嵌入式实时操作系统的昨天、今天和明天

何小庆
嵌入式系统联谊会

关键词：实时操作系统　物联网操作系统　微内核　汽车软件　虚拟化

引言

20世纪70年代，随着微处理器的出现，操作系统有了新的舞台，遗憾的是通用型操作系统没能在基于微处理器的设备中快速发展。到了80年代初，市面上开始出现被称为实时操作系统（Real Time Operating System，RTOS）的产品。得益于16位微处理器在嵌入式系统中的应用，市场对实时操作系统的需求锐增，实时操作系统于90年代进入蓬勃发展的黄金10年，典型产品有VRTX、OS-9、pSOS和VxWorks[1]。

2000年以后，片上系统（SoC）处理器渐渐上位，市场上涌现了大量高性能的嵌入式处理器（MPU），包含通信设备和智能终端的产品纷纷采用嵌入式处理器，这些产品对硬实时的需求不高。但传统的实时操作系统在设备驱动和应用软件上很难满足这类产品的需求，开源的Linux开始进入人们的视野，亦有企业开始提供Linux实时性、快速引导、开发工具和处理器移植等技术服务，比如MontaVista、TimeSys和Wind River。

今天的技术就大不一样了，现代嵌入式系统和物联网设计主要依赖低成本、高性能的32位微控制器（MCU），它们往往具有足量的片上存储空间，并提供多种外设功能。物联网架构和应用驱动分布式实时系统被采用，边缘智能兴起，端侧的实时性、通信能力和低功耗成为系统重要的指标。市面上有许多传统RTOS（现在普遍称为“嵌入式实时操作系统”）通过组件技术支持物联网应用，比如QNX和RT-Thread。新型的物联网操作系统，在实时内核基础上发展成为“端－边－云”一站式解决方案，大大提高了物联网应用开发效率，比如AliOS things、鸿蒙OS和OneOS。

开源是RTOS创新的源泉

开源是软件产业发展的重要源动力，以Linux为代表的开源操作系统在电子信息产业落地生根。然而，嵌入式实时操作系统领域没有一个可以与Linux相媲美的知名项目，历史上小有名气的RTOS开源项目主要活跃在90年代、2000年和2010年以后这三个阶段。

RTEMS开源项目

实时多处理器系统（The Real-Time Executive for Multiprocessor Systems，RTEMS）是一个开源的实时嵌入操作系统。90年代，美国开始将RTEMS用于国防系统，它早期的名称为“实时导弹系统”，后改名为“实时军用系统”。RTEMS是最早支持POSIX、TCP/IP协议和多处理器架构的RTOS，支持许多经典的微处理器，比如PPC/68K/MIPS/Arm。RTEMS项目的活跃度一直很高，现在由OAR公司负责维护。同期的开源RTOS，比如eCos被RedHat

收购，后又被放弃，名气以及活跃度不如 RTEMS。eCos 最大的特点是内核可配置，并且用 C++ 书写。

FreeRTOS开源项目

FreeRTOS 是理查德·巴里（Richard Barry）在英国创建的开源项目。FreeRTOS 开始于 2000 年初，每一个版本在正式发布成压缩包之前，都经过完整的测试以确保产品的稳定。2017 年 FreeRTOS 成为亚马逊的开源项目，FreeRTOS v10.0 以后使用 MIT 开源协议。

FreeRTOS 一开始专注于针对微控制器，代码量小、开源免费。借助物联网的蓬勃发展，FreeRTOS 成为市场领先的小型实时操作系统。2017 年，FreeRTOS 每 3 分钟就被下载一次，成为世界上最受开发者欢迎的 RTOS 之一。FreeRTOS 还有商业版本 OpenRTOS 和安全版 SAFERTOS。SAFERTOS 是面向微控制器的、经过安全认证的嵌入式实时操作系统。SAFERTOS 及其工业设计保证包已通过 TÜV SÜD 的 IEC 61508 SIL 3 预认证。

