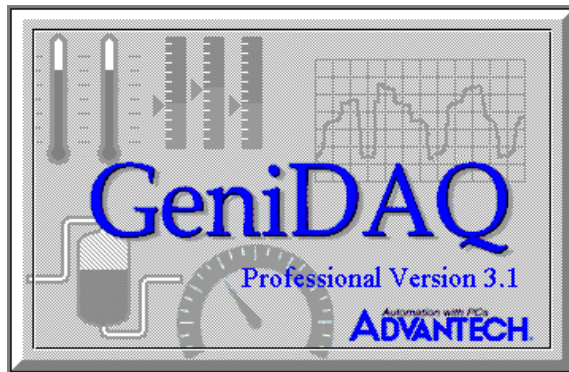# User's Guide

Advantech

GeniDAQ

2nd Edition

Contents

Table of Contents

Copyright
Notice
Trademarks
Technical Support

PART I: Getting Started

1 Introduction

2 Preparing to Install GeniDAQ

3 Tutorial

PART II: Essential Skills

4 Device Installation and Configuration

5 Task Designer

6 Display Designer

7 Report Designer

8 GeniDAQ System Administration

9 GeniDAQ Runtime

PART III: Advanced Skills

10 Script Designer

11 Data Center

12 Writing User Defined DLL's

Appendix A

Appendix B

Copyright

Trademarks

Technical Support

AMERICAN ADVANTECH provides telephone support to registered users of GeniDAQ.

If you purchased your GeniDAQ package from a dealer or your hardware interface vendor, the dealer or vendor may have special arrangements for providing you with technical support.

If you encounter a problem while using GeniDAQ, first consult the on-line help and this manual. If you are unable to resolve the problem, or if you have questions specific to your application, don't hesitate to FAX or call for technical support. You can also look for Q&A at http://www.advantech.com.tw

USA:

American Advantech Corporation
750 East Arques Avenue
Sunnyvale, CA 94086
(408) 245-6678
FAX (408) 245-5678
E-mail: IAInfo@www.advantech.com

ASIA:

Advantech CO., LTD.
Fl. 4, No. 108-3, Ming-Chuan Rd.
Shing-Tien City, Taipei, Taiwan, R.O.C.
TEL: (886-2) 2218-4567
FAX: (886-2) 2218-1989
E-mail: iasupport@acl.advantech.com.tw

EUROPE:

Advantech Germany
Karlsruherstr. 11/1
D-70771 Leinf.-Echterdingen
Germany
TEL: +49(0) 711 797 333 60
FAX: +49(0) 711 797 333 85

Advantech Italy
Via Don Verderio,
4/B-20060 Cassina de,
Pecchi (MI), Italy.
TEL: 39-2-95343054
FAX: 39-2-95343067

MAINLAND CHINA:

Beijing office:
No. 7, 6th street, Shang Di Zone,
Haidian District, 100085 Beijing,
China
TEL: 86-10-62984345~47, 62986314~17
FAX: 86-1-62984341~42

Shanghai office:
Rm#701, 7 F., Hua-Fu Building A,
585 Long Hua W. Rd.
200232 Shanghai, China
TEL: 86-21-64696831, 64697910
FAX: 86-21-64696834

Figures

1

Getting Started

1 Introduction

1.1 GeniDAQ Overview

Congratulations on your purchase of ADVANTECH'S application builder for data acquisition and control — GeniDAQ.

GeniDAQ is a comprehensive application development tool for data acquisition and control. It supports functions and utilities to develop automation applications for use in Windows 3.x and Windows 95 environments. GeniDAQ provides an icon-based, mouse driven system for designing real-time Automation and Control Strategies, System Monitor Displays, and Dynamic Operator Displays.

The magic of GeniDAQ lies in its ability to provide advanced programming features and tools while maintaining ease of use. A library of Icon Blocks representing data acquisition and control, mathematical and control functions is provided through Task Designer. You simply arrange Icon blocks into a strategy, connect them, and then draw your dynamic display or configure your daily report. Display Designer provides a variety of graphic objects to design monitoring and control displays. Report Designer features a configurable format design utility and scheduler to generate reports automatically. In addition to the features listed above, GeniDAQ's built-in VBA compatible programming tools strengthen its ability to perform complex calculation or analysis.

The GeniDAQ package consists of two major software modules and several utility programs. The two major executables are GeniDAQ Strategy Editor/Runner ("GENIE.EXE"), and GeniDAQ Runtime only program ("GWRUN.EXE"). GENIE.EXE is for designing and testing Strategies. GWRUN.EXE is for running strategies in a live environment. GWRUN.EXE uses fewer resources and achieves better runtime performance because it does not perform validity checks on objects and links between objects. It does not allow any changes to be made to the strategy being run.

1.2 GeniDAQ System Architecture

A major improvement over Version 2.xx is the change in system architecture. We design GeniDAQ 3.x with a modular-oriented, open integrated architecture. The open platform allows you to easily integrate GeniDAQ with other applications to share real-time control data. The performance and number of I/O blocks GeniDAQ can support are increased significantly through this new architecture. The new architecture is outlined in figure 1-1.
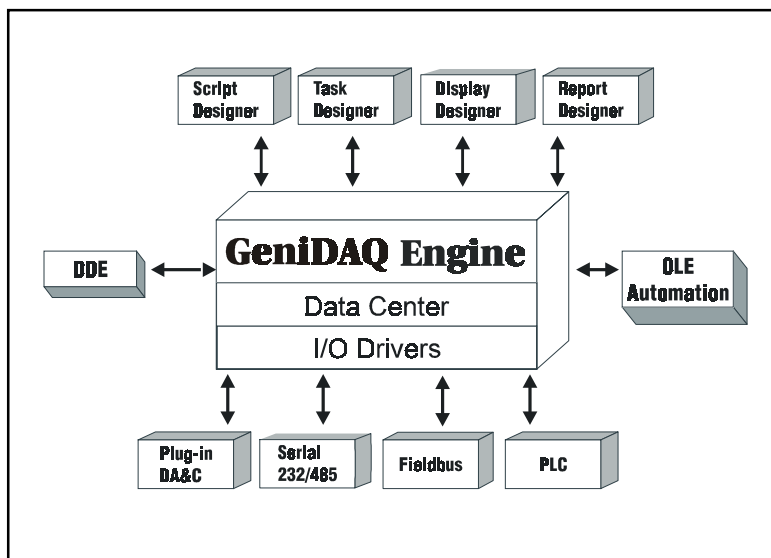
Figure 1-1   GeniDAQ 3.x system architecture

Module Description

Data Center
The central repository for data acquisition and control data. It manages all GeniDAQ real-time data and provides three set of interfaces to the outside world; DDE & OLE Automation and C API. Through these three interfaces, other applications can retrieve or input data to GeniDAQ.

I/O Driver
Responsible for accessing real-time data from hardware equipment. GeniDAQ I/O drivers cover all Advantech industrial automation hardware, including plug-in DA&C cards, PC-based modular controller MIC-2000, ADAM-4000 remote modules and ADAM-5000 distributed modules.

Task Designer
Responsible for designing the application system. GeniDAQ provides many standard data acquisition and control function blocks. Users just drag and drop function blocks to design a system without any programming. GeniDAQ 3.x allows you to design and execute multiple tasks simultaneously. That is, you can divide a complex system into several independent tasks and execute them at the same time.

Display Designer
Responsible for creating a dynamic display for industrial automation. Display Designer features many common graphic objects to simplify building a display. In addition, GeniDAQ provides several drawing tools to create custom displays.

Report Designer

Responsible for designing and generating operation reports. Users can design the report format using a tabular form tool and define a schedule to print the report automatically. Report Designer will periodically collect the real-time data and combine the data with data format to generate report at the scheduled time.

Script Designer

Responsible for controlling tasks and calculating and analyzing real-time data. GeniDAQ provides a VBA compatible development environment for script programming.

The design work in GeniDAQ is saved as a strategy file. A strategy file (with extension .GNI) is a binary file that stores all information about an editing session. A strategy is defined as one or more "tasks" together with one or more "displays" and one or no "main script". "Task", "Display", and "Main Script" are the three primary elements used to design strategies. A simple strategy has one task with one display and no main script, because a strategy can not function without at least one Task in it. Task are essential to have a strategy. Because the most important job of a task is to provide timing for "scanning", so "Scan Task" is used interchangeably with "Task" throughout this manual.

A task is a collection of icon blocks. A display is a collection of display items. Icon blocks and display items are the building blocks of strategy. You can see them as similar objects, except that display items have a graphic representation (provide some kind of GUI) when running while icon blocks do not. They are connected to each other by connection wires or invisible links. The connections between icon blocks are visible in the task designer window. The links are referred to as "wires" because of their appearance. The connection between an icon block and a display item, or one display item and another, is not visible. Hence, these are referred to as "Links".

Each task and display has its own properties. When first created, default properties are assigned to the new task or new display. Users can change properties according to their needs. A task has properties such as scan period, sample rate, starting method and stopping method. A display has properties such as display name, visibility, etc.

GeniDAQ features four different editors: Task Designer, Display Designer, Report Designer and Script Designer. They are used to edit Tasks, Displays, Reports and Main Script respectively. Because there can be multiple tasks in one strategy, you can create multiple Task Designer windows inside GeniDAQ. For the same reason, you can create multiple Display Designer windows for editing multiple Displays. In any application there can be only one Main Script. Therefore, only one Script Designer window can be opened. GeniDAQ can have many other types of script, which will be described in Chapter 10.

Any number of Blocks and I/O Devices can be used at one time, limited only by your system's speed and memory. It is suggested that the number of icon blocks not exceed 500. If the number of icon blocks exceeds 500 system performance may be adversely affected. From small applications interfacing only a few blocks, through full-scale industrial process control systems running many I/O Devices simultaneously, GeniDAQ provides you with the quickest and most efficient acquisition/control solution.

1.3 GeniDAQ Component Description

The intent of this section is to provide a general idea about what's been added to this new version of GeniDAQ. GeniDAQ has been transformed from data acquisition and control application software to an application builder for all kinds of automation. We have placed an emphasis on expandability, capacity, and programmability. The architecture is modified to be more open, allowing access to all data items in GeniDAQ. Also, hardware support has been greatly expanded. Hardware support now covers virtually all Advantech hardware families, as well as a myriad of products from other vendors.

1.3.1 VBA Compatible Script Designer

GeniDAQ Script Designer is a VBA-compatible basic script engine. Script Designer not only features a robust Visual Basic programming engine, but also includes many tasks and real time data access functions. Through this script engine, users can call DDE, OLE Automation and ODBC (SQL) functions to integrate with other applications.

VBA Compatible Script Editing
The script designer is basically a text editor with some convenient features for editing script code. The script source will be compiled into p-code after editing so it won't need to be compiled again at runtime. The syntax of the BasicScript is compatible with Microsoft VBA (Visual Basic for Application in Excel, Word, Access, etc.) and Microsoft Visual Basic. It is possible to take a Visual Basic source code and compile and execute it under BasicScript without changing a word if only common functions are used. The two development environments are very similar. In the source code design stage it features cut, copy and paste functions.

Script Debug Functions
The GeniDAQ editing program can run strategies in a debug mode, in which you work through a program line by line and review a scan task block by block. This major improvement allows users to design and debug complicated strategies in the editor before using the runtime only program. The runtime program now uses even less memory and offers improved performance.

Programming Tasks and Real-time I/O Directly
The Script Designer is used for editing the main script and scripts inside a task. The main script controls the entire runtime, including starting a task and/or stopping a task. In addition to task management, GeniDAQ provides a variety of commands to process I/O data.

Main Script
The main script is used to control and manage tasks. The initialize statement initializes all the data related to a task. The Start/Stop statement is used to start and run a task to completion. Single Scan statement will do a one-time scan of the task specified.

Pre-task Script
The pre-task script is used to define task properties and initialize all data related to a task.

Post-task Script
The post-task script is used to clear-out task related data.

Basic Script Icon

BasicScript is used to get, analyze and set I/O data.

1.3.2 Task Designer

GeniDAQ Task Designer uses a dataflow programming model that frees you from the linear architecture of text-based languages. To create a process monitoring and control application, you construct the block diagram without worrying about the many syntactical details of conventional programming. Simply select objects (icons) from the Icon Toolbox and connect them with wires to pass data from one block to the next.

Multiple Tasks to Improve Performance

GeniDAQ Task Designer allows editing of multiple tasks at the same time. Each task is contained in a task window and has it's own properties such as scan rate, start/stop method, etc. One strategy file is used to store all scan tasks that are related to a control strategy. For simple strategy with only one task, it runs the same way as before. But for strategies that have more than one scan task, a top-level main script is required to manage the execution of all scan tasks. GeniDAQ supports up to 8 tasks.

A large complicated task can be broken into several smaller, simpler tasks. This not only simplifies the editing job, but also increases the performance at runtime, as fewer blocks need to be processed at each scan.

Block Sequence Arrangement

GeniDAQ Task Designer features the block sequence arrangement functions that shows the order of execution on all blocks. With displayed order number, users can arrange the order of execution of the blocks (icons) based on the priority of operations to meet the requirements of system needs.

Virtual Tag

The Virtual Tag is a powerful feature that provides the ability to let developer to create customized tag in Task Designer without using User Defined DLL. The virtual tag is created by Task Designer and stored in data center as other built-in blocks. The virtual tags are global available to all tasks, so you can use virtual tags to share data among multiple tasks.

1.3.3 Display Designer

If you have MMI requirements, GeniDAQ screen designer will help you quickly create intuitive standard graphical displays by provided graphical wizards. And, you can further customize your MMI with drawing tools and user-defined display tool.

Drawing Tools

GeniDAQ enhances the man-machine interface (MMI) by providing graphic tools to draw pumps, valves, rectangles, circles, segments, and polygons in the screen designer's toolbox. In addition, it allows the user to configure the colors and sizes of these figures. These drawing tools include oval, rectangle, round rectangle, polygon and line. In addition to drawing tools, GeniDAQ also provides "Make Object" and "Break Object" commands to let you integrate drawing components into a meaningful picture for your data acquisition and control.

1.3.4 Report Designer

GeniDAQ Report Designer provides a configurable environment in which users can define the contents of a given report. It collects the required data at specific time intervals and these reports are printed automatically at a user-defined time. The interfaces provided in Report Designer may also be used to select and print reports manually.

GeniDAQ Report Designer includes five major functions: data collection, report format configuration, report scheduler, report generation and alarm report generation.

Data Collection
Data collection function creates database files (.DBF files) for each defined TAG point at a user-specified time. Data collection function is designed for report generation. The shortest time interval for data collection from a given TAG point is 10 minutes. If needed, high-speed data collection is accomplished through other trend data collection functions.

Report Format Configuration
Report format configuration function provides user interface dialog boxes, which allow users to set up the report format and report print time. Report format entries are organized in a table form and users enter text or specified keywords to define each table column. Information from each report format is saved to a format file, and extracted during report generation.

Report Generation
Report generation function combines the format file and data collection database file to produce a printed report. At the moment, reports are limited to tabular format report. A graphical report with daily trend output will be developed in the future.

Report Scheduler
Report scheduler sets the time at which reports will be generated. At a user-defined time, report scheduler calls the report generation function to generate the report. Report Scheduler also informs users of report printing status.

Alarm Report
Alarm report generation function produces equipment fault reports. These reports provide information on time of fault occurrence, operator acknowledgment and recovering records.

1.3.5 Open Data Center

Data Center is the central repository for all process monitoring and control data in GeniDAQ. You can easily integrate real-time data from the data center into a company-wide information system to efficiently support decision-making. Data Center features two different sets of interfaces to the outside world: <u>C API</u>, <u>OLE automation,</u> and <u>DDE</u> functions.

C API is the most efficient interface, so it is used for all communication between components of GeniDAQ. OLE automation is designed to interface with OLE-aware applications.

1.4 What's New in GeniDAQ 3.1

Compatible with Windows 98/95

GeniDAQ 3.1 has been thoroughly tested for use with the Microsoft Windows 98/95 operating system.

Year 2000 Compliance

GeniDAQ 3.1 is Y2K compliant. If you are currently using Advantech GENIE 3.0, download the GeniDAQ patch file from Advantech's web site to upgrade your existing software to Y2K compatibility.

Enhanced Installation Program

GeniDAQ 3.1 includes an installation and uninstallation program. Executing the uninstallation program will automatically remove all program files, directories and Windows registry entries. You do not have to delete any files manually.

Improved Reliability and Robustness

GeniDAQ includes many substantial improvements over Advantech GENIE 3.0. GeniDAQ 3.10 provides memory monitoring functions and protection against insufficient hard disk space during continuous operation. In addition, it is protected against improper use and hardware failure.

Other Enhancements

GeniDAQ 3.1 includes a few new functions over Advantech GENIE 3.0. These include new Basic Script commands, an improved Report Designer and other features that lead to higher productivity.

Limitations and Restrictions

In order to assist users in using GeniDAQ more effectively, this release note includes more information about usage and limitations with respect to using Basic Script, the Report Designer, deploying GENIE in a network environment, etc.

1.4.1 Compatible with Windows 98/95

GeniDAQ 3.1 has resolved the compatibility problems that Advantech GENIE had with Windows 98/95 operation. This includes the issues with Alarm Printing and Networking functions.

Note: Since GeniDAQ is a 16-bit application, it cannot write data to a file with a long
          filename, e.g., Log_File_Block.txt. However, it can read data from a file with a long
          filename, e.g., Input_File_Block.txt, Conditional_Wavefile_Block.txt and
          Conditional_Bitmap_Display.txt.

1.4.2 Year 2000 Compliance

GeniDAQ 3.1 is Y2K compliant. The allowable date range is from January 1, 1980 to January 18, 2038. This limitation is due to Microsoft Visual C++, the development tool that was used to write the program code. If you are a GENIE 3.0 user, please download the GeniDAQ patch file from Advantech's web site to upgrade your existing software for Y2K compliance.

Detailed information about how GeniDAQ handles date information is as follows:

(1) Item: Time Stamp Block

The date format of Time Stamp Block is MON Sep 23 14:43:52 1992. Note that the year is in a four-digit format.

(2) Item: Timer Clock Block

For Time of Date or Elapsed Time (both cyclic and non-cyclic), the block output is based on the seconds elapsed since midnight (i.e., 00:00:00) January 1, 1970 Universal Coordinated Time.

(3) Item: Alarm Log Block

If the Alarm Message Format: Date (MM/DD/YYYY) option is checked, the date message for the alarm will be stored in the GENIE.ELF file in MM/DD/YYYY format, where YYYY represents the year. It is a four-digit year format, and thus Y2K compliant. This file is used in the Report Designer for a listing report.

(4) Item: Historical Trend Block

This stores one file per day in binary format and the filename is YYMMDDXX.HST, where YY represents the year. For example, if it is 1998, then YY = 98. For the year 2001, YY = 01. If your system is running from December 31, 1999 to January 1, 2000, GeniDAQ will store the historical data into two binary files: one is 99123101.HST and the other is 00000101.HST.

You can search back and forth to view the data on December 31, 1999 and January 1. 2000. It displays the date as month/day/year, where the year is in a four-digit format.

In addition, the HIST Conversion under the File menu can accept a four-digit year in the Start Date and End Date fields for the conversion range. GeniDAQ can retrieve the right historical binary file (*.HST) and convert it to an ASCII file (*.TXT).

(5) Item: Log File Block

For the Open method: "Open at n minutes from midnight" under Advanced Option, the file will be opened according to the day which is independent of the year.

(6) Item: Runtime Error Message File: RUNERR.LOG

The date format is MM/DD/YY HH:MM:SS. Although the year is stored in two-digit format, this is only for storage purposes.

(7) Item: Basic Script Commands

The date-related commands (i.e., Date and Year) both accept and output 4-digit years.

(8) Item: Report Designer

For the formatted report, i.e., Monthly/Yearly reports, the database filename is MNYYYYMM.DBF where YYYY represents the year. It is in a four-digit format.

For the listing report, i.e., Alarm/Control report, it will retrieve the correct data for the report date which accepts a four-digit year format.

The date-related commands: $DATE outputs four-digit year format like MM-DD-YYYY.

1.4.3 Improved Setup/Uninstallation Utility

GeniDAQ 3.1 includes a comprehensive installation and uninstallation utility. After installing GeniDAQ 3.1, it will create an Advantech GeniDAQ folder by default. It contains the following icons: GeniDAQ Builder, GeniDAQ Runtime, Device Installation, Uninstall GeniDAQ, User Guide Help, Runtime Help, Basic Script Help and Release Notes. If you wan to uninstall GeniDAQ, you simply click on the Uninstall GeniDAQ icon. GeniDAQ, including program files, installation folders and Windows registry entries, will be automatically removed from your computer. You do not have to worry about deleting files manually.

1.4.4 Improved Robustness and Reliability

Program robustness and reliability have been improved considerably in this release. GeniDAQ 3.1 provides memory monitoring functions and protection against insufficient hard disk space during continuous operation. In addition, there is also improved protection against improper use and hardware failure.

(1) Memory Monitoring Functions

A new tag named "FreeSpace" has been added under "VIRTASK". It is used to monitor the free memory space of the system when running. To enable this function, the user has to set a value in the \Windows\GENIE.INI file manually as follows:

```
[System]
MemoryCheck = 1
```

After adding this entry to the GENIE.INI file you should open GeniDAQ and create a tag block. Configure this tag block by selecting VIRTASK under the Display/Virtual Tag field, and then select FreeSpace in the Tag Name field.

(2) Protect Historical Data Against Insufficient Hard Disk Space

In the Historical Trend Display, GeniDAQ will record the historical data to files by date. When GeniDAQ is run continuously for long periods of time, there is a chance of running out of hard disk space. To protect against this, GeniDAQ 3.1 removes the files generated by Historical Trend Display automatically. It automatically removes historical files that are older than 30 days by default. Users can change this period by setting "HistDaysAgo" under the [System] section in the \Windows\GENIE.INI files as follows:

```
[System]
HistDaysAgo = 15
```

This modification will cause GeniDAQ will remove historical files that are older than 15 days.

Note: If you want to keep all historical files, you should set "HistDaysAgo = 0". GeniDAQ will not remove any historical files.

In addition, GeniDAQ 3.1 keeps the runtime from logging the same error messages. This will save space within the log files and lessen the chance of running out of hard disk space.

(3) Protect Software Against Improper Use and/or Hardware Failure

To protect GeniDAQ runtime from a strategy that is partially or improperly configured, GeniDAQ 3.1 requests the user to configure all icons correctly. Partially configured or improperly configured strategies will not be allowed to run in GeniDAQ 3.1. In addition, this protects GeniDAQ from hardware malfunction by isolating the input.

1.4.5 More Enhancements

GeniDAQ 3.1 provides some new functions, including

(1) A New Basic Script Command

Command	Description
Quit(0)	Stops the overall system

(2) A New Decimal Formatting Command in Report Designer

This command is used to specify the number of decimal digits in the Report Designer. The old version is fixed to 2 decimal digits. In GeniDAQ 3.1, it can be configured by the user. The commands format is as follows:

```
COMMAND.N
```

where "COMMAND" can be $NOW, $HRnn, $MAX, $MAXT, $MIN, $MINT, or $AVE. "N" represents the number of decimal digits which can be from 0 to 9. For example, if a tag's value is 5.123456789, you can use the follow commands to format the value:

Command in format file	Result
$NOW(@TASK1#AI1) ms	5.12 ms (by default)
$NOW(@TASK1#AI1).0 ms	5 ms
$NOW(@TASK1#AI1).1 ms	5.1 ms
$NOW(@TASK1#AI1).9 ms	5.123456789 ms

(3) Network Communication Status in Event Windows and Log Files

For networking communication status, the connection/disconnection messages will show in the Event window. The will also log into the alarm file (GENIE.ELF) by enabling the Runtime Preference event log in GeniDAQ.

(4) Save and restore previous values

This function is used to save the values of control displays at system stop. These values will then be restored at the next system start. This function is included in Binary Button Control , Conditional Button Control, Numeric Control, Knob Control, Slider Control and Log File Block.

1.7   Limitations and Restrictions

GeniDAQ 3.1 includes the following information about usage and limitations of the software to help users use GeniDAQ correctly.

(1) Basic Script Programming

Category	Task	Description
ScanTask	Start/stop a task	* To start/stop a task by Script, the task's starting method must be Inactive in Task Property *In Basic Script block or Pre/Post-task Script, it cannot start or stop the task by itself. * The possible task names passed into GetScanTask ("task name") command refer to "the tags created by GeniDAQ".
Tag	Retrieve/ set value to or from Data Center	* To retrieve/set values of tags in the Data Center via Basic Script. Make sure that the data type is correct. Refer to the Data Center for data types of tags. The user can also use Variant type in the Script to retrieve or set values of tags. * If the channel count of the specified tag is greater than 1, the user needs to use the Tag.Array[n] command to retrieve/set a value for channel n. Make sure n does not exceed the maximum number of channels. If using the Tag.Value command to retrieve/set a value, it will be the channel 0. * The possible tag names refer to "the tags created by GeniDAQ".
Output commands in Basic Script block	Output value in Basic Script to another block	* Don't use these commands in main script or pre/post-task script. It is only for a Basic Script block. * Select the correct command to output the value. For example, if you need to output a string, select OutputS command and pass a string type parameter.
Quit(0) command	Stops the overall system	* new command

Notice

(a) In the debug environment for the Main Script, Pre/post-task Script or Basic Script block, the following commands are disabled:

- ScanTask.start
- ScanTask.stop
- ScanTask.SingleScan
- ScanTask.GetStatus
- OutputI
- OutputL
- OutputF
- OutputS
- Display

- Quit

However, all commands will function at runtime.

(b) When using the Msg command to create modeless dialogs or the Sleep command for waiting, do not stop GeniDAQ when the modeless dialog is still active or the Sleep command is still working.

(c) Be careful when programming the script to avoid an infinite loop.

Limitations

(a) Strings are limited in length to 32764 characters.

(b) The data area that holds public variables is limited to 16 KB.

(c) The size of source code script is limited to 65534 characters.

(d) Arrays can have up to 60 dimensions.

(e) Variable names are limited to 80 characters.

(f) Labels are limited to 80 characters.

(g) The number of open DDE channels is not fixed; rather, it is limited only by available memory and system resources.

(h) The number of open files is limited to 512 or the operating system limit.

(i) The size of an array cannot exceed 32 KB.

(2) The Tags Created by GeniDAQ are the Following:

Block	Tag name	Data type	Channel count
Analog Input	TASK1/AI1	Floating point	32
Analog Output	TASK1/AO1	Floating point	1
Digital Input	TASK1/DI1	Integer	1
Digital Output	TASK1/DO1	Long	1
Temperature	TASK1/TMP1	Floating point	1
Hardware Counter	TASK1/CTFQ1	Floating point	1
Hardware Alarm	TASK1/ALM1	Long	1
RS-232	TASK1/SER1	String	8
Input File	TASK1/INF1	Floating point	1
Alarm Log	TASK1/ALOG1	Long	1
Ramp	TASK1/RMP1	Floating point	1
Moving Average	TASK1/AVG1	Floating point	1
Single Calculation	TASK1/SOC1	Floating point	1
Timer	TASK1/ET1	Long	1
Time Stamp	TASK1/TS1	String	1
Counter	TASK1/CNT1	Long	1
On/Off Control	TASK1/ONF1	Integer	1
PID Control	TASK1/PID1	Floating point	1
DDE Client	TASK1/DDEC1	String	1
Conditional Sound	TASK1/SOUND1	Long	1
Beep	TASK1/SP1	Long	1
Button Control	DISP1/BBTN1	Long	1
Knob Control	DISP1/KNOB1	Floating point	1
Numeric Control	DISP1/NCTL1	Floating point	1
Slider Control	DISP1/SPIN1	Floating point	1
Ameter Display	DISP1/METER1	Floating point	1
Bar Display	DISP1/BAR1	Floating point	1
Indicator Display	DISP1/INDI1	Long	1
Drawing Display	DISP1/CELL1	Long	1
Conditional Button	DISP1/CBTN11	Long	1
Conditional Bitmap	DISP1/BMP1	Long	1
Conditional Text	DISP1/CTXT	String	1
Historical Trend	DISP1/HIST1	Floating point	8
Virtual Tags	VIRTASK/V1	Floating point	1

Notice

(a) TASK1 can change to TASK2 to TASK8 that depends on the tag in which TASK. The tag name consists of the block type and ordinal number; for example, AI block with ordinal number 3, then the tag name is AI3.

(b) DISP1 is the title of the display that is assigned in Display Property Menu.

(c) VIRTASK stores the VIRTAG tags that are added by the Add/Delete Virtual Tags menu.

(d) The maximum string length for string type tags is 128.

(3) Report Designer

(a) $NOW command applies to all tags created by GeniDAQ. Please refer to "The Tags Created by GeniDAQ".

(b) $HRnn, $MAX, $MAXT, $MIN, $MINT and $AVG commands only apply to the following tags:

- Analog Input (AI)
- Analog Output (AO)
- Digital Input (DI)
- Digital Output (DO)
- Temperature (TMP)
- Hardware Counter (CTFQ)
- Hardware Alarm (ALM)
- Time Stamp (TS)
- Counter (CNT)
- PID Control (PID)
- On/off Control (ONF)
- Ramp Block (RMP)
- Moving Average (AVG)
- Single Calculation (SOC)
- DDE Client (DDEC)
- VIRTASK's Tags (VIRTASK/User-defined Tag names)

(c) Add a new Decimal Formatting command. It is used to specify the number of decimal digits for Report Designer. The old version is fixed to 2 decimal digits. In GeniDAQ 3.10, it can be configured by the user. The command format is as follows:

```
COMMAND.N
```

where COMMAND can be $NOW, $HRnn, $MAX, $MAXT, $MIN, $MINT or $AVE. N represents the number of decimal digits which can be from 0 to 9. For example, if a tag's value is 5.123456789, you can use the following commands to get the following results.

Command in format file	Result
$NOW(@TASK1#AI1) ms	5.12 ms (by default)
$NOW(@TASK1#AI1).0 ms	5 ms
$NOW(@TASK1#AI1).1 ms	5.1 ms
$NOW(@TASK1#AI1).9 ms	5.123456789 ms

(4) Alarm Printing

GeniDAQ 3.1 resolves compatibility problems between the alarm printing function and Windows 95/98. If you will be using this feature in GeniDAQ 3.1, we strongly suggest that you use a dedicated dot-matrix printer for alarm printing.

According to the Microsoft Knowledge Base, Windows 95/98 does not support single line printing.

Note:

(a) Alarm printing does not work if the printer is shared

(b) Alarm printing is only compatible with dot matrix printers or other printers capable of printing a single line or a single line feed. It will not work with HP LaserJet printers that must print a complete page per printing session.

(5) Historical Trend

(a) Support up to 6 Historical Trends. Each Historical Trend can accept up to 8 traces.

(b) GeniDAQ 3.1 removes the files generated by Historical Trend Display automatically. It removes historical log entries that are 30 days old by default. Users can change it to the desired number of days by setting "HistDaysAgo" under the [System] section in the \Windows\GENIE.INI file as follows:

```
[System]
   HistDaysAgo = 15
```

This example will cause GeniDAQ to remove historical file that are older than 15 days old.

Note: If you want to keep all historical files, you should set "HistDaysAgo = 0". GeniDAQ will not remove any historical files.

(6) Networking Support

(a) One station can connect to a maximum of fourteen stations, the maximum number of output stations is seven and the maximum number of input stations is seven. There is no limitation on the total number of stations in a network.

(b) One station can setup network input and network output blocks simultaneously. The maximum number of network blocks is 100; this allows a maximum of 800 points in the network.

(c) GeniDAQ has a dedicated timer for network communication. The polling time of the timer can be different from the scan time. The polling time is set in the configuration file \WINDOWS\GENIE.INI (the entry NetPollTime). The value of polling time depends on the number of stations and blocks. You can use the following equation to calculate the approximate polling time:

Polling time = (Station * (Station - 1)) / 2) * base polling time

Total time for transferring all blocks

= (Network blocks/10) * polling time

which base polling time depends on your PC's performance. Based on our tests, for three stations running on a Pentium 166 with 64M RAM, and forty blocks for each station, the polling time can be up to 500 milliseconds.

(d) The maximum string size for each channel of a network block is 32 characters (including a Null character).

(e) GeniDAQ networking can work in Windows 95/98 but not in Windows 3.1.

(f) When restarting networking, you must wait for a few moments to make sure that the remote station receives the stopping message. If you fail to check this, proper connection with the remote station may not occur. If this happens, you will have to stop all stations and start again. You can check the Event window to make sure that the remote station receives the stopping message.

(g) Connection status will show in the Event window. It will also be entered into the alarm log file (GENIE.ELF) by enabling the event log under Runtime Preferences.

(7) Input Range for Display Items

Display Items	Field	Range
Bar Graph	From	-999999.0 to +999999.0
	To	-999999.0 to +999999.0
Trend Graph	Range of X axis (From)	-999999.0 to +999999.0
	Range of X axis (To)	-999999.0 to +999999.0
	Range of Y axis (From)	-999999.0 to +999999.0
	Range of Y axis (To)	-999999.0 to +999999.0
	Update Rate	0 to 9999
XY Graph	Range of X axis (From)	-999999.0 to +999999.0
	Range of X axis (To)	-999999.0 to +999999.0
	Range of Y axis (From)	-999999.0 to +999999.0
	Range of Y axis (To)	-999999.0 to +999999.0
	Size of the Trace Buffer	0 to 8192
Group Box	Border Width	0 to 32767
Numeric/String	Update Rate	0 to 9999
Numeric Control	Privilege Level	0 to 255
	Initial Value	-999999.0 to +100000000.0
	Step Value	-999999.0 to +100000000.0
	High Limit	-999999.0 to +100000000.0
	Low Limit	-999999.0 to +100000000.0
Knob Control	Privilege Level	0 to 255
	Initial Value	-5 to 5
	Start Tics	2 to 40
	End Tics	2 to 40
	Tics Rate	2 to 40
	Decimal Places	0 to 6
Anameter	Tics Number	2 to 40
Historical Trend Char Span		depends on graph size and resolution
	Number of Tics	2 to 40

2

Preparing to Install GeniDAQ

2 Preparing to Install GeniDAQ

GeniDAQ requires specific hardware and software to be installed in your computer before installation Verify that your computer system conforms or exceeds the requirements defined in the sections *System Hardware Requirements* and *System Software Requirements* before attempting to install the GeniDAQ system software.

2.1 System Requirements

Your computer system must meet or exceed the following requirements in order to run GeniDAQ.

System Hardware Requirements
- IBM PC/AT 486 chip compatible computer or higher
- 16 MB RAM memory (minimum)
- 10 MB free hard disk drive space (minimum)
- One CDROM drive
- VGA controller and monitor
- Serial or PS/2 mouse
- Math co-processor recommended

GeniDAQ requires that certain operating systems and system settings be present on your system. The following are minimum requirements to run GeniDAQ successfully.

System Software Requirements

- **DOS version 3.3 or higher**
- **Windows 3.1, Windows 95 or Windows 98**

In addition, there must be a CONFIG.SYS file in your root directory that includes the following statements:

```
FILES = 50
BUFFERS = 10
```

- DOS TSR (terminate, but stay resident) application programs may be loaded while using GeniDAQ. Remember that these programs do take up memory and can affect system performance.

- An extended memory manager, such as MS-DOS HIMEM.SYS, QEMM, or EMM386 must be installed.

2.2 Installation and Configuration

This section lists the contents of your GeniDAQ system and instructions to install the GeniDAQ system software to a level that enables you to run the DEMO Strategy supplied on the SETUP disk.

GeniDAQ Package Contents

Your GeniDAQ package includes the following items:

- GeniDAQ Setup CD, including all GeniDAQ files, DLL drivers and example strategies

- This manual (Advantech GeniDAQ User's Guide)

GeniDAQ Program Installation

Complete the steps listed below to install and configure your GeniDAQ software.

During the installation process, the setup program (SETUP.EXE) allows you to specify where you would like the GeniDAQ software installed. The default destination directories are:

GeniDAQ specific files: `C:\GENIDAQ`
Example strategies: `C:\GENIDAQ\STRATEGY`

The program group "Advantech GeniDAQ" will be automatically created within WINDOWS, and the eight GeniDAQ icons (GeniDAQ Builder, GeniDAQ Runtime, Device Installation, Uninstall GeniDAQ, User Guide Help, Runtime Help, BasicScript Help and Release Notes) will be placed inside.

If you make an error when entering a response to a question, simply click the "Back" or "Retry" button to return to the previous step. The installation process can be aborted at any time by clicking "exit".

1. Turn on your computer and boot up with DOS 3.3 or above.

2. Once in WINDOWS, insert the GeniDAQ Setup CD into the CDROM drive.

3. The installation Program will be launched automatically. Or you can use your Windows Explorer or the Windows Run command, execute the following file on the GeniDAQ installation CD-ROM disc

       `D:\disk1\Setup.exe`

Where D is the drive into which you inserted your GeniDAQ Setup CD and click Run..

Following on-line description, you can install GeniDAQ on your computer with ease.

When all the files have been copied, you should see the **Advantech GeniDAQ** group created within Windows. The eight GeniDAQ icons will be displayed in the group window. The setup screen will return to display a success message. GeniDAQ is now installed. Should you have any problems installing GeniDAQ (any error messages), call Advantech Technical Support.

Modify GENIE.INI in Windows directory, for example C:\WINDOWS, if GeniDAQ doesn't install in C:\GENIDAQ. Then change the installation path stored in GENIE.INI.

```
[Report Management]
INSTALLPATH=C:\GENIDAQ
```

For example, if GeniDAQ installs in C:\GENIDAQ.310, modify GENIE.INI as follows:

```
[Report Management]
INSTALLPATH=C:\GENIDAQ.310
```

Networking Installation

The Network I/O feature in GeniDAQ allows you to transfer data from any block over a Local Area Network (LAN) that supports Novell's IPX protocol (Novell NetWare is not required). Two blocks are used to support this feature: Network In and Network Out.

(For Windows 3.1)

To execute the network feature, the user must setup the network settings in the SYSTEM.INI file in the Windows directory as follows:

```
[boot]
network.drv=netware.drv
or
secondnet.drv=netware.drv

[386Enh]
network=vnetware.386, vipx.386
or
secondnet=vnetware.386, vipx.386
```

(For Windows 95/98)

The user must install three network components in the Network icon under Control Panel as follows:

```
[Configuration Tab]
Client for Netware Networks
NE2000 Compatible
IPX/SPX-compatible Protocol
```

You must run the network driver (ipx.com or ipxodi.com) before running WINDOWS. In addition, three files must be present in your working directory or \WINDOWS directory: nwipxspx.dll, tli_spx.dll and tli_win.dll. The GeniDAQ setup program installs the three DLLs.

Testing with Demo I/O Device

The DEMO I/O device is a software I/O board simulation that allows you to demonstrate all of the GeniDAQ functions without connecting to actual I/O hardware. The DEMO I/O device consists of three channels of analog input and two channels of digital I/O. Analog input channel 0 simulates a ± 5 volt sine wave; Channel 1, a ± 5 volt square wave; and Channel 2, a ± 5 volt saw tooth wave. The digital I/O consists of two bits (channels) of digital input that are internally connected to the two channels of digital output, to simulate an I/O board that has a hardware loop cable attached between the digital input and output connectors.

The DEMO Strategy, DEMO01.GNI, is a simple strategy that interfaces the DEMO I/O Device to the Display.

1. Load the DEMO Strategy into GeniDAQ, enter the GeniDAQ Task Designer by double clicking on the GeniDAQ Builder icon in the Advantech GeniDAQ group. Once in the GeniDAQ Main screen, click the File menu, then open an existing file. You will see a displayed directory tree containing the GeniDAQ directory and the STRATEGY directory. Double click on the Strategy directory, and you will see all *.gni files contained in the STRATEGY directory. Highlight the one called "DEMO01.GNI", and press OK.
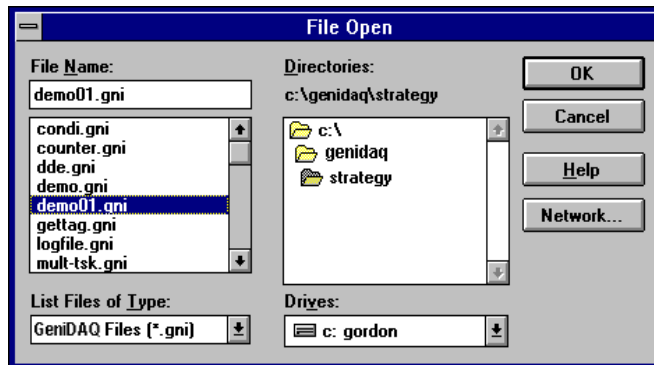
Figure 2-1   Opening strategy files

DEMO01.GNI is a simple strategy that configures the DEMO I/O Device analog input channel 0 to a trend graph display item. Double click on the AI icon and you will see a box displaying information about the Analog Input Block. Click on the arrow to the right of the box labeled **Device**, and you should see a list of all installed devices that provide the Analog Input function. Highlight the DEMO device. This is the device from which we will collect Analog Input information within this AI block. Choose channel 0 for sine wave and click OK to accept your choices.
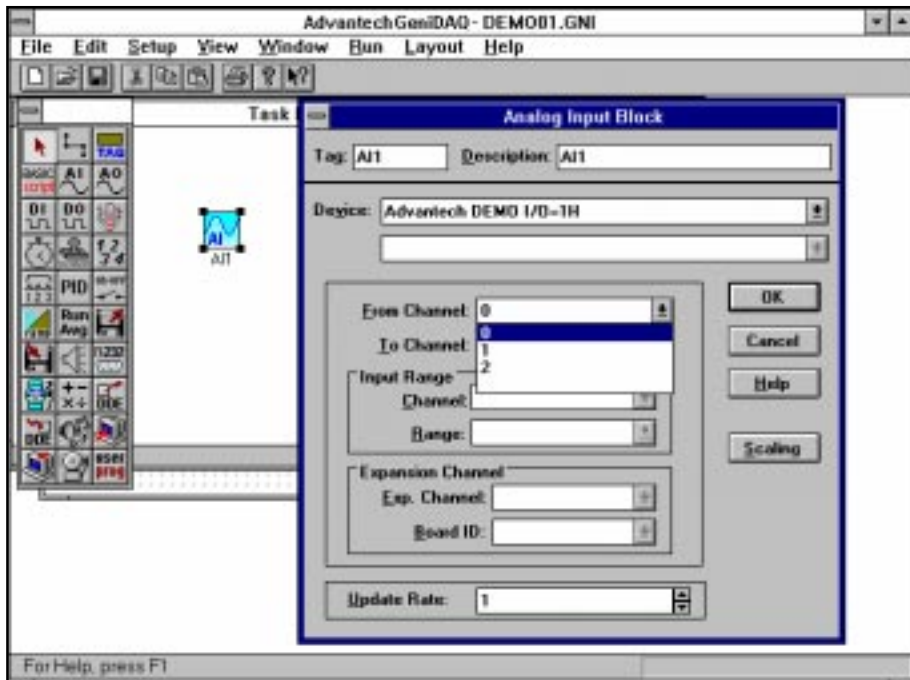
Figure 2-2   Analog Input block configuration

Double-click on the trend graph display icon in DISP1 within Display Designer. The Display Designer should open, along with a trend graph Display Item. This is one of many display types you can choose when you later design your display window.
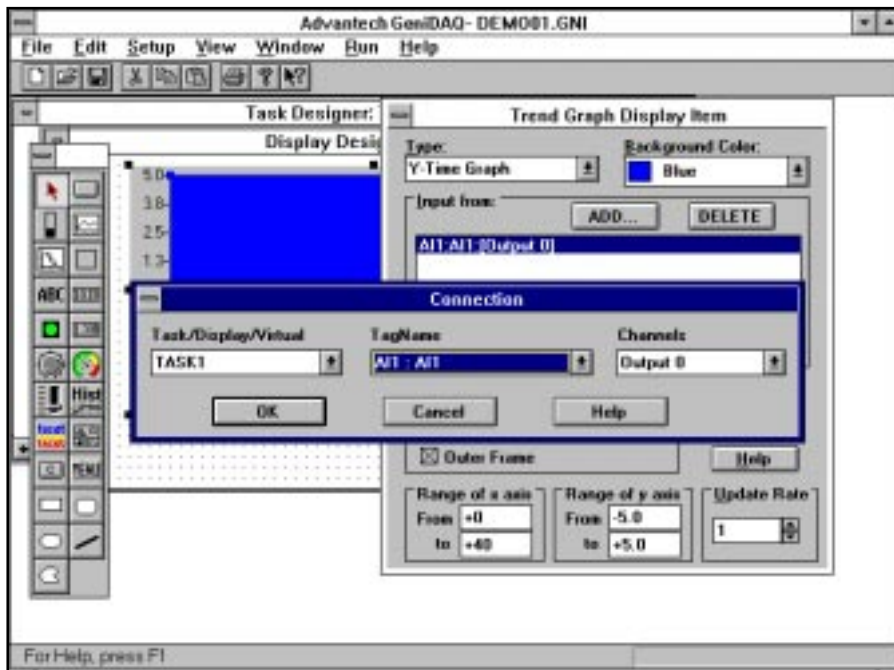
Figure 2-3   Task and Display connection

2. Run the DEMO Strategy. To run the Strategy from within GeniDAQ Builder, first save the Strategy (File, Save), and then press Run, located on the menu bar at the top of the Strategy Editor. The DEMO I/O Device's ± 5 volt sine wave will be generated in real time on the Y-T display . The sine wave continue to be generated until you press Stop on the Run Menu bar. You can also run from an icon called GeniDAQ Runtime.
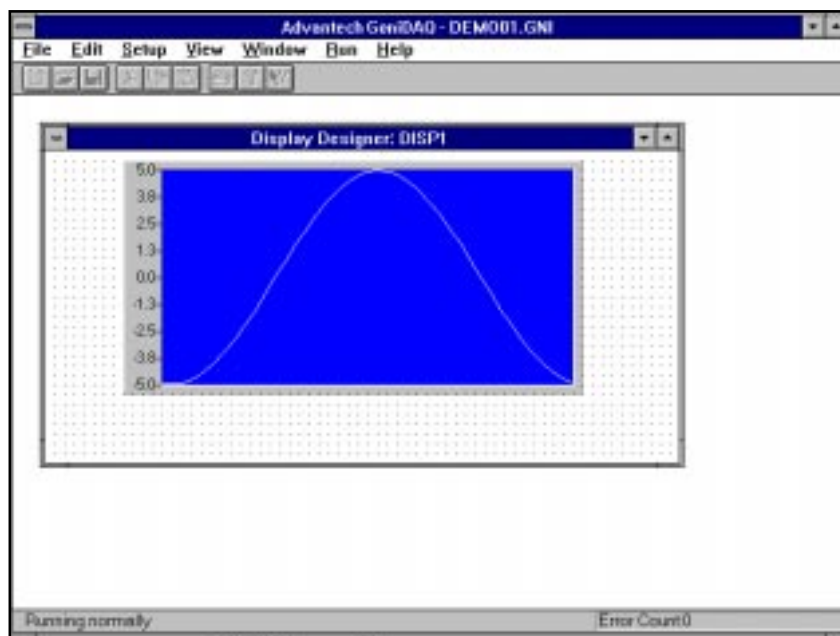
Figure 2-4   DEMO01 running

For a more detailed explanation of the GeniDAQ demonstration strategies, refer to *Chapter 3, Tutorial.*

Uninstalling GeniDAQ

If you wish to completely remove GeniDAQ from your system, you just click on the Uninstall GeniDAQ icon. GeniDAQ will be removed from your system automatically. You don't worry about deleting files.

2.3 Upgrade from GENIE 2.XX to GeniDAQ 3.1

Since the architecture of the program has been improved, we needed to modify the file format significantly to support these changes. This version cannot use the previous strategy file directly. Before the old strategy file can be loaded, the file must be converted. This is done by using a file conversion utility program called GWCONV.EXE in the GeniDAQ directory. This utility program will convert most of the data from the old format to the new format so that your old strategy file can be used by this version of GeniDAQ.

NOTE: *1) Some of the blocks that are no longer included in GeniDAQ cannot be translated from earlier versions and will be discarded (for example DISP block).*

    2) Some wiring between blocks may not be converted. In such instances, the user will have to set them up again.

    3) GENIE 2.x's C-like script is no longer supported, thus the old script cannot be converted. Since the GENIE 2.x scripts are stored in files that are separate from the strategy files, the conversion program will not recognize any old script program. In GeniDAQ 3.1, the BasicScript is stored in strategy files together with the tasks and display information. This has the advantage of easy installation and simplifies moving a strategy from one machine to another.

2.4 GeniDAQ Program Group Icons

Program Icons	Description
GeniDAQ Builder	The GeniDAQ icon allows you to invoke the GeniDAQ development environment to create new GeniDAQ application, select recently designed GeniDAQ applications, or scan drives for existing GeniDAQ strategy files. You may also modify the application directory list being displayed as well as create and/or edit the application.
GeniDAQ Runtime	The Runtime icon allows you to invoke only the GeniDAQ runtime environment without the development functions. You can execute GeniDAQ strategy files only without worrying about any changes being made. This utility is often used after you finished your system and run it at the end user site.
Device Installation	This icon is used for installing H/W drivers with GeniDAQ to connect equipment hardware. You can easily connect all Advantech hardware through supplied drivers.
BasicScript Help	The BasicScript icon is a help file to tell you how to use the BasicScript development environment. It includes a BasicScript function library and BasicScript program syntax.
User Guide Help	This icon invokes a help file which can be used to quickly access context-sensitive help for your operations.
Runtime Help	This icon invokes a help file which can be used to quickly access context-sensitive help for GeniDAQ Runtime.
Release Notes	This icon links to a text file introducing What's new in GeniDAQ.
Uninstall GeniDAQ	This icon is used for removing GeniDAQ from your system.

3

Tutorial

3 Tutorial

The intent of this chapter is to provide some general instructions and guidelines for performing basic GeniDAQ operations. Throughout this chapter it is assumed that you can understand the fundamentals of GeniDAQ. It is assumed that the GeniDAQ system software has been successfully installed. If the GeniDAQ software has not been installed, install it now. The installation procedure is provided in *Chapter 2: Preparing to install GeniDAQ*.

3.1 Working with GeniDAQ

The sections that follow describe how to use some of the basic functions in the GeniDAQ Task Designer, Display Designer, Report Designer, and Script Designer. This chapter does not provide detailed instructions for all GeniDAQ functions. Specific instructions for the various GeniDAQ functions are provided in the following chapters.

Before You Begin

You use a mouse to construct strategies and operator display panels within the Task and Display Designers. The ability to work with a mouse while developing your process strategies and corresponding displays helps to make using GeniDAQ intuitive.

Mouse support has been extended to the Runtime system. This allows operators to use a mouse or other pointing device, such as a trackball, touchscreen, etc., to interface with the process. The following terms are used throughout the manual.

Pointing
Positioning the cursor on an icon block, object, field, etc., on the screen by moving the mouse. The mouse is often referred to as the pointing device since it allows you to indicate graphically where you want to work.

Clicking or Double-Clicking
Positioning the cursor on an icon block, object, field, filename, etc., then pressing and releasing the left mouse button quickly. Double-clicking is to press the left mouse button twice in quick succession. This simultaneously selects and performs a field-associated operation.

Selecting
Pointing to something on the screen and clicking on it. Selecting a Block in the Task Designer or Display Item in the Display Designer causes square selection dots to form around the perimeter of the selected object. The dots remain (object selected) until another operation is performed.

Dragging
Holding the left mouse button down while you move the mouse, then finally releasing the button. As you move the mouse across your desk, the respective item moves across the screen and stops when you release the mouse button.

Planning Your Strategy

Before you begin using GeniDAQ to construct your strategy and corresponding operator display panels you should outline exactly what you expect to accomplish with the strategy. Make a sketch detailing the flow of data, timing, and so forth. This may save you valuable re-work time later in your project.

Starting GeniDAQ

After the GeniDAQ software and the desired I/O Device Drivers have been successfully installed, you are ready to start using GeniDAQ. The first step in using GeniDAQ is to develope a strategy from within GeniDAQ Task Designer.

To begin working with GeniDAQ, you must first boot with Windows 95/98 or Windows 3.1, enter the Advantech GeniDAQ program group and double-click on the GeniDAQ Builder icon. The GeniDAQ software loads, and you should see a pop-up window to introduce GeniDAQ and its version. After several seconds, the pop-up window disappears and a blank GeniDAQ main window opens.
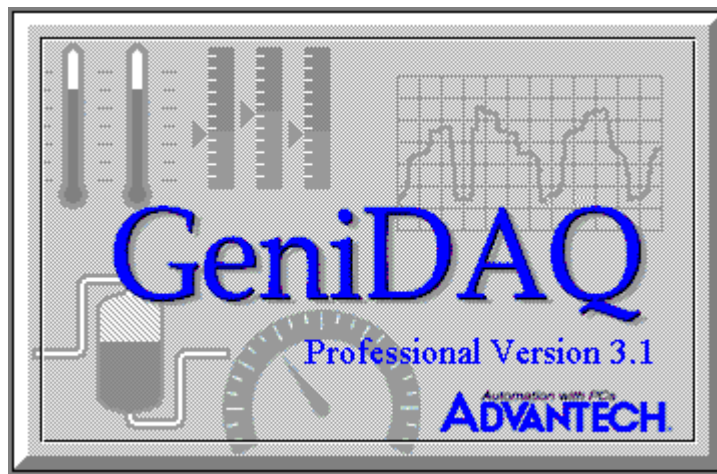
Figure 3-1   GeniDAQ information

Using On-Line Help

You can get Help while using GeniDAQ by choosing a command from the Task Designer Help menu or by pressing F1. GeniDAQ also includes a Help button in each dialog box, so that you may access context-sensitive help on a specific topic.

To access Help from GeniDAQ:

1. From the Help menu in the Task Designer, choose a Help command.

2. Or press F1 while using GeniDAQ.

3. Or choose the Help button in a dialog box.

A Help window appears. The topic that is displayed depends on which Help command you chose, what was selected when you pressed F1, or which dialog box you were using when you chose the Help button. The Help Contents for GeniDAQ appear when you enter Help using the menu item labeled Index. Within GeniDAQ, a Help topic on the selected command or dialog box appears when you use the Help Button within the current dialog box.

Building Your Strategy

Before you can use GeniDAQ to run a process strategy interactively with its corresponding operator displays in real time, you must first develop a strategy and design the corresponding operator panel(s). GeniDAQ Task Designer and Display Designer are used to accomplish these tasks.

After entering the GeniDAQ program, a new strategy file can be designed (invoked first because the process strategy is the basis of your process control/monitoring). Select New form the File menu bar in the Advantech GeniDAQ Main window. You will see a Task Designer window entitled "Task Designer - TASK1" and a Display Designer window entitled "Display Designer - DISP1" on your screen.
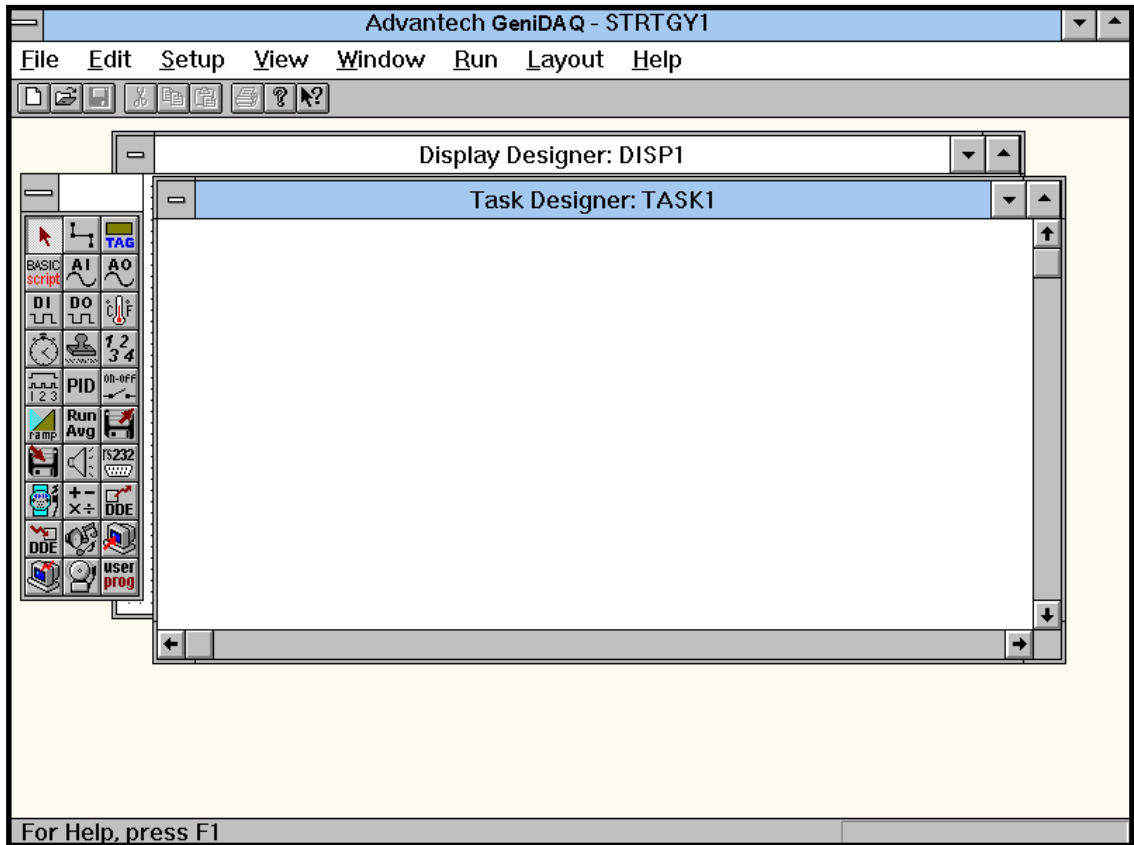
Figure 3-2   New strategy file

You start GeniDAQ by double clicking the GeniDAQ Builder icon in the GeniDAQ program group or executing GENIE.EXE in the command line. A blank window will be displayed within the GeniDAQ main window. This is the beginning of developing a strategy. If you click symbol in the Main window, Task Designer and Display Designer will be activated and appear in the upper part of the Main window. You can activate Script Designer by clicking on File, Add/Delete, Add Main Script option of the Main window. You also can activate Report Designer by clicking on Setup menu and Report... options of the Main window to design your operation report and its format.

3.2 GeniDAQ Tutorials

In order to help you become familiar with GeniDAQ quickly, we provide nine tutorial programs located in the GeniDAQ\Strategy directory. All example strategies use the Advantech DEMO I/O=1H device that is included with GeniDAQ, allowing you to run the demonstration programs without additional hardware.

The following is a step-by-step tutorial on how to use the demonstration strategies, as well as how to design strategies within GeniDAQ. It is designed with the beginner in mind. If you follow the tutorial, you should not encounter problems. The first demonstration, "TUTOR1.GNI", will be fully explained. Following examples will be more concise and use the understanding gained in the first demonstration as building blocks.

3.2.1 Tutorial 1: One Task with Display

Purpose

The purpose of this tutorial is to demonstrate how to work with GeniDAQ. In this tutorial, we will show you the easy way to use GeniDAQ to acquire data and display it with numeric display and trend graphs in the display window. This tutorial introduces GeniDAQ.

Function

Use Advantech DEMO I/O device AI block to acquire simulated I/O data and show the data by numeric value and trend chart in the display window.

Procedure

1. Invoke GeniDAQ by double clicking on the GeniDAQ Builder icon in the GeniDAQ workgroup. After double clicking, a blank window will be displayed on your screen. Click once on the File menu to pull down the menu, and choose New. After creating a new strategy file, two blank windows will pop up. One is TASK1 of Task Designer and another is DISP1 of Display Designer. The new TASK1 window will appear on the front of your screen and TASK block toolbox will be displayed on the left of TASK1 window. The window should look like Figure 3-3.

2. Choose the AI block from the Task block Toolbox and click on it. AI block will be chosen. Move the mouse to the working area of the TASK1 window and press the left mouse button. An AI block will be added to the working area as AI1. This operation is called Drag and Drop
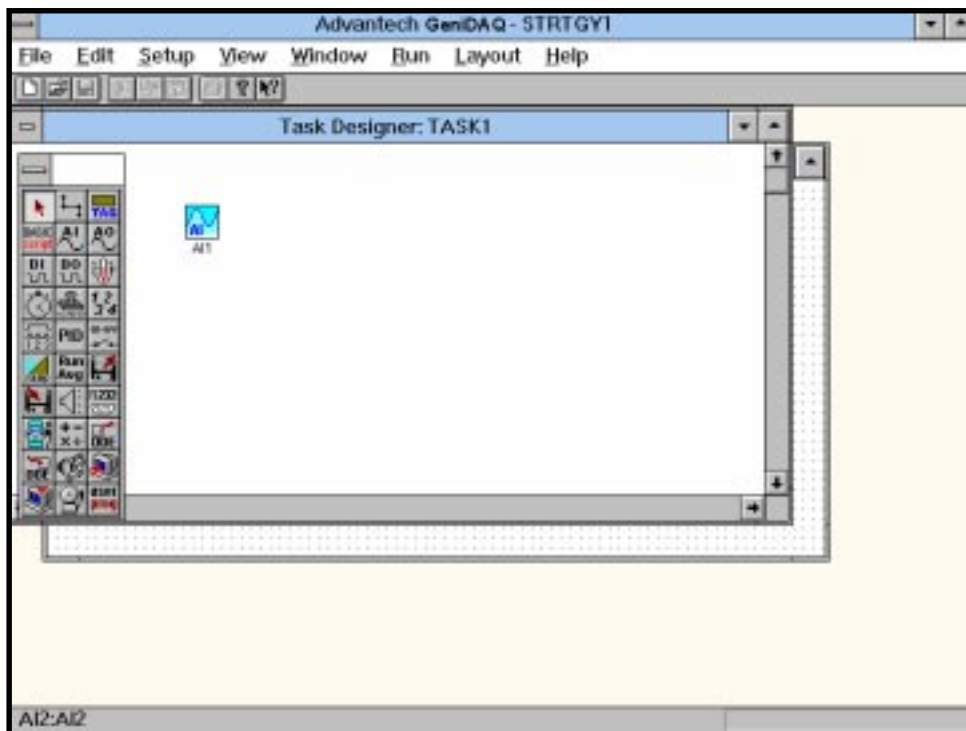
Figure 3-3   Add an AI block

3. Double click on the AI1 block. You will see a dialog box like that shown below:
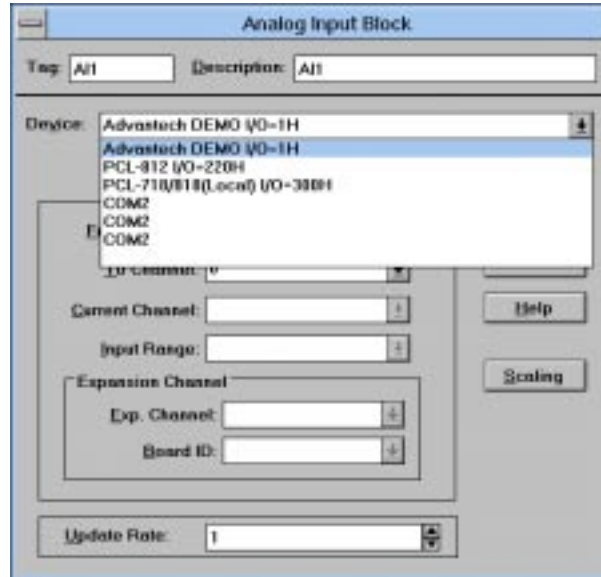
Figure 3-4   Configure AI block I/O device

4. In the AI configuration dialog, if the Device: box is empty, click on the downward pointing arrow at the end of the box and click once on Advantech DEMO I/O:0H. This will configure this analog input block to use the data from the Advantech DEMO I/O device, which has an I/O address of 0H (0 hexadecimal). The channel should be set to 0 to view a sine wave. Now that the AI1 block is configured, click once on the OK button, which will save the configuration of the block and close the dialog box, returning you to the task window.

5. After creating an AI block in Task Designer "TASK1", you can change the activated window to Display Designer "DISP1" by clicking once on the Window menu and pulling the menu down. Select "2 Display Designer : DISP1" item to activate Display Designer "DISP1". A Display item Toolbox will be displayed on the left of the Display1 window, like the following:
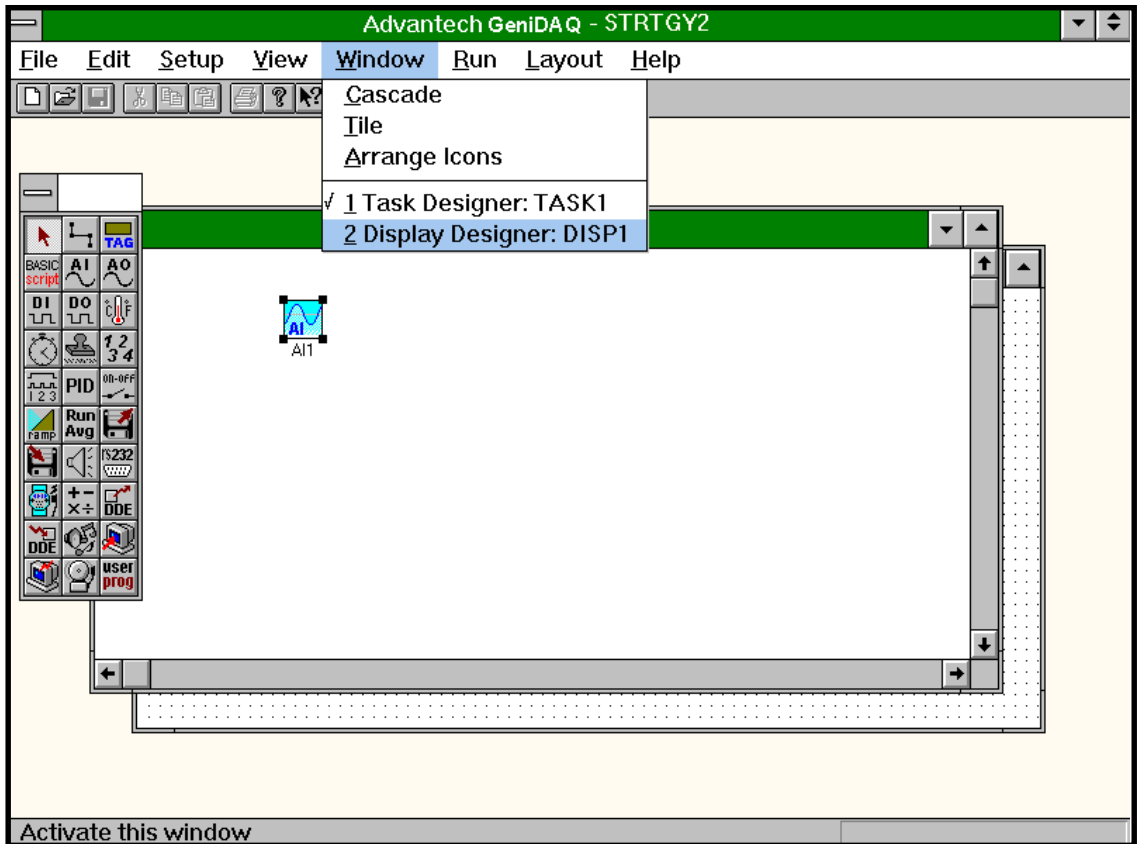
Figure 3-5   Switch Task and Display

6. Choose the numerical display item from the display item toolbox. The numerical display item will be in depressed mode. Move your mouse to the working area of DISP1 window and press the left button of the mouse. A numerical display item is added to the Display1 window, as shown below:
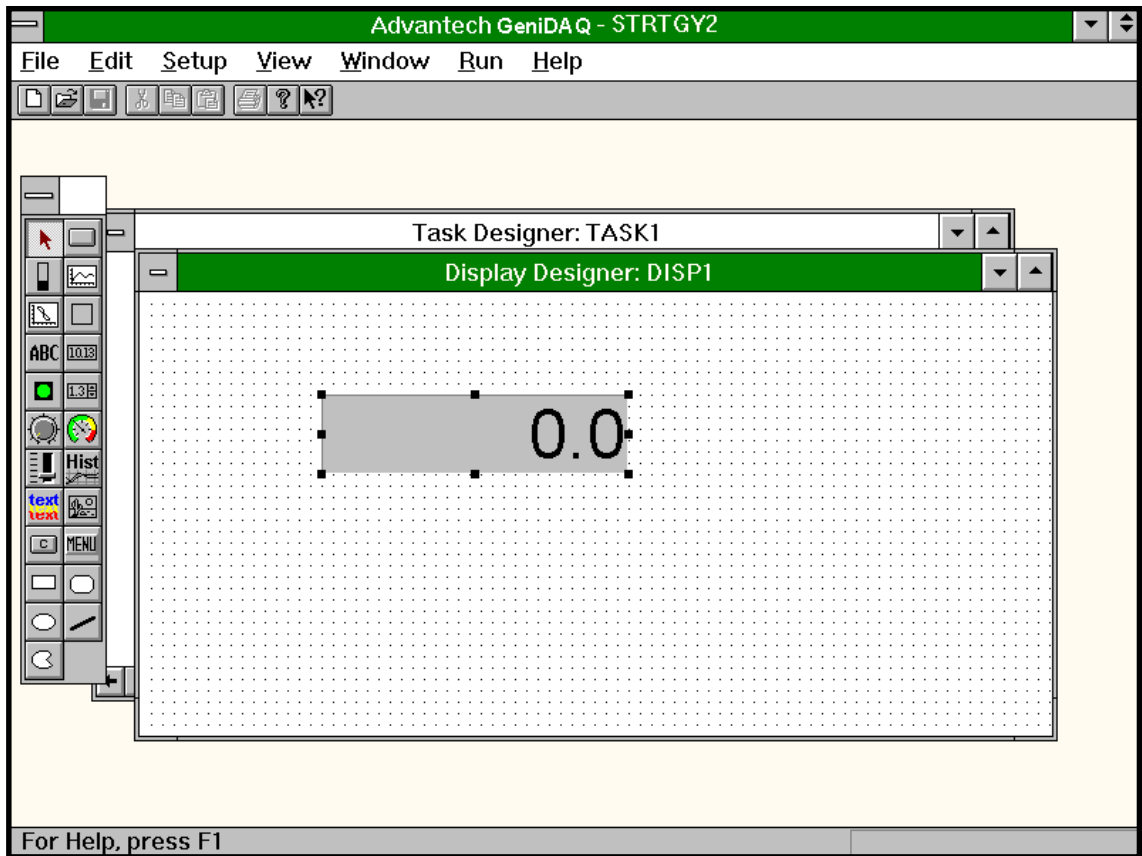
Figure 3-6   Add a numerical display item

7. Double click on the numerical display item. You will see a dialog box appear like the following:
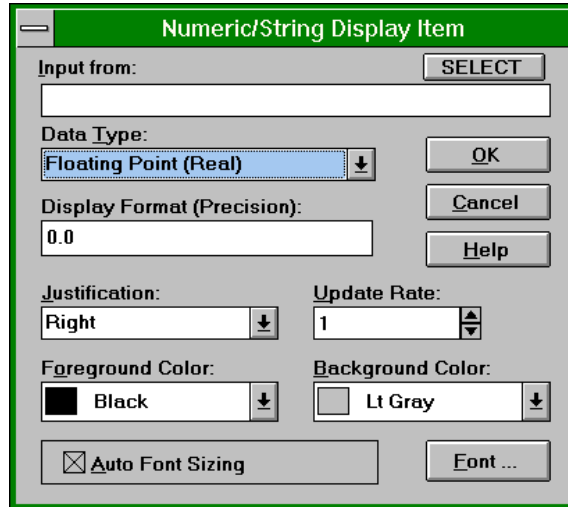
Figure 3-7   Configure a numeric display item

If the Input from box is empty, click on the Select button once. A pop-up dialog panel will appear to configure the data input from, like the following:

Figure 3-8   Connect Task and Display

In this pop-up dialog window, you have to choose TASK1 from the Task/Display field and choose AI1 from the TAG field. The channel field will display the default channel number: Output 0. You can select another channel number from this field. We use Output 0 in this example.

