嵌入式RFID中间件的设计与实现

邓毅华， 谢胜利
(华南理工大学 电子与信息学院，广东 广州 510640)

摘 要：对基于嵌入式系统的RFID中间件进行了研究。该中间件架构在多个针对嵌入式系统的开源软件系统平台之上，系统中各组件通过环形缓存和守护进程之间的配合构成松散耦合关系，由ALE_Reporter模块完成EPCglobal的ALE标准规划的数据处理逻辑。这种软件架构方法可以缩短开发周期，使各模块基本保持原有面貌，利于将来对软件系统进行维护和扩展。

关键词：中间件；ALE标准；嵌入式系统；软件架构；射频识别
中图法分类号：TP311 **文献标识码**：A **文章编号**：1000-7024 (2008) 07-1716-03

Design and implementation of embedded RFID middleware

DENG Yi-hua, XIE Sheng-li
(School of Electronic and Information Engineering, South China University of Technology, Guangzhou 510640, China)

Abstract: A design of RFID middleware based on embedded system is presented. Several software components is relied on, which are specially designed for embedded system and their source codes are open and free. By the circle caches, these software components work together smoothly, they help the ALE_Reporter achieve the ALE procession. The relations of the components in the architecture is loose coupling which make the software components keep their own characters and make the works of software maintenance and upgrades much easier and sooner.

Key words: middleware; ALE protocol; embedded system; software architecture; RFID

0 引 言

RFID中间件是在物理RFID阅读器和上层应用软件之间的一层软件，按照EPCglobal的结构框架(EPCglobal architecture framework)的定义，它处在EPCIS(EPCglobal information service)系统与RFID阅读器之间，被称为应用层事件(application level events，ALE)标准。对于上层应用程序，RFID中间件屏蔽了各种不同的阅读器和空间传输标准，使上层应用软件得到统一、不变的数据和控制接口；另外，它还能对原始数据进行整合和过滤，产生用户定制的数据报表。在大规模应用RFID技术时，将产生海量数据，应用RFID中间件将使数据处理和传输变得相对简单。中间件的承上启下作用使RFID系统设计更为灵活，维护更为简单，不论是后端应用程序的变化，还是前端阅读器变化，都不会影响另外一端，省去维护多对多连接的复杂性问题。

RFID中间件是RFID技术应用领域和应用规模发展的产物，当今市场上已有一些产品，如：SUN、Microsoft和BEA的RFID中间件产品，EPCglobal组织为此制定了相关参考标准，即上述的应用层事件ALE标准。目前，大多数公司都是参照该标准规范和设计自己的产品。然而，这些产品大都是运行在PC机或服务器端的软件，这使RFID系统在布置上不灵活，而且使系统成本上升。若可以将RFID中间件集成在具有一定"智能"的阅读器中，将可以在一定程度上解决上述问题。本文将探索一种可集成在阅读器上的RFID中间件实现方案。

1 嵌入式RFID中间件的组成

带有中间件的"智能"阅读器可以管理一定数量的"哑"阅读器，如在实施时，可以用一台"智能"阅读器配多台"哑"阅读器，形成"智能"阅读器网络，既节约了成本，也方便了系统的布置、安装和维护，大大提高了整体RFID系统的性能。这种集成在阅读器中的RFID中间件利用了阅读器本身的嵌入式系统的资源，本文所述系统硬件平台：ARM9内核、32 Mbyte内存、16Mbyte Flash和10M/100M网口，软件系统平台：嵌入式LINUX操作系统平台。由此可见，系统资源要求不算高，对于目前和未来设计的阅读器而言，这样的要求是可以接受的。

嵌入式RFID中间件系统组成如图1所示，由6部分组成：①DSP接口部分，阅读器本身可以读写RFID标签，阅读器上DSP模块完成标签信号的基带处理，利用中间件系统的DSP接口，可实现本阅读器标签数据与RFID中间件系统之间的交互；②外部阅读器接口，该接口是中间件与其它外部阅读器连

收稿日期：2007-04-28 **E-mail**：dengyih@tom.com
基金项目：广东省工业攻关课题基金项目(2005B10101005)。
作者简介：邓毅华(1973—)，男，广东佛山人，博士研究生，讲师，研究方向为嵌入式系统设计、RFID系统设计；谢胜利(1958—)，男，博士，教授，博士生导师，研究方向为盲信号分离。

