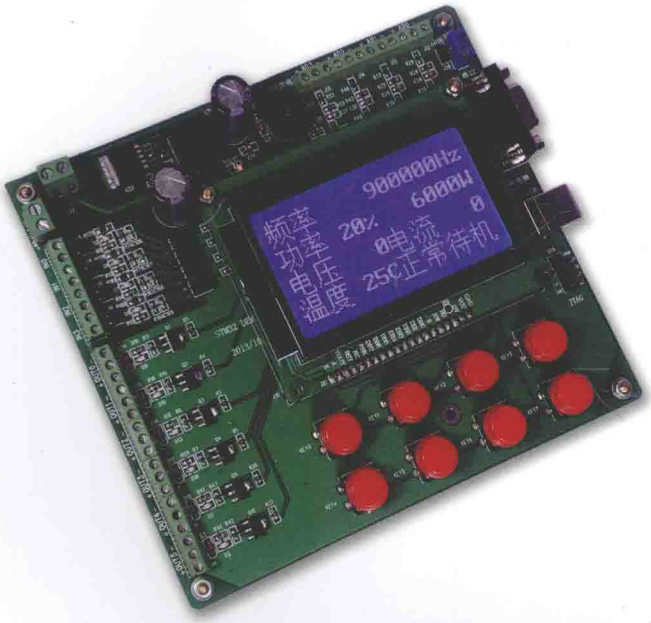
HZ BOOKS
华章科技

本书脱胎于开源项目“msOS”，是对该项目开发方法和开发套路的系统总结。
嵌入式微系统基于通用程序架构的思想平台，总结了MCU级嵌入式编程方法。作者从嵌入式软件工程师、产品经理等多个维度，围绕“msOS”平台，系统揭示了嵌入式产品创业必备的技能和知识。

电子与嵌入式系统设计丛书

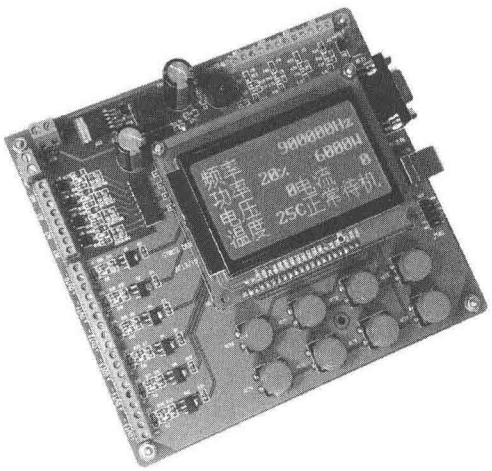


嵌入式微系统

王绍伟 郑德智 吴玉勇 编著

机械工业出版社
China Machine Press

电子与嵌入式系统
设计丛书



嵌入式微系统

王绍伟 郑德智 吴玉勇 编著

机械工业出版社
China Machine Press

图书在版编目（CIP）数据

嵌入式微系统 / 王绍伟，郑德智，吴玉勇编著 . —北京：机械工业出版社，2016.6
（电子与嵌入式系统设计丛书）

ISBN 978-7-111-53912-4

I. 嵌… II. ①王… ②郑… ③吴… III. 微型计算机 - 系统设计 IV. TP360.21

中国版本图书馆 CIP 数据核字（2016）第 118400 号

嵌入式微系统

出版发行：机械工业出版社（北京市西城区百万庄大街 22 号 邮政编码：100037）
责任编辑：陈佳媛　　责任校对：殷　虹
印　　刷：三河市宏图印务有限公司　　版　　次：2016 年 7 月第 1 版第 1 次印刷
开　　本：186mm × 240mm　1/16　　印　　张：14.5
书　　号：ISBN 978-7-111-53912-4　　定　　价：59.00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换
客服热线：（010）88379426　88361066　　投稿热线：（010）88379604
购书热线：（010）68326294　88379649　68995259　　读者信箱：hzit@hzbook.com

