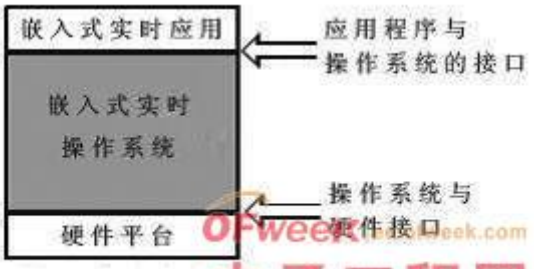
嵌入式操作系统 HAL 原理及 BSP 实现方法

随着计算机软硬件技术的快速发展,出现了越来越多的便携设备和智能设备。这些设备中通常包含控制用的 CPU 和相应的操作系统;这类特殊的计算机系统叫做嵌入式实时系统。嵌入式实时系统以其简洁高效等特点在计算机、通信等领域中广泛使用。

由于嵌入式实时系统应用环境的特殊性,因此在设计实现过程中存在着许多特殊问题。其中,操作系统及其他系统软件模块与硬件之间的接口形式是嵌入式实时系 统的主要特征和系统设计过程中的必需环节,也是影响嵌入式系统应用前景的关键问题。经过近些年的发展,随着通用嵌入式操作系统技术的日趋成熟和应用的不断 扩大,一种统一的接口形式得到广泛的认可和应用,这就是通常所说的板级支持包,即 BSP。

1 嵌入式系统硬件抽象层的原理

1.1 硬件抽象层的引入



嵌入式实时系统作为一类特殊的计算机系统自底向上包含三个部分,如图 1 所示。

(1)硬件环境:是整个嵌入式实时操作系统和实时应用程序运行的硬件平台;不同的应用通常有不同的硬件环境;硬件平台的多样性是嵌入式系统的一个主要特点。

(2)嵌入式实时操作系统——RTOS:完成嵌入式实时应用的任务调度和控制等核心功能,具有内核较精简、可配置、与高层应用紧密关联等特点。嵌入式操作系统具有相对不变性。

(3)嵌入式实时应用程序:运行于操作系统之上,利用操作系统提供的实时机制完成特定功能的嵌入式应用。不同的系统需要设计不同的嵌入式实时应用程序。

由于嵌入式系统应用的硬件环境差异较大,因此,如何简洁有效地使嵌入式系统能够应用于各种不同的应用环境是嵌入式系统发展中所必须解决的关键问题。

经过不断的发展,原先嵌入式系统的三层结构逐步演化成为一种四层结构。这个新增加的中间层次位于操作系统和硬件之间,包含了系统中与硬件相关的大部分功 能。通过特定的上层接口与操作系统进行交互,向操作系统提供底层的硬件信息;并根据操作系统的要求完成对硬件的直接操作。由于引入了一个中间层次,屏蔽了 底层硬件的多样性,操作系统不再直接面对具体的硬件环境。而是面向由这个中间层次所代表的、逻辑上的硬件环境。因此,把这个中间层次叫做硬件抽象层 HAL(Hardware Abstraction Layer)。在目前的嵌入式领域中通常也把 HAL 叫做板级支持包 BSP(Board Support Package)。图 2 显示了引入 HAL 以后的嵌入式系统结构。BSP 的引入大大推动了嵌入式实时操作系统的通用化,从而为嵌入式系统的广泛应用提供了可能。

1.2 BSP 的特点与功能

HAL/BSP 的提出使通用的嵌入式操作系统及高层的嵌入式应用能够有效地运行于特定的、应用相关的硬件环境之上,使操作系统和应用程序能够控制和操作具体的硬件设备,完成特定的功能。因此,在绝大多数的嵌入式系统中,BSP 是一个必不可少的层次。

由于在系统中的特殊位置,因此 BSP 具有以下主要特点:

(1)硬件相关性

因为嵌入式实时系统的硬件环境具有应用相关性,所以,作为高层软件与硬件之间的接口,BSP 必须为操作系统提供操作和控制具体硬件的方法。

(2)操作系统相关性

不同的操作系统具有各自的软件层次结构,因此,不同的操作系统具有特定的硬件接口形式。

在实现上,BSP 是一个介于操作系统和底层硬件之间的软件层次,包括了系统中大部分与硬件相关的软件模块。在功能上包含两部分:系统初始化及与硬件相关的设备驱动。

2 BSP 的设计与实现

为实现上述两部分功能,设计一个完整的 BSP 需要完成两部分工作:

(1)设计初始化过程,完成嵌入式系统的初始化;

(2)设计硬件相关的设备驱动,完成操作系统及应用程序对具体硬件的操作。

2.1 嵌入式系统初始化以及 BSP 的功能

嵌入式系统的初始化过程是一个同时包括硬件初始化和软件(主要是操作系统及系统软件模块)初始化的过程;而操作系统启动以前的初始化操作是 BSP 的主要 功能之一。由于嵌入式系统不仅具有硬件环境的多样性,同时具有软件的可配置性,因此,不同的嵌入式系统初始化所涉及的内容各不相同,复杂程度也不尽相同。 但是初始化过程总是可以抽象为三个主要环节,按照自底向上、从硬件到软件的次序依次为:片级初始化、板级初始化和系统级初始化。

(1)片级初始化:主要完成 CPU 的初始化,包括设置 CPU 的核心寄存器和控制寄存器,CPU 核心工作模式以及 CPU 的局部总线模式等。片级初始化把 CPU 从上电时的缺省状态逐步设置成为系统所要求的工作状态。这是一个纯硬件的初始化过程。

(2)板级初始化:完成 CPU 以外的其他硬件设备的初始化。除此之外,还要设置某些软件的数据结构和参数,为随后的系统级初始化和应用程序的运行建立硬件和软件环境。这是一个同时包含软硬件两部分在内的初始化过程。

(3)系统级初始化:这是一个以软件初始化为主的过程,主要进行操作系统初始化。BSP 将控制转交给操作系统,由操作系统进行余下的初始化操作。包括加载和初始化与硬件无关的设备驱动程序,建立系统内存区,加载并初始化其他系统软件模块,比如网络系统、文件系统等;最后,操作系统创建应用程序环境并将控 制转交给应用程序的入口。

经过以上三个层次的操作,嵌入式系统运行所需要的硬件和软件环境已经进行了正确设置,从这里开始,高层的实时应用程序可以运行了。

需要指出:系统级初始化不是 BSP 的工作。但是,系统级初始化成功与否的关键在于 BSP 的前两个初始化过程中所进行的软件和硬件的正确设置,而且系统级初始化也是由 BSP 发起的。因此,设计 BSP 中初始化功能的重点主要集中在前两个环节。图 3 显示了嵌入式系统的初始化过程。

