TRF7970A 在近场通信中的应用

Michael Qian（钱 堃） SZ EP FAE

摘要

本文章主要介绍 TI TRF7970A 产品在 RFID/NFC(RADIO FREQUENCY IDENTIFICATION/ NEAR FIELD COMMUNICATION)应用上的设计要点与代码实现，TRF7970A 芯片硬件支持所有 RFID 协议以及 NFC 功能，配套前端 MCU 的驱动软件和协议栈实现卡片读写和点对点数据通信。本文详细论述了 NFC 和 RFID 基本知识，如何利用 MSP430 驱动 TRF7970A 完成卡片的读写和点对点通信，并对 RFID/NFC 协议和 TRF7970A 驱动代码的实现做了详细的论述。该文章不仅可以有助工程师快速上手 NFC 领域产品研发，并可配合 TI MSP430 系列超低功耗 MCU(MICRO CONTROLLER UNIT)快速设计出高性价比低功耗手持 RFID/NFC 产品。

关键词：TRF7970A，RFID/NFC，卡片读写，低功耗

目录

1. NFC 介绍

1.1. NFC 技术介绍

NFC 技术是由 Philips、Nokia 和 Sony 主推的一种近距离无线通信技术（NFCIP-1），并向 ECMA 国际组织提交标准草案。这项开放技术规格 NFCIP-1 被认可为 ECMA-340 标准，并由 ECMA 向 ISO/IEC 提交标准，现在该技术被批准纳入 ISO/IEC18092。2003 年 NFCIP-1 被 ETSI 批准为 TS 102 190 v1.1.1。为了兼容非接触式智能卡(传统 RFID)，2004 年 NFC 论坛又推出了 NFCIP-2 规范，并被相关组织批准为 ECMA-352、ISO/IEC 21481 和 ETSI TS 102 312 V1.1.1。

NFCIP-1 标准详细规定 NFC 设备的调制方案、编码、传输速度与 RF 接口的帧格式，以及主动与被动 NFC 模式初始化过程中，数据冲突控制所需的初始化方案和条件。此外，这些标准还定义了传输协议，其中包括协议启动和数据交换方法等。

NFCIP-2 则指定了一种灵活的网关系统，用来检测和选择三种操作模式之一：NFC 卡模拟模式、读写器模式和点对点通信模式。选择既定模式以后，按照所选的模式进行后续动作。网关标准还具体规定了 RF 接口测试方法（ISO/IEC 22536 和 ECMA-356）和协议测试方法（ISO/IEC 23917 和 ECMA-362）。这意味着符合 NFCIP-2 规范的产品将可以用作 ISO/IEC 14443 A 和 B 以及 Felica（Proximity Cards）和 ISO 15693（Vicinity Cards）的读写器。

NFC 将非接触读卡器、非接触卡和点对点（Peer-to-Peer）功能整合进一块单芯片，为消费者的生活方式开创了不计其数的全新机遇。这是一个开放接口平台，可以对无线网络进行快速、主动设置，也是虚拟连接器，服务于现有蜂窝状网络、蓝牙和无线 802.11 设备。

与 RFID 一样，NFC 信息也是通过频谱中无线频率部分的电磁感应耦合方式传递，但两者之间还是存在很大的区别。首先，NFC 是一种提供轻松、安全、迅速的通信的无线连接技术，其传输范围比 RFID 小，RFID 的传输范围可以达到几米、甚至几十米，但由于 NFC 采取了独特的信号衰减技术，相对于 RFID 来说 NFC 具有距离近、带宽高、能耗低等特点。 其次，NFC 与现有非接触智能卡技术兼容，目前已经成为得到越来越多主要厂商支持的正式标准。再次，NFC 还是一种近距离连接协议，提供各种设备间轻松、安全、迅速而自动的通信。与无线世界中的其他连接方式相比，NFC 是一种近距离的私密通信方式。最后，RFID 更多的被应用在生产、物流、跟踪、资产管理上，而 NFC 则在门禁、公交、手机支付等领域内发挥着巨大的作用。

1.2 NFC 操作模式

NFC 可以运行在 ISO/IEC 18092, NFC IP-2, 和 ISO/IEC 14443 三种无线智能卡标准下（contactless smart card standard），其功能框架如图 1-1 所示。

1、读/写 (R/W, PCD)

在这种模式，开启 NFC 功能的手机可以读写任何支持的标签， 读取其中的 NFC 数据格式标准的数据。

2、点对点 (P2P)

在这种模式下，两个 NFC 设备可以交换数据。 例如，你可以分享启动蓝牙或 Wi-Fi 连接的参数来启动 蓝牙或 Wi-Fi 连接。你可以交换如虚拟名片或数字相片等数据。点对点模式符合 ISO/IEC 18092 标准。

3、卡片模拟 (CARD EMULATION, PICC)

NFC 设备在与标签交互时可以扮演读取器的角色。在这种模式 NFC 设备也可做为标签或被读取的无线卡片，被别人读取或者存储数据。

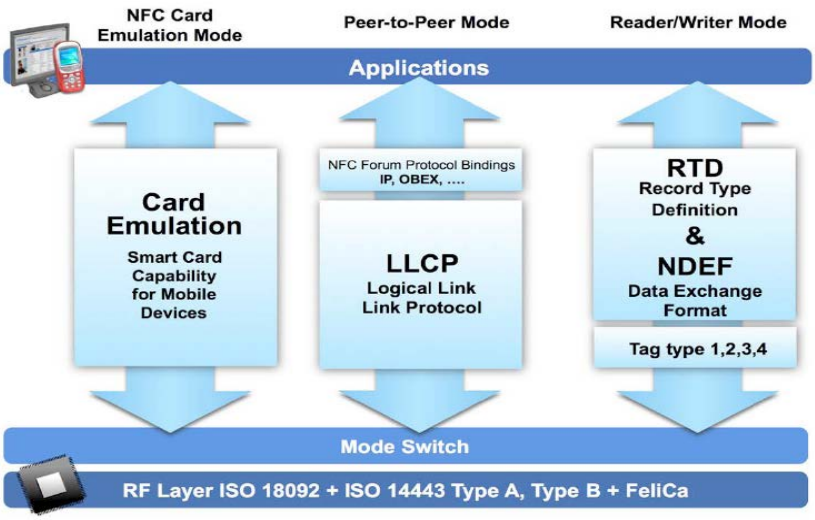

NFC Card Emulation Mode
Peer-to-Peer Mode
Reader/Writer Mode
Applications
Card Emulation
Smart Card Capability for Mobile Devices
NFC Forum Protocol Bindings IP, OBEX, ...
LLCP
Logical Link Link Protocol
RTD
Record Type Definition
&
NDEF
Data Exchange Format
Tag type 1,2,3,4
Mode Switch
RF Layer ISO 18092 + ISO 14443 Type A, Type B + FeliCa

图 1-1　　NFC 功能框架

2. TRF7970A 芯片的硬件平台设计:

2.1 TRF7970A 芯片介绍

TRF7970A 是一款用于 13.56MHz RFID/近场通信系统的集成模拟前端和数据组帧器件。 内置编程选项使得此器件适合于广泛的相邻或者附近识别系统的应用。它能够执行以下三种模式中的任一模式: RFID/NFC 读取器、NFC 对等点、或者卡仿真模式。 内置用户可配置编程选项使得此器件适合于范围宽广的应用。 通过在控制寄存器中选择所需的协议可对 TRF7970A。 到所有控制寄存器的直接存取可根据需要对不同的读取器参数进行微调。该芯片支持近场通信 (NFC) 标准 NFCIP-1 (ISO/IEC 18092) 和 NFCIP-2 (ISO/IEC 21481)，针对 ISO15693，ISO18000-3，ISO14443A/B，和 Felica 的完全集成协议处理，针对所有三位速率 (106kbps，212kbps，424kbps) 和卡仿真的集成编码器、解码器、和数据组帧 NFC 发送方、有源和无源目标方操作操作方式，针对 NFC 无源应答机仿真操作的带有可编程唤醒电平的 RF 场检测器，针对 NFC 物理层冲突避免的 RF 场。输入电压范围：2.7VDC 至 5.5VDC，内置 LDO 可输出 3.3V 电压给 MCU 供电。

2.2 TRF7970A 射频模块解析

TRF7970A 由一个集成的模拟前端 (AFE)和一个针对 ISO15693，ISO 14443A，ISO14443B，和 FeliCa 的内置数据组帧引擎组成。 这包括针对 ISO14443 的高达 848kbps 的数据速率，包括板上全部组帧和同步任务（在默认模式下）。 TRF7970A 也支持 NFC 标签类型 1，2，3，和 4 操作，后续章节会详细描述。 这个架构使得用户能够建立一个完整且划算而又高性能的多协议 13.56MHz RFID/NFC /NFC 系统和一个低成本微控制器，例如，MSP430。通过使用器件提供的直接模式中的两个，可执行其它标准，甚至定制的协议。 这些直接模式（0 和 1）使得用户能够完全控制模拟前端 (AFE) 并获得到原始副载波数据或者非成帧数据（但已经是 ISO 格式数据）和相关（被提取的）时钟信号的存取权限。接收器系统有一个双输入接收器架构。 此接收器还包括多种自动和手动增益控制选项。 接收到的输入带宽可被选择来包含广泛范围的输入副载波信号选项。通过 RSSI 寄存器可获得接收到的来自应答机、周围信号源或者内部电平的信号强度。 接收器输出可在一个数字化副载波信号和任一集成型副载波解码器间进行选择。所选择的副载波解码器将数据比特流和数据时钟作为输出发送 TRF7970A 还包括一个接收器组帧引擎。这个接收器组帧引擎执行 CRC 或者奇偶校验，移除 EOF 和 SOF 设置，并且将数据组织成用于 ISO14443-A/B，ISO15693，和 FeliCa 协议的字节格式。 然后保存在一个 128 字节 FIFO 寄存器，微控制器 (MCU) 可访问该帧的数据。

