文章编号:1671-4598(2017)05-0143-04 DOI:10.16526/j.cnki.11—4762/tp.2017.05.040 中图分类号:TP3 文献标识码:A

基于 ARM—Core4x 嵌入式软件平台的实现方法

王永刚，应站煌，陈玉峰
（许继电气股份有限公司，河南 许昌 461000）

摘要：针对基于 ARM—Core4x 微处理器的管脚功能可配置的特点，降低应用业务模块对硬件驱动层关注度及业务代码与驱动代码耦合性，提出了一种基于 ARM—Core4x 微处理器的嵌入式软件开发平台设计方法，该平台可提供了基于 POSIX 标准的硬件操作接口和操作系统及网络协议基本应用的接口，并通过子模块间逻辑调用机制的合理设计，实现了平台核心模块和应用模块独立编译且运行期间相互调用功能模块，实现开发平台的整体功能；通过对该平台软件的应用模块二次开发，减少用户应用开发对 ARM 微处理器硬件的关注度，使用户开发重点聚焦在应用及业务的实现上；实践证明，该软件平台可方便应用在低压继电保护装置以及智能化设备产品的开发，提高了用户应用开发效率和降低开发难度。

关键词：ARM—Core4x；嵌入式系统；可视化开发；中间层

Implementation Method of Embedded Software Platform Based on ARM—Core4x

Wang Yonggang，Ying Zhanhuang，Chen Yufeng
（XJ Electric Co.，Ltd. Automatic Protection Corporation，Xuchang 461000，China）

Abstract：According to the characteristics of the pin configuration function based on ARM—Core4x microprocessor can be reduced，application service module of hardware driver layer attention and business code and driver code coupling，proposes a design method of embedded software development platform based on ARM—Core4x microprocessor，the platform can provide a standard POSIX hardware interface and operating system and network protocol the basic application interface based on reasonable design and the sub module between logical call mechanism，the platform to achieve the core module and Application module independently compile and run the mutual call function module，the realization of the overall function of development platform. Through the application of the platform software module two development，reduce the user application development of ARM microprocessor hardware attention，the user development focus on the application and implementation of the business. Practice has proved that the software platform can be used in low voltage relay protection devices and intelligent equipment product development，improve the user application development efficiency and reduce the difficulty of development.

Keyword：ARM—Core4x；embedded system；visualization development；middle layer

0 引言

近年来，嵌入式应用领域中以 ARM 处理器发展最为突出，ARM 被公认为业界领先、优秀的 32 位嵌入式处理器结构[1]。ARM 系列处理器凭借高性能、低成本和低功耗等特点，在嵌入式应用领域中占据了绝对的市场份额，在建筑安全系统、工厂自动化、家庭娱乐、太阳能面板、测试与计量得到广泛的应用[2-4]。随着业务应用领域的扩展，在产品开发时面临业务多样化，业务功能实现代码与底层驱动代码耦合性高，不利于公共业务功能的代码的重用和迭代，对其产品质量产生不利影响。

本文提出了一种基于 ARM—M4 处理器的软件开发平台设计方法，通过将 ARM 硬件及通用功能开发和应用业务开发分离，提供一种易用、通用的开发平台，使产品开发者关注业务的实现，而平台开发者注重于硬件驱动及平台的开发，两者协作一致完成产品的开发，有效缩短产品开发周期，并有利于开发平台复用性。

收稿日期：2016-11-03； **修回日期**：2017-01-05。

作者简介：王永刚(1969-)男，上海人，博士，高级工程师，主要从事嵌入式系统微处理器方向的研究。

1 软件框架总体设计

1.1 设计要求

为解决业务模块代码与硬件驱动层代码的耦合性，业务代码模块中不应含有驱动层代码，硬件驱动层代码由平台模块实现，并其功能函数以指针方式集中存放可查找到数据特殊区域，以便应用业务模块所调用。因此，需要平台模块和应用模块相互独立，可单独编译和升级，以达到降低应用业务代码与驱动代码耦合性。

