文章编号:1671-4598(2023)03-0083-06 DOI:10.16526/j.cnki.11—4762/tp.2023.03.013 中图分类号:TP3 文献标识码:A

基于动态可重构技术的装备测试系统研究

罗东明，马 跃，董 旭，连文涛
（北京机电工程研究所，北京 100074）

摘要：当前国内装备测试系统存在实时性差、测试资源冗余、成本高等问题，针对以上问题，提出了基于FPGA部分动态重构技术的装备测试系统，该系统基于FPGA动态可重构技术并结合嵌入式操作系统实现测试资源的动态管理，并开发了用于测试过程的硬件装备测试任务编程模型，提出了一种用于重构任务加载的ICAP控制器；该系统实现测试过程的并发执行，从而增强装备测试系统测试的实时性，进而提高测试的准确性与覆盖性；在验证试验中，将动态重构测试系统应用于装备测试实例中，试验结果表明硬件重构测试任务加载正常，各测试资源功能执行正确。

关键词：动态可重构；装备测试；FPGA；实时操作系统

Realization of Automatic Test System Based on Dynamic Reconfigurable Technology

LUO Dongming, MA Yue, DONG Xu, LIAN Wentao
(Beijing Electro-mechanical Engineering Institute, Beijing 100074, China)

Abstract: Current domestic automatic test system has the problems of poor real-time performance, redundant test resource and high cost. Based on above problems, an automatic test system based on FPGA partial dynamic reconfiguration technology is proposed, The FPGA dynamic reconfigurable technology is used to be combined with the embedded system, realize the dynamic management of test resources, develop a hardware automatic test task programming model for the test process, and propose a reconfigurable test loading internal configuration access port (ICAP) controller. The system truly realizes the concurrent execution of the test process, thereby enhancing the real-time performance of the automatic test system, and then improving the accuracy and coverage of the test. In the verification test, the dynamic reconfiguration test system is applied to the automatic test. The test results show that the hardware reconfiguration loading is normal, and the function of each test resource is correct.

Keywords: dynamically reconfigurable; automatic test; FPGA; real-time operating system

0 引言

当前国内的装备保障与测试系统大多是基于Windows XP系统开发，但装备系统内部任务具有高实时性的特性，因而当前装备测试系统无法满足其测试实时性的要求；其次，当前测试设备普遍使用总线板卡的方式进行设计，使得测试设备体积大，板卡资源存在极大冗余，进而使得测试设备成本较高。同时基于软件的并行测试受到测试设备处理器性能的影响，软件的并发运行并不是真正的硬件并发执行，而仅仅是对系统时间片的分时复用。因而，开发一款高实时性、并发处理能力强的装备测试系统具有重要的现实意义，能够极大的提高对装备系统测试的实时性，进而提高测试的覆盖性。

由于较高的灵活性和相比专用集成电路具有更低的价格，（FPGA，field programmable gate array）正被广泛应用到嵌入式系统中。同时现代FPGA的灵活性被其动态可重构（dynamic partial reconfiguration）功能进一步增强[1]。该特性能够在运行过程中，实现用户动态改变FPGA特定区域逻辑资源的功能，并且在改变过程中其他区域的逻辑功能不受任何影响，同时配置的次数和时间不受限制[2]。FPGA内部集成的硬件逻辑使其实现了真正的并发处理与运行，而动态配置的特性，使得在运行中修改部分逻辑资源功能得以实现。

随着开源的Linux系统在嵌入式中的广泛应用，Xilinx公司推出了ZYNQ系列的FPGA，该系列的FPGA将处理器的软件可编程能力与FPGA的硬件可编程能力实现了完美的结合。ZYNQ系列的FPGA，保留了V系列FPGA的可重构特性[3]。其支持通过PS控制重构端口进行重构，从而实现在运行过程中对硬件逻辑资源功能的动态改变。本文基于开源Linux系统，并结合ZYNQ系列FPGA的可重构特性，设计了具有高并发处理能力的实时装备测试系统，提高了装备测试系统的测试实时性，增强了测试过程中并发处理的能力，进而提高了测试的可靠性与准确性。整个装备测试系统是基于ZYNQ系列FPGA进行开发。

1 典型飞航装备测试系统

装备测试系统是指采用计算机控制，能够实现自动化

收稿日期:2022-11-06； **修回日期**:2022-12-05。

作者简介:罗东明(1992-)，男，山西朔州人，硕士研究生，工程师，主要从事装备的装备测试与故障诊断方向的研究。

引用格式:罗东明，马 跃，董 旭，等.基于动态可重构技术的装备测试系统研究[J].计算机测量与控制，2023，31(3)：83-88，101.

