

NUC-UPS

6-48V Intelligent Automotive grade Uninterruptible Power Supply

Before you start...

Often times, rushing into installing the unit can result in serious damage to your DCDC-NUC UPS board or computer. Always double check the polarity of your wires with a voltmeter.

Always double check the polarity of the batteries before introducing into the UPS device. Improper configuration leads to system damage!!!

Contents

- 1 Introduction
- 2 Features
- 3 Diagram & Schematics
- 4 Connectors
 - 4.1 Power Input connectors
 - 4.2 Interface connectors
- 5 NUC and OS Settings
- 6 Bootloader Mode
- 7 Parameter List
- 8 Software manual
 - 8.1 Windows OS built-in support
 - 8.2 Configuration software
 - 8.2.1 The first main screen is the "Status-UPS"
 - 8.2.2 The second main screen is the "Status-Charger"
 - 8.2.3 The third main screen is the "Settings"
 - 8.3 Windows System monitor
 - 8.4 Download software
 - 8.5 Developer manual
 - 8.5.1 Visual C++ Example
 - 8.5.2 Visual Basic Example
 - 8.5.3 Visual C# Example
 - 8.6 Download API and example projects

Introduction

The NUC-UPS was designed to provide +12V regulated power output from a wide input voltage(6V-48V). It has a range of intelligent functions not found in any of the traditional USB converters.

The unit is able to gracefully shut down motherboards based on Ignition sensing or battery voltage level, by pulsing the motherboard on/off pins. This makes it an ideal device for automotive or battery powered installations.

It features 4 replaceable Li-Ion batteries for battery backup integrated in the device, there is no need for any kind of special cabling between cells and UPS. The charger is a Synchronous Buck-Boost converter capable of charging the batteries with input voltage below, equal or above battery voltage level. It charges the battery in

three phases: preconditioning, constant current mode and constant voltage mode.

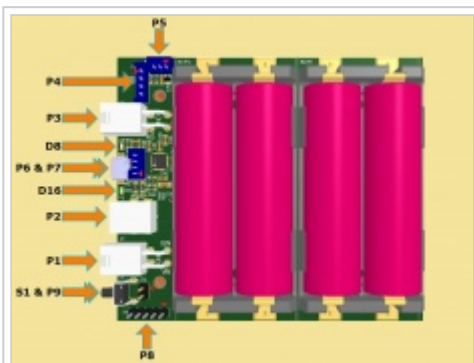
The output converter is a synchronous Buck converter providing regulated 12V output. The unit can be used as an embedded UPS or as a Portable Energy Pack providing the user with regulated 12V wherever needed.

Features

- USB interface, works with Windows devices (Linux API planned)
- Input between 6-48V
- Programmable thresholds, timings and configuration modes
- Generates regulated 12V output voltage
- Output current 5A continuous, 6A peak
- Supports Li-Ion (3.7V) battery chemistry
- 3 state charger(precharge, constant current, constant voltage)
- Battery balancing, 4S configuration.
- Charge voltage 4x4.2V, charge current 1A, precharge current ~0.3A
- Coulomb counting
- Battery temperature monitoring for each cell and temperature compensated charge
- Start/Stop button and header
- Motherboard ON/OFF pulse control*
- Motherboard detection by power consumption or 5VUSB presence
- Missing battery cells detection
- Ultra low power consumption(5uA- in Deep Sleep mode) or (250uA- in Standby mode)
- Windows detects device as “Battery”, with no special drivers required **

Diagram & Schematics

Connectors



Connectors nomenclature

Power Input connectors

P1 Input (mini-FIT JR, 1x4 pins)

P1.1,P1.2-GND

P1.3-Ignition sensing

P1.4-Input voltage

P2 Input (DC Power Jack

2DC-G213-B49, Mating plug 5.50x2.50)

P3 Output (mini-FIT JR, 1x4 pins)

P3.1,P3.2-GND

P3.3,P3.4-Output voltage

P4 Output (B4B-EH-A(LF)(SN), 1x4 pins)

P4.1,P4.2-GND

P4.3,P4.4-Output voltage

Interface connectors

P5 Motherboard POWER SW connection, no polarity (JST_B3B-PH-KL, 1x3pins)

P5.1: SW1

P5.2: GND

P5.3: SW2

P7 USB header (B4B-PH-K-S(LF)(SN), 1x4pins)

