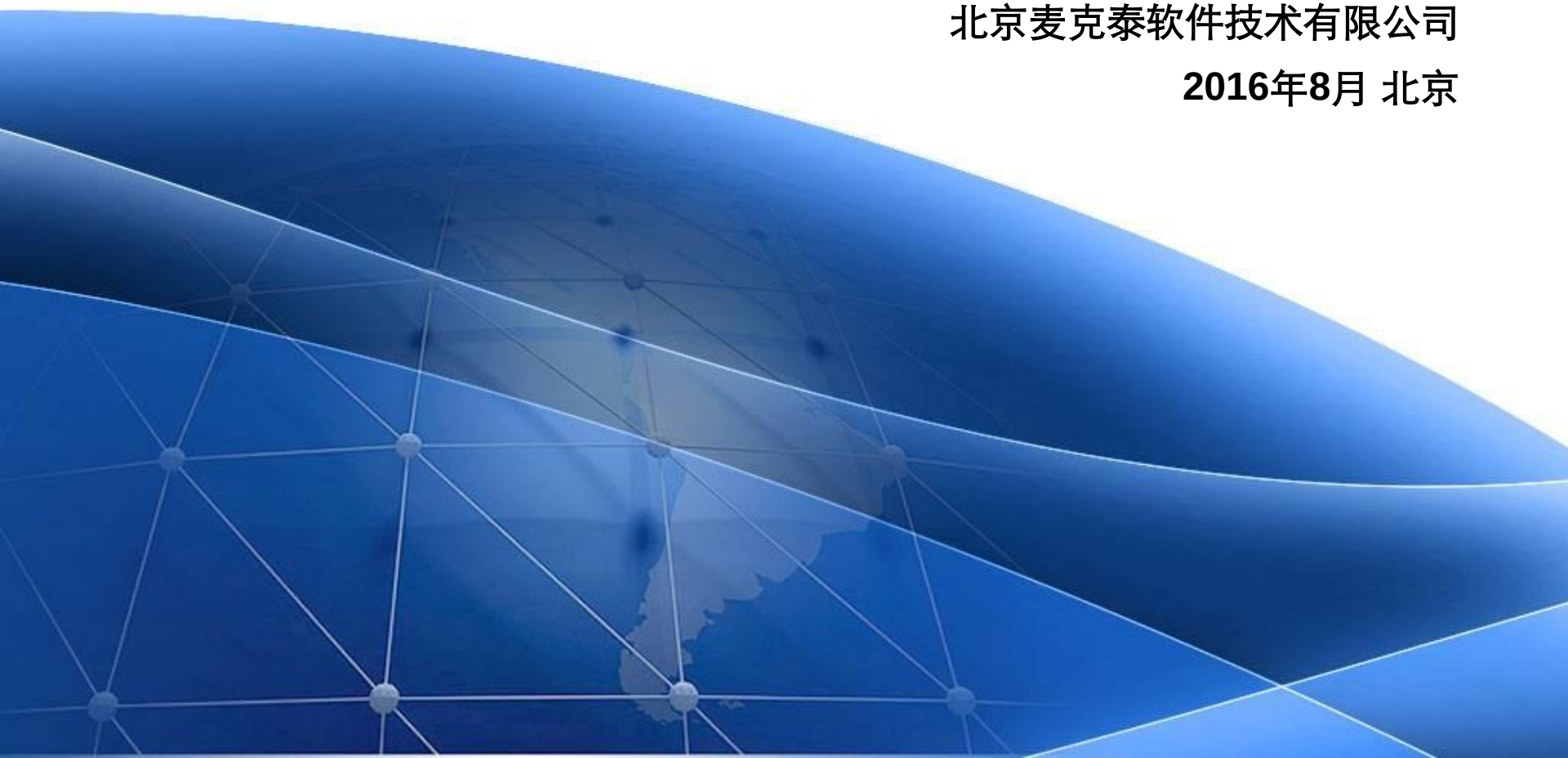
BMR
麦克泰技术

学习和掌握一种RTOS

何小庆

北京麦克泰软件技术有限公司

2016年8月 北京



什么是RTOS?

