Q&A: LabVIEW 8.5 Adds Another High-Level “Model of Computation” for Design

John Pasquarette, NI director of software marketing, discusses the use of abstraction in high-level design tools and how National Instruments LabVIEW graphical system design software incorporates these tools to accelerate the design process.

Q: What is a “model of computation”?

Pasquarette: The term “model of computation” has been used in academia to create abstract definitions of a computer system. In short, a model of computation is a particular way of describing how a piece of software behaves. We use the term to describe different high-level ways of programming desktop and embedded systems. Examples of models of computation include text-based, object-oriented, state diagram and graphical dataflow. Each model tends to have relative strengths and weaknesses that apply to particular domains and applications.

The ability to use different models of computation is a key pillar of our graphical system design vision. NI LabVIEW users have several models of computation from which to choose when programming their applications. LabVIEW includes graphical dataflow programming, dynamic system simulation, text-based programming, text-based math and object-oriented programming. With LabVIEW 8.5, we have added another model of computation – LabVIEW statechart programming, which is based on the UML specification of statechart diagrams – to make it easy for users to design complex systems using states, transitions and events. LabVIEW users also can mix and match models of computation to best describe their systems. For example, programmers can design a laser control system using a statechart to define the states, graphical data flow to implement the control logic on an FPGA and a simulation diagram to simulate the dynamics of the laser.

Q: Why did NI choose statecharts as the next “model of computation”?

Pasquarette: For years, designers have used classic state diagrams to quickly sketch the behaviors of their systems. Statecharts expand classic state diagrams to add the notions of concurrency and hierarchy, so designers can describe systems that contain parallel tasks. Additionally, statecharts add a formal way to respond to events, making them ideal for describing reactive systems. This is especially useful for designing embedded devices, control systems and complex user interfaces. Moreover, statecharts provide a natural means of documenting the functionality of a system. When combined with LabVIEW graphical data flow to define the behavior of each state, statechart diagrams can function as executable specifications.

Q: Who benefits from high-level design tools?

Pasquarette: The main beneficiaries of high-level design tools are those we like to call “domain experts.” These are the engineers and scientists who are not expert embedded developers; rather, they are innovators in fields such as biomedical instruments, mechatronics and high-energy physics. These are the people with ideas who are trying to make it to market with their revolutionary products. When they combine a high-level design tool such as LabVIEW with off-the-shelf modular hardware, they can iterate quickly through their designs to validate their algorithms with real-world inputs. The high-level development tools empower the domain experts to take their designs to embedded hardware without having to become embedded experts.

Q: Do users not sacrifice low-level control when using high-level design tools?

Pasquarette: The conventional wisdom is that there is no free lunch, and the same holds true for design software. High-level design tools offer less optimization capabilities when compared to low-level tools. However, the trade-off is increasingly worth the cost because designs are becoming more complex and the time to market expectations are getting shorter. Designers can no longer afford to wait for the embedded guru to develop the assembly code.

There are several key elements required for a high-level software tool to be used throughout the design process. It must provide hooks into the low-level features and functions that may be required to complete a design. Additionally, high-level tools must provide ways to reuse and integrate legacy code. This is why we have always provided low-level programming structures and functions within LabVIEW as well as methods to call into existing code such as C code, VHDL in the case of LabVIEW FPGA and textual math. Finally, the code developed with high-level tools must be reusable on hardware platforms that can be shipped in high volumes. For example, a machine designer can prototype his or her control and monitoring application using LabVIEW and any of the LabVIEW models of computation, including statechart and simulation, on a desktop system, and then use the same code on an embedded control system such as National Instruments CompactRIO.

Q: Many engineers view LabVIEW as a test tool. What role does LabVIEW play in the design arena?

Pasquarette: LabVIEW has been used as a design tool since it was invented in the 1980s. The original graphical dataflow semantics of LabVIEW turned out to be a great way to design a wide range of systems from telescope control to analytical instruments. The vast LabVIEW IP for advanced math, analysis and signal processing includes specific tools such as the LabVIEW Digital Filter Design Toolkit, which drastically reduces the development time of these systems.

In the last 10 years, the design capabilities of LabVIEW have grown to bring the power of LabVIEW to embedded real-time hardware. We have introduced technologies such as the LabVIEW FPGA Module so engineers can design hardware logic with graphical programming. Because graphical data flow can intuitively represent parallel software behavior, it is an ideal model of computation for an inherently parallel processing element – the field-programmable gate array (FPGA). The latest version of LabVIEW brings this similar intuitive experience to the programming of real-time multicore systems. Many of our users have experienced up to 10 times faster development using LabVIEW for embedded design. Additionally, with the LabVIEW Microprocessor SDK (Software Development Kit), we have extended the reach of LabVIEW to any 32-bit microprocessor. The growing support for embedded hardware and the multiple models of computation that LabVIEW supports make it a viable design tool with significant productivity advantages.

NATIONAL INSTRUMENTS™

11500 N Mopac Expwy • Austin, TX 78759-3504 USA
Tel: (800) 433-3488 • Fax: (800) 683-9300
info@ni.com • ni.com