8. Now that the numerical display block is configured, click once on the OK button at the dialog window to save the configuration.

9. Do the same way as step 6 to add a Trend graph display item at the Display1 window and configure it to link with the same task block (AI1) and the same channel.
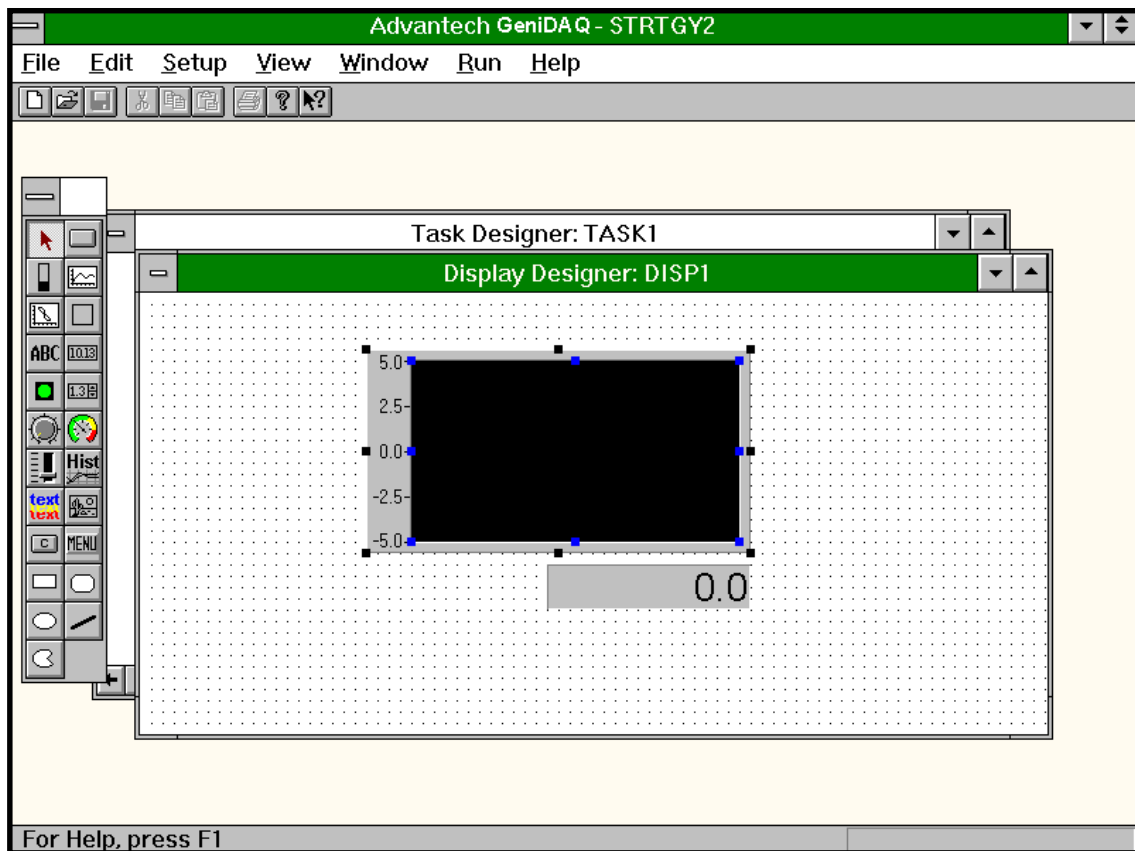
Figure 3-9   Add a trend graph item

9. Click once on the File menu and choose the Save option from the submenu. A window will pop up to save your strategy file. The saving operation is the same as any other Windows file saving operation. Enter the filename tutor1.gni, choose the desired directory, and press the OK button. The strategy file will be saved to disk.

Figure 3-10   Configure a trend graph item
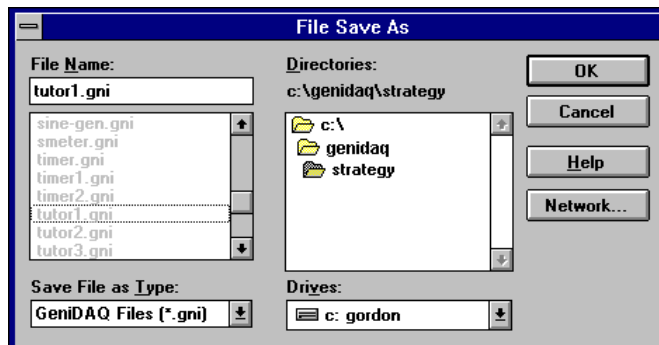
Figure 3-11   Save a strategy file

After saving your strategy file, you can run this strategy file immediately. Click on the Run menu and choose the Task option on the drop down menu. The strategy file will be executed and Numeric Display item and Trend Graph Display Item will immediately show the current value in the Display window, like the following:
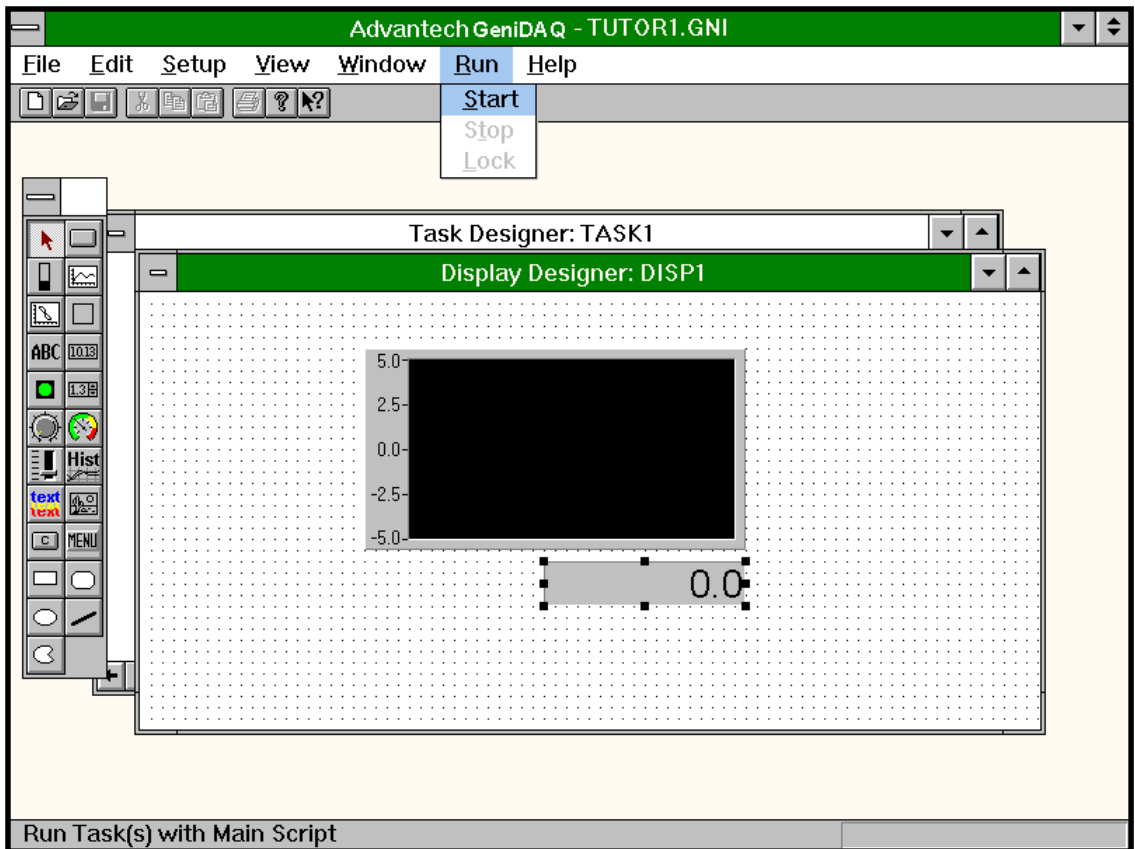
Figure 3-12   Start to run strategy file
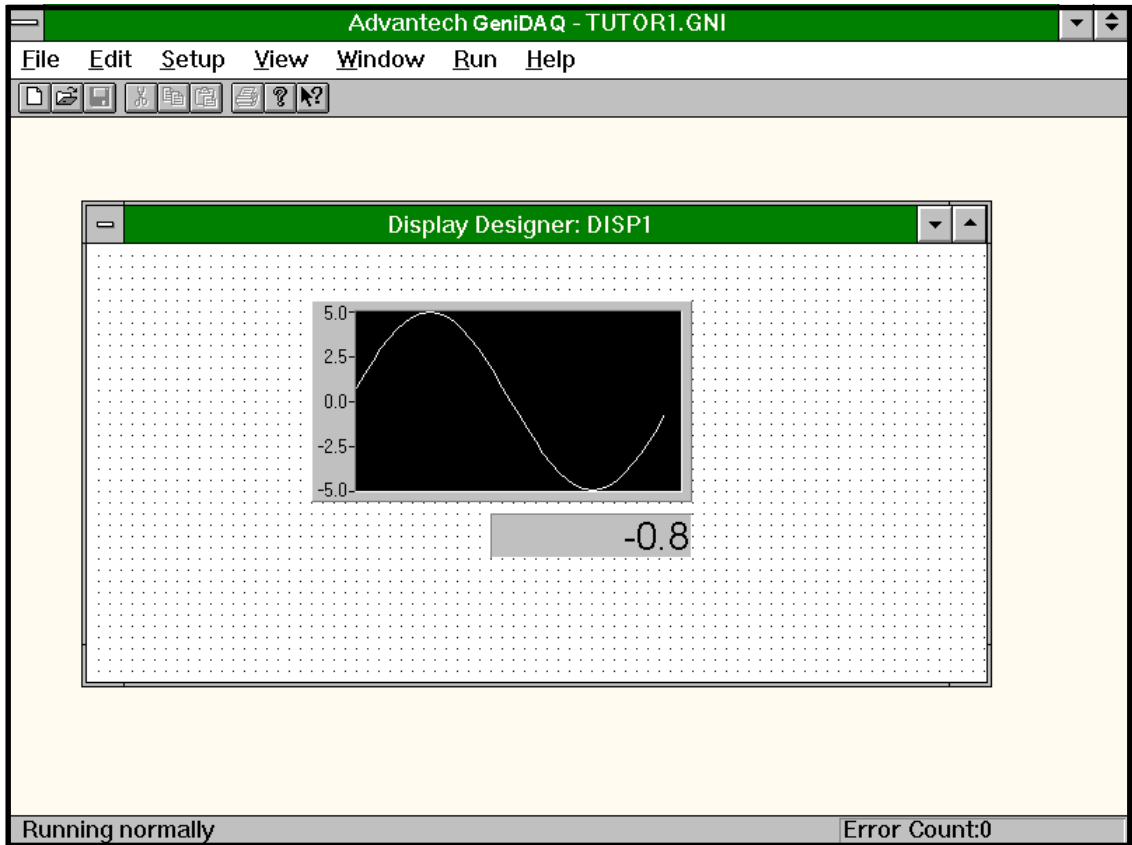
Figure 3-13   Strategy file execution results

3.2.2 Tutorial 2: Multiple Displays and switching

Purpose

The purpose of this tutorial is to teach you how to design multiple display windows in GeniDAQ and switch the display window while running.

Function

Use Advantech DEMO I/O device AI block to acquire simulated I/O data and show the data by numeric value and trend chart in different display windows. Add a menu button on each display window for switching between windows.

Procedure:

1. Refer to Tutorial 1 Step 1.
2. Refer to Tutorial 1 Step 2 to Step 8 to create an AI block at Task1, a Numeric Display item at DISP1 and a trend graph display item at DISP2.
3. Click once on the File menu and choose Add Display from the Add/Delete submenu. "DISP2" display window will be added at the front of the screen. Create a trend graph display item at DISP2.
4. Choose the Menu Button Control item from display item toolbox at DISP1. Drag the item to DISP1 working area and click the left mouse button to drop it in DISP1.
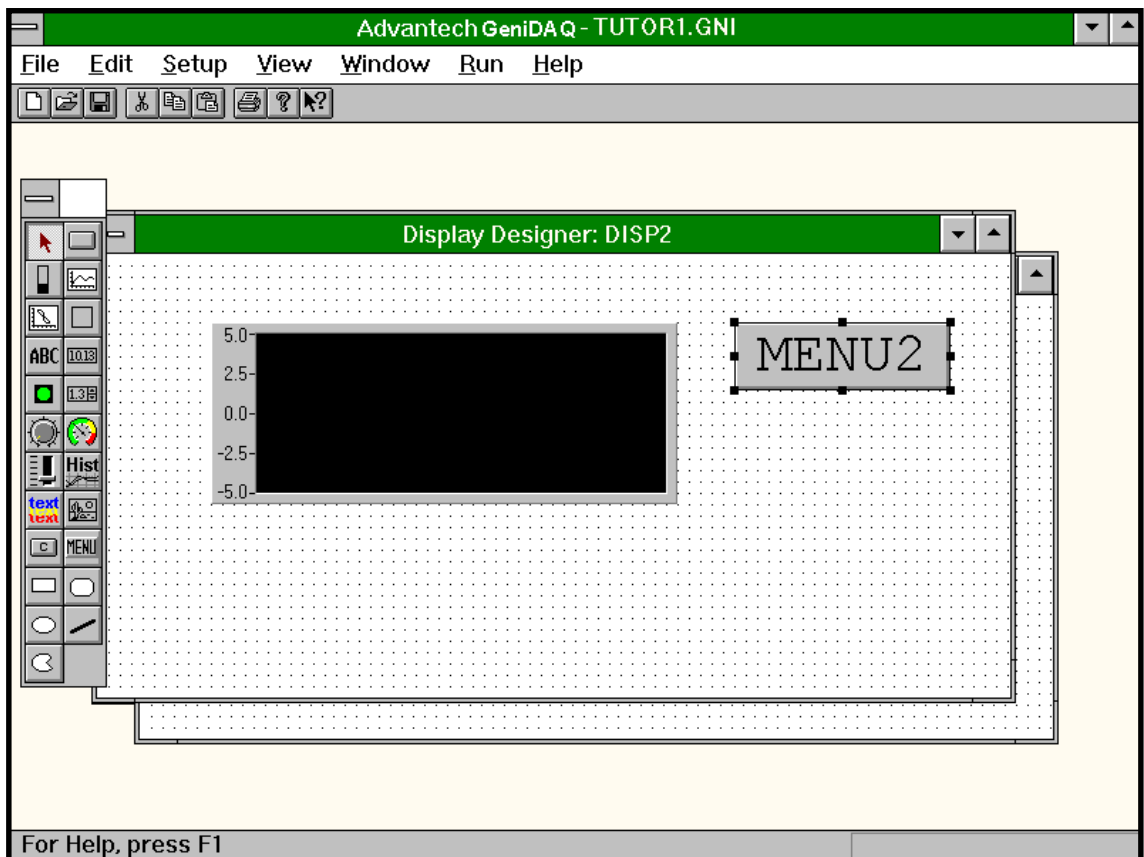
Figure 3-14   Add a display window

Follow the same procedures to add a Menu button control item to DISP2.

5. Double click on the menu button control item MENU1 at DISP1. A dialog box will appear to config-
   ure the Menu button. Set the function field in the dialog to be Display Switching and select the
   Display Switch field of the dialog to be DISP2. Save the configuration by clicking on the OK button.
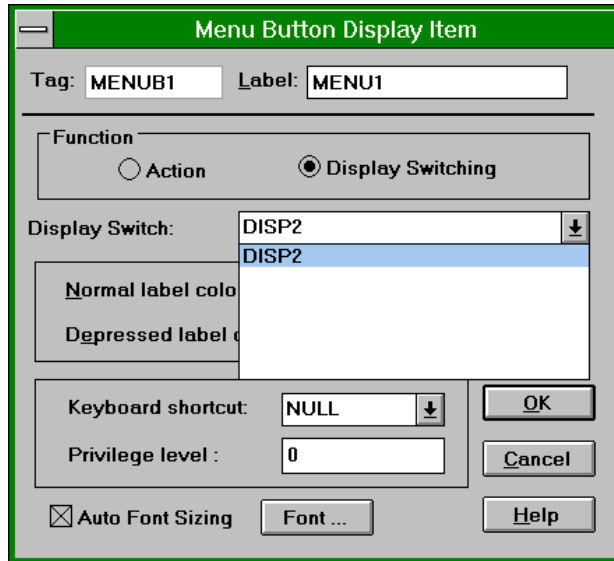
Figure 3-15   Configure display switch to DISP2

6. Follow the same procedures for DISP2, but change the Display Switch field of the dialog window to
   be DISP1.
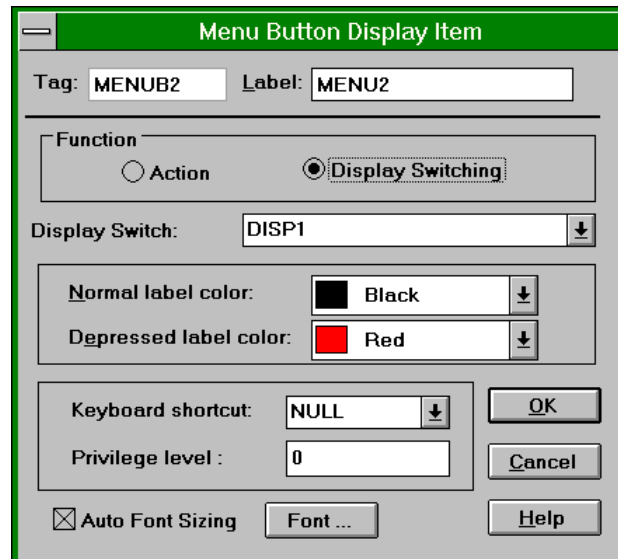
Figure 3-16   Configure display switch to DISP1

7. Save the strategy file as "TUTOR2.GNI"

8. Run the strategy file by selecting Task from the Run menu. The screen will first show DISP1 on the screen. DISP1 shows a numerical item value that is updated dynamically. Press the menu button on the DISP1 window and the display window will be changed to be DISP2 window. You will see a trend graph chart drawing a sine wave. Press the menu button on the DISP2 window, and the screen will return to the DISP1 window.
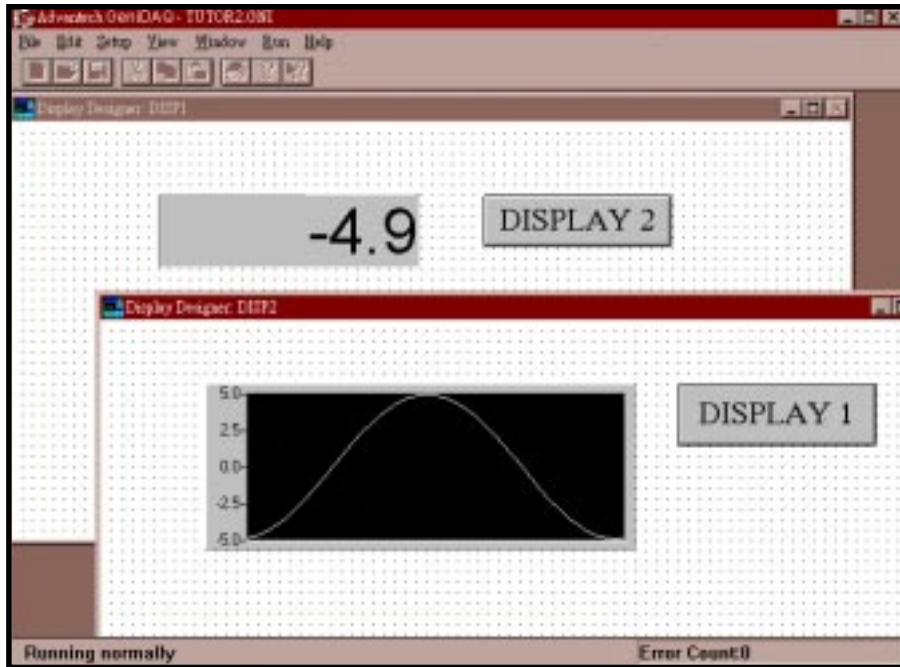
Figure 3-17   Change between screens

3.2.3 Tutorial 3: Execution Order Arrangement

Purpose

The purpose of this tutorial is to teach you how to design multiple tasks in a strategy file and how to arrange the execution order of task blocks.

Function

Create two AI block and one single operator block in a task and execute a symbol, addsymbol operation in single operator to add two AI blocks and show their result. Repeat the above operations with another task and show the result on the same display. Rearrange the execution order to see the impact.

Procedure

1. Refer to Tutorial 1 to start GeniDAQ and add a new Task.

2. Add two AI blocks and a single operator calculation block Add into Task1. Configure AI1 block to be the input from Advantech DEMO I/O channel 0 and AI2 block to be the input from the same device and channel.
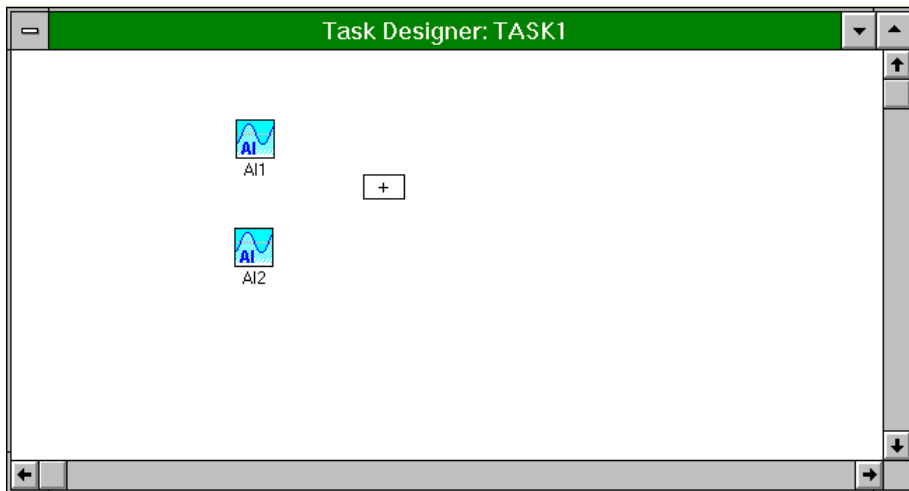
Figure 3-18   Add two AI blocks and a single operator block

3. Click on the Wire block in the task block toolbox. The mouse will change to a wiring icon. Click once on the AI1 block to be the source of a link connection and move the mouse to the single operator block and click once again. A wire connection will be established between AI1 and the single operator block. A pop-up panel will be displayed to choose the output channel number of AI1. Select the channel 0 to be the output channel, refer to Fig 3-20. Another pop-up panel will be displayed to choose the operand of operator Add. Select the operand 1. Refer to Fig 3-21.
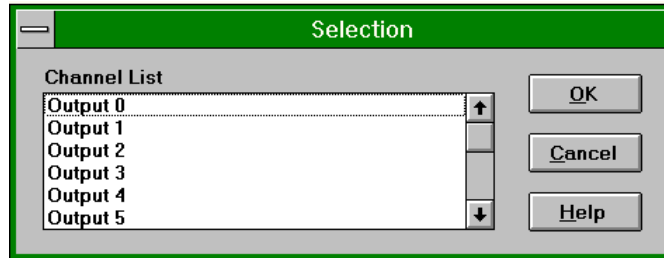
Figure 3-19   Select output channel of AI block
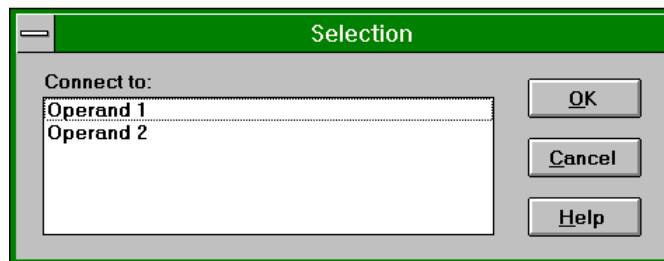
Figure 3-20   Select the operand of single operator

Do the same thing as above for AI2 block to link with single operator block. Select channel 0 and operand 2 as the output channel and the operand of operator Add.
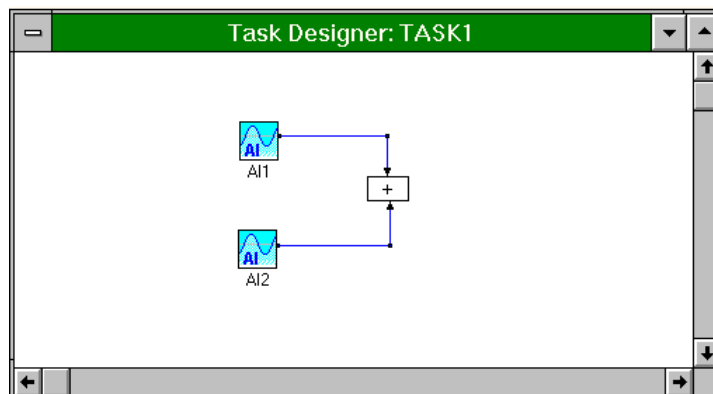
Figure 3-21   Wire two AI blocks to a single operand block

4. Repeat Step 2 and Step 3 to add a single operator calculation block Subtract.
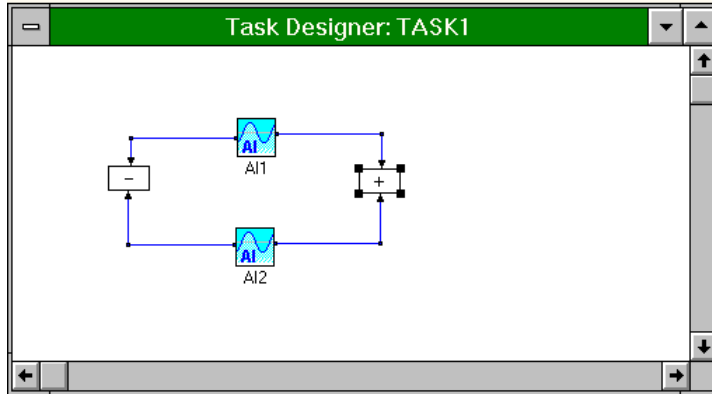
Figure 3-22   Wire two AI blocks to two single operands

5. Click on the View menu and activate Order Layout option to show the execution order number at the upper half of each block.
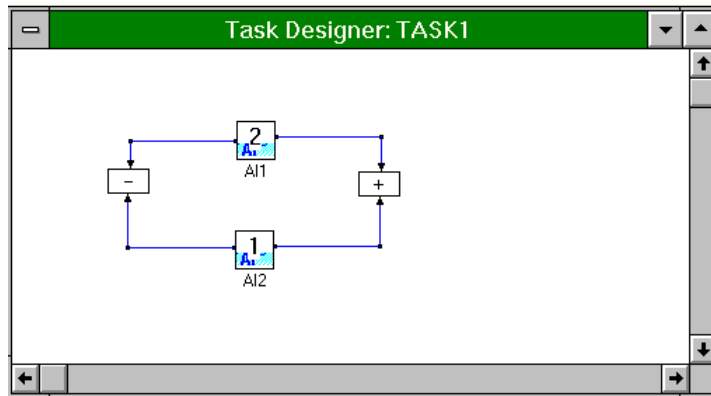
Figure 3-23   View the execution order

6. To arrange the execution order, click on the Layout menu. From the Layout submenu select the Complete reorder menu option. You can change the order of task blocks immediately. Move the mouse to the AI1 block and click once, the order number of AI1 will be 1. Move the mouse to the AI2 and click once, the order number of AI2 will be 2. Move the mouse to single operator calculation block Add and click once. The single operator block will show order number 3 at the upper half of icon. The single operator calculation block Subtract will be set as number 4.
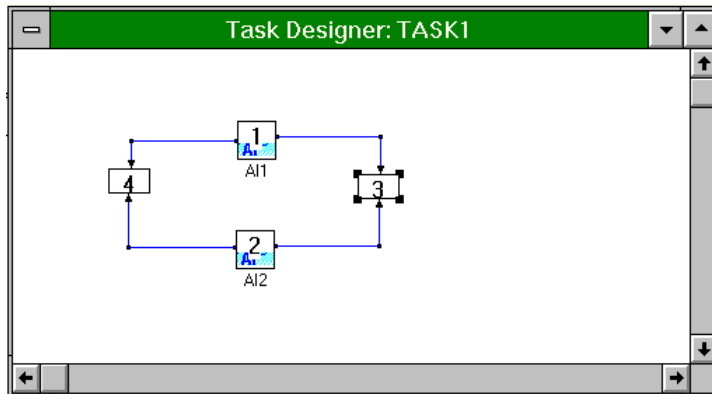
Figure 3-24   Arrange the execution order

7. If you want to exchange the order between single operator calculation block Subtract and single operator calculation block Add, you have to click on the Layout menu and activate the Exchange order option. After activating Exchange order function, move the mouse to the desired two blocks and click on them. The order numbers will be changed automatically.
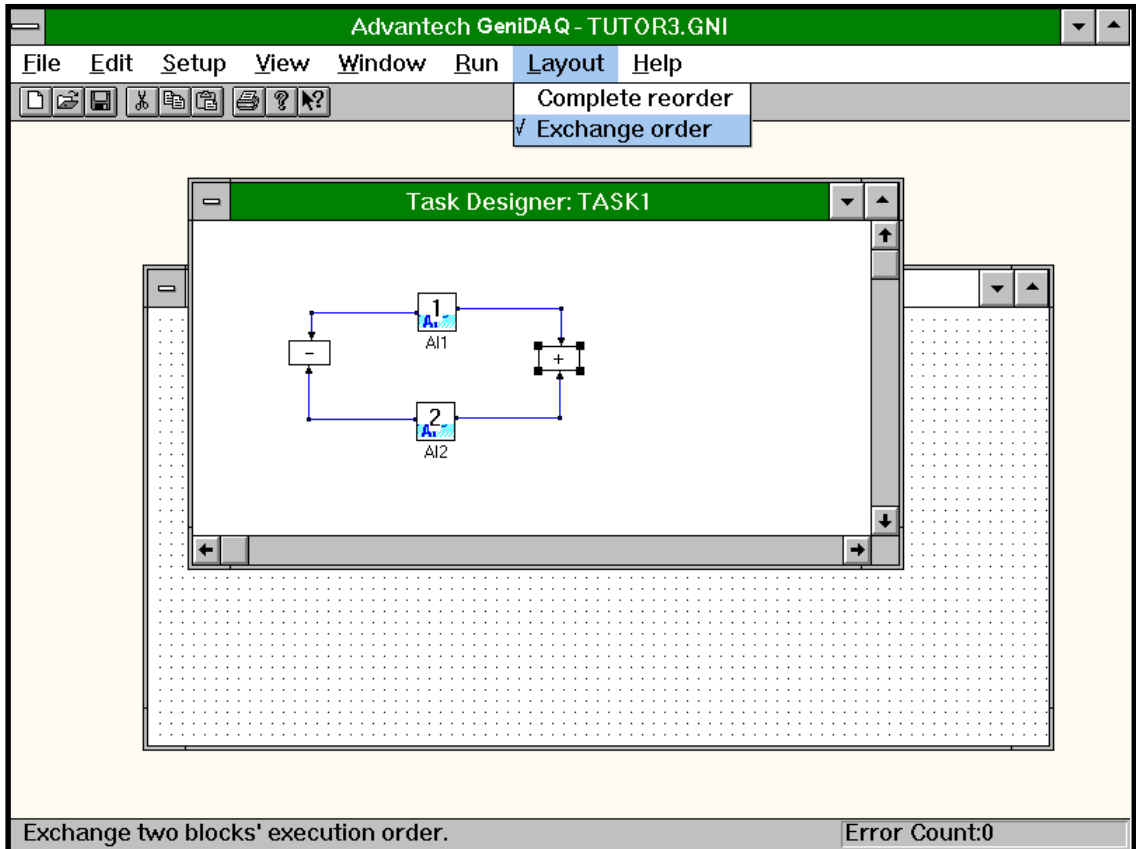
Figure 3-25   Exchange execution order

8. Refer to Tutorial 1 Step 9 to save the strategy file as "TUTOR3.GNI".

9. Run the strategy file to see the impact of different ordering.

3.2.4 Tutorial 4: Drawing tool in Display Designer

Purpose

The purpose of this tutorial is to teach you how to use the drawing tools in display item toolbox to customize you display.

Function

Use oval and rectangle drawing blocks to draw a pump and combine them into a pump object using the Make Object option of the Edit menu. After that, link a binary control display item to the combined object and use the control item to assign different values to the pump object and change its color.

Procedure

1.   Refer Tutorial 1 to start GeniDAQ and add new Tasks.

2. Add a binary button control item at the Display1. Configure this control button to be an ON/OFF control and output a 0 or 1 data value.
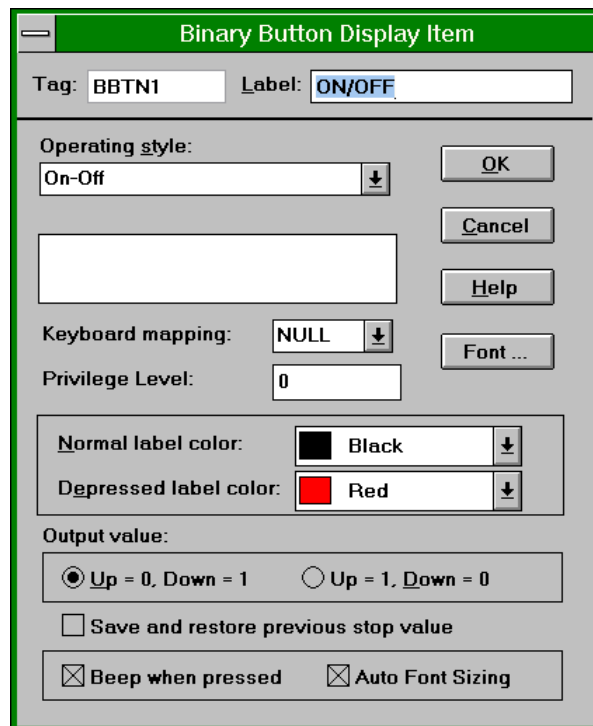
Figure 3-26   Add a binary button control item

3. Click once on the oval drawing item in the display item toolbox. Drag the oval drawing item to the display working area. Choose the added oval item and press "Ctrl+C" to copy it. Press "Ctrl+V" to paste it to the same display working area. Select one oval drawing item and move your mouse to one corner of the rectangle and enlarge the drawing item.
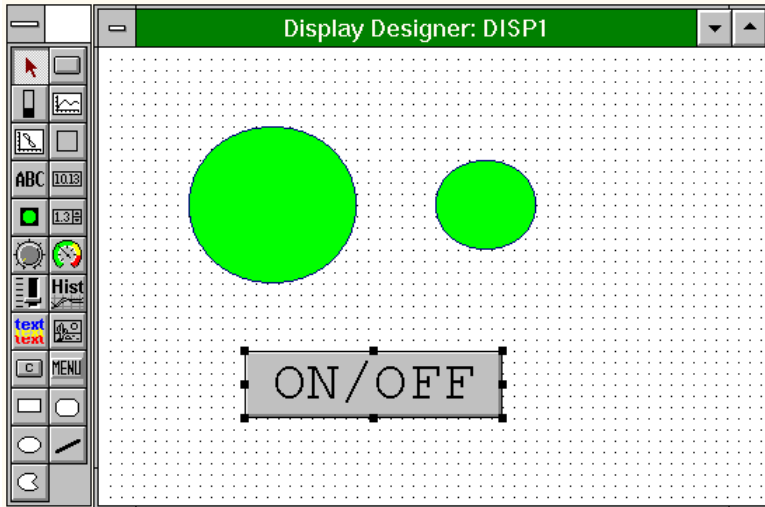
Figure 3-27   Add a binary button control item

4. Add two rectangle drawing items on the same display working area by drag and drop or copy and paste operations.
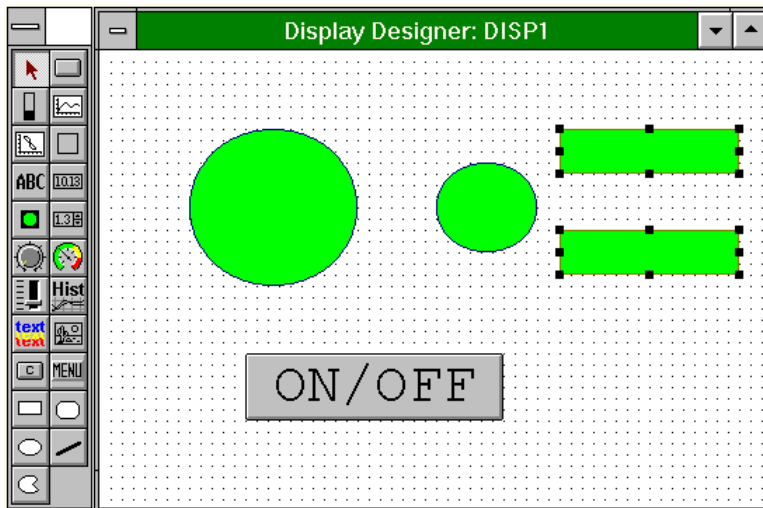
Figure 3-28   Add two rectangle drawing items

5. Configure the drawing items by double clicking on each one. A pop-up dialog will be displayed for configuring the input.
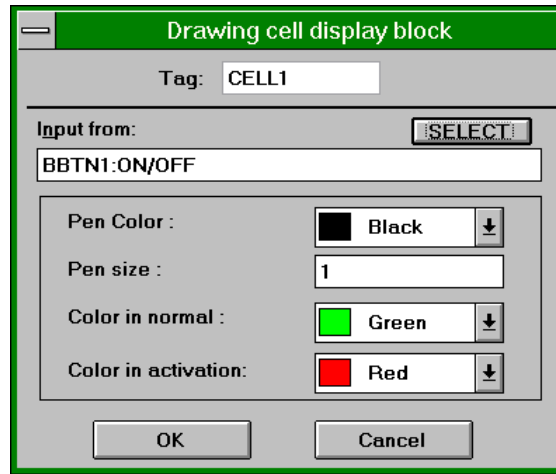
Figure 3-29   Configure Drawing item

Select Display1 from Task/Display field and BBTN1:ON/OFF from TAG field in configuration dialog. Set the color to be red while the input is equal to 1 and Set the color to be green while the input is equal to 0. Press OK to save the configuration.

6. Arrange these drawing items to be a pump picture, like the following. You can send the rectangle objects to the back of oval objects by clicking on the Edit menu and Send to Back option. After that, select all the parts you wants to build into an object, and click on the Edit menu and Make Object menu option in Display Designer to combine these four drawing items into one drawing item, like the following.
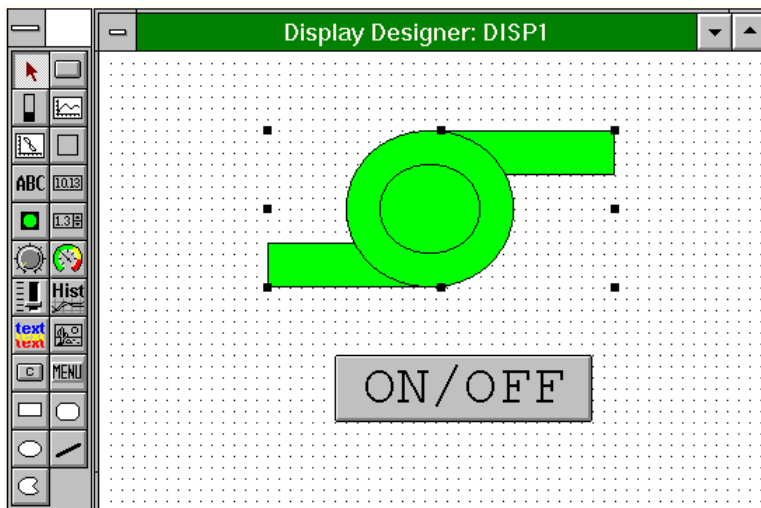
Figure 3-30   Make object

7. Save the strategy file to be "TUTOR4.GNI".

8. Run the strategy file. You will see the color of pump picture will be red while you press the binary control item. The color will be green while you press the binary control item again.
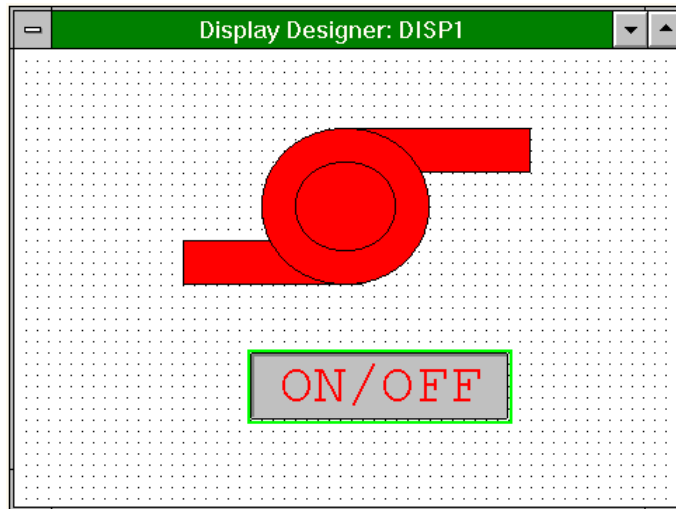
Figure 3-31   Run strategy file "TUTOR4.GNI"

9. If you want to modify the integrated drawing item, select the Break Object option from the Edit menu in Display Designer.

3.2.5 Tutorial 5: TAG block for integrating Task with Display

Purpose

The purpose of this tutorial is to teach you how to use TAG block in Task Designer to retrieve user input data in Display and process it in Task.

Function

Creates a numeric control item to input a control value in a display. Design TAG block to retrieve user input value and Alarm block to check if the input value is out of alarm range. If it is out of range, then a red lamp is displayed, otherwise a green lamp is displayed.

Procedure

1. Refer to Tutorial 1 to start GeniDAQ and add new Tasks and Displays.

2. Design a numeric control display item and a indicator display item in the Display1 window. Configure the numeric control display input type to be a floating number and configure the indicator display to be a round lamp shape. Set the color to be red while the input is equal to 1 and green while input is equal to 0.
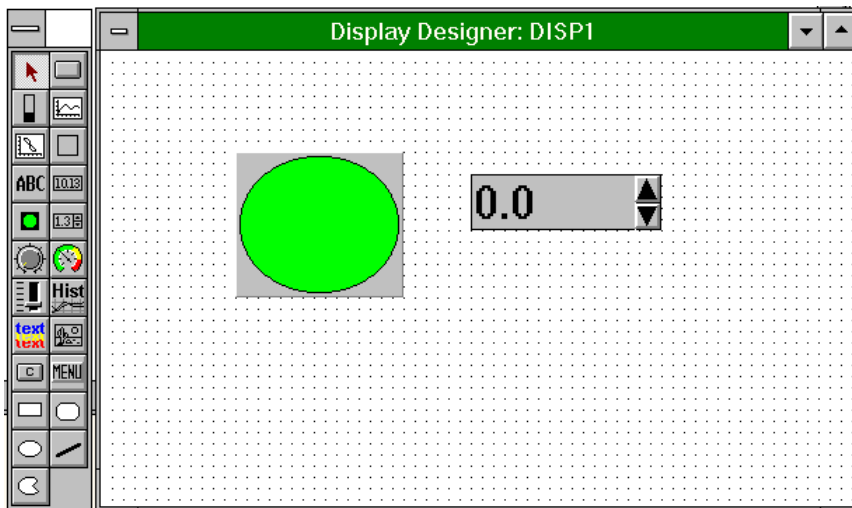
Figure 3-32   Add numeric control and indicator display item
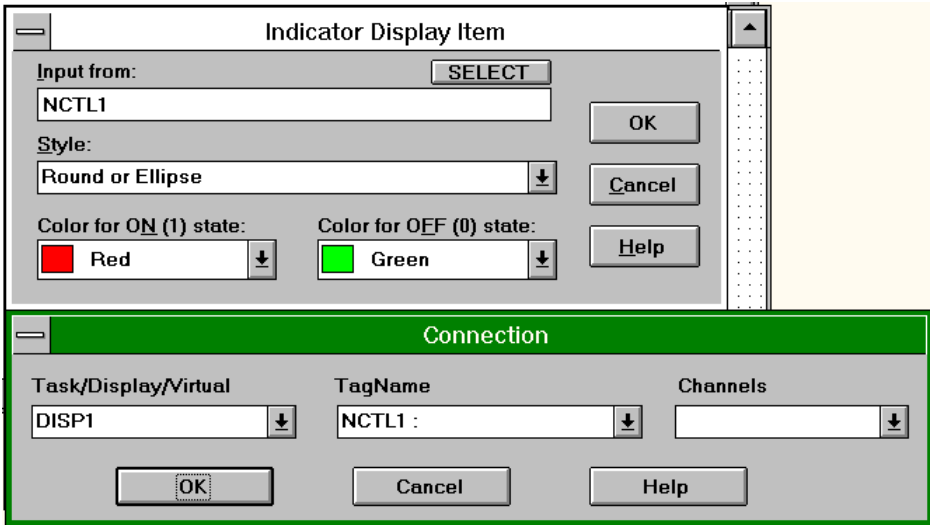
Figure 3-33   Configure indicator display and link to Task

3. Design an Alarm block and a TAG block in Task1.

Figure 3-34   Add alarm block and TAG block

Double click Alarm block to configure the value of hi-hi field to be 90, the value of Hi field to be 80, the value of lo field to be 20 and the value of lo-lo field to be 10.

Figure 3-35   Configure alarm block

Double click TAG block to configure TAG as a linkage to the numeric control item of Display1 by setting the Task/Display field to be Display1, the TAG field to be CTRL1 in configuration dialog.

Figure 3-36   Configure TAG to link with display item

4. Connect the TAG block to Alarm block in Task1 by wire link. Take TAG value as the input value for alarm block and compare the value with hi-hi, hi, lo, and lo-lo field of alarm block. If the TAG value is exceeded or beyond the set values of the alarm block, an alarm value will be output to the button display.
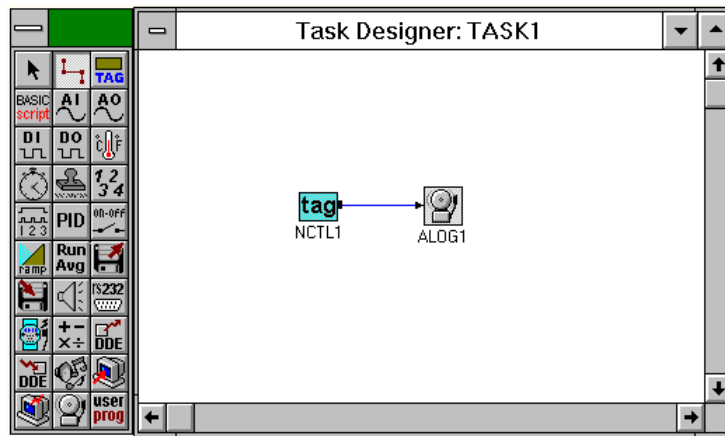
Figure 3-37   Connect TAG block to alarm block

5. Double click indicator display item in DISP1 to configure it and link to the output value of alarm block in Task1. You are advised to add two label display items in DISP1 for showing hi and lo alarm values. In this example, the label display items are "Upper Limit=80" and "Lower Limit=20".
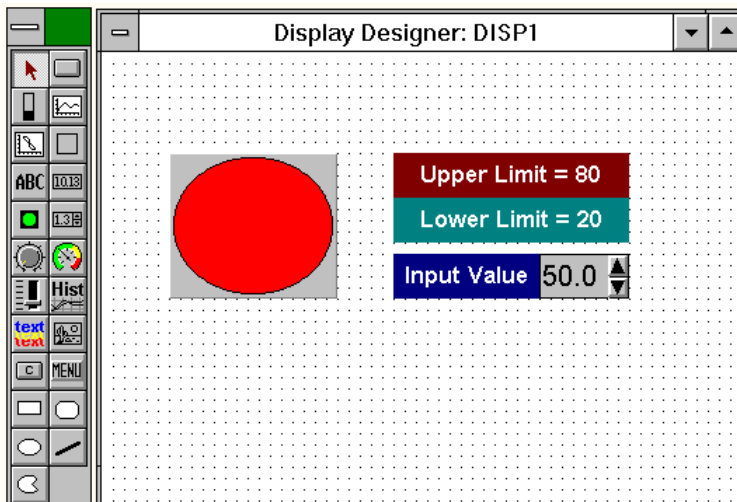
Figure 3-38   Run strategy file "TUTOR5.GNI"

6. Refer to Tutorial 1 Step 9 to save the strategy file as "TUTOR5.GNI".

7. Run the strategy file and type a value in the numeric control item. If you type 50 in the field, you will find the indicator display is green. However, if you type 81 or 19, the indicator display will be red.

3.2.6 Tutorial 6: BasicScript block

Purpose

The purpose of this tutorial is to give you a guide on how to program the BasicScript block.

Function

1. Get Advantech DEMO I/O AI channel 0 value

2. Compare the AI channel 0 value with the value the user typed in.

3. If AI value is greater than input value, then turn on the alarm lamp to be red, otherwise turn it off to be green.

Procedure

1.   Refer Tutorial 1 to start GeniDAQ and add new Tasks.

2. Add a numeric display item, a numeric control display item, a indicator display item and two text display items into Display1 window.
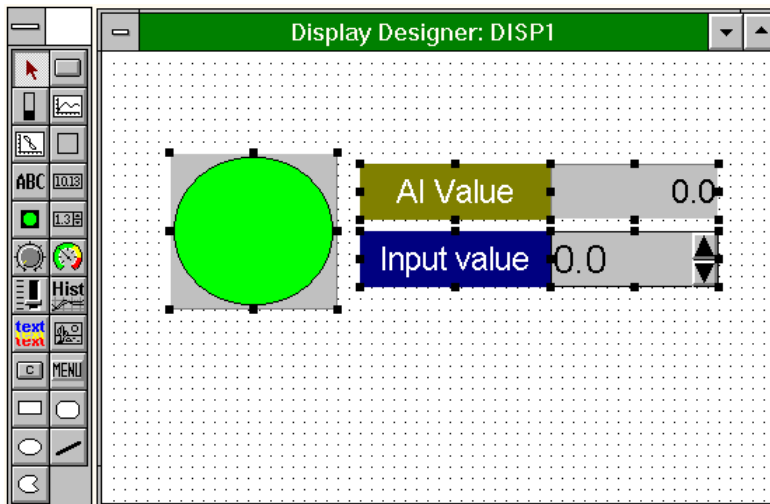
Figure 3-39   Build display screen

3. Add AI block, BasicScript block and TAG block in Task1. Configure AI block to be Advantech DEMO I/O channel 0. Connect AI1 block to BasicScript block and NCTL1 block to BasicScript block by wire link.
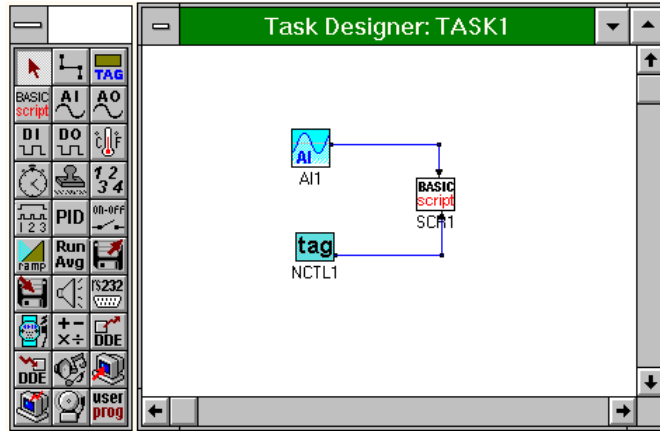
Figure 3-40   Build Task icon and wiring

Link TAG1 with numeric control item in Display1 window.
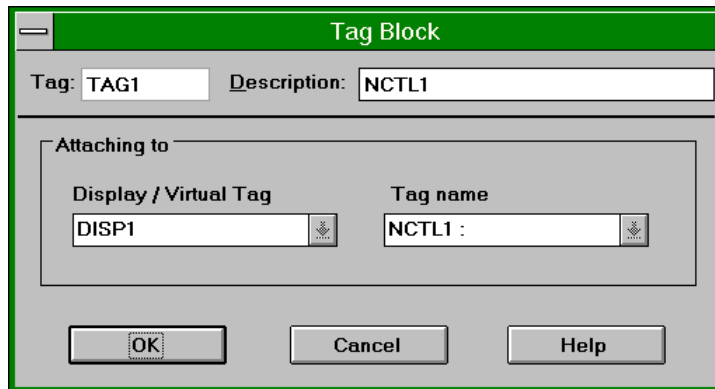
Figure 3-41   Link TAG to numeric control item

4. Double click the BasicScript block to edit a Basic script program. Key in the following text program in the pop-up window:

```
Sub SCR1()
      dim mytag1 as TAG
      dim mytag2 as TAG
      set mytag1 = GetTag("Task1", "AI1")
      set mytag2 = GetTag("Disp1", "NCTL1")

      if mytag1 > mytag2 then
         outputi 1
      else
         outputi 0
      end if
End Sub
```

5. Double click on the numeric display item in Display1 and configure it to display the value of AI1 in Task1. Double click on the indicator display item in display1 and configure it to display the value of scr1 in Task1.
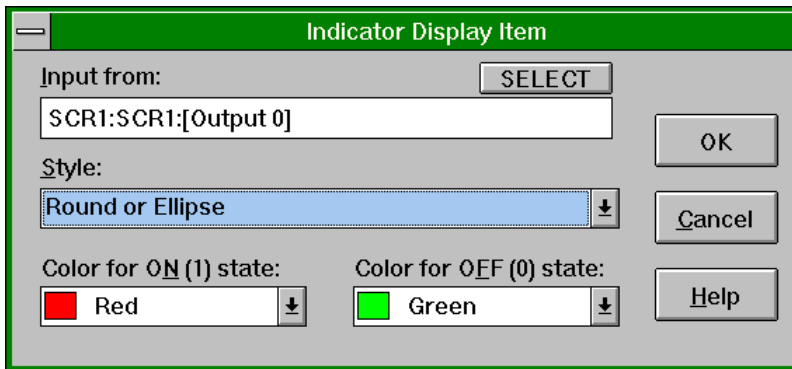
Figure 3-42   Link BasicScript output to indicator display

6. Refer to Tutorial 1 Step 9 to save the strategy file as "TUTOR6.GNI".

7. Run the strategy file. If you type 0 in the numeric control display item, the indicator display item will be red if the AI1 current value is positive, otherwise the indicator display item will be green.

3.2.7 Tutorial 7: BasicScript block with Virtual TAG block

Purpose

The purpose of this tutorial is to guide you on how to use BasicScript block and Virtual TAG block to calculate or analyse I/O data.

Function

Add three blocks in Task1 window. The first one is an AI block in Task1 that inputs values from Advantech DEMO I/O channel 0. The second one is a virtual TAG block that stores the max. value of AI block. The last is a BasicScript block that calculates the max value of AI block and stores it into virtual block. After that, use five items in display designer to show the current AI value, label the max AI value, and add a lamp for max. value changed.

Procedure

1. Refer to Tutorial 1 to start GeniDAQ and create a new task and display.

2. Create a virtual TAG in system by clicking on the SETUP menu and choose the Add/Delete Virtual Tags option, as follows:
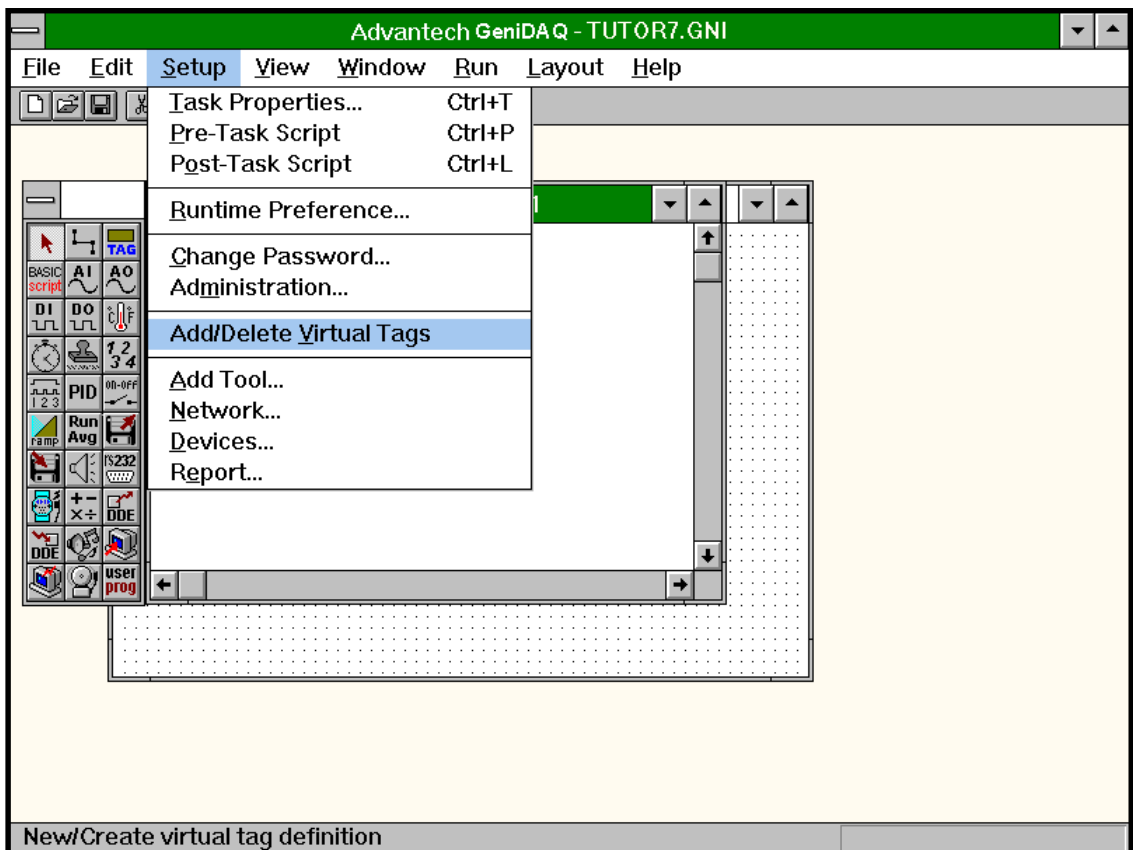
Figure 3-43   Add Virtual TAG

A pop-up window will be displayed to add or delete virtual tags. Choose VR1 at input field and press ADD button to add the virtual tag "VR1". Pressing OK will exit this pop-up window and return you to Task Designer.
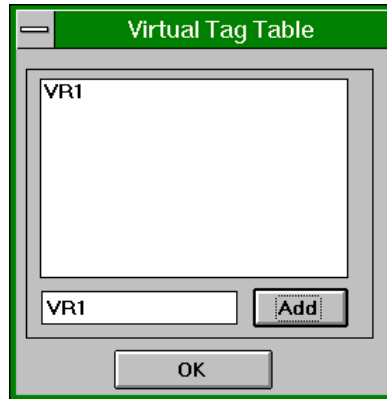
Figure 3-44   Create a new Virtual TAG

3. Add an AI block, a BasicScript block and a TAG block in Task1.
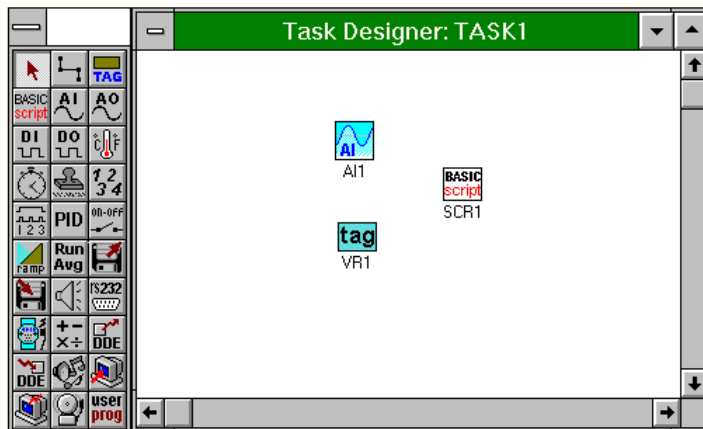
Figure 3-45   Build Task icons

Double click on the AI1 block to configure the input value from Advantech DEMO I/O channel 0. Double click on TAG block to link with Virtual Tag.
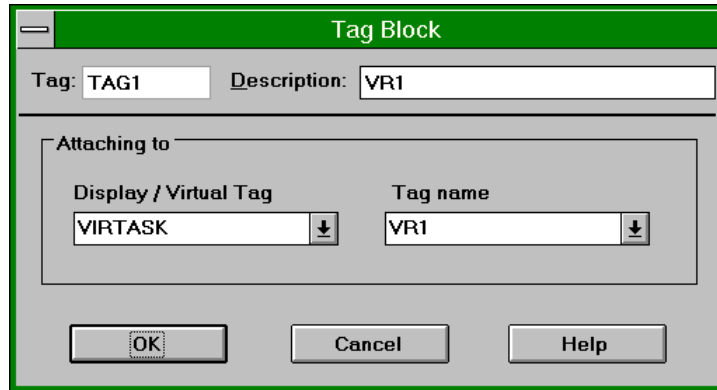
Figure 3-46   Link Tag block to Virtual TAG

Select VIRTASK in the Display/Virtual Tag field and VR1 in tag name field of the above window.

4. Double click BasicScript block to edit Basic script program. Key in the following text program in the pop-up window:

```
Sub SCR1()

      dim mytag1 as TAG
      dim mytag2 as TAG

      set mytag1 = GetTag("TASK1", "AI1")
      set mytag2 = GetTag("VIRTASK", "VR1")

      if (mytag1.value > mytag2.value) then
         mytag2.value = mytag1.value
         outputi 1
      else
         outputi 0
      end if

End Sub
```

5. Add 2 numeric display items, 2 labels and 1 indicator display item in the Display1 window. The 2 numeric display items will be configured to link and display the AI1 value and the virtual tag value in Task1. "Max Value" reflects the virtual tag value and "Current Value reflects the AI1 value.
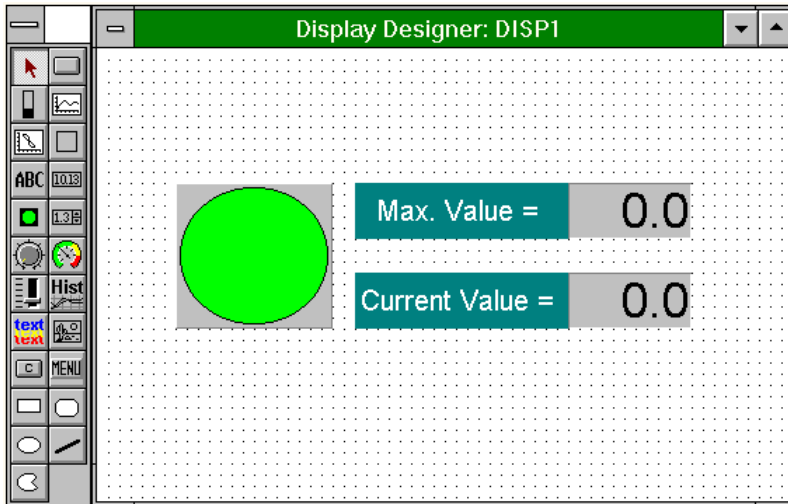
Figure 3-47   Build display screen

Configure the indicator display item to link with the output 0 of BasicScript block.
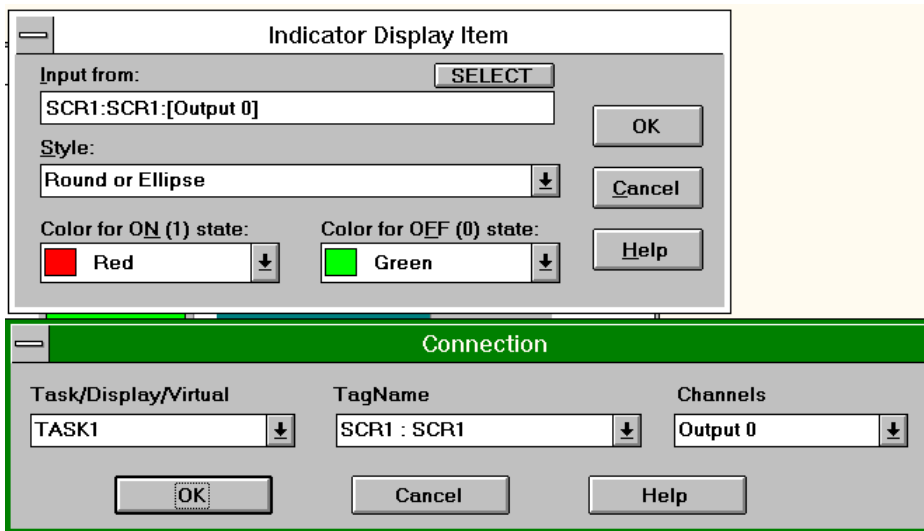
Figure 3-48   Link BasicScript output to indicator display

6. Refer Tutorial 1 Step 9 to save the strategy file as "TUTOR7.GNI".

7. Run the strategy file. At the display window, you will see the AI current value and its max. value. If the AI1 current value is greater that its max. value, the indicator display item will be red. Otherwise, the indicator display item will be green.

3.2.8 Tutorial 8: Main Script programming

Purpose

The purpose of this tutorial is to give you a guide on how to design a Main Script program and use it to control tasks.

