NI platform for automated test
Customers will not accept products whose quality is lower than existing products, thus product quality must improve over time. This is achieved by engineering products to a higher standard, more rigorous pre-production and production testing, increased test coverage and by improving the manufacturing process.

Similarly, as companies strive to deliver the latest and greatest products to their customers, the complexity and electronic content of almost all devices is increasing rapidly.

The third industry trend involves companies moving to globalized manufacturing which allows for strategically condensing the number of manufacturing locations. While this eliminates the challenge of having to manage many factories, it also means that many of the factories will be expected to produce higher volumes.

As we all know, the latest consumer electronic products and technologies today will sell for significantly less a year from now. The unit price of consumer electronic devices over time decreases, both by customer expectation and by improvements to the manufacturing process and increased volumes.

Finally product life cycles and time to market are decreasing as design to manufacturing cycles permit quicker product introductions.
Manufacturing Test Challenges

- Reduce Test System Costs
- Keep Up with Shorter Product Life Cycles
- Improve Quality

- Lower Development Time
- Increase Throughput
- Reduce Maintenance

Recent industry trends have led test engineers to encounter a number of test challenges when building automated test equipment (ATE) systems. One of the main goals in any manufacturing environment is to lower the overall test system costs. With the increase in product complexity, quality standards and production volume, companies are realizing higher costs to test. In order to stay competitive by offering products at lower prices to customers, companies must be able to keep test system costs to a minimum, while maintaining production volumes and quality levels.

In order to accomplish these goals, companies must be able to lower development time, increase throughput and reduce maintenance costs. One way that many companies are beginning to accomplish all three of these goals is by taking advantage of a modular test architecture that leverages management software such as TestStand and modular measurement hardware based on PXI to implement flexible parallel test systems. During this presentation we will focus more on how parallel testing can be successfully implemented using these tools.
An important aspect of a modular test system architecture is the ability for your test system architecture to evolve with your testing needs. A common example of this involves testing a product throughout its development lifecycle. During a product’s design phase, a design engineer may need only the core measurement functionality delivered by the test modules, measurement services, and measurement hardware. However, as the product transitions into the validation and production phase, the number of channels to measure in an automated test system increases substantially. Switching hardware and switch management software (such as NI Switch Executive) are an extremely valuable component of a modular test architecture for these areas of the product development lifecycle.
PXI is the fastest growing standard in the test industry since GPIB (IEEE-488). According to Frost and Sullivan, the PXI market grew by more than 38% in 2001 and is forecasted to continue rapid growth in coming years. By the end of 2002, there were 883 PXI modules available from all vendors. That represents 55% growth over 2001. Counting CPCI modules that are targeted at test and measurement applications, there are about 1,100 modules available.
The major strength of PXI is that it uses proven industry-standard technology. PXI is built on CompactPCI, an industrial version of the PCI bus found in almost all desktop computers. On top of CompactPCI, PXI adds timing, triggering, and synchronization similar to functionality delivered in VXI. For easy integration, PXI uses driver and networking standards from the PC software market, including plug-and-play drivers.

Because PXI and CompactPCI are completely interoperable, users can use any core CompactPCI product in a PXI system and vice-versa.
This table illustrates the key components of the major test platforms—GPIB, PXI, VXI, and PCI, and how they compare with each other. PXI combines the size, cost, and throughput advantages of the PC platform with the instrumentation-specific features of GPIB and PXI, giving the user the best of both worlds.
Just as PXI leverages off other standards, these other instrumentation architectures can leverage off PXI technology. With readily available interface modules, a PXI system can be used as the main controller for VXI, VME, and GPIB systems. PXI and CompactPCI modules and systems are completely interoperable. Because PXI also works together with these other standards, the result is the preservation of investments in other technologies.

Until recently, you could not use standard desktop PCs to directly control a PXI system without an embedded controller. With the introduction of the new MXI-3 technology, direct PC control of PXI/CompactPCI systems is now possible.
Choose whether you want to control your system from a standard PC with MXI-3 or with a fully integrated embedded computer.

If your application precludes the use of an external PC and you require a completely integrated, compact solution, then an embedded computer is the right choice. An embedded computer is required for controlling portable chassis with integrated displays. National Instruments offers a variety of embedded computers with your choice of processor speed, I/O configuration, operating system, and application development software. All embedded computers include a built-in hard drive, floppy drive, and video. Peripheral ports such as USB, serial, parallel, mouse, and keyboard are also standard. GPIB and Ethernet ports are also available as options on some embedded computers.

