

<<嵌入式系统导论>>

图书基本信息

书名：<<嵌入式系统导论>>

13位ISBN编号：9787111360216

10位ISBN编号：7111360214

出版时间：2011-12

出版时间：机械工业出版社

作者：（美）Edward Ashford Lee,（美）Sanjit Arunkumar Seshia

译者：李实英,贺蓉,李仁发

版权说明：本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问：<http://www.tushu007.com>

<<嵌入式系统导论>>

内容概要

本书是业界第一本关于CPS的专著，重点论述系统模型与系统实现的关系，以及软件和硬件与物理环境的相互作用。

从CPS的视角，围绕系统的建模、设计和分析这三个方面，本书分成四大部分。

第一部分（第2～6章）分别讲述动态建模、离散建模和混合建模，以及状态机的并发组合与并发计算模型。

第二部分（第7～11章）强调嵌入式系统中处理器、存储器架构、输入和输出、多任务处理和实时调度的算法与设计

，以及这些设计在CPS中的主要作用。

第三部分（第12～15章）重点介绍一些系统特性的精确规格、规格之间的比较方法、规格与产品设计的分析方法以及嵌入式软件特性的定量分析方法。

第四部分包括两个附录，提供了一些数学和计算机科学的背景知识，有助于读者加深对文中所介绍知识的理解。

本书通过大量实例深入浅出地介绍了设计和实现CPS的整体过程及各阶段的细节。

本书适合作为高等院校相关专业“嵌入式系统”课程的教材或教学参考书。

<<嵌入式系统导论>>

作者简介

Edward Ashford Lee 拥有加州大学伯克利分校博士学位，曾为加州大学伯克利分校电子工程与计算机科学系主任，现为该系Robert S. Pepper特聘教授。

