关键词

- TX522A 读卡模块
- Mifare 1 S50, S70
- 扇区 密钥
- 一卡多用 一卡通
- 非接触式 智能卡

摘 要

TX522 系列模块是基于 13.56MHz 频率的 Mifare 卡读写模块，符合 ISO14443A 标准，可支持 Mifare1 S50、Mifare1 S70、Mifare Light、Mifare UltraLight、Mifare Pro。TX522 系列 Mifare 读写模块具有易用、高可靠、多种接口、体积小等特点，可帮助用户方便、快捷地将当今最流行的非接触式 IC 卡技术融入到系统中，提高产品的档次。

本文详细介绍了 TX522A 模块的基于 51 单片机的 C51 库函数的使用方法。

目　录

1. 适用范围

本文对 TX522A Mifare 卡读写模块在 51 单片机下的 C51 库函数做了非常详细的讲解，使用 51 系列单片机与 TX522A 进行通讯的 TX522A 用户参考本文提供的各函数，可以非常容易的编写操作 Mifare 卡的应用程序。不使用 51 系列单片机，而使用其它处理器的 TX522A 用户，也可以参考本文所描述提供的通讯格式和函数，将本文提供的函数很容易的移植到其它处理器上。

对于 C51 函数的调用，用户可不必关心数据块格式，只要理解函数的功能，输入、输出参数即可。当用户自己编写函数（包括非 C51 下的函数）时，就需了解数据块格式，必须按照数据块的格式来编写函数。

2. TX522A 简介

TX522A 是一个简单的 SPI 串行读写模块。用户必须通过主机（包括单片机、ARM、DSP 以及 PC 机等）向 TX522A 模块发送命令来对 TX522A 进行读写控制。本应用指南将重点描述 TX522A 与主机之间的 SPI 串行通信协议和命令。

2.1 引脚描述

表 1 与用户 MCU 接口 J2

接 口	管脚	符 号	IO 类型	功能描述
J2[1] TX522A	J2-1	SPI_CLK	输入/I	SPI 接口的时钟，由外部 MCU 产生
	J2-2	SPI_DATA	双向/IO	SPI 接口的数据线，可双向传输数据
	J2-3	SPI_nCS	双向/ IO	SPI 接口的片选线，低电平有效，同时作为模块与外部 MCU 数据传输的启动信号线
	J2-4	VCC	电源/Power	电源正极
	J2-5	NC		请悬空
	J2-6	GND	地/Power	地
	J2-7	CTRL	输出/O	控制信号输出
	J2-8	BZ	输出/O	外部蜂鸣器驱动电路控制信号

（1） J2 为模块与用户控制器的接口。

表 3 与用户 MCU 接口 J3[1]

接 口	管脚	符 号	IO 类型	功能描述
J3 TX522A	J3-1	NC	-	预留未来使用，请勿连接
	J3-2	NC	-	预留未来使用，请勿连接
	J3-3	NC	-	预留未来使用，请勿连接
	J3-4	NC	-	预留未来使用，请勿连接
	J3-5	GND	地/Power	地

2.2 技术参数

表 2 TX522 模块技术参数表

产品型号	TX522 系列
功 耗	60 毫安/直流 5V；
工作频率	13.56 兆赫兹
读卡距离	20～100 毫米（mifare1 卡）
接口方式	自定义 SPI
数据传输速率	自定义 SPI：50K
支持卡类型	mifare1 S50、mifare1 S70、mifare UltraLight、mifare Pro
尺 寸	天线一体化（带后缀 T）：58mm×34.5mm×2.5mm

2.3 极限参数

每个管脚的对地电压 ……-0.5～+5.5V
天线一体化模块 Vcc 对地的电压……-0.3～+12V
天线分体模块 Vcc 对地的电压……-0.3～+7V
每个管脚的最大 I_{OL} ……20mA
湿度（相对湿度）……5～95％

超出“绝对最大额定值”列出的值的条件下工作会造成器件的永久损坏。以上列出的是器件正常工作的额定值，并未涉及器件在这些条件或超出这些条件下的功能操作。器件不能长时间工作在绝对最大额定值条件下，否则会影响其可靠性。

2.4 直流特性

VCC=+5.0V，器件都工作在建议的温度范围-30～85℃条件下，除非特别说明。

表 3　TX522B 模块的直流特性

符 号	参　数	测试条件	最小值	典型值(1)	最大值	单位
VCC	TX522T工作电压		＋3.6	＋5.0	＋12	V
VDD	输出电压			3.3V		V
T_{OK}	上电后稳定工作时间		5			ms
I_{CC}	电流消耗	读卡芯片配置成功		60		mA
V_{IL}	输入低电平		-	0.7	1.3	V
V_{IH}	输入高电平		2	3.3	-	V
V_{OL}	输出低电平	I_{OL}=20mA		0.6	1.0	V
		I_{OL}=3.2mA		0.2	0.3	V
V_{OH}	输出高电平	I_{OH}=-20uA	3	3.1		V
I_{IL}	逻辑低电平输入电流	Vpin=0.4V			-80	μA
I_{TL}	逻辑1到0变化电流	Vpin=2V	-30		-450	μA
I_{OL}	低电平时的灌电流				-20	mA
I_{OH}	高电平时的拉电流				20	μA
C_{IO}	管脚输入电容				15	pF
T_{OP}	工作温度(I)		-30		+85	℃
T_{STR}	存储温度		-55		+125	℃

(1)　典型值是难以保证的，这个值是在常温条件下测试得到。

(2)　模块上电后，必须等待 5ms 以上时间才能稳定工作。

(3)　如果需要更高输入电压，请联系我们

2.5 封装及机械尺寸

天线分体式 TX522 封装尺寸如**错误！未找到引用源。**所示，为保证用户目标板和 TX522 系列模块接口尺寸一致，用户可以在 http://www.txrfid.com 上获取 TX522 系列模块元件库和 PCB 封装。

单位: mm

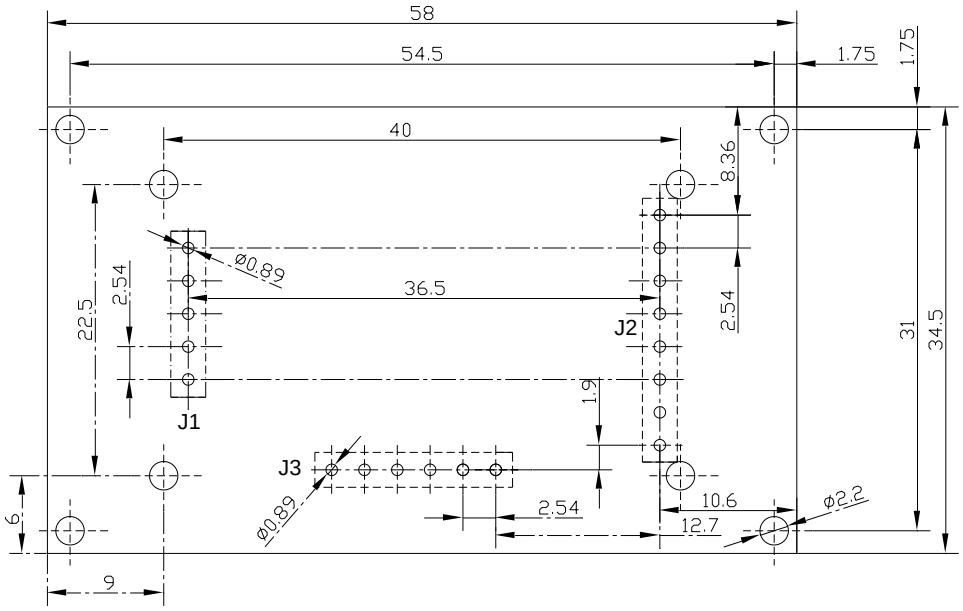

图 1 TX522T 封装

3. TX522A 读卡模块数据传输协议

3.1 自定义 SPI 接口描述

自定义 SPI 接口采用三线制主从通信，三条通讯线包括：SPI_CLK、SPI_DATA、SPI_nCS。

- 主机：用于控制 TX522A 的控制器，可以是 51 单片机、ARM、DSP 等。在一次数据传输中可以是数据接收方或发送方，但必须产生 SPI_CLK 信号。
- 从机：TX522A 模块。在一次数据传输中可以是数据接收方或发送方，接收 SPI_CLK 信号。
- 数据发送方：在一次传输中，控制 SPI_nCS 信号和写数据的一方。
- 数据接收方：在一次传输中，响应 SPI_nCS 信号和读数据的一方。

各通讯线的功能描述如下：

(1) SPI_nCS：数据发送使能线，双向，由数据发送方控制。空闲状态下，主机和从机都释放 SPI_nCS 线（设为输入状态即可释放，普通 51 单片机可以置 1），SPI_nCS 被 TX522A 内的电阻上拉到高电平。数据发送方拉低 SPI_nCS 以启动传输；结束一帧数据传输后，数据发送方置高 SPI_nCS 以释放 SPI_nCS 线；建议主机的 SPI_nCS 设置为准双向模式或者开漏模式加外部上拉电阻，这样，通信过程中就不用切换模式了，普通 51 单片机就是准双向模式。如果主机只有推挽输出模式，则不要输出高电平。同时建议 SPI_nCS 连接到中断脚，以中断方式检测从机的数据发送。（TX522A 内部有上拉电阻）。

