摘 要

随着电子信息技术的发展，智能卡（IC 卡）已经在我们的生活中随处可见。射频识别卡正逐渐取代传统的接触式 IC 卡，成为智能卡领域的新潮流。研究、开发射频识别卡的读写技术与读写设备，对其推广有着重要的实际意义。

本文首先介绍了射频识别卡及射频识别系统的工作原理。为了使应用系统的开发人员无需掌握复杂的射频识别技术就可快速开发射频识别卡应用产品，本文基于模块化的设计思想设计开发了射频识别卡 **Mifare1** 的读写模块。该读写模块不仅能完成对射频识别卡的控制和读写操作，而且可供用户在其基础上进行二次开发。文中详细讨论了读写模块的具体实现。硬件部分介绍了系统的组成、MCU 与读写芯片的接口设计与硬件电路的实现。软件部分重点阐述如何实现射频识别通信的底层驱动，并在此基础上将卡片的操作用函数形式封装，以供用户调用。函数的封装严格按照软件工程的要求，具有硬件无关性，方便用户二次开发。然后，给出了读写模块的两个应用实例：RFID 卡通用读写卡器及带网络接口的考勤机，简要的介绍了其设计方法和用途。

最后，对本文所做工作进行了总结，并给出今后研究工作的展望。

关键词：射频识别，IC 卡，读写设备

作者：徐丽华

指导老师：王宜怀

ABSTRACT

With the rapid development of electronic information technology ，smart cards (IC card) are now very popular in our life. Radio Frequency Identification (RFID) card is becoming a new fashion in the application field of smart card, replacing the traditional contacting IC card. So it is of great practical significance to study the technology of RFID Card and develop the read/write device of RFID card for its generalization.

The common concept of the RFID card and the basic working principle of RFID system are explained chiefly in the first part of this paper. Then, a read/write module of Mifare1 RFID card is developed, based on the modular designing mind. The users of application system may develop their practical products rapidly with our module without understanding the details of RFID technology. The read/write module can not only do the control and read/write operations of the RFID card, but also can be redeveloped. The way to implement the read/write module is discussed in detail, including two parts, the hardware and the software. The former part includes the constitutes of the system, the design of the interface between MCU and the chip of RFID module, and the realization of the hardware circuits. And the latter mainly introduces how to drive the RFID communication and encapsulate the operation of the card into functions which can be transferred by the customers conveniently. The functions are irralated with hardware by doing its encapsulation according to the rules of software engineering.Then, two application examples based on read/write module are given, one is a general read/write device and the other is a check-in machine with network interface.

At last, all the work are summarized and a research prospect of the subject in future is promised .

Keywords：Radio Frequency Identificatiion(RFID)，IC card，read/write device，

Author:Xu Lihua

Supervisor:Wang Yi huai

目录

第一章 概述

射频识别卡技术是近几年发展起来的一项新技术，它成功地结合射频识别技术和IC卡技术解决了无源（卡中无电源）和免接触的难题，是电子信息技术领域的一大突破。由于其方便性、耐用性，且可高速通信和多卡操作等特点，射频识别卡在门禁安防、身份识别、公共交通等众多领域正逐渐取代传统的接触式IC卡，在市场上所占的份额越来越大。射频识别卡的应用日益广泛，相应的促进了嵌入式应用领域开发人员对其读写技术的研究和对读写设备的开发。根据实际应用的需求，本文将应用Freescale（原Motorola）的新型8位MCU实现射频识别卡的射频读写技术，并将其封装成可供用户二次开发的读写模块。

本章首先介绍了射频识别技术、射频识别卡的概念及射频识别卡读写设备的一般组成，然后提出了射频识别卡读写模块的构思，最后介绍了本文的研究内容。

1.1 射频识别卡

1.1.1 关于射频识别技术

射频识别（Radio Frequency Identification, RFID）技术是一种非接触自动识别技术，利用射频信号通过空间耦合(电感或电磁耦合)实现无接触信息传递并通过所传递的信息达到识别目的[1]。

射频识别技术的显著优点在于非接触性，因此完成识别工作时无须人工干预，能够实现识别自动化且不易损坏；可识别高速运动物体并可同时识别多个射频标签，操作快捷方便；射频标签不怕油渍、灰尘污染等恶劣的环境。当前，射频识别技术在国内最广泛的应用是射频识别卡。

1.1.2 智能卡(IC卡)

智能卡（“Smart Card”），也称作集成电路卡（Integrated Circuit card），即IC卡。它一般指将集成电路芯片嵌装于塑料等基片上制成的卡片，外形与磁卡相似[2]，芯片具有存储、加密及数据处理等功能。1970年，法国人罗兰德·莫瑞诺（Roland Moreno）第一次将可进行编程设置的IC芯片放于卡片中[3]，使卡片具有更多的功能，由此诞生了IC卡。在此后的时间里，随着超大规模集成电路技术、计算机技术以及信息安全技术的发展，IC卡技术也更趋成熟。现在，IC卡产品已经进入到金融、电

信、交通、医疗、身份证明等各种领域[3]。

根据读写方法的不同，IC 卡可以分为接触式 IC 卡和非接触式 IC 卡。两种卡的集成电路均密封在塑料卡基片内部，可防水，防尘，防磁。接触式 IC 卡的表面可以看到一个方型镀金接口，共有八个或六个镀金触点，用于与读写器接触，通过电流信号完成读写。非接触式 IC 卡的卡内除了包含 IC 卡电路，还含有相关射频收发电路及天线线圈。IC 卡在一定距离内即可接收读写器的信号，实现非接触读写。

1.1.3 射频识别卡

射频识别卡（简称射频卡、RFID 卡）也被称作非接触式 IC 卡（Contactless Smart Card，CSS）或非接触 IC 卡、非接触卡、感应卡[4]，诞生于 20 世纪 90 年代初。由于成功地结合射频识别技术和 IC 卡技术，解决了无源（卡内无电池）和免接触的难题[5]，RFID 卡拥有磁卡和接触式 IC 卡不可比拟的优点。其一问世便立即引起广泛关注，并以惊人的速度得到推广应用。

RFID 卡由 IC 芯片、感应天线组成，完全密封在一个标准 PVC 卡片中，无外露部分（见图 1-1）。

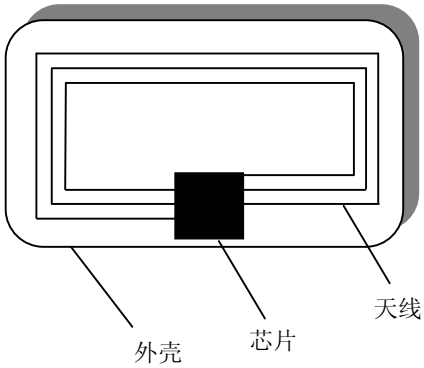


图 1-1 RFID 卡结构示意图（卡上带芯片和天线）

1.1.4 RFID 卡的优点

与接触式 IC 卡相比，RFID 卡具有以下优点：

(1) 高可靠性：由于无触点，避免了由接触读写而产生的各种故障。提高了抗静电和环境污染(如油烟、灰尘、水汽等)的能力，因此提高了使用的可靠性、读写设备和卡片的使用寿命。

(2) 易用性：操作方便、快捷，无需插拔卡，完成一次操作只需 0.1～0.3 秒[4]。使用时，卡片可以任意方向掠过读写设备表面。

(3) 高安全性：序列号是全球唯一的．出厂后不可更改。卡与读写设备之间采用双向互认验证机制：即读写器验证卡的合法性，同时卡验证读写器的合法性。通讯过

程中所有的数据都加密。卡片上不同分区的数据可用不同的密码和访问条件进行保护。

(4) 高抗干扰性：对有防冲突电路的 **RFID** 卡，在多卡同时进入读写范围内时，读写设备可一一对卡进行处理，抗干扰性高。

(5) 一卡多用：卡片上的数据分区管理，可以很方便的实现一卡多用、一卡通。

(6) 多种工作距离：作用距离从几厘米到几米，适应不同的应用场合。

1.1.5 RFID 卡的应用

RFID 卡以其方便交易、速度快、应用领域广而增长迅速，从长远角度看，**RFID** 卡将会替换目前广泛使用的接触式 **IC** 卡。在国内，**RFID** 卡主要应用在公共交通、身份识别、门禁控制等领域。

(1) 公共交通 **RFID** 卡应用潜力最大的领域之一就是公共交通领域[6]。例如公交、地铁，乘客将 **RFID** 卡做的电子车票放在钱包或者包里就可以检票，方便快捷。公交经营者也宜于管理、减少支出。

(2) 身份识别 使用 **RFID** 卡做为身份识别方式，比一般的证件卡片具有更高的防伪性，存储更多信息，便于管理。我国第二代公民身份证即采用 **RFID** 卡，卡中输入生物特征信息及身份信息[7]，以进一步加强防伪，同时便于全国实时管理。

(3) 门禁控制 采用基于 **RFID** 卡的控制系统，可以自动检查每个人进入大楼、管理区的准入权限，并记录出入时间。

另外，还有高速公路收费，停车场收费，加油站收费，智能卡水表、电表、煤气表等应用，使用非接式 **IC** 卡都是首选。

RFID 卡的应用前景日益广泛，其应用关键需要大量的读写设备。

1.1.6 RFID 卡读写设备

RFID 卡读写设备（或称阅读设备、读写器）是连接 **RFID** 卡与应用系统间的桥梁，是 **RFID** 卡应用中至关重要的一个环节。**RFID** 卡读写设备的基本任务就是启动 **RFID** 卡，与 **RFID** 卡建立通信，在应用系统和卡片间传递数据[2]。

RFID 卡读写器将要发送的信息编码后加载到一固定频率的载波上，当 **RFID** 卡（卡片内有一个谐振电路，其频率与读写器发送的载波频率相同）进入读写器的工作区域后，谐振电路产生共振并产生电荷积累，当电荷积累到一定数值时，就能为 **RFID** 卡内的电路提供工作电压，使 **IC** 卡内的芯片开始正常工作，处理读写器发送的数据信息。

一般来说，完整的 **RFID** 卡读写设备的基本结构包括以下几个部分（参见图 **1—2**）：

(1) MCU：MCU 是读写设备的数据处理控制核心。它不仅要控制射频处理模块完成对 **RFID** 卡的读写，还要负责通过通信接口与主机或应用系统进行通信以及对键盘、显示设备等其他外部设备的控制。

(2) 射频处理模块：射频处理模块负责射频信号的处理和数据的传输，完成对 **RFID** 卡的读写。射频处理模块可以采用厂商提供的专用模块或射频基站芯片[8]。射频基站芯片即 **RFID** 卡读写芯片，也称射频读写芯片。

(3) 天线：天线的作用就是产生磁通量，为卡片提供电源，在读写设备和卡片之间传送信息。天线的有效电磁场范围就是系统的工作区域。

(4) MCU 与主机的通信接口以及键盘、**LED/LCD** 显示等其它外部设备。

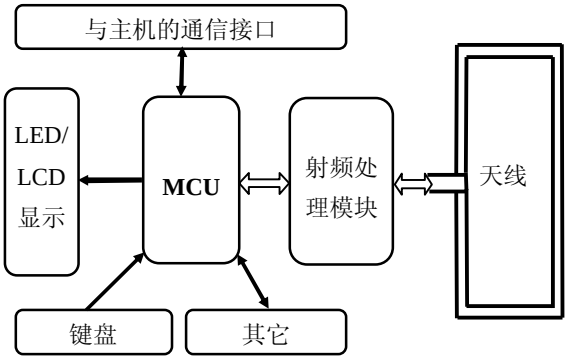


图 1-2 RFID 卡读写设备系统组成图

1.2 关于本课题

1.2.1 RFID 卡读写模块构思

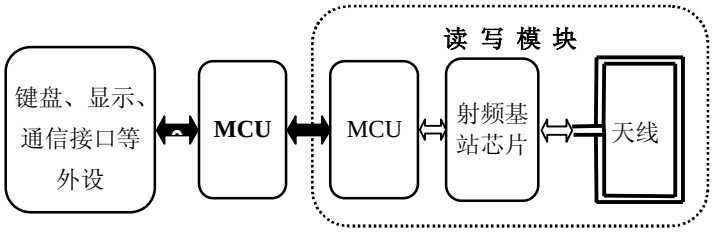
国内市场上的 **RFID** 卡读写设备正处于发展阶段。最先出现的第一代读写设备，其射频处理部分采用的是专用读写模块：体积大，功耗高，且价格昂贵。随着集成电路技术的发展，市场上出现了新一代的集成化单颗射频基站芯片，应用灵活且价格低廉。射频基站芯片取代老一代专用读写模块是大势所趋。采用射频基站芯片的第二代读写设备开始逐渐成为市场主流。

但是，直接用射频基站芯片实现与 **RFID** 卡的底层通信对一般应用开发人员来说还是比较困难的，需要花费大量时间和精力。在实际应用中，也是完全没有必要的。因此，应用系统开发人员需要 一种软硬件接口明确、使用方便的读写模块来完成与 **RFID** 卡的底层通信，这样他们就可不必了解对射频基站芯片的底层驱动，而只需对读写模块发送相关指令即实现对卡片的操作。为完成上述功能，读写模块内使用一个

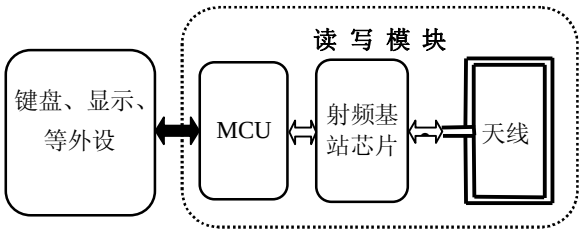
主控 MCU 来控制射频基站芯片完成所有底层的射频通信，对外仅需提供封装好的命令接口。读写模块可直接接收命令，根据命令完成相关对 RFID 卡的操作并返回操作结果。

由于实现了对 RFID 卡操作的模块化封装，这种读写模块很受市场欢迎。使用这类读写模块开发读写设备，用户可以大大减少开发工作量，然而代价是还需要另外一个 MCU 来控制读写模块、通信接口、键盘输入、显示输出及其他外部设备（见图 1-3A）。这样，从应用系统到卡片，数据的通信流程是：应用系统ßà读写设备主控 MCUßà读写模块 MCUßà读写模块射频基站芯片ßàRFID 卡。可以看出，读写设备内部使用了两个 MCU，不仅延长了通信的时间，降低了工作效率，更是对资源的浪费。

为解决上述问题，本文结合在线编程[9]思想设计一种 RFID 卡核心读写模块，不仅为用户提供射频通信函数接口，实现对 RFID 卡的操作，而且为用户提供可继续二次开发的软硬件平台。用户可在读写模块的基础上继续做扩展开发，而不需要另外再使用 MCU（见图 1-3B）。该读写模块为用户提供了一种效率更高、成本更低的读写设备解决方案。



A－基于市场上读写模块的读写设备结构



B－基于可供二次开发的读写模块的读写设备结构

图 1-3 基于读写模块的读写设备结构图

1.2.2 读写模块设计思路

(1) 对 RFID 卡的控制

即读写模块可实现与 RFID 卡的通信、完成与卡片数据的交换。这通过两个方面来实现：在硬件上，将与 RFID 卡通信的电路以及天线集成在读写模块内部，使用户

只需考虑与读写模块的通信接口的交互，而不需要再去专门开发射频通信的底层硬件；在软件上，将与 RFID 卡通信的底层驱动程序，以及在其基础上封装的高级函数驻留在读写模块的 MCU 中供用户调用。

(2) 可供用户二次开发

在读写模块的基础上进行二次卡发，可以通过如下途径实现。首先在硬件上将读写模块主控 MCU 的 I/O 口引出来，使得用户可以利用这些 I/O 口继续扩展设计，方便用户将读写模块集成到其应用系统中。其次是通过提供的对 MCU Flash存储区的在线擦除、写入功能，实现用户应用程序的开发。新型 MCU 内大多集成有 Flash 存储器[10]，Flash 存储器具有电可擦除、无需后备电源保护数据、可在线编程（在线编程 In-Circuit Program，指允许单片机内部运行的程序去改写 Flash 存储器的内容[9]）等特点。利用这个特点，开发芯片的监控程序并将其驻留在 Flash 的某个区域——一般称其为监控区。相对于监控区，剩余 Flash 区域为用户区。运行监控区的监控程序，就可实现对用户区的写入、擦除等操作。本文将读写模块的函数也封装在 Flash 的监控区中，供用户调用。这样用户就可将其应用程序写入 Flash 的用户区，在读写模块基础上继续开发其 RFID 卡应用产品。

1.3 本文工作与论文结构

1.3.1 本文工作

本文的主要工作安排如下：

(1) 设计 RFID 卡读写模块硬件。

- 芯片选型，确定选用的芯片；
- 了解芯片的外围电路，分析芯片间的接线方式，设计硬件原理图；
- 绘制 PCB 电路图，联系厂家制作电路板；
- 元器件的选型与采购等；
- 焊接、测试，完成硬件系统；

(2) 设计读写模块软件。

- 根据读写芯片的时序要求，编写访问读写芯片寄存器的程序；
- 根据射频通信协议，按照 RFID 卡与读写芯片的通信流程编写读写芯片的底层驱动；
- 为方便用户调用，提供卡片操作函数；

(3) 应用 RFID 卡读写模块，设计开发通用 RFID 卡读写设备。

- 设计、制作通用 RFID 卡读写设备的硬件；

- 设计并完成通用 RFID 卡读写设备的 MCU 方程序以及可供应用系统用户直接调用的 PC 方高级语言的函数接口。

(4) 升级读写模块的主控 MCU，将读写模块软件移植到新 MCU 中，开发带网络接口的考勤机。

- 设计、制作带网络接口的考勤机硬件系统；

- 在考勤机主控 MCU 中集成读写模块函数和网络通信接口。实现刷卡签到时，考勤机通过网络接口将读写模块提供的操作结果传给远程上位机的功能。

1.3.2 论文结构

全文共六章，各章的内容安排如下：

第一章介绍射频识别卡及其应用前景，提出了毕业设计的内容和论文结构。

第二章介绍射频识别技术理论、射频识别卡国际标准等相关应用基础。

第三章讲述射频识别卡读写模块的硬件设计方案，硬件系统组成，以及硬件的测试情况。

第四章阐述在硬件基础上的软件设计，包括接口编程，射频通信协议的实现等。

第五章介绍应用射频识别卡读写模块开发 通用读写设备和带网络接口的考勤机这两个实例。

第六章对本文的工作进行了总结，并提出了一些尚待研究的问题。

第二章 相关理论与技术

射频识别（RFID）技术作为先进的自动识别技术，被列为新世纪十大重要技术项目之一[11]。其实射频识别技术最早的应用可追溯到第二次世界大战中用于区分联军和纳粹飞机的“敌我辨识”系统[12]。不过直到上世纪八十年代，射频识别技术及产品才真正进入到商业应用领域。九十年代以后，射频识别技术标准化问题日趋得到重视，射频识别技术的理论得到进一步丰富和完善，射频识别产品开始广泛应用。

了解射频识别的相关理论与技术是开发射频识别卡读写模块的基础。本章将主要介绍射频通信实现的一些原理、方法与相关技术。

2.1 射频识别卡的基本原理与相关技术

2.1.1 射频识别系统的基本原理

最基本的 RFID 系统由三部分组成（见图 2-1）：

① 电子标签(Tag，或称射频标签、应答器)：由芯片及内置天线组成。芯片内保存有一定格式的电子数据，作为待识别物品的标识性信息，是射频识别系统真正的数据载体。内置天线用于和射频天线间进行通信。

② 阅读器：读取或读/写电子标签信息的设备，主要任务是控制射频模块向标签发射读取信号，并接收标签的应答，对标签的对象标识信息进行解码，将对象标识信息连带标签上其它相关信息传输到主机以供处理。

③ 天线：标签与阅读器之间传输数据的发射、接收装置。

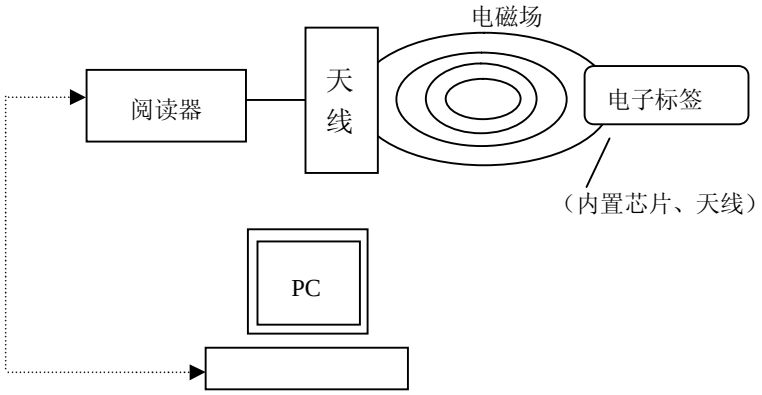


图 2-1 RFID 系统基本模型图

RFID 的工作原理是：电子标签进入天线磁场后，如果接收到阅读器发出的特殊

射频信号，就能凭借感应电流所获得的能量发送出存储在芯片中的产品信息（无源标签），或者主动发送某一频率的信号（有源标签），阅读器读取信息并解码后，送至中央信息系统进行有关数据处理[13]。

发生在阅读器和电子标签之间的射频信号的耦合类型有两种：

(Ⅰ) 电感耦合[1]。变压器模型，通过空间高频交变磁场实现耦合，依据的是电磁感应定律，如图 **2-2** 所示。

(Ⅱ) 电磁反向散射耦合[1]。雷达原理模型，发射出去的电磁波，碰到目标后反射，同时携带回目标信息，依据的是电磁波的空间传播规律，如图 **2-3** 所示。

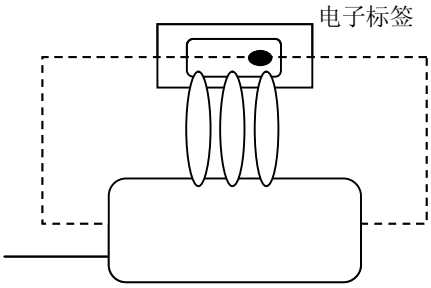


图 2-2 电感耦合

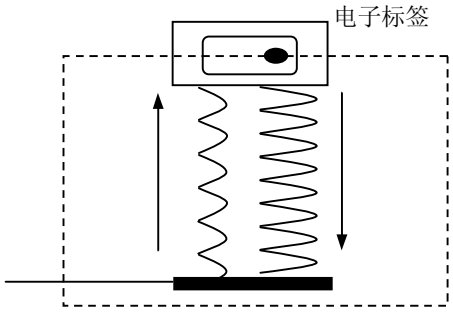


图 2-3 电磁反向散射耦合

2.1.2 射频识别系统的分类

射频识别系统中标签与读写器之间的作用距离是射频识别系统应用中的一个重要指标。通常，这种作用距离也就是指标签与阅读器之间能可靠交换数据的距离。根据作用距离，标签天线和读写器之间的耦合可以分为三类：密耦合系统、遥耦合系统和远距离系统[1] [6]。

(1) 密耦合系统

密耦合系统的典型作用距离范围是 **0～1cm**。实际应用中，必须把标签插入阅读器中或者放置到阅读器的天线表面。密耦合系统的标签与阅读器之间是电感耦合。其工作频率一般在 **30MHz** 以下。密耦合系统适合于安全要求较高，但不要求作用距离的应用系统，如电子门锁等。

(2) 遥耦合系统

遥耦合系统的典型作用距离可以达到 **1m**。遥耦合系统又可以细分为近耦合系统和疏耦合系统，前者的典型作用距离为 **15cm**，后者为 **1m**。所有遥耦合系统在阅读器和标签之间都是电感耦合。遥耦合系统的典型工作频率为 **13.56MHz,**也有其他频率，如 **6.75MHz**，**27.125MHz** 或者 **135kHz** 以下。

(3) 远距离系统

远距离系统的典型作用距离是 1～10m，个别系统也有更远的作用距离。所有的远距离系统的阅读器和标签之间都是电磁反向散射耦合。远距离系统都是在微波范围内用电磁波工作的，发送频率通常为 2.45GHz,也有系统使用的频率为 5.8GHz 和 24.125GHz。

2.1.3 能量传送

RFID 卡卡内无电源，供芯片运行所需要的全部能量必须要由阅读器传送。阅读器和 RFID 卡之间能量的传递基于耦合变压器原理[5]，参见图 2-4 所示。阅读器终端天线产生强大的高频磁场以便传送能量，最常用的频率有 125kHz 和 13.56MHz。如果一个 RFID 卡被放到阅读器天线附近，阅读器天线的磁场的一部分就会穿过卡的线圈，在卡的线圈里感生电压 U_i。这个电压被整流后就用来对芯片供电。由于阅读器天线与卡片线圈的耦合非常弱，因此需要使天线线圈里的电流量增大，以便达到必要的磁场强度，这通过给线圈 L_T 并联一个电容 C_T 来实现。电容的值要经过选择，以使其和天线的并联谐振频率与所传递的信号频率相匹配。

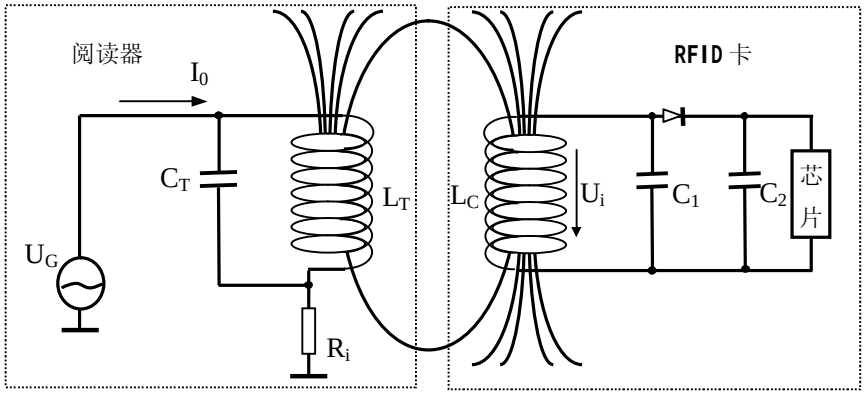


图 2-4 用电感耦合方式给RFID卡供电

2.1.4 数据传送

（1）信号调制

从阅读器到 RFID 卡的数据传送，可以使用所有已知的数字调制技术。常用的技术有振幅键控(Amplitude Shift Keying,ASK)、频移键控(Frequency Shift Keying,FSK)和相移键控(Phase Shift Keying,PSK)[1]，是对电磁波的三个参数——功率、频率和相位分别进行调制的方法。由于容易解调，ASK 和 PSK 更为常用。

