A low-cost, on-site, reconfigurable client DAQ system

By Srirama Chandra

On a large factory floor, several types of real-world signals that represent various parameters, for example, temperature, pressure, and voltage, from different locations may be monitored, logged, processed, and controlled. In such cases, a distributed data acquisition systems network is used. In a distributed data acquisition system, a centralized server communicates with client Data Acquisition (DAQ) systems and logs all the acquired data at a centralized location for further analysis. The central system communicates with the client DAQ systems through a communication network (Figure 1).

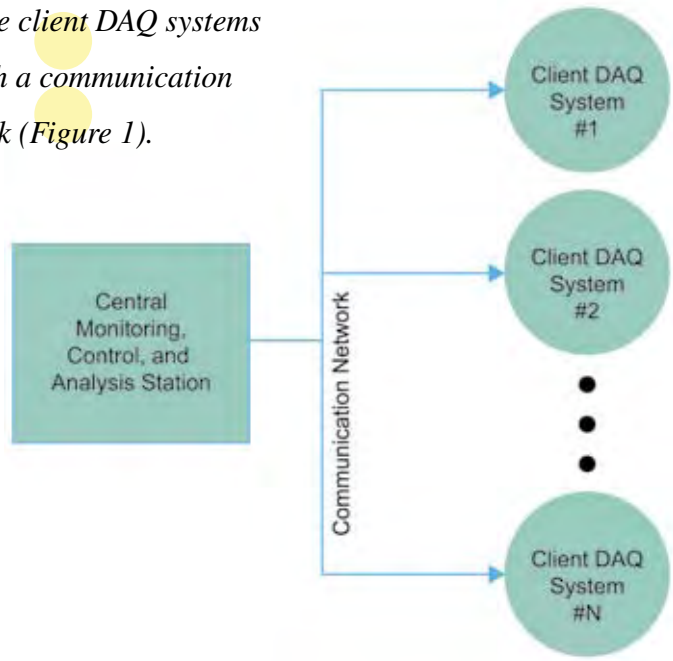

Figure 1

Types of communication networks include Ethernet, wireless link, and fiber optic link. The type of communication network depends on the rate of data collection, the environment, allowable measurement, and control latency.

Not only must the client DAQ system accommodate different communication interfaces, it also must be able to acquire and measure different sets of real-world signals. Srirama examines some of the issues involved in designing such client DAQ systems and then proposes a standard architecture for a low-cost client DAQ system that can easily be customized on-site.

Introduction

Interfacing with different types of networks and measuring different types of real-world signals must be considered when designing client DAQ systems, which, with all of the options, is no easy task. Plus, the ability to make modifications on-the-fly as the needs change would be beneficial.

Interfacing with different types of networks

Most distributed systems are installed with several issues in mind, including expected traffic, expandability, type of environment, security, robustness, fail-safe operation, interoperability, and interference immunity. Some of the types of networks are: Ethernet, fieldbus (for example, RS-485), and wireless (IEEE 802.11b, 802.15.1, 802.15.4, and so on). Each type of network has associated advantages, in fact, so much so that no single network type can satisfy all of the monitoring requirements. Additionally, networks need support for

both revolutionary and evolutionary infrastructures. The approach taken is a hybrid-networking environment in which there are several types of networks on a factory floor.

The client DAQ system must be capable of interfacing with multiple types of networks. If a process moves from one network domain to another, the client DAQ system must be able to move with it.

Measuring different types of real-world signals

Client DAQ systems should be able to interface with different types of real-world signals that are represented by analog or digital signals. The analog signals are generated by sensors or transducers that convert temperature, pressure, sound, or light into voltage. The electronic sampling of analog signals is called Analog to Digital Conversion (ADC). The digital output from the ADC should then be further processed or stored. Some signals can be generated from high-speed real-world signals (for example, vibration measurement) and some from much lower speed signals (for example, power supply voltage measurement).

