文章编号:1671-4598(2020)07-0148-05 DOI:10.16526/j.cnki.11—4762/tp.2020.07.030 中图分类号:TP273 文献标识码:A

基于 LabWindows/CVI 的远程接口单元测试系统软件设计

李 亚,邵引平
(西安翔迅科技有限责任公司,西安 710068)

摘要:针对航空机电系统中远程接口单元(RIU)信号种类多、总线接口控制文档数量大、功能复杂的特点,测试软件采用模块化设计思想,基于 LabWindows/CVI 平台开发,将硬线配置、总线管理、测试流程、用户管理等信息都存储于数据库中,降低应用软件的复杂度、提高软件的灵活性、可维护性;测试软件提供了自动测试功能,操作简洁,减轻工作人员测试工作量;自动测试流程可在数据库中编辑,通用性好,灵活性高;实验应用表明,该软件功能满足测试需求,性能稳定,已成功应用于远程接口单元的现场测试中。

关键词:LabWindows/CVI;数据库;自动测试;通用性;灵活性

Design of Remote Interface Unit Testing System Software Based on LabWindows/CVI

Li Ya, Shao Yinping
(Xi'an Xiangxun Technology Co., Ltd., Xi'an 710068, China)

Abstract: The remote interface unit in the aviation electromechanical system is characterized by multiple types of signals, large number of ICDs and complex functions. For this the test software adopts modular design idea, based on LabWindows/CVI platform development, to store the hard line, bus management, test procedures and user management in a database, reduce the complexity of the application software, improve the software's flexibility and maintainability. The automatic test process can be edited in the database, with good generality and high flexibility. The application proves this program meets the testing needs and has excellent performances in stability. It has been successfully applied to the field test of remote interface unit.

Keywords: LabWindows/CVI; database; automatic test; generality; flexibility

0 引言

航空机电系统是飞机中执行飞行保障功能子系统的总称,主要包括供燃油系统、动力系统、供电系统、液压系统、环境控制系统等。远程接口单元(RIU)隶属于机电系统,是新型机载机电系统的终端信号处理设备,主要完成与其交联机电子系统的状态采集与输出控制[1-2]。远程接口单元具有设备数量多,信号种类多,测试压力大的特点。

远程接口单元测试系统硬件秉承模块化、通用化的原则,基于 PXI 系统平台设计,采用虚拟仪器技术,使各个功能模块独立运作,可靠性好,耦合度低[3-4]。

测试系统软件需要基于系统硬件完成板卡的驱动与管理,测试流程的处理与控制,测试结果的显示与保存,提供人机交互界面等工作,软件是测试系统的灵魂。开发操作简洁、通用性好、灵活性高的软件是测试系统软件的发展趋势。传统测试软件大多将信号的配置在程序中写定,将测试流程按执行顺序也固化在代码中。这样造成软件的通用性、维护性差,且面临开发周期长,调试复杂度高等问题。此测试软件开发基于 LabWindows/CVI 开发平台,平台具有很好的虚拟仪器开发工具,控件丰富,广泛应用于测控领域[5-7]。测试软件将硬线配置、总线管理、测试流程、用户管理等都在数据库中完成,通过 SQL Toolkit 工具包访问 Access 数据库,将大量配置信息通过数据库存储,便于修改和扩展;软件采用自动测试的形式,人机交互界面操作简单,减轻工作人员测试压力;软件采用层次化结构设计,自底向上,将复杂应用细节化,结构清晰,灵活易用。

1 系统概述

为了满足远程接口单元的测试需求,确保测试效率及稳定性,减少人工干预,针对测试内容和接口特性,测试系统需满足以下要求:

1)信号输入和输出接口种类及数量需要覆盖待测试 RIU 的接口需求,并留有不小于 10%的备份。

2)采用成熟的数据传输技术,测试系统应具备较大的带宽余量,满足未来可能存在的数据增长的需求。

3)硬件应具备可靠性、维护性、稳定性。

测试系统硬件由工控机系统、信号调理系统和独立的

收稿日期:2019-11-26; **修回日期**:2019-12-30。

作者简介:李 亚(1991-),女,河北乐亭人,硕士,工程师,主要从事航空机电、航电设备测试技术方向的研究。

邵引平(1967-),男,陕西宝鸡人,高级工程师,主要从事自动测试技术方向的研究。

电源系统组成：包括电源箱、PXI 工控机系统、调理箱、Nport（串口通讯服务器）、设备供电直流电源、断连板部分和线缆部分。测试信号包括离散量输入和输出，模拟量的输入和输出，电阻信号、功率信号、HB6096 总线信号、GJB289A 总线信号等。系统硬件总体架构如图 1 所示。