从 RFID 卡到阅读器，使用的是幅度调制方式 ASK，用数据信号来对卡里的负载进行数字调节（负载调谐）[14]。如果把一个调谐为终端谐振频率的 RFID 卡放到阅读器附近的磁场中，它就从磁场中汲取能量。这将引起阅读器感应线圈里的电流 I_0 增加，可以通过跨内部电阻 R_i 所增加的压降把它检测出来（参见图 2-5）。RFID 卡可

以通过改变其线圈的负载（在电路中把负载电阻 R_2 接入和断开）来改变终端的电压 U_0（振幅调制）。如果电阻 R_2 的开关是由数据信号控制，那么数据在阅读器里就可以被检测和计算出来。

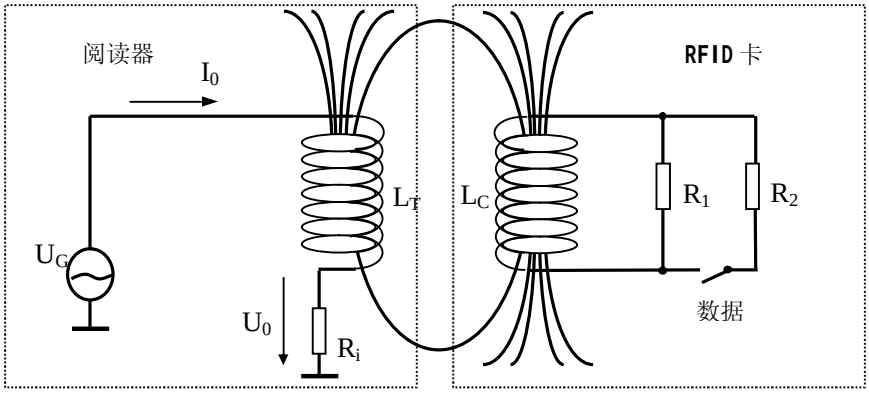


图 2-5 RFID 卡传送数据的电路原理

由于阅读器和卡的线圈间的耦合度较低，所以在阅读器里由负载调制所感应的电压变化也是非常小的。在实践中，可用的信号幅度只有几毫伏，被阅读器传输的强大信号（大约 80dB）所覆盖，只能用精密的电路才能检测出来。因此，一般采用一个频率为 f_H 的辅助载波频率，在阅读器上要接收的数据信号就出现在频率为 $f_T \pm f_H$ 的两个边带上，如图 2-6 所示。使用带通滤波器将这些数据信号从非常强的阅读器发送信号中滤出来并放大。这样，这些信号就很容易被解调。

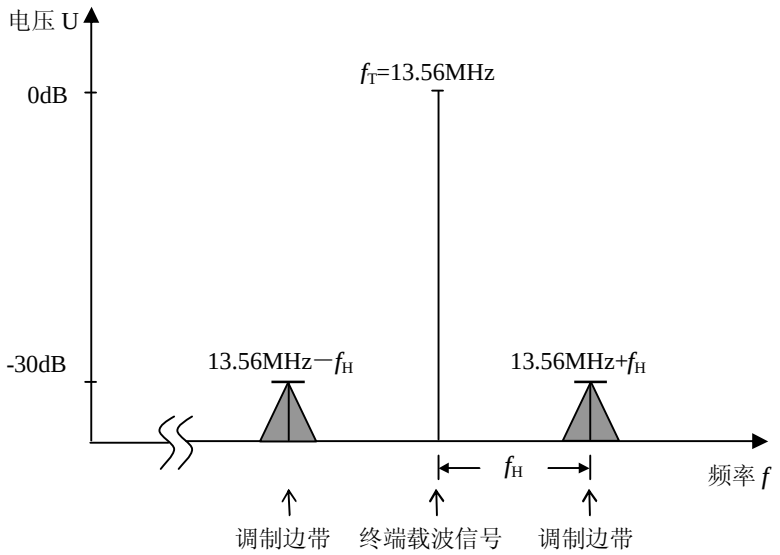


图 2-6 用辅助载频进行负载调制产生两个边带（从阅读器的输出频率中把辅助载波 f_H 值分出，实际的信息包含在两个辅助载频边带里）

（2）数据编码

数据编码，一方面便于数据传输，另一方面可以对传输的数据进行加密。射频识

别系统通常使用的编码方法有：NRZ 编码(Non-Return-to-Zero)、曼彻斯特编码(Manchester)、单极性归零制编码、米勒编码(Miller)、修正的米勒编码、差动双向编码、差动编码[6]。

2.1.5 数据完整性

（1）校验

使用射频识别技术传输数据时很容易遇上干扰，使传输的数据发生改变从而导致传输错误。通常使用数据检错与纠错算法来识别传输错误并启动校正措施[1]。

在射频识别的通信过程中最常用的校验方法是奇偶校验、循环冗余校验(CRC)还有纵向冗余校验。

（2）多标签同时识别与系统防冲突

射频识别系统的一个优点是同时识别多个标签。射频识别系统工作时，在一个阅读器的作用范围内，可能同时有多个标签。这时，系统中有两种最基本的通信：从阅读器到标签的通信和从标签到阅读器的通信。

从阅读器到标签的通信，阅读器发送的数据流同时被所有的标签接收，这种通信方式类似于无线电广播方式，被称作“无线电广播”。从标签到阅读器的通信，是多个标签的数据同时传送给阅读器，这种通信方式称作多路存取。无线电通信系统中多路存取方法基本上有以下几种：空分多路法(SCDMA)、时分多路法(TDMA)、频分多路法(FDMA)、码分多路法(CDMA)[1]。在射频识别系统中，一般采用的是 TDMA。

时分多路（TDMA）法是把整个可供使用的通道容量按时间分配给多个用户的技术。可以通过阅读器控制实现，即所有的标签同时由阅读器进行观察和控制。通过一种特定的算法，在阅读器工作范围内的标签中选中一个，然后完成阅读器和标签之间的通信（如读出、写入数据等）。在同一时间只能建立一个通信关系，阅读器在解除与原来的标签的通信关系后，继续与下一个标签通信。上述实现 TDMA 的特定算法也被称作防冲突算法[1]。

2.1.6 数据安全性

在与安全有关的射频识别系统的应用中，例如出入系统或支付系统，必须采取安全措施来防止遭受恶意攻击。射频识别系统常用的安全手段有：

（1）相互对称的鉴别

当某个标签进入阅读器的工作范围时，需要断定参与通信的双方是否同属一个应用系统。从阅读器看，需要防止伪造的数据。从应答器看，同样需要防止其存储数据未被认可的读取或写入。阅读器和标签使用相同的密钥 K，采用相互对称的鉴别方式

[15]，阅读器和标签在通信中互相检验对方的密码。

（2）加密的数据传输

通信时的数据在传输时可能会受到非法的攻击。射频识别系统在阅读器与应答器之间传输数据时，使用密钥和加密算法将传输数据（明文）变换为秘密数据（密文），可以有效防止攻击[15]。若不了解加密算法和密钥 K，攻击者无从解释其截获的密文。

2.2 RFID 卡的国际标准

2.2.1 RFID 卡的国际标准

标准是 IC 卡设计制造与应用的支撑点。自诞生以来，RFID 卡的推广与使用与标准的制定密不可分。根据不同的作用距离，国际标准化组织 ISO/IEC 已编制了三种不同的 RFID 卡国际标准[7]（见表 2-1）。

表 2-1 RFID 卡的国际标准

标准	卡类型	读写器	作用距离（约）
ISO/IEC 10536	密耦合 IC 卡(Close Coupled ICC，CICC)	CCD	0～1cm
ISO/IEC 14443	近耦合 IC 卡(Proximity ICC，PICC)	PCD	0～10cm
ISO/IEC 15693	疏耦合 IC 卡(Vicinity ICC，VICC)	VCD	0～1m

注：ICC（Integrated Circuit Card）为集成电路卡，CD（Coupling Device）指读写设备。

密耦合 IC 卡的生产成本高且与接触式 IC 卡相比优点很少，在市场上几乎没什么应用。目前市场上应用较多的是载波频率为 13.56MHz，工作距离在 2.5～10cm 的近耦合 IC 卡，其国际标准为 ISO/IEC 14443。下面简单介绍一下国际标准 ISO/IEC 14443。

2.2.2 近耦合 IC 卡国际标准 ISO/IEC 14443

识别卡－近耦合集成电路卡的国际标准是 ISO/IEC 14443，由以下四个部分组成：第 1 部分——物理特性；第 2 部分——射频能量和信号接口；第 3 部分——初始化和防冲突；第 4 部分——传输协议[16]。

近耦合 IC 卡的物理特性及尺寸与 ISO/IEC 7810 中的规定相符，为 85.72mm×54.03mm×0.76mm±容差。与磁卡、接触型 IC 卡标准尺寸完全一致，为兼容接触型 IC 卡和磁卡提供了有效途径和方案，使得非接触型的双界面卡、多功能组合卡的推出成为可能。

卡的能量是由阅读器的射频（RF）电磁场提供的。RF 场的频率是 13.56MHz±7kHz，磁场强度在 1.5A/m 和 7.5A/m 之间。

阅读器（PCD）和近耦合 IC 卡（PICC）之间的数据传输有两种完全不同的方法，ISO/IEC 14443 分别将其定义为 A 型（TypeA）和 B 型(TypeB)。一张 PICC 只需两种通信方法之一来支持，PCD 可以在两种通信方法间周期的转换来支持所有的卡。但在 PCD 和 PICC 间通信的过程中不允许在两种方法间转换。

Type A 和 Type B 的主要区别在于载波的调制深度及二进制数的编码方式。A 型卡在阅读器向卡传送信号时，是通过 13.56MHz 的射频载波传送信号，采用改进的 Miller 编码方式，通过 100%ASK 传送；当卡向阅读器传送信号时，使用振幅键控（ASK）调制 847kHz 的副载波传送，编码采用曼彻斯特编码。而 B 型卡在从阅读器向卡传送信号时，也是通过 13.56MHz 的射频载波信号，但采用的是 NRZ 编码方式，通过 10％ ASK 传送；在卡向阅读器传送信号时，是通过对 NRZ 编码的数据流的 847kHz 副载波采用相位键控调制（BPSK）。如表 2-2 所示。Type A技术的主要厂商代表是 Philips 公司，Type B 技术的主要代表是 Freescale 公司（原 Motorola 半导体）[17]。

表 2-2 Type A 与 Type B 的比较

		A 型	B 型
PCD —> PICC	调制	ASK 100%	ASK 10%
	位编码	改进的 Miller 编码	NRZ 编码
	位速率	106Kb/s	106Kb/s
	同步	位级同步(帧起始，帧结束标记)	每个字节有 1 个起始位和 1 个结束位
PICC —> PCD	调制	用 ASK 调制 847kHz 的负载调制的副载波	用 BPSK 调制 847kHz 的负载调制的副载波
	位编码	曼彻斯特编码	NRZ 编码
	位速率	106Kb/s	106Kb/s
	同步	1 位“帧同步”(帧起始，帧结束标记)	每个字节有 1 个起始位和 1 个结束位

相应的，A 型和 B 型卡在和 PCD 通信时采用的协议和防冲突方法均不相同，标准的第三部分、第四部分对其分别做了规定。

2.3 RFID 卡——Mifare

Philips 是世界上最早研制 RFID 卡的公司，其 Mifare 技术已经被制定为 ISO / IEC14443 TYPE A 国际标准。使用 Mifare 芯片的 RFID 卡占世界范围同类智能卡销量的 60%以上，在我国市场也占据着绝对优势[18]。

下面介绍一下 Mifare standard 卡：MF 1 IC S50（以下简称 Mifare 1 或 MF1 卡）。

2.3.1 Mifare 1 卡的特性

Mifare 1卡片除了微型芯片 **IC** 及一个高效率天线外，无任何其它元件。卡片电路不用任何电池供电，工作时的能量由读写器天线发送频率为 **13.56MHz** 无线电载波信号，以非接触方式耦合到卡片天线上而产生电能，通常可达 **2V** 以上。标准操作距离高达 **10cm**，卡与读写器之间的通信速率高达 **106Kbit/s**。芯片设计有增／减值的专项数学运算电路，非常适合公共交通、地铁车站等行业的检票／收费系统，或充值钱包等多项应用，其典型交易时间最长不超过 **100ms**[19]。

芯片内建 **8K Bits** 的 **E^2PROM** 存储器。其空间被划分为可由用户单独使用的 **16** 个扇区。数据的擦写能力超过 **10** 万次以上，数据保存期大于 **10** 年，抗静电保护能力达 **2KV**。

Mifare 1 卡的芯片在制造时具有全球唯一的序列号。具有先进的数据通信加密和双向密码验证功能。具有防冲突功能，可以在同一时间处理重叠在读写器天线有效工作距离内的多张卡片。

2.3.2 Mifare 1 芯片逻辑结构

Mifare 1芯片内部结构较为复杂，可分为射频接口、数字处理单元、**E^2PROM**（**1K** 字节）三部分，逻辑框图见图 **2-7**。

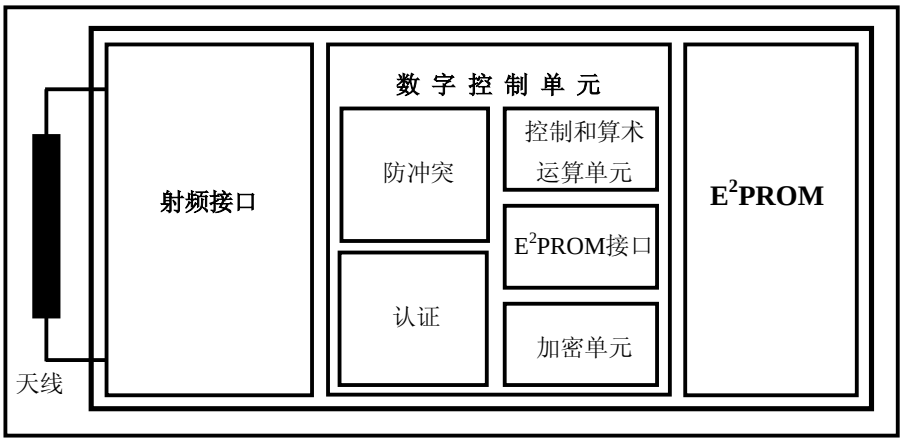


图 2-7 MF1 卡逻辑结构图

① 射频接口：在 **RF** 射频接口电路中，包括有波形转换模块。它可接收读写器上的 **13.56MHZ** 的无线电调制频率，一方面送调制/解调模块，另一方面进行波形转换，然后对其整流滤波，接着对电压进行稳压等进一步的处理，最终输出供给卡片上的电路工作。

② 防冲突模块：如果有多张 **Mifare 1**卡片处在读写器的天线的工作范围之内时，防冲突模块的防冲突功能将被启动工作：根据卡片的序列号来选定一张卡片。被选中

的卡片将直接与读写器进行数据交换，未被选择的卡片处于等待状态，准备与读写器进行通信。

③ 认证模块：在选中一张卡片后，任何对卡片上存储区的操作都必须要经过认证过程，只有经过密码校验才可对数据块进行访问。**Mifare 1** 卡片上有 **16** 个扇区，每个扇区都可分别设置各自的密码，互不干涉。因此每个扇区可独立地应用于一个应用场合。整个卡片可以设计成“一卡通”形式来应用。

④ 控制和算术运算单元：这一单元是整个卡片的控制中心，是卡片的“大脑”。它主要对整个卡片的各个单位进行微操作控制，协调卡片的各个步骤；同时还对各种收／发的数据进行算术运算处理、**CRC** 运算处理等等。

⑤ **E^2PROM** 接口：连接到 **E^2PROM**。

⑥ 加密单元：**Mifare** 的 **CRYPTO1** 数据流加密算法将保证卡片与读写器通信时的数据安全。

⑦ **E^2PROM**：**1K** 字节，分 **16** 个扇区。每扇区 **4** 个块，每块 **16** 字节。

2.3.3 存储器组织结构

Mifare 1卡片的存储容量为 **8192×1** 位字长(即 **1K X 8**位字长），采用 **E^2PROM** 作为存储介质。整个结构划分为 **16** 个扇区，编为扇区 **0**～**15**。每个扇区有 **4** 个块（**Block**）,分别为块 **0**,块 **1**,块 **2** 和块 **3**。每个块有 **16** 个字节。一个扇区共有 **16 Byte X 4 = 64 Byte**。如图 **2-8** 所示。

扇区	块	描述
15	63	第 15 扇区尾块
	62	数据块
	61	数据块
	60	数据块
14	59	第 14 扇区尾块
	58	数据块
	57	数据块
	56	数据块
1	7	第 1 扇区尾块
	6	数据块
	5	数据块
	4	数据块
0	3	第 0 扇区尾块
	2	数据块
	1	数据块
	0	厂商标志块

图 2-8 MF1 卡存储区的组织示意图

每个扇区的块 **3**（即第四块）也称作尾块， 包含了该扇区的密码 **A(6** 个字节**)**、存取控制**(4** 个字节**)**、密码 **B(6** 个字节**)**。其余三个块是一般的数据块。

扇区 **0** 的块 **0** 是特殊的块，包含了厂商代码信息，在生产卡片时写入，不可改写。其中：第 **0**～**4** 个字节为卡片的序列号，第 **5** 个字节为序列号的校验码；第 **6** 个字节为卡片的容量“**SIZE**”字节；第 **7**，**8** 个字节为卡片的类型号字节，即 **Tagtype** 字节；其他字节由厂商另加定义。

2.3.4 对 Mifare 1 卡的读写控制

每个扇区的尾块（**16** 字节）包含了该扇区的两个密码信息以及对本扇区中各块的读写权限信息，是扇区的控制块。控制块使用两个密码，为用户提供多重控制方式。

例如，用户可以用一个密码控制对数据块的读操作，用另一个密码控制对数据块的写操作。尾块的组成结构(参见图 2-9)中，前六个字节为 A 密码（KeyA），KeyA 是永远不能读出的，但在满足一定条件下，可被改写；后六个字节为 B 密码（KeyB），KeyB 当密钥使用时，也是不可读的，但 KeyB 的六个字节可用来存储数据，此时 KeyB 可读；中间四个字节为权限位(Access Bits),存放本扇区的四个数据块的访问条件。

字节	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	密码 A						权限位				密码 B					
	位：7		6		5		4		3		2		1		0	
6	$\overline{C2}_3$		$\overline{C2}$		$\overline{C2}_1$		$\overline{C2}_0$		$\overline{C1}_3$		$\overline{C1}_2$		$\overline{C1}_1$		$\overline{C1}_0$	
7	$C1_3$		$C1_2$		$C1_1$		$C1_0$		$\overline{C3}_3$		$\overline{C3}_2$		$\overline{C3}_1$		$\overline{C3}_0$	
8	$C3_3$		$C3_2$		$C3_1$		$C3_0$		$C2_3$		$C2_2$		$C2_1$		$C2_0$	
9																

图 2-9 尾块组成及访问权限字节结构

权限代码			访问权限						说明
			密码 A		权限字节		密码 B		
$C1_3$	$C2_3$	$C3_3$	读	写	读	写	读	写	—
0	0	0	N	A	A	N	A	A	密码 B 可读
0	1	0	N	N	A	N	A	N	密码 B 可读
1	0	0	N	B	A/B	N	N	B	—
1	1	0	N	N	A/B	N	N	N	—
0	0	1	N	A	A	A	A	A	密码 B 可读
0	1	1	N	B	A/B	B	N	B	—
1	0	1	N	N	A/B	B	N	N	—
1	1	1	N	N	A/B	N	N	N	—

图 2-10 尾块的权限代码与访问权限
（注：N 表示不能，A 表示 KeyA，B 表示 KeyB，A/B 表示 KeyA 或者 KeyB）

图 2-9 中，用 C1,C2,C3 三个数据位表达各块的具体访问权限，下标 0、1、2、3 分别表示在扇区内的块号。“$C1_3$ $C2_3$ $C3_3$”即为扇区第 3 块（尾块）的访问权限。为了可靠，访问条件的每一位都同时用原码和反码存储，存储了两遍。尾块的读写权限的意义见图 2-10。

在空卡状态下每个扇区的尾块数据(16 进制)为：“0x 000000000000 FF078069 FFFFFFFFFFFF”。空卡时的密码 A 和密码 B 均为“0xFFFFFF”，由于 A 密码不可读，读出的数据显示为“0x000000”。在空卡默认读写权限下可以利用密码 A 对所有块进行读写操作，以及更改各块的读写权限。但不可以利用密码 B 进行读写操作（此时 B 密码可读）。

权限位为：“0xFF078069”，由图 2-9 ，有：

C13=0 C12=0 C11=0 C10=0

C23=0 C22=0 C21=0 C20=0

C33=1 C32=0 C31=0 C30=0

“$C1_3$ $C2_3$ $C3_3$”=001，对应图 2-13 的第 5 行，表示 A 密码不可读，可用 A 密码改写（即通过 A 密码校验后，可改写 A 密码），权限字节及 B 密码的读写权限均可用 A 密码读写。

由图 2-10 可知尾块的下列属性：密码 A 永远不可读，因此一旦设定就必须记住，不过在 000、100、001、011 几种情况下可以改写；访问权限字节仅在 001、011、101 三种状态下可写；密码 B 在 000、100、001、011 四种状态下可写，在 000、010、001 三种状态下可读（此时 B 密码的六个字节用于存储数据，不再作为密钥）。

权限代码			访问权限				应用
$C1_i$	$C2_i$	$C3_i$	读	写	增值	减值	
0	0	0	A/B	A/B	A/B	A/B	空卡默认状态
0	1	0	A/B	N	N	N	读写块
1	0	0	A/B	B	N	N	读写块
1	1	0	A/B	B	B	A/B	数值块
0	0	1	A/B	N	N	A/B	数值块
0	1	1	B	B	N	N	读写块
1	0	1	B	N	N	N	读写块
1	1	1	N	N	N	N	读写块

图 2-11 数据块(i＝0、1、2)的权限代码与访问权限

数据块的读写权限，如图 2-11 所示。对数据块的增值、减值操作，仅在状态“110”和“001”时可进行。而第 0 块（厂商数据块）虽然也属数据块，但是它不受权限字节影响，永远只读。在空卡情况下，数据块的读写权限代码：C1i、C2i、C3i=0 0 0（对应于图中第 1 行），A 密码和 B 密码读写均为 A/B，表示可用密码 A 或者是密码 B 对各数据块进行读写，但实际上由于在空卡默认状态下 B 密码是可读的，所以不可用 B 密码读写数据。

第三章 读写模块硬件设计

为使上层应用系统对 RFID 卡的访问尽可能透明，将 RFID 卡的射频读写技术封装起来，构成 RFID 卡读写模块（简称读写模块）。读写模块的设计，可以分成硬件和软件两个部分。硬件系统是读写模块的核心，也是软件设计的基础。

本章将详细分析读写模块的硬件设计，包括硬件系统的组成、芯片的选型、硬件的连接，最后还讨论了对硬件系统的测试。

3.1 硬件系统组成

为了增强读写模块的通用性和可扩展性，在硬件设计时遵循模块化的设计思想。整个读写模块由三大部分组成：

（1） 主控 MCU；

（2） 射频读写芯片；

（3） 天线及匹配电路。

第一部分，主控 MCU，主要提供对射频读写芯片的控制操作。这种控制操作体现在两个方面：

①对射频读写芯片的电源控制。通过对射频读写部分的独立电源控制，用户可以在 MCU 中根据自己的需要选择或关闭射频读写功能。当应用系统有低功耗要求，不需要射频读写芯片一直工作时，这种控制方式是必不可少的。而且，通过 MCU 的供电控制，可以用软件方式实现射频读写芯片的上电复位。

②MCU 通过数据线、地址线、控制线等并行控制接口与射频读写芯片连接，控制射频读写芯片的正常工作，实现与 RFID 卡的通信。

另外，主控 MCU 通过串行通信接口与 PC 方进行通信，方便用户将开发的应用程序载入到 MCU 中。同时，将主控 MCU 的剩余 I/O 口及中断引脚引出，可供用户扩展使用。

第二部分，射频读写芯片，它负责接收主控 MCU 的控制信息并完成与 RFID 卡的通信操作。为了正常工作，射频读写芯片须选用合适的并行接口与 MCU 连接。而为了发送、接收稳定的高频信号，射频读写芯片要通过高频滤波电路与天线部分连接。

第三部分，天线部分，包括线圈及匹配电路，这是读写模块实现射频通信必不可少的一部分。读写模块要依靠天线产生的磁通量为 RFID 卡提供电源、在读写模块与

RFID 卡之间传送信息。为使天线正常工作，天线线圈要通过无源的匹配电路连接射频读写芯片的天线引脚。

综上所述，RFID 卡读写模块的硬件系统组成的框图参见图 3-1。

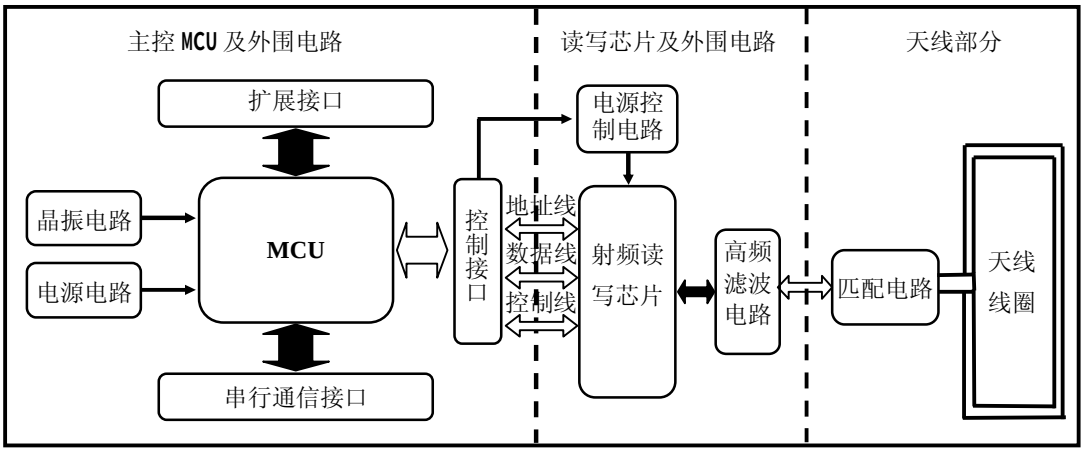


图 3-1 读写模块硬件系统组成

3.2 芯片选型

在系统组成确定后，就需要考虑具体的硬件选择。如前所述，读写模块的硬件核心是主控 MCU 以及射频读写芯片。

3.2.1 嵌入式微控制器 MCU

在嵌入式应用开发过程中选择 MCU 时，需要综合考虑以下几个方面：

（1）MCU 对应用系统的适用性。从对应用系统的适用性出发，要考虑 MCU 是否含有所需的 I/O 端口数目——如果 MCU 的 I/O 端口数太少，那么就不可能满足有关的功能；而 I/O 端口数太多，就会造成 MCU 资源的浪费。另外还有 MCU 是否含有所需的外围端口部件、CPU 是否有合适的吞吐量、极限性能是否满足要求等。