测试的系统，装备测试系统广泛应用于设备调试、故障检测和定位、系统状态确认等场合。装备测试设备是在总装厂或技术阵地完成装备技术准备和周期性维护测试工作的维护设备，其在测试过程中需要与装备控制装置进行指令交互以及实时采集装备的关键信号并依据判据完成测试，从而确定装备的性能状态。

目前国内装备测试系统主要是基于标准软件开发环境二次开发的测试平台，平台一般依赖用户及操作系统，测试流程库是基于数据库开发，具有一定的二次开发能力，但系统实时性较差，同时内部的并发测试完全通过软件调度实现，导致测试过程并行执行能力较差，进而限制了对装备瞬时状态的监测能力。

为了方便用户进行测试序列开发和操作，另一种测试系统是通过用户鼠标拖拽方式实现测试流程的开发与仪器配置，缩短了测试程序集的开发周期和难度，但系统实时性较差，同时由于其测试程序执行过程为顺序执行，因此其不具备并发测试的功能。

以上提及的装备测试系统，实时性和并发测试能力都无法满足当前装备测试的实时性需求，因而，有必要开发一款实时性强且并发执行能力好的装备装备测试系统。

2 重构系统结构

本文提出的装备测试系统是在嵌入式 Linux 系统的基础上，以 FPGA 为主控制器并结合其部分动态可重构配置的特性，实现测试任务的硬件并发执行，进而满足飞航装备对装备测试系统实时性以及并发性的需求；FPGA 部分动态可重构的技术使得装备测试系统能够进行软扩展，减少了由于被测对象的改变而带来的硬件修改，缩短了设计周期；同时该特性使得装备测试系统的主控制器体积大大降低，进而推动测试系统向着小型化、高集成发展。系统总体结构见图 1 所示。

整个装备测试系统由底层的硬件系统和基于开源 Linux 的嵌入式软件系统两部分组成。底层的硬件系统是由硬件描述语言 VHDL 进行设计，其中包含存储子系统、ICAP（internal configuration access port）控制器、以及装备测试硬件重构任务（HT，hardware task）[4]。软件部分用于实现在 Linux 多线程的基础上控制底层硬件。