Zephyr开源项目

Zephyr 项目是由英特尔、新思科技（Synopsys）、恩智浦半导体（NXP）等公司在 2016 年发起的开源实时操作系统项目，现在由 Linux 基金会管理。该项目旨在联合整个行业的领导者，针对资源受限的小型设备构建可扩展的实时操作系统。从 Zephyr 社区的角度看，Zephyr 希望覆盖 Linux 无法支持的应用市场。

Zephyr 的历史比较短，但起点很高，这得益于发起公司的积累和项目团队多年的经验。Zephyr 最初的代码来自风河，风河的 VxWorks RTOS 在工业和航空航天领域极有影响力。与 FreeRTOS 和 Contiki 等开源 RTOS 相比，Zephyr 体系架构完整，中间件丰富。在安全设计方面，Zephyr 有缜密的考量，遵循 MISRA C 规范标准；在功能安全认证上，Zephyr 选择支持 IEC61508，并考虑支持汽车安全标准 ISO26262。这些功能，其他开源 RTOS 项目还做不到[2]。

Zephyr 有一个充满活力的国际开发社区，在物联网与人工智能新技术融合方面活跃度很高，比如支持机器学习 TensorFlow Lite for Microcontrollers 和安全架构 Arm TrustZone，Zephyr 还集成了 Arm TF-M 安全固件。

Zephyr 目前在中国的关注度比较低，但其技术上的产品发展思路很值得国产操作系统借鉴。近年来国产开源 RTOS 发展迅速，知名的产品有 RT-Thread 和 OpenHarmony。RT-Thread 是国内开发者非常熟悉的开源 RTOS，在中国物联网市场有广泛的生态基础和市场占有率，最新开源了 RT-Smart 微内核操作系统。RT-Smart 专注于对安全、多核和高性能处理器的支持，开源 RT-Smart 对高端嵌入式市场的生态建设将起到积极作用。

RTOS是物联网操作系统的底座

物联网操作系统的发展历程

物联网操作系统（IoT OS）起源于两个传感网的操作系统 TinyOS 和 Contiki 项目。最早的 IoT OS 始于 2014 年，标志性产品是 ARM Mbed OS 和 2015 年华为发布的 Lite OS，它们是 IoT OS 的典型代表。2016 年 Linux 基金会推出 Zephyr OS，2017 年阿里宣布推出 AliOS Things，2017 年亚马逊宣布推出 Amazon FreeRTOS，2019 年腾讯推出轻量级物联网操作系统 TencentOS tiny，IoT OS 驶入发展的快车道[3]。

2020 年，IoT OS 发展之势不减：中国移动推出轻量级的 IoT OS OneOS，微软推出基于 ThreadX 的 Azure RTOS，小米推出基于开源 NuttX RTOS 的物联网软件平台小米 Vela，以及近两年颇为引人注目的华为鸿蒙 OS（也称为 HarmonyOS）。华为将鸿蒙 OS 定义为新一代智能终端的操作系统，面向不同设备的智能化互联和协同，提供全场景交互体验。

什么是物联网操作系统?

无论产业界还是学术界，都还没有对 IoT OS 给出一个统一和明确的定义。在产业界，阿里把 AliOS Things 称为面向 IoT 领域的物联网轻量级嵌入式操作系统，亚马逊称 Amazon FreeRTOS 是针

对 MCU 的物联网操作系统，ARM 称 Mbed OS 是开源的、易用的、面向物联网的操作系统。学术界更侧重研究代码受限的物联网端侧设备的操作系统需求，比如低功耗、互联互通、实时性和安全等技术特性。综合 IoT OS 技术和产业生态发展，IoT OS 可以归纳出以下五大技术特征 [4]。

