LabWindows/CVI在半实物分布式仿真系统中的应用

李 海， 武小栋

北京理工大学 电子工程系，北京100081

2008-09-09

摘 要： 以一个通信仿真系统的开发为例，针对传统的LabWindows/CVI调用外部DLL的方法不能应用于HLA系统的问题，提出创建LabWindows/CVI的DLL和利用外部编译器两种解决方案将LabWindows/CVI与HLA仿真系统相结合。

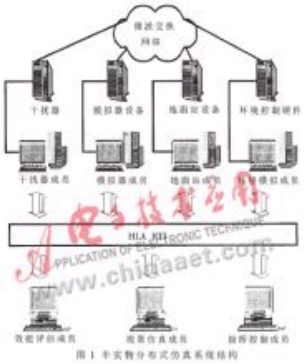
关键词： LabWindows/CVI 虚拟仪器 半实物分布式仿真系统 HLA RTI

半实物仿真又称为硬件在回路中的仿真(Hardware-in-the-loop Simulation)，是一种将实物接入仿真回路中的仿真试验。半实物仿真已经成为航空航天、武器系统等研究领域不可缺少的重要手段。随着计算机仿真技术的迅猛发展，仿真系统变得越来越复杂，许多复杂的仿真牵涉到一些不同类型系统的仿真联合，而这些系统各自的仿真环境组成了整个仿真环境。为了解决不同类型的仿真间的互操作和仿真部件的重用，美国国防部公布了建模与仿真领域里的高层体系结构HLA(High Level Architecture)。HLA正日益得到重视和广泛的应用，基于HLA的分布式系统仿真在半实物仿真中的比重日益增加[1]。而另一方面，NI公司的LabWindows/CVI集成了GPIB、VXI、PXI、RS232/485和插入式(PCI、USB)数据采集设备等通信的功能，支持DataSocket和TCP/UDP等技术与远程应用程序通信。作为虚拟仪器开发工具在数据采集和界面控制方面具有明显的优势。所以，将LabWindows/CVI应用于半实物分布式仿真非常有利于系统的快速开发。

本文以一个通信对抗仿真系统的开发为例，分析了LabWindows/CVI应用于基于HLA的半实物仿真系统所存在的问题，并提出两种解决方案将LabWindows/CVI与HLA仿真系统完美结合。

1 半实物仿真系统结构

笔者所开发的半实物通信对抗仿真系统中的干扰机和通信设备均为硬件实物，而干扰机和通信设备之间的电磁环境是使用软硬件模拟产生的。系统的简化框图如图1所示。



该仿真系统共有七个HLA成员，HLA成员之间通过RTI发送和接收数据。各成员的主要功能如下：

· 指挥控制成员负责仿真场景与系统参数的设置，通过HLA进行时间推进，并不断发送控制参数给各成员进行多种侦察和干扰实验。

· 干扰器成员、模拟器成员和地面站成员将从HLA系统接收到的仿真命令发送给硬件实物并根据从硬件采集到的数据进行分析，将捕获时间、误码率等信息发送给HLA中的其他成员。

· 环境模拟成员随着仿真推进实时计算各设备的位置、姿态、速度、距离衰减和多普勒频移，并控制连接各设备的微波网络，以模拟通信设备之间的链路。

· 效能评估成员根据从实物采集到的数据计算干扰效能。

· 视景仿真成员显示各硬件实物在仿真场景中的位置、姿态和其他信息。

由于设备的用途、使用环境各异，考察的指标各有侧重，设备实物和工控计算机之间的连接类型也各不相同，见表1。借助LabWindows/CVI的数据采集和处理能力，可以使每台工控机变成一台虚拟仪器，完成控制数据的发送、遥测数据的采集和实时显示功能。这些工控机程序同时又是HLA的仿真成员，在HLA的协调下工作。工控机的双重使命使研究如何将虚拟仪器技术和分布式仿真平台HLA结合成为必要。

