文章编号：1672-2892(2011)01-0101-04

VI 仿真环境下嵌入式软件测试研究与应用

漆莲芝，谢 敏，张 军

(中国工程物理研究院 电子工程研究所，四川 绵阳 621900)

摘 要：嵌入式软件测试与通用软件测试有相通之处，也有很大差异，嵌入式系统测试与系统软硬件平台关系密切且对测试环境、测试方法都有特殊要求。测试用例设计是软件测试的核心，决定了软件测试的最终效果。在软件测试环境能更方便地进行测试用例注入，且软件测试环境直接影响着软件测试的效率。本文论述了嵌入式软件测试技术，分析了采用 UML 生成测试用例的方法，论述了采用 VI 技术搭建仿真测试环境的原理。采用所述方法实现了某型号分系统嵌入式软件测试。

关键词：嵌入式软件测试；统一建模语言；VI 技术；仿真环境

中图分类号：TN91；TP311.5 **文献标识码：**A

Embedded software testing under VI simulation environment and its application

QI Lian-zhi, XIE Min, ZHANG Jun

(Institute of Electronic Engineering, China Academy of Engineering Physics, Mianyang Sichuan 621900, China)

Abstract: There exit similarities as well as a great difference between embedded software testing and common software testing. Embedded system testing has close relationship to system software & hardware platform, and also has special requirements on test environment and test methods. Test case design is the core of software testing, which determines the effect of software testing. Software testing environment can be more easily injected for test case and directly affects the efficiency of software testing. This paper discusses embedded software testing techniques, analyzes the method to generate test case by using UML, presents the principle about adopting VI technique to construct simulation test environment. A certain embedded software system has been tested by using the methods proposed in this paper.

Key words: embedded software testing; Unified Modeling Language; Virtual Instrument; simulation environment

软件是人思维的逻辑产品，而人的逻辑思维很难达到完美无缺的境界，存在软件缺陷在所难免，同样对于嵌入式软件也是如此[1]。软件测试是通过分析软件系统或者软件系统中各个组件来检测已规定的行为与所观察的软件现有行为之间差异的一个过程[1]。嵌入式软件测试也必须遵循软件测试的基本规则，嵌入式软件与普通软件无论是在开发模式还是调试方法方面都有其自身的特点，且嵌入式软件运行是实时的，运行过程中嵌入式软件、硬件以及外部设备频繁交互，嵌入式软件系统本身对性能的要求也高，因此，所有这些特点决定了嵌入式软件测试是一类最难的测试。嵌入式软件测试是控制嵌入式软件的软件质量、提升软件可靠性的一种有效手段，图 1 是嵌入式软件测试过程示意图。如图 1 所示，嵌入式软件测试借助仿真测试环境向被测嵌入式软件注入测试用例并驱动被测软件按照已规定的流程进行工作，通过仿真测试环境监测被测软件的运行状态和输出结果，最后测试人员通过分析软件的运行状态和输出结果与预期结果之间的差异，找出软件问题。

1 测试用例生成

测试用例设计是软件测试的核心，决定了软件测试的最终效果，也是软件测试活动中最重要的部分之一。只

收稿日期：2010-05-27；修回日期：2010-07-18

有采用合适的测试用例设计方法，才能设计出高效和测试充分的测试用例。UML 提出了一套 IT 专业人员期待多年的统一的标准建模符号，随着 1996 年对象管理组织(Object Management Group)对它的采用，UML 成为了一个被广泛接受的工业标准[OMG03a] [OMG04] [RJB05][2]。Cavarra 等人结合测试意图、测试约束及一些覆盖准则提出了采用 UML 对测试过程进行建模[3]，图 2 是采用 UML 进行测试用例设计的过程，采用 UML 构造被测软件 SUT(Software Under Test)的模型，生成所有的测试场景，根据测试场景对应的所有输入生成测试用例。