P7.1: GND

P7.2: USB D-

P7.3: USB D+

P7.4: +5V

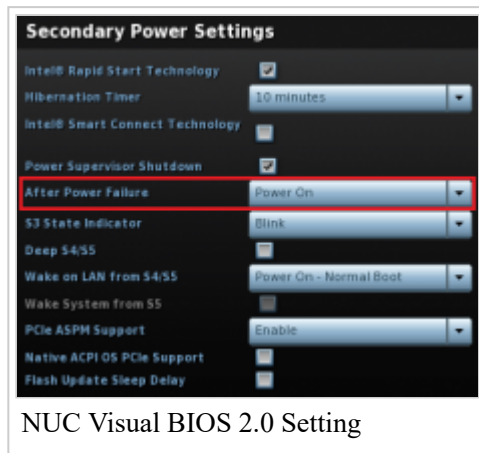
P10 Start/Stop button header (M20-9990246, 1x2pins)

P8 Programming header

P8.1: MCLR P8.2: GND P8.3: 5V P8.4: PGD P8.5: PGC

NUC and OS Settings

■ NUC related settings



Setting up the NUC's behavior when power is re-applied:

- press F2 key during the boot sequence to entering the NUC's BIOS (Visual BIOS 2.0)

- first click on Advanced then click on Power menu button

- in Secondary Power Settings section select Power On option for After Power Failure

- the picture shows the right option in a red framework

- now the power supply should be able to START the NUC by applying corresponding voltage to its power input

Bootloader Mode

It is recommended that you only connect USB power to the device when making a firmware update via "bootloader mode".

There are two ways to enter in bootloader mode: 1a) or 1b).

1a) Place a jumper on the P8.4 and P8.5 pins then connect the unit to USB

1b) Connect the unit to USB then press the Switch to bootloader button on the configuration software's user interface.

2) Start the HIDBootloader v2.9j.exe

software provided to flash the new firmware.

3) Press File->Import Firmware Image

4) Locate the hex file and open it.

5) Press Program->Erase/Program/Verify Device

6) If used option 1a) remove the jumper 6) Press Program->Reset Device

7) After the device reconnects on USB with the configuration software the new firmware version will be displayed in the title bar of the software.

Parameter List

NAME	Description
NUC_UPS_MODE[-]	<div style="border: 1px dashed black; padding: 5px;"> <p data-bbox="715 181 847 208">0-Dumb mode (DUMB)</p> <p data-bbox="715 210 919 237">1-Automotive mode (AUTOMOTIVE)</p> </div> <p data-bbox="715 293 970 320">NUC_UPS_MODE</p>
UPS_ON_BATTERY_TOUT	<p data-bbox="715 412 1506 555">Deep sleep timeout. If system is running on battery and this period has elapsed with no input power applied, the UPS will initiate shut down procedure. "Never" is allowed as value (to disable this feature). Default is "Never".</p>
INIT_TOUT	<p data-bbox="715 638 1493 745">When all power supply start-up conditions are met, the PSU will wait this time before continuing with the start-up sequence.</p>
VIN_MIN_STARTUP	<p data-bbox="715 828 1449 936">If the input voltage is beyond this threshold and all other start-up conditions are met, the PSU can start. Default is 11000mV.</p>
VIN_MIN_RUNNING	<p data-bbox="715 1019 1509 1090">If input voltage is below this threshold, the output is powered from battery. Default is 8000mV.</p>
VIN_COUNT	<p data-bbox="715 1176 1241 1202">Input voltage filtering. Default is 100ms.</p>
IGN_COUNT	<p data-bbox="715 1288 1273 1314">Ignition voltage filtering. Default is 100ms.</p>
IGN_HIGH_THRESHOLD	<p data-bbox="715 1400 1410 1471">If ignition voltage is beyond this threshold, ignition is considered to be ON. Default is 6000mV.</p>
IGN_LOW_THRESHOLD	<p data-bbox="715 1556 1394 1628">If ignition voltage is below this threshold, ignition is considered to be OFF. Default is 5000mV.</p>
IGN_ON_TO_OUTPUT_ON_TOUT	<p data-bbox="715 1713 1490 1785">After ignition is considered ON, the PSU will wait this time before the output is turned ON. Default is 2000ms.</p>
IGN_CANCEL_TOUT	<p data-bbox="715 1870 1497 1942">After the motherboard boots up, the ignition voltage sensing will be disabled for this period. Default is 60s.</p>
OUTPUT_ON_TO_MOB_PULSE_ON_TOUT	<p data-bbox="715 2027 1503 2134">After the output was turned on, the UPS will wait this period before sending the ON pulse to the motherboard. Default is 20ms</p>

