第 3 章　嵌入式系统

随着信息技术的发展，嵌入式系统的应用越来越广，同时，在我国软件产业发展的规划中，也把嵌入式系统应用软件作为一个重点发展方面。因此，系统架构设计师必须熟悉有关嵌入式系统的基础知识，掌握嵌入式系统架构设计技术。

根据考试大纲，本章要求考生掌握以下知识点：

（1）信息系统综合知识：包括嵌入式系统的特点、嵌入式系统的硬件组成与设计、嵌入式系统应用软件及开发平台、嵌入式系统网络、嵌入式系统数据库、嵌入式操作系统与实时操作系统。

（2）系统架构设计案例分析：包括实时系统和嵌入式系统特征、实时任务调度和多任务设计、中断处理和异常处理、嵌入式系统开发设计。

3.1　嵌入式系统概论

嵌入式系统是一种以应用为中心，以计算机技术为基础，可以适应不同应用对功能、可靠性、成本、体积、功耗等方面的要求，集可配置可裁减的软、硬件于一体的专用计算机系统。它具有很强的灵活性，主要由嵌入式硬件平台、相关支撑硬件、嵌入式操作系统、支撑软件以及应用软件组成。

3.1.1　嵌入式系统的特点

嵌入式系统具有以下特点：

（1）系统专用性强。嵌入式系统是针对具体应用的专门系统。它的个性化很强，软件和硬件结合紧密。一般要针对硬件进行软件的开发和移植，根据硬件的变化和增减对软件进行修改。

（2）软、硬件依赖性强。嵌入式系统的专用性决定了其软、硬件的互相依赖性很强，两者必须协同设计，以达到共同实现预定功能的目的，并满足性能、成本和可靠性等方面的严格要求。

（3）系统实时性强。在嵌入式系统中，有相当一部分系统对外来事件要求在限定的时间内及时做出响应，具有实时性。

（4）处理器专用。嵌入式系统的处理器一般是为某一特定目的和应用而专门设计的，通常具有功耗低、体积小、集成度高等优点，能够把许多在通用计算机上需要由板卡完成的任务和功能集成到芯片内部，从而有利于嵌入式系统的小型化和移动能力的增强。

（5）多种技术紧密结合。嵌入式系统通常是计算机技术、半导体技术、电力电子技术及机械技术与各行业的具体应用相结合的产物。通用计算机技术也离不开这些技术，但它们相互结合的紧密程度不及嵌入式系统。

（6）系统透明性。嵌入式系统在形态上与通用计算机系统差异很大，它的输入设备往往不是常见的鼠标和键盘之类的设备，甚至没有输出装置，用户可能根本感觉不到它所使用的设备中有嵌入式系统的存在，即使知道也不必关心这个嵌入式系统的相关情况。

（7）系统资源受限。嵌入式系统为了达到结构紧凑、可靠性高及降低系统成本的目的，其存储容量、I/O 设备数量和处理器的处理能力都比较有限。

3.1.2 实时系统的概念

简单地说，实时系统可以看成对外部事件及时响应的系统。现实世界中，并非所有的嵌入式系统都具有实时特性，所有的实时系统也不一定都是嵌入式的。但这两种系统并不互相排斥，兼有这两种特性的系统称为实时嵌入式系统（Real-Time Embedded System，RTES），通常简称为实时系统。我们先介绍与 RTES 相关的几个概念。

（1）逻辑（或功能）正确：指系统对外部事件的处理能够产生正确的结果。

（2）时间正确：指系统对外部事件的处理必须在预定的周期内完成。

（3）死线（Deadline）：指系统必须对外部事件处理的最迟时间界限，错过此界限可能产生严重后果。通常，计算必须在到达死线前完成。

（4）实时系统：指功能正确和时间正确同时满足的系统，二者同等重要。换言之，实时系统有时间约束并且是死线驱动的。但是在某些系统中，为了保证功能的正确性，有可能牺牲时间的正确性。

根据对错失死线的容忍程度不同，可以将 RTES 分为软 RTES 和硬 RTES。

（1）硬 RTES：必须满足其灵活性接近零死线要求的 RTES。死线必须满足，否则就会产生灾难性后果，并且死线之后得到的处理结果或是零级无用，或是高度贬值。在硬 RTES 中，错失死线后的处理结果价值为零，错失死线的惩罚是灾难性的。

（2）软 RTES：必须满足死线的要求，但是有一定灵活性。死线可以包含可变的容忍等级、平均的时间死线，甚至是带有不同程度的可接受性的响应时间的统计分布。在软 RTES 中，错失死线后处理结果的价值根据应用的性质随时间按某种关系下降，死线错失不会导致系统失败。由于一个或多个错失的死线对软 RTES 的运行没有决定性的影响，一个软 RTES 不必预测是否可能有悬而未决的死线错失。相反，软 RTES 在探知到错失一个死线时可以启动一个恢复进程。

希赛教育专家提示：在 RTES 中，任务的开始时与同死线或完成时间同样重要，由于任务缺少需要的资源（例如，CPU 和内存等），就有可能阻碍任务的开始并直接导致错失任务的完成死线，因此，死线问题演变成资源的调度问题。

3.2　嵌入式系统的基本架构

嵌入式系统一般都由软件和硬件两个部分组成，其中嵌入式处理器、存储器和外部设备构成整个系统的硬件基础。嵌入式系统的软件部分可以分为 3 个层次，分别是系统软件、支撑软件和应用软件，其中系统软件和支撑软件是基础，应用软件则是最能体现整个嵌入式系统的特点和功能的部分。

3.2.1　硬件架构

微处理器是整个嵌入式系统的核心，负责控制系统的执行。外部设备是嵌入式系统同外界交互的通道，常见的外部设备有 Flash 存储器、键盘、输入笔、触摸屏、液晶显示器等，在很多嵌入式系统中还有与系统用途紧密相关的各种专用外设。嵌入式系统中经常使用的存储器有 3 种类型，分别是 RAM、ROM 和混合存储器。系统的存储器用于存放系统的程序代码、数据和系统运行的结果。

嵌入式系统的核心部件是各种类型的嵌入式处理器，根据目前的使用情况，嵌入式处理器可以分为如下几类：

（1）嵌入式微处理器。由通用计算机中的 CPU 演变而来，在功能上跟普通的微处理器基本一致，但是它具有体积小、功耗低、质量轻、成本低及可靠性高的优点。通常，嵌入式微处理器和 ROM（Read Only Memory，只读存储器）、RAM（Random Access Memory，随机存取存储器）、总线接口及外设接口等部件安装在一块电路板上，称为单板计算机。

（2）嵌入式微控制器。又称为单片机，整个计算机系统都集成到一块芯片中。嵌入式微控制器一般以某一种微处理器内核为核心，芯片内部集成有存储器、总线、总线逻辑、定时器/计数器、监督定时器、并口/串口、数模/模数转换器、闪存等必要外设。与嵌入式微处理器相比，嵌入式微控制器的最大特点是单片化，因而体积更小、功耗和成本更低，可靠性更高。

（3）嵌入式数字信号处理器。一种专门用于信号处理的处理器，DSP（Digital Signal Processor，数字信号处理器）是芯片内部采用程序和数据分开的结构，具有专门的硬件乘法器，广泛采用流水线操作，提供特殊的 DSP 指令，可以用来快速实现各种数字信号的处理算法。目前，DSP 在嵌入式系统中使用非常广泛，如数字滤波、快速傅里叶变换及频谱分析等。

（4）嵌入式片上系统。一种在一块芯片上集成很多功能模块的复杂系统，例如，把微处理器内核、RAM、USB（Universal Serial Bus，通用串行总线）、IEEE 1394、Bluetooth（蓝牙）等集成到一个芯片中，构成一个嵌入式片上系统，从而大幅度缩小了系统的体积、降低了系统的复杂度、增强了系统的可靠性。在大量生产时，生产成本也远远低于单元

部件组成的电路板系统。根据用途不同，嵌入式片上系统可以分为通用片上系统和专用片上系统两类。专用类的嵌入式片上系统一般是针对某一或某些系统而设计的。

3.2.2 软件架构

随着嵌入式技术的发展，特别是在后 PC 时代，嵌入式软件系统得到了极大的丰富和发展，形成了一个完整的软件体系，如图 3-1 所示。这个体系自底向上由 3 部分组成，分别是嵌入式操作系统、支撑软件和应用软件。

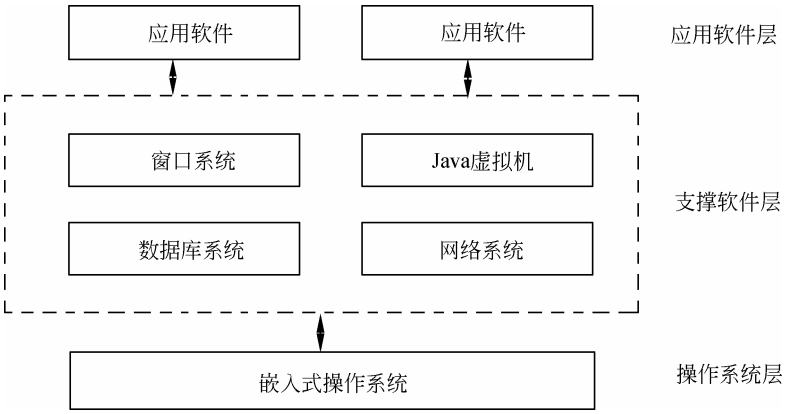

图 3-1 嵌入式系统的软件架构

嵌入式操作系统（Embedded Operating System，EOS）由操作系统内核、应用程序接口、设备驱动程序接口等几部分组成。嵌入式操作一般采用微内核结构。操作系统只负责进程的调度、进程间的通信、内存分配及异常与中断管理最基本的任务，其他大部分的功能则由支撑软件完成。