MXI-3 kits give you a fully transparent link from your PC to your PXI system. You plug a MXI-3 interface card into your PC and connect it with a copper or fiber optic cable to a MXI-3 module in slot 1 of your PXI system. MXI-3 extends the PCI bus at full speed so that the processor in your PC transparently configures and controls the PXI modules—whether the PXI system is 2 m or 200 m away. MXI-3 is a low cost control solution that allows you to benefit from the very latest in PC technology.
PXI Advisor is an online utility that takes you through the five steps of configuring a PXI system.

PXI Advisor does the following:

- Customizes and orders a PXI measurement and automation system
- Eliminates the need to visit 10 to 15 different online store pages to order a complete PXI system
- Eliminates errors in orders
- Provides a visual representation of your custom PXI system for presentations, proposals, and more

PXI Advisor leads you through a series of questions such as chassis type, processor type, processor speed, and memory. Then you can select and place PXI modules into your chassis graphically. You can use PXI Advisor to print a graphical image of your system, export an image for use in other documents, or generate an online order for the system. The part numbers representing your PXI system configuration are automatically added to a shopping cart. You can continue shopping for other products in the online store or simply check out. Every unique PXI configuration is assigned an ID so you can retrieve your configuration at a later time without starting over. In addition, because you can save your unique ID, a sales engineer can start a configuration for you and you can later view it when entering your assigned configuration ID. You also can use the ID to create a new configuration from a saved one by modifying the modules you choose.
As a concrete example of the value of virtual instrumentation, let’s look at a typical traditional instrumentation system, which might contain an oscilloscope, signal source, digital multimeter, and perhaps a switching system.

While this can be a powerful setup, it takes up valuable space in your lab, is extremely expensive, and it is not tightly integrated, creating more work for you.
Now, you may think that this looks like a traditional box instrument such as those shown on the previous slide.

However, the equipment on this slide actually incorporates all four of the traditional box instruments, as well as PC components such as the power supply, display, processor, hard disk, and memory, to create a compact, tightly integrated virtual instrumentation system.

This particular piece of equipment is from our rugged industrial PXI platform. The system contains a Pentium class computer running NI LabVIEW, a 2-channel oscilloscope, two arbitrary waveform generators, a 5½-digit DMM, and a 24-channel relay switch.

By reusing PC components, instrumentation systems developed with modular instrumentation occupy a much smaller footprint on your lab bench or in your test rack. Additionally, both the software and hardware are tightly integrated with the PC components, significantly increasing the value of your instrumentation system.
Example Unified PXI-Based JTAG station
As your product enters the final validation and production test stages of its development lifecycle, sophisticated test management software such as NI TestStand completes the modular test architecture by easily integrating with the core measurement functionality in addition to the important switching needs of high channel count applications.
Benefits of Test Management Software

• Enhanced development
  – Eliminate duplicate effort
  – Reuse test code
• Streamlined throughput
  – Intelligent, efficient and scalable testing
  – Optimized execution engine
• Reduced maintenance
  – Eliminate obsolescence
  – Lower support & training

Streamlined throughput is achieved by implementing intelligent decision making in the test framework that will ensure Units Under Test (UUTs) only undergo tests which are appropriate. This can save substantial time, and hence money, on production lines that are running at capacity. For example if a UUT fails a power-up test, it should skip all subsequent tests because the unit is not powered correctly.

Loading test parameters dynamically at run-time also streamlines throughput. This functionality enables diversity testing, where a number of slightly different UUTs, such as mobile phones with different specified bandwidths, can be tested with zero downtime.

Optimized execution is also key when running production at or near capacity, or even when there are many tests executing sequentially. You can minimize the overhead in calling test modules by implementing an optimized test management framework with TestStand, resulting in increased throughput. Parallel or multi-UUT testing also significantly improves throughput by allowing many UUTs to be tested simultaneously using the same test station. This maximizes instrument usage and reduces the number of testers required.
Manufacturing products faster is certainly one way to reduce the cost to test, but another is to reduce the amount of development time taken to create the test systems. By using an off-the-shelf product, the development is automatically cut because the core framework is already provided. This framework is the most difficult part to develop in any test system, as it contains all of the engine, the decision making components, and the reporting and database tools. Avoiding duplicated efforts saves years of development, and allows you to focus instead on the organization’s core priorities, such as which tests need to be specifically created for a new UUT.

