GEORGIA SOFTWORKS

Directed Terminal I/O Engine for Windows
NT/XP/VISTA/2000/2003/2008 R2/2012 R2

User Manual

THIS PAGE INTENTIONALLY LEFT BLANK

GEORGIA SOFTWORKS

Directed Terminal I/O (DTIO) Engine

This Page Left Intentionally Blank

Table of Contents

OVERVIEW **3**

PROGRAMMATIC HOOKING 3
OPERATION DESCRIPTION 4
Standard Data Present Model used by SSH2/Telnet *6*
Intelligent Data Present Model used with RF DTIO *7*
Intelligent Write with RF DTIO *8*
Intelligent Read with RF DTIO *9*
GSW RF DTIO 32-BIT / 64-BIT VERSIONS 10

INSTALLATION **11**

REGISTRATION **14**

FLOATING LICENSE – OVERVIEW 14
Floating License – Hardware Key Installation Instructions *16*
Uninstall Floating License – (Hardware Key) *20*
REGISTRATION VIA SOFTWARE SERIAL NUMBER 21
How to Register the Software *21*

RUNNING RF DTIO **25**

RF DTIO Installation Status *25*
Run RF DTIO *25*

CONFIGURATION **27**

General Configuration Screen *28*
Configuration Categories *29*
Bell Processing *30*
Timing Configuration *31*
Trace Configuration *33*
Profiling Configuration *34*
MODE CON 35

RF DTIO PACK (DIRECTED TERMINAL I/O ENGINE & GSW UTS) **36**

GSW RF DIRECTED TERMINAL IO SUBSCRIPTION **36**

HOW TO UPDATE THE SOFTWARE 37
HOW TO RENEW THE GSW SUBSCRIPTION 37
SYSTEM SIGNATURE – PLEASE READ 37

TECHNICAL SUPPORT **38**

TABLE OF FIGURES

Figure 1: Data Flow Overview between Application and Client Device 4
Figure 2: How does SSH2/Telnet know when new data is present? 5
Figure 3: Data Present Model used by SSH2/Telnet 6
Figure 4: Standard Operation without RF DTIO 7
Figure 5: Intelligent Operation with RF DTIO 7
Figure 6: Intelligent Write Data Present Model used by RF DTIO 8
Figure 7: Intelligent Read with RF DTIO Diagram 9
Figure 8: Platform / OS / Software Decision Chart 10
Figure 9: Initial Setup Screen 11
Figure 10: Installation Welcome Screen 11
Figure 11: Installation - Choose Destination Folder 12
Figure 12: Installation - Select Program Folder 12
Figure 13: Installation - Setup Complete 13
Figure 14: Floating License – Parallel Port 15
Figure 15: Floating License - USB Port 15
Figure 16: Floating License – HW Key Initial Splash Screen 16
Figure 17: Floating License – Welcome Screen 17
Figure 18: Floating License - License Agreement 18
Figure 19: Floating License - Accept License Agreement 18
Figure 20: Floating License - HW Key - Installation Status 19
Figure 21: Floating License Drivers Successful Installation 19
Figure 22: Registration with Serial Number - Initial Screen 21
Figure 23: Registration - User Information 22
Figure 24: Registration - Serial Number Applied 23
Figure 25: Registration Successful Screen 23
Figure 26: GSW DTIO is now registered 24
Figure 27: GSW UTS Installation Status Tool 25
Figure 28: RF DTIO Configuration - ini file format 27
Figure 29: RF DTIO Configuration - General 28
Figure 30: Configuration of Bell Processing 30
Figure 31: Configuration - Timing 31
Figure 32: Configuration - Trace Messages and Events 33
Figure 33: Configuration - Profiling 34

Table of Tables

Table 1: Floating Licenses - USB and Parallel 15
Table 2: RF DTIO Engine Configuration Parameters 29
Table 3: Version Upgrade Pricing **with** GSW Subscription Plan 36
Table 4: Version Upgrade Pricing **Without** Subscription Plan 36
Table 5: Steps to renew the GSW Subscription Plan 37

Typographic Conventions

Italics:	are used to emphasize certain words, especially new terms or phrases when they are introduced.
Initial Caps Bold:	Words that appear in initial caps boldface represent menu options, buttons, icons or any object that you may click.
`Courier`:	This font represents anything you must type.
"<**enter**>"	This represents the enter key.

This Page Left Intentionally Blank

Features at a Glance

Georgia SoftWorks Directed Terminal I/O Engine
• Full Support for **Bell Feature** • Increase the Number Sessions on a Server • Increase Performance

Overview

The GSW Directed Terminal I/O Engine Boosts Performance

Thank you for purchasing the Georgia SoftWorks Directed Terminal I/O Engine for Windows.

The GSW Directed Terminal I/O Engine is an add-on component that intercepts a specific set of terminal input/output operating system calls initiated by your Windows 32-bit application[1] / 64-bit application and directs terminal I/O through a specialized high performance interface within the GSW Universal Terminal Server (UTS).

A new performance standard is realized when using the GSW Directed Terminal I/O Engine with the GSW UTS for Windows. Large systems will experience a dramatic performance improvement as well as a substantial increase in the number of sessions on a server.

The Directed Terminal I/O Engine is designed for use with the GSW UTS Server. You will be pleased with the innovative yet seamless integration between the GSW Directed Terminal I/O Engine, your application and the GSW UTS Server.

The GSW DTIO Engine is specialized software, focused on a narrow set of goals. In addition to expected and well know GSW mission critical reliability the primary objectives are to:

1. Provide significant performance improvements with respect to Terminal I/O processing
2. Increase the number of sessions on a server
3. Allow the Bell to sound on the client instead of on the server as one would expect
4. Allow a framework where specialized features can be incorporated on a custom basis

The GSW DTIO is effective on systems where processing bottlenecks/ CPU Limitations are a result of the number of simultaneous or concurrent sessions and/or systems that require Bell processing or specialized features.

Increases in performance by RF DTIO Engine are due to optimizations on a per session basis by reducing the number of instructions executed and number of context switches, and DTIO Super Fine Tuning configuration. GSW RF DTIO performance improvements will be negligible on systems where the processing is not burdened by Terminal I/O.

Programmatic Hooking

The GSW Directed Terminal I/O (DTIO) Engine programmatically hooks the application executable. This involves the injection of a library into the application address space. This is a well established technique similar to one described in the book "Programming Application for Microsoft

[1] If you are using a DOS / 16-bit application please look at GSW DosBoss as RD DTIO does not work with 16-Bit DOS applications.

Windows, Fourth Edition", published by Microsoft Press. Author: Jeffrey Richter. **If you are not the developer/owner of the application software, be sure that you are not in violation of your software license prior to using the GSW RF DTIO Engine**.

Operation Description

This section may be skipped as a detailed understanding is not necessary to enjoy the benefits of the GSW RF DTIO Engine. If you skip this section you can proceed to the installation section (page 10).

To understand how the processing time is reclaimed by the GSW RF DTIO it may be useful to review the data flow mechanism from the application to the device when using SSH2/Telnet.

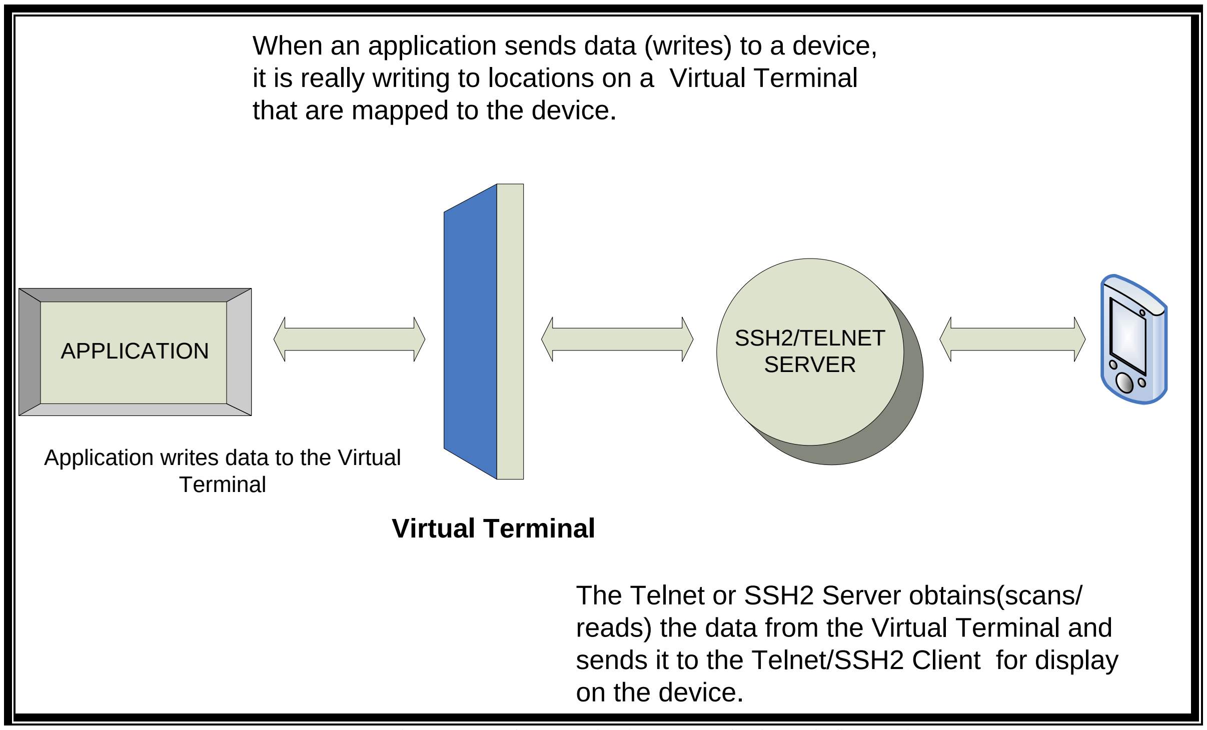

When an application sends data (writes) to a device, it is really writing to locations on a Virtual Terminal that are mapped to the device.
APPLICATION
Application writes data to the Virtual Terminal
Virtual Terminal
SSH2/TELNET SERVER
The Telnet or SSH2 Server obtains(scans/ reads) the data from the Virtual Terminal and sends it to the Telnet/SSH2 Client for display on the device.

Figure 1: Data Flow Overview between Application and Client Device

However, there is a complication with the mechanism used by the Application and the SSH2/Telnet Server to recognize when the application writes data to the Virtual Terminal.

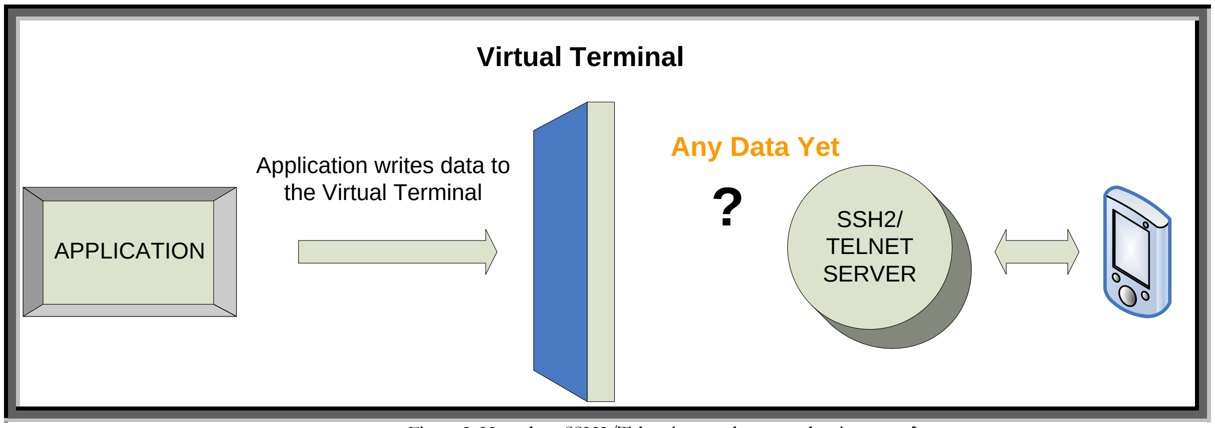


Figure 2: How does SSH2/Telnet know when new data is present?

- The Application and the SSH2/Telnet Server are unrelated programs. The application simply writes data to the VT.
- SSH2/Telnet does not have the intelligence to know when the application has written data to the Virtual Terminal.

So, how does the SSH2/Telnet Server know when there is new data at the Virtual Terminal?

Standard Data Present Model used by SSH2/Telnet

The SSH2/Telnet server periodically polls each row/column location on the Virtual Terminal for new data from the application. In brief the steps are described below.

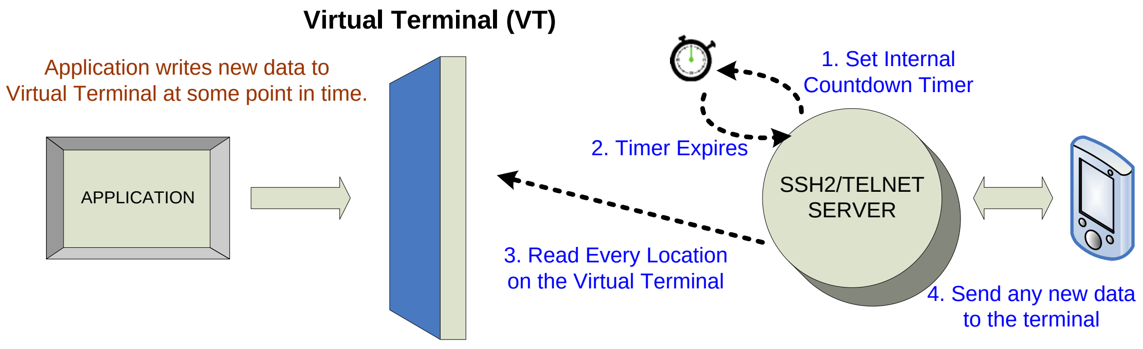

Virtual Terminal (VT)
Application writes new data to Virtual Terminal at some point in time.
APPLICATION
1. Set Internal Countdown Timer
2. Timer Expires
SSH2/TELNET SERVER
3. Read Every Location on the Virtual Terminal
4. Send any new data to the terminal

Figure 3: Data Present Model used by SSH2/Telnet

A countdown timer is set (1). When the timer expires (2) SSH2/Telnet reads (3) every location on the VT to determine if new data is present. If new data is present it is sent (4) to the client device.

