中国西部嵌入式系统与单片机技术论坛

2005 学术年会

论 文 集

四川省单片机与嵌入式系统专委会　主编

北京航空航天大学出版社

需要全本请访问 www.ertongbook.com

内 容 简 介

本书为“中国西部嵌入式系统与单片机技术论坛 2005 学术年会”论文集，共收集论文 60 篇。书中内容反映了近年来嵌入式系统与单片机领域的一些新兴、前沿和热点技术，是国内各行业作者最新科研成果的总结。

本书包括 7 个专题：嵌入式软件设计与开发技术、嵌入式操作系统技术、嵌入式硬件与 FPGA、嵌入式网络技术、测控技术与数据采集、汽车电子应用技术以及其他嵌入式应用。

本书内容丰富，对现阶段从事嵌入式系统与单片机技术研究和产品开发的技术人员有重要参考价值，适合嵌入式系统与单片机业界专家、科技工作者、产品开发人员及高等院校教师和研究生等阅读和参考。

图书在版编目(CIP)数据

中国西部嵌入式系统与单片机技术论坛 2005 学术年会论文集/四川省单片机与嵌入式系统专委会主编. —北京：北京航空航天大学出版社，2005.11

ISBN 7-81077-738-6

Ⅰ.中… Ⅱ.四… Ⅲ.①微型计算机—学术会议—文集②单片微型计算机—学术会议—文集
Ⅳ.TP36-53

中国版本图书馆 CIP 数据核字(2005)第 135072 号

©2005，北京航空航天大学出版社，版权所有。

未经本书出版者书面许可，任何单位和个人不得以任何形式或手段复制或传播本书内容。侵权必究。

中国西部嵌入式系统与单片机技术论坛
2005 学术年会论文集

四川省单片机与嵌入式系统专委会 主编
责任编辑 芦潇静 冯 颖

*

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：787×1 092 1/16 印张：17 字数：435 千字
2005 年 11 月第 1 版 2005 年 11 月第 1 次印刷
ISBN 7-81077-738-6 定价：50.00 元

中国西部嵌入式系统与单片机技术论坛 2005 学术年会

2005 年 12 月　成都

指导单位：

中国电子学会

四川省信息产业厅

主办单位：

四川省电子学会

重庆市电子学会

陕西省电子学会

承办单位：

四川省电子学会单片机与嵌入式系统技术专业委员会

协办单位：

电子科技大学

北京航空航天大学出版社

《单片机与嵌入式系统应用》杂志社

四川省电子学会单片机与嵌入式系统技术专业委员会：

主任委员	陈德敷	成都力源单片机技术有限公司	总经理　高工
副主任委员兼秘书长	奚大顺	成都理工大学	教授
副主任委员	古天祥	电子科技大学	教授　博导
	熊光泽	电子科技大学	教授　博导
	吴家碚	成都纺织高专	系主任
	陆　坤	成都理工大学	教授
	李广军	电子科技大学	教授　博导
	张恩阳	四川长虹公司技术中心	副主任委员主任　高工

论文集编审组成员：

熊光泽	电子科技大学
古天祥	电子科技大学
李广军	电子科技大学
奚大顺	成都理工大学
陆　坤	成都理工大学
郑　萍	西华大学
谢明元	成都信息工程学院
林水生	电子科技大学
詹惠琴	电子科技大学
詹瑾瑜	电子科技大学

前　言

为推动中国西部嵌入式系统与单片机技术和产业发展，中国西部嵌入式系统与单片机技术论坛首届学术交流会将于2005年12月在成都举行。

本届学术大会由四川省电子学会、重庆市电子学会和陕西省电子学会主办，中国电子学会和四川省信息产业厅指导，四川省电子学会单片机与嵌入式系统技术专业委员会承办，电子科技大学、北京航空航天大学出版社和《单片机与嵌入式系统应用》杂志社协办。

本次大会筹备组先后收到学术论文79篇，经过论文集专家评审组审定，共录用论文60篇，并全文载入《中国西部嵌入式系统与单片机技术论坛2005学术年会论文集》。该论文集包含7个专题内容，由北京航空航天大学出版社正式出版发行。

本论文集是国内各行业作者的最新科研成果总结，内容新颖，覆盖面广，从多方位、多角度反映了21世纪关于嵌入式系统与单片机的新概念、新技术和新产品，展示了最新的应用成果和经验，是一本反映国内嵌入式系统与单片机技术最新技术综合信息的参考书，将进一步推动我国西部嵌入式系统与单片机研究和应用的迅速发展。

本次大会得到了业内专家、学者的广泛支持。由于本论文集评审和出版时间非常紧张，错漏之处敬请指正。在这里，感谢论文作者对本次大会的支持以及付出的辛勤劳动；感谢各位审稿专家在百忙之中审阅论文；感谢北京航空航天大学出版社的大力支持；感谢所有关心本论文集和热心支持本次会议的同仁！

我们相信，在大家的共同努力下，本次大会一定能够顺利召开，并预祝本次大会圆满成功，也祝愿各位与会代表在成都期间身体健康，心情愉快！

四川省电子学会单片机与嵌入式系统技术专委会

2005年11月4日

目 录

第一篇 嵌入式软件设计与开发技术

嵌入式系统软/硬件协同设计方法学研究 …… 詹瑾瑜 熊光泽(2)
嵌入式集成开发环境的研究与设计 …… 刘义才 雷 航(7)
一种嵌入式软件自动化测试系统的设计与实现
…… 谭 李 罗克露 高 峰 梁军峰 戴颖萌(11)
一种通用组件调度框架模型 …… 陈克力 罗克露(16)
LabWindows/CVI 环境下数据库应用技术 …… 俞 波 王厚军 黄建国(20)
交叉调试中的 ROM Monitor 方式的研究与实现 …… 邹楚雄 桑 楠 康涌泉(23)
嵌入式汇编测试中虚拟插桩技术的研究与实现
…… 谢瑞东 雷 航 范黎明 陈 祎 林文华(27)
虚拟逻辑分析仪在嵌入式仿真中的研究与实现
…… 王 鑫 罗克露 邢 茜 雷 剑 吴权国(30)
ATLAS 语言测试程序结构分析与应用 …… 何松涛 王厚军 付在明(35)