为实现平台模块和应用模块升级或更新，需要独立升级模块，升级模块独立平台和应用模块之外，以便其实现升级功能。

为了使整个开发平台框架能够尽可能满足不同产品硬件配置要求，平台模块中驱动设计时应将驱动有关功能函数和硬件配置进行必要分离，以便于不同应用背景下对硬件进行选择性配置功能的实现，提高开发平台应用范围。

1.2 总体框架

基于以上设计原则的考虑，软件框架设计如图 1 所示，整个软件框架包括 4 个部分：驱动库、引导模块、平台模块和应用模块。

1）驱动库模块。驱动模块是以库文件方式，提供给引导模块、平台模块所使用。

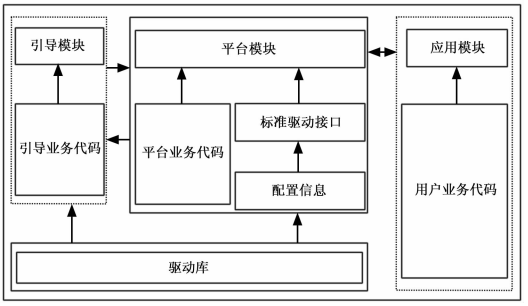

图 1 总体框架结构示意图

2）引导模块。引导模块负责平台模块、应用模块的下载。

3）平台模块。平台模块是该软件平台的核心模块，也是应用模块工作环境和资源，能够提供给用户足够多的接口，以便用户开发业务应用。

4）应用模块。应用模块是用户主要业务功能开发模块，但由于要调用平台库所提供的资源以及系统正常工作的需要，必须满足一些设计框架的要求，否则应用模块无法正常的工作。

从逻辑结构来说，引导模块可以视为一个单独应用。在引导模块中，可实现程序下载和授权管理以及引导模块向平台模块的切换。鉴于开发的分层设计的考虑，平台模块和应用模块，虽以独立的应用程序而存在，但在逻辑上平台模块是应用模块实现业务功能基础模块。

2 各子模块的设计

2.1 驱库设计

驱动库主要提供给引导模块和平台模块进行有关硬件操作的驱动接口。从引导模块的通用性和平台模块的可移植性考虑，驱动库提供对外硬件接口在接口定义要统一性，即不同 ARM－Core4x 处理器，对外提供相同的接口驱动库，驱动接口的差异性仅在驱动库执行体不同。驱动接口的统一化可有效简化不同 ARM－Core4x 处理器的引导模块和平台模块的生成过程，有利于各子模块的通用性和可移植性设计目标的实现。

ARM－Core4x 微处理器的管脚功能具备可配置性，即同一管脚可配置模拟量采样开入，亦可配置成开出管脚。而管脚的可配置性决定于产品业务的硬件要求，因此基于业务应用的硬件驱动只能在平台模块，故驱动库接口仅限于单个类型硬件驱动的实现，而板卡类的驱动只能在驱动库接口基础之上实现。

在硬件驱动库内容上，包括 ARM－Core4x 处理器支持所有的硬件驱动，即按照可允许最大化配置的硬件类型进行硬件驱动的设计，即支持所有可能配置方式。

在 ARM－Core4x 生产厂商中，大部分已经提供了基于微处理器硬件驱动库或者提供基于这些硬件各个操作接口，这些接口需要通过适当二次封装，形成统一、规范对外驱动接口，其开发工作量相对较小。

2.2 引导模块设计

为了软件平台的通用化和兼容性目标的实现，引导模块的设计必须保证其通用化和可移植性。考虑到 ARM－Core4x 微处理器硬件的差异性，尤其是各厂商生产基于 ARM－Core4x 标准的微处理器，但其硬件配置和硬件控制寄存器地址，均存在一定差异性[5-6]。因此，引导业务代码中不应含有硬件的任何信息，否则，无法保证引导业务代码的通用性。