自　序

长期以来，低端嵌入式行业（如MCU51、ARM7、Cortex-M3）因为受到CPU性能、ROM和RAM容量及其他因素的制约，软件无法做得太大，加之低端需求也不需要很大，所以开发人员往往设计随意，规划不强。此外，由于低端嵌入式系统需求多样，没有一家专门的公司或机构为其设计一套通用的软件架构，大家各自为政，甚至是一个公司的几个嵌入式人员所编写的代码都完全不同，而新来的嵌入式人员往往因无法读懂前人的代码而推翻其成果重做设计，导致这种重复无用劳动的原因是没有一个软件架构标准。

我也一直被这种无序的现状困扰着，在开发高频感应加热电源的时候，很希望找到一套比较简单易用的嵌入式软件架构帮助自己完成设计，尤其是GUI部分，可惜事与愿违。为了整个公司的代码统一性、与PC编程接轨的可行性及操作维护的长效性，必须找到一套标准和公认的模板，同时把嵌入式行业出现的优秀元素（如RTOS、GUI、面向对象设计、分层设计等思想）引入这个架构中，通过合理的组织形成完整的系统。

这个系统不能复杂，必须要简单易用，因为嵌入式面对的应用场合千千万万，各不相同，无法提供所有的需求，即使提供了，也会因ROM容量有限、CPU性能有限而受限，所以必须给出一个可让使用者容易读懂，且自己容易修改、增删的系统，每个功能提供一两个实例，用户根据实际项目的情况进行修改、增删。

虽然我曾做过一段时间软件，但长期负责硬件，后来创业，管理公司，可以说软件对我来说是弱项，正因为是弱项，让自己不拘泥于软件技术本身，而是用硬件尤其是企业管理者的思维来看待软件，从需求入手建立自己想要的软件系统，即嵌入式微系统（msOS）。msOS成型后，我认识到这就是我日常的企业管理思维：分层设计，各模块独立运作，实现高内敛低耦合思想。这些思想都是日常管理中的基本常识，所以msOS文档的描述将更多的是基于常识的讲解，淡化一些专用名词、专用术语，卸下包袱，让常识自然融入其中。

msOS开发完成后，获得了同事的普遍认可，他们认为msOS用代码方式总结了嵌入式行业多年来的发展成果。现把它编写成书，尤其是把它的需求、历史写出来，让更多的嵌入式人员从中找到自己想要的东西，而不仅仅是代码。

学习 msOS，目的是应用，尽可能地降低学习费用，把精力放在快速开发产品上。

最后感谢各位合作伙伴，在开发 msPLC/msOS 的 4 年多时间里，你们不仅支撑了公司的发展，还积极参与改进 msPLC/msOS 的各种缺陷，让它更加完美。

为 msPLC/msOS 做出重要贡献的人员如下。

妻子刘颖：接管家庭事务，协助公司事务，并完善部分代码，参与部分章节写作、修订。

周庆国教授：在学业和工作上给予我很大帮助，尤其是在人生的几次重要转折点上。

郑德智、吴玉勇：接手我原来的工作，并全力支持 msPLC/msOS 开发。

孔海文、朱志惠：提供各种工业产品参考，设计上给予指导。

陈永强、皮云仙、苏鹏、田飞峰、高茂光：积极协助开发，并多次提出改进意见。

易艳辉、彭娟、李小龙：基于 msPLC/msOS 开发各种设备，并报告一些 msPLC/msOS 的缺陷。

感谢 msPLC/msOS QQ 群的各位网友对 msPLC/msOS 的支持和改进。

王绍伟

2016 年 4 月

引　言

2001年大学毕业后，经兰州大学周庆国教授介绍我进入中科院半导体所工作，师从祝宁华教授、谢亮研究员，第一次真正接触到了MCU51的C语言开发（大学时期接触过单片机汇编）。听谢研究员讲解C语言的优点，尤其是可以写成很多子函数，形成一个个库，便于今后复用，我就想应该设计一套属于自己的C语言库。研究所里的学术氛围很好，对于很多技术问题的讨论都很开放，大家也很看重我的观点，这让我的自信心得到了极大的提升。

在研究所工作了近半年之后，在周庆国教授的介绍下我来到了深圳市经纬科技有限公司，跟周荣洁博士做手机设计，因为周博士负责公司软件开发，于是我就从硬件设计人员转变成了一名软件设计人员，主要负责驱动，比如LCD、MIDI音频。那个时候刚进入经纬，C语言没有真正写过，却直接面对大规模的手机软件，难度可想而知，是周博士手把手教我如何看代码，如何理解每一句话的含义，尤其是字库显示部分代码分析，这个场景现在还历历在目。