Some applications will require client DAQ systems to generate analog voltages to drive chart recorders, audio amplifiers, process actuators, and other devices requiring an analog driving voltage. This is achieved with the use of Digital to Analog Converters (DACs). Many data acquisition boards have both ADCs and DACs.

Client DAQs should also provide Digital I/O (DIO) lines to operate relays, measure the speed of a fan-through tachometer, and turn a system on or off. Typically, digital signal monitoring requires timers, counters, and frequency measurements. The digital control signals should be able to perform frequency generation and pulse-width modulation, among other functions.

A client DAQ system should be able to interface with any combination of real-world signals. Further, the system also should be able to perform some of the signal processing function before transmitting to the central station for further analysis. The DIO systems also require timers and counters.

An on-site, customizable client DAQ system

The client DAQ system should not only accommodate any of the standard interfaces on the network side, but also any combination of data acquisition and control interfaces to real-world signals. This requires that the client DAQ hardware provide all types of network interfaces as well as hardware support for processing all types of acquisition interfaces. Not surprisingly, such a system will be very expensive because it needs all types of network interfaces and data acquisition interfaces. However, only a small subset of its functionality will be used at any installation location.

To reduce the cost of hardware, the proposed client DAQ architecture uses a standard base system that can be customized depending on the installation location. Simply plugging in the network interface module can customize the network side of the client DAQ system. Plugging in the required set of data acquisition interface modules customizes the data acquisition section of this module. Implementing all of the required interface and processing logic within the base module reduces the cost of these modules. To optimize the cost of the base module, it is customized with only the required processing and interfacing logic necessary for the installed configuration.

The base architecture, once customized with the necessary plug-in modules both on the networking and data acquisition side, communicates with the main central server and downloads the required logic for communication module interface,

data acquisition module interface, and the corresponding processing algorithms.

One possible on-site, customizable client DAQ system architecture uses nonvolatile FPGAs with a soft processor core. To better understand this architecture, a brief description of the technologies used follows.

In-system upgradeable nonvolatile FPGA

Most FPGAs require an external nonvolatile memory device to store the device configuration. After power-on, the FPGA configures itself by downloading the configuration from the external memory device. Often the external memory device is serial, and it may take an FPGA hundreds of milliseconds to download the configuration.

SRAM with on-chip embedded flash technology provides an advantage over the traditional FPGA in that it can be reconfigured in microseconds from its on-chip flash memory, as opposed to a traditional FPGA taking hundreds of milliseconds for reconfiguration. In addition, once the FPGA is operational using the configuration stored in SRAM memory, the flash configuration can be updated with a new configuration. This new configuration can be uploaded into the SRAM with minimum disruption to system operation.

> A client DAQ system should be able to interface with any combination of real-world signals.

Microcontroller soft core

A microcontroller in the FPGA at the client DAQ provides the ability to process and analyze data near the collection point before passing on to the network. A microcontroller dedicated to each DAQ function allows the use of a simple and low-cost 8-bit solution. An open IP core license, which applies many of the concepts of the successful open source movement to programmable logic applications, makes it even easier to develop appropriate algorithms.

Client DAQ description

To facilitate customization on both the networking and the data acquisition sides, the proposed architecture for the client DAQ system has two modes of operation, the preconfiguration mode and postconfiguration mode. The LatticeXP nonvolatile FPGA and the LatticeMICO8 (see Figure 2) soft processor core are used as an example.

In the preconfiguration mode, the base client DAQ system will have only the logic required to detect the networking and data acquisition modules.

In the postconfiguration mode the client DAQ system is equipped with all of the required data acquisition interface modules and the network interface modules plugged in. Additionally, in this mode, the base module logic has all the necessary interfacing and processing logic.