（2）MCU 的可购买性。在 MCU 能够适合应用系统时，还要考虑这种型号的 MCU 的可购买性：是否可直接购买到、是否有足够的供应量、是否仍然在生产之中、是否在改进之中。在应用中选择有改进版本推出的 MCU，将有利于产品的升级换代。

（3）MCU 的可开发性。所选择的 MCU 是否有足够的开发手段，直接影响到 MCU 能否顺利开发，并且较快的应用于被控对象中。开发 MCU 需要考虑：编译软件、程序写入工具、调试工具、技术支持、语言体系与熟悉程度等因素。

在研究生阶段的学习过程中，作者对 Freescale 公司的 08 系列 MCU 做了大量的研究，熟悉其指令系统、编程结构和工作性能，积累了一定的实践开发经验。而且实

验室有比较完备的开发 08 系列 MCU 所需要的编译软件、程序写入工具等条件。因此，作者选用 Freescale 的 08 系列 MCU 作为读写模块的主控 MCU。

另外，由于 08 系列 MCU 指令系统相同，可以根据实际需要选用不同型号的 MCU 担任读写模块的主控 MCU，实现其硬件的通用性。可以在编写程序时，将与硬件相关的内容集成在一个头文件中，做到源程序仅与头文件相关。当在应用中，需要改换 MCU 时，读写模块的软件只需更改头文件部分就可以移植到新的 MCU 中，具有较强的可移植性。

Freescale 的 08 系列 MCU 型号丰富，有一百种之多。不同型号的 MCU 资源各不相同，即使是同一种型号也有多种封装形式，不同封装形式的 I/O 口数目不同。表 3-1 表现了 08 系列 MCU 的资源差异情况。

表 3-1 08 系列 MCU 的资源差异情况表

产品型号	ROM (字节)	RAM (字节)	EEPROM (字节)	FLASH (字节)	I/O 数	Serial	A/D	最高的总线频率 (MHZ)
MC68HC908GP32	307	512	—	32K	31	SCI SPI	8 通道 8 位	8.0
MC9S08GB60	-	4K	-	60K	56	I^2C SCI SPI	8 通道 10 位	20.0
MC68HC08AB16A	16K	512	512	-	51	SCI SPI 8	通道 10 位	8.0
MC68HC908EY16	-	512	-	16K	24	ESCI SPI	8 通道 10 位	8.0
MC68HLC908QT2	-	128	-	1.5K	6	-	4 通道 8 位	8.0

经过比较，作者最终决定用 MC68HC908GP32 作为读写模块的主控 MCU。GP32 有 3 种不同的封装[20]，其中 SDIP 42 封装有 31 个通用 I/O 脚，不仅完全满足与读写芯片的并行接口的需要，还可以满足扩展控制键盘、显示、蜂鸣器、指示灯等其他外部设备。GP32 的处理器 CPU 08 内部总线频率高达 8HMZ，最小的指令执行时间为 125ns，最长的 16 位/8 位除法指令周期也只有 875ns，其运算能力完全可以满足控制读写设备的需要。GP32 有 32K 的 Flash 存储区，足够用作程序存储空间。另外，GP32 还有串行通信模块、计算机操作正常（COP）模块等功能模块。

GP32 芯片应用广泛，购买方便。而且，Freescale 的 08 系列 MCU 有着旺盛的生命力，不断有功能更强大的新型号推出。GP32 的指令系统和内部模块与 08 系列的其他型号兼容，在 GP32 芯片上开发的应用系统可以平滑的转移到功能更强大的其它型号上去，有利于读写模块的功能扩展和改进。

3.2.2 射频读写芯片

不同类型的 RFID 卡，由于采用的通信协议不同，相应的射频读写芯片也不同。目前在中国的市场上，RFID 卡主要的厂商有：中国的华虹、复旦微电子、以及荷兰

Philips、瑞士 LEGIC、法国意法半导体（ST）、日本索尼[21]等，其中基于 Philips 公司 Mifare 芯片的产品在市场上占有绝对的优势。鉴于国内市场上使用 Mifare 芯片的 RFID 卡应用广泛，本文采用 Philips 公司生产的射频处理基站芯片开发读写模块。

Philips 公司现已推出的 13.56MHz 非接触式通信、支持 ISO/IEC 14443A 协议的 Mifare 读写器芯片的列表如 3-2 所示：

表 3-2 Mifare 读写器芯片简表

名称	通信速率	电源	支持协议	接口类型	最大读写距离
MF RC531	可达 848Kbps	5V	ISO/IEC 14443A&B	并行、SPI	100mm
MF RC530	可达 848Kbps	5V、3.3V	ISO/IEC 14443A	并行、SPI	100mm
MF RC500	106Kbps	5V	ISO/IEC 14443A	并行	100mm
MF CM500	106 Kbps	5V	ISO/IEC 14443A	并行	100mm
MF CM200	106 Kbps	5V	ISO/IEC 14443A	并行	25mm

MF CM200 与 MF CM500 是第一代 Mifare 读写器模块，现已停产。Philips 新推出的集成化单颗射频基站芯片 RC 系列是 CM 模块系列的替代产品，且性能更稳定、功耗更低、应用更灵活、价格更低廉。

MF RC530 由于可以支持 3.3V 电源供电，一般多用于手持设备；MF RC531 则主要应用于可以支持 TypeB 型卡的场合。支持 ISO/IEC 14443A 协议的射频读写芯片中，MF RC500 的性价比最高，市场应用最为广泛，购买也最方便。本文选用 MF RC500 射频读写芯片来进行读写模块的设计。

3.3 微控制器 MC68HC908GP32

3.3.1 GP32 特性

GP32 芯片的主要特性如下：

(1) 32KB 的片内 FLASH 存储器，具有在线编程能力和保密功能；

(2) 512B 片内 RAM；

(3) 增加型串行通信口 SCI 和串行外围接口 SPI；

(4) 两个 16 位双通道定时器接口模块；

(5) 系统保护特性：计算机工作正常（COP）复位，低电压检测复位，可选为 3V 或 5V 操作，非法指令码检测复位，非法地址检测复位；

(6) 时钟发生器模块，用 32KHZ 晶振的锁相环电路，可产生最高达 8MHZ 的内部总线工作频率；

(7)具有三种封装形式，分别为 40 脚、42 脚、44 脚，最多可有 33 根通用 I/O

脚；

(8) 8 位键盘唤醒口。

3.3.2 GP32 主要功能模块

GP32 芯片的功能结构框图参见附录 A.1，其主要功能模块包括：

(1) CPU 08：GP32 的处理器，内部总线频率高达 8HMZ，最小的指令执行时间为 125ns，最长的 16 位/8 位除法指令周期也只有 875ns。指令系统功能强，寻址方式多，编程方便。

(2) 系统操作正常监视模块(Computer Operating properly COP)：俗称看门狗电路。在微控制器工作不正常时，产生一个复位信号。该模块有一个计数器，COP 允许后，软件必须周期性向$FFFF 写入任意值，以清除 COP 计数器。若系统由于某种原因使软件工作不正常，COP 计数器得不到清零，那么溢出时便产生复位信号，以防程序进入不可预料的操作。

(3) 异步串行通信接口模块(Serial Communication Interface，SCI)：实现诸如 RS-232、RS-485 等能使用异步串行通信规程的通信，主要用于和其他计算机的数据传输。

(4) 存储器：GP32 可寻址 64KB 空间，包括有：

① 32KB 的闪速存储器 Flash

② 32256B 的用户空间

③ 512B 的随机存储器 RAM

④ 36B 用户定义的矢量区(属于 Flash 存储器)

⑤ 307B 的监控 ROM

3.4 射频读写芯片 MF RC500

MF RC500是 ISO/IEC 14443 标准下低成本、高集成、高性能的 Mifare 非接触式读写芯片，基于 13.56MHz 的非接触通讯模式。MF RC500 内部有高集成的调制解调模块，内部发射器可直接驱动基于 13.56MHz 的非接触式天线，最大距离可达 10cm。主要应用于各种基于 ISO/IEC 14443A 标准的非接触式通信的应用场合，如公共交通终端、手持终端、计量设备、非接触式公用电话等。

3.4.1 MF RC500 的功能结构

MF RC500 的功能结构如图 3-2[22]。

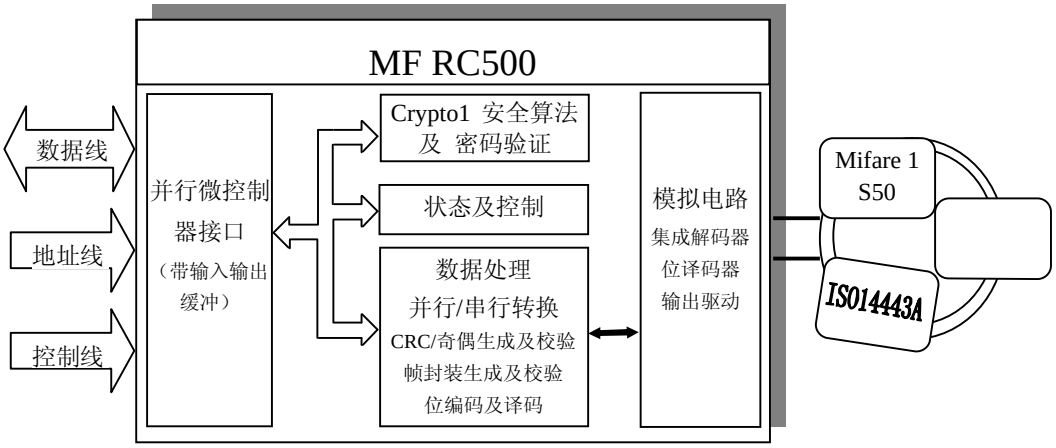


图 3-2 MF RC500功能框图

MF RC500支持ISO14443A所有的层，模拟电路部分内含射频发送器和接收器。发送器不需要增加有源电路就可以直接驱动工作距离最高达10cm的天线，接收器对来自符合ISO 14443A协议的卡的信号进行解调、译码。MF RC500的8位并行微控制器接口可自动检测连接的接口类型，它包括一个双向FIFO缓冲区和一个可设置的中断输出。方便的并行接口可与各种8位微处理器直接连接，给读写卡器/终端的设计提供了极大的灵活性。数据处理部分则主要进行ISO14443A帧的封装和错误检测（支持CRC校验和奇偶校验）。通过状态和控制部分可以对芯片进行配置，以适应环境并使芯片性能调节到最佳状态。此外，它还支持快速CRYPTO1加密算法，用于验证Mifare系列产品。

3.4.2 MF RC500的引脚说明

MF RC500的引脚图[22]参见图3-3。MF RC500共有32个引脚，可以分为以下几类：

① 电源类引脚

为使EMC特性和信号解耦方面达到最佳性能，器件使用了3个独立的电源。分别是：

TVDD，TVSS（6脚，8脚）：天线驱动部分的单独电源。

AVDD，AVSS（26脚，28脚）：模拟部分的单独电源。

DVDD，DVSS（25脚，12脚）：数字部分的单独电源。

另外，还有天线部分的内部参考电压：VMID（30脚）。

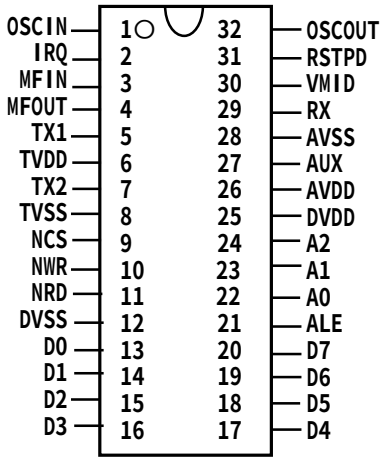


图 3-3 MFRC500引脚图

②天线引脚

TX1，TX2（**5** 脚，**7** 脚）：发送器引脚。**MF RC500**通过 **TX1** 和 **TX2** 发送 **13.56MHz** 的能量载波。

RX（**29** 脚）：接收器引脚。接收从天线耦合得到的来自卡片的 **13.56MHz** 输入载波。

③ 复位引脚

RSTPD（**31** 脚）：当复位引脚出现一个从高到低的电平跳变时 **RC500** 复位。该引脚为高时，**RC500** 停止工作；该引脚被置为低电平，**RC500** 才能正常工作。

④ 晶振引脚

OSCIN，OSCOUT（**1** 脚，**32** 脚）：晶振的输入、输出引脚。**RC500** 使用 **13.56MHz** 的晶振。

⑤ **MIFARE** 接口

MFIN，MFOUT（**3** 脚，**4** 脚）：**MIFARE** 接口输入、输出引脚。

⑥ 并行接口

MF RC500 有 **16** 个引脚用于控制并行接口。

AD0～AD7（**13** 脚至 **20** 脚）：**8** 位双向数据总线（也可复用为地址线）。

A0～A2（**22** 脚至 **24** 脚）：地址线输入。

NCS（**9** 脚）：片选信号，选择 **RC500** 的并行微控制器接口。输入，高电平有效。

NWR（**10** 脚）：写信号线，输入，低电平有效。

NRD（**11** 脚）：读信号线，输入，低电平有效。

ALE（**21** 脚）：地址锁存允许引脚，输入，高电平有效。

IRQ（**2** 脚）：中断请求引脚，当有中断事件发生时产生中断信号。输出，高电平有效。

3.4.3 MF RC500 的寄存器

MCU 对 **MF RC500** 的控制是通过对其内部寄存器的读写来实现的。**MF RC500** 内部共有 **64** 个寄存器，分成 **8** 页，每页 **8** 个寄存器。关于 **MF RC500**的寄存器描述请参见附录 **2**。

3.4.4 MF RC500 的并行接口

MF RC500的并行接口支持与各种类型的 **MCU** 的直接连接，还可以与 **PC** 机的增强型并行接口（**Enhanced Parallel Port,EPP**）直接相连。

在上电复位或硬件复位后，**MF RC500** 将自动检测当前与其并行微控制器接口连

接的控制器的接口类型。表 **3-3** 显示了 **MF RC500** 与不同类型的并行接口的连接[22]。

表 3-3 不同类型的并行接口的连接

MF RC500	并行接口类型				
	独立的读、写信号线		普通读写信号线		
	专用地址线	复用地址线	专用地址线	复用地址线	带握手信号的复用地址线（EPP）
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

微控制器使用独立的读、写信号线与普通的读写信号线这两种不同的并行接口连接 **MF RC500**，相应的总线时序也是不一样的。要注意根据表中的接口类型进行正确连接。当使用独立的读、写信号线时，关于地址线，可以选择专用地址线，也可以使用数据/地址线复用的方式——见图 **3-4**。

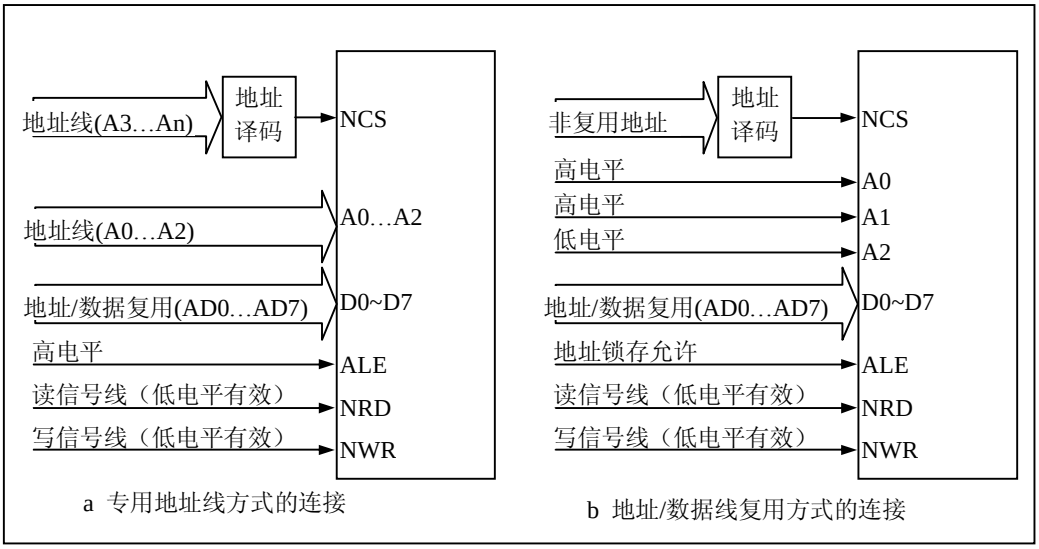


图 3-4 使用独立的读、写信号线与 MCU 的连接

3.5 读写模块硬件说明

读写模块的硬件设计中，最关键有两个部分。一是 **GP32** 与 **MF RC500**芯片的连接，这是实现 **GP32** 控制 **MF RC500**正常工作的硬件基础。二是读写模块的天线部分的设计及其与射频读写芯片的连接，这部分的性能将直接决定射频读写操作能否正常进行。

3.5.1 GP32 与 MF RC500 的连接

主控芯片 GP32 通过 I/O 引脚控制 MF RC500 的并行接口以及 MF RC500 的供电。

（1）与 MF RC500 并行接口的连接

GP32 通过 MF RC500 并行接口实现对 MF RC500 芯片的控制和数据传输。GP32 对 MF RC500的并行接口采用独立的读、写信号线连接，用两个 I/O 脚分别控制 MF RC500 的读、写信号线。为了节省 I/O 口，这里采用了地址/数据线复用的方式，这样就不需要专门的 I/O 口来控制地址线。GP32 与 MF RC500 之间的连接根据图 3-4b 的连接方式进行。

如图 3-5 所示。GP32 的 PTB0～PTB7 连接 RC500 的 D0～D7，作为数据/地址线，传输数据及地址信息。由于采用数据/地址复用的连接方式，RC500 的地址线引脚 A0、A1、A2 未被使用，按连接要求分别给其接高电平、高电平和低电平。

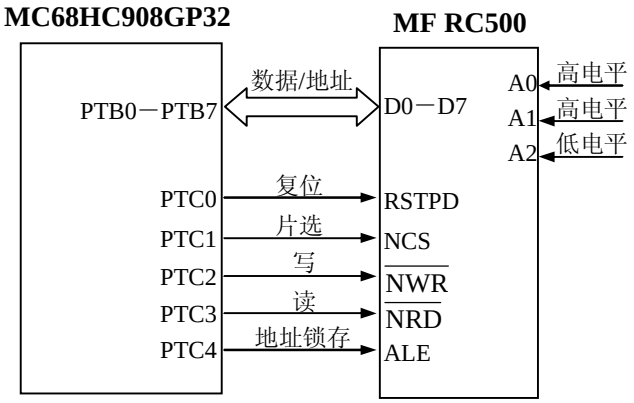


图 3-5 GP32 与 MF RC500 连接示意图

GP32 的 PTC0 连接 RC500 的 RSTPD，PTC0 脚输出高电平到低电平的跳变，将引起 RC500 的复位。PTC1、PTC2、PTC3 和 PTC4 分别接 RC500 的 NCS、NWR、NRD 和 ALE，是 GP32 访问 RC500 寄存器的读写控制信号线。PTC1 接 NCS,控制片选信号。PTC2 接 NWR，控制写信号线。PTC3 接 NRD，控制读信号线。PTC4 接 ALE，是地址锁存允许信号线。需要注意的是，GP32 不同于 51 系列单片机，没有专门的读、写信号线，其读写操作是根据时序要求通过对 I/O 脚编程来实现的。

（2）电源控制电路的连接

MF RC500 正常工作时，需要 5V 电源供电。在读写模块中使用了电源控制电路，通过 GP32 I/O 引脚输出的高低电平控制 MF RC500 的电源供电。如图 3-6 所示，左边是电源控制电路输入控制－－GP32 的 I/O 引脚 PTE0；右边是电源控制电路的输出－－MF RC500 的电源电压

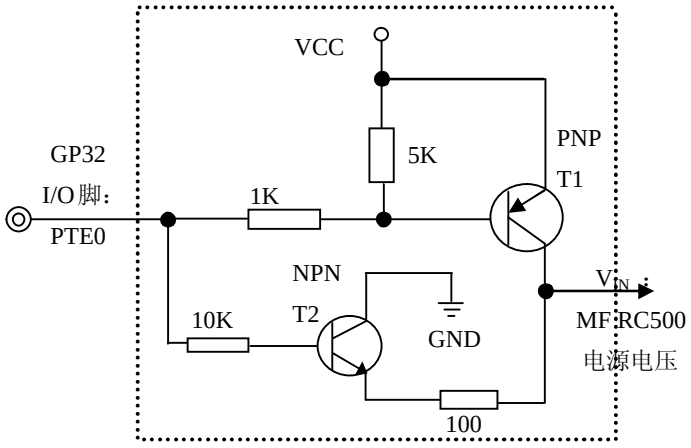


图 3-6 MF RC500 电源控制电路

V_{IN}。当 PTD1 输出为 0（低电平）时，三极管 T1 导通、T2 截止，输出端 V_{IN} 为 VCC（5V）；当 PTD1 输出为 1（高电平）时，三极管 T1 截止、T2 导通，输出端 V_{IN} 为 0V。

3.5.2 天线及相关电路的设计

MF RC500 根据其寄存器的设定对发送数据进行调制得到发送的信号，通过由天线驱动引脚 TX1 和 TX2 驱动的天线以 13.56MHz 的电磁波形式发送出去。在其射频范围内的 RFID 卡采用 RF 场的负载调制进行响应。天线接收到卡片的响应信号经过天线匹配电路送到 MF RC500 的接收引脚 RX，芯片内部的接收器对接收信号进行解调、译码，并根据寄存器的设定进行处理，最后将数据发送到并行接口由微控制器读取。

为了获得稳定、可靠的射频信号，天线部分的电路设计非常关键，这也是一门很深入的技术[12]。在设计读写模块的天线电路时，这一部分参考了 Philips 公司所提供文档中的推荐电路图[23]。

（1）高频滤波电路

为了减少信号线上的干扰，使用了EMC 高频滤波电路。MF RC500的天线引脚 TX1、TX2、RX 以及参考电压 VMID 先经过 EMC 滤波电路，然后再与天线匹配电路连接。参见图 3-7 的 EMC 滤波电路图，L1、L2、C7、C8、C9、C10 组成了 MF RC500 射频发送信号的滤波电路；R1、R2、C5、C6 组成了接收信号的滤波电路，为了达到良好的电磁兼容，在制作印刷电路板（PCB）时，这部分的电路必须紧靠 MF RC500的天线引脚 RX、TX1、TX2。

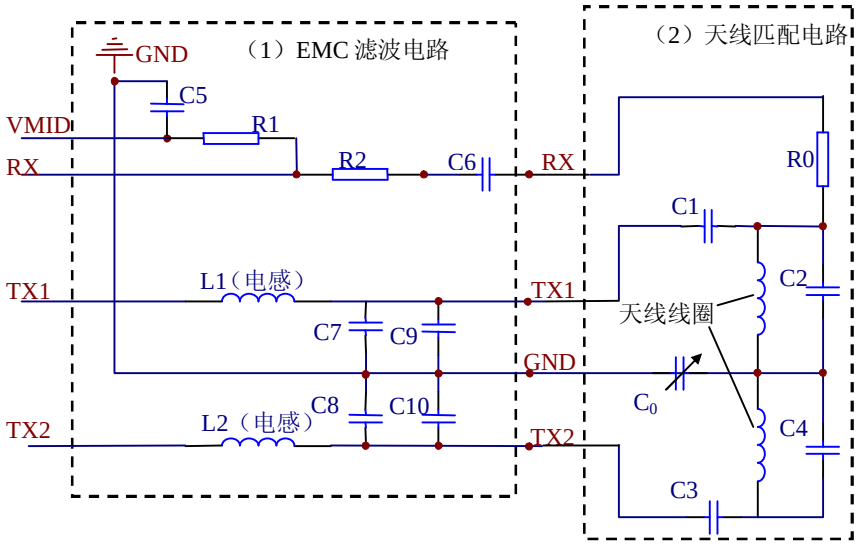


图 3-7 射频天线电路原理图

（2）天线及匹配电路

为了给 RFID 卡提供足够的能量，天线与卡片间必须实现紧耦合，耦合系数最少

为 0.3（耦合系数为 0 时，即由于距离太远或磁屏蔽导致完全去耦；耦合系数为 1 即全耦合）。因此天线线圈采用直径为 1mm 的导线，设计为三圈的 65mm×54mm 方形天线。此时，天线线圈产生的电感，有下列公式计算：

$L=2\times I\times \ln(1/D)\times N^{1.8}$ [3.1] [24]；

其中：

L 为天线电感；

I 为环形导体的长度（即一圈的周长），单位为 cm；

D 为导体的宽度，即导线直径，单位为 mm；

N 为线圈的圈数。

由公式[3.1]可计算出天线线圈的电感值约为 1uH。为了使天线线圈接收的来自芯片天线引脚的射频信号尽可能减少损失与辐射，采用了如图 3-7 所示的天线匹配电路对其进行阻抗转换。天线匹配电路的电容 C1、C2、C3、C4 的参数由天线的电感值决定。由于每块不同的天线电路板实际的天线线圈电感值总是会稍有差异，因此在天线匹配电路上使用了一个可调电容 C0，通过调整可调电容将每块天线板的读写距离调整到最佳。

3.6 硬件测试

硬件系统的稳定和可靠是软件编写的前提条件。如果硬件系统本身存在不稳定因素，会给软件的编写、调试带来极大的困难。因此在硬件设计阶段，为确保硬件系统的可靠性，消除硬件系统的隐患，进行大量的实验来测试硬件系统是必不可少的。

对于读写模块的硬件系统，主要的硬件测试可以分成三个部分：GP32 微控制器部分，GP32 对 MF RC500 的控制以及 MF RC500 的天线部分。

3.6.1 GP32 微控制器系统的测试

对于 GP32 微控制器部分的测试，即测试 GP32 最小系统的工作是否正常。GP32 最小系统[25]外围电路包括电源供电电路、锁相环 PLL 滤波电路、晶振电路、复位电路以及串行通信电平转换电路（见图 3-8）。

