第 1 章

绪　　论

本章将对嵌入式系统进行概述，介绍嵌入式系统的组成、实时操作系统、嵌入式系统的软件、嵌入式系统的分类、嵌入式系统的应用领域、嵌入式系统的体系和嵌入式系统的设计方法。

1.1　嵌入式系统

微课视频

随着计算机技术的不断发展，计算机的处理速度越来越快，存储容量越来越大，外围设备的性能越来越好，满足了高速数值计算和海量数据处理的需要，形成了高性能的通用计算机系统。

以往按照计算机的体系结构、运算速度、结构规模、适用领域，计算机可分为大型机、中型机、小型机和微型机，并以此组织学科和产业分工，这种分类沿袭了约 40 年。近 20 年来，随着计算机技术的迅速发展，以及计算机技术和产品对其他行业的广泛渗透，以应用为中心的分类方法变得更加切合实际。

电气与电子工程师学会（Institute of Electrical and Electronics Engineers，IEEE）定义的嵌入式系统（Embedded Systems）是“用于控制、监视或辅助操作机器和设备运行的装置”[①]。这主要是从应用上加以定义的，从中可以看出嵌入式系统是软件和硬件的综合体，还可以涵盖机械等附属装置。

国内普遍认同的嵌入式系统的定义是：以计算机技术为基础，以应用为中心，软件、硬件可剪裁，适合应用系统对功能可靠性、成本、体积、功耗严格要求的专业计算机系统。在构成上，嵌入式系统以微控制器及软件为核心部件，两者缺一不可；在特征上，嵌入式系统具有方便、灵活地嵌入其他应用系统的特征，即具有很强的可嵌入性。

按嵌入式微控制器类型划分，嵌入式系统可分为以单片机为核心的嵌入式单片机系统、以工业计算机板为核心的嵌入式计算机系统、以数字信号处理器（Digital Signal Processor，

① 原文为 devices used to control，monitor，or assist the operation of equipment，machinery or plants。

DSP）为核心的嵌入式数字信号处理器系统、以现场可编程门阵列（Field Programmable Gate Array，FPGA）为核心的嵌入式可编程片上系统（System on a Programmable Chip，SOPC）等。

嵌入式系统在含义上与传统的单片机系统和计算机系统有很多重叠部分。为了方便区分，在实际应用中，嵌入式系统还应该具备以下 3 个特征。

（1）嵌入式系统的微控制器通常是由 32 位及以上的精简指令集计算机（Reduced Instruction Set Computer，RISC）处理器组成的。

（2）嵌入式系统的软件系统通常是以嵌入式操作系统为核心，外加用户应用程序。

（3）嵌入式系统在特征上具有明显的可嵌入性。

嵌入式系统应用经历了无操作系统、单操作系统、实时操作系统和面向 Internet 4 个阶段。21 世纪无疑是一个网络的时代，互联网的快速发展及广泛应用为嵌入式系统的发展及应用提供了良好的机遇。“人工智能”这一技术一夜之间人尽皆知，而嵌入式在其发展过程中扮演着重要角色。

嵌入式系统的广泛应用和互联网的发展导致了物联网概念的诞生，设备与设备之间、设备与人之间以及人与人之间要求实时互联，导致了大量数据的产生，大数据一度成为科技前沿，每天世界各地产生的数据量呈指数增长，数据远程分析成为必然要求，云计算被提上日程。数据存储、传输、分析等技术的发展无形中催生了人工智能，因此人工智能看似突然出现在大众视野，实则经历了近半个世纪的漫长发展，其制约因素之一就是大数据。而嵌入式系统正是获取数据的最关键的系统之一。人工智能的发展可以说是嵌入式系统发展的产物，同时人工智能的发展要求更多、更精准的数据，以及更快、更方便的数据传输。这促进了嵌入式系统的发展，两者相辅相成，嵌入式系统必将进入一个更加快速的发展时期。

1.1.1 嵌入式系统概述

嵌入式系统的发展大致经历了以下 3 个阶段。

（1）以嵌入式微控制器为基础的初级嵌入式系统。

（2）以嵌入式操作系统为基础的中级嵌入式系统。

（3）以 Internet 和实时操作系统（Real Time Operating System，RTOS）为基础的高级嵌入式系统。

嵌入式技术与 Internet 技术的结合正在推动着嵌入式系统的飞速发展，嵌入式系统市场展现出了美好的前景，也对嵌入式系统的生产厂商提出了新的挑战。

通用计算机具有计算机的标准形式，通过装配不同的应用软件，应用在社会的各个方面。现在，在办公室、家庭中广泛使用的个人计算机（Personal Computer，PC）就是通用计算机最典型的代表。

而嵌入式计算机则是以嵌入式系统的形式隐藏在各种装置、产品和系统中。在许多应用领域，如工业控制、智能仪器仪表、家用电器、电子通信设备等，对嵌入式计算机的应用有

着不同的要求，主要如下。

（1）能面对控制对象，如面对物理量传感器的信号输入、面对人机交互的操作控制、面对对象的伺服驱动和控制。

（2）可嵌入应用系统。由于体积小、低功耗、价格低廉，可方便地嵌入应用系统和电子产品中。

（3）能在工业现场环境中长时间可靠运行。

（4）控制功能优良。对外部的各种模拟和数字信号能及时捕捉，对多种不同的控制对象能灵活地进行实时控制。

可以看出，满足上述要求的计算机系统与通用计算机系统是不同的。换句话讲，能够满足和适合以上这些应用的计算机系统与通用计算机系统在应用目标上有巨大的差异。一般将具备高速计算能力和海量存储，用于高速数值计算和海量数据处理的计算机称为通用计算机系统。而将面对工控领域对象，嵌入各种控制应用系统、各类电子系统和电子产品中，实现嵌入式应用的计算机系统称为嵌入式计算机系统，简称为嵌入式系统。

嵌入式系统将应用程序和操作系统与计算机硬件集成在一起，简单地讲，就是系统的应用软件与系统的硬件一体化。这种系统具有软件代码小、高度自动化、响应速度快等特点，特别适合面向对象的要求实时和多任务的应用。

特定的环境和特定的功能要求嵌入式系统与所嵌入的应用环境成为一个统一的整体，并且往往要满足紧凑、可靠性高、实时性好、功耗低等技术要求。面向具体应用的嵌入式系统，以及系统的设计方法和开发技术，构成了今天嵌入式系统的重要内涵，也是嵌入式系统发展为一个相对独立的计算机研究和学习领域的原因。

1.1.2　嵌入式系统和通用计算机系统比较

作为计算机系统的不同分支，嵌入式系统和人们熟悉的通用计算机系统既有共性，也有差异。

1. 嵌入式系统和通用计算机系统的共同点

嵌入式系统和通用计算机系统都属于计算机系统，从系统组成上讲，它们都是由硬件和软件构成的；它们的工作原理是相同的，都是存储程序机制。从硬件上看，嵌入式系统和通用计算机系统都是由中央处理器（Central Processing Unit，CPU）、存储器、输入 / 输出接口和中断系统等部件组成；从软件上看，嵌入式系统软件和通用计算机软件都可以划分为系统软件和应用软件两类。

2. 嵌入式系统和通用计算机系统的不同点

作为计算机系统的一个新兴的分支，嵌入式系统与人们熟悉和常用的通用计算机系统相比，又具有以下不同点。

（1）形态。通用计算机系统具有基本相同的外形（如主机、显示器、鼠标和键盘等）并且独立存在；而嵌入式系统通常隐藏在具体某个产品或设备（称为宿主对象，如空调、洗衣

机、数字机顶盒等）中，它的形态随着产品或设备的不同而不同。

（2）功能。通用计算机系统一般具有通用而复杂的功能，任意一台通用计算机都具有文档编辑、影音播放、娱乐游戏、网上购物和通信聊天等通用功能；而嵌入式系统嵌入在某个宿主对象中，功能由宿主对象决定，具有专用性，通常是为某个应用量身定做的。

（3）功耗。目前，通用计算机系统的功耗一般为 200W 左右；而嵌入式系统的宿主对象通常是小型应用系统，如手机、智能手环等，这些设备不可能配置容量较大的电源，因此低功耗一直是嵌入式系统追求的目标，如日常生活中使用的智能手机，其待机功率为 100 ～ 200mW，即使在通话时功率也只有 4 ～ 5W。

（4）资源。通用计算机系统通常拥有大而全的资源（如鼠标、键盘、硬盘、内存条和显示器等）；而嵌入式系统受限于嵌入的宿主对象（如手机、智能手环等），通常要求小型化和低功耗，其软硬件资源受到严格的限制。

（5）价值。通用计算机系统的价值体现在“计算”和“存储”上，计算能力（处理器的字长和主频等）和存储能力（内存和硬盘的大小和读取速度等）是通用计算机系统的通用评价指标；而嵌入式系统往往嵌入某个设备和产品中，其价值一般不取决于其内嵌的处理器的性能，而体现在它所嵌入和控制的设备。例如，一台智能洗衣机往往用洗净比、洗涤容量和脱水转速等指标衡量，而不以其内嵌的微控制器的运算速度和存储容量等来衡量。

1.1.3 嵌入式系统的特点

通过嵌入式系统的定义和嵌入式系统与通用计算机系统的比较，可以看出嵌入式系统具有以下特点。

1. 专用性强

嵌入式系统通常是针对某种特定的应用场景，与具体应用密切相关，其硬件和软件都是面向特定产品或任务而设计的。不但一种产品中的嵌入式系统不能应用到另一种产品中，甚至都不能嵌入同一种产品的不同系列。例如，洗衣机的控制系统不能应用到洗碗机中，甚至不同型号洗衣机中的控制系统也不能相互替换，因此嵌入式系统具有很强的专用性。

2. 可裁剪性

受限于体积、功耗和成本等因素，嵌入式系统的硬件和软件必须高效率地设计，根据实际应用需求“量体裁衣”，去除冗余，从而使系统在满足应用要求的前提下达到最精简的配置。

3. 实时性好

许多嵌入式系统应用于宿主系统的数据采集、传输与控制过程时，普遍要求嵌入式系统具有较好的实时性。例如，现代汽车中的制动器、安全气囊控制系统，武器装备中的控制系统，某些工业装置中的控制系统等，这些应用对实时性有着极高的要求，一旦达不到应有的实时性，就有可能造成极其严重的后果。另外，虽然有些系统本身的运行对实时性要求不是很高，但实时性也会对用户体验感产生影响，如需要避免人机交互的卡顿、遥控反应迟钝等情况。

4. 可靠性高

嵌入式系统的应用场景多种多样，面对复杂的应用环境，嵌入式系统应能够长时间稳定可靠地运行。在某些应用中，嵌入式系统硬件或软件中存在的一个小 Bug，都有可能导致灾难性后果的发生。例如，波音 737 MAX 客机在 2018—2019 年相继发生的两起重大空难，都是因为迎角传感器（Angle of Attack，AOA）的数据错误，触发了防失速控制系统自动操作，机头不断下压。飞行员多次手动拉伸未果，最终导致飞机坠毁的灾难性事故。由此可见，高可靠性要求是特殊应用中嵌入式系统的显著特征。

5. 体积小、功耗低

由于嵌入式系统要嵌入具体的应用对象体中，其体积大小受限于宿主对象，因此往往对其体积有着严格的要求。例如，心脏起搏器就只有一粒胶囊的大小。2020 年 8 月，埃隆·马斯克发布的拥有 1024 个信道的 Neuralink 脑机接口只有一枚硬币大小。同时，由于嵌入式系统在移动设备、可穿戴设备以及无人机、人造卫星等应用设备中不可能配置交流电源或大容量的电池，因此低功耗也往往是嵌入式系统所追求的一个重要指标。

