Document Version 1.4

Linear Computing Inc.

progressive design

4132 – Short Time UPS (STUPS)

User Manual

Document Number: 41329002

Document History

Document number: 41329002

Version	Date	Who	Change Description
1.0	11/12/2013	BB	Initial version for engineering samples
1.1	1/28/2014	BB	• Modified specification • added description of LEDs and basic connection diagrams • information updates for production version HWNum: 41321201
1.2	2/5/2014	BB	• Updates for firmware version v1.01A • Added Application Consideration paragraph • Added description of hibernate state • Added circuits and description of digital outputs
1.3	4/25/2014	BB	• Updates for firmware version 1.02A • Added mounting hole pattern • Modified StupsViewVB6 • Added StupsView section
1.4		BB	• Updates for firmware 1.03A

Overview

Short Time UPS (STUPS) is a device designed to provide protection against unexpected power loss and consequent potential loss of data or integrity of the system. STUPS was specifically designed to meet the requirements of embedded applications where high reliability of unsupervised systems is required.

The main purpose of the device is to provide **back-up power** to allow the embedded system to finish any active transactions, save data and shut down the operating system. An embedded system equipped with the STUPS is guaranteed to execute proper shutdown sequence every time, when power is accidentally or purposely disconnected. This significantly simplifies power on/off procedures for the system. When STUPS is integrated, it is acceptable for a user to simply flick the on/off switch or unplug the unit if he desires to turn off the system - STUPS will handle the shutdown sequence properly in any situation.

As an option, it is possible to apply delay between the moment power level drops below limit and the moment when system starts shutdown. This makes allowances for short power glitches without any disruption to the service or operation of the embedded system. This feature is beneficial in industrial environments where short power blips are common and low quality of power causes high failure rates of embedded systems.

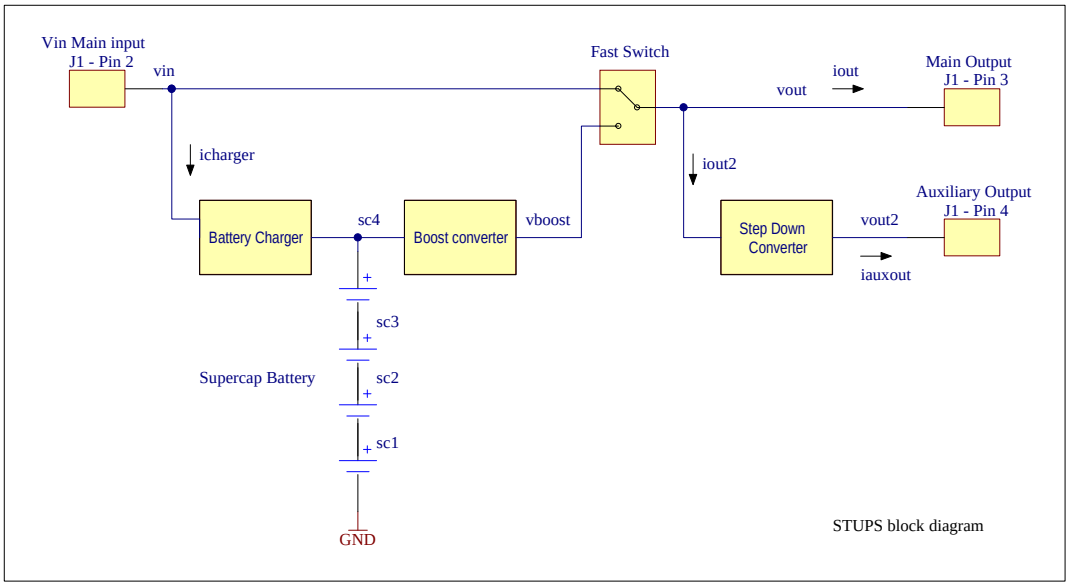
Secondary features of STUPS include auxiliary power output (3.3V/5A or 5V/4A), **system diagnostics** (voltage, current and temperature measurements) and **system start-stop management**.

STUPS can be also used to turn the power to the powered device after a specified time elapsed. This can be useful for example in battery powered data acquisition systems that need to perform some measurement in periodic intervals, for example once per hour. After initial startup the device would perform the measurement task and then send request to STUPS to enter a **hibernate state**. Device would then start it's operating system shutdown. STUPS would detect when the device finished shutdown (by time or output current measurement) and turn off power to the device. After time specified in hibernate message, the STUPS would turn the power back on, device would boot up, perform it's measurement task and repeat the process again. Assuming 1.5A power consumption of device, 25mA consumption of STUPS in hibernate state, tasks duration including boot up and shutdown of 5 minutes and 1 hour task interval, the average system power consumption would drop from 18W/hour to 1.78W/h. The battery would last 10 times longer!

Application Considerations

This paragraph includes list of items to consider when evaluating if STUPS is right for your application or project.

Output voltages



Main Output

When running on main input, the Main Output is connected to main input and output voltage is very close to the input voltage (except of the small voltage drop on connectors, internal traces, current measurement shunt and output switch resistance). Output voltage is thus not regulated and it will change if input voltage changes.

When running on backup source the main output voltage is generated by the boost converter and is regulated to value specified by `BackupOutputVoltage` parameter. With current version of firmware the value can be set between 11 and 13 Volts. Default Value is 12 Volts. If your project needs different backup voltage please contact Linear Computing.

Auxiliary Output

Auxiliary output voltage is generated by step-down converter and is regulated to value specified by `AuxOutputVoltage` parameter. With current version of firmware the value can be set between 3.0 and 5.5 volts. By default the auxiliary output is set to 0 Volts and user must set the voltage to desired value to enable the output. If your project needs different auxiliary voltage please contact Linear Computing.

Maximum power consumed by device(s)

Please verify that STUPS can provide enough output current to safely power your device(s). The sum of current flowing to main output and current flowing to auxiliary output step-down converter (`iout` + `iout2`) must be less or equal to 3.5A @12V (42W) in average. While the battery is charged more then 50%, STUPS can safely provide higher output current 5A @ 12V (60W) for short duration of time - this provides some contingency in case your system demands more power. If you are not using auxiliary output, then `iout2` is practically equal to zero and all the current can be used by the device connected to main output.

If auxiliary output is used then you need to verify that the device connected to auxiliary output does not require more then 4A @ 5V (20W) or 5A @ 3.3V (16.5W). The auxiliary power is generated from the main output by step-down converter, please subtract the current floating to step-down converted from the budget of main output:

Step-Down Converter input current estimate: `iout2` = `iauxout` * `vout2` / (`vout * 0.94`), where iout2 is current flowing to the step-down converter, `iauxout` is current flowing to the auxiliary output device, vout2 is voltage of auxiliary output and vout is voltage of main output. `Iout2`, `vout` and `vout2` are measured by STUPS, you can use StupsView applications to read these values.

Backup Run-time

Please verify that STUPS can provide enough run-time for your application. To calculate the maximum duration of backup power that the STUPS will provide:

First, calculate the power consumption of your system (in Watts) by multiplying the device input voltage in Volts (V) by the current flowing to the device in Amperes (A).

Then, divide the STUPS capacity (Watt-Seconds) by your calculated power consumption to get the run time in seconds.

Example:

A device with an input voltage of 12V and 2 amperes of current has a power consumption of 24W, 12V * 2A = 24W.

With 1000 Watt-seconds capacity, the run time is: 1000Ws / 24W = 42 seconds.

Output Current [A]	Output Power [W]	41321201 Runtime [sec]	41321202 Runtime [sec]
0.5	6	2 min 47 sec	9 min 43 sec
1	12	1 min 23 sec	4 min 52 sec
1.5	18	56 sec	3 min 14 sec
2	24	42 sec	2 min 26 sec
2.5	30	33 sec	1 min 57 sec
3	36	27 sec.	1 min 37 sec
3.5	42	23 sec.	1 min 23 sec

Figure 1 - Runtime for 12V System

Examples of embedded systems:

1. Intel Atom N270/D525 1.6GHz SBC, 8'' LCD display, 2.5'' SSD HD

Power: 12V, 1.5A: 18W

Run time P/N 41321201: 56 seconds

Run time P/N 41321202: 3 minutes, 14 seconds

2. ARM A9 1.0GHz, 6'' LCD display

Power: 12V, 0.6A: 7.2W

Run time P/N 41321201: 2 minutes 18 seconds

Run time P/N 41321202: 8 minutes, 6 seconds

Estimated run-time is calculated by STUPS based on capacity of that particular unit (programmed during factory calibration) and actual power consumption of connected devices. Value can be read using StupsView application.

Please note that backup capacity in the specifications is the initial capacity available at the time of STUPS manufacturing. This value will slowly decrease over time. The rate of decrease is predominantly affected by operating temperature.

Effect of Time and Temperature on Capacity

Supercapacitors have a lot of advantages compared to batteries – they offer practically unlimited number of recharge cycles, are relatively small and light, charge extremely fast and perform great at very low temperatures. These features make them considerably more reliable for a long term use in UPS applications.

However, if they are exposed to high temperatures for long time, their capacity decreases over time rapidly. This makes the STUPS or any other supercapacitor based device impractical for applications where it would be exposed to high temperatures constantly. It is acceptable if high temperatures are present occasionally as long as the majority of the operation time is at typical industrial temperatures.

Please note that all supercapacitor available on the market today (Jan 2014) suffer from this trait even though this fact is not readily advertised by supercapacitor manufacturers or manufacturers of devices that use supercapacitor. We believe that we selected the best performing supercapacitors in this respect for STUPS.

Approximate number of years of continuous operation for 30% decrease of initial capacity based on operating temperature:

25 DegC	40 DegC	45 DegC	50 DegC	55 DegC
34 years	12.9 years	9 years	6.4 years	4.5 years

60 DegC	70 DegC	85 DegC	70% @ 40 DegC 30% @ 60 DegC	90% @ 40 DegC 10% @ 85 DegC
3.4 years	1.8 years	0.6 years	7 years	4.2 years

If 30% decrease of initial run time is acceptable for your application, then STUPS should provide 9 years or continuous operation at 45 DegC. If your application requires only one third of the initial STUPS capacity then you should be able to reach approximately double the life-time indicated in the table above.

Thermal Management

All current consumed by the STUPS internal circuitry will be converted to heat which will cause temperature increase. Amount of this increase depends on dissipated heat of STUPS and other installed devices and thermal configuration of the system.

If auxiliary output is not used the power dissipated by STUPS is approximately 0.8W @ Vin=12V in normal operation and 0.3W in hibernated state. These values are a small fraction of power dissipated by other products that use converters to generate main output power – inefficiency of converter would be directly converted to heat.

If the auxiliary output is used it generates additional dissipated heat of approximately 6% of auxiliary output power, e.g. 1.2W with 5V/4A aux. output consumption.

The simplest method to find out what this means for your particular configuration might be using the STUPS temperature sensor. Assemble a prototype of your system will all components, enclosures and typical loads, turn in on and leave it running for a few hours in typical configuration that you would use. Read the temperature from STUPS before and during the operation to find approximate temperature difference. You can use this temperature difference to estimate what will be STUPS internal components temperature at various ambient temperatures. You can use provided StupsView application t oread temperature, send command `read temp` to get temperature in DegC.

Communication Method

At minimum, the STUPS will have to notify the powered device that it switched to battery backup source, so that the device can act accordingly. Following communication methods are available:

- Opto-isolated digital outputs (not available on “B” version of devices)
- Serial port (RS-232)
- USB using Virtual Comm. Port

Opto-isolated Digital Outputs

Please note that the “B” version of devices do not include digital outputs.

Two outputs are available; mode of operation of each can be configured using parameters `DO1Mode` and `DO2Mode`. One output is typically used to report “on main” versus “on backup” operation, the other typically reports state of start/stop push button – this can be used to notify the application that the user wants to turn the system off.

State of each output can be also set by host using `setdo` command so these outpus can be used for other purpose that your application needs.

Serial Port (RS-232) and USB

Virtual Comm. Port is created when STUPS is connected to the host using USB. Both serial port and USB offer the same communication format and features.

If your host device includes a free native RS-232 port and also a free USB port, we would recommend to consider to choose RS-232. In our experience, RS-232 ports are more reliable compared to USB in industrial settings as the hardware, microcontroller firmware and especially the drivers code running on host is significantly simpler and those less prone to errors. In general we do not recommend a USB to RS-232 dongles or converters; quality of implementation and reliability of drivers varies widely.