2.2.1 发射器 – 模拟部分

TRF7970A 需要外挂 13.56M hz 晶振，13.56MHz 振荡器为 PA 级生成 RF 信号。 此功率放大器包含一个可选额定输出电阻为 4Ω 或者 8Ω 的驱动器。发射功率级由芯片状态控制寄存器 (0x00) 的 B4 位控制。 当设置为 5V 自动运行时，发射功率级可在 100mW（半功率）或者 200mW（全功率）之间进行选择。当设置为 3V 自动运行时，发射功率级可在 33mW（半功率）或者 70mW（全功率）之间进行选择。ASK 调制深度由调制器中的 B0，B1，和 B2 位以及 SYS_CLK 控制寄存器 (0x09) 控制。 ASK 调制深度可在 7% 至 30% 或者 100% 之间调节 (OOK)。有可能通过将 ISO 控制寄存器 (0x01) 设置为直接命令模式而实现对发射调制深度的外部控制。 当在直接命令模式下操作 TRF7970A 时，可通过在引脚 12 上选择调制类型 ASK 或者 OOK 来完成发射调制。 只有将调制器和 SYS_CLK 控制寄存器(0x09)的 B6 设置为 1，才能对调制类型进行外部控制。在正常运行模式下，调制脉冲的长度由 ISO 控制寄存器 (0x01) 所选的协议定义。 通过使用一个高 Q 天线，调制脉冲通常被拉长，并且标签侦测一个比目标脉冲更长的脉冲。对于这些情况，需要使用 TX 脉冲长度寄存器 (0x05) 对调制脉冲长度进行修正。如果此寄存器为全 0，则脉冲长度由协议选择管理。 如果寄存器的值不是 0x00，则脉冲长度等于寄存器的值乘以 73.7ns；因此，脉冲长度可在 73.7ns 和 18.8ns（增量 73.7ns）间进行调节。

2.2.2 发射器 – 数字部分

发射器的数字部分是一个接收器的镜像。 控制 ISO 控制寄存器 (0x01) 的设置正如接收器一样被应用于发射器。 在 TRF7970A 的缺省模式下，TRF7970A 自动添加这些特别信号：通信开始，通信终止，SOF，EOF，奇偶校验位和 CRC 字节。然后此数据被编码为调制脉冲电平并被发送到 RF 输出级调制控制单元。就像和接收器一起工作，这意味着此外部系统 MCU 只需载入含数据的 FIFO，所有的微编码自动完成，这又为固件开发人员节省了编码空间和编码时间。 此外，当一个全新选择进入 ISO 控制寄存器 (0x01) 时，所有用于发射参数控制的寄存器自动预置到最佳值。

2.2.3 数据发射操作启动方法

载入将要发送的字节数到寄存器 0x1D 和 0x1E 并将即将发送的数据载入到 FIFO（地址 0x1F），随后发送一个发射命令（参见直接命令部分）。 当接收到发射命令后，传送开始。发送发射命令和最先发射的字节数，然后开始发送数据到 FIFO。 当第一个数据字节被写入 FIFO 时，传送开始。如果数据长度大于 FIFO 的容量，当来自 FIFO 的大部分数据已经被发射时，TRF7970A 通过发送一个含有标志（此标志位于 IRQ 寄存器内用于标示 FIFO 的低或高状态）的中断请求来通知外部系统 MCU。 外部系统应该通过向 FIFO 加载下一个数据包来做出响应。在发射操作的末尾，一个含有标志（位于 IRQ 寄存器 (0x0C) 内用于标示 TX 已经完成（示例值 = 0x80）的中断请求 (IRQ) 通知外部系统 MCU。TX 长度寄存器还支持不完整字节传送。 寄存器 0x1D 中的 2 个高半字节和寄存器 0x1E 中包含位 B4 到 B7 的半字节存储已经被发射的完整字节的数量。 寄存器 0x1E 中的位 B0 是一个标志，用于标示还有额外的位将被发射，这些位并未形成一个完整的字节。位的数量存储在同一寄存器 (0x1E) 的 B1 到 B3。一些协议有选项，并且有 2 个次级配置寄存器可用来选择 TX 协议选项。

ISO14443B TX 选项寄存器 (0x02)。 这个寄存器控制 SOF 和 EOF 选择和 ISO14443B 协议的 EGT 选择。

ISO14443A 高比特率选项和奇偶校验寄存器 (0x03)。 这个寄存器使得在 ISO14443 高比特率协议中为

RX 和 TX 启用不同的比特率并且还可选择 ISO14443 A 高比特率协议的奇偶校验方法。数字部分也有一个定时器。 此定时器可在与一个所选事件一致的规定时间开始发射操作。

2.3 TRF7970A 硬件电路设计

2.3.1 MCU 选型：

如图 2-1 所示，参考设计中选择的 MCU 为 MSP430F2370（32kB 闪存，2k SRAM），MCU 程序含 ISO15693，ISO14443，ISO18092,Flica 等多种协议以及上位机 GUI 通讯协议。但最小 MCU 需求取决于应用要求和编码。 假设只需支持一个 ISO 协议或者一个协议的有限命令集，则对于 MCU 闪存和 RAM 的要求将会大大减少。 请注意递归目录和防冲突命令比单槽运行要求更多的 RAM。 例如，ISO15693（含主机接口）目前的基准固件大约为 8kB，使用 512B RAM，推荐使用 MSP430G2xxx 系列 MCU。对于所有支持的协议（具有同样的主机接口），此基准固件接近 12kB 并且最少使用 1kB 的 RAM。 如果要支持所有 NFC 协议栈已经读写指令，推荐使用 MSP430F5529(128K 闪存，8K SRAM)。

2.3.2 硬件设计:

1.接口设计:

TRF7970A 与 MCU 连接方式可以采用并口模式或者 SPI 口模式，给客户更加灵活的通讯选择方式，较短的通讯线路，走线需要对无线电设备频率线路的正确隔离，和一个恰当的接地区域以避免信号串扰。SPI 模式下 DATA_CLK 线路上的推荐时钟频率为 2MHz。

2.电源设计：

TRF7970A 正电源输入 VIN（引脚 2）为 3 个内部稳压器提供输出电压 VDD_RF 功率放大器调节器，VDD_A 模拟电源稳压器，和 VDD_X 数字电源稳压器。 所有稳压器使用外部旁路电容器用于电源噪音过滤并必须如参考电路原理图中指示的那样进行连接。 这些稳压器提供一个 RFID 读取器系统所需的高电源抑制比 (PSRR)。这些稳压器并不是独立的并且在寄存器 0x0B 中有公共控制位用于输出电压设置。 这些稳压器可被配置运行在自动或者手动模式下（寄存器 0x0B，位 7）。 自动稳压器设置模式确保 PSRR 和 RF 输出的最高可能电源电压之间达到最优折中方案（为了保证最大 RF 功率输出）。 手动模式允许用户手动配置此稳压器的设置。

利用 MCU 选择读取器的输入电源电压模式。 这一操作由在芯片状态控制寄存器 0x00 中完成。 寄存器 0x00 的位 0 在 5V 或者 3V 输入电源电压中进行选择。 默认配置为 5V，反映出运行电源电压范围为 4.3V 至 5.5V。如果电源电压低于 4.3V，则应使用 3V 配置。多种稳压器可被配置为运行在自动或者手动模式下。 这一操作由寄存器 0x0B 配置 完成。

利用内部电源管理模块输出 3.3V 电压，完全可以满足驱动 MCU 需求。具体参考电路见图 2-1 所示。

3.时钟设计:

TRF7970A 外围电路外挂 13.56M hz 或者 27.12MHZ 振荡器作为内部震荡频率，并可以通过 MCU 设置分频倍数并输出。通过控制芯片状态控制寄存器 0x00)和 EN，EN2 端子来控制 13.56MHz 或者 27.12MHz 晶体（或者振荡器）。 振荡器为 RF 输出级生成 RF 频率以及用于数字部分的时钟源。 引脚 27 (SYS_CLK)上为任何其它的外部电路提供缓冲时钟信号。 调制和 SYS_CLK 寄存器 (0x09) 内的 B4 和 B5 用于将引脚 27 上的外部 SYS_CLK 信号分成 1 份，2 份，或者 4 份，该时钟作为时钟输出作为 MCU 时钟源，亦或可以使用 MSP430 内部 DCO 校准时钟作为 MCU 时钟源。

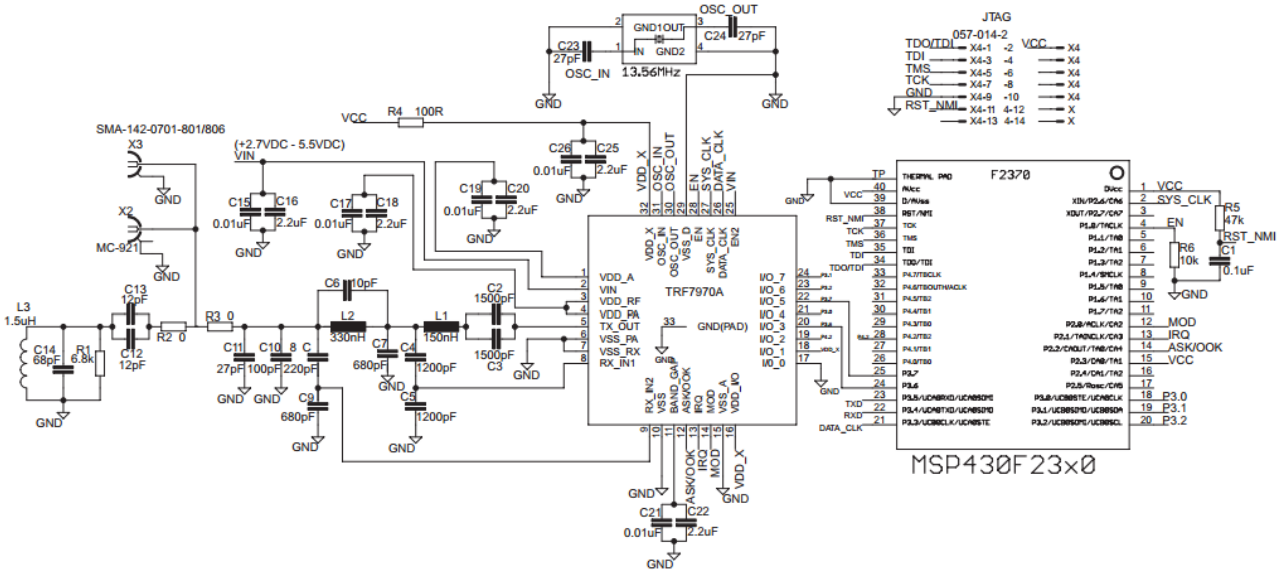


图 2-1 TRF7970A 硬件参考电路

2.4 TRF7970A NFC 芯片工作操作流程

由图 2-2 所示，AFE 即为 TRF7970A 芯片， MCU 完成 TRF7970A 的配置，控制 ISO 协议栈的运行，以及发送数据。以 TI 官网例程(下载地址:http://www.ti.com/product/trf7970a)，讲解操作流程：

