УДК 621.317.73

В. Бондирєв, Є. Походило
Національний університет “Львівська політехніка”,

ПРИНЦИПИ ПОБУДОВИ ІНСТРУМЕНТАЛЬНИХ ДРАЙВЕРІВ У СЕРЕДОВИЩІ ПРОГРАМУВАННЯ LABWINDOWS CVI

Розглянуто принципи побудови інструментальних драйверів у середовищі програмування LabWindows CVI. Проаналізовано внутрішню структуру та зовнішній інтерфейс IVI–сумісного драйвера. Наведено практичні результати розроблення прикладної програми для керування вимірювачем CLR параметрів BR2827 фірми MCP Lab Electronics.

In the paper principles of developing IVI – compliant instrumentation driver in LabWindows CVI environment are considered. Internal design and external interface of IVI driver are discussed. Application program for automated control of the CLR measuring instrument BR2827 (MCP Lab Electronics) was developed. Practical measurement results are presented.

Вступ

Без перебільшення можна стверджувати, що історія розвитку вимірювальної техніки досягла своєї поворотної точки, коли була реалізована можливість контролю за вимірювальним приладом за допомогою комп’ютера. Програмне керування приладом без втручання людини надало процесу вимірюваня додаткової швидкості і точності. Поява математичних пакетів з потужними аналітичними можливостями і гнучким представленням отриманих даних додатково розширили можливості експериментатора. Саме ці міркування були головним стимулом при створенні таких середовищ розробки, як LabView та LabWindows від фірми National Instruments. Додатковою безсумнівною перевагою цих середовищ стало те, що в них із самого початку були закладені широкі комунікаційні можливості для керування вимірювальною апаратурою з використанням різноманітних інтерфейсів. З їх появою задача побудови прикладної програми значно полегшилася, проте розробник так і залишився сам на сам із непростою проблемою створення драйвера, оскільки це передбачає детальне вивчення послідовності програмування конкретного пристрою та вникнення в деталі програмної реалізації низькорівневих протоколів обміну даними.

Виникнення організації IVI

Інструментальним драйверам минулих років були притаманні такі недоліки: деякі з них були занадто сильно прив’язані до програмних продуктів виробника вимірювальної системи, деякі були занадто складними для модифікації та покращання. Окрім цього, часто драйвери від різних виробників не могли безконфліктно співіснувати в межах однієї прикладної програми. Очевидно, що назріла необхідність у створенні драйверів із відкритою архітектурою, побудованих за чітко визначеними стандартами, які можна було б розвивати та вдосконалювати. Дотримання стандартів мало б стати гарантією взаємної сумісності і відсутності конфліктів у програмному продукті та його здатності бути застосованим на нових операційних системах.

З метою розвитку вищенаведеної концепції як відкритого промислового стандарту у липні 1998 р. було створено Організацію взаємозамінних віртуальних інструментів (Interchangeable Virtual Instruments Foundation). Членами IVI Foundation стали такі лідери–виробники вимірювальної апаратури, як Advantest, Boeing, GenRad, Hewlett-Packard, Lockheed-Martin, Lucent Technologies, Marconi North America, National Instruments, Raytheon, Tektronix, Wavetek та деякі інші. Сьогодні

кількість офіційних членів організації IVI Foundation сягає 35. Головна мета, яку поставили перед собою члени асоціації – досягти повної взаємозамінності приладів різних виробників незалежно від інтерфейсу керування (COM, LPT, GPIB (IEEE 488), ISA, PCI, CompactPCI, VXI, PXI) [1].

Варто зазначити, що головна рушійна роль у цьому належить National Instruments, оскільки саме її розробки стали фундаментом, на який спирається концепція IVI. Ще задовго до створення Організації взаємозамінних віртуальних інструментів (IVI) піонерські ідеї віртуальних приладів від National Instruments знайшли своє втілення у програмних пакетах LabView та LabWindows [2], [3].

Структура IVI інструментального драйвера

Розглянемо структуру IVI – сумісного драйвера (рис. 1) на прикладі його реалізації у програмному пакеті LabWindows/CVI, що став найпопулярнішим середовищем програмування для спеціалістів, які програмують мовою С та розробляють програмні засоби для автоматизації вимірювань, тестувань та контролю виробництва.