1. 管理物的能力 ：这里的“ 物 ”是指物联网边缘节点上的嵌入式实时低功耗设备。

2. 泛在的通信功能 ：即支持各种无线和有线、近场和远距离的通信方式和协议，比如蓝牙、Wi-Fi、Zigbee、NB-IoT、LoRa 和 NFC 等通信技术。

3. 物联网设备的可维护性 ：即要支持设备的安全动态升级（OTA）和远程维护。

4. 物联网安全 ：物联网安全是一个广泛的概念，包含设备、通信和云安全，具备防御外部入侵和篡改的能力。

5. 物联网云和边缘智能 ：通过物联网云平台完成远程设备管理、数据存储和分析、安全控制和业务支撑，通过边缘计算加入人工智能处理架构和能力。

综上所述，我们可以将 IoT OS 理解为一种面向“ 物 ”的软件平台，具备嵌入式实时操作系统、物联网的通信协议和物联网云平台三个重要部分。典型的 IoT OS 架构如图 1 所示 [5]。

RTOS是物联网操作系统的底座

目前市场上多数物联网操作系统的内核都是基于 RTOS，比如 Amazon FreeRTOS 内核是 FreeRTOS，Azure RTOS 的内核是 ThreadX，HarmonyOS 内核是 LiteOS，Mbed OS 内核是 RTX（CMSIS-RTOS 兼容）。一部分 IoT OS 为了能更好地支持高端嵌入式处理器，同时将 Linux 作为内核之一，比如 HarmonyOS。

RTOS 为 IoT 组件和应用提供任务、队列、存储和时间管理等机制。RTOS 支持电源管理机和低功耗调度机制，这对 IoT 应用非常重要。RTOS 具备广泛的 MCU 和 MPU 支持，可以适配大量的 IoT 设备，是 IoT 应用的基础架构。RTOS 可以方便地增加软件组件，为 IoT 应用带来更多便利。RTOS 支持各种编程语言和接口标准，比如底层 C 语言、高阶 C/C++、JS 和 Python（MicroPython），RTOS 支持的 POSIX 和 CMSIS API 帮助应用软件移植和重用。

大型 RTOS 已经包含了一系列基本软件组件（也称为中间件），比如 VxWorks 和 QNX 内置了 TCP/IP、文件和图形软件。小型 RTOS 只包含内核基本服务，比如 FreeRTOS 通过 FreeRTOS Plus 提供网络和文件系统基本组件。IoT 需要更多组件实现“ 万物互联 ”，比如 AT 组件完成整个 AT 命令数据交互流程，支持各种无线通信的 IoT 模组。为了适配更多的网络协议栈类型，IoT 提供了一套套接字抽象层（SAL）组件，轻量级消息协议 MQTT/CoAP 是物联网系统核心组件，此外物联网安全 TLS/DTLS/SSL 协议组件非常重要，FreeRTOS 的一个 IoT 组件范例如图 2 所示 [6]。