IGN_OFF_TO_MOB_PULSE_OFF_TOUT	In automotive mode, after the ignition is considered OFF, this UPS will wait this period before sending the OFF pulse to the motherboard. Default is 5s
MOB_PULSEWIDTH	The length of the start-up/shutdown pulse sent to the motherboard. Default is 500ms.
HARD_OFF_TOUT	After the shutdown pulse is sent to the motherboard, the UPS will wait this period before the output is turned OFF. This time-out allows the operating system to perform a clean shutdown. Default is 60s.
PWM_SPREAD_MODULATING_FREQ	Modulation frequency parameter of the Random Spread Frequency Modulation module (used for EMI reduction purposes)
PWM_SPREAD_PERCENT	Frequency deviation parameter of the Random Spread Frequency Modulation module (used for EMI reduction purposes)
PWM_FREQ	Operating frequency of the Switched Mode Power Supply
CONFIG1	<p>Configuration register [b7..b0]. Used for enabling/disabling modules. 0 - disabled, 1 - enabled</p> <ul style="list-style-type: none"> b7 - Unused b6 - Unused b5 - Unused b4 - Unused b3 - If set, the USB sense is enabled, +5V USB is used to detect the Motherboard alive status b2 - If set, the Power measurement (Vout x Iout) is used to detect the Motherboard alive status b1 - If set, shutdown pulse is enabled through the PWRSW connector. b0 - If set, startup pulse is enabled through the PWRSW connector. <p>Depending motherboard alive status the shutdown/startup pulse is sent if b2/b3 is enabled. The motherboard presence can be smartly detected if b2/b3 is enabled. For example: In case motherboard is already shut down the pulse is not sent in case of a shutdown scenario. This is also true in case of a startup scenario.</p>
CONFIG2	Configuration register. Used for enabling/disabling modules. 0 - disabled, 1 - enabled

	b7..b0 - Reserved/Unused
WRITE_COUNT	The number of times the internal Flash program memory has been written.
IOUT_OFFSET	This value will be summed with the measured input current, this is a calibration parameter. Default is 0mA
CAPACITY	Battery Capacity. Default is 1000mAh.
UPS_BUTTON_ON_TOUT	Filtering time for button pressing. Default is 100ms.
UPS_VCELL_MIN_START	If Vin is not present and all the battery cells are above this threshold, the UPS can start and will run on battery. Default is 3.6V.
UPS_VCELL_MIN_STOP	If VIN is not present and one of the battery cells is less than this threshold , the UPS will initiate a shutdown procedure. Default is 3.3V.
UPS_VCELL_MIN_HARD_STOP	If VIN is not present and one of the battery cells is less than this threshold, the UPS will instantly shut down to preserve battery. Default is 3V.
UPS_OVERLOAD	Max allowed discharge current. In case the discharge current exceeds this threshold, shutdown will be initiated. Default is 6000mA
DCHG_TEMP_COLD	COLD temperature threshold for discharge. In case exceeded UPS will shutdown when on battery. Default is -20°C (FW ver 1.1)
DCHG_TEMP_HOT	HOT temperature threshold for discharge. In case exceeded UPS will shutdown when on battery. Default is 60°C (FW ver 1.1)
CHG_COND_TOUT	Conditioning/Precharge time. Charge current is limited until cell voltage exceeds CHG_VCOND value and it is applied during this time. Default is 30 sec.
CHG_BULK_STOP_VOLTAGE	