Для кращого розуміння введемо такі поняття:

1) модель внутрішньої архітектури – відображає внутрішню будову тіла інструментального драйвера.

2) модель зовнішнього інтерфейсу – показує, як інструментальний драйвер взаємодіє із іншими компонентами програми.

Рис. 1. Модель внутрішньої архітектури IVI – сумісного інструментального драйвера

До моделі внутрішньої архітектури входить набір функцій, які можна умовно поділити на три категорії: функції верхнього рівня, функції нижнього рівня та функції – обробники подій. Функції нижнього рівня реалізують специфічні для певного вимірювального приладу операції (наприклад, зміна рівня тестового сигналу (функція №1), перемикання діапазону (функція №2), перемикання тригера запуску вимірювання (функція №3) тощо). Функції верхнього рівня є фактично

комбінацією функцій нижнього рівня і дають змогу користувачу простіше взаємодіяти із приладом (наприклад, здійснення операції вимірювання, яка передбачає виконання функцій №1 – №3 у певній послідовності). Функції, що обробляють події, викликаються ядром IVI у певні моменти часу. Ядро IVI запам'ятовує стан приладу, перевіряє виконання поданих команд шляхом контролю його стану. Серед інших функцій, які виконує ядро IVI, слід зазначити можливість емуляції приладу, що значно полегшує процес створення і відлагодження драйвера за відсутності самого вимірювального пристрою.

До моделі зовнішнього інтерфейсу належать: візуальний інтерактивний інтерфейс користувача, програмний інтерфейс розробника, внутрішній підпрограмний інтерфейс та інтерфейс вводу – виводу VISA. Останній відіграє найважливішу роль, оскільки саме VISA беспосередньо відповідає за здійснення обміну даними із вимірювальним приладом [4].

Інтерфейс вводу – виводу VISA

Важливим кроком Організації взаємозамінних віртуальних інструментів (IVI Foundation) на шляху до програмної сумісності незалежно від апаратної платформи стало розроблення стандарту програмної реалізації задач вводу – виводу, який отримав назву системної архітектури віртуальних інструментів (Virtual Instrument System Architecture), або VISA. Функції VISA залишаються одними і тими самими, незалежно від того, який інтерфейс використовується для взаємодії із вимірювальним приладом – GPIB, GPIB-VXI, VXI, PXI, TCP/IP чи послідовний контролер, на відміну від функцій API (Application Programming Interface), які залишаються прив'язаними до конкретного типу інтерфейсу. Реалізація стандарту VISA компанією National Instruments у своїх програмних пакетах отримало назву NI-VISA. У середовищах LabView та LabWindows NI-VISA поставляється із вбудованою утилітою, що має назву Інтерактивне Керування VISA (VISA Interactive Control, VISAIC). Ця утиліта через зрозумілий і наочний графічний інтерфейс дає змогу налаштовувати всі основні ресурси VISA. VISAIC є зручною відправною точкою для оволодіння навичками програмування у середовищах LabView та LabWindows із застосуванням інтерфейсу вводу – виводу VISA [5].