6. 注重制造成本

与其他商品一样，制造成本会对嵌入式系统设备或产品在市场上的竞争力有很大的影响。同时，嵌入式系统产品通常会进行大量生产。例如，现在的消费类嵌入式系统产品，通常的年产量会在百万、千万甚至亿数量级。节约单个产品的制造成本，意味着总制造成本的海量节约，会产生可观的经济效益。因此，注重嵌入式系统的硬件和软件的高效设计，在满足应用需求的前提下有效地降低单个产品的制造成本，也成为嵌入式系统所追求的重要目标之一。

7. 生命周期长

随着计算机技术的飞速发展，像桌面计算机、笔记本电脑以及智能手机这样的通用计算机系统的更新换代速度大大加快，更新周期通常为 18 个月左右。然而，嵌入式系统和实际具体应用装置或系统紧密结合，一般会伴随具体嵌入的产品维持 8 ～ 10 年的相对较长的使用时间，其升级换代往往是和宿主对象系统同步进行的。因此，相较于通用计算机系统，嵌入式系统产品一旦进入市场后，不会像通用计算机系统那样频繁换代，通常具有较长的生命周期。

8. 不可垄断性

代表传统计算机行业的 Wintel（Windows-Intel）联盟统治桌面计算机市场长达 30 多年，形成了事实上的市场垄断。而嵌入式系统是将先进的计算机技术、半导体电子技术和网络通信技术与各个行业的具体应用相结合后的产物，其拥有更为广阔和多样化的应用市场，行业细分市场极其宽泛，这一点就决定了嵌入式系统必然是一个技术密集、资金密集、高度分散、不断创新的知识集成系统。特别是 5G 技术、物联网技术、人工智能技术与嵌入式系统的快速融合，催生了嵌入式系统创新产品的不断涌现，没有一家企业能够形成对嵌入式系统市场的垄断，给嵌入式系统产品的设计研发提供了广阔的市场空间。

1.2 嵌入式系统的组成

嵌入式系统是一个在功能、可靠性、成本、体积和功耗等方面有严格要求的专用计算机系统，因此具有一般计算机组成结构的共性。从总体上看，嵌入式系统的核心部分由嵌入式硬件和嵌入式软件组成；从层次结构上看，嵌入式系统可划分为硬件层、驱动层、操作系统层以及应用层 4 个层次，如图 1-1 所示。

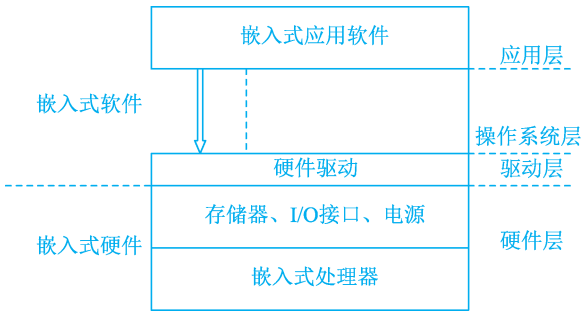

图 1-1 嵌入式系统的组成结构


微课视频

嵌入式硬件（硬件层）是嵌入式系统的物理基础，主要包括嵌入式处理器、存储器、输入 / 输出（I/O）接口和电源等。其中，嵌入式处理器是嵌入式系统的硬件核心，通常可分为嵌入式微处理器、嵌入式微控制器、嵌入式数字信号处理器以及嵌入式片上系统等主要类型。

存储器是嵌入式系统硬件的基本组成部分，包括随机存取存储器（Random Access Memory，RAM）、Flash、电擦除可编程只读存储器（Electrically-Erasable Programmable Read-Only Memory，EEPROM）等主要类型，承担着存储嵌入式系统程序和数据的任务。目前的嵌入式处理器中已经集成了较为丰富的存储器资源，同时也可通过 I/O 接口在嵌入式处理器外部扩展存储器。

I/O 接口及设备是嵌入式系统对外联系的纽带，负责与外部世界进行信息交换。I/O 接口主要包括数字接口和模拟接口两大类，其中数字接口又可分为并行接口和串行接口，模拟接口包括模数转换器（Analog to Digital Converter，ADC）和数模转换器（Digital to Analog Converter，DAC）。并行接口可以实现数据的所有位同时并行传输，传输速度快，但通信线路复杂，传输距离短。串行接口则采用数据位一位位顺序传输的方式，通信线路少，传输距离远，但传输速度相对较慢。常用的串行接口有通用同步/异步收发器（Universal Synchronous/Asnchronous Receiver/Transmitter，USART）接口、串行外设接口（Serial Peripheral Interface，SPI）、芯片间总线（P2C）接口以及控制器局域网（Controller Area Network，CAN）接口等，实际应用时可根据需要选择不同的接口类型。I/O 设备主要包括人机交互设备（按键、显示器等）和机机交互设备（传感器、执行器等），可根据实际应用需求选择所需的设备类型。

嵌入式软件运行在嵌入式硬件平台之上，指挥嵌入式硬件完成嵌入式系统的特定功能。嵌入式软件可包括硬件驱动（驱动层）、嵌入式操作系统（操作系统层）以及嵌入式应用软件（应用层）3 个层次。另外，有些系统包含中间层，中间层也称为硬件抽象层（Hardware Abstract Layer，HAL）或板级支持包（Board Support Package，BSP），对于底层硬件，它主要负责相关硬件设备的驱动；而对于上层的嵌入式操作系统或应用软件，它提供了操作和控制硬件的规则与方法。嵌入式操作系统（操作系统层）是可选的，简单的嵌入式系统无须嵌入式操作系统的支持，由应用层软件通过驱动层直接控制硬件层完成所需功能，也称为“裸金属”（Bare-Metal）运行。对于复杂的嵌入式系统，应用层软件通常需要在嵌入式操作系统内核以及文件系统、图形用户界面、通信协议栈等系统组件的支持下，完成复杂的数据管理、人机交互以及网络通信等功能。

嵌入式处理器是一种在嵌入式系统中使用的微处理器。从体系结构来看，与通用 CPU 一样，嵌入式处理器也分为冯•诺依曼（von Neumann）结构的嵌入式处理器和哈佛（Harvard）结构的嵌入式处理器。冯•诺依曼结构是一种将内部程序空间和数据空间合并在一起的结构，程序指令和数据的存储地址指向同一个存储器的不同物理位置，程序指令和数据的宽度相同，取指令和取操作数通过同一条总线分时进行。大部分通用处理器采用的是冯•诺依曼结构，也有不少嵌入式处理器采用冯•诺依曼结构，如 Intel 8086、Arm7、MIPS、PIC16 等。哈佛结构是一种将程序空间和数据空间分开在不同的存储器中的结构，每个空间的存储器独立编址，独立访问，设置了与两个空间存储器相对应的两套地址总线和数据总线，取指令和执行能够重叠进行，数据的吞吐率提高了一倍，同时指令和数据可以有不同的数据宽度。大多数嵌入式处理器采用了哈佛结构或改进的哈佛结构，如 Intel 8051、Atmel AVR、Arm9、Arm10、Arm11、Arm Cortex-M3 等系列嵌入式处理器。

从指令集的角度看，嵌入式处理器也有复杂指令集（Complex Instruction Set Computer，CISC）和精简指令集（Reduced Instruction Set Computer，RISC）两种指令集架构。早期的处理器全部采用的是 CISC 架构，它的设计动机是要用最少的机器语言指令完成所需的计算任务。为了提高程序的运行速度和软件编程的方便性，CISC 处理器不断增加可实现复杂功能的指令和多种灵活的寻址方式，使处理器所含的指令数目越来越多。然而，指令数量越多，完成微操作所需的逻辑电路就越多，芯片的结构就越复杂，器件成本也相应越高。相比之下，RISC 指令集是一套优化过的指令集架构，可以从根本上快速提高处理器的执行效率。在 RISC 处理器中，每个机器周期都在执行指令，无论简单还是复杂的操作，均由简单指令的程序块完成。由于指令高度简约，RISC 处理器的晶体管规模普遍都很小而且性能强大。因此，继 IBM 公司推出 RISC 架构和处理器产品后，众多厂商纷纷开发出自己的 RISC 指令系统，并推出自己的 RISC 架构处理器，如 DEC 公司的 Alpha、SUN 公司的 SPARC、HP 公司的 PA-RISC、MIPS 技术公司的 MIPS、Arm 公司的 Arm 等。RISC 处理器被广泛应用于消费电子产品、工业控制计算机和各类嵌入式设备中。RISC 处理器的热潮出现在 RISC-V 开源指令集架构推出后，涌现出了各种基于 RISC-V 架构的嵌入式处理器，国外的有 SiFive 公司的

U54-MC Coreplex、GreenWaves Technologies 公司的 GAP8、Western Digital 公司的 SweRV EH1，国内的有睿思芯科（深圳）技术有限公司的 Pygmy、芯来科技（武汉）有限公司的 Hammingbird（蜂鸟）E203、晶心科技（武汉）有限公司的 AndeStar V5 和 AndesCore N22，以及平头哥半导体有限公司的玄铁 910 等。

1.3 实时操作系统

作为管理计算机硬件与软件资源的程序，操作系统（Operating System，OS）在现代计算机体系结构中占有重要的地位。目前，广泛使用的操作系统有 3 类，即批处理操作系统、分时操作系统和实时操作系统。

批处理操作系统不具有用户交互性，用户将作业提交给操作系统执行后就不再干预，多用于早期的大中型计算机系统，如 IBM 公司的 DOS/VSE 系统。

分时操作系统可以让多个计算机用户共享系统资源，及时响应和服务于联机用户，多应用于网络操作系统，如 UNIX 系统。但是，对于分时操作系统，软件的执行对时间的要求并不严格。


微课视频

实时操作系统是对事件进行实时处理，虽然事件可能在无法预知的时刻到达，但是软件在事件发生时必须能够在严格的时限（系统响应时间）内作出响应，即使是在尖峰负荷下，也应如此。若不满足时限，则可能造成灾难性的后果，尤其在航天、军事、工业控制等系统响应时间要求较为严格的领域中。同时，实时操作系统的重要特点是具有系统的可确定性，即系统能对运行情况的最好和最坏运行时间作出精确估计。因此，实时操作系统在现代工业、军事、能源、自动化、通信等方面有着重要的应用。

1.3.1 实时系统的概念

在实时系统中，程序执行的正确性既包括逻辑结果的正确性，也包括运行时间的约束性。如果一个系统能够及时地响应外部或内部的事件请求，在指定或确定的时间内完成对该事件的处理，那么称该系统具有“实时性”。而一个操作系统若能够在优先确定的时间内对异步输入进行处理并输出结果，同时具有事件驱动性，则这个操作系统具备了“实时性”，可以称为实时操作系统（Real Time Operating System，RTOS）。

实时操作系统是一种既能提供有效机制和服务保证系统的实时性调度和资源管理，也能保证时间和资源的可预测性以及可计算性的操作系统。

在大部分实时操作系统中都提供了基于任务的应用程序接口（Application Programming Interface，API）函数，这些 API 使用较小且独立的应用代码片，通过抽象的时序关系相互作用以减少 API 之间的相互依赖。同时，实时操作系统还提供了任务的优先级机制，可以将非关键性任务和关键性任务分开处理。实时操作系统采用事件驱动机制，不会浪费时间片去处理未发生的事件，这也保证了实时性。

与通用操作系统相比，实时操作系统占用更少的内存，资源利用率更高，系统响应时间高度可预测。因此，实时操作系统不仅要满足应用功能性要求，还要满足实时性要求，在应用环境不可预测的条件下，始终保证系统行为的可预测性。

通用操作系统注重系统的平均表现（如系统响应时间、吞吐量以及资源利用率等），强调“整体表现”。而实时操作系统注重的是每个实时任务在最坏情况下的实时性要求，强调“最坏情况下的个体表现”。

通用操作系统与实时操作系统的区别主要体现在四方面：任务调度策略、内存管理方式、中断处理方式和系统管理方式。

在任务调度方面，通用操作系统中的任务调度策略一般采用基于优先级的抢先式调度策略，对于优先级相同的进程则采用时间片轮询调度方式。实时操作系统中的任务调度策略通常采用静态表驱动方式和固定优先级抢先式调度方式。