	Maximum allowed bulk charge voltage/cell during constant current/constant voltage charging. Default is 4.1 [V/cell].
CHG_HYSTERESIS	An overvoltage value (CHG_BULK_STOP_VOLTAGE + CHG_HYSTERESIS) that is allowed when charging. If one of the cells exceeds this value, charging is immediately stopped. Default is 100mV/cell.
CHG_START_VOLTAGE	If cell voltage is below this value, charging can be started. Default is 3.85V/cell.
CHG_GLOBAL_TOUT	Global charge timeout. Default is 240 min.
CHG_TOPPING_TOUT	After an overvoltage condition is detected for a cell (for Lithium based batteries), a resting period is set by this timer before applying a small topping charge - in case other cells are still not charged. Default is 60s.
CHG_TEMP_COLD	COLD temperature threshold for temperature compensated charge current regulation. Default 5°C(FW ver 1.0)
CHG_TEMP_COOL	COOL temperature threshold for temperature compensated charge current regulation. Default 10°C(FW ver 1.0)
CHG_TEMP_WARM	WARM temperature threshold for temperature compensated charge current regulation. Default 50°C(FW ver 1.0)
CHG_TEMP_HOT	HOT temperature threshold for temperature compensated charge current regulation. Default 55°C(FW ver 1.0)
UPS_VCELL_ADC_OFFSET	This value will be summed with the measured cell voltages, this is a calibration parameter. Default is 0mV.
CHG_CELL_VCOND	Conditioning/Pre-charge voltage. Charge current is limited until cell voltage exceeds this value and for at least CHG_TCOND time . Default is 3.3V
BAL_VCELL_MIN	Balancing is allowed if cell voltages are above this value. Default is 3.4V
BAL_VCELL_DIFF_START	If the voltage difference between cells exceeds this value, cell balancing will start. Default is 80mV.

BAL_VCELL_DIFF_STOP	If the voltage difference between cells is less than this value, cell balancing will stop. Default is 40mV.
OCV_SOC0	Open Circuit Voltage State Of Charge detection for initial 0% fuel gauge estimation. Default is 3.3V.
OCV_SOC10	Open Circuit Voltage State Of Charge detection for initial 10% fuel gauge estimation. Default is 3.68V.
OCV_SOC25	Open Circuit Voltage State Of Charge detection for initial 25% fuel gauge estimation. Default is 3.76V.
OCV_SOC50	Open Circuit Voltage State Of Charge detection for initial 50% fuel gauge estimation. Default is 3.82V.
OCV_SOC75	Open Circuit Voltage State Of Charge detection for initial 75% fuel gauge estimation. Default is 3.97V.
OCV_SOC100	Open Circuit Voltage State Of Charge detection for initial 100% fuel gauge estimation. Default is 4.17V.
POUT_HIGH_THRESHOLD	If output power measured is higher than this threshold the connected motherboard is considered to be ON. Together with POUT_LOW_THRESHOLD parameter sets a hysteresis for motherboard status. Default is 3000mW.
POUT_LOW_THRESHOLD	If output power measured is lower than this threshold the connected motherboard is considered to be OFF. Together with POUT_HIGH_THRESHOLD parameter sets a hysteresis for motherboard status. Default is 1000mW.

Software manual

Windows OS built-in support

The NUC-UPS implements two generic USB classes, therefore most of the operating systems are recognizing it without any additional driver installation. This two classes are:

a.) "HID UPS Battery" for OS built-in communication.

Microsoft Windows versions are recognizing this class automatically and the device should appear in device manager at Batteries as a "Hid ups battery", together with all settings and power plan possibilities provided by the OS for any battery.

b.) "HID" for communication and configuration

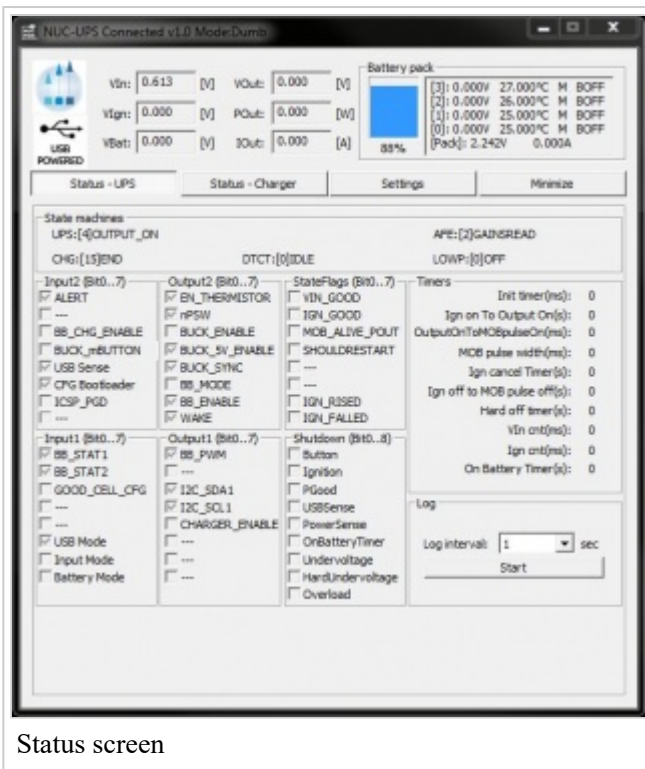
This endpoint is a generic HID and it is used by our configuration and monitor software to read/write voltages and other important parameters of the NUC-UPS. This endpoint does not need any additional driver either, Microsoft Windows OS should recognize it as a generic HID USB device.