Part Numbers

Part Number	Description	Size WxDxH (mm)
41321201	STUPS-C10 w. 1000Ws, 0 to +60	90 x 96 x 32
41321202	STUPS-C35 w. 3500Ws, 0 to +60	90 x 96 x 74
41321203	STUPS-EC10 w. 1000Ws, -40 to +80	90 x 96 x 32
41321204	STUPS-EC35 w. 3500Ws, -40 to +80	90 x 96 x 74
41321205	STUPS-BC10 w. 1000Ws, 0 to +60, no 3.3V/5V output, no DIO	90 x 96 x 32
41321206	STUPS-BC35 w. 3500Ws, 0 to +60 no 3.3V/5V output, no DIO	90 x 96 x 74
41321207	STUPS-EBC10 w. 1000Ws, -40 to +80, no 3.3V/5V output, no DIO	90 x 96 x 32
41321208	STUPS-EBC35 w. 3500Ws, -40 to +80, no 3.3V/5V output, no DIO	90 x 96 x 74

Figure 2 - PN: 41321201

Figure 3 - PN: 41321202

Typical Embedded System Configuration

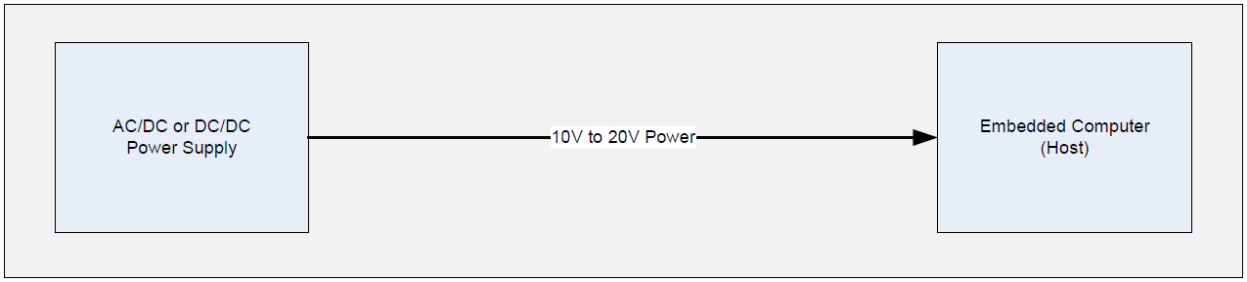


Figure 4 – Embedded System without STUPS

- Host starts to boot up immediately when external AC power is connected
- Host is up while the AC power is present
- Uncontrolled shutdown happens when external power goes off

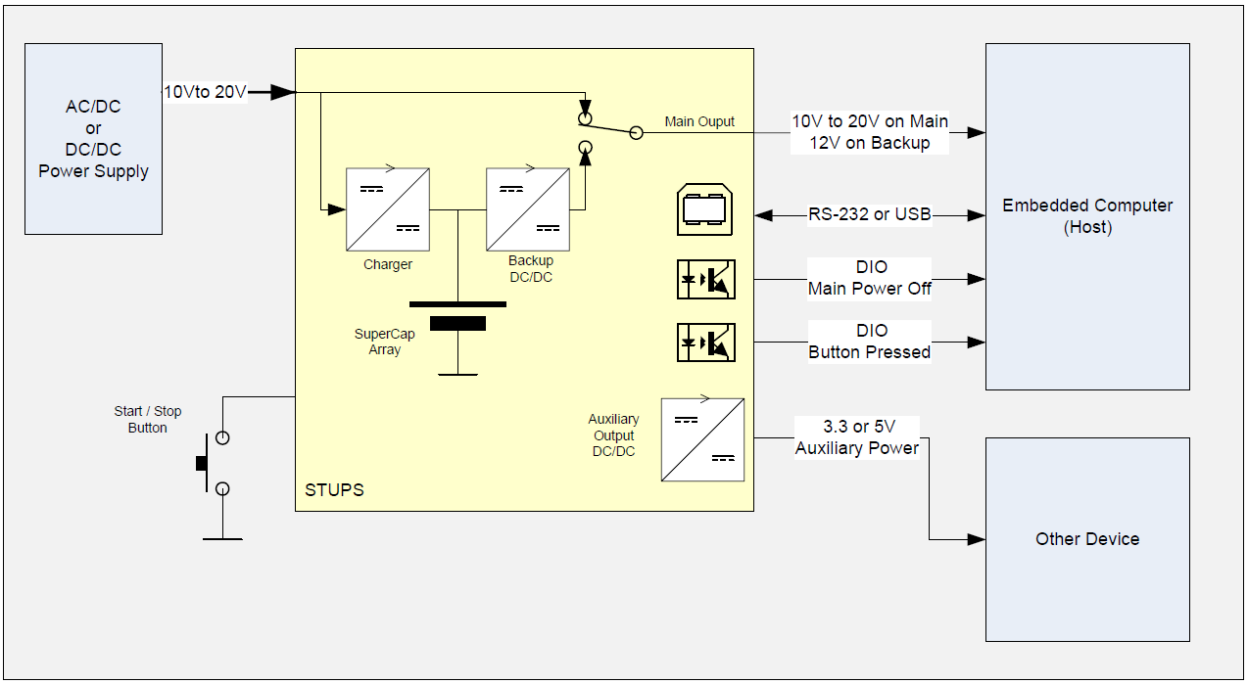


Figure 5 - Embedded System with STUPS configured in Start/Stop mode

- When Start/Stop button is pressed, STUPS turns on power to host. Host boots up.
- If main power goes off, STUPS switches to backup power and notifies Host that main power is off. Host application saves data and starts proper shutdown. Once current flowing to host decreases after shutdown is complete, STUPS turns off power to host.
- If user presses Start/Stop button during operation, STUPS will notify host that button is pressed. Host will save data and start shutdown.

Specifications

Initial Usable Backup Capacity	P/N 41321201: 1000 Watt seconds P/N 41321202: 3500 Watt seconds
Operating Temperature Range Extended temperature models	0 to 60 Deg.C -40 to 80 Deg.C
Input voltage	8.5 to 20V (11.5V or higher is required to reach full backup capacity)
Main Output Voltage When running on main input When running on backup source	 same as input voltage 12V nominal, 11 to 13V adjustable
Auxiliary Output Voltage	3.0 to 5.5V, software adjustable
Maximum Continuous Output Current Main output @ 12V Auxiliary Output @ 5V Auxiliary Output @ 3.3V	 3.5A 4A 5A
Maximum Total Continuous Output Power (sum of Main and Auxiliary outputs power)	42W
Maximum Peak Power Battery charge more then 50%, max 10 seconds, non-repeatable, auxiliary output not used Peak Main Output Current @ 12V Peak Total Output Power	 5.0A 60W
Charging Time (empty to 99% capacity, 2A charging current)	P/N 41321201: 120 seconds P/N 41321202: 420 seconds
Charger Input Current	0.2 to 2A, software adjustable
Switchover time Main to backup switchover	8 microseconds max.
Output Voltage drop below preset threshold Main to backup switch, max load	0.2V max. (with 11.0V switch-over threshold the output will not drop below 10.8V)
Efficiency On main power Backup converter efficiency Auxiliary output converter efficiency Charging efficiency	 STUPS quiescent current 60 mA typ 95% typ 94% typ 93% typ
Communication with host system	RS-232, USB and opto-isolated DIO

Features

- Super-capacitor backup source of energy is more reliable than other battery technologies and provides:
  - Higher number of charging cycles: 500,000
  - Longer life cycle
  - Excellent performance at low temperatures
  - Faster charging
- Very high efficiency when running on main power to minimize power loss and limit dissipated heat
- Fast switch over to backup power to minimize output voltage drop
- Reliable switch over from backup to main power through gradual current ramp-up and input voltage confirmation sequence
- High speed, high efficiency charging circuitry results in quick charging time
- Adjustable charging current setting ensures current (Amperes) capacity of external power supply is not exceeded.
- Active, non-dissipative cell balancing maximizes charging efficiency, minimizes charge time and dissipated heat, and provides long term life of super-capacitors
- Flexible communication scheme for host system notification
  - RS-232, polled or interrupt modes, STUPS can provide interrupt on DSR, RI or CTS line to notify that the main power is off or user pressed the start-stop button
  - USB (uses virtual COM port driver)
  - Opto-isolated digital outputs
- System power diagnostics
  - STUPS can provide current and voltage readings of main and auxiliary outputs, and system temperature to host
- Start-Stop management feature ensures proper system shutdown procedures and data integrity. System can be configured, for example, to provide complete start-stop sequence as in a laptop:
  - Initial button press (power on): STUPS will turn on internal circuits and turn on power to host device. Device boots up.
  - Second button press: STUPS will communicate to the host that the user pressed the button again to shut down the system. Host will save data, confirm shutdown with STUPS and start system shutdown. Once STUPS detects that system shutdown has finished and output current to the device has dropped, STUPS will turn off power to the host and turn itself off. System is now completely off. Other power management modes are available.
- Auxiliary opto-isolated input for general use.
- PC-104 compatible physical form factor and mounting holes

Device Setup

For projects that do not require auxiliary output and use serial port (RS-232) or USB for communication with host, typically only `MaxInputCurrent` parameter needs to be set to match project setup.

If auxiliary output is used parameter `AuxOutputVoltage` must be set to desired auxiliary output voltage (by default it is set to 0 Volts).

Other parameters that are often customized are `UpdateMsgFields` and `UpdateMsgPeriodMs.` These specify how often and what information is sent in STUPS periodic interrupt messages. Please see How to Communicate with STUPS section.

MaxInputCurrent Parameter

This parameter should be set to ampere rating of power supply connected to input of STUPS. STUPS will use this value to regulate charger current to make sure that power supply is not overloaded.

By default the parameter is set to 5.0 Amps. If your power supply is rated at lower value, the STUPS might overload the power supply as it provides current to powered device and charges the battery (this might effectively double the drawn current). Typically, the power supply would turn off, STUPS would switch to backup, after awhile the power supply would turn the power on again, STUPS would switch back to main input, start charger, overload the power supply again, and the switching back and forth would continue until the STUPS battery gets completely depleted. You might not see this problem if the supercap battery is fully charged, but could potentially run into this later. **We strongly recommend to set this parameter to match your particular setup.**

Typically, the ampere rating is indicated on the power supply label. Occasionally only the maximum output power is provided - you can calcuate the rated current by dividing max. output power by output voltage, e.g. for 12V, 42W supply, you calcuate 42 W / 12 V = 3.5 Amps.

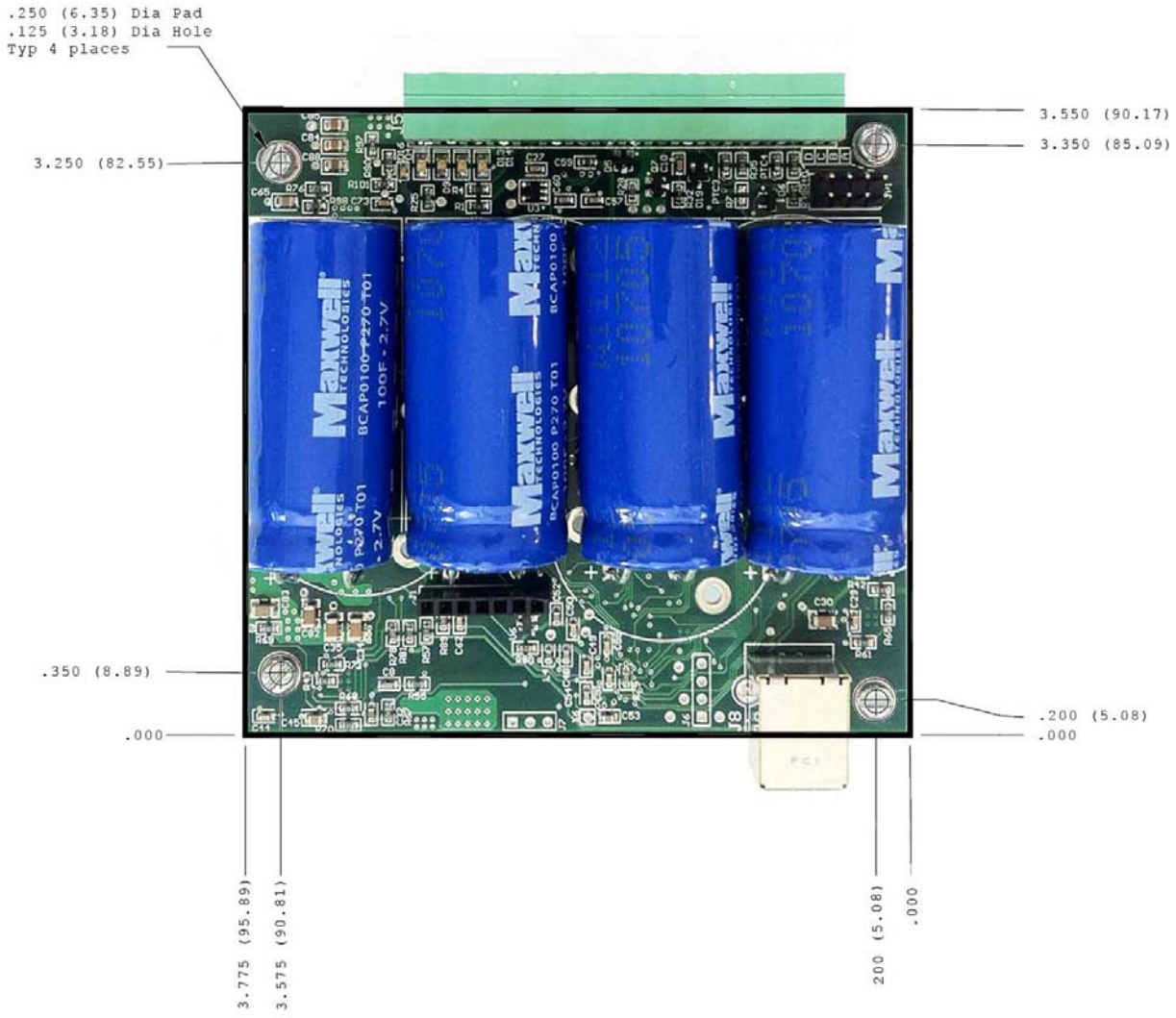
Setting parameters to desired value

The easiest way to set the parameter might be to start the StupsViewCS application, connect to STUPS, type following command to cmd text box and press Send button:

```
servicemode 1; setpar maxinputcurrent 3.5; setpar auxoutputvoltage 5.0;
setpar updatemsgfields status:runtime:battery:iout; storepars;
```

This will change the parameters to specified value and store them into non-volatile flash memory. This needs to be done once only.