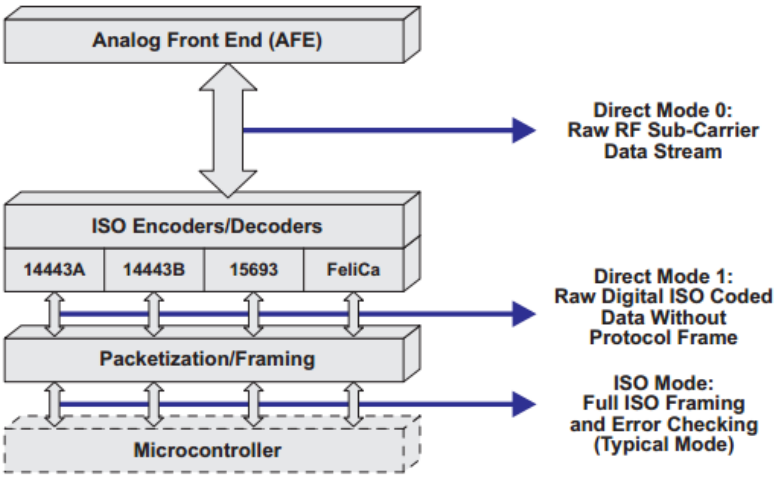


图 2-2 NFC 工作流程

1. 系统上电，初始化 MCU，初始化 SPI，使能 TRF7970A。
2. 写 command (0x03)初始化 TRF7970A。
3. 写 command (0x00) 设置 TRF7970A 处于空闲状态。

4. 对(0x09)寄存器进行配置，配置 MCU 分频 6.78MHz 时钟输出, OOK 100%模式。

5. 配置 MCU 时钟为外部 6.78M HZ 时钟，重新初始化 UART SPI。

6. 设置 TFR7970A 状态寄存器(0x00)，打开射频收发模块，并设置全功率发射模式。

7. 设置协议模式，开启透传,标准 RFID 或者 NFC 模式，如选择 NFC 模式，选择 NFC 处于主动模式或者被动模式。

8. 按照不同协议标准发送数据到 FIFO，然后由 TRF7970A 向空中传输数据。

9. 发送数据结束，IRQ 产生中断，读取(0x0C) IQR 状态寄存器，判定是何种中断。

10. IRQ 中断读取，判定是否为 RX 数据接收中断，如果是，接收 FIFO 内数据暂存与 buffer。

11. 重复步骤 6-11。

该步骤讲述 MCU 控制 TRF7970A 完成数据发送和接收的流程，具体卡片读写以及点对点数据传输需要 ISO 协议栈配合，后续章节会详细结合协议栈去讲解。

3. 利用 TRF7970A 对卡片的操作

3.1 标签种类概述

高频电子标签主要工作频率为 13.56M HZ，主要用于身份识别，图书馆管理，生产线控制，资产管理等应用，其工作温度范围在-25C 到+70C，内存为 EEPROM，可多次进行读写，工作距离在 10CM 左右，视工作环境和配套天线存在差异，读头结构如图 3-1 所示。

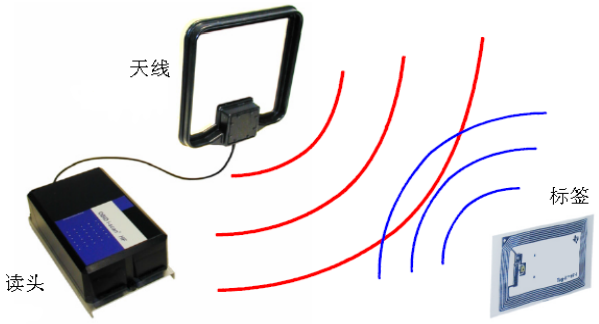


图 3-1 RFID 组成

在 NFC 论坛的技术规范网站 http://www.nfc-forum.org/specs/spec_list/查看 NFC 论坛标签种类。标签的规范定义了实现 阅读器/擦写器的技术信息和相应的与之互动的 NFC 设备的控制功能。

第 1 类标签（Tag 1 Type）：此类型基于 ISO14443A 标准。此类标签具有可读、重新写入的能力，用户可将其配置为只读。存储能力为 96 字节，用来存网址 URL 或其他小量数据富富有余。然而，内存可被扩充到 2k 字节。此类 NFC 标签的通信速度为 106 kbit/s。此类标签简洁，故成本效益较好，适用于许多 NFC 应用。

第 2 类标签（Tag 2 Type）：此类标签也是基于 ISO14443A，具有可读、重新写入的能力，用户可将其配置为只读。其基本内存大小为 48 字节，但可被扩充到 2k 字节。通信速度也是 106 kbit/s。

第 3 类标签（Tag 3 Type）：此类标签基于 Sony FeliCa 体系。目前具有 2k 字节内存容量，数据通讯速度为 212 kbit/s。故此类标签较为适合较复杂的应用，尽管成本较高。

第 4 类标签（Tag 4 Type）：此类标签被定义为与 ISO14443A、B 标准兼容。制造时被预先设定为可读/可重写、或者只读。内存容量可达 32k 字节，通信速度介于 106 kbit/s 和 424 kbit/s 之间。

15693 标签（Tag F Tpye）：此类标签也是基于 ISO15693，具有可读、重新写入的能力，用户可将其配置为只读。其基本内存大小为 256 字节，但可被扩充到 2k 字节。通信速度也是 106 kbit/s。

3.2 NDEF 格式

3.2.1 NDEF 介绍：

NDEF(全称 NFC data exchange format)使 NFC 的各种功能更加适用各种标签类型进行数据传输，它封装了 NFC 标签的种类细节信息，使得应用不用关心是在与何种标签通信。为实现标签和 NFC 设备，以及 NFC 设备之间的交互通信，NFC 论坛定义了称为 NFC 数据交换格式(NDEF)的通用数据格式，NDEF 是轻量级的紧凑的二进制格式，可带有 URL，vCard 和 NFC 定义的各种数据类型。

3.2.2 NDEF 定义：

NDEF(NFC Data exchange format)交换的信息由一系列记录(Record)组成。每条记录包含一个有效载荷，记录内容可以是 URL、MIME 媒质或者 NFC 自定义的数据类型。使用 NFC 定义的数据类型，载荷内容必须被定义在一个 NFC 记录类型定义(RTD)文档中，

记录中的数据类型和大小由记录载荷的头部(Header)注明。 这里的头部包含：

1、类型域。用来指定载荷的类型。

2、载荷的长度数。单位是字节(octet)。

3、可选的指定载荷是否带有一个 NDEF 记录。

NFC 定义的数据类型需要载荷内容被定义在 RTD 文档中，其结构见图 3-2 所示。

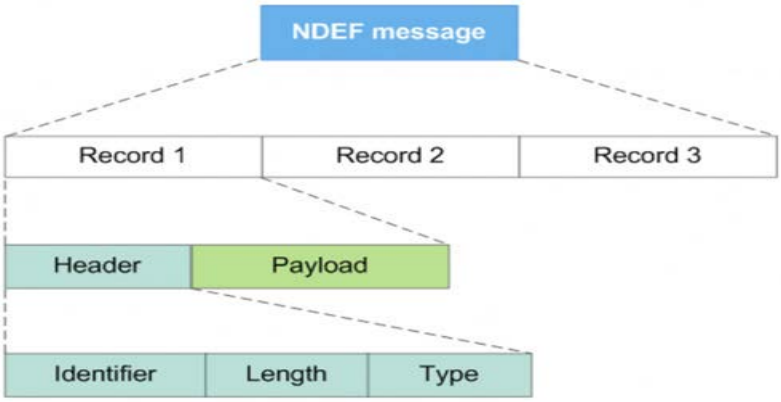


图 3-2 NDEF 格式标准

3.2.3 NDEF 数据在卡片内的存储

EEPROM 本身只是只是作为存储介质，NDEF 是在之上的数据存储格式。NDEF 数据存储从 EEPROM 第一个块开始起。

第一个 Block 存储 NDEF 数据定义了和卡片相关的信息（Capability Container Byte Definitions），占用 4 个字节数，存储信息为 NDEF 消息序列号，读写条件，片上存储空间大小，IC 是否支持多块读写操作。

第二个 Block 开始存储 NDEF 数据结构，见表 3-1 所示：第一个字节定义 TLV 结构类型：

表 3-1　TLV 结构标准

TLV Byte Name	Value	Description
Null TLV	0x00	Pad
Lock Control TLV	0x01	
Memory Control TLV	0x02	
NDEF Message TLV	0x03	Transponder contains NDEF Message
Proprietary TLV	0xFD	
Terminator TLV	0xFE	End of Message

以一个例子说明，NDEF 数据为空时：Block 内的数据为如表 3-2 所示。

表 3-2　空数据时 EEPROM 中 NDEF 数据

	CC0	CC1	CC2	CC3
Block 0	0xE1	0x40	0x20	0x00
	NDEF Message Present (Magic Number)	Version #, Read/Write Access Conditions	User Memory = CC2 value x 8 = 32 bytes	Does not support Read Multiple Blocks
Block 1	0x03	0x00	0xFE	0x00
	NDEF Message Present	Empty	TLV Terminator	Start of Available Memory

当 NDEF 数据非零时，其数据组织方式又有些不同，见表 3-3 所示:

需要定义记录类型，NDEF 数据长度等信息，详细 NDEF 定义见 NFCForum-TS-NDEF_1.0。

表 3-3　有数据时 EEPROM 中 NDEF 数据

	CC0	CC1	CC2	CC3
Block 0	0xE1	0x40	0x04	0x00
	NDEF Message Present (Magic Number)	Version #, Read/Write Access Conditions	User Memory = CC2 value x 8 = 32 bytes	Does not support Read Multiple Blocks
Block 1	0x03	0x13	0xD1	0x01
	NDEF Message Present	Length (19 bytes)	Record Header	Type Length
Block 2	0x0B	0x55	0x01	0x67
	Payload Length	Record Type U (URI)	URI Header Identifier	g
Block 3	0x6F	0x6F	0x67	0x6C
	o	o	g	l
Block 4	0x65	0x2E	0x63	0x6F
	e	.	c	o
Block 5	0x6D	0xFE	0x00	0x00
	m	TLV Terminator	Empty (Don't Care)	Empty (Don't Care)

假设我们需要按 NDEF 标准写 URL 链接 http://www.google.com 到 15693 卡片上，我们必须如下配置 NDEF 数据格式。

0x03 NDEF消息起始位，0x0f数据长度为19位，0xD1记录类型为普通文本文档，0x01载荷的长度数，0x0B有效数据长度包括积累类型和消息数据，0x55记录类型为URI链接，0x01 表示 http://www，0x67,0x6F,0x6F，0x67，0x6C，0x65，0x2E，0x63，0x6F，0x6D 表示的是“google.com”0xFE为消息截止字，数据存储见表3-3所示。

另外，TI提供了一套自动NDEF生成工具，填写需要的数据类型和数据，自动帮你转换为NDEF数据 http://www.ti.com/lit/zip/sloa187，利用TFR7970A读头可以直接写入转换后的二进制数据到不同种类标签，另外点对点通信的数据传输最小单元同样是NDEF数据，该问题将在后续章节进行描述。