在内存管理方面，通用操作系统资源充足且实时性要求不高，考虑到内存的性能以及安全问题，常引入虚拟存储器管理机制。但是，这种机制增加了系统开销，因此在实时操作系统中一般不采用。

在中断处理方面，由于通用操作系统对实时性要求没有那么严格，中断处理程序始终先于其他用户进程被响应；而实时操作系统则尽可能地屏蔽中断以保证系统的响应时间。

在系统管理方面，实时操作系统对于系统调用、内存管理等开销通常会设定时间上界，而通用操作系统则无此要求；同时，通用操作系统核心态系统调用为不可重入，而实时操作系统则设计为可重入以保证系统的可预测性。

1.3.2　实时操作系统的基本特征

实时操作系统具有实时性、可靠性、可确定性和容错性等基本特征。

1. 实时性

实时操作系统保证系统能够在预定或规定的时间内完成对外部事件的响应和处理。每个实时任务具有时间约束，所有实时任务必须要保证在任何情况下，都能按照实时任务调度算法满足其时间约束。实时性是实时操作系统最基本的特征。为了保证系统的实时性，系统通常既需要依赖于硬件提供精确的时钟精度，也需要本身具有高精度计时系统，以确定实时应用中某个任务或函数执行的时间。

2. 可靠性

实时系统执行中产生的故障可能会导致人身、财产等重大损失。实时操作系统需要保证系统能够正确、实时地执行，并且将产生故障的概率降到可控范围内。同时，通过特定的机制使系统在故障产生时仍然能够有效地执行部分关键性任务，最大限度地降低不可抗拒故障因素给系统带来的不良影响。

3. 可确定性

实时操作系统的任务可确定性是指其任务必须按照预定时间或时间间隔进行调度和执

行。当多个任务竞争使用处理器和资源时，实时操作系统可以采取中断机制或调度算法保证任务的正确执行。系统实时性的保证依赖于系统能够准确地对任务的执行时间进行判断，包括硬件延迟的确定性和应用程序响应时间的确定性，保证应用程序执行时间是有界的，从而保证实时任务的执行能够满足时间约束。

4. 容错性

对于系统在执行过程中产生的错误或故障，实时操作系统能够通过有效的算法等机制预估并最小化其带来的影响，保证系统在最坏的情况下对于部分关键性任务仍然能正确执行。容错性是相对于可靠性而言的，可靠性着重于系统错误发生之前，通过优化调度算法等机制尽力消除所有潜在的故障。而容错性则强调即使发生了故障，也要保障系统能够正常工作。尽管系统在设计时通过了严格的测试和验证，但是在运行中产生的软件和硬件错误还是不可避免的，所以实时操作系统的容错性至关重要。

1.3.3 实时操作系统性能的衡量指标

对于一个实时操作系统性能评估的测量基准有 Wetstone、Dhrystone、Hartstone 以及 Rhealstone 等。Rhealstone 基准是目前工业界评判实时系统性能指标的主要基准之一。通过对实时操作系统的 6 个时间量进行计算，将其加权获得 Rhealstone 数，数值越小，证明该系统实时性越好。这 6 个时间量如下。

（1）任务切换时间（上下文切换时间，Context Switch Time）：系统在两个独立的任务之间切换所需要的时间。

（2）抢占时间（Preemption Time）：系统从正在执行的低优先级任务转移到开始执行高优先级任务所需要的时间。

（3）中断延迟时间（Interrupt Latency Time）：从接收到中断信号到操作系统作出响应，并完成进入中断服务例程所需要的时间。

（4）信号量混洗时间（Semaphore Shuffling Time）：由于系统多任务执行的互斥原理，在一个任务执行完成后释放信号量到另一个等待该信号量的任务被激活所需要的时间间隔。

（5）死锁解除时间（Deadlock Breaking Time）：系统进入死锁状态到解除死锁顺利执行所需要的时间。

（6）数据报吞吐时间（Datagram Throughput Time）：一个任务进行数据传输到另一个任务，每秒可以传输的字节数。

1.3.4 实时操作系统的分类

任务若没有在截止时间完成，根据所造成后果的严重程度将实时操作系统分为硬实时和软实时操作系统。

1. 硬实时操作系统

硬实时操作系统具有刚性的时限，要求所有任务必须在任何条件下（即使在最坏的各种

处理负载下）都能在规定的时间内完成，任何任务超过其截止时间都会造成系统整体的失败。任务超过截止时间会导致灾难性的后果，系统的收益为负无穷。例如，导弹控制系统、自动驾驶系统等的操作系统都属于硬实时操作系统。

2. 软实时操作系统

相对于硬实时操作系统，软实时操作系统具有一定的“容忍度”。软实时操作系统具有灵活的时限，任务超过其截止时间所造成的后果没有那么严重，可能仅仅降低了系统的吞吐量。任务超过截止时间不会导致严重的后果，系统的收益也可能为正。例如，IPTV 数字电视机顶盒，需要实时解码视频流，如果丢失了一个或几个视频帧，显然会造成视频的品质变差，但是只要进行简单的去抖处理，丢失几个视频帧不会对整个系统造成不可挽回的影响。

1.3.5　POSIX 标准

可移植性操作系统接口（Portable Operating System Interface of UNIX，POSIX）是由 IEEE 开发的操作系统接口标准。作为一个标准的软件接口，POSIX 的主要目的是在跨越 UNIX 各种变型操作系统之间，完善应用程序的互操作性和可移植性。理论上，只要两个操作系统都支持 POSIX，如果写的程序和编译的程序能够在一个操作系统上执行，那么该程序也可以在不改变源代码的情况下通过编译在另一个操作系统上执行。

POSIX 标准定义了操作系统为应用程序提供的标准接口，以及与操作系统各个方面相关的内容和协议。目前，该标准已经成为实时操作系统的主要标准之一。虽然 POSIX 早期是为在 UNIX 环境下应用程序的可移植性而开发，但发展起来后，也适用于包括 UNIX 在内的其他操作系统。

POSIX 标准设计的目的是保证一个支持 POSIX 兼容操作系统的程序可以应用在任何一个（即使来自其他开发厂商）POSIX 操作系统上，无须修改便可以编译执行。因此，POSIX 具有跨平台开发的特性。而与那些依靠虚拟机等这类中间层支持、损失部分性能而获取跨平台开发能力的标准不同，POSIX 不需要中间层的支持，而是在自身原有应用程序接口之上，再封装一层 POSIX 兼容层，来提供对 POSIX 的支持，因此保留了原来操作系统的性能。

下面举个简单的例子描述 POSIX 工作的基本原理。在不同平台下，内核在完成同一功能时往往使用的函数有所不同。例如，在 Linux 系统下进程创建函数是 Fork()，而在 Windows 系统下进程创建函数是 CreatProcess()，现需要将一个具有进程创建函数的 Linux 程序应用移植到 Windows 平台，根据 POSIX 标准，需要先将 Linux 系统下的 Fork() 函数封装成 POSIX_Fork() 函数，将 Windows 系统下的 CreatProcess() 函数封装成 POSIX_Fork() 函数，然后同时在头文件中进行声明，这样程序就在源代码级别可移植了。

POSIX 标准定义了一套语言规范，包括访问核心服务的编程语言标准接口和特殊语言服务的标准接口。POSIX 标准基于 C 语言，面向应用，支持实时扩展，无超级用户和系统管理，对原有程序代码和原有实现进行最小代价的封装和修改，并对各种接口设备以及各种数据格式进行定义和说明。

1.3.6 实时操作系统的典型应用

实时操作系统主要应用在需要实时处理大数据量或大运算量的并行系统中。

航空航天领域采用硬实时系统来确保时间。例如，美国“好奇号”火星探测车就是采用 VxWorks 硬实时操作系统。

在工业制造领域，涉及人身安全或执行重要任务时通常采用硬实时操作系统，如飞机控制系统以及汽车安全气囊系统等。

在军事领域也通常采用性能稳定的实时系统，如美国的 F/A-18、F-16 战斗机以及“爱国者”导弹等。

在通信或电子产品领域，例如，IPTV 机顶盒通常采用软实时操作系统对视频进行实时解码。

1.4 嵌入式系统的软件


微课视频

嵌入式系统的软件一般固化于嵌入式存储器中，是嵌入式系统的控制核心，控制着嵌入式系统的运行，实现嵌入式系统的功能。由此可见，嵌入式系统的软件在很大程度上决定整个嵌入式系统的价值。

从软件结构上划分，嵌入式系统的软件分为无操作系统和带操作系统两种。

1.4.1 无操作系统的嵌入式软件

对于通用计算机，操作系统是整个软件的核心，不可或缺。然而，对于嵌入式系统，由于其专用性，在某些情况下无需操作系统。尤其在嵌入式系统发展的初期，由于较低的硬件配置、单一的功能需求以及有限的应用领域（主要集中在工业控制和国防军事领域），嵌入式软件的规模通常较小，没有专门的操作系统。

在组成结构上，无操作系统的嵌入式软件仅由引导程序和应用程序两部分组成，如图 1-2 所示。引导程序一般由汇编语言编写，在嵌入式系统上电后运行，完成自检、存储映射、时钟系统和外设接口配置等一系列硬件初始化操作。应用程序一般由 C 语言编写，直接架构在硬件之上，在引导程序之后运行，负责实现嵌入式系统的主要功能。

图 1-2 无操作系统的嵌入式软件结构

1.4.2 带操作系统的嵌入式软件

随着嵌入式应用在各个领域的普及和深入，嵌入式系统向多样化、智能化和网络化发展，

其对功能、实时性、可靠性和可移植性等方面的要求越来越高，嵌入式软件日趋复杂，越来越多地采用嵌入式操作系统 + 应用软件的模式。相比于无操作系统的嵌入式软件，带操作系统的嵌入式软件规模较大，其应用软件架构于嵌入式操作系统上，而非直接面对嵌入式硬件，可靠性高，开发周期短，易于移植和扩展，适用于功能复杂的嵌入式系统。

带操作系统的嵌入式软件的体系结构如图 1-3 所示，自下而上包括设备驱动层、操作系统层和应用软件层等。

图 1-3　带操作系统的嵌入式软件的体系结构

1.4.3　嵌入式操作系统的分类

按照嵌入式操作系统对任务响应的实时性来分类，嵌入式操作系统可以分为嵌入式非实时操作系统和嵌入式实时操作系统（RTOS），它们的主要区别在于任务调度处理方式不同。

1. 嵌入式非实时操作系统

嵌入式非实时操作系统主要面向消费类产品应用领域。大部分嵌入式非实时操作系统都支持多用户和多进程，负责管理众多的进程并为它们分配系统资源，属于不可抢占式操作系统。非实时操作系统尽量缩短系统的平均响应时间并提高系统的吞吐率，在单位时间内为尽可能多的用户请求提供服务，注重平均表现性能，不关心个体表现性能。例如，对于整个系统，注重所有任务的平均响应时间而不关心单个任务的响应时间；对于某个单个任务，注重每次执行的平均响应时间而不关心某次特定执行的响应时间。典型的嵌入式非实时操作系统有 Linux、iOS 等。

2. 嵌入式实时操作系统

嵌入式实时操作系统主要面向控制、通信等领域。嵌入式实时操作系统除了要满足应用的功能需求，还要满足应用提出的实时性要求，属于抢占式操作系统。嵌入式实时操作系统能及时响应外部事件的请求，并以足够快的速度予以处理，其处理结果能在规定的时间内控制、监控生产过程或对处理系统作出快速响应，并控制所有任务协调、一致地运行。因此，嵌入式实时操作系统采用各种算法和策略，始终保证系统行为的可预测性。这要求在系统运行的任何时刻，在任何情况下，嵌入式实时操作系统的资源调配策略都能为争夺资源（包括 CPU、内存、网络带宽等）的多个实时任务合理地分配资源，使每个实时任务的实时性要求都能得到满足（要求每个实时任务在最坏情况下都要满足实时性要求）。嵌入式实时操作系统总是执行当前优先级最高的进程，直至结束执行，中间的时间通过 CPU 频率等可以推算