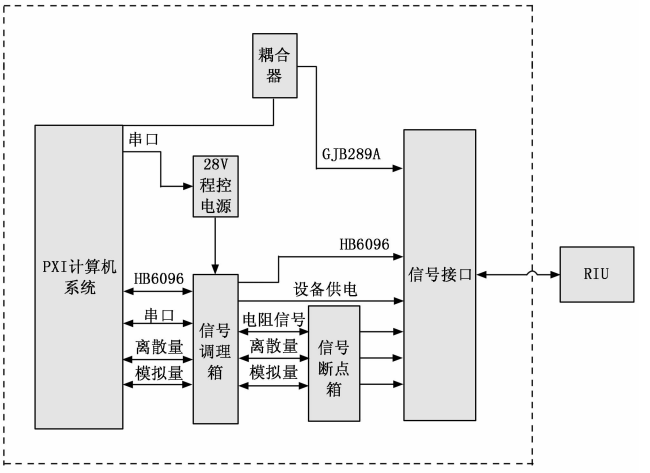

图 1　测试系统硬件总体架构

测试系统使用环境为联试实验室，采用标准机柜框架实现，机柜间尽量减小线缆使用，便于拆卸和运输。

测试系统的工作原理如下：首先将测试系统通过线缆与 RIU 相连。打开测试软件，测试 RIU 的硬线输出接口时，测试系统通过 GJB289A 总线输出激励信号，控制 RIU 输出指定信号并进行硬线采集操作、显示采集数据，判断 RIU 的硬线输出功能是否正常；测试 RIU 的输入接口时，测试系统按需求输出相应信号的激励值，并通过 GJB289A 总线读取 RIU 的反馈信息加以显示，据此判断 RIU 的硬线采集功能是否正常。RIU 的 HB6096 总线信号测试时，测试系统设置 HB6096 总线激励信号，接收 GJB289A 总线读取 RIU 的反馈信息加以判断。

2　软件详细设计

远程接口单元仿真测试系统软件的开发秉承“高效、易用、稳定、美观、可扩展”的设计原则，采用模块化的设计方法，在 LabWindows/CVI 开发平台下开发完成。按照软件功能需求将程序划分为启动、系统管理、测试功能、退出等几大模块，整个软件结构清晰，便于维护和升级。模块划分图如图 2 所示。