3.3 Type F 卡片读写规范

Tag-it HF-I Plus 是 TI 支持 ISO15693 协议标准的标签，具有 2048 位（256 字节）用户可编程存储区的标签芯片，具有一个 64 位唯一 UID 序列号，DSFID 和 AFI 标示符。另外还具有用户锁定功能。数据可以由工厂或者终端客户自行修改和读出，内部存储区采用 EEPROM 存储，EEPROM 可以擦除 10 万次，数据可以保持 10 年。用户存储器数据被组织在 4 字节块（存储单元 0-63），详情见图 3-3。每块分别可由工厂或终端用户擦写并可以被锁定，以保护数据不被修改。一旦数据已被锁定时，芯片将变为只读，不可再写入。

TI 15693 标签分为标准类型标签和超级标签，超级标签为标准标签升级版本，增加更多的指令，使得标签的读写更具安全性能（增加带密码写操作，可以注销标签功能）。15693 卡片的操作指令图 3-4 所示。

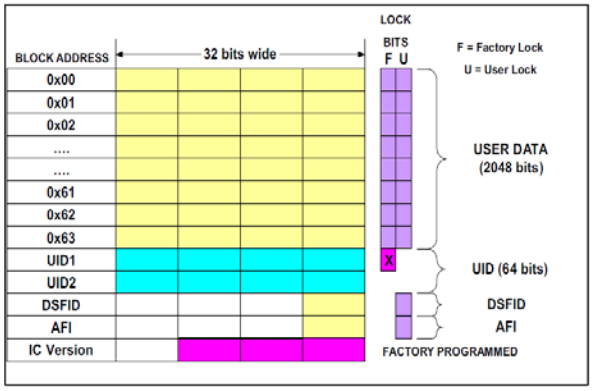


图 3-3 15693 卡片存储空间

Cmd	Type	Description
01	Mandatory	Inventory
02	Mandatory	Stay Quiet
20	Optional	Read Single Block
21	Optional	Write Single Block
22	Optional	Lock Block
23	Optional	Read Multiple Blocks
24	Optional	Write Multiple Blocks
25	Optional	Select
26	Optional	Reset to Ready
27	Optional	Write AFI
28	Optional	Lock AFI
29	Optional	Write DSFID
2A	Optional	Lock DSFID
A2	Custom	Write 2 blocks
A3	Custom	Lock 2 blocks

图 3-4 15693 卡片命令

3.3.1 UID 号编码规则：

UID 为卡片独立唯一序列号，信息为 64 位字长，占用芯片两个块资源，其内容按照 ISO15693 规范定义该卡片类型，芯片制造商以及卡片本身的序列号码，如图 3-5 所示。

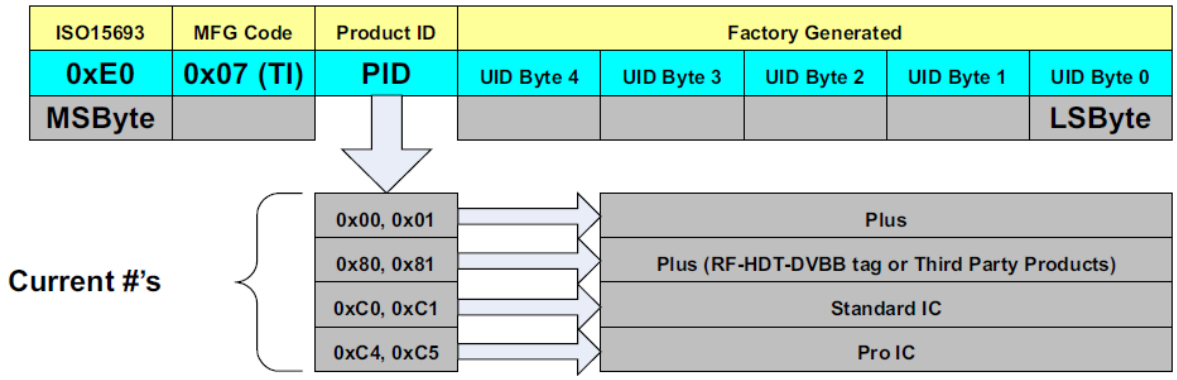


图 3-5　15693 卡片 UID 结构

64 位 UID 的 56-63 位编码规定为 0x E0 表示该卡片是 ISO15693 类型卡片，48-55 为芯片制造商编号，0x07 表示是 TI 生产。41-47 为表示产品编号，TI 不仅生产制造晶源，还生产出售带天线的晶源，俗称内嵌物。

0000 000 (0)bin 00hex or 01hex .. Tag-it HF-I Plus

1000 000 (0)bin 80hex or 81hex .. Tag-it HF-I Plus (Over-moulded Tag or 3rd party products)

1100 000 (0)bin C0hex or C1hex .. Tag-it HF-I Std

1100 010 (0)bin C4hex or C5hex .. Tag-it HF-I Pro

0-40 位则代表工厂出厂序列号。

- The remaining bits will be given a unique TI number.

Example: E007C01234567890 - ISO15693 TI HF-I Standard Inlay

3.3.2 用户存储区结构：

以 256byte 存储空间为例，256byte 被划分为 64 个块，每一块 4byte 字节，读写操作都是按块进行，一次读写 4byte 数据，同时卡片指令支持一次读写多块信息。

3.3.3 AFI 卡片应用类型：

AFI 表示读写器锁定的应用类型，用领域识别号代表读写器的应用方向，用来从所有感应区内的卡片中选出符合应用标准的卡片。AFI 占用单独一个数据块，具有单独的读写操作指令，这个 8 位值保存在-2 块的字节 2 中，只有当读写器命令中有 AFI 标志才有效。AFI 可用命令来编程和锁定。如果卡片不支持 AFI 而读写器命令中 AFI 已设置，不论卡片的 AFI 是否正确，卡片都将不做出响应；如果卡片支持 AFI，卡片将根据下表描述的匹配规则做出响应。

Main	Sub	Application Family
0	0	All
0	1~F	Proprietary
1	0~F	Transport
2	0~F	Financial
3	0~F	Identification
4	0~F	Telecommunication
5	0~F	Medical
6	0~F	Multimedia
7	0~F	Gaming
8	0~F	Data storage
9	0~F	EAN-UCC System
A	0~F	ISO/IEC 15418 (ISO JTC 1/SC 31
B	0~F	IATA (ISO/IEC JTC 1/SC 31
C	0~F	UPU (ISO/IEC JTC 1/SC 31)
D,E,F	0~F	RFU (ISO/IEC JTC 1/SC 31)

图 3-6　15693 卡片 AFI 涵义

高 4bit 定义了主要应用，如果设置为 0 表示所有应用；低 4 位表示具体应用。

例如低四位 10 表示交通运输类;B1 表示邮件包裹类;00 表示所有类型等等，详情见图 3-6 所示。

3.4 TI Type F 卡片读写命令的软件实施：

在初始化 MCU 和 TRF7970A 后，就可以开始从协议层对不同种类的卡片进行读写，下面就 ISO15693 类型的卡片读写进行说明，软件只涉及 Command 发送部分软件。卡片数据读取以及 UID 的读取都会在发送 command 之后接收中断中，读 FIFO 内数据即可得到卡片返回数据。

3.4.1 寻卡(Inventory)：

ISO15693-3 规定一帧标准的寻卡指令如下表 3-4 所述，Invenroty command 为 0x01，mask length and mask value 作为防冲突用途，不是本文讲述的重点，更多要了解防冲突请参见 TI 参考文献 SLOA138 – April 2009。

表 3-4 ISO15693 UID Command

SOF	FLAGS	INVENTORY COMMAND	MASK LENGTH	MASK VALUE	CRC	EOF
	8 bits	8 bits	8 bits	0 to 8 bytes	16 bits	

当 TRF7970A 下达寻卡指令后，若有卡片在天线读取范围内，卡片将回复其 UID 号码给到到 TRF7970A。

这样就完成了一次寻卡。invenroty 程序代码如下所示，buf[0-2]是对 TRF7970A 寄存器进行设置，buf[3-4]是对 FIFO 发送长度设置，从 buf[5]开始则为对卡片进行操作。

```
buf[0] = 0x8f; /* Reset FIFO command */
buf[1] = 0x91; /* Send with CRC */
buf[2] = 0x3d; /* Write continuous from register 1D */
buf[3] = (char) (size >> 8); /* Data for register 1D */
buf[4] = (char) (size << 4); /* Data for register 1E */
buf[5] = 0x06; /* ISO15693 flag with 16 slots bit set*/
buf[6] = 0x01; /* ISO15693 anti collision command code */
buf[7] = length; /* Mask length */
```

3.4.2 读卡操作：

对卡片块进行读操作可分为单块读写和多块读写，卡片读只支持对块进行操作，不支持读块内某一字节进行读操作。块寻址范围为 0-256，每一块最大字节数支持 32 字节（详细参见厂家规格书）。

(0x23)为多块读写指令，多块读需要指定起始地址和连续读的块个数；(0x20)为单块读写指令，单块读需要指定块地址，详细指令见表 3-5 所示。

另外需要注意，根据 15693 规定，读写操作可以带 UID 进行读写，也可以不带 UID 进行读写，如 15693 读操作所述，64Bit UID 可以省略也可以对卡片进行读操作，但是这样做会留下卡片读写管理的潜在危险，再次不针对该问题细究。

表 3-5　ISO15693 块读操作 Command

1	2	3	4	5	6	7	8
SOF	FLAGS	READ MULTIPLE BLOCKS	UID	FIRST BLOCK NUMBER	NUMBER OF BLOCKS	CRC	EOF
	8 bits	8 bits	64 bits	8 bits	8 bits	16 bits	

不带UID的多块卡片读：

```
flags is the ISO15693 flags byte.
buf[0] = 0x8f; /* Reset FIFO command */
buf[1] = 0x91; /* Send with CRC */
buf[2] = 0x3d; /* Write continuous from register 1D */
buf[3] = (char) (size >> 8); /* Data for register 1D */
buf[4] = (char) (size << 4); /* Data for register 1E */
buf[5] = 0x00; /* ISO15693 flag with option flag not set*/
buf[6] = 0x23; /* Read multiple blocks command code */
buf[7] = First block number
buf[8] = Number of blocks
```

带UID卡片的读程序：

```
buf[0] = 0x8f; /* Reset FIFO command */
buf[1] = 0x91; /* Send with CRC */
buf[2] = 0x3d; /* Write continuous from register 1D */
buf[3] = (char) (size >> 8); /* Data for register 1D */
buf[4] = (char) (size << 4); /* Data for register 1E */
buf[5] = 0x00; /* ISO15693 flag with option flag not set*/
buf[6] = 0x23; /* Read multiple blocks command code */
buf[7] to buf[14] contains UID
buf[15] = First block number
buf[16] = Number of blocks

buf[7] = 1C;                    // UID = E0007000006D6AC1C
buf[8] = AC;
buf[9] = D6;
buf[10] = 06;
buf[11] = 00;
buf[12] = 00;
buf[13] = 07;
buf[15] = E0;
```