出来。由于虚存技术访问时间的不可确定性，在嵌入式实时操作系统中一般不采用标准的虚存技术。典型的嵌入式实时操作系统有 VxWorks、μC/OS-Ⅱ、QNX、FreeRTOS、eCOS、RTX 和 RT-Thread 等。

1.4.4 嵌入式实时操作系统的功能

嵌入式实时操作系统满足了实时控制和实时信息处理领域的需要，在嵌入式领域应用十分广泛，一般包含实时内核、内存管理、文件系统、图形接口、网络组件等。在不同的应用中，可对嵌入式实时操作系统进行裁剪和重新配置。一般来讲，嵌入式实时操作系统需要完成以下管理功能。

1. 任务管理

任务管理是嵌入式实时操作系统的核心和灵魂，决定了操作系统的实时性能。任务管理通常包含优先级设置、多任务调度机制和时间确定性等。

嵌入式实时操作系统支持多个任务，每个任务都具有优先级，任务越重要，被赋予的优先级越高。优先级的设置分为静态优先级和动态优先级两种。静态优先级指的是每个任务在运行前都被赋予一个优先级，而且这个优先级在系统运行期间是不能改变的。动态优先级则是指每个任务的优先级（特别是应用程序的优先级）在系统运行时可以动态地改变。任务调度主要是协调任务对计算机系统资源的争夺使用，任务调度直接影响到系统的实时性能，一般采用基于优先级抢占式调度。系统中每个任务都有一个优先级，内核总是将 CPU 分配给处于就绪态的优先级最高的任务运行。如果系统发现就绪队列中有比当前运行任务优先级更高的任务，就会把当前运行任务置于就绪队列，调入高优先级任务运行。系统采用优先级抢占方式进行调度，可以保证重要的突发事件得到及时处理。嵌入式实时操作系统调用的任务与服务的执行时间应具有可确定性，系统服务的执行时间不依赖于应用程序任务的多少，因此，系统完成某个确定任务的时间是可预测的。

2. 任务同步与通信机制

实时操作系统的功能一般要通过若干任务和中断服务程序共同完成。任务与任务之间、任务与中断间任务及中断服务程序之间必须协调动作、互相配合，这就涉及任务间的同步与通信问题。嵌入式实时操作系统通常是通过信号量、互斥信号量、事件标志和异步信号实现同步的，通过消息邮箱、消息队列、管道和共享内存提供通信服务。

3. 内存管理

通常在操作系统的内存中既有系统程序，也有用户程序，为了使两者都能正常运行，避免程序间相互干扰，需要对内存中的程序和数据进行保护。存储保护通常需要硬件支持，很多系统都采用内存管理单元（Memory Management Unit，MMU），并结合软件实现这一功能。但由于嵌入式系统的成本限制，内核和用户程序通常都在相同的内存空间中。内存分配方式可分为静态分配和动态分配。静态分配是在程序运行前一次性分配相应内存，并且在程序运行期间不允许再申请或在内存中移动；动态分配则允许在程序运行的

整个过程中进行内存分配。静态分配使系统失去了灵活性，但对实时性要求比较高的系统是必需的；而动态分配赋予了系统设计者更多自主性，系统设计者可以灵活地调整系统的功能。

4. 中断管理

中断管理是实时系统中一个很重要的部分，系统经常通过中断与外部事件交互。评估系统的中断管理性能主要考虑的是否支持中断嵌套、中断处理机制、中断延时等。中断处理是整个运行系统中优先级最高的代码，它可以抢占任何任务级代码运行。中断机制是多任务环境运行的基础，是系统实时性的保证。

1.4.5　典型的嵌入式操作系统

使用嵌入式操作系统主要是为了有效地对嵌入式系统的软硬件资源进行分配、任务调度切换、中断处理，以及控制和协调资源与任务的并发活动。由于 C 语言可以更好地对硬件资源进行控制，嵌入式操作系统通常采用 C 语言编写。当然，为了获得更快的响应速度，有时也需要采用汇编语言编写一部分代码或模块，以达到优化的目的。嵌入式操作系统与通用操作系统相比在两方面有很大的区别。一方面，通用操作系统为用户创建了一个操作环境，在这个环境中，用户可以和计算机交互，执行各种各样的任务；而嵌入式系统一般只是执行有限类型的特定任务，并且一般不需要用户干预。另一方面，在大多数嵌入式操作系统中，应用程序通常作为操作系统的一部分内置于操作系统中，随操作系统启动时自动在只读存储器（Read-Only Memory，ROM）或 Flash 中运行；而在通用操作系统中，应用程序一般是由用户选择加载到 RAM 中运行的。

随着嵌入式技术的快速发展，国内外先后问世了 150 多种嵌入式操作系统，较常见的国外嵌入式操作系统有 μC/OS、FreeRTOS、Embedded Linux、VxWorks、QNX、RTX、Windows IoT Core、Android Things 等。虽然国产嵌入式操作系统发展相对滞后，但在物联网技术与应用的强劲推动下，国内厂商也纷纷推出了多种嵌入式操作系统，并得到了日益广泛的应用。目前较为常见的国产嵌入式操作系统有华为 Lite OS、华为 HarmonyOS、阿里巴巴 AliOS Things、翼辉 SylixOS、赛睿德 RT-Thread 等。

1. 华为 Lite OS

Lite OS 是华为技术有限公司（简称华为）于 2015 年 5 月发布的轻量级开源物联网嵌入式操作系统，遵循 BSD-3 开源许可协议。其内核包括任务管理、内存管理、时间管理、通信机制、中断管理、队列管理、事件管理、定时器、异常管理等操作系统的基础组件。各组件均可以单独运行。另外，Lite OS 还提供了软件开发工具包 Lite OS SDK。目前 Lite OS 支持 Arm Cortex-M0/M3/M4/M7 等芯片架构，适配了包括 ST、NXP、GD、MindMotion、Silicon、Atmel 等主流开发商的开发板，具备零配置、自发现和自组网的能力。

Lite OS 的主要特点如下。

（1）高实时性、高稳定性。

（2）超小内核，基础内核体积可以裁剪至不到 10KB。

（3）低功耗，最低功耗可在微瓦（μW）级。

（4）支持动态加载和分散加载。

（5）支持功能静态裁剪。

（6）开发门槛低，设备布置以及维护成本低，开发周期短，可广泛应用于智能家居、个人穿戴、车联网、城市公共服务、制造业等领域。

2. 华为 HarmonyOS

HarmonyOS（鸿蒙 OS）是华为推出的基于微内核的全场景分布式嵌入式操作系统。HarmonyOS 采用了微内核设计，通过简化内核功能，使内核只提供多进程调度和多进程通信等最基础的服务，而让内核之外的用户态尽可能多地实现系统服务，同时添加了相互之间的安全保护，拥有更强的安全特性和更低的时延。HarmonyOS 使用确定时延引擎和高性能进程间通信（Inter-Process Communication，IPC）两大技术解决现有系统性能不足的问题。确定时延引擎可在任务执行前分配系统中任务执行优先级及时限，优先级高的任务资源将优先保障调度，同时微内核结构小巧的特性使 IPC 性能大大提高。HarmonyOS 的分布式 OS 架构和分布式软总线技术具备公共通信平台、分布式数据管理、分布式能力调度和虚拟外设四大能力，能够将分布式应用底层技术的实现难度对应用开发者进行屏蔽，使开发者聚焦于自身业务逻辑，像开发同一终端应用那样开发跨终端分布式应用，实现跨终端的无缝协同。HarmonyOS 2.0 已在智慧屏、PC、手表 / 手环和手机上获得应用，并将覆盖到音箱、耳机以及虚拟现实（Virtual Reality，VR）眼镜等应用产品中。

3. 阿里巴巴 AliOS Things

AliOS Things 是阿里巴巴集团控股有限公司（简称阿里巴巴）面向物联网领域推出的轻量级开源物联网嵌入式操作系统。除操作系统内核外，AliOS Things 包含了硬件抽象层、板级支持包、协议栈、中间件、AOS API 以及应用示例等组件，支持各种主流的 CPU 架构，包括 Arm Cortex-M0+/M3/M4/M7/A7/A53/A72、RISC-V、C-SKY、MIPS-I 和 Renesas 等。AliOS Things 采用了阿里巴巴自主研发的高效实时嵌入式操作系统内核，该内核与应用在内存及硬件的使用上严格隔离，在保证系统安全性的同时，具备极致性能，如极简开发、云端一体、丰富组件、安全防护等关键能力。AliOS Things 支持终端设备到阿里云 Link 的连接，可广泛应用在智能家居、智慧城市、新出行等领域，正在成长为国产自主可控、云端一体化的新型物联网嵌入式操作系统。AliOS Things 已应用于互联网汽车、智能电视、智能手机、智能手表等不同终端，正在逐步形成强大的阿里云物联网（Internet of Thing，IoT）生态。

4. 翼辉 SylixOS

SylixOS 是由北京翼辉信息技术有限公司推出的开源嵌入式实时操作系统，从 2006 年开始研发，经过多年的持续开发与改进，已成为一个功能全面、稳定可靠、易于开发的大型嵌入式实时操作系统平台。翼辉 SylixOS 采用小而巧的硬实时内核，支持 256 个优先级

抢占式调度和优先级继承，支持虚拟进程和无限多任务数，调度算法先进、高效、性能强劲。SylixOS 目前已支持 Arm、MIPS、PowerPC、x86、SPARC、DSP、RISC-V、C-SKY 等架构的处理器，包括主流国产的飞腾全系列、龙芯全系列、中天微 CK810、兆芯全系列等处理器，同时支持对称多处理器（Symmetrical Multi-Processor，SMP）平台，并针对不同的处理器提供优化的驱动程序，提高了系统的整体性能。SylixOS 支持掉电安全文件系统（True Power Safe File System，TpsFs）、文件配置表（File Allocation Table，FAT）、YAFFS（Yet Another Flash File System）、ROOTFS（根文件系统）、PROCFS（进程文件系统）、网络文件系统（Network File System，NFS）、ROMFS（只读文件系统）等多种常用文件系统，以及 Qt、MicroWindows、μC/GUI 等第三方图形库。SylixOS 还提供了完善的网络功能以及丰富的网络工具。此外，SylixOS 的应用编程接口符合《军用嵌入式实时操作系统应用编程接口》（GJB 7714—2012）和 IEEE、国际标准化组织（International Organization for Standardization，ISO）、国际电工委员会（International Electrotechnical Commission，IEC）相关操作系统的编程接口规范，用户现有应用程序可以很方便地进行迁移。目前，SylixOS 的应用已覆盖网络设备、国防安全、工业自动化、轨道交通、电力、医疗、航空航天、汽车电子等诸多领域。

5. 睿赛德 RT-Thread

RT-Thread 的全称是 Real Time-Thread，是由上海睿赛德电子科技有限公司推出的一个开源嵌入式实时多线程操作系统。RT-Thread 3.1.0 及以前的版本遵循 GPL V2+ 开源许可协议，3.1.0 以后的版本遵循 Apache License 2.0 开源许可协议。RT-Thread 主要由内核层、组件与服务层、软件包 3 部分组成。其中，内核层包括 RT-Thread 内核和 libcpu/BSP（芯片移植相关文件 / 板级支持包）。RT-Thread 内核是整个操作系统的核心部分，包括多线程及其调度、信号量、邮箱、消息队列、内存管理、定时器等内核系统对象的实现，而 libcpu/BSP 与硬件密切相关，由外设驱动和 CPU 移植构成。组件与服务层是 RT-Thread 内核之上的上层软件，包括虚拟文件系统、FinSH 命令行界面、网络框架、设备框架等，采用模块化设计，做到组件内部高内聚、组件之间低耦合。软件包是运行在操作系统平台上且面向不同应用领域的通用软件组件，包括物联网相关的软件包、脚本语言相关的软件包、多媒体相关的软件包、工具类软件包、系统相关的软件包以及外设库与驱动类软件包等。RT-Thread 支持所有主流的嵌入式微控制器（Microcontroller Unit，MCU）架构，如 Arm Cortex-M/R/A、MIPS、x86、Xtensa、C-SKY、RISC-V，即支持市场上绝大多数主流的 MCU 和 Wi-Fi 芯片。相较于 Linux 操作系统，RT-Thread 具有实时性高、占用资源少、体积小、功耗低、启动快速等特点，非常适用于各种资源受限的场合。经过多年发展，RT-Thread 已经拥有一个国内较大的嵌入式开源社区，同时被广泛应用于能源、车载、医疗、消费电子等多个行业，累计装机量超过 2000 万台，成为我国自主开发、国内最成熟稳定和装机量最大的开源嵌入式实时操作系统之一。