接的通道，通过该接口，中间件可以收集和处理来自不同阅读器的数据，并向这些外部阅读器发出控制命令；③SQLite模块，SQLite是一个源码开放的小型嵌入式数据库，它提供大量SQL语言操作，工作速度快，能基本满足本系统的数据处理实时性要求，本系统通过它完成标签数据的处理、过滤和查询等功能；④GoAhead模块，GoAhead是嵌入式Web服务器，它不仅小巧，而且能提供丰富的功能，包括ASP、JavaScript、CGI等，在本系统中利用它完成与上层应用(如：客户端应用(Client)，远端大型ALE服务器(ALE Server)等)的通信，并为客户提供基于Web的对阅读器管理的平台；⑤ALE_Reporter模块，该模块架构在SQLite和GoAhead模块之上，完成ALE数据处理功能，是中间件的核心功能模块；⑥Timer模块，它是系统中的定时器，ALE在处理ECSpec[1]时使用状态机机制，而且在系统中还有许多应用需要时间作为标准，为此专门设计一个Timer模块。

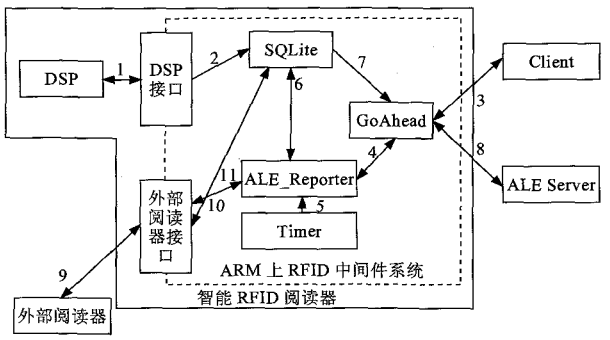

图1 嵌入式RFID系统组成

该系统的工作原理如下：①DSP接口接收来自DSP读取的RFID标签数据；②DSP接口将RFID标签数据存放在SQLite中；③外部用户(Client或ALE Server)定义(define)ECSpec，并申请(Subscribe)ECSpec；④GoAhead将用户定义和申请等请求(request)发往ALE_Reporter处理；⑤Timer时间条件触发ALE_Reporter进行相关操作；⑥ALE_Reporter向SQLite发送产生报表命令；⑦SQLite报表发往GoAhead；⑧GoAhead将报表发送到目标用户；⑨外部阅读器上传数据和接收命令；⑩外部阅读器接口将上传数据发往SQLite，进行数据存储；⑪外部阅读器接口将外部阅读器属性发往ALE_Reporter，并接收ALE_Reporter发往外部阅读器的命令。

ALE_Reporter是整个RFID中间件的核心，它负责处理ECSpec状态机，负责产生各种报表的数据库查询请求，负责维护整个数据库中的数据，其余模块都是辅助它完成ALE处理逻辑的部分，它是本系统开发的核心。

2 ALE_Reporter的架构

目前世界上有两大RFID标准体系，一是国际标准化组织的ISO18000系列，另一个是EPCglobal组织的EPC标准。前者主要定义了空中接口，涵盖了从低频频段一直到微波频段的RFID空中接口；后者不仅为RFID的空中接口定义了标准，而且为整个RFID应用体系的各个层面制定了标准。本嵌入式RFID中间件如果放在整个EPCglobal的框架体系[3]中，是处在EPCIS Capture Application[3]层和RFID Reader[4]层之间的部分，功能涵盖了Filtering & Collection (ALE) Interface、Filtering & Collection("RFID Middleware")和Reader Protocol等3部分。在这3部分中，ALE是本中间件面向上层的接口；Reader Protocol是本中间件面向下层的接口；ALE_Reporter完成Filtering & Collection这一层功能，其构架如图2所示。

本设计遵从ALE v.1.0标准，其工作原理如下：

(1)远端用户程序按ALE V1.0标准，通过http-XML[1]方式发define请求，该define请求以XML方式描述了ECSpec的结构。

(2)GoAhead接收http请求，将http请求中的XML数据包提取出来，根据XML数据包中ECSpec数据生成GECSpec数据结构的实例，将该实例插入请求队列(rqs_pool)中。