(2) SPI_CLK：时钟线，单向，始终由主机控制。在数据传送过程中，主机产生串行时钟信号，以同步传输。SPI_CLK 信号必须严格遵守时序规范，否则将出现通信错误。SPI_CLK 的时序规范见 3.3；建议主机的 SPI_CLK 设为准双向或者输出模式，通信过程中不用切换模式。

(3) SPI_DATA：数据线，双向，由数据发送方控制。由数据发送方在 SPI_CLK 时钟信号的相应位置设置。SPI_DATA 在一次传输开始时还同时作为数据接收方的响应信号，当数据发送方准备发送数据时 SPI_nCS 为高，SPI_DATA 为低，当数据发送方拉低 SPI_nCS 发起数据传输时，数据接收方应该拉高 SPI_DATA 响应，以表示准备好接收数据。建议主机把 SPI_DATA 设置为准双向模式或者开漏模式加外部上拉电阻。

空闲状态定义：空闲时，主机和从机释放 SPI_nCS 线，SPI_nCS 线被上拉到高电平；SPI_CLK 输出低电平；SPI_DATA 被主机和从机同时拉低。

3.2 协议描述

通信必须先由主机发送命令和数据给 TX522A，TX522A 执行命令完毕后，再将执行命令后的状态和响应数据发回给主机。

开始通信前，收发双方(主机和 TX522A)必须处于空闲状态。

首先主机发出 nCS 下降沿信号，然后等待 TX522A 在 DATA 线上的响应。若在 50ms 内未检测到此响应，应退出本次传输，将错误代码返回给主程序，由主程序进行错误处理。若

TX522A 正确响应，则主机可将命令和数据发送出去。

主机发送完命令和数据后，等待 TX522 发回的状态和响应数据。也即等待 nCS 线上的下降沿的产生。主机可用软件查询，也可用外部中断查询 nCS 上的下降沿。主机在发送完命令和数据后，若在 500ms 内未检测到 TX522 的传输请求信号(nCS 上的下降沿)，应退出本次传输，可重新向 TX522A 发送命令和数据，或且向主程序报告错误代码。若主机检测到了 TX522A 的传输请求，则可接收 TX522A 发送来的状态和数据。

若主机或从机中的一方有数据要发送给另一方，数据发送方首先检查总线是否空闲，即检查 SPI_nCS 是否为高电平。若总线空闲发送方把 SPI_DATA 设为输入（在 51 单片机中将 IO 口置 1）；接着发送方拉低 SPI_nCS 线，准备启动传输。

数据接收方检测到 SPI_nCS 为低时，应把 SPI_DATA 设为输入，准备接收数据。

数据发送方在拉低 SPI_nCS 后，应检测 SPI_DATA 上状态，当为高电平时，说明数据接收方已经响应（准备好数据接收），此时数据发送方应将 SPI_DATA 口设为输出，开始启动传输。

传输结束后，数据发送方重新释放 SPI_nCS，并拉低 SPI_DATA。主机把 SPI_CLK 拉低。总线又处于空闲状态了。

3.3 自定义 SPI 接口时序

无论数据传输的方向如何，SPI 接口信号的波形总是如图 2 所示。

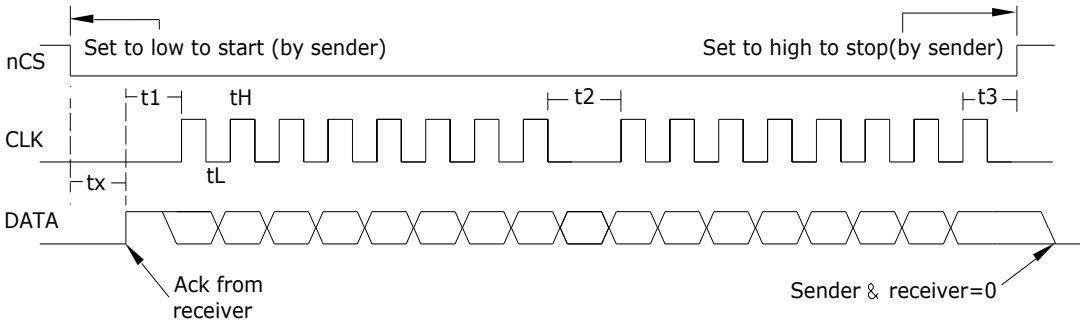

图 2 自定义 SPI 接口时序图

由图 2 可以看出在 nCS 为低电平时，时钟 CLK 和数据线 DATA 上的信号才有效。数据线 DATA 在 CLK 为低时变化，在 CLK 为高时数据线 DATA 应保持稳定，数据接收方在 CLK 高电平时读取数据。

从数据发送方请求开始至数据接收方响应的时间 tx 是不确定的，取决接收数据方内的 MCU 是否忙。因此，主机在发送数据给 TX522A 时，有必要设置一个看门狗定时器对 TX522A 的响应进行监视。主机必须根数据传输的方向严格控制各时间，以确保数据传输无误。注意：在数据传输的不同方向时，对时间 t1、t2、t3、tH 和 tL 都有各自不同的要求。

表 4 时序中各时间参数

符合	含义	主机→TX522	TX522→主机
t1	数据接收方响应至主机产生第一个 CLK 上升沿的时间	>7μ s	>14μ s
t2	两个字节传输之间，CLK 低电平的持续时间	>14μ s	>16μ s
t3	传输最后字节的最后一位的 CLK 上升沿至 nCS 上升沿的时间	任意	>9μ s
tH	CLK 信号的高电平持续时间	>7μ s	>6μ s
tL	CLK 信号的低电平持续时间	>9μ s	>6μ s
tx	数据发送方发起传输信号至数据接收方响应的时间。	该时间是不确定的	

3.4 主机→TX522 传送数据（主机写）

主机往 TX522A 传输数据（主机写）包括主机往 TX522A 发送命令和数据。

主机发起数据传输，首先检查总线是否空闲，是则先将 SPI_DATA 设为输入（对于 51 单片机，将 IO 设为 1 即可），SPI_CLK 输出 0。然后在 nCS 线上产生一个下降沿，发出请求数据传输的信号。由于 TX522A 的 SPI_DATA 口空闲时为低，因此 SPI_DATA 线上线与后的状态为低。此后主机应查询 SPI_DATA 线是否为 1 来检测 TX522A 响应。若在 50ms 内未检测到响应，应退出本次传输。

如果 TX522A 检测到了主机在 SPI_nCS 上产生的下降沿，TX522A 将释放 SPI_DATA，SPI_DATA 被上拉到高电平，以响应主机发起的数据传输请求。主机检测到 SPI_DATA 为高电平后，应将主机的 SPI_DATA 口设置为输出，本次数据传输开始。此后主机应产生满足时序要求的 CLK 信号，并根据 CLK 对应时刻将数据送到 SPI_DATA 上，TX522A 将在 CLK 为高时读取 SPI_DATA 线上的数据。主机将所有数据传输完毕后，应在 nCS 线上产生一个上升沿以结束本次传输。之后，主机应将本身的 SPI 设置为空闲状态，准备下一次传输。

主机往 TX522A 传输数据（主机写）的过程如表 5 所示。

表 5 主机往 TX522A 传输数据（主机写）过程表

序号	发起者	动作	接收者	说明
1	主机	检查总线是否空闲（nCS 是否为 1）	-	空闲才能启动传输
2	主机	置 DATA 为输入，CLK=0，nCS=↓	TX522A	数据传输开始请求
3	TX522A	TX522A 查询到 nCS=↓后： DATA=输入（上拉到 1）	主机	本次数据传输响应（接收准备好了）
4	主机	主机查询到 DATA 为 1 后： 置 DATA 输出，并输出串行数据	TX522A	数据传输
5	主机	CLK=↑、延时、↓	TX522A	主机产生时钟脉冲、高电平时 TX522A 读数据
6	主机	重复动作 3、4，传送 N×8 位	TX522A	数据传输 N 字节
7	主机	nCS=↑，置 DATA＝0，CLK=0	TX522A	本次数据传输结束，主机空闲
8	TX522A	查询 nCS=↑和 DATA=0 后，置 DATA=0		TX522A 空闲

3.5 TX522A 往主机传送数据（主机读）

TX522A 往主机传送数据（主机读）是指主机从 TX522A 读出数据。在 TX522A 往主机传送数据前，主机应先向 TX522A 发送读取数据的命令。

主机读过程中，响应信号、SCLK 信号由主机产生，nCS 信号和 DATA 信号由 TX522A 产生。

当 TX522A 需传送数据给主机时，首先检查总线是否空闲，是则在 nCS 线上产生一个下降沿，发出请求数据传输的信号，然后等待主机的响应。主机可以通过查询或中断的方式判断 SPI_nCS 上是否出现低电平。当主机发现 TX522A 的数据传送请求后，应设置 SPI_DATA 为输入准备接收数据，本次数据传输开始。主机应该在设置其 SPI_DATA 为输入后，根据时序要求产生 CLK 信号并 CLK 为高时读取 DATA 线上的数据。TX522A 将在 CLK 的上升沿把数据传送到 DATA 上。数据传输完毕后，TX522A 将会在 nCS 线上产生一个上升沿，主机应检测 nCS 上的信号以判断数据传输是否结束。当主机检测到 TX522A 在 nCS 上产生的上升沿后，将 CLK 和 DATA 置低，本次数据传输结束。

