工程师经验手记

玩转.NET Micro Framework移植

——基于STM32F10x处理器

莫 雨 编著

北京航空航天大学出版社
BEIHANG UNIVERSITY PRESS

工程师经验手记

玩转.NET Micro Framework 移植
——基于 STM32F10x 处理器

莫　雨　编著

北京航空航天大学出版社

内 容 简 介

本书循循善诱，带领大家进入.NET Micro Framework 移植这个神奇的领域。全书内容总体上分为三个部分：第一部分介绍.NET Micro Framework 的基本概况，比如应用领域、发展前景、嵌入式系统的对比等，让读者大致了解它所处的地位；第二部分是熟悉开发环境，比如需要什么开发工具、如何编译代码、如何调试等，让读者了解移植所需要做的准备工作；第三部分是全书之重，主要介绍如何将.NET Micro Framework 移植到 STM32F103ZE 处理器上及需注意的要点，内容涉及向量表、USB 驱动、FLASH 驱动等，让读者明白如何从无到有进行移植。本书附录中有“快速上手指南”，读者可根据其中的步骤快速地进行系统编译。本书共享书中所有源代码，请到作者博客或北京航空航天大学出版社网站下载。

本书的读者对象是：对.NET Micro Framework 移植非常感兴趣的朋友，只要具备基础的C++知识，就能根据书中的内容一步一步实现移植；对于想了解和使用 STM32F10x 的读者，也具备一定的参考价值；当然，还有对嵌入式开发有着浓厚兴趣，一直支持 norains 的朋友们。

图书在版编目(CIP)数据

玩转.NET Micro Framework 移植 ：基于 STM32F10x 处理器 / 莫雨编著. -- 北京 ：北京航空航天大学出版社，2012.4

ISBN 978-7-5124-0723-7

Ⅰ.①玩… Ⅱ.①莫… Ⅲ.①计算机网络—程序设计 Ⅳ.①TP393.09

中国版本图书馆 CIP 数据核字(2012)第 024983 号

玩转.NET Micro Framework 移植
——基于 STM32F10x 处理器

莫 雨 编著

责任编辑 宋淑娟

*

北京航空航天大学出版社出版发行

北京市海淀区学院路 37 号(邮编 100191) http://www.buaapress.com.cn

发行部电话：(010)82317024 传真：(010)82328026

读者信箱：emsbook@gmail.com 邮购电话：(010)82316936

涿州市新华印刷有限公司印装 各地书店经销

*

开本：710×1 000 1/16 印张：18.75 字数：422 千字

2012 年 4 月第 1 版 2012 年 4 月第 1 次印刷 印数：3 000 册

ISBN 978-7-5124-0723-7 定价：39.00 元

序

2006年才偶然知道.NET Micro Framework，那是无意间翻阅一本当年的《程序员》杂志，发现有一篇马宁所写的介绍.NET Micro Framework的文章。文章称.NET Micro Framework不仅可以自启动，而且所提供的托管代码库还可以非常方便地操作硬件，比如操作GPIO、串口、USB和以太网之类的接口，这一瞬间就把仅有Windows编程和工控经验的我深深吸引住了。

也就是在同一年，微软中国和CSIP（工业和信息化部软件与集成电路促进中心）签署合作备忘录。2007年9月18日，微软中国与CSIP联合主办的.NET Micro Framework技术大会在京隆重召开，正式把.NET Micro Framework引入中国。

从那个时候开始，我便着手研究.NET Micro Framework，并写一些关于.NET Micro Framework的技术文章，在积极推进当时我所在公司与CSIP合作开发.NET Micro Framework项目未果的情况下，在马宁的引荐下，2008年我加入了微软中国.NET Micro Framework项目组，从而得以进入嵌入式领域，开始了我的ARM开发生涯。

虽然从2007年开始，CSIP和微软中国便大力推广.NET Micro Framework，但定位仅仅是与高校合作，其推出的近乎天价的几万元的双子星教育箱，也只有高校可以买单，一般嵌入式爱好者是无缘使用的。而国外推出的.NET Micro Framework开发板，动不动几千元的价格，也不是一般爱好者所能接受的。此外更致命的是，.NET Micro Framework Porting Kit不仅TinyCLR不开源，而且还收取600美金的授权使用费，所以更限制了.NET Micro Framework在中国，乃至在世界的推广。

直到2009年，微软才幡然醒悟，不仅.NET Micro Framework Porting Kit完全免费，而且还以更为彻底的源代码授权方式（Apache 2.0 license）全部开源了.NET Micro Framework代码，并且源代码也交予社区进行开发维护。