Finally, a test management service reduces maintenance, which is one of the largest hidden costs in a test system. “Maintaining” systems covers many areas, but the biggest and most avoidable issues revolve around obsolescence. Looking at the past 10 years, there were at least five shifts in operating systems, some larger than others. These shifts often cost organizations hundreds of thousands of dollars because of the code rewrites, re-engineering, and lost manufacturing capability while the system is being upgraded. By transferring to an off-the-shelf system, such as TestStand, the majority of these obsolescence costs can be avoided since we work out all of these issues before the official release of the next operating system. Another area of reducing the cost of maintenance is training. With an off-the-shelf system, your engineers no longer have to spend their time creating and maintaining internal training materials. Instead, National Instruments provides standard or customized training courses and manuals, locally and globally.
Building custom test management software can be a daunting task. The operator interface, sequence development environment, database integration, and interface with various test development languages are just a few of the factors you must consider.
Fortunately, TestStand provides many off-the-shelf features and a development framework that fulfills a large percentage of typical test management software specifications. Also, because each organization has different requirements for its operator interface, reporting, and so on, TestStand is specifically designed for customization and has tools for providing the additional features needed to complete the specifications. The bottom line: **TestStand saves you time and money** by providing the core functionality you would otherwise have to design, code, maintain, and support yourself.
TestStand – Test Management Software

- Call tests written in any test language
- Parallel testing
- Automatic report generation
- Modular and customizable
- Flexible operator interfaces

National Instruments TestStand software, awarded the **2002 Test Product of the Year** by readers of *Test & Measurement World* magazine, is a customizable, off-the-shelf test management software. TestStand is a full-featured tool that provides flexibility to meet a diverse set of requirements. TestStand includes off-the-shelf features such as parallel sequence execution, operator interfaces, report generation, database logging, and much more. TestStand gains its power and flexibility through the ability to customize many of these features.
Before looking at the different features in the TestStand environment, it is important to understand the TestStand Architecture, which is comprised of four main components: the TestStand Engine, the Sequence Editor, the Operator Interface(s), and the Module Adapters.

At the core of the TestStand architecture is the TestStand engine. The engine is an ActiveX server that is implemented as a set of 32-bit Dynamic Link Libraries (DLLs), which export an ActiveX API. The TestStand engine handles test executive tasks such as sequencing, looping, limit checking, data allocation and user management. The engine is optimized for speed and designed for maximum flexibility and long-term compatibility.

The TestStand Sequence Editor and operator interface(s) act as ActiveX clients and utilize the engine API for accomplishing such tasks as creating, editing, executing and debugging sequences. You can call the engine API from any programming environment that supports access to ActiveX Automation servers.

For the engine to be able to communicate with external code modules, it must know the code module type, how to call it, and how to pass parameters to it. This functionality is provided by the module adapters that act as an interface between the engine and external test development environments. Currently there are adapters for LabVIEW, LabWindows/CVI, C-style Dynamic Linked Libraries (DLLs), ActiveX Automation servers, and HTBasic.
The TestStand Sequence Editor development environment is used to create and manage your test sequences and overall test system. The TestStand Sequence Editor simplifies the creation, editing, managing, execution, and debugging of sophisticated test systems. It also allows for instantly configuring and viewing test reports or data stored in a database while also providing a complete user management to control access to your tests. Lastly, the TestStand Sequence Editor provides the ability to customize practically every aspect of the TestStand architecture and also assists in deploying test systems once development is completed.
There are several key terms that TestStand refers to within the Sequence Editor for test development. You can think of these concepts in a hierarchical manner, in which sequence files are at the top of the hierarchy. Sequence files can contain one or more sequences. Each sequence contains three step groups: setup, main and cleanup. The step groups are used to organize the steps within the sequence for logical flow. The order of execution of these step groups is setup, main and cleanup. Each step group can contain any number of steps or none at all. Steps can either call native TestStand functions, or be configured to call external test modules, using the appropriate test language adapter.
TestStand provides a framework for calling test modules written in many different programming languages. The framework provides module adapters for the most common test development environments. The module adapters also provide interfaces to each development environment for creating and editing test code from within the sequence editor.
The Sequence Editor is the development environment you use to create test sequences. Using TestStand as an off-the-shelf test executive, you can immediately start building your test sequences by choosing from **over 30 built-in step types** provided in the Sequence Editor. The wide selection of built-in step types were created based on common testing needs and input from test engineers such as yourself. The built-in step types include types that call external test modules, such as Numeric Limit Tests and String Value Tests, to application specific types used to directly control Interchangeable Virtual Instruments (IVI) compatible instruments, synchronize multiple test threads, and access databases within your test sequence without writing any external code modules. A variety of built-in step types are also included for performing multipurpose actions in your test sequences, such as Goto and Message Popup steps. Using the built-in step types allows you to build your test sequences very quickly and also **provides a modular test structure that enables code reuse.**
TestStand for Boundary-Scan

- JTAG Step Types
- Parameter Support
  - Test type
    - (infra, inter, memory, cluster, and so on)
  - Program data source
  - Start/stop locations
TestStand Debugging Features

- Instantly debug test code
- Debug LabVIEW, LabWindows/CVI, and Measurement Studio for Visual Basic code directly from TestStand
- Monitor TestStand variables, properties, and expressions at run time with watch window

In any test system, it is inevitable that problems will arise in your test modules during development or after deployment. Debugging an individual test module within the environment it was created in is often not adequate. To fully reproduce a problem, all test steps must be run in the same manner in which they are run during an actual test. Thus, it is important to have the ability to debug test code from the test management environment. TestStand uses an ActiveX interface to communicate with test development environments, and thus handle direct debugging of test code from within TestStand. For instance, you can step into a LabVIEW VI from a TestStand sequence and TestStand automatically launches LabVIEW, allowing you to single step through the VI source code with highlighted execution illustrating the flow of data.

