文章编号:1001 - 9081(2005)12Z - 0364 - 02

基于虚拟机的嵌入式软件测试环境

王轶辰,刘 斌,钟德明
(北京航空航天大学 工程系统工程系,北京 100083)
(buaa_wangyichen@yahoo.com.cn)

摘 要:嵌入式软件的系统测试需要系统测试环境的支持,而当前使用最多的专用测试环境,因为可移植性较差,极大的限制了测试用例的可重用性。文中提出了一种全新的基于测试虚拟机的软件测试环境,并且详细介绍了测试虚拟机的原理以及利用微内核模式对测试虚拟机的体系结构进行设计。

关键词:嵌入式软件测试环境;可重用性;可移植性;测试运行时系统;测试虚拟机;微内核
中图分类号: TP311 **文献标识码**:A

0 引言

随着嵌入式系统在许多重要领域日渐广泛地使用,嵌入式软件较高的失效率逐渐成为困扰其使用者的一个严重问题。据统计,嵌入式系统的运行失效中有 75% 是由软件失效所引起的。目前,通过进行各种类型的软件测试来提高嵌入式软件的质量和可靠性已经逐渐成为人们的共识,在各阶段的软件测试活动中,系统测试是一种全面而且有效的方法,它是对嵌入式软件和其运行的硬件环境集成之后的嵌入式系统进行的一种动态黑盒测试。嵌入式软件的系统测试通常利用软件仿真来模拟被测系统的输入/输出从而驱动被测软件的运行以达到对嵌入式系统的测试。这种自动化的测试需要有测试环境的支持,而且测试环境的优劣是影响软件测试质量的一个重要因素。

1 嵌入式软件的系统测试环境

嵌入式软件的系统测试环境,即能对嵌入式软件进行自动的、实时的、非侵入性测试的闭环测试系统,它能够逼真的模拟被测软件运行所需的真实物理环境的输入和输出;能够向被测软件发送输入数据,驱动被测软件运行;同时接收被测软件的输出结果。在系统测试环境的设计与实现中包括三个关键技术:测试描述语言、测试运行时环境和测试环境接口。

1) 测试描述语言用来描述测试用例,即测试输入的组织与预期的测试输出,用测试描述语言描述的测试用例称为测试脚本。

2) 测试运行时环境提供了测试脚本运行时所需的一切资源。如果测试描述语言采用解释执行的方式,则测试运行时环境中包括测试脚本解释器,测试脚本经过解释器处理后得到执行,完成测试。

3) 测试环境接口是连接测试环境与被测系统的真实I/O设备。因为系统测试是一种黑盒测试,所以测试环境只有提供与被测系统相一致的接口类型,才能实现相对全面的测试。

任何一个系统测试环境都可以看成是对这三项关键技术的一种实现。

目前,大多数嵌入式软件的系统测试采用的都是专用测试环境,即专门针对某一个或某一类嵌入式系统而设计。在专用测试环境中通常没有一种明确的测试描述语言,测试脚本、测试运行时环境以及测试环境接口与被测软件紧密相关。当测试不同软件时,需要重新构建测试环境,这就导致了大量具有重用价值的测试脚本无法在新的测试环境中运行。而且,专用测试环境往往需要与嵌入式软件的研发过程同时进行,它与被测系统之间很强的关联性使得被测系统进行的任何修改或升级都可能导致测试环境大规模的改动,甚至可能需要重新设计测试环境。所以测试脚本的可移植性低和测试环境通用性差是目前专用系统测试环境在使用中面临的主要问题。

2 测试虚拟机的提出

虚拟机是软件运行时环境的一种实现方法,它能够很好的提供应用程序的可移植性。当需要建造一个能够支持应用程序执行的基础设施以便满足一大类应用程序的使用时,虚拟机是一种很好的解决方案。因此,基于虚拟机原理设计系统测试环境,可以满足测试脚本的移植性需求以及系统测试环境的通用性需求。

