

一种面向嵌入式开发的软件体系结构^{*})

许晓伟 王知行 曹晓叶

(华南理工大学计算机科学与工程学院 广州 510641)

摘要 在基于操作系统的嵌入式应用开发中,特别是在软件架构上还存在着一些具有共性的问题,制约着整个行业的发展。针对这些问题,在嵌入式软件体系结构和开发过程两个方面进行了改进和规范:在软件体系结构上采用了层模式、微核模式、映像模式相结合的架构;在开发过程的规范和构件复用上采用了领域工程、应用工程和集成开发环境的思想。改进后的体系结构和规范在实际开发中得到了验证,值得进一步完善和推广。

关键词 嵌入式系统,层结构模式,微核模式,映像模式,复用

A Kind of Software Architecture for Embedded System Development

XU Xiao-Wei WANG Zhi-Yan CAO Xiao-Ye

(School of Computer Science and Engineering, South China Univ. of Tech., Guangzhou 510641)

Abstract There are some serious technology and especially architecture problems in the embedded application development based on embedded operation system, which restrict the development of the industry. Improvement and standardization have been made in software architecture and development process in this paper. Level pattern, microkernel pattern and reflection pattern have been combined in this software architecture. And the ideas of domain engineering, application engineering and IDE (Integration Development Environment) have been used in the standardization of the development process and the reuse of the components. These improvements have made good effect in practice, and farther works should be continued in the future.

Keywords Embedded system, Level pattern, Microkernel pattern, Reflection pattern, Reuse

1 引言

随着信息技术的飞速发展,特别是互联网的迅速普及,数字化的后 PC(Post-PC)时代已经来临,嵌入式设备成为这一时代的主流产品,嵌入式系统广泛地应用到科学研究、工程设计、军事技术各个领域以及人们的日常生活等方方面面中。嵌入式系统(Embedded Systems)是以应用为中心,以计算机技术为基础,软件硬件可剪裁的适应应用系统对功能、可靠性、成本、体积、功耗等严格综合性要求的专用计算机系统。目前,嵌入式系统的应用数量上已经远远超过通用计算机系统,但是嵌入式应用的多样性,使其很难进行通用化设计。近年来随着硬件复杂性的增加和对系统功能和性能需求的大幅度提高,使得嵌入式软件的开发工作量可占到全部工作的70%以上,成为制约嵌入式系统开发的瓶颈。

在我们的票据影像项目^[1,2]的研究成果市场化阶段,研究使用了一种无操作系统的微型嵌入式系统作为便携使用的卡片字符读入终端^[3],而在卡片贴标机项目中我们又开发了一种使用嵌入式操作系统的工业控制器作为卡片图像的获取终端。在嵌入式开发中,我们发现针对大量不同的应用,其软件开发过程基本类似:每一次应用开发都要从硬件底层开始,导致大量重复的编码工作,严重影响开发效率。即使是非常相似的应用,由于环境变化、硬件变更等种种原因也面临着代码移植问题。由于嵌入式软件开发技术不同于通用软件开发技术,它依赖于开发人员的经验,特别需要熟悉底层硬件特性,这就容易影响到系统的可靠性、稳定性、扩展性和可维护

性。在许多行业应用中,开发人员希望能够在领域应用上发挥他们的行业专长,方便的开发应用程序。他们基本掌握了基于通用计算机的开发技术,但却不太希望学习嵌入式系统的开发技术,毕竟还要考虑成本因素和对系统开发周期的影响。

基于上述原因,我们在研究过程中将各种应用需求和硬件环境抽象出来,在软件体系结构和软件开发过程的层次上设计了这些问题的统一解决方案。

2 嵌入式开发原理及存在的问题

在制卡行业中,卡片字符的实时校验是生产线上卡片喷码和贴标之间必不可少的关键工序,需要采用高速的在线字符识别系统,如图1所示:首先采集卡片图像,然后实时识别卡片上喷印的字符信息,并与数据库进行校验核对,如果发现错误,则发出控制信号利用机械装置把卡片从传送带上剔除。



图1 卡片贴标机在线字符识别系统

^{*}基金项目:科技部科技型中小企业技术创新基金无偿资助项目(立项代码,02C26214400224);广东省科技计划资助项目(项目编号:2002A1020104)。许晓伟 博士生;王知行 教授,博士生导师,IEEE会员;曹晓叶 讲师。