在整个装备测试系统中，测试过程是由硬件重构测试

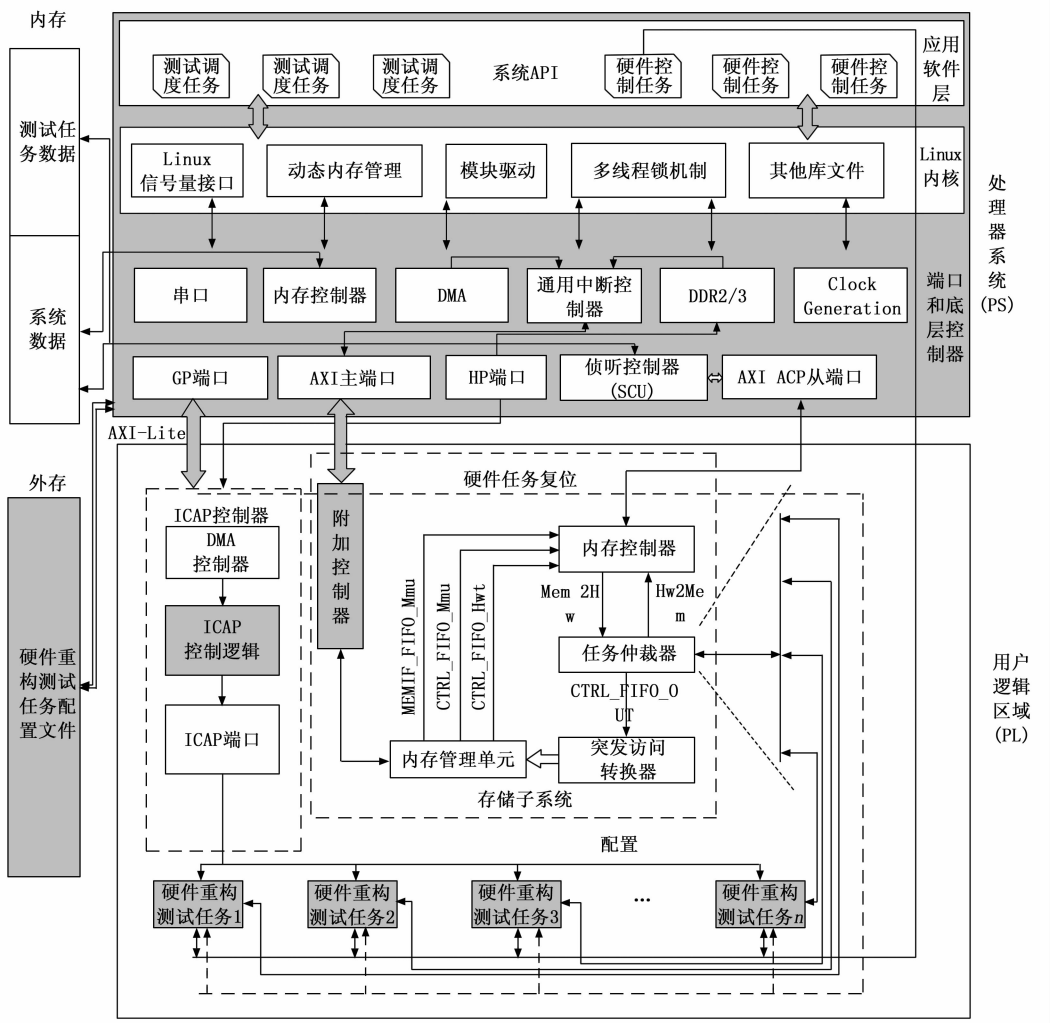

图 1 系统总体结构

任务和软件任务共同完成，硬件重构测试任务是运行在 FPGA 硬件逻辑资源上功能电路，其用来执行相关信号的采集与数据传输。软件任务实现对硬件重构测试任务的控制，辅助系统上层与底层 FPGA 资源进行信息交互。

用户根据测试需求，向装备测试系统发起重构请求，即此次测试过程包含的硬件重构测试任务数量，系统通过 ICAP 控制器加载相应硬件重构测试任务的配置文件，为测试提供相应的仪器资源；硬件重构测试任务是根据测试功能通过 Xilinx 的 PLANAHEAD 工具进行生成，生成之后将其存储在外部存储器中；在硬件重构测试任务建立之后，系统为每一个任务建立相应的软件任务，以便硬件重构测试任务与重构系统进行数据交互和控制作用。

在整个过程中，系统的内存用来存储硬件重构测试任务的数据以及系统数据。系统的外存用来存放硬件重构测试任务的部分比特文件和重构任务放置的位置信息配置文件。位置配置文件中描述了每一个硬件重构测试任务的放置信息。在测试过程中，装备测试系统通过 ICAP 控制器加载相应的硬件重构测试任务，通过查询位置配置文件中任务位置信息，在线修改配置文件的位置配置信息，将比特文件加载到重构任务指定的位置，实现对测试系统中测试资源的动态配置；在装备测试系统执行测试的过程中，通过系统上层的软件任务实现对硬件重构测试任务的数据读写，完成对装备系统的测试过程。

动态重构系统为软件和硬件提供了统一的系统服务和编程接口。在 Linux 多线程模型的基础上，系统增加了对硬件重构任务的支持。在用户角度中，通过委托任务使得硬件任务和软件任务具有相同的使用机制，方便了用户使用 FPGA 动态可重构特性。

重构硬件任务是用户基于硬件描述语言开发的具有特定功能的逻辑电路，包含与上层系统进行同步的同步逻辑部分以及用户自定义的功能模块；ICAP 控制器负责从外存中加载相应硬件重构任务的比特配置文件，并配置到指定的重构区域中；存储子系统负责硬件重构任务与系统内存间进行数据交互；Linux 内核负责提供系统开发过程中依赖的库文件以及底层模块的驱动文件；应用软件层提供用户用于重构过程控制的系统接口，包括基于 Linux 系统调用的重构软件任务以及委托任务创建、硬件任务动态配置以及任务间进行数据同步。