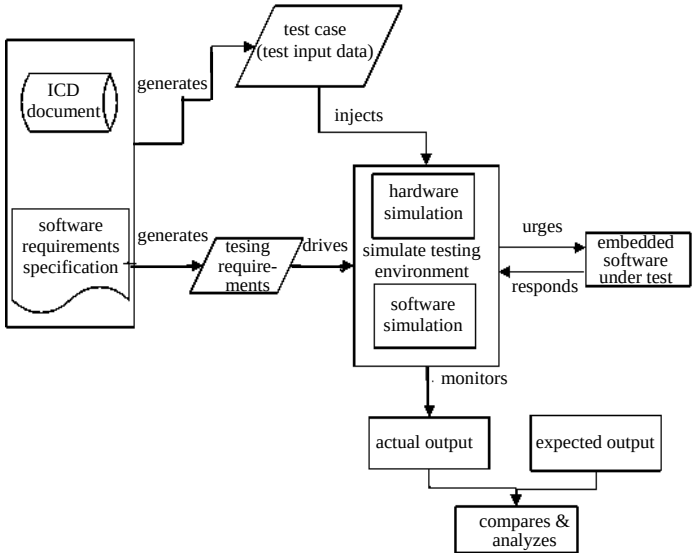

Fig.1 Embedded software testing
图 1 嵌入式软件测试

采用 UML 与覆盖准则相结合进行嵌入式软件测试用例的设计步骤为：

步骤 1　采用 UML 活动图动态建模：活动图的核心是活动，它借用了流程图、状态转换图和 Petri 网的思想，表示了活动可能发生的顺序[4]。活动图的形式化定义如下[5-6]:

定义 1　一个活动图是一个四元组 $D=(W,T,F,C)$；

1) W 是非空的有限活动集，w_0 为唯一的初始活动，w_n 为唯一结束活动；

2) T 是非空的有限转移集；

3) C 是有限条件转移表达式，c_i 对应 t_i；

4) $F \subseteq (W\times T\times C)\cup(T\times C\times W)$表示活动与转移之间的流关系。

定义 2　CW 为 D 的当前活动状态，对于所有的 $t\in T$：

1) $\curlywedge t=\{w\in W(w,t)\in F\}$ 和 $t'=\{w\in W|(t,w)\in F\}$ 分别表示 t 的前集和后集；

2) $enabled(CW)$表示只能从 CW 触发的转移集，$enable(CW)=\{t|\curlywedge t$ 满足 $c(t)$条件的 $CW\}$；

3) $fired(CW)$表示某时刻只能被从 CW 触发的转移，$fired(CW)=\{t|t\in enabled(CW) and (CW-\curlywedge t)\cap t'=\varnothing\}$，当 t 被触发，新的活动状态 $CS'=(CW-\curlywedge t)\cup t'$，如果满足条件的转移不唯一，可以选择其中任何一个未触发的转移。

步骤 2　测试场景生成：在活动图中以初始状态为起点，按照 DFS(Depth First Search method)遍历方法遍历活动图，遇到分支时将其拆分为不同场景，环路最多执行一次，对于并发活动需要指定约束条件，得到所有基本路径即是测试场景。从 w_0 开始按照 DFS 算法遍历活动图，测试场景的形式化定义[5]和生成算法如下：

定义 3　TS 是活动图 D 的测试场景，$ts\in TS$。TS 是一系列的活动和条件转移，$ts=CW_0\xrightarrow{[c_0]t_0}CW_1\xrightarrow{[c_1]t_1}\cdots\xrightarrow{[c_{n-1}]t_{n-1}}CW_n$，其中 $CW_0=\{w_0\}$，$CW_n=\{w_n\}$，CW_i 为当前活动，$t_i=fired(CW_i), i\geqslant 0$，$CW_i=\{(CW_{i-1}-\curlywedge t_{i-1})\cup t'_{i-1}\}, i\geqslant 1$。

测试场景生成算法：

1) 如果当前活动 $CW[i]$不为空，将当前活动入栈；

2) 如果 $CW[i]$可以触发的转移集不为空，则：$enabled(CW[i])=enabled(CW[i])-fired(CW[i]), CW[i+1]=CW[i]-\curlywedge t+t'$，将$[c_i]t_i$ 入栈，i++；

