准产品应用的加密算法叫做 Crypto1。它基于私有的流密码，密匙长度为 48 位。要访问 MIFARE 标准卡的数据，必须了解相应的密匙。要成功进行卡的验证和随后对卡内 EEPROM 数据进行访问，MF RC500 必须具有正确的密匙。在选择以 ISO14443A 定义的卡后，用户可以继续使用 MIFARE 标准协议。这种情况下，强制执行卡的验证。Crypto1 验证是一个 3-pass 验证。该过程通过执行 Authent1 和 Authent2 命令自动完成。在卡的验证过程中，加密算法初始化。在成功验证之后，与 MIFARE 标准卡的通信被加密。

17.2 Crypto1 密匙处理

在验证命令中 MF RC500 从内部密匙缓冲区读出密匙。密匙总是取自密匙缓冲区。因此，Crypto1 验证的命令不需要对密匙进行寻址。用户必须确保在卡验证启动之前密匙缓冲区内准备的密匙是正确的。

密匙缓冲区可以下面的方式装载：

- 用 LoadKeyE2 命令从 E2PROM 中装载（见 16.8.1）
- 使用 LoadKey 命令通过 FIFO 缓冲区从微处理器直接装载

如下图所示：

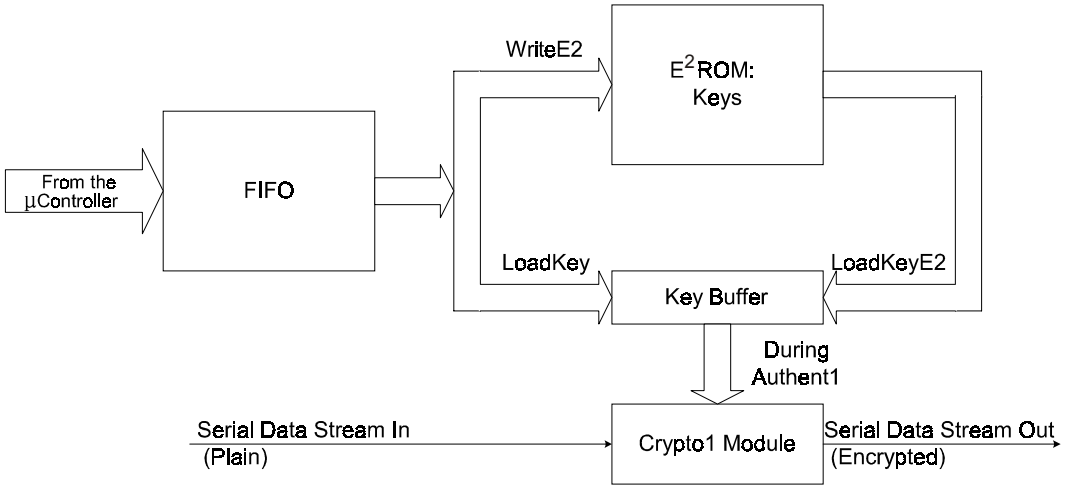


图 17-1　密匙处理：方框图

17.3　执行 MIFARE 标准验证

执行 Crypto1 加密算法来确保对 MIFARE 标准卡的验证。为了获得有效的验证，MF RC500 密匙缓冲区内的密匙必须正确。

步骤 1：通过 LoadKeyE2 或 LoadKey 命令将密匙装入密匙缓冲区

步骤 2：启动 Authent1 命令，结束时检测错误标志以获得该命令执行的状态。

步骤 3：启动 Authent2 命令，结束时检测错误标志和 Crypto1On 以获得命令执行的状态。

18　典型应用

18.1　电路图

下图所示为一个典型的应用，天线直接连到 MF RC500。

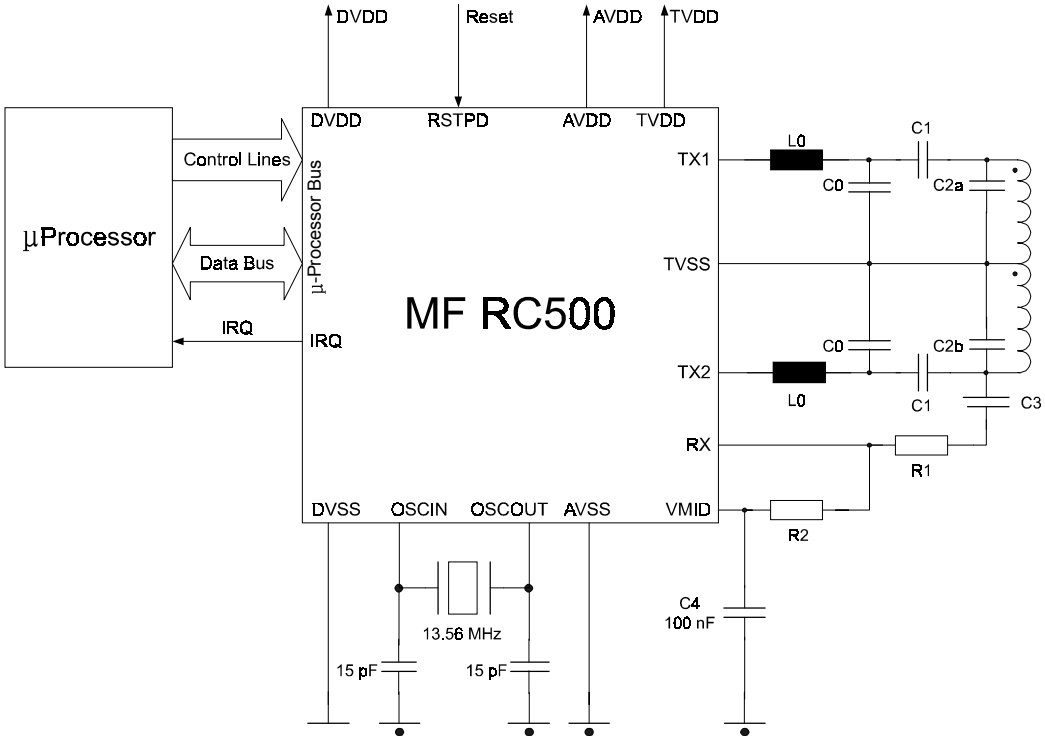


图 18-1　应用示例的电路图：直接匹配天线

匹配电路包括一个 EMC 低通滤波器、接收电路、天线匹配电路和天线。更多关于天线设计和调谐的细节请参阅应用手册“MF RC500 匹配电路和天线的设计”。

18.2 电路描述

18.2.1 EMC 低通滤波器

MIFARE 系统在 13.56MHz 频率下操作。该频率由一个石英晶振产生用于驱动 MF RC500 以及作为驱动天线的 13.56MHz 能量载波的基频。这样不仅会产生 13.56Mhz 的发射功率而且会发射更高的谐波。国际 EMC 条例定义了在广播频段中发射功率的幅值。因此，要符合这一规范，必须对输出信号进行适当的滤波。