主机往 TX522A 传输数据（主机写）的过程如表 6 所示。

表 6 TX522A 往主机传输数据（主机读）过程表

序号	发出者	动作	接收者	动作说明
1	TX522A	检查总线是否空闲（nCS 是否为 1）	-	空闲才能启动传输
2	TX522A	设置 DATA 为输入，nCS=↓	主机	本次数据传输开始请求
3	主机	主机查询到 nCS=↓后： 设 DATA 为输入（被上拉到 1）	TX522A	本次数据传输响应（接收准备好了）
4	TX522A	DATA=串行数据	主机	数据传输
5	主机	SCLK=↑、延时、↓	TX522A	主机产生时钟，且读取数据
6	双方	重复动作 3、4，传送 N×8 位	双方	数据传输 N 字节
7	TX522A	nCS=↑，DATA=0	主机	本次数据传输结束
8	主机	查询到nCS=↑和DATA=0后置DATA＝0，CLK=0		主机空闲

3.6 数据块格式

3.6.1 主机→TX522（命令模式）

表 7 主机发送给 TX522A 的数据块格式

SEQNR	COMMAND	LENGTH	DATA[0…N]	BCC
INFO[0]	.	.	. . . .	INFO[n]

SEQNR： 1 Byte 数据交换包的序号
COMMAND：1 Byte 命令字符
LENGTH： 1 Byte 数据的长度
DATA[…]： Len Byte 数据字节
BCC： 1Byte 的 BCC 校验

3.6.2 TX522→主机（响应模式）

表 8 TX522A 发送给主机的数据块格式

SEQNR	STATUS	LENGTH	DATA[0…N]	BCC
INFO[0]	.	.	. . . .	INFO[n]

SEQNR： 1 Byte 数据交换包的序号
STATUS： 1 Byte 状态字符
LENGTH： 1 Byte 数据的长度
DATA[…]： Len Byte 数据字节
BCC： 1Byte 的 BCC 校验

3.6.3 数据块格式描述

数据交换包的序号由主机发送数据块时产生，取值范围为 0-255。在经过一次正确的数据交换后，主机在发送下一个命令时，将数据包的序号加 1。TX522A 返回最近接收的包序号。通常主机应用程序最好检查命令/响应包交换时的数据包的序号。

当 TX522A 执行命令时出现了任何错误，响应包中的数据长度为 0（*LENGTH* = 0）。

BCC 校验码计算数据块中所有的 INFO 字节。然后将结果作为数据块的最后一个字节，BCC 校验码的计算如下式所示：

INFO[n] = BCC = ~（INFO[0] ⊕ INFO[1] ⊕ ... ⊕ INFO[n-1]） (⊕ ... XOR、~…NOT)

通常主机应该判断 TX522A 发送数据块的 BCC 校验码。

4. TX522A 的 C51 函数

为了加速用户对 TX522A 的开发进程，TX522A 提供了 C51 函数库，使用 C51 单片机的客户可直接调用这些函数库中的函数，其它系列的微控制器可参考 C51 函数库来进行移植，移植过程只要进行少量的修改。

4.1 函数列表

表 9 TX522A 提供的串行通讯库函数列表

函数名称	命令值	输入参数(发送)	输出参数（接收）	功能描述
TX_Config	0x52	--	--	复位且配置模块
TX_Close	0x3F	--	--	关闭模块
TX_Get_Info	0x4F	--	*Info	读取模块信息
TX_Load_Key	0x4C	KeyAB,Sector,*Key	--	改变存贮在模块内密匙区中的密钥
TX_Get_CardSnr	0x10	ReqCode,*TagType, *Sak,*SnrLen, *Snr	--	激活卡片并获取卡号
TX_Write_Auth	0x11	KeyAB, Key_Sector , Block, idata *Data	--	带验证的写操作
TX_Read_Auth	0x12	KeyAB, Key_Sector , Block	*Data	带验证的读操作
TX_WriteValue_Auth	0x13	KeyAB,Key_Sector, Block, *Value	--	带验证的写值块操作
TX_Value_Auth	0x14	KeyAB,Key_Sector, ValueMode,Block, *Value, Trans_Block	--	带验证的值块操作
TX_ReadValue_Auth	0x15	KeyAB,Key_Sector, Block,	*Value	带验证的读值块操作
TX_Halt	0x45	--	--	将卡置于挂起模式
TX_Reset	0x4E	Msec	--	关闭天线输出数 ms，使卡复位
TX_Request	0x41	ReqCode	*TagType	请求卡，检查在有效范围内是否有卡
TX_Casc_Anticoll	0x74	Bcnt, Select_Code	*SNR	可实现三层防碰撞协议
TX_Casc_Select	0x75	Select_Code, *SNR	*Sak	可实现三层选择
TX_Auth2	0x72	KeyAB,Sector,Key_Sector	--	使用模块内部密钥区中的密码对指定的卡的扇区 Sector 进行验证
TX_Auth_Key	0x73	KeyAB,Sector,*Key	--	直接密码验证
TX_Write	0x47	Block ,*Data	--	向卡中指定块写入一 16 字节的数据块
TX_Read	0x46	Block	*Data	从卡中指定块中读出一个 16 字节的块
TX_Value	0x70	ValueMode,Block,*Value, Trans_Block	--	包含加、减、恢复函数，并可进行不同块之间的自动传送
TX_Set_Control_Bit	0x50	--	--	将控制位置为高电平
TX_Clr_Control_Bit	0x51	--	--	将控制位置为低电平
TX_Buzzer	0x60	Freguence,Time	--	输出驱动蜂鸣器信号，能控制蜂鸣器频率和动作时间

表 9 所有函数的返回值都是该函数执行后的状态结果，具体返回值请查看各函数说明。如果各函数有返回数据，则都是以指针的形式返回。

除表 9 提供的通过串口与 TX522A 通讯的库函数，表 10 列出了 3 个其它函数。

表 10 TX522A 提供的其它库函数列表

函数名称	输入参数(发送)	输出参数（接收）	功能描述
SPI_Init	--	--	主机 SPI 初始化

4.2 函数返回状态值列表

表 11 TX522A 库函数的用到的状态值列表

名称	值	描述
OK, COMM_OK	0	函数调用成功
NO_TAG_ERR	1	在有效区域内没有卡
CRC_ERR	2	从卡中接收到了错误的 CRC 校验和
EMPTY	3	值溢出
AUTH_ERR	4	不能验证
PARITY_ERR	5	从卡中接收到了错误的校验位
CODE_ERR	6	通信错误
SENDR_ERR	8	在防冲突时读到了错误的串行码
KEY_ERR	9	证实密码错
NOT_AUTH_ERR	10	卡没有验证
BIT_COUNT_ERR	11	从卡中接收到了错误数量的位
BYTE_COUNT_ERR	12	从卡中接收了错误数量的字节
TRANS_ERR	14	调用 Transfer 函数出错
WRITE_ERR	15	调用 Write 函数出错
INCR_ERR	16	调用 Increment 函数出错
DECR_ERR	17	调用 Decrment 函数出错
READ_ERR	18	调用 Read 函数出错
COLL_ERR	24	冲突错
ACCESS_TIMEOUT	27	访问超时
QUIT	30	上一次了送命令时被打断
CHK_WR_OK	0	Check Write 正确
CHK_WR_FAILED	1	Check Write 出错
CHK_WR_COMP_ERR	2	Check Write:写出错（比较出错）
COMM_ERR	255	串行通信错误

4.3 函数描述

下面是 C51 函数声明，包含在头文件“TX522A.h”中，写应用程序时，将其包含在应用函数中即可。对于 C51 函数的调用，用户可不必关心数据块格式，只要理解函数的功能，输入、输出参数即可。当用户自己编写函数（包括非 C51 下的函数）时，就需了解数据块格式，必须按照数据块的格式来编写函数。

4.3.1 配置—Config

函数原型：uchar TX_Config(void);
输入参数：无
输出参数：无
函数返回：可能的状态值：OK，QUIT，COMM_ERR。
功能描述: 对模块进行初始化，初始化成功后，模块上的指示灯将点亮，此时天线发射载波信号，任何进入天线感应区的卡可得电进入 IDLE 状态，可使用任一函数对卡进行操作。此时读卡芯片完全被激活，所消耗的电流最大。模块上电后，模块内部会自动执行初始化，指示灯将点亮，因此上电后，用户可不用执行该函数，而直接进行其它操作。该函数可用于不上电情况下的重新初始化。
数据块格式描述：
主机→TX522 命令模式）：
SEQNR： 0 （可自定义）
COMMAND： 0x52
LENGTH： 0
DATA[…]： 无

TX522→主机（响应模式）：
SEQNR： 0
STATUS： OK，QUIT，COMM_ERR
LENGTH： 0
DATA[…]： 无

4.3.2 关闭—Close

函数原型：uchar TX_Close(void);
输入参数：无
输出参数：无
函数返回：可能的状态值：OK， COMM_ERR。
功能描述: 此函数将关闭 TX522A 模块，指示灯熄灭，天线不发送载波信号，模块消耗的电流最小，在此状态在，模块不能使用。若要重新使用模块，需要调用 TX_Config()函数对 TX522A 重新进行配置。
数据块格式描述：
主机→TX522 命令模式）：
SEQNR： 0 （可自定义）
COMMAND： 0x3F
LENGTH： 0
DATA[…]： 无

TX522→主机（响应模式）：
SEQNR： 0
STATUS： OK， COMM_ERR
LENGTH： 0
DATA[…]： 无

4.3.3 获取信息—Get Info

函数原型：uchar TX_Get_Info(uchar idata *Info);

