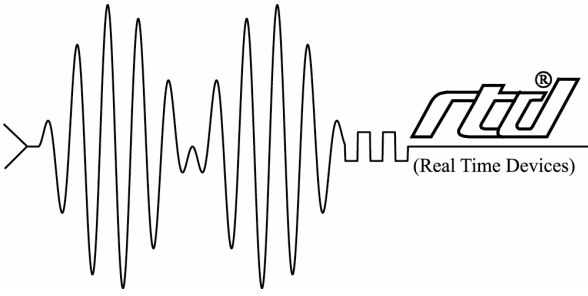
UPS25 API Manual for Windows 2000/XP/Vista

Version 3.0



RTD Embedded Technologies, Inc.
"Accessing the Analog World"®

SWM-640020008
Rev. B

RTD Embedded Technologies, INC.
103 Innovation Blvd.
State College, PA 16803-0906

Phone: +1-814-234-8087
FAX: +1-814-234-5218

E-mail
sales@rtd.com
techsupport@rtd.com

web site
http://www.rtd.com

Revision History

09/23/2004	Revision A issued Documented for ISO9000
10/20/2008	Revision B issued Updated to match version 3.0 of the Windows software Improved layout and section naming

UPS25 API Manual for Windows 2000/XP/Vista
Published by:

RTD Embedded Technologies, Inc.
103 Innovation Blvd.
State College, PA 16803-0906

Copyright 2004 by RTD Embedded Technologies, Inc.
All rights reserved
Printed in U.S.A.

The RTD Logo is a registered trademark of RTD Embedded Technologies. cpuModule and utilityModule are trademarks of RTD Embedded Technologies. PS/2, PC/XT, PC/AT and IBM are trademarks of International Business Machines Inc. MS-DOS, Windows, Windows 98, Windows NT, Windows 2000, Windows XP and Windows Vista are trademarks of Microsoft Corp. PC/104 is a registered trademark of PC/104 Consortium. All other trademarks appearing in this document are the property of their respective owners.

Table of Contents

WINDOWS INSTALLATION **5**

INSTALLING THE WINDOWS SOFTWARE 5

USING THE DRIVER **6**

GETTING A HANDLE TO THE BOARD 6
CATCHING INTERRUPTS 6

IOCTL REFERENCE **7**

BOARD CONTROL IOCTLS 8
UPS25_SET_CONTROL *8*
UPS25_SET_DISCONNECTBATT *8*
UPS25_SET_MODE *9*
UPS25_SET_NODISCONNECTBATT *9*
UPS25_SET_POWER_DOWN_CONTROL *10*
BOARD INFORMATION IOCTLS 11
UPS25_GET_CONTROL *11*
UPS25_GET_INFO *11*
UPS25_GET_STATUS *12*
UPS25_GET_POWER_DOWN_CONTROL *12*

LIMITED WARRANTY **14**

Windows Installation

Installing the Windows Software

Before installing the Windows software, you need to install the UPS25 board in your system. Please follow the instructions in the hardware manual to install the board.

In the software package, you can find the **Setup.exe** program, which installs the UPS25 device driver and monitoring application. After starting the setup program, please follow the instructions on the screen to install the Windows software. You can select the directory where the files will be installed. The setup program adds a 'RTD\UPS25' folder to the 'Programs' section of the 'Start menu'. This folder contains shortcuts to the monitor application, Changes.txt and ReadMe.txt. The setup program also automatically installs the driver for a UPS25 device. Once the setup program is complete, make sure to configure the device's resources using 'Device Manager'. 'Device Manager' can be opened by selecting the 'System' tool in 'Control Panel' and then navigating to the 'Hardware' tab and clicking the 'Device Manager' button.

The source code for the monitor application will be installed in the target directory under the 'Application' folder. This folder contains the program source for the UPS25 monitor application. The program is compiled with Microsoft Visual C++ version 6.0.

Uninstallation: Under Windows 2000 or Windows XP, you can uninstall the Windows software using the 'Add/Remove Programs' tool in the 'Control Panel'. Under Windows Vista, you can uninstall the Windows software using the 'Programs and Features' tool in the 'Control Panel'.

See ReadMe.txt for additional information!

Using The Driver

The UPS25 can be controlled from software via the driver's IOCTL interface. Using an IOCTL interface is the standard way of communicating with a Windows driver. This interface is accessed using the' DeviceIOControl' function.

In the application source code you can find different examples of how to use the driver. Some special examples are given below.