因为有周博士的帮助，加上自身对硬件的理解，我很快上手并获得了公司的认可，这样日子就轻松很多，有了空闲时间。因为掌握了C语言，加上之前接触过MCU51，这时我想把以前建立一套软件库的想法实现，于是在2002年11月左右从周立功公司购买了一套MCU51开发板，型号为DP51，按照参考示例逐个练习，这就是子函数的应用。学了半个月之后，觉得这些太简单了，管理也很麻烦，因为一个个例子、子函数各自独立，无法合在一起，还需要专门给这些子函数建目录，时间一长可能会丢失。因为做过了手机软件，其技术层次远远高于单片机，尤其是平台思想，于是我想引入平台概念，把可能会用到的例子、子函数都在这个平台下管理起来，形成一个有机的整体，这也就是“实用单片机系统”（McuSystem，简称MS）思想的来源。

实现的难度往往远大于想法。虽然那时身边有一群软件高手，但他们都不做单片机开发，所以只能靠我自己独立完成MS。如何构建这个平台，把这些例子衔接起来，则基于大学时期学的VB开发，因为它是典型的消息驱动方式，简单、易用，于是我设计了一套消息机制作为核心。在做手机驱动的时候，经常用到系统节拍、软件定时器、串口调试。系统节

拍固定时间有一个中断回调，可以实现按键、数码管等例行扫描程序。软件定时器可以用在一些功能超时处理方面，比如进入某个菜单超时退出、按键音、闹钟或者一些游戏、动画，尤其是可以实现一定时间间隔的自我循环，类似一种伪任务。串口调试是因为手机平台过于庞大，需要实时运行，运行无法通过仿真器调试，所以一般用串口来打印信息了解运行情况。后来随着 FlashROM 技术的成熟，尤其是 STC 的单片机，支持在线编程 ISP，根本不需要用仿真器，那么串口调试就显得很有价值。依靠以上想法，经过两三个月的开发，MS 第一版本基本上成型。

MS 出来的时候，恰好祝教授有测量激光器特性的项目要做，于是基于 MS 做了两个项目，也进一步完善了细节。过了一年在周立功深圳分公司认识了一位客户，他需要开发基于 CAN 通信的轮胎硫化设备，但因为资金问题，只让我帮他完成框架，具体的他自己来实现。我基于 MS 完成框架后让他自己设计，本来以为他不可能实现，最后还会来找我，没想到只懂一点点 C 语言的他竟然自己搞定了，这让我意识到 MS 的价值：简单、易用。后来我把 MS 分享给身边人，尤其是当时还在周立功深圳分公司的陈茂华先生，他对 MS 大加赞赏，认为 MS 简单易用，非常适合入门，很有价值，有了他的认同让我感觉到需要进一步完善，并且通过网络分享给大家，2003 年我将 MS 放在 21IC 电子网站的 FTP 上共享。

2006 年来到深圳市华禹高科技有限公司，因为是研发负责人，需要招聘，所以经常在 21IC 出没。过了一年“程序匠人”搞了一个侃单片机版面的竞选版主活动，我一时兴起，也去参加，竞选的作品是 MS3（当时升级为第三个版本）。当时，为了提高 MS3 的代码质量，还专门让妻子刘颖优化代码（她是计算机系研究生，曾在华为手机部门工作，编程基础远比我强）。竞选活动很是激烈，参与者众多，大家对 MS 印象较深，感觉不错，受到多数人的肯定与支持，但 MS3 跟 21IC 的一个元老级网友“农民讲习所”的一个“通用处理程序”不相上下，双方都有较多的支持者，于是免不了一番激烈的竞争，这进一步推动了 MS 的扩散。客观地讲，“通用处理程序”跟 MS 理念有一定的相似性，但出发点完全不同。“通用处理程序”的基本思想来自于 RTOS 方面，属于主流技术派。相反，MS 源自手机的平台思想，是一个开发平台架构，并且因为 MS 中很少用高难度的技术，甚至都没用指针，编写的代码都很简单、易用，架构清晰明了，让人一目了然，所以很适合入门的嵌入式群体，也获得他们的最大支持。所以在后来的推广中，MS 胜出。陈永强（一位后来跟我们一起创业的股东）在看到 MS3 后，很惊讶原来 MCU51 还可以这么写程序，于是就这么入了伙。

