GPIB: Challenges and Potentials

Overview

Adopted by a wide range of instrumentation applications, the General Purpose Interface Bus (GPIB) is now being challenged by emerging standards such as USB and LXI. This article examines the development potentials for GPIB and analyses the future of this technology. A brief overview discusses the GPIB's achievement in hardware implementation for extension in various form factors including ISA, PCI, USB and LAN, and its software compatibility to advance VISA. This article compares GPIB with newer instrument interface standards such as USB and LXI and explains the pros and cons of each technology. Summing up the article is a presentation of the advantages, disadvantages, and challenges of an alternative concept that implements the protocol control code of the GPIB bus instead of the traditional GPIB ASIC controller.

GPIB in retrospect

The IEEE 488 standard, better know as the General Purpose Interface Bus or GPIB, is a popular interface that connects instruments to computers to form automated test equipment (ATE). GPIB was developed initially by Hewlett-Packard and was recognized as an IEEE standard in 1978. Since then, the IEEE has released IEEE 488.1(1978) to define the GPIB hardware specifications including its electrical, mechanical and basic protocol parameters, and IEEE 488.2(1987) to define related software specifications. GPIB is widely-accepted and used by instrument vendors for decades. It can be noted that GPIB today is the most popular interface between computers and instruments.

Current challenges to GPIB

Revolutionary changes being applied to the PCI I/O bus, such as higher throughput and smaller mechanical footprint, have boost the popularity of legacy ISA bus and more mature PCI bus standards. These standards have significantly surpassed the RS-232 in terms of speed. On top of these are the USB and LAN interfaces which are proven to be more versatile, faster, and higher in performance. Because of their cost-effectiveness and easy connectivity, all current PCs are equipped with both USB and LAN interfaces. Meanwhile, after more than three decades of enhancements and wide-ranging development, countless traditional GPIB devices now support hot-plug functionality and remote access to keep in pace with IP-based instruments which test engineers are more familiar with. Because of these parallel GPIB innovations, it will be difficult for newer and faster I/O interfaces - USB or LAN - to completely replace the standard in the ATE industry.

Realising the GPIB's strong position in this market, major instrument manufacturers implemented a bridge communication protocol from GPIB to USB or LAN. This move takes advantage of the flexibility and high data throughput of USB and LAN, while maintaining GPIB-based instrument investments in full operation. The bridge communication protocol comes in a form of a GPIB to USB/LAN adapter, GPIB-LAN gateway, and converters. This and similar technological trends indicate that GPIB is here to stay.

From the hardware point of view, GPIB's interconnectivity with USB and LAN delivers faster integration and more convenient maintenance as it is no longer necessary to physically install an

ISA/GPIB card into an available expansion slot. The collaboration thus lowers the cost and complexity of test systems.

From the software viewpoint, combining GPIB with USB/LAN is similarly advantageous as most mainstream operating systems are capable of monitoring the LAN or USB port status so any newly-connected instruments are automatically detected and recognised. Highly integrated environments, such as the Agilent I/O Library Expert, automatically recognises connected instruments and dispatches a dedicated software resource for it, thus eliminating the need to manually search, identify, and initialise connected devices.

Adopting GPIB: What you need to know

Before implementing automated test equipment, try answering the following questions/considerations:

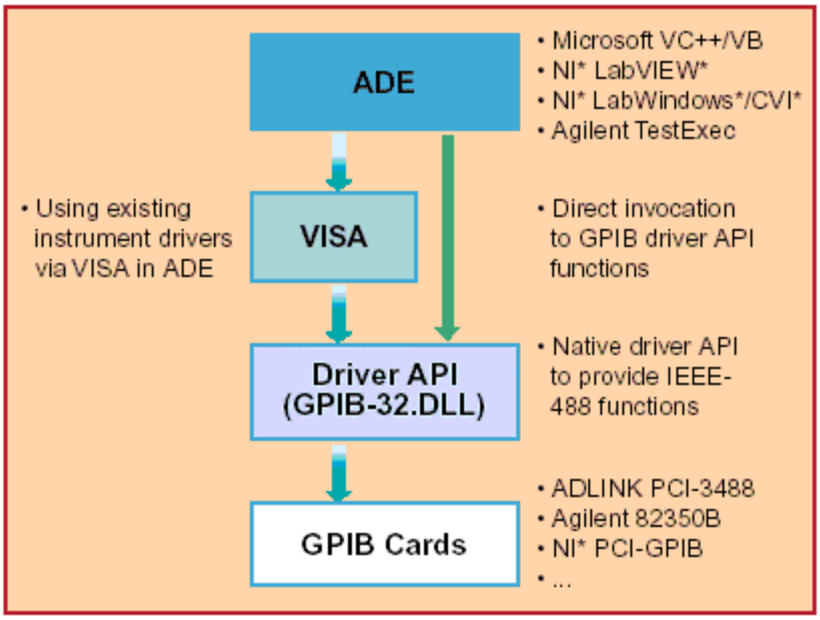
- What is your preferred physical interface that will connect with your instrument: GPIB, USB, or LAN?
- What are the software requirements, specifications, capability, and performance?
- Based on the selected software application, what software development environment will you use to control and communicate with the instrument?