首先测试 GP32 的工作：

（1） 上电复位后，测量 GP32 的管脚 VSS、VDD，测试电源供电电路是否正常供电；测量复位引脚 RST 是否处于高电平状态。

（2） 用示波器测量晶振工作的波形，测试晶振电路是否工作正常。

（3） 在 GP32 的一个通用 I/O 脚上接测试指示灯。编写一个简单的测试程序写到 GP32 内，让接有测试指示灯的 I/O 口的高低电平周期性变换，观察指示灯的闪烁情况。测试程序中包括对 GP32 进行一些初始化设置，包括堆栈指针的设定、PLL 锁相环的配置、看门狗模块的设置。通过调整测试程序中的精确延时部分，可以看到指示灯闪烁快慢的变化，因此可以判断 GP32 芯片工作正常。

在确定 GP32 正常工作之后，接下来测试串行通信电路的工作。在外围焊上以 MAX232 芯片为核心的串行通信电平转换电路，通过串行接口和 PC 机相连。设置 GP32 的 SCI 串行通信模块，编写一个串行通信测试程序。在 PC 机端用 VB（或其他软件）通过串行通信控件编写一个与之配合的简单的串行数据接收和发送程序，测试 GP32 的 SCI 串行模块及外围串行通信电路的工作情况。

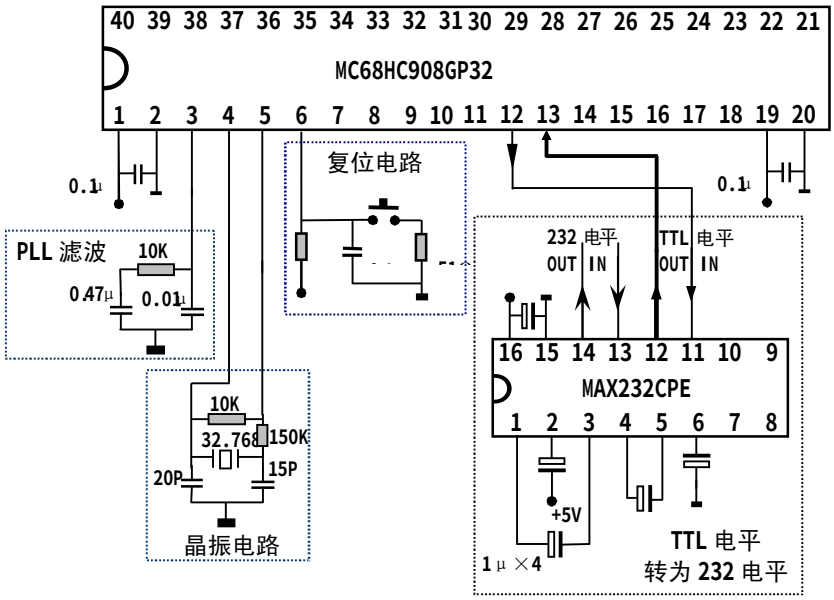


图 3-8 具有串行通信功能的 MC68HC908G932 最小系统电路原理图

3.6.2 GP32 对 MF RC500 的控制

测试 GP32 对 MF RC500的控制部分实际上包括 MF RC500电源供电电路、MF RC500 的晶振电路及 GP32 与 MF RC500 的连接。

（1） 读写模块对 MF RC500的供电是由 GP32 来控制的，通过 I/O 脚 PTE0 输出的高低电平控制电源控制电路输出 0V 或者 5V 电压给 MF RC500。在 MCU 方编写测试程序，设置 PTE0 为输出脚，当 PTE0 输出为 0（低电平）时，测量 MF RC500 的电源引脚 AVDD、DVDD、TVDD 是否为 5V；当 PTE0 输出为 1（高电平）时，测量 MF RC500 的电源引脚 AVDD、DVDD、TVDD 是否为 0V，据此测试 MF RC500 的电源供电电路是否工作正常。

（2） 为 MF RC500 供电，使用示波器测量 MF RC500 的晶振引脚，测试其晶振电路是否工作。

（3） 测试 GP32 与 MF RC500 的连接。首先测量 GP32 的 I/O 控制口是否与 MF RC500 的并行接口口正常连接。然后在上电后，编写测试程序，程序中给 PTC 口输出高或低电平，使用万用表检查 MF RC500并行接口控制线的电平是否与程序输出一致。接下来通过编写读写 MF RC500 寄存器的程序来检测 GP32 与 MF RC500 的连接。若根据 MF RC500 的读写时序编写的读写寄存器程序可以正确执行，则可以判断 GP32 与 MF RC500 的硬件连接是正确的。在实际过程中，这一步实验既包含对硬件的检测，也包括对读写寄存器的程序的验证(读写寄存器的程序参见第四章，4.4.1)。因此当出现问题，既有可能是硬件连接的错误，也有可能是软件上程序的原因，准确判断非常关键。

3.6.3 MF RC500 的天线测试

制作的 MF RC500 的天线板包括天线线圈及匹配电路，其测试包括两个方面：一是天线的正常工作，即可以与 RFID 卡正确通信；二是天线的射频范围，即读写模块的工作距离。这两方面的关键都在于电路中电子元件参数的选。

进行天线测试时，使用的是软件、硬件相结合的方式。

在软件测试程序中，读写模块向卡片发送 Request 询卡指令（可参考第四章软件设计部分关于询卡操作），如果卡片正确响应，MCU 就使某个 I/O 口发光二极管（LED）亮；卡片不能正确响应时，LED 不亮。

在硬件上，使用可调电容试验元件 C1、C2、C3、C4 的参数。开始时使卡紧靠读卡器，当 LED 灯亮后不断增大卡与读卡器之间的距离，直到 LED 灯熄位置，记录这时候的大概读写距离和电容值。不断改变电容的值，可以得到不同的读写距离，直到得到最大读写距离。经实验测试，最终天线板上使用的 C1、C2、C3、C4 的参数为 22pf、18pf、24pf、160pf；天线的最大询卡距离大约为 6cm。每块天线板的性能稍有差异，此时可以对可调电容 C0 进行调整，以达到天线的最佳工作距离。

第四章 读写模块软件设计

在上一章读写模块的硬件设计的基础上，本章将重点介绍读写模块的软件设计。软件设计的总体思想是：通过对主控 MCU 的编程，控制射频读写芯片根据 ISO/IEC14443A 协议与 RFID 卡(Mifare 1)进行射频通信，完成对 RFID 卡的各种操作，并将有关操作以函数形式合理封装，供二次开发的用户调用。

为了实现读写模块的通用性，作者选用 GP32 作为主控芯片进行软件设计时，首先对硬件相关的部分进行了分析，将与硬件联系密切的部分独立出来，以便将来选用其他系列的主控芯片时只需要做尽可能少的改动就实现软件的移植。

4.1 软件设计概述

4.1.1 软件功能概述

读写模块的软件要实现两个基本功能：一是实现在线编程，可将用户自己的应用程序在线写入到 GP32 Flash存储区的用户程序空间,支持用户的二次开发。这部分是由 GP32 监控程序完成的；二是实现对 RFID 卡的操作，提供方便的函数给用户的应用程序调用。这部分其实包括两个层次，底层的与 RFID 卡通信的驱动函数，以及在其之上封装起来的供外部调用的接口简单、明确的高层命令接口函数。

使用专门的 GP32 的写入器向空白的 GP32 芯片内烧入读写模块软件，除非再使用写入器将其擦除，否则读写模块软件就一直驻留在 GP32 的 Flash 存储区。GP32 芯片 32K 的存储空间被分成了两个部分，驻留的读写模块软件占用一部分 Flash 存储区，剩下的存储空间供用户的应用程序使用。读写模块的软件部分由三部分组成：GP32 监控程序、读写卡操作高层命令接口以及与 RFID 卡通信的底层通信函数（见图 4-1）。

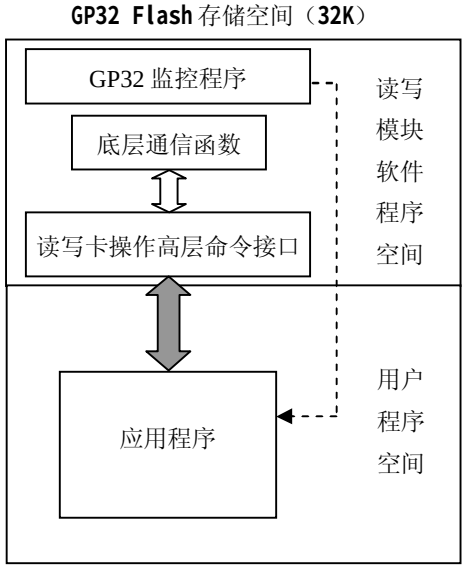


图 4-1 读写模块软件结构图

读写模块在射频识别应用中 是应用系统与 RFID 卡之间数据交换的接口，见图 4-2。应用系统只需要向读写模块发送操作命令，由读写模块完成具体的对 RFID 卡进行读、写等操作。非接触通信的所有具体细节，如建立通信、防止碰撞、

身份验证及对卡的操作等，均是由底层通信函数完成。而用户直接调用的则是按统一的命令格式封装好的读写卡操作函数，如询卡、读卡、写卡等。底层的通信函数对应用系统或者用户的应用程序来说其实是透明的。

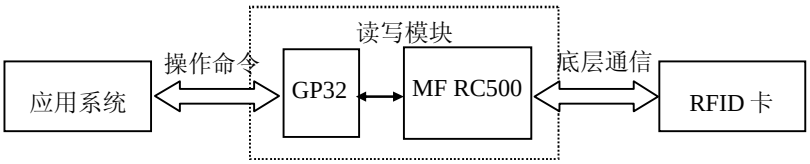


图 4-2 读写模块应用系统数据流向

4.1.2 软件开发环境

软件设计的一个重要环节是确定开发工具和编程语言[26]。在进行读写模块的软件开发时，作者使用的是实验室自主开发的“SD-1 集成开发环境”来编写、调试、下载程序。“SD-1 集成开发环境”适用于 Freescale 8 位 MCU 的程序开发，提供了 C 编译器、宏汇编、连接器、库管理和工程项目管理以及程序下载等完整的开发环境，支持 Freescale 08C 语言（简称 08C）和汇编语言。

为了给用户提供明确、方便的函数接口，读写模块主体程序的实现使用 08C 语言进行。但主控 MCU 和射频读写芯片的接口部分程序对时序要求非常严格，汇编语言成了唯一的选择，因此在软件设计中，需要使用 C 语言和汇编语言混合编程技术。

4.2 读写模块中的在线编程技术

驻留在芯片中的监控程序可以实现用户应用程序的在线写入。其内部包含了通信握手、Flash 页擦除、Flash 页写入、校验、数据接收与发送及断点调试处理等主要功能模块。每次系统复位后首先运行驻留的监控程序，在系统初始化后进行 MCU 和 PC 机的串行通信握手，如果握手成功，MCU 接收并执行 PC 机方的命令：在线写入、断点调试等。如果复位后的规定的时间内握手不成功，则判断用户存储空间是否有用户程序，若有则转入用户程序执行，否则监控程序继续发送握手信号。

图 4-3 是实现在线写入功能的监控程序主流程图。

驻留在芯片中的监控程序与 PC 机端的软件“SD-1 集成开发环境”配合使用，可以非常方便地完成对 GP32 芯片的在线编程。

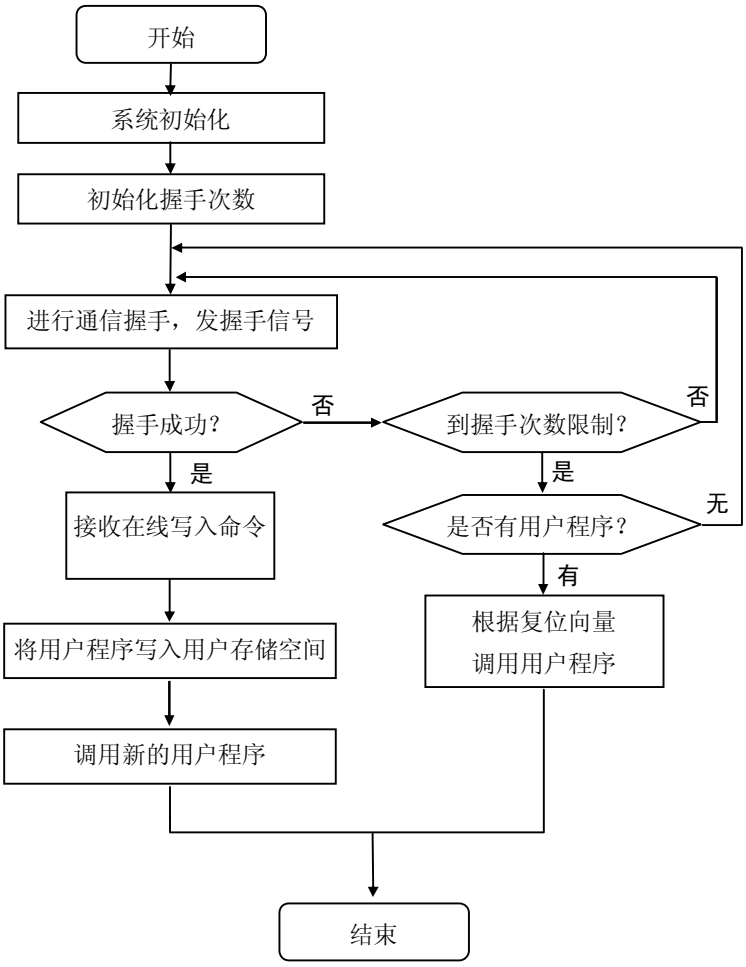


图 4-3 监控程序主流程图

4.3 软件设计中与主控芯片相关部分

读写模块是一个嵌入式应用系统，设计者主要是对主控 MCU 进行编程操作，这种低端的程序设计与主控 MCU 的关系很密切。但本文所开发的读写模块不仅要实现对 **RFID** 卡的读写功能，而且要具有通用性，要能很方便地集成到其它的嵌入式应用系统中去，所以在进行读写模块的软件开发时，尽量做到程序的硬件无关性。为实现软件的硬件无关性，要考虑以下几个方面：

（1）主控 MCU 存储空间的不同。存储空间有程序代码的 FLASH 存储空间和数据的 RAM 空间两部分，不同的 MCU 这些存储空间的大小和位置都是不同的。但是选用嵌入式 08C 语言作为开发工具，将 MCU 的存储空间分配交给编译器去完成，在编译环境中设定 MCU 的存储空间。如图 4-4 是在 **SD-1** 集成开发环境中对 **GP32** 的设置值。虽然这个工作交给了编译器去完成，但在使用单片机的 **C** 语言编程时，对内存

资源的使用仍然要“省吃俭用”，在进行程序设计过程中的变量尽量使用局部变量。对于需要汇编编程的部分，变量空间也是从堆栈中开辟空间，使用结束，立即恢复堆栈指针，释放占用的堆栈空间。另外，在程序实现中，要开设一个读写的数据缓冲区供读写模块进行读写数据交换。

图 4-4 GP32 存储空间设置值

（2）主控 MCU 总线频率的差异。不同的 MCU 执行速率有差别，即使对于特定的 MCU 也会根据不同的场合设定不同的工作频率，例如 GP32 通过对锁相环模块的设置，可将总线频率设置在 2.4576MHz~8MHz 之间[27]。这种速率的不同，对软件的通用性影响很大，特别是时序接口和延时部分。作者在设计软件时，将总线频率进行了宏定义，在软件中与总线频率有关的延时部分根据这个宏进行循环，决定延时时间。

（3）主控 MCU 的 I/O 口的差别。不同 MCU 的 I/O 口数目及口地址不同，在实际应用中与读写模块的连接口也会有所差别。本文使用口地址的宏定义来屏蔽这种差别，将与连接有关的 I/O 口宏定义集中在一个头文件中，在不同的连接方式中，只需要更改这种宏定义就可以适应不同的 MCU。根据第三章图 **3-5** 的连接方式，GP32 的 **PTB0～PTB7** 接数据地址复用线 **D0～D7**，**PTC0** 接 **RSTPD,PTC1** 接片选信号线 **NCS**，**PTC2** 接写信号线 **NWR,PTC3** 接读信号线 **NRD**，**PTC4** 接地址锁存允许信号线 **ALE**。头文件中 **I/O** 口定义如下所示：

```
;硬件引脚定义：B 口为数据口，C 口为控制口
RC500Data     EQU PTB          ; RC500Data为为数据口 PTB
RC500DataD    EQU DDRB         ; RC500DataD为数据口方向
RC500DataPUE  EQU PTBPUE       ; RC500DataPUE为数据口上拉
RC500CtlD     EQU PTC          ; RC500CtlD为控制口 PTC
RC500CtlDD    EQU DDRC         ; RC500CtlDD为控制口方向
RC500_RSTPD   EQU 0            ; RC500_IRQ接 PTC0
RC500_NCS     EQU 1            ; RC500_NCS接 PTC1
RC500_NWR     EQU 2            ; RC500_NWR接 PTC2
```

```
RC500_NRD      EQU 3              ; RC500_NRD接 PTC3
RC500_ALE      EQU 4              ; RC500_ALE接 PTC4
```

（4）主控 MCU 的初始化操作的不同。不同的 MCU，其初始化过程也不相同，对于这一部分，实际应用中要根据具体的 MCU 作相应操作。

4.4 GP32对 MF RC500 的基本操作

GP32 通过对 MF RC500的控制，实现 RFID 卡的读写操作。GP32 对 MF RC500的控制有三种方式：

（1）设置 MF RC500 的状态。

（2）发送命令，要求 MF RC500 执行相应的动作。

（3）通过读 MF RC500 的状态标志来监测 MF RC500 的工作情况。

无论上述的哪一种方式，都是通过读/写 MF RC500 的寄存器来实现的：配置 MF RC500 就是设置寄存器的某些位；执行命令要向命令寄存器写入命令代码以及通过 FIFO 缓冲区寄存器向缓冲区写入命令参数；监测 MF RC500即读状态寄存器的标志位。因此，读写寄存器是所有操作的基础。

4.4.1 访问 MF RC500 寄存器

（1）MF RC500 的寄存器分页机制

RC500 内部共有 64 个寄存器（寄存器列表请参见附录 2），分 8 页。每页 8 个寄存器，每页的第一个寄存器均称为页寄存器 Page-Register。

页寄存器的结构如图 4-5 所示。

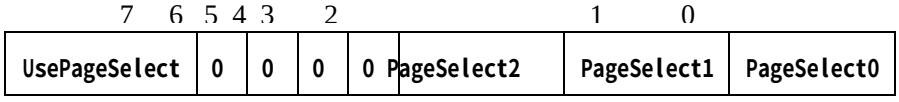


图 4-5 页寄存器结构图

访问 MF RC500 的寄存器时，地址可以采用页模式或线性地址这两种方式（见表 4-1）。

表 4-1 寄存器地址的两种访问方式

地址访问方式	页寄存器第 7 位	寄存器地址					
页模式	1	PageSelect2	PageSelect1	PageSelect0	AD2	AD1	AD0
线性地址	0	AD5	AD4	AD3	AD2	AD1	AD0

当页寄存器的使用页选择（UsePageSelect）位为 1 时，页寄存器的页选择位 PageSelect0、PageSelect1、PageSelect2 有效，其值即为页号，而地址/数据线上

的 **AD0～AD2** 则确定是该页中的第几个寄存器，这种确定寄存器地址的方式称为页模式。当页寄存器的 **UsePageSelect** 位为 **0**，页选择无效，此时，寄存器的地址组成就由地址/数据线上的 **AD0～AD5** 确定，这种方式称为线性地址，

在使用页模式访问寄存器时，由于每次访问的寄存器可能在不同页，则访问寄存器的操作实际包括两个步骤。第一步要向写页寄存器，选中寄存器所在的页，第二步才可进行具体的访问寄存器操作。使用线性地址则无须考虑分页机制，因此使用线性地址方式访问寄存器效率更高。

由于读写模块的硬件连接使用了地址/数据线复用的方式，为提高工作效率，本文对寄存器的访问采用线性地址。需要注意的是，**MF RC500** 每次复位后，其页寄存器默认值为“**$80**”（**$**表示 **16** 进制），即默认使用页模式且选中第 **0** 页。因此，在复位操作后，立即将页寄存器的值改为“**$00**”,使页模式失效，即可使用线性地址访问寄存器。

（**2**）读写 **MF RC500** 寄存器

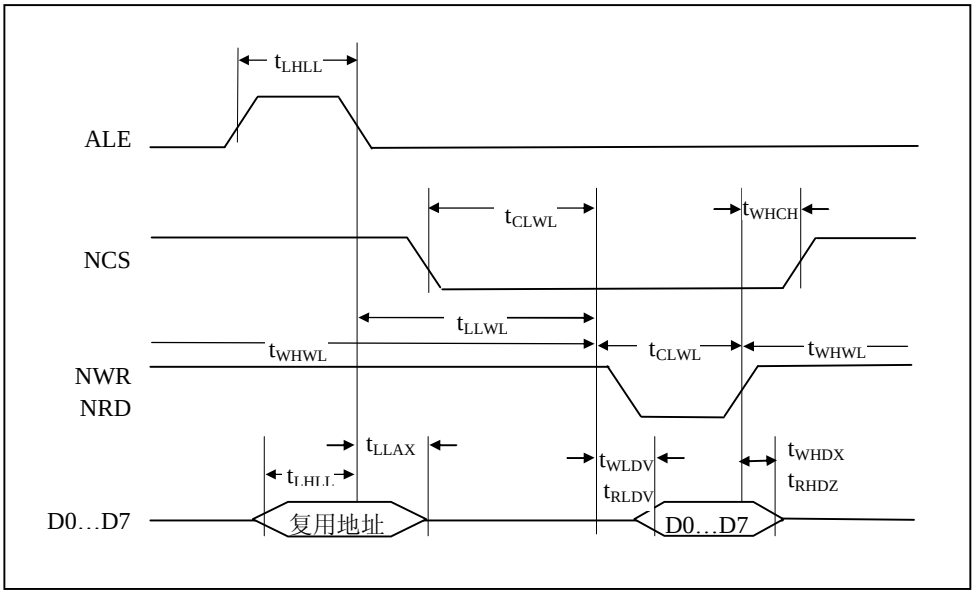


图 4-6 独立读写信号线的时序图

读写 **MF RC500**的寄存器的操作严格按照读写信号线的时序进行,关于时序图及说明参见图 **4-6**、表 **4-2**。

表 4-2 独立读写信号线的时间定义

时间标志	参数	最小（ns）	最大(ns)
t_{LHLL}	ALE 脉冲宽度	20	—
t_{AVLL}	复用线上地址有效直到 ALE 为低（地址设置时间）	15	—
t_{LLAX}	ALE 为低之后复用线上地址有效（地址保持时间）	8	—
t_{LLWL}	ALE 为低到 NWR/NRD 为低	15	—

t_{CLWL}	NCS 为低到 NRD/NWR 为低	0	—
t_{WHCH}	NRD/NWR 为高到 NCS 为高	0	—
t_{RLDV}	NRD 为低直到数据有效	—	65
t_{WLDV}	NWR 为低直到数据有效	—	35
t_{WHDX}	在 NWR 为高之后数据的保持时间	8	—
t_{WLWH}	NRD/NWR 脉冲宽度	65	—
t_{WHWL}	连续读写操作之间的时间	150	—

根据时序图，可以写出 **GP32** 读、写**MF RC500**寄存器的程序。下面给出读 **MF RC500** 寄存器的汇编语言程序（入口参数为操作地址，通过寄存器 **X** 传递，出口参数为读取的值，通过寄存器 **A** 传递）。

```
PSHX                        ; X暂存入口参数，操作地址
PSHA                        ; A存放出口参数，读出的值
;① MF RC500 数据（RC500Data）口初始化
LDA   #%11111111            ;RC500Data口方向初始为输出
STA   RC500DataD
;② 延时 1us
LDA   #!1                   ;延时入口参数 n=1
JSR   Delay_Nus             ;调延时子程序
;③ 时序图步骤:
BSET  RC500_ALE,RC500CtlD   ;步骤 1:ALE=1,地址锁存有效
STX   RC500Data             ;步骤 2:寄存器地址上线
BCLR  RC500_ALE,RC500CtlD   ;步骤 3:ALE:1->0,地址无效
BCLR  RC500_NCS,RC500CtlD   ;步骤 4:片选信号线 NCS:1->0,片选有效
LDA   #%00000000            ;步骤 5:数据口方向改为输入
STA   RC500DataD
LDA   #$FF                  ;步骤 6: 为得到稳定数据信号，数据口上拉
STA   RC500DataPUE
BCLR  RC500_NRD,RC500CtlD   ;步骤 7:读信号线 NRD:1-->0,读有效
LDA   RC500Data             ;步骤 8:读取数据口数据
BSET  RC500_NRD,RC500CtlD   ;步骤 9:NRD:0->1 ，读信号线无效
BSET  RC500_NCS,RC500CtlD   ;步骤 10:NCS:0->1，片选无效
PULA
PULX
```

由于 **GP32** 工作频率设置为 **2.4756MHz**，一个时钟周期 **400ns**，大于时序表中 t_{WLDV} 的最大值 **35ns**。为保证写信号线有效后数据及时上线，在写寄存器的程序中，将数据上线的操作调整到写信号线有效之前执行。从而解决了时序中关于 t_{WLDV} 的限制。下面给出写 **MF RC500** 寄存器的汇编言语程序（入口参数有两个：寄存器 **X** 传递操作地

址，寄存器 A 传递写入的值）：

```
PSHX                        ; X暂存入口参数，操作地址
PSHA                        ; A暂存入口参数，写寄存器的值
;① MF RC500 数据（RC500Data）口初始化
LDA   #%11111111            ;RC500Data口方向初始为输出
STA   RC500DataD
;② 延时 1us
LDA   #!1                   ;延时入口参数 n=1
JSR   Delay_Nus             ;调延时子程序
;③ 时序图步骤:
BSET  RC500_ALE,RC500CtlD   ;步骤 1:ALE=1,地址锁存有效
STX   RC500Data             ;步骤 2:寄存器地址上线
BCLR  RC500_ALE,RC500CtlD   ;步骤 3:ALE:1->0,地址无效
BCLR  RC500_NCS,RC500CtlD   ;步骤 4:片选信号线 NCS:1->0,片选有效
PULA                        ;步骤 5 数据口数据上线（数据取自堆栈）
STA   RC500Data
BCLR  RC500_NWR,RC500CtlD   ;步骤 6:写信号线 NRD:1-->0,写有效
NOP                         ;一个指令周期空闲操作，确保写操作完成
BSET  RC500_NWR,RC500CtlD   ;步骤 7:NRD:0->1 ，写信号线无效
BSET  RC500_NCS,RC500CtlD   ;步骤 8:NCS:0->1，片选无效
PULA
PULX
```

