实时线程操作系统 0.3.0 概述

-- 面向 TI 流明 LM3S 微控制器版本

实时线程操作系统是一款面向实时领域的操作系统，这个和通常的通用操作系统有着很大的不同。通用操作系统通常面对的是日常应用，例如打开浏览器上网，播放音乐，采用字处理软件编辑文档。

在古老的时代，通用操作系统一般只能处理一件事，而随着计算机技术日新月异的发展，硬件更新换代，上 GHz 主频、上 GB 内存、多核技术走进普通人的生活中，通用操作系统也就朝着并行化计算的方向发展。通用操作系统更多讲究的是，对这并行事务处理的公平性上，例如一个个民工在不同的窗口进行排队购买火车票，好的调度系统能够保证每个民工的公平性。

实时系统和这种通用系统有很大的差别。实时系统指的是，当外界有系统关注的相应事件发生时，系统能够在指定的时间内（deadline）进行正确的响应。用于实时系统的操作系统就叫做实时操作系统。从系统的定义也可以看得出，实时操作系统和通用操作系统在事务的处理上有明显的区别，实时操作系统有非常强的针对性，对相应的事件力求做到这固定的时间内进行响应；而通用操作系统则需要努力地做到各个事务的公平性（某些系统也会非常注意数据的吞吐量，例如网络服务器）。

实时线程操作系统（英文名 RT-Thread）面向的正是这么一类的实时系统，因为其小型的特点也可以看成是一个嵌入式操作系统（嵌入式系统一般是针对一些专有目的而存在，比较吝啬于成本。而基于专有目的的特点，也注定了嵌入式系统或多或少的具有一些实时性的特点）。这种系统可以用于自动售票机，税控机，移动通信设备，mp3/mp4 等便携式音乐设备，飞行器控制，车体导航控制，打印机，复印机，各类监控设备，路由器，ADSL，机顶盒等网络设备，医疗设备等等。

RT-Thread/LM3S 0.3.0 rc1 版本

TI 流明 LM3S 系列芯片是基于 ARM Cortex M3 v7 构架的 32 位芯片，其中 LM3S S6000，S8000，S9000 系列芯片携带网络功能。RT-Thread 的标准内核可以运行在除 LM3S S100 系列外的所有系列芯片上。这次 RT-Thread 针对于 LM3S 进行移植验证并经过压力测试的是 LM3S6918 芯片，芯片携带 64K 片内静态内存，256K 闪存，频率是 50MHz。在这个平台上，RT-Thread 支持的特性包括：

⌦ 完善的实时核心

- 面向对象方式的实时核心（但依然保留了 C 语言的优雅、小巧风格）；
- 默认 32 线程优先级的全抢占式实时内核（亦可配置成 256 线程优先级）；相同优先级线程时间片轮转调度；
- 相同优先级线程实施时间片可配置的分时时间片轮转调度；
- 线程间同步机制：信号量和防止优先级翻转的互斥锁；
- 完善高效的线程间通信机制，包括邮箱，消息队列和事件；
- 支持线程挂起和唤醒的固定内存块管理及线程安全的动态内存堆管理；
- 向上层提供基于名字的统一接口设备驱动模型；

⌦ **FinSH shell 命令行**

- 命令即 C 代码的命令行方式；
- 直接在命令行中调用系统内核函数；
- 直接在命令行中访问系统全局变量；
- 历史记录及命令自动补全；

⌦ **面向小型设备的虚拟文件系统**

- 向上层应用提供 POSIX 风格的 API 接口；
- 支持多种具体文件系统实现；
- LM3S 分支内置 SD 卡驱动程序；

⌦ **LwIP 轻型 TCP/IP 协议栈**

- 标准的 BSD Socket 接口；
- IP、ICMP、UDP、TCP 标准协议支持；
- DNS，DHCP，PPP 协议支持；
- TFTP、HTTP、FTP 应用协议支持（见 netutil 组件）；
- LM3S 分支内置以太网驱动；

⌦ **开发环境支持：**

- GNU GCC (scons 做为构建工具)
- Keil MDK

以上是 RT-Thread/LM3S 0.3.0 的特性，这些特性在 RT-Thread 0.3.x 分支中不会有大的改变。

技术指标及优势

看完 RT-Thread 的特性后，看看一些其他有吸引力的地方。

首先是体积。看网上有网友说，RT-Thread 是否是基于 Linux，或者直接使用 Linux，这里不得不说，Linux 并不是任何事情都做得到，它做不到在数 KByte 的内存占用上依然能够非常好的运行，而这类设备非常多。例如 LM3S 这类芯片，本身只有大约 64K 或更少的片内静态内存，另外就是闪存（通常在 128K - 512K 之间）。外扩内存基本上不太可能，这类芯片是完全的成本敏感型芯片，硬件决定了它已经不能外扩内存（LM3S 最新款的已经支持能够外扩内存了，不过手上还没拿到）。