Supposing the user decided to adopt GPIB as the interface to control instruments, the next step is to determine the I/O software kits for instrument communication. These I/O software kits are regarded as a software layer positioned between the integrated application designs and the physical interface to instruments. There are two ways to create an automated test application: perform native driver API or via high-level instrument drivers.

The first method involves native driver API conventions. These API conventions are usually provided by most adapter vendors and come in the form of ANSI C functions. For users requiring a more detailed instrument control and maximum system throughput, using a driver API with SCPI string commands is highly-recommended.

For users who want to keep away from complicated instrument commands, high-level instrument drivers such as VISA or IVI-COM is the ideal solution. VISA is a type of software interface that provides standardised input and output functions to communicate with measurement instruments.

High-level instrument libraries provide transparent software compatibility with all types of connection interfaces, and come with functions that are mostly independent from the device interface used. Whether you intend to access measurement instruments through RS-232, GPIB, USB, or LAN, high-level instrument drivers do not give you the additional burden of modifying software codes when you change the communication bus type. Lastly, high-level instrument drivers give you more time to focus on software development and make user programs more scalable for later re-integration.



After deciding on the software I/O layer, the next consideration would be to select the most appropriate ADE (Application Development Environment). The ADE and software tool kit combination is crucial and may directly impact the total cost of development and the application completion time. Because of this, serious considerations must be made in terms of software price and time to learn or training. You may also consider using available software development kits which accelerate test system development. Software developments may be divided into two groups: graphical and textual.

Currently, there are plenty of graphical programming environments that cater to test and measurement engineering. The most popular ones are Agilent VEE and NI LabVIEW? – These software are intended for novice programmers because of their user-friendly GUI and programming approaches. An easy-to-learn graphical programming environment allows rapid creation of prototype test systems and enables the user to efficiently handle the data flow between several concurrent activities. The straightforward programming offered by graphical environments is far easier compared with creating a program using textual programming. In addition, you do not need to learn complex syntax as it allows you to learn and share predefined codes easily.

Textual ADE programming is suitable for large-scale applications and for improving system throughput. However, this type requires an experienced programmer. The good news is that in recent months, the run-time performance difference between graphical and textual programming is shrinking.

GPIB under the scope

Fast and reliable connectivity is the most important consideration for instrument manufacturer and users. With the ever-increasing performance of commercial desktop and notebook PCs comes an evolution of communication fundamentals between PCs and instruments. In spite of GPIB being the de facto standard for connecting instruments and the PCI bus as the standard I/O interface for industrial control and measurement, new generation USB and Ethernet are crossing the line and venturing into instrument control. Because of this, it is relevant to evaluate and compare these interface standards. Here is a table of comparison between the standards based on key specifications.

	PCI-based GPIB	USB 2.0	Gigabit LAN
Theoretical bandwidth*	1.8 MB	480 Mbps	1000 Mbps
Maximum connection length (no repeater)	20 m	5 m	100 m
Transmission latency	Minimum	General	General
Real-time response	Best (with SRQ line)	Normal (N/A)	Better (depend on traffic)
Connector mechanism	Robust	Normal	Better
Expansion capability	Normal (needs additional slots)	Good (via a hub)	Best (via Ethernet)
Portability	X	Best	Good
Installation cost	Good	Best	Good
Hot-swap operation	X	Good	Good
Remote and shared access	X	X	Good

* The theoretical bandwidth is not identical with the bus throughput and depends on the host computer's processor speed, installed devices, and variations in data block size bursts.

Selecting which I/O interface to use is the first task in building your automated test equipment. You may use a legacy instrument with a pure GPIB implementation or adopt a modern instrument with LAN or USB. A combination of GPIB and USB/LAN offers a comprehensive solution to meet all kinds of requirements. As mentioned previously, when all your chosen instruments support high-level drivers such as VISA, so you can take advantage of the flexibility in creating hybrid, high-performance test systems.

GPIB controller: The FPGA alternative

The GPIB bus controller - a key GPIB component - is an ASIC (Application-Specific Integrated Circuit). Since very few suppliers produce this component, the cost of ASIC-based GPIB is expensive. Even though there are assurances from component vendors that ASIC-based GPIB is more superior in performance, adopting ASIC-based GPIB has not become a cost-effective solution versus FPGA implementation especially for prototype verification or small-scale production.

With current advancements in electronic design automation (EDA) tools, the FPGA (Field Programmable Gate Array) offers an alternative solution to expensive ASIC implementations. In the meantime, as more and more FPGA standards emerge and as verified IP cores become readily available from the Internet, building the GPIB protocol into FPGA is already in the horizon and opens a promising opportunity for test and measurement applications.

We have asserted that the GPIB continues to serve as a reliable I/O interface for a vast number of instruments that are still in use. GPIB also provides a convenient way to manage complex hardware handshaking. However, critical time-to-market considerations are forcing GPIB interface hardware designers and instrument manufacturers to release their products in the market early on. The time-to-market pressure is immediately diminished by carefully selecting the communication protocol between the host controller and the remote device, and by understanding the GPIB's electronic signals. The sooner these parameters are satisfied, the faster the product development.