- R(real) T(time) OS (Operating System)实时多任务操作系统
  - RTOS一种操作系统，属于嵌入式操作系统
  - RTOS 有三大特征 ：确定性，实时性和可靠性
  - 具有并行、异步处理和中断处理能力
  - RTOS种类很多；有商业，DIY和开源
  - 和其他嵌入式OS比较：RTOS 比较小巧和专用。

什么样OS 是RTOS ？

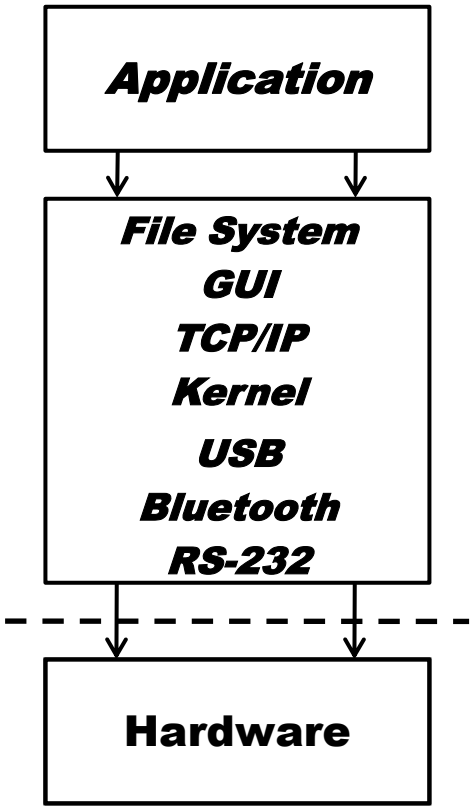
- 那么什么样OS能称为RTOS呢?

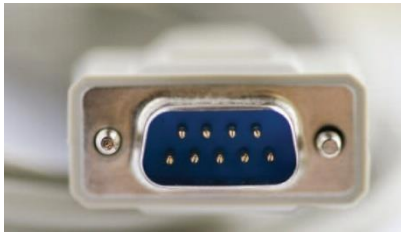
IEEE的实时UNIX分委会认为应具备

- 异步的事件响应
- 确定的切换时间和中断延迟时间优先级中断和调度
- 抢占式调度
- 内存锁定
- 连续文件
- 同步
- ……

RTOS vs. RTOS Kernel

- 操作系统是一系列软件的集合，提供管理资源管理和应用代码服务的能力
- RTOS 已经包含了一系列的软件库（中间件）





RTOS vs. RTOS *Kernel*

- ***The terms operating system and kernel are often used interchangeably***
- ***A kernel is actually a subset of an operating system***
  - ***It can be viewed as the glue that holds the other components together***
- ***FreeRTOS 和 µC/OS-III is a real-time kernel***
- ***Vxwork is RTOS***

RTOS 的历史

- RTOS 已经有超过30年的历史
- 比较著名的商业产品有；（按照时间顺序）
  - VRTX　Microtec　（Mentor 公司收购）
  - pSOS　Wind RiverSystem　wrs.com （WRS 公司收购）
  - OS-9　Microware　Microware.com （Metorworks 收购）
  - SMX　Micro Digtal　www.smxrtos.com
  - VxWorks　Wind RiverSystem　wrs.com （Intel 公司收购）
  - LynxOS　lynuxwork　ynuxworks.com
  - QNX　QNX　www.qnx.com （黑莓收购）
  - CMX　CMX system　www.cmx.com
  - Nucleus　ATI www.mentor.com/esd （Mentor收购）
  - THREADX　Expresslogic　www.rtos.com
  - uC/OS －II/III　Micrium　www.micrium.com
  - INTEGRITY　Gree Hill　www.ghs.com
  - 全球超过100多种，中国几种，更有许多用户自己设计RTOS

开源的RTOS

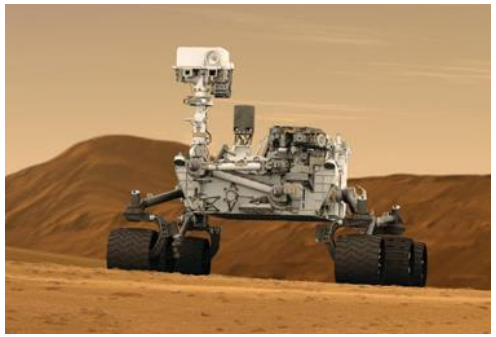
- RTEMS
  - 实时多处理器系统，最早运用在美国防系统
  - 由OAR 公司维护，广泛用在航空航天和军工
- FreeRTOS
  - 比较清晰的表现其目标和专注点在支持8-16-32位 MCU ，但整体缺乏系统性和配套
- eCOS
  - 基于GNU 的RTOS，含TCP/IP和文件系统，Redhad 曾拥有，eCOcentric维护，消费电子应用
- TinyOS
  - 基于传感网络的的RTOS ，有超低功耗管理
- Zephyr
  - Linux基金会宣布了一个微内核项目,由Intel 主导，风河提供技术。