对于多层电路板强烈建议使用上图所示的低通滤波器。低通滤波器包括元件 L0 和 C0。下表给出了它们的值。

注：为了实现最佳的性能，所有元件都至少应当达到所推荐的元件的质量。

18.2.2 接收电路

MF RC500 的内部接收部分使用一个受益于副载波双边带的概念装入卡响应的调整。推荐使用内部产生的 VMID 电势作为 RX 脚的输入电势。为了提供一个稳定的参考电压，必须在 VMID 脚接一个对地的电容 C4。读卡器的接收部分需要在 RX 和 VMID 脚之间连接一个分压器。此外，在天线线圈和分压器之间推荐使用一个串接的电容。上面所示的电路图包括了推荐的接收电路。接收电路包括的元件有 R1，R2，C3 和 C4。它们的值见下表：

元件	值	备注
L0	2.2μH±10%	磁屏蔽 例如 TDK ACL3225S-T
C0	47pF±2%	NP0 材料，根据天线的电感量取值
R1	10kΩ±5%	
R2	820Ω±5%	
C3	15pF±2%	NP0 材料

表 18-1 EMC 滤波器和接收电路的取值

注：电容不推荐使用 X7R 材料。

18.3 天线线圈电感量的计算

对天线线圈电感量的精确计算是不可行的。但电感量可以通过下面的公式估算出来。我们推荐将天线设计成环形或者矩形。

$$L_1[\mathrm{nH}] = 2 \cdot I_1[\mathrm{cm}] \cdot \left\{ \ln\left|\frac{I_1}{D_1}\right| - K \right\} N_1^{1,8}$$

I_1……...导体环一圈的长度

D_1…….导线的直径或者 PCB 导体的宽度

K……天线形状因素（环形天线 K=1.07，矩形天线 K=1.47）

N_1……..圈数

ln……自然对数函数

18.3.1 直接匹配天线的阻抗匹配

要设计一个直接匹配天线的匹配电路，建议使用 18.1 中的电路。电容 C1，C2a 和 C2b 的值取决于天线的电气特性和环境的影响。

下表所示的电容值为指导性的值。它们用于调谐过程的启动值。

天线线圈电感量[μH]	C1[pH]	C2a[pF]	C2b[pF]
0.8	27	270	330
0.9	27	270	270
1.0	27	220	270
1.1	27	180‖22	220
1.2	27	180	180‖22
1.3	27	180	180
1.4	27	150	180
1.5	27	150	150
1.6	27	120‖10	150
1.7	27	120	150
1.8	27	120	120

