Moxa DA-682A
Windows 7 Software Manual

The software described in this manual is furnished under a license agreement and may be used only in accordance with the terms of that agreement.

Copyright Notice

© 2013 Moxa Inc. All rights reserved.

Trademarks

The MOXA logo is a registered trademark of Moxa Inc.
All other trademarks or registered marks in this manual belong to their respective manufacturers.

Disclaimer

Information in this document is subject to change without notice and does not represent a commitment on the part of Moxa.

Moxa provides this document as is, without warranty of any kind, either expressed or implied, including, but not limited to, its particular purpose. Moxa reserves the right to make improvements and/or changes to this manual, or to the products and/or the programs described in this manual, at any time.

Information provided in this manual is intended to be accurate and reliable. However, Moxa assumes no responsibility for its use, or for any infringements on the rights of third parties that may result from its use.

This product might include unintentional technical or typographical errors. Changes are periodically made to the information herein to correct such errors, and these changes are incorporated into new editions of the publication.

Technical Support Contact Information

Moxa Americas
Toll-free: 1-888-669-2872
Tel: +1-714-528-6777
Fax: +1-714-528-6778

Moxa Europe
Tel: +49-89-3 70 03 99-0
Fax: +49-89-3 70 03 99-99

Moxa India
Tel: +91-80-4172-9088
Fax: +91-80-4132-1045

Moxa China (Shanghai office)
Toll-free: 800-820-5036
Tel: +86-21-5258-9955
Fax: +86-21-5258-5505

Moxa Asia-Pacific
Tel: +886-2-8919-1230
Fax: +886-2-8919-1231

www.moxa.com/support
# Table of Contents

1. **Introduction** .......................................................................................................................... 1-1
   Software Components .............................................................................................................. 1-2

2. **System Initialization** .............................................................................................................. 2-1
   Setting Up an Administrative Account and Creating User Accounts ................................... 2-2
   Basic Administrative Setup ................................................................................................. 2-2

3. **Enabling Windows 7 Write Protections** .............................................................................. 3-1
   Enhanced Write Filter ............................................................................................................ 3-2
   Overview ............................................................................................................................... 3-2
   Enabling Enhanced Write Filter ......................................................................................... 3-3
   File-Based Write Filter ....................................................................................................... 3-5
   Overview ............................................................................................................................... 3-5
   Enabling File-Based Write Filter ......................................................................................... 3-5

4. **Installing Moxa Software** .................................................................................................. 4-1
   The Synmap™ Virtualization Layer: Full Interoperability with Any Moxa Device ............... 4-2
   Overview ............................................................................................................................... 4-2
   Moxa Synmap OIDs List ....................................................................................................... 4-3
   Installing and Using an NMS ............................................................................................... 4-5
   Installing a Convenient NMS: Moxa MXview ................................................................. 4-6
   Basic Configuration of MXview ......................................................................................... 4-7
   Loading the Synmap MIB File ............................................................................................ 4-10
   Using Synmap OIDs to Control the DA-682A ...................................................................... 4-13

5. **Customizable Sample Code** ............................................................................................... 5-1
   Sample Code for Customizing the DA-682A........................................................................... 5-2
   The LED Control Walkthrough ............................................................................................ 5-2
   Watchdog Control Code ....................................................................................................... 5-3
   The Watchdog Control Code Walkthrough ........................................................................ 5-3

6. **System Recovery** .................................................................................................................. 6-1
   Overview: Setting Up the Recovery Environment ............................................................... 6-2
   Step 1: Prepare the USB drive .............................................................................................. 6-2
   Two Types of Recovery: Base Install, and Fully Configured ............................................. 6-3
   Step 2 (opt.): Recovering to a Stock OS ............................................................................ 6-4
   Step 3: Setting the BIOS to Boot via USB ......................................................................... 6-4
   Step 4 (opt.): Create a Custom System Image ................................................................... 6-5
   Step 4: Configure a Stock OS Restoration ......................................................................... 6-6
   Step 5: Perform a Trial Recovery ....................................................................................... 6-8
   Step 6: Reset the BIOS to its Original State ..................................................................... 6-10

A. **The 2013 Synmap OID Table** .......................................................................................... A-1
   Moxa Synmap: The Full OID Table ...................................................................................... A-2

B. **Sample Code for DA-682A Customization** ....................................................................... B-1
   LED Controls ......................................................................................................................... B-2
   Watchdog Controls .............................................................................................................. B-4
Thank you for buying Moxa’s DA-682A rackmount computer. This model comes with the Windows 7 Embedded software platform.

- **Software Components**
# Software Components

The following table details the software components of the Windows Embedded Standard 7 on this computer.

## Windows Embedded Standard 7

### Core OS:
- 32-bit support
- Remote Client
- Remote Procedure Call

### Applications and Services Development:
- .Net Framework 3.5
- Remote Desktop Protocol 7.1
- COM OLE Application Support
- COM+ Application Support
- MSMQ

### Internet Services:
- Internet Explorer 8.0
- IIS 7.0

### File Systems and Data Store:
- Windows Data Access Components
- Windows Backup and Restore

### Diagnostics:
- Common Diagnostic Tools
- Problem Reports and Solutions

### Fonts:
- Chinese (Trad. and Simp.), Japanese, Korean, Western, Middle Eastern, South East Asian, and South Asian Fonts

### Graphics and Multimedia:
- MPEG DTV-DVD Audio Decoder (MPEG-2, AAC)
- MPEG Layer-3 Audio Codecs(MP3)
- MPEG4 Decoders
- Windows Media Video VC-1 (WMV) Codecs
- DirectX and Windows Device Experience
- Windows Media Player 12

### International:
- IME Simplified Chinese Support
- IME Traditional Chinese Support
- IME Japanese Support
- IME Korean Support

### Management:
- Group Policy Management
- Windows Management Instrument (WMI)
- Windows Update

### Networking:
- Extensible Authentication Protocol (EAP)
- Internet Authentication Service
- Telnet Server
- Bluetooth
- Domain Services
- Network Access Protection
- Network and Sharing Center
- Quality of Service
- Remote Access Service (RAS)
- Telephony API Client
- Windows Firewall
- Wireless Networking
Security:
• Credential Roaming Service
• Credentials and Certificate Management
• Windows Authorization Manager (AZMAN)
• Windows Security Center
• Active Directory Rights Management
• Security Base
• Encrypted File System (EFS)

Embedded Features:
• Enhanced Write Filter (EWF)
• File-Based Write Filter (FBWF)
• Message Box Default Reply
• Registry Filter
• WSDAPI for .NET

Embedded Self-Health Diagnostic Software: SNMP-based remote scripting layer for monitoring, reporting, and control
This chapter describes how to initialize the system settings after booting up the computer for the first time.

The following topics are covered in this chapter:

- Setting Up an Administrative Account and Creating User Accounts
  - Basic Administrative Setup
Setting Up an Administrative Account and Creating User Accounts

After booting into the computer for the first time, administrators will need to create their own account and set up user accounts.

Basic Administrative Setup

1. Boot into the computer.
2. Now, you must set the password for the system. You will be presented with a dialog that will ask you for a new password.

First, type the password, and then confirm it by retyping the password in the next dialog. You may also type a password hint in case you forget your password; it is advised to save a “hint for the hint” in some safe place.

If you do not want to set a password for the administrative account (this is strongly advised against; it is a dire security failure), leave the entry box blank and click Next.

3. Select the Windows Update option, and then select your time zone and whether you will use daylight savings time.

4. Next, select the usage environment. Windows will automatically apply a preset bundle of security settings based on the network environment in which it is located. Most restrictive is public, least restrictive is home.