热点专题

- 信心09,冬天来了,春天还会远吗?
- 低功耗技术，是鸡还是蛋?
- 华北计算机系统工程研究所(电子六所)总结表彰暨春节联欢会
- Powerwise高效能解决方案
- 2008Security China中国国际社会公共安全产品博览会
- 视频信号处理技术
- 2008嵌入式技术创新及...
- 2008飞思卡尔技术论坛
- Altera公司SOPC...
- 第十届高交会电子展
- 科技闪耀北京奥运
- ADLINK DAY—2008年量测与自动化技术国际高峰研讨会
- 中国电子学会Xilinx杯开放源码硬件创新大赛
- 赛灵思公司Virtex-5系列FPGA
- 3G知识
- IPTV
- 触摸屏技术
- RoHS

杂志精华

- 基于CC2430的无线传感器...
- 无线传感器网络应用系统综述
- 无线传感器网络在野外测量中的...
- 基于竞争的无线传感器网络
- 用于矿井环境监测的无线传感器...
- 具有自适应通信能力的无线传感...
- 基于传感器网络技术的深孔测径...
- 基于无线传感器网络的家庭安防...
- 基于ATmega128L与C...
- 无线传感器网络中移动节点设备...

表 1 仿真成员和实物之间的硬件接口关系

仿真成员	接口类型	用 途
干扰器成员	RS485	遥控指令和回传状态。
	RS422	回传中频采集数据。
模拟器成员	CAN 总线	遥控指令和总传误码率。
	7 路 CMOS 开关信号	备份加电控制接口。
	40 路 CMOS 开关信号和 10 路 0~5V 模拟信号	模拟器实物的状态监测。
	RS422 和 LVDS	回传 1 组中频采样数传、3 组解调数据数传和 3 组反向数传数据
地面站成员	100M 以太网	遥控指令和回传状态。
环境模拟成员	USB 2.0 高速模式	指令控制和修正数据。

2 LabWindows/CVI应用于HLA系统的两种解决方案

LabWindows/CVI提供了多种与其他开发工具的编程接口，其中最常用的方法是以LabWindows /CVI编写主程序，使用其他开发工具编写DLL(Dynamic Link Library)，再将DLL加入到LabWindows/CVI系统中使用[2]。这种方法对一般的硬件采集和处理非常简便，但是对于本文所讨论的HLA系统无法适用。所有HLA应用均通过RTI接口库调用HLA所提供的功能，RTI库一般以C++或Java类库的形式提供，而LabWindows/CVI只能调用使用C语言接口的DLL库文件。曾经有人尝试过将HLA 的服务封装成MEX(Matlab规定的C语言接口的DLL)[3]，但一直没有进入实用阶段。因为RTI提供100多种仿真服务，要将这些复杂的服务都以C语言接口的形式封装成DLL，不但费时费力，而且必然削弱HLA作为面向对象的分布式仿真平台的功能。所以，必须寻找其它方法解决这个问题。在实际开发中，笔者先后使用以下两种方案解决前述问题。

2.1 基于LabWindows/CVI的DLL方法

通常的LabWindows/CVI程序开发是在其集成环境中编写C程序，最终编译生成可执行文件(.exe)。采用基于LabWindows/CVI的DLL方法与此不同。该方法将LabWindows/CVI程序编译成为DLL库，然后在Visual C++编写的主程序中加以调用。软件的主程序使用Visual C++编写，可以调用RTI库加入HLA仿真系统，并且调用LabWindows/CVI编写的DLL所提供的输出函数进行界面显示、数据采集与控制。当用户在界面上进行操作或者数据采集完成时，LabWindows的DLL程序可以借助Visual C++主程序提供的回调(callback)函数通知主程序，并将数据发送给主程序。软件各模块的调用关系可用图2表示。

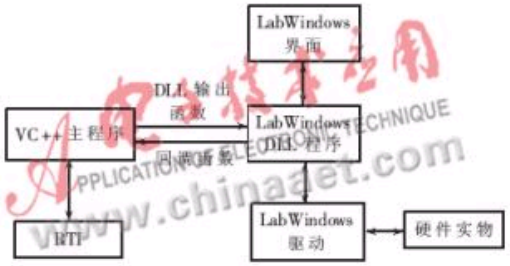


图 2 基于 LabWindows/CVI 的 DLL 方法的调用关系

要生成DLL库，需要在LabWindows/CVI中选择菜单“Build | Target Type | Dynamic Link Library”。选择菜单“Edit | Insert Construct | DllMain”向.c文件中加入DLL所需要的DllMain函数。

