

## **Measurement Computing introduces a new cross-platform DAQ solution for OEMs**

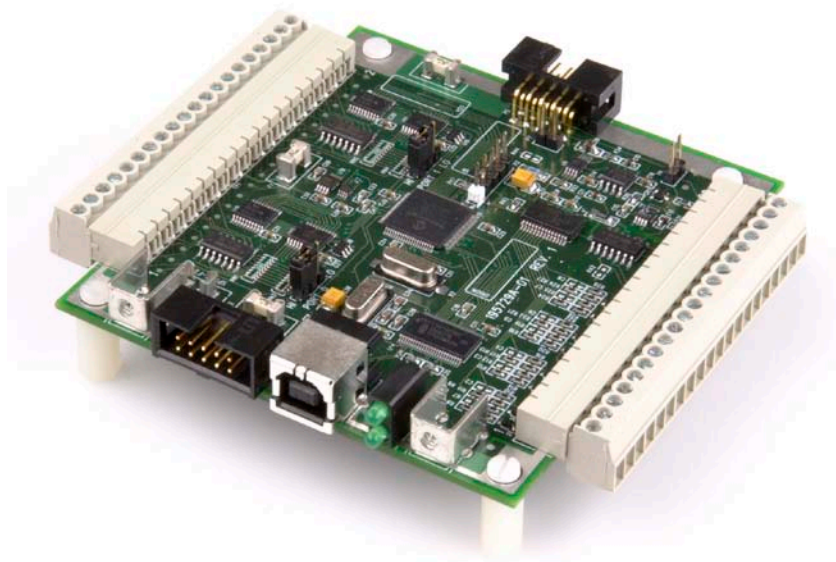
### *Overview*

Measurement Computing has long served the OEM data acquisition market by providing off-the-shelf products that deliver performance and ease-of-use at a low price. This approach provides a great solution to many OEM customers who need Microsoft® Windows® support or a broad selection of form factors including PCI, PCI Express®, and USB.

Increasingly, however, OEMs are looking for more, including:

- The ability to develop an application on one platform and easily port it to another platform.
- Support for non-Windows operating systems
- Small driver footprint
- Low-cost
- Small form-factor

To meet these needs, Measurement Computing is introducing a new line of data acquisition products designed from the ground up for the growing OEM DAQ market.



This new product line—the MCC 7000 series—combines a small form-factor, bus-powered USB hardware, and a developer-friendly software framework which can be ported to multiple operating systems. The Message-Based DAQ software framework—a core technology to the MCC 7000 series—is further explained in this white paper.

## What is Message-Based DAQ (MBD)?

Message-Based DAQ is a well-defined protocol that permits the programming of DAQ devices using simple text-based messages. A common command set greatly simplifies driver and application development.

The MBD DAQ framework (*Figure 1*) consists of 3 sections. The MBD Application API and the Device Driver reside in the computer, while the MBD Firmware API resides in the data acquisition device.

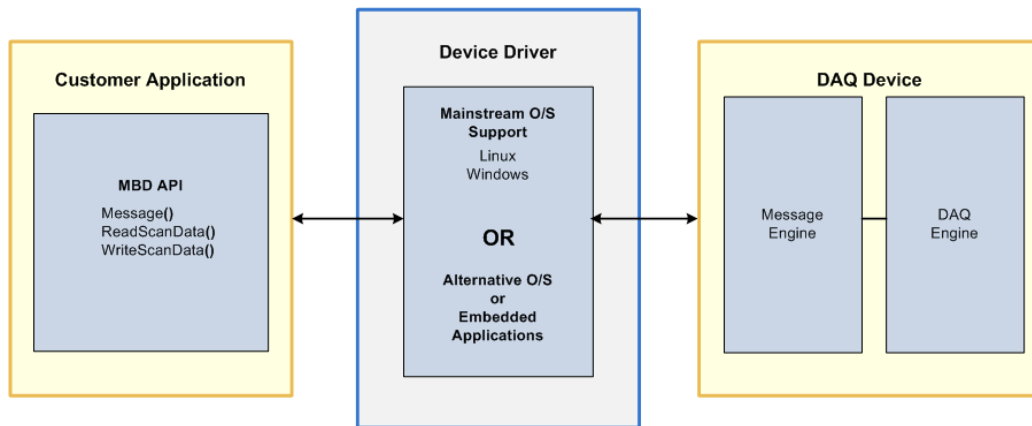


Figure1 – MBD Framework

### MBD Application API

Message-Based DAQ has a simple API that is common to all MBD devices. This API was developed to allow users to write application code that is operating system independent. With the MBD Application API, developers only need to use a handful of methods allowing for a short learning curve and rapid application development.