3) 如果 i!=n，goto 2)，否则 goto 5)；

4) 如果测试场景没有覆盖所有的活动和转移，回到上一个活动，goto 2)；

5) 如果 w_n 不存在可以触发的转移，则将 w_n 入栈，然后读出栈中数据作为测试场景，否则 goto 4)。

步骤 3　生成测试用例：为了满足测试充分性，生成测试用例时可以采用的覆盖准则有节点覆盖准则、迁移覆盖准则或逻辑路径覆盖准则。对于所有分支路径需要考虑至少执行 1 次，对于循环逻辑，考虑进入循环至少 1 次或不进入循环这 2 种情况。对活动图中的全部路径进行遍历，采用 1 种覆盖准则，结合执行过程中的控制流、数据流、分支节点的条件，确定每一路径的输入条件和状态条件，从而生成测试用例。输入值、方法调用和期望的输出值构成了一个测试用例。

2　仿真测试环境搭建

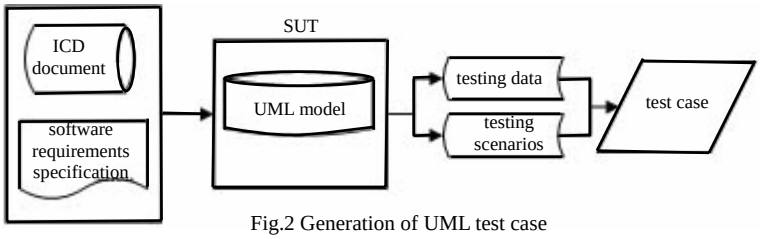

Fig.2 Generation of UML test case
图 2 UML 测试用例生成

软件测试环境能更方便地进行测试用例注入，更方便地观察软件的运行状态，更快捷地分析出软件运行的结果，因此软件测试环境直接影响着软件测试的效率。而软件仿真测试环境是为了更真实模拟嵌入式软件的实际运行环境而开发，利用仿真测试环境可以实现对被测软件自动的、实时的、非侵入式的闭环测试，所谓仿真，就是用特定的软件或硬件模拟设备的功能，达到简化测试环境的目的[1]。

仿真环境应该逼真地表现出真实运行环境的各种特性，对于嵌入式软件系统的测试而言，当仿真环境与测试相关的功能足够真实时，就可以满足测试要求。虚拟仪器技术可以经济、方便、快捷地设计软件仿真测试环境，为测试环境设计提供了一种新方法。如图 2 所示，虚拟仪器含硬件和软件 2 部分[7]，硬件主要用于获取被测软件发出的信号和向被测软件发送信号，提供信号传输的通道，利用 I/O 接口设备完成信号的采集、测量和调理。而软件主要用于控制，实现数据采集、分析、处理、显示和存储等，其实质是用计算机显示器的功能模拟传统仪器的控制面板，以多种形式表达发送的数据和数据采集的结果，利用计算机强大的软件功能实现信号数据的运算、分析和处理，具有操作方便、界面友好、数据管理方便等优点，容易观察被测软件的运行状态，并分析被测软件运行输出的结果，为提高软件测试效率提供了保障。

