

基于 RTX 的实时通用测控软件设计与实现

李俊贤¹, 李红宇^{1,2}

(1 桂林电子科技大学 电子工程与自动化学院, 广西 桂林 541004; 2 中国航天科工集团 801 厂, 广西 柳州 545006)

摘 要: 针对 Windows 系统实时性不足、传统实时测控软件通用性不强的问题, 应用 RTX 技术增强 Windows 实时性, 采用灵活的脚本语言描述测试任务增强实时测控软件通用性; 设计并实现了基于 RTX 的实时通用测控软件平台; 重点阐述了软件设计与实现过程中的设备识别、脚本规范设计、I/O 驱动开发等关键技术. 实验结果表明: 一方面, RTX 技术的应用解决了 Windows 系统实时性不足的问题, 另一方面, 设计的软件能够通过编辑执行不同测试脚本完成不同测试任务, 对解决实时测控软件通用性不强的问题具有重要意义.

关键词: 实时性; 通用性; 实时扩展技术; 脚本技术; 测试任务

中图分类号: TP391

文献标识码: A

文章编号: 1000-7180(2016)10-0120-05

Design and Realization of Real-Time General Measurement and Control Software Based on RTX

LI Jun-xian¹, LI Hong-yu^{1,2}

(1 School of Electrical Engineering and Automation, Guilin University of Electronic Science and Technology, Guilin 541004, China; 2 Factory 801 of China Aerospace Science and Industry Corporation, Liuzhou 545006, China)

Abstract: For Windows system shortage of real-time monitoring and traditional control software are not universal, applicate RTX technology to enhance real-time Windows, use flexible scripting language to describe the test tasks, enhanced real-time monitoring and control software commonality; design and implement real-time monitoring and control software system based on RTX; describe the software design and implementation of key technologies, such as device identification, script specification design, I/O-driven development. The results show that on the one hand, RTX technology application solves the problem of lack of real-time Windows system, on the other hand, the design of the software system is able to complete different testing tasks by editing and performing different test scripts, and has great significance to solve the problem that Real-time monitoring and control software versatility is not strong.

Key words: real-time; versatility; real-time extention technology; scripting technology; test tasks

1 引言

许多测控场合, 要求操作系统一方面有图形显示、人性化操作界面、多功能可扩展总线技术等功能, 另一方面要有良好的实时性, 以支持在特定周期对目标系统的实时驱动^[1-3].

RTX 的出现很好地解决了以上问题. 它是基于 Windows 平台的硬实时系统, 对 Windows 的硬件

抽象层进行了实时性扩展, 是当前 Windows 平台上基于软件的唯一硬实时解决方案; RTX 实时子系统 RTSS 的线程优先于所有 Windows 线程, 提供对 IRQ、I/O、内存的精确控制, 其定时器时钟分辨率为 100 ns, 最低定时周期达到 100 μ s, 为实时任务的执行提供了足够的可靠性; RTX 还支持多种开发环境以及处理器多核技术, 开发难度低, 开放性好, 易于维护, 而且成本也较低, 具有图形界面和硬件兼容性

方面的优势^[4]. 因此, RTX 技术为解决测控软件的实时性问题提供了良好的解决方案.

针对以具体应用为出发点开发的实时测控软件测试任务比较专一, 扩展较为困难, 通用性不强这个问题, 本文借鉴软件测试自动化中的脚本技术^[5], 提出在测试过程中采用灵活的脚本语言描述测试任务, 并在设计的软件系统中解释执行. 该方法针对不同测试对象, 能够通过编辑执行不同测试脚本完成相应测试, 有效地解决了在变更测试任务需求时传统实时测控软件无法快速适应的问题, 且开发难度小, 维护成本低.

2 软件系统设计方案

整个软件平台由 Win32 进程、RTX 进程两大部分组成, 整体架构如图 1 所示.

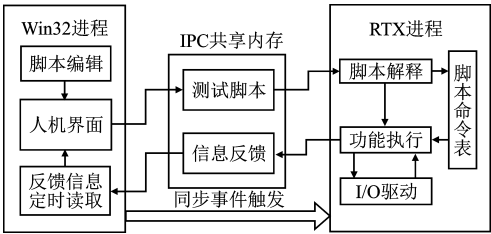


图 1 软件整体架构图

为了利用 Windows 丰富的绘图接口和友好的用户界面优势, Win32 进程利用 MFC 进行开发, 主要负责编辑加载测试脚本、启动与停止测试、控制脚本执行、数据实时显示与记录以及数据查询等非实时性工作; RTX 进程是整个软件系统工作的基础, 对数据的实时采集和输出都在该层完成, 它相当于一个伺服作动器, 测试开始后负责解释执行 Win32 进程发送下来的测试脚本语句, 并将脚本语句执行结果信息(包括反馈命令和测试数据)写入共享内存, 供 Win32 进程读取. 两进程之间通过共享内存进行通信, 通过事件机制完成同步.