Getting a handle to the board

The 'DeviceIOControl' function (which is used to communicate with the board) requires a handle as the first parameter. To get the handle you must open a device file using the following code:

```
HANDLE hDevice;
hDevice = CreateFile(   "\\\\.\\Ups250",
                        GENERIC_READ | GENERIC_WRITE,
                        0,
                        NULL,
                        OPEN_EXISTING,
                        0,
                        NULL);
```

Catching Interrupts

The driver can signal a program when an interrupt is received from the board. To do this, you should use the 'UPS25_SET_MODE' IOCTL:

```
ENABLEINTERRUPTS Msg;
OVERLAPPED o;
ULONG i;

o.hEvent = CreateEvent(NULL, FALSE, FALSE, NULL);

Msg.IntEvent = o.hEvent;

DeviceIoControl(  hDevice,
                  UPS25_SET_MODE,
                  &Msg,
                  sizeof(ENABLEINTERRUPTS),
                  NULL,
                  0,
                  &i,
                  &o);
```

When an interrupt is received from the board, the event 'o.hEvent' will be signaled. Normally, you would have a thread waiting on the event before performing an action. See the 'ServiceThread' function in the application source for how to do this. Also see the 'OpenDevice' function where it creates a thread using 'CreateThread'.

IOCTL Reference

Board Control IOCTLs

UPS25_SET_CONTROL

Description:

This IOCTL sets the current configuration of the board.

Parameters:

Input: An eight bit value:

BIT 0: 0 – Interrupts Disabled
1 – Interrupts Enabled
BIT 2: 0 – Batteries disconnected from board to prevent discharge
1 – Batteries connected to board for backup purposes
BIT 3: 0 – Charger active. Charging can take place
1 – Charger inactive. No charging possible.

Output: None

Example:

```
UCHAR Control;
ULONG BytesReturned;

DeviceIOControl(  hDevice,
                  UPS25_SET_CONTROL,
                  &Control,
                  sizeof(Control),
                  NULL,
                  0,
                  &BytesReturned,
                  NULL);
```

UPS25_SET_DISCONNECTBATT

Description:

This IOCTL tells the driver to disconnect the battery when the system shuts down.

Parameters:

Input: None

Output: None

Example:

```
ULONG BytesReturned;
```

```
DeviceIoControl(  hDevice,
                  UPS25_SET_DISCONNECTBATT,
                  NULL,
                  0,
                  NULL,
                  0,
                  &BytesReturned,
                  NULL);
```

UPS25_SET_MODE

Description:

This IOCTL initializes an event object that will be set when an interrupt occurs on the board.

Parameters:

Input: An ENABLEINTERRUPTS structure.

Output: None

Example:

```
ENABLEINTERRUPTS Msg;
OVERLAPPED o;
ULONG i;

o.hEvent = CreateEvent(NULL, FALSE, FALSE, NULL);

Msg.IntEvent = o.hEvent;

DeviceIoControl(  hDevice,
                  UPS25_SET_MODE,
                  &Msg,
                  sizeof(ENABLEINTERRUPTS),
                  NULL,
                  0,
                  &i,
                  &o);
```

UPS25_SET_NODISCONNECTBATT

Description:

This IOCTL tells the driver not to disconnect the battery when the system shuts down.

Parameters:

Input: None

Output: None

Example:

```
ULONG BytesReturned;
```

```
DeviceIoControl(  hDevice,
                  UPS25_SET_NODISCONNECTBATT,
                  NULL,
                  0,
                  NULL,
                  0,
                  &BytesReturned,
                  NULL);
```

UPS25_SET_POWER_DOWN_CONTROL

Description:

This IOCTL alters the state of the backlight switch.

Parameters:

Input: An 8 Bit value:

BIT 0: 0 – Backlight is disabled.
1 – Backlight is enabled.

Output: None

Example:

```
UCHAR PowerDown;
ULONG BytesReturned;

PowerDown = 1;    // Enable the backlight

DeviceIOControl(  hDevice,
                  UPS25_SET_POWER_DOWN_CONTROL,
                  &PowerDown,
                  sizeof(PowerDown),
                  NULL,
                  0,
                  &BytesReturned,
                  NULL);
```

Board Information IOCTLs

UPS25_GET_CONTROL

Description:

This IOCTL returns the current configuration of the board.

Parameters:

Input: None

Output: An eight bit value:

BIT 0: 0 – Interrupts Disabled
1 – Interrupts Enabled
BIT 2: 0 – Batteries disconnected from board to prevent discharge
1 – Batteries connected to board for backup purposes
BIT 3: 0 – Charger active. Charging can take place
1 – Charger inactive. No charging possible.

Example:

```
UCHAR Control;
ULONG BytesReturned;

DeviceIOControl(  hDevice,
                  UPS25_GET_CONTROL,
                  NULL,
                  0,
                  &Control,
                  sizeof(Control),
                  &BytesReturned,
                  NULL);
```

UPS25_GET_INFO

Description:

This IOCTL gathers information about the UPS25 driver.