测试软件主界面如图 3 所示。

测试软件执行的流程图如图 4 所示。

2.1　软件结构

远程接口单元测试软件采用层次化结构设计，软件结构如图 5 所示。系统分层结构主要由硬件操作层、数据访问层、业务逻辑层和人机交互层组成。

硬件操作层是软件与硬件紧密结合的层次，负责完成具体的硬件操作和数据通讯，包括总线数据的收发、离散量输入输出、模拟量输入输出等，其中总线 HB6096，总线

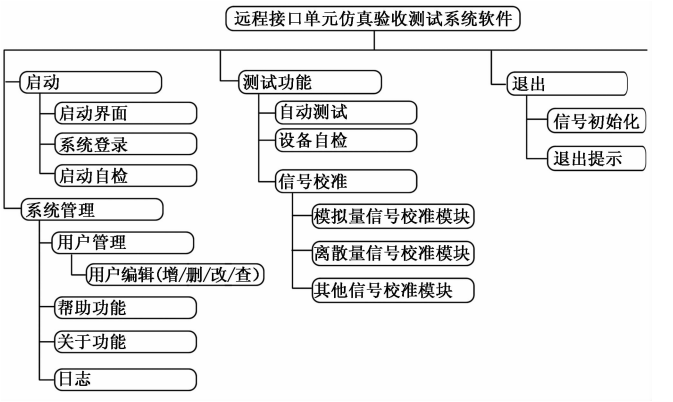

图 2　软件模块划分框图

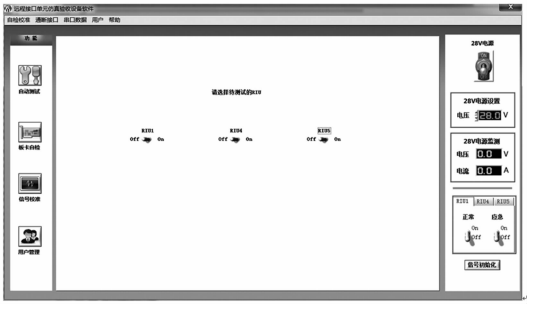

图 3　软件主界面

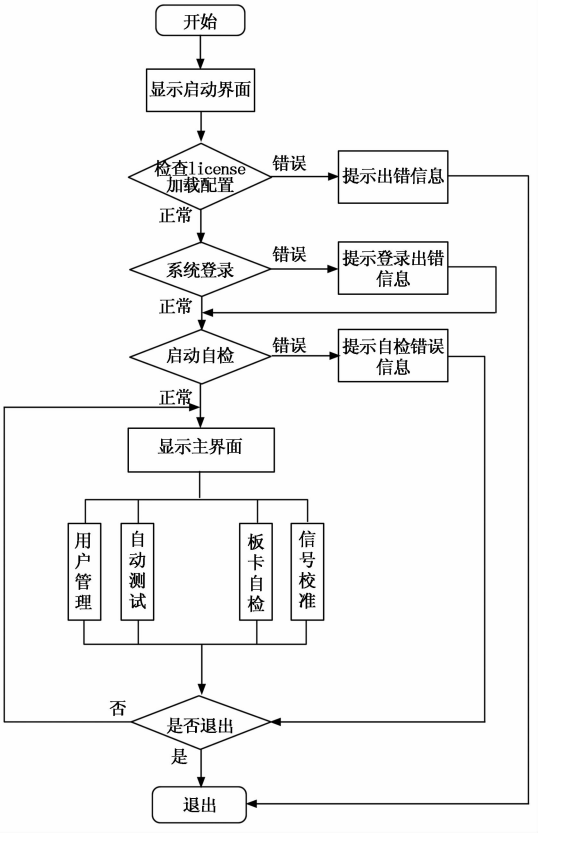

图 4　软件流程图

GJB289A 的操作通过 IOServer（总线管理程序）控制，在

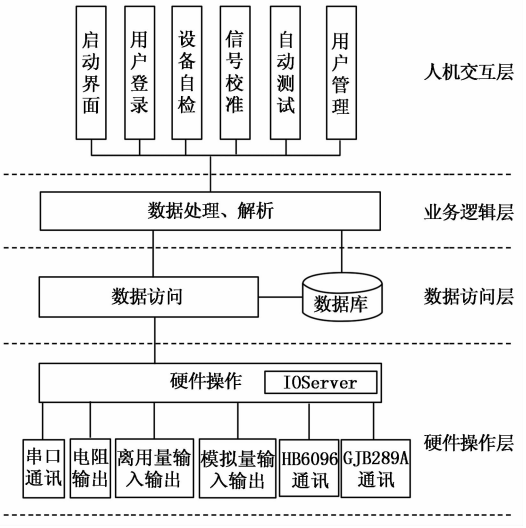

图 5 系统分层结构图

2.3 章节中详细解释。

数据访问层是访问数据库信息的层次，数据库存储了硬线信号、总线信号、测试流程、用户信息等数据。合理设计的数据库功能强大，可优化软件数据结构，精简代码，配合软件实现应用功能。软件获取数据库内容后便可向下操作硬件信号，向上进行数据处理与控制，是承上启下的重要一环。

业务逻辑层是实现测试软件功能的核心，将系统校准、系统自检、信号测试、系统管理的逻辑都包含其中，精炼合理的业务逻辑设计是软件可靠性的保障。系统校准可以进行硬线输入输出信号尤其是模拟量信号的校准，并保存校准系数于数据库中；系统自检对所有硬线、总线信号进行打开、自检、初始化操作，为信号测试做准备，如果有信号自检失败，将弹框提示操作者；信号测试是软件的核心功能，为了方便操作者使用，减轻操作负担，软件提供的是自动测试；系统管理包含用户管理、帮助、关于、日志等功能。