3.4.3 写卡操作

对卡片块进行写操作易可分为单块读写和多块写，卡片读操作只支持对块进行操作，不支持读块内某一字节进行读操作。15693 规定的块寻址范围为 0-256，每一块最大字节数支持 32 字节（详细参见不同厂家卡片规格书），详细指令参考表 3-6。

(0x21)为多块写指令，多块写需要指定起始地址和连续读的块个数；(0x24)为单块读写指令，单块写需要指定块地址。

另外需要注意，根据 15693 规定，读写操作可以带 UID 进行读写，也可以不带 UID 进行读写，15693 读操作所述，64Bit UID 可以省略也可以对卡片进行读操作，但是这样做会留下多张卡片写管理混乱的潜在危险，再次不针对该问题细究。

表 3-6　ISO15693 块写操作 Command

SOF	Flags	Write multiple block	UID	First block number	Number of blocks	Data	CRC16	EOF
	8 bits	8 bits	64 bits	8 bits	8 bits	Block length	16 bits	

不带UID的多块卡片读：

```
buf[0] = 0x8f; /* Reset FIFO command */
buf[1] = 0x91; /* Send with CRC */
buf[2] = 0x3d; /* Write continuous from register 1D */
buf[3] = (char) (size >> 8); /* Data for register 1D */
buf[4] = (char) (size << 4); /* Data for register 1E */
buf[5] = 0x00; /* ISO15693 flag with option flag not set*/
buf[6] = 0x23; /* Read multiple blocks command code */
buf[7] = First block number
buf[8] = Number of blocks
```

带UID卡片的读程序：

```
buf[0] = 0x8f; /* Reset FIFO command */
buf[1] = 0x91; /* Send with CRC */
buf[2] = 0x3d; /* Write continuous from register 1D */
buf[3] = (char) (size >> 8); /* Data for register 1D */
buf[4] = (char) (size << 4); /* Data for register 1E */
buf[5] = 0x00; /* ISO15693 flag with option flag not set*/
buf[6] = 0x24; /* write multiple blocks command code */
buf[7] to buf[14] contains UID
buf[15] = First block number
buf[16] = Number of blocks
buf[7] = 1C;                                    // with UID = E0007000006D6AC1C
buf[8] = AC;
buf[9] = D6;
buf[10] = 06;
buf[11] = 00;
buf[12] = 00;
buf[13] = 07;
buf[14] = 00;
buf[15] = E0;
```

3.5 Type 2 卡片读写规范

Type 2 类型标签芯有 48byte，64byte，144Byte 用户可编程存储区，具有一个 112 位(7byte)唯一 UID 序列号。芯片还具有用户锁定功能，数据可以由工厂或者终端客户自行修改和读出。内部存储区采用 EEPROM 存储，EEPROM 可以擦除 10 万次，数据可以保持 10 年。用户存储器数据被组织在 4 字节块（存储单元 0-63）。每块分别可由工厂或终端用户擦写并可以被锁定，以保护数据不被修改。一旦数据已被锁定时，芯片将变为只读，不可再写入。卡片 EEPROM 前端有一状态机选择卡片当前状态，接收 command，状态描述如图 3-7 所示。

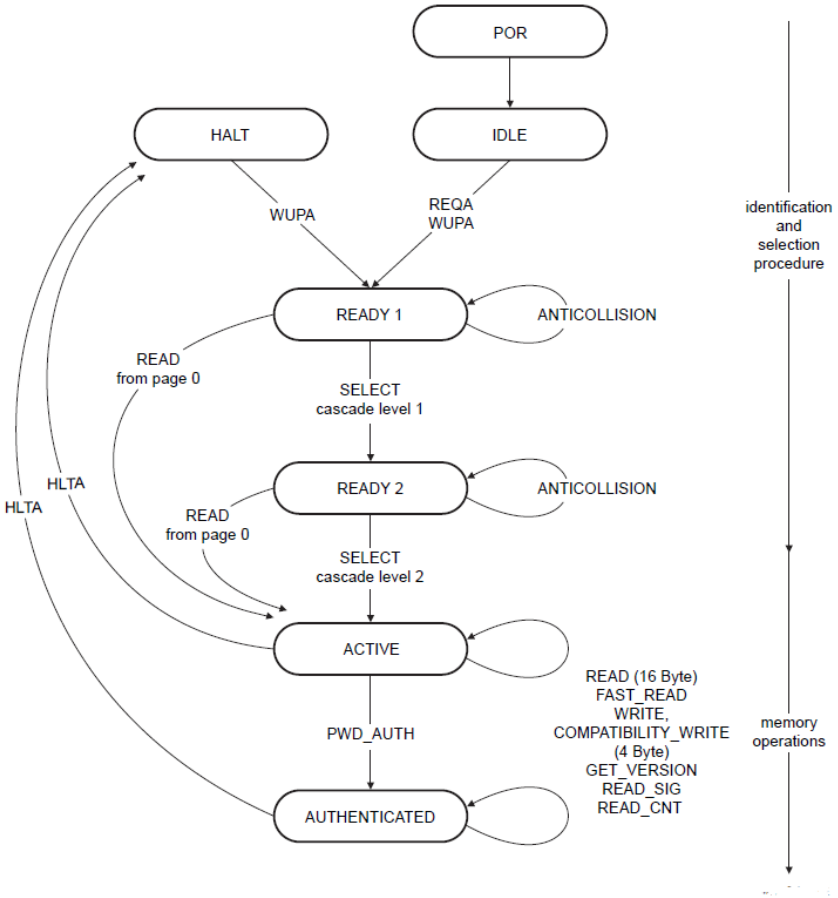


图 3-7　卡片操作状态机

状态机接收读头指令，控制卡片进入相应状态，当接收到非法指令，状态机进入 Halt 等待模式。

3.5.1 内存组织结构

以 NXP NTAG203 为例，该芯片 EEPROM 总共有 42 页(见表 3-7)，每页 4 个字节，总共 168 字节。用户可编程的字节数为 36 页，144 字节。

表 3-7　　type2 卡片内存结构

Page address		Byte number			
Decimal	Hex	0	1	2	3
0	00h	serial number			
1	01h	serial number			
2	02h	serial number	internal	Lock byte 0	Lock byte 1
3	03h	Capability Container (CC)			
4 to 39	04h to 27h	user memory	user memory	user memory	user memory
40	28h	Lock byte 2	Lock byte 3	-	-
41	29h	16-bit counter	16-bit counter	-	-

3.5.2 卡片唯一序列号(UID)

Type 2 标签规定 7 字节唯一序列号(UID)，该序列号存储于芯片第一页至第三页，第一页存储 UID 前 3 字节和一个效验字节，效验字节为 0x88 与第一页前三字节异或：第二页为 UID 余下 4 字节。第三页首字节为第二页四字节数据异或结果，详见图 3-8 所示。自此，卡片唯一序列号被标定出。

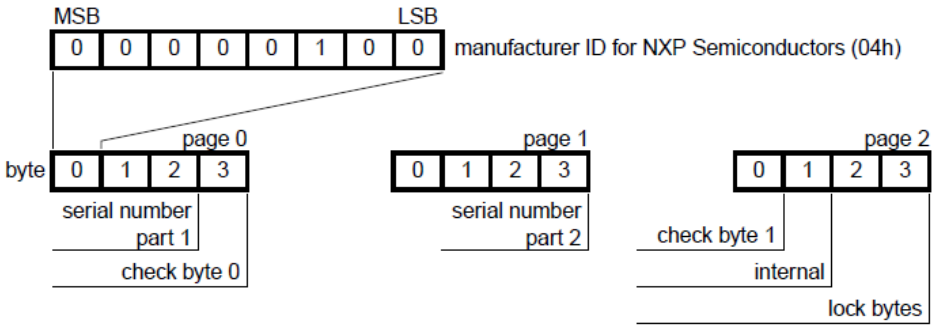


图 3-8　　UID 卡片内容结构

3.6　　Type2 卡片读写命令的软件实施：

下面就 ISO15693 类型的卡片读写进行说明，只针对 Command 发送部分软件。数据读取以及 UID 的读取都会在发送 command 之后接收中断中，读 FIFO 内数据即可得到卡片返回数据，详见图 3-9 所示。

3.6.1 寻卡(REQA)：

卡片处于读头线圈下时，卡片状态机上电，处于空闲模式，等待读头 REQA 指令，当接收到 0x26 指令后，卡片进入下一状态模式：准备 1 模式，开始接收 UID 和进行防冲突。

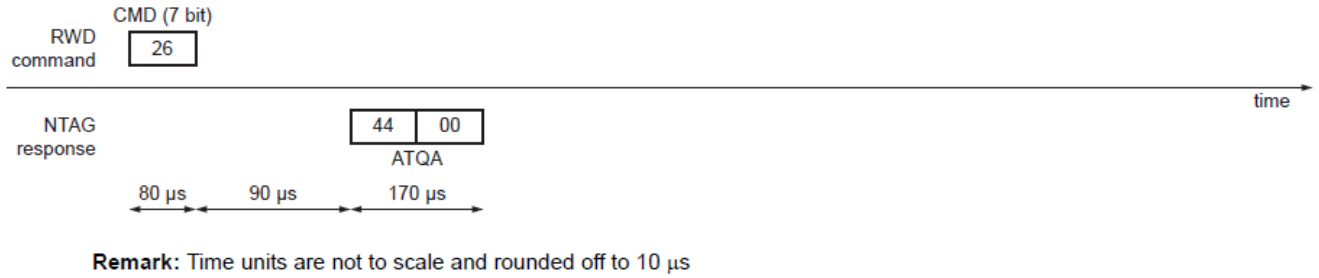


图 3-9　type2 卡片寻卡指令

```
buf[0] = 0x8f; /* Reset FIFO */
buf[1] = 0x90 or 0x91; /* Transmit with no CRC for ANTICOLLISION command or transmit with CRC for
SELECT command*/
buf[2] = 0x3d; /* Write continuous to register 0x1D */
buf[3] = size; /* Data for register 0x1D */
buf[4] = size; /* Data for register 1E */
buf[5] = 0x26; /* Can be 0x93, 0x95 or 0x97 */
```

3.6.2 UID 读取：

卡片在接收到 REQA(0x26)或者 WUPA(0x52)后，进入准备 1 模式，等待主机端 cascade level 1 指令，读头发出 0x93 指令后，状态机回送 4byte 数据，并进入到准备准备模式 2，等待主机端 cascade level 2 指令，读头发出 0x95 指令后，状态机回送 4byte 数据，进入活动模式，可以开始进行读写工作（详见图 3-10 所示）。在此状态机工作时，发送错误的指令，或者非法指令将导致状态机进入暂停状态，等待 WUPA 指令唤醒状态机，详见图 3-11 所示。