重构过程是由硬件任务、软件任务和委托任务共同完成的，硬件任务是运行在 FPGA 硬件逻辑资源上功能电路，而软件任务是基于 Linux 多线程机制进行开发的特定软件线程。委托任务是硬件任务与用户之间的桥梁，通过委托任务，用户可以像控制软件任务一样控制硬件任务，包括向硬件任务发送数据和命令。

在系统运行中，用户向系统提出重构任务请求，请求内容包含此次重构包含的硬件任务和软件任务数目，重构系统根据用户请求的硬件任务数量，通过 ICAP 控制器将硬件任务比特配置文件从外部存储中重配置到 FPGA 内部；相应的比特配置文件是通过 Xilinx 的 PlanAhead 工具开发并生成，存储在外部存储器中；在硬件任务建立的同时，系统为每一个硬件任务建立相应的委托任务，用于硬件任务与上层系统进行数据和指令的交互；与此同时，建立相应数量的软件任务用于与硬件任务并行执行计算任务。在重构过程中，硬件任务在本地完成数据处理和外围辅助电路控制，然后通过存储子系统将处理后的数据转发到系统软件上层，以便其他软件任务或硬件任务调用，最终完成总体的装备测试任务。

2.1　存储子系统的结构设计

在装备测试过程中，应用系统上层需要与硬件重构测试任务进行数据交互，为此，本文设计了存储子系统用于实现 FPGA 底层硬件逻辑与软件上层之间的信息交互。整个存储子系统由内存控制器、任务仲裁器、地址转换器、存储管理单元以及附加控制器组成，存储子系统执行流程见图 2 所示。

2.1.1　硬件重构测试任务向存储系统读取数据

硬件重构测试任务向任务仲裁器申请读数据请求，经仲裁之后，任务仲裁器将读数据地址和读取长度发送到突发访问转换器，转换器根据当前页面的大小对读取地址进行转换，将地址转换为短地址以适应存储页面的大小。突发访问转换器将转换后的地址发送到内存管理单元（MMU，memory management unit），MMU 通过附加控制器从 CPU 获取当前活动进程的转换表基地址作为下面地址转换的基准。MMU 首先在本地的转换检测缓存区（TLB，translation lookaside buffer）中查找相应的地址信息，如果在 TLB 中找到相应的地址信息，则将小页基地址信息返回到物理地址生成逻辑，如果在 TLB 中没有找到相应的信息，那么 MMU 将委托内存控制器向 CPU 查询地址信息，并将生成物理地址依赖的中间地址信息返回，然后在 MMU 中生成相应的物理地址，并将物理地址发送到内存控制器，通过内存控制器向 CPU 读取物理地址中的数据，并将数据经过 AXI 总线返回到硬件任务。

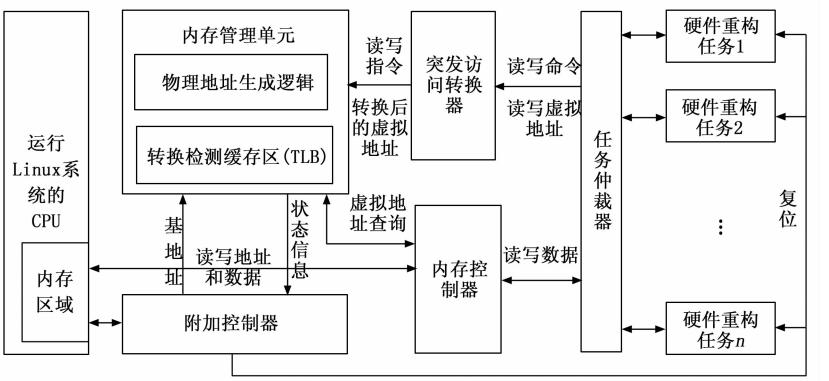

图 2　存储子系统执行流图

2.1.2 硬件重构测试任务向存储系统写入数据

硬件重构测试任务向任务仲裁器申请写数据请求，经仲裁之后，任务仲裁器将写数据地址和写数据长度发送到突发访问转换器，转换器根据当前页面的大小对写地址进行转换，将地址转换为短地址以适应存储页面的大小。写操作的地址转换过程与读操作的一致。内存控制器将写物理地址发送到 CPU，CPU 回复内存控制器写地址准备就绪信号，然后硬件任务通过任务仲裁器向系统内存中的物理地址写入数据。