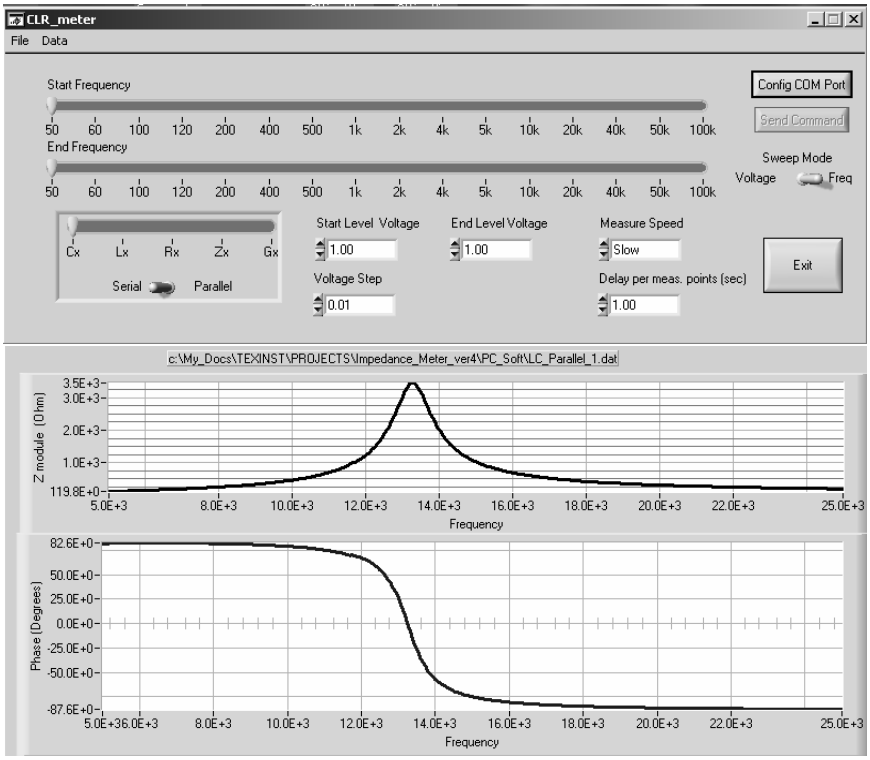

Рис. 2. Зовнішній вигляд графічного інтерфейсу прикладної програми керування вимірювачем CLR параметрів

З використанням концепції IVI – сумісного драйвера у програмному пакеті LabWindows було розроблено прикладну програму для керування вимірювачем CLR параметрів BR2827 фірми MCP Lab Electronics. Керування приладом здійснюється по послідовному порту на швидкості 9600 b/s. Обмін даними (передача команд та отримання результатів вимірювання) відбувається у символьному ASCII форматі. Інтерфейс програми разом із результатами вимірювання зображено на рис. 2. У цьому випадку вимірюваною величиною є імпеданс (модуль $|\bar{z}|$ та кут φ) паралельно сполученого LC кола залежно від частоти прикладеного тестового сигналу.

Висновки

1. Застосування програмних пакетів LabView та LabWindows фірми National Instruments для розроблення програмних засобів автоматизації експерименту, тестувань та здійснення вимірювань на виробництві є надзвичайно перспективним і зручним завдяки принципам віртуальних інструментів, закладеним в цих пакетах.

2. Важливою перевагою цих середовищ є широкі комунікаційні можливості для керування вимірювальною апаратурою з використанням уніфікованого інтерфейсу вводу – виводу VISA. Застосування VISA дає змогу абстрагуватися від типу конкретного застосованого інтерфейсу, що значно полегшує процес створення інструментального драйвера.

3. Розроблення інструментального драйвера згідно з вимогами Організації взаємозамінних віртуальних інструментів (Interchangeable Virtual Instruments Foundation) є запорукою його здатності бути перенесеним на майбутні операційні системи і відсутності конфліктів у вимірювальному програмному комплексі.

1. Анфиногенов А. ComputerWeekly, 45_98. – С. 24–30. 2. LabWindows/CVI User Manual, National Instruments, February 1998 Edition Part Number 320681D-01. 3. LabWindowsTM/CVITM Programmer Reference Manual National Instruments June 2003 Edition Part Number 323643A-01. 4. LabWindows™/CVI ™ Instrument Driver Developers Guide, National Instruments,April 2003 Edition, Part Number 370699A-01. 5. VISA: NI-VISA™ User Manual, National Instruments, September 2001 Edition, Part Number 370423A-01.

УДК 62-82:658.512.011.56

А. Петренко, В. Ладогубець, О. Безносик, О. Фіногенов, О. Чкалов
ННК “ІПСА” НТУУ “КПІ”

ВИКОРИСТАННЯ ПАКЕТІВ СХЕМОТЕХНІЧНОГО ПРОЕКТУВАННЯ ДЛЯ ПОБУДОВИ МОДЕЛЕЙ МЕХАНІЧНИХ КОМПОНЕНТІВ

Розроблено метод синтезу моделей гетерогенних інтегральних елементів МЕМС для схемотехнічного рівня та наведено результати побудови схемних рішень.

The method of synthesis of models of heterogeneous integrated MEMS’s elements is - process developed for a schematic and technical level is developed and results of construction of schematic solutions are resulted.

Вступ

При проектуванні сучасних пристроїв важливу роль відіграє можливість використання єдиного інструментарію для моделювання об'єктів, в яких відбуваються різні фізичні процеси: електричні, механічні, оптичні, теплові тощо. Це вимагає представлення різних підсистем