The transition from the preconfiguration and postconfiguration modes, also referred to as commissioning, is achieved as follows:

1. Plug in all necessary networking and data acquisition modules to the preconfigured base module.
2. The system is connected to the network and is powered on.
3. The onboard DSP processor communicates with the main central server through the plugged-in network module and sends the configured DAQ module information.
4. The main central server then sends the entire configuration for the nonvolatile FPGA required for operating with the plugged modules and the configuration-specific DSP processing algorithm.
5. The client DAQ system then updates the FPGA code and the flash memory with the newly downloaded codes.
6. The client DAQ system is now ready to perform the data acquisition operation with the site-specific networking as well as the data acquisition interfaces and is in postconfiguration mode.

Before changing either the networking interface or the data acquisition interface of the postconfigured client DAQ system, it should be decommissioned to preconfiguration mode. The decommissioning process is initiated from the server. During the decommissioning process, both the preconfiguration FPGA code and the preconfiguration DSP processing algorithm are downloaded through the networking interface. The client DAQ system then reprograms the FPGA as well as the flash memory with the newly downloaded code to get back into the preconfiguration mode. Once in the preconfigured mode, both the networking interface as well as the data acquisition interface can be changed.

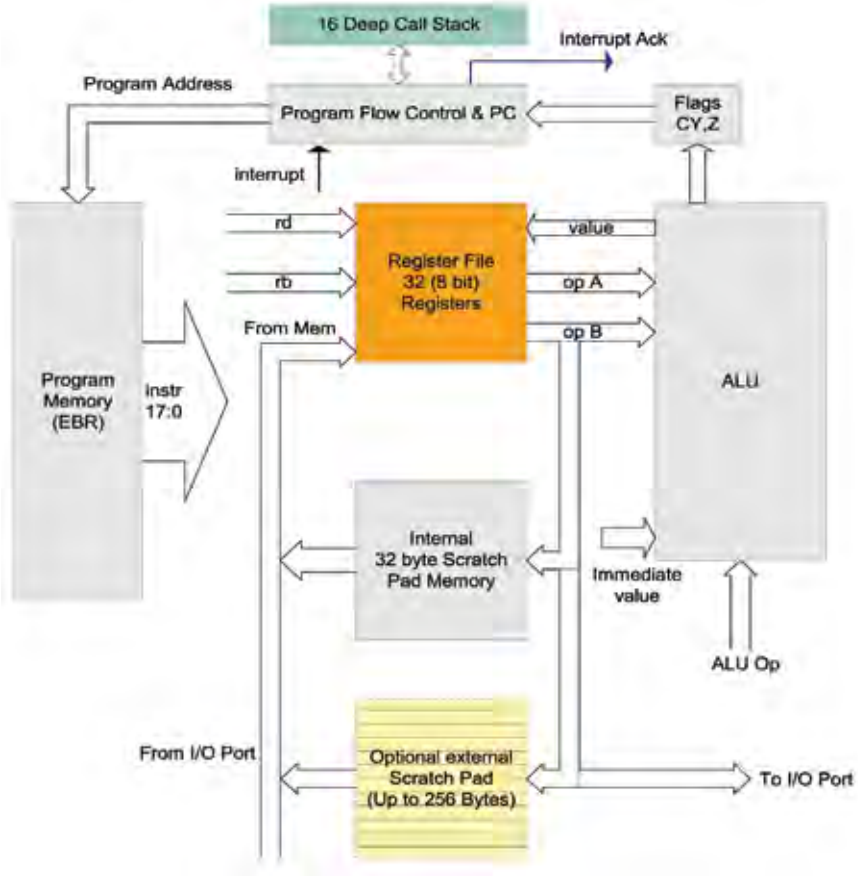

Figure 2

Client DAQ system in preconfigured mode

Figure 3 shows the block diagram of the proposed client DAQ system architecture in the preconfigured mode.

The system is divided into three sections: On-site customizable communication interface, common base module, and on-site customizable data acquisition and control interface.

