'Programmable logic controllers (PLCs) traditionally used for plant control do not offer the high-speed I/O required for diagnosis and monitoring tasks'

acquisition as it often involves multiple I/O tasks such as analogue input, analogue output, digital input and digital output as well as multiple counter/timers. To operate all of these tasks simultaneously, a data acquisition device must process multiple streams at the same time and be able to quickly move data to and from PC memory.

NEW APPROACH

What is needed is a new approach to handling multiple streams of data, one that delivers some of the same characteristics as using DMA on PCI and PCI Express.

NI has developed signal streaming technology that enables sustained high-speed and bidirectional data streams over USB.

Before explaining NI signal streaming, it is important to review USB communication and legacy USB DAQ designs.

USB is a serial protocol that transfers data such as addresses, requests and confirmations in packets and it shares available bandwidth across all devices present.

USB systems incorporate three components – the host (typically a PC or laptop), a USB device and the USB cable. The USB host initiates all data transfers to a USB device, thus requiring the transmission of three or more packets for a simple transaction – host request, data transfer, and host confirmation.

Information is sent to and from a USB device through its endpoints. An endpoint is equivalent to a gate through which all data must pass to get on or off the USB device. Endpoints can transfer data only in one direction, requiring devices to have multiple endpoints to achieve both input and output functions. Data is transferred to or from the device endpoints to the host controller using four basic types of transfers, one for USB bus control and three others for data transfer.

Interrupt data transfer provides low-volume data with timely and reliable data transfers; often used for a peripheral device such as a keyboard or mouse.

Isochronous data transfer offers pre-negotiated bandwidth with possible data loss; often used when on-time data delivery is more important than data accuracy, such as streaming audio and video.

Bulk data transfer delivers large data transfers with no loss of data; often used for applications where lots of data must be transferred with no loss of data such as external hard drives.

With built-in error checking and large data transfer capabilities, bulk data transfer is the logical choice for USB data acquisition. But in practice, bulk transfers can achieve fast performance when only a single stream of data is sent through the USB port. Any additional streams can considerably reduce transfer rates, especially if the transfers are bidirectional.

TRADITIONAL APPROACH

Using the traditional approach, there are three technical issues that contribute to the inability of USB to meet throughput and latency expectations for high-performance data acquisition.

The first issue is the inefficient use of a processor for data management. The current standard for data management between the data acquisition front end and USB endpoints is a slow, commercial off-the-shelf (COTS) processor. Its purpose is to divert the appropriate information to each individual USB endpoint. Compared to a custom design, it is an inexpensive solution with a fairly easy implementation.

These processors are instruction-based, single-threaded pieces of silicon that create a switch-like behaviour. Only able to run one single instruction at any given time, they are the bottleneck of all typical USB data acquisition devices.

Any higher-priority tasks, such as handling control requests from the host, may delay the processor in handling data transfers, causing low throughput and possible buffer underflow/overflow errors. Due to its architecture, this type of processor cannot simultaneously manage several data streams and handle incoming control requests, making high-performance data acquisition impossible.

In other words, USB cannot achieve the PCI DMA-like performance of streaming multiple I/O types simultaneously.

TIME DELAY

Another issue is latency. Latency is defined as the time delay between initiating a request for data and the beginning of actual data transfer.

USB has higher latency than PCI because of its OS-based access and serial nature. A typical single-point voltage acquisition could require dozens of register-level instructions, which, when multiplied by 1 ms of latency, is unacceptable.

Finally, there is a trade-off between data throughput and latency. To have a fast data throughput rate, engineers need to move larger data sets at every transfer. Doing so reduces the responsiveness of the system and increases system latency, thus leaving the device unresponsive during each large transfer. ▶