输入参数：无

输出参数：模块的信息*Info，Info 为保存信息空间的首地址。Info[0]～Info[4]为模块类型标识，Info[5]～Info[8]为读卡芯片的序列号，Info[9]为固件版本号，高四位为版本号的整数，取值从 1 到 15，低四位为版本号的小数，取值从 0 到 9。

函数返回：可能的状态值：OK，QUIT，COMM_ERR。

功能描述: 获取 TX522A 模块的信息。

数据块格式描述：

主机→TX522 命令模式）：

SEQNR： 0 （可自定义）

COMMAND： 0x4F

LENGTH： 0

DATA[…]： 无

TX522→主机（响应模式）：

SEQNR： 0

STATUS： OK，QUIT，COMM_ERR

LENGTH：10

DATA[0.9]：产品类型标识及版本

4.3.4 装载密钥—Load_Key

函数原型：uchar TX_Load_Key(uchar KeyAB,uchar Key_Sector,uchar idata *Key)

输入参数：KeyAB：密钥类型（1 字节）。可取值为 KeyAB＝0x00（KEYA）—密钥 A，或 KeyAB＝0x04（KEYB）—密钥 B。

Key_Sector：模块内的密钥区号（1 字节）：取值范围 0～15

*Key：需要装载到模块内密钥区的密钥（6 字节）

输出参数：无

函数返回：可能的状态值：OK，QUIT，AUTH_ERR，COMM_ERR。

功能描述：此函数的作用是将指定的密码（*Key）装载到模块内指定的密钥区（Key_Sector），并非改变 Mifare1 卡内扇区的密码。本函数只对模块进行操作，模块与卡之间没有数据传输。

模块内有 16 个密码区（区号 0——15），称它为密钥区号 Key_Sector。每个区分密钥 A（0x60）和密钥 B（0x61）两个，总共 32 个密码。装载成功后，可用该密钥对 Mifare1 卡进行验证。

在 M1 卡中也有 16 个存储区，称它为扇区号 Sector。若要改变 Mifare1 卡内的密钥，可在用原密码验证通过后，直接用写块数据 TX_Write()函数，将密码块改写。Mifare 卡出厂后的初始密钥为 6 个 FFH，A 和 B 密钥都一样。

数据块格式描述：

主机→TX522 命令模式）：

SEQNR： 0 （可自定义）

COMMAND：0x4C

LENGTH： 8
DATA[0]： Mode
DATA[1]： Key_Sector
DATA[2]： Key[0]
…
DATA[7]： Key[5]

TX522→主机（响应模式）：
SEQNR： 0
STATUS： OK，QUIT，AUTH_ERR，COMM_ERR
LENGTH： 0
DATA[0]： 无

4.3.5 激活卡片并获取卡号—Get_CardSnr

函数原型：uchar TX_Get_CardSnr(uchar ReqCode, uchar idata *TagType, uchar idata *Sak, uchar idata *SnrLen, uchar idata *Snr);

输入参数： ReqCode：请求模式 ReqCode 取值为 1 或 0
ReqCode＝0（IDLE），请求天线范围内 IDLE 状态的卡（HALT 状态的除外）
ReqCode＝1（ALL），请求天线范围内的所有卡。

输出参数：
(1) *TagType：请求应答：2 个字节的卡片类型，其意义见 4.3.13。
(2) *Sak：最后一级选择应答的应答,其意义见 4.3.14。
(3) *SnrLen：返回卡片序列号的长度。
(4) *Snr：返回卡片的序列号。
函数返回：可能的状态：OK，QUIT，COMM_ERR。
功能描述：该命令为后面将要介绍的请求、防碰撞和选择三条命令的组合。成功执行该命令后即可进行验证及后续操作。

数据块格式描述：
主机→TX522 命令模式）：
SEQNR： 0 （可自定义）
COMMAND：0x10
LENGTH： 1
DATA[0]： ReqCode

TX522→主机（响应模式）：
SEQNR： 0
STATUS： OK，QUIT，COMM_ERR
LENGTH： 4 字节+序列号的长度，Mifare1 S50、S70、Light 卡：8 字节，
Mifare0 UltraLight 和 Mifare3 Desfire 卡：11 字节
DATA[0..1]： *TagType：请求应答，2 个字节的卡片类型
DATA[2]： *Sak：最后一级选择应答的应答
DATA[3]： *SnrLen：返回卡片序列号的长度

DATA[4..4+ SnrLen]： *Snr：返回卡片的序列号

4.3.6 带验证的写—Write_Auth

函数原型：uchar TX_Write_Auth(uchar KeyAB, uchar Key_Sector ,uchar Block, uchar idata *Data)

输入参数：

(1) KEYAB--密钥 AB（1 字节）： 0x00——密钥 A
0x04——密钥 B

(2) Key_Sector：模块内的密钥区号（1 字节）：取值范围 0～15。

(3) Block--卡块号（1 字节）： S50：1~63
S70：1~255

(4) *Data：16 字节数据指针，Data 为写入的 16 字节数据的首地址。

输出参数： 无

函数返回：执行后可能返回：OK，QUIT，NO_TAG_ERR，CRC_ERR，NOT_AUTH_ERR，PARITY_ERR ， BIT_COUNT_ERR ， COMM_ERR ， CHK_WR_FAILED ，CHK_WR_COMP_ERR。

功能描述：该函数在 Get_CardSnr 后执行，先对卡内某一块进行验证，成功后对指定块进行写操作（只要访问条件允许），其中包括位于扇区尾的密码块，这是更改密码的唯一方法。该函数在写入数据后会立即进行读操作并进行数据写入正确性判断。

数据块格式描述：

主机→TX522 命令模式）：

SEQNR： 0 （可自定义）

COMMAND：0x11

LENGTH： 19

DATA[0]： KEYAB

DATA[1]： Key_Sector

DATA[2]： Block

DATA[3]： 所要写的第一个字节

：

DATA[18]： 所要写的最后一个字节

TX522→主机（响应模式）：

SEQNR： 0

STATUS： OK， QUIT， NO_TAG_ERR， NOT_AUTH_ERR， WRITE_ERR，BIT_COUNT_ERR，COMM_ERR

LENGTH： 0

DATA[0]： 无

补充说明：

注意：在将数据写到卡片上的某一扇区时，一定要小心。因为有些 block 中存储了密码数据以及存储允许使能数据。特别是每一个扇区的 Block3 中存放了该扇区的存取条件，包含有 KEYA，KEYB 及该扇区的控制字。Mifare 1 卡片出厂时的 Block3 有缺省值，为：“a0a1a2a3a4a5ff078069b0b1b2b3b4b5”,共 16 个 Bytes。

涉及 Mifare 1 卡片的存储结构等信息，请参考 Mifare 1 卡片数据手册。

程序员在使用 Mifare 1 卡片做应用时，一定要清清楚楚记住每一个扇区的 Block3 的数据，这样也就记住了扇区的密码和存取控制字。否则，扇区的存储空间将不执行 Read/Write 等操作而失效。

任何试图用任何方式来读写不知密码的卡片或某一扇区都是徒劳无益的。

卡片应放在安全的地方，即不要放在离模块天线较近的地方。因为当模块对其它卡片执行某些指令时，有可能无意间对这一卡片进行了读/写等操作，从而操作卡片的失效。

4.3.7 带验证的读—Read_Auth

函数原型：uchar TX_Read_Auth(uchar KeyAB, uchar Key_Sector ,uchar Block,uchar idata *Data)

输入参数：

(1) KEYAB--密钥 AB（1 字节）： 0x00——密钥 A
0x04——密钥 B

(2) Key_Sector：模块内的密钥区号（1 字节）：取值范围 0～15。

(3) Block--卡块号（1 字节）： S50：1~63
S70：1~255

输出参数：*Data：Data 为读回 16 字节数据的首地址。

函数返回：执行后可能返回：OK，QUIT，NO_TAG_ERR，CRC_ERR，NOT_AUTH_ERR，PARITY_ERR，BIT_COUNT_ERR，COMM_ERR。

功能描述：该函数在 Get_CardSnr 后执行，先对卡内某一块进行验证，成功后读 Mifare 卡中相应块的数据。密码数据不能被读取。

数据块格式描述：

主机→TX522 命令模式）：

SEQNR： 0 （可自定义）

COMMAND：0x12

LENGTH： 3

DATA[0]： KEYAB

DATA[1]： Key_Sector

DATA[2]： Block

TX522→主机（响应模式）：

SEQNR： 0

STATUS： OK ， QUIT ， NO_TAG_ERR ， CRC_ERR ， NOT_AUTH_ERR ，PARITY_ERR，BIT_COUNT_ERR，COMM_ERR

LENGTH： 16

DATA[0]： 所访问块的第一个字节

：

DATA[15]： 所访问块的最后一个字节

4.3.8 带验证的写值块操作—WriteValue_Auth

函数原型：uchar TX_WriteValue_Auth(uchar KeyAB, uchar Key_Sector , uchar Block, uchar idata *Value);

输入参数：

(1) KEYAB--密钥 AB（1 字节）： 0x00——密钥 A
0x04——密钥 B

(2) Key_Sector：模块内的密钥区号（1 字节）：取值范围 0～15。

(3) Block--卡块号（1 字节）： S50：1~63
S70：1~255

(4) *Value：4 字节数据指针，用来存储减少值或增加值，当进行恢复操作时，该值为空值。Value 是减少值或增加值存放的首地址，存放时，低地址存放高字节。