当此时也，物联网风起云涌，ARM推出了Cortex系列芯片，各大厂商的云计算平台更是甚嚣尘上，而恰恰最重要的“端”这个依托平台正是.NET Micro Framework最适合的施展舞台。

为了顺应这个发展潮流，我以个人之粗见，尽微薄之力，推出了全球第一款基于Cortex－M3的.NET Micro Framework开发板，使.NET Micro Framework爱好者

能用较低的代价，便可以进入.NET Micro Framework学习的殿堂。

也就是在那个时候，我结识了来自深圳的莫雨，他采用了自己的方式，用了近半年的时间，一步步、认认真真地完成了.NET Micro Framework的移植工作。难能可贵的是，他把自己的移植过程结集成书，使有心进行.NET Micro Framework移植的朋友多了一盏指路明灯。尤其值得一提的是，莫雨的这本新书，应该是全球第一本写.NET Micro Framework底层移植的书。

需要指出的是，对于没有多少嵌入式基础的读者，.NET Micro Framework应该是进入嵌入式开发殿堂最好的切入点，为什么这么说呢？因为.Net Micro Framework相对于其他嵌入式系统而言，既不简单（相对于μC/OS-Ⅱ），也不复杂（相对于Windows CE、嵌入式Linux）；并且包罗万象，知识面涉及很广，不仅包含一个小巧的操作系统，而且还能在CLR精简运行时，包含一个强大的在线调试系统，真可谓“麻雀虽小，五脏俱全”。

此外，学习.Net Micro Framework也是Windows平台开发人员顺利过渡到真正嵌入式开发的最佳渠道。而且从这一点出发，还会很容易地过渡到其他的嵌入式系统上，如μC/OS-Ⅱ、μCLinux、嵌入式Linux等。而Windows CE则不然，学过Windows CE的人都知道，其开发的难点就是驱动开发和平台移植，而这种代码的编写、编译和调试都要基于微软自己的PB开发环境（目前已作为插件成为Visual Studio的一部分），开发者很难接触到MDK、RVDS、GCC等开发工具，因而也就很难转入到其他的嵌入式系统。至于Windows CE的应用开发，它与PC平台开发几乎没有什么区别，也许你已经进行了几年的Windows CE应用开发，那么从严格意义上讲，你仍不是一个真正的嵌入式开发人员。当然，如果仅仅学习嵌入式Linux的应用开发，那么你也称不上一个真正的嵌入式开发人员，真正的嵌入式开发人员至少要有与中断、芯片寄存器打交道的经历。

总之一句话，如果你已经学习了一阵子.Net Micro Framework的应用开发，而且已经不满足当前所学，还想进一步深入研究和开发.Net Micro Framework，那么莫雨的这本关于底层移植的书，你不得不读，它会帮你拨开底层移植的层层迷雾，让你尽享.Net Micro Framework底层移植的快乐。

刘洪峰（网名：叶帆）
2011年8月于北京

前　言

一、初识.NET Micro Framework

接触到.NET Micro Framework 其实是一个非常偶然却又必然的机缘。

当时我在做车载设备，其架构分为两个主要部件，分别是导航板和控制板。导航板用的是 ARM11 核心的 CPU，运行的是 Windows CE 系统，主要用来运行导航软件；而控制板则用的是 MCU 或低端的 ARM，用来控制外围设备以及与汽车的沟通。当时因为公司人员配置的问题，控制板这块几乎没有人手有能力去开发，所以只能购买其他公司做好的板子。而这对于一个公司来说，无异于喉咙被对手扼住，生存和死亡就看对方是否高兴。

鉴于这种情形，我开始了控制板的研究。但以前习惯了有操作系统作为支撑的开发方式，现在陡然进入一个对自己犹如一片白纸的领域，可谓无从下手。比如在 Windows CE 中创建多任务，只需要调用几个简单的 API 函数即可；但是在 MCU 这个区域，根本就没有操作系统的支撑，一切都只能自己动手——自己写调度算法、自己写逻辑关系等。

于是，为了打破这种困境，我开始寻找轻量级的嵌入式操作系统。经过多方比较，找到了 μC/OS-Ⅱ。只可惜 μC/OS-Ⅱ的结构化不符合自己的要求，因为系统与应用的关联度太大了。比如说，创建一个任务，就必须要修改操作系统代码，这对于极度追求稳定性的自己来说是不符合要求的——因为谁也无法保证是否能够完全避开“地雷”。后来，我便索性不再搜索成熟的嵌入式操作系统，而打算自己重写一个，只要能够完成最简单的任务即可。也许冥冥中天注定，在这期间看到了网友叶帆关于.NET Micro Framework 的一系列文章，而.NET Micro Framework 又刚好满足系统与应用分隔的原则，于是就开始了与.NET Micro Framework 的不解之缘。