图 1 基于测试虚拟机的测试环境

测试虚拟机是测试运行时环境的一种实现。它提供了测试脚本在测试执行时所需要的一切资源。基于虚拟机的系统测试环境提供测试描述语言及其解释器,凡是使用该语言编写的测试脚本,都可以在测试虚拟机中经过解释器的处理后得到运行。而测试虚拟机经过简单的移植后,即可适应相应的测试环境接口,实现不同被测系统的测试要求。基于测试虚拟机的系统测试环境如图 1 所示,它由测试开发环境、测试虚拟机、测试环境接口三个部分构成。

采用虚拟机模式设计测试运行时环境的最主要优势有以下两点:1)利用测试描述语言所描述的测试脚本,可以在针

收稿日期:2005 - 03 - 04;修订日期:2005 - 05 - 31
作者简介:王轶辰(1977 -),男,博士研究生,主要研究方向:嵌入式软件测试、嵌入式软件自动化测试.

对不同被测系统而构建的测试虚拟机中运行,在对不同的被测系统进行测试时,测试脚本可以进行简单修改或根本不进行任何修改就能够在移植后的测试虚拟机上运行。2)测试虚拟机很好的可移植性可以极大的提高测试环境的通用性,使得对不同的被测系统进行测试时只需要对测试虚拟机进行很小的改动即可。

然而,传统的虚拟机模式是以牺牲运行效率和较大的内存消耗为代价来优化应用程序的可移植性。为了避免测试虚拟机存在相同的问题,可以采用以下两点措施:

1)通过把测试虚拟机建立在实时操作系统上提供系统测试环境的实时性,以满足具有实时性要求的嵌入式软件的测试需求。

2)通过良好的体系结构设计,提供测试虚拟机的可剪裁性,以解决可能存在的内存消耗问题。另外,合理的体系结构设计还可以增强测试虚拟机的可扩展性和可移植性,提高系统测试环境的通用性。