为了实现引导业务代码通用性，将其引导业务代码中硬件操作接口进行抽象化，且在初始化时将硬件抽象化接口进行实例化。抽象化处理硬件操作接口，仅能保证引导代码的通用性，并不能保证编译后的引导程序可以适用任何硬件环境下引导功能，但可以简单通过替换不同 ARM－Core4x 处理器的驱动库，可以方便生成适用不同 ARM－Core4x 处理器的引导模块。

硬件操作接口抽象化对象范围需要充分考虑引导模块需要的实现功能以及与功能相关的硬件对象。引导模块的基本功能是下载功能和引导功能。上电后需要从引导模块运行，并在许可协议合法和平台模块满足安全运行的条件下，切换到平台模块运行进行用户应用业务的处理。因此，与引导业务模块相关的硬件包括：①以太网或者 UART，下载功能所必须通信硬件；②FLASH 模块，平台和应用程序保存介质；③其他模块，如时钟模块、中断控制模块、网络协议模块，处理器运行所必需基本模块。

从实现业务需要考虑，引导模块还包括下载通信协议等其他中间层模块，通信协议中涉及到通信硬件驱动也采用抽象化硬件操作接口，并在通信协议初始时对抽象化硬件操作接口进行实例化，保证下载通信协议通用性。在面临不同的微处理器硬件的引导模块的移植时，可以替换硬件驱动库，可简单快捷生成基于新的硬件引导模块。

2.3 平台模块的设计

为了实现用户业务的实现，作为资源库和应用环境的平台模块包含以下几部分功能：

（1）标准的硬件操作接口。硬件的操作是应用模块实现其业务最基本的硬件操作要求，也是平台模块必须提供基本功能之一。为了减少用户模块开发时对硬件操作难度，平台模块在驱动库的基础上，并结合硬件设计要求，提供基于 POSIX 标准的硬件操作接口。为了实现硬件驱动库的标准化操作，所有的硬件均支持 open、close、read、write 和 iocrl 五种类型接口，各个接口的真正的实现与否视具体的硬件类型而定。

（2）操作系统接口。操作系统接口提供有关任务管理、信号量、互斥量、邮箱、事件组等系统级接口。考虑到成本问题，本平台操作系统选择 FreeRTOS。鉴于内存动态申请不利于平台编译期间发现内存溢出问题[7]，移植 FreeRTOS 操作系统时，对任务内存管理进行适当的调整，所有内存申请的由动态申请修改为静态申请，并规定任务创建个数限制，以免出现程序在编译期和运行期出现内存使用不一致性，保证系统的可靠性。

（3）网络协议解析接口。为了控制平台程序大小，网络协议层选用 UIP 协议代码，UIP 具备代码量少、占用内存少有利于在资源稀少的 ARM 处理器中使用[8-10]。为了进一步控制协议层内存使用，对 TCP 协议所中由内置基于连接个数的数据缓存内存申请，改由外部用户使用自己申请，增强 UIP 对用户要求连接个数的适应性。

（4）数据表操作接口。数据表操作接口提供基于数据库方式的数据访问方式，并针对不同任务访问数据表采取安全措施，提高数据表访问的安全性。

（5）其他功能模块。如远程调试功能模块支持、自我发现协议实现。

2.4　应用模块的设计

应用模块是对用户开放模块，其功能是在遵循基本应用框架之上，实现用户业务及应用需要。应用模块框架包括 3 个主要功能模块：

（1）用户初始化模块。

用户初始化模块主要为用户提供一些初始化操作，如初始化应用需要内存以及应用任务的创建。基于业务创建的任务处理时，需要统筹考虑各任务的优先级别，以便在整体上保持较高运行效率。

（2）用户层驱动加载模块。

用户层驱动加载模块，提供用户设计驱动模块在符合一些特殊要求情况下，可以被被平台库模块所加载，用户模块以标准设备操作的方式去控制设备，从而保证用户操作硬件设备标准化。

