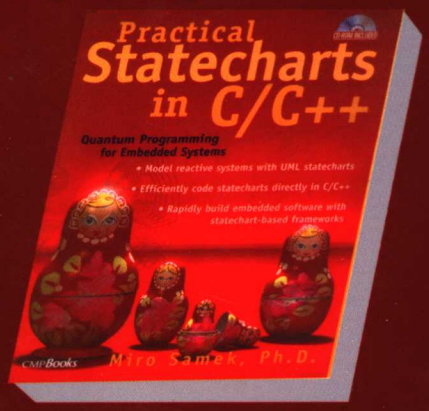
嵌入式系统译丛

[美] Miro Samek Ph.D. 著
敬万钧 陈丽蓉 译

嵌入式系统的微模块化程序设计——实用状态图C/C++实现

Practical Statecharts in C/C++
Quantum Programming
for Embedded Systems



CD-ROM INCLUDED

CMPBooks
北京航空航天大学出版社
http://www.buaapress.com.cn

内 容 简 介

建模反应式系统而不使用重量级的工具，作者的量子编程(QP)是一种新的范型。它把状态图作为一种设计方法，而不是作为特殊工具来使用。在本书的第1部分，给出相关概念清晰而明确的叙述，包括传统的有限状态机和状态图，以及基于状态图的设计模式；给出了可运行代码，使读者能通过实际操作来学习量子编程，以及学到状态嵌套如何导致行为继承和如何通过按差异编程而实现重用。第2部分完整地叙述了量子框架的实现，以及说明如何在应用中使用它，并移植到所选用的RTOS。本书适于嵌入式系统、实时系统及UML状态图的相关工程设计人员使用，并可作计算机科学和电气工程高年级学生的教学用书。所附光盘包含了作者的量子框架的全部源代码、散见于全书的所有练习的答案以及一个RTOS-32的评估板——X86处理器的32位实时操作系统。

图书在版编目(CIP)数据

嵌入式系统的微模块化程序设计：实用状态图C/C++实现/(美)萨梅克(Samek, M.)著；敬万钧等译.
北京：北京航空航天大学出版社，2004.8
ISBN 7-81077-415-8

Ⅰ.嵌… Ⅱ.①萨…②敬… Ⅲ.模块化程序设计
Ⅳ.TP311.11

中国版本图书馆CIP数据核字(2004)第058172号

嵌入式系统的微模块化程序设计
——实用状态图C/C++实现
Practical Statecharts in C/C++
Quantum Programming
for Embedded Systems

[美] Miro Samek Ph. D. 著　　敬万钧　陈丽蓉译
责任编辑　王　瑛
责任校对　戚　爽

*

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(100083)　发行部电话：010-82317024　传真：010-82328026
http://www.buaapress.com.cn　E-mail:bhpress@263.net
涿州市新华印刷有限公司印装　各地书店经销

*

开本：787×960　1/16　印张：21.5　字数：482千字
2004年8月第1版　2004年8月第1次印刷　印数：6 000册
ISBN 7-81077-415-8　　定价：48.00元(含光盘1张)

译者序

Miro Samek 博士是美国 Integri Nautics 公司的首席软件结构师。他在该公司以及此前在 GE 医疗系统都是从事实时软件开发，长期的工作使他在嵌入式实时软件开发方面积累了丰富的实践经验。他将其所积累的嵌入式实时软件开发的经验写成了这本理论和实践并重，而实践性更强的著作。

本书中除概念的叙述之外，Samek 博士还提供了完整的源代码，包括完整的软件基础结构和用以执行裁剪自多线程嵌入式应用的状态图。灵活的、有效的、可移植的、可伸缩的以及可维护的轻量级量子框架允许快速、直接地从 UML 状态图以 C 或 C++ 着手实时系统的工作。

量子框架是完整的实时嵌入式实现环境——一个设计用以与任何 RTOS 一起工作的活动对象框架。它支持快速原型构作，可在开发的任何阶段容易地进行状态的转换，以及可选择实现语言(C 或 C++)，以与目标系统的资源约束相匹配；它最大的好处是非常紧凑，整个 QP 代码典型地要求少于 5 KB 的代码和数据。这些都降低了实现抽象概念的困难程度，对希望了解层次式状态机能力的嵌入式软件工程师是非常必要的。

关于本书的译名，为适合我国当前的具体情况，将其定为“嵌入式系统的微模块化程序设计”，而原作名应直译为“嵌入式系统

的量子程序设计”。但量子程序设计并不为我国读者所熟悉，经与有关专家商议，确定使用现译名。而在正文中，由于作者多处将 QP 与量子力学概念进行比拟，为维持叙述的一致性，故在正文中仍使用“量子程序设计”一词，请读者加以注意。

在本书的译出中，得到了电子科技大学“嵌入式实时系统研究中心”的大力支持，以及中心主任熊光择教授的关心和帮助；同时在北京航空航天大学出版社马广云老师的鼎力帮助下，本书才得以及时面世。在此均表示由衷的感谢。

本书第 1～6 章由敖万钧翻译，第 7～11 章由陈丽蓉翻译。译者虽然尽心尽力，多次修改译稿，但疏忽欠妥之处定有，恳请业内同仁予以指正。

译　者

2003.8 于成都

目　录

第 1 部分　状态图

第 1 章　量子编程的快速浏览

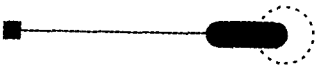
第 2 章　状态图速成

第 6 章 继承状态模型

第 2 部分 量子框架

第 7 章 量子框架介绍

第 8 章 量子框架的设计



第 1 部分　状态图

必须对输入事件作出反映的系统就是事件驱动系统。这是当前一种流行的系统，而为了定义和实现事件驱动系统，状态机是一种很好的形式方法，UML状态图代表了状态机理论和表示法的当前状况。