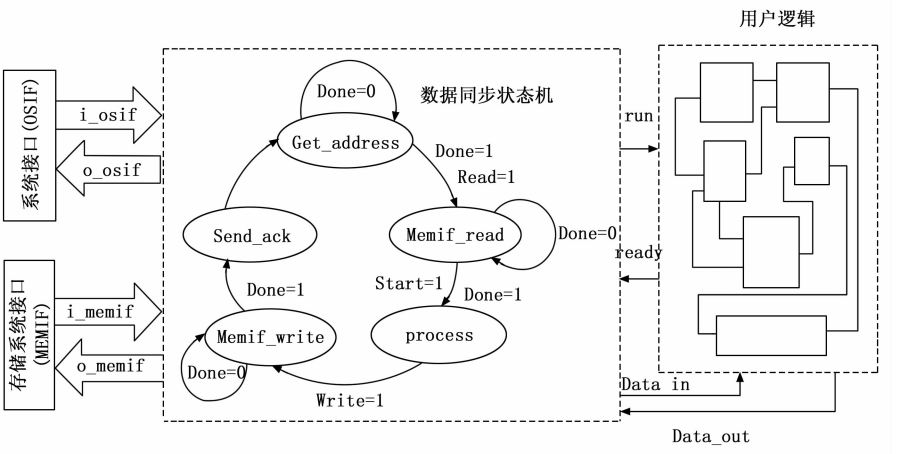

图 3 硬件重构任务结构

2.2 测试任务结构

整个装备测试系统基于 Linux 开源系统。在 Linux 系统中，应用程序是基于多线程编程模型进行开发的，通常由线程、消息队列以及信号量等实体组成。并且 Linux 对这些实体进行了严格的定义和接口说明。因此，为了充分发挥 Linux 实体的强大功能，装备测试系统开放给用户的最小控制单元是重构任务。重构任务分为硬件重构测试任务、软件任务。软件任务是运行在系统 CPU 之上的不占用硬件逻辑资源的任务，主要的功能是实现系统上层对硬件重构测试任务的控制。重构硬件测试任务是放置在指定逻辑位置的基于硬件描述语言实现的单元，其在运行过程中需要接收 CPU 的控制，并且访问系统内存中的数据。下面将重点介绍硬件重构测试任务的设计。

硬件重构测试任务是放置在指定槽位的硬件逻辑，需要接收 CPU 的控制，并且访问存储系统中的数据。为了统一管理，需要每一个硬件重构测试任务都具有相同的接口；同时在硬件重构测试任务访问存储系统的过程中，同一时刻只能有一个任务进行访问，因而，需要建立一个同步机制，使得每一时刻访问存储系统只能有一个硬件重构测试任务。

在每一个硬件重构测试任务中，都有一个同步状态机(synchronization finite state machine)，用来管理每一个任务对装备测试系统的存储子系统访问。

每一个重构任务都有 4 个 FIFO。其中的两个用于硬件重构测试任务与 CPU 之间进行数据交互，另外两个用于存储子系统的访问。同步状态机控制硬件任务与 CPU 之间的数据交互过程，具体过程见图 3 所示。在测试系统开发过程中，用户可以通过修改用户逻辑进而改变每一个硬件重构测试任务的功能，实现飞航装备系统的测试资源在线重配置。

2.3 ICAP 控制器的结构

在 ZYNQ 系列的 FPGA 中，提供了 ICAP 和 PCAP 两种端口支持重构[5]，但是 PCAP 在加载比特文件的过程中会阻塞（block）CPU，这不满足重构系统的要求。因而，采用了 ICAP 端口。

ICAP 控制器主要负责将内存中的硬件重构测试任务的配置文件加载到 ICAP 端口，实现对硬件重构测试任务的在线重配置。在这个过程中，如果直接使用 CPU 进行比特文件的加载，会导致系统频繁进行读写操作，则使 CPU 发生阻塞。同时使用 CPU 进行比特文件加载的速度有限，这会使整个重构过程变的较慢，失去重构任务实时加载的特性。因而，本文采用 DMA（DMA，Direct Memory Access）控制器与 ICAP 相结合的设计。