The common base module section is equipped with a socket for the network module, as well as multiple sockets for data acquisition interface. The start-up program is stored in the flash memory. The Lattice Power Manager IC performs the sequencing, reset generation, and supervisory functions. The Lattice ispClock chip provides all the clocks required by the board. The LatticeXP FPGA in the preconfigured mode is programmed with the following subfunctions:

- Interfaces to all types of networking modules
- Processor bus interface
- Flash and DDR memory interface
- DAQ module detection hardware

The DSP processor code in the flash memory enables plug-in module detection and communication with the central server system over any of the communication interfaces.

Configuring the client DAQ system in preconfiguration mode

The first step in installing a client DAQ system is to plug in the necessary networking module as well as the site-specific data acquisition modules. After all the modules are plugged in, the system is turned on and connected to the network. The DSP processor first starts to execute from the flash memory. The contents of the flash memory are then transferred into the DDR memory through the processor interface section of the FPGA, and from then on the processor executes from the DDR memory, freeing the flash memory for updating. The processor then detects the plugged-in communication module as well as the DAQ modules, and sends a message to the server or the main controller through the network interface about the configured status of the client DAQ module.

The main controller then downloads both the site-specific processor algorithm as well as the site-specific FPGA configuration through the communication network. The downloaded DSP algorithm is then programmed directly into the flash memory. Because the LatticeXP FPGA operates from its SRAM configuration memory, its nonvolatile configuration memory is free for updating while the FPGA is functioning. The DSP processor then reprograms the nonvolatile on-chip FPGA configuration memory.

With the new code in both the flash memory as well as in the FPGA nonvolatile configuration store, the client DAQ system is now ready to perform the actual site-specific data acquisition task.

Postconfiguration mode

Figure 4 shows the postconfigured client DAQ system, which has the following modules plugged in:

- 10/100 Ethernet interface communication module
- Slow ADC and DAC module
- Digital relays/FET control module
- Fast ADC interface module

As noted previously, the LatticeXP FPGA logic in the postconfiguration mode depends on the network interface as well as the installed data acquisition modules. There is a one-to-one relationship between the networking module interface and the associated logic within the FPGA. But, the logic specific to a data acquisition module is realized by the use of the standard LatticeMICO8 functional block, and the associated processing function is implemented using the execution code loaded into the on-chip embedded RAM. For every data acquisition module, a copy of LatticeMICO8 core is instantiated within the LatticeXP FPGA fabric. The implementation logic between individual types of DAQ blocks differs only in the executable code loaded into the on-chip Embedded RAM.