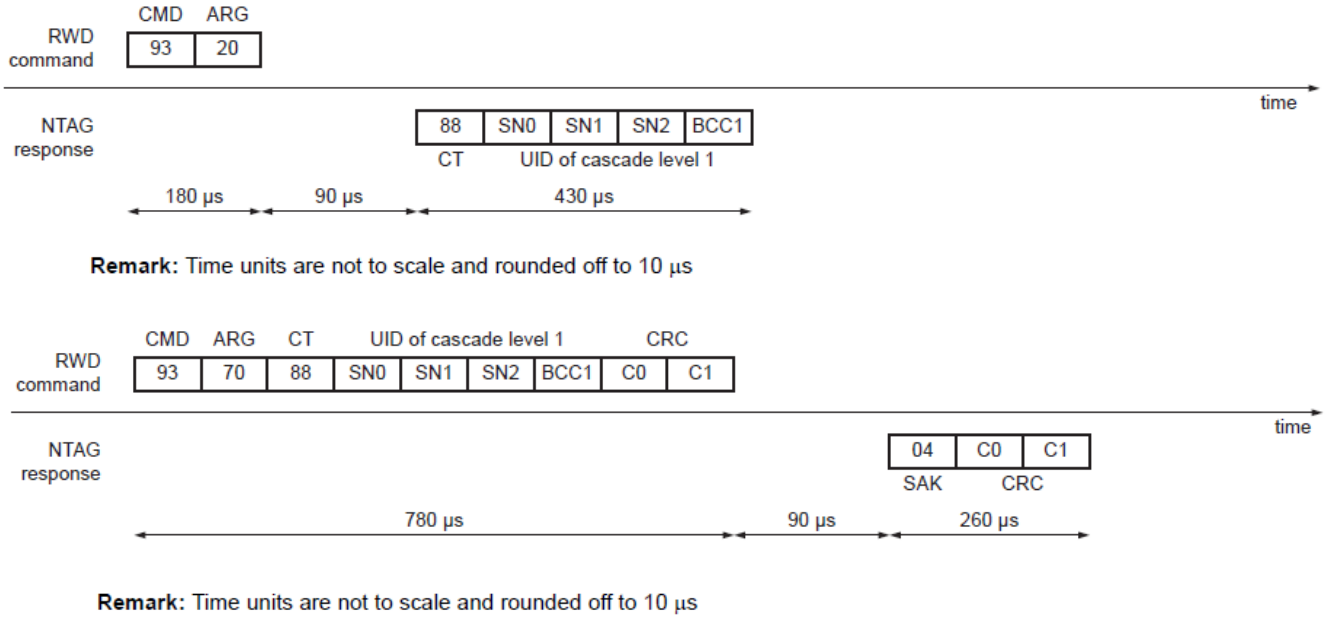


图 3-10　type2 卡片第一级级防冲突操作

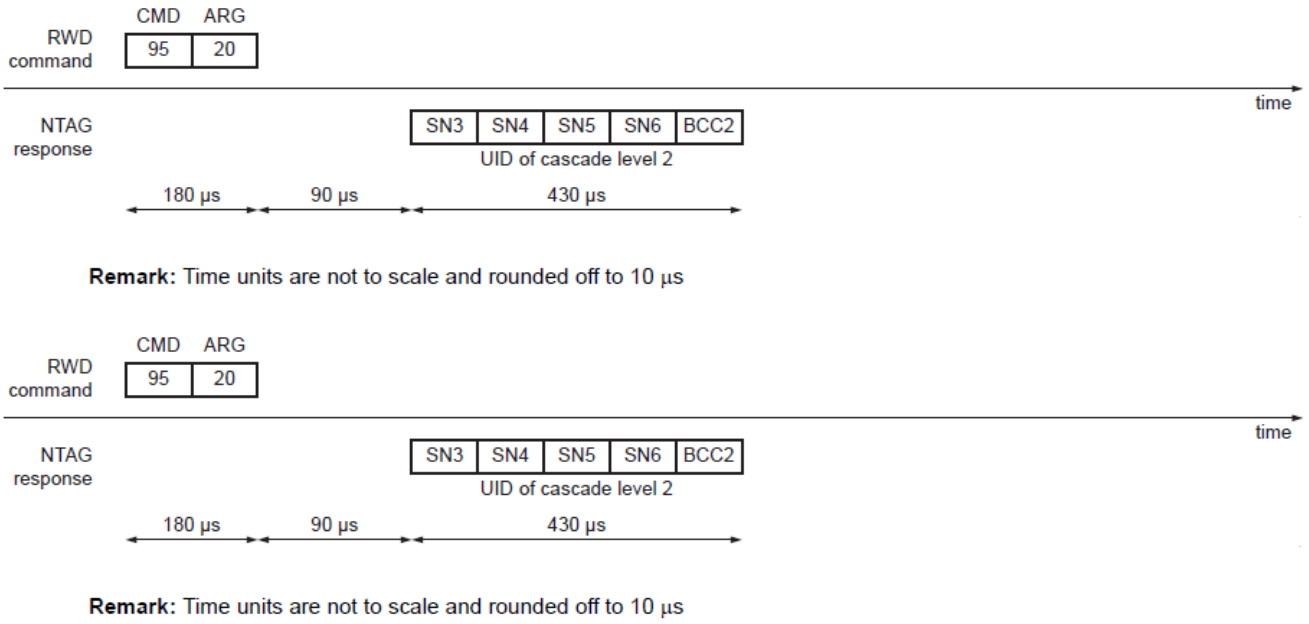


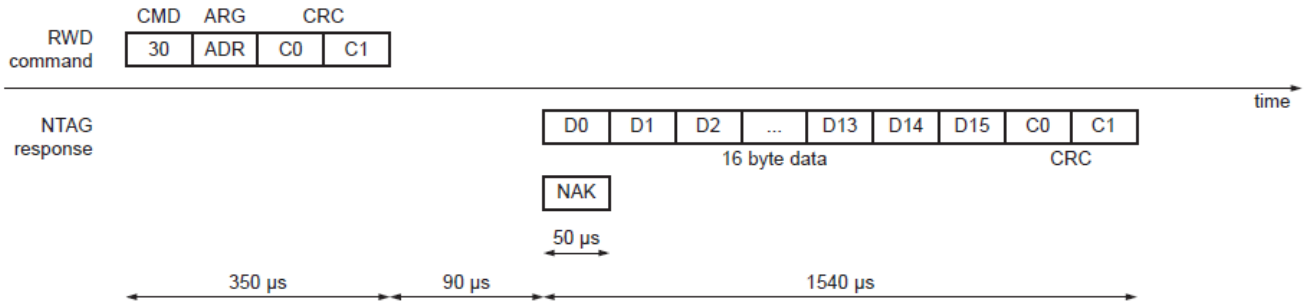
图 3-11　type2 卡片第二级防冲突操作

```
buf[0] = 0x8f; /* Reset FIFO */
buf[1] = 0x90 or 0x91; /* Transmit with no CRC for ANTICOLLISION command or transmit with CRC for
SELECT command*/
buf[2] = 0x3d; /* Write continuous to register 0x1D */
buf[3] = 0x00; /* Data for register 0x1D */
buf[4] = NVB & 0xf0; /* Data for register 1E */
if ((NVB & 0x07) != 0x00) /* Number of complete bytes – Data for reg. 0x1E */
        buf[4] |= ((NVB & 0x07) << 1) + 1; /* Number of broken bits with Broken byte flag set in reg. 0x1E */
buf[5] = select; /* Can be 0x93, 0x95 or 0x97 */
buf[6] = NVB; /* Number of valid bits */

buf[7] to buf[7+searchlength] = UID search criterion
```

3.6.3 读卡操作：

状态机进入活动模式下才可以对卡片进行读操作，卡片的读需要指定需要读取页的地址。数据从第 4 页开始，一次连续顺序读取 4 页，16byte 数据。例如：指定读取的地址起始页从第 4 页开始，会一次读取 0x04, 0x05, 0x06, 0x07 四页数据，如果，读取的起始地址为最后一页 0x29，则读取的 4 页数据为 0x29, 0x00, 0x01, 0x02, 0x03，详见图 3-12 所示。



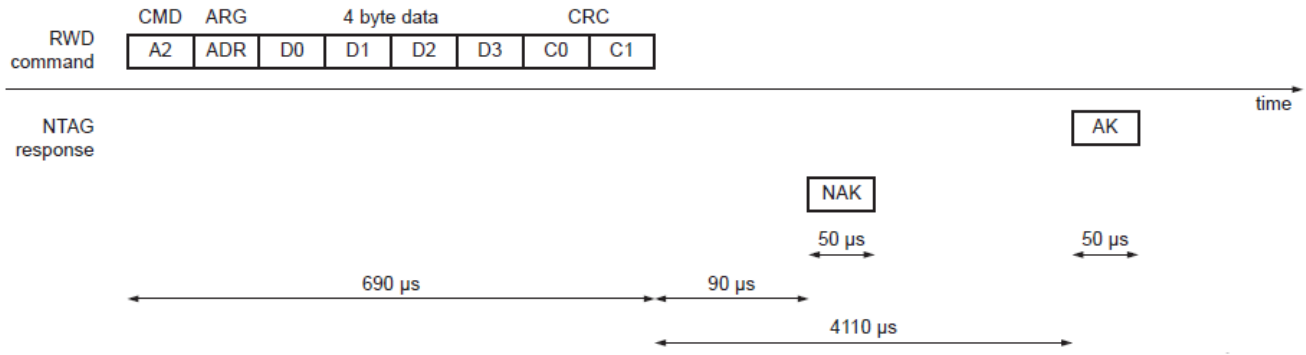
Remark: Time units are not to scale and rounded off to 10 µs

图 3-12　type2 卡片 数据块读操作

```
buf[0] = 0x8f; /* Reset FIFO */
buf[1] = 0x90 or 0x91; /* Transmit with no CRC for ANTICOLLISION command or transmit with CRC for SELECT command*/
buf[2] = 0x3d; /* Write continuous to register 0x1D */
buf[3] = size; /* Data for register 0x1D */
buf[4] = size; /* Data for register 1E */
buf[5] = 0x30; /* read cmd */
buf[6] = 0xaddress; /* read  from */
```

3.6.4 写卡操作：

状态机进入活动模式下才可以对卡片进行写操作，卡片的读需要指定需要指定写页地址。写数据还可以对 0x02 页进行写操作，用于锁住用户读写页，使得其数据不能被再次更改。写数据指定地址后，会加上 4byte 数据，及对该页完成写操作，详见图 3-13 所示。