第二篇 嵌入式操作系统技术

Linux2.6 内核对象机制研究 …… 康涌泉 桑 楠 邹楚雄(41)
基于 QNX 的 GUI 应用程序的实现与应用 …… 谢晓娜 常政威 傅 鹏(45)
基于 Windows CE 强制访问控制技术的研究与实现 …… 李 欢 雷 航(49)
安全性加强的 Win CE 嵌入式操作系统 …… 陈志平 雷 航 杨 霞 李 欢(53)
航空电子系统中分区通信机制的研究 …… 江 维 熊光泽(58)
一种优化的嵌入式 Linux 中文化机制 …… 王 良 罗克露(62)
Linux 实时化技术研究 …… 肖皓月 罗克露(66)
进程间通信机制的分析与研究 …… 王钦骞 罗克露 龚 伟(71)
嵌入式多处理器实时系统的设计与实现 …… 陈 筠 桑 楠(77)
使用 AGB64LV01－QC 实现嵌入式 GUI …… 石 川 詹惠琴 古天祥(81)
操作系统的扩展技术研究 …… 马 红 桑 楠 江 维(85)

第三篇 嵌入式硬件与 FPGA

基于 32 位 MCU 的纯小数定点开平方快速算法 …… 沈家贵 李广军(89)
FPGA 在数字存储示波器时基电路中的应用 …… 金 映(94)
基于 ARINC659 总线的容错系统调试器设计 …… 黄晓智 罗 蕾(97)
基于 CPCI 的 GPIB 控制器接口设计 …… 侯 飞 黄建国 李 力(101)
基于 FPGA 的数字存储示波器的显示技术 …… 李 涛(104)
基于 VXI 总线 I/O 模块驱动器的开发 …… 冉 瑞 黄建国 李 力(107)
基于单片机的数控恒流源设计 …… 张 军 丁杰雄(111)
嵌入式多媒体播放器的低功耗设计研究 …… 林章渊 杨晨晖 晏 松(115)
应用 FPGA 实现手持式数字示波表的峰值采样 …… 孙彦明 王厚军 周玉鸿(121)

需要全本请访问 www.ertongbook.com

第四篇 嵌入式网络技术

嵌入式家庭网关的研究与开发 …… 蒋小洛 胡大可 朱海涛(125)
无线传感器网络的时间同步 …… 李宜安(130)
无线传感器网络技术 …… 唐秀辉 周小佳 李 玲 闫 斌(135)
无线终端平台的研究及下载平台的实现 …… 程 帅(140)
基于ARM的无线数据通信模块的实现 …… 林 华 罗环敏 习友宝(144)
基于FPGA片上PowerPC和VxWorks的以太网通信 …… 李 明 万 坚(148)
基于RS485通信网络的银行排队机系统设计与实现 …… 梁文海(153)
基于解析树的WAP浏览器的实现技术 …… 钟 遥 罗克露(157)
基于嵌入式Internet考勤机的设计与实现 …… 彭宣戈 朱 兵(162)

第五篇 测控技术与数据采集

PCI总线的转速采集卡设计 …… 赵继军 古天祥(165)
多路异速数据采集系统的通道设计 …… 唐 伟 詹惠琴 古天祥(169)
基于Labview和Epp并口的数据采集系统 …… 景少辉 詹惠琴 古天祥(173)
基于LPC2138与FPGA等效采样示波器的设计 …… 石明江 王厚军 周玉鸿(177)
基于PCI总线和DSP技术的数据采集处理卡的设计 …… 杨 浩 古天祥(182)
基于单片机和CPLD的大容量数据采集存储系统的设计 ……
…… 贾春华 詹惠琴 古天祥(183)

第六篇 汽车电子应用技术

车载嵌入式系统设计 …… 邓竹莎 雷 航(192)
单片机在汽车柴油发动机燃油控制中的应用 …… 熊 建 熊 燕 赖于树(196)
东风4型内燃机车防火报警系统 …… 胡 林 刘 平 彭宇明(201)
基于LH7A404的PDA交通管理系统终端设计 …… 罗东云 郑 萍(204)
数字温度传感器DS18B20在车道控制机中的应用 …… 钱怀风(208)
原油计量采集系统中上位机通信子系统的设计与实现 …… 徐 玲 何 巍(216)

第七篇 其他嵌入式应用

CGM－240128在原油计量采集仪中的应用 …… 何 巍 向守兵(221)
DSP与FPGA控制LCD显示的一种程序设计方法 …… 卫 刚(226)
VXI频谱分析模块温度补偿的实现 …… 肖 鹏 秦开宇(230)
多媒体教学平台远程监控系统的设计 …… 任丽云 朱清新(234)
滚动码技术及其在加密键盘中的应用 …… 梁 纯 缪燕子(239)
基于ARM芯片的电力监控系统RTU设计 …… 张丛耀 邓晴源 吴仲光(242)
基于Sunplus公司SPHE1001及Linux的视频播放系统 …… 李 宁 雷 霖(247)
嵌入式系统在超市物流中的应用 …… 李翔宇(251)
微机原理与接口技术实验教学平台设计 …… 余 梅 习友宝 古天祥(255)
自动加料机键盘系统的设计 …… 冯彦辉 陈 鸿(259)

第一篇

嵌入式软件设计与开发技术

嵌入式系统软/硬件协同设计方法学研究

詹瑾瑜 熊光泽
电子科技大学计算机学院,成都,610054

摘 要 随着微电子技术和计算机技术的飞速发展,嵌入式产品广泛应用于消费电子、智能家电、通信设备等多个领域。本文介绍了嵌入式系统现状,分析了今后的发展趋势,阐述了传统方法的缺陷,提出了一个新的设计方法学——嵌入式系统软/硬件协同设计方法学,并介绍了支撑新方法学的相关技术。

关键词 嵌入式系统,协同设计,重用,SoC,IP 核,软件构件,协同综合,测试调度

1 引 言

计算机自 1946 年诞生以来,经历了一个快速发展的过程。现在的计算机没有变成庞然大物,而是更加小巧玲珑。它们无处不在,功能强大,却又隐藏起来,这就是嵌入式系统。嵌入式系统是以应用为中心,以计算机技术为基础,软件硬件可剪裁,适用于应用系统对功能、可靠性、成本、体积、功耗有严格要求的专用计算机系统。嵌入式系统是将先进的计算机技术、微电子技术和现代电子系统技术与各个行业的具体应用相结合的产物,这一点决定了其必然是一个技术密集、高度分散、不断创新的知识集成系统。其广泛应用于国民经济和国防各个领域,发展非常迅速。有调查数据表明,嵌入式系统每年的增长速度约为 18%,大约是整个信息技术产业平均增长速度的 2 倍。目前,世界上大约有 2 亿台通用计算机,而嵌入式处理器大约有 60 亿个,嵌入式系统产业将会是 21 世纪信息产业的重要增长点。