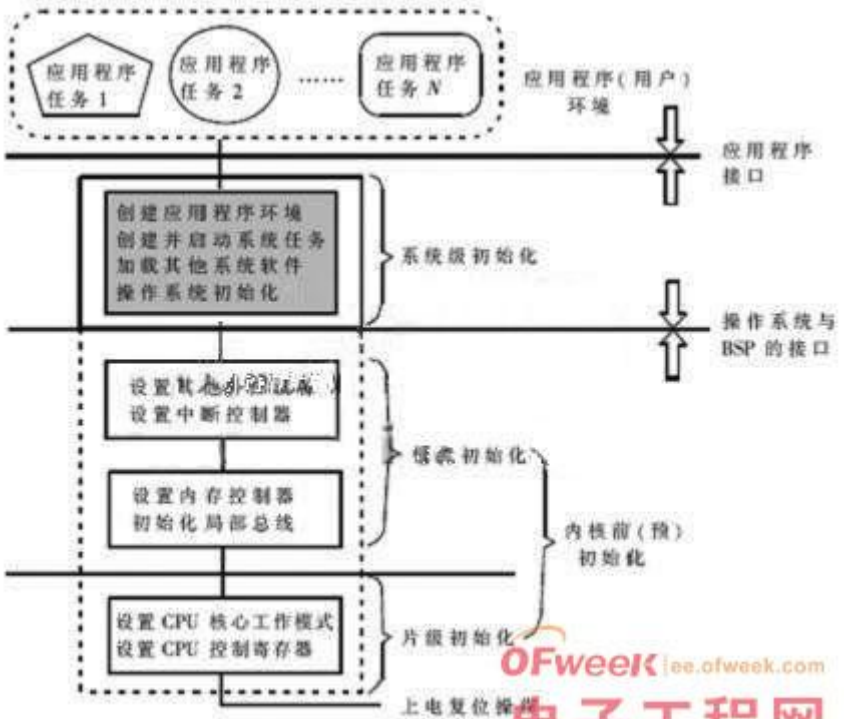
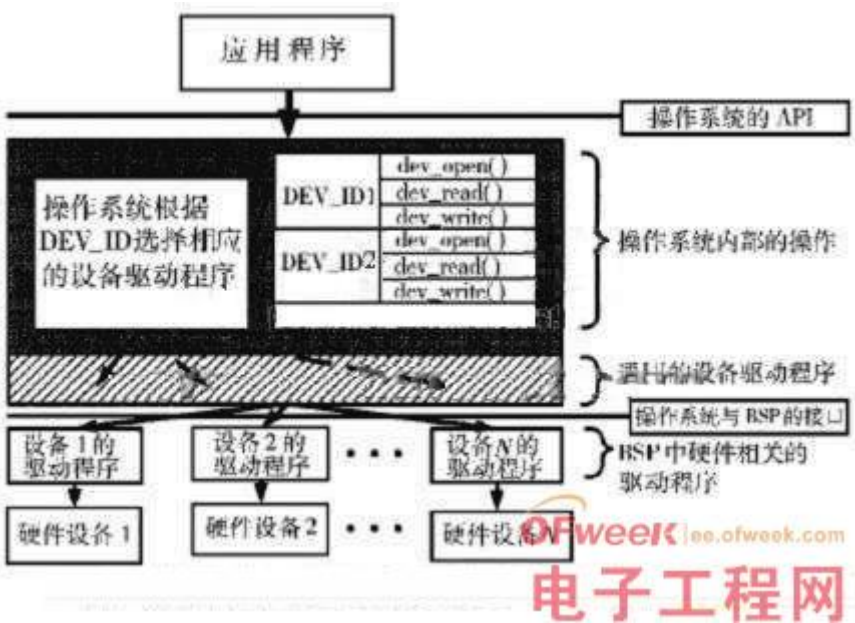


图 3 嵌入式系统的初始化过程及 BSP 的功能

2.2 硬件相关的设备驱动程序

BSP 另一个主要功能是硬件相关的设备驱动。与初始化过程相反,硬件相关的设备驱动程序的初始化和使用通常是一个从高层到底层的过程。

尽管 BSP 中包含硬件相关的设备驱动程序,但是这些设备驱动程序通常不直接由 BSP 使用,而是在系统初始化过程中由 BSP 把它们与操作系统中通用的设备驱动程序关联起来,并在随后的应用中由通用的设备驱动程序调用,实现对硬件设备的操作。设计与硬件相关的驱动程序是 BSP 设计中另一个关键环节。图 4 显示 了调用设备驱动程序时系统各个层次之间的关系。



3 设计实现 BSP 的方法

3.1 设计实现 BSP 的一般方法

因为 BSP 同时具有硬件相关性和操作系统相关性,是一个介于硬件与软件之间的中间层次。因此 BSP 的开发不仅需要具备一定的硬件知识,例如 CPU 的控 制、中断控制器的设置、内存控制器的设置及有关的总线规范等;同时还要求掌握操作系统所定义的 BSP 接口。另外,在 BSP 的初始化部分通常会包含一些汇编 代码,因此还要求对所使用的 CPU 汇编指令有所了解,例如 X86 的汇编和 PowerPC 的汇编指令等;对于某些复杂的 BSP 还要了解所使用的开发工具,例 如 GNU、Diab Data 等。