为什么要学习RTOS （today）？

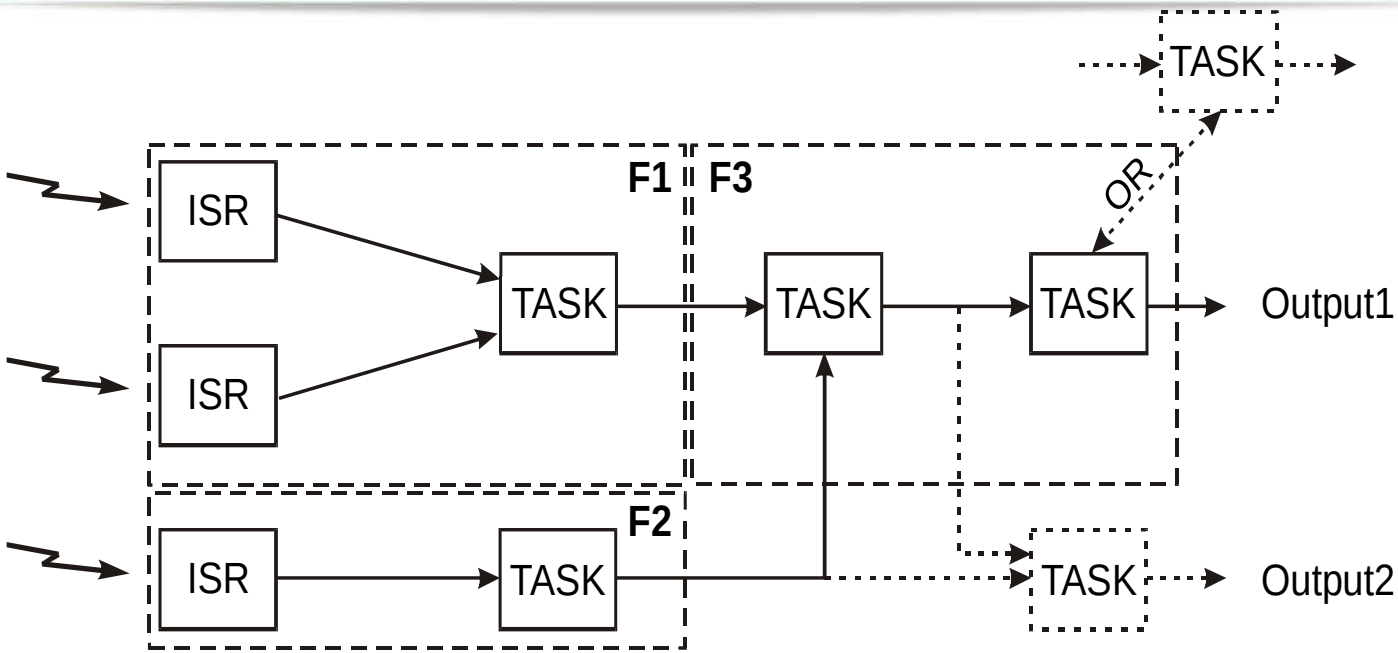
- 物联网的大潮，MCU 迎来一个新的发展机遇：
  - 物联网系统表现为是一个分散的、普适计算的嵌入式系统，大量的MCU替代过去的、运行Linux 的嵌入式处理器(AP)。
  - 物联网应用需要动态功耗效率，以使系统能够在一定的频率范围（50 到 300+ MHz）内，以最低的功耗水平运行。高功耗效率（DMIPS/mW）意味着应用可以在较低的频率下运行，从而降低有效功耗。
  - 物联网系统是连接性系统，需要高度的安全性。目前有多种方法可以在MCU 内部实现这一点。未来MCU支持硬件虚拟化技术非常重要，这可实现应用和数据在多个受信任的分立单元内的隔离。

RTOS 的应用

- 工业控制装置
- 通信设备
- 消费电子产品
- 仪器仪表
- 军事电子设备
- 航空航天系统
- 计算机外设
- 医疗电子产品
- …….