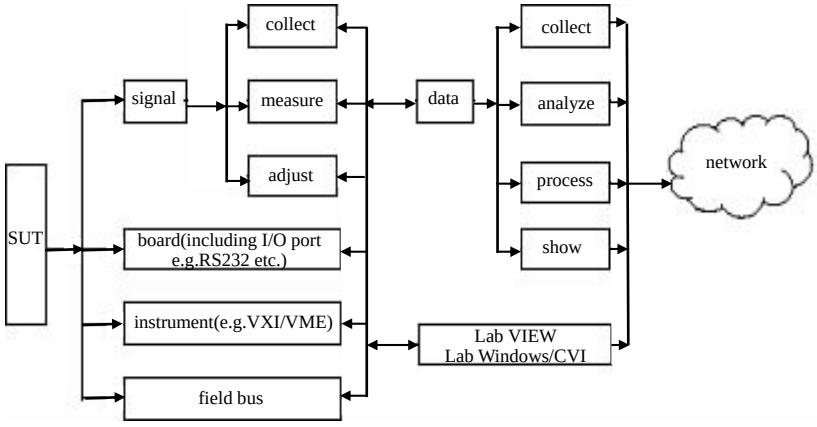

Fig.3 VI system diagram
图 3 VI 系统框图

仿真测试环境的软件部分是虚拟仪器技术最直接的运用和体现。通过专业虚拟仪器软件开发平台，NI 公司推出图形化软件编程平台 LabView，可以方便地定义和连接代表各种功能模块的图标，迅速建立起高水平的应用程序。LabWindows/CVI 是面向仪器的交互式 C 语言开发平台，其形象直观的界面设计让测试人员感觉就像操作传统仪器一样。

3　应用

以下以一个嵌入式软件的测试说明在 VI 仿真测试环境下，向测试环境注入由 UML 与覆盖准则相结合生成的技术测试用例，进行的嵌入式软件测试。被测软件由一系列功能相对独立的模块化软件组合而成，每个软件模块既具备完备的独立运行功能，又能够互相通信，组成完整的控制网络。被测嵌入式软件依托硬件模块运行，其工作不能离开单片机硬件系统，软件与硬件结合，完成全系统的控制与数据采集功能。被测软件系统中引起故障发生的参数多，故障判决条件复杂，采用 UML 与覆盖准则相结合技术生成测试用例 159 个。

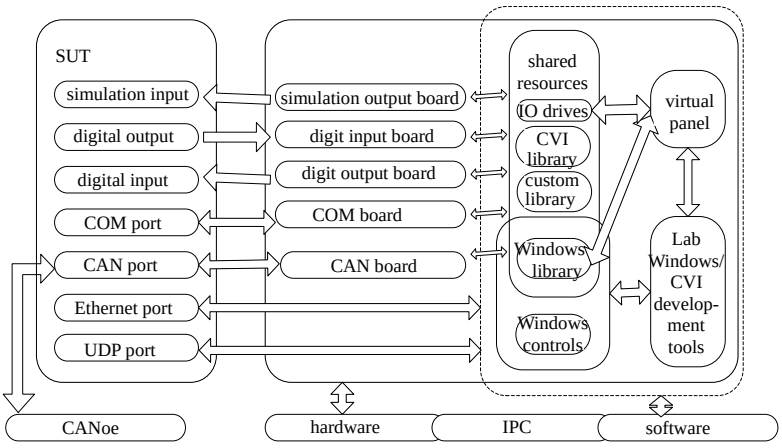

Fig.4 Simulation testing environment structure of software under test
图 4 被测软件仿真测试环境结构

被测软件接口很多，包括 CAN 接口、RS232 接口、RS485 接口、以太网接口等，仿真测试环境需要开发多种等效软件，图 4 是采用 VI 技术搭建的仿真测试环境结构示意图。