Mounting Hole Pattern



Connectors

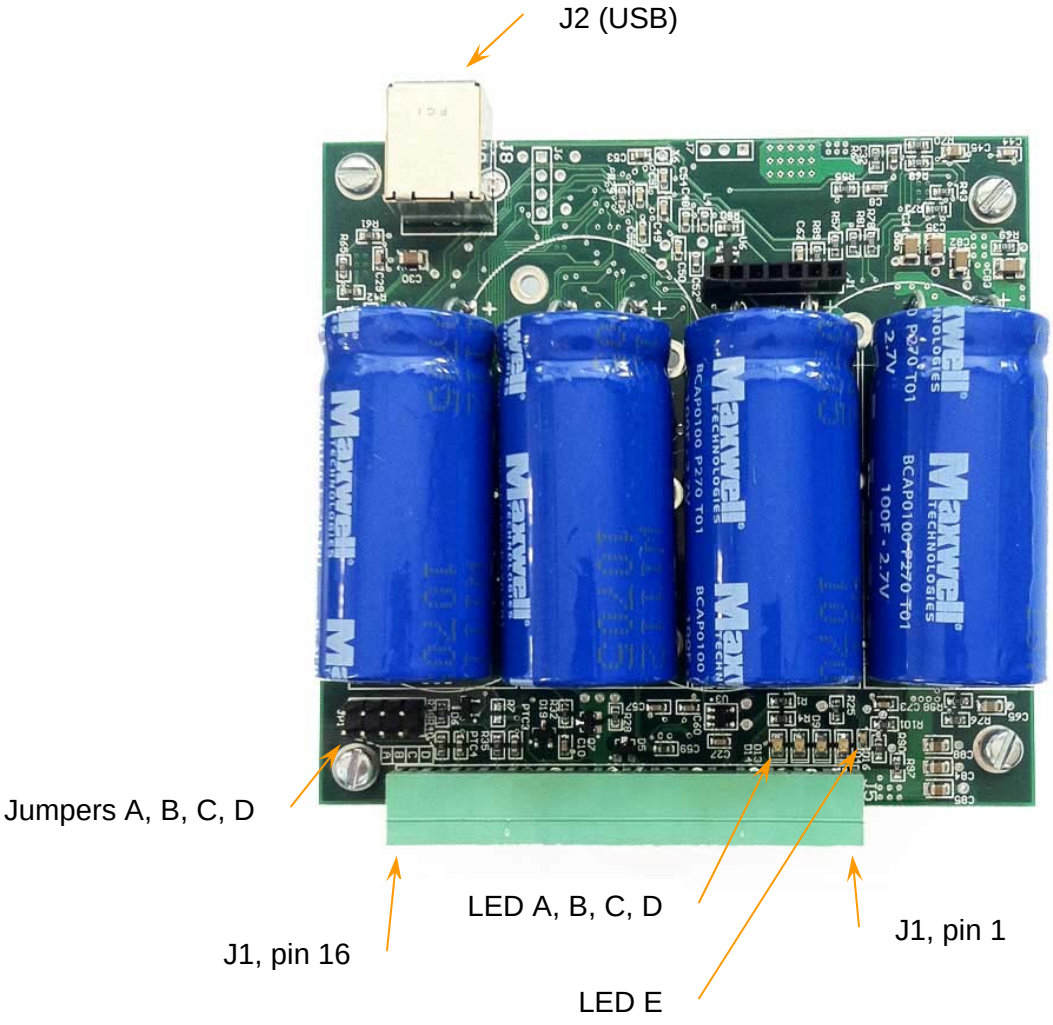


Figure 6 – Device connectors, jumper locations and LED locations

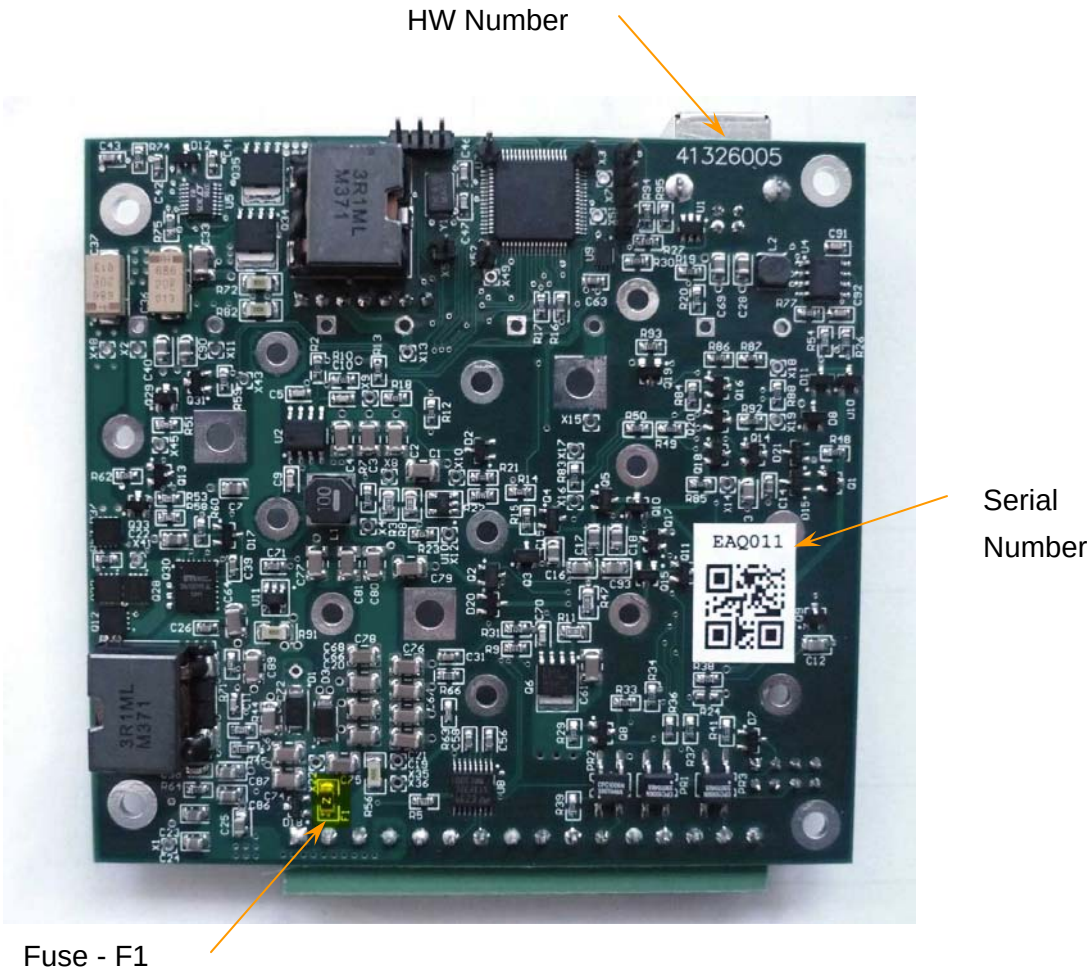


Figure 2 – HW number, serial number and fuse locations

J1 – I/O Connector

STUPS includes mating plug with screw terminals for connector J1 (On Shore Technology PN: OSTTJ1631530, Digikey PN: ED2874-ND)

This connector contains all power and I/O signals except USB communication signals. Typically, input power is connected to pins 1 and 2; output power to host device is connected to pins 3, 4 and 5 (pin 4 is connected only if auxiliary output is present and is used by the host device). RS-232 signals are connected to pins 6 and 7. Start/Stop push button or wire jumper is connected between pins 8 and 9.

Pins 1, 5 and 8 are Gnd signals; they are internally connected in the device and are interchangeable.

Note that B version of devices contains shorter version of connector with pins 1 to 9 only.

Pin #	Description	Signal Type
1	Ground for all power and I/O signals	Gnd
2	Vin – positive side of input voltage source (10 to 20V)	Power input
3	Main output (12V)	Power Out
4	Auxiliary output (3 to 5.5 V)	Power Out
5	Ground for all power and I/O signals	Gnd
6	STUPS Rx pin – connect to host Tx pin (pin 3 on host D-Sub 9)	RS-232
7	STUPS Tx pin – connect to host Rx pin (pin 2 on host D-Sub 9)	RS-232
8	Ground for all power and I/O signals	Gnd
9	Start/Stop push button input. Active when shorted to any gnd pin. If Start/Stop function is not used and STUPS should automatically start when input power is present then place wire jumper between pin 9 and any gnd pin (e.g. pin 8). Host application can read state of this pin using `Read` or `ReadIO` cmd.	I/O Input
10	Opto-isolated input – positive side. Functionality depends on STUPS configuration. This pin can be used to start or wake-up STUPS by external signal or can be used as a general purpose input. Host application can read state of this pin using `Read` or `ReadIO` cmd.	I/O Input

11	Opto-isolated input – negative side	I/O Input
12	Pull-up input used with opto-isolated outputs	I/O output
13	Opto-isolated output 1 – side A. Function depends on STUPS configuration. Host application can set state of this output.	I/O output
14	Opto-isolated output 1 – side B	I/O output
15	Opto-isolated output 2 – side A. Function depends on STUPS configuration. Host application can set state of this output.	I/O output
16	Opto-isolated output 2 – side B	I/O output

Table 7 – J1 - I/O Connector Pin-out

LEDs

Please see figure 1 for location of LEDs.

Pin #	Description
LED A	Internal 3.3V – LED is on when STUPS microcontroller power is present.
LED B	Auxiliary Output – LED is on when auxiliary output is on (J1 pin 4)
LED C	Main Output – LED is on when main output is on (J1 pin 3)
LED D	Vin Input – LED is on when input voltage is present (J1 pin 2)
LED E	Status LED – please see table below for description

Table 3 – LED description

Status LED (LED E) Description

Very slow flashing – 1 flash per 3 seconds

Slow flashing – 1 flash per second

Fast flashing – 5 flashes per second

pattern	Description
Very slow green flashes	STUPS is in hibernated state, output power is off
Slow green flashes	Normal operation of STUPS, running on power from main input
Fast green flashes	Running on backup power, battery charge more than 25%
Fast red flashes	Running on backup power, battery charge less than 25%
Slow red flashes	Output power is off

Very Slow red flashes	State machine disabled. This typically indicates device that was not successfully calibrated. Please contact Linear Computing.
1 red, 1 green flash	Boot-loader code running. This typically indicates incorrect flash parameters or internal STUPS fault. Please contact Linear Computing.
1 red, 2 green flashes	Main program not present. Please contact Linear Computing.
1 red, 3 green flashes	Main program CRC error. Please contact Linear Computing.
3 red, 1 green flash	Unrecognized parameter name. Please contact Linear Computing.
3 red, 2 green flashes	Parameter value string too long. Please contact Linear Computing.
3 red, 3 green flashes	Invalid Parameter Value. Please contact Linear Computing.
3 red, 4 green flashes	Invalid Parameter Format. Please contact Linear Computing.
3 red, 5 green flashes	Invalid Parameter index. Please contact Linear Computing.
3 red, 6 green flashes	Parameter section CRC mismatch. Please contact Linear Computing.
3 red, 7 green flashes	Parameters flash operation failed. Please contact Linear Computing.
4 red, 1 green flash	Flash operation error. Please contact Linear Computing.
5 red, 1 green flash	ADC reset calibration timeout. Please contact Linear Computing.
5 red, 2 green flashes	ADC calibration timeout. Please contact Linear Computing.

Table 4 – Status LED states description

Connection Diagrams

Pin 1, 5 and 8 on STUPS J1 are ground pins. They are internally connected and are fully interchangeable. You can use any of these pins for input, output, push button or RS-232.

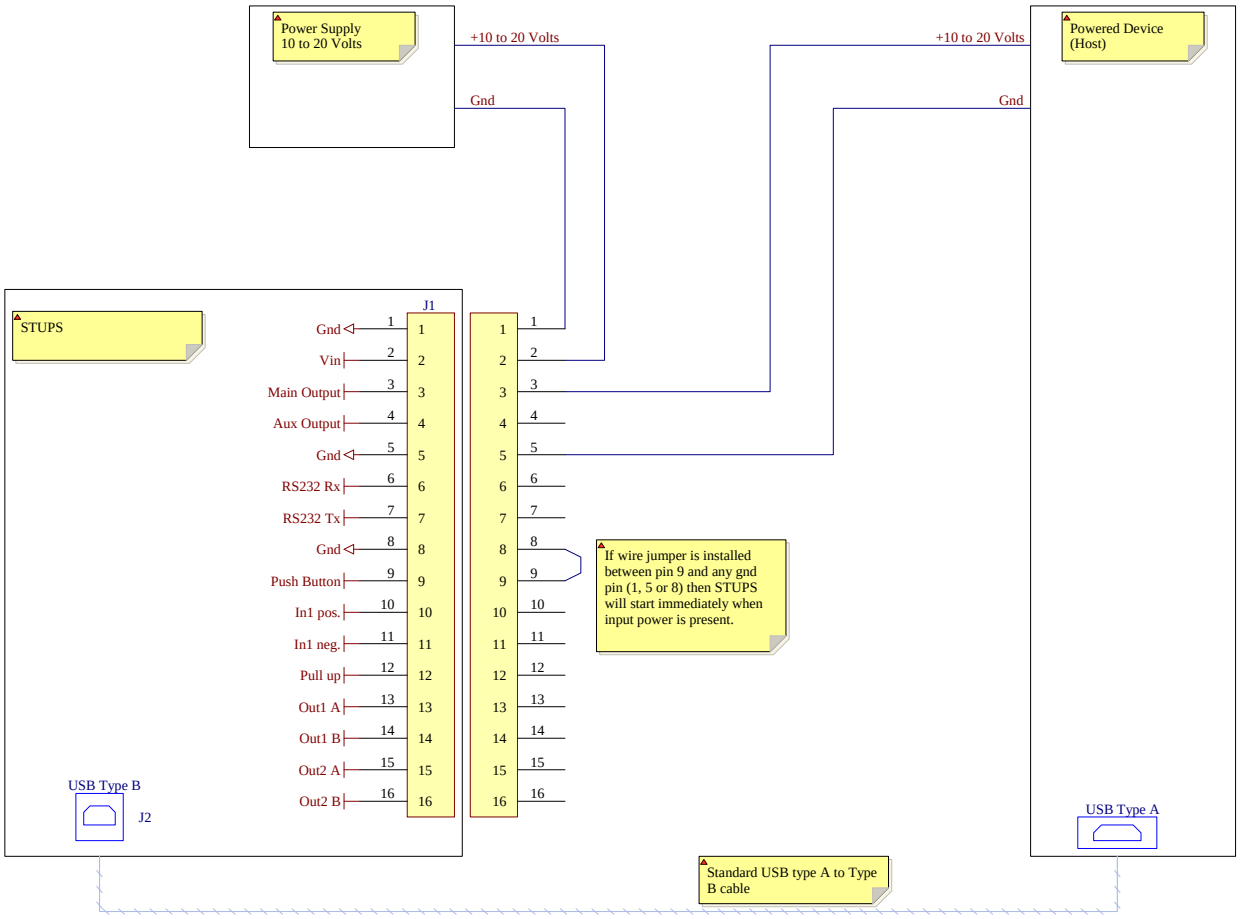
To turn on STUPS and main/auxiliary outputs one of the following must be installed:

- Wire jumper between pin 9 and any gnd pin (1, 5 or 8). STUPS will start when input power is connected and will run until the input power goes off and batteries are completely depleted.
- Plastic jumper on jumper D position (see figure 6 in Connectors paragraph). STUPS will start when input power is connected and will run until the input power goes off and batteries are completely depleted.
- Start/Stop Push button between pin 9 and any gnd pin (1, 5 or 8). STUPS will start when input power is present and push button is pressed.
- Jumper C is present and opto-isolated input 1 is wired. STUPS will start when input power is present and input is active.