5. Now you can start to use DA-682A embedded computer.
Enabling Windows 7 Write Protections

This chapter describes how to set up and configure Windows 7 bit-level and file-level write protections on block storage volumes.

The following topics are covered in this chapter:

- Enhanced Writer Filter
- File-Based Write Filter
Enhanced Write Filter

Overview

Enhanced Write Filter (EWF) allows Windows 7 users to protect their all information on their storage drive from permanent changes of any sort, at the lowest level of hardware protection available: the bit level. EWF allows the operating system (OS) to boot from the hard disk, but protects the system by creating a virtual file system called an overlay. All writes to an EWF-protected volume (the hard disk, in Fig. 1) are only recorded on this virtual overlay (the EWF Volume, in Fig. 1), which is stored independently in random access memory (RAM). Because EWF does not write data directly to the hard disk but instead only records system writes to this virtual RAM overlay, any data that is “written” during system operation will disappear upon the next re-boot. This approach allows the system to operate as if it is writeable when in reality all OS and user-space file systems are stored in a permanent, read-only state. If desired, the data written to the overlay may be committed to the protected volume, but this requires additional effort. Refer to the following figure (from Microsoft) for an overview of the EWF structure.

To get more details about EWF configuration and usage, you may:

- Visit Microsoft’s EWF Volume Configuration help pages.
- Visit Microsoft’s EWF overview on the official Microsoft EWF help pages.
- Visit Microsoft’s detailed description of EWF modes on the EWF help pages.
- Visit Microsoft’s detailed description of the EWF API.

For the EWF commands, refer to the MSDN web site:

http://msdn.microsoft.com/en-us/library/ms940853%28v=winembedded.5%29.aspx
Enabling Enhanced Write Filter

Follow these steps to enable the Enhanced Write Filter

1. First open the console by running `cmd.exe`.

2. To verify that Enhanced Write Filter is disabled, type `C:\>ewfmgr c:`.
3. To enable the Enhanced Write Filter, type `C:\....>ewfmgr c: -enable`.

4. Reboot the system so the changes may take effect, and then verify that Enhanced Write Filter has been enabled at boot by typing `C:\....>ewfmgr c`:

5. To disable the Enhanced Write Filter, type `C:\....>ewfmgr c: -commitanddisable`.
File-Based Write Filter

Overview

File-Based Write Filter (FBWF) is similar to EWF, but is enforced at the file level rather than at the hardware (bit) level. This means it is slightly less secure, but this allows it to provide more user features than is possible with EWF.

File-Based Write Filter (FBWF) allows the Windows Embedded platform to maintain the appearance of read and write access on write-sensitive or read-only storage. FBWF makes read and write access transparent to applications.

Writing to storage media may be undesirable or impossible in embedded devices. FBWF redirects all writes targeted for protected volumes to a RAM cache called an overlay. Used in this context, an overlay is similar to a transparency overlay on an overhead projector. Any change made to the overlay affects the picture as seen in the aggregate, but if the overlay is removed, the underlying picture remains unchanged.

One of FBWF’s more advanced features allows the user to specify a directory where data may be more conveniently written to the data drive than is possible with EWF. The DA-682A’s default settings have already configured a default directory for this; you may use the c:\temp directory for files which require write access.

Enabling File-Based Write Filter

To enable file-based write filtering, do the following:

1. To verify that Enhanced Write Filter is disabled, type `C:\...\fbwfmgr /displayconfig` to check the current status.

2. Type `fbwfmgr /enable` to enable the FBWF, and then reboot the system to take effect.
3. After the system has rebooted, type `C:\...\>fbwfmgr /displayconfig` again to check if the status has been changed to enabled and will start at boot-time.

To get more details about EWF configuration and usage, you may:

Go to Microsoft’s [FBWF Installation and Configuration](#) help pages.

Go to Microsoft’s [FBWF overview](#) on the official Microsoft EWF help pages.

Go to Microsoft’s detailed description of [FBWF features](#) on the EWF help pages.

Go to Microsoft’s detailed description of the [FBWF API](#).
This chapter describes how to configure and install the bundled Moxa software that comes with this computer. Moxa software allows users to build scripts and custom software to automatically or remotely control and monitor this computer for both process- & infrastructure-level needs.

The following topics are covered in this chapter:

- **The Synmap™ Virtualization Layer: Full Interoperability with Any Moxa Device**
  - Overview
  - The Synmap Design Concept
  - Moxa Synmap OIDs List

- **Installing the Synmap Virtualization Layer**

- **Installing and Using an NMS**
  - Installing a Convenient NMS: Moxa MXview
  - Basic Configuration of MXview
  - Loading the Synmap MIB File
  - Using Synmap OIDs to Control the DA-682A
The Synmap™ Virtualization Layer: Full Interoperability with Any Moxa Device

Overview

Synmap™ is Moxa’s revolutionary software virtualization, an evolutionary advance in network device control that adapts solid, reliable SNMP into a fully portable remote procedure interface. Synmap allows engineers to automate remote processes using SNMP object identifiers (OIDs) rather than device- or OS-specific API addressing, making a scripted Synmap procedure fully interoperable with any other Synmap device. This means that a script created for one Synmap device may be directly copied to another, immediately conferring the same functionality. This eliminates the need for rewriting and compiling code for newly configured devices, significantly reducing maintenance and deployment times.

SNMP is lightweight and easy-to-configure, and is already long-popular with IT professionals; it also enjoys comprehensive native support in high-level languages like .NET, Java, Python, or Ruby. For these reasons, the Synmap framework has re-imagined SNMP as a universal configuration and control interface for remote procedures, adapting it to not only monitor and control device internals like temperature, BIOS parameters, and local interfaces, but also to report on and automate tasks at the process layer, as well. Easily integrated into any existing Network Management System (NMS), Synmap devices are a flexible and cost-effective upgrade that returns obvious benefits to any IA network.

Synmap currently allows you to use SNMP for remote monitoring and control of a select set of computer processes, but its list of features is rapidly growing. Using Synmap’s fully portable scripts, engineers will soon be able to:

• Access, monitor, control, and report on digital I/O at both the process and hardware layers
• Use OIDs to monitor, configure, and give process control over serial ports and other interfaces
• Monitor and control system attributes and process events via any NMS
• Build automated remote procedures using Synmap OIDs called by simple shell scripts, or a preferred high-level language like Python, Perl, or VBScript—all without any need for low-level APIs, or platform-specific libraries
• Significantly simplify and reduce development times for custom utilities and automated executables
• Gain scripting and automation independence from OS-dependent libraries

All of this may be achieved using simple, reliable, and familiar SNMP, the easily accessible standard that IT engineers are already familiar with.

The Synmap Design Concept

Synmap is a software design concept that offers programmers a wholly unique and superior conception of infrastructure development for IA control. Instead of using low level APIs, Synmap adapts the higher level SNMP protocol to serve as a universal API across all machines. With Synmap, application developers gain several benefits, the two biggest being a significantly reduced learning curve for control APIs and remarkable code portability. For example, if a user wants to control GPIO in a Linux environment, an application developer needs to generate code that follows the pseudo code shown below:

1. Open() the device node
2. Read() the file descriptor
3. Read() the return value, and make a logical decision
4. Perform an ioctl function on the file descriptor
5. Close() the file descriptor

The above example shows how this is done in a *NIX environment. In a Windows environment, it looks a little different, but the process is essentially the same, and of equal simplicity:
1. Open a required file handle using `mxgpio_open`.
2. Get data using the file handle, an assigned port, and `mxgpio_get_data`.
3. Evaluate the returned data, and make a logical/control decision.
4. Use `mxgpio_set_data` with the file handle to set a value.
5. Use `mxgpio_close` to close the file handle.