### Device Driver

Applications for Message-Based DAQ devices can be developed using standard USB drivers like WinUsb for Windows, Libusb for Linux<sup>®</sup> or Measurement Computing-provided drivers. In these cases, driver development is already done, enabling OEM developers to focus on software applications without worrying about the underlying communication details.

OEMs who need support for an alternative O/S or custom applications can build their own driver. Since the MBD framework is open source, Measurement Computing provides OEM developers with the detailed documentation and technical support required including example code.

### MBD Firmware API

The MBD firmware API consists of efficient, text-based messages that are sent to the device through USB Control\_Out transfers. The firmware in the device message engine parses and converts these messages into DAQ-specific commands that control the device or process data. The device then responds to the driver via a subsequent Control\_In transfer. Larger, higher throughput data sets are handled by Bulk\_In and Bulk\_Out transfers to and from the driver.

The Message-Based DAQ cross-platform architecture offers many advantages to systems designers including a very small memory footprint, and an easy to learn API and command set.

## Message-Based DAQ Framework Detail

### MBD Application API

The Message-Based DAQ application API is cross-platform and open source, and includes a library, an out-of-the-box application that runs on Windows and Linux, and programming examples in C# and VB .NET. The source code is included for all of these components.

Additionally, the MBD software requires only a handful of API methods—one for sending commands to a device and others to manage data transfers. The intuitive command set makes programming an MBD device a simple task. Below is a code example:

```
DaqResponse = MyDevice.SendMessage("AISCAN:RANGE=BIP10V")
DaqResponse = MyDevice.SendMessage("AISCAN:LOWCHAN=0")
DaqResponse = MyDevice.SendMessage("AISCAN:HIGHCAN=3")
DaqResponse = MyDevice.SendMessage("AISCAN:RATE=1000")
DaqResponse = MyDevice.SendMessage("AISCAN:SAMPLES=4096")
DaqResponse = MyDevice.SendMessage("AISCAN:START")
ScanData = MyDevice.ReadScanData(4096)
```

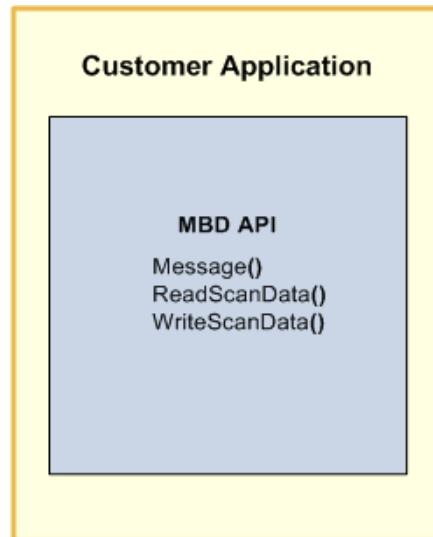


Figure 2 – MBD Application API

This simple and intuitive API provides:

- A minimal collection of methods which promotes a simplified learning curve for application developers
- A cross-platform architecture
- An extremely small SW footprint
- Flexibility with regard to various data types
- Management of memory allocation and buffer indexing
- Scalability

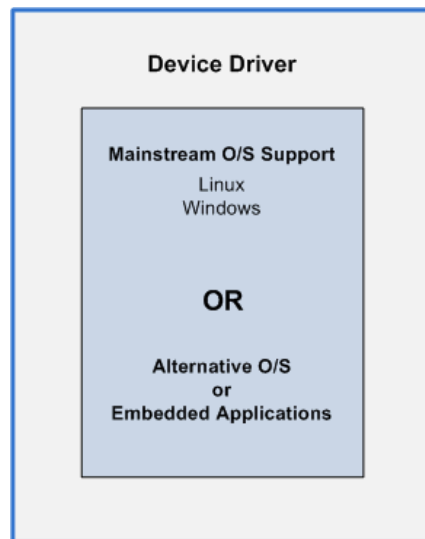
The MBD API Library is written in C#, and runs on Windows using the Microsoft .NET framework and Common Language Runtime (CLR). It also runs on Linux using the open-source Mono® framework and CLR.