它是一种用于工业控制的嵌入式系统,如图 2A 所示,其原理与通用计算机系统基本一致,包含硬件和软件两部分:硬件架构上以嵌入式控制器 MCU(或嵌入式处理器 MPU)为中心,配置存储器、I/O 设备、通信模块等必要的外设。在硬件设计上,系统使用闪存(Flash Memory)作为存储介质以减小体积、提高可靠性。需要说明的是,由于受硬件资源所限,过去许多简单的应用不需要操作系统(如单片机控制);但随着技术的发展,需求的提高,复杂程度也越来越高,特别是有多控制环路、并需要在优先级可控的情况下能预测其运行状态时,嵌入式实时操作系统 RTOS(Real Time Operation System)的使用变得越来越重要和广泛。这里我们所研究的是这种复杂的具有操作系统的嵌入式系统体系结构,如图 2B 所示。

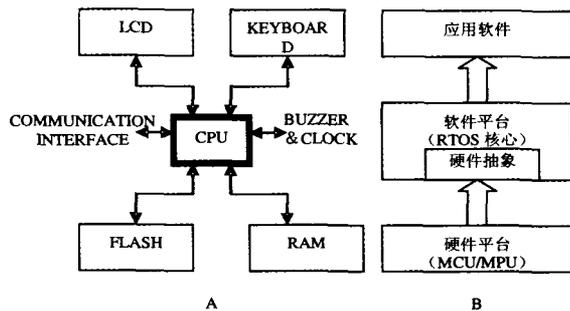


图 2 嵌入式系统组成;A 硬件组成, B 层次结构

2.1 嵌入式开发原理

目前基于这种系统结构的系统设计和软件开发过程大体一致:硬件设计定型之后,基于这个特定硬件平台选用某种嵌入式实时操作系统,它是一段在系统启动后首先执行的背景程序 RTOS,用户的应用程序是运行于 RTOS 之上的各个任务,RTOS 根据各个任务的要求,进行资源管理、消息管理、任务调度、异常处理等工作。应用软件的开发上还是基于特定硬件和操作系统的指令集(一般提供兼容标准 C/C++ 的编译器)。与通用软件开发不同,嵌入式软件开发对实时性要求高,对代码尺寸也有要求,并且在开发、调试过程中有宿主机和目标机的角色之分,如图 3 所示:宿主机是执行软件编译、链接、定址过程的计算机,目标机指运行嵌入式软件的硬件平台。首先在宿主机上把应用程序转换成可以在目标机上执行的二进制代码,然后将操作系统映像和应用程序映像下载到目标机的存储介质中。系统工作运行过程如下:通电复位启动嵌入式操作系统,然后加载存储介质中的应用程序,由该程序完成数据的采集、处理、保存、显示、通信等操作。

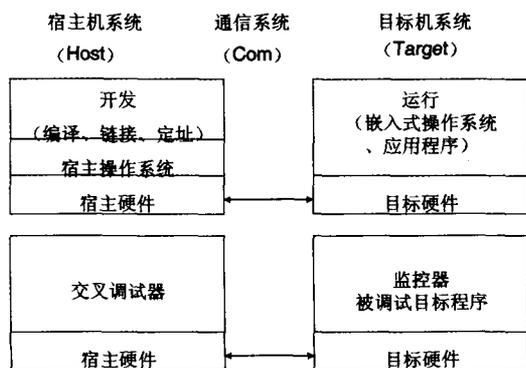


图 3 交叉开发、调试系统

2.2 嵌入式开发存在的问题

通过多个不同应用的开发,我们发现开发过程中的一些共性问题:

(1)从上节的图 2B 可以看出,应用软件需要对操作系统提供的硬件驱动接口进行操作,由于嵌入式设备的千差万别,决定了嵌入式操作系统的专业性和多样性(目前已经有了上百种嵌入式操作系统),不同系统之间兼容性差,往往需要针对不同的硬件编写实时操作系统内核移植代码。

(2)从嵌入式操作系统的发展过程来看,由于受到传统操作系统内核实现的影响,实现其内核所采用的具体结构和技术不合理^[4],影响了系统的性能和效率,造成了嵌入式操作系统的非常低的通用性和日益缩短的嵌入式产品上市时间的矛盾。