ICAP 控制器中（图 4 所示），DMA 控制器接收 PS 的控制，直接从外存中将数据加载到位转换状态机中，用于部分重配置，这个过程中，CPU 只负责过程的开始和结束，在传输的过程中并不会占用 CPU 的资源，因而，有效的降低了 CPU 的负载，同时加快了重配置的速度。

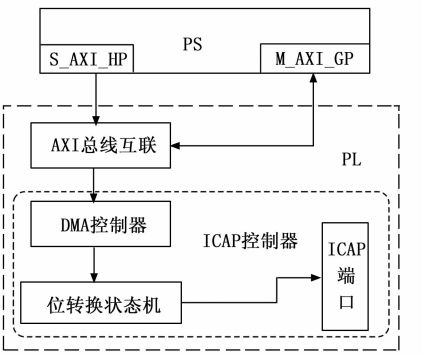

图 4 ICAP 控制器结构

本文提出的 ICAP 控制器支持在线修改比特文件的配置位置，根据比特文件的结构信息，如图 5 所示，在每一个部分比特文件中都有一个独一无二的 32 位地址，称为 FAR(frame address register)，来决定这个部分比特文件配置的起始位置，通过修改部分比特文件的 FAR，实现对部分比特文件放置到其他重构位置。在比特文件中包含有一个 CRC 校验码，其位于比特文件的尾部，用于对配置文件的有效性进行检查。为了使修改之后的比特文件能够正确载入到配置区域，需要将配置过程的 CRC 校验禁止[6-8]。

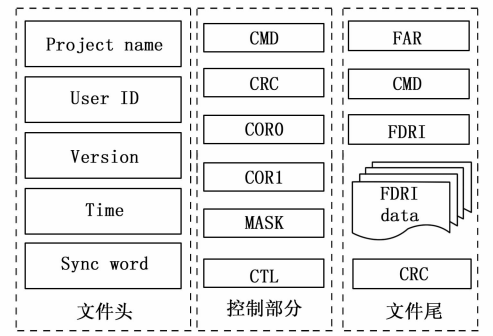

图 5　比特文件结构

在重构装备测试系统中，DMA 控制器和 ICAP 控制器驱动程序部署在 Linux 系统下[9]，实现对 ICAP 控制器的调用与操作，提高了重构系统的二次开发效率。

3　系统验证分析

本文提出的测试系统支持用户根据自身功能需求进行二次开发。开发过程具体如下。第一部分是源文件准备：采用 C 语言编写软件任务；采用硬件描述语言编写硬件任务；根据系统设计修改重构系统的配置文件，以及对系统和平台进行描述。第二部分是生成流程：使用 Linux 下的交叉编译环境对系统的软件任务进行编译生成相应的可执行文件；执行脚本文件，启动 Xilinx EDK 软件工具生成重构系统的系统网表文件以及重构任务的网表文件[10-12]，然后在 Xilinx PlanAhead 软件对重构区域进行划分并生成对应的比特配置文件。在系统运行过程中，通过 FPGA 系统板与上位机交互进行实现。上位机向 FPGA 系统板发送重构请求，系统板接收指令并通过 NFS 文件系统加载指定的比特文件完成硬件重构任务的加载，同时创建软件任务，实现软件任务与硬件任务并行处理数据，在处理数据的过程中，重构硬件任务通过委托任务与系统进行指令交互，并结合存储子系统实现与系统内存的数据交互。系统板最终将数据结果返回上位机进行显示，供用户对结果进行查看。

为了验证系统的有效性，本文基于装备测试系统设计了“硬件重构测试任务＋软件任务”的验证试验，用来验证装备测试系统功能的正确性。首先，针对装备系统测试过程中测试资源的需求，设计了典型的硬件重构测试任务，包含用于实现继电器控制的硬件重构测试任务、用于处理经外围光电隔离电路之后的数字量采集硬件重构测试任务以及用于进行串并转换的 422 通信硬件重构测试任务。实验过程中通过在线重构设计的典型硬件重构测试任务，并通过软件任务实现对硬件重构测试任务的控制，模拟装备系统测试过程，从而验证系统的有效性。

验证重构系统所使用的主控芯片是 Xilinx Zynq-7000 系列 xc7z020clg484-1 FPGA[13]，其内部集成了双 ARM 内核以及具有丰富的片上逻辑资源和外设，能够满足重构测试系统的验证需求。