(3)守护线程ws_rqs监视rqs_pool，一旦发现有被占用的

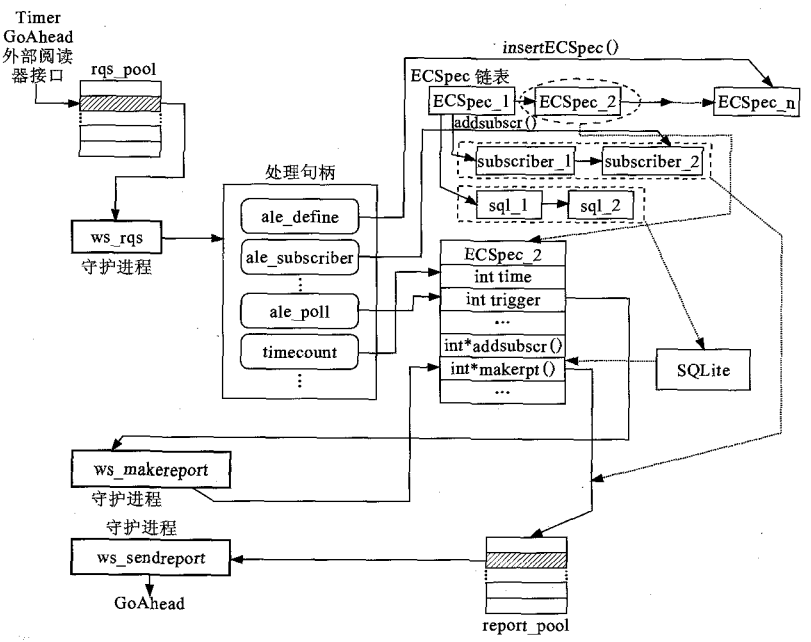

图2 ALE_Reporter架构

槽位，便取出槽中数据，并将它分发到相应的处理句柄，同时清空槽位。

(4)ale_define是处理define请求的处理句柄，它将判断GECSpec实例的有效性，无效的丢弃，并将错误信息写入报告队列(report_pool)中；有效的创建一个新的ECSpec_n，并调用insertECSpec()将ECSpec_n插入到ECSpec链表中。在GECSpec数据类型中包含了ECBoundarySpec和ECReportSpecs，其中涉及到标签数据的过滤和聚合，这些条件在ale_define中被翻译成多步数据库操作的动作，每一个动作为一个sql_i，这些sql_i构成一个链表，如图2的sql_1、sql_2。

(5)define之后，用户会继续发来订阅(subscribe)的请求。GoAhead接收到subscribe之后，将订阅用户信息插入rqs_pool中，由ws_rqs发现该请求，将该请求分发到ale_subscriber处理句柄，该句柄生成subscriber实例，并调用addscriber()，将新实例插入到相应的ECSpec对应的subscriber链表中，如图2中subscriber_1和subscriber_2。

(6)Timer在每隔1秒时产生一个时间计数，它将计时请求(timecount)插入rqs_pool中。守护线程ws_rqs发现该计时请求，将计时请求分发至timecount()处理句柄进行处理。timecount()将轮询ECSpec链表，对其中需要计时的ECSpec_i进行操作，若该ECSpec_i达到了触发要求，标志其触发状态为TRUE。

(7)守护线程ws_makereport轮询ECSpec链表，当发现有ECSpec的触发状态为TRUE时，调用makerpt()完成报表的生成。

(8)makerpt()通过对应ECSpec的数据库操作动作链表，如图中ECSpec_1的sql_1和sql_2，对SQLite进行操作，另外makerpt()通过对应ECSpec获得报表订阅用户信息，最后将数据库查询结果和用户信息写入report_pool缓存中，形成报表。

(9)守护线程ws_sendreport监视report_pool，当发现其中有需要发送的报表时，取出报表，并向相应用户发送报表。

ALE_Reporter模块采用了多线程处理结构，每个线程担负着不同任务，通过公共缓冲区(rqs_pool、report_pool)完成数据交接，构成松散耦合的结构，为今后系统的扩展提供了灵活和简单的方法。