综上所述,测试虚拟机可以较好的解决专用测试环境中测试脚本可移植性和测试环境通用性较差的问题。

3　测试虚拟机的体系结构

软件体系结构设计是测试虚拟机规范中的重要部分,它提供了虚拟机的实现框架。为了实现其本身良好的可移植性,测试虚拟机采用微内核模式进行设计。

如图2所示,测试虚拟机由三个主要部件组成:微内核、外部服务和内部服务。

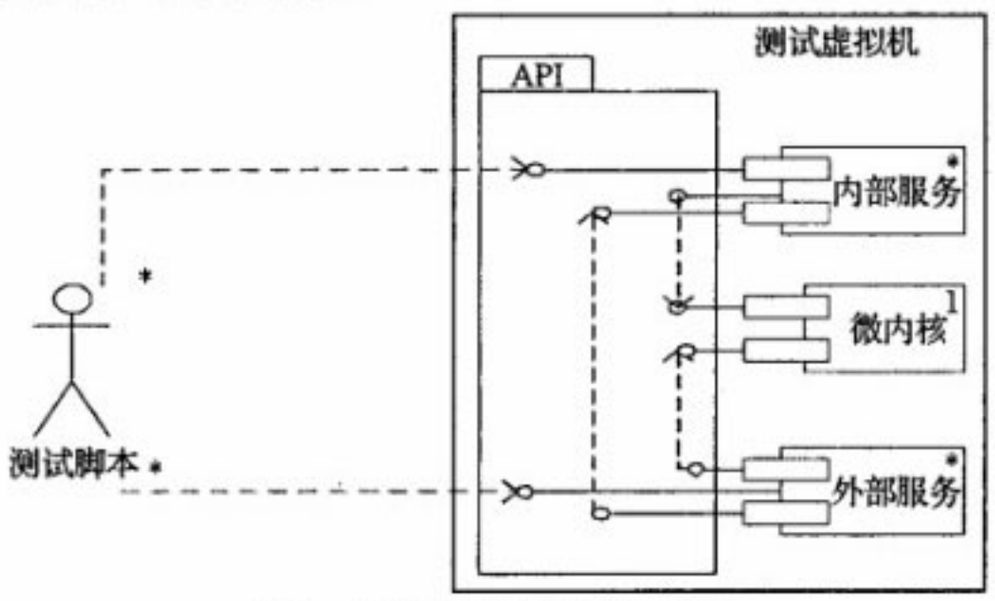

图2　测试虚拟机的微内核结构

3.1　测试虚拟机微内核

微内核是测试虚拟机的主要组件之一,它实现的是一些支撑测试虚拟机的原子服务。测试虚拟机所需要的内部和外部服务全部或部分地建立在微内核所提供的基本服务之上。微内核是测试虚拟机体系结构中最稳定的部分,针对不同被测系统进行虚拟机移植时,微内核几乎不需要作任何的修改。微内核中包括的组件有:测试任务管理组件、测试任务调度组件以及任务间通信组件。

3.1.1　测试任务管理组件

测试任务管理组件的功能是管理和控制测试虚拟机中的所有测试任务。管理与控制测试任务的依据是一个测试任务链表,表中记录每一个测试任务的所有相关信息,包括:任务名称、任务的类型(周期型或异步型)、任务的优先级、任务的触发机制、任务的启动时间和销毁时间以及任务的运行周期等。任务管理组件向外部服务和内部服务提供一组接口函数,例如任务注册和任务销毁等。其他服务可以通过任务注册接口向微内核注册各自的测试任务。任务管理组件根据任务注册信息进行任务的创建和销毁等管理工作。

3.1.2　测试任务调度组件

测试任务调度组件的功能是在测试运行期间决定在某一时刻执行什么测试任务。测试任务在运行期间主要有三个状态:运行状态、就绪状态和阻塞状态。

测试任务调度组件在缺省的情况下,使用"基于任务周期的可抢占式"调度机制,即调度程序根据任务的工作周期,在每个任务的工作周期到来之时才激活这个任务,相同周期的任务则根据优先级来调度。当有异步型任务要执行时,调度程序会优先处理它,之后再恢复周期型任务的调度。另外,测试任务调度组件还提供了调度机制的选择机制,其他服务可以通过调度组件提供的接口函数选择是否采用可抢占式调度。

3.1.3　任务间通信组件

微内核中的通信组件为测试任务之间以及测试任务与被测软件之间提供通信机制。由于测试环境通过真实的 I/O 接口与被测系统相连,测试数据最终要经过硬件接口被送往被测系统,而不同的被测系统具有不同类型的 I/O 接口,所以测试环境接口的驱动程序是测试虚拟机进行移植时有可能要进行更改的部分。为了测试虚拟机的可移植性,通信组件采用分层结构的设计,目的就是要把这些具体的硬件信息封装起来达到对测试任务的完全透明。分层结构的通信机制提供一组通信函数,供内部和外部服务组件在传输数据和消息时使用,而在使用过程中服务组件并不需要知道数据和消息传输所使用到的硬件信息。

3.2　测试虚拟机内部服务

虚拟机的内部服务提供了一组依赖于微内核的可选服务,客户可以通过一组接口对其访问。内部服务是对测试虚拟机微内核功能的扩展。

测试虚拟机的内部服务在虚拟机移植时可以进行相应的修改,但是多数情况下是对可选服务集的取舍。测试虚拟机中的内部服务利用微内核提供的原子服务实现了测试运行时的多项基本功能,包括:测试显示与监控服务、测试数据收集服务、测试模型配置服务和测试脚本解释器等。

3.2.1　测试显示与监控服务

显示监控服务包括测试数据的实时显示和测试状态的实时监控。

显示服务由测试人员配置使用,测试开始后,对被测系统的输入数据和从被测系统接收的测试结果数据都被存入到数据库中,显示服务根据用户的配置,从数据库中取出相关的数据并显示在图形界面上。用户的配置信息包括要求显示的数据项名称、数据的显示周期以及数据的显示格式等。