表 18-2　匹配电路的电容值

然而，要得到最佳的性能，必须通过调谐过程找到精确的值。上表假设天线线圈的杂散电容值为 15pF。电容 C1 和 C2s 应当采用允差为±2%的 NP0 电介质。

天线的电感和电容的实际值取决于不同的参数。例如：

- 天线的结构（PCB 的类型）
- 导体的厚度
- 绕组之间的距离
- 屏蔽层
- 附近环境中的金属或铁氧体

19　测试信号

19.1　概述

MF RC500 允许不同信号的测量。使用串行信号转换的可能性，可将测量用于检测内部产生和接收到的信号（见 15）。另外，用户可通过寄存器选项选择内部模拟信号在脚 AUX 处对它们进行测量，以及选择内部数字信号在脚 MFOUT 对它们进行观察。在设计阶段用于优化接收器的特性或者作为测试用途，这些测量都是很有用的。

19.2　使用串行信号转换进行测量

通过脚 MFOUT 处的串行信号转换，用户可观察到发送到卡或者接收自卡的数据。下表为不同可能的信号的汇总。

SignalToMFOUT	MFOUTSelect	输出到 MFOUT 脚的信号
0	0	低
0	1	高
0	2	包络
0	3	发送 NRZ
0	4	带副载波的 Manchester
0	5	Manchester
0	6	RFU
0	7	RFU
1	X	数字测试信号

表 19-1　输出到 MFOUT 脚的信号

19.2.1 TX 控制

下图所示为使用串行信号转换控制发送到卡的数据，在 MFOUT 测得的信号。将标志 MFOUTSelect 设置为 3，发送到卡的数据以 NRZ 编码显示。将标志 MFOUTSelect 设置为 2，显示 Miller 编码信号。

直接在天下测量的 RFOut 信号显示了 RF 信号的脉冲形状。

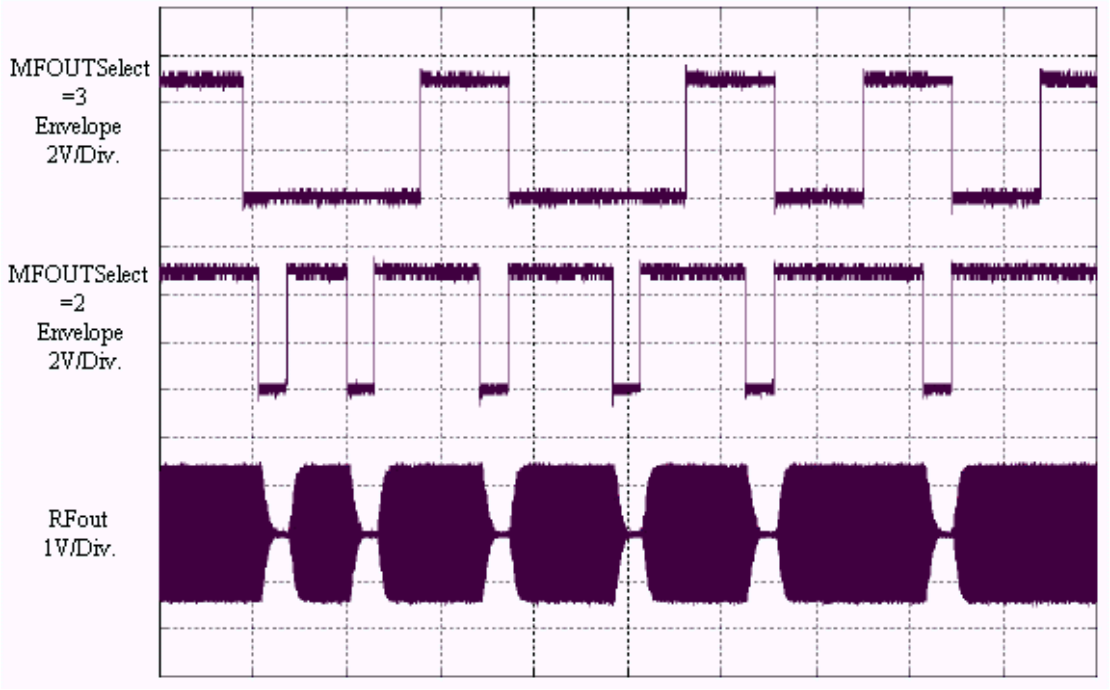


图 20 TX 控制信号

19.2.2 RX 控制

下图所示为卡对一个请求信号的应答的开始。直接在天线上测量的 RF 信号显示了 RF 的电压，这使得卡的装载调制可见。MFOUTSelect 设置为 4 显示带副载波的 Manchester 解码信号。MFOUTSelect 设置为 5 显示 Manchester 解码信号。