This type of operation works very well and is fast. However certain environments demand utmost efficiency so that the overall session throughput is maximized to gain an increased number of sessions and/or optimal session response times.

To achieve higher session throughput, the area with greatest opportunity for processor utilization is the mechanism that identifies when data is present at the Virtual Terminal. The current mechanism works well but extra processing occurs in order to provide the fastest response times to the users.

For example, the countdown timer may expire but there may not be any new data on the VT. If there is no data, there is no need to spend the processing power to read every location (row and column) on the VT. This may not consume much processor time with just a few devices but it’s a different story when there are 40, 100, 200 or more devices.

Typically a screen is scanned for the presence of new data 10 times a second. Two context switches are required each time the screen is scanned, consuming even more time. If the number of sessions is 20, and each screen is scanned 10 times a second and that amounts 200 scans occurring each second. The math is simple; a system with 100 sessions will have 1,000 screen scans performed every second. And 300 sessions requires 3,000 scans every second. On a 1.47 GHz AMD Athlon system running Microsoft Windows XP, this amounts to 117 ms out of every second spent scanning for new data, and that doesn’t include context switches! The unfortunate aspect is that in real world situations, most often there is no new data to transmit.

If your scanner performs 1 scan every 5 seconds then **98% of your polling time is wasted** when no new data is present at the Virtual Terminal.

Intelligent Data Present Model used with RF DTIO

The root of the unnecessary polling is the "Lack of Intelligence" between the application and the SSH2/Telnet Server. Ideally the best solution is for the application to notify the SSH2/Telnet Server when new data is present. This would eliminate the unnecessary time spent polling when no new data is present. The RF DTIO Engine provides this intelligence.

Let's dive a little deeper into what happens when the application read/writes to the Virtual Terminal. The application uses Operating System Application Programming Interfaces (API's) to actually perform the reads and writes.

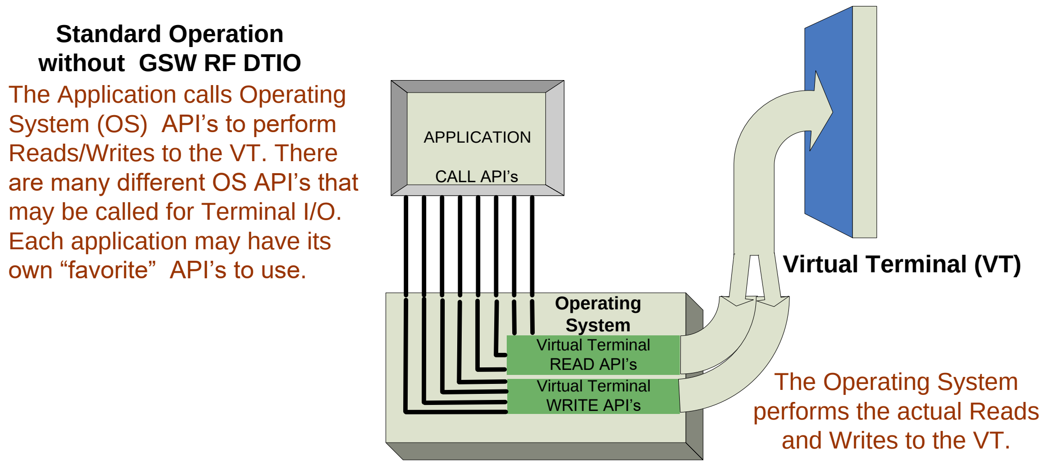

Standard Operation
without GSW RF DTIO
The Application calls Operating System (OS) API's to perform Reads/Writes to the VT. There are many different OS API's that may be called for Terminal I/O. Each application may have its own "favorite" API's to use.
APPLICATION
CALL API's
Operating System
Virtual Terminal READ API's
Virtual Terminal WRITE API's
Virtual Terminal (VT)
The Operating System performs the actual Reads and Writes to the VT.

Figure 4: Standard Operation without RF DTIO

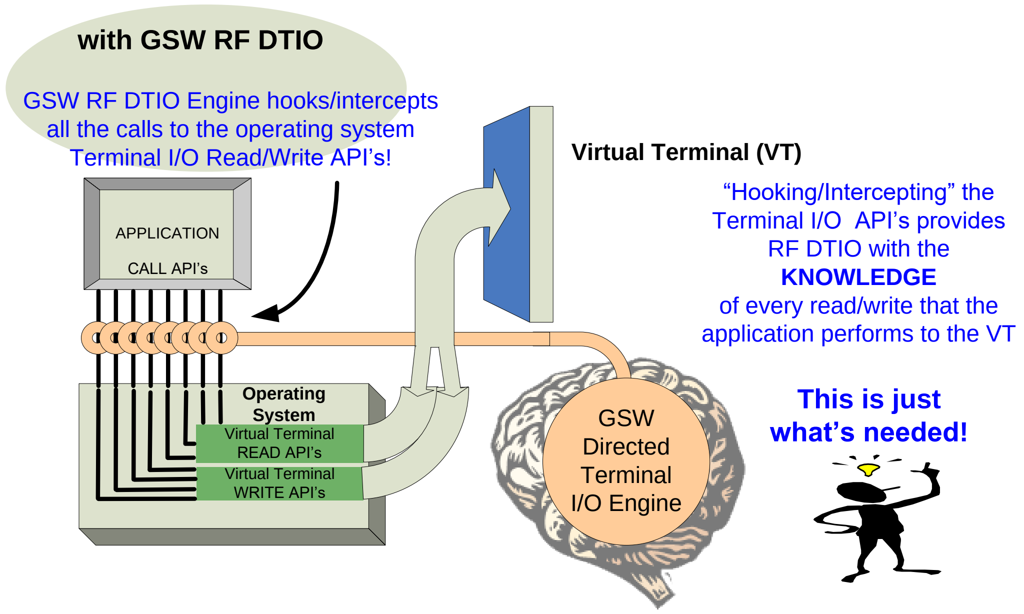

with GSW RF DTIO
GSW RF DTIO Engine hooks/intercepts all the calls to the operating system Terminal I/O Read/Write API's!
APPLICATION
CALL API's
Operating System
Virtual Terminal READ API's
Virtual Terminal WRITE API's
Virtual Terminal (VT)
"Hooking/Intercepting" the Terminal I/O API's provides RF DTIO with the KNOWLEDGE of every read/write that the application performs to the VT
GSW Directed Terminal I/O Engine
This is just what's needed!

Figure 5: Intelligent Operation with RF DTIO

Intelligent Write with RF DTIO

With new intelligence provided by RF DTIO Engine, unnecessary polling/scanning for new data is eliminated. When the application is launched, it is launched by RF DTIO. This allows GSW RF DTIO Engine to "hook" all console I/O API's that the application needs to write/read to the Virtual Terminal.

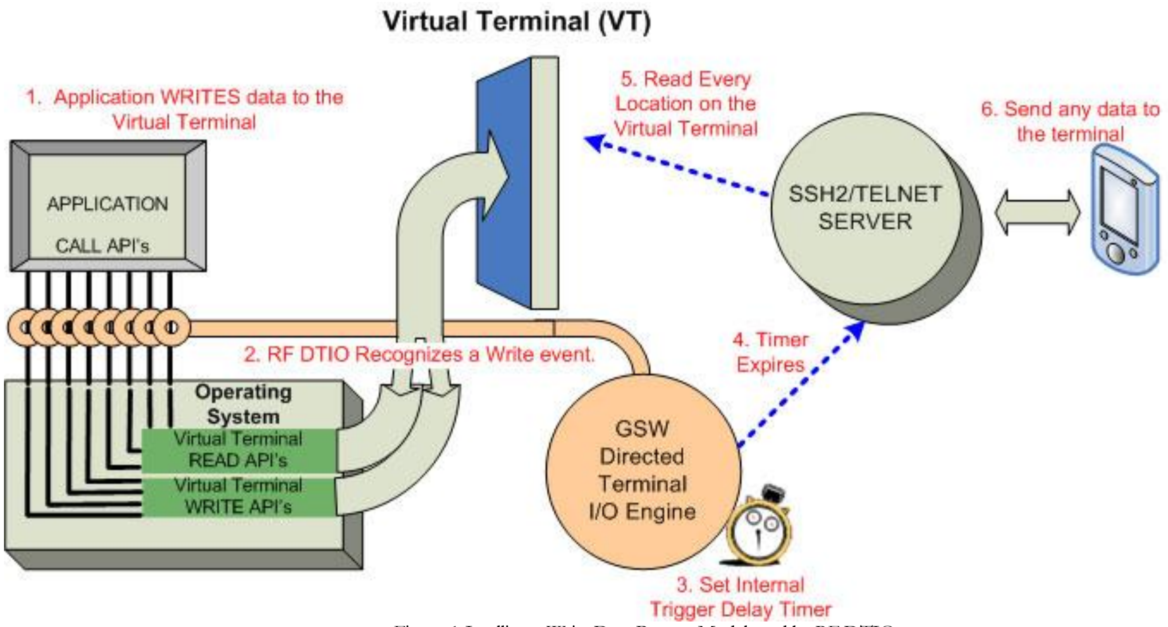

Virtual Terminal (VT)
1. Application WRITES data to the Virtual Terminal
APPLICATION
CALL API's
2. RF DTIO Recognizes a Write event.
Operating System
Virtual Terminal READ API's
Virtual Terminal WRITE API's
GSW Directed Terminal I/O Engine
3. Set Internal Trigger Delay Timer
4. Timer Expires
5. Read Every Location on the Virtual Terminal
SSH2/TELNET SERVER
6. Send any data to the terminal

Figure 6: Intelligent Write Data Present Model used by RF DTIO

Since the RF DTIO "hooks" all Console I/O API's it already has the knowledge every time the application reads or writes data to the Virtual Terminal. Two main cases to review are when the application Reads and when the application Writes.

First, let's look at what happens when the application performs a write. When the application writes (1) to the Virtual Terminal, RF DTIO recognizes (2) the event and sets (3) the Trigger Delay Timer. The trigger delay timer is set because in most cases, applications will write several times before the screen is complete. The timer prevents thrashing by reading too soon and having to read again and again. When the Trigger Delay Timer expires (4) the SSH2/Telnet Server is notified that data is present on the Virtual Terminal. Upon notification the SSH2/Telnet (5) reads (scans) the Virtual Terminal. Next, the data is sent (6) to the terminal.

Intelligent Read with RF DTIO

Let's look at what happens when the application performs a read. Usually (but not always) when an application's writes are complete, a read is performed to see if there is a response from the operator. If the application performs a read after completing screen updates then RF DTIO can be **optionally configured** to provide further performance gains.

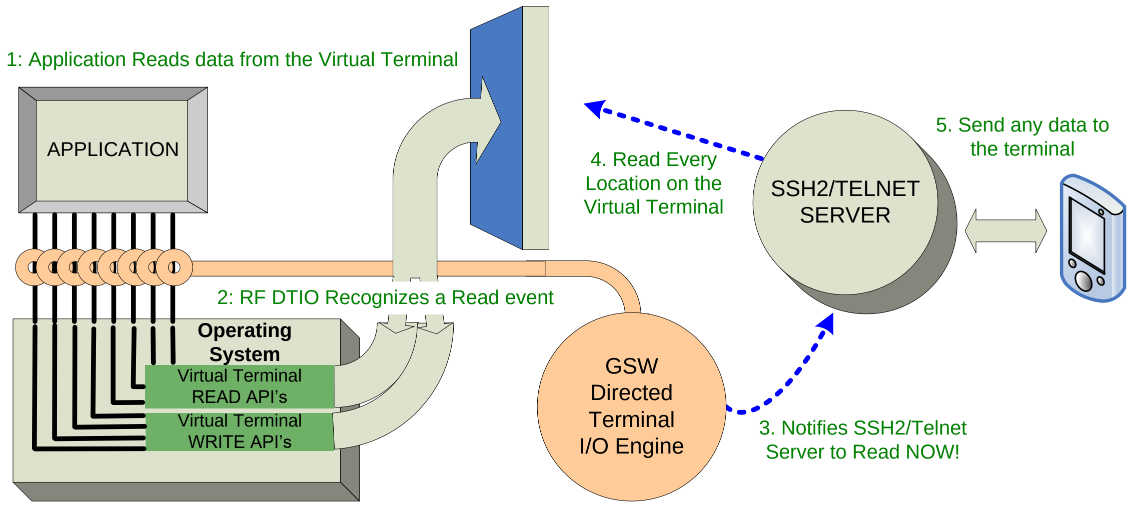

1: Application Reads data from the Virtual Terminal
APPLICATION
2: RF DTIO Recognizes a Read event
Operating System
Virtual Terminal READ API's
Virtual Terminal WRITE API's
GSW Directed Terminal I/O Engine
3. Notifies SSH2/Telnet Server to Read NOW!
4. Read Every Location on the Virtual Terminal
SSH2/TELNET SERVER
5. Send any data to the terminal

Figure 7: Intelligent Read with RF DTIO Diagram

Knowing that an application always performs reads after screen writes allows further optimization such as described here. When the application Reads (1) from the Virtual Terminal, RF DTIO recognizes (2) this event and immediately notifies (3) the SSH2/Telnet Server that data is present on the Virtual Terminal. No timers are set! Upon notification the SSH2/Telnet (4) immediately reads (scans) the Virtual Terminal Next, the data is sent (5) to the terminal.