RTOS 的精髓—多任务系统

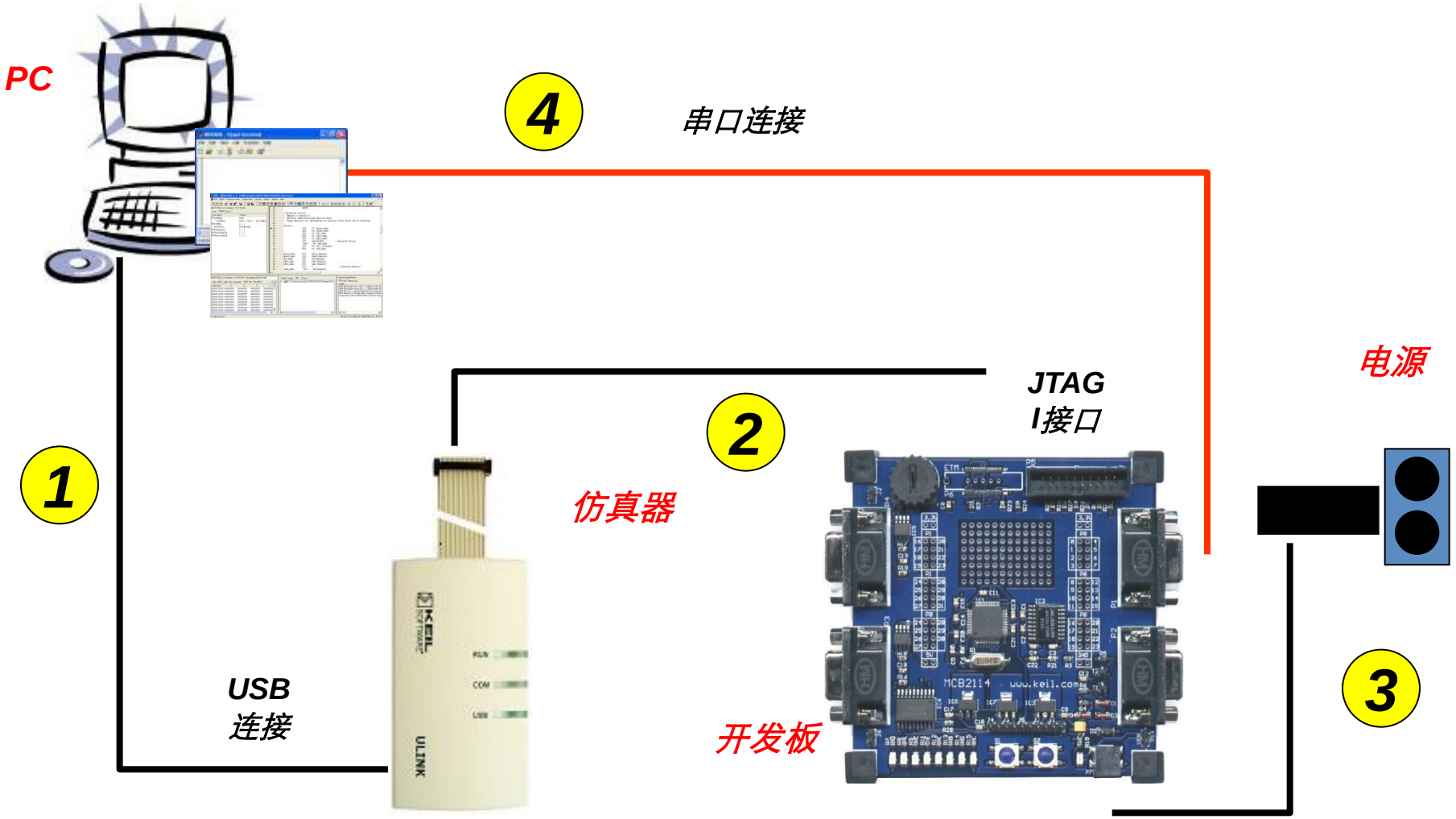


- 任务独立，基于优先级任务调度
- 任务间通信，异步处理
- F1，F2，F3三个功能模块接口清晰
- 易于加入任务

RTOS内核基本功能

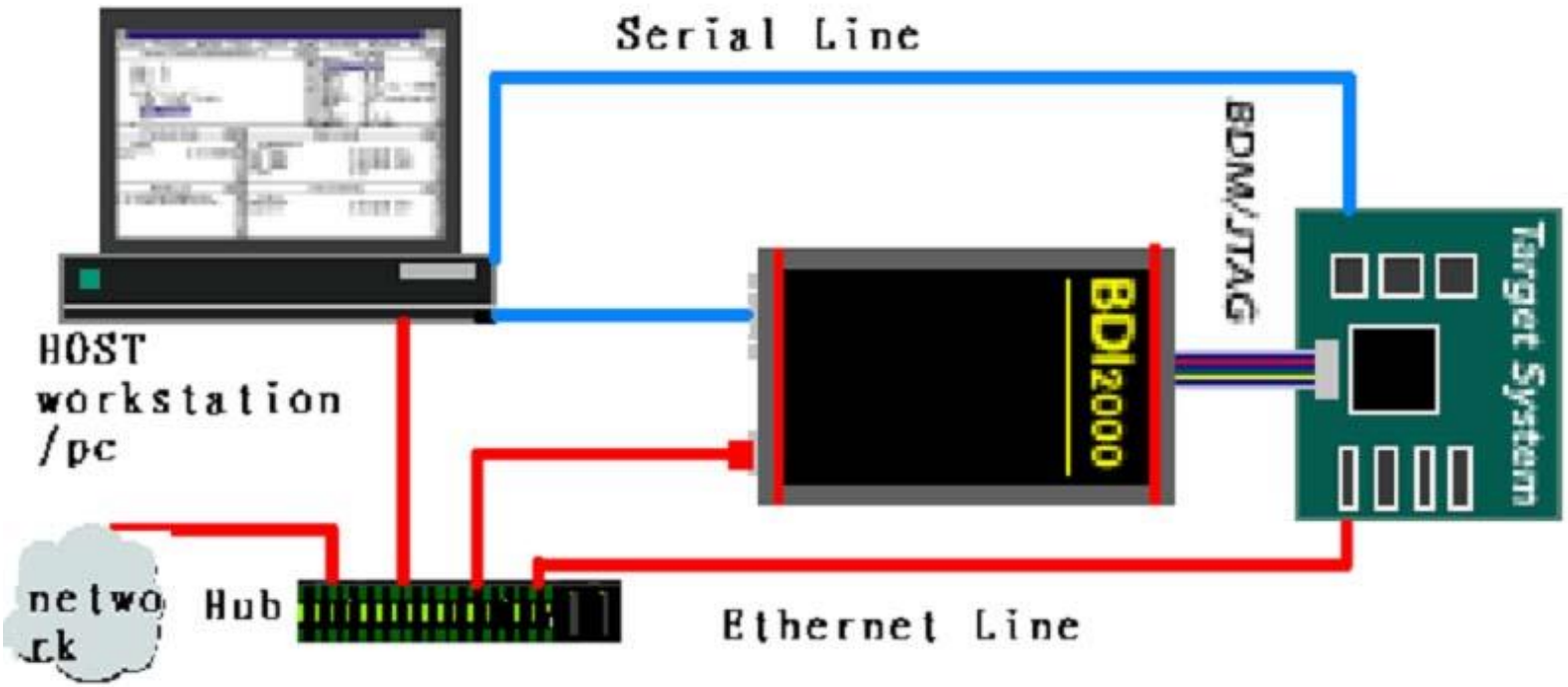
- 系统结构
  - 内核一般很小（几K-几十K）、架构多是单核CPU，个别是微内核
- 存储管理
  - 采用简单、快速的内存分配方案，静态或动态
- 多任务调度
  - 优先级抢占和时间片轮转调度
- 多任务间通信
  - 任务间通信和同步互斥
  - 提供的机制有信号量、邮箱、消息队列、事件标志、互斥