4.4.2 MF RC500 的 FIFO 缓冲区机制

MF RC500 内部有 64 字节的 FIFO（First In First Out,先进先出）缓冲区，是 MCU 与 RC500 之间输入和输出数据流的缓存。缓冲区中数据的流向按照先进先出的顺序进行。

与 FIFO 缓冲区状态关系紧密的寄存器有：

(1) FIFO 缓冲区数据寄存器（地址$02）：FIFOData

FIFO 缓冲区的输入输出数据总线直接连接到 FIFO 缓冲区数据寄存器 FIFOData 上。因此，写FIFOData 寄存器即是向 FIFO 缓冲区内存入一个字节并使缓冲区的内部写指针增 1。连续写 n 次 FIFOData 寄存器，就向 FIFO 缓冲区存入了 n 个字节数据。读 FIFOData 寄存器，就是将读指针指向的缓冲区内的内容读出并使其内部读指针增 1。连续读 n 次 FIFOData 寄存器，就从 FIFO 读走 n 个字节数据。

(2) FIFO 缓冲区数据长度寄存器（地址$04）：FIFOLength；

FIFOLength 的值是存储在缓冲区中的数据的字节数，即其内部读指针与写指针之间的距离。向 **FIFOData** 寄存器写一个字节，**FIFOLength** 的值增 **1**；读 **FIFOData** 寄存器，**FIFOLength** 的值减 **1**。

(3) FIFO 缓冲区大小寄存器（地址**$29**）：**FIFOLevel**；

FIFO 缓冲区的容量是 **64** 字节，在实际使用时，用户可通过 **FIFOLevel** 寄存器设定需要使用的 **FIFO** 缓冲区大小 **Waterlevel**。**Waterlevel** 是判断 **FIFO** 缓冲区是溢出还是不满的警戒线。**FIFO** 缓冲区中实际存储的数据个数（即 **FIFOLength** 寄存器的值）与 **Waterlevel** 的比较，将会影响基本状态寄存器 **PrimaryStatus** 的位第 **0** 位 **LoAlert** 标志、第 **1** 位 **HiAlert** 标志。

若 **FIFOLength≤WaterLevel**，则标志位 **LoAlert=1**；

若 **64－FIFOLength≤WaterLevel**，则标志位 **HiAlert=1**。

例如，若设定 **WaterLevel＝4**，则：

FIFOLength＝60 时，**HiAlert=1**；**FIFOLength＝59** 时，**HiAlert=0**

FIFOLength＝4 时，**LoAlert=1**； **FIFOLength＝5** 时， **LoAlert=0**

(4) 控制寄存器（地址**$09**）：**Control**

7	6	5	4	3	2	1	0
0	0	StandBy	PowerDown	Crypto1On	TStopNow	TstartNow	FlushFIFO

除了读写缓冲区会影响缓冲区指针外，通过设定控制寄存器（**Control**）的第 **0** 位 **FlushFIFO** 的值为 **1** 可复位缓冲区指针。当 **FlushFIFO＝1** 时，相应的缓冲区长度寄存器 **FIFOLength** 的值将变为 **0**，错误标志寄存器（**ErrorFlag**）的第 **4** 位 **FIFOOvll** 将被清除，实际存放在缓冲区内的所有字节将被清除。

FIFO 缓冲区作为并行接口的缓存，在程序中的一个重要作用就是传递执行 **MF RC500** 命令时需要的参数。当 **MCU** 启动一个命令操作时，**MF RC500** 到 **FIFO** 缓冲区去取得执行这个命令的参数。实际中只有一个 **FIFO** 缓冲区，而对缓冲区的访问有读入和取出两个方向，因此在写程序的时一定要小心注意对 **FIFO** 缓冲区的访问。

4.4.3 MF RC500 的命令

RC500 内部有一个状态机，可以执行命令（**Command**）寄存器（地址**$01**）中的命令。命令的启动只需要将相关命令代码写到 **Command** 寄存器中。执行命令所需要的变量以及数据都是通过 **FIFO** 缓冲区来传递。读这个寄存器可以知道正在执行哪条命令。

MF RC500 的命令一共有 **13** 条，分别是：开始（**StartUp**）命令、空闲（**Idle**）命令、传送（**Transmit**）命令、接收（**Transceive**）命令、写 **E²PROM**（**WriteE2**）命

令、读E^2PROM（WriteE2）命令、取E^2PROM 中的密码（LoadKeyE2）命令、取密码（LoadKey）命令、认证 1（Authent1）命令、认证 2（Authent2）命令、载入配置（LoadConfig）命令及计算 CRC（CalcCRC）命令。关于这些命令的代码、动作含义以及通过 FIFO 传递的参数等具体说明请参见附录 3.

其中，比较特殊的指令有两条：开始（StartUp）命令和空闲（Idle）命令，这两条命令都不需要参数。StartUp 命令执行的是复位及初始化操作，只有上电复位或硬件复位才引起该命令的执行，不能通过软件方法写 Command 寄存器来执行该命令。Idle 命令的执行结果是即芯片不做任何动作。当一条命令正在执行时，向 Command 寄存器写入一个新的命令代码就会中断这条命令的执行。因此，可以通过向 Command 寄存器写 Idle 命令来中止当前命令的执行。

4.5 与 Mifare 1 的射频识别通信

4.5.1 Mifare 1 的状态及射频通信处理流程

（1）MF1 的状态

卡片在与读写设备的通信过程中的状态见图 4-7。

- POWER-OFF 状态

在 POWER-OFF 状态，卡片由于缺少负载能量而处于断电状态。

- IDLE 状态

在 IDLE 状态，卡片有电，可以侦听并识别出询卡命令 REQA、WUPA。在执行过询卡命令后，卡片进入 READY 状态。

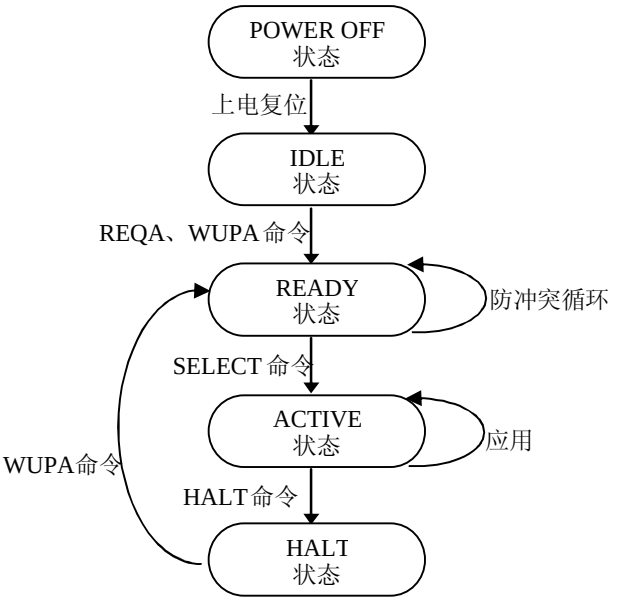


图 4-7 MF1 卡片状态图

- READY 状态

在 READY 状态，可应用防冲突方法得到完整的 UID。当根据完整的 UID，卡片被选中（SELECT）后，进入 ACTIVE 状态。

- ACTIVE 状态

在 ACTIVE 状态，卡片可执行应用操作。当接收到一个有效的挂起（HLTA）命令后，卡片进入 HALT 状态。

- HALT 状态

在 **HALT** 状态，卡片仅对 **WUPA** 命令有反应。

（**2**）**MF1** 的射频通信处理流程

MF1 的射频通信处理流程如图 **4-8** 所示。首先，读写模块发**Request**询卡命令给天线工作范围内的所有卡片。卡片在上电复位（**Power On Reset,POR**）后会响应这个询卡命令。在通过防冲突循环后，读写模块得到一张卡的序列号，于是根据该序列号选中一张卡。接下来，需要对准备访问的卡片的存储区的密码进行鉴别。在通过了密码验证后，读写模块可以对该存储区的数据进行读、写、增值、减值以及挂起等操作。

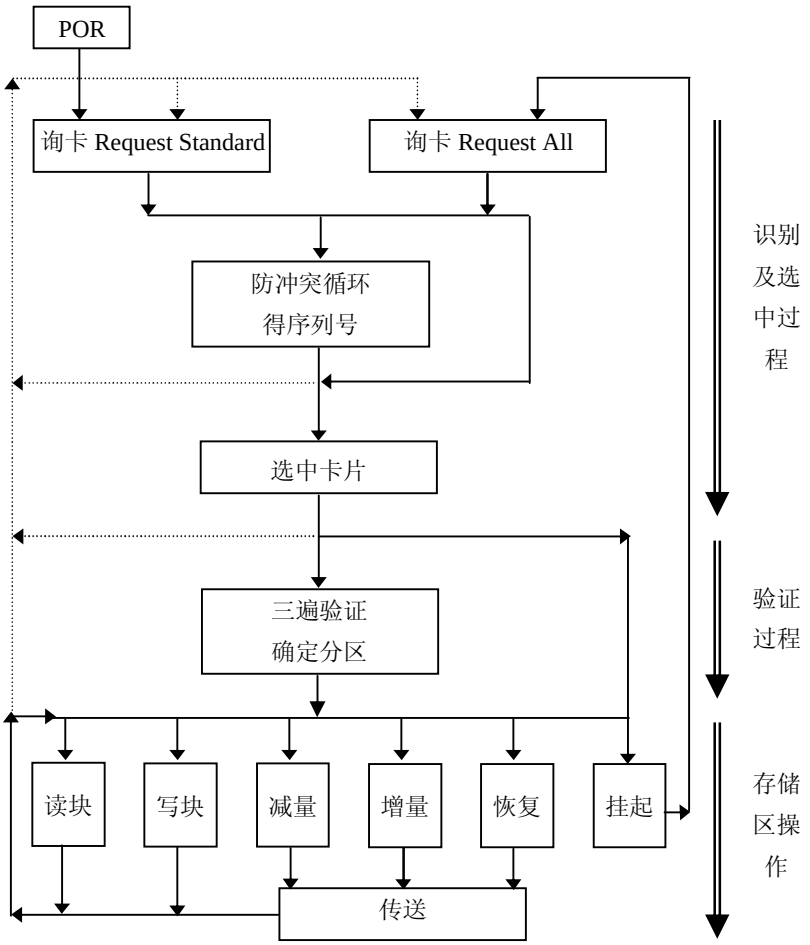


图 4-8 RFID 卡 MF1 的通信流程

读写模块与 **RFID** 卡 **MF1** 之间的通信需遵循上述流程进行。可以将该流程分成三个部分：识别及选中过程、（密码）验证过程以及存储区操作。

4.5.2 卡片识别及选中过程

（**1**）通信数据格式

读写模块与卡片在初始化建立通信及防冲突过程中通信的数据以帧的形式封装。读写模块与卡片的通信使用了三种不同类型的帧：

① 短帧

短帧用在初始通讯和一些命令中，其组成为

	LSB				MSB			
S b1		b2	b3	b4	b5	b6	b7	E

图 4-9 短帧

（参见图 **4－9**）：开始标志 **S**＋**7** 个数据位（首先传输低有效位）＋结束标志 **E**。

② 标准帧

标准帧用于数据交换，其组成为（参见图 **4-10**）：开始标志 **S**＋**n*(8** 位数据＋**1** 位奇偶校验位**)**＋结束标志 **E**，**n**≥**1**。

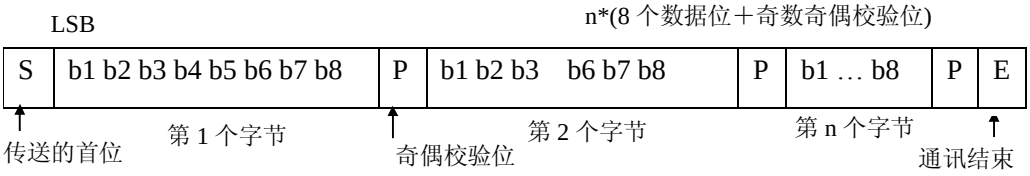


图 4-10 标准帧

③ 定向位冲突帧

当有两个 **PICC** 向 **PCD** 传输不同的位时，将发生冲突。定向位冲突帧仅在位帧防冲突循环中使用。它是 **7** 字节的标准帧，分成两个部分：第一部分是从 **PCD** 传送到 **PICC**；第二部分是从 **PICC** 传送到 **PCD**。规定数据位的总长度为 **56** 位，第一部分长度最小为 **16** 位，第二部分长度最大为 **55** 位。那么相应的，第二部分长度最小为 **1** 位，最大为 **40** 位。

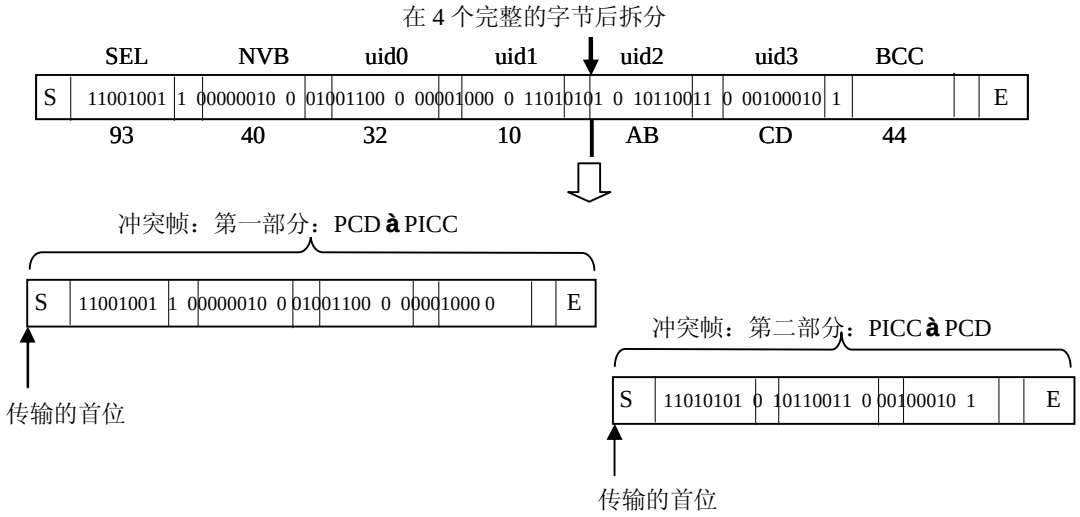


图 4-11 完整字节情况下位定向冲突帧的位组织和传送

由于拆分可能发生在一个字节的任何一个位置，那么将有两种情况：拆分发生在一个完整字节之后（参见图 **4-11**），此时将在第一部分的最后位之后加上一个奇偶校验位；或者，拆分发生在一个字节内（参见图 **4-12**），此时在第一部分的最后位之后不需加奇偶校验位，第二部分的第一个奇偶校验位被忽略。

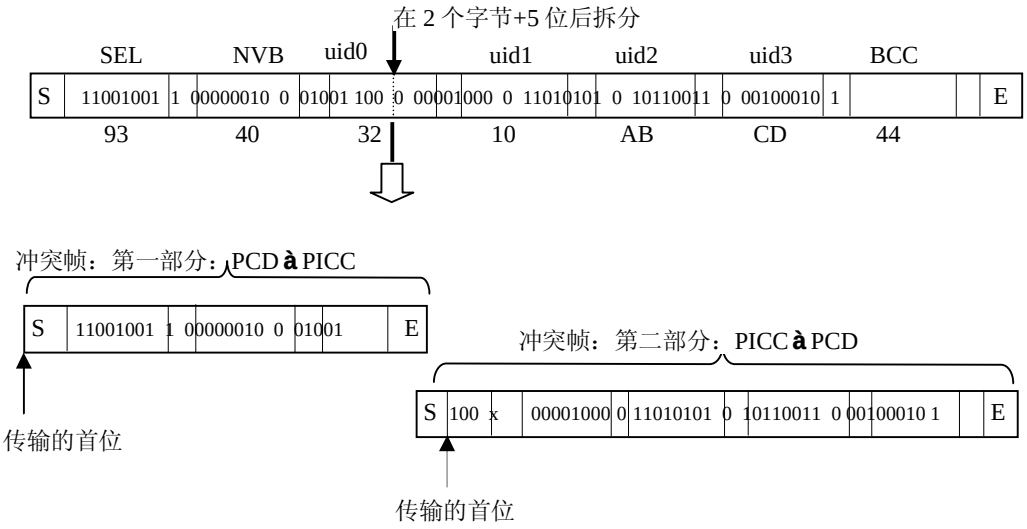


图 4-12 拆分字节情况下位定向冲突帧的位组织和传送

（2）通信命令

在读写模块与其工作范围的卡片通信进行识别及选中的过程中，使用的通信命令有：**REQA**、**WUPA**、**ANTICOLLISION**、**SELECT** 和 **HLTA**。

① **REQA**、**WUPA** 命令

REQA 和 **WUPA** 命令即询卡命令，是由读写模块发送的，用于搜寻其射频范围内的 **A** 型近耦合 **RFID** 卡，使用短帧格式发送。**REQA** 和 **WUPA** 的主要区别在于，**WUPA** 命令可识别处在挂起（**HLTA**）状态的卡，而 **REQA** 不能。其编码格式如表 **4-3** 所示。

表 4-3 短帧的询卡命令编码

b7	b6	b5	b4	b3	b2	b1	Meaning
0	1	0	0	1	1	0	‘26’=REQA
1	0	1	0	0	1	0	‘52’=WUPA

表 4-4 UID 结构定义

UID(4 字节)	UID(7 字节)	UID(10 字节)	UID CL
UID0	CT	CT	UID CL1
UID1	UID0	UID0	
UID2	UID1	UID1	
UID3	UID2	UID2	
BCC	BCC	BCC	
	UID3	CT	UID CL2
	UID4	UID4	
	UID5	UID5	
	UID6	UID6	
	BCC	BCC	
		UID6	UID CL3
		UID7	
		UID7	
		UID9	
		BCC	

② **ANTICOLLISION**、**SELECT** 命令

这两条命令用于防冲突循环，命令组成为：选择代码 **SEL**（1 字节），有效位数量 **NVB**（1 字节），以及由 **NVB** 决定的 **UID CLn**（第 **n** 层序列号，0～40 位）。当 **NVB** 指示其后有 **40** 个有效位时是 **SELECT** 命令，否则为 **ANTICOLLISION** 命

令。

ISO/IEC14443A 协议规定卡的唯一序列号 **UID** 可以有三种：**4** 个字节、**7** 个字节和 **10** 个字节。**UID CLn**（**1**≤**n**≤**3**）是 **UID** 的一部分。**UID** 结构定义见表 **4-4**。表中的 **CT** 为级联标志，编码为“**88**”。**BCC** 为 **UID CLn** 的校验，是前四个字节的异或值。

选择代码 **SEL** 的编码有三种情况：“**93**”表示选择 **UID CL1**，“**95**”表示选择 **UID CL2**，“**97**”表示选择 **UID CL3**。

NVB 的编码：**NVB** 高 **4** 位表示命令的有效字节数（包括 **SEL** 和 **NVB** 字节），当 **UID CLn**＝**0** 时，有最小的字节数 **2**；当 **UID CLn**＝**40** 时，有最大字节数 **7**。**NVB** 低 **4** 位表示 **UID CLn** 的最后一个字节的有效位数。

因此，**ANTICOLLISION** 命令和 **SELECT** 命令的格式为：

SEL	NVB	UID CLn 数据位	BCC
1 字节	1 字节	0 字节～4 字节	1 字节

注：校验位 **BCC** 仅当 **UID CLn** 为四个字节时才有。

③ **HALT** 命令

使用标准帧传送挂起命令 **HLTA**，其格式包括两个字节代码及其后的 **CRC_A** 校验。

在执行了 **HALT** 命令后卡片将处于挂起状态。此时，除非使用 **WUPA** 命令将其唤醒，否则卡片将不响应读写模块的任何命令。

（**3**）初始化和防冲突流程

按照 **ISO/IEC14443-3,**对卡片的初始化和防冲突流程如图 **4-13** 所示。读写模块首先要与其工作区域的所有 **RFID** 卡建立通信连接：向卡片发送询卡命令 **REQA**。在接收到卡片的响应 **ATQA** 后，通过防冲突循环最终得到一张卡的序列号。最后，若接收到这张卡片的 **SAK** 响应，则表示已经完成与该卡的初始化通信连接（即选中该卡），可以继续进行下一步的密码校验过程了。

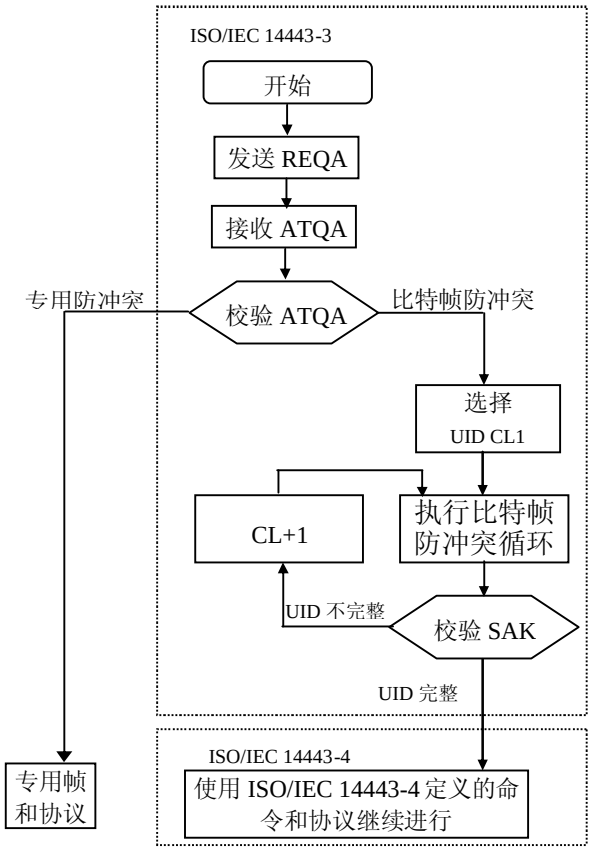


图 4-13 初始化和防冲突流程

流程图中，在读写模块发送了一个询卡 REQA 命令之后，所有在其工作范围处于空闲状态的卡片都会同步的以 ATQA 进行响应。ATQA 的格式见图表 4-5：

数据位 b7 和 b8 的编码表示序列号 UID 的类型，编码含义见表 4-6。数据位 b1、b2、b3、b4 和 b5 中若有一位被置 1，表示防冲突过程使用比特帧防冲突[16]。

表 4-5 ATQA 编码

MSB LSB

b16	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1
RFU				专有代码				UID 类型		RFU	比特帧防冲突				

卡片发送给读写模块的 SAK 是一个标准帧：包括一个字节的 SAK 代码和 2 个字节的 CRC 校验。其中，b3 位表示 UID 是否完整：若 b3＝0，表示卡片的 UID 已经被读写模块确认，若 b3＝1，表示还有部分 UID CLn未确认，n＝2 或 3。

表 4-6 b7、b8 位的编码

b8	b7	含义
0	0	UID 类型：4 字节
0	1	UID 类型：7 字节
1	0	UID 类型:10 字节
1	1	RFU

（4）初始化的具体实现

读写模块与 RFID 卡的通信的第一步，就是初始化询卡操作。在这之前，要使天线正常工作。关于天线，根据读写模块的硬件连接，需要对 MF RC500 的相关寄存器进行如下配置：

写$41 到寄存器 RxControl2：译码源来自内部；

写$5B 到寄存器 TxControl：使能 TX1 和 TX2 脚，设置调制源来自内部；

写$AD 到寄存器 BitPhase：设置天线的位相值；

写$00 到寄存器 MfOutSelect：设 MFOUT 引脚为低电平；

询卡操作的具体实现步骤如下：

① 相关寄存器的设置：

写$03 到寄存器 ChannelRedundancy：禁止 RxCRC、TxCRC,允许 parity 校验；

将寄存器 Control 的 Crypto1On 位置 0：禁止加密单元 Crypto1；

写$07 到寄存器 BitFraming，由于询卡命令是短帧格式，因此设置有效位为 7；

② 执行询卡命令

清 FIFO 缓冲区，向缓冲区写询卡命令代码：REQA=“52”或 WUPA=“26”；

写“Transceive”（代码为“1E”）指令到命令寄存器，向卡片发送询卡命令；

③ 接收卡片应答

读 FIFO 缓冲区，即接收卡片的应答数据

使用 REQA 命令询卡，也称为使用“ALL”模式询卡，该模式可以使工作范围内的所有卡片对其响应。使用 WUPA 命令询卡，叫做“IDLE”模式，该模式仅处在被挂起状态的卡片有效。读写模块对 MF1 卡执行询卡操作，接收卡片返回的 ATQA 为 2 个字

节，低字节为“04”，高字节为“00”。根据 ATQA 编码，MF1 卡的唯一序列号是 4 个字节，MF1 卡的防冲突使用比特帧防冲突算法。因为所有的 MF1 卡对询卡操作返回的 ATQA 都是这两个字节的内容，因此一般也称其为 MF1 卡的类型号 TagType。若询卡操作能正确得到类型号“$04$00”，说明在读写模块的工作范围内至少有一张 MF1 卡。

（5） 防冲突循环

ISO/IEC14443 协议使用的防冲突算法称作比特帧防冲突。对应于图 4-13 中“执行比特帧防冲突循环”的具体防冲突循环的流程如图 4-14 所示，算法步骤如下：