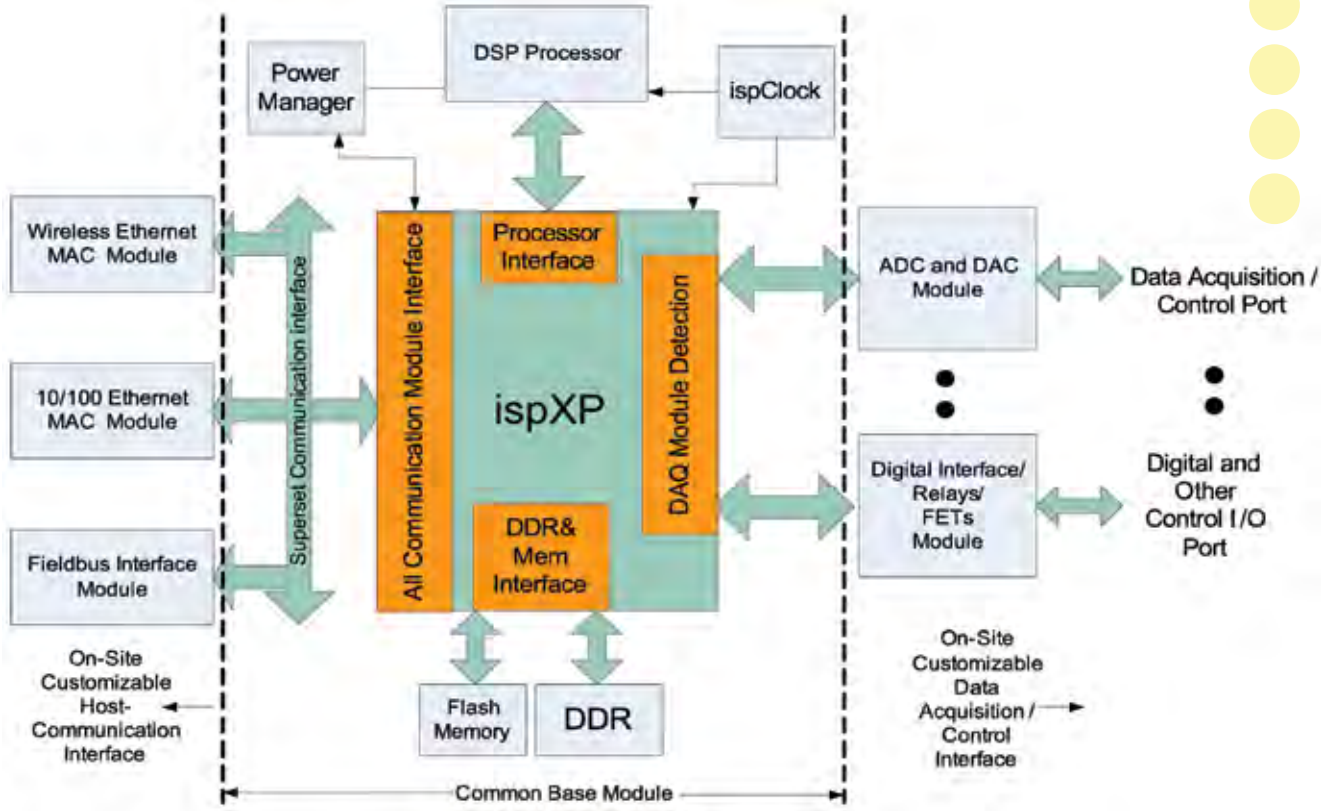

Figure 3

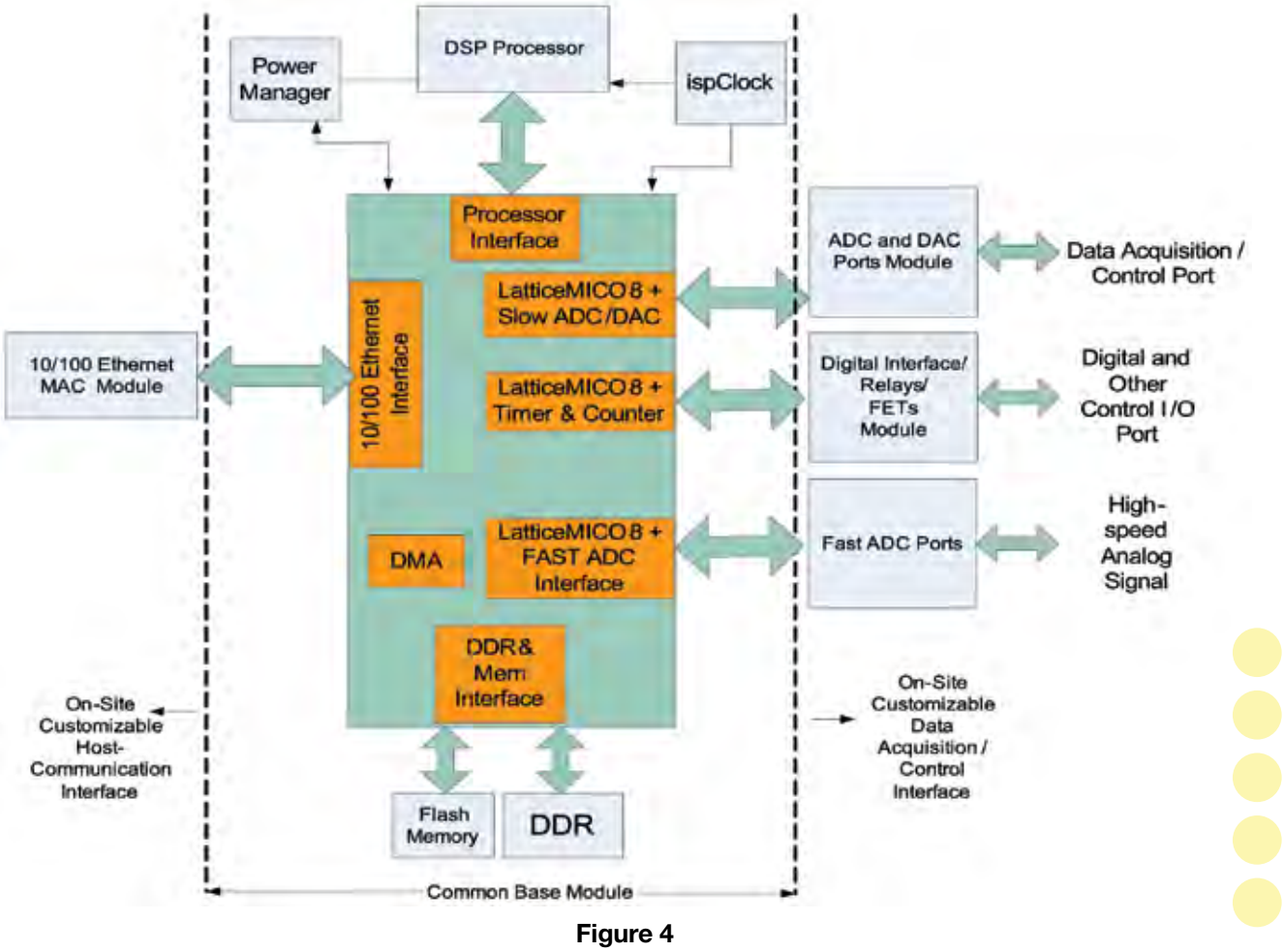
Power Manager
DSP Processor
ispClock
Processor Interface
10/100 Ethernet Interface
LatticeMICO8 + Slow ADC/DAC
LatticeMICO8 + Timer & Counter
DMA
LatticeMICO8 + FAST ADC Interface
DDR& Mem Interface
10/100 Ethernet MAC Module
ADC and DAC Ports Module
Data Acquisition / Control Port
Digital Interface/ Relays/ FETs Module
Digital and Other Control I/O Port
Fast ADC Ports
High-speed Analog Signal
On-Site Customizable Host-Communication Interface
On-Site Customizable Data Acquisition / Control Interface
Flash Memory
DDR
Common Base Module