Remark: Time units are not to scale and rounded off to 10 µs

图 3-13　type2 卡片 数据块写操作

```
buf[0] = 0x8f; /* Reset FIFO */
buf[1] = 0x90 or 0x91; /* Transmit with no CRC for ANTICOLLISION command or transmit with CRC for
SELECT command*/
buf[2] = 0x3d; /* Write continuous to register 0x1D */
buf[3] = size; /* Data for register 0x1D */
buf[4] = size; /* Data for register 1E */
buf[5] = 0xA2; /* write cmd */
buf[6] = address; /* read  from */
buf[7] = data;
buf[8] = data;
buf[9] = data;
buf[10] = data;
```

4. TRF79A0 P2P 通讯

TRF7970A 芯片除了可以完成传统的卡片读写功能，还可以完成点对点数据传输以及卡片模拟功能。NFC 论坛在针对点对点数据传输制定了一套标准:18092/NFCIP-1 协议，符合该协议栈配合 14443A 以及 Felica 物理层射频传输用以完成点对点通讯功能，P2P 模式支持 106Kbit/s,212Kbit/s,424Kbit/s 速度进行数据传输，协议规范如图 4-1 所示。

	NFC devices		Contactless Smart Card systems	
	Initiator	**Target**	**Reader/writer**	**Contactless smart cards**
Communication principle	**Active communication mode:** Both initiator and target generate RF field		Reader generates RF field and card answers using load modulation	
	Passive communication mode: Initiator generates RF field and target answers using load modulation			
Initialization	**Active communication mode:** RF collision avoidance		Initialization and anticollision	
	Passive communication mode: Initialization and Anticollision			
Speed at initialization [kbit/s]	106, 212, 424		106 for ISO 14443-A 212, 424 for FeliCa	
Communication protocol	NFC IP-1 data exchange protocol		ISO 14443 Transmission protocol MIFARE®: fixed command set FeliCa™: fixed command set	
Speed at Communication protocol[kbit/s]	106, 212, 424		106, 212, 424, 848 for ISO 14443-A incl. amendments 106 for MIFARE® 212 for FeliCa™	

图 4-1 18092 P2P 协议规范

4.1 NFC 通信模式

NFC 通信通常在发起设备和目标设备间发生，任何的 NFC 装置都可以为发起设备或目标设备。两者之间是以交流磁场方式相互耦合，并以 ASK 方式或 FSK 方式进行载波调制，传输数字信号。发起设备产生无线射频磁场来初始化 NFCIP-1 的通信（调制方案、编码、传输速度与 RF 接口的帧格式）。目标设备则响应发起设备所发出的命令，并选择由发起设备所发出的或是自行产生的无线射频磁场进行通信。

4.1.1 主动模式

在主动模式下，每台设备要向另一台设备发送数据时，都必须产生自己的射频场。如图 4-2 所示，发起设备和目标设备都要产生自己的射频场，以便进行通信。这是点对点通信的标准模式，可以获得非常快速的连接设置。

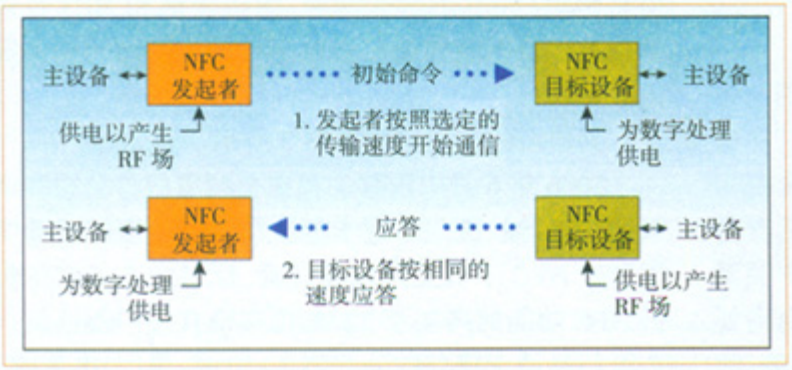


图 4-2　主动通信模式

4.1.2 被动模式

在被动模式下，NFC 发起设备（也叫主设备，启动 NFC 通信的设备），在整个通信过程中提供射频场（RF-field），如图 4-3 所示。它可以选择 106kbit/s、212kbit/s 或 424 kbit/s 其中一种传输速度，将数据发送到另一台设备。另一台设备称为 NFC 目标设备（从设备），不必产生射频场，利用感应的电动势提供工作所需的电源，使用负载调制（Load Modulation）技术进行数据收发。

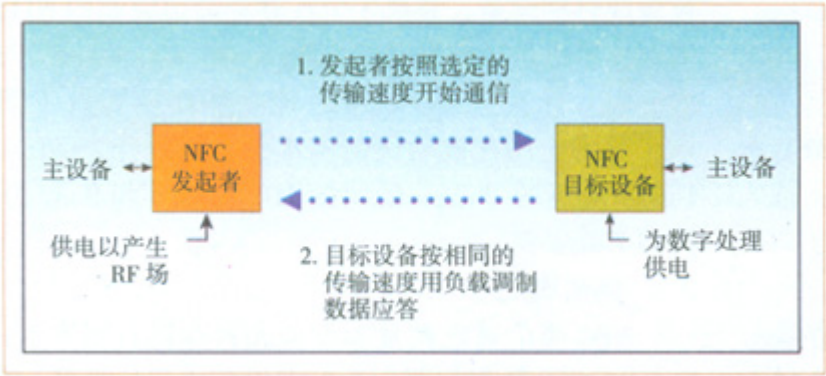


图 4-3　被动通信模式

4.2　利用 TFR7970 初始化 P2P 通讯：

点对点通讯与传统的 RFID 读卡的区别在于，虽然通讯两端有主从之分，但是数据的传输是双向传输，从端可以接受也可以发送数据。为了使 TRF7970A 在 P2P 模式下工作，需要做如下初始化工作：

1. 初始化 MCU，配置 MCU 使用内部 DCO 时钟，设置 SPI 处于 4 线模式，使能 TRF7970A, 写 command (0x03)初始化 TRF7970A。
2. 写 command (0x00)　设置 TRF7970A 处于空闲状态。
3. 对 TRF7970A 芯片(0x09)寄存器进行配置，配置 MCU 分频 6.78MHz 时钟输出, OOK 100%模式。
4. 配置 MCU 时钟为外部 6.78M HZ 时钟，并切换 MCU 时钟源为 6.78MHZ 外部时钟，重新初始化 UART SPI。
5. 配置 TF7970A　ISO 管理寄存器(0x01)，使得 TRF7970A 处于 NFC 模式，选择 NFC 处于主动模式或者被动模式，例如这里设置(0x01)寄存器的值为(0x23),TRF7970A 处于 424kbps,　NFC 被动模式。
6. 设置 TFR7970A 状态寄存器(0x00)，打开射频收发模块，并设置全功率发射模式。
7. 设置 NFC UID 卡片长度：写寄存器(0x18)设置 UID 长度，有 4 位，7 位和 10 位 UID 长度设置。
8. 写 UID，写寄存器(0x17),(0x17)内的数据即为 UID。
9. 设置 IRQ 中断事件：写(0x14)寄存器，发送 32 字节产生一次中断，接收 96 字节产生一次 IRQ 中断。
10. 写 command 0x16，使 TRF7970A 接收模块处于复位状态，以增强抗干扰性能。
11. 清 IRQ 状态寄存器 0x0C 内的值，等待 IRQ 中断。
12. IRQ 中断，判断何种中断类型：接收中断, 发送中断, 冲突中断，并进行相应处理。

4.3 P2P 协议栈分析：

设置完 TRF7970A 芯片工作在 P2P 模式下之后需要按照 ISO18092 搭建协议栈，符合 NFC 数据传输标准，才能完成 P2P 数据通讯。

ISO18092 规定了 NFC 连接和传输的三步骤，NFC 主设备需要寻找，发起 NFC 连接，称之为 Sense Request，NFC 从设备受到主设备的 Request 命令后，返回 Sense Response 指令。从而完成第一层数据连接过程，主设备接收到 Sense response 指令发起第二层属性连接指令 Attribute Request，从设备根据主设备的指令回复 Attribute Response 指令，第二层连接完毕，此时，NFC 主从设备成功连接上。第三层开始传输数据，主端发起 DEP_REQ 数据传输，从端收到收据并回传 DEP_RES 数据到主端，这样完整的 NFC 数据传输就。

值得注意的是，当 NFC 主从设备完成第一层和第二层建联以后，就可以一直保持在第三层传输数据，根据数据传输的大小，NFC 论坛制定 LLCP 和 SNEP 标准用以控制,协调,有效的进行 NFC 数据传输。传输流程如图 4-4 所示。用一个关系可以说明 LLCP,SNEP,NDEF 的关系：｛ LLCP ｛ SNEP ｛ NDEF ｛ RTD ｝｝｝｝。LLCP 控制数据传输，通常携带着 SNEP 数据，SNEP 作为 LLCP 的子集，通常携带着 NDEF 数据；NDEF 携带了 RTD 数据； RTD 数据则是实际的应用数据。