Configuration software

The configuration software provides interface for NUC-UPS monitoring, logging and setup. It's recommended to be used by users with deeper understanding of the NUC-UPS hardware since permits setting voltages, currents and other parameters which can be dangerous if they are set without precaution. The configuration software has three main screens (Status-UPS, Status-Charger and Settings) and a header with the important voltage/current values.

The first main screen is the "Status-UPS"

Example of this screen is shown in the next image:



Status screen

The title bar shows the connection status, the firmware version and the mode of the NUC-UPS. Example: "NUC-UPS Connected v1.0 Mode: Dumb"

The header of the status screen contains:

- VIn: Input Voltage
- VIgn: Ignition Voltage
- VBat: Battery Voltage
- VOut: Output Voltage
- POut: Output Power
- IOOut: Output Current
- Battery pack %: Battery percentage
- [0]: Battery 0 status
- [1]: Battery 1 status
- [2]: Battery 2 status
- [3]: Battery 3 status
- [Pack]: Battery pack overall voltage and current

The "Status-UPS" screen contains extended information about the current state of the NUC-UPS like internal state machine, voltages, currents, temperature, different read only state flags. The user also has the possibility to

log the current state into a *.csv file in the "Log" section.

The second main screen is the "Status-Charger"

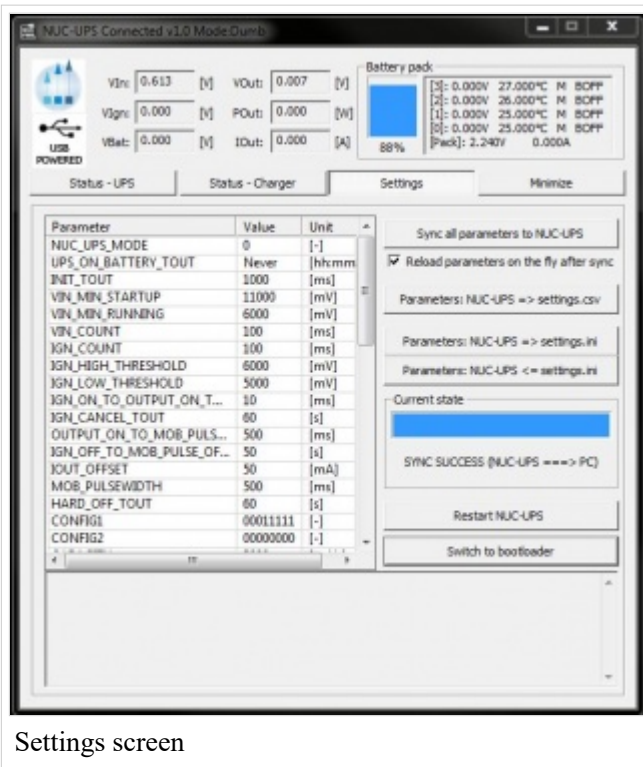
Contains extended information about current state of the NUC-UPS battery charger (internal state machines, flags)



Status screen

The third main screen is the "Settings"

Example of this screen is shown in the next image:



Settings screen

This screen contains two main sections: the individual parameter setup for experienced users and the parameter save/load into/from file section.

The main section of the "Settings" screen is the individual parameter settings.

This is recommended to be done only by experienced users. Any parameter of the NUC-UPS can be set from here.