(3)在工程中,许多嵌入式软件进行升级、功能扩展,需要向嵌入式操作系统平台进行代码移植。

(4)从软件开发过程来看,开发中的编译、链接、下载,以及软件测试过程过于专业、烦琐,重复工作量较大,尤其是在调试阶段,需要不断地重复这些过程。

(5)宿主机系统和目标机系统之间的差异也产生了一些问题,如字节顺序(大端、小端)、字节对齐等数据类型不匹配问题。

上述问题造成了嵌入式软件易出错、开发效率低、开发周期长,并极大的影响了产品的可靠性、稳定性和可移植性。开发工作只能由熟悉嵌入式系统的专业开发人员完成,而在实际的行业应用中,由于领域知识及其他原因,希望能将开发工作转移到由领域开发人员完成。

经过对上述情况的总结、抽象,我们认为目前的嵌入式开发模式需要解决以下两个问题:

- 1)软件复用问题:包括代码复用、构件复用和过程复用。
- 2)特殊性屏蔽问题:这包括对目标系统硬件特殊性和对操作系统平台特殊性的屏蔽。

3 混合模式的体系结构解决方案

要解决上述问题,使开发人员只需具有通用软件开发知识就可从事跨平台的嵌入式应用开发,经过反复实验,我们将系统的层模式^[5]体系结构进行了改进,采用模块化设计,来实现跨硬件体系结构的复用和跨嵌入式软件应用部分的复用。如图 4 所示,系统按层次划分为:应用软件层→中间层→层操作系统层→功能级抽象层→寄存器级抽象层→硬件层,而开发的或定制的嵌入式操作系统则被纳入微核模式^[5]的框架,与相应的层次对应。体系结构的各层之间有不同的功能和责任。

操作系统层采用微内核结构,内核(微核)只提供基本的功能,比如:任务的调度、任务之间的通信与同步、内存管理、时钟管理等,其它的应用构件,比如网络功能、文件系统、GUI 系统等均在用户态,以系统进程或函数调用的方式工作。因此,用户可以根据自己的需要选用相应的应用构件,布置在内部服务器或外部服务器上,以实现复用。

内部服务器构件屏蔽了对硬件层的操作特殊性,它包含两层封装:1)寄存器级抽象层。直接与硬件相关,封装 CPU 和外围设备的寄存器接口及其行为,一般需要定义完整的寄存器序列名称,如 GAL。针对不同的芯片要编写合适的驱动程序,寄存器级封装反映了实现同一功能元器件的代码复用。2)功能抽象层。封装硬件的功能行为,如板级支持包 BSP

(Board Support Package),通常定义成若干可以执行命令的函数,可进行功能代码复用。

中间层作为操作系统和应用程序间的接口,通过外部服务器向应用层提供服务,展现了基于应用领域的用户个性化视图。它包含了用户开发所希望的嵌入式系统的一些特征,根据这些特征,结合下层微核和内部服务器中的相关模块,就能生成定制的系统软件平台。这些特征包括配置嵌入式系统所需的各种信息,如:硬件信息(寄存器号)、系统架构、功能模块、初始化及相关参数等。除了规范平台体系结构特征外,中间层本身又可以含有可供大多数行业应用的公共服务模块,如键盘输入、LCD输出、文件系统、图形、网络、串口/并口通信、蜂鸣器等。

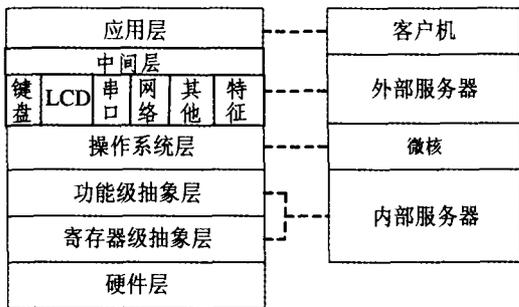


图4 改进的体系结构模式:层模式+微核模式

在这种改进后的体系结构下进行应用开发,通过复用技术,应用层只需调用中间层提供的服务,避免了对特定硬件和操作系统的直接操作,避免了重复开发工作,增强了系统的可剪裁性、可移植性和可扩展性,并且有效地减轻了微核的负担,提高了系统的性能和效率。需要指出的是,根据应用类型的不同,平台的复杂程度会有较大的不同。例如,在我们的应用中,涉及到的高速图像采集输入和控制信号输出,它们都对中间层提供的串口服务模块进行调用。