（3）用户业务循环处理模块。

用户业务循环处理模块，用于非任务模式下常规事务处理与调用，循环模块会被平台模块相应部分执行。在操作系统运行的条件下，本循环模块没有被调用的可能，其原因在于操作系统的任务调度是通过中断调用和任务调度策略相结合运行模式下，平台的循环模块体几乎没有时间被调用可能，从而导致本模块被调用机会很少。

3　关键问题的实现

3.1　平台模块业务流程的设计

开发平台是一个完整业务处理的平台，其子模块承担各自功能，并通过业务流程来规定各子模块主要业务实现流程，以其整体上实现整个开发平台设计目标。

开发平台的业务流程图如图 2 所示，其中引导模块和平台模块进行存在切换控制功能，而平台模块和应用模块关系相对复杂，这种复杂性主要体现：应用模块中主要功能模块被平台模块所调用而运行，而应用模块中任何功能模块都可随时调用平台模块资源库接口函数，以便应用业务功能的实现，因此需要对平台和应用模块之间调用控制逻辑进行合理规划和设计。

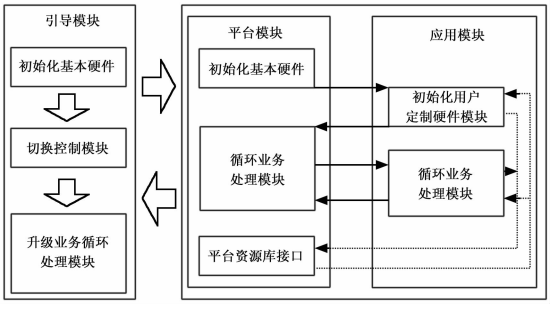

图 2　开发平台业务流程图

3.2　子模块间逻辑调用的实现

1）平台模块与引导模块。

引导模块与平台模块两者处理关系为切换功能和引导功能，引导模块在启动以后，在满足一定的条件下，可直接切换到平台模块运行，而平台模块在外部某种干预下，可以切换到引导模块。前一种是整个软件正常运行的模式，而后一种模式是进行平台模块或应用模块进行升级程序模式。切换方式采用直接跳转相应模块入口地址实现。

2）平台模块与应用模块。

平台模块与应用模块逻辑关系为接口函数相互调用，即相互独立模块接口函数相互调用的实现。平台模块和应用模块关系复杂性是由平台与应用模块独立设计以及应用与平台模块相互调用功能要求所决定的，因此需要对平台和应用模块进行严谨的设计，保证相互调用功能的实现。

图 3 说明模块间调用功能实现逻辑方法，应用模块和平台模块相互调用逻辑的实现步骤如下：

A）在平台模块内部，声明一数据结构，其成员为开发接口函数的类型。定义该数据结构的常量，其成员为开发接口函数的指针，并将该常量数据结构的地址存放到平台模块中断向量表一个固定位置中。

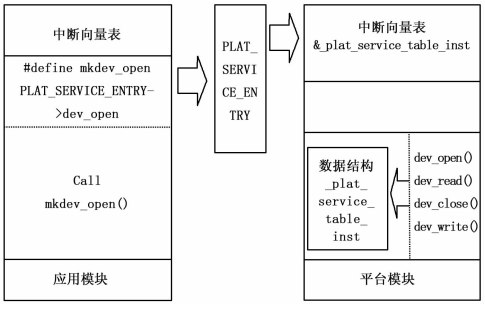

图 3　应用模块调用平台实现图

B）应用模块调用平台模块接口函数时，通过以下几个步骤：首先，获取平台模块的存放接口函数的数据结构地址的中断向量表，通过其内容获得接口函数数据结构所在的位置；其次，通过声明获取平台模块接口函数在数据结构中偏移位置，进而获取到平台模块中被调用接口函数地址，被调用函数执行完毕，继续处理应用模块其他相关业务。