Function

1. Create a main script and a task
2. Use main script to start the task and run the task 100 times in main script.

Procedure:

1. Refer to Tutorial 1 to start GeniDAQ and add new Tasks.

2. Repeat Tutorial 1 steps to show Advantech DEMO I/O channel 0 value using a Trend graph.

3. Change the properties of Task1 to be controlled by main script. Click on the Setup menu and Task properties... option to invoke the task properties pop-up window.
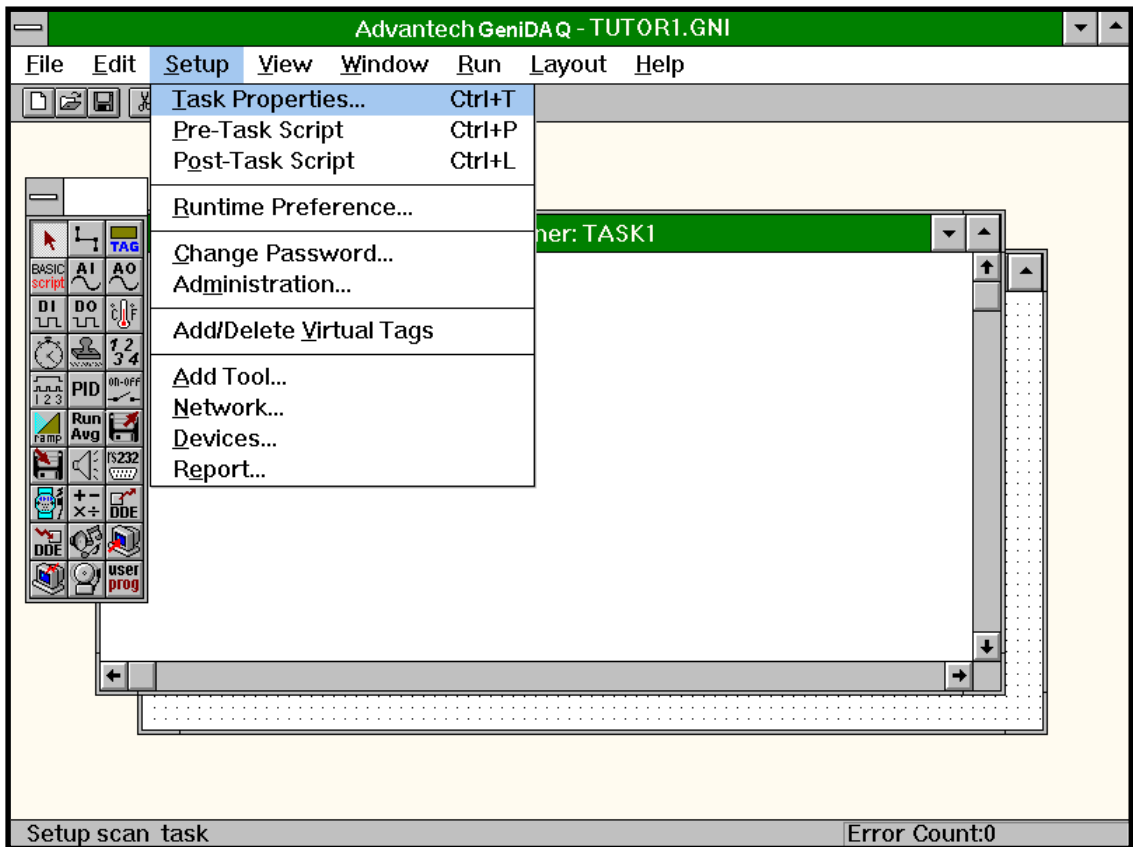
Figure 3-49   Setup Task properties

In the task properties window, change the starting method field to be inactive (activated by script command).

Figure 3-50   Setup ScanTask properties

4. Click on the File menu and choose the Add/Delete, Add Main Script option to invoke Script Designer for editing the Main Script program.
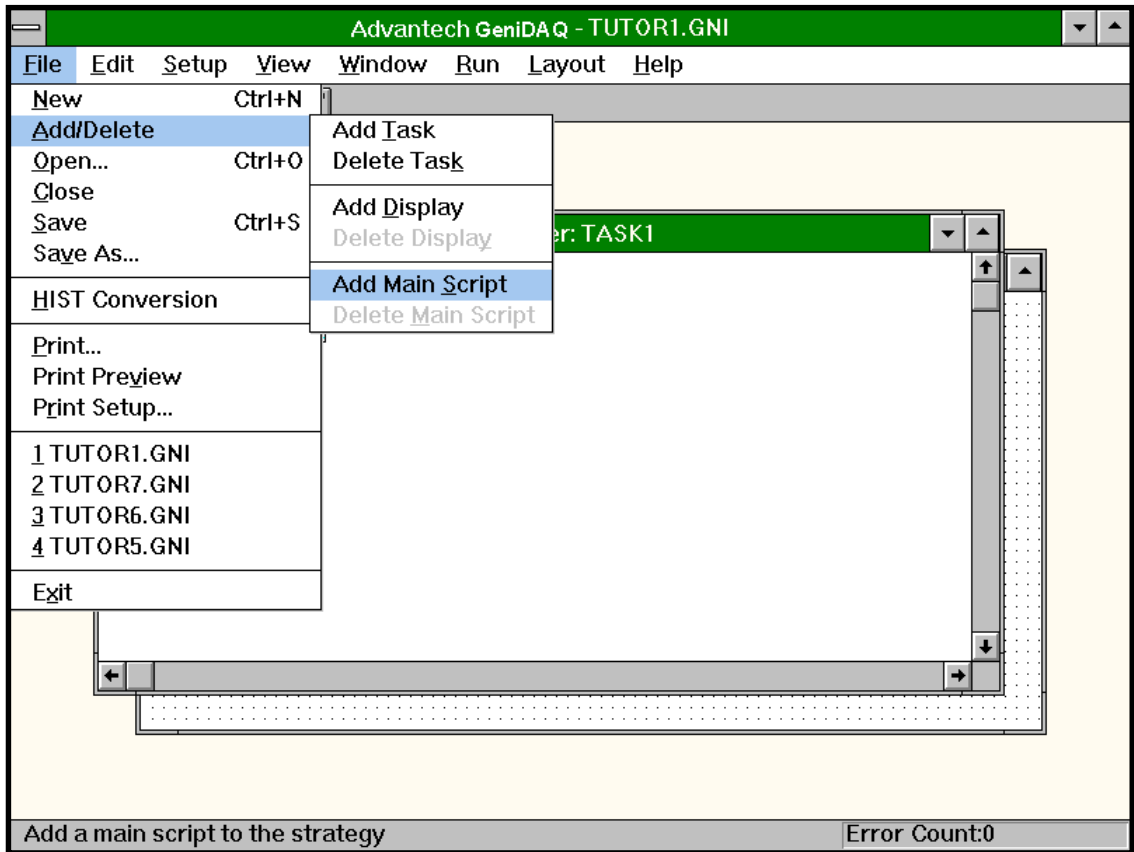
Figure 3-51   Add a main script to the strategy

5. Edit the following program text in the Script Designer window:

```
Sub Main()
      dim mytask as ScanTask
      set mytask = GetScanTask("Task1")
      mytask.start
End Sub
```


Figure 3-52   Edit main script program

6. Refer to Tutorial 1 Step 9 to save the strategy file as "TUTOR8.GNI".

7. Run the main script program by clicking on the Run menu and Main Script menu options in Task Designer.

3.2.9 Tutorial 9: Controlling multiple tasks

Purpose

The purpose of this tutorial is to give you a guide on how to control multiple tasks in your data acquisition and control system through task properties.

Function

1. Create two tasks within an AI block individually

2. Create a display window and add two trend graphs and labels to show the value of AI blocks in two different tasks.

3. Configure task properties to different working modes.

Procedure:

1.   Refer to Tutorial 1 for instructions on starting GeniDAQ and creating a new strategy.

2.   From the File menu select the Add/Delete, Add Task option to add Task2 window.
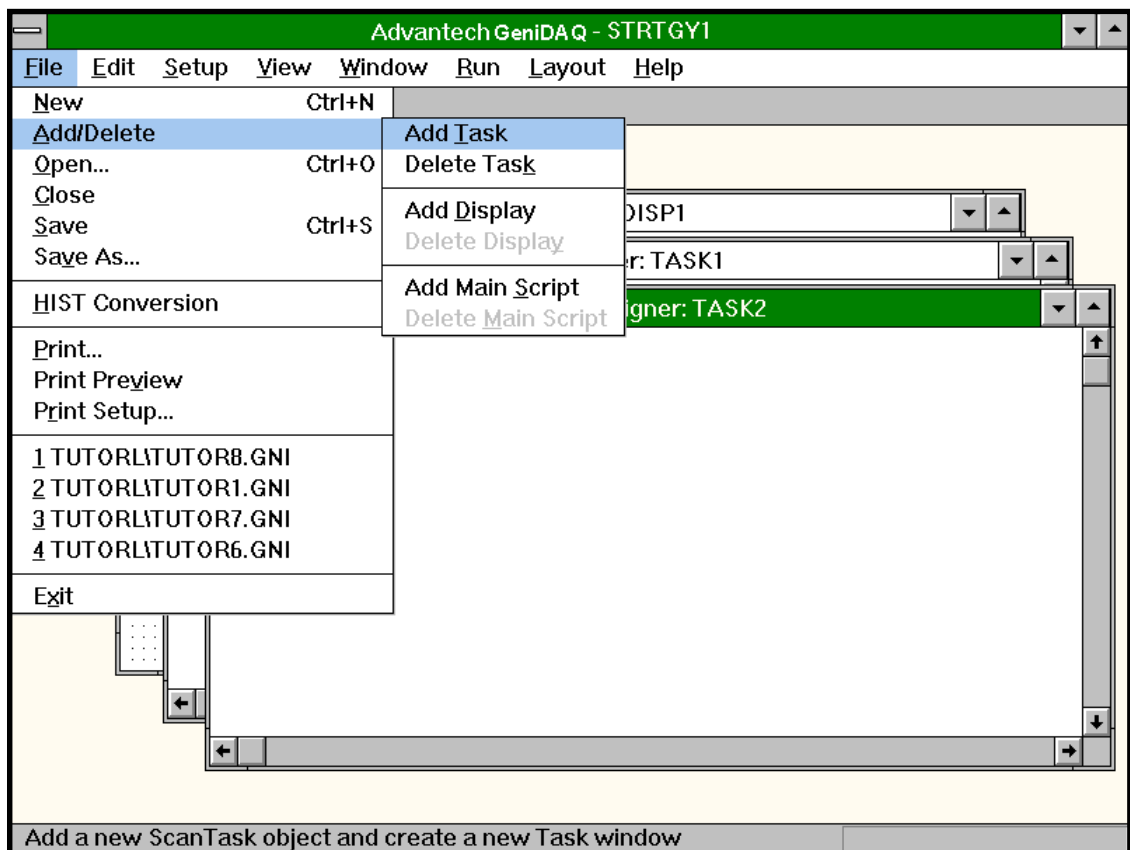
Figure 3-53   Create a new Task

3. Add AI blocks into Task1 and Task2 windows. Configure the AI1 block in Task2 to display the value of Advantech DEMO I/O channel 0 and AI2 block in Task1 to display the value of Advantech DEMO I/O channel 1.
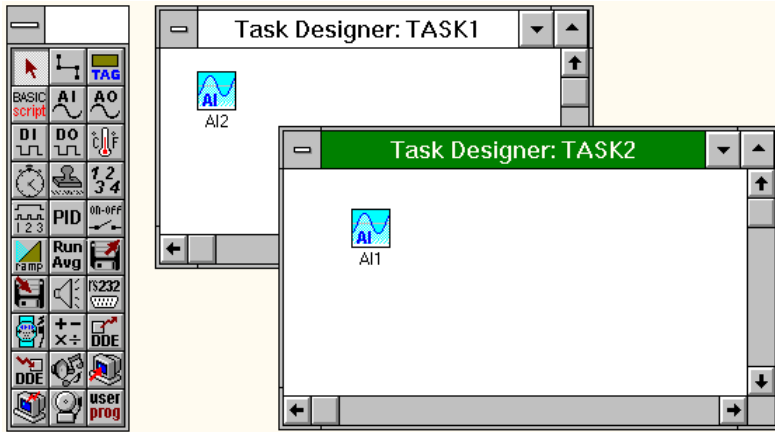
Figure 3-54   Build multiple Task icons

4. Add two trend graphs and two labels in the DISP1 window. Configure the two graphs to show the values of AI1 and AI2.
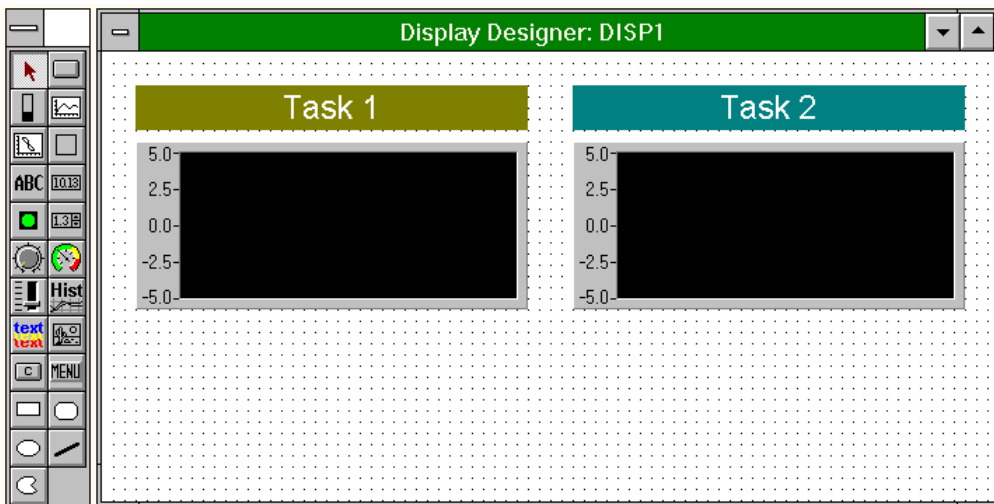
Figure 3-55   Configure trend graphs

5. Configure task properties by activating the task window and clicking on Setup menu and Task properties... options. The duration of Task1 is set to be Indefinitely and the starting method is set to be Immediate. The duration of Task2 is set be 100 times and the starting method is set to be Delayed 5 seconds.

Figure 3-56   Setup TASK1 to be started immediately
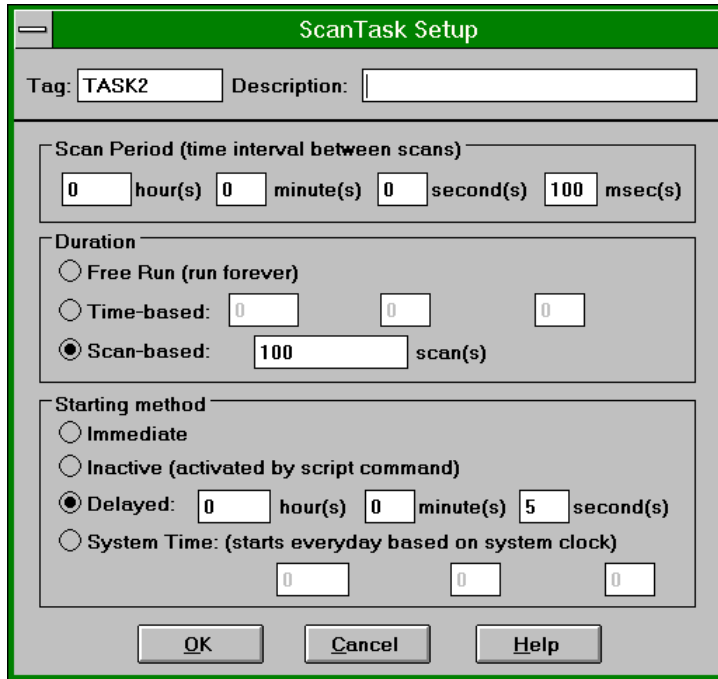
Figure 3-57   Setup TASK2 to be started after a 5 second delay

6. Refer Tutorial 1 Step 9 to save the strategy file as "TUTOR9.GNI".

7. Run multiple tasks by clicking on the Run menu and Start menu options in Task Designer.

4

Device Installation and Configuration

4 Device Installation and Configuration

This section describes how to install and configure your I/O hardware using Advantech I/O device drivers.

After you installed the GeniDAQ program, you can enter the Advantech Device Installation Program (DEVINST) to install and configure each I/O device. The instructions in Section 4.1, Configuring I/O Devices, must be performed for each I/O device you wish to install.

4.1 Configuring I/O Devices

To add or set up an I/O device within the Advantech Device Installation program, go into the SETUP menu and click DEVICE. You will notice a dialog box used to set up I/O devices (pictured below).
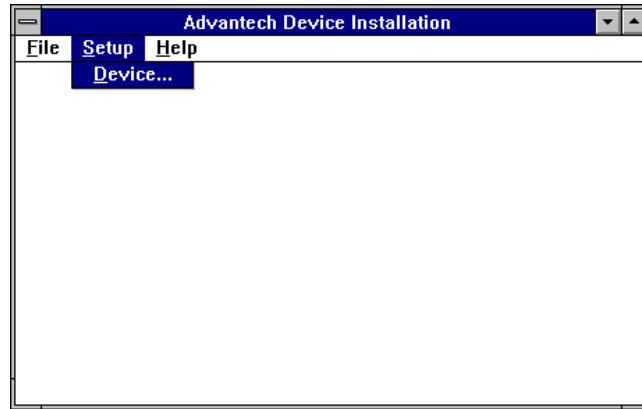
Figure 4-1   Add or set up an I/O device

You can add, setup or remove devices from this dialog box. After you add or remove devices, the existing device list will show the status of I/O devices currently installed.

Setting Up or Configuring a Device
To set up or configure/re-configure an I/O device within the Device Installation program, go into the SETUP menu, and click on DEVICE. Highlight the previously installed I/O device you wish to set up, then press the Setup button. This will bring you to a device-specific dialog box that allows you to configure or re-configure the device.
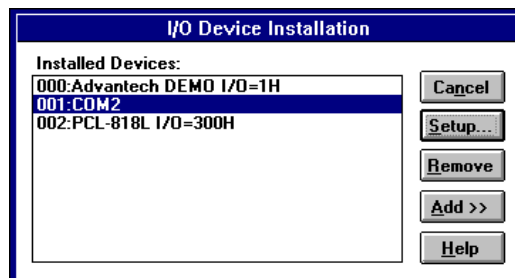
Figure 4-2   Set up or configure an I/O device

Configure the device, and when you're satisfied with your entries, press OK. This will bring you back to the I/O DEVICE INSTALLATION dialog box, where you can see a list (Installed Devices) reflecting your I/O device configuration.

Adding a Device

To add and set up a new I/O device in the Device Installation program, go into the SETUP menu, and click DEVICE. Choose the Add button, and you will see a List of Devices pop-up box. This list includes the device driver(s) previously installed.

Figure 4-3   Add an I/O device

Highlight the device and press INSTALL, or double click on it. This will bring you to a device-specific dialog box that allows you to configure the new device.
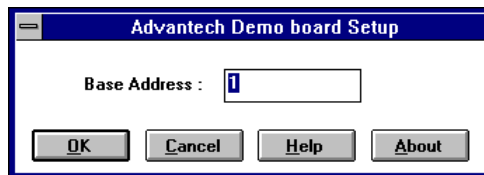
Figure 4-4   Install Advantech DEMO Board

Configure the device, and when you're satisfied with your entries, press OK. This will bring you back to the List of Devices box, where you can add another I/O device of the same type (double click, or INSTALL). When you have completed your device installation and setup, there should be a displayed list reflecting all installations in the I/O DEVICE INSTALLATION dialog box.

Removing a Device

To remove a previously set up I/O device within the Device Installation program, go into the SETUP menu, and click DEVICE. Highlight the device instance you wish to remove in the I/O DEVICE INSTAL-LATION dialog box, and then press the REMOVE button.
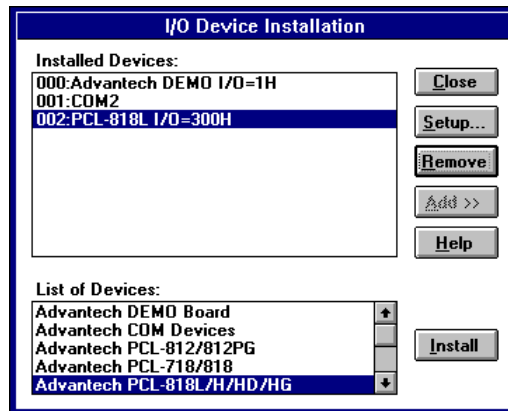
Figure 4-5   Remove an I/O device

When you have completed your device installation and setup, a list will be displayed reflecting all installations and removals. The Remove button in the I/O DEVICE INSTALLATION dialog box will only remove each instance of the I/O device, not the DLL driver itself.

4.2 Example: Install the PCL-818L

This section provides you with an installation example using the Advantech PCL-818L.

1) To add and set up each PCL-818L, go into the SETUP menu and click DEVICE. Choose the Add button, and you will see a box with a list of PCL-818 device drivers installed. Highlight PCL-818L and press INSTALL, or double click on it.
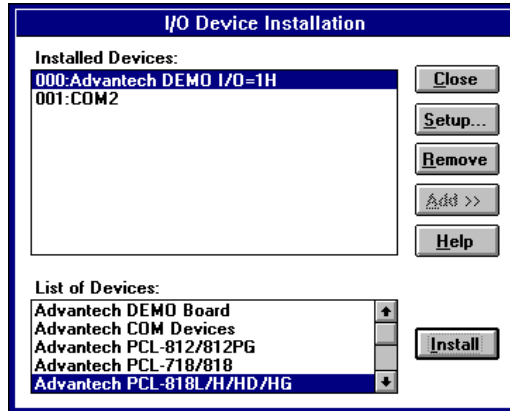
Figure 4-6   Add Advantech PCL-818L

This will bring you to the PCL-818L device specific dialog box that allows you to configure each PCL-818L. Configure the device by changing the base address, A/D range, etc., corresponding to your hardware's switch and jumper positions. When you're satisfied with your entries, press OK. (Device-specific HELP may be accessed by clicking on the HELP button in the PCL-818L configuration dialog box). This will bring you back to the I/O DEVICE INSTALLATION dialog box. From here you can either add another PCL-818L I/O card at a new base address (INSTALL), or you can REMOVE, change the SETUP, or CANCEL to exit the program. When you have completed your device installation and setup, there should be a listing reflecting your installation displayed in the I/O DEVICE INSTALLATION dialog box, such as "PCL-818L I/O = 300H".
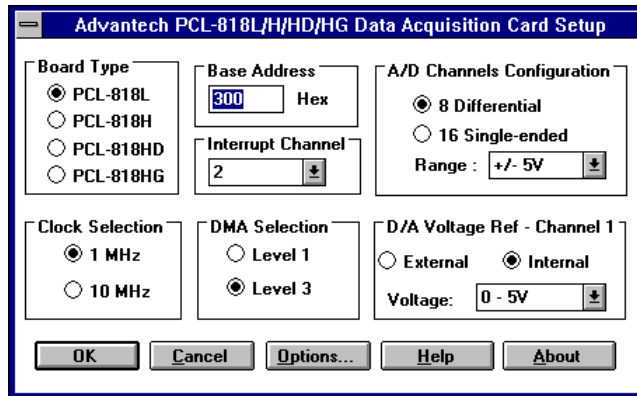
Figure 4-7   Configure Advantech PCL-818L

2) Once the PCL-818L is installed, you may change its parameters by either double clicking on its entry (such as "PCL-818L I/O = 300H") in the list box, or highlighting the listing and pressing the Setup button.
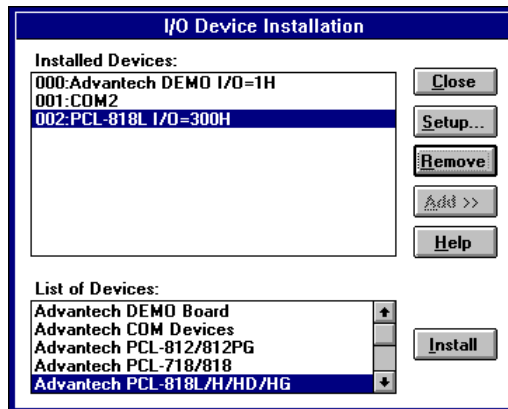
Figure 4-8   List installed PCL-818L

4.3 Example: Install ADAM-4000
This topic provides you with an Installation Example using the Advantech ADAM-4000.

1) To add and set up each ADAM-4000 I/O device, go into the SETUP menu of the Device Installation program, and click DEVICE.
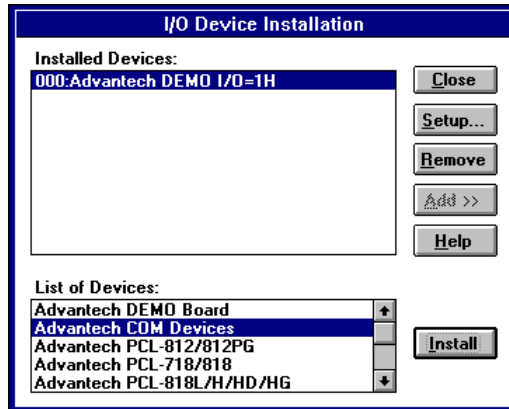
Figure 4-9   Add Advantech COM devices

Choose the Add button, and you will see a box containing a list of the Advantech COM devices drivers you just installed. Highlight Advantech COM devices and press INSTALL, or double click on it. This will bring you to the COM port devices window. Press Port to select the communication port used to connect with ADAM-4000 devices.
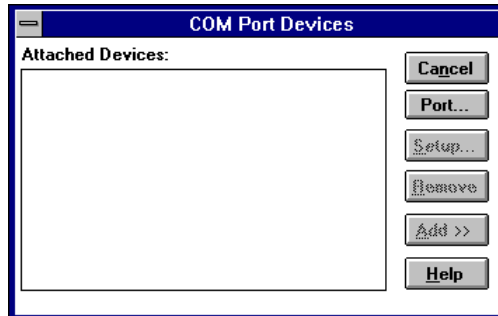
Figure 4-10   Add COM port device
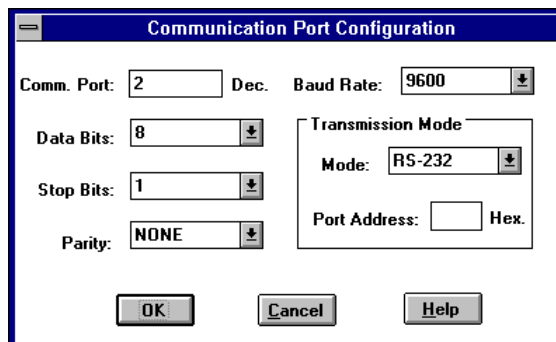
Figure 4-11   Configure COM port device

The Communication Port Configuration window will be displayed. After configuring the COM port, choose the Add button. You will see a box containing a list displaying specific dialog box that allows you to configure Advantech ADAM-4000 and ADAM-5000 series devices.
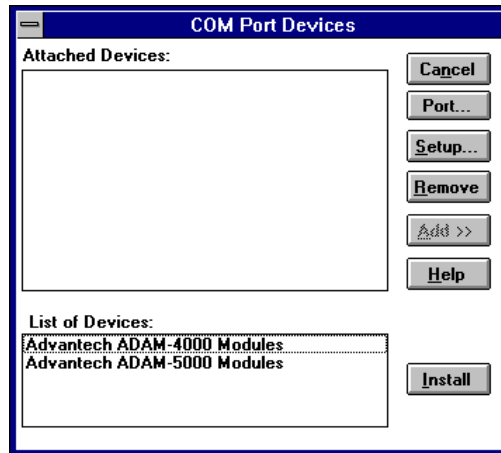
Figure 4-12   Install ADAM-4000/5000 modules

Select Advantech ADAM-4000 devices and press the Install button to specify an ADAM-4000 device. Configure ADAM-4000 modules by selecting the module type, module address, time-out, checksum, alarm enabled, prefix string and postfix string, etc., corresponding to your hardware's positions. When you're satisfied with your entries, press OK. (Device-specific HELP can be accessed by clicking on the HELP button in the ADAM-4000 configuration dialog box).
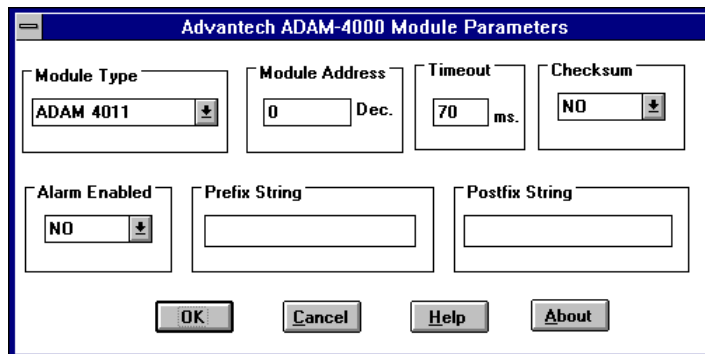
Figure 4-13   Configure ADAM-4000 module

This will bring you back to the I/O DEVICE INSTALLATION dialog box. From here you can either add another ADAM-4000 I/O module at a new base address (INSTALL), or you can REMOVE, change the SETUP, or CANCEL to exit the program. When you have completed your device installation and setup, there will be a list reflecting your installation displayed in the I/O DEVICE INSTALLATION dialog box, such as "COM2".
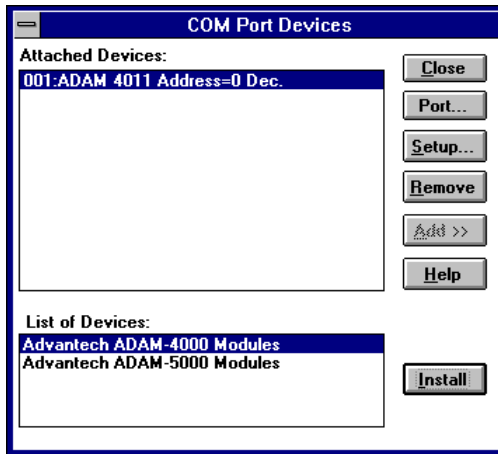
Figure 4-14   List installed ADAM-4000 modules

2) Once the ADAM-4000 device is installed, you may change its parameters by either double clicking on its entry (such as " COM2") in the list box, or highlighting the listing and pressing the Setup button. You will see a box containing a list of ADAM-4000 modules installed. You can modify Advantech ADAM-4000 devices by selecting the desired module and pressing the setup button. A configuration window will appear to guide you through the process of modification.
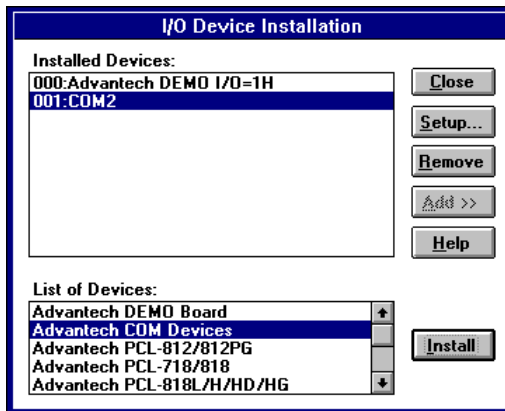
Figure 4-15   List installed COM ports

5

Task Designer

5 Task Designer

GeniDAQ Task Designer uses a dataflow programming model that frees you from the linear architecture of text-based languages. You can construct the block diagram without worrying about the many syntactical details of conventional programming.

Scan Tasks are developed in the Task Designer. The Task Designer uses a toolbox containing icon blocks to simplify the development of your process control and/or data acquisition strategy. Each icon represents one particular built-in function that you can use to solve your problem. Some icons interface with low level drivers, which in turn interact with hardware devices that you want to control.

5.1 Working with Task Designer

Using the Toolbox and Working area

Task Designer consists of two major components: Task Toolbox and Working Area. Task Toolbox contains many icon blocks to simplify the development of your process control and/or data acquisition strategy. Each icon represents one particular built-in function that you can use to design your system. Working Area is the area that you drag and drop icon blocks in to construct the system control scheme.

Figure 5-1   Task Designer window

Using the mouse, these icon blocks are applied in developing the logic that will be used for interaction with your process. All Task Designer block related operations, such as placement, sizing, moving, connections, etc., are made in that area.

A strategy is developed by clicking on the desired icon block, pointing the mouse to the desired spot in the working area, and clicking the mouse again to "drop" the block at that location. The block must be connected to other block(s) and then configured.

When all of the blocks necessary to perform the desired process application have been placed, connected, configured, and saved, the system operations are done completely and you can design your operator displays through Display Designer and configure display items to link with Task icon blocks to dynamically display data.

Connecting Blocks

All block connections are made while the **Connect function** is enabled. Enabling the Connect function is accomplished by moving the mouse cursor inside the Connect icon (the angled wire) and clicking the left mouse button. When the function is enabled, a "roll of wire" appears instead of the standard mouse cursor. In this mode, connections can be started and terminated between various Task Designer blocks. Connections may be made:

1) Block to Block

This is the most common connection in a strategy. You simply click on the source block, click to make a wires turning or corner point, and finally click on the target block. If the source block or target block provides multiple inputs or outputs, then you need to select one of them in the listbox.

2) Block to Display Item

When connecting a block to a display item, you have to go into the Display Designer window, double-click to bring up the dialog box of the display item you want to connect to, and then select the source block from the combo box in the dialog. This completes the connection from a block to a display item.
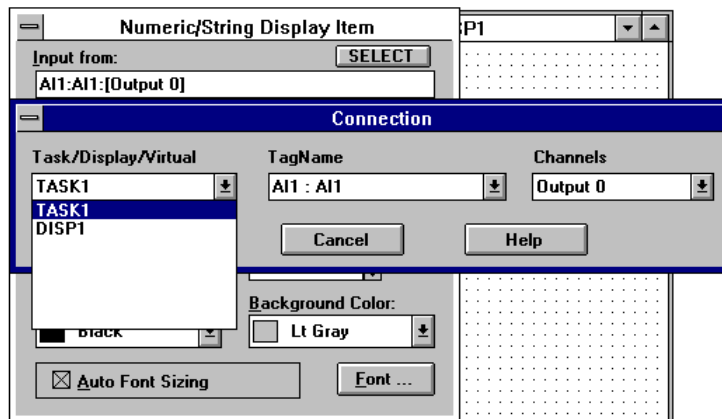
Figure 5-2   Connect a block to a display item

3) Display Item to Block

If the source of the connection is a display item then a Tag block in the Task toolbox needs to be created first in order to make a connection. After creating a display control item in Display Designer window, you have to create a TAG block in Task Designer window and configure it to represent this display control item. This is different from GENIE 2.xx. This improvement allows one display control item to connect to several blocks and allows one display control item to connect to or configure multiple blocks in one or more tasks.
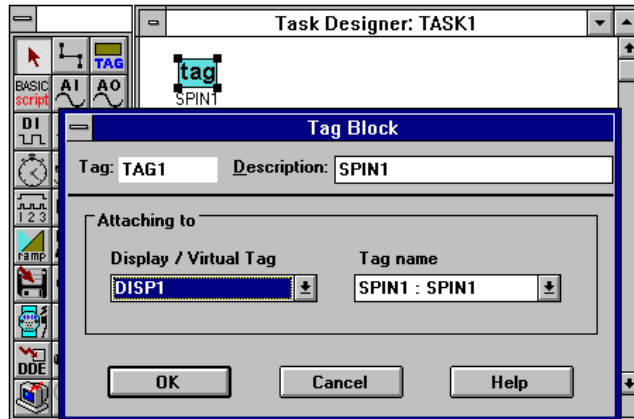
Figure 5-3   Connect display control item to Task block

4) Display Item to Display Item

This type of connection can be made between different display windows or in the same window. There are two ways to implement this type of connection. One is to create a first display item and then configure a second display item to link with the first. These two display items can be in different display windows. The other way is to link different display items through TAG in Task Designer. That is, one display item can pass data to the TAG of Task Designer, and another display item is configured to access this TAG.
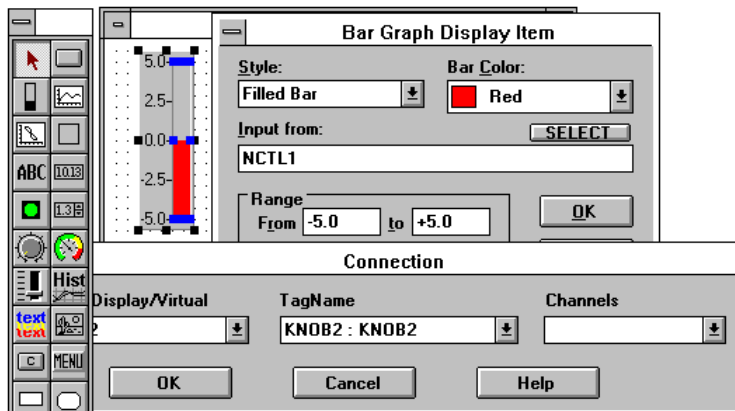
Figure 5-4   Connect display item to display item

88

Connecting Icons:

Moving the mouse causes the line to grow/shrink and move about the perimeter of the block. The line is always perpendicular to the block. Clicking on the left mouse key causes the current line segment to be anchored. This feature allows you to lay down line segments, as you require, for instance to move around other blocks or connections. Continue to lay down connection line segments until you arrive at the destination block (a block that accepts data as an input). If the block has multiple input/channel capability (such as a block with a reset/hold or other input choice), a dialog box will be displayed. You will be allowed to choose which connection parameter or channel to input before connections can be made.
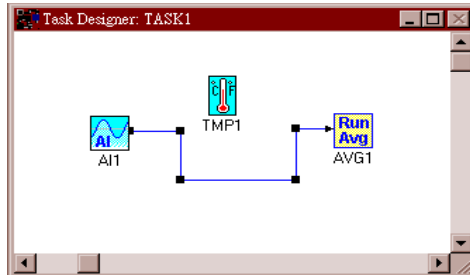
Figure 5-5   Connect icons
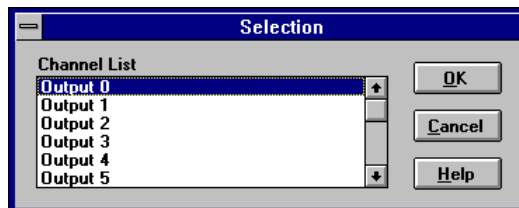
Figure 5-6   Select multiple output/channel

When connection is made, a blue "wire" will connect the two blocks, indicating that the connection is valid. If the connection is not valid, an error message will appear, informing you that you cannot make the specified connection and the reason for the error (you may be trying to connect two blocks that only have an output, etc.). GeniDAQ performs data type checking in the Task Designer. A link cannot be made between incompatible types of data. Keep repeating the connection procedure until all the necessary connections have been made. Blocks should be interconnected so as to form a logical flow, duplicating the signal flow in your process. When you're satisfied with the connections you have made, save the strategy.

Configuring Blocks

After you have completed the block connection process (the final step in configuring your process strategy), configuring the blocks can begin. Once all blocks are connected, they must be configured by **double clicking** on each icon and setting configuration information in each associated **dialog box**.

Entering strategy block parameters is accomplished by simply entering the appropriate parameter values from the keyboard into the necessary fields, or by selecting/scrolling values with the mouse. Each block is initially displayed with default values that can be overwritten with your desired values. The arrow cursor controlled by the mouse button and/or the TAB key is used to move to the various parameter fields in each configuration dialog box. When you have entered all of the desired parameter values for the current block, pressing the OK button, or pressing the [Enter] key incorporates the contents of the block's configuration menu into the strategy database. Pressing the CANCEL button discards all parameter changes made and restores the previous values.

The configuration information is saved into the strategy database whenever the strategy is saved. Once a block has been configured, its configuration parameters can be viewed and edited at any time.

Rearrange Blocks' execution order

The new task designer is also improved by adding the block sequence arrangement feature that shows the order of execution on all blocks. Users can rearrange the order of execution of the blocks (icons) under some restrictions. It shows the order of execution among blocks in run-time program. The user selects **Order Layout** from the **View** menu to display the order number of each block. The order of blocks can be changed using the **Complete Reorder** and the **Exchange Order** options from the **Layout** menu. Please refer to *Chapter 5.2 Task Designer Menu* for detailed information.
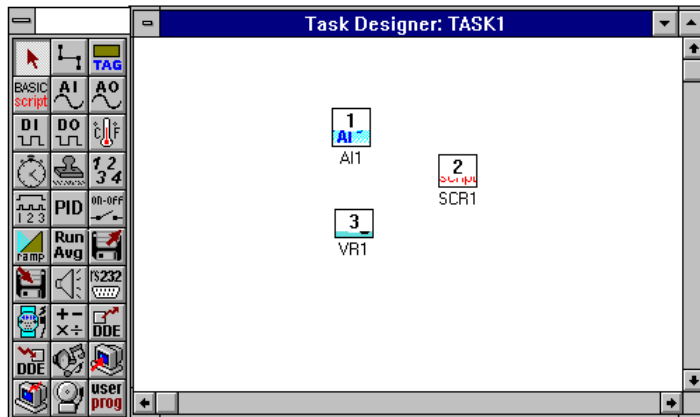
Figure 5-7   View blocks' execution order

Saving and Loading Your Strategy

The Task Designer provides a simple means for saving and loading strategies to and from your hard disk. The **File** menu on the left side of the menu bar is used to accomplish both of these functions. If you are familiar with conventional Windows functions, you will find GeniDAQ to be the same as other Windows programs. Clicking on **File** menu when in the Task or Display Designer will invoke the File submenu. This will allow you to **Save** the strategy if already named, or use **Save As** if naming or renaming a strategy file. Strategy files always have the extension **.GNI** for GeniDAQ. For instance, you could name your strategy file "TEST.GNI".

Saving the Strategy

Periodically, or whenever a significant amount of work has been completed in the current strategy, it is advisable that the strategy be saved to disk to insure that the work is stored in the event of an accident.

If the strategy has never been saved, or if you desire to save the strategy under a different name, use the **Save As** command. Enter the path (if the file will reside in a directory other than the GeniDAQ directory) and file name of the strategy in the edit box provided. You can also just type the file name after using your mouse to choose the file path, which is displayed on the right in the **Save As** dialog box. When satisfied with your entry, press the OK button with your mouse, or simply press [Enter]. Your GeniDAQ strategy file will now be saved, and you can continue design or exit the Task Designer to later resume work.

If the strategy has been named and saved to disk before, to save the current work, simply click on Save with your mouse. The strategy will be saved under the name chosen before.

If the strategy has a block or display item not yet configured, the message **Save Partially Configured file?** will be shown. You can save or press the Cancel button to abort the saving process. A partially configured strategy can not be run.

Figure 5-8   Show undefined block or display item
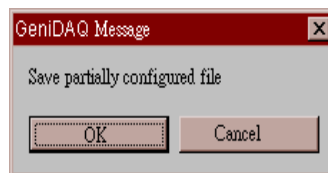
Figure 5-9   Warning message for saving file

Loading a Strategy

When GeniDAQ is invoked, the strategy worked on most recently will automatically be loaded and displayed. Should you need to load a different strategy from the disk into the Task Designer, the File submenu's **Open** function provides the tools to accomplish this task.

To open an existing file, Click on **Open** in the File menu. A dialog box will appear which allows you to either type in the path and filename of the "gni" file you wish to open, or browse the directory path to choose the desired file. Highlight your choice. When you're satisfied with your entry, press OK, double-click on the entry, or press [Enter]. The strategy will then be displayed.
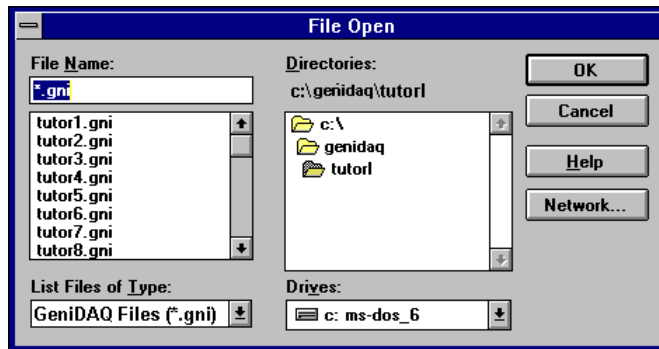
Figure 5-10   Open strategy file

If changes were made to a strategy currently displayed, when the **Open** function is invoked a message box will appear prompting you to save the current changes before the new strategy can be loaded.

If you want a fresh screen, or a new strategy is to be designed, use the File submenu's **New** function.

Modifying Your Strategy

During your strategy editing process there will be times when you need to make changes. The **Edit** menu provides functions that allow you to modify your strategy quickly and easily.

Selecting Task Designer Blocks

Blocks in the Task Designer can be "selected" to perform various operations, either separately or in sets. When an item is selected, enlarged black squares appear either on the perimeter of an icon or display item (see below), or along a wire connection.

To select an item, simply point to the item with the mouse and click. The enlarged squares mentioned above should then appear.

To select multiple blocks, "grab" or select the blocks you wish to select by pointing your mouse somewhere outside the group of blocks you wish to select. Press the left mouse button, and as you hold down the button, drag the pointer diagonally across the blocks. You will see a box form around the blocks. When the box is around the desired blocks, let go of the mouse button. When blocks are selected properly, there will be heavy black squares at the corners of all selected blocks. Now you can perform commands on selected blocks. All blocks remain selected until another mouse operation (click) is performed.
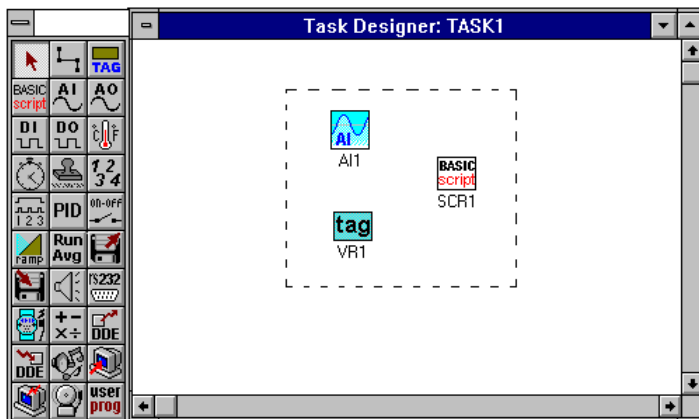
Figure 5-11   Select multiple blocks

To select all blocks in the currently displayed strategy, the Edit submenu function **Select All** may be chosen.

Moving Selected Blocks

To move one or many selected blocks in the GeniDAQ Task Designer, press the left mouse button as you point to one of the selected blocks. As you hold down the left button, drag the selected blocks to the desired location, and let go of the mouse button when the location is reached.

Deleting

The **Delete** function in the **Edit** submenu allows you to perform the operation of deleting any number of selected blocks. Simply select blocks using the above procedure, and then click on **Delete**, or press the [**Del**] key. Clicking on this command will cause all previously selected blocks to be erased (after a prompt asking you if you're sure).

Copying

To copy blocks in GeniDAQ, first select the blocks you wish to copy (see above, Selecting Blocks). Use your mouse to choose the copy command. This will copy the selected blocks into a buffer. The blocks may now be "Pasted" to another location by pressing "**Paste**" in the **Edit** menu. The selected blocks are now copied to the upper left of the screen area with new tag names. Now they may be moved to the desired location as a group, since they are all selected. Refer to the **Moving Selected Blocks** section above for instructions on moving groups of blocks.

Exiting the Task Designer

When you have finished your work in the Task Designer Display Designer and desire to return to Windows or DOS, clicking on the **Exit** command in the **File** submenu will accomplish this task. You will be prompted as to whether you would like to save any changes.

Icon Blocks (Task Designer Objects):

Analog Input
Analog Output
Alarm Log
BasicScript
Digital Input
Digital Output
Event Counter
File, Input
File, Log
Hardware Event/Frequency Counter/ Pulse Output
Hardware Alarm
Moving Average
Network In
Network Out
On/Off Control
PID Control
Ramp
RS-232
Single Operator Calculation (SOC)
Speaker
Temperature Measurement
Timer
Time Stamp
Triggering
Wave file Playback

A more detailed review is made in the *chap 5.3 Task Toolbox.*

5.2 Task Designer Menu

The Task Designer Menu Bar consists of a number of menus that enable you to manipulate files, edit your strategy, setup I/O devices, run your strategy, view your strategy or manipulate windows. There is also on-line help available through the Help menu.

Task Designer Menus
> *File Menu*
> *Edit Menu*
> *Setup Menu*
> *View Menu*
> *Window Menu*
> *Run Menu*
> *Layout Menu*
> *Help Menu*

Following is a description of the Task Designer menus:

5.2.1 File Menu

The File menu includes commands that enable you to open, close, save, and print new or existing GeniDAQ Strategy Files. Commands include:
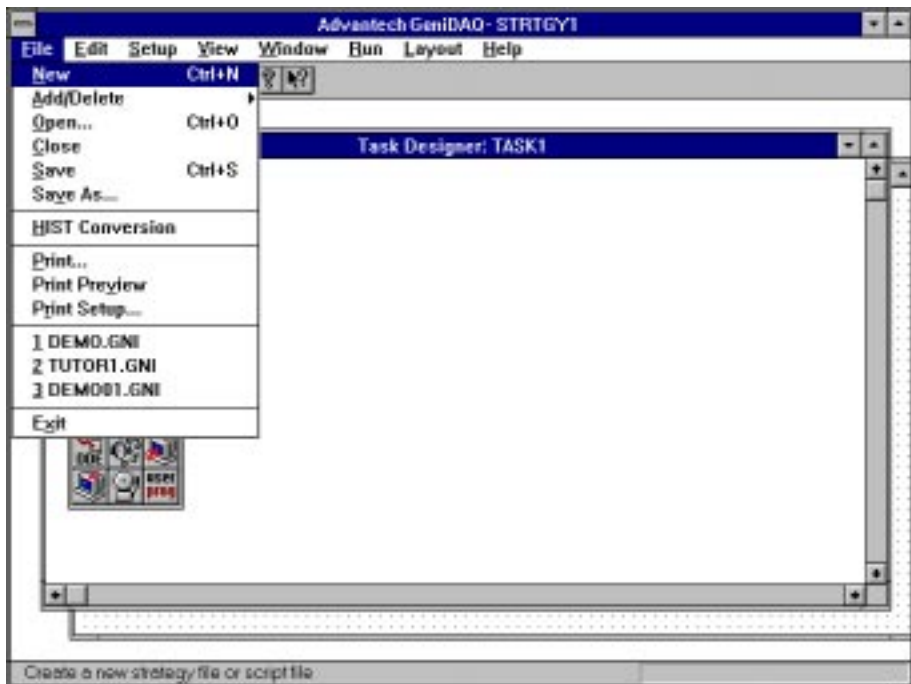
Figure 5-12   Task Designer — File menu

New

Use this command to start a New strategy. During or after you create a strategy, you can save the strategy by using the **Save As** command.

Add/Delete

Use this command to add/delete a "task", "display" and "main script" in GeniDAQ strategy.
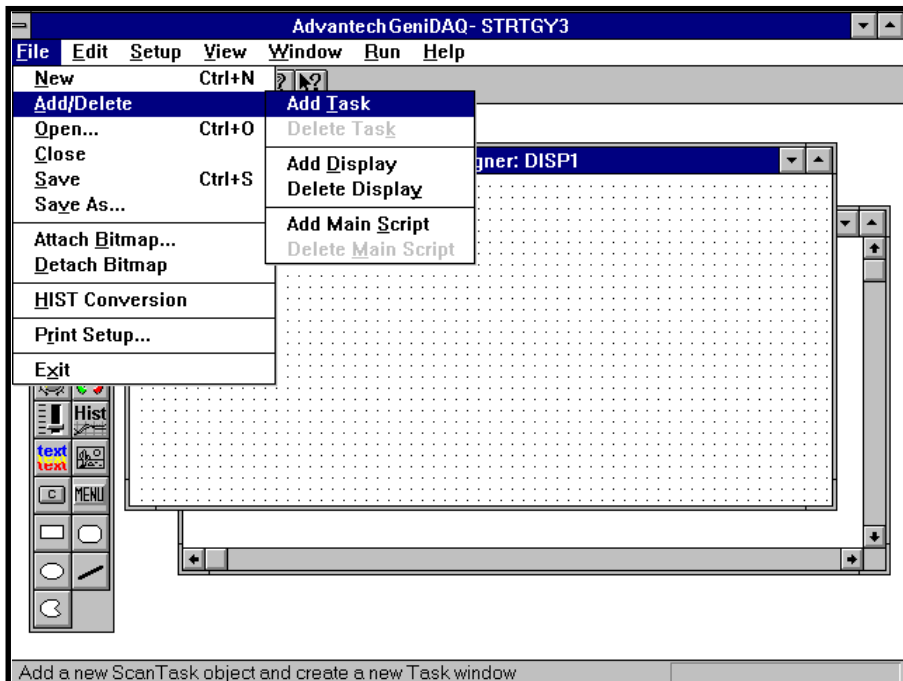
Figure 5-13   Task Designer — Add/Delete submenu

Add Task/Delete Task

Use this command to add or delete a "task". A task is a collection of blocks that are related to each other. GeniDAQ supports up to eight tasks, and each task can be designed independently and executed simultaneously. You can toggle between multiple tasks during runtime or development time. Basically, one task ("TASK1") will be created automatically when you create a strategy file.

Add Display/Delete Display

Use this command to add or delete a "display". A display is a collection of display blocks that are related to each other. Each display has its display properties that are set by clicking **setup** menu and **display properties** options. GeniDAQ supports multiple displays to be designed independently and executed simultaneously. You can toggle between the displays during runtime or development time. Basically, one display ("DISP1") will be created automatically when you create a strategy file.

Add Main Script/Delete Main Script

Use this command to add or delete a "main script" for each strategy file. A main script file is used to control tasks in a strategy file. In GeniDAQ, there is only one main script file for each strategy.

Open

Use this command to Open an existing strategy. Normally, existing strategies will be stored in the **GeniDAQ** directory.

Close

Use this command to Close your strategy. Make sure that you save it first.

Save

Use this command to Save your previously named strategy. To save an unnamed strategy, GeniDAQ will activate a "File save as" window. Users can key in a filename or use the **Save As** command. New strategy files must be saved before they can be run.
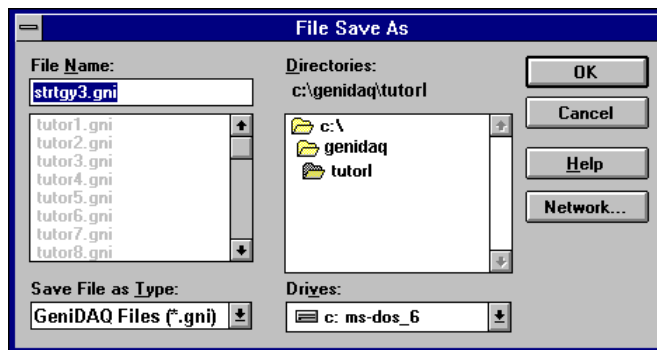
Figure 5-14   File Save As pop-up panel

Save As

Use this command to save a previously unnamed strategy. You should provide a path to where you would like the strategy stored, if different from the default C:\GeniDAQ directory.

Hist Conversion

Use this command to convert the historical trend data file from binary file format to ASCII file format for query or other applications use.
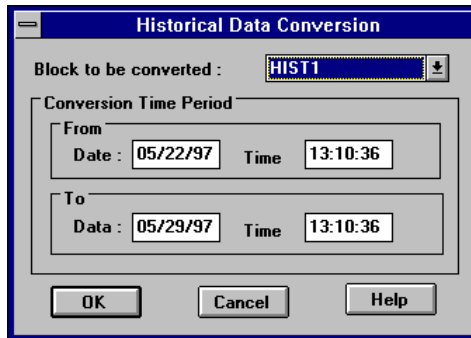
Figure 5-15   Hist Conversion pop-up panel

Print

Use this command to Print your strategy connections. For this version, you will print only blocks and block connections of the first task (default is "Task1").

Print Preview

Use this command to preview what will be printed.

Print Setup

Use this command to Setup your printer. All Windows supported printers are supported.

Exit

Use this command to Exit the GeniDAQ Task Designer.

5.2.2 Edit Menu

The Edit menu includes commands that enable you to edit blocks in your GeniDAQ Strategy. To use Edit commands, you must first "grab" or select the blocks you wish to copy by pointing your mouse somewhere outside the group of blocks you wish to select. Press the left mouse button, and as you hold down the button, drag the pointer diagonally across the blocks you wish to select. You will see a box form around the blocks. When the box is around the desired blocks, let go of the mouse button. When blocks are selected properly, there will be heavy black squares at the corners of all selected blocks. Now you can perform commands on selected blocks.
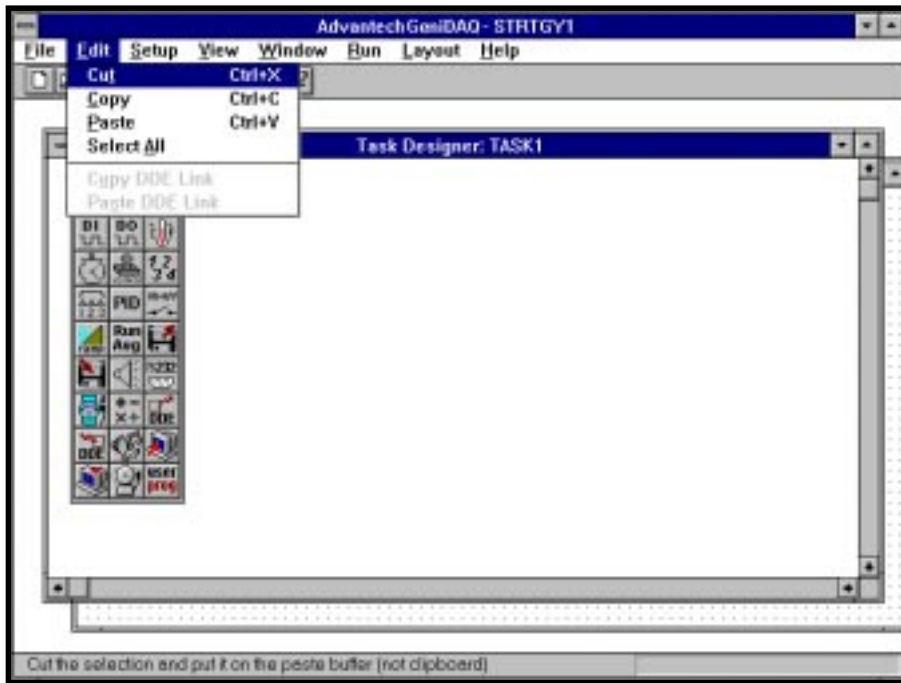
Figure 5-16   Task Designer — Edit menu

Cut
To Cut blocks in GeniDAQ, first select blocks that you wish to cut (see above, to Select Blocks). Use your mouse to select the cut command. This will cut the selected blocks. The blocks are removed from the screen and *may not* be "Pasted" to another location.

Copy
To Copy blocks in GeniDAQ, first select blocks that you wish to copy (see above, To Select Blocks). Use your mouse to select the copy command. This will copy the selected blocks into a buffer. The blocks may be "Pasted" to another location.

Paste

Press "**Paste**" in the Edit menu to paste the copied blocks to the upper left area of the screen, with new tag names. The group of blocks can now be moved to the desired location, using the instructions for **Move** (above). Remember that tag names must always be unique for like-blocks. This is the way that GeniDAQ Runtime distinguishes one block from another. Once blocks are selected (see above, To Select Blocks), they may be copied to another location by first pressing Copy, and then pressing the Paste command.

Select All

To select every block in the current Strategy, use this command.

Copy DDE Link

GeniDAQ provides "Copy DDE Link" and "Paste DDE Link" menu options to establish the DDE connections between GeniDAQ and other applications. After you select "DDE link - Server" blocks, you can select the "Copy DDE Link" menu option in Edit menu to setup DDE linkage for other applications.

Paste DDE Link

GeniDAQ provides "Copy DDE Link" and "Paste DDE Link" menu options to establish the DDE connections between GeniDAQ and other applications. After you select "DDE link - Client" blocks, you can select the "Paste DDE Link" menu option in the Edit menu to paste the DDE linkage from another application.

5.2.3 Setup Menu

The Setup menu includes commands that enable you to install, add, or remove an I/O device, and also to assign the Strategy to a particular Task. Commands are as follows:
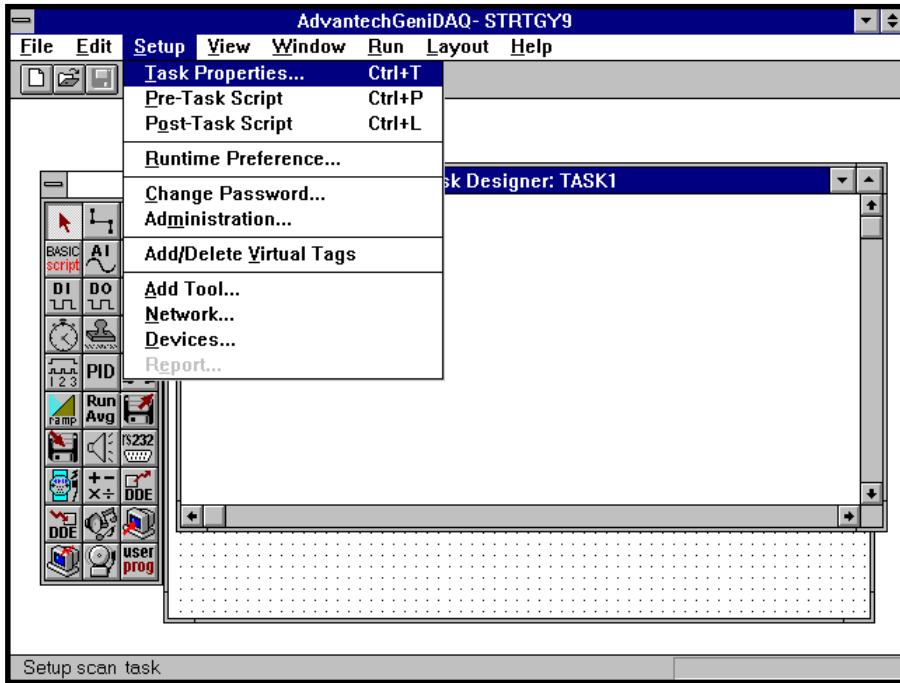
Figure 5-17 Task Designer — Setup menu

Task Properties

This command allows you to define the "scan period", "duration" and "starting method" of a specified "Task". You assign each scan task its own sample period, down to milliseconds resolution in the Scan Task Setup dialog box. All blocks in the task will be executed based on the scan period or scan rate. The scan period of a task block can be assigned to be inversely proportional to the sample period of the task.
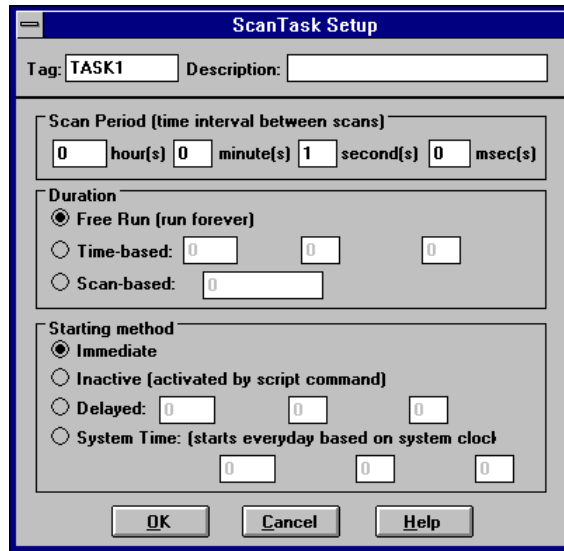
Figure 5-18   ScanTask Setup menu

From this dialog box, you can supply a description for the preassigned task name. You can also set the Scan period (the time interval between two samples) in hours, minutes, seconds, and milliseconds. It should be noted that the scan period should be set to the longest time that your strategy will allow, thereby reducing the processing load on the program.

The **Scan Period** of the assigned task can be set based on the time intervals between scans that you want the Scan task to run. You can set the scan period to be hours, minutes, seconds and milliseconds. Please refer to the task properties dialog window.

The **Duration** of the assigned task can also be set, based on whether you want the Scan Task to run forever (Free Run), run for a limited time (Time Based), or to run for a limited number of samples (Sample Based). Once all blocks have been selected, you can Close the menu (OK), Cancel, or choose Help by selecting the appropriate button.

The **Starting method** of the assigned task can be set, based on whether you want to start the task immediately, inactive (activated by script command), delayed (hours, minutes and seconds) and system time (starts everyday based on system clock). For the inactive starting method, you can start a task using the command "task.start" in the main script file and the run main script. For the delayed starting method, the task will be pending until expiration of a time delay. For the system time starting method, you can specify the time at which the task will start every day. For example, you can set the starting time of task B to be 9:00 AM. The task will not be executed until 9:00 AM each day.

Pre-Task Script/Post-Task Script
Each scan task has a pre-task script and a post-task script. Pre-task script is executed once before executing task and post-task script is executed once after executing task. Users can use these two scripts to initialize values before running a task or to reset values after exiting the task. You can apply powerful Visual Basic functions and data manipulation functions in the pre-task and post-task scripts.
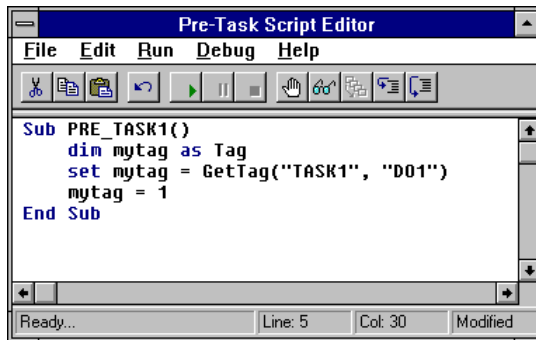
Figure 5-19   Pre-Task script panel
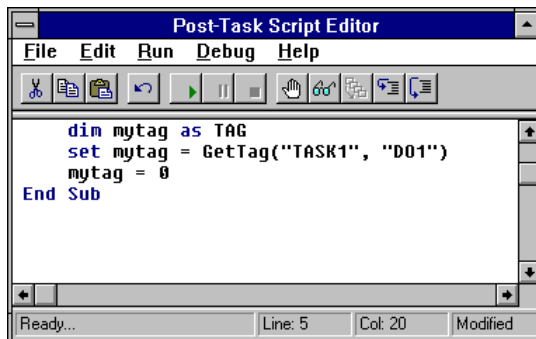
Figure 5-20   Post-Task script panel

Runtime Preference...

The Runtime Preference command is used to configure the run time properties of a task. In GeniDAQ you can specify how many errors will be allowed to occur before Runtime will stop because of excessive errors. A **Runtime error** is an error that usually occurs as a result of improper settings within GeniDAQ, or because of hardware problems. A list of Runtime errors is included in the appendix of this manual. Acceptable values for "**No. of errors allowed before stopping**" are between 0 and 32767.
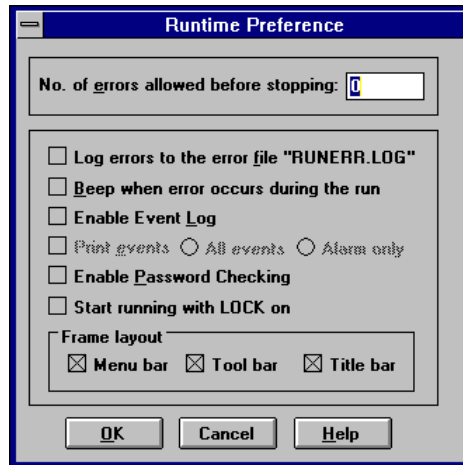
Figure 5-21   Strategy runtime preference panel

In addition to the "No. of errors allowed before stopping" item, you can enable/disable the following blocks to invoke the related functions during task execution.

Log errors to the error file "RUNERR.LOG"

If this item is enabled, runtime errors are logged to the file "RUNERR.LOG". The "RUNERR.LOG" file is created in the GeniDAQ directory. The shortcut word is "f" or "F". You can press "Alt+f" or "Alt+F" to enable/disable this option.

Beep when error occurs during the run

If the item is enabled, the GeniDAQ workstation will generate a beep sound when an error occurs during runtime. The shortcut word is "b" or "B". You can press "Alt+b" or "Alt+B" to enable/disable this option.

Enable Event Log

If this item is enabled, GeniDAQ events occurring will be written to the event log files "GENIE.ELF" (first 100 events) and "GENIE.ELH" (remaining events). Except log files, the **Event Log Viewer/Alarm Acknowledgment Dialog Box** can be displayed during runtime to reflect the event log information. GeniDAQ events consist of:

1) The start and stop time/date of the strategy

2) If in secure mode (password has been enabled by checking "Enable Password Checking"), password user ID/time of day will be logged when user logs in or out.