Note: Some applications have scenarios that do not perform a read after it is done writing. For example, the application may write a message such as "Processing" to the VT while handling intense database lookups (or other time consuming operations) to inform the operator that they have to wait a little longer before the next screen is displayed. Since the application is not waiting on data from the operator a read may not be performed. In this case RF DTIO would not recognize new data until an internal timer fires.

GSW RF DTIO 32-bit / 64-bit Versions

As 64-bit computing rapidly progresses towards mainstream computing, Georgia SoftWorks provides a 64-bit version of the GSW RF DTIO Engine. This version is for 64-bit edition of Microsoft operating systems (XP, VISTA and Windows 2003 Servers). The GSW RF DTIO x64 provides all the performance benefits and addressing capabilities expected when running on 64-bit platforms. Additionally extraordinary high session counts can be attained with the GSW UTS when running on 64-bit platforms.

Both the GSW RF DTIO x86 and GSW RF DTIO x64 versions are included on the CD and both x86 and x64 software libraries are installed when the setup program is run. You select which version is run when you launch the application to run with RF DTIO (See page 25).

The GSW RF DTIO Engine x86 (32-bit version) runs on 64-bit platforms as well as on 32-bit platforms. When running a GSW RF DTIO x86 (32-bit version) on a 64-bit platform significant performance benefits are also realized.

If the application used with the GSW RF DTIO is a 64-bit application then you should use the UTS x64. If the application you are using is a 32-bit application then the GSW RF DTIO x 86 (32-bit) versions is required. Of course hardware platform and operating system choices can boost performance.

The diagram below may assist in selecting the correct combinations of software, operating system and hardware platform.

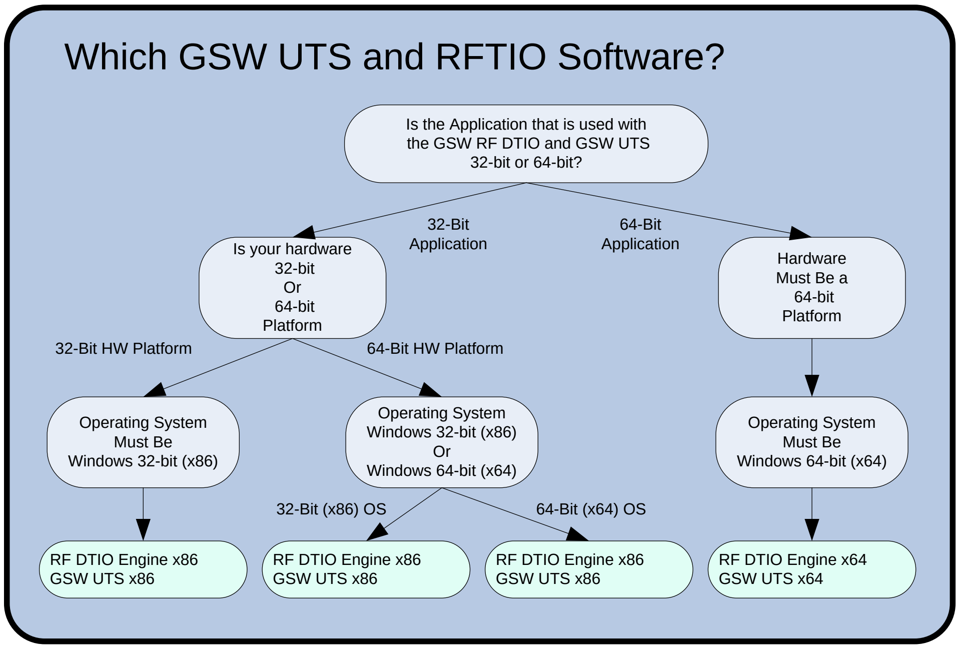

Which GSW UTS and RFTIO Software?
Is the Application that is used with the GSW RF DTIO and GSW UTS 32-bit or 64-bit?
32-Bit Application
64-Bit Application
Is your hardware 32-bit Or 64-bit Platform
Hardware Must Be a 64-bit Platform
32-Bit HW Platform
64-Bit HW Platform
Operating System Must Be Windows 32-bit (x86)
Operating System Windows 32-bit (x86) Or Windows 64-bit (x64)
Operating System Must Be Windows 64-bit (x64)
32-Bit (x86) OS
64-Bit (x64) OS
RF DTIO Engine x86 GSW UTS x86
RF DTIO Engine x86 GSW UTS x86
RF DTIO Engine x86 GSW UTS x86
RF DTIO Engine x64 GSW UTS x64

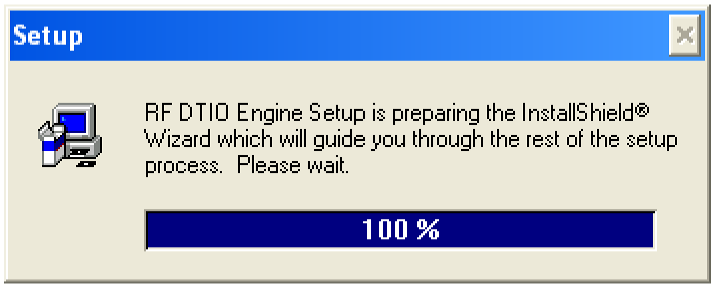
Figure 8: Platform / OS / Software Decision Chart

Installation

Installation of the GSW Directed Terminal I/O Engine software is simple and quick. From Windows NT/XP/VISTA/2000/2003/2008/2012 (R2) perform the following:

Note: Both the GSW RF DTIO x86 and GSW RF DTIO x64 editions are included on the CD when purchased. Both are made available with the same setup program. Simply run the setup program.

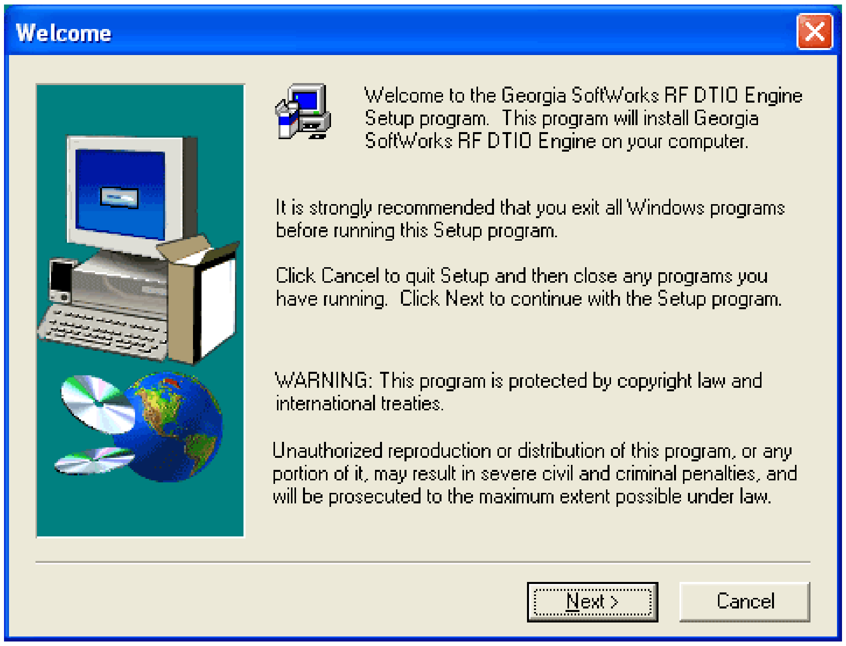
1. Run the setup.exe program.



RF DTIO Engine Setup is preparing the InstallShield® Wizard which will guide you through the rest of the setup process. Please wait.

Figure 9: Initial Setup Screen

2. The Welcome screen of the setup program is displayed and you are reminded and urged to exit all windows programs before continuing. You are also reminded that you must have administrative privileges to install this program. Click **Next.**



Welcome to the Georgia SoftWorks RF DTIO Engine Setup program. This program will install Georgia SoftWorks RF DTIO Engine on your computer.

It is strongly recommended that you exit all Windows programs before running this Setup program.

Click Cancel to quit Setup and then close any programs you have running. Click Next to continue with the Setup program.

WARNING: This program is protected by copyright law and international treaties.

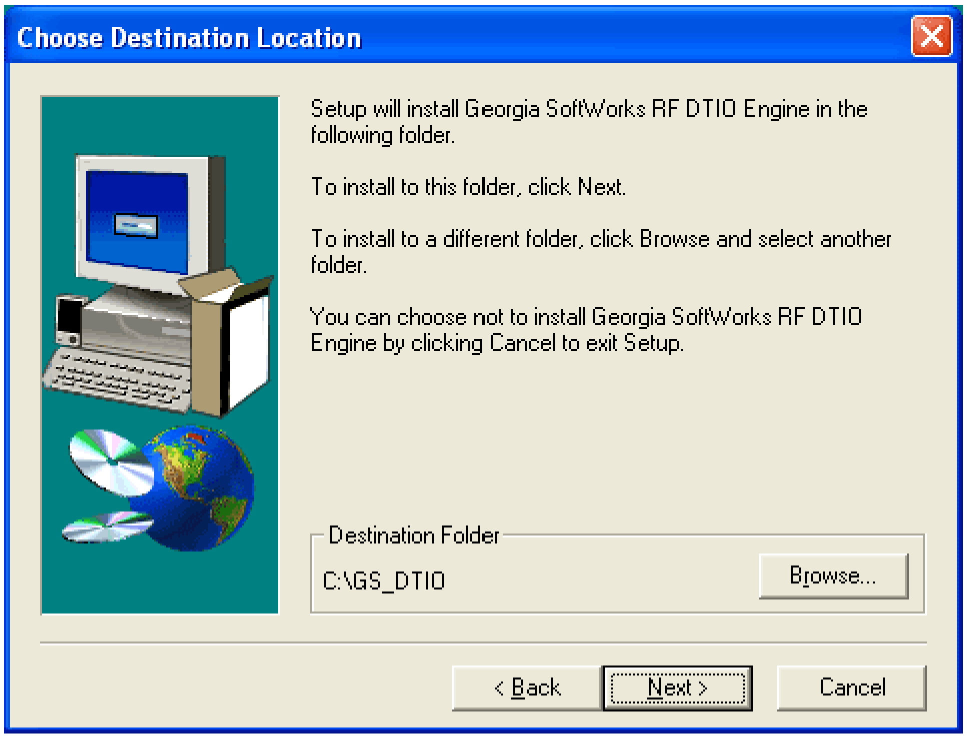
Unauthorized reproduction or distribution of this program, or any portion of it, may result in severe civil and criminal penalties, and will be prosecuted to the maximum extent possible under law.

Figure 10: Installation Welcome Screen

3. A screen is displayed indicating the folder that the GSW Directed Terminal I/O Engine will be installed. The default is:

```
C:\GS_DTIO
```

You may change the installation directory at this time. *Note: Make sure that the users of the Directed Terminal I/O Engine have full access to the installation directory*. Click **Next**.



Choose Destination Location

Setup will install Georgia SoftWorks RF DTIO Engine in the following folder.

To install to this folder, click Next.

To install to a different folder, click Browse and select another folder.

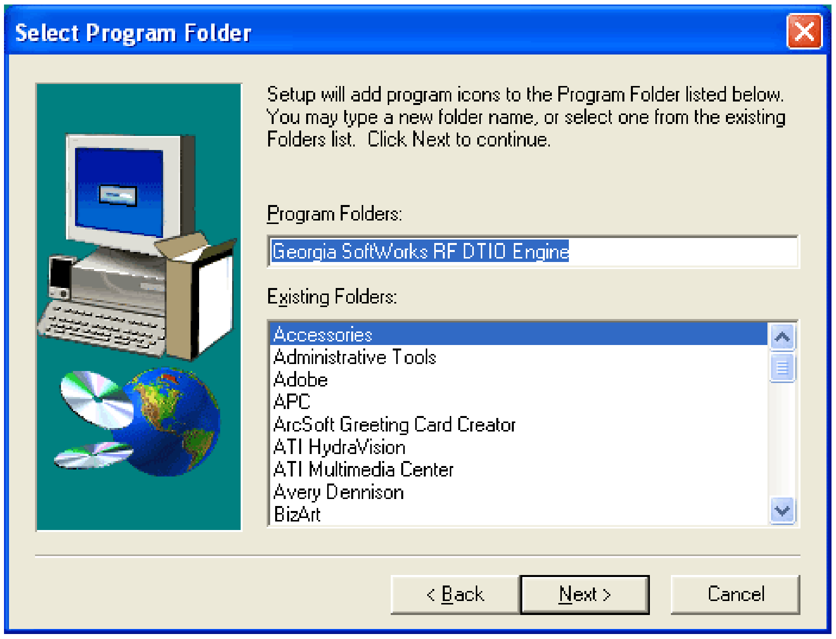
You can choose not to install Georgia SoftWorks RF DTIO Engine by clicking Cancel to exit Setup.

Destination Folder

C:\GS_DTIO

Figure 11: Installation - Choose Destination Folder

4. Select the Program Folder for the Directed Terminal I/O Engine. Click **Next.**



Select Program Folder

Setup will add program icons to the Program Folder listed below. You may type a new folder name, or select one from the existing Folders list. Click Next to continue.