他的主要研究方向是嵌入式与实时计算系统的设计、建模和模拟。

Lee教授是IEEE会员，于1997年获得工程教育领域的Frederick Emmons Terman奖。

Sanjit Arunkumar Seshia 拥有卡内基-梅隆大学计算机专业博士学位，现为美国加州大学伯克利分校电子工程与计算机科学系副教授。

他的主要研究方向是可信计算和计算逻辑。

他获得了科学和工程领域的总统早期职业生涯奖（PECASE）和Alfred P. Sloan研究奖金。

<<嵌入式系统导论>>

书籍目录

Introduction to Embedded Systems—A Cyber?Physical Systems

Approach

出版者的话

译者序

前言

符号

第1章 绪论

1.1 应用

1.2 一个实例

1.3 设计过程

1.3.1 建模

1.3.2 设计

1.3.3 分析

1.4 小结

第一部分 动态行为建模

第2章 连续动态

2.1 牛顿力学

2.2 参量模型

2.3 系统的特性

2.3.1 因果关系系统

2.3.2 无记忆系统

2.3.3 线性和时不变性

2.3.4 稳定性

2.4 反馈控制

2.5 小结

练习

第3章 离散动态

3.1 离散系统

3.2 状态的概念

3.3 有限状态机

3.3.1 转移

3.3.2 发生响应时

3.3.3 升级函数

3.3.4 确定性和可接受性

3.4 扩展状态机

3.5 非确定性

3.5.1 形式化模型

3.5.2 非确定性的用途

3.6 行为和轨迹

3.7 小结

练习

第4章 混合系统

4.1 模态模型

4.1.1 状态机的参量模型

4.1.2 连续输入

<<嵌入式系统导论>>

- 4.1.3 状态精化
- 4.2 混合系统的分类
 - 4.2.1 时间自动机
 - 4.2.2 高阶动态
 - 4.2.3 管理控制
- 4.3 小结
- 练习
- 第5章 状态机的组合
 - 5.1 并发组合
 - 5.1.1 并列同步组合
 - 5.1.2 并列异步组合
 - 5.1.3 共享变量
 - 5.1.4 级联组合
 - 5.1.5 通用组合
 - 5.2 分层状态机
 - 5.3 小结
 - 练习
- 第6章 并发计算模型
 - 6.1 模型结构
 - 6.2 同步响应模型
 - 6.2.1 反馈模型
 - 6.2.2 形式规范和形式不规范模型
 - 6.2.3 构建一个固定点
 - 6.3 数据流计算模型
 - 6.3.1 数据流原理
 - 6.3.2 同步数据流
 - 6.3.3 动态数据流
 - 6.3.4 结构化数据流
 - 6.3.5 进程网络
 - 6.4 实时计算模型
 - 6.4.1 时间触发模型
 - 6.4.2 离散事件系统
 - 6.4.3 连续时间系统
 - 6.5 小结
 - 练习
- 第二部分 嵌入式系统设计
 - 第7章 嵌入式处理器
 - 7.1 处理器类型
 - 7.1.1 微控制器
 - 7.1.2 DSP处理器
 - 7.1.3 图形处理器
 - 7.2 并行处理
 - 7.2.1 并行处理与并发处理
 - 7.2.2 流水线
 - 7.2.3 指令级并行
 - 7.2.4 多核架构
 - 7.3 小结