3) Alarm information and acknowledgment if Alarm Log Block has been connected.
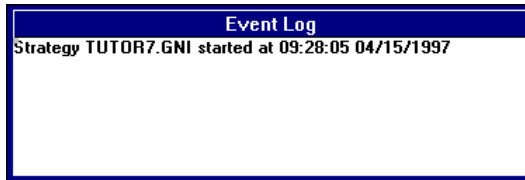
Figure 5-22   Event Log Viewer/Alarm Acknowledgment dialog box

Event Log Viewer/Alarm Acknowledgment Dialog Box

This item can be invoked by selecting Event Log from the View menu while the strategy is running. If you want to hide this dialog box, click on Event Log menu option again and the dialog box will disappear.
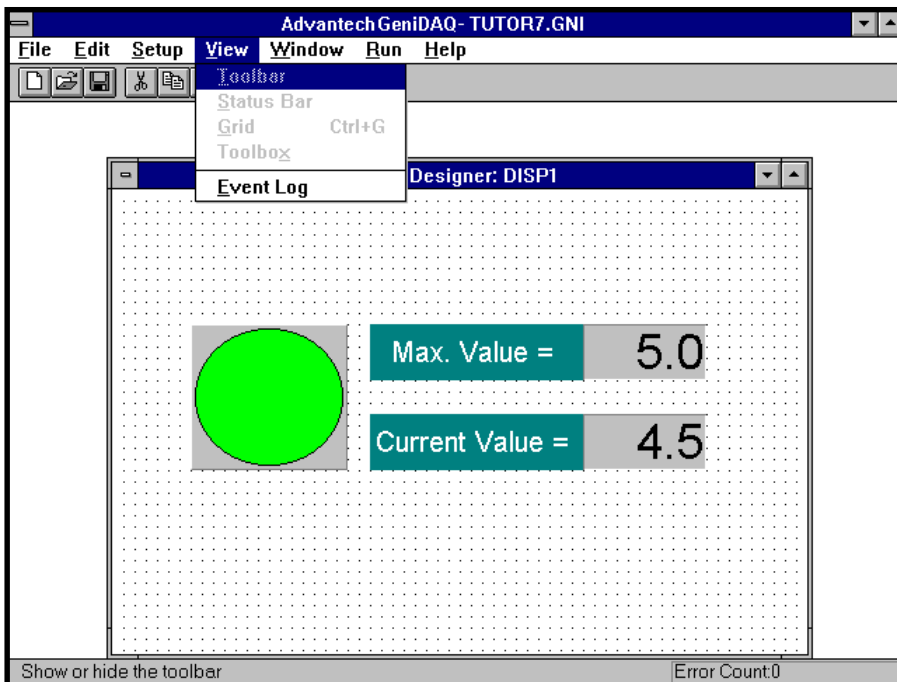
Figure 5-23   Invoke Event Log option

While the strategy is running, alarms may be acknowledged by double-clicking on the Alarm Log Event (ALOG1.HI, etc.). Alarms will be in the color red until acknowledged.

The shortcut word is "l" or "L". You can press "Alt+l" or "Alt+L" to enable/disable this option.

Print events

After you enable the "Event Log" option, this "Print events" option will be activated and you can enable/disable it. If you enable the "Print events" option, you can select to print all GeniDAQ events that occur or alarm events only. This enabled setting will output events immediately to the connected printer. This provides real-time event printing. The shortcut word is "e" or "E". You can press "Alt+e" or "Alt+E" to enable/disable this option.

Enable Password Checking

Before selecting the "Enable password checking" option, you are advised to add users and passwords in GeniDAQ system using **"Administration..."**. If "Enable Password Checking" is selected, GeniDAQ will request you to key in a user ID and password when you start to run a strategy or unlock a locked strategy while it is in runtime. The shortcut key is "p" or "P". You can press "Alt+p" or "Alt+P" to enable/disable this option.
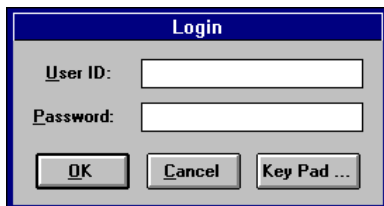
Figure 5-24   Login user ID and password

Start running with LOCK on

If this option is enabled, GeniDAQ will lock the screen when you start to run a strategy. You have to key in user ID and password to unlock it if the "Enable Password Checking" option is enabled. If the "Enable Password Checking" option is disabled, you can press "ESC" to unlock the screen. This option is used for system security. There is no shortcut for this option.

Frame layout

Frame layout option is used to set up the display layout while it is in runtime. There are three options: Menu bar, Tool bar and Title bar. You can enable/disable these three items, as you want.

Change Passwords...

After you add users via the "Administration..." option, you can change users' passwords using this option. The password is used to unlock the screen during GeniDAQ runtime. The password can contain up to 16 characters or spaces, with no constraints on format. The dialog box is shown below:
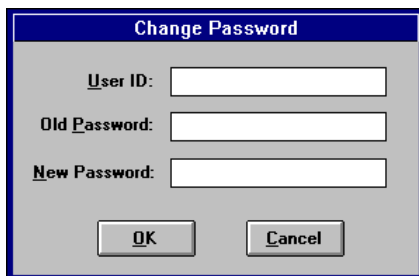
Figure 5-25   Change Password panel

Administration...

The administrator may add and delete users with the use of this dialog box for the Lock feature in Runtime. To Add and delete users:

1) Enter the Supervisor Password previously entered in the Setup/Change Password dialog box. When GeniDAQ is started for the first time, the Supervisor Password is blank. Once a Supervisor Password is entered by using the Setup/Change Password dialog Box, it remains in effect unless changed.

Figure 5-26   Enter Supervisor Password

Note: If the Supervisor Password has been inadvertently forgotten, you may start fresh by deleting the SECURITY.PW file located in the GeniDAQ directory.

2) Add each User ID and corresponding Password by pressing Add User. The user ID and password can consist of as little as no characters or spaces, for which simply the ESC key will unlock the strategy. The user ID and password can, however, contain up to 16 characters, with no constraints on format. The dialog box follows:
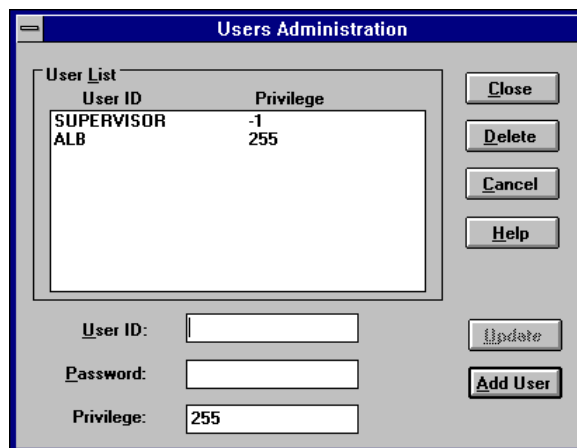
Figure 5-27   Add a new user

Add/Delete Virtual Tags

Virtual Tag is defined as an internal tag that does not link with any hardware. The Virtual Tag can be added or deleted from within Task Designer and created at the data center. After adding, Task Designer will retrieve its value from the data center by linking with TAG block to manipulate or display its value. After manipulation, the value of Virtual Tag will be saved to data center. For detailed information on Virtual Tag, please refer to *section 5.4 Virtual Tag.*

Figure 5-28   Add a Virtual Tag

Add Tool...

You can design specific User defined DLLs, install them in the GeniDAQ Task Designer toolbox and use them as a block in your strategy. The installation of a User Defined DLL is described below:

1) Click on the Setup menu and Add Tool... submenu in GeniDAQ Task Designer.

2) You will see a dialog box where you can install your User Defined DLL by specifying and pressing the Browse button to select the desired DLL file. A User defined DLL icon will now be present in the Task Designer toolbox.
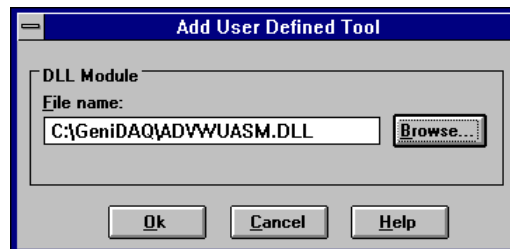
Figure 5-29   Add user defined DLL

When adding your User Defined DLL to the GeniDAQ Task Designer toolbox, GeniDAQ modifies the file named GENIE.TOL. This file defines all installed User Defined DLLs. If you wish to delete your User Defined DLL from the toolbox, simply use any ASCII editor to modify the GENIE.TOL file.

Network...

The Network I/O feature in GeniDAQ allows you to transfer data from any block over a Local Area Network (LAN) that supports Novell's IPX protocol (Novell NetWare is not required). Two blocks are used to support this feature: Network In and Network Out.

Network settings are displayed in the dialog box below:
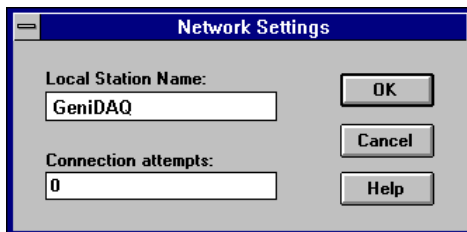
Figure 5-30   Network settings

1) Local Station Name

After making sure your network setting are as described above for each machine on your GeniDAQ Network, you must enter the Setup/Network menu to name each station. Each station on the network has to have a unique name. The result will be unpredictable if two stations on the network have the same name. Since the Station Name is stored in the strategy file (.GNI), each station should load and run a different strategy file.

2) Connection Attempts

If, after connection, a network block cannot get any data from a remote station, the network block will show time-out in the status bar. A zero (0) entry in this text box indicates that as many attempts as are necessary to establish network connection will be made.

Devices...

The Devices command allows you to enter the Device Installation utility from the Task Designer menu bar. Please refer to chapter 4, Device Installation.

Report...

Report generation is one of the main functions in a SCADA system, which records the operation of the system and presents the information in a user-defined format.

Report Designer can be activated from the GeniDAQ Task Designer and GeniDAQ Run Time modules only. Report Designer cannot be run alone. Major GeniDAQ modules must be running before Report Designer can be called. Within GeniDAQ Task Designer, users can set up report parameters, schedule and report format. No TAG point data archives and automatic report scheduling will be performed.

After opening a strategy file, users can select Report from the Setup menu. The Report Designer checks if the working directory has been set up for the current strategy file. If no working directory is set, Report Designer displays the Report Parameter Set Up Dialog Box for the user to configure the current working directory. When the current strategy's report parameters are set, Report Designer displays the report schedule dialog box. For more information, please refer to *chapter 7, Report Designer.*
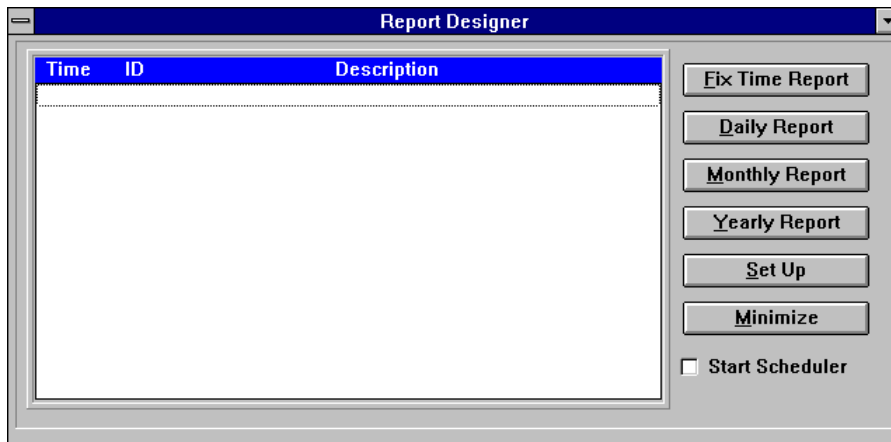
Figure 5-31   Report Schedule dialog box

5.2.4 View Menu

The View menu includes commands that enable you to zoom to fit the screen, changing the view of selected items in the GeniDAQ Strategy. The View Menu also enables the toolbar , toolbox , and status bar to be displayed when in a Task Designer Window. It also provides options to show labels under each icon block. View menu commands and their descriptions are listed below:
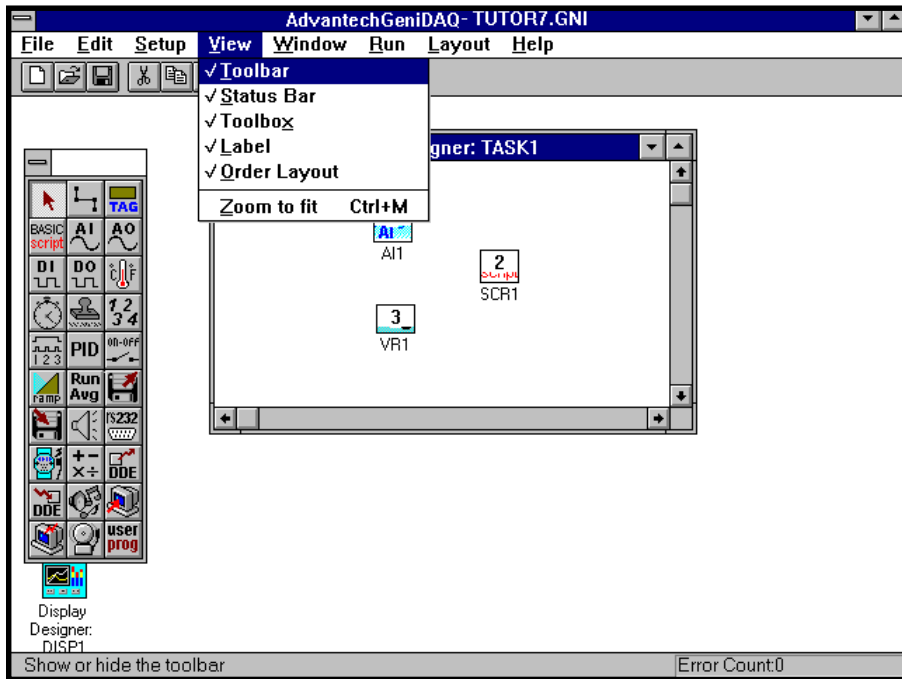
Figure 5-32   View menu commands

Toolbar

To display or hide the Toolbar at the top of the Task Designer screen, highlight this command with your mouse. The Toolbar and associated buttons allow you to perform certain menu operations more quickly than actually going through the menus. Button commands are as follows:

Figure 5-33   Task Designer toolbar

- Create a New Strategy

This button allows you to create a new strategy on the screen and discard (after saving to disk) the old one, if any.

- Open an Existing Strategy

This button opens an existing strategy file.

- Save the Current Strategy

This button saves the currently displayed strategy to disk.

- Cut the Current Selection

This button cuts the currently selected icons.

- Copy the Current Selection

This button copies the currently selected icons to the buffer.

- Insert (Paste) the Buffer Contents

This button allows you to paste the buffer contents to the upper left of the strategy editor screen.

- Print the Active Document

By pressing this button, the currently displayed strategy will be printed to the active printer.

- Display Program Information

Display program information, including version number and copyright.

- Context Sensitive help

This button changes the mouse cursor to a special help cursor that allows you to select an object and display help for selected buttons, menus, and windows.

Status Bar

To display or hide a Status bar at the bottom of the Task Designer screen, highlight this command with your mouse. The Status Bar and associated buttons are explained in the section below entitled *Task Designer Status Bar.*

| For Help, press F1 | Error Count:0 |

Figure 5-34   Task Designer status bar

Toolbox

To display or hide the Toolbox containing the icons, highlight this command with your mouse.

Figure 5-35   Task Designer toolbox

Label

To display user-defined block labels. If you choose not to show labels, your display will be somewhat less understandable. You may, however, be able to see the connection wires more clearly.

with Label   without Label

Figure 5-36   Display user-defined block labels

Order Layout

To display or hide the execution order of each block in Task Designer, highlight this command. The execution order numbers will be displayed at the upper half of block icons.
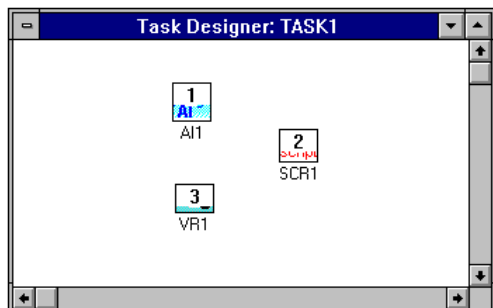
Figure 5-37   Task Designer order layout

Zoom to Fit

This option enables the task window to zoom to the right size for displaying all tag blocks.

5.2.5 Window Menu

The Window menu includes commands that enable you to arrange multiple windows within GeniDAQ. Standard window operations are supported (Cascade, Tile, and Arrange Icons). You can also use this menu to select between Task Designer windows and other windows for a chosen file.
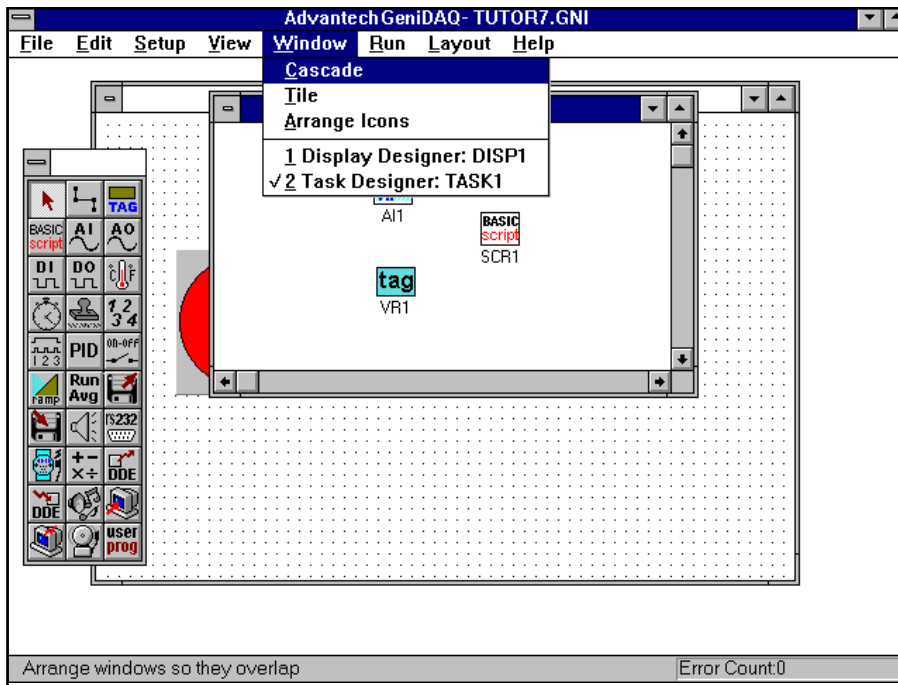
Figure 5-38   Task Designer window menu

5.2.6 Run Menu

The Run menu is actually a command that allows you to directly run GeniDAQ while you are in GeniDAQ design environment. Unlike previous versions of GeniDAQ where you had to start and then jump to GeniDAQ Runtime, GeniDAQ combines design time and run time into a complete environment. Given this, you can run GeniDAQ for testing and debugging any time when you are designing your strategy file.
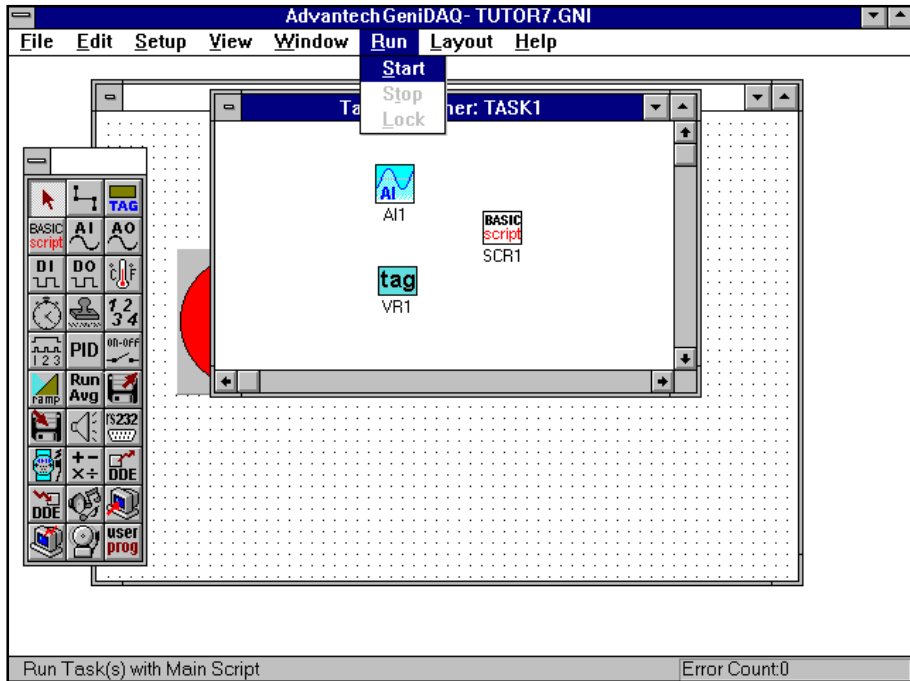
Figure 5-39   Task Designer Run menu

Start

Initiate strategy tasks using this button. After tasks start to run, this option will be disabled.

Stop

To stop running strategy tasks, select this button. This option is enabled after tasks are started.

Lock

Locks the screen to prevent unauthorized operations while GeniDAQ is in runtime. You can unlock the screen by inputting user ID when password checking is enabled. When password checking is not enabled, you can press "ESC" to unlock the screen.

.
5.2.7 Layout Menu

The Layout menu includes commands that enable you to arrange the execution order of blocks in a task. You can change the order of a limited number of blocks or rearrange the order of all blocks in a task. When complete reorder is enabled, you click once on each block to reassign an order number to each block. Order numbers will be increased from 1 to N. If you just want to exchange the order of two blocks, you can enable the Exchange Order menu option and click once on each of the blocks. The order number is sequentially increased based on the order in which you click on the blocks.
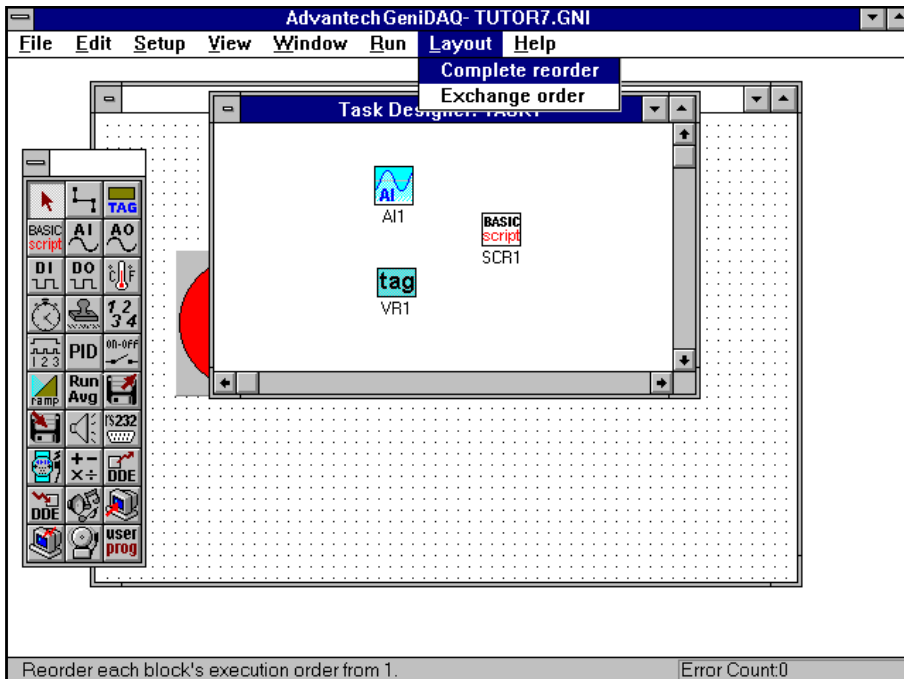
Figure 5-40   Task Designer Layout menu

Complete Reorder
The Complete Reorder option is used to change or reassign the order of all blocks. After selecting the Complete Reorder option, you can start to click the blocks. The order number is sequentially increased based on the order of your clicking on blocks.

Exchange Order
This Exchange Order menu option is used to change or reassign the order of two blocks. After enabling Exchange Reorder menu options, you can click on these two blocks. The order of these two blocks will be set based on the order in which they are selected.

5.3 Task Toolbox

Icon Blocks

This section provides a description of each of the icon blocks that are available in the Task Designer. The blocks are located in the Toolbox, which is located on the Task Designer screen.

From the Task Designer Toolbox, you have at your fingertips all icon blocks available in the GeniDAQ Task Designer, and the means to connect them together.

A description of all Toolbox items and icon blocks for use in Task Designer is included below.

5.3.1 Connection Wire

The connection wire is located in the upper right of the Toolbox. This wire is used to connect icon blocks. When this tool is selected with the mouse, the cursor changes shape from a standard pointer to a shape that resembles a roll of wire. This roll of wire is used to logically connect icon blocks. Point the end of the wire roll to an icon containing output capability, click the left mouse button, moving the wire roll to an icon containing input capability, and click again. Wires will thus be auto-routed. Routing of wires can be performed manually by clicking the mouse on the screen surface at desired right-angle wire-bend locations during connection.

Depending on which block you click on first, the flow of data through the connection wire will be different. For example, if you wish to send data **from** an Analog Input block **to** an Average block, you would click first on the AI block, then connect it to the Average block. You will notice that the end of the connection wire that is touching the Average block has an arrow that points into the Average block. This is correct, and denotes data flowing **into** the Average block from the AI block.

Some blocks will only allow data to flow one way. Notice that if you try to connect a wire between the Analog Input block and the data file block, you will get a message from GeniDAQ saying that the data file block cannot accept input. The data file block is one that will only output data, and will never accept data. **The direction of the arrow at the end of the connection wire denotes the flow of data from one block to another**.

5.3.2 Analog Input Block (AI)

This block has output capability that supplies other blocks with analog input information from the I/O device's analog input section. Double click on the AI block and an AI configuration window will be displayed for setting parameters of analog input.

Figure 5-41   Set analog input parameters
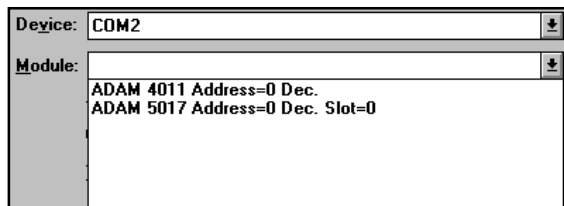
Device field

When you press the down arrows besides the Device field, all I/O devices currently installed that have available analog inputs are displayed in an associated dialog box. When a device is selected, all analog input channels for the device are displayed below it.

Figure 5-42   Analog input device settings

If you select COM port devices (for example, ADAM-4000 and ADM-5000 modules), the module field will be displayed below the device field to select the connected module. You are advised to input the correct address and channel settings when installing the device.
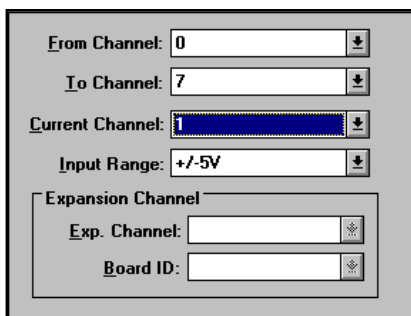
Figure 5-43   Select connected AI module

Channel field

After selecting the device or module, you have to specify the hardware's analog input channels to be connected to a particular AI block. An AI block can link up to 16 channels depending on the selected hardware devices. For example, if you select an ADAM-5018 module, GeniDAQ can pull all 7 channels' data at one time. In this channel field, you have to set the following fields:

Figure 5-44   AI block channel setting

From channel:
Specifies the starting polling channel. The value ranges from zero (0) to the maximum number of hardware channels.

To channel:
Specifies the ending polling channel. The value should be equal to or larger than the From Channel number.

Input Range:
Specifies the input value range of each hardware channel, for example -5 V to +5 V. This value is dependent on the hardware device. If there are multiple channels for a connected hardware device, you are advised to specify each channel's input range.

Expansion channel:
Specifies the expanded daughterboard on a connected mainboard. Use this feature to select the correct expansion channel and board ID.

122

Update rate field

The Update rate is a divisor that allows the Analog Input block to have a different effective scan rate than the rest of the strategy. This is useful if, for example, your task is running at 100 Hz, but you only want to sample at a rate of 20 Hz. For this example, you would set the Update Rate to 5 (100 divided by 5 (the update rate) gives an effective scan rate of 20 Hz). In this scenario, you will still get 100 Hz data if you send the output to a Log File block or a display block, but only one in five samples will be "real". The other samples are merely copies of the "real" sample.

Valid values for the update rate are between 1 and 32,767.

Figure 5-45   AI block update rate setting

Scaling button

Scaling button is used to re-scale input value range to the desired value range. For example, if the input range is -5 to +5, then you can adjust it to be 0 to 10.
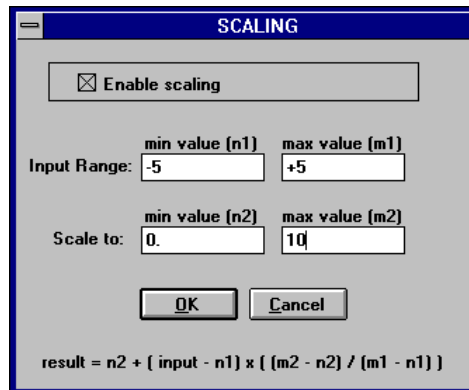
Figure 5-46   AI block scaling setting

Wiring In: GeniDAQ will display a window with the error message "Cannot accept input".

Wiring Out: Multiple channels can be selected to output data to another connected block. The maximum number of channels is 16.

5.3.3 Analog Output Block (AO)

AO1

This block has input capability that accepts another block's analog data, then forwards the data to the selected I/O device's analog output section.
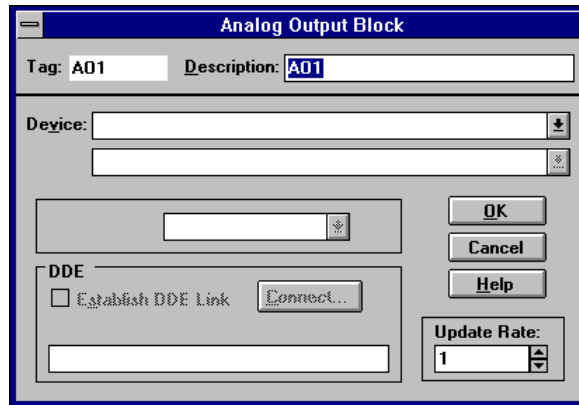
Figure 5-47   Analog Output block

Device field

When this block is selected, all I/O devices currently installed that have available analog outputs are displayed in a dialog box. For each I/O device displayed, all available analog output channels for the device are also displayed. You choose the hardware's analog output channel to be connected to a particular AO block (one AO block per I/O device AO channel) by clicking on the desired channel. Please refer to AI block Device field.

Channel field

This field is used to specify the hardware device channel that AO block will output data to. The value of channel field will depend on the connected hardware device.

Figure 5-48   AO block channel setting

Update rate field

The Update rate is a divisor that allows the Analog Output block to have a different effective scan rate than the rest of the strategy. This is useful if, for example, your strategy is running at 100 Hz, but you only want to output data at a rate of 20 Hz. For this example, you would set the Update Rate to 5 (100 divided by 5 (the update rate) gives an effective scan rate of 20 Hz).

DDE field

The Analog Output block has DDE capabilities, which allow it to exchange data with other Windows applications. Through DDE, other windows applications can exchange data with AO block and/or output data to hardware device(s) directly. For more information about DDE, see "DDE Blocks (Client and Server)" below.

Figure 5-49   AO block DDE setting

Wiring In: Another block provides output data to the AO block and connected hardware device(s). There is only one input source. If an input source already exists, "The input already connected" message will be displayed.

Wiring Out: Passes the input data to a connected block directly.

5.3.4 DDE Blocks (Client and Server)

DDE, or Dynamic Data Exchange, is a standard windows feature included in GeniDAQ. DDE allows you to exchange data between GeniDAQ and other Windows applications.

There are two DDE blocks included in GeniDAQ, known as DDE Server, and DDE Client. In addition, there are four blocks that have DDE capability built into them. These include the Analog Output, Digital Input, Digital Output, and Temperature Measurement blocks. For these four blocks, DDE capability is accessed through the dialog box that appears when you setup the blocks.

The DDE Server block provides data *from* GeniDAQ *to* another Windows application**.**   Since DDE uses a broadcast-type communication, the DDE server will "publish" its data, and other applications have the responsibility to find it and use it as they wish. When a block is connected to the DDE server block, a link is established. Double clicking on the DDE server block will tell you the name of the **Service**, **Topic** and **Item**. This information is what the other applications will need to locate a link that you create in GeniDAQ.
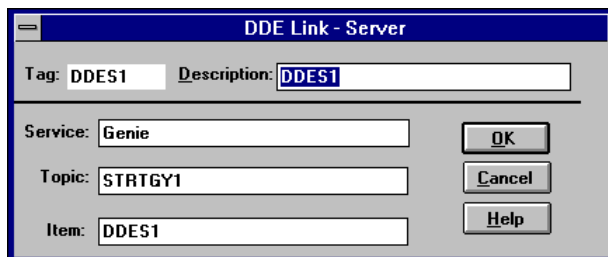
Figure 5-50   DDE server block setting

The **Service**, in this case, is GeniDAQ. It is the **program name** of the Windows application that is providing the data.

The **Topic** is the **name of the particular file** that is providing the data. In GeniDAQ, it would be the name of your particular strategy (*.GNI). The extension (.GNI) is not needed here, only the name of the strategy. For example, if your strategy were named DDEDEMO.GNI, the topic would be **DDEDEMO**.   Keep in mind that an untitled strategy will not be able to provide data through DDE; it must be a unique, saved filename.

The **Item** is the **tagname** of the block that is providing the data (DDES1, DDES2, DDEC1, etc.). Other valid tagnames are AI1, DO3, etc. (first Analog input block, third digital output block, third elapsed timer block, etc.). These I/O blocks have DDE capability built-in to their dialog boxes. It is important to realize that these are the **tagnames** that GeniDAQ assigns to the particular blocks, and not the **description** that the user can change.

These three fields are what the other Windows applications will be looking for. Usually, most other Windows applications will separate the fields in the following manner:

    Service|Topic!Item

Note that this is *not* a universal standard, only common practice. You should refer to the documentation for your particular Windows application to learn how DDE syntax is implemented for each program.

Wiring In: Accepts the input data and pass this data to other windows DDE client applications. One data source is allowed. If there is more than one input, "The input already connected" message will be displayed.

Wiring Out: Passes the input data to a connected block directly.

DDEC1

The DDE Client block receives data from another Windows application. This block will require that you input the Service, Topic, and Item for the application that will be providing data to GeniDAQ. It may also be necessary for other Windows applications to be set up to publish data. You should refer to documentation provided with the application for this information.
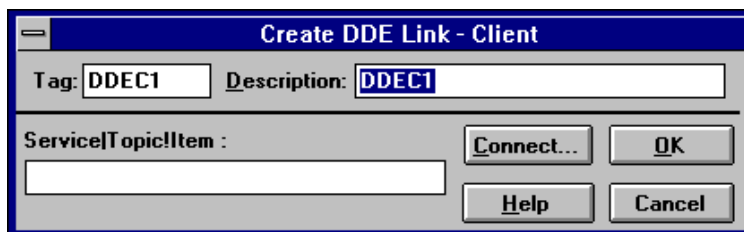
Figure 5-51   DDE Client block

The DDE client block will receive data from another application, then input that data to the blocks that are connected to the DDE client block via the connection wire. Any number of blocks can be connected to a DDE client block, and they can all use the data that the DDE client block is receiving from the server application.

To setup a DDE client block, your other application should be running in the background. Place the DDE client block on the Task Designer workpad, and connect it to the blocks to which you wish to input data (from the DDE client block or from another application). Next, double click on the DDE client block. This will invoke the DDE client dialog box. Within this dialog box, click on the **Connect...** button. In the dialog box that appears (Create DDE Link), you should see your application name in the box under **Service**. All of the applications that are currently running which have DDE capabilities should be displayed in the **Service** box. It is possible that some applications will not be shown. If the other applications are listed, click once on the Service that you wish to use, which will then place the available **Topics** in the next box under *Service Topics*. Click once on the appropriate topic, which will then place the available **Items** in the box under *Topic Items*. Clicking on the **OK** button after going through this sequence will place the service, topic and item in the format

```
Service|Topic!Item
```

in the *Create DDE Link* dialog box. The link is then created and you can receive data from the other application.
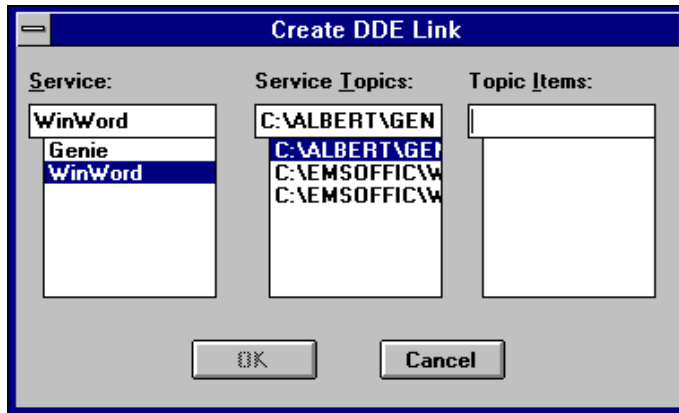
Figure 5-52   Create DDE links

If your other application's Service, Topic and Item is not shown in the list boxes and you want to exchange data with them, you must manually enter the Service, Topic and Item in the appropriate Text Box within the DDE Client dialog box.

The DDE link can still be used after saving your strategy and restarting later. If the server application is not running the next time that you invoke the strategy, GeniDAQ will ask you if you would like to start it. For this to take place, the server application must be explicitly mentioned in the PATH= statement in your AUTOEXEC.BAT file. If not, GeniDAQ will not be able to start the server application.

Wiring In: GeniDAQ will display a window with the error message "Cannot accept input".

Wiring Out: Passes the input data from another DDE server Windows application to a connected block.

5.3.5 Digital Input Block (DI)

This block has output capability that supplies other blocks with digital input information from the I/O device's digital input section.
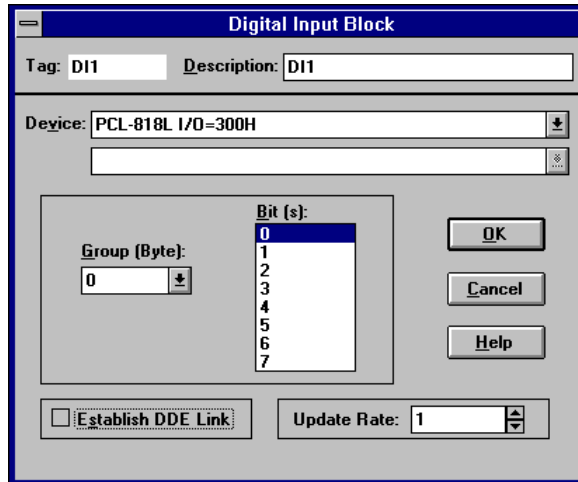
Figure 5-53   Digital Input block

Device field

When this block is selected, all I/O devices currently installed that have available digital inputs are displayed in an associated dialog box. For each I/O device displayed, all available digital input channels for the device are also displayed. You choose the hardware's digital input channel (bit) or channels (up to one byte of packed I/O) to be connected to a particular DI block by clicking on the desired device's channel(s) or bit(s).

Group(Byte)/Bit field

After selecting a hardware device, you have to select the DI block group or bits. The group and bit number are dependent on the hardware device. GeniDAQ will list the available group and bit number for your selection.

Update rate field

The Update rate is a divisor that allows the Digital Input block to have a different effective scan rate than the rest of the strategy. This is useful if, for example, your strategy is running at 100 Hz, but you only want to sample at a rate of 20 Hz. For this example, you would set the Update Rate to 5 (100 divided by 5 (the update rate) gives an effective scan rate of 20 Hz).

Establish DDE Link field

The Digital Input block also has DDE capabilities which allow it to exchange data with other Windows applications. For more information about DDE, see "DDE Blocks (Client and Server)" elsewhere in this section. Please refer to DDE Client/Server block.

Wiring In: GeniDAQ will display a window with the error message "Cannot accept input".

Wiring Out: Passes the hardware device data to a connected block directly.

5.3.6 Digital Output Block (DO)

This block has input capability that accepts another block's digital data and forwards the data to the selected I/O device's digital output section.
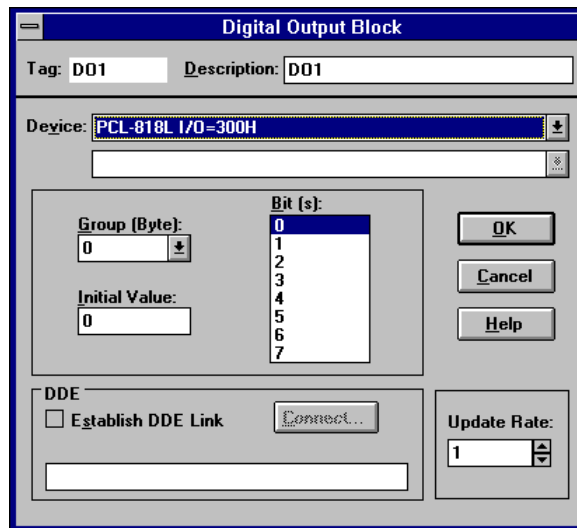
Figure 5-54   Digital Output block

Device field

When this block is selected, all I/O devices currently installed that have available digital outputs are displayed in an associated dialog box. For each I/O device displayed, all available digital output channels (bits) for the device are also displayed. You choose the hardware's digital output channel(s) and group (byte) to be connected to this DO block by clicking on the desired device's channel(s) or bit(s).

Group(Byte)/Bit/Initial field

After selecting a hardware device, you have to select the DO block group or bits. The group and bit number are dependent on hardware device. GeniDAQ will list the available group and bit number for your selection. For each output bit, you can assign the initial value. For example, assign 0 as the initial value.

Update rate field

The Update rate is a divisor that allows the Digital Output block to have a different effective scan rate than the rest of the strategy. This is useful if, for example, your strategy is running at 100 Hz, but you only want to output data at a rate of 20 Hz. For this example, you would set the Update Rate to 5 (100 divided by 5 (the update rate) gives an effective scan rate of 20 Hz).

Establish DDE Link field

The Digital Output block also has DDE capabilities, which allow it to exchange data with other Windows applications. For more information about DDE, see "DDE Blocks (Client and Server)" elsewhere in this section.

Wiring In: Another block directly outputs data to DO block and connected hardware device(s). There is only one input. If there is more than one input, "The input already connected" message will be displayed.

Wiring Out: Passes the hardware device data to connected block(s) directly.

5.3.7 Hardware Event Counter/Frequency Measurement/Pulse Output Block

CTFQ1

This block has input and output capability that supplies other blocks with Event Counter, Frequency Measurement, or Pulse Output information from the I/O device's counter/timer section.

Figure 5-55   Adjust Counter/Timer settings

Device/Channel field

When this block is double-clicked upon, all I/O devices currently installed that have available counter/timer channels are displayed in an associated dialog box. When a device is selected, all counter/timer channels for the device are displayed below it. You choose the hardware's counter/timer channel to be connected to a particular Counter/Frequency/ Pulse Output block by clicking on the desired device's channel.

Counter Input/Frequency Measurement Mode

A hardware device's counter/timer chip is used as a **hardware event or frequency counter** that counts digital rising edges (digital high events) from an external source that supplies digital information (TTL or other level 1's and 0's). The block may be used to control the output of the timer/counter chip (when supported by the driver), to be used as a **pulse generator**. The block's output can be sent to another block, such as the display block. All counting or pulse output is performed by the I/O card's timer/counter chip, and is not dependent on the sample rate of the strategy. The event counter is implemented as an **up - counter**; that is, the count starts at 0 (zero) and counts until it reaches the counter's maximum value (hardware/driver dependent). A **down - counter** may be constructed by using this block in conjunction with a User Program Block or Single Operation Calculation (SOC) block and subtracting the counter output from the maximum count. In this way, the count will start at the maximum and will end at zero.

If the **Start/Stop from** is connected, the counter or pulse generator is started and stopped by this input. This allows for total control over counter operation during runtime. If the start/stop input is not connected, the counter will start when the strategy begins and stop when it ends. To **start** the counter using the start/stop input, provide a rising edge (from low (zero) to high) to this input from another block. This will start the hardware counter from zero. Counter status is output from the counter/frequency/pulse output block with every scan of GeniDAQ, unless the divisor (explained below) is set to a value other than 1 (one). By applying a falling edge (from high to low) digital value to the start/stop input, the counter may be **stopped** at the current value.

By connecting a rising edge value to the block's **Reset from**, the count can be reset to its starting value and counting is resumed. A falling edge value to the reset input will have no effect on counting (reset input not connected is treated as a low).

Pulse Output Mode

You can create a pulse generator output from the counter/timer section of the I/O device if supported by the DLL driver. To do so, specify the **Total Pulse** and **first 1/2 cycle time** (in seconds). You can use the static values specified in the dialog box or supply the block with floating point values on its input from another block. Not all I/O devices support a varying first 1/2 cycle due to hardware limitations of the counter/timer chip. In this case, when you specify a total period the device will generate a 50% duty cycle square wave on its output pin. Check the DLL driver on-line help for specifications on hardware support.

If the **Total Pulse Time from** is connected to a block that supplies floating point output, such as the User Programmable Block, the total pulse time (1/frequency) of the pulse output is controlled by this input. This allows for control over total pulse frequency operation during runtime. If the Total Period input is not connected, the counter will use the static values entered in the dialog box when the strategy begins.

If the **First 1/2 Cycle Time from** is connected to a block that supplies floating point output such as the User Programmable Block, the first 1/2 cycle time (1/frequency) of the pulse output is controlled by this input. This allows for total control over first 1/2 cycle pulse frequency operation during runtime. If the first 1/2 cycle input is not connected, the counter will use the static values entered in the dialog box when the strategy begins.

Gate Mode: External Gating

This field is enabled when you select Counter Input mode. The counter/timer chip may be started and stopped via external level control on a special input pin. If the hardware timer/counter and DLL driver supports external hardware gating, setting this value to high level or low level will enable the timer/counter chip to allow for gating control. Thus, if the counter is started at either the strategy start or using the start/stop input, the actual counting can't take place until the external gate senses the proper level on its input pin.

Update rate field

The **Update rate** is a divisor that allows the Counter/Frequency/Pulse Output block to have a different effective scan rate than the rest of the strategy. This is useful if, for example, your strategy is running at 20 Hz, but you only want to sample the counter at a rate of 4 Hz. For this example, you would set the Update Rate to 5 (20 divided by 5 (the update rate) gives an effective scan rate of 4 Hz). In this scenario, you will still get 20 Hz data if you send the output to a Log File block or a display block, but only one in five samples will be "real". The other samples are merely copies of the "real" sample. Valid values for the update rate are between 1 and 32767.

Note: A separate Counter/Frequency/Pulse Output block is needed for each individual hardware counter/timer channel.

Wiring In: There are four input items for this block: Start/Stop Input, Reset Input, Total Period Input and First 1/2 Cycle Input. For each wiring input, you have to select one of the above.
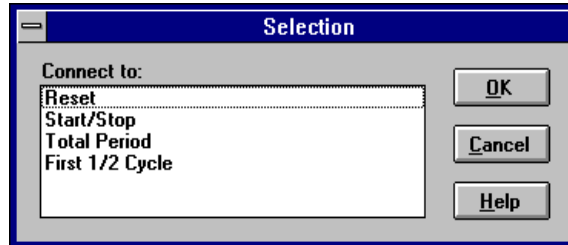
Figure 5-56   CTFQ block input connection

Wiring Out: Passes the counter/frequency/pulse data to another connected block**.**

5.3.8 Hardware Alarm Block

ALM1

This block has output capability that supplies other blocks with hardware alarm status from the I/O device's alarm section.
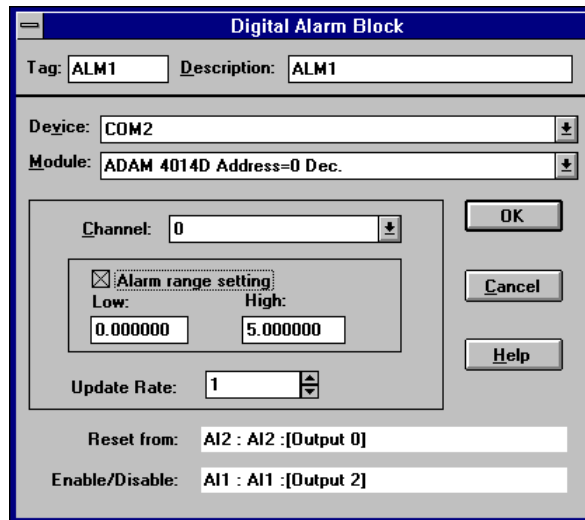
Figure 5-57   Hardware alarm block

Device field
When this block is selected, all I/O devices currently installed that have available hardware alarm channels are displayed in an associated dialog box. When a device is selected, all alarm channels for the device are displayed below it. You choose the hardware's alarm channel to be connected to a particular Hardware Alarm block by clicking on the desired device's channel. Please make sure the hardware alarm is enabled when the device is installed ("Alarm enabled" field is YES).
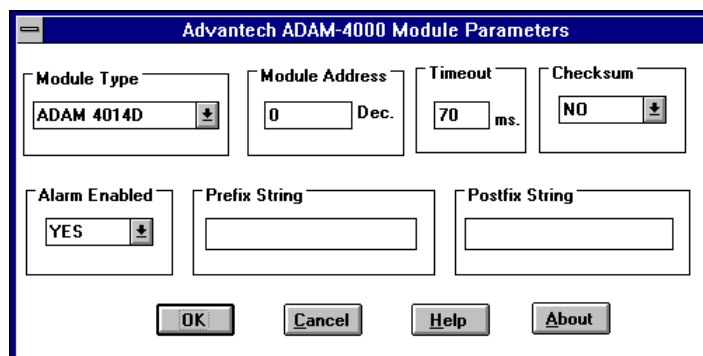
Figure 5-58   ADAM-4014D alarm enabled

After selecting the hardware alarm channel, you can set the alarm range (high and low values).

Enable/Disable option

If the **Enable/Disable option** is connected, the alarm is enabled and disabled using this input. This allows for total control over alarm operation during runtime. If the enable/disable input is not connected, the alarm will be enabled when the strategy begins and disabled when it ends. To **enable** the alarm using the enable/disable input, provide a rising edge (from low (zero) to high) to this input from another block. Alarm status is output from the hardware alarm block with every scan of GeniDAQ, unless the divisor (explained below) is set to a value other than 1 (one). By applying a falling edge (from high to low) digital value to the enable/disable input, the alarm may be **disabled**.

Reset from option

By connecting a rising edge value to the block's **Reset from**, the alarm is reset to zero and alarm operation is resumed. If the option is not enabled, a falling edge will be treated as a low.

Update rate field

The **Update rate** is a divisor that allows the Alarm block to have a different effective scan rate than the rest of the strategy. This is useful if, for example, your strategy is running at 20 Hz, but you only want to sample the alarm status at a rate of 4 Hz. For this example, you would set the Update Rate to 5 (20 divided by 5 (the update rate) gives an effective scan rate of 4 Hz). In this scenario, you will still get 20 Hz. data if you send the output to a Log File block or a display block, but only one in five samples will be "real". The other samples are merely copies of the "real" sample. Valid values for the update rate are between 1 and 32767.

Wiring In: Hardware Alarm Block accepts the input data as reset or enable/disable and displays the information in the configuration dialog box.
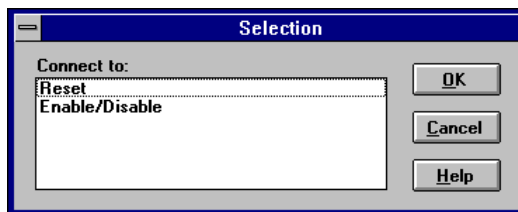
Figure 5-59   Hardware alarm connection selection

Wiring Out: The Hardware Alarm block outputs one of three levels:

      0 = No Alarm has occurred
      1 = Low Alarm is now set
      2 = High Alarm is now set

5.3.9 Event Counter Block

CNT1

This block has both input and output capability. A software event counter that counts digital rising edges (digital high events) from any block supplying digital information (1's and 0's). The block's output can be sent to another block. Each count is performed with each scan; therefore counting speed is equal to the sample period of the system. By connecting a high digital value to the reset input, the count can be reset to its starting value and counting is stopped. A low value to the reset input will enable counting. If reset input is not connected, this value is simply treated as a low. By applying a high digital value to the hold input, the count may be temporarily held at the current value. A low on the hold input will resume counting.
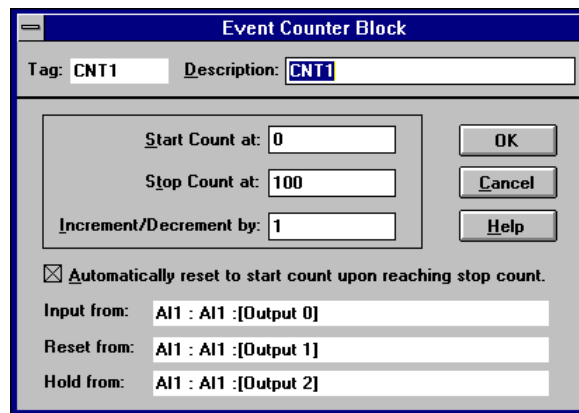
Figure 5-60   Event counter block

Start Value
An integer value at which the Counter will start. (maximum 65535)

Stop Value
An integer value at which the Counter will finish. The value can be above or below the Start value. (maximum 65535)

Increment/Decrement
Each integer up/down count will be equal to this value.

Input from
The block from which you would like to count pulses (rising edges).

Reset from
The trigger block (any digital type block) used to apply a digital high/low to reset/start the counter. A digital high applied to this input resets the counter and stops counting. A digital low allows counting to occur. No reset input is treated as a low input; counting will free-run in this case.

Hold from
Any digital type block used to hold the count at the current value. A high digital signal applied to this input stops counting at the current value. A low applied to this input will enable/resume counting. No hold input is treated as a low input.

Wiring In: Event Counter Block accepts data as input, reset and/or and then displays the data in the configuration dialog box.
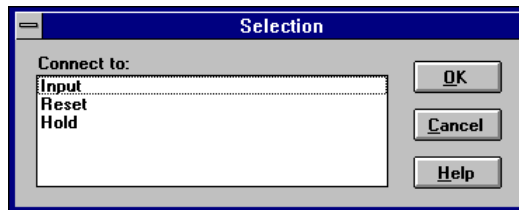
Figure 5-61   Event Counter connection selection

Wiring Out: Event Counter Block outputs the current counter value to connected block(s).

5.3.10 Data File Block

This block has output capability. By using this block, data can be retrieved from a file. The data will be retrieved one line at a time with each system scan (sample). When the end of the data is reached, it will be re-scanned (loop back will occur). An ASCII file must be created containing the data formatted into one column of either integer or floating point values. Specify the filename in the associated dialog box. Connect the Data file block to the block to where data will be sent.
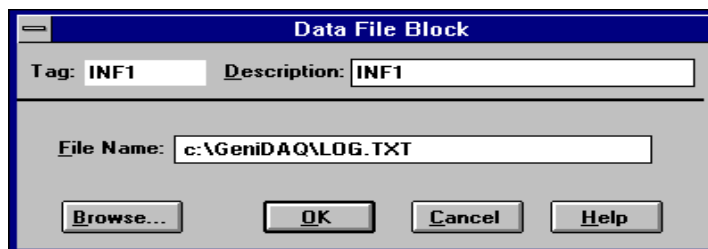
Figure 5-62   Data file block

138

Wiring In: GeniDAQ will display a window with the error message "Cannot accept input".

Wiring Out: Data File Block will pass the value from data file to connected block.

5.3.11 Log File Block

This block has input capability (8 inputs maximum). The block allows data from inputs to be logged to a file in multiple column format. Each block from which data is to be logged corresponds with one column in the file.
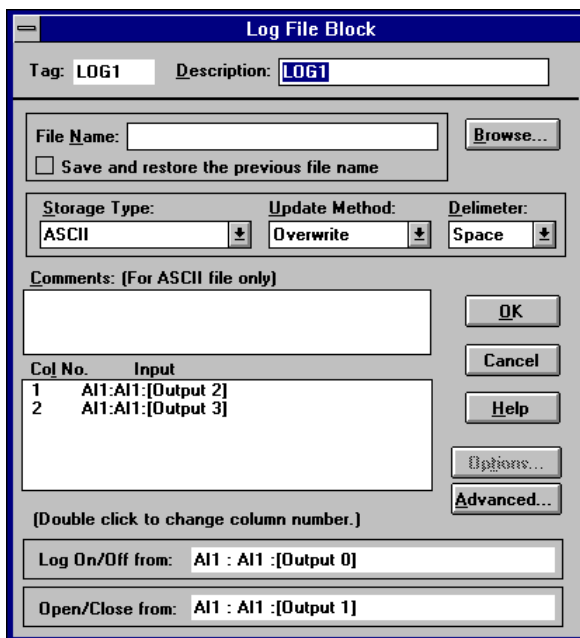
Figure 5-63   Log file block

The **Save and restore the previous file name** option, if checked, let you save the last used file name when you stop the run-time. A new file name follows the saved file name will become the file name at the next run. This option is useful only when the file name contains wildcard characters. Note the filename will be saved only when you stop the running session prorerly.

You can specify the **storage type** (format) for the file as ASCII, binary float, byte, integer, or long integer.
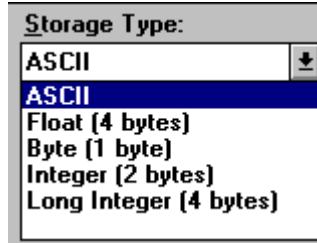
Figure 5-64   Select data storage type

The **update method** can be either to append new data or to overwrite old data.
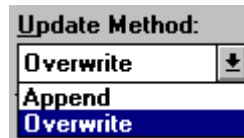
Figure 5-65   Select file update method

The delimiter between data in log file can be space, comma or tab.
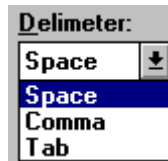
Figure 5-66   Select the delimiter between data

You can include a **header** and **comments** at the beginning of your file if you desire. You can specify the width, in characters, of each column of data, along with decimal point placement. For each input to be logged, you can specify to which column number the input block's data will be sent. When the file log block is selected, all input block tag names are displayed with the default column number starting at one (1).

You can change the column numbers by double clicking on each input tagname so that a dash (—) is displayed instead of the column number. Double clicking again on each input's tagname, starting at the one you want to be logged as column one (1) in the file, will cause them to be numbered the way you want. By pressing the **Options** button, you can specify each column's width (in characters), desired header information and decimal point format. For example, if the format is specified as "0.000", the output would be displayed as "9.876. If you specify the output as "0.0", the output value would be 9.8.
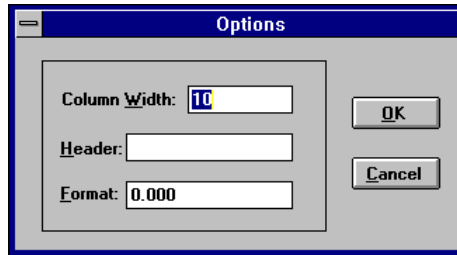
Figure 5-67   Log file information

The "**Log On/Off**" input, if connected, is used to control the logging of the input data. A nonzero input allows input data to be logged, while a zero input suspends the data logging process. You can use this to select the right data to be logged or to take one of every N readings.

The "**Open/Close**" input, if connected, is used to control the opening and closing of the log file. This option only works with a wildcard (*) file name, since opening a file will trigger the block to use the next file name based on the wildcard characters. The **Advanced** button will allow you to set up a total of 16 possible combinations for opening and closing the file (4 options for opening and 4 options for closing).
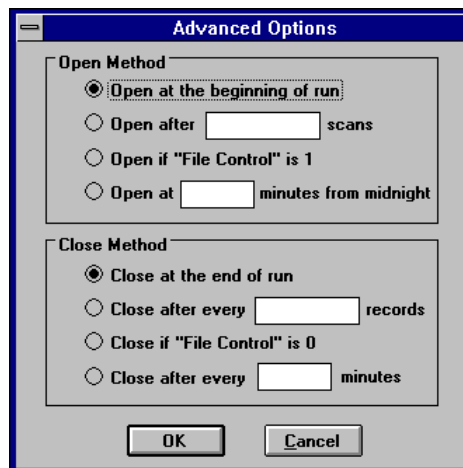
Figure 5-68   Log file open/close methods

The four **OPEN** options are as follows:

A. **Open at the beginning of the run**: The file is opened immediately when the run is started.

B. **Open after every N scans**: GeniDAQ waits until N scans have passed to open the file.

C. **Open if "File Control" is 1**: "File Control" is a control signal to the Log File Block from another block. The file is opened when this signal is 1. The file is closed when this signal is 0.

D. **Open at N minutes from midnight**: This option is good for doing a job that is repeated daily at the same starting time. This allows the file to be opened at N minutes from midnight for the first file-open. The subsequent files are controlled by file-close options.

The four **CLOSE** options are as follows:

A. **Close at the end of the run**: This is the same as the original file-close of version 1.0. The file is closed immediately after the run is stopped.

B. **Close after every N records**: GeniDAQ waits until N records have been recorded to close the file.

C. **Close if "File Control" is 0**: "File Control" is a control signal to the Log File Block from another block. The file is closed when the signal becomes 0. The next file is opened when the signal becomes 1 again.

D. **Close after every N minutes**: Starting from the moment the first file is opened, this option allows the file to be closed every N minutes. The next file is opened right after the previous file is closed, if needed.

In order to generate a different filename each time a file is created, some of the OPEN/CLOSE options listed above require wildcard character(s) in the filename. The only wildcard character supported is the # (pound sign). The "#" character can appear at any position in the standard DOS file name in the place of a regular character. It will be replaced by a digit (0-9) to form a real file name. Use more "#" characters together if you need a larger number of files. A wildcard such as "FILE###.LOG" can generate 1000 file names from "FILE000.LOG" to "FILE999.LOG". Make sure there are enough digits there for your strategy. All wildcard characters must be together (contiguous).

Valid filename examples using wildcards:

```
MYFILE##.LOG        MY###R.LOG          MYFILE.###
MYFILE.L##          ########.TXT        #RECORD.TXT
```

Invalid filename examples using wildcards:

```
MY#FILE#.LOG        # sign should be adjacent to each other
FILE##.##T          # sign should be adjacent to each other
MYFILE###.TXT       File name too long, should be in 8.3 format
```

Wiring In: Log File Block accepts input data as logged data, open/close control and data logging on/off. This data is displayed in the block configuration dialog box.
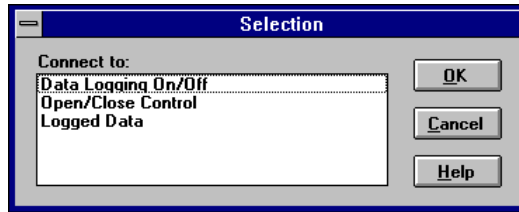
Figure 5-69   Log file connection selection

Wiring Out: No output available

5.3.12 Average Block

This block allows for input and output. There are two averaging methods available. If "**Moving Average**" method is selected, the input is averaged over a number of samples. The moving average is defined in the "**Number of points to be averaged**" field. If "**Whole Average**" method is used then the output is the average of all samples. For instance, if it is desired to average the present sample value with the previous 9 samples, then the user chooses the moving average method and uses 10 as the number of samples to be averaged. This will tend to smooth out noise and any extraneous signals. The Moving Average block should be placed between the sampling block and the next logical block, i.e. the display block, etc.
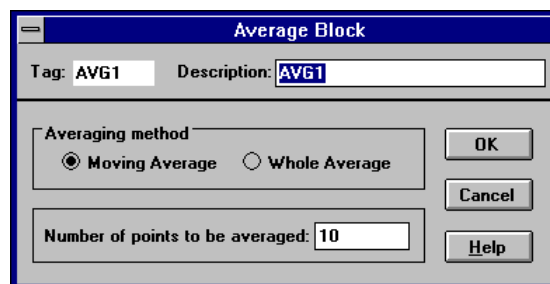
Figure 5-70   Average block

Wiring In: Average Block accepts the wiring input data as sampled data and does calculation.

Wiring Out: Average Block outputs the average value to the connected block.

5.3.13 On/Off Control Block

ONF1

This block allows both input and output. The input consists of a measured value (feedback) to be controlled to within a certain tolerance (deadband), determined by a setpoint (either dynamic or static). The output is either digital low or high, depending on controller output.

Theory:
Non-proportional control, in which the controlled process input is either fully ON or fully OFF, depends on whether the measured value (feedback) is above or below the control point (setpoint) deadband.
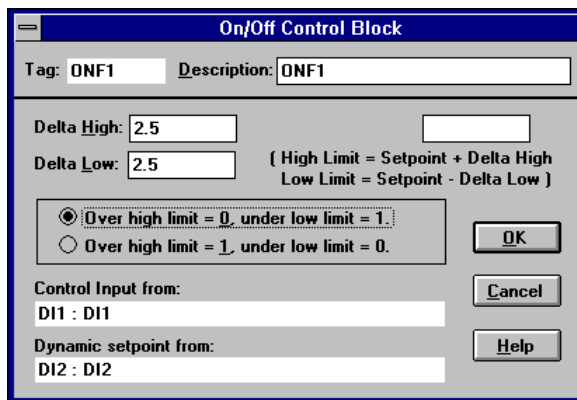
Figure 5-71   On/Off control block

Setpoint
The On/Off setpoint value, or desired value. This value can be changed on the fly (dynamic) by specifying a ramp or other block to be used as the dynamic setpoint. If there is no dynamic setpoint from connection, a constant setpoint will be enabled. You can enter the static setpoint.

Delta Low
Low value for generation of the deadband. Lower section of deadband is equal to Setpoint - Delta Low.

Delta High
High value for generation of the deadband. Upper section of the deadband is equal to Setpoint + Delta High.

144

In addition, you have the option of outputting either a 0 or a 1 value for inputs over the high limit, which would make the output 1 or 0, respectively, for inputs under the low limit.

If no deadband is desired, simply set the delta low and delta high values to zero. In this way, ALARM CONTROL can be achieved using a static setpoint.

Wiring In: On/Off Control Block accepts data as input data or setpoint data.
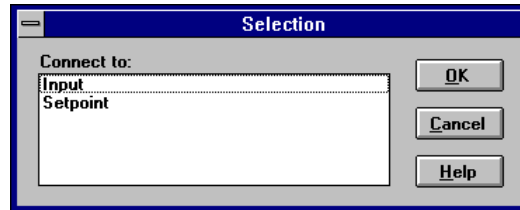
Figure 5-72   On/Off connection selection

Wiring Out: On/Off Control Block outputs either digital low or high to connected block.

5.3.14 PID Control Block

This block allows for input and output. The input consists of a measured value (feedback) to be controlled by the setpoint value (either dynamic or static). The block's output is the controller output.

Theory:
A controller is capable of receiving a signal from a sensor (usually temperature) within a process and regulating an input to that process in order to maintain a selected value or set point/control point.

The PID controller is the most widely used type of process controller. It is the ability to tune its control action to specific time constants and therefore to deal with process changes over time. These features have earned the PID controller wide acceptance. To perform the control, the object is to measure the difference (error) between the desired value (setpoint) and the measured value (feedback), and reduce that error. The PID controller is the most efficient type of process controller.

The control functions of the PID controller may be separated according to application requirements into three types. The three functions are: one, two, and three mode control.

One Mode Control (Proportional)
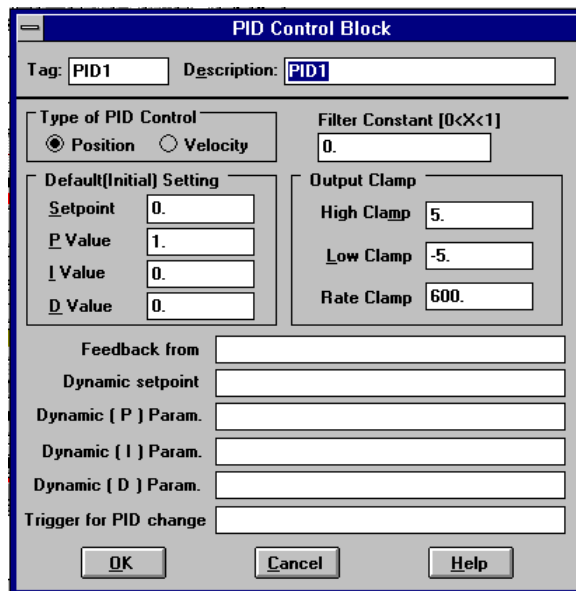
This is the simplest type of proportional control. Using this method, the controlled process input is regulated to a value proportional to the difference between the control setpoint and the measured value. This controls the proportional band. The larger the value of the proportional constant, the harder the system will react to differences between the setpoint and the actual measured value. A simple proportional controller may be achieved by setting the reset rate (I) and derivative times (D) to zero.

Two Mode Control (Proportional/Integral)

This method introduces an additional reset control action depending upon the accumulated error x time product (integral). The sum of the error and reset signals continuously act to make the actual error signal smaller, stopping when the measured value reaches the desired control setpoint. A PI controller is created by setting the derivative (D) value to zero.

Three Mode Control (Proportional/Integral/Derivative)

This mode uses an additional rate sensing (derivative) action, which reduces the tendency to over-shoot a control setpoint. This is done by anticipating the approach of a zero value error signal and initiating a control response reversal before the measured value (usually sensed temperature) actually gets there. Proper use of the derivative term can result in much faster process response.