通过以上步骤，调用模块可方便调用到被调用模块提供的接口函数，进一步为实现用户业务创建提供必要资源和驱动支持。

平台模块调用应用模块实现机制和应用模块调用平台模块实现机制相似，差别仅在于调用函数接口对象的数据结构成员定义、数据结构和数据结构地址存放的中断向量表的位置不同。为了避免平台模块和应用模块相互调用而导致模块内存数据被改写，平台模块和应用模块分配独立内存模块，确保平台模块和应用模块各自内存数据不会被对方修改，从而保证平台模块和应用模块相互调用功能的实现。

3.3　可视化开发模式的实现

可视化开发模式是指用户将平台模块提供的各类接口作为资源，加载到可视化开发工具中，并在可被调用。为了保证可视化开发工具生成的代码能有效运行，并具备调用平台模块提供各类资源接口函数处理应用业务，可视化开发工具生成的代码必须是在一定应用模块框架基础进行操作。

图 4 为可视化开发工具与平台软件逻辑生成示意图。平台软件提供基本接口资源，而可视化工具将接口作为资源加载到工具中，开发时可通过拖入功能函数到目标视图区，并对函数输入输出进行适当图形连接，同时生成与此图形逻辑对应的 C 代码，写入到相应的用户业务开发模块框架中，调用相应编译器最终完成由图形到代码、代码到机器码生成的任务。

4　结论

通过应用软件平台设计开发方法，较方便实现基于 TI 和 Freescale 厂商不同硬件的软件开发平台，实现平台模块和应用模块独立编译和单独升级，降低应用业务代码和底层驱动代

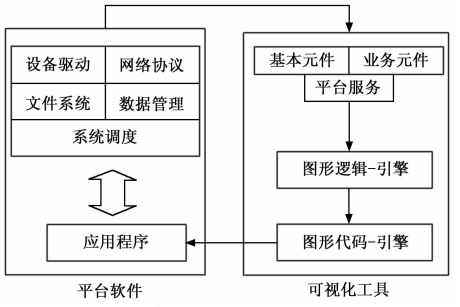

图 4 平台软件与可视化工具逻辑

码耦合性，加快产品开发进度和提高开发质量。

实践证明，通过分层分模块设计，软件平台实现系统预期设计目标。通过配置信息的调整，可以满足基于同一 ARM－Core4x 微处理器而不同配置板卡产品开发的需要。平台软件分模块设计，可引导产品开发进行分层开发，平台人员侧重平台的开发，而产品应用人员开发注重业务的实现，整体提高产品开发效率。

参考文献：

[1] 窦晓波，徐 科，胡敏强，等. 基于 ARM 处理器的低压微机保护装置 [J]. 电力系统自动化，2005，29 (20)：93－96，99.
[2] 孟艳清，赵宏伟，邹育霖. 基于 ARM 的开关柜智能监控装置研究 [J]. 高压电器，2014，50 (3)：29－35.
[3] 孙延岭，赵雪飞，张红芳，等. 基于 ARM 嵌入式系统的微型智能可编程控制器 [J]. 电力系统自动化，2010，34 (10)：101－103.
[4] 舒双宝，罗家融，王勤湧，等. 基于 DSP 和 ARM 便携式电能质量监测系统的设计与实现 [J]. 电力系统保护与控制，2010，38 (24)：185－189.
[5] 范蟠果，邢保毫，米晓亮，等. 基于嵌入式 S3C2440 系统 Bootloader 设计与实现 [J] 计算机测量和控制，2016，24 (9)：12－14.
[6] 吕 敏，沙 莎. 可视化编程数字图像处理平台的界面设计与实现 [J]. 计算机系统应用，2010，19 (9)：211－213.
[7] 郭开荣，温渤婴. FreeRTOS 在微机继电保护实验装置中的应用 [J]. 继电器，2006，34 (19)：4－6.
[8] 伊文斌，周贤娟，鄢化彪，等. Uip TCP/IP 协议分析及其在嵌入式系统中的应用 [J]. 计算机技术与发展，2007，17 (9)：240－243.
[9] 周邵文. 一种基于 UIP 协议栈的多应用层协议支持方案的设计 [J]. 电力系统保护与控制，2010，38 (9)：109－112.
[10] 徐林峰，李 星. 基于 ARM Cortex－M3 微控制器与 QDUC 的嵌入式激励器设计 [J]. 计算机测量和控制，2016，24 (2)：182－185.