2.1 Win32 进程设计

Win32 进程主要完成一些非实时性任务, 核心是测试脚本编辑模块、人机界面模块、反馈信息定时读取模块.

测试的第一步是开发测试脚本, 测试脚本是对特定测试任务的描述, 包含了脚本命令以及命令参数等信息, 开发流程如下:

- (1)分析被测对象输入特性, 确定是否需要系统为测试对象提供激励信号, 以及激励信号的参数指标;
- (2)分析被测对象输出特性, 确定被测对象输出信号的参数指标;

(3)根据步骤(1)和(2)得到的被测对象输入输出特性, 分析输入输出之间的逻辑关系, 依据脚本规范编辑并保存测试脚本;

(4)执行测试脚本, 判断被测对象的输出是否满足设计要求.

脚本编辑模块的设计与实现大大提高了测试任务开发的灵活性、系统的可维护性, 实现了测试任务开发的通用性. 人机界面应用 MFC 开发, 主要负责加载测试脚本、启动与停止测试, 以图形化方式实时显示测试信息等功能, 发挥了 Windows 界面友好、功能强大、操作便捷的优势^[6]. 反馈信息定时读取模块负责定时读取脚本语句执行结果信息, 将反馈信息发送给人机界面进行实时显示, 并以文本形式进行保存, 以及根据反馈信息控制脚本执行.

Win32 进程流程如图 2 所示.

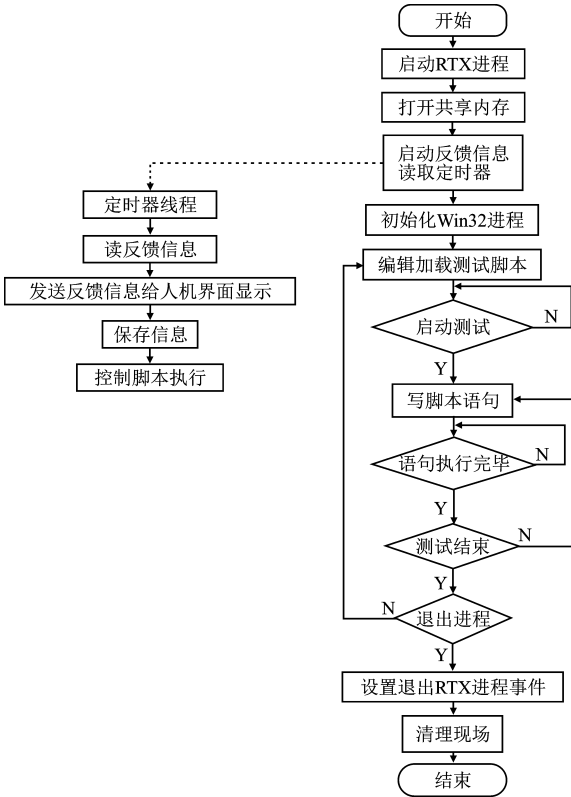


图 2 Win32 进程流程图

2.2 共享内存设计

共享内存是 RTX 技术提供实现与非实时进程信息沟通的重要方式, 共享内存存在整个软件平台中充当着“桥梁”的角色, 共享内存机制的实现也是实时测控软件开发成功与否的关键.

软件设计采用了两片共享内存, 一片负责 Win32 进程向 RTX 进程传递测试脚本, 对于该片共享内存, Win32 进程只具有写的权限, RTX 进程

只具有读的权限. 另一片负责 RTX 进程向 Win32 进程传递脚本语句执行结果信息, 此片共享内存, Win32 进程只具有读的权限, RTX 进程只具有写的权限. 这种共享内存设计方案使得 Win32 进程与 RTX 进程之间的通信更加清晰^[7].

2.3 RTX 进程设计

RTX 进程是整个软件的核心, 启动测试任务后, 负责解释执行测试脚本, 对实时性要求较高的任务, 如实时数据的采集与处理、实时控制量的输出等, 都在 RTX 进程执行.