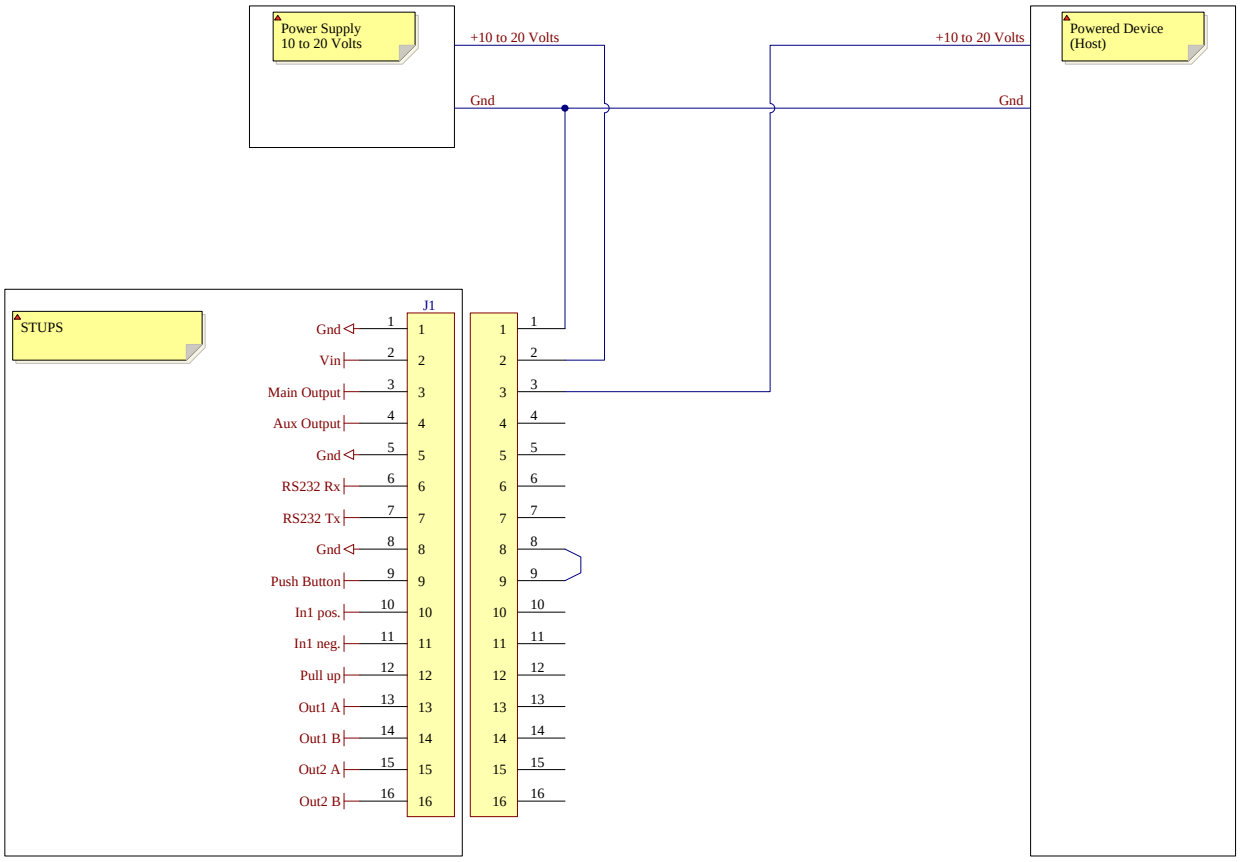
STUPS will turn off main/auxiliary outputs when:

- Input power is not present and STUPS battery is fully exhausted
- `PowerOff` command was sent by host and shutdown conditions were met.

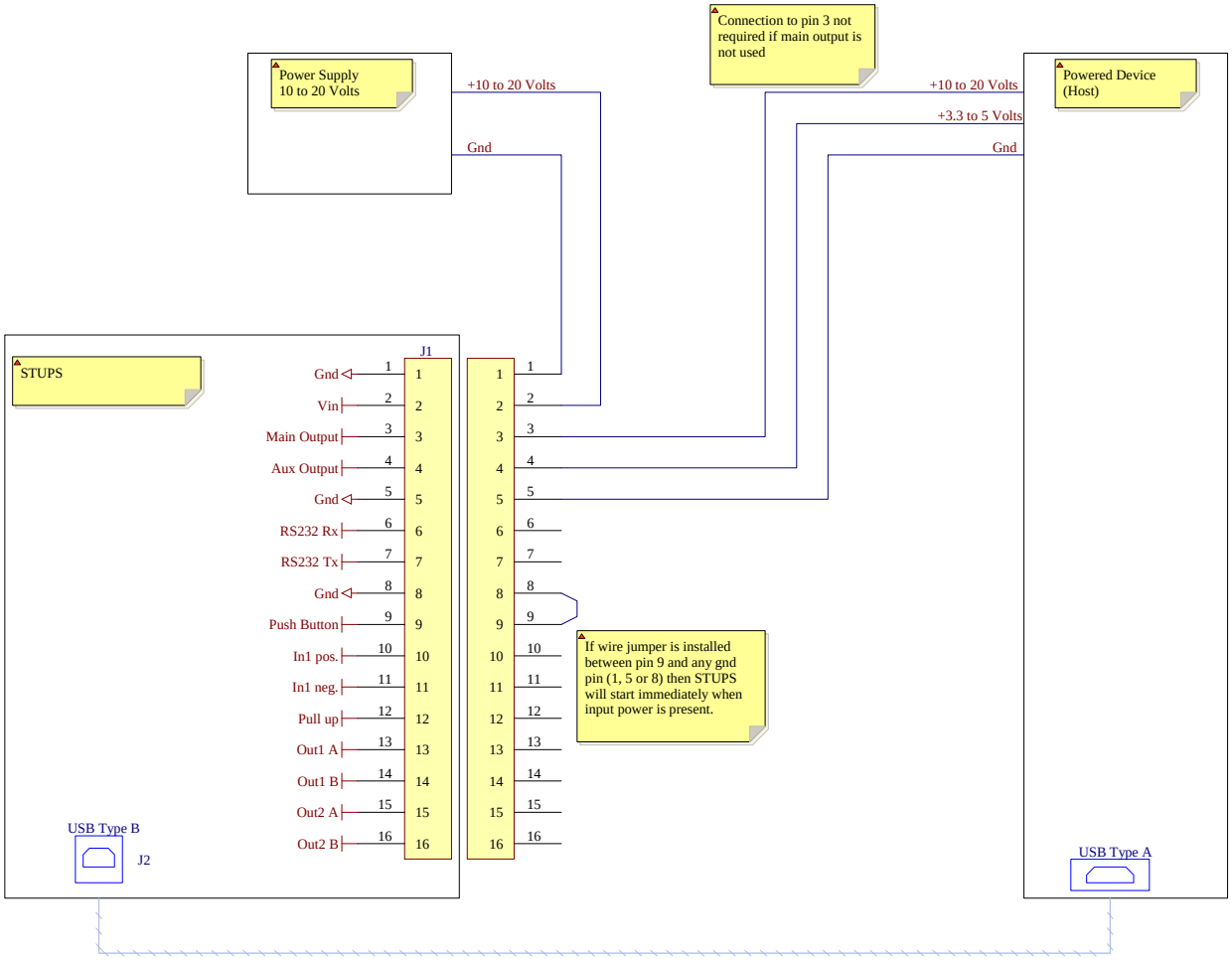
Basic connections - USB



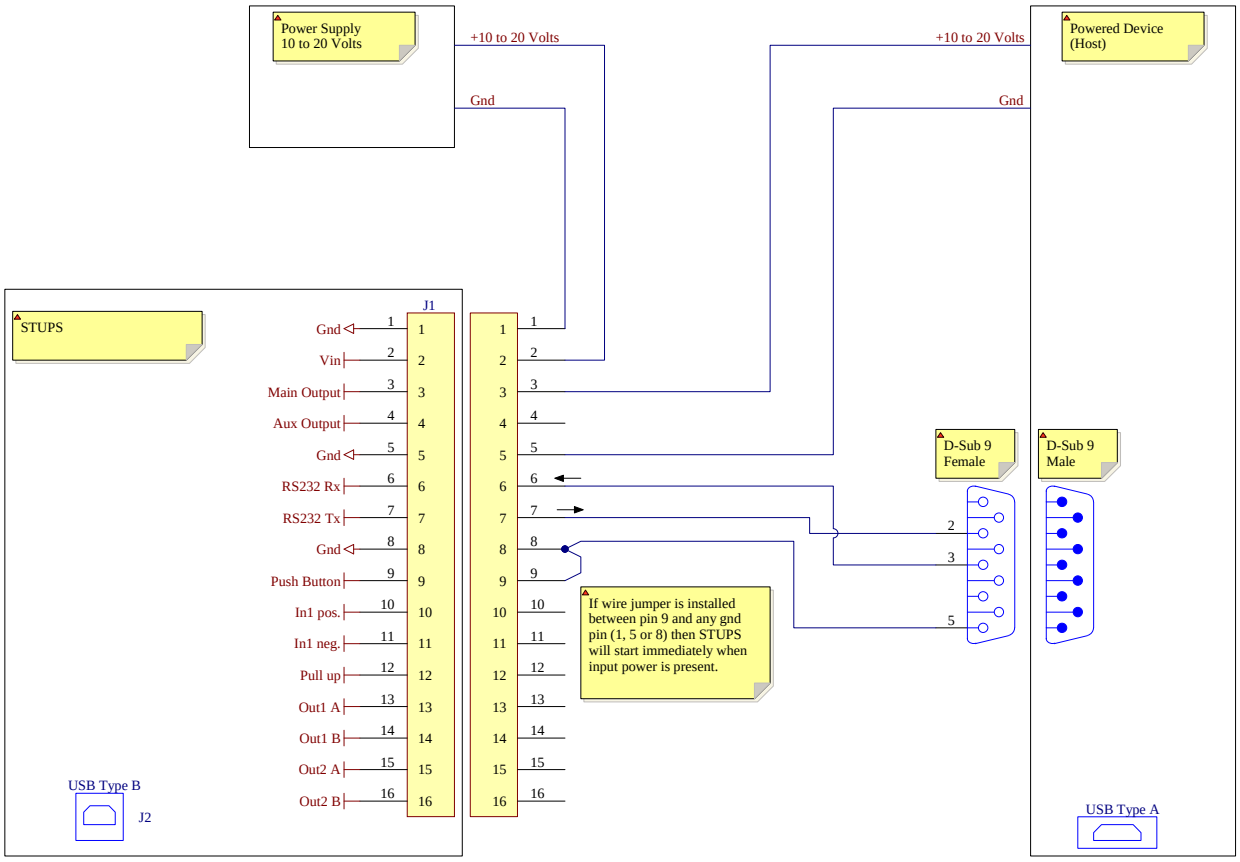
Alternative ground connection



Auxiliary Output Connections



RS-232 Connections



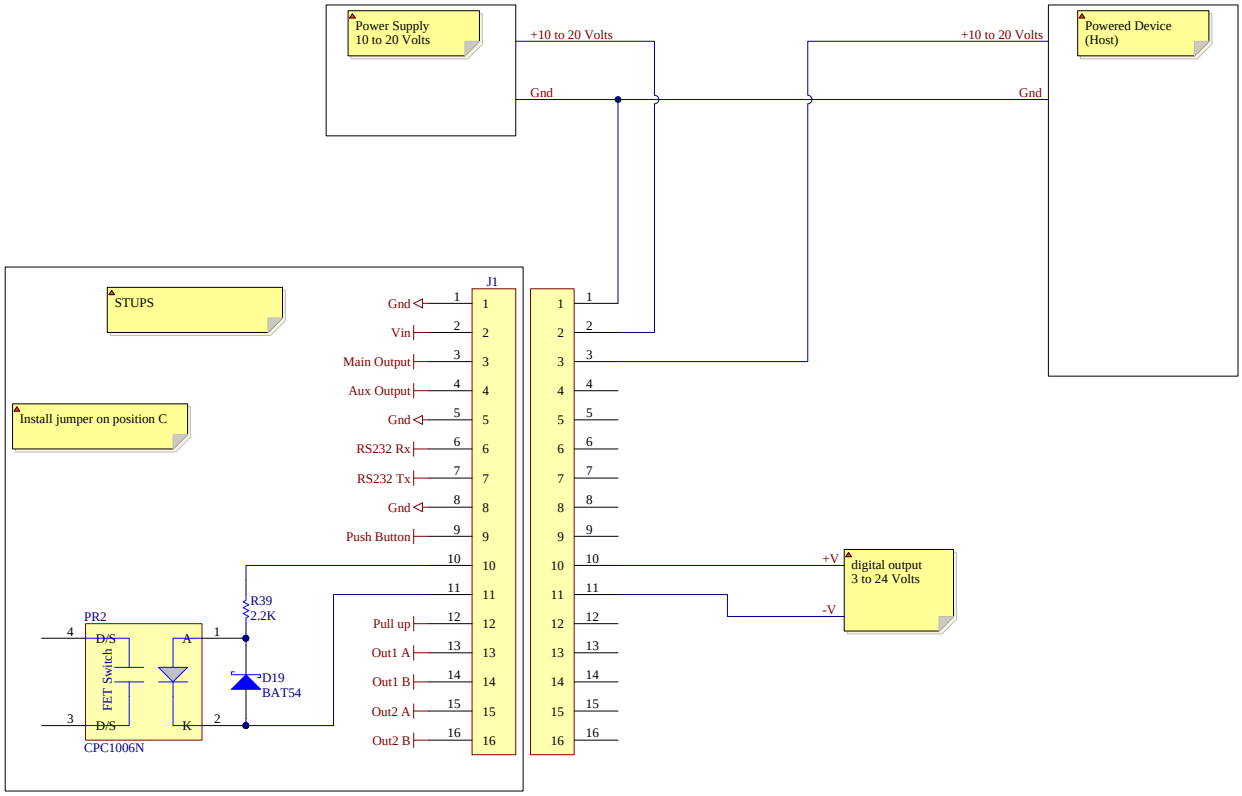
Opto-Isolated Input 1 Connections

As only 0.5 mA is required to activate the STUPS opto coupler (IXYS CPC1006N), almost any device will be able to interface to STUPS with no additional circuitry. Pins 10 and 11 are fully isolated from STUPS circuit.