二、内容特色

本书主要介绍与.NET Micro Framework 移植相关的内容。说到“移植”二字，可能不少初学者闻之色变，认为这是不可企及的高度，特别是将整个框架移植到新的 CPU 中，感觉难度更如登天。不过先别着急，虽然本书打着“移植”的旗号，但实际上是面对初学者的。只要具备 C++的基本知识，并按照本书的介绍一步一步去完成，就能真正踏入嵌入式领域。

本书的移植目标是STM32F10x，它是ST公司出品的一款高性能、低功耗的CPU。为什么选用这款CPU，而不是市面上常见的三星系列呢？因为STM32F10x采用的是Cortex-M3核心，是ARM11的下一代产品，同时也是ARM的未来发展趋势。更为重要的是，Cortex相对于之前的ARM系列，变动很大，特别是中断机制方面更是大相径庭。虽然M3是Cortex性能较低的一个版本，但指令集基本是一致的，因此，只要熟悉了STM32F10x的工作原理，对日后转为更高阶的Cortex版本就具有非常重要的参考价值。更为有意思的是，.NET Micro Framework并没有完全实现Cortex核心的代码，而需要用户自己去更改相应的流程，但这对于进一步了解.NET Micro Framework的工作原理却是大有裨益。

虽然本书是基于STM32F10x编写的，但却不会太过深入讲解该CPU的具体特性，而是点到为止——.NET Micro Framework需要什么，就只说什么。因为本书的主要目标是介绍.NET Micro Framework的移植，如果额外增加对STM32F10x特性的详细说明，则无疑会增加书的厚度，更何况市面上关于STM32F10x的优秀书籍也不少，我又何苦在这再造轮子呢？如果读者您是STM32F10x的忠实粉丝，那么不妨将本书当做是对STM32F10x一个具体项目的实现。

本书的目标在于带领各位读者进入.NET Micro Framework移植的大门，根据本书的介绍来移植一个能运行托管代码的最简单的TinyCLR。该目标听起来似乎并不那么宏伟，但麻雀虽小，五脏俱全，只要能够达成这一目标，也就意味着你对于.NET Micro Framework的了解更深了一层，后续更多的动作也就能够很容易地举一反三了。

三、致 谢

在本书编写过程中得到了很多人的帮助。负责书中源代码测试的有：蓝应志、余海标、朱艳锋、龚军波、王靖、钟镇轩和刘翔宇。负责搭建硬件平台，为软件提供测试基础的有：马俊、黄明飞、覃玉恩和龙晓波。负责书中插图设计，为本书添光增彩的有：覃思、莫多、洪玲和梁菲。

还有一些朋友需要特别感谢。首先是网友叶帆，正是你的文章指引我进入了.NET Micro Framework领域，并且在移植过程中给予我不少建议，让我少走很多弯路，你不愧为微软的.NET Micro Framework项目组成员，更无愧于微软MVP的称号。叶帆的博客不能不推荐，其地址为http://blog.csdn.net/yefanqiu。

其次是向飞，一个实力非常高超的网友，如果没有你的无私帮助，说不定我现在还在USB的泥潭中苦苦挣扎，书中那么多的错误也不会及时地被发现。

下一个是老尹，你让我知道除了车载以外还有那么广阔的领域，而那些都是.NET Micro Framework触角可以碰触的地方，这让我对.NET Micro Framework的前景充满了信心。

当然还有曾盛洲，如果不是你及时而又耐心地回答我工作中那些繁杂的问题，我

根本就不会有这么多时间去研究.NET Micro Framework。

最后还有我的妻子，如果没有你坚定的支持，那么我在工作的抉择上还是犹豫不决，根本就无法如此心平气和地完成本书。

当然，还要感谢北京航空航天大学出版社的工作人员，本书的顺利出版离不开你们。

尽管我尽了极大的努力，但限于经验水平，书中的错误还是难免。如果读者您找到了这些错误，还望不吝指教，可以直接在我的博客 http://blog.csdn.net/norains 上留言，或者发邮件到 norains@gmail.com。在此，先行拜谢！