These examples show, in concise form, the difficulties application developers face when dealing with low level APIs. Developers must understand each system’s API and track down various device node IDs from within the user manual, the sample code, or the general system. Synmap significantly simplifies this situation. In comparison to the example just shown, the pseudo code that replaces it will look something like this:

- GET an OID using SNMP and the localhost connection (127.0.0.1)
- Evaluate the returned data, make a logical decision
- SET an OID using SNMP and the localhost connection (127.0.0.1)

The benefits of using SNMP in this way should be clear.

- First, the code is easily migrated across different computers and even different operating systems, because Moxa’s SNMP libraries are supported on both Windows XPE and Linux, as well as a host of other platforms.
- Second, the program can just as easily be ported to the network for remote operations simply by changing the localhost connection (127.0.0.1) to the target IP address and hostname.
- Third, the time needed to learn how to control a peripheral is drastically cut; all one needs to do is understand how to use an SNMP OID, and start scripting.
- Fourth, Developers are free to choose any kind of programming languages or utilities with which they might be familiar, so long as they are apropos to the platform(s) on which they will be used. For example, in place of the C API, Microsoft developers might want to use the SNMP libraries in .NET or Java to control remote Linux devices, or it can be flipped around so that Linux developers use Net-SNMP libraries to control remote Windows XPE machines.

All of these things mean that the Synmap virtualization makes the work of programming custom applications much faster and simpler, and dramatically increases code interoperability. Complex controls such as USB notify, mounting information, and BIOS settings have been integrated into the Synmap engine, so that creating a customized monitoring or control application now only requires the coordination of a few SNMP SET/GET calls, potentially allowing developers to save on hundreds of lines of code when authoring new applications.

### Moxa Synmap OIDs List

The following table shows the OIDs currently supported on the DA-682A. For a full review of all Moxa Synmap OIDs, check the Appendix section.

<table>
<thead>
<tr>
<th>Item Name</th>
<th>OID</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>productName</td>
<td>1.3.6.1.4.1.8691.17.1.1.1</td>
<td>read-only</td>
<td>Returns the product name</td>
</tr>
<tr>
<td>productDesc</td>
<td>1.3.6.1.4.1.8691.17.1.1.2</td>
<td>read-only</td>
<td>Returns a short device description</td>
</tr>
<tr>
<td>productVersion</td>
<td>1.3.6.1.4.1.8691.17.1.1.3</td>
<td>read-only</td>
<td>Returns product version</td>
</tr>
<tr>
<td>productBuildDate</td>
<td>1.3.6.1.4.1.8691.17.1.1.4</td>
<td>read-only</td>
<td>Returns the last software build date, YYMMDHH format</td>
</tr>
<tr>
<td>tempSensorsIndex</td>
<td>1.3.6.1.4.1.8691.17.1.5.1.1.1.1</td>
<td>read-only</td>
<td>Reference index showing all available temperature sensors; starts from 1</td>
</tr>
<tr>
<td>tempSensorsDevice</td>
<td>1.3.6.1.4.1.8691.17.1.5.1.1.1.2</td>
<td>read-only</td>
<td>Returns a list of the unique string values that the database associates with a temperature sensor index value. May be used with</td>
</tr>
<tr>
<td>Temp Sensor Index</td>
<td>Description</td>
<td></td>
<td></td>
</tr>
<tr>
<td>------------------</td>
<td>-------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>tempSensorIndex</strong></td>
<td>To identify temperature sensors by name/position, and to subsequently call them in scripts</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**tempSensorsValue**

1.3.6.1.4.1.8691.17.1.5.1.1.1.3 read-only

The reading returned by the thermometer, in degrees Celsius

*Note: On the DA-682A, temp. is currently represented as the current reading truncated to its base integer, without rounding*

**accelerometerIndex**

1.3.6.1.4.1.8691.17.1.5.1.3.1.1 read-only

Reference index showing all available accelerometers; starts from 1

**accelerometerAxis**

1.3.6.1.4.1.8691.17.1.5.1.3.1.2 read-only

Shows the format in which accelerometer’s axial values will be returned. This will always be in $X_n, Y_n, Z_n$ form, with $n$ the index number of the accelerometer with which the axis is associated

**accelerometerValue**

1.3.6.1.4.1.8691.17.1.5.1.3.1.3 read-only

Returns a block of the last hundred recorded values for all three axes of a particular accelerometer, taken from the log file; this will be g-force in micro gravities ($\mu G$).

*Note: this OID requires the accelerometer’s log file to be activated and up-to-date. If the log file is unreadable for any reason, this OID will not return data.*

**accelerometerTimestamp**

1.3.6.1.4.1.8691.17.1.5.1.3.1.4 read-only

Returns the accelerometer timestamp

**usbDeviceProductID**

1.3.6.1.4.1.8691.17.1.6.4.1.3.1.3 read-only

Returns the USB’s hexadecimal product ID

**usbDeviceActiveClass**

1.3.6.1.4.1.8691.17.1.6.4.1.3.1.4 read-only

Returns the USB device class for any connected device

**watchdogPeriod**

1.3.6.1.4.1.8691.17.1.6.6.2.1 read-write

Sets the watchdog’s timeout interval, in seconds:

- Entering 0 disables the watchdog
- Entering an integer from 1 to 255 configures the timeout interval to that number of seconds

**watchdogStatus**

1.3.6.1.4.1.8691.17.1.6.6.2.2 read-write

Returns the watchdog’s current status and timeout interval

**powerPolicy**

1.3.6.1.4.1.8691.17.1.7.2 read-write

Returns the system’s current power policy, as configured by the Windows 7 Embedded OS
Installing the Synmap Virtualization Layer

The following steps will install Synmap.

1. Double click mxSynmap_setup.msi, found in the Utility folder located on the software DVD under \utility\1.mxSynmap\. Then click Next to start the Synmap setup wizard.

2. In the middle of the dialog, the button Disk Cost will display how much space the Synmap software package will occupy on your storage drive, as well as the remaining storage space on the drive where the system is stored. At the bottom of the dialog, select whether Synmap will be installed for every user across the entire system, or just for the current user account. Above that (in the text dialog), you may click the Browse button to browse the file tree and select the folder where you want to install the package, or simply click Next to install Synmap to the default folder.
3. Click through the next few dialogs to complete the installation of the Net-SNMP agent. The SNMP agent will not begin working until you reboot the DA-682A computer.

Installing and Using an NMS

For full implementation, Synmap requires (like any SNMP-based system) an NMS to become fully functional; an NMS with an MIB browser also makes using SNMP a far simpler task. If you already have your own MIB browser, you can skip this section. However, if your network is lacking an NMS then you may install a free version of Moxa’s MXview to get Synmap up and running. MXview provides an MIB browser and an interface that will allow you to monitor and control any Synmap enabled device. This section will walk you through a basic MXview installation, and show you how to use the MXview MIB browser to start working with the DA-682A’s MIB.

Installing a Convenient NMS: Moxa MXview

1. MXview is included on your DA-682A software DVD. Double click mxView_Trial_V2.3.msi in the Utility folder, which you can find on the software DVD in \utility\mxViewTrial. Select OK to choose the language, and when the next dialog appears click Next to continue.

2. Accept the licensing agreement and click Next to move to the licensing dialog.

3. On the next dialog you may change the folder and path where MXview will be installed. On the next, you may select where MXView shortcuts will be stored in the Windows Start Menu.
4. Next, you may register MXview as a Windows service and create a desktop shortcut.

5. After you have completed the pre-install configuration, click **Install** to transfer MXview to disk and wind up the installation.

6. After MXview has installed, you must enter the IP address of the machine on which it is located. This may be the *localhost* address, **127.0.0.1**, or if you are connecting to MXview over a LAN it will be a remote IP address. Additionally, you must configure the ports which MXview will use for HTTP and HTTPS communications. Once the installation is complete, you may choose to restart the computer to get MXview up and running.