Input is active if voltage between pins 10 (positive side) and 11 (negative side) is between 3 and 24 volts. Output is not active when voltage is less than 1 volt.

Input value can be read using `Read` or `ReadIO` commands or by configuring interrupt message appropriately.

Input can be used to turn on STUPS. If a plastic jumper must be installed on jumper C position (see figure 6) then the active input will turn on STUPS.



Opto-Isolated Digital Outputs

Please note that the “B” version of devices do not include digital outputs.

Two outputs are available; mode of operation of each can be configured using parameters `DO1Mode` and `DO2Mode`. One output is typically used to report “on main” versus “on backup” operation, the other typically reports state of start/stop push button – this can be used to notify the application that the user wants to turn the system off.

State of each output can be set using `SetDo` command.

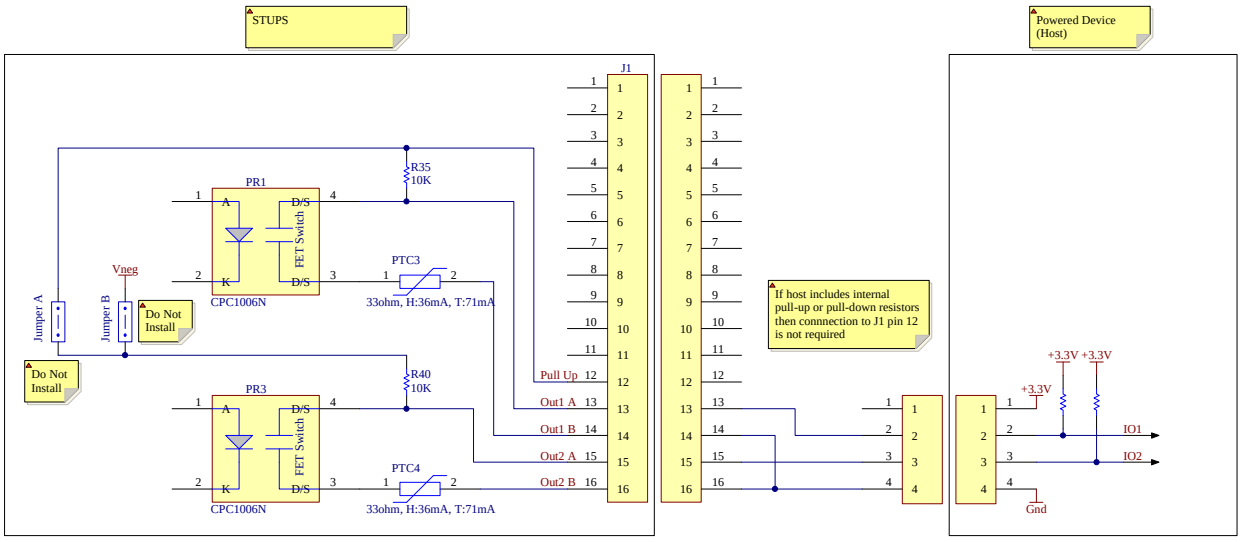
STUPS includes a 10K pull up resistors so in most cases no external components are required. Circuits can be wired in positive or negative logic and allow to interface to systems with voltages between 3 and 30 Volts.

Each digital output contains one opto coupler (IXYS CPC1006N), one pull up/down resistor and automatically resettable poly-fuse to protect output from damage by over current. Outputs can deliver switch up to 36 milliamp, higher current might trip the polyfuse. Once tripped, the polyfuse would turns into high resistance. Once the source of high current is removed and the polyfuse cools down it will automatically starts conducting again. Nominal polyfuse impedance is 33 ohms. Polyfuse PN: Murata PRG18BB330MB1RB.

If outputs are connected to inductive load (relay or solenoids) fly back diode must be connected, otherwise the output will be damaged by high voltage pulse generated by inductor.

Host with built in pull up resistors

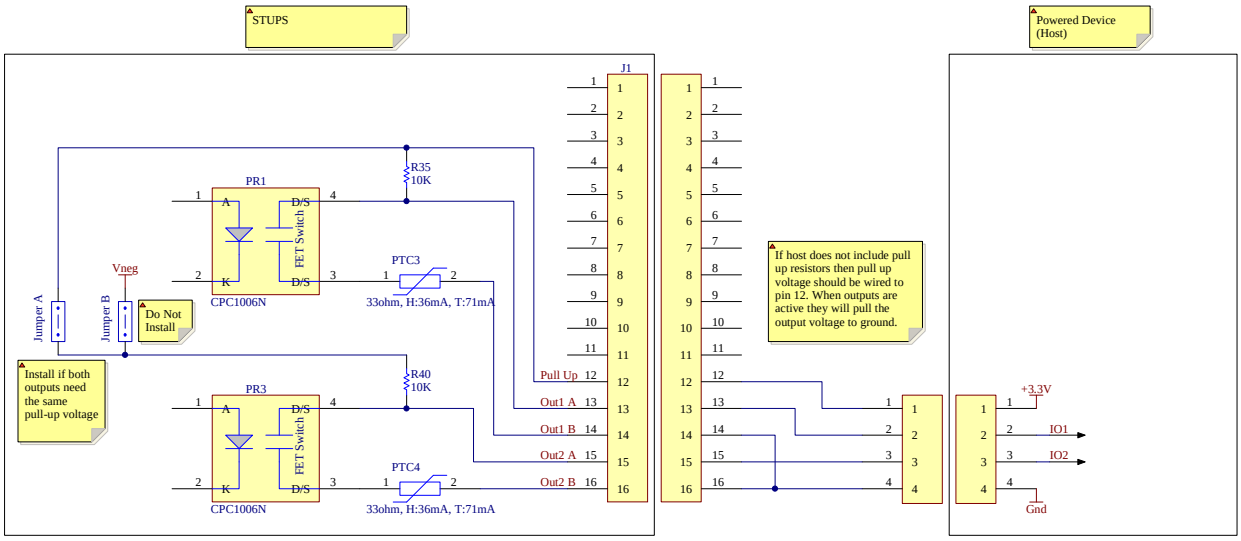
If digital inputs in host device include pull up or pull down resistors then these should be used and no connection to J1 pin 12 should be made. Plastic jumpers A and B should not be installed. Please see circuit below.



Host without internal pull up resistors

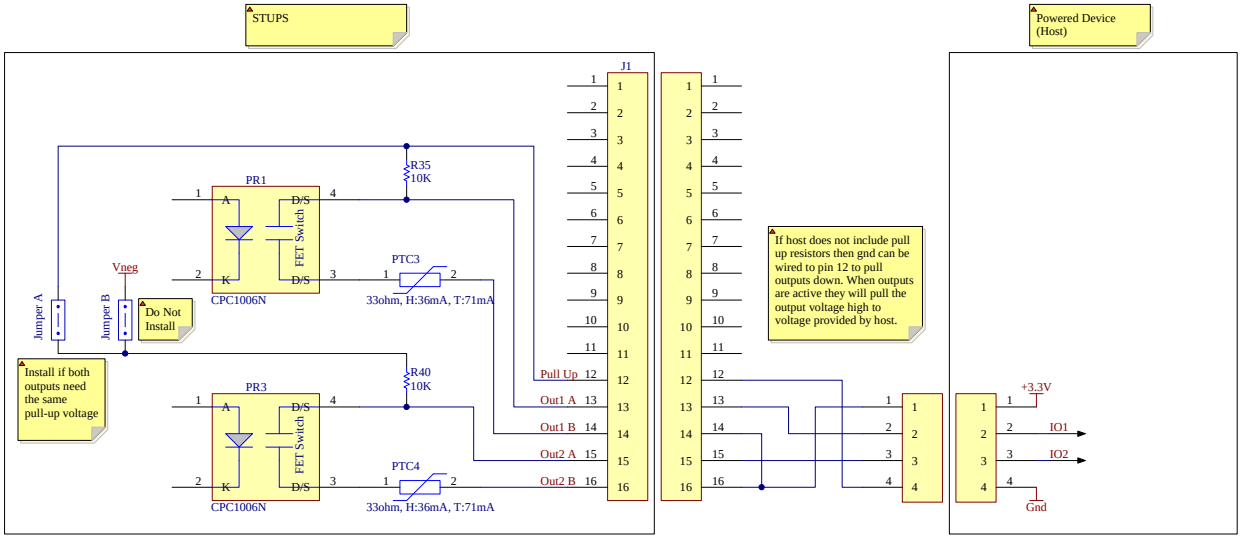
If host does not include internal pull up or pull down resistors then STUPS pull up resistors should be connected by wiring pull up / pull down voltage source to pin 12. Jumper B should not

be installed. Jumper A should be installed if both outputs are connected to the same host. Please see circuit below.



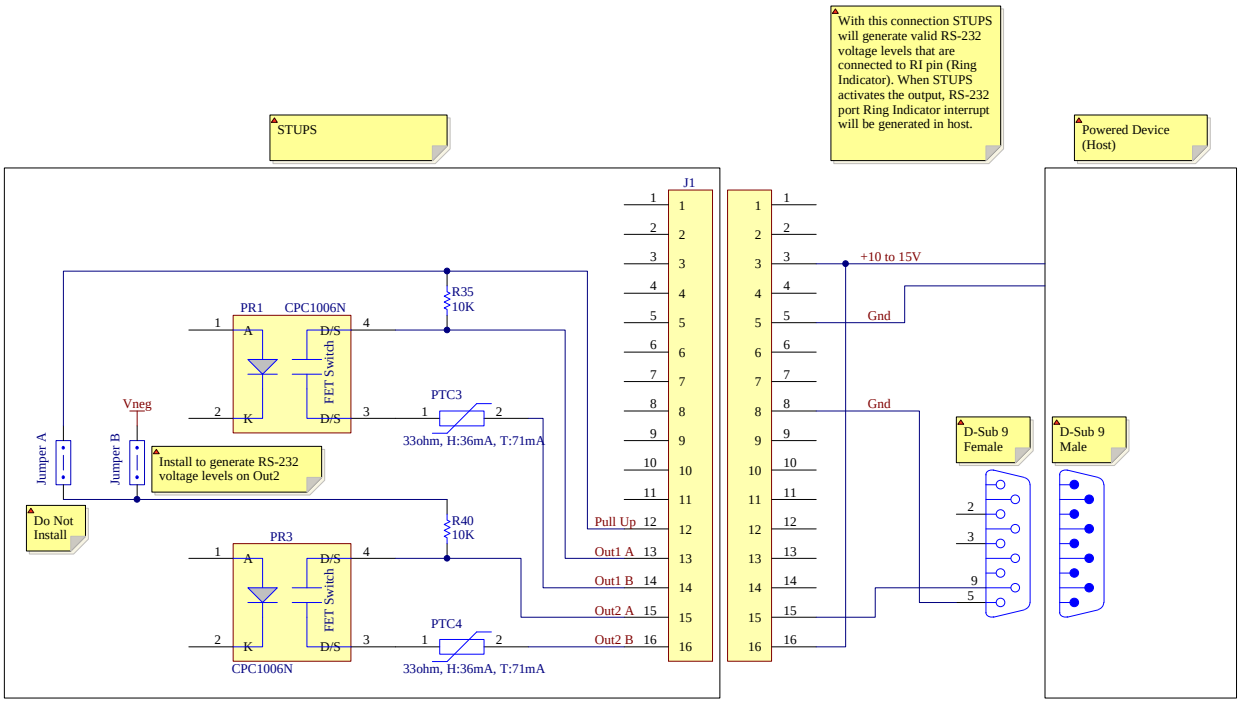
Alternative inverted connection for host without internal pull up resistors

Alternatively, if host does not include pull up or pull down resistors, circuit below can be used. Jumper B should not be installed. Jumper A should be installed if both outputs are connected to the same host.



Output with RS-232 voltage levels

Output 2 can be used to to connect to RS-232 port. When the output is active it will generate interrupt on DSR, RI or CTS line of host RS-232 port to notify that the main power is off or user pressed the start-stop button. Jumper B should be installed, it will connect negative voltage to output 2 pull up resistor. Jumper A should not be installed.



Communication

Following methods are available for communication between STUPS and host:

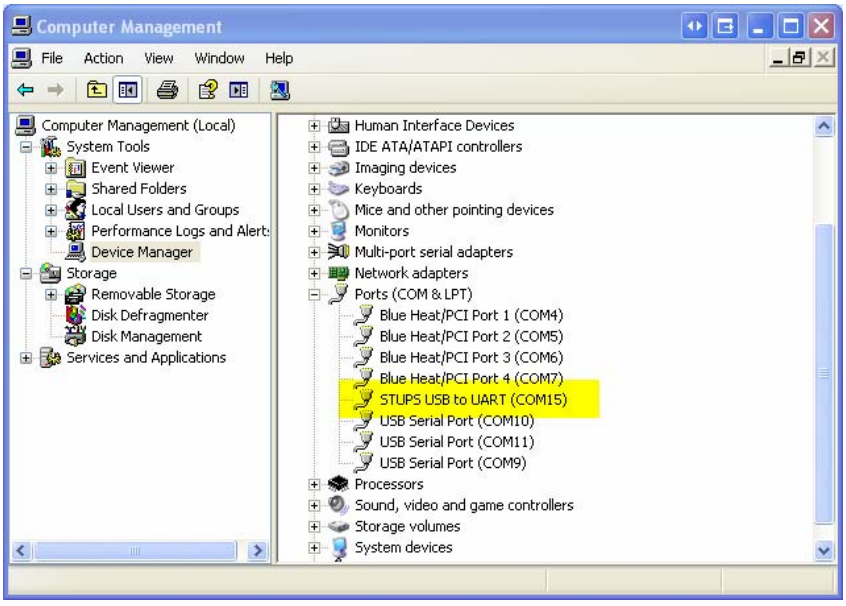
- Serial port (RS-232)
- USB 2.0
- Opto-isolated digital IO

USB Communication

When STUPS USB driver is installed, a new virtual com. port is created. Applications can open this port and communicate with STUPS the same way as if serial (RS-232) connection was used.