Program Folders:

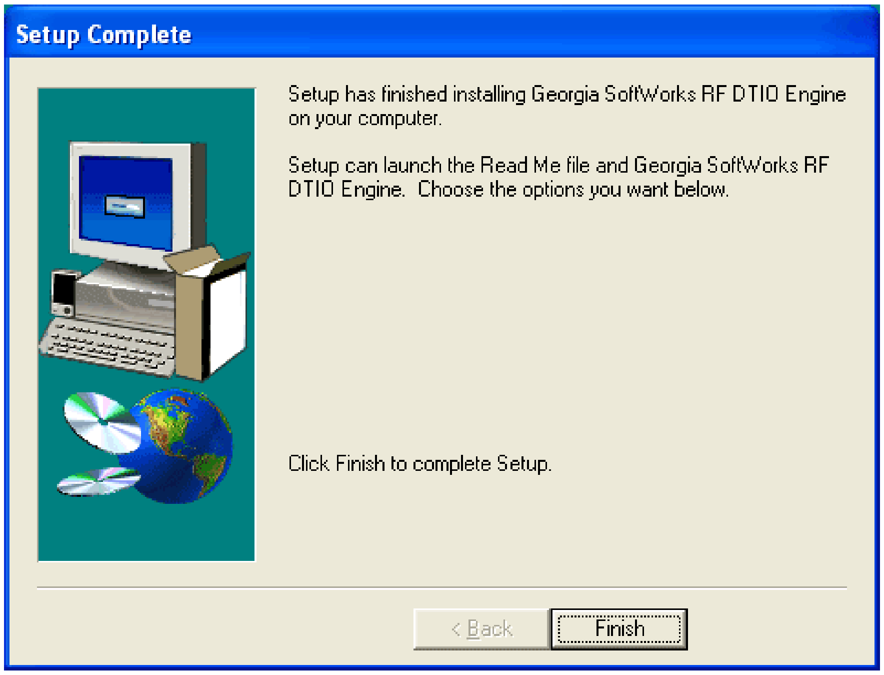
Georgia SoftWorks RF DTIO Engine

Existing Folders:

Accessories
Administrative Tools
Adobe
APC
ArcSoft Greeting Card Creator
ATI HydraVision
ATI Multimedia Center
Avery Dennison
BizArt

Figure 12: Installation - Select Program Folder

5. Now the Setup is complete! *Now its time to register the Directed Terminal I/O Engine*!



Setup has finished installing Georgia SoftWorks RF DTIO Engine on your computer.

Setup can launch the Read Me file and Georgia SoftWorks RF DTIO Engine. Choose the options you want below.

Click Finish to complete Setup.

Figure 13: Installation - Setup Complete

Please view the `readme.txt` file as it may contain late breaking information about the Directed Terminal I/O Engine that has not yet made it into the user guide. Release notes are also contained in the `readme.txt` file.

NOTE: **Successful installation will result in the Registration Program Item showing up under Georgia SoftWorks Directed Terminal I/O Engine.**

Registration

The GSW Directed Terminal I/O Engine is licensed for a single server. The license must be *activated* for the software to operate. To activate the license a valid *Serial Number* is required and is examined periodically by the Directed Terminal I/O Engine software. The Serial Number also allows new versions to be downloaded and installed for the duration of your subscription plan.

Two methods exist to obtain a valid Serial Number.

1. Registration via Floating License (default method)
   The Serial Number is pre-programmed into a specific hardware key that came with your purchase. The hardware key connects to a parallel or USB port on the server. See page 14 for details on registration via the Floating License.

2. Registration via Software Serial Number.
   This method exists for environments that do not support parallel or USB ports. In brief this entails providing GSW with a machine specific Product ID. A Serial Number is generated based on the Product ID. This is usually performed via email, fax or telephone.

Floating License – Overview

The Georgia SoftWorks Floating License provides the flexibility to rapidly move the GSW DTIO from one machine to another. *If you are unable to use the Floating License - skip this section and go to the section on Registration via Software Serial Number on page 21.*

NOTE: When a Directed Terminal IO Pack is purchased (Directed Terminal I/O Engine and GSW Telnet Server), the same physical Floating License will contain a valid Serial Number for both products.

With the Floating License **NO** software registration is required for the DTIO to operate.

Common scenarios where the Floating License is useful include:

- **Laboratory usage in a development or test environment** where the DTIO is required for short periods of time on any particular machine and then moved to a new machine.

- **Backup Servers in a production environment**. Typically multiple RF DTIOs are purchased for backup systems, however with a Floating License the Hardware Key can be quickly moved from the primary machine to the backup without any other registration requirements.

- **Environments where a failed server must be replaced or rebuilt and immediately restored to operation with full DTIO capability**.

The Georgia SoftWorks floating license is a hardware key that can be ordered for a USB Port or a Parallel port.

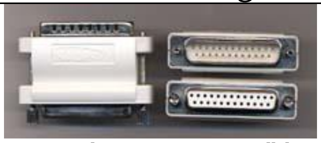
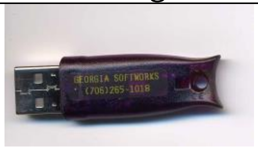
Parallel Port Floating License	USB Floating License
 Figure 14: Floating License – Parallel Port The Parallel Port Floating License is a Pass Through allowing normal function of the port.	 Figure 15: Floating License - USB Port Not attached to a Server
The Parallel Port Floating License connects to a female parallel port on the server and does not impact functionality of the port for other uses. It acts as a pass though allowing normal connections to the other side of the key.	USB LED Lights when Installed

Table 1: Floating Licenses - USB and Parallel

GSW RF DTIO will recognize the presence of the key and activate the software and the proper date for which free version upgrades can be obtained. It does not matter which parallel or USB port on the server the Hardware Key is installed, as all ports will be scanned for the installation of the key.

The Floating License currently is installed using the manufacturer (Aladdin) of the hardware key's setup program. It is described below. The name of the hardware key is HASPHL and you will see it displayed in the setup screens

Floating License – Hardware Key Installation Instructions

Note: If you are using a *USB Floating License on a Windows NT system* run the file `aksnt4usb.exe` prior to the following steps.

1. Install the GSW DTIO software as described on page 10 (if it is not already installed).
2. Copy the files from the Floating License folder (hardkey) on the provided CD to the hard drive on your server.
3. Run the `HASPUserSetup.exe` program and follow the installation instructions.

   You will first see the Aladdin Splash Screen. The Aladdin Splash Screen will display for about 5 seconds.

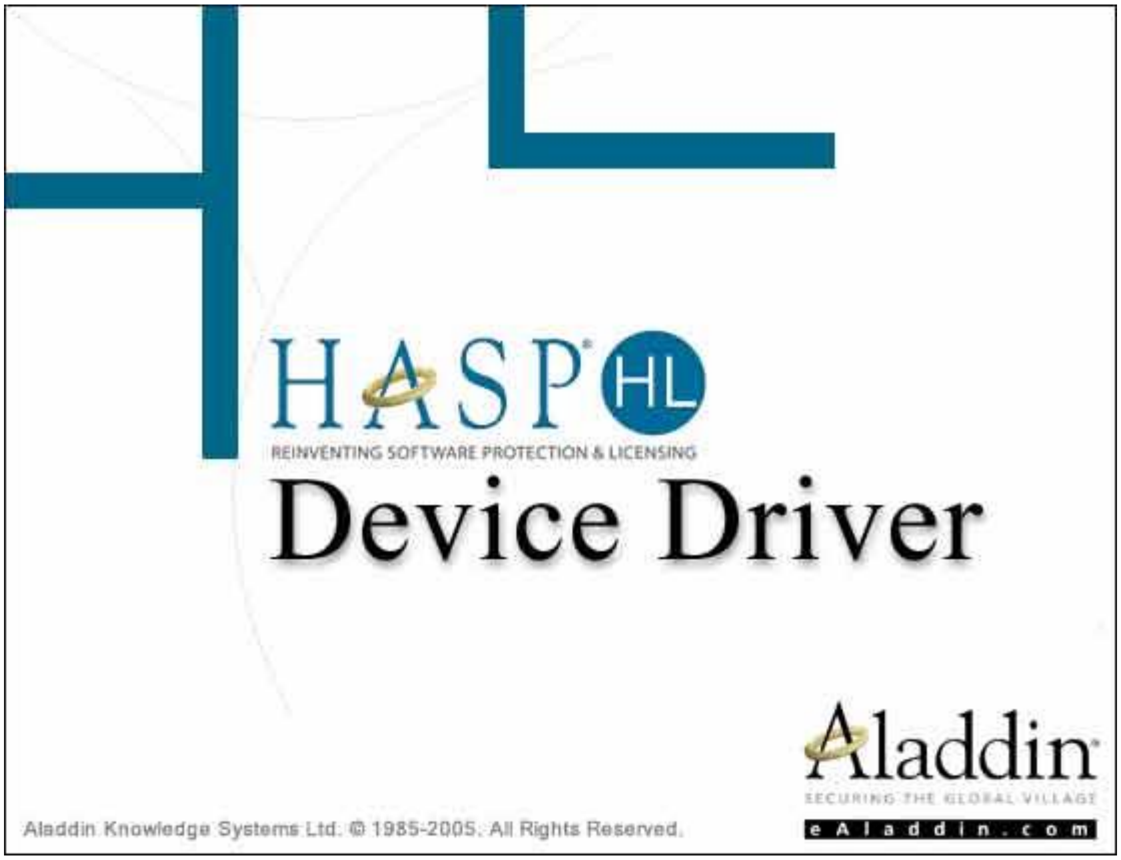
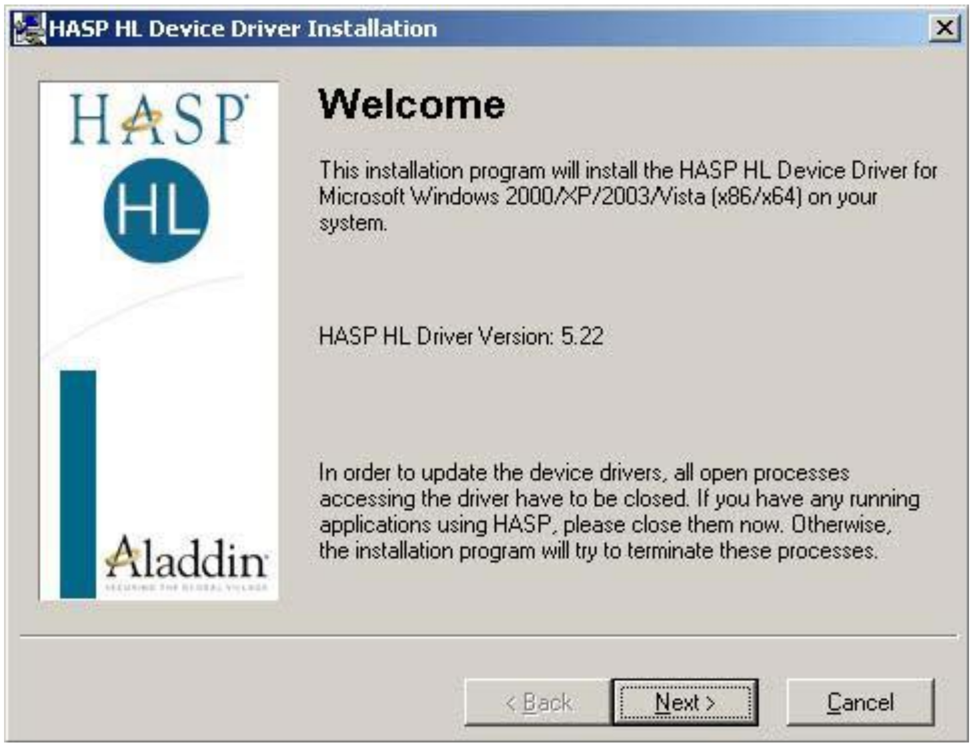

HASP HL
REINVENTING SOFTWARE PROTECTION & LICENSING
Device Driver
Aladdin
SECURING THE GLOBAL VILLAGE
eAladdin.com
Aladdin Knowledge Systems Ltd. © 1985-2005. All Rights Reserved.

Figure 16: Floating License – HW Key Initial Splash Screen

4. The next screen displayed is the Aladdin Welcome Screen.


HASP HL Device Driver Installation

Welcome

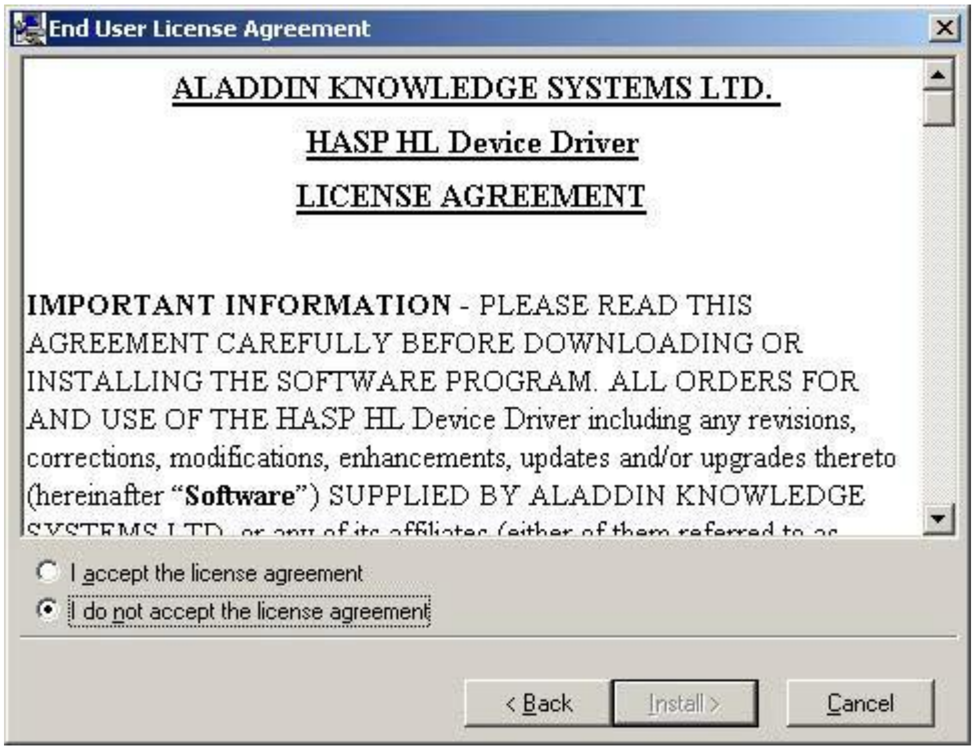
This installation program will install the HASP HL Device Driver for Microsoft Windows 2000/XP/2003/Vista (x86/x64) on your system.