RTOS技术趋势

嵌入式实时操作系统随着物联网和人工智能产业的发展而不断发展和创新，智能关键系统安全（功能安全和信息安全）需求增加，汽车和工业电子处

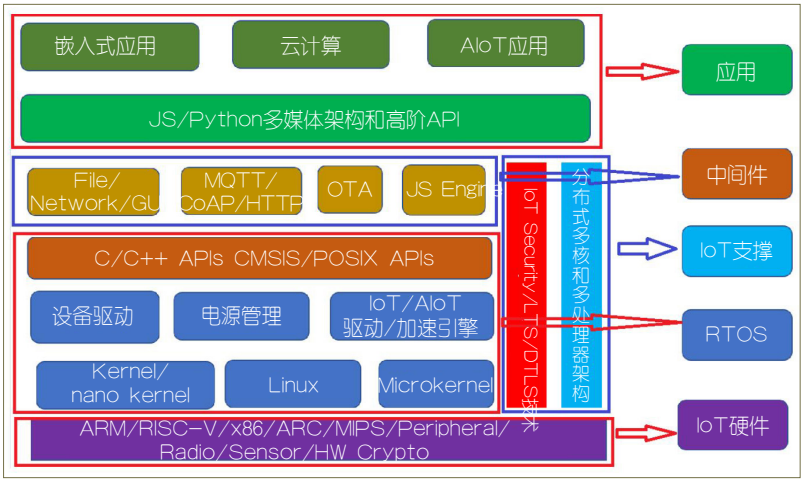

图1　典型的IoT OS 架构框图

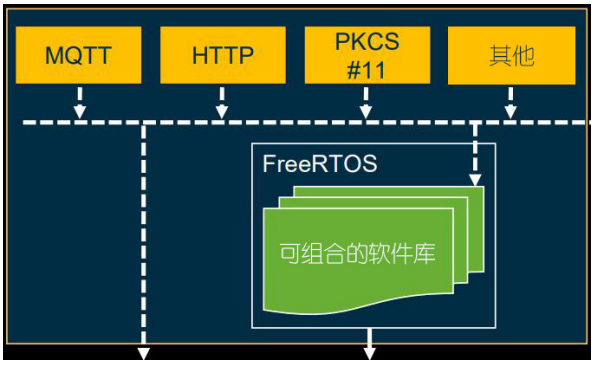

图2 FreeRTOS IoT OS组件范例

理器架构升级，多核异构处理器混合布置等，促进了微内核、安全和虚拟化技术的发展。

微内核技术

在操作系统中，内核一般指核心的、拥有完全处理器和硬件访问权限的软件部分。内核代码运行在内核态，可以运行所有的处理器指令；其外的代码运行在用户态，用户态代码不能影响系统功能和安全的关键操作。

许多传统内核遵循宏内核（monolithic）的设计思路，在内核中直接包含了调度、内存管理、驱动和组件（文件系统和网络等）等公共功能，Linux内核是典型的宏内核。微内核（microkernel）的概念于20世纪80年代出现，其主要设计思路是内核只提供最低限度的内核功能，这减少了运行在内核态的代码数量。微内核主要提供资源管理、基本调度和进程间通信（IPC）功能，其他的操作系统功能由用户态的服务提供，例如驱动程序和网络协议。

微内核具备更高的系统灵活性和可扩展性，可以更快地适应新硬件和应用。当系统针对平台进行修改时，只有一部分组件需要改动。微内核具有更高的可靠性，当系统的一部分失效时不会影响到内核，整个操作系统不会崩溃。相比宏内核，微内核在性能上也有弱点，微内核进程需要通过IPC向提供服务的进程发送服务请求，并等待服务进程通过IPC返回结果。这一过程中的上下文切换和传送消息产生的开销会不可避免地影响性能。

知名的开源微内核有L4和针对形式化安全认证开发的seL4，成功的商业微内核RTOS是QNX，第一代QNX Neutrino于1995年面世，QNX Neutrino微内核RTOS最新版本为7.X。Neutrino RTOS平台通过了ISO/IEC 15408、IEC 61508 SIL3、ISO 26262 ASIL D等安全认证，并被广泛应用于汽车、工业自动化、制造、公共交通、医疗等行业领域。RT-smart是基于RT-Thread衍生的新分支，面向带MMU的中高端嵌入式处理器，是一个开源软件。它采用微内核的设计思想，将系统的服务组件设计成一个个独立的可执行程序，以进程的形式在用户态运行，相互之间通过内核提供的通道机制完成数据交换，系统结构见图3[7]。

微内核在诞生以来的40年中不断进化，日渐成熟的理论和实践让微内核从最初饱受性能所困，发展到可广泛应用于嵌入式系统和关键安全设备中。我们有理由相信，已经证明了的安全性、可靠性和系统可自定义性的微内核在物联网和人工智能时代将会有更大的用武之地。