Figure 4

SIDEBAR

Functional blocks of the LatticeXP FPGA

- **The 10/100 Ethernet interface:** This functional block performs all of the Media Access Control (MAC) layer functions and enables the client DAQ system to communicate with the central server system.
- **DDR memory interface:** This drives the DDR memory and provides simple memory interface structure to the rest of the blocks.
- **Processor interface:** This handles all of the DSP processor bus interface state machine logic and maps execution and data memory into DDR.
- **DMA controller:** This is a multichannel DMA controller with a channel dedicated to transferring data to and from the DDR memory and the networking interface, each of the data acquisition module interfaces, and the DSP processor interface.
- **LatticeMICO8 instantiation per DAQ module:** For every DAQ module plugged in, a copy of LatticeMICO8 and its executable code are instantiated within the FPGA. Each of the instantiations differs only in the processing algorithm loaded into the embedded memory of the LatticeXP FPGA. The maximum number of DAQ modules supported determines the size of the FPGA selected. The data to and from each of the LatticeMICO8 processors and the memory is handled through the DMA controller that is also instantiated on the FPGA. A description of the algorithm loaded into the FPGA's embedded memory block corresponding to each of the LatticeMICO8 follows.
- **LatticeMICO8 + slow ADC/DAC interface:** This is one of the instantiations of the LatticeMICO8 soft processor with its algorithm to periodically acquire the code from the ADC, and to perform preprocessing operations on each of the acquired samples such as offset shifting. The LatticeMICO8 processor also compares the input voltage level with a preset level and interrupts the processor if the input level exceeds the threshold level. Otherwise, it simply logs the voltage value in a prefixed location in the memory using the DMA block. If there is a command from the DSP processor, it picks the data from the external memory using the DMA and sends the data to the DAC.
- **LatticeMICO8 + timers and counters:** The LatticeMICO8 performs the counter and timer function as determined by the logic interface function of the digital and other I/O module DAQ.
- **LatticeMICO8 + fast ADC interface:** This module is coded such that the data from the ADC is sent directly from the DAQ interface to the DDR memory using the DMA controller. Once the data transfer is complete, the DSP processor is interrupted with information about the status of the transfer and the memory block location to which the ADC data is transferred. The DSP processor performs all the signal processing functions directly in the memory, builds the packet, and transmits it to the main processor.

The flash memory consists of the algorithm specific to the configuration.

The LatticeXP FPGA is now configured with the following function blocks:

- 10/100 Ethernet interface
- DDR memory interface
- Processor interface
- DMA controller
- One instantiation of LatticeMICO8 per DAQ module

Decommissioning

If the equipment needs to be brought back to preconfiguration status, it can be initiated directly from the central station. For this, both the flash memory contents as well as the LatticeXP FPGA configuration are changed to preconfiguration code using the onboard DSP processor.

Once the programming is complete, the client DAQ system can be removed from the network and used anywhere.

Summary

Because the on-site specific configuration and not a superset of all possible configurations determines the size of the LatticeXP FPGA, the size of the FPGA is small. The LatticeMICO8 provides an easy method to implement the module-specific algorithm using software. This enables the module-specific algorithm implementation without having to refit the logic within the FPGA. Because the LatticeMICO8 occupies very few LUTs (200), multiple instantiation of the LatticeMICO8 does not require a large FPGA. Additionally, the LatticeXP FPGA family offers one of the most economical solutions at any given FPGA size. All these factors contribute to reducing the cost of the implementation.

This architecture enables a single base module to interface not only with different DAQ modules, but also with different networks through on-site customization of the entire system. Because the same hardware board is used across all locations, process managers can standardize their entire distributed data acquisition system using these modules.

The throughput of the system is increased because the LatticeMICO8 processors offload the main DSP processor from all the slow peripheral operations as well as manage the data transfer buffer memory management. The processor just has to perform the communication and fast ADC data processing functions. Consequently, a slower, less expensive DSP processor will be satisfactory for a given function.

In conclusion, a low-cost nonvolatile FPGA such as the LatticeXP used in concert with an open source code soft processor such as the LatticeMICO8 enable the implementation of a lowest-cost solution client DAQ system that can be used as a standard solution in a distributed data acquisition system. ECD

Srirama Chandra *is the marketing manager for the in-system programmable mixed-signal products at Lattice Semiconductor Corp. Prior to joining Lattice, Srirama worked for Vantis and AMD in sales and applications and previously was a telecom design engineer with Indian Telephone Industries. Chandra received his MS degree in Electrical Engineering from Indian Institute of Technology, Madras.*

To learn more, contact Srirama at:

Lattice Semiconductor Corp.
5555 N.E. Moore Court • Hillsboro, OR 97124
Tel: 503-268-8634 • Fax: 503-268-8347
E-mail: shyam.chandra@latticesemi.com
Website: www.latticesemi.com