Figure 5-73   PID Control block

Type of PID control

The output of the position method is an absolute value for setting the controlled variable. Since the output of the velocity method is a relative amount to change the controlled variable. For example, if the current value of the controlled variable is 5, and the output of the PID position algorithm is 2, then the controlled variable is set to 2. If the current value of the controlled variable is 5, and the output of the PID velocity algorithm is 2, then the controlled variable should be set to 5+2=7.

P — PID proportional control constant

I — PID integral mode gain constant

D — PID derivative mode gain constant

The P, I, D parameters can be set on the fly (dynamic) by specifying data from other blocks as the dynamic inputs. The change of any of the P, I, D parameters is not effective until the Trigger for PID change is triggered.

Trigger for PID change — If you use dynamic P, I, D parameters, you have to connect a value from another block to the input (Trigger for PID change) to activate the PID changes. In other words, dynamic P, I, D and Trigger should all be connected. If they are not, the control system will not work. When dynamic values are used, static values will be disabled and ignored.

Low Clamp — Low clamping value for output. Prevents voltage swings from exceeding the range of the hardware used for the control (usually D/A converter).

High Clamp — High clamping value for the output.

Rate Clamp — Rate of change clamp for output, measured in units/minute. If there is a large change in the setpoint or feedback, the rate clamp parameter will prevent the PID output value from making a change larger than the hardware is able to support. Instead the output value will be limited to change at the specified rate.

Filter Constant — The value for the measured variable can be automatically filtered if it is noisy. Filter value of 0.0 means that no filtering takes place. As the values increase toward 1.0, the filtering effect becomes more and more pronounced.

Setpoint — The PID setpoint value, or desired value. The value can be changed on the fly (dynamic) by specifying a ramp or other block to be used as the dynamic setpoint. If a constant setpoint is desired, enter the static setpoint.

Dynamic Setpoint — If an outside value is connected to this entry, the setpoint value becomes dynamically changed by the outside value instead of using the constant value. When the dynamic setpoint is active, the static one is disabled and ignored.

Wiring In: PID Control block accept wiring input data as Feedback, Setpoint, P parameter, I parameter, D parameter and Trigger for PID change value and use them in block configuration.
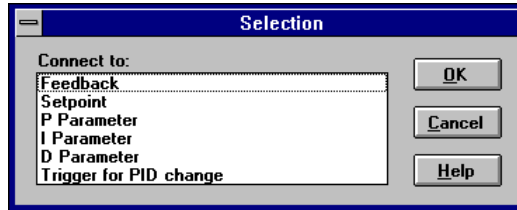
Figure 5-74   PID Control connection selection

Wiring Out: PID Control block outputs the controller value to a connected block.

5.3.15 Ramp Block

This block allows input and output. A ramp of floating point values may be generated by using this block. By connecting a high digital value to the reset input, the ramp can be reset to its starting value. By applying a high digital value to the hold input, the ramp may be temporarily held at the current value. The rate of increase/decrease of the ramp is proportional to the sample rate (scan task period) selected.
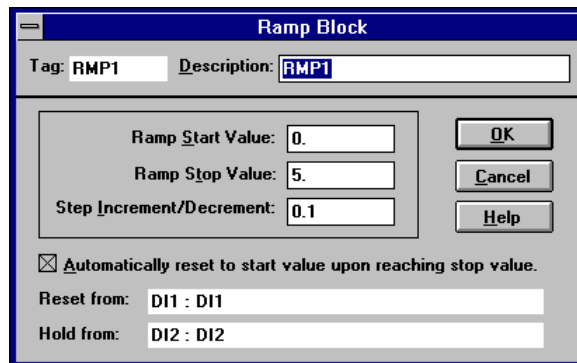
Figure 5-75   Ramp block

Start Value
A floating point value at which the ramp will start.

Stop Value
A floating point value at which the ramp will finish. Can be above or below the Start value.

Increment/Decrement
Each step up/down the ramp will be equal to this value.

Reset from Block
Any digital type block used to apply a digital high/low to reset/start the ramp. A digital high applied to this input resets the ramp and stops ramping. A digital low allows ramping to occur. No reset input is treated as a low input.

Hold from Block
Any digital type block used to hold the ramp at the current value. A high digital signal applied to this input stops ramping at the current value. A low applied to this input will enable/resume ramping. No hold input is treated as a low input.

Wiring In: Ramp blocks accept input data as Reset from or Hold from data and use the data in block configuration.
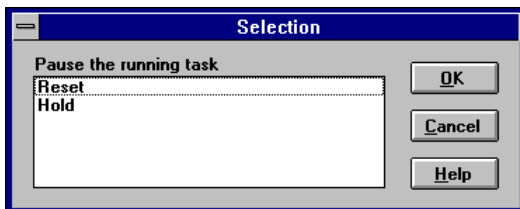
Figure 5-76   Ramp connection selection

Wiring Out: Ramp blocks output generated analog values to connected blocks.

5.3.16 RS-232 Block

SER1

The RS-232 block (or Serial Interface block) is used for communication between the GeniDAQ host computer and other serial devices (other computers, ADAM modules, etc.) that support the RS-232 standard.

Data in the form of a **prompt string** can be sent out to the serial device, and the response can be fed to another block within GeniDAQ. If the prompt string is to be static (not to be changed), then the string is entered in the dialog box by the user. As a dynamic alternative, the prompt string may be sent in the form of an input to the block by connecting a User Programmable or other block capable of outputting string variable types.

Double clicking on this block will bring up a dialog box where you can configure the communication parameters that GeniDAQ will use. Standard conventions are used for the port, data bits, parity, and stop bits.

Figure 5-77   Configure serial communication parameters

Idle time is the time (in ms) that GeniDAQ will wait before it starts looking for a response from the serial device. This feature is very convenient since some serial devices are very slow to respond. Using an idle time will eliminate time-out errors when slow serial devices are used.

Wait time is the time (in ms) that GeniDAQ will wait for a return string after the idle time has elapsed. If the wait time passes with no string received, or no final character, then GeniDAQ will retry sending the prompt string (see below).

Retry is the number of times that GeniDAQ will retry sending the prompt string before terminating. Acceptable numbers for retry are between 0 and 3.

Note: The sum (Idle+Wait) must be less than the scan task period.

The initial string **(Init String)** can be used to configure the serial device. It will be sent out as a first string, and the data that is sent back to GeniDAQ will be thrown away.

After the initial string is sent, and the response (which is ignored) is received, the **Prompt String** will be sent, after which all data received back **(Response String)** can be used by another block that is connected to the RS-232 block. You must specify the starting and ending point with respect to the data in the string. By pressing **More**, additional response strings may be entered in the **Additional Response Strings** dialog box (shown below).

The "Init String" and "Prompt String" fields allow the user to enter control characters in the range of ASCII 0 to ASCII 31.

For example, entering "This ^A^B^C^[String^M" is converted to "This <Ctrl_A><Ctrl_B><Ctrl_C><ESC> String<CR>" where <Ctrl_A> is ASCII 1 and <ESC> is ASCII 27.

ASCII	Enter	Meaning
0	^@	Null
1	^A	Start of Heading
2	^B	Start of Text
3	^C	End of Text
4	^D	End of Tape
5	^E	Enquiry
6	^F	Acknowledge
7	^G	Bell
8	^H	Backspace
9	^I	Horizontal Tab
10	^J	Line Feed
11	^K	Vertical Tab
12	^L	Form Feed
13	^M	Carriage Return
14	^N	Shift Out
15	^O	Shift In
16	^P	Data Link Escape
17	^Q	Device Control 1
18	^R	Device Control 2
19	^S	Device Control 3
20	^T	Device Control 4
21	^U	Negative Acknowledge
22	^V	Synchronize
23	^W	End of Transmitted Block
24	^X	Cancel
25	^Y	End of Medium
26	^Z	Substitute
27	^[	Escape
28	^\	File Separator
29	^]	Group Separator
30	^^	Record Separator
31	^_	Unit Separator

The string entered will be converted to a final string using the following scheme:

1. The "^" character followed by a character from "@" to "_" is interpreted as the lower 32 control characters in the ASCII table. ^@ is NUL, ^A is control-A, ^B is control-B, etc. (letters: 'A' to 'Z' are upper case only)

2. Backquote character "`", which is also called a grave accent, means the next character will be recognized literally, rather than converted to a special character. For example "`|" is the ASCII 124, "`^" is the "^" character, and "``" is a single backquote character because the first single backquote character protects the second one from being converted.

By pressing the **More** button, an **Additional Response Strings** dialog box will appear. In this dialog box, you can specify up to seven (7) additional response starting and ending locations with respect to characters in the string.

Wiring In: RS-232 blocks accept input data as prompt string. If a prompt string is to be in static mode (not to be changed), then the string is entered in the dialog box by the user. As a dynamic alternative, the prompt string may be sent in the form of an input to the block by connecting a User Programmable or other block capable of string variable types.

Wiring Out: RS-232 blocks output the response to another connected block within GeniDAQ.

Note: *If entered strings input from other blocks, the conversion will not work properly.*

5.3.17 Single Operator Calculation Block (SOC)

This block does single operator math computations, such as add, subtract, multiply, etc. At least one block must be connected as an input block; this will be the first operand. The second operand can be either another block or a constant entered in the Single Operand Calculation block's dialog box.
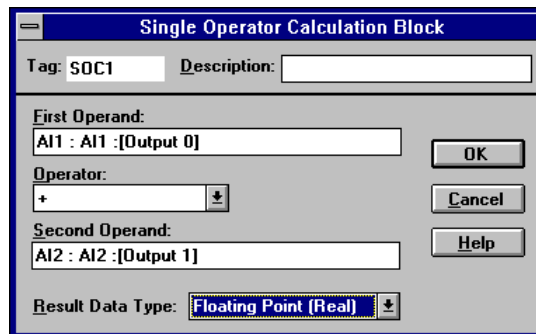
Figure 5-78   Single operator calculation block

Once two operands are selected, the order can be switched by pressing the **Swap Operands** button. This is important, since some operators, such as **div** or **mod**, will yield different results depending upon what values you use for OP1 and OP2 (first operand and second operand). The first block that is connected to the Single Operator Calculation block will be the default first operator in the dialog box. If you wish to have a different order, press the **Swap Operands** button. This will switch the order of operation of the SOC block.

The Result Data Type (output data type) can be either Floating Point (Real) or Integer.

Single Operator Calculation Block Operators and Functions

Operator	Function (Output)
nop	Outputs 0 always
+	OP1 + OP2
-	OP1 - OP2
x	OP1 * OP2
/	OP1 / OP2
pow	OP1 ^ OP2
* mod	Outputs remainder of OP1 / OP2
* and	Logical AND of OP1 and OP2
* or	Logical OR of OP1 and OP2
* xor	Logical XOR of OP1 and OP2
max	Outputs the maximum of OP1 and OP2
min	Outputs the minimum of OP1 and OP2
>=	Outputs 1 if OP1 >= OP2, 0 otherwise
<=	Outputs 1 if OP1 <= OP2, 0 otherwise
>	Outputs 1 if OP1 > OP2, 0 otherwise
<	Outputs 1 if OP1 < OP2, 0 otherwise
equ	Outputs 1 if OP1 == OP2, 0 otherwise
neq	Outputs 1 if OP1 is not equal to OP2, 0 otherwise
abs	Outputs the absolute value of OP1
* not	Logical NOT of OP1
inv	Outputs 1 / OP1 (inverse of OP1)
sqrt	Outputs the square root of OP1
log	Outputs the log of OP1 (base 10)
ln	Outputs the natural log of OP1 (base e)
exp	Outputs e ^ OP1
jct	Junction block (see below)

Operators with an asterisk "*" next to them require an integer operand. Some of the above mentioned operators need only one operand, while others need two. The logical operators (AND, OR, XOR) require two operands which must be integers. Other operators (ABS, NOT, INV, SQRT, LOG, LN, EXP, JCT) only require one operand, which can be either integer or floating point format, depending on the operator.

Some care is needed when providing input operands to certain operators. The MOD and "/" operators cannot have 0 (zero) as an operand to avoid a divide-by-zero situation. In the same way, some operators (SQRT, LN, LOG) require a positive value as an operand. Runtime errors will occur if these rules are not followed.

The JCT operator has a special function of simply outputting its input. It's useful for connection between a button display item and many other blocks. In the display editor, the button display item is designed to output to one and only one other block. Use of the JCT operator allows you to connect the button display item to many other blocks, by using the JCT operator as a branching point. For this example, you would configure the button display item to output to the SOC block, and connect the display block to the SOC block (arrow pointing into the SOC block). Then, configure the SOC block to function as a JCT, or junction. You can now connect as many blocks as you wish to the SOC JCT block, and this in turn will give control of all of those daughter blocks to the single button.

Wiring In: Single Operator Calculation block accepts input data as operands for the operator.

Wiring Out: Single Operator Calculation block outputs the calculated value to connected block(s).

5.3.18 Beep Block

This block accepts one input. It provides alarm output to the speaker in the PC or an external speaker. Digital blocks are used as input. This block was called Speaker block in previous versions of GeniDAQ.
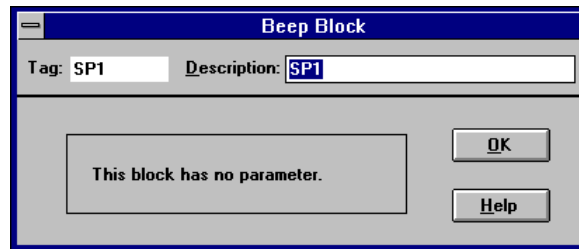
Figure 5-79   Beep block

Wiring In: Beep blocks accept digital input data to trigger sound alarms.

Wiring Out: Beep blocks pass the input data to connected block(s).

5.3.19 Temperature Measurement Block

Similar to the AI (analog input) block, this block allows for outputting data to another block directly from the I/O device. Transforms the analog input data from the I/O device into linearized temperature data in degrees C,F,K, or R. Thermocouple types supported are J, K, S, T, B, R, and E.

The Update rate is a divisor that allows the Temperature Measurement block to have a different effective scan rate than the rest of the strategy. This is useful if, for example, your strategy is running at 100 Hz, but you only want to output data at a rate of 20 Hz. For this example, you would set the Update Rate to 5 (100 divided by 5 (the update rate) gives an effective scan rate of 20 Hz). In this scenario, you will still get 100 Hz data if you send the output to a Log File block or a display block, but only one in five samples will be "real". The other samples are merely copies of the "real" sample.

The Temperature Measurement block also has DDE capabilities that allow it to exchange data with other Windows applications. For more information about DDE, see "DDE Blocks (Client and Server)" elsewhere in this section.
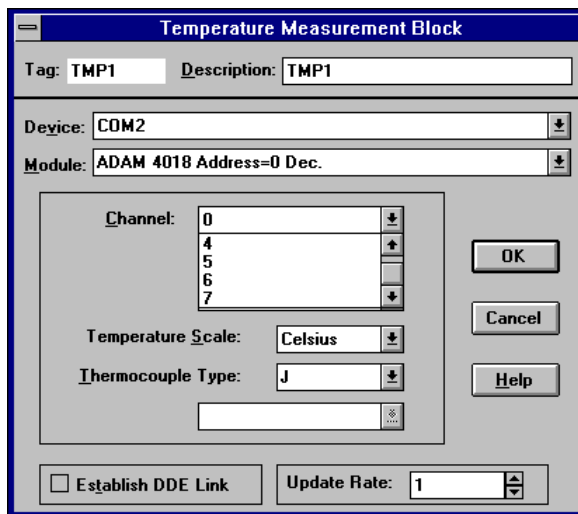
Figure 5-80   Temperature measurement block

Device/Module — Choose the device from the installed I/O device list.

Channel — Choose the analog input device channel.

Input Range — Choose the input range, if applicable.

Expansion Channel — Choose the expansion (mux) card channel.

Temperature Scale — Choose the desired temperature scale in degrees C (Celsius), F (Fahrenheit) ,K (Kelvin) , or R (Rankine).

Thermocouple Type — Choose the type of thermocouple that the hardware supports.

Establish DDE Link — Allows you to establish a link with other Windows applications. See "DDE Blocks (Client and Server)" elsewhere in this section.

Update Rate — Change the effective scan rate of this particular block. The value entered acts as a divisor; the scan rate divided by the update rate gives the effective scan rate for this block.

Wiring In: No wiring input connection. "Cannot accept input" message will be displayed.

Wiring Out: Temperature Measurement block transforms the analog input data from the I/O device into linearized temperature data in degrees C,F,K, or R. Thermocouple types supported are J, K, S, T, B, R, and E.

5.3.20 Timer Block

Accepts inputs (for reset purposes) and also allows outputs. Time can be absolute, or elapsed, and resolution can be in ticks (0.1 seconds) or seconds. The time cycle range can be set from each minute to each year. Output (elapsed or absolute time) can be sent to another block.

The Timer Block's output is a long integer from 0 to 4294967295. This block is very useful for any type of control strategy that involves time as an element.

The unit of the timer's output value is in either seconds or ticks, depending upon the resolution that you choose. Because of the constraints of the Windows environment, it is difficult to get better resolution than that provided here.

The output of the timer block is cyclic if one of the cycle options is chosen. For example, the output of the timer block with "Elapsed time" type, "Second" resolution and "Minutely" cycle goes from 0 to 59 and back to 0. The same block with hourly cycle will output 0 to 3599 and back to 0.

The timer block can be reset by another block; a value of 1 input to the timer block reset will reset the timer.
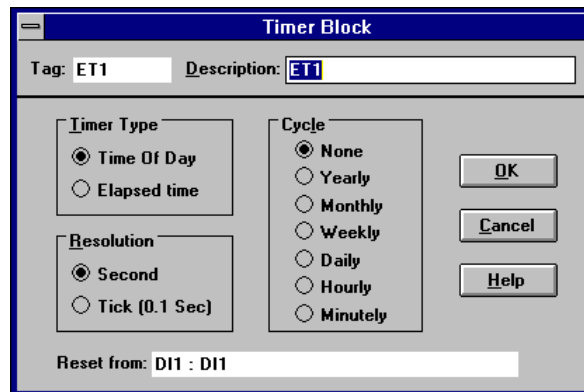
Figure 5-81   Timer block

For a timer block with Time-Of-Day (absolute) type, the output doesn't start from 0 at the beginning of the run. It gives the number of seconds or 1/10 seconds since the boundary of the cycle, according to the computer's clock. For example, the output of an hourly timer (with seconds resolution) is 0 every hour on the hour, then incremented to 1,2,3, ... ,3599, then back to 0 on the next hour. The boundary of various timer cycles are listed as follows:

YEARLY:	Starts from 0 at 00:00:00 AM January 1st every year.
MONTHLY:	Starts from 0 at 00:00:00 AM the first day of each month.
WEEKLY:	Starts from 0 at 00:00:00 AM on Sunday every week.
DAILY:	Starts from 0 at 00:00:00 AM every day.
HOURLY:	Starts from 0 every hour on the hour.
MINUTELY:	Starts from 0 every minute on the minute.

It's easy to calculate the actual time based on the cycle type. For example, the weekly timer outputs 86400 at 0:00 AM Monday morning and 518400 at 12:00 PM Friday midnight. You can use this value as input to the User Program Block and use it to turn something off during the weekend or turn something on during the weekdays. You can also use several timers with different cycles to do more complicated timer control.

See TIMER.GNI (in the GeniDAQ\Strategy directory) as an example of how to use the Timer Block. This strategy makes 3 beeps at 0th, 2nd and 4th second of every minute for the first 5 minutes of every hour Monday through Friday.

Wiring In: Timer blocks accept input data to reset timer(s).

Wiring Out: Timer blocks output elapsed or absolute time values (a long integer), to a connected block.

5.3.21 Time Stamp Block

TS1

This block has output capability. The current time may be assigned to the display or to a log file by connecting this block's output to a log file or Numeric/string display item.

Different output formats are allowed, and these outputs are in the form of a string.
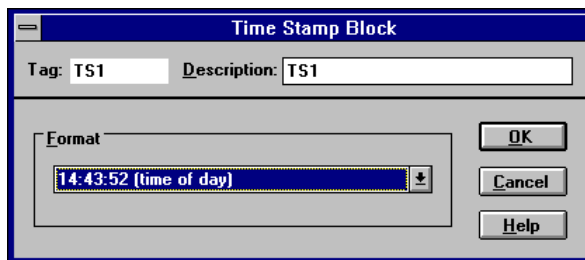
Figure 5-82   Time Stamp block

Wiring In: No wiring input connection. "Cannot accept input" message will be displayed.

Wiring Out: Time Stamp block outputs the current time to connected block(s). There are multiple time formats for selection.
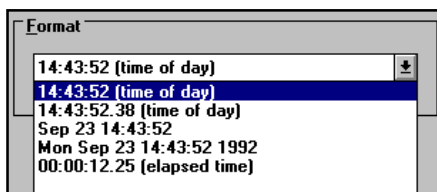
Figure 5-83   Time Stamp output format

5.3.22 User Programmable Block

This block allows for both input (up to 8 channels) and output (up to 8 channels). You can connect the inputs and perform math operations on those blocks. You can also write a program that acts on the inputted blocks as variables, with one or many lines in the program. Up to eight inputs and up to eight outputs are supported by the User Programmable Block. You are advised to use a BasicScript block to replace this block.

This block is designed to provide total flexibility so you can do comparisons, calculations, conditional statements, branches and loops. It uses C-like syntax without the complexities of actual C programming. It is, however, only an interpreter, not a compiler. Given this, some speed is compromised in the name of simplicity. The User Program block was designed for small amounts of code, not large, complex programs. It supports multiple inputs (no limit) from all types of blocks. Its output can be routed to any number of blocks. You can write a program to manipulate inputs, calculate an output based on the inputs, or simply skip to the next scan (sample) without any output for the current scan. In the last case, all blocks that have this programmable block as an input will not be processed for the current scan. You can also halt the entire runtime session by use of this block.
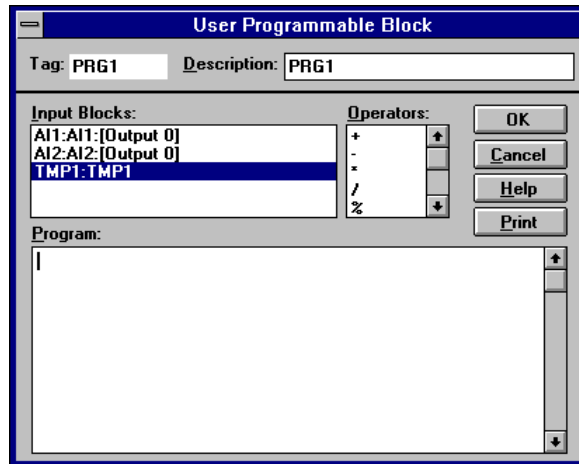
Figure 5-84   User programmable block

The user programmable block dialog box consists of three sub-boxes: the **input blocks** box, the **operators** box , and the **program editor** box. The *input blocks* list box provides the list of all inputs currently connected to the user programmable block. The *operators* list box provides the list of all available operations (see list below). The *program editor* box is an editor where you can type-in program statements. Clicking on an input block in the input block list inserts the tag name of that input block into the program editor box. Clicking on an operator has the same effect.

Program
A program consists of one or more statements that follow the correct syntax.

Statements
Statements consist of keywords, expressions, and other statements. A statement that forms a component of another statement is called the "body" of the enclosing statement. Carriage-return and line-feed characters add no meaning to a program, so a statement can be in a single line or broken into several lines. The User Programmable Block recognizes the keywords **if, else, while, output, skip,   stop, display, substr,** and **sprintf**; any connected block's tagname variables (**AI1, DO2, etc.**); and the variables **a - z** and **a1 - z1** to **a9 - z9.** Each of the following statement syntax types is valid:

1) *expression* **;**
2) **if (** *logical expression* **) ** *statement* **;**
3) **if (** *logical expression* **) ** *statement* **; ** **else** *statement* **;**
4) **while (** *logical expression* **) ** *statement* **;**
5) **output** *expression* **;**
6) **skip ;**
7) **stop ;**
8) **display** *expression* **;**
9) **substr** *expression* **;**
10) **sprintf** *expression* **;**

The keyword **output** allows you to output an expression to the next block (connected to this block's output). When outputting data from the User Programmable Block, the data output may be in the form of an integer, floating point value (real number), or string. Since the block can output eight different data values, the syntax for the output keyword is **output** *expression.* This expression contains as the first parameter the output channel number (#0 - #7), while the second parameter contains a value to be sent. If outputting a string, the **string must be in quotes** ("String"). If no channel number is provided, the default (channel one) is used. Examples are as follows:

1) **output(#2, "This String");**   // outputs a string to channel 2
2) **output "Another String";**   // outputs a string to channel 1 (default channel number)
3) **output(#7, "String" + a4);**   // outputs string + a4, where a4 = "Another String" to channel #7
4) **output(#1, AI1 + AI2);**   // outputs tagnames AI1 + AI2 to channel #1
5) **output(#7, 1);**   // outputs 1 to channel #7

When the output of this block is connected to the input of another block, a dialog box will appear asking you to specify the output channel number desired (#0 - #7). If the program does not assign a value to a particular channel number specified at any given moment, its output will be zero (0) until an output is valid.

The keyword **skip** allows you to skip (wait) until the next scan or sample before proceeding.

The keyword **stop** allows you to halt the runtime session immediately.

The keyword **display** allows you to change between different displays during runtime. The syntax for the display keyword is **display** *expression* where expression contains as the parameter the display number (N). Examples are as follows:

1) **display(2);** // changes display to display number 2

The keyword **substr** allows you to parse a string into a sub-string during runtime. The syntax for the display keyword is **substr** *expression*, where expression contains as the first parameter the string or string variable to be parsed. The 2nd parameter contains the character number at which to start parsing (the first character in the string is represented by a one (1)). The 3rd parameter contains the sub-string length, or how many characters to parse. Example is as follows:

d5 = "This string";

1) **substr("abcdef", 5, 2);** // starting at the 5th character, e, parse out 2 characters, ef

2) **substr(d5, 6, 6);** // starting at the 6th character, s, parse out 6 characters, string

The keyword **sprintf** allows you to format and store a series of characters and values in a buffer variable. The syntax for the sprintf keyword is **sprintf** *expression,* where expression contains as the first parameter the string or string variable to act as a *buffer* for the formatted string to be stored. The 2nd parameter contains the *format* type for the argument. The 3rd parameter contains the *argument* to be formatted. The argument is converted and output to a buffer according to the format specification in *format*. The format consists of ordinary characters and has the same form and functionality as the format argument in the ANSI C function printf(). However, only one argument can be formatted with this version of sprintf. Examples are as follows:

```
// This program uses sprintf to format various
// data and place them in the string named b.

s = "computer";
c = "l";
i = 35;
f = 1.7320534;
   // Format various data:
sprintf( b, "\tString:    %s\n", s );
sprintf( b, "\tInteger:   %3d\n", i );
sprintf( b, "\tReal:      %6.3f\n", f );
```

The word *statement* in italics represents a single statement or compound statement. A compound statement is one or more statements enclosed by a pair of curly brackets { }. Note that a compound statement doesn't need to be ended by a semicolon character like a single statement does, so there is no semicolon after the closing bracket. The brackets are just a way of grouping single statements.

To include comments within the User Program block, use two forward slash marks (//) to begin each comment line.

Expressions
An *expression* is a sequence of operands (variables or numbers) and operators (math functions) that compute a value. The computed value can be a number or a logical true/false. The simplest expression is a constant (operand) without any operator.

Examples of valid expressions are:

```
3
AI1+AI2
(abs(AI1)*2.5) + 1
AI1 > 0 && AI2 < 0
```

The last example contains a *logical expression*. Some operators have higher precedence than others do. An operator with higher precedence is always processed before the one with lower precedence. The following table (table 1) shows the precedence in descending order and associativity of the operators:

Table 1. Precedence and Associativity of Operators

Symbol	Type of operation	Associativity
( )	Expression	Left to right
- !	Unary minus Unary negation	Right to left
* / %	Multiply Divide Remainder	Left to right
+ -	Additive	Left to right
< > <= >=	Relational	Left to right
== !=	Equal, Not equal	Left to right
&	Bitwise AND	Left to right
|	Bitwise OR	Left to right
~	Bitwise NOT	Right to left
^	Bitwise XOR	Left to right
<< >>	Shift Left Shift Right	Left to right
&&	Logical-AND	Left to right
||	Logical-OR	Left to right
=	Assignment	Right to lef

Table 2: Mathematical Functions

Function	Argument	Returns...
abs()	integer or float	Absolute value of argument
cos()	Float or integer (radians)	Cosine of argument
int()	Float	Integer from Floating point argument
rnd()	Integer seed value	Random number starting from argument (seed)
sin()	Float or integer (radians)	Sine of argument
sqr()	Float or integer	Square-root of argument
ln()	Float or Integer	Natural Log of argument
log10()	Float or Integer	Log of argument
exp()	Float or Integer	e to the x argument
pow(x,y)	Float or Integer	x argument to the y power
tan()	Float or integer (radians)	Tangent of argumen

The mathematical functions that are available in the User Program block are shown in Table 2: Mathematical Functions.

The int() function deserves special attention here. When an expression that involves a floating point value and an integer is evaluated in the user program block, the return value is then converted to a floating point value. In some cases, you may want an integer value returned instead, and int() will do this. Int() converts a floating point value to an integer value. For example, if the floating point value is 1.49, int() will return a value of 1. If the floating point value is 1.51, int() will return 2. Therefore, Int() will return the floating point value, rounded to the nearest integer.

Variables

The variables (**a - z** and **a1 - z1** through **a9 - z9**) can be used for any purpose. It must be remembered that if a variable is used in one user programmable block, its value will be seen by all other user programmable blocks that use the same variable. This effect may be desirable in some cases, but it is usually good practice to use different variables for each user program block. The displayed input block variables (tagnames AIN, DIN, etc.) can also be used as operands.

For example:

```
if  ( AI1 > AI2 )
{
    a = AI1 * 0.117;            // this is how to
    output (a);                 // use a comment
}
```

The above program checks if analog input 1 (AI1) is greater than analog input 2 (AI2). If it is, the variable 'a' is used to hold the product of AI1 and the constant 0.117 (scaling factor). The product is then sent to the next block (**output**). All statements inside the curly brackets (compound statement) are done only if AI1 > AI2.

Wiring In: This block allows for up to 8 channels input. You can connect the inputs and perform math operations on those blocks.

Wiring Out: This block allows for up to 8 channels output to connected blocks.

5.3.23 Conditional Wavefile Block

SOUND1

This block has input capability. It accepts a value between zero (0) and seven (7) from another block, each value providing capability to select a wavefile to be played via a sound card during runtime. When this block is double clicked on, a dialog box will appear displaying all wavefiles currently installed.
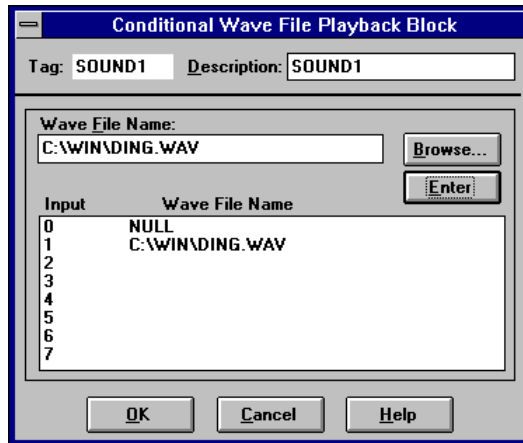
Figure 5-85   Conditional wavefile block

Wavefile Name
Specify the wavefile to be selected with the currently highlighted Input Value. Enter the full path, or **Browse** to locate the desired wavefile.

Wiring In: This block accepts input values between zero (0) and seven (7) from another block, each value providing capability to select a wavefile to be played via a sound card during runtime

Wiring Out: No output available

5.3.24 Alarm Log Block

ALOG1

This block has input and output capability. The block allows Alarm data from inputs to be logged to the GeniDAQ Event Log File (\GENIDAQ\GENIE.ELF). This data can also be displayed and acknowledged during runtime via the Event Log Viewer when the input data falls within the following regions:

1) If the input data is above the high-high value

2) If the data is between the high and high-high value

3) If the data is between the high and low value

4) If the data is between the low and low-low value

5) If the data is below the low-low value

The block also outputs a value depending on what data is seen by the block. This value can be fed to one of the conditional display items to show the state of the alarm. For the following regions (regions annotated above):

1) The block outputs a four (4)

2) The block outputs a two (2)

3) The block outputs a zero (0)

4) The block outputs a one (1)

5) The block outputs a three (3)

Figure 5-86   Alarm log block

While the strategy is running, alarms may be acknowledged by double-clicking on the Alarm Log Event (ALOG1 HI, etc. below). Alarms will be in the color red until acknowledged. For the Event Log dialog box to appear during Runtime, you must enable it via the View Menu.

Wiring In: The block accepts input data and logs to the GeniDAQ Event Log File (\GENIDAQ\GENIE.ELF). Input data can also be displayed and acknowledged during runtime via the Event Log Viewer.

Wiring Out: The block forwards input values to connected block(s).

5.3.25 BasicScript Block

This block is designed to provide total flexibility so you can do comparisons, calculations, conditional statements, branches and loops. It uses Microsoft Visual Basic and Visual Basic for Applications compatible syntax and functions without the complexities of programming. It is, however, only an interpreter, not a compiler, and so some speed is compromised in the name of simplicity. The BasicScript block was designed for small amounts of code, not large, complex programs. It supports multiple inputs (no limit) from all types of blocks. Its output can be routed to 8 blocks. You can write a program to manipulate inputs, calculate an output based on the inputs, or simply skip to the next scan (sample) without any output for the current scan. In the last case, all blocks that have this program-mable block as an input will not be processed for the current scan.
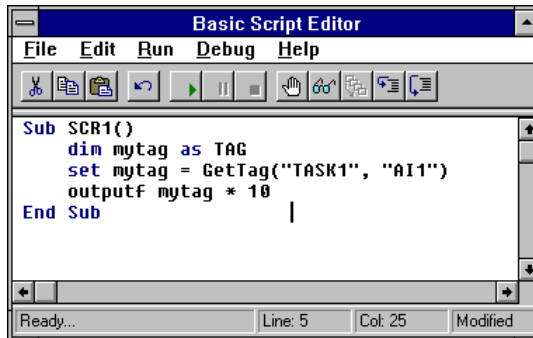
Figure 5-87   BasicScript block

The BasicScript block dialog box is a BasicScript Editor. You can edit, debug and trace a script program in this editor. It is a robust script development environment. For detailed information, please refer to *chapter 10 Script Designer.* The BasicScript Editor window is a pop up window which must be exited before manipulating other Designers (for example, Task designer or Display Designer).

Wiring In: BasicScript block access data directly from data center. The input will not take effect for this block.

Wiring Out: This block allows for up to 8 channels output to connected blocks.

5.3.26 TAG Block
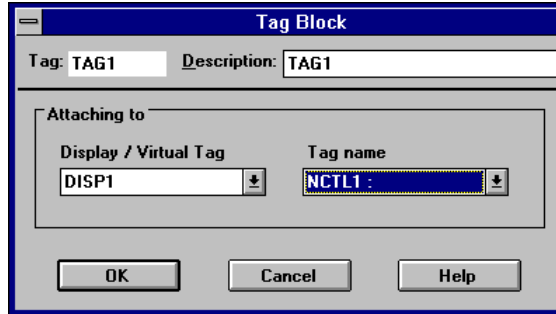
Figure 5-88   TAG block

This block is used to link a task block to a display item or a virtual tag. The value of a display control item can be passed to other blocks. Virtual tags must be selected from within the TAG block. Once the virtual tag is thus linked, its value can be applied to other task blocks.

Wiring In: No input. TAG blockaccepts the input value from the display window.

Wiring Out: TAG block passes the values to connected block(s).

5.4 Virtual Tags

Virtual Tag is a powerful feature that provides the ability to create customized tags in Task Designer without using User Defined DLL. The virtual tag is created by Task Designer and stored in data center as other built-in blocks. The virtual tags are globally available to all tasks, and can thus be used to share data among multiple tasks.

Figure 5-89   Virtual Tag table

Add/Delete virtual tag

To create a virtual tag, select the Add/Delete Virtual Tag option from the Setup menu. A dialog box will be displayed to add/delete virtual tags. When the dialog box appears, a list of available virtual tags will be displayed in the dialog box. To add a virtual tag, key in the name of the new virtual tag at the input line and press the Add button. A new virtual tag will be added to the list of available virtual tags. After adding, you can press the OK button to exit the dialog box. To Delete a virtual tag, click on the same menu and option as used to add a virtual tag, select the desired virtual tag in the available listing and press the Delete button.

Using Virtual Tag in Task Designer

Once created, virtual tags are not displayed in or added to the Toolbox. It is an internal tag block. You have to use and configure a TAG block to link with it. The configuration is displayed in the dialog box below. Choose VIRTASK in the Task/Display field of dialog box and select the name of the virtual tag in TAG field. The configuration is then complete. After linking with a TAG block, you can connect the virtual tag with other blocks in Task Designer which have input and output capabilities. That is, virtual tags can accept data from other blocks, such as an AI block, and output values to other blocks, such as DO blocks.
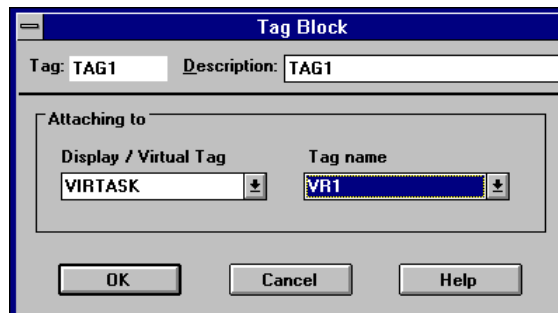
Figure 5-90   Link Virtual Tag to Tag block

Using Virtual Tag in Display Designer

Virtual tag values can be displayed on screen by configuring the display item to link with virtual tag. Refer to the figure below. Choose VIRTASK in Task/Display field of the dialog box and select the name of the virtual tag in the TAG field. The configuration is then complete.
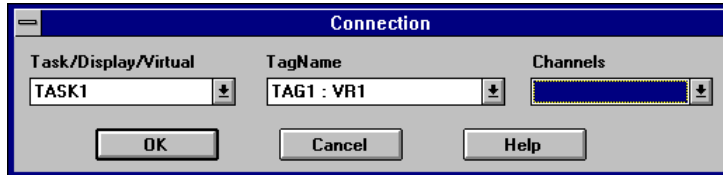
Figure 5-91   Use Virtual Tag in Display Designer

Using Virtual Tag in BasicScript

Virtual tag is same as other blocks in BasicScript. You can get tag values and set tag values using the GetTag and SetTag commands. The difference between virtual tag and task blocks is that task block values do not read back to GeniDAQ tasks from data center after setting new values. Virtual tag values can be read back to GeniDAQ tasks from the data center. Because of this difference, you can retrieve data from BasicScript programs or other applications for GeniDAQ tasks or displays.
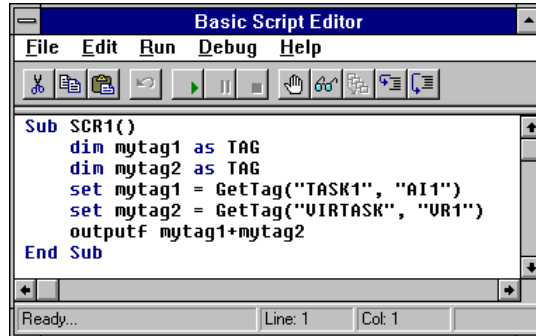
Figure 5-92   Use Virtual Tag in BasicScript block

6

Display Designer

6 Display Designer

This section provides a description of the Display Designer. Display Designer is the tool that you use to create your Operator Display Panel, allowing a dynamic interface with the running strategy. You can create a panel that is similar to test equipment or industrial process displays. You can also attach a bitmap background to customize your display. The operator display allows you to monitor, supervise, and control your process during runtime. When complete, the Panel can resemble Meters, Bar graphs, Waveform Displays, Chart Recorders, Buttons, Numeric Readouts, and LED Indicators.

6.1 Working with the Display Designer

Once you have developed your process strategy, you are ready to decide which functions of the strategy will be viewed by operators when it is executed in GeniDAQ Runtime. Take the time to plan out which blocks (variables) in the strategy you wish to access (control or monitor). Also, consider the display(s) and how they will be graphically represented during Runtime execution. You can "tap" into the desired blocks or locations in your strategy by configuring the display item to the location where you would like a variable to be controlled or monitored. Multiple displays and/or operator control panels can be created. You can access one or many variables with each panel, using one or many Display Items from the Display Designer Toolbox.

When you make dynamic connections, you are actually making a link into your strategy's database. The appropriate strategy must be loaded into the Task Designer prior to making dynamic connections to any Display Item(s).

Enter the Display Designer by clicking on the File menu and Add/Delete Task/Display. The Add Display option opens a new Display operator panel within Display Designer. You can now create a custom operator panel by arranging various Display Items (bar graphs, YT graphs, indicators, buttons, numeric controls, and text displays) on the screen and configuring each. The Display Items are dynamically linked with each connected block.

Selecting Display Items from the Toolbox
From Display Item Toolbox, you have at your fingertips all Display Items available in the GeniDAQ Display Designer.

Figure 6-1   Display Item Toolbox

Using the Display Item Toolbox is identical to using the Icon Block Toolbox in the Task Designer. You get the Display Items from the Toolbox by pointing the mouse cursor at the desired item, clicking, placing the mouse cursor at the desired location on the screen, and dropping the item by again clicking the mouse button. You can re-size the Display Items and configure them according to your specifications.

A complete list of Display Items and their associated function(s) is provided in *Chapter 6.3 Display Item Toolbox*.

Configuring Display Items

When placing the Display Items on the screen, you can re-size them. First, select the Display Item that you wish to re-size. When selected, place your mouse pointer on one of the black squares at the Display Item's edge, and drag the side of the item while the mouse button is still pressed. Once sized, you must configure the Display item by double clicking anywhere on the Display Item. A dialog box appears, where you can configure the Display Item, dynamically linking it to display/control a tagged Task Designer block's data.
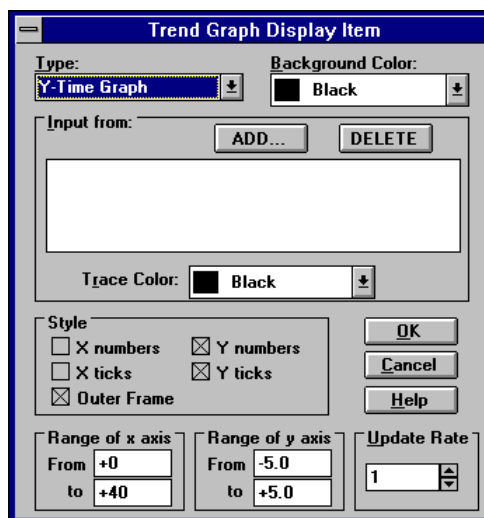
Figure 6-2   Configure a Trend Graph Display Item

Some of the Display items have scalable font properties. See the individual blocks for more information. For specific information about configuring each Display Item type, refer to *Section 6.3 Display Item Toolbox*.

Using a Bitmap Background in Your Display

If you wish to add a background to your display, you can use a previously designed bitmap picture as a background, similar to Windows Wallpaper. Select File/Attach Bitmap from the Display Designer Menu Bar to load a bitmap picture to the current display. You can then place display items on top of the bitmap to enhance your display.
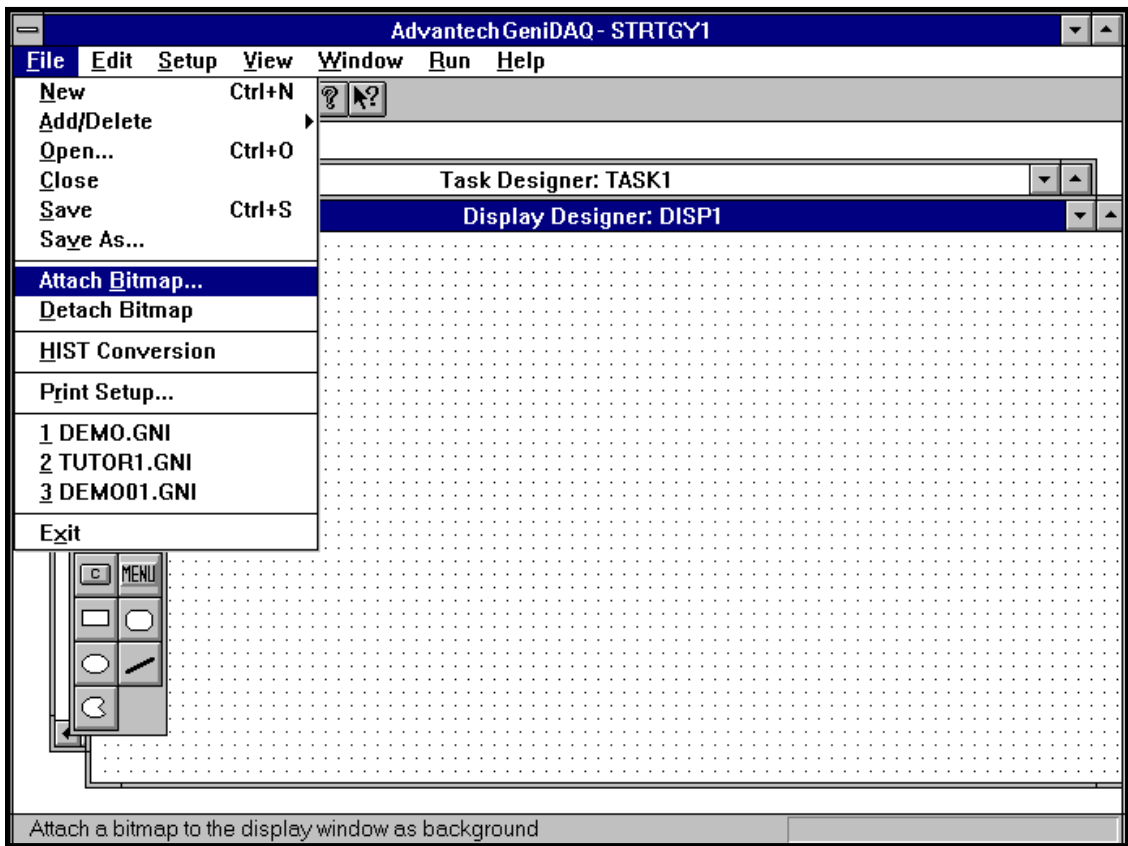
Figure 6-3   Adding a bitmap background to your display

Saving and Loading Your Display

The Display is Saved whenever strategy is saved, as described above.

The Display can be Loaded by simply loading your strategy. You can view your display using the Window menu to selected it.

Modifying Your Display

During your display editing process there will be times when you need to make changes. The Edit menu provides functions that allow you to modify your operator display panels quickly and efficiently. All functions that are available in the Task Designer Edit menu are available when in the Display Designer. For Edit menu functions and procedures, see *Section 6.2 Display Designer Menu*.

Exiting the Display Designer

When you wish to return to the Task Designer, click on the Window menu and select the desired window. This allows you to switch the active window to Task Designer or other display windows.

Exiting the Task and Display Designers in One Step.

It is also possible to exit the GeniDAQ program from the Display Designer screen. When you have finished your work in the Display Designer and desire to return to Windows or DOS, click on the Exit command in the File menu. You will be prompted as to whether you would like to save any changes that were made in the current session. Upon giving the appropriate response, you will exit GeniDAQ.

Editing the Display

To create a display, Display Items are arranged on the screen as you would like them to look when running your strategy. You get the Display Items from the Toolbox on the Display Designer screen. Choose whichever item best fits your strategy. When placing the Display Items on the screen, you can re-size them by placing your mouse pointer on one of the black squares (Display Item's edge) and dragging while the mouse button is still pressed. Once sized, you must configure the Display item by first double clicking on the Display Item. A dialog box will appear. From this dialog box you can configure the Display Item, dynamically linking it to display/control a tagged Strategy Editor block's data.

Click on File, Add/Delete Task/Display when in the GeniDAQ main window. This will invoke the Display Designer. The typical procedure for developing operator display panels using GeniDAQ Display Designer involves the following steps:

1) Use the Display Item Toolbox to select desired Display Items with your mouse and place them on the screen at the desired locations.

2) Change the size of the Display Item by selecting handles on the Display Item's edge with your mouse. While holding down the left mouse key, drag the edge until the desired size is attained.

3) Configure each Display Item by double-clicking anywhere on its surface. A dialog box will appear where you can choose which Strategy Editor block's data to dynamically display (or control, if supported).

4) Add Group Boxes and Text to the display panel to relate static display items.

5) Add a background picture (if desired) to your display by selecting File/Attach Bitmap.

6) Save the Display.

Drawing Tools

GeniDAQ enhances the man-machine interface (MMI) by providing graphic tools to draw pumps, valves, rectangles, circles, segments, and polygons in the screen designer's toolbox. In addition, it allows the user to configure the colors and sizes of these figures. These drawing tools include oval, rectangle, round rectangle, polygon and line. In addition to drawing tools, GeniDAQ provides "Make Object" and "Break Object" commands to let you integrate drawing components into a meaningful picture for your data acquisition and control.

Figure 6-4   Drawing tools

Setup Display Properties

In order to make your display friendlier, GeniDAQ provides the capability of setting windows properties to control the display. For example, maximize or minimize the window size, overlap or hide the window. The display properties are set by selecting Display Properties from the Setup menu. Before setting your display properties, you have to focus the display designer window as the primary window.

Support Multiple Display

GeniDAQ supports multiple displays in a strategy system. You can switch to or toggle between the displays using the menu button in the Display toolbox. The maximum number of multiple displays is dependent on the resources of your PC. However, you are advised to use less than 20 display windows at the same time.

6.2   Display Designer Menu

Display Designer Menus

> *File Menu*
>
> *Edit Menu*
>
> *Setup Menu*
>
> *View Menu*
>
> *Window Menu*
>
> *Run Menu*
>
> *Layout Menu*
>
> *Help Menu*

The following is a description of the Strategy Editor menus:

6.2.1 File Menu

The File menu includes commands that enable you to open, close, save, and print new or existing GeniDAQ Strategy files. Commands include:

File	
New	**Ctrl+N**
Add/Delete	▶
Open...	Ctrl+O
Close	
Save	Ctrl+S
Save As...	
Attach Bitmap...	
Detach Bitmap	
HIST Conversion	
Print Setup...	
Exit	
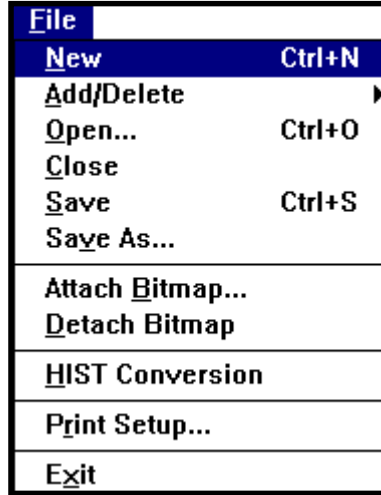
Figure 6-5   Display Designer File menu

New

Use this command to start a New strategy. During or after you create a strategy, you can save the strategy by using the Save As command.

Add/Delete

Use this command to add/delete a "task", "display" or "main script" in a GeniDAQ strategy.

- Add/Delete Task

Use this command to add a "task". A task is a collection of blocks and Display Items that are related to each other. Each task runs at the same rate, or "scan period" as set in the Setup/Task dialog box. In earlier versions, only one task can be created. In GeniDAQ 3.x you can design multiple tasks, and can toggle between the tasks during runtime.

- Add/Delete Display

Use this command to add or delete a "display". A display is a collection of display items that are related to each other. Each display has its display properties that are set by selecting **display properties** from the **setup** menu. In GeniDAQ you can design multiple displays, and can toggle between the displays during runtime.

- Add/Delete Main Script

Use this command to add or delete a "main script" for each strategy file. A main script file is used to control tasks in a strategy file. In GeniDAQ, you can have only one main script file for each strategy.

Open

Use this command to Open an existing strategy. Normally, existing strategies will be stored in the GeniDAQ directory.

Close

Use this command to Close your strategy. Make sure that you save it first.

Save

Use this command to Save your previously named strategy. To save an unnamed strategy, use the Save As command.

Save As

Use this command to Save a previously unnamed strategy. You can provide a path to where you would like the strategy stored, if different from the default C:\GENIDAQ directory.

Attach Bitmap

Use this command to attach a bitmap file as the background of a specified display window

Detach Bitmap

Use this command to detach a previously attached bitmap file (see Attach Bitmap, above). The display window background is returned to blank.

HIST Conversion

After GeniDAQ Runtime System acquisition is complete, use this command to Convert the Historical Trending Display Item Log files from Binary to ASCII format. When converting the format to ASCII, you can then examine the contents using any standard ASCII editor.

The Historical Trending Display Item allows data logging to be started and stopped when the strategy is started and stopped. A single Historical Trending Graph can support up to eight (8) process variables corresponding with eight (8) strategy tag names. Each Historical Trending Graph can write to one or more log file(s). Log files are binary files with a naming convention as follows:

```
YYMMDD##.HST
```

where:

```
YY = Year file was created
MM = Month file was created
DD = Day file was created
## = Historical Trending Graph Number
```

The name of the file contains the strategy start date and Historical Trending Graph number. Each log file can contain one or more Historical "Sections". Each section corresponds with the time between the start and stop of a strategy, if the start/stop time is within the file date. A new file will be created only when the strategy is stopped and then restarted after the file date ends. HIST conversion files are saved in the default directory (C:\GENIDAQ).

178
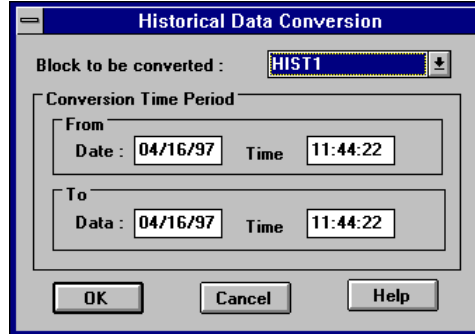
Figure 6-6   Historical data conversion

Block to be converted
This field is used to specify which historical block to be converted

From and To fields
These fields are used to specify the starting time and ending time of conversion.

Print Setup
Use this command to Setup your printer. All printers supported by Windows can be used with GeniDAQ.

Exit
Use this command to Exit the GeniDAQ Strategy Editor.

6.2.2 Edit Menu

The Edit menu includes commands that enable you to edit items in your GeniDAQ Strategy. To use Edit commands, you must first "grab" or select the items you want to edit. Do this by pointing your mouse somewhere outside the group of items you wish to select. Press the left mouse button, and as you hold down the button, drag the pointer diagonally across the items you wish to select. You will see a box form around the items. When the box is around the desired items, let go of the mouse button. When items are selected properly, there will be heavy black squares at the corners of all selected items. Once the items are thus selected, you can perform the desired commands.

Edit	
Cut	**Ctrl+X**
Copy	Ctrl+C
Paste	Ctrl+V
Select All	
Make Object	
Break Object	
Rotate Clockwise	Ctrl+O
Rotate C-Clockwise	Ctrl+R
Bring to Front	
Send to Back	

Figure 6-7   Display Designer Edit menu

Cut

To Cut items in GeniDAQ, select items that you wish to cut (see above, to Select Items). When the items are selected, point your mouse to the cut command and press the left mouse button. This will delete the selected items. Note that in GeniDAQ items which are Cut cannot be Pasted.

Copy

To Copy items in GeniDAQ, select items that you wish to copy (see above, To Select Items). When the items are selected point your mouse to the copy command and press the left mouse button. This will copy the selected items into a buffer. The items may now be "Pasted" to another location. Press "Paste" in the Edit menu. The selected items are now copied to the upper left of the screen area, with new item names. The group of blocks can now be moved to the desired location, using the instructions for Move (above). Remember that item names must always be unique for like-blocks. This is the way that GeniDAQ Runtime distinguishes one block from another.

You can also copy blocks from one strategy to another, by following the above procedure for copying. However, before pasting into another strategy, you must first close the initial strategy. You can then start a new strategy and paste the blocks into it, or paste the blocks into an existing strategy. Remember that the blocks that you paste retain their prior configuration!

Note that the linkage between a Display Item and Task Tag can't be copied and pasted as a configured item.

Paste

Once items are selected (see above, To Select Items), they may be copied to another location by first pressing Copy, and then pressing the Paste command. The items will be pasted to the upper left area of the screen.

Select All

To select every item in the current Strategy, use this command.

Make Object

After drawing graphic items in display window, you can group these graphic items to be one graphic item by using the Make Object command. Users can configure the color and size of the grouped object and link the object to task runtime values.
.

Break Object

The Break Object command is used to ungroup the graphic object into separate graphic items.

Rotate clockwise

The Rotate Clockwise command is used to rotate graphic items or a grouped graphic object in a clockwise direction.

Rotate C-Clockwise

The Rotate C-Clockwise command is used to rotate graphic items or a grouped graphic object in counterclockwise direction.

Bring to Front

The Bring to Front command is used to bring the graphic item to the front of all display objects in a display window.

Send to Back

The Send to Back command is used to send the graphic item to the back of all display objects in a display window.

6.2.3 Setup Menu

The Setup menu includes commands that enable you to install, add, or remove an I/O device, and to assign the Strategy to a particular Task. Commands are as follows:
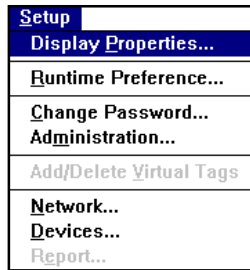
Figure 6-8   Display Designer setup menu

Display Properties

Display properties pop-up window is used to configure the properties of a display window. You can assign title, window style, state, and button settings:

Title

The title description will be displayed at the top of a window to describe the purpose of this display window.

Style

This field is used to set the display methods of multiple windows. For example, if you set the style to **Overlap** mode, all windows will be displayed on screen using the overlap method. If you set the style to **Replace** mode, only one display window is shown on screen, and you have to use the window menu to select another display window.

Default state

This field is used to set up the size of the display window while running. You can maximize or minimize the size of display window and you can show the display window at normal size.

Button Setting

This field is used to enable/disable display the window's buttons. You can enable/disable the system menu button. You can also enable/disable the maximize or minimize button of the display window.
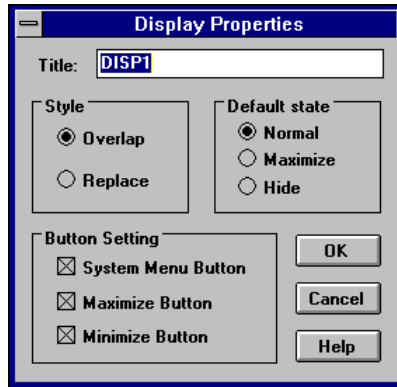
Figure 6-9   Display window properties

Runtime Preference

The Runtime Preference command allows you to configure GeniDAQ with respect to how many errors that will be allowed before Runtime will stop because of excessive errors. A Runtime error is an error that usually occurs as a result of improper settings within GeniDAQ, or because of hardware problems. A list of Runtime errors is included in the appendix of this manual. Acceptable values for "No. of errors allowed before stopping" is between 0 and 32767. This setting is same as the one in Task designer. Please refer to *Chapter 5.2.3 Task Designer Setup menu*.

Change Password

This setting is same as the one of Task designer, please refer to *Chapter 5.2.3 Task Designer Setup menu*.

Administration...

The administrator may add and delete users using this dialog box. To Add and delete users:

1) Enter the Supervisor Password previously entered using the Setup/Change Password dialog box. When GeniDAQ is started for the first time, the Supervisor Password is blank. Once a Supervisor Password is entered by using the Setup/Change Password dialog Box, the Supervisor Password stays in effect unless changed.

   Note: If the Supervisor Password has been inadvertently forgotten, you may start fresh by deleting the SECURITY.PW file located in the GeniDAQ directory.

2) Enter each User ID and corresponding Password by pressing Add User. The user ID and password can consist of as little as no characters or spaces, for which simply pressing the ESC key will unlock the strategy. The user ID and password may contain up to 16 characters or spaces, with no constraints on format.

This setting is same as the one of Task Designer. Please refer to *Chapter 5.2.3 Task Designer Setup menu*.

Network

The Network I/O feature in GeniDAQ allows you to transfer data from any block over a Local Area Network (LAN) that supports Novell's IPX protocol (Novell NetWare is not required). Two blocks are used to support this feature: Network In and Network Out.

Local Station Name

After making sure your network setting are as described above for each machine on your GeniDAQ Network, you must enter the Setup/Network menu to name each station. Each station on the network has to have a unique name. The result will be unpredictable if two stations on the network have the same name. Since the Station Name is stored in the strategy file (.GNI), each station should load and run a different strategy file.

Connection Attempts

After connection, if a network block cannot get any data from a remote station after it tries N times, then the network block will show time-out in the status bar. A zero (0) entry in this text box indicates that as many attempts as are necessary will be made for network connection. Any other trigger in the text box will allow for connection attempts to stop after N number of tries.

This setting is same as that in Task designer. Please refer to *Chapter 5.2.3 Task Designer Setup menu*.

Devices

The Devices command allows you to enter the Device Installation utility from the Task Designer menu bar:

This setting is same as that in Task designer, please refer to *Chapter 5.2.3 Task Designer Setup menu*.

Report

Report generation is one of the main functions in a SCADA system, which records the operation of the system and presents the information in a user-defined format.

Report Designer can be activated in GeniDAQ Task Designer and GeniDAQ Run Time modules only. Report Designer cannot be run alone. Major GeniDAQ modules must be running before Report Designer can be called.

This setting is same as that in Task designer. Please refer to *Chapter 5.2.3 Task Designer Setup menu*.

6.2.4 View Menu
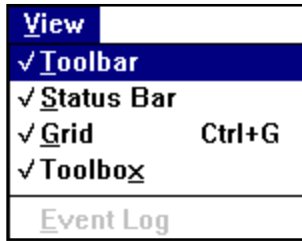
Figure 6-10   Display Designer view menu

Toolbar

To display or hide the Toolbar at the top of the Display Designer screen, highlight this command with your mouse. The Toolbar and associated buttons are explained in the section (below) entitled *Display Designer Toolbar.*

Status Bar

To display or hide a Status bar at the bottom of the Display Designer screen, highlight this command with your mouse. The Status Bar and associated buttons are explained in the section (below) entitled *Display Designer Status Bar.*

Grid

To display or hide grid lines at the display window. These grid lines will be the bases of display item arrangement. If you need finer tuning for arranging display items, please disable this function.

Toolbox

To display or hide the Toolbox, highlight this command with your mouse.

6.2.5 Window Menu
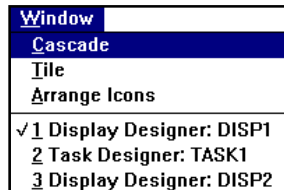
Figure 6-11   Display Designer Window menu

The Window menu includes commands that enable you to arrange multiple windows within GeniDAQ. Standard windows operations are supported (Cascade, Tile, Arrange Icons, and Close All). You can also select between a Strategy Editor Window and a Display Window for a chosen file here.

6.2.6 Run Menu

Figure 6-12   Display Designer Run menu

The Run menu is actually a command that allows you to directly run GeniDAQ while you are in the GeniDAQ design environment. In earlier versions of GeniDAQ you had to start GeniDAQ and jump to Runtime. GENIDAQ 3.x combines design time and runtime into a complete environment. This allows you to run GeniDAQ for testing and debugging any time when you are designing your strategy file.

Start

To start to run strategy file tasks, use this button. After tasks start to run, this option will be disabled.

Stop

To stop running strategy file tasks, use this button. This option is enabled after tasks start.
.
Lock

To lock the screen, thus preventing unauthorized operations while GeniDAQ is in runtime. You can unlock the screen by inputting user ID and password when runtime preferences are set for password checking. You can also press "ESC" to unlock screen when password checking is not enabled.

6.3 Display Item Toolbox

This section provides a description of each of the Display Items that are available in the Display Designer. The Display Items are located in the Toolbox (pictured above), that is located on the left of the Display Designer screen. From this Toolbox, you have at your fingertips all Display Items available in the Display Designer.

Figure 6-13   Display Item Toolbox

Display Designer Items

Anameter Displays
Bar Displays
Button Displays
Conditional Bitmap Displays
Conditional Button Displays
Conditional Text Displays
Group Box Displays
Historical Trending Displays
Knob Control Displays
Numeric Displays
Numeric Control Displays
Indicator Displays
Slider Control Displays
Text String Displays
YT Graph Displays
XY Graph Displays
Rectangle Drawing Displays
Rounded Rectangle Drawing Displays
Oval Drawing Displays
Polygon Drawing Displays
Line Drawing Displays

6.3.1 Bar Graph Display Item

The Bar graph display item allows you to view output data from the task designer icon block during runtime.
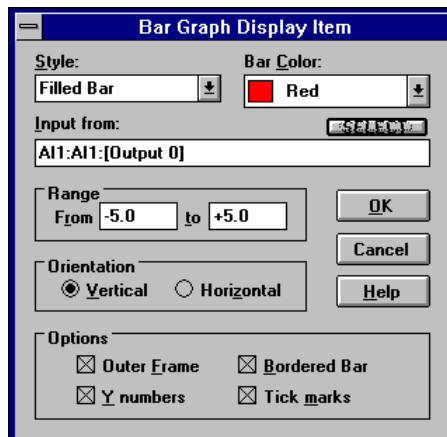
Figure 6-14   Configure Bar Graph Display Item

Input from

The bar graph may be drawn and interfaced to a Task Block variable with a certain tag name. The color and size of the bar graph may be chosen. When in associated configuration dialog box (double-click on the bar graph), you must first choose which Task Designer block's data you would like to display. You can select the available icon blocks by pressing select button and setting the appropriate task/display name, tag name and channel name. The tagged block's dynamic value is displayed during runtime.
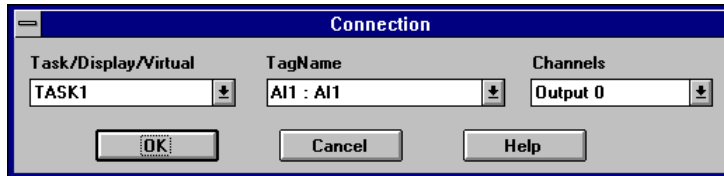
Figure 6-15   Connect Bar Graph display with Task

You can choose the bar's color, the range, the orientation (vertical or horizontal), and the style of the bar graph. The style section is a series of check boxes allowing you to choose whether you see an outer frame, bordered bar, or whether or not you want numbers or tick marks to be displayed.

Style
This field is used to specify the graph style as filled bar or moving mark.

Bar Color
This field is used to specify the color of graph. There are 16 colors provided in GeniDAQ.

Range
This field is used to specify the value range that is displayed by the bar graph.

Orientation
This field is used to specify the orientation of bar graph motion. That is, you can set the bar graph to move either vertically or horizontally.

Options
This field is to enable/disable the display of outer frame, boarded bar, Y numbers and tick marks on the bar graph.

6.3.2 Binary Button Control Display Item

```
 text
```

A binary button control display may be drawn and interfaced to a Task Block variable with a certain tag name. The size of the display may be chosen. This display item is an output from the display. The button is pressed by use of the mouse, and a digital 1 or 0 is sent to the tagged digital Task Block. The Button Display may also be toggled by use of the ENTER key, if the focus is currently on the Button. (The "focus", a standard Windows term, refers to the display item which can currently be controlled by the keyboard). Focus may be shifted from one display item to another using the TAB key. In addition, a "hot key" may be specified to activate the button without changing the focus. This is done by changing the Keyboard Mapping parameter in the dialog box.   Using control buttons, Supervisory Control can be achieved.

The size and properties of the font used in the button text can be changed in this block by pressing the Font... button in the Binary Button Display Item dialog box.
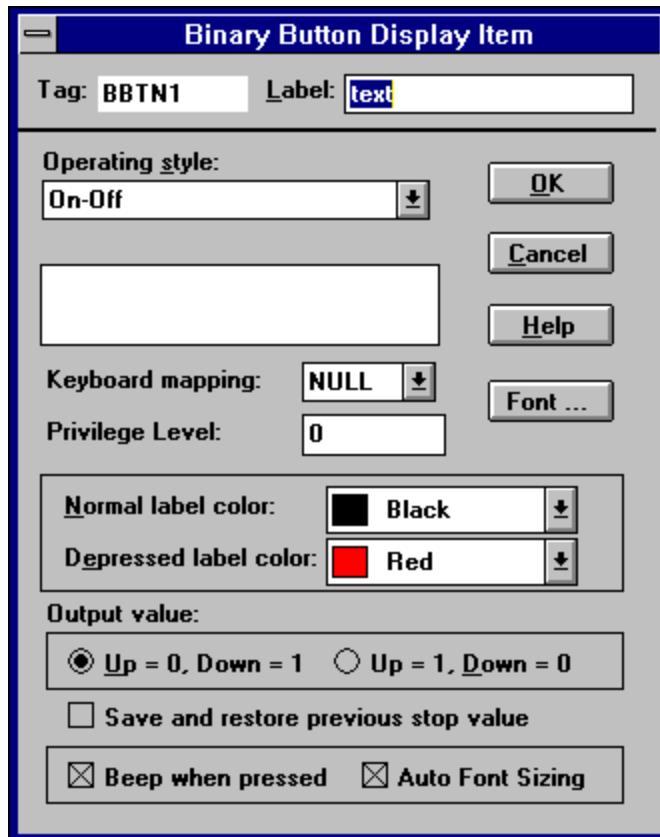
Figure 6-16   Configure Binary Button display item

Tag

This field is used to specify the tag name of the binary button control display item. The tag name is used to represent the output value of this button. If you want to access this button value, you can configure a TAG in task designer to link with display number and tag name and this button value will be available for other blocks in task designer.