(上接第 142 页)

表 1 测试用例

用例	水位数据个数	振幅相位数据个数	水位个数/振幅个数
T1	37272	311	120
T2	47496	132	360
T3	59430	180	330
T4	156600	1305	120
T5	8760	25	350
T6	90240	3760	24
T7	90240	250	361

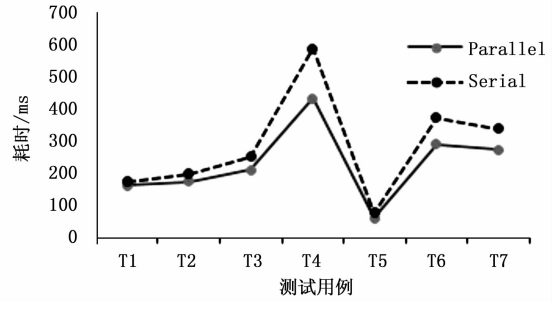

图 6 测试结果图

在图 6 中，横坐标为测试用例，纵坐标为串行程序或并行程序的运行耗时。无论是串行程序还是并行程序，都是随着处理数据量的增加而处理时间有所增长。总的来看，程序并行化后，系统的运行时间有所缩短，在所有的测试用例中，并行程序的耗时都比串行程序低，并行程序性能明显高于串行程序。在测试用例 T4 中，并行性能大大高于串行程序性能，表明当程序运算任务繁重时，并行程序可以显著改善串行程序的性能，相反并行程序并不具有优势，如 T5 用例，水位数据和振幅相位数据都最少，系统并行化性能提升非常有限。T6、T7 的水位数据相同，T6 振幅相位数据多于 T7，换句话说，根据 3.2 节任务的划分，T6 中任务 t2 和 t3 负载较 T7 更均衡一些，因此 T6 并行性能明显高于 T7。

在实际测试也发现，如果任务数目过多或负载分配不均衡会大大影响并行程序的性能。

5 结论

本文基于 C#.NET 程序设计语言中任务机制实现了 Baytap－G 潮汐分析辅助软件并行绘制振幅、相位图功能，并就相关技术和关键细节进行阐述。程序并行化后，系统运行效率有了较大提升。但是相对于串行程序而言，并行编程仍然存在很大挑战，如何设计数目合理的任务、如何处理任务之间的关系、如何合理分配任务都会影响到系统的性能。任何一个环节处理不好，都有可能适得其反，这也是当前并行程序设计存在的主要难点。

参考文献：

[1] 张 晶，陈荣华，杨林章，等. 强震前形变潮汐异常判识与机理研究 [J]. 地震学报，2006，28 (2)：150－157
[2] 廖 欣，刘春平，杨贤和，等. 承压井水位对含水层潮汐应力响应是否满足不排水条件的检验 [J]. 地震学报，2011，33 (2)：234－242，270
[3] 熊 峰，张 起，查 斯，等. 海拉尔地震台重力潮汐扰动分析 [J]. 山西地震，2015 (2)：21－23
[4] 陈志遥，吕品姬，李正媛，等. 汶川 Ms8.0 地震前的潮汐变化分析 [J]. 大地测量与地球动力学，2009，29 (4)：48－50
[5] 陈晓东，孙和平. 一种新的重力潮汐数据预处理和分析方法 [J]. 大地测量与地球动力学，2002，22 (3)：83－87
[6] Tamura Y，Sato T，et al. A procedure for tidal analysis with a Bayesian information criterion [J]. Geophysical Journal International，1991 (104)：507－516
[7] 潘佳宾，胡越黎，陈晓君，等. 基于多核的车牌识别的架构实现 [J]. 计算机测量与控制，2015，23 (1)：213－217.