嵌入式系统中的支撑软件由窗口系统、网络系统、数据库管理系统及 Java 虚拟机等几部分组成。对于嵌入式系统来讲，软件的开发环境大部分在通用台式计算机和工作站上运行，但从逻辑上讲，它仍然被认为是嵌入式系统支撑软件的一部分。支撑软件一般用于一些浅度嵌入的系统中，如智能手机、个人数字助理等。

嵌入式系统中的应用软件是系统整体功能的集中体现。系统的能力总是通过应用软件表现出来的。

3.3 嵌入式操作系统

EOS 是指运行在嵌入式计算机系统上支持嵌入式应用程序的操作系统，是用于控制和管理嵌入式系统中的硬件和软件资源、提供系统服务的软件集合。EOS 是嵌入式软件的一个重要组成部分。它的出现提高了嵌入式软件开发的效率，提高了应用软件的可移

植性，有力地推动了嵌入式系统的发展。

随着各种类型 EOS 的成熟和发展，与通用系统的开发方法相似，基于操作系统的开发方案逐渐成为开发的主流。EOS 作为应用软件的运行平台，已经成为许多嵌入式系统的关键。

3.3.1　特点与分类

与通用操作系统相比，EOS 主要有以下特点：

（1）微型化。EOS 的运行平台是嵌入式计算机系统。这类系统一般没有大容量的内存，几乎没有外存，因此，EOS 必须做得小巧，以尽量少占用系统资源。为了提高系统速度和可靠性，嵌入式系统中的软件一般都固化在存储器芯片中，而不是存放在磁盘等载体中。

（2）代码质量高。在大多数应用中，存储空间依然是宝贵的资源，这就要求程序代码的质量要高，代码要尽量精简。

（3）专业化。嵌入式系统的硬件平台多种多样，处理器的更新速度快，每种都是针对不同的应用领域而专门设计。因此，EOS 要有很好适应性和移植性，还要支持多种开发平台。

（4）实时性强。嵌入式系统广泛应用于过程控制、数据采集、通信、多媒体信息处理等要求实时响应的场合，因此实时性成 EOS 的又一特点。

（5）可裁减、可配置。应用的多样性要求 EOS 具有较强的适应能力，能够根据应用的特点和具体要求进行灵活配置和合理裁减，以适应微型化和专业化的要求。

嵌入式操作系统的实时性上，可以分为实时嵌入式操作系统（Real-Time embedded Operating System，RTOS）和非实时嵌入式操作系统两类。

（1）实时嵌入式操作系统。RTOS 支持实时系统工作，其首要任务是调度一切可利用资源，以满足对外部事件响应的实时时限，其次着眼于提高系统的使用效率。RTOS 主要用在控制、通信等领域。目前，大多数商业嵌入式操作系统都是 RTOS。与通用操作系统系统相比，RTOS 在功能上具有很多特性。RTOS 和通用操作系统之间的功能也有很多相似之处，如它们都支持多任务，支持软件和硬件的资源管理以及都为应用提供基本的操作系统服务。RTOS 特有的不同于通用操作系统的功能主要有：满足嵌入式应用的高可靠性；满足应用需要的上、下裁减能力；减少内存需求；运行的可预测性；提供实时调度策略；系统的规模紧凑；支持从 ROM 或 RAM 上引导和运行；对不同的硬件平台具有更好的可移植性。

（2）非实时嵌入式操作系统。这类操作系统不特别关注单个任务响应时限，其平均性能、系统效率和资源利用率一般较高，适合于实时性要求不严格的消费类电子产品，如个人数字助理、机顶盒等。

3.3.2 一般结构

与通用计算机系统上的操作系统一样，EOS 隔离了用户与计算机系统的硬件，为用户提供了功能强大的虚拟计算机系统，如图 3-2 所示。EOS 主要由应用程序接口、设备驱动、操作系统内核等几部分组成。

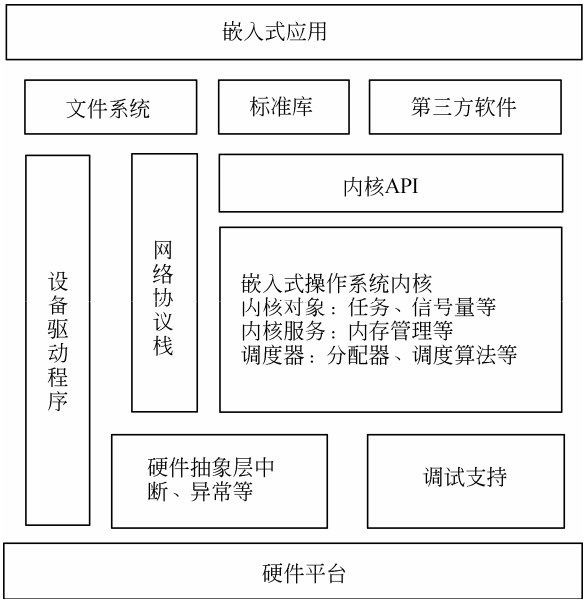

图 3-2 EOS 的一般结构

EOS 是一个按时序方式调度执行、管理系统资源并为应用代码提供服务的基础软件。每个 EOS 都有一个内核。另一方面，EOS 也可以是各种模块的有机组合，包括内核、文件系统、网络协议栈和其他部件。但是，大多数内核都包含以下 3 个公共部件：

（1）调度器。是 EOS 的心脏，提供一组算法决定何时执行哪个任务。

（2）内核对象。是特殊的内核构件，帮助创建嵌入式应用。

（3）内核服务。是内核在对象上执行的操作或通用操作。

3.3.3 多任务调度机制

首先，我们介绍几个基本概念：

（1）任务。任务是独立执行的线程，线程中包含独立的可调度的指令序列。实时应用程序的设计过程包括如何把问题分割成多个任务，每个任务是整个应用的一个组成部分，每个任务被赋予一定的优先级，有自己的一套寄存器和栈空间。在大多数典型的抢占式调度内核中，在任何时候，无论是系统任务还是应用任务，其状态都会处于就绪、

运行、阻塞三个状态之一。另外，某些商业内核还定义了挂起、延迟等颗粒更细的状态。

（2）任务对象。任务是由不同的参数集合和支持的数据结构定义。在创建任务时，每个任务都拥有一个相关的名字，一个唯一的标识号 ID，一个优先级、一个任务控制块、一个堆栈和一个任务的执行例程，这些部件一起组成一个任务对象。

（3）多任务。多任务是操作系统在预定的死线内处理多个活动的能力。多任务的运行使 CPU 的利用率得到最大地发挥，并使应用程序模块化。随着调度的任务数量的增加，对 CPU 的性能需求也随之增加，主要是由于线程运行的上下文切换增加的缘故。

（4）调度器。调度器是每个内核的心脏，调度器提供决定何时哪个任务运行的算法。多数实时内核是基于优先级调度的。

（5）可调度实体。可调度实体是一个可以根据预定义的调度算法，竞争到系统执行时间的内核对象。

（6）上下文切换。每个任务都有自己的上下文，它是每次被调度运行时所要求的寄存器状态，当多任务内核决定运行另外的任务时，它保存正在运行的任务的上下文，恢复将要运行下一任务的上下文，并运行下一任务，这个过程称为上下文切换。在任务运行时，其上下文是高度动态的。调度器从一个任务切换到另一个任务所需要的时间称为上下文切换开销。

（7）可重入性。指一段代码被一个以上的任务调用，而不必担心数据遭到破坏。具有可重入性的函数任何时候都可以被中断，一段时间以后继续运行，相应数据不会遭到破坏。

（8）分发器。分发器是调度器的一部分，执行上下文切换并改变执行的流程。分发器完成上下文切换的实际工作并传递控制。任何时候，执行的流程通过三个区域之一：应用任务、ISR（Interrupt Service Routines，中断服务程序）或内核。

根据如何进入内核的情况，分发的情况也有所不同。当一个任务是用系统调用时，分发器通常在每个任务的系统调用完成后退出内核。在这种情况下，分发器通常是以调用为基础的，因此，它可以协调由此引起的任何系统调用的任务状态转移。另一方面，如果一个 ISR 做系统调用，则分发器将被越过，直到 ISR 全部完成它的执行。

当前，大多数内核支持两种普遍的调度算法，即基于优先级的抢占调度算法和时间轮转调度算法。

1．基于优先级的抢占调度

基于优先级的抢占调度又可以分为静态优先级和动态优先级。静态优先级是指应用程序在执行的过程中各任务的优先级固定不变。在静态优先级系统中，各任务以及它们的时间约束在程序编译时是已知的；动态优先级是指应用程序在执行的过程中各任务的优先级可以动态改变。这种类型的调度，在任何时候运行的任务是所有就绪任务中具有最高优先级的任务，任务在创建时被赋予了优先级，任务的优先级可以由内核的系统调用动态更改，这使得嵌入式应用对于外部事件的响应更加灵活，从而建立真正的实时响

应系统。

一般情况下，可以采用单调执行速率调度法（Rate Monotonic Scheduling，RMS）来给任务分配优先级，基本原则是执行最频繁的任务优先级最高。RMS 做了如下假设：