看几个体积指标：

RT-Thread 标准 Kernel（标准 Kernel 指得是没经过剪裁的内核）：

9.5K 只读数据和执行代码占用，1.5K 内存占用（通常只读数据和执行代码放置在闪存中）

包括上面说的完整组件，即标准 Kernel，finsh shell，文件系统，网络协议栈：

80K 只读数据和执行代码占用，13.5K 内存占用，当运行时，会有 5K 左右的动态内存占用。即当系统运行时，大约剩余 45K 内存给用户使用。

和 LM3S 提供的无操作系统 LwIP，FatFS 文件系统比较：

120K 只读数据和执行代码占用，35K 内存占用。即当系统运行时，大约剩余 20K 内存给用户使用。

再看看针对网络的一些性能指标

对比情况采用了相同的 netio 测试得到的数据统计

RT-Thread/LM3S

NETIO - Network Throughput Benchmark, Version 1.26

(C) 1997-2005 Kai Uwe Rommel

TCP connection established.

Packet size 1k bytes: 704 KByte/s Tx, 5131 Byte/s Rx.
Packet size 2k bytes: 704 KByte/s Tx, 1950 KByte/s Rx.
Packet size 4k bytes: 704 KByte/s Tx, 2197 KByte/s Rx.
Packet size 8k bytes: 704 KByte/s Tx, 2200 KByte/s Rx.
Packet size 16k bytes: 706 KByte/s Tx, 2196 KByte/s Rx.
Packet size 32k bytes: 709 KByte/s Tx, 2136 KByte/s Rx.
Done.

TI/无操作系统情况下的 LwIP
NETIO - Network Throughput Benchmark, Version 1.26
(C) 1997-2005 Kai Uwe Rommel
TCP connection established.
Packet size 1k bytes: 870 KByte/s Tx, 5187 Byte/s Rx.
Packet size 2k bytes: 870 KByte/s Tx, 2463 KByte/s Rx.
Packet size 4k bytes: 870 KByte/s Tx, 3322 KByte/s Rx.
Packet size 8k bytes: 870 KByte/s Tx, 3239 KByte/s Rx.

（上面的是 PC 端 NETIO 输出的结果，Tx 对应 LM3S 开发板上的接收，Rx 对应 LM3S 开发板上的发送）

从上面可以看出，在大数据块发送时，RT-Thread/LM3S 的移植会有一些损耗，但总的来说损耗不算太大，特别是报文在 1k - 2k 范围时数据相差不大。如何看待这个差异：通常没有操作系统时，系统能够更专注的做一件事。而有操作系统的情况下，它能够额外的做一些事务，这些事务在进行切换时，会产生一定的系统资源开销。总的来说，体积上的改善一定程度上弥补了性能的差距（例如 RT-Thread/LM3S 能够支持 16k、32k bytes 大小的网络包发送接收，而无操作系统的 LwIP 则不能）。

最后看看一些实时性能指标

这组数据是与著名的开源实时操作系统 ecos 的比较。测试代码完全相同，硬件平台相同（采用的是 PXA310），编译器相同（GNU GCC），编译参数相同：

基本任务测试 RTT/ecos 1.40 倍
协作调度测试 RTT/ecos 1.20 倍
抢占调度测试 RTT/ecos 1.33 倍
同步处理测试 RTT/ecos 1.86 倍
内存分配测试 RTT/ecos 2.50 倍

当一个实时操作系统能够做到稳定运行、性能比拟普遍运行的系统有一定优势、后续能够稳定发展时，那么她将是一款能够得到普遍使用的系统，而事实也正是如此：在中国，已经有十来家公司采用 RT-Thread 运用于他们的产品中，而后续打算在产品中采用 RT-Thread 的公司还有更多。

许可证

作为一套基础组件，就例如电脑中的中文输入法一样，它不应该收费，因此它能够免费的使用于商业产品中（0.3.x 系统仅需要在我们这边进行产品信息备案，更换 GPLv2 许可证为商业许可证！0.4.x 将更换产品许可证为 BSD 或 Apache 开源许可证）。