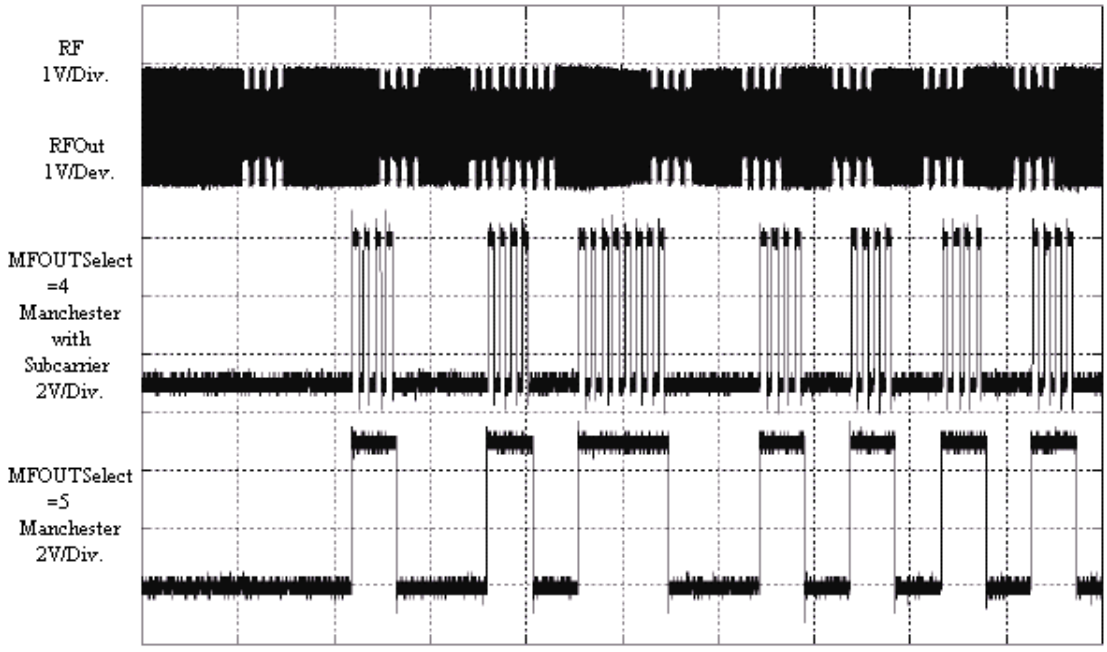


图 21 RX 控制信号

19.3 模拟测试信号

模拟测试信号可通过选择寄存器位 TestAnaOutSel 输出到脚 AUX。

值	信号名称	描述
0	V_{mid}	内部节点 Vmid 的电压
1	$V_{bandgap}$	由带隙产生的内部参考电压
2	$V_{RxFollI}$	使用 I-时钟从解调器输出的信号
3	$V_{RxFollQ}$	使用 Q-时钟从解调器输出的信号
4	V_{RxAmpI}	经过放大和滤波的 I 信道副载波信号
5	V_{RxAmpQ}	经过放大和滤波的 Q 信道副载波信号
6	V_{CorrNI}	由 I 信道副载波信号馈送的 N 信道相关器输出信号
7	V_{CorrNQ}	由 Q 信道副载波信号馈送的 N 信道相关器输出信号
8	V_{CorrDI}	由 I 信道副载波信号馈送的 D 信道相关器输出信号
9	V_{CorrDQ}	由 Q 信道副载波信号馈送的 D 信道相关器输出信号
A	V_{EvalL}	从左半位测出的信号
B	V_{EvalR}	从右半位测出的信号
C	V_{Temp}	源于带隙的温度电压
D	rfu	保留将来之用
E	rfu	保留将来之用
F	rfu	保留将来之用

表 19-2 模拟测试信号选项

19-4 数字测试信号

数字测试信号可通过将 SignalToMFOUT 位置 1 输出到脚 MFOUT。数字测试信号可以通过寄存器 TestDigiSelect 中的 TestDigiSignalSel 选择。如下表所示：

TestDigiSignalSel	信号名称	描述
F4H	s_data	接收自卡的数据
E4H	s_valid	显示为 1，信号 s_data 和 s_coll 有效
D4H	s_coll	显示为 1，当前位检测到冲突
C4H	s_clock	内部串行时钟：发送时作为编码器时钟，接收时作为接收器时钟。
B5H	rd_sync	内部同步读信号（来自微处理器并行接口）
A5H	wr_sync	内部同步写信号（来自微处理器并行接口）
96H	int_clock	内部 13.56MHz 时钟
00H	无测试信号	由 MFOUTSelect 定义的信号输出到脚 MFOUT

表 19-3 数字测试信号选项

如果不使用测试信号，TestDigiSelect 寄存器的值应当为 00H。

注：TestDigiSignalSel 所有其它的值仅用于产品测试。

19.5 模拟和数字测试信号举例

图 22 显示了一个 MIFARE 标准卡通过 Q 时钟接收通道对一个请求命令的应答。RX 参考端用来显示 RX 脚的 Manchester 调制信号。该信号在接收器电路中被解调并放大。VRXAmpQ 显示了使用 Q 时钟解调的放大的边带信号。相关性电路中产生的信号 VCorrDQ 和 VCorrNQ 在计算和数字化电路中被计算和数字化。VEvalR 和 VEvalL 显示了对左右半位的计算信号。最后，数字测试信号 S_data 显示发送到内部数字电