2 嵌入式系统设计的发展趋势

系统设计是指将用户的客观需求转换成与之对应的技术系统的过程,是应用计算机技术建立起来的一个从现实世界到抽象世界的映射。随着嵌入式系统用户需求和相关技术的迅速发展,其趋势如下:

(1) 随着用户需求不断增加,嵌入式系统设计的复杂性不断提高,设计规模不断增大,设计对象由单机走向分布式系统;

(2) 随着嵌入式系统应用领域不断扩大,不同场合对系统设计的功能、功耗、实时性、面积等需求各不相同,其设计要求由单目标走向多目标;

(3) 随着嵌入式技术大量应用于手机等以电池供电的移动设备中,系统设计的功耗和体积限制要求不断增强,嵌入式产品的集成度越来越高;

(4) 随着半导体技术不断发展,硬件集成度不断提高,系统级芯片 SoC 的诞生使得将 CPU、存储器和 I/O 接口等 IP 核集成在单个硅片上成为可能,并且逐步成为当今嵌入式系统设计的主流;

(5) 随着嵌入式产品更新速度加快,系统设计周期不断缩短,新产品问世所需时间不断缩短,系统设计更加强调设计重用,软件系统更多地采用构件重用,硬件系统更多地采用 IP 核重用;

(6) 不同于在通用计算机系统上开发软件,嵌入式系统是一个软件和硬件并存的系统它们互相联系,相互补充,且互相制约,因此,设计时要从软件和硬件两个领域综合考虑。

3 传统设计方法的局限性

过去的嵌入式系统硬件部分设计容易,系统开发主要是在目标电路板上进行编程和交叉调试,所以那时的嵌入式系统开发很大精力都放在软件上,通常认为只要软件开发质量能够保证,就可以很好地满足整个系统的功能需求。传统的嵌入式系统设计流程如图 1 所示。系统软件和硬件划分完成后,首先进行硬件子系统的设计与实现,再进行软件子系统的设计与实现,整个系统的设计与实现基本是一个串行过程。

但随着嵌入式产品需求不断增加,硬件集成度不断提高,传统的嵌入式开发方法逐渐暴露出很多不足[1]:

(1) 软件和硬件的开发过程割裂,缺乏沟通。嵌入式系统在设计早期就分开进行,由于异构型系统软硬件复杂,这种设计过程使软件和硬件不能得到协调优化,导致设计效率低下。

(2) 设计自动化层次低。系统级设计由设计人员手工完成,设计的好坏依赖于设计人员的经验,而随着系统规模不断增大,设计复杂度将往往超出人的思维范围,导致设计不当。

(3) 设计过程串行化延长了设计周期。目前的设计流程主要采用先硬件后软件的开发模式,即实现了硬件的物理原型后才开始开发软件,因此,串行设计不能充分、及时地进行全系统综合考虑,导致设计失误增加,设计过程拖延。

(4) 缺乏设计重用支持。目前的嵌入式系统设计几乎都是从零开始的,没有很好地利用过去开发的成果,导致产品问世周期过长,市场竞争力下降。

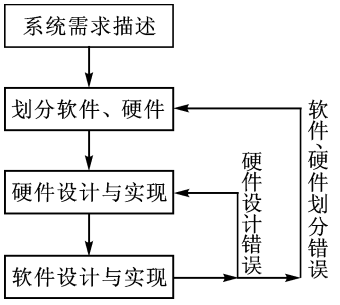


图 1　传统的嵌入式系统设计流程

图 2 所示为设计复杂性和设计生产率的增长趋势。由此可知,目前设计生产率的提高速度赶不上设计复杂度增加的速度,设计方法和工具成为制约嵌入式系统发展的瓶颈,嵌入式系统设计方法学亟须革新。

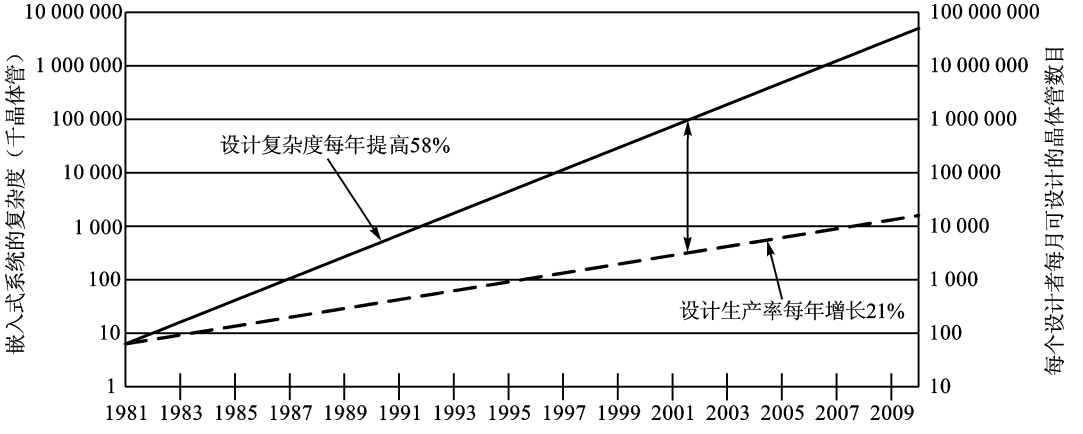


图 2　设计复杂度和设计生产率的增长趋势

4　软/硬件协同设计方法

针对嵌入式系统设计面临的问题与挑战,研究者们开始探索新的设计方法学——软/硬件协同设计(**Hardware/Software Co-Design**)方法学。软/硬件协同设计方法学的研究始于 20 世纪 90 年代初期。第一届 International Workshop on Hardware/Software Codesign 会议于 1993 年召开,它标志着软/硬件协同设计方法学的研究正式展开,软/硬件协同设计领域正式确立。