人机交互层主要由启动界面、用户登录界面、板卡自检界面、信号校准界面、自动测试界面等交互界面组成，需求简洁明了，易于操作。

2.2 数据库设计

为了更好的简化软件流程代码的编写，增强软件的灵活性，软件设计时将信号信息、测试信息合理设计，配置于数据库中。数据访问层设计了资源配置库进行数据存储。数据表包括板卡资源描述表、硬件信号软件配置表、串口通道配置表、用户管理表、自动测试流程表、自动测试激励值表、自动测试期望值表，以此来配置软件的板卡信息、串口信息、硬线信号信息、自动测试信息、用户信息。测试软件通过 SQL 操作访问数据库的内容，在程序中设计合理数据结构、保存成队列或哈希表的形式，进而参与信号操作与数据处理。

其中硬件信号软件配置表是软件操作底层硬线资源的基础。此表中设计了信号名称、信号类型、所属板卡、通道号、所属设备、校准系数、公式系数、最大值、最小值、航插号、缺省值等属性。根据此表的信息即可方便完成信号的输出、采集、校准、量纲转换、范围判定、初始化等操作和设置。

通过资源配置库可以使程序处理无需固化大量信息于缓冲区内。代码中只含操作和控制方法、人机界面交互的内容，代码整洁轻量，修改方便灵活。这样当遇到板卡通道变更、串口波特率变化、初始化值变更、测试流程增减修改等情形时，只需要修改数据表即可，易于维护和升级。

2.3 总线管理

众所周知，航空总线协议较为复杂[8]，编程难度较大。尤其 GJB289A 总线，是一种分时指令/响应型多路传输数据总线[9]。有 3 种类型的终端，分别是 BC（总线控制器）、RT（远程终端）、总线监视器（BM）；信息格式有 BC 到 RT、RT 到 BC、RT 到 RT、广播方式和系统控制方式；软件编码时除了板卡操作，还需要对消息链、重试条件、矢量字等进行配置，较为复杂，且不同此厂家的板卡配置使用方法千差万别，难度大。为此设计的 IOServer（总线管理程序），将总线操作都整合起来，软件工程师通过配置总线数据库即可管理总线，无论哪个厂家的板卡，都可通过同样的接口函数调用实现归一化操作，在测试软件中仅通过调用表 1 中几个函数即可实现总线的启动和收发。

表 1 IOServer 操作

LaunchIOServer	启动 IOServer 控制台进程
StartIOServer	执行总裁通讯任务
StopIOServer	停止总线通讯任务
ReadShareBuffer	读共享内存
WriteShareBuffer	写共享内存
TerminatorIOServer	退出 IOServer

IOServer 的关联数据库中，每种总线单独设计一个数据库。GJB289A 总线定义于 MbiComm. mdb 中，软件工程师依次按实际应用进行板卡配置、通道配置、协议配置、协议通道映射、信号定义进行数据表的填写，提供的字典表方便用户配置通讯模型。配置完成之后在测试程序中启动 IOServer，IOServer 就会将数据表中的内容整合生成共享对象，应用程序通过共享对象的读写即可访问总线数据，大大降低应用程序中总线操作的使用难度。HB6096 总线定义于 Arinc429Comm. mdb 中，数据库配置内容与 GJB289A 总线不同，但使用方式相同，都是通过共享内存访问。

远程接口单元总线的接口控制文档（icd）也在这里配置，它包含了整套总线系统的数据定义，包括名称、信号类型、起始位、长度、精度、最大值、最小值等信息[10]。程序访问数据表中的 icd 内容，可以完成信号的显示、解析、组包操作。采取这种总线管理的方式，开发、测试效率显著提高。

2.4 自动测试

测试软件的测试内容包括硬线信号测试、HB6096 总线

测试、GJB289A 总线测试。硬线信号测试按类型可分为地/开离散量输入测试、28 V /开离散量输入测试、28 V 直流 0.1 A 输出、地/开输出、28VPWMSSPC/3A 功率输出、28 V/开 1A－SSPC 功率输出、28 V/开输出、28 V 状态输出、4～20 mA 输入、PT1000 输入、28 V 直流输入、28 V 直流电压状态采集输入、900～1 700 欧电阻输入、电阻开关门限输入、115 V 交流电压状态采集、0～10 V 输入等。信号种类 20 有余，数量高达 600 多路，通过硬线信号与总线信号的输入输出组合逻辑进行测试。如果采用手动测试的方式，那么工作量繁重。为此测试软件设计了自动测试的功能[11]，极大的减少测试人员工作负担。