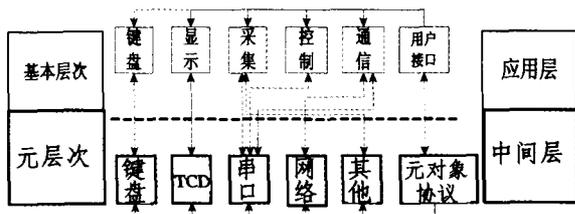


图5 改进的体系结构模式:层模式+映像模式

为了提高嵌入式系统的整体性能,没有采用微核模式中的客户机-适配器-服务器结构,用户应用层直接依赖于中间层。因此,在实际开发中,我们发现需要投入大量精力解决涉及到中间层接口的应用代码移植、数据类型转换等方面的问题。为此我们考虑使用映像模式^[5]的结构来解决这个问题,其做法是把中间层构造为元层次,将目标系统的中间层提供的服务及其有特殊要求的数据类型封装成一个个的元对象,而对应于用户应用层的基本层次只使用元层次的中间层接口和数据类型来完成应用逻辑,如图5所示。就目前的需求而言,由于宿主系统和目标系统是一一对应完全固定的,不需要动态改变系统的结构和行为,所以我们把中间层的特征记录作为元对象协议,以实现元对象的静态操作。协议格式采用元标记语言 XML(Meta-markup Language)。XML语言规范遵循数据描述和数据显示分离的原则,与操作系统、编程语

言和应用程序无关,可实现跨平台的信息表达和传递。引入映像模式的主要目的是消除宿主机和目标机两系统之间的差异,避免了应用层的直接依赖性,方便用户应用软件的代码移植。这种模式为以后的宿主系统和目标系统的多对多情况、嵌入式设备网络接入技术、嵌入式移动技术提供了可扩展性。

4 嵌入式应用软件开发过程的规范

在软件体系结构模式上经过这两方面的改进后,形成的嵌入式软件平台可以提供两个层面上的复用:一是硬件层次上操作系统的移植,使之可加载功能级抽象层和寄存器级抽象层来实现对硬件底层的支持,二是在中间层基础上的应用软件的开发和移植。

与通用软件开发工作流程类似,嵌入式应用软件开发过程包含以下几个阶段:需求分析、硬件选型和软件设计、编码、测试、维护等。如果每个应用的开发都是从头开始,就必然存在大量的重复劳动,如:用户需求获取的重复、需求分析和设计的重复、编码的重复、测试的重复和文档写作的重复等。因此,有必要对开发流程中的共性操作进行抽象和规范。我们把这些可重复利用的部分做一个分析和规范,主要包括三类成分:1)通用基本构件,如基本的数据结构、用户界面元素等;2)领域共性构件:应用系统所属领域的共性构成成分;3)应用专用构件:每个应用系统的特有构成成分。有了这些可复用的对象和构件,在嵌入式系统集成开发环境 IDE(Integration Development Environment)中,实施领域工程和应用工程^[6],如图6,就可以使用这些已有的软件设计和开发成果,半自动地生成所需系统,这一流程规范化过程我们称之为过程复用。