软/硬件协同设计不仅是一种设计技术,同时也是一种新的设计方法学,其核心问题是协调软件子系统和硬件子系统。

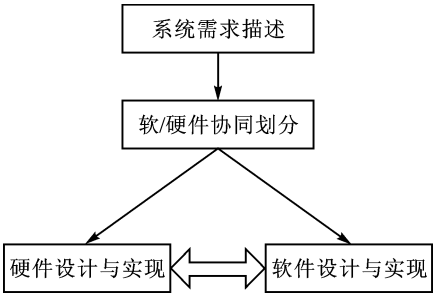


图 3　嵌入式系统软/硬件协同设计流程

与传统的嵌入式系统设计方法不同,软/硬件协同设计强调软件和硬件设计开发的并行性和相互反馈,其设计流程如图 3 所示,克服了传统方法中把软件和硬件分开设计所带来的种种弊端,协调软件和硬件之间的制约关系,达到系统高效工作的目的,软/硬件协同设计提高了设计抽象的层次,拓展了设计覆盖的范围。与此同时,软/硬件协同设计还强调利用现有资源,即重用构件和 IP 核,缩短系统开发周期,降低系统成本,提高系统性能,保证系统开发质量。

5　软/硬件协同设计方法学

目前,嵌入式系统软/硬件协同设计方法学研究还处于发展阶段,许多技术仍未成熟和实用化,但是其将给

嵌入式系统设计带来革命性的变化,极大地提高设计生产力。软/硬件协同设计方法学及其相关技术的研究意义重大。

5.1 软/硬件协同综合技术

嵌入式系统是软件和硬件一体化系统,系统中很多功能既可以由硬件来完成,也可以由软件来实现,硬件速度高,而软件成本低,这就需要权衡系统的时间、成本等性能指标之间的关系。设计嵌入式系统时,要分析不同的软件和硬件组合情况,决定系统各个模块是由硬件完成还是由软件完成,这是一个非常重要的工作。这个划分系统软件和硬件的过程被称为嵌入式系统软/硬件协同综合或者协同划分。目前,嵌入式系统软/硬件协同综合相关技术还不成熟,面向 SoC 的商业化自动综合设计工具尚在孕育之中,而嵌入式系统结构日益复杂,开发时间要求日益紧迫,使得软/硬件协同综合问题成为嵌入式系统软/硬件协同设计方法学首要解决的关键问题之一[2]。

现有的嵌入式系统软/硬件协同综合方法主要存在三方面的缺陷:第一,在传统的嵌入式设计方法中,划分软件和硬件的工作是由设计者手工完成的,划分结果的好坏依赖于设计者的经验,且效率低下,这个问题在系统设计开发过程中变得日益突出,因此需要一种自动化程度高的嵌入式系统软/硬件协同综合方法;第二,设计一个嵌入式系统不仅要考虑系统功能的需求,还要考虑其性能需求和限制条件,然而很多嵌入式系统软/硬件协同综合方法[3]只能评价一种或少数几种性能指标,因此需要一种能够同时评价多种性能指标(如价格、面积、功耗、时间性等)的嵌入式系统软/硬件协同综合方法;第三,随着嵌入式系统硬件日益复杂,集成度越来越高,SoC 已经成为设计嵌入式产品的主流技术,然而现有方法[4]并没有考虑到 SoC 设计的一些特殊要求(如 IP 核重用问题),因此需要一种适应于 SoC 设计的软/硬件协同综合方法。

5.2 软/硬件协同验证技术

嵌入式产品竞争越来越激烈,只有缩短产品问世周期才能在竞争中取胜,因此必须在系统设计和开发过程中尽早发现系统中存在的问题和错误。由于嵌入式系统自身的特点,系统不仅对功能有特殊要求,同时对各种性能指标(如功耗、面积、成本、时间性等)也都有严格的限制。如果在系统设计开发完毕才对各个指标进行评价,那么一旦某些指标不满足要求势必会推迟系统问世的时间,因此需要一种新的技术在系统设计开发完成以前的各个不同阶段根据系统性能指标要求对设计方案进行综合评价,以验证系统实际的合理性和可行性。

目前,嵌入式系统软/硬件协同验证的研究主要有两个方向:一是仿真验证,二是形式化验证。

仿真验证方法是在硬件开发以前用硬件描述语言(如 Verilog HDL、VHDL、SystemC、HandleC 等)完成硬件子系统的描述,用软件来搭建硬件以达到系统软件和硬件并行开发的目的,能够完成系统的联合调试,灵活地纠正软件和硬件的设计错误,避免了人力和物力的浪费;然而,硬件描述语言能够描述的硬件只是普通嵌入式系统硬件的一个子集,不适合描述大型的复杂系统,而且仿真系统与真实系统相比,功能和性能偏差也很大。因此,仿真验证方法通常只能验证一些简单系统的逻辑需求,对于功能复杂或实时性要求较强的系统就无能为力了。

形式化验证方法是建立被验证系统的数学模型,然后用数学方法证明被验证系统的正确性以及各种性能指标是否满足要求,形式化验证方法比仿真方法更精确,所以目前很多国内外专家都将研究重点转移到形式化验证方法上。卡内基梅隆大学计算机系的 Edmund M. Clarke 教授和他的学生从 20 世纪 90 年代就开始对硬件电路的形式化验证方法进行研究,并提出了一种基于有限状态机的模式检验理论[5]。近年来,模式检验理论也开始应用于嵌入式系统的软/硬件验证。断言抽象本来是一种验证纯软件系统正确性的形式化验证方法,首先由 Graf 和 Saidi 在文献[6]中提出,近年来很多学者对其进行了改进,同样使其应用到了软件和硬件并存的系统验证中。Luis Alejandro Cortes 等人提出了 PRES(Petri net based Pepresentation for Embedded System)形式描述方法[7],用 Petri 网对传统嵌入式系统软件和硬件进行描述,用 Alur 等人在文献[8]中提出的基于混合自动机模型对此模型进行验证。清华大学蒋屹新等人提出了基于 Petri 网的模型检验研究[9],但此方法是一种仅针对软件系统的形式化验证方法,没有关于硬件方面的考虑,无法直接有效地应用于验证嵌入式产品。综合分析上述各方法,它们都只是针对传统的嵌入式系统或特定的软件系统的验证方法,没有考虑到基于 IP 核和构件重用的嵌入式系统软/硬件协同验证相关问题,这将是今后研究的重点。