测试状态的实时监控根据测试运行时的状态为测试用户提供信息,包括:测试时间、测试状态(正在进行或是暂停状态等)以及测试是否出现异常等。

3.2.2　测试数据收集服务

测试数据收集服务可以保存和备份测试中的数据,包括:被测系统的输入数据和从被测系统接收的测试结果数据等。这些数据的来源各不相同,有的来自用户的测试脚本,有的来自测试模型,还有的来自被测系统软件,数据收集服务就是要把所有这些数据统一地收集到数据库中,以便满足其他服务的数据需求,例如显示监控服务中要求显示的数据等。

3.2.3　测试模型配置服务

测试模型配置服务的主要功能是根据测试者的需求对测

试模型进行配置,配置内容包括:

1) 为测试模型变量分配物理空间;

2) 确定模型变量之间的 I/O 类型。模型与模型之间以及模型与被测系统之间的可以有很多种 I/O 接口,例如 MIL-STD-1553B、ARINC429、A/D、D/A、串口以及网络等。

3) 确定模型变量所使用的 I/O 接口的物理信息。例如,如果模型变量通过 ARINC429 发送,那么要指出所使用的通道号,如果通过 MIL-STD-1553B 总线传输,则要指出终端地址、子地址及数据字长等。

4) 在测试虚拟机的分布式实现中,还要确定每一个测试模型运行的节点。

3.2.4 测试脚本解释器

脚本解释器专门针对测试描述语言而设计[3]。解释器由两部分组成:主控程序和解释程序。主控程序的作用在于对测试脚本进行预处理,生成按时间排列的任务表,然后根据实时调度时钟信号和条件判断(如反馈判断)触发解释程序的调用。解释程序的主要功能是在被主控程序按时钟信号触发后,加载并解释执行脚本代码,以达到对测试模型的控制。

3.3 测试虚拟机外部服务

外部服务又称为个性服务,它使用微内核和内部服务所提供的服务实现了以具体应用为基础的面向用户的服务。它提供了一组基于微内核和内部服务的可选服务,并通过一组接口提供给客户访问。测试虚拟机的外部服务是进行虚拟机移植时改动最大的部分,它集中体现了不同被测系统的测试特点。

测试虚拟机中的外部服务主要由一系列的测试模型构成。测试模型是对被测软件的外围环境的计算机仿真,外围环境包括被测软件所嵌入的嵌入式系统的交联系统和被测软件接收的传感器信号等。测试模型产生并发送测试数据,驱动被测软件运行,然后基于模型进行软件测试结果的验证。测试模型由测试脚本直接控制,它与测试脚本之间有较强的关联。事实上,移植测试虚拟机时测试模型要根据测试脚本的要求重新编写,测试虚拟机提供了测试模型的编写规范,并专门为模型提供了图形化的开发环境,为测试模型的移植提供了极大的便利。

4 结语

为了解决嵌入式软件系统测试过程中测试重用性和测试环境通用性差的问题,本文提出了基于测试虚拟机的嵌入式软件测试环境设计思想,并且利用微内核模式对测试虚拟机的体系结构进行了设计,实现了测试环境良好的可移植性,从而在一定程度上提高了测试的重用性和测试环境的通用性。

参考文献:

[1] BLUNDEN B. 虚拟机的设计与实现——C/C++. 杨涛, 等译. 北京: 机械工业出版社, 2002.

[2] DOUGLASS BP. 实时设计模式-实时系统的强壮的、可扩展的体系结构[M]. 麦中凡, 陶伟, 译. 北京: 北京航空航天大学出版社, 2004.

[3] 殷永峰, 刘斌, 陆民燕. 实时嵌入式软件测试脚本技术研究[J]. 计算机工程, 2003, (1).

[4] VENNERS B. 深入 Java 虚拟机[M]. 第 2 版. 曹晓刚, 蒋靖, 译. 北京: 机械工业出版社, 2003.

[5] BUSCHMANN F, MEUNIER R, 等. 面向模式的软件体系结构·卷 1: 模式系统[M]. 贲可荣, 郭福亮, 赵皑, 等译. 北京: 机械工业出版社, 2003.