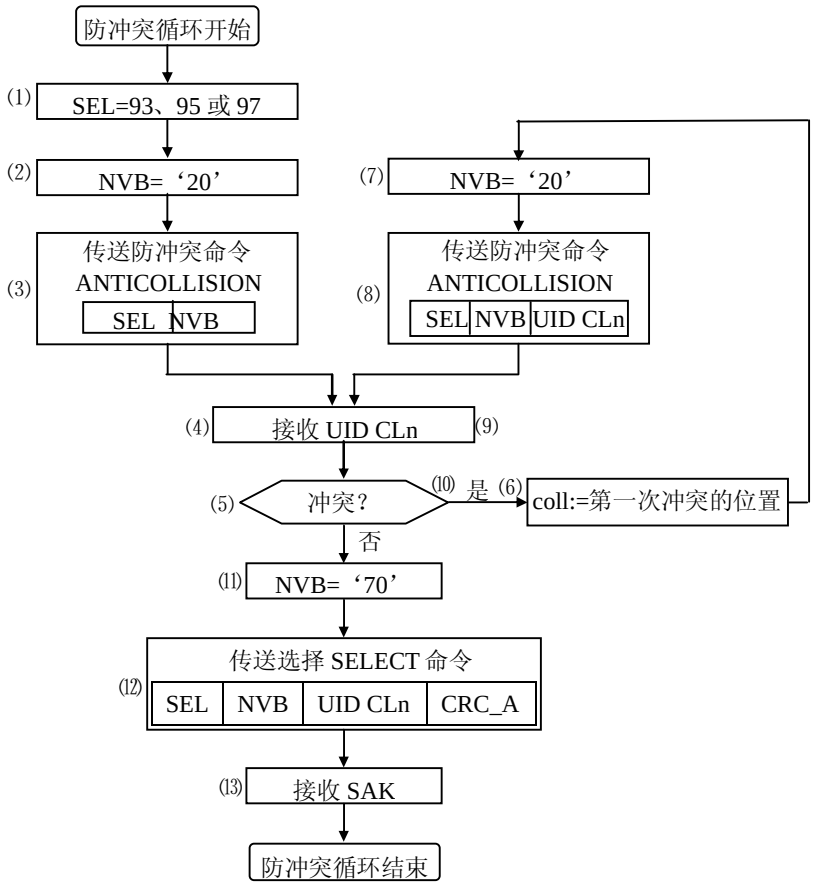


图 4-14 防冲突循环流程图

⑴ 读写模块根据唯一序列号 UID 的 CLn 指定防冲突命令 SEL 的代码。CL1、CL2、CL3 对应的 SEL 代码分别为“93”、“95”、“97”。

⑵ 读写模块指定 NVB 的值为“20”。该值表示 UID CLn 部分是 0 位，即不传送 UID CLn，而是让射频范围内的所有 RFID 卡以其完整的 UID Cln 响应。

⑶ 读写模块发送由 SEL 和 NVB 组成的防冲突命令。

⑷ 射频范围内的所有 RFID 卡以其完整的 UID CLn 响应。

⑸ 射频范围内的所有 RFID 卡都有唯一的序列号，那么当有不止一张的 PICC 进

行响应，就会发生数据位冲突。如果没有发生冲突，步骤 6 到 10 可以跳过。

⑹ 读写模块读冲突位置寄存器 CollPos 的第一个冲突发生的位置。

⑺ 读写模块指定 NVB 的值为接收到的有效数据位数。有效数据位是在冲突发生前接收到的 UID CLn的一部分，最后的数据位（即冲突位）可能是一个 “0”或“1”，读写模块默认其为 1。

⑻ 读写模块传送 SEL 和 NVB，和上述的有效数据位。

⑼ UID CLn与读写模块传送过来的有效数据位的那部分相同的 RFID 卡将其剩余的那部分 UID CLn 再传给读写模块。

⑽ 如果再次有冲突发生，重复步骤 6 到步骤 9。循环的最大次数可为 32。

⑾ 如果再没有冲突发生，指定 NVB 的值为“70”，说明读写模块将传送完整的 UID CLn。

⑿ 读写模块将传送 SEL 、NVB 及全部 40 位的 UID CLn，后面加 CRC_A 校验。

⒀ 与这 40 位 UID CLn匹配的 RFID 卡以其 SAK 响应（至此，图 4-14 的防冲突流程结束）。

⒁ 接下来（参见图 4-13），根据 SAK 判断唯一序列号 UID 是否完整，如果 UID 完整，则卡片被激活，初始化通信完成，可继续对其进行密码验证等操作；否则，读写模块将增加 CLn，继续进一步的防冲突循环。

下面举例说明上述初始化和防冲突过程。设在读写模块的工作 范围内有两张 RFID 卡，1 号卡的 UID 长度为 4 个字节，其中 UID0＝‘10’；2 号卡的 UID 长度为 7 个字节。读写模块与 RFID 卡之间的交互时序如图 4-15 所示，分三个阶段：

① 询卡（Request）

- 读写模块发 REQA 命令；
- 所有卡片均以 ATQA 响应：

  1 号卡的 ATQA 说明了其 UID 为四个字节，采用比特帧防冲突；2 号卡的 ATQA 说明其 UID 为 7 个字节，采用比特帧防冲突。

② 防冲突循环 Cascade Level1（CL1）

- 读写模块发送防冲突 ANTICOLLISION 命令：

  SEL=‘93’说明是 CL1；NVB=‘20’表示此次不发送 UID CL1 部分的数据。
- 工作范围内所有卡均以完整的 UID CL1 响应。
- 由于 1 号卡的 UID0＝‘10’和 2 号卡的级联标志 CT＝‘88’,两张卡在第一个字节的第 4 位就发生了第一次冲突。

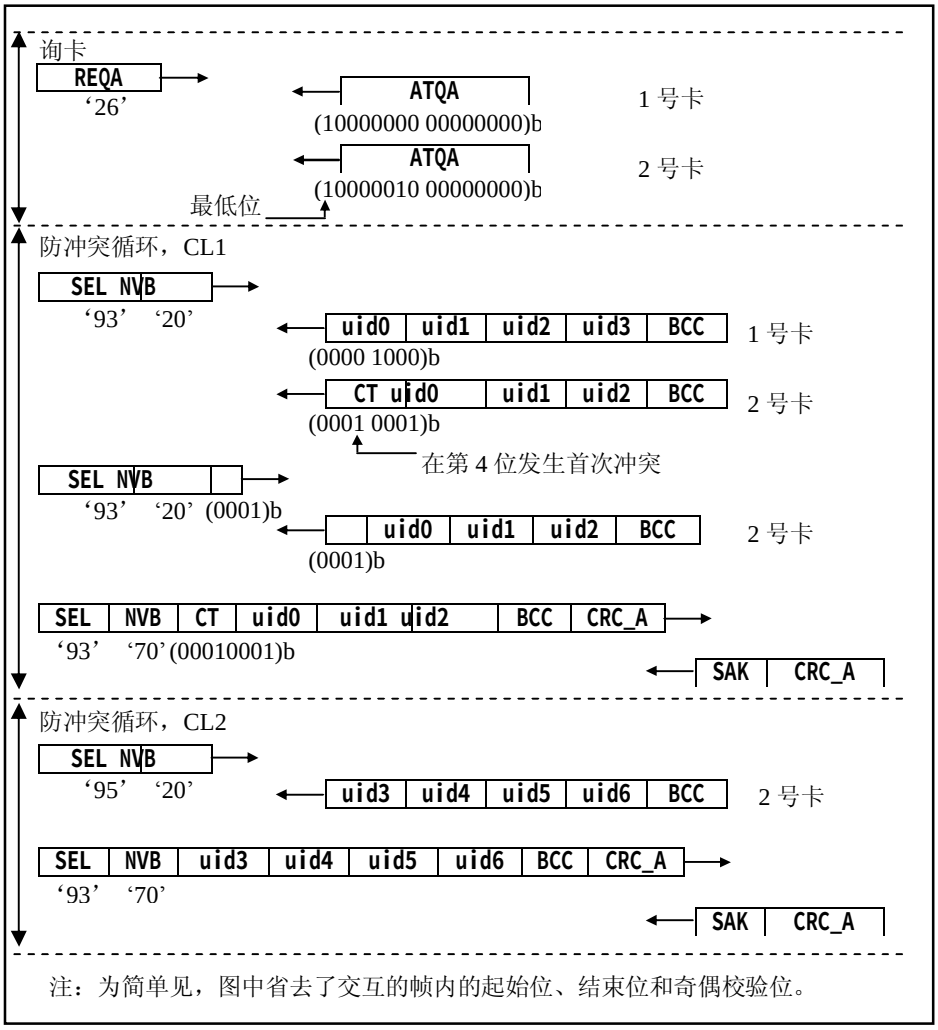


图 4-15 比特帧防冲突帧防冲突实例图

- 读写模块发送 **ANTICOLLISION** 命令：

  SEL＝‘**93**’,**UID CL1**部分包括冲突发送前的 **3** 位和冲突位（令其为‘**1**’），因此 **NVB**＝‘**24**’。

- 由于 **2** 号卡的前 **4** 位与读写模块发送过来的上述命令中的 **UID CL1**部分相同，因此，**2** 号卡响应读写模块的这个 **ANTICOLLISION** 命令，将其 **UID CL1** 的剩余 **36** 位全部发送给读写模块。

- 读写模块接收到了 **2** 号卡的 **UID CL1** 的全部数据，发送选择 **SELECT** 命令。

  命令组成：**SEL**＝‘**93**’,**NVB**＝‘**70**’,以及 **2** 号卡的完整 **UID CL1**。

- **2** 号卡向读写模块发送 **SAK** 应答，指出其 **UID** 不完整。

- 读写模块增加防冲突级别：**CL+1**。

③ 防冲突循环 **Cascade Level2**（**CL2**）

- 读写模块发送防冲突 **ANTICOLLISION** 命令

  SEL=‘**95**’说明是 **CL2**,**NVB**＝‘**20**’ 表示此次不发送 **UID CL2**部分的数据。

- 工作范围的所有卡以其完整的 UID CL2 响应上述命令，2 号卡向读写模块发送发送其 UID CL2。
- 读写模块接收到了 2 号卡的 UID CL2 的全部数据，向 2 号卡发送选择命令。命令组成：SEL＝‘95’,NVB＝‘70’,以及 2 号卡的完整 UID CL2。
- 2 号卡向读写模块发送 SAK 应答，指出 UID 已经完整。2 号卡被激活，即被读写模块选中，可以继续与读写模块进行下一步应用通信。

4.5.3 密码验证过程

（1） Mifare 的密码验证和加密机制

Mifare 内部的安全加密算法叫做 Crypto1，使用的密码长度是 48 bits，即 6 个字节。Mifare 卡中的数据，都有密码保护。只有使用正确的密码，才能成功进行卡密码校验，然后才能访问存储在 EEPROM 中的卡片数据。在按照 ISO14443A 协议成功选中一张卡后，用户若要继续访问该卡，就必须首先进行卡的密码验证。Crypto1 是一种三遍验证算法。MF RC500 内部将该算法进行了封装，只要执行 Authent1 和 Authent2 命令，就可自动完成这个验证过程。在卡片验证过程中，Crypto1 模块开始初始化。在验证通过之后，读写模块与卡片的通信信息都将被加密。

在验证命令期间，MF RC500 是从其内部密码缓冲区（key buffer）读取密码。因此，用户必须保证在执行 Authen1 命令前就已经将密码放到 key buffer 中了。

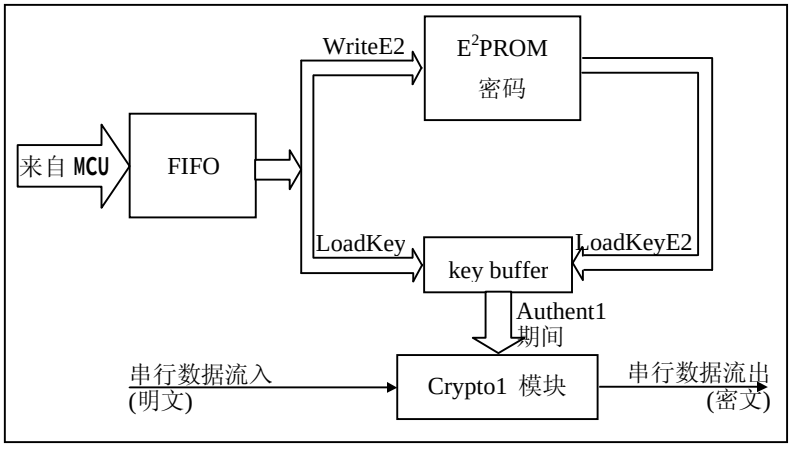


图 4-16 密码处理框图

将密码放到 key buffer有两种方式：使用 LoadKeyE2 命令从 EEPROM 中取得或者使用 LoadKey 命令通过 FIFO 直接从 MCU 获得。如图 4-16 所示。

MF RC500在其 EEPROM 中保留了 384 个字节的空间用来存储 Crypto1 密码，可以使用 WriteE2 命令向 EEPROM 中写入需要暂存的密码。每个密码 12 个字节，384 个字

节一共可以存储 **32** 个不同的密码。从 **EEPROM** 中取得密码载入到内部密码缓冲区，需要使用 **LoadKeyE2** 命令。

也可以直接使用 **LoadKey** 命令向内部密码缓冲区载入密码。

（**2**） 密码存储格式和密码模式

将密码载入到内部密码缓冲区或者暂存到 **EEPROM** 中时都要使用特定的密码存储格式。每个密码字节被分成两个半字节，低半字节（**k0-k3**）和高半字节（**k4-k7**）。每半个字节都在一个字节内存储了两次，其中一次要取反。这种特定的密码存储格式是正确执行 **LoadKey** 和 **LoadKeyE2** 命令的前提。

由于这种格式，用户使用的 **6** 字节的密码实际上需要 **12** 字节的 **E2PROM** 来存储。

密码存储格式如表 **4-7** 所示：

表 4-7 密码存储格式

	第 0 字节（最低字节）：A0		…	第 5 字节（最高字节）：A5	
密码数据位	$\overline{k7k6k5k4}$ k7k6k5k4	$\overline{k3k2k1k0}$ k3k2k1k0	…	$\overline{k7k6k5k}$ k7k6k5k44	$\overline{k3k2k1k0}$ k3k2k1k0
EEPROM 字节地址	n	n+1	…	n+10	n+11
密码(16 进制)	5A	F0	…	5A	A5

例如，若实际密码为“**0xA0 A1 A2 A3 A4 A5**”,实际存储到 **EEPROM** 和写到内部密码缓冲区中的密码的值为：“**0x5A F0 5A E1 5A D2 5A C3 5A B4 5A A5**”。

RFID 卡存储区有两种密码模式——**A** 密码或者 **B** 密码，可供用户用于不同目的。例如，读数据块、写数据块这两种不同操作可以分别使用两个 **A**、**B** 密码保护。**RFID** 卡对 **A** 密码进行验证的命令“**AUTHENT1A**”代码为“**60**”，对 **B** 密码进行验证的命令“**AUTHENT1B**”代码为“**61**”。

（**3**） 密码验证过程的实现

使用正确的密码进行有效的密码验证需要经过三个阶段：

第一步： 通过 **LoadKeyE2** 命令或 **LoadKey** 命令向内部密码缓冲区载入密码。

第二步： 开始 **Authent1** 命令。命令执行后，可通过密码错误标志位（寄存器 **ErrorFlag** 的第 **6** 位）来检查命令的执行情况。

第三步： 开始 **Authent2** 命令。命令执行后，可通过密码错误标志位（寄存器 **ErrorFlag** 的第 **6** 位）以及 **Crypto1On** 位（寄存器 **Control** 的第 **3** 位）来检查命令的执行情况。

下面以直接向内部密码缓冲区载入密码这种方法为例介绍读写模块实现密码验证的过程：

① 从内存取密码直接载入密码缓冲区：

先清 **FIFO** 缓冲区，然后将内存中暂存的密码（按特定格式存储，**12** 个字节）写到 **FIFO**

缓冲区；

写 **LoadKey** 命令到命令寄存器（**LoadKey** 命令代码为“**19**”），执行“直接载入密码”命令；

命令完成后，根据错误标志（**ErrorFlag**）寄存器的 **KeyErr** 位是否为 **1** 判断密码载入是否正确；

② 若密码载入正确，开始 **Authent1** 命令

写“**$07**”到寄存器 **ChannelRedundancy**，允许 **TxCRC**、**parity** 校验；

先清 **FIFO** 缓冲区，然后向 **FIFO** 缓冲区写入参数：包括密码模式（**A** 密码模式或 **B** 密码模式）、存储区号码、卡片序列号；

写 **AUTHENT1** 命令到命令寄存器（**AUTHENT1** 命令代码为“**0C**”），执行 **Authent1** 命令

根据错误标志（**ErrorFlag**）寄存器的 **KeyErr** 位是否为 **0** 判断 **AUTHENT1** 命令是否执行正确；

③ 若 **Authent1** 执行正确，开始 **Authent2** 命令

写“**$03**”到寄存器 **ChannelRedundancy**，禁止 **RxCRC**、**TxCRC**，允许 **parity** 校验；

写 **AUTHENT2** 命令到命令寄存器（**AUTHENT2** 命令代码为“**14**”），执行 **Authent2** 命令；

命令完成后，根据错误标志（**ErrorFlag**）寄存器的 **KeyErr** 位是否为 **0**、**Control** 寄存器的 **Crypto1On** 位是否为 **1** 判断 **AUTHENT2** 命是否执行正确；

若 **Authent2** 命令执行正确，则通过了对该卡的存储密码的校验，可以进一步对卡片的存储区进行操作。另外，在经过密码验证之后，读写模块与 **RFID** 卡之间的任何通信数据都会经 **Crypto1** 单元进行加密处理。

4.5.4 对 MF1 存储区的操作

（**1**） **MF1** 卡的访问存储器命令

MF1 卡可以根据下列命令对存储器进行操作：

- **READ** 读存储区的一个数据块；
- **WRITE** 写存储区的一个数据块；
- **DECREMENT** 对存储在数值块中的数值做减法操作，并将结果存到数据寄存器；
- **INCREMENT** 对存储在数值块中的数值做加法操作，并将结果存到数据寄存器；
- **TRANSFER** 将数据寄存器的内容写入数值块；
- **RESTORE** 将数值块内容存入数据寄存器；

（2） MF1 卡的数据块

MF1 共有 16 个不同的存储区，每个存储区分成 4 个块，3 个数据块以及一个控制块。MF1 卡的数据块有两种：普通数据块和数值块。

普通数据块用于存储一般的 16 字节数据，仅可以执行 READ、WRITE 命令。

数值块是特殊的数据块，专门用来存储数值，不仅可以对其执行普通的 READ、WRITE 命令，还可以执行用于实现电子钱包功能的命令：增值 INCREMENT、减值 DECREMENT、传送 TRANSFER 和重存 RESTORE。存储在数值块中的数值为四个字节（包括符号位），按特定格式存储。设数值为 ABCD（4 个字节），数值块的地址（即块号）为 addr，则存储格式为：“ABCD（4 个字节）＋ABCD 的反码（4 个字节）＋ABCD（4 个字节）＋addr（1 个字节）＋addr 的反码（1 个字节）＋addr（1 个字节）＋addr 的反码（1 个字节）”，参见图 4-17。例如，若要将块号为 5 的数值块赋数值 0，则用 WRITE 命令向数据块中写入下列数据：“00 00 00 00 FF FF FF FF 00 00 00 00 05 FA 05 FA”(16 进制)。

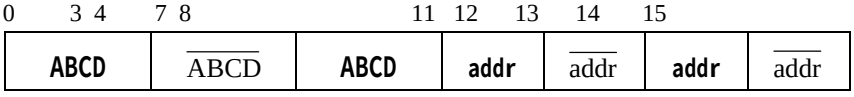


图 4-17 数值块存储格式

数值块中的内容第一次由 WRITE 命令写入后，以后可用 INCREMENT、DECREMENT 和 RESTORE 命令修改内容，结果暂存在其内部的 DATA 寄存器（数据寄存器）中，然后用 TRANSFER 命令再重新写回数值块。

（3） 对 MF1 卡存储区数据块的操作

读写模块对卡片数据块操作的实现过程如下。

读数据块的操作步骤为：

① 设置寄存器 ChannelRedundancy($22)值为$0F,允许 RxCRC,TxCRC,Parity 校验；

② 设置 DecoderControl 寄存器,将第 6 位 RxMultiple 置 1（接收 1 帧以上数据）；

③ 清除 FIFO 缓冲区，设置寄存器 FIFOLevel 值（>16）；

④ 向 FIFO 缓冲区写参数：1.“读 16 字节数据块”的命令代码（$30），2.块号；

⑤ 向命令寄存器写 TRANSCEIVE 命令，发送 READ 命令代码及块号给卡片

⑥ 等待指令完成，读 FIFO 缓冲区接收 16 字节数据

写数据块的操作步骤为：

① 设置寄存器 ChannelRedundancy($22)值为$07，即禁止 RxCRC,允许 TxCRC,Parity 校验；

② 清除 FIFO 缓冲区，向 FIFO 缓冲区写参数：1.“写 16 字节数据块”的命令代码（$A0），

2.块号；

③ 向命令寄存器写 TRANSCEIVE 命令，发送 WRITE 命令代码及块号给卡片；

④ 清除 FIFO 缓冲区,设置寄存器 FIFOLevel 值（>16）；

⑤ 将要写入数据块的 16 字节数据写入到 FIFO 缓冲区；

⑥ 向命令寄存器写 TRANSCEIVE 命令，将数据发送给卡片；

⑦ 延时等待命令完成。

对数值块的增/减值操作步骤为：

① 设置寄存器 ChannelRedundancy($22)值为$07，即禁止 RxCRC,允许 TxCRC,Parity 校验；

② 清除 FIFO 缓冲区,向 FIFO 缓冲区写参数:1.命令代码(增值 INCREMENT 代码为‘C1’，减值 DECREMENT 代码为‘C0’），2.块号；

③ 向命令寄存器写 TRANSCEIVE 命令，发送增/减值命令代码及块号给卡片；

④ 清除 FIFO 缓冲区,将增/减值内容（4 个字节）写入到 FIFO 缓冲区；

⑤ 向命令寄存器写 TRANSCEIVE 命令，将数据发送给卡片；

⑥ 延时等待命令完成

⑦ 清除 FIFO 缓冲区，向 FIFO 缓冲区写参数：1.传送命令 TRANSFER 代码（‘B0’），2.块号；

⑧ 向命令寄存器写 TRANSCEIVE 命令，将传送指令代码及块号发送给卡片；

⑨ 延时等待命令完成。

4.6 读写模块的接口函数

4.6.1 读写模块的底层通信函数

按与 MF1 卡片的通信流程,将上述与卡片的底层通信操作使用函数的形式封装起来,就是读写模块的底层通信函数(或称低级函数）.包括有:询卡函数 mf_request(),防冲突函数 mf_anticoll(),选中卡片函数 mf_select(),密码验证函数 mf_authen(),读数据块函数 mf_read(),写数据块函数 mf_write(),增值函数 mf_increment(),减值函数 mf_decrement(),卡片挂起函数 mf_halt()。底层通信函数的具体说明请见附录 4。调用底层通信函数的流程图如图 4-18 所示。

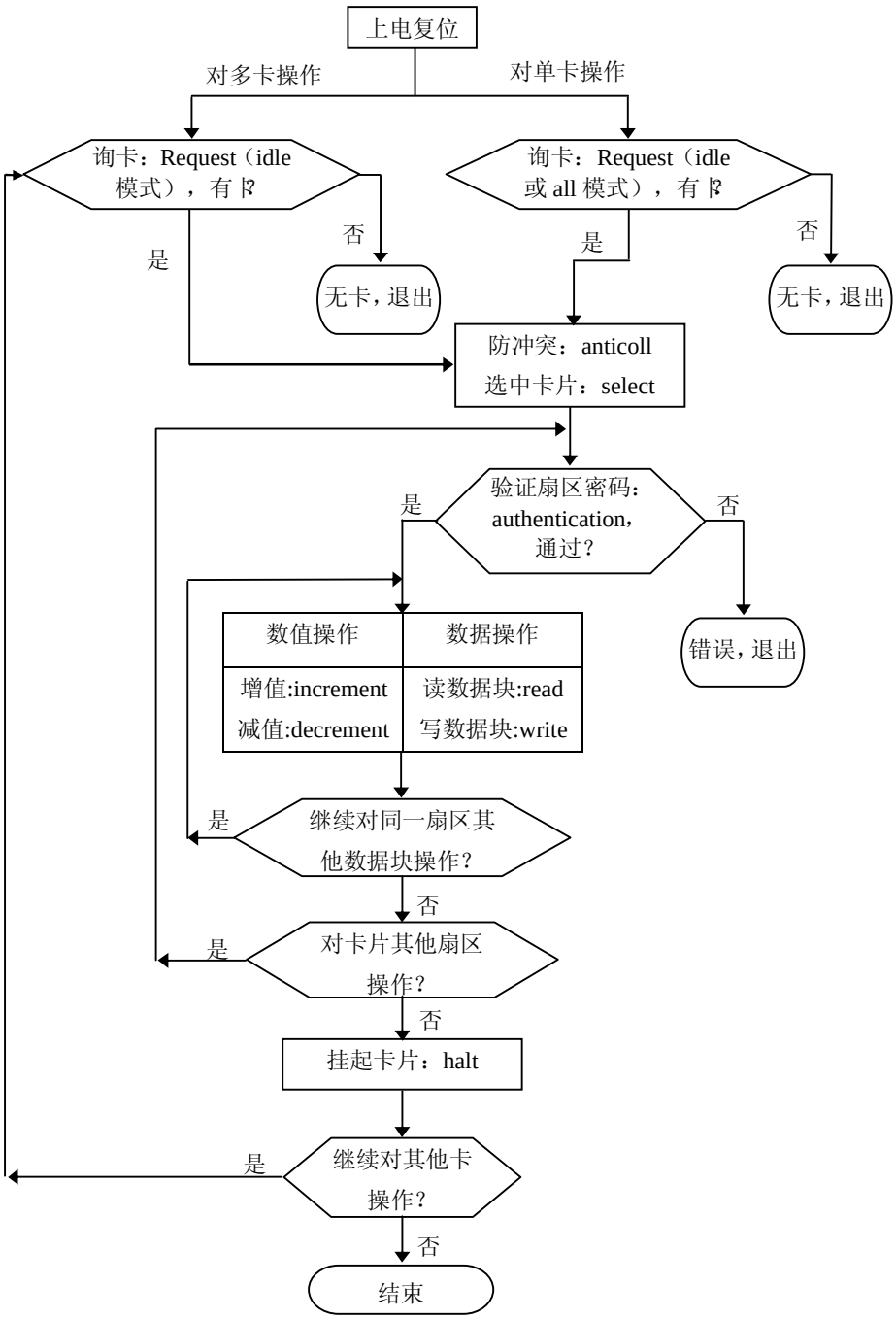


图 4-18 MF1 卡低级函数调用流程

4.6.2 读写模块的高级接口函数

为了方便用户，在上述底层通信函数的基础上将对 **RFID** 卡的操作按统一的接口形式封装起来，就构成了高级函数。每个高级函数都集成了一系列低级函数。

① 得卡的序列号函数：**CardGetSn**（）,询卡并返回选中卡片的序列号。**CardGetSn**（）集成了三个低级函数：询卡—**>**防冲突**->**选中。

② 读卡片数据块函数：**ReadCard**（），选中一张卡并通过验证后读 **1** 个数据块

内容。**ReadCard**（）函数的操作流程是：询卡－>防冲突->选中->密码验证－>读数据块－>挂起。

③ 写卡片数据块函数：**WriteCard**（），选中一张卡并通过验证后写 **1** 个数据块内容。**WriteCard**（）函数的操作流程是：询卡－>防冲突->选中->密码验证－>写数据块－>挂起。