LabWindows/CVI中最主要的工作是编写DLL输出函数。例如下面的CallCVI函数显示LabWindows的用户面板TestUI.uir：

```
int __stdcall CallCVI ()
{
if ((panelHandle = LoadPanelEx (0, 'TestUI.uir',PANEL,__CVIUserHInst)) < 0)
    return -1;
DisplayPanel (panelHandle);
RunUserInterface ();
DiscardPanel (panelHandle);
return 0;
}
```

这里使用LoadPanelEx函数，而不是通常使用的LoadPanel函数。如果使用LoadPanel函数，在Visual C++中运行时会因为找不到.uir文件而报告错误。在LabWindows中要实现DLL输出函数，需要将函数的声明写到一个头文件(.h)中，不要直接将DLL输出函数的声明写到面板对应的.h文件中。因为修改面板时，LabWindows/CVI会重新生成.h文件，从而丢失手工添加的定义。建立一个export.h加入到工程中，在export.h中加入CallCVI函数的定义；再选择菜单“Build | Target Settings”，点击对话框中Exports框架的Change按钮。在“DLL Export Options”对话框中，设置“Export What”为“Include File Symbols”，并选中export.h。最后选择菜单“Build | Create Debuggable Dynamic Link Library”就可以创建.dll库文件和.lib文件。

将.dll、.lib、.uir文件和export.h文件拷贝到Visual C++的工程目录下，从Visual C++菜单中执行“Project | Add To Project | Files”，将.lib和export.h添加到工程中。在程序中调用DLL中的函数CallCVI()即可显示LabWindows/CVI的面板。借助DLL输出函数的参数，可以实现从Visual C++程序向LabWindows/CVI程序发送数据。

反过来，如果希望将LabWindows/CVI程序中的用户操作或外部输入数据传递给Visual C++，可以通过由Visual C++提供回调函数来实现。可以修改前面的CallCVI的定义为：

```
typedef int (CVICALLBACK*VCPROC)(void*callbackData);
static VCPROC pCallbackFunc = 0;

int __stdcall CallCVI (void* pFunc)
{
```

```
        pCallbackFunc = (VCPROC)pFunc;
        ……
    }
```

这里使用typedef定义了一个回调函数类型VCPROC。VCPROC实际上是一个函数指针，函数的参数为void类型。函数指针VCPROC的返回值和参数可以根据实际应用的情况灵活修改。Visual C++调用CallCVI时，需要提供一个回调函数的地址作为参数，如：

```
    CallCVI((void*)HLAProc);
```

这里的HLAProc 是Visual C++程序中的一个用户函数，其参数和返回值都要与VCPROC中的定义一致。CallCVI中，LabWindows/CVI程序将回调函数的地址保存在全局变量pCallbackFunc中。当LabWindows/CVI中某个事件发生时，可以调用pCallbackFunc变量中所保存的函数指针通知Visual C++程序。下面的例子中，用户点击LabWindows/CVI程序面板上的按钮后，程序调用回调函数发送一个字符串给Visual C++，执行加入HLA联邦的操作：

```
    int CVICALLBACK ClickCallback (int panel, int control, int event, void *callbackData, int eventData1, int eventData2)
    {
    switch (event)
        {
        case EVENT_COMMIT:
            if(pCallbackFunc)
            { // 加入HLA联邦
                (*pCallbackFunc)((void*)'Join Fed');
            }
            break;
        }
    return 0;
    }
```

在Visual C++函数HLAProc函数中，可以实现对RTI的CreateFederation、JoinFederation等服务的调用。

LabWindows/CVI 编写的DLL也可以进行源代码级调试，但需要做一些额外的设置。先在Visual C++中编写好调用DLL的程序并编译为.exe文件，然后在LabWindows/CVI中打开工程，选择菜单“Run | Specify External Process”，指定Visual C++编写的.exe文件作为外部运行程序。在LabWindows/CVI程序中设置断点后，选择菜单“Run | Debug Project”进行程序调试。LabWindows/CVI将先启动Visual C++编写的程序，当执行到LabWindows/CVI的DLL内部的代码时即可进行源代码级跟踪调试。

2.2 基于外部编译器的方法

另一种解决方案是利用Visual C++作为外部编译器来实现HLA与LabWindows/CVI程序之间的相互调用。LabWindows/CVI自身编译器的核心是David R. Hanson开发的lcc，这个编译器以容错和调试为擅长，但是优化性能较弱[4]。使用外部编译器还可以生成优化的代码，从而提高程序运行效率。