7. After rebooting, the MXview shortcut will appear on your desktop (shown at right). Click on the shortcut to continue on to the next section and begin the MxView setup.

**Basic Configuration of MXview**

1. Open MXview (see step 7 of the last section, immediately above) and select **Start** to initialize the MXview NMS; wait for the **System Status** notification to change to **Running**, then click **Launch Client**.
2. If opening Microsoft Internet Explorer for the first time, make sure to turn off the **suggested sites** feature (shown at right). If you wish to use another browser you may, and IE’s other settings may be configured to your own preferences.

3. The IP address for MXview will be 127.0.0.1 followed by a colon and the HTTP port you configured MXview to communicate over (in **step 6 of MXview**). If you have used the suggested settings above, then to login using HTTP would be 127.0.0.1:81, and using HTTPS you would use 127.0.0.1:443.

For the login, the default username is **admin**, with a blank password.

![MXview](image)

**WARNING**

For security’s sake, Moxa strongly recommends resetting the password to a strongly secure password of at least 8 characters, mixing numbers and symbols in a non-word series.

4. When opening your browser for the first time, a warning message will pop up telling you to install the Java runtime environment. Click **OK** to continue.

![Java Runtime](image)

5. Click the title bar, and select **File Download Blocked-->Download File** to continue.
6. Select **Run** to download and install the Java Runtime Environment (JRE), and when Windows posts a security warning asking if you wish to run the installer, click **Run** again.

7. Click **Install** to continue.

8. Click **Close** to complete.
9. In Windows IE, a banner will appear at the top of the browser window. Click the message and select **Enable Intranet Settings**.

10. A security warning will appear, telling you that intranet settings are not secure enough for the open Internet. Click **Yes** to ignore this, and when another security warning appears telling you that the application’s digital signature is not recognized, click **Run**.

11. The **Moxa MXview Setup Wizard** will now appear. You may click **Next** if you wish to enter the setup routine, or select **Cancel** to launch the program immediately.
   
   If you click **Cancel**, the program will be launched. It should look like the screenshot below.

### Loading the Synmap MIB File

To load the Synmap MIB file you must first have a running NMS; if you do not have an NMS, you may install the free version of MXview included with your DA-682A computer software. If you have already started MXview, go directly to step 5 of this section.
1. Click the MxView Service shortcut on the desktop.

2. Click **Start**, wait for the **System Status** indicator to show **Running**, and then select **Launch Client**. When the MXview Setup Wizard appears, click Cancel to skip the setup process and directly open the MXview interface.

3. Select **MIB-->MIB Browser**.

4. After the MIB browser has opened, select **File** from the browser’s upper left corner, and then **Load MIB**.
5. Navigate to c:\usr\share\snmp\mibs\ and select MOXA-SYS-MIB.txt.

6. After opening the Synmap MIB in the browser, check that it appears in the File window. If it is not, then it is likely because the MIB file is corrupted. To remedy this, re-copy the MIB file from the software DVD, and re-load the MIB file following the instructions above.
Using Synmap OIDs to Control the DA-682A

Follow these steps to use Moxa Synmap to read the MIB and set up controls for this computer.

Retrieving Basic Device Information

1. In this first step, we will use Synmap to retrieve specific device information about this computer. First, use the SNMP Get Next command to navigate the OID tree by clicking through these items: MOXA-SYS-MIB\VALUES\Moxa\embeddedComputer\MoxaSystem\productInfoMgmt

2. When you reach the final layer of OIDs, you will need to select GetSubTree to display the available information. When you use the MIB viewer to select the productInfoMgmt OID, you will see the following information displayed in the MIB viewer’s information window: Product Name (DA-682A), Product Version (1.0.0), and Product Build Date (13080611).
Using Synmap to Control the Programmable LEDs

The following figure shows the locations of the LED indicators on the DA-682A computer. Using custom scripts, you can set these LEDs to be used as indicators for your own applications.

![Programmable LED](image)

The following table shows the available OIDs for the DA-682A LEDs.

<table>
<thead>
<tr>
<th>Item Name</th>
<th>OID</th>
<th>Access</th>
<th>Values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ledNumber</td>
<td>1.3.6.1.4.1.8691.17.1.6.2.1</td>
<td>read-only</td>
<td>N/A</td>
<td>Returns the total number of LEDs</td>
</tr>
<tr>
<td>ledIndex</td>
<td>1.3.6.1.4.1.8691.17.1.6.2.2.1.1</td>
<td>read-only</td>
<td>N/A</td>
<td>Returns a list of numbers that correspond with the LEDs, used by SNMP to identify the LEDs; begins with 1</td>
</tr>
<tr>
<td>ledPort</td>
<td>1.3.6.1.4.1.8691.17.1.6.2.2.1.2</td>
<td>read-only</td>
<td>N/A</td>
<td>Returns the names by which the index of LEDs may be called when used in scripts; begins with 1</td>
</tr>
<tr>
<td>ledValue</td>
<td>1.3.6.1.4.1.8691.17.1.6.2.2.1.3</td>
<td>read-write</td>
<td>0(off), 1(on)</td>
<td>Returns/sends a value indicating/changing the LED state</td>
</tr>
</tbody>
</table>

Follow these steps to configure the programmable LED indicators on the DA-682A computer.

1. Start up MXview (or some other NMS) and open the MIB browser. For detailed instructions on loading MXview, you may refer to [Loading the Synmap MIB File](#), steps 1 to 3.

2. In the MIB browser, navigate the OID tree to `MOXA-SYS-MIB\VALUES\Moxa\embeddedComputer\MoxaSystem\peripheralMgmt\perLedMgmt\ledTable\ledEntry\ledValue`. To change the state of an LED from on to off (or vice versa) click Set on the corresponding ledValue OID (shown in the figure at right). This will open a dialog called the Set Value dialog.
3. The Set Value dialog has four fields: one for the OID that is being called; another called Index, that identifies the LED being manipulated; and a third called Value, to set the state of the LED. The fourth field (Syntax) informs you of the type of character to be used to set the value. In this case, we are using the integers 0 and 1: 0 turns the LED off, and 1 turns it on. After entering the correct information into the fields (as shown at right), click on the Set button and check if the LED1 indicator has lit up (LED 1 is shown in the figure at the top of this section, Using Synmap to Control the Programmable LEDs).

4. Next, set the value to 0 and click Set to turn off the LED, then visually verify that the LED has successfully turned off. If you experience problems at this point, please review the instructions above and if the problem persists, contact Moxa technical support or visit our website at http://www.moxa.com/support/.

Using Synmap to Read the Temperature Sensor

The following table shows the temperature sensor OIDs.

<table>
<thead>
<tr>
<th>Item Name</th>
<th>OID</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tempSensorsIndex</td>
<td>1.3.6.1.4.1.8691.17.1.5.1.1.1</td>
<td>read-only</td>
<td>Returns a list of numbers (beginning with 1) that corresponds with the available T-sensors; these identifiers are used by SNMP and for system scripts.</td>
</tr>
<tr>
<td>tempSensorsDevice</td>
<td>1.3.6.1.4.1.8691.17.1.5.1.1.2</td>
<td>read-only</td>
<td>Returns a list of string values identifying the temperature sensors by name/location. Possible values are System, or CPU, for the CPU's internal thermometer.</td>
</tr>
<tr>
<td>tempSensorsValue</td>
<td>1.3.6.1.4.1.8691.17.1.5.1.1.3</td>
<td>read-only</td>
<td>The reading returned by the thermometer, in mini-degrees Celsius. Note: On the DA-682A, temp. is currently represented as the current reading truncated to its base integer, without rounding.</td>
</tr>
</tbody>
</table>

To use Synmap to check the temperature sensor (T-sensor) of the DA-682A computer, follow the steps below.