④ 电子钱包初始化函数：**FormatPurse**（），将卡片的某个数据块初试化为数值块。**CardInitValue**（）的操作流程是：询卡－>防冲突->选中->密码验证－>写数据块（按数值块格式要求）－>挂起。

⑤ 电子钱包增值函数：**Increase**（），将卡片的某个数值块增值。**Increase**（）的操作流程是：询卡－>防冲突->选中->密码验证－>增值－>传送－>挂起。

⑥ 电子钱包减值函数：**Decrease**（），将卡片的某个数值块减值。**Decrease**（）的操作流程是：询卡－>防冲突->选中->密码验证－>减值－>传送－>挂起。

注：关于高级函数的具体封装说明请见附录 **3**。

用户使用高级函数将要方便很多，但有时速度会比直接使用低级函数慢很多，因为高级函数中有许多低级函数的执行是重复的。例如，当要读卡片的某个扇区的 **3** 个数据块，使用高级函数需调用三遍读卡片数据块函数 **ReadCard** 来完成，即要执行 **3** 遍的询卡、防冲突、密码校验操作。由于同一扇区的密码相同，如果调用低级函数，只需要执行一次密码验证就可以对该扇区进行重复操作。因此，在读写模块软件中，低级函数和高级函数的接口都为用户开放，用户可根据实际需要选择调用。

第五章 应用实例

有了封装好的读写模块，用户不需要了解射频读写芯片 MF RC500、RFID 卡 MF1 的工作原理，就可以很方便地在应用系统中使用读写模块完成对 MF1 卡的所有控制及读写操作，并且基于读写模块开发自己的读写卡器或者其他 RFID 卡应用产品。

本章给出了两个读写模块的应用实例：第一个应用实例，是在读写模块的软硬件基础上开发具有汉字液晶显示功能的通用读写卡器。该通用读写卡器可直接集成于应用系统中，高层开发人员只需调用 PC 方的库函数就可完成对卡片的读写。第二个应用实例，是将读写模块软件移植到带有网络接口的 RFID 卡考勤机中，用户刷卡后的签到信息通过网络接口直接传送到远端服务器。

5.1 通用读写卡器

5.1.1 通用读写卡器系统组成

RFID 卡读写设备（读写卡器）是 RFID 卡广泛应用的关键。使用GP32 作为主控 MCU 开发的读写模块使用方便、价格低廉，可以广泛应用于开发低端读写卡设备。下面介绍在读写模块的基础上开发的通用读写卡器。

读写卡器的系统组成如图 5-1 所示，在读写模块的基础上继续扩展了液晶（LCD）显示模块和电源模块。读写模块可以实现对符合 ISO/IEC 14443A协议的 RFID 卡的读写操作，且由于读写模块的 MCU 中有芯片的监控程序，因而可以实现读写器在线升级。为了在读写卡过程中显示当前的操作状态、操作内容以及卡中的信息，在读写卡器中开发了汉字点阵液晶 GXM12232 接口驱动，支持连接 LCD 显示模块。读写卡器使用串行通信方式与上位机（PC）通信。电源供电方式既可以使用 5V 电源模块，也可以使用 USB 接口从 USB 取电。

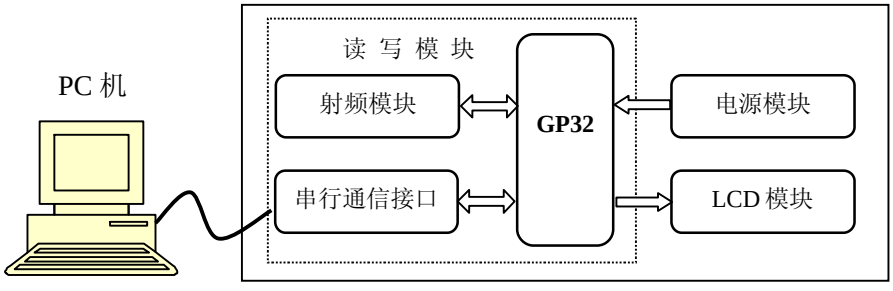


图 5-1 读写卡器系统组成

5.1.2 通用读写卡器硬件说明

通用读写卡器的硬件结构建立在读写模块的基础上，读写模块的 MCU(GP32)同时也是读写卡器的控制 MCU。GP32 SDIP42的 31 个通用 I/O 引脚，读写模块使用了 B 口（8 个）、C 口（5 个）以及 PTE1 脚共 14 个 I/O 引脚。还有 17 个 I/O 脚在读写卡器中有两个用途：一是连接控制读写卡器的液晶显示模块，一是用来控制读写卡器的电源指示、刷卡提示等发光二极管（LED）和蜂鸣器。下面重点介绍读写卡器的液晶显示接口。

（1）LCD 模块接口

GXM12232LCD 汉字液晶模块可显示汉字或字符。由于 GXM12232LCD 不带字库，故要显示的汉字、字符、图形的点阵字节须由主控芯片发给 LCD。GXM12232LCD 的屏幕大小有两种：32×122 点阵和 32×160 点阵。在32×160 的 LCD 上，若显示 16×16 点阵的汉字，则一行最多可显示 7 个汉字，最多可显示两行。一般情况下，可以将点阵字节事先存储在主控芯片的 ROM 或 Flash 中，当需要显示时，由主控芯片将点阵字节发往 LCD 显示。

读写卡器采用 MC68HC908GP32 作为主控芯片，考虑到 Flash 的大小（32K 字节），使用 512 字节存放显示的点阵内容，可以存放 16 个 16*16 点阵的汉字字模。(一个 16*16 点阵的汉字字模需要 16*16/8=32 字节，故 512 字节总共可存储 16 个汉字(512/32=16)。

液晶显示采用的 GXM12232 汉字 LCD 模块。模块封装图见图 5-2。

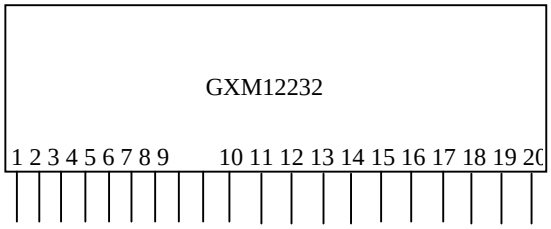


图 5-2 LCD 封装图

引脚说明：

【1 脚】Vss：地；

【2 脚】Vcc：电源；

【3 脚】Vo：参考电压；

【4 脚】A0：数据/指令通道选择引脚；

【5 脚】NC（保留未用）；

【6 脚】R/W E2：屏幕后半区域选择脚；

【7 脚】NC（保留未用）；

【8 脚】/RD(E1) ：屏幕前半区域选择脚；

【9 脚】R/W：读/写选择引脚；

【10 脚～17 脚】：DB0～DB7，数据信号引脚；

【18 脚】RES：时序选择引脚；

【19 脚】LEDA：LCD 背光驱动引脚；

【20 脚】LEDK：地。

（2） GP32 与 LCD 的连接

GP32 与 LCD 的连接如图 5-3。对读写模块的 操作与显示数据的更新不是同时进行，因此 GP32 的数据口 PTB 可以复用：PTB0～PTB7 接 LCD 的 DB0～DB7,传送数据信号。

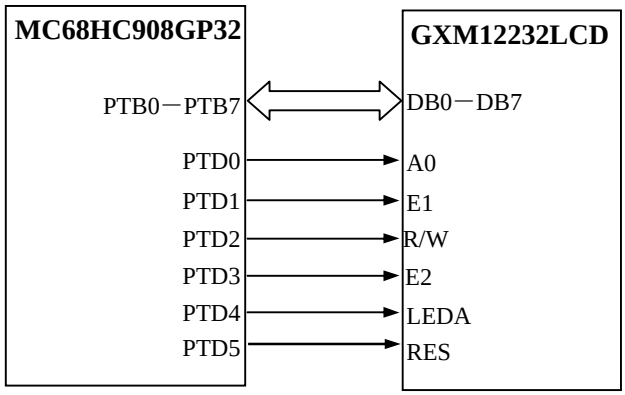


图 5-3　GP32 与 LCD 连接示意图

PTD0 接 LCD 的 A0：控制数据/指令通道选择；PTD1 接 LCD 的 E1：控制选择屏幕前半区域；PTD2 接 LCD 的 R/W：读写控制；PTD3 接 LCD 的 E2：控制选择屏幕后半区域；PTD4 接 LEDA，控制 LCD 的背光驱动；PTD5 接 RES，控制时序选择。

（3） 通用读写卡器硬件电路原理图

通用读写卡器的硬件电路原理图参见图 5-4。

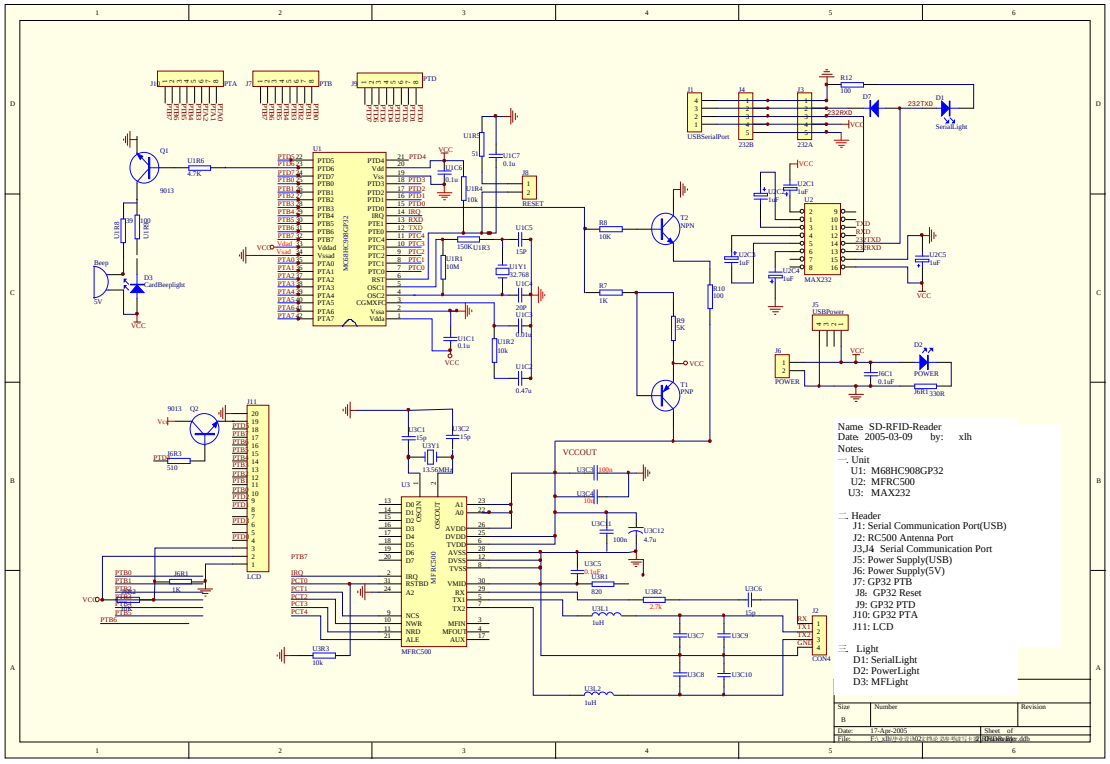


图 5-4　通用读写卡器硬件电路原理图

5.1.3 通用读写卡器 MCU 方程序

通用读写卡器 MCU 内除固化的读写模块的对卡片操作函数外，为实现完整的读

写器功能还有 LCD 液晶显示函数、蜂鸣器函数、LCD 背光驱动函数等函数接口。

如图 5-5 所示,通用读写卡器系统通信协议 MCU 方设计主循环流程如下:

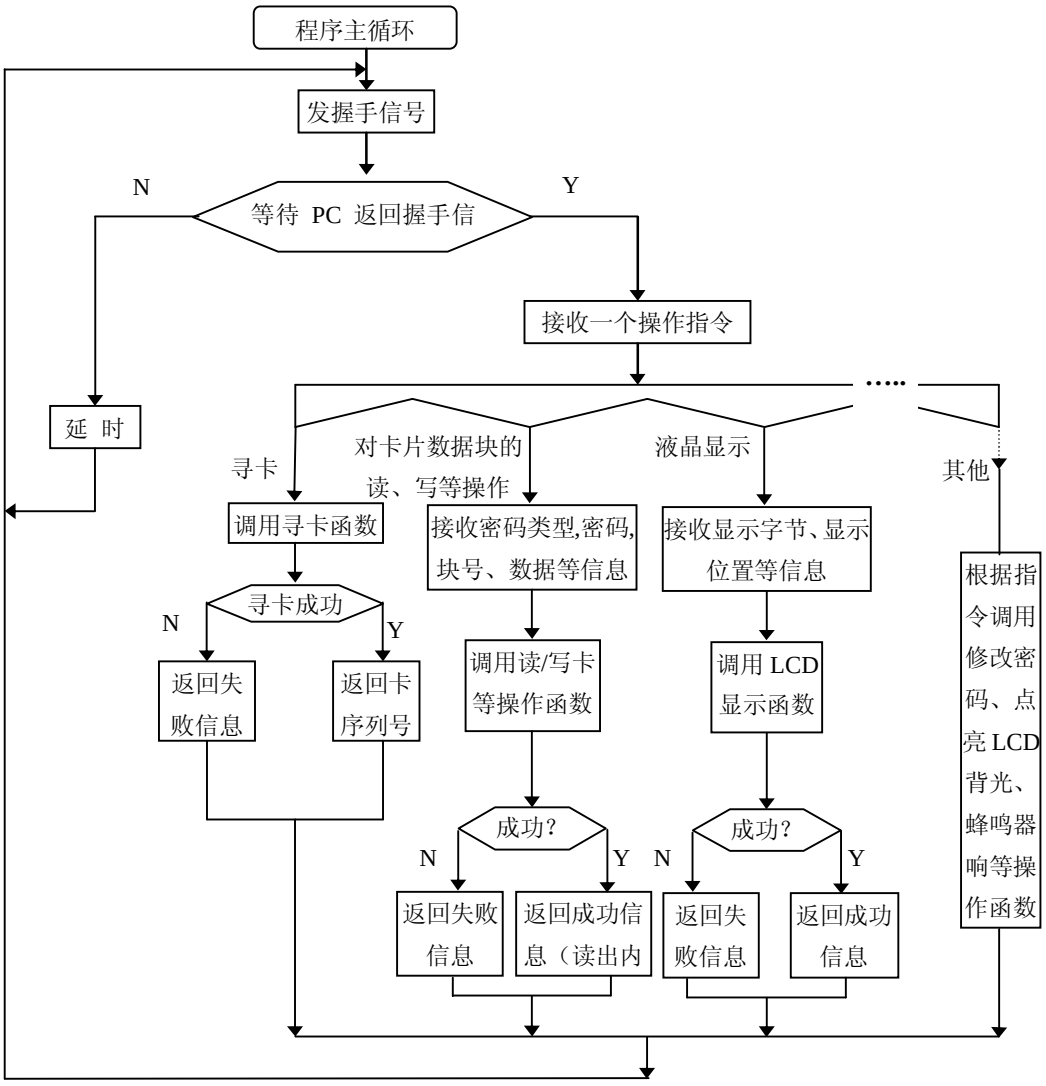


图 5-5 通用读写卡器通信协议 MCU 方主循环流程设计

① 读写卡器主动发送一个字节的握手信号。

② 等待接收一个字节的握手返回信号,收到返回的握手信号转 3,否则发送无卡信息并延时后转 1。

③ 接收一个字节卡的操作指令及其后的操作参数,若为无效信息则跳转至 1,若为有效指令转至相关处理程序。

④ 若为寻卡指令，则调用寻卡子程序，寻卡成功返回卡的序列号信息，否则返回无卡信息，转 1。

⑤ 若为其他对卡片操作的指令，则调用相应读卡、写卡、电子钱包增值、电子钱包减值、电子钱包初试化等读写模块的函数，操作不成功返回失败信息给 PC，

若成功则返回相关信息（例如读出的数据等）给 PC。

⑥ 若为对 LCD 操作的指令，则调用 LCD 显示函数，操作不成功返回失败信息给 PC，若成功则返回成功信息给 PC。

⑦ 其他指令，例如修改密码、驱动 LCD 背光、蜂鸣器响等，根据指令调用相关函数，然后返回操作结果给 PC。

5.1.4 通用读写卡器 PC 机方函数库

为方便使用，在PC 机方为用户提供了配套的 VB 模块函数，包括对读写卡器的各个功能模块的操作。用户只需在 PC 机端直接调用 VB 模块函数，就可以将读写卡器无缝地接入到各种不同的应用系统中。该读写卡器不是专为某个系统定制，适用范围广泛，因此称其为通用读写卡器。

VB 模块函数库中，相应的提供有卡操作函数、液晶操作函数（包括 LCD 背光驱动函数）、蜂鸣器函数、串口操作函数。图 5-6 是调用读写卡器 VB 模块函数完成对 MF1 卡片数据读、写、修改密码等操作的演示系统界面。

图 5-6 读写卡演示界面

图 5-7 通用读写卡器实物图

5.1.5 通用读写卡器应用

通用读写卡器为应用系统用户提供了一套完整的 RFID 卡操作实现方案，用户不需要再花精力、时间开发 RFID 卡的读写设备及相关程序，只需在 PC 方直接调用卡片操作的库函数即可。

作者开发的通用读写卡器已经应用于上海玛亚克软件公司开发的党员信息化管理系统中（参见图 5-7）。该系统使用 Mifare 1卡做为党员证件卡，用于会议的刷卡签到、记录党费交纳等用途。

5.2 带有网络接口的考勤机

本应用实例的设计思想是将读写模块应用于带有网络接口的考勤机，用户刷卡后的签到信息通过网络接口直接传送到远端服务器。通过本实例来体现作者所设计的 RFID 卡读写模块的通用性、可移植性。

5.2.1 嵌入式网络接口技术

嵌入式网络接口技术[28]是在嵌入式系统中添加网络接口，从而实现网络数据传输。通过这种方式，嵌入式系统的主控 MCU 所采集的数据就可以及时发送到网上。作者所在实验室已经实现了 8 位 MCU 的嵌入式网络接口技术[29]，其核心思想是在主控 MCU 中编写实现 uIP 协议的程序，通过以太网控制器与局域网相连。

5.2.2 读写卡模块和嵌入式网络接口的结合

（1）主控 MCU 的选取

GP32 最多只能提供 33 个 I/O 引脚，与读写芯片的连接已经占用了 14 个 I/O 引脚，还剩下 19 个引脚进行网络连接是不够的。而且由于涉及到嵌入式网络设计，对 MCU 资源要求比较苛刻，需要容量比较大的 RAM 以及足够的数据程序存储空间[30]。Freescale 新近推出的 MC9S08GB60（以下简称 GB60）可以满足这种需求。GB60 具有 56 个 I/O 引脚，4KB 的 RAM，60KB 的 FLASH 存储空间，2 个定时器模块，其总线率频高达 20MHZ[31]。

（2） 硬件接口设计

在这个应用中，以主控芯片 GB60 为核心，需要实现数据的采集和数据的网上传输。数据的采集部分就是前面所设计的读写模块，和前面所讲述的具体设计的差别在于选取的主控 MCU 不同。数据传输部分采用实验室中比较成熟的以太网接口设计方案。以太网接口芯片选用 Realtek 公司的 RTL8019AS[32]，整个应用系统的逻辑框图如图 5-8 所示。

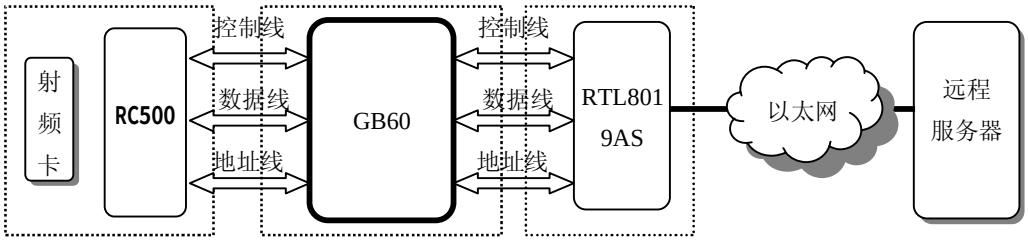


图 5-8 带网络接口考勤机系统逻辑框图

根据以上的设计方案，GB60 的 I/O 口进行如下分配：

① GB60 和 RC500 的连接

GB60 的 PTB0~PTB7 用于数据线和地址线(二者复用)。

GB60 的 PTE2～PTE7 用做控制线，其中PTE2～电源控制、PTE3～读控制、PTE4～写控制、PTE5～使能端、PTE6～片选、PTE7～复位控制。

② GB60 和 RTL8019AS 的连接

GB60 的 PTC0～PTC7 和 PTD0～PTD7 用于数据线(16 位数据传输)。

GB60 的 PTF0～PTF4 用于地址线。

GB60 的 PTG4～PTG7 用于控制线，其中 PTG4 为读控制、PTG5 为写控制、PTG6 为地址使能端、PTG7 为复位控制。

在这个种硬件设计方案中，使用了硬件模块化的设计思想。图 5—7 中主控芯片 GB60 的左边部分是射频读写芯片，右边是以太网接口部分。设计电路板时，将主控芯片 GB60 的 PTB 和 PTE 口连接到一个连接口上，再通过排线和射频读写芯片部分连接起来就实现了两部分的连接；将主控芯片的 PTC、PTD、PTF 和 PTG 的相关口合理安排，同样通过排线和以太网接口部分连接起来就实现这两部分的连接。通过这种方法，某个局部的硬件变动对整个系统的改变就会减少到最低限度，从而提高了硬件设计的重用性。

5.2.3 关健技术说明

（1） 射频读写模块与硬件相关的软件修改

射频读写模块在设计 时的主体思想是通用性，在不同的应用中，根据不同的需要做尽量少的修改，以适应新的应用系统。

首先需要进行编程环境的设置，在 SD-1 的编译环境中设置 GB60 相关的地址参数，其界面如图 5-9 所示。

图 5-9 主控 MCU 与硬件相关参数设置界面

其次，修改头 GB60 和 RC500 接口的头文件 MCU_RC500.H，适应新的连接口。

```
*-------RC500 和 GB60 硬件接口引脚定义---------------------*

RC500Data      EQU PTB

RC500DataD     EQU DDRB

RC500DataPUE   EQU PTBPUE

RC500CtlD      EQU PTE
```

```
RC500CtlDD    EQU DDRE
RC500_Power   EQU 2
RC500_NRD     EQU 3
RC500_NWR     EQU 4
RC500_ALE     EQU 5
RC500_NCS     EQU 6
RC500_RSTPD   EQU 7
*-----------------------------------------------------*
```

进行了以上两项操作后，原读写模块软件就可以适应新的主控 MCU GB60，完成 RFID 卡的读写操作。

（2） 用户刷卡操作

用户使用 RFID 卡进行刷卡签到，由 GB60 调用读写模块函数来实现。在经过系统确认后（系统密码校验），GB60 从卡片的某个特定数据块（由应用系统决定）获取员工的 ID 号(10 字节)，并将 ID 号和当前的系统时间(小时-分钟-秒，3 字节)保存在 RAM 缓冲区中。同时，GB60 将当前的系统时间写入用户卡片的特定数据块。这样，如果在数据处理或网络传输过程中数据出错，可以通过重读用户卡的时间数据块中的数值来确认用户的刷卡时间。

（3） 数据上传到服务器的操作

考勤机本质是一个嵌入式系统，其资源有限，不能对用户的刷卡信息做长期存储，也不能暂存大批量的用户刷卡信息，所以需要在合适的时候要将数据传送到远程服务器。

如果采用“刷卡à数据上传”模式，可以及时将数据传送到远程服务器，但这也不是一个最理想的选择。一次刷卡，产生的数据量很少，只有 13 字节，但组装成以太帧需要添加以太帧头、TCP/IP 头，有些浪费，同时这种频繁的数据发送，需要占用主控 MCU，影响刷卡的及时性响应。同时考虑到实际情况：上班高峰期，用户刷卡需要及时响应，只要准确记录了用户上班时间，数据上传可以不需要立即传送，而是到了一定的刷卡量或是空闲了一段时间再进行数据上传。所以作者在实现考勤机实例时，数据上传的时机选择在 100 个用户刷卡或上一次用户刷卡后空闲 10 分钟时，启动网络接口，进行数据上传。

（4） RAM 存储器分配问题

基于上述的刷卡及数据传输的协调性思想，可以确定GB60的内存分配。GB60内存总共4K。从$0080~$00FF是直接页，对该页的访问效率最高，因此一般在该页存放使用频率高的数据。例如，日期和时间的相关数据、虚拟寄存器的内存区等数据空间要分配到该页。

一次用户刷卡占用13个字节，则100个用户刷卡所占用的空间就是1300个字节，所以刷卡数据缓冲区分配1300字节；1个以太帧信包最大字节数1513个字节[33]，同时为了保存TCP/IP连接信息，所以给网络数据缓冲区分配2000个字节；读写模块的数据交换需占用64字节。

表5-1 应用系统的内存分配映像表

地址	字节数	用途
$0080 ↓ $00FF	直接页,128字节	虚拟寄存器的内存区日期和时间
$0100 ↓ $01FF	256字节	保留
$0200 ↓ $0239	64字节	读写模块用于数据交换
$0240 ↓ $0753	1300字节	刷卡数据缓冲区
$0754 ↓ $0F23	2000字节	网络数据缓冲区
$0F24 ↓ $107F	348字节	堆栈

依据上述思想进行RAM分配的内存映像图如表5-1所示。

（5） 定时器的使用

考勤机本身需要维持一个系统时间。在考勤机上电工作时，和远程服务器进行一次信息交互，获取服务器时间，然后启用主控MCU本身的定时器事件，定时中断的时间为1秒，中断过程更新时间。

5.2.4 服务器方测试软件

服务器方配合考勤机的软件主要完成两方面的功能：

（1） 接收考勤机的数据。

（2） 保存数据到数据库。

这里对服务器方软件就不作详细介绍了。使用该软件对考勤机进行测试的演示界面如图5-10所示。

图5-10 考勤系统服务器程序运行界面

第六章 总结

射频识别卡由于使用方便、交易速度快、便于维护和使用寿命较长等优点，正在各种场合逐渐替代目前广泛使用的接触式 IC 卡，应用前景十分广阔。

本文基于 Freescale 的新型 8 位 MCU 和射频读写芯片 MF RC500设计实现了可供用户二次开发的 RFID 卡读写模块。该读写模块能完成对 RFID 卡的控制和读写操作，使用户无需了解射频识别技术的细节，就可将其集成到应用系统中。同时读写模块还具有在线编程特性，因此可以很方便地供用户基于读写模块继续开发 RFID 卡应用产品。读写模块大大方便了开发人员对 RFID 卡读写技术的掌握，对推广RFID 卡有着重大意义。