Changing one parameter is simple:

- select the desired parameter from the "Parameter" list (simple click to select, double-click to edit). Below the parameter list a helper text is displayed (same from this manual).
- after double click introduce the new value in the new popup dialog and press OK
- the introduced value is checked - if something is wrong (out of limit, bad value etc.) error message will be shown
- the ! sign will blink on the "Sync all parameters to the NUC-UPS" button to show edited but not saved/synced

variables

- after You have done with all parameter setting press the "Sync all parameters to the NUC-UPS" button to send all values to the NUC-UPS. IMPORTANT: without this step the new values will be lost, nothing is sent to the NUC-UPS!

IMPORTANT: any parameter setting is taken into account by the NUC-UPS in this cases:

- after a full restart either with power cut from all sources (usb, vin)
- hitting the "Restart NUC-UPS" button
- keeping the "Reload parameters on the fly after sync" checked.

Do any parameter change with precaution, check the parameters and wires before applying it!

For users who need to setup more devices with the same NUC-UPS settings, it is recommended to use the save/load parameters buttons. The "Parameters: NUC-UPS ==> File (settings.ini)" button loads a full configuration from the NUC-UPS and saves it to the settings.ini file. You can disconnect the current NUC-UPS from the USB and insert a new one, than press the "Parameters: NUC-UPS <=== File (settings.ini)" button to send the last saved configuration into the new NUC-UPS.

The "Parameters: NUC-UPS ==> CSV File (settings.csv)" button loads all parameters from the connected NUC-UPS and saves it into a csv file. This type of file can be opened by any spreadsheet editor (OpenOffice, Microsoft Excel etc.) and contains the full set of parameters in human readable form.

The "Switch to bootloader" button is intended to be used for firmware updates. After You press this button the NUC-UPS will disconnect, it will switch to bootloader mode and firmware can be updated as described here

Every save/load/sync operation on the "Settings" screen affects the progress bar and the status bar on the bottom of the screen (labelled with "State:"). In rare cases You might get error here with "try again" message. This happens in case of one parameter byte get's corrupted or timeout occurs during USB communication and/or NUC-UPS flashing operation. Please try again and contact our support team only if the device gives this error 4-5 times in a row.

Windows System monitor

The system monitor is a tray bar software which shows the current state on the tray bar icon and a semi transparent "always on top" capable small window.

The popup window can be moved anywhere on the screen and can be customized. Our current setup has two skins but any combination is possible playing with the "skin*.mbs" files installed together with this application. The current skin can be selected right clicking on the tray icon. The "skin*.mbs" files are simple text ones editable with any text editor (notepad for example).

Adding a new skin is pretty simple – make a skin1.mbs (use the existing skin0.mbs for starting content) and start playing with the values from the new file.

The values are self explanatory – skin name, background image files, font descriptions and label/value pair coordinates for all the important NUC-UPS values. The size of the popup is defined by the background image – transparent parts can be defined as well (see for example: "bubble1.bmp"). Example screenshots:

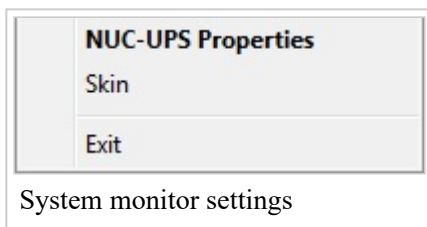


System
monitor skin1
connected



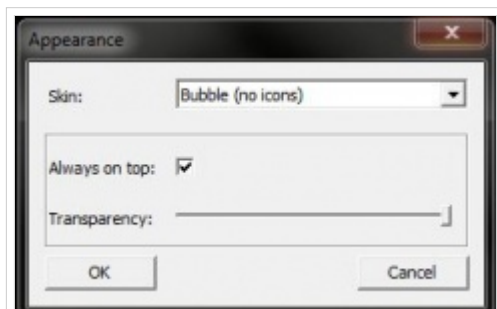
System monitor
skin1
disconnected

Right clicking on the tray icon will pop-up a simple menu:



System monitor settings

From which You can see firmware version and state of the NUC-UPS from the NUC-UPS Properties and set some visual parameters of the application (transparency, skin) from Skin:



System monitor settings

For auto start with the system make a shortcut of AppTray.exe from the standalone package in the system Startup (Windows 7 (<http://windows.microsoft.com/en-us/windows/run-program-automatically-windows-starts#1TC=windows-7>) , Windows 8 (<http://support.microsoft.com/kb/2806079>)).