5.3 性能指标评价技术

嵌入式系统是以应用为中心,对系统功能和性能指标都有严格要求的专用计算机系统。准确评价各性能指标是保证嵌入式系统需求的必要条件。

5.3.1　成　本

系统成本由开发成本、生产成本等多种因素组成。不考虑开发成本，仅考虑系统实现软件和硬件的成本，是片面的、不现实的。成本参数有累加的特点，若将系统划分成几个子系统，则系统总成本等于各子系统成本之和。系统成本公式描述如下：

$$C(system) = \sum_{i \in system} C_i$$

5.3.2　面　积

随着嵌入式系统相关技术在消费电子、智能家电和通信设备等领域的日益广泛应用，嵌入式硬件的集成度越来越高，产品对硬件面积的要求也越来越严格，众多体积小、重量轻的嵌入式产品纷纷问世。系统硬件面积和各硬件组成部分直接相关，基本可通过各个硬件单元面积累加的方法获得。系统硬件面积公式描述如下：

$$S(system) = \sum_{i \in hardware\ (system)} S_i$$

5.3.3　功　耗

功耗已经被公认为是设计现代电子产品的重要指标。它是针对延长便携式产品电池的使用寿命而提出的，同时还要考虑到尽可能降低芯片封装和冷却装置的成本，提高系统可靠性以及降低环境因素的影响等。由于电池材料技术的发展不能满足系统日益增长的功耗需求，功耗问题已经成为系统设计过程中须重点考虑的性能指标之一。随着设计层次的不断提高，为降低系统功耗所作的响应处理也被提到了更高的层次。

系统功耗由静态功耗和动态功耗两部分组成，其计算公式描述如下：

$$P(system) = P_{static} + P_{dynamic}$$

$$P_{static} = \sum_{i \in system} P_{static}(i)$$

$$P_{dynamic} = \sum_{\substack{i \in system \\ Task \in Task(software(system))}} P(i, Task)$$

5.3.4　时间性

大多数嵌入式系统都是实时系统，它们对系统的时间性有着或高或低的要求，时间性已成为嵌入式系统设计的一个重要性能指标。时间性由硬件运行时间和软件运行时间组成：硬件运行时间是系统硬件运行所用的时间，可以通过将各个硬件单元运行时间进行累加得到；软件运行时间是系统中所有任务运行所用的时间，它与系统 CPU 的类型和任务数量有关。系统运行时间的公式描述如下：

$$T(system) = T_{hardware} + T_{software}\ T_{hardware} = \sum_{i \in hardware(system)} T_{hardware}(i)\ T_{software} = \sum_{\substack{i \in hardware(system) \\ Task \in Task(software(system))}} T(i, Task)$$

5.4　SoC 测试调度问题

随着微电子技术的飞速发展，集成电路制造技术不断进步，芯片的特征尺寸越来越小，系统集成度越来越高，使得制造 SoC 成为可能。为了满足现代电子产品快速增长的功能需求和日益紧迫的时间需求，设计者通常采用可重复使用的 IP 核来搭建 SoC，不仅能够简化设计过程，还有利于缩短产品制造周期和开发成本，因此，SoC 技术逐渐成为当前嵌入式产品设计的主流技术。

对于一个成熟的嵌入式产品，人们总是追求成本最小化，嵌入式产品成本一般包括设计费用、制造费用和包装测试费用。SoC 设计方法可以缩短设计周期，降低设计费用，还需要尽可能地降低测试费用，即缩短产品的测试时间。随着设计复杂度的增加，SoC 上重用的 IP 核数量越来越多，为了缩短芯片的测试时间，需要尽可能并行地测试芯片上的 IP 核。芯片上的 IP 核规模有大有小，各 IP 核所需要的测试数据端口也有多有少，而片上的封装引脚和测试总线数目是一定的，所以可以将几个较小的 IP 核输入/输出端动态分配到同一测试总线上，允许它们同时进行测试，因此，需要在 SoC 设计中进行测试调度。

SoC 测试包括 3 个子问题：测试访问机制(Test Access Mechanisum，TAM)设计问题、核封装设计问题和测试调度问题。

在 SoC 中，IP 核嵌入到芯片中作为芯片的一部分，因此无法从芯片引脚直接访问到 IP 核的输入/输出端口，必须要为 IP 核提供相应的测试访问通道。测试访问机制是将测试激励从芯片的输入引脚传送到核输入端口上，再将核输出端口的测试响应传送到芯片的输出引脚上。常用的 TAM 结构有并行直接访问结构、串行访

需要全本请访问 www.ertongbook.com

问结构、测试总线访问结构、可寻址测试端口访问结构等。

核封装提供一个IP核与TAM间的界面,其可提供多种操作模式,如正常工作模式、核测试模式、互连测试模式和分流模式。另外,核封装还能匹配核端口数量与TAM宽度。

测试调度是确定SoC中各IP核测试开始与结束时间的过程,其原则即尽可能缩减测试时间。目前,已经有很多国内外专家从事此问题的研究,并取得了一些初步成果。Chakrabarty[10]证明了SoC测试调度问题的一个NP-Complete问题,并建立了测试调度问题的整数线性规划(Integer Linear Programming,ILP)模型。虽然基于ILP的测试调度方法取得了很好的实验效果,但其计算复杂度高,不适于测试大规模SoC。目前,一些专家正在从事采用启发式算法解决SoC测试调度问题的研究。

6 结束语

嵌入式系统软/硬件协同设计方法学是一个非常广泛的研究课题,主要包括系统建模、软/硬件协同综合、设计功能和性能指标评价技术、软/硬件协同仿真、软/硬件协同验证、SoC测试调度技术等方面,且分为不同的设计层次。本文介绍了嵌入式系统现状,分析了今后的发展趋势,阐述了传统方法的缺陷,提出了一个新的设计方法学——嵌入式系统软/硬件协同设计方法学,并介绍了支撑该新方法学的相关技术。

近年来,国内外对嵌入式系统软/硬件协同设计方法学的研究开始重视,但仍处于发展状态,许多相关技术仍未成熟和实用化,没有成型的商业产品问世,这给我们带来了机遇和挑战。

参考文献

1 Jay K. Adams, Donald E. Thomas. The Design of Mixed Hardware/Software Systems. 33rd Design Automation Conference, 1996

2 Henkel J, Ernst R. An Approach to Automated Hardware/Software Partitioning Using a Flexible Granularity that is Driven by High-Level Estimation Techniques. Very Large Scale Integration(VLSI) Systems, 2001, 9: 273～289

3 Ernst R, Henkel J, Benner T, et al.. The COSYMA Environment for Hardware/Software Cosynthesis of Small Embedded Systems. Microprocessors and Microsystems, 1996, 20(3): 159～166