本文介绍了读写模块的设计实现过程——包括硬件设计和软件设计。硬件设计包括硬件的选型、硬件电路的设计、硬件电路的测试等内容。软件设计包括实现在线编程特性的 MCU 监控程序、与RFID 卡通信的底层通信函数以及供用户调用的 RFID 卡操作函数等内容。

本文还给出了读写模块的两个应用实例。一个是在读写模块的基础上继续扩展实现通用读写卡器，该读写卡器已经应用于上海玛雅克软件公司开发的 RFID 卡管理系统中。另一个应用是升级读写模块的主控 MCU，将读写模块软件移植到性能更优的 GB60 上，开发带网络接口的考勤机。

然而，由于时间等因素所限，对射频识别卡读写技术的研究还有继续深入的地方。例如：

(1) 为读写卡设备加装键盘、实现键盘输入；

(2) 进一步研究实现 TypeB 型卡的读写技术；

(3) 研究遥耦合射频识别技术，实现 ISO/IEC 15693 协议。

作为倍受瞩目的新技术，射频识别技术、射频标签在未来将深入到人们工作、生活的各个方面。本文的研究在整个射频识别领域中只是很小一部分，但是相信在众多研究工作者、开发人员的共同努力下，必将推动射频识别技术在我国的应用和进一步发展！

致 谢

在论文完成之际，首先要感谢我的导师王宜怀教授。本文从前期选题，到实验过程，到最后论文完成，都是在导师的悉心指导和严格要求下进行的。从导师渊博的知识、严谨的治学态度和忘我的工作精神中，我学会了做学问、做事和做人的重要态度和基本方法，将使我终身受益。在此，向导师及其夫人张建英老师表示最诚挚的谢意！

特别感谢我的师兄刘晓升老师！这三年，从入门到毕业，他给了我无数的建议和帮助。本课题的顺利完成，离不开他的支持与指点。

同时，感谢共同学习的汤龙梅、蒋健武，他们踏实、勤奋的态度一直都是我的榜样，他们对我的论文提出了许多有益见解。感谢师弟帅辉明、田宏伟、郭继伟与师妹郑宏静、刘雪兰等人在研究过程中的帮助。还要感谢我的室友林霞、蒋萍、杨蓓红，她们在生活中给予我很多热心帮助。

深深感谢我的母亲，她无私的付出、无微的关心一直以来是我最大的动力，她给了我面对所有困难的勇气。

最后，再次向所有的老师、同学、朋友所给予的关心、帮助与鼓励致谢！

参考文献

[1] 游战清，李苏剑. 无线射频识别技术（RFID）理论与应用[M]. 电子工业出版社，2004

[2] 王爱英. 智能卡技术－IC卡. 清华大学出版社[M],2000

[3] 贺利芳，范俊波 .非接触 IC 卡技术及其发展和应用 [J]. 通信与信息技术 ，2003,(6)

[4] 杨肇敏，张忠全.初论非接触IC卡技术[J]. 计算机工程与应用，1999,35(12)，P44-47

[5] 边红丽.非接触IC卡技术及应用漫谈[J]. 金卡工程，2002,(6),P37-38

[6] [德]Klaus Findenzeller. 射频识别技术[M]. 电子工业出版社，2001

[7] 边红丽.非接触IC卡技术应用趋势[J]. 金卡工程，2003,(5),P39-41

[8] 蔡凡弟.虽有家财万贯 不如金卡傍身－非接触读卡设备的核心模块[J]. 电子世界，2002,(11),P54-55

[9] 王宜怀. 王林. MC68HC908GP32 MCU 的 flash 存储器在线编程技术[J]. 微电子学与计算机，2002,(07),P15-19

[10] 潘立阳. 朱钧. flash存储器技术与发展[J]. 微电子学，2002,32(1),P1-6

[11] 解析无线射频识别技术[Z]. http://www.wx800.com/.2004-12

[12] 李锦涛，郭俊波等. 射频识别技术及其应用[Z]. 信息技术快报（中科院计算技术研究所内部刊物）,2004,(11)

[13] 林树功，蔡竟业. 射频识别技术 原理分析[Z].http://www.eetchina.com/，2004-10

[14] [德]Wolfgang Rankl,Wolfgang Effing.智能卡大全－智能卡德结构·功能·应用[M].电子工业出版社，2002

[15] [英]Mike Hendry 著.智能卡安全与应用[M].人民邮电出版社，2002

[16] International Standard ISO/IEC， FDIS 14443 . Identification Cards－Contactless Integrated Circuit(s) Cards-Proximity Cards[Z]，2000
－Part 1:Physical Characteristics
－Part 2: Radio frequency power and signal interface
－Part 3: Initialization and anticollision
－Part 4: Transmission protocol

[17] TYPE A与TYPE B两类非接触式IC卡比较. http://www.hengbo.com.2004

[18] 袁茂峰.全方位了解非接触式智能卡技术. 金卡工程，2003，8），P15-20

[19] Philips Semiconductors. Data Sheet-Mifare Standard Card IC MF1 IC S50 Functional Specification[Z]. http://www.philips.com/semiconductors，2001

[20] MC68HC908GP32 technical data[Z]. http://www.freescale.com/，1999

[21] 中国非接触IC卡行业的现状及发展趋势. http://www.smartcard.org.cn,2004

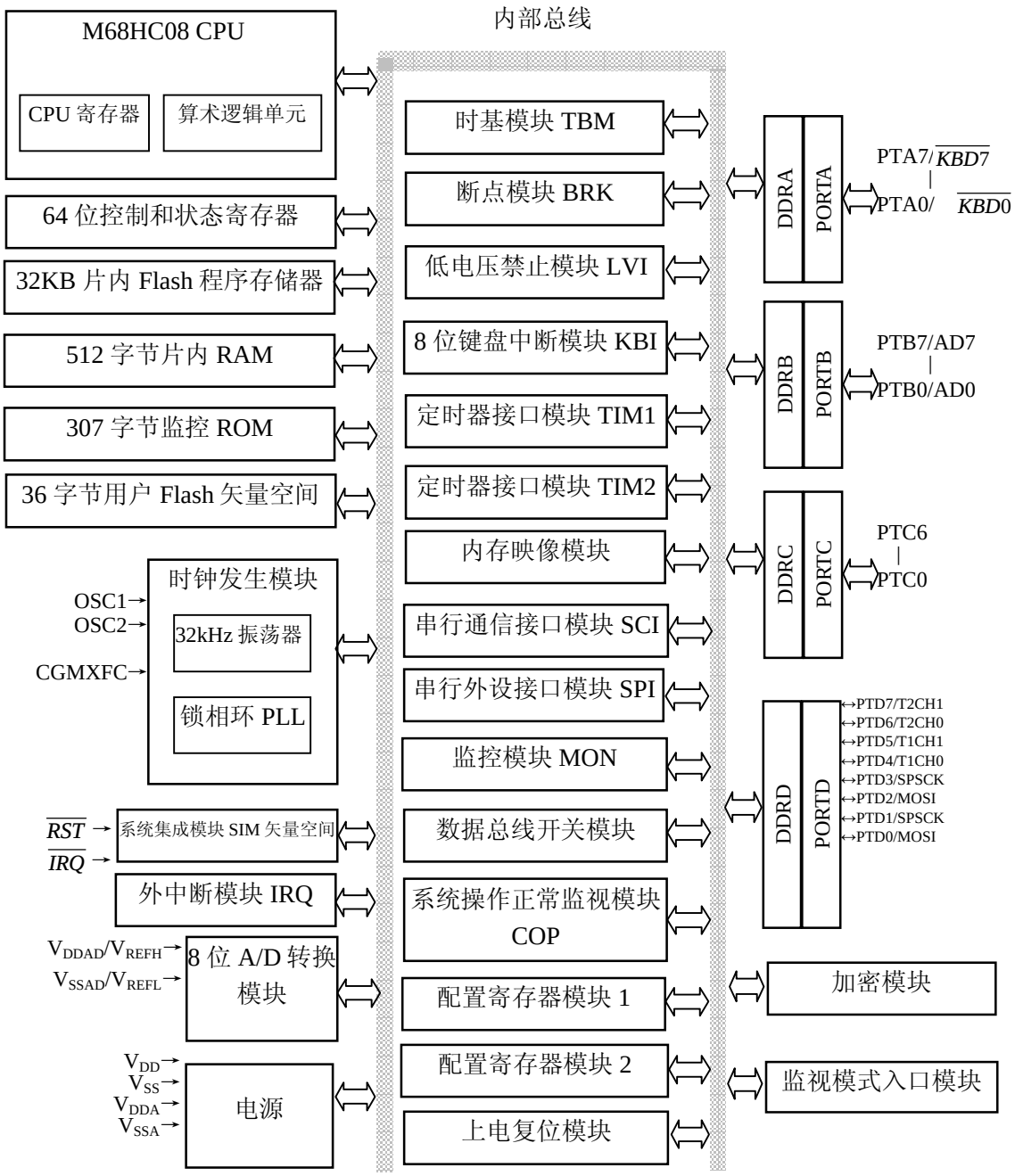
[22] Philips Semiconductors. Data Sheet-Mifare MF RC500 Highly Integrated ISO 14443A Reader IC[Z]，http://www.philips.com/semiconductors,2003

[23] Philips Semiconductors. Application Note- Mifare MF RC500 Active Antenna Concept[Z]，http://www.philips.com/semiconductors,2003

[24] Philips Semiconductors. Technical Documents- MF(14443A) 13.56 MHz RFID

Proximity Antennas, http://www.philips.com/semiconductors,2003
[25] 王宜怀著.单片机原理及其嵌入式应用教程[M].北京希望电子出版社，2002
[26] [美]Arnold Berger 著. 嵌入式系统设计[M]. 电子工业出版社,2002
[27] 刘银慧,程建平等著. Morotola微控制器 MC68HC08 原理及嵌入式应用[M]. 清华大学出版社,2001
[28] 葛永明 ,林继宝 . 嵌入式系统以太网 接口的设计 [J]. 电子技术应用,2002,28(3),P25-27
[29] 刘晓升著. 基于 8 位 MCU 的嵌入式 Internet 设计与实现[D].苏州大学，2004
[30] 张毅,赵国锋. 嵌入式 Internet 的几种接入方式比较[J]. 重庆邮电学院学报,2002,14(4),P83-86
[31] MC9S08GB60 technical data[Z] . http://www.freescale.com/
[32] RTL8019AS Data Sheet[Z]. REALTEK SEMI-CONDUCTOR CO.LTD. 2002.
[33] 史治国,王勇,王涛. 嵌入式 Internet 中 TCP 协议的实现[J]. 计算机工程与应用，2003,39(6),P148-150

附录 1 MC68HC908GP32 结构框图



附录 2 MF RC500 的寄存器

页	地址（16 进制）	寄存器名	功能
第 0 页 命令和状态	0	页寄存器（Page）	选择寄存器页
	1	命令寄存器（Command）	开始（停止）命令的执行
	2 FIFO	数据寄存器（FIFOData）	64 字节 FIFO 的输入输出
	3	主状态寄存器（PrimaryStatus）	接收器/传送器/FIFO 的状态标志
	4 FIFO	长度寄存器（FIFOLength）	FIFO 中存储数据的字节数
	5	第二状态寄存器（SecondaryStarus）	不同的状态标志
	6	中断允许寄存器（InterruptEn）	使能请求中断传送的控制位
	7	中断请求寄存器（InterruptRq）	中断请求标志
第 1 页 控制和状态	8	页寄存器（Page）	选择寄存器页
	9	控制寄存器（Control）	不同的控制标志，例如：定时、功耗等
	A	错误标志寄存器（ErroFlag）	显示最后一次执行的命令的错误状态的标志
	B	冲突位置寄存器（CollPos）	在 RF 接口检测到的第一个冲突位的位置
	C	定时器值寄存器（TimerValue）	定时器的实际值
	D CRC	低字节寄存器（CRCResultLSB） CRC	协处理器寄存器的最低有效字节
	E CRC	高字节寄存器（CRCResultMSB） CRC	协处理器寄存器的最高有效字节
	F	位封装寄存器 BitFramingI	调整位定向帧
第 2 页 传送器和编码控制	10	页寄存器（Page）	选择寄存器页
	11	传送器控制寄存器 TxControl	控制天线驱动引脚 Tx1、Tx2 的逻辑行为
	12	电导寄存器（CWConductance）	选择天线驱动引脚 Tx1、Tx2 的电导
	13	预置寄存器（PreSet13）	这些值不可以改变
	14	预置寄存器（PreSet14）	这些值不可以改变
	15	脉冲宽度寄存器（ModWidth）	选择调制脉冲的宽度
	16	预置寄存器（PreSet16）	这些值不可以改变
	17	预置寄存器（PreSet17）	这些值不可以改变
第 3 页 接收器及解码控制	18	页寄存器（Page）	选择寄存器页
	19	接收器控制寄存器（RxControl1）	控制接收器行为
	1A	解码器控制寄存器（DecoderControl）	控制解码器行为
	1B	位相寄存器（BitPhase）	选择接收器和传送器时钟间的位相
	1C	接收器阈值寄存器（RxThreshold）	选择位解码器的阈值
	1D	预置寄存器（PreSet1D）	这些值不可以改变
	1E	接收器控制寄存器 2（RxControl2）	控制解码器行为并定义接收器的输入源
	1F	时钟控制寄存器（ClockQControl）	控制时钟产生
第 4 页	20	页寄存器（Page）	选择寄存器页
	21	接收器等待寄存器（RxWait）	选择在传送之后，接收器工作之前的时间间隔

射频时间和通道冗余	22	通道冗余寄存器（ChannelRedundancy）	选择验证 RF 通道数据完整性得类型和模式
	23 CRC	预置低字节寄存器（CRCPresetLSB）	CRC 寄存器预置值的最低有效字节
	24 CRC	预置高字节寄存器（CRCPresetMSB）	CRC 寄存器预置值的最高有效字节
	25	预置寄存器（PreSet25）	这些值不可以改变
	26 MFOUT	选择寄存器（MFOUTSelect）	选择应用到 MFOUT 引脚的内部信号
	27	预置寄存器（PreSet27）	这些值不可以改变
第 5 页 FIFO、定时器及中断引脚配置	28	页寄存器（Page）	选择寄存器页
	29 FIFO	大小寄存器（FIFOLevel）	定义 FIFO 的大小，是
	2A	定时器时钟寄存器（TimerClock）	选择时钟的分频
	2B	定时器控制寄存器（TimerControl）	选择定时器的开始和结束条件
	2C	定时器重载寄存器（TimerReload）	定义定时器的预置值
	2D	中断引脚控制寄存器（IRQPinConfig）	配置引脚 IRQ 的输出状态
	2E	预置寄存器（PreSet2E）	这些值不可以改变
	2F	预置寄存器（PreSet2F）	这些值不可以改变
第 6 页 预留	30	页寄存器（Page）	选择寄存器页
	31	预留寄存器 RFU	预留未来使用
	32	预留寄存器 RFU	预留未来使用
	33	预留寄存器 RFU	预留未来使用
	34	预留寄存器 RFU	预留未来使用
	35	预留寄存器 RFU	预留未来使用
	36	预留寄存器 RFU	预留未来使用
	37	预留寄存器 RFU	预留未来使用
第 0 页 测试控制	38	页寄存器（Page）	选择寄存器页
	39	预留寄存器 RFU	预留未来使用
	3A	选择模拟测试寄存器（TestAnaSelect）	选择模拟测试模式
	3B	预留寄存器 RFU	预留未来使用
	3C	预留寄存器 RFU	预留未来使用
	3D	选择数字测试寄存器（TestDigiSelect）	选择数字测试模式
	3E	预留寄存器 RFU	预留未来使用
	3F	预留寄存器 RFU	预留未来使用

附录 3 MF RC500 的命令集

名称	代码	动作	通过 FIFO 传递的参数和数据	接收到的数据
开始（StartUp）	3F	执行复位及初始化操作。该命令不能通过软件运行，上电复位或硬件复位会引起该命令的执行。	无	无
空闲（Idle）	00	不做任何动作：取消当前执行的命令。	无	无
传送（Transmit）	1A	将 FIFO 中的数据传送给卡片。	数据流	无
接收（Receive）	16	启动接收器工作。该命令仅能由 MCU 使用，与传送命令没有时间关系。	无	数据流
传收（Transceive）	1E	将 FIFO 中的数据传送给卡片，完成后自动启动接收器工作。这条命令其实是传送命令与接收命令的综合。	数据流	数据流
写 E^2PROM（WriteE2）	01	从 FIFO 中读取数据到内部 E2PROM。	低字节起始地址 高字节起始地址 数据流	无
读 E^2PROM（WriteE2）	03	从内部 E2PROM 中读数据然后放入 FIFO。	低字节起始地址 高字节起始地址 数据字节个数	数据字节
取 E^2PROM 中的密码（LoadKeyE2）	0B	复制 E^2PROM 中的密码到密码缓冲区。	低字节起始地址 高字节起始地址	无
取密码（LoadKey）	19	从 FIFO 中取密码到密码缓冲区。	字节 0（最低字节） 字节 1…… 字节 11（最高字节）	无
认证 1（Authent1）	0C	使用 Crypto1 算法执行卡片认证的第一部分。	卡片的认证代码 卡片的块地址 4 字节卡片序列号	无
认证 2（Authent2）	14	使用 Crypto1 算法执行卡片认证的第二部分。	无	无
载入配置（LoadConfig）	07	从 E^2PROM 中读取数据初始化 MF RC500 的寄存器。	低字节起始地址 高字节起始地址	无
计算 CRC（CalcCRC）	12	启动 CRC 协处理器。CRC 计算的结果可从寄存器 CRCResultLSB 和寄存器 CRCResultMSB 中获得。	数据字节流	无

附录 4 读写模块函数说明

A 底层通信函数

（1）询卡函数

函数定义：　int mf_request(unsigned char req_mode, unsigned int *Tagtype);

功能：　　该函数发送询卡命令，若接收到返回的 TagType，说明工作范围有卡片。

参数：　req_mode　：入口参数，询卡模式。req_mode ＝0，使用 IDLE 模式询卡，只有处在 IDLE 状态的卡片才会响应询卡命令；req_mode ＝1，使用 ALL 模式询卡，IDLE 状态和 HALT状态的卡片都会响应询卡命令。

Tagtype：出口参数，返回卡片类型。MF1 卡的卡片类型是 Ox0004。

返回：　＝0：　成功；　≠0：　失败

（2）防冲突函数

函数定义：　int mf_anticoll(unsigned char bcnt,unsigned char *snr);

功能：　　实现防冲突功能：如果有几张 MF1 卡片在读写设备工作范围内，将会选择一张卡片，并返回该卡的序列号供将来调用 mf_select 函数时使用。

参数：　bcnt　：入口参数，防冲突循环过程中已接收到的卡片序列号的数据位数,调用时初始值为 0.

snr：出口参数，返回防冲突得到的卡片序列号。

返回：　＝0：　成功；　≠0：　失败

（3）选中卡片函数

函数定义：　int mf_select(unsigned char *snr,unsigned char *sak);

功能：　　用指定的序列号选择卡片，返回卡片的应答。

参数：　snr　：入口参数，卡片序列号，为执行防冲突命令得到的卡片序列号。

sak：出口参数，返回卡片确认被选中的应答标志。

返回：　＝0：　成功；　≠0：　失败

（4）密码验证函数

函数定义：　int **mf_authen**(unsigned char key_type, unsigned char *key, unsigned char block);

功能：　　验证读写器中的密码与需要访问的卡片某扇区的密码是否一致。若密码匹配，验证通过，接下来传输的数据将用加密算法加密。

参数：　key_type　：入口参数，密码模式。key_type ＝Ox60，验证 A 密码；key_type ＝Ox61，验证 B 密码；

key：入口参数，读写设备中的密码。

block：入口参数，要访问的数据块号（0≤block≤**256**）。

返回：　＝0：　成功；　≠0：　失败

（5）读数据块函数

函数定义：　int **mf_read**(unsigned char block,unsigned char *blockdata);

功能：　　从一张选定并通过密码验证的卡片中读取一个数据块的内容（16 个字节）。

参数：　block　：入口参数，块号。

blockdata：出口参数，读到的数据内容。

返回：　＝0：　成功；　≠0：　失败

（6）写数据块函数

函数定义：　int **mf_read**(unsigned char block,unsigned char *blockdata);

功能：　　从一张选定并通过密码验证的卡片中读取一个数据块的内容（16 个字节）。

参数：　block　：入口参数，块号。

blockdata：入口参数，要写的数据内容。

返回： ＝0： 成功； ≠0： 失败

（7）增值函数

函数定义： int **mf_increment** (unsigned char block,unsigned char *value);

功能： 对数值块的内容进行增值操作。

参数： block ：入口参数，块号。

value：入口参数，要增加的数值。

返回： ＝0： 成功； ≠0： 失败

（8）减值函数

函数定义： int **mf_decrement** (unsigned char block,unsigned char *value);

功能： 对数值块的内容进行减操作。

参数： block ：入口参数，块号。

value：入口参数，要减少的数值。

返回： ＝0： 成功； ≠0： 失败

（9）卡片挂起函数

函数定义： int **mf_halt** (void);

功能： 将卡片设为“Halt”状态，只有用 ALL 模式调用 request 函数或当该卡再次复位（即重新进入读写设备工作区域）时，读写设备才能够再次激活它。

参数： 无

返回： ＝0： 成功； ≠0： 失败

B 高级调用函数

（1）得卡的序列号函数

函数定义： int CardGetSn(unsigned char Mode,unsigned Char *_Snr);

功能： 寻卡并返回卡片的系列号；

参数： Mode ：入口参数，寻卡模式，＝0 为 IDLE 模式，一次只操作一张卡；＝1 为 ALL 模式，一次可操作多张卡；＝2 为选择模式，只操作选中的卡片。

_Snr：出口参数，返回卡片的系列号；

返回： ＝0： 成功； ≠0： 失败；

内部调用低级函数流程:：mf_request, rf_anticoll 和 rf_select 。

（2）读卡片数据块函数

函数 定 义 ： int ReadCard(unsigned char _Mode,unsigned char _Adr,unsigned Char *_Snr,unsigned char *_Data,unsigned Char *_NSnr);

功能： 高级读函数，从选定并通过验证的卡片某个扇区读出 1 块 16 个字节数据。

参数： _Mode: 入口参数，寻卡模式；_Adr: 入口参数，块地址；

_Snr: 入口参数，卡片序列号（仅用于模式 2）；

_Data: 从卡片中读出的数据 (长度为 16 字节)；

_NSnr: 返回卡片序列号；

返回： ＝0： 成功； ≠0： 失败；

内部调用低级函数流程: mf_request、mf_anticoll、mf_select、**mf_authen**、**mf_read**、halt。

（3）写卡片数据块函数

函数定义： int WriteCard(unsigned char _Mode,unsigned char _Adr,unsigned Char *_Snr,unsigned char *_Data);

功能： 高级写函数，向选定的并通过密码验证的卡片写入 1 块 16 个字节数据。

参数：　_Mode:　寻卡模式；
　　_Adr:　块地址；
　　_Snr:　卡片序列号（仅用于 ALL 模式）；
　　_Data:　从卡片中读出的数据 (长度为 16 字节)；
返回：　=0：　成功；　≠0：　失败；
内部调用低级函数流程: mf_request、mf_anticoll、mf_select、**mf_authen**、**mf_**write、halt。
（4）电子钱包初始化函数
函数定义：　int FormatPurse (unsigned char _Mode,unsigned char _SecNr,unsigned Char *_Value,unsigned Char *_Snr);
功能：　电子钱包初始化函数，将卡片某个扇区初始化为电子钱包数值块格式

参数：　_Mode:　寻卡模式；
　　_SecNr:　扇区号（0～15）
　　_Value:　初始化的值
　　_Snr:　卡片序列号（只在模式 2，选择模式中使用）
返回：　=0：　成功；　≠0：　失败；
内部调用低级函数流程: mf_request、mf_anticoll、mf_select、**mf_authen**、**mf_**write、**mf_read** 、compare、halt。
（5）电子钱包增值函数
函数定义：　int　**Increase**(unsigned char _Mode,unsigned char _SecNr,unsigned Char *_Value,unsigned Char *_Snr,unsigned Char *_NValue,unsigned Char *_NSnr);
功能：　电子钱包初始化函数，将卡片某个扇区初始化为电子钱包数值块格式

参数：　_Mode:　寻卡模式；
　　_SecNr:　扇区号（0～15）
　　_Value:　初始化的值
　　_Snr:　卡片序列号（只在模式 2，选择模式中使用）
　　_Nvalue:　将要增加的数值；
　　_NSnr:　返回卡片序列号；
返回：　=0：　成功；　≠0：　失败；
内部调用低级函数流程: mf_request、mf_anticoll、mf_select、**mf_authen**、**mf_increment** 、halt。
（6）电子钱包减值函数
函数定义：　int　**Decrease**(unsigned char _Mode,unsigned char _SecNr,unsigned Char *_Value,unsigned Char *_Snr,unsigned Char *_NValue,unsigned Char *_NSnr);
功能：　电子钱包初始化函数，将卡片某个扇区初始化为电子钱包数值块格式

参数：　_Mode:　寻卡模式；
　　_SecNr:　扇区号（0～15）；
　　_Value:　初始化的值；
　　_Snr:　卡片序列号（只在模式 2，选择模式中使用）；
　　_Nvalue:　将要减去的数值；
　　_NSnr:　返回卡片序列号；
返回：　=0：　成功；　≠0：　失败；
内部调用低级函数流程: mf_request、mf_anticoll、mf_select、**mf_authen**、**mf_decrement** 、halt。

攻读学位期间公开发表的论文及参与的鉴定项目

[1] 徐丽华、王宜怀， MC68HC908JL8 MCU中虚拟 EEPROM 特性的应用及剖析.计算机工程与应用， 2004，40(28)，P106-108

[2] 参与《RFID 卡系列读写器的研制》项目，该项已于 2004 年 12 月 17 日通过江苏省科学技术厅鉴定

[3] 参与王宜怀、刘晓升编著的《嵌入式应用基础技术教程》中第 18 章“智能卡”的撰写，该书将于 2005 年 7 月在清华大学出版社出版