6. μC/OS-Ⅱ

μC/OS-Ⅱ（Micro-Controller Operating System Two）是一种基于优先级的可抢占式的硬实时内核。它属于一个完整、可移植、可固化、可裁剪的抢占式多任务内核，包含了任务调度、任务管理、时间管理、内存管理和任务间的通信和同步等基本功能。μC/OS-Ⅱ嵌入式系统可用于各类 8 位单片机、16 位和 32 位微控制器和数字信号处理器。

μC/OS-Ⅱ嵌入式系统源于 Jean J.Labrosse 在 1992 年编写的一个嵌入式多任务实时操作系统，1999 年改写后命名为 μC/OS-Ⅱ，并在 2000 年被美国航空管理局认证。μC/OS-Ⅱ系统具有足够的安全性和稳定性，可以运行在航天器等对安全要求极为苛刻的系统之上。

μC/OS-Ⅱ系统是专门为计算机的嵌入式应用而设计的。μC/OS-Ⅱ系统中 90% 的代码是用 C 语言编写的，CPU 硬件相关部分是用汇编语言编写的。总量约 200 行的汇编语言部分被压缩到最低限度，便于移植到任何一种其他 CPU 上。用户只要有标准的 ANSI 的 C 交叉编译器，有汇编器、连接器等软件工具，就可以将 μC/OS-Ⅱ系统嵌入所要开发的产品中。μC/OS-Ⅱ系统具有执行效率高、占用空间小、实时性能优良和可扩展性强等特点，目前几乎已经移植到了所有知名的 CPU 上。

μC/OS-Ⅱ系统的主要特点如下。

（1）开源性。μC/OS-Ⅱ系统的源代码全部公开，用户可直接登录 μC/OS-Ⅱ的官方网站下载，网站上公布了针对不同微处理器的移植代码。用户也可以从有关出版物上找到详尽的源代码讲解和注释。这样使系统变得透明，极大地方便了 μC/OS-Ⅱ系统的开发，提高了开发效率。

（2）可移植性。绝大部分 μC/OS-Ⅱ系统的源代码是用移植性很强的 ANSI C 语句写的，和微处理器硬件相关的部分是用汇编语言写的。汇编语言编写的部分已经压缩到最小限度，使 μC/OS-Ⅱ系统便于移植到其他微处理器上。μC/OS-Ⅱ系统能够移植到多种微处理器上的条件是：只要该微处理器有堆栈指针，有 CPU 内部寄存器入栈、出栈指令。另外，使用的 C 编译器必须支持内嵌汇编（In-line Assembly）或该 C 语言可扩展、可连接汇编模块，使关中断、开中断能在 C 语言程序中实现。

（3）可固化。μC/OS-Ⅱ系统是为嵌入式应用而设计的，只要具备合适的软、硬件工具，μC/OS-Ⅱ系统就可以嵌入用户的产品中，成为产品的一部分。

（4）可裁剪。用户可以根据自身需求只使用 μC/OS-Ⅱ系统中应用程序需要的系统服务。这种可裁剪性是靠条件编译实现的。只要在用户的应用程序中（用# define constants 语句）定义那些 μC/OS-Ⅱ系统中的功能是应用程序需要的就可以了。

（5）抢占式。μC/OS-Ⅱ系统是完全抢占式的实时内核。μC/OS-Ⅱ系统总是运行就绪条件下优先级最高的任务。

（6）多任务。μC/OS-Ⅱ系统 2.8.6 版本可以管理 256 个任务，目前预留 8 个给系统，因此应用程序最多可以有 248 个任务。系统赋予每个任务的优先级是不相同的，μC/OS-Ⅱ系统不支持时间片轮转调度法。

（7）可确定性。μC/OS-Ⅱ系统全部的函数调用与服务的执行时间都具有可确定性。也就是说，μC/OS-Ⅱ系统的所有函数调用与服务的执行时间都是可知的，即 μC/OS-Ⅱ系统服务的执行时间不依赖于应用程序任务的多少。

（8）任务栈。μC/OS-Ⅱ系统的每个任务有自己单独的栈，μC/OS-Ⅱ系统允许每个任务有不同的栈空间，以便压低应用程序对 RAM 的需求。使用 μC/OS-Ⅱ系统的栈空间校验函数，可以确定每个任务到底需要多少栈空间。

（9）系统服务。μC/OS-Ⅱ系统提供很多系统服务，如邮箱、消息队列、信号量、块大小固定的内存的申请与释放、时间相关函数等。

（10）中断管理，支持嵌套。中断可以使正在执行的任务暂时挂起。如果优先级更高的任务被该中断唤醒，则高优先级的任务在中断嵌套全部退出后立即执行，中断嵌套层数可达 255 层。

μC/OS-Ⅱ系统一些典型的应用领域如下。

（1）汽车电子方面：发动机控制、防抱死系统（Anti-lock Braking System，ABS）、全球定位系统（Global Positioning System，GPS）等。

（2）办公用品：传真机、打印机、复印机、扫描仪等。

（3）通信电子：交换机、路由器、调制解调器、智能手机等。

（4）过程控制：食品加工、机械制造等。

（5）航空航天：飞机控制系统、喷气式发动机控制等。

（6）消费电子：MP3/MP4/MP5 播放器、机顶盒、洗衣机、电冰箱、电视机等。

（7）机器人和武器制导系统等。

7. 嵌入式 Linux

Linux 诞生于 1991 年 10 月 5 日（这是第 1 次正式向外公布时间），是一套开源、免费使用和自由传播的类 UNIX 操作系统。Linux 是一个基于 POSIX 和 UNIX 的支持多用户、多任务、多线程和多 CPU 的操作系统。它能运行主要的 UNIX 工具软件、应用程序和网络协议，支持 32 位和 64 位硬件。Linux 继承了 UNIX 以网络为核心的设计思想，是一个性能稳定的多用户网络操作系统，存在许多不同的版本，但它们都使用了 Linux 内核。Linux 可安装在计算机硬件中，如手机、平板电脑、路由器、视频游戏控制台、台式计算机、大型机和超级计算机。

Linux 遵循 GPL（GNU 通用公共许可证）协议，无须为每例应用交纳许可证费，并且拥有大量免费且优秀的开发工具和庞大的开发人员群体。Linux 有大量应用软件，并且其中大部分都遵循 GPL 协议，是源代码开放且免费的，可以在稍加修改后应用于用户自己的系统，因此软件的开发和维护成本很低。Linux 完全使用 C 语言编写，应用入门简单，只要懂操作系统原理和 C 语言即可。Linux 内核精悍，运行所需资源少，稳定，并具备优秀的网络功能，十分适合嵌入式操作系统应用。

嵌入式 Linux 具有以下特点。

（1）嵌入式 Linux 是完全开源的，因此它广泛应用于高校教学。研究嵌入式 Linux 代码的专家、学者远比研究其他操作系统的多，而且 Internet 上的资源丰富，还有大量的图书、资料，使学习 Linux 系统的代价最小。

（2）嵌入式 Linux 是免费的，不涉及任何版权和专利。这一点被商界所看重，因此，大部分嵌入式产品在初期都使用过嵌入式 Linux 版本。嵌入式 Linux 被很多团体和组织二次开发后，形成具有独立知识产权的嵌入式操作系统，所以，嵌入式 Linux 变种系统非常多，如 WindRiver Linux 和 μCLinux 等。

（3）嵌入式 Linux 与 Qt 相结合，使嵌入式 Linux 具有良好的图形人机界面，甚至可以和 Windows CE 相媲美，而且 Qt 目前也是开源的。

（4）嵌入式 Linux 的移植能力强，其变种形式几乎可应用于所有主流嵌入式系统中。嵌入式 Linux 对外设的驱动能力很强，驱动接口程序设计相对容易，网络上有大量常用设备的驱动代码可供参考借鉴。

（5）嵌入式 Linux 在内核、文件系统、网络支持等方面均有突出的特点。新的 Linux 内核具有 200 多万行源代码，可支持 32 个 CPU，实时性显著提高（但严格意义上不是实时操作系统），采用了更有效的任务调度器，增加了对多种嵌入式处理器的支持，在多媒体和网络通信方面也有很大提高。

8. VxWorks

VxWorks 是美国 WindRiver 公司于 1983 年设计研发的一种嵌入式实时操作系统，具有良好的持续发展能力、可裁剪微内核结构、高效的任务管理、灵活的任务间通信、微秒级的中断处理、友好的开发环境等优点。由于其良好的可靠性和卓越的实时性，VxWorks 被广泛地应用在通信军事、航空、航天等高精尖技术及实时性要求极高的领域，如卫星通信、军事演习、弹道制导、飞机导航等。VxWorks 不提供源代码，只提供二进制代码和应用接口。

VxWorks 具有以下特点。

（1）可靠性极高。VxWorks 通过了 Do-178B、ARINC 653 和 IEC 61508 等标准严格的安全性验证，因而它主要应用于军事、航空、航天等对安全性和实时性要求极高的场合。稳定性和可靠性高是 VxWorks 最受欢迎的特点。

（2）实时性好。实时性是指能够在限定时间内执行完规定功能并对外部异步事件作出响应的能力。VxWorks 系统实时性极好，系统本身开销很小，进程调度、进程间通信、中断处理等系统程序精练有效，造成的任务切换延时很短，提供了优先级抢先式和时间片轮换方式多任务调度，使硬件系统发挥出最好的实时性。例如，美国的 F-16 战斗机、B-2 隐身轰炸机和“爱国者”导弹，甚至 1997 年的火星探测器上也使用了 VxWorks 系统。

（3）可裁剪性好。VxWorks 内核只有 8KB，其他系统模块可根据需要定制，使 VxWorks 系统具有灵活的可裁剪性，既可用于极小型单片系统，也可用于大规模网络系统。VxWorks 的存储脚本（Memory Footprint）可以指定系统运行内存空间大小（这里的存储脚

本可理解为基于 VxWorks 的应用程序可执行代码）。

（4）开发环境友好。基于图形化的集成开发环境 WindRiver Workbench，可开展基于 VxWorks 和 WindRiver Linux 系统应用的工程开发。WindRiver Workbench 是一个完备的设计、调试、仿真和工程集成解决方案。

9. Android

目前，Google 的 Android 系统已经是家喻户晓的嵌入式操作系统，也是苹果（Apple）公司的 iOS 的主要竞争对手。有趣的是，正是依靠与 iOS 的商业竞争，Android 系统才得以诞生和发展。

Android 系统基于 Linux 系统，是 Google 在 2005 年并购 Danger 公司后发展其 Android 计划的成果（当时由于 iPhone 取得了巨大成功，该计划实质上制定了与 iOS 竞争的策略）。Andy Rubin 是这个计划的负责人，该计划主要针对智能手持设备。Android 的运行库文件只有 250KB，最基本内存配置为 32MB 内存、32MB 闪存和 200MHz 处理器。

作为嵌入式操作系统，比较 Android、Windows CE 和 iOS 的意义不大，因为它们都实现了对硬件资源的抽象和美观的图形用户界面，并且 Android 系统是开源的。但是，Android 系统还可被视为一个应用系统，其集成的一些软件的附加值相当高。例如，Google 地图以及与 Google 地图相关的生活关爱软件能从根本上为人们节省时间并改善人们的生活。此外，多媒体娱乐软件和基于云计算与网络服务的软件也相当出色，这些是 Android 系统的独特优势。