RTOS开发工具（过去）

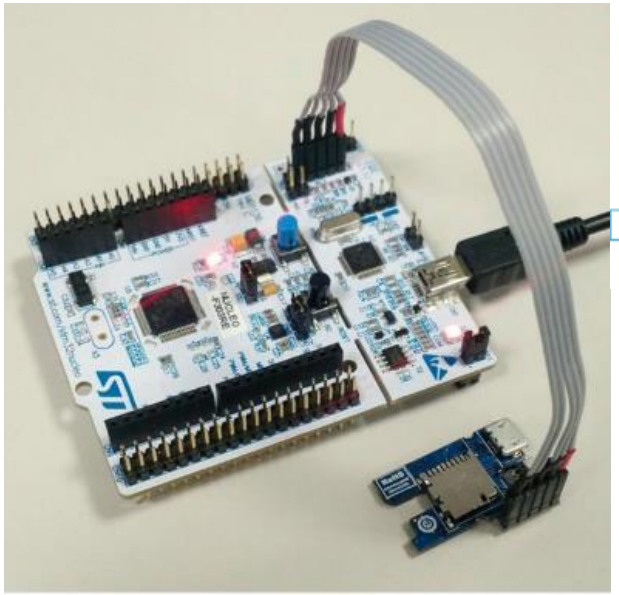


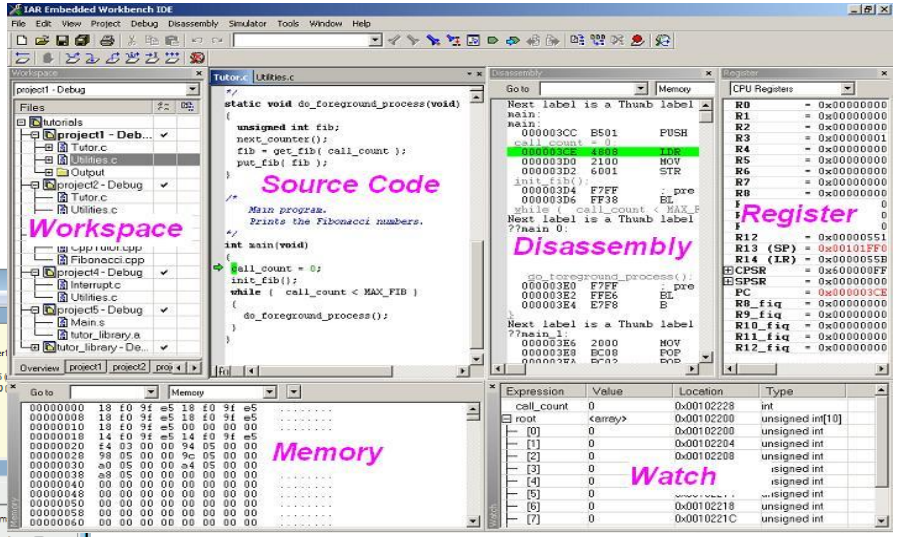
嵌入式Linux 开发工具

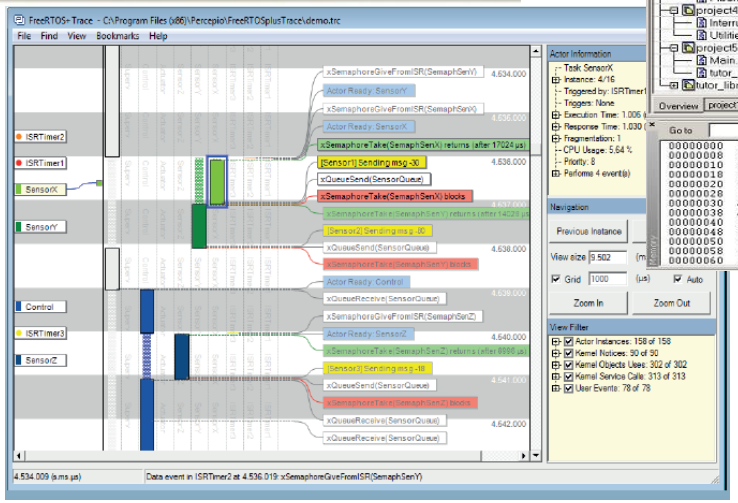
- 目标板、主机（PC 运行虚拟机)、JTAG 和网络。



RTOS 的开发方式（现在）







一台PC
一块开发板
一个USB

BMR 麦克泰技术

学习RTOS 方法

- 任务管理是重点
  - 掌握任务建立，调度，通信和互斥等机制
  - 掌握内存管理方式（静态作为重点）
  - 学习RTOS 内核和硬件相关部分—中断和时钟管理
- 一种简单的驱动编写，比如串口
  - 移植过去是重点，芯片公司和社区会参与多
- RTOS 应用编程接口-缺少标准但要掌握一种
  - POSIX—UNIX 标准
  - uITRON 日本标准
  - OSEK/VDX—汽车电子和交通标准
  - CMSISRTOS-ARM 制定Cortex MCU RTOS接口标准
- RTOS 编程语言和工具
  - C/C++. IAR/KEIL/GCC

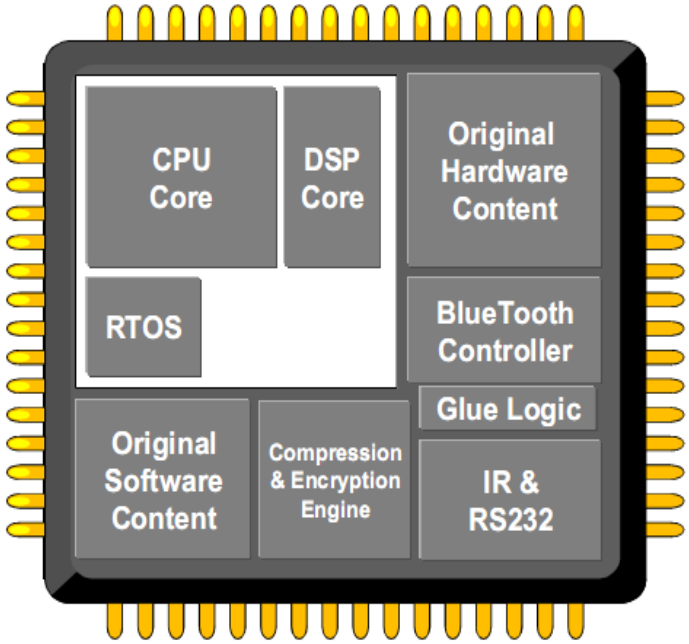
RTOS的组件

- OS 组件越来越多、越来越重要
  - 协议—TCP/IP
    - 开源LWIP 和免费nichelite 相对成熟一点
    - 商业的USB 协议,蓝牙协议和CANOPEN的价格相对要贵
  - 文件系统，Flash NAND， SD/MMC，USB 盘支持优化
  - 图形模块，uC/GUI （emWin）和TouchFX ,纯软件模块对于MCU 消耗大，软硬结合方案，多点触屏和2D/3D图形是未来趋势。
  - 芯片公司开始提供RTOS 的组件(源代码和二进制)
- 大型的RTOS 基本包括了基本组件
  - 比如 Vxwork,QNX 包含TCP/IP,FILE和GUI
- 小型的RTOS 组件是外加的
  - uc/os-III 有uc/GUI,uc/FS,uc/TCP等