使用这种方法，先借助LabWindows/CVI生成显示面板和硬件操作所需要的C语言代码，然后把代码和LabWindows/CVI的头文件和库文件添加到Visual C++的工程中进行编译。由于LabWindows/CVI生成的代码与Visual C++生成的代码是在同一个工程中，可以直接相互调用，不需要像上一种方法那样设计DLL输出函数和Visual C++回调函数。

要使用外部编译器，最主要的工作是为LabWindows的面板生成对象表。在LabWindows/CVI环境中开发程序时，LabWindows/CVI会自动为所有面板上所有控件的回调函数建立一个对象表并链接到程序中，运行时利用这个对象表将.uir文件中的控件对象和控件的回调函数对应起来。而使用Visual C++等外部编译器，并不能自动建立这样的对象表，也就不能自动识别控件对象的回调函数。因此，必须先在LabWindows/CVI中手工生成对象表，才能在外部编译器中使用LabWindows/CVI的控件对象。基本操作步骤是在设计好LabWindows/CVI面板后，从菜单中执行“Build | External Compiler Support...”，在“External Compiler Support”对话框中设置“UIR CallBacks”为“Object File”，并输入文件路径和文件名callback.obj，点击Create按钮即可创建callback.obj文件。然后将.uir文件拷贝到与Visual C++编译生成的可执行文件目录下(如Debug或Release目录)。这里需要特别注意的是：.uir所在目录与前一种方法是不同的。再将生成的.obj、.c和.h文件都添加到Visual C++工程中，同时还需要将cvi70/extlib/msvc目录下的文件cvirt.lib、cvisupp.lib添加到工程中。在Visual C++中调用LabWindows/CVI程序时，需要在#include部分添加对cvirte.h和userint.h的包含。在调用LabWindows/CVI生成的代码之前，还需要调用InitCVIRTE函数进行初始化，如：

```
    if (InitCVIRTE (AfxGetInstanceHandle(), 0, 0) == 0)
        return; /* 内存不够 */
```

本文所介绍的两种方法都可以将LabWindows/CVI与HLA系统结合，而且两种方法各有特点：基于LabWindows/CVI的DLL的方法比较独立于外部编译环境，对于一些不使用 C++接口的RTI(如使用Java语言接口的pRTI)也是适用的，比较通用，应用范围较广；而基于外部编译器的方法可以直接调用Visual C++所提供的各种功能，从而突破了LabWindows/CVI自身的ANSI C编译器的种种局限，通信模式简单，易于编程，但这种方法不易于借助LabWindows/CVI的环境对硬件进行调试。硬件设备开发阶段推荐使用前一种方案，而如果硬件设备已经定型或采用现成商用硬件，采用后一种方法更简便。

参考文献

1 周 彦，戴剑伟.HLA仿真程序设计[M].北京：电子工业出版社，2002

2 刘君华,白 鹏,汤晓君等.基于LabWindows/CVI的虚拟仪器设计[M].北京:电子工业出版社，2003

3 Pawletta, S., Drewelow, W., Pawletta, T. HLA-based simulation within an interactive engineering environment[J], Distributed Simulation and Real-Time Applications, 2000. (DS-RT 2000). Proceedings. Fourth IEEE International Workshop on, 2000;(8)

4 National Instruments Corp. LabWindows/LVI Programmer Reference Manual[M].2003

关于“LabWindows/CVI在半实物分布式仿真系统中的应用”，我有如下需求或意向：

用户名： 密码： 验证码： 5829 欢迎注册 提 交

相关应用

- 基于虚拟仪器的网络化自动测试系统的构架及实现
- 基于LabWindows/CVI的ATS软件框架
- A/D转换芯片的测试环境构成及测试方法
- 基于虚拟仪器的圆盘式电流变传动机构的动态分析
- LabVIEW与MATLAB混合编程
- 基于LabVIEW的USB实时数据采集处理系统的实现

地址：北京海淀区清华东路25号电子六所大厦
联系电话：82306084 / 82306085 传真：62311179 京ICP备05053646号

推荐分辨率1024*768 IE6.0版本

APPLICATION OF ELECTRONIC TECHNIQUE