多处理器操作系统

嵌入式多处理器系统分为基于芯片的设计（多核芯片）和板级设计（多处理器计算系统）。比如多核处理器芯片含有两个CPU，并共同包含一些关键器件：中断管理、存储和定时器，这样的处理器嵌入到一颗芯片的CPU里，这颗CPU还包含各种

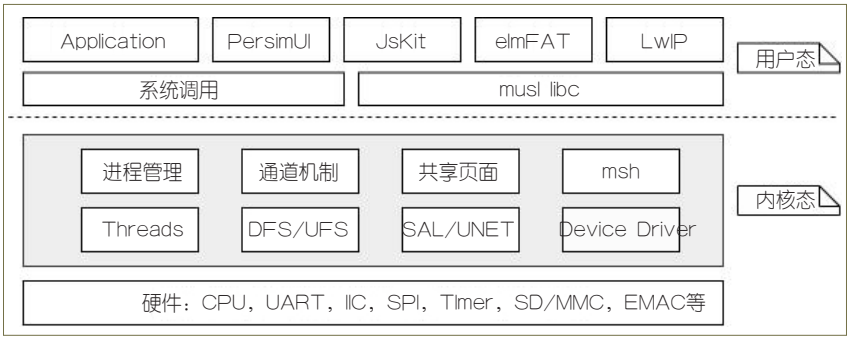

图3 微内核RT-Smart系统结构

应用器件。相反，多处理器计算架构则包含一系列单板机，每块单板上含有一颗 CPU。这些 CPU 的品种不一定相同，比如第一块板子上是一颗典型的 Arm Cortex CPU，第二块板子上则是一个 DSP。

从硬件的视角看，多处理器有两种形式 ：同构和异构。在一个同构多处理器设计中，每个处理器单元都是一样的，而异构的多处理器单元是不同的。

从软件的视角看，人们将这两种形式称作对称和非对称。非对称多处理器系统的操作系统和调度问题通常并不复杂，每一个处理器都有自己的执行程序（或者是一个全功能的 RTOS），适合规划好的作业 / 任务。比如，TMS320DM6443 可以在它的 ARM 处理器核上运行 Linux OS，在 DSP 处理器核上运行 TI 的实时内核（TI-RTOS）。这种操作模式通常被称为非对称多处理器（AMP）。对称多处理器方案就复杂多了，它有不同的模式，比如任何任务 / 线程均可在任意一个处理器上运行，即同构的多处理器模式（SMP），或者任务 / 线程与指定的处理器绑定，这种模式也称为绑定多处理器（BMP）[8]。然而，我们看到针对更广泛的嵌入式应用，SMP 不是最佳的解决方案，这是因为：

1. 通常情况下，SMP 应用无法预测每一个线程的执行顺序，即使我们使用了优先级可抢占的调度器。如果执行顺序是关键要素（比如需要协调的任务），设计代码的时候需要明确保证这样的执行顺序。

2.SMP 应用无法预测进程在哪个核上运行，当软件完美无瑕时，这当然不是个问题，但当软件有 bug 的时候，则需要判断是哪个处理器出现了 bug。

市场上商业的 RTOS 中，QNX 操作系统有 20 年多核处理器经验，支持 AMP、SMP 和 BMP 模式。国产 Sylix OS 操作系统对高端嵌入式处理器 SMP 的支持很成熟，对国产芯片支持的力度更大。开源的 RTOS 比如 RTEMS 和 RT-Thread 支持 SMP 和 AMP，FreeRTOS 也支持 SMP。

虚拟化技术是另一个重要的趋势。借助对底层处理器内核、内存和外设的抽象，这种技术使多个虚拟机可以运行在同一个物理处理器上。虚拟化提供了多操作系统的运行环境，例如可以在同一个设备中同时运行高实时性操作系统（例如 Wind River VxWorks）和通用操作系统（比如 Wind River Linux）。虚拟化技术是今天多处理器系统全新的解决方案，它平衡了性能与可靠性两方面需求。智能工业场景下的混合关键系统应用可以借助多处理器系统以及虚拟化技术布置，如图 4 所示的 Intewell 操作系统的解决方案 [9]。