<<嵌入式系统导论>>

练习

第8章 存储器架构

8.1 存储技术

8.1.1 RAM

8.1.2 非易失性存储器

8.2 存储器层次结构

8.2.1 存储映射

8.2.2 寄存器文件

8.2.3 便签式存储器 and 高速缓冲存储器

8.3 存储模型

8.3.1 存储地址

8.3.2 栈

8.3.3 存储器保护单元

8.3.4 动态存储分配

8.3.5C 的存储模型

8.4 小结

练习

第9章 输入和输出

9.1 I/O硬件

9.1.1 脉宽调制

9.1.2 通用数字I/O

9.1.3 串行接口

9.1.4 并行接口

9.1.5 总线

9.2 并发环境下的顺序软件

9.2.1 中断和异常

9.2.2 原子性

9.2.3 中断控制器

9.2.4 中断建模

9.3 模拟/数字接口

9.3.1 数模转换和模数转换

9.3.2 信号调节

9.3.3 采样和走样

9.4 小结

练习

第10章 多任务处理

10.1 命令式程序

10.2 多线程

10.2.1 创建线程

10.2.2 实现多线程

10.2.3 互斥

10.2.4 死锁

10.2.5 存储一致性模型

10.2.6 多线程问题

10.3 进程和消息传递

10.4 小结

练习

<<嵌入式系统导论>>

第11章 调度

11.1 调度的基础知识

11.1.1 调度决策

11.1.2 任务模型

11.1.3 调度程序的比较

11.1.4 调度程序的实现

11.2 单调速率调度

11.3 最早时限优先

11.4 调度和互斥

11.4.1 优先级倒置

11.4.2 优先级继承协议

11.4.3 优先级上限协议

11.5 多处理器调度

11.6 小结

练习

第三部分 分析和验证

第12章 不变量与时序逻辑

12.1 不变量

12.2 线性时序逻辑

12.2.1 命题逻辑公式

12.2.2 LTL公式

12.2.3 LTL公式的应用

12.3 小结

练习

第13章 等价与精化

13.1 规格建模

13.2 类型等价与类型精化

13.3 语言等价与包含

13.4 模拟

13.4.1 模拟关系

13.4.2 形式化模型

13.4.3 传递性

13.4.4 模拟关系的非唯一性

13.4.5 模拟与语言包含

13.5 互模拟

13.6 小结

练习

第14章 可达性分析和模型检测

14.1 开放式与封闭式系统

14.2 可达性分析

14.2.1 Gp验证

14.2.2 显态模型检测

14.2.3 符号化模型检测

14.3 模型检测中的抽象

14.4 活跃属性的模型检测

14.4.1 属性的自动机表达

14.4.2 寻找可接受循环

<<嵌入式系统导论>>

14.5 小结

练习

第15章 定量分析

15.1 关注的问题

15.1.1 极限分析

15.1.2 阈值分析

15.1.3 一般情况分析

15.2 程序图

15.2.1 基本块

15.2.2 控制流图

15.2.3 函数调用

15.3 执行时间的决定因素

15.3.1 循环界限

15.3.2 指数的路径空间

15.3.3 路径的可行性

15.3.4 存储层次

15.4 执行时间分析的基础

15.4.1 最优化问题的形式化

15.4.2 逻辑流约束

15.4.3 基本块的界限

15.5 其他定量分析问题

15.5.1 存储界限分析

15.5.2 能耗和功耗分析

15.6 小结

练习

第四部分 附录

附录A集合和函数

附录B复杂度和可计算性理论

参考书目

<<嵌入式系统导论>>

章节摘录

版权页：插图：DSP处理器通常增加一个或两个额外的阶段来执行乘法，提供一个单独的ALU用于地址计算，提供一个双数据存储器来同时访问两个运算子（后一种设计使用的是哈佛架构）。

然而，一个没有独立ALU单元的简单版本就足以说明嵌入式系统设计师所需面对的问题。

锁存器之间的流水线部分是并行执行的。

显而易见，同时有5条指令执行，每条指令都处于不同的执行阶段。

如图7—3所示的预留表（reservation table）表示得很清楚。

该表给出了可以在左侧同时使用的硬件资源。

在这种情形下，寄存器组出现三次，因为图7—2中的流水线假设每个周期可以出现两次读和一次写寄存器文件。

图7—3的预留表给出一个程序中的指令序列A, B, C, D, E。

在第5个周期，取指令E，这时指令D从寄存器组读取数据，指令C使用ALU，指令月从存储器读取数据或向存储器写入数据，指令A将结果写回寄存器组。

指令A的写操作出现在第5周期，但是指令月的读操作出现在第3周期。

因此，指令B读取的值不是指令A写回的值。

这种现象称做数据冲突（data hazard），是流水线冲突（pipeline hazard）的一种形式。

流水线冲突是由图7—2中虚线标示部分引起的。

程序员通常希望如果指令A出现在指令月之前，那么A计算的结果对月是可用的，因此这种行为可能是不被接受的。

计算机架构师已经通过很多方法来解决流水线冲突问题。

最简单的一种方法是显式流水线（explicit pipeline）。

利用这种方法，只是将流水线冲突记录下来，程序员（或编译器）必须对其进行处理。

以B读取A写入寄存器的数据为例，编译器会在指令A和B之间插入三个no—op指令（该指令什么都不做）来保证写操作出现在读操作之前。

这些no—op指令形成暂停流水线的流水线气泡（pipeline bubble）。

一种更加有效的方法是提供互锁（interlock）。

利用这种方法，当遇到指令B读取A写入寄存器的数据时，指令译码硬件会检测到冲突并延迟指令B的执行，直到A完成写回阶段。

对于这个流水线，B需要延迟三个时钟周期来等待A完成，如图7—4所示。

如果利用复杂一些的前向（forwarding）逻辑，可以检测到A写入的位置与B读取的位置相同，并且直接提供所需数据而不需要在读操作之前进行写操作，这样可以将延迟减少至两个时钟周期。

<<嵌入式系统导论>>

编辑推荐

《嵌入式系统导论:CPS方法》不同于大多数嵌入式系统的书籍着重于计算机技术在嵌入式系统中的应用，《嵌入式系统导论:CPS方法》的重点是论述系统模型与系统实现的关系，以及软件和硬件与物理环境的相互作用。

《嵌入式系统导论:CPS方法》适合作为高等院校相关专业“嵌入式系统”课程的教材或教学参考书。

<<嵌入式系统导论>>

版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:<http://www.tushu007.com>