（1）所有的任务都是周期性的。

（2）任务间不需要同步，没有共享资源，没有任务间的数据交换等问题。

（3）系统采用抢占式调度，总是优先级最高且就绪的任务被执行。

（4）任务的死线是其下一周期的开始。

（5）每个任务具有不随时间变化的定长时间。

（6）所有的任务具有同等重要的关键性级别。

（7）非周期性任务不具有硬死线。

要使一个具有 n 个任务的实时系统中的所有任务都满足硬实时条件，必须使下述定理成立。

RMS 定理：

$$\sum_i \frac{E_i}{T_i} \leqslant n\left(2^{1/n}-1\right)$$

式中，E_i 是任务 i 最长执行时间，T_i 是任务 i 的执行周期，E_i/T_i 是任务 i 所需的 CPU 时间。

希赛教育专家提示：基于 RMS 定理，要所有的任务满足硬实时条件，则所有有时间要求的任务总的 CPU 利用时间（或利用率）应当小于 70%。通常，作为实时系统设计的一条原则，CPU 利用率应当在 60%～70%之间。

2．时间轮转调度

时间轮转调度算法为每个任务提供确定份额的 CPU 执行时间。显然，纯粹的时间轮转调度是不能满足实时系统的要求的。取而代之的是，基于优先级抢占式扩充时间轮转调度，对于优先级相同的任务使用时间片获得相等的 CPU 执行时间。内核在满足以下条件时，把 CPU 控制权转交给下一个就绪态的任务：

（1）当前任务已无事可做。

（2）当前任务的时间片还没用完，任务就已经结束了。

如图 3-3 所示，任务 Task1、Task2、Task3 具有相同的优先级，它们按照时间片运行，任务 Task2 被更高优先级的任务 Task4 抢占，当 Task4 执行完毕后恢复 Task2 的执行。

3．任务操作

内核提供任务管理服务，也提供一个允许开发者操作任务的系统调用，典型的任务操作有任务创建和删除、任务调度控制、任务信息获取。

3.3.4 内核对象

RTOS 的用户可以使用内核对象来解决实时系统设计中的问题，如并发、同步与互

斥、数据通信等。内核对象包括信号量、消息队列、管道、事件与信号等。

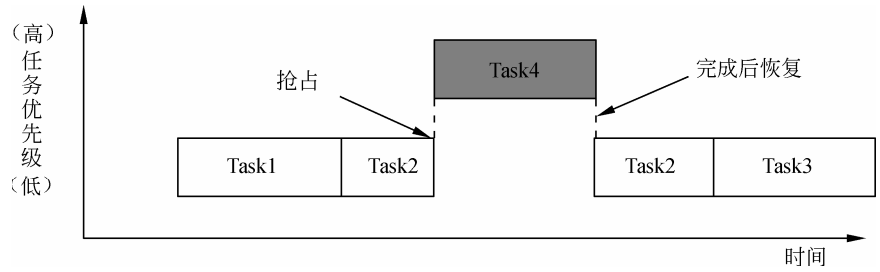

图 3-3　时间轮转调度

1．信号量

为了同步一个应用的多个并发线程和协调它们对共享资源的互斥访问，内核提供了一个信号量对象和相关的信号量管理服务。信号量是一个内核对象，就像一把锁，任务获取了该信号量就可以执行期望的操作或访问相关资源，从而达到同步或互斥的目的。

信号量可以分为如下 3 类：

（1）二值信号量。二值信号量只能有两个值：0 或 1，当其值为 0 时，认为信号量不可使用。当其值为 1 时，认为信号量是可使用的。当二值信号量被创建时，既可以初始化为可使用的，也可以初始化为不可使用的。二值信号量通常作为全局资源，被需要信号量的所有任务共享。

（2）计数信号量。计数信号量使用一个计数器赋予一个数值，表示信号量令牌的个数，允许多次获取和释放。初始化时，如果计数值为 0，表示信号量不可用；计数值大于 0，表示信号量可用。每获取一次信号量其计数值就减 1，每释放一次信号量其计数值就加 1。在有些系统中，计数信号量允许实现的计数是有界的，有些则无界。同二值信号量一样，计数信号量也可用做全局资源。

（3）互斥信号量。互斥信号量是一个特殊的二值信号量，它支持所有权、递归访问、任务删除安全和优先级反转，以避免互斥固有的问题。互斥信号量初始为开锁状态，被任务获取后转到闭锁状态，当任务释放该信号量时又返回开锁状态。

通常，内核支持以下几种操作：创建和删除信号量操作、获取和释放信号量操作、清除信号量的等待队列操作以及获取信号量信息操作。

2．消息队列

多数情况下，任务活动同步并不足以满足实时响应的要求，任务之间还必须能够交换信息。为了实现任务之间的数据交换，内核提供了消息队列对象和消息队列的管理服务。

消息队列是一个类似于缓冲区的对象，通过它任务和 ISR 可以发送和接收消息，实现数据通信。消息队列暂时保存来自发送者的消息，直到有接收者准备读取这些消息

为止。

大多数内核支持以下消息队列操作：创建和删除消息队列、发送和接收消息以及获取消息队列的信息等操作。

3．管道

管道是提供非结构化数据交换和实现任务同步的内核对象。每个管道有两个端口，一端用来读，另一端用来写。数据在管道中就像一个非结构的字节流，数据按照 FIFO 方式从管道中读出。一般 EOS 内核支持两类管道对象：

（1）命名管道。具有一个类似于文件名的名字，像文件或设备一样，出现在文件系统中，需要使用命名管道的任何任务或 ISR 都可以用该名字对其进行引用。

（2）无名管道。一般动态创建，且必须使用创建时返回的描述符才可引用此类型的管道。

通常，管道支持以下几种操作：创建和删除一个管道、读、写管道、管道控制、管道上的轮询。

4．事件

某些特殊的 EOS 提供一个特殊的寄存器作为每个任务控制块的一部分，称为事件寄存器。它是一个属于任务的对象，并由一组跟踪指定事件的二值事件标志组成。EOS 支持事件寄存器机制，创建一个任务时，内核同时创建一个事件寄存器作为任务控制块的一部分。经过事件寄存器，一个任务可以检查控制它执行的特殊事件是否出现。一个外部源（例如，另一个任务或中断处理程序）可以设置该事件寄存器的位，通知任务一个特殊事件的发生。任务说明它所希望接收的事件组，这组事件保存在寄存器中，同样，到达的事件也保存在接收的事件寄存器中。另外，任务还可以指示一个时限说明它愿意等待某个事件多长时间。如果时限超过，没有指定的事件达到任务，则内核唤醒该任务。

5．信号

信号是当一个事件发生时产生的软中断，它将信号接收者从其正常的执行路径移开并触发相关的异步处理。本质上，信号通知其他任务或 ISR 运行期间发生的事件，与正常中断类似，这些事件与被通知的任务是异步的。信号的编号和类型依赖于具体的嵌入式系统的实现。通常，嵌入式系统均提供信号设施，任务可以为每个希望处理的信号提供一个信号处理程序，或是使用内核提供的默认处理程序，也可以将一个信号处理程序用于多种类型的信号。信号可以有被忽略、挂起、处理或阻塞等 4 种不同的响应处理。

6．条件变量

条件变量是一个与共享资源相关的内核对象，它允许一个任务等待其他任务创建共享资源需要的条件。一个条件变量实现一个谓词，谓词是一组逻辑表达式，涉及共享资源的条件。谓词计算的结果是真或假，如果计算为真，则任务假定条件被满足，并且继续运行，反之，任务必须等待所需要的条件。当任务检查一个条件变量时，必须原子性地访问，所以，条件变量通常跟一个互斥信号量一起使用。

一个任务在计算谓词条件之前必须首先获取互斥信号量，然后计算谓词条件，如果为真，条件满足继续执行后续操作；否则，条件不满足，原子性地阻塞该任务并先释放互斥信号量。条件变量不是共享资源同步访问的机制，大多数开发者使用条件变量，让任务等待一个共享资源到达一个所需的状态。

3.3.5　内核服务

大多数嵌入式处理器架构都提供了异常和中断机制，允许处理器中断正常的执行路径。这个中断可能由应用软件触发，也可由一个错误或不可预知的外部事件来触发。而大多数 EOS 则提供异常和中断处理的“包裹”功能，使嵌入式系统开发者避免底层细节。

1．异常与中断

异常是指任何打断处理器正常执行，迫使处理器进入特权执行模式的事件。异常可以分为同步异常和异步异常。同步异常是指程序内部与指令执行相关的事件引起的异常，例如，内存偶地址校准异常、除数为零异常等；异步异常是指与程序指令不相关的外部事件产生的异常，例如，系统复位异常、数据接收中断等。

同步异常可以分为精确异常和不精确异常。精确异常是指处理器的程序计数器可以精确地指出引起异常的指令。而在流水线或指令预取的处理器上则不能精确地判断引起异常的指令或数据，这时的异常称为不精确异常。

异步异常可以分为可屏蔽的异常和不可屏蔽的异常。可以被软件阻塞或开放的异步异常称为可屏蔽的异常，否则，为不可屏蔽异常。不可屏蔽的异常总是被处理器处理，例如，硬件复位异常。许多处理器具有一个专门的不可屏蔽中断请求线（NMI），任何连接到 NMI 请求线的硬件都可以产生不可屏蔽中断。

所有的处理器按照定义的次序处理异常，虽然每一种嵌入式处理器处理异常的过程不尽相同，但一般都会按照优先级次序来处理。从应用程序的观点看，所有的异常都具有比操作系统内核对象更高的优先级，包括任务、队列和信号量等。

中断也称为外部中断，是一个由外部硬件产生的事件引起的异步异常，大多数嵌入式处理器架构中将中断归为异常的一类。实时内核最重要的指标是中断关了多长时间。所有的实时系统在进入临界代码时都要关中断，执行完临界代码之后再开中断。中断延迟时间是指关中断的最长时间与开始执行中断服务子程序的第一条指令的时间之和，中断恢复时间是微处理器返回到被中断的程序代码所需要的时间。