USB Driver Installation - Windows

- Connect STUPS to host
- Turn on input power to STUPS, verify that STUPS is running (check status LED)
- Operating system should detect a new device and open installation dialog
- Select advanced method and when windows asks for driver files navigate to inf file provided by Linear Computing.
- If driver is installed correctly a new port should be shown in device manager in ports section. Host application should open this port as it would open any other RS-232 serial port. For example, system on picture below would use comm. port 15 to talk to STUPS.



USB Driver Installation – Linux

No driver installation is required under most versions of Linux as the stups Linux built in standard Virtual Comm Port driver is used. When the stups is connected to the host a new entry should be created automatically under /dev for example: /dev/ttyUSB0.

Serial Port (RS-232) Communication

Communication settings:

- 1 start bit, 8 data bits, no parity, 1 stop bit
- No handshaking
- Baud rate: 115200

Text mode is used for communication with STUPS. All messages sent and received from STUPS contain only readable characters. Hyperterminal, putty or any other terminal application can be used to send commands to STUPS. StupsView application also allows to send user commands.

Command sent to host should be terminated with any of following sequences:

- Carriage return (CR, '\r', ascii decimal value 13)
- New Line (LF, '\n', ascii decimal value 10)
- 2 character CR+LF sequence

By default, all terminal applications use one of the following sequences when user presses ENTER key. To send message to STUPS, open terminal application of your choice, set communication settings listed above, type cmd (e.g. "ping") and press Enter. This should be enough for STUPS to properly receive host request.

By default, all messages sent by STUPS to host will be terminated with 2 character CR+LF sequence.

How to communicate with STUPS

In typical setup the application running on host will open a port and starts listening to STUPS interrupt periodic messages. These messages include current value of STUPS status, battery state, estimated run-time and values of measurement points as configured by parameter `UpdateMsgFields.` Parameter `UpdateMsgPeriodMs` specifies how often is the message sent.

By default, the interrupt message is sent every 3 seconds and every time when STUPS status changes (for example, when STUPS switches from main to backup source or from backup to main). With default value of parameter `UpdateMsgFields,` fields sent are STUPS status and estimate of available run-time in seconds, e.g.

```
*Main 97
*Backup 32
```

Please see Interrupt messages section for more details.

Parameters `UpdateMsgFields and UpdateMsgPeriodMs` should be configured and stored to non-volatile memory to meet the needs of each particular project. One of the software applications provided by Linear Computing can be used to do this.

Receiving and processing of interrupt message is often all the host application needs to do. As this message is sent automatically, it is possible to implement full UPS functionality without a need to send any messages to STUPS.

Most important value is the `status` field – this describes if the STUPS runs on main input or on backup battery source. Application should process this value and start operating system shutdown when the STUPS indicates that it runs on battery.

Optionally, the host application can delay start of shutdown for a few seconds to ride-through short input power blips that are often present in industrial applications. If such a blip occurs, STUPS would report Main to Backup status change, and when the Vin input power is restored to valid levels for time specified by parameter `PowerGoodTimeSec` (default 1 second), it would attempt to switch back to main source. If successful, the STUPS would send another interrupt message notifying that it runs on main source again. Below is an example of sequence of messages:

```
*Backup 32
*Main 30
```

For more advanced features the application can send requests to STUPS at any time.

It is also possible to completely disable the interrupt messages by setting parameter `UpdateMsgPeriodMs` to -1; host application should then use `Read` or `GetStatus` command to query status of STUPS periodically.

Interrupt messages

Interrupt messages are sent by STUPS:

- Periodically, as specified by parameter `UpdateMsgPeriodMs` (by default every 3 seconds)
- Every time when STUPS status changes (for example, when STUPS switches from main to backup source or from backup to main)

Example interrupt messages sent by STUPS:

```
*Main 97
*Backup 32
*Off 36
*Error34 42
```

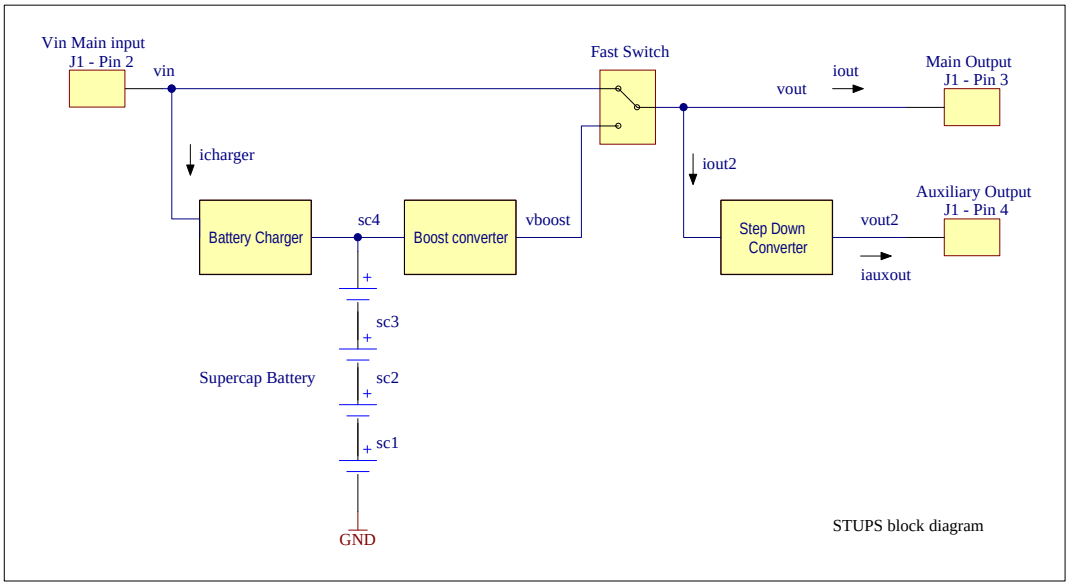
First character in interrupt message will always be '*', ascii decimal value 42. This notifies that this is an interrupt message sent by STUPS (this is not a response to request sent by host)

Current value of one or more fields will be sent. Field values are separated by space. Number of fields and what fields are sent depends on value of parameter `UpdateMsgFields.` Following fields are supported:

Interrupt Message Field Names

- `status` – STUPS status with one of following values:
  - `Main` – outputs are on, powered by main input
  - `Backup` – outputs re on, powered by backup supercapacitor battery
  - `Off` – outputs are off
  - `Errorxx` – fatal error present. xx is error code matching the status LED flashing pattern. First digit describes number of red flashes, second number of green flashes. For example, Error34 (3 red and 4 green flashes indicates invalid parameter format). See Status LED section for more details.
- `runtime` – estimated run-time in seconds based on current output consumption and charge left in battery. When running on backup this value will gradually decrease, STUPS will turn off outputs when the value reaches 0.
- `battery` – battery charge in percent, 100% - fully charged, 0% - completely exhausted, 50% charge corresponds to 50% run-time left.
- `temp` – temperature of STUPS PCB in DegC.
- `di` – state of digital inputs, see ReadIO command for more information
- `do` – state of digital outputs, see ReadIO command for more information
- `vin` – main input voltage, J1 pin 2
- `vout` – main output voltage, J1 pin 3
- `vout2` – auxiliary output voltage, J1 pin 4
- `vboost` – voltage of boost converter, this voltage will be connected to main output when STUPS switches from main to backup source
- `iout` – current flowing out from main output, J1 pin 3

- `iout2` – current flowing from output of boost converter **into** step-down converter generating auxiliary output. Current flowing out from auxiliary output (J1 pin 4) can be estimated as: `iauxout` = `iout2` * `vout` / `vout2` * 0.94
- `iauxout` – calculated value of current flowing from auxiliary output to load. Please note that this is estimate only.
- `icharger` – current flowing from main input to battery charger circuit
- `sc1` – voltage on the top of first supercapacitor (bottom supercapacitor)
- `sc2` – voltage on the top of second supercapacitor
- `sc3` – voltage on the top of third supercapacitor
- `sc4` – voltage on the top of fourth supercapacitor (top supercapacitor) – this is also battery voltage and input to boost converter



UpdateMsgFields Parameter

Parameter `UpdateMsgFields` contains colon separated list of field names to send in interrupt message, default value is: `status:runtime`

To change parameter to send status, runtime, battery percentage and main output current, send following command to STUPS:

```
servicemode 1;setpar updatemsgfields status:runtime:battery:iout;storepars
```

This request also stores parameter to non-volatile flash memory so new value will be used even if STUPS microcontroller reboots.

We recommend to always include `status` field in the first field as LCI demo applications will expect it there.

UpdateMsgPeriodMs Parameter

Parameter specifies how often should be interrupt message sent in milliseconds. Default value is 3000 (corresponds to 3 seconds). If this parameter is set to special value of 0, then interrupt messages are sent only when STUPS power status changes. If this parameter is set to special value of -1, then interrupt messages are not sent at all (application should then use `Read` or `GetStatus` command to query status of STUPS periodically.)

Host Request Format

Example of host request and STUPS response:

```
read vin vout iout
$read OK 12.313 12.290 2.013
```

Each request starts with command name. Optional parameters separated with one or more space characters can follow. Request must be terminated with one of the terminating sequences described in previous paragraph.

Case Sensitivity

STUPS will process command names in lower or upper case. However some parameters must be specified with exact case otherwise STUPS will respond with invalid parameter value error. We recommend to send all text in lower case.

STUPS Response Format

First character in response is always ‘$’, ascii decimal value 36. This notifies that this is a response to host request (as opposed to interrupt message sent by STUPS). As STUPS sends interrupt messages independently, it is possible that interrupt message will be sent before the response to the host request. For example:

```
read vin vout iout
*Main 30
$read OK 12.313 12.290 2.013
```

Host application can use the first character of received message to distinguish between interrupt messages and host responses.

STUPS sends response to every request. Each response will be terminated with 2 character CR+LF sequence. This should be used to detect the end of response.

First word in the response is the same as the first word in request (this is command name)

Second word indicates if the request was successful, STUPS will send “OK” for success or “ERR” for error response.

In successful response, one or more words separated with one or more space characters can follow.

In error response the third word is an error number followed with description of error, for example:

```
read vouy
$read ERR 3 unrecognized meas point
```

Response Time

STUPS will respond in less then 200 milliseconds from the moment when terminating sequence of request is received. For most commands the response will arrive in approximately 20 milliseconds.

Retry Logic

We recommend to implement retry logic in host to make communication with STUPS more robust. Following logic can be used:

1. Clear receive buffer.
2. Send request to STUPS
3. Start timeout timer
4. Wait until complete message is received (detect complete message by CR+LF sequence)
   - 4.1. If this is interrupt message (first character is ‘*’), process or ignore it.
   - 4.2. If this is response (first character is ‘$’) then verify if first word in response matches request. If not, ignore the response and keep waiting. If matching response is received exit the waiting loop and process response.
5. If response is not received in 200 milliseconds, retry sequence from step 1 two more times

Sending multiple commands in one request

Multiple commands can be sent in one request by separating them with semicolon character ";", ascii decimal value 59. STUPS will respond by concatenating words of responses to both commands with semicolon. Example:

Two commands can be sent one by one:

```
getstatus
$getstatus OK Main 36

read icharger
$read OK 0.042
```

Or can be sent in one request:

```
getstatus;read icharger
$getstatus OK Main 36;read OK 0.042
```

Multiple commands can be sent in one request as long as both request and expected response are shorter than 256 characters; otherwise they will not fit in the STUPS RX or TX buffer.

Host Commands

Ping

```
ping
$ping OK
```

Command is used to test communication with STUPS; the only action performed is to respond back to host.

ReadPar

```
readpar MaxInputCurrent
$readpar OK 3.8
```

Read parameter value. Second word is parameter name (case not sensitive). See section Parameters for more information on parameter names.

ReadParDefaultValue

```
readpardefaultvalue MaxInputCurrent
$readpardefaultvalue OK 5.0
```

Read parameter default value. Second word is parameter name (case not sensitive). See section Parameters for more information on parameter names.

SetPar

```
servicemode 1
$servicemode OK
setpar MaxInputCurrent 3.2
$setpar OK
```

Set parameter value. Second word is parameter name (case not sensitive). Third word is new parameter value. To prevent inadvertent change of behavior, STUPS must be set to service mode before setpar command is sent. Setpar command will change value of parameter in RAM (volatile) memory only. If this value is not stored in non-volatile Flash memory, the new value will be lost on next reset or power-up. Use StorePars command to save all current parameter values to Flash memory. See section Parameters for more information on parameter names.

ClearPar

```
servicemode 1
$servicemode OK
clearpar MaxInputCurrent
$clearpar OK
```

Set parameter to it's default value. To prevent inadvertent change of behavior, STUPS must be set to service mode before clearpar command is sent. Clearpar command will change value of parameter in RAM (volatile) memory only. If this value is not stored in non-volatile Flash memory, the new value will be lost on next reset or power-up. Use StorePars command to save all current parameter values to Flash memory. See section Parameters for more information on parameter names.

ClearAllPars

```
servicemode 1
$servicemode OK
clearallpars
$clearallpars OK
```

Set all parameters to it's default value. To prevent inadvertent change of behavior, STUPS must be set to service mode before clearallpars command is sent. Clearallpars command will change value of parameter in RAM (volatile) memory only. If this value is not stored in non-volatile Flash memory, the new value will be lost on next reset or power-up. Use StorePars command to save all current parameter values to Flash memory. See section Parameters for more information on parameter names.

This command takes approximately 160 milliseconds to execute so the response will be sent later then for other commands.

StorePars

```
storepars
$storepars OK
```

StorePars command will save all current parameter values to non-volatile Flash memory. Flash memory in STUPS microcontroller is rated to maximum 1000 flash write operations. As each StorePars request performs one flash write, we recommend to change all parameters to desired value first with SetPar command and then call StorePars command once to save them all.

