Global Data Sharing Overview

Global Data Sharing™, or GDS, is included with the supply of each 2350 Digital Controller in the form of a Windows software component supplied with MTL or MS Control Software. All current 2350 software products are GDS compatible, which drastically simplifies the development of added capabilities or test application programs to the standard package.

GDS is a proprietary software technology that provides several significant benefits to test engineers:

- Permits concurrent tasks and applications to share real-time control input and output data.
- Real time interface with Microsoft Office products.
- High performance application development capability for users familiar with Microsoft Office macros.
- Open source application specific software products enabling users to change and add new functionality.
- Capability for multiple tasks to concurrently handle an ongoing test.
- Capability for real-time debugging new application software.

An example test application where GDS uniquely serves needs is in the lab having the need to “oversee” the quality of test control, such as in a fatigue test. The test engineer may insist on monitoring how accurately the control software program maintains the cyclic mean and amplitude in a long duration fatigue test, so that a qualification certificate based on real measured data may be issued.

With standard fatigue test software, the machine supplier’s process of performing the test and collecting test results is generally accepted to be the basis of good test quality. Independent verification of actual loads applied is then effectively ruled out, except in the few cases where customers hook up an additional load cell or a separate data acquisition system is used to independently measure applied force versus time.

Commercial test software supplied by test machine manufacturers suffer the problem that the test engineer has no means of observing in “real-time”, data flow through the controller and computer system. Test machines today employ ‘host’ Microsoft Windows computers which, since Windows 98 and NT operating systems (OS), all ‘feature’ a 32-bit memory protection scheme, with the consequence that multiple tasks cannot share common data. Memory space belonging to a particular task cannot be accessed by another task. Data sharing under Windows is implemented by copying data to an intermediary (e.g. resource on the OS), which in turn, passes it on as “legal tender” to the target application. Microsoft Office applications also share ‘phased-in-time’ data using that scheme, which supposedly provides memory protection against corruption by other tasks. Arguably, the increased overhead goes unnoticed in non real time operations, given the speeds of today’s PCs.

However, the scheme presents serious shortcomings with regard to any real-time application. Significant PC slowdown and memory “hogging” can occur as multiple copies of large amounts of data are passed to different participating tasks, as shown in figure 1a below. In the case of real-time data, it is very unlikely that participating tasks “see” the same data at the same time. An individual load readout may not “appear the same” at a given time across multiple tasks and give rise to conflicting decision making among tasks.

Our goal behind GDS was to regain the advantages of multi-tasking without compromising synchronization and throughput of real-time data flow among tasks. In GDS, we implemented a data structure and access scheme within Microsoft Windows that protects the integrity of the application and, at the same time, eliminates the need for data messaging – See figure 1b.