Download software

Version	Change list
1.0 (NUCUPSv1.0.zip)	First version
1.1 (NUCUPSv1.1.zip)	Update for Firmware 1.1

Developer manual

CarTFT.com provides one NUC-UPS API in a DLL (NUCUpsLib.dll) and examples in Visual C++, Visual Basic and Basic and Visual C#.

Basic C++/Visual Basic/C# knowledge is needed to use this examples together with the API. The API dll has manifest embedded to permit C# and Visual Basic dynamic load.

The API has a set of functions exported to access the full functionality of the NUC-UPS. This functions are:

```
extern "C" NUCUPSLIB_API const char* STATE_MACHINE_UPS[14]; //state machine UPS
extern "C" NUCUPSLIB_API const char* STATE_MACHINE_AFE[8]; //state machine AFE
extern "C" NUCUPSLIB_API const char* STATE_MACHINE_CHG[16]; //state machine Charger
extern "C" NUCUPSLIB_API const char* STATE_MACHINE_DTCT[16]; //state machine Detect
extern "C" NUCUPSLIB_API const char* STATE_MACHINE_LOWP[4]; //state machine low power states
extern "C" NUCUPSLIB_API unsigned char nucupsOpenDeviceHandler(unsigned int timer); //open
device handler. timer sets the refresh period in miliseconds (4 or 5 messages will be sent in
this period e ending from debug mode) IMPORTANT: the handler can be kept open to notice any
NUC-UPS plugged in
extern "C" NUCUPSLIB_API void nucupsCloseDeviceHandler(); //close device handler
extern "C" NUCUPSLIB_API void getNUCUpsDevicePath(char* path); //Get opened device path @param
path - recommended length 1024, will return empty string if no device opened
extern "C" NUCUPSLIB_API unsigned char isNUCUpsConnected(); //0=not connected, 1=normal
state,2=loading settings from device,3=saving settings from pc,4=saving settings from file
extern "C" NUCUPSLIB_API unsigned char getNUCUpsMode(); //get NUC-UPS mode: 0=Dumb,
1=Automotive
```

```

extern "C" NUCUPSLIB_API unsigned int getNUCUPSInputFlags();//get NUC-UPS input flags
extern "C" NUCUPSLIB_API unsigned int getNUCUPSOutputFlags();//get NUC-UPS output flags
extern "C" NUCUPSLIB_API unsigned int getNUCUPSChargerFlags();//get NUC-UPS charger flags
extern "C" NUCUPSLIB_API unsigned int getNUCUPSStateFlags();//get NUC-UPS state flags
extern "C" NUCUPSLIB_API unsigned int getNUCUPSShutdownFlags();//get NUC-UPS shutdown flags
extern "C" NUCUPSLIB_API float getNUCUPSVIN();//get NUC-UPS Input Voltage
extern "C" NUCUPSLIB_API float getNUCUPSIOut();//get NUC-UPS Output Current
extern "C" NUCUPSLIB_API float getNUCUPSVOut();//get NUC-UPS Output voltage
extern "C" NUCUPSLIB_API float getNUCUPSVBats(int i);//get NUC-UPS battery voltages
extern "C" NUCUPSLIB_API float getNUCUPSTemperature(int i);//get NUC-UPS battery temperature
extern "C" NUCUPSLIB_API unsigned char getNUCUPSCellBalanceOn(int i);//get NUC-UPS balanced
battery (discharged)
extern "C" NUCUPSLIB_API unsigned char getNUCUPSCellDetected(int i);//get NUC-UPS battery
detected
extern "C" NUCUPSLIB_API float getNUCUPSVIgnition();//get NUC-UPS Ignition Voltage
extern "C" NUCUPSLIB_API float getNUCUPSPOut();//get NUC-UPS Output Power
extern "C" NUCUPSLIB_API float getNUCUPSVBat();//get NUC-UPS Bat voltage
extern "C" NUCUPSLIB_API float getNUCUPSVPack();//get NUC-UPS Pack voltage
extern "C" NUCUPSLIB_API float getNUCUPSIChgDchg();//get NUC-UPS Charge/Discharge current
extern "C" NUCUPSLIB_API unsigned int getNUCUPSspecConsts(unsigned int type, unsigned int
index);//get NUC-UPS special constants