ReadParByIndex

```
readparbyindex 1
$readparbyindex OK Device STUPS-C
readparbyindex 0
$readparbyindex OK ProductName STUPS-C10
readparbyindex 0
$readparbyindex OK PartNumber 41321201
…
readparbyindex 0
$readparbyindex OK NominalOutputPower 25.0
readparbyindex 0
$readparbyindex OK ParamsEnd 0xA5
readparbyindex 0
$readparbyindex ERR 8 no more parameters
```

This command allows to read names and values of all STUPS parameters. Start the sequence by sending ReadParByIndex with value 1; this value indicates that STUPS should reset parameter index and send back values for the first parameter. Keep sending ReadParByIndex with value of 0 until STUPS returns parameter name is “ParamsEnd”. After this point the STUPS will return error code to ReadParByIndex with value 0. Sequence can be restarted at any time by sending ReadParByIndex with value 1.

Returned values contain parameter name as third word and value in consecutive word(s).

ServiceMode

```
servicemode 1
$servicemode OK
```

ServiceMode command is used to enter service mode. Some commands require this mode, otherwise they will return with error, for example Setpar or Reset commands. This is to prevent inadvertent change of behavior. Second word is 1 to enter service mode or 0 to exit service mode. Service mode does not change the behavior of STUPS in any way – it only allows to execute some commands.

PowerOff

```
poweroff time 40
$poweroff OK

poweroff current 0.3
$poweroff OK
```

Command is sent to request to turn off power to main and auxiliary outputs. Command is typically sent when system is running on backup and host saved all data and is ready to start operating system shutdown.

In the first version with “time” as second word, the STUPS will turn off output power after specified amount of time, in seconds. STUPS will then turn off power to its microcontroller.

The second version with “current” as second word, STUPS will turn off output power when first of following happens:

- Sum of currents iout + iou2 drops below specified value (in amps) – see block diagram in Read command
- 3 minutes pass after receipt of the command

This is typically used to detect when the host computer finished shutdown; as consumed current often drops to zero or to level significantly lower than normal host current consumption. For this to work properly, the specified current value must be safely lower than normal current consumption and safely higher than powered off current. Please take into consideration that STUPS current measurement error might be as big as ±50mA.

Hibernate

```
hibernate time 40 3600
$hibernate OK

hibernate current 0.3 86400
$hibernate OK
```

Hibernate command will perform the same power off sequence as the `PowerOff` command. After the output power is turned off, STUPS will deactivate most of internal circuitry to lower it’s current consumption and will enter a hibernate state. STUPS will automatically wake-up and turn on power to outputs to start-up the host after the time specified in the last word (time is in

seconds). Time to wake-up is measured from the moment when the command was received (as opposed to time from the end of shutdown).

System can be also woken-up before the time specified using start/stop push button or opto-isolated input if enabled by parameter `HibernateWakeUpEnable.`

Maximum valid value of hibernate time is 2000000 seconds. Second and Third word have the same meaning as in `PowerOff` command. Please see note about hibernate state in Overview paragraph.

Read

```
read vin
$read OK 12.312

read vin vout iout
$read OK 12.319 12.298 1.414
```

Command reads current value of one or more measurement points or other STUPS fields. One or more measurement point names or fields must be specified (point names are case sensitive). See Interrupt Message Field Names for list of supported fields.

GetStatus

```
getstatus
$getstatus OK Main 36
```

STUPS will respond to this request with the same information as in interrupt message – see paragraph interrupt messages.

ReadIO

```
readio
$readio OK vin 12.3 vout 12.2 iout 0.0 vout2 0.2 iout2 0.0 battery 100 di 1 do 1
```

STUPS will respond to this request with information about measurement points, battery charge percentage and current state of digital inputs and outputs. See Read command for description of measurement points.

Value returned after “battery” is battery charge status in percent. Hundred percent will provide backup runtime as indicated in Specifications paragraph. Fifty percent charge will provide half of runtime, etc.

Value after “di” is current state of digital inputs, 2 bit value specified in decimal.

Bit 0 is state of push button input (J1 pin 9):

- 0 when push button not actuated, pin is pulled up high through internal resistor
- 1 when push button is activated and pin is shorted through push button to ground

Bit 1 is state of opto-isolated input (J1 pin 10 and 11):

- 0 when input not active (current does not flow through opto-coupler)
- 1 when input is active, voltage between pin 10 and 11 is higher than 3 volts, current flows through opto coupler

Value after “do” is current state of digital outputs, 2 bit value specified in decimal:

Bit 0 is state of opto-isolated digital output 1 (pins 13 and 14):

Bit 1 is state of opto-isolated digital output 2 (pins 15 and 16):

- 0 when output is not in active state (for normal polarity: the coupler is in non-conductive state with high resistance)
- 1 when output is in active state (for normal polarity: the coupler is conducting and there is low resistance between pins - approx. 33 ohms)

SetDO1

SetDO2

```
setdo1 1
$setdo1 OK
```

Set digital output to specified value. Note that `DO1Mode/DO2Mode` parameters must have value of 0x0, otherwise this command will return with error.

Value can be one of:

- 0: set output to high-resistance state
- 1: set output to conducting state
- 2: generate 1Hz pulses on output
- 3: generate 4Hz pulses on output
- 4: generate 50Hz pulses on output

AuxOut

```
auxout on
auxout OK

auxout off
auxout OK
```

Turn auxiliary output on or off. Parameter `AuxOutputVoltage` must be set to valid value to be able to turn the output on.

Version

```
version
$version OK Serial:EAQ011 Product:STUPS-C10 Part:41321201 HW:41326005
HWVer:v1.00A SW:41328101 SWVer:v1.00A BootVer:v1.00A
```

STUPS will respond to this request with version information about hardware and firmware versions and other traceability information.

- Serial – device serial number
- Product – product family name
- Part – device part number
- HW – hardware part number
- HWVer – hardware version and revision
- SW – firmware part number
- SWVer – firmware version and revision
- BootVer – bootloader firmware version and revision

Parameters

See commands ReadPar, SetPar, StorePars, ReadParByIndex, ClearPAr, ClearAllPars, ReadParDefaultValue for more information related to STUPS parameters.

StupsViewCS sample application can use used to read and set parameters as well as save all parameters to file or upload them to another STUPS.

R – parameter is read only

RW – parameter is read/write

Parameter Name	R/W	Description
`Device`	R	Device name
`ProductName`	R	Product family name
`PartNumber`	R	Device part number
`HWNumber`	R	Hardwire part number
`HWVer`	R	Hardware version and revision
`SWNumber`	R	Firmware version and revision
`SerialNumber`	R	Device serial number
`CalID`	R	Calibration ID number
`MaxInputCurrent`	RW	Rating of power supply providing power to STUPS. Used to limit charger current so that STUPS does not overload power supply.
`BatteryReadyThrPercent`	RW	Percentage of battery charge required to turn on outputs when STUPS start-up.
`BatteryUsableEnergyWs`	R	STUPS usable energy measured during factory calibration. Used to calculate estimated runtime.
`SwithToBackupThrVolts`	RW	STUPS will switch to backup source when main input voltage Vin drops below this level
`PowerGoodHisteresisVolts`	R	STUPS will consider main input Vin valid when voltage is higher than `SwithToBackupThrVolts` +

		PowerGoodHisteresisVolts
PowerGoodTimeSec	RW	STUPS will attempt to switch back to main input source when input power is valid for this period of time
LoadRampSlowDownFactor	R	Timing factor used that specifies how fast will current increase when STUPS ramps up current from input to dummy load before switchover from backup to main source.
BackupOutputVoltage	RW	Main output voltage when running on backup source.
AuxOutputVoltage	RW	Auxiliary output voltage
UpdateMsgPeriodMs	RW	Period for sending interrupt messages.
UpdateMsgPeriodOnBackupMs	RW	Period for sending interrupt messages when STUPS is running on backup battery source.
UpdateMsgFields	RW	Colon separated list of field names sent by STUPS in interrupt message
NominalOutputPower	RW	STUPS measures output current to calculate estimate run time. If output current is too low then STUPS uses value of this parameter (in Watts) instead of measured output current to calculate run time.
RunningOnBatteryDelaySec	RW	Delay in seconds for notification of “running on battery” state for digital outputs.
DO1Mode	RW	Mode of operation of digital output 1.
DO2Mode	RW	Mode of operation of digital output 2.
HibernateWakeUpEnable	RW	Enables/disables wake up from hibernate state by start/stop push button or opto-isolated input
RunTimeThreshold	RW	Threshold value for runtime in seconds used by digital outputs logic
BatteryThreshold	RW	Threshold value for battery charge in percent used by digital outputs logic
ParamsEnd	R	Marker to indicate last parameter.

Device

Default value: STUPS-C

Device name.

ProductName

Default value: STUPS-C10

Product family name.

PartNumber

Device part number.

HWNumber

Hardwire part number.

HWVer

Hardware version and revision.

SWNumber

Firmware version and revision.

SerialNumber

Device serial number.

CalID

Calibration ID number.

MaxInputCurrent

Default value: 5.0

Rating of power supply providing power to STUPS in amperes. Used to limit charger current so that STUPS does not overload power supply while making charging time as short as possible.

For example if power supply has rating 15V/45W, then this value should be set to 3.0. If current flowing to outputs is 2.5 amps, then STUPS will limit charger current to 0.5 Amps. When the current to outputs changes then STUPS will adjust charger current accordingly.

This value should be adjusted for each STUPS application to match rating of used power source.

BatteryReadyThrPercent

Default value: 10.0 (in firmware v1.01 and lower), 0.0 (in firmware v1.02 and higher)

Upon STUPS startup the outputs are off. STUPS will turn on outputs only when battery charge in percent reaches this parameter value. Lower values mean that output will be turned on earlier. Higher values mean that STUPS will have more available run time in case that input power goes off shortly after STUPS powered up.

BatteryUsableEnergyWs

STUPS usable energy measured during factory calibration. Used to calculate estimated runtime. Value is specified in Joules (Watt seconds).

SwithToBackupThrVolts

Default value: 11.0

STUPS will switch to backup source when main input voltage Vin drops below this level.

PowerGoodHisteresisVolts

Default value: 0.5

STUPS will consider main input Vin valid when voltage is higher than `SwithToBackupThrVolts` + `PowerGoodHisteresisVolts.` With default settings, STUPS will start power good timer and eventually switch from backup to main when input voltage is above 11.5 Volts.

PowerGoodTimeSec

Default value: 1.0

STUPS will attempt to switch back to main input source when input power is valid for this period of time. With default setting STUPS will start switchover from backup to main sequence when input voltage is higher than 11.5 Volts for more than 1 second.

LoadRampSlowDownFactor

Default value: 1

Timing factor used that specifies how fast will current increase when STUPS ramps up current from input to dummy load before switchover from backup to main source. With default value, the input current ramp-up will start with 0 amps and will increase approximately 0.5 Amps per milliseconds until it reaches current output current. If input voltage is valid during the whole ramp up sequence, the STUPS will then switch from backup source to main input.

BackupOutputVoltage

Default value: 12.0

Main output voltage when running on backup source. Valid values are 11.0 to 13.0 volts. STUPS microcontroller reset or power cycle is required for parameter change to take effect.

AuxOutputVoltage

Default value: 0.0

Auxiliary output voltage. This must be set by user to valid value, otherwise the auxiliary output will not be enabled. Valid values are 3.0 to 5.5 Volts. STUPS microcontroller reset or power cycle is required for parameter change to take effect.

UpdateMsgPeriodMs

Default value: 3000

Period for sending interrupt messages in milliseconds. If this parameter is set to special value of 0, then interrupt messages are sent only when STUPS power status changes. . If this parameter is set to special value of -1, then interrupt messages are not sent at all (application should then use `Read` or `GetStatus` command to query status of STUPS periodically). When STUPS is running on backup battery source then the value of parameter UpdateMsgPeriodOnBackupMs is used instead. See interrupt messages for more information.

UpdateMsgPeriodOnBackupMs

Default value: 200

This parameter has the same meaning as UpdateMsgPeriodMs, this value is used when STUPS is running on backup battery source. In typical application it is often desired to get status updates faster when running on battery so that host has up-to-date information on battery percentage and remaining runtime and shutdown decision is not delayed.

UpdateMsgFields

Default value: status:runtime

Colon separated list of field names sent by STUPS in interrupt message. Please see interrupt message section for more information.

NominalOutputPower

Default value: 25.0

STUPS measures output current to calculate estimate run time. If output current is too low, then STUPS uses value of this parameter (in Watts) instead of measured output current to calculate run time.

RunningOnBatteryDelaySec

Default value: 3.1

Digital outputs can be configured to indicate state when STUPS is running on battery backup source. It is possible to apply delay so that short power blips would be ignored. This parameter specified the delay. With default setting, if STUPS can successfully switch to main power in less then 3.1 seconds from the initial time when power went off then STUPS will not report “on-battery-status”. Please not that STUPS will need to detect valid input power levels for time specified by `PowerGoodTimeSec` time before it attempts to switch back to main input, so with default values the power blip must be shorter then 2.1 seconds not to report it.

DO1Mode

DO2Mode

Default value: 0x0

Parameters specify mode of operation of digital outputs. Each is composed of multiple fields. Values can be set using decimal or hexadecimal format (e.g. 0x0E00). STUPS will report it in hexadecimal format.

Bits 8 to 15, if set, specify conditions that will make the digital output active. More bits can be set at the same time:

- Bit 8 (mask 0x0100): if set then output will be active if STUPS is running on battery backup source
- Bit 9 (mask 0x0200): if set then output will be active if STUPS is running on battery backup source for time longer then specified by `RunningOnBatteryDelaySec` parameter
- Bit 10 (mask 0x0400): if set then output will be active if start/stop pushbutton is pressed
- Bit 11 (mask 0x0800): if set then output will be active if opto-isolated input is active
- Bit 12 (mask 0x1000): if set then output will be active if estimated runtime in seconds is below value specified by parameter `RunTimeThreshold`
- Bit 13 (mask 0x2000): if set then output will be active if estimated runtime in seconds is below value specified by parameter `RunTimeThreshold` and STUPS is running on backup battery source
- Bit 14 (mask 0x4000): if set then output will be active if battery charge in percentage is below value specified by parameter `BatteryThreshold`
- Bit 15 (mask 0x8000): if set then output will be active if battery charge in percentage is below value specified by parameter `BatteryThreshold` and STUPS is running on backup battery source

If no bits 8-15 are set then output initially starts in high-resistance state and changes state only when host sends `SetDO` command.