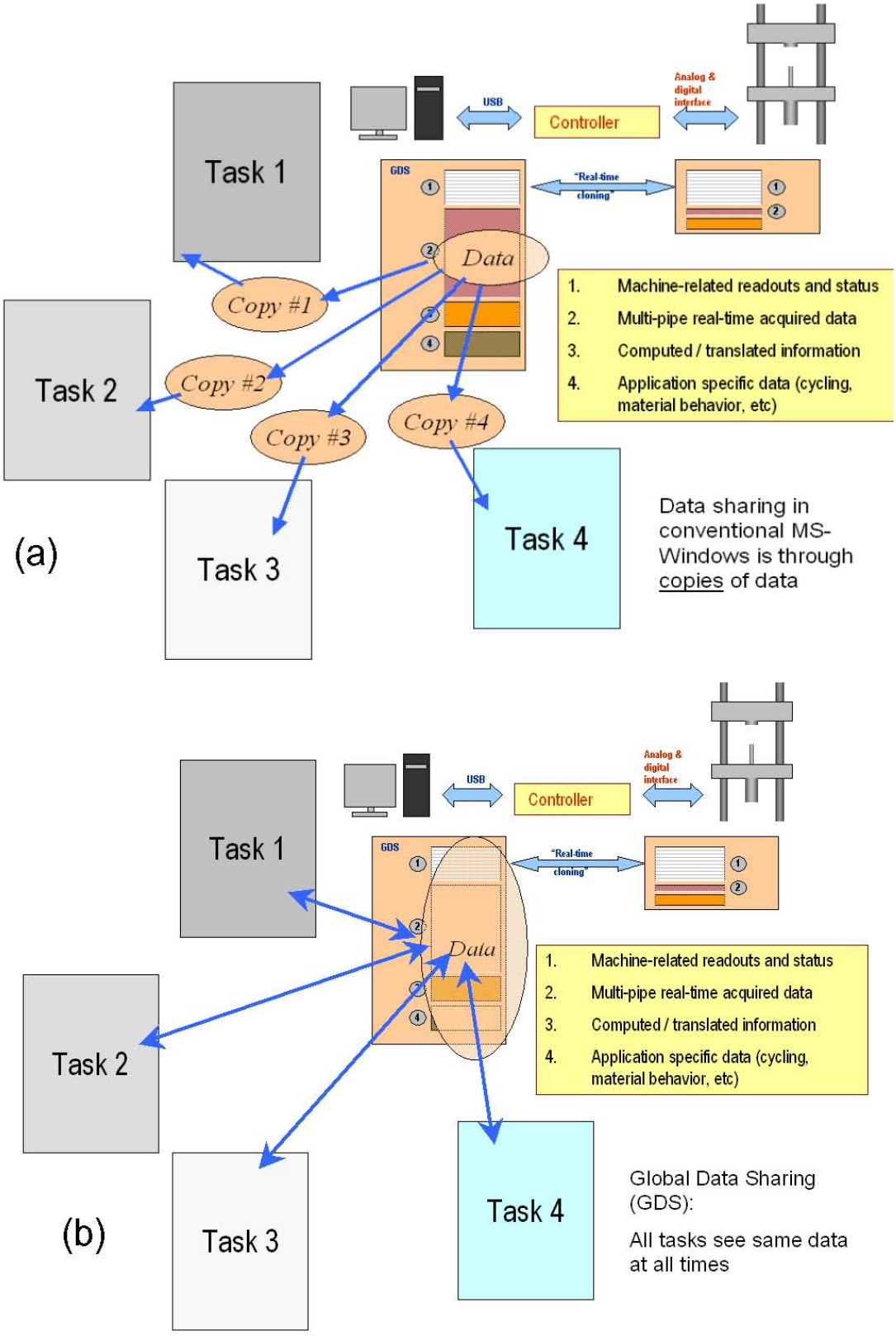

FIG. 1 – (a) Conventional data sharing between concurrent tasks in Windows (Dynamic Data Exchange, or DDE and ActiveX) drags down real-time multi-tasking speeds. (b) GDS environment enables concurrent tasks to directly access the same physical address space, speeding up real-time computing.

All participating tasks directly and concurrently operate on both static as well as dynamic areas of a Virtual Machine (VM) that we created. The contents of the VM appear as arguments in individual tasks and so when a value is changed by one of the tasks, all other tasks instantaneously “see” the new value – even if one or more of them is in debug or trace mode. Dynamic data are in the form of circular buffers carrying incoming or outgoing data from/to the controller. The buffers are large enough to carry data covering over a 100 seconds of transfer. Different tasks may access them asynchronously --- without affecting their integrity. If the OS briefly hangs due to time critical operations on the disk or on the network, the real-time data flow, even if interrupted, will not lose data.

Up to 64 independent tasks can “register” for access to GDS. The process ensures tasks unrelated to concerned applications cannot corrupt the GDS. The process also enables the operation of more than one GDS, making it possible for multiple test systems to be connected to a single ‘host computer’ and for multiple sets of the same or different applications to hook up to individual GDSs. New versions under development offer the promise that GDS can access across networks, including the Internet.

Eventually other application programs, including laboratory management software, will be able to interface to multiple test systems in real time, and remote machine control can be implemented.

GDS is software platform independent. Code developed on different development platforms works together, allowing end-users and developers alike to choose the platform of their choice to control their test system. Currently, we offer developers kits with sample real-time applications written in Visual C++, VB, MS-Office Macros, Lab-Windows, Lab-View and Delphi.

User-access to the real-time application using GDS implicitly offers customers increased security and control over their test machine and the quality of data it produces.

Applying GDS to Microsoft Office

Microsoft Office has become the market leader for test documentation and report generation.

Microsoft Excel specializes in providing relatively complex processing, data arrangement, and vast storage of data. Excel provides for data representation in user-definable report format, and features Visual Basic Macro (VBM), enabling user-written code to process and store information in Excel Worksheets.

The GDS environment offers users the freedom to extend useful Excel features to real time test activities by enabling VBM applications. A user can now create a special application to set up and control a test in real-time, running concurrently with the standard MTL or MS Control Software supplied with the 2350 controller. The application could be developed to collect, process and store test data in the manner required for the test or the users preferences. An application can be created to perform any test, however complex it may appear. Figure 2 shows a typical user interface for an Excel based fatigue crack growth test application, where real time calculated variables are used to perform machine servocontrol.