1. Start up MXview (or some other NMS) and open the MIB browser. For detailed instructions on loading MXview, you may refer to Loading the Synmap MIB File, steps 1 to 3.

2. Navigate to MOXA-SYS-MIB\VALUES\moxa\embeddedComputer\moxaSystem\sensorMgmt\sensorObject\tempSensorTable in the MIB Browser, and then select Get Subtree to open the T-sensor’s OID tree.
3. All of the T-sensor information should now be displayed in the MIB browser; if it is not, then there is a problem with your SNMP/Synmap configuration or a malfunction in the T-sensor. The temperatures are all displayed in Celsius.
Using Synmap to Read the Voltage Sensor

The following table shows the voltage sensor OIDs.

<table>
<thead>
<tr>
<th>Item Name</th>
<th>OID</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>voltSensorsIndex</td>
<td>1.3.6.1.4.1.8691.17.1.5.1.2.1.1</td>
<td>read-only</td>
<td>Returns a list of numbers (beginning with 1) that corresponds with the available T-sensors; these identifiers are used by SNMP and for system scripts.</td>
</tr>
<tr>
<td>voltSensorsDevice</td>
<td>1.3.6.1.4.1.8691.17.1.5.1.2.1.2</td>
<td>read-only</td>
<td>Returns a list of string values identifying the temperature sensors by name/location. Possible values are V_CPU, V_GFX, VTT_CPU, VCCSA_CPU.</td>
</tr>
<tr>
<td>voltSensorsValue</td>
<td>1.3.6.1.4.1.8691.17.1.5.1.2.1.3</td>
<td>read-only</td>
<td>The reading returned by the voltage, in mili-Voltage.</td>
</tr>
</tbody>
</table>

To use Synmap to check the voltage sensor of the DA-682A computer, follow the steps below.

1. Start up MXview (or some other NMS) and open the MIB browser. For detailed instructions on loading MXview, you may refer to [Loading the Synmap MIB File](#), steps 1 to 3.

2. Navigate to `MOXA-SYS-MIB\VALUES\moxa\embeddedComputer\moxaSystem\sensorMgmt\sensorObject\voltSensorTable` in the MIB Browser, and then select **Get Subtree** to open the Voltage sensor's OID tree.

3. All of the Voltage sensor information should now be displayed in the MIB browser; if it is not, then there is a problem with your SNMP/Synmap configuration or a malfunction in the Voltage sensor. The Voltages are all displayed in Celsius. In the screenshot below.
This chapter uses sample code to show how scripting may be used to add customized capabilities to the DA-682A computing platform.

The following topics are covered in this chapter:

- **Sample Code for Customizing the DA-682A**
  - The LED Control Walkthrough

- **Watchdog Control Code**
  - The Watchdog Control Code Walkthrough
Sample Code for Customizing the DA-682A

This computer comes with several pieces of sample code that users may use to customize its behavior for LED notifications; temperature, vibration and GPS events; or to initiate emergency reboots when critical system applications come to a halt.

The LED Control Walkthrough

The DA-682A is designed with 8 programmable LEDs that integrators and system administrators may customize for their notification needs. The source code for controlling LED behavior is located in the folder `<Software DVD>\examples\Example\C++\LED`, while the compiled executable LED.exe is located under `<Software DVD>\examples\Release`. You can follow the steps below to test the LED control script, or you may freely modify the control script to create customized patterns that are associated with specific system events. To review the LED control code, see The LED Control Script section in Appendix B of this manual.

1. Create the `c:\programs\examples` folder and copy LED.exe into that folder. Run LED.exe.
2. You will be presented with a menu of three choices; first, select 1 to display the LED's current status.
3. In the screenshot below, the user has selected 1 and is viewing the current status of all LEDs. The value 0 shown next to each entry indicates that all of the LEDs are currently turned off. Visually verify this by examining the DA-682A's front panel.
4. You may now run LED.exe again, but this time select 2. At the next prompt, enter the ID number (0 to 7) of an LED to activate, and when prompted enter either 1 (to turn it on), or 0 (to turn it off). The LEDs on the main DA-682A panel correspond to 0 through 3; 4 and 5 correspond to those on the first module; and 6 and 7 to those on the second.
Watchdog Control Code

The code for controlling the watchdog/COP timer is the simplest and least-customizable of the included sample scripts. The code itself is provided on the software DVD, under examples\Example\C++\WatchDog, and the executable file Watchdog.exe is on the software DVD \examples\Release. This sample code may be modified to integrate the watchdog timer with specific applications. Using this code any program may be set up so that the watchdog timer will provide a last-line failsafe against application crashes. For instance, the DA-682A may be set up so that whenever a mission-critical application fails the watchdog timer will send a message to a system administrator and then initiate an automatic reboot. To test the watchdog executable, follow the steps below.

The Watchdog Control Code Walkthrough

1. If you haven’t yet, create the folder c:\programs\examples, then copy over Watchdog.exe and run the script.

2. To keep the system from rebooting, users need to press Enter at least once every 10 seconds, or the system will reboot automatically.

3. To stop the watchdog, press q to exit the program.
The DA-682A ready-to-run embedded computers are a Windows Embedded Standard 7 platform. This chapter describes the recovery process in the event of system instability.

The following topics are covered in this chapter:

- **Overview: Setting Up the Recovery Environment**
- **Step 1: Prepare the USB drive**
  - Two Types of Recovery: Base Install, and Fully Configured
- **Step 2 (opt.): Recovering to a Stock OS**
- **Step 3: Setting the BIOS to Boot via USB**
- **Step 4 (opt.): Create a Custom System Image**
- **Step 4: Configure a Stock OS Restoration**
- **Step 5: Perform a Trial Recovery**
- **Step 6: Reset the BIOS to its Original State**
Overview: Setting Up the Recovery Environment

A DA-682A computer, a 4 GB (min.) USB drive, and a copy of the recovery suite are all required to set up the DA-682A's system recovery environment.

The recovery procedure itself requires only a DA-682A computer and a bootable USB drive.

The following steps describe the basic process of setting up the system recovery environment:

1. First, prepare the USB drive by copying over to it a bootable recovery environment; this comes in the form of an ISO image, and can be found on your software CD.
2. Here, you may choose to create a bare-bones stock OS recovery image; if you choose this option as your recovery method, keep in mind that any applications or scripts you install later will be lost if a recovery is required.
3. Here, you will reset the BIOS so the USB port is the first boot priority. If you are initiating a recovery from a key you have already configured, this will be your starting point. The system will re-booted into the Clonezilla recovery environment found on the USB.
4. This step describes how to create an exact copy of a fully configured system on the USB drive. This is the alternative to the stock OS recovery offered in Step 2.
5. This step describes how to perform a recovery; you may use it to run a trial recovery and test your setup.
6. This step explains how and why to return the BIOS to its original state.

Step 1: Prepare the USB drive

1. From the software DVD that came with your computer start the Clonezilla imaging program (within the current OS) by starting tuxboot-windows-23.exe, which is found in the \recovery\ folder.
2. At the right, select Pre-Downloaded and set the dropdown to ISO.
3. Browse the CD to locate the Clonezilla ISO image by clicking the button with an ellipsis (...).

![Tuxboot window](image)

*Homepage:* [http://clonezilla.org](http://clonezilla.org)
*Description:* CloneZilla live is a distribution used for disk backup and imaging. The stable branch of CloneZilla Live is based on Debian.
*Install Notes:* CloneZilla live is bootable and runs in live mode; no installation is required to run it.
*Download Path:* [Clonezilla Live Stable 2.6 source](http://clonezilla.org)

4. In Windows Explorer, navigate to the \recovery\ folder on the software DVD and select the Clonezilla recovery environment’s ISO image.