有了这些基础后，我经常收到询问 MS3 问题的邮件，也有人加我 QQ 问一些问题，在这个过程中，有个别网友很好奇我是如何把这些元素融合起来的，他们在溯源我当时的想法是什么，但可惜的是，刨根问底的只有很个别的几个网友，而大部分网友只是看代码而已。在

询问过程中，我发现大部分网友做的项目过于简单，没有接触复杂的需求，尤其是软件定时器，他们无法理解它是干什么用的，为什么要加入这个功能，而这些问题导致今后的文档注重讲解功能的来源及我当时的想法。后来，我经常把 MS3 作为公司内部软件招聘和培训之用，一些单片机项目也都以它为基础开发，可以说整个公司都熟悉 MS3，这带来了较大的沟通便利性。

作为第一版本，MS1 还很不成熟，尤其是那个时候，代码写得不多，所以很多细节处理显得幼稚，甚至包括编写规范性，但其思想已经体现。MS2 把一些没必要的东西都去掉了，甚至都不包含指针，只需要一颗 MCU51 即可，可以在 Keil 仿真器下直接运行。通过 UART 模拟仿真，最好从 MS2 入手。MS3 是比较成熟的版本，尤其是 3.20 版本，因为竞选版主打下的基础，所以客户群体比较广泛，实际使用最多，2011 年高频感应加热电源项目初期就基于 MCU51，用的就是 MS3。

随着高频感应加热电源的深入，涉及高速信号例行处理（10k/s），这个时候被迫放弃 MCU51 而迁移到 Cortex M3 平台，于是基于 NXP 的 LPC1343 把 MS3 升级为 MS4，除了保留原来的功能外，主要引入了函数指针做界面设计，对于简单的项目来说，比较容易实现。此外，根据项目需求把系统节拍按需求细分为 10 k/s、1 k/s、100/s、10/s，紧急响应采用中断，I/O 状态检测、高速执行用 10 k/s，水压、数码管扫描显示之类的用 1 k/s，按键扫描用 100/s，LCD 屏、数码管数据显示用 10/s。这样处理可以很好地把低速节拍分散到高速系统节拍中，不堆积在一个节拍中执行，避免单个系统节拍占用时间过长的问题。外部采样检测，一般不建议用中断，尽可能用扫描方式的原因是：一是没有这么多中断口；二是中断容易因为毛刺，导致多次中断无法识别；三是可以采用滤波处理提高抗干扰能力。高频感应加热电源曾经一度想上 RTOS 实现实时响应，但遭到大家反对，因为那个时候我们对 RTOS 都不太熟悉，虽然有所了解，但没有真正深入，尤其是实时这个概念都不是很清晰，所以认为深入分析透彻项目需求才是出路，后来准确分析项目需求后，提出了系统节拍的速度分级，很好地解决了实时性问题，系统稳定可靠。

MS4 基本上没对外宣传，只是在博客中发布过，主要是因为本人精力有限。随着高频感应加热电源的深入，需求越来越复杂，而这些需求大部分都来自界面设计，原有的架构体系不足以支撑复杂的界面，写着写着代码自己都晕，不仅要设计业务逻辑，还要花很多时间设计界面，而这个界面设计又干扰了正常的业务逻辑设计，处理得不好甚至会导致系统混乱，而这不是我想看到的，于是就想要改变了。

对于小项目来说，界面设计不复杂，但对于稍微大一些的项目，界面一多，尤其是一些动态数据需要显示，参数需要设置，界面设计就变得相当复杂。此外，黑白 LCD 驱动一般

总线速度不高，显示刷新还不能太快，不然动态数据无法看清，这些导致界面显示必须要与业务逻辑独立分离。复杂的界面需求没有一个简单、统一的标准开发模板，导致在大部分嵌入式项目中，界面部分的代码最难被别人看懂，传承性差。虽然现在有 μC/GUI 等标准化的界面设计库，但它适合于彩色点阵屏，消耗资源较大，并且系统过于复杂。而很多工业类嵌入式项目一般都用黑白屏，黑白屏具有简单可靠、开发难度低、对处理器要求也低的特点，毕竟工业控制的重点不是界面的色彩，而是系统本身的性能。高频感应加热电源就采用 128×64 点阵的黑白字库屏，支持 8×4 个汉字或者 16×4 个字母。