启动测试后, RTX 进程对 Win32 进程发送下来的每一条脚本语句进行解释, 抽象出脚本命令和命令参数, 形成脚本命令表, 然后根据得到的脚本命令, 读取命令参数, 执行脚本语句功能, 并将每一条脚本语句执行结果信息写入共享内存, 直到测试结束. 脚本语句目前有以下几种: I/O 板卡操作语句、数据处理语句、条件循环语句、条件跳转语句以及脚本延时语句等.

RTX 进程流程如图 3 所示.

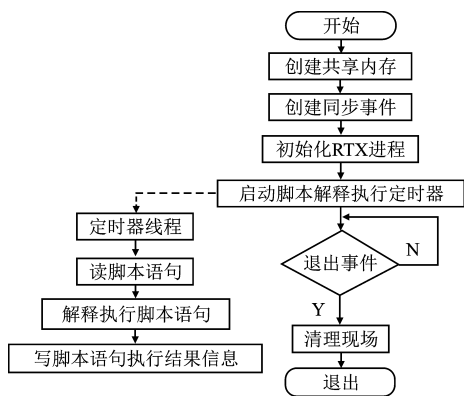


图 3 RTX 进程流程图

3 关键技术

实时通用测控软件设计存在以下关键问题需要解决: (1) 在 RTX 环境下, 不同硬件配置 I/O 板卡通常通过厂商 ID 和设备 ID 加以识别, 而相同硬件配置 I/O 板卡的厂商 ID 和设备 ID 均相同, 无法用此方法加以识别, 因此, 如何在 RTX 进程中识别相同硬件配置 I/O 板卡成为测试脚本执行的首要任务; (2) 测试脚本的编辑需要有特定的脚本规范, 良好的脚本规范能够保证脚本的统一解释运行, 使得测试运行时具有良好的交互性; (3) I/O 板卡厂家不提供 RTX 环境下驱动程序, 需要用户自主开发, 驱动程序设计的好坏, 关系到能否实现对 I/O 板卡的精准控制. 关键技术的实现如下.

3.1 设备识别方法

根据 PCI 总线配置机制, 每一个 I/O 板卡在 PCI 总线上的坐标由总线号、设备号、功能号确定, 并且是唯一的, 因此, 针对如何在 RTX 进程中识别相同硬件配置的 I/O 板卡这一问题, 本文提出在测试脚本的编辑与执行过程中, 采用“设备 ID-总线号+设备号+功能号”作为逻辑设备名来识别设备, 如“1716-500”代表设备 ID 为 1716, 总线号为 5、设备号为 0、功能号为 0 的 I/O 板卡. RTX 进程运行时通过遍历 PCI 总线, 枚举每一个 I/O 设备获得逻辑设备名, 并保存成文本文件, 供脚本解释执行时查询比对以找到目标板卡, 完成对板卡的操作. 这种“设备 ID-总线号+设备号+功能号”的设备识别方案, 解决了多个硬件配置相同 I/O 板卡在 RTX 环境下容易混淆的问题. 设备识别流程如图 4 所示.

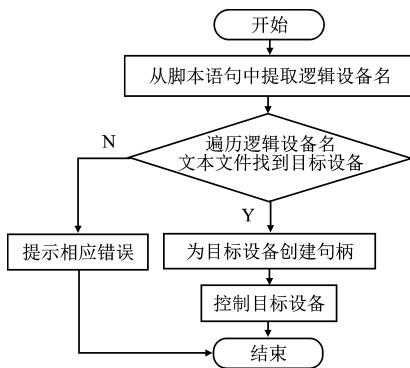


图 4 设备识别流程图

3.2 脚本规范设计

通过参考 SCPI(可编程仪器命令标准), 本文借鉴成熟的软件测试中关键字驱动脚本技术^[8], 定义了 I/O 板卡操作、数据处理、条件循环、条件跳转、脚本延时运行等部分基本测试脚本语句规范.

(1) I/O 板卡操作语句语法格式

Operate: DevName: Para 1, Para 2, …, Para n, \$

其中, Operate 为 I/O 板卡操作定义, 如“Open”(打开)、“Close”(关闭)、“AD”(AD 采集)、“DA”(DA 输出); DevName 为逻辑设备名, 如“1716-500”(设备 ID 为 1716, 所在 PCI 总线号为 5、设备号为 0、功能号为 0 的 I/O 板卡)、“1716-510”(设备 ID 为 1716, 所在 PCI 总线号为 5、设备号为 1、功能号为 0 的 I/O 板卡); Para n 为 I/O 板卡参数, 如输出信号的类型、通道号、频率、幅值, 输入信号的通道号、采样频率、接线方式、采样点数等, 其顺序关系确定后就被固定下来; \$ 代表参数提取结束. 如语句:

DA: 1716-500: DC, Ch 1, Mode 1, Voltage 6.