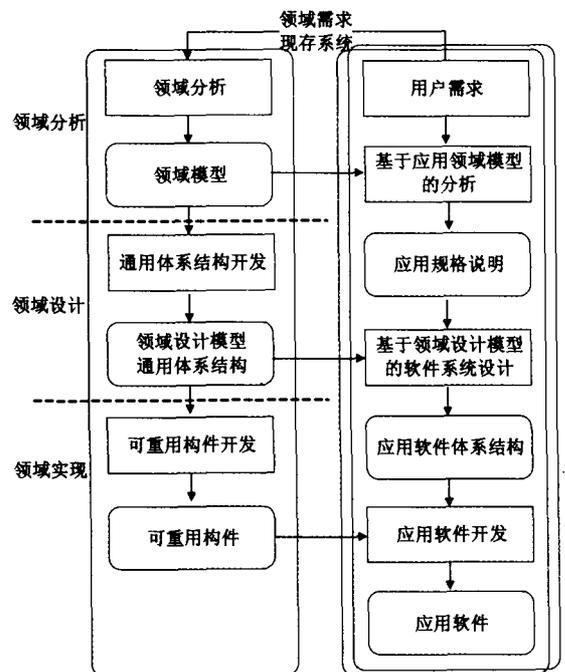


图6 领域工程和应用工程

5 应用实例

在我们的字符识别系统中,采用的集成开发环境包括 Tasking C/C++编译器、软件模拟器和 Nohau 实时在线仿真器,硬件配置为 P51XAS3,嵌入式操作系统为源代码开放

(下转第 274 页)

为复杂大系统的设计和验证提供有力保证。下一步的研究工作是进一步推广共享子网合成满足动态性质的保持性条件,并研究对其它性质(如结构性)的保持性问题。

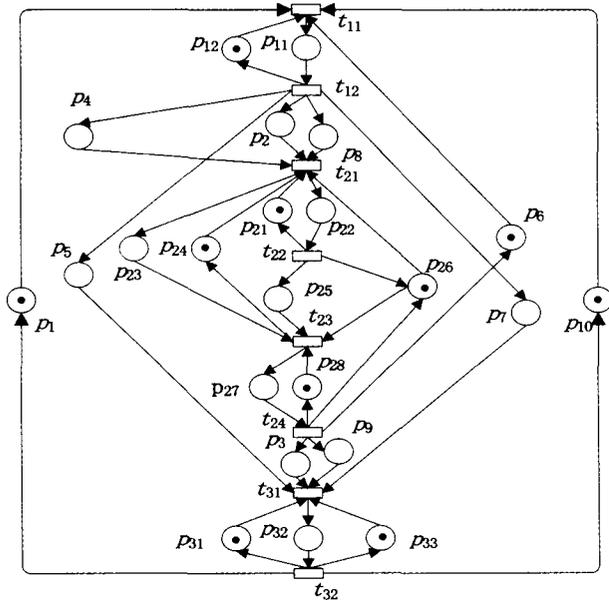


图4 柔性制造合成网系统

参考文献

- 1 Jeng M D. A petri net synthesis theory for modeling flexible manufacturing systems. IEEE Transaction on systems, Man, and Cybernetics-Part B; Cybernetics, 1997, 27(2): 169~183
- 2 Jeng M D, DiCeara F. A review of synthesis techniques for Petri

- nets with application to automated manufacturing systems. IEEE Transaction on systems, Man, and Cybernetics, 1993, 23(1): 301~312
- 3 Jeng M D, DiCeara F. Synthesis using control nets for modeling share resource systems. IEEE Transaction on Systems, Man, and Cybernetics, 1996, 11(3): 317~324
- 4 吴哲辉. 有界 Petri 网的活性和公平性的分析与实现, 1989, 12(4): 267~278
- 5 Esparza J, Sila M. On the analysis and synthesis of free choice systems. In: Rozenberg G, ed. Lecture Notes in Computer Science. New York: Springer-Verlag, Vol 483, 1990. 243~286
- 6 Esparza J, Sila M. Top-down synthesis of live and bounded free choice nets. In: Rozenberg G, ed. Lecture Notes in Computer Science. New York: Springer-Verlag, Vol 524, 1991. 118~139
- 7 Badouel E, Darondeau Ph. The synthesis of Petri nets from path-automatic specifications. Information and Computation, 2004, 193: 117~135
- 8 Franceschinis G, Gribaudo M, et al. Compositional Modeling of Complex Systems; Contact Center Scenarios in OsMoSys. In: Proc. the 25th International Conference on Application and Theory of Petri Nets. Bologna, Italy, 2004. 177~196
- 9 Juhás G, Lorenz R, et al. Synthesis of Controlled with Modules of Signal Nets. In: Proc the 25th International Conference on Application and Theory of Petri Nets. Bologna, Italy, 2004. 238~257
- 10 Yoo Dong-Hoon, Lee Dong-Ik, et al. Operation Net System: A Formal Design Representation Model for High-Level Synthesis of Asynchronous Systems Based on Transformations. In: Proc. the 25th International Conference on Application and Theory of Petri Nets. Bologna, Italy, 2004. 435~453
- 11 夏传良. 一种 Petri 网合成方法及其在系统建模中的应用. 系统仿真学报, 2006, 18(2): 271~277
- 12 夏传良, 陆维明. 论 Petri 网链路合成. 系统工程理论与实践, 2005, 25(6): 19~26
- 13 Chao D Y. Petri net synthesis and synchronization using knitting technique. Journal of Information Science and Engineering, 1999, 15: 543~568
- 14 王维平, 朱一凡, 华雪倩, 等. 离散事件系统建模与仿真. 长沙: 国防科技大学出版社, 1997. 99~107