路的接收数据，S_valid 指示接收的数据流有效。

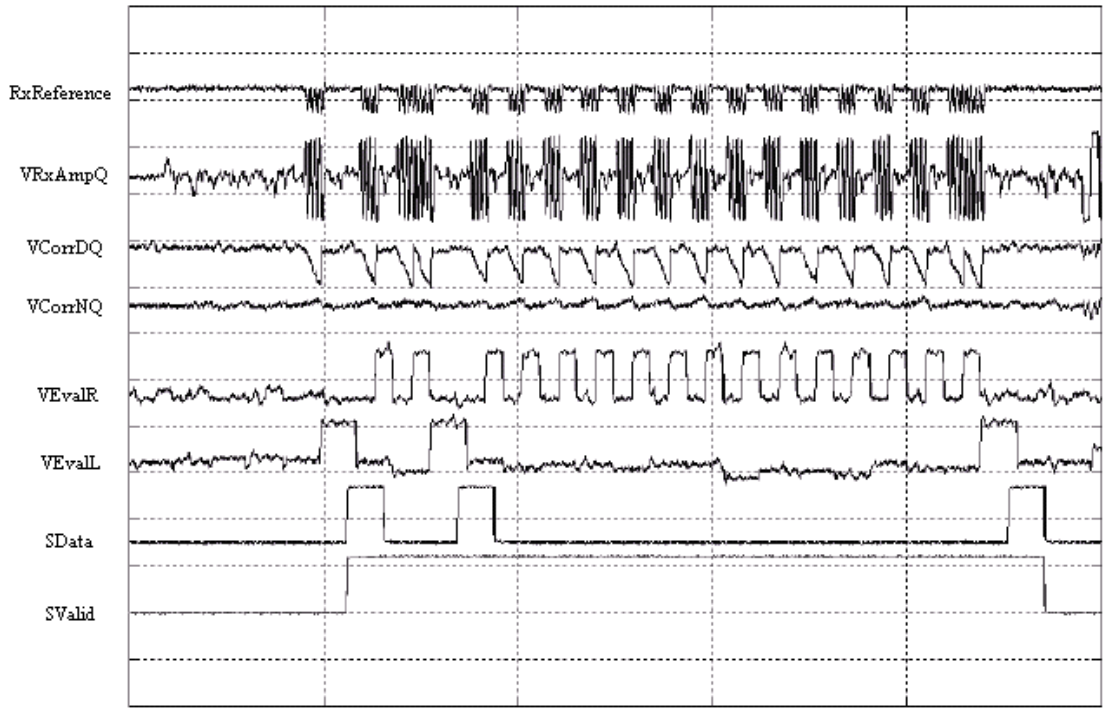


图 22　接收通道 Q 时钟

20　电气特性

20.1　极限参数

符号	参数	最小	最大	单位
Tamb,abs	环境或储存温度范围	-40	+150	℃
DVDD AVDD TVDD	DC 电源电压	-0.5	6	V
Vin,abs	任意数字脚对 DVSS 的电压	-0.5	DVDD+0.5	V
Vrx,abs	RX 脚对 AVSS 的电压	-0.5	AVDD+0.5	V

20.2　操作条件范围

符号	参数	条件	最小	典型	最大	单位
Tamb	环境温度	-	-25	+25	+85	℃
DVDD	数字电源电压	DVSS=AVSS=TVSS=0V	4.5	5.0	5.5	V
AVDD	模拟电源电压		4.5	5.0	5.5	V
TVDD	发送器电源电压		3.0	5.0	5.5	V

20.3 电流消耗

符号	参数	条件	最小	典型	最大	单位
I_{DVDD}	数字电源电压	Idle 命令		6		mA
		待机模式		3		mA
		软件掉电模式		800		μA
		硬件掉电模式		1		μA
I_{AVDD}	模拟电源电压	Idle 命令,接收器打开		29		mA
		Idle 命令,接收器关闭		10		mA
		待机模式		8		mA
		软件掉电模式		1		μA
		硬件掉电模式		1		μA
I_{TVDD}	发送器电源电压	连续波			150	mA
		TX1 和 TX2 未连接 TX1RFEn,TX2RFEn=1		4.5	9	mA
		TX1 和 TX2 未连接 TX1RFEn,TX2RFEn=0		65	130	μA