The library is compiled to a format that consists of Common Intermediate Language (CIL). The MBD API can be used to develop and build an application on one platform, and deploy the same binaries to other platforms. When the executable is launched on the target platform, the CLR compiles the executable to native code and runs the application as native code.

### MBD Device Driver

Communication with a hardware device on any operating system requires a device driver. With Message-Based DAQ, there are several device driver options (*Figure 3*):

- Use of standard user-mode drivers such as WinUsb and Libusb
- Use of Measurement Computing-developed drivers
- Development of device drivers for other operating systems or custom embedded applications



*Figure 3 – Driver Options*

Applications for MBD devices can be developed using standard USB drivers like WinUsb for Windows, Libusb for Linux or Measurement Computing-provided drivers. In these cases, driver development is already done, enabling OEM developers to focus on software applications without worrying about the underlying communication details.

For OEMs who want more control, or who are writing a driver for an alternative O/S or custom applications, driver developers require the following:

- Concise firmware API documentation
- Experience developing custom drivers
- Knowledge of USB enumeration, control, and bulk transfers

In this situation, MCC provides OEM developers with the detailed documentation and technical support required.

## MBD Firmware API

Unlike traditional DAQ devices that have different register maps or protocols within a family, Message-Based DAQ standardizes on a specific command set utilizing USB control transfers. This common command set eliminates the need to write a new driver as new devices are implemented. All MBD devices share the same firmware interface.

The MBD protocol consists of efficient, text-based messages that are sent to the device through USB Control\_Out transfers. The firmware in the device “message engine” parses and converts these messages into DAQ-specific commands that control the device or process data. The device then responds to the driver via a subsequent Control\_In transfer. Larger, higher throughput data sets are handled by Bulk\_In and Bulk\_Out transfers to and from the driver (Figure 4).

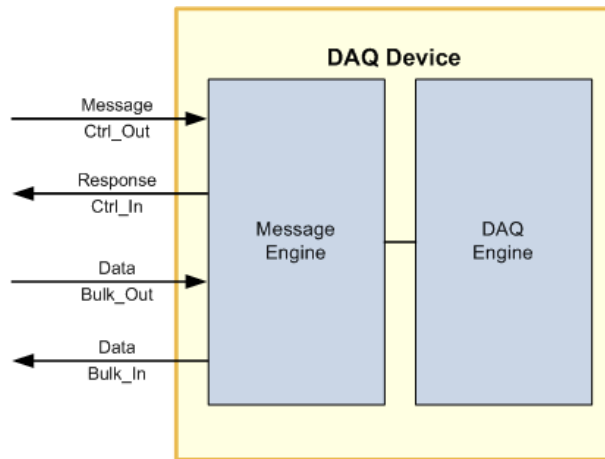


Figure 4 – MBD Device Transfer Types

MBD firmware standardizes on these communication mechanisms. These mechanisms, along with the common command set facilitate development of an application that works across all MBD devices - Only the message content and the product device ID's need to change.

*The MBD firmware interface provides developers with more options for developing DAQ applications and drivers that can target multiple platforms (Figure 5)*