![Windows Explorer](image)
5. Set the **Device Type** (lower left-hand corner) as **USB Drive**, then set the **Drive** dialog to the letter under which the USB is currently mounted.

6. Click **OK**, and the Clonezilla recovery environment (plus bootloader) will be copied to your USB drive.

**Two Types of Recovery: Base Install, and Fully Configured**

Because of the naming conventions used, for any given computer only a single system image may be stored on any given USB drive. Consequently, at this point, users need to make a decision about which sort of system recovery is preferred:

A. A recovery image of a **fully configured OS**, with user-installed software applications and scripts, or

B. A recovery image of only the **basic, newly-installed root OS**.

A: To configure the recovery environment to copy over a fully configured system, users should click **Reboot Now** to close the installation environment and restart the computer. They should then proceed to the next section, **Step 3: Setting the BIOS to Boot via USB** and continue the installation of the recovery environment by continuing to **Step 4 (opt.): Creating a Custom System Image**.

B: To configure the recovery environment to boot into a clean OS image with no applications, users should instead click **Exit** here to complete the installation and return to the OS. At this point, **Step 1** has been completed, and you should proceed to **Step 3: Setting the BIOS to Boot via USB**, and then go directly to **Step 4: Perform a Test Restoration**.
Step 2 (opt.): Recovering to a Stock OS

The instructions which follow describe how to set up the recovery environment that will restore the operating system to a pristine post-install state. If you have installed any software on your system, then following these directions will result in all custom applications and code being wiped from the operating system. If the computer has already been heavily customized with user applications and local scripts, then skip this section and instead go to the next section, Step 4: Setting the BIOS to Boot via USB. There, you will begin the process of copying over a full system image.

Creating a post-install rescue drive involves just two steps: preparing the USB drive, and then copying the rescue image (found on your Moxa software CD) to it.
1. First, if you haven’t already, complete step 1 by preparing the USB drive.
2. Next, copy the stock OS image (found on the software CD that shipped with your computer) over to the USB drive; the image will be found in the recovery directory, /media/cd0/recovery/os_image, and will be copied to the USB file tree at /media/usb0/home/partimag. Depending on how the USB and CD have been automounted, you will use a command much like this:

```
moxa@Moxa:~# cp /media/cd0/recovery/os_image /media/usb0/home/partimag/
```

That’s it. You have now configured a USB recovery key that will recover your computer to the stock operating system it shipped with. If you wish, you may now undertake a trial recovery To do this, continue on to step 3, setting the BIOS to boot over USB, skip step 4, and then go on to step 5, performing a system recovery.

Step 3: Setting the BIOS to Boot via USB

At this stage, users will set the BIOS so that the system boots directly from the USB. This must be done before the rest of the system recovery environment may be configured

1. Turn on the computer and, during the POST process, press F2 until you hear a long beep. You should then enter the BIOS setup menu. Select SCU to enter the BIOS setup menu.

2. Use the right/left arrow keys to navigate to the Boot tab, use the up/down arrow keys to navigate to the Legacy option, and then press Enter.
3. Select **Boot Type Order** to open the dialog that will allow you to set the boot priority for the system drives.

4. Use the arrow keys to highlight **USB** and then press the plus key (+) to move it to the first position, or select the other options above it and use the dash key (−) to move them down.

5. Press F10 and then press Enter to save and exit the BIOS configuration interface. This should initiate the next reboot, during which your system should now boot from the USB drive.

**Step 4 (opt.): Create a Custom System Image**

The instructions which follow are only to be used if you decided in Step 1 of this process to create a full copy of an already-configured system. If you have not yet installed any software on your system, then skip this section and instead go to Step 4: Perform a Test Restoration to copy over a clean OS image.

Using this procedure, you will save to the USB drive a copy of the entire system as it is currently configured to be used as a full system recovery image should the system crash. **All files under F:\home\partimag\ will be overwritten.**
You should have already changed the BIOS settings to set the USB drive as the first boot priority. If you have not yet reset the boot priority, first return to Step 3: Setting the BIOS to Boot via USB, just above, and follow the directions there.

1. Once the system has launched and the DA-682A has booted the recovery environment from the USB drive, navigate to the entry Clonezilla Live Save Disk, and select it by pressing Enter. This will take you into the recovery image creation environment, allowing you to copy your full system setup to the USB drive.

2. The DA-682A will now boot into the image creation environment. Wait for the boot process to finish.

3. Once the image creation environment has completed booting up, you will be given a warning and asked if you wish to continue. Please keep in mind that if you create the recovery image, then any residual files currently copied to the /home/partimag directory will be deleted. If there are any files remaining in the USB partition image directory and you wish to save them, you must exit the recovery environment and copy these files to another disk. If you wish to continue with the image creation, press Y (case insensitive) to continue (screenshot on the next page).
WARNING

The same filename is used for all recovery images, whether for the full system backup or for the clean OS image installation. This means that currently, it is impossible to have more than one system image per USB drive.

4. At this point, the recovery environment will copy of the entire hard drive to your USB drive. This will likely take several minutes, and perhaps as long as half an hour. Do not remove the USB drive during this time; wait patiently for the process to finish. Depending on the speed of your USB drive, this may be a good time to get a cup of coffee, or take a nap.
5. At this point you may choose to power down the computer (press 0), reboot (press 1), enter a console terminal (access a console TTY -- press 2), or re-initiate the entire procedure (press 3). **Do not remove the USB drive until you have rebooted or powered down the system.**

6. Once you have powered down the system and removed the USB drive, you have finished configuring the recovery environment. The USB drive should be clearly labeled and stored in a safe place. You may now continue to the next section, where you will return the BIOS to its original state (Step 4) and test the recovery procedure for successful configuration (Step 5).

**Step 4: Configure a Stock OS Restoration**

From within the desktop environment, the user should then manually copy the directory containing the base OS from the software DVD over to the USB drive. To do this, copy `#:\<Software DVD>\recovery\os_image` over to the partition image directory, `F:\home\partimag\`.

**Step 5: Perform a Trial Recovery**

Connect the USB drive to any of the DA-682A's USB ports and then reboot the computer. The system will boot from the USB into the Clonezilla boot loader.

1. Select **Clonezilla Live Restore Disk** to boot into the system restoration environment.
2. Wait for the boot process to finish.

3. At this point, the system will remind you that you are about to overwrite your entire operating system with a new drive image, and ask you if you want to continue. When prompted, enter Y (case insensitive) from the keyboard to start the system restoration process. Any other letter or Ctrl-C will cancel it and exit Clonezilla.

4. The system will give you another warning that you are about to overwrite your hard drive, and erase all data on the partition listed (sdal, in the example below). If you wish to continue, enter Y (case insensitive).
5. Now, Clonezilla will copy the system image you have configured on to your primary system drive. Your original system (and any stored data or configurations that were made after the recovery disk was created) will be entirely wiped clean. Wait for the process to finish; depending on the system, this should take about 10 minutes.

6. At this point, complete the restoration by selecting (0) Poweroff. This will shut down the computer; however, if the Power Switch remains inserted in the front panel of the computer and is left in the ON position, then the system will immediately initiate a soft reboot. To avoid this, users may use the switch to cut power to the computer immediately following the shutdown, or may simply remove the power switch from the front panel and then use the console to shut down the computer by pressing 0.

