



BY DAN STRASSBERG

## Choosing a Real-Time Platform: FPGAs vs. Real-Time Operating Systems

FPGAs (FIELD-PROGRAMMABLE GATE ARRAYS) are ICs that contain large numbers of unconnected gates whose function you determine by downloading to the FPGA a wiring list that determines how the gates interconnect. The FPGA dynamically turns semiconductor switches on and off in accordance with the wiring list, making the connections modifiable in the field.

Because of their moderate cost, software-controlled reconfigurability, and high operating speed, FPGAs are the fastest-growing way to implement digital hardware; they are everywhere—in cell phones, cars, home-entertainment systems, and PCs.

Even so, classical methods of designing and debugging FPGAs don't make for quick product development. Unlike microprocessors, whose fixed hardware structure performs functions described in programs, FPGAs' functions depend on the hardware configuration. Writing a program for an FPGA is like designing hardware that executes specific functions.

Until now, relatively few engineers have been able to use FPGA technology due to the extensive knowledge of VHDL-devel-

opment software or other complex design tools required for programming. However, the rapid rise in the use of FPGAs has created a need for FPGA-design tools that help designers to quickly become productive. NI's LabVIEW FPGA lets you use

**LabVIEW FPGA and LabVIEW Real-Time bring the power of graphical programming to the design and prototyping of hard-real-time control systems.**

LabVIEW's familiar graphical-programming environment to develop code for FPGAs and target that code to an FPGA on NI's reconfigurable PXI-7831R board (Figure 1).

Using LabVIEW to program FPGAs significantly reduces the expertise and time required for application development: Using the LabVIEW FPGA module, you simply create a LabVIEW block diagram and download it to the FPGA. Because this block diagram executes in hardware, you have direct, immediate control over all of the I/O signals on the reconfigurable I/O hardware—enabling high-performance, user-configurable timing and synchronization and onboard decision making. This special technology provides user-defined hardware for many applications, such as custom discrete and analog control, simulation, digital-protocol emulation, and other applications that require precise timing and control.

### What type of platform should you choose?

Prototyping and deploying advanced test or control systems requires a variety of hardware and software, and you must decide which platform best suits your real-time application needs: an embedded PXI system, LabVIEW FPGA, or traditional FPGA-development tools.

An embedded real-time PXI system consists of a PXI chassis with a controller running NI LabVIEW Real-Time (RT), and a variety of I/O boards, such as data-acquisition, digital-I/O, or even motion-control and machine-vision boards. Using such hardware, you develop your application on a host computer and target it to the LabVIEW RT PXI controller, which easily interacts with various I/O pieces, such as reflective-memory and LVDT modules.

A system that uses LabVIEW FPGA requires a host computer, a LabVIEW FPGA plug-in board, and the LabVIEW FPGA development software. You develop your LabVIEW FPGA code on the host computer and then target it to the FPGA board.

Table 1: Where different platforms shine

	PC with Standard Desktop Operating System	LabVIEW Real-Time Embedded PXI System	LabVIEW FPGA	Traditional FPGA	Traditional DSP
<b>Ease of Development</b>	●	●	●	○	◐
<b>Overall Performance</b>	○	○	◐	●	●
<b>Control Loop Rate</b>	○	○	◐	●	●
<b>Computational Speed</b>	◐	◐	●	●	●
<b>Parallism</b>	○	○	●	●	◐
<b>Code Size Capability</b>	●	●	○	○	◐
<b>Floating Point Calculations</b>	●	●	○	○	◐
<b>Reliability</b>	○	◐	●	●	●
<b>Availability of I/O</b>	●	●	○	○	○
<b>Determinism</b>	○	◐	●	●	◐
<b>Size and Form Factor</b>	○	○	◐	●	●
<b>Power Consumption</b>	○	○	◐	◐	●
<b>Cost</b>	○	○	◐	◐	●

**FOR MORE  
INFORMATION**

- To learn more about real-time platforms go to [www.ni.com/realtime](http://www.ni.com/realtime)
- Real-Time online tutorials: <http://digital.ni.com/express.nsf/bycode/exqdig?opendocument&lang=&node=>

For traditional FPGA development, you usually use a prototype board with VHDL-development software, with which you target your VHDL code to the FPGA-prototype board.

As you can see from Table 1, some tasks are well suited to LabVIEW FPGA and some are better suited to an embedded real-time PXI system. Sometimes, you're better off using a traditional FPGA. LabVIEW RT with a PXI controller is an excellent choice for applications that handle many I/O points of different types, floating-point calculations, and extensive code. LabVIEW FPGA is best in situations in which control-loop rate, speed, parallelism, and reliability are most important. Some applications, for example complex real-time control systems that receive large numbers of sensor inputs, benefit from combining LabVIEW FPGA (for the sensors) and LabVIEW RT (for the high-level control problem).

Sometimes you should consider using a traditional FPGA. The main difference between traditional FPGAs and LabVIEW FPGA is that the PXI-7831R plug-in module is primarily for prototyping special applications, extending measurement and control systems' timing and synchronization capabilities, and where time-to-market is critical. You should choose a traditional FPGA when you will mass-produce a product that you have prototyped with LabVIEW FPGA or when you are incorporating the FPGA into other hardware.




**Figure 1: The NI-PXI 7831R allows you to implement the FPGA design you create with LabVIEW FPGA and then test, debug, and refine the design. If you are building large quantities of a system, cost considerations may suggest porting the debugged design to a traditional FPGA IC**

### Application examples

A good example of where you would use each of these technologies is in HIL (hardware-in-the-loop) testing. In HIL, an electronic system simulates a physical system, such as an engine, brake, or fuel injection unit, and is used to verify that the control system works properly. Having physical hardware in the control loop helps you verify that the system responds properly to physical I/O. In this situation, you use LabVIEW

FPGA in combination with the NI PXI-7831R to create simulated physical-system signals that are electrically the same as the real signals. You would also use LabVIEW RT for running the overall control loop and for doing the supervisory control of the whole system.

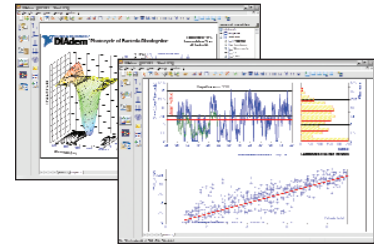
An example of when you might want to use LabVIEW FPGA by itself is in the design of a special motion-control algorithm. You could use LabVIEW FPGA to prototype and test the algorithm. If you then need to mass-produce the system, you would base the VHDL code on what you learned from the prototype. 

*Contributing Technical Editor Dan Strassberg has written about test and measurement for EDN for more than 16 years. He holds an MSEE from MIT (Cambridge, MA). Reach him at [StrassbergEDN@att.net](mailto:StrassbergEDN@att.net).*



## Accelerate Your Data Analysis

National Instruments DIAdem™ gives you a unified environment for data analysis and report generation. This software offers analysis and report generation capabilities that directly increase your productivity.



### Maximize your data investment and reduce your data analysis time by:

- Managing data from multiple databases and file formats
- Inspecting data in a dynamic environment
- Analyzing data using engineering math libraries
- Reporting results with customized, reusable reports

Read the white paper on how to reduce your data analysis time by as much as 90 percent. Visit [ni.com/info](http://ni.com/info) and enter **ebr42s**.

**(800) 991-9013**