因为界面的这些需求，导致项目越做越复杂，时间长了自己都迷糊，而现实又逼着自己抽身离开高频感应电源项目组去开拓新的业务：机械自动化的可编程逻辑控制器（PLC），所以必须要让继承者掌握这些软件，而继承者基本没有软件基础，虽然在我的指点下，加上他对高频感应电源本身的熟悉，能看懂一些业务逻辑并且能稍做修改，但对于界面设计基本上一头雾水。这些都促使我不得不重新设计全新的系统，解决界面问题，再加上 PLC 的软件需求，想把两者的需求统一起来一并解决。

MS4 是基于 NXP 的 LPC1343 芯片，这是因为我有朋友出售 NXP，容易采购，但 NXP 的通用性、资料性毕竟没有 ST 强，客户群体也没有 ST 广泛，所以在设计新系统的时候，就考虑基于 ST 最常用的 STM32F103 来设计。基于驱动库把 MS4 移植上来，再增加一些高频感应电源上用到的功能。升级后的版本为 MS5，属于过渡版本，在 2012 年年底发布，也没推广。

为了解决业务逻辑与菜单界面的分离问题，2012 年年底着手学习 RTOS，参考了多家 RTOS，最终选择客户群体最为广泛的 μC/OS-II，因为有资料可以参考。很多网友向我抱怨 μC/OS 看都看不懂，关于应用，心里抵触感很强。经过一番分析，我发现他们主要是被过多的宏定义、数据结构、指针、不常用的附属功能所困扰，严重地影响了对程序的阅读，他们甚至不知道哪些函数是重点。对我来说，这个问题同样头疼，导致思维不清晰，于是我从精简入手，先把跟系统关系不大的、一些不常用的附属功能去掉，再去掉一些没用的宏，甚至把任务数从 64 个变成了 8 个，去掉了复杂的优先级计算，去掉了复杂的链表结构而改用数组，去掉了很多不常用的函数，如删除任务、删除事件之类，这样一个 μC/OS 只剩下最核心的内核和必要的功能函数，只有 3 个文件，看起来简单易懂，语法非常简单。之后把 MS5 与初步精简化的 μC/OS 整合为一个版本，MS5 作为 μC/OS 的最低优先级任务，因为这个结合了 MS，所以就叫 msOS，但考虑到种种原因，尤其是有把两个东西强行合在一起的感觉，所以基本上没有推广。

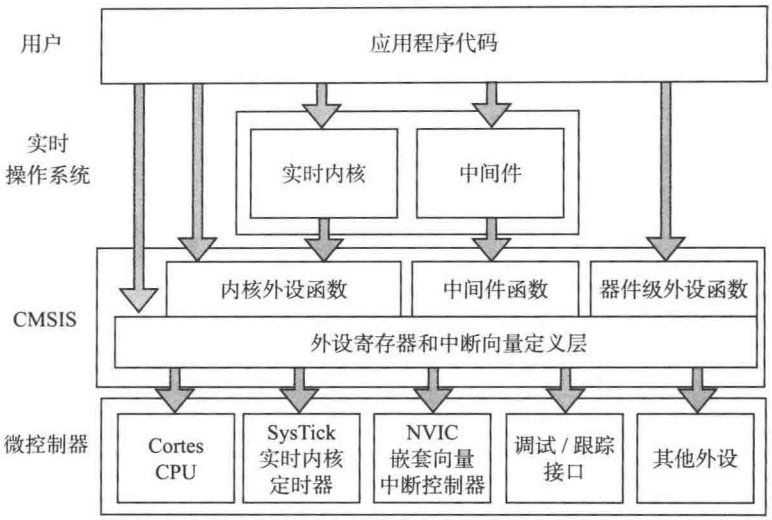
做到这儿，自己都迷惑了，如何把 MS5 的优点与 μC/OS 完美地结合起来，既解决菜单