20.4 管脚特性

20.4.1 输入管脚特性

管脚 D0 到 D7 和 A1 具有 TTL 输入特性，见下表的定义

符号	参数	条件	最小	最大	单位
I_{Leak}	输入漏电流		-1.0	+1.0	μA
V_T	阀值		0.8	2.0	V

数字输入脚 NCS，NWR，NRD，ALE，A2 和 MFIN 具有施密特触发器特性，见下表的定义

符号	参数	条件	最小	最大	单位
I_{Leak}	输入漏电流		-1.0	+1.0	μA
V_{T+}	上升沿阀值		1.4	2.0	V
V_{T-}	下降沿阀值		0.8	1.3	V

脚 RSTPD 具有施密特触发器 CMOS 特性。此外，它通过一个 RC 低通滤波器进行内部滤波，这使得复位信号有一个相应的传递延迟。

符号	参数	条件	最小	最大	单位
I_{Leak}	输入漏电流		-1.0	+1.0	μA
V_{T+}	上升沿阀值		0.65DVDD	0.75DVDD	V
V_{T-}	下降沿阀值		0.25DVDD	0.4DVDD	V
$t_{RSTPD,p}$	传递延迟			20	μs

模拟输入脚 RX 输入电容如下

符号	参数	条件	最小	最大	单位
C_{RX}	输入电容			15	pF

20.4.2 数字输入管脚特性

管脚 D0 到 D7，MFOUT 和 IRQ 具有 TTL 输出特性，见下表定义

符号	参数	条件	最小	典型	最大	单位
V_{OH}	输出电压高	DVDD=5V,I_{OH}=-1mA	2.4	4.9		V
		DVDD=5V,I_{OH}=-10mA	2.4	4.2		V
V_{OL}	输出电压低	DVDD=5V,I_{OL}=1mA		25	400	mV
		DVDD=5V,I_{OL}=10mA		250	400	mV
I_O	输出电流	DVDD=5V			10	mA

注：IRQ 脚也可配置为集电极开路。那样 V_{OH} 的值不再适用。

20.4.3 天线驱动器输出管脚特性

天线驱动器脚 TX1 和 TX2 用于驱动高电平的源电导可通过 CwConductance 寄存器中的 GsCfgCW 进行配置。而它们用于驱动低电平的源电导为常量。

作为默认的配置，输出特性定义如下：

符号	参数	条件	最小	典型	最大	单位
V_{OH}	输出电压高	DVDD=5V,I_{OL}=20mA		4.97		V
		DVDD=5V,I_{OL}=100mA		4.85		V
V_{OL}	输出电压低	DVDD=5V,I_{OL}=20mA		30		mV
		DVDD=5V,I_{OL}=100mA		150		mV
I_{TX}	发送器输出电流	连续波			200	mA

20.5 AC 电气特性

20.5.1 AC 符号

每个时序的符号都有 5 个字符。第一个字符总是为“t”表示时间。其它字符用来表示信号名或信号的逻辑状态（取决于它们的位置），说明如下：

名称	信号	名称	逻辑电平
A	地址	H	高
D	数据	L	低
W	NWR 或 nWait	Z	高阻
R	NRD 或 R/NW 或 nWrite	X	任意电平或数据
L	ALE 或 AS	V	任意有效信号或数据
C	NCS		
S	NDS 或 nDStrb 和 nAStrb		

例如：t_{AVLL}=从地址有效到 ALE 为低的时间。

20.5.2.1 Read/Write 独立选通的总线时序

符号	参数	最小	最大	单位
t_{LHLL}	ALE 脉宽	20		ns
t_{AVLL}	复用地址总线有效到 ALE 低(地址建立时间)	15		ns
t_{LLAX}	ALE 低之后的复用地址总线有效 (地址保持时间)	8		ns
t_{LLWL}	ALE 低到 NWR,NRD 低	15		ns
t_{CLWL}	NCS 低到 NRD,NWR 低	0		ns
t_{WHCH}	NRD,NWR 高到 NCS 高	0		ns
t_{RLDV}	NRD 低到 DATA 有效		65	ns
t_{RHDZ}	NRD 高到 DATA 高阻		20	ns
t_{WLDV}	NWR 低到 DATA 有效		35	ns
t_{WHDX}	NWR 高之后的 DATA 保持(数据保持时间)	8		ns
t_{WLWH}	NRD,NWR 脉宽	65		
t_{AVWL}	独立地址总线有效到 NRD,NWR 低(建立时间)	30		ns
t_{WHAX}	ALE 低之后的独立地址总线有效 (地址保持时间)	8		ns
t_{WHWL}	连续读/写访问的间隔	150		ns