3 缓冲区 (rqs_pool和report_pool) 的设计

在ALE_Reporter中，入口和出口均使用了缓冲区(rqs_pool和report_pool)，这两个缓存区采用环形缓冲器结构，是通过共享内存和信号灯的配合来实现。本设计参考了文献[2]中第13章“多个客户通过共享内存向一个服务器发送消息”的设计方法，另外，由于本设计采用System V的进程间通信机制和其它一些程序设计要求，在实际设计上做了一些改动。这两个缓存的结构大体相同，此处仅分析rqs_pool。rqs_pool的数据结构如下：

```
struct  rqs_pool{
    int    pool_mutex; /*rqs_pool 中使用的信号灯集*/
    int    nput; /*插入新槽位时的位置*/
    int    nget; /*下一次取出槽位时的位置*/
    long   rqs_off[NMESG]; /*rqs 的偏移地址*/
    request rqsdata[NMESG*sizeof(request)];/*实际的 rqs*/
};
```

pool_mutex信号灯集包含3个信号灯：sem[0]、sem[1]和sem[2]。sem[0]用作二值信号灯，用于维护rqs_pool各槽位的插入和移除，初始化为1。sem[1]用作计数信号灯，信号灯值反映空槽位数量，初始化为NMESG(槽位的总数)，当移除旧槽位时，该信号灯值加1，插入新槽位时，该信号灯减1。sem[2]用作计数信号灯，信号灯值反映当前已占用槽位，初始化为0，当移除旧槽位时，该信号灯值减1，插入新槽位时，该信号灯加1。通过这些信号灯协同操作，实现各线程的同步，这个方法在文献[2]中有详细论述。

在rqs_pool中，request是每个槽位对应的数据结构，它包含请求的类型，以及简单的请求数据。如果是复杂的请求，如上述来自GoAhead的define请求，它包含大数据块(GECSpec)，在这种情况下，大数据块被写入一个内存文件中，request中只保留对该内存文件的索引，ws_rqs通过该索引可以打开内存文件，并获取GECSpec数据。对于rqs_pool，复杂的request是很少有的，只在系统初始化，或添加新阅读器和新用户时才出现，但对于report_pool，大数据块(report)是经常出现的，将数据分别对待成为了必要：简单小数据块写入规则的数据槽，将复杂大数据块作为文件，由数据槽对文件进行索引。这样的设计可以兼顾大块或小块的数据，可以更有效地使用内存。

4 结束语

RFID阅读器的一个发展趋势是阅读器集成嵌入式系统，使原来只能完成单一功能的“哑”阅读器变为有强大处理能力的“智能”阅读器。本文所述的RFID中间件利用“智能”阅读器上本身的嵌入式系统资源，架构基于多个开源软件平台上的嵌入式系统，既可以减小系统的开发周期，又大大节约开发成本。通过试验，利用该方法实现的RFID中间件可以很好地收集、过滤和上报来自20~25台RFID阅读器的数据，各接口和功能基本达到了ALE v1.0规范提出的要求。

参考文献：

[1] AUTO-ID Center. The application level events (ALE) specification version 1.0 [S]. www.epcglobalinc.org/ standards_technology/EPCglobal_Application_Level_Events (ALE) _Specification_v1.pdf.

[2] Richard Stevens W. UNIX网络编程第2卷:进程间通信[M]. 北京:清华大学出版社, 2000.

[3] AUTO_ID Center. The EPCglobal architecture framework [S]. www.epcglobalinc.org/standards_technology/Final-epcglobal-arch-20050701.pdf.

[4] AUTO-ID Center. Auto-ID reader protocol 1.0 [S]. www.epc.org.mx/contenido/WD-reader-protocol- 200309051.pdf.

[5] 赵毅强,曾隽芳.Web Services在RFID系统中的应用综述[J]. 计算机应用研究,2006,23(12):1-3.

[6] 雷小俊,李伟. SQLite在嵌入式Web服务器中的应用[J].信息技术,2006(6):127-130.

[7] 许炜,程文青,刘威.一种灵活的RFID事件规则管理框架[J].计算机应用研究,2007,24(1):86-89.

[8] 贺平,蒋亚军.EPC系统的Savant中间件技术及其设计实现[J]. 计算机工程与应用,2006,42(9):221-227.