HASP HL Driver Version: 5.22

In order to update the device drivers, all open processes accessing the driver have to be closed. If you have any running applications using HASP, please close them now. Otherwise, the installation program will try to terminate these processes.

Figure 17: Floating License – Welcome Screen

As the dialog indicates, if you have any running application please close them now. **Click Next**.



ALADDIN KNOWLEDGE SYSTEMS LTD.

HASP HL Device Driver

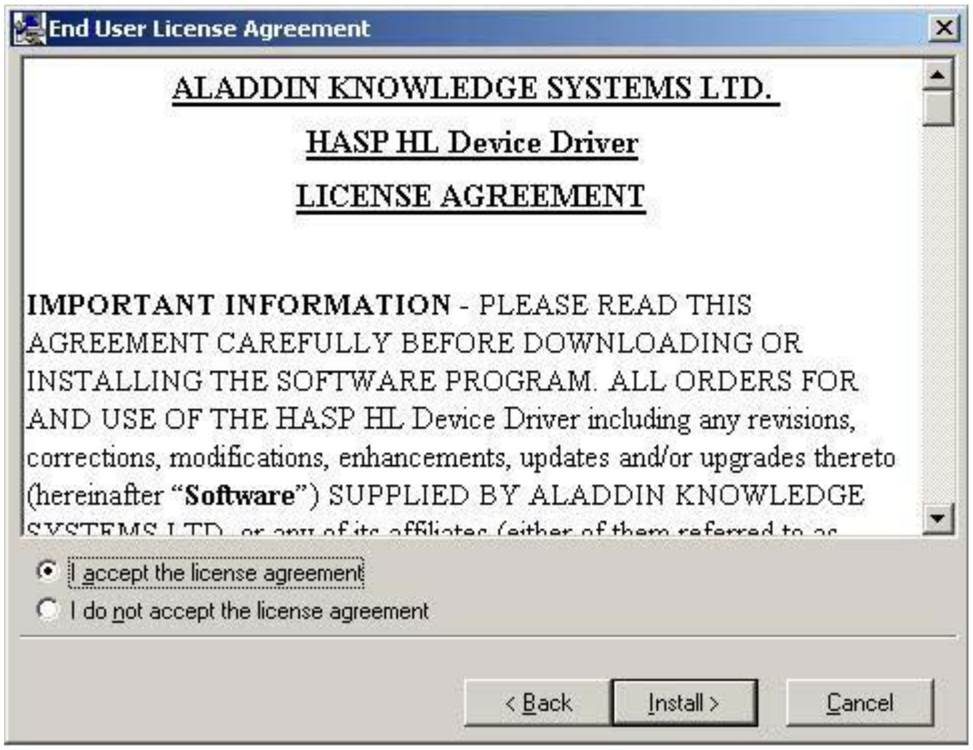
LICENSE AGREEMENT

IMPORTANT INFORMATION - PLEASE READ THIS AGREEMENT CAREFULLY BEFORE DOWNLOADING OR INSTALLING THE SOFTWARE PROGRAM. ALL ORDERS FOR AND USE OF THE HASP HL Device Driver including any revisions, corrections, modifications, enhancements, updates and/or upgrades thereto (hereinafter "**Software**") SUPPLIED BY ALADDIN KNOWLEDGE

( ) I accept the license agreement
(●) I do not accept the license agreement

Figure 18: Floating License - License Agreement

Read the license agreement and select "I accept the license agreement", and then **Click Install**.



ALADDIN KNOWLEDGE SYSTEMS LTD.

HASP HL Device Driver

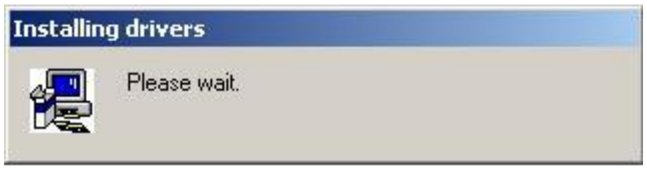
LICENSE AGREEMENT

IMPORTANT INFORMATION - PLEASE READ THIS AGREEMENT CAREFULLY BEFORE DOWNLOADING OR INSTALLING THE SOFTWARE PROGRAM. ALL ORDERS FOR AND USE OF THE HASP HL Device Driver including any revisions, corrections, modifications, enhancements, updates and/or upgrades thereto (hereinafter "**Software**") SUPPLIED BY ALADDIN KNOWLEDGE

(●) I accept the license agreement
( ) I do not accept the license agreement

Figure 19: Floating License - Accept License Agreement

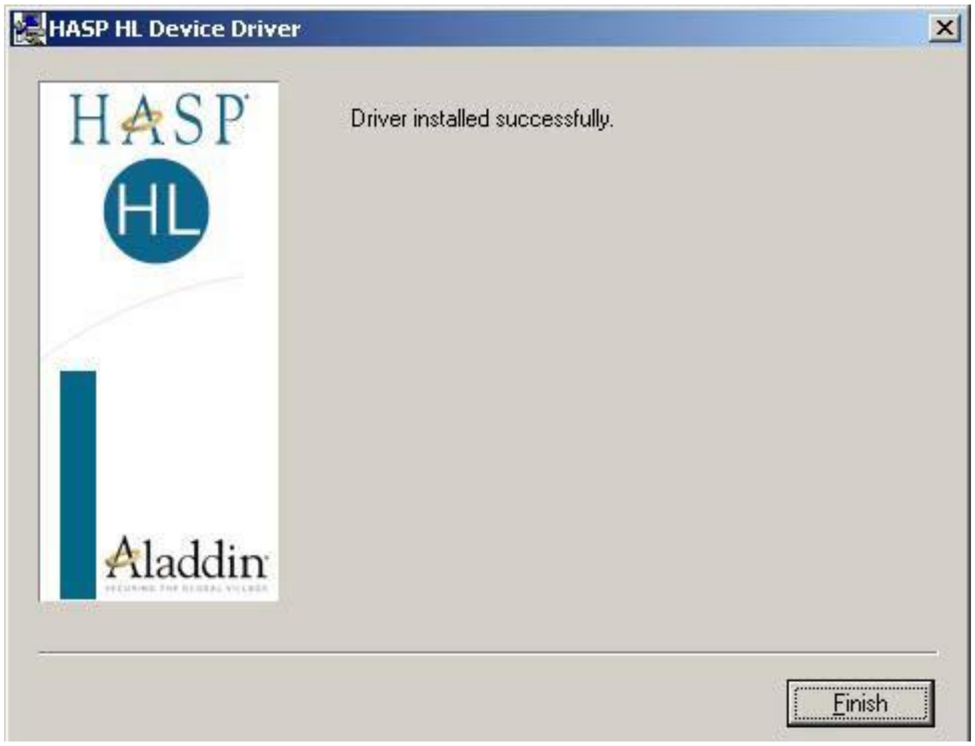
5. An installation status progress meter is quickly displayed and when the status gathered is completed the screen below is displayed.



Please wait.

Figure 20: Floating License - HW Key - Installation Status

6. When the installation of the Aladdin Hasp Device driver is complete the screen below is displayed. **Click Finish**.



Driver installed successfully.

Figure 21: Floating License Drivers Successful Installation

7. Plug the hardware key onto the parallel or USB port on the server.

   NOTE: On some systems you may have to reboot the server after installation. If the Floating License is not recognized (by the GSW DTIO) after installing the driver, please reboot the server.

Uninstall Floating License – (Hardware Key)

In the event that you need to uninstall the Floating License (Aladdin HaspHL) please use the Windows Control Panel Add/Remove Programs administrative utilities.

NOTE: Removing or uninstalling the Floating License will disable the GSW DTIO Software.

Registration via Software Serial Number

To run the GSW Directed Terminal I/O Engine you must first register the software. (*This registration is **NOT** required if you installed the Floating License. Page 14*) Registration via Software Serial Number entails just a few steps that involve obtaining the Product ID and providing this Identification to Georgia SoftWorks so a Serial Number can be generated. - **NOTE:** Read System Signature chapter at the end of manual.

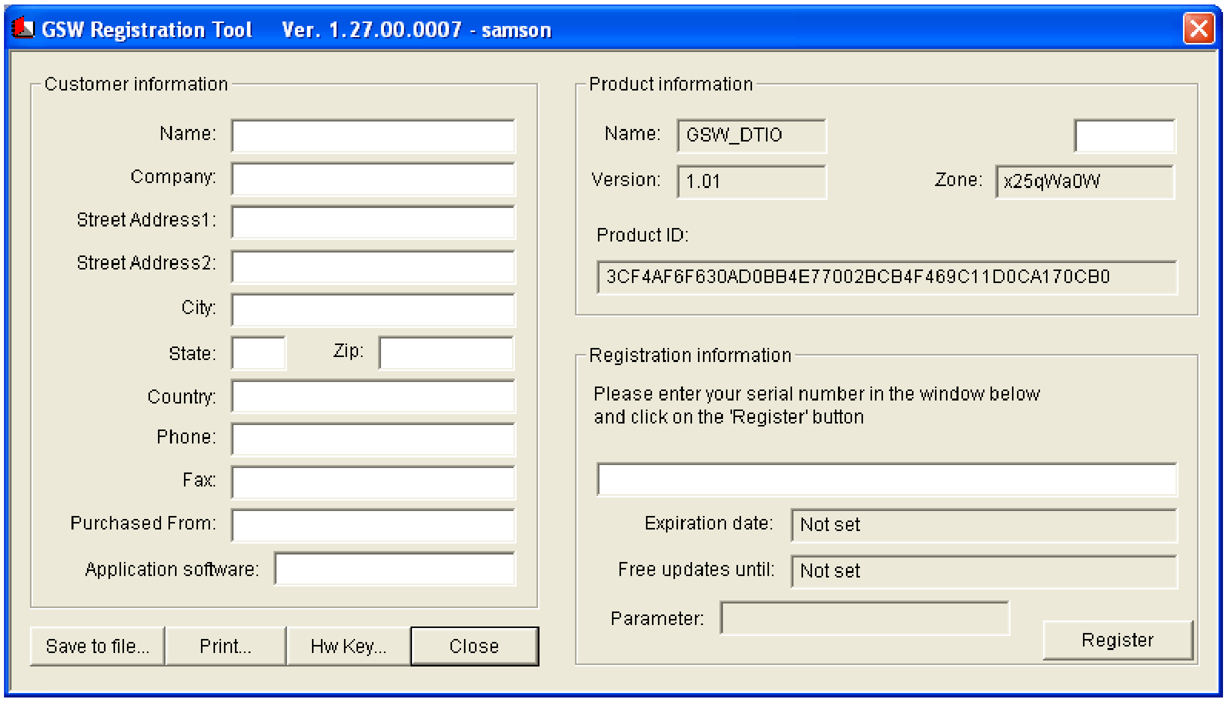
How to Register the Software

To run the registration software -

- Select the *Start* button on the task bar; select *Programs*, then *Georgia SoftWorks Directed Terminal I/O Engine* and then *Registration*.

Note: The Product Information *Name* and *Version* must contain valid data or it will not generate a correct Product ID.

The registration screen is displayed. The Registration software automatically fills in the Product Information fields as shown in the figure below.



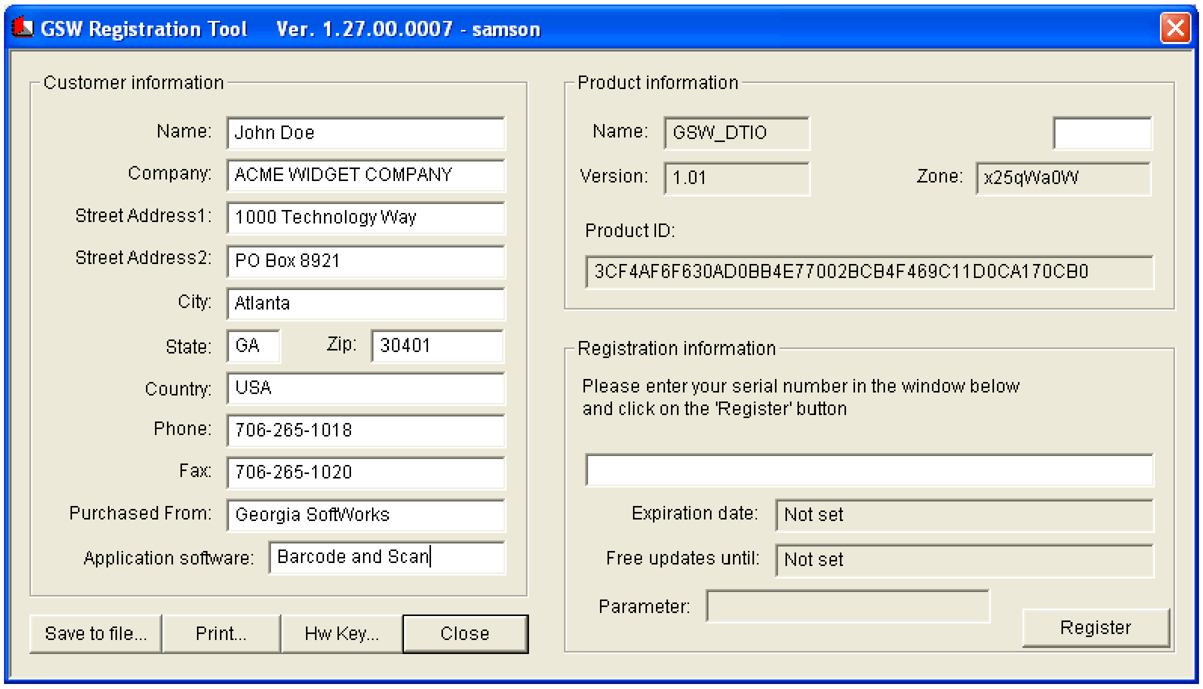
Name: GSW_DTIO
Version: 1.01
Zone: x25qWa0W
Product ID: 3CF4AF6F630AD0BB4E77002BCB4F469C11D0CA170CB0

Please enter your serial number in the window below and click on the 'Register' button

Expiration date: Not set
Free updates until: Not set
Parameter:

Figure 22: Registration with Serial Number - Initial Screen

1. Please complete the *Customer Information*, the *Purchased From* and the *Application Software* fields in the Registration Screen. The Purchased From and the Application software fields are very important when requesting support. In our example below it indicates that the GSW RF DTIO was obtained directly from Georgia SoftWorks and the application that the GSW RF DTIO is being used with is the Barcode and Scan software package.