Label

This field is used to specify the meaning of the button. The label will be displayed at the front of the button. You can change it using any alphanumeric characters. The maximum number of characters is 30.

Operating Style

This field is used to specify the operating style of binary button to be On-Off, momentary or radio button. For On-Off style, the button will act as a switch to turn on or turn off and output digital 1 or 0 (based on the output value field setting). For Momentary style, the button will act as a pulse; that is it will return to depressed status immediately after the button is pressed. The button will output a down value and an up value immediately (based on the output value field setting). For Radio Button style, the button will be exclusive from other binary buttons. If the button is pressed down, other related buttons will come back up. In this style, you have to configure the related buttons by double clicking buttons in the "Buttons to eject when pressed" field. A start character will pop up at the front of related buttons. The output values of Radio Button style are the same as the On-Off style.
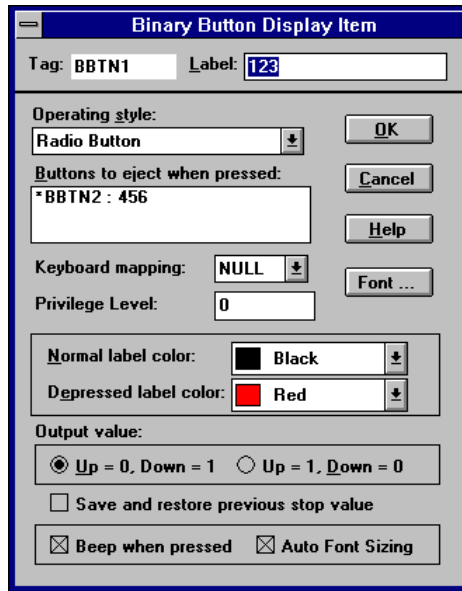
Figure 6-17   Configure Operating Style to be Radio Button

Keyboard mapping

This field is used to set up the linkage between buttons and keywords. You can configure function key ('F2' to 'F8') or alphabetic character ('A' to 'Z') as a shortcut key for this button.

Privilege level

This field is used for protection of system control. The privilege level is from 0 to 255, with larger numbers having the higher privilege. For example, if the privilege level of a button is 100, then the user's privilege must be larger than or equal to 100 to press this button.

Label Color

This field is used to select the normal label color and the depressed label color of the button. GeniDAQ provides 16 colors from which to select.

Output value

This field is used to configure the output value of a button. There are two output modes for your selection. You can select "Up = 0 and Down = 1" or "Up = 1 and Down = 0".

Options

You can enable/disable these option fields as you want.

Save and restore previous stop value

This field is used to save the value of the control display at system stop. This value will be restored at the next system start.

6.3.3 Conditional Button Display Item
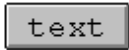
The Conditional Button Display Item has both input and output capability. The **output** may be interfaced to a Task Block variable with a certain tag name. The button is pressed by use of the mouse, and a digital 1 or 0 is sent to the tagged digital Task Block. The Conditional Button Display may also be toggled by use of the ENTER key, if the focus is currently on the Conditional Button. The "focus", a standard Windows term, refers to which display item currently can be controlled by the keyboard. Focus may be shifted from one display item to another using the TAB key. Using conditional buttons, Supervisory Control may be achieved. The **input** may be interfaced to a Task Block variable with a certain tag name and is used for controlling the button's status from the Task (by sending a digital 1 or 0) instead of with the mouse. The size of the display may be re-sized with the mouse.

The size and properties of the font used in the button text can be changed in this block by pressing the Font... button in the Button Display Item dialog box.

Figure 6-18   Configure Conditional Button display item

Tag
This field is used to specify the tag name of the binary button control display item. The tag name is used to represent the output value of this button. If you want to access this button value, you can configure a TAG in task designer to link with display number and tag name and this button value will be available for other blocks in task designer.

Label
This field is used to specify the meaning of the button. The label will be displayed at the front of the button. You can change it using any alphanumeric characters. The maximum number of characters is 30.

Input from

The Conditional Button Display Item may be drawn and interfaced to a Task Block variable with a certain tag name. When in associated configuration dialog box (double-click on the Conditional Button Display Item), you must first choose which Task block's data you would like to display. You can select the available icon blocks by pressing select button and setting the appropriate task/display name, tag name and channel name. The tagged block's dynamic value is displayed during runtime.

Label Color

This field is used to select the normal label color and the depressed label color of the button. GeniDAQ provides 16 colors from which to select.

Privilege level

This field is used for protection of system control. The privilege level is from 0 to 255, with larger numbers having the higher privilege. For example, if the privilege level of a button is 100, then the user's privilege must be larger than or equal to 100 to press this button.

Output value

This field is used to configure the output value of a button. There are two output modes for your selection. You can select "Up = 0 and Down = 1" or "Up = 1 and Down = 0".

Options

You can enable/disable these option fields as you want.

Save and restore previous stop value

This field is used to save the value of the control display at system stop, and the value will be restored at next system start.

6.3.4 Group Box Display Item

A box or frame may be drawn around groups of Display Items to enhance the appearance of your display. No interface with Task blocks is provided. The box may be re-sized, as can the frame width.

Figure 6-19   Configure Group Box display item

Border Width

This field is used to set up the width of a frame. The minimum value is 1 and maximum value is 255.

6.3.5 Numeric/String Display Item

The numeric display item displays output data from the Task block during runtime.

A numeric/string display may be drawn and interfaced with a Task block variable with a certain tag name. The size of the display may be chosen. The data format can be set as floating point (real), integer, or string. You can also set the display format, including the number of digits and location of the decimal point (for floating point format). In addition, you can choose the font, size and color of the numbers or text to be displayed. Justification is also possible.
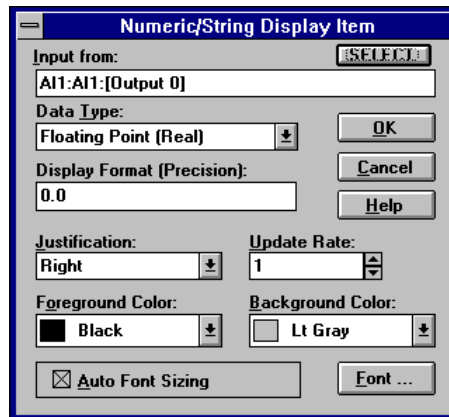
Figure 6-20   Configure Numeric/String display item

Input from
The Numeric/String Display Item may be drawn and interfaced to a Task block variable with a certain tag name. When in associated configuration dialog box (double-click on the Numeric/String Display Item), you must first choose which Task block's data you would like to display. You can select the available icon blocks by pressing select button and setting the appropriate task/display name, tag name and channel name. The tagged block's dynamic value is displayed during runtime.

Data Type
This field is used to select the data type of a display value. The type can be floating point (real), integer or string.

Display Format (precision)
If you select the data type to be floating point or integer, then you can set the display format for output value. For example, set the format to be "0.00"

Justification
You can use this setting to adjust the position of display output in the display field.

Foreground color/Background color

This field is used to select the Foreground color/Background color of numeric/string value. GeniDAQ provides 16 colors to choose from.

Update rate

The Update rate is a divisor that allows the Numeric/String Display block to have a different effective scan rate than the rest of the Task. This is useful if, for example, your Task is running at 100 Hz, but you only want to display data at a rate of 20 Hz. For this example, you would set the Update Rate to 5 (100 divided by 5 (the update rate) gives an effective scan rate of 20 Hz). For this example, only one in five samples would be displayed; the others would be ignored.

6.3.6 Numeric Control Display Item
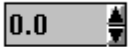
A numeric-type control may be drawn and interfaced to control a Task block variable with a certain tag name. The size of the display may be chosen. This display item is used as an output from the keyboard or mouse; data is to be sent to a Task block variable by an operator who specifies the data. Using this method, Supervisory Control may be achieved. The data format can be set to integer or real. The display format (how many digits and the location of the decimal point) can be set for floating point format only.

The size and properties of the font used in this block can be changed by pressing the Font... button in the Numeric Control Display Item dialog box.
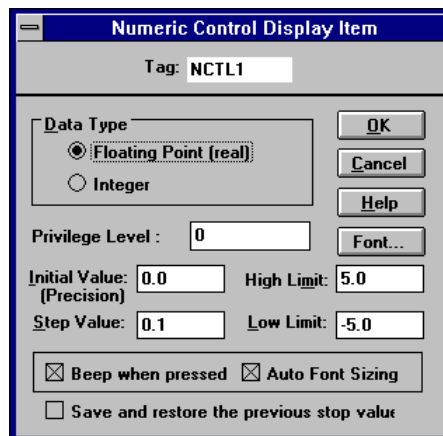
Figure 6-21   Configure Numeric Control display item

Data Type

This field is used to select the data type of a display value. The type can be floating point (real), integer.

Privilege level

This field is used for protection of system control. The privilege level is from 0 to 255, where the larger number has the higher privilege. For example, if the privilege level of a button is 100, then a user's privilege must be larger than or equal to 100 to press this button.

Initial Value

This field is used to set the initial value of the numeric control field.

Step Value

This field is used to set the value of the numeric control field for each step.

High limit/Low limit

This field is used to set high/low limit values of the numeric control field.

Save and restore previous stop value

This field is used to save the value of the control display at system stop, and the value will be restored at next system start.

6.3.7 Indicator Display Item

An LED indicator, displaying the output state of a tagged digital block, may be simulated here by specifying a digital tag name from the Task variable. A digital 1 turns on the indicator and a 0 turns it off. The color and size of the indicator may be chosen.
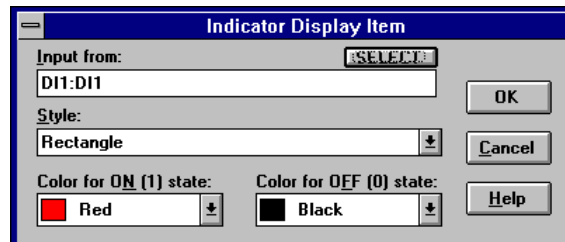
Figure 6-22   Configure Indicator display item

Input from

The Indicator Display Item may be drawn and interfaced to a Task block variable with a certain tag name. When in associated configuration dialog box (double-click on the Indicator Display Item), you must first choose which Task block's data you would like to display. You can select the available icon blocks by pressing the select button and setting the appropriate task/display name, tag name and channel name. The tagged block's dynamic value is displayed during runtime.

Style

This field is used to select the graph for indicator display. There are two kinds of graphs: Rectangle, Round or Ellipse.

Color for On/Off State

This field is used to select the colors for On/Off value of indicator display. GeniDAQ provides 16 colors to choose from.

6.3.8 Text String Display Item

```
text
```

A text string label may be entered for display purposes only. No interface to Task blocks is provided. Within the Text String Display Item, you can choose the font, color and size of the displayed text by pressing the Font... button in the Text String Display Item dialog box.
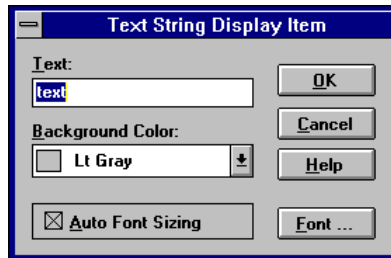
Figure 6-23   Configure Text String display item

Text

This field is used to input desired text for display.

Background Color

This field is used to specify the background color of this text field. If you want to specify the text color, you can assign it by disabling **Auto Font Sizing** and selecting the **Font...** button.
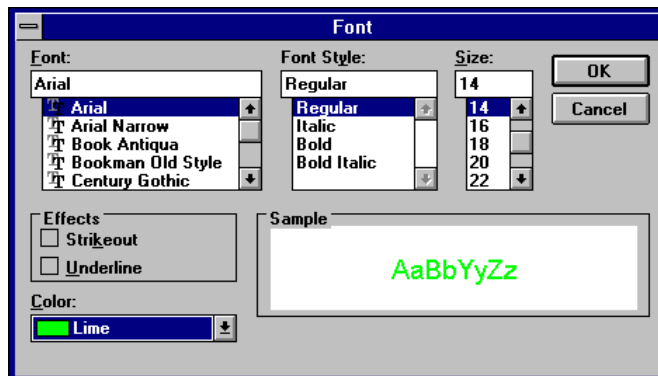
Figure 6-24   Configure Text String font

6.3.9 XY or YT Graph Display Item

An XY Graph or a YT graph may be drawn and interfaced with one or many Strategy Block variable(s) with a certain tag name or names. The color and size of the graph may be chosen.   A Y-Time Graph displays data from any number of Strategy Blocks on the Y-axis, against time on the X-axis. The XY Graph displays data from two Strategy Blocks on its axes.
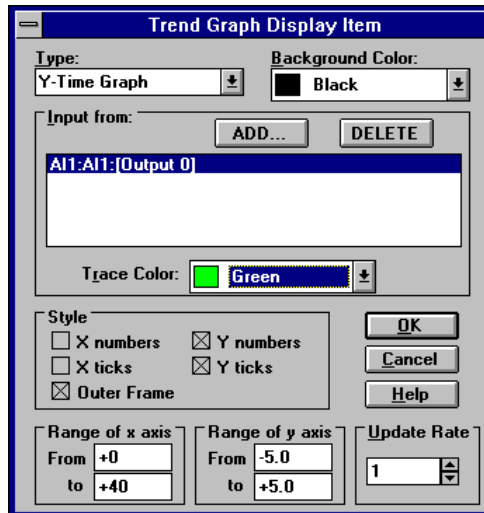
Figure 6-25   Configure Trend Graph display item

YT Trend Graph

For the YT Trend graph, you can choose the colors and the ranges that will be displayed. You next need to choose which Strategy Editor block's data you would like to display, corresponding with your chosen trace colors. Available icon blocks are displayed in a list box labeled "Input Blocks". Any number of blocks can be interfaced for display in one graph (however, eight seems to be the practical limit). By double clicking on each desired icon block tagname, tagnames are selected for graphical display. When blocks are selected, an asterisk appears to the left of the tagname. As you select tagnames for display, you can choose separate trace colors for each tagname. The color for a trace is displayed and can be changed each time a selected tagname is highlighted with the mouse. You can also change the style of your graph. The style section is a series of check boxes displaying available options.

The Update rate is a divisor that allows the YT Trend Graph Display item to have a different effective scan rate than the rest of the Task. This is useful if, for example, your Task is running at 100 Hz, but you only want to display data at a rate of 20 Hz. For this example, you would set the Update Rate to 5 (100 divided by 5 (the update rate) gives an effective scan rate of 20 Hz). For this example, only one in five samples would be displayed; the others would be ignored. Using the update rate option in this way will alleviate a flickering effect when using a very fast scan rate.

XY Graph

The XY graph allows you to plot data from two separate blocks against each other. The colors of the trace and background can be chosen, as well as the length of the trace. Using a trace with a smaller number of points will use less machine memory, and your display will run much more efficiently.

The graph style can be selected, giving you flexibility in your display, allowing you to show ticks and/or numbers on the axes. In addition, ranges for both axes can be selected. This permits you to make your display any size or scale.

Note: Keep in mind that when using an X-axis with a large scale (i.e. long time spans), and especially if you are displaying more than one trace, memory requirements are increased. All data points that are displayed must reside in memory; the more data points you display, the more memory you will need.

Background color

This field is used to specify the color of background for a given trend display.

Input from

The XY or YT Graph Display Item may be drawn and interfaced to a Task block variable with a certain tag name. When in the associated configuration dialog box (double-click on the XY or YT Graph Display Item), you must first choose which Task block's data you would like to display. You can select the available icon blocks by pressing the select button and setting the appropriate task/display name, tag name and channel name. The tagged block's dynamic value is displayed during runtime.

You also can add or delete the input data using the ADD or DELETE buttons. When you add input data into an XY or YT Graph Display, you have to specify the color of the trend line. You are advised to choose different colors to represent different input data.

Style

This field is used to enable/disable several display items in an XY or YT Graph Display chart, including X numbers, Y numbers, X ticks, Y ticks and outer frame.

Range of X/Y axis

These fields are used to specify the maximum and minimum value ranges of X and Y display data.

Update rate

The Update rate is a divisor that allows the Numeric/String Display block to have a different effective scan rate than the rest of the Task. This is useful if, for example, your Task is running at 100 Hz, but you only want to display data at a rate of 20 Hz. For this example, you would set the Update Rate to 5 (100 divided by 5 (the update rate) gives an effective scan rate of 20 Hz). For this example, only one in five samples would be displayed; others would be ignored.

6.3.10 Knob Control Display Item

A numeric-type knob control may be drawn and interfaced (Output) to control a Task block variable with a certain tag name. The size of the display may be adjusted. This display item is used as an output from the keyboard or mouse; data is to be sent to a Task block variable by an operator who specifies the data. The knob may also be turned using the keyboard UP and DOWN arrow keys, if the focus is currently on the knob to be turned. (The "focus", a standard Windows term, refers to which display item currently can be controlled by the keyboard). The TAB key can be used to shift the focus from one item to another.

Knobs can be used to achieve supervisory control. The data is in real (floating point) format, and the display format (number of digits and location of the decimal point) may be chosen. The size and properties of the font used in this block can be changed by pressing the Font... button in the Knob Control Display Item dialog box.

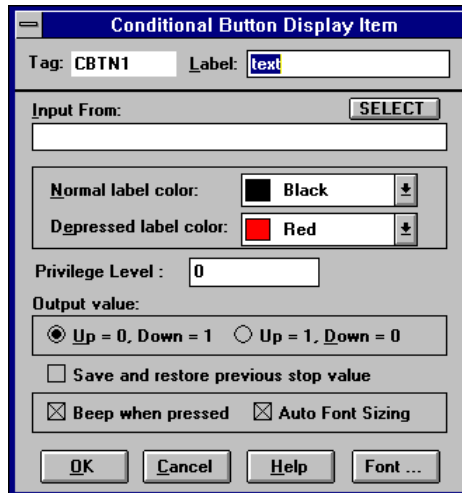*Figure 6-26   Configure Knob Control display item*

Display Current Value

Specify whether the numeric display below the knob is active or not.

Knob Action

Specify the knob action (either smooth or incremental).

Decimal Places

Specify the desired number of decimal places after the decimal point.

Initial Value

Specify to which numeric value the knob will point at Runtime Startup.

Privilege level

This field is used for protection of system control. The privilege level is from 0 to 255, with the larger number having a higher privilege. For example, if the privilege level of a button is 100, then a user's privilege must be larger than or equal to 100 to press this button.

Show Tics

Specify whether Tics will be displayed.

Start Tics

Specify the lowest number the knob will output (i.e. where the Tics will start).

End Tics

Specify the highest number the knob will output (i.e. the Tics ending number).

Tics Rate

Specify the rate at which the Tics will increment.

Save and restore previous stop value
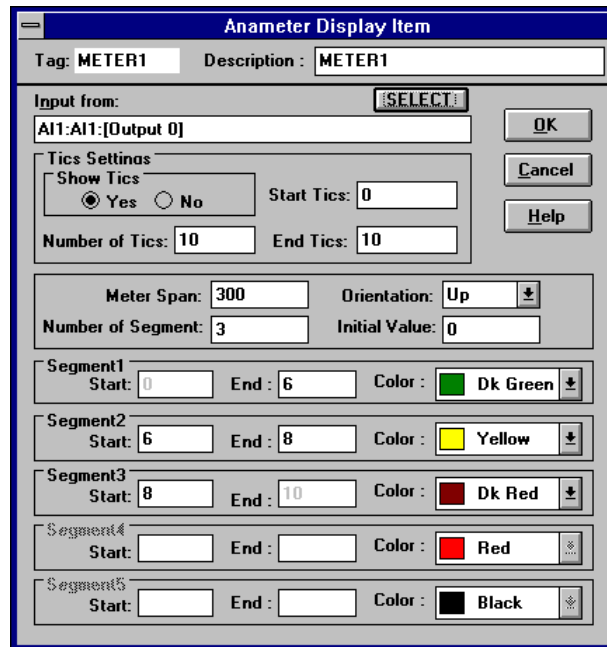
This field is used to save the value of the control display at system stop, and the value will be restored at next system start.

6.3.11 Anameter (Analog Meter) Display Item

The Anameter display item displays output data from the Task block during runtime in the form of an analog meter. The Anameter may be drawn and interfaced to a Task block variable with a certain tag name. The number, color and size of the color segments may be chosen. When in the associated configuration dialog box (double-click on the Anameter), you must first choose which Task block's data you would like to display.

Available icon blocks are displayed in a list box labeled "Input from". The tagged block's dynamic value is displayed during runtime. You can choose the segment color, range, orientation (vertical or horizontal), and the initial Anameter value. Up to five segments can be specified. The style section is a series of check boxes allowing you to choose whether you see an outer frame, bordered bar, or whether or not you want numbers or tick marks to be displayed.

Figure 6-27   Configure Anameter display item

Show Tics
Specify whether Tics will be displayed.

Start Tics
Specify the Tics start number.

End Tics
Specify the Tics ending number.

Number of Tics
Specify how many total Tics

Meter Span
Specify the total numeric span of the meter.

Orientation
Specify the meter orientation (Up, Down, Right, or Left).

Initial Value
Specify the value to which the meter needle will point upon Runtime Startup.

Number Of Segments
Specify the total number of color segments (up to five) for the span of the meter.

Segment 1-5 Start, End, and Color
For each enabled meter segment, specify the numeric start, end, and color of the meter segment.

6.3.12 Slider Control Display Item

A numeric-type slider control may be drawn and interfaced (Output) to control a Task block variable with a certain tag name. The display and slider size may be changed. This display item is used as an output from the keyboard or mouse. Data is to be sent to a Task block variable by an operator who specifies the data.

The slider may also be moved using the UP and DOWN arrow keys, provided the focus is currently on the slider (The "focus", a standard Windows term, refers to which display item currently can be controlled by the keyboard). Choosing which display item currently has the focus may be done using the TAB key. The slider control can be used to achieve Supervisory Control. The data is in real (floating point) format.

Figure 6-28   Configure Slider Control display item

Slider Action
Specify the slider action (either smooth or incremental).

Initial Value
Specify to which numeric value the slider will point at Runtime Startup.

Privilege level
This field is used for protection of system control. The privilege level is from 0 to 255, with the larger number having the higher privilege. For example, if the privilege level of a button is 100, then the user's privilege must be larger than or equal to 100 to press this button.

Show Tics
Specify whether Tics will be displayed.

Tics Start
Specify the lowest number the slider will output (i.e. where the Tics will start).

Tics End
Specify the highest number the slider will output (i.e. the Tics ending number).

Tics Number
Specify how many total Tics will be displayed. This entry will be grayed out if "Show Tics" is set to "No".

Save and restore previous stop value
This field is used to save the value of the control display at system stop, and the value will be restored at next system start.

6.3.13 Conditional Bitmap Display Item

This display item has input capability. It accepts a value between zero (0) and seven (7) from a Task block, each value providing capability to select a bitmap file to be displayed during Runtime. When this display item is double-clicked upon, a dialog box will appear displaying all bitmaps currently installed.
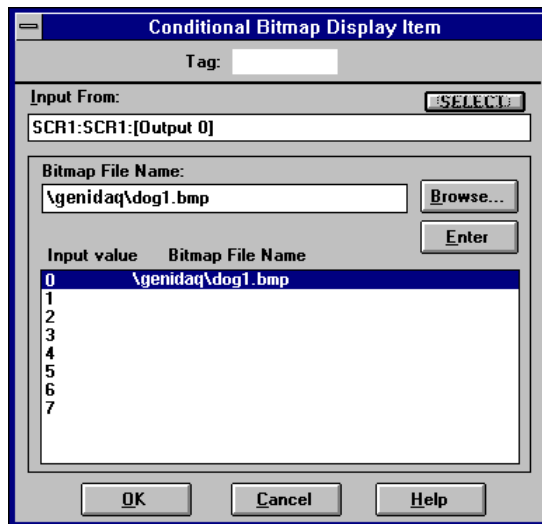
Figure 6-29   Configure Conditional Bitmap display item

Bitmap File Name

Specify the Bitmap file to be selected corresponding with the currently highlighted Input Value. Enter the full path, or Browse to locate the desired Bitmap file. For setting corresponding the bitmap filename, you have to select the related input value and select bitmap file, then press the ENTER button to configure it.

6.3.14 Historical Trending Display Item

A Historical Trending Display may be drawn and interfaced with one or many Task block variable(s) with a certain tag name or names. The size of the graph may be chosen. A Historical Trending Graph displays data from up to eight Task blocks on the Y-axis, against time on the X-axis.

You need to choose which Task block's data you would like to display, corresponding with your chosen trace colors. Available icon block tag names are displayed in a list box labeled "Historical Trending Channel". Up to eight blocks can be interfaced for display in one graph. By double clicking on each desired icon block tagname, blocks are selected for graphical display. When blocks are selected, a double asterisk appears to the left of the tagname. As you select tagnames for display, you can choose separate trace colors for each tagname. The color for a trace is displayed and can be changed each time a selected tagname is highlighted with the mouse.
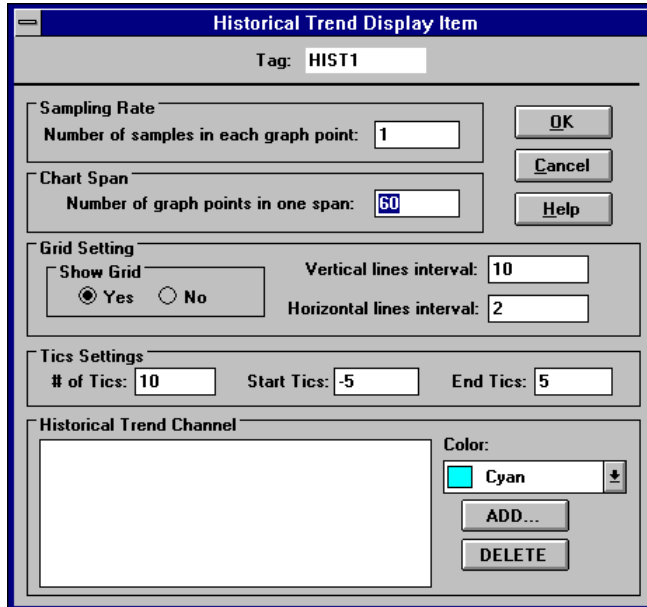
Figure 6-30   Configure Historical Trend display item

Number of Samples for Each Graph Point
A divisor similar to the "Update Rate" in many blocks. For each point in the Historical Trending Graph, you can specify how many actual strategy samples will pass before the point is displayed.

Number of Graph Points in One Span
Specify the total number of Graph Points that will be displayed at once, (the "Span" of the graph)

Show Grid
Specify whether or not to display the grid.

Vertical/Horizontal Resolutions
Specify how many Tics between each grid line.

# of Tics
Specify how many Tics for the vertical (Y-axis) will be displayed.

Start Tics
Specify the Tics start number.

End Tics
Specify the Tics ending number.

Historical Trend Channel
This field is used to add input data for displaying its historical values. You can add a new input or delete an existing input. You can also specify the pen color of each data to draw the historical trend.

6.3.16 Conditional Text Display Item

text

This display item has input and output capability. It accepts a value between zero (0) and seven (7) from a Task block, each value providing capability to select text to be displayed. In addition, the string may be output to a block accepting a string as input, such as to the RS-232 block. When this display item is double-clicked upon, a dialog box will appear displaying all text currently installed.
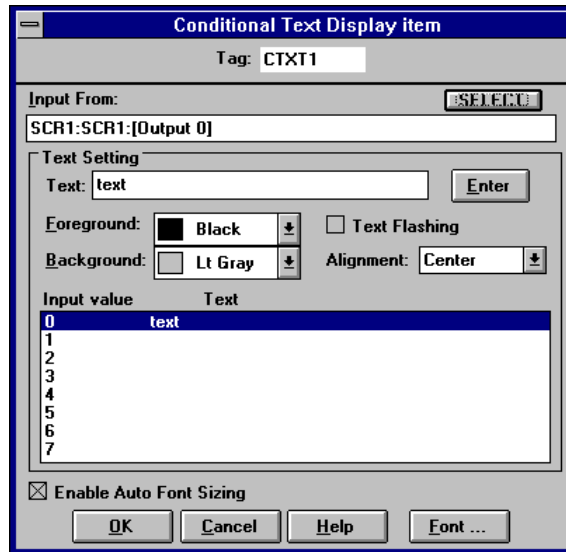
Figure 6-31   Configure Conditional Text display item

Displayed Text
Specify the Text file to be selected corresponding with the currently highlighted Input Value.

Text/Background Color
Specify the Text and Background color for the currently highlighted Input Value.

Alignment
Specify the Text Alignment for the currently highlighted Input Value.

6.3.17 Menu Display Item

MENU

This menu item is used while the system is in Runtime mode. There are two functions provided by this menu button. One is to start/stop/halt/resume system execution. Another is to switch between multiple display windows.
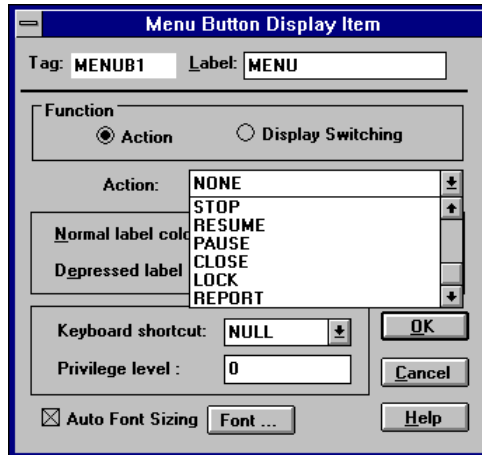
Figure 6-32   Menu Button Display Item

Function

You can set the menu button to act as a display switch or to control system operation. If you select 'action' mode, you can stop, pause or resume system operation while in Runtime mode. In the 'action' mode, you can also call reports directly from GeniDAQ Runtime.

If you select 'display switching' mode, you can specify display window names (for example, "DISP1") in this field and use this button to switch between displays while in Runtime.

6.3.18 Rectangle Drawing Display Item

This display item is used to draw a rectangle graphic object in the display window. When a graphic is selected, you can drag the corner to size the object. This display item has input capability. It accepts a value between zero (0) and one (1) from a Task block, each value providing capability to select color to be displayed. When this display item is double-clicked upon, a dialog box will appear for configuration.
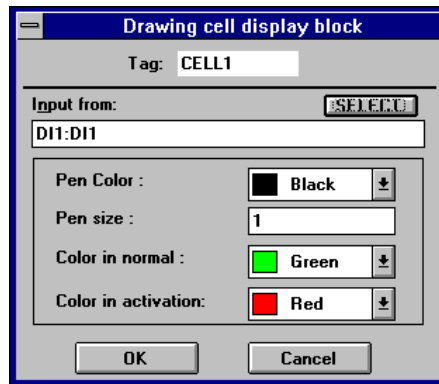
Figure 6-33   Configure Rectangle Drawing display item

Input from
Specify the input from value.

Pen Color
Specify the color of pen to fill the rectangle graphic object. There are 16 colors from which to choose.

Pen Size
Specify the size of pen to draw the shape of graphic object. The unit of size is dots.

Color in Normal
Specify the display color of the graphic object while the input value is equal to 1.

Color in Activation
Specify the display color of the graphic object while the input value is equal to 0.

6.3.19 Rounded Rectangle Drawing Display Item

This display item is used to draw a rounded rectangle graphic object at the display window. The functions of the rounded rectangle drawing tool are the same as those of the rectangle drawing tool. Please refer to the previous section.
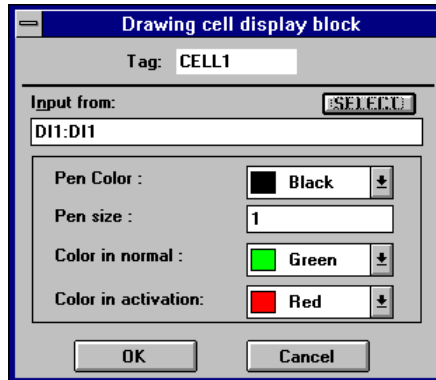
Figure 6-34   Configure Rounded Rectangle Drawing display item

Input from
Specify the input from value.

Pen Color
Specify the color of pen to fill the rectangle graphic object. There are 16 colors from which to choose.

Pen Size
Specify the size of pen to draw the shape of graphic object. The unit of size is dots.

Color in Normal
Specify the display color of the graphic object while the input value is equal to 1.

Color in Activation
Specify the display color of the graphic object while the input value is equal to 0.

6.3.20 Oval Drawing Display Item

This display item is used to draw an oval graphic object at the display window. The functions of the oval drawing tool are the same as those of the rectangle drawing tool. Please refer to the previous section.
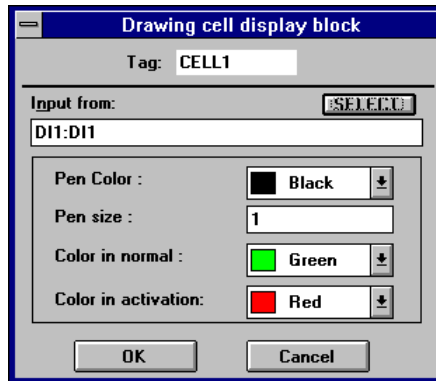
Figure 6-35   Configure Oval Drawing display item

Input from
Specify the input from value.

Pen Color
Specify the color of pen to fill the rectangle graphic object. There are 16 colors from which to choose.

Pen Size
Specify the size of pen to draw the shape of the graphic object. The unit of size is dots.

Color in Normal
Specify the display color of the graphic object while the input value is equal to 1.

Color in Activation
Specify the display color of the graphic object while the input value is equal to 0.

6.3.21 Polygon Drawing Display Item

This display item is used to draw a polygon graphic object at the display window. The functions of the polygon drawing tool are the same as those of the rectangle drawing tool. Please refer to the previous section.
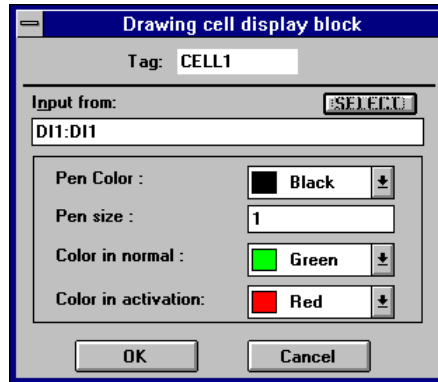
Figure 6-36   Configure Polygon Drawing display item

Input from
Specify the input from value.

Pen Color
Specify the color of pen to fill the rectangle graphic object. There are 16 colors from which to choose.

Pen Size
Specify the size of pen to draw the shape of graphic object. The unit of size is dots.

Color in Normal
Specify the display color of the graphic object while the input value is equal to 1.

Color in Activation
Specify the display color of the graphic object while the input value is equal to 0.

6.3.22 Line Drawing Display Item

_____

This display item is used to draw a line graphic object at the display window. The functions of the line drawing tool are the same as those of the rectangle drawing tool. Please refer to the previous section.
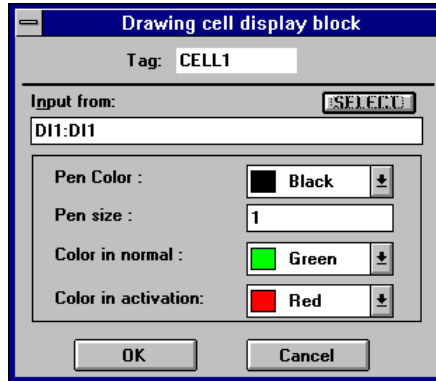
Figure 6-37   Configure Line Drawing display item

Input from
Specify the input from value.

Pen Color
Specify the color of pen to fill the pen color graphic object. There are 16 colors from which to choose.

Pen Size
Specify the size of pen to draw the shape of graphic object. The unit of size is dots.

Color in Normal
Specify the display color of the graphic object while the input value is non-zero.

Color in Activation
Specify the display color of the graphic object while the input value is equal to 0.

7

Report Designer

7 Report Designer

7.1 Report Designer Overview

The Report Designer provides a user-configurable environment in which to define the report contents. It collects the TAG channel data at specific time intervals and automatically prints user reports at a predefined time. Users can take advantage of the interfaces provided by the Report Designer to view and print old reports manually. There are four main components in Report Designer:

Report parameters and format configuration

Provides user interface dialog boxes that allow users to set up the report format and print time. Report format entries are organized in a table and users edit each column with text characters or specified keywords for the defined TAG point data. Information of each report format is stored into a format file, and extracted during report generation.

Data collection

Data collection is activated by Report Designer's internal timer, which is set for a 10-second interval. Data collection records TAG point data in a daily data storage database file at the user specified time. Report Designer's data collection is designed solely for report generation, while high-speed data collection should be handled by GeniDAQ 3.x's other trend data collection functions.

Report Scheduler

Report Scheduler monitors the report print time during the day. At a user-defined time, Report Scheduler activates the Report Generation component to generate the required report. Report Scheduler also informs users of report printing status.

Report Generation

The Report Generation component combines the format file and data collection database file, and sends the user desired report to the printer. The report is currently limited to a tabular format report. A graphical report with daily trend output is currently under development.

Data Collection and Report Scheduler components are hidden from the user and are activated only at GeniDAQ Runtime. Users can configure each report's parameters and format from the Report Designer's user interfaces.

7.1.1 Report Types

Report generation is one of the main functions in a SCADA system. Report generation records the operation of the system and presents the information in a user-defined format. There are two main types of reports, the *System Operation Status Report* and the *System Operation Summary Report*.

System Operation Status Report reflects the current running status of all equipment in the system. Users may need to record status of a device or system throughout the day. This information is useful for system operators to examine if the equipment is normally running during the day.

System Operation Summary Report records the operation summary information of an equipment after a certain period of operation. Summary information includes the maximum, minimum and average values during the period. This information is useful for the system operator to determine the system peaks and any abnormal condition during the recorded period.

Report Designer provides four different types of reports according to the scheduled print time:

Fixed Time Report
Reports are printed at user-defined times. Up to 24 reports' timing can be scheduled during a day. Fixed Time Reports are designed for System Operation Status Reports, which record real-time status of all TAG points at specific times during the day.

Daily Report
Daily Reports are designed for System Operation Summary Reports in a daily period. User can only set up one time during the day for the report to be printed. Report Designer will always print the summary report for the previous day at a user-defined time. A user may, for example, set the daily report print time to 1:00 am every day. On 2 January at 1:00 am, Report Designer would print the daily operation summary for 1 January.

Monthly Report
Monthly Reports are designed for System Operation Summary Reports in a monthly period. Users can only set up one time during the month for the report to be printed, which includes the day of the month and the time of the day. Report Designer will always print the summary report for the previous month at the user-defined time.

Yearly Report
Yearly Reports are designed for System Operation Summary Reports in a yearly period. Users can only set up one time during the year for the report to be printed, which includes the month of the year, the day of the month and the time of the day. Report Designer will always print the summary report for the previous year at the user-defined time.

7.1.2 Formatted Reports

To deal with various requirements and different equipment, users may require numerous kinds of reports to finish their jobs. The major advantage of Report Designer is the flexibility in report formation. Users can create their own report in minimal time without redesign their system configuration. The main idea behind the flexible report generation is the combination of the archived TAG point database with the user-defined format file. Each report format is kept in a report format file. When a report is printed, Report Designer combines the report format file with available TAG point database to generate the required report.

Report Designer views its report in a tabular form, with all printed information organized in different boxes in the table.

7.2 Installation

Report Designer is installed as a part of the general GeniDAQ installation.

7.2.1 Directory Set Up

As the Report Designer is installed, related executable files and working database files will be copied to user-defined directories. There are three kinds of directories set up for the Report Designer:

Standard Directory

Working Directory

Database Directory

Standard Directory contains all the system standard report format files and empty TAG point database templates. Report Designer will use the database template files to create daily, monthly and yearly storage databases in the Database directory defined by users. Standard report format files are a pool of operating report formats. Users can refer to these report formats for future report development. System developers can put their system specific standard reports into the standard directories for user selection. When Report Designer is installed, a standard directory will be created under the directory where GeniDAQ was installed.

Working Directory is the main directory for Report Designer to record all report schedules and locate the TAG point storage databases. Users must define the path of report working directory for each individual GeniDAQ strategy. The working directory information of each strategy will be maintained by the Report Designer. This information is accessed when users open an old strategy. Report Designer will automatically load in the corresponding report schedule files. Users should note that when two GeniDAQ strategies share the same working directory, they will operate under the same report schedule.

Database Directory contains the daily and monthly TAG point storage database files. When users define the working directory for a GeniDAQ strategy, the database directory is automatically created under the Report working directory.

The following diagram represents the directory tree of Report Designer operation:
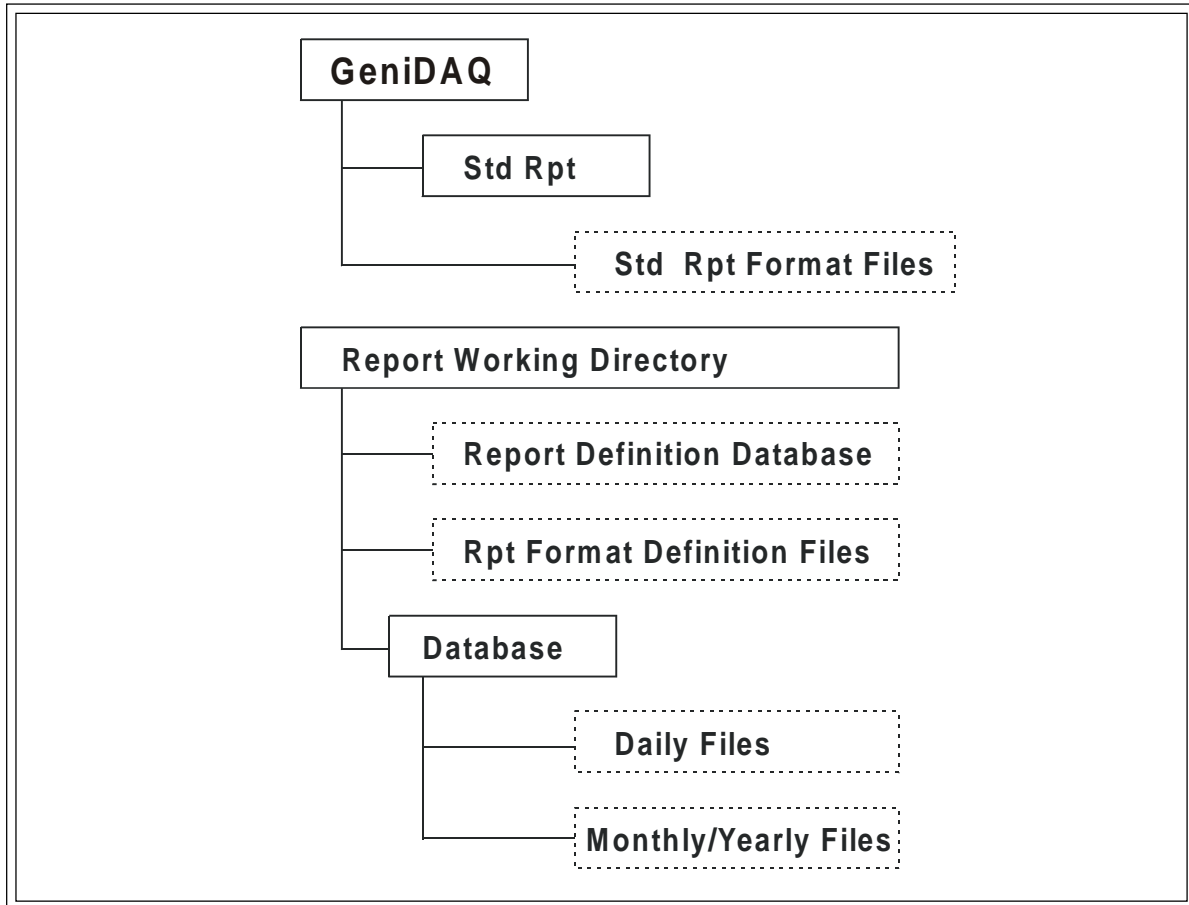
Figure 7-1   Report Designer operation directory

7.3 User Interfaces

7.3.1 Starting Up

Report Designer can be activated in the GeniDAQ Builder and GeniDAQ Runtime modules only. Report Designer cannot be run alone. Major GeniDAQ modules must be running before Report Designer can be called.

The GeniDAQ Builder can be used to set up report parameters, schedule and format. No TAG point data archives and automatic report scheduling will be performed.

After opening a strategy file, users can select Report from Setup in the editor's menu bar. The Report Designer checks if the working directory has been set up for the current strategy file. If no working directory is set, Report Designer displays the Report Parameter Set Up Dialog Box for the user to configure the current working directory. When the current strategy's report parameters are set, Report Designer displays the report schedule dialog box for user interaction.
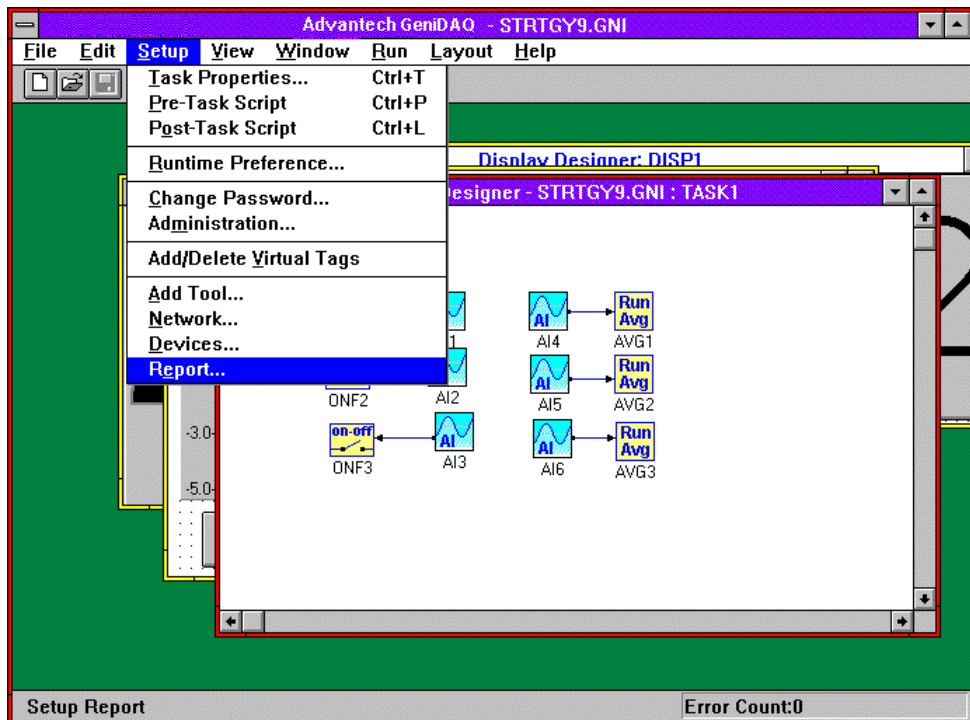
Figure 7-2   Run Report Designer from GeniDAQ Builder

Note: *The Report... option in the Setup menu is enabled after the strategy file is saved*

To allow Report Designer to be run during GeniDAQ Runtime, users can configure a Menu Button in any user-designed display. Users must set the Action of the Report Menu Button to "REPORT" during display editing. At GeniDAQ Runtime, users can push the specific Report Menu Button to activate the Report Designer. Report data collection and report print out scheduling can only be performed during GeniDAQ Runtime.
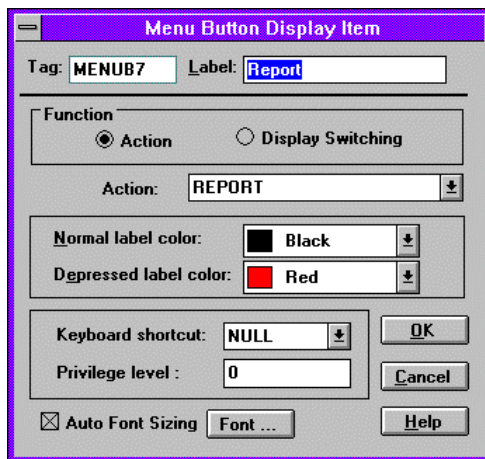
Figure 7-3   Run Report Designer from menu button

Once the Report Designer is run, users should minimize the task and let it run as a background task. This allows the report data collection to collect required data at fixed intervals and allows the report scheduler to perform automatic report print out.

Report Designer will terminate itself when a user changes the strategy file or exits GeniDAQ.

Report Scheduler

Report Scheduler Dialog Box is the main entry to the Report Designer. When the report scheduler is activated, a list of today's scheduled report is listed in the center of the dialog box. User can monitor or check the report printout status from the list. A column of menu buttons is designed at the left-hand side, which allow users to perform required functions:

1) Fix Time Report menu button

Calls the Fix Time Report Management dialog box. Allows user to add, delete and modify fix time report schedules and format.

2) Daily Report menu button

Calls the Daily Report Management dialog box. Allows user to add, delete and modify daily report schedules and format.

3) Monthly Report menu button

Calls the Monthly Report Management dialog box. Allows user to add, delete and modify monthly report schedules and format.

4) Yearly Report menu button

Calls the Yearly Report Management dialog box. Allows user to add, delete and modify yearly report schedules and format.

5) Set Up menu button

Calls the Report Parameter Set Up dialog box. Allows user to change the report set up parameters for current GeniDAQ strategy file.

6) Minimize menu button

Minimizes Report Designer. Report Designer will not be terminated but run as a background task for data collection and report printout.

A select box is provided for the user to enable or disable the report scheduler. This select box is only available at GeniDAQ Runtime. When users choose to disable the report scheduler, no automatic report print out will be generated. However, data collection will still be performed by the Report Designer when scheduler is disabled.
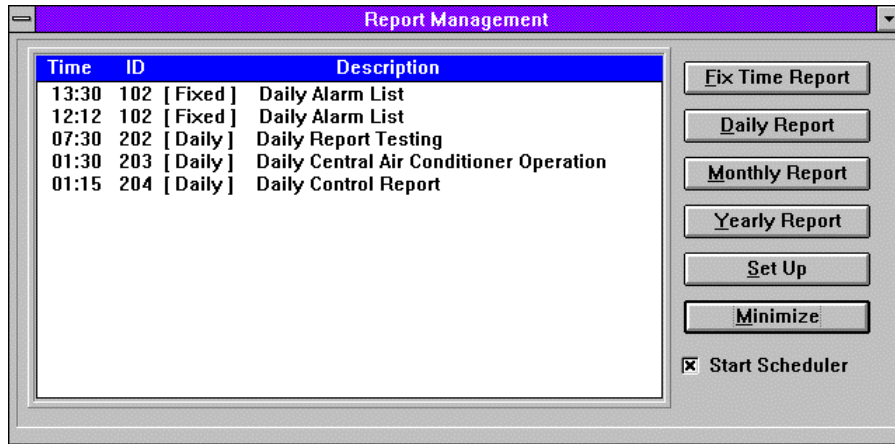
Figure 7-4   Report Designer main screen

Report Designer Parameters Set Up

Report Designer Parameters Set Up dialog box is displayed when a user selects the Set Up Menu Button in the Report Scheduler Dialog Box. It is also displayed when Report Designer finds that there is no working directory defined for the current GeniDAQ Strategy file. Three important pieces of information can be entered in the dialog box: the report working path, security level and report selection listing length.

User can access the path entry in the dialog box to set up the report working directory for the current strategy file. This information will be saved and when same strategy file is opened again, Report Designer will recall the working path for the old strategy file. When the same path is defined for more than one strategy file, these strategies share the same working directory and report schedule.

Access to Report Designer's functions is controlled by the user's login privilege level. Users who do not have system administrator authority can only view and print the existing report at run time. System Administrator can add, delete or modify any report configuration at run time. The access Security Level entry box allows the administrator to change the access level of Report Designer. Users with higher level authority (i.e. user login code is larger than the defined value) are treated as a system administrator by the Report Designer.

Report Listing Selection controls the type of previous reports available in the Report Print Dialog Box. Three entries are provided for the daily, monthly and yearly reports respectively.
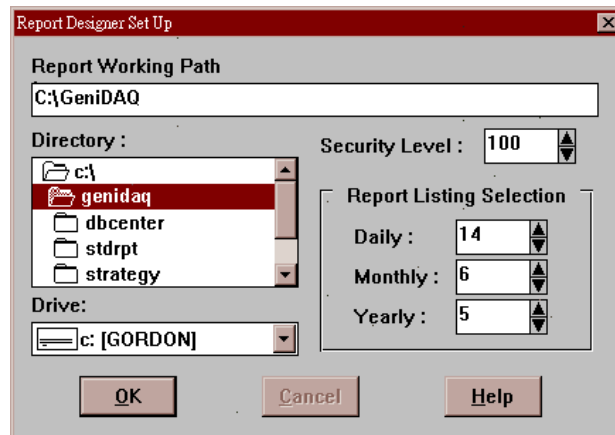
Figure 7-5   Setup Designer parameters

7.3.2 Create Report

To create a report in the Report Designer, users must decide which kind of report is required from the Report Scheduler dialog box. Refer to *Section 7.1.1 Report Types* for the differences in report type definitions employed in Report Designer.

When users push the required report type menu button, the Report Designer main screen is displayed. In the center of the dialog box, a list of all defined reports of the specific type is displayed. Each report has a corresponding ID number. This ID number is maintained by the Report Designer and used internally to identify individual reports. Report Status shows if the report is disabled or enabled. When a report is disabled, no print out can be generated for it.

Four user actions are available in the menu:

Print	Print selected report (enabled)
Add	Add new report
Configure	Configure and modify currently selected report
Delete	Delete selected report

Only a system administrator can add, delete or modify any report configuration. If the user is not logged in as a system administrator, only the Print button is enabled.

To add a new report, users can push the Add Menu Button. The Report Parameter Configuration Dialog Box of the respective report type is displayed.
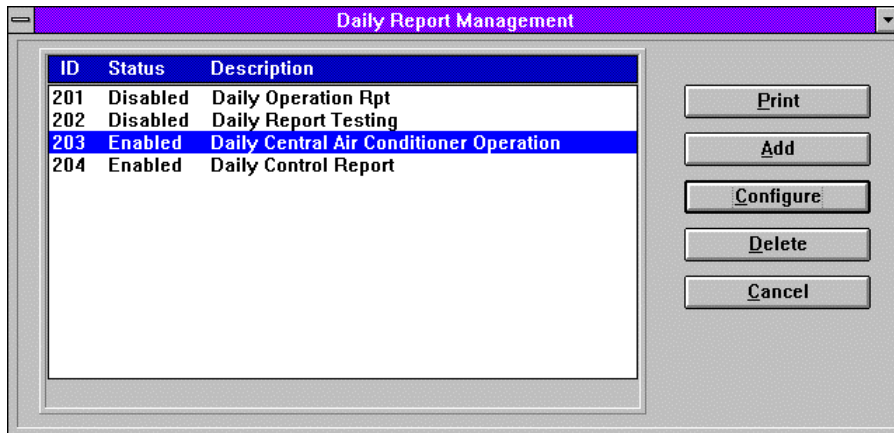
Figure 7-6   Configure/Add Daily Report

Report Parameters Configuration

Report Designer requires users to define each report's parameters in the Report Parameter Configuration Dialog Box. Each report defined in the system carries a unique ID number, which is used internally to identify the report. This ID number is automatically generated by the software. Users can modify the ID number from the entry box; however, it is better to leave the ID for the system.

Users can enter a Report Description for the respective entry. The descriptive text is used in the report listing selection to help identify each report. Maximum number of characters allowed in the description entry is 50.

Users decide the report type for the newly created report in the Report Type selection. Two types of reports are available: Listing Report and Formatted Report. Listing Reports are provided in a form of executable files, which gather alarm information and produce the alarm summary report. Formatted Report allows user to configure user report format.

When users select the Formatted report, the list in the center of dialog box displays currently available formats for user selection. There are two source directories of available report formats from which users may choose. When User is selected from the format location radio button, all current report formats in the working directory are listed. When the user selects the System radio button, standard system report templates from the Standard Directory are listed. User can select one of the existing format files from the list, or create a new one by not selecting any of the report formats from the list.

To edit the format, push the Format Push Button. The Format editing dialog box will be displayed.

Figure 7-7   Configure Report parameters

Print Status selection allows user to enable or disable the report print out action. When users choose to disable a report, no report printout is generated either automatically or manually. This function allows users to create different kinds of reports as required and enable the report only if needed.

After the report parameter editing is finished, users should save the configuration to respective database files by pushing the Save menu button.

The report print time setup is discussed in next section. Different report types have different report print time setup requirements.

Report Print Time

As mentioned in the previous section, report print time entries vary according to report type. The following diagram shows the report print time entries for Fixed Time Report.

Figure 7-8   Fixed Time Report Parameter Configuration dialog box

Up to 24 Fixed Time reports' timing can be scheduled during the day. Users enter the print hour and minute in the available entry boxes. Users must enter the report time based on sequential number order. Reports are printed at exact minute times.

For the Daily Reports, user can only define the time of the day when the report is required. The Daily Report Parameter Configuration dialog box allows only an hour entry and a minute entry. Report Designer prints the report at a user-defined time each day.

For Monthly Reports, user defines the time and day of the month when the monthly report is required. For Yearly report users are required to enter the report printing month, day and time.

Report Format Configuration

Report Format Configuration Dialog Box provides three text entry boxes for Report Comment, Report Header and Report Footer. An Excel-like entry table is supported for format detail entries. Users fill in each table field with text characters and specific symbols and keywords for required TAG point data.

Figure 7-9   Report Format Configuration dialog box

At the bottom of the dialog box, control menu buttons are provided for table edition and format saving:

Insert Col	Insert an empty column next to the marked field location
Insert Row	Insert an empty row next to the marked field location
Delete Col	Delete the marked column from the table
Delete Row	Delete the marked row from the table
Save As	Save the format with another file name
Cancel	Exit without save. If changes have been made, system will ask for confirmation.

Refer to the following table for the symbols and keyword accepted by the Report Generator.

Symbol	Character	Description	Example
&	H	Report Header character string	&HCentral Air Conditioner Daily Report
&	T	Report Footer character string	&TGENIE Standard Report
@	xxxx	Task Name	@TASK1
#	yyyyyyyy	TAG Name	#AI1
[ ]	nn	Channel Number for TAG block contains multiple channels (optional)	[01]
$	Now	Real Time value of specified TAG & channel	$NOW(@TASK1#AI1[01])
$	Hrzz	Value of specified TAG & channel at specified time hour	$HR01(@TASK1#AI1[01])
$	MAX	Maximum value of specified TAG & channel during the report period	$MAX(@TASK1#AI1[01])
$	MAXT	Time recorded for the max. value above	$MAXT(@TASK1#AI1[01])
$	MIN	Minimum value of specified TAG & channel during the report period	$MIN(@TASK1#AI1[01])
$	MINT	Time recorded for the min. value above	$MINT(@TASK1#AI1[01])
$	AVE	Average value of specified TAG & channel during the report period	$AVE(@TASK1#AI1[01])
:	aaaaa	Text to be displayed when value returned from the TAG point is equal to zero. Max. 20 characters. Used mainly for Digital Input Tag points	$NOW(@TASK1#DI1[01]:Open|Close)
|	bbbbb	Text to be displayed when value returned from the TAG point is not equal to zero. Max. 20 characters. Used mainly for Digital Input Tag points	$NOW(@TASK1#DI1[01]:Open|Close)
$	DATE	Current Date	$DATE
$	TIME	Current Time	$TIME
,		Column separate with vertical line	
;		Column separate without vertical line	
^		Row separate without upper line	
+		Addition Operator	
-		Minus Operator	
*		Multiply Operator	
/		Divide Operator	
.	n	number of decimal digits	$Now (@TASK1#AI1[01]).4

Notice

(a) $NOW command applies to all tags created by GeniDAQ. Please refer to Chapter 11.

(b) $HRnn, $MAX, $MAXT, $MIN, $MINT and $AVG commands only apply to the following tags:

- Analog Input (AI)
- Analog Output (AO)
- Digital Input (DI)
- Digital Output (DO)
- Temperature (TMP)
- Hardware Counter (CTFQ)
- Hardware Alarm (ALM)
- Time Stamp (TS)
- Counter (CNT)
- PID Control (PID)
- On/off Control (ONF)
- Ramp Block (RMP)
- Moving Average (AVG)
- Single Calculation (SOC)
- DDE Client (DDEC)
- VIRTASK's Tags (VIRTASK/User-defined Tag names)

7.3.3 Delete Report

To delete a specific report from the Report Designer, select the report from the respective report management dialog box and push the Delete button. The selected report will be removed from the report scheduler database. The report format file will be kept in the working directory for future use.
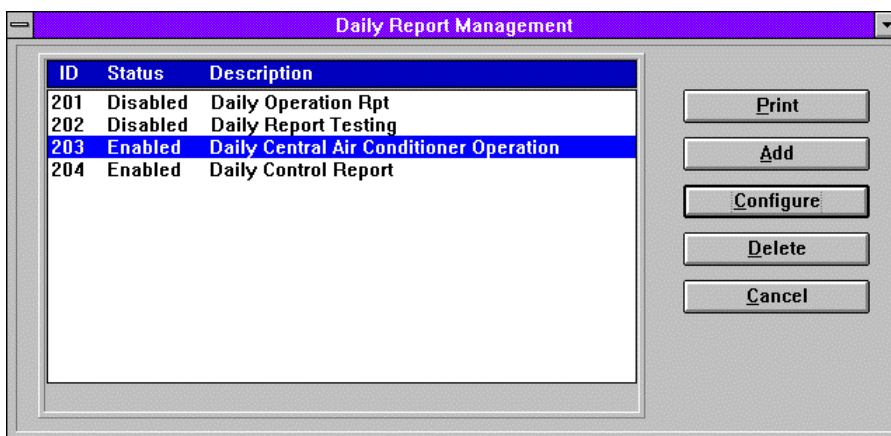
Figure 7-10   Delete existing Report

7.3.4 Print Report

All reports are printed automatically by the Report Designer according to the user-defined schedule. Report Designer also allows users to print previous reports as required. To print a specific report from the Report Designer, select the report from the respective report designer dialog box and push the Print button.

The Report Print Dialog is displayed below. Users can select previous reports from the drop down list box. The number of available reports is defined in the Report Designer Parameters Set Up Dialog Box. When a previous report is selected, users can decide to send it to printer or view the report using the Report Preview function.

The Report Preview button may be disabled if Report Designer can't find the respective database file or format file for the selected report. The button below Cancel is used to indicate whether a database file exists or not.
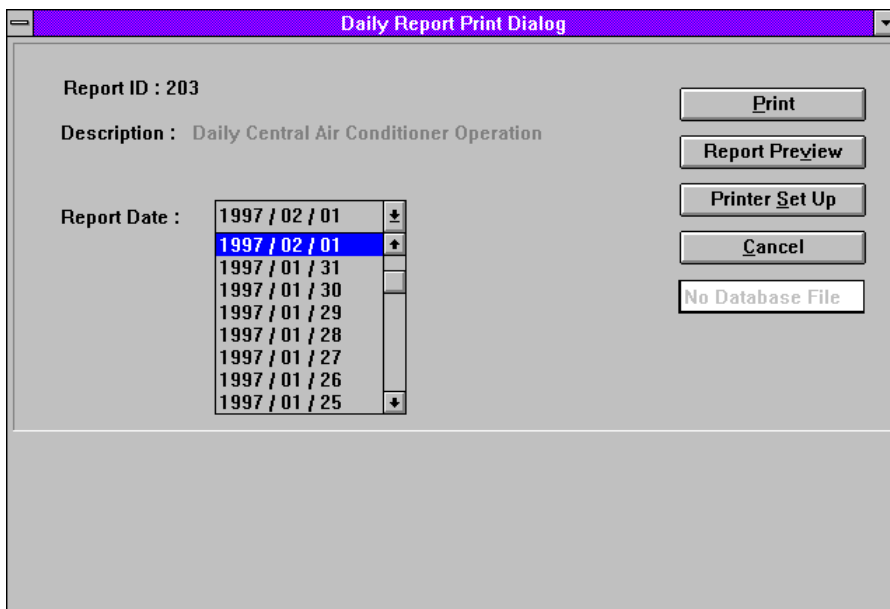
Figure 7-11   Print Report manually

Report Preview

Report Preview function allows users to check out the report before it is sent to a printer. Preview dialog is organized in a tableaus format. All calculated TAG point data are present in their final form. Users can scroll through the table to check the entire report. A Close button is provided for the user to return back to the main Print Dialog. The Cancel button is disabled in the preview dialog box.
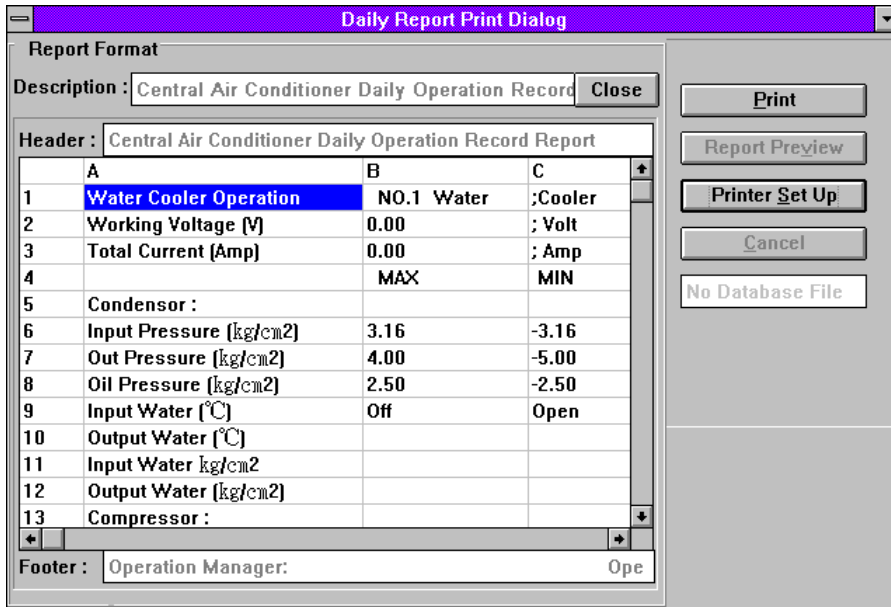
Figure 7-12   Preview printed report

Printer Setup

When users decide to modify any printer parameters or change the printer type, the Printer Set Up button should be selected in the Report Print Dialog. A standard Windows printer setup dialog box is displayed for printer setting modification.

8

GeniDAQ System Administration

8 GeniDAQ System Administration

In order to avoid unintentional or unauthorized operations which may affect system stability, GeniDAQ System Administration provides privilege control and password protection. Password protection offers built-in log-on with up to 255 levels of assignable access, providing extensive control for password access and conditional operations.

8.1 Administration

The administrator may add and delete users using the Lock feature in Runtime. To Add and delete users:
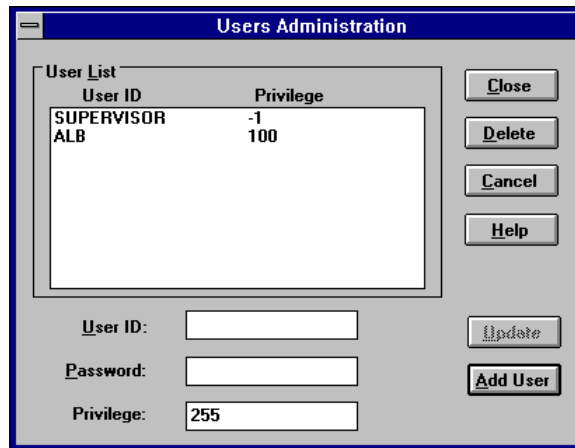
Figure 8-1   Users Administration dialog box

1) Enter the Supervisor Password previously entered using the Setup/Change Password dialog box. When GeniDAQ is started for the first time, the Supervisor Password is blank. Once a Supervisor Password is entered by using the Setup/Change Password dialog Box, the Supervisor Password stays in effect unless changed.

Note: If the Supervisor Password has been inadvertently forgotten, you may start fresh by deleting the SECURITY.PW file located in the GeniDAQ directory.

2) Enter each User ID and corresponding Privilege and Password by pressing Add User. The user ID and password can consist of as little as no characters, for which simply pressing the ESC key will unlock the strategy. The user ID and password can contain up to 16 characters or spaces, with no constraints on format.

3) Specify the privilege number of each User. The privilege is used to manage security of the system. You can design your system, display window and control object with privilege constraints to lock and protect user options. The privilege number is from 1 to 255. The larger number, the higher one's level of privilege to access the display window and control objects. The privilege number '-1' is reserved for the supervisor.