从应用的观点来看，异常和外部中断是外部硬件和应用程序通信的一种机制。一般来讲，异常和中断可以在如下两个方面用在设计中：内部错误和特殊条件管理、硬件并发和服务请求管理。

2．计时器

计时器是实时嵌入式系统的一个组成部分。时间轮转调度算法、存储器定时刷新、网络数据包的超时重传以及目标机监视系统的时序等都严格依赖于计时器。许多嵌入式

系统用不同形式的计时器来驱动时间敏感的活动，即硬件计时器和软件计时器。硬件计时器是从物理计时芯片派生出来的，超时后可以直接中断处理器，硬件计时器对精确的延迟操作具有可预测的性能。而软件计时器是通过软件功能调度的软件事件，能够对非精确的软件事件进行有效的调度，使用软件计时器可以减轻系统的中断负担。

这里有几个相关概念，需要考生了解：

（1）实时时钟：存在于嵌入式系统内部，用来追踪时间、日期的硬件计时设备。

（2）系统时钟：用来追踪从系统加电启动以来的事件时间或流失时间，可编程的间隔计时器驱动系统时钟，计时器每中断一次，系统时钟的值就递增一次。

（3）时钟节拍：它也称为时钟滴答，是特定的周期性中断。中断之间的间隔取决于不同的应用，一般在 10ms～200ms 之间。而且时钟节拍率越快，系统的额外开销就越大。

（4）可编程计时器：一般是集成在嵌入式系统内部的专门计时硬件，用做事件计数器、流失时间指示器、速率可控的周期事件产生器等。使用独立的硬件计时器可以有效地降低处理器的负载。

（5）软件计时器：它是应用程序安装的计数器，每次时钟中断，会递减一次，当计数器到达 0 时，应用的计时器超时，系统会调用安装的超时处理函数进行有关处理。

3．I/O 管理

从系统开发者的观点看，I/O 操作意味着与设备的通信，对设备初始化、执行设备与系统之间的数据传输以及操作完成后通知请求者。从系统的观点看，I/O 操作意味着对请求定位正确的设备，对设备定位正确的驱动程序，并保证对设备的同步访问。

I/O 设备、相关的驱动程序等共同组合成嵌入式系统的 I/O 子系统。图 3-4 是一个典型的微内核系统的层次模型图。

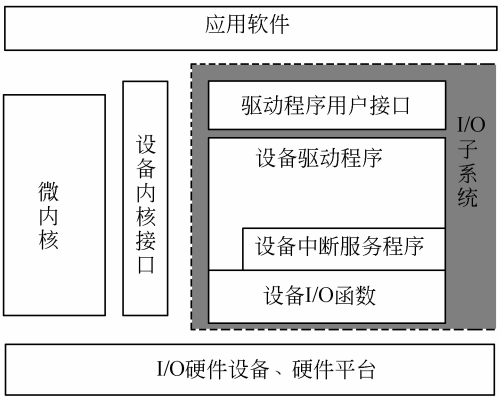

图 3-4　I/O 子系统层次模型

I/O 子系统定义一组标准的 I/O 操作函数，以便于对应用隐藏设备的特性。所有的设备驱动程序都符合并支持这个函数集，给应用提供一个能够跨越各种类型 I/O 的设备的

统一的接口。

I/O 子系统通常维护一个统一的设备驱动程序表，使用 I/O 子系统的工具函数，可以将任何驱动程序安装到此表或从表中删除。另外，还使用一个设备表来跟踪为每个设备所创建的实例。

3.3.6　常见的嵌入式操作系统

本节介绍目前最常见的 4 种嵌入式操作系统。

1．VxWorks

VxWorks 具有良好的持续发展能力、高性能的内核以及友好的用户开发环境。首先，它十分灵活，具有多达 1800 个功能强大的应用程序接口（Application Programming Interface，API）。其次，它适用面广，可以适用于从最简单到最复杂的产品设计。另外，它可靠性高，可以用于从防抱死刹车系统到星际探索的关键任务。最后，适用性强，可以用于所有流行的 CPU 平台。

2．Palm

Palm 是一种 32 位的嵌入式操作系统，提供了串行通信接口和红外线传输接口，利用它可以方便地与其他外部设备通信、传输数据；拥有开放的 OS 应用程序接口，开发商可根据需要自行开发所需的应用程序。Palm 是一套具有强开放性的系统。在编写程序时，Palm 充分考虑了掌上电脑内存相对较小的情况，因此它只占有非常小的内存。由于基于 Palm 编写的应用程序占用的空间也非常小（通常只有几十千字节），所以，基于 Palm 的掌上电脑虽然只有几兆字节的内存，但可以运行众多应用程序。

由于 Palm 产品的最大特点是使用简便、机体轻巧，因此决定了 Palm 应具有以下特点。

（1）操作系统的节能功能。由上掌上电脑要求使用电源尽可能小，因此在 Palm 的应用程序中，如果没有事件运行，则系统设备进入半休眠的状态；如果应用程序停止活动一段时间，则系统自动进入休眠状态。

（2）合理的内存管理。Palm 的存储器全部是可读写的快速 RAM，动态 RAM 类似于 PC（Personal Computer，个人计算机）上的 RAM，它为全局变量和其他不需永久保存的数据提供临时的存储空间；存储 RAM 类似于 PC 上的硬盘，可以永久保存应用程序和数据。

（3）Palm 的数据是以数据库的格式来存储的，为保证程序处理速度和存储器空间，在处理数据的时候，Palm 不是把数据从存储堆复制到动态堆后再进行处理，而是在存储堆中直接处理。

（4）Palm 与同步软件结合可以使掌上电脑与 PC 上的信息实现同步，把台式机的功能扩展到了掌上电脑。

3．Windows CE

Windows CE 也是一个开放的、可升级的 32 位嵌入式操作系统，是基于掌上电脑类的电子设备操作。Windows CE 的图形用户界面相当出色。与 PC 上的 Windows 不同的是，Windows CE 是所有源代码全部由微软自行开发的嵌入式新型操作系统，其操作界面虽来源于 PC 上的 Windows，但 Windows CE 是基于 Win32 API 重新开发的、新型的信息设备平台。Windows CE 具有模块化、结构化和基于 Win32 应用程序接口以及与处理器无关等特点。Windows CE 不仅继承了传统的 Windows 图形界面，并且在 Windows CE 平台上可以使用 PC 上的 Windows 的编程工具（如 Visual Basic、Visual C++等）、使用同样的函数、使用同样的界面风格，使绝大多数的应用软件只需简单地修改和移植就可以在 Windows CE 平台上继续使用。

Windows CE 的设计目标是：模块化及可伸缩性、实时性能好，通信能力强大，支持多种 CPU。它的设计可以满足多种设备的需要，这些设备包括了工业控制器、通信集线器以及销售终端之类的企业设备，还有像照相机、电话和家用娱乐器材之类的消费产品。一个典型的基于 Windows CE 的嵌入系统通常为某个特定用途而设计，并在不联机的情况下工作。它要求所使用的操作系统体积较小，内建有对中断的响应功能。

Windows CE 的特点有：

（1）具有灵活的电源管理功能，包括睡眠/唤醒模式。

（2）使用了对象存储技术，包括文件系统、注册表及数据库。它还具有很多高性能、高效率的操作系统特性，包括按需换页、共享存储、交叉处理同步、支持大容量堆等。

（3）拥有良好的通信能力。广泛支持各种通信硬件，亦支持直接的局域连接以及拨号连接，并提供与 PC、内部网以及 Internet（因特网）的连接，还提供与 PC 上的 Windows 的最佳集成和通信。

（4）支持嵌套中断。允许更高优先级别的中断首先得到响应，而不是等待低级别的 ISR 完成。这使得该操作系统具有嵌入式操作系统所要求的实时性。

（5）更好的线程响应能力。对高级别中断服务线程的响应时间上限的要求更加严格，在线程响应能力方面的改进，帮助开发人员掌握线程转换的具体时间，并通过增强的监控能力和对硬件的控制能力帮助他们创建新的嵌入式应用程序。

（6）256 个优先级别。可以使开发人员在控制嵌入式系统的时序安排方面有更大的灵活性。

（7）Windows CE 的 API 是 Win32 API 的一个子集，支持近 1500 个 Win32 API。有了这些 API，足可以编写任何复杂的应用程序。当然，在 Windows CE 系统中，所提供的 API 也可以随具体应用的需求而定。

4．Linux

Linux 是一个类似于 UNIX 的操作系统，Linux 系统不仅能够运行于 PC 平台，还在嵌入式系统方面大放光芒，在各种嵌入式 Linux 迅速发展的状况下，Linux 逐渐形成了

可与 Windows CE 等嵌入式操作系统进行抗衡的局面。嵌入式 Linux 的特点如下：

（1）精简的内核，性能高，稳定，多任务。

（2）适用于不同的 CPU，支持多种架构，如 x86、ARM、ALPHA、SPARC 等。

（3）能够提供完善的嵌入式图形用户界面以及嵌入式 X-Windows。

（4）提供嵌入式浏览器、邮件程序、音频和视频播放器、记事本等应用程序。

（5）提供完整的开发工具和软件开发包，同时提供 PC 上的开发版本。

（6）用户可定制，可提供图形化的定制和配置工具。

（7）常用嵌入式芯片的驱动集，支持大量的周边硬件设备，驱动丰富。

（8）针对嵌入式的存储方案，提供实时版本和完善的嵌入式解决方案。

（9）完善的中文支持，强大的技术支持，完整的文档。

（10）开放源码，丰富的软件资源，广泛的软件开发者的支持，价格低廉，结构灵活，适用面广。

3.4　嵌入式系统数据库

嵌入式 DBMS 就是在嵌入式设备上使用的 DBMS。由于用到嵌入式 DBMS 的系统多是移动信息设备，诸如掌上电脑、PDA（Personal Digital Assistant，个人数字助理）、车载设备等移动通信设备。因此，嵌入式数据库也称为移动数据库或嵌入式移动数据库，其作用主要是解决移动计算环境下数据的管理问题，移动数据库是移动计算环境中的分布式数据库。