开发 Android 系统应用程序与开发 Windows CE 应用程序类似，可基于软件开发工具包（Software Development Kit，SDK）和 Eclipse 集成开发环境，或基于 Android Studio 集成开发环境实现，就目前来说，相对于 Windows CE 和 iOS，Android 系统还没有明显的劣势。

Android 是一种基于 Linux 的自由及开放源代码的操作系统，主要应用于移动设备，如智能手机和平板电脑。Android 逐渐扩展到其他领域，如电视、数码相机、游戏机、智能手表等。

10. Windows CE

Windows Embedded Compact（即 Windows CE）是微软公司嵌入式、移动计算平台的基础，它是一个可抢先式、多任务、多线程并具有强大通信能力的 32 位嵌入式操作系统，是微软公司为移动应用、信息设备、消费电子和各种嵌入式应用而设计的实时系统，目标是实现移动办公、便携娱乐和智能通信。

Windows CE 是模块化的操作系统，主要包括 4 个模块，即内核（Kernel）、文件子系统、图形窗口事件子系统（GWES）和通信模块。其中，内核负责进程与线程调度、中断处理、虚拟内存管理等；文件子系统管理文件操作、注册表和数据库等；图形窗口事件子系统包括图形界面、图形设备驱动和图形显示 API 函数等；通信模块负责设备与 PC 间的互联和网络通信等。目前 Windows CE 的最高版本为 7.0，作为 Windows 10 操作系统的移动版。Windows CE 支持 4 种处理器架构，即 x86、MIPS、Arm 和 SH4，同时支持多媒体设备、图

形设备、存储设备、打印设备和网络设备等多种外设。除了在智能手机方面得到广泛应用之外，Windows CE 也被应用于机器人、工业控制、导航仪、掌上电脑和示波器等设备上。

相对于其他嵌入式实时操作系统，Windows CE 具有以下优点。

（1）具有美观的图形用户界面，而且该界面与桌面 Windows 系统一脉相承，操作直观简单。

（2）开发基于 Windows CE 的应用程序相对简单，因为 Windows CE 的 API 函数集是桌面 Windows 系统 API 函数的子集，熟悉桌面 Windows 程序设计的程序员可以很快地掌握 Windows CE 应用程序的设计方法，所以 Windows CE 应用程序的开发成本较低。

（3）Windows CE 的文件管理功能非常强大，支持桌面 Windows 系统的 FAT、FAT32 等。

（4）Windows CE 的可移植性较好。

（5）Windows CE 的设备驱动程序开发相对容易。

（6）Windows CE 的电源管理功能较好，主要体现在 Windows Phone 上。

（7）Windows CE 的进程管理和中断处理机制较好。

（8）Windows CE 支持桌面 Windows 系统的众多文件格式，如 Word 和 Excel 等，这种兼容性方便桌面 Windows 用户在 Windows CE 设备上处理文档和数据。


微课视频

Windows CE 凭借上述突出优点，在便携设备、信息家电和工业监控等领域得到了广泛的应用。

1.4.6 软件架构选择建议

从理论上讲，基于操作系统的开发模式具有快捷、高效的特点，开发的软件移植性、后期维护性、程序稳健性等都比较好。但不是所有系统都要基于操作系统，因为这种模式要求开发者对操作系统的原理有比较深入的掌握，一般功能比较简单的系统，不建议使用操作系统，毕竟操作系统也占用系统资源；也不是所有系统都能使用操作系统，因为操作系统对系统的硬件有一定的要求。因此，在通常情况下，虽然 STM32 单片机是 32 位系统，但不主张嵌入操作系统。如果系统足够复杂，任务足够多，又或者有类似于网络通信、文件处理、图形接口需求加入，不得不引入操作系统管理软硬件资源时，也要选择轻量化的操作系统，如选择 μC/OS-Ⅱ的比较多，其相应的参考资源也比较多；建议不要选择 Linux、Android 和 Windows CE 这样的重量级操作系统，因为 STM32F1 系列微控制器硬件系统在未进行扩展时，是不能满足此类操作系统的运行需求的。

1.5 嵌入式系统的分类

嵌入式系统应用非常广泛，其分类也可以有多种多样的方式。可以按嵌入式系统的应用对象进行分类，也可以按嵌入式系统的功能和性能进行分类，还可以按嵌入式系统的结构复杂度进行分类。

1.5.1　按应用对象分类

按应用对象分类，嵌入式系统主要分为军用嵌入式系统和民用嵌入式系统两大类。

军用嵌入式系统又可分为车载、舰载、机载、弹载、星载等，通常以机箱、插件甚至芯片形式嵌入相应设备和武器系统之中。军用嵌入式系统除了在体积小、重量轻、性能好等方面的要求之外，往往也对苛刻工作环境的适应性和可靠性提出了严格的要求。

民用嵌入式系统又可按其应用的商业、工业和汽车等领域进行分类，主要考虑的是温度适应能力、抗干扰能力以及价格等因素。

1.5.2　按功能和性能分类

按功能和性能分类，嵌入式系统主要分为独立嵌入式系统、实时嵌入式系统、网络嵌入式系统和移动嵌入式系统等类别。

独立嵌入式系统是指能够独立工作的嵌入式系统，它们从模拟或数字端口采集信号，经信号转换和计算处理后，通过所连接的驱动、显示或控制设备输出结果数据。常见的计算器、音视频播放器、数码相机、视频游戏机、微波炉等就是独立嵌入式系统的典型应用。

实时嵌入式系统是指在一定的时间约束（截止时间）条件下完成任务执行过程的嵌入式系统。根据截止时间的不同，实时嵌入式系统又可分为硬实时嵌入式系统和软实时嵌入式系统。硬实时嵌入式系统是指必须在给定的时间期限内完成指定任务，否则就会造成灾难性后果的嵌入式系统，如在军事、航空航天、核工业等一些关键领域应用的嵌入式系统。软实时嵌入式系统是指偶尔不能在给定时间范围内完成指定的操作，或在给定时间范围外执行的操作仍然是有效和可接受的嵌入式系统，如人们日常生活中所使用的消费类电子产品、数据采集系统、监控系统等。

网络嵌入式系统是指连接着局域网、广域网或互联网的嵌入式系统。网络连接方式可以是有线的，也可以是无线的。嵌入式网络服务器就是一种典型的网络嵌入式系统，其中所有嵌入式设备都连接到网络服务器，并通过 Web 浏览器进行访问和控制，如家庭安防系统、ATM、物联网设备等。这些系统中所有传感器和执行器节点均通过某种协议进行连接、通信与控制。网络嵌入式系统是目前嵌入式系统中发展最快的分类。

移动嵌入式系统是指具有便携性和移动性的嵌入式系统，如手机、手表、智能手环、数码相机、便携式播放器以及智能可穿戴设备等。移动嵌入式系统是目前嵌入式系统中最受欢迎的分类。

1.5.3　按结构复杂度分类

按结构复杂度分类，嵌入式系统主要分为小型嵌入式系统、中型嵌入式系统和复杂嵌入式系统三大类。

小型嵌入式系统通常是指以 8 位或 16 位处理器为核心设计的嵌入式系统。其处理器的

内存（RAM）、程序存储器（ROM）和处理速度等资源都相对有限，应用程序一般用汇编语言或嵌入式 C 语言编写，通过汇编器或 / 和编译器进行汇编或 / 和编译后生成可执行的机器码，并采用编程器将机器码烧写到处理器的程序存储器中。例如，电饭锅、洗衣机、微波炉、键盘等就是小型嵌入式系统的一些常见应用。

中型嵌入式系统通常是指以 16 位、32 位处理器或数字信号处理器为核心设计的嵌入式系统。这类嵌入式系统相较于小型嵌入式系统具有更高的硬件和软件复杂性，嵌入式应用软件主要采用 C 语言、C++ 语言、Java 语言、实时操作系统、调试器、模拟器和集成开发环境等工具进行开发，如 POS 机、不间断电源（Uninterruptible Power Supply，UPS）、扫描仪、机顶盒等。

复杂嵌入式系统与小型和中型嵌入式系统相比具有极高的硬件和软件复杂性，实现更为复杂的功能，需要采用性能更高的 32 位或 64 位处理器、专用集成电路（Application Specific Intergrated Circuit，ASIC）或现场可编程逻辑阵列（FPGA）器件进行设计。这类嵌入式系统有着很高的性能要求，需要通过软、硬件协同设计的方式将图形用户界面、多种通信接口、网络协议、文件系统甚至数据库等组件进行有效封装。例如，网络交换机、无线路由器、IP 摄像头、嵌入式 Web 服务器等系统就属于复杂嵌入式系统。


微课视频

1.6 嵌入式系统的应用领域

嵌入式系统主要应用在以下领域。

1. 工业控制

基于嵌入式芯片的工业自动化设备将获得长足的发展，目前已经有大量的 8 位、16 位、32 位嵌入式微控制器在应用中，网络化是提高生产效率和产品质量、节省人力资源的主要途径，如工业过程控制、数字机床、电力系统、电网安全、电网设备监测、石油化工系统。就传统的工业控制产品而言，低端型采用的往往是 8 位单片机，但是随着技术的发展，32 位、64 位的处理器逐渐成为工业控制设备的核心，在未来几年内必将快速发展。

2. 交通管理

在车辆导航、流量控制、信息监测与汽车服务方面，嵌入式系统已经获得了广泛的应用，如内嵌 GPS 模块、全球移动通信系统（Global System for Mobile Communication，GSM）模块的移动定位终端已经在各种运输行业获得了成功的使用，目前 GPS 设备已经从尖端领域进入了普通百姓的家庭。

3. 信息家电

信息家电将成为嵌入式系统最大的应用领域，冰箱、空调等的网络化、智能化将引领人们的生活步入一个崭新的空间。即使用户不在家里，也可以通过电话线、网络进行远程控制，在这些设备中，嵌入式系统将大有用武之地。

4. 家庭智能管理系统

水、电、煤气的远程自动抄表，安全防火、防盗系统，其中嵌有的专用控制芯片将代替传统的人工检查，并实现更高、更准确和更安全的性能。目前在服务领域，如远程点菜器等已经体现了嵌入式系统的优势。

5. POS 网络及电子商务

公共交通无接触智能卡（Contactless Smartcard，CSC）发行系统、公共电话卡发行系统、自动售货机、各种智能 ATM 终端将全面走入人们的生活，手持一卡就可以行遍天下。

6. 环境工程与自然

嵌入式系统在水文资料实时监测、防洪体系及水土质量监测、堤坝安全、地震监测网、实时气象信息网、水源和空气污染监测等方面有很广泛的应用，在很多环境恶劣、地况复杂的地区，将实现无人监测。

7. 机器人

嵌入式芯片的发展将使机器人在微型化、高智能方面优势更加明显，同时会大幅度降低机器人的价格，使其在工业领域和服务领域获得更广泛的应用。

8. 机电产品


微课视频

相对于其他的领域，机电产品可以说是嵌入式系统应用最典型、最广泛的领域之一。从最初的单片机到现在的工控机、片上系统（System on a Chip，SoC），嵌入式系统在各种机电产品中均有着巨大的市场。