0000,\$,

其功能是利用设备 1716-500 的 DA 通道 1 输出 6 V 直流电平,输出电压范围(Mode)为 1(−10~10 V).

(2)数据处理语句语法格式

DP; Operate; DevName; Para 1, Para 2, …… , Para n, \$

其中,DP 为数据处理操作定义,如“Save”(保存)、“Max”(求最大值)、“Min”(求最小值)、“FFT”(对数据做 FFT 变换);Operate 为数据采集操作定义,如“AD”、“DI”;DevName 为逻辑设备名;Para n 为数据处理参数,如输入信号的通道号、数据保存文件名等;\$ 代表参数提取结束.如语句:

Save; AD; 1716-500; Filename data0, Ch 0, \$

其功能是将设备 1716-500 的 AD 通道 0 数据缓存区的数据保存到文件 data0;

Max; AD; 1716-500; Ch 2, \$

其功能是对设备 1716-500 的 AD 通道 2 数据缓存区的数据求最大值.

(3)条件循环语句语法格式

do;
语句 1
……
语句 n
until;结束条件

其中,do 表示开始循环执行循环体内的脚本语句;until 表示满足结束条件时跳出循环,执行循环体下面的脚本语句,否则将会一直执行循环体内脚本语句;语句 n 为对 I/O 板卡操作或对数据处理的脚本语句.

(4)条件跳转语句语法格式

if:跳转条件
goto;ScriptLineNumber

该语句的功能是如果满足跳转条件,跳转到行号为 ScriptLineNumber 的脚本语句处开始执行脚本.

(5)脚本延时运行语句语法格式

Delay;延时值

脚本延时运行语句与设备没有关系,它的作用是延迟脚本语句的执行,如当前数据采集还没有完成,需要延迟下一条脚本语句的执行.

本文仅针对 I/O 板卡部分功能(AD、DA、DI、DO)制定并实现了以上基本测试脚本语句规范,在以后系统的升级维护过程中,可根据测试任务需求

制定并实现更多的脚本语句规范,以不断完善整个测试系统.

3.3 I/O 驱动开发方法

驱动程序是直接控制设备进行工作的程序,是应用程序和硬件设备之间沟通的“桥梁”.RTX 对 I/O 板卡的控制通过配置内部寄存器方式完成.要完成寄存器的配置,需要知道寄存器的地址,寄存器地址等于基地址加上偏移地址,偏移地址可通过查阅厂家提供的板卡寄存器操作手册获得,而基地址需要遍历 PCI 总线号、设备号以及功能号获得,如图 5 所示.从 I/O 设备的配置结构体 PciData 可以得到 I/O 设备的厂商 ID 与设备 ID,若和目标板卡的相同,则 PciData 中存放的地址即为该板卡的基地址.

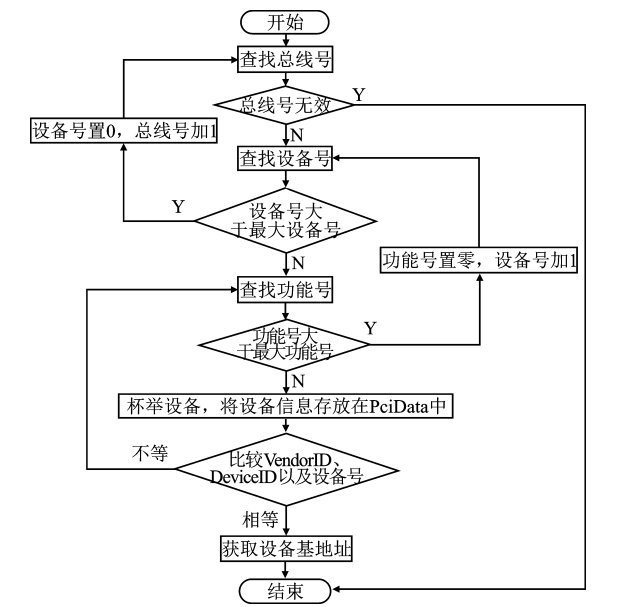


图 5 基地址获取流程图

获取设备基址后,通过调用 RTX 提供的 I/O 读写函数,按照设备寄存器操作手册里的偏移量和格式对寄存器进行配置.作者采用这种方法,在 RTX 环境下开发了研华系列 I/O 板卡驱动程序,并投入到了实际应用当中,效果证明驱动程序运行良好.