在验证过程中，首先通过上位机串口向运行于 FPGA 系统板之上的重构测试系统发送测试请求，该请求包含测试过程中所包含的硬件重构测试任务和软件任务，重构系统接收用户的测试请求，并根据指定的软件与硬件重构测试任务的类型，通过网络文件系统读取存放在上位机指定路径的硬件重构测试任务，然后通过 ICAP 控制器完成动态重配置，以及在重构系统上层创建相应的软件任务，实现对硬件重构测试任务的控制。在重构系统测试过程中，硬件重构测试任务并行的实现对数据的采集以及数据的传输，

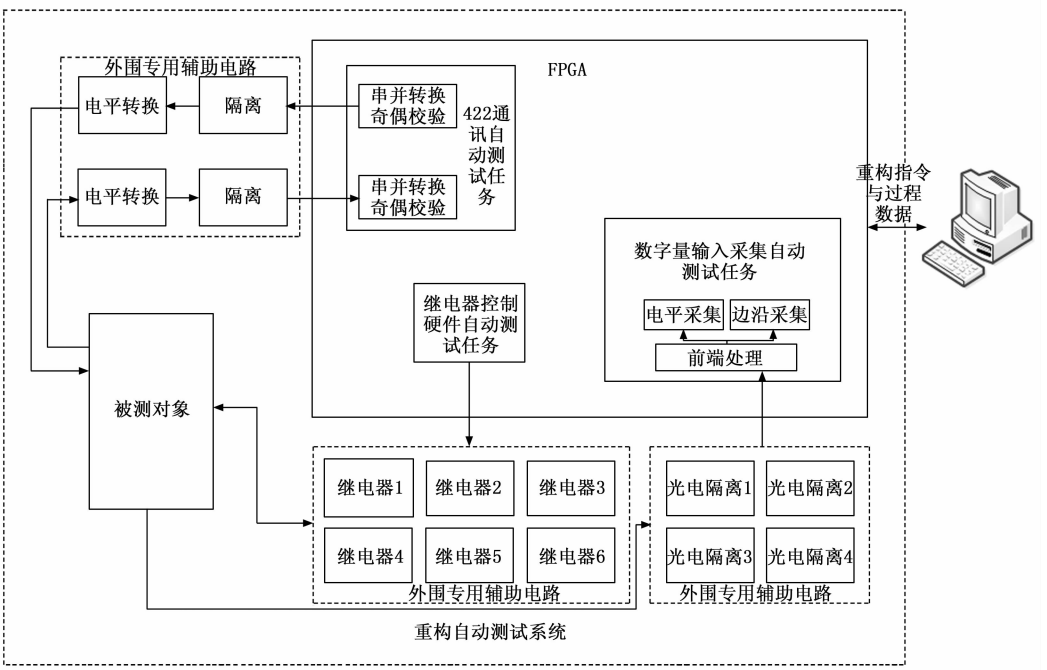

图 6　装备测试系统验证

通过软件任务实现对采集数据的上传和通信数据的交互，最后通过串口将测试过程数据进行输出。

在验证过程中，通过对 422 通讯任务、继电器控制任务和数字量输入采集任务过程数据的记录，具体结果详见表 1 所示，通过对原始数据分析可知，重构装备测试系统创建的硬件和软件测试任务在与被测对象测试中运行正常，各任务之间信息交互正常，且 FPGA 中的控制逻辑与外围的辅助电路功能执行正常。

表 1 结果分析

重构测试系统	被测对象	结果
422 通讯重构任务与被测对象进行数据交互	成功接收到重构任务通讯数据	422 通讯正常
继电器重构任务控制外围辅助电路中的继电器 1－6 闭合和断开	被测对象分别采集继电器 1－6 发出的开关量，并通过 422 通讯链路传输至上位机	继电器重构控制任务运行正常
数字量采集测试任务启动后实时采集被测对象输出的 28 V 状态量，并通过上位机现实采集结果	被测对象按照指定的时序控制对外状态量输出	状态量采集结果正常