Customer information

Name:	John Doe
Company:	ACME WIDGET COMPANY
Street Address1:	1000 Technology Way
Street Address2:	PO Box 8921
City:	Atlanta
State:	GA
Zip:	30401
Country:	USA
Phone:	706-265-1018
Fax:	706-265-1020
Purchased From:	Georgia SoftWorks
Application software:	Barcode and Scan

Product information

Name:	GSW_DTIO
Version:	1.01
Zone:	x25qWa0W
Product ID:	3CF4AF6F630AD0BB4E77002BCB4F469C11D0CA170CB0

Registration information

Please enter your serial number in the window below and click on the 'Register' button

Expiration date:	Not set
Free updates until:	Not set
Parameter:	

Figure 23: Registration - User Information

2. The registration information must be provided to Georgia SoftWorks to obtain the Serial Number. Several methods are available for your convenience.

   a. Save the information to a file and email it to Georgia SoftWorks - *Preferred method*. Please save (using the **Save to file** button on the registration screen) this information to a file and email to Georgia SoftWorks registration@georgiasoftworks.com

   OR

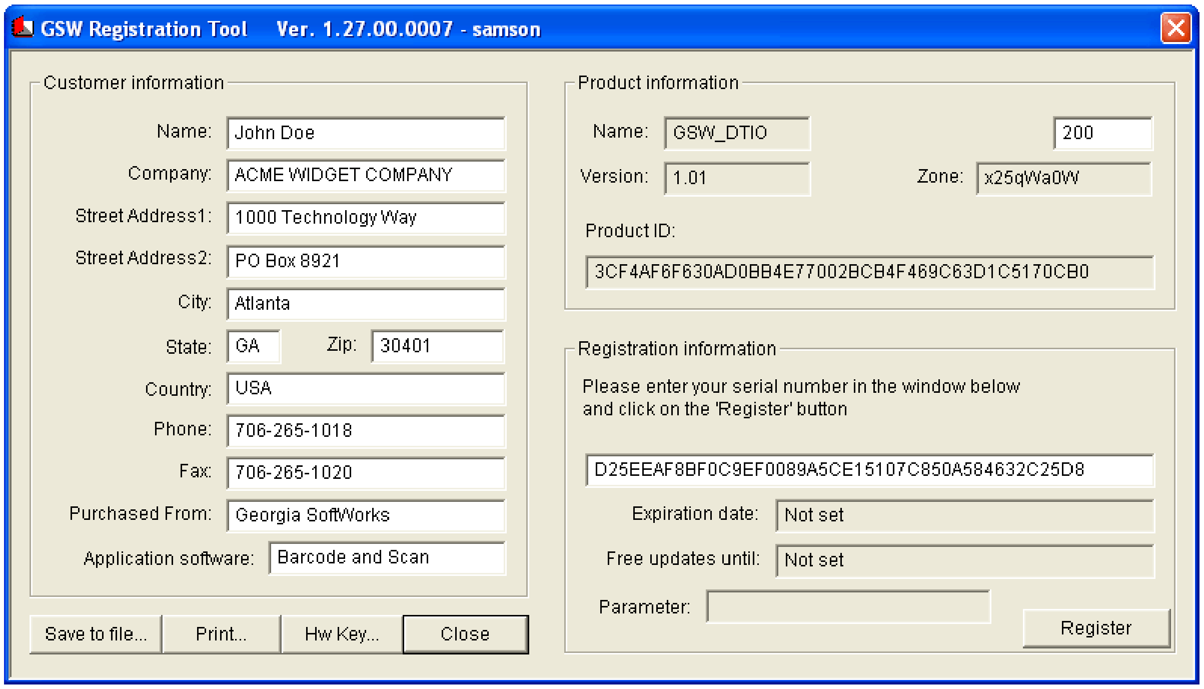
   b. Print the information and Fax it to Georgia SoftWorks. Please print (using the **Print** button on the registration screen) this information and fax to Georgia SoftWorks - 706.265.1020

   OR

   c. Call GSW at 706-265-1018 to obtain verbal registration.

Once Georgia SoftWorks receives the information, we can generate a Serial Number on demand. We will reply back via Fax or email. You may close the registration program at this time.

3. When the Serial Number is provided Run the Registration Program again and enter the Serial Number. The easiest method to get the serial number is to highlight the returned Serial Number and copy (ctrl-c). Then position the mouse in the Serial Number field in the Registration Information box and paste (ctrl-v).



Customer information

Name:	John Doe
Company:	ACME WIDGET COMPANY
Street Address1:	1000 Technology Way
Street Address2:	PO Box 8921
City:	Atlanta
State:	GA
Zip:	30401
Country:	USA
Phone:	706-265-1018
Fax:	706-265-1020
Purchased From:	Georgia SoftWorks
Application software:	Barcode and Scan

Product information

Name:	GSW_DTIO
	200
Version:	1.01
Zone:	x25qWa0W
Product ID:	3CF4AF6F630AD0BB4E77002BCB4F469C63D1C5170CB0

Registration information

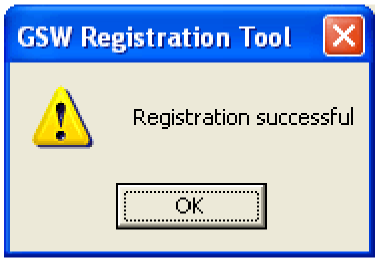
Please enter your serial number in the window below and click on the 'Register' button

D25EEAF8BF0C9EF0089A5CE15107C850A584632C25D8

Expiration date:	Not set
Free updates until:	Not set
Parameter:	

Figure 24: Registration - Serial Number Applied

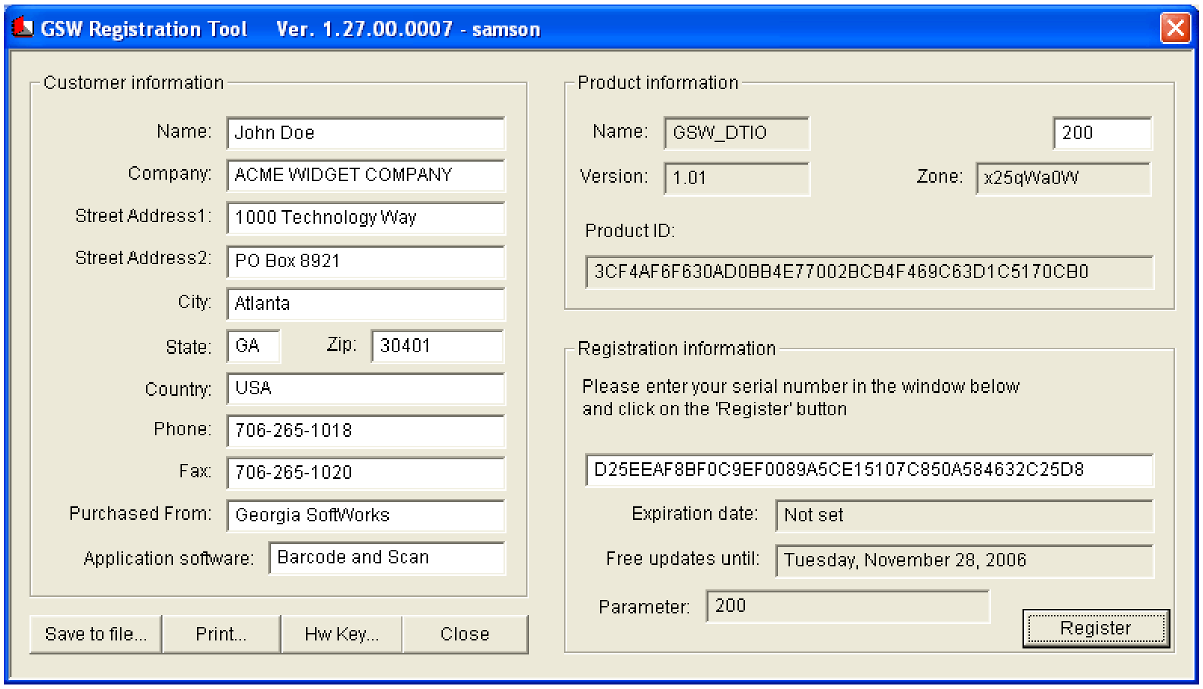
4. **Click Register**.



Registration successful

Figure 25: Registration Successful Screen

Now the software is registered. You may now run the Georgia SoftWorks Directed Terminal I/O Engine. Note that you will be able to obtain Free Updates until the date specified.



Customer information

Name:	John Doe
Company:	ACME WIDGET COMPANY
Street Address1:	1000 Technology Way
Street Address2:	PO Box 8921
City:	Atlanta
State:	GA
Zip:	30401
Country:	USA
Phone:	706-265-1018
Fax:	706-265-1020
Purchased From:	Georgia SoftWorks
Application software:	Barcode and Scan

Product information

Name:	GSW_DTIO
	200
Version:	1.01
Zone:	x25qWa0W
Product ID:	3CF4AF6F630AD0BB4E77002BCB4F469C63D1C5170CB0

Registration information

Please enter your serial number in the window below and click on the 'Register' button

D25EEAF8BF0C9EF0089A5CE15107C850A584632C25D8

Expiration date:	Not set
Free updates until:	Tuesday, November 28, 2006
Parameter:	200

Figure 26: GSW DTIO is now registered

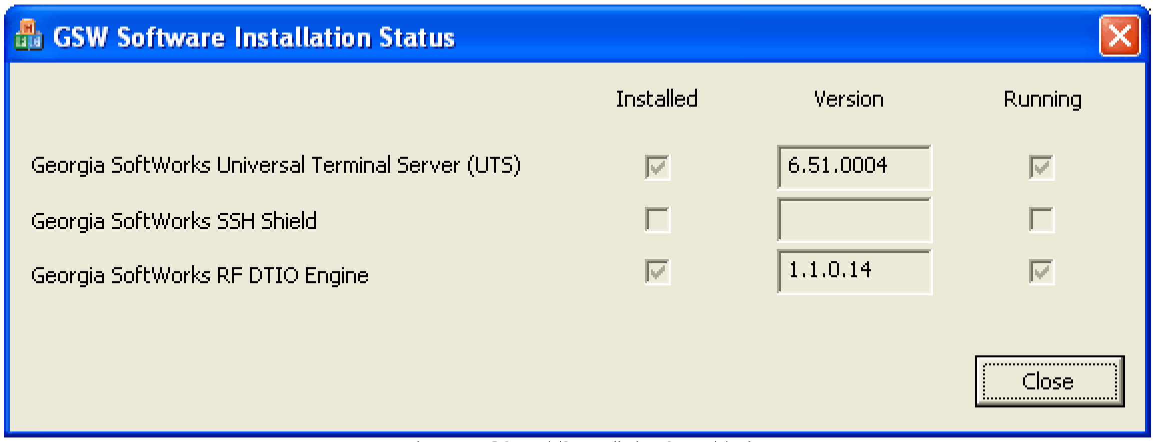
IMPORTANT: READ SYSTEM SIGNATURE CHAPTER AT END OF MANUAL (PAGE 36).

Running RF DTIO

Once the GSW Directed Terminal I/O Engine is installed and registered it is ready to run.

RF DTIO Installation Status

You can verify that DTIO is properly installed by viewing the GSW UTS Installation Status Screen. The Installed checkbox should be checked once installed. You can easily view the version number of the GSW RF DTIO from this screen. The Running checkbox will be checked when the RF DTIO is running.



GSW Software Installation Status

	Installed	Version	Running
Georgia SoftWorks Universal Terminal Server (UTS)	✓	6.51.0004	✓
Georgia SoftWorks SSH Shield			
Georgia SoftWorks RF DTIO Engine	✓	1.1.0.14	✓

Figure 27: GSW UTS Installation Status Tool

Run RF DTIO

RF_DTIO must be run under a Telnet or SSH2 session. RF DTIO is a command line program that is usually entered in your GSW UTS logon scripts[2]. Please review Logon Scripting in the GSW UTS Users Guide. A command line argument is used to specify the application that RF DTIO will "hook".

Command: `GSWDTIO or GSWDTIOx64`

Description: Command used to initiate RF DTIO operation for an application.

Syntax 1: `GSWDTIO APPLICATION_WITH_ITS_ARGUMENTS` for x86 or x64 platforms

Syntax 2: `GSWDTIO`**`x64`**` APPLICATION_WITH_ITS_ARGUMENTS` for x64 platforms

The RF DTIO command line is the application and its command line arguments. Depending on your application it may or may not take additional command line arguments. Any application arguments are simply added to the command line in the format required by the application.

[2] Of course it can be run from the command line directly too.

Please make sure that the each path for the GSWDTIO and the application is specified. RF DTIO default installation folder is C:\GS_DTIO and this is where the `gswdtio` program executable resides.

Example – Launch RF DTIO for the GS_ADMIN utility.

The GSW UTS GS_ADMIN utility is an easy example for using RF DTIO. We gather the necessary path and executables.