4 系统测试

4.1 实时性测试

在硬件环境为 Intel(R) Core(TM) i5-4670 @ 3.4 GHz 台式机上,利用 RTX 提供的 RTX Demo 定时器精度测试程序对 Windows 和 RTX 系统的 1 ms 定时器的延时进行了约 60 s 的测试,测试结果如图 6 所示.

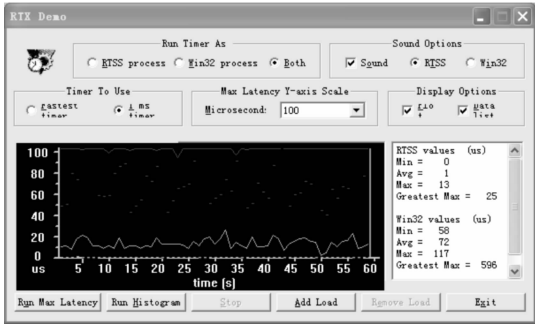


图 6 Windows 和 RTX 系统 1 ms 定时器延时测试图

从图 6 中可以看出 Windows 系统 1 ms 定时器的延时在 58~596 μs 之间变化,虽然平均延时在 72 μs 附近,但评判一个系统实时性的优劣应从最坏的情况考虑,因此,Windows 系统并不能满足较高的实时性需求,而 RTX 系统的 1 ms 定时器延时在 0~25 μs 之间,远远优于 Windows 系统。

此外,在相同测试硬件环境下,利用 RTX 提供的 Platform Evaluator 工具对 Windows 和 RTX 系统的线程切换延时进行了测试,测试结果为:Windows 系统的线程切换延时在 12.4~622.8 μs 之间,RTX 系统的线程切换延时在 1.4~42.6 μs 之间。RTX 系统同样远远优于 Windows 系统。

经过以上对比可知 RTX 具有卓越的实时性能,能够为实时性测试提供有力保障。

4.2 通用性测试

通过编辑执行两个能够完成不同测试任务的测试脚本对系统的通用性进行了验证。

4.2.1 电源模块测试

电源模块测试脚本如图 7(a)所示。该脚本完成的功能是测试 10 V 转 5 V 电源模块工作是否正常。首先利用板卡 1716-500 的 DA 通道 1 输出一个 10 V 直流电压给电源模块输入端,然后利用板卡 1716-500 的 AD 通道 8 测试电源模块输出电压值,并计算测试电压的平均值,规定当测试电压平均值大于 5.5 V 时,说明电源模块输出异常,然后关闭设备,结束测试。测试结果如图 7(b)所示。

4.2.2 监视输出电压

监视输出电压的测试脚本如图 8(a)所示。该脚本完成的功能是监视 I/O 板卡输出的 3.3 V 电压是否正常。首先利用板卡 1716-500 的 DA 通道 1 输出 3.3 V 电压,然后利用板卡 1716-510 的 AD 通道 2 对输出电压进行回采,并计算回采电压最大值,规定当输出电压最大值小于 2 V 时,说明 I/O 板卡输出电压异常,然后关闭设备,结束测试。图 8(b)为监

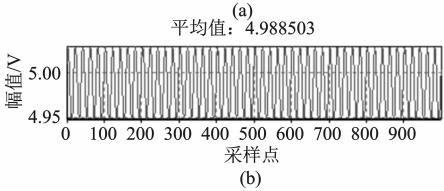


图 7 电源模块测试脚本及测试结果图

视到电压正常输出的图形显示。

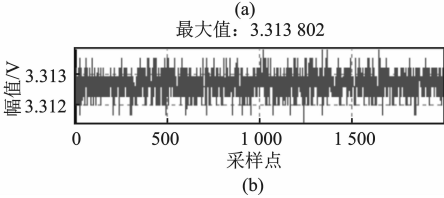


图 8 监视输出电压测试脚本及测试结果图

通过编辑与执行不同测试脚本,系统可以完成不同测试任务,以及 Windows 的图形显示、人性化操作界面、多功能可扩展总线技术等功能,证明了该系统具有较强的通用性。

5 结束语

本文设计并实现了基于 RTX 的实时通用测控软件系统,该系统将 RTX 的实时性与 Windows 良好的交互性结合在一起,解决了 Windows 实时性不足的问题;针对不同测试对象,通过编辑执行不同的测试脚本,实现相应测试,达到了增强实时测控软件通用性的目标。