Parameters:

Input: None

Output: An 8 bit value:

BIT 2: 0 – Interrupts Disabled. The IRQ resource is absent in the current configuration. If you want to use interrupts you should select a configuration with interrupts.
1 – Interrupts Enabled.

Example:

```
UCHAR Info;
ULONG BytesReturned;

DeviceIOControl(  hDevice,
                  UPS25_GET_INFO,
                  NULL,
                  0,
                  &Info,
                  sizeof(Info),
                  &BytesReturned,
                  NULL);
```

UPS25_GET_STATUS

Description:

This IOCTL gathers the current status of the board.

Parameters:

Input: None

Output: An 8 bit value:

BIT 0: 0 – Main Supply voltage low. System is in backup state
1 – Main power OK.
BIT 1: 0 – Battery Voltage Low.
1 – Battery Voltage OK.
BIT 2: Charger Status LED. See Table 4-1 'Charger status LED functional description' in the hardware manual.
BIT 3: Charger Temperature LED
0 – Battery Temperature out of limits. No charging possible.
1 – Battery Temperature OK. Charging can take place.

Example:

```
UCHAR Status;
ULONG BytesReturned;

DeviceIOControl(  hDevice,
                  UPS25_GET_STATUS,
                  NULL,
                  0,
                  &Status,
                  sizeof(Status),
                  &BytesReturned,
                  NULL);
```

UPS25_GET_POWER_DOWN_CONTROL

Description:

This IOCTL returns the current status of the backlight switch.

Parameters:

Input: None

Output: An 8 bit value:

BIT 0: 0 – Backlight is disabled.
1 – Backlight is enabled.

Example:

```
UCHAR PowerDown;
ULONG BytesReturned;

DeviceIOControl(  hDevice,
                  UPS25_GET_POWER_DOWN_CONTROL,
                  NULL,
                  0,
                  &PowerDown,
                  sizeof(PowerDown),
                  &BytesReturned,
                  NULL);
```

Limited Warranty

RTD Embedded Technologies, Inc. warrants the hardware and software products it manufactures and produces to be free from defects in materials and workmanship for one year following the date of shipment from RTD Embedded Technologies, INC. This warranty is limited to the original purchaser of product and is not transferable.

During the one year warranty period, RTD Embedded Technologies will repair or replace, at its option, any defective products or parts at no additional charge, provided that the product is returned, shipping prepaid, to RTD Embedded Technologies. All replaced parts and products become the property of RTD Embedded Technologies. Before returning any product for repair, customers are required to contact the factory for an RMA number.

THIS LIMITED WARRANTY DOES NOT EXTEND TO ANY PRODUCTS WHICH HAVE BEEN DAMAGED AS A RESULT OF ACCIDENT, MISUSE, ABUSE (such as: use of incorrect input voltages, improper or insufficient ventilation, failure to follow the operating instructions that are provided by RTD Embedded Technologies, "acts of God" or other contingencies beyond the control of RTD Embedded Technologies), OR AS A RESULT OF SERVICE OR MODIFICATION BY ANYONE OTHER THAN RTD Embedded Technologies. EXCEPT AS EXPRESSLY SET FORTH ABOVE, NO OTHER WARRANTIES ARE EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, ANY IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, AND RTD Embedded Technologies EXPRESSLY DISCLAIMS ALL WARRANTIES NOT STATED HEREIN. ALL IMPLIED WARRANTIES, INCLUDING IMPLIED WARRANTIES FOR MECHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, ARE LIMITED TO THE DURATION OF THIS WARRANTY. IN THE EVENT THE PRODUCT IS NOT FREE FROM DEFECTS AS WARRANTED ABOVE, THE PURCHASER'S SOLE REMEDY SHALL BE REPAIR OR REPLACEMENT AS PROVIDED ABOVE. UNDER NO CIRCUMSTANCES WILL RTD Embedded Technologies BE LIABLE TO THE PURCHASER OR ANY USER FOR ANY DAMAGES, INCLUDING ANY INCIDENTAL OR CONSEQUENTIAL DAMAGES, EXPENSES, LOST PROFITS, LOST SAVINGS, OR OTHER DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PRODUCT.

SOME STATES DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES FOR CONSUMER PRODUCTS, AND SOME STATES DO NOT ALLOW LIMITATIONS ON HOW LONG AN IMPLIED WARRANTY LASTS, SO THE ABOVE LIMITATIONS OR EXCLUSIONS MAY NOT APPLY TO YOU.

THIS WARRANTY GIVES YOU SPECIFIC LEGAL RIGHTS, AND YOU MAY ALSO HAVE OTHER RIGHTS WHICH VARY FROM STATE TO STATE.

RTD Embedded Technologies, Inc.
103 Innovation Blvd.
State College PA 16803-0906
USA
Our website: www.rtd.com