第 1 部分介绍状态图的概念，叙述直接用 C 和 C++编码状态图的具体技术，并且提供主要的基于状态图设计模式的一个小目录。读者将会了解到状态图是一种强有力的设计方法，可以使用它而不必借助于复杂的代码综合工具来进行设计。

第1章

量子编程的快速浏览

我发现，为了找到你喜欢或讨厌的人，除了同他一起旅行外，没有更可行的办法。

——**Mark Twain**

在过去的30年间的软件开发中，图形用户界面的成功是最为引人注目的[①]。今天，这个概念已为大家所熟悉而不需要再加以赘述。虽然从一开始，窗口、图标、菜单以及指示器就已是直观的，并易于为用户接受的，但是对程序设计者来说，它们仍然是一种挑战。这是因为GUI内部结构对很多新手是困难的。他们经常发现它是陌生的、不合潮流的、思路不清或古怪的。GUI编程之所以有这些不同之处，是因为它不像传统的数据处理。它是完全事件驱动的，事件能以任何顺序、在任何时刻出现，应用程序必须随时准备好去处理它们。GUI是复杂反应式系统的一个例子。

没必要去寻找其他的反应式系统的例子，事实上，所有PC，MAC以及其他通用计算机的CPU仅仅消耗世界范围微处理器产品的1%左右，而其他99%的微处理器芯片都为各种嵌入式系统所用，嵌入式自然是最主要的反应式系统。尽管具有如此的普遍性，但绝大多数此类系统的编码都是非常难以理解、调整和维护的。建造此类软件的理论基础已经出现了10多年，但是，这些想法却并未成为主流。量子编程(QP)就是要使这些现代方法为程序员们所接受。QP是直观设计模式、习惯方式、具体实现以及易于明白技术的集合，这使读者可立即着手使用而无需去研究各种复杂的工具。

1.1 终结钩子——GUI应用程序的剖析

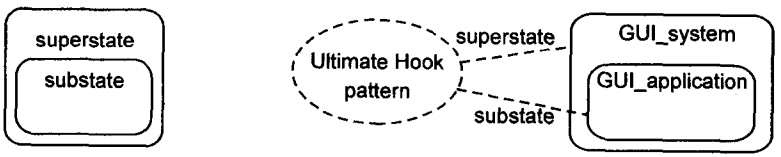
早期的GUI设计师面临着困难的任务：一方面，GUI应用程序必须使得从开矩形窗口到划分屏幕，以及令人眼花缭乱的屏幕帮助等事情成为非常惯用的动作；另一方面，系统又必须保证观感一致性，而且具有这些标准性能的应用程序还必须简单。对这种矛盾

① 窗口、图标、菜单和指示器(WIMP)界面的概念是由斯坦福研究所的Douy Englebart和他的队伍在1968年的西部联合计算机会议上首次公布的。

的要求如何来进行折衷呢?

现在这个问题似乎简单了——使用终结钩子的办法[Petzold,1996]。其想法是非常简单的,GUI 系统(如 Windows)首先分发所有的事件到应用程序(Windows 调用在应用程序中的一个特殊函数)。如果应用程序没有处理,则各事件返回系统,这就建立了一个事件处理的层次顺序。应用程序在概念上处于层次的最低级,有机会对每一事件作出反应;因此,应用程序能够定制它的事件行为的各个方面,同时,所有未被处理的事件返回较高级层次(即回到系统),系统对它们按照标准的观感处理。这是一个"按差异编程"的例子,因为应用程序人员仅仅对不同于标准的系统行为进行编码。

David Harel 独立地将同样的想法应用到有限状态机形式方法[Herel,1998]。大约在 1983 年,他发明了状态图。作为规定复杂反应式系统的一种强有力的方法,状态图较之传统有限状态机的主要创新之处在于层次式状态的引入。为了了解它的意思,将被嵌套状态(子状态)和包围状态(超状态)之间的关系示于图 1.1(a)。此状态图首先要处理子状态范围中的所有事件,如果子状态不能处理这些事件,状态图自动地送事件到下一个较高级状态(即超状态)。当然,状态的嵌套是可以进一步加深的,而处理事件的简单规则反复地被用于嵌套的任意级。



(a) 嵌套状态的状态图表示法　　(b) 表示终结钩子设计模式的状态图

图 1.1　状态层次图

Harel 的状态层次语义学是构成 GUI 系统的终结钩子设计的核心。换句话说,状态图直接支持终结钩子模式。当其把超状态改名为 GUI_system,子状态改名为 GUI_application 时,这就变得非常明白了,如图 1.1(b)所示。现在有了一个非常有趣的结果:基本的终结钩子设计模式转化成了非常简单的状态图!这个强有力的状态图是非常简单的,因为它在这个抽象级没有包含任何状态转换。

通常,在状态图中引入的状态层次已经证实下述是正确的:

结果,高度复杂的行为已不能用简单的、"平面的"状态转换图来方便地描述。其根源在于不可管理的大量的状态,它们可能导致无结构的和混乱的状态转换图。

——[Herel+,1998]

当然,层次式图形经常比传统的平面图有更为简单和更好的结构,但是,这并不是状态层次重要性的主要理由,而只是副作用之一。状态层次即使在没有大量状态和转换的