Such an Excel file can store the test assignment including operator and customer details, collected data, processed results as well as the Test Report. The same file would contain the VBM code used to execute the test and process results. With this open source approach, test data and report guarantees a higher degree of traceability which is not possible without GDS. In this case, the code is unprotected, and so becomes open source and amenable to change, including improvement by others.

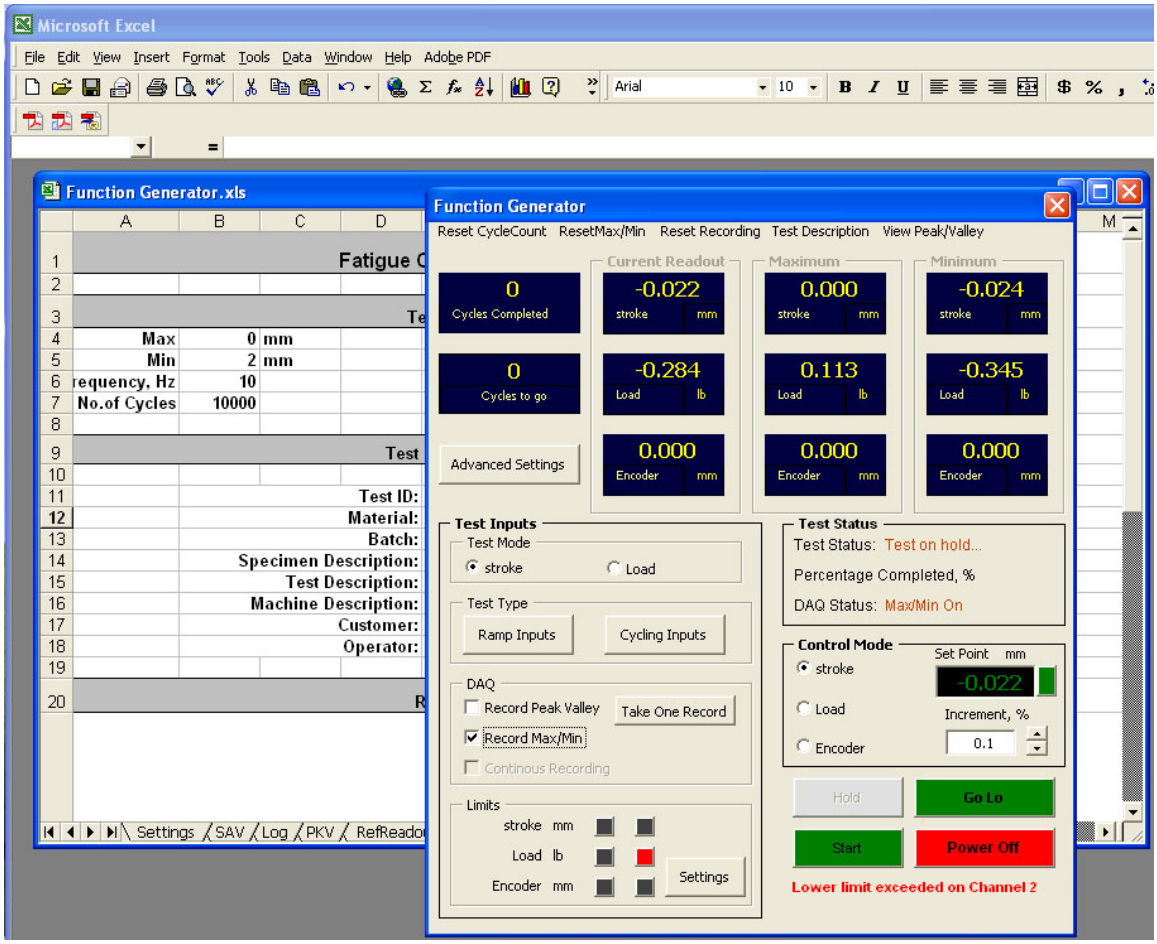

FIG. 2 – Example of a user interface of an Excel-based real-time application written as VB macros. Real-time commands to perform test control as well as data acquisition are embedded in the Excel file as open-source code. Data are written directly into the desired Excel worksheets. The end user is free to process the data in a different way if desired either manually, or, by writing new macros. The source code can also be changed if required to control the test in a different manner and to perform additional operations as required. Most end users are comfortable working in a software environment that appears familiar and intuitive.

Open source software is gaining popularity as users demand fair trade practices that do not confuse knowledge with technology. GDS provides a unique opportunity to develop open source software, reducing the cost of routine testing.

Contact us for more information.