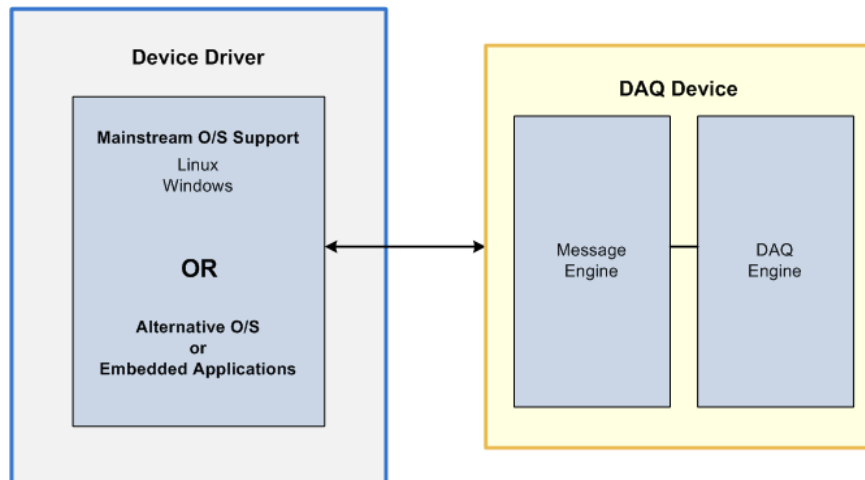


Figure 5 – MBD Driver/Firmware Interface

## Getting Started with Message-Based DAQ

In addition to a simple application API, an interactive C# console application is included with your Message-Based DAQ product that enables you to see your new product in action—on Windows and/or Linux—minutes after you unpack it and install the software.

This application automatically recognizes available MBD devices, and detects the text-based commands supported by the selected device. You can then send a message to a device and observe its response.

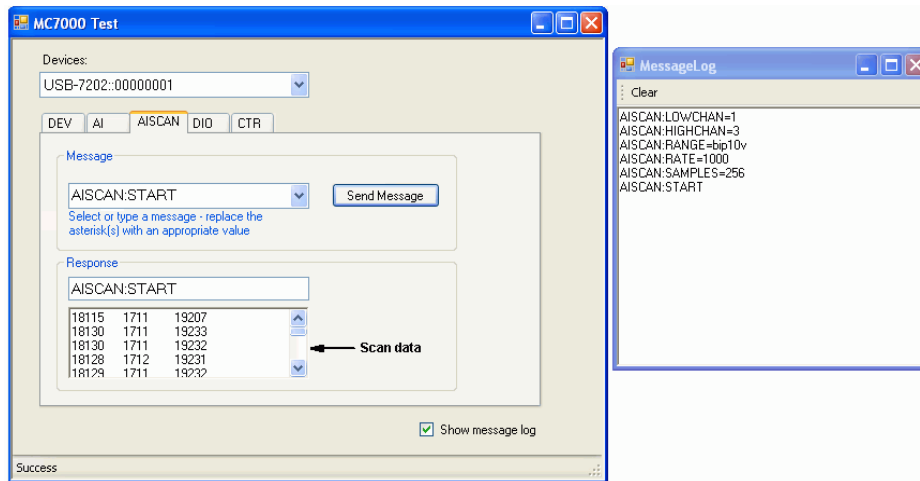


Figure 6 – MBD Test Application

### Conclusion: A simple and flexible solution for OEMs

Measurement Computing has developed a new architecture specifically for OEMs. The Message-Based DAQ architecture is ideal for developers who value a small driver footprint, an easy to learn API, and the ability to port their application to virtually any operating system.

The common command set and standardized firmware interface support:

- OEM development of products that operate across mainstream operating systems (Windows and Linux to start, with more planned for future releases)
- OEM development on lesser-known or proprietary operating systems
- OEM development of products with no target operating system—products that simply communicate over a USB root port in their embedded system

The MBD application API is open source so OEM developers can strip it down, enhance it, and otherwise modify it depending on their specific needs.

Measurement Computing continues to build on two decades of success as the market leader in low-priced data acquisition products. MBD is not a replacement for our established hardware and software offerings, but an added product offering for new and existing customers.

To learn more about the MCC 7000 series, go to [www.mccdaq.com/7000-series.aspx](http://www.mccdaq.com/7000-series.aspx).

---

© Copyright 2009 Measurement Computing Corporation. All rights reserved.

Microsoft and Windows are registered trademarks of Microsoft Corporation in the United States and other countries.

Linux® is the registered trademark of Linus Torvalds in the United States and other countries.

Mono is a registered trademark of Novell, Inc. in the United States and other countries.

PCI Express is a registered trademark of PCI-SIG.