You also can monitor the values of variables, properties, and expressions in TestStand during run-time using the built-in watch window in the sequence editor.
Flexible Operator Interfaces

- Deployable view of test system
- Ready-to-run versions written in LabVIEW, LabWindows/CVI, Visual Basic and Delphi
- Source code provided for customization

One of the most important aspects of any test system is how it is presented to the operator in the validation lab or on the factory floor. The operator interface can vary widely among different testing needs and applications and between different users such as operators and technicians. Unlike its counterpart, the sequence editor, you purposely cannot edit or create sequences within the operator interface. However, since the audience for the operator interface is so wide, TestStand comes with a set of operator interfaces written in several common languages. The operator interfaces can be customized to meet your exact requirements. These operator interfaces are provided in source code for LabVIEW, LabWindows/CVI, Visual Basic, and Delphi so that you can customize them, if required, to fit your exact needs.

The architecture of TestStand provides long term support of your customized operator interface, and all of your customizations. As new versions of the TestStand engine provide increasing capability, you can easily plug these new versions into existing operator interfaces that you have customized.
## Additional TestStand Features

- Parallel testing
- Product family testing
- Enterprise connectivity
- Modular switching integration

Today’s sophisticated test systems require test management software that supports powerful, easy to implement features like parallel and distributed testing, the ability to dynamically change test sequences and test properties on-the-fly for slight differences in product variation, seamless connection to enterprise database systems to upload and download crucial test information, and direct communication with and control of test and switching hardware.
Most production lines fan out at functional test to multiple testers to keep up with the speed of the line. Often, this fanning out is to N identical, independent testers. TestStand’s multithreading technology allows you to **execute multiple tests in parallel** from a single PC to **better utilize hardware** and achieve **more efficient testing**. When multiple sequences are executed concurrently in TestStand, **each sequence is run in its own thread**. It is also possible to debug or abort one thread without affecting others that are executing.

More advanced multithreading techniques can also be implemented, including batch testing, synchronization, and data sharing. More information on parallel testing in TestStand can be found in Application Note 136: *Using Multithreading to Increase Test Throughput*, available online at www.ni.com/teststand.
Let’s examine a simple example illustrating diversity testing. When testing a mobile phone, it is common to have several variations of a particular model based on slightly different features such as different firmware based on the service provider buying the phone. A single TestStand sequence can be used to test the multiple types of phones using dynamic sequencing and/or property loading. When the unit is placed in the tester, the serial number is scanned. TestStand can use this serial number to do a serial number lookup in a database which returns the model type of the unit, i.e. Model A or Model B. Based on this information TestStand determines at run-time which initialization routine to execute for that specific model. It then loads the test properties from a file or database corresponding to that model type. These properties can store data that might be passed to the test code, or used within TestStand as specification limits to determine if a particular test passes or fails.
Increasingly, today’s automated tests integrate with databases to store requirements and results. TestStand’s **built-in database components**, such as the database step types and the test result database logger, provide seamless integration with any ADO or ODBC compliant databases. Information from the database can be **downloaded to** and **uploaded from** a TestStand sequence. TestStand also provides a useful **Database Viewer** tool for viewing and/or modifying data in your database from within the Sequence Editor environment.
Large measurement systems often require many measurement devices, and it can be complex to manage these resources efficiently. To aid you in this task, NI Switch Executive software is available to manage the switching configurations in your large Automated Test Equipment (ATE) systems.
TestStand and PXI, together with the JTAG boundary-scan tools, deliver a truly modular test architecture that provides for creating a unified test system for structural and functional test. Thus, improving performance and reducing the overall cost to test by eliminating the need for additional test systems.

Summary

National Instruments TestStand is designed to automate a wide variety of test systems. Right out of the box, TestStand is a ready-to-run test management software that organizes, controls, and executes your automated prototype, validation, or production test systems. TestStand is completely customizable, so you can modify and enhance it to match your specific needs. Furthermore, TestStand provides the performance you need to keep up with increasing demands for faster test times, quicker test development, and more intelligent data sharing and test sequencing. TestStand uses a speed-optimized, multithreaded sequence engine capable of running parallel sequences. TestStand also features a flexible framework for sharing variables, allowing you to test more products faster.
Contact information

National Instruments Finland
+358 (0)9 725 725 11
ni.finland@ni.com