应用决定需要那些组件，组件也决定了使用和选择哪种RTOS

RTOS 的发展趋势

- 技术角度
  - 微内核技术将是主流
  - 虚拟化技术将通用OS与RTOS 融合
  - 物联网安全
- 应用角度
  - 标准化和安全认证是趋势
  - AUTOSAR, MISRA C, SAE J2640, LIN,FlexRay,MOST,IEC61508,DO178 B,FDA
- 半导体技术与软件结合
  - SoC/FPGA 与RTOS 的结合



如何选择一个RTOS?

- 首选你是仅仅需要一个kernel 还是需要组件?
  - FreeRTOS 自己只有kernel ，其他第三方，uc/OS-II/II 相当完整
- 你的硬件设计使用的是MCU 还是AP
  - 两者都可以使用RTOS ，但是后者可以支持Linux 或者Android
- 嵌入式和物联网安全的需求:
  - 借助MCU/AP 的 MPU和MMU 可以实现系统内存保护，从而获得安全认证和预认证的安全产品，比如 SaftRTOS
  - 网络安全协议： TLS/SSL/VPN等，只有部分RTOS产品支持
- 芯片和硬件平台的支持
  - 每家的MCU公司的SDK 都会支持1-2 RTOS。
- 价格、技术支持和服务升级
  - 开源？一次性版式，还是unit/per CPU/side
  - 一年和本地化服务…

Comparing microcontroller real-time operating systems
Sergey Kolesnik, Telecard-Pribor Ltd. - December 08, 2013

RTOS的比较：代码尺寸与实时性

代码尺寸比较

	uC/OS-II	uC/Linux
内核	代码：8-24 KB 数据：1- 3 KB	代码：600KB – 1.5MB 数据：几百KB
文件系统	ROM：Min 18 KB RAM：Min 5 kB	ROM：2MB以上
TCP/IP	ROM：75 - 120 KB	ROM：几百KB

注：以上数据均为ARM7下的比较结果

实时性比较

RTOS	任务切换时间
embOS	约4 μs
uC/OS-II	约6 μs
RT-Linux	约50 μs

结 论

1）Linux系统不适合硬实时要求高的应用；

2）商业的嵌入式操作系统的代码要小得多，对用户来说

- 可以选用集成闪存的单片机，价格更便宜、硬件布线更为简化

smx vs. FreeRTOS

- **Ralph moore – SMX 创始人和架构师做了一下测试:**

Task Scheduling

Both kernels implement preemptive task scheduling, which is the best method for real-time embedded systems. Both also support cooperative and time slice scheduling. Task switching times are an important characteristic of RTOS kernels.

FreeRTOS task switching time is fast for the Cortex-M port.

smx uses special algorithms to achieve very fast task switching — up to 290,000 task switches per second on a 400 MHz ARM9. The et2 example in the examples for smx (esmx) package can be used to measure task switching rate on any supported processor.

Semaphores

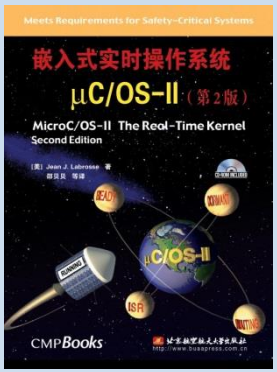
FreeRTOS offers counting and binary event semaphores and resource semaphores. A basic API is provided for them.

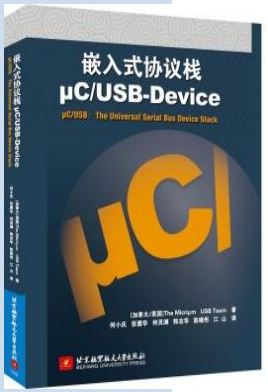
smx adds **threshold** and **gate** semaphores. A threshold semaphore is a counting semaphore that counts signals up to the specified threshold, then resumes the top task and reduces the count by the threshold. This is useful for taking an action every Nth event, waiting for N slave tasks to report back, etc. A gate semaphore is more-or-less the opposite — it resumes all waiting tasks on a single signal. It is like opening the gate of a corral so all the horses can run out.