此书即将面市，吾姑且言之，众位看官姑且听之：

吾乃一名沉溺于嵌入式开发而不知日月轮转的工程师，2012 年新晋微软最有价值专家。凡是与技术相关之种种，无论大小繁杂，均欲一窥究竟，故涉猎甚广。期间所获之造诣，均载于所建博客，于业界颇有其名。曾不知地厚天高，2010 年以《Windows CE 大排档》一书献丑，所幸友人们顾及薄面，不致板砖遍地，倒有鲜花不少，于吾心有戚戚焉。现再推新作，虽已“二度进宫”，然仍忐忑不安。尤恐读者不满，板砖鸡蛋伺候，以致环境污染，千古罪人是也。怎奈书稿既成，如独自暗藏，恐心痒难耐。久思熟虑之后，乃纵性横心，拉脸弃面，令其曝光于世。若对本书愤懑，乃吾之过也，不若将其置之桌脚，便可疏导气闷。切记，勿忘！

莫雨(norains)
2011 年 8 月于深圳

目 录

第 1 章 概 述

本章只介绍.NET Micro Framework的一些基础概况，并不涉及非常具体的技术问题，如果各位朋友对此已经熟稔，可以跳到第2章。

1.1 什么是.NET Micro Framework

.NET Micro Framework是专门为小型的、资源有限的设备准备的。它为这些设备提供了一个完整的并且可以谓之革命性的开发和运行环境，并以此来加快产品的研发进度。

对于目前的.NET开发者来说，这意味着他们所创建的程序可以运行于非常多的嵌入式设备，比如PC的远程控制、服务器，甚至云计算，而这一切都可以使用同样的模型和工具。.NET的开发者们，此时是否觉得热血沸腾了？

而对于目前的嵌入式开发者来说，又带来了怎样的变革呢？他们可以更容易地开发针对于具体应用的程序，并且与以前相比大大缩短了产品上市的时间。时间意味着市场，仅就此一点来说，也许很多嵌入式开发者也开始跃跃欲试了吧？

可以这么说，.NET Micro Framework的意义在于将一个与桌面开发一致的Visual Studio体验带到了嵌入式的世界。

说了这么多，那么.NET Micro Framework究竟能干什么呢？简单来说，借助它可以：

① 更容易地开发强大的、更具影响力的复杂的应用程序；

② 更安全地通过有线或无线协议来连接设备；

③ 能够以更低的成本、更快的速度进行可信赖的开发，而这一切包含了设备，服务器和云计算。

相对于文字来说，也许图1.1.1更能让各位读者明白.NET Micro Framework究竟能做什么。

图 1.1.1 .NET Micro Framework 设备

1.2 .NET Micro Framework 的架构

任何一个框架都必定有其独特的架构，对于.NET Micro Framework来说也是如此，其架构如图1.2.1所示。

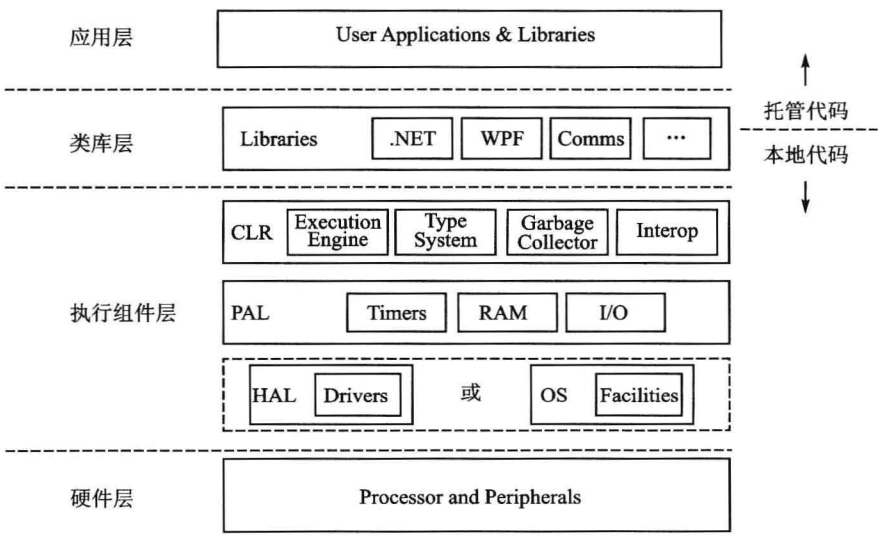

图 1.2.1 .NET Micro Framework 架构

1.2.1 Hardware Layer(硬件层)

对于 Hardware Layer 来说,有多种型号的微处理器可以选择,比如 ARM7,ARM9,Cortex,Xscale,ARC 和 ADI。Hardware Layer 也可以放在操作系统之上,当然这并不意味着硬件不存在了,只不过是.NET Micro Framework 无法直接与硬件沟通而已。那么,此时.NET Micro Framework 该如何操作硬件呢?只能通过系统提供的服务了。比如说,在 Windows XP 操作系统中,可以通过 CreateFile 来对串口进行操作。事实上,在.NET Micro Framework SDK 中用到的模拟器就是一个最好的将.NET Micro Framework 部署到操作系统之上的样例。这种模式的部署给.NET Micro Framework 带来的最大影响便是,其所能做的,只能取决于其所属的系统。