自动测试程序如果按执行流程在程序中写定，那么会导致软件的通用性、维护性差。所以测试软件采用在数据库中配置自动测试信息的方式。自动测试信息由三张表——自动测试流程表、自动测试激励值表、自动测试期望值表配置完成。自动测试流程表定义测试的内容，属性包含测试系统、测试项目、测试信号、测试步骤、激励信号索引、期望信号索引。激励信号索引在自动测试激励值表中定义，详细描述激励信号的信号名称，信号类型、设定值等。期望信号索引在自动测试期望值表中定义，详细描述期望信号的信号名称，信号类型、预期值、预期容差等。在测试软件中遍历自动测试流程表的内容，执行激励信号操作，通过比较测试数据和预期值判断测试结果。

自动测试界面如图 6 所示，左侧的树形控件是测试项目，可以自主选择测试项；上方有测试方式选择，测试进度显示，测试控制按钮，工作指示灯等控件进行控制和显示；主体是测试表格，靠左的表格显示详细测试内容，靠右的表格显示测试项目的测试结果。测试结束后会自动生成测试记录，测试记录与下图中右侧的表格内容一致。

图 6　自动测试界面图

2.5　功能函数的编程实现

2.5.1　硬件信号信息获取编程实现

程序访问资源配置库，读取数据库中“硬件信号软件配置表”的内容，在程序中保存为链表或哈希表。不同的存储结构应用于不同的功能。函数为 Access _ GetSignalInfo（int hDB，char tableName [])。其中有 SignalField 和 CALIBRATION 两个重要数据结构存储内容。SignalField 结构对应硬件信号软件配置表的属性，并添加了设置值，实际值等属性，CALIBRATION 结构定义如下：

```
typedef struct
{
  int   IndexID;   // 索引 ID
  char keyChar[30];  // 键值
  char pinChar[30];  // 测试引脚
}CALIBRATION;
```

在 Access_GetSignalInfo 函数中创建以下 CALIBRATION 结构链表：

```
g_ListOfAI＝ListCreate(sizeof(CALIBRATION)); //模拟量输入信号
g_ListOfAO＝ListCreate(sizeof(CALIBRATION)); //模拟量输出信号
g_ListOfDI＝ListCreate(sizeof(CALIBRATION)); //离散量输入信号
g_ListOfRE＝ListCreate(sizeof(CALIBRATION)); //电阻输出信号
g_ListOfDODK＝ListCreate(sizeof(CALIBRATION)); //地开离散量输出信号
g_ListOfDOVK＝ListCreate(sizeof(CALIBRATION)); //28V开离散量输出信号
g_ListOfSWITCH＝ListCreate(sizeof(CALIBRATION)); //继电器信号
```

创建包含全部信号以信号缩写为键值的哈希表 HashOfSignal：

```
HashTableCreate (count＋20, C_STRING_KEY, 0,sizeof(oSignalField),&g_HashOfSignal);
```

两种结构配合主要参与校准功能的实现。

还需要针对细分的信号类型保存 SignalField 链表：

```
GetDODKList();
GetDO28VKList();
Get28V0P1AList();
GetDKOList();
Get28PWMSSPCList();
GetI28VKList();
GetI28V5KList();
Get4TO20MAList();
Get0TO10VList();
Get28VList();
GetPT1000List();
GetRESwitchDLList();
```

以上结构应用于自动测试。

2.5.2　IOServer 应用编程实现

IOServer 在执行 LaunchIOServer（dir）后运行控制台进程，根据总线的配置生成共享内存，完成板卡初始化、自检工作。之后可以在系统自检完成后，调用 StartIOServer（）函数，运行 IOServer，执行总线通讯任务。

在接收 429 或 1553 总线消息的函数中：

ReadShareBuffer （ bufferInfo. bufferName， rcvBuf，&len)，根据 bufferInfo. bufferName（共享内存名）和 len（缓冲区长度）读取 429 总线对应通道或者 1553 对应消息的