4 Daniel R, Peter S, Paul S. A Detailed Cost Model for Concurrent Use with Hardware/Software Co-Design. In Proc. of the 39th Conference on Design Automation, 2002: 269～274

5 Clarke E M, Grumberg O, Peled D. Model Checking. MIT Press, 1999

6 Graf S, Saidi H. Construction of Abstraction State Graphs with PVS. Proc. of the CAV'97, 1997: 72～83

7 Cortes L A, Eles P, Peng Z. Formal Coverification of Embedded Systems Using Model Checking. Proc. of the 26th Euromicro Conference, 2000: 106～113

8 Alur R, Henzinger T A, Ho P H. Automatic Symbolic Verification of Embedded Systems. IEEE Transactions on Software Engineering, 1996, 22(3): 181～201

9 Jiang Y X, Lin C, Qu Y, Yin H. Research on Model Checking Based on Petri Nets. Journal of software, 2004, 15(9): 1265～1276

10 Chakrabarty K. Design of System on Chip Test Access Architecture using Linear Programming. In Proc. 18th IEEE VLSI Symposium. Washington, D.C.: IEEE Computer Society, 2000: 127～134

Abstract With the increasing development of the technology in microelectronics and computer science, embedded applications are widely used in many fields, such as consumer electronics, intelligent electronic household appliances, and communication equipments. This paper introduces the present status of embedded system, analyses the trend in the future, presents the deficiency of the traditional design methods, proposes hardware/software co-design methodology of embedded systems, and give the supporting technologies.

嵌入式集成开发环境的研究与设计

刘义才　　雷　航
电子科技大学计算机学院,成都,610054

摘　要　集成开发环境是一个复杂的系统软件。针对这一系统软件,如何构建一个灵活、稳定的软件体系结构,是构建集成开发环境的关键所在。本文在软件总线的基础上,结合构件化思想,提出一种嵌入式集成开发环境的系统结构模型 EmIDE。以软件总线为核心,通过总线适配器接入集成开发环境相关的功能部件,整合形成一个完整的集成开发环境。本文最后设计实现了该模型的一个实例。

关键词　嵌入式,集成开发环境(IDE),软件总线,构件化

1　引　言

目前,嵌入式系统开发已经成为计算机工业的热点之一,遍布从尖端复杂的航空航天、精密仪器、医疗器材到时尚生活的信息家电、消费电子等众多领域。但由于嵌入式系统应用领域的多样性和应用个体的特殊性,使得嵌入式集成开发环境的集成工具也相应具有多样性和特殊性。因此,如何构建一个具有灵活、稳定软件体系结构[1]的集成开发环境,对于提高嵌入式系统的开发效率非常重要。目前,大多数的集成开发环境都集成了辅助开发的一系列工具,但是用户大多只能使用单一工具提供者提供的工具,不能随意定制自己所需的工具,选择面较窄,且这些集成机制大多没有采用面向对象的技术和框架,这些都制约了工具的选择面和开发环境的可扩展性。

本文提出了一个以构件化策略和软件总线[2]为基础实现的嵌入式 Linux 软件开发平台模型 EmIDE,将集成开发环境的各功能模块以构件化[3]方法进行划分和实现,并用软件总线技术实现的通用接口对各功能模块进行集成,形成了一个完整、统一的集成开发环境。

2　EmIDE 体系结构模型

2.1　软件总线和构件化

软件总线的概念首先出现在美国 Maryland 大学的 James Purtilo 和 Richard snodgrass 于 1994 年发表的一篇论文中。在此基础上,人们进一步提出了 OO 软件总线,即面向对象的软件总线。软件总线是相对于硬件总线而言的,它规定一个标准接口,把满足这个接口规范的功能构件像零件一样加载到软件中,实现功能构件的即插即用。

构件[4]是指一组封装的、规范的、可重用的软件模块,是组织系统的基本单位。它是以面向对象为基础发展起来的软件构件技术,严格来说构件技术仍然是面向对象技术。

软件构件化技术就是以软件功能对象的构件化思想为基础,把软件功能对象拆分成功能相对单一的构件,然后又像机器零件一样将构件进行组装,以提高软件的复用率,加快软件的开发速度,提高软件的开发质量。

以上表明软件构件化侧重于对软件进行“拆分”,而软件总线则侧重于对软件进行“连接”。本文正是基于软件构件化思想,将一个复杂的软件系统拆分成多个功能相对单一的功能对象,再以软件总线提供的通用接口对这些细化的功能对象进行有机连接,以形成一个功能完整的整体。与通常的设计方法相比,构件化使软件的功能结构更加清楚,实现相对简单,而软件总线也使集成开发环境能够更加灵活地集成工具。

2.2　EmIDE 子系统构件化与消息总线

以软件构件化思想为基础,可把 EmIDE 拆分成项目管理器、文件管理器等功能相对单一的子系统构件。但 EmIDE 的子系统与传统意义上的构件又有不同,它是个特殊的构件。在 EmIDE 中的子系统是一组相对独立,并且具有一定独立功能的程序,通过消息总线提供的接口实现构件间信息的传递和处理,能够像 COM 组件一样,当需要某个子系统时可加载该子系统,不需要时又可将其卸载,具有普通构件的特征。传统构件以简单的软件对象作为一个实现原型,复用也只是将一个简单的功能构件作为复用对象,而子系统则是一个非常复杂的功能对象,甚至本身就是一个可单独运行的应用软件,作为 IDE 功能的一部分加载到 IDE 系统上。子系统的特殊性就在于其本身可能就是一个应用程序,软件的复用级别也达到了应用软件的高度,所以说子系统是

一个功能复杂的大构件,是一个特殊的构件。

消息总线是软件总线的一种实例。其规定了一个标准接口,只要满足此接口标准,构件就能通过该接口以及消息总线和其他满足这个接口的构件实现信息传递。EmIDE 子系统间正是通过消息总线进行消息的传递。

2.3 EmIDE 体系结构模型

EmIDE 是基于层次消息总线的,构件间通过消息总线实现有机连接,如图 1 所示。EmIDE 体系结构模型分为三层:负责实现项目开发的各功能子系统(应用层),负责通信周转的消息总线(通信层),将子系统与消息总线连接起来的总线接口(连接层)。