界面与业务逻辑的分离问题，又保留简单易用的前后台系统？但第一版的 msOS 这种生硬的结合很不协调，没有解决分离问题，还需要深入分析菜单界面编程到底是怎么回事，于是跟雨滴科技公司的几位软件负责人沟通，我把我的需求告诉了苏鹏，他长期做 Linux，曾师从 RT-Linux/GPL 维护人 Nicholas McGuire 教授，对开源和互联网具有天然的敏感性。因为他长期负责 MTK 手机开发和 Java 设计，所以对界面有非常深入的理解，我们讨论之后，他建议我在 MS4 下重新构建高频感应加热电源的程序，实践一下菜单界面与业务逻辑的分离。在 MS4 的消息机制下，把菜单界面用一个低速定时的消息来激活，以更新显示内容，有一点点类似后来的面向对象编程风格，但这种方法会导致一个问题，那就是每次刷屏显示都要从头到尾来一次完整的程序执行，中间不能打断退出，这会导致业务逻辑响应速度降低，所以开发完成之后没有实际使用，但为后来的菜单界面设计奠定了基础。

有了这次实践经验，加深了我对菜单界面编程的理解，于是我开始寻找当前主流的菜单界面编程方式。在 PC 编程中菜单界面技术非常成熟，早期有 VB、Delphi、VC 等，现在流行的有 Java、C#。恰好雨滴科技公司有 WinCE 项目，用 C# 开发，而 C# 则是微软总结了自己这么多年的开发经验，吸收了 VB、Delphi、Java、VC 的优点整合起来的接近 C 语言的一门新语言，特点是简单、易用、接地气，这符合我的理念，于是同事向我推荐采用 C#，并且做了演示，当时就让我感受到，这正是我想要的东西：一是语法来自于 C 语言，具有兼容性；二是它的命名法非常好，长命名看词识意，提高了阅读性；三是整个架构设计非常完美，命名空间给我留下很深的印象，可以很好地解决一些大项目的重名问题，尤其是全局变量；四是面向对象设计，可以很好地解决菜单界面问题；五是按照 C# 模板设计，可以同时把上位机 C# 编程也学了，之后公司内部可以有很好的语言统一性，而 C# 跟 Java 又类似，这些都可以水到渠成。

有了 C# 作为参考模板，接下来的问题就是如何把 C 语言写成 C# 风格。C 语言没有命名空间这一说法，如何表达？C 语言没有直接的面向对象概念，如何建立？于是我把这些问题跟软件人员交流沟通，最后确定下来选择结构体为核心解决命名空间和面向对象设计，代码写作规范一律参考 C#，这样便于标准化，再结合精简化的 μC/OS，基础框架就搭起来了。与此同时，与其配套的硬件平台 msPLC-DEMO 也已经开发完成，2013 年 6 月正式开始在其上开发软件全新的 msOS。

因为准备工作做得比较充足，前期开发比较顺利，并且遵循 ARM Cortex 微控制器软件接口标准 CMSIS 提出的分层结构（如下图所示）。



把整个系统分为 App 和 System 两层，分别存放在两个目录中。

App 是应用层，存放具体项目的需求；System 是系统层，提供各种应用接口，支撑应用层运行。System 下一般包含三部分：Device、OS 及 GUI，也可以扩展其他的中间件，比如 Modbus、PID 等。

Device 是设备层，为 OS、GUI 及 App 提供底层设备接口，它包含了 ST 提供的硬件驱动标准库。OS 为 μC/OS-II 的精简版本，支持 8 个任务，去掉了很多动态连接的链表而以数组代替，代码精简明了、非常易懂。GUI 是为 App 层提供菜单设计的标准设计库，目前硬件支持 128×64 点阵黑白字库屏，控件支持页面背景字体（BackText）、表（Chart）、标签（Label）及文本框（TextBox）。