全部数据。IOServer 本身会根据配置信息控制板卡接口总线上的数据，存入共享内存。

在发送 429 或 1553 总线消息的函数中：

WriteShareBuffer（sharedField. BufferName，c _ Buf，&len），向 sharedField. BufferName（共享内存名）中写入 len（数据长度）的 c _ Buf（char 型数组）数据，IOServer 会根据共享内存的配置控制板卡发送数据。

软件关闭时调用 StopIOServer（）停止总线通讯，调用 TerminatorIOServer（）结束控制台进程。

2.5.3 自动测试编程实现

自动测试是测试软件的核心功能。为了保证界面的正常刷新，自动测试单独创建一个线程：

```
CmtScheduleThreadPoolFunction (DEFAULT _ THREAD _ POOL_HANDLE,ThreadFunc_AutoTest,0,&ThreadID_Auto);
```

测试线程中，逐层遍历测试项列表，在测试步骤层级，根据不同测试信号类型，执行不同的激励输出：

```
for(int i=0; i<numIncent; i++)
{
  //测试记录
setValue = atof(ArrayIncent[i]. setVal);
if(strcmp(ArrayIncent[i]. signalType, "硬线") 0)
{
IncentHardwareSignalOp(ArrayIncent[i]. signalName, setValue);
}
else if(strcmp(ArrayIncent[i]. signalType, "1553 总线") 0)
{
Incent1553SignalOp(ArrayIncent[i]. signalName, setValue);
}
else if(strcmp(ArrayIncent[i]. signalType, "429 总线") 0)
{ Incent429SignalOp(ArrayIncent[i]. signalName, setValue);
}
else if(strcmp(ArrayIncent[i]. signalType, "Delay") 0)
{
  Delay(setValue);
}
}
```

然后根据不同信号类型，采取不同采集/接收、解析数据方法，获取实际数据：

```
for(int i=0; i<numExpect; i++)
{
//测试数据获取
if(strcmp(ArrayExpect[i]. signalType, "硬线") 0)
{
realVal = ExceptHardwareSignalOp (ArrayExpect [i]. signalName, *loadMult, i+1);
   *loadMult = 1;
}
else if(strcmp(ArrayExpect[i]. signalType, "1553 总线") 0)
{
   realVal = Except1553SignalOp (ArrayExpect [i]. signalName);
}
else if(strcmp(ArrayExpect[i]. signalType, "429 总线") 0)
{
   realVal = Except429SignalOp (ArrayExpect [i]. signalName);
}
}
```

最终根据期望值进行判断测试是否通过，如果期望的预期值容差为 0，则直接判断期望值；如果预期值容差为正无穷，最大值为极大值（999999），最小值为 expectVal，需判断范围；如果预期值容差为负无穷，最大值为 expectVal，最小值为极小值（－999999）需判断范围；其他最大值为 expectVal＋预期值容差，最小值为 expectVal－预期值容差。

```
if(atof(ArrayExpect[i]. errRange) 0 && strlen(ArrayExpect[i]. errRange) 1) //预期值容差 = 0
{
  expectValue = atof(ArrayExpect[i]. expectVal);
  expectFlag = 0;
}
else
{
  if(strcmp(ArrayExpect[i]. errRange, "正无穷") 0)
  {
  expactMax = 999999;
  expactMin = atof(ArrayExpect[i]. expectVal);
  }
else if(strcmp(ArrayExpect[i]. errRange, "负无穷") 0)
  {
  expactMin = －999999;
  expactMax = atof(ArrayExpect[i]. expectVal);
  }
  else
  {
  expactMin = atof(ArrayExpect[i]. expectVal)－ atof(ArrayExpect[i]. errRange);
  expactMax = atof(ArrayExpect[i]. expectVal)＋ atof(ArrayExpect[i]. errRange);
  }
  expectValue = atof(ArrayExpect[i]. expectVal);
  expectFlag = 1;
}
```

3 结束语

远程接口单元仿真测试系统是基于 PXI 总线，可测量多种型号远程接口单元的综合测试设备，功能全面，可靠性高。远程接口单元仿真测试系统软件基于 LabWindows/CVI 平台，界面友好易用，通过数据库技术进行资源配置，

（下转第 157 页）