在嵌入式系统中引入数据库技术，主要因为直接在 EOS 或裸机之上开发信息管理应用程序存在如下缺点：

（1）所有的应用都要重复进行数据的管理工作，增加了开发难度和代价。

（2）各应用之间的数据共享性差。

（3）应用软件的独立性、可移植性差，可重用度低。

在嵌入式系统中引入 DBMS 可以在很大程度上解决上述问题，提高应用系统的开发效率和可移植性。

3.4.1　使用环境的特点

嵌入式数据库系统是一个包含嵌入式 DBMS 在内的跨越移动通信设备、工作站或台式机以及数据服务器的综合系统，系统所具有的这个特点以及该系统的使用环境对嵌入式 DBMS 有着较大的影响，直接影响到嵌入式 DBMS 的结构。其使用环境的特点可以简单地归纳如下：

（1）设备随时移动性。嵌入式数据库主要用在移动信息设备上，设备的位置经常随使用者一起移动。

（2）网络频繁断接。移动设备或移动终端在使用的过程中，位置经常发生变化，同时也受到使用方式、电源、无线通信及网络条件等因素的影响。所以，一般并不持续保持网络连接，而是经常主动或被动地间歇性断接（断开和连接）。

（3）网络条件多样化。由于移动信息设备位置的经常变化，所以移动信息设备同数据服务器在不同的时间可能通过不同的网络系统连接。这些网络在网络带宽、通信代价、网络延迟、服务质量等方面可能有所差异。

（4）通信能力不对称。由于受到移动设备的资源限制，移动设备与服务器之间的网络通信能力是非对称的。移动设备的发送能力都非常有限，使得数据服务器到移动设备的下行通信带宽和移动设备到数据服务器之间的上行带宽相差很大。

3.4.2 关键技术

一个完整的嵌入式 DBMS 由若干子系统组成，包括主 DBMS、同步服务器、嵌入式 DBMS、连接网络等几个子系统，如图 3-5 所示。

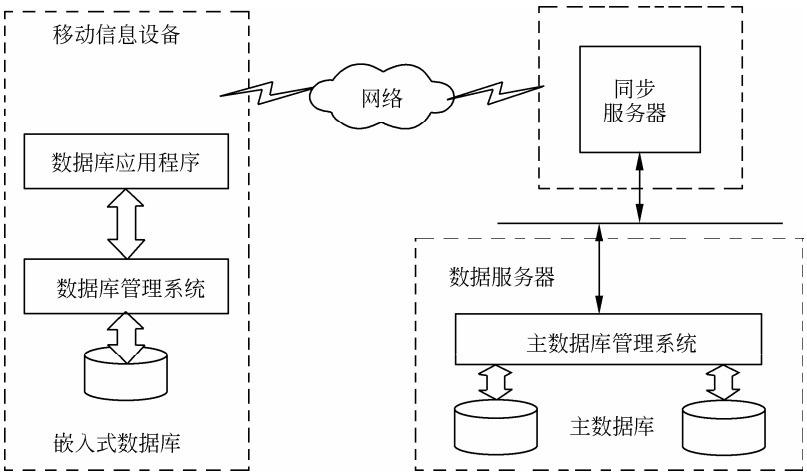

图 3-5 嵌入式数据库系统组成

（1）嵌入式 DBMS。嵌入式 DBMS 是一个功能独立的单用户 DBMS。它可以独立于同步服务器和主 DBMS 运行，对嵌入式系统中的数据进行管理，也可以通过同步服务器连接到主服务器上，对主数据库中的数据进行操作，还可以通过多种方式进行数据同步。

（2）同步服务器。同步服务器是嵌入式数据库和主数据库之间的连接枢纽，保证嵌入式数据库和主数据库中数据的一致性。

（3）数据服务器。数据服务器的主数据库及 DBMS 可以采用 Oracle 或 Sybase 等大型通用数据库系统。

（4）连接网络。主数据库服务器和同步服务器之间一般通过高带宽、低延迟的固定网络进行连接。移动设备和同步服务器之间的连接根据设备具体情况可以是无线局域网、红外连接、通用串行线或公众网等。

1．移动 DBMS 的关键技术

嵌入式移动数据库在实际应用中必须解决好数据的一致性（复制性）、高效的事务处理和数据的安全性等问题。

（1）数据的一致性。嵌入式移动数据库的一个显著特点是，移动数据终端之间以及与同步服务器之间的连接是一种弱连接，即低带宽、长延迟、不稳定和经常性断接。为了支持用户在弱环境下对数据库的操作，现在普遍采用乐观复制方法，允许用户对本地缓存上的数据副本进行操作。待网络重新连接后再与数据库服务器或其他移动数据终端交换数据修改信息，并通过冲突检测和协调来恢复数据的一致性。

（2）高效的事务处理。移动事务处理要解决在移动环境中频繁的、可预见的断接情况下的事务处理。为了保证活动事务的顺利完成，必须设计和实现新的事务管理策略和算法。

（3）数据的安全性。许多应用领域的嵌入式设备是系统中数据管理或处理的关键设备，因此嵌入式设备上的 DBS 对存取权限的控制较严格。同时，许多嵌入式设备具有较高的移动性、便携性和非固定的工作环境，也带来潜在的不安全因素。同时某些数据的个人隐私性又很高，因此在防止碰撞、磁场干扰、遗失、盗窃等方面对个人数据的安全性需要提供充分的保证。

2．移动 DBMS 的特性

移动 DBMS 的计算环境是传统分布式 DBMS 的扩展，它可以看作客户端与固定服务器结点动态连接的分布式系统。因此移动计算环境中的 DBMS 是一种动态分布式 DBMS。由于嵌入式移动 DBMS 在移动计算的环境下应用在 EOS 之上，所以它有自己的特点和功能需求：

（1）微核结构。考虑到嵌入式设备的资源有限，嵌入式移动 DBMS 应采用微型化技术实现，在满足应用的前提下紧缩其系统结构以满足嵌入式应用的需求。

（2）对标准 SQL 的支持。嵌入式移动 DBMS 应能提供了对标准 SQL 的支持。支持 SQL92 标准的子集，支持数据查询（连接查询、子查询、排序、分组等）、插入、更新、删除多种标准的 SQL 语句，充分满足嵌入式应用开发的需求。

（3）事务管理功能。嵌入式移动 DBMS 应具有事务处理功能，自动维护事务的完整性、原子性等特性；支持实体完整性和引用完整性。

（4）完善的数据同步机制。数据同步是嵌入式数据库最重要的特点。通过数据复制，可以将嵌入式数据库或主数据库的变化情况应用到对方，保证数据的一致性。

（5）支持多种连接协议。嵌入式移动 DBMS 应支持多种通信连接协议。可以通过串行通信、TCP/IP、红外传输、蓝牙等多种连接方式实现与嵌入式设备和数据库服务器的

连接。

（6）完备的嵌入式数据库的管理功能。嵌入式移动 DBMS 应具有自动恢复功能，基本无须人工干预进行嵌入式数据库管理并能够提供数据的备份和恢复，保证用户数据的安全可靠。

（7）支持多种 EOS。嵌入式移动 DBMS 应能支持 Windows CE、Palm 等多种目前流行的 EOS，这样才能使嵌入式移动 DBMS 不受移动终端的限制。

另外，一种理想的状态是用户只用一台移动终端（如手机）就能对与它相关的所有移动数据库进行数据操作和管理。这就要求前端系统具有通用性，而且要求移动数据库的接口有统一、规范的标准。前端管理系统在进行数据处理时自动生成统一的事务处理命令，提交当前所连接的数据服务器执行。这样就有效地增强了嵌入式移动 DBMS 的通用性，扩大了嵌入式移动数据库的应用前景。

希赛教育专家提示：在嵌入式移动 DBMS 中还需要考虑诸多传统计算环境下不需要考虑的问题，例如，对断接操作的支持、对跨区长事务的支持、对位置相关查询的支持、对查询优化的特殊考虑，以及对提高有限资源的利用率和对系统效率的考虑等。为了有效地解决这些问题，诸如复制与缓存技术、移动事务处理、数据广播技术、移动查询处理与查询优化、位置相关的数据处理及查询技术、移动信息发布技术、移动 Agent 等技术仍在不断地发展和完善，会进一步促进嵌入式移动 DBMS 的发展。

3.4.3 实例介绍

本节简单介绍 SQL Anywhere Studio 和 Adaptive Server Anywhere 嵌入式 DBMS。

SQL Anywhere Studio 主要用于笔记本计算机、手持设备、智能电器等领域。SQL Anywhere Studio 提供了一系列的工具，包括：

（1）Adaptive Server Anywhere 嵌入式 DBMS。这个 DBMS 可以单机运行，也可以作为数据库服务器运行。

（2）UltraLite 提交工具。通过该工具，DBS 可以分析出一个特定的应用系统需要哪些 DBMS 的功能，并根据实际需要对数据库进行精简。

（3）MobiLink 同步服务器。SQL Anywhere Studio 支持双向的信息同步。在同步环境下通过 MobiLink 来完成，在异步环境下通过 SQL Remote 来完成。

（4）PowerDesigner Physical Architect 数据库模型设计工具。

（5）PowerDynamo Web 动态页面服务器。

（6）JConnect JDBC（Java DataBase Connectivity，Java 数据库互连）驱动器。

（7）Sybase Central 图形化管理工具。用于对数据库、移动用户和数据复制提供便利。

Adaptive Server Anywhere 嵌入式 DBMS 的主要特性如下：

（1）支持多种操作系统。Adaptive Server Anywhere 可支持多种常见的操作系统，同时，Adaptive Server Anywhere 占用的资源很少，却具有强大的功能，包括参照完整性、