在 LabWindows/CVI 开发平台中，通过调用 IO 卡驱动库实现控制面板对板卡的访问和控制，图 5 是 VI 仿真测试环境模块化虚拟仪器面板示意图。

通过向被测软件仿真测试环境进行测试用例注入，观察软件的运行状态和分析软件运行结果，发现软件缺陷 78 个。

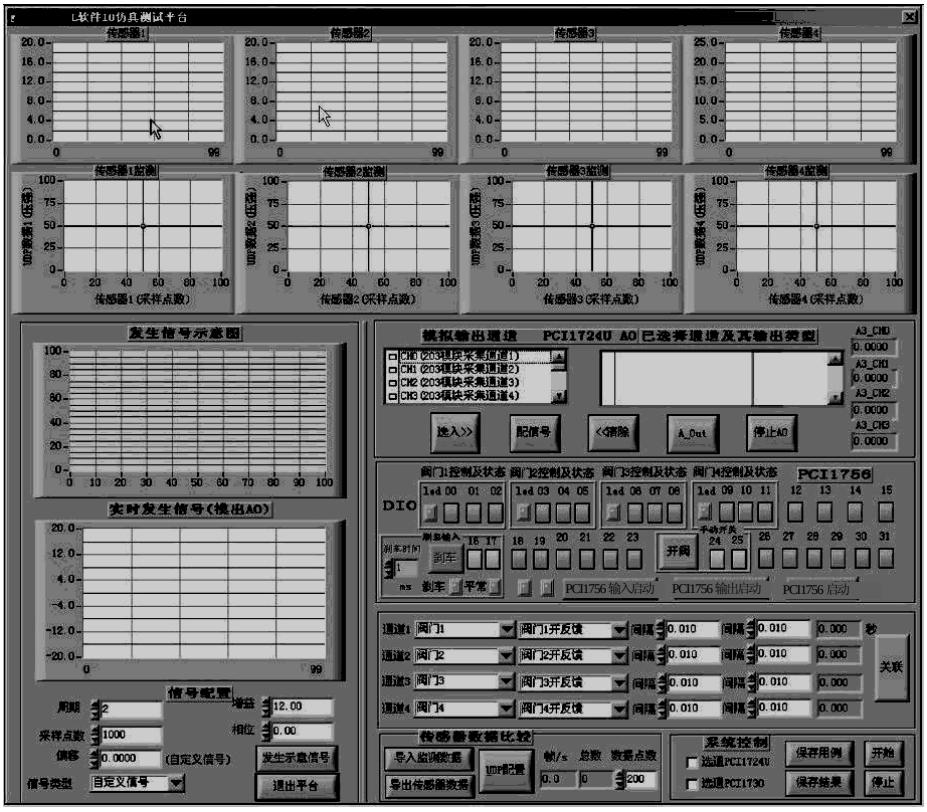

Fig.5 Modular virtual instrument panel of simulation testing environment for software under test
图 5 被测软件仿真测试环境模块化虚拟仪器面板

4 结论

随着嵌入式系统的专用程度越来越高，对嵌入式系统的可靠性要求也越来越高。本文通过向 VI 仿真测试环境中注入测试用例进行嵌入式软件系统测试。实践证明，将 UML 与覆盖准则相结合用于指导软件测试设计，既有效地解决了 UML 模型到测试用例的自动生成，又减少了测试用例的编写时间。同时采用 VI 技术建立仿真测试环境，大大降低了设计仿真测试环境的成本，仿真测试环境与被测设备之间连线简单，保证了测试环境构建更加快速、便捷、可靠，同时 VI 软件也便于管理、监控和进行数据分析处理。

参考文献：

[1] 康一梅. 嵌入式软件测试[M]. 北京:机械工业出版社, 2008.
[2] Bran Selic. 统一建模语言(UML)[EB/OL]. (2005-04-29) [2010-05-27]. http://www.ibm.com/developerworks/cn/rational/321_ uml/.
[3] Cavarra A,Crichton C,Davies J. A method for the automatic generation of test suites from object models[C]// Proc. of the 2003 ACM Symposium on Applied Computing. UK:[s.n.], 2003:1104-1109.
[4] 梁义芝,王延章. UML 活动图的形式语义及分析[J]. 计算机工程与应用, 2003,39(18):28-31.
[5] 牟凯,顾明. 基于 UML 活动图的测试用例自动生成方法研究[J]. 计算机应用, 2006,26(4):844-847.
[6] 殷永峰,刘斌,姜同敏. 基于 UML 的嵌入式软件测试用例生成方法研究[J]. 计算机应用研究, 2008,25(10):3018-3022.
[7] 谢敏,罗永红. 基于虚拟仪器的可重构仿真测试环境设计[C]// 系统仿真技术及应用, 合肥:[s.n.], 2009:402-405.

作者简介：

漆莲芝(1978-)，女，成都市人，主要研究方向为计算机软件与理论.email:qlzscu@126.com.

谢 敏(1981-)，男，四川省三台县人，主要从事测控技术、测控系统应用、面向信号处理的可配置计算技术研究.

张 军(1972-)，男，四川省资阳市人，主要研究方向为嵌入式软件测试.