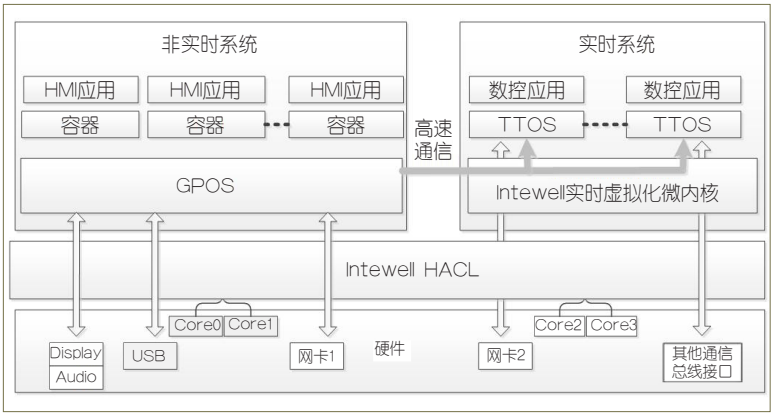

图4　可配置混合异构系统体系结构

汽车软件是RTOS发展的新机遇

近年智能和电动汽车发展迅猛，汽车电子电气系统结构正在经历一场革命，汽车中使用的软件出现了惊人的增长，使开发汽车嵌入式软件的工程师面临更大的挑战。为了实现汽车软件功能安全（safety），需要遵循汽车安全开发标准。除功能安全外，汽车软件还需考虑信息安全（security），软件的可重用性，以及确保软件具有满足应用要求的技术能力。

汽车安全标准和方法

MISRA C 是一套由汽车产业软件可靠性协会（MISRA）提出的 C 语言开发标准，其目标是促进嵌入式系统，特别是用 C 语言编程的系统中的代码安全性、可移植性和可靠性，通过强制实施良好的编码实践，产生更安全、行为可预测的代码。

ISO 26262 标准定义了汽车设备的功能安全，适

用于所有汽车电子和电气安全相关系统的整个生命周期。ISO 26262 的第 6 部分规定了汽车应用软件开发的要求，定义了 ASIL A/B/C/D 安全等级，通过对潜在危害执行风险分析，检查车辆运行场景的故障严重性（severity of failure）、暴露度（probability of exposure）和可控性（controllability），实现安全等级划分。

ISO 26262 标准中还希望使用软件运行时监视器检测、监测和处理 ASIL C 和 D 安全等级的系统故障。为了完全满足该需求，需要额外的监控功能，部分 RTOS 可以提供此类机制（如 SAFERTOS）。

在现代网联汽车中，信息安全非常重要。功能安全软件需要很长时间开发和验证，确保稳健可靠，因此很少更新 ；而安全威胁在不断演变，攻击变得越来越复杂，这导致软件须定期更新以抵御黑客攻击。但 ISO 26262 第 6 部分仅规定了软件开发标准，不涉及信息安全。

汽车软件和操作系统

汽车软件的类型有很多，具有不同的功能。基于对车辆的控制程度对软件进行分类，包括控制、信息娱乐软件、高级驾驶辅助软件（ADAS）、自动驾驶软件及安全关键软件。

在传统汽车电子电气系统中，存在两类操作系统。一类运行在汽车动力总成、车身、底盘等汽车内部 ECU 上，被称为 ECU-OS。一类运行在车载信息娱乐上，被称为 IVI-OS。ECU-OS 必须遵循汽车操作系统相关标准，如 OSEK/VDX、AUTOSAR CP 等。其中 OSEK/VDX 对 ECU-OS 内核提出规范，Autosar CP 则对 ECU-OS 进行了标准补充，应用到现有商用汽车操作系统中，以满足多核 ECU 架构需求 [10]。