1. The path for GS_ADMIN utility default installation folder is:

   ```
   C:\GS_UTS
   ```

   The executable is `GS_Admin.exe`

2. The path for RF DTIO Engine default installation folder is:

   ```
   C:\GS_DTIO
   ```

   The executable is `gswdtio.exe`

3. In our login script we can enter the line:

   ```
   C:\GS_DTIO\gswdtio C:\GS_UTS\GS_Admin.exe
   ```

When the user's logon script[3] gets invoked, the RF DTIO will launch the application GS_Admin.exe.

If you are not using Global Logon scripting, then you should add the line to launch RF DTIO for each user's specific logon script that you want to use RF DTIO.

If you are using the RF DTIO x64 then the line listed above would simply be:

```
C:\GS_DTIO\gswdtiox64 C:\GS_UTS\GS_Admin.exe
```

[3] Please see the GSW UTS User Manual for details on Logon Scripting.

Configuration

Once the GSW Directed Terminal I/O Engine is installed and registered it is ready to run. The default configuration is fine for most systems.

There is a configuration utility provided for modification of the RF DTIO default values if required. The RF DTIO configuration provides an easy to use interface for setting values as well as providing a Super Fine Tuning capability to adjust precisely certain aspects so as to bring to the highest level of performance. In many cases Super Fine Tuning settings will need to be empirically derived with the help of profiling.

Two methods exist for configuration of the GSW DTIO Engine.

- Graphical Interface (Recommended Interface)
- Text based *ini* file configuration (for advanced SSH2/Telnet users)

Changes made using either method is reflected in both the *ini* file and the graphical interface. There is no need to worry about mismatched configurations. The GSW RF DTIO configuration is backup friendly as all configuration is contained within the *ini* file which is easily backed up.

The text based *ini* file configuration is available locally or when you are remotely connected via SSH2/Telnet. The location of the *ini* configuration is in the installation folder for the DTIO and the file has the name `gswdtio.ini`. The `gswdtio.ini` has the historical Windows *ini* format. For example, the contents of the file will look something like the following starting in column 1.

```
[Profiling]
APICallCount =0
APIByteCount =0
LoadLibraryCalls =0
APIHooking =0
KeyboardReadAPIs =0
ModifyScreenAPIs =0

[Timing]
TriggerDelay = 100
TriggerOnReadConsoleInputA = 0
TriggerOnReadConsoleInputW = 0
TriggerOnReadConsoleA = 0
TriggerOnReadConsoleW = 0
TriggerOnReadFile = 0

[Bell]
EnableBellProcessing =1
LogBellProcessing =0

[Trace]
EnableTrace =0
ShowTriggerEvents =0
```

Figure 28: RF DTIO Configuration - ini file format

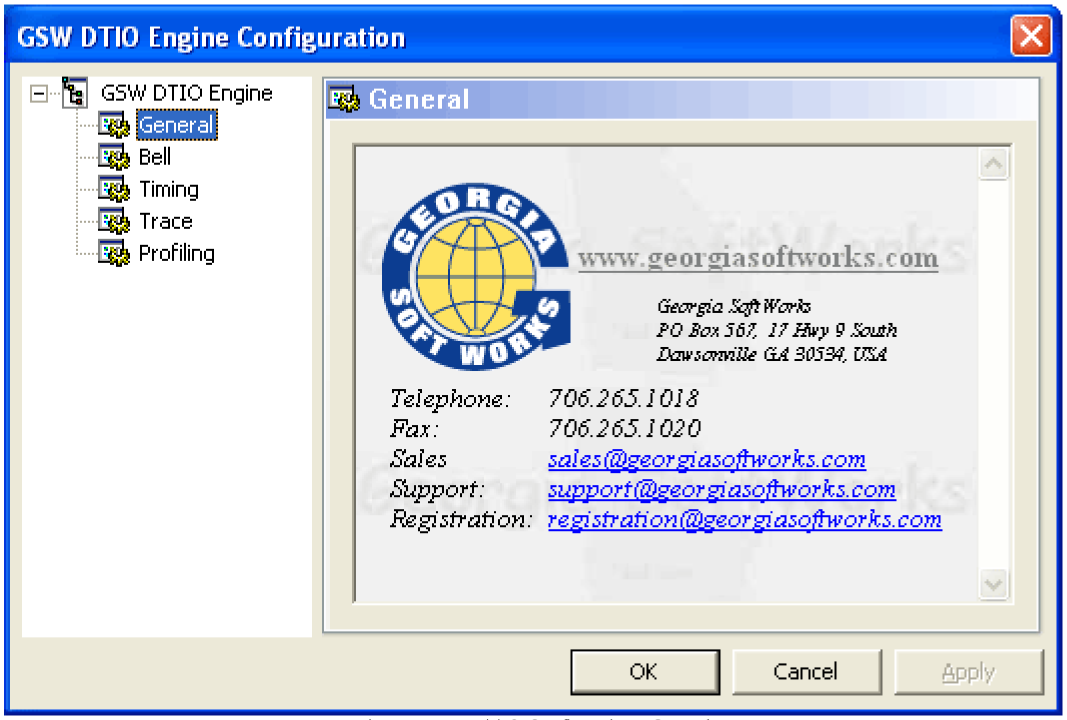
For items that have enabled/disabled settings –

```
0 = Disabled
1 = Enabled.
```

Descriptions of each of the items in the diagram above will be described in the graphical interface section.

General Configuration Screen

The graphical configuration interface is available when you are local to the server/workstation that the GSW DTIO is installed. A navigational tree provides access to the different resource and configuration screens.



www.georgiasoftworks.com

Georgia Soft Works
PO Box 567, 17 Hwy 9 South
Dawsonville GA 30534, USA

Telephone:	706.265.1018
Fax:	706.265.1020
Sales	sales@georgiasoftworks.com
Support:	support@georgiasoftworks.com
Registration:	registration@georgiasoftworks.com

Figure 29: RF DTIO Configuration - General

The navigational sections are grouped as follows:

- **General** – GSW contact information. Web links, email addresses and telephone numbers.
- **Bell** – Bell Handling configuration.
- **Timing** – Sets timing of screen scans and enables trigger events for screen scans. Note: Settings should only be modified in a controlled testing environment.
- **Trace** – Enables tracing and logging. NOTE: This should be done only in a controlled testing environment as large amounts of log information may be created and negatively impact system performance.
- **Profiling** – Provides statistics on usage of different hooked API's. This can be used to help determine the Super Fine Tuning settings. Note: Settings should only be modified in a controlled testing environment.

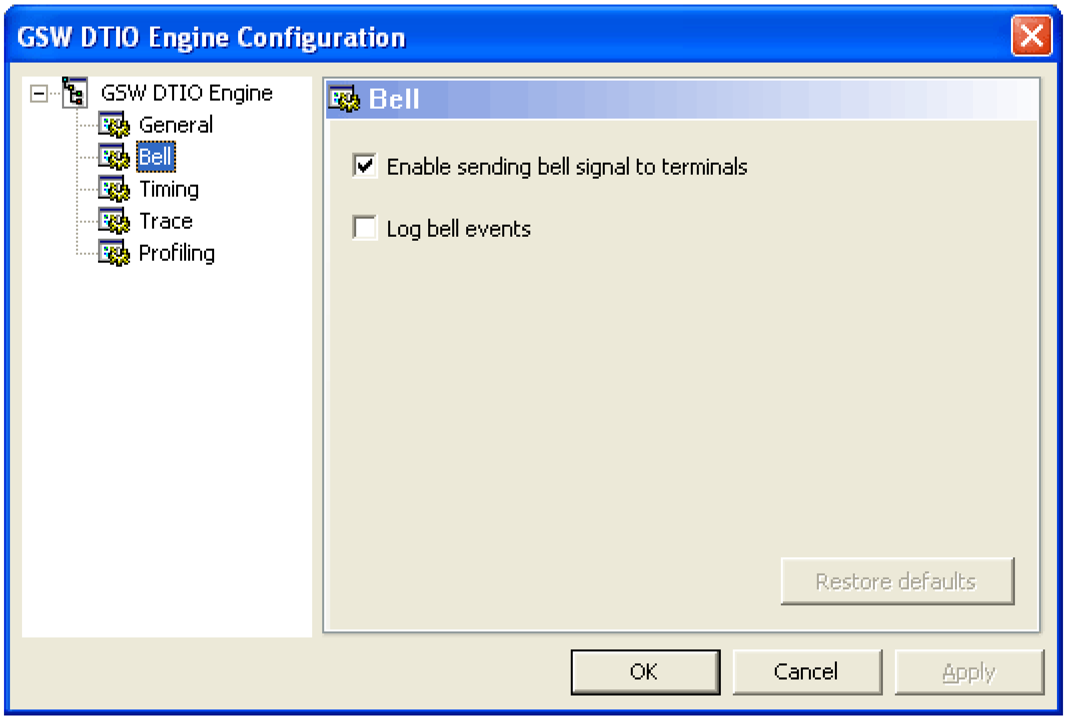
Configuration Categories

The table below provides a view of the configuration parameters within each grouping along with a brief description. When the options available for many of the configuration parameters are listed in parenthesis, the default setting is boldfaced.

	Configuration Parameter	Brief Description
	Bell Processing	
1	Enable Sending Bell Signal to Terminal	(***Enable***/Disable) Sends the Bell signal to the Terminal
2	Log Bell Events	(Enabled/***Disable***) Enter a log entry each time a bell is sent to the terminal
	Timing	
1	TriggerDelay	**(**100-1000ms**, Default**: **100ms** (1/10 of a sec)) Time to delay screen scan after occurrence of a screen write API. Units: Milliseconds;
2	TriggerOnReadConsoleInputA	(Enable/***Disable***) Trigger immediate screen scan after ReadConsoleInputA is completed.
3	TriggerOnReadConsoleInputW	(Enable/***Disable***) Trigger immediate screen scan after ReadConsoleInputW is completed.
4	TriggerOnReadConsoleA	(Enable/***Disable***) Trigger immediate screen scan after ReadConsoleA is completed.
5	TriggerOnReadConsoleW	(Enable/***Disable***) Trigger immediate screen scan after ReadConsoleW is completed.
6	TriggerOnReadFile	(Enable/***Disable***) Trigger immediate screen scan after ReadFile is completed.
	Trace	
	Enable Trace Messages	
	Log Trigger Events	
	Profiling	
1	APICallCount	(Enable/***Disable***)Provides statistics on Terminal I/O API call counts
2	APIByteCount	(Enable/***Disable***)Provides statistics on Terminal I/O API Byte transfer size counts
3	LoadLibraryCalls	(Enable/***Disable***) Shows which libraries are loaded.
4	APIHooking	(Enable/***Disable***) Logs/Displays the APIs as they are hooked by GSW
5	Keyboard Read APIs	(Enable/***Disable***) Logs/Displays the Keyboard Read APIs that are hooked
6	Screen Write APIs	(Enable/***Disable***) Logs/Displays the Screen Write APIs that are hooked

Table 2: RF DTIO Engine Configuration Parameters

Bell Processing



☑ Enable sending bell signal to terminals

☐ Log bell events

Figure 30: Configuration of Bell Processing

Enable / Disable sending the bell signal to the terminals. Default: ENABLED

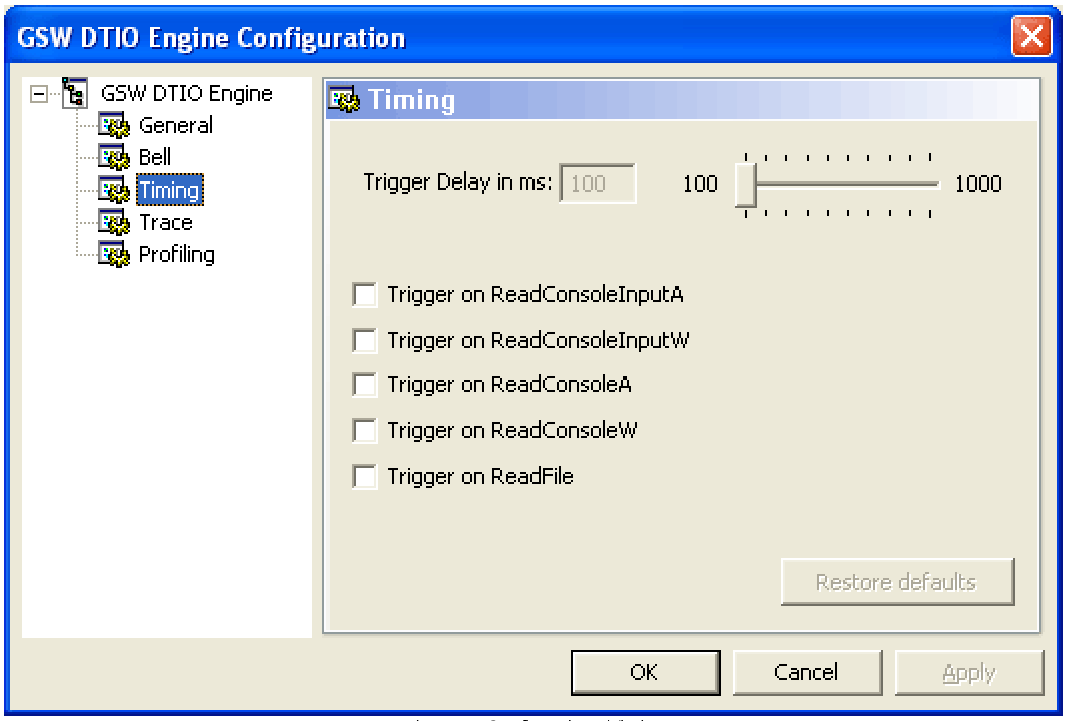
When an application writes the character sequence to sound the bell, by default (without using GSW RF DTIO) it sounds on the local system. When using SSH2/Telnet the desired result is to have the bell sound on the remote device and not on the local device. The GSW RF DTIO Engine provides this capability by recognizing when the application sounds the bell. Instead of sounding the bell on the local system the GSW RF DTIO intercepts the bell and sends it to the proper device or terminal.