9. 物联网

嵌入式系统已经在物联网方面取得大量成果，在智能交通、POS 收银、工厂自动化等领域已经广泛应用，仅在智能交通行业就已经取得非常明显的社会效益和经济效益。

随着移动应用的发展，嵌入式系统移动应用的前景非常广阔，包括可穿戴设备、智能硬件、物联网。随着低功耗技术的发展，随身可携带的嵌入式应用将会深入人们生活的各个方面。

1.7　嵌入式系统的体系

嵌入式系统是一个专用计算机应用系统，是一个软件和硬件集合体。图 1-4 所示为一个典型嵌入式系统的组成结构。

嵌入式系统的硬件层一般由嵌入式处理器、内存、人机接口、复位 / 看门狗电路、I/O 接口电路等组成，它是整个系统运行的基础，通过人机接口和 I/O 接口实现和外部的通信。嵌入式系统的软件层主要由应用程序、硬件抽象层、嵌入式操作系统和驱动程序、板级支持包组成。嵌入式操作系统主要实现应用程序和硬件抽象层的管理，在一些应用场合可以不使用，直接编写裸机应用程序。嵌入式系统软件运行在嵌入式处理器中。在嵌入式操作系统的管理下，设备驱动层将硬件电路接收控制指令和感知的外部信息传递给应用层，经过处理后，将控制结果或数据再反馈给系统硬件层，完成存储、传输或执行等功能要求。

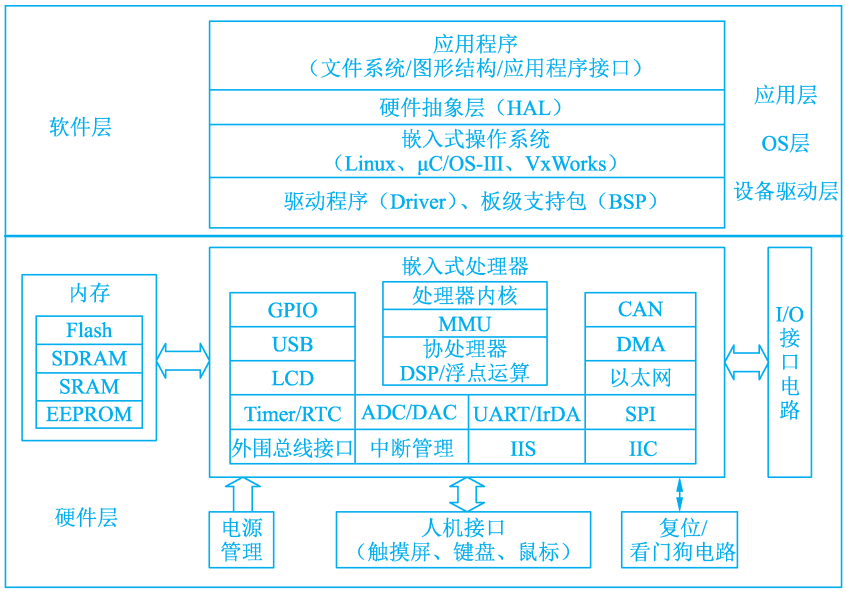

图 1-4　典型嵌入式系统的组成结构

1.7.1　硬件架构

嵌入式系统的硬件架构以嵌入式处理器为核心，由存储器、外围设备、通信模块、电源及复位等必要的辅助接口组成。嵌入式系统是量身定做的专用计算机应用系统，不同于普通计算机组成，在实际应用中的嵌入式系统硬件配置非常精简。除了微处理器和基本的外围设备，其余的电路都可根据需要和成本进行裁剪、定制，因此嵌入式系统硬件配置非常经济、可靠。

随着计算机技术、微电子技术及纳米芯片加工工艺技术的发展，以微处理器为核心的集成多种功能的 SoC 芯片已成为嵌入式系统的核心。这些 SoC 集成了大量的外围 USB、以太网、ADC/DAC、互联网信息服务（Internet Information Services，IIS）等功能模块。可编程片上系统（SOPC）结合了 SoC 和可编程逻辑器件（Programmable Logic Device，PLD）的技术优点，使系统具有可编程的功能，是可编程逻辑器件在嵌入式应用中的完美体现，极大地提高了系统在线升级、换代的能力。以 SoC/SOPC 为核心，用最少的外围器件和连接器件构成一个应用系统，以满足系统的功能需求，是嵌入式系统发展的一个方向。

因此，嵌入式系统设计是以嵌入式微处理器 /SoC/SOPC 为核心，结合外围接口设备，包括存储设备、通信扩展设备、扩展设备接口和辅助设备（电源、传感器、执行器等），构成硬件系统以完成系统设计的。

1.7.2　软件层次

嵌入式系统软件可以直接面向硬件的裸机程序开发，也可以基于操作系统的嵌入式程序

开发。当嵌入式系统应用功能简单时，相应的硬件平台结构也相对简单，这时可以使用裸机程序开发方式，不仅能够降低系统复杂度，还能够实现较好的系统实时性，但是要求程序设计人员对硬件构造和原理比较熟悉。如果嵌入式系统应用较复杂，相应的硬件平台结构也相对复杂，这时可能需要一个嵌入式操作系统管理和调度内存、多任务、周边资源等。在进行基于操作系统的嵌入式程序设计开发时，操作系统通过对驱动程序的管理，将硬件各组成部分抽象成一系列 API 函数，这样在编写应用程序时，程序设计人员就可以减少对硬件细节的关注，专注于程序设计，从而减轻程序设计人员的工作负担。

嵌入式系统软件结构一般包含 3 个层面：设备驱动层、OS 层、应用层（包括硬件抽象层、应用程序）。由于嵌入式系统应用的多样性，需要根据不同的硬件电路和嵌入式系统应用特点，对软件部分进行裁剪。现代高性能嵌入式系统的应用越来越广泛，嵌入式操作系统的使用成为必然发展趋势。

1. 设备驱动层

设备驱动层一般由板级支持包和驱动程序组成，是嵌入式系统中不可或缺的部分。设备驱动层的作用是为上层程序提供外围设备的操作接口，并且实现设备的驱动程序。上层程序可以不管设备内部实现细节，只须调用设备驱动的操作接口即可。

应用程序运行在嵌入式操作系统上，利用嵌入式操作系统提供的接口完成特定功能。嵌入式操作系统具有应用的任务调度和控制等核心功能。根据不同的应用，硬件平台所具备功能各不相同，而且所使用的硬件也不相同，具有复杂的多样性。因此，针对不同硬件平台进行嵌入式操作系统的移植是极为耗时的工作，为简化不同硬件平台间操作系统的移植问题，在嵌入式操作系统和硬件平台之间增加了硬件抽象层（HAL）。有了硬件抽象层，嵌入式操作系统和应用程序就不需要关心底层的硬件平台信息，内核与硬件相关的代码也不必因硬件的不同而修改，只要硬件抽象层能够提供必需的服务即可，从而屏蔽底层硬件，方便进行系统的移植。通常硬件抽象层是以板级支持包的形式完成对具体硬件的操作的。

1）板级支持包

板级支持包（BSP）是介于主板硬件和嵌入式操作系统中驱动程序之间的一层。BSP 是所有与硬件相关的代码的集合，为嵌入式操作系统的正常运行提供了最基本、最原始的硬件操作的软件模块。BSP 和嵌入式操作系统息息相关，为上层的驱动程序提供了访问硬件的寄存器的函数包，使之能够更好地运行于主板硬件。

BSP 具有以下三大功能。

（1）系统上电时的硬件初始化，如对系统内存、寄存器及设备的中断进行设置。这是比较系统化的工作，硬件上电初始化要根据嵌入式开发所选的 CPU 类型、硬件及嵌入式操作系统的初始化等多方面决定 BSP 应实现什么功能。

（2）为嵌入式操作系统访问硬件驱动程序提供支持。驱动程序经常需要访问硬件的寄存器，如果整个系统为统一编址，那么开发人员可直接在驱动程序中用 C 语言的函数访问硬件的寄存器。但是，如果系统为单独编址，那么 C 语言将不能直接访问硬件的寄存器，只

有汇编语言编写的函数才能对硬件的寄存器进行访问。BSP 就是为上层的驱动程序提供访问硬件的寄存器的函数包。

（3）集成硬件相关和硬件无关的嵌入式操作系统所需的软件模块。BSP 是相对于嵌入式操作系统而言的，不同的嵌入式操作系统对应于不同定义形式的 BSP。例如，VxWorks 的 BSP 和 Linux 的 BSP 相对于某一 CPU 来说尽管实现的功能一样，但是写法和接口定义是完全不同的，所以写 BSP 一定要按照该系统 BSP 的定义形式（BSP 的编程过程大多数是在某个成型的 BSP 模板上进行修改的），这样才能与上层嵌入式操作系统保持正确的接口，良好地支持上层嵌入式操作系统。

2）驱动程序

只有安装了驱动程序，嵌入式操作系统才能操作硬件平台，驱动程序控制嵌入式操作系统和硬件之间的交互。驱动程序提供一组嵌入式操作系统可理解的抽象接口函数，如设备初始化、打开、关闭、发送、接收等。一般而言，驱动程序与设备的控制芯片有关。驱动程序运行在高特权级的处理器环境中，可以直接对硬件进行操作，但正因如此，任何一个设备驱动程序的错误都可能导致嵌入式操作系统的崩溃，因此好的驱动程序需要有完备的错误处理函数。

2. OS 层

嵌入式操作系统是一种支持嵌入式系统应用的操作系统软件，是嵌入式系统的重要组成部分。嵌入式操作系统通常包括与硬件相关的底层驱动软件、系统内核、设备驱动接口、通信协议、图形界面、标准化浏览器等。嵌入式操作系统具有通用操作系统的基本特点。例如，能有效管理越来越复杂的系统资源；能把硬件虚拟化，将开发人员从繁忙的驱动程序移植和维护中解脱出来；能提供库函数、驱动程序、工具集及应用程序。与通用操作系统相比较，嵌入式操作系统在系统实时高效性、硬件的相关依赖性、软件固态化及应用的专用性等方面具有较为突出的特点。嵌入式操作系统具有通用操作系统的基本特点，能够有效管理复杂的系统资源，并且把硬件虚拟化。

一般情况下，嵌入式操作系统可以分为两类，一类是面向控制、通信等领域的嵌入式实时操作系统（RTOS），如 VxWorks、PSOS、QNX、μCOS-Ⅱ、RT-Thread、FreeRTOS 等；另一类是面向消费电子产品的嵌入式非实时操作系统，如 Linux、Android、iOS 等，这类产品包括智能手机、机顶盒、电子书等。

3. 应用层

1）硬件抽象层

硬件抽象层本质上就是一组对硬件进行操作的 API，是对硬件功能抽象的结果。硬件抽象层通过 API 为嵌入式操作系统和应用程序提供服务。但是，在 Windows 和 Linux 操作系统下，硬件抽象层的定义是不同的。

Windows 操作系统下的硬件抽象层定义：位于嵌入式操作系统的最底层，直接操作硬件，隔离与硬件相关的信息，为上层的嵌入式操作系统和驱动程序提供一个统一的接口，

起到对硬件的抽象作用。HAL 简化了驱动程序的编写，使嵌入式操作系统具有更好的可移植性。

Linux 操作系统下的硬件抽象层定义：位于嵌入式操作系统和驱动程序之上，是一个运行在用户空间中的服务程序。

Linux 和所有 UNIX 一样，习惯用文件抽象设备，任何设备都是一个文件，如 /dev/mouse 是鼠标的设备文件名。这种方法看起来不错，每个设备都有统一的形式，但使用起来并没有那么容易，设备文件名没有什么规范，从一个简单的文件名无法得知它是什么设备、具有什么特性。乱七八糟的设备文件，让设备的管理和应用程序的开发变得很麻烦，所以有必要提供一个硬件抽象层为上层应用程序提供统一的接口，Linux 的硬件抽象层就这样应运而生了。

2）应用程序

应用程序是为完成某项或某几项特定任务而被开发运行于嵌入式操作系统之上的程序，如文件操作、图形操作等。在嵌入式操作系统上编写应用程序一般需要一些应用程序接口。应用程序接口（API）又称为应用编程接口，是软件系统不同组部分衔接的约定。应用程序接口的设计十分重要，良好的接口设计可以降低系统各部分的相互依赖性，提高组成单元的内聚性，降低组成单元间的耦合程度，从而提高系统的维护性和扩展性。