表 20-10 独立读/写选通的时序定义

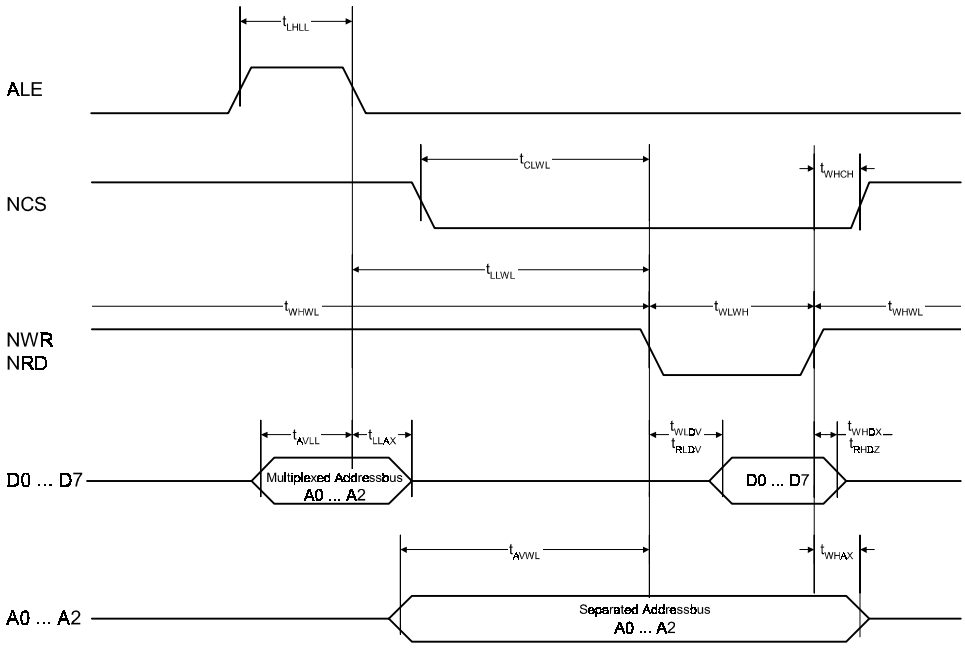


图 20-1 独立读/写选通的时序图

注：用于独立的地址和数据总线时，ALE 信号无关联且数据总线上的复用地址不用考虑。用作复用的地址和数据总线时，地址线 A0 到 A2 必须按照 4.3 所述连接。

20.5.2.2 共用读/写/选通的总线时序

符号	参数	最小	最大	单位
t_{LHLL}	AS 脉宽	20		ns
t_{AVLL}	复用地址总线有效到 AS 低(地址建立时间)	15		ns
t_{LLAX}	AS 低之后的复用地址总线有效 (地址保持时间)	8		ns
t_{LLSL}	AS 低到 NDS 低	15		ns
t_{CLSL}	NCS 低到 NDS 低	0		ns
t_{SHCH}	NDS 高到 NCS 高	0		ns
$t_{SLDV,R}$	NDS 低到 DATA 有效(读周期)		65	ns
t_{SHDZ}	NDS 低到 DATA 高阻(读周期)		20	ns
$t_{SLDV,W}$	NDS 低到 DATA 有效(写周期)		35	ns
t_{SHDX}	NWR 高之后的 DATA 保持(写周期,保持时间)	8		ns
t_{SHRX}	NDS 高之后的 R/NW 保持	8		ns
t_{SLSH}	NDS 脉宽	65		ns
t_{AVSL}	独立地址总线有效到 NDS 低(保持时间)	30		ns
t_{SHAX}	NDS 低之后的独立地址总线有效 (建立时间)	8		ns
t_{SHSL}	连续读/写访问的间隔	150		ns
t_{RVSL}	R/NW 有效到 NDS 低	8		ns