7. After the computer has powered down, remove the USB drive and store it in a safe place.

Step 6: Reset the BIOS to its Original State

Now you will need to return the boot priority to its original configuration so that the system will boot from the original disk. This is done for two reasons; the first is security, so that the machine may not be rebooted from unauthorized USB drives.
The second reason, however, is functional: currently, if the DA-682A is set to boot from the USB drive, then the DA-682A will hang any time a USB data drive (i.e.: non-bootable image) is inserted in the machine at boot time. The DA-682A does not currently have the capacity to distinguish between simple USB data drives and boot-capable OS drives.

1. Reboot the system, and, during the POST process, press F2 until you hear a long beep. You should then enter the BIOS setup menu. Select SCU to enter the BIOS setup menu.

2. Use the left/right arrow keys to navigate to the Boot tab, and then press Enter.

3. Use the up/down arrows to highlight Legacy in the boot tab’s menu, and press Enter.

4. Use the up/down arrow keys to navigate to the Boot Type Order link, and then press Enter.

5. Use the up/down arrows to highlight Hard Disk Drive and then use the plus/minus signs (+ -) to move it to the first boot priority position.

6. Press F10 and then press Enter to save and exit the BIOS configuration interface. This should initiate the next reboot, and your system should now boot from the USB drive.
The 2013 Synmap OID Table

This appendix describes the full Synmap OID table as of July, 2013.

The following topics are covered in this appendix:

- Moxa Synmap: The Full OID Table
### Moxa Synmap: The Full OID Table

The following table shows the full list of the Moxa Synmap OID.

<table>
<thead>
<tr>
<th>Item Name</th>
<th>OID</th>
<th>Access</th>
<th>Description</th>
<th>Supported</th>
</tr>
</thead>
<tbody>
<tr>
<td>productName</td>
<td>1.3.6.1.4.1.8691.17.1.1.1</td>
<td>read-only</td>
<td>Returns product name.</td>
<td>■</td>
</tr>
<tr>
<td>productDesc</td>
<td>1.3.6.1.4.1.8691.17.1.1.2</td>
<td>read-only</td>
<td>Returns product short description.</td>
<td>■</td>
</tr>
<tr>
<td>productVersion</td>
<td>1.3.6.1.4.1.8691.17.1.1.3</td>
<td>read-only</td>
<td>Returns product version.</td>
<td>■</td>
</tr>
<tr>
<td>productBuildDate</td>
<td>1.3.6.1.4.1.8691.17.1.1.4</td>
<td>read-only</td>
<td>Returns product last build date, the format is YYMMDDHH.</td>
<td>■</td>
</tr>
<tr>
<td>biosVersion</td>
<td>1.3.6.1.4.1.8691.17.1.1.4.1</td>
<td>read-only</td>
<td>Returns the BIOS version.</td>
<td>■</td>
</tr>
<tr>
<td>biosSaveSetting</td>
<td>1.3.6.1.4.1.8691.17.1.1.4.2</td>
<td>read-write</td>
<td>Write 1 to save bios setting, and read 0 mean setting had been applied.</td>
<td>■</td>
</tr>
<tr>
<td>biosSettingStatus</td>
<td>1.3.6.1.4.1.8691.17.1.1.4.3</td>
<td>read-only</td>
<td>Returns compare of bios CMOS setting and bios new setting.</td>
<td>■</td>
</tr>
<tr>
<td>bootDeviceStatus</td>
<td>1.3.6.1.4.1.8691.17.1.1.4.4.1</td>
<td>read-only</td>
<td>Returns the current support boot device.</td>
<td>■</td>
</tr>
<tr>
<td>firstBootDevice</td>
<td>1.3.6.1.4.1.8691.17.1.1.4.4.2</td>
<td>read-write</td>
<td>read show current first boot device, write set boot device.</td>
<td>■</td>
</tr>
<tr>
<td>pwrOnAfterPwrFail</td>
<td>1.3.6.1.4.1.8691.17.1.1.4.8.1</td>
<td>read-write</td>
<td>Select power on after power fail behavior.</td>
<td>■</td>
</tr>
<tr>
<td>pwrLanWakeUp</td>
<td>1.3.6.1.4.1.8691.17.1.1.4.8.3</td>
<td>read-write</td>
<td>Enable/Disable wake on LAN functionality.</td>
<td>■</td>
</tr>
<tr>
<td>tempSensorsIndex</td>
<td>1.3.6.1.4.1.8691.17.1.5.1.1.1.1</td>
<td>read-only</td>
<td>Reference index for each observed device.</td>
<td>■</td>
</tr>
<tr>
<td>tempSensorsDevice</td>
<td>1.3.6.1.4.1.8691.17.1.5.1.1.1.2</td>
<td>read-only</td>
<td>The name of the temperature sensor we are reading.</td>
<td>■</td>
</tr>
<tr>
<td>tempSensorsValue</td>
<td>1.3.6.1.4.1.8691.17.1.5.1.1.1.3</td>
<td>read-only</td>
<td>The temperature of this sensor in mC.</td>
<td>■</td>
</tr>
<tr>
<td>voltSensorsIndex</td>
<td>1.3.6.1.4.1.8691.17.1.5.1.2.1.1</td>
<td>read-only</td>
<td>Reference index for each observed device.</td>
<td>■</td>
</tr>
<tr>
<td>voltSensorsDevice</td>
<td>1.3.6.1.4.1.8691.17.1.5.1.2.1.2</td>
<td>read-only</td>
<td>The name of the device we are reading.</td>
<td>■</td>
</tr>
<tr>
<td>voltSensorsValue</td>
<td>1.3.6.1.4.1.8691.17.1.5.1.2.1.3</td>
<td>read-only</td>
<td>The voltage in mV.</td>
<td>■</td>
</tr>
<tr>
<td>ioDiNumber</td>
<td>1.3.6.1.4.1.8691.17.1.6.1.1.1.1</td>
<td>read-only</td>
<td>Number of digital input pin in current system.</td>
<td>■</td>
</tr>
<tr>
<td>diIndex</td>
<td>1.3.6.1.4.1.8691.17.1.6.1.1.2.1.1</td>
<td>read-only</td>
<td>Reference index for each digital input pin.</td>
<td>■</td>
</tr>
<tr>
<td><strong>Object</strong></td>
<td><strong>OID</strong></td>
<td><strong>Type</strong></td>
<td><strong>Description</strong></td>
<td></td>
</tr>
<tr>
<td>------------------</td>
<td>----------------------------------------------</td>
<td>------------</td>
<td>---------------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td><code>diPort</code></td>
<td>1.3.6.1.4.1.8691.17.1.6.1.2.1.2</td>
<td>read-only</td>
<td>The port number of digital input pin.</td>
<td></td>
</tr>
<tr>
<td><code>diValue</code></td>
<td>1.3.6.1.4.1.8691.17.1.6.1.2.1.3</td>
<td>read-only</td>
<td>The digital input status, 0 is low, 1 is high.</td>
<td></td>
</tr>
<tr>
<td><code>diTrapEnable</code></td>
<td>1.3.6.1.4.1.8691.17.1.6.1.2.1.4</td>
<td>read-write</td>
<td>Agent will send trap message when digital input pin status changed and this object enbeled.</td>
<td></td>
</tr>
<tr>
<td><code>ioDoNumber</code></td>
<td>1.3.6.1.4.1.8691.17.1.6.1.3</td>
<td>read-only</td>
<td>Number of digital output pin in current system.</td>
<td></td>
</tr>
<tr>
<td><code>doIndex</code></td>
<td>1.3.6.1.4.1.8691.17.1.6.1.4.1.1</td>
<td>read-only</td>
<td>Reference index for each digital output pin.</td>
<td></td>
</tr>
<tr>
<td><code>doPort</code></td>
<td>1.3.6.1.4.1.8691.17.1.6.1.4.1.2</td>
<td>read-only</td>
<td>The port number of digital output pin.</td>
<td></td>
</tr>
<tr>
<td><code>doValue</code></td>
<td>1.3.6.1.4.1.8691.17.1.6.1.4.1.3</td>
<td>read-write</td>
<td>The digital output status, 0 is low, 1 is high.</td>
<td></td>
</tr>
<tr>
<td><code>ledNumber</code></td>
<td>1.3.6.1.4.1.8691.17.1.6.2.1</td>
<td>read-only</td>
<td>Number of LED in current system</td>
<td></td>
</tr>
<tr>
<td><code>ledIndex</code></td>
<td>1.3.6.1.4.1.8691.17.1.6.2.2.1.1</td>
<td>read-only</td>
<td>Reference index for each LED.</td>
<td></td>
</tr>
<tr>
<td><code>ledPort</code></td>
<td>1.3.6.1.4.1.8691.17.1.6.2.2.1.2</td>
<td>read-only</td>
<td>The port number of LED.</td>
<td></td>
</tr>
<tr>
<td><code>ledValue</code></td>
<td>1.3.6.1.4.1.8691.17.1.6.2.2.1.3</td>
<td>read-write</td>
<td>The LED status, 0 is low, 1 is high.</td>
<td></td>
</tr>
<tr>
<td><code>uartNumber</code></td>
<td>1.3.6.1.4.1.8691.17.1.6.3.1</td>
<td>read-only</td>
<td>Number of internal UART in current system.</td>
<td></td>
</tr>
<tr>
<td><code>uartIndex</code></td>
<td>1.3.6.1.4.1.8691.17.1.6.3.2.1.1</td>
<td>read-only</td>
<td>Reference index for each UART port.</td>
<td></td>
</tr>
<tr>
<td><code>uartType</code></td>
<td>1.3.6.1.4.1.8691.17.1.6.3.2.1.2</td>
<td>read-write</td>
<td>The UART mode, 0 is RS232, 1 is RS485 2 wires, 2 is RS422, 3 is RS485 4 wires.</td>
<td></td>
</tr>
<tr>
<td><code>usbNumber</code></td>
<td>1.3.6.1.4.1.8691.17.1.6.4.1.1</td>
<td>read-only</td>
<td>The number of ports regardless of their current state in the usb general port table.</td>
<td></td>
</tr>
<tr>
<td><code>usbDeviceIndex</code></td>
<td>1.3.6.1.4.1.8691.17.1.6.4.1.3.1.1</td>
<td>read-only</td>
<td>The index is dentical to usbPortIndex for the correspondent USB port.</td>
<td></td>
</tr>
<tr>
<td><code>usbDeviceVendorID</code></td>
<td>1.3.6.1.4.1.8691.17.1.6.4.1.3.1.2</td>
<td>read-only</td>
<td>The USB device port vendor HEX-formatted string as it is provided to the USB host by the USB device.</td>
<td></td>
</tr>
<tr>
<td><code>usbDeviceProductID</code></td>
<td>1.3.6.1.4.1.8691.17.1.6.4.1.3.1.3</td>
<td>read-only</td>
<td>The product ID HEX-formatted string as</td>
<td></td>
</tr>
<tr>
<td>OID</td>
<td>Description</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>usbDeviceActiveClass</td>
<td>It is provided to the USB host by the USB device.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>usbPlugTrapEnable</td>
<td>Agent will send trap message when USB device inserted or removed and this object enabled.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>moxaSystemTrapIP</td>
<td>Set Trap IP address.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>moxaSystemTrapCommunity</td>
<td>Trap community.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **usbDeviceActiveClass**: 1.3.6.1.4.1.8691.17.1.6.4.1.3.1.4 (read-only) - This object returns USB Device Class type of the active configuration.