1.2.2 Runtime Component Layer(执行组件层)

Runtime Component Layer 包含三个组件:CLR(公共语言运行库)、HAL(硬件抽象层)和 PAL(平台抽象层)。在.NET Micro Framework 的术语当中,本层被称为 firmware(固件)。

1. CLR(公共语言运行库)

.NET Micro Framework 其实是.NET Framework CLR 的子集,其所用的运行环境也是由.NET Framework 提供的。这两者最大的不同在于,.NET Micro Framework 做了裁剪,令其能够更适合小型的嵌入式设备。

部署工具包提供了公共语言运行库的代码,而这些代码是与硬件无关的,所以它能够适应于多种编译器以及不同的 CPU 架构类型。

2. HAL(硬件抽象层)和 PAL(平台抽象层)

如果 CLR 需要操作硬件的话,那么它就必须透过 HAL 和 PAL。无论是 HAL 还是 PAL,其实本质都是用 C++编写的驱动。正如名字所表示出的意义一样,HAL 是与硬件密切相关的,而 PAL 则是完全独立于硬件的。

一般来说,驱动的 HAL 和 PAL 是成对出现的,并且是互相协调来完成一个任务。因为 CLR 会调用 PAL 的代码,然后在 PAL 内部又会调用 HAL 的代码,因此,正是这一层衔接一层的调用让 CLR 实现了操作硬件的功能。

HAL 除了包含驱动以外,还包含启动代码(bootstrap code)。当设备开始上电时,启动代码就进行低阶的硬件初始化,接着再运行 CLR,由 CLR 来进行设备的高阶初始化。不仅如此,HAL 还包含了配置信息,比如芯片支持包和板间支持包,前者指定特定的芯片,后者则指定相应的开发板。

1.2.3 Class Library Layer(类库层)

Class Library Layer 是.NET Micro Framework 中可复用的面向对象的组件,

开发者可将之用于嵌入式程序的开发。当然，第三方的开发者也可将自己的类库添加到组件中，最简单的例子便是开发板的厂家，他们会提供一系列操作开发板外围器件的类库供开发者使用，以减少开发者的时间。

1.2.4 Application Layer(应用层)

Application Layer 位于.NET Micro Framework 的顶层，在该层中用户可使用托管代码为设备开发程序，只不过稍显遗憾的是，目前只能使用C#进行托管代码的开发。当然，完全可以有理由相信，以后将有更多的语言添加到这个领域中来。

说点题外话，从技术角度而言，其实应用层完全可以称得上是固件；但在.Net Micro Framework 的术语中，“固件”这个词只能给予执行组件层。那么，应用层应该叫什么呢？因为在微软的文档中也没有说明，所以本书在这里也只好留白了。☺

1.3 .NET Micro Framework 与嵌入式系统的比较

熟悉微软的嵌入式产品线的朋友都知道，微软的嵌入式产品可谓丰富，最常见的有 Windows CE，以及国内朋友可能接触比较少的桌面 Windows 嵌入式系列。既然已经有了如此多的嵌入式产品，那么为什么还要有一个.NET Micro Framework 呢？或是说，.NET Micro Framework 有什么优势呢？在回答这个问题之前，先看看表1.3.1所列的比较。

表 1.3.1 .NET Micro Framework 与其他嵌入式系统的比较

平台 / 属性	.NET Micro Framework	Windows CE	Windows XPe
设备范例	传感器，辅助显示，健康监控设备，远程控制，机器人	手持 GPS，PDA，汽车自动化，机顶盒	小客户端，ATM 设备，信息查询设备
设备特性	联网，小型，耐用度高，图形界面	联网，图形设备，服务器，浏览器，RAS，DirectX	计算机性能级别，计算机网络
资源占用	250～500 KB 托管代码	300 KB 以上，无托管代码；12 MB 托管代码	40 MB 以上，取决于所选特性
电源	功耗非常低	功耗低	功耗一般
CPU	ARM7，ARM9，无 MMU	X86，MIPS，SH4，ARM，带 MMU	X86
实时特性	非实时	硬实时	通过第三方扩展可以具备实时特性
托管代码 vs. 本地代码	通过.NET Micro Framework 运行托管代码，本地代码仅仅是在构建框架的时候使用	两者都支持，但托管代码必须借助.NET Compact Framework	两者都支持，但托管代码必须借助.NET Framework