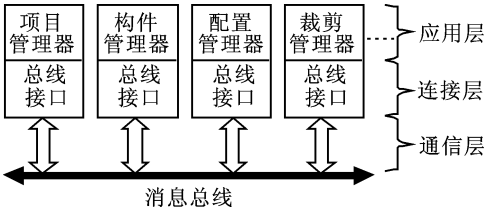


图 1 EmIDE 体系结构图

各子系统通过总线接口连接到消息总线上,并向系统登记自己敏感的消息,子系统根据需要通过总线接口向消息总线发送消息。

总线接口负责把顶层的各子系统和消息总线无缝连接起来,使它们形成一个整体。子系统发送消息时,把消息按照总线接口的要求(接口协议)进行封装,然后将消息发送到消息总线。与此相对,子系统接收消息时,把从消息总线发来的消息按照协议进行分解。

消息总线是子系统之间进行通信的中转枢纽。它负责接收来自子系统的消息,然后将此消息转发给相应的处理子系统,并从处理子系统以消息的形式将处理结果返回给最先发送消息的子系统。在此过程中,子系统与消息总线之间的信息传递始终以消息的形式进行传递。

从 EmIDE 的体系结构可以看出其所具有的一些特点:

(1) 体系结构和消息传递具有层次性,其在体系结构上分为构件层、连接层和通信层三层;

(2) 每个子系统实现 EmIDE 平台的一项或多项功能,并通过消息总线把各子系统整合在一起构成 EmIDE 整体;

(3) 每个子系统都是一个相对独立的整体,可以并发运行,但功能上也具有一定的依赖性;

(4) 软总线技术使其具有很强的扩展性。

利用 EmIDE 体系结构的特点进行集成环境软件开发,具有如下优势:

(1) 层次清晰,结构简单;

(2) 各子系统使用统一的接口规范,使软件的扩展性得到大大提高;

(3) 可根据需要定制功能子系统,可任意进行增删(核心功能子系统除外,如项目管理子系统、信息库等);

(4) 强大的工具集成功能,可根据需要将工具包装成子系统加入 IDE 系统;

(5) 消息总线为各子系统的消息进行服务,能够保证其获得统一的数据信息。

3 EmIDE 要实现的重要模块

3.1 消息总线

消息总线实现的主要功能包括:

(1) 创建唯一的消息总线对象,以便子系统能够获得统一的消息服务;

(2) 核实子系统是否加载,若没有加载,对其进行加载并登记加载信息;

(3) 卸载子系统,当消息总线退出时要将所有加载的子系统卸载并登记,当子系统退出时也要把相应退出信息交给消息总线并登记;

(4) 接收来自子系统的消息,经过分析,将该消息转发给相应的子系统进行处理,并通过回调接口,返回处理结果,最后把结果再返回给最先发送消息的接口。

EmIDE 采用 COM 组件[5]技术实现消息总线接口。重要接口和数据结构的定义如下:

```
Interface IMBAMsg:  public Idispatch//消息接口
{
    [id(1), helpstring("method RecMsg")] HRESULT RecMsg ([in] CMessage * Msg);
    [id(2),helpstring("method SendMsg")] HRESU L T SendMsg ([in] CMessage * Msg);
```

```
    …
};
interface IMBMgr:public Iunkown //子系统管理接口
{
    [id(1), helpstring("method Register")] HRESULT Register (…);//注册子系统
    [id(2), helpstring("method UnRegister")] HRESULT UnRegister (…);//注销子系统
    [id(3), helpstring("method Load")] HRESULT Load(…);//加载子系统
    [id(4), helpstring("method UnLoad")] HRESULT UnLoad(…);//卸载子系统
};
```

3.2　总线接口

总线接口定义了一组在子系统和消息总线之间进行通信的接口,及组件数据交互必须遵守的协议。这里采用 COM 组件的连接点技术,即在 COM 组件上定义组件的一个引出接口,在 COM 组件的服务器端仅定义此引出接口及其相关的接口函数,而接口的实现则分别放在组件的客户端即各子系统,这样既满足组件接口通用性的要求,也可满足子系统特殊性的要求。使用连接点技术[6]的总线接口实现方法如图 2 所示。

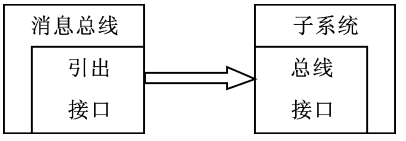


图 2　连接点技术实现的总线适配器接口

3.3　EmIDE 子系统

EmIDE 总的功能表示为 Func(EmIDE) = {Func(Child1),…,Func(child$i$)},其中每个 Func(Child$i$)表示一个子系统所代表的功能。对于一个构造良好的软件,子功能模块的划分须满足以下两点要求:

(1) Func(EmIDE)由所有的 Func(Child$i$)(i=1,2…)累加而成,即

$$Func(EmIDE) = \sum Func(Child\,i)\ (i,j = 1,2\cdots);$$

(2) 任意子系统间功能的交集为空,即

$$Func(Child\,i) \cap Func(Child\,j) = \Phi(i,j = 1,2\cdots);$$

对 EmIDE 的功能进行划分,可以分为以下四个主要的功能子系统。

(1) 项目管理子系统:负责项目方面的功能,包括项目文件管理、项目开发、项目环境配置等。

(2) 构件管理子系统:以构件的形式管理嵌入式 Linux 操作系统配置选项,使开发者可直观地裁剪 Linux 操作系统。

(3) 信息库管理子系统:作为一个数据库管理系统,负责对信息库进行管理。

(4) 版本控制子系统:负责协调应用程序开发。在版本服务器上为项目生成开发日志库,多个用户可以通过网络在版本服务器的协调下同时对该项目进行开发。

3.4　IDE 辅助工具集成模块

消息总线结构的主要优点是强大的工具集成能力,它能够很轻易地集成第三方的开发工具。工具集成模块负责对第三方工具进行集成,其过程主要由以下三个步骤完成:

(1) 对工具软件进行注册;

(2) 在配置管理器中选择该工具;

(3) 在项目管理器的菜单上自动为该工具添加一个菜单选项。

这样,该工具就集成到了集成开发环境中。可通过该项目管理器为该工具提供的菜单选项调用该工具。

4　应用实例

基于 EmIDE 结构模型实现了一个嵌入式集成开发平台,运行于 Windows 环境下,其大致框架如图 3 所示。项目管理子系统是用户进行软件开发调试的模块。其中,资源编辑管理器用所见即所得的方式实现软件;信息管理库子系统负责信息库的管理;构件管理子系统负责 Linux 内核裁减配置管理;信息库负责 IDE 环境数据信息的存储;消息总线的功能即将以上几个子系统连接起来形成整体。本平台已应用于“基于 ARM 的嵌入