表 1 相关文献对比				
	摄像头类型	标记物	校正是否需要针尖触碰	针尖校正误差/mm
本文	可见光	棋盘格角点	否	1.37
Treepong ^[2]	红外	红外标记球	是	2.85

5 结束语

本文使用两可见光摄像头、KUKA 机械手和普通手术针搭建手术导航系统,提出基于可见光标记物的系统校正方法,实现了各系统构件间的校正.这种用于手术导航的持针器械校正方法优势是实现了对手术针的无接触式校正,保证了手术针的无菌性.实验数据表明,该手术导航系统的校正精度能满足当前经皮胸腔手术的下针需求,有效解决了该类手术仅凭医生经验扎针导致的手术精度低的问题.后续工作将以本文提出的系统校正算法为基础,继续实现相应的手术针跟踪定位以及机械手控制算法,从而完善整个手术导航系统.

参考文献:

[1] Naghavi M, Wang H, Lozano R, et al. Global, regional, and national age-sex specific all-cause and cause-specific mortality for 240 causes of death, 1990-2013; a systematic analysis for the Global Burden of Disease Study 2013[J]. Lancet, 2015, 385(9963): 117-171.

[2] Treepong B, Tanaiutchawoot N, Wiratkapun C, et al.

On the design and development of a breast biopsy navigation system: path generation algorithm and system with its GUI evaluation[C]// Biomedical and Health Informatics (BHI), 2014 IEEE-EMBS International Conference on. [s. l.]:IEEE, 2014: 273-276.

[3] Bootsma G J, Siewerdsen J H, Daly M J, et al. Initial investigation of an automatic registration algorithm for surgical navigation[C]// Engineering in Medicine and Biology Society, 30th Annual International Conference of the IEEE. Washington, DC, USA: IEEE, 2008: 3638-3642.

[4] Yang R, Wang Z, Liu S, et al. Design of an accurate near infrared optical tracking system in surgical navigation[J]. Lightwave Technology, Journal of, 2013, 31(2): 223-231.

[5] Moré J J. The levenberg-marquardt algorithm: implementation and theory[M]. Numerical analysis. Springer Berlin Heidelberg, 1978: 105-116.

[6] Bouguet J. Camera calibration toolbox for matlab[EB/OL]. [2016-01-13] http://www.vision.caltech.edu/bouguetj/calib_doc/.

作者简介:

杨一帆 男,(1989-),硕士研究生.研究方向为计算机视觉和机械手. E-mail:yytbupt@126.com.

谢翔 男,(1971-),副研究员.研究方向包括集成电路设计、图像处理、生物医学电子等.

李国林 男,(1970-),副教授.研究方向包括集成电路设计、互连分析以及 RFIC 设计.

(上接第 124 页)

参考文献:

[1] 姚建勇,焦宗夏,吴帅,等. RTX 在半实物仿真中的延时问题及解决[J]. 机床与液压, 2008, 36(12): 162-165.

[2] 黄键,宋晓,薛顺虎. RTX 平台下实时仿真系统的设计方法[J]. 计算机应用与软件, 2009, 26(4): 167-169.

[3] 张前福,王华闯,许俊平,等. 基于 RTX 的实时处理系统设计[J]. 微电子学与计算机, 2012, 29(8): 58-61, 65.

[4] Ardence 公司. RTX 技术白皮书[R]. 北京航天捷越(美斯比)科技有限公司, 2004.

[5] 杜德鑫,孙继银,张翼飞. 一种面向应用的测试任务自动生成与运行方法[J]. 测控技术与仪器仪表, 2009, 35(9): 98-100.

[6] 杨小亭,曹义东,宋科. 基于六自由度关节机器人的一种计算机控制系统[J]. 微电子学与计算机, 2012, 29(7): 192-196.

[7] 周林雪,李岩,王建军,等. 基于 RTX 的激光制导武器半实物仿真研究[J]. 计算机仿真, 2013, 30(1): 45-49, 119.

[8] 凌永发,张云生,郭秀萍. 软件测试自动化中的脚本技术[J]. 云南民族学院学报(自然科学版), 2002, 11(1): 544-548.

作者简介:

李俊贤 男,(1989-),硕士研究生.研究方向为自动测试总线与系统. E-mail:lijunxian625@163.com.

李红宇 男,(1974-),研究员,硕士生导师.研究方向为自动测试总线与系统、虚拟仪器.