- **usbPlugTrapEnable**: 1.3.6.1.4.1.8691.17.1.6.4.1.4 (read-write) - Agent will send trap message when USB device inserted or removed and this object enabled.

- **moxaSystemTrapIP**: 1.3.6.1.4.1.8691.17.1.9.1 (read-write) - Set Trap IP address.

- **moxaSystemTrapCommunity**: 1.3.6.1.4.1.8691.17.1.9.2 (read-write) - Trap community.
Sample Code for DA-682A Customization

This section gives you a hard copy of the sample code included with the DA-682A. These short programs are intended to be used either as standalone scripts, or to be included in scripts created to build custom features for end users.

The following topics are covered in this appendix:

- LED Controls
- Watchdog Controls
LED Controls

Below we reproduce the LED control code. This is a simple code that users may freely modify this code to suit their needs.

```c
/* Copyright (C) MOXA Inc. All rights reserved. 
   This software is distributed under the terms of the 
   MOXA License. See the file COPYING-MOXA for details. 
*/

#include "stdafx.h"
#include <windows.h>
#include "..\Include\mxdev.h"

#define LED_PORT_NUMBER 8

int _tmain(int argc, _TCHAR* argv[])
{
    HANDLE hLedDev;
    int port_no, data;
    int nLED = 0;
    int nRet = 0;
    int port = 0;
    int mode = 0;
    int n = 0;
    _TCHAR sin;
    _TCHAR smode;

    _tprintf(_T("LED Test Program\n"));
    _tprintf(_T("(0) Exit Program\n"));
    _tprintf(_T("(1) Display LED\n"));
    _tprintf(_T("(2) Set LED value\n"));

    sin = _gettchar();
    n = _tstoi(&sin);

    do
    {
        switch (n)
        {
            case 1:
                // if char == '1', display the LED output status
                case 1:
                    // Open device
                    hLedDev = mxgpio_open();
                    for (int i = 0; i < LED_PORT_NUMBER; i++)
                    {
                        // Get digital input
                        port_no = i;
                        nLED = mxgpio_get_data(hLedDev, port_no);
                        _tprintf(_T("LED[%d] = %d\n"), port_no, nLED);
                    }
                    break;
        }
    } while (n != 0);
}
```
// Close device
mxgpio_close(hLedDev);
break;

// if char == '2', Set the LED output
case 2:
    // Get Port Number
    _gettchar();
    _tprintf(_T("Input the Port Number (0 ~ %d) = \r\n"), LED_PORT_NUMBER-1);
    smode = _gettchar();
    port_no = _tstoi(&smode);

    // Get Value
    _gettchar();
    _tprintf(_T("Input the value (0 or 1) = "));
    smode = _gettchar();
    data = _tstoi(&smode);

    // Open device
    hLedDev = mxgpio_open();

    // Set LED
    nRet = mxgpio_set_data( hLedDev, port_no, data);
    if ( nRet == -1 )
    {
        _tprintf(_T("Set led signal fail!\r\n"));
    }
    else
    {
        _tprintf(_T("Set led signal success!\r\n"));
    }

    // Close device
    mxgpio_close(hLedDev);
    break;
    _gettchar();
    sin = _gettchar();
    n = _tstoi(&sin);
} while (n != 0);
return 0;
Watchdog Controls

/* Copyright (C) MOXA Inc. All rights reserved.  
   This software is distributed under the terms of the 
   MOXA License. See the file COPYING-MOXA for details. 
*/

#include "stdafx.h"
#include <windows.h>
#include '..\Include\mxdev.h'

int _tmain(int argc, _TCHAR* argv[])
{
    PVOID fd; // Handle to device, obtain from mxwdg_open
    ULONG time;

    // starts watchdog timer
    time = 10;
    fd = mxwdg_open(time);

    while ( TRUE )
    {
        _tprintf(_T("Press \"ENTER\" in 10 seconds\n, 'q' to exit"));
        TCHAR ch = _gettchar();
        if ( ch == (TCHAR)'q' )
        {
            break;
        }

        // refresh watchdog timer
        mxwdg_refresh(fd);
    }

    // stops watchdog timer
    mxwdg_close(fd);
    return 0;
}