总之,开发 BSP 要求具备比较全面的软、硬件知识和必要的编程经验。由于设计实现的复杂性,在设计特定 BSP 时很少从零开始,而是采用以下两种快捷方法。

方法一:以经典 BSP 为参考

在设计 BSP 时,首先选择与应用硬件环境最为相似的参考设计,例如 Motorola 的 ADS 系列评估板等。针对这些评估板,不同的操作系统都会提供完整的 BSP,这些 BSP 是学习和开发自己 BSP 的最佳参考。针对具体应用的特定环境对参考设计的 BSP 进行必要的修改和增加,就可以完成简单的 BSP 设计。

下面以设计 pSOS 操作系统的 BSP 初始化过程为例。pSOS 系统初始化的层次非常清晰,与初始化过程相对应的是以下三个文件:

1)init.s :对应于片级初始化;完成 CPU 的初始化操作,设置 CPU 的工作状态;

2)board.c :对应于板级初始化;继续 CPU 初始化,并设置 CPU 以外的硬件设备;

3)sysinit.c :对应于系统级初始化;完成操作系统的初始化,并启动应用程序。

以参考 BSP 为切入点,针对初始化过程的具体环节,在对应的文件中进行某些参数的修改及功能的增加就可以实现 BSP 的系统初始化功能。

因为 BSP 具有操作系统相关性,因此,不同的操作系统会使用不同的文件完成类似的初始化操作。

BSP 中硬件相关的设备驱动程序随操作系统的不同而具有比较大的差异,设计过程中应参照操作系统相应的接口规范。

方法二:使用操作系统提供的 BSP 模板

除了提供某些评估板的 BSP 以外,很多操作系统还提供相应的 BSP 模板(一组需要编写的文件),根据模板的提示也可以逐步完成特定 BSP 的设计。

相比较而言,第一种方法最为简单快捷。因此,在实际的设计过程中,通常以第一种方法为主,同时结合使用第二种方法。

在设计实现 BSP 两部分功能时应采用以下两种不同方法:

(1)“自底向上”地实现 BSP 中的初始化操作:从片级初始化开始到系统级初始化;

(2)“自顶向下”地设计硬件相关的驱动程序:从 API 开始,到操作系统内部的通用设备驱动程序,再到 BSP 内部的硬件相关的设备驱动程序,最后到底层具体的硬件设备。

3.2 BSP 设计方法的不足与改进

从以上介绍的两种设计方法可以看出:目前 BSP 的设计与实现主要是针对某些特定的文件进行修改。这种方法比较原始,它不仅要求设计人员了解 BSP 的各个 组成部分及所对应的文件和相关参数的具体含义,还要求具备比较全面的软硬件知识。直接修改相关文件容易造成代码的不一致性,增加软件设计上的隐形错误,从 而增加系统调试和代码维护的难度。随着底层硬件功能的日益复杂,开发 BSP 所涉及的内容也越来越多。这种原始方法的不足之处也越来越突出。进行 BSP 设计 方法和工具的创新成为一个日益突出的问题。

解决这个问题的一个可行办法是:设计实现一种具有图形界面的 BSP 开发设计向导,由该向导指 导设计者逐步完成 BSP 的设计和开发,并最终由向导生成相应的 BSP 文件,而不再由设计人员直接对源文件进行修改。这样不仅可以大大缩短 BSP 的开发周 期,减少代码不一致性,而且系统排错、调试以及维护都很简单。

因此,这种方法是目前嵌入式领域中 BSP 设计的一个趋势和研究方向。但是,由于嵌入式系统硬 件环境的多样性,设计向导的实现仍需解决若干关键问题。为此,作者仍将在这一方面作进一步研究。

文中提出的方法在华环公司的宽带网络工程中得到实践和应用,并取得了非常良好的应用成果。