后期开发过程中，GUI 部分（尤其是 TextBox）让我花了将近一个月时间才真正分析透彻，首先，因为 TextBox 涉及太多需求，比如显示数据、按键响应、光标移动、焦点闪烁等。其次是因为起初我对分层设计和一些指针的用法理解得不是很准确，当完成整个软件开发后，有一种豁然开朗的感觉。

msOS 初步开发完成后，建立了一个 QQ 群推广，因为有之前 MS 奠定的基础，QQ 群迅速扩大到上千人。考虑到要方便很多初学者，尤其是高校学生，特意把基于 MCU51 的 MS3 按 msOS 的风格统一，重新命名为 MS 推出，衔接 msOS。此外，在群友的建议及启发下，msOS 不断完善，考虑到 msOS 实际需求只有菜单界面和业务逻辑两个任务，不需要八任务的 μC/OS，业务逻辑抢占菜单界面，只要有业务逻辑消息，就需要执行业务逻辑，这样子都可以去掉任务优先级表，其 OS 演化为与 μC/OS 无关的专用版本，这个版本因其任务切换非

常简单、易学易用，深受大家喜爱。

因为 msOS 是针对工控行业开发的，所以需要扩展工业相关的库，比如 Modbus 协议扩展了 HMI 串口屏，解决了大彩屏问题。扩展 TMC262 步进电动机驱动，让步进开发更加容易。扩展 PID 算法，用于温控等场合。

msOS 可以说是完全基于自身的需求推动的，因为自己需要这些功能，所以不停地寻找，而在解决这些的过程中，受到了很多人帮助、肯定与支持，尤其是周庆国教授把 msOS 推荐给清华大学“第二届开源操作系统技术年会”，让它在众多操作系统中，获得好评。广西河池学院彭建盛教授看到 msOS 的价值，专门成立一个重点实验室，跟我公司合作开发、推广 msOS，在此对所有为 msOS 做出贡献的朋友表示深深的感谢。开源推广 msOS，我希望对后来者有参考作用，更希望这个作用不仅仅限于代码方面，更多的应该是这种思维方式，因为这种思维方式很务实。

最后，特别感谢我的妻子刘颖，在开发 msOS 的 4 年中，她不仅承担了所有的家庭事务，还主动梳理公司事务，让我安心做好 msOS。作为一个创业家庭，四年的时间里不赚钱去做开源项目，这是很难被家人所理解的，但她做到了，作为一个丈夫，深深地表示感谢！

欢迎加入 msPLC/msOS 技术交流 QQ 群，群号为 291235815，或登录交流论坛 http://bbs.huayusoft.com/forum.php 的 msPLC/msOS 版面下载最新资料。

王绍伟

2016 年 4 月

目　录

第 1 章
前后台软件架构

最初接触嵌入式，往往从一些简单的例子开始，其中的 main 函数里面有一个 while（1）大循环，很多功能（比如按键、端口访问等）都放在这里处理，此外还有一些中断的处理功能，这类最原始的程序结构，通俗地讲叫“裸奔”，学术名词称为前后台软件架构。中断在前，处理紧急事务，大循环在后，处理低速事务。

1.1 MCU51 的发展历史

国内大部分嵌入式人员了解嵌入式是从学校开设的 MCU51 开始的，笔者也一样，2000 年接触的是 Atmel 公司的 AT89C51 系列处理器，8 位数据总线，16 位地址总线，最高 24 MHz 外部时钟，12 个时钟周期，4KB ～ 64KB FlashROM 和 128 KB ～ 1KB RAM。该系列处理器，因为工作主频低，并且一个指令周期需要 12 个时钟周期，实际等于一秒钟最多只能执行 2 M 条指令，处理速度低。加上其内部的 ROM、RAM 太少，只能做一些简单的控制，这也是 MCU 的名称来源：“微控制单元”，所以这个阶段，很多开发采用汇编语言来编程，C 语言编程处于萌芽阶段，目标文件需要用 FlashROM 烧录器，程序调试依靠仿真器来完成。因为当时低阶半导体制程，工作电压 5 V，相比今天的高阶工艺的 ARM 芯片，抗干扰、抗静电能力较强，加上 MCU 厂商较少，竞争压力小，测试完善，所以 ATMEL 的 MCU51 可靠性很高，口碑很好。