Bits 0-7 compose a 7 bit field that specifies output state when output is in active state:

- 0: positive logic, output opto-isolator is conducting if output is in active state
- 1: negative logic, output opto-isolator is conducting if output is in not in active state
- 2: 1Hz pulses if active is active, high-resistance state if not active
- 3: 4Hz pulses if active is active, high-resistance state if not active
- 4: 50Hz pulses if active is active, high-resistance state if not active
- Other values: reserved

To control the output from host set the value to 0x00. To use the output to notify the host that STUP is running on battery (delayed) or push button is pressed or opto-isolated input is active, set the value to 0x0E00. These are condition that typically require system shutdown (either the

power is off or user wants to shutdown the system). If output state should be negated then change the value to 0x0E01.

HibernateWakeUpEnable

Default value: 0x0

Enables/disables wake up from hibernate state by start/stop push button or opto-isolated input. Values can be set using decimal or hexadecimal format (e.g. 0x1F). STUPS will report it in hexadecimal format.

- Bit 0: if set then start/stop button will wake-up system when button is pressed
- Bit 1: if set then opto-isolated input will wake-up system if it is active

Multiple bits can be set. If no bits are set (value is 0x0) then the system will wake up when specified hibernate time has passed.

RunTimeThreshold

Default value: 30 (in seconds)

Digital outputs can be configured to indicate that estimated runtime in seconds dropped below certain level. This parameter specified the level in seconds. See `DO1Mode` and `DO2Mode` parameters.

BatteryThreshold

Default value: 30 (in percent)

Digital outputs can be configured to indicate that batter charge in percent dropped below certain level. This parameter specified the level in percent. See `DO1Mode` and `DO2Mode` parameters.

ParamsEnd

Marker to indicate last parameter.

StupsView

StupsView application can be use as an interface to STUPS on windows operating system (XP and higher) for following purpose:

- Help with integration and evaluation of STUPS in your system
- Setup of STUPS parameters
- To copy parameters from one STUPS to another
- As a production software that will initiate windows shutdown when configured conditions occur
- As an example of C# code for communication with STUPS

Application is written in C# and source code is included. Application can be configured to start automatically when OS starts and run as an icon in the system tray (notification area).

Installation

Two methods of installation are possible:

1. Copy StupsVew.exe from Exe subfolder to any location on host hard-drive, for example to newly created folder under Program Files. No other installation steps are required.
2. or, Run Setup.exe in Setup subfolder. This will create entry for STUPS in Windows Start menu

Once installed, start the application and configure settings on the main form. All settings are automatically saved and will be used the next time the application runs.

To start StupsView when Windows starts

Create shortcut to StupsView.exe file. Copy shortcut to Startup folder under Start/All Programs menu.

To run StupsView as an icon in system tray (notification area)

Create shortcut to StupsView.exe file. Add following arguments:

-startminimized –systray

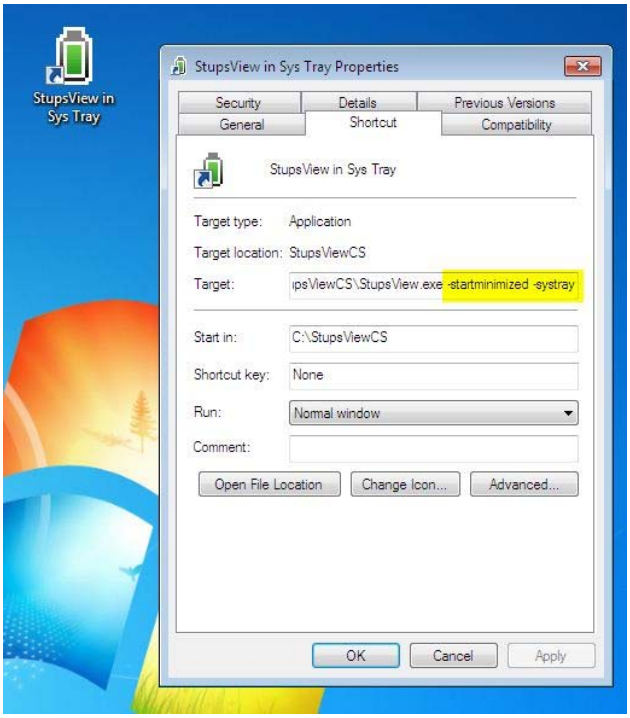
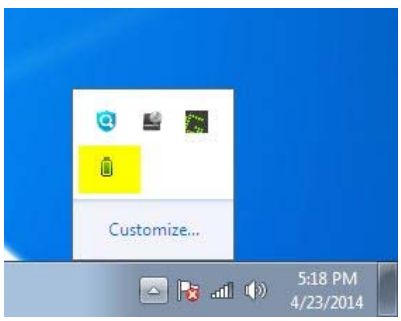
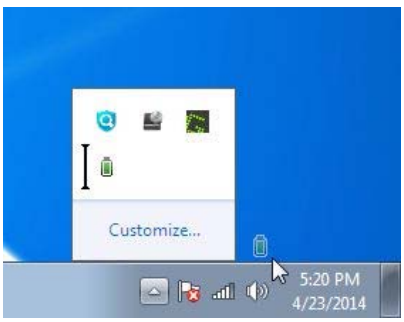


Figure 8 – modify shortcut to include StupsView arguments

First argument will make the application start minimized. Second specifies that when the application is minimized it should be placed to the system tray. Please note that when you run this shortcut you will not longer see the application main form, only a new icon will be added to the menu near the bottom-right area of the desktop. Open the menu and drag the icon to the tray.



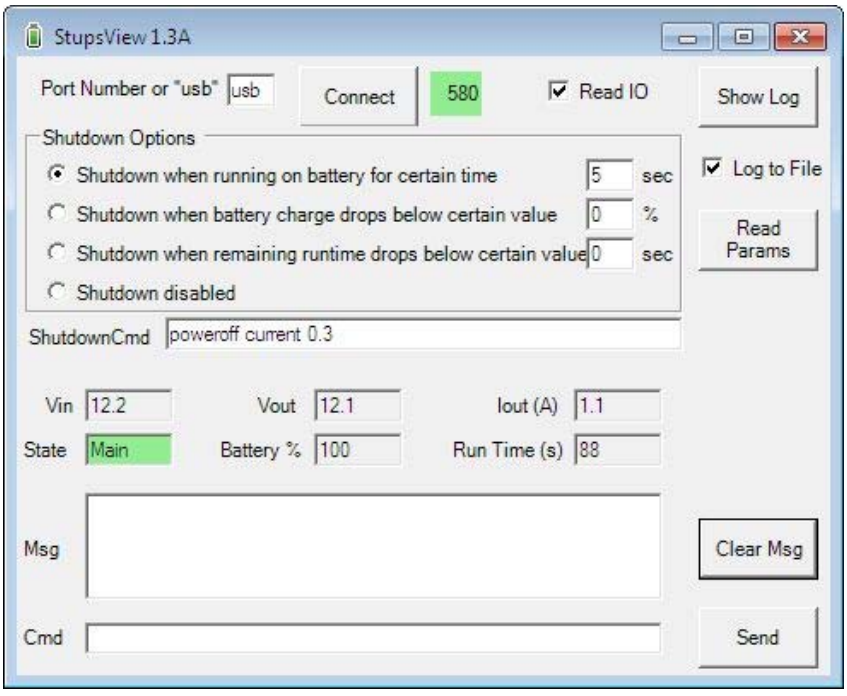


When the StupsView runs as an icon in the system tray, double click the icon to open the main form. Click minimize button on main form to collapse the application back to the icon in system tray.

Icon will change color depending on STUPS status:

	Normal State, STUPS running on main input supply.
 flashing	STUPS running on battery, battery charge higher then 25%.
 flashing	STUPS running on battery, battery charge less then 25%.
 flashing	STUPS output is off.
	Communication or other Error.

Main Form



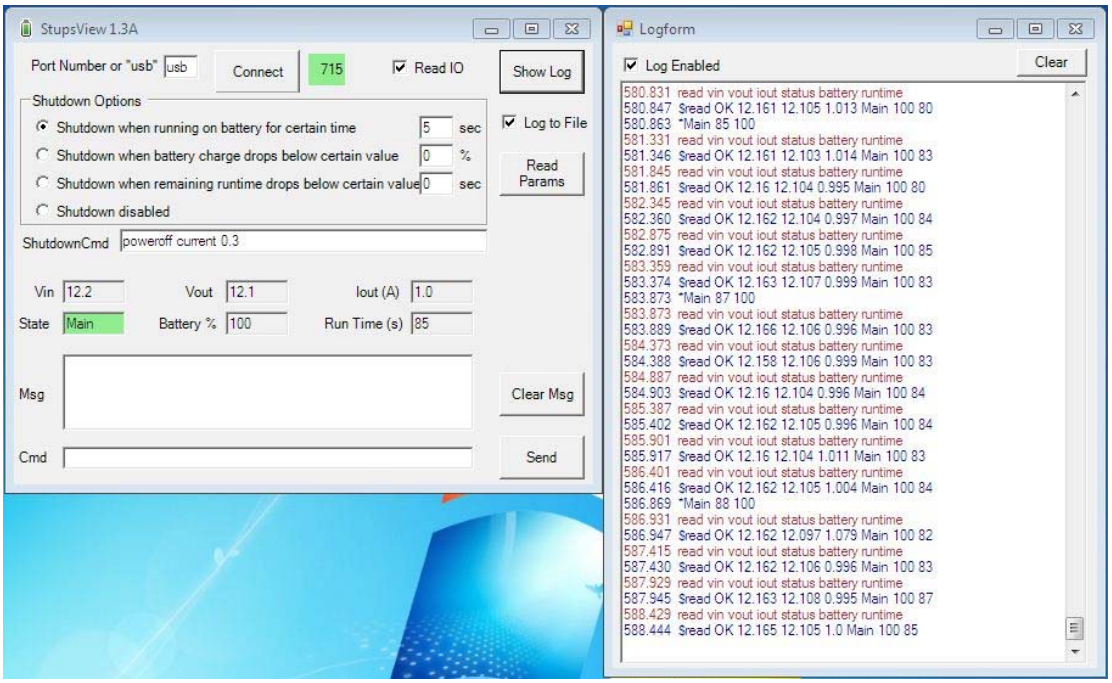
All settings are automatically saved and will be used the next time the application starts.

If RS-232 is used for communication with STUPS then enter comm. port number to Port Number text box and press Connect button. If USB is used enter text “usb” instead. See section USB driver Installation for more details on USB.

Once the Connect button is pressed, the application will attempt to communicate with STUPS. If connection is successful, the background of text box on the right of Connect button with turn green and number will start to increment whenever a new message is received.

If Read IO checkbox is checked then the application will send periodic requests to read values of voltages, currents and status.

Show Log button opens a form that shows all sent and received messages.



To send command to STUPS, type it to the Cmd text box near the bottom of the Main Form and press Send. When Cmd text box has focus, pressing up and down arrow will act as a history function and will show previously typed commands.

Shutdown Settings

These shutdown options are supported:

- Shutdown will start when STUPS runs on battery for fixed amount of time. This is typically used to ride through short power blips in input power.
- Shutdown will start when STUPS is running on battery and battery charge drops below specified percentage.

- Shutdown will start when STUPS is running on battery and estimated remaining runtime drops below certain value. Estimated runtime is continuously calculated by STUPS based on current flowing to powered device. Set time in seconds to value safely longer then time required for your system to shut down completely.
- Shutdown will not be started. Typically used during system setup and testing.

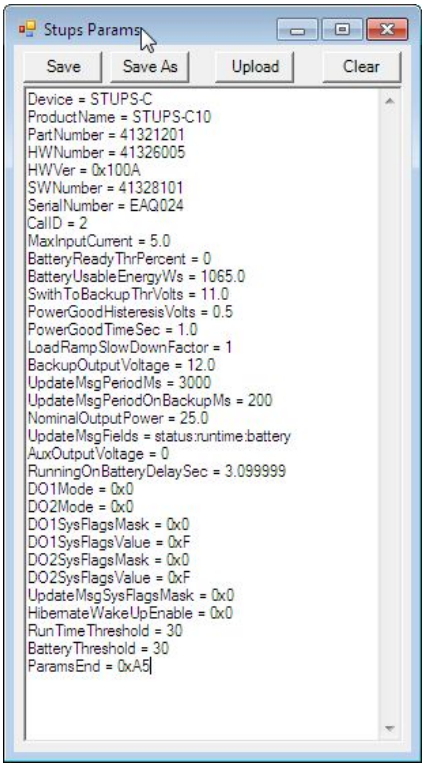
Command specified in ShutdownCmd will be sent to STUPS just before the shutdown is initiated. This is required to notify STUPS that system is being shut-down and STUPS can turn off power to the device and itself when shutdown is complete. Turning off power output is needed so that STUPS can start again itself and power device immediately when input power is restored.

Please see more details under PowerOff command.

If Log to File checkbox is checked then all messages are logged to log file in the same folder as the StupsView executable. File size is limited and will not grow larger then 5MB (oldest messages are removed).

Parameters

Click Read Params button on main form to read all parameters. This will open Parameters form:



Press Save button to save parameter values to file in the same folder as executable. Press Save As to specify file where parameter values should be stored.

Press Upload button to set all writable STUPS parameters to values in the specified file. Only user writable parameters will be set (so for example, serial number will not be overwritten).

Once the desired STUPS parameter values are determined it is possible to copy them to another unit(s):

1. Save them to file first from the “source” STUPS
2. Disconnect “source” STUPS from the host
3. Connect “destinaton” STUPS to host, verify that the connection is OK
4. Press Upload button to set parameters
5. Repeat from step 3 as needed

StupsViewVB6

Application provides similar features as StupsView. It is written in Visual Basic 6. Source code is included.