表 20-11 共用读/写选通的时序定义

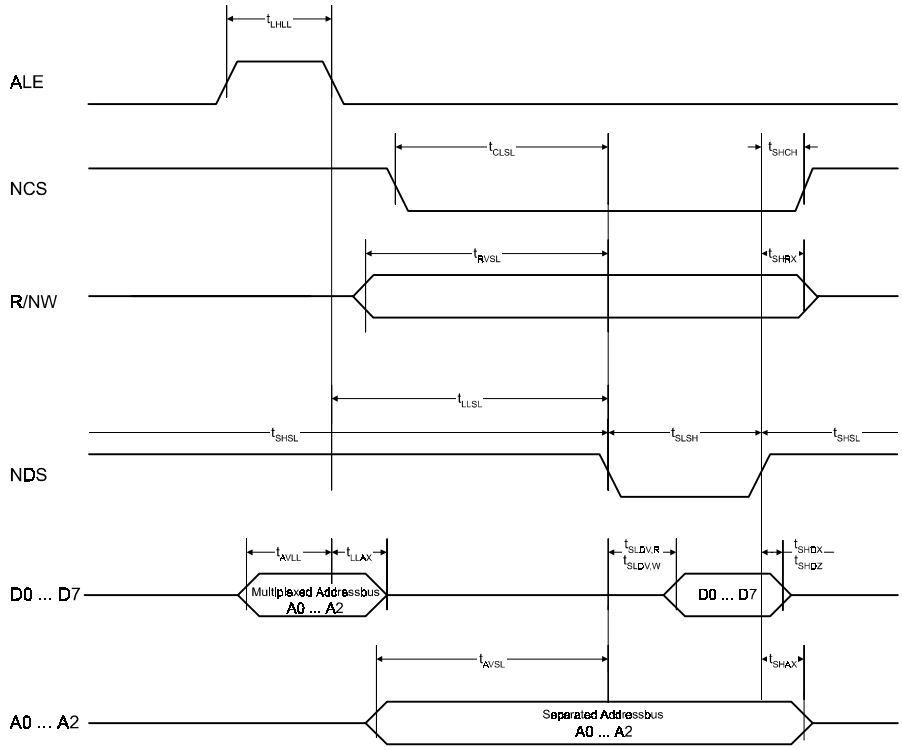


图 20-2 共用读/写选通的时序图

注：用于独立的地址和数据总线时，ALE 信号无关联且数据总线上的复用地址不用考虑。用作复用的地址和数据总线时，地址线 A0 到 A2 必须按照 4.3 所述连接。

20.5.2.3 EPP 的总线时序

符号	参数	最小	最大	单位
t_{LLLH}	nAStrb 脉宽	20		ns
t_{AVLH}	复用地址总线有效到 nAStrb 高(建立时间)	15		ns
t_{LHAX}	nAStrb 高之后的复用地址总线有效 (保持时间)	8		ns
t_{CLSL}	NCS 低到 nDStrb 低	0		ns
t_{SHCH}	nDStrb 高到 NCS 高	0		ns
$t_{SLDV,R}$	nDStrb 低到 DATA 有效(读周期)		65	ns
t_{SHDZ}	nDStrb 低到 DATA 高阻(读周期)		20	ns
$t_{SLDV,W}$	nDStrb 低到 DATA 有效(写周期,建立时间)		35	ns
t_{SHDX}	nDStrb 低之后的 DATA 保持(写周期,保持时间)	8		ns
t_{SHRX}	nDStrb 高之后的 nWrite 保持	8		ns
t_{SLSH}	nDStrb 脉宽	65		ns
t_{RVSL}	nDStrb 低之后的 nWrite 有效	8		ns
t_{SLWH}	nDStrb 低到 nWait 高		75	ns
t_{SHWL}	nDStrb 高到 nWait 低		75	ns

表 20-12 共用读/写选通的时序定义

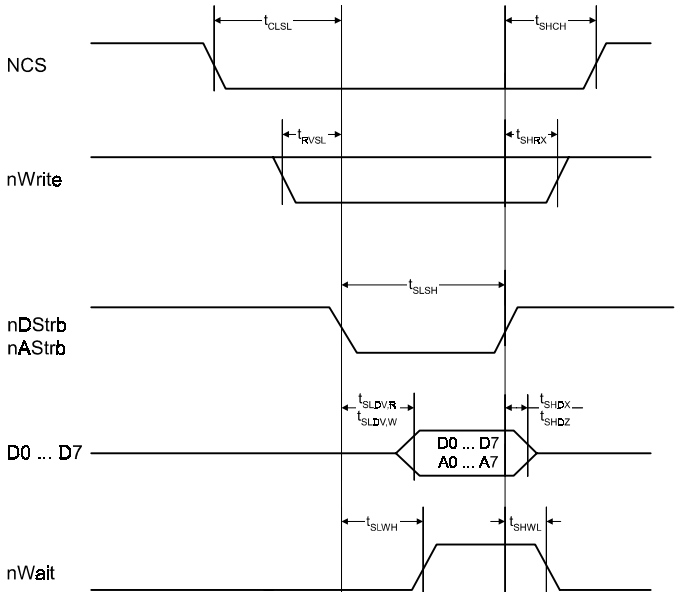


图 20-3 共用读/写选通的时序图

20.5.3 时钟频率

时钟从管脚 1,OSCIN 输入。

参数	符号	最小	典型	最大	单位
时钟频率（由时钟滤波器检测）	f_{OSCIN}		13.56		MHz
时钟频率占空比	d_{FEC}	40	50	60	%
时钟边沿抖动	t_{jitter}			10	ps

提供给 MF RC500 的时钟作为同步系统的编码器和解码器的时基，因此时钟频率的稳定性是正确执行的一个重要因素。为了获得最佳的性能，时钟的抖动应当尽可能小。最好使用带推荐电路的内部振荡器缓冲区（见 12）。

21 E^2PROM 特性

E^2PROM 的规格为 32×16×8＝4,096 位

符号	参数	条件	最小	最大	单位
$t_{EEEndurance}$	数据耐久性		100,000		擦/写次数
$t_{EERetention}$	数据保持		10		年
$t_{EEErase}$	擦除周期			2.9	ms
$t_{EEWrite}$	写周期			2.9	ms

表 21-1 E^2PROM 特性