(上接第 260 页)

的、可裁剪的 CMX。系统多任务内核为 FOS_XA_C.H、dos.h、fos.h、FOS_INL.H 四个文件,环境和硬件参数定义包含在 stdlib.h 和 hal.h 文件中,可供用户修改。

系统主程序 main.c 部分代码如下:

```
main()//主程序
{
    initiate(); //初始化系统硬件
    OSInit(); //多任务系统初始化
    OSInit_XA();
    OSTaskCreate();
    OSStart();
}

void TASK_START(void) //任务系统启动及管理
{
    InitRealTime(); //初始化数据参数
    OS_ENTER_CRITICAL(); //开中断
    settimer(); //设置定时器
    OS_EXIT_CRITICAL(); //关中断
    ... //不掉电 RAM 数据处理
    ... //初始化串口、以太网
    OSMboxCreate(); //创建界面邮箱
    OSTaskCreate(); //创建键盘处理任务
    ...
}
```

网络通讯程序 ethernet.c 部分代码如下:

```
#include "..\hal\hal.h" //硬件环境参数
#include "..\ethernet\socket.h" //中间层网络接口
#include "..\ethernet\ethernet.h" //应用层定义
```

中间层采用 XML 语言对系统规范说明示例:

```
<EmbeddedSystem>
    <RTOS> CMX </RTOS>
    <FunctionLibrary> TIMER1 </FunctionLibrary>
    <Communication> COM_MSG </Communication>
    ...
    <MainTask> main() </MainTask>
</EmbeddedSystem>
```

结束语 当今嵌入式市场日益扩大,全球竞争日趋激烈,嵌入式系统的开发成本、开发周期、质量显得越来越重要,而其开发设计能力远远落后于系统复杂性需求,传统的嵌入式

开发方法难以解决面临的新问题,通用计算机系统的软件体系结构标准又难以直接运用,嵌入式系统的发展面临新的机遇和挑战。良好的软件体系结构和合适的开发方法,对于系统开发的成功具有重大的意义。

本文是对涉及软件体系结构、软件开发过程进行改进的多个嵌入式系统项目的总结,解决了嵌入式软件在一定范围内的跨平台性。从实际应用效果来看,显著提高了开发效率,取得了较好的效益。专业人员和领域开发人员反应积极,并不断就开发细节提出改进意见,希望能进一步完善,形成规范。文中介绍了基于嵌入式操作系统应用软件开发宏观体系结构的设计和表达,这方面的研究工作还有待进一步的努力,才能加以规范并推广使用。

参考文献

- 1 Li Yi, Wang Zhiyan, Zeng Haizan. Correlation filter: an accurate approach to detect and locate low contrast character strings in complex table environment[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2004, 26(12): 1639~1644
- 2 李谊, 王知衍, 曾海赞. 采用混合神经网络高精度提取机票字符. 计算机工程与应用[J], 2004, 40(8): 209~211
- 3 严和平, 邓荣佐, 王知衍. 一种用于嵌入式应用开发的软件体系结构[J]. 计算机应用研究, 2005(11): 89~91
- 4 潘清. 操作系统体系结构研究[J]. 计算机科学, 2000, 27(4): 14~17
- 5 Buschmann F, Meunier R, Rohnert H, Sommerlad P, Stal M. 面向模式的软件体系结构 卷 1: 模式系统[M]. 北京: 机械工业出版社, 2003. 1
- 6 杨美清, 王千祥, 梅宏, 陈兆良. 基于复用的软件生产技术[J]. 中国科学 E 辑, 2004, 31(4): 363~37