更多的信息访问 www.smxinfo.com

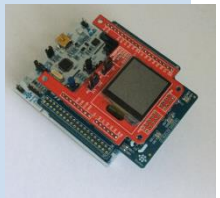
学习资料

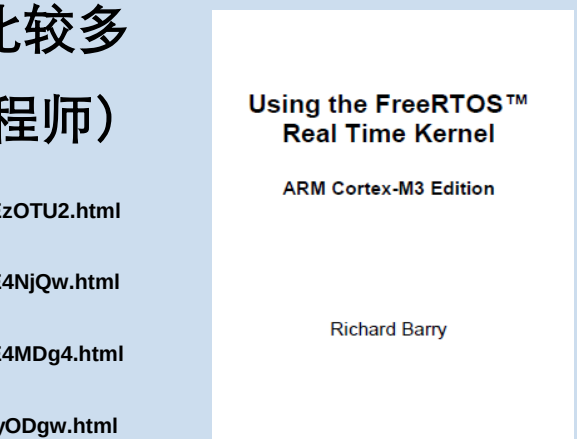
- uC/OS-II 和 III 有非常好中文图书和开发板（官方）

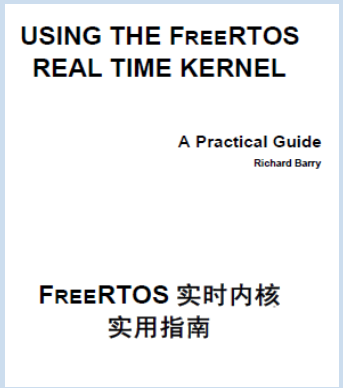




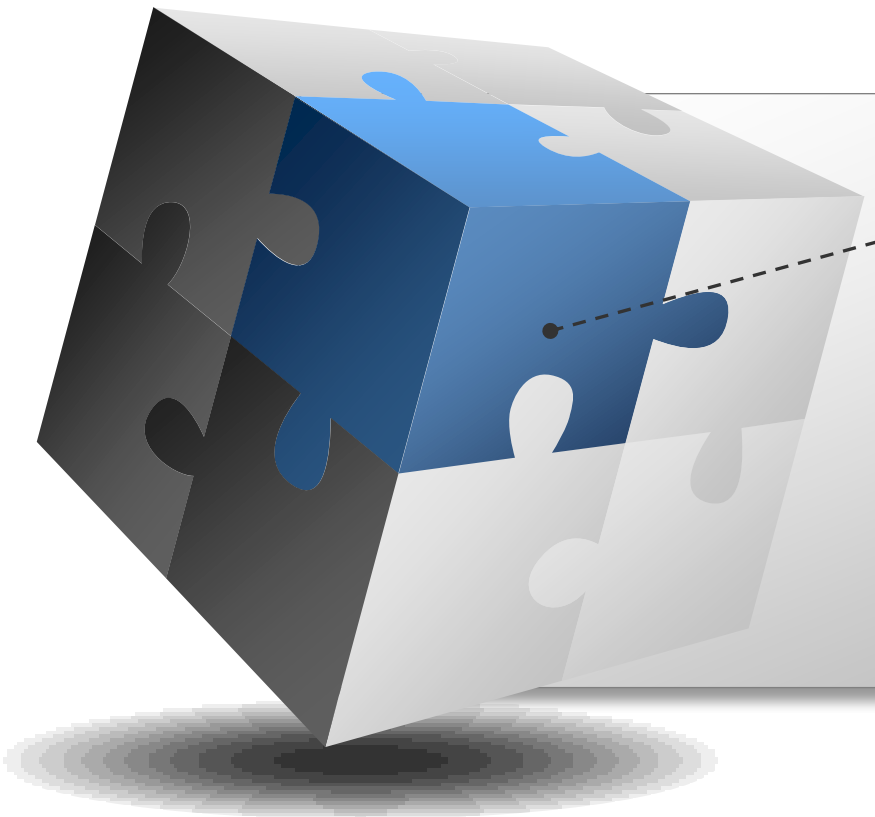
- FreeRTOS 资料比较少，官方的手册需要购买（PDF)
- 芯片公司的BSP 比较多
- 视频（ATMEL 工程师）
  - 1 RTOS 介绍
  - http://v.youku.com/v_show/id_XNTgyMTEzOTU2.html
  - 2 RTOS 特性和API
  - http://v.youku.com/v_show/id_XNTgyMTE4NjQw.html
  - 3 FreeRTOS使用
  - http://v.youku.com/v_show/id_XNTgyMTE4MDg4.html
  - 4 深入了解FreeRTOS
  - http://v.youku.com/v_show/id_XNTgyMTIyODgw.html







感谢您的关注，欢迎交流！



何小庆 www.hexiaoqing.net

公司网址：www.bmrtech.com

邮箱：allan.he@bmrtech.com

北京：010-62975900

上海：021-62127690

深圳：0755-82977971

版本信息：2010年 第一次撰写 v1.0

2016年第二次撰写 v2.0

BMR 麦克泰技术