extern "C" NUCUPSLIB_API unsigned char getNUCUPSverMajor();//get NUC-UPS major version of the
firmware
extern "C" NUCUPSLIB_API unsigned char getNUCUPSverMinor();//get NUC-UPS minor version of the
firmware
extern "C" NUCUPSLIB_API unsigned char getNUCUPSstate();//0 - not connected, 1 - running from
battery, 2 - from vin, 3 - from USB
extern "C" NUCUPSLIB_API void setNUCUPSDBGMode(unsigned char dbg);//set dbg mode
extern "C" NUCUPSLIB_API unsigned char getNUCUPSdbgByte(int i);//get NUC-UPS debug bytes -
valid only after setNUCUPSDBGMode(1)
extern "C" NUCUPSLIB_API unsigned char getNUCUPSdbg2Byte(int i);//get NUC-UPS debug bytes -
valid only after setNUCUPSDBGMode(1)
extern "C" NUCUPSLIB_API unsigned char getNUCUPSdbg3Byte(int i);//get NUC-UPS debug bytes -
valid only after setNUCUPSDBGMode(1)
extern "C" NUCUPSLIB_API unsigned int getNUCUPStimer(unsigned int cnt);//get NUC-UPS timer
extern "C" NUCUPSLIB_API unsigned int getNUCUPSstateMachines(unsigned int cnt);//get NUC-UPS
internal state machines 0=UPS 1=AFE 2=Charger 3=Detect 4=Batterymanager
extern "C" NUCUPSLIB_API unsigned int getNUCUPSchgTimer(unsigned int cnt);//get NUC-UPS
charger timers
extern "C" NUCUPSLIB_API unsigned char getNUCUPSchargeEndedCondition();//get NUC-UPS charge
ended
extern "C" NUCUPSLIB_API unsigned char getNUCUPSbatteryLevel();//get NUC-UPS battery level
(percents 0-100)
extern "C" NUCUPSLIB_API unsigned int getNUCUPSCPUUsage();//get NUC-UPS CPU usage
extern "C" NUCUPSLIB_API void restartNUCUPS();//restart NUC-UPS
extern "C" NUCUPSLIB_API void restartNUCUPSinBootloaderMode();//restart NUC-UPS in bootloader
mode
extern "C" NUCUPSLIB_API unsigned int getNUCUPSmaxVariableCnt();//get NUC-UPS maximum variable
count
extern "C" NUCUPSLIB_API unsigned char getNUCUPSvariableData(unsigned int cnt, char* name,
char* value, char* unit, char* comment);//get NUC-UPS variable data
extern "C" NUCUPSLIB_API void startNUCUPSloadingSettings(unsigned char to_file, unsigned char
compare_with_old);//start loading data from device
extern "C" NUCUPSLIB_API unsigned char getNUCUPSloadingSettingsState();//get load settings
current state: 0-64 - steps, 100=success, 0xF1-0xFF=failure
extern "C" NUCUPSLIB_API unsigned char setNUCUPSvariableData(unsigned int cnt, char*
value);//set NUC-UPS variable data for a given variable
extern "C" NUCUPSLIB_API void startNUCUPSsaveSettings(unsigned char from_file);//start saving
data to device
extern "C" NUCUPSLIB_API unsigned char getNUCUPSsaveSettingsState();//get saving current
state: 0-64 - steps, 100=success, 0xF1-0xFF=failure

```

See the examples for usage.

IMPORTANT: the API dll needs 4 files from Visual Studio 2005 redistribution pack (Microsoft.VC80.CRT.manifest, msvcm80.dll, msvcp80.dll, msvcr80.dll).

IMPORTANT: the API supports only one NUC-UPS connected to the computer.

Visual C++ Example

Open TestAPI.sln from the package, set CLibTest project as active project, run it and see CLibTest.cpp for usage example.

Visual Basic Example

Open TestAPI.sln from the package, set VBLibTest project as active project, run it and see Module1.vb for usage example.

Visual C# Example

Open TestAPI.sln from the package, set CSLibTest project as active project, run it and see Program.cs for usage example.

Download API and example projects

Version	Change list
1.0 (NUCUPSApiTestV1.0.zip)	First version
1.1 (NUCUPSApiTestV1.1.zip)	Update for Firmware 1.1