8.2 Change Password

Enter a password for the Lock feature in Runtime. The password can consist of as little as no characters, for which simply pressing the ESC key will unlock the strategy. The password can contain up to 16 characters or spaces, with no constraints on format. You can modify the user's password by keying in the user ID, the old password and a new password. The dialog box is as follows:
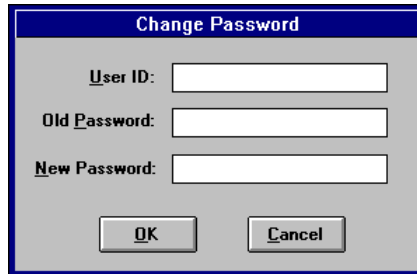
Figure 8-2   Change Password dialog box

8.3 Network I/O

8.3.1 Network Drivers Configuration
The Network I/O feature in GeniDAQ allows you to transfer data from any block over a Local Area Network (LAN) that supports Novell's IPX protocol (Novell NetWare is not required). Two blocks are used to support this feature; Network In and Network Out.

(For Windows 3.1)
To execute the network feature, the user must setup the network settings in the SYSTEM.INI file in the Windows directory as follows:

```
[boot]
network.drv=netware.drv
```
or
```
secondnet.drv=netware.drv

[386Enh]
network=vnetware.386, vipx.386
```
or
```
secondnet=vnetware.386, vipx.386
```

You must run the network driver (ipx.com or ipxodi.com) before running WINDOWS.

(For Windows 95/98)
The user needs to install three network components in the Network icon under Control Panel as follows:

```
[Configuration Tab]
Client for Netware Networks
NE2000 Compatible
IPX/SPX-compatible Protocal
```

Network Settings in GeniDAQ configuration file: \WINDOWS\GENIE.INI

```
[System]
NetPollTime=500
RetryToEnd=2000
TimeoutInSec=0
NetMsgLog=0
```

Entry	Description
NetPollTime	Specifies the network polling time in milliseconds. The default is 500 milliseconds. The minimum polling time is 100 milliseconds.
RetryToEnd	Specifies the retry count for acknowledging a remote station disconnecting a connection. The default is 2000 times. The minimum is 1000 times.
TimeoutInSec	Specifies the timeout of a connection in seconds. The minimum timeout is 30 seconds. The value 0 means never the connection never times out.
NetMsgLog	Enables logging communication message in NETMSG.LOG. The default is disabled (0), 1 is enabled.

GeniDAQ sets these default values for general applications. Users can change these values for their needs.

8.3.2 Network Settings

Before using NETIN and NTOUT blocks in task designer, you are advised to configure network settings for each workstation in your system. Click on the "Setup" menu and "Network..." submenu to invoke the Network Settings dialog box.
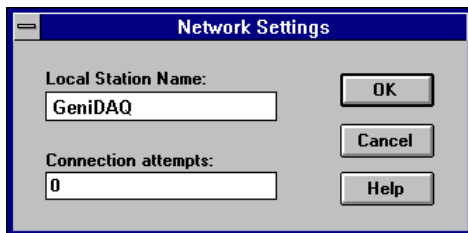
Figure 8-3   Network Settings dialog box

Local Station Name:
After making sure of your network settings as described above for each machine on your GeniDAQ Network, you must enter the Setup/Network Menu to name each Station. Each station on the network has to have a unique name. The result will be unpredictable if two stations on the network have the same name. Since the Station Name is stored in the strategy file (.GNI), each station should load and run a different strategy file.

Connection Attempts:

After connection, if a network block can not get any data from a remote station after it tries N times, then the network block will show timeout in the status bar. A zero (0) entry in this text box indicates that as many attempts as are necessary will be made for network connection. Any other integer in the text box will allow for connection attempts to stop after N number of tries.

Blocks are described in the following section.

8.3.3 Network Input Block

To use this block, your network must support the IPX Protocol. This block has output capability of up to eight channels. Up to eight (8) values are received from a corresponding Network Out block on another node (Remote Station) running GeniDAQ. The value(s) (string, float, or integer) sent from the Remote Station may in turn be routed to other strategy blocks or displays. This block can handle only one type of data for all channels used. If you need more than one data type, use additional Net-in/Net-out pairs. Each pair of Net-in and Net-out blocks must transfer the same data type. If they do not, the data will be incorrect on the receiving side. When this block is double clicked on, a dialog box will appear. Connection status will be shown in the status bar during Runtime.
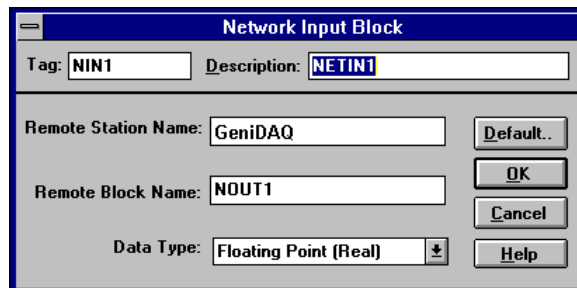
Figure 8-4   Network Input Block dialog box

Remote Station Name

Name of Remote Station from which data will be received.

Remote Block Name

Name of Remote Network Out block from which data will be received.

Data Type

Integer, Floating point (Real), or String Data types are supported.
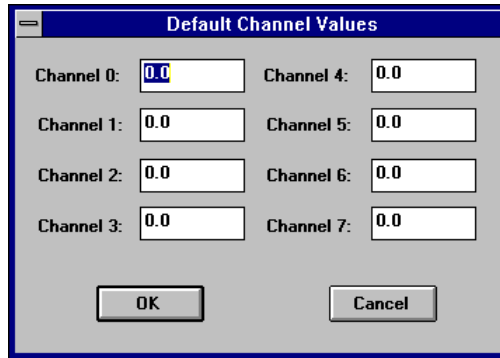
Figure 8-5   Default Channel Values dialog box

User-defined default values are set for the eight channels of the Network Input Block. This means that the Network Input Block will output the default value when it does not receive any data from a remote station. The user can use this value to determine if the data from Network Input Block is correct data.
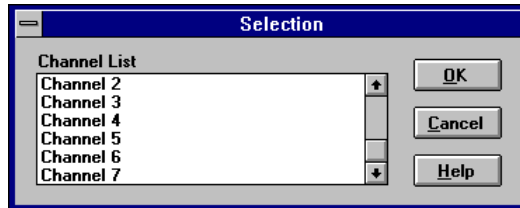
Figure 8-6   Network Input Block Selection dialog box

Choose the remote station's channel from which to send data to another block in your local strategy.

8.3.4 Network Output Block

NETOUT1

To use this block, your network must support the IPX Protocol. This block has input capability for up to eight channels. Up to eight (8) values are transmitted to a corresponding Network In block on another node (Remote Station) running GeniDAQ. The value(s) (string, float, or integer) sent from the Strategy blocks may be routed to a Remote Station.   This block can handle only one type of data for all channels used. If you need more than one data type, use additional Net-in/Net-out pairs. Each pair of Net-in and Net-out blocks must transfer the same data type. If they do not, the data will be incorrect on the receiving side. When this block is double-clicked upon, a dialog box will appear. Connection status will be shown in the status bar during Runtime.
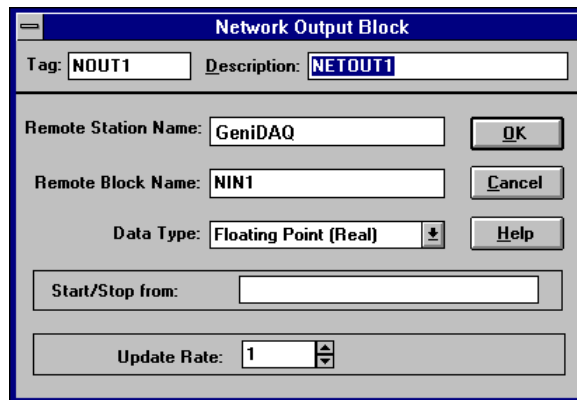
Figure 8-7   Network Output Block dialog box

Remote Station Name
Name of Remote Station to which data will be sent.

Remote Block Name
Name of Remote Network In block to which data will be sent.

Data Type
Integer, Floating point (Real), or String Data types are supported.

Start/Stop from
On/Off control for Network Output Block. Network Output Block has one input to control data to be sent. To start sending data, send a digital one to the block input from another block or display item, such as from a button. To stop sending data, send a digital zero to the block's "Start/Stop" input. If this input is not connected, data will be sent upon runtime initialization. Note that the receiving block (Network Input Block) will use the default value as it's output if no data is received.

Update Rate

The Update Rate is a divisor that allows the Network Output block to have a different effective scan rate than the rest of the task. This is useful if, for example, your task is running at 100 Hz, but you only want to send network data at a rate of 20 Hz. For this example, you would set the Update Rate to 5 (100 divided by 5 (the update rate) gives an effective scan rate of 20 Hz). In this scenario, you will still send 100 Hz data if you send the output to a Log File block or a display block, but only one in five samples will be "real". The other samples are merely copies of the "real" sample.

Valid values for the update rate are between 1 and 32767.

8.3.5 Network Performance and Capacity

- One station can connect to a maximum of fourteen stations, the maximum number of output stations is seven and the maximum number of input stations is seven. There is no limitation on the total number of stations in a network.

- One station can setup network input and network output blocks simultaneously. The maximum number of network blocks is 100; this allows a maximum of 800 points in the network.

- GeniDAQ has a dedicated timer for network communication. The polling time of the timer can be different from the scan time. The polling time is set in the configuration file \WINDOWS\GENIE.INI (the entry NetPollTime). The value of polling time depends on the number of stations and blocks. You can use the following equation to calculate the approximate polling time:

   Polling time = ((Stations * (Stations-1))/2) * base polling time

   Total time for transferring all blocks = (Network blocks/10) * polling time

   which base polling time depends on your PC's performance. Based on our tests, for three stations running on a Pentium 166 with 64M RAM, and forty blocks for each station, the polling time can be up to 500 milliseconds.

- The maximum string size for each channel of network block is 32 characters (including Null character).

- GeniDAQ Networking can work in Windows 95/98.

8.3.6 Communication event messages

Event Messages	Description
Connect to Station (Server): "Station name", "time", "date"	Setup a connection with a remote output station
Terminated by Station (Server): "Station name", "time", "date"	Terminate a connection with a remote output station
Timeout for Station (Server): "Station name", "time", "date"	A connection with a remote output station timed out
Connect to Station (Client): "Station name", "time", "date"	Setup a connection with a remote inputstation
Terminated by Station (Client): "Station name", "time", "date"	Terminate a connection with a remote input station
Timeout for Station (Client): "Station name", "time", "date"	A connection with a remote input station timed out

Note: You can enable the option "Enable Event Log" in "Runtime Preference.." under the Setup menu to display communication event messages.

8.3.7 Error messages and trouble shooting

Error Messages	Description
Network Initialize IPX failed	Might be caused by no network adapter
Network Bind Socket failed	Number of sockets is insufficient. You should increase the entry for Maximum sockets in the Network icon in Control Panel.
Network Initialize TLI failed	Might be caused by insufficient sockets.
Local station name cannot be the same as the remote station name	You should use different station name for each station.
Conflict data type in multiple NETIN blocks or no memory	The data types in multiple Network inputblocks with the same remote station name and remote block name are mis matched.
Remote station cannot receive termination messages	You should terminate all connected stationsto insure a successful connection in the future, or increase the entry for EntryToEnd in GENIE.INI to eliminate the messages.

Note: If the network settings are all correct, but the points still cannot connect to each other, you can change the Frame type in the Network icon in Control Panel from Auto-detecting to Ethernet 802.2, and try again.

9

GeniDAQ Runtime

9 GeniDAQ Runtime

GeniDAQ Runtime takes advantage of the Windows operating system to provide a real-time, multiple-tasking environment that combines the logical flow of a previously defined strategy with any number of Display Items in the Operator Panel. Runtime executes your strategy in real time, based upon data received from I/O devices or by manual operator entry. Through GeniDAQ Runtime, the process can be monitored and controlled, data can be logged to a disk, replayed, and manipulated by standard and/or user-defined functions. The GeniDAQ Runtime system allows simultaneous execution of control, graphics, alarming, trending, data logging, file transfer, and multiple I/O drivers. Its purpose is to execute strategies created in the GeniDAQ Task Designer.

9.1 Working with GeniDAQ Runtime

The GeniDAQ Runtime system behaves in different ways, depending on the choices made by the designer who created the strategy and corresponding displays. Its overall behavior is determined by the way Task Designer blocks have been arranged and connected, in conjunction with the Scan Period(s) set in the Setup/Tasks menu (Refer to the *Setup Menu* section in *Chapter 5.2: Task Designer Menu*).   If the Display has been designed with output capability (i.e., buttons, numeric displays that can be edited, etc.), the associated parameters can be changed during GeniDAQ Runtime. During GeniDAQ Runtime, commands and data can be entered using either the keyboard or the mouse.

9.1.1 Starting Runtime

Starting the Genie Runtime session is performed from Windows, DOS, or from the GeniDAQ Task Designer menu bar.

1) If Running from Windows, enter the Advantech GeniDAQ group, and double-click on the GeniDAQ Runtime icon. Open the desired strategy file with the suffix ".gni", and press START on the GeniDAQ Runtime Menu Bar.

2) If running from DOS, type:

```
win gwrun <strategy>
```

where *strategy* is the name of the strategy.gni file that was designed with the Task Designer. Make sure that the GENIDAQ\STRATEGY directory is in your path (autoexec.bat).

```
─                  Notepad - AUTOEXEC.BAT              ▼  ▲
File   Edit   Search   Help
@LH /L:0;1,30800 /S COMMAND /E:512 /CC:\NWCLIENT\START  ▲
LH /L:0;1,16400 /S C:\DOS\SMARTDRV.EXE /X /U
SET TEMP=C:\TEMP
MOUSE
@ECHO OFF
PROMPT $P$G
PATH C:\WIN;N:\APPL\CCMAIL;C:\CWIN;C:\;C:\DOS;C:\TOOLS
LH /L:1,27984 c:\mscdex.exe /d:mscd000
C:\DOS\SHARE.EXE
@ GENIE RUNTIME
WIN GWRUN SHOWALL.GNI|
                                                       ▼
◄                                                     ►
```
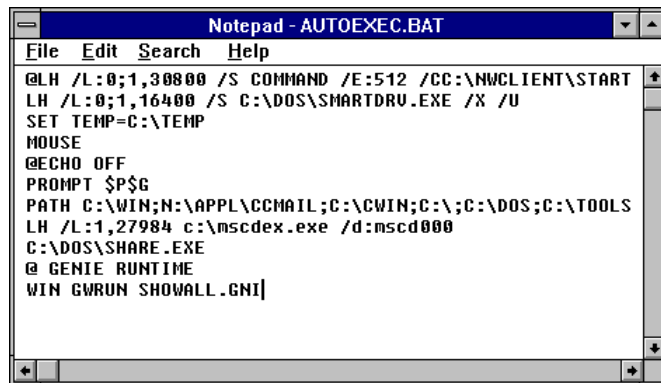
Figure 9-1   Run GeniDAQ at Start-up

3) You can also create a new icon with "GWRUN *<strategy>*" as the command line. This will cause the strategy to be loaded and started upon running.

4) If running from the Task Designer menu bar, after you have saved the currently displayed strategy, press "RUN/START" with your mouse.

9.1.2 Exiting Runtime

When you have finished working in Runtime, exiting back to Windows 95/98 is performed by pressing the STOP menu item on the Runtime menu bar. After you stop the runtime system, you can either restart by pressing start, or you can exit the GeniDAQ Runtime system by entering the File menu and pressing Exit.

9.2 Runtime Menu

The Runtime Menu Bar consists of a number of menus that enable you to manipulate files, start or stop running your strategy, or manipulate windows. There is also on-line help available through a help menu.

Figure 9-2   GeniDAQ Runtime main screen

Runtime Menus

> File Menu
> View Menu
> Start Menu
> Stop Menu
> Resume Menu
> Pause Menu
> Lock Menu
> Window Menu
> Help Menu

9.2.1 File Menu

The File menu includes commands that enable you to open and close existing Genie Strategy Files.
Commands include:

Figure 9-3   Runtime File menu options

Open
Use this command to Open an existing strategy. Normally, existing strategies will be stored in the
Genie directory.

Close
Use this command to Close your strategy. Make sure that you save it first.

HIST Conversion
Use this command to Convert the Historical Trending Display Item Log files from Binary to ASCII
format. These files have a naming convention as follows:

```
YYMMDD##.HST
```

where:

```
YY = Year file was created
MM = Month file was created
DD = Day file was created
## = Historical Trending Graph Number
```

after acquisition is complete in the GeniDAQ Runtime System. When converted to ASCII, you can
examine the contents using any standard ASCII editor. Please refer to chapter 6.2.1 File Menu.

Exit
Use this command to Exit the GeniDAQ Runtime System.

9.2.2 View Menu

The View menu includes the following options that allow you to adjust the display when running the system:

Toolbar

Enable/Disable to display the Toolbar on screen while the system is running. This option will be disabled once the system is running.

Status Bar

Enable/Disable to show the Status Bar at the bottom of the window. This option will be disabled once the system is running.

Event Log

Enable/Disable to show the Event Log pop-up panel. It is enabled when the system starts to run.

9.2.3 Start Menu

A command that enables you to start your strategy as soon as you press "start".

9.2.4 Stop Menu

A command that enables you to stop your strategy as soon as you press "stop".

9.2.5 Resume Menu

A command that enables you to Resume your strategy after previously pausing.

9.2.6 Pause Menu

A command that enables you to pause your strategy temporarily, to be continued at the same spot later.

9.2.7 Lock Menu

A command that enables you to "lock" your strategy from inputs from the keyboard or mouse. Refer to section 5.2, "Task Designer Menu Bar" under the section Setup/Runtime Preference for information on setting a password.

To "unlock" a "locked" strategy, press the ESC key, then type the password that was chosen in Setup/ Runtime Preference. If no password is selected in Setup/Runtime Preference, then simply pressing the ESC key will "unlock" the strategy.

9.2.8 Window Menu

The Window menu includes commands that enable you to arrange multiple windows within Genie. You can also select between a Runtime Window, Task Designer Window or a Display Window for a chosen file here.

9.3 Runtime Toolbar

To display or hide the Toolbar at the top of the Display Designer screen, highlight the View/Toolbar command with your mouse. The Toolbar and its associated buttons allow you to perform certain operations more quickly than actually going through the menus. Button commands are as follows:

Figure 9-4   Runtime Toolbar options

Open an Existing Strategy

This button opens an existing strategy file.

Close the Current Strategy

This button closes currently displayed strategy.

Printer Setup

This button displays the print setup dialog box.

Start Runtime

This button starts the strategy running.

Stop Runtime

This button stops the currently running strategy.

Resume Runtime

By pressing this button, runtime may be resumed after first pausing.

Pause Runtime

Pressing this button pauses runtime until the Resume button is pressed.

Context Sensitive Help

This button changes the mouse cursor to a special help cursor that allows you to select an object and display help for buttons, menus, and windows.

Runtime Status Bar

Ready	Error Count:0

To display or hide the Status Bar at the bottom of the Runtime screen, highlight the View/Status Bar command with your mouse. The Status Bar allows you to get quick, concise help on menu commands. As you highlight a menu item with your mouse, you can view information on the item in the status bar at the bottom of the runtime window. During runtime, status information is displayed such as any errors that have occurred, or whether the strategy is running normally. Errors generally include hardware driver errors or errors due to timing problems. The error codes are listed in the Appendix under Runtime Error Code Listing.

10

Script Designer

10 Script Designer

This chapter explains how to use Script Designer; a tool that enables you to edit and debug your BasicScript scripts. It begins with some general information about working with Script Designer and then discusses editing your scripts, running your scripts to make sure they work properly, debugging them if necessary, and exiting from Script Designer.

Contents
- Script Designer Basics
- Editing Your Scripts
- Running Your Scripts
- Debugging Your Scripts
- Programming with GeniDAQ

10.1 Script Designer Basics

This section provides general information that will help you work most effectively with Script Designer. It includes an overview of Script Designer's application window—the interface you'll use to edit, run, and debug your BasicScript scripts—as well as lists of keyboard shortcuts and information on using the Help system.

10.1.1 Toolbar

The following list briefly explains the purpose of each of the tools on Script Designer's toolbar. These tools are discussed in more detail later in the chapter, in the context of the procedures in which they are used.

Tool		Function
	Cut	Removes the selected text from the script and places it on the Clipboard.
	Copy	Copies the selected text, without removing it from the script, and places it on the Clipboard.
	Paste	Inserts the contents of the Clipboard at the current position of the insertion point.
	Undo	Reverses the effect of the proceeding editing change(s).
	Start	Begins execution of a script.
	Break	Suspends execution of an executing script and places the instruction pointer on the next line to be executed.
	End	Stops execution of a script.
	Toggle Breakpoint	Adds or removes a breakpoint on a line of BasicScript code.
	Add Watch	Displays the Add Watch dialog box, in which you can specify the name of a BasicScript variable. That variable, together with its value (if any), is then displayed in the watch pane of Script Designer's application window.
	Calls	Displays the list of procedures called by the currently executing BasicScript script. Available only during break mode.
	Single Step	Executes the next line of a BasicScript script and then suspends execution of the script. If the script calls another BasicScript procedure, execution will continue into each line of the called procedure.
	Procedure Step	Executes the next line of a BasicScript script and then suspends execution of the script. If the script calls another BasicScript procedure, BasicScript will run the called procedure in its entirety.

Note: *If you forget the name of a toolbar tool, pass the pointer over the tool to display its name.*

10.2 Editing Your Script

This section explains how to use Script Designer to edit BasicScript code. Although, in some respects, editing code with Script Designer is like editing regular text with a word-processing program, Script Designer also has certain capabilities specifically designed to help you edit BasicScript code.

You'll learn how to move around within your script, select and edit text, add comments to your script, break long BasicScript statements across multiple lines, search for and replace selected text, and perform a syntax check of your script.

This subsection provides an overview of the editing operations you can perform with Script Designer—including inserting, selecting, deleting, cutting, copying, and pasting material—and explains how to undo, or reverse, the most recent editing operations. You may wish to refer to the lists of keyboard shortcuts in the preceding section, which contain a group of editing shortcuts that can be used to perform many of the operations discussed here.

10.2.1 Navigating within a Script

The lists of keyboard shortcuts in the preceding section contain a group of navigating shortcuts, which you can use to move the insertion point around within your script. When you move the insertion point with a keyboard shortcut, Script Designer scrolls the new location of the insertion point into view if it is not already displayed. You can also reposition the insertion point with the mouse and the Goto Line command, as explained below.

Script Designer differs from most word-processing programs in that it allows you to place the insertion point anywhere within your script, including in "empty spaces." (Empty spaces are areas within the script that do not contain text, such as a tab's expanded space or the area beyond the last character on a line.)

Here's how to use the mouse to reposition the insertion point. This approach is especially fast if the area of the screen to which you want to move the insertion point is currently visible.

To move the insertion point with the mouse:

1. Use the scroll bars at the right and bottom of the display to scroll the target area of the script into view if it is not already visible.

2. Place the mouse pointer where you want to position the insertion point.

3. Click the left mouse button.

The insertion point is repositioned.

Note: *When you scroll the display with the mouse, the insertion point remains in its original position until you reposition it with a mouse click. If you attempt to perform an editing operation when the insertion point is not in view, Script Designer automatically scrolls the insertion point into view before performing the operation.*

Here's how to jump directly to a specified line in your script. This approach is especially fast if the area of the screen to which you want to move the insertion point is not currently visible but you know the number of the target line.

To move the insertion point to a specified line in your script:

1. Press F4.

   Script Designer displays the Goto Line dialog box.

2. Enter the number of the line in your script to which you want to move the insertion point.

3. Click the OK button or press Enter.

The insertion point is positioned at the start of the line you specified. If that line was not already displayed, Script Designer scrolls it into view.

Note: The insertion point cannot be moved so far below the end of a script as to scroll the script entirely off the display. When the last line of your script becomes the first line on your screen, the script will stop scrolling, and you will be unable to move the insertion point below the bottom of that screen.

10.2.2 Inserting Text

In Script Designer, inserting text and other characters such as tabs and line breaks works much the same as it does in a word-processing program: you position the insertion point at the desired location in the script and start typing. However, as noted in the preceding subsection, Script Designer lets you position the insertion point in "empty spaces". This means that you can also insert text into empty spaces—a feature that comes in handy when you want to insert a comment in the space beyond the end of a line in your script. Adding comments to your script is discussed later in this section. When you insert characters beyond the end of a line, the space between the insertion point and the last character on the line is backfilled with tab characters.

Another way in which Script Designer differs from word-processing programs is that in Script Designer, text does not wrap. If you keep entering text on a given line, eventually you will reach a point at which you can enter no more text on that line. Therefore, you control the line breaks by pressing Enter when you want to insert a new line in your script. The effect of pressing Enter depends on where the insertion point is located at the time:

• If you press Enter with the insertion point at or beyond the end of a line, a new line is inserted after the current line.

• If you press Enter with the insertion point at the start of a line, a new line is inserted before the current line.

• If you press Enter with the insertion point within a line, the current line is broken into two lines at that location.

If you press Tab, a tab character is inserted at the location of the insertion point, which causes text after the tab to be moved to the next tab position. If you insert new text within a tab's expanded space, the text that originally appeared on that line is moved to the next tab position each time the new text that you are entering reaches the start of another tab position.

When you enter certain types of text in Script Designer, the text automatically appears in a distinctive color. The default colors, which you can change, are as follows:

- When you type keywords, they appear in blue.

- When you type identifier text, it appears in black.

- When you type comments (beginning with either an apostrophe or "REM"), they appear in green.

10.2.3 Selecting Text

You can use either the mouse or the keyboard to select text and other characters in your script. Regardless of which method you use, you should be aware that in Script Designer, you can select either a portion of one line or a series of whole lines. You *cannot* select a portion of one line plus one or more whole lines. When you are selecting multiple lines and start or end your selection partway through a line, Script Designer automatically extends the selection to include the entire starting and ending lines.

To select text with the mouse:

1. Place the mouse pointer where you want your selection to begin.

2. While pressing the left mouse button, drag the mouse until you reach the end of your selection, and release the mouse button.

-Or-

While pressing Shift, place the mouse pointer where you want your selection to end and click the left mouse button. The selected text is highlighted on your display.

Another way to select one or more whole lines with the mouse is to start by placing the mouse pointer in the left margin beside the first line you want to select. The pointer becomes a reverse arrow, which points toward the line of text. Click the left mouse button to select a single line; press the left mouse button and drag up or down to select multiple lines.

Here's how to use keyboard shortcuts to select text in your script.

To select text with the keyboard:

1. Place the insertion point where you want your selection to begin.

2. While pressing Shift, use one of the navigating keyboard shortcuts to extend the selection to the desired ending point.

The selected text is highlighted on your display.

Note: *When you intend to select an entire single line of text in your script, it is important to remember to extend your selection far enough to include the hidden end-of-line character. This character inserts a new line in your script.*

Here's how to use the keyboard to select one or more whole lines in your script.

To select an entire line of text with the keyboard:

1. Place the insertion point at the beginning of the line you want to select.

2. Press Shift + Down arrow. The entire line, including the end-of-line character, is selected.

3. To extend your selection to include additional whole lines of text, repeat step 2.

Once you have selected text within your script, you can perform a variety of other editing operations on it, including deleting the text, placing it on the Clipboard by either cutting the text or copying it), and pasting it.

10.2.4 Deleting Text

When you delete material, it is removed from your script without being placed on the Clipboard. This section explains how to remove one or more characters, selected text or entire lines from your script.

To delete text:
To remove a single character to the left of the insertion point, press BkSp once; to remove a single character to the right of the insertion point, press Del once. To remove multiple characters, hold down BkSp or Del.

-Or-
To remove text that you have selected, press BkSp or Del.

-Or-
To remove an entire line, place the insertion point in that line and press Ctrl+Y.

Here's how to remove an unwanted line break from your script.

To combine the current line with the following line:

1. Place the insertion point after the last character on the current line.

2. Press Del once to delete the hidden end-of-line character.

The current line and the following line are combined.

Note: *If any spaces were entered at the end of the current line, you may have to press Del one or more additional times to remove these hidden characters first before you can delete the end-of-line character. Pressing BkSp with the insertion point at the start of a line has no effect—that is, it will not combine the current line with the preceding line.*

10.2.5 Cutting and Copying Text

You can place material from your script on the Clipboard by either cutting it or copying it.

To cut a selection:
• Press Ctrl+X.
The selection is removed from your script and placed on the Clipboard.

To copy a selection:
• Press Ctrl+C.
The selection remains in your script, and a copy of it is placed on the Clipboard.

10.2.6 Pasting Text

Once you have cut or copied material to the Clipboard, here's how to paste it into your script at another location.

To paste the contents of the Clipboard into your script:

1. Position the insertion point where you want to place the contents of the Clipboard.

2. Press Ctrl+V.

The contents of the Clipboard appear at the location of the insertion point. If you wish to delete a block of text and insert the contents of the Clipboard in its place, you can combine the two operations by first selecting the text you want to remove and then pressing Ctrl+V to replace it with the contents of the Clipboard.

Undoing Editing Operations
You can undo editing operations that produce a change in your script, including:

• The insertion of a series of characters

• The insertion of a block of text from the Clipboard

• The deletion of a series of characters

• The deletion or cutting of a block of text

You can't undo operations that don't produce any change in your script, such as moving the insertion point, selecting text, and copying material to the Clipboard.

To undo an editing operation:
• Press Ctrl+Z.
Your script is restored to the way it looked before you performed the editing operation.

10.2.7 Adding Comments to Your Script

You can add comments to your script to remind yourself or others of how your code works. Comments are ignored when your script is executed. In BasicScript, the apostrophe symbol (‘) is used to indicate that the text from the apostrophe to the end of the line is a comment.

Here's how to designate an entire line as a comment.

To add a full-line comment:

1. Type an apostrophe (‘) at the start of the line.

2. Type your comment following the apostrophe.

When your script is run, the presence of the apostrophe at the start of the line will cause the entire line to be ignored.

Here's how to designate the last part of a line as a comment.

To add a comment at the end of a line of code:

1. Position the insertion point in the empty space beyond the end of the code line.

2. Type an apostrophe (‘).

3. Type your comment following the apostrophe.
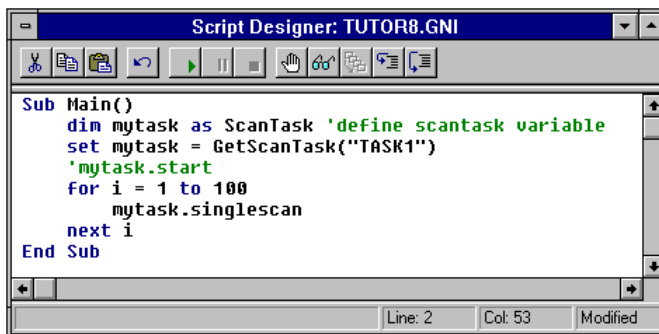
Figure 10-1   Add comments to script

When your script is run, the code on the first portion of the line will be executed, but the presence of the apostrophe at the start of the comment will cause the remainder of the line to be ignored. Although you can place a comment at the end of a line containing executable code, you cannot place executable code at the end of a line containing a comment. The presence of the apostrophe at the start of the comment would cause the balance of the line (including the code) to be ignored.

10.2.8 Breaking a BasicScript Statement across Multiple Lines

By default, in Script Designer, a single BasicScript statement can extend only as far as the right margin, and each line break represents a new statement. However, you can override this default if you want to break a long statement across two or more lines.

Here's how to indicate that two or more lines of BasicScript code should be treated as a single statement when your script is run.

To break a BasicScript statement across multiple lines:

1. Type the BasicScript statement on multiple lines, exactly the way you want it to appear.

2. Place the insertion point at the end of the first line in the series.

3. Press the spacebar once to insert a single space.

4. Type an underscore (_).

Note: *The underscore is the* line-continuation character, *which indicates that the BasicScript statement continues on the following line.*

5. Repeat steps 2–4 to place a line-continuation character at the end of each line (except the last) in the series.
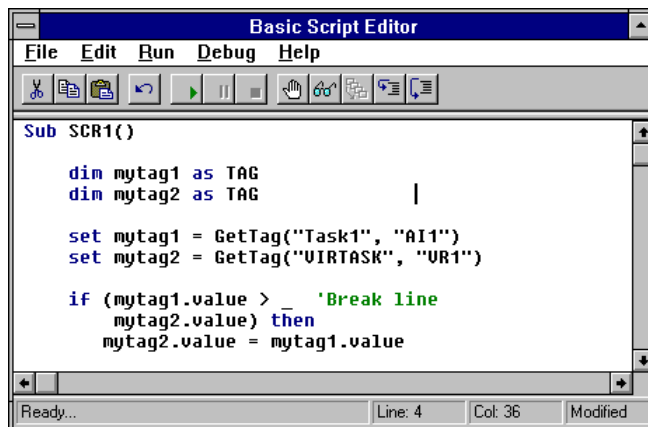
Figure 10-2   Break statement across multiple lines

When you run your script, the code on this series of lines will be executed as a single BasicScript statement, just as if you had typed the entire statement on the same line.

10.2.9 Checking the Syntax of a Script

When you try to run or debug a script whose syntax hasn't been checked, Script Designer first performs a syntax check automatically.

Here's how to perform a syntax check manually when you are editing your script, without having to run it.
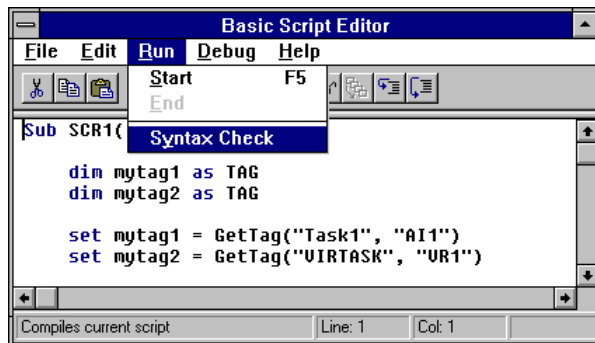
Figure 10-3   Check script's syntax

To perform a syntax check:

1. From the Run menu, choose the Syntax Check command.

    Script Designer either indicates that no errors have been found or displays an error message. This message specifies the first line in your script where an error has been found, and briefly describes the nature of the error.

2. Click the OK button or press Enter.

    If Script Designer has found a syntax error, the line containing the error is highlighted on your display.

3. Correct the syntax error.

4. Repeat steps 1–3 until you have found and corrected all syntax errors.

10.3 Running Your Scripts

Once you have finished editing your script, you will want to run it to make sure it performs the way you intended. You can also pause or stop an executing script.

Here's how to compile your script, if necessary, and then execute it.

To run your script:

• Click the Start tool on the toolbar.

-Or-
Press F5.

The script is compiled (if it has not already been compiled), the focus is switched to the parent window, and the script is executed.

Note: *During script execution, Script Designer's application window is available only in a limited manner. Some of the menu commands may be disabled, and the toolbar tools may be inoperative.*

To pause an executing script:

• Press Ctrl+Break.

Execution of the script is suspended, and the instruction pointer (a gray highlight) appears on the line of code where the script stopped executing.

Note: *The instruction pointer designates the line of code that will be executed next if you resume running your script.*

To stop an executing script:

• Click the End tool on the toolbar.

Note: *Many of the functions of Script Designer's application window may be unavailable while you are running a script. If you want to stop your script but find that the toolbar is currently inoperative, press Ctrl+Break to pause your script, then click the End tool.*

10.4   Debugging Your Scripts

This section presents some general information that will help you more effectively use Script Designer's debugging capabilities. The section also explains how to trace the execution of your script, how to set and remove breakpoints, and how to add watch variables and modify their value.

10.4.1 Using the BasicScript Debugger
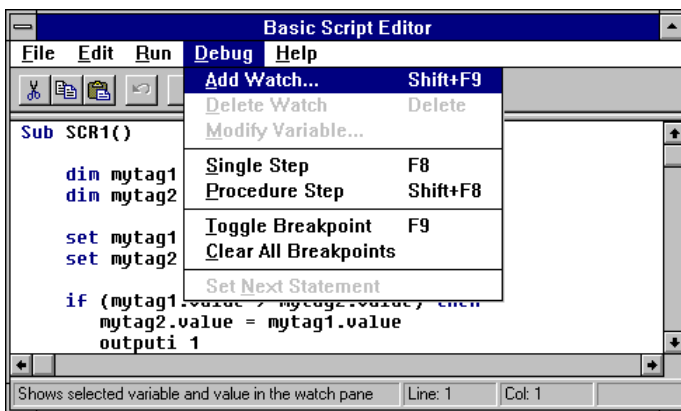
Figure 10-4   Using the BasicScript Debugger

While debugging, you are actually executing the code in your script line by line. Therefore, to prevent any modifications to your script while it is being run, the edit pane is read-only during the debugging process. You are free to move the insertion point throughout the script, select text and copy it to the Clipboard as necessary. You can also set breakpoints, and add and remove watch variables, but you cannot make any changes to the script until you stop running it.

To let you follow and control the debugging process, Script Designer displays an *instruction pointer* on the line of code that is about to be executed. This line will be executed next if you either proceed with the debugging process or run your script at full speed. When the instruction pointer is on a line of code, the text on that line appears in black on a gray background that spans the width of the entire line.

10.4.2 Tracing Script Execution

Script Designer gives you two ways to trace script execution—single step and procedure step—, both of which involve stepping through your script code line by line. The distinction between the two is that the single step process traces into calls to user-defined functions and subroutines, whereas the procedure step process does not trace into these calls (although it does execute them).
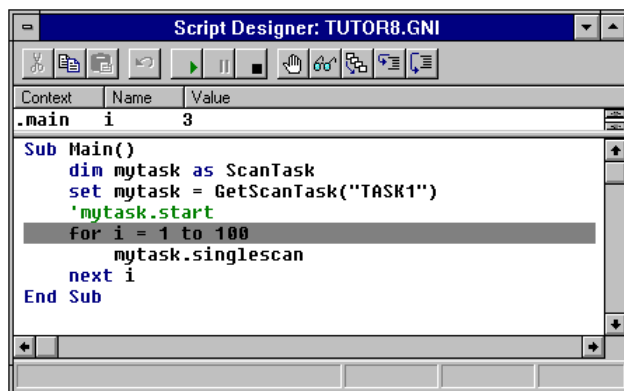
Figure 10-5   Trace Script Statement

Here's how to trace the execution of your script using the single step or procedure step method.

To step through your script:

1. Click the Single Step or Procedure Step tool on the toolbar.

-Or-
Press F8 (Single Step) or Shift+F8 (Procedure Step).

Script Designer places the instruction pointer on the Sub Main line of your script.

Note: *When you initiate execution of your script with any of these methods, the script will first be compiled, if necessary. Therefore, there may be a slight pause before execution actually begins. If your script contains any compile errors, it will not be executed. To debug your script, first correct any compile errors, then initiate execution again.*

2. To continue tracing the execution of your script line by line, repeat step 1.

   Each time you repeat step 1, Script Designer executes the line containing the instruction pointer and moves the instruction pointer to the next line to be executed.

3. When you finish tracing the execution of your script, click the Start tool on the toolbar (or press F5) to run the balance of the script. Click the End tool to halt execution of the script.

When you are stepping through a subroutine, you may need to determine the procedure calls by which you arrived at that point in your script.

Here's how to use the Calls dialog box to obtain this information.

To display the Calls dialog box:

1. Click the Calls tool on the toolbar.

   Script Designer displays the Calls dialog box, which lists the procedure calls made by your script in the course of arriving at the present subroutine.

2. From the Calls dialog box, select the name of the procedure you wish to view.

3. Click the Show button.

Script Designer highlights the currently executing line in the procedure you selected, scrolling that line into view if necessary (During this process, the instruction pointer remains in its original location in the subroutine.). When you are stepping through a subroutine, you may want to repeat or skip execution of a section of code.

Here's how to use the Set Next Statement command to move the instruction pointer to another line within that subroutine.

To move the instruction pointer to another line within a subroutine:

1. Place the insertion point in the line where you want to resume stepping through the script.

2. From the Debug menu, choose the Set Next Statement command.

The instruction pointer moves to the line you selected, and you can resume stepping through your script from there.

Note: *You can only use the Set Next Statement command to move the instruction pointer within the same subroutine. If you place the insertion point on a line that is not in the same subroutine, the Set Next Statement command will be disabled in the Debug menu.*

10.4.3 Setting and Removing Breakpoints

🤚

If you want to start the debugging process at the first line of your script and then step through your code line by line, the method described in the preceding subsection works fine. If you only need to debug one or more portions of a long script, that method can be cumbersome.
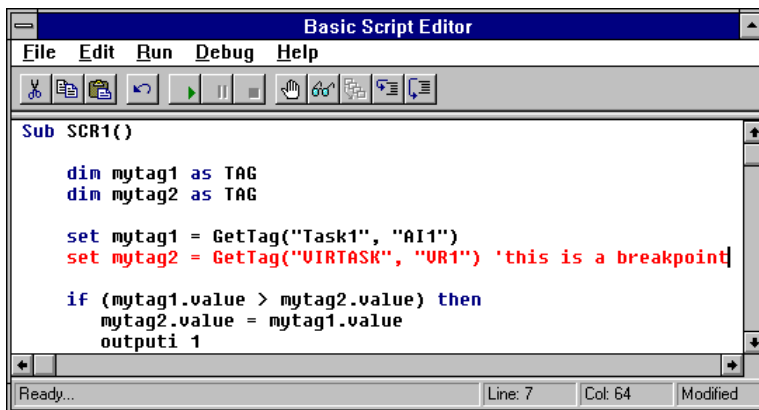
Figure 10-6   Set and remove breakpoints

An alternate strategy is to set one or more *breakpoints* at selected lines in your script. Script Designer suspends execution of your script just before it reaches a line containing a breakpoint, thereby allowing you to begin or resume stepping through the script from that point.

Setting Breakpoints
You can set breakpoints to begin the debugging process partway through your script, to continue debugging at a line outside the current subroutine, and to debug only selected portions of your script.

Valid breakpoints can only be set on lines in your script that contain code, including lines in functions and subroutines. Although you can set a breakpoint anywhere within a script prior to execution, when you compile and run the script, invalid breakpoints (that is, breakpoints on lines that don't contain code) are automatically removed. While you are debugging your script, Script Designer will beep if you try to set a breakpoint on a line that does not contain code.

Here's how to begin the debugging process at a selected point in your script.

To start debugging partway through a script:

1. Place the insertion point in the line where you want to start debugging.

2. To set a breakpoint on that line, click the Toggle Breakpoint tool on the toolbar.

-Or-
Press F9.

The line on which you set the breakpoint now appears in contrasting type.

3. Click the Start tool on the toolbar.

-Or-
Press F5.

Script Designer runs your script at full speed from the beginning and then pauses prior to executing the line containing the breakpoint. It places the instruction pointer on that line to designate it as the line that will be executed next when you proceed with debugging or resume running the script.

If you want to continue debugging at another line in the same subroutine, you can use the Set Next Statement command in the Debug menu to move the instruction pointer to the desired line.

If you want to continue debugging at a line that *isn't* within the same subroutine, here's how to move the instruction pointer to that line.

To continue debugging at a line outside the current subroutine:

1. Place the insertion point in the line where you want to continue debugging.

2. To set a breakpoint on that line, press F9.

3. To run your script, click the Start tool on the toolbar or press F5.

The script executes at full speed until it reaches the line containing the breakpoint and then pauses with the instruction pointer on that line. You can now resume stepping through your script from that point.

If you only need to debug parts of your script, here's how to facilitate the task by using breakpoints.

To debug selected portions of your script:

1. Place a breakpoint at the start of each portion of your script that you want to debug.

Note: *Up to 255 lines in your script can contain breakpoints.*

2. To run the script, click the Start tool on the toolbar or press F5.

   The script executes at full speed until it reaches the line containing the first breakpoint and then pauses with the instruction pointer on that line.

3. Step through as much of the code as you need to.

4. To resume running your script, click the Start tool on the toolbar or press F5.

   The script executes at full speed until it reaches the line containing the second breakpoint and then pauses with the instruction pointer on that line.

5. Repeat steps 3 and 4 until you have finished debugging the selected portions of your script.

Removing Breakpoints

Breakpoints can be removed either manually or automatically.

Here's how to delete breakpoints manually one at a time.

To remove a single breakpoint manually:

1. Place the insertion point on the line containing the breakpoint that you want to remove.

2. Click the Toggle Breakpoint tool on the toolbar.

-Or-
Press F9.

The breakpoint is removed, and the line no longer appears in contrasting type.

Here's how to delete all breakpoints manually in a single operation.

To remove all breakpoints manually:

• Select the Clear All Breakpoints command from the Debug menu.

Script Designer removes all breakpoints from your script.

Breakpoints are removed automatically under the following circumstances: (1) as mentioned earlier, when your script is compiled and executed, breakpoints are removed from lines that don't contain code. (2) When you exit from Script Designer, all breakpoints are cleared.

Adding a Watch Variable

As you debug your script, you can use Script Designer's watch pane to monitor selected variables. For each of the variables on this watch variable list, Script Designer displays its context, name, and value. The values of the variables on the watch list are updated each time you enter break mode.

Here's how to add a variable to Script Designer's watch variable list.

To add a watch variable:

1. Click the Add Watch tool on the toolbar.
-Or-
Press Shift+F9.

Script Designer displays the Add Watch dialog box.
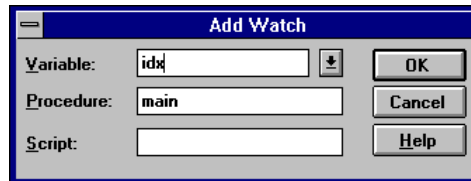
Figure 10-7   Add a watch variable

2. In the Variable box, type or select the name of the variable you want to add to the watch variable list.

   If you are executing the script, you can click the arrow in the Variable box to display the names of all variables that are "in scope," or defined within the current function or subroutine. You can then select the variable you want from the open list.

3. In the Procedure box, type or select the name of the procedure containing the variable you want to watch. If the variable you want to watch is a private or public variable, then type or select "(All Procedures)".

4. In the Script box, type or select the name of the script containing the variable you want to watch. If the variable you want to watch is a public variable, then type or select "(All Scripts)".

5. Click OK to add the variable to the watch variable list.

Script Designer displays the context, name, and value of the variable in a three-column list in the watch pane. If you have previously added other watch variables, Script Designer will display these as well.

Types of Variables You Can Watch

Script Designer permits you to monitor only variables of fundamental data types, such as Integer, Long, Variant, and so on; you cannot watch complex variables, such as structures or arrays, or expressions using arithmetic operators. You can, however, watch individual elements of arrays or structure members using the following syntax:

[*variable* [(*index*,...)] [.*member* [(*index*,...)]]...]

where *variable* is the name of the structure or array variable, *index* is a literal number, and *member* is the name of a structure member.

For example, the following are valid watch expressions:

Watch Variable	Description
a(1)	Element 1 of array a
person.age	Member age of structure person
company(10,23).person.age	Member age of structure person that is at element 10,23 within the array of structures called company

10.4.4 Modifying or Deleting a Watch Variable

In order to modify the value of a variable or remove the variable from the watch pane, you must first select the desired variable. Here's how to select a variable on the list.

To select a watch variable:

• Place the mouse pointer on the variable you want to select and click the left mouse button.

-Or-
If one of the variables on the watch list is already selected, use the arrow keys to move the selection highlight to the desired variable.

-Or-
If the insertion point is in the edit pane, press F6 to highlight the most recently selected variable on the watch list and then use the arrow keys to move the selection highlight to the desired variable.

Note: Pressing F6 again returns the insertion point to its previous position in the edit pane.

When the debugger has control, you can modify the value of any of the variables on Script Designer's watch variable list. Here's how to change the value of a selected watch variable.

To modify the value of a variable on the watch variable list:

1. Place the mouse pointer on the name of the variable whose value you want to modify and double-click the left mouse button.

-Or-
Select the name of the variable whose value you want to modify and press Enter or F2.
Script Designer displays the Modify Variable dialog box.

Note: The name of the variable you selected on the watch variable list appears in the Name field. If you want to change another variable, you can enter a different variable in the Name field. You can also select a different variable from the Variables list box, which shows variables defined within the current function or subroutine. When you use the Modify Variable dialog box to change the value of a variable, you don't have to specify the context. Script Designer first searches locally for the definition of that variable, then privately, then publicly.

2. Enter the new value for your variable in the Value field.

3. Click the OK button.

The new value of your variable appears on the watch variable list.

When changing the value of a variable, Script Designer may convert the value you entered to match the type of the variable. For example, if you change the value of an Integer variable to 1.7, Script Designer converts this value from a floating-point number to an Integer, assigning the value 2 to the variable.

When modifying a Variant variable, Script Designer needs to determine both the type and value of the data. Script Designer uses the following logic in performing this assignment (in this order):

If the new value is	*Then*
Null	The Variant variable is assigned Null (VarType 1).
Empty	The Variant variable is assigned Empty (VarType 0).
True	The Variant variable is assigned True (VarType 11).
False	The Variant variable is assigned False (VarType 11).
number	The Variant variable is assigned the value of *number*. The type of the variant is the smallest data type that fully represents that number. You can force the data type of the variable by using a type-declaration letter following *number*, such as %, #, &, !, or @.
date	The Variant variable is assigned the value of the new date (VarType 7).
Anything else	The Variant variable is assigned a String (VarType 8).

Script Designer will not assign a new value if it cannot be converted to the same type as the specified variable.

Here's how to delete a selected variable from Script Designer's watch variable list.

To delete a watch variable:

1. Select the variable on the watch list.

2. Press Del.

The selected variable is removed from the watch list.

10.4.5 Exiting from Script Designer

Here's how to get out of Script Designer. What happens when you exit depends on (1) whether you have made changes to your script and (2) whether your script contains errors.

To exit from Script Designer:

• Choose the Exit and Return command from the File menu.

If you *have* made changes to your script, Script Designer displays a dialog box asking whether you want to save the script. If you either click the No button or click the Yes button and your script contains no errors, you exit from Script Designer immediately. If you click the Yes button and your script contains errors, Script Designer highlights the line containing the first error and displays a dialog box asking whether you want to exit anyway. If you click the Yes button, Script Designer saves your script, errors and all, and then you exit from Script Designer.

If you *haven't* made any changes to your script, you exit from Script Designer immediately, regardless of whether the script contains errors from a previous editing session.
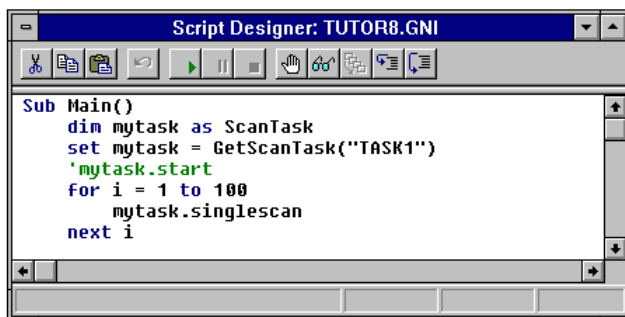
10.5 Programming with GeniDAQ

This powerful Visual Basic for Application (VBA) compatible scripting engine is licensed from Summit Software Inc. BasicScript is the most important component providing programmability in GeniDAQ. Since Visual Basic is very popular and easy to learn, the change to BasicScript should increase GeniDAQ's programmability tremendously. BasicScript replaces Version 2.x C-like scripting. While this may cause some compatibility problems between 2.x and 3.0, it is a necessary step to further enhance GeniDAQ.

 The script engine is a set of DLLs that facilitate the compilation of script programs at build-time and execution of scripts at runtime. It not only provides the capability to manipulate tasks, but also provides the interface to interact with DOS, Windows, and other applications through DDE, OLE, ODBC (SQL). The syntax of BasicScript is compatible with Microsoft VBA (visual basic for application in Excel, Word, Access, etc.) and MS Visual Basic. Both BasicScript and VBA have functions that are not included in the other, but over 95% of the functions and procedures are identical. It is possible to take a Visual Basic source code, then compile and execute it under BasicScript. This can be done without changing a word if only common functions are used. Error number and error message in BasicScript is also compatible with Visual Basic. A dialog box editor is included so that program designers can create their own dialog box to interact with the operators. The scripting feature is what makes GeniDAQ3.x suitable for real-life situations where customization is necessary to make the program work smoothly with the existing operating procedures in the factory and assembly line.

The script designer is basically a text editor with some convenient features for editing script code. The same editor is used for editing the main script and scripts inside a task. The main script, if present, takes control during the entire runtime once it is started. The main script can be used to do things such as starting a task, stopping a task, etc. Each scan task has a pre-task script and a post-task script. These two scripts are used to initialize or reset tag values on some conditions. The main script is executed only once for the entire strategy while the pre-task and post-task scripts are executed once every scan. The script will be compiled into p-code after editing so it won't need to be compiled again.

Main Script

Figure 10-8   Main Script program

The main script, if present, takes control of the entire runtime once it is started. The main script can be used to do things such as starting a task, stopping a task, etc.

Pre-Task and Post-Task script
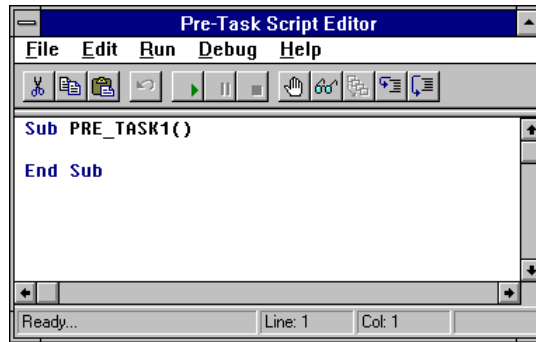
Figure 10-9   Pre-task Script Editor
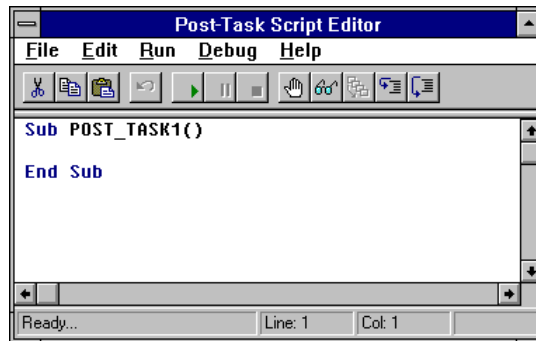
Figure 10-10   Post-task Script Editor

Each scan task has a pre-task script and a post-task script. User can use these two scripts to initialize or reset tag value on some conditions.

BasicScript
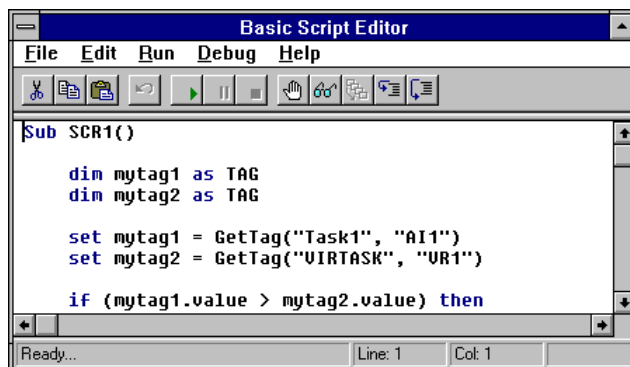
Figure 10-11   BasicScript Block Editor

BasicScript is a task block in Task Designer Toolbox. It is designed to provide total flexibility so you can employ Visual Basic functions to do comparisons, calculations, etc. BasicScript supports multiple inputs from all types of blocks. Its output can be routed to any number of blocks. For detailed information, please refer to *chapter 5.3.27 BasicScript block.*

Note: *"DIM" may be used to declare one variable per line. If you declare two or more variables in one line, the execution result will be unpredictable.*

BasicScript GeniDAQ commands

In order to manipulate tasks and runtime I/O data in the BasicScript engine, GeniDAQ adds many functions into BasicScript and can be programmed using standard Visual Basic functions. The Added GeniDAQ functions are categorized in the following sets:

Scan Task
This set of function calls is used to manipulate scan tasks in GeniDAQ. You can use these function calls in the Main Script program or in the Pre-Task and Post-Task script programs.

- *ScanTask*   Define a ScanTask object
- *GetScanTask*   Return the ScanTask object specified
- *Start*   Start to run the ScanTask
- *Stop*   Stop the ScanTask
- *SingleScan*   Do a one time scan of the ScanTask
- *GetStatus*   Return the status of specified ScanTask

Tag
This set of function calls is used to access task blocks in the GeniDAQ data center. You can use these function calls in the Main Script program, Pre-Task and Post-Task script programs and/or BasicScript blocks.

- *Tag*   Define a Tag Object
- *GetTag*   Return the Tag object specified
- *Value*   Property of the tag, return current value of tag
- *Array*   Property of the tag, return current value of tag's channel data.

Display
This set of function calls is used to manipulate multiple displays in GeniDAQ. You can use these function calls in the Main Script program or in the Pre-Task and Post-Task script programs.

- *Display*   Switch to the display specified

BasicScript Block

This set of function calls is used to manipulate the output of BasicScript blocks in GeniDAQ. These function calls are used in BasicScript blocks only.

- ***Outputi***   Output integer to another block
- ***Outputl***   Output long integer to another block
- ***Outputf***   Output single float to another block
- ***Outputs***   Output string to another block

10.5.1 ScanTask

ScanTask (object type)

Syntax
```
ScanTask
```

Description

An object type used to declare a variable which accepts object returned from GetScanTask() function*.*

Comments

When declaring more than one variable on the same line, the syntax is:
dim MyTask1 as ScanTask, MyTask2 as ScanTask.

Example
```
'This example start the task MyTask
Sub Main()
   dim MyTask as ScanTask
   set MyTask = GetScanTask("TASK1")
   MyTask.Start
End Sub
```

See Also

GetScanTask(Function), GetStatus(method), Start(method), Stop(method), SingleScan(method)

ScanTaskObject.**GetStatus (method)**

Syntax

> *MyTask*.GetStatus

Description

A method of the ScanTask object used to get the status of specified task.

Return

An integer value will be returned to represent the current status of specified task.

0 means the specified task is idle, that is task is not started or is stopped.

2 means the specified task is waiting for delayed time or system time set in task properties.

3 means the specified task is running for GeniDAQ scan.

Comments

Applies to ScanTask object. It is suggested that this method be used before start or stop task.

Example

```
'This example scan the task MyTask for 100 times
Sub Main()
   dim i as integer
   dim Mystatus as integer
   dim MyTask as ScanTask
   set MyTask = GetScanTask("TASK1")
   MyStatus = MyTask.GetStatus
   if (MyStatus == 0)
   then
        MyTask.Start
       MyTask.Stop
   end if
End Sub
```

See Also

GetScanTask(Function), GetStatus(method), Initialize(method), Start(method), Stop(method)

ScanTaskObject.**SingleScan (method)**

Syntax

    MyTask.SingleScan

Description

    A method the ScanTask object uses to do a single scan of a task.

Comments

    Applies to ScanTask object. The ScanTask is scanned immediately regardless of the scan period specified in the ScanTask Setup dialog box. This call does not return until the scan is completed.

Example

```
'This example scan the task MyTask for 100 times
Sub Main()
   dim i as integer
   dim MyTask as ScanTask
   set MyTask = GetScanTask("TASK1")
   for i = 1 to 100
      MyTask.SingleScan
   next i
End Sub
```

See Also

    GetScanTask(Function), GetStatus(method), Start(method), Stop(method)

ScanTaskObject.**Start (method)**

Syntax

> *MyTask*.Start

Description

A method of the ScanTask object used to start a task.

Comments

Applies to ScanTask object. The ScanTask is started regardless of the starting method specified in the ScanTask Setup dialog box. This call returns immediately without waiting for the task to complete.

Example

```
'This example start the task MyTask
Sub Main()
   dim MyTask as ScanTask
   set MyTask = GetScanTask("TASK1")
   MyTask.Start
End Sub
```

See Also

GetScanTask(Function), GetStatus(method), Stop(method), SingleScan(method)

ScanTaskObject.**Stop (method)**

Syntax

> *MyTask*.Stop

Description

A method of the ScanTask object used to stop a task.

Comments

Applies to ScanTask object.

Example

```
'This example stop the task MyTask
'Assume MyTask is running already
Sub Main()
   dim MyTask as ScanTask
   dim FileLength as Integer
   set MyTask = GetScanTask("TASK1")
   Open "test.dat" For Input As #1
   FileLength = Lof(1)
   If (FileLength > 32000) Then
       MyTask.Stop
   End If
End Sub
```

See Also

GetScanTask(Function), GetStatus(method), Start(method), SingleScan(method)

GetScanTask (function)

Syntax

```
Set ScanTaskObj = GetScanTask(taskname)
```

Description

Returns an object of type ScanTask if successful. Returns *Null* if the taskname specified can not be found or not valid.

Comments

`ScanTaskObj` is a variable declared as type ScanTask. `taskname` is a string expression specifying the task. The taskname should be in capital letters, for example, "TASK1".

Example

```
'This example start the task MyTask
Sub Main()
 dim MyTask as ScanTask
 set MyTask = GetScanTask("TASK1")
 MyTask,Initialize
 MyTask.Start
End Sub
```

See Also

**ScanTask(object type), Start(method), Stop(method),
SingleScan(method),GetStatus(method),**

10.5.2 Tag

Tag (object type)

Syntax
```
Tag
```

Description
An object type used to declare a variable which accepts objects returned from GetTag() function**.**

Comments
When declaring more than one variable on the same line, the syntax is:
dim MyTag1 as Tag, MyTag2 as Tag.

Example
```
'This example obtain a value from a block
Sub Main()
   dim MyTag as Tag
   dim voltage as single
   set MyTag = GetTag("TASK2", "AI1")
   voltage =MyTag.Value
End Sub
```

See Also
GetTag (function); *tagObj*.**Value** (property).

tagObj.**Value (Property)**

Syntax
```
tagObj.Value
```

Description
Value is a property of Tag object. It is used to reference the value of a input tag.

Example
```
'This example obtain a value from a block
Sub Main()
   dim MyTag as Tag
   dim voltage as single
   set MyTag = GetTag("TASK2", "AI1")
   voltage =MyTag.Value
End Sub
```

See Also
GetTag (function); **Tag** (object type); **Array**(channel number).

tagObj.**Array(Channel number)**

Syntax
```
tagObj.array(channel number)
Channel number: integer, from 0 to 15
```

Description
Array is a property of Tag object. It is used to reference the values of tag's multiple channels.

Example
```
'This example obtain a value from the first channel of AI block
Sub Main()
    dim MyTag as Tag
    dim voltage as single
    set MyTag = GetTag("TASK2", "AI1")
    voltage =MyTag.Array(0)
End Sub
```

See Also
GetTag (function); **Tag** (object type); **Value**(Property).

GetTag (function)

Syntax
```
Set tagObj = GetTag(tableName$, tagName$)
```

Description
Returns an object of type Tag.

Comments
The **GetTag** function takes the following parameters:

Parameter	Description
tableName$	String expression containing the name of the task or display or virtual.
tagName$	String expression containing the name of the tag (block). Should be capitals.

Example
```
Sub Main()
 dim MyTag as Tag
 dim voltage as single
 set MyTag = GetTag("TASK2", "AI1")
 voltage = MyTag.Value
End Sub
```

See Also
Tag (object type); *tagObj*.**Value** (property)

10.5.3 Display

Display (statement)
Syntax
```
Display dvalue
```

Description
Used in the Main Script program or Pre-Task and Post-Task script to control the display order of multiple displays in a system.

Comments
dvalue is the number of display windows. For example, use Display(1) to show display1 window on screen.

Parameter	Description
dvalue	An integer specifying the number of display windows to show on screen.

Example
```
Sub Main()
   dim i as integer
   dim Mystatus as integer
   dim MyTask as ScanTask
   set MyTask = GetScanTask("TASK1")
   MyStatus = MyTask.GetStatus
   if (MyStatus == 3)
   then Display(1) " to show Display1
   end if
End Sub
```

10.5.4 BasicScript Block

Outputf (statement)

Syntax
```
Outputf fvalue
Outputf channel , fvalue
```

Description
Used in the Basic Script Block to output a single float value to another block.

Comments
If the channel argument is not given, channel 0 is assumed.

The **Outputf** statement takes the following parameters:

Parameter	Description
channel	Integer containing the output channel. Valid channel numbers are from 0 to 7.
fvalue	A single float specifying the value to be output.

Example
```
'This example outputs a number to channel 0
dim voltage as single
If Voltage > 0.0 Then
    Outputf  voltage
else
    Outputf  - Voltage
End If
```

See Also
Outputi (statement); **Outputl** (statement); **Outputs** (statement).

Outputi (statement)

Syntax
```
Outputi ivalue
Outputi channel , ivalue
```

Description

Used in the Basic Script Block to output an integer value to another block.

Comments

If the channel argument is not given, channel 0 is assumed.

The **Outputi** statement takes the following parameters:

Parameter	Description
channel	Integer containing the output channel. Valid channel numbers are from 0 to 7.
ivalue	Integer specifying the value to be output.

Example
```
'This example output a number to channel 0
dim count as integer
If count > 0 Then
 Outputi count
Else
 Outputi  -1
End If
```

See Also

Outputl (statement); **Outputf** (statement); **Outputs** (statement).

Outputl (statement)

Syntax
```
Outputl longvalue
Outputl channel , longvalue
```

Description

Used in the Basic Script Block to output a long integer value to another block.

Comments

If the channel argument is not given, channel 0 is assumed.

The **Outputl** statement takes the following parameters:

Parameter	Description
channel	Integer containing the output channel. Valid channel numbers are from 0 to 7.
longvalue	long Integer specifying the value to be output.

Example
```
'This example output a number to two channels
dim count as long
If count >= 0 Then
 Outputl 0, count
 Outputl 1, count
End If
```

See Also

Outputi (statement); **Outputf** (statement); **Outputs** (statement).

Outputs (statement)

Syntax
```
Outputs svalue$
Outputs channel , svalue$
```

Description

Used in the Basic Script Block to output a string to another block.

Comments

If the channel argument is not given, channel 0 is assumed.

The **Outputs** statement takes the following parameters:

Parameter	Description
channel	Integer containing the output channel. Valid channel numbers are from 0 to 7.
svalue$	string expression specifying the value to be output.

Example
```
'This example output a string
dim S1 as string
dim S2 as string
S1 = "Hello,"
S2 = "World!"
Outputs 3, S1 & S2
```

See Also

Outputi (statement); **Outputl** (statement); **Outputf** (statement).

10.5.5 System

Quit (statement)

Syntax
```
Quit dvalue
```

Description

Terminate runtime system.

Comments

dvalue is reserved. It must be 0.

Example
```
Sub Main()
    dim i as integer
    dim Mystatus as integer
    dim MyTask as ScanTask
    set MyTask = GetScanTask("TASK1")
    MyStatus = MyTask.GetStatus
    if (MyStatus == 3)
    then Quit(0) " terminate runtime system
    end if
End Sub
```

Notice

(a) In the debug environment for the Main Script, Pre/post-task Script or Basic Script block, the following commands are disabled:

- ScanTask.start

- ScanTask.stop

- ScanTask.SingleScan

- ScanTask.GetStatus

- OutputI

- OutputL

- OutputF

- OutputS

- Display

- Quit

However, all commands will function at runtime.

(b) When using the Msg command to create modeless dialogs or the Sleep command for waiting, do not stop GeniDAQ when the modeless dialog is still active or the Sleep command is still working.

(c) Be careful when programming the script to avoid an infinite loop.

Limitations

(a) Strings are limited in length to 32764 characters.

(b) The data area that holds public variables is limited to 16 KB.

(c) The size of source code script is limited to 65534 characters.

(d) Arrays can have up to 60 dimensions.

(e) Variable names are limited to 80 characters.

(f) Labels are limited to 80 characters.

(g) The number of open DDE channels is not fixed; rather, it is limited only by available memory and system resources.

(h) The number of open files is limited to 512 or the operating system limit.

(i) The size of an array cannot exceed 32 KB.

11

Data Center

11 Data Center

Data center is a set of DLLs that work as a holding place for all data in GeniDAQ Runtime. Data center supports three different sets of interface to the outside world: C API, DDE, and OLE automation. C API is the most efficient interface, so it is used for all communication between components of GeniDAQ. OLE automation is designed to interface with OLE-aware applications.

This module is the central location for all the data in GeniDAQ. All the results of the blocks or the user data entered on the screen are passed to this centralized module. It is stored in memory for fast update and retrieval. The tag name is used as a key to find the data item in the data center. A hash table or B-tree structure can be used to speed up the update and retrieval. Each data item in the data center has a tag name, block ID, update timestamp, read count, and a value associate with it. Other applications, like drivers and GeniDAQ, would use the tag name to get a block ID from the data center. The block ID is another key to find the data item in the data center.