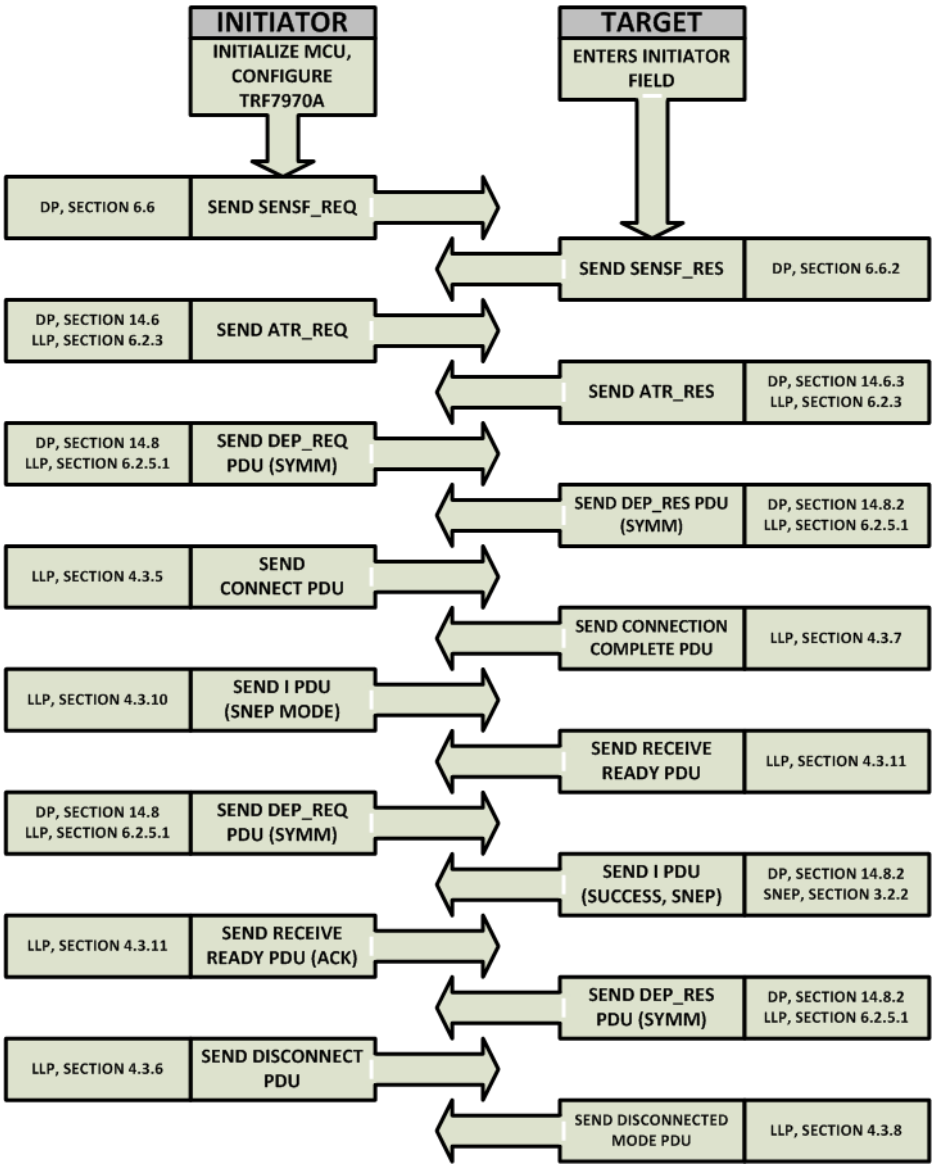


图 4-4　P2P 通讯协议标准流程

4.4　简单 P2P 协议收发状态机：

参考代码出处于 http://www.ti.com/lit/zip/slaa512.工程的 nfc.c 文件，函数名称为 NFC_ProcessReceivedData()，该函数专门用来控制 NFC 三层数据连接的建立与解析：

```
Switch (command)
{
       Case SENSF_REQ:
       /* SENSF_REQ:
        | 0        | 1   |   2:3    |  4    |  5  |
        | LEN:6 | 00 | FFFF  | RC  | TSN |
         SENSF_RES:
        | 0       | 1:2  | 3:12   |  13  |  14 | 15  | 16  |17   | 18+     |

     */
       //RADIO_READSINGLE(0x0B, &temp2);
       //temp2 = temp2 + 1;
       // for debug usage
       slots = data[5];

       response[0] = 20; // Length
       response[1] = 0x01;
       for(i = 0; i < 8; i++){
          response[2 + i] = TargetCID[i];
       }
       ……..
       response[19] = 0xAB;

       __delay_cycles(300);   //2.4ms


       for(i = 0; i < (slots + 1); i++){
          if(i == 0){                                                    //send the reply in the calculated
timeslot
             {
                NFCSend(&response[0]);
                P1OUT ^= BIT4;              //led flashing
                NFC_State = PRESELECTED_STATE; // If Ok, move to selected state
                break;
             }
          }
          __delay_cycles(400);  //1.2ms
       }

       break;

       Case ATR_REQ:
      /* ATR_REQ:
       | 0            | 1:2     | 3:12      |  13   |  14  | 15  | 16   | 17+     |
       | LEN:17+n | D400 | NFCIDi | DIDi | BSi | BRi | PPi | Gi[0:n] |
        ATR_RES:
       | 0           | 1:2     | 3:12     |  13    |  14  | 15   | 16   |17    | 18+     |
       | LEN:17   | D501 | NFCIDt | DIDt | BSt | BRt | TO  | PPt | Gi[0:n] |
     */
          //RADIO_READSINGLE(0x0B, &temp2);
          //temp2 = temp2 + 1;
          //debug usage
//          P1OUT ^= BIT3;
```

```
response[0] = 28; // Length
          // ATR_RES command
          response[1] = (ATR_RES&0xFF00)>>8;
          response[2] = (ATR_RES&0x00FF);

          if ((data[3] != TargetCID[0]) || (data[4] != TargetCID[1]))
             return RET_ERR;

          for(i = 0; i < 10; i++){
                    response[3 + i] = TargetCID[i];
                    }
         response[13] = 0x00;              // DIDt is same as DIDi
         ……..
response[27] = 0xFF;

                      __delay_cycles(8000);   //2.4ms

        for(i = 0; i < (slots + 1); i++){
          if(i == 0){                                                    //send the reply in the calculated timeslot
             {
                NFCSend(&response[0]);
                P1OUT ^= BIT3;
                NFC_State = SELECTED_STATE; // If Ok, move to selected state
                break;
             }
          }
          __delay_cycles(4000);
       }
       break;

       case DEP_REQ:
      /* DEP_REQ:
        | 0       | 1:2  | 3    |
        | LEN     | D406 | PFB  |
         LLCP:
        | 4:5                          | 6       | 7          | 8:22             |
        | DSAP, PTYPE, SSAP  | Type  | Length  | Value          |
        | 0521               | 06    | 0F      | Service Name    |
        | SDP, CONNECT, 21   | SN    | n       | com.android.npp |
     */
          // Check PFB  type
          if ((buffer[3]&0xE0) == 0x00) //information PDU
          {

            if (buffer[4] == 0 && buffer[5] == 0)            //反向回传数据
            {
              resp = Android_Back_Sending(buffer);
            }
            else if( buffer[4] == 0x85 && buffer[5] == 0x90 ) //暂时还没有用到
            {
              resp = Android_CC_received(buffer);
            }
```

```
        else if(buffer[3] == 0 && buffer[4] == 0x05 && buffer[5] == 0x21) //接收数据一段数据
        {
          resp = Android_CONNECT_received(buffer+6);
        }
        else if(buffer[4] == 0x43 && buffer[5] == 0x21)
        {
          resp = Android_Finish_received(buffer+6);      //接收到全部数据，置接收完成标志位
        }

        else{
          __no_operation();
        }

        for(i = 0; i < (slots + 1); i++){
            if(i == 0){                                          //send the reply in the calculated timeslot
               {
                  NFCSend(resp);
                  P1OUT ^= BIT2;
                  NFC_State = SELECTED_STATE; // If Ok, move to selected state
                  break;
               }
            }
            __delay_cycles(4000);
          }
       }
       else{
         __no_operation();
       }
       // else ignore
    break;

    default:
    __no_operation();
    break;

  }
```

参考文档

1. *TRF7970A User Guide*
2. *Implementation of the ISO15693 Protocol*
3. *Using Texas Instruments Tag-it™ HF-I Transponder*
4. *NTAG203F Datasheet*
5. *NFCForum-TS-Type-2-Tag_1.1*
6. *Implementation of ISO14443A Anti Collision Sequence in TRF7970A*
7. *Implementing an SSP A2DP Bluetooth® Audio Receiver Using NFC With TRF7970A*
8. *NFC Data Exchange Format (NDEF) Technical Specification NFC Forum TM NDEF 1.0*
9. *Telecommunications and information exchange between systems — Near Field Communication — Interface and Protocol (NFCIP-1).*

有关 TI 设计信息和资源的重要通知

德州仪器 (TI) 公司提供的技术、应用或其他设计建议、服务或信息，包括但不限于与评估模块有关的参考设计和材料（总称“TI 资源”），旨在帮助设计人员开发整合了 TI 产品的 应用； 如果您（个人，或如果是代表贵公司，则为贵公司）以任何方式下载、访问或使用了任何特定的 TI 资源，即表示贵方同意仅为该等目标，按照本通知的条款进行使用。

TI 所提供的 TI 资源，并未扩大或以其他方式修改 TI 对 TI 产品的公开适用的质保及质保免责声明；也未导致 TI 承担任何额外的义务或责任。TI 有权对其 TI 资源进行纠正、增强、改进和其他修改。

您理解并同意，在设计应用时应自行实施独立的分析、评价和 判断， 且应全权负责并确保 应用的安全性， 以及您的 应用 （包括应用中使用的所有 TI 产品）） 应符合所有适用的法律法规及其他相关要求。你就您的 应用声明，您具备制订和实施下列保障措施所需的一切必要专业知识，能够 (1) 预见故障的危险后果，(2) 监视故障及其后果，以及 (3) 降低可能导致危险的故障几率并采取适当措施。您同意，在使用或分发包含 TI 产品的任何 应用前， 您将彻底测试该等 应用 和该等应用所用 TI 产品的 功能。除特定 TI 资源的公开文档中明确列出的测试外，TI 未进行任何其他测试。

您只有在为开发包含该等 TI 资源所列 TI 产品的 应用时， 才被授权使用、复制和修改任何相关单项 TI 资源。但并未依据禁止反言原则或其他法理授予您任何TI知识产权的任何其他明示或默示的许可，也未授予您 TI 或第三方的任何技术或知识产权的许可，该等产权包括但不限于任何专利权、版权、屏蔽作品权或与使用TI产品或服务的任何整合、机器制作、流程相关的其他知识产权。涉及或参考了第三方产品或服务的信息不构成使用此类产品或服务的许可或与其相关的保证或认可。使用 TI 资源可能需要您向第三方获得对该等第三方专利或其他知识产权的许可。

TI 资源系“按原样”提供。TI 兹免除对 TI 资源及其使用作出所有其他明确或默认的保证或陈述，包括但不限于对准确性或完整性、产权保证、无屡发故障保证，以及适销性、适合特定用途和不侵犯任何第三方知识产权的任何默认保证。

TI 不负责任何申索，包括但不限于因组合产品所致或与之有关的申索，也不为您辩护或赔偿，即使该等产品组合已列于 TI 资源或其他地方。对因 TI 资源或其使用引起或与之有关的任何实际的、直接的、特殊的、附带的、间接的、惩罚性的、偶发的、从属或惩戒性损害赔偿，不管 TI 是否获悉可能会产生上述损害赔偿，TI 概不负责。

您同意向 TI 及其代表全额赔偿因您不遵守本通知条款和条件而引起的任何损害、费用、损失和/或责任。

本通知适用于 TI 资源。另有其他条款适用于某些类型的材料、TI 产品和服务的使用和采购。这些条款包括但不限于适用于 TI 的半导体产品 (http://www.ti.com/sc/docs/stdterms.htm)、评估模块和样品 (http://www.ti.com/sc/docs/sampterms.htm) 的标准条款。

邮寄地址：上海市浦东新区世纪大道 1568 号中建大厦 32 楼，邮政编码：200122