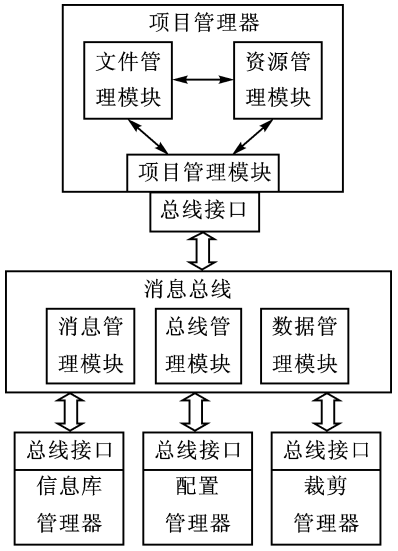


图 3 EmIDE 开发环境结构示意图

式 **Linux** 集成开发平台”科研开发课题，证明其能够灵活地集成所需的工具，且不同工具能够并发协调稳定地工作，取得了设计的预期效果。

5 结束语

本文以构件化和软件总线技术为基础实现的集成开发环境，并在此基础上提出了嵌入式集成开发环境 EmIDE 的体系结构和工作原理。与普通的 IDE 模型开发方法相比，EmIDE 具有以下特点：

(1) 以构件化的思想对整个 IDE 进行功能划分，并由此形成功能相对独立的 IDE 子系统；

(2) 消息总线作为各子系统的连接器，给开发环境集成带来了灵活性，即可以有选择地加载功能子系统；

(3) 集成环境提供了统一的数据源和标准的事件过程，提高了集成开发环境的通用性和交互能力。

在此框架上实现的 EmIDE 的项目管理器、构件管理器、用户管理器和裁剪管理器等子系统，接下来要实现基于 EmIDE 框架的快速应用开发(RAD)工具——程序产生器和可视开发工具，使 EmIDE 的功能更加强大。

参 考 文 献

1 万建成，卢雷．软件体系结构的原理、组成与应用[M]．北京：科学出版社，2002

2 Purtilo J M．The Polylith Software Bus[J]．ACM Transaction on Programming Languages and Systems，1994，16(1)：151～174

3 Component Development Strategies[EB/OL]．http://www.cutter.com/consortium

4 Welsy Addison．Component Software：Beyond Object－Oriented Programming[Z]．1998

5 潘爱民．COM 原理与应用[M]．北京：清华大学出版社，1999

6 Tom Armstrong Ron Patton－ATL Developer's Guide．2nd Edition

7 郭江，寥越虹．工具集成的框架及工具间的关系．计算机研究与发展，1995，1(32)

Abstract Integrated developing environment (IDE) is complicated system software, and the key is how to build agile and stable software architecture. In this paper, we combined software bus and the idea of component to form an embedded integrated developing environment model－EmIDE. The software bus, which is the kernel, plus corresponding IDE function components using bus adapter formed the integrated IDE. At last, we will propose an example of this model.

一种嵌入式软件自动化测试系统的设计与实现

谭　李　　罗克露　　高　峰　　梁军峰　　戴颖萌
电子科技大学计算机科学与工程学院,成都,610054

摘　要　嵌入式软件的应用与开发是当今计算机软件发展领域的一个热点,其软件质量也变得尤为重要。首先提出了一种针对C语言的嵌入式软件自动化测试系统的解决方案,然后给出了嵌入式软件测试从源代码分析到最后生成测试报告的实现过程。

关键词　嵌入式软件,远程处理,Lex和Yacc,测试用例,覆盖测试

1　引　言

嵌入式软件作为计算机软件的重要分支,在消费电子产品、交通系统、工业控制、医疗设备和军事领域等方面应用得越来越广泛。嵌入式系统安全性的失效可能会导致灾难性的后果,即使是非安全性系统,由于大批量的生产也会导致严重的经济损失。这就要求对嵌入式系统,包括嵌入式软件,进行严格的测试、确认和验证。随着越来越多地使用软件和微处理器控制各种嵌入式设备,对日益复杂的嵌入式软件进行快速、有效的测试显得愈加重要。

由于嵌入式软件自身的一些特点(实时性、嵌入性和专用性等),难以用传统的方法对其进行测试,因此对测试技术提出了更新的挑战。目前国外的Panorama、Logiscope和ParaSoft等都曾做了一些卓有成效的工作,但也还存在很多不足,而国内尚无比较成熟的产品出现。

2　系统概述

本系统主要研究对嵌入式软件进行分析测试的先进技术和质量评价的科学方法,提供一种方便的、自动化的测试过程。采用的是Client/Server结构,主要由测试服务器、测试数据库、工作终端、测试终端和目标机等几部分组成。

2.1　测试系统总体功能

(1)测试工程的层次管理结构:测试工程→测试集→测试用例;
(2)支持并行的团队测试工作,提高测试效率;
(3)支持嵌入式软件的集成测试、单元测试、黑盒测试及白盒测试;
(4)从cvs源码库中下载被测源码,按照C语言标准规范对被测源码进行自动分析;
(5)自动生成测试用例,并自动完成用例源码的编译和链接;
(6)自动执行指定的单个或多个测试用例,支持测试用例的并行执行;
(7)与Lambda GCP工具集成,支持覆盖测试,自动统计覆盖率并提供图形化显示;
(8)支持被测软件系统函数静态调用关系的图形化显示;
(9)自动生成测试报告。

2.2　测试系统组成

测试系统硬件和软件组成结构分别如图1和图2所示。

测试服务器:处理用户对测试工程、测试集和测试用例管理的操作,处理用户并发的测试操作,管理源码分析、测试用例生成、测试用例运行与数据采集各模块,以及和数据库之间的交互。

工作终端:即客户端,包含测试工程、测试集和测试用例管理逻辑、用户界面管理、相关的数据库操作逻辑,以及与测试系统后台服务器通信进行用户指定的测试操作。

测试终端:与目标机之间的通信,管理测试用例的运行、测试结果与覆盖数据的采集。

目标机:嵌入式的x86体系结构硬件平台,可直接启动TRA(不通过BIOS),测试用例的运行环境。

需要全本请访问 www.ertongbook.com