11.1 GeniDAQ Architecture
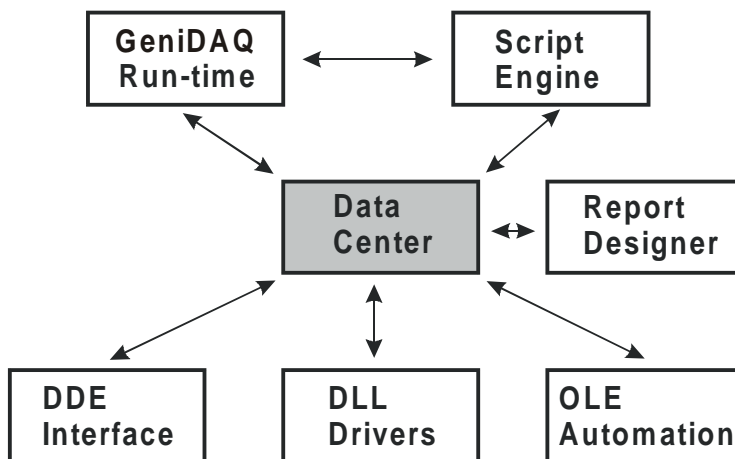
Figure 11-1   GeniDAQ Architecture

The new architecture is what really make GeniDAQ 3.x more flexible and powerful. Each component is explained below:

GeniDAQ Runtime
The runtime is similar to Version 2.0 except it allows switching between different tasks and running of a single scan of a task. This is how GeniDAQ 3.x is able to execute multiple tasks simultaneously. Because one scan of a task is the smallest execution unit, users can design their strategy so that related blocks are grouped together in the same task. By using the main script to control the execution of tasks, you can invoke a task when certain conditions occur. This enables design of complicated system by combining simple tasks. A simple strategy with only one task can now be executed without script.

Script Engine

The script engine is a set of DLLs that facilitates the compilation of script programs at build-time and execution of scripts at runtime. It not only provides the capability to manipulate tasks but also provides the interface to interact with DOS, Windows, and others applications through DDE, OLE, ODBC (SQL). The syntax of the BasicScript is compatible with Microsoft VBA (visual basic for application in Excel, Word, Access, etc.) and MS Visual Basic. Both BasicScript and VBA have their own functions that are not covered by the other, but over 95% of the functions and procedures are identical. It is possible to take a Visual Basic source code, then compile and execute it under BasicScript without changing a word if only common functions are used. Error number and error message in BasicScript is also compatible with Visual Basic. A dialog box editor is included so that designer of the strategy can create their own dialog box to interact with the operators. The scripting feature is what makes GeniDAQ 3.x suitable for real-life situations where customization is necessary to make the program work smoothly with the existing operating procedures in the factory and assembly line.

DLL Drivers

GeniDAQ 3.x, like previous versions of GeniDAQ, uses DLL drivers. The drivers use the C interface to update data or retrieve data from GeniDAQ runtime.

Device drivers can write to the data center or read from the data center directly via C function calls. GeniDAQ and other application programs can also update and retrieve data the same way. The Basic-Script should have full access to the data center as well. Since the DDE interface is slower than the C function call, it is used as a supplement to the C interface. The DDE interface is important for exchanging data with other software packages such as Excel, ADAM DDE Server, etc.

For I/O blocks, those with the same tag name can not exist in the same task, but they are allowed in different tasks. I/O blocks don't belong to any task. The data center keeps a global variable to store all I/O blocks.

Report Designer

The Report Designer is a separate application program completely independent from GeniDAQ runtime. Report designer allows users to print data to the screen or to a printer.

The Tags Created by GeniDAQ are the Following:

Block	Tag name	Data type	Channel count
Analog Input	TASK1/AI1	Floating point	32
Analog Output	TASK1/AO1	Floating point	1
Digital Input	TASK1/DI1	Integer	1
Digital Output	TASK1/DO1	Long	1
Temperature	TASK1/TMP1	Floating point	1
Hardware Counter	TASK1/CTFQ1	Floating point	1
Hardware Alarm	TASK1/ALM1	Long	1
RS-232	TASK1/SER1	String	8
Input File	TASK1/INF1	Floating point	1
Alarm Log	TASK1/ALOG1	Long	1
Ramp	TASK1/RMP1	Floating point	1
Moving Average	TASK1/AVG1	Floating point	1
Single Calculation	TASK1/SOC1	Floating point	1
Timer	TASK1/ET1	Long	1
Time Stamp	TASK1/TS1	String	1
Counter	TASK1/CNT1	Long	1
On/Off Control	TASK1/ONF1	Integer	1
PID Control	TASK1/PID1	Floating point	1
DDE Client	TASK1/DDEC1	String	1
Conditional Sound	TASK1/SOUND1	Long	1
Beep	TASK1/SP1	Long	1
Button Control	DISP1/BBTN1	Long	1
Knob Control	DISP1/KNOB1	Floating point	1
Numeric Control	DISP1/NCTL1	Floating point	1
Slider Control	DISP1/SPIN1	Floating point	1
Ameter Display	DISP1/METER1	Floating point	1
Bar Display	DISP1/BAR1	Floating point	1
Indicator Display	DISP1/INDI1	Long	1
Drawing Display	DISP1/CELL1	Long	1
Conditional Button	DISP1/CBTN11	Long	1
Conditional Bitmap	DISP1/BMP1	Long	1
Conditional Text	DISP1/CTXT	String	1
Historical Trend	DISP1/HIST1	Floating point	8
Virtual Tags	VIRTASK/V1	Floating point	1

(a) TASK1 can change to TASK2 to TASK8 that depends on the tag in which TASK. The tag name consists of the block type and ordinal number; for example, AI block with ordinal number 3, then the tag name is AI3.

(b) DISP1 is the title of the display that is assigned in Display Property Menu.

(c) VIRTASK stores the VIRTAG tags that are added by the Add/ Delete Virtual Tags menu.

(d) The maximum string length for string type tags is 128.

11.2 Application Programming Interface (API)

Data center supports three different sets of interface to the outside world: C API, DDE, and OLE automation. C API is the most efficient interface, so it is used for all communication between components of GeniDAQ. OLE automation is designed to interface with OLE-aware applications. It will be the interface to link with OPC (OLE for Process Control) compatible drivers in the future.

11.2.1 C API

C API function calls are listed below:

GetTagID	Return a tag ID to stand for task_name + tag_name
SetData	Set data value to specified tag in data center
GetData	Return data value from specified tag in data center
SetDataExp	Quickly set data value to specified tag in data center by Tag ID
GetDataExp	Quickly return data value from specified tag in data center by Tag ID
GetTagList	Return the tag list of specified task
GetTaskList	Return the task list in GeniDAQ
ReleaseTagList	Release or free the memory allocated by tag list
ReleaseTaskList	Release or free the memory allocated by task list

For the declarations of these function calls, please refer to header file "DBCENTER.H" as follows:

```
#ifndef DBEXPORT_H
#define DBEXPORT_H

#include "dbdata.h"


#ifdef __cplusplus
extern "C"
{
#endif

typedef struct
{
```

```
        /**************************************************************/
        //      DB_INT        0x0000
        //      DB_INT_A      0x1000
        //      DB_LONG                  0x0001
        //      DB_LONG_A     0x1001
        //      DB_FLOAT      0x0002
        //      DB_FLOAT_A    0x1002
        //      DB_LPSTR      0x0004
        //      DB_LPSTR_A    0x1004
        /**************************************************************/
        WORD wDataType;
        WORD wFrom;   //this attribute is only used when pData is an array
        WORD wSize;   //this attribute is only used when pData is an array
                                //and in this data setting function
        DWORD dwTime; //the updated time of the returned data
        MIXDATA Data;
}DB_PARAM;
typedef DB_PARAM far * LPDB_PARAM;

typedef struct T_LIST_RECORD
{
        LPCSTR lpTaskName;
        LPCSTR lpTagName;
        T_LIST_RECORD * Next;
}LIST_RECORD;
typedef LIST_RECORD far * LPLIST_RECORD;

        //wCount: using to tell DBCenter how many data entry this tag has.
extern DWORD CALLBACK AddTag
(
LPCSTR lpTaskName,
LPCSTR lpTagName,
WORD wDataType,
WORD wCount
);

extern DWORD CALLBACK ChangeTagName
(
LPCSTR lpTaskName,
LPCSTR lpOld,
LPCSTR lpNew
);

extern DWORD CALLBACK GetTagID(LPCSTR lpTaskName, LPCSTR lpTagName);

extern BOOL CALLBACK SetData
(
LPCSTR lpTaskName,
LPCSTR lpTagName,
LPDB_PARAM pParam
);
```

```
extern BOOL CALLBACK GetData
(
LPCSTR lpTaskName,
LPCSTR lpTagName,
LPDB_PARAM pParam
);

extern BOOL CALLBACK SetDataExp(LPDB_RECORD lpDB, LPDB_PARAM pParam);

extern BOOL CALLBACK GetDataExp(LPDB_RECORD lpDB, LPDB_PARAM pParam);

extern LPLIST_RECORD CALLBACK GetTagList(LPCSTR lpTaskName);

extern LPLIST_RECORD CALLBACK GetTaskList();

extern void CALLBACK ReleaseTagList(LPLIST_RECORD lpList);

extern void CALLBACK ReleaseTaskList(LPLIST_RECORD lpList);

#ifdef __cplusplus
}
#endif

#endif
```

11.2.2 OLE Automation

OLE Automation function calls are listed below:

GetTagDataLong	Return long integer value of specified tag from data center.
GetTagDataFloat	Return single float value of specified tag from data center.
GetTagDataSTR	Return string value of specified tag from data center.
SetTagDataLong	Set long integer value to specified tag in data center
SetTagDataFloat	Set single float value to specified tag in data center
SetTagDataSTR	Set string value to specified tag in data center

For the declarations of these function calls, please refer to header file "OLEDB.H" as follows:

```
// oledb.h : header file
//
/////////////////////////////////////////////////////////////////////
// COLEDB command target

class COLEDB : public CCmdTarget
{
      DECLARE_DYNCREATE(COLEDB)
protected:
      COLEDB();                      // protected constructor used by dynamic creation

// Attributes
public:

// Operations
public:

// Overrides
      virtual void OnFinalRelease();
```

```
// Implementation
protected:
      virtual ~COLEDB();

      // Generated message map functions
      //{{AFX_MSG(COLEDB)
            // NOTE - the ClassWizard will add and remove member functions here.
      //}}AFX_MSG
      DECLARE_MESSAGE_MAP()
      DECLARE_OLECREATE(COLEDB)
      // Generated OLE dispatch map functions
      //{{AFX_DISPATCH(COLEDB)
      afx_msg long GetTagDataLong(LPCTSTR szTaskName, LPCTSTR szTagName, short
Index);
      afx_msg float GetTagDataFloat(LPCTSTR szTaskName, LPCTSTR szTagName, short
Index);
      afx_msg BSTR GetTagDataSTR(LPCTSTR szTaskName, LPCTSTR szTagName, short
Index);
      afx_msg BOOL SetTagDataLong(LPCTSTR szTaskName, LPCTSTR szTagName, short
Index, long Data);
      afx_msg BOOL SetTagDataFloat(LPCTSTR szTaskName, LPCTSTR szTagName, short
Index, float Data);
      afx_msg BOOL SetTagDataSTR(LPCTSTR szTaskName, LPCTSTR szTagName, short
Index, LPCTSTR Data);
      afx_msg long AddTag(LPCTSTR szTaskName, LPCTSTR szTagName, short wType);
      afx_msg long AddTask(LPCTSTR szTaskName);
      afx_msg void DeleteTask(LPCTSTR szTaskName);
      afx_msg void DeleteTag(LPCTSTR szTaskName, LPCTSTR szTagName);
      afx_msg long ChangeTaskName(LPCTSTR szOldTask, LPCTSTR szNewTask);
      afx_msg long ChangeTagName(LPCTSTR szTaskName, LPCTSTR szOldTag, LPCTSTR
szNewTag);
      //}}AFX_DISPATCH
      DECLARE_DISPATCH_MAP()
};

///////////////////////////////////////////////////////////////////////////
```

The Tag list manipulation functions in OLE Automation are listed below:

SelectTask	Select the desired task to manipulate tag list.
GoTop	Go to the top of tag list.
EndofList	Check whether it is at the end of tag list
GoBottom	Go to the bottom of tag list
GetTaskName	Return the name of specified task
GetTagName	Return the name of specified tag
GoNext	Go to next tag in tag list

For the declarations of these function calls, please refer to header file "TAGLIST.H" , as follows:

```
// taglist.h : header file
//
#include "dbexport.h"
/////////////////////////////////////////////////////////////////////////////
// CTAGLIST command target

class CTAGLIST : public CCmdTarget
{
      DECLARE_DYNCREATE(CTAGLIST)
protected:
      CTAGLIST();                     // protected constructor used by dynamic creation

// Attributes
public:

// Attributes
protected:
      LPLIST_RECORD lpListH;
      LPLIST_RECORD lpListCur;
      LPLIST_RECORD lpListT;

// Operations
public:

// Overrides
      virtual void OnFinalRelease();

// Implementation
protected:
      virtual ~CTAGLIST();

      // Generated message map functions
      //{{AFX_MSG(CTAGLIST)
            // NOTE - the ClassWizard will add and remove member functions here.
      //}}AFX_MSG
      DECLARE_MESSAGE_MAP()
      DECLARE_OLECREATE(CTAGLIST)
      // Generated OLE dispatch map functions
      //{{AFX_DISPATCH(CTAGLIST)
      afx_msg BOOL SelectTask(LPCTSTR szTaskName);
      afx_msg void GoTop();
      afx_msg BOOL EndofList();
      afx_msg void GoBottom();
      afx_msg BSTR GetTaskName();
      afx_msg BSTR GetTagName();
      afx_msg void GoNext();
      //}}AFX_DISPATCH
      DECLARE_DISPATCH_MAP()
};

/////////////////////////////////////////////////////////////////////////////
```

The Task list manipulation functions in OLE Automation are as listed below:

GoTop   Go to the top of task list.
EndofList   Check whether it is at the end of task list
GoBottom   Go to the bottom of task list
GoNext   Go to next tag in tag list
Get   Return current task in task list

For the declarations of these function calls, please refer to header file "TASKLIST.H", as follows:

```
// tasklist.h : header file
//
#include "dbexport.h"
/////////////////////////////////////////////////////////////////////
// CTASKLIST command target

class CTASKLIST : public CCmdTarget
{
      DECLARE_DYNCREATE(CTASKLIST)
protected:
      CTASKLIST();                    // protected constructor used by dynamic creation

// Attributes
public:

// Attributes
protected:
      LPLIST_RECORD lpListH;
      LPLIST_RECORD lpListCur;
      LPLIST_RECORD lpListT;
```

```
// Operations
public:

// Overrides
      virtual void OnFinalRelease();

// Implementation
protected:
      virtual ~CTASKLIST();

      // Generated message map functions
      //{{AFX_MSG(CTASKLIST)
            // NOTE - the ClassWizard will add and remove member functions here.
      //}}AFX_MSG
      DECLARE_MESSAGE_MAP()
      DECLARE_OLECREATE(CTASKLIST)
      // Generated OLE dispatch map functions
      //{{AFX_DISPATCH(CTASKLIST)
      afx_msg void GoTop();
      afx_msg void GoBottom();
      afx_msg BSTR Get();
      afx_msg void GoNext();
      afx_msg BOOL EndofList();
      //}}AFX_DISPATCH
      DECLARE_DISPATCH_MAP()
};

/////////////////////////////////////////////////////////////////////////
```

12

Writing User Defined DLLs

12 Writing User Defined DLL's

12.1 Introduction

The User DLL is a powerful feature that provides the ability to create customized Task Blocks. Using various parameters and functions invoked through a C or C++ program, information is passed between the User DLL program and the GeniDAQ data acquisition program.

It is assumed in this document that the user has a working knowledge of the Microsoft or Borland C languages and of the Microsoft Software Development Kit (SDK) for WINDOWS 3.1+. This is required knowledge for successful writing of User DLLs, and it is strongly recommended that you go no further before fully acquainting yourself with these two areas.

User DLLs add power to an application. For example, they can be used to compute long mathematical formulae, to compute a power spectrum and return the four largest harmonics, to add advanced control algorithms to an application, or to change the interface of a hardware device to make it behave like a device driver.

GeniDAQ 3.x allows for the creation and use of User DLL's. User DLL's open the architecture, and will greatly expand the capability of GeniDAQ software through a growing library of specialized User DLL's.

The combination of the User DLL's and the Task Designer creates an advanced object oriented development environment. This environment allows extremely rapid development and prototyping of real-time application software.

The GeniDAQ Task Designer lets users represent their User DLL programs with personalized Icons. The graphic below shows three such icons — USER1, TAB2 and SQWAVE3. The USER1 icon indicates that the corresponding DLL module cannot be found when GeniDAQ tries to install it. Either it is not a valid User DLL or it is not in the working directory or in the \WINDOWS or \WINDOWS\SYSTEM directory. If you see this icon, try moving the User DLL file to one of these locations.

Minimum Requirements
Microsoft Visual C/C++ or any compiler capable of compiling WINDOWS DLL's,
GeniDAQ 3.x.

Marketing Information
If you would like to let other GeniDAQ users know about a User DLL that you have developed, just send us the name and a description of its function, as well as contact information (name of contact person and phone number). We will provide a list of available User DLL's to our customers through the Advantech BBS or by request. Customers will then contact you directly for sales and support. Advantech can also provide DLL development services to the customers who need to have a specific function implemented as a DLL.

Installation

User DLL package installation in complete when the GeniDAQ 3.x package is installed. The User DLL development kit provided with GeniDAQ 3.x has three subdirectories, \INCLUDE, \USERDLL\SQWAVE and \USERDLL\STRTABLE.

The \INCLUDE directory contains the header file (gendata.h) you will need in which to compile and link your User DLLs.

The \USERDLL\SQWAVE and \USERDLL\STRTABLE directories contain example User DLL's and all associated source code. The SQWAVE example User DLL generates a pulse train of varying magnitude and duration for positive and negative half-cycles. The STRTABLE User DLL accepts an input to the block and outputs a string based on the block input value. A text file contains a table of inputs and corresponding output strings.

Toolbox Installation of User DLLs

You can install User DLLs into the GeniDAQ Toolbox and use them in your strategy. To install User DLLs:

1) Click on the SETUP menu item in the GeniDAQ Strategy Designer

2) You will see a dialog box where you can install your User DLL by specifying and pressing the OK button. Your User DLL icon will now be present in the Strategy Designer toolbox.

`GENIE.TOL File`

When adding your User DLL to the GeniDAQ Task Designer toolbox, GeniDAQ modifies the file named GENIE.TOL. This file defines all installed User DLL's. If you wish to delete your User DLL from the toolbox, simply use any ASCII editor to modify the GENIE.TOL file.

12.2 Overview

The GeniDAQ User DLL development kit includes the header gendata.h that you must link with your C/
C++ program so it can communicate with the GeniDAQ data acquisition program.

The gendata.h header defines the data types that you can use in your DLL:

```
#define TAGSIZE         9              //  How many characters for icon's tagname
#define DESCSIZE        31             //  How many characters for icon's de-
scription

// the following  macros are for defining  data type passed between DLL and main
program.
#define GDT_NULL        0x0000      //  NULL data type
#define GDT_LONG        0x0001      //  LONG
#define GDT_INT         0x0001      //  INTEGER
#define GDT_FLOAT       0x0002      //  FLOAT
#define GDT_LPSTR       0x0004      //  STRING
#define GDT_USERDEF1    0x1000      //  User Defined
#define GDT_USERDEF2    0x2000      //  User Defined
```

The User DLL actually becomes part of the GeniDAQ Strategy Designer Toolbox, and during Runtime
communicates with GeniDAQ just like all other icon blocks.

To write your User DLL, you must write and export functions that GeniDAQ calls during design time
(Task Designer, GENIE.EXE) and run time (GeniDAQ Runtime, GWRUN.EXE). These functions allow
your DLL to interface with the rest of the GeniDAQ system.

GeniDAQ Task Designer interface functions include:

```
IsUserDefinedDLL()   Tells GeniDAQ whether or not this is a valid User DLL.
GetDataArea()        Allocates data structure memory.
FreeDataArea()       Frees data structure memory.
StartClicked()       Indicates what happens when you connect this icon block to
                     another icon.
EndClicked()         Indicates what happens when you connect another icon block to
                     this icon block.
DoubleClicked()      Indicates what happens when icon block is double-clicked upon,
                     (usually dialog box)
```

GeniDAQ Runtime interface functions include:

```
PrepareToRun()       When user just presses the start button during runtime
Run()                What happens once per scan during runtime
StopRunning()        What happens when user presses stop button during runtime
```

Function Reference:

IsUserDefinedDLL()
Used to tell whether this DLL matches the format of a GeniDAQ User defined DLL. GeniDAQ calls this function when you try to add a DLL to the GeniDAQ toolbox. If this function is not exported by the DLL, then your icon won't appear in the toolbox.

GetDataArea()
Used to allocate a user-defined data area. This data area is used for a communication channel between the GeniDAQ application and the user designed DLL.

In general, this function allocates memory for the USERDATA and RUNDATA. These are data structures that you must define, adding any custom variables at the bottom of the USERDATA structure as follows:

```
/*****************************************************************************/
/*      Structure used for runtime data variables. Users shouldn't change this    */
/*      structure.                                                           */
/*****************************************************************************/

typedef struct _RUNDATA           //  Runtime Data Structure
{
        WORD wDataType; //data types defined in gendata.h
        UINT uLabelID;
        union {
                long            l;
                float           f;
                unsigned long   u;
                int        i;
                unsigned short  w;
                char            c;
                unsigned char   b;
                LPSTR      p;
            } data;
} RUNDATA;
typedef RUNDATA FAR* LPRUNDATA;

typedef struct _USERDATA          //  User Data Structure
{
```

```
/**************************************************************************/
/*     The following lines must be kept the same for all  User DLL's.    */
/*     These attributes are most important in the communication between   */
/*     the DLL and Genie.                                                 */
/**************************************************************************/
    int cbSize;                    // size of _USERDATA struct
    HWND hWndParent;               // GenView win handle
    char szTagName[TAGSIZE];       //user block's tagname
    char szDescription[DESCSIZE];  //user block's description
    HGLOBAL hthisData;             //memory handle of user data
    HGLOBAL hInData;               //memory handle of inData
    HGLOBAL hOutData;              //memory handle of outData
    int outDataCnt;                //num of outputs
    LPRUNDATA outData;             // output data of this block.
    int inDataCnt;                 // num of inputs
    LPRUNDATA inData;              // input data of this block
    HGLOBAL hUserBlock;            //memory handle of user defined data block
    LPVOID lpUserBlock;            //far pointer of user defined data block

/**************************************************************************/
/*     The following lines are a private data area, used just by this object.  */
/*     Users can define their own attributes here.                       */
/**************************************************************************/

        float variable1;
        float variable2;
        int variable3;
        //  Etc.
} USERDATA;

typedef USERDATA FAR* LPUSERDATA;
```

FreeDataArea(LPUSERDATA lpUserData)
Frees user memory allocation for this DLL. Users should free the memory allocated during
GetDataArea() here to prevent any memory problems.

StartClicked (LPUSERDATA lpUserData)
When a customer tries to connect the DLL's icon to the other icon(s) in Genie application this function
will be called. If there is output from the block, the function must return CLK_OK. If no outputs exist,
return CLK_REJECTSTART. Definitions of these return values must be made in your header file as
in the example that follows.

EndClicked (LPUSERDATA lpUserData, int nType)
When customers try to connect some icon object in the Genie application to the DLL icon, this
function will be called. If there is no input to the block, the function must return CLK_REJECTEND. If
there is input to the block, the function must return CLK_OK. Definitions of these return values must
be made in your header file as in the example that follows.

DoubleClicked (LPUSERDATA lpUserData)
When customers double click the mouse's left button on this DLL's icon, this function will be called. In this function, you should design or call a dialog box procedure for creating a popup dialog for the customer to setup their expected value(s).

PrepareToRun (LPUSERDATA lpUserData)
Perform any Runtime initialization here. Just before the strategy is executed (runtime), GeniDAQ will call this function so DLL designers can pre-set their object attribute's default value(s). Return value is zero.

Run (LPUSERDATA lpUserData)
This function is called once per task period. DLL designer should handle all input values and output values to be changed during scanning in this function. Return value is zero.

StopRunning (LPUSERDATA lpUserData)
Any Runtime termination code for your User DLL goes here. When users select STOP menu item or push the STOP button, this function will be called to handle DLL stopping procedure. Return value is zero.

In the sections that follow, we will be stepping through an actual working User DLL coding example. This example is called SQWAVE.DLL (\USERDLL\SQWAVE directory) and is used to output a square wave to other blocks in GeniDAQ.

12.2.1 Compilation

To compile and link, you must use your compiler to generate a Windows DLL. Always include the header file gendata.h. Here is an example of a makefile for Microsoft Visual C++ V1.5, to compile the example sqwave.dll:

```
#*************************************************************************
#  makefile, sqwave.dll
#  Copyright - American Advantech Corp. 1995
#
#*************************************************************************

PROJ = SQWAVE            #driver name
DEBUG = 0                #debug flag
DRV_SEG =  DRV_MAIN      #Define driver code segment
DLG_SEG =  DRV_DLG       #define dialog code segment
MODEL =M                 #memory model
WARNING=3                #warning level

# C compile options
CC  = cl
CFLAGS_G  = /A$(MODEL) /W$(WARNING) /G2 /Zp /BATCH  /NT  #general options
CFLAGS_D  =  /Od /Zi /Gsw                                #debug options
CFLAGS_R  =  /O2 /Gsw            #non-debug options

# C++ compile options
CXX  = cl
CXXFLAGS_G  = /G2 /W$(WARNING) /ASw  /Zp /BATCH  #general options
CXXFLAGS_D  =  /Zi /Od /Gsw                      #debug options
CXXFLAGS_R  =  /O2 /Gsw               #non-debug options

MAPFILE_D  = NUL
MAPFILE_R  = NUL

# Linker option
LFLAGS_G  = /BATCH /ONERROR:NOEXE    #general options
LFLAGS_D  = /CO /NOE              #debug options
LFLAGS_R  = /NOE                  #non-debug options
LLIBS_G  = LIBW.LIB               #general library
LINKER  = link
ILINK  = ilink
LRF  = echo > NUL
ILFLAGS  = /a /e
RC  = rc
IMPLIB  = implib
LLIBS_R  = /NOD $(MODEL)DLLCEW
LLIBS_D  = /NOD $(MODEL)DLLCEW
```

```
FILES  = SQWAVE.C SQWAVE.RC SQWAVE.H SQWDLG.H GENDATA.H\
        SQWAVE.DEF

#path for header files
HEADERS = ..\INCLUDE\\
INCLUDE = $(INCLUDE);..\INCLUDE

DEF_FILE  = SQWAVE.DEF         #definition

#object files
OBJS  = SQWAVE.obj
RESS  = SQWAVE.res

all: $(PROJ).res $(PROJ).dll

.SUFFIXES:
.SUFFIXES: .obj .res .c .rc


SQWAVE.res : SQWAVE.RC sqwave.h sqwdlg.h
        $(RC) $(RCFLAGS1) /r /fo $*.res $*.RC


SQWAVE.obj : SQWAVE.C sqwave.h sqwdlg.h $(HEADERS)gendata.h
!IF $(DEBUG)
        @$(CC) @<<$(PROJ).rsp
/c $(CFLAGS_G) $(DRV_SEG)
$(CFLAGS_D) /Fo$*.obj $*.C
<<
!ELSE
        @$(CC) @<<$(PROJ).rsp
/c $(CFLAGS_G) $(DRV_SEG)
$(CFLAGS_R) /Fo$*.obj $*.C
<<
!ENDIF
```

```
$(PROJ).dll : $(DEF_FILE) $(OBJS) $(PROJ).res
!IF $(DEBUG)
        $(LRF) @<<$(PROJ).lrf
$(RT_OBJS: = +^
) $(OBJS: = +^
)
$@
$(MAPFILE_D)
$(LIBS: = +^
) +
$(LLIBS_G: = +^
) +
$(LLIBS_D: = +^
)
$(DEF_FILE) $(LFLAGS_G) $(LFLAGS_D);
<<
!ELSE
        $(LRF) @<<$(PROJ).lrf
$(RT_OBJS: = +^
) $(OBJS: = +^
)
$@
$(MAPFILE_R)
$(LIBS: = +^
) +
$(LLIBS_G: = +^
) +
$(LLIBS_R: = +^
)
$(DEF_FILE) $(LFLAGS_G) $(LFLAGS_R);
<<
!ENDIF
        $(LINKER) @$(PROJ).lrf
        $(RC) $(RCFLAGS2) $(RESS) $@
```

12.3 Header File

_____

gendata.h

Generic header to be used by all User DLL's. This header must be included and cannot be modified
by the user. It defines the data types used between blocks in GeniDAQ. The user will use these data
types when transferring block data. See source code for examples.

_____

```
#ifndef GENDATA_H
#define GENDATA_H

#define TAGSIZE            9
#define DESCSIZE           31

// the following  macros are for defining  data type passed between DLL and main
program.
#define GDT_NULL           0x0000
#define GDT_LONG           0x0001
#define GDT_INT            0x0001
#define GDT_FLOAT          0x0002
#define GDT_LPSTR          0x0004
#define GDT_USERDEF1       0x1000
#define GDT_USERDEF2       0x2000

#endif
```

_____

sqwave.h

Contains data structures and function prototypes for User DLL functions. A similar header must be
created for any User DLL that includes these data structures.

_____

```
#include "gendata.h"

#define CLK_REJECTSTART    0
#define CLK_REJECTEND      0
#define CLK_OK             1
#define CLK_REJECTQUIET    2
#define CLK_OKLONG         10
#define CLK_OKFLOAT        11
#define CLK_OKSTRING       12

#define ST_LOW             0
#define ST_HIGH            1

#define INDATACNT          0       // Number of block inputs
#define OUTDATACNT         1       // Number of block outputs

/
```

```
/********************************************************************************/
/*      Structure used for runtime data variables. Users shouldn't change this    */
/*      structure.                                                                 */
/********************************************************************************/

typedef struct _RUNDATA            //  Runtime Data Structure
{
        WORD wDataType; //data types defined in gendata.h
        UINT uLabelID;
        union {
                long            l;
                float           f;
                unsigned long   u;
                int             i;
                unsigned short  w;
                char            c;
                unsigned char   b;
                LPSTR           p;
             } data;
} RUNDATA;
typedef RUNDATA FAR* LPRUNDATA;


typedef struct _USERDATA           //   User Data Structure
{

/********************************************************************************/
/*      The following lines must be kept the same for all  User DLL's.            */
/*      These attributes are most important in the communication between          */
/*      the DLL and Genie.                                                         */
/********************************************************************************/
    int cbSize;                    // size of _USERDATA struct
    HWND hWndParent;               // GenView win handle
    char szTagName[TAGSIZE];       //user block's tagname
    char szDescription[DESCSIZE]; //user block's description
    HGLOBAL hthisData;             //memory handle of user data
    HGLOBAL hInData;               //memory handle of inData
    HGLOBAL hOutData;              //memory handle of outData
    int outDataCnt;                //num of outputs
    LPRUNDATA outData;             // output data of this block.
    int inDataCnt;                 // num of inputs
    LPRUNDATA inData;              // input data of this block
    HGLOBAL hUserBlock;            //memory handle of user defined data block
    LPVOID lpUserBlock;            //far pointer of user defined data block

/********************************************************************************/
/* The following lines are a private data area, used just by this object.         */
/* Users can define their own attributes here.                                    */
/********************************************************************************/

        float hi_magnitude;        // square wave high magnitude
        float lo_magnitude;        // square wave low magnitude
        int hi_duration;           // high pulse duration
        int lo_duration;           // low pulse duration
        int counter;               // pulse counter
        int state;                 // high or low state
} USERDATA;
```

```
typedef USERDATA FAR* LPUSERDATA;


                              // Functionsthat GeniDAQ calls for User DLL Operation
LPUSERDATA FAR PASCAL __export GetDataArea(void);              //  for memory alloca-
tion
void  FAR PASCAL __export FreeDataArea(LPUSERDATA lpUserData); // for freeing of
memory
int   FAR PASCAL __export StartClicked(LPUSERDATA lpUserData); // for output
connection
int   FAR PASCAL __export EndClicked(LPUSERDATA lpUserData, int nType);  // for
input connection
void  FAR PASCAL __export DoubleClicked(LPUSERDATA lpUserData); // for configura-
tion dialog box
DWORD FAR PASCAL __export PrepareToRun(LPUSERDATA lpUserData); // for runtime
initialization
DWORD FAR PASCAL __export Run(LPUSERDATA lpUserData); // for runtime scanning
DWORD FAR PASCAL __export StopRunning(LPUSERDATA lpUserData); // for runtime
termination
BOOL  FAR PASCAL __export IsUserDefinedDLL(void);  // for determining if User DLL



_____−

sqwdlg.h

Unique Dialog Box Procedure Resource Control ID's for SQWAVE.DLL.
_____−
#define IDD_SQW           200
#define IDC_TAG           202
#define IDC_HIMAG         206
#define IDC_DESC          204
#define IDC_LOMAG         208
#define IDC_HIDUR         210
#define IDC_LODUR         212


///////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////
```

12.4 Definition File

_____

sqwave.def

Required definition file for creating DLL with exportable functions — you must export these functions also.
_____

```
LIBRARY       SQWAVE
DESCRIPTION   'Genie Example Resource DLL'
EXETYPE       WINDOWS
STUB          'WINSTUB.EXE'
CODE          LOADONCALL MOVEABLE
DATA          PRELOAD MOVABLE SINGLE

HEAPSIZE      1024

EXPORTS       WEP           @1 RESIDENTNAME
              GetDataArea       @2
              FreeDataArea      @3
              StartClicked      @4
              EndClicked        @5
              DoubleClicked     @6
              PrepareToRun      @7
              Run               @8
              StopRunning       @9
              IsUserDefinedDLL  @10
```

//
//

12.5 Resource File

sqwave.rc

Dialog Box Resource file for setting of parameters during Strategy Design time. This is a custom dialog box designed specifically for this User DLL. You design your own custom dialog box for your User DLL.

```
#include <windows.h>
#include "sqwdlg.h"


/*————————————————————————————————*/
/*    bitmaps                      */
/*————————————————————————————————*/
                                        //  User defined icon block
bitmaps
up_bmp              BITMAP       sqw_up.bmp  //  Bitmap for toolbox in up position
down_bmp            BITMAP       sqw_dn.bmp  //  Bitmap for toolbox in down posi-
tion
block_bmp           BITMAP       sqw_blk.bmp //  Bitmap for block in strategy

/*————————————————————————————————*/
/*    dialog box                   */
/*————————————————————————————————*/

IDD_SQW DIALOG 79, 29, 194, 119
STYLE DS_MODALFRAME | WS_POPUP | WS_VISIBLE | WS_CAPTION | WS_SYSMENU
CAPTION "Square Wave Generator"
FONT 8, "MS Sans Serif"
BEGIN
    LTEXT           "Tag:", -1, 2, 10, 20, 8
    CONTROL         "", -1, "Static", SS_GRAYFRAME, 20, 7, 45, 15
    LTEXT           "Description:", -1, 72, 10, 42, 8
    EDITTEXT        IDC_DESC, 114, 8, 78, 12, ES_AUTOHSCROLL
    LTEXT           "HI Magnitude:", -1, 6, 39, 49, 8
    EDITTEXT        IDC_HIMAG, 59, 37, 32, 12, ES_AUTOHSCROLL
    LTEXT           "LO Magnitude:", -1, 103, 39, 49, 8
    EDITTEXT        IDC_LOMAG, 154, 37, 32, 12, ES_AUTOHSCROLL
    LTEXT           "HI Duration:", -1, 6, 62, 46, 8
    EDITTEXT        IDC_HIDUR, 59, 60, 32, 12, ES_AUTOHSCROLL
    LTEXT           "LO Duration:", -1, 103, 62, 46, 8
    EDITTEXT        IDC_LODUR, 154, 60, 32, 12, ES_AUTOHSCROLL
    PUSHBUTTON      "Ok", IDOK, 44, 93, 40, 14
    PUSHBUTTON      "Cancel", IDCANCEL, 101, 93, 40, 14
    EDITTEXT        IDC_TAG, 21, 10, 43, 9, ES_AUTOHSCROLL | ES_READONLY |
                    NOT WS_BORDER
    CONTROL         "", -1, "Static", SS_BLACKFRAME, 0, 26, 194, 1
END

///////////////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////////////
```

12.6 C Source Code File

───

sqwave.c

All User DLL function coding resides in this file. Supports all necessary GeniDAQ function calls and the DLL's configuration dialog box procedure. Your coding should support all the same functions, but with unique code tailored for your User DLL.

───

```c
#include <windows.h>
#include <stdlib.h>
#include "sqwave.h"
#include "sqwdlg.h"

static HINSTANCE hInst;
BOOL CALLBACK __export DClickedDlgProc(HWND hDlg, UINT msg, WPARAM wParam,
        LPARAM lParam);
int latoi(LPCSTR lszString);
double latof(LPCSTR lszString);
void lftoa(LPCSTR lszString, float fvalue, int d);

/*****************************************************************************/
/*      SQWAVE.DLL   : This dll accepts 0 inputs and has 1 output           */
/*      (the square wave float)                                             */
/*****************************************************************************/
/*─────────────────────────────────────────── */
/*      standard DLL init call                 */
/*─────────────────────────────────────────── */

int FAR PASCAL LibMain
(
        HINSTANCE hModule,
        WORD    wDataSeg,
        WORD    cbHeapSize,
        LPSTR   lpszCmdLine
)
{
        hInst = hModule;
        return (1);
}

/*───────────────────────────────────────────*/
/*  standard DLL termination call             */
/*───────────────────────────────────────────*/

int CALLBACK WEP
(
        int     bSystemExit
)
{
        return 0;
}
```

```
/*————————————————————————————————————————————————————————*/
/*       GeniDAQ functions — edit to fit your User DLL             */
/*————————————————————————————————————————————————————————*/


//    3.7 GeniDAQ INTERFACE FUNCTIONS



/****************************************************************************/
/*  IsUserDefinedDLL()                                                    */
/*      Purpose: Used to tell whether this DLL matches the format of a    */
*/      Genie User defined DLL.                                           */
/****************************************************************************/

BOOL FAR PASCAL __export IsUserDefinedDLL()
{
        return(TRUE);
}




/
/****************************************************************************/
/*      GetDataArea()                                                     */
/*      Purpose:     Allocates a user defined data area.                  */
/*      This data area is used for a communication channel between        */
/*      the Genie application and the user designed DLL.                  */
/*      Note:  There are several things that users must be concerned of:  */
/*      1. allocate USERDATA memory.                                      */
/*      2. allocate inData memory, if this object accepts input data.     */
/*      3. allocate outData memory, it this object must output data.      */
/*      4. Remember to free the memory that this DLL                      */
/*      allocated in FreeDataArea function call.                          */
/*      Suggestion:Users can keep most this function contents the same,   */
/*      except INDATACNT, OUTDATACNT definition, data type                */
/*      assignment, label ID assignment and user local attribute          */
/*      initialization.                                                   */
/****************************************************************************/

LPUSERDATA FAR PASCAL __export GetDataArea()
{
```

```
/******************************************************************************/
/* Try to keep this part the same for all User DLL's , except the definition of */
/*  INDATACNT = # block inputs and OUTDATACNT = # block outputs                */
/*  ( defined in  header file).                                               */
/******************************************************************************/
        LPUSERDATA lpUserData;
        HGLOBAL hData;

        hData = GlobalAlloc(GMEM_SHARE, (DWORD)sizeof(USERDATA));    //  Allocate
memory
        lpUserData = (LPUSERDATA)GlobalLock(hData);
        lpUserData->hthisData = hData;

        lpUserData->inDataCnt = INDATACNT;
        if (INDATACNT > 0)
        {
                lpUserData->hInData = GlobalAlloc(GMEM_SHARE,
                                    (DWORD)sizeof(RUNDATA)*INDATACNT);
                lpUserData->inData = (LPRUNDATA)GlobalLock(lpUserData->hInData);
        }
    else
    {
                lpUserData->inData = NULL;
                lpUserData->hInData = 0;
        }

        lpUserData->outDataCnt = OUTDATACNT;
        if (OUTDATACNT > 0)
        {
                lpUserData->hOutData = GlobalAlloc(GMEM_SHARE,
                              (DWORD)sizeof(RUNDATA)*OUTDATACNT);
                lpUserData->outData = (LPRUNDATA)GlobalLock(lpUserData->hOutData);
        }
    else
    {
                lpUserData->outData = NULL;
                lpUserData->hOutData = 0;
        }
        lpUserData->cbSize = sizeof(USERDATA);

        lpUserData->outData[0].wDataType = GDT_FLOAT;  // Type of Output Data for
this block
        lpUserData->outData[0].uLabelID = 0;  //  Label ID # starts at zero
        lstrcpy(lpUserData->szTagName, "SQWAVE");  //  Tagname for Block (change to
desired name)

/******************************************************************************/
/*          Initialization of custom users dll private attributes            */
/******************************************************************************/
```

```
        lpUserData->hi_magnitude = 5.0f;  // square wave magnitude and duration ini-
tialization
        lpUserData->lo_magnitude = -5.0f;  //  these attributes can be changed in
dialog box procedure
        lpUserData->hi_duration = 3;      //  by GeniDAQ user
        lpUserData->lo_duration = 3;

        return (lpUserData);
}




/*****************************************************************************/
/*  FreeDataArea(LPUSERDATA lpUserData)                                      */
/*     Purpose:  Free user's memory allocation for this DLL.                 */
/*     Notes: Users should free their memory allocation here to             */
/*            prevent any memory problems.                                   */
/*****************************************************************************/
void FAR PASCAL __export FreeDataArea(LPUSERDATA lpUserData)
{
        //user's private attribute for freeing memory

        //communication data free memory statements
        HGLOBAL hData;
        if (lpUserData->hInData != 0)
        {
                GlobalUnlock(lpUserData->hInData);
                GlobalFree(lpUserData->hInData);
        }
        if (lpUserData->hOutData != 0)
        {
                GlobalUnlock(lpUserData->hOutData);
                GlobalFree(lpUserData->hOutData);
        }
        hData = lpUserData->hthisData;
        GlobalUnlock(hData);
        GlobalFree(hData);
}
```

```
/****************************************************************************/
/* StartClicked(LPUSERDATA lpUserData)                                      */
/*     Purpose: When customer tries to connect the DLL's icon to the        */
/*     other icon(s) in Genie application this function will be called.      */
/****************************************************************************/

int FAR PASCAL __export StartClicked(LPUSERDATA lpUserData)
{
        return(CLK_OK);
}




/****************************************************************************/
/* EndClicked(LPUSERDATA lpUserData, int nType)                             */
/*     Purpose:  When customers try to connect some icon object in the      */
/*     Genie application to the DLL icon, this function will be called.      */
/****************************************************************************/

int FAR PASCAL __export EndClicked(LPUSERDATA lpUserData, int nType)
{
        return(CLK_REJECTEND);   //  Used for no input(s)
        //return(CLK_OK);                       //  Used if there are inputs
}




/****************************************************************************/
/* DoubleClicked(LPUSERDATA lpUserData)                                     */
/*     Purpose:  When customers double click the mouse's left button        */
/*     on this DLL's icon, this function will be called.                     */
/*     In this function, designer should design a popup dialog for the customer */
/*     dialog for the customer to setup their expected value(s).             */
/****************************************************************************/

void FAR PASCAL __export DoubleClicked(LPUSERDATA lpUserData)
{
        DLGPROC          pfnDlgProc;
        int           rc;

        pfnDlgProc = (DLGPROC)MakeProcInstance ((FARPROC) DClickedDlgProc, hInst);
        rc = DialogBoxParam (hInst, MAKEINTRESOURCE (IDD_SQW),
                lpUserData->hWndParent, pfnDlgProc,
                (LPARAM) (LPVOID) lpUserData);
        FreeProcInstance ((FARPROC) pfnDlgProc);

        return;
}
```

```
/****************************************************************************/
/*      DClickedDlgProc(HWND hDlg,UINT msg,WPARAM wParam,LPARAM  lParam)     */
/*      Purpose:  This procedure is used to handle the dialog main           */
/*      activities.  Users should design their own functions here.          */
/****************************************************************************/

BOOL CALLBACK __export DClickedDlgProc
(
        HWND    hDlg,
        UINT    msg,
        WPARAM  wParam,
        LPARAM  lParam
)
{
        char szString[65];
        GLOBALHANDLE hPropMem;
        LPUSERDATA lpUserData;
        LPUSERDATA _far *plpUserData;

        hPropMem = GetProp (hDlg, "sqwave");
        if (hPropMem)
        {
                plpUserData = GlobalLock (hPropMem);
                lpUserData = *plpUserData;
                GlobalUnlock (hPropMem);
        }

        switch(msg)
        {
        case WM_INITDIALOG:
                {
                        lpUserData = (LPUSERDATA) (LPVOID) (lParam);
                        hPropMem = GlobalAlloc (GHND, (DWORD) sizeof(LPUSERDATA));
                        plpUserData = GlobalLock (hPropMem);
                        *plpUserData = lpUserData;
                        GlobalUnlock (hPropMem);
                        SetProp (hDlg, "sqwave", hPropMem);
                        SetDlgItemText(hDlg, IDC_TAG, lpUserData->szTagName);
                        SetDlgItemText(hDlg, IDC_DESC, lpUserData->szDescription);
                        lftoa((LPCSTR)szString, lpUserData->hi_magnitude, 2);
                        SetDlgItemText(hDlg, IDC_HIMAG, szString);
                        lftoa((LPCSTR)szString, lpUserData->lo_magnitude, 2);
                        SetDlgItemText(hDlg, IDC_LOMAG, szString);
                        wsprintf(szString, "%d", lpUserData->hi_duration);
                        SetDlgItemText(hDlg, IDC_HIDUR, szString);
                        wsprintf(szString, "%d", lpUserData->lo_duration);
                        SetDlgItemText(hDlg, IDC_LODUR, szString);
                        break;
                }
```

```
        case WM_COMMAND:
                switch (wParam)
                {
                case IDOK:
                        GetDlgItemText(hDlg, IDC_DESC, lpUserData->szDescription, 31);
                        GetDlgItemText(hDlg, IDC_HIMAG, szString, 10);
                        lpUserData->hi_magnitude = (float)latof((LPCSTR)szString);
                        GetDlgItemText(hDlg, IDC_LOMAG, szString, 10);
                        lpUserData->lo_magnitude = (float)latof((LPCSTR)szString);
                        GetDlgItemText(hDlg, IDC_HIDUR, szString, 10);
                        lpUserData->hi_duration = latoi(szString);
                        GetDlgItemText(hDlg, IDC_LODUR, szString, 10);
                        lpUserData->lo_duration = latoi(szString);

                case IDCANCEL:
                        RemoveProp(hDlg, "sqwave");
                        GlobalFree(hPropMem);
                        EndDialog(hDlg, 0);
                        break;
                }
                break;

        default:
                return FALSE;
        }

        return TRUE;
}
```

```
// 3.8 USER DEFINED FUNCTION EXAMPLES

/****************************************************************************/
/*      Purpose: User defined function for ASCII to integer              */
/****************************************************************************/

int latoi
(
        LPCSTR lszString
)
{
        int len, i, sum;
        BOOL negative = FALSE;
        len = lstrlen(lszString);
        sum = 0;
        for (i = 0; i < len; i++)
        {
                if (lszString[i] != ' ')
                        break;
        }
        if (lszString[i] == '-')
        {
                negative = TRUE;
                i++;
        }
        for (; i < len; i++)
                sum = sum * 10 + lszString[i] - '0';
        if (negative)
                sum = -sum;
        return(sum);
}
```

```
/******************************************************************************/
/*      Purpose:  User defined function for ASCII to float                    */
/******************************************************************************/

double latof
(
       LPCSTR lszString
)
{
       int len, i;
       double sum, devider;
       BOOL negative = FALSE;
       len = lstrlen(lszString);
       sum = 0.0;
       for (i = 0; i < len; i++)
       {
              if (lszString[i] != ' ')
                     break;
       }
       if (lszString[i] == '-')
       {
              negative = TRUE;
              i++;
       }
       for (; i < len; i++)
       {
              if (lszString[i] == '.')
              {
                     i++;
                     break;
              }
              sum = sum * 10 + lszString[i] - '0';
       }
       devider = 10.0;
       for (; i < len; i++)
       {
              sum += (lszString[i] - '0') / divider;
              devider *= 10;
       }
       if (negative)
              sum = -sum;
       return(sum);
}
```

```
/*****************************************************************************/
/*          User defined function for float to ASCII                        */
/*****************************************************************************/

void lftoa
(
       LPCSTR lszString,
       float  fvalue,
       int    d
)
{
       long tmpLong;
       float tmpFloat;
       int i;
       char tmpString[20];

       tmpLong = fvalue;
       wsprintf(tmpString, "%d", tmpLong);
       lstrcpy(lszString, (LPCSTR)tmpString);
       for (i = 0, tmpFloat = 1.0; i < d; i++)
              tmpFloat *= 10.0f;
       if (fvalue < 0)
              tmpLong = -((fvalue - tmpLong) * tmpFloat);
       else
              tmpLong = (fvalue - tmpLong) * tmpFloat;
       wsprintf(tmpString, ".%d", tmpLong);
       lstrcat(lszString, (LPCSTR)tmpString);
}
```

```
//  3.9 GeniDAQ RUNTIME FUNCTIONS




/*****************************************************************************/
/* PrepareToRun(LPUSERDATA lpUserData)                                    */
/*     Purpose:  Before strategy is executed (runtime), GeniDAQ will call this  */
/*     function to let DLL designers pre-set their object attribute's       */
/*     default value(s).                                                   */
/*****************************************************************************/

DWORD FAR PASCAL __export PrepareToRun(LPUSERDATA lpUserData)
{
      lpUserData->counter = 0;
      return(0L);
}



/*****************************************************************************/
/* Run(LPUSERDATA lpUserData)                                             */
/*     Purpose:  This function is called once per task period.             */
/*     DLL designer should handle all input values and output values       */
/*     tobe changed in this function.                                     */
/*****************************************************************************/

DWORD FAR PASCAL __export Run(LPUSERDATA lpUserData)
{
      lpUserData->counter++;
      if (lpUserData->state == ST_LOW)
      {
            if (lpUserData->counter >= lpUserData->lo_duration)
            {
                  lpUserData->state = ST_HIGH;
                  lpUserData->counter = 0;
            }
            lpUserData->outData[0].data.f = lpUserData->lo_magnitude;
            return(0L);
      } else
      {
            if (lpUserData->counter >= lpUserData->hi_duration)
            {
                  lpUserData->state = ST_LOW;
                  lpUserData->counter = 0;
            }
            lpUserData->outData[0].data.f = lpUserData->hi_magnitude;
            return(0L);
      }
}
```

```
/****************************************************************************/
/* StopRunning(LPUSERDATA lpUserData)                                       */
/*      Purpose:  When users select STOP menu item or push STOP button      */
/*      this function will be called to handle DLL stopping process.        */
/****************************************************************************/

DWORD FAR PASCAL __export StopRunning(LPUSERDATA lpUserData)
{
        return(0L);
}
```

Appendix A:

Runtime Error Codes

Appendix

Runtime Error Code Listing

The following is a list of possible errors and warnings that you may encounter during GeniDAQ Runtime. These error messages can aid tremendously in troubleshooting various hardware problems when running GeniDAQ.

Runtime Error/Warning Codes are as follows:

"Open COM %d Failure"

Where **%d** is the serial communications port number, usually ranging from 1-8. This message flags a fatal error during Runtime initialization. The configured I/O device is capable of serial communications, but for some reason, the chosen communications port is not responding.

Possible causes:

1) The configuration of the driver is invalid. You may have configured the I/O device to be connected to a certain communications port that does not exist in your machine.

2) The port itself is invalid. The port may not exist in your machine. The port may be used already, such as with a mouse or other serial device.

3) The RS-232 communications port hardware is bad.

"Unable to transmit to COM %d Address %d"

Where COM %d is the serial communications port and Address %d is the address of the module, such as with RS-485 based ADAM module addressing.

This problem is possibly caused by bad RS-232 transmitter section hardware.

"Unable to receive from COM %d Address %d"

Where COM %d is the serial communications port and Address %d is the address of the module, such as with RS-485 based ADAM module addressing.

Possible causes:

1) Bad RS-232 or RS-485 bus wiring.

2) Bad remote hardware, such as a bad ADAM module or remote power supply.

3) Bad RS-232 receiver section hardware.

4) Scan rate (sample rate) is set much too fast.

"Invalid Data Received From COM %d Address %d"

Where COM %d is the serial communications port and Address %d is the address of the module, such as with RS-485 based ADAM module addressing.

1) Bad remote hardware, such as a bad ADAM module or remote power supply.

2) Bad RS-232 or RS-485 bus wiring.

"Initialization Failure On I/O=%d"

Where %d is the I/O port address. This error usually occurs when there is a driver software problem, such as a configuration error. However, this error can also point to the I/O hardware, such as a data acquisition card that requires programmable gain setting, etc. and checking is done by the driver to see if on-board registers have been initialized properly.

"Device Reset Failure On I/O=%d"

Where %d is the I/O port address. The device resetting routine cannot execute properly. This error usually occurs when there is a driver software problem, such as a configuration error. However, this error can also point to the I/O hardware, such as a data acquisition card that requires programmable gain setting, etc. during board reset. Checking is done by the driver to see if on-board registers have been reset properly.

"Analog Input Initialization Failure On I/O=%d"

Where %d is the I/O port address. The Analog Input section cannot initialize properly. Usually a driver software problem, but can also point to the I/O device hardware Analog Input section, if I/O card register checking is done by the driver.

"Analog Output Initialization Failure On I/O=%d"

Where %d is the I/O port address. The Analog Output section cannot initialize properly. Usually a driver software problem, but can also point to the I/O device hardware Analog Output section, if I/O card register checking is done by the driver.

"Digital Output Initialization Failure On I/O=%d"

Where %d is the I/O port address. The Digital Output section cannot initialize properly. Usually a driver software problem, but can also point to the I/O device hardware Digital Output section, if I/O card register checking is done by the driver.

"Digital Input Initialization Failure On I/O=%d"

Where %d is the I/O port address. The Digital Input section cannot initialize properly. Usually a driver software problem, but can also point to the I/O device hardware Digital Input section, if I/O card register checking is done by the driver.

"Temperature Initialization Failure On I/O=%d"

Where %d is the I/O port address. The Temperature section cannot initialize properly. Usually a driver software problem.Can also point to the I/O device hardware Analog Input section, if I/O card register checking is done by the driver.

"Analog Input Section Failure On I/O=%d"

Where %d is the I/O port address. Cannot receive analog input data from the I/O device. Can be a problem with the driver software. Also could be a problem in the I/O device hardware Analog Input section. However, this error can also occur if the I/O card base address set on the I/O device does not match that of the software configuration setting.

"Analog Output Section Failure On I/O=%d"

Where %d is the I/O port address. Can be a problem with the driver software. Also could be a problem in the I/O device hardware Analog Output section. However, this error can also occur if the I/O card base address set on the I/O device does not match that of the software configuration setting.

"Digital Output Section Failure On I/O=%d"

Where %d is the I/O port address. Can be a problem with the driver software. Also could be a problem in the I/O device hardware Digital Output section. However, this error can also occur if the I/O card base address set on the I/O device does not match that of the software configuration setting.

"Digital Input Section Failure On I/O=%d"

Where %d is the I/O port address. Can be a problem with the driver software. Also could be a problem in the I/O device hardware Digital Input section. However, this error can also occur if the I/O card base address set on the I/O device does not match that of the software configuration setting.

"Temperature Section Failure On I/O=%d"

Where %d is the I/O port address. Cannot receive analog input (temperature) data from the I/O device. Can be a problem with the driver software. Also could be a problem in the I/O device hardware Analog Input section. However, this error can also occur if the I/O card base address set on the I/O device does not match that of the software configuration setting.

"Device Not Found"

I/O device is not properly installed. Either the Strategy Editor I/O Icon Block (AI, AO, DI, DO, or Temperature block) is not properly configured, or the I/O Device does not exist. Configure the block and save the strategy, or enter the Advantech Device Installation Utility and configure the device.

"Function Not Supported On I/O=%d"

Where %d is the I/O port address. Either the driver or GeniDAQ does not support an I/O device's function, or the I/O device does not support a function that the driver or GeniDAQ is trying to access during Runtime. ALWAYS a software problem, since the driver knows all supported functions, and GeniDAQ queries the driver for the supported device features.

"Timer Overrun, slow down the sampling rate"

The computer cannot process one complete scan of the strategy in the time allowed in the Scan Task setup. Allow more time for your scan task.

"Not Enough Memory"

Memory allocation error. Try to free up memory by doing less multi-tasking, install more memory, use a good memory manager program, etc.

"File Not Found"

Occurs when trying to access a file, such as a log file, data file, or strategy file.

"Fail to Get Device List"

Occurs when there is a corrupted or missing "GDEVCFG.INI" file in the Windows directory. This file contains specific configuration information for all of your installed I/O devices. If this file is bad or missing, you must re-create another GDEVCFG.INI file. Create an empty file and run the Advantech Device Installation utility to re-create all desired device instances.

NOTE: To implement strategies created on another system, the GDEVCFG.INI file from the initial system must be copied into the WINDOWS directory of the new system. In addition to this, the *.GNI strategy files must be copied into the GeniDAQ directory of the new system.

The GDEVCFG.INI file contains all of the driver information as configured in the initial system. For your strategy to function properly in another system, this file must accompany the previously created strategies.

Appendix B:

Glossary

Glossary

Actuator
A/D Converter
Add
Analog
C, Celsius, Centigrade
CJC, Cold Junction Compensation
Comp Loop
Configure
Controller, Temperature
Conversion Rate
Counts
D/A Converter
Deadband (Hysteresis)
Differential Input
F, Fahrenheit
Hold
Hz
Impedance
I/O Device
Input Resistance
Install
Instance
K, Kelvin
Measuring Junction
Multiplex
Overshoot
Range
Reference Junction
Remove
Resolution
Response Time
RTD
Secondary Junction
Settling Time
Setup
Signal Conditioner
Single Ended Input
Temperature Limit
Temperature Span
Temperature Stability/Instability
Thermocouple
Thermocouple Break Protection
Transducer
Wheatstone Bridge

Actuator

A device that activates process control equipment by using pneumatic, hydraulic, or electronic signals. For example, a valve actuator is used to control fluid rate for opening and closing valves.

A/D Converter

Analog-to-digital converter. A circuit or device used for producing a set of digital output signals representing the magnitude of a voltage applied to its input. The resolution of the device is proportional to the number of its digital output bits. For instance, a 16-bit A/D converter is higher in resolution than a 12-bit A/D.

Add

To add a device means to increase the number of I/O devices of the same type as the installed device driver. When a device is added to the system, only one driver is installed (using the device DRIVERS utility provided in the Control Panel, Main Window). Configuration information is then added by using the Advantech Device Installation program. You may have more than one I/O device of the same type installed in a system at one time. To add an I/O card, for instance, you would press the ADD button in the Advantech Device Installation Program and then configure the base address, and possibly other parameters.

Analog

In electrical quantities having the property of varying in a continuous, rather than incremental or discrete-step manner.

C, Centigrade, Celsius

Zero degrees = Freezing and 100 degrees = Boiling point of water at sea level.

CJC, Cold Junction Compensation

If two wires of different metals are connected end to end, there are two junctions formed. Current will flow in the resulting circuit if one junction is at a different temperature than the other (Seebeck Coefficient). This effect is the same for either junction except for polarity. Call one of the junctions the measuring junction. Separate the other junction and connect the two leads to a current meter. When making temperature measurements, if the current produced by this other junction (cold junction) is ignored, an unknown error is created. If the temperature of the cold junction is noted at the same time as a measurement is made, then the error from the cold junction can be added or subtracted from the reading to give a correct answer.

Automatic cold junction compensation is accomplished by sensing the terminal temperature (cold junction) with an RTD. The RTD is in a circuit that produces a current equal and opposite to that produced by the cold junction. Thus the current change from the cold junction plus that from the compensation circuit cancel one another. Once this is accomplished, it can then be assumed that the current meter reading represents the current produced by the measuring junction only.

Comp Loop

An extra pair of wires going to the tip of an RTD but not connected to the element, a novel way of lead wire resistance compensation.

Configure

To configure an I/O device is to set the parameters for use with the software the same as those set on the actual I/O device. The values set in the device-specific dialog box should reflect those of the switches and jumpers you have set on the I/O device, board, or card. When an instance of a device is added, it must be configured before it can be used with the software.

Controller, Temperature

A device or software program capable of receiving a signal from a temperature sensing probe within a process and regulating an input to that process in order to maintain a selected temperature (control point).

Conversion Rate

The number of analog-to-digital (A/D) conversions (samples) performed per second.

Counts

The number of events, or events counted by the event counter. The event counter increments or decrements whenever a rising edge of a digital signal is sensed at the counter input.

D/A Converter

A device that converts digital information from the computer into a corresponding analog voltage or current. This allows the computer to control real world events. Analog outputs may directly control equipment in a process that is then measured by an analog input. It is possible to perform closed loop or PID control with the D/A function. Analog outputs can also generate waveforms (function genera-tor). The resolution of the device is proportional to the number of its digital input bits. For instance, a 16-bit D/A converter is higher in resolution than a 12-bit D/A.

Deadband (Hysteresis)

In a digital (on/off) controller, there may be one switching point at which the signal increases and another switching point at which the signal decreases. The difference between the two switching points is called hysteresis or deadband.

Differential Input

A signal-input circuit where SIG LO and SIG HI are electrically floating with respect to analog ground (I/O device analog ground, which is normally tied to digital ground). This allows the measurement of the voltage difference between two signals tied to the same ground and provides superior common-mode noise rejection.

F (Fahrenheit)
32 degrees = Freezing and 212 degrees = Boiling point of water at sea level.

Hold
An external input which is used to stop the process (event counter, ramp, A/D, etc.) temporarily, freezing the device's output at the current value.

Hz
Frequency in cycles per second pulses per second, events per second, etc.

Impedance
Resistance to electrical flow in an AC circuit. It is designated by the symbol Z and is expressed in ohms.

I/O Device
An I/O device is a hardware device capable of data input and output. Generally, this would encompass Analog to digital Conversion (A/D), Digital to Analog conversion (D/A), and Digital Input and Output. The devices capable of such I/O can be in the form of plug-in I/O cards and/or remote devices. The I/O device's hardware should be configured (if necessary) first, and then the software can be configured to match the hardware's settings. Some I/O devices can be configured directly through the software's settings. Check your hardware manufacturer's manual for how to configure the hardware. After any hardware configuration is done, the software configuration can be set up by use of the Advantech Device Installation Program. The device driver should have been previously installed using the DRIVERS program in the CONTROL PANEL (Main Window).

Input Resistance
Resistance measured across the input terminals with the signal leads disconnected.

Install
To install a device means to:

1) Install its device driver in WINDOWS, using the device DRIVERS installation utility provided in the Control Panel, Main window.

2) Once the driver is installed, devices of the driver's type (instances) may be installed (set up or configured), and added by using the Advantech Device Installation Program.

Instance

An instance of an I/O device is a device that has been added and configured uniquely. For instance, there may be three Advantech PCL-818 multi-I/O cards installed in a system. Each PCL-818, installed at separate base addresses and having different parameters configured for each, is an "instance" of the PCL-818 driver (DLL) that was installed in WINDOWS.

K (Kelvin)

The basic temperature unit of the thermodynamic scale, symbol K.   Zero degrees K = zero degrees C - 273.16.

Measuring Junction

That junction of a thermocouple subjected to the temperature to be measured.

Multiplex

A technique which allows different input (or output) signals to use the same lines at different times, controlled by an external signal. Multiplexing is used to save on wiring and I/O ports.

Overshoot

The ratio of the over-travel of the output of a controller beyond a new steady state to the change in steady state when a new constant value of the measured quantity is suddenly applied (step input).

Range

The span of values over which a device will function without entering an overload condition.

Reference Junction

The other junction (usually at ice point) to which the measuring junction thermocouple junction is compared. The output voltage of a thermocouple is approximately proportional to the temperature difference between the measuring (hot) junction and the reference (cold) junction.

Remove

To remove a device means:

Once the driver is installed, devices of the driver's type (instances) may be removed, by highlighting the listing in the dialog box of the Advantech Device Installation Program and pressing the remove button.

To remove the driver means:

Remove the device's driver from WINDOWS, using the device DRIVERS utility provided in the Control Panel, Main window.

Resolution

The degree to which nearly equal values of a quantity can be discriminated. In analog devices, the difference between the values represented by two adjacent divisions. In digital values, the value represented by a one-digit change in the least-significant digit.

Response Time

The time necessary for a device (sensor, A/D, D/A) to reach 63.2% of a step change in measured quantity (temperature, voltage, etc.)

RTD

Resistance Temperature Detector — sensor, bulb, transducer; precision winding of copper, nickel, balco (nickel-iron), platinum (industry standard), or tungsten element used for temperature measurement. Connected via 2, 3, or 4 wire hook-ups.

Secondary Junction

An unwanted connection between a pair of thermocouple wires tending to produce a signal representative of the secondary junction temperature rather than the measuring junction temperature.

Settling Time

The time taken for the display to settle within one digit final value when a step is applied to the A/D input.

Setup

To set up an I/O device is to set the parameters for use with the software the same as the settings of the actual I/O device. The values set in the device-specific dialog box should reflect those of the switches and jumpers you have set on the I/O device, board, or card. Alternatively, if you are using a software programmable device, the device can be configured directly through the device-specific dialog box (in most cases).

Signal Conditioner
A circuit module that offsets, attenuates, linearizes, or filters the signal for input to the A/D converter. The typical output span of a signal conditioner is +/- 5VDC.

Single Ended Input
Amplifier with one input referenced to ground.

Temperature Limit
The full capability of the system from the lowest point to the highest point: limited by the sensor.

Temperature Span
Those two points anywhere within the temperature limit to which the signal conditioner/amplifier can be calibrated. FORMULA: Maximum temperature - minimum temperature = span.

Temperature Stability/Instability
The unwanted change (error) of an instrument, sensor, amplifier, etc. caused by changes in the temperature surrounding it. Usually expressed as the total change between two temperature limits, or % error/degree C or F.

Thermocouple
Two dissimilar metals with a voltage output proportional to temperature. ANSI types:

Type	Composition	Max Degrees F
J	Iron-Constantan	2192
K	Chromel-Alumel	2501
T	Copper-Constantan	752
E	Chromel-Constantan	1832
R	Platinum-Plat.13%Rhodium	3214
S	Platinum-Plat.10%Rhodium	3214
B	Plat.6%Rhod.-Plat.30%Rhod.	3308
C	Tu 5%Rhenium-26%Rhenium	5000

Thermocouple Break Protection
A safety feature to indicate when a thermocouple has failed in an open circuit condition. Its purpose is to eliminate the possibility of an ambiguous reading. In the case of a temperature controller, it eliminates the dangerous condition of thermal runaway.

Transducer
A device which converts temperature, pressure, level, length, position, etc. into a voltage, current, frequency, or pulses, etc.

342

Wheatstone Bridge

A full resistance two (2) wire bridge with RTD, power source, zero and sensitivity adjustments balanced at some reference temperature. Heating or cooling causes a resistance change and discrete circuit unbalance, which indicates the temperature. Various RTD bridge network connections use 2, 3, or 4 wire hook-up arrangements, depending on the accuracy required in the temperature measurement. These are designed to balance out lead wire resistance between the sensor and the bridge. Accuracy obtained is usually:

2 wire +/- >=0.5% (0.5-50 degrees C)
3 wire +/- >=0.25-0.5% (0.25-0.5 degrees C)
4 wire +/- >=0.15-0.25% (0.1-0.25 degrees C)
Index

Index

Index

— A —

Administration 108, 183, 234
Alarm Log 10, 166
Alarm Report 7
Analog Input 121
Analog Output 124
Architecture 3, 288
Array 272
Attach Bitmap 178
Average 143

— B —

Bar Graph 187
BasicScript Block 167, 273, 281
BasicScript block 59, 62
BasicScript Help 28
Beep 104, 155
Block Sequence 6
Break Object 181
Breakpoints 264, 266

— C —

C API 291
Change Password 107, 183, 235
Complete Reorder 119
Conditional Bitmap Display 205
Conditional Button Display 192
Conditional Text Display 208
Conditional Wavefile Block 165
connection line segments 89
Connection Wire 120
Copy DDE Link 100

— D —

Data Center 288
Data Collection 7
Data collection 216

Starting Runtime 243
System Requirements 20

— T —

Tag 6
Task Designer 3
Task Properties 102
Temperature 155
Temperature Scale 156
Testing with Demo I/O Device 23
text string label 197
Time Stamp 158
Timer 10
Toolbox 86, 115, 120, 172, 185, 186, 301
Total Pulse 133
trace script 262
Tutorial 34
Tutorial 1 34
Tutorial 2 43
Tutorial 3 46
Tutorial 4 51
Tutorial 5 55
Tutorial 6 59
Tutorial 7 62
Tutorial 8 66
Tutorial 9 70

— U —

Update rate 123
User Defined DLL 300

— V —

VBA 5
Virtual TAG 62
Virtual Tag 6

— W —

Wait time 151
Watch Variable 266
Whole Average 143

— X —

— Y —

— Z —