存储过程、触发器、行级锁、自动的时间表和自动恢复功能等。

（2）支持 Java。Adaptive Server Anywhere 全面支持 Java 技术，支持 Java 存储过程和数据类型，并且支持开发人员在数据库中创建和存储 Java 类，从而可以在服务器中实现复杂的商业应用。

（3）支持 Internet 应用。SQL Anywhere Studio 可以支持各种 Internet 标准，可以很好地利用已有的 Web 应用程序，一方面可以支持 Web 应用，另一方面可以将应用程序和数据信息在本地保存，使得通信断接时，移动用户仍可以运行应用程序。

（4）支持多种应用程序接口。Adaptive Server Any where 支持使用 ODBC、JDBC、OpenClientTM、OLEDB 和嵌入式开发数据库应用程序。通过技术，开发人员可以使用这些数据库接口对移动信息设备和智能电器进行数据访问。

（5）易于管理。由于 Adaptive Server Anywhere 的自管理和自调优功能，所以系统很少需要 DBA 干预。图形化管理工具 Sybase Central 易于使用，可以方便地集中管理远程数据库和同步环境。SQL Anywhere Studio 提供了企业系统和远程设备之间可伸缩的双向信息复制技术。

（6）系统规模配置灵活。通常，DBMS 和应用是分离的，即使应用只用到了 DBMS 的一部分功能，DBMS 仍然包括了所有的功能模块，致使 DBMS 过于庞大。为此，开发应用时，先以 Adaptive Server Anywhere 为数据库开发平台开发应用程序，然后对应用分析，找出必需的功能模块，生成一个精简的 DBMS，从而减少系统资源的占用。

3.5　嵌入式系统网络

嵌入式网络是用于连接各种嵌入式系统，使之可以互相传递信息、共享资源的网络系统。嵌入式系统在不同的场合采用不同的连接技术，如在家庭居室采用家庭信息网，在工业自动化领域采用现场总线，在移动信息设备等嵌入式系统则采用移动通信网，此外，还有一些专用连接技术用于连接嵌入式系统。

3.5.1　现场总线网

现场总线（FieldBus）也被称作工业自动化领域的计算机局域网，是一种将数字传感器、变换器、工业仪表及控制执行机构等现场设备与工业过程控制单元、现场操作站等互相连接而成的网络。它具有全数字化、分散、双向传输和多分支的特点，是工业控制网络向现场级发展的产物。现场总线是一种低带宽的底层控制网络，位于生产控制和网络结构的底层，因此也被称为底层网，它主要应用于生产现场，在测量控制设备之间实现双向的、串行的、多结点的数字通信。

现场总线控制系统（Field Contral System，FCS）是运用现场总线连接各控制器及仪表设备而构成的控制系统，FCS 将控制功能彻底下放到现场，降低了安装成本和维护费

用。实际上 FCS 是一种开放的、具有互操作性的、彻底分散的分布式控制系统。

嵌入式现场控制系统将专用微处理器置入传统的测量控制仪表，使其具备数字计算和数字通信能力。它采用双绞线、电力线或光纤等作为总线，把多个测量控制仪表连接成网络，并按照规范标准的通信协议，在位于现场的多个微机化测量控制设备之间以及现场仪表与远程监控计算机之间，实现数据传输与信息交换，形成了各种适用实际需要的自动控制系统。FCS 把单个分散的测量控制设备变成网络结点，以现场总线为纽带，使这些分散的设备成为可以互相沟通信息，共同完成自动控制任务的网络系统。借助于现场总线技术，传统上的单个分散控制设备变成了互相沟通、协同工作的整体。

现场总线主要有总线型与星型两种拓扑结构。FCS 通常由以下部分组成：现场总线仪表、控制器、现场总线线路、监控、组态计算机。这里的仪表、控制器、计算机都需要通过现场总线网卡、通信协议软件连接到网上。因此，现场总线网卡、通信协议软件是现场总线控制系统的基础和神经中枢。

现场总线克服了在传统的集散控制系统中通信由专用网络实现所带来的缺陷。把基于专用网络的解决方案变成了基于标准的解决方案，同时把集中与分散相结合的集散控制结构变成了全分布式的结构。把控制功能彻底放到现场，依靠现场设备本身来实现基本的控制功能。归纳起来，FCS 有如下优点：

（1）全数字化。将企业管理与生产自动化有机结合一直是工业界梦寐以求的理想，但只有在 FCS 出现以后这种理想才有可能高效、低成本地实现。

（2）全分布。在 FCS 中各现场设备有足够的自主性，它们彼此之间相互通信，完全可以把各种控制功能分散到各种设备中，而不再需要一个中央控制计算机，实现真正的分布式控制。

（3）双向传输。传统的 4~20mA 电流信号，一条线只能传递一路信号。现场总线设备则在一条线上既可以向上传递传感器信号，也可以向下传递控制信息。

（4）自诊断。现场总线仪表本身具有自诊断功能，而且这种诊断信息可以送到中央控制室，以便于维护，而这一点在只能传递一路信号的传统仪表中是做不到的。

（5）节省布线及控制室空间。传统的控制系统每个仪表都需要一条线连到中央控制室，在中央控制室装备一个大配线架。而在 FCS 系统中多台现场设备可串行连接在一条总线上，这样只需极少的线进入中央控制室，大量节省了布线费用，同时也降低了中央控制室的造价。

（6）多功能。数字、双向传输方式使得现场总线仪表可以摆脱传统仪表功能单一的制约，可以在一个仪表中集成多种功能，做成多变量变送器，甚至集检测、运算、控制于一体的变送控制器。

（7）开放性。1999 年年底现场总线协议已被 IEC 批准正式成为国际标准，从而使现场总线成为一种开放的技术。

（8）互操作性。现场总线标准保证不同厂家的产品可以互操作，这样就可以在一个

企业中由用户根据产品的性能、价格，选用不同厂商的产品集成在一起，避免了传统控制系统中必须选用同一厂家的产品限制，促进了有效的竞争，降低了控制系统的成本。

（9）智能化与自治性。现场总线设备能处理各种参数、运行状态信息及故障信息，具有很高的智能。能在部件，甚至网络故障的情况下独立工作，大大提高了整个控制系统的可靠性和容错能力。

（10）可靠性。由于 FCS 中的设备实现了智能化，因此与使用模拟信号的设备相比，从根本上提高了测量与控制的精确度，减小了传送误差。同时，由于系统结构的简化，设备与连线减少，现场仪表内部功能加强，使得信号的往返传输大为减少。进一步提高了系统的可靠性。

3.5.2　嵌入式 Internet

随着 Internet 和嵌入式技术的飞速发展，越来越多的信息电器，如 Web 可视电话、机顶盒、以及信息家电等嵌入式系统产品都要求与 Internet 连接，来共享 Internet 所提供的方便、快捷、无处不在的信息资源和服务，即嵌入式 Internet 技术。

1．嵌入式 Internet 的接入方式

（1）直接接入式 Internet。嵌入式设备上集成了 TCP/IP 协议栈及相关软件，这类设备可以作为 Internet 的一个结点，分配有 IP 地址，与 Internet 直接互联。这种接入方式的特点是：设备可以直接连接到 Internet，对 Internet 进行透明访问，不需要专门的接入设备，设备的协议标准化；需要的处理器性能和资源相对较高，需要占用 IP 资源，由于目前 IPv4 资源紧张，这种方案在 IPv6 网中可能更易实现。

（2）通过网关接入 Internet。即采用瘦设备方案，设备不直接接入 Internet，不需要复杂的 TCP/IP 协议全集，而是通过接入设备接入 Internet。这种接入方式的特点是：对接入设备的性能和资源要求较低，接入设备的协议栈开销较小，不需要分配合法的 IP 地址，可以降低系统的整体成本；设备可以实现多样化、小型化。

2．嵌入式 TCP/IP 协议栈

嵌入式 TCP/IP 协议栈完成的功能与完整的 TCP/IP 协议栈是相同的，但是由于嵌入式系统的资源限制，嵌入式协议栈的一些指标和接口等与普通的协议栈可能有所不同。

（1）嵌入式协议栈的调用接口与普通的协议栈不同。普通协议栈的套接字接口是标准的，应用软件的兼容性好，但是实现标准化接口的代码开销、处理和存储开销都是巨大的。因此，多数厂商在将标准的协议栈接口移植到嵌入式系统上的时候，都作了不同程度的修改简化，建立了高效率的专用协议栈，它们所提供的 API 与通用协议栈的 API 不一定完全一致。

（2）嵌入式协议栈的可裁剪性。嵌入式协议栈多数是模块化的，如果存储器的空间有限，可以在需要时进行动态安装，并且都省去了接口转发、全套的因特网服务工具等几个针对嵌入式系统非必需的部分。

（3）嵌入式协议栈的平台兼容性。一般协议栈与操作系统的结合紧密，大多数协议栈是在操作系统内核中实现的。协议栈的实现依赖于操作系统提供的服务，移植性较差。嵌入式协议栈的实现一般对操作系统的依赖性不大，便于移植。许多商业化的嵌入式协议栈支持多种操作系统平台。