传统装备测试系统多数通过采用 PXI 总线架构[14]，各功能板卡均通过 PXI 总线实现互联和系统集成[15]，本文的验证过程如通过传统测试设备实现，则需要 3 块 PXI 功能板卡，分别是 422 通讯板卡、继电器板卡、数字量采集板卡，每块板卡均需要一块 FPGA 作为板卡主控芯片参与逻辑控制，则完成上述验证试验功能需要 3 块具有一定规模的 FPGA 芯片，除此之外，还需要一块标准零槽控制器作为系统的主控制单元[16-20]；而本文提出的基于动态重构技术的测试系统，其仅需要一块具有动态重构功能的 FPGA 芯片，该芯片即作为操作系统的运行载体，又是各重构任务加载的载体，即二者共享 FPGA 的片上资源，极大的提高了 FPGA 资源利用率，同时随着系统所需芯片数量的较少，也极大地降低了系统的硬件成本。

综合上面实验，首先证明了重构系统成功的完成硬件装备测试任务的动态加载以及软件任务的创建。其次重构装备测试系统较传统的装备测试系统体积小、功耗低、在测试资源的分配和管理上具有很好的灵活性。最后，由于重构测试系统减少了 FPGA 芯片的数量，从而降低了测试设备成本，推动装备测试系统向着高度集成化、低成本、测试资源分配灵活的方向发展。

4 结束语

本文针对当前装备测试系统所存在的测试过程实时性差、测试资源冗余以及并发测试能力弱的问题，提出了基于 FPGA 部分动态可重构技术的测试系统，并搭建了实际的验证系统，对重构测试系统的功能进行验证，实验结果证明，本文提出的重构测试系统可有效减少测试系统功能单元的 FPGA 使用数量，提高资源利用率，并充分利用硬件重构任务并发执行特性，提高测试任务实时性；当前重构测试系统仅验证了简单的总线通讯和状态量采集控制功能，尚未开展复查测试任务的充分验证。

参考文献：

[1] Ebrahim A, Benkrid K, Iturbe X, Hong C. A novel high performance fault-tolerant ICAP controller [C] //2012 NASA/ESA Conference on Adaptive Hardware and Systems (AHS), Nuremberg, Germany, Jun, 2012: 259－263.

[2] Williams J A, Bergmann N W, Xie X. FIFO communication models in operating systems for reconfigurable computing [C] // Proc IEEE 13th Symposium on Field-Programmable Custom Computing Machines (FCCM'05), Los Alamitos, American, Apr, 2005: 277－278.

[3] Iturbe X, Benkrid K, Hong C, Ebrahim A, Torrego R, Martinez I. R3TOS: A Novel Reliable Reconfigurable Real-Time Operating System for Highly Adaptive [J]. Efficient and Dependable Computing on FPGAs, IEEE Trans, Computers, 2013, 62 (8): 1542－1556.

[4] Iturbe X, Ebrahim A, Benkrid K, et al. R3TOS Based Autonomous Fault-Tolerant Systems [J]. IEEE Micro, 2014, 34 (6): 20－30.

[5] Styles H, Luk W. Compilation and Management of Phase-Optimized Reconfigurable Systems, The International Conference on Field Programmable Logic and Applications (FPL) [C] // Tampere, Finland, Aug, 2005: 311－316.

[6] Vipin K, Fahmy S A. Automated partial reconfiguration design for adaptive system with CoPR for Zynq [C] // Int. Symp. Field Programmable Custom Computing Machines, Boston, Massachusetts, May, 2014: 202－205.

[7] Ebrahim A, Arslan T, Iturbe X. A Fast and Scalable FPGA Damage Diagnostic Service for R3TOS Using BIST Cloning Technique [C] // The International Conference on Field Programmable Logic and Applications (FPL), Munich, Germany, Sept, 2014: 1－4.

[8] Damaj I, Diab H. Performance Analysis of Extended Vector－Scalar Operations Using Reconfigurable Computing [C] // Proceeding of the ACS/IEEE International Conference on Computer System and Applications. Beirut, Lebanon, 2001: 25－29.

[9] Zebulum R, Stoica A, Keymeulen D. A Flexible Model of a CMOS Field Programmable Transistor Arrays Targeted for Hardware Evolution [A]. Third International Conference on Evolvable Systems: From Biology to Hardware (ICES2000) [C] // Springer-Verlag London, 2000: 274－283.

[10] Silva C F, Alice M. RECASTER: Synthesis of Fault-Tolerant Embedded Systems based on Dynamically Reconfigurable FPGAs [A]. Tokarnia Proceedings of the 18th International Parallel and Distributed Processing Symposium [C] // IEEE, 2004: 38－45.

(下转第 101 页)