汽车软件非常复杂，为了简化集成，通过 OSEK 和 AUTOSAR 标准化软件架构并使用可兼容的软件，一些 RTOS 提供了可选的 OSEK OS 适配层，允许将其插入 OSEK OS 兼容系统中。AUTOSAR 使用三层架构，包括基础软件、运行时环境和应用层。AUTOSAR 操作系统规范基于 OSEK/VDX 标准。许多现代 RTOS 提供了 OSEK OS API 封装层，允许将其 OS 集成到 AUTOSAR 环境中。

开发汽车软件既复杂又耗时，但有许多可用的解决方案。市场趋势是使用已有模块构建汽车软件。许多模块已经通过 ISO 26262 的预认证。预认证的软件提供强大且可靠的功能，建议选择基于处理器和编译器组合设计和经过预认证验证的软件，国内外不少商业 RTOS 均有 ISO 26262 ASIL-D 的预认证产品，比如 QNX、SAFERTOS、Sylix OS 和 RT-Thread Smart for Safety。

IVI-OS 对实时性要求并不高，需要为大量复杂应用提供合适的应用程序编程接口、复杂的设备驱动和软件栈，因此，业界通常采用 Linux 内核构建。典型的 IVI-OS 有 Android Automotive OS、华为 YueYing OS 和 QNX 等。

结语

嵌入式实时操作系统是各种电子产品的心脏，应用覆盖消费、工业、军事、航空航天和通信各行各业。嵌入式实时操作系统生命周期从数年到数十年不等，有极为严格的稳定性和可靠性要求，随着微处理器技术的发展而变化，跟着智能系统应用新需求而创新，如今正在步入一个新的历史阶段。国产嵌入式实时操作系统的研究、应用和生态建设将迎来前所未有的发展机遇 [11]。 ■

何小庆

CCF 专业会员、CCF 嵌入式专委会常务委员。麦克泰软件公司创始人，嵌入式系统联谊会秘书长。长期从事嵌入式与物联网技术、产业和教育工作。 allan@esbf.org

（本文责任编委：**周庆国**）

参考文献

[1] 何小庆. 嵌入式操作系统风云录:历史演进与物联网未来[M]. 机械工业出版社, 2017.

[2] Hahm O, Tsiftes N, Petersen H, et al. Operating Systems for Low-End Devices in the Internet of Things: a Survey[J]. *IEEE Internet of Things Journal*, 2016, 3(5):720-734.

[3] Silva M, Cerdeira D, Pinto S, et al. Operating Systems for Internet of Things Low-end Devices: Analysis and Benchmarking[J]. IEEE Internet of Things Journal, 2019, 6(6):10375-10383.
[4] 何小庆. 多处理器电网调度自动化系统设计与实现[D]. 北京;北京航空航天大学研究院, 1991.
[5] 熊谱翔, 全召. RT-Thread Smart 微内核操作系统概述[J]. 单片机与嵌入式系统应用, 2021, 21(3):5.
[6] Jim Cooling. Real-time Operating System Book1-Theory Lindentree Association, 2021
[7] 黄一智, 李仁发. 从安全角度评述汽车操作系统[J]. 单片机与嵌入式系统应用, 2021, 21(3):6..
[8] 郭建川, 殷灿菊. 面向数控机床的可配置混合异构系统体系架构设计[J]. 单片机与嵌入式系统应用, 2022, 22(3):4.
[9] 何小庆. 3种物联网操作系统分析与比较[J]. 微纳电子与智能制造, 2020, 2(1):8.
[10]何小庆. 国产嵌入式操作系统发展与思考[J]. 单片机与嵌入式系统应用, 2019, 19(12):3.
[11]Barry R. Quickly add connectivity using FreeRTOS components Webinar Series/nxp