（4）嵌入式协议栈的高效率。嵌入式协议栈的实现通常占用更少的空间，需要的数据存储器更小，代码效率高，从而降低了对处理器性能的要求。

3.6 嵌入式系统软件开发环境

嵌入式系统的软件开发方法不同于通用的开发方法，而是采用交叉式开发方法。本节主要介绍嵌入式系统软件开发的交叉编译环境的基本概念和特点，以及软件调试常用的几种方法。

3.6.1 嵌入式系统开发概述

嵌入式系统的软件开发采用交叉平台开发方法（Cross Platform Development，CPD），即软件在一个通用的平台上开发，而在另一个嵌入式目标平台上运行。这个用于开发嵌入式软件的通用平台称为宿主机系统，被开发的嵌入式系统称为目标机系统。而当软件执行环境和开发环境一致时的开发过程则称为本地开发。

图 3-6 是一个典型的 CPD 环境，通常包含 3 个高度集成的部分：

（1）运行在宿主机和目标机上的强有力的交叉开发工具和实用程序。

（2）运行在目标机上的高性能、可裁剪的 RTOS。

（3）连接宿主机和目标机的多种通信方式，例如，以太网、串口线、ICE（In-Circuit Emulator，在线仿真器）、ROM 仿真器等。

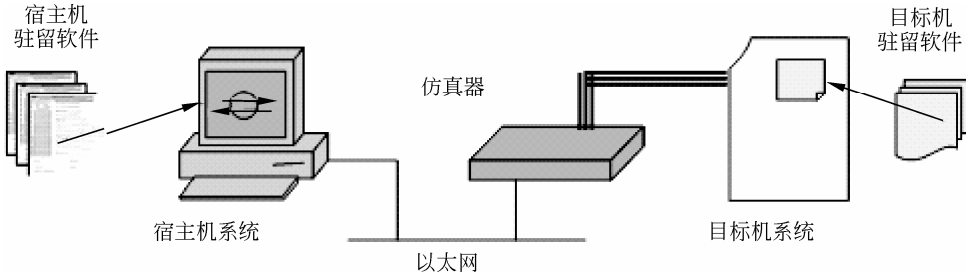

图 3-6 典型交叉平台开发环境

宿主机提供的基本开发工具有交叉编译器、交叉链接器和源代码调试器等，作为目标机的嵌入式系统则可能提供一个动态装载器、链接装载器、监视器和一个调试代理等。在目标机和宿主机之间有一组连接，通过这组连接程序代码映像从宿主机下载到目标机，

这组连接同时也用来传输宿主机和目标机调试代理之间的信息。

目前，嵌入式系统中常用的目标文件格式是 COFF（Common Object File Format，通用对象文件格式）和 ELF（Executable Linking Format，可执行链接格式）。另外，一些系统还需要有一些专门工具将上述格式转换成二进制代码格式才可使用。典型地，一个目标文件包含：

（1）关于目标文件的通用信息，如文件尺寸、启动地址、代码段和数据段等具体信息。

（2）机器架构特定的二进制指令和数据。

（3）符号表和重定位表。

（4）调试信息。

3.6.2　开发过程

嵌入式系统软件的开发过程可以分为项目计划、可行性分析、需求分析、概要设计、详细设计、程序建立、下载、调试、固化、测试及运行等几个阶段。

项目计划、可行性分析、需求分析、概要设计及详细设计等几个阶段，与通用软件的开发过程基本一致，都可按照软件工程方法进行，如采用原型化方法、结构化方法等。

希赛教育专家提示：由于嵌入式软件的运行和开发环境不同，开发工作是交叉进行的，所以每一步都要考虑到这一点。

程序建立阶段的工作是根据详细设计阶段产生的文档进行的，主要是源代码编写、编译链接等子过程，这些工作都在宿主机上进行，不需要用到目标机。产生应用程序的可执行文件后，就要用到交叉开发环境进行调试，根据实际情况可以选用 3.6.3 节中提到的调试方法或其有效组合来进行。由于嵌入式系统对安全性和可靠性的要求比通用计算机系统要高，所以，在对嵌入式系统进行白盒测试时，要求有更高的代码覆盖率。

最后，要将经调试后正确无误的可执行程序固化到目标机上。根据嵌入式系统硬件配置的不同，可以固化在 EPROM（Erasable Programmable ROM，可擦除可编程 ROM）和 Flash 等存储器中，也可固化在 DOC（DiskOnChip）等电子盘中，通常还要借助一些专用编程器进行。

3.6.3　调试方法

通用系统与嵌入式系统的软件调试过程存在着明显的差异。对于通用系统，调试工具与被调试的程序位于同一台计算机上，调试工具通过操作系统的调试接口来控制被调试的程序。但是在嵌入式系统中，由于资源的限制，不能在其上直接开发应用程序，调试过程通常也以交叉方式进行的。在实际开发实践中，经常采用的调试方法有直接调试法、调试监控法、在线仿真法、片上调试法及模拟器法等。

1．直接调试法

直接调试法是将目标代码下载到目标机上，让其执行，通过观察指示灯来判断程序的运行状态。在嵌入式系统发展的早期一般采用这种方式进行，其基本步骤是：

（1）在宿主机上编写程序。

（2）在宿主机上编译、链接生成目标机可执行程序代码。

（3）将可执行代码写入到目标机的存储器中。

（4）在目标机运行程序代码。

（5）判断程序的运行情况，如有错误则纠正错误，重复以上步骤，直到正确为止。

（6）将可执行代码固化到目标机，开发完成。

这种方法是最原始的调试方法，程序运行时产生的问题，只有通过检查源代码来解决，因而开发效率很低。

2．调试监控法

调试监控法也叫插桩法。目标机和宿主机一般通过串口、并口或以太网相连接，采用这种方法还需要在宿主机的调试器内和目标机的操作系统上分别启动一个功能模块，然后通过这两个功能模块的相互通信来实现对应用程序的调试。在目标机上添加的模块叫做桩（调试服务器、调试监控器），主要有两个作用：一是监视和控制被调试的程序；二是与宿主机上的调试程序通信，接受控制指令，返回结果等。

在进行调试的时候，宿主机上的调试器通过连接线路向调试监控器发送各种请求，实现目标机内存读/写和寄存器访问、程序下载、单步跟踪和设置断点等操作。来自宿主机的请求和目标机的响应都按照预定的通信协议进行交互。

使用插桩法作为调试手段时，开发应用程序的基本步骤如下：

（1）在宿主机上编写程序的源代码。

（2）在宿主机编译、链接生成目标机可执行程序。

（3）将目标机可执行代码下载到目标机的存储器中。

（4）使用调试器进行调试。

（5）在调试器帮助下定位错误。

（6）在宿主机上修改源代码，纠正错误，重复上述步骤直到正确为止。

（7）将可执行代码固化到目标机上。

相对于直接测试法，插桩法明显地提高了开发效率，降低了调试的难度，缩短了产品的开发周期，有效降低了开发成本。但是插桩法仍有明显的缺点，主要体现在以下几个方面：

（1）调试监控器本身的开发是个技术难题。

（2）调试监控器在目标机要占用一定的系统资源，如 CPU 时间、存储空间以及串口或网络接口等外设资源。

（3）调试时，不能响应外部中断，对有时间特性的程序不适合。

（4）在调试过程中，被调试的程序实际上在调试监控器所提供的环境中运行，这个环境可能会与实际目标程序最终的运行环境有一定的差异，这种差异有可能导致调试通过的程序最后仍不能运行。

为了克服插桩法的缺点，出现了一种改良的方法，即 ROM 仿真器法。

ROM 仿真器可以认为是一种用于替代目标机上 ROM 芯片的硬件设备，ROM 仿真器一端跟宿主机相连，另一端通过 ROM 芯片的引脚插座和目标机相连。对于嵌入式处理器来说，ROM 仿真器像是一个只读存储器，而对于宿主机来说，像一个调试监控器。ROM 仿真器的地址可以实时映射到目标机的 ROM 地址空间里，所以它可以仿真目标机的 ROM。ROM 仿真器在目标机和宿主机之间建立了一条高速信息通道，其典型的应用就是跟插桩法相结合，形成一种功能更强的调试方法。该方法具有如下优点：

（1）不必再开发调试监控器。

（2）由于是通过 ROM 仿真器上的串行口、并行口或网络接口与宿主机连接，所以不必占用目标机上的系统资源。

（3）ROM 仿真器代替了目标机上原来的 ROM，所以不必占用目标机上的存储空间来保存调试监控器。

（4）另外，即使目标机本身没有 ROM，调试依然可以进行，并且不需要使用专门工具向 ROM 写入程序和数据了。

3．在线仿真法

ICE 是一种用于替代目标机上 CPU 的设备。对目标机来说，ICE 就相当于它的 CPU，ICE 本身就是一个嵌入式系统，有自己的 CPU、内存和软件。ICE 的 CPU 可以执行目标机的所有指令，但比一般的 CPU 有更多的引脚，能够将内部信号输出到被控制的目标机上，ICE 的存储器也被映射到用户的程序空间，因此，即使没有目标机，仅用 ICE 也可以进行程序的调试。

ICE 和宿主机一般通过串口、并口或以太网相连接。在连接 ICE 和目标系统时，用 ICE 的 CPU 引出端口替代目标机的 CPU。在用 ICE 调试程序时，在宿主机运行一个调试器界面程序，该程序根据用户的操作指令控制目标机上的程序运行。

ICE 能实时地检查运行程序的处理器的状态，设置硬件断点和进行实时跟踪，所以提供了更强的调试功能。ICE 支持多种事件的触发断点，这些事件包括内存读写、I/O 读写及中断等。ICE 的一个重要特性就是实时跟踪，ICE 上有大容量的存储器用来保存每个指令周期的信息，这个功能使用户可以知道事件发生的精确时序，特别适于调试实时应用、设备驱动程序和对硬件进行功能测试。但是，ICE 的价格一般都比较昂贵。