输出参数： 无

函数返回：执行后可能返回：OK, QUIT, NO_TAG_ERR, BIT_COUNT_ERR, TRANS_ERR, CODE_ERR, COMM_RERR。

功能描述：该函数在 Get_CardSnr 后执行，先对卡内某一块进行验证，成功后往 Mifare 卡中相应块写入值块格式的数据（只要访问条件允许）。该函数在写入数据后会立即进行读操作并进行数据写入正确性判断。

数据块格式描述：

主机→TX522 命令模式）：

SEQNR： 0 （可自定义）
COMMAND：0x13
LENGTH： 7
DATA[0]： KEYAB
DATA[1]： Key_Sector
DATA[2]： Block
DATA[3]： Value(LL)
DATA[4]： Value(LH)
DATA[5]： Value(HL)
DATA[6]： Value(HH)

TX522→主机（响应模式）：

SEQNR： 0
STATUS： OK, QUIT, NO_TAG_ERR, BIT_COUNT_ERR, TRANS_ERR, CODE_ERR, COMM_RERR
LENGTH： 0
DATA[0]： 无

4.3.9 带验证的值块操作—Value_Auth

函数原型：uchar TX_Value_Auth (uchar KeyAB, uchar Key_Sector ,uchar ValueMode, uchar Block, uchar idata *Value, uchar Trans_Block);

输入参数：

(1) KEYAB--密钥 AB（1 字节）： 0x00——密钥 A
0x04——密钥 B

(2) Key_Sector：模块内的密钥区号（1 字节）：取值范围 0～15。

(3) ValueMode：0xC0—减； 0xC1—加； 0xC2—恢复

(4) Block--卡块号（1 字节）： S50：1~63

S70：1~255

(5) *Value：4 字节数据指针，用来存储减少值或增加值，当进行恢复操作时，该值为空值。Value 是减少值或增加值存放的首地址，存放时，低地址存放高字节。

(6) Trans_Block：传输块地址，取值范围：S50：1～63；S70：1～255。

输出参数： 无

函数返回：执行后可能返回：OK, QUIT, NO_TAG_ERR, BIT_COUNT_ERR, TRANS_ERR, CODE_ERR, COMM_RERR。

功能描述：该函数在 Get_CardSnr 后执行，先对卡内某一块进行验证，成功后对卡内的某一块进行加、减或数据备份，该块必须为值块格式，并支持自动传送。若卡块号与传输块号相同，则将操作后的结果写入原来的块内；若卡块号与传输块号不相同，则将操作后的结果写入传输块内，结果传输块内的数据被覆盖，原块内的值不变。当模式为“恢复”时，“值”无意义。

数据块格式描述：

主机→TX522 命令模式）：

SEQNR： 0 （可自定义）

COMMAND：0x14

LENGTH： 9

DATA[0]： KEYAB

DATA[1]： Key_Sector

DATA[2]： ValueMode

DATA[3]： Block

DATA[4]： Value(LL)

DATA[5]： Value(LH)

DATA[6]： Value(HL)

DATA[7]： Value(HH)

DATA[8]： Trans_Block

TX522→主机（响应模式）：

SEQNR： 0

STATUS： OK, QUIT, NO_TAG_ERR, BIT_COUNT_ERR, TRANS_ERR, CODE_ERR, COMM_RERR

LENGTH： 0

DATA[0]： 无

4.3.10 带验证的读值块操作—ReadValue_Auth

函数原型：uchar TX_ReadValue_Auth(uchar KeyAB, uchar Key_Sector , uchar Block, uchar idata *Value);

输入参数：

(1) KEYAB--密钥 AB（1 字节）： 0x00——密钥 A
0x04——密钥 B

(2) Key_Sector：模块内的密钥区号（1 字节）：取值范围 0～15。

(3) Block--卡块号（1 字节）： S50：1~63
S70：1~255

输出参数： *Value：Value 为读回 4 字节数据的首地址。

函数返回：执行后可能返回：OK，QUIT，NO_TAG_ERR，BIT_COUNT_ERR，TRANS_ERR，CODE_ERR，COMM_RERR。

功能描述：该函数在 Get_CardSnr 后执行，先对卡内某一块进行验证，然后读出值块。注意读出值块时，会对值块的正确性进行判断，如果值块不合法，返回错误，不会返回值块数据。

数据块格式描述：

主机→TX522 命令模式）：

SEQNR： 0 （可自定义）

COMMAND：0x15

LENGTH： 3

DATA[0]： KEYAB

DATA[1]： Key_Sector

DATA[2]： Block

TX522→主机（响应模式）：

SEQNR： 0

STATUS： OK，QUIT，NO_TAG_ERR，BIT_COUNT_ERR，TRANS_ERR，CODE_ERR，COMM_RERR

LENGTH： 4

DATA[0]： Value(LL)

DATA[1]： Value(LH)

DATA[2]： Value(HL)

DATA[3]： Value(HH)

4.3.11 暂停—Halt

函数原型：uchar TX_Halt(void)

输入参数：无

输出参数：无

函数返回：可能的状态值：OK，QUIT，COMM_ERR。

功能描述：将天线区所选择卡置为挂起状态。如果要进行重新选择，则应用 ALL 模式调用 TX_Request 命令。如果要进行重新选择,也可以将卡离开天线操作区再进入，或执行复位函数 TX_Reset()。

可以配合使用 TX_Request()和 TX_Halt()函数，进行一次性扣费，如卡进入感应区后只扣一次（一元钱），离开后，下次进入再扣一次，若卡在感应区内停留时间较长，也不会扣多一次。

数据块格式描述：

主机→TX522 命令模式）：

SEQNR： 0 （可自定义）

COMMAND：0x45

LENGTH： 0

DATA[0]： 无

TX522→主机（响应模式）：
SEQNR：　0
STATUS：　OK，QUIT，COMM_ERR
LENGTH：　0
DATA[0]：　无

4.3.12 复位—Reset

函数原型：uchar TX_Reset(uchar Msec);
输入参数：Msec：取值 0~255，模块上射频电路关闭时间（以 ms 毫秒为单位），Msec＝0 时，一直关闭。
输出参数：　无
函数返回：可能的状态值：OK，QUIT，COMM_ERR。
功能描述：该函数使模块上的射频电路关闭，关闭的时间由参数 Msec 指定，若 Msec=0，射频电路将一直处于关闭状态，一直到下一个 TX_Request 命令到来。关闭射频电路能使天线内的所有卡复位。

数据块格式描述：
主机→TX522 命令模式）：
SEQNR：　0 （可自定义）
COMMAND：0x4E
LENGTH：　1
DATA[0]：　Msec

TX522→主机（响应模式）：
SEQNR：　0
STATUS：　OK，QUIT，COMM_ERR
LENGTH：　0
DATA[0]：　无

4.3.13 请求—Request

函数原型：uchar TX_Request(uchar Mode,uchar idata *TagType);
输入参数：　Mode：请求模式 Mode 取值为 1 或 0
　　Mode＝0（IDLE），请求天线范围内 IDLE 状态的卡（HALT 状态的除外）
　　Mode＝1（ALL），请求天线范围内的所有卡。
输出参数：*TagType：2 个字节的卡片类型，*TagType 低字节 *(TagType+1)高字节，对于 mifare 1 卡，返回类型为 0x0004,则*TagType ＝ 0x04 ，*(TagType+1)＝0x00。 当发生错误时，不返回任何内容（LENGTH=0）。
*TagType 的含义如表 12 所示：

表 12 *TagType 的含义

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0

保留	序列号大小 00:4bytes 01:7bytes 10:10bytes	保留	任何位为 1 时，则为比特帧防碰撞方式

各种常用卡返回的数据如下表 13 所示：

表 13 常用卡的*TagType 值

卡类型	*TagType	卡类型	*TagType
Mifare1 S50	0x0004	SHC1101	0x0004
Mifare1 S70	0x0002	SHC1102	0x3300
Mifare Light	0x0010	11RF32	0x0004
Mifare UltraLight	0x0044		

函数返回：可能的状态值：OK，QUIT，NO_TAG_ERR，BIT_COUNT_ERR，COMM_ERR。

功能描述：检查在 TX522A 有效范围内是否有卡存在。卡片进入天线区域后，能获能能量，从而得电复位处于 IDLE 模式，TX_Request 函数可用 ALL 或 IDLE 任意一种模式进行请求，卡片均能响应，并返回卡片类型号 TagType（2 个字节）。在选择一张新的卡时必须调用该函数。若对某一张卡成功进行了挂起操作（TX_Halt 命令），卡片将进入 Halt 状态，此时的卡只能响应 ALL 模式的请求，除非该卡离开天线区域然后再重新进入。

给 TX522A 发送 0x41 命令完成请求卡。注意：对同一张卡（不进入 HALT 状态）连续进行请求时，总是一次成功一次失败。

数据块格式描述：

主机→TX522 命令模式）：

SEQNR： 0 （可自定义）

COMMAND： 0x41

LENGTH： 1

DATA[0]： Mode

TX522→主机（响应模式）：

SEQNR： 0

STATUS： OK，QUIT，NO_TAG_ERR，BIT_COUNT_ERR，COMM_ERR 中某一个

LENGTH： 2 (当发生错误时，不返回任何内容（LENGTH=0）)

DATA[0]： tagtype (低字节)

DATA[1]： tagtype (高字节)

补充说明：

TX_Request (IDLE)的使用是很重要的，它可以防止 TX522 多次选择同一卡片。当某一张卡片在天线的有效的工作范围（距离）内，TX_Request (IDLE)指令在成功地读取这一张卡片之后，将一直等待卡片的使用者拿走这一张卡片，直到有新一张的卡片进入天线的有效的工作范围（距离）内。当然，这里的“新一张的卡片”亦可以是刚刚拿开那张卡片。