Log bell events Default: DISABLED

The system administration may be interested in knowing when the bell sound events were sent to the devices. Enabling the Log bell events will create an entry in the log file for each occurrence.

Timing Configuration

Timing is an advanced feature that is used by sophisticated users and GSW Engineers to provide custom optimization and fine-tuning for your system.



Trigger Delay in ms: 100 (slider 100 – 1000)

- [ ] Trigger on ReadConsoleInputA
- [ ] Trigger on ReadConsoleInputW
- [ ] Trigger on ReadConsoleA
- [ ] Trigger on ReadConsoleW
- [ ] Trigger on ReadFile

Figure 31: Configuration - Timing

Trigger Delay ***Default: 100ms***

This is the Time to delay the screen scan after the occurrence of a screen write. Units are Milliseconds – Default 100 ms.

The trigger delay timer is set because in most cases, applications will write several times before the screen is complete. The timer prevents thrashing by reading too soon and having to read again and again.

If screen is changing by itself without keyboard input (Such as a “Please Wait – Processing” message) then the trigger delay time of 100ms will be incurred when set to the default value.

Trigger on ReadConsoleInputA ***Default: Disabled***

On completion of `ReadConsoleInputA` ,Cancel Trigger Delay timer (if pending) and perform screen update immediately.

Trigger on ReadConsoleInputW ***Default: Disabled***

On completion of `ReadConsoleInputW`, Cancel Trigger Delay timer (if pending) and perform screen update immediately.

Trigger on ReadConsoleA ***Default: Disabled***

On completion of `ReadConsoleA`, Cancel Trigger Delay timer (if pending) and perform screen update immediately.

Trigger on ReadConsoleW ***Default: Disabled***

On completion of `ReadConsoleW`, Cancel Trigger Delay timer (if pending) and perform screen update immediately.

Trigger on ReadFile ***Default: Disabled***

On completion of `ReadFile`, Cancel Trigger Delay timer (if pending) and perform screen update immediately.

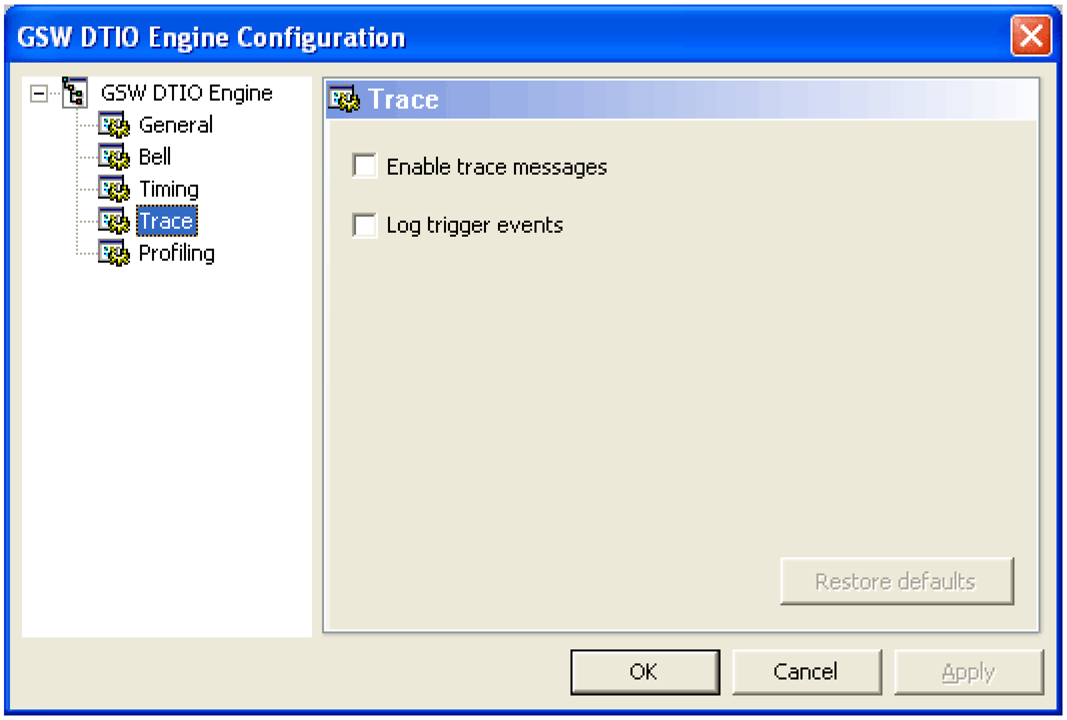
For those with EE background, the timing logic of DTIO can be easily compared to the classic TTL 74LS123 retriggerable monostable multivibrator (One-Shot). The detection of a screen output API call serves as the trigger and retrigger input. The duration of the pulse is controlled by TriggerDelay parameter. The OnConsoleInputA and others are OR-ed and serve as optional CLR for the pulse. The screen update action is triggered on the high-to-low transition of the Q output.

In layman’s terms, the mechanism of DTIO's performance improvement is based in part on the fact the screen output APIs called by applications are often grouped and come in bursts making it inefficient to perform the client screen update immediately following the application's screen API call. Instead of immediate screen update DTIO starts a timer and waits for possible additional screen updates before updating client's screen. When the timer expires the client's screen update will occur. However if another screen update is performed by the application then the timer is restarted in anticipation of additional screen API calls.

In result the screen update will occur only once, TriggerDelay milliseconds after the last call to screen APIs belonging to one logical screen change performed by the application. This assumes no screen API calls are spaced apart more then TriggerDelay milliseconds.

Trace Configuration

The logging of Trace messages and Trigger Events can be `enabled/disabled`. Please only use in a controlled testing environment as this option will impact performance.



GSW DTIO Engine Configuration

- GSW DTIO Engine
  - General
  - Bell
  - Timing
  - Trace
  - Profiling

Trace

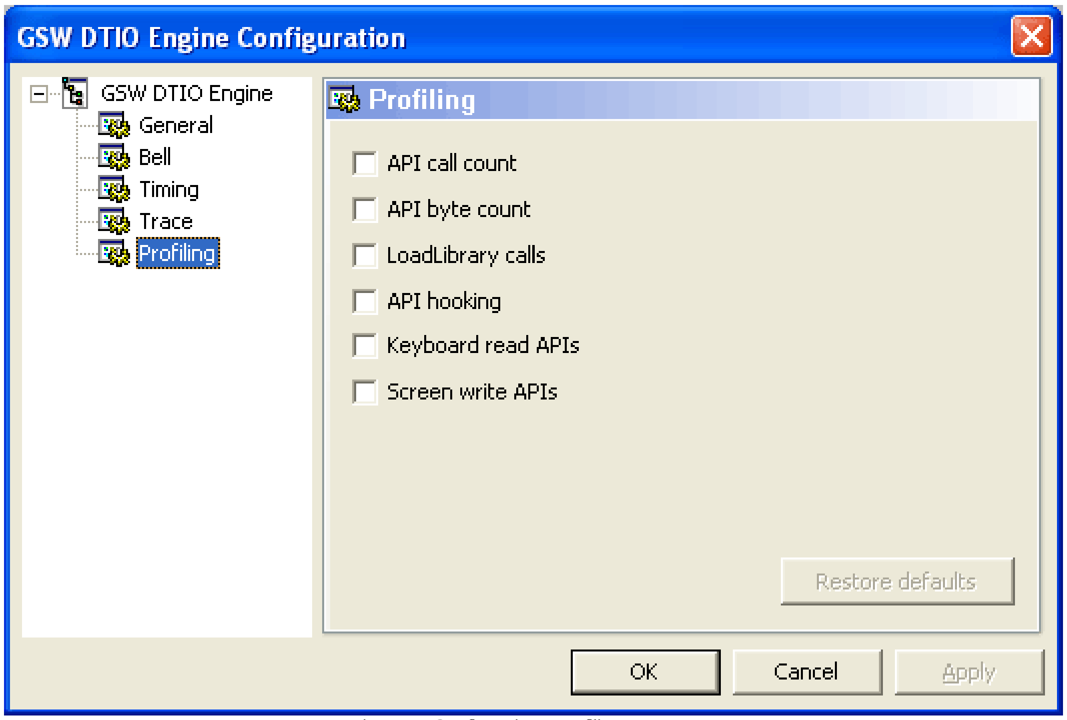
☐ Enable trace messages

☐ Log trigger events

Figure 32: Configuration - Trace Messages and Events

Profiling Configuration

Profiling is an advanced feature that is used by GSW Engineers to gather highly specific information about your environment in order to provide custom optimization and fine-tuning for your system.



- [ ] API call count
- [ ] API byte count
- [ ] LoadLibrary calls
- [ ] API hooking
- [ ] Keyboard read APIs
- [ ] Screen write APIs

Figure 33: Configuration - Profiling

APICallCount ***Default: DISABLED***

This enables gathering call count statistics for each API hooked.

APIByteCount ***Default: DISABLED***

This enables gathering statistics about the number of bytes that each API has transferred (Only for APIs which actually transfer any data).

LoadLibraryCalls ***Default: DISABLED***

When enabled, the names of the libraries loaded by the application are entered into the log.

API Hooking ***Default: DISABLED***

When enabled, shows what API's are "hooked" by the GSW DTIO Engine.

Keyboard Read APIs ***Default: DISABLED***

When enabled, profiles Keyboard Read APIs as they are called by the application.

Screen Write APIs ***Default: DISABLED***

When enabled, profiles Screen Write API's as they are called by the application.

MODE CON

The console command

```
mode con: lines=xx cols=yy
```

where `xx` is the number of lines, and `yy` is the number of columns in the display.

is **very** important to include in your Logon Script for the **GSW UTS Server** for maximum performance. This ensures that the GSW SSH2/Telnet Server scans the exact number of rows and columns.

RF DTIO Pack (Directed Terminal I/O Engine & GSW UTS)

The RF DTIO Pack is when the GSW Directed Terminal I/O Engine and the GSW UTS Server are packaged together rather than being sold individually. The GSW Directed Terminal I/O Engine and the GSW UTS Server will be installed individually and each will have its own User Guide and Software CD.

NOTE: The GSW Directed Terminal I/O Engine and the GSW UTS Server will share the same physical Floating License device when purchased as a RF DTIO Pack.

GSW RF Directed Terminal IO Subscription

The GSW Subscription plan provides access to the most current versions of the software as well as priority support.

In general, Georgia SoftWorks releases new versions as soon as new features are ready rather than waiting for scheduled quarterly or annual dates. Due to our development and release generation methods and JIT User Manual production we can release software on a much more frequent basis than other organizations. As soon as features or defect resolutions are Alpha and Beta tested we generate a release. This provides our customers with features much quicker than the "*grouping*" or "schedule" method used by other companies.

The GSW RF DTIO Engine (and RF DTIO Pack) Subscription plan provides access to free version upgrades for the duration of the Subscription. The duration is either 1, 2 or 3 years. This is good as you can obtain new versions of the software at your convenience obtaining all new features and defect resolutions.

NOTE: New versions can be downloaded from our web site at your convenience.

The GSW Subscription plan is an excellent value. Even if you upgrade the software once every few years you will save with the Subscription.

Version Upgrade Pricing with Subscription Plan	
TIME FROM DATE OF PURCHASE	**PRICE**
For the Duration of Plan (1, 2 and 3 year plans are available).	**Free**

Table 3: Version Upgrade Pricing **with** GSW Subscription Plan

The pricing for version upgrades without the Subscription is based on the period of time since the date of the original purchase or last version upgrade.

Version Upgrade Pricing without Subscription Plan	
TIME FROM DATE OF PURCHASE	**PRICE**
Less than 90 days	Free
Greater than 90 days but less than 1 year	50% of the current list
Greater than 1 year	90% of the current list

Table 4: Version Upgrade Pricing **Without** Subscription Plan

HOW TO UPDATE THE SOFTWARE

1. Download the software or use the supplied CD
2. Make sure the Directed Terminal I/O Engine is not in use.
3. Run the Setup Program for the Update as done in the original installation.
4. You may specify the same or different installation folder.

HOW TO RENEW THE GSW Subscription

Please use the following procedure when renewing the GSW Directed Terminal I/O Engine.

Step	Who		Action
1.	GSW		Send notice to customer indicating that the subscription is about to expire. The notice is sent approximately 4 to 8 weeks prior to the expiration of the plan.
2.		Customer	Places order for new subscription
3.	GSW		Confirms Order
4.	GSW		Ships current software, documentation (if applicable) and new Floating License
5.		Customer	Install new Floating License (and software if desired)
6.		Customer	Ships OLD Floating License back to GSW

Table 5: Steps to renew the GSW Subscription Plan

SYSTEM SIGNATURE – PLEASE READ

NOTE: This section only applies to Software Registration

The registration software obtains a system signature that is unique to your system. This signature is an added security measure to inhibit unauthorized personnel to obtain working copies of the GSW Directed Terminal I/O Engine.

The signature is comprised of hardware and software identifiers that exist on your system that make the target system unique. These identifiers are hashed into a Product ID and a Serial Number can be generated from this Product id.

If major hardware components of your system are removed, replaced or modified your **Serial Number** may discontinue to work and you may need a new **Serial Number** to obtain access to the Directed Terminal I/O Engine. Please contact Georgia SoftWorks Technical Support if needed.

Technical Support

In order to keep Technical Support Free please help keep our cost down.

- Gather all relevant system information.
- Write your question down. This not only helps us but also helps you in articulating the question.

If the question is not an emergency, please use e-mail at support@georgiasoftworks.com. We try to respond within 24 hours.

Or Call 706.265.1018 EST, M-F 9:00 a.m. to 5:00 p.m. and have your Product ID ready