4．片上调试法

片上调试（In Circuit Debugger，ICD）是 CPU 芯片内部的一种用于支持调试的功能模块。按照实现的技术，ICD 可以分为仿调试监控器、后台调试模式（Background Debugging Mode，BDM）、连接测试存取组（Joint Test Access Group，JTAG）和片上仿

真（On Chip Emulation，OnCE）等几类。

目前使用较多的是采用 BDM 技术的 CPU 芯片。这种芯片的外面没有跟调试相关的引脚，这些引脚在调试的时候被引出，形成一个与外部相连的调试接口，这种 CPU 具有调试模式和执行模式两种不同的运行模式。当满足了特定的触发条件时，CPU 进入调试模式，在调试模式下，CPU 不再从内存中读取指令，而是通过其调试端口读取指令，通过调试端口还可以控制 CPU 进入和退出调试模式。这样在宿主机上的调试器就可以通过调试端口直接向目标机发送要执行的指令，使调试器可以读/写目标机的内存和寄存器，控制目标程序的运行以及完成各种复杂的调试功能。

该方法的主要优点是：不占用目标机的通信端口等资源，调试环境和最终的程序运行环境基本一致，无须在目标机上增加任何功能模块即可进行；支持软、硬断点，支持跟踪功能，可以精确计量程序的执行时间；支持时序逻辑分析等功能。

该方法的主要缺点是：实时性不如 ICE 法强，使用范围受限，如果目标机不支持片上调试功能，则该方法不适用；实现技术多样，标准不完全统一，工具软件的开发和使用均不方便。

5．模拟器法

模拟器是运行于宿主机上的一个纯软件工具，它通过模拟目标机的指令系统或目标机操作系统的系统调用来达到在宿主机上运行和调试嵌入式应用程序的目的。

模拟器适合于调试非实时的应用程序，这类程序一般不与外部设备交互，实时性不强，程序的执行过程是时间封闭的，开发者可以直接在宿主机上验证程序的逻辑正确性。当确认无误后，将程序写入目标机上就可正确运行。

模拟器有两种主要类型：一类是指令级模拟器，在宿主机模拟目标机的指令系统；另一类是系统调用级模拟器，在宿主机上模拟目标操作系统的系统调用。指令级的模拟器相当于宿主机上的一台虚拟目标机，该目标机的处理器种类可以与宿主机不同。比较高级的指令级模拟器还可以模拟目标机的外部设备，如键盘、串口、网络接口等。系统调用级的模拟器相当于在宿主机上安装了目标机的操作系统，使得基于目标机的操作系统的应用程序可以在宿主机上运行。被模拟的目标机操作系统的类型可以跟宿主机的不同。两种类型的模拟器相比较，指令级模拟器所提供的运行环境与实际目标机更为接近。

使用模拟器的最大好处是在实际的目标机不存在的条件下就可以为其开发应用程序，并且在调试时利用宿主机的资源提供更详细的错误诊断信息，但模拟器有许多不足之处：

（1）模拟器环境和实际运行环境差别很大，无法保证在模拟条件下通过的应用程序也能在真实环境中正确运行。

（2）模拟器不能模拟所有的外部设备，但嵌入式系统通常包含诸多外设，但模拟器只能模拟少数部分。

（3）模拟器的实时性差，对于实时类应用程序的调试结果可能不可靠。

（4）运行模拟器需要宿主机配置较高。

尽管模拟器有很多的不足之处，但在项目开发的早期阶段，其价值是不可估量的，尤其对那些实时性不强的应用，模拟器调试不需要特殊的硬件资源，是一种非常经济的方法。

希赛教育专家提示：从软件工程的角度来看，调试与测试是不同的。而本节中的“调试”相当于测试和调试的统一体，而且更加侧重于测试。有关软件测试和软件调试的更加详细的知识，请阅读 8.8 节。

3.7　例题分析

在系统架构设计师考试中，有关嵌入式系统的试题可能出现在上午的考试（信息系统综合知识）中，也可能出现在下午的考试（案例分析和论文试题）中。为了帮助考生理解有关基础概念，本节分析 5 道典型的上午考试试题。

例题 1

在嵌入式系统设计时，下面几种存储结构中对程序员是透明的是_____。

A．高速缓存　　B．磁盘存储器

C．内存　　D．Flash 存储器

例题 1 分析

本题主要考查嵌入式系统程序设计中对存储结构的操作。

4 个选项中，高速缓存就是 Cache，它处于内存与 CPU 之间，是为了提高访问内存时的速度而设置的，这个设备对于程序员的程序编写是完全透明的。

磁盘存储器与 Flash 存储器都属于外设，在存储文件时，需要考虑到该设备的情况，因为需要将文件内容存于相应的设备之上。

内存是程序员写程序时需要考虑的，因为内存的分配与释放，是经常要用到的操作。

例题 1 答案

A

例题 2

内存按字节编址，利用 8K×4bit 的存储器芯片构成 84000H～8FFFFH 的内存，共需_____片。

A．6　　B．8　　C．12　　D．24

例题 2 分析

本题的题型在软考中较为常见，其难度在于计算时需要注意技巧，如果不注意技巧，将浪费大量时间于无谓的计算过程。8FFFFH–84000H+1=（8FFFFH+1）–84000H=90000H–84000H=C000H，化为十进制为 48K。由于内存是按字节编址，所以存储容量为：48K×8bit，48K×8bit/(8K×4bit)=12。

例题 2 答案

C

例题 3

挂接在总线上的多个部件，_____。

A．只能分时向总线发送数据，并只能分时从总线接收数据

B．只能分时向总线发送数据，但可同时从总线接收数据

C．可同时向总线发送数据，并同时从总线接收数据

D．可同时向总线发送数据，但只能分时从总线接收数据

例题 3 分析

本题考查考生对总线概念的理解。总线是一个大家都能使用的数据传输通道，大家都可以使用这个通道，但发送数据时，是采用的分时机制，而接收数据时可以同时接收，也就是说，同一个数据，可以并行的被多个客户收取。如果该数据不是传给自己的，数据包将被丢弃。

例题 3 答案

B

例题 4

以下关于嵌入式系统开发的叙述，正确的是_____。

A．宿主机与目标机之间只需要建立逻辑连接

B．宿主机与目标机之间只能采用串口通信方式

C．在宿主机上必须采用交叉编译器来生成目标机的可执行代码

D．调试器与被调试程序必须安装在同一台机器上

例题 4 分析

在嵌入式系统开发过程中，有 3 种不同的开发模式，这 3 种开发模式就会涉及本题所述的宿主机与目标机（调试程序运行的机器称为宿主机，被调试程序运行的机器称为目标机）。下面将详细说明这 3 种开发模式。

本机开发：本机开发也就是在目标机（在嵌入式系统中通常把嵌入式系统或设备简称为目标机）中直接进行操作系统移植及应用程序的开发。在这种方式下进行开发，首先就得在目标机中安装操作系统，并且具有良好的人机开发界面。

交叉开发：意思就是在一台宿主机（在嵌入式系统中通常把通用 PC 称为宿主机）上进行操作系统的裁剪，以及编写应用程序，在宿主机上应用交叉编译环境编译内核及应用程序，然后把目标代码下载到目标机上运行。这就需要在宿主机上安装、配置交叉编译环境（交叉开发工具链），使其能够编译成在目标机上运行的目标代码。

模拟开发：建立在交叉开发环境基础之上。除了宿主机和目标机以外，还得提供一个在宿主机上模拟目标机的环境，使得开发好的内核和程序直接在这个环境下运行以验证其正确性，这就不需要每次的修改都下载到目标机中，待程序正确后再下载到目标机

上运行。这样就可以达到在没有目标机的情况下调试软件的目的。比较著名的模拟开发环境有 SkyEye，它能够模拟如 ARM 等处理器的开发环境。模拟硬件环境是一件比较复杂的工程，所以多数商业嵌入式系统的开发采用的是交叉开发模式。

从以上解释可以看出，宿主机与目标机可能是一台机器上，也可能在不同机器上。宿主机与目标机之间既要有逻辑连接，还要有物理连接。至于通信方式，串口只是其中一种标准，还可采用其他方式。

例题 4 答案

C

例题 5

_____不是反映嵌入式实时操作系统实时性的评价指标。

A．任务执行时间　　B．中断响应和延迟时间

C．任务切换时间　　D．信号量混洗时间

例题 5 分析

影响嵌入式操作系统实时性的 6 个主要因素。

（1）常用系统调用平均运行时间：即系统调用效率，是指内核执行常用的系统调用所需的平均时间。

（2）任务切换时间：任务切换时间是指事件引发切换后，从当前任务停止运行、保存运行状态（CPU 寄存器内容），到装入下一个将要运行的任务状态、开始运行的时间间隔。

（3）线程切换时间：线程是可被调度的最小单位。在嵌入式系统的应用系统中，很多功能是以线程的方式执行的，所以线程切换时间同样是考察的一个要点。测试方法及原理与任务切换类似，此处不再介绍。

（4）任务抢占时间：任务抢占时间是高优先级的任务从正在运行的低优先级任务中获得系统控制权所消耗的时间。

（5）信号量混洗时间：信号量混洗时间指从一个任务释放信号量到另一个等待该信号量的任务被激活的时间延迟。在嵌入式系统中，通常有许多任务同时竞争某一共享资源，基于信号量的互斥访问保证了任一时刻只有一个任务能够访问公共资源。信号量混洗时间反映了与互斥有关的时间开销，是 RTOS 实时性的一个重要指标。

（6）中断响应时间：中断响应时间是指从中断发生到开始执行用户的中断服务程序代码来处理该中断的时间。中断处理时间通常不仅由 RTOS 决定，而且还由用户的中断处理程序决定，所以不应包括在测试框架之内。

例题 5 答案

A