基于当时 MCU51 主频低、ROM、RAM 少、价格贵的特征，软件开发基本上都是以汇编为主，以提高效率、降低资源，从而降低 MCU51 的成本，幸好那个时期控制的对象并不复杂，主要是一些机械、仪表类控制对象，都是简单的逻辑处理，带一些数码管或者是黑白液晶显示器，汇编确实比较适合，这算是最早的“裸奔”软件架构。

2000 年以后，周立功单片机公司代理飞利浦半导体公司（独立后改名为 NXP），推广 NXP 的 MCU51，比如 P89C52x2，可以通过串口 TXD、RXD、PSEN 和 Reset 直接下载，不再需要专用的 FlashROM 烧录器，因为烧录简单，时间又短，都可以采用串口调试来取代仿真器，这大大降低了嵌入式开发人员的开发门槛。同时周立功公司开始推广 Keil-C51 编译器，让嵌入式人员真正走上了 C 语言编程。基于以上两点，周立功公司通过一系列

MCU51 评估板，迅速占领了嵌入式评估板市场，尤其是早期最出名的 DP-51 开发板，笔者就是基于这款评估板正式走上嵌入式之路的，基于它开发了“实用单片机系统 MS”，也为后来的 msOS 打下了基础。

NXP 的芯片相比 ATMEL 来说，虽然价格相差不大，但此时的价格相比以前已下降了不少，接近普及，并且因为支持串口下载，6 时钟周期模式性能可提升一倍，增加了很多特殊寄存器，扩展了一些常用功能如 AD、DA、PWM、CAN，极大地丰富了 MCU51 的内涵，让客户有更多的应用选择，这一切都让大家意识到 MCU51 普及应用时代的到来。后来国内单片机厂商宏晶科技推出了基于上电串口下载程序，不需要 PSEN 和 Reset 脚的解决方案，集成了更多的功能，细分了各种品种，并且以低廉的价格正式引爆了 MCU51 市场。我们来回顾一下 MCU51 的发展历史：

1）INTEL 发明了 8031；

2）ATMEL 在 MCU 内部集成了 FlashROM 和 RAM，实现了真正意义上的单芯片方案；

3）NXP 采用四线（TXD、RXD、PSEN、Reset）串口下载程序，抛弃了仿真器；

4）周立功普及 MCU51 开发板，推广 Keil-C51 编译器，让 MCU51 广泛地进入学校。

5）STC 采用两线（TXD、RXD）上电串口下载程序，增强性能及功能，细分品种，让 MCU51 无处不在。

以上几点大家可以看到，MCU51 的进步除了 Intel 发明之外，还有就是基于需求、方便使用，属于微创新，而这些微创新，却极大地普及了 MCU51，在市场中产生了质变，然而这些微创新往往不受技术人员重视，他们甚至嗤之以鼻，看不到背后的市场效果。

STC 创始人姚永平对技术与市场的独到认识如下：

1）STC 做的是通用 MCU51，评估板市场已经很成熟，所以初期不出评估板，避免技术支持压力，以透明低价直接挤占大厂市场，虎口夺肉。

2）充分挖掘老工程师对于 MCU51 的认同，细分市场，细分功能，形成一个系列，覆盖大部分需求，区分价格。进一步提高主频、集成度，减少外围器件，比如单指令周期、40 MHz 时钟、支持内部 RC 时钟，无需外部时钟，甚至考虑内置滤波电容，加上其原有的高抗干扰、抗静电能力，让 MCU51 成为一颗真正意义上的单芯片方案，用户不需要太多的注意细节，比如 PCB 布板、干扰、静电等问题。此外，专门针对退出 MCU51 市场的大厂提供替换型号，承接它们的客户。

3）MCU51 简单易用，特别适合高校等教学市场，在 ARM 的 Cortex 系列大举进军嵌入式市场，与 MCU51 重叠之时，深入挖掘高校、学生市场，推出开发板，出版各种书籍，继续延续 MCU51 生命。

今天 STC 的单片机已经广泛地被国内外同仁接受，这都是姚先生独到的微创新带来的成就。姚先生作为一个技术人员，深刻认识到市场对技术的影响，尤其是他的这句话深深地影响着笔者：自己做的产品，要建立品牌，天天给她施肥、浇水，一点点地完善，她就能茁壮成长，切忌喜新厌旧。