微课视频

根据嵌入式系统应用需求，应用程序通过调用嵌入式操作系统的 API 函数操作系统硬件，从而实现应用需求。一般情况下，嵌入式应用程序建立在主任务基础之上，可以是多任务的，通过嵌入式操作系统管理工具（信号量、队列等）实现任务间通信和管理，进而实现应用需要的特定功能。

1.8　嵌入式系统的设计方法

1.8.1　嵌入式系统的总体结构

在不同的应用场合，嵌入式系统呈现出的外观和形式各不相同，但通过对其内部结构进行分析可以发现，一个嵌入式系统一般都由嵌入式微处理器系统和被控对象组成。其中，嵌入式微处理器系统是整个系统的核心，由硬件层、中间层、软件层和功能层组成；被控对象可以是各种传感器、电机、输入 / 输出设备等，可以接收嵌入式微处理器系统发出的控制命令，执行所规定的操作或任务。下面对嵌入式系统的主要组成进行简单描述。

1. 硬件层

硬件层由嵌入式微处理器、外围电路和外部设备组成。在一个嵌入式微处理器的基础上增加电源电路、复位电路、调试接口和存储器电路，就构成一个嵌入式核心控制模块。其中，操作系统和应用程序都可以固化在 ROM 或 Flash 中，为方便使用，有的模块在此基础上增加了液晶显示（Liquid Crystal Display，LCD）、键盘、USB 接口，以及其他一些功能的扩

展电路和接口。

嵌入式系统的硬件层是以嵌入式处理器为核心的，最初的嵌入式处理器都是为通用目的而设计的。后来，随着微电子技术的发展，出现了 ASIC，它是一种为具体任务而特殊设计的专用集成电路。由于 ASIC 在设计过程中进行了专门优化，其性能、性价比都非常高，采用 ASIC 可以降低系统软硬件设计的复杂度和系统成本。有的嵌入式微处理器利用 ASIC 实现，但 ASIC 的前期设计费用非常高，而且一旦设计完成，就无法升级和扩展，一般只有在一些产量非常大的产品设计中才考虑使用 ASIC。

2. 中间层

硬件层与软件层之间为中间层，也称为 BSP（板级支持包），将系统软件与底层硬件部分隔离，使系统的底层设备驱动程序与硬件无关，一般应具有相关硬件的初始化、数据的输入 / 输出操作和硬件设备的配置等功能。BSP 是主板硬件环境和操作系统的中间接口，是软件平台中具有硬件依赖性的那一部分，主要目的是支持操作系统，使之能够更好地运行于硬件主板上。

一般来说，纯粹的 BSP 所包含的内容是与系统有关的驱动程序，如网络驱动程序和系统中的网络协议、串口驱动程序和系统的下载调试等。离开这些驱动程序，系统就不能正常工作。

3. 软件层

软件层主要是操作系统，有的还包括文件系统、图形用户接口和网络系统等。操作系统是嵌入式应用软件的基础和开发平台，实际上是一段程序，系统复位后首先执行，相当于用户的主程序，用户的其他应用程序都建立在操作系统之上。操作系统是一个标准的内核，将中断、I/O、定时器等资源都封装起来，以方便用户使用。

操作系统的引入大大完善了嵌入式系统的功能，方便了应用软件的设计，但同时也占用了宝贵的嵌入式系统资源。一般在大型的或需要多任务的应用场合才考虑使用嵌入式操作系统。

4. 功能层

功能层由基于操作系统开发的应用程序组成，用来完成对被控对象的控制功能。为了方便用户操作，往往需要具有友好的人机界面。

对于一些复杂的系统，在系统设计的初期阶段就要对系统的需求进行分析，确定系统功能。然后将系统的功能映射到整个系统的硬件、软件和执行装置的设计过程中，这个过程称为系统的功能实现。

1.8.2 嵌入式系统设计流程

嵌入式系统的应用开发是按照一定的流程进行的，一般由 5 个阶段构成：需求分析、体系结构设计、软 / 硬件设计、系统集成和代码固化，各个阶段之间往往要求不断地重复和修改直至最终完成设计目标。

嵌入式系统开发已经逐步规范化，在遵循一般工程开发流程的基础上，必须将硬件、软件、人力等各方面资源综合起来。嵌入式系统发都是软件、硬件的结合体和协同开发过程，这是其最大的特点。嵌入式系统设计流程如图 1-5 所示。

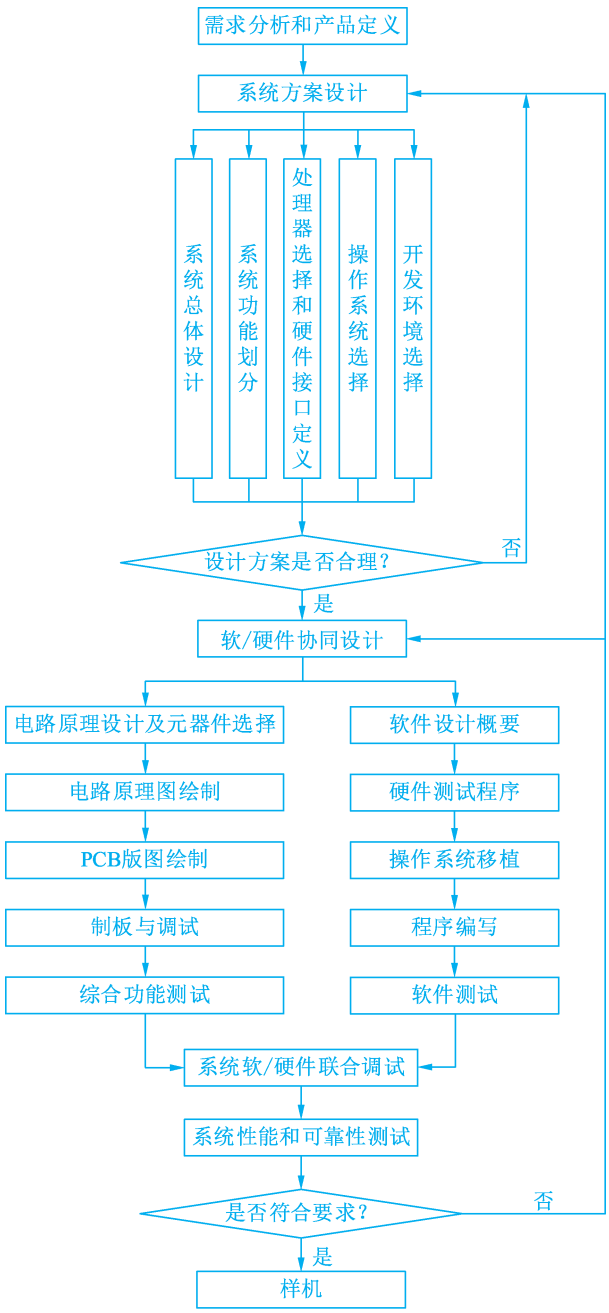

图 1-5　嵌入式系统设计流程

1. 需求分析

嵌入式系统的特点决定了系统在开发初期的需求分析中就要搞清楚需要完成的任务。在此阶段需要分析系统的需求，一般分为功能需求和非功能需求两方面。功能需求是系统的基本功能，如输入 / 输出信号、操作方式等；非功能需求包括系统性能、成本、功耗、体积、重量等因素。

根据系统的需求，确定设计任务和设计目标，并编写设计规格说明书，作为正式指导设计和验收的标准。

2. 体系结构设计

需求分析完成后，根据设计规格说明书进行体系结构的设计，包括对硬件、软件的功能划分，以及系统的硬件和操作系统的选型等。

3. 软 / 硬件设计

基于体系结构对系统的软件、硬件进行详细设计。为了缩短产品开发周期，设计往往是并行的。对于每个处理器，硬件平台都是通用的、固定的、成熟的，在开发过程中减少了硬件系统错误的引入机会。同时，嵌入式操作系统屏蔽了底层硬件的很多复杂信息，开发者利用操作系统提供的 API 函数可以完成大部分功能。对于一个完整的嵌入式应用系统的开发，应用系统的程序设计是嵌入式系统设计一个非常重要的方面，程序的质量直接影响整个系统功能的实现，好的程序设计可以克服系统硬件设计的不足，提高应用系统的性能；反之，会使整个应用系统无法正常工作。

不同于基于 PC 平台的程序开发，嵌入式系统的程序设计具有其自身的特点，程序设计的方法也会因系统或人而异。

4. 系统集成和代码固化

把系统中的软件、硬件集成在一起，进行调试，发现并改进单元设计过程中的错误。

嵌入式软件开发完成以后，大多数在目标环境的非易失性存储器中运行，程序写入 Flash 固化，保证每次运行后下一次运行无误，所以嵌入式软件开发与普通软件开发相比，增加了固化阶段。嵌入式应用软件调试完成以后，编译器要对源代码重新编译一次，以产生固化到环境的可执行代码，再烧写到 Flash。可执行代码烧写到目标环境中固化后，整个嵌入式系统的开发就基本完成了，剩下的就是对产品的维护和更新了。

1.8.3 嵌入式系统的软/硬件协同设计技术

传统的嵌入式系统设计方法，硬件和软件分为两个独立的部分，由硬件工程师和软件工程师分别按照拟定的设计流程分别完成。这种设计方法只能改善硬件、软件各自的性能，而有限的设计空间不可能对系统进行较好的性能综合优化。从理论上来说，每个应用系统都存在一个硬件、软件功能的最佳组合，如何从应用系统需求出发，依据一定的指导原则和分配算法对软/硬件功能进行分析及合理的划分，从而使系统的整体性能、运行时间、能量耗损、存储能量达到最佳状态，已成为软 / 硬件协同设计的重要研究内容之一。

系统协同设计与传统设计相比有以下两个显著的区别。

（1）描述硬件和软件使用统一的表示形式。

（2）软 / 硬件划分可以选择多种方案，直到满足要求。

显然，这种设计方法对于具体的应用系统而言，容易获得满足综合性能指标的最佳解决方案。传统方法虽然也可改进软 / 硬件性能，但由于这种改进是各自独立进行的，不一定使系统综合性能达到最佳。

传统的嵌入式系统开发采用软件开发与硬件开发分离的方式，其过程可描述如下。

（1）需求分析。

（2）软 / 硬件分别设计、开发、调试、测试。

（3）系统集成。

（4）集成测试。

（5）若系统正确，则结束，否则继续进行。

（6）若出现错误，需要对软 / 硬件分别验证和修改；返回步骤（3），再继续进行集成测试。

虽然在系统设计的初始阶段考虑了软 / 硬件的接口问题，但由于软件与硬件分别开发，各自部分的修改和缺陷很容易导致系统集成出现错误。由于设计方法的限制，这些错误不但难以定位，而且更重要的是对它们的修改往往会涉及整个软件结构或硬件配置的改动。显然，这是灾难性的。

为避免上述问题，一种新的开发方法应运而生——软 / 硬件协同设计方法。首先，应用独立于任何硬件和软件的功能性规格方法对系统进行描述，采用的方法包括有限态自动机（Finite-State Machine，FSM）、统一化的规格语言（CSP、VHDL）或其他基于图形的表示工具，其作用是对软 / 硬件统一表示，便于功能的划分和综合。然后，在此基础上对软 / 硬件进行划分，即对软 / 硬件的功能模块进行分配。但是，这种功能分配不是随意的，而是从系统功能要求和限制条件出发，依据算法进行的。完成软 / 硬件功能划分之后，需要对划分结果进行评估。一种方法是性能评估，另一种方法是对硬件、软件综合之后的系统依据指令级评价参数进行评估。如果评估结果不满足要求，说明划分方案选择不合理，需重新划分软 / 硬件模块，以上过程重复，直至系统获得一个满意的软 / 硬件实现为止。

软 / 硬件协同设计过程可描述如下。

（1）需求分析。

（2）软 / 硬件协同设计。

（3）软 / 硬件实现。

（4）软 / 硬件协同测试和验证。

这种方法的特点是在协同设计、协同测试和协同验证方面，充分考虑了软 / 硬件的关系，并在设计的每个层次上给予测试验证，尽早发现和解决问题，避免灾难性错误的出现。