TX_Request (IDLE)指令是非连续性的读卡指令，只读一次。但有个例外，当某一次 Request 指令读卡片失败时，例如，卡片没能通过密码认证或其他原因而出错时，TX_Request (IDLE)指令将连续地读卡，直到读卡成功才进入非连续性的读卡模式。

TX_Request (IDLE)指令适用于那些需要有人工干预的场合。

TX_Request (ALL)指令的使用和 TX_Request (IDLE)指令刚巧相反，TX_Request (ALL)指令是连续性的读卡指令。当某一张卡片在天线的有效的工作范围（距离）内，TX_Request (ALL)

指令在成功地读取这一张卡片之后，进入对卡片的其他操作。如果其他操作完成之后，程序员又执行 TX_Request (ALL)指令操作，则 TX_Request (ALL)指令将连续性地再次进行读卡操作，而不管这张卡片是否被拿走。只要有一张卡片进入天线的有效的工作范围（距离）内，TX_Request (ALL)指令将始终连续性地再次进行读卡操作。

TX_Request (ALL)指令适用于那些不需要有人工干预的场合，即全自动的场合。例如，宾馆，酒店，高级写字楼等场所的门禁控制系统（Door Access Control），高速公路，停车场等的不停车收费系统（Non-Stop Road Tolling），等等。

后面即将介绍的 TX_Halt()命令函数使被选中的卡进入 HALT 模式，进入该模式的卡只能用 ALL 方式进行请求，也即它对 IDLE 方式的请求不响应。若要使它再次响应 IDLE 请求，唯一方式是使卡复位，如执行 TX_Reset()函数，或使卡离开天线感应区后再进入。

可以配合使用 TX_Request()和 TX_Halt()函数，进行一次性扣费，如卡进入感应区后只扣一次（一元钱），离开后，下次进入再扣一次，若卡在感应区内停留时间较长，不得扣多于一次。方法如下：

```
while(1)
{
    while(TX_Request(IDLE, databuf)!= OK);
    …
    扣费操作
    …
    TX_Halt();
}
```

4.3.14 带层级设置的防碰撞—Casc_Anticoll

函数原型：uchar TX_Casc_Anticoll(uchar Select_Code,uchar Bcnt,uchar idata *SNR);

输入参数：Select_Code：防碰撞层级编码：一层（ANTICOLL1）—0x93；二层（ANTICOLL2）—0x95；三层（ANTICOLL3）—0x97。Bcnt：预选卡已经知道的序列号的的位数，通常都设置 Bcnt=0。

输出参数：*SNR：4 个字节的卡的序列号，低字节放在低地址处*SNR 低字节 *(SNR+i)高字节。若卡的序列号超过 4 个字节（如 Mifare UltraLight），序列号不完整，则最低字节的值为 0x88，表示需要进行更高一级的防碰撞 TX_Casc_Anticoll（0x95）。

函数返回：可能的状态值：OK，QUIT，NO_TAG_ERR，BIT_COUNT_ERR，COMM_ERR。

功能描述：可选级数的防碰撞操作。该函数必须在调用 TX_Request 命令后立即调用。当多个卡片位于天线区域时，执行函数后，能得到序列号最大的那个卡片的序列号中的 4 个字节。卡的序列号长度有三种：4 字节、7 字节和 10 字节。4 字节的只要用一级防碰撞即可得到完整的序列号，如 Mifare1 S50 S70 等；7 字节的要进行二级防碰撞才能得到完整的序列号，前一级所得到的序列号的最低字节为级联标志 0x88，在序列号内只后 3 字节可用，后一级防碰撞能得到 4 字节序列号，两者按顺序连接即为 7 字节序列号，如 UltraLight 和 DesFire 等；10 字节的以此类推，但至今没有此类卡。需要进行二级防碰撞操作，可通过修改参数 Select_Code 来实现。第一级使用 Select_Code＝0x93，第二级使用 Select_Code＝0x95，对于现在所有得 Mifare 卡，不需要进行三级防碰撞，因为所有 Mifare 卡的序列号最大为 7 个字节。当知道了所要选择卡的序列号后，就没有必要执行该 TX_AntiColl 函数。此时，调用 TX_Request 后，直接调用 TX_Select 函数即可。

数据块格式描述：

主机→TX522 命令模式）：
SEQNR： 0 （可自定义）
COMMAND：0x74
LENGTH： 2
DATA[0]： Select_Code
DATA[1]： Bcnt

TX522→主机（响应模式）：
SEQNR： 0
STATUS：OK，QUIT，NO_TAG_ERR，BIT_COUNT_ERR，COMM_ERR
LENGTH：4
DATA[0]：卡的序列号第 1 字节（最低字节）
DATA[1]：卡的序列号第 2 字节
DATA[2]：卡的序列号第 3 字节
DATA[3]：卡的序列号第 4 字节
补充说明：

符合 ISO14443A 标准卡的序列号都是全球唯一的，正是由于唯一性，才能实现防碰撞的算法逻辑。如果有多张卡同时在天线感应区内则这个函数能够找到一张序列号较大的卡来操作。实际上由于天线辐射的磁场能量有限，同时在天线感应区内的所有卡都要从辐射场中吸收，因此同时在天线感应区内的卡不能太多，否则没有一张卡能获得足够的能量来正常工作。

在执行选择 TX_Select 命令之前，若已知所要操作的卡的序列号，则可跳过此步，直接执行选择 TX_Select 命令；若不知道卡的序列号，则必须调用防碰撞函数，得到感应区内卡的序列号。

程序员在发送 TX_Casc_Anticoll 指令，TX522 将返回 SNR，一般地应在程序中对所接收到的 SNR 进行校验，以确保数据的正确性。具体的方法是对所接收到的 SN 的 bit 位进行异或校验。这个工作已经在 TX522A 中完成，用户不必再进行校验。

对于 MIFARE 1 卡片来说，返回某一张卡片的有效序列号 SNR（4 个字节）可能为：007e0a42h。

4.3.15 带层级设置的选择—Casc_Select

函数原型：uchar TX_Casc_Select(uchar SelectCode, uchar idata *SNR,uchar idata *Sak);

输入参数：Select_Code：防碰撞层级编码：一层（ANTICOLL1）—0x93；二层（ANTICOLL2）—0x95;三层（ANTICOLL3）—0x97。该参数应该与之前的 TX_Casc_Anticoll() 中的 Select_Code 参数相同。

*SNR：前一次防碰撞返回的卡的序号，或已知的卡的序列号。无符号 4 字节，低字节放在低地址处。如果卡得序号号大于 4 个字节，则*SNR 的最低字节值为 0x88，表示需要进行更高一级的防碰撞。

输出参数：*Sak：是否选择成功的应答,其意义如表 14 所示：

表 14 *Sak 含义

b7	b6	b5	b4	b3	b2	b1	b0
RFU			RFU		判断是否选择完成	RFU	

bit2 位用来判断是否还有序列号未读出，bit2＝0 时，表示所有序列号输入完毕，选择

成功，bit2＝1 时表示序列号没有全部输入，选择没有完全成功，还要进行下一级的防碰撞和选择操作。

当*Sak＝xxxxx1xx 时，序列号没有完成，还要进行下一级的防碰撞和选择操作。

当*Sak＝xx1xx0xx 时，选择成功，该卡符合 ISO/IEC 14443-4 标准。

当*Sak＝xx0xx0xx 时，选择成功，该卡不符合 ISO/IEC 14443-4 标准。

各种常用卡返回的数据如下所表 15 所示：

表 15 常用卡的*Sak 的值

卡类型	*Sak	卡类型	*Sak
Mifare1 S50	0x08	Mifare0 UltraLight	0x04
Mifare1 S70	0x18	SHC1101	0x22
Mifare1 Light	0x01	11RF32	0x08

函数返回：可能的状态值：OK, QUIT, NO_TAG_ERR, CRC_ERR, PARITY_ERR, BIT_COUNT_ERR,COMM_ERR。

功能描述：该函数选择某一个序列号的卡，与之建立通信连接，并返回是否选择成功的应答*Sak。该函数一般与 TX_Casc_Anticoll()配对使用，进行多级防碰撞后卡的选择。参数 Select_Code 表示防碰撞的级数，应该与该函数执行前的 TX_Casc_Anticoll()中的 Select_Code 相同。在任意一个防碰撞函数成功执行后，或在任何时候当程序员想实际地与已知序列号的卡片进行通信时，必须使用 TX_Casc_Select 或后面得 TX_Select 函数，以建立与所选卡的通信。卡的序列号长度有三种：4 字节、7 字节和 10 字节。4 字节的只要用一级选择即可,如 Mifare1 S50 S70 等；7 字节的要用二级选择才能完成，如 UltraLight 和 DesFire 等。如果需要进行二级选择，第一级选择输入的序列号为，第一级防碰撞所得到的序列号,其中最低字节为级联标志 0x88，只后 3 字节可用，第二级选择输入得序列号为第二级防碰撞得到 4 字节序列号。在程序中可用*Sak 的 bit2 位来判断是否还有序列号未读出，如 if(*SAK & 0x04){…}。

此外，当模块成功选择一张卡片后，模块内部会记录下该卡片的的序列号，并供其它后续需要卡号的函数使用，因此后续的操作函数（如 TX_Auth）都不需再输入卡片的序列号。

数据块格式描述：

主机→TX522 命令模式）：

SEQNR： 0 （可自定义）

COMMAND：0x75

LENGTH： 5

DATA[0]： Select_Code

DATA[1]： 卡的序列号第 1 字节（最低字节）

DATA[2]： 卡的序列号第 2 字节

DATA[3]： 卡的序列号第 3 字节

DATA[4]： 卡的序列号第 4 字节

TX522→主机（响应模式）：

SEQNR： 0

STATUS： OK, QUIT, NO_TAG_ERR, CRC_ERR, PARITY_ERR, BIT_COUNT_ERR, COMM_ERR

LENGTH： 1

DATA[0]： Sak

4.3.16 证实 2—Auth2

函数原型：uchar TX_Auth2(uchar KeyAB, uchar Sector, uchar Key_Sector);

输入参数：KeyAB：密钥类型（1 字节）。可取值为 KeyAB＝0x00（KEYA），利用密钥 A 进行验证，或 KeyAB＝0x04（KEYB），利用密钥 B 进行验证密钥 B。

Sector：所要验证的卡扇区号（也即将要访问的卡的扇区号），取值范围 0～15，不能用于 S70 卡。所要验证的卡扇区号（也即将要访问的卡的扇区号），取值范围 0～39，能用于 S70 卡。

Key_Sector：模块内的密钥区号（1 字节）：取值范围 0～15。

输出参数： 无

函数返回：可能的状态值：OK，QUIT，NO_TAG_ERR，PARITY_ERR，BIT_COUNT_ERR，COMM_ERR。

功能描述：使用模块内部密钥区中 Key_Sector 中的密码对指定的卡的扇区 Sector 进行验证，若卡 Sector 区中的密码与存储在模块内 Key_Sector 中的密码相同，则验证成功，返回 OK。与 TX_Auth 函数不同的是，该函数参数中同时有 Sector 和 Key_Sector，模块中的密钥区和卡中的扇区可以是交叉的关系。并且该函数可指定卡的 Sector 和模块内部 Key_Sector，因此 Sector 的取值范围为 0～39，可用于 S70 卡的证实。

与 TX_Auth 一样，该函数也依赖与 TX_Load_Key 函数曾经成功执行过，因为模块内部密码区（Key_Sector）中的密码要由 TX_Load_Key 函数事先装载。该函数适用于对于所有卡来说密码相同的应用，密钥的装载可以在一个安全的场合一次性装入。

数据块格式描述：

主机→TX522 命令模式）：

SEQNR： 0 （可自定义）

COMMAND：0x72

LENGTH： 3

DATA[0]： KeyAB

DATA[1]： Sector

DATA[2]： Key_Sector

TX522→主机（响应模式）：

SEQNR： 0

STATUS： OK， QUIT， NO_TAG_ERR， PARITY_ERR， BIT_COUNT_ERR，COMM_ERR

LENGTH： 0

DATA[0]： 无

4.3.17 直接密码证实—Auth_Key

函数原型：uchar TX_Auth_Key (uchar KeyAB, uchar Sector, uchar idata *Key);

输入参数：KeyAB：密钥类型（1 字节）。可取值为 KeyAB＝0x00（KEYA），利用密钥 A 进行验证，或 KeyAB＝0x04（KEYB），利用密钥 B 进行验证密钥 B。

Sector：所要验证的卡扇区号（也即将要访问的卡的扇区号），取值范围 0～39，能用于 S70 卡。

Key：用于证实的密码首地址，,应在外部定义一个共 6 个字节的数组用于存放密码

输出参数： 无

函数返回：可能的状态值：OK，QUIT，NO_TAG_ERR，AUTH_ERR, PARITY_ERR，BIT_COUNT_ERR，COMM_ERR。

功能描述：TX_Auth 和 TX_Auth2 函数都必须要依赖于 TX_Load_Key()函数的曾经的成功执行，在进行证实之前，一定要确认正确的密钥已经存于模块的密钥区内。而 TX_Auth_Key 函数则将密钥存放于 Key 指针所指向的 6 个字节存储区内，TX_Auth_Key 函数执行时不对密钥区进行操作，因此也称为直接密码证实。若卡中的密钥与所传输的密码相匹配。则证实成功，函数将返回 OK。

直接密码证实一般用于对每一张卡来说密钥都不同的应用，如在使用安全模块（PSAM 卡）的消费应用中，消费机首先将卡的序列号读出，然后与 PSAM 卡中消费主密钥一起生成导出密钥，然后直接用导出密钥与卡的一个应用扇区相互证实。

数据块格式描述：

主机→TX522 命令模式）：

SEQNR： 0 （可自定义）

COMMAND：0x73

LENGTH： 8

DATA[0]： KeyAB

DATA[1]： Sector

DATA[2]： Key[0]

…

DATA[7]： Key[5]

TX522→主机（响应模式）：

SEQNR： 0

STATUS： OK， QUIT， NO_TAG_ERR， PARITY_ERR， BIT_COUNT_ERR，COMM_ERR

LENGTH： 0

DATA[0]： 无

4.3.18 读—Read

函数原型：uchar TX_Read(uchar Block,uchar idata *Data)

输入参数：Block：卡块号（1 字节）： S50：0～63；S70：0～255

输出参数：*Data：Data 为读回 16 字节数据的首地址。

函数返回：可能的状态值：OK，QUIT，NO_TAG_ERR，CRC_ERR，NOT_AUTH_ERR，PARITY_ERR，BIT_COUNT_ERR，COMM_ERR。

功能描述：在验证成功后，使用该函数读 Mifare 卡中相应块的数据。Mifare 卡中一个块的数据是 16 字节，因此读写一次均是 16 个字节。

所读块号必须与之前所验证的块号在同一个扇区内，mifare1 卡从块号 0 开始按顺序每 4 个块 1 个扇区。

因为 Mifare 卡每个扇区的密码可能不相同，若要对一张卡中的多个扇区进行操作，在对某一扇区操作完毕后，必须进行一条读命令才能对另一个扇区直接进行验证命令，否则必须从请求开始操作。

密码数据不能被读取。

数据块格式描述：

主机→TX522 命令模式）：
SEQNR： 0 （可自定义）
COMMAND：0x46
LENGTH： 1
DATA[0]： Block

TX522→主机（响应模式）：
SEQNR： 0
STATUS： OK， QUIT， NO_TAG_ERR， CRC_ERR， NOT_AUTH_ERR，PARITY_ERR，BIT_COUNT_ERR，COMM_ERR
LENGTH： 16
DATA[0]： 所访问块的第一个字节
：
DATA[15]： 所访问块的最后一个字节

4.3.19 写—Write

函数原型：uchar TX_Write(uchar Block,uchar idata *Data)
输入参数：Block：卡块号（1 字节）： S50：1～63；S70：1～255
*Data：16 字节数据指针，Data 为写入的 16 字节数据的首地址。
输出参数： 无
函数返回：可能的状态值：OK， CHK_WR_OK， CHK_WR_FAILED，CHK_WR_COMP_ERR， QUIT， NO_TAG_ERR， CRC_ERR， NOT_AUTH_ERR，PARITY_ERR，BIT_COUNT_ERR，COMM_ERR。
功能描述：对卡内某一块进行验证成功后，即可对同一扇区的各个块进行写操作（只要访问条件允许），其中包括位于扇区尾的密码块，这是更改密码的唯一方法。该函数在写入数据后会立即进行读操作并进行数据写入正确性判断。
Mifare 卡中一个块的数据是 16 字节，因此读写一次均是 16 个字节。所读块号必须与之前所验证的块号在同一个扇区内，mifare1 卡从块号 0 开始按顺序每 4 个块 1 个扇区。
数据块格式描述：
主机→TX522 命令模式）：
SEQNR： 0 （可自定义）
COMMAND：0x47
LENGTH： 17
DATA[0]： Block
DATA[1]： 所要写的第一个字节
：
DATA[16]： 所要写的最后一个字节

TX522→主机（响应模式）：
SEQNR： 0
STATUS： OK， QUIT， NO_TAG_ERR， NOT_AUTH_ERR， WRITE_ERR，BIT_COUNT_ERR，COMM_ERR
LENGTH： 0

DATA[0]： 无

补充说明：

注意：在将数据写到卡片上的某一扇区时，一定要小心。因为有些 block 中存储了密码数据以及存储允许使能数据。特别是每一个扇区的 Block3 中存放了该扇区的存取条件，包含有 KEYA，KEYB 及该扇区的控制字。Mifare 1 卡片出厂时的 Block3 有缺省值，为：“a0a1a2a3a4a5ff078069b0b1b2b3b4b5”,共 16 个 Bytes。

涉及 Mifare 1 卡片的存储结构等信息，请参考 Mifare 1 卡片数据手册。

程序员在使用 Mifare 1 卡片做应用时，一定要清清楚楚记住每一个扇区的 Block3 的数据，这样也就记住了扇区的密码和存取控制字。否则，扇区的存储空间将不执行 Read/Write 等操作而失效。

任何人试图用任何方式来读写不知密码的卡片或某一扇区都是徒劳无益的。

卡片应放在安全的地方，即不要放在离模块天线较近的地方。因为当模块对其它卡片执行某些指令时，有可能无意间对这一卡片进行了读/写等操作，从而操作卡片的失效。

4.3.20 带内部自动传送的值操作 1—Value

函数原型： uchar TX_Value(uchar Mode, uchar Block, uchar idata *Value, uchar Trans_Block);

输入参数：Mode：0xC0—减； 0xC1—加； 0xC2—恢复

Block：卡内块地址，对该块进行值操作，取值范围：S50：1～63；S70：1～255

*Value：4 字节数据指针，用来存储减少值或增加值，当进行恢复操作时，该值为空值。Value 是减少值或增加值存放的首地址，存放时，低地址存放高字节。

Trans_Block：传输块地址，取值范围：S50：1～63；S70：1～255。

输出参数： 无

函数返回：可能的状态值：OK, QUIT, NO_TAG_ERR, BIT_COUNT_ERR, TRANS_ERR, CODE_ERR, COMM_RERR。

功能描述：此函数对卡内的某一块进行加、减或数据备份，该块必须为值块格式，并支持自动传送。该函数其实是 TX_Increment、TX_Decrement 或 TX_Restore 函数与 TX_Transfer 函数的组合。因此可以用该函数替换上述函数。若卡块号与传输块号相同，则将操作后的结果写入原来的块内；若卡块号与传输块号不相同，则将操作后的结果写入传输块内，结果传输块内的数据被覆盖，原块内的值不变。当模式为“恢复”时，“值”无意义。

数据块格式描述：

主机→TX522 命令模式）：

SEQNR： 0 （可自定义）

COMMAND：0x70

LENGTH： 7

DATA[0]： Mode

DATA[1]： Block

DATA[2]： Value(LL)

DATA[3]： Value(LH)

DATA[4]： Value(HL)

DATA[5]： Value(HH)

DATA[6]： Trans_Block

TX522→主机（响应模式）：
SEQNR：　0
STATUS：　OK，　QUIT，　NO_TAG_ERR，　BIT_COUNT_ERR，　TRANS_ERR，CODE_ERR, COMM_RERR
LENGTH：　0
DATA[0]：　无

4.3.21 置位控制位—Set_Control_Bit

函数原型：uchar TX_Set_Control_Bit(void);
输入参数：　无
输出参数：　无
函数返回：可能的状态值：OK，QUIT，COMM_ERR。
功能描述：此函数设置 TX522A 模块的控制位 Ctrl（J2_1 脚）为高电平。
数据块格式描述：
主机→TX522 命令模式）：
SEQNR：　0 （可自定义）
COMMAND：0x50
LENGTH：　0
DATA[0]：　无

TX522→主机（响应模式）：
SEQNR：　0
STATUS：　OK，QUIT，COMM_ERR
LENGTH：　0
DATA[0]：　无

4.3.22 清除控制位—Clr_Control_Bit

函数原型：TX_Clr_Control_Bit(void);
输入参数：　无
输出参数：　无
函数返回：可能的状态值：OK，QUIT，COMM_ERR。
功能描述：此函数设置 TX522A 模块的控制位 Ctrl（J2_1 脚）为低电平。

数据块格式描述：
主机→TX522 命令模式）：
SEQNR：　0 （可自定义）
COMMAND：0x51
LENGTH：　0
DATA[0]：　无
TX522→主机（响应模式）：
SEQNR：　0
STATUS：　OK，QUIT，COMM_ERR

LENGTH： 0
DATA[0]： 无

4.3.23 输出蜂鸣器信号—Buzzer

函数原型：TX_Buzzer(uchar Frquence, uchar Time);
输入参数：Frequence: 输出方波频率，取值（0～255），对应频率（0.73～4K），0 为直流，198 对应 2K。
Time: 方波输出持续时间，取值（0～255），10ms 的分辨率
输出参数： 无
函数返回：可能的状态值：OK， COMM_ERR。
功能描述：此函数输出一方波用于驱动交流蜂鸣器或低电平驱动直流蜂鸣器，驱动的频率、持续时间可设定。下面是频率对照表。

表 16 设定值与输出频率对照表

Frequence	输出频率（KHz）	Frequence	输出频率（KHz）
0	输出低电平（直流）	140	1.333
1	0.735	160	1.515
20	0.787	180	1.74
40	0.847	198	2.00
60	0.913	200	2.04
80	0.990	220	2.50
100	1.081	240	3.17
120	1.198	255	4.00

数据块格式描述：
主机→TX522 命令模式）：
SEQNR： 0 （可自定义）
COMMAND：0x60
LENGTH： 2
DATA[0]： Frequence
DATA[1]: Time

TX522→主机（响应模式）：
SEQNR： 0
STATUS： OK， COMM_ERR
LENGTH： 0
DATA[0]： 无

4.4 利用 spi_init()初始化 SPI 接口

函数原型：void SPI_Init(void);
输入参数：无
输出参数：无
函数返回：无
功能描述：此函数将 SPI 接口初始化成空闲状态，接 nCS 线的外部中断 1 为下降沿触发中断，且将定时器 1 配置成 SPI 接口的看门狗定时器。

4.5 SPI 看门狗定时器

可将定时器0或1作为SPI接口看门狗定时器，该定时器被设置成50ms溢出。
发送时开定时器中断，若中断之前通信未能完成（TX522A在DATA线上未返回响应信号），而造成该定时器产生中断，则取消本次传输，发送子程序返回COMM_ERR。
接收时关中断，用软件判断溢出次数，若在500ms内未收到TX522A返回的数据（nCS线上未产生下降沿），则退出本次命令的执行，命令返回COMM_ERR。

5. Mifare 卡工作状态介绍

Mifare 卡工作状态如表 17 所列。

表 17 Mifare 卡工作状态列表

状　态	描　　述
POWER OFF （断电）	卡片不位于读卡器有效区域，由于缺少射频磁场能量而处于断电状态，卡片不工作。
IDLE （空闲）	卡片进入读卡起有效区域内，被电磁场能量激活，延迟数毫秒后将进入 IDLE 状态。在该状态下，卡片能够解调读卡器传来的调制信号，并能对读卡器的 Request（以 IDLE 或 ALL 方式）命令进行应答，应答后返回卡片的类型。
READY （就绪）	卡片对读卡器的 Request 命令进行应答后，就进入了 READY 状态。在该状态中，可以采用比特帧防冲突算法。当卡片的唯一序列号被读卡器发来的 Selection 命令选中时，就退出本状态。
ACTIVE （激活）	当卡片的唯一序列号被读卡器选中时就进入 ACTIVE 状态。在该状态中，卡片完成本次应用所要求的全部操作。
HALT （停止）	卡片应用完成后，读卡器可通过发送 Halt 命令，使卡片进入 HALT 状态。在该状态中，卡片只对读卡器以 ALL 方式发送的 Request 命令进行应答（或被唤醒），从而又进入 READY 状态。

图 3 为 Mifare 卡的状态转换图。

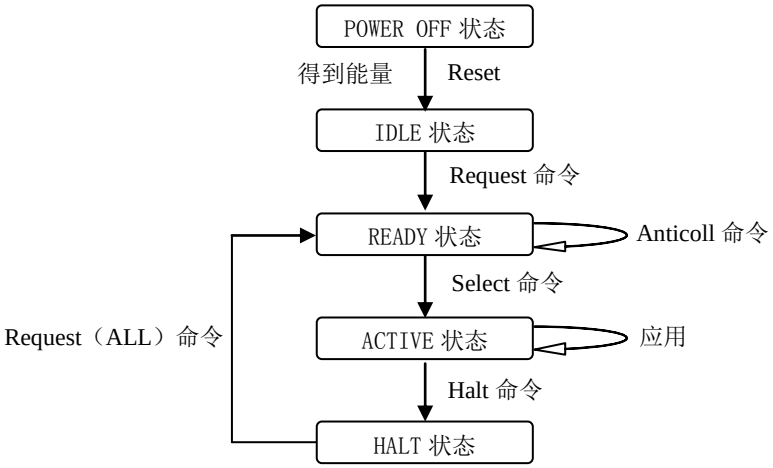

图 3 Mifare 卡状态转换示意图

6. 修订历史

版本	日期	原因
V1.00	2006/06/08	创建文档。
V1.10	2008/01/05	为所有函数增加 TX_前缀
V1.20	2008/03/15	1.将通讯协议描述得更详细； 2.增加自动寻卡函数，Auto_Detect 和 Read_ADDT； 3. 修改应用电路图，控制蜂鸣器的管脚串联电阻； 4.增加了自动寻卡中断指示管脚 INT_OUT; 5.删除了 checkwrite 函数，直接在 write 函数中检查 6 .增加了技术参数、直流特性、封装等说明。
V1.21	2008.09.09	1. 增加了只读卡号的简单应用说明。 2. 去掉自动寻卡命令中中断关闭的选择，始终输出中断信号。 3.增加了 Get_CardSnr、Write_Auth、read_auth、value_auth、writevalue_auth、readvalue_auth 命令。 4.根据应用先后顺序调整了函数描述的先后。 5.改正 request 命令补充说明处的错误。 6.对文档重新编号。
V1.22	2009.02.05	1.删减了 mifare 卡介绍内容。 2. .增加了上电时间说明。
V1.23	2009.08.11	1.修改了图 2，增加了续流二极管。
V1.25	2012.03.15	1.从 TX500AT 说明书修改到 TX522AT

7. 销售信息

东莞市同欣智能科技有限公司
地　　址：广东省东莞市石碣镇沙腰管理区林屋洲
邮　　编：523292
销售电话：0769-86019851-168; 13652608930 QQ:872089468
技术支持：0769-86019851-138; 0769-86019853; 18666865339　QQ：14754020
传　　真：0769-86019852
网　　址：http